Instituto Federal de Educação, Ciência e Tecnologia de São Paulo

***Campus* São João da Boa Vista**

Curso de Engenharia de Controle e Automação

Laboratório de Programação de Microcontroladores

Controle PID de motor CC com FPGA

Camila da Silva Bertazzi

Emerson Castelhano Voltarelli

São João da Boa Vista

Dezembro de 2019

Camila da Silva Bertazzi

Emerson Castelhano Voltarelli

Controle PID de motor CC com FPGA

Trabalho apresentado na disciplina de Laboratório de Programação de Microcontroladores, do curso de Engenharia de Controle e Automação do Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, como requisito para conclusão da referida disciplina.

Professor: Daniel Espanhol Razera e Muriell de Rodrigues e Freire.

São João da Boa Vista

Dezembro de 2019

SUMÁRIO

INTRODUÇÃO

MOTOR CC

Um motor CC é um motor alimentado por corrente contínua (CC), essa alimentação pode ser proveniente de uma bateria ou qualquer outro tipo de alimentação CC. A sua comutação (troca de energia entre rotor e estator) pode ser através de escovas ou sem escovas e sua velocidade pode ser controlado apenas variando a sua tensão. O motor de corrente continua é composto de duas estruturas magnéticas: estator (enrolamento de campo ou imã permanente) e o rotor (enrolamento de armadura). O esquema elétrico de um motor CC pode ser visto na Figura 1.

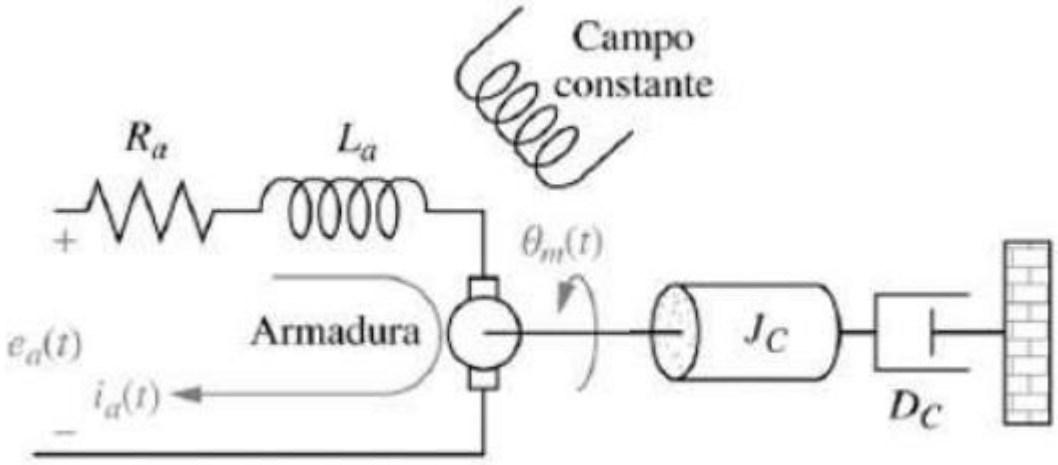

Figura 1- Esquema elétrico de um motor CC.

O estator é composto de uma estrutura ferromagnética com polos salientes aos quais são enroladas as bobinas que formam o campo, ou de um imã permanente. Já o rotor é um eletroímã constituído de um núcleo de ferro com enrolamentos em sua superfície que são alimentados por um sistema mecânico de comutação. Esse sistema é formado por um comutador, solidário ao eixo do rotor, que possui uma superfície cilíndrica com diversas lâminas as quais são conectados os enrolamentos do rotor; e por escovas fixas, que exercem pressão sobre o comutador e que são ligadas aos terminais de alimentação. O propósito do comutador é o de inverter a corrente na fase de rotação apropriada de forma a que o conjugado desenvolvido seja sempre na mesma direção.

Dependendo de sua aplicação, os acionamentos em corrente contínua são geralmente os que apresentam os maiores benefícios, também em termos de confiabilidade, operação amigável e dinâmica de controle. Por outro lado, esse tipo de acionamento apresenta algumas desvantagens.

Vantagens

• Operação em 4 quadrantes com custos relativamente mais baixos;

• Ciclo contínuo mesmo em baixas rotações;

• Alto torque na partida e em baixas rotações;

• Ampla variação de velocidade;

• Facilidade em controlar a velocidade;

• Os conversores CA/CC requerem menos espaço;

• Confiabilidade

• Flexibilidade (vários tipos de excitação);

• Relativa simplicidade dos modernos conversores CA/CC.

Desvantagens

• Os motores de corrente contínua são maiores e mais caros que os motores de indução, para uma mesma potência;

• Maior necessidade de manutenção (devido aos comutadores);

• Arcos e faíscas devido à comutação de corrente por elemento mecânico (não pode ser aplicado em ambientes perigosos);

• Tensão entre lâminas não pode exceder 20V, ou seja, não podem ser alimentados com tensão superior a 900V, enquanto que motores de corrente alternada podem ter milhares de volts aplicados aos seus terminais;

• Necessidade de medidas especiais de partida, mesmo em máquinas pequenas.

PID

PID é o algoritmo de controle mais utilizado na indústria devido ao seu desempenho robusto em uma ampla gama de condições de funcionamento além da sua simplicidade funcional, permitindo aos engenheiros operá-los de uma forma simples e direta. O algoritmo PID é composto por três coeficientes: proporcional, integral e derivativo, que são variados para obter a resposta ideal.

A ideia básica por trás de um controlador PID é ler um sensor, calcular a resposta de saída do atuador através do cálculo proporcional, integral e derivativo e então somar os três componentes para calcular a saída, como pode ser visto na Figura 2.

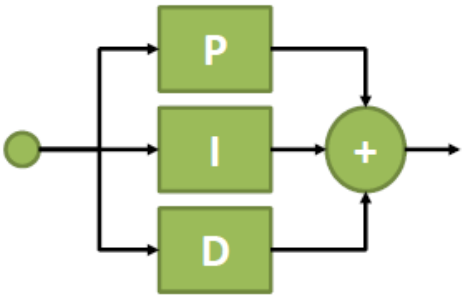

Figura 2- Cálculo do PID.

Em um sistema de controle típico, a variável do processo é o parâmetro do sistema que precisa ser controlado como temperatura (ºC), pressão (psi) ou fluido (litros/minuto). Um sensor é usado para medir a variável de processo e fornecer um feedback para o sistema de controle. O set point é o valor desejado ou comando para a variável de processo. A diferença entre a variável de processo e o set point é usada pelo algoritmo do sistema de controle (compensador) para determinar a saída desejada do atuador, que por sua vez, irá acionar o sistema (planta). Isso é chamado de um sistema de controle em malha fechada, porque o processo de leitura dos sensores para fornecer feedback constante e o cálculo para definir a saída desejada do atuador se repete continuamente a uma taxa fixa, como pode ser visto na Figura 3.

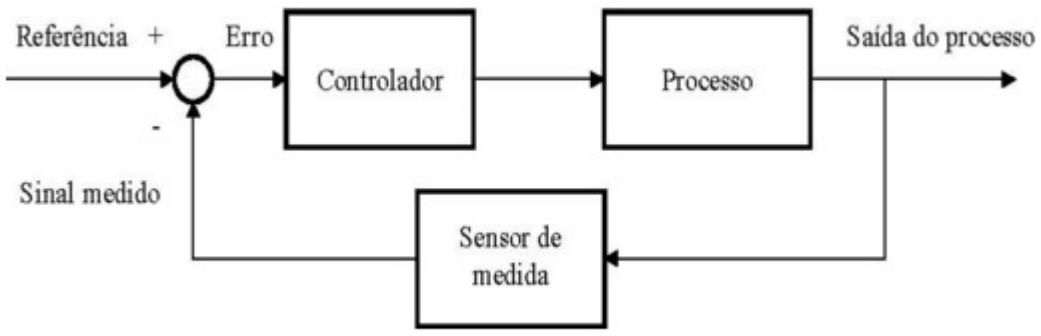

Figura 3- Sistema PID em malha fechada.

O processo de projeto de controle começa pelos requisitos de desempenho. O controle de desempenho do sistema geralmente é medido pela aplicação de uma função degrau definida como comando set point, em seguida é medida a resposta da variável de processo. A resposta é quantificada pelas características da onda de resposta, como pode ser visto na Figura 4.

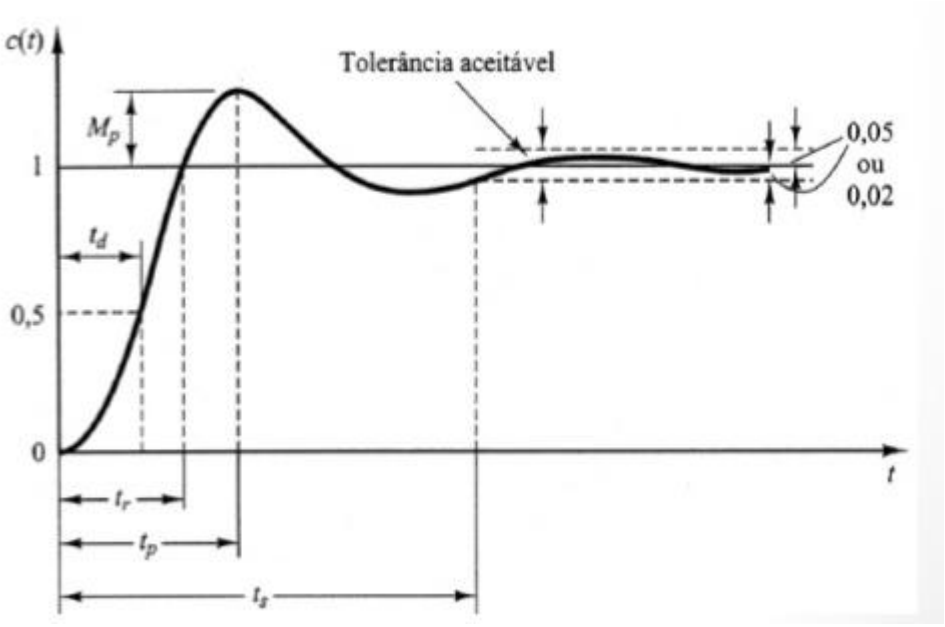

Figura 4- Forma de onda da resposta de um PID.

O tempo de subida (Tr) é o tempo que o sistema leva para ir de 10% a 90% do estado estacionário, ou valor final. O Overshoot (Tp) é o valor que a variável de processo ultrapassa o valor final, expresso como uma porcentagem do valor final. Settling time (Ts) é o tempo necessário para a variável do processo chegar dentro de uma determinada porcentagem (normalmente 5%) do valor final. Steady-State de erro é a diferença final entre as variáveis do processo e o set point.

A componente proporcional depende apenas da diferença entre o ponto de ajuste e a variável de processo. Esta diferença é referida como o termo de erro. O ganho proporcional (Kp) determina a taxa de resposta de saída para o sinal de erro. Em geral, aumentando o ganho proporcional irá aumentar a velocidade da resposta do sistema de controle. No entanto, se o ganho proporcional é muito grande, a variável de processo começará a oscilar. Se Kd é aumentado ainda mais, as oscilações ficarão maior e o sistema ficará instável e poderá oscilar até mesmo fora de controle.

A componente integral soma o termo de erro ao longo do tempo. O resultado é que mesmo um pequeno erro fará com que a componente integral aumente lentamente. A resposta integral irá aumentando ao longo do tempo a menos que o erro seja zero, portanto, o efeito é o de conduzir o erro de estado estacionário para zero. O Steady-State de erro é a diferença final entre as variáveis do processo e do set point.

A componente derivada faz com que a saída diminua se a variável de processo está aumentando rapidamente. A derivada de resposta é proporcional à taxa de variação da variável de processo. Aumentar o parâmetro do tempo derivativo (Td) fará com que o sistema de controle reaja mais fortemente à mudanças no parâmetro de erro aumentando a velocidade da resposta global de controle do sistema. Na prática, a maioria dos sistemas de controle utilizam o tempo derivativo (Td) muito pequeno, pois a derivada de resposta é muito sensível ao ruído no sinal da variável de processo. Se o sinal de feedback do sensor é ruidoso ou se a taxa de malha de controle é muito lenta, a derivada de resposta pode tornar o sistema de controle instável.

FPGA

FPGA (Field Programmable Gate Array) é um dispositivo ou hardware que pode ser reconfigurado ou programado pelo usuário. Trata-se de um circuito integrado que vem com um grande conjunto de blocos que, através de programação podem ser configurados de tal forma que o conjunto exerça um determinado número de funções específicas. Trata-se de um chip que pode ser programado pelo usuário para fazer o que se deseja, diferentemente de um microcontrolador que vem com os blocos prontos numa estrutura definida e que pode ser programado para determinadas funções, o FPGA vem com os blocos desligados, sendo então interconectados por programação pelo usuário para fazer o que ele deseja. O microcontrolador vem com os blocos de funções fixos, enquanto que no FPGA o usuário os cria. A FPGA utilizada é o Cyclone II da Altera, como pode ser visto na Figura 5.

Figura 5- FPGA utilizada.

Um FPGA típico conta com três tipos de blocos. O primeiro bloco pode ser visto na Figura 6.

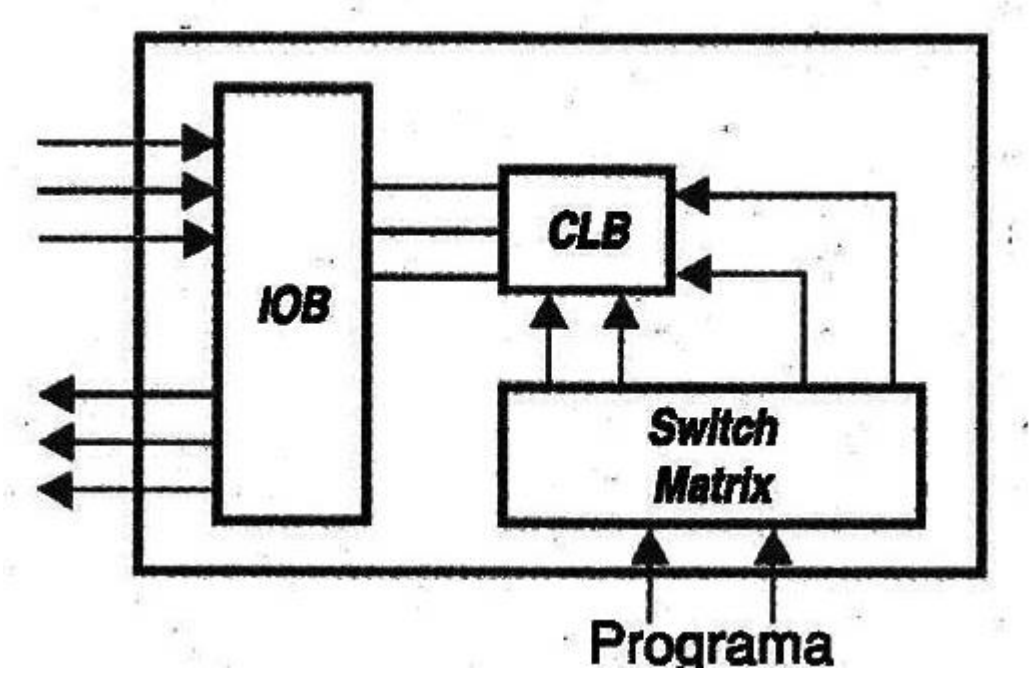

Figura 6- Primeiro bloco FPGA.

O bloco CLB (Configuration Logical Blocos) trata-se do coração do FPGA, consistindo em circuitos formados pela reunião de flip-flops além de lógica combinacional. Com eles é possível criar funções lógicas. Assim, nos FPGAs em que os grãos são grandes, os blocos lógicos podem ser complexos contendo pequenos microprocessadores, memórias, unidades lógicas e aritméticas, etc. Nos FPGAs com granulação média, já encontramos blocos lógicos menos complexos como funções lógicas de média complexidade. Finalmente, nos FPGAs de granulação fina ou grãos pequenos, encontramos funções simples em cada bloco como multiplexadores, flip-flops, etc.

Para cada empresa encontramos arquiteturas típicas com diversos tipos de granulação que influem na velocidade do dispositivo. Podemos então em primeiro lugar citar os FPGAs da Altera que utilizam LUTs ou Look-Up Table em que temos células de armazenamento que são empregadas para implementar as funções. Nesta arquitetura, cada bloco armazena apenas um bit, ou seja, um nível lógico 0 ou 1, conforme mostra um bloco típico na Figura 7.

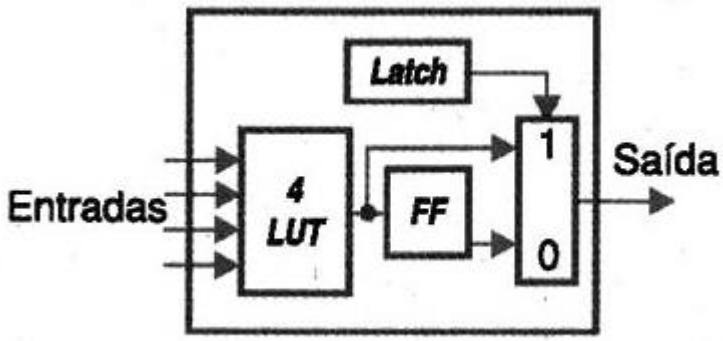

Figura 7- Bloco Look-Up Table.

Neste bloco temos diversas entradas que entram numa matriz programada que vai determinar que tipo de função o bloco executa. Cria-se uma tabela verdade para o bloco, conforme a vontade do usuário. A saída deste bloco é então armazenada num flip-flop e utilizada quando solicitada pelo clock.

MATERIAIS UTILIZADOS

- Kit motor CC;
- FPGA;
- Osciloscópio;
- 2 resistores de 10 KΩ;
- 1 resistor de 100 KΩ;
- 1 resistor de 330 Ω;
- 1 resistor de 560 Ω;
- 1 diodo 1N4007;
- 1 TIP 122;
- 1 opto acoplador 4n25;
- 1 LM358;
- 1 BC547.

PROCEDIMENTO EXPERIMENTAL

Primeiramente, para obter a função de transferência do motor CC, mediu-se a resistência de armadura (Ra), entre os terminais do motor. Para obtenção da constante elétrica (Ke) e do amortecimento viscoso (B), aplicou-se diferentes tensões continuas na entrada (Va) e mediu-se as respectivas correntes de armadura (Ia) e a velocidade de saída no eixo (ω), para assim poder-se calcular os valores de Ke e B, a partir das Equações 1 e 2 e de suas retas médias.

O momento de inercia (J) foi obtido medindo a constante de tempo mecânica (τm), através do ensaio run down test, onde nesse ensaio, alimenta-se o motor a vazio, marca-se a velocidade em regime estacionário (ωo) e, em seguida, remove-se a tensão de armadura e mede-se o tempo para a velocidade chegar a 37% da velocidade ωo, com a constante de tempo e o amortecimento viscoso, é possível calcular a inercia do eixo pela Equação 3.

Para a indutância de armadura do motor (La), aplicou-se uma corrente alternada de 3V nos terminais, mediu-se a corrente e com isso, calculou-se a impedância do circuito, podendo então obter-se a reatância indutiva e finalmente, pela Equação 4, a indutância de armadura (La). Com todos os parâmetros necessários, pode-se obter a função de transferência em malha aberta do respectivo motor CC, dado pela Equação 5.

Primeiramente, montou-se o motor CC, como pode ser visto na Figura 8, para a obtenção do modelo do circuito através do método empírico, encontrando assim, os coeficientes da equação característica de um motor CC.

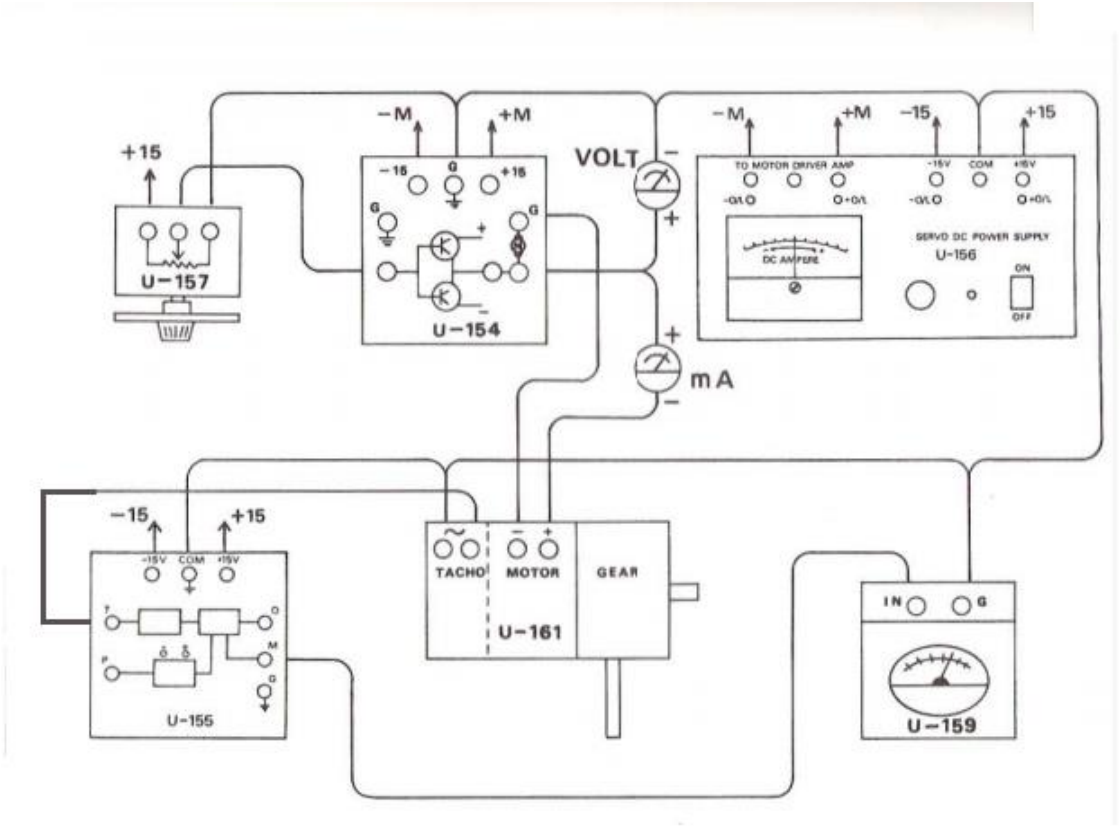

Figura 8- Montagem do motor CC.

Após a obtenção da equação do motor, utilizou-se o programa Matlab, para projetar um controlador juntamente com uma taxa de amostragem otimizada para controle digital.

Para implementar o PID no motor, retirou-se da montagem o potenciômetro (U-157) e o conversor de potência (U-154), mantendo apenas o sistema de leitura de velocidade para conferir os valores lidos. Introduziu-se um circuito de saída PI, Figura 9, que é um drive de potência tradicional, onde um opto acoplador isola o sistema de controle da carga que possui um transistor de chaveamento para o PWM e um diodo de rola livre devido ao uso da carga indutiva.

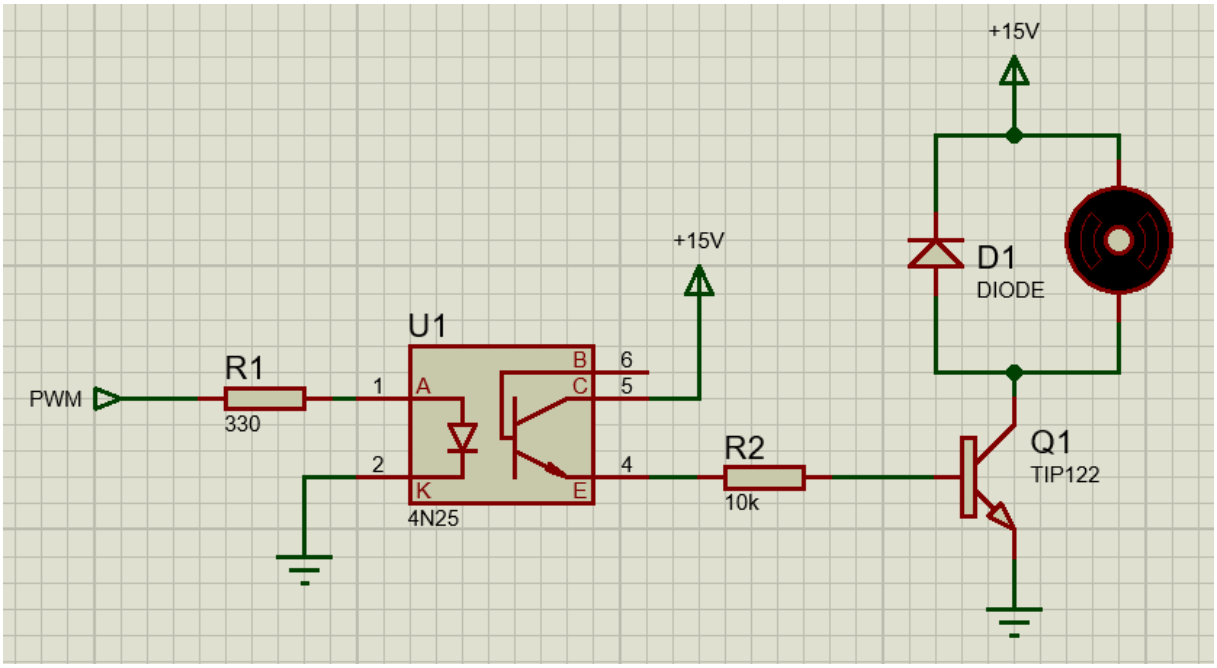

Figura 9- Circuito de saída PI.

Introduziu-se também um circuito na entrada do tacômetro, Figura 10, onde, a partir do sinal senoidal gerado pelo tacômetro, passa por um Amp Op comparador, onde, quando o tacômetro muda para o semi ciclo positivo, ele emite um sinal em nível alto enquanto no semi ciclo negativo o sinal é baixo. O transistor é utilizado para condicionar o sinal para 3,3 V, uma vez que a FPGA reconhece apenas 3,3 V.

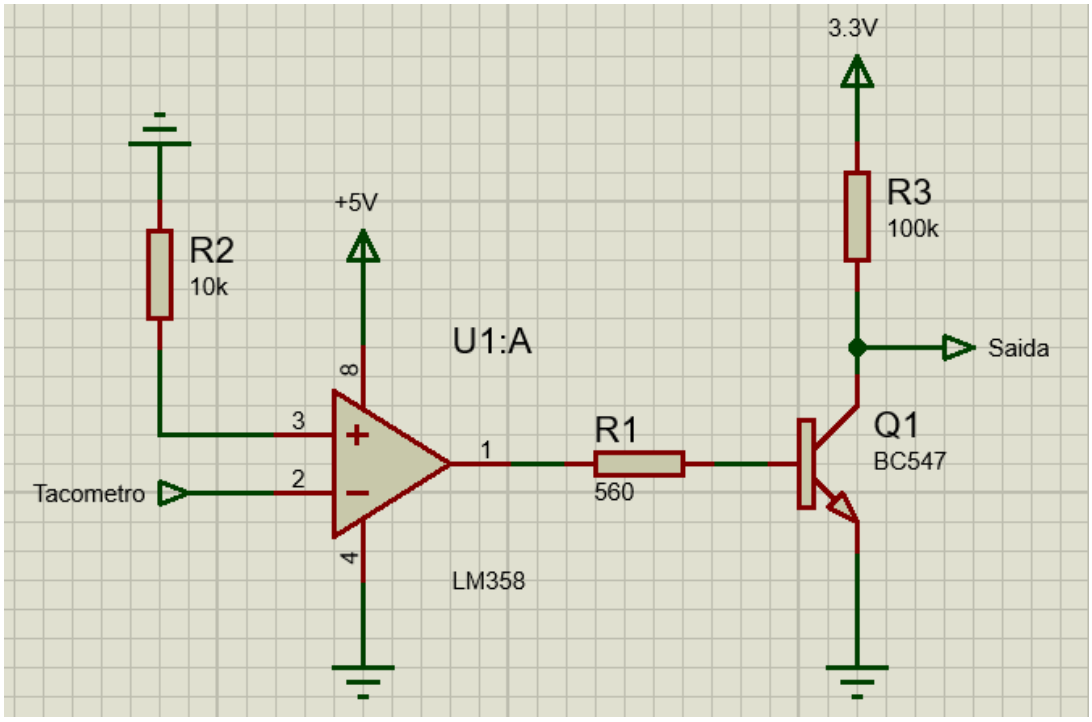

Figura 10- Circuito tacômetro.

RESULTADOS OBTIDOS

Para a resistência de armadura mediu-se o valor de Ra = 4Ω. Após isso, para os diferentes valores de tensão de entrada (Va), mediu-se a corrente de armadura (Ia) e a velocidade de rotação (ω) e compilou-se na Tabela 1 abaixo.

Tabela 1 - Valores de tensão de entrada, corrente de armadura e velocidade de rotação.

Va (V)	Ia (I)	ω (rpm)
0	0	0
1	0,0652	400
2,07	0,0823	800
3,06	0,0996	1200
4,16	0,1144	1700
5,04	0,1254	2100
6,06	0,1341	2600
7,07	0,1413	3000
8	0,15	3400
9,04	0,1582	3900

Assim, pelas Equações 1 e 2 abaixo, pode-se calcular os valores da constante elétrica (Ke) e do amortecimento viscoso (B) do circuito.

$$Ke = \frac{V - Ia.Ra}{\omega} \quad (1)$$

$$B = \frac{(V - Ia.Ra).Ia}{\omega^2} \quad (2)$$

Compilando os valores da Tabela 1 com as Equações 1 e 2, pode-se montar a Tabela 2 abaixo, complementando os valores de Ke e B.

Tabela 2 - Valores complementares da constante elétrica e do amortecimento viscoso.

Va (V)	Ia (A)	ω (rpm)	Ke	B (N.m.s/rad)
0	0	0	0,002134	$1{,}53327.10^{-7}$
1	0,0652	400	0,001848	$3{,}01224.10^{-7}$
2,07	0,0823	800	0,002176	$2{,}23856.10^{-7}$
3,06	0,0996	1200	0,002218	$1{,}84094.10^{-7}$
4,16	0,1144	1700	0,002178	$1{,}46559.10^{-7}$
5,04	0,1254	2100	0,002161	$1{,}29051.10^{-7}$
6,06	0,1341	2600	0,002124	$1{,}09573.10^{-7}$
7,07	0,1413	3000	0,002168	$1{,}02125.10^{-7}$
8	0,15	3400	0,002176	$9{,}60208.10^{-8}$
9,04	0,1582	3900	0,002156	$8{,}74437.10^{-8}$

Utilizando a reta média para os respectivos valores, obteve-se o valor da constante Ke = Kt = K = 0,002134 e de B = 1,533.10^{-7} N.m.s/rad. Com o valor de B, através da Equação 3, pode-se calcular o valor da inercia (J) do eixo do motor, sendo τ a constante de tempo mecânica do sistema.

$$J = \tau_m . B \quad (3)$$

$$J = 0.26 * 1{,}533 * 10^{-7} = 3{,}9865 * 10^{-8} \frac{N.m.s^2}{rad}$$

Necessitando apenas o valor da indutância de armadura (La), com a tensão de entrada em 3,03 V e a frequência a 60 Hz, obteve-se uma corrente de armadura Ia = 0,76A, que resulta em uma impedância Z = 5,615 Ω, assim, a reatância indutiva do circuito é X_L = Z - R = 5,615 – 4 = 1,615 Ω. Com o valor da reatância, pode-se calcular a indutância do circuito, pela Equação 4 abaixo.

$$La = \frac{X_L}{2\pi f} \quad (4)$$

$$La = \frac{5{,}615}{2 * \pi * 60} = 4{,}283\ mH$$

Com todos os valores necessários para o modelo do sistema, junto a Equação 5 a seguir, obteve-se a função de transferência em malha aberta do motor.

$$G = \frac{K}{(La.s + Ra)(J.s + B) + K^2} \quad (5)$$

$$G = \frac{0{,}002134}{(0{,}004283s + 4)(3{,}9865 * 10^{-8}s + 1{,}533 * 10^{-7}) + 0{,}002134^2}$$

$$G = \frac{1{,}25 * 10^7}{(s + 904{,}3)(s + 33{,}47)}$$

Pela ferramenta pidtool() do matlab, obteve-se a resposta representada pela Figura 11 abaixo.

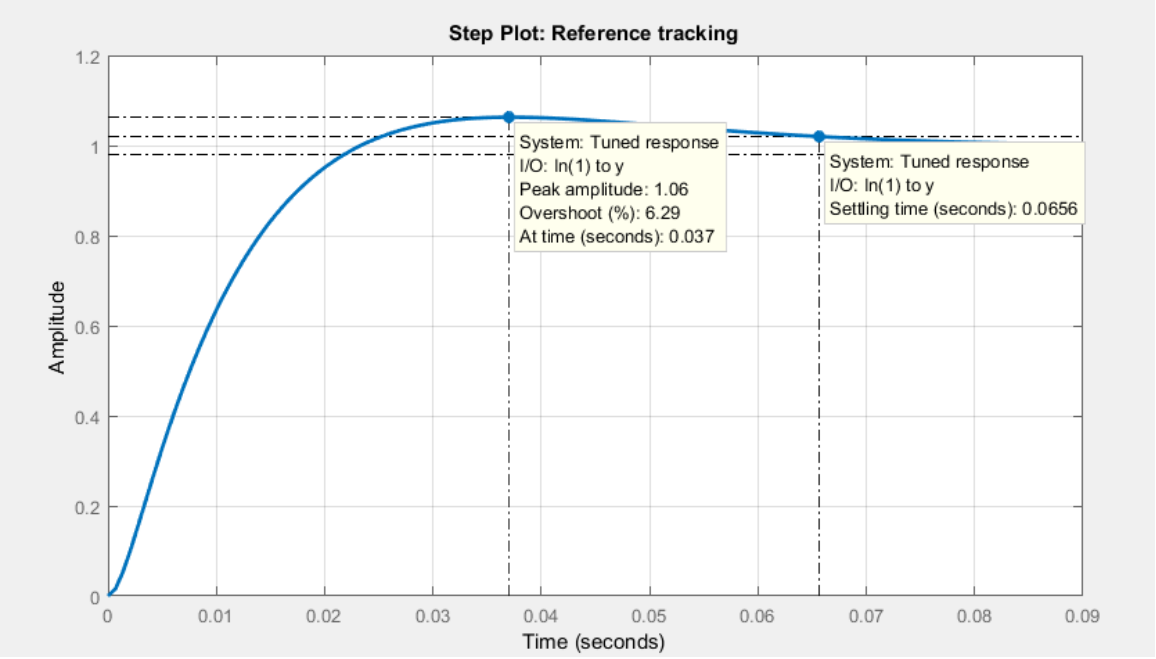

Figura 11- Saída com controle PI.

Os valores de Kp e Ki obtidos foram: Kp = 0.006 e Ki = 0.215. Assim, para obter o tempo de amostragem para o controlador digital, comparou-se o método analítico com o da boa prática, sendo que, para o método analítico, obteve-se o diagrama de bode da Figura 12 a seguir, para a função em malha aberta com controlador.

```
margin(C*G)
```

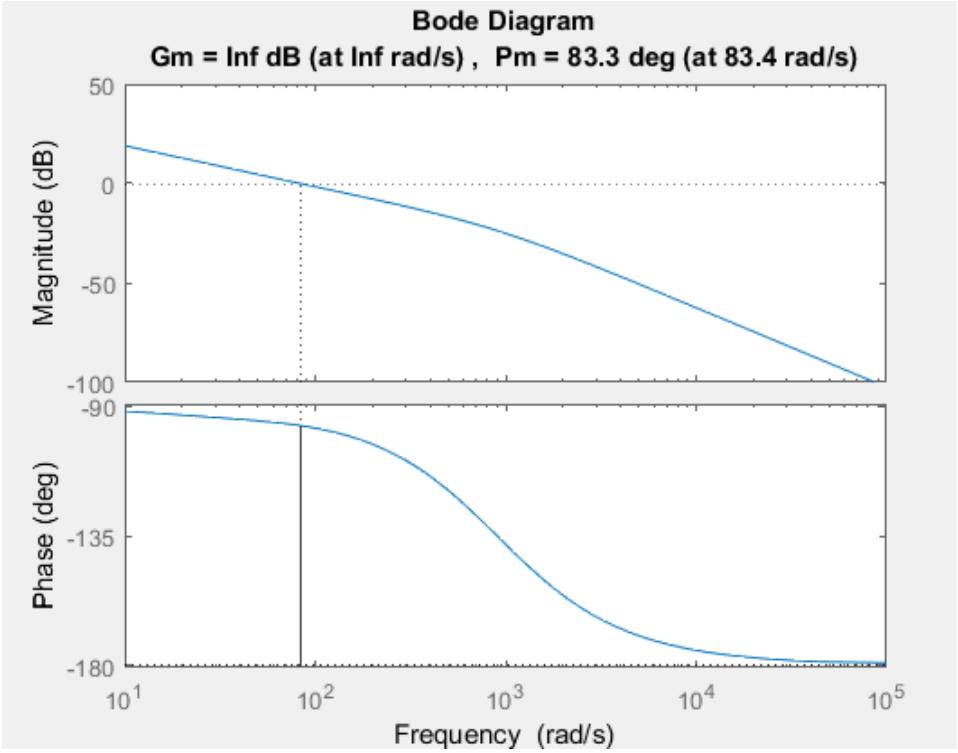

Figura 12- Diagrama de bode.

Como a frequência na margem de fase é de φ_m=83,4 rad/s, os valores para o tempo de amostragem possíveis podem ser determinados pela Equação 6.

$$\frac{0{,}15}{\varphi_m} < T < \frac{0{,}5}{\varphi_m} \quad (6)$$

$$\frac{0{,}15}{83{,}4} < T < \frac{0{,}5}{83{,}4}$$

$$0{,}0018 < T < 0{,}0060$$

Para o método prático, pode-se obter a maior frequência natural do sistema, como pode ser visto na Figura 13.

```
[wn,z] = damp(G)
  wn =

    33.4675
   904.3034


  z =

     1
     1
```

Figura 13- Frequências e amortecimentos do sistema.

Com a maior frequência (904,3034 rad/s), pode-se converter para o respectivo tempo, através da Equação 7.

$$T = \frac{2\pi}{20.\omega_n} \quad (7)$$

$$T = \frac{2\pi}{20.904{,}3034} = 0{,}00035\ s$$

Assim, pode-se ou escolher um tempo de amostragem entre 0,0018s e 0,0060s, obtidos pelo método analítico, ou um tempo próximo ao valor calculado pelo método prático, de 0,00035s. Devido ao tempo de leitura do tacômetro ser muito maior do que os valores estipulados, uma vez que a frequência acaba sendo numericamente metade do valor da velocidade real em RPM, para uma frequência de 1000 rpm, o tempo de leitura seria de 0,5ms, ou seja, o tempo de amostragem do sistema não pode ser escolhido pelos métodos analisados, levando então a escolher um valor

arbitrário de 0,1ms de tempo de amostragem que, mesmo sendo um tempo relativamente lento para a resposta do motor, consegue acomodar a leitura do tacômetro na sua maior faixa de valores.

No desenvolvimento do código, reutilizou-se o processo visto em aula para gerar um clock de 10Hz, visando remover as oscilações de leitura dos botões e usar de tempo de amostragem do controlador, como pode ser visto juntamente a declaração da arquitetura e variáveis, nas Figuras 14, 15 e 16 abaixo.

```
LIBRARY IEEE;
USE  IEEE.STD_LOGIC_1164.ALL;
USE  IEEE.STD_LOGIC_ARITH.ALL;
USE  IEEE.STD_LOGIC_UNSIGNED.ALL;
LIBRARY WORK;
USE WORK.PACOTELCD.ALL;
USE WORK.PACOTETIPO.ALL;
USE WORK.PACOTEPI.ALL;

ENTITY MOTOR IS
   PORT(RESET, CLK_50MHZ          : IN   STD_LOGIC;
        LCD_RS, LCD_E, LCD_ON     : OUT  STD_LOGIC;
        LCD_RW                    : BUFFER STD_LOGIC;
        TACO                      : IN STD_LOGIC;
        DATA_BUS                  : INOUT   STD_LOGIC_VECTOR(7 DOWNTO 0);
        MOTOR1                    : OUT STD_LOGIC;
        BTMAIS1,BTMENOS1          : IN STD_LOGIC);
END MOTOR;
```

Figura 14- Declaração da arquitetura.

```
ARCHITECTURE A OF MOTOR IS
SIGNAL SOMA: INTEGER RANGE 0 TO 8388607:=0;
SIGNAL ERRO: INTEGER := 0;
SIGNAL ERROANT: INTEGER := 0;
SIGNAL KP: INTEGER := 6;
SIGNAL KI: INTEGER := 215;
SIGNAL CLK_10HZ,  INICIA, AGORA,ATPID: STD_LOGIC;
SIGNAL CLK_COUNT_10HZ: STD_LOGIC_VECTOR(24 DOWNTO 0);-- SINAL DE 100MS
SIGNAL CLK_127HZ : STD_LOGIC;
SIGNAL CLK_COUNT_127HZ: STD_LOGIC_VECTOR(8 DOWNTO 0);-- SINAL DE 2MS/255
SIGNAL LINHA1: FRASE;
SIGNAL LINHA2: FRASE;
SIGNAL CONTADOR1: INTEGER RANGE 0 TO 5000;
SIGNAL VALORTACO1: INTEGER RANGE 0 TO 5000;
SIGNAL CONTANT: INTEGER := 0;
SIGNAL AUX,AUX2,AUX3: INTEGER RANGE 0 TO 5000;
SIGNAL CONTATACO1: INTEGER RANGE 0 TO 100000:=0;
SIGNAL PI2: INTEGER RANGE 0 TO 8388607;
SIGNAL PID: INTEGER RANGE 0 TO 255;
SIGNAL MIL1,CENT1,DEZ1,UNI1,CKP,DKP,UKP,CKI,DKI,UKI,MVEL,CVEL,DVEL,UVEL,MKI: INTEGER RANGE 0 TO 10;
```

Figura 15- Variáveis utilizadas.

```
PROCESS
 BEGIN
  WAIT UNTIL CLK_50MHZ'EVENT AND CLK_50MHZ = '1';
    --CLK 10 HZ
    IF RESET = '0' THEN
     CLK_COUNT_10HZ <= "00000000000000000000000000";
     CLK_10HZ <= '0';
    ELSE
          IF CLK_COUNT_10HZ < X"4C4B40" THEN
           CLK_COUNT_10HZ <= CLK_COUNT_10HZ + 1;
          ELSE
           CLK_COUNT_10HZ <= "00000000000000000000000000";
           CLK_10HZ <= NOT CLK_10HZ;
          END IF;
    END IF;
 END PROCESS;
```

Figura 16- Processo para gerar o clock de 10 Hz.

Para a leitura do tacômetro, criou-se o processo que verificava e contava as bordas de subida da onda de entrada, como pode ser visto na Figura 17 a seguir.

```
PROCESS (TACO)
BEGIN
   IF TACO'EVENT AND TACO = '1' THEN
      CONTATACO1 <= CONTATACO1 +1;
   END IF;
END PROCESS;
```

Figura 17- Leitura do tacômetro.

No desenvolvimento do controle PI, utilizou-se um processo com o clock de 10Hz, onde nele rodava-se o algoritmo do controlador PI, juntamente com o debounce dos botões, representado pela Figura 18.

```
PROCESS (CLK_10HZ)
BEGIN
   IF CLK_10HZ'EVENT AND CLK_10HZ = '1' THEN
      VALORTACO1  <= ((CONTATACO1 - CONTANT)*10);
      CONTANT <= CONTATACO1;
      ERRO <= CONTADOR1 - VALORTACO1 ;
         SOMA <= SOMA + ((ERRO+ERROANT)/2); -- VAI INTEGRANDO O ERRO
         IF SOMA > 400000 THEN
            SOMA <= 400000;
         END IF;
         PI2 <= (KP*ERRO*10)+(KI*SOMA); -- CALCULA O PI
         IF(PI2 > 51000000) THEN --LIMITA PI EM 4000
            PI2 <= 51000000;
         END IF;
         PID <= PI2/200000;--AJUSTA PI PARA IR DE 0-255
         ERROANT <= ERRO;
      --DEBOUNCE DOS BOTOES
      IF BTMAIS1 = '0' AND (CONTADOR1 < 4000)  THEN
         CONTADOR1 <= CONTADOR1 + 100;
      ELSE
         IF BTMENOS1 = '0' AND (CONTADOR1 > 0) THEN
                CONTADOR1 <= CONTADOR1 - 100;
         END IF;
      END IF;
   END IF;
END PROCESS;
```

Figura 18- Algoritmo de controle PI e debounce dos botões.

Por fim, para a representação no display LCD, optou-se por exibir o valor do setpoint e da velocidade atual na primeira linha, com o valor da saída PI e o erro atual na segunda linha, como pode ser visto na Figura 19 abaixo.

```
MIL1 <= (CONTADOR1/1000);
AUX <= CONTADOR1 - (MIL1*1000);
CENT1 <= AUX / 100;
DEZ1 <= (AUX REM 100)/10;
UNI1 <= (AUX REM 100) REM 10;

MVEL <= VALORTACO1/1000;
AUX2 <= VALORTACO1 - (MVEL*1000);
CVEL <= AUX2 / 100;
DVEL <= (AUX2 REM 100)/10;
UVEL <= (AUX2 REM 100) REM 10;

CKP <= PID/ 100;
DKP <= (PID REM 100)/10;
UKP <= (PID REM 100) REM 10;

MKI <= (ERRO/1000);
AUX3 <= ERRO - (MKI*1000);
CKI <= AUX3 / 100;
DKI <= (AUX3 REM 100)/10;
UKI <= (AUX3 REM 100) REM 10;
LINHA1 <= ("SP=" & CHARACTER'VAL(MIL1+48) & CHARACTER'VAL(CENT1+48) &  CHARACTER'VAL(DEZ1+48)&...
LINHA2 <= ("PID=" & CHARACTER'VAL(CKP+48) &  CHARACTER'VAL(DKP+48)&( CHARACTER'VAL(UKP+48)) &...
CALCPI: COMPONENTEPI PORT MAP
(PID, MOTOR1, CLK_50MHZ);--(PID, MOTOR, CLK_50MHZ);
DISP1: COMPONENTELCD PORT MAP
(RESET, CLK_50MHZ ,LINHA1,LINHA2, LCD_RS, LCD_E, LCD_ON , LCD_RW,  DATA_BUS );
```

Figura 19- Divisão dos algarismos das variáveis e linhas do LCD.

CONCLUSÃO

Com o desenvolvimento do projeto, foi possível verificar os diferentes desafios no desenvolvimento de um controle digital para um motor CC, desde a obtenção do modelo do motor, necessário para o projeto do controlador via Matlab, até a implementação do algoritmo de controle PI em VHDL na FPGA. Os circuitos de interfaceamento entre o motor e a FPGA são relativamente simples e muito práticos para diversas situações, especialmente o drive de potência. Na aquisição dos parâmetros do motor, deve-se salientar que o método para obtenção da indutância de armadura não é muito eficiente, uma vez que despreza a força contraelemotriz gerada na armadura do motor, considerando apenas o circuito RL. Para a utilização do controle digital PI, não seria recomendado o tacômetro utilizado, pois não permite-se que o tempo de amostragem obtido pelos métodos analíticos ou práticos seja aplicado ao circuito, uma vez que a leitura do tacômetro é muito lenta e isso levava a instabilidade no circuito, sendo necessário um maior tempo de amostragem, que deixou a resposta lenta, mas controlável.

REFERÊNCIAS

FREIRE, Muriell de Rodrigues e. **Motor CC: Controle de Velocidade com PID Analógico**. São João da Boa Vista: 2019. 30 slides.

LABORATÓRIO DE SISTEMAS EMBARCADOS CRÍTICOS (São Paulo). Universidade de São Paulo. Estudos de Controle: São Carlos: Slide, 2019. 35 slides, color.

NISE, Norman S. Engenharia de sistemas de controle. 6. ed. Rio de Janeiro: LTC, 2012.

COSTA, Prof. Dr. Cesar de. Modelagem de um motor de corrente continua. Disponível em: <http://professorcesarcosta.com.br/upload/imagens_upload/Modelagem%20de%20um%20motor%20de%20corrente%20continua.pdf>. Acesso em: 04 dez. 2019.

INSTRUMENTS, National. Explicando a Teoria PID. 2019. Disponível em: <https://www.ni.com/pt-br/innovations/white-papers/06/pid-theory-explained.html>. Acesso em: 04 dez. 2019.

BRAGA, Newton C. FPGA: Alternativas para Projeto. 2008. Disponível em: <https://www.newtoncbraga.com.br/index.php/novos-componentes/52-artigos-tecnicos/artigos-diversos/12343-fpga-alternativa-para-projetos-art2873>. Acesso em: 04 dez. 2019.