UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO, DE CIÊNCIAS EXATAS E EDUCAÇÃO
DEPARTAMENTO DE ENG. DE CONTROLE, AUTOMAÇÃO E COMPUTAÇÃO
CURSO DE GRADUAÇÃO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Igor Becker Boos

Desenvolvimento de um supervisório para testes e monitoramento de células individuais de um inversor de média tensão

Blumenau
2023

Igor Becker Boos

Desenvolvimento de um supervisório para testes e monitoramento de células individuais de um inversor de média tensão

Trabalho de Conclusão de Curso submetido ao curso de Graduação em Engenharia de Controle e Automação do Centro Tecnológico, de Ciências Exatas e Educação da Universidade Federal de Santa Catarina como requisito parcial para a obtenção do título de Engenheiro de Controle e Automação.

Orientador(a): Prof.(a) Dr.(a) Janaina Gonçalves Guimarães

Blumenau

2023

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Boos, Igor Becker
Desenvolvimento de um supervisório para testes e monitoramento de células individuais de um inversor de média tensão / Igor Becker Boos ; orientador, Janaina Gonçalves Guimarães, 2023.
47 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Campus Blumenau, Graduação em Engenharia de Controle e Automação, Blumenau, 2023.

Inclui referências.

1. Engenharia de Controle e Automação. 2. Monitoramento. 3. inversor multinível de média tensão. 4. células do inversor. I. Guimarães, Janaina Gonçalves . II. Universidade Federal de Santa Catarina. Graduação em Engenharia de Controle e Automação. III. Título.

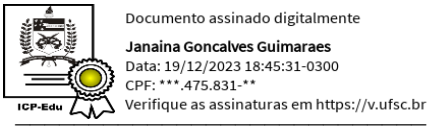
Igor Becker Boos

Desenvolvimento de um supervisório para testes e monitoramento de células individuais de um inversor de média tensão

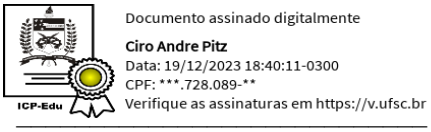
Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de "Engenheiro de Controle e Automação" e aprovado em sua forma final pelo Curso de Graduação em Engenharia de Controle e Automação

Blumenau, 12 de Dezembro de 2023

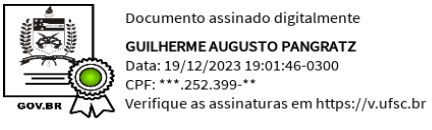
Banca examinadora:



Prof. Janaina Gonçalves Guimarães, Dr.(a)
Universidade Federal de Santa Catarina



Prof. Ciro André Pitz, Dr.
Universidade Federal de Santa Catarina



Eng. Guilherme Augusto Pangratz
WEG

AGRADECIMENTOS

Agradeço aos meus pais Márcio Valmor Boos e Isabel de Fátima Becker Boos, por terem me proporcionado o apoio necessário para prosseguir com meus estudos ao longo desses anos, e por investirem no meu aprendizado, uma oportunidade que eles mesmos não tiveram.

Agradeço à minha namorada Maria Gabriela Prada de Souza, por todo apoio durante esses anos na minha jornada acadêmica, sempre me incentivando a nunca desistir.

Agradeço aos meus professores, em especial a minha orientadora Janaina Gonçalves Guimarães, que graças à sua disponibilidade e orientação foi possível concretizar este trabalho de conclusão de curso

Agradeço à empresa WEG, por me dar a oportunidade de estagiar e me desenvolver profissionalmente.

Agradeço aos meus amigos do setor MVW que me auxiliaram na realização deste trabalho.

RESUMO

Durante o processo de produção de um determinado inversor de média tensão, são feitos diversos testes para garantir a conformidade do produto com os padrões previamente estipulados. Um desses testes refere-se aos parâmetros manipulados pelo cartão de controle de cada célula do inversor. Tendo em vista a necessidade de aperfeiçoar estes testes e monitorar individualmente cada célula, o presente trabalho teve como objetivo criar um supervisório para a visualização dos parâmetros que envolvem a célula testada e armazenamento destes dados em um arquivo CSV. Para isso, criou-se um protótipo para simular fisicamente a célula do inversor em funcionamento e programou-se a interface de comunicação entre a célula e o supervisório. Para esta comunicação ocorrer, transmitiu-se os dados do cartão de controle para o conversor de fibra óptica via USB e, em seguida, utilizou-se a comunicação serial (por meio do protocolo UART) para transferência desses dados para o computador. Para detecção dos erros, os dados foram tratados utilizando o algoritmo CRC-8. Com o supervisório implementado, a expectativa é que haja uma melhoria na realização dos testes, oferecendo uma ferramenta para o monitoramento das células e contribuindo para o progresso tecnológico que envolve este inversor.

Palavras-chave: Monitoramento; inversor multinível de média tensão; células do inversor.

ABSTRACT

During the production process of a specific medium-voltage inverter, various tests are conducted to ensure product compliance with predetermined standards. One of these tests focuses on the parameters controlled by the control card of each inverter cell. Recognizing the need to enhance these tests and individually monitor each cell, this study aimed to create a supervisory system for visualizing the parameters associated with the tested cell and storing this data in a CSV file. To achieve this, a prototype was developed to physically simulate the functioning of the inverter cell, and a communication interface between the cell and the supervisory system was programmed. For this communication to take place, data from the control card was transmitted to the optical fiber converter via USB, followed by the use of serial communication (through the UART protocol) to transfer this data to the computer. Error detection was implemented using the CRC-8 algorithm. With the implemented supervisory system, the expectation is a significant impact on test execution, providing an excellent tool for cell monitoring and contributing to technological progress in this inverter.

Keywords: Monitoring; medium-voltage multilevel inverter; inverter cells.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ADC - Analog to digital converter

CCE - Cartão de controle engenheirado

CHB - Cascaded H-bridge

CRC8 - Cyclic Redundancy Check

CSV - Comma separated values

DC - Direct current

DIP - Dual in line package

IGBT - Insulated gate bipolar transistor

IHM - Interface homem máquina

MCC1 - Módulo de cartão de controle

PIC2 - Power Interface Card 2

PWM - Pulse width modulation

PS24 - Power supply 24V

UART - Universal asynchronous receiver/transmitter

USB - Universal serial bus

SUMÁRIO

1 INTRODUÇÃO

O MVW3000 (Figura 1), produzido pela empresa catarinense WEG, é um inversor ponte completa em cascata ou *Cascaded H-bridge inverter* (CHB). Este equipamento é utilizado para o acionamento de motores de média tensão, sobretudo em processos industriais que necessitam de variação e controle de velocidade, como: bombas, compressores, ventiladores, correias transportadoras e moinhos. As formas de onda de saída quase senoidais produzidas permitem o emprego deste inversor em motores de indução, sem a necessidade de isolamento especial.

Figura 1 - MVW3000

Fonte: WEG (2023)

Durante o processo de produção do inversor de média tensão MVW3000, diversos testes são feitos a fim de garantir a conformidade com padrões previamente estipulados. Um desses testes refere-se aos parâmetros manipulados pelo cartão de controle de cada célula do inversor. Atualmente os testes das células são feitos em tempo real, analisando os dados com o auxílio de instrumentos. O problema é que utilizar-se da instrumentação não é prático. Além disso, existe a possibilidade de acontecer, durante os testes, algum tipo de erro ou dano que corrompa o histórico de dados do inversor, impossibilitando o mesmo de realizar uma análise mais apurada da provável causa do problema.

Sendo assim, este trabalho visa criar um supervisório para a visualização e armazenamento de parâmetros nos testes de uma célula individual do inversor de frequência de média tensão MVW3000. Com isso, será possível obter melhorias no processo de testes, auxiliando na visualização de parâmetros específicos que envolvem esta única célula e no

armazenamento dos dados em um arquivo CSV (*Comma Separated Values* ou, em português, Valores Separados por Vírgula) por meio de um computador externo.

Para criar o supervisório, foi necessário confeccionar um protótipo para simular fisicamente uma das células do inversor. Em seguida, programou-se a comunicação entre a célula e o supervisório. Para isso, transmitiu-se os dados do módulo de cartão de controle (MCC1) para o conversor de fibra óptica via USB e, em seguida, utilizou-se o protocolo UART para transferência desses parâmetros para o computador. Para detecção dos erros, o tratamento dos dados foi feito por meio do algoritmo CRC-8.

1.1 OBJETIVO GERAL

Criar um supervisório para a visualização e armazenamento de parâmetros nos testes das células de um inversor de frequência de média tensão.

1.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos deste trabalho são listados a seguir:

- confeccionar um protótipo para simular fisicamente uma célula do inversor em funcionamento;
- analisar estados e variáveis do inversor e das células;
- programar a interface de comunicação entre célula e supervisório;
- criar tela de visualização dos parâmetros em tempo real;
- criar tela de visualização dos parâmetros em interface gráfica;
- armazenar parâmetros da comunicação em arquivo CSV;
- visualizar arquivos CSV gerados anteriormente diretamente no supervisório.

2 FUNDAMENTAÇÃO TEÓRICA

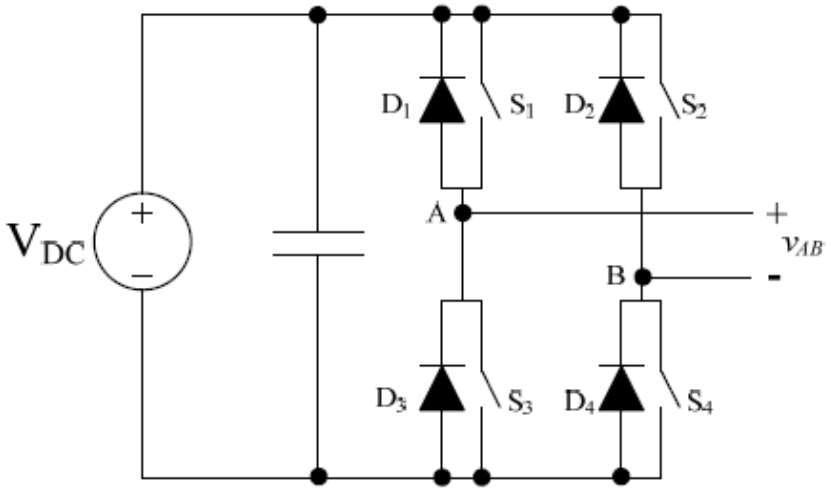
2.1 INVERSORES MULTINÍVEIS

Os conversores multiníveis possuem vantagens como a “qualidade na alimentação, excelente distribuição de tensão nos interruptores, boa compatibilidade eletromagnética, baixa perda de comutação e capacidade de trabalhar em altas tensões e potências” (WALTRICH, 2009, p. 2). O conversor multinível empregado neste trabalho é o inversor de tensão ponte completa em cascata, que pode ser abreviado como CHB, devido ao termo em inglês *Cascaded H-bridge inverter* (WALTRICH, 2009). Este tipo de inversor será melhor detalhado no tópico a seguir.

2.1.1 Inversor de Tensão Ponte Completa em Cascata (CHB)

O inversor multinível CHB é frequentemente utilizado para acionamentos e aplicações em média tensão (MT) de alta potência. Esta topologia é formada por múltiplas células H, que costumam ser conectadas em cascata no lado alternado, permitindo operações em média tensão com baixa distorção harmônica (WU, 2006). A Figura 2 apresenta uma célula H de um inversor CHB.

Figura 2 - Célula H de um inversor CHB



Fonte: WALTRICH (2009)

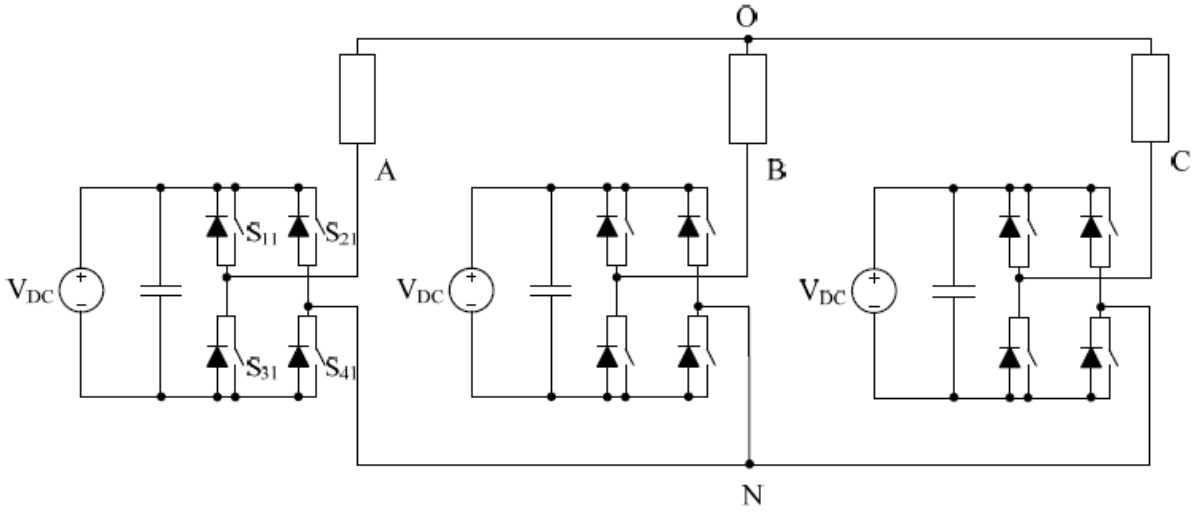
Cada célula H necessita de uma fonte isolada, o que é uma das desvantagens deste método. “As fontes contínuas normalmente são obtidas a partir de retificadores multipulsos ou de conversores CC-CC isolados bidirecionais” (WALTRICH, 2009, p. 13). O inversor monofásico ponte H é formado por duas pernas do inversor com dois dispositivos IGBT (*Insulated-Gate Bipolar Transistor*, ou Transistor Bipolar de Porta Isolada) em cada perna. A tensão de barramento CC Vdc é fixa. Mas, a tensão de saída (Vab) pode ser ajustada utilizando modulação bipolar ou unipolar (WU, 2006).

Nos inversores CHB, a distorção harmônica diminui conforme o número de níveis aumenta (GAIKWAD E ARBUNE, 2016). O número de níveis por fase (n_F) dos inversores CHB podem ser encontrados pela equação a seguir:

$$n_F = 2H_n + 1 \quad (1)$$

onde H_n é o número de células por fase. A Figura 3 mostra um inversor de tensão CHB trifásico de 3 níveis. Neste tipo de inversor, o valor de n_F é sempre ímpar, logo há também inversores CHB com 5 níveis, 7 níveis, 9 níveis, entre outros.

Figura 3 - inversor CHB trifásico de 3 níveis



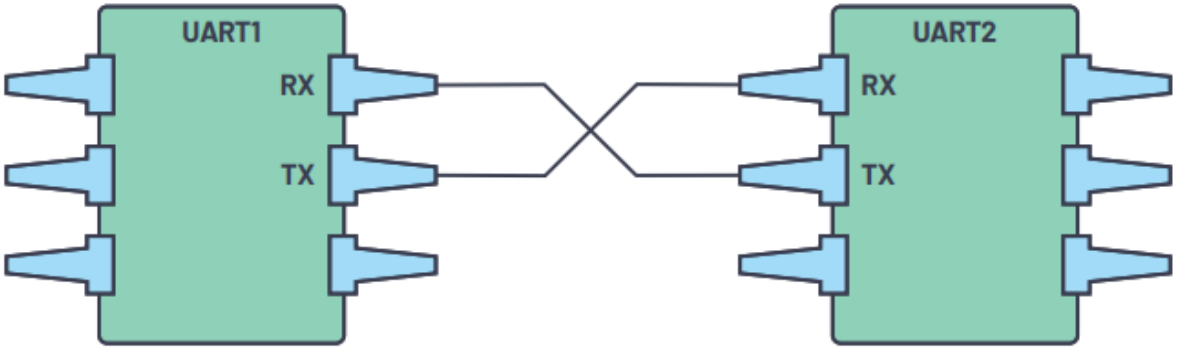
Fonte: WALTRICH (2009)

Os inversores de tensão ponte completa em cascata possuem como vantagens a estrutura modular, a baixa tensão de distorção harmônica na tensão de carga e a operação em alta tensão sem interruptores em série. Porém, necessitam de um grande número de fontes isoladas e elevada quantidade de componentes (WALTRICH, 2009).

2.2 PROTOCOLO UART

A sigla UART vem do inglês *universal asynchronous receiver-transmitter* e significa receptor-transmissor assíncrono universal. Trata-se de um protocolo de transmissão e recebimento de dados que utiliza comunicação serial, em que os dados são transmitidos bit a bit. Por se tratar de uma comunicação assíncrona com velocidade configurável, não há sinal de *clock* para sincronizar os bits do dispositivo transmissor para o receptor (PEÑA e LEGASPI, 2020). Na Figura 4, pode-se observar duas UARTs se comunicando entre si.

Figura 4 - Comunicação entre duas UARTs

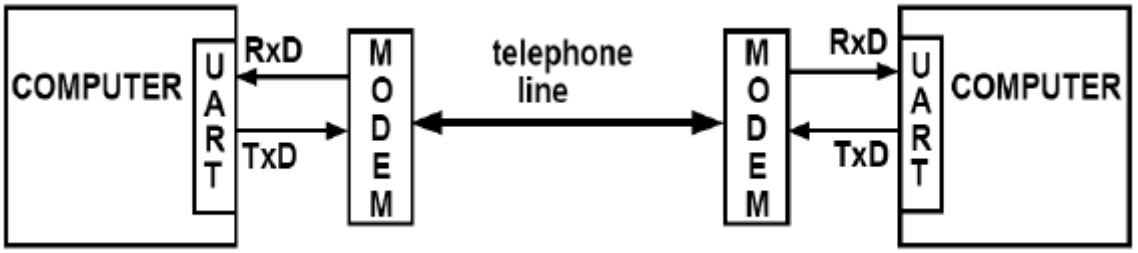


Fonte: PEÑA e LEGASPI (2020)

Segundo os autores supracitados, na comunicação entre duas UARTs, dois fios são utilizados para que o transmissor (TX) envie dados seriais para o receptor (RX). Este protocolo de comunicação entre dispositivos é comumente empregado em sistemas embarcados, microcontroladores e computadores. Um exemplo pode ser visto na Figura 5, no qual os dados são transmitidos de um local remoto por meio de uma linha telefônica conectada ao modem que, por sua vez, está trocando dados com uma porta serial. Vale

ressaltar que no presente trabalho, a transmissão ocorre por meio de cabos de fibra óptica, não necessitando do compartilhamento de GND entre dispositivos.

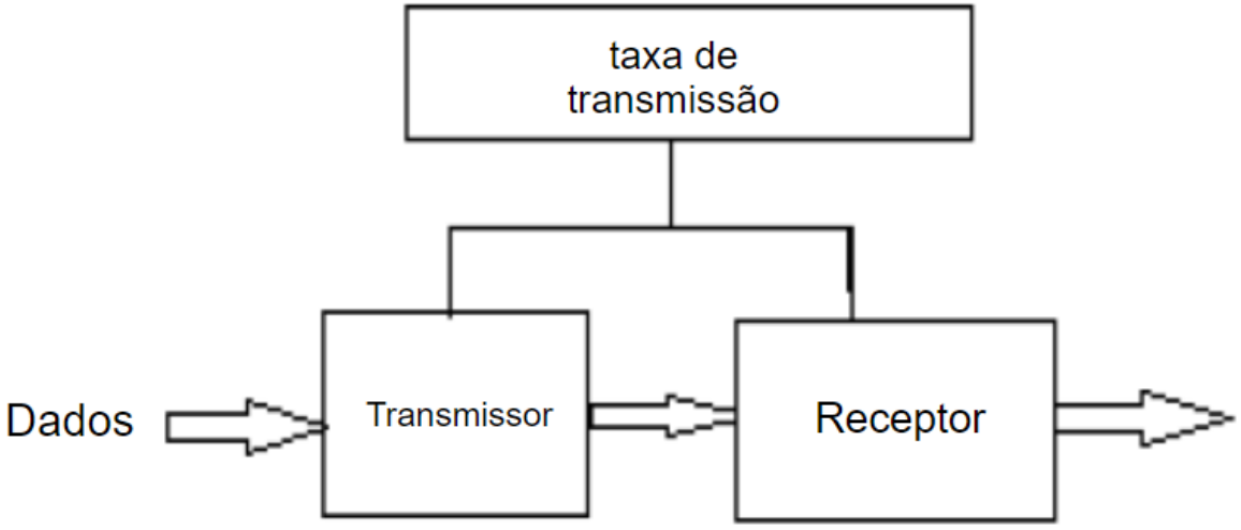
Figura 5 - Transmissão Serial de Dados



Fonte: NANDA e PATTNAIK (2016)

Conforme Nanda e Pattnaik (2016), há três componentes principais no diagrama de blocos UART: o controle do transmissor, o controle do receptor e o gerador de taxa de transmissão. Desta forma, a implementação do módulo UART consiste na realização destes três submódulos, conforme pode ser visto na Figura 6.

Figura 6 - Módulo UART



Fonte: adaptado de LADDHA e THAKARE (2013)

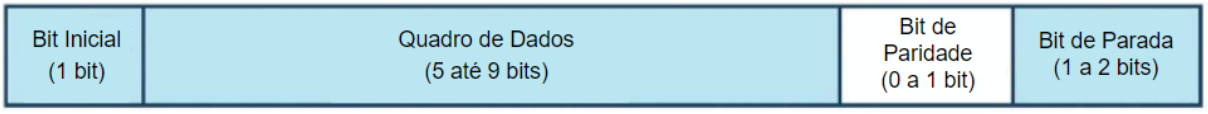
O módulo de transmissão converte os bytes em bits seriais de acordo com o formato básico do quadro e transmite esses bits através do pino de transmissão (TX). O módulo receptor recebe os sinais no pino de recepção (RX) e os converte em dados paralelos. Para controlar a transmissão e recepção dos dados, o gerador de taxa de transmissão produz um

sinal de *clock* local maior do que a taxa de transmissão (LADDHA e THAKARE, 2013). No caso da porta serial, a taxa de transmissão é o número máximo de bits por segundo a serem transferidos (PEÑA e LEGASPI, 2020).

Como os dispositivos de transmissão e recepção não são sincronizados por um sinal de relógio, "o transmissor gera um fluxo de bits baseado em seu sinal de *clock* enquanto o receptor usa seu sinal de *clock* interno" (PEÑA e LEGASPI, 2020, p. 2, tradução nossa).

A transmissão dos dados na interface UART se dá na forma de pacotes (Figura 7). Cada pacote consiste em: um bit inicial, um quadro de dados, um bit de paridade e bits de parada. Quando não está transmitindo dados, a linha de transmissão é mantida em estado lógico alto. Ao iniciar a transmissão, o UART transmissor faz a linha de transmissão ficar em estado lógico baixo durante um ciclo de *clock*. Dessa forma, o UART receptor começa a ler os bits no quadro de dados. De modo semelhante, para indicar o fim do pacote de dados, o UART transmissor puxa a linha de transmissão de um estado lógico baixo para um estado lógico alto por um a dois bits de duração (PEÑA e LEGASPI, 2020).

Figura 7 - Pacote UART



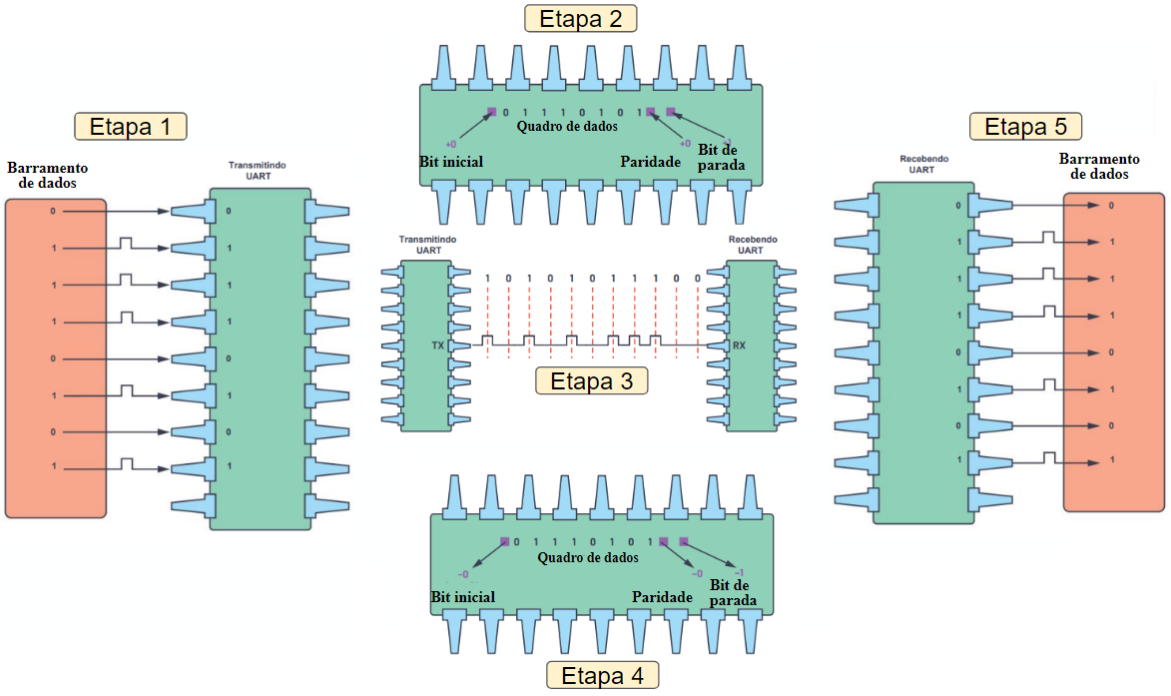
Fonte: adaptado de PEÑA e LEGASPI (2020)

Se nenhum bit de paridade for utilizado, o quadro de dados pode ter de cinco a nove bits. Porém, fatores como a radiação eletromagnética, taxas de transmissão incompatíveis ou transferências de dados de longa distância podem alterar os dados durante a transmissão. Deste modo, recomenda-se utilizar o bit de paridade para identificar se houve alguma modificação indevida, uma vez que ele "conta o número de bits com valor 1 e verifica se o total é um número par ou ímpar [...] Quando o bit de paridade corresponde aos dados, a UART sabe que a transmissão estava provavelmente livre de erros" (PEÑA e LEGASPI, 2020, p. 2, tradução nossa). Se um bit de paridade for utilizado, a quantidade máxima de bits no quadro de dados é oito.

A Figura 8 representa as etapas de transmissão UART descritas pelos autores Peña e Legaspi (2020). Como pode ser visto na imagem, o UART transmissor está conectado a um barramento de dados que envia para ele os dados de forma paralela (Etapa 1). Em seguida, o

bit de início, o bit de paridade e o(s) bit(s) de parada são acionados (Etapa 2). Somente então, o pacote de dados é transmitido serialmente, bit a bit, do UART transmissor para o UART receptor (Etapa 3). Logo após, o UART receptor exclui o bit inicial, o bit de paridade e o bit de parada do quadro de dados (Etapa 4) e converte os dados seriais em paralelo para devolvê-los ao barramento de dados na extremidade receptora (Etapa 5).

Figura 8 - Etapas de transmissão UART



Fonte: adaptado de PEÑA e LEGASPI (2020)

O custo e a simplicidade do protocolo UART faz com que este sistema seja bastante atraente e amplamente utilizado. Além disso, por usar menos fios, apresenta menor distorção do sinal do que a comunicação paralela, sendo ideal para transmissões de dados a longa distância (LADDHA e THAKARE, 2013). Neste sentido, conhecer como funciona a comunicação UART é útil para evitar erros na transmissão e aplicá-lo em projetos futuros.

2.3 VERIFICAÇÃO CÍCLICA DE REDUNDÂNCIA (CRC)

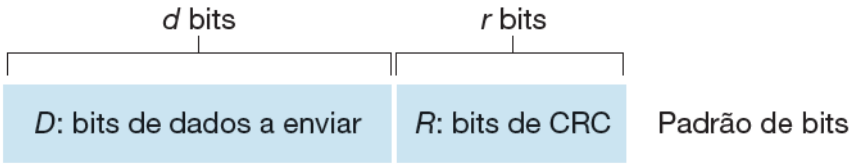
Tanto as interferências elétricas de fontes naturais (como os raios), quanto àquelas de fontes artificiais (como fontes chaveadas, motores, geradores, linhas de energia, etc) podem

ocasionar erros na transmissão de dados. Por isso, monitorar a comunicação e determinar quando ocorrem erros é fundamental. Para isso, algumas técnicas de detecção de erros são empregadas, como a Verificação Cíclica de Redundância (que pode ser abreviada como CRC, devido ao termo em inglês *Cyclic Redundancy Check*).

O termo "Redundância" consiste em duplicar cada unidade de dados com a finalidade de detectar erros. Contudo, um circuito de comunicação de dados pode ter apenas alguns metros de comprimento ou vários milhares de metros de comprimento. Isto é, apesar de eficaz, a "Redundância" pode se tornar cara, sobretudo em mensagens longas. Por isso, é muito mais eficiente e barato adicionar bits às unidades de dados com o propósito de verificar se há algum erro de transmissão. Este método, é chamado de "Verificação de Redundância" (TOMASI, 2013).

Conforme Kurose e Ross (2013), os códigos CRC consideram a cadeia de bits a ser enviada como um polinômio cujos coeficientes são os valores 0 e 1 na cadeia. Considerando que o emissor deseje enviar ao receptor a parcela de *d* bits de dados (D). Para isso, é preciso que o emissor e o receptor entrem em acordo sobre o padrão (*r* + 1) bits, conhecido como polinômio gerador (G), que também é chamado de polinômio divisor. Para determinada parcela de dados (D), o emissor escolherá *r* bits adicionais (R), que serão os dados redundantes utilizados, para anexá-los a D, conforme a Figura 9.

Figura 9 - Códigos de CRC



Fonte: KUROSE e ROSS (2013)

Dessa forma, conforme pode ser visto na imagem acima, o padrão resultante de bits é (*d* + *r*), que é interpretado como um número binário. O receptor divide este padrão resultante (*d* + *r*) por G. Se não existir erros introduzidos durante a transmissão, então o receptor deverá ser capaz de dividir o polinômio recebido por G exatamente, deixando um resto zero. Se o polinômio recebido não for mais divisível exatamente por G, isto é, se o receptor obter na divisão um resto diferente de zero, isto indicará que houve erro de transmissão (KUROSE e ROSS, 2013).

Cabe ressaltar que a aritmética utilizada nesta divisão não é a mesma utilizada com números inteiros, mas sim a aritmética polinomial módulo 2. Isto porque, como já dito, os coeficientes podem ser apenas 1 ou 0. Para evitar que um erro passe sem ser notado, é necessário selecionar o polinômio G de modo que seja muito improvável que a divisão seja exata para uma mensagem que possui erros (PETERSON e DAVIE, 2013).

Isto é, considere que o polinômio transmitido (d + r) seja P e há a introdução de erros com o acréscimo de outro polinômio E. Dessa forma, a mensagem transmitida com erros seria P + E. Sabe-se que P pode ser exatamente dividido por G. Se E também puder ser dividido por G, isto fará com que o erro passe sem ser notado. Por isso, é importante que a escolha de G se dê de forma que isso seja muito pouco provável de acontecer para os tipos de erros mais comuns (PETERSON e DAVIE, 2013). Seis versões de G são muito utilizadas e podem ser vistas no Quadro 1.

Quadro 1 - Polinômios de CRC Comuns

CRC	**Polinômio Gerador (G)**
CRC-8	$x^8 + x^2 + x^1 + 1$
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^1 + 1$
CRC-12	$x^{12} + x^{11} + x^3 + x^2 + x + 1$
CRC-16	$x^{16} + x^{15} + x^2 + 1$
CRC-CCITT	$x^{16} + x^{12} + x^5 + 1$
CRC-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

Fonte: adaptado de PETERSON e DAVIE (2013)

O presente trabalho utilizou o CRC-8 para detecção de erros. Cabe destacar que existe uma diferença entre a detecção e a correção de erros. Na detecção de erros somos obrigados a descartar a mensagem e transmitir uma nova cópia, ocupando largura de banda e aumentando a latência. Contudo, apesar da correção parecer mais atrativa, nem sempre é a melhor opção. Isto porque a correção exige um maior número de bits redundantes para enviar um código corretor de erros do que um código que apenas detecta erros. Isto é, enquanto o código detector de erros exige que mais bits sejam enviados quando ocorrem erros, o código corretor de erros exige isso o tempo todo. Tendo isso em vista, códigos como os CRCs são opções interessantes que viabilizam o tratamento dos dados.

3 DESENVOLVIMENTO

3.1 COMPONENTES DO PROTÓTIPO DA CÉLULA DO INVERSOR

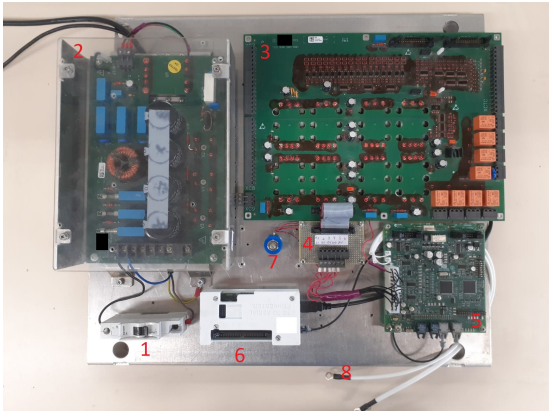
Para o desenvolvimento do protótipo, utilizou-se o MVW3000. Este inversor de frequência de média tensão possui 36 células, cada célula contém um Módulo de Cartão de Controle, denominado MCC1. O MCC1 é responsável pela comunicação dos sensores e controle dos atuadores, onde o mesmo transmite os dados relativos ao controle para o Cartão de Controle Engenheirado (CCE). Este último é a unidade de controle geral do inversor.

O protótipo, que foi desenvolvido no decorrer deste trabalho de conclusão de curso, visa simular o MCC1 em funcionamento, para que seja possível criar e validar o supervisório, sem que haja necessidade de ligar o inversor. Dada a demanda de aperfeiçoar o processo dos testes que envolvem o MVW3000, o supervisório produzido tem o intuito de transmitir e armazenar os dados gerados por uma das células do inversor para o computador. Esta nova interface possibilita verificar mais rapidamente os dados da célula do inversor na área de testes.

Com isso em vista, neste tópico será apresentado separadamente cada componente referente ao protótipo, mostrando os esquemas relevantes, justificativas das escolhas das peças e como se deu o processo de confecção do protótipo.

Conforme pode ser visto na Figura 10, os componentes principais do protótipo estão dispostos em uma placa de alumínio para dissipação do calor que é gerado pela Placa de Interface de Energia ou, em inglês, *Power Interface Card* (PIC2).

Figura 10 - Protótipo final de simulação do MCC1



Fonte: o autor (2023)

Os números presentes na Figura 10 indicam os principais componentes do protótipo que são nomeados no Quadro 2**.**

Quadro 2 - Componentes do protótipo de simulação do MCC1

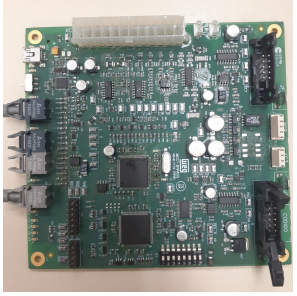
Legenda	**Componentes do Protótipo**
1	Disjuntor
2	PS24 (*Power Supply 24V*)
3	PIC2 (*Power Interface Card 2*)
4	Adaptador com cartão perfurado para alimentação PIC2 → MCC1
5	MCC1 (Módulo de Cartão de Controle)
6	Conversor de Fibra Óptica para USB
7	Potenciômetro
8	Sensor de Temperatura

Fonte: o autor (2023)

3.1.1 Módulo de Cartão de Controle (MCC1)

O MCC1 (Figura 11) é o cartão eletrônico de controle das células de potência do MVW3000. Este cartão recebe comandos e configurações do controle central, gera os sinais de *Pulse Width Modulation* (PWM) que são aplicados nas chaves de potência, realiza medições do estado do sistema e realimenta o controle central com as informações sobre a célula.

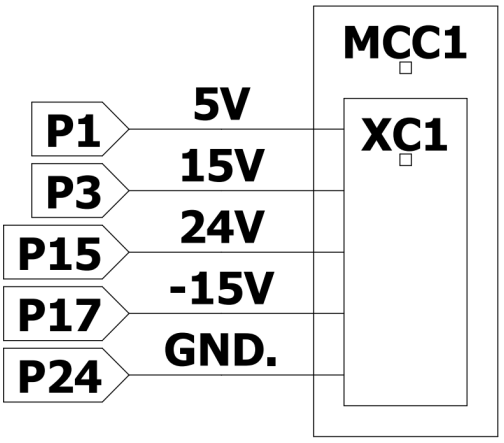
Figura 11 - Módulo de Cartão de Controle (MCC1)



Fonte: o autor (2023)

A alimentação do MCC1 se faz necessária, para isso o esquemático deste cartão de controle foi estudado com a intenção de localizar o local de alimentação. Conforme a Figura 12, a alimentação pode ser feita em vários terminais. Os terminais escolhidos para o desenvolvimento do protótipo foram 1, 3, 15, 17 e 24 da conexão XC1, com tensões variadas de 5V, 15V, 24V, -15V e GND, respectivamente.

Figura 12 - Esquemático de alimentação do MCC1



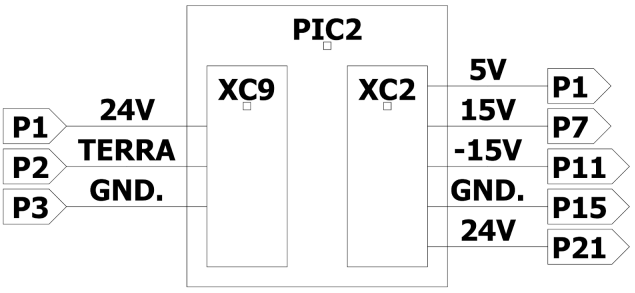
Fonte: o autor (2023)

Em virtude do esquemático revelar detalhes significativos do circuito interno da MCC1 e para garantir o direito à propriedade intelectual da WEG, o esquema apresentado na imagem foi simplificado contendo somente as entradas e saídas de alimentação.

3.1.2 *Power Interface Card* (PIC2)

Para a correta alimentação do MCC1, foi necessário encontrar alguma fonte geradora de tensões variadas. Devido a disponibilidade no local, a fonte de alimentação de tensões variadas isoladas utilizada foi a PIC2. Analisando o esquemático da PIC2 (Figura 13), foi encontrado um dos conectores possíveis onde todas as tensões necessárias para o MCC1 se fazem presente.

Figura 13 - Esquemático de entrada e saída da PIC2



Fonte: o autor (2023)

Os terminais 15 (GND), 1 (5V), 7 (15V), 11 (-15V), 21 (24V) da conexão XC2 foram utilizados para a alimentação necessária do MCC1.

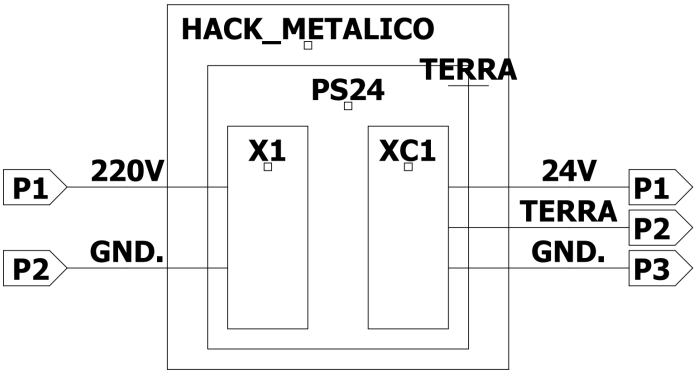
Devido o conector de saída da PIC2 ser incompatível com o conector de entrada da MCC1, uma placa de circuito perfurada foi inserida no protótipo, visando proporcionar ao mesmo uma alimentação de tensões variadas. Esta placa será melhor detalhada no subtópico 3.1.4.

A PIC2 necessita de uma alimentação de 24V, assim outra fonte de alimentação foi adicionada ao protótipo, conforme esquema apresentado na Figura 13. A alimentação foi feita pelos terminais 1 (24v), 2 (Terra) e 3 (GND) do conector XC9.

3.1.3 *Power Supply 24V* (PS24)

A PS24 é utilizada para a correta alimentação da PIC2. O esquemático dos terminais de entrada e saída da PS24 pode ser visto na Figura 14.

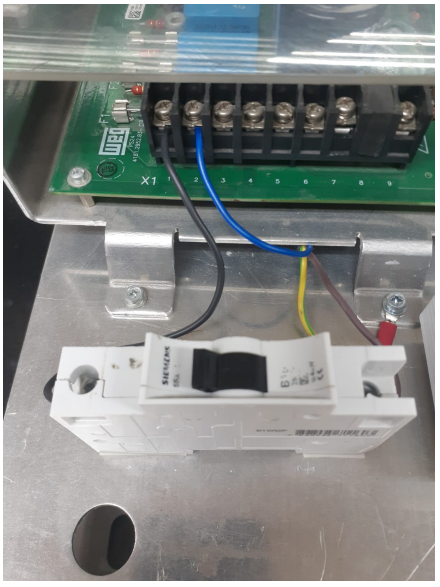
Figura 14 - Esquemático de entrada e saída de alimentação da PS24



Fonte: o autor (2023)

A PS24 é uma fonte DC/DC isolada, possui uma ponte retificadora de diodos com um filtro de capacitores ligados em paralelo. Para facilitar o acionamento da fonte, um disjuntor foi colocado junto à bancada (Figura 15).

Figura 15 - Alimentação da PS24 pelo disjuntor



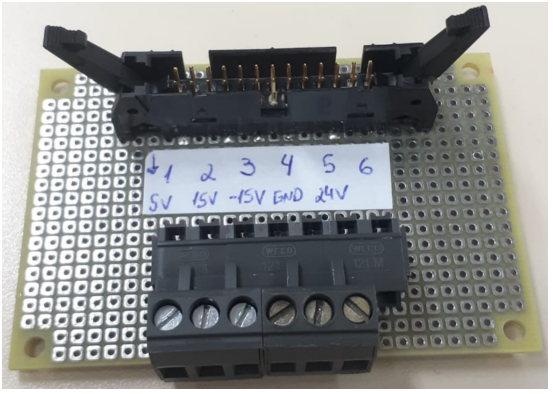
Fonte: o autor (2023)

A alimentação da PS24 se deu pela conexão X1 nos terminais 1 (220V) e 2 (GND). Os terminais de saída são 1 (24V), 2 (Terra) e 3 (GND). Além do aterramento de saída no pino 3, há também o aterramento que foi acoplado diretamente na carcaça metálica.

3.1.4 Conexão PIC2 - MCC1

Para a interface entre a PIC2 e o MCC1, foi confeccionado uma conexão entre as várias tensões, utilizando um cartão de circuito perfurado. A ligação foi pensada na versatilidade da utilização das tensões separadamente, tendo em vista tanto a utilização do protótipo quanto para atender outras demandas do setor de Desenvolvimento de Inversores em Média Tensão. Conforme a Figura 16, os terminais de saída 1 a 5 possuem as seguintes tensões, respectivamente, 5V, 15V, -15V, GND, 24V.

Figura 16 - Interface de conexão PIC2 - MCC1

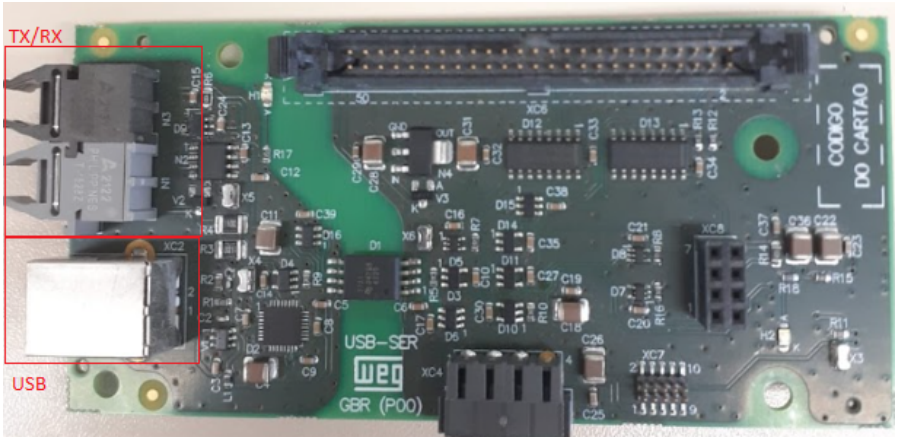


Fonte: o autor (2023)

3.1.5 Conversor de Fibra Óptica para USB

O MCC1 se comunica por meio de duas fibras ópticas TX e RX, respectivamente. Para a comunicação com o computador, um conversor de fibra óptica para porta serial é necessário. O conversor utilizado pertence à empresa WEG e pode ser conectado diretamente em um conector USB (Figura 17).

Figura 17 - Conversor de fibra óptica TX/RX para USB



Fonte: o autor (2023)

Como a comunicação é somente da transmissão do MCC1 para o conversor, é preciso apenas uma via de fibra óptica. Ou seja, somente o TX (transmissor) do MCC1 e o RX (receptor) do conversor são utilizados.

3.1.6 Acessórios

Para validação dos valores corretos transmitidos do sensor de temperatura para o MCC1, foi adicionado ao protótipo duas termorresistências do tipo PT100. O PT100 é um sensor de temperatura que pertence à classe dos sensores de resistência de platina. Ele possui alta precisão e estabilidade em uma ampla faixa de temperaturas, onde é realizada a conversão para temperatura correspondente.

Para os testes envolvendo os conversores analógicos digitais (ou ADCs, devido o termo em inglês *Analog to Digital Converter*), foi adicionado um potenciômetro. Dessa forma, é possível validar a correta amostragem dos dados no supervisório, que será abordado posteriormente. Ambos acessórios visam auxiliar nos testes envolvendo o supervisório final.

3.2 PROGRAMAÇÃO

Neste tópico será apresentado toda a implementação realizada em relação aos dois Softwares, o primeiro no MCC1 (Módulo de cartão de controle da célula) onde ajustes foram feitos em relação a comunicação e coleta de dados, e segundo o programa do supervisório que foi totalmente desenvolvido, onde os mesmos serão apresentados adiante.

3.2.1 Software do MCC1

A programação do MCC1 conta com vários arquivos nos quais a parte de controle de potência é implementada. Neste tópico, serão abordados os passos para que possa ser desenvolvido o tratamento e transmissão de dados para um dispositivo externo.

3.2.1.1 Dados gerenciados pelo MCC1

O MCC1 é o cartão de controle de uma das células do inversor, isto é, todas as variáveis relevantes referente à potência e atuação da célula são gerenciadas pelo mesmo. Para dar início à atualização de software no MCC1, foi feita uma análise de todo o código já

existente e suas funcionalidades com o intuito de encontrar como o mesmo poderia se comunicar, onde a comunicação via fibra óptica já estava implementada, necessitando apenas habilitar a mesma e realizar uma função que recebesse todos os bytes a serem transmitidos.

Para a comunicação da célula com o programa desenvolvido, foi necessário atualizar o software do MCC1 e enviar os dados relevantes via comunicação de fibra óptica, utilizando o protocolo de comunicação UART.

A linguagem de programação utilizada no MCC1 é a C. A comunicação se deu a partir de uma interrupção interna de tempo em que, a cada 100 milissegundos (ms), um vetor de bytes (*buffer*) é passado via fibra óptica para a placa de conversão de dados.

3.2.1.2 Vetor de comunicação

Neste tópico serão abordados todos os bytes que são enviados periodicamente e suas especificidades. A comunicação é feita via interrupção interna do MCC1 como já mencionado, todos os bytes são encapsulados em um único pacote que é transmitido via protocolo UART. A Tabela 1 apresenta todos os dados que são transmitidos.

Tabela 1 - Transmissão de dados

Byte	Descrição
0	Bits: 10101010
1	Bits: 00000010
2	ADC 0 Byte mais significativo
3	ADC 0 Byte menos significativo
4	ADC 1 Byte mais significativo
5	ADC 1 Byte menos significativo
6	ADC 2 Byte mais significativo
7	ADC 2 Byte menos significativo
8	ADC 3 Byte mais significativo
9	ADC 3 Byte menos significativo
10	ADC 4 Byte mais significativo
11	ADC 4 Byte menos significativo
12	ADC 5 Byte mais significativo
13	ADC 5 Byte menos significativo
14	ADC 6 Byte mais significativo
15	ADC 6 Byte menos significativo
16	ADC 7 Byte mais significativo
17	ADC 7 Byte menos significativo
18	ADC 8 Byte mais significativo
19	ADC 8 Byte menos significativo
20	ADC 9 Byte mais significativo
21	ADC 9 Byte menos significativo
22	ADC 10 Byte mais significativo
23	ADC 10 Byte menos significativo

Byte	Descrição
24	ADC 11 Byte mais significativo
25	ADC 11 Byte menos significativo
26	ADC 12 Byte mais significativo
27	ADC 12 Byte menos significativo
28	ADC 13 Byte mais significativo
29	ADC 13 Byte menos significativo
30	ADC 14 Byte mais significativo
31	ADC 14 Byte menos significativo
32	ADC 15 Byte mais significativo
33	ADC 15 Byte menos significativo
34	Byte da possição das Dips
35	Byte mais significativo vetor de falhas
36	Byte do vetor de falhas
37	Byte do vetor de falhas
38	Byte menos significativo vetor de falhas
39	Byte vetor de alarmes
40	Byte mais significativo temperatura da placa
41	Byte menos significativo temperatura da placa
42	Byte mais significativo temperatura do ambiente sensor 1
43	Byte menos significativo temperatura do ambiente sensor 1
44	Byte mais significativo temperatura do ambiente sensor 2
45	Byte menos significativo temperatura do ambiente sensor 2
46 AO 62	Reservado
63	CRC8

Fonte: o autor (2023)

Para a comunicação, um *buffer* de 64 Bytes foi analisando onde todos os valores são transmitidos. As observações de cada uma das variáveis estão descritas a seguir:

- *Bytes 0 e 1:* Estes dois bytes foram implementados pensando na sincronização da transmissão devido a mesma ser assíncrona. O receptor final faz a verificação se os dois primeiros bytes recebidos são respectivamente *header* 1 e 2, se sim o receptor recebe os outros 62 bytes finais. Isto será abordado mais detalhadamente no tópico do software do receptor.
- *Bytes 2 ao 33:* Bytes referente aos conversores analógicos digitais. A definição deles é de 14 bits, necessitando que o dado seja armazenado em dois bytes.
- *Byte 34:* O cartão MCC1 possui 1 *dip switch* de 8 vias, totalizando 1 Byte.
- *Bytes 35 ao 38:* são bits de *flags* agrupadas em 4 bytes. A Tabela 2 especifica cada bit do vetor de falhas.

Tabela 2 - Transmissão dos bits de falhas

Byte	bit	Descrição
35	0	Falha de Sobretensão do barramendo DC
	1	Falha de Subtensão do barramendo DC
	2	Falha de Sobretemperatura no IGBT
	3	Falha de Subtemperatura no IGBT
	4	Falha de Fonte de energia
	5	Falha de Sobretensão de entrada
	6	Falha de Subtensão de entrada
	7	Falha de Desbalanceamento do barramento
36	0	Falha de isolação
	1	Modo de processamento ativado
	2	MCC1 HW ID não configurado
	3	Overcurrent/Dessat on CC70x
	4	DTC neutral arm
	5	DTC feedback phase arm
	6	PWM feedback neutral arm
	7	PWM feedback phase arm
37	0	Dessat neutral arm
	1	Dessat phase arm
	2	Sync ok
	3	Bypass status
	4	Em falha
	5	Falha de fonte
	6	Timeout comunicação
	7	RX CRC Error
38	X	Reservado

Fonte: o autor (2023)

- *Byte 39:* bits de *Flags*. A Tabela 3 especifica cada bit do vetor de alarme.

Tabela 3 - Transmissão dos bits de alarme

Byte	bit	Descrição
39	0	Alarme de Sobretemperatura no IGBT
	1	Alarme de Sobretensão de entrada
	2	Alarme de Subtensão de entrada
	3	Reservado
	4	Reservado
	5	Reservado
	6	Reservado
	7	Reservado

Fonte: o autor (2023)

- *Bytes 40 e 41:* 2 Bytes que referenciam a temperatura da placa da MCC1.
- *Bytes 42 ao 45:* 2 Bytes que referenciam a temperatura dos sensores de temperatura.
- *Bytes 46 ao 62:* Reservado, possibilitando expansão de novos parâmetros caso necessário.
- *Byte 63:* Byte reservado para o valor calculado pelo CRC-8 do MCC1, onde do lado do receptor será feito também o cálculo do CRC-8 para comparação. Se o valor for

igual, a transmissão foi efetuada com sucesso (sem erros associados ao meio de transmissão). Se o valor divergir, o pacote de transmissão é descartado.

3.2.1.3 Observações Software MCC1

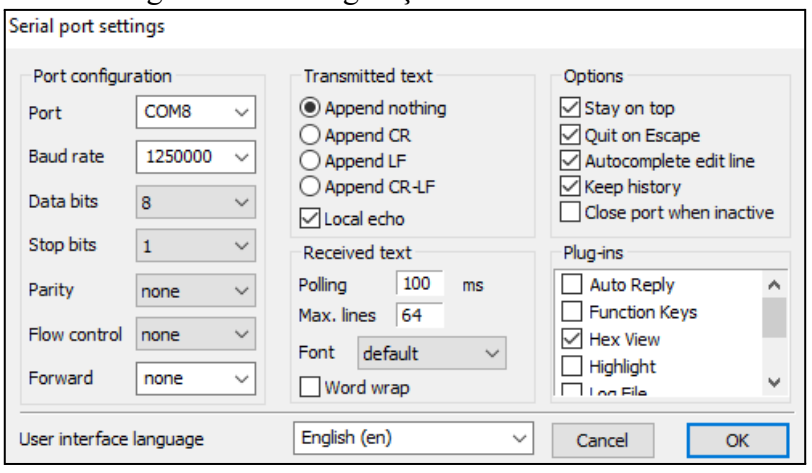
A transmissão do pacote foi inserida em uma interrupção por tempo de 100ms. A taxa de transmissão é de 640 bytes por segundo, levando em consideração que os bytes não utilizados são transmitidos de qualquer forma.

O supervisório final não possui necessidade de uma taxa de atualização maior que 10 vezes por segundo, tendo em vista que os testes não necessitam de uma grande precisão em relação ao tempo.

Para validação da comunicação do MCC1, o software Termite foi utilizado. O Termite é um terminal RS-232 que possui uma interface para visualização dos dados que estão sendo transferidos.

Para a visualização da transmissão, foi necessário configurar de acordo com os parâmetros corretos. Neste caso, foi configurado os valores de *BaudRate:*1250000 (Parâmetro fixo do MCC1); *Polling* = 100ms; *Maxlines* = 64; *Plug-ins* = *hex view*. Configurando o software para transmissão de 64 bytes em hexadecimal, é possível verificar a transmissão. A Figura 18 apresenta a configuração do Termite.

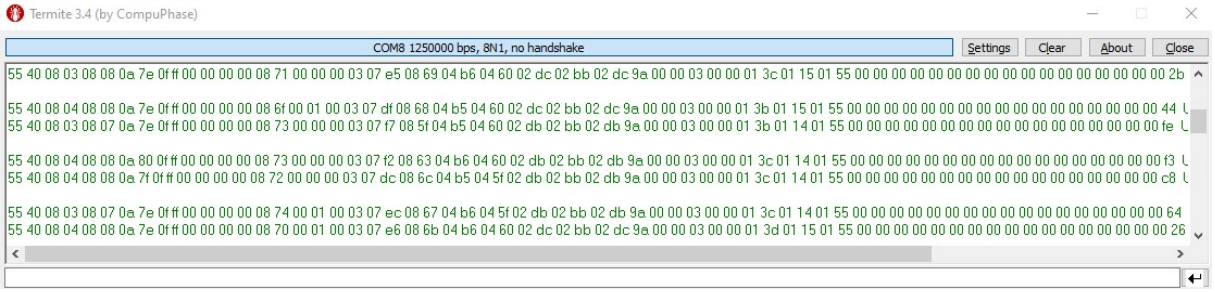
Figura 18 - Configuração do Termite



Fonte: o autor (2023)

Pode-se visualizar a transmissão dos 64 Bytes em cada linha na Figura 19. Verifica-se também que os dois primeiros bytes, referente ao *Header*, são os valores esperados. Isto é, Byte 0 = 55 (hexadecimal) e Byte 1 = 40 (hexadecimal). Dada a visualização feita com o Termite, foi possível concluir que a implementação da transmissão ocorreu sem nenhum problema.

Figura 19 - Testes no Termite



Fonte: o autor (2023)

3.2.2 Programação do supervisório

Por ser uma linguagem que possui versatilidade e bibliotecas bem estabelecidas, o supervisório foi desenvolvido na linguagem Python. O supervisório conta com algumas funcionalidades, como: exportação de dados para um arquivo CSV em tempo de execução; possibilidade de abrir arquivos gerados em outros testes; *plot* dos dados em função do número de amostras; entre outras. Estas funcionalidades serão aprofundadas nos subtópicos a seguir.

3.2.2.1 Bibliotecas utilizadas para o supervisório

Foram utilizadas um conjunto de bibliotecas que permitiram a amostragem correta dos dados recebidos do MCC1. Abaixo está uma breve descrição das bibliotecas e de para que foram utilizadas no programa.

- *Biblioteca PySerial:* Ela encapsula o acesso à porta serial, fornecendo funções para *Python* rodando em Windows, OSX, Linux. Foi usada com a finalidade de obter

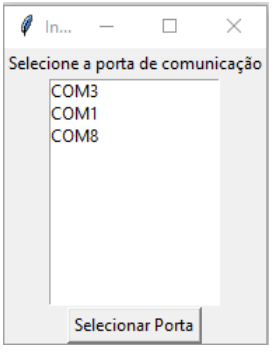
acesso aos dados via comunicação serial, para determinar o *baudrate* da transmissão e qual porta seria empregada.

- *Biblioteca TKinter:* Oferece ferramentas para o desenvolvimento de interfaces gráficas. É ela que permitiu a criação da interface.
- *MatPlotLib:* A ferramenta possui recursos de plotagem para a criação de gráficos 2D. Foi utilizada para a criação dos gráficos dos dados.
- *DateTime:* Possui classes para manipulação de datas e horas. É utilizada para armazenar o horário atual do computador.
- *Os:* é uma biblioteca padrão útil quando se trata de interagir com o sistema operacional. Foi utilizada para navegar entre diretórios internos.

3.2.2.2 Interface

O supervisório possui duas interfaces gerais: a interface para seleção da porta de comunicação e a interface para a amostragem dos dados. A interface para seleção da porta de comunicação (Figura 20) é uma *label* com uma *listbox*. Com o auxílio da biblioteca Pyserial, a lista é criada com todas as portas que estão se comunicando com o computador. Após a porta ser selecionada, a interface é fechada e a interface principal é executada.

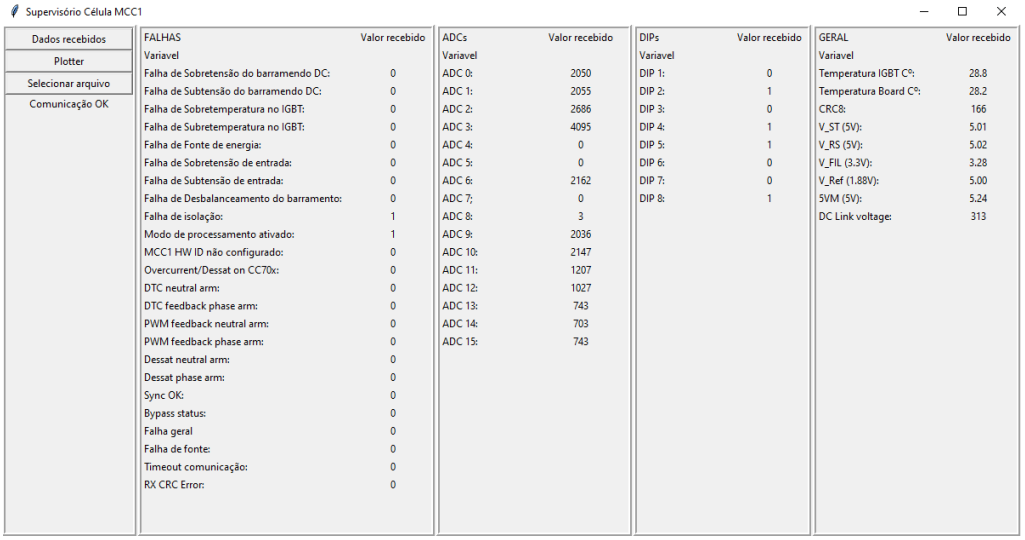
Figura 20 - Interface para seleção da porta de comunicação



Fonte: o autor (2023)

A interface principal é onde todos os dados são apresentados, ela conta com 6 painéis que são dispostos em duas telas. A primeira tela pode ser vista na Figura 21, com a descrição dos seus respectivos painéis.

Figura 21 - Primeira tela da interface principal



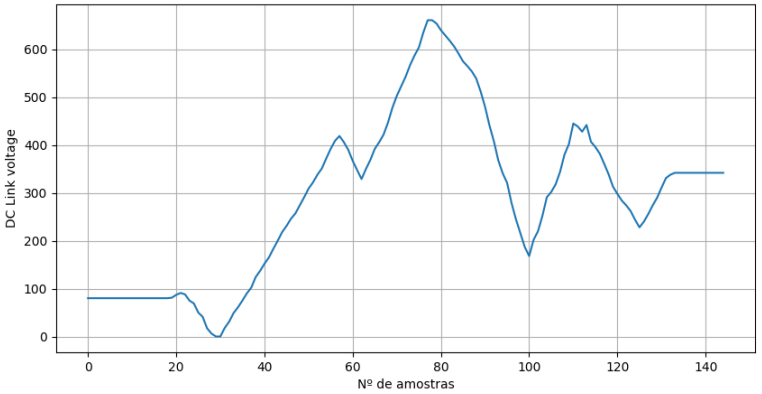
Fonte: o autor (2023)

O primeiro painel é de botões, o segundo de amostragem dos dados da categoria de falhas, o terceiro painel consta a amostragem dos dados da categoria ADCs, o quarto apresenta a amostragem dos dados da categoria DIPs e o quinto a amostragem dos dados da categoria geral. O sexto painel não está na primeira tela, porém é o *plot* do gráfico do dado selecionado em função do número de amostras.

O painel 1 possui três botões: o botão dos “dados recebidos”, que é um gatilho para voltar à tela 1 (caso o usuário esteja na segunda tela); o botão “*plotter”*, que é o gatilho para seleção da tela 2; e, por fim, o botão “selecionar arquivo”, que é um gatilho para seleção de um arquivo CSV externo em vez dos dados que estão sendo transmitidos no momento do teste.

Quando a tela 2 é selecionada pelo botão “*plotter”*, os painéis 2 ao 5 são ocultados e o painel 6 surge no lugar dos mesmos. Como já mencionado, o painel 6 exibe o *plot* do gráfico, em que o eixo X é o número de amostras e o eixo Y representa alguma variável selecionada. A Figura 22 é um exemplo de gráfico que pode ser plotado no painel 6. Valores do gráfico foram obtidos variando manualmente a resistência do potenciômetro acoplado em um dos ADCs da MCC1.

Figura 22 - Exemplo de um gráfico apresentado no sexto painel



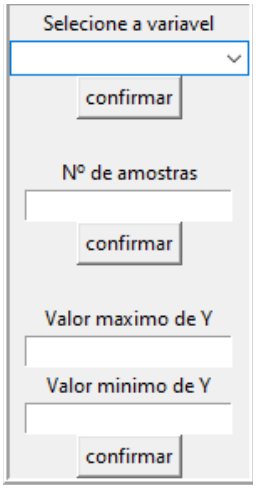
Fonte: o autor (2023)

Junto com isso, quando a tela 2 é selecionada, abre a possibilidade do usuário ajustar valores referente ao gráfico como:

- *Seleção da variável mostrada:* Todos os dados que são transmitidos do MCC1 podem ser plotados.
- *Seleção do fundo de escala (Eixo Y):* É possível ajustar o valor máximo e mínimo no eixo, caso o usuário deseje uma faixa de valor específica.
- *Seleção do número de amostras (Eixo X):* Caso o usuário queira visualizar somente um número específico de amostras, como, por exemplo, as últimas 20 amostras.

Estes ajustes são feitos a partir do painel 1, onde o mesmo recebe as funcionalidades citadas acima, como pode ser vista na Figura 23.

Figura 23 - Modificações no painel 1 ao abrir a tela 2



Fonte: o autor (2023)

3.2.2.2 Verificação dos dados

O cartão de controle envia periodicamente 64 bytes de dados. Como a comunicação é assíncrona, é preciso garantir a sincronização da comunicação. Devido a comunicação possuir chances de variar sua taxa de transferência, é possível que os dados sejam transmitidos com um tempo impreciso, podendo ocorrer um despareamento entre transmissor e receptor.

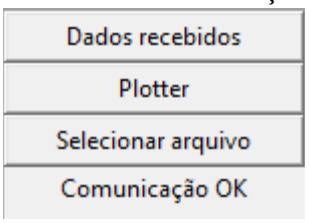
O método utilizado para contornar esse problema foi garantir que os primeiros dois bytes a serem armazenados sejam obrigatoriamente um *header* e somente após esta confirmação o restante dos bytes serão armazenados. Para isso, o programa realiza as duas verificações que estão descritas abaixo. Se as condições forem verdadeiras nessas duas etapas, os dados são armazenados e tratados posteriormente

- *Primeira verificação:* Se faz a leitura de apenas um byte. Se este byte corresponder ao primeiro byte da tabela de comunicação (ou seja, o valor de 85 na base decimal) passa para a próxima verificação.
- *Segunda verificação:* Se faz a leitura de apenas um byte. Se este byte corresponder ao segundo byte da tabela de comunicação (ou seja o valor de 40 na base decimal) é então lido os 62 bytes restantes.

Com a transmissão sincronizada e todos os bytes recebidos, pode-se verificar se os dados não sofreram nenhuma alteração durante a transmissão. Para isso, o cálculo do CRC-8 é feito com base nos primeiros 63 bytes. Logo em seguida, é feita a comparação do resultado com o byte 64. Se os valores divergirem, todos os dados são descartados. Se os valores forem os mesmos, o programa passa para a próxima etapa.

Caso os dados não forem armazenados no arquivo externo CSV, pelos motivos acima, a mensagem “Falha de comunicação” é mostrada no painel 1, informando ao usuário que existe algum problema relativo à comunicação. Se não houver problemas, a mensagem “Comunicação OK” será apresentada, conforme Figura 24.

Figura 24 - Status da comunicação (Painel 1)



Fonte: o autor (2023)

3.2.2.3 Armazenamento dos dados

Os dados recebidos pelo cartão de controle são convertidos para valor decimal e salvos em um vetor. Para salvar permanentemente os dados recebidos, o vetor é gravado em um arquivo externo no formato CSV.

O arquivo externo sempre é criado no momento de execução do programa, em uma pasta chamada "*output*", com o nome gerado a partir da data do computador, impedindo assim que dois arquivos sejam criados com o mesmo nome. Toda vez que o pacote de dados é recebido e validado, os valores são enviados a este arquivo externo a fim de ter um histórico.

3.2.2.4 Tratamento dos dados

Depois que os dados são enviados ao arquivo externo para gravação, é necessário pegá-los novamente e armazená-los em um vetor para o tratamento dos dados. Para os painéis 2 ao 5, somente é mostrado em tela a última amostragem (linha) do arquivo CSV. O painel 6 (*plot* de gráfico) utiliza todas as amostras do arquivo para mostrar o parâmetro escolhido em função do número de amostras. Para uma melhor organização, os dados são divididos em 4 categorias (Quadro 3).

Quadro 3 - Categorias de organização dos dados

Categoria	**Dados**	**Painel**
Categoria 1	Dados das Falhas	Painel 2
Categoria 2	Dados dos ADCs	Painel 3

Categoria 3	Dados dos DIPs	Painel 4
Categoria 4	Dados Gerais	Painel 5

Fonte: o autor (2023)

A categoria 1 é referente aos bytes 35 ao 38 (vetor de falhas). Este vetor referencia cada bit a algum tipo de falha, logo é necessário converter os 4 bytes que estão em decimal para binário e concatená-los, como pode ser visto no exemplo apresentado no Quadro 4.

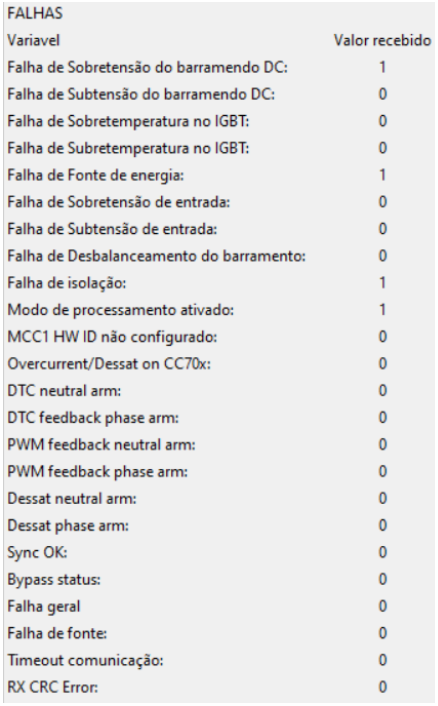
Quadro 4 - Exemplo de conversão e concatenação dos bytes

Bytes 35 ao 38: 15 25 84 0 *(convertidos para binário e concatenados)*
00001111 (15) + 00011001 (25) + 01010100 (84) + 0000000 (0) = 00001111000110010101010000000000

Fonte: o autor (2023)

Com os bits separados em um vetor de bits de falhas, os valores são devidamente associados às suas respectivas falhas e mostrados no painel 2, como pode-se ver na Figura 25.

Figura 25 - Dados das Falhas (Painel 2)



Fonte: o autor (2023)

A categoria 2 refere-se aos ADCs dos bytes 2 ao 33 (vetor de ADCs). Como são necessários 2 bytes para cada ADC, é preciso transformar os valores de dois bytes para um único valor de ADC. A equação utilizada é apresentada a seguir:

$$Byte_x + Byte_{(x+1)}.2^8 = Valor_{ADC} \quad (2)$$

onde o $Byte_x$ é o menos significativo e o $Byte_{(x+1)}$ é o mais significativo. Obtendo assim, os valores finais dos ADCs convertidos para decimal, conforme a Figura 26.

Figura 26 - Dados dos ADCs (Painel 3)

ADCs	
Variavel	Valor recebido
ADC 0:	2051
ADC 1:	2045
ADC 2:	2689
ADC 3:	1541
ADC 4:	0
ADC 5:	0
ADC 6:	2122
ADC 7;	0
ADC 8:	4
ADC 9:	1876
ADC 10:	2092
ADC 11:	2227
ADC 12:	4034
ADC 13:	660
ADC 14:	653
ADC 15:	660

Fonte: o autor (2023)

A categoria 3 refere-se aos DIPs, que estão no byte 3. O processo de tratamento é similar ao da categoria 1, isto é, converte-se o byte de decimal para binário e em seguida, atribui-se cada bit ao seu respectivo DIP. O resultado é apresentado na Figura 27.

Figura 27 - Dados dos DIPs (Painel 4)

DIPs	
Variavel	Valor recebido
DIP 1:	0
DIP 2:	0
DIP 3:	1
DIP 4:	0
DIP 5:	0
DIP 6:	1
DIP 7:	0
DIP 8:	0

Fonte: o autor (2023)

Na categoria 4 constam os valores de determinadas medições para análise técnica. A Tabela 4 representa onde cada valor é recebido.

Tabela 4 - Parâmetros referente aos Dados Gerais (Painel 5)

Byte	Variavel	Valor recebido
42 e 43	Temperatura ambiente Cº	Temperatura ambiente
40 e 41	Temperatura Board Cº	Temperatura Board
63	CRC8	CRC8
2 e 3	V_ST (5V)	ADC 0
4 e 5	V_RS (5V)	ADC 1
6 e 7	V_FIL (5V)	ADC 2
8 e 9	V_Ref (5V)	ADC 3
22 e 23	5VM (5V)	ADC 10
44 e 45	DC Link Voltage	DCLink

Fonte: o autor (2023)

A definição das variáveis de tensão (com exceção do *DC Link Voltage*) é de 12 bits. Dessa forma, há a necessidade de escalonar os valores. Para isso, utiliza-se a equação apresentada a seguir:

$$\frac{(Adc_{decimal} \cdot V_{ref} \cdot 2)}{2^{12}} = T_{tensão} \qquad (3)$$

onde V_{ref} (tensão de referência) é de 5V e o $Adc_{decimal}$ refere-se ao valor convertido. Obtendo assim, o valor $T_{tensão}$ em Volts (V). Após a conversão dos valores, pode-se visualizar o painel 5 completo (Figura 28).

Figura 28 - Dados Gerais (Painel 5)

GERAL	
Variavel	Valor recebido
Temperatura IGBT C°:	24.0
Temperatura Board C°:	26.8
CRC8:	94
V_ST (5V):	5.01
V_RS (5V):	4.99
V_FIL (3.3V):	3.28
V_Ref (1.88V):	1.88
5VM (5V):	5.11
DC Link voltage:	1231

Fonte: o autor (2023)

3.2.2.5 Funcionalidades

Com a implementação do supervisório, o usuário possui as seguintes funcionalidades:

- *Seleção da porta de comunicação*: para a correta comunicação, a porta do conversor de fibra óptica para USB precisa ser selecionada. Caso a porta não seja a correta, o usuário precisa informar novamente o endereço da porta.
- *Visualizar parâmetros em tempo real*: na tela principal, o usuário pode visualizar todos os parâmetros transmitidos pelo MCC1 que foram tratados. Os valores exibidos são em relação à última amostra recebida.
- *Visualizar gráfico de cada parâmetro em função das amostras:* caso o usuário queira a visualização de uma variável específica em função das amostras, a tela 2 implementa esta funcionalidade. Esta tela permite configurar qual variável o usuário deseja utilizar e escolher o valor máximo e mínimo no eixo Y e o número de amostras do eixo X.
- *Visualizar arquivos gerados em outros momentos:* caso seja de preferência do usuário, o mesmo pode abrir um arquivo CSV já existente e visualizar os

parâmetros na tela 2. A gravação de amostras do teste atual ainda funciona em segundo plano, não interferindo um arquivo no outro.

4 CONSIDERAÇÕES FINAIS

Ao compreender detalhadamente o processo de comunicação da célula do inversor com o CCE, foi possível especificar os requisitos para a implementação do supervisório. Tendo em vista a necessidade de melhoria nos processos de testes já existentes do inversor, o trabalho realizado alcançou seu objetivo geral de criar um supervisório para a visualização e armazenamento de parâmetros nos testes das células, auxiliando assim no processo atual de validação dos inversores de média tensão.

A proposta final dispõe-se de um supervisório implementado na linguagem Python que possui duas telas principais, uma contendo a visualização em tempo real dos parâmetros e a outra envolvendo a visualização dos dados em um gráfico de linha em função do número de amostras que foram previamente salvas no arquivo CSV. Além da possibilidade de abrir arquivos de testes feitos anteriormente, caso necessário.

Para a realização dos ensaios durante a implementação, se fez necessário produzir uma bancada de testes contendo o módulo de cartão de controle das células (MCC1), os cartões de alimentação para o módulo, a placa de conversão de fibra óptica para USB, um potenciômetro e dois sensores de temperatura.

O primeiro passo foi confeccionar um protótipo para simular fisicamente uma das células do inversor em funcionamento. No tópico que debruça sobre o desenvolvimento do hardware, foi apresentado como se deu a alimentação das placas de circuito. Em seguida, buscou-se analisar os parâmetros do inversor e das células, necessitando do estudo do software interno do MCC1.

Somente então, foi possível programar a interface de comunicação entre a célula e o supervisório. Para isso, foi preciso organizar o vetor de transmissão e aplicar alguns métodos de detecção de erros intrínsecos à comunicação, como por exemplo o CRC-8. Posteriormente, criou-se a tela de visualização dos parâmetros em tempo real e a tela de visualização dos parâmetros em interface gráfica. A implementação destas duas telas permite que os dados relevantes sejam facilmente visualizados pelo operador que aplica os testes.

Para garantir o armazenamento correto dos dados, os parâmetros da comunicação foram armazenados em um arquivo CSV e criou-se a possibilidade de visualizar os arquivos CSV gerados anteriormente diretamente no supervisório.

Por fim, com base nos objetivos alcançados, a expectativa é que o supervisório sugerido possibilite uma melhoria relevante na realização dos testes, oferecendo uma ferramenta para o monitoramento das células. Dessa forma, ele contribuirá para o progresso tecnológico e aperfeiçoamento contínuo dos testes que envolvem o inversor de média tensão MVW3000.

5 REFERÊNCIAS

GAIKWAD, Asha; ARBUNE, Pallavi Appaso. Study of cascaded H-bridge multilevel inverter. In: INTERNATIONAL CONFERENCE ON AUTOMATIC CONTROL AND DYNAMIC OPTIMIZATION TECHNIQUES (ICACDOT), 2016, Pune. **Proceedings**. Pune: International Institute Of Information Technology (I²It), 2016. p. 179-182.

KUROSE, Jim; ROSS, Keith. Redes de computadores e a internet: uma abordagem top-down. 6. ed. São Paulo: Pearson Education do Brasil Ltda, 2013. 658 p.

LADDHA, Neha R.; THAKARE, A. P.. A Review on Serial Communication by UART. International Journal Of Advanced Research In Computer Science And Software Engineering. Amravati, p. 366-369. 1 jan. 2013.

NANDA, Umakanta; PATTNAIK, Sushant Kumar. Universal Asynchronous Receiver and Transmitter (UART). **International Conference On Advanced Computing And Communication Systems (ICACCS)**, Coimbatore, v. 3, p. 1-5, 2016.

PEÑA, Eric; LEGASPI, Mary Grace. UART: A Hardware Communication Protocol Understanding Universal Asynchronous Receiver/Transmitter. **Analog Dialogue**, United States, v. 54, n. 4, p. 1-5, 2020. Trimestral.

PETERSON, Larry L.; DAVIE, Bruce S.. **Redes de computadores uma abordagem de sistemas**. 5. ed. Rio de Janeiro: Elsevier Editora Ltda, 2013. 554 p.

TOMASI, Wayne. Advanced Electronic Communications. England: Pearson New International Edition, 2013. 621 p.

WALTRICH, Gierri. **Estudo e implementação de um inversor multinível trifásico em cascata empregando sub-células de comutação**. 2009. 158 f. Dissertação (Mestrado) - Curso de Mestrado em Engenharia Elétrica, Universidade Federal de Santa Catarina, Florianópolis, 2009.

WEG. **MVW3000**: inversor de frequência de média tensão. 2023. Disponível em: https://static.weg.net/medias/downloadcenter/hbd/ha2/WEG-MVW3000-catalogo-50071946-pt.pdf. Acesso em: 27 nov. 2023.

WU, Bin. **High-power converters and ac drives**. New Jersey: Ieee, 2006. 333 p.