UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
PROJETO DE GRADUAÇÃO

LUCAS MARTINS RIMOLO

TRANSMISSÃO DE DADOS CLÍNICOS MULTIPARAMÉTRICOS POR MEIO DE UM SISTEMA DE COMUNICAÇÃO HÍBRIDO PLC-VLC

VITÓRIA
2021

LUCAS MARTINS RIMOLO

TRANSMISSÃO DE DADOS CLÍNICOS MULTIPARAMÉTRICOS POR MEIO DE UM SISTEMA DE COMUNICAÇÃO HÍBRIDO PLC-VLC

Parte manuscrita do Projeto de Graduação do aluno **Lucas Martins Rimolo**, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Orientador: Prof. Dr. Jair Adriano Lima Silva
Coorientador: Eng. Renan Costa Lazaro

VITÓRIA
2021

LUCAS MARTINS RIMOLO

TRANSMISSÃO DE DADOS CLÍNICOS MULTIPARAMÉTRICOS POR MEIO DE UM SISTEMA DE COMUNICAÇÃO HÍBRIDO PLC-VLC

Parte manuscrita do Projeto de Graduação do aluno **Lucas Martins Rimolo**, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Aprovada em 01, de Julho de 2021.

COMISSÃO EXAMINADORA:

_____

Prof. Dr. Jair Adriano Lima Silva
Universidade Federal do Espírito Santo
Orientador

_____

Eng. Renan Costa Lazaro
Universidade Federal do Espírito Santo
Coorientador

_____

Prof. Dr. Helder Roberto de Oliveira Rocha
Universidade Federal do Espírito Santo
Examinador

_____

Me. Higor Araújo Fim Camporez
Universidade Federal do Espírito Santo
Examinador

Ao meu avô Dudu.

AGRADECIMENTOS

Agradeço inicialmente aos meus orientadores, Prof. Dr. Jair Adriano Lima Silva e Eng. Renan Costa Lazaro, por todo o apoio e confiança em mim investidos.

A todos os integrantes do Laboratório de Telecomunicações da UFES (LabTel) que também contribuíram e, de alguma forma, fazem parte deste trabalho.

À FAPES, pelo financiamento do projeto "Aplicação da Tecnologia de Comunicação por Luz Visível em Monitorização de Recém Nascidos de Alto Risco", parte do Programa Pesquisa para o SUS: Gestão Compartilhada em Saúde (PPSUS).

Aos servidores e professores da UFES, além de quem mais se importa com a educação no país.

Ao LCEE, à *ITMO University* e ao Planetário de Vitória pelas portas abertas e experiências acumuladas.

Ao Núcleo de Línguas, aos projetos de extensão, aos trabalhadores das cantinas, limpeza e serviços em geral pelo acolhimento.

Aos meus amigos, de dentro e fora do curso, que sempre estão lá por você nos dias em que precisa (e também em todos os demais). Por vocês aprendi a ser uma pessoa melhor.

E, finalmente, à toda minha família. Em especial aos meus pais, Margareth e Valter, responsáveis não apenas por minha educação formal mas também por me apoiarem incondicionalmente, as suas maneiras, em cada um dos dias dessa jornada. Agradeço a vocês também pela minha vida e por todos os seus cuidados.

Ao meu avô Dudu, ou Augusto, vai não somente um agradecimento por estar comigo nesta luta até o final, mas também por ser em minha vida o exemplo de carinho e coragem que foi, é e sempre será.

À UFES, deixo uma mensagem de "muito obrigado". Mas, também, outra com um "até breve"!

RESUMO

Este trabalho buscou apresentar uma solução para os desafios de se transmitir e receber dados em ambientes hospitalares classificados. Para isso desenvolveu-se um sistema híbrido a partir das tecnologias de comunicação por meio da rede elétrica (PLC, do inglês *power-line comunication*) e da luz visível (VLC, do inglês *visible light communication*). O principal objetivo do projeto consistiu na implementação física do sistema PLC-VLC, de fato, sob o qual se é capaz de transmitir e receber informações de sinais vitais de pacientes internados, armazenadas na forma de arquivos de extensão HL7, padrão amplamente estabelecido na área médica. Para representação da fonte geradora dos sinais vitais, utilizou-se um computador com função de enviar, em tempo real, arquivos-base HL7 previamente coletados. Também fizeram parte do sistema um conjunto de dois módulos de comunicação PLC, um aparelho *smartphone*, uma lâmpada de LED, um módulo relé de estado sólido, além de microcomputadores do tipo *Raspberry Pi* (RPi). A modulação utilizada foi a de chaveamento liga-desliga (OOK, do inglês *on-off keying*) e o sinal foi codificado pela técnica conhecida por não retorno ao zero (NRZ, do inglês *non-return to zero*), ambos no RPi. O projeto teve uma metodologia de pesquisa experimental e aplicada, com a concepção das ideias a partir de exemplos da literatura e a execução ocorrendo por meio de um desenvolvimento prático. Os testes foram realizados com o auxílio de programação e eletrônica, sendo mensurados quantitativamente e avaliados em caráter explicativo. Os resultados obtidos podem ser listados como a verificação de uma comunicação *back-to-back*, entre dispositivos RPi, em até 200 kbps. A transmissão e a recepção com PLC ocorreu em camada de aplicação com taxas iguais ou acima de 90 Mbps. Além disto, caracterizou-se o trecho VLC, a partir da teoria fotométrica, gerando-se uma curva das iluminâncias capturadas pelo sensor de luminosidade ambiente do celular em função das distâncias observadas. Bem como recebeu-se, confiavelmente, dados enviados a 1 bps em meio ao sinal luminoso transmitido a até 1,8 m de distância de um aparelho *smartphone* na recepção. Nesse sentido, o projeto foi bem-sucedido em seu objetivo de oferecer uma prova de conceito para o sistema híbrido proposto, além de contribuir com a criação de protocolos baseados em arquivos HL7, vinculado-os às tecnologias VLC e PLC, neste trabalho estudadas.

Palavras-chave: PLC. VLC. HL7. LED. *Raspberry pi*. *Smartphone*. OOK. NRZ.

ABSTRACT

This work aims to present a solution to the challenges of transmitting and receiving data in classified hospital environments. In this sense, a hybrid system was developed based on communication technologies through powerline (PLC) and visible light (VLC) communication. The main goal of the project was the physical implementation of a PLC-VLC system fully capable to transmit and receive vital signs information of hospitalized patients, stored in the form of HL7 files – a widely established standard in the medical area. To represent the source of the information, a computer was used to send in real time previously collected HL7 files. A set of two PLC communication modules, a smartphone device, a LED lamp, a solid state relay module and a pair of Raspberry Pi (RPi) microcomputers also are part of the setup. The modulation applied to the signal was the technique named on-off keying or OOK. The line code signal was the non-return to zero (NRZ) technique with both processes happening in the RPi microcomputer. The project had an experimental and applied research methodology with the conception of the ideas based on examples from the literature in addition to the execution through a practical experimentation. The tests were performed with both programming and electronics and measured quantitatively along to explanatory evaluations. The results obtained can be listed as the verification of a back-to-back communication between the RPi devices with speed up to 200 kbps. The transmission and reception with PLC took place at the application layer at rates equal to or above 90 Mbps. Furthermore, the VLC section of the setup was characterized with the help of the photometric theory generating a curve of illuminances captured by the cell's ambient luminance sensor as a function of the observed distances. Finally, a reliable data stream at 1 bps and 1.8 m away has been transmitted and received through the visible light. In this sense, the project was successful in its objective to provide a proof of concept for the proposed hybrid system in addition to the contribution with creating protocols based on HL7 files and linking them to the VLC and PLC technologies studied in this text.

Keywords: PLC. VLC. HL7. LED. Raspberry Pi. Smartphone. OOK. NRZ.

LISTA DE FIGURAS

Figura 1 – Respostas no tempo e em frequência de um sinal em banda base 23
Figura 2 – Exemplificação dos códigos NRZ e RZ 25
Figura 3 – Sinal modulado com OOK 25
Figura 4 – Diagrama de blocos de uma comunicação digital em banda base 26
Figura 5 – Exemplos de diferentes formas de modulação OOK 27
Figura 6 – Exemplo de aplicação residencial de um sistema PLC 29
Figura 7 – Espectro eletromagnético e a faixa de luz visível 31
Figura 8 – Diagrama de blocos generalizado para a comunicação VLC 32
Figura 9 – Ilustração do sistema VLC implementado em um ambiente hospitalar 32
Figura 10 – Exemplo de uma mensagem típica com padrão HL7 33
Figura 11 – Mensagem típica do protocolo SFTP 35
Figura 12 – Princípio de funcionamento do protocolo SFTP 35
Figura 13 – Diferentes possibilidades de representações com a comunicação UART 36
Figura 14 – MPDU típico do protocolo *HomePlug* AV 37
Figura 15 – PPDU típico do protocolo *HomePlug* AV 37
Figura 16 – Representação conceitual do *setup* 1 (B2B) 38
Figura 17 – Medidor multiparamétrico de sinais vitais 39
Figura 18 – Microcomputador *Raspberry Pi* 41
Figura 19 – Mapeamento dos pinos GPIO em uma RPi3 41
Figura 20 – Representação conceitual do *setup* 2 (PLC-B2B) 42
Figura 21 – Conjunto extensor de alcance *powerline* utilizados no projeto 43
Figura 22 – Representação conceitual do *setup* 3 (PLC-VLC) 44
Figura 23 – Dispositivos LED emitindo luz 45
Figura 24 – Princípio de funcionamento de um LED 46
Figura 25 – Módulo relé de estado sólido 47
Figura 26 – *Smartphone* 49
Figura 27 – Sensor de luminosidade ambiente 50
Figura 28 – Taxa de amostragem padrão do ALS 50
Figura 29 – Estrutura de uma mensagem com o protocolo LabTel-I 54
Figura 30 – Estrutura de uma mensagem com o protocolo LabTel-II 55
Figura 31 – Estrutura de uma mensagem com o protocolo LabTel-III 56
Figura 32 – Leitura para o RPi.GPIO 59

Figura 33 – Princípio de funcionamento da leitura para o *Pigpio* 61
Figura 34 – Princípio de funcionamento da leitura para o Aplicativo 1 (possibilidade 1) ... 64
Figura 35 – Princípio de funcionamento da leitura para o Aplicativo 2 (possibilidade 2) .. 64
Figura 36 – Captura de tela do Aplicativo 2 65
Figura 37 – Configuração 1 (B2B) 66
Figura 38 – Conexão *back-to-back* entre os RPi 67
Figura 39 – Pacote de testes pseudoaleatório 68
Figura 40 – Forma de onda recebida com a biblioteca *Serial* (UART) 69
Figura 41 – Mensagem LabTel-I com a biblioteca *Serial* (UART) 70
Figura 42 – Atrasos percentuais testados 70
Figura 43 – Taxas escolhidas para o primeiro teste com a biblioteca RPi.GPIO 71
Figura 44 – Taxas escolhidas para o segundo teste com a biblioteca RPi.GPIO 73
Figura 45 – Forma de onda recebida com a biblioteca RPi.GPIO 74
Figura 46 – Mensagem recebida com a biblioteca RPi.GPIO 74
Figura 47 – Forma de onda recebida com a biblioteca *Pigpio* 75
Figura 48 – Taxas escolhidas para o primeiro teste com *Pigpio* 77
Figura 49 – Taxas escolhidas para o segundo teste com a biblioteca *Pigpio* 78
Figura 50 – Pacotes de testes pseudoaleatórios, alternados e contínuos 79
Figura 51 – Mensagem recebida com a biblioteca RPi.GPIO 82
Figura 52 – Configuração 2 (PLC-B2B) 83
Figura 53 – Ilustração de uma das medições da taxa de *powerline* (aplicativo tpPLC) 84
Figura 54 – Configuração 3 (PLC-VLC) 87
Figura 55 – Imagem em tela do Aplicativo 2 (menor amostragem) 88
Figura 56 – Imagem em tela do Aplicativo 2 (maior amostragem) 89
Figura 57 – Pacotes pseudoaleatórios a serem recebidos pelo aparelho *smartphone* 90
Figura 58 – Mensagem LabTel-III (App 1) 93
Figura 59 – Fluxogramas dos programas principais do RPi 100
Figura 60 – Fluxogramas dos aplicativos *android* 101
Figura 61 – Comparação dos sinais transmitidos e recebidos 102
Figura 62 – Captura de tela com um pacote do PLC 103
Figura 63 – Planta elétrica do ambiente testado 104

LISTA DE GRÁFICOS

Gráfico 1 – Curva de erros por atrasos 71
Gráfico 2 – Curva de erros entre 50 e 100 bps 72
Gráfico 3 – Curva de BER entre 50 e 100 bps 72
Gráfico 4 – Curva de erros entre 50 e 75 bps 73
Gráfico 5 – Curva de BER entre 50 e 75 bps 73
Gráfico 6 – Relação dos tempos medidos (10 e 200 kbps) 76
Gráfico 7 – Relação das taxas medidas entre 10 e 200 kbps 76
Gráfico 8 – Curvas de erros entre 10 e 200 kbps 77
Gráfico 9 – Curvas de BER entre 10 e 200 kbps 77
Gráfico 10 – Relação dos tempos (50 e 100 kbps) 79
Gráfico 11 – Relação das taxas medidas (50 e 100 kbps) 80
Gráfico 12 – Curvas de erros entre 50 e 100 kbps 80
Gráfico 13 – Curvas de BER entre 50 e 100 kbps 81
Gráfico 14 – Curvas de BER entre 50 e 100 kbps (ampliada) 81
Gráfico 15 – Curva da taxa de *powerline* em função das distâncias medidas 85
Gráfico 16 – Curvas de iluminâncias de referência e observadas 87
Gráfico 17 – Curvas de erros entre 10 e 50 cm 91
Gráfico 18 – Curvas de BER entre 10 e 50 cm 91
Gráfico 19 – Sinal luminoso modulado 92
Gráfico 20 – Diagrama de olho 93

LISTA DE QUADROS

Quadro 1 – Especificações técnicas da lâmpada utilizada..... 46
Quadro 2 – Faixas de operação do SSR e do RPi 48
Quadro 3 – Informações das tomadas, circuitos e novas medições da comunicação PLC .. 84

LISTA DE ABREVIATURAS E SIGLAS

ALS	*Ambient Light Sensor*
App	Aplicativo Móvel
ASCII	*American Standard Code for Information Interchange*
B2B	*Back-to-Back Connection*
BER	*Bit Error Ratio*
CC	Corrente Contínua
CPU	*Central Process Unit*
CSV	*Comma-Separated Values*
DMA	*Direct Memory Access*
EMI	*Eletromagnetic Interference*
GPIO	*General-Purpose Input/Output*
HL7	*Health Level 7*
HUCAM	Hospital Universitário Cassiano Antonio Moraes
I²C	*Inter-Integrated Circuit*
ICT	*Information and Communication Technologies*
IEEE	*Institute of Electrical and Electronics Engineers*
IoT	*Internet of Things*
ITS	*Institute for Telecommunication Sciences*
LabTel	Laboratório de Telecomunicações
LAN	*Local Access Network*
LED	*Light-Emitting Diode*
NRZ	*Non-Return-to-Zero*
OOK	*On-Off Keying*
OMS	Organização Mundial da Saúde
OWC	*Optical Wireless Communication*
PCM	*Pulse-Code Modulation*
PLC	*Power-Line Communication*
PPSUS	Programa de Pesquisa para o SUS
RF	Rádio-Frequência
RPi	*Raspberry Pi*
SFTP	*SSH File Transfer Protocol*
SO	Sistema Operacional

SSR	*Solid State Relay*
SUS	Sistema Único de Saúde
TIA	*Telecommunication Industry Association*
UART	*Universal Asynchronous Receiver/Transmitter*
UFES	Universidade Federal do Espírito Santo
VLC	*Visible Light Communication*
Wi-Fi	*Wireless Fidelity*
XML	*Extensible Markup Language*

LISTA DE SÍMBOLOS

BER	Taxa de bits errados
N_e	Número de bits errados recebidos
N_b	Número total de bits recebidos
Th	*Threshold* (lux)
s	Sensibilidade da recepção
E	Iluminância (lux)

SUMÁRIO

1 INTRODUÇÃO 17
1.1 Apresentação 17
1.2 Justificativa 19
1.3 Objetivos 21
1.3.1 Objetivo Geral 21
1.3.2 Objetivos Específicos 21
1.4 Organização do Trabalho 22

2 EMBASAMENTO TEÓRICO 23
2.1 Fundamentos de Comunicação Digital 23
2.1.1 Codificação NRZ 24
2.1.2 Modulação OOK 25
2.1.3 Taxa de Erros de Bit (BER) 27
2.2 Comunicação em Rede Elétrica (PLC) 27
2.3 Comunicação por Luz Visível (VLC) 30
2.4 Padrões e Protocolos 33
2.4. 1 HL7 33
2.4.2 SFTP 34
2.4.3 UART 35
2.4.4 *HomePlug* AV 36

3 SISTEMAS HÍBRIDOS DE COMUNICAÇÃO IMPLEMENTADOS 38
3.1 Configuração 1 (B2B) 38
3.1.1 *Notebook* como Monitor Multiparamétrico 39
3.1.2 Par de Microcomputadores *Raspberry Pi* 40
3.2 Configuração 2 (PLC-B2B) 42
3.2.1 Extensores de Alcance *Powerline* 42
3.3 Configuração 3 (PLC-VLC) 44
3.3.1 Lâmpada de LED 45
3.3.2 Relé de Estado Sólido (SSR) 47
3.3.3 *Smartphone* 49

4 PREPARAÇÃO DOS CÓDIGOS UTILIZADOS NOS SISTEMAS 51

4.1 Arquivos HL7 51
4.1.1 Geração e Envio 51
4.1.2 Leitura e Atualização 52
4.1.3 Escrita e Armazenamento 52
4.1.4 Fluxograma 53
4.2 Protocolos Sugeridos para as Aplicações 53
4.2.1 LabTel-I 54
4.2.2 LabTel-II 55
4.2.3 LabTel-III 56
4.3 Bibliotecas 57
4.3.1 *Serial* (UART) 57
4.3.2 RPi.GPIO 58
4.3.3 *Pigpio* 60
4.4 Aplicativos *Android* 61
4.4.1 Aplicativo 1 62
4.4.2 Aplicativo 2 65

5 RESULTADOS E DISCUSSÕES 66
5.1 Configuração 1 (B2B) 66
5.1.1 Serial UART 67
5.1.2 RPI.GPIO 70
5.1.3 Pigpio 75
5.2 Configuração 2 (PLC-B2B) 83
5.3 Configuração 3 (PLC-VLC) 86

6 CONCLUSÕES E TRABALHOS FUTUROS 94

REFERÊNCIAS BIBLIOGRÁFICAS 96

APÊNDICE A – FLUXOGRAMAS DOS PROGRAMAS PRINCIPAIS 100

APÊNDICE B – FLUXOGRAMAS DOS APLICATIVOS 101

APÊNDICE C – COMPARAÇÃO DOS SINAIS (TX) E (RX) 102

APÊNDICE D – CAPTURA DE TELA COM O PACOTE DO PLC 103

APÊNDICE E – PLANTA ELÉTRICA DO AMBIENTE TESTADO 104

1 INTRODUÇÃO

1.1 Apresentação

Ao exercer as práticas da enfermagem ou da medicina, não é exagero afirmar que os profissionais responsáveis por tais atividades devem continuadamente verificar o quadro de saúde do paciente. Para tanto, costumam utilizar equipamentos eletrônicos que os ajudem no processo de monitoramento dos sinais vitais do enfermo. Processo, este, no qual falhas podem significar a diferença entre a vida e a morte. Além disso, para evitar erros de diagnóstico, estes profissionais também necessitam de algum auxílio nas constantes tomadas de decisão, pelas quais são naturalmente responsabilizados. Nesse sentido, os processos de coleta e armazenagem dos dados monitorados tornam-se indispensáveis e, portanto, sistemas de comunicação confiáveis, capazes de transmitir e receber tal informação, se fazem também cada vez mais necessários (SHORTLIFFE et al., 2001).

Vislumbrando tal cenário, pode-se imaginar que soluções já bem estabelecidas como a expoente internet das coisas (IoT, do inglês *Internet of Things*), operando à base de Wi-Fi, poderiam muito bem acomodar esta necessidade observada. Entretanto, existem alguns problemas ainda não tão tratados quando se adentra um contexto de telecomunicação em ambientes classificados, como é o caso de instalações hospitalares. Problemas como, por exemplo, a interferência eletromagnética (EMI, do inglês *eletromagnetic interference*) entre equipamentos eletrônicos (SILBERBERG, 1993).

Máquinas ou instrumentos eletroeletrônicos que se localizem muito próximos uns aos outros, podem facilmente danificar-se mutuamente em caso de ambos não se encontrarem devidamente isolados. Equipamentos de radiologia, lâmpadas fluorescentes, alarmes de incêndio e até mesmo computadores são exemplos de fontes de radiofrequência em ambientes hospitalares. É importante pontuar que, mesmo para casos mais leves de interferência, as consequências já podem ser catastróficas. Um mal funcionamento, ainda que momentâneo, de um equipamento mantenedor de vida, por exemplo, pode resultar diretamente na morte do paciente internado (SILBERBERG, 1993).

Além disso, ao se utilizar um ponto de acesso de rede cuja irradiação se dá por meio de ondas em radiofrequência, como o 4G ou novamente o Wi-Fi, na prática se está adicionando uma fonte de radiação extra ao ambiente, o que é geralmente indesejável. Foi observado por Lapinsky (2006) que frequências na faixa entre 10 kHz e 1 GHz são as mais problemáticas quando se trata de EMI. No caso do 4G, as transmissões ocorrem a taxas acima de 2 Mbps, enquanto que o Wi-Fi utiliza as frequências de 2,4 e 5 GHz. Além disso, este último opera com potências mais baixas e sendo assim é menos provável de causar o problema destacado. Entretanto, ainda é uma potencial ameaça a distâncias menores que um metro e com exposição prolongada. Nesse sentido, se faz necessário investigar por alternativas que sejam garantidamente menos nocivas tanto para o ambiente classificado em questão, quanto para os seres humanos em geral (LAPINSKY, 2006).

O trabalho realizado nesta monografia apresenta como problema de pesquisa a verificação da possibilidade de se desenvolver um protótipo funcional de um sistema de telecomunicação, utilizando as vantagens das tecnologias de comunicação por meio da rede elétrica (PLC, do inglês *power-line communication*) e da luz visível (VLC, do inglês *visible light communication*). Assim, espera-se oferecer uma possível solução para o problema de interferência eletromagnética entre equipamentos que operam em radiofrequência e que são regularmente encontrados em ambientes hospitalares.

A configuração física do sistema (*setup*), basicamente foi dividida em três partes, de modo a separar as tecnologias umas das outras a fim de uma compreensão mais clara acerca de cada etapa específica do projeto. Na configuração final, um sistema híbrido de comunicação PLC-VLC foi apresentado, demonstrando a aplicabilidade da solução dentro de uma realidade financeira adequada à proposta. Ou seja, utilizou-se equipamentos de baixo custo, como microcomputadores *Raspberry Pi*, extensores de alcance *Powerline* e lâmpadas de LED, representando este caráter mais experimental, ou de pesquisa, do projeto.

Quanto aos procedimentos metodológicos, adotou-se a técnica de pesquisa experimental, com natureza de aplicação prática, passando uma ou mais vezes por etapas como concepção da ideia, simulação ou testes, desenvolvimento e avaliação (ou reavaliação) dos resultados obtidos. Utilizou-se sempre de hipóteses pautadas, ou na literatura, ou no contexto tecnológico atual, dando-se preferência sempre que possível a informações dos próprios fabricantes ou

desenvolvedores em questão. Por fim, buscou-se uma avaliação quantitativa dos resultados, além de uma análise de pesquisa em caráter explicativo, observando-se sempre o preceito básico de causa e efeito.

1.2 Justificativa

De sua divulgação no ano de 2000 para os dias atuais, os números da tecnologia VLC só cresceram, como observado por Pathak e outros (2015). Nesse sentido, o presente trabalho buscou sua utilização não só pela relevância da tecnologia no momento do estudo mas, principalmente, por sua capacidade de contornar os obstáculos inerentes à comunicação em radiofrequência (RF), expostos na seção 1.1. Além disso, sabe-se que a luz visível encontra-se distante do espectro RF, o que a torna uma candidata em potencial para solucionar os problemas aqui tratados. Como é explicado em Tanaka, Haruyama e Nakagawa (2000), a luz ainda traz vantagens importantes como não possuir espectro regulamentado, sendo livre, além de não adicionar radiação ou mesmo infraestrutura extras ao meio, visto que sua utilização já se faz necessária em ambientes *indoor* do mundo moderno.

Existem, porém, também algumas desvantagens em sua utilização, naturalmente. Sendo a principal delas o seu relativo baixo alcance, quando comparada a diferentes sistemas de comunicação. Se por um lado se agrega segurança à transmissão tendo a luz sendo bloqueada por paredes, também se perde alcance formando-se as chamadas “ilhas de informação” (LIU et al., 2020). É importante ressaltar, entretanto, que essa deficiência em especial abriu portas, justamente, para a utilização da outra tecnologia proposta pelo projeto, a PLC. Dessa forma, acredita-se que a solução híbrida implementada de certa forma complementa as desvantagens individuais de uma e de outra.

A inclusão da tecnologia PLC no projeto tem como objetivo principal proporcionar o aumento do alcance do sinal transmitido, para níveis em que a implementação VLC não é capaz de oferecer. Trazendo ao contexto clínico proposto pelo trabalho, cumprir maiores distâncias poderia significar, por exemplo, levar o sinal elétrico modulado pela informação dos dados vitais de diversos pacientes (de um determinado quarto), diretamente à sala do médico que os monitora. Soma-se a isto a praticidade e o baixo custo de instalação, como já visto, faz com que a tecnologia seja tratada como uma boa solução para o problema.

De fato, Jani, Garg e Gupta (2019) demonstram que a integração da tecnologia PLC à VLC, formando uma espécie de sistema híbrido entre ambos, como proposto no presente projeto, já vem sendo, desde 2003, regularmente difundida na literatura. Não apenas por oferecer uma solução ao problema da falta de alcance, a PLC ainda é sugerida para prover a energia de alimentação necessária para os equipamentos de uma instalação em VLC. Ou seja, a implementação do sistema híbrido justifica-se cada vez mais pelo fato das tecnologias virem se mostrando, de fato, complementares. Nesse sentido o PLC se faz de grande importância para o projeto.

Como era de se esperar, a tecnologia PLC também tem suas desvantagens. Mas ao colocar todos os pontos na balança, acredita-se que esta implementação é preferível à opção sem fio representada pelo Wi-Fi, encontrada atualmente como solução padrão em clínicas e hospitais. Isto se deve, basicamente, pelo motivo do sinal encontrar um maior confinamento dentro de um fio do que se estivesse em um meio livre como o ar, além de apresentar menor atenuação ao longo do canal. Ainda, no próprio artigo onde Rhee, Rhee e Park (2008) trazem o assunto à tona, eles acabam por concluir que os níveis de interferência observados foram pequenos e abaixo dos limites legais do país em que o experimento foi testado (Coréia do Sul). Sendo assim, ao menos em um primeiro momento, o sistema PLC segue sendo a melhor opção para oferecer maiores distâncias à solução final, em VLC, desejada, sem que para isso comprometa a condição inicial de uma comunicação mais “limpa”.

Este trabalho foi idealizado a partir de uma proposta de participação em um projeto de pesquisa vinculado ao Laboratório de Telecomunicações da UFES (LabTel) e ao Sistema único de Saúde (SUS), com o título “Aplicação da Tecnologia de Comunicação por Luz Visível em Monitorização de Recém Nascidos de Alto Risco”. O projeto em questão continua em funcionamento até os dias de hoje e faz parte de um programa ainda maior, de âmbito nacional, conhecido por PPSUS (Programa de Pesquisa para o SUS). Programa este que, como o próprio nome indica, seleciona projetos junto à diversas universidades do país, que apresentem potencial de desenvolvimento tecnológico para o já estabelecido sistema de saúde pública do Brasil.

Inicialmente, foi sugerida a implementação de apenas uma parte do sistema: o trecho *Back-to-Back* (entre dois microcomputadores *Raspberry Pi*), como forma de uma configuração "introdutória". Entretanto, por uma nova sugestão do coordenador deste projeto, foi-se adicionando os componentes necessários (LED, PLC, *smartphone*, dentre outros) para completar, ainda que de forma rudimentar, o *setup* atual, esperado também pelo projeto maior, parte do programa PPSUS. Dessa forma, acredita-se que o projeto possa contribuir como uma espécie de comprovação da aplicabilidade dos conceitos de comunicação por tecnologias VLC e PLC, dentro deste contexto hospitalar.

1.3 Objetivos

1.3.1 Objetivo Geral

O objetivo geral do trabalho consiste no desenvolvimento e na implementação de um sistema experimental de monitoramento de sinais vitais de pacientes internados, utilizando-se das tecnologias PLC e VLC para a comunicação de dados multiparamétricos em ambientes hospitalares classificados.

1.3.2 Objetivos Específicos

Os objetivos específicos do trabalho são:

a) Verificar a factibilidade prática de uma comunicação experimental entre dois microcomputadores RPi para a troca de mensagens clínicas em formato HL7;
b) Aferir a viabilidade da inserção de um conjunto extensor de sinal PLC ao sistema a fim de oferecer a este um maior alcance;
c) Verificar a factibilidade da integração, ao sistema, de um aparelho *smartphone* como receptor do sinal no espectro da luz visível, convertido de elétrico para óptico com o auxílio de uma lâmpada e um relé de estado sólido.

Excetuando-se os itens referentes a inserções de equipamentos no sistema, todos os demais objetivos específicos serão alcançados quase que exclusivamente por meio de programação dos dispositivos correspondentes. Com o desfecho positivo do projeto, espera-se que o sistema final sirva de base para outros trabalhos, bem como para a continuidade do projeto participante do

PPSUS, reforçando a plausibilidade de sua proposta. Espera-se, também, que os resultados aqui obtidos sirvam de ajuda para futuras tomadas de decisão, por parte do grupo do LabTel envolvido com o projeto, sempre em prol de uma implementação prática e financeiramente realística, porém também eficiente.

1.4 Organização do Trabalho

O trabalho foi dividido em 5 seções, a fim de uma melhor organização, os quais serão brevemente introduzidos nesta seção.

Iniciando-se pela segunda seção, tratou-se do embasamento teórico, parte fundamental para se compreender o projeto. Nesta etapa, explica-se os conceitos de comunicação PLC e VLC, tecnologias em destaque no projeto. Ainda, busca-se fazer uma breve revisão sobre os fundamentos de comunicação digital, além de introduzir os equipamentos que serão utilizados, bem como os padrões e protocolos que nortearão o trabalho.

Na seção 3, detalhou-se a estrutura física do projeto. Nele, se explicará como os equipamentos estarão interligados, o porquê de se utilizá-los e ainda como eles se encontram configurados (quando for o caso).

A seção 4, por sua vez, tratou da programação dos dispositivos. Como será explicado, para o funcionamento do sistema proposto, deverá ser elaborado uma grande quantidade de algoritmos, escritos em diferentes linguagens de código. Nesse sentido, esta seção busca descortinar os processos pelos quais cada algoritmo propõe tratar, seja por descrição explícita ou por meio de fluxogramas resumidos.

Por fim, a seção 5 apresentou os resultados finais obtidos a partir de testes experimentais em cima da metodologia proposta. O trabalho é concluído com discussões acerca destes valores, sempre com o sentido de buscar por soluções a se propor em trabalhos futuros.

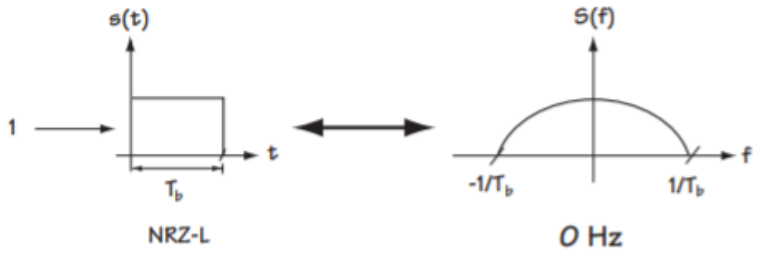
2 EMBASAMENTO TEÓRICO

2.1 Fundamentos de Comunicação Digital

Comunicação digital pode ser definida como o processo de se transmitir e receber um sinal de dados por meio de um conjunto finito de formas de ondas, normalmente operando em dois únicos estados (SKLAR, 2001). Nesse sentido, são sistemas muito eficazes pois quando do processo de captura do sinal, normalmente apresentam espaçados limiares de decisão para se classificar tal sinal. Sendo assim, é necessário que os possíveis distúrbios no canal sejam elevados o suficiente para que se considere um nível alto como baixo, ou o contrário (SKLAR, 2001).

Os sistemas de comunicação ainda podem ser classificados como sendo de banda base ou de banda passante. No primeiro, a comunicação se dá sem que se altere a frequência do sinal, transmitindo a informação do jeito em que esta originalmente se encontra. Já no segundo, ocorre um deslocamento do espectro do sinal, fazendo com que as frequências antes negativas agora transitem para o semieixo positivo, deixando o sinal centrado em frequência zero (NASSAR, 2001). Observa-se que neste trabalho será implementado um sistema digital em banda base, cujo funcionamento pode ser observado na Figura 1.

Figura 1 – Respostas no tempo e em frequência de um sinal em banda base



Fonte: Nassar (2001).
Nota: Adaptado pelo autor.

Além disso, se faz necessário que o sinal seja codificado e, então, modulado, antes de ser transmitido ao longo de um canal de comunicação que pode vir a ser um fio ou mesmo o ar livre. Estes processos serão detalhados nas seções 2.1.1 e 2.1.2. Finalmente, a fim de uma recuperação confiável da informação, espera-se que o receptor tenha uma taxa de amostragem

ao menos duas vezes maior do que a maior frequência transmitida, observando-se assim os princípios do teorema da amostragem de Nyquist (CASELLA; ANPALAGAN, 2018).

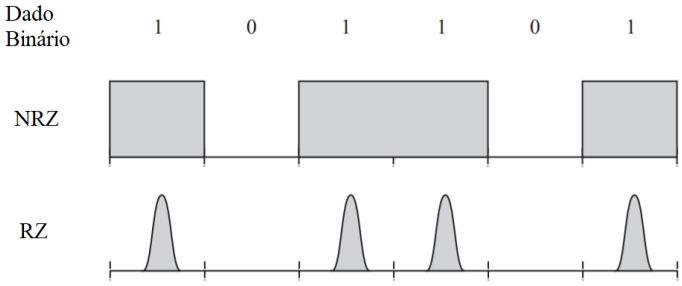
2.1.1 Codificação NRZ

Codificação em linha é um termo utilizado para as técnicas que representam digitalmente a informação binária como formas de onda, geralmente originadas pela presença ou ausência de uma sequência de pulsos. Também chamadas de modulação em código de pulso (PCM, do inglês *pulse-code modulation*), as técnicas de codificação em linha resultam em sinais puramente abstratos (SKLAR, 2001). Para que estes sejam capazes de tomar o meio físico, de fato, deverão ser posteriormente modulados, como explicado na seção 2.1.2. Existem quatro classificações básicas para as formas de onda em PCM, a seguir representadas: a de não retorno ao zero (NRZ, do inglês *non-return-to-zero*), a que retorna ao zero (RZ, do inglês *return-to-zero*), codificados em fase e, finalmente, binária de multinível.

A primeira, NRZ, é um exemplo de codificação em linha que durante todo o tempo de bit, mantém o sinal ativo (+V ou –V), ou seja, diferente de zero. Como observado por Sklar (2001), esta técnica é considerada como uma das mais simples e, em razão disso, também é a, provavelmente, mais utilizada. Nesse sentido, a codificação NRZ foi a escolhida para ser implementada neste trabalho justamente por conta de sua simplicidade, já que o objetivo deste é prover uma prova de conceito, inicial, para o sistema proposto. Na classificação RZ, ou de retorno ao zero, o sinal retorna ao nível de 0V ainda dentro de um tempo de bit. Já a classificação conhecida por codificação em fase possibilita que a forma de onda esteja tanto em nível alto (+V) quanto em nível baixo (-V) dentro de um mesmo período, representando as transições. Finalmente, a classificação binária de multinível, também chamada de bipolar, sugere que o sinal transite entre todos os três níveis (+V, 0 e –V) ao longo do tempo.

A Figura 2 demonstra a característica do NRZ em manter-se ativo durante todo o período de bit, em contraste com a técnica de codificação em linha de caráter antagônico à primeira (RZ). Nesta última, o sinal de fato é ativado durante o nível lógico alto do dado transmitido. Entretanto, ele “desliga”, e também demora a “ligar”, resultando em pequenos intervalos (menores que o tempo de *bit*) com sinal em 0 V, mesmo quando o dado encontra-se em um nível alto.

Figura 2 – Exemplificação dos códigos NRZ e RZ



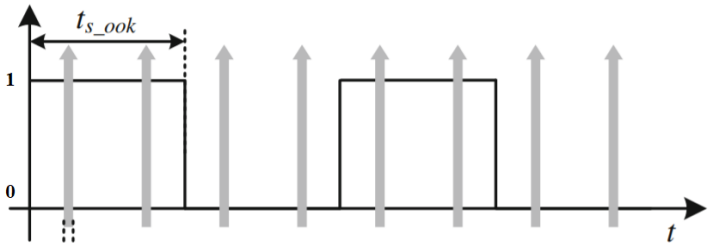
Fonte: Ramaswami, Sivarajan e Sasaki (2010).
Nota: Adaptado pelo autor.

2.1.2 Modulação OOK

Modulação de dados é basicamente o processo pelo qual os sinais digitais manipulados por um aparelho, são convertidos em sinais analógicos a fim de adentrarem e, então, serem transmitidos por um meio. Isto é necessário pois na realidade não existem canais digitais, apenas analógicos. Logo, sinais transmitidos em qualquer forma de comunicação, necessitam ser também analógicos como, por exemplo, impulsos elétricos, ópticos ou ainda ondas de rádio (TORRES, 2001).

Modulação por chaveamento liga-desliga (OOK, do inglês *on-off keying*), como o nome indica, é uma técnica de modulação digital que utiliza de apenas dois estados (binária) para representar os dados: ligado ou desligado (SKLAR, 2001). A Figura 3 representa uma das várias possíveis formas que estes dois estados poder tomar ao longo do tempo. Neste exemplo, o valor 1 representa um nível alto por todo o tempo de bit e o valor 0, um nível baixo pelo mesmo período.

Figura 3 – Sinal modulado com OOK

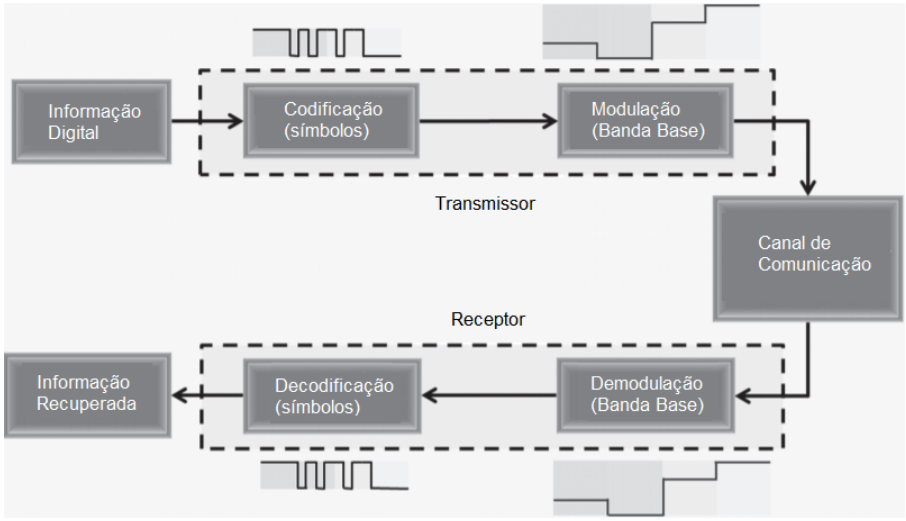


Fonte: Uysal e outros (2016).
Nota: Adaptado pelo autor.

Esta técnica também é conhecida como modulação binária por chaveamento em amplitude (BASK, do inglês *binary amplitude shift keying*), e é por vezes considerada uma técnica de modulação simples. Sua origem data do início do século XX, quando era muito comum de ser encontrada em sistemas de radiotelegrafia (SKLAR; 2001).

Como o projeto deste trabalho utilizará um método de modulação (OOK) em banda base, representou-se na Figura 4 um diagrama de blocos para demonstrar o funcionamento do processo de comunicação digital de um sinal assim modulado. Inicialmente, a informação é codificada com algum código de linha (como por exemplo o NRZ). Em seguida, esta é modulada para alguma forma de sinal analógico utilizando, por exemplo, o OOK. Algum dispositivo transdutor também pode ser usado para alterar a natureza do sinal. No caso do projeto, a lâmpada de LED tem tal função, convertendo energia elétrica em luminosa. Por fim, o sinal é recebido e demodulado de forma a se tornar novamente digital, quando é então decodificado, geralmente recuperando a informação original.

Figura 4 – Diagrama de blocos de uma comunicação digital em banda base



Fonte: Casella e Anpalagan (2018).
Nota: Adaptado pelo autor.

Por fim, é importante salientar que os sinais modulados em OOK não precisam necessariamente transitar entre apenas dois níveis. Na realidade, a Figura 5 demonstra que eles podem variar entre tantos níveis quanto se deseja, de forma a tornar a aparência da onda similar à de sinais analógicos, como é o caso das formas triangulares presentes na imagem.

Figura 5 – Exemplos de diferentes formas de modulação OOK

Fonte: DeCusatis (2014).

2.1.3 Taxa de Erros de Bit (BER)

Taxa de erro de bits (BER, do inglês *bit error ratio*) é um conceito definido por Breed (2003) como um indicador quantitativo utilizado para se verificar o desempenho, de ponta a ponta, de um sistema de comunicação. Seu resultado representa a proporção de erros observados em relação ao total de bits transmitidos em um dado intervalo de tempo, conforme pode ser observado na equação (1).

$$BER = \frac{N_e}{N_b} \tag{1}$$

Onde:

N_e é o número de bits errados;

N_b é o número total de bits recebidos.

No que tange os fatores que afetam o desempenho do BER, o ruído proveniente de componentes eletrônicos presentes no circuito, bem como do próprio canal de comunicação, pode ser considerado o seu maior inimigo. Outro fator que interfere diretamente na qualidade do BER, enquanto indicador de desempenho, é a quantização de erros decorrentes de uma eventual reconstrução do sinal digital quando realizada de forma incorreta ou ambígua (BREED, 2003).

2.2 Comunicação em Rede Elétrica (PLC)

Comunicação via rede elétrica (PLC) é uma tecnologia que surgiu com a promessa de possibilitar uma comunicação segura de dados aproveitando da infraestrutura e onipresença de tais sistemas. Com o passar dos anos, também foi se tornando cada vez mais popular devido à economia e à praticidade advindas junto a estas características. Sua utilização *indoor* é hoje

amplamente encontrada em aplicações residenciais, prediais, industriais, dentre outros (JANI; GARG; GUPTA, 2019).

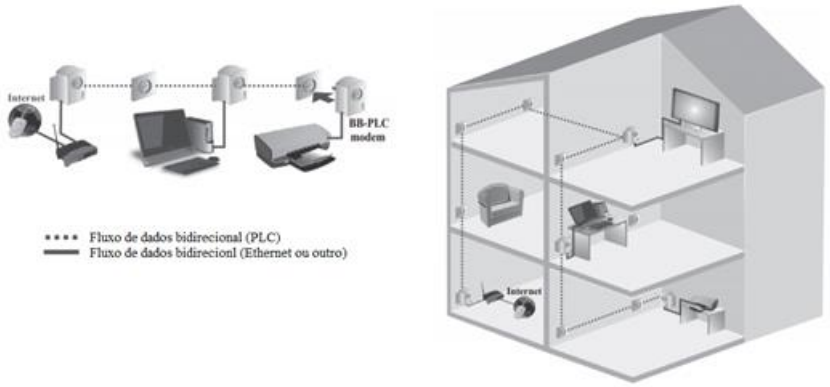
PLC é descrita por Casella e Anpalagan (2018) como uma antiga (datada de 1838), porém pouco utilizada, tecnologia de comunicação de dados sobre a rede elétrica. Segundo os autores, a dificuldade de se transmitir informação com segurança e confiabilidade sobre um meio que não foi projetado para esse propósito é a principal razão pela qual a tecnologia demorou a emplacar no cenário. No entanto, com o avanço das técnicas de comunicação digital, passou-se a ter um maior controle sobre a rede, possibilitando, assim, um aumento de qualidade no processo. Hoje, já é possível encontrar diversas áreas que utilizam a tecnologia PLC, como a automação residencial, indústria automotiva, empresas de distribuição de energia elétrica, dentre outros setores (CASELLA; ANPALAGAN, 2018). Quanto ao seu princípio de funcionamento, basicamente a modulação (em banda base ou passante) do sinal digital a ser transmitido (em altas frequências), para então injetá-lo na rede elétrica, onde já existe um sinal elétrico sendo transmitido a 60 Hz, por meio de um acoplamento de circuitos. É válido observar que o sistema também deve funcionar para redes que operem com corrente contínua (CASELLA; ANPALAGAN, 2018).

A tecnologia pode ser dividida em três grupos. O primeiro, de banda ultrafina (UNB, do inglês *ultra narrow band - plc*), opera com frequência entre 30 e 3000 Hz e é basicamente utilizado para gerenciamento de cargas na rede elétrica. Possuem grande alcance porém com taxas significantemente baixas (até 120 bps). Já o segundo, de banda estreita (NB-PLC, do inglês *narrow band - plc*), funciona entre 3 e 500 kHz, frequências baixas o suficiente para não interferirem no sinal das ondas de rádio AM. Assim, podem ser utilizados em ambientes ao ar livre sem nenhuma restrição. De fato, este grupo costuma ser encontrado em aplicações de medições em redes inteligentes (*smartgrids*). Por fim, o terceiro e último grupo, de banda larga (BB-PLC, do inglês *broad band - plc*), opera entre 1,8 e 250 MHz, podendo causar interferências em ondas de rádio FM (88, 1 a 108,1 MHz). Este grupo é o que atinge as taxas de transmissão mais elevadas (até 2 Gbps) e, por esse motivo, são geralmente utilizados em um maior conjunto de aplicações (CASELLA; ANPALAGAN, 2018).

No trabalho, os aparelhos utilizados enquadram-se no terceiro grupo (banda larga), os quais funcionam sob um padrão que já se encontra bem estabelecido no mercado e atende a mais de

200 milhões de aparelhos, o *HomePlug*, aprofundado na seção 2.4.4 (CASELLA; ANPALAGAN, 2018). Na Figura 6, observa-se como dispositivos PLC podem ser configurados de modo a oferecer uma rede de acesso local (LAN, do inglês *local access network*) por toda uma residência. Trocando-se a casa por uma clínica ou hospital, pode-se visualizar como a tecnologia seria capaz de prover comunicação entre diferentes ambientes de um mesmo local.

Figura 6 – Exemplo de aplicação residencial de um sistema PLC



Fonte: Casella e Anpalagan (2018).
Nota: Adaptado pelo autor.

É necessário pontuar que a comunicação pela rede elétrica também tem suas desvantagens. A primeira delas, e de maior impacto neste trabalho, é o fato de que este método também pode ser uma fonte de interferência eletromagnética, o que é fortemente evitado pela presente pesquisa. Sistemas PLC podem prover comunicação em taxas baixas (100 kbps) utilizando frequências de portadora também mais baixas, mas também podem oferecer transmissões até 500 Mbps operando com frequências entre 1 e 30 MHz, segundo observações de Rhee, Rhee e Park (2008) e de Movchan e outros (2020).

Como discutido anteriormente, essas faixas enquadram-se no espectro de RF e, sendo assim, podem ser responsáveis por causar interferências nos equipamentos médicos. Além disso, tais sistemas também sofrem com forte atenuação natural da rede elétrica para frequências mais altas ou mesmo distâncias maiores. Bem como são passíveis de grande flutuação devido às próprias cargas instaladas na rede (MOVCHAN et al., 2020).

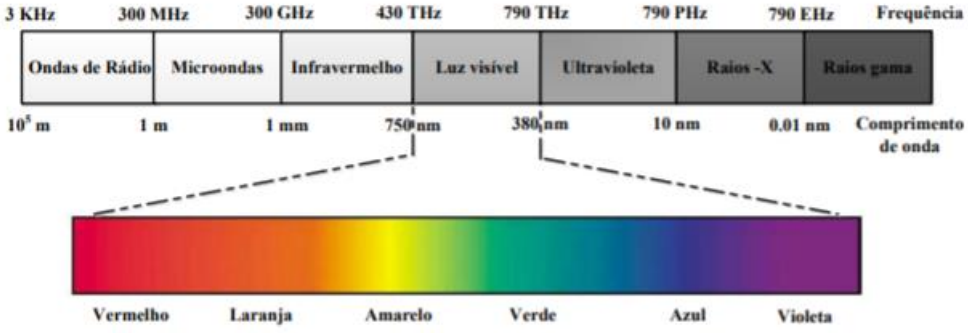
2.3 Comunicação por Luz Visível (VLC)

Comunicação por luz visível (VLC) é uma tecnologia de comunicação óptica (*wireless* ou sem fio), relativamente recente, que vem se popularizando ao longo dos anos pela possibilidade de se transmitir dados em altas velocidades simultaneamente a sua função básica de iluminar o ambiente sob o qual o sistema é instalado. A tecnologia foi primeiramente divulgada, sob este contexto de aplicação, no ano de 2000 pela Universidade Keio na cidade de Tóquio (Japão), com uma proposta de utilização de diodos emissores de luz (LEDs, do inglês *light-emtting diodes*) na cor branca, tanto para cumprir a função de iluminação residencial quanto como uma espécie de ponto de acesso local à rede mundial de computadores, ou internet (PATHAK et al., 2015).

Em seguida, já em novembro de 2003, foi-se criado um consórcio local (VLCC, do inglês *visible light comumunications consortium*) com propósito de regulação da tecnologia recém concebida, impulsionado pela crescente quantidade de pesquisas, acerca do assunto, divulgadas até então naquele país. Rapidamente a novidade se espalhou pelo mundo, culminando com o estabelecimento de seu primeiro padrão internacional oficial, o IEEE 802.15.7, datado de 2011. Dentre outros, o padrão trouxe as primeiras especificações técnicas de design de ambas as camadas físicas e de enlace (*link*) para sistemas que utilizem alguma forma de comunicação por meio da luz visível (PATHAK et al., 2015).

VLC é descrita por Ghassemlooy e outros (2017) como uma subárea da tecnologia definida pelo termo comunicação óptica sem fio (OWC, do inglês *optical wireless communication*). De seu nome, extrai-se a faixa do espectro sob a qual a VLC é capaz de operar. Isto é, entre 380 e 750 nm de comprimento de onda (ou 430 a 790 THz de frequência), como pode ser observado na Figura 7. Esta faixa é caracterizada pela luz visível, diferenciando-se, por sua vez, do infravermelho (impossível de ser enxergado pelo olho humano). Sua utilidade encontra-se no fato de poder aproveitar estruturas com função de iluminação, já instaladas e em funcionamento, para simultaneamente propagar dados com elevadas taxas de transmissão, como 10 Gbps.

Figura 7 – Espectro eletromagnético e a faixa de luz visível



Fonte: Neves (2018).

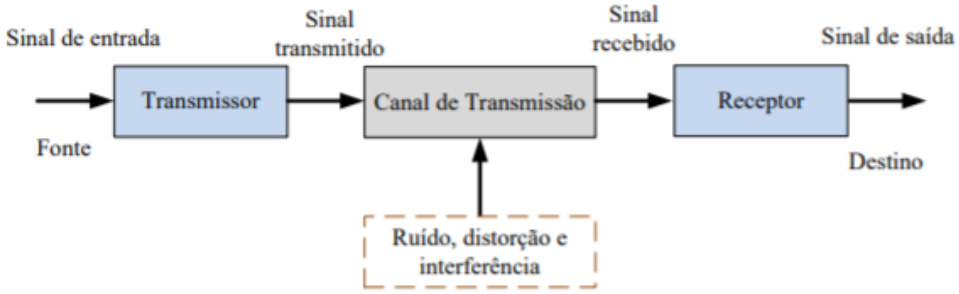
Tendo aplicação em variados segmentos como em redes de acesso pessoal, serviços de localização *indoor*, redes de veículos e transportes subterrâneos, a tecnologia vem ganhando espaço na medida em que dispositivos eletrônicos tornam-se cada vez mais conectados. No cenário de internet das coisas (IoT, do inglês *internet of things*), buscar alternativas para as sobrecarregadas redes Wi-Fi torna-se uma necessidade, principalmente em função de sua banda ser licenciada e também limitada. Assim, expandir o espectro da comunicação para outras regiões livres de regulamentação (cerca de centenas de terahertz) é tido como solução em potencial para o problema em análise (GHASSEMLOOY et al, 2017).

Outra vantagem associada a sistemas de comunicação VLC é a segurança com relação a confidencialidade do sinal. Devido ao seu natural confinamento entre paredes, as informações não podem ser acessados de fora do ambiente em que o sinal é transmitido. Assim, eventuais dados confidenciais enviados, ainda mantêm-se ao menos em relação ao exterior do local. Por fim, ainda é válido destacar a característica dos próprios LEDs em serem dispositivos eficientes e duráveis, consumindo pouca energia elétrica em geral (NEVES, 2018).

Seu princípio de funcionamento pode ser visualizado com o auxílio da Figura 8. Nela, observa-se um diagrama de blocos simplificado de um sistema de comunicação VLC. Para o caso de sua implementação, tem-se usualmente um sinal de entrada elétrico que por sua vez é convertido para óptico por meio de um dispositivo transdutor, a saber, um único LED ou um conjunto destes. Em seguida, o novo sinal toma o ar como canal de transmissão, onde ruídos e interferências (brilhos de possíveis fontes próximas) podem ser adicionados à onda. Finalmente,

esta é capturada no lado receptor do sistema por um dispositivo que possua sensores de fotodetecção, de onde então se decodifica a informação original diretamente do sinal portador do dado.

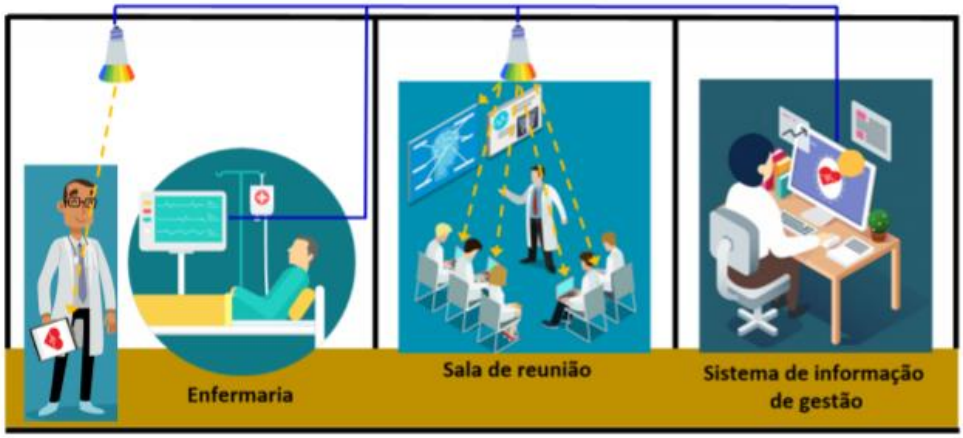
Figura 8 – Diagrama de blocos generalizado para a comunicação VLC



Fonte: Neves (2018).

É válido ressaltar que existem várias técnicas de se codificar a informação e também de se modular os sinais elétrico e óptico. Neste projeto, por exemplo, serão utilizados modulação OOK e codificação NRZ, ambos já detalhados na seção 2.1. Por fim, destaca-se que no presente trabalho a tecnologia VLC encontra-se imersa em um contexto de utilização hospitalar. Nesse sentido, a Figura 9 busca ilustrar tal forma de aplicabilidade.

Figura 9 – Ilustração do sistema VLC implementado em um ambiente hospitalar



Fonte: Neves (2018).

2.4 Padrões e Protocolos

2.4.1 HL7

HL7 (acrônimo para *Health Level Seven*) é um padrão de mensagens clínicas utilizado como referência na comunicação de dados clínicos pela maior parte dos sistemas de informática médica encontrados na atualidade. O objetivo do protocolo é prover uma estrutura padronizada de mensagens médicas que contemplem informações acerca da identificação do paciente, de seu estado de saúde, dentre muitas outras funcionalidades (OFFICE OF THE ASSISTANT SECRETARY FOR PLANNING AND EVALUATION, 2003).

Na Figura 10 observa-se um exemplo de uma mensagem HL7, coletada diretamente de um monitor multiparamérico, equipamento que faz as medições dos sinais vitais do paciente internado em um hospital. A mensagem foi posteriormente modificada de forma a se colocar o nome do autor deste projeto como se fosse o paciente em monitoramento, dentre outras modificações, para efeitos de demonstração do protocolo.

Figura 10 – Exemplo de uma mensagem típica com padrão HL7

```
MSH|^~\&|northern|patient monitor^|||20132034620000||ORU^R01^ORU_R01|4056|P|2.6|
PID|||911||Lucas^MartinsRimolo||00000000|M|
PV1||I|^^
OBR|1||||||20132034620000|
OBX|1|NM|147842^MDC_ECG_HEART_RATE^MDC|1.7.4.147842|100|264864^MDC_DIM_BEAT_PER_MIN^MDC|||||F||||
OBX|2|NM|150456^MDC_PULS_OXIM_SAT_O2^MDC|1.3.1.150456|100|262688^MDC_DIM_PERCENT^MDC|||||F||||
OBX|3|NM|149530^MDC_PULS_OXIM_PULS_RATE^MDC|1.3.1.149530|100|264864^MDC_DIM_BEAT_PER_MIN^MDC|||||F||||
OBX|4|NM|151578^MDC_TTHOR_RESP_RATE^MDC|1.7.1.151578|100|264928^MDC_DIM_RESP_PER_MIN^MDC|||||F||||
OBX|5|NM|150344^MDC_TEMP^MDC|1.2.1.150344|100|268192^MDC_DIM_DEGC^MDC|||||F||||
OBX|6|NM|150344^MDC_TEMP^MDC|1.2.2.150344|100|268192^MDC_DIM_DEGC^MDC|||||F||||
```

Fonte: Produzido pelo próprio autor.

A estrutura da mensagem basicamente consiste na divisões entre segmentos e campos. Cada linha do arquivo delimita um segmento, cuja funcionalidade pode ser de transmitir a informação dos dados pessoais de um paciente, de uma medição de um determinado sinal vital (observação) ou mesmo da própria mensagem em si, como a versão do protocolo em uso, por exemplo. De fato, há mais de cem funcionalidades disponíveis na versão 2.6, utilizada pelo monitor do laboratório. Para o projeto, entretanto, somente cinco serão transmitidas pelos arquivos de teste, já que foram as únicas recebidas quando o equipamento foi testado. De volta à Figura 10, é possível observá-las através de seus cabeçalhos, representados logo nos inícios das linhas, a saber: MSH, PID, PV1, OBR e OBX.

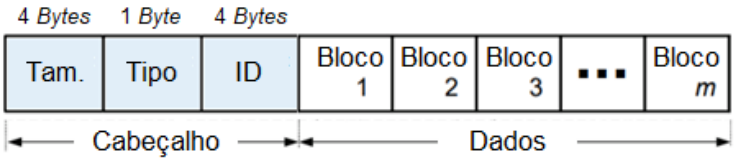
O primeiro cabeçalho do exemplo, MSH, é aquele que define a origem, o destino, dentre outras sintaxes da mensagem. Nele se observa a versão do protocolo e a data do arquivo. É importante ressaltar que o caractere de barra vertical ("|") é o responsável por separar os campos dentro de um mesmo segmento. Seguindo, o segundo cabeçalho, PID, é utilizado para a identificação do paciente. Nele se observa seu ID (911 para o exemplo), nome e sexo. PV1 refere-se à condição da pessoa monitorada enquanto sua permanência no hospital. O "I" observado indica *inpatient* ou internado. Finalmente OBR e OBX tratam das medições. O primeiro indica o pedido pela transmissão, onde o valor 1 mostra que foi o primeiro pedido (também é possível observa a data). Já o segundo, é onde se transmite de fato.

Os segmentos OBX são os mais importantes para este trabalho, visto que neles se carrega a informação do valor medido pelo sensor do medidor multiparamétrico utilizado. É importante destacar que cada segmento OBX retorna uma única medição de um único sensor. Por exemplo, na última linha do exemplo, o termo "TEMP" indica que foi feita uma medição de temperatura. Ainda, deste segmento, extrai-se o número da medição, 6, o valor desta, 100, além da identificação (ID) do sensor. O campo "F" recebido ao final não indica sexo, mas na realidade o estado quanto ao resultado da observação (neste caso F indica resultado final). Em conclusão, informa-se que as finalidades dos demais segmentos, bem como dos campos a eles associados, da versão 2.6 podem ser pesquisados mais aprofundadamente em Caristix (2021), caso necessário.

2.4.2 SFTP

SFTP, acrônimo para *SSH File Transfer Protocol*, é um protocolo que provê acesso, transferências e edições de arquivos de forma segura sobre qualquer fluxo bidirecional (confiável) de dados. Na realidade, o SFTP é uma espécie de união entre outros dois protocolos: um de transferência simples de arquivos entre duas máquinas, FTP, e outro que oferece um método de acesso remoto, seguro, sobre redes essencialmente vulneráveis, o SSH versão 2.0 (IETF DATATRACKER, 2006). Pode-se observar, na Figura 11, a estrutura de um pacote SFTP padrão.

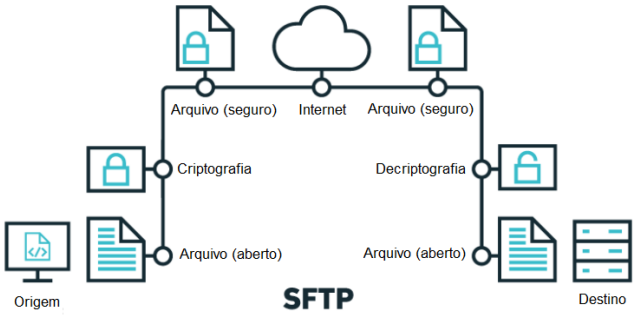
Figura 11 – Mensagem típica do protocolo SFTP



Fonte: Produzido pelo próprio autor.

Seu princípio de funcionamento, por sua vez, encontra-se ilustrado na Figura 12. Como pode-se observar para um exemplo de envio de arquivo, inicialmente criptografa-se o dado logo após este sair da máquina de origem (cliente). Em seguida, o arquivo adentra um túnel de comunicação segura, estabelecido entre ambos os computadores. Desse modo a transferência estará protegida mesmo que, na realidade, esta ocorra sob uma rede insegura, como, por exemplo, a internet. Finalmente, ao chegar ao destino, o arquivo é decriptografado, voltando ao seu estado de apresentação original.

Figura 12 – Princípio de funcionamento do protocolo SFTP



Fonte: WP Engine (2021).
Nota: Adaptado pelo autor.

2.4.3 UART

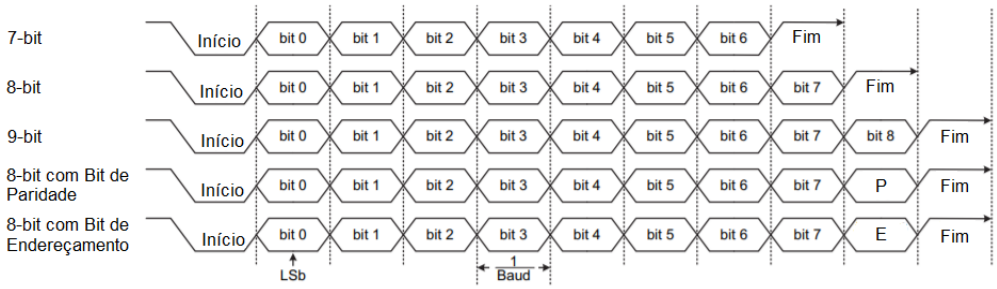
UART, acrônimo para *Universal Asynchronous Receiver/Transmitter*, é um protocolo que atua sobre a comunicação serial de equipamentos como, por exemplo, o *Raspberry Pi*. Na realidade, o termo UART também pode ser utilizado para se referir ao circuito eletrônico, presente em computadores e microcontroladores, que de fato realiza a conversão do formato do dado de paralelo para serial e vice-versa (MACKAY et al., 2004). Junto a outros métodos de comunicação como o SPI e o I²C, o *hardware* UART já vem incluso em diversos microcontroladores, tais quais o *Arduino UNO*, ESP32, ou *Raspberry Pi* (RPi). Entretanto,

diferentemente destes outros protocolos de comunicação, o UART destaca-se por ser assíncrono. Ou seja, os dispositivos não precisam compartilhar uma espécie de sinal de relógio (*clock*), por meio de um fio, como visto em outros métodos (MICROCHIP, 2017).

Para que a comunicação seja possível, entretanto, existe a necessidade de se enviar *bits* que indiquem o início e o fim desta, adicionados à informação como cabeçalhos e rodapés, respectivamente. Normalmente, utiliza-se o bit 0 como indicador de início de transmissão e o bit 1 como de término. Isto sugere que o sinal de espera (*idle*) é mantido em nível alto desde o estabelecimento da comunicação. Assim, o dispositivo de recepção busca constantemente pela descida do sinal, indicando que um pacote foi enviado e precisa ser lido. Ainda, é importante destacar que o indicador de parada pode ser configurado para dois bits, enviando '11' em vez de apenas '1', por exemplo. Tal método funciona como uma garantia adicional de parada.

Outra característica interessante do protocolo é a possibilidade de se adicionar bits de paridade e de endereçamento logo após o último bit de dados. Pode-se ainda observar que o UART permite que sejam transmitidos de 7 a 9 bits puramente de dados, ou seja, possibilita que o tamanho da informação também seja ajustável. A Figura 13 demonstra as diversas possibilidades disponibilizadas pelo protocolo.

Figura 13 – Diferentes possibilidades de representações com a comunicação UART



Fonte: Microchip (2017).
Nota: Adaptado pelo autor.

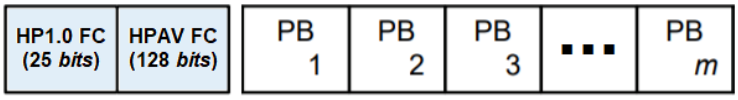
2.4.4 *HomePlug* AV

HomePlug AV é uma especificação de comunicações via rede elétrica, datada de 2005, que tem como principal objetivo, padronizar a forma com que aparelhos PLC de diferentes marcas comunicam-se entre si. Outro propósito do protocolo é aumentar a qualidade da comunicação,

bem como sua velocidade. De fato, a especificação prevê uma taxa no canal de 200 Mbps para áudio, vídeo e dados em geral, sendo 150 Mbps a taxa de informação útil, representando um considerável aumento em relação aos 14 Mbps da versão anterior (SOLWISE, 2005).

Com o auxílio da Figura 14, pode-se observar como é estruturado o pacote (MPDU) de subcamada MAC, acrônimo para *medium access control*, do protocolo especificado. Na cor azul, encontram-se os cabeçalhos, que têm como função fazer a interpretação ou o controle do quadro, daí a sigla *frame control* (FC). É nessa parte que se indica quantos blocos de dado ou então qual a versão do protocolo serão utilizados. O primeiro bloco do cabeçalho, de 25 bits, só é transmitido caso algum dispositivo da rede necessite se comunicar sob um protocolo mais antigo (*HomePlug* 1.0), sendo totalmente dispensável em caso contrário. É possível perceber que não há um número limite para envio de blocos de dados, ou *payload*, representados na imagem pela cor branca (LATCHMAN et al., 2013).

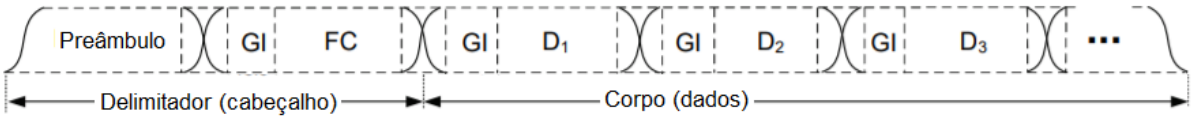
Figura 14 – MPDU típico do protocolo *HomePlug* AV



Fonte: Latchman e outros (2013).
Nota: Adaptado pelo autor.

Já na camada física, os pacotes MPDU são encapsulados e então transmitidos pela rede elétrica com o auxílio da técnica de modulação conhecida como multiplexação por divisão de frequências ortogonais (OFDM, do inglês *orthogonal frequency multiplexing/multiple access*). Quando isto ocorre, os pacotes passam a ser chamados de PPDU, acrônimo para *Physical* PDU. Basicamente, cada bloco do MPDU é codificado e tratado de forma a representar um único símbolo OFDM, que então é transmitido simultaneamente em diversas frequências diferentes, por meio de subportadoras, melhor aproveitando a capacidade do canal (LATCHMAN et al., 2013). A Figura 15 mostra um PPDU padrão.

Figura 15 – PPDU típico do protocolo *HomePlug* AV



Fonte: Latchman e outros (2013).
Nota: Adaptado pelo autor.

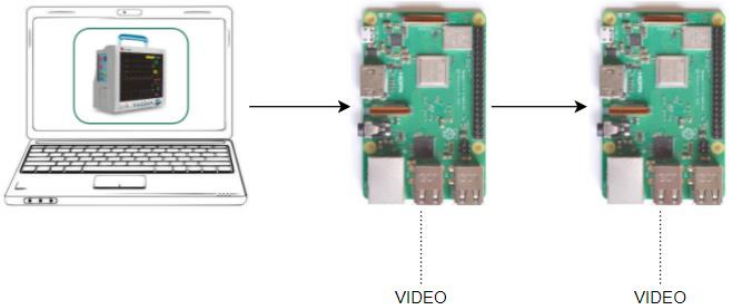
3 SISTEMAS HÍBRIDOS DE COMUNICAÇÃO IMPLEMENTADOS

Nesta seção, discute-se a solução física que foi proposta e implementada no projeto. A fim de uma maior organização, dividiu-se a instalação em três diferentes etapas ou configurações. A primeira delas, trata exclusivamente da comunicação com fio, entendida como um passo inicial para o estabelecimento da configuração final do sistema. Em seguida, adiciona-se o conjunto extensor de alcance para transmitir e receber os sinais modulados por meio da rede elétrica, tecnologia conhecida como *Powerline Communication*. Por fim, integram-se no sistema os dispositivos foto-emissores e fotodetectores a fim de se estabelecer o trecho final da comunicação sob o domínio da luz visível.

3.1 Configuração 1 (B2B)

A configuração inicial do projeto consiste basicamente na interconexão por meio de cabos e fios (*Back-to-Back*) entre os dispositivos que geram e recebem os sinais estudados. Como pode ser observado no diagrama da Figura 16, o computador é o equipamento responsável pela geração dos arquivos HL7, que contêm todas as informações pertinentes à situação clínica de um determinado paciente sob monitoramento. É importante ressaltar que a máquina não será utilizada quando de uma eventual aplicação prática em hospitais, visto que esta só se faz necessária para o projeto em função da necessidade de uma substituição do medidor multiparamétrico (PROLIFE modelo P12) emprestado, ao LabTel, pelo HUCAM. Finalmente, após tomar o meio físico, o sinal chega ao primeiro microcomputador *Raspberry Pi*, dito de transmissão, para enfim entrar no canal da comunicação e lá se propagar até chegar ao dispositivo espelhado no lado da recepção.

Figura 16 – Representação conceitual do *setup* 1 (B2B)



Fonte: Produzido pelo próprio autor.

3.1.1 *Notebook* como Monitor Multiparamétrico

Monitor multiparamétrico é um equipamento capaz de ler, interpretar e apresentar, de forma visual, múltiplos sinais elétricos recebidos das saídas de sensores digitais ou analógicos como alguma forma de monitoramento (AURION, 2021). É possível perceber que nesta aplicação específica se faz necessária uma comunicação de dados com rapidez e alta confiabilidade, já que trata-se da preservação de vidas humanas. Nesse sentido, falhas em qualquer ponto do sistema podem significar perdas de vidas. Quanto aos sinais a serem obtidos no monitoramento clínico, podem-se extrair das especificações técnicas da família de monitores PROLIFE, presentes em Aurion (2021), os seguintes exemplos de indicadores disponíveis para tal fim: pulsação cardíaca, pressão invasiva e não-invasiva, temperatura, frequência respiratória e oximetria (SpO2).

Ressalta-se, ainda, que a forma como as medições encontram-se dispostas em um arquivo HL7 serão abordadas, neste texto, à altura da seção 4. Como forma de ilustração na Figura 17 pode-se observar em maiores detalhes um exemplo situacional de uma tela de um medidor clínico-hospitalar.

Figura 17 – Medidor multiparamétrico de sinais vitais

Fonte: Aurion (2021).

A escolha de um computador pessoal portátil (*notebook*) para representação, dentro do projeto, de um medidor multiparamétrico, em nada prejudica o desenvolvimento do trabalho. Justifica-se, para tanto, que o primeiro aparelho é igualmente capaz de gerar os mesmos arquivos HL7,

do equipamento clínico, através de medições simuladas. Ou seja, em vez de buscar por novos valores medidos junto aos sensores, dada uma determinada taxa de atualização, este projeto utiliza valores já pré-definidos (salvos em arquivos HL7) e os utiliza, arquivo por arquivo, como uma nova atualização. Ainda, foi tomado como base uma taxa de uma nova amostra, ou arquivo, a cada segundo, reafirmando a ideia de um monitoramento contínuo e ininterrupto. Para fins de documentação do projeto, declara-se que a máquina utilizada como medidor foi um notebook *Dell Inspiron* i14-7460-A20G, com processador *Intel* de sétima geração, 8 GB de memória RAM e *Windows* 10 instalado.

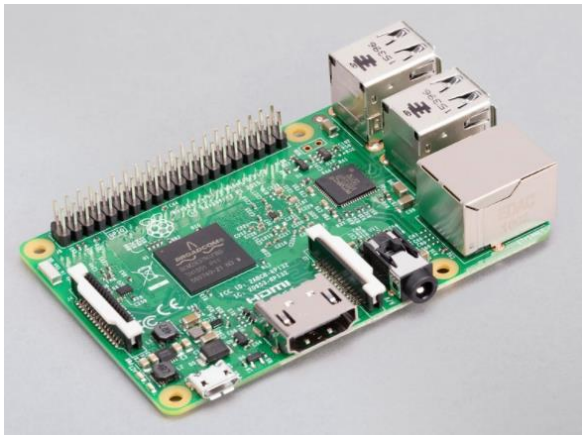
A comunicação entre o computador e o *Raspberry* de transmissão, se deu com o auxílio de um cabo Ethernet (CAT5, de até 100 Mbps) com conectores 8P8C (popularmente conhecido como RJ45), sob supervisão do protocolo seguro de transferências de arquivos SFTP, já detalhado na seção 2.5.2. Nesse cabo, os arquivos são passados de um ponto ao outro, controlados por um código de programação em linguagem de programação *Python* desenvolvido pelo autor e melhor detalhado na seção 4.

É importante frisar que a seleção do protocolo de transferências de arquivos se deu em razão de sua capacidade de prover segurança à mensagem transmitida, além de representar uma implementação em *Python* e sob auxílio da biblioteca conhecida como *Paramiko* (PARAMIKO, 2021).

3.1.2 Par de Microcomputadores *Raspberry Pi*

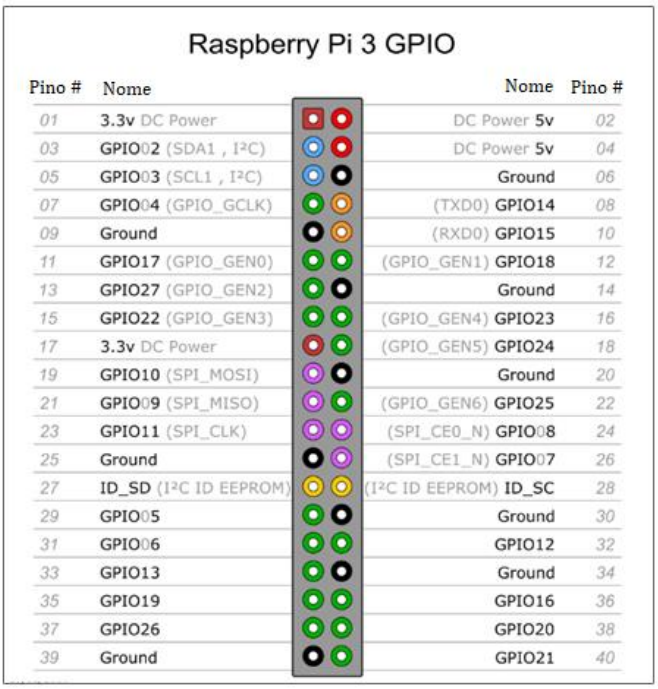
Raspberry Pi (Figura 18) é uma série de microcomputadores de boa performance e baixo custo que auxilia pessoas de todas as idades no aprendizado de programação em linguagens como *Scratch* e *Python* (RASPBERRY PI, 2015). Além de oferecer tudo o que se espera de um computador regular, o *Raspberry* ainda é capaz de interagir com o mundo real por meio de seus pinos de de entrada e saída, abreviados por GPIO (do inglês, *General Purpose Input/Output)*, novamente conforme observado em Raspberry Pi (2015). Os dois exemplares utilizados no projeto, pertencem ao modelo B da terceira geração de placas *Pi*, cada um contendo 40 pinos, como demonstrado na Figura 19. Em sua página oficial, encontrada em Raspberry Pi (2021a), podem ser extraídas as demais especificações da placa.

Figura 18 – Microcomputador *Raspberry Pi*



Fonte: Raspberry Pi (2021a).

Figura 19 – Mapeamento dos pinos GPIO em uma RPi3



Raspberry Pi 3 GPIO

Pino #	Nome	Nome	Pino #
01	3.3v DC Power	DC Power 5v	02
03	GPIO02 (SDA1 , I²C)	DC Power 5v	04
05	GPIO03 (SCL1 , I²C)	Ground	06
07	GPIO04 (GPIO_GCLK)	(TXD0) GPIO14	08
09	Ground	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	Ground	14
15	GPIO22 (GPIO_GEN3)	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Ground	20
21	GPIO09 (SPI_MISO)	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	(SPI_CE0_N) GPIO08	24
25	Ground	(SPI_CE1_N) GPIO07	26
27	ID_SD (I²C ID EEPROM)	(I²C ID EEPROM) ID_SC	28
29	GPIO05	Ground	30
31	GPIO06	GPIO12	32
33	GPIO13	Ground	34
35	GPIO19	GPIO16	36
37	GPIO26	GPIO20	38
39	Ground	GPIO21	40

Fonte: Medium (2021).
Nota: Adaptado pelo autor.

A utilização de um par de microcomputadores *Raspberry Pi* pelo projeto, deveu-se principalmente por sua capacidade de geração, codificação, modulação, além da própria recepção de sinais elétricos de forma eficaz e controlada. Alia-se a isto, também o seu maior poder de processamento, quando comparado a microcontroladores comuns no mercado tais como *o Arduino UNO* e o ESP32. Ainda, destaca-se que os dispositivos foram utilizados em lugar do gerador de sinais de um osciloscópio que, até o início deste trabalho, foi o responsável pelos sinais em estudo. Este último equipamento teve importância para o desenvolvimento da

pesquisa enquanto o *Raspberry* não podia ser nela integrado. Por fim, a ideia inicial do projeto era oferecer duas formas de recepção do sinal óptico transmitido: uma por *smarthphone* e a outra por um segundo RPi (acoplado a uma placa com fotodetectores). Nesse sentido, o *Raspberry* receptor já se encontrava disponível para os testes de recuperação de sinal do presente trabalho aqui em análise.

3.2 Configuração 2 (PLC-B2B)

A segunda configuração implementada no projeto foi obtida com a simples adição de uma rede PLC ao circuito transmissor original, como pode ser bem observado no diagrama da Figura 20. Para tanto, foi necessário lançar mão de dispositivos conhecidos como extensores de alcance *Powerline*, para fazer esse intermédio. O aparelho transmissor recebe o sinal (arquivo HL7) diretamente do computador e o transmite pela rede elétrica ao ser colocado junto à tomada.

Figura 20 – Representação conceitual do *setup* 2 (PLC-B2B)

Fonte: Produzido pelo próprio autor.

Do lado da recepção, um segundo dispositivo PLC também é ligado na tomada para, desta vez, capturar a informação modulada pela rede elétrica. Em seguida, ele é conectado ao *Raspberry*, no qual os dados poderão ser tratados de acordo com algum dos protocolos propostos neste projeto, para então serem finalmente transmitidos para um segundo *Raspberry*, exatamente como na primeira configuração.

3.2.1 Extensores de Alcance *Powerline*

Como explicado anteriormente, para a implementação de uma sistema de comunicação com tecnologia PLC, é necessário dispor de um par de dispositivos extensores de alcance. Sendo assim, o presente trabalho contou com um conjunto obtido novamente junto ao Laboratório de

Telecomunicações da UFES. O transmissor, observado no lado esquerdo da Figura 21, de modelo TL-PA4010 (TP-LINK, BR), apresenta a frase “AV 500 *Nano Powerline Adapter*” como sua descrição de produto. O termo “AV500”, presente na descrição dos produto, indica que ele atende à especificação conhecida como *HomePlug* AV, garantindo intercompatibilidade entre diversos equipamentos de outras marcas ou modelos. Além disso, o numeral 500 indica que o produto também encontra-se limitado a taxas de transmissão de até 500 Mbps. Quanto ao seu alcance máximo, é indicado em suas especificações técnicas que o aparelho foi projetado para alcançar até 300 metros de circuito com um bom sinal (TP-LINK, 2017).

Figura 21 – Conjunto extensor de alcance *powerline* utilizados no projeto

Fonte: TP-LINK (2021).

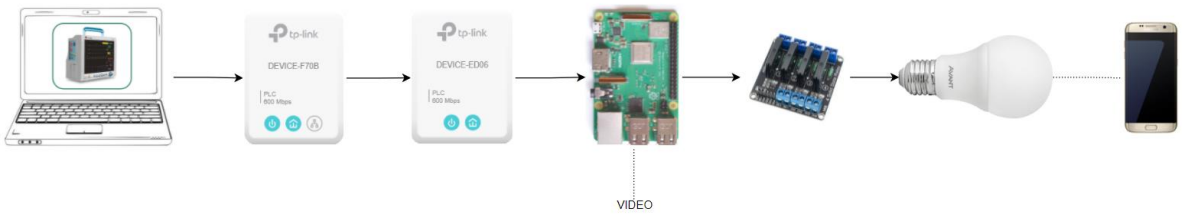
Na outra ponta do circuito, o adaptador da recepção ilustrado no lado direito da Figura 23, e de modelo TL-WPA4220 (TP-LINK, BR), possui o termo “300Mbps AV500 *WiFi Powerline Extender*” como descrição. Novamente, é possível observar que ele também atende ao protocolo *HomePlug* AV e, além disso, de suas especificações técnicas encontradas em TP-Link (2016), extrai-se ainda sua limitação de alcance, a mesma do dispositivo transmissor (300 metros). O seu diferencial, entretanto, encontra-se na sua capacidade adicional de atuar como ponto de acesso Wi-Fi, propagando simultaneamente o sinal em uma rede sem fio (de até 300 Mbps) sempre que recebido pela rede elétrica, caso assim seja configurado. Todavia, para este projeto em específico, tal função não será utilizada, de forma a se manter uma coerência com o que se espera para eventuais aplicações futuras, do sistema proposto em ambientes hospitalares classificados (TP-LINK, 2016).

É interessante destacar que essas limitações são apenas simbólicas, visto que são obtidas a partir de testes com condições muito próximas das ideais e que, na prática, não se percebem no dia a dia. No caso da limitação de velocidade, por exemplo, tal problema torna-se ainda pior: ambos os dispositivos foram projetados com portas Ethernet que permitem tráfego de dados em até 100 Mbps. Ou seja, por mais que se cumpra todos os requisitos para uma transmissão limpa e veloz, o seu sistema fica limitado a esta limitação das portas.

3.3 Configuração 3 (PLC-VLC)

Nesta última etapa do *setup*, adicionam-se uma lâmpada de LED e um aparelho *smartphone* ao circuito, de forma a representar a tecnologia VLC no projeto. Para isso, o *Raspberry* da recepção teve de ser removido juntamente com os fios que o ligavam ao dispositivo espelhado do lado transmissor. Com a implementação de tais alterações, finalmente pôde-se chegar à configuração física definitiva pretendida para o projeto, como pode-se observar através da Figura 22.

Figura 22 – Representação conceitual do *setup* 3 (PLC-VLC)



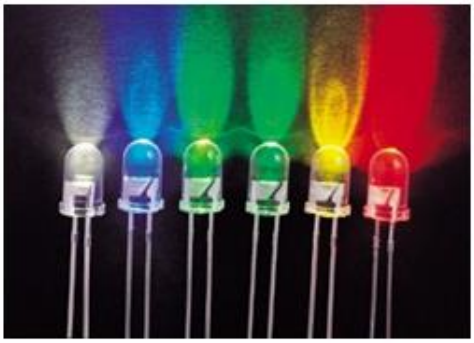
Fonte: Produzido pelo próprio autor.

De forma sucinta, a informação que antes era passada em forma de sinal elétrico – saindo diretamente de um pino GPIO do primeiro microcomputador e chegando, com o auxílio de um fio, a outro pino em um segundo dispositivo *Raspberry Pi* – deve agora ser transformada em um sinal luminoso advindo de uma lâmpada de LED. Tão somente quando o brilho deste último tome o ar livre por canal de transmissão, sendo então precisamente percebido pelo sensor de luminosidade de um celular *smartphone*, pode-se dizer que a comunicação por luz visível foi estabelecida.

3.3.1 Lâmpada de LED

O Diodo emissor de luz (LED, do inglês *Light-Emitting Diode*) é por definição um diodo de junção P-N, especialmente dopado, capaz de emitir luz (Figura 23). Em um sistema VLC, este componente, ou um conjunto deles (lâmpadas de LED), é quem de fato transmite os dados por meio de variações em seu brilho. Como um exemplo, quando o brilho medido na recepção do sinal, dado um instante *t* qualquer, é superior a um valor já pré-estabelecido como nível alto, associa-se um *bit* 1 para a informação naquele instante específico. De maneira semelhante, quando o brilho encontra-se abaixo de um valor pré-definido e esperado, associa-se um *bit* 0 à informação capturada.

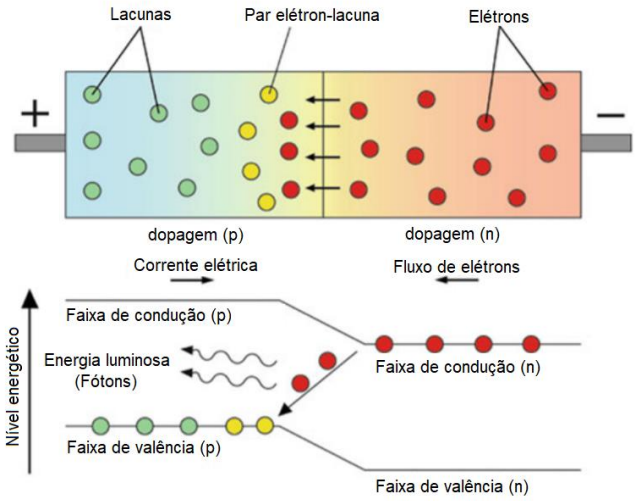
Figura 23 – Dispositivos LED emitindo luz



Fonte: Creativentechno (2021).

É interessante perceber que, para a comunicação VLC, ao menos da parte da transmissão, os dispositivos LED funcionam também como transdutores. Pode-se assim dizer pois são estes os componentes responsáveis pela conversão do sinal elétrico em óptico, para dado sistema. O princípio de funcionamento deste fenômeno (Figura 24) baseia-se na própria premissa dos diodos, que, quando positivamente polarizados, têm seus elétrons e lacunas movimentando-se e trocando constantemente de posição uns com os outros. À medida que o tempo vai passando, as cargas negativas vão tomando os lugares das lacunas na junção P e, por conseguinte, descem para níveis mais baixos de energização. Quando isso ocorre, são liberados pacotes de energia no espectro de luz visível, ou fótons. Previsivelmente, a energia óptica liberada em forma de fótons, ou luz, tem justamente o valor da diferença energética observada entre os dois níveis ocupados pelos elétrons (GUPTA, 2017).

Figura 24 – Princípio de funcionamento de um LED



Fonte: Gupta (2017).
Nota: Adaptado pelo autor.

Para o presente projeto, a lâmpada utilizada foi a PERA LED AVANT (AVANTLUX, BR) de 9W, cor branca e linha de série MK-D, cujas especificações técnicas podem ser encontradas em Avant (2021). Destaca-se também que o modelo utilizado encontra-se ilustrado na Figura 24, relativa à configuração final do *setup*. Ainda, de acordo com suas especificações, pôde-se exprimir importantes informações acerca das características físicas e de seu funcionamento, as quais podem ser resumidas no Quadro 1.

Quadro 1 – Especificações técnicas da lâmpada utilizada

Grandeza	Valor
Potência	9 W
Fluxo luminoso	803 lm
Eficiência luminosa	89 lm/W
Temperatura de cor	6500 K
Tensão nominal	100 V – 240 V
Corrente nominal a 127 V	70 mA
Corrente nominal a 220 V	40 mA
Fator de potência	> 0,70
Frequência	60 Hz

Fonte: Produzido pelo próprio autor.

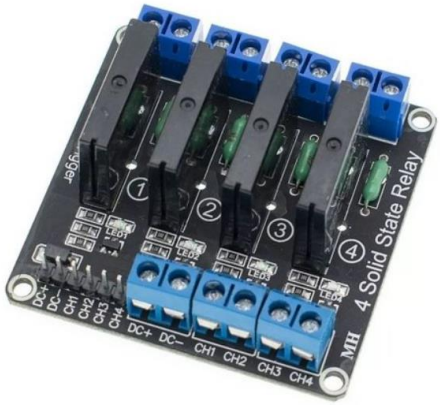
Há dois pontos a se ressaltar, entretanto. O primeiro, de que a lâmpada foi escolhida por conveniência, não havendo a necessidade de uso exclusivo para o modelo colocado em questão. O segundo ponto levanta a questão de que não existe a possibilidade de se alimentar uma

lâmpada de LED, comum, encontrada no mercado, diretamente dos pinos de saída GPIO de um *Raspberry Pi*. Isto se deve ao fato de que a lâmpada foi projetada para ser ligada diretamente à rede elétrica, operando sob regime em corrente alternada (CA) e alimentando-se com picos de tensão, no melhor cenário, acima de 100 V. O *Raspberry*, por outro lado, não é capaz de oferecer valores de tensão (e mesmo de corrente) tão altos. A limitação de seus pinos vai desde a natureza digital do sinal, dispondo de somente dois níveis de tensão como saída, 0 ou 3,3 V (RASPBERRY PI, 2021b), até a própria limitação física da placa, podendo oferecer no máximo 16 mA por pino e 50 mA no total (RASPBERRY PI, 2021c).

3.3.2 Relé de Estado Sólido (SSR)

A Figura 25 mostra um módulo relé de estado sólido (SSR, do inglês *solid state relay*), também utilizado no projeto, a fim de cumprir dois objetivos dentro deste. O primeiro: permitir o acionamento de uma carga CA com nível de tensão acima dos 100 V (lâmpada de LED) a partir de um controlador *Raspbery Pi*. E o segundo: fazê-lo de forma rápida. É sabido que relés convencionais podem cumprir com o primeiro requisito mas, a bem da verdade, também é notório que tais componentes não são muito eficazes, ao menos quando se trata de operações com alta velocidade.

Figura 25 – Módulo relé de estado sólido



Fonte: Loja da Hora (2021).

Relés tradicionais realizam chaveamentos mecânicos, portanto apresentam uma natural lentidão na resposta além de um considerável menor período de vida útil. Por outro lado, relés de estado sólido conseguem superar tais obstáculos já que realizam um chaveamento a base de

transistores, ou seja, eletrônico. Nesse sentido, a aquisição de uma placa com base em SSRs para o chaveamento de cargas de maior potência, verifica-se uma solução factível.

Compatível com microprocessadores como o *Raspberry Pi*, o módulo opera em uma faixa de tensão que vai de 100 a 240 V e de corrente de carga de 0,1 a 2 A nas saídas dos canais. Já nas entradas, é alimentado com 0 (GND) e 3,3 V (VCC) como sinais de referência, além do sinal de controle que pode ser tanto 0 (nível lógico baixo), quanto 3,3 V (nível lógico alto). Ressalta-se que todos os sinais serão advindos do próprio *Raspberry Pi*. De fato, das especificações da placa, deriva-se que qualquer leitura acima de 2,5 V é reconhecida pelo módulo como nível alto, enquanto que somente leituras abaixo de 0,5 resultam em nível baixo (SAINSMART, 2021).

Além disso, conforme citado em Raspberry Pi (2021c), tais valores encontram-se dentro da faixa entregue pelo microcomputador enquanto em sua função de saída (*output*). Em números, tem-se que VOH é 3,0 V (maior que o limiar superior do módulo) e VOL vale 0,14 V (menor que o limiar inferior do módulo), onde VOH e VOL são os limiares de tensão de saída do RPi para os níveis alto e baixo, respectivamente. O Quadro 2 resume as faixas de operação tanto do módulo quanto do RPi a fim de se oferecer uma melhor compreensão.

Quadro 2 – Faixas de operação do SSR e do RPi

Grandeza	**Módulo SSR**	***Raspberry Pi***
VIH	0,5 V	-
VOH	-	3,0 V
VIL	2,5 V	-
VOL	-	0,14 V

Fonte: Produzido pelo próprio autor.

Finalmente, quanto a sua frequência de chaveamento, pode-se extrair do *datasheet* do SSR implementado no módulo, a saber um OMROM modelo G3MB-202P, que seu tempo de operação vale metade do período do sinal da fonte de alimentação da carga (acrescido de 1 ms referente ao tempo de resposta ou de subida). E assim ocorre pois o relé opera com o chamado controle de cruzamento em zero (do inglês *zero crossing control*). Na prática, isto significa que para realizar o chaveamento, o relé espera que o sinal que alimenta a carga inverta sua fase, ou seja, passe pelo 0 V. Portanto, como o ambiente de execução do projeto tem a rede elétrica operando em uma frequência de 60 Hz, o tempo mínimo para se realizar um único chaveamento

será de aproximadamente 9,33 milissegundos. Consequentemente, resulta em uma frequência máxima de chaveamento de até 107 Hz (OMROM-RUSSIA, 2017).

3.3.3 *Smartphone*

O aparelho utilizado para a recepção do sinal óptico da lâmpada de LED, no projeto, foi um *smartphone* de modelo *Samsung Galaxy* S7 (Figura 26). Sendo assim, é fundamental ressaltar que os resultados obtidos neste trabalho podem não ser válidos para outros *setups* devido a possíveis incompatibilidades entre diferentes modelos de celulares, sendo este um parâmetro de grande relevância no estudo dada sua alta variabilidade. Em Samsung (2017) encontram-se as principais características do aparelho, das quais serão citadas, a seguir, aquelas mais relevantes para o presente estudo:

- Sistema Operacional *Android*;
- Sensor de Luz Ambiente (frontal);
- Processador *Octa Core* 64-bit;
- Memória RAM de 4GB;
- Memória Interna de 32GB.

Figura 26 – *Smartphone*

Fonte: Wikipedia (2021).

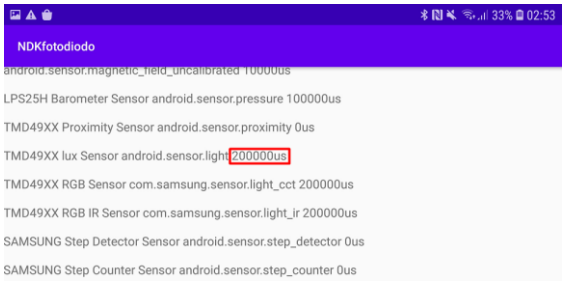
Como visto anteriormente neste documento, o componente responsável pela detecção do brilho da lâmpada é o sensor de luminosidade ambiente (ALS, do inglês *Ambient Light Sensing*)

localizado na parte frontal do aparelho celular, como pode ser observado na Figura 27. A partir do resultado em tela de um código-teste, foi possível descobrir o modelo do sensor (AMS TMD49XX), bem como sua taxa de atualização padrão: 200000 microssegundos, o equivalente a 5 Hz (Figura 28). Logo, é fácil perceber que o *smartphone* será o gargalo do sistema em se tratando de velocidade da transmissão.

Figura 27 – Sensor de luminosidade ambiente

Fonte: Samsung (2021).

Figura 28 – Taxa de amostragem padrão do ALS



Fonte: Produzido pelo próprio autor.

É importante pontuar que a tecnologia VLC ainda não se encontra estabilizada no mercado, como já argumentado neste texto, e, à medida em que tal condição for mudando, é presumível que os *smartphones* também se adaptarão para integrá-la. Um dado animador é que o próprio sensor ALS do aparelho utilizado no projeto já é capaz de comunicar-se, com o dispositivo, com taxas de até 400 kbps segundo as suas especificações técnicas (AMS, 2015). Sendo assim, controlá-lo por meio de programação, mesmo para velocidades mais baixas devido à outras limitações, faz-se necessário como um primeiro passo neste processo.

4 PREPARAÇÃO DOS CÓDIGOS UTILIZADOS NOS SISTEMAS

Nesta seção foram brevemente discutidos todos os processos referentes à programação dos códigos utilizados no projeto. A fim de uma maior organização, o tema foi divido em quatro tópicos.. A seção 4.1 trata dos arquivos HL7 desde quando são gerados até chegarem ao RPi de transmissão, discutindo-se como foram realizadas suas leituras e armazenamentos. Já a seção 4.2 introduz as distintas formas de apresentação do arquivo a ser transmitido. Na lógica implementada, os responsáveis por tais alterações da informação foram os métodos aqui classificados como protocolos. A seção 4.3 identifica as distintas técnicas escolhidas para se transmitir o sinal por meio de três diferentes bibliotecas disponíveis em linguagem *Python*. E finalmente a seção 4.4 trazendo as lógicas implementadas em plataforma *Android*.

4.1 Arquivos HL7

4.1.1 Geração e Envio

Os arquivos HL7 utilizados como testes no projeto foram, em sua maioria, editados de outros originalmente capturados pelo monitor multiparamétrico ao LabTel emprestado. As mudanças foram apenas alterações nos valores medidos pelos sensores, com o intuito de se demonstrar variação dos mesmos para os casos de execução com mais de uma lida. Os próprios arquivos originais (armazenados) também foram exitosamente testados.

Quanto ao envio, foi concebido, do lado do computador, um código em *Python* baseado na biblioteca *Paramiko*, encontrada em Paramiko (2021), cujo principal objetivo seria buscar uma comunicação SFTP com o RPi transmissor. Para isso, ele deveria realizar as seguintes tarefas:

- Buscar por um dispositivo específico (*Raspberry*) dado um endereço IP indicado;
- Estabelecer uma conexão segura entre os dispositivos;
- Buscar por todos os arquivos com extensão HL7 presentes no diretório local indicado;
- Transmitir um arquivo por vez, a cada intervalo de tempo pré-definido, para um diretório remoto indicado;
- Encerrar a sessão ao fim da última transferência.

4.1.2 Leitura e Atualização

Já do lado do RPi de transmissão, implementou-se um laço (*loop*) cujo objetivo inicial seria escanear uma pasta específica em seu SO, buscando por novos arquivos (terminados com extensão HL7) no local. Para isto ser possível, se fez necessário armazenar em uma determinada variável, o caminho do último arquivo encontrado. Sempre que o mais recente (*current_file*) tiver o mesmo caminho daquele previamente salvo (*old_file*), o algoritmo deve escrever em tela que não houve arquivo encontrado. Exceto pela primeira lida na qual ainda não se tem armazenado o arquivo antigo, o laço continua até que algum outro dado apareça na pasta.

No exato momento em que o HL7 chega ao diretório, a leitura é, de fato, iniciada. Em seguida, o arquivo é lido linha por linha e tem seus parâmetros salvos em listas. Ao final do processo de leitura, ocorre a codificação da informação que será modulada e, então, transmitida. Eventualmente, um novo processo, agora de escrita em um arquivo de texto, também pode ocorrer. Em seguida, incrementa-se a variável que indica o número da medição. Por fim, o *loop* volta ao seu início e assim continua o ciclo até que o usuário encerre o programa de forma abrupta ou então tenha passado um parâmetro de número limite de leituras quando antes da execução do programa.

4.1.3 Escrita e Armazenamento

O conceito de escrita, no contexto do projeto, deve ser entendido como a própria transmissão. Nesse sentido, transmitir o sinal seria, repetidamente, escrever, ou atribuir, um nível alto ou baixo a um determinado pino do barramento de saída do RPi (GPIO), por um determinado espaço de tempo. Logo, o processo de escrita resume-se às escolhas do protocolo (codificação a nível de bits) e da biblioteca de comunicação direta com o pino de saída GPIO, ambas representadas no programa em forma de funções. Cada uma dessas funções será melhor detalhada nas próximas seções deste documento.

O armazenamento, por sua vez, refere-se à escrita da informação HL7 em arquivos de texto para fins de registro e eventuais análises comparativas entre recepção e transmissão. Logicamente, a opção de gravação foi adicionada em ambos os microcomputadores *Raspberry Pi*. Por fim, destaca-se que foi implementado que o arquivo HL7 pode ser tanto integralmente

transmitido, quanto ter suas informações resumidas a linhas de um arquivo CSV para otimização da armazenagem e também do próprio monitoramento.

4.1.4 Fluxograma

É importante destacar que, até o momento, todos os códigos descritos nesta seção encontram-se ou no *notebook* (medidor multiparamétrico) ou no RPi de transmissão. Entretanto, também foi necessário implementar códigos no lado do dispositivo *Raspberry* de recepção, tanto para testes quanto para demonstração da comunicação sob a configuração *Back-to-Back*.

Destaca-se que, para fins de organização, os códigos em ambos os lados, de transmissão e recepção, foram divididos em arquivos “principais” e “auxiliares”. O primeiro basicamente consiste em definições dos parâmetros de funcionamento do código, por parte do usuário, bem como em estabelecer a função raíz do programa, responsável por realizar as chamadas de todas as outras funções necessárias ao código. Já o segundo, consiste em uma coletânea destas demais funções. Além disso, não pode-se deixar de mencionar que também foram elaborados códigos de testes para agilizar a coleta dos dados necessários para a geração das curvas de resultados. Entretanto, por motivo de concisão, tais códigos não terão seus pormenores detalhados neste texto.

Finalmente, no Apêndice A encontram-se reunidos, de forma simplificada, todos os procedimentos computacionais implementados para o estabelecimento do sistema de comunicação proposto no projeto. Para tanto, conta-se com o auxílio de um fluxograma, representado na imagem do apêndice citado, a fim de se oferecer uma melhor compreensão de tudo o que foi implementado.

4.2 Protocolos Sugeridos para as Aplicações

Nesta seção, o termo protocolo deve ser associado à formatação dos pequenos pacotes de informação, para os quais os arquivos HL7 deverão ser transformados quando do momento da comunicação, de fato. Basicamente, são as formas de se empacotar a mensagem de acordo com o nível de detalhamento desejado por ambas as partes da “conversa”. Nesse sentido, criou-se três formas de representação para este tipo de arquivo, além de, claro, sua própria formatação

em caráter integral. É nesta parte do documento que serão discutidos temas como os porquês da inclusão (ou ausência) de: cabeçalhos, rodapés, *bits* de análise de paridade, dentre outras técnicas de representação dos dados.

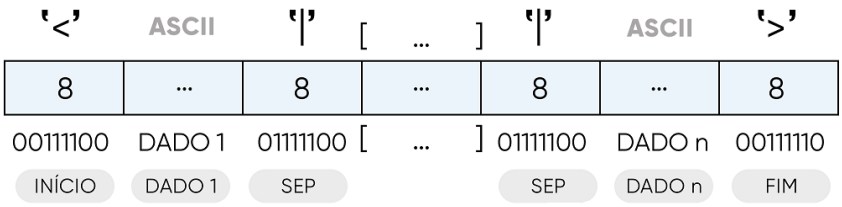
4.2.1 LabTel-I

O protocolo denominado “LabTel-I” foi proposto como uma forma de otimizar a transmissão, principalmente levando-se em consideração que muitos dos dados associados ao HL7 carregam pouca informação ao longo do tempo. Não somente símbolos como os caracteres de separação dos campos de texto mas também dados como os códigos inerentes ao protocolo, nomes dos sensores com suas respectivas unidades, dentre outros, não são factíveis de que possam variar frequentemente. Dessa maneira, *bits* que poderiam ser economizados ou então reaproveitados de uma melhor forma como, por exemplo, em análises de redundância, acabam sendo desperdiçados. Além disso, tendo em mente que os celulares ainda podem não se encontrar preparados para receberem o sinal óptico em taxas mais elevadas, reduzir o número de bits a se transmitir pode tornar-se um fator de grande relevância.

A partir da Figura 29, pode-se observar que o protocolo sugere:

- Cabeçalho com apenas um caractere ASCII, a saber, de número 60 (“<”) e 8 bits;
- Separador de leituras com o caractere ASCII de número 124 (“|”) e 8 bits;
- Rodapé com apenas um caractere ASCII, a saber, de número 62 (“>”) e 8 bits;
- Seleção de informações mais relevantes ao processo de monitorização, como, por exemplo, o nome do paciente e os valores medidos.

Figura 29 – Estrutura de uma mensagem com o protocolo LabTel-I



Fonte: Produzido pelo próprio autor.

4.2.2 LabTel-II

A fim de satisfazer demandas dentro do grupo de pesquisa do qual o presente projeto faz parte, foi sugerida a implementação de mais um protocolo, desta vez com tamanho fixado em 201 *bits*. Com nome proposto de LabTel-II, este novo protocolo teve como objetivos a otimização da representação numérica transmitida, além de padronizar o tamanho das mensagens, restringindo, ainda mais, o conjunto de informações repassadas.

Destaca-se que, em nome da padronização, todos os valores numéricos agora passam a ser representados por pontos flutuantes de precisão simples, com 32 bits cada. Além disso, também ressalta-se que somente as medições dos parâmetros vitais passarão a ser tratadas pelo protocolo. Todos os demais campos acabam por serem descartados com a nova representação a fim de um melhor aproveitamento de espaço. Ainda, como observado na Figura 30, percebe-se que foram retirados os caracteres de separação dos parâmetros, visto que estes agora possuem tamanho conhecido. Por outro lado, um cabeçalho de quatro bits ‘um’ e um rodapé de quatro bits ‘zero’ foram adicionados a fim de se indicar, respectivamente, o início e o término do pacote. Concluindo o protocolo, resolveu-se por adicionar um bit de paridade par antes do rodapé como forma de se facilitar possíveis análises de erros.

Figura 30 – Estrutura de uma mensagem com o protocolo LabTel-II

4	32	32	32	32	32	32	1	4
1111	DADO 1	DADO 2	DADO 3	DADO 4	DADO 5	DADO 6	0/1	0000
INÍCIO	DADOS						P	FIM

Fonte: Produzido pelo próprio autor.

Finalmente, uma recomendação plausível quando de sua utilização poderia ser a aplicação conjunta deste protocolo com o anterior. Transmitir o protocolo LabTel-I em uma primeira medição do dia, ou do turno médico, enquanto que todas as demais por meio do LabTel-II, pode ser uma prática aconselhável do ponto de vista do médico ou enfermeiro. Dessa maneira, os profissionais de saúde poderiam manter-se relativamente atualizados com as informações não-monitoráveis, como o nome do paciente internado, ao mesmo tempo em que otimiza-se o tamanho da informação transmitida.

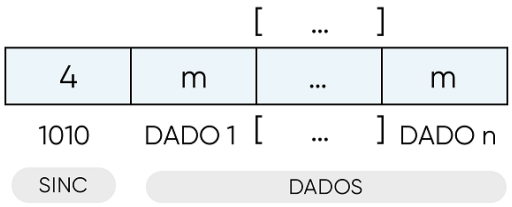
4.2.3 LabTel-III

O último protocolo proposto neste documento é tratado pelo termo “LabTel-III”. A motivação para se adotar mais uma forma de representação de mensagens HL7, reside na limitação do equipamento que mais à frente se mostrará o gargalo do projeto: o aparelho *smartphone*.

É sabido que o sensor de luminosidade ambiente do celular utilizado pelo LabTel consegue comunicar-se com o núcleo de processamento do aparelho com conexão I²C e taxas na casa das centenas de kilohertz. Entretanto, devido ao próprio funcionamento do sistema operacional do equipamento (baseado em eventos), bem como a limitada capacidade de memória para os múltiplos processamentos em tempo real requisitados pelos experimentos, tais taxas não demonstram-se factíveis de serem alcançadas em uma primeira implementação do projeto. Dessa forma, a fim de se garantir a recepção, buscou-se enviar a menor quantidade de bits possível para representar as medições e, para tanto, chegou-se à conclusão de que converter números inteiros (base decimal) diretamente para binários, seria a melhor solução para o problema.

Nesse sentido, para diminuir ainda mais o tamanho dos pacotes, resolveu-se continuar com a ideia de número fixo de bits por parâmetro, porém deixado livre (dentro do código) para o responsável pela comunicação escolher tal quantidade. Quanto ao novo cabeçalho, foi adotado a sequência binária ‘1010’ por motivos de sincronização. O aplicativo de celular deverá ser capaz de utilizar estes quatro primeiros bits para determinar a taxa de transmissão sob a qual os pacotes deverão ser enviados. Finalmente, ressalta-se que a Figura 31 ilustra todas as características comentadas.

Figura 31 – Estrutura de uma mensagem com o protocolo LabTel-III



Fonte: Produzido pelo próprio autor.

Logicamente, há inúmeras outras possibilidades para a otimização da transmissão quando se trata do número mínimo necessário de bits para a comunicação. Todas com suas respectivas vantagens e desvantagens. Porém esta solução (LabTel-III) foi considerada satisfatória dentro do contexto experimental do projeto, sendo a representação escolhida para os testes com o dispositivo *smartphone* atuando como receptor.

4.3 Bibliotecas

De forma semelhante ao item anterior, nesta seção utiliza-se o termo "biblioteca" geralmente empregando-o como um "método de transmissão". Como já introduzido no início da seção, foram implementados três diferentes métodos para se transmitir o sinal a partir dos pinos de saída do RPi. Cada um deles refere-se a uma biblioteca específica disponível em linguagem de programação *Python*. Elas são relevantes a ponto de definirem métodos próprios de transmissão pelo importante fato de que são estas bibliotecas as responsáveis pela forma de acesso, bem como do controle de escrita ou leitura das portas GPIO, a partir do processador. Na prática, isto significa que algoritmos que utilizam bibliotecas diferentes, tendem a apresentar resultados também diferentes, ainda que escritos de maneira mais similar quanto o possível.

4.3.1 *Serial* (UART)

A primeira implementação de um algoritmo que fosse capaz de interpretar valores binários e convertê-los para níveis alto e baixo nas saídas das portas GPIO de um microcomputador *Raspberry Pi*, se deu por meio do padrão de comunicação conhecido por UART. No RPi, tal protocolo é implementado através do uso da biblioteca *Serial*, definida em sua documentação como um módulo responsável por encapsular o acesso às portas de mesmo nome, paralelamente ao gerenciamento da comunicação destas com uma interface sob a qual se rodam códigos em Python (PYSERIAL, 2017). Nela, o autor pôde configurar parâmetros como o tempo de espera de abertura da porta, taxa de transmissão e o formato dos pacotes a se enviar.

De seu próprio nome, como já discutido em seções anteriores, observa-se que o padrão trata de comunicação em caráter assíncrono. Nesse sentido, sua implementação no projeto se faz oportuna, visto que sistemas VLC, conceitualmente, não são projetados para utilizarem de canais com fios. E, de fato, tal característica destacou o UART em meio às outras opções

oferecidas pelo *Raspberry Pi*, tais como o SPI ou o I²C (ambos síncronos). Outro fator de enorme relevância para o estabelecimento do padrão como a primeira escolha para o projeto se deu por conta de sua capacidade de comunicar-se com altas taxas de transmissão, a depender da limitação física do equipamento em que se está ligando (PYSERIAL, 2017).

Um ponto negativo, porém, encontra-se na particularidade do padrão em quebrar a transmissão em diversos pacotes menores, com no máximo 1 *byte* de dados cada. Tal necessidade resulta em um encapsulamento do protocolo escolhido na seção 4.2 para dentro de um outro próprio estabelecido pelo UART. Sendo assim, a cada byte de dados adiciona-se um bit 0 de cabeçalho e um bit 1 de rodapé, trazendo o sinal de volta ao nível alto. Dessa maneira, o sistema perde eficiência pois reduz em 20% (8 bits úteis de 10 totais) a quantidade útil de dados enviados.

Quanto ao seu funcionamento nos códigos do trabalho, o lado transmissor simplesmente interpreta a mensagem diretamente em ASCII e automaticamente a converte para *bytes* de "zeros e uns". Em seguida, a o *byte* é espelhado, de forma a se transmitir primeiro o *bit* mais significativo, e então é concatenado com os *bits* de sinalização de início e fim da transmissão. Finalmente, o pino de transmissão serial do RPi é configurado para o nível alto sempre que se observa o *bit* com valor igual a "1", ou então para o nível baixo sempre que se tem o *bit* com valor "0" na fila do *buffer* reservado para a comunicação serial.

Já no lado da recepção, o código basicamente faz o pino de recepção serial iniciar em nível lógico alto, para em seguida escutar, em um *loop* contínuo, por bordas de descida (*bit* 0) que indicariam o início da transmissão. Quando detectada esta condição, o programa lê o arquivo de linha em linha, ou seja, só termina a leitura quando encontra o caractere especial de final de linha. Assim, à medida em que as medições vão sendo recuperadas, seus valores são mostrados em tela, diretamente em formato ASCII (UTF-8) e em tempo real, além serem armazenados em arquivos CSV para eventuais consultas ao final do turno médico.

4.3.2 RPi.GPIO

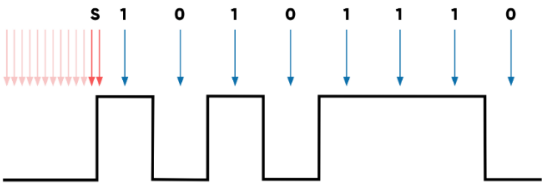
Em seguida, em conformidade com o a proposta do protocolo LabTel-II, foi sugerido ao autor a implementação de um método de transmissão livre de qualquer encapsulamento, indesejado, de sua informação. De fato, o protocolo de tamanho fixo baseado em ponto flutuante exigia

cabeçalho e rodapé bem definidos e diferentes dos quais a comunicação serial UART poderia oferecer. Além dos bits de referência do nível do sinal, ainda persistia o problema de quebra do pacote em outros, menores, de oito bits cada. Nesse sentido, foi decidido dar um passo atrás na busca por um sistema otimizado em suas taxas de transmissão a fim de solucionar o problema. Portanto, escolheu-se a biblioteca-padrão de controle dos pinos do *Raspberry Pi*, de nome RPi.GPIO, a fim de se alcançar a liberdade desejada na estrutura da mensagem.

Um grande problema, entretanto, foram os indícios de que a biblioteca não seria, de fato, capaz de transmitir dados em taxas consideráveis. Em verdade, de sua própria documentação pode-se obter que a aplicação não havia sido projetada para operar sob regimes em tempo real, ou com características temporais em níveis mais críticos, devido ao caráter "multitarefas" do sistema operacional (SO) do microcomputador, sistema este baseado em *Linux*. Dessa forma, o programa torna-se instável, sendo impossível de se conseguir prever quando o *Python* poderá estar ocupado com outras tarefas ou mesmo quando o próprio SO estará dando prioridade de acesso ao processador (do inglês CPU, *Central Process Unit*) a outros processos mais fundamentais à rotina do sistema (PYPI, 2019).

Quanto a sua implementação, por outro lado, o novo código requereu uma complexidade maior que o anterior visto que algumas funções, antes implícitas com a biblioteca Serial, tiveram que ser pensadas e escritas manualmente pelo autor deste texto. Exemplos como adicionar os bits de cabeçalho, paridade e rodapé, bem como a conversão dos dados de caracteres ASCII para binários (e vice-versa) e a busca pela borda de subida no início das leituras. Em relação ao seu funcionamento, pode-se observar na Figura 32 que as leituras (em azul) só iniciam, de fato, após a detecção da borda de subida (em vermelho). Quando isto ocorre, começa-se a contagem dos tempos de *bit*, definidos no início do programa como o inverso da frequência escolhida, para cada nova leitura. Exceto somente pela primeira, na qual se lê com um atraso de meio tempo de bit, acrescentado logo após a resposta com a primeira transição.

Figura 32 – Leitura para o RPi.GPIO



Fonte: Produzido pelo próprio autor.

4.3.3 *Pigpio*

O último método de gerenciamento dos pinos GPIO do microcomputador *Raspberry Pi*, utilizado tanto para escrita quanto para leitura dos mesmos, se deu com o auxílio da biblioteca conhecida como *Pigpio*, encontrada am Abyz (2001). Esta ferramenta é bastante difundida nos fóruns especializados e tem como principal vantagem o fato de que ter sido escrita em linguagem de programação C, sendo a sua implementação em *Python* apenas um módulo no qual a biblioteca original é chamada. Dessa forma, a aplicação permite ao programa atingir velocidades muito mais elevadas visto que esta é uma das características de programas, compilados, escritos em linguagem procedural e estruturada tal qual o C. Ao estudar a documentação da biblioteca, confirma-se que ela é, de fato, capaz de oferecer taxas elevadas. Nesta, indica-se a possibilidade de se ler e escrever nos pinos GPIO com precisão na escala dos microssegundos (ABYZ, 2021).

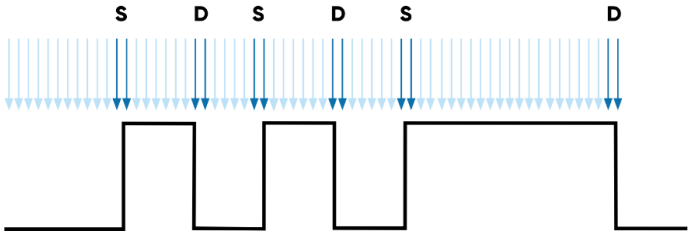
Quanto à programação dos códigos de transmissão e recepção em si, utilizou-se dos programas já elaborados previamente como base para o novo desenvolvimento, ainda que mudanças tenham sido realizadas. Do lado da transmissão, a rotina de conversão em *bits* se mantém, porém a de envio se dá de uma forma diferente. Em vez de, manualmente, ter que configurar um nível alto ao se encaminhar um *bit* 1 (e nível baixo para um *bit* 0), utilizou-se métodos já implementados pela biblioteca. Basicamente, a cada *bit* um cria-se um pulso (objeto da classe *pulse)*, no qual se define as características do sinal como, por exemplo, o tempo de *bit*. Em seguida, instancia-se um objeto da classe *wave*, que adiciona todos os pulsos, relacionando-os a instantes específicos no tempo. Finalmente a “onda” é enviada com a função *wave_send_once()* e então deletada com *wave_delete()* ao término do envio.

É importante ressaltar que este métodos permitem uma transmissão mais precisa, principalmente pelo fato de permitirem a escrita no pino do GPIO com acesso direto à memória (DMA, do inglês *Direct Memory Access*). Tal método de leitura ou escrita é consideravelmente mais rápido do que outras formas de registro por não precisar passar pela supervisão do processador ao realizar suas rotinas. Nesse sentido, pode-se considerar esta uma vantagem especialmente importante no caso de um *Raspberry Pi*, visto que, como já comentado anteriormente, o SO do RPi tende a interferir negativamente no funcionamento de aplicações com requisições em tempo real.

Já o lado da recepção foi onde as maiores mudanças aconteceram. A principal delas sendo a forma de se ler os bits. Por recomendação da própria documentação da biblioteca, foi necessário realizar as leituras em função das transições (sempre que existirem) e não mais confiar unicamente no tempo de *bit*. Para implementar a solução, foram utilizados outros métodos nativos do *Pigpio* cujos funcionamentos baseiam-se nas funções conhecidas como “funções de *callback* assíncrono" (ABYZ, 2021). No *Pigpio*, a finalidade deste tipo de função se dá para atingir-se uma maior precisão quando do momento das leituras, como já mencionado anteriormente. Isto é possível pelo fato das detecções de transições serem reportadas de forma muito mais rápida do que a função de leitura de estado atual, padrão da biblioteca, poderia oferecer.

A Figura 33 demonstra que todo o processo passa a funcionar em função da detecção de transições, ou seja, subidas (S) e descidas (D). Basicamente inicia-se uma contagem de tempo a partir da primeira subida, terminando a leitura somente após a observação do tempo necessário para o número de bits, esperados e referenciados no início do programa, terem sidos todos recebidos. Por fim, sabe-se quantos bits mantiveram-se em um determinado nível ao se comparar os tempos de transição com o tempo de bit referenciado.

Figura 33 – Princípio de funcionamento da leitura para o *Pigpio*



Fonte: Produzido pelo próprio autor.

4.4 Aplicativos *Android*

Até o momento, foi discutido neste texto a programação necessária para se implementar o sistema base (*Back-to-Back*) daquele desejado ao final do projeto, ou seja, o PLC-VLC. A segunda etapa, de inclusão do PLC, não necessita de códigos adicionais, bastando incorporar os novos dispositivos no circuito para o seu funcionamento. Já em se tratando do VLC, completando a solução proposta neste projeto, observa-se que houve a necessidade da

implementação de um código adicional: a criação de um aplicativo de celular capaz de capturar e decodificar os dados recebidos, pelo aparelho, em meio ao brilho da lâmpada. Na verdade, houve ainda a necessidade de um segundo aplicativo para a realização dos testes de caracterização do sistema sensor-lâmpada. Assim, esta seção busca apresentar e explicar o funcionamento de ambos os códigos.

4.4.1 Aplicativo 1

Para a elaboração do primeiro aplicativo (*app*), cujo objetivo seria receber e decodificar o sinal, instalou-se no computador do Autor deste texto, um *software* conhecido como *Andoid Studio*, de versão número 4.1.3. Sua função dentro do projeto é prover o ambiente de programação, além das demais ferramentas de testes, necessários para a criação de aplicativos para *smartphones* trabalhando com sistema operacional *Android*. Como é este o SO rodando no aparelho disponibilizado pelo LabTel (um *Samsung* S7, já apresentado), o *software* instalado foi a escolha direta para auxiliar esta nova etapa de programação.

Fundamentalmente, o *app* pode ser dividido em dois códigos: um em formato XML, responsável pela aparência, ou *layout* do programa, e o outro em linguagem de programação Java, onde a lógica do código deve ser, em si, programada. No arquivo XML, basicamente adicionou-se campos de texto, sob os quais devem ser associados os títulos e os valores das variáveis mais representativas para o âmbito do projeto (na seção de resultados serão disponibilizadas capturas de tela demonstrando a aparência final do aplicativo). Já no arquivo com extensão '.java', encontra-se toda a estrutura de busca contínua por alterações nas medições de um determinado sensor do aparelho celular, aqui escolhido o de luminosidade ambiente. Ressaltam-se que o SO funciona baseados em atualizações de eventos, semelhante às funções de busca por transições de níveis lógicos utilizadas em códigos anteriores.

Sempre que o sensor notifica o sistema sobre a presença de um novo "evento de luz", a iluminância medida deve ser armazenada em forma de uma variável e assim se repete até que o vetor de medições seja considerado cheio, acionando uma indicação de fim de leitura (*flag_decode*). Somente quando este momento é chegado é que finalmente ocorre a decodificação do sinal recebido. Neste processo, ressalta-se que o algoritmo de decodificação foi totalmente baseado no protocolo definido (neste trabalho) como LabTel-III. Assim, espera-

se encontrar um cabeçalho de quatro bits (‘1010’), um rodapé com dois (‘10’), além de dados binários que possam ser interpretados como números em base decimal.

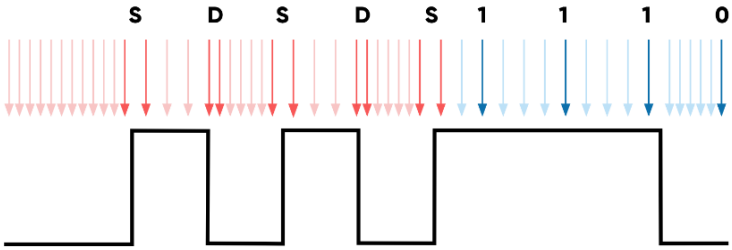
O programa ainda deve ser capaz de interpretar o nível sob o qual se mantém o sinal, por conta própria. Isto ocorre pois o sensor ALS do *smartphone* retorna valores multiníveis advindos de uma leitura originalmente analógica. Nesse sentido, estabelece-se o conceito de *threshold*, que nada mais é do que um limiar de marcação lógica (zero ou um) para a medição da iluminância, recebida em lux. Inicialmente, o *threshold* é declarado com um valor bastante elevado (100000 lux), de forma a se evitar que pequenas movimentações no aparelho pudessem causar um acionamentos inesperados. Em seguida, o limiar é atualizado a cada dez eventos (amostras) capturados. O novo valor é calculado por meio da equação (2).

$$Th = \frac{(1+s)}{10}\sum_{i=1}^{10} E_i \tag{2}$$

Onde:
Th é o valor de *threshold* atualizado (lux);
s é a sensibilidade da recepção;
i é o índice do cálculo da média;
E_i é o valor de iluminância medido (lux).

Como todo programa baseado em eventos, o aplicativo passa a rodar em *loop* assim que inicializado, capturando as medições do sensor de iluminação (e atualizando o limiar de marcação), porém sempre buscando pelo nível alto, indicando o início da transmissão. Quando este finalmente chega, ocorre um processo definido como de ‘sincronização’. Nele, os quatro primeiros *bits* recebidos (cabeçalho) têm as suas medições salvas em uma variável em separado, a fim de se buscar solucionar um problema de amostragem distinta entre ‘zeros’ e ‘uns’, observado em testes experimentais. Este problema ocorre no momento da amostragem do sinal, quando esta última aparenta ser realizada de forma mais rápida, ou menos espaçada no tempo, para leituras em níveis de iluminação mais baixos. Esta possibilidade encontra-se representada na ilustração Figura 34.

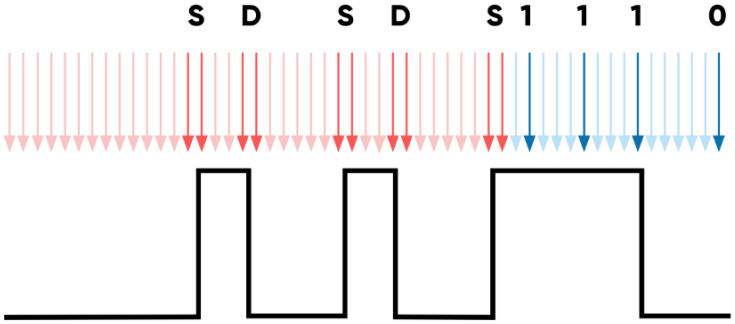
Figura 34 – Princípio de funcionamento da leitura para o Aplicativo 1 (possibilidade 1)



Fonte: Produzido pelo próprio autor.

Já a Figura 35 levanta a possibilidade do problema se encontrar no próprio sinal de saída do LED. Como se pode observar na ilustração, dessa vez o tempo de bit em nível alto é estreitado.

Figura 35 – Princípio de funcionamento da leitura para o Aplicativo 2 (possibilidade 2)



Fonte: Produzido pelo próprio autor.

Retornando à etapa de sincronização do sinal, a estratégia implementada simplesmente calcula quantas amostras são colhidas enquanto se envia o primeiro *bit* um, e quantas são colhidas durante o envio do segundo *bit* transmitido, necessariamente um *bit* zero. Em seguida o mesmo procedimento se repete para os dois próximos bits (novamente um e zero), a fim de se confirmar os números recém-recebidos. Dessa forma, o código é capaz de estimar por si próprio a taxa aproximada de envio, e assim readequar os instantes designados para leitura ao considerar que as amostragens de *bits* zero e um mantém-se inalteradas ao longo da comunicação. Caso isto não ocorra, definiu-se que fosse declarada uma mensagem que indicaria a falta de sincronia do sinal e o pacote com a medição deveria então ser descartado.

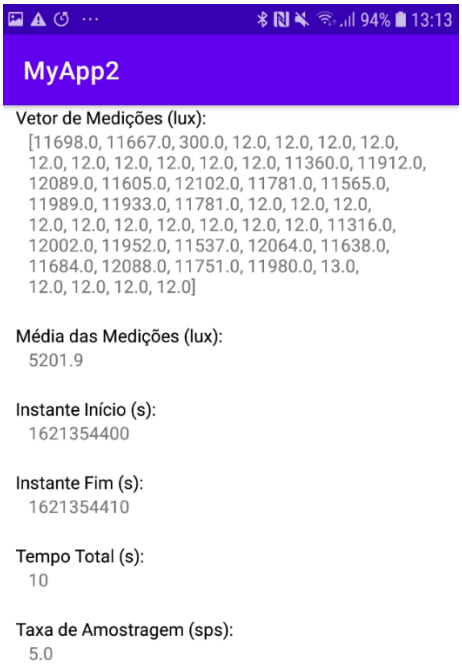
Quanto à leitura dos bits, esta ocorre de forma semelhante ao método da seção 4.3.3. Entretanto, o símbolo (zero ou um) passa a ser representado como um conjunto, ou intervalo, de amostras

classificadas como de um nível igual, em vez de basear-se no tempo em que este é mantido. Por fim, observa-se que para se evitar uma elongação excessiva do texto, os demais detalhes do código (como o momento exato da leitura ou os critérios de parada do algoritmo) podem ser encontrados sob a forma de um fluxograma disponibilizado no Apêndice B.

4.4.2 Aplicativo 2

O último código implementado no projeto pode ser considerado bem mais simples. Ele aproveita a estrutura de leitura do sensor de luminosidade do primeiro aplicativo e basicamente trabalha com o armazenamento destes valores, realizando adicionalmente alguns cálculos sobre eles como, por exemplo, a média aritmética da iluminância. Na Figura 36 pode-se observar quais métricas são oferecidas a fim de se avaliar a recepção do sinal luminoso pelo aparelho smartphone.

Figura 36 – Captura de tela do Aplicativo 2



Fonte: Produzido pelo próprio autor.

Novamente, disponibiliza-se um fluxograma com o funcionamento do código, o qual também pode ser observado no Apêndice B. Uma discussão mais detalhada sobre as métricas do aplicativo e a necessidade de seu uso também será retomada na seção de resultados.

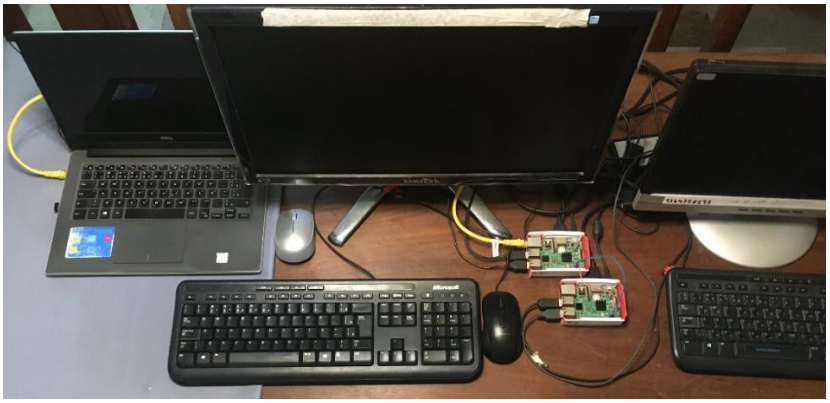
5 RESULTADOS E DISCUSSÕES

Como última etapa do projeto, esta seção trata de todos os resultados obtidos com as práticas experimentais do trabalho. Ainda, discute as origens de possíveis fontes de erros, além de buscar oferecer soluções plausíveis e realísticas, dentro do escopo do projeto, para cada uma delas. A fim de uma melhor organização, dividiu-se a seção em três partes, uma para cada configuração do sistema. A primeira, trata dos resultados obtidos com a recepção em um segundo *Raspberry*. A segunda, foca na caracterização dos equipamentos, recém-adicionados, responsáveis pelo PLC. Finalmente, a terceira trata da caracterização do par *smartphone*-lâmpada de LED, entrando em substituição ao RPi de recepção e, desse modo, concluindo as instalações com a inserção do trecho VLC no sistema.

5.1 Configuração 1 (B2B)

A primeira etapa de testes teve como objetivo demonstrar as capacidades e limitações de cada uma das três diferentes formas de transmissão e recepção, no RPi, implementadas. Nesse sentido, para a realização dos testes com a biblioteca Serial (UART), ligou-se, com um fio azul comum (*jumper*), o pino GPIO14 (08, saída de TX) ao GPIO15 (10, entrada de RX). Em seguida, foi necessário referenciar ambos os microcomputadores com um mesmo nível de 0 V, conectando-os a partir de seus pinos *ground* (06) com o fio verde. Já para os testes com as demais bibliotecas (RPi.GPIO e *Pigpio*), ligou-se os pinos GPIO03 (05) de ambos os dispositivos por meio de um fio laranja. A Figura 37 ilustra a configuração B2B ao passo que a Figura 38 representa as conexões.

Figura 37 – Configuração 1 (B2B)



Fonte: Produzido pelo próprio autor.

Figura 38 – Conexão *back-to-back* entre os RPi

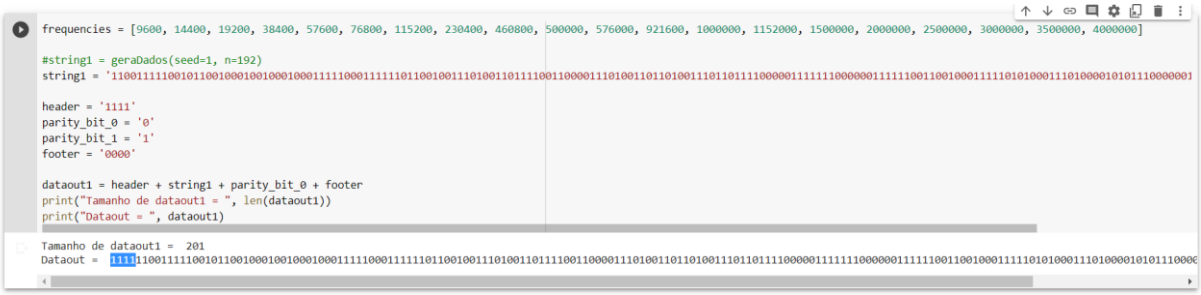
Fonte: Produzido pelo próprio autor.

5.1.1 Serial UART

Para se testar o primeiro método de transmissão, com a biblioteca Serial, definiu-se tanto no código de testes, do lado transmissor, quanto no código de recepção, uma variável de referência pseudoaleatória (string1). Esta variável do tipo *string* e com 192 caracteres, divididos entre '0's e '1's, foi gerada com o auxílio da classe "*random*" da biblioteca *NumPy*, disponível em *Python*. O objetivo de se usar dados pseudoaleatórios para os testes foi de garantir a imparcialidade destes. Porém, para o caso de uma possível necessidade de confirmação futura destes dados, foi também utilizada a função *random.seed*(1), da mesma classe. Assim, ainda se garantiu a igualdade de distribuição de valores 0 e 1, com metade do tamanho total (96) para cada.

Após a geração pseudoaleatória dos dados, o próximo passo foi encapsulá-los dentro do protocolo LabTel-II, com cabeçalho e rodapé de 4 caracteres cada, além de mais um representando o *bit* de paridade. Foi escolhido este protocolo para exemplificar a utilização no contexto clínico proposto, ainda que não existisse tal necessidade. Desse modo, cada pacote de teste (dataout1) ficou com 201 caracteres ASCII (UTF-8) cada, como se pode observar através da Figura 39, capturada da plataforma *online* utilizada para a geração dos resultados gráficos deste trabalho (*Google Colab*).

Figura 39 – Pacote de testes pseudoaleatório



Fonte: Produzido pelo próprio autor.

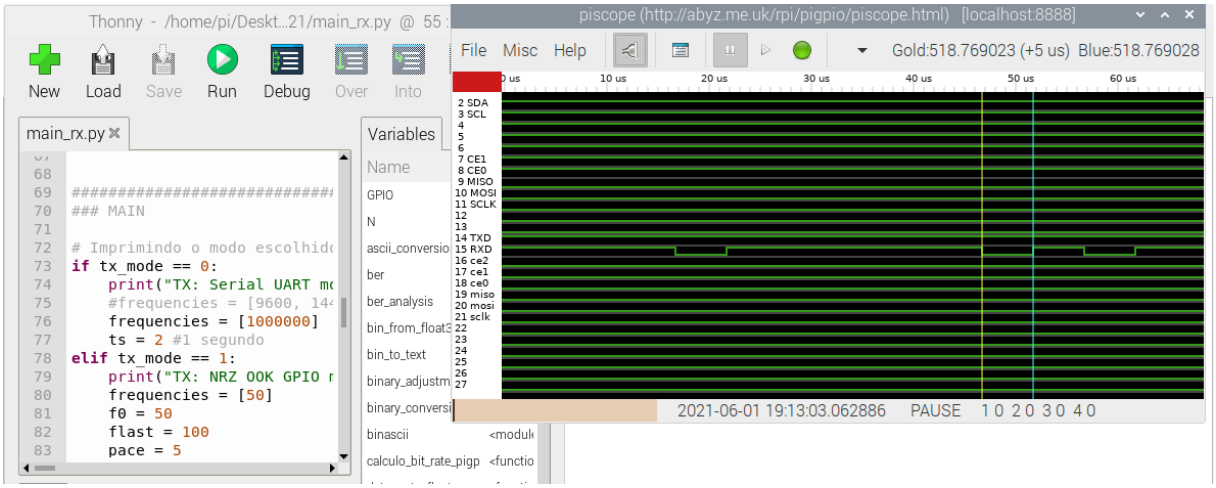
Já em se tratando do teste em si, foram enviados dois pacotes dataout1, de um RPi ao outro, a fim de se verificar a taxa de erros de bit (BER) acusada pela recepção ao final do processo. Foi escolhido enviar dois pacotes em razão de se buscar averiguar o resultado do método em análise para uma escala de referência de 1×10^{-3} de BER, nível tido como desejável em sistemas de comunicação. Para este e outros testes, definiu-se por enviar ao menos três vezes o inverso do BER desejado, a fim de se garantir um nível de confiança mais alto para os resultados das medições. Ou seja, para a escala de referência desejada, deve-se enviar ao menos 3000 bits.

Deve-se aqui ressaltar que a biblioteca Serial utiliza por padrão de comunicação o protocolo UART, que fundamentalmente converte cada caractere ASCII (UTF-8) para sua representação binária (1 *byte* ou 8 bits), acrescentando ainda os bits de demarcação de início e final da transmissão. Ou seja, cada caractere na realidade termina por transmitir 10 bits cada. Sendo assim, o teste envia um total de 4020 bits, quantidade maior do que os 3000 inicialmente desejados.

Quanto aos envios, é possível observar, ainda na Figura 41, que foram escolhidas 20 diferentes taxas de transmissão ou frequências. Ou seja, enviou-se o primeiro pacote para cada uma das taxas selecionadas e em seguida repetiu-se o processo para o segundo pacote, representando um segundo conjunto de medições. O resultado obtido foi de certa forma surpreendente pois todos os bits enviados foram recebidos pela RPi (RX) sem que houvesse qualquer erro. Assim, não foi possível se obter uma curva de BER por taxa de transmissão, por exemplo. Entretanto, poderia ser factível afirmar que o sistema opera abaixo do nível de referência de 1×10^{-3} BER para todas as frequências analisadas.

Um fator que pesa a favor desta última indicação, foi uma limitação observada no sistema e que também pode ser encontrada na Figura 40. Nela, pode-se observar a forma de onda recebida com o auxílio do analisador de ondas digital *Piscope*. Neste programa, ilustrou-se a forma do sinal capturado pelo RPi de recepção para um sinal transmitido com a taxa de 1.000.000 bps (1 Mbps), escolhida para exemplificação. Entretanto, o que foi observado é que neste caso, o menor tempo de bit foi de 5 microssegundos. Isto indica que, na realidade, o sistema não operava em 1 Mbps, mas em 200 kbps, mas mesmo mais lento não apresento erros.

Figura 40 – Forma de onda recebida com a biblioteca *Serial* (UART)



Fonte: Produzido pelo próprio autor.

O Autor acredita que uma justificativa para tal resultado pode residir na robustez do protocolo UART, ou então do próprio *hardware* serial em detectar e minimizar possíveis fontes de erros, como atestado em MICROSHIP (2017). Pois se os dados continuaram chegando de forma correta em uma taxa observada na recepção bem abaixo do que aquela escolhida pela transmissão, poderia ser um indício de que a seleção original de 1 Mbps foi ignorada e reajustada para o novo valor de 200 kbps.

Outra alternativa, entretanto, pode ser uma própria limitação física do microcomputador, previamente conhecida pela biblioteca, a qual também deverá ser encontrada nos demais métodos de transmissão propostos, caso seja este o caso de fato. Finalmente, pode-se observar na Figura 41 uma exemplo de mensagem de protocolo LabTel-I recebida com a biblioteca Serial (UART).

Figura 41 – Mensagem LabTel-I com a biblioteca *Serial* (UART)

```
-----------------------------------------------------------------------
Medicao Numero 1:
<|20210331181834|911|Klaas^van der Zwaag|F|1.7.4.147842|73|1.3.1.150456|97|1.3.1.149
530|73|1.7.1.151578|1106|1.2.1.150344|1106|1.2.2.150344|1106|>

> Numeros de Erros:  0
> BER:  0.0
> Numero de Bits de Referência:  11680
> Numero de Bits Recebidos:  11680
> Tempo de Execução: 3.265257 s
> Taxa: 3577.053420 amostras/s
> Baudrate =  9600

-----------------------------------------------------------------------

> empty
> empty
> empty
> empty
> empty
> empty
> empty
```

Fonte: Produzido pelo próprio autor.

5.1.2 RPI.GPIO

Para o segundo método, já com a biblioteca RPi.GPIO, adotou-se uma abordagem semelhante à anterior, utilizando o mesmo pacote (dataout1) de 201 bits embaralhados de forma pseudoaleatória. Inicialmente, buscou-se analisar, por meio de testes, qual fator de atraso (em porcentagem de uma unidade de tempo de bit) resultaria na menor ocorrência de erros. O objetivo foi verificar se o atraso de meio tempo de bit, originalmente assumido como ideal na seção 4.3.2, seria de fato o que melhor atuaria. Sendo assim, foi enviado um conjunto de 10 medições para cada atraso proposto na Figura 42:

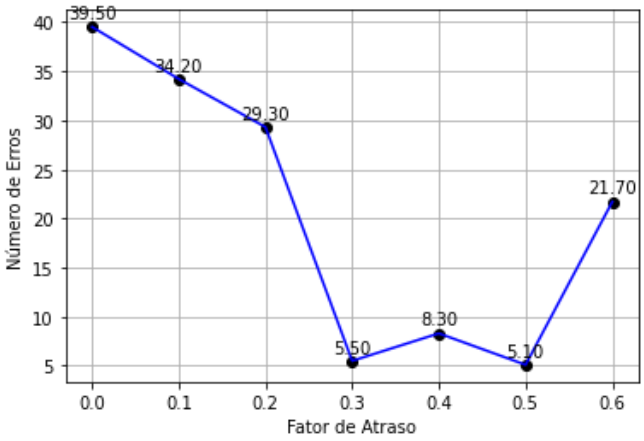
Figura 42 – Atrasos percentuais testados

```
[30] atrasos = [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6]
```

Fonte: Produzido pelo próprio autor.

O primeiro seria, na realidade, sem nenhum atraso enquanto que o último representa um retardo na leitura de 60% de um tempo de bit. Considerando que mesmo que seja escolhido o primeiro caso, já existirá um atraso natural entre o tempo real de leitura do bit e o tempo em que esta é notificada e armazenada pelo programa. Sendo assim, é plausível imaginar que para se ler no “meio do bit’, como desejado, deva-se adicionar atrasos sempre menores ou no máximo iguais à metade de um período. Entretanto, testou-se também a comunicação para um atraso de 60% como forma de reforçar esta última afirmação, ou seja, para este valor o número de erros deveria tender a um aumento. Por fim, a taxa escolhida para este experimento foi de 100 bps e os resultados obtidos encontram-se no Gráfico 1.

Gráfico 1 – Curva de erros por atrasos



Fonte: Produzido pelo próprio autor.

É possível observar que os melhores resultados ocorreram para os atrasos de 30 e 50% de tempo de bit, sendo o melhor deles este último. Ainda, foi possível confirmar que para valores maiores que 0,5 o número de erros torna a crescer consideravelmente. Sendo assim, escolheu-se, de fato, o 0,5 como atraso definitivo em todos os testes subsequentes para a biblioteca RPi.GPIO. É importante reforçar que estes atrasos são adicionados somente na leitura do primeiro bit, servindo de base como o instante de início desta. Entretanto, para as demais leituras, considera-se sempre um tempo de bit inteiro como já mostrado na Figura 34.

Voltando aos testes de BER, enviou-se 15 pacotes de 201 bits para cada frequência listada a fim de se chegar ao mesmo nível de confiança do BER estabelecidos na seção anterior. Assim, cada taxa de transmissão observada na Figura 43 foi testada com 3015 bits cada. Ainda, buscou-se espaçar as taxas de transmissão de 5 em 5 bps de forma a facilitar a localização da região do gráfico cujo BER associado esteja dentro dos níveis desejados, encontrando-se, assim, a limitação do método

Figura 43 – Taxas escolhidas para o primeiro teste com a biblioteca RPi.GPIO

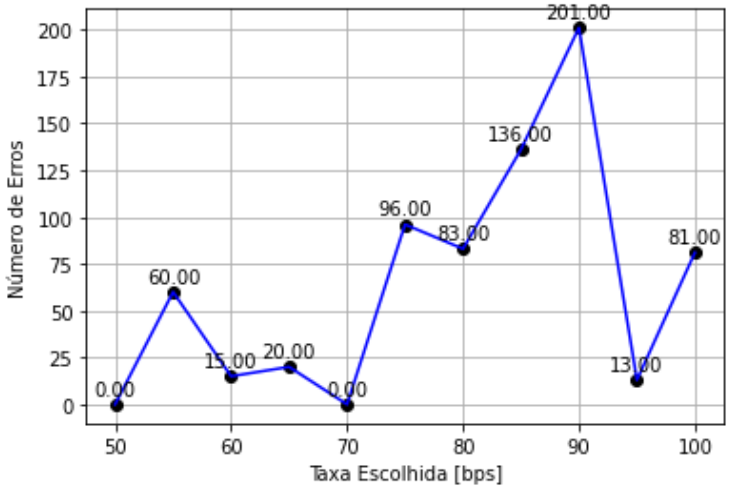
```
[ ]  frequencies = [50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100]
```

Fonte: Produzido pelo próprio autor.

Percebe-se do Gráfico 2 que as taxas 50 e 70 bps não apresentaram erros, sugerindo que possivelmente seja esta a região onde se encontra a frequência limite para a biblioteca. Já o

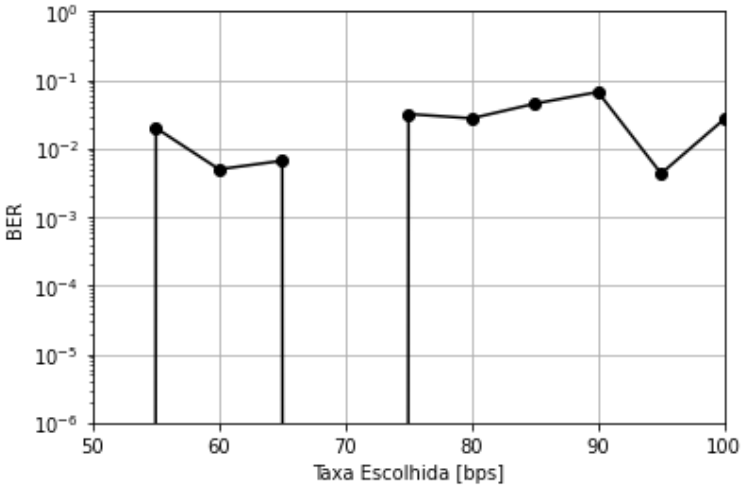
Gráfico 3 também corrobora tal questionamento pois as frequências 60 e 65 bps já se encontram na escala de referência para o BER, com valores aproximados de $4{,}98\times10^{-3}$ e $6{,}63\times10^{-3}$, respectivamente.

Gráfico 2 – Curva de erros entre 50 e 100 bps



Fonte: Produzido pelo próprio autor.

Gráfico 3 – Curva de BER entre 50 e 100 bps



Fonte: Produzido pelo próprio autor.

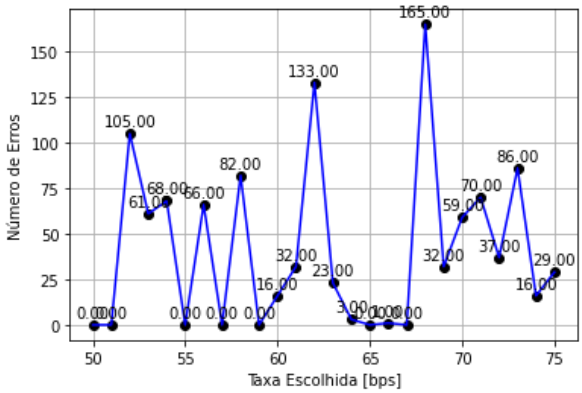
A fim de se chegar a resultados ainda mais precisos, elaborou-se um novo teste, agora com cada unidade de frequência compreendida entre 50 e 75 bps, como se observa na Figura 44. Já o Gráfico 4 demonstra uma grande flutuação do número de erros, dificultando a interpretação dos dados. Entretanto, pode-se perceber que a partir dos 68 bps o número de erros dá um salto e não retorna mais a zero.

Figura 44 – Taxas escolhidas para o segundo teste com a biblioteca RPi.GPIO

```
[23] frequencies = [50, 51, 52, 53, 54, 55, 56, 57, 58, 59,
                    60, 61, 62, 63, 64, 65, 66, 67, 68, 69,
                    70, 71, 72, 73, 74, 75]
```

Fonte: Produzido pelo próprio autor.

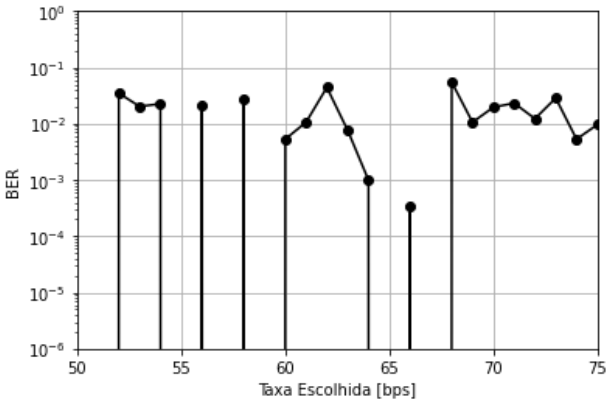
Gráfico 4 – Curva de erros entre 50 e 75 bps



Fonte: Produzido pelo próprio autor.

Já o Gráfico 5 indica que aproximadamente metade das novas taxas encontra-se com BER abaixo de 1×10^{-2}, ou seja, na faixa de 10^{-3}. Apesar do aumento de erros ocorrido a partir dos 68 bps, percebe-se que para taxas menores ainda podem ser encontrados resultados de BER em níveis indesejáveis. Assim, um valor mais confiável para se recomendar como taxa limite poderia ser os 50 bps. Esta última taxa é relevante não somente por não apresentar erros neste último teste, mas por não o ter feito também no teste anterior. Ou seja, de 6030 bits enviados a 50 bps, todos foram recebidos de forma correta.

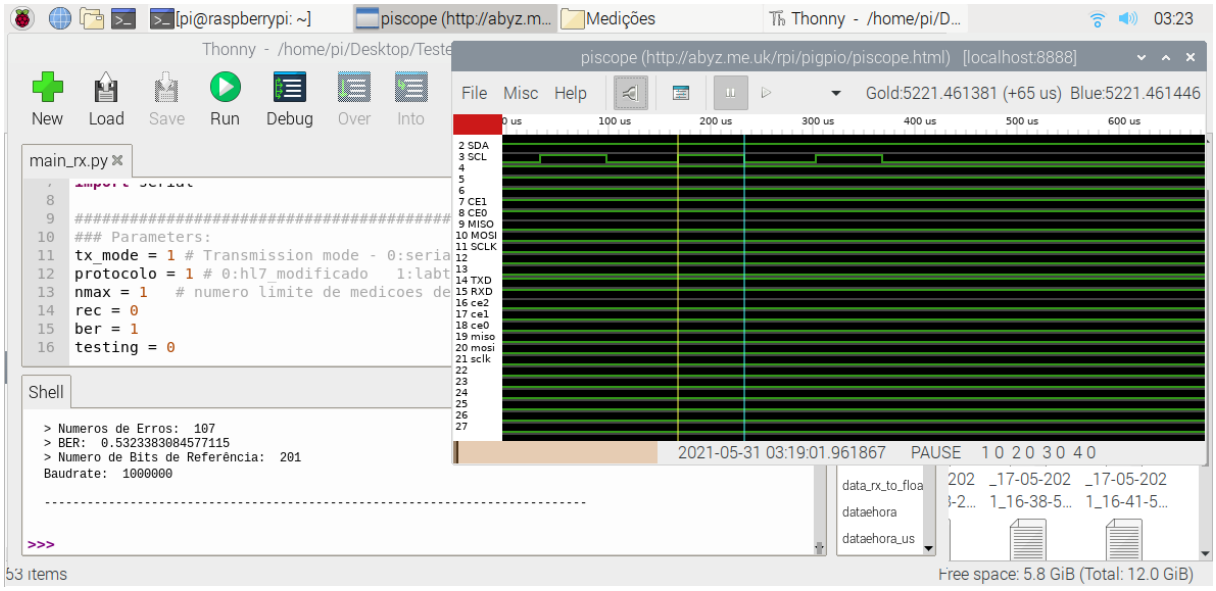
Gráfico 5 – Curva de BER entre 50 e 75 bps



Fonte: Produzido pelo próprio autor.

Outro resultado interessante é a aparente limitação física de chaveamento associada à biblioteca. Por ter sido completamente programada em *Python*, e sem acesso direto à memória, a RPI.GPIO só foi capaz de chavear a uma taxa máxima de aproximadamente 15,385 kbps (dado o tempo de bit medido em 65 µs), ainda que escolhido 1 Mbps como ilustrado na Figura 45. Entretanto, conforme visto nesta mesma seção, tal resultado esteve longe de ser verificado com um alto nível de confiança para o código implementado.

Figura 45 – Forma de onda recebida com a biblioteca RPi.GPIO



Fonte: Produzido pelo próprio autor.

Finalmente, pode-se observar na Figura 46 um exemplo de mensagem recebida com a biblioteca RPi.GPIO.

Figura 46 – Mensagem recebida com a biblioteca RPi.GPIO

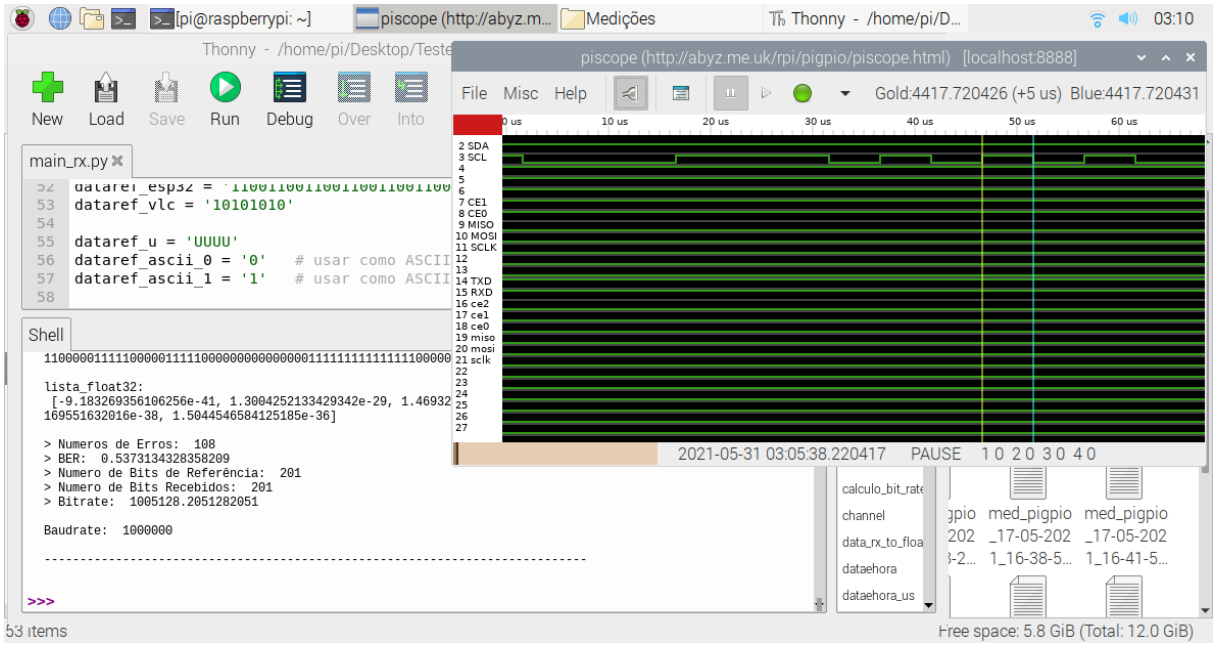
```
TX: NRZ OOK GPIO mode
------------------------------------------------------------------------

listening...
> Numeros de Erros:  0
> BER:  0.0
> Numero de Bits de Referência:  201
> Numero de Bits Recebidos:  201

t = 2.051038 s
f (baudrate) = 100 amostras/s
Taxa = 97.999170 bits/s
------------------------------------------------------------------------
```

Fonte: Produzido pelo próprio autor.

5.1.3 Pigpio

O último modo de transmissão e recepção da primeira configuração do sistema foi aquele baseado na biblioteca Pigpio. Iniciando-se com a limitação do método, é possível observar na Figura 47 que o menor tempo de bit foi de 5 microssegundos. Assim, a maior taxa observada foi de 200 kbps, a mesma encontrada para transmissão serial (UART), ainda que se tenha buscado os 1000000 bps (1 Mbps) ilustrados no canto inferior da citada figura. Entretanto, é necessário ressaltar que, diferentemente do UART, a comunicação nesta taxa limite passa a ocorrer com muitos erros, tornando-a inaceitável para os padrões de qualidade previamente estabelecidos.

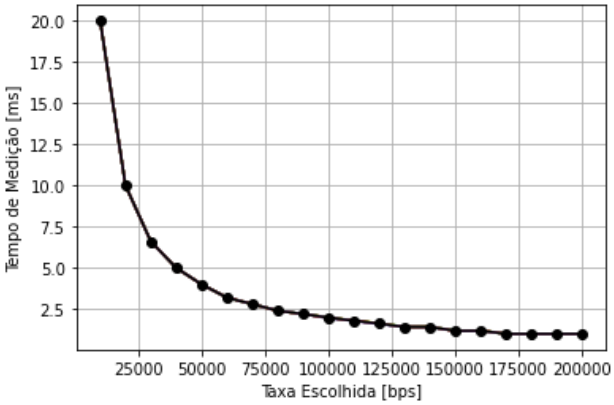
Figura 47 – Forma de onda recebida com a biblioteca *Pigpio*



Fonte: Produzido pelo próprio autor.

De fato, o Gráfico 6 confirma a limitação a partir da relação entre os tempos de medição observados e as taxas de bit selecionadas em ambos os lados do sistema. Percebe-se que mesmo que se escolha taxas de transmissão cada vez maiores, a curva tende ao valor de aproximadamente 1 ms, o que indica um tempo de bit próximo a 4,97 µs. Sendo assim, a taxa de transmissão tenderia a 201 kbps, valor que corroboraria a inferência anteriormente proposta.

Gráfico 6 – Relação dos tempos medidos (10 e 200 kbps)

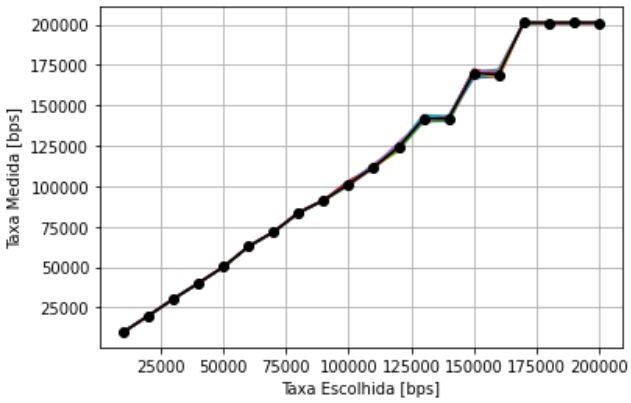


Fonte: Produzido pelo próprio autor.

Na realidade, o resultado corrobora também com a própria documentação da biblioteca Pigpio, que define como valor padrão o período de amostragem de 5,00 µs. Ou seja, quando se escolhe transmitir a 200 kbps, se está transmitindo com uma taxa igual a de amostragem. Assim, pelo teorema de Nyquist comentado na seção 2.1, a recepção será incapaz de recuperar o sinal. Logo, para que esta possa fazê-lo, ainda de acordo com o teorema, a comunicação deverá ser realizada com uma taxa abaixo ou igual a 100 kbps.

O Gráfico 7 também é importante pois demonstra uma linearidade na leitura para as frequências mais baixas, onde a taxa de bit medida corresponde à taxa escolhida. Entretanto, a partir de 130 kbps, percebe-se que o sistema começa a ter dificuldades em responder a novos incrementos, deixando de fazê-lo de forma linear. Isto implica em uma perda de precisão na comunicação, provavelmente resultando em um consequente aumento do número de erros.

Gráfico 7 – Relação das taxas medidas entre 10 e 200 kbps



Fonte: Produzido pelo próprio autor.

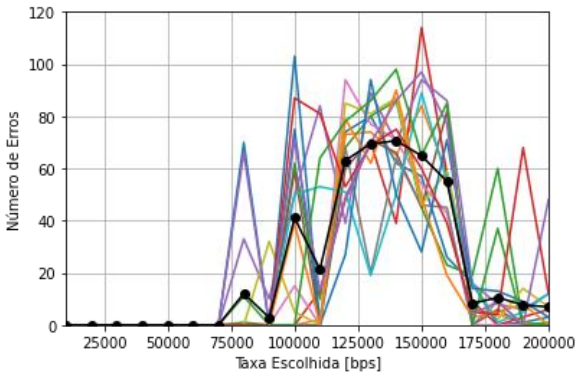
Os Gráficos 8 e 9, bem como os Gráficos 6 e 7, foram gerados a partir de um teste similar àquele realizado para a biblioteca RPi.GPIO. Enviou-se, para cada frequência listada na Figura 48, 15 pacotes de 201 bits organizados de forma pseudoaleatória, totalizando 3015 bits transmitidos. É importante ressaltar que cada curva colorida representada nos gráficos citados refere-se a um dos 15 pacotes enviados. Excetuando-se apenas a curva na cor preta, que representa as médias de todas as demais, sendo então utilizada como referência. Para os casos em que as curvas coloridas não sejam visíveis nestes gráficos, entenda-se que estas estejam sobrepostas e escondidas pela curva mais escura.

Figura 48 – Taxas escolhidas para o primeiro teste com *Pigpio*

```
[ ] nbits_ref = 201
    nmax = 15
    frequencies = [10000, 20000, 30000, 40000, 50000,
                   60000, 70000, 80000, 90000, 100000,
                   110000, 120000, 130000, 140000, 150000,
                   160000, 170000, 180000, 190000, 200000]
```

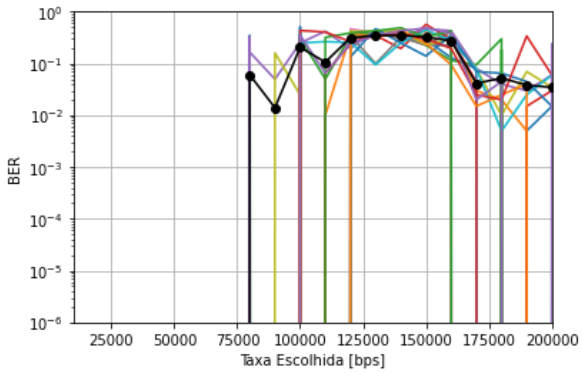
Fonte: Produzido pelo próprio autor.

Gráfico 8 – Curvas de erros entre 10 e 200 kbps



Fonte: Produzido pelo próprio autor.

Gráfico 9 – Curvas de BER entre 10 e 200 kbps



Fonte: Produzido pelo próprio autor.

Ainda nos Gráficos 8 e 9, pode-se observar em ambos que entre o 120 e o 160 kbps ocorre um pico ainda maior no número de erros. Acredita-se que isto ocorra devido à perda de linearidade do sistema, já mencionada, embora já esteja ocorrendo erros também para frequências mais baixas. De fato, após os 160 kbps a incidência de erros continua com níveis não aceitáveis porém sofre uma redução drástica. Uma possível explicação para tal fenômeno pode ser a observância de um novo trecho de linearidade na curva, onde ela passa a manter-se constante ao atingir sua condição limitante (200 kbps).

Quanto aos erros observados aproximadamente entre 70 e 120 kbps, atribuem-se ao fenômeno conhecido como *jitter*. Como explicado por Sklar (2001), *jitter* temporais são pequenas variações nos intervalos de amostragem de reconstrução do sinal, causando assim uma dessincronização caso o processo dependa das posições exatas das amostras. Sendo este o caso do método utilizado.

Tanto as curvas do Gráfico 8 quanto do Gráfico 9 indicam que uma comunicação com níveis de erro aceitáveis pode ocorrer para até valores próximos dos 70 kbps. Sendo assim, novos testes foram realizados de forma a se observar melhor o que acontece nesta região. Para os testes, foram enviados 45 pacotes a cada nova frequência. Destes, 15 foram do mesmo conjunto de dados pseudoaleatórios (dataout1), 15 de bits alternados (dataout2) e os outros 15 de bits repetidos, dito contínuo (dataout3). Todos os pacotes continham 201 bits a serem enviados. As novas frequências podem ser vistas na Figura 49, enquanto que os dois novos conjuntos de dados estão ilustrados na Figura 50.

Figura 49 – Taxas escolhidas para o segundo teste com a biblioteca *Pigpio*

```
[ ] nbits_ref = 201
    nmax = 3
    frequencies = [50000, 51000, 52000, 53000, 54000, 55000,
                   56000, 57000, 58000, 59000, 60000,
                   61000, 62000, 63000, 64000, 65000,
                   66000, 67000, 68000, 69000, 70000,
                   71000, 72000, 73000, 74000, 75000,
                   76000, 77000, 78000, 79000, 80000,
                   81000, 82000, 83000, 84000, 85000,
                   86000, 87000, 88000, 89000, 90000,
                   91000, 92000, 93000, 94000, 95000,
                   96000, 97000, 98000, 99000, 100000]
```

Fonte: Produzido pelo próprio autor.

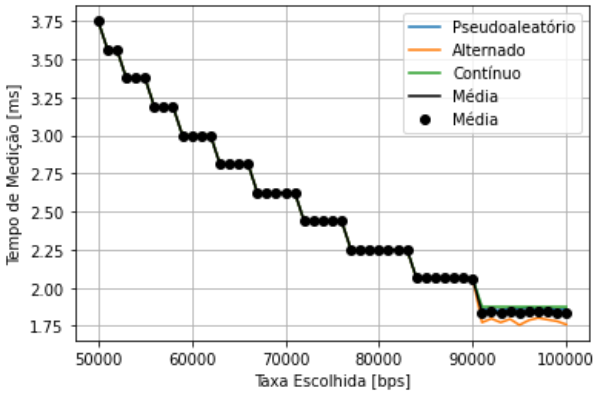
Figura 50 – Pacotes de testes pseudoaleatórios, alternados e contínuos

```
[168] #string1g = geraDados(seed=1, n=192)
      string1 = '1100111110010110010001001000100011111000111111011001001110100110111100110000111010011011010011101101111000001111111000000111111001100100011111010100011101000010101110000001
      string2 = '0101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101
      string3 = '0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                
      header = '1111'
      parity_bit_0 = '0'
      parity_bit_1 = '1'
      footer = '0000'

      dataout1 = header + string1 + parity_bit_0 + footer
      dataout2 = header + string2 + parity_bit_0 + footer
      dataout3 = header + string3 + parity_bit_0 + footer

      lista_dataout = [dataout1, dataout2, dataout3]
      #lista_dataout = nmax*[dataout1]   # nmax = 3, 15

      print("dataout1 = ", dataout1)
      print("dataout2 = ", dataout2)
      print("dataout3 = ", dataout3)

      dataout1 =  1111110011111100101100100010010001000111110001111101100100111010011011110011000011101001101101001110110111100000111111000000111111001100100011111010100011101000010101110000
      dataout2 =  1111010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101
      dataout3 =  1111000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
```

Fonte: Produzido pelo próprio autor.

O objetivo de se adicionar os dois novos conjuntos de dados foi verificar o comportamento do sistema em situações mais extremas. Seja para quando os dados variam ininterruptamente (dataout2), seja para quando eles praticamente não variam (dataout3). Sendo assim, pode-se observar no Gráfico 10 a constatação prévia de que o tempo de medição tende cada vez mais a se acomodar para um valor constante. Isto pode ser evidenciado pelo crescente número de amostras para um mesmo nível, ou “degrau”, presente na curva, quando esta vai se aproximando de 1ms.

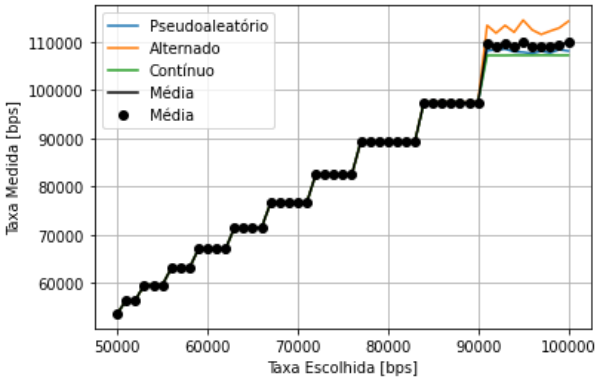
Gráfico 10 – Relação dos tempos (50 e 100 kbps)



Fonte: Produzido pelo próprio autor.

Já o Gráfico 11 demonstra um comportamento bem parecido com o anterior, também se acomodando em torno de um valor limite e constante, que já se sabe ser próximo dos 200 kbps. Estes dois últimos gráficos são interessantes pois mostram que a partir de 91 kbps o sistema começa a responder de forma diferente, dependendo da natureza dos dados

Gráfico 11 – Relação das taxas medidas (50 e 100 kbps)

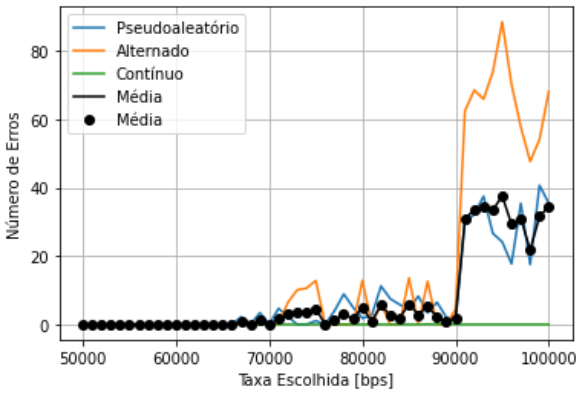


Fonte: Produzido pelo próprio autor.

Percebe-se claramente que os pacotes de bits contínuos (curva verde) passam a apresentar uma resposta mais lenta, porém precisa, enquanto que os pacotes de bits alternados (curva laranja) atuam de forma contrária. Como era de se esperar, os pacotes pseudoaleatórios (curva azul) apresentam um comportamento intermediário, justamente por misturar características dos dois conjuntos anteriores.

Quanto à análise de erros, representados pelo indicador BER, observa-se dos Gráficos 12 e 13 que, de fato, eles começam a surgir bem próximos à frequência de 70 kbps. Além disso, observa-se que os pacotes de bits contínuos apresentaram os melhores resultados, com erros somente em três frequências específicas (81, 87 e 89 kbps), mas ainda com BER dentro da faixa de 1×10^{-3}.

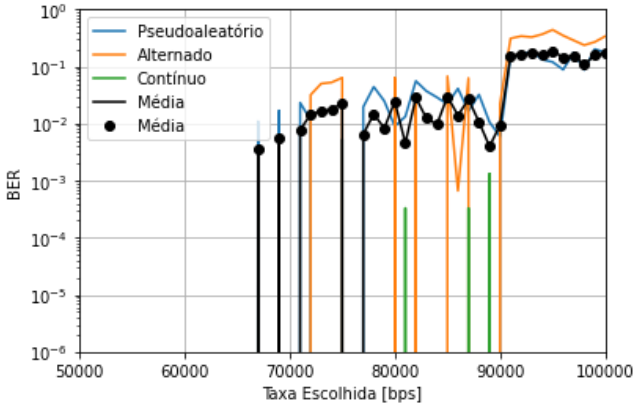
Gráfico 12 – Curvas de erros entre 50 e 100 kbps



Fonte: Produzido pelo próprio autor.

Já os pacotes de bits alternados apresentou resultados semelhantes aos pseudoaleatórios até que a, já comentada, frequência de 91 kbps fosse atingida. A partir deste ponto, o sinal sofre uma degradação drástica, tornando a comunicação impraticável (com ordem de BER de 1×10^{-1}, ou seja, acima dos 10%).

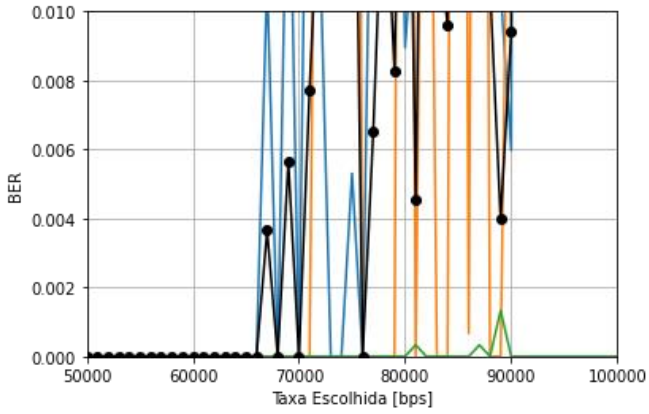
Gráfico 13 – Curvas de BER entre 50 e 100 kbps



Fonte: Produzido pelo próprio autor.

O Gráfico 14 ainda ajuda a melhor observar que até 71 kbps o sistema apresenta um resultado abaixo de 1×10^{-3}, valor considerado como aceitável no presente trabalho. Porém um limite mais conservador a se considerar poderia ser transmitir a informação a 66 kbps, em razão de ser esta a frequência para a qual os primeiros erros começam a ocorrer.

Gráfico 14 – Curvas de BER entre 50 e 100 kbps (ampliada)



Fonte: Produzido pelo próprio autor.

É válido destacar que 66 kbps representa 1/3 da taxa de amostragem da biblioteca (200 kbps). Ou seja, está dentro da taxa de transmissão limite estabelecida pelo critério de Nyquist (100 kbps, metade da amostragem) e representa 2/3 ou 66% da largura de banda. Ainda, a biblioteca

permite que se aumente a taxa de amostragem até 1 Mbps, sugerindo uma taxa de transmissão máxima de 500 kbps ou 333 kbps, livre de erros, comparando-se com o sistema atual. Entretanto, sua própria documentação não recomenda tal utilização a fim de se evitar forçar o hardware do RPi, o que reduziria seu tempo de vida.

Outro fator relevante na discussão é a fonte dos erros observados. Conforme Periph Io (2017), processos de leitura costumam ser testados com sequências de um único bit por apresentarem uma maior dificuldade em manter a leitura. Por outro lado, processos de escrita são geralmente testados alternando-se os bits o máximo de vezes quanto possível, já que apresentam dificuldade em fazer um chaveamento persistente em taxas mais altas. Desse modo, pode-se concluir que o sistema em questão é limitado principalmente pelo lado da transmissão, já que para frequências mais altas os pacotes de bits alternados apresentam um elevado índice de erros, enquanto que os pacotes de bits contínuos mantêm o bom desempenho.

Como também disponibilizado para as demais bibliotecas, pode-se observar na Figura 51 um exemplo de mensagem em protocolo LabTel-II recebida com a biblioteca *Pigpio*.

Figura 51 – Mensagem recebida com a biblioteca RPi.GPIO

```
Listening...

--------------------------------------------------------------------------
Medição 1:
string_final:
 1111010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101
010101010101010101010101010101010101010101010101010101010101010101010101010100000

lista_float32:
 [14660154687488.0, 14660154687488.0, 14660154687488.0, 14660154687488.0, 14660154687488.0, 14660154687488.0]

> Numeros de Erros:  0
> BER:  0.0
> Numero de Bits de Referência:  201
> Numero de Bits Recebidos:  201
> Bitrate:  1005.0712733283674

Baudrate:  1000

--------------------------------------------------------------------------
```

Fonte: Produzido pelo próprio autor.

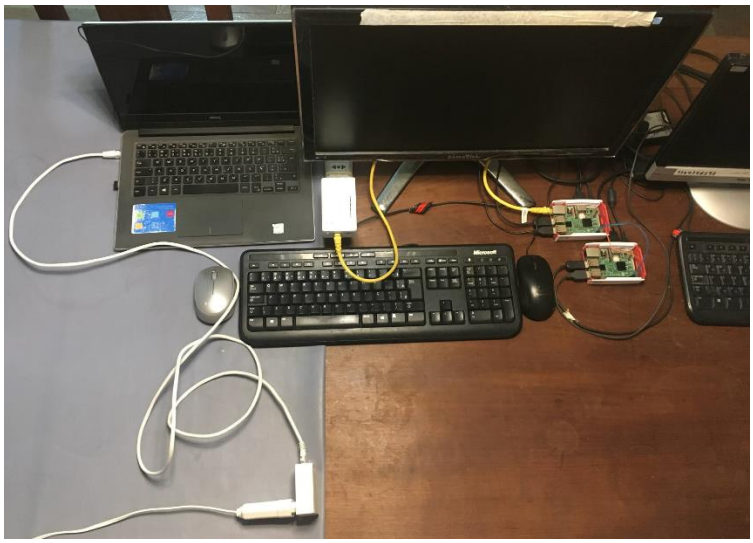
Por fim, foi disponibilizado no Apêndice C uma coletânea de figuras com os sinais transmitidos junto aos recebidos, lado a lado, a fim de se ilustrar a veracidade da comunicação. Observa-se em especial que no caso da biblioteca Serial (UART), os pacotes de 10 bits representando os valores “0” e “1” (incluindo os bits de cabeçalho e rodapé) foram separadamente destacados. Já para a RPi.GPIO mostra-se que quando ela está em seu limite, alguns bits em nível alto não são corretamente recuperados. Por fim, demonstra-se com a biblioteca *Pigpio* que quando a taxa de transmissão sai de um valor dentro de seu limite (66 kbps) para outro que está acima

dele, surge o fenômeno do *jitter* temporal (primeiro nível alto), causando uma grande quantidade de erros.

5.2 Configuração 2 (PLC-B2B)

Para a segunda configuração e, consequentemente, seus respectivos testes, adicionou-se o par transmissor-receptor de sinal PLC, ao sistema, mantendo a conexão *back-to-back* dos microcomputadores RPi. Exceto quando da realização dos testes, momento em que o trecho PLC foi desconectado do setup para ser estudado separadamente. A Figura 52 ilustra como ficou o novo *setup*.

Figura 52 – Configuração 2 (PLC-B2B)



Fonte: Produzido pelo próprio autor.

Para se caracterizar o canal PLC dentro do sistema, realizou-se medições das chamadas taxas de *powerline* (*powerline rate*) variando a posição do receptor PLC ao longo do ambiente de testes. O dispositivo de transmissão manteve-se conectado ao mesmo ponto na rede (tomada 0) durante todo o experimento. Quanto às medições, foram feitas 5 coletas de dados, escolhendo-se tomadas de forma que se pudesse cobrir o lado de maior distância do apartamento. Tanto no Apêndice E quanto no Quadro 3, pode-se encontrar a numeração dessas tomadas, indo de zero a cinco. Neste último, é possível ainda observar as distâncias de cada tomada para o dispositivo transmissor.

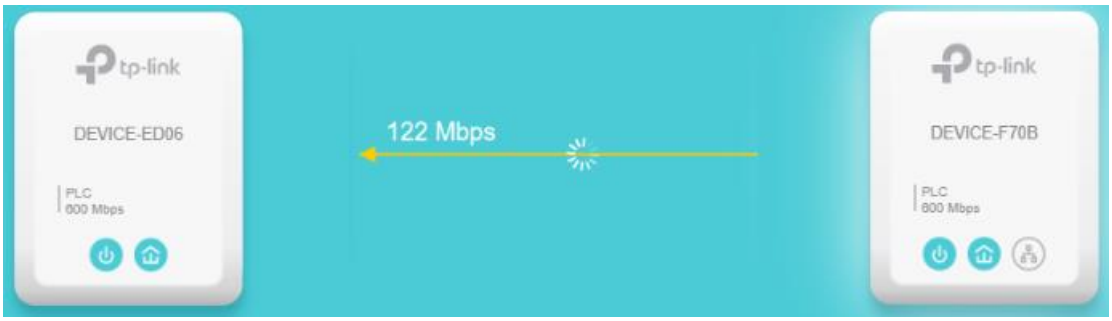
Quadro 3 – Informações das tomadas, circuitos e novas medições da comunicação PLC

Local	Tomada	Circuito	Distância (m)	Taxa *Powerline* (Mbps)
Quarto dos fundos (TX)	0	10	0	-
Quarto dos fundos	1	10	6,2	362
Área de Serviço	2	9	8,1	234
Cozinha	3	3	8,7	152
Sala I	4	1	10,9	102
Sala II	5	1	15,8	90

Fonte: Produzido pelo próprio autor.

Inicialmente, buscou-se utilizar, para as medições, o programa *Wireshark*. Este *software* é conhecido por ser capaz de capturar pacotes de dados em uma determinada rede e interpretá-los de forma gráfica em tela (WIRESHARK, 2021). No Apêndice D pode-se encontrar uma das capturas de tela do programa mostrando que, de fato, o protocolo *HomePlug* AV foi reconhecido e que vários pacotes, sob este protocolo, foram enviados do dispositivo PLC de transmissão para o notebook. Em seguida, utilizou-se o programa *tpPLC Utility*, da TP-LINK (marca dos extensores de sinal), onde foi possível obter as taxas de *powerline* para cada um dos pontos analisados. A Figura 53 mostra um recorte da tela do programa quando em funcionamento.

Figura 53 – Ilustração de uma das medições da taxa de *powerline* (aplicativo tpPLC)



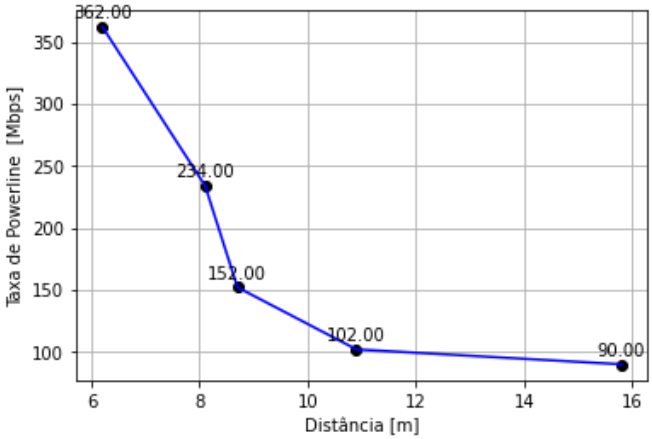
Fonte: Produzido pelo próprio autor.

Para a realização das medições, utilizou-se a planta elétrica (disponibilizada no Apêndice E) do apartamento que serviu de ambiente para o experimento, a fim de se obter resultados mais precisos do que uma simples medição direta na parede. Através da análise da planta e também de testes junto ao quadro de distribuição, foi possível observar que as tomadas encontravam-se dispostas, na maior parte delas, em circuitos diferentes umas das outras. Dessa forma, a simples medição direta entre as tomadas não reproduziria de maneira fiel a topologia em que se encontravam os pontos em análise. Logo, calculou-se os tamanhos dos circuitos utilizando-se

de valores conhecidos como referência como, por exemplo, a altura real das tomadas e do quadro, bem como as dimensões da sala. Assim, pôde-se calcular as distâncias entre as tomadas. Observa-se, ainda, que para aquelas tomadas que não se encontravam sob o mesmo circuito do dispositivo transmissor, considerou-se suas distâncias, primeiramente, até o quadro e, por conseguinte, do quadro até o ponto de instalação da tomada de número zero.

É importante ressaltar que a planta disponibilizada não se encontra completamente atualizada, visto que ocorreram obras, não documentadas, após sua instalação original. As maiores mudanças, entretanto, ocorreram nas numerações dos circuitos. Assim, o Autor contou com informações do proprietário e responsável pela obra, além dos próprios testes junto ao quadro para concluir os cálculos. Finalmente, sugere-se que o documento seja utilizado apenas para melhor se compreender as localizações sob as quais as tomadas encontram-se dispostas, sendo marcadas com a cor rosa e numeradas conforme encontrado no Quadro 3. Finalmente, pode-se observar no Gráfico 15 o resultado obtido com as medições.

Gráfico 15 – Curva da taxa de *powerline* em função das distâncias medidas



Fonte: Produzido pelo próprio autor.

Da curva, pode-se perceber que esta apresentou a tendência de decair à medida que a distância do receptor ao transmissor aumentava. Este resultado era, de fato, esperado pois tal comportamento é reflexo da atenuação natural do canal, conforme estudado em Çelebi (2010). É importante ressaltar que nenhuma outra carga esteve ligada à rede quando do momento das medições, exceto pelas lâmpadas de iluminação dos respectivos ambientes. Tal medida facilitou na observação do comportamento esperado visto que outras cargas podem atuar como fontes de ruído e interferência na rede, distorcendo as medições.

Entretanto, observou-se também que a medição apresentou um valor um pouco acima da curva que se desenha quando se conecta todos os pontos medidos. Uma possibilidade para esse comportamento pode ser uma simples flutuação natural da medição, por exemplo por parte do próprio programa de captura. Porém uma outra possibilidade de efeito multipercurso, comumente presente em redes elétricas residenciais devido a descasamento de impedâncias nos circuitos, não pode ser descartada. É importante ressaltar que, como estudado por Çelebi (2010), este efeito junto ao ruído e a atenuação são os três principais causadores de distorção e, consequentemente, degradação da performance de um sistema PLC.

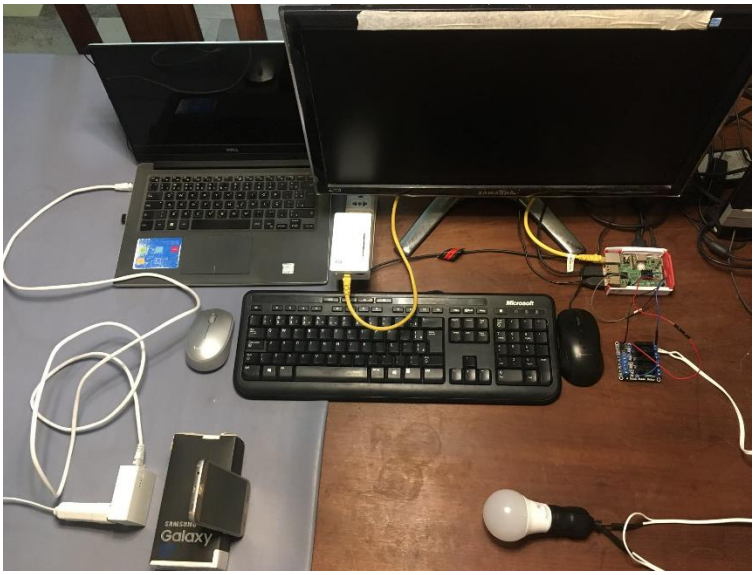
Finalmente, ressalta-se que as taxas medidas não são necessariamente as que de fato se observa na realidade. Questões como a quantidade de tráfego, número de cargas ligadas à rede e até mesmo a versão das próprias portas de entrada e saída dos dispositivos PLC fazem com que os valores medidos caiam quando do momento da aplicação (TP-LINK, 2020).

5.3 Configuração 3 (PLC-VLC)

Para a última configuração, conectou-se os pinos 01, 36 e 39 (3,3V, GPIO16 e *ground*, respectivamente) do RPi de transmissão nas respectivas entradas do módulo SSR de acionamento da lâmpada. Esta, por sua vez foi ligada, em série, entre um dos fios da rede elétrica e uma das portas do tipo “normalmente aberta” do módulo, de forma a fechar o circuito com um segundo fio ligado diretamente entre este último e a tomada. Por fim, o smartphone foi colocado em pé, de frente para a lâmpada, apoiado em sua própria caixa de embalagem. A Figura 54 ilustra a nova configuração.

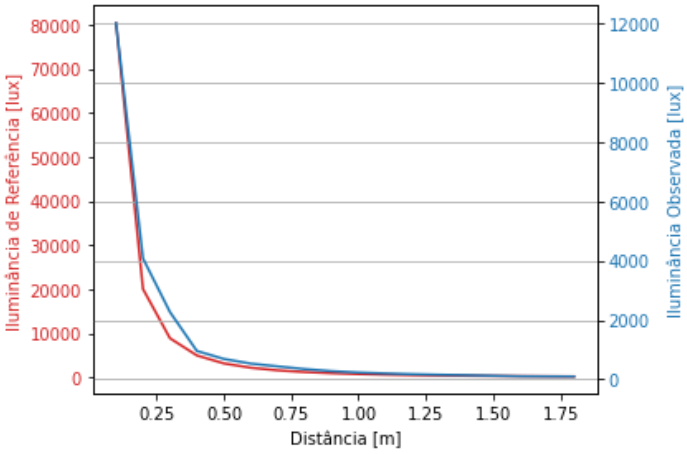
Em relação aos testes, inicialmente buscou-se caracterizar o trecho VLC do sistema através de medições da iluminância percebida pelo sensor de luminosidade ambiente do aparelho celular, sendo o parâmetro medido em lux. Para tanto, utilizou-se o Aplicativo 2, alterando a posição do *smartphone* a fim de variar sua distância para a fonte de luz, resultando em 18 pontos analisados. Com início a 10 cm e fim a 1,80 m de distância, adotou-se incrementos constantes (10 cm cada) de modo a se gerar a curva azul do Gráfico 16.

Figura 54 – Configuração 3 (PLC-VLC)



Fonte: Produzido pelo próprio autor.

Gráfico 16 – Curvas de iluminâncias de referência e observadas



Fonte: Produzido pelo próprio autor.

Já a curva vermelha foi levantada baseada na equação de iluminância em função do fluxo luminoso incidente e da distância do ponto à fonte de luz, encontrada em Schubert (2006). O valor de fluxo luminoso incidente utilizado como referência foi de 803 lm, retirado do Quadro 1 e relativo ao modelo de lâmpada utilizado. De ambos os resultados pode-se inferir que a intensidade observada do brilho LED decai rapidamente com a distância, assim como era esperado pela curva vermelha. Na verdade, pode-se perceber que as curvas apresentam um comportamento muito semelhante, tendo pequenas divergências em apenas poucos pontos (0,20 e 0,30 m), provavelmente devido à imprecisão nas medições.

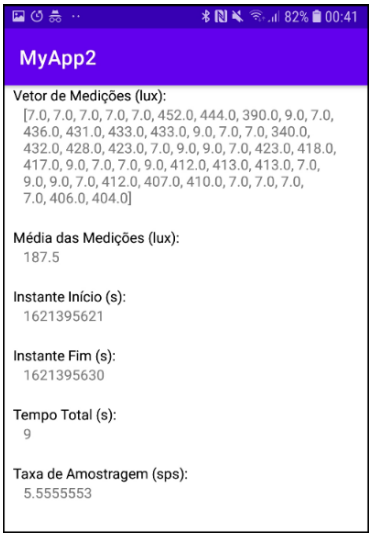
Apesar das curvas apresentarem um comportamento parecido, os levantamentos observados obtiveram valores em lux bem abaixo dos esperados, como se pode reparar pelos eixos verticais do Gráfico 16. Isto sugere que o sensor do celular insere um ganho de aproximadamente 0,1495 ou 15% nas medições, inerente ao processo de conversão analógica para digital do sinal luminoso coletado. Destaca-se, contudo, que a curva vermelha não representa o comportamento exato do brilho da lâmpada mas, na realidade, a percepção que o olho humano tem ao observá-la. Isto porque iluminância é uma grandeza dita fotométrica, ou seja, que realiza medições da luz em termos da capacidade visual humana (SCHUBERT, 2006).

Em seguida, para as curvas de BER, novamente enviou-se três formas distintas de pacotes: pseudoaleatórios, alternados e contínuos. Entretanto, todos eles foram baseados em um mesmo protocolo, LaTel-III, contendo cabeçalho '1010' e mais 6 bits de dados representando a leitura de 3 parâmetros (2 bits cada) de um monitor multiparamétrico. Foi escolhido enviar pacotes com pouco volume de dados devido às limitações, relativas à taxa de transmissão, observadas no dispositivo *smartphone*. A partir das Figuras 55 e 56, pode-se observar as taxas de amostragem, mínima e a máxima, capturadas pelo Aplicativo 1: 0,84 e 5,55 amostras por segundo, respectivamente.

Figura 55 – Imagem em tela do Aplicativo 2 (menor amostragem)

Fonte: Produzido pelo próprio autor.

Figura 56 – Imagem em tela do Aplicativo 2 (maior amostragem)



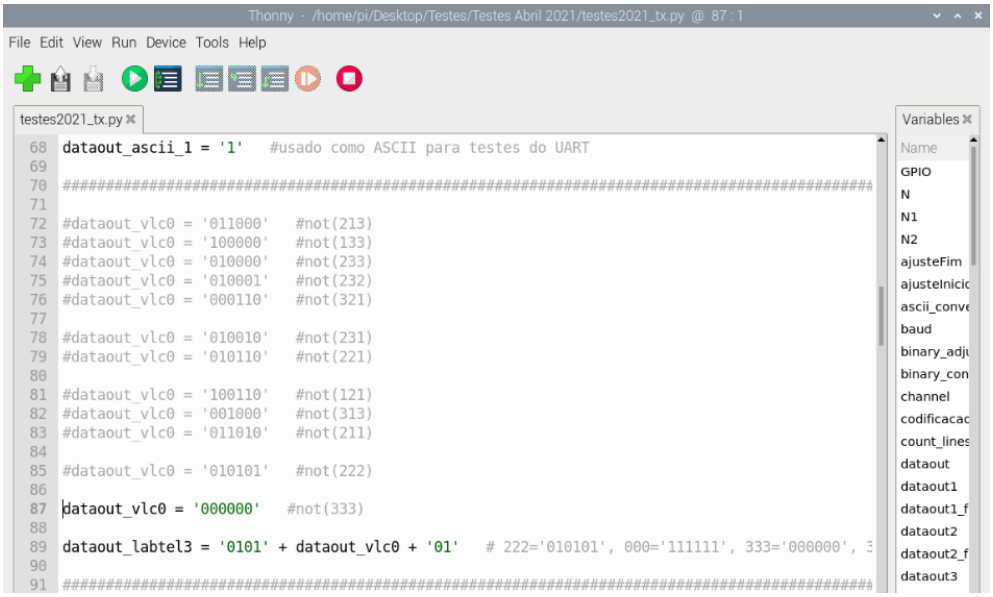
Fonte: Produzido pelo próprio autor.

De fato, percebe-se que os valores são muito baixos, limitando o sistema. Para o melhor caso, demorou-se 9 segundos para coletar 50 amostras. Para o pior caso, demorou-se quase 1 minuto (59 segundos) para se coletar a exata mesma quantidade de amostras. Logo, é razoável considerar que o envio de muitos bits, resultando em longas transmissões, de certa forma desvirtua o objetivo de aplicação do projeto, o qual exige uma rapidez na comunicação.

Em razão de tais resultados, escolheu-se taxas de envio de 1, 2 e 3 bps a fim de uma análise comparativa ao final dos testes. É importante ressaltar que todas as medições foram realizadas tanto com a luz totalmente acesa, quanto com ela totalmente apagada. Assim, ao se descontar esta última leitura da primeira, pôde-se desconsiderar os efeitos da luminosidade ambiente já presente no local.

Retornando aos pacotes, foram escolhidos 10 sequências de 6 bits de forma a se considerar como os dados pseudoaleatórios. Explica-se: cada par de bit representa um valor em base decimal que pode ser igual a 0 (‘00’), 1 (‘01’), 2 (‘10’) ou 3 (‘11’). Assim, escolheu-se dez diferentes combinações, três a três, dos pares de bits referentes aos valores 1, 2 e 3, buscando-se enviá-las de forma aleatória. Na Figura 57 pode-se encontrar tais combinações dispostas como linhas de código, comentadas, entre os números 72 ao 83.

Figura 57 – Pacotes pseudoaleatórios a serem recebidos pelo aparelho *smartphone*



Fonte: Produzido pelo próprio autor.

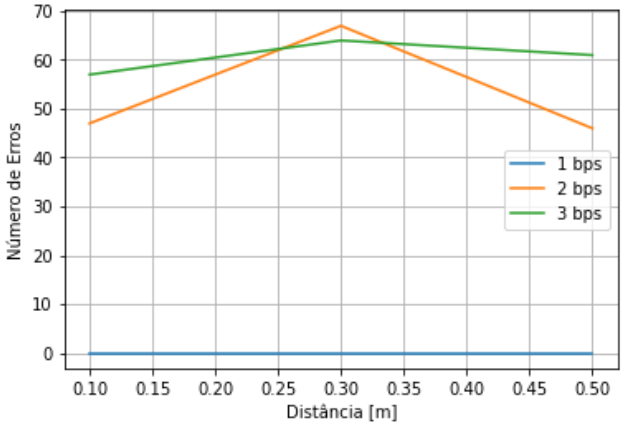
É importante destacar que os bits encontram-se trocados pois o SSR atua com lógica inversa. Logo, para se fechar o circuito e acender a lâmpada, deve-se enviar um sinal de bit 0, enquanto que para desligar se envia um bit 1. Quanto a lógica da escolha dos valores, o objetivo foi ordená-los com uma distribuição igualitária, a fim de se evitar favorecimento de um ou de outro. Assim, do total dos 10 pacotes selecionados, se tem 30 valores de medição, de fato, sendo 10 iguais a 1, 10 iguais a 2 e, finalmente, 10 iguais a 3.

Em relação aos pacotes de bits alternados e contínuos, estes foram representados pelas combinações 222 (‘101010) e 333 (‘111111’), respectivamente, também podendo ser encontrados na Figura 59 pelas linhas 85 e 87. Para ambos os casos, foram enviados 10 pacotes cada. Assim, pôde-se chegar a 300 bits no total para cada taxa de transmissão em cada distância analisadas. Ressalta-se que diferentemente dos experimentos em conexão *back-to-back*, os quais eram quase que totalmente automatizados e podendo facilmente gerar grandes quantidades de dados, os testes com o *smartphone* foram todos manuais, dificultando esse processo de coleta.

Já tratando dos erros obtidos, o Gráfico 17 mostra claramente como o aumento da taxa de transmissão impossibilita a comunicação para o sistema em análise. De fato, isto já era esperado ao menos para a taxa de 3 bps, já que esta encontra-se acima do limite imposto, pelo teorema da amostragem, para a recepção com o aparelho *smartphone* (entre 2,5 e 2,75 bps). Para a taxa

de 2 bps a análise é praticamente a mesma: apesar de não estar acima do limite do teorema, encontra-se muito próxima deste, o que também resulta em erros. Entretanto para 1 bps os resultados são completamente diferentes. Neste caso nenhum erro foi observado após o envio de 900 bits em tal taxa, o que configura uma bom indicador para a comunicação.

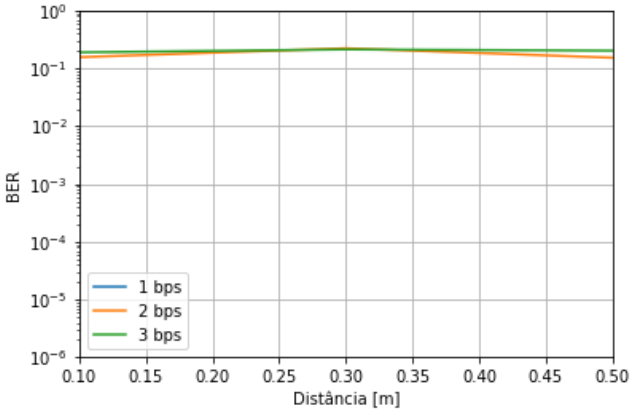
Gráfico 17 – Curvas de erros entre 10 e 50 cm



Fonte: Produzido pelo próprio autor.

De fato o Gráfico 18 confirma o resultado anterior. Enquanto para 2 e 3 bps o nível de BER é intolerável, para 1 bps nem é possível representá-lo no gráfico pois nenhum erro foi encontrado. Já sabendo que somente a taxa de 1 bps permite uma comunicação com níveis de erro aceitáveis, para o sistema PLC-VLC completo, realizou-se testes também para as distâncias de 1,0 e 1,8 m a fim de se observar como o sistema se comporta nestas condições. Neste caso, novamente não observou-se ocorrência de erros, totalizando agora 1500 bits transmitidos corretamente.

Gráfico 18 – Curvas de BER entre 10 e 50 cm

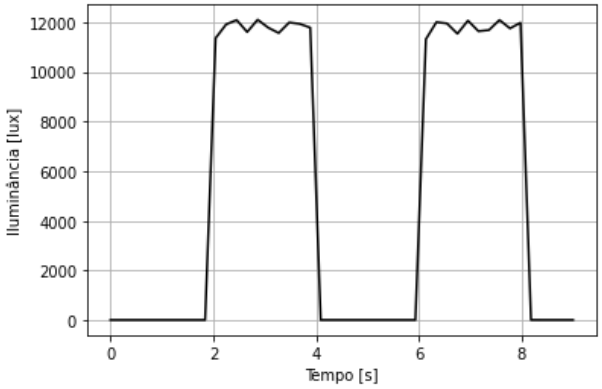


Fonte Produzido pelo próprio autor.

Logicamente, mais dados são necessários para se garantir uma precisão de no mínimo 1×10^{-3} indicando uma comunicação confiável a nível de erros. O resultado do Gráfico 18 não garante a inexistência de leituras incorretas, apenas sugere que o valor do BER encontra-se mais próximo da escala de 10^{-3} (dada a consideração de se enviar três vezes o número de bits do inverso da escala que se busca). De qualquer forma, os resultados obtidos já são suficientes para demonstrar que o sistema VLC proposto neste texto é factível de ser implementado mesmo com todas as limitações impostas pelo *setup*.

Com o auxílio do Aplicativo 2 e do *Google Colab*, pôde-se ainda realizar um processamento, dito off-line, do sinal luminoso recebido pelo sensor do celular. Para o novo experimento, enviou-se, com o RPi, a sequência '0011001100' a fim de se poder realizar uma análise qualitativa do sinal a partir de seu diagrama de olho. A frequência de envio foi de 1 bps e a distância foi de 10 cm. No Gráfico 19 pode-se observar que o sinal apresenta uma maior dificuldade em se manter estável quando em níveis mais altos. Isto é esperado pois realmente é de conhecimento comum que lâmpadas sofrem com pequenas flutuações em seus brilhos (devidas a ruídos na rede elétrica) ainda que não sejam visíveis ao olho humano.

Gráfico 19 – Sinal luminoso modulado

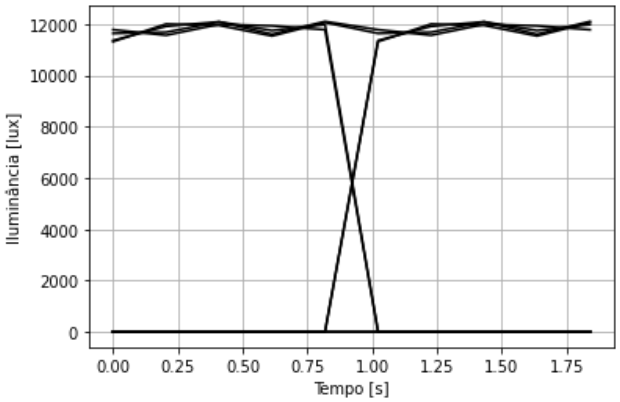


Fonte: Produzido pelo próprio autor.

Já no Gráfico 20 disponibiliza-se o diagrama de olho do sinal luminoso. É importar ressaltar que, ao menos para o código utilizado, o smartphone não foi capaz de indicar valores de tempo com boa precisão, arredondando-os sempre para a escala de segundos, como pôde ser observado nas na seção 4.4.2. Desse modo, análises como o tempo de subida ou abertura horizontal para verificação de *jitter* acabam ficando comprometidas dadas as condições. Entretanto, a análise de abertura vertical do olho se mantém válida. De fato, o olho bastante aberto verticalmente

indica um limiar de decisão limpo para a classificação dos bits como sendo altos ou baixos. Assim, o resultado corrobora com a observação de poucos ou nenhum erro encontrado.

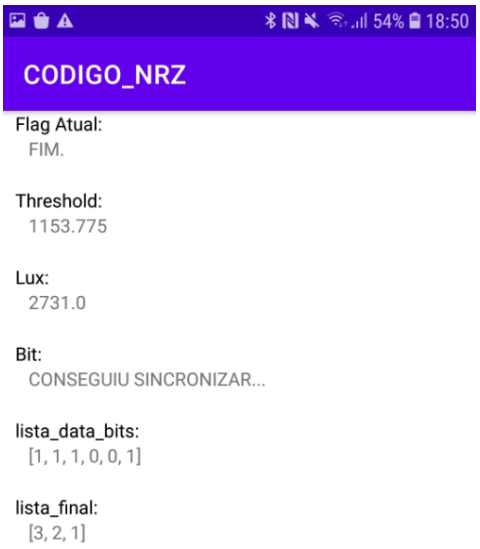
Gráfico 20 – Diagrama de olho



Fonte: Produzido pelo próprio autor.

Como ainda há muito espaço entre os níveis, também é possível inferir que caso se aumente a distância entre os dispositivos da comunicação (reduzir o canal), o limiar de decisão continue bem definido, garantindo a manutenção da qualidade da recepção. Isto também bate com as observações, visto que para até 1,80 m de distância o comportamento do sistema foi semelhante, ainda que com níveis de lux mais baixos. Finalmente, pode-se observar na Figura 58 um exemplo de mensagem em protocolo LabTel-III recebida com o Aplicativo 1.

Figura 58 – Mensagem LabTel-III (App 1)



Fonte: Produzido pelo próprio autor.

6 CONCLUSÕES E TRABALHOS FUTUROS

Este projeto buscou documentar o processo de implementação (em um contexto hospitalar) de um sistema de comunicação PLC-VLC sob caráter experimental. Inicialmente, foram apresentados os equipamentos necessários para a realização do projeto, sendo organizados de forma a demonstrar, de forma clara, as três etapas subsequentes de configurações do sistema proposto. Em seguida, procurou-se explicar cada código utilizado a fim de se oferecer uma melhor compreensão do projeto. Foram utilizadas três formas distintas de transmissão, cada uma utilizando uma biblioteca específica da linguagem de programação *Python* (*Serial*, RPi.GPIO e *Pigpio*). Também foram sugeridos três protocolos distintos para encapsulamento da mensagem, além da transmissão integral de arquivos com extensão HL7. No lado da recepção, já com o *smartphone*, ainda criou-se dois aplicativos em linguagem de programação *Java*. Um para tratar da recepção do sinal em si e o outro para caracterizar o sistema VLC implementado.

Em relação às contribuições deste trabalho, demonstrou-se que é possível utilizar um microcomputador de baixo custo, do tipo *Raspberry Pi*, como um dispositivo transmissor de sinais VLC quando em conjunto com LED's ou relés de estado sólido e lâmpadas. Ainda, que é possível se utilizar um aparelho *smartphone* para receber informação por meio de um sinal luminoso, decodificando-a em tempo real. Outra contribuição do projeto se dá pela demonstração de que aparelhos PLC também podem ser tranquilamente utilizados para agregar distâncias em sistemas que busquem se privar de radiações em frequências mais elevadas, como é a utilizada pelo Wi-Fi.

Algumas das dificuldades encontradas na elaboração do projeto não poderiam deixar de ser a pandemia que hoje assola o país e o mundo, restringindo o acesso aos laboratórios da UFES para recolhimento de equipamentos e realização de testes. Restringindo também as interações entre os membros do projeto PPSUS, sob o qual este trabalho faz parte. As consequências da pandemia ainda dificultaram o acesso às lojas para compra de componentes como a lâmpada e o módulo SSR. Entretanto, também houveram outros fatores que adicionaram complexidade ao trabalho como a ineficiência do aparelho celular em receber as transmissões, sempre com taxas muito baixas. De positivo fica o aprendizado e a experiência adquirida. Apesar de todos os problemas, acredita-se que o projeto foi capaz de cumprir com as propostas iniciais de

desenvolvimento e instalação física de um sistema de comunicação PLC-VLC, cuja recepção fosse implementada diretamente em um aparelho celular convencional, sem qualquer circuito adicional.

Os principais resultados encontrados foram a possibilidade de comunicação entre dispositivos *Raspberry Pi*, em conexão *back-to-back*, em até 200 kbps utilizando biblioteca *Serial* e protocolo UART. Uma comunicação confiável em até 66 kbps utilizando a biblioteca *Pigpio* com qualquer forma desejada de representação do pacote ou da mensagem. Além disso, demonstrou-se que o trecho PLC do sistema encontra-se apto para transmitir e receber com taxas iguais ou acima de 90 Mbps. Ainda, caracterizou-se, corretamente, o brilho da lâmpada de LED em função das distâncias até o dispositivo receptor e de acordo com a teoria fotométrica. Finalmente, foi possível receber a informação, codificada em NRZ e modulada em OOK, de um sinal luminoso enviado a 1 bps, a até 1,8 m de distância.

Como trabalhos futuros, ficam as sugestões de substituição do relé de estado sólido por uma placa mais elaborada (*Bias Tee*) capaz de adicionar um nível CC ao circuito. Fica também o incentivo a se buscar por outras formas de recepção para o *smartphone*, seja com a câmera do celular, seja com a inserção de adaptadores com sensores de luminosidade ambiente externos. Os códigos em si também podem ser aprimorados. No momento em que a parte física garantir bandas de transmissão maiores, sugere-se a troca da codificação NRZ por Manchester, a fim de se melhorar a sincronização e garantir o nível CC tal qual seja desejado. Este código, inclusive, já começou a ser escrito pelo autor do presente projeto. Por fim, sugere-se também o estudo de outros métodos mais complexos de modulação do sinal, a fim de se obter resultados mais otimizados em termos de taxas de transmissão, erros e ruídos.

REFERÊNCIAS BIBLIOGRÁFICAS

ABYZ. **The pigpio library**. 2021. Disponível em: http://abyz.me.uk/rpi/pigpio. Acesso em: 10 jun. 2021.

AMS. **TMD4903**: Color Sensor and Proximity Detection Module with IRBeam Optical Pattern Generator. 2015. Disponível em: https://ams.com/documents/20143/36005/TMD4903_FS000143_2-00.pdf/c9f63c52-a71b-545f-2d46-796572dabeb0. Acesso em: 10 jun. 2021.

AURION. **P15, P12 e P10**: Monitores de Sinais Vitais Multiparamétricos. 2013. Disponível em: http://www.aurion.com.br/_catalogos/monitores.pdf. Acesso em: 10 jun. 2021.

AVANT. **Catálogo Avant 2021**. 2021. Disponível em: https://avantlux.com.br/wp-content/uploads/2021/04/Catalogo-Concept.pdf. Acesso em: 10 jun. 2021.

BREED, G. Bit Error Rate: Fundamental Concepts and Measurement Issues. **High Frequency Electronics**, [*s.l.*], v. 2, n. 1, p. 46-48, jan. 2003. Disponível em: https://www.highfrequencyelectronics.com/Jan03/HFE0103_Tutorial.pdf. Acesso em: 10 jun. 2021.

CARISTIX. **HL7-Definition V2**. 2021. Disponível em: https://hl7-definition.caristix.com/v2/. Acesso em: 10 jun. 2021.

CASELLA, I. R. S.; ANPALAGAN, A. **Power Line Communication Systems for Smart Grids**. 1. ed. London: The Institution of Engineering and Technology, 2019.

ÇELEBI, H. B. **Noise and Multipath Characteristics of Power Line Communication Channels**. 2010. Dissertação (Master of Science in Electrical Engineering) – Department of Electrical Engineering, College of Engineering, University of South Florida, Tampa, 2010.

CREATIVENTECHNO. **How LED Works?**. 2012. Disponível em: https://creativentechno.wordpress.com/2012/01/07/how-led-works. Acesso em: 10 jun. 2021.

DECUSATIS C. **Handbook of Fiber Optic Data Communication**: A Practical Guide to Optical Networking. 3. ed. Orlando, Florida: Academic Press, 2008.

GHASSEMLOOY, Z.; ALVES L. N.; ZVÁNOVEC S.; KHALIGHI M. A. **Visible Light Communications**: Theory and Applications. 1. ed. Boca Raton, Florida: CRC Press, 2019.

GUPTA, S. D. **Light Emitting Diodes for Agriculture**: Smart Lighting. 1. ed. Singapore: Springer, 2017.

IETF DATATRACKER. **SSH File Transfer Protocol**. 2006. Disponível em: https://datatracker.ietf.org/doc/html/draft-ietf-secsh-filexfer-13. Acesso em: 10 jun. 2021.

JANI, M.; GARG, P.; GUPTA, A. Performance Analysis of a Mixed Cooperative PLC-VLC System for Indoor Communication Systems. **IEEE Systems Journal**, [*s. l.*], v. 14, n. 1, p. 469-476, mar. 2020.

LAPINSKY, S. E.; EASTY, A. C. Electromagnetic interference in critical care. **Journal of Critical Care**, [*s. l.*], v. 21, n. 3, p. 267-270, jan. mar. 2006.

LATCHMAN, H. A.; KATAR, S.; YONGE III, L. W.; GAVETTE, S. **HomePlug AV and IEEE 1901**: A Handbook for PLC Designers and Users. 1. ed. New Jersey: John Wiley & Sons Inc., 2013.

LIU, H.; ZHU P.; CHEN Y.; HUANG M. Power Allocation for Downlink Hybrid Power Line and Visible Light Communication System. **IEEE Access**, [*s. l.*], v. 8, p. 24145-24152, fev. 2020.

LOJA DA HORA. **Módulo Relé de Estado Sólido de 4 Canais para Arduino**: Descrição. 2021. Disponível em: https://www.lojadahora.com.br/modulo-rele-de-estado-solido-ssr-de-4-canais-para-arduino. Acesso em: 10 jun. 2021.

MACKAY, S; WRIGHT, E.; REYNDERS, D.; PARK, J. **Industrial Data Networks**: Design, Installation and Troubleshooting. 1. ed. Oxford: Newnes, 2004.

MEDIUM. **Raspberry Pi 3 GPIO Header**. 2016. Disponível em: https://medium.com/@rxseger/i2c-interfacing-on-the-bus-pirate-and-raspberry-pi-to-serial-eeproms-for-a-hat-c776ea50f8f4. Acesso em: 10 jun. 2021.

MICROCHIP. **TB3156**: Universal Asynchronous Receiver Transmitter (UART) on 8-Bit PIC Microcontrollers. 2017. Disponível em: http://ww1.microchip.com/downloads/en/AppNotes/TB3156-Univ-Asynchronous-Receiver-Transmitter-on-8bit-PIC-90003156A.pdf. Acesso em: 10 jun. 2021.

MOVCHAN, A. K.; ROGOZHNIKOV E. V.; DMITRIYEV E. M.; POKAMESTOV D. A.; PETROVSKIY K. V. Approach to Implementation Full-duplex Communication Technology in Power Line Communication Systems. *In*: INTERNATIONAL CONFERENCE OF YOUNG SPECIALISTS ON MICRO/NANOTECHNOLOGIES AND ELECTRON DEVICES (EDM), 21., 2020, Chemal. **Proceedings** [...]. Chemal: IEEE, 2020. v. 1, p. 112-117.

NASSAR, C. **Telecommunications Demystified**: A Streamlined Course in Digital Communications (and some Analog) for EE Students and Practicing Engineers. 2. ed. Eagle Rock, Virginia: LLH Technology Publishing, 2001.

NEVES, J. L. C. **Adequação ao Requisito Cintilação em Sistemas de Comunicação via Luz Visível Mediante uso de Sinais OFDM com Envoltória Constante**. 2008. Dissertação (Mestrado em Engenharia Elétrica) – Programa de Pós-Graduação em Engenharia Elétrica, Centro Tecnológico, Universidade Federal do Espírito Santo, Vitória, 2008.

OFFICE OF THE ASSISTANT SECRETARY FOR PLANNING AND EVALUATION. **Standards Organizations for the NHII**: Health Level 7 (HL7). 2003. Disponível em: https://aspe.hhs.gov/standards-organizations-nhii. Acesso em: 10 jun. 2021.

OMROM-RUSSIA. **OMRON Solid State Relay**: G3MB. 2017. Disponível em: http://www.omron-russia.com/doc/relay/ssr/g3mb.pdf. Acesso em: 10 jun. 2021.

PARAMIKO. **Welcome to Paramiko!**. 2021. Disponível em: https://www.paramiko.org. Acesso em: 10 jun. 2021.

PATHAK, P. H.; FENG, X.; HU, P.; MOHAPATRA, P. Visible Light Communication, Networking, and Sensing: A Survey, Potential and Challenges. **IEEE Communications Surveys & Tutorial**s, [*s. l.*], v. 17, n. 4, p. 2047-2077, jan. 2015.

PERIPH IO. **Reaching 80 MHz**: How we cranked up GPIO performance in v2.10. 2017. Disponível em: https://periph.io/news/2017/gpio_perf. Acesso em: 10 jun. 2021.

PYPI. **RPi.GPIO 0.7.0**: A module to control Raspberry PI GPIO channels. 2019. Disponível em: https://pypi.org/project/RPi.GPIO. Acesso em: 10 jun. 2021.

PYSERIAL. **PySerial**: Read the Docs. 2017. Disponível em: https://pyserial.readthedocs.io/en/latest/pyserial.html. Acesso em: 10 jun. 2021.

RAMASWAMI, R.; SILVARAJAN, K. N.; SASAKI, G. H. **Optical Networks**: A Practical Perspective. 3. ed. Burlington, Massachusetts: Morgan Kaufmann Publishers, 2000.

RASPBERRY PI. **What is a Raspberry Pi?**. 2015. Disponível em: https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi. Acesso em: 10 jun. 2021.

_____. **Raspberry Pi 3 Model B**: Single-board computer with wireless LAN and Bluetooth connectivity. 2021a. Disponível em: https://www.raspberrypi.org/products/raspberry-pi-3-model-b. Acesso em: 10 jun. 2021.

_____. **GPIO pads control**. 2021b. Disponível em: https://www.raspberrypi.org/documentation/hardware/raspberrypi/gpio/gpio_pads_control.md . Acesso em: 10 jun. 2021.

_____. **GPIO**. 2021c. Disponível em: https://www.raspberrypi.org/documentation/hardware/raspberrypi/gpio/README.md. Acesso em: 10 jun. 2021.

RHEE, J.; RHEE, E.; PARK, J. Electromagnetic interferences caused by power line communications in the HF bands. *In*: IEEE INTERNATIONAL SYMPOSIUM ON POWER LINE COMMUNICATIONS AND ITS APPLICATIONS, 2008, Jeju. **Proceedings** [...]. Jeju Island: IEEE, 2008. v. 1, p. 249-252.

SAINSMART. **2/4/8-Ch 5V Solid State Relay**. 2017. Disponível em: https://www.sainsmart.com/products/2-4-8-ch-5v-solid-state-relay. Acesso em: 10 jun. 2021.

SAMSUNG. **Specifications**: Galaxy S7 Single Sim. 2017. Disponível em: https://www.samsung.com/uk/smartphones/galaxy-s7/more. Acesso em: 10 jun. 2021.

SCHUBERT, E. F. **Light-Emitting Diodes**. 2. ed. New York: Cambridge University Press, 2006.

SHORTLIFFE, E. H.; PERREAULT L. E.; WIEDERHOLD G.; FAGAN L. M. **Medical Informatics**: Computer Applications in Health Care and Biomedicine. 2. ed. New York: Springer-Verlag New York Inc, 2001.

SILBERBERG J. L. Performance Degradation of Electronic Medical Devices Due to Electromagnetic Interference. **Compliance Engineering**, [*s. l.*], v. 1, n. 10, p. 25-39, jan. 1993.

SKLAR, B. **Digital Communications**: Fundamentals and Applications. 2. ed. New Jersey: Prentice Hall P T R, 2001.

SOLWISE. **HomePlug AV White Paper**. 2005. Disponível em: https://www.solwise.co.uk/downloads/files/hpav-white-paper_050818.pdf. Acesso em: 17 maio 2021.

TANAKA, Y.; HARUYAMA, S.; NAKAGAWA, M. Wireless Optical Transmissions with White colored LED for Wireless Home Links. *In*: IEEE INTERNATIONAL SYMPOSIUM ON PERSONAL INDOOR AND MOBILE RADIO COMMUNICATIONS (PIMRC), 11., 2000, London. **Proceedings** [...]. London: IEEE, 2000. v. 2, p. 1325-1329.

TP-LINK. **300Mbps AV500 WiFi Powerline Extender Starter Kit**: TL-WPA4220KIT. 2016. Disponível em: https://static.tp-link.com/res/down/doc/TL-WPA4220KIT(EU)_1.0.pdf. Acesso em: 10 jun. 2021.

_____. **AV500 Nano Powerline Adapter Starter Kit**: TL-PA4010KIT. 2017. Disponível em: https://static.tp-link.com/TL-PA4010KIT_V1.0_Datasheet.pdf. Acesso em: 10 jun. 2021.

_____. **Explicação sobre parâmetros de velocidade dos produtos TP-Link**: Q&A of functional explanation or specification parameters. 2020. Disponível em: https://www.tp-link.com/br/support/faq/2866/. Acesso em: 10 jun. 2021.

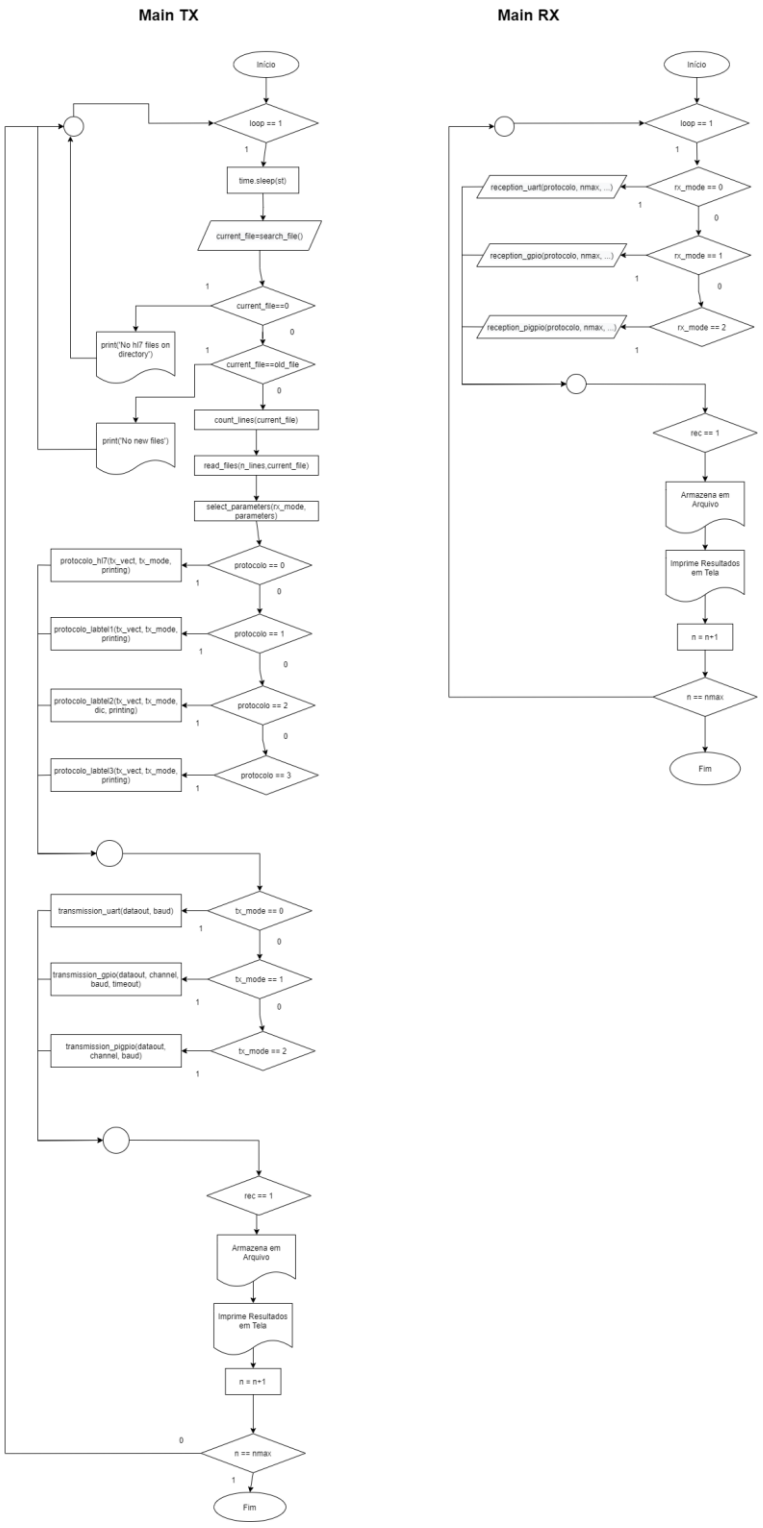
UYSAL, M.; CAPSONI, C.; GHASSEMLOOY, Z.; BOUCOUVALAS, A.; UDVARY, E. **Optical Wireless Communications**: An Emerging Technology. 1. ed. Cham: Springer International Publishing, 2016.

WIRESHARK. **About Wireshark**. 2021. Disponível em: https://www.wireshark.org/. Acesso em: 02 jul. 2021

WP ENGINE. **How to Access WordPress Files With SFTP**: How Does SFTP Work?. 2017. Disponível em: https://wpengine.com/resources/how-to-access-wordpress-files-using-sftp/. Acesso em: 10 jun. 2021.

APÊNDICE A – FLUXOGRAMAS DOS PROGRAMAS PRINCIPAIS

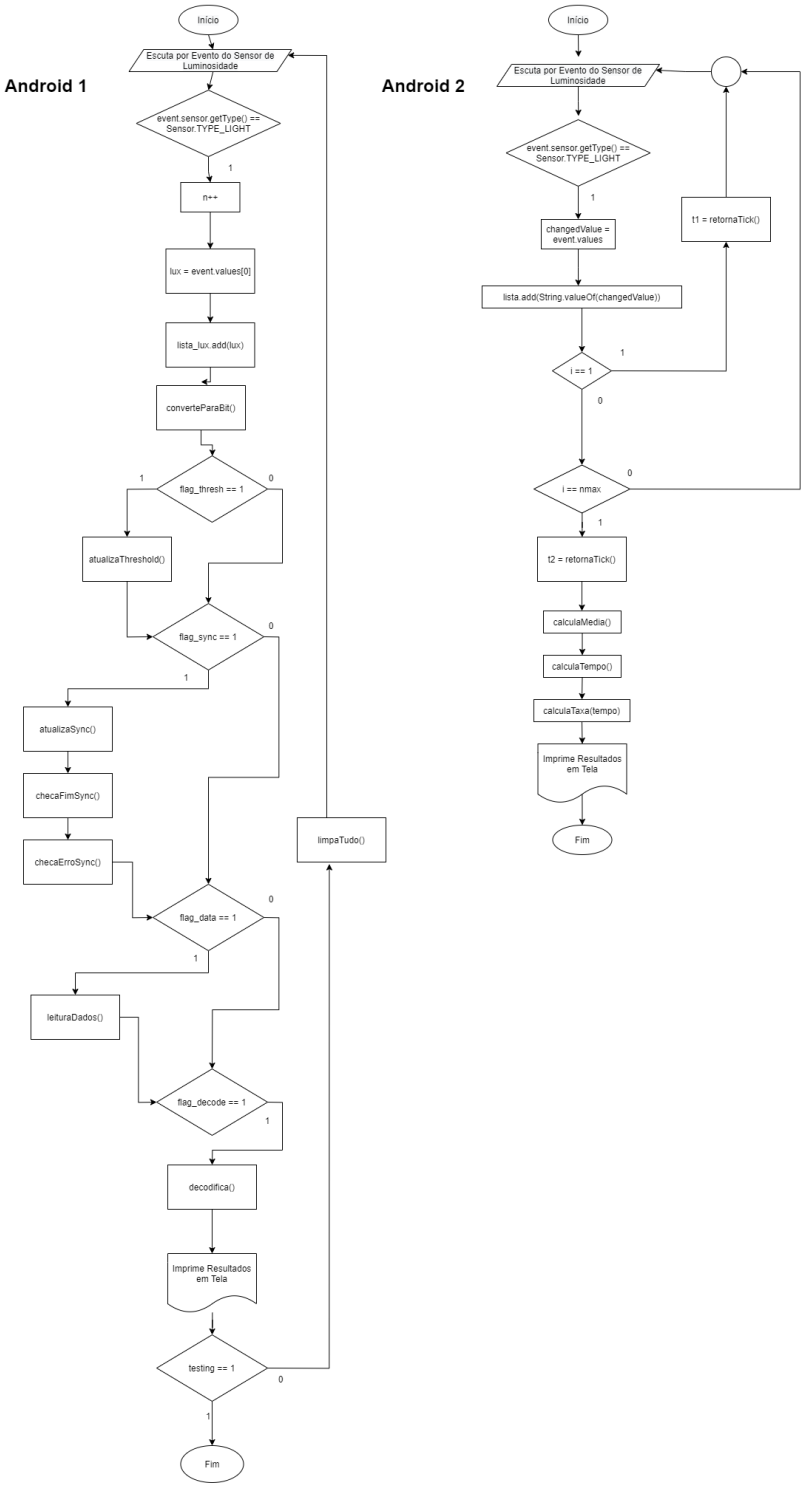
Figura 59 – Fluxogramas dos programas principais do RPi



Fonte: Produzido pelo próprio autor.

APÊNDICE B – FLUXOGRAMAS DOS APLICATIVOS

Figura 60 – Fluxogramas dos aplicativos *android*

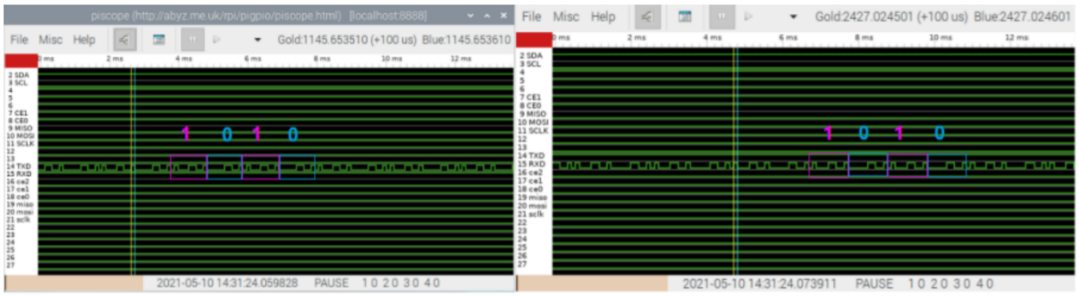


Fonte: Produzido pelo próprio autor.

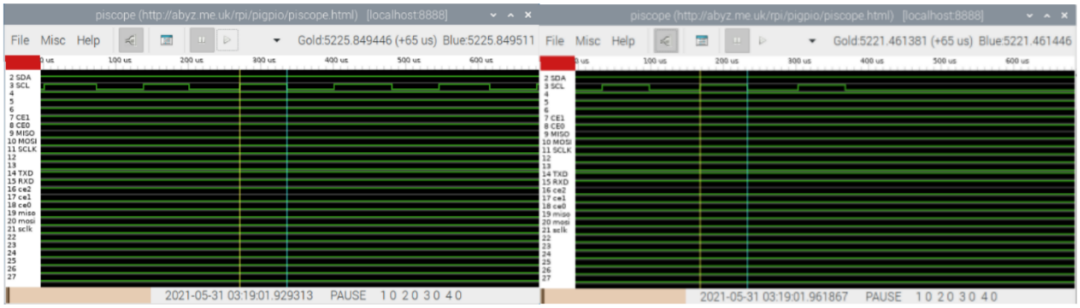
APÊNDICE C – COMPARAÇÃO DOS SINAIS (TX) E (RX)

Figura 61 – Comparação dos sinais transmitidos e recebidos

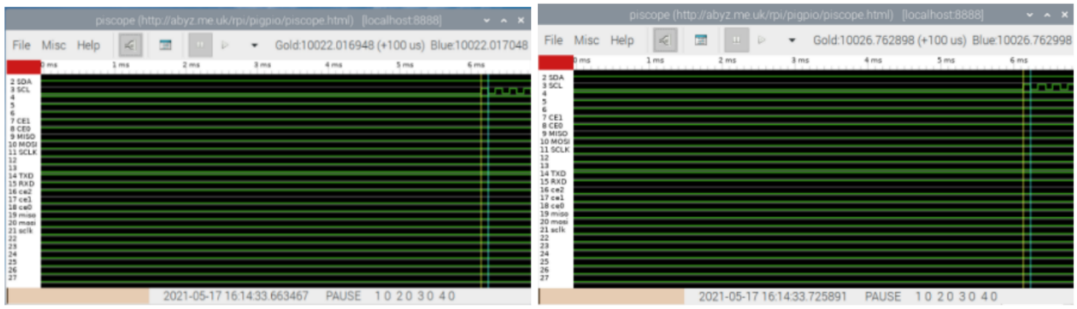
Serial (UART)



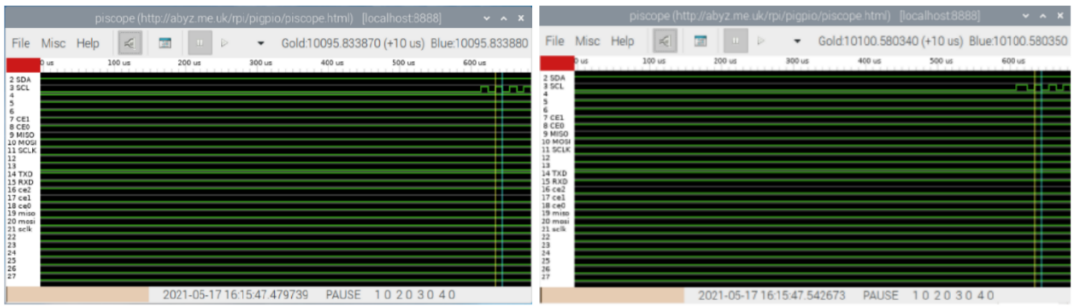
RPi.GPIO



Pigpio (10 kbps)



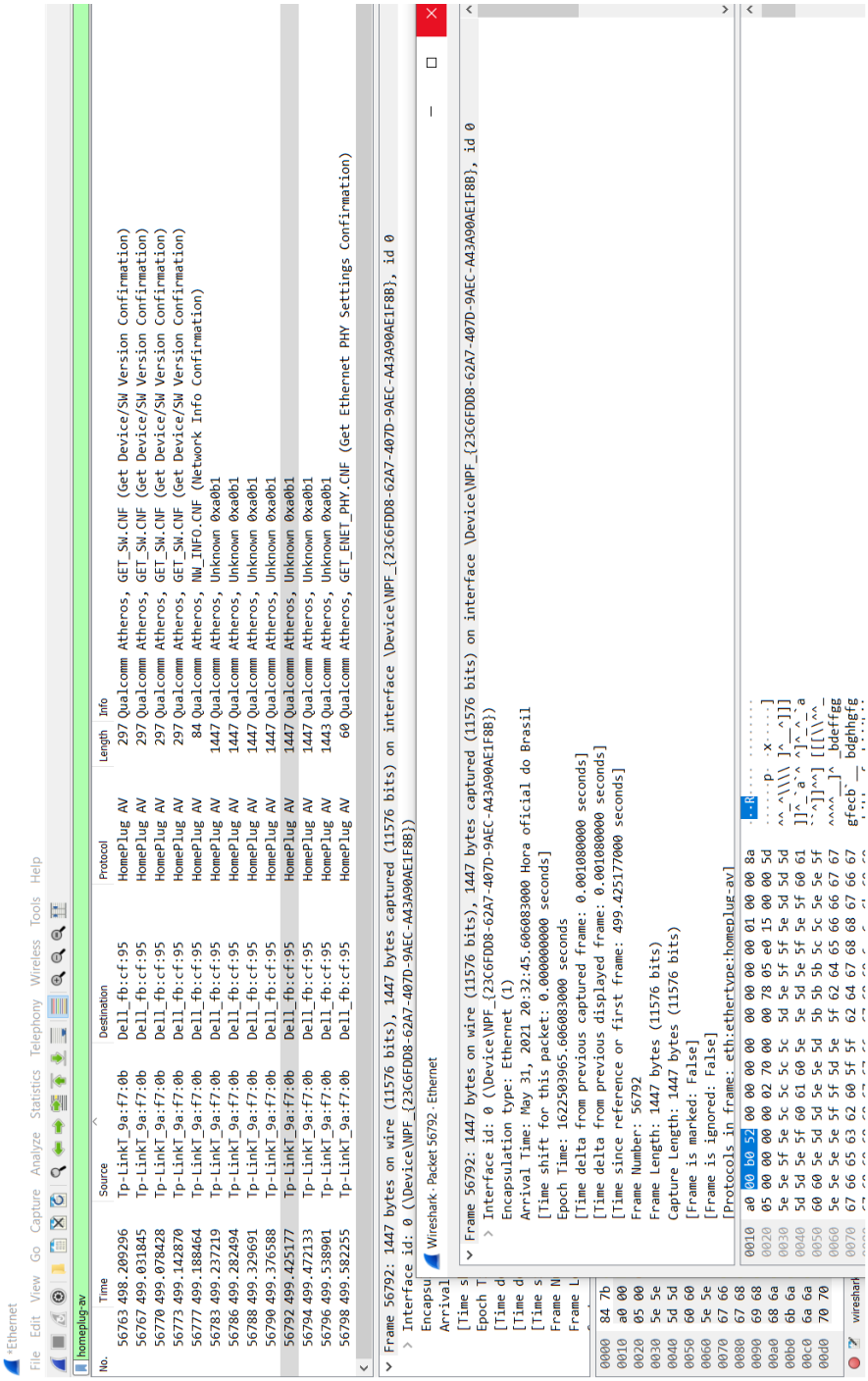
Pigpio (100 kbps)



Fonte: Produzido pelo próprio autor.

APÊNDICE D – CAPTURA DE TELA COM O PACOTE DO PLC

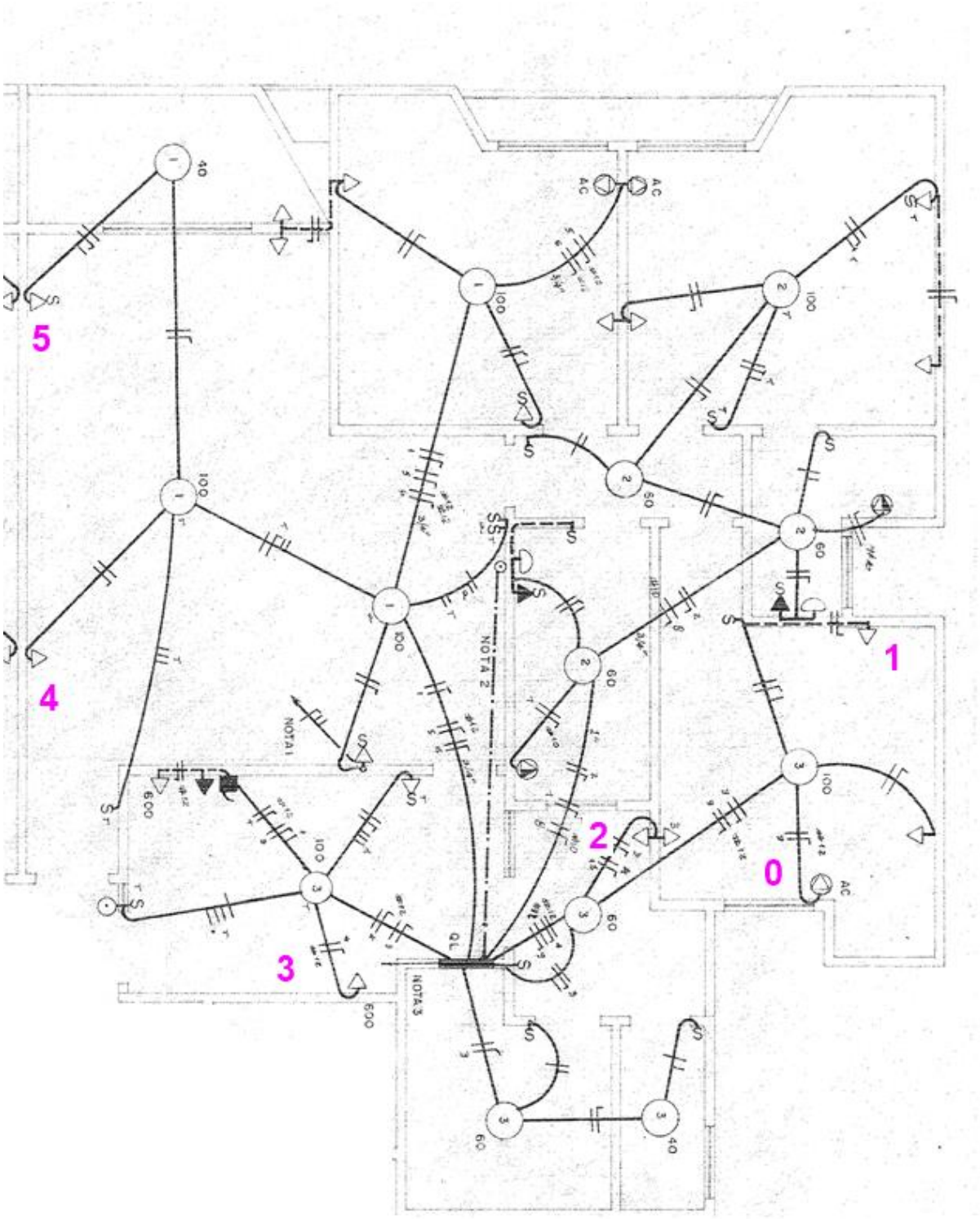
Figura 62 – Captura de tela com um pacote do PLC



Fonte: Produzido pelo próprio autor.

APÊNDICE E – PLANTA ELÉTRICA DO AMBIENTE TESTADO

Figura 63 – Planta elétrica do ambiente testado



Fonte: Produzido pelo próprio autor.