

UNIVERSIDADE REGIONAL DE BLUMENAU - FURB
CENTRO DE CIÊNCIAS TECNOLÓGICAS DO DEPARTAMENTO DE
ENGENHARIA ELÉTRICA
CURSO DE ENGENHARIA ELÉTRICA

JAIRO EVANDRO LENFERS

ESTAÇÃO METEOROLÓGICA AUTOSSUSTENTÁVEL DE CÓDIGO ABERTO

BLUMENAU

2017

JAIRO EVANDRO LENFERS

ESTAÇÃO METEOROLÓGICA AUTOSSUSTENTÁVEL DE CÓDIGO ABERTO

Trabalho de conclusão de curso apresentado ao Curso de Graduação em Engenharia Elétrica do Centro de Ciências Tecnológicas da Fundação Regional de Blumenau, como requisito parcial para a obtenção do grau de Engenheiro Eletricista.

Prof. Sérgio H. Lopes Cabral – Orientador

BLUMENAU

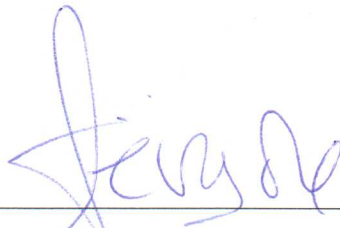
2017

JAIRO EVANDRO LENFERS

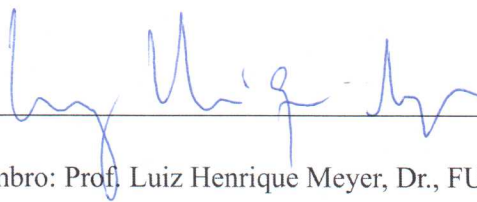
ESTAÇÃO METEOROLÓGICA AUTOSSUSTENTÁVEL DE CÓDIGO ABERTO

Trabalho de conclusão de curso aprovado para
obtenção do grau de Engenheiro Eletricista,
pela Banca examinadora formada por:

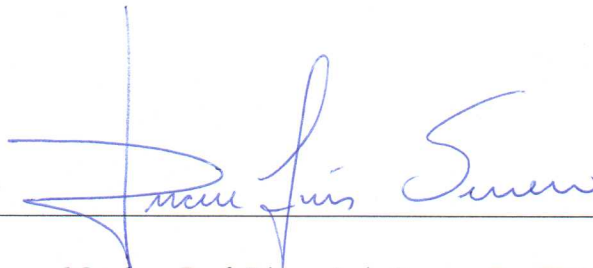
Aprovado em: 26/07/2017.



Presidente: Prof. Sérgio Henrique Lopes Cabral, Dr. - Orientador, FURB



Membro: Prof. Luiz Henrique Meyer, Dr., FURB



Membro: Prof. Dirceu Luis Severo, Dr., FURB

AGRADECIMENTOS

Primeiramente, agradeço a Deus, meu Pai Celestial, por ter me dado saúde e força para superar as dificuldades ao longo da minha jornada, e que em todos os momentos é o maior mestre que alguém pode ter.

A minha mãe, Iracema Isensee, a minha esposa, Jessica e ao meu filho, Arthur, pelo extenso carinho e amor incondicional cedidos a mim ao longo de minha vida, guiando-me sempre pelos caminhos do bem e despertando em mim a sede pelo conhecimento.

À Blunix Tecnologia, meu sócio Maicon e meus colaboradores, que por muitas vezes trabalharam exaustivamente em minha ausência para que eu tivesse tempo para me dedicar aos estudos.

Ao meu amigo Eduardo Maes, que por diversas vezes me mostrou as vantagens do *Software* Livre e sempre me ajudou com suas ideias para melhorar meus projetos.

De forma especial agradeço ao meio orientador, professor Sérgio Henrique Lopes Cabral, que me incentivou a continuar firme nos estudos e sempre acreditou no meu potencial, pela atenção despendida e o suporte nas orientações e correções deste trabalho.

Aos meus amigos, Frederico Marks e Rafael Bruns, que no final da graduação resgataram a minha vontade de estudar e me ajudaram nos estudos finais de semana para que eu pudesse concluir esta graduação.

Aos meus colegas, amigos e professores do curso de Engenharia Elétrica, no qual estivemos juntos nesta caminhada, pela companhia e amizade em todos os momentos.

A todas as pessoas que de alguma forma contribuíram para a realização deste trabalho.

Não cruze os braços diante de uma dificuldade,
pois o maior homem do mundo morreu de braços
abertos! (Autor desconhecido)

RESUMO

O uso de estações meteorológicas está cada vez mais comum, seja para uso agrícola, para utilizar as informações coletadas para tomadas de decisões ou até mesmo para *hobby*. O principal problema das estações meteorológicas comerciais é sua comunicação, que muitas vezes é *offline*, onde somente é possível visualizar as informações no local, e quando estas informações são *online*, utilizam protocolos fechados desenvolvidos pela própria empresa, sem compatibilidade com outros sistemas. A utilização desta estação inicialmente será focada com destino do uso em *hobby* em um clube de aeromodelismo, para ter informações das condições meteorológicas em tempo real antes de se deslocar para o local, pois dependendo do tempo, fica inviável a prática. Este trabalho trata desde a aquisição dos sinais recebidos pelos sensores até a transmissão pela *internet* possibilitando acessar de um aplicativo móvel. Serão utilizados sensores prontos, adquiridos na internet e apresentados detalhes do funcionamento destes individualmente. A pesquisa encontra-se em código aberto, possibilitando que outras pessoas reproduzam os métodos aqui descritos em sua totalidade e consigam contribuir com a melhoria deste projeto a fim de torná-lo cada vez melhor.

Palavras-chave: Estação meteorológica. Tecnologia. Internet das coisas. Arduino. ESP8266. Eletrônica.

ABSTRACT

The use of weather stations is frequently increasing, whether for the use in agriculture, for the use of information collected for decision making or even as a *hobby*. The main problem of commercial weather stations is its communication, which is often offline and only accessible in loco and, when this information is online, closed protocols developed by the company itself are used, not compatible with other systems. The use of this station will initially be focused for the use of a *hobby* at an aeromodelling club, in order to obtain weather information in real time before heading to the location since, depending on the weather it is unfeasible to practice the hobby under certain circumstances. This study begins with the acquisition of the signals received by the sensors to the transmission through the internet, allowing access from a mobile application. It will utilize ready for use sensors acquired from the internet and the operational details of these sensors will be presented individually. The research is an open source, enabling others to reproduce the methods described herein as a whole and to contribute to the improvement of this project in order to make it even better.

Key words: Weather station. Technology. Internet of things. Arduino. ESP8266. Electronics.

LISTA DE ILUSTRAÇÕES

Figura 1 – Climas no Brasil.....	12
Figura 2 – Estações meteorológicas do INMET em Santa Catarina	13
Figura 3 – Anemômetro	16
Figura 4 – Pluviômetro manual	17
Figura 5 – Tanque demonstrando 1mm de água.....	17
Figura 6 – Pluviômetro eletrônico	18
Figura 7 – Sensor de temperatura e umidade	19
Figura 8 – Data logger	20
Figura 9 – Clube de aeromodelismo Asas do Vale em Gaspar	23
Figura 10 – Exemplo troca de informações sobre o tempo local	24
Figura 11 – Diagrama de blocos	25
Figura 12 – Placa solar	26
Figura 13 – Carregador solar	27
Figura 14 – Bateria VRLA 6v 12a.....	28
Figura 15 – Anemômetro velocidade do vento – parte interna.....	29
Figura 16 – Anemômetro velocidade do vento – parte superior.....	29
Figura 17 – Esquema elétrico anemômetro	30
Figura 18 – Biruta direção do vento	31
Figura 19 - Orientação bússola.....	32
Figura 20 – Esquema elétrico – Biruta eletrônica	32
Quadro 1 – Relação de resistência em relação à posição do anemômetro	33
Figura 21 – Sensor temperatura e umidade DHT22	34
Figura 22 – Esquema elétrico DHT22	34
Figura 23 – Abrigo sensor temperatura	35
Figura 24 – Parte superior do pluviômetro	36
Figura 25 – Interior do pluviômetro	36
Figura 26 – Esquema elétrico pluviômetro.....	37
Figura 27 – NODEMCU ESP12-E.....	39
Figura 28 – Esquema de ligação do microcontrolador com sensores.....	40
Figura 29 – Percurso do sol durante o dia	42
Figura 30 – Clube Aeromodelismo Aeroblu – Blumenau	43
Figura 31 – Local exato da estação meteorológica.....	43
Figura 32 – Estação meteorológica montada no suporte	44
Figura 33 – Exemplo de utilização do sistema telegram	45
Quadro 2 – custos aproximados de construção da estação meteorológica	46

LISTA DE ABREVIATURAS E SIGLAS

INMET – Instituto Nacional de Meteorologia

mA – mili ampéres

A - ampéres

V - volts

VRLA (*Valve-regulated lead-acid*)

PWM - Pulse-Width Modulation

mm – milímetros

ADC – Conversor analógico para Digital

WiFi – Rede Wireless

SUMÁRIO

1	INTRODUÇÃO	11
1.1	HISTÓRICO MUNDIAL	11
1.2	HISTÓRICO NACIONAL	12
1.3	OBJETIVOS	14
1.3.1	Objetivo Geral	14
1.3.2	Objetivos Específicos	14
2	REVISÃO DE LITERATURA	15
2.1	ANEMÔMETRO	15
2.2	PLUVIOMETRO	16
2.3	TEMPERATURA E UMIDADE DO AR	18
2.4	DATA LOGGER	19
2.5	SOFTWARE LIVRE	20
3	PROPOSTA.....	22
4	DESENVOLVIMENTO	25
4.1	MODULO SOLAR	26
4.2	CARREGADOR SOLAR	27
4.3	ARMAZENAMENTO DA ENERGIA – BATERIA.....	28
4.4	ANEMÔMETRO.....	28
4.4.1	Esquema elétrico.....	30
4.4.2	Calibração.....	30
4.4.3	Código	31
4.5	BIRUTA ELETRÔNICA.....	31
4.5.1	Esquema elétrico.....	32
4.5.2	Calibração.....	32
4.5.3	Código	33
4.6	SENSOR TEMPERATURA E UMIDADE DO AR.....	33
4.6.1	Esquema elétrico.....	34
4.6.2	Calibração.....	34
4.6.3	Código	35
4.7	PLUVIÔMETRO	35
4.7.1	Esquema elétrico.....	37
4.7.2	Calibração.....	37
4.7.3	Código	38

4.8	MICRO CONTROLADOR.....	38
4.9	TELEGRAM	40
5	ESTUDO DE CASO.....	42
6	ANÁLISE DO RESULTADO, CONCLUSÕES E PROPOSTAS FUTURAS.....	47
	REFERÊNCIAS	48
	ANEXO 1 – CÓDIGO MICROCONTROLADOR ESP8266	51

1 INTRODUÇÃO

As alterações climáticas estão recebendo mais atenção da mídia nesses últimos tempos e se tornou um tema habitual em jornais, sites e revistas. As informações são repassadas, algumas de maneira equivocada devido ao sensacionalismo das emissoras de televisão e muitas vezes sem fundamento teórico.

A necessidade de obter informações das condições climáticas remotamente em pontos específicos e em tempo real está se tornando cada vez mais comum. Com a imensa extensão do nosso país e sua diversidade, uma informação sobre o tempo pode mudar completamente em distâncias curtas. Por este motivo, dificilmente é possível utilizar uma informação coletada em uma estação meteorológica instalada em um local distante do ponto em que é preciso medir, pois podem sofrer alterações.

Com o uso de sensores adquiridos da internet, é possível desenvolver uma estação meteorológica com comunicação pela internet. Os sensores vão captar os dados de temperatura, umidade relativa do ar, direção do vento, velocidade do vento e intensidade da chuva. Os dados dos sensores são processados por um microcontrolador e estão disponíveis para serem acessados via internet de qualquer lugar do mundo, através de um aplicativo de mensagem instantânea em aparelhos celulares.

1.1 HISTÓRICO MUNDIAL

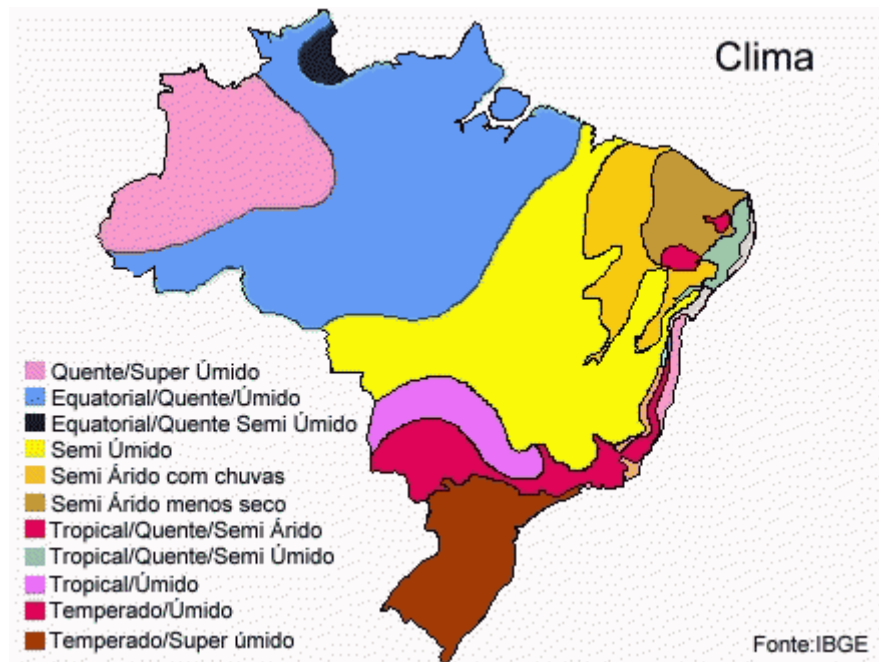
Diante das mudanças climáticas do planeta, tais como o aquecimento global e o derretimento das geleiras, acelerados pela ação humana, torna-se cada vez mais difícil prever as mudanças climáticas. Em consequência disso, tem-se o aumento dos níveis dos oceanos, desertificação, alteração do regime das chuvas, inundações e a redução da biodiversidade [1].

Estas alterações climáticas derivadas do aquecimento global são muito graves e podem impactar profundamente o planeta. Estudos indicam que a temperatura global subiu 5°C nos últimos 10 mil anos - contados desde a última geração até 10 mil anos atrás – e pode aumentar os mesmos 5°C nos próximos 200 anos [2], mostrando que o aquecimento global é grande o suficiente para que uma “pessoa na rua” perceba que o tempo está realmente mais quente [3].

1.2 HISTÓRICO NACIONAL

Não muito diferente do histórico mundial com o aquecimento global, o histórico nacional tem mostrado o mesmo problema. Com uma diversidade climática imensa, extensão territorial de 8,5 milhões de km, motivo ao qual é conhecido por “país tropical” [4]. A diversificação do clima depende de muitos fatores tais como umidade, correntes marítimas, ventos, pressão atmosférica, dentre outros. A Figura 1 representa o mapa do Brasil que mostra os climas e suas diversidades.

Figura 1 – Climas no Brasil



Fonte: IBGE [4]

Com estas diferenças climáticas entre as regiões, a informação do tempo de cada local torna-se imprescindível para planejar a melhor colheita de determinado alimento utilizando dados estatísticos, coletados nos anos anteriores.

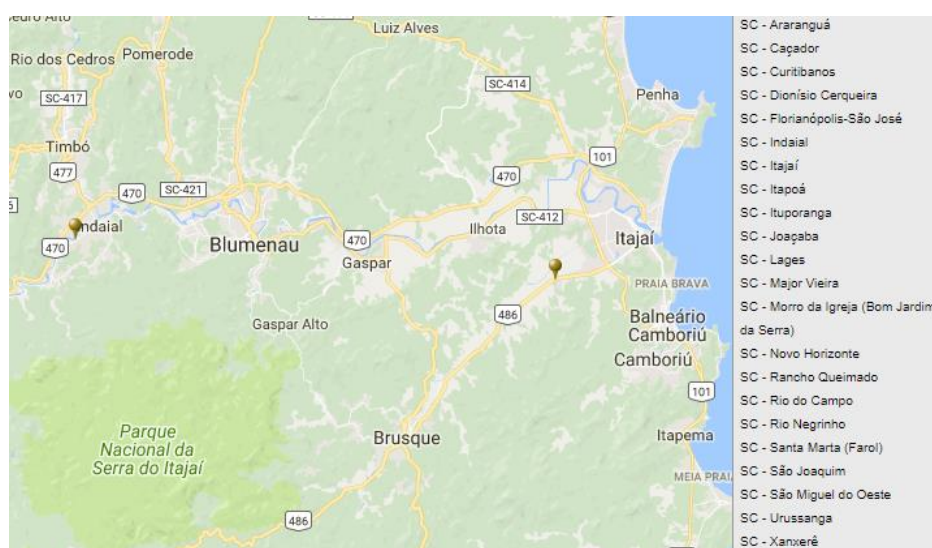
Com a vinda dessas alterações pode-se notar diversos prejuízos econômicos, como a crise que afeta diversos setores, físicos na estrutura das cidades e sociais na população. Tais prejuízos são impostos em questão de minutos, horas, semanas mas levam anos para serem restabelecidos.

Na região de Blumenau, Santa Catarina, tem-se históricos de chuvas abundantes em determinados momentos, ocasionando enxurradas e enchentes. Estas chuvas ocorreram em curto espaço de tempo e causaram bastante estragos [5]. Tais estragos poderiam ser menores caso a população tivesse uma rede de estações meteorológicas interligadas, informando o local exato e quantidade das chuvas. Desta forma, seria possível identificar onde a chuva está ocorrendo exatamente e com qual intensidade. Com uma análise de dados, pode-se estipular de onde e para onde estas chuvas estão indo e tomar certas decisões.

Muito embora haja estações meteorológicas instaladas, de modelos e diversos fabricantes, cada uma destas estações trabalha com um protocolo diferente e não se comunicam entre si. Além do fato de terem protocolos diferentes, muitas destas estações passam por um tratamento de dados antes de ser apresentado ao público e algumas vezes com certo tempo de atraso. Em alguns casos, as medições são feitas manualmente, onde um profissional se encarrega de coletar os dados dos sensores e caso o acesso a este local esteja comprometido, a medição também estará comprometida devido à dificuldade de chegar ao local.

A medição exata do local em que se precisa da informação meteorológica também é um fator decisivo. Atualmente existem no Brasil diversas estações meteorológicas instaladas, porém, estão em locais específicos, determinados por critérios específicos do órgão que a instalou. O Instituto Nacional de Meteorologia (INMET) possui atualmente 22 estações meteorológicas instaladas no estado de Santa Catarina e somente duas estações estão próximas à Blumenau conforme mostra a Figura 2.

Figura 2 – Estações meteorológicas do INMET em Santa Catarina



Fonte: INMET (2017) [6].

1.3. OBJETIVOS

1.3.1 Objetivo Geral

O objetivo geral visa pesquisar e desenvolver um projeto que atenda as funcionalidades de uma estação meteorológica e que seja autossustentável de código aberto, englobando o uso de sensores de alta precisão e baixo custo, aliados ao uso de tecnologias e rede de transmissão dos dados, utilizando e mantendo todo o projeto com *hardware* e *software* livre, para que outras pessoas possam desenvolver mais funcionalidades e agregar a este projeto novas funções e possíveis correções.

O uso desta estação meteorológica tem a finalidade de atender a necessidade de um clube de aeromodelismo em que faz necessário saber as informações meteorológicas em ponto específico.

1.3.2 Objetivos Específicos

Para atingir o objetivo geral deste projeto, é preciso que alguns objetivos específicos sejam estudados, tais como:

- O funcionamento dos sensores independentes (anemômetro, biruta, temperatura e umidade do ar).
- Desenvolver o software do microcontrolador para receber as informações dos sensores, processar estas informações e transmitir pela internet.
- Pesquisar a utilização de aplicativo específico para receber as mensagens com informações coletadas pela estação meteorológica.
- Avaliar qual licenciamento utilizar para que este projeto fique disponível para melhorias futuras e possível comercialização.

2 REVISÃO DE LITERATURA

Nesta revisão de literatura, serão abordados todos os itens que compõem uma estação meteorológica tais como: o anemômetro que registra a velocidade e direção do vento, o pluviômetro, que mede a quantidade de chuva em determinado período, sensores de temperatura e umidade, abrigos seguros para estes sensores, *dataloger* que registra os dados de vários sensores.

Como este trabalho tem o propósito de ser em código aberto, será abordado também fundamentos do *software* livre e licenciamento aberto, que permite que outras pessoas possam contribuir com este trabalho.

2.1 ANEMÔMETRO

Anemômetros são instrumentos utilizados para indicar a direção e medir a velocidade do vento. Inspirado nos cata-ventos, eles são calibrados de forma que o total de voltas dadas por suas pás correspondam a uma velocidade específica, ou seja, se no túnel de vento em que foi calibrado a corrente do ar sopra a dez quilômetros por hora, e as pás giram cem vezes por minuto, ele é programado para indicar 10 km/h sempre que o anemômetro atingir 100 rotações por minuto, e assim por diante [7].

O modelo de anemômetro mais utilizado é o modelo de copos (Anemômetro de Robinson), mostrado na Figura 3, que utiliza três ou mais conchas montadas simetricamente e é do tipo rotativo. A velocidade de rotação depende da velocidade do vento, não interferindo em que direção ele chega no sensor. Geralmente utilizam um sensor magnético que conta o número de voltas e de acordo com este número, é calculado a velocidade instantânea.

Figura 3 - Anemômetro



Fonte: CECHIN (2011). [7]

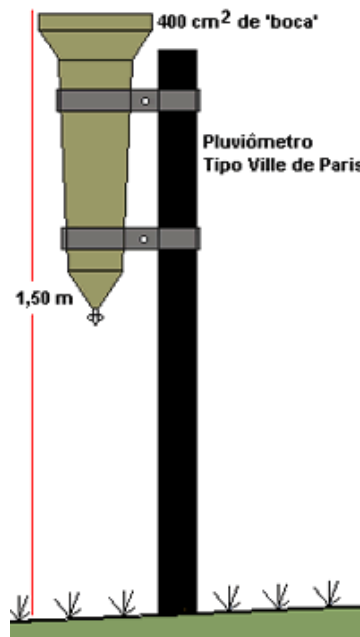
Acoplado a este anemômetro está o sensor de direção do vento. Este sensor funciona com uma pá que mantém o “nariz” do sensor sempre apontado para frente, indicando a direção que o vento chega no sensor. É possível medir a velocidade do vento de duas formas eletronicamente, com um potenciômetro ou com sensores magnéticos. A desvantagem de se utilizar potenciômetro é sua vida útil e o possível arrasto que ele pode ter ao se movimentar.

2.2 PLUVIOMETRO

O pluviômetro é utilizado para medir em milímetros a quantidade de líquidos ou sólidos, precipitados em determinado tempo e local [8].

Os modelos mais conhecidos, funcionam de modo que captam em um intervalo, a quantidade de chuva precipitada, armazenando esta quantidade de chuva capturada em um reservatório, conforme ilustrado na Figura 4. No final do dia ou no final da chuva, um técnico especializado vai até o local para medir a quantidade de chuva no reservatório, funcionando de forma totalmente manual.

Figura 4 – Pluviômetro manual

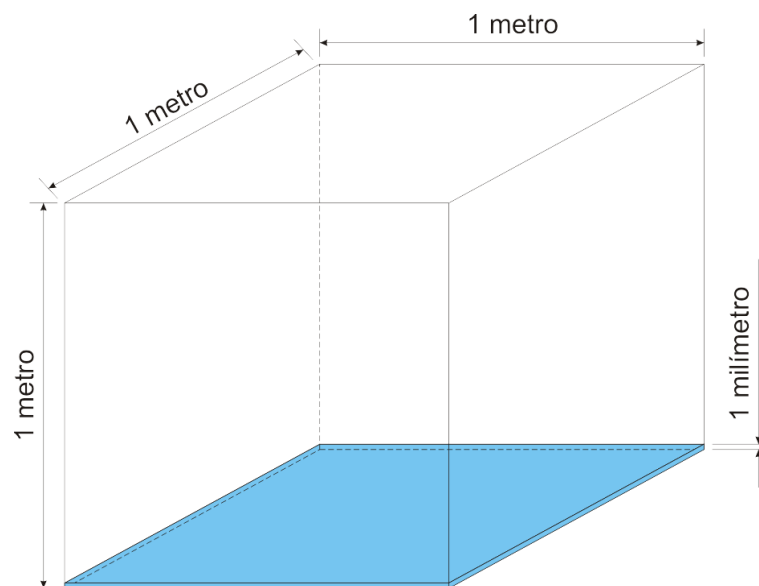


Fonte: HIDRAULIS [8].

Para entender como funciona a medição de um pluviômetro, é preciso saber como funciona a unidade de medida de um pluviômetro.

Os pluviômetros registram as precipitações em milímetros (mm). O tanque representado na Figura 5 possui uma área de base de 1m^2 , se colocar 1 litro de água neste tanque, se tem 1mm de água.

Figura 5 – Tanque demonstrando 1mm de água



Fonte: MARTINS (2016). [9]

Da mesma forma é utilizado o pluviômetro manual, podemos descrever o pluviômetro eletrônico, porém, ao invés de ter um reservatório para armazenar a quantidade de água captada em um determinado período, ele possui um dispositivo chamado “gangorra” com dois lados intitulado báscula, no qual, enche um pequeno reservatório e cada vez que este pequeno reservatório enche, é enviado um pulso elétrico para o controlador efetuar a contagem [10]. De acordo com a quantidade de pulsos, é possível mensurar a quantidade de chuva precipitada conforme mostrado na Figura 6.

Figura 6 – Pluviômetro eletrônico



Fonte: BROOKE (2012). [11]

2.3 TEMPERATURA E UMIDADE DO AR

A temperatura pode ser medida com o uso de termômetros convencionais, que fornecem um valor instantâneo desta variável. Em Meteorologia, termômetros convencionais são do tipo líquido-em-vidro, cujo princípio de funcionamento se baseia na variação do volume de um líquido apropriado, em resposta a uma mudança de temperatura do meio em que está situado o instrumento. [12]

Termômetros não convencionais são os modelos que possuem sensores elétricos, que são resistores com fio muito fino (cerca de 0,01 mm de diâmetro, cuja variação da resistência elétrica causada pelo sensor é analisada por circuito eletrônico. A umidade relativa do ar úmido, submetido a uma determinada temperatura, é o quociente entre a pressão parcial do vapor e a pressão de saturação àquela temperatura. [12].

O sensor de temperatura e umidade geralmente vem em um único encapsulamento, sendo que o mesmo sensor pode ser utilizado para efetuar as duas medições. A temperatura e umidade do ar necessitam ser captadas por um sensor isolado e protegido, pois se o sol chegar diretamente no sensor, pode interferir nos resultados, da mesma forma o sensor de umidade que pode sofrer interferências com o orvalho que cai diretamente sobre ele. A umidade do ar deve ser medida à sombra, em local ventilado e protegido da precipitação [13], motivo ao qual é utilizado uma proteção para este sensor, mostrado na Figura 7.

Figura 7 – Sensor de temperatura e umidade



Fonte: CLEAN (2012). [14]

De acordo com a Organização Meteorológica Mundial (OMM) – entidade internacional ligada à ONU que coordena as atividades operacionais na área das Ciências Atmosféricas e estabelece normas e alturas padrões para instalação de equipamentos meteorológicos – estes sensores de temperatura e umidade devem estar em uma altura entre 1,25 a 2,00 m acima do terreno e precisa ficar protegido [15].

2.4 DATA LOGGER

Data logger são aparelhos construídos de um micro controlador e memórias específicas para armazenar determinados valores (Figura 8). Conseguem gravar uma série de dados, dos diversos sensores das estações meteorológicas.

Muitas vezes, os gravadores de registros funcionam de forma *offline*, sem comunicação com a internet, sendo possível visualizar as informações apenas estando no local. Em outros casos, estas estações possuem um protocolo de comunicação próprio, onde comunicam-se com um servidor na internet e enviam os dados de tempos em tempos para este servidor próprio. Neste caso, as informações ficam em posse da empresa que desenvolveu esta estação meteorológica, além de não saber se estas informações são utilizadas para outros fins, não se tem garantia pois a empresa pode descontinuar o produto ou cobrar pelo serviço futuramente.

Figura 8 – Data logger



Fonte: ACCEL. [16]

2.5 SOFTWARE LIVRE

Um projeto, programa ou criação em geral pode ser licenciado de várias formas, dentre elas licenciamento fechado ou aberto. Neste trabalho, é utilizado o licenciamento baseado em código livre ou código aberto, que significa que todo seu conteúdo, seja os códigos, desenhos, projetos, podem ser reproduzidos, copiados, alterados e comercializados. Esse tipo de licença foi idealizado por Richard Stallman, que criou a *Free Software Foundation* [17].

O *software* livre deve possuir quatro liberdades fundamentais e essenciais:

- 0 – executar o programa;
- 1 – acesso ao código fonte (*open source* = código aberto): estudar como ele funciona e adapta-lo às suas necessidades;

- 2 – distribuir cópias livremente;
- 3 – melhorar o programa e liberar seus aperfeiçoamentos, para que toda comunidade se beneficie.

A licença utilizada neste trabalho é a Massachusetts Institute of Technology (MIT) que tem a seguinte característica:

- É possível usar, copiar e modificar o *software* pois não tem impedimento de uso o que possibilita utilizá-lo em qualquer projeto (inclusive proprietários);
- Permite a distribuição gratuita e venda sem restrições;
- A única exigência é que o produto deve ser acompanhado com a licença.

Dentre as inúmeras vantagens do *software* livre, é possível ressaltar que o usuário pode ter acesso ao código fonte do programa, contribuir com a melhoria do código ou até mesmo entrar em contato diretamente com o desenvolvedor do *software* para esclarecer dúvidas, avisar sobre falhas no sistema ou sugerir melhorias, resultando em uma agilidade no processo de correção e melhorias. Vantagens estas que não se encontra em *software* proprietários e dificilmente consegue-se entrar em contato diretamente com o desenvolvedor, direcionando a solicitação para a equipe de suporte e que podem demorar até que cheguem ao desenvolvedor.

Um exemplo clássico de uso em *software* livre é o sistema operacional *Linux*, gratuito, desenvolvido por Linux Torvalds com a ajuda de milhares de programadores em todo mundo, por se tratar de um sistema operacional de código aberto. Em contrapartida, o sistema operacional *Windows* de propriedade da Microsoft ® é um sistema de código fechado, onde somente os funcionários da empresa mantêm as atualizações das funcionalidades do seu sistema operacional.

Desta forma, utilizando o licenciamento de código aberto, este trabalho poderá sofrer alterações e melhorias, pois, o movimento do *software* livre tem como filosofia ser colaborativo, que permite as pessoas contribuírem em áreas específicas, aprimorando a ideia central do projeto.

3 PROPOSTA

Quando se fala em estações meteorológicas tem-se a ideia apenas das condições meteorológicas de forma geral. Todavia, é possível utilizá-las para outros fins como meio de informação do ambiente, principalmente em locais específicos. Tais necessidades são utilizadas para trabalho ou para *hobby*.

A ideia de uma estação meteorológica surge a partir de um *hobby* praticado aos finais de semana, o aeromodelismo, que depende da velocidade e força do vento, dependendo das condições meteorológicas o voo torna-se impossível. Outro exemplo são as atividades de asa delta ou parapente, a rampa de salto é direcionada para um local e caso o vento não esteja em direção contrária, impossibilita o voo. Os praticantes destes esportes têm que se deslocar até o local para a prática da atividade e muitas vezes acabam demorando para chegar, para constatar que o vento não está de acordo. Utilizando os dados de outras estações meteorológicas, não se consegue as informações precisas do local do voo e mudam muito de local para local.

Na imagem seguinte, é apresentado um exemplo de local para ser instalado a estação meteorológica. O Clube de Aeromodelismo Asas do Vale (Figura 9), permite a decolagem e pouso dos aeromodelos que são feitos com o vento preferencialmente paralelo à pista. Caso o vento esteja perpendicular à pista, é conhecido como vento de través e necessitam de certa habilidade para pousar.

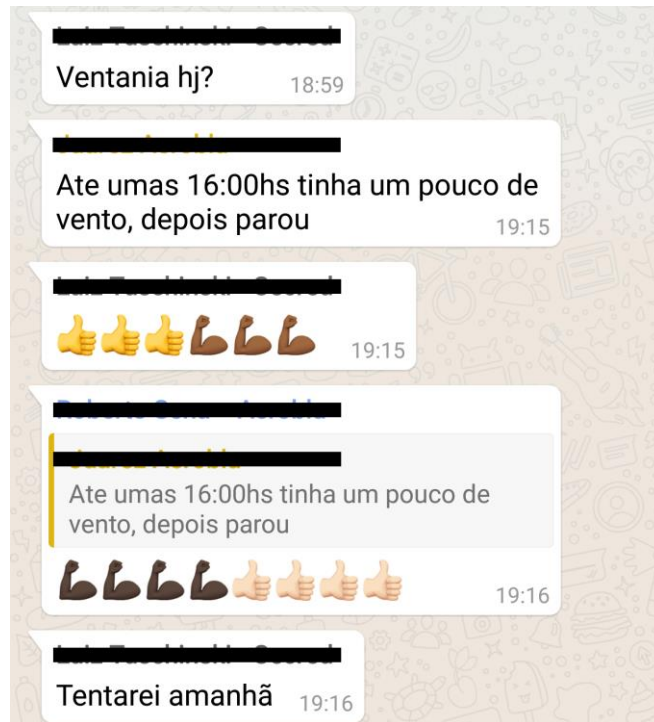
Figura 9 – Clube de aeromodelismo Asas do Vale em Gaspar



Fonte: GOOGLE (2017). [18]

Com base em conversas dos praticantes de aeromodelismo no clube citado, historicamente tem-se os registros que aos sábados e domingos, o vento está paralelo à pista somente até o horário 12h00. Geralmente a tarde o vento está de través e acontecem alguns incidentes por conta disto. A comunicação dos sócios que estão no clube com os sócios que estão para ir ao clube geralmente são de como está o vento e se está chovendo, conforme a Figura 10.

Figura 10 – Exemplo troca de informações sobre o tempo local



Fonte: O autor (2017).

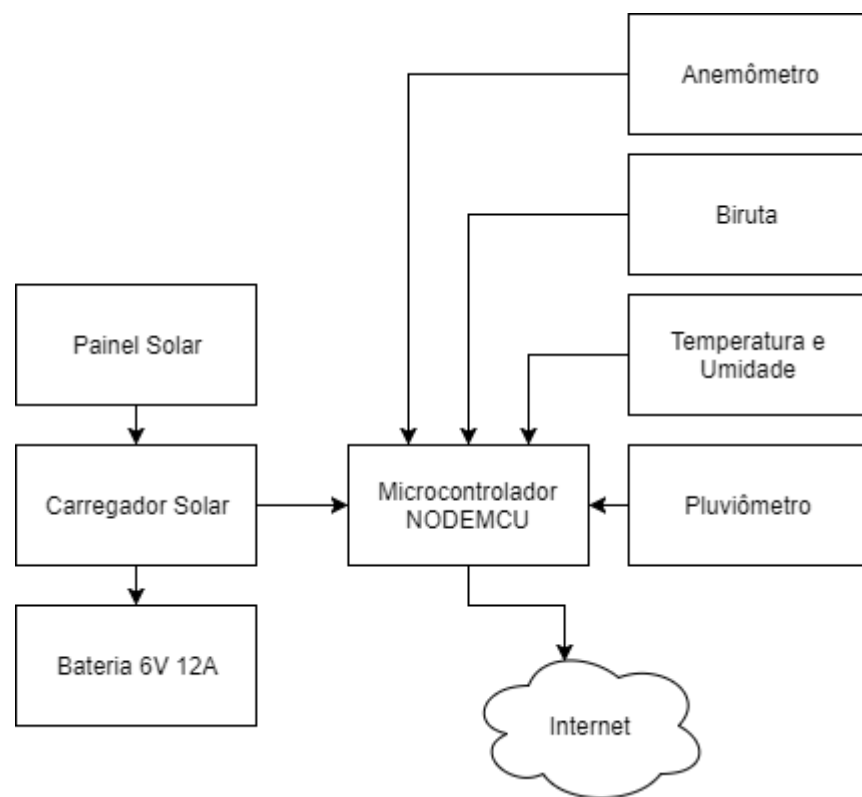
Outro problema é mensurar a velocidade do vento, o que é forte para alguns pode ser fraco para outros, dependendo da habilidade de cada piloto. Por isso é importante uma estação meteorológica no clube para que cada piloto acompanhe em tempo real a velocidade e direção do vento para que se sinta mais seguro em voar.

Além das informações de velocidade e direção do vento, é importante incluir um pluviômetro para medir a quantidade de chuva que caiu no local, um sensor de temperatura e umidade para registrar as variações de temperatura, um sistema de transmissão dos dados e um sistema de carregamento solar, para que funcione de forma autônoma. Desta forma, a estação meteorológica poderá ser utilizada não somente nos clubes de aerodelismo e voo livre, mas também em outras aplicações em que requerem uma exatidão na informação de meteorologia, tais como fazendas para ter um controle mais específico das safras.

4 DESENVOLVIMENTO

O desenvolvimento da estação meteorológica autossustentável se divide em vários módulos conforme é mostrado no diagrama de blocos na Figura 11. Irá contar com vários assuntos relacionados ao curso de Engenharia Elétrica como: geração de energia, armazenamento de energia, análise e processamento dos sinais dos sensores, conversão de dados, processamento em tempo real e transmissão das informações via internet para um dispositivo móvel (celular).

Figura 11 – Diagrama de blocos



Fonte: O autor (2017).

Geração de energia: Este projeto conta com geração de energia solar com um módulo que vai regular a tensão recebida pela placa solar e gerenciar o carregamento da bateria.

Armazenamento de energia: A estação utiliza uma bateria para que o sistema funcione de modo *offline* quando a geração solar não for suficiente ou em períodos noturnos.

Análise e processamento dos sinais dos sensores: Esta estação conta com diversos sensores de diversos tipos. Nem todos contam com a mesma interface e o mesmo protocolo de comunicação. O projeto em questão vai abordar a forma de comunicação e protocolo destes sensores.

Processamento em tempo real: Muitas vezes um determinado microcontrolador não consegue atender todas as necessidades de certo projeto. Entradas e saídas são fundamentais. A escolha entre custo e benefício é essencial. Este projeto conta com um micro controlador que atenda estas necessidades.

Transmissão de dados via internet: Praticamente uma das partes mais importantes deste trabalho é a comunicação online da estação meteorológica com dispositivos móveis.

4.1 MODULO SOLAR

A captação de energia é exclusivamente de forma solar, portanto a escolha de um painel que atenda às necessidades deste projeto de forma adequada é essencial. Mediante a medição do consumo elétrico médio do circuito, chegou-se no valor de 100mA x hora, logo, precisa-se de um painel que forneça energia suficiente para alimentar o projeto e carregar as baterias. O módulo solar escolhido foi o silício policristalino [19] com dimensões de 147x147mm, tensão de 12 volts e corrente nominal de 250 mA x hora por placa.

Para garantir o correto funcionamento, são utilizados quatro painéis em paralelo, para os dias em que a geração solar não estiver no seu valor necessário. As quatro placas solares montadas em seu suporte são mostradas na Figura 12.

Figura 12 – Placa solar



Fonte: O autor (2017).

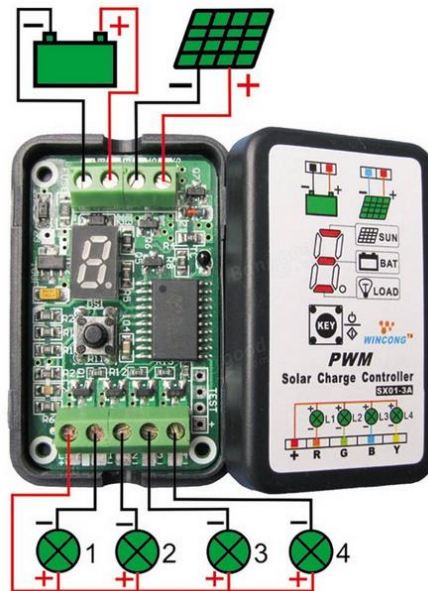
4.2 CARREGADOR SOLAR

A energia captada pelas placas solares não pode ser injetadas diretamente na bateria, para que não haja sobrecorrente e não danifique as baterias, assim, optou-se por utilizar um gerenciador de energia solar [20], com função para carregar as baterias conforme ilustrado na Figura 13. O modelo escolhido foi o SX01-3A com saída de 3 amperes.

Dentre suas especificações, as características abaixo são importantes para este projeto:

- Compatibilidade com bateria VRLA;
- Proteção contra sobrecarga da bateria;
- Chaveamento automático entre a célula solar e a bateria, de acordo com a intensidade do sol;
- Controle de carregamento PWM, que garante eficiência de 3% a 16% maior que os carregadores não-PWM;
- Proteção do painel solar contra curto circuito.

Figura 13 – Carregador solar



Fonte: ALIEXPRESS. [20]

4.3 ARMAZENAMENTO DA ENERGIA – BATERIA

A bateria escolhida para alimentar este projeto quando a geração de energia solar não for suficiente é a bateria VRLA (*Valve-regulated lead-acid*), também conhecida como bateria selada de chumbo ácido, que é livre de manutenção.

Por ser selada, pode ser utilizada em ambientes fechados pois não libera produtos nocivos. Como a saída da célula solar é de 12v, optou-se por utilizar uma bateria de 6v. Outra vantagem de se utilizar bateria de 6v é não precisar de um regulador externo para adequar a tensão para o microcontrolador, uma vez que o regulador interno funciona muito bem com esta tensão.

A capacidade escolhida foi de 12 amperes / hora, que devido ao baixo consumo do projeto permite o uso do sistema por aproximadamente 120 horas, levando em conta um consumo de 100mA / hora. A imagem da bateria é mostrada na Figura 14.

Figura 14 – Bateria VRLA 6v 12a



Fonte: O autor (2017).

4.4 ANEMÔMETRO

Este sensor possui a finalidade de medir a velocidade do vento, utilizando um sensor magnético do tipo *reed switch*, que ao entrar em contato com um elemento magnético (ímã), fecha os seus contatos, deixando fluir a corrente elétrica entre seus terminais.

Para medir a velocidade do vento, é utilizado um sistema do tipo conchas, que capta a velocidade do vento e transforma em movimento mecânico. Por um sensor magnético na base

e um elemento magnético na parte superior, é possível mensurar a velocidade em que o sensor está girando. Cada volta do sensor é enviado um pulso elétrico para o micro controlador, e este fará a contagem. A parte interna do sensor juntamente com o sensor magnético é mostrado na Figura 15 e a parte superior mostrando o imã é mostrado na Figura 16.

Figura 15 – Anemômetro velocidade do vento – parte interna



Fonte: PHOTOS. [21]

Na parte superior do anemômetro que mede a velocidade do vento, tem-se um elemento magnético que ao completar uma volta completa, entra em contato as duas extremidades do sensor magnético, enviando um pulso elétrico para o micro controlador e assim registrando sua velocidade.

Figura 16 – Anemômetro velocidade do vento – parte superior

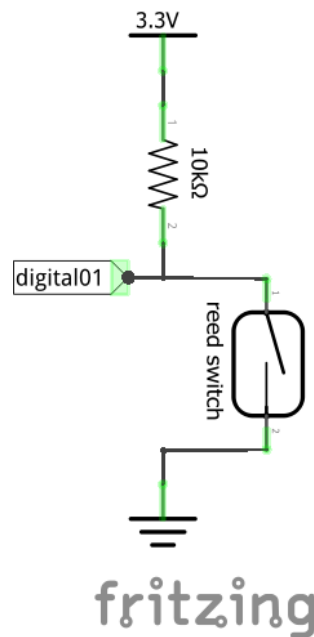


Fonte: PHOTOS. [21]

4.4.1 Esquema elétrico

O esquema elétrico para o funcionamento do anemômetro pode ser visto na Figura 17, com o *reed switch* e um resistor de 10k *push up*, que mantém uma resistência com a tensão positiva evitando que a entrada do sinal flutue entre o positivo e terra, enquanto o sensor não está com o pulso do sinal.

Figura 17 – Esquema elétrico anemômetro



Fonte: O autor (2017).

4.4.2 Calibração

A calibração do anemômetro, consiste em uma constante definida pelo fabricante. São efetuados testes em túneis de vento com o anemômetro e é definido o valor de uma constante para cada modelo de anemômetro. Esta constante é multiplicada na fórmula dentro do micro controlador para mensurar a velocidade correta do anemômetro. Pode-se aferir a velocidade de um anemômetro utilizando outro anemômetro comercial.

4.4.3 Código

Para o funcionamento do anemômetro, foi configurado o microcontrolador para acionar uma função (Contador Anemometro) quando houver uma interrupção na porta digital em que o anemômetro está conectado. Cada vez que houver uma interrupção, é adicionado um valor em uma variável (numRevsAnemometro). Após um período pré-determinado (PERIODO_ANEMOMETRO), o microcontrolador calcula quantas voltas houve no anemômetro neste período e multiplica pela constante de calibração (CTE_CAL_ANEMOMETRO), para determinar a velocidade do mesmo.

4.5 BIRUTA ELETRÔNICA

Para detectar a direção do vento, é utilizado o mesmo princípio do sensor anterior utilizando sensores magnéticos do tipo *reed switch*, juntamente com um guia que manterá o “nariz” do sensor sempre para frente indicando a direção em que o vento chega no sensor conforme mostra a Figura 18.

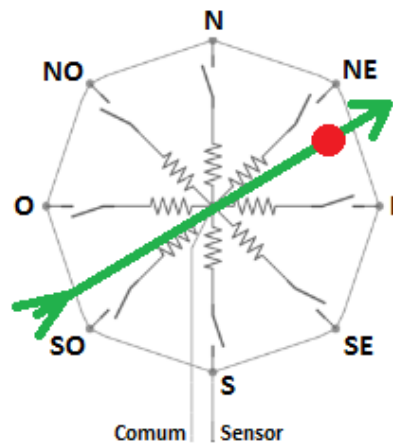
Figura 18 – Biruta direção do vento



Fonte: PCE. [22]

Neste sensor é utilizado oito sensores magnéticos conforme mostrado na Figura 19 e apenas um elemento magnético (ímã) fixado na parte superior do sensor representado pelo círculo vermelho. De acordo com a posição do ponteiro, vai entrar em contato um resistor de determinado valor que vai mensurar a posição que se encontra atualmente.

Figura 19 - Orientação bússola

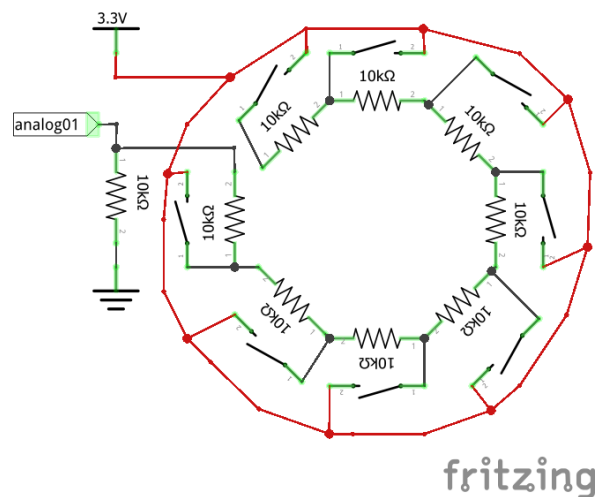


Fonte: O autor (2017).

4.5.1 Esquema elétrico

O esquema elétrico da biruta eletrônica pode ser conferido na Figura 20, com oito *reed switch* ligados alimentados por uma tensão de 3,3volts em paralelo com oito resistências de 10k, e são lidos na porta analógica do micro controlador.

Figura 20 – Esquema elétrico – Biruta eletrônica



Fonte: O autor (2017).

4.5.2 Calibração

De acordo com a posição da biruta, o sensor vai fechar uma resistência e como estão em série, os valores podem ir de 10k a 80k. O Norte foi arbitrado que está posicionado no primeiro resistor de 10k e foi identificado no sensor para quando instalar no local, alinhar

corretamente com o Norte. Foi utilizado um resistor externo de 10k como divisor de tensão e foi medido a tensão de saída conforme mostra o Quadro 1 para conectar na porta do conversor analógico / digital do micro controlador.

Quadro 1 – Relação de resistência em relação à posição do anemômetro

Direção (ângulo)	Resistencia (Ohms)	Tensão (V=3,3v, R=10k)	Valor ADC
0° - N	10k	1,63v	(335 a 327)
45° - NE	20k	1,09v	(207 a 194)
90° - E	30k	0,82v	(151 a 139)
135° - SE	40k	0,65v	(115 a 105)
180° - S	50k	0,54v	(91 a 84)
225° - SO	60k	0,47v	(77 a 72)
270° - O	70k	0,41v	(68 a 63)
315° - NO	80k	0,36v	(61 a 58)

Fonte: O autor (2017).

4.5.3 Código

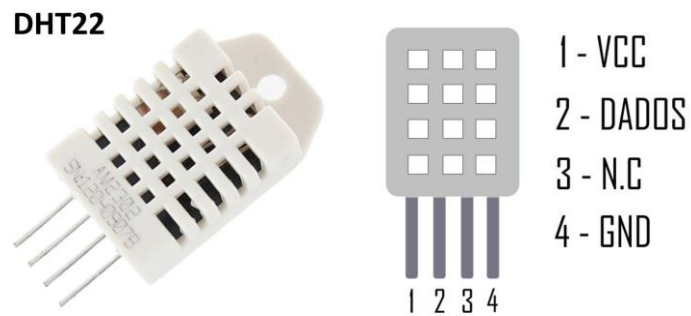
Para detectar a posição da biruta, foi conectada ao microcontrolador na porta analógica e utilizando a função “*analogRead*”, que converte os valores das tensões em valores digitais. De acordo com o valor digital medido e mostrado no Quadro 1 acima, é possível detectar a posição em que se encontra a biruta eletrônica.

4.6 SENSOR TEMPERATURA E UMIDADE DO AR

O sensor de temperatura e umidade utilizado foi o modelo DHT22 que possui saída digital calibrada. Ele utiliza um sensor capacitivo de umidade e um termistor para medir a temperatura do ar. Possui tamanho compacto, baixo consumo e alta precisão.

O sensor DHT22 consegue medir temperaturas de -40° a 80° Celsius, e umidade na faixa de 0 a 100%. Sua precisão para temperatura é de 0,1° e para umidade de 0,1%. Na Figura 21 verifica-se a pinagem de ligação deste sensor e o datasheet encontra-se no link [23].

Figura 21 – Sensor temperatura e umidade DHT22

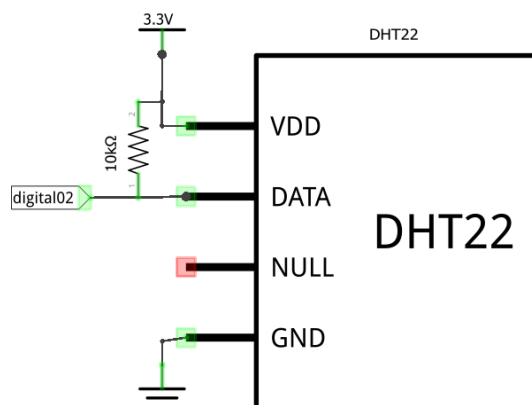


Fonte: FILIPEFLOP (2017). [24]

4.6.1 Esquema elétrico

O sensor DHT22 possui um microcontrolador de 8bits que produz um sinal digital no pino de dados, facilitando a comunicação com o microcontrolador. O fabricante fornece uma biblioteca de código aberto para comunicação entre o sensor e o microcontrolador. A ligação deste sensor é relativamente simples devido ao sinal já vir pronto, ilustrado na Figura 22.

Figura 22 – Esquema elétrico DHT22



fritzing

Fonte: O autor (2017).

4.6.2 Calibração

Por possuir um microcontrolador interno que interpreta os sinais do termistor e do sensor capacitivo de umidade, este sensor dispensa calibração, pois contém seu

encapsulamento fechado e inacessível. O único item que pode interferir na medição deste sensor é ele ficar exposto diretamente ao sol, que pode interferir nas medições. Por isso foi adquirido um abrigo especialmente desenvolvido para sensores de temperatura, que contem aletas que permitem a circulação do ar sem que tenha contato direto com o sensor conforme mostra a Figura 23.

Figura 23 – Abrigo sensor temperatura



Fonte: O autor (2017).

4.6.3 Código

O código para funcionamento deste sensor foi fornecido pelo fabricante (*DHT.h*) com todas funções necessárias para a comunicação deste sensor. Basta definir o tipo de sensor utilizado (`#define DHTTYPE DHT22`), informar o pino em que o sensor está conectado () e inicializar o sensor (`DHT dht(DHTPIN, DHTTYPE, 11);`). Após estas configurações, foi programado para chamar a função necessária para ler a temperatura instantânea (`dht.readTemperature()`) e para ler a umidade relativa do ar (`dht.readHumidity()`).

4.7 PLUVIÔMETRO

O funcionamento do pluviômetro baseia-se em um funil que capta uma área e direciona a chuva captada para uma balança que ao receber água e transborda, movimenta-se como uma gangorra, fechando os contatos de um sensor magnético fazendo fluir um pulso elétrico para a entrada do microcontrolador. Cada pulso do sensor indica uma mudança de

estado da báscula registrando aquela quantidade em milímetros, que é a unidade padrão de precipitação. A Figura 24 mostra a parte superior do pluviômetro com o funil para capturar a chuva e a Figura 25 mostra o interior de um sensor pluviométrico, com a báscula e o elemento magnético.

Figura 24 – Parte superior do pluviômetro



Fonte: O autor (2017).

Figura 25 – Interior do pluviômetro

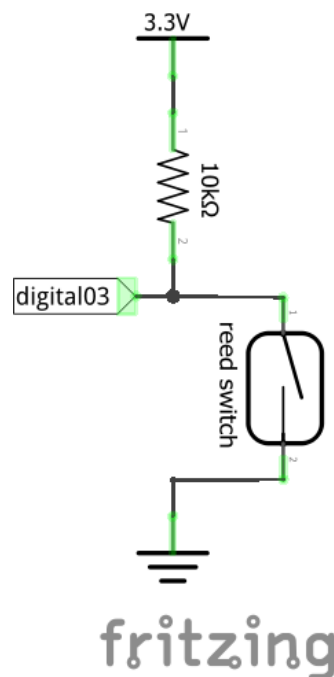


Fonte: O autor (2017).

4.7.1 Esquema elétrico

O esquema elétrico para o funcionamento do pluviômetro é idêntico ao esquema elétrico do anemômetro, com o *reed switch* e um resistor de 10k *push up* que pode ser visto na Figura 26, que mantém uma resistência com a tensão positiva evitando que a entrada do sinal flutue entre positivo e terra, enquanto o sensor não está com o pulso do sinal.

Figura 26 – Esquema elétrico pluviômetro



Fonte: O autor (2017).

4.7.2 Calibração

A calibração deste sensor conta com uma constante fornecida pelo fabricante, onde cada balsa cheia de água muda o estado da gangorra e representa uma quantidade de chuva captada por um pequeno reservatório. O valor fornecido pelo fabricante deste pequeno reservatório é de 0,25mm por batida de balsa.

Pode-se aferir este valor, utilizando o método [25] descrito abaixo:

- A boca do funil que capta a chuva possui um raio de 73mm.
- Área da boca do funil = $\pi * r^2 = 167,33 \text{ cm}^2$ ou $0,016733 \text{ m}^2$
- Quantidade de batidas da balsa com 500 ml de água = 67

$$C_b = V / (B * A) = 0,5 \text{ l} / (67 \text{ batidas} * 0,016733 \text{ m}^2) = 0,4459 \text{ mm} \cong 0,45 \text{ mm}$$

onde C_b é a quantidade de chuva por batida de báscula, V é o volume de água derramada, B é a contagem de batidas por cada 500 ml e A é a área da “boca” do pluviômetro.

Para uma melhor aferição do valor medido, é utilizado um regador com a quantidade de água citada acima, simulando uma queda de água livre igual as chuvas.

Com este experimento, podemos concluir que o volume de água por batida de báscula é diferente do valor fornecido pelo fabricante e como este projeto foi programado para aceitar um valor de calibração, após efetuar testes com aparelhos com maior precisão, poderá ser alterada esta constante de calibração.

4.7.3 Código

O pluviômetro possui código semelhante ao anemômetro, onde foi configurado o microcontrolador para acionar uma função (*contadorPluviometro*) quando houver uma interrupção na porta digital em que o pluviômetro está conectado. Cada vez que houver uma interrupção, é adicionado um valor em uma variável (*numBatidasBascula*). Após um período pré-determinado (*PERIODO_PLUVIOMETRO*), o microcontrolador calcula quantas batidas de báscula houve no pluviômetro neste período e multiplica pela constante de calibração (*CTE_CAL_PLUVIOMETRO*), para determinar quantos milímetros choveu naquele instante.

4.8 MICRO CONTROLADOR

Este é um dos itens mais importante do projeto, o microcontrolador ESP8266 que está incorporado em uma placa de desenvolvimento NODEMCU ESP-12E. Será executado o programa principal que vai se comunicar com os sensores, conectar na internet e enviar os dados para os dispositivos móveis. O código fonte completo do microcontrolador ESP8266 para funcionamento desta estação está no Anexo I.

Dentre as inúmeras vantagens de se utilizar este micro controlador, podemos citar algumas que são muito importantes para este projeto:

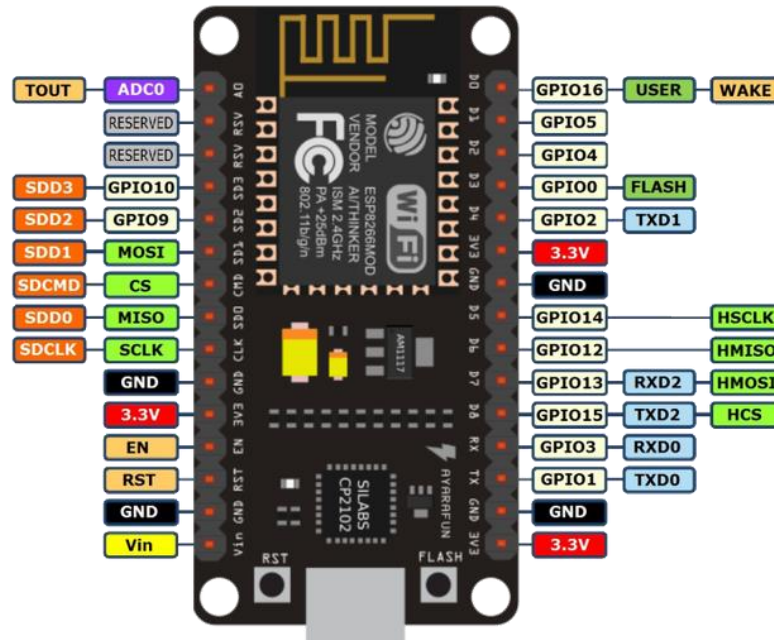
- Compatibilidade com código e bibliotecas do Arduino;
- Microprocessador ARM de 32 bits;
- Suporte integrado para Rede WIFI;
- Antena WiFi embutida, já soldada na placa (livre de mal contatos);
- Compatível com IDE do Arduino;

- 10 entradas digitais GPIOs operando a 3,3V
- 1 entrada analógica GPIO a 1,8 V;
- Baixo custo;
- Tamanho reduzido da placa;
- Baixo consumo de energia;
- Temperatura de trabalho: -40 °C ~ +125 °C;
- Alimentação: 4.5V ~ 9V (10VMAX), USB-powered.

A maior vantagem de utilizar este microcontrolador é sua compatibilidade com códigos e bibliotecas do Arduino, que por sua vez permite acessar muitos exemplos de uso na *internet*, dos mais variados tipos. O suporte nativo à conexões *WiFi* também é uma grande vantagem, dispensando componentes externos.

As portas de entrada e saída são compartilhadas com as funções internas do micro controlador e neste caso necessitam de uma atenção especial ao dimensionar qual entrada vai atender qual função do projeto, conforme mostrado na Figura 27.

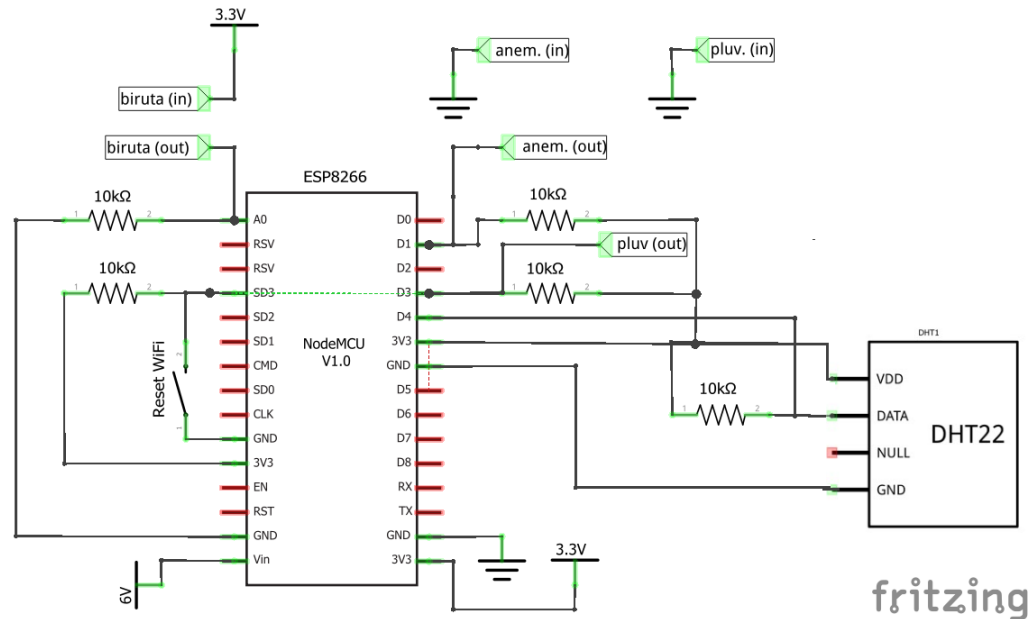
Figura 27 – NODEMCU ESP12-E



Fonte: TRONIXLABS. [26]

Esta placa de desenvolvimento NODEMCU contém reguladores de tensão internos e todos componentes internos que facilitam a ligação desde o início. A tensão de trabalho do microcontrolador é de 3,3v, motivo ao qual os sensores têm que operar nesta tensão. O circuito de ligação dos sensores com o micro controlador pode ser visto na Figura 28.

Figura 28 – Esquema de ligação do microcontrolador com sensores



Fonte: O autor (2017).

O *software* desenvolvido para operar este microcontrolador, foi o da plataforma de programação do Arduino. O Arduino é uma plataforma eletrônica de código aberto baseada em *hardware* e *software* fáceis de usar.

As placas Arduino são capazes de ler entradas digitais e analógicas, além de possuir inúmeras bibliotecas de códigos disponíveis na internet. Ao longo dos anos, o Arduino tem sido o cérebro de milhares de projetos, desde objetos comuns até instrumentos científicos complexos.

Uma comunidade mundial de fabricantes – estudantes, hobistas, artistas, programadores e profissionais, se reuniram em torno desta plataforma de código aberto. Suas contribuições somaram uma incrível quantidade de conhecimento acessível que pode ser de grande ajuda para novatos e especialistas [27].

4.9 TELEGRAM

Este *software* é o responsável em fazer a comunicação entre este projeto e os dispositivos móveis. Telegram é um serviço de mensagens instantâneas baseado na nuvem. A vantagem de utilizar este aplicativo é a compatibilidade com vários sistemas operacionais (Android, iOS, Windows Phone), além de compatibilidade com computadores (Windows, Linux, OSX) [28]. Este *software* possui código aberto, mesmo princípio utilizado neste

trabalho, que possibilita que outras pessoas estudem seu código e reproduzam caso tenham interesse.

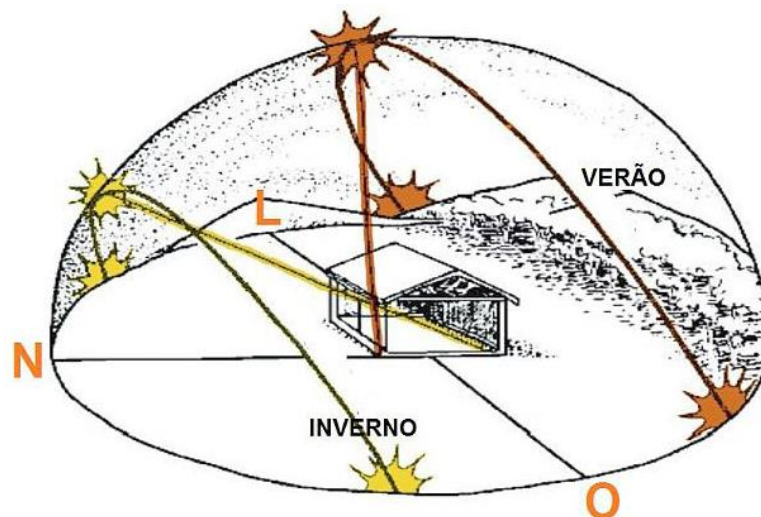
O serviço responsável por fazer a interface entre o Telegram e a estação meteorológica é a possibilidade de criar *bots*, que são *softwares* programados para interagir de acordo com comandos enviados para ele. Por exemplo, se digitar o comando `/vento`, o micro controlador vai interpretar que o usuário quer saber informações sobre o vento e vai retornar para o usuário. É possível incluir *bot* nos grupos do Telegram, para interagir com várias pessoas de uma vez. Mais informações sobre o funcionamento dos *bots* do Telegram podem ser conferidos no *link* [29].

5 ESTUDO DE CASO

A finalidade deste projeto será para uso em um clube de aeromodelismo, para medir as informações sobre a velocidade e direção do vento, temperatura, umidade relativa do ar e chuva. Para alimentar este projeto, será utilizado duas placas solares de 3W cada e será necessário dimensionar o melhor local para fixar estas placas para o melhor aproveitamento do sol.

A posição ideal para s painéis fotovoltaicos no Brasil é voltado para o Norte. O Sol nasce no Leste, sobe se inclinando para o Norte e se põe no Oeste conforme é mostra na Figura 29.

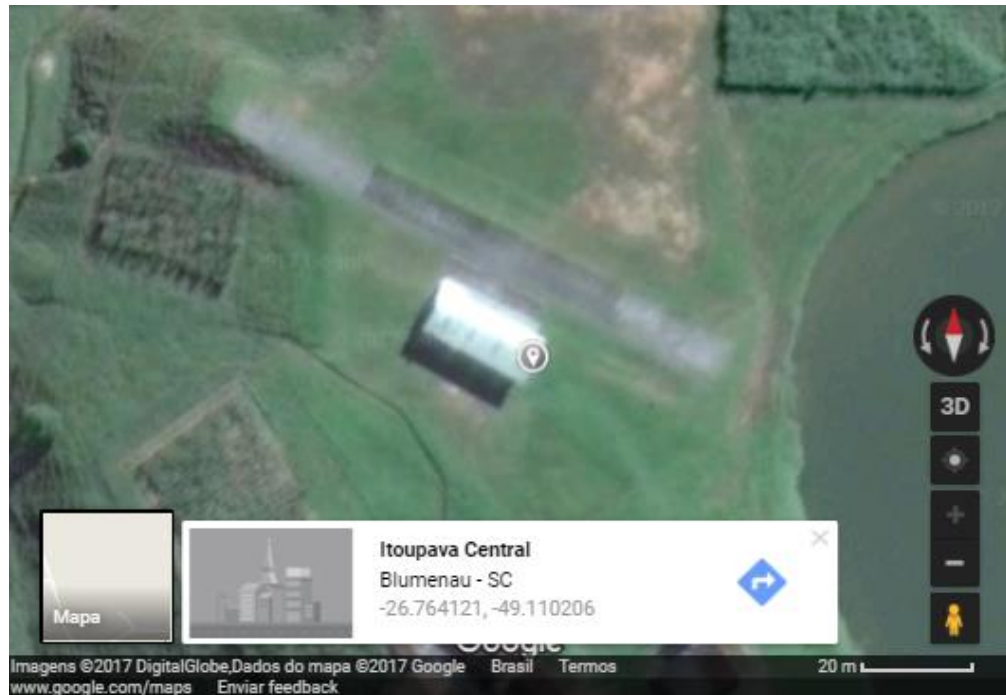
Figura 29 – Percurso do sol durante o dia



Fonte: PORTAL. [30]

A melhor inclinação do painel solar é igual ao ângulo da Latitude do local onde deseja-se instalar o painel. Neste projeto foi utilizado o Clube de Aeromodelismo Aeroblu, em Blumenau, com Latitude -26,764, logo, a inclinação ideal do painel solar neste local será de 26°. A Latitude do local e posição do Norte é mostrado na Figura 30.

Figura 30 – Clube Aeromodelismo Aeroblu - Blumenau



Fonte: GOOGLE (2017).

A posição onde será instalada a estação meteorológica no clube de aeromodelismo Aeroblu está projetada para ficar na parte demarcada com a cor vermelha na frente da Sede e na parte superior do telhado conforme ilustrado na Figura 31, a fim de efetuar as coletas sem interferências externas.

Figura 31 – Local exato da estação meteorológica



Fonte: O autor (2017).

A estação meteorológica foi montada inicialmente no solo para a execução deste projeto, programação e calibração. A posição dos sensores é fundamental para o correto funcionamento da estação meteorológica. Na parte superior está situado o Pluviômetro e abaixo dele o abrigo do sensor de temperatura.

Logo abaixo tem-se a biruta e o anemômetro. Por último está a placa solar e a caixa para abrir o carregador, bateria e componentes eletrônicos. A estação completa com seus sensores pode ser visualizada na Figura 32.

Figura 32 – Estação meteorológica montada no suporte



Fonte: O autor (2017).

Para a sua comunicação, foi registrado um *bot* no programa para dispositivos móveis *telegram*, com o nome de *Aeroblu_bot*, que aceita os seguintes comandos:

/status – informa o status atual

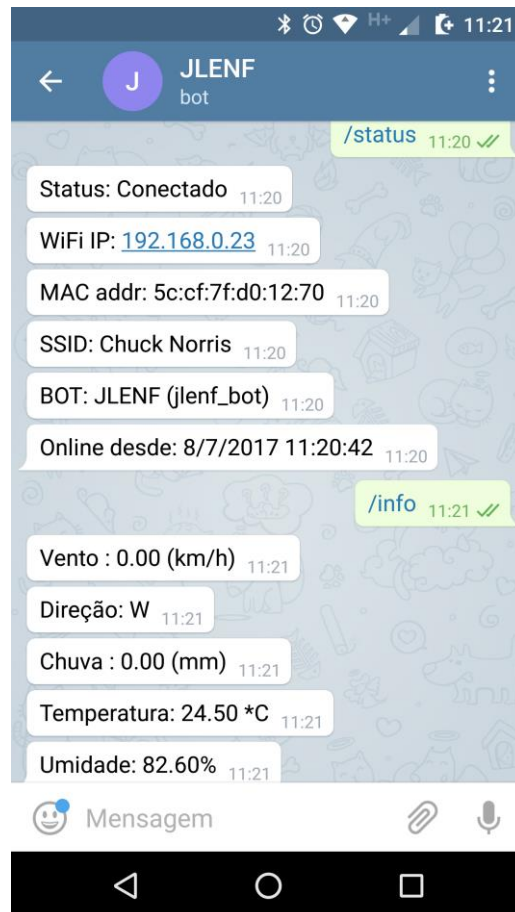
/vento – informa a velocidade e direção do vento

/chuva – informa quantos milímetros choveu na última hora

/temp – informa a temperatura e umidade

Este *bot* do telegram foi inserido no grupo dos sócios do Aeroblu, para que todos os sócios interajam com ele, solicitando informações em qualquer hora, conforme é mostrado na Figura 33.

Figura 33 – Exemplo de utilização do sistema telegram



Fonte: O autor (2017).

O desenvolvimento contemplou todos os módulos abordados, utilizando os itens descritos no quadro 2, com os seus respectivos valores, cotados na data em que este documento foi escrito.

Quadro 2 – custos aproximados de construção da estação meteorológica

DESCRIÇÃO	VALOR	SITE
Placa solar 12v 3W	\$ 19,30	http://aliex.co/vp2ll
Carregador bateria solar	\$ 6,86	http://aliex.co/yua54
Bateria 6v 12A	R\$ 79,00	<u>Unisat comercial</u>
Anemômetro	R\$ 129,95	http://produto.mercadolivre.com.br/MLB-702511982-anemometro-digital-arduino-raspberry-codigo-ccabo-c-10m- JM
Biruta eletrônica	R\$ 128,90	http://produto.mercadolivre.com.br/MLB-687879199-indicador-direco-vento-c-sensor-resistivo-p-arduino-pic- JM
Pluviômetro	R\$ 149,99	http://produto.mercadolivre.com.br/MLB-724251088-pluvimetro-automatico-estaco-meteorologica-arduino-pic- JM
Abrigo sensor temperatura	R\$ 99,90	http://produto.mercadolivre.com.br/MLB-793409156-abrigo-alojamento-externo-sensor-temperatura-dht22-arduino- JM
Sensor temperatura e umidade DHT-22	R\$ 41,48	http://proesi.com.br/catalog/product/view/id/6572/s/sensor-dht22/
ESP8266 Nodemcu	US\$ 5,19	https://www.banggood.com/NodeMcu-Lua-WIFI-Internet-Things-Development-Board-Based-ESP8266-CP2102-Wireless-Module-p-1097112.html
Abrigo eletrônica (bateria, carregador, esp8266)	R\$ 32,00	<u>Elétrica Zata</u>
Suporte ferro aço 1,20mt	R\$ 45,00	Netvis
Mão obra (suportes e abraçadeiras)	R\$ 150,00	Airton Jung

Fonte: O autor (2017).

*Preços em 14/07/2017

6 ANÁLISE DO RESULTADO, CONCLUSÕES E PROPOSTAS FUTURAS

No momento tecnológico em que se vive, as estações meteorológicas estão cada vez mais comuns no dia a dia. É possível adquirir facilmente uma estação meteorológica pela internet, que possui várias funções, tais como anemômetro, temperatura, umidade, radiação solar, pluviômetro, dentre outras. Porém, o principal problema encontrado nas estações meteorológicas é a sua comunicação, pois muitas vezes elas operam de modo *offline*, sem contato com a internet em tempo real, ou a tecnologia é de arquitetura fechada, impedindo a integração com outros sistemas.

Além de possibilitar a construção de uma estação meteorológica nova, é possível utilizar as estações meteorológicas existentes, utilizando seus sensores neste projeto e viabilizando o uso sem depender de programas de terceiros, de forma totalmente aberta.

A vantagem do uso de código aberto é a melhoria contínua onde futuramente pode ser desenvolvido os sensores próprios, tais como o anemômetro, biruta e pluviômetro, para que tenha um melhor rendimento e maior confiabilidade. Além dos sensores utilizados atualmente, poderá ser incluído novos sensores, tais como radiação solar, pressão barométrica, detecção de gases, intensidade luminosa, dentre outros. Poderá também sofrer alterações em seu código fonte, de forma que seja mais otimizado e compatível com as tecnologias que estão surgindo, como por exemplo, o ipv6. Novas versões deste projeto poderão ser encontradas no endereço [31].

Atualmente, o projeto está condicionado a enviar informações da estação meteorológica quando for solicitado via *WiFi*, através de aplicativo de mensagem instantânea e devido a esta configuração, não mantém o registro dos sensores.

Como proposta futura, poderá ser desenvolvido outro método de envio dos dados para um servidor central para o armazenamento destas informações e consequentemente possibilitar efetuar consultas dos registros. Com estes dados armazenados, poderá ser feito um estudo comportamental do tempo na região em que a estação está instalada, bem como uma análise regional dos dados entre as estações. Poderá ser desenvolvido também outro método de conexão, tal como GPRS, para utilizar em lugares onde não tenha sinal de internet *WiFi*.

REFERÊNCIAS

- [1] FRANCISCO, Wagner de Cerqueira e. **Consequências do aquecimento global**. Disponível em: <<http://brasilescola.uol.com.br/geografia/consequencias-do-aquecimento-global.htm>>. Acesso em: 25 jun. 2017.
- [2] BRASIL. Instituto Nacional de Pesquisas Espaciais (INPE). **Quais as consequências do aquecimento global?**. Disponível em: <<http://www.inpe.br/acessoainformacao/node/483>>. Acesso em: 03 fev. 2017.
- [3] SATO, James Hansen and Makiko. **Fraction of "Warm" Stations**. Disponível em: <https://data.giss.nasa.gov/gistemp/warm_stations/>. Acesso em: 05 mar. 2017.
- [4] SÓ GEOGRAFIA. **Climas do Brasil**. Disponível em: <<http://www.sogeografia.com.br/Conteudos/GeografiaFisica/Clima/>>. Acesso em: 05 mar. 2017.
- [5] PREFEITURA de Blumenau. **Enchentes registradas**. 2017. Disponível em: <<http://alertablu.cob.sc.gov.br/p/enchentes>>. Acesso em: 15 jun. 2017.
- [6] INSTITUTO Nacional de Meteorologia. **Estação Meteorológica de Observação de Superfície Automática**. Disponível em: <<http://www.inmet.gov.br/portal/index.php?r=estacoes/estacoesAutomaticas>>. Acesso em: 15 jun. 2017.
- [7] CECHIN, Lucas. **O que são anemômetros**. 2011. Disponível em: <https://pt.slideshare.net/lucas_cechin/o-que-so-anemmetros>. Acesso em: 22 abr. 2017.
- [8] HIDRAULIS. **Pluviômetro**. Disponível em: <<http://www.pluviometros.com.br>>. Acesso em: 22 abr. 2017.
- [9] MARTINS, Samantha. **Como funcionam as medições de precipitação?**. 2015. Disponível em: <<http://meteoropole.com.br/2015/05/como-funcionam-as-medicoes-de-precipitacao/>>. Acesso em: 12 mar. 2017.
- [10] MARTINS, Samantha. **Dúvida do leitor: capacidade de um pluviômetro**. 2016. Disponível em: <<http://meteoropole.com.br/2016/04/duvida-do-leitor-capacidade-de-um-pluviometro/>>. Acesso em: 22 abr. 2017.
- [11] BROOKE Clarke. **Components**. 2012. Disponível em: <<http://www.prc68.com/I/UltimeterWeatherStation.shtml>>. Acesso em: 12 mar. 2017.
- [12] VAREJÃO-SILVA, Mário Adelmo; INSTITUTO NACIONAL DE METEOROLOGIA, (Brasil). **Meteorologia e climatologia**. 2. ed. Brasília, D.F : Instituto Nacional de Meteorologia, 2001. xvi, 515p, il.
- [13] TUBELIS. Antonio; NASCIMENTO, Fernando Jose Lino do. **Meteorologia descritiva: fundamentos e aplicações brasileiras**. São Paulo : Nobel, 1986. 374 p, il.

- [14] CLEAN Enviroment Brasil. **Sensor de Temperatura e Umidade Relativa do Ar**. 2012. Disponível em: <<http://www.clean.com.br/Blog/Post/waterlog-h-380>>. Acesso em: 12 mar. 2017.
- [15] GIOVELLI, Igor Ari. **Sítio de Estação Meteorológica**. 2007. Disponível em: <<https://www.agsolve.com.br/pdf/artigos/sitio.pdf>>. Acesso em: 28 abr. 2017.
- [16] ACCEL System. **Automatic Weather Station Data Logger: Observe-Met**. Disponível em: <<http://accel-systems.co.in/work/automatic-weather-station-data-logger-observe-met>>. Acesso em: 28 abr. 2017.
- [17] PERROTTA, Thiago Barroso. **Você conhece as licenças de software livre? já está na hora de saber né?**. 2013. Disponível em: <<http://sejalivre.org/voce-conhece-as-licencas-de-software-livre-ja-esta-na-hora-de-saber-ne/>>. Acesso em: 17 mai. 2017.
- [18] GOOGLE Maps. **Clube de Aeromodelismo Asas do Vale**. 2017. Disponível em: <<https://www.google.com.br/maps/place/Clube+de+Modelismo+Asas+do+Vale/@-26.916226,-48.9172567,680m/data=!3m1!1e3!4m5!3m4!1s0x94df237761c05c77:0x63cd0a916e83f650!8m2!3d-26.916226!4d-48.915068>>. Acesso em: 17 mai. 2017.
- [19] ALIEXPRESS. **3 W 12 V Pannel Solar Flexível China Placa de Silício Policristalino de Células Solares DIY 145x145mm Pannel Solars Charger**. Disponível em: <<http://aliex.co/vp2ll>>. Acesso em: 18 abr. 2017.
- [20] ALIEXPRESS. **3A 6 V 12 V PWM Controlador de Luz Do Pannel Solar Regulador de Carga Da Bateria para o Sistema de Pannel Solar Carregador Solar Inteligente controlador**. Disponível em: <<http://aliex.co/yua54>>. Acesso em: 18 abr. 2017.
- [21] PHOTOS of the inside of the maplin & fine offset weather station modules. Disponível em: <<http://www.philpot.me/weatherinsider.html>>. Acesso em: 18 mai. 2017.
- [22] PCE Instruments UK Ltd. **Biruta**. Disponível em: <<http://www.industrial-needs.com/technical-data/weather-station-fws20.html>>. Acesso em: 18 mai. 2017.
- [23] AOSONG. **Temperature and humidity module AM2302 Product Manua**. 11p. Disponível em: <<http://akizukidenshi.com/download/ds/aosong/AM2302.pdf>>. Acesso em: 23 mar. 2017.
- [24] FILIPEFLOP. **Sensor temperatura e umidade DHT22**. Disponível em: <<http://www.filipeflop.com>>. Acesso em: 23 mar. 2017.
- [25] ESTAÇÃO meteorológica modular. **Teste experimental do pluviômetro: Teste de quantidade de chuva por batida de báscula**. Disponível em: <http://cta.if.ufrgs.br/projects/estacao-meteorologica-modular/wiki/Teste_de_quantidade_de_chuva_por_batida_de_b%C3%A1scula>. Acesso em: 23 mar. 2017.
- [26] TRONIXLABS. **NODEMCU ESP12-E**. Disponível em: <<https://tronixlabs.com.au/>>. Acesso em: 18 mai. 2017.

[27] <https://www.arduino.cc/en/Guide/Introduction>

[28] WIKIPÉDIA. **Telegram (aplicativo)**. Disponível em: [<https://pt.wikipedia.org/wiki/Telegram_\(aplicativo\)>](https://pt.wikipedia.org/wiki/Telegram_(aplicativo)). Acesso em: 23 mar. 2017.

[29] BOTS: An introduction for developers. Disponível em: [<https://core.telegram.org/bots>](https://core.telegram.org/bots). Acesso em: 23 mar. 2017.

[30] PORTAL Solar. **A posição ideal para os seus painéis fotovoltaicos no Brasil é voltado para o Norte**. Disponível em: [<http://www.portalsolar.com.br/a-melhor-direcao-do-painel-solar-fotovoltaico.html>](http://www.portalsolar.com.br/a-melhor-direcao-do-painel-solar-fotovoltaico.html). Acesso em: 18 mai. 2017.

[31] LENFERS, Jairo. **Estação meteorológica**. 2017. Disponível em: [<https://github.com/JLENF/estacao-meteorologica>](https://github.com/JLENF/estacao-meteorologica). Acesso em 2017. Acesso em: 10 jun. 2017.

ANEXO 1 – CÓDIGO MICROCONTROLADOR ESP8266

```

/*
*****
*   Estação meteorológica auto sustentável em código aberto           *
*   Projeto desenvolvido no TCC de Engenharia Elétrica - FURB         *
*   Para obtenção do título de Engenheiro Eletricista                 *
*   Data: 12/03/2017                                                  *
*   Última alteração: 09/07/2017                                       *
*   Changelog:                                                         *
*   - incluído função para mostrar a data e hora atual (/hora)        *
*   - incluído função para mostrar temp. e umid. máximas (/max)       *
*   - incluído função para mostrar pluviômetro por uma hora (/chuva) *
*   *                                                                    *
*   Este projeto é de código aberto e utiliza partes de código        *
*   de outros projetos de código aberto:                               *
*   - http://cta.if.ufrgs.br                                           *
*   - http://air.imag.fr/mediawiki/index.php/SEN-08942                *
*   - https://github.com/PaulStoffregen/Time                          *
*   - https://github.com/esp8266/Arduino/issues/313                   *
*   *                                                                    *
*   Bibliotecas necessárias:                                           *
*   - https://github.com/arduino-libraries/NTPClient                  *
*   - https://github.com/PaulStoffregen/Time                          *
*   - https://github.com/adafruit/DHT-sensor-library                  *
*   - https://github.com/Gianbacchio/ESP8266-TelegramBot              *
*   *                                                                    *
*   Para compilar o ESP8266 no Arduino, cole a URL abaixo            *
*   em Arquivo -> Preferências -> URL:                                *
*   http://arduino.esp8266.com/stable/package_esp8266com_index.json  *
*   *                                                                    *
*   Você pode contribuir com este projeto e acompanhar seu           *
*   desenvolvimento no endereço oficial do projeto:                   *
*   http://www.github.com/jlenf/                                       *
*   Autor: Jairo Lenfers - jairolenfers@gmail.com                     *
*   *                                                                    *
*****
*/

// Bibliotecas externas
#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>
#include <ESP8266TelegramBOT.h>
#include <DHT.h>
#include <NTPClient.h>
#include <WiFiUdp.h>
#include <TimeLib.h>

// Definição de constantes de calibração
#define CTE_CAL_ANEMOMETRO 0.9011 // 1 rev/segundo = 0.9011 kph
#define CTE_CAL_PLUVIOMETRO 0.4459 // 1 batida = 0.4459 mm
#define DHTTYPE DHT22 // Modelo do sensor de temperatura

// Definição de constantes de WiFi
const char* ssid = "rede_wifi"; // Nome da rede WiFi
const char* password = "*****"; // Senha da rede WiFi

// Definição de constantes do Telegram BOT
// Utilize o BOT Father para criar seu token: https://core.telegram.org/bots
#define BOTtoken "123456789:*****" // Token do BOT
#define BOTname "BOTNAME" // Nome do BOT
#define BOTusername "user_bot" // usuário do BOT
String numero_tel = "12345678"; // Código telefone que recebe primeira msg (ver na
serial)
// Período entre as medidas em milissegundos
#define PERIODO_ANEMOMETRO 5000
#define PERIODO_DIR_VENTO 5000

```

```

#define PERIODO_PLUVIOMETRO 5000

// Pinos para conexão com o ESP8266
#define ANEMOMETRO_PIN 5 // Digital D1 Nodemcu
#define PLUVIOMETRO_PIN 4 // Digital D2 Nodemcu
#define DIR_VENTO_PIN A0 // Analog A0 Nodemcu
#define DHTPIN 2 // Digital D4 Nodemcu

// Variáveis globais
float tempf ,humf, temp_maxf, hum_maxf;
String temp, hum, temp_max, hum_max, data_temp_max, data_hum_max;
int16_t utc = -3; //UTC -3:00 Brasil
String uptime_data;
String uptime_hora;
String atual_data;
String atual_hora;
String clientMac = "";
unsigned char mac[6];
int Bot_mtbs = 1000;
long Bot_lasttime;
bool Start = false;
const long intervalo_alerta = 60000; // = 60 segundos
double volume_minutos[60]; // volume pluviômetro últimos 60 minutos
int ult_minuto = 0; // qual o ultimo minuto antes de mudar de estado
double volume_hora_mm;

// Variáveis para incrementação
volatile int numRevsAnemometro = 0;
volatile int numBatidasBascula = 0;

// Variáveis para realização do polling
unsigned long proximaMedidaAnemometro = 0;
unsigned long proximaMedidaPluviometro = 0;
unsigned long proximaMedidaDirVento = 0;
unsigned long tempo = 0;
unsigned long previousMillis = 0; // diferenca entre tempo de aquisição da temp.
const long interval = 2000;

// Variáveis para converter para String para mostrar no Telegram
String S_vel_med;
String S_local_ip;
String S_ssid;
String S_direcao;
String S_volume_mm;
String S_volume_hora_mm;
String S_bot_name;
String S_bot_username;

// Inicialização gerais
DHT dht(DHTPIN, DHTTYPE, 11); // 11 works fine for ESP8266
TelegramBOT bot(BOTtoken, BOTname, BOTusername);
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "a.st1.ntp.br", utc*3600, 60000);

// Direção do vento, valores de leitura para diferenciar cada direção:
// int adc[8] = {26, 45, 77, 118, 161, 196, 220, 256};
int adc[8] = {104, 180, 308, 472, 644, 784, 880, 1024};
// Relação entre os valores analógicos lidos e o que eles representam
// Para facilitar pode-se usar a biblioteca String
char *direcoes[8] = {"W","NW","N","SW","NE","S","SE","E"};
int direcaoInicial = 0;

void setup() {
    pinMode(LED_BUILTIN, OUTPUT);
    digitalWrite(LED_BUILTIN, HIGH);
    Serial.begin(9600);
    dht.begin(); // inicia o sensor de temperatura

```

```

// Definição de entradas e saídas
pinMode(ANEMOMETRO_PIN, INPUT_PULLUP);
pinMode(PLUVIOMETRO_PIN, INPUT_PULLUP);
attachInterrupt(digitalPinToInterrupt(ANEMOMETRO_PIN), contadorAnemometro,
FALLING);
attachInterrupt(digitalPinToInterrupt(PLUVIOMETRO_PIN), contadorPluviometro,
FALLING);

// Conecta na rede WiFi
Serial.println();
Serial.println();
Serial.print("Conectando na rede ");
Serial.println(ssid);
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) { // Enquanto tenta conectar, coloca um
caracter . na serial
    delay(500);
    Serial.print(".");
}
Serial.println("");
Serial.println("WiFi conectado");
Serial.println(WiFi.localIP()); // Mostra o ip que adquiriu

WiFi.macAddress(mac); // MAC Address que adquiriu
clientMac += macToStr(mac);

bot.begin(); // Inicia o bot telegram
Serial.println("BOT started");
pinMode(2, OUTPUT); // led do wifi conectado
bot.sendMessage(numero_tel,"Estação 01 - Conectada!", ""); // envia mensagem para
numero padrao dizendo que ligou

// Inicia NTP
delay(2000);
timeClient.begin();
timeClient.update();
setSyncProvider(&ntpSyncProvider);

// Pega a data e hora que ligou o sistema
delay(2000);
uptime_data_hora();

// Converte as variaveis para String para mostrar no Telegram
S_local_ip = WiFi.localIP().toString();
S_ssid = String(ssid);
S_bot_name = String(BOTname);
S_bot_username = String(BOTusername);
S_bot_name.concat(" (");
S_bot_name.concat(S_bot_username);
S_bot_name.concat(")");

void loop() { // Loop principal

// Verifica se tem requisição do telegram
if (millis() > Bot_lasttime + Bot_mtbs) {
    bot.getUpdates(bot.message[0][1]); // Chama a API GetUpdates
    Bot_ExecMessages(); // Responde as mensagens com ECHO
    Bot_lasttime = millis();

// Realizando o polling
tempo = millis();

if (tempo >= proximaMedidaAnemometro) {
    Serial.print("Vento (km/h): ");Serial.println(calcVelocidadeVento(), 2);
    proximaMedidaAnemometro = tempo + PERIODO_ANEMOMETRO;
if (tempo >= proximaMedidaDirVento) {
    Serial.print("Direcao: ");Serial.println(calcDirecaoVento());
    proximaMedidaDirVento = tempo + PERIODO_DIR_VENTO;

```

```

    }
    if (tempo >= proximaMedidaPluviometro) {
        Serial.print("Chuva (mm): "); Serial.println(calcQuantidadeChuva(), 3);
        proximaMedidaPluviometro = tempo + PERIODO_PLUVIOMETRO;
    }
    gettemperature(); // Fica atualizando a temperatura para variaveis globais
}

void uptime_data_hora() { // Funcao para pegar data e hora que o sistema ligou
    time_t t = now();
    uptime_hora = String(hour(t));
    uptime_hora.concat(":");
    uptime_hora.concat(minute(t));
    uptime_hora.concat(":");
    uptime_hora.concat(second(t));
    uptime_data = String(day(t));
    uptime_data.concat("/");
    uptime_data.concat(month(t));
    uptime_data.concat("/");
    uptime_data.concat(year(t));
    uptime_data.concat(" ");
    uptime_data.concat(uptime_hora);
}

void atual_data_hora() { // Funcao para pegar a data e hora atual
    time_t a = now();
    atual_hora = String(hour(a));
    atual_hora.concat(":");
    atual_hora.concat(minute(a));
    atual_hora.concat(":");
    atual_hora.concat(second(a));
    atual_data = String(day(a));
    atual_data.concat("/");
    atual_data.concat(month(a));
    atual_data.concat("/");
    atual_data.concat(year(a));
    atual_data.concat(" ");
    atual_data.concat(atual_hora);
}

String macToStr(const uint8_t* mac){ // Função para pegar o MAC Address e converter
em String
    String result;
    for (int i = 0; i < 6; ++i) {
        result += String(mac[i], 16);
        if (i < 5)
            result += ':';
    }
    return result;
}

time_t ntpSyncProvider() { // Função para o NTP
    return timeClient.getEpochTime();
}

void contadorAnemometro() { // Funções de callback de interrupção anemometro
    numRevsAnemometro++;
    Serial.println("===== INTERRUPCAO ANEMOMETRO ====="); //
    DEBUG para serial quando recebeu interrupcao
}

void contadorPluviometro() { // Funções de callback de interrupção pluviometro
    numBatidasBascula++;
    Serial.println("===== INTERRUPCAO PLUVIOMETRO ====="); //
    DEBUG para serial quando recebeu interrupcao
double calcVelocidadeVento() { // Função para converter as aquisições com o periodo
informado
    double velocidadeMedia;

```

```

    velocidadeMedia = numRevsAnemometro;
    velocidadeMedia *= 1000.0*CTE_CAL_ANEMOMETRO;
    velocidadeMedia /= PERIODO_ANEMOMETRO;
    // Resetando contador de pulsos do anemometro
    numRevsAnemometro = 0;
    S_vel_med = String(velocidadeMedia);
    return velocidadeMedia;
}

char* calcDirecaoVento() { // Função para transformar os valores analogicos em
direções do vento
    int valor, x;
    valor = analogRead(DIR_VENTO_PIN);
    for (x = 0; x < 8; x++) {
        if (adc[x] >= valor)
            break;
    }
    // Ajustando direção inicial
    x = (x + direcaoInicial) % 8;
    S_direcao = String(direcoes[x]);
    return direcoes[x];
}

void pluv_ultima_hora(double vol_5s){ // Funcao para incluir os valores do
pluviometro na última hora
    time_t u = now();
    int now_min = minute(u);
    if (ult_minuto != now_min){
        volume_minutos[now_min] = 0; // inicia uma nova contagem no minuto atual
quando ele chegar
        ult_minuto = now_min;
    }
    volume_minutos[now_min] += vol_5s;
}

void calc_pluv_ultima_hora(){ // Função para somar todos os valores da última hora
do pluviometro
    int x;
    volume_hora_mm = 0;
    for (x = 0; x < 60; x++) {
        volume_hora_mm += volume_minutos[x];
        Serial.print(x);Serial.print(" ");Serial.println(volume_minutos[x]); //
DEBUG
    }
    S_volume_hora_mm = String(volume_hora_mm);
}

void zera_pluv_ultima_hora(){ // Função para zerar os valores do pluviometro da
última hora
    int x;
    volume_hora_mm = 0;
    for (x = 0; x < 60; x++) {
        volume_minutos[x] = 0;
    }
    S_volume_hora_mm = String(volume_hora_mm);
}

double calcQuantidadeChuva(){ // Função para calcular a quantidade de chuva
instantânea
    double volumeMedio;
    volumeMedio = numBatidasBascula;
    volumeMedio *= 1000.0*CTE_CAL_PLUVIOMETRO;
    volumeMedio /= PERIODO_PLUVIOMETRO;
    numBatidasBascula = 0;
    pluv_ultima_hora(volumeMedio); // envia para funcao para calcular a media da
ultima hora
    S_volume_mm = String(volumeMedio); // transforma em String para funcionar no
Telegram
    return volumeMedio;
}

```



```

}

void gettemperature() { // Função para pegar os dados de temperatura e umidade do
sensor DHT22
    // Espera dois segundos entre as medicoes.
    // se a diferenca do tempo entre o tempo atual e o ultimo tempo que leu
    // o sensor for maior que o intervalo setado, vai ler o sensor
    // Funciona melhor que delay
    unsigned long currentMillis = millis();
    if(currentMillis - previousMillis >= interval) {
        // salva o ultimo tempo que leu o sensor
        previousMillis = currentMillis;
        humf = dht.readHumidity();          // Faz a leitura da Umidade (percentual)
        tempf = dht.readTemperature();      // Faz a leitura da Temperatura
        if (isnan(humf) || isnan(tempf)) { // Função para detectar se houve erro na
leitura do sensor
            Serial.println("Falha ao ler o sensor DHT22!");
            return;
        }else{
            temp = String(tempf);
            hum = String(humf);
            if (tempf >= temp_maxf){ // Detecta se a temperatura é maior que a maxima
registrada
                atual_data_hora(); // Se for, anota a data e hora que ocorreu
                data_temp_max = atual_data;
                temp_max = String(tempf);
                temp_maxf = tempf;
            }
            if (humf >= hum_maxf){ // Detecta se a umidade é maior que a maxima
registrada
                atual_data_hora(); // Se for, anota a data e hora que ocorreu
                data_hum_max = atual_data;
                hum_max = String(humf);
                hum_maxf = humf;
            }
        }
    }
}

void Bot_ExecMessages() { // Função que processa as mensagens do telegram
for (int i = 1; i < bot.message[0][0].toInt() + 1; i++) {
    bot.message[i][5]=bot.message[i][5].substring(1,bot.message[i][5].length());
    Serial.println(bot.message[i][5]); // DEBUG temporario - para saber o que ele
esta recebendo, para saber como tratar
    if (bot.message[i][5] == "status") {
        bot.sendMessage(bot.message[i][4], "Status: Conectado", "");
        bot.sendMessage(bot.message[i][4], "WiFi IP: " + S_local_ip, "");
        bot.sendMessage(bot.message[i][4], "MAC addr: " + clientMac, "");
        bot.sendMessage(bot.message[i][4], "SSID: " + S_ssid, "");
        bot.sendMessage(bot.message[i][4], "BOT: " + S_bot_name, "");
        bot.sendMessage(bot.message[i][4], "Online desde: " + uptime_data, "");
    }
    if (bot.message[i][5] == "vento") {
        bot.sendMessage(bot.message[i][4], "Vento : " + S_vel_med + " (km/h)", "");
        bot.sendMessage(bot.message[i][4], "Direção: " + S_direcao, "");
    }
    if (bot.message[i][5] == "chuva") {
        calc_pluv_ultima_hora();
        bot.sendMessage(bot.message[i][4], "Chuva (instantânea): " + S_volume_mm + "
(mm)", "");
        bot.sendMessage(bot.message[i][4], "Chuva (ultima hora): " + S_volume_hora_mm
+ " (mm)", "");
    }
    if (bot.message[i][5] == "zera") {
        zera_pluv_ultima_hora();
        calc_pluv_ultima_hora();
        bot.sendMessage(bot.message[i][4], "Informações da chuva na última hora
zeradas!", "");
    }
}

```

```

        bot.sendMessage(bot.message[i][4], "Chuva (ultima hora): " + S_volume_hora_mm
+ " (mm)", "");
    }
    if (bot.message[i][5] == "temp") {
        gettemperature();
        bot.sendMessage(bot.message[i][4], "Temperatura: " + temp + " *C", "");
        bot.sendMessage(bot.message[i][4], "Umidade: " + hum + "%", "");
    }
    if (bot.message[i][5] == "max") {
        gettemperature();
        bot.sendMessage(bot.message[i][4], "Temperatura (max): " + temp_max + "
*C", "");
        bot.sendMessage(bot.message[i][4], "Data: " + data_temp_max, "");
        bot.sendMessage(bot.message[i][4], "-----", "");
        bot.sendMessage(bot.message[i][4], "Umidade (max): " + hum_max + "%", "");
        bot.sendMessage(bot.message[i][4], "Data: " + data_hum_max, "");
    }
    if (bot.message[i][5] == "info") {
        bot.sendMessage(bot.message[i][4], "Vento : " + S_vel_med + " (km/h)", "");
        bot.sendMessage(bot.message[i][4], "Direção: " + S_direcao, "");
        bot.sendMessage(bot.message[i][4], "Chuva : " + S_volume_mm + " (mm)", "");
        bot.sendMessage(bot.message[i][4], "Temperatura: " + temp + " *C", "");
        bot.sendMessage(bot.message[i][4], "Umidade: " + hum + "%", "");
    }
    if (bot.message[i][5] == "conf") {
        numero_tel = bot.message[i][4];
        bot.sendMessage(bot.message[i][4], "Configurado alerta para mostrar aqui", "");
    }
    if (bot.message[i][5] == "hora") {
        atual_data_hora();
        bot.sendMessage(bot.message[i][4], "Data/Hora atual (ntp -3):", "");
        bot.sendMessage(bot.message[i][4], atual_data, "");
    }
    if (bot.message[i][5] == "start") { // mensagem padrão quando adiciona o bot,
mostrando as funções
        String wellcome = "Estação Meteorológica WiFi";
        String wellcome1 = "/status : informa o status atual";
        String wellcome2 = "/vento : velocidade e direção do vento";
        String wellcome3 = "/chuva : milímetros que choveu última hora";
        String wellcome4 = "/zera : zerar informações da chuva na última hora";
        String wellcome5 = "/temp : temperatura e umidade";
        String wellcome6 = "/max : mostra temperatura e umidade maximas";
        String wellcome7 = "/info : mostra todas informações dos sensores";
        String wellcome8 = "/conf : configura onde vai mostrar os alertas";
        String wellcome9 = "/hora : mostra a hora atual do sistema (ntp)";
        bot.sendMessage(bot.message[i][4], wellcome, "");
        bot.sendMessage(bot.message[i][4], wellcome1, "");
        bot.sendMessage(bot.message[i][4], wellcome2, "");
        bot.sendMessage(bot.message[i][4], wellcome3, "");
        bot.sendMessage(bot.message[i][4], wellcome4, "");
        bot.sendMessage(bot.message[i][4], wellcome5, "");
        bot.sendMessage(bot.message[i][4], wellcome6, "");
        bot.sendMessage(bot.message[i][4], wellcome7, "");
        bot.sendMessage(bot.message[i][4], wellcome8, "");
        bot.sendMessage(bot.message[i][4], wellcome9, "");
        Start = true;
    }
    Serial.println(bot.message[i][4]);
}
bot.message[0][0] = ""; // todas mensagens foram respondidas
}

```