DOI 10.21727/rt.v14i2.3057

Automação de baixo custo, um estudo de caso: inversor de frequência controlado por supervisório e arduino

Low cost automation, a case study: frequency inverter controlled by supervisory and arduino

Wallace Soares da Siva[1], Alex Franco Ferreira[2]

Como citar esse artigo. Silva, W.S; Ferreira, A.F. Automação de baixo custo, um estudo de caso: inversor de frequência controlado por supervisório e arduino. Revista Teccen. 2021 Jul./Dez.; 14 (2): 28-35.

Resumo

O Arduino foi criado com o intuito de facilitar a programação de microcontroladores para estudantes e leigos na área de eletrônica, mas rapidamente se expandiu e hoje existem várias comunidades que criam e compartilham projetos para esta plataforma, devido a seu custo e facilidade de programação. O presente trabalho teve como intuito a elaboração de um sistema de baixo custo, fácil acesso e programação, que consiste no controle de um inversor de frequência através da plataforma Arduino e de um sistema supervisório. Para realizar tal controle foi utilizado o software Elipse SCADA, nele foi criado um programa que possui um botão e um slider, quando o botão é acionado, o inversor e sua função de controle de frequência são habilitados, a variação de frequência é feita através do slider. Este controle é realizado através da lógica presente no Arduino. A comunicação entre o supervisório e o Arduino é feita através da porta USB via rede Modbus RTU utilizando o driver Modbus do Elipse. Já a comunicação entre o inversor e o Arduino é feita através das portas digitais e analógicas presentes tanto no Arduino, quanto no inversor de frequência. O objetivo foi concluído com êxito, pois foi criada uma alternativa de fácil utilização e implementação, além de ser uma ótima opção em relação ao seu custo.
Palavras-Chave: Arduino, Elipse SCADA, Supervisório, Modbus, Inversor de Frequência, Automação.

Abstract

The Arduino was created in order to facilitate the programming of microcontrollers for students and lay people in the electronics area, but it quickly expanded and today there are several communities that create and share projects for this platform, due to its cost and ease of programming. The present work aimed to develop a low cost, easy access and programming system, which consists of controlling a frequency inverter through the Arduino platform and a supervisory system. To perform such control, the Elipse SCADA software was used, a program was created that has a button and a slider, when the button is pressed, the inverter and its frequency control function are enabled, the frequency variation is done through the slider. This control is carried out through the logic present in the Arduino. The communication between the supervisory and the Arduino is made through the USB port via Modbus RTU network using Elipse's Modbus driver. The communication between the inverter and the Arduino is done through the digital and analog ports present in both the Arduino and the frequency inverter. The objective was successfully completed, as an alternative that was easy to use and implement was created, in addition to being a great option in relation to its cost.
Keywords: Arduino, Elipse SCADA, Supervisory System, Modbus, Frequency Inverter.

Introdução

O Arduino foi criado com o intuito de facilitar a programação de microcontroladores para estudantes e leigos na área de eletrônica, porém, com o passar do tempo, foram surgindo várias comunidades que criam e compartilham variados tipos de projetos para a plataforma, que vão desde pequenos projetos com LED's, a automações residenciais complexas (McRoberts, 2010).

O objetivo deste trabalho foi criar uma forma de variar a velocidade de um inversor de frequência, com comandos executados de forma remota através de um supervisório, com o principal intuito de ser uma forma alternativa, de fácil acesso e principalmente de baixo custo.

Afiliação dos autores:

Universidade de Vassouras, Vassouras, RJ, Brasil.

* Email para correspondência: Wallace_soares28@hotmail.com

Recebido em: 02/11/21. Aceito em: 11/11/21.

DOI 10.21727/rt.v14i2.3057

Para cumprir tal objetivo, foi utilizado a plataforma Arduino, que segundo Arduino.cc (2018), é uma plataforma de código aberto baseadas em software e hardware fácil de usar. Com ela foi possível criar um software dentro de sua IDE, capaz de se comunicar com um supervisório, dentro do mesmo é possível gerar comando que atuam dentro da lógica do software do Arduino, que conectado a um inversor de frequência, variam a velocidade de um motor de indução trifásico.

Através deste sistema, basicamente qualquer pessoa pode operar o supervisório, pois é uma ferramenta bem prática e simples de ser utilizada, bastando alguns cliques do mouse do computador, é possível executar controles com no caso deste apresentado, variar a velocidade de um motor através de um inversor de frequência.

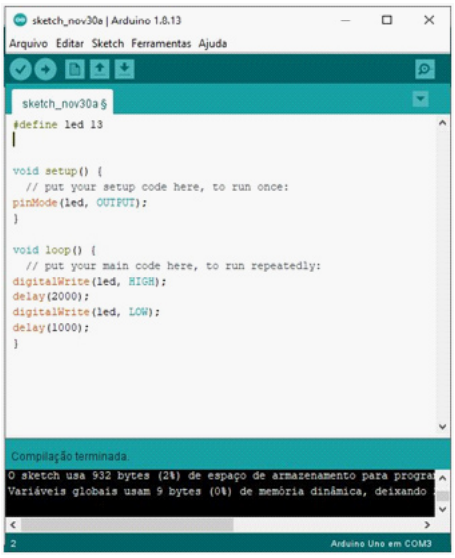

Figura 1. Arduino IDE.

Fonte: Autores, 2020.

Arduino

O Arduino é uma plataforma *open source,* ou seja, de código aberto, criada pelo italiano Massimo Banzi (e outros colaboradores) com a intenção de ensinar programação de microcontroladores para estudantes de design e leigos em eletrônica, para que fossem utilizados em projetos automação, robótica e arte (Ferroni, et al., 2014). Para Mcroberts (2010), em termos leigos, ele é um computador pequeno com suporte de entradas e saídas integradas, que é possível programar para que o mesmo processe os dados dos equipamentos externos conectados a ele, com isso, ele é conhecido como uma plataforma física capaz de interagir com o meio externo através de *hardware* e *software*.

O Arduino consiste em uma placa com um design de *hardware* aberto que possui um processador Atmel AVR e um conjunto de entradas e saídas integrado (McRoberts, 2010). A programação é feita através de uma IDE (ambiente de desenvolvimento integrado), baseada em linguagem *Processing*. Esta IDE é executada no computador onde será feita a programação é chamada de sketch, conforme a Figura 1. Após a edição, a programação é compilada, para que seja verificado se possui algum erro e é feito o *upload* para a placa.

Por ser uma plataforma de código aberto, ter um custo muito baixo e também ter um bom nível de tolerância a erros comuns de iniciantes, se popularizou muito rápido e foram criadas várias comunidades e fóruns, onde existem muitos tutoriais para compartilhar e discutir projetos e ideias. Deste modo qualquer pessoa com o mínimo conhecimento teórico pode fazer *download* de projetos e executa-los da forma que preferir (Arduino.cc, 2018).

Conforme a Figura 2, é demonstrada a placa Arduino Uno R3, nela é possível ver vários conectores, eles servem para a conexão e interface de equipamentos externos. Nele é possível identificar os seguintes

Figura 2. Arduino Uno R3.

Fonte: Autores, 2020.

conectores:

Pinos 0 a 13: São entradas ou saídas digitais, os pinos identificados com "~" possuem também a função de PWM (modulação por largura de pulso) e os pinos 0 e 1 também possuem a função de Rx e Tx, usados para comunicação;

Pinos A0 a A5: Usados para entradas analógicas, ou seja, recebem um valor de tensão que deve estar entre 0 a 5V e convertem em um valor de 0 a 1023 (Arduino.cc, 2018).

A placa pode ser alimentada pelo USB do computador a que estiver conectado, por uma bateria de 9V, através de um adaptador e por um adaptador AC com tensão recomendada entre 7 a 12V.

DOI 10.21727/rt.v14i2.3057

Elipse SCADA

É um software desenvolvido pela Elipse Software, para criação de sistemas de supervisão e controles de processos. Ele alia grande versatilidade com alto rendimento, aliado a diversos recursos presentes que facilitam o desenvolvimento da aplicação. Ao usuário é permitido todos os tipos de configurações, permite o monitoramento de variáveis do processo em tempo real através de gráficos e objetos. Também é possível acionar equipamentos e coletar informações de dispositivos de aquisição de dados presentes na planta. Através do Elipse Basic, é possível automatizar tarefas, para atender as necessidades específicas da empresa (Elipse Software, 2015).

Pacotes do Elipse SCADA

A plataforma Elipse SCADA está disponível em diferentes tipos de pacotes, deforma atender as necessidades de cada tipo de cliente (Elipse Software, 2015).

Elipse View

É indicado para aplicações como interface com operador, monitoração e acionamentos, ou seja, aplicações mais simples. Não possui ferramentas para armazenar dados de históricos, alarmes ou relatórios, além funcionalidades de pacotes avançados (Elipse Software, 2015).

Elipse MMI (*Man Machine Interface*)

Possui um sistema de supervisão completo, acompanha banco de dados proprietário, relatórios formatados, histórico, receita, alarmes e controle estatístico de processo. Podendo ser ainda um servidor de dados para outras aplicações Elipse. Possui todas as funções do pacote View (Elipse Software, 2015).

Elipse Pro

É a ferramenta mais avançada, pois permite trocar dados com outras estações em tempo real, transferir ou atualizar bancos de dados, programar *setpoints* e realizar comandos via rede. Possui todas as funções do pacote MMI. Este pacote é ideal para sistemas corporativos pois possui suporte para variados tipos de rede e permite a troca de informações com softwares dedicados a controle (Elipse Software, 2015).

Elipse CE

Permite executar aplicações Elipse SCADA em dispositivos com Windows CE embarcado, como IHMs e em geral dispositivos sem disco e dispositivos moveis. Ele suporta todas as funcionalidades dos pacotes anteriores (Elipse Software, 2015).

Módulos de Operação

O Elipse SCADA permite através de um dispositivo de proteção (*hardkey*) conectado ao computador, três módulos de operação: Runtime, Configurador e Master (inclui os módulos Configurador e Runtime). No módulo Runtime, só é permitido executar a aplicação, enquanto que nos módulos Configurador e Master foram desenvolvidos para criação e desenvolvimento de aplicativos (Elipse Software, 2015).

Na ausência do *hardkey*, o software executa em modo Demonstração, que possui quase toda as funções do módulo Configurador, exceto que só é permitido salvar até 20 tags, 5 conexões com Elipse Web e executar a aplicação por até duas horas (Elipse Software, 2015).

Ainda existem as versões Lite dos módulos Runtime e Master, onde as mesmas continuam com as mesmas funcionalidades, porém com limitação de tags (variáveis): Lite 75, com limite de 75 tags e Lite 300, com limitação de 300 tags (Elipse Software, 2015).

Plug-Ins

São ferramentas adicionais que permitem a expansão de recursos do software, eles podem ser adquiridos separadamente e são acessíveis em qualquer módulo ou verão de *software* (Elipse Software, 2015).

Inversor de Frequência

Há algum tempo atrás, para que se pudesse ter um controle de velocidade, eram utilizados motores de corrente contínua. Porém os custos desses equipamentos eram altos, além do que, era necessário a retificação da tensão. Com a evolução da eletrônica e diminuição de custos, surgiu então o inversor de frequência, um equipamento capaz de controlar a velocidade de motores de indução trifásico e assim substituir os motores de corrente contínua (Franchi, 2008).

Funcionamento do Inversor

O princípio de funcionamento do inversor consiste na variação da frequência da fonte de tensão, provendo assim um ajuste contínuo de conjugado e velocidade em relação a carga mecânica presente no motor (Franchi, 2008).

Quando o motor é alimentado diretamente pela tensão da rede, recebe a frequência da mesma que é 60Hz, sendo assim, a velocidade do motor será a nominal, respeitando a seguinte Equação 1:

$$n = \frac{120\ x\ f\ x\ (1 - s)}{p}$$

Onde:
n = velocidade em rotações por minuto (RPM)
f = frequência da rede em Hertz (Hz)
s = escorregamento
p = número de polos

De acordo com a equação pode-se perceber que ao se a frequência for alterada, a velocidade também será e justamente o que o inversor faz (WEG, 2005)

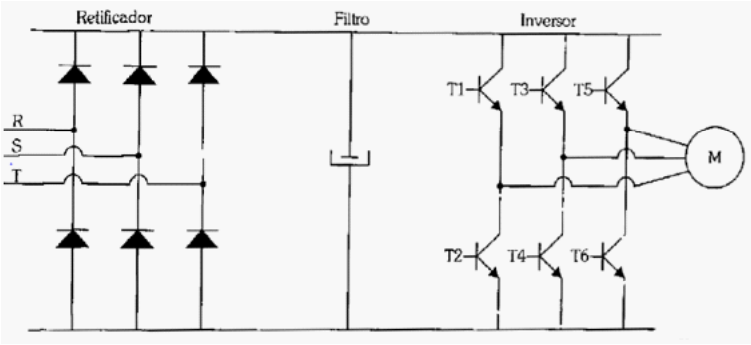

Figura 3. Diagrama de Blocos do Inversor.

Fonte: Franchi, 2008.

O diagrama da Figura 3 demonstra a topologia do inversor de frequência. Nele é possível identificar a parte retificadora que tem a função de transformar a corrente alternada de entrada, em corrente contínua, que posteriormente é filtrada e conectada aos terminais de saída através dos terminais dos tiristores T1 a T6 que funcionam como chaves estáticas, em corte ou saturação controladas pelo circuito de comando (Franchi, 2008).

Existem variados tipos de tecnologias implementados em inversores, mas a mais utilizada é a de IGBT (transistor bipolar com porta isolada). Com eles é utilizado a estratégia PWM (*Pulse Width Modulation)* ou "Modulação por largura de Pulso" que permite através do chaveamento dos IGBTs a criação de ondas senoidais de frequência variável (WEG , 2005). A forma de onda gerada na saída é demonstrada através da Figura 4.

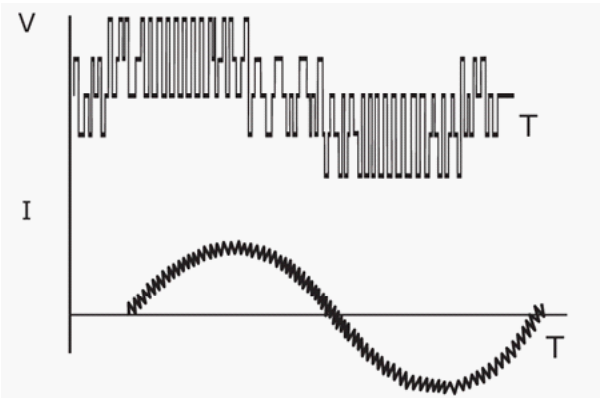

Figura 4. Forma de Onda Gerada por PWM.

Fonte: WEG , 2005.

Métodos de Controle dos Inversores de Frequência

Quanto ao tipo de controle, podem ser classificados em:

- Controle Escalar – baseia-se no controle chamado de "V/F constante", que mantém o torque nominal do motor constante igual a nominal, independentemente da velocidade do motor. É chamada desta forma pois frequência e tensão são variadas proporcionalmente. Este tipo de controle é usado em aplicações normais onde não é exigida muita dinâmica e precisão do processo (WEG , 2005).

- Controle Vetorial – No modo "V/F" o inversor usa a velocidade como referência para o controle dos parâmetros de tensão e frequência, já no modo vetorial, é feito o cálculo da corrente necessária para produção do torque requerido pelo motor, através do cálculo da corrente de estator e magnetização. Em alguns sistemas vetoriais faz-se necessário o uso de sensores de rotação acoplados ao eixo para que se tenha uma melhor dinâmica durante o processo. Este controle é utilizado em aplicações dinâmicas e que necessitam de controle de torque (WEG , 2005).

Modbus

Protocolo de comunicação desenvolvido pela empresa Modicon Industrial Automation Systems, que posteriormente veio a se tornar Schneider, para comunicação entre dispositivos mestre-escravo/cliente-servidor. Embora seja comumente usado em conexões seriais como RS-232, também pode utilizado na forma TCP/IP sobre Ethernet e MAP (Seixas, 2007).

Este protocolo define basicamente uma estrutura de mensagens definidas por bytes que diversos dispositivos são capazes de identificar que independe do tipo de rede utilizada (ALFA Instrumentos, 2000).

DOI 10.21727/rt.v14i2.3057

A comunicação é baseada na técnica mestre-escravo, onde apenas o dispositivo mestre pode iniciar comunicação (*query*). Os dispositivos escravos respondem ao mestre enviando os dados solicitados. Ao mestre é permitido enviar mensagens diretamente para cada escravo através do endereçamento do escravo ou acessar todos da rede através de mensagens em cadeia (*broadcast*). Quando o mestre envia uma mensagem endereçado a um escravo, somente ele responde (*response*), nunca são gerados *responses* quando a mensagem for *broadcast* (ALFA Instrumentos, 2000).

Quando se tratando de comunicação serial, mestre e escravos são fixos, entretanto em outros tipos de rede, os dispositivos podem assumir ambos os papéis, não simultaneamente (Seixas, 2007).

Na Figura 5, é possível observar o ciclo de pergunta-resposta da comunicação mestre-escravo.

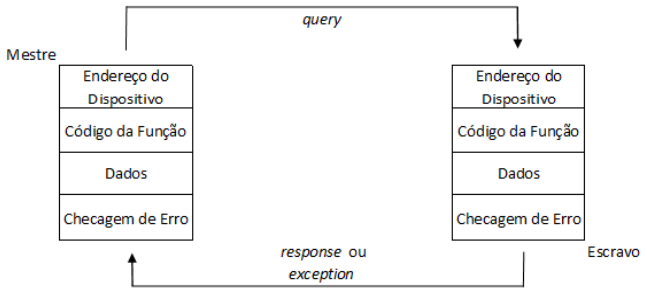

Figura 5. Ciclo de Pergunta-Resposta.

Fonte: WEG , 2005.

Endereço – a faixa de endereço válido vaio de 0 a 247, sendo que os endereços vão de 1 a 247, pois 0 é reservado ao *broadcast*

Código de função – vai de 1 a 255, porém são utilizados de 1 a 127 pois o bit mais significativo é usado para resposta de exceção.

Dados – variam de acordo com código da função e o papel da mensagem, requisição ou resposta, ou mesmo ser um campo vazio.

Checagem de erro – contém valor de 8 ou 16, dependendo do modo de transmissão.

Modo de Transmissão Modbus

Há dois modos de transmissão, modo ASCII (*American Standard Code for Information Interchange*) e modo RTU (*Remote Terminal Unit).*

- Modo ASCII – Neste modo, cada byte de mensagem é enviado como dois caracteres ASCII. São permitidos intervalos de até segundo são permitidos sem que a mensagem seja truncada. Em alguns casos tais intervalos de silêncio são usados como delimitadores de fim de mensagem. É formado por:

10 bits por byte;
1 start bit;
7 bits de dados LSb enviado primeiro;
1 bit de paridade (par/ímpar) + 1stop bit;
0 bit de paridade + 2 stop bits;

No campo de checagem de erros é utilizado LRC (*Longitudinal Redudancy Check*). (Elipse Software, 2015)

- Modo RTU – neste modo é enviado um caractere no padrão hexadecimal em cada palavra mensagem. Sua vantagem em relação ao modo ASCII é possui maior densidade de dados em uma mesma mensagem, melhorando o desempenho da comunicação. A mensagem deve ser transmitida de maneira contínua, pois uma pause de 1,5 caractere trunca a mensagem. Sua é mensagem formada por:

11 bits por byte;
1 start bit;
8 bits de dados;
1 bit de paridade + 1 stop bit ou;
0 bit paridade + 2 stop bits.

O campo de checagem de erros é utilizado CRC (*Cyclical Redundancy Check*). (Elipse Software, 2015)

Variáveis Analógicas

São sinais contínuos que podem variar com o tempo, geralmente são advindos de grandezas físicas como pressão, temperatura entre outras. Por exemplo, um sensor de pressão gera um sinal de 0 – 10V, este sinal pode ter qualquer valor de tensão entre o intervalo e por serem contínuos sempre irá apresentar algum nível de tensão (PLC Academy, 2018).

Quando este tipo de sinal entra em um dispositivo como um CLP (Controlador Lógico Programável), passa através de um conversor A/D ou conversor analógico/digital. Que por sua vez, converte este sinal em sinais digitais de acordo com a resolução em bits (PLC Academy, 2018).

O Projeto

Foi desenvolvido na IDE do Arduino, um software capaz de se comunicar com a plataforma Elipse SCADA através da rede Modbus RTU e que através das portas digitais do Arduino, ser capaz de controlar o inversor de frequência. O controle e ajuste de velocidade é feito através do Elipse SCADA, ficando a encargo do Arduino executar a parte lógica do sistema.

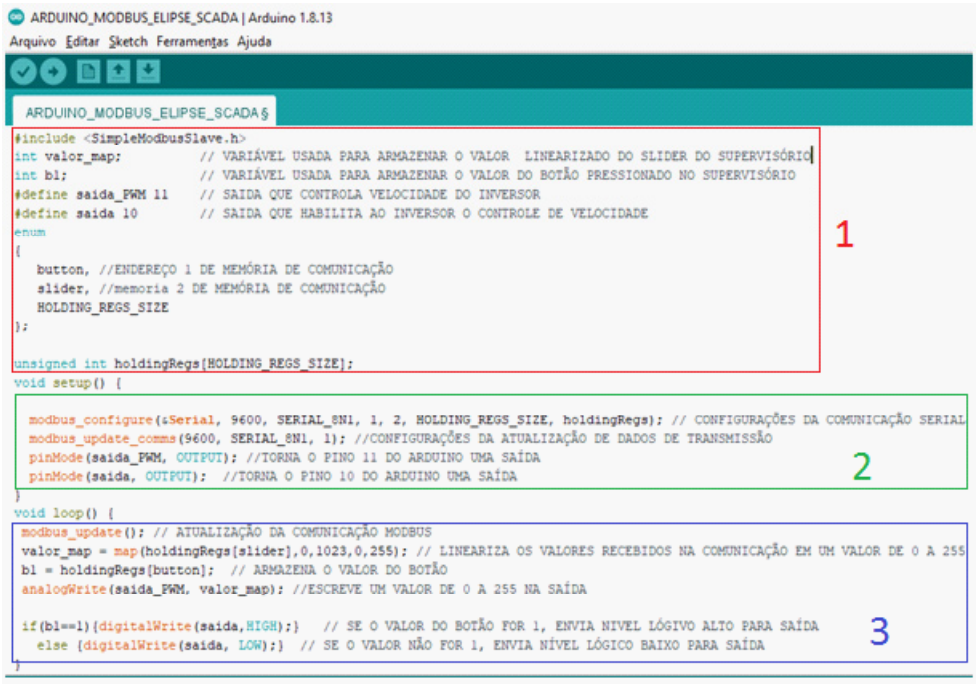

Figura 6. Software Presente no Arduino.

Fonte: Autores, 2020.

De acordo com a Figura 6, o software foi dividido da seguinte forma:

1 – Nesta área são declaradas as variáveis e bibliotecas utilizadas no software. Na criação deste software foi necessário utilizar uma biblioteca. Elas fornecem funcionalidades extras que podem ser para trabalhar com hardwares ou manipular dados, várias dessas bibliotecas já vem pré-instalada na IDE do Arduino, mas também podem ser baixadas ou criadas (Arduino.cc, 2018). No caso do software criado, esta biblioteca foi baixada e a mesma é responsável por fazer o controle da comunicação Modbus.

2 – Void setup - Esta área é reservada para configurações feitas no software, nela estão as configurações da rede Modbus e a configuração dos pinos do Arduino, indicando que serão usados como saídas.

3 – Void loop – Onde de fato foi escrita a lógica do software. Possui um comando com atualização da rede Modbus, linearização do sinal do *slider* advindo do supervisório, pois o mesmo varia entre 0 e 1023 e é necessário a conversão para um valor de 0 a 255 que é a resolução da saída PWM (Modulação por Lagura de Pulso), do Arduino. Possui também a lógica do botão responsável por habilitar o inversor.

Na Figura 7 é possível ver o hardware do Arduino utilizado para simulação, onde o led verde, conectado ao pino 11, é usado para simular o sinal PWM, ou seja, de acordo com a modulação do *slider* presente no supervisório, a luminosidade do led verde irá variar. Já o led azul, conectado ao pino 10, será usado para simular o botão habilita, que ao ser pressionado acenderá o led azul. O Arduino será conectado ao computador através de um cabo USB.

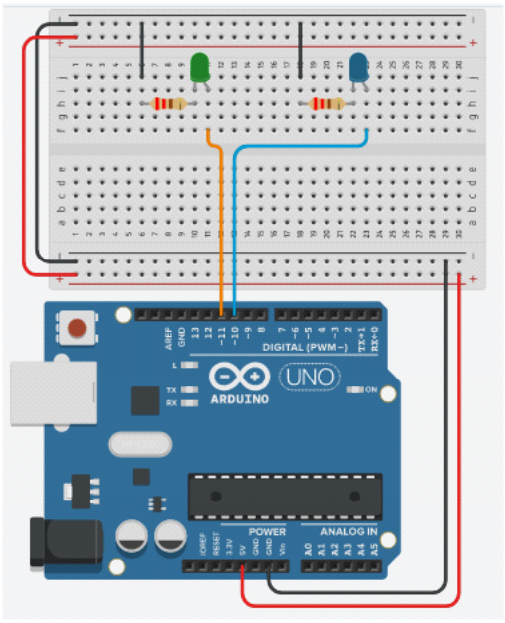

Figura 7. Hardware Arduino.

Fonte: Autores, 2020.

Revista Teccen. 2021 Jul./Dez.; 14 (2): 28-35

DOI 10.21727/rt.v14i2.3057

Tela do Supervisório

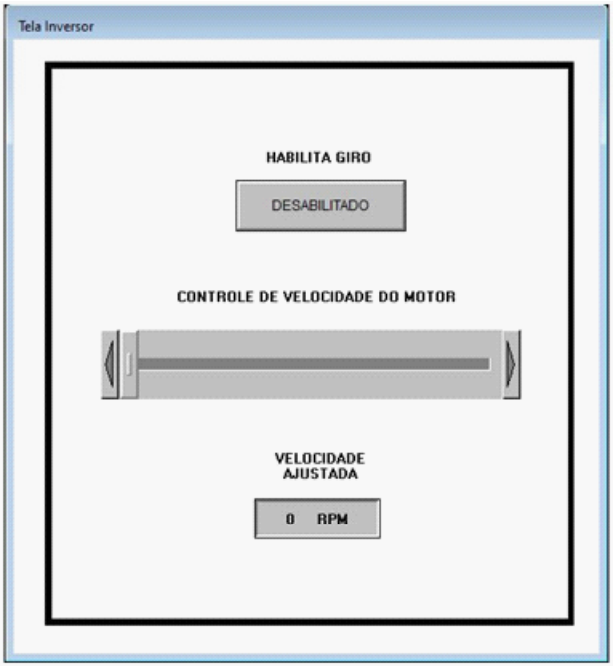

Figura 8. Tela Supervisório.

Fonte: Autores, 2020.

Na Figura 8 é demonstrada a tela do supervisório, área em que são comandadas todas as operações do inversor. Nela é possível identificar o botão responsável por habilitar e desabilitar o inversor, o *slider*, que é responsável por fazer o ajuste de velocidade e um display utilizado para visualizar a velocidade ajustada.

Para que supervisório fosse capaz de se comunicar com o Arduino, foi necessária a utilização do driver Modbus Master, ele está disponível no site da Elipse Software.

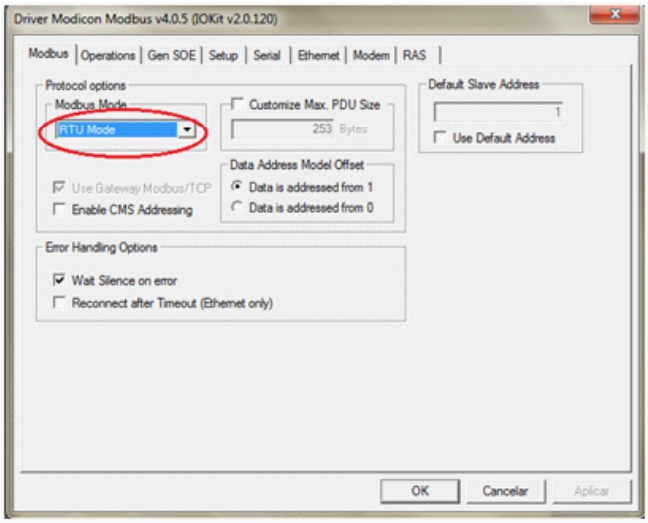

Figura 9. Configuração Driver Modbus.

Fonte: Autores, 2020.

Ao inserir o Driver na aplicação foi necessário configurá-lo, a Figura 9 mostra a tela de configurações extras do driver. Na aba Modbus foi necessário deixar a opção *Modbus Mode* em "RTU *Mode"*, ou seja, funcionar no modo RTU, as outras opções desta aba não foram alteradas.

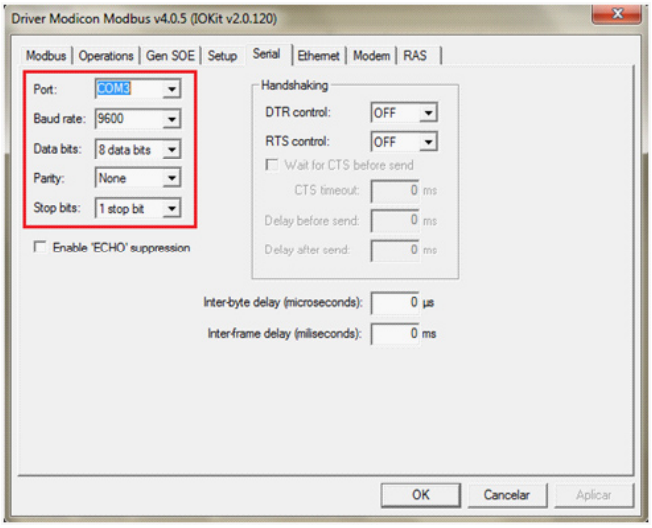

Figura 10. Tela Serial.

Fonte: Autores, 2020.

A próxima aba configurada, foi a aba Serial, em que se alterou a porta em que o Arduino foi conectado, também foi configurado velocidade, *Data bits, Parity* e *Stop bits.*

Após inserir e configurar o driver, foi adicionado os *tags* de comunicação, responsáveis pela troca de dados entre o supervisório e o Arduino. Foram criados 2 *tags,* um para o botão e outro para o *slider* e *display*. Para configuração dos *tags* foi utilizado o manual que vem acompanhado o driver.

Feita todas as configurações e associações dos *tags* aos itens na tela, o supervisório ficou pronto para ser executado.

Com o *software* do Arduino e supervisório prontos, foi utilizado leds para simular o inversor de frequência, durante os testes. Após os testes foi possível ver a variação de luminosidade do led, ou seja, estando conectado ao inversor, esta variação controlaria a velocidade de um motor conectado a ele.

Para conectar o Arduino ao inversor é preciso inserir uma placa conversora capaz de converter um sinal de 0 a 5 Volts, que é o sinal que o Arduino trabalha, em um sinal de 0 a 10 Volts, sinal de trabalho do inversor. A não utilização deste equipamento faz com que o inversor não chegue a velocidade de máxima regulada.

Conclusão

O sistema criado se mostrou muito eficiente, mesmo sem a utilização de um inversor de frequência conectado, alcançando o objetivo de ser uma programação simples e de baixo custo. A tela de controle do supervisório, apesar de simples, cumpre bem o principal objetivo de variar a velocidade do inversor, que para fins de simulação, foi utilizado um led com sua luminosidade sendo variada.

Para que o sistema cumpra com eficiência seu papel, deve-se conectar o Arduino ao inversor utilizando um conversor de sinal de 0 a 5 Volts, para um sinal de 0 a 10 Volts, desta forma o sistema funcionará perfeitamente.

Sendo assim, pode-se afirmar que o objetivo foi concluído com êxito, pois foi criada uma alternativa de fácil utilização e implementação, além de ser uma ótima opção em relação ao seu custo.

Referências

ALFA Instrumentos. (2000). *Protocolo de Comunicação Modbus RTU/ ASCII.* Recuperado de https://www.dca.ufrn.br/~affonso/FTP/DCA447/modbus/modbus_manual.pdf. Acessado em 5 de Dezembro de 2020.

Arduino.cc. (2018). *Arduino cc*. Retirado de Arduino cc: https://www.Arduino.cc/reference/en/libraries/. Acessado em 30 de Novembro de 2020.

Elipse Software. (2015). *Elipse Scada Manual do Usuário.*

Ferroni, E. H., Vieira, H. R., Nogueira, J. H., Santos, R. K., Lemos, R. M., & Rodrigues, T. B. (2014). *A Plataforma Arduíno e Suas Aplicações.* UNIS-MG. Recuperado de https://revistas.rcaap.pt/uiips/article/download/14354/10740. Acessado em 28 de Novembro de 2020.

Franchi, C. M. (2008). *Acionamentos Elétricos.* São Paulo: Érica Ltda.

McRoberts, M. (2010). *Beginning Arduino*. Recuperado de https://www.elecrow.com/download/[Beginning.Arduino].Michael.McRoberts.pdf. Acessado em 30 novembro de 2020

PLC Academy. (Março de 2018). *All About PLC Analog Input and Output Signals and Programming*. Recuperado de PLC Academy: https://www.plcacademy.com/plc-analog-input-output/ Acessado em 8 de Dezembro de 2020.

Seixas, C. (200?). *Protocolos Orientados a Caracter.* UFMG - Departamento de Engenharia Eletrônica. Retirado de https://www.dca.ufrn.br/~affonso/FTP/DCA447/modbus/ProtocolosCaracter.pdf. Acessado em 6 de Dezembro de 2020

Souza, L. C. (1999). *Desenvolvimento de um Driver Padrão OPC para Controlador Lógico Programável, em Ambiente Windows NT.* Belo Horizonte. Retiarado de https://www.ppgee.ufmg.br/defesas/270M.PDF. Acessado em 7 de Dezembro de 2020.

WEG . (2005). *Guia de Aplicação de Inversores de Frequência.* Santa Catarina.