

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA DE ENGENHARIA / ENGENHARIA ELÉTRICA
Trabalho Final de Curso II

Antonio Carlos da Rocha Júnior
Nilo Pereira de Araújo

**PROTÓTIPO PARA MONITORAMENTO DOS PARÂMETROS DE
TRANSFORMADORES POR INTERNET DAS COISAS – IoT**

Trabalho Final de Curso II como parte dos requisitos para obtenção do título de bacharel em Engenharia Elétrica apresentado à Pontifícia Universidade Católica de Goiás

BANCA EXAMINADORA:

Prof. MSc. Carlos Alberto Vasconcelos Bezerra – Orientador. ENG-PUC Goiás

Prof. MSc. Carlos Daniel de Souza Bezerra – IFG.

Prof. MSc. Luis Fernando Pagotti – ENG-PUC Goiás.

Goiânia, 23 de novembro de 2020.

PROTÓTIPO PARA MONITORAMENTO DOS PARÂMETROS DE TRANSFORMADORES POR INTERNET DAS COISAS – IoT

Antonio Carlos da Rocha Júnior, Nilo Pereira de Araújo, Carlos A. V. Bezerra

Abstract - This course conclusion paper presents the development of a prototype for monitoring the parameters of transformers through IoT technologies. This technology includes possibilities for integrating sensors and actuators via the Internet. The use of narrowband, with NB-IoT protocol, allows computers, sensors and actuators in general, to interact with each other and process information and data. Communication via Wi – Fi is another possibility for integrating these technologies. These two technologies are addressed in this work, but only communication via Wi-Fi was developed in the final prototype, because the narrowband connectivity (NB-IoT) with the mobile phone operators, is not yet accessible until the moment of completion of this job. For the development of the prototype, a study of the technologies involved and computational simulations were carried out with the purpose of making its project feasible. The results presented and discussed allow us to glimpse the great potential for application of the developed system.

Keywords – NB-IoT, transformers, Wi-Fi.

Resumo - Este trabalho de conclusão de curso apresenta o desenvolvimento de um protótipo para monitoramento dos parâmetros de transformadores através de tecnologias de internet das coisas - IoT. Essa tecnologia abrange possibilidades de integração de sensores e atuadores via Internet. O uso da banda estreita, com protocolo NB-IoT, possibilita que os computadores, os sensores e atuadores em geral, interajam entre si e processem informações e dados. A comunicação através do Wi-Fi é outra possibilidade de integração destas tecnologias. Estas duas tecnologias são abordadas neste trabalho, porém somente a comunicação via Wi-Fi foi desenvolvida no protótipo final, pois a conectividade da banda estreita (NB-IoT) junto às operadoras de telefonia móvel, não está ainda acessível até o momento de conclusão deste trabalho. Para o desenvolvimento do protótipo foi realizado um estudo das tecnologias envolvidas e simulações computacionais realizadas com o propósito de viabilizar o seu projeto. Os resultados apresentados

e discutidos permitem vislumbrar o grande potencial de aplicação do sistema desenvolvido.

Palavras-chaves: NB-IoT, Transformadores, Wi-Fi.

1. Introdução

A energia elétrica está presente na vida de grande parte da população mundial, sua origem e manipulação englobam as etapas de geração, transmissão e consumo. A geração de energia brasileira possui uma matriz baseada em energia hidráulica sendo que essa, majoritariamente composta por hidroelétricas, em torno de 71,6%, correspondente do total [1].

Após a geração, a eletricidade é transmitida aos centros consumidores. Os pontos de geração normalmente estão longe dos consumidores finais e então uma linha de transmissão que utiliza diversos equipamentos é usada, além de uma boa área construída. Por fatores econômicos o metal utilizado nos condutores precisa ser reduzido a fim de diminuir seu peso, reduzindo os esforços mecânicos e o custo total do material. Em contrapartida isso é feito elevando-se a tensão para que a corrente nos condutores seja reduzida o máximo possível, mantendo a eficiência do sistema. Um, dentre todos equipamentos utilizados se destaca: o transformador. Ele é o responsável por manipular os níveis de tensão tornando possível o melhor aproveitamento de material na construção de linhas de transmissão. O transformador é um componente dentre muitos nos sistemas de conversão de energia, embora não desempenhe esse papel. Sua utilização torna mais econômica a manipulação e geração de energia elétrica [2].

Dentre todos os testes possíveis de serem feitos, abordaremos uma nova forma de monitoramento de grandezas competentes ao funcionamento do equipamento. Essa forma consiste no envio de dados pela tecnologia IoT, a chamada internet das coisas.

A internet das coisas automatiza alguns setores da economia e da vida social. Como logística, transporte de pessoas, agricultura, saúde, produção industrial

dentre outros. Dados recentes da União Internacional de Telecomunicações (UIT) mostram que 95% da população global já vivem em áreas cobertas com rede celular (2G ou mais) e 84% têm acesso à banda larga móvel. Soma-se a isso o rápido crescimento das redes 4G, tecnologia já acessível para 4 bilhões de pessoas [3].

Existem fortes divergências em relação ao conceito de IoT¹, não havendo, portanto, um conceito único que possa ser considerado pacífico ou unânime. Um dos pontos em comum das diversas definições de IoT é que elas se concentram em como computadores, sensores e objetos interagem uns com os outros e processam informações/dados em um contexto de hiper-conectividade [3].

Uma rede IoT pode ser usada para monitorar e diagnosticar equipamentos em campo, em qualquer lugar onde chegue conexão com a internet.

O protótipo aqui proposto visa o desenvolvido de um sistema que monitore transformadores de potência e que retire as principais variáveis competentes ao seu funcionamento e envie estas variáveis para uma página da Internet. Para esta finalidade será monitoradas as variáveis tensão, corrente e temperatura de um transformador, utilizando-se de sensores apropriados e a tecnologia de Wi-Fi será responsável pela transmissão dos dados.

Nas seções seguintes serão detalhadas as etapas de construção do protótipo e aplicações do mesmo. Por fim, são descritos os resultados e as conclusões gerais sobre o trabalho.

2. Fundamentação Teórica

A internet das coisas (IoT), são tecnologias emergentes que modificará a forma como os sensores e atuadores em um sistema irão se comunicar. Estas tecnologias ficam a cada dia, mais acessíveis e se tornam importantes nas conexões entre equipamentos e processos. O estudo da conectividade para estes sistemas nas redes das operadoras de telefonia móvel e o desenvolvimento de Hardware para conectar equipamentos e processos à internet é fundamental para a formação do Engenheiro Eletricista na atualidade. A Internet das coisas é uma rede de tecnologias interligadas baseadas em protocolos de comunicação. [5].

O Sistema empregado neste trabalho visa abordar uma opção de transporte de dados coletados em campo através de um protocolo de IoT para a internet, de forma que possam ser acessadas diretamente da mesma.

As seções seguintes listam os conceitos fundamentais para o desenvolvimento do protótipo proposto neste trabalho.

2.1. Tecnologia NB-IoT

O organismo de normas internacionais (3GPP – Resolução 13 LTE Advanced Pro, em junho 2016), normatizaram o NB-IoT (Narrow Band IoT) em meados de 2016, fixando frequências licenciadas (algumas frequências do 4G) permitindo que algumas estações compartilhem sua infraestrutura [4].

A rede faz a troca de dados entre participantes em seu domínio funcional, dentro do sistema e entre sistemas. A troca de dados pode incluir atualizações de sensores, dados de telemetria, eventos, alarmes, logs, arquivos, mudanças de estado, comando de controle, e atualização de configurações.

A figura 1 ilustra um diagrama de blocos para um sistema em que o protocolo NB-IoT é utilizado. O Sim 7020E é um circuito dedicado a transmitir informações por NB-IoT em banda estreita, fazendo a conectividade com a operadora de telefonia móvel. Este módulo possui uma quantidade de entradas analógicas limitadas e, portanto para empregá-lo em um sistema maior se faz necessário uso do Arduino. A conexão com o Arduino se estabelece pela porta serial, fazendo uso dos pinos RX e TX do Hardware de ambos.

O SIM 7020E é um modem NB IoT que permite o acesso a banda estreita da operadora. No caso do Brasil a banda utilizada é a Banda 28. Este módulo pode ser programado utilizando os comandos AT de modem e a conectividade com a operadora se faz através de um contrato, onde a mesma irá fornecer os chips NB-IoT necessários.

¹ (IoT – Do inglês: Internet of Things)

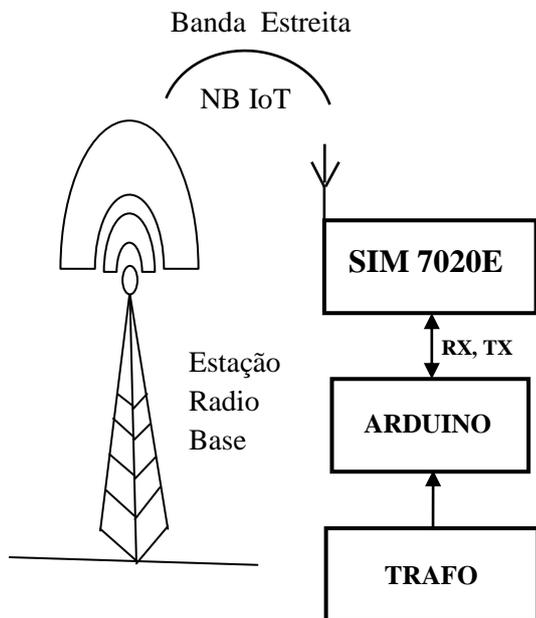


Figura 1 – Diagrama de Blocos NB-IoT

A possibilidade do uso de NB-IoT no trabalho desenvolvido foi descartada assim que as operadoras de telefonia móvel não deram o suporte para a conectividade com o SIM 7020E.

2.2. Tecnologia Wi-Fi

Wi-Fi é uma tecnologia de rede sem fio que permite que computadores, dispositivos móveis e outros equipamentos se conectem à Internet. O Wi-Fi permite que esses e muitos outros dispositivos troquem informações entre si, criando uma rede.

Um sistema Wi-Fi pode estabelecer a conectividade com a internet através de uma rede particular de acordo com o diagrama de blocos ilustrado na Figura 2. O circuito ESP 8266-12E é um microcontrolador com capacidade de conexão Wi-Fi, não necessitando de nenhum módulo externo para conectar com redes Wireless.

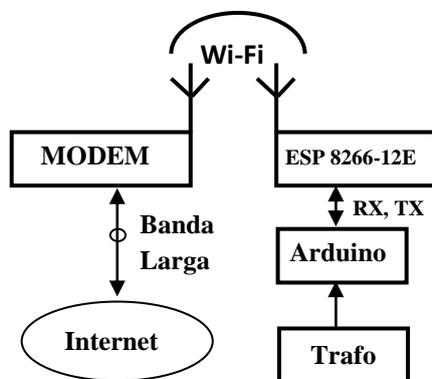


Figura 2 – Diagrama de Blocos WI-FI

A sua programação se dá através de comandos AT e pode ser desenvolvida utilizando a interface de programação do Arduino (IDE).

O Arduino é necessário quando for necessário aumentar a capacidade de entradas e saídas do sistema. A comunicação com o Arduino se estabelece através da comunicação serial RX, TX do Hardware de ambos.

2.3. Transformador

Um transformador é composto basicamente pelo acoplamento de espiras em meio a um núcleo metálico. Logo para seu funcionamento dentro dos níveis de tensões e correntes requeridos aos sistemas de potência no mundo real uma série de partes foi adicionada.

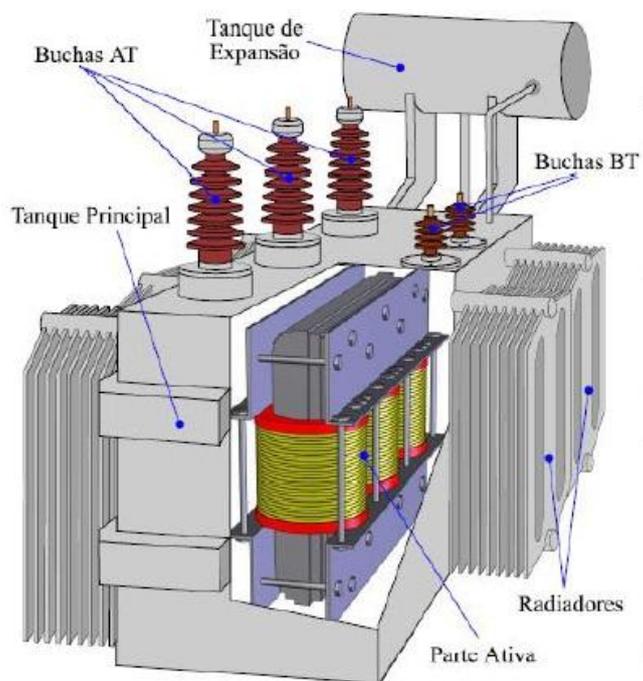


Figura 3 – Transformador e suas partes [7]

A figura 3 esquematiza um transformador de potência, normalmente para uso em potência elevada, são trifásicos e presentes em subestações de energia elétrica. A energia elétrica o atravessa pelos contatos acima das buchas de baixa tensão (BT) e alta tensão (AT), obviamente as buchas AT isolam os contatos que recebem o maior potencial do Trafo. Em funcionamento a Parte Ativa, onde estão os enrolamentos e núcleo magnético aquecem muito e precisam ser refrigerados. Dentre os tipos de refrigeração, a utilização de óleo no interior do transformador é a mais utilizada em transformadores para grandes tensões e potências. Este óleo perde calor para o meio através das superfícies de contato

dos radiadores por onde ele passa, em casos de extremo aquecimento um sistema de ventilação forçada sobre as aletas do radiador deve ser instalado. O fluido circula entre tanque principal, radiador e tanque de expansão perdendo calor para meio de acordo com a figura 4.

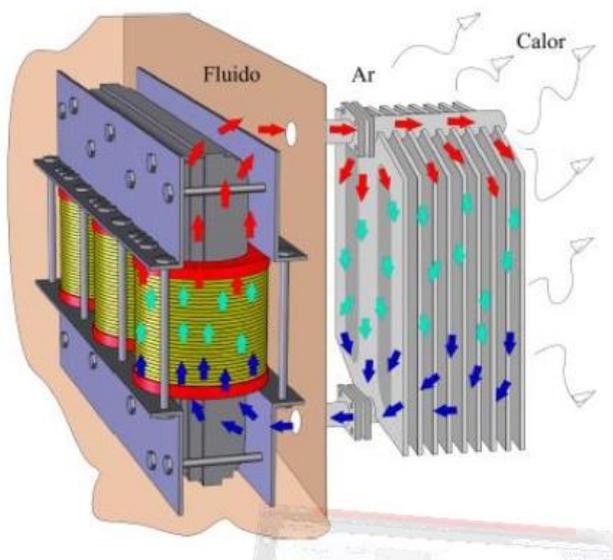


Figura 4 – Sistema de refrigeração de um Transformador [7]

Toda superfície de contato dos radiadores é pensada de forma que emita o máximo de calor possível a fim de estabilizar a temperatura do fluido e assim as condições de funcionamento do equipamento como um todo. Altas temperaturas podem corromper a isolamento das bobinas entre si e umas com as outras. Essas alterações podem modificar os parâmetros do transformador descaracterizando seu funcionamento.

Normas técnicas aplicáveis:

Dentre as principais normas técnicas competentes à manipulação de transformadores de potência estão:

ABNT NBR 5356-1,2,3,4 e 5 :2007 Transformadores de Potência

CELG/ANEEL - NTC-58- Transformadores de Potência Secos Especificação e Padronização

NBR 10295/1988 – Transformadores de potências secos

NBR 12454/1990 Transformadores de potência de tensões máximas até 36,2 kV e potência de 225 kVA até 3750 kVA

ANSI/IEEE C57.12.00/2006 – Requisitos gerais para distribuição, energia e transformadores imersos em líquido

IEC 60076/2008 – Transformadores de Potência.

2.4.Arduino

O Arduino, ilustrado na figura 5, é uma plataforma de hardware e software abertos projetados para facilitar o processo de automação. Pode ser modificado de acordo com cada projeto. Possui pinos digitais que podem ser programados como entradas ou saídas. A conexão com sensores analógicos pode ser realizada pelas entradas analógicas, que são conversores A/D de 10 bits. A linguagem de programação é derivada do C/C++ e a programação é realizada pela IDE (Integrated Development, ou Ambiente de Desenvolvimento Integrado). Existem diversas possibilidades de utilização, vários sensores e componentes compatíveis. Neste trabalho serão utilizados sensores de tensão, corrente e temperatura.



Figura 5 – Arduino Uno [8]

Na programação do Arduino, existem alguns comandos mais elaborados, onde sua utilização necessita da inclusão de uma biblioteca na linha de programação.

Uma biblioteca é definida como um conjunto de instruções desenvolvidas para executar tarefas específicas relacionadas a um determinado dispositivo [9]. Várias bibliotecas da Adafruit² e Sparkfun³, podem ser acessadas pelo GitHub e utilizadas para programação da plataforma Arduino.

Na proposta de trabalho a ser desenvolvido, as seguintes bibliotecas serão utilizadas, com a finalidade de estabelecer os programas para o Arduino e o NODE MCU 8266-12E: EmonLib.h; LiquidCrystal.h;

² <https://www.adafruit.com>

³ <https://www.sparkfun.com>

max6675.h; ESP8266WiFi.h; ESP8266WebServer.h [10].

Estas bibliotecas facilitam o processo de programação, pois são específicas para cada situação encontrada no ambiente de aplicação dos dispositivos eletrônicos envolvidos no trabalho.

2.4.1. Bibliotecas do Arduino utilizadas no Trabalho

OpenEnergyMonitor: É uma biblioteca de monitoramento de eletricidade, com funções de cálculo e monitoramento de grandezas elétricas como tensão, corrente, potência ativa, Potência aparente e Fator de potência fornecida gratuitamente pela comunidade Github [10]. As funções desta biblioteca possuem aplicações relacionadas com o desenvolvimento deste projeto, sua utilização é de extrema importância e facilita a programação do Arduino.

LiquidCrystal.h: Esta biblioteca permite o controle de display LCD baseado no chip de controle HD44780 ou compatível, que é encontrado na maioria dos LCDs baseados em texto. A biblioteca funciona no modo de 4 ou 8 bits (ou seja, usando 4 ou 8 linhas de dados além das linhas de controle rs, enable, opcionalmente as linhas de controle RW).

Esta biblioteca possui várias funções para o controle de escrita em LCDs.

Max6675.h: Esta biblioteca realiza a leitura das informações geradas no circuito integrado (CI) MAX6675 que digitaliza o sinal de um termopar tipoK com resolução de 12 bits e faz a compensação da junta fria. A comunicação é realizada com o protocolo SPI.

ESP8266WiFi.h: A biblioteca ESP8266WiFi é uma adaptação completa (uma porta) da biblioteca WiFi para Arduino desenvolvida pela espressif⁴ (Fabricante do módulo ESP8266).

ESP8266WebServer.h: Esta biblioteca configura servidores HTTP para módulos Wi-Fi ESP8266/ESP32.

Algumas funções das bibliotecas citadas estão listadas nos anexos 2, 3, 4, 5 e 6.

2.5. Monitoração de Tensão e Corrente

Para monitoração da tensão e corrente do transformador, foram utilizados sensores que pudessem ser acoplados nas entradas analógicas do Arduino UNO que operam em 10 bits e tensão variando de 0 a 5V. A figura 6 apresenta um diagrama de blocos para medições de tensão e corrente no trabalho proposto.

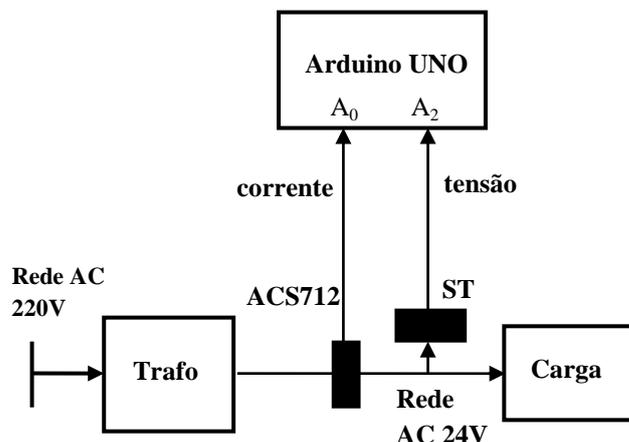


Figura 6 – Diagrama de blocos do sensor de corrente

Neste diagrama observa-se a ligação dos sensores de corrente (ACS712) e tensão (ST) com as entradas analógicas do Arduino A₀ e A₂. As seções a seguir detalham os sensores de corrente e tensão utilizados no trabalho

2.5.1. Sensor de Corrente

O sensor de corrente adotado é um circuito eletrônico contendo o circuito integrado ACS712-30A, que através do efeito Hall⁵ mede corrente de até 30A e fornece uma tensão elétrica na faixa de 0 a 5V, proporcional a corrente do circuito. Esta tensão está deslocada da origem com um Off-Set de 2,5 V, portanto ideal para a medição em corrente alternada com a utilização de uma entrada analógica do Arduino. A figura 7 apresenta uma fotografia do sensor de efeito Hall ACS712-30A selecionado para as medições de correntes.

⁴ <https://www.espressif.com>

⁵ Esse fenômeno, descoberto em 1879 por Edwin Herbert Hall

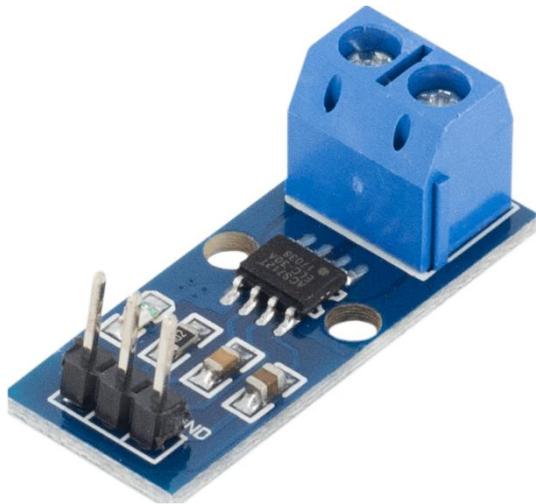


Figura 7 – ACS712-30

2.5.2. Sensor de Tensão

O sensor de tensão empregado foi projetado para atender especificamente o objetivo do trabalho. Utiliza-se de um divisor de tensão para pegar uma amostra da tensão da rede. Esta tensão é deslocada de 2,5V para atender os requisitos da entrada analógica do Arduino. A figura 8 apresenta o diagrama eletrônico do sensor.

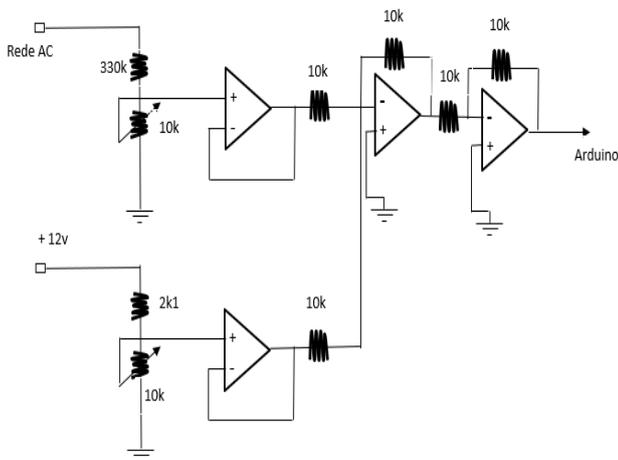


Figura 8 – Sensor de tensão

Foi empregado amplificadores Operacionais (AOs) para elaboração do mesmo.

2.6. Monitoração de temperatura

O sensor de temperatura utilizado no trabalho é um termopar tipo K acoplado em um circuito integrado MAX 6675. Este CI é um conversor serial de resolução de 12-bits para termopar tipo K, realiza compensação de junção fria, correção de linearidade, detecção de defeito no termopar. Utiliza a comunicação SPI de três fios, para enviar os dados

para o Arduino. A figura 9 ilustra o sensor MAX 6675.



Figura 9 – Sensor de temperatura

2.7. Conexão com a Internet

Para realizar a conexão com a Internet, inicialmente foi pensado em um módulo eletrônico com o protocolo NB IoT. Depois de algumas pesquisas foi optado pelo Módulo SIM 7020E, pois o mesmo atenderia aos requisitos da Banda Estreita utilizada no Brasil (Banda 28). Com a impossibilidade do acesso à Banda Estreita, optou-se por um módulo que pudesse ser conectado à rede Wi-Fi. Assim foi utilizado o módulo MCU 8266-12E. Este módulo possibilitou a transmissão das variáveis do transformador, caracterizando bem o que foi proposto no trabalho. As seções a seguir detalham informações desses módulos.

2.7.1. Módulo SIM 7020E

O sim7020E é uma solução multi-banda de módulo NB-Iot. É projetado para aplicações que precisam de baixa latência, baixa transmissão de dados em uma variedade de condições de propagação de rádio. Devido à combinação única de desempenho, segurança e flexibilidade, este módulo é ideal para aplicações M2M, como medição, rastreamento de ativos, monitoramento remoto etc. Com o modo de economia de energia (psm) e recepção descontínua prolongada (edrx), o sim7020E pode prolongar a vida útil da bateria para 10 anos. O sim7020E fornece uma cobertura mais profunda em comparação com GSM. A figura 10 apresenta uma foto do módulo SIM 7020E. Nesta figura pode ser observado o conector para os cartões NB-IoT habilitados pelas operadoras de telefonia móveis.



Figura 10 – Sim 7020E⁶

2.7.2. Módulo MCU ESP8266-12E

O módulo ESP8266-12E é um chip compacto projetado para atender as necessidades de conectividade sem fio (padrão 802.11 B/G/N) podendo tanto ser utilizado independentemente ou em conjunto com um microcontrolador externo (Arduino, PIC, ARM, etc). Com este módulo é possível adicionar conectividade Wi-Fi (802.11) a qualquer sistema. Modos de operação AP e station (Access Point ou Estação). Possui protocolos da camada de rede internos (Stacks), não necessitando de biblioteca adicional ou firmware atualizável. A figura 12 ilustra este módulo com todos os pinos de conexões possíveis. A programação deste pode ser realizada através de comandos AT e utilizando a IDE do Arduino.

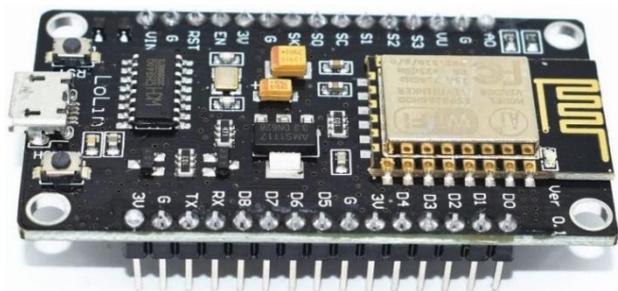


Figura 11 – Módulo ESP8266-12E⁷

O apêndice 4 ilustra o diagrama deste módulo com as conexões mínimas para sua operação.

O Anexo1 fornece o diagrama com os pinos do módulo MCU 8266-12E.

⁶ FONTE: <https://portuguese.alibaba.com/product>

⁷ FONTE: <https://athoselectronics.com/esp8266-o-que-e>

3. Metodologia

Com o intuito de monitorar as variáveis em um transformador, utilizando tecnologia de IoT, foi desenvolvido um protótipo com a capacidade de conexão a internet através da rede Wi-Fi e os dados transmitidos a uma página da Internet, sendo atualizados constantemente. O projeto foi dividido em duas etapas. Na primeira etapa foi realizado as simulações computacionais para ajustes práticos dos sistemas envolvidos. Na segunda etapa foi desenvolvido o Hardware e os softwares necessários aos sistemas.

3.1. Simulação do sistema

A simulação de partes do sistema proposto foi realizada no Software ISIS PROTEUS, do fabricante Labcenter (<https://www.labcenter.com/downloads/>). A figura 11, ilustra a conexão no Arduino UNO utilizado. Nesta figura é possível verificar a conexão dos sensores de tensão e corrente nas entradas analógicas A₀ e A₂ do Arduino. A conexão do sensor de temperatura é realizada de forma digital, utilizando os pinos 4, 5 e 6 do Arduino (SCK, CS e SO). O protocolo utilizado para esta comunicação é o SPI. As conexões do display LCD foram realizadas através dos pinos 7, 8, 9, 10,11e 12 nas sequências indicadas (RS, E, D4, D5, D6 e D7).

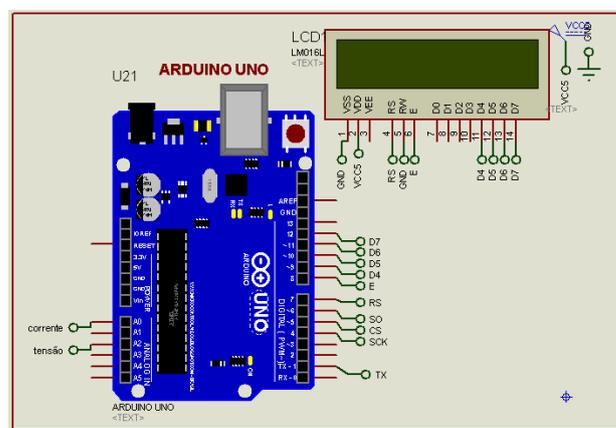


Figura 12 - Simulação: Conexões no Arduino

A figura 13 ilustra o esquema elétrico do sensor de temperatura que utiliza o Circuito Integrado (CI) MAX 6675. Este CI digitaliza as informações de temperatura realizadas com auxílio de um termopar tipo K (TCK), fazendo a correção da junta fria, e transmite esta informação através do protocolo SPI para o Arduino. A função readThermocouple() foi utilizada para realizar a leitura do sensor (Apêndice1), pois a biblioteca max6675.h, que poderia ser utilizada, possui falhas com o uso no Proteus.

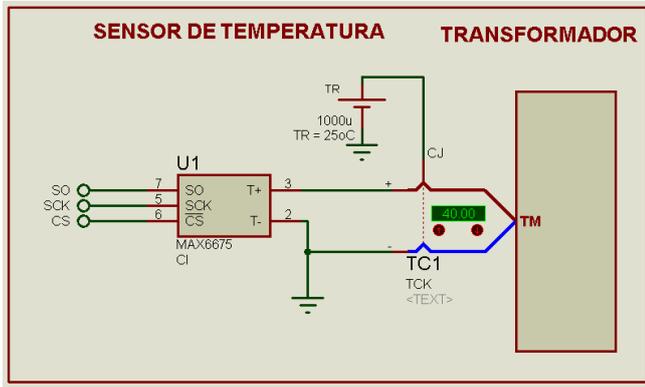


Figura 13 - Simulação: Sensor de temperatura

A figura 14 apresenta o sensor de tensão RMS utilizado na simulação. Foi utilizado por simplicidade um divisor de tensão com elevação da referência 2,5V DC. Isto porque as entradas analógicas do Arduino trabalham com tensões DC de 0 a 5V. Desta forma a referência AC do sinal de tensão fica deslocada de 2,5V DC, podendo o sinal AC ser ajustado para variar dentro desta faixa.

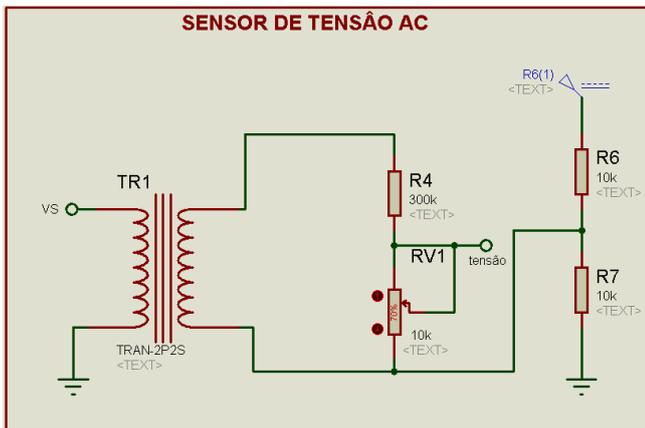


Figura 14 – Simulação: Sensor de tensão

Na figura 15, está indicado o esquema elétrico do sensor de corrente RMS, que utiliza o CI ACS 712-ELC30A da Allegro, conectado a um transformador com carga RL. Este sensor possui o princípio de medição baseado por efeito Hall e fornece uma tensão variando de 0 a 5V DC proporcional a corrente aplicada no mesmo (0 a $\pm 30A$). A tensão de saída está deslocada de 2,5V DC, sendo uma solução apropriada para medições de correntes AC através do Arduino.

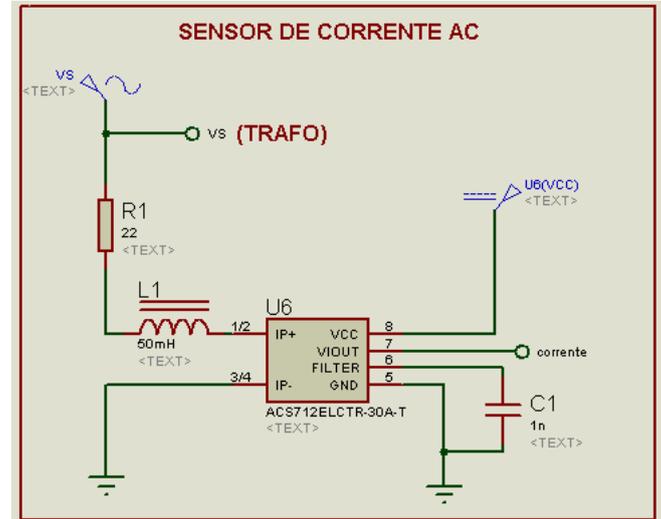


Figura 15 – Simulação: Sensor de corrente

O sistema apresentado foi utilizado para simulação das variáveis no Transformador e ajustes dos sensores para as medições e posterior uso destes no protótipo final. A programação do Arduino foi realizada com auxílio das bibliotecas EmonLib.h, LiquidCrystal.h e o programa final para simulação está indicado no apêndice 1.

A figura 16 ilustra a tensão de saída do sensor de tensão RMS. Neste é possível verificar o deslocamento da referência em 2,5V DC. O sinal AC está sobreposto e a biblioteca EmonLib.h, extrai apenas o valor AC para os cálculos das potências Ativa, Aparente e Fator de Potência no Transformador. Estas variáveis farão parte do pacote de dados enviados pela rede Wi-Fi para uma página configurada na Internet. O sinal de corrente do sensor ACS 712, possui as mesmas características do sinal do sensor de tensão, sendo tratado pela biblioteca EmonLib.h da mesma forma.

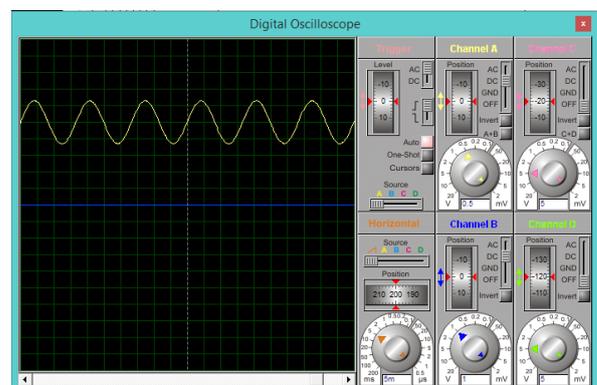


Figura 16 – Simulação: Saída do sensor de tensão

A saída do sensor de temperatura é um sinal digital com protocolo SPI, que é tratado pela função readThermocouple() na programação do Arduino anexo 1.

A figura 17 apresenta as variáveis medidas no transformador, através do terminal virtual do Proteus, para uma temperatura de 40 °C no transformador, tensão de 20V RMS no trafo e carga indutiva RL de 22Ω e 50mH.

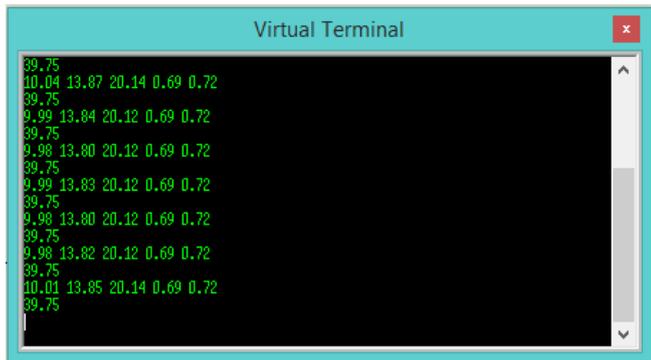


Figura 17 – Simulação: Variáveis medidas no trafo

Para a situação apresentada (últimas leituras), os valores das variáveis medidas estão ilustrados na tabela 1.

Tabela1 – Variáveis simuladas no transformador

Potência Ativa	10,01[W]
Potência Aparente	13,85 [VA]
Tensão RMS	20,14[V]
Corrente RMS	0,69[A]
Fator de Potência	0,72
Temperatura	39,75[°C]

Na figura 18, tem-se estas grandezas indicadas no Display LCD do sistema.

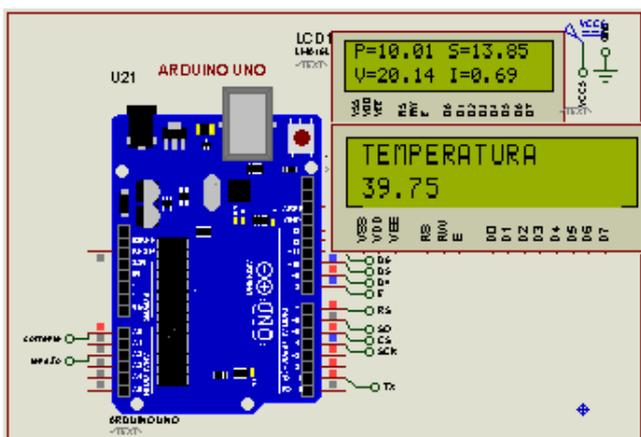


Figura 18 – Simulação: Variáveis indicadas no LCD

3.2. Montagem do protótipo

O Hardware do sistema foi montado em bancada para o estudo proposto no trabalho. A figura 19 apresenta o

diagrama de blocos do sistema proposto. Neste diagrama é possível identificar os sensores de tensão, corrente (ACS 712) e temperatura (MAX 6675). A conexão dos sensores de tensão e corrente são realizados através das entradas analógicas do Arduino A₀ e A₂. O sensor de temperatura comunica com o Arduino pelas portas digitais (D4, D5 e D6) através do protocolo SPI. A comunicação do Arduino com o NODE MCU 8266-12E é realizada pelas portas seriais de ambos. Como o Arduino trabalha com uma tensão de 5V e o NODE MCU com 3,3V foi colocado um conversor digital entre as portas RX e TX de ambos. Foi empregado um transformador de 220/24 V – 1A, alimentando uma carga resistiva com resistor de fio de 22Ω - 20W. Os sensores de corrente ACS 720-30A e temperatura MAX6675 foram adquiridos montados em placas de circuitos impressos e foram conectados ao sistema de modo que a corrente e temperatura pudessem ser monitoradas e estes conectados diretamente ao Arduino. O sensor de tensão foi necessário o desenvolvimento e montagem do circuito para a aplicação ao sistema.

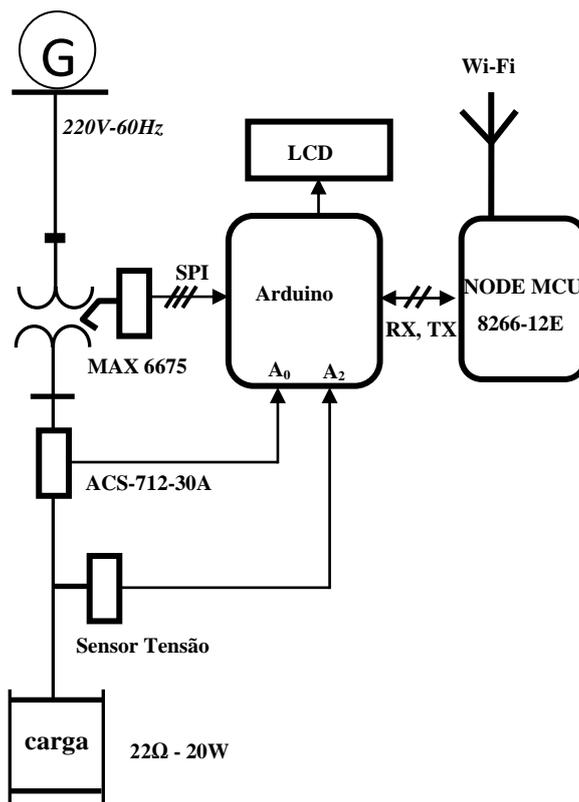


Figura 19 – Diagrama de blocos do protótipo

A figura 20 apresenta uma fotografia do Hardware desenvolvido para o estudo proposto. Nesta figura é possível verificar cada componente do sistema.

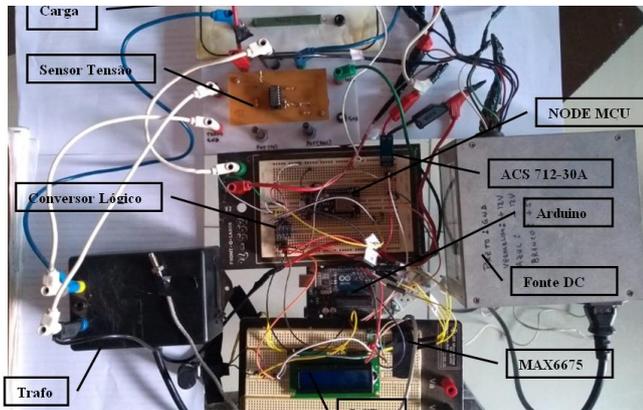


Figura 20 – Hardware do protótipo

4. Resultados e discussões

Foi realizada a programação do Arduino para realizar as leituras das variáveis no transformador de acordo com a programação utilizada na simulação. Foi acrescentada a função de comunicação com o NODE MCU.

Para desenvolver a programação do Arduino [Apêndice 2], foram utilizadas as bibliotecas: EmonLib.h [Anexo 2], LiquidCrystal.h [Anexo 3] e max6675.h [Anexo 4]. Não foi desenvolvido um protocolo específico para a comunicação entre o Arduino e o NODE MCU, a comunicação se dar através das portas seriais e o sincronismo realizado pelo reset das duas placas. Desta forma, a comunicação se estabelece e os dados das variáveis são transferidos para o NODE MCU.

A figura 21 apresenta os dados transferidos para o NODE MCU através do monitor serial da IDE do Arduino.

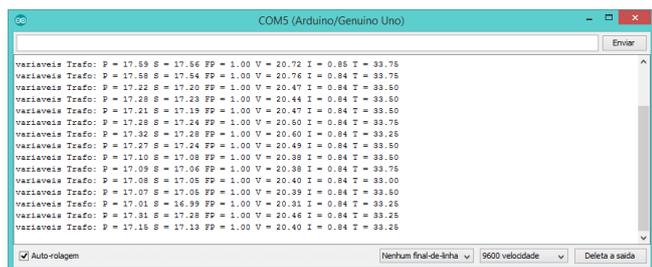


Figura 21 – Dados transferidos do Arduino para o NODE MCU8266-12E

No Hardware desenvolvido é possível visualizar em um Display LCD estas variáveis. A figura 22 ilustra as variáveis medidas através desta metodologia. No LCD estão os valores de P (Potência Ativa), S (Potência Aparente), V (Tensão) e I (Corrente) medidos no sistema.



Figura 22 - Variáveis indicadas no LCD

Na figura 23, tem-se o valor da temperatura no transformador.



Figura 23 - Temperatura do transformador indicada no LCD

Para conectar o sistema à internet, via Wi-Fi, desenvolveu-se uma programação para o NODE MCU 8266-12E e este está disponibilizado no Apêndice 3. Para esta programação foram utilizadas as bibliotecas ESP8266WiFi.h, ESP8266WebServer.h e SoftwareSerial.h, necessárias a estabelecer uma comunicação com a rede Wi Fi do sistema proposto.

Inicialmente o NODE MCU é conectado na rede Wi-Fi, utilizando as bibliotecas ESP8266WiFi.h e ESP8266WebServer.h que irá estabelecer a conexão e configurar o NODE MCU como um servidor na rede. Os dados são transmitidos do Arduino, por uma porta serial e chega até uma porta serial do NODE MCU que foram estabelecidas por Software através da biblioteca SoftwareSerial.h. A figura 24, ilustra as informações da conexão com a rede. O endereço IP: 192.168.100.17 é o endereço do NODE MCU na rede.



Figura 24 – Conexão com a rede Wi-Fi

Uma página de internet foi configurada na programação do NOD MCU, para receber as variáveis do transformador. Com o endereço IP digitado em um navegador de internet o acesso à página é realizado. As variáveis são atualizadas a cada 2 segundos. A figura 25 ilustra a página criada para receber as variáveis medidas

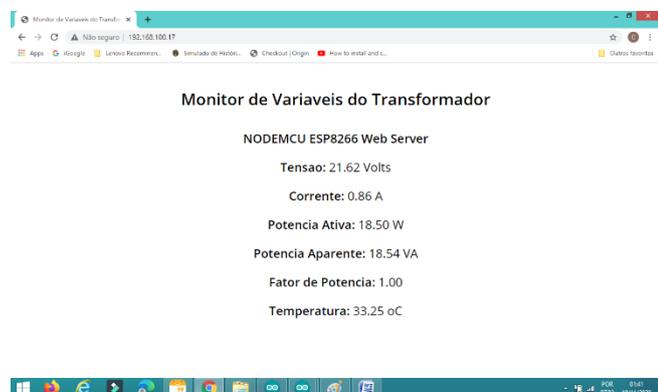


Figura 25 – Página com as variáveis do transformador

5. Conclusões

Este trabalho apresenta a proposta de elaboração de um protótipo para monitoramento dos parâmetros de transformadores por internet das coisas – IoT. Foram estudadas as tecnologias disponíveis para a transmissão dos parâmetros por IoT e decidido pelo uso daquela que estava disponível no momento, ou seja a transmissão via Wi-Fi com a conectividade utilizando o módulo MCU 8266-12E. Com a tecnologia de IoT definida foi realizado simulações computacionais que serviram de apoio ao desenvolvimento do protótipo. Em um segundo momento foi projetado a estrutura que o sistema deveria apresentar. Os sensores de tensão, corrente e temperatura foram adequados ao projeto de forma que a conexão com a plataforma Arduino se estabelecesse. A programação do Arduino e do módulo Wi-Fi MCU8266-12E foi desenvolvida utilizando a IDE do Arduino e selecionando as bibliotecas que atendessem os propósitos previstos neste projeto. A conectividade através do protocolo NB-IoT, não foi possível por falta de suporte da operadora que foi contactada.

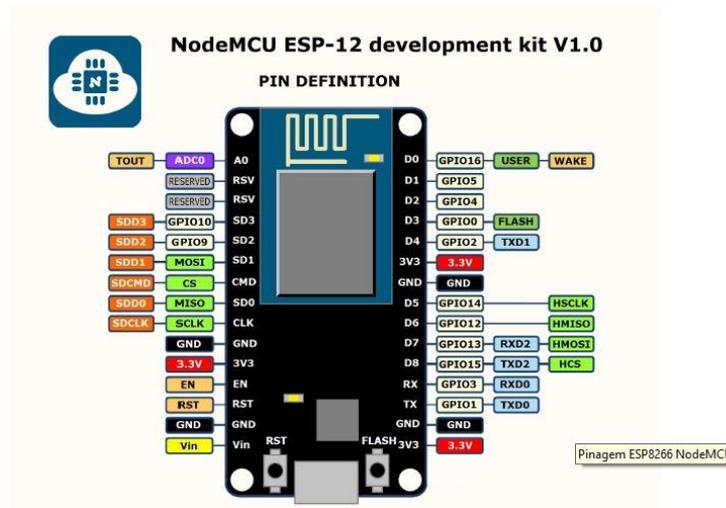
Com este trabalho concluiu-se que de fato é possível uso de novas tecnologias de transporte de informações no monitoramento de transformadores de potência em campo. Com o uso da tecnologia Wi-Fi obteve-se um resultado satisfatório na obtenção de dados em campo para monitoramento de transformadores de potência.

6. Referências

- [1] Site da ONS, Operadora Nacional do Sistema Elétrico, acessado em outubro de 2020: <http://www.ons.org.br/paginas/energia-agora/carga-e-geracao>
- [2] Umans, Stephen D. Máquinas Elétricas – 7º Edição 2014.
- [3] Magrani, Eduardo. A Internet das Coisas – FGV 2018.
- [4] Lora Long Range, Site: https://www.gta.ufrj.br/ensino/eel878/redes1-2019-1/vf/lora/outras_tecno.html, acessado em outubro de 2020.
- [5] CERP IoT, 2009. Site: <http://www.internet-of-thingsresearch.eu/pdf>, acessado em outubro de 2020.
- [6] Leite, J.R. Emiliano; Martins, Paulo S; Ursini, Edson. <http://lcv.fee.unicamp.br/images/BTSym-17/Paper/76926.pdf>, acessado em outubro de 2020.
- [7] Costa, David P.; Silva, Tamara talita; Silva, Vinicius Monteiro. Site: http://univertecefetmg.blogspot.com/2013/11/toshiba-transformadores-aspectos_19.html, acessado em outubro de 2020.
- [8] Site: <https://store.arduino.cc/usa/arduino-uno-rev3>, acessado em outubro de 2020.
- [9] Site: <https://www.filipeflop.com/blog/desenvolvendo-uma-biblioteca-arduino/> acessado em outubro de 2020.
- [10] Site: <https://github.com/openenergymonitor/EmonLib/>, acessado em outubro de 2020.

7. Anexos e Apêndices

Anexo 1: Diagrama com pinos de conexão do módulo Wi-Fi ESP 8266-12E



Anexo2: Biblioteca Energy Monitor emon1

- **Energy Monitor emon1:** Cria uma instância denominada de emon1 para medição das variáveis elétricas;
- **emon1.current (pinoSensor, CURRENT_CAL):** Passa para a função os parâmetros pino analógico do sensor de corrente (*pinoSensor*) e uma constante para calibração (*CURRENT_CAL*);
- **emon1.voltage (pinoSensorV, VOLTAGE_CAL, 1.7):** Passa para a função os parâmetros pino analógico do sensor de tensão (*pinoSensorV*) e uma constante para calibração (*VOLTAGE_CAL*);
- **emon1.calcVI (TEM_CICLOS, LIM_TEMP):** Função de cálculo das grandezas elétricas com a quantidade de ciclos determinados por *TEM_CICLOS* e com o tempo limite para fazer a medição determinado por *LIM_TEMP* em mS;
- **emon1.realPower:** Fornece a potência Ativa em W;
- **emon1.apparentPower:** Fornece a potência Aparente e VA;
- **emon1.powerFactor:** Fornece o Fator de Potência;
- **emon1.Vrms:** Fornece a tensão RMS em V;
- **emon1.Irms:** Fornece a corrente RMS em A;
- **emon1.serialprint():** Envia para a porta serial as variáveis elétricas.

Apêndice 3: Biblioteca LiquidCrystal

- **LiquidCrystal lcd (rs, en, d4, d5, d6, d7):** Cria uma variável do tipo LiquidCrystal denominada lcd . A exibição é controlada usando 4 linhas de dados (d4, d5, d6 e d8) e os pinos rs e en do display;
- **lcd.begin (col, lin):** Inicializa a interface para o display LCD e especifica as dimensões (coluna e linha) do LCD;
- **lcd.clear():** Apaga todas as informações no LCD;
- **lcd.setCursor(col, lin):** Posiciona o cursor na coluna *col* e linha *lin*;
- **lcd.print("text"):** Imprime o texto especificado na string text.

Anexo 4: Biblioteca Max6675.h

- **MAX6675 thermocouple(CLK, CS, DO):** Cria uma variável denominada thermocouple associada aos pinos de comunicação digital com o protocolo SPI (CLK, CS e DO) ligados aos respectivos pinos do CI;
- **thermocouple.readCelsius():** Fornece a temperatura medida em Graus Celsius.

Anexo 5: Biblioteca WiFi.h

- **WiFi.begin(ssid, password):** Inicializa a comunicação Wi-Fi na rede ssid com senha password especificada;

•**WiFi.status()**: Verifica o status da conexão.

Anexo 6: Biblioteca ESP8266WebServer.h

•**ESP8266WebServer server(80)**: Configura a porta padrão para o servidor da web.

•**server.on("/", handle_OnConnect)**: Servidor recebe uma solicitação HTTP - chama a função handle_OnConnect;

•**server.onNotFound(handle_NotFound)**: Servidor recebe uma solicitação HTTP não especificada - chama a função handle_NotFound;

•**server.begin()**: Inicializa o servidor;

•**server.handleClient()**: Chama o método handleClient();

•**server.send(200,"text/html",EnvioHTML(variáveis))**: Envia as informações usando o código 200, especifica o conteúdo como "text/html" e chama a função EnvioHTML.

Apêndice 1: Software de programação do Arduino UNO para simulação no ISIS PROTEUS.

```
#include "EmonLib.h" // Biblioteca: Tratamento das variáveis de tensão e correntes AC
#include <LiquidCrystal.h> //Biblioteca: Tratamento do Display LCD
#define MAX6675_CS 5
#define MAX6675_SO 6
#define MAX6675_SCK 4
#define CURRENT_CAL 15.1
//VALOR DE CALIBRAÇÃO DE CORRENTE (DEVE SER AJUSTADO EM //PARALELO COM UM //MULTÍMETRO
MEDINDO A CORRENTE DA CARGA)
const int pinoSensor = A0;
//PINO ANALÓGICO EM QUE O SENSOR DE CORRENTE ESTÁ CONECTADO
const int pinoSensorV = A2;
// PINO ANALÓGICO EM QUE O SENSOR DE TENSÃO ESTÁ CONECTADO
LiquidCrystal lcd(7, 8, 9, 10, 11, 12);
EnergyMonitor emon1;
//CRIA UMA INSTÂNCIA PARA MEDIÇÃO DAS VARIÁVEIS ELÉTRICAS DO //TRAFO
void setup(){
Serial.begin(9600); //INICIALIZA A SERIAL PARA O MONITOR SERIAL
emon1.current(pinoSensor, CURRENT_CAL);
//PASSA PARA A FUNÇÃO OS PARÂMETROS (PINO ANALÓGICO / VALOR DE //CALIBRAÇÃO)
emon1.voltage(pinoSensorV, 44.5, 1.7);
//PASSA PARA A FUNÇÃO OS PARÂMETROS (PINO ANALÓGICO / VALOR DE //CALIBRAÇÃO/ZERO CROSSING)
lcd.begin(16, 2);
delay(2000);}
void loop(){
emon1.calcVI(17, 300);
//FUNÇÃO DE CÁLCULO (17 SEMICICLOS / TEMPO LIMITE PARA FAZER A //MEDIÇÃO)
emon1.serialprint(); // Manda para a serial as variáveis elétricas
float P = emon1.realPower; //Potência Ativa
float S = emon1.apparentPower; //Potência Aparente
```

```

float FP= emon1.powerFactor; //Fator de potência
float V = emon1.Vrms; //Tensão
float I = emon1.Irms; //Corrente
float T = readThermocouple(); //Temperatura
Serial.println(T);
//Visualização no LCD
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("P=");
lcd.setCursor(8, 0);
lcd.print("S=");
lcd.setCursor(0, 1);
lcd.print("V=");
lcd.setCursor(8, 1);
lcd.print("I=");
lcd.setCursor(2, 0);
lcd.print(P);
lcd.setCursor(10, 0);
lcd.print(S);
lcd.setCursor(2, 1);
lcd.print(V);
lcd.setCursor(10, 1);
lcd.print(I);
delay(1000);
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("TEMPERATURA");
lcd.setCursor(0, 1);
lcd.print(T);
delay(1000);
}
//Função de Leitura do sensor de temperatura MAX6675-Protocolo SPI
double readThermocouple() {
uint16_t v;
pinMode(MAX6675_CS, OUTPUT);
pinMode(MAX6675_SO, INPUT);
pinMode(MAX6675_SCK, OUTPUT);

digitalWrite(MAX6675_CS, LOW);
delay(1);
v = shiftIn(MAX6675_SO, MAX6675_SCK, MSBFIRST);
v <<= 8;

```

```

v |= shiftIn(MAX6675_SO, MAX6675_SCK, MSBFIRST);
digitalWrite(MAX6675_CS, HIGH);
if (v & 0x4)
{
// Bit 2 indica se o termopar está desconectado
return NAN;
}
// Os três bits inferiores (0,1,2) são bits de status descartados
v >>= 3;
// Os bits restantes são o número de contagens de 0,25 graus (C)
return v*0.25;
}

```

Apêndice2: Software de Programação do Arduino UNO para comunicação com o NODE MCU 8266-12F

```

#include "EmonLib.h" //INCLUSÃO DE BIBLIOTECA
#include <SPI.h>
#include "max6675.h"
#include <LiquidCrystal.h>
#include <SoftwareSerial.h>

#define CURRENT_CAL 15.1 //VALOR DE CALIBRAÇÃO DE CORRENTE (DEVE SER AJUSTADO EM //PARALELO
COM UM MULTÍMETRO MEDINDO A CORRENTE DA CARGA)

const int pinoSensor = A0; //PINO ANALÓGICO EM QUE O SENSOR DE CORRENTE ESTÁ CONECTADO
const int pinoSensorV = A2; // PINO ANALÓGICO EM QUE O SENSOR DE TENSÃO ESTÁ CONECTADO
int thermoDO = 6; // SO = Digital 6
int thermoCS = 5; // CD = Digital 5
int thermoCLK = 4; // SCK = Digital 4
float envia[6];
int i;
MAX6675 thermocouple(thermoCLK, thermoCS, thermoDO);
SoftwareSerial EspSerial(2,3); // RX, TX Para comunicação com o NODE MCU ESP 8266-12E
LiquidCrystal lcd(7, 8, 9, 10, 11, 12);
EnergyMonitor emon1; //CRIA UMA INSTÂNCIA PARA MEDIÇÃO DAS VARIÁVEIS ELÉTRICAS DO TRAFIO
void setup(){
Serial.begin(9600); //INICIALIZA A SERIAL PARA O MONITOR SERIAL
EspSerial.begin(115200); // INICIALIZA A SERIAL DE COMUNICAÇÃO COM NODE MCU
emon1.current(pinoSensor, CURRENT_CAL); //PASSA PARA A FUNÇÃO OS PARÂMETROS (PINO //ANALÓGIO /
VALOR DE CALIBRAÇÃO)
emon1.voltage(pinoSensorV, 44.5, 1.7); //PASSA PARA A FUNÇÃO OS PARÂMETROS (PINO //ANALÓGIO / VALOR DE
CALIBRAÇÃO)
lcd.begin(16, 2);
delay(2000);
}
void loop(){

```

```

emon1.calcVI(100, 2000); //FUNÇÃO DE CÁLCULO (100 SEMICICLOS / TEMPO LIMITE PARA FAZER A //MEDIÇÃO)
float P = emon1.realPower;    //Potência Ativa
float S = emon1.apparentPower; //Potência Aparente
float FP= emon1.powerFactor; //Fator de potência
float V = emon1.Vrms;        //Tensão
float I = emon1.Irms;        //Corrente
float T = thermocouple.readCelsius(); //Temperatura
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("P=");
lcd.setCursor(8, 0);
lcd.print("S=");
lcd.setCursor(0, 1);
lcd.print("V=");
lcd.setCursor(8, 1);
lcd.print("I=");
lcd.setCursor(2, 0);
lcd.print(P);
lcd.setCursor(10, 0);
lcd.print(S);
lcd.setCursor(2, 1);
lcd.print(V);
lcd.setCursor(10, 1);
lcd.print(I);
delay(1000);
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("TEMPERATURA");
lcd.setCursor(0, 1);
lcd.print(T);
envia[0] = V;
envia[1] = I;
envia[2] = P;
envia[3] = S;
envia[4] = FP;
envia[5] = T;
delay(1000);
for (i = 0; i < 6; i++) {
EspSerial.println(envia[i]);
}
}

```

Apêndice 3: Software de Programação para o NODE MCU 8266-12E

```
#include<ESP8266WiFi.h> //Biblioteca para funcionamento do WiFi do ESP
#include<ESP8266WebServer.h> //Biblioteca para o ESP funcionar como servidor
#include "SoftwareSerial.h"
#define D7 13
#define D8 15
const char* ssid = "Alencar Oi 2.4G"; // Rede WiFi
const char* password = "Carloshenrique"; //Senha da Rede WiFi
ESP8266WebServer server(80); //server na porta 80
SoftwareSerial EspSerial(D7, D8); // RX = D7, TX = D8
float Tensao =0; //variável para armazenar a temperatura
float Corrente =0; //Variável para armazenar a umidade
float PotenciaAtiva =0; //variável para armazenar a temperatura
float PotenciaAparente =0; //Variável para armazenar a umidade
float FatorPotencia =0;
float Temperatura =0;
int recebe;
void setup() {
EspSerial.begin(115200); //inicia EspSerial
pinMode(D7,INPUT); //D7 é RX, receiver, definido como entrada
pinMode(D8,OUTPUT); //D8 é TX, transmitter, definido como saída
Serial.begin(9600); //Inicializa a comunicação serial
delay(50); // Intervalo para aguardar a estabilização do sistema
Serial.println("Conectando a Rede: "); //Imprime na serial a mensagem
Serial.println(ssid); //Imprime na serial o nome da Rede Wi-Fi
WiFi.begin(ssid, password); //Inicialização da comunicação Wi-Fi
//Verificação da conexão
while (WiFi.status() != WL_CONNECTED) { // Aguardando status da conexão
delay(1000);
Serial.print("."); //Imprime pontos
}
Serial.println("");
Serial.println("WiFi Conectado");
Serial.print("IP: ");
Serial.println(WiFi.localIP()); //Função para exibir o IP da ESP
server.on("/", handle_OnConnect); //Recebe uma solicitação HTTP - chama a função handle_OnConnect
server.onNotFound(handle_NotFound); //Servidor recebe uma solicitação HTTP não especificada - chama a //função
handle_NotFound
server.begin(); //Inicializa o servidor
Serial.println("Servidor HTTP inicializado");
}
void loop() {
```

```

server.handleClient();//Chama o método handleClient()
}
void handle_OnConnect() {
delay(500);
if(EspSerial.available()) {
Tensao = EspSerial.parseFloat();
Corrente = EspSerial.parseFloat();
PotenciaAtiva = EspSerial.parseFloat();
PotenciaAparente = EspSerial.parseFloat();
FatorPotencia = EspSerial.parseFloat();
Temperatura = EspSerial.parseFloat();
}
server.send(200, "text/html", EnvioHTML(Tensao, Corrente, PotenciaAtiva, PotenciaAparente, FatorPotencia, Temperatura));
//Envia as informações usando o código 200, especifica o conteúdo como "text/html" e chama a função EnvioHTML
}
void handle_NotFound() { //Função para lidar com o erro 404
server.send(404, "text/plain", "Não encontrado"); //Envia o código 404, especifica o conteúdo como //"text/pain" e envia a
mensagem "Não encontrado"
}
String EnvioHTML(float Tensaostat, float Correntestat, float PotenciaAtivastat, float PotenciaAparentestat, float
FatorPotenciastat, float Temperaturastat) { //Exibindo a página da //web em HTML
String ptr = "<!DOCTYPE html> <html>\n"; //Indica o envio do código HTML
ptr += "<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1.0, user-scalable=no\">\n"; //Torna a
página da Web responsiva em qualquer navegador Web
ptr += "<meta http-equiv='refresh' content='2'>"; //Atualizar browser a cada 2 segundos
ptr+="<linkhref=\"https://fonts.googleapis.com/css?family=Open+Sans:300,400,600\"rel=\"stylesheet\">\n";
ptr += "<title>Monitor de Variaveis do Transformador</title>\n"; //Define o título da página
//Configurações de fonte do título e do corpo do texto da página web
ptr += "<style>html { font-family: 'Open Sans', sans-serif; display: block; margin: 0px auto; text-align: center;color:
#000000;}\n";
ptr += "body{margin-top: 50px;}\n";
ptr += "h1 {margin: 50px auto 30px;}\n";
ptr += "h2 {margin: 40px auto 20px;}\n";
ptr += "p {font-size: 24px;color: #000000;margin-bottom: 10px;}\n";
ptr += "</style>\n";
ptr += "</head>\n";
ptr += "<body>\n";
ptr += "<div id=\"webpage\">\n";
ptr += "<h1>Monitor de Variaveis do Transformador</h1>\n";
ptr += "<h2>NODEMCU ESP8266 Web Server</h2>\n";
//Exibe as informações das variáveis do trafo na página web
ptr += "<p><b>Tensao: <b>";
ptr += (float)Tensaostat;
ptr += " Volts</p>";

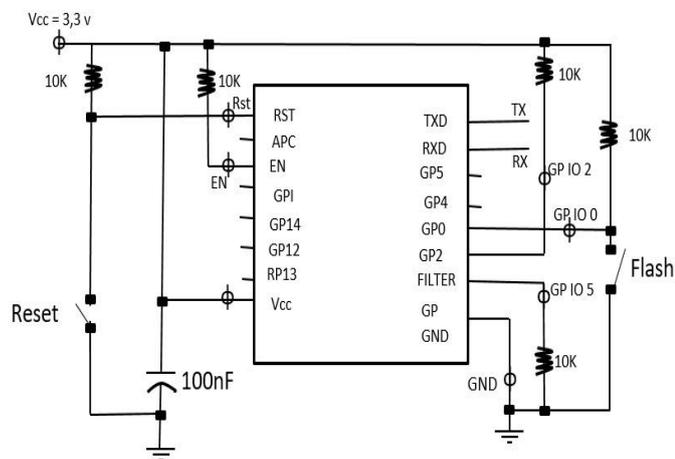
```

```

ptr += "<p><b>Corrente: </b>";
ptr += (float)Correntestat;
ptr += " A</p>";
ptr += "<p><b>Potencia Ativa: </b>";
ptr += (float)PotenciaAtivastat;
ptr += " W</p>";
ptr += "<p><b>Potencia Aparente: </b>";
ptr += (float)PotenciaAparentestat;
ptr += " VA</p>";
ptr += "<p><b>Fator de Potencia: </b>";
ptr += (float)FatorPotenciastat;
ptr += "</p>";
ptr += "<p><b>Temperatura: </b>";
ptr += (float)Temperaturastat;
ptr += " oC</p>";
ptr += "</div>\n";
ptr += "</body>\n";
ptr += "</html>\n";
return ptr;
}

```

Apêndice 4: Diagrama de conexão do Módulo MCU ESP 8266-12E



OBS: No [arduino](#) fechas (RST-GND)

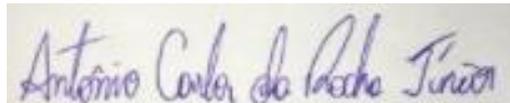
RESOLUÇÃO n°038/2020 – CEPE

ANEXO 3

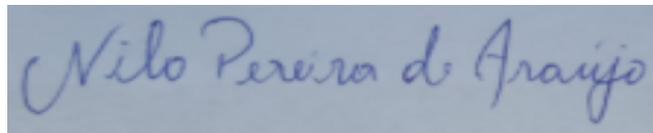
Termo de autorização de publicação de produção acadêmica

Os estudantes Antônio Carlos da Rocha Júnior, do Curso de Engenharia Elétrica matrícula 2011.2.0038.0011-2, telefone: (62) 985329191 e-mail eng-antonio@outlook.com, e Nilo Pereira de Araujo do Curso de Engenharia Elétrica matrícula 2014.2.0038.0171-4, telefone: (62) 981211113 e-mail niloempreender@gmail.com, nas qualidades de titulares dos direitos autorais, em consonância com a Lei n° 9.610/98 (Lei dos Direitos do autor), autoriza a Pontifícia Universidade Católica de Goiás (PUC Goiás) a disponibilizar o Trabalho de Conclusão de Curso intitulado PROTÓTIPO PARA MONITORAMENTO DOS PARÂMETROS DE TRANSFORMADORES POR INTERNET DAS COISAS - IoT, gratuitamente, sem ressarcimento dos direitos autorais, por 5 (cinco) anos, conforme permissões do documento, em meio eletrônico, na rede mundial de computadores, no formato especificado (Texto (PDF); Imagem (GIF ou JPEG); Som (WAVE, MPEG, AIFF, SND); Vídeo (MPEG, MWV, AVI, QT); outros, específicos da área; para fins de leitura e/ou impressão pela internet, a título de divulgação da produção científica gerada nos cursos de graduação da PUC Goiás.

Goiânia, 11 de dezembro de 2020.



Antônio Carlos da Rocha Júnior



Nilo Pereira de Araujo



Professor-orientador: Carlos Alberto Vasconcelos Bezerra