Sistema de Monitoramento de ECG Conectado por IOT

Gabriel Martins Ribeiro, 140020667
Engenharia Eletrônica
UnB - Faculdade do Gama (FGA)
Email: gabrielmartinsgmr.10@gmail.com

Fabricio de Matos Alves, 202028196
Engenharia Eletrônica
UnB - Faculdade do Gama (FGA)
Email: fabriciomatos02@gmail.com

I. Introdução

O monitoramento de pacientes que possuem diagnóstico de problemas cardíacos ou têm a suspeita de desenvolvê-los, precisa ser feito de forma recorrente. Esse sistema busca minimizar esse problema através do monitoramento do sinal de ECG com sensores conectados na pele dos pacientes, podendo ser usado na casa do paciente a fim de evitar o contato social do paciente em um hospital.

Segundo a Organização Mundial de Saúde (OMS) as doenças cardiovasculares (DVC) são a principal causa de mortes no mundo, com aproximadamente 17,9 milhões de vidas perdidas no ano de 2019, número este que representa cerca de 32% de todas as mortes registradas no mundo. Ainda segundo a Organização, cerca de 75% das mortes causadas por DVCs ocorrem em países pobres ou em desenvolvimento devido a falta de assistência médica primária para a detecção dessas doenças ainda em estágios iniciais, sendo as populações mais pobres as mais afetadas. No Brasil, segundo o Ministério da Saúde, cerca 300 mil pessoas por ano são acometidas por casos de Infarto Agudo do Miocárdio, sendo destes, 30% levam a casos de morte, e é estimado que haverá aumento de casos de até 250% até 2040.

Segundo estudo realizado para o ano de 2015, os custos estimados relacionados a tratamento clínicos e tratamentos cirúrgicos de eventos de DVC na rede SUS é da ordem de R$ 5.1 bilhões, além dos custos na ordem de R$ 700 milhões com órteses, próteses e materiais especiais.

Como forma de classificar doenças e problemas relacionados a saúde, a OMS elaborou a CID (Classificação Estatística Internacional de Doenças e Problemas Relacionados com a Saúde) que em sua décima versão capítulo IX CID-10:IX agrupa as chamadas ”Doenças do aparelho circulatório”que são codificadas com a letra I seguida de dois números. Doenças como Doenças Reumáticas de Valva (I05, I06 e I07), Doença Cardíaca Hipertensiva (I11), Infarto Agudo do Miocárdio (I22), Pericardite Aguda (I30), Hipotensão (I95), entre outras. Pacientes com tais doenças podem se beneficiar de um monitoramento eficaz e à distância de ECG junto do acompanhamento de médico especialista, a fim de obter tratamento adequado com antecedência.

II. Fundamentação Teórica

Eletrocardiografia (ECG) é o processo de aquisição de atividade elétrica do coração por um período de tempo usando eletrodos posicionados em contato com a pele do paciente. O ECG é considerado o melhor método para detecção de anormalidades cardíacas, onde os eletrodos detectam pequenas mudanças elétricas na pele provenientes dos padrões eletrofisiológicos de despolarização e repolarização durante cada batimento cardíaco.

Para o ECG de 12 canais, dez eletrodos são posicionados nos membros do paciente e na superfície do peito. A magnitude do potencial elétrico é medido em 12 ângulos diferentes e é gravado por um período de tempo (geralmente 10 segundos). Existem 3 componentes principais em um ECG : a onda P, que representa a despolarização do átrio; o complexo QRS, que representa a despolarização dos ventrículos; e intervalo QT, que representa a duração total da sístole (período de esvaziamento do ventrículo) elétrica ventricular. Além destes, outros pontos/ondas são observáveis em um registro de ECG, como:

- P representa o pulso de contração do átrio;
- Q representa a deflexão para baixo imediatamente antes da contração ventricular;
- R representa o pico da contração ventricular;
- S representa a deflexão para baixo imediatamente depois da contração ventricular;
- T representa a recuperação dos ventrículos;
- U representa o sucessor da onda T, mas é pequeno e quase nunca observado

Durante cada batimento cardíaco, um coração saudável tem uma progressão ordenada de despolarização que começa com “células marcapassos”no nó sinoatrial, se espalha pelo átrio, passa pelo nó sinoatrial até o pacote de HIS (anterior à bifurcação que leva aos ramos direito e esquerdo) e dentro das fibras de Purkinje, se espalhando para baixo e para a esquerda pelos ventrículos.

Na Figura 2 é apresentado o diagrama de blocos que representa o sistema do equipamento proposto de aquisição de ECG. A ideia principal do bloco Sensor de ECG é composto por sensores cardíacos posicionados na pele do paciente. O bloco Arduino é responsável pela aquisição e condicionamento

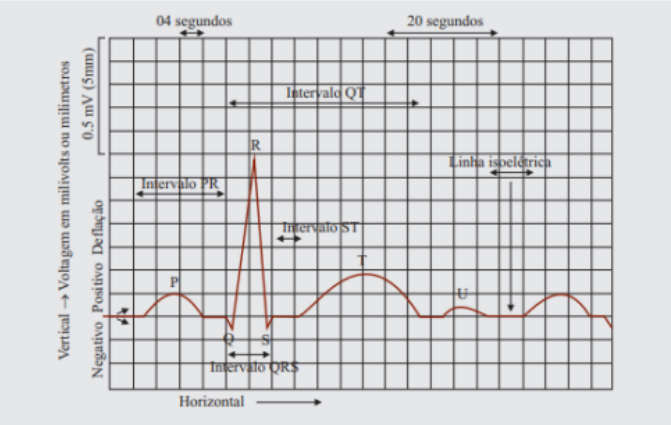

Figura 1: Caracterização do registro ECG. Fonte: ECG: manual prático do eletrocardiograma.

do sinal a partir do seu conversor A/D com resolução de 10 bits com clock podendo variar entre 50 kHz e 200 kHz para tal resolução. O bloco Raspberry Pi 3 modelo B é responsável por energizar o sistema de aquisição e apresentar o sinal registrado nos periféricos como computador, nuvem IOT e em um Website.

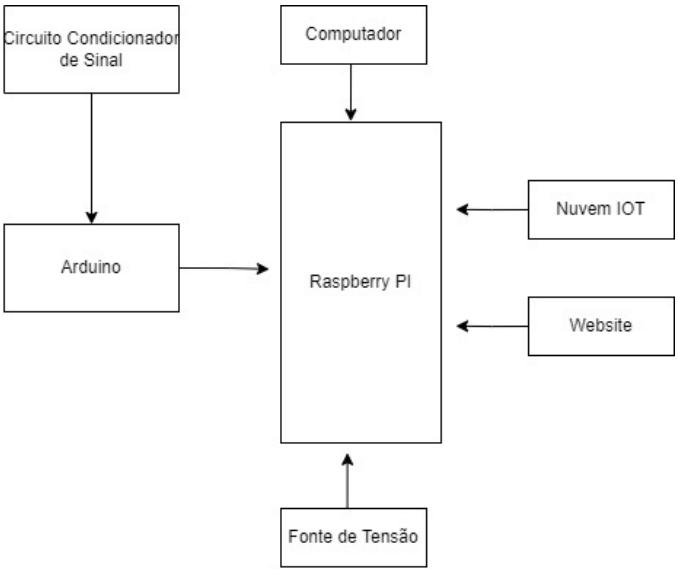

Figura 2: Diagrama geral proposto. Fonte: Autoria própria.

A. *Medição de ritmo cardíaco (bpm)*

A medição de parametros de ritmo cardíaco encontra-se presente em diversas cenários. Para detecção de episódio de tarquicardia, por exemplo, é verificado se o ritmo cardiaco encontra-se acima de 100 bpm (batimentos por minutos). Com a informação do ritmo cadíaco do paciente, o profissional de saúde pode fazer intervenção mais rápidas e precisas, permitindo com quem mais vidas sejam salvas e doenças diagnosticadas. Atletas usam valores de bpm para monitorar o condicionamento físico, considerando que o ritmo cardíaco está diretamente relacionado a capacidade do músculo cardíaco de realizar o bombeamento sanguíneo para os átrios.

O valor de pico do sinal de ECG encontra-se dentro dos limites do intervalo QRS, mais especificamente no instante R. Esse ponto do ECG representa o pico de contração ventricular. A distancia entre 2 valores de pico do sinal de ECG configura o intervao-R-R, logo, o número de valores de pico R em 1 minuto. Precisa de um minuto completo para um único valor de rítmo cardíaco aparecer. Porém, também estima-se o ritmo pela fórmula:

$$HR = \frac{60}{IntervaloRR} \quad (1)$$

Pode parecer muito rápido, mas não está correto para diagnóstico clínico, pois assume que cada intervalo R-R pelo 1 minuto completo é o mesmo. Para um ser humano médio saudável, o tempo levado para um 1 batimento completo está entre 800-850 ms (65-75 batimentos por minuto). Esse valor pode variar caso a pessoa esteja descansando ou trabalhando. Independentemente da situação, os intervalos de batimento cardíaco não variam muito. Mas para uma pessoa com fibrilação atrial ou arritimia varia significamente durante o ECG, independentemente da média de batimentos estar parecendo ser normal. O parametro para identificar isso chama-se Variabilidade de Ritmo Cardíaco.

III. Protótipo Funcional

- Esteremos usando o raspberry pi model 3B, pois atende os requisitos do nosso projeto, não terá problema de consumo de energia, pois o nosso sistema só funcionará conectado na rede elétrica. Usaremos um arduino UNO para obtenção e processamento dos sinais ECG. Além disso, o sistema não possuirá interface gráfica, pois as informações serão enviadas para outro servidor (clinica/medico).
- O sistema operacional utilizado será o Windows 10 Home com o software Ubuntu for Windows que é suficiente para acessar os aplicativos e bibliotecas para o projeto, não sendo necessário a instalação de outro sistema operacional. O sistema da raspberry será o Raspberry OS (Raspbian), não será utilizado desktop, pois não terá interface gráfica.
- A instalação do sistema operacional foi feita, primeiramente, baixando o sistema no site da raspberry (https://www.raspberrypi.com/software/operating-systems/), foi instalado o sistema ”Raspberry Pi OS with desktop and recommended software”, após descompactar o arquivo baixado, foi usado um software que grava um arquivo de imagem de disco em um dispositivo removível (SD CARD) (https://sourceforge.net/projects/win32diskimager/). Depois, o cartão SD foi conectado a raspberry e conectando-a a energia, o sistema foi instalado automaticamente.
- A Raspberry foi conectada via cabo ethernet ao modem de internet para que, por meio do PC, pudesse obter o IP da raspberry, foi utilizado o seguinte software : https://www.advanced-ip-scanner.com/br/. A conexão foi feita por meio do

software PUTTY (https://www.putty.org/) a partir do SSH. Nas configurações da rapsberry o SSH e o VNC foram habilitados para acesso remoto via cabo ethernet e wi-fi no PC. No caso do projeto, a raspberry pi está conectada a rede local de internet e o acesso é via wi-fi.

- Quanto ao circuito condicionador de sinal, utilizou-se o módulo AD8232. Esse módulo adota um amplificador operacional usado para construir um filtro para eliminar o ruído. Ele implementa um filtro passa-alta de dois pólos pala eliminar ruido por movimento do paciente. Conecta-se, também, 3 eletrodos na pele do paciente.
- A ferramenata de nuvem utilizada foi Ubidots

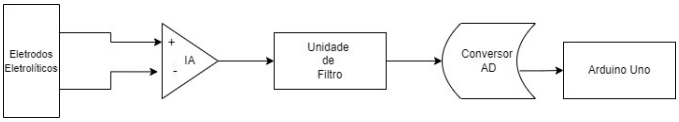

Figura 3: Circuito condicionador de sinal. Fonte: autoria própria

IV. Desenvolvimento

Dentre os requisitos do sistema pode-se destacar:

- Baixo consumo de enrgia
- Alta acurácia
- Confiabilidade
- Tamanho pequeno

Os esquemático do projeto está mostrado na figura em anexo, está caractrizando a conexão por meio da comunicação serial UART entre o arduino UNO e Raspberry PI, além disso, a conexão do módulo AD8232 que conecta os eletrodos com o arduino UNO que fará a leitura e transmissão dos dados.

Os dados lidos pela Raspberry PI serão enviandos para uma aplicação IoT, gratuita e disponivel, chamada Ubidots. A plataforma é uma API, onde os dados enviados pela raspberry PI se tornam visíveis através de gráficos atualizados em tempo real, sendo assim, possivel visualizar o sinal ECG do paciente.

Figura 4: Interface inicial da plataforma Ubidots.

A. Testes de Aquisição do Sinal

A aplicação do sensor é feita conforme a figura 6, onde os eletrodos são distribuídos em posições estratégicas e definidas para uma boa captação do sinal. Quanto mais perto do coração estiverem as almofadas, melhor será a medição. Os cabos são codificados por cores para ajudar a identificar o posicionamento correto.

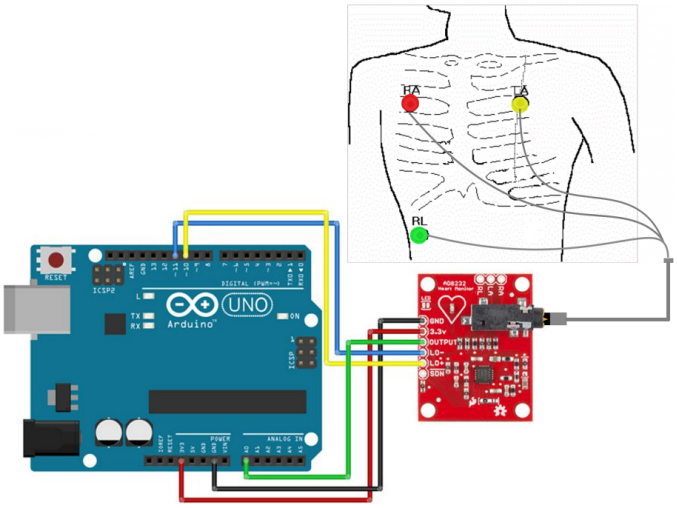

Figura 5: Módulo AD8232, Arduino Uno e ligações dos eletrodos no corpo humano.

O código do arduino, mostrado em anexo, que faz a aquisição dos dados do módulo sensor AD8232 é estruturado da seguinte forma, a função setup configura porta serial par um baud rate de 115200 bts, que é a velocidade de transmissão de dados, os pinos 2 e 3 do arduino são configurados para conectar aos Leads off positivo e negativo do módulo. Na função loop, é verificado se os pinos 2 e 3 estão em nivel lógico alto, se sim, indica que o eletrodo está desligado, portanto, não há medição. Caso contrário, o valor lido é da porta analógica A0 é mostrado na tela. A taxa de leitura dos dados está em 500 Hz, observando que ao faixa de frequência dos sinais de ECG está entre 0,05 a 150 Hz. Desta forma, taxa de amostragem do sinal sendo 500 Hz está de dentro do critério de Nyquist, não havando distorção ou perda de informação na reconstrução

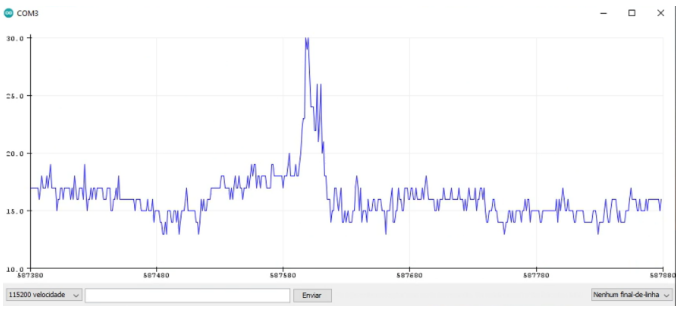

Figura 6: Signal obtido pelo plotter serial do arduino.

B. Testes de Comunicação

A comunicação do arduino UNO com a raspberry PI foi feita através dos GPIO de transmissão e recepção dos

dispositivos por meio da comunicação serial UART, onde foi realizado um teste para envio de dados do arduino para a raspberry. Na figura 14 mostra as conexões realizadas, o GPIO 14 (TX) da raspberry é conectado diretamente ao pino 0 (RX) do arduino, já o GPIO 15 (RX) é conectado por um divisor de tensão para 3,3 V que vem do pino 1 (TX) do arduino.

No código em C da raspberry Pi é utilizado as bibliotecas *unistd*, *fcntl* e *termios* para a configuração da comunicação UART. Nas configurações tem-se o baud rate da comunicação sendo de 115200 e a utilização de 8 bits para informação. Após as configurações é criado um buffer para receber os blocos de bytes, formando uma string com os valores lidos. A função *read* é responsável por ler esses bytes e salvar no buffer de string. Após a lida dos valores recebidos, a string é convertida em número inteiro para que possa ser enviada ao codigo em python *rasp_to_ubidots.py*. O comando *system* executa o no sistema o codigo python com o valor enviado através do argumento. Esse comando está no loop infinito com a função de leitura de dados, ou seja, a cada leitura o código é executado e o dado será enviando ao Ubidots. No entanto, para a etapa final o sinal ECG não será enviado para o ubidots, devido a taxa de upload não ser suficiente para representar o sinal em tempor real. Nesse sentido, o sinal ECG é salvo em um arquivo *.xlsx* com um tamanho suficiente para 30000 amostras, ou seja, cerca de 60 segundos de sinal ECG. O monitoramento dos batimentos cardíaco através da análise do sinal ECG será feito, e os dados do batimento cardíaco serão envidados ao ubidots para monitoramento remoto.

No código em python, a função *sys.argv* ler o argumento enviado do código em C e realiza o *post_request* que faz uma requisição HTTP para o servidor da Ubidots, para isso teve que importar a biblioteca *requets*. Nessa função é criada a requisição HTTP, onde é necessário o token e nome da variavel criada na plataforma Ubidots, essas informações são concedidas pela plataforma para serem incorporadas ao código e realizar a comunicação com sucesso para o envio dos dados em sequência. A função retorna *true* se a requisição for feita com sucesso e *false* se houve alguma falha de comunicação. E na rotina principal *main*, a função *receive_data* é chamada com nome da variável configurada no site da Ubidots. O Ubidots recebe os dados dos sinal ECG / batimentos cardíacos e monta um gráfico que pode ser visualizado na tela e que se atualiza a cada envio de dado.

V. RESULTADOS

A. *Testes com Gerador de Onda Quadrada*

Tendo em vista que os eletrodos são de materiais descartáveis e o seu uso é limitado em aplicações, isso significa que a cada uso há um desgaste considerável. Nesse sentido, para dá continuidades aos testes finais, foi utilizado um gerador de onda quadrada como uma fonte de um sinal analógico, sendo assim, possível realizar maiores quantidades de testes. O gerador é descrito na figura 7 a seguir, sendo utilizado um CI LM555, configurando como oscilador.

O resultado do teste feito com o gerador de onda quadrada está na figura 9 a seguir, onde a quanda quadrada gerada é lida pelo arduino e transmitida para raspberry que faz a comunicação com o Ubidots, enviando os dados em tempo real, com alguns atrasos da ordem dos milissegundos, na figura 9

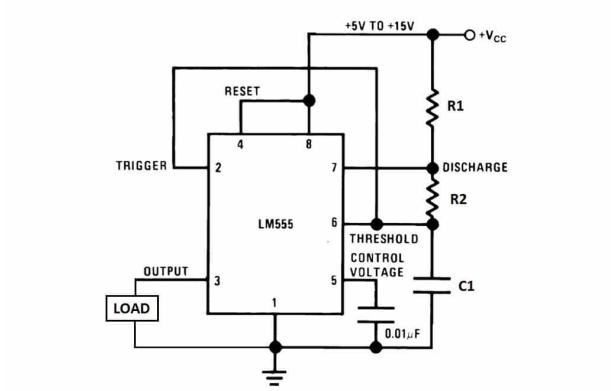

Figura 7: Circuito gerador de onda quadrada.

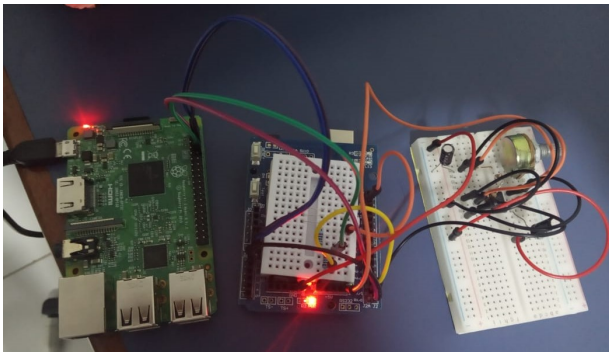

Figura 8: Conexão do circuito gerador de onda quadrada com o arduino e a raspberry PI.

é possivel notar que o sinal da onda quadrada é representada de forma clara, sem grandes distorções, o que caracteriza o sucesso no envio dos dados e na comunicação com a API do Ubidots.

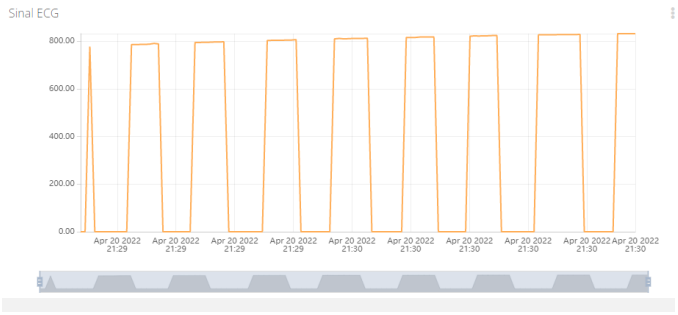

Figura 9: Gráfico do sinal do gerador de onda quadrada enviado da raspberry PI para o dashboard do Ubidots.

B. *Testes com o sensor AD8232*

Os testes foram feitos com o sensor AD8232 conectado ao corpo humano, conforme o esquemático da figura 5. Os dispositivos devidamente conectados estão mostrados na figura 10 a seguir.

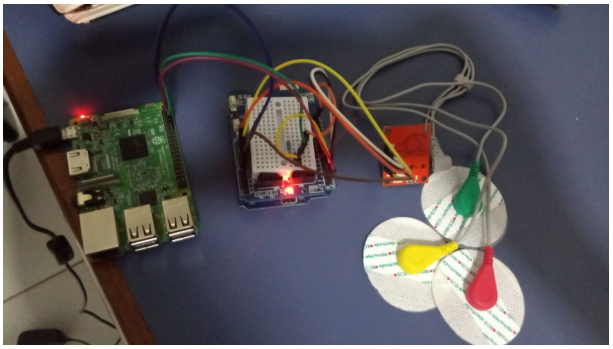

Figura 10: Conexão do circuito do sensor AD8232 com o arduino e a raspberry PI.

O sinal obtido no dashboard do Ubidots é mostrado na figura 11 a seguir.

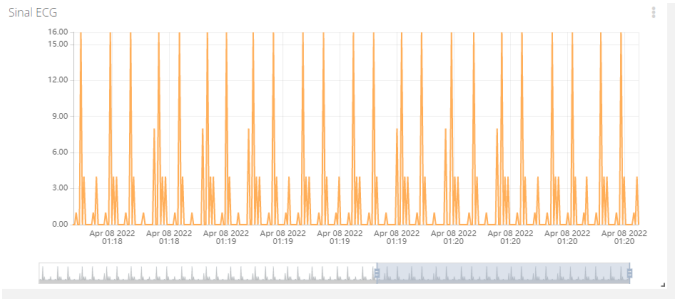

Figura 11: Gráfico do sinal ECG enviado da raspberry PI para o dashboard do Ubidots.

Nota-se que o sinal obtido possui algumas caracteristicas do sinal ECG, porém, devido ao tempo de updload dos dados no Ubidots, o sinal está claramente distorcido. Devido as dificuldades de obter o sinal ECG de forma clara no ubidots, optou-se por representar os batimentos cardíacos em por minuto, sendo calculado através do sinal. No caso dos batimentos cardíacos a latência não interfere significativamente na representação dos dados, pois os intervalos possuem períodos maiores que a taxa de upload do ubidots.

C. *Testes para monitoramento do batimento cardíaco*

Os batimentos cardíacos (taxa cardíaca (TC)) foram determinados conforme a expressão a seguir, onde se calcula o intervalo de período entre os picos de duas ondas R (T_{R-R}). Os picos são determinados por um limiar de cerca de 100 resultante do conversor A/D. E para uma estimativa de batimentos por minuto (bpm), o valor de 60 que representa os segundos é dividido pelo período entre os picos de onda.

$$TC = \frac{60}{T_{R-R}} \tag{2}$$

A expressão foi feita no algoritmo em C, conforme em anexo, para o cálculo dos batimentos cardíacos, tendo em vista que a faixa de batimentos cardíacos pode está entre 40 a 130 bpm, os demais valores fora desta faixa serão considerados espúrios e serão descartados. Nesse sentido, os resultados no Ubidots estão mostrados nas figuras 12 e 13 abaixo.

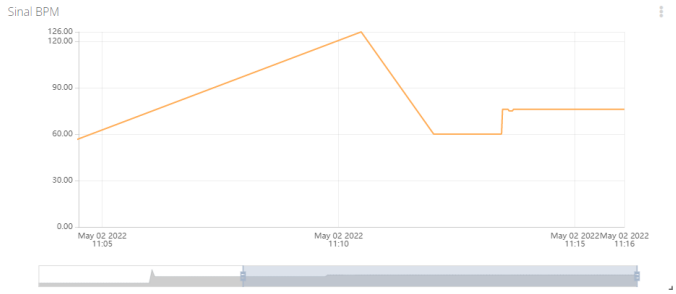

Figura 12: Primeiro teste com os batimentos cardíacos enviado da raspberry PI para o dashboard do Ubidots.

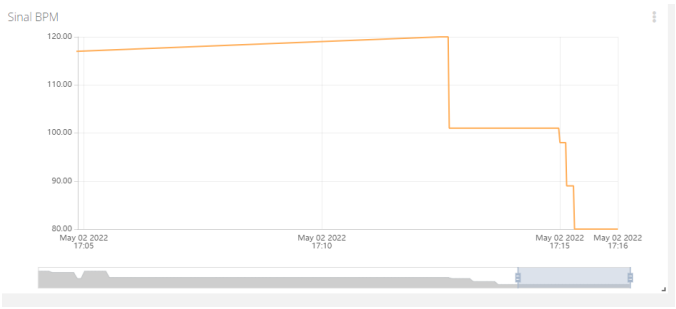

Figura 13: Segundo teste com os batimentos cardíacos enviado da raspberry PI para o dashboard do Ubidots.

Tendo em vista a forte presença de ruidos e interferências, foi utilizado uma média móvel na saída para evitar possiveis variações abruptas dos dados.

VI. Bill of Materials (BOM)

- Módulo AD8232
- 3 eletrodos comum eletrolíticos
- Raspberry Pi 3b
- Arduino Uno
- Laptop

VII. Conclusão

Na execução do projeto, alguns dos objetivos foram alcançados, tendo em vista que o sinal foi enviado para o servidor online, porém não sendo possivel o monitoramento do mesmo remotamente devido taxa de upload do servidor Ubidots. Por outro lado, o monitoramento dos batimentos cardiocos ser tornou viável, devido as taxas de aquisição desses dados. Algumas dificuldades foram encontradas no decorrer do desenvolvimento das partes analógica e digital.

Após os testes de continuidade descobriu-se que o modulo sensor ad8232 estava com a identificação dos respcetivos cabos LA e RL com indentificação invertida O cabo amarelo do eletrodo representado por LA (Left Arm) é na verdade RL (Right Leg) representado pelo cabo verde e virce-versa.

Os cabos do ECG devem estar devidamente desenrolados, dado que ocorre interferência eletromagnética entre eles.

As dificuldades no circuito analógico foram: sensor comercial de baixo custo para prototipagem sem qualidade técnica para diagnóstico clínico preciso; sinal ECG sensivel ao ruído; interferência eletromagnética; interferência do sinal eletromiográfico; condutividade da pele e mal contato nas conexão dos dispositivos eletrônicos. Na parte digital, foram encontradas as seguintes dificuldades: latência no upload dos dados para API Ubidots; limitação da comunicação da rasberry PI com servidor do ubidots.

Este projeto pode ser incrementado com novas funcionalides. A seguir, apresenta-se algumas delas:

- Componentes mais robustos;
- Escolha de um novo servidor, que faça o upload mais rápidamente dos dados;
- Desenvolvimento de uma *case*
- Caso continue usando o Ubidots, fazer a requisição http pela linguagem C, o que poderia acelerar o upload dos dados.
- Desenvolvimento de um algoritmo de *machine learning* para o auxílio no diagnóstico de patologias cardíacas.

Referências

[1] Becchetti, C.; Neri, A.; Medical Instrument Design and Development: From Requirements to Market Placements; Ed 1; Chinchester: John Wiley & Sons, 2013.

[2] Reis, H. J. L. (el al.); ECG: manual prático de eletrocardiograma; São Paulo: Atheneu, 2013. Disponível em: ECG-Manual-Prático-de-Eletrocardiograma-HCor.pdf (uff.br).

[3] Siqueira, A. da S. E. (et al.); Análise do Impacto Econômico das Doenças Cardiovasculares nos Últimos Cinco Anos no Brasil; v.109; n.01; 2017. Disponível em: https://doi.org/10.5935/abc.20170068.

[4] Lim, Y. G. (et al.); ECG Recording on a Bed During Sleep Without Direct Skin-Contact; IEEE Transactions on Biomedical Engineering; v.54; n.04; 2007.

[5] https://www.embarcados.com.br/arduino-taxa-de-amostragem-conversor-ad/#Leitura-padrao-do-AD-no-Arduino.

[6] https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds).

[7] https://www.medicinanet.com.br/cid10/i.htm.

[8] ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. ABNT NBR 60601-1: Equipamento eletromédico Parte 1: Requisitos gerais para segurança básica e desempenho essencial. Rio de Janeiro: ABNT, 2016.

[9] ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. ABNT NBR 60601-1-2: Equipamento eletromédico Parte 1-2: Requisitos gerais para segurança básica e desempenho essencial — Norma Colateral: Perturbações eletromagnéticas — Requisitos e ensaios. Rio de Janeiro: ABNT, 2017.

[10] ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. ABNT NBR 60601-1-6: Equipamento eletromédico Parte 1-6: Requisitos gerais para segurança básica e desempenho essencial — Norma colateral: Usabilidade. Rio de Janeiro: ABNT, 2011.

[11] ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. ABNT NBR 60601-1-11: Equipamento eletromédico Parte 1-11: Requisitos gerais para segurança básica e desempenho essencial - Norma Colateral: Requisitos para equipamentos eletromédicos e sistemas eletromédicos utilizados em ambientes domésticos de cuidado à saúde. Rio de Janeiro: ABNT, 2015.

[12] ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. ABNT NBR ISO 14971: Produtos para a saúde — Aplicação de gerenciamento de risco a produtos para a saúde. Rio de Janeiro: ABNT, 2009.

[13] ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. ABNT IEC/TR 80002-1:2020: Software de produto para saúde Parte 1: Orientação sobre a aplicação da ABNT NBR ISO 14971 a software para produtos para a saúde. Rio de Janeiro: ABNT, 2020.

[14] ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. ABNT IEC/TR 60601-4-2:2016: Equipamento eletromédico Parte 4-2: Orientações e interpretação — Imunidade eletromagnética: desempenho de equipamentos eletromédicos e sistemas eletromédicos. Rio de Janeiro: ABNT, 2016.

VIII. Anexos

Código C

Raspberry recebe os dados de ECG do Arduino

```c
#include <stdio.h>
#include <stdlib.h>
#include <signal.h>
#include <string.h>
#include <unistd.h>
#include <math.h>
#include <fcntl.h>
#include <termios.h>
#define TTY "/dev/ttyS0"

int uart0_fd,num;
char comandoCompleto[100];
void ctrl_c(int sig)
{
        puts(" CLOSING " TTY "...");
        close(uart0_fd);
        exit(-1);
}
void disconnect_uart()
{
        puts("UART DISCONNECTED");
        close(uart0_fd);
        exit(-1);
}

int main(void)
{
        struct termios options;
        signal(SIGINT, ctrl_c);
        uart0_fd = open(TTY, O_RDWR); // | O_NOCTTY)
            ; // | O_NDELAY);
        if(uart0_fd==-1)
        {
                puts("ERROR OPEN UART");
                return -1;
        }
        tcgetattr(uart0_fd, &options);
        options.c_cflag = CS8 | CREAD | CLOCAL;
        options.c_iflag = 0;
        options.c_oflag = 0;
        options.c_lflag = 0;
        options.c_cc[VTIME] = 0;
        options.c_cc[VMIN]  = 1;
        cfsetospeed(&options, B115200);
        cfsetispeed(&options, B115200);
        tcflush(uart0_fd, TCIOFLUSH);
        tcsetattr(uart0_fd, TCSANOW, &options);
        unsigned char rx_buffer[6];
              FILE *arq;
        arq=fopen("dados_ecg.xlsx", "wt");
        if(arq == NULL)
        {
                printf("ERRO NA ABERTURA DE ARQUIVO\
                    n");
                return -1;
        }
        int cont_pico_R=0;
        int tempo = 0;
        double batimentos_card;
        int result;
        int rx_length;
        int vetor_bc,j,i=0;
        double media=0.0, media_movel_bc[100];
        while(1)
        {
            int rx_length = read(uart0_fd, (void*)
                rx_buffer, 5);
            usleep(2000);
            cont_pico_R++;
            tempo++;
            if (rx_length < 0)
            {
                disconnect_uart(); //An error
                    occured (will occur if there are
                     no bytes)
            }
            if (rx_length == 0)
            {
                printf("NO DATA");//No data waiting
            }
          result =fputs(rx_buffer,arq);
            if (result == EOF)
            {
                printf("ERROR RECORDING\n");
            }
          num = strtol(rx_buffer, NULL, 10); //
            Transformando para n mero inteiro de
            base 10
            if (num >= 100)
            {
                //printf("%d\n",cont_pico_R);
                batimentos_card=60/(cont_pico_R
                    *0.002);
                if (batimentos_card>=40 &
                    batimentos_card<=130){
                    i++;
                    if(i==2){
                        vetor_bc=batimentos_card;
                        media =(double)vetor_bc;
                        printf("%.f\n",media);
                        //snprintf(comandoCompleto,
                            100, "python3
                            rasp_to_ubidots.py %.f",
                             batimentos_card); //
                            concatena a frase >
                        //system(comandoCompleto);
                    }
                    if(i>2){
                        vetor_bc+=batimentos_card;
                        media=(media+vetor_bc)/i;
                            printf("%.f\n",media);
                        //snprintf(comandoCompleto,
                            100, "python3
                            rasp_to_ubidots.py %.f",
                             batimentos_card); //
                            concatena a frase >
                        //system(comandoCompleto);
                    }
                cont_pico_R=0;
            }
            else
            {
               if (i>=2){
                    for(j=1;j<=300;j++){
                        if(cont_pico_R==100*j){
                            snprintf(comandoCompleto
                                , 100, "python3
                                rasp_to_ubidots.py
                                %.f",
                                batimentos_card); //
                                 concatena a frase >
                            system(comandoCompleto);
                        }
                    }
              }
            }
            memset(rx_buffer,0,6);
            if (tempo == 30000)
            {
                    fclose(arq);
                    close(uart0_fd);
                    exit(-1);
            }
          }
}
```

Código Python

Envia os dados para a API do Ubidots

```
!/usr/bin/env python3
import string
import time
import requests
import math
import sys
from sys import argv

TOKEN = "BBFF-L48Awxo7p33L5GSmURXCcaqt3Cl5Bl"  # Put your TOKEN here
DEVICE_LABEL = "ECG_signal_monitoring_raspberry"  # Put your device label here
VARIABLE_LABEL_1 = "ecg_sensor_data"  # Put your first variable label here
serialdata=[]
def post_request(payload):
    # Creates the headers for the HTTP requests
    url = "http://industrial.api.ubidots.com"
    url = "{}/api/v1.6/devices/{}".format(url, DEVICE_LABEL)
    headers = {"X-Auth-Token": TOKEN, "Content-Type": "application/json"}

    # Makes the HTTP requests
    status = 400
    attempts = 0
    while status >= 400 and attempts <= 5:
        req = requests.post(url=url, headers=headers, json=payload)
        status = req.status_code
        attempts += 1
        #time.sleep(0.1)

    # Processes results
    print(req.status_code, req.json())
    if status >= 400:
        print("[ERROR] Could not send data after 5 attempts, please check \
            your token credentials and internet connection")
        return False

    print("[INFO] request made properly, your device is updated")
    return True

if __name__ == '__main__':
    serial_data= sys.argv[1:]
    payload = {VARIABLE_LABEL_1:serial_data}
    post_request(payload)
```

Código Arduino

Arduino envia dados pra Raspberry

```
void setup() {
   Serial.begin(115200);
   pinMode(3, INPUT); // Configura  o para detec  o de deriva  es LO +
   pinMode(2, INPUT); // Configura  o para detec  o de leads off LO -
}

void loop() {

  if((digitalRead(10) == 1)||(digitalRead(11) == 1)){
    Serial.println('0');
  } else {
    // envia o valor da entrada anal gica 0:
    Serial.println(analogRead(A0));
  }
  // Espere um pouco para evitar que os dados seriais saturem
  delay(2);
}
```

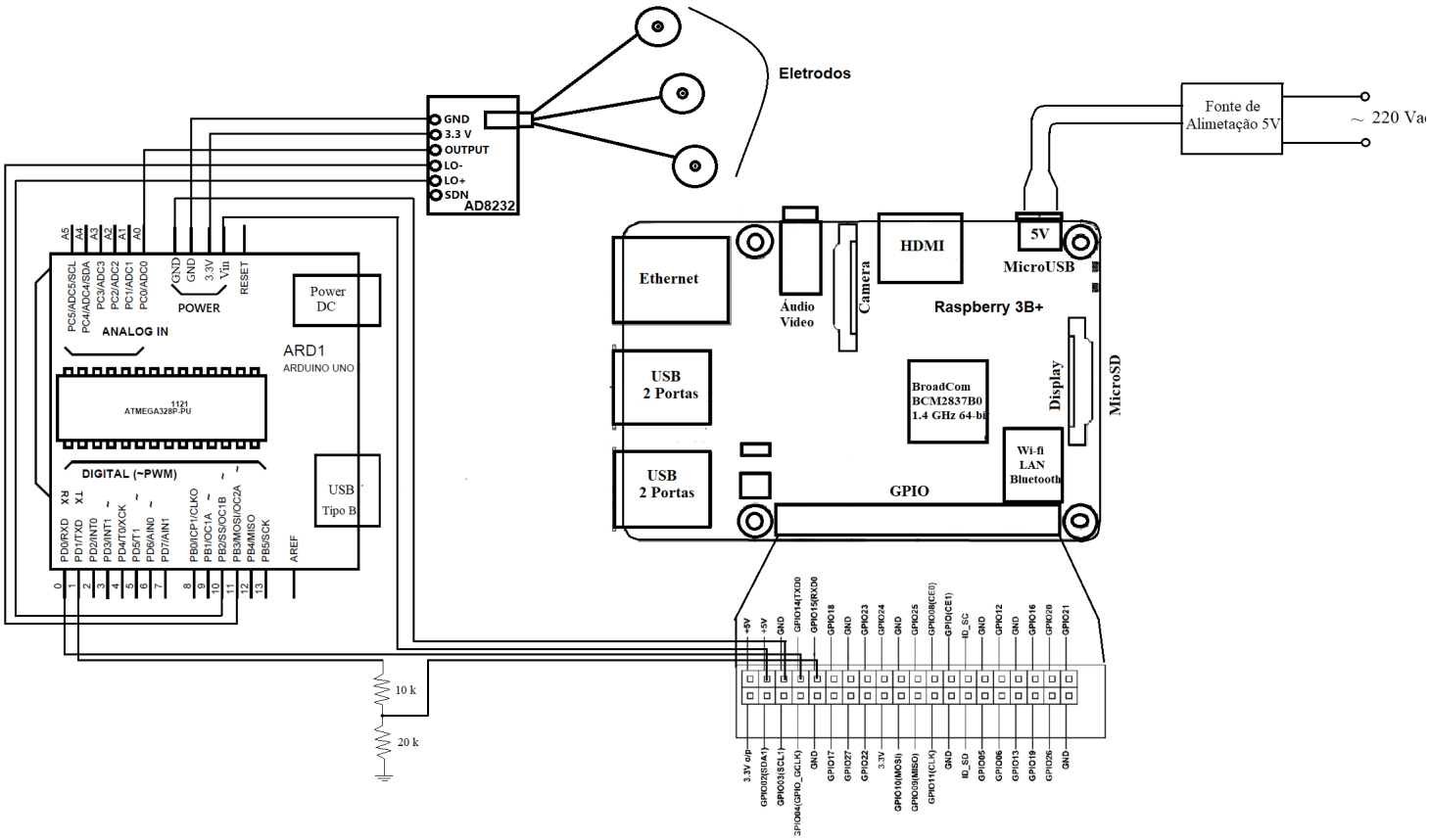

Figura 14: Esquemático do projeto com as conexões da Raspberry PI, arduino UNO e módulo AD8232. Fonte: autoria própria