UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE AUTOMAÇÃO E SISTEMAS
CURSO DE GRADUAÇÃO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Lucas Eduardo O. Cândido

Desenvolvimento do firmware de um módulo de extensão de medição de energia

Florianópolis
2021

Lucas Eduardo O. Cândido

Desenvolvimento do firmware de um módulo de extensão de medição de energia

Relatório final da disciplina DAS5511 (Projeto de Fim de Curso) como Trabalho de Conclusão do Curso de Graduação em Engenharia de Controle e Automação da Universidade Federal de Santa Catarina em Florianópolis.
Orientador: Prof. Werner Kraus Jr, Dr.
Supervisor: Sandro Alberton Kirchner, Eng.

Florianópolis
2021

Ficha de identificação da obra

A ficha de identificação é elaborada pelo próprio autor.

Orientações em:

http://portalbu.ufsc.br/ficha

Lucas Eduardo O. Cândido

Desenvolvimento do firmware de um módulo de extensão de medição de energia

Esta monografia foi julgada no contexto da disciplina DAS5511 (Projeto de Fim de Curso) e aprovada em sua forma final pelo Curso de Graduação em Engenharia de Controle e Automação

Florianópolis, 29 de setembro de 2021.

Prof. xxxx, Dr.
Coordenador do Curso

Banca Examinadora:

Prof. xxxx, Dr.
Orientador
UFSC/CTC/DAS

xxxx, Eng.
Supervisora
Empresa/Universidade xxxx

Prof. xxxx, Dr.
Avaliador
Instituição xxxx

Prof. xxxx, Dr.
Presidente da Banca
UFSC/CTC/DAS

Este trabalho é dedicado aos meus colegas de trabalho, meus amigos, minha namorada, meu irmão e aos meus queridos pais.

RESUMO

Este PFC trata do desenvolvimento de uma extensão IoT de medição de energia capaz de medir distorções harmônicas para ser utilizada em transformadores com um *endpoint standalone* dotado de conectividade *mobile*. A medição é feita usando transformadores de corrente (TCs) com saída em corrente e com o uso de um circuito integrado (CI) para este propósito. O conjunto também conta com um acelerômetro para medir inclinação e detectar impactos. A leitura de distorções harmônicas é o principal indicativo de qualidade de energia distribuída. O desenvolvimento inclui validação do *hardware*, conversão de protocolos eletrônicos de comunicação, desenvolvimento de rotinas de leituras de registradores, interpretação de dados e geração de pacotes de medição. O escopo deste projeto inclui também o desenvolvimento do *driver* da extensão, ou seja, a parte do código escrita no *endpoint standalone* para que este se comunique devidamente com a extensão e efetue todas suas medições.

Palavras-chave: Internet das coisas. Sistemas embarcados. Medição de energia. Firmware.

ABSTRACT

This end of course project is about the development of a harmonic distortion measuring energy meter IoT extension module to be used with a standalone endpoint with mobile connection. The measurement is made with the use of current transformers (CTs) with current as output and with the use of an integrated circuit (IC) for the measuring purpose. The set also features an accelerometer capable of inclination measuring and tap detection. The measurement of harmonic distortions is the main indicator of delivered energy quality. The development includes hardware verification, electronic communication protocols conversion, development of routines for registers readings, data parsing and measurement packages generation. The project scope also covers the development of the extension module driver, that is, the piece of code wich is written in the standalone endpoint firmware so that it can successfully communicate with the extension module and read its measurements.

Keywords: Internet of Things. Embedded systems. Energy measurement. Firmware.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

1.0.1 A Internet das coisas e a Khomp

A internet das coisas é citada pelo instituto Gartner como uma das maiores tendências do milênio (INSTITUTE, 2021). Trata-se de um paradigma de automatização que visa integrar o mundo real ao virtual, fazendo com que mudanças ou eventos no mundo real sejam observáveis no mundo virtual e ações no mundo virtual se reflitam no mundo real. Nos tempos atuais isso significa a integração à nuvem de sensores e atuadores, fazendo com que as medições daqueles possam ser acessadas e estes podem ser comandados através da rede virtual.

No cenário atual esse tipo de solução encontra-se em monitoramentos de temperatura (especialmente em aplicações hospitalares), de abertura ou fechamento de portas ou válvulas, intensidade sonora e ruídos e, no que concerne este PFC, consumo e qualidade de energia.

A Khomp é uma empresa brasileira fundada em 1996 que desenvolve soluções em telecomunicações. Em sua criação a empresa desenvolvia produtos focados em comunicação telefônica, contanto atualmente com uma extensa linha de *gateways* de telefonia. A partir de 2017 a Khomp começou a atuar e foi pioneira no Brasil no mercado de Internet das coisas, trabalhando, nesse mercado, principalmente com as redes LoraWan (ALLIANCE, 2015) IEE 802.15.4 (Zigbee) e móvel.

Na última década, com o advento da internet das coisas, surgiram no mercado tecnologias para satisfazer a demanda do setor crescente. Entre elas destacam-se as redes SigFox, LoraWan, IEEE 802.15.4 e recentemente até as operadoras de telefonia móvel passaram a integrar este mercados com as tecnologias CAT-M1 (4G) e NB - IoT (*narrow band*).

A Khomp, como descrito acima, desenvolve soluções nas tecnologias LoraWan, Zigbee e redes móveis. Em linhas gerais a rede LoraWan usa uma topologia em estrela, onde vários dispositivos (*endpoints*) estão ligados a um roteador central (*gateway*) e através desse à um servidor na internet. A rede Zigbee, por sua vez, usa uma topologia em malha, onde alguns dispositivos intermediários podem se comportar como repetidores, para garantir que todos estejam conectados ao *gateway* e, novamente, através dele integrados à internet. Por fim os dispositivos de tecnologia *mobile* recebem a nomenclatura de *standalone*, pois não precisam de um *gateway* para conectarem-se à rede.

Além do desenvolvimento de *endpoints* das tecnologias supracitadas, o setor de Internet das Coisas da Khomp também desenvolve dispositivos que se conectam a esses *endpoints* e integram-lhes novas funcionalidades de monitoramento ou de atuação. Tais dispositivos são chamados de Extensões de Medição. Este PFC trata do desenvolvimento de uma extensão de medição de qualidade e consumo de energia

em sistemas trifásicos para ser integrada primordialmente ao *endpoint standalone* da Khomp também a ser lançado, o ITS 402.

1.0.2 A medição de energia usando transformadores de corrente (TCs)

Desde que a energia começou a ser distribuída à população, a sua medição se tornou uma informação vital para as companhias fornecedoras. Com o advento da internet das coisas, este cenário de monitoramento também entrou no escopo das soluções desenvolvidas. Integrando a medição de energia à nuvem, o consumidor pode ter uma percepção diária da quantidade de energia que está sendo consumida por sua residência ou seu estabelecimento. No contexto deste PFC, a demanda atendida é a do monitoramento de transformadores de distribuição.

Dentre os métodos de medição de energia, o uso de transformadores de corrente (TCs, doravante) constitui um meio não invasivo, permitindo que suas instalação e manutenção, caso necessária, requeiram pouca ou nenhuma intervenção no sistema já existente de distribuição energética. Neste método, ilustrado na figura 1, um transformador é colocado no canal onde a corrente a ser medida trafega. Em seu secundário surgirá uma corrente de menor intensidade que será medida diretamente pelo sistema de medição.

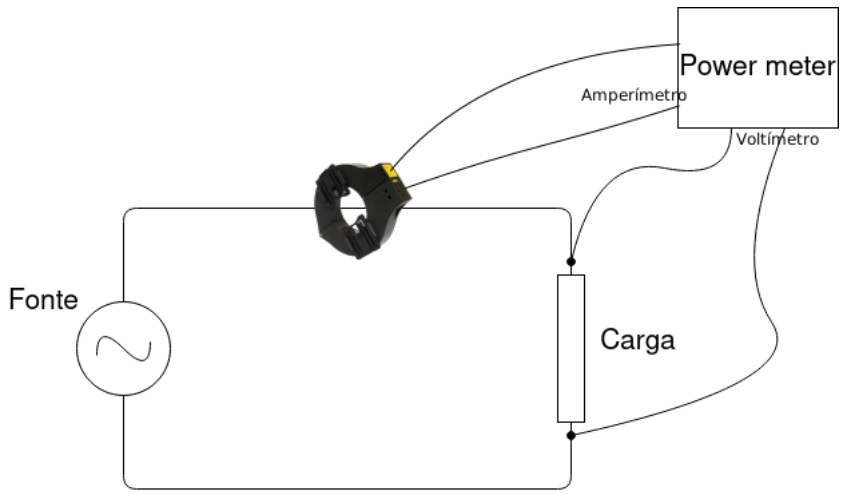

Figura 1 – Medição de energia com transformador de corrente (Autoral)

Conhecendo-se a relação entre as correntes no primário e no secundário do transformador, conhece-se a relação entre o consumo energético estimado pelo sistema de medição e o consumo real da carga em questão. Neste PFC, além do consumo energético há o interesse do cliente pela medição de distorções harmônicas (a serem explanadas mais à frente) para um monitoramento da qualidade da energia entregada e de possíveis riscos ao sistema de distribuição.

Por fim, como o objetivo do *kit* é ser instalado em postes, o sistema contará com um acelerômetro integrado para medição de inclinação e detecção de impactos, além

de GPS, medição de temperatura e integração com sensores de contato (magnéticos). No escopo deste PFC estão as integrações das medições elétricas e do acelerômetro. As integrações com sondas de temperatura, GPS e sensores magnéticos são nativas do *endpoint standalone*.

O projeto é realizado em parceria com uma empresa de Joinville. Ela tem por vocação integrar as soluções em IoT aos setores da indústria onde tais soluções podem ser aplicadas, um papel que recebe o nome de integrador. Foi ela quem verificou a oportunidade de negócio em medidores de energia em transformadores e trouxe para a Khomp a demanda.

1.1 OBJETIVOS

Este projeto visa atender a demanda supracitada das distribuidoras de energia por um dispositivo de monitoramento integrado à internet. Assim ele se divide em duas partes: desenvolvimento da extensão de medição (doravante EM-T103 ou apenas EM) e desenvolvimento do *driver* de comunicação para fazer parte do *firmware* do *endpoint standalone* (doravante ITS, ou ITS NB).

O ITS conta nativamente com a possibilidade de atualização do seu *firmware* de maneira remota ou local, por isso decidiu-se que tanto quanto realizável, o processamento dos dados será feito nesse dispositivo. Isto significa que o microcontrolador na extensão deve ter o comportamento de um intermediário entre o ITS e o circuito integrado (doravante CI) de medição elétrica (doravante *power meter*. O *design* do *hardware* e seleção dos componentes da extensão foi feito pelo engenheiro responsável, assim o escopo deste projeto de fim de curso é constituído por:

- Desenvolvimento do *firmware* da EM
- Integração do ITS com o *power meter* através o MCU intermediário
- Integração do ITS com o acelerômetro também presente na EM-T103
- Validação experimental das grandezas medidas e calibrações
- Geração de um pacote personalizado dos dados coletados pela EM (a ser discorrido na seção 4.1.2.4);

2 CONCEITOS BÁSICOS

2.1 O PARADIGMA DAS SOLUÇÕES IOT

Embora existam diversas opções de rede no contexto da internet das coisas, as soluções seguem sempre uma diretriz: um conjunto de *endpoints* monitora um cenário e envia seus dados a um servidor na nuvem, seja diretamente, como no caso dos dispositivos *standalone*, seja através de um *gateway*, como nas redes Zigbee e LoRaWan. O servidor na nuvem disponibiliza esses dados para o cliente final, geralmente na forma de gráficos, estatísticas e alarmes. Caso os dispositivos em campo permitam, também é possível enviar comandos de atuação ou configuração para eles, nestes casos o usuário faz isso a partir do servidor na nuvem. A figura 2 ilustra a organização dos elementos de uma aplicação IoT.

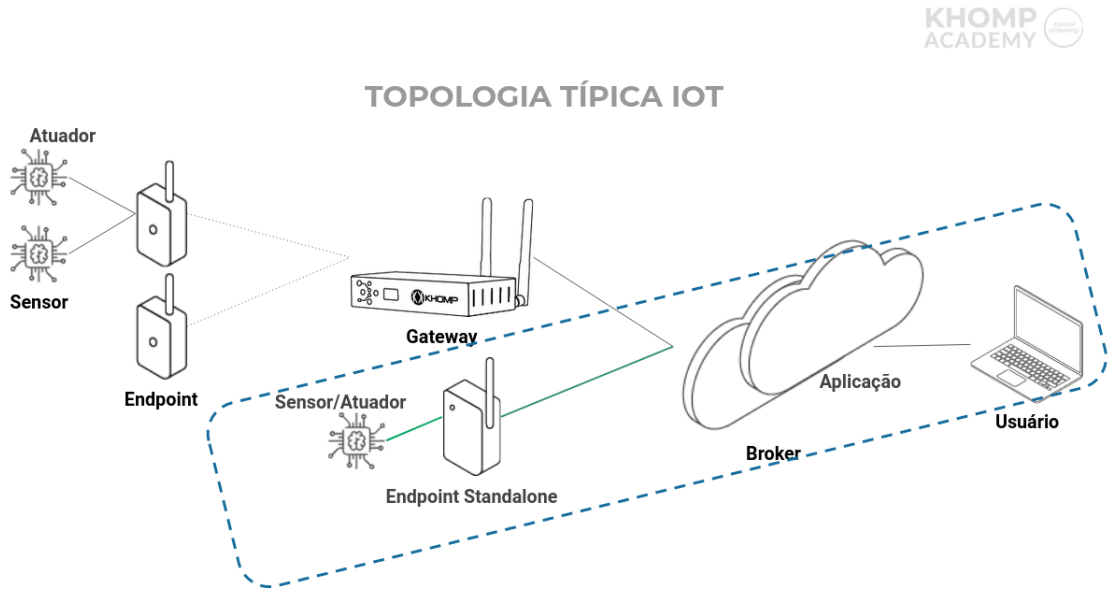

Figura 2 – Topologia das aplicações IoT

Na Khomp apenas são desenvolvidos os dispositivos de campo, *endpoints* e *gateway*, ficando o desenvolvimento da aplicação em *cloud* a cargo do integrador, neste caso, a empresa parceira. Para a comunicação com a aplicação de *cloud* o principal protocolo utilizado é o MQTT (*message queue telemetry transport*), que é o utilizado pelo ITS.

2.2 A IMPORTÂNCIA DA MEDIÇÃO DE COMPONENTES HARMÔNICAS

A rede de energia é feita para operar a 50 ou 60 Hz, dependendo da região do mundo. Entretanto a presença de cargas não lineares pode fazer com que as curvas de tensão e corrente desviem seus comportamentos no tempo de senoides

e tenham outras formas mais diversas. O matemático Jean Baptiste Joseph Fourier demonstrou em 1807 que qualquer curva cíclica em um período pode ser escrita como uma somatória de senos e cossenos com frequências de uma oscilação por período e frequências múltiplas naturais dessa. À frequência de uma oscilação por período dá-se o nome de fundamental, enquanto às múltiplas, o nome de harmônicas. Portanto o estudo de formas de onda diversas é feito através da sua decomposição em senoides (e cossenoides) nas frequências fundamental e harmônicas.

Como um exemplo do efeito da presença de harmônicas em um circuito, Mack Grady em Understanding Power System Harmonics (2012) (GRADY, 2012) apresenta o circuito da figura 3:

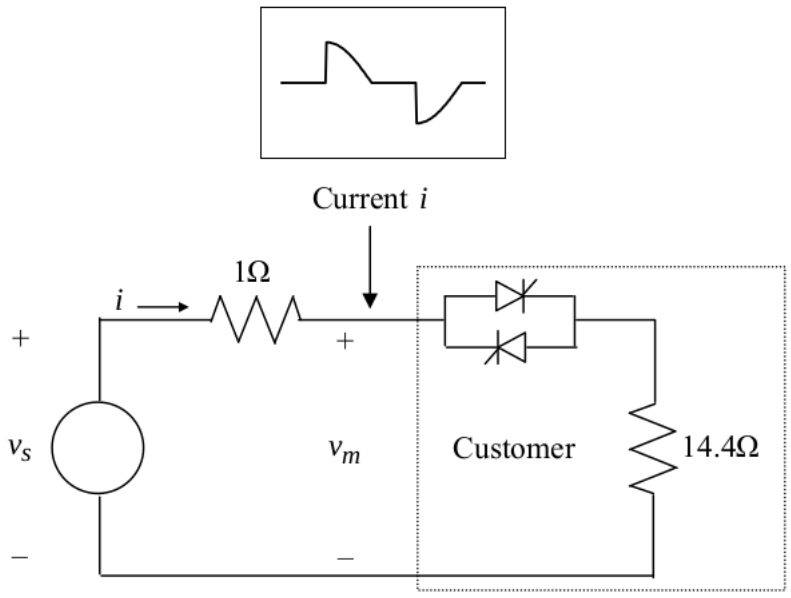

Figura 3

Uma fonte de tensão senoidal de 120 V RMS alimenta uma rede com resistência interna de 1 Ohm e uma lâmpada de 14, 4 Ohms, 100 Watts. A lâmpada é acompanhada de um controlador via triacs, operando com um ângulo de ativação de 90°, de tal modo que a lâmpada opere com metade do consumo. Usando $v_S(t) = 120\sqrt{2}sin(\omega_1 t)$ temos a série de Fourrier para a corrente no circuito, truncada na quinta ordem:

$$i(t) = 6{,}99\sin(\omega_1 t - 32{,}5^\circ) + 3{,}7\sin(3\omega_1 t - 90{,}0^\circ + 1{,}25\sin(5\omega_1 t - 90{,}0^\circ)$$

Para a tensão v_m no conjunto da lâmpada (lâmpada e controlador a triacs) temos:

$$v_m(t) = v_S(t) - Ri(t) = 120\sqrt{2}\sin(\omega_1 t) - 1(6{,}99\sin(\omega_1 t - 32{,}5^\circ) + 3{,}7\sin(3\omega_1 t - 90{,}0^\circ) + \\ + 1{,}25\sin(5\omega_1 t - 90{,}0^\circ))$$

$$v_m(t) = 163{,}8\sin(\omega_1 t + 1{,}3^\circ) + 3{,}7\sin(3\omega_1 t - 90{,}0^\circ) + +1{,}25\sin(5\omega_1 t - 90{,}0^\circ) \quad (1)$$

E por fim a potência média desenvolvida pelo circuito é

$$P_{avg} = \frac{163,8 \cdot 6,99}{2} \cos(1,3^\circ - (-32,5^\circ)) + \frac{3,75 \cdot 3,75}{2} \cos(90^\circ - (-90^\circ)) + \\ + \frac{1,25 \cdot 1,25}{2} \cos(90^\circ - (-90^\circ)) \\ P_{avg} = 475,7 - 7,03 - 0,78 = 467,9W \tag{2}$$

Nota-se que a cada ciclo a potência média desenvolvida pelo circuito tem 3 componentes: uma na frequência fundamental, uma na terceira e uma na quinta harmônica. Entretanto essas duas últimas têm sinal negativo, o que significa que nestas frequências, o circuito está cedendo estas potências, que estão sendo dissipadas na rede. Este comportamento é parasita e indesejado, entre as consequências que a presença de harmônicas pode causar em um circuito estão (GRADY, 2012):

- Ressonância: quando as correntes harmônicas geradas por cargas não lineares interagem com a impedância do sistema, resultando em altas tensões nas frequências harmônicas;
- Ruídos em cargas sensíveis: algumas cargas controladas por computadores podem ser sensíveis a distorções harmônicas maiores que 10%;
- Degradação de capacitores: Como a impedância de um capacitor diminui com a frequência, tensões em altas harmônicas podem gerar correntes altamente prejudiciais a estes componentes.

Por esses motivos a medição de distorções harmônicas é de grande interesse para companhias de distribuição de energia e um atributo chave da EM T103.

2.3 A EM-T103, EXTENSÃO DE MEDIÇÃO DE ENERGIA

Conforme supracitado o projeto surgiu para atender a demanda de um medidor de energia integrado à rede, capaz de operar autonomamente uma vez ligado à energia, de ser configurado remota ou localmente e de operar no alcance da rede móvel Cat-M1 (4G). Como um dispositivo já em desenvolvimento atendia a algumas destas especificações, a EM T103 contemplou as que restavam. Ela possui três núcleos principais: o *power meter*, o microcontrolador e o acelerômetro, ilustrados na figura 4.

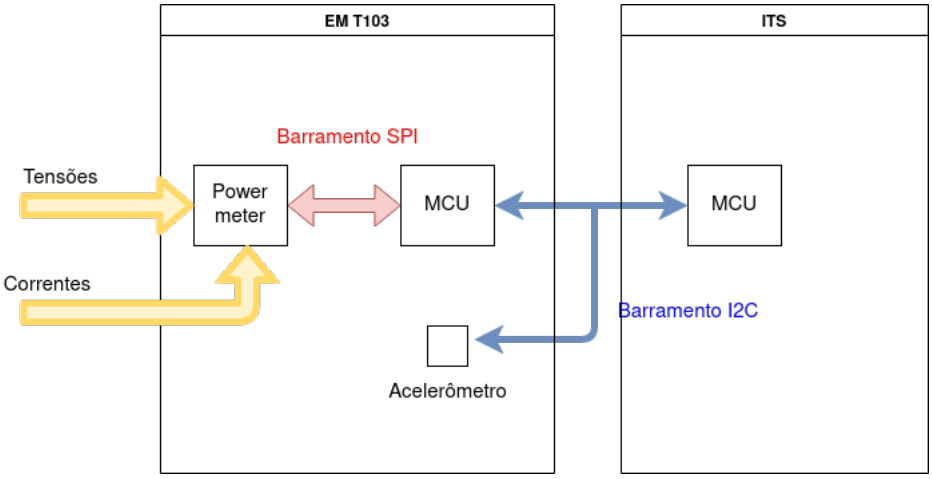

Figura 4 – Esquema lógico de núcleos da EM T103

O *hardware* foi projetado pelo engenheiro responsável e a empresa com quem o projeto é feito em parceria se responsabilizou pelo invólucro do kit. As especificações do projeto são:

- Funcionalidades
  - Alimentação fornecida pela rede de energia
  - Permite até 3 TCs medindo corrente, bornes medindo tensão
  - Acelerômetro simples permitindo medições de XYZ
  - Medição de taxas harmônicas até nona ordem
  - Tratamento de verniz na PCB
  - Gabinete protegido para o ambiente externo com IP65
- Restrições:
  - Medição trifásica exclusivamente a 4 fios (3 fases + neutro)
- Premissas:
  - Usa o ITS NB/CAT M atual com SIM card da Arquia/Vivo na comunicação,
  - Os TCs serão de um parceiro neste projeto.
  - O invólucro do produto será feito pelo parceiro e poderá ser fornecido depois para a Khomp
  - A aplicação em nuvem será do integrador parceiro deste Projeto

2.4 METODOLOGIA DE DESENVOLVIMENTO

Assim como a maior parte dos *endpoints* do setor de internet das coisas, o *firmware* da EM T103 é relativamente pequeno e portanto requer um ou no máximo dois desenvolvedores trabalhando nele. Por isso procura-se usar métodos ágeis para o desenvolvimento e neste caso o método usado foi o *scrum* com *sprints* semanais. A ferramente *Redmine* foi usada para documentação das tarefas e das etapas do desenvolvimento e a ferramenta *git* foi usada com a plataforma *Git Lab* para o versionamento do código.

Nas *sprints* semanais eram avaliadas a etapa atual do desenvolvimento e os problemas atualmente enfrentados, caso existisse algum. Nestes casos também eram propostas diretivas para contorná-los ou superá-los, além de planejamento das etapas seguintes. O fluxo do desenvolvimento pode ser visto na figura 5:

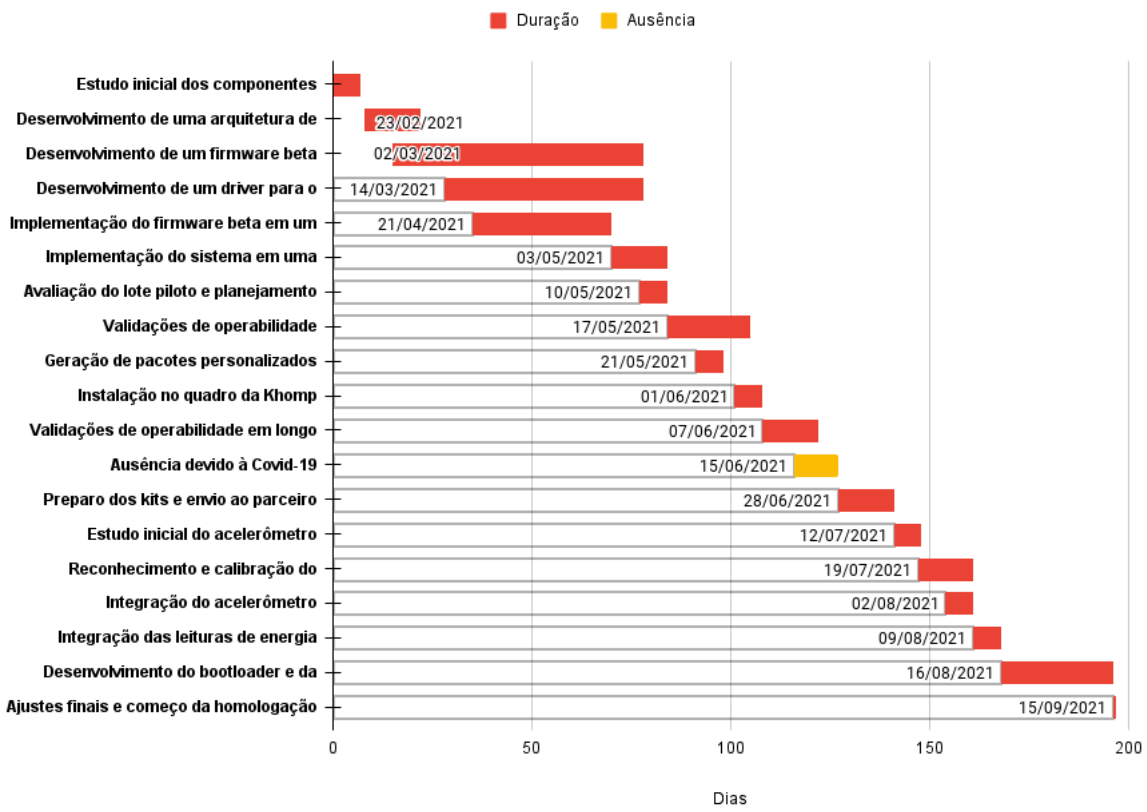

Figura 5 – Desenvolvimento

Etapas:

- Estudo inicial dos componentes
- Desenvolvimento de uma arquitetura de software
- Desenvolvimento de um firmware beta

- Desenvolvimento de um driver para o sistema
- Implementação do firmware beta em um hardware piloto e do driver no sensor standalone
- Implementação do sistema em uma versão inicial do hardware
- Avaliação do lote piloto e planejamento de correções/melhorias
- Validações de operabilidade
- Geração de pacotes personalizados
- Instalação no quadro da Khomp
- Validações de operabilidade em longo tempo
- Ausência devido à Covid-19
- Preparo dos kits e envio ao parceiro
- Estudo inicial do acelerômetro
- Reconhecimento e calibração do acelerômetro
- Integração do acelerômetro
- Integração das leituras de energia
- Desenolvimento do bootloader e da atualização de firmware

3 DESENVOLVIMENTO

3.1 ETAPAS DO DESENVOLVIMENTO

3.1.1 Estudo inicial dos componentes

Esta etapa consistiu em um estudo do *datasheet* do *power meter* utilizado e verificação das medições disponibilizadas por ele, além do entendimento de sua interface de comunicação SPI. Ela começou na segunda semana de fevereiro e durou uma semana. Um documento foi escrito descrevendo as principais funcionalidades do CI. Elas são:

- Medições: o *power meter* disponibiliza, em cada fase, as leituras
  - Tensão RMS
  - Corrente RMS
  - Fator de potência
  - Potência aparente média
  - Potência ativa média
  - Potência reativa média
  - Ângulo de fase
  - Análise harmônica
    - Potência média ativa total na frequência fundamental
    - Potência média ativa total nas frequências harmônicas
- Mensuração de energia: o *power meter* calcula energia consumida e disponibiliza a leitura nas modalidades:
  - ativa direta
  - ativa inversa
  - reativa direta
  - reativa inversa
  - aparente
  - direta ativa harmônica
  - direta ativa fundamental
  - reversa ativa harmônica
  - reversa ativa fundamental

- Monitoramento: o *power meter* pode ser configurado para gerar interrupções em pinos predefinidos ao detectar:
  - Detecção de queda de tensão
  - Detecção de perda de fase
  - Detecção de sobrecorrente e sobretensão
  - Detecção de variação na frequência de acordo com uma margem configurada
  - Detecção de sobrecorrente na linha neutra
  - Detecção de erro de sequência de fase

3.1.2 Elaboração de um protocolo de comunicação entre os dispositivos

Esta foi a etapa inicial do desenvolvimento da extensão e do *driver* propriamente. Visando garantir que a chamada de escrita tivesse confirmação do valor escrito no *power meter*, foi instituído que após tais chamadas, o microcontrolador da extensão geraria autonomamente uma chamada de leitura do registrador recém escrito e retornaria o valor lido ao ITS. Além disso como o processamento dos dados será feito no ITS, foi requisitado ao engenheiro de *hardware* que o barramento da interrupção ficasse disponível tanto ao ITS quanto à extensão. Assim quando uma interrupção fosse gerada, o ITS teria acesso à informação imediatamente. Por fim foi decidido que haveriam comandos especiais do ITS para a extensão que não se tratariam de chamadas de escrita ou leitura do *power meter*. Estes seriam identificados por bits não utilizados no *payload* enviado ao CI. Este protocolo é ilustrado na figura 6.

Também nessa etapa foi decidido que a extensão contaria com a possibilidade de atualização de *firmware*, que já era presente no ITS. Foi solicitado ao engenheiro de *hardware* que além da comunicação I2C a extensão compartilhasse um canal de UART com o ITS para que futuramente ocorra o desenvolvimento de um *bootloader* e um protocolo para atualização do *firmware* da extensão.

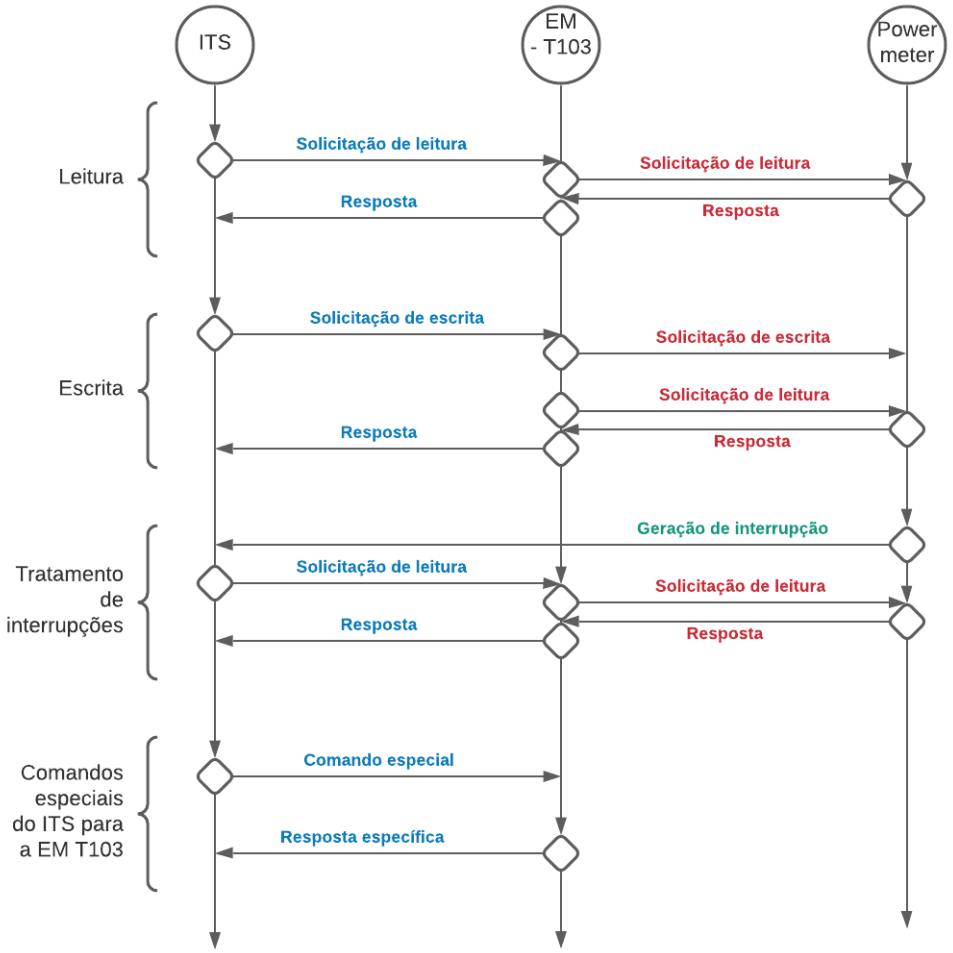

Figura 6 – Protocolo de comunicação entre o ITS e a EM T103

3.1.3 Verificação do *hardware* protótipo

O primeiro *hardware* em que o projeto foi desenvolvido foi um protótipo improvisado utilizando-se dois produtos que o setor já tinha em desenvolvimento: um que utilizava o *power meter* e um que utilizava um microcontrolador do mesmo modelo que seria usado na extensão. A primeira etapa do desenvolvimento foi a validação dos seus canais de comunicação: UART, I2C e SPI.

O canal UART foi verificado imediatamente e usado para *debug* durante todo o desenvolvimento do kit. Na verificação do canal SPI notou-se que o *power meter* estava inativo e foi necessário trocar o dispositivo. Depois disso a consistência do canal foi comprovada através de chamadas ao CI para leitura de tensões conhecidas *a priori*. Para a verificação do canal I2C foi necessário criar no ITS uma chamada à extensão e nesta foi adicionada uma instrução de ouvir o canal e esperar ser chamado. Após confirmar a comunicação, passou-se para a próxima etapa.

Esta validação começou no início de março de 2021 e durou aproximadamente uma semana, devido ao problema com o *power meter*.

3.1.4 Desenvolvimento inicial no *hardware* protótipo

Esta etapa consistiu em desenvolver na extensão o protocolo ilustrado na imagem 6. Durante esse passo, bem como no passo anterior e em todo o desenvolvimento do *firmware* da extensão foi usado o *software Cube MX* fornecido pela *ST Microeletronics* para a geração automática do código referente à inicialização de periféricos e *GPIOs*. Além disso durante todo o desenvolvimento foi usada a placa de desenvolvimento núcleo f411re, também da *ST Microeletronics*, para gravação do *firmware* tanto na extensão quanto no ITS. A figura 7 mostra a lógica implementada.

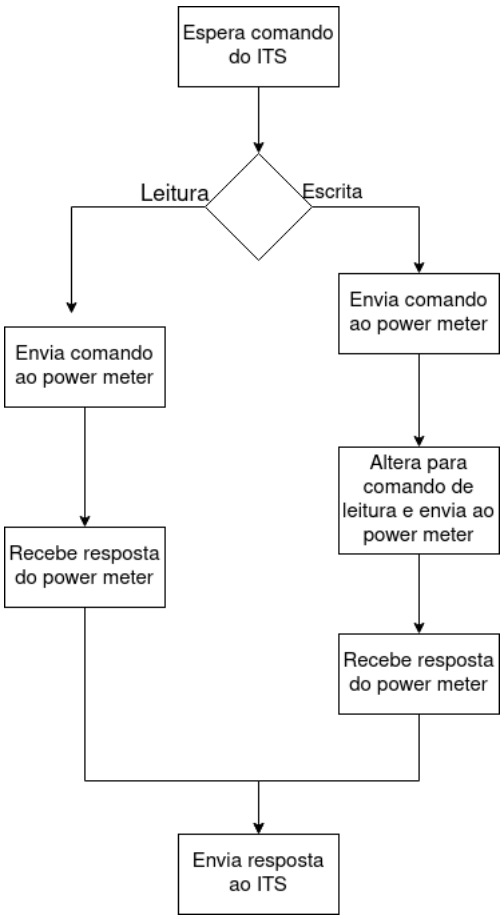

Figura 7 – Fluxo de operação do MCU intermediário da EM T103

Em um momento inicial do desenvolvimento, o fluxo de operação da EM T103 foi escrito de um modo simples e direto, apenas visando implementar a leitura e escrita em registradores do *power meter*. Ao verificar que elas estavam viáveis, começou-se o desenvolvimento do *driver* da extensão no ITS. Inicialmente foram escritas as rotinas base para leitura e escrita em registradores e depois disso as rotinas mais altas para realizar medições em todas as fases ou leituras divididas em dois registradores. Elas passaram a ser consistentes quando foi aplicada no dispositivo uma calibração baseada em outros produtos já desenvolvidos que usavam CIs da mesma linha.

Após algumas semanas de uso do protótipo o *power meter* foi trocado para o

modelo que seria de fato usado na placa. Este novo modelo possui algumas diferenças de configuração e leituras de taxas harmônicas em até 32 harmônicas, com possibilidade de expansão desse número através de algumas configurações. Neste momento foram necessárias algumas alterações de configuração e foram adicionadas as leituras de taxas harmônicas, que são um dos grandes destaques do produto.

Também neste *hardware* foi feito o contato inicial com o acelerômetro que seria usado através de uma placa separada para desenvolvimento, entretanto apenas o reconhecimento inicial dele foi feito. O *hardware* protótipo foi utilizado entre fevereiro e março de 2021.

3.1.5 Desenvolvimento da versão beta na primeira versão do *hardware*

Com a chegada da primeira versão da placa de extensão foi necessário alterar a pinagem do MCU pois o canal SPI usado mudou. Além disso, o código escrito anteriormente foi refatorado para permitir uma melhor flexibilidade de desenvolvimento. Foi adicionado o tratamento do LED da placa e tratamento para comandos especiais do ITS, como reiniciar a extensão e solicitar versão de *firmware*.

Uma versão beta do produto deveria estar disponível ao final de junho e por isso a partir daí os esforços se voltaram a validar a confiabilidade das medições ainda não verificadas, que eram as que dependiam da corrente elétrica. Até esse ponto as leituras validadas eram essencialmente tensão, frequência e temperatura do CI. Neste momento também verificou-se a existência de uma instabilidade no barramento I2C da extensão, que quando deixada operando com o ITS por muito tempo perdia a comunicação e derrubava o canal, fazendo com que o MCU do sensor *standalone* perdesse a comunicação com todos os CIs que utilizavam o canal I2C.

A queda do barramento I2C tomou algumas semanas do desenvolvimento para ser solucionada. A medida mais promissora para solucionar esse problema foi a adição de um período de *cooldown* entre leituras consecutivas do *power meter* em uma tentativa de subcarregar o barramento, mas ainda assim ao deixar o kit operacional durante a noite, na manhã seguinte a comunicação entre os dispositivos estava inoperante. Além dessa medida, foi adicionada uma tratativa para que o ITS reiniciasse via *hardware* a extensão caso houvesse queda do barramento. Ela também foi promissora mas não fez com que o problema desaparecesse. A questão foi finalmente solucionada quando, com a ajuda do engenheiro de *hardware* do projeto, o conjunto foi analisado sob um osciloscópio e verificou-se que a falha no barramento era de fato devido à características eletrônicas da placa, ficando evidente a necessidade de alteração dos resistores de *pullup* do barramento I2C. As medidas acima mencionadas foram mantidas, entretanto, e o problema não voltou a ocorrer.

Nesta fase também optou-se por desenvolver um protocolo de comunicação exclusivo para as medições da extensão. Por padrão, o ITS, assim como o *gateway*

IoT da Khomp, *parseia* suas leituras e envia-as à nuvem no formato SenML (IETF, 2018). Entretanto para a adição de todas as leituras do *power meter* a esse pacote, uma quantidade absurda de bytes seria adicionada a cada envio, e grande parte disso poderia ser evitada. Assim optou-se por utilizar um modelo personalizado para a geração do pacote com as leituras elétricas (as do acelerômetro foram integradas ao protocolo já utilizado pelo ITS).

Esse protocolo consiste na utilização de uma mensagem no formato JSON (*Java Script Object Notation*, (ECMA, 2017)) em que as chaves são *strings* identificando a medição e a unidade e os valores são um vetor (*array*) contendo os valores de cada fase. Embora a medição exclusivamente trifásica estivesse entre as premissas do projeto, optou-se por permitir que medições de uma ou duas fases fossem possíveis, entretanto elas deveriam ser feitas nas fases A ou A e B (no caso de uma ou duas fases, respectivamente). Com o desenvolvimento do gerador de pacotes personalizado, essa restrição de fases ficou consolidada, pois as fases são identificadas unicamente por suas posições no vetor, e caso haja um ou dois valores apenas, tratar-se-á da fase A ou das fases A e B, respectivamente. No caso das medições de taxas harmônicas foi usada uma matriz ao invés de um vetor, para aglutinar todas as harmônicas e as fases.

Além da liberação de um *firmware* beta para o final de junho, estava previsto o envio de dois kits para a empresa parceira, para validação do conceito e dimensionamento da *case* do produto final (o conjunto ITS + EM T103). Entretanto antes desse envio todas as medições da extensão deveriam ser validadas na Khomp, por isso um kit foi instalado no quadro geral da Khomp para verificação do comportamento e comparação com o medidor já instalado lá (também desenvolvido pelo setor de IoT). Esta instalação imediatamente revelou um erro de configuração do outro medidor instalado no quadro, pois os dispositivos estavam enviando valores diferentes para a mesma grandeza. Com um multímetro foi verificado que a leitura da EM T103 era a correta e posteriormente foi confirmado que o outro medidor (ITE 10Zi) estava configurado para um TC diferente do que estava sendo de fato usado, então esta configuração foi corrigida. Além disso essa instalação revelou algumas falhas que não haviam sido vistas antes, como erros no tratamento de sinalização de algumas leituras, e um fato interessante: as potências totais harmônicas nas três fases estavam negativas (e ainda estão na data de escrita desse PFC), mostrando que o problema ilustrado na seção 2.2 está ocorrendo com a energia consumida pela Khomp.

Houveram alguns atrasos na geração da versão beta e na instalação no quadro da Khomp e ao final do mês de junho o autor teve covid. Isso resultou que os kits foram enviados apenas em meados de julho.

3.1.6 Desenvolvimento da primeira *release candidate*

Na data de escrita deste projeto de fim de curso, não foi liberada ainda a primeira *release candidate* (RC0) do produto, mas o desenvolvimento de software que sucedeu o envio à empresa parceira foi e vem sendo executado tendo em vista o fechamento desta versão.

A primeira adição no conjunto foi o acelerômetro, que até o momento tinha sido deixado em segundo plano. A sua integração começou verificando a leitura de seu registrador *who am I* para confirmação da estabilidade do canal e depois disso foi desenvolvido o roteiro de *self test* do dispositivo, seguindo os passos fornecidos pela ST Microeletronics no *datasheet* do mesmo. Após essa verificação foi desenvolvida a medição de inclinação usando as leituras de aceleração X, Y e Z do dispositivo. Em um primeiro momento lê-se as acelerações na posição inicial (assumindo que o sensor esteja estático, para que a gravidade seja a única força agindo sobre ele) e a partir daí, para saber a inclinação em relação àquela posição, lê-se as acelerações em um novo momento e calcula-se o ângulo entre os vetores usando a fórmula

$$\theta_{v_1,v_2} = \arccos\left(\frac{< v_1, v_2 >}{||v_1||.||v_2||}\right)$$

em que $< v_1, v_2 >$ representa o produto interno entre os vetores, $||v||$ a norma do vetor e a função arccos é a inversa da função cosseno.

Além disso também foi desenvolvida a detecção de impactos com o acelerômetro, configurando-o para gerar uma interrupção caso detecte, em qualquer eixo, uma aceleração maior que um limiar configurado. Na escrita desse PFC a *feature* está praticamente completa, restando apenas a definição deste limiar, que no momento tem um valor provisório.

A etapa seguinte à integração do acelerômetro foi a adição das leituras de energia. Durante o desenvolvimento da versão beta foram desenvolvidas rotinas para ler os registradores de energia do *power meter*, mas esses registradores marcavam sempre zero, então concluiu-se que alguma configuração faltava ser feita. Após um período de análise, a causa da ausência de leitura foi encontrada em uma configuração que (por um motivo desconhecido) era *resetada* para seu valor inicial após sua escrita. Em seu valor inicial essa configuração habilitava o *checksum* do CI e como as rotinas estavam escritas considerando-o desabilitado, o *power meter* verificava a inconsistência e por causa disso bloqueava a função de *metering*, ou acumulação de energia. Essa questão foi solucionada pela adição de um *reset* de *software* no *power meter* antes das configurações iniciais. Isso assegurou que o valor escrito no registrador fosse mantido, desabilitando o *checksum* e hablilitando o *metering*.

No que diz respeito à geração de pacotes da extensão, foi criada uma geração para as leituras de energia separada das leituras instantâneas e as medições do acelerômetro foram integradas à geração padrão do ITS, no formato SenML. Além

disso foi criada uma mensagem própria para a detecção de impactos, separada do sensoriamento padrão do ITS, também no formato SenML.

A última grande adição ao medidor de energia (e talvez a mais importante) é a de um mecanismo de atualização de *firmware*, nesse caso um *bootloader* e um canal de comunicação com o ITS usando o protocolo XMODEM. No começo do projeto foi definido que embora o *firmware* da extensão fosse simples para evitar ter que atualizá-lo, a atualização deveria ser possível de ser feita, e foi reservado um canal de comunicação UART compratilhado entre os dispositivos para realizar essa transferência de arquivo. Assim sendo o ITS vai receber da nuvem um arquivo de *firmware* , verificar que se trata de um pacote para a extensão e enviar para ela. Isso exige alterações no *bootloader* do ITS (que não foi liberado ainda, então trata-se de um código que pode ser alterado) e criação de um *bootloader* para a extensão, de tal modo que aquele possa enviar um arquivo pelo protocolo XMODEM e este possa recebê-lo.

No momento de escrita deste PFC este *bootloader* está em desenvolvimento, estando por serem verificados a transmissão e recepção via XMODEM.

4 RESULTADOS

4.1 RESULTADOS OBTIDOS ATÉ AGORA

4.1.1 Modularidade e reaproveitamento de código

Antes de discorrer sobre os resultados de fato obtidos vale ressaltar que grande parte do código desenvolvido nesse projeto foi feito de forma modular, separando as chamadas de função de *hardware* em uma camada de abstração separada. Isso significa que bastante do código pode ser reaproveitado em outras plataformas caso haja necessidade e caso precise ser modificado tais modificações não serão excessivamente trabalhosas pois a lógica de operação está devidamente separada das chamadas à funções específicas da arquitetura utilizada (*e.g.* funções de comunicação I2C ou SPI, que podem variar entre diferentes microcontroladores).

Esta modularidade é particularmente importante em se tratando do *driver* da extensão, pois permite que futuramente outros dispositivos Khomp possam integrar operabilidade com essa extensão, e da comunicação via protocolo XMODEM, que poderá ser integrada em outros dispositivos e facilmente poder-se-há adicionar novos canais para uso do protocolo (no momento ele está implementado para uso via UART, mas novos canais podem ser adicionados, conforme demanda).

4.1.2 *Features* desenvolvidas

4.1.2.1 Leituras instantâneas

As principais leituras da EM T103 são feitas a cada ciclo de monitoramento do ITS (período configurável) e enviadas à nuvem. Elas são:

- Tensão RMS em cada fase;
- Corrente RMS em cada fase;
- Fator de potência em cada fase;
- Potência harmônica total em cada fase;
- Taxas harmônicas de tensão em cada fase;
- Taxas harmônicas de corrente em cada fase;

Estas grandezas foram selecionadas para que o integrador possa, ao receber os dados na nuvem, calcular e disponibilizar para o cliente final:

- Potência aparente (em cada fase) S:

$$S = V_{rms} \cdot I_{rms}$$

- Potências ativa P e reativa Q (em cada fase):

$$P = S \cdot Pf$$

$$Q = S \cdot \sqrt{1 - Pf^2}$$

- Potência ativa fundamental $P_{fundamental}$ (em cada fase):

$$P_{fundamental} = P - P_{harmonic}$$

- Distorção harmônica total (para tensão ou corrente, em cada fase):

$$THD_k = \sqrt{\sum_{m=2}^{n} k_m^2}$$

  em que k pode ser tensão (V) ou corrente (I) e m é o índice da harmônica. Note que k começa com o índice 2 pois $k = 0$ trata da componente CC da corrente ou tensão e $k = 1$ trata da componente na frequência fundamental, assim a primeira harmônica está em $k = 2$.

Estas leituras foram calibradas com base em um ensaio feito em laboratório usando o mesmo CI utilizado nessa extensão. Os dados obtidos nesse ensaio foram adaptados para se adequarem ao TC usado no projeto e ao *shunt* da placa.

4.1.2.2 Acumulação de energia

O *power meter* calcula internamente a energia consumida pela carga que o sistema alimenta e disponibiliza também essas leituras. Para essa aplicação são entregues à nuvem as energias:

- Ativa direta;
- Reativa direta;
- Ativa inversa;
- Reativa inversa;
- Aparente;
- Direta ativa harmônica;
- Inversa ativa harmônica;

4.1.2.3 Medição de inclinação e detecção de impactos

Conforme descrito acima a inclinação do poste é medida indiretamente, usando as diferenças entre as leituras do acelerômetro no momento inicial e no momento da medição. Já a detecção de impactos é feita diretamente pelo acelerômetro, e sua saída é remapeada para o ITS através de uma interrupção externa. Vale ressaltar que esse recurso é confiável apenas para situações em que a alimentação do conjunto não seja cortada devido ao impacto, pois o ITS não conta com o recurso de *last gasp*.

4.1.2.4 Geração de pacotes de medição

Conforme discorrido na seção 3.1.6, optou-se por desenvolver um protocolo de comunicação personalizado para os pacotes da extensão, para otimizar o uso de dados. Para esse fim foi desenvolvido um outro subprojeto (separado do *driver* e do *firmware* da extensão para implementar a geração dos pacotes de acordo com esse protocolo. Esse projeto foi criado com o intuito de ser utilizado como um submódulo *git*, bem como o *driver* da extensão, o que faz com que possa ser facilmente portado para outros projetos.

O pacote consiste, como citado acima em uma mensagem no formato *JSON* em que as chaves identificam o dispositivo, o momento da medição, cada grandeza medida e sua unidade e os valores são o número de série do dispositivo, a *timestamp* do momento da medição e *arrays* com as leituras de cada fase, respectivamente em cada caso. O gerador de pacotes consiste em funções que recebem como parâmetro uma estrutura de dados com as medições e retornam o pacote devidamente montado.

Identificador	Grandeza medida	Unidade	
v_rms-V	Tensão RMS	Volt (V)	Valor efetivo da tensão em cada fase
i_rms-A	Corrente RMS	Ampere (A)	Valor efetivo da corrente em cada fase
pf-	Fator de potência	Inexistente	Valor do fator de potência em cada fase
harm_pwr-W	Potência ativa harmônica	Watts	Potência total em frequências harmônicas em cada fase
v_hr-%	Taxas harmônicas de tensão	Percentual	Percentual de tensão em cada harmônica e em cada fase
i_hr-%	Taxas harmônicas de corrente	Percentual	Percentual de corrente em cada harmônica e em cada fase

Identificador	Grandeza medida	Unidade
fwd_a_en-kWh	Energia direta ativa	quilowatt-hora (kWh)
fwd_r_en-kVArh	Energia direta reativa	quilovolt-Ampere-reativo-hora (kVArh)
rvs_a_en-kWh	Energia inversa ativa	quilowatt-hora (kWh)
rvs_r_en-kVArh	Energia inversa reativa	quilovolt-Ampere-reativo-hora (kVArh)
apparent_en-kVAh	Energia aparente	quilovolt-Ampere-hora (kVAh)
fwd_a_h_en-kWh	Energia direta ativa harmônica	quilowatt-hora (kWh)
rvs_a_h_en-kWh	Energia inversa ativa harmônica	quilowatt-hora (kWh)

4.1.3 *Features* a serem finalizadas

4.1.3.1 Atualização de *firmware*

Esta funcionalidade divide-se em duas grandes etapas: desenvolvimento de um *bootloader* e desenvolvimento de comunicação XMODEM entre a extensão e o ITS.

No que diz respeito à primeira etapa, foi desenvolvida uma aplicação para rodar na inicialização do dispositivo e pular para a aplicação real. Com isso altera-se a região de memória em que a aplicação principal é gravada e coloca-se o *bootloader* no começo da memória *flash* do microcontrolador. Assim fica reservado o espaço para adicionar os mecanismos de recepção de um novo arquivo de *firmware* do ITS e validar sua integridade. A esses mecanismos corresponde a segunda etapa, que está em desenvolvimento no momento da escrita desse PFC. Até então foi desenvolvida a transferência de arquivos usando o protocolo XMODEM entre o ITS e a extensão, mas apenas foi verificada com arquivos de texto. Além disso a gravação interna de um novo *firmware* na memória do microcontrolador ainda está por ser desenvolvida.

4.1.3.2 Detecção de queda de energia

Esta *feature* corresponde ao uso da função de monitoramento do *power meter*. O canal para que as interrupções geradas por este em caso de perda de fase ou detecção de queda de tensão ou corrente foi desenvolvido no *hardware* pelo engenheiro responsável, ficando pendentes as tarefas de configurar os *thresholds* de tensão e corrente e tratar essas interrupções quando elas acontecerem. O autor já possui *know how* para ambas as etapas, assim prevê-se que elas não demandarão um grande esforço.

Além destas *features*, está também por ser implementada a adição da medição de distorção harmônica total (THDN) direta do *power meter*. Na seção 4.1.2.1 foi mostrado um método para calcular esse valor. Se calculado na nuvem, esse valor levará em conta apenas as 9 componentes harmônicas enviadas pelo dispositivo. Como o *power meter* consegue ler até 32 harmônicas e também disponibiliza essa grandeza,

o seu envio à *cloud* constituirá uma análise mais refinada do que a feita com apenas as componentes enviadas, embora possa-se dizer que a análise usando os dados atualmentes já é consideravelmente fina, pois a maior parte da energia nas frequências harmônicas está nessas primeiras.

4.2 ETAPAS FUTURAS

4.2.1 Integração com o ITS no *hardware* definitivo

Durante todo o desenvolvimento desse PFC o ITS também estava em desenvolvimento e ainda não foi liberado comercialmente. Entretanto em certo momento foi decidido que seriam feitas grandes alterações em seu projeto, principalmente devido à falta de disponibilidade de alguns componentes. Isso fez com que o desenvolvimento da EM-T103 acontecesse usando como referência para integração uma placa do modelo que não será mais utilizado.

Na parte de *software*, crê-se que a integração do *driver* da extensão será de pouca complicação, devido ao explicado na seção 4.1.1. Entretanto ainda é necessário validar a operação e a comunicação entre eles nos protocolos desenvolvidos no projeto: I2C e UART, este último para a atualização de *firmware*, além da recepção do ITS de uma interrupção gerada pela extensão, sinalizando a detecção de impacto do acelerômetro. No momento da escrita desse PFC o autor tem acesso a uma placa com o mesmo microcontrolador utilizdo no ITS novo para validar tais aspectos operacionais, mas ainda não começou tal trabalho.

4.2.2 Integração com o *endpoint LoRa* da Khomp

Embora este projeto seja feito com o intuito principal de operar com o ITS, trata-se de um módulo de extensão, o que significa que ele pode se acoplar a *endpoints* desenvolvidos pelo setor. Assim sendo caso surja demanda o *endpoint LoRa* da Khomp pode ganhar integração com essa extensão. Essa etapa, entretanto, é apenas um projeto no momento e não há ainda data previsão de quando isso será feito. De qualquer modo, devido aos motivos supracitados, também crê-se que seja de fácil implementação.

4.3 HABILIDADES DESENVOLVIDAS E FERRAMENTAS DOMINADAS

Este trabalho foi integralmente desenvolvido em linguagem C em ambiente Linux e utilizando o *git* como ferramenta de versionamento de código. Durante o desenvolvimento desse PFC (e do estágio que o antecedeu) estas foram claramente as principais ferramentas nas quais o autor desenvolveu sua maestria. No decorrer dos estágios o autor esteve a todo momento explorando e se familiarizando com o ambiente Linux:

navegação pelo terminal, comandos básicos e intermediários, estrutura de diretórios básica e comandos de compilação do gcc (*gnu compiler collection*).

O uso do *git* também permeou o período que o autor esteve na empresa e obteve-se familiaridade com, além da ferramenta, as boas práticas de uso e os *fluxos* de uso, como *git lab flow* e *git hub flow*. Além disso o autor aprendeu, durante esse período, os conecitos de *branches*, e de *head* de um repositório *git*, tendo tido algumas experiências em navegar por *commits* de um projeto e resolver incompatibilidades. Mais adiante foi também introduzido o submódulo *git* e seu uso fez parte do projeto. Fortemente relacionado à integração de submódulos no projeto esteve o domínio da ferramenta *Makefile*, para criação de scripts para facilitar a compilação de diversos arquivos. No que diz respeito a essa ferramenta, o autor se familiarizou tanto que chegou a ministrar aos colegas do setor um pequeno treinamento interno sobre o uso dela em conjunto com o submódulo com submódulos *git* para integrar módulos em um projeto.

No decorrer do projeto o autor também trabalhou *soft skills*, como trabalho em equipe e boas práticas de desenvolvimento, para facilitar a compreensão do código por outros desenvolvedores. Nesse âmbito, o autor exercitou a capacidade de descrever suscintamente, no uso do *git*, como um *commit* altera o código e porque tal alteração é feita. Mais além, na criação de *merge requests* (processo na ferramenta *git lab* para unificar *branches*), o autor aprendeu a descrever mais detalhadamente os aspectos acima. Em todas essas situações, bem como no uso do *Redmine* o autor aprendeu a documentar o desenvolvimento para manter acessível informações chave a futuros desenvolvedores. Com efeito, próximo à escrita desse PFC, esse hábito de documentar se mostrou formidável, em um episódio onde o problema que um colega vinha enfrentando no desenvolvimento de uma *feature* já havia sido mapeado pelo autor quando este estudou a API utilizada vários meses antes. As principais conclusões daquele estudo tinham ficado documentadas e ao procurar na tarefa onde haviam ficado essas anotações, o colega verificou que lá estava a explicação do problema que ele teve e o que era necessário fazer para solucionar.

Outros *softwares* dignos de menção são: o STM32CubeMX, fornecido pela ST Microeletronics para definição de pinagem e geração de código base para os microcontroladores dessa fabricante, o *picocom* para verificação de comunicação UART nas portas USB do computador (usando um conversor) e os softwares *mosquitto* e MQTT box, para operações de publicação e inscrição em tópicos de comunicação no protocolo MQTT (largamente predominante nos dispositivos integrados à nuvem no setor de IoT da Khomp).

5 CONCLUSÃO

As *features* de medição elétrica, medição de energia, inclinação e detecção de impactos foram devidamente implementadas e inicialmente verificadas no *kit* instalado no quadro da Khomp. O fluxo de operação da extensão com o *hardware* antigo do ITS também foi colocado à prova durante esse período e se mostrou adequado: o sistema não entrou em *deadlock* ou *livelock* durante o período e chegou a ficar 5 dias sem precisar reinicar. Esse período apenas foi interrompido para instalação de atualizações no *kit*, que exigiam um *reboot* do ITS. Com isso tem-se evidências para dizer que o núcleo do que era esperado está bem implementado e em boas condições.

A atualização de *firmware* tem se mostrado a dificuldade mais resiliente até o momento, estando por ser finalizada ainda na data de escrita deste PFC. Ainda assim o desenvolvimento desse atributo já acrescentou bastante conhecimento ao autor, como métodos para superar as peculiaridades de uma arquitetura córtex-M0 no que diz respeito a um *bootloader* e um entendimento inicial do *linker script* e do arquivo de inicialização do microcontrolador.

De uma perspectiva mais ampla, o desenvolvimento desse projeto de fim de curso acrescentou ao autor uma enorme gama de conhecimentos, como os relatados na seção 4.3, e serviu para que muito do que ele aprendeu no curso fosse colocado em prática, desde conhecimentos técnicos, como a própria programação, até habilidades sutis, como leitura e compreensão de *datasheets* e definição de arquiteturas de *software*. Acima disso, no decorrer do desenvolvimento o autor aprofundou muito seu entendimento da linguagem C e principalmente do seu uso em microcontroladores. Em seu ponto de vista, a execução desse PFC supriu tudo que faltava em um estudante para se tornar de fato um engenheiro.

REFERÊNCIAS

ALLIANCE, LoRa. **LoRa Alliance**. 2015. Disponível em: https://lora-alliance.org/. Acesso em: 17 set. 2021.

GRADY, Prof. Mack. **Understanding Power System Harmonics**. [*S.l.*], 2012. Disponível em: https://web.ecs.baylor.edu/faculty/grady/understanding_power_system_harmonics_grady_april_2012.pdf. Acesso em: 17 set. 2021.

INSTITUTE, Gartner. **Internet of Things: Unlocking True Digital Business Potential**. 2021. Disponível em: https://www.gartner.com/en/information-technology/insights/internet-of-things. Acesso em: 17 set. 2021.

JENNINGS, C.; CISCO, Z. Shelby; ARM; ARKKO, J.; KERANEN, A.; ERICSSON; BORMANN, C. **Sensor Measurement Lists (SenML)**. [*S.l.*], ago. 2018. Disponível em: https://datatracker.ietf.org/doc/html/rfc8428. Acesso em: 17 set. 2021.

THE JSON Data Interchange Syntax. [*S.l.*], dez. 2017. Disponível em: https://www.ecma-international.org/wp-content/uploads/ECMA-404_2nd_edition_december_2017.pdf. Acesso em: 17 set. 2021.