SÃO PAULO
FACULDADE SENAI DE TECNOLOGIA MECATRÔNICA
REVISTA BRASILEIRA DE MECATRÔNICA

PLANTA SERVOPNEUMÁTICA MONITORADA POR SISTEMA SUPERVISÓRIO COM CÓDIGO ABERTO

SERVOPNEUMATIC PLANT MONITORED BY SUPERVISORY SYSTEM WITH OPEN CODE

Caio Vinícius Ribeiro da Silva [1], [i]
Paulo Sebastião Ladivez [2], [ii]

RESUMO

O objetivo deste artigo é demonstrar a implantação de um *software* livre para o monitoramento de uma planta servopneumática, que é dotada de um controlador baseado na plataforma Arduino Due. O *software* utilizado é o ScadaBr, que tem como principal vantagem o fato de ser baseado em código aberto e ser acessível para os profissionais da automação. Inicialmente, será apresentada a planta servopneumática e o controlador, bem como a programação do Arduino Due. Em seguida, será explicada a instalação do ScadaBr, a configuração da rede Modbus e como é feita a comunicação com o *software* supervisório. Por fim, será descrita uma aplicação em que o ScadaBr monitora, através de um gráfico, o deslocamento de um eixo da planta servopneumática.

Palavras-chave: Software livre. ScadaBr. Modbus. Arduino. Arduino Due.

ABSTRACT

The objective of this article is demonstrating the implementation of a free software for the monitoring of a servo pneumatic plant, which is equipped with a controller based on Arduino due plataform. The software used is ScadaBr, which has the main advantage of being open source based and accessible to automation professionals. Initially, the servo pneumatic plant and the controller will be presented, as well as the programming of the Arduino Due. Next, the installation of the ScadaBr, the configuration of the Modbus network and how the communication with the supervisory software will be explained will be explained. Finally, an application will be described in which the ScadaBr monitors, through a graph, the displacement of an axis of the servo pneumatic plant.

Keywords: Free Software. ScadaBr. Modbus. Arduino. Arduino Due.

Data de submissão: 24/02/2018
Data de aprovação: 25/07/2018

[1]Tecnólogo e Professor da Escola e Faculdade de Tecnologia SENAI Mariano Ferraz. E-mail: caio.cvrs@gmail.com
[2]Professor da Faculdade SENAI de Tecnologia Mecatrônica. E-mail: paulo.ladivez@sp.senai.br

1 INTRODUÇÃO

O mundo contemporâneo vive um momento onde a automação está presente em praticamente todas as cadeias produtivas. Sobre esse aspecto, Dorf (2001) ressalta que:

> O controle de um processo industrial (manufatura, produção e assim por diante) de modo automático em vez de manual é chamado frequentemente de automação. A automação é utilizada amplamente nas indústrias químicas, de energia elétrica, de papel, automobilística e siderúrgica, dentre outras. (2001, p. 7).

A presença massiva da automação pode ser justificada pelo salto tecnológico nas últimas décadas, que proporcionou um avanço em progressão geométrica no cenário mundial. O que antes era feito somente a partir de componentes eletromecânicos, atualmente ganha força e impulso com o advento da informática e dos sistemas microcontrolados. Os aplicativos de computadores invadiram as indústrias, especialmente aqueles usados para programar controladores e supervisionar determinados processos.

Os principais aplicativos difundidos na indústria, em escala mundial, são propriedades de empresas multinacionais. Essas transnacionais, como: *Siemens*, *Festo*, *Schneider Electric*, dentre outras, investiram capital financeiro e intelectual para desenvolver *softwares* e *hardwares* compatíveis com o padrão internacional.

Todavia, para ter acesso à essas tecnologias, o investimento é alto. Nem todas as empresas estão dispostas ou têm condição financeira para adquirir um controlador ou um aplicativo de supervisão consagrado no mercado. Os estudantes também ficam refém em relação a questão financeira, e muitas vezes utilizam uma versão limitada do aplicativo que pode ou não atender as suas necessidades.

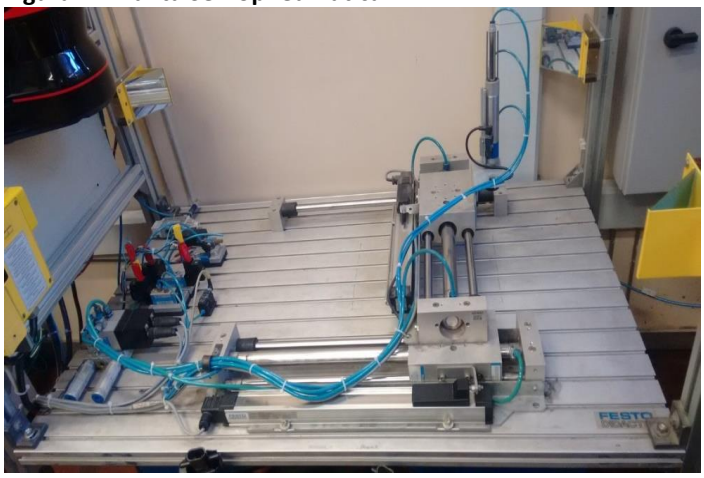
Diante desse problema, uma alternativa viável é o uso de *softwares* de código livre. Por serem desenvolvido por comunidades de profissionais e entusiastas, são livremente licenciados. Isso significa que o usuário pode fazer *download* gratuitamente, além de acessar o código fonte, e assim, melhorar ou modificar de acordo com o desejado.

Se por um lado existe a vantagem da gratuidade e da customização, por outro lado essa abertura pode ocasionar eventuais problemas técnicos ou instabilidade funcional. Portanto, esse artigo irá abordar o funcionamento do *software* ScadaBr numa situação real, utilizando uma planta servopneumática didática e constatar como o sistema se comporta.

2 PLANTA SERVOPNEUMÁTICA

A planta servopneumática utilizada está localizada na Faculdade de Tecnologia SENAI Mariano Ferraz. A sua finalidade é para que os alunos do curso superior de Tecnologia em Automação Industrial possam implementar sistemas de controle convencionais em conjunto com sistemas supervisórios, conforme mostra a figura 1.

Figura 1 - Planta Servopneumática



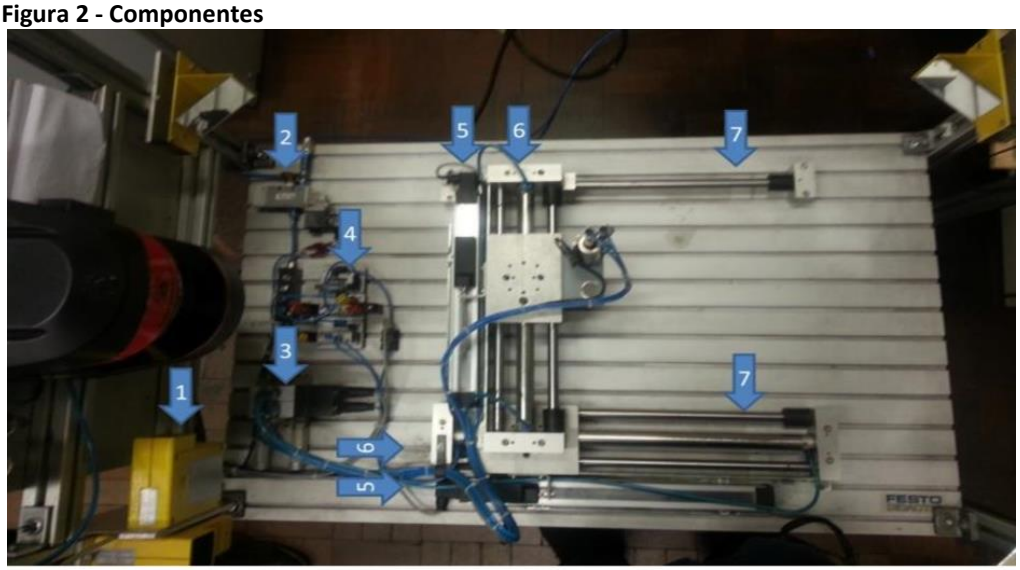
Fonte: Dados do autor

Esse sistema de automação possui três atuadores pneumáticos, sendo que dois deles são para as movimentações na horizontal, chamados de *eixo x* e *eixo y*, e o último para a movimentação na vertical, chamado de *eixo z*. Cada um dos atuadores com movimentação na horizontal é controlado por uma válvula pneumática proporcional e o atuador situado no *eixo z* é controlado por uma válvula pneumática convencional. Nesse eixo, está acoplada uma ventosa, que realiza a sucção de peças, por meio de uma válvula convencional. Há também um sistema de segurança por barreira de luz, configurado para cortar a alimentação pneumática assim que o campo de atuação do sensor for interrompido.

A planta servopneumática é composta também por duas réguas potenciométricas, uma para o eixo x e outra para o eixo y. As réguas potenciométricas são os sensores que indicam a posição atual dos eixos na horizontal, o que caracteriza a planta como um sistema de controle de malha fechada.

Os componentes pneumáticos e eletropneumáticos estão dispostos em uma mesa didática, conforme figura 2.

Figura 2 - Componentes



Fonte: Dados do autor

De acordo com a figura 2, podem ser observados os seguintes equipamentos:

a) Barreira de segurança laser;
b) Válvula de alimentação pneumática;
c) Válvulas eletropneumáticas proporcionais;
d) Válvulas eletropneumáticas convencionais;
e) Réguas potenciométricas;
f) Eixos de movimentação;
g) Guia de movimentação.

3 CONTROLADOR

Esse sistema possui um microcontrolador da Atmel, modelo SAM3X8E ARM Cortex-M3, inserido na plataforma de prototipagem Arduino Due. É importante ressaltar que o microcontrolador trabalha na faixa dos 3,3 V, enquanto os outros componentes funcionam na faixa de 24V. Para resolver esse problema, a planta servopneumática é dotada de uma placa eletrônica, que realiza as devidas conversões de sinais.

Outro fator importante é que esse microcontrolador possui duas saídas analógicas, chamadas respectivamente de DAC0 e DAC1, com resolução de 12 bits. Ambas as saídas são utilizadas para o controle das duas válvulas proporcionais. Além dessas conexões, a plataforma Arduino Due possui doze entradas analógicas, sendo que apenas duas estão em uso para a leitura das réguas potenciométricas. Essa leitura é realizada na resolução de 10 bits.

4 PROGRAMAÇÃO

A configuração da rede Modbus no Arduino Due será feita pela biblioteca *SimpleModbusSlave_DUE*, (Figura 3) que permite a criação de registradores no parâmetro *enum.* O único registrador, designado *READX,* será associado a leitura analógica do pino A0.

Figura 3 - Configuração da Biblioteca Modbus

```
#include <SimpleModbusSlave_DUE.h> //Inclui a bibioteca Modbus
#define  snsPtx A0

enum {
  READX,   //ENTRADA ANALÓGICA - Leitura do eixo X
  HOLDING_REGS_SIZE
};

unsigned int holdingRegs[HOLDING_REGS_SIZE];

//Função para avançar o Eixo X
void avancaX(){
  analogWrite(DAC0, 2350);
}

//Função para recuar o Eixo X
void recuaX(){
  analogWrite(DAC0, 1800);
}
//Função de Configuração
void setup(){

  analogWriteResolution(12);//configura a resolução da saída analógica
  pinMode(DAC0, OUTPUT);    //define o DAC0 como saída

  //Configura os parâmetros da rede modbus
  modbus_configure(&Serial, 9600, 1, 2, HOLDING_REGS_SIZE, holdingRegs);
  modbus_update_comms(9600, 1);

  recuaX();
  delay(5000);
}
```

Fonte: Dados do autor

A programação principal, localizada na função *void loop,* é baseada em duas instruções *while,* uma para avanço e outra para recuo do eixo x. Enquanto a leitura da régua potenciométrica, alocada na entrada analógica *A0*, estiver abaixo de 870, significa que o atuador precisa avançar até atingir esse valor. Dentro da instrução *while*, o registrador *READX* recebe o valor da leitura convertido para porcentagem, através da função do arduino chamada *map*. Em seguida, a função de avanço[3] é chamada e o registrador é atualizado. A partir dessa atualização, feita pela função *modbus_update*[4], o *datapoint* criado no ScadaBr recebe o valor convertido, que será utilizado para fornecer dados para um gráfico. A figura 4 a seguir mostra os detalhes da programação.

Figura 4 - Programação principal

```
35  void loop(){
36
37    while(analogRead(A0)<870){ //enquanto o eixo X não atingir o fim de curso
38
39      // holdingRegs[READX] = map (analogRead(A0),0,870,0,100); //conversão para porcentagem
40      holdingRegs[READX] = map (analogRead(A0),0,1023,0,100); //conversão para porcentagem
41      avancaX(); //função para avançar o eixo X
42      modbus_update(); //atualiza os registradores modbus
43    }
44    delay(2000);
45
46    while(analogRead(A0)>0){ //enquanto o eixo X não atingir o início de curso
47      // holdingRegs[READX] = map (analogRead(A0),0,870,0,100); //conversão para porcentagem
48      holdingRegs[READX] = map (analogRead(A0),0,1023,0,100); //conversão para porcentagem
49      recuaX(); //função para recuar o eixo X
50      modbus_update(); //atualiza os registradores modbus
51    }
52
53    delay(2000);
54  }
```

Fonte: Dados do autor

5 SCADABR

A planta servopneumática descrita nesse artigo possui sensores, responsáveis pela aquisição de dados, e válvulas, que vão atuar no sistema. Esses dispositivos podem ser monitorados/controlados por sistemas SCADA (*supervisory control and data aquisition*). Segundo Ahihara *et al.* (2001):

> Um sistema de supervisão é responsável pelo monitoramento de variáveis de controle do sistema, com o objetivo principal de fornecer subsídios ao operador (homem-máquina) para controlar ou monitorar um processo automatizado mais rapidamente, permitindo a leitura de variáveis em tempo real e o gerenciamento e controle do processo automatizado. (AHIHARA *et al.* 2001, apud ROSÁRIO, 2005, p. 294):

O sistema de supervisão utilizado será o ScadaBr, descrito como "um *software* livre, gratuito e de código-fonte aberto, para desenvolvimento de aplicações de automação, aquisição de dados e controle supervisório" (SCADABR, 2017). O *download* dessa plataforma pode ser feito diretamente no site, sem qualquer custo.

[3] AvancaX()

[4] A função *modbus_update()* faz parte da biblioteca *SimpleModbusSlave_DUE*

Durante o processo de instalação, alguns procedimentos precisam ser seguidos. O primeiro deles é a configuração da Máquina Virtual Java. Após vários testes com diferentes versões, foi constatado que a versão 1.8.0_144, tanto para o *java* quanto para o *javac*[5], é a eficaz. Outras versões, em especial as mais atualizadas, comprometem o funcionamento do software. Em seguida, a porta de conexão deve ser inserida de acordo com o uso do computador, que para esse caso foi a 8085. Ao final da instalação, uma aplicação do Tomcat[6] irá abrir e através da aba *general,* o servidor poderá ser habilitado. Contudo, é essencial que duas variáveis de ambiente sejam associadas: path[7] e JAVA_HOME[8]. Como o *software* funciona a partir do navegador[9] em conjunto com o Java, é necessária a instalação do Adobe Air. Após o procedimento de instalação, o usuário é redirecionado para a tela de *login* no navegador. Os campos *User id* e *Password* podem ser preenchidos com admin, que é o usuário padrão do sistema (figura 5).

Figura 5 – Login no Sistema

Fonte: Dados do autor

6 REDES INDUSTRIAIS

A complexidade dos processos industriais nas últimas décadas requisitou soluções de automação mais dinâmicas. Em outras palavras, o que antes era controlado no local e de maneira mais simples passou a ser controlado a distância e de forma mais sofisticada, utilizando computadores interligados aos controladores. É nesse contexto que surgem as redes industriais, conforme destaca Rosário (2005):

> As redes industriais surgiram da necessidade de interligar PCs e CLPs, que se proliferavam operando independentemente. A interligação desses equipamentos em rede permitiu o compartilhamento de recursos e base de dados, as quais passaram a ser únicas e não mais replicadas, o que conferiu mais segurança aos usuários da informação. (ROSÁRIO, 2005, p. 318).

Boa parte dos profissionais da automação industrial precisa lidar constantemente com sistemas que possuem uma determinada rede industrial. Essa ferramenta possibilita a interligação de diversos dispositivos, tais como sensores, atuadores, controladores, entre outros dispositivos. A rede industrial utilizada neste artigo será a Modbus.

[5] Utilizado para compilar as classes em Java.
[6] Servidor web Java.
[7] Path: ;%JAVA_HOME%;
[8] JAVA_HOME: C:\Program Files\Java\jdk1.8.0_144
[9] Internet Explorer, Mozila Firefox, dentro outros. Neste projeto foi utilizado o Google Chrome.

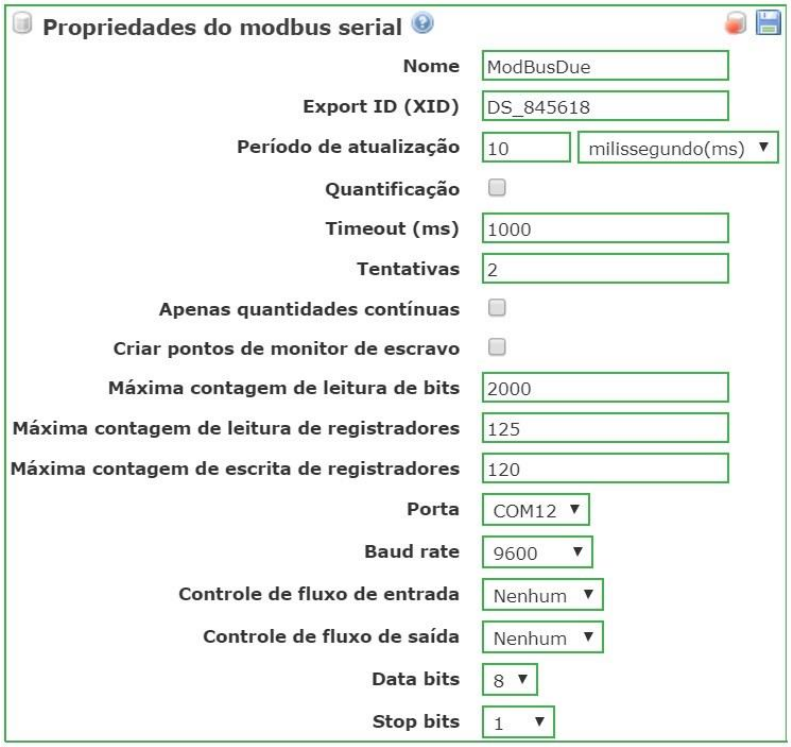
6.1 Modbus

É uma rede industrial de comunicação criada em 1979, com a finalidade de ser um padrão de comunicação aberto. De acordo com a Schneider (2017), a atual proprietária, a rede Modbus "suporta uma grande quantidade de produtos vendidos no mercado atualmente.". A popularidade desse protocolo industrial se deve a facilidade de implementação, que pode ser aplicado na configuração mestre/escravo. Além disso, a rede Modbus tem 256 endereços, que podem ser utilizados de acordo com a necessidade do programador.

6.2 Configuração da rede ModBus no ScadaBr

A partir do campo chamado *data sources*, localizado na parte superior do ScadaBr, a rede Modbus Serial é criada, figura 6, sendo necessário preencher os campos: Nome, período de atualização, Porta, Baud rate, Data bits e Stop bits. Esses últimos campos devem ser os mesmos configurados na programação do Arduino.

Figura 6 - Configuração da Rede Modbus



Fonte: Dados do autor

Com a rede criada e configurada, a próxima etapa é criar o *data point*, que é responsável por receber os valores atualizados do *holding register*, declarados na estação escrava, conforme figura 7.

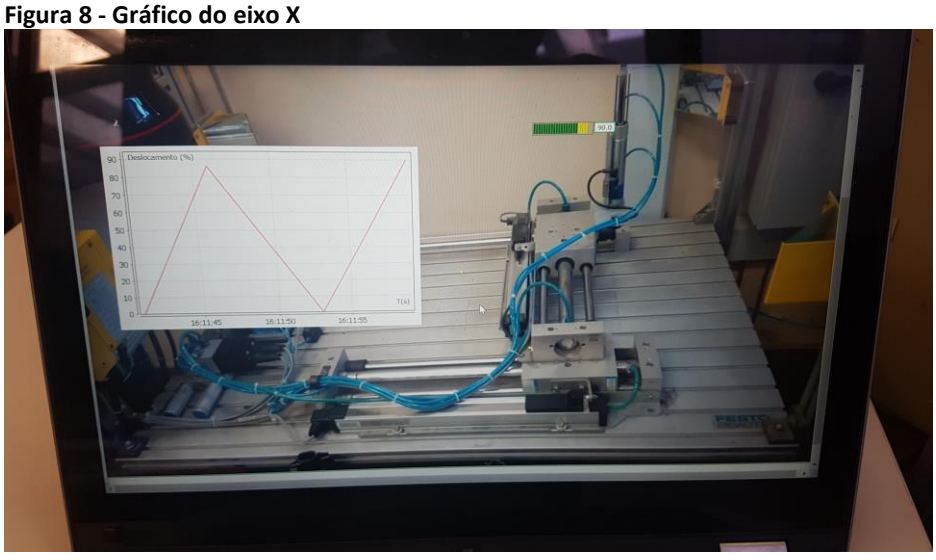
Figura 7 - Configuração do *data point*

Fonte: Dados do autor

7 MONITORAMENTO DO EIXO X

Após a programação do Arduino e a configuração do sistema de supervisão, a última etapa realizada foi criar uma tela de monitoramento do eixo. O ScadaBr oferece o recurso "Gráfico", em que o programador deve associar um *data point* para visualizar os dados. O *data point* associado foi o *READX,* que, após a conexão da planta servopneumática via Arduino, criou uma forma de onda triangular, conforme o avanço e recuo do atuador. A versatilidade desse *software* permite alterar o período de atualização do gráfico, desde segundos até em anos. A figura 8 mostra monitoramento do eixo x.

Figura 8 - Gráfico do eixo X



Fonte: Dados do autor

8 CONSIDERAÇÕES FINAIS

A comunicação entre o Arduino Due e o ScadaBr se mostrou estável e atendeu às expectativas. O gráfico foi gerado de acordo com o avanço e recuo do eixo x. Durante a fase de desenvolvimento, foi necessário um investimento em pesquisa sobre o funcionamento da biblioteca Modbus e a instalação do *software* de supervisão.

O ScadaBr possui ótimos recursos gráficos, além trabalhar com várias redes industriais. Essa versatilidade abre um leque de várias possibilidades para que os profissionais de automação, estudantes e entusiastas possam criar projetos utilizando a plataforma Arduino ou mesmo outros controladores suportados pelas redes industriais utilizadas pelo sistema supervisório.

REFERÊNCIAS

BONACORSO, Nelson Gauze, NOLL, Valdir. **Automação eletropneumática**. São Paulo: Érica, 1997. 137 p.

BOYER, Stuart A. **SCADA supervisory control and data acquisition.** 4. ed. Carolina do Norte: ISA, 2010. 257 p.

DORF, Richard C; Bishop, Robert H. **Sistemas de controle modernos.** 8. ed. Rio de Janeiro: LTC, 2001. 680 p.

GROOVER, Mikell P. **Automação industrial e sistemas de manufatura.** 3 ed. São Paulo: Pearson Prentice Hall, 2011. 582 p.

LUGLI, Alexandre B.; Santos, Max Mauro Dias. **Redes industriais para automação Industrial AS-I, Profibus e Profinet.** 8 ed. Rio de Janeiro: LTC, 2001. 680 p.

MARCONI, Marina de Andrade, LAKATOS, Eva Maria. **Metodologia do trabalho Científico.** 7 ed. São Paulo: Atlas, 2007. 225 p.

MCROBERTS, Michael. **Arduino básico.** São Paulo: Novatec Editora, 2011. 453 p.

ROSÁRIO, João Maurício. **Princípios de mecatrônica.** 1 ed. São Paulo: Erica, 2010. 174 p.

SCADABR. **Página principal**. 2017. Disponível em: <http://www.ScadaBr.com.br>. Acesso em: 22 set. 2017.

SCHNEIDER Electric. **Modbus**. Widely used serial fieldbus for all application, 2017. Disponível em: <http://www.schneider-electric.us/en/product-range-presentation/547-modbus/>. Acesso em: 29 set. 2017.

AGRADECIMENTOS

Ao meu pai, Nivaldo, pelo incentivo e apoio nos estudos desde a minha infância, e por me apresentar a Escola SENAI quando eu tinha apenas 15 anos. A minha mãe, Iraneide, que sempre me motivou e me amparou em todas as minhas necessidades.

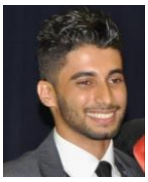
Aos meus professores do Curso de Aprendizagem Industrial em Eletricista de Manutenção, Curso Técnico em Mecatrônica, Curso Superior de Tecnologia em Eletrônica Industrial e Curso de Pós-Graduação em Automação Industrial. Todos são responsáveis pelo meu interesse e avanço acadêmico/profissional nessa área tão magnífica que é a automação.

A Faculdade de Tecnologia SENAI Anchieta por me receber e dar suporte, após anos de formado. A Faculdade de Tecnologia SENAI Mariano Ferraz, por ceder a planta servopneumática, usada neste trabalho. A toda equipe da Faculdade SENAI de Tecnologia Mecatrônica, pela excelência no ensino.

Ao Prof. Esp. Paulo Sebastião Ladivez, por nossas ótimas conversas sobre automação, por sua orientação eficiente e a sua incrível disposição em esclarecer as dúvidas.

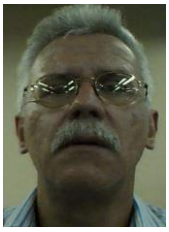
Sobre os autores:

[i] **Caio Vinícius Ribeiro da Silva**



Possui curso Técnico em Mecatrônica pela Escola Técnica Estadual Getúlio Vargas, graduação em Teologia pela Universidade Presbiteriana Mackenzie (2015), graduação em Tecnologia em Eletrônica Industrial pela Faculdade de Tecnologia SENAI Anchieta (2012). Cursando atualmente a Pós-Graduação em Automação Industrial pela Faculdade SENAI de Tecnologia Mecatrônica (2017). Tem experiência na área de Engenharia Eletrônica, com ênfase em Projetos de Automação e Sistemas Microcontrolados. Atualmente é professor na Escola e Faculdade de Tecnologia SENAI Mariano Ferraz.

[ii] **Paulo Sebastião Ladivez**



Possui graduação em Engenharia Elétrica pela Universidade Mogi das Cruzes (1984) com especialização em Tecnologias e Sistemas de Informação pela Universidade Federal do ABC (2013). Atualmente é professor da Faculdade SENAI de Tecnologia Mecatrônica, lecionando as disciplinas Projetos, Microcontroladores, Linguagem de Programação no curso Tecnológico em Mecatrônica Industrial e na Pós-Graduação em Automação Industrial. Tem experiência na área de Engenharia Eletrônica, com ênfase em Automação Industrial e Mecatrônica, atuando principalmente nos seguintes temas: Mecatrônica, Manufatura Digital, Redes Industriais, Automação Industrial, Microcontroladores e Controle.