Universidade Federal de Ouro Preto - UFOP
Escola de Minas
Colegiado do curso de Engenharia de Controle e Automação - CECAU

Gustavo Ferreira Rodrigues

Desenvolvimento de um sistema de automação predial para gestão e monitoramento em tempo real do sistema hidráulico

Monografia de Graduação em Engenharia de Controle e Automação

Ouro Preto, 2019

Gustavo Ferreira Rodrigues

Desenvolvimento de um sistema de automação predial para gestão e monitoramento em tempo real do sistema hidráulico

Monografia apresentada ao Curso de Engenharia de Controle e Automação da Universidade Federal de Ouro Preto como parte dos requisitos para a obtenção do Grau de Engenheiro de Controle e Automação.

Orientador: Prof. Dr. Antonio Santos Sánchez

Ouro Preto, 2019

R696d Rodrigues, Gustavo Ferreira.
Desenvolvimento de um sistema de automação predial para gestão e monitoramento em tempo real do sistema hidráulico [manuscrito] / Gustavo Ferreira Rodrigues. - 2019.

73f.: il.: color; grafs; tabs; mapas.

Orientador: Prof. Dr. Antonio Santos Sánchez.

Monografia (Graduação). Universidade Federal de Ouro Preto. Escola de Minas. Departamento de Engenharia de Controle e Automação e Técnicas Fundamentais.

1. Automação Predial. 2. Água. 3. Reaproveitamento. I. Sánchez, Antonio Santos. II. Universidade Federal de Ouro Preto. III. Titulo.

CDU: 681.5

Catalogação: ficha.sisbin@ufop.edu.br

Universidade Federal de Ouro Preto
Escola de Minas
Departamento de Engenharia de Produção, Administração e Economia – DEPRO
Campus Universitário Morro do Cruzeiro
35400.000 – Ouro Preto, MG
Tel: 31-3559-1540 – Fax: 3559-1555 – E-mail: depro@depro.em.ufop.br

ATA DE DEFESA

Aos vinte e nove dias do mês de maio de dois mil e dezenove, às 08h00, no auditório da Escola de Minas do Campus Morro do Cruzeiro, foi realizada a defesa de Trabalho Final de Graduação - Monografia pelo aluno **Gustavo Ferreira Rodrigues**, sendo a banca examinadora constituída pelo professor Antonio Santos Sánchez (orientador), e pelos professores Agnaldo José da Rocha Reis e André Luis Silva. O aluno apresentou a monografia intitulada: "***Desenvolvimento de um sistema de automação predial para gestão e monitoramento em tempo real do sistema hidráulico***". A banca examinadora deliberou, por unanimidade, pela aprovação do aluno, concedendo-lhe o prazo de quinze dias para incorporação, no texto final, das alterações sugeridas. Na forma regulamentar, foi lavrada a presente ata, que é assinada pelos membros da banca examinadora e pelo aluno.

Ouro Preto, 29 de maio de 2019.

Antonio Santos Sánchez
Orientador

Agnaldo José da Rocha Reis
Membro

André Luis Silva
Membro

Gustavo Ferreira Rodrigues
Aluno

Resumo

Grande parte da água distribuída é perdida pelo caminho por meio de vazamentos ou roubos. Uma parte dessa água que chega em um edifício pode ser reaproveitada para fins não potáveis, tais como descarga sanitária, irrigação de jardins e lavagens de carros e pisos. Gerir de forma eficiente as perdas bem como utilizar técnicas de reuso de água além de trazerem benefícios para o meio ambiente com a diminuição da demanda nas águas das superfícies, representa uma economia efetiva para os consumidores residenciais e comerciais.

Ademais, para monitorar ou reutilizar qualquer tipo de recurso, principalmente água, precisamos de dados precisos e em tempo real para realizarmos uma gestão inteligente e efetiva.

Diante disso, o presente projeto propõe o desenvolvimento de um sistema economicamente viável de automação predial, que seja capaz de realizar o monitoramento e gestão deste recurso ambiental. Para isto, o sistema deverá ser constituído de hardware e software capaz de fazer o monitoramento em tempo real da água de forma inteligente e eficiente.

O projeto em questão foi dividido em duas etapas. A primeira delas foi um teste de viabilidade do sistema de mediação do consumo de água em alguns pontos de uma residência. Os pontos escolhidos foram aqueles que são os maiores produtores de água cinza e serviram como justificativa para a importância de um sistema de reaproveitamento de água. A segunda etapa foi o desenvolvimento de uma plataforma intuitiva para monitoramento e gestão em tempo real de um sistema hidráulico de uma casa.

Assim, espera-se que este projeto contribua para a preservação dos recursos naturais, bem como auxilie na economia para o consumidor final e conscientize a população sobre a necessidade de preservar este recurso.

Palavras-chaves:água, automação predial, economia, reaproveitamento, sustentabilidade.

Abstract

Much of the water distributed is lost along the way through leaks or thefts. A part of this water that arrives in a building can be reused for non-potable purposes such as sanitary discharge, irrigation of gardens and car washes and floors. Efficiently managing losses as well as using water reuse techniques in addition to bringing benefits to the environment with decreasing of demand in surface waters represents an effective economy for residential and commercial consumers.

In addition, to monitor or reuse any type of resource, especially water, we need accurate and real-time data to perform an intelligent and effective management.

In view of this, the present project proposes the development of an economically feasible system of building automation, which is capable of performing the monitoring and management of this environmental resource. For this, the system must be made up of hardware and software capable of real-time monitoring of water in an intelligent and efficient way.

The project in question was divided into two stages. The first of these was a viability test of the water consumption mediation system at some points of a residence. The points chosen were those that are the largest producers of gray water and served as justification for the importance of a water reuse system. The second step was the development of an intuitive platform for real-time monitoring and management of a home's hydraulic system.

Thus, it is expected that this project contributes to the preservation of natural resources, as well as help the economy to the final consumer and make the population aware of the need to preserve this resource.

Keywords: water, building automation, economy, reuse, sustainability.

Lista de ilustrações

Lista de tabelas

Lista de siglas

GPIO *General Purpose Input/Output*

ONU Organização das Nações Unidas

IDE *Integrated Development Environment*

IoT *Internet of Things*

IBM *International Business Machines*

M2M *Machine to Machine*

MQTT *Message Queuing Telemetry Transport*

PCI Placa de Circuito Impresso

RTC *Real Time Clock*

SoC *System-on-a-Chip*

Sumário

1 Introdução

A água é um recurso natural renovável que está presente em toda a superfície terrestre. Apesar de que 3/4 da superfície da Terra seja ocupada pela água, 3% desse total são de água doce, dos quais apenas 20% encontram-se imediatamente disponíveis para o homem, somente 0,266% da água doce representa toda a água dos lagos, rios e reservatórios (significa 0,007% do total de água doce e salgada existente no planeta) o restante é dividido calotas polares e montanhas (68,9%) e água subterrânea (29,9%) (SILVEIRA, 2008).

Além disso, a água é distribuída de forma desigual, há lugares que sofrem com a falta d'água permanente, já há outros em que isso não é um problema. Ademais, a distribuição desigual da água pelas diferentes regiões do planeta faz com que haja escassez do recurso em vários países. Lugares onde a água é um bem escasso, são geralmente lugares onde a população e governo investem em formas de economizar e reaproveitar esse bem essencial para vida. Em lugares onde há abundância, população e governo geralmente não fazem esforço para uma política efetiva de economia, ao contrário, gastam em demasia. A falta de consciência no uso, o crescimento da população, o desperdício e a falta de políticas de reaproveitamento são fatores que contribuem para a escassez da água.

Apesar de ser um recurso renovável, a demanda por água cresce muito mais rápido que sua capacidade de renovação (ciclo biológico da água). Exigindo assim mais expertise e capacidade de gestão dos setores governamentais, industriais e domésticos. A utilização e oferta da água não são um problema regional, estão presente em todo o globo.

A crise hídrica do Brasil nos últimos anos pegou de surpresa toda população, e trouxe à tona uma discussão importante na sociedade: a necessidade de buscar formas alternativas de economizar água. Com o avanço da crise econômica, os consumidores brasileiros viram seu poder de compra e suas economias pessoais se derretendo, e mesmo com a amenização da crise hídrica, a necessidade de economizar é cada vez mais gritante no nosso meio.

Segundo dados apresentados pela Organização das Nações Unidas (ONU), 110 litros de água por dia são suficientes para atender as necessidades de uma pessoa.(ISA, 2008) A vazão de um chuveiro, por exemplo, pode variar de 6 a 25 litros por minuto, dependendo do seu modelo, ou seja, em um banho de 10 minutos, o gasto de água pode variar de 60 a 250 litros (ROTAGINE, 2017), superando assim o número recomendados pela ONU, sem contar ainda com outros gastos da rotina de uma pessoa como: escovar os dentes, lavar as mãos, lavar as louças, lavar roupas.

Com a evidência da escassez dos recursos hídricos no mundo, é notória a importância da correta aplicação deste recurso ambiental para assim diminuir sua demanda e preservar este recurso vital.

Com a popularização de tecnologias no mercado mundial, bem como o amadurecimento da ideia que essas tecnologias podem ser aproveitadas não só para o conforto pessoal, mas também como forma de auxílio na gestão eficiente dos recursos hídricos e energéticos de uma residência, a Automação Predial se tornou uma grande aliada do ser humano e passa a ser não só mais uma tendência e sim uma realidade cada vez mais presente pelo mundo e que não pode ser ignorada no mercado brasileiro.

Não só o consumidor final, mas também o setor industrial e agrícola, que são os maiores consumidores de água devem ficar atentos ao advento de novas tecnologias que buscam melhorar a gestão energética e hídrica de suas plantas industriais e plantações agrícolas.

1.1 Objetivos

1.1.1 Objetivo Geral

O projeto consiste em produzir um protótipo de *hardware* e *software* economicamente viável, que seja capaz de realizar o acompanhamento do consumo de água e medição do nível do reservatório, podendo ser aplicado para o monitoramento em redes de água potável e água de reúso (em redes de água não potável). O sistema proposto deverá gerenciar eventuais problemas na distribuição da rede, para então disponibilizar os dados do monitoramento em tempo real, buscando um consumo consciente do recurso hídrico. Para isso, será utilizado o microcontrolador ESP8266 para a captura dos dados dos sensores transmitindo a leitura para uma página Web em tempo real.

1.1.2 Objetivos Especícificos

(a) realizar a revisão da bibliografia sobre sistemas gestão de consumo de água, monitoramento em tempo real e aplicabilidade em sistemas de reaproveitamento de água de chuva e de reúso (em particular águas cinzas);

(b) conhecer as características e funcionalidades do *hardware* do ESP8266;

(c) realizar a revisão da bibliografia sobre sistemas de medição de consumo de água;

(d) projetar e construir um protótipo do sistema com um supervisório capaz de realizar o acompanhamento do consumo de água, medição do nível do reservatório e gerenciar eventuais problemas no abastecimento da rede. Isso tudo será apresentado através de uma interface com um usuário;

1.2 Estrutura do trabalho

O presente trabalho está organizado em 5 capítulos. O Capítulo 1 apresenta algumas vantagens do emprego de tecnologias em residências. Este capítulo também descreve o objetivo do trabalho, que é propor um sistema de automação residencial sem fio capaz de realizar o monitoramento e gestão dos recursos hídricos de uma residência.

O Capítulo 2 apresenta uma revisão sobre os *hardwares* e *softwares* utilizados nas etapas de desenvolvimento do trabalho, bem como uma revisão teórica dos temas relacionados.

A metodologia detalhada utilizada nesse estudo encontra-se no capítulo Capítulo 3.

Os testes realizados estão no Capítulo 4. O Capítulo 5 traz as considerações finais do projeto.

1.3 Justificativa

Com a evidência da escassez dos recursos hídricos no mundo, é notório a importância da correta aplicação deste recurso ambiental para assim diminuir sua demanda e preservar este recurso vital.

Diante deste cenário, pode-se notar que é necessário a utilização da tecnologia em prol do meio ambiente. A automação predial se tornou um grande aliado do ser humano, tanto para facilitar o dia-a-dia quanto também para auxiliar na gestão e economia de recursos ambientais.

Assim, este projeto terá como foco a utilização de um sistema de automação predial que seja capaz de realizar o gerenciamento da água consumida em edificações em tempo real.

1.4 Metodologia Utilizada

O projeto foi dividido em duas etapas, sendo a primeira delas a medição do consumo de água de alguns pontos de um residência com os dados sendo salvos em uma planilha online do Google. Na segunda etapa foi desenvolvido uma interface gráfica para o acompanhamento em tempo real do consumo total da casa usando a plataforma ThingSpeak.

2 Revisão Bibliográfica

Este capítulo tem por finalidade apresentar a revisão dos equipamentos e softwares envolvidos no sistema, bem como expor os conceitos teóricos necessários para a compreensão do sistema proposto. Assim, analisa-se:

- Fontes Alternativas de Água
- Perdas na Distribuição de Água
- Internet das Coisas
- ESP8266
- Sistemas Embarcados
- *Message Queuing Telemetry Transport* - MQTT
- Sensores de Vazão Aplicados a Medição de Água
- Sensores de Nível Aplicados a Medição de Água

2.1 Fontes Alternativas de Água

Podemos definir fontes alternativas de água como sendo fontes opcionais àquelas comumente disponível nas residências. As mais conhecidas fontes alternativas de água são as águas de chuva, água dessalinizada, água de reúso industrial e água cinza.

O mais comum e barato método de utilização alternativa de água é através da água proveniente das chuvas. Ela é totalmente viável em regiões onde o nível pluviométrico é alto e bem distribuído durante o ano.

Segundo Bazzarella (2005), a utilização de água pluvial é uma prática antiga que foi deixada de lado a medida que os sistemas públicos de abastecimento foram sendo implementados em grande escala. As cidades mais modernas retomaram essa prática antiga sendo características das cidades de países desenvolvidos.

2.1.1 Aproveitamento de Água de Chuva

Os sistemas de aproveitamento de água da chuva podem ser usados em instalações prediais, sejam residenciais ou comerciais. Eles são regulamentados pela ABNT 15527

NBR (2007) e recolhem, filtram, armazenam e disponibilizam a água da chuva para uso externo ou interno.

Esses métodos sustentáveis ajudam a economizar água e se transformam em alternativa em épocas de racionamento hídrico. Nas áreas urbanas são usados, geralmente, para abastecer descargas de vasos sanitários, regar hortas e jardins, para limpeza de áreas comuns e também de automóveis.

As áreas de captação são geralmente os telhados das casas ou indústrias, lajes e pisos.

Para captação e transporte da água de chuva até o reservatório são necessárias calhas e coletores que podem ser de material tipo PVC ou metálicos, comumente achados no mercado.

Quando chove a primeira água lava os telhados trazendo consigo muita sujeira e devem ser removidas. Isso pode ser feito manualmente com uso de tubulações que podem ser desviadas do reservatório ou automaticamente através de dispositivos de autolimpeza.

A (Figura 1), exemplifica um separador de fluxo (*by-pass*). É um sistema que veda e desvia o fluxo da primeira água proveniente dos telhados para fora do reservatório. Para esvaziar a câmara, o separador conta com uma válvula de descarte lento, que após a chuva pode ser esvaziada manualmente ou por um sistema automático.

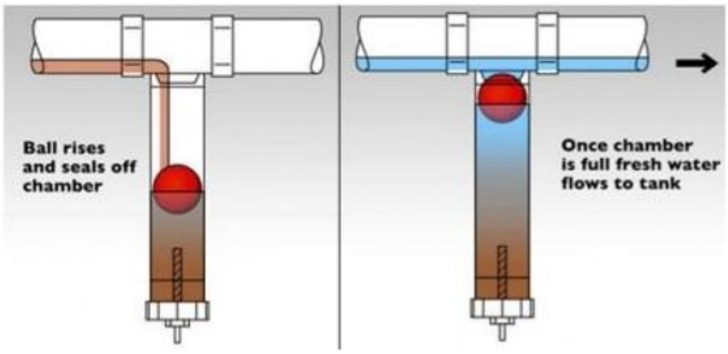


Figura 1: Sistema By-Pass

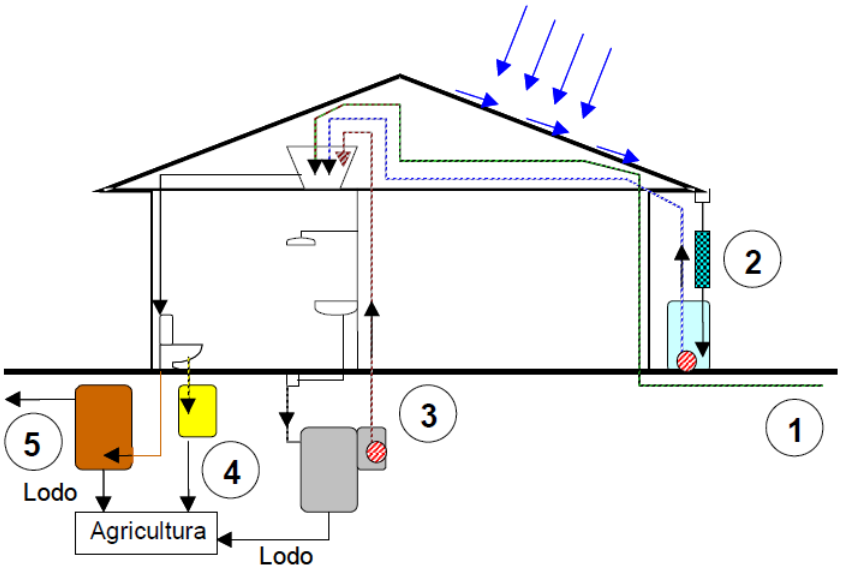
Fonte: (OLIVEIRA, 2017)

A utilização de sensores de nível para monitorar o reservatório de água pluvial, bem como sensores de vazão para saber a quantidade de água que entrou no sistema e a quantidade de água consumida, são soluções imprescindíveis para uma boa gestão do uso dessa fonte alternativa de captação de água.

2.1.2 Reuso de Águas Cinzas

Outro sistema de reaproveitamento de água é a reutilização das águas cinzas, mas para isso é necessário conceituar esses termos.

O princípio de separação dos efluentes domésticos com o objetivo de reaproveita-los ou reduzir seu lançamento no meio ambiente é conceituado como Saneamento Ecológico.



① Suprimento de água convencional, a partir da rede pública.
② Coleta e aproveitamento de água de chuva a partir do telhado da edificação;
③ Coleta, tratamento e reúso das águas cinza na descarga de vasos sanitários;
④ Coleta, tratamento e reúso de águas amarelas (urina) na agricultura;
⑤ Coleta, tratamento e reúso das águas negra na agricultura;

Figura 2: Esquema de Gerenciamento de Águas de uma Edificação

Fonte: (BAZZARELLA, 2005)

Todo o esgoto gerado em uma edificação pode ser separado segundo Bazzarella (2005), da seguinte forma (Figura 2):

- Águas negras (blackwater): são definidas como aquelas que contêm excretas humanas, oriundas das bacias sanitárias e águas cinzas como aquelas resultantes do asseio corporal, da lavagem de pisos e de roupas (TOMAZ, 2007);
- Água Cinza (greywater): águas servidas, excluindo o efluente dos vasos sanitários (BAZZARELLA, 2005);
- Água amarela: representando somente a urina;
- Água marrom: representando somente as fezes.

Assim como na captação de águas da chuva, um sistema baseado em IoT capaz de monitorar e quantificar a produção de água proveniente desse sistema de reúso é primordial para uma boa gestão do recurso. Uma residência com tal sistema instalado tem autonomia de alocar a quantidade de água de reúso nas aplicações que a mesma pode ser usada de forma eficiente e proporcionalmente a quantidade disponível desse recurso.

2.2 Perdas no Sistema de Distribuição de Água

Uma boa parte da água que sai das companhias de saneamento e distribuição não chega no consumidor final. Essas perdas além de gerarem prejuízo para as companhias, carregam também o peso de desperdiçar um bem tão essencial e que segue uma tendência de escassez. No Brasil em média temos 40,8% de perdas no sistema de distribuição (MCIDADES, 2018).

Independentemente do quão complexo ou simples é o sistema de distribuição de água, as perdas inerentes ao mesmo representam uma perda direta de receita financeira, bem como o desperdício de dois recursos ambientais cada vez mais escassos: água e energia. Um sistema capaz de monitorar em tempo real o consumo e as perdas de água aliado a um planejamento bem feito,contribui para a redução das perdas de água e consequentemente reduz ou estabiliza a retirada de água dos reservatórios e, portanto, garante o abastecimento.

As perdas de água podem ser classificadas por tipos:

- **Perdas Reais:** volume de água correspondente às perdas físicas até ao medidor do cliente, quando o sistema está pressurizado.(ALEGRE et al., 2005)
- **Perdas Aparentes:** esta parcela das perdas contabiliza todos os tipos de imprecisões associadas às medições da água produzida e da água consumida, e ainda o consumo não autorizado (por furto ou uso ilícito). (ALEGRE et al., 2005)

No caso do Brasil as perdas reais podem chegar a 72% do total, enquanto as perdas aparentes representam os outros 28%, sendo 19,6% de imprecisões de medição de clientes e 8,4% de consumo não autorizado.(LIMA, 2018)

Com o advento dos sistemas baseados em IoT, uma alternativa para as companhias de distribuição de água no Brasil é investir pesado nesse tema. A medição em tempo real do fluxo de água que sai do sistema de distribuição e medição setorizada desse fluxo, aliado a dados como pressão, ruídos do sistema e tendo como contrapartida os dados de água consumidos pelos clientes podem tornar muito mais eficiente a detecção de vazamentos na linha e possíveis roubos de água.

2.3 Sistemas Embarcados

Grande parte dos microprocessadores fabricados são para atividades dedicadas e não para uso geral como um computador convencional. Segundo Chase (2009), o que diferencia um computador convencional dos sistemas dedicados como DVD, micro-ondas, GPS, é o projeto baseado em um conjunto dedicado e especialista constituído por *Hardware*, *Software* e periféricos – Sistema Embarcado.

Ball (2002), trata o sistema embarcado como sendo dedicado a uma única tarefa e com interação ao meio através de atuadores e sensores. Por estar sempre em contato com o meio, o sistema embarcado exige do projetista maturidade, experiência e dedicação em diversas área como, programação, sistemas em tempo real, aquisição de dados, eletrônica digital e de potência, sensores e atuadores, e principalmente otimização para que o código seja eficaz e seguro, além de estar atento a restrição ao consumo de energia e a pouca disponibilidade de memória.

Azevedo et al. (2002), levantou o ponto de que o aumento do nível de integração dos componentes eletrônicos levou o desenvolvimento de chips ou circuitos integrados que apresentam a funcionalidade de um sistema complexo contendo memoria, periféricos e processador em um core. Esse modelo de desenvolvimento é chamado de *System-on-a-Chip* (SoC).

Com a popularização dos sistemas embarcados ou dedicados e sua gama cada vez maior de funções complexas, consequentemente tem aumentado o tamanho dos programas e ocupando cada vez mais memória, recurso este que na maioria das vezes é escasso. Com isso as técnicas de otimização, bem como compressão de códigos e a dedicação do projetista em deixar o sistema mais eficaz são determinantes para um bom desempenho do sistema projetado.

Existem vários modos de funcionamento de um sistema embarcado e o sistema estudado e desenvolvido nesse trabalho usam basicamente dois modos:

- Reativo: um sistema reativo tem como características respostas a eventos externos, sejam eles periódicos ou assíncronos. Há a necessidade de entrada de alguma dado, que pode ser de um sensor, ou de um botão acionado pelo usuário para que aconteçam ações.

- Controle em tempo real: de acordo com Farines, Fraga e Oliveira (2000) um sistema de controle em tempo real deve reagir a estímulos oriundos do seu ambiente em prazos específicos. Ou seja, há limites de tempo para a realização de tarefas, como por exemplo, leitura de sensores, atuação em determinada situação por parte de atuadores, atualização de informações em uma página WEB, etc. Os sistemas de tempos real podem ser classificados em:

1. *Soft Real Time*: Ainda segundo Farines, Fraga e Oliveira (2000), os sistemas podem ser classificados do ponto de vista de segurança. São chamados Sistemas Não críticos de tempo Real (*Soft Real Time*) quando as consequências de uma falha devida ao tempo é da mesma ordem de grandeza que os benefícios do sistema em operação normal.
2. *Hard Real Time*: Já os Sistemas Críticos de Tempo Real (*Hard Real Time*) são entendidos quando as consequências de pelo menos uma falha temporal excedam em muito os benefícios normais do sistema. Como exemplo pode-se pensar nos sistemas de controle de um avião, onde uma falha pode resultar em queda e perdas de vidas.

2.4 ESP8266

O ESP8266 (Figura 3) é um módulo WiFi de baixo custo e possui tecnologia de baixo consumo energético. Ele é integrado com o microcontrolador TenSilica L 106 de 32 bits. A arquitetura para que o módulo seja capaz de poupar energia funciona em 3 modos: modo ativo, modo de suspensão, modo de sono profundo. O ESP8266 consome cerca de 60uA no modo de sono profundo com o relógio *Real Time Clock* (RTC) ainda em execução. A integração de sensores, atuadores e dispositivos específicos da aplicação com a placa ESP8266 é simples através dos pinos *General Purpose Input/Output* (GPIO), que faz uma interface da placa ESP8266 com mundo externo.

Figura 3: Microcontrolador ESP8266 NodeMCU

Fonte: FilipeFlop, 2018

Segundo Oliveira (2017), inicialmente os microcontroladores se resumiam em interfaces de I/O, e ao longo do tempo foram agregando memoria ROM, RAM, circuitos de *clock*, interfaces de comunicação e mais recentemente interfaces de rede.

Um microcontrolador que agregue todas essas funcionalidades é capaz de realizar diversas aplicações, principalmente as baseadas em *Internet of Things* (IoT), que vem se popularizando nos últimos anos. O ESP 8266, baseado em microcontroladores atuais possui como já descrito baixíssimo consumo de energia. Sendo capaz de ser mantido em funcionamento por um bateria por um bom tempo, sem a necessidade de recarga ou substituição.

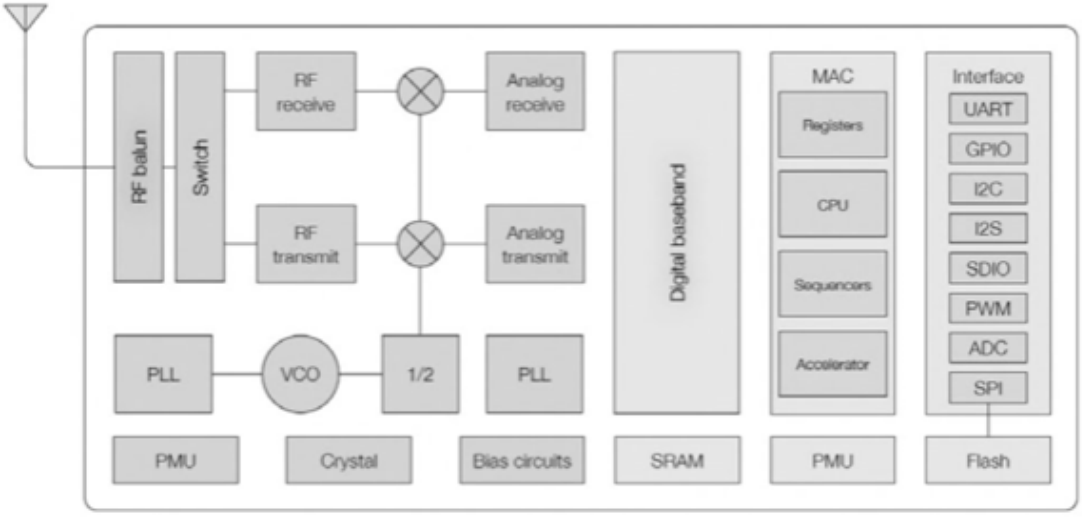


Figura 4: Arquitetura Interna do Microcontrolador ESP8266

Fonte: (OLIVEIRA, 2017)

A Figura 4 exemplifica o diagrama funcional do ESP8266. Há interfaces de entradas e saídas para diversos protocolos, conjuntos de componentes de processamento como CPU e Memória.

As diversas funcionalidades apresentadas pelo ESP8266, aliado ao seu baixo custo, o credencia para as mais variadas aplicações. No trabalho em questão o ESP8266 fará a interface com sensores, transmissores, atuadores com uma aplicação WEB de monitoramento do sistema hidráulico de uma residência.

2.5 Internet das coisas

Segundo (ASHTON, 2009), um dos precursores da Internet das Coisas - IoT, esta pode ser definida como a rede mundial de objetos interconectados com exclusividade

endereçável com base em protocolos de comunicação padrão. Já segundo Sundmaeker et al. (2010), essas "Coisas"são participantes ativos em negócios, informações e processos onde são capazes de interagir e comunicar entre si e com o ambiente, trocando dados e informações sobre o meio, enquanto reagem autonomamente aos eventos do mundo real / físico, executando processos que desencadeiam ações e/ou criam serviços com ou sem intervenção humana direta.

Atualmente, com a popularização dos sistemas embarcados e com o acesso amplo a internet, a IoT já não é mais algo futurístico e sim algo real e presente. Hoje existe grande demanda e infinitas soluções para as mais variados áreas utilizando Internet das Coisas, desde sistemas complexos de uma planta industrial até um acionamento de uma lâmpada via internet numa residência.

> "A IoT acontecerá em um ambiente que influenciará e será influenciado nas suas forças, seus comportamentos e seus papéis, incluindo governo, iniciativa privada, sociedade e economia como um todo. O governo faz parte do ambiente, assumindo papéis como o de regulador, incentivador e demandante. A iniciativa privada, para atender às suas características e aos seus interesses, atuará no ambiente de IoT. A sociedade deve ser atendida, estar preparada, ser respeitada e participar no ambiente de IoT. A economia, com suas forças, pode garantir e restringir as iniciativas nesse ambiente de IoT."(ALBERTIN; ALBERTIN, 2017, p. 6)

2.6 Message Queuing Telemetry Transport - MQTT

O *Message Queuing Telemetry Transport* (MQTT) , é um protocolo aberto de mensagens voltado para comunicação *Machine to Machine* (M2M) desenvolvido pela *International Business Machines* (IBM) em 1999. Ele utiliza basicamente o sistema *publish/subscribe* e adota o protocolo TCP para a troca de mensagens. Foi desenvolvido para hardwares modestos e com taxas de conexões pequenas, mas com confiabilidade e garantia de entrega das mensagens.

Diferentemente de um modelo cliente/servidor tradicional, onde o servidor envia diretamente ao cliente as mensagens requisitadas, o protocolo MQTT tem um *message broker* e inúmeros clientes. Segundo Yuan (2017), o *broker* funciona como um servidor que recebe as mensagens dos clientes e as roteia para os clientes de destino relevantes. (Figura 5) Esse cliente pode ser qualquer coisa que interaja com o *broker*, como sensores, microcontroladores, supervisórios, etc.

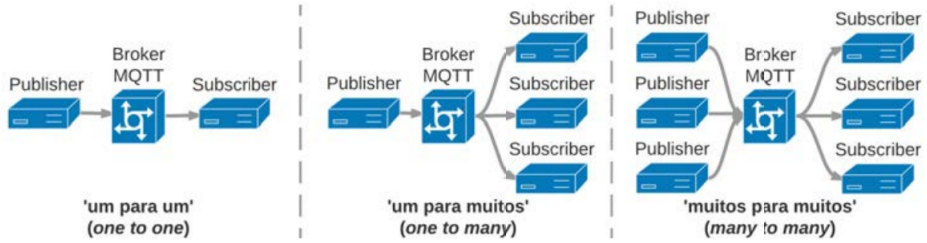


Figura 5: Protocolo MQTT

Fonte: (TORRES; ROCHA; SOUZA, 2016)

2.7 Sensores de Vazão Aplicados a Medição de Água

Medir o fluxo de água remota aos egípcios e romanos por volta de 2000 a.C., onde a construção de aquedutos ficaram famosas, segundo relatos históricos. Já no ano de 90 d.C. há relatos precisos de medidores de vazão de água por parte do governador e engenheiro Julius Frontinus (30 -103 d.C.). (MECATRÔNICA, 2005)

Medidores de vazão podem ser definidos como instrumentos capazes de determinar a quantidade de líquidos, gases e sólidos que irão passar por um determinado local em um determinado espaço de tempo. (HORIGOSHI, 2016)

Os variados tipos de medidores de vazão são adotados de acordo com a situação na qual eles são aplicados. Eles podem ser classificados de acordo com seu princípio de medição:

- **Geradores de Pressão diferencial:** Placa de orifício, Bocal, Venturi, Pitot, Pitot de média, Centrífugos, Laminares e Jato;
- **Medidores lineares:** Área variável, Coriolis, Eletromagnético, Térmico, Turbina, Ultrassônico, Vórtice e Efeito Hall;
- **Volumétricos:** Diafragma, Disco de nutação, Palheta, Pistão oscilante, Pistões recíprocos, Lóbulo, Engrenagem e Semi-imerso;
- **Em canais abertos:** Calhas e Vertedores;
- **Medidores especiais:** Força, Correlação e Laser.

Os principais modelos de medidores de vazão utilizados para a medição de água são o Ultrassônico e o Pulsado por Efeito Hall.

2.7.1 Medidor de Vazão Ultrassônico

Os medidores de vazão ultrassônicos são dispositivos não invasivos que utilizam vibração acústica para aferir a vazão de um liquido. Medidores Ultrassônicos por efeito Doppler e por tempo de transito são os dois tipos que existem. Independentemente do tipo de sensor ultrassônico utilizado, o mesmo pode ser preso externamente na tubulação evitando assim perdas de cargas, interrupção da linha de vazão, possíveis vazamentos em emendas, corrosão e deterioração do medidor.

A frequência de uma onda sonora sofre alterações quando existe movimento relativo entre a fonte transmissora e a receptora. Por exemplo, quando o fluido está se movendo em relação ao duto e o detector (sensor ultrassônico) está parado em relação ao duto, o movimento altera o comprimento de onda da onda sonora e, portanto, a frequência detectada (HALLIDAY; RESNICK; WALKER, 2009). Com isso podemos medir a vazão desse fluido (Figura 6).

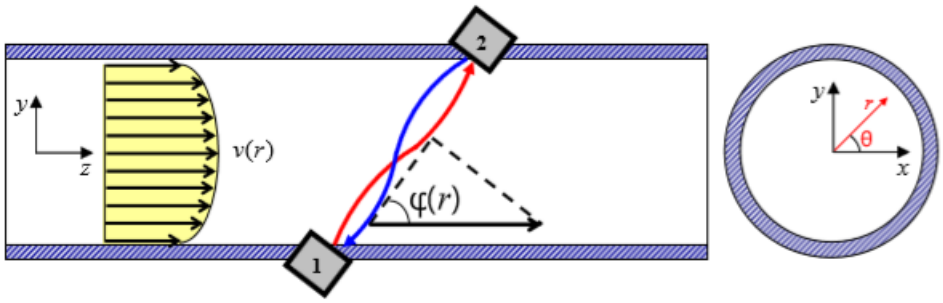


Figura 6: Efeito de arrasto do sinal ultrassônicos pelo escoamento

Fonte: (MATHIAS, 2010)

2.7.2 Medidor de Vazão por Efeito Hall

Descoberto por E. H. Hall em 1879, o efeito Hall é definido por Coraucci et al. (2008) como um efeito físico que ocorre em materiais que conduzem corrente na presença de um campo magnético. Já Boylestad (2012), define um sensor tendo como princípio de funcionamento o Efeito Hall como sendo um semicondutor que, quando uma corrente elétrica I o percorre e estando suscetível a um campo magnético B, surgirá ortogonalmente à corrente e ao campo magnético uma tensão de saída VH, conforme ilustrado na (Figura 7).

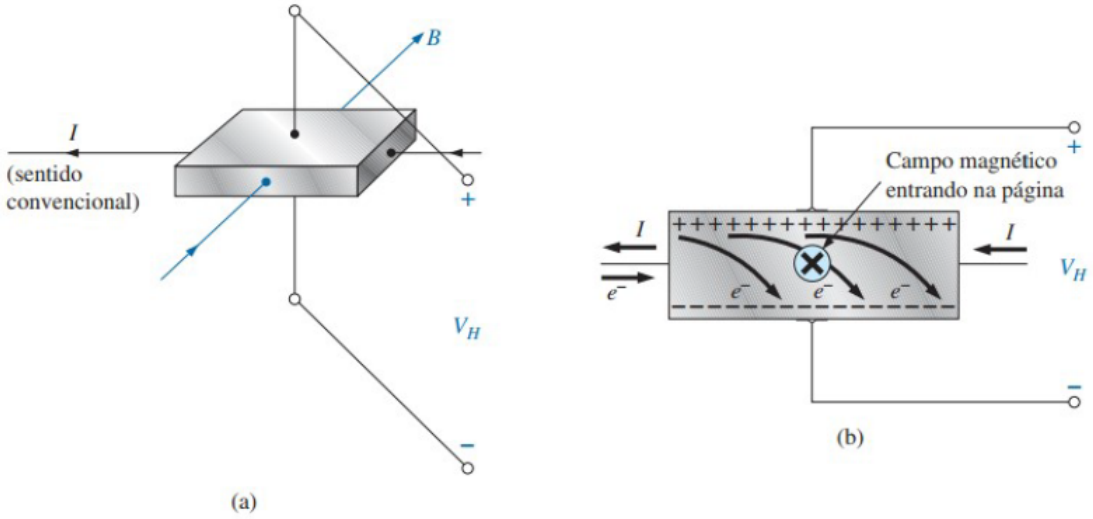


Figura 7: Funcionamento de um Sensor de Efeito Hall

Fonte: (BOYLESTAD, 2012)

Quando o fluxo da água passa pelo interior de um sensor pulsado por efeito Hall ele movimenta a turbina, e pelo fato de conter um ímã permanente, o sensor sofre a ação de um campo magnético com intervalos proporcionais à vazão da água. Consequentemente, quanto maior a vazão de água, maior será a frequência de pulsos de tensão gerados na saída do sensor por Efeito Hall.

2.8 Sensores de Nível Aplicados a Medição de Água

Neto e Santos (2016) nos dizem que um sensor de nível pode aferir duas variáveis: a posição da superfície do líquido tendo um ponto de referência e a altura hidrostática criada pelo líquido cuja superfície não se conhece. Existem os mais variados tipos de sensores capazes de aferir essas variáveis, sendo os mais comuns do tipo boia (lateral, pêra e magnética), sensores ultrassônicos e hidrostáticos (Figura 8).

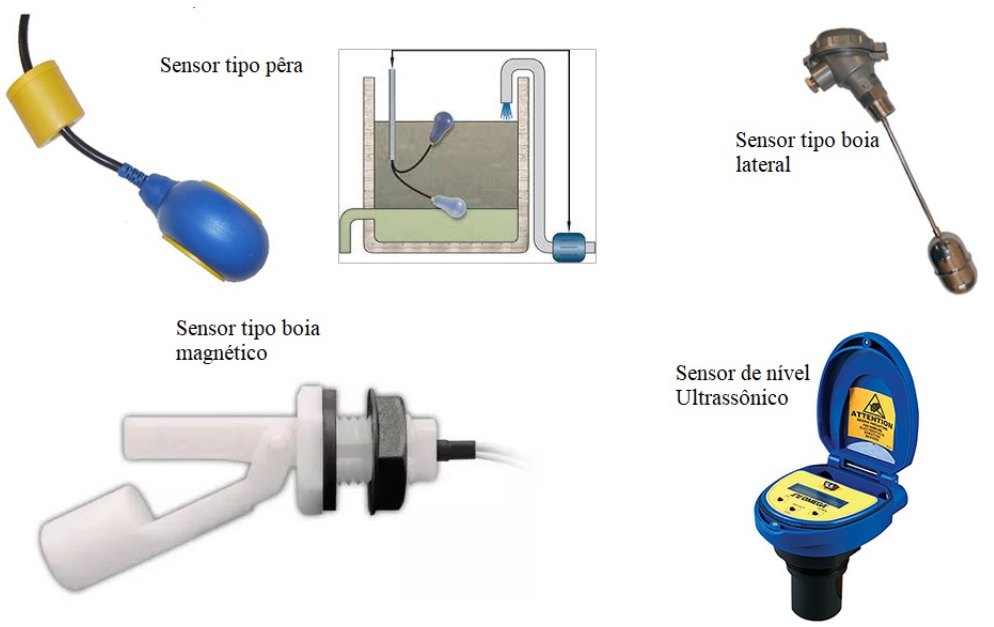


Figura 8: Tipos de Sensores de nível

Fonte: (ICOS, NIVITEC, OMEGA, 2019)

Os sensores dos tipos boia lateral, pêra e magnética tem seus princípios de funcionamento baseados na lei do empuxo, onde seus flutuadores acionam mecanismos de indicação de nível através de um contato elétrico, mecânico ou magnético.

Já os sensores de nível ultrassônicos podem se basear na emissão e reflexão de ondas ultrassônicas ou as ondas ultrassônicas que se atenuam à medida em que o nível de água aumenta.

Todos esses tipos de sensores de nível são largamente utilizado em projetos de IoT para medir níveis de reservatórios prediais e para o projeto em questão utilizamos um sensor de nível tipo boia magnética.

3 Metodologia

3.1 Visão geral do projeto

O projeto em questão foi dividido em duas etapas. A primeira delas foi um teste de viabilidade do sistema de mediação de consumo de água em alguns pontos da residencia. Os pontos escolhidos foram aqueles que são os maiores produtores de água cinza e serviram como justificativa para a importância de um sistema de reaproveitamento de água. A segunda etapa foi o desenvolvimento de uma plataforma intuitiva para monitoramento e gestão em tempo real de um sistema hidráulico de uma casa.

3.2 Primeira Etapa

O objetivo nessa etapa foi mensurar a quantidade de água consumida em alguns pontos da residência e guardar esses dados de forma automática em tempo real em uma planilha online. Os pontos escolhidos foram as principais fontes produtoras de água cinza tendo assim o objetivo de alertar sobre a importância de se reaproveitar água para fins não potáveis. Para isso foram instalados sensores de vazão de Efeito Hall YF-S201B (Figura 9).

Por questões de logística e disponibilidade integral para realizações de testes e intervenções físicas no sistema hidráulico, escolhi minha residência como estudo de caso. A casa é localizada no município de Nova Lima/MG, distante 18 Km da capital do Estado, com população estimada em 93.577 segundo dados da Prefeitura Municipal de Nova Lima.

3.2.1 Sensor de Vazão

No projeto foi utilizado o sensor de vazão de água do modelo YF-S201B (Figura 9), que tem como princípio de funcionamento o Efeito Hall, gerando pulsos que são convertidos em vazão (l/min). O modelo em questão foi escolhido pelo fato de ser facilmente encontrado no mercado e ter um excelente custo benefício, além de sua precisão e vasta utilização em outros projetos.

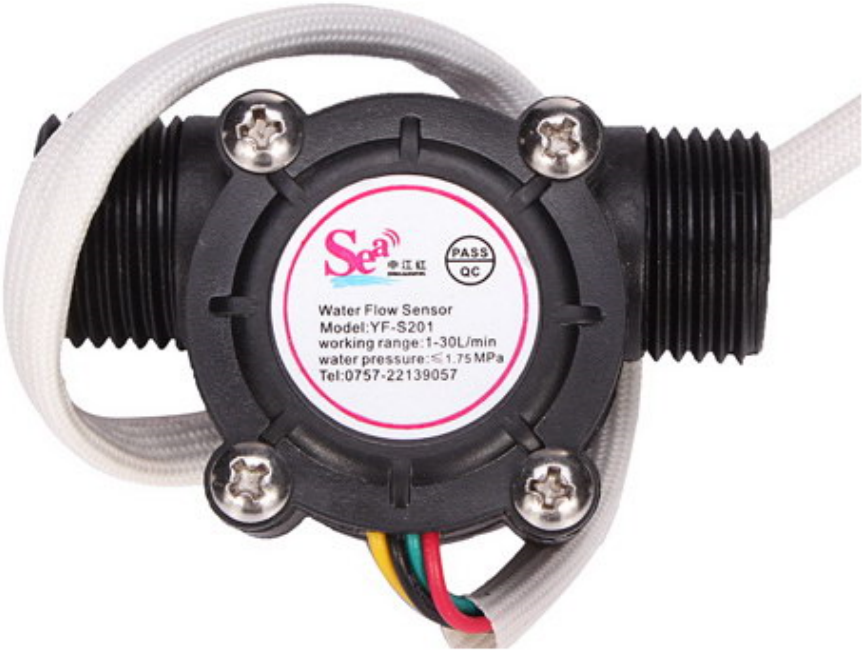


Figura 9: Sensor de Vazão YF-S201B

Fonte: FilipeFlop, 2018

Segundo informações fornecidas pelo vendedor (Flipeflop) e datasheet, o sensor de vazão em questão apresenta as seguintes características:

– Modelo: YF-S201b

– Tipo de sensor: Efeito Hall

– Tensão de operação: 5-24V

– Corrente máxima: 15mA (5V)

– Faixa de fluxo: 1-30L/min

– Pressão máxima: 2,0 MPa

– Pulsos por litro: 450

– Frequência (Hz) = 7,5*Fluxo(L/min)

– Temperatura de trabalho: -25 a 80°C

– Exatidão: 10%

– Comprimento do cabo: 15cm

– Dimensão conexão: 1/2"

– Dimensão diâmetro interno: 0,78"

– Dimensão externa: 2,5"x 1,4"x 1,4"

O sensor de vazão YF-S201B, é um sensor de Efeito Hall que possui uma turbina e um ímã permanente em seu interior (Figura 10).

Figura 10: Interior do Sensor de Vazão

Fonte: FilipeFlop, 2018

3.2.2 Medindo Consumo de Água

Para medir o consumo de água dessa residência utilizou-se sensores de vazão em pontos escolhidos por serem as principais fontes produtoras de água cinza: pia do banheiro, chuveiro e pia da cozinha. Devido a falta de recursos para a compra de diversos sensores de vazão, durante 1 semana foi utilizado apenas 1 banheiro da casa e os outros 2 foram isolados. Com isso utilizou-se apenas 3 sensores de vazão: 1 na pia do banheiro, 1 na pia

da cozinha e outro no chuveiro. Obteve-se assim a produção semanal de água cinza e fez-se uma projeção proporcional da produção mensal, para assim obter a porcentagem de água que poderia ser reaproveitada em relação ao consumo total de água da residência.

Os sinais de saída do sensor de vazão são incompatíveis com o nível de tensão de operação das GPIOs da placa ESP8266. As GPIOs trabalham na faixa de tensão de 3.3V, enquanto o sinal de saída do sensor é 5V. Em consequência disso, precisou-se adequar a saída do sensor no nível de tensão das GPIOs. O método mais barato e eficaz para contornar o problema é utilizando um divisor de tensão resistivo.

3.2.2.1 Adequação do Nível de Tensão de Operação do Sensor de Vazão

Um divisor de tensão é um circuito muito eficaz e simples. Uma tensão V é gerada a partir de uma tensão V_0 de uma fonte, sendo V alguma fração de V_0 .

A Figura 11 mostra um divisor de tensão junto com um voltímetro.

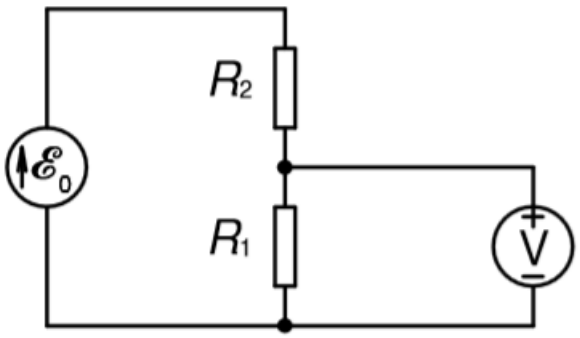


Figura 11: Divisor de Tensão com voltímetro

Fonte: (LESCHE, 2013)

Analisando o circuito da figura 11, sabe-se que a corrente não flui pelo voltímetro, cuja a resistência pode ser aproximada como infinita. Com isso precisa-se definir a corrente do circuito. Como é um circuito em série, a corrente que flui pela fonte e pelos resistores R_1 e R_2 é a mesma I. Escolhendo o sentido de fluidez da corrente como o sentido horário e pela Lei de Malhas obtém-se:

$$-\varepsilon_0 + R_2 I + R_1 I = 0 \tag{3.1}$$

Aplicando a Lei das Malhas na malha formada pelo Voltímetro e pelo resistor R_1, obtém-se,

$$-R_1 I + V_{leitura} = 0 \tag{3.2}$$

onde $V_{leitura}$ é a tensão indicada pelo voltímetro. Eliminando I do sistemas formado pelas equações 3.1 e 3.2, obtém-se

$$V_{leitura} = \frac{R_1}{R_1 + R_2}\varepsilon_0 \tag{3.3}$$

Para o projeto em questão, tem-se como tensão de entrada 5V e necessita-se de 3.3V de tensão na saída. Substituindo na equação 3.3, ε_0 por 5V e $V_{leitura}$ por 3.3V tem-se:

$$3.3 = \frac{R_1}{R_1 + R_2}x5 \tag{3.4}$$

$$0.66 = \frac{R_1}{R_1 + R_2} \tag{3.5}$$

$$0.66x(R_1 + R_2) = R_1 \tag{3.6}$$

$$0.66xR_1 + 0.66xR_2 = R_1 \tag{3.7}$$

$$0.66xR_2 = (1 - 0.66)xR_1 \tag{3.8}$$

$$0.66xR_2 = 0.34xR_1 \tag{3.9}$$

$$\mathbf{1.94 \cong \frac{R_1}{R_2}} \tag{3.10}$$

A partir da equação 3.10, pode-se encontrar o valor de um resistor fixando um valor em outro. Para isso escolheu-se o resistor de $10K\Omega$ para R_1 e obteve-se

$$R_2 \cong 5.15K\Omega$$

Um resistor de $5.15K\Omega$ não é de valor disponível comercialmente, o valor mais próximo é o resistor de $4.7K\Omega$.

De acordo com o manual do fabricante da placa ESP8266, a mesma entende como nível lógico alto a faixa de 0.75*VDD até VDD + 0.3. Considerando o VDD = 3.3V, o ajuste feito pelo divisor de tensão é satisfatório. Usando os resistores de $10K\Omega$ e $4.7K\Omega$, tem-se uma tensão de saída de V= 3,4V.

3.2.2.2 Montagem do Circuito

Feito os cálculos do divisor de tensão, passou-se para a montagem do circuito em *protoboard* para testes. Foram necessários os seguintes materiais (Figura 12):

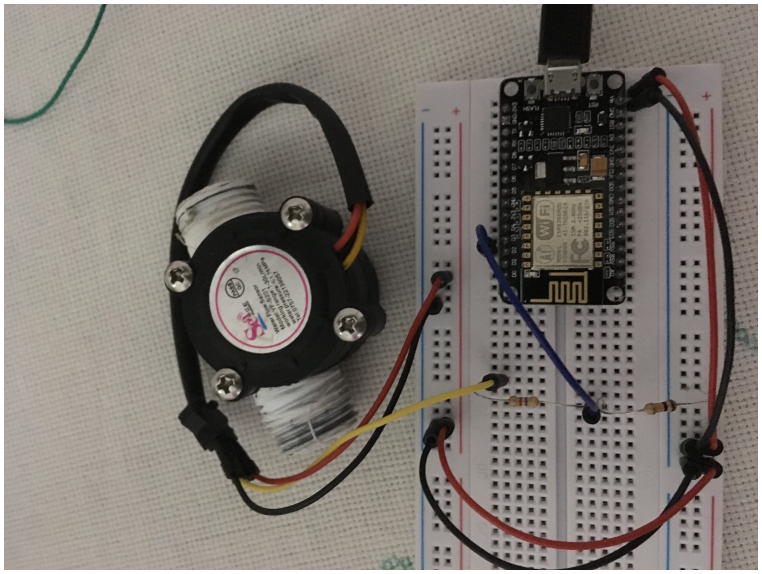


Figura 12: Materiais Utilizados no Circuito e Montagem

- 1 - Protoboard
- 1 - Sensor de Fluxo YF-S201B
- 1 - Resistor de $10k\Omega$
- 1 - Resistor de $4.7K\Omega$
- 1 - Placa NodeMCU V1.0
- Jumpers

Para a alimentação da Placa NodeMCU foi utilizado fonte externa 127Vac/5Vcc ligada através da porta micro-USB do módulo. A tensão de alimentação do módulo fica disponível no pino 15 do mesmo. O sensor de fluxo é alimentado em 5V por essa porta e referenciado com o GND do NodeMCU. A saída do sensor que está em 5V, passa pelo divisor de tensão e entrega 3.4V no pino D2 da placa. O pino D2 é configurado como uma entrada digital e permite o uso de interrupções. O circuito pode ser visto na Figura 13.

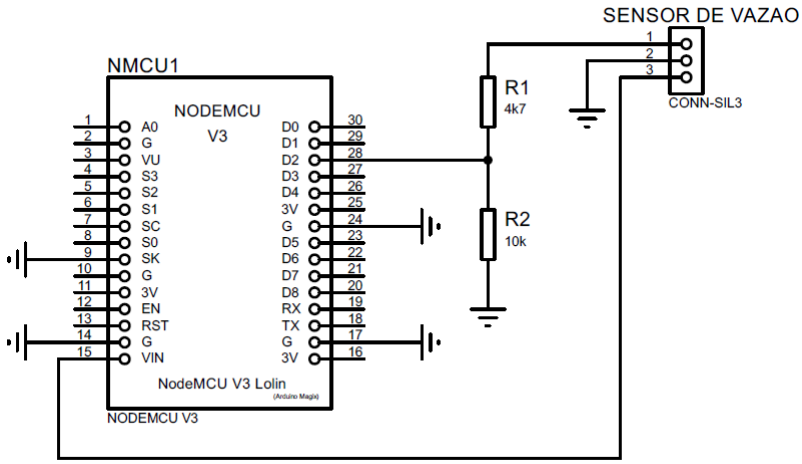


Figura 13: Circuito Sensor de Vazão

3.2.2.3 Ambiente de Programação

A *Integrated Development Environment* (IDE) utilizada como ambiente de programação do módulo NodeMCU, foi a IDE do Arduino na versão V1.6.3 (Figura 14).

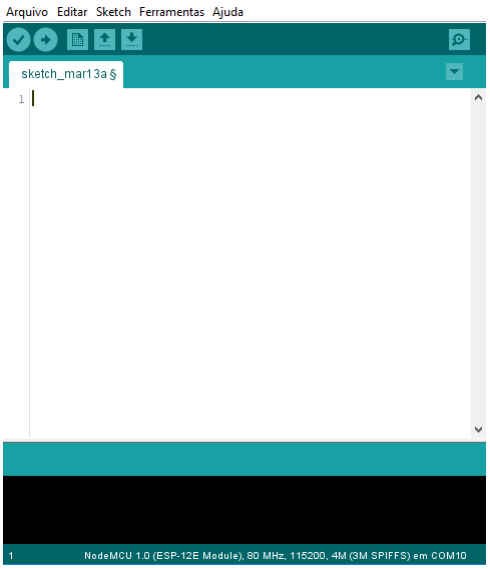


Figura 14: IDE do Arduino

Uma IDE é um aplicativo de *software* que fornece recursos abrangentes para programadores de computador para desenvolvimento de *software*. Normalmente consiste em um editor de código-fonte,compilador e um depurador.A IDE do Arduino pode rodar em diversos sistemas operacionais (Windows, macOS, Linux) e é escrito na linguagem de programação Java. É usada para escrever e carregar programas para a placa Arduino e outras diversas placas como o NodeMCU ESP8266. O Arduino IDE suporta as linguagens C e C ++ usando regras especiais de estruturação de código.

Para o desenvolvimento da aplicação usou-se vários artifícios de programação que serão exemplificados em partes a seguir. O código completo encontrasse em anexo no Apêndice A.

Primeiramente cria-se um Cliente seguro para ter acesso ao HTTPS. Depois monta-se uma string com o método GET para solicitar dados do servidor e os parâmetros seguintes são enviados junto ao URL da solicitação, conforme trecho abaixo.

```
1 WiFiClientSecure client;
2 String textFix = "GET
  ↪ /forms/d/e/1FAIpQLSd7GLUpOQXhLALTQr4-vg/formResponse?ifq&entry.156818112765=";
```

Logo em seguida conectou-se a placa a uma rede WiFi, definiu-se o pino D2 como entrada digital e iniciou-se a memória EEPROM para guardar os dados obtidos caso ocorra alguma queda de energia.

```
1  void setup()
2  {
3    Serial.begin(115200);
4    WiFi.mode(WIFI_STA);
5    delay(10);
6    EEPROM.begin(8);
7    litros=EEPROM.read(addr);
8    pinMode(D2, INPUT);
9    WiFi.begin(ssid, password);
10  }
```

Em seguida trata-se os dados enviados pelo sensor, ou seja, toda vez que a contagem de pulsos for maior que 440 quer dizer que passou 1L pela tubulação. Soma-se a variável litros e envia esse dado com a *string 'tosend'* completando o método GET. Com isso o dado é enviado para o formulário google que por sua vez o armazena em uma planilha online.

```
1  void loop()
2  {
3     if(contagem > 440 )
4    {
5      litros++;
6      Serial.println();
7      Serial.print("Litros: ");
```

```
8       Serial.print(litros);
9       EEPROM.write(addr, litros);
10      EEPROM.commit();
11      contagem = 0;
12    }
13       String toSend = textFix;
14       toSend +=litros;
15       toSend += "&submit=Submit HTTP/1.1"
16      delay(5000);
17  }
```

3.2.2.4 Configurando Planilha Online

Para gravar os dados de vazão obtidos pelos sensores nesta primeira etapa, utilizou-se como banco de dados o Google Planilhas. O Google Planilhas é uma espécie de "Excel Online", que permite ao usuário enviar dados, gravar, acessar e processar remotamente, sem a necessidade de instalar nada no computador ou em um servidor dedicado.

> "Podemos criar essas tabelas em conjunto do Google Forms, para enviar os dados dos nossos sensores, clientes e etc., sem a necessidade de um computador ou servidor pago hospedando o banco de dados, já que a Google se encarrega totalmente do serviço."(MORAIS, 2017)

Para configurar esse banco de dados online, deve-se possuir uma conta Google e criar um Formulário e uma Planilha. Cada sensor equivale a uma pergunta do formulário (Figura 15) e sua leitura é a resposta dessa pergunta. Essa resposta é armazenada em uma linha da planilha (Figura 16), que por sua vez registra data e hora que recebeu o dado.

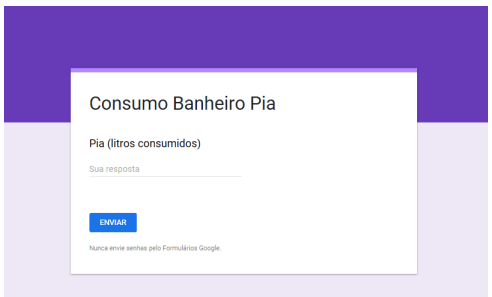
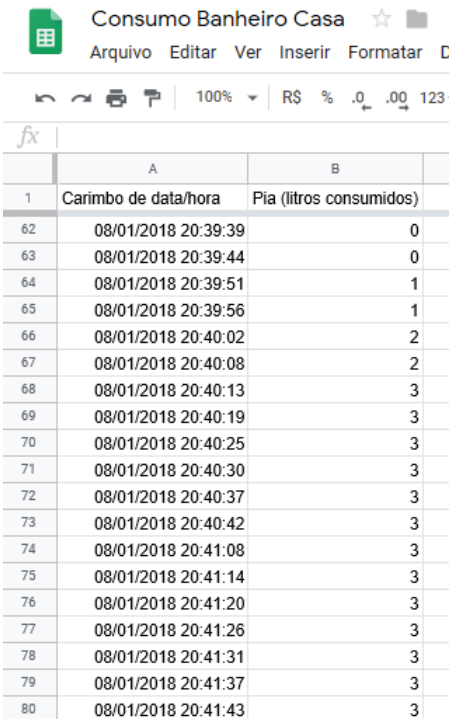


Figura 15: Formulário para Coletar Dados do Sensor



Consumo Banheiro Casa

Arquivo Editar Ver Inserir Formatar D

100% R$ % .0 .00 123

	A	B
1	Carimbo de data/hora	Pia (litros consumidos)
62	08/01/2018 20:39:39	0
63	08/01/2018 20:39:44	0
64	08/01/2018 20:39:51	1
65	08/01/2018 20:39:56	1
66	08/01/2018 20:40:02	2
67	08/01/2018 20:40:08	2
68	08/01/2018 20:40:13	3
69	08/01/2018 20:40:19	3
70	08/01/2018 20:40:25	3
71	08/01/2018 20:40:30	3
72	08/01/2018 20:40:37	3
73	08/01/2018 20:40:42	3
74	08/01/2018 20:41:08	3
75	08/01/2018 20:41:14	3
76	08/01/2018 20:41:20	3
77	08/01/2018 20:41:26	3
78	08/01/2018 20:41:31	3
79	08/01/2018 20:41:37	3
80	08/01/2018 20:41:43	3

Figura 16: Planilha para Armazenar Dados do Sensor

O campo de resposta de cada pergunta do formulário tem um ID/KEY único, através disso é possível configurar cada sensor pra sua respectiva resposta individual e única.

3.3 Segunda Etapa

O objetivo nessa segunda etapa foi o desenvolvimento de uma plataforma intuitiva para monitoramento e gestão em tempo real do sistema hidráulico. Foi também desenvolvido uma Placa de Circuito Impresso (PCI) para o circuito eletrônico do sistema.

3.3.1 Sensor de Nível

Para obtenção de informações referentes à capacidade volumétrica disponível dos reservatórios de água desse projeto, utilizou-se sensores de nível do tipo boia magnética com saída de contato seco (NA/NF) da ICOS (Figura 17).

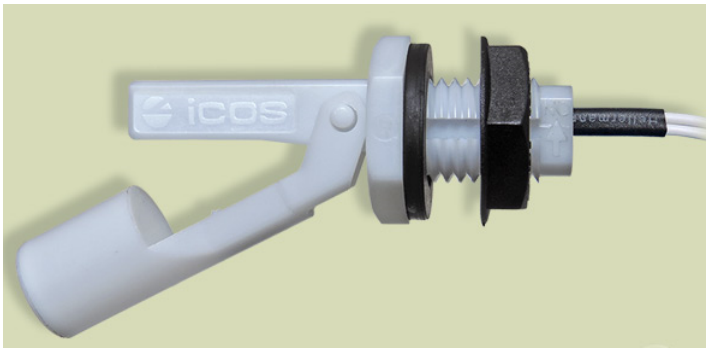


Figura 17: Sensor de Nível ICOS LA16M-40

Fonte: ICOS, 2018

O sensor LA16M-40 da fabricante ICOS, utilizado nesse projeto, funciona através da disposição do flutuador (boia) que quando em contato com o líquido, fecha-se o contato seco produzindo um sinal magnético que é interpretado por um microcontrolador como um sinal alto.

O fabricante nos fornece as seguintes informações técnicas do sensor:

– Material: POM - Poliacetal

– Porca em PA

– Pressão máx. de trabalho: 2bar

– Temp. de trabalho: -10°C a 100°C

– Cor: Branco

– Densidade mín. do líquido (SG): 0,76

– Peso:

– Saída: Contato On/Off

– Características elétricas: NA/NF - SPST

– Conexão elétrica: Cabo 2 x 0,5mm² x 40cm

– Grau de proteção: IP66

– Montagem: Lateral interna em furo de Ø16mm

– Vedação: Arruela NBR (borracha nitrílica)

– Espessura máx. parede reservatório: 9mm

– Raio mín. reservatório cilíndrico: 150mm

Tensão de Trabalho	Potência Máxima	Corrente Máxima	Corrente de Pico
110Vac	20VA	0,2A	0,5A @20ms
220Vac	20VA	0,1A	0,5A @20ms
5Vdc	2,5W	0,5A	1A @20ms
12Vdc	5W	0,5A	1A @20ms
24Vdc	10W	0,5A	1A @20ms

Tabela 1: Dados Sensor de Nível ICOS LA16M-40

Fonte: ICOS, 2018

3.3.2 Medindo o Nível de Água no Reservatório

Para medir o nível do reservatório da residência através do sensor ICOS LA16M-40, montou-se um circuito eletrônico com transistor funcionando como chave.

O Transistor tem diversas aplicações no mundo da eletrônica e uma das mais utilizadas é como chave eletrônica. Na figura 18 exemplifica-se o funcionamento do transistor nessa configuração. Trabalhando em regime de corte a “chave” está aberta, ou seja, IC=0 e VCE = VCC e quando trabalha em regime de saturação esssa "chave"está fechada, IC = VCC/RC.

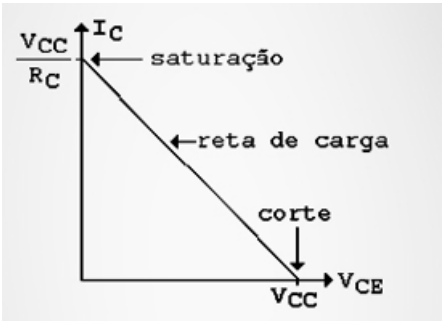


Figura 18: Regime de Corte e Saturação Transistor

Fonte: (MORAES, 2018)

Sendo,

VCC: tensão aplicada no coletor;

VBB: tensão aplicada na base;

VCE: tensão entre coletor e emissor;

VBE: tensão entre base e emissor;

IC: corrente no coletor;

IB: corrente na base;

RC: resistor conectado no coletor;

RB: resistor conectado na base.

Quando a base do transistor não tiver sinal, ou seja no nosso caso quando o sensor de nível estiver aberto, o transistor funcionará como uma chave aberta, IC= 0 e VCE = VCC (Figura 19).

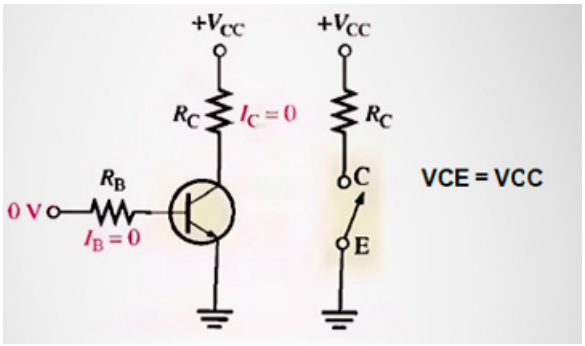


Figura 19: Transistor como chave aberta

Fonte: (MORAES, 2018)

Quando aplicamos um sinal na base do transistor, ou seja no nosso caso quando o sensor de nível estiver fechado, o transistor funciona como uma chave fechada de forma que VCE = 0,2V (Figura 20).

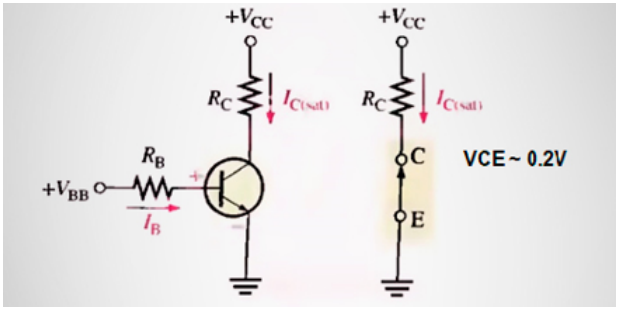


Figura 20: Transistor como chave fechada

Fonte: (MORAES, 2018)

O circuito envolvendo os quatro sensores de nível, que quando acionados representam uma porcentagem do total de água presente no reservatório, foi desenvolvido e simulado no *software Proteus* e pode ser visto na figura 21.

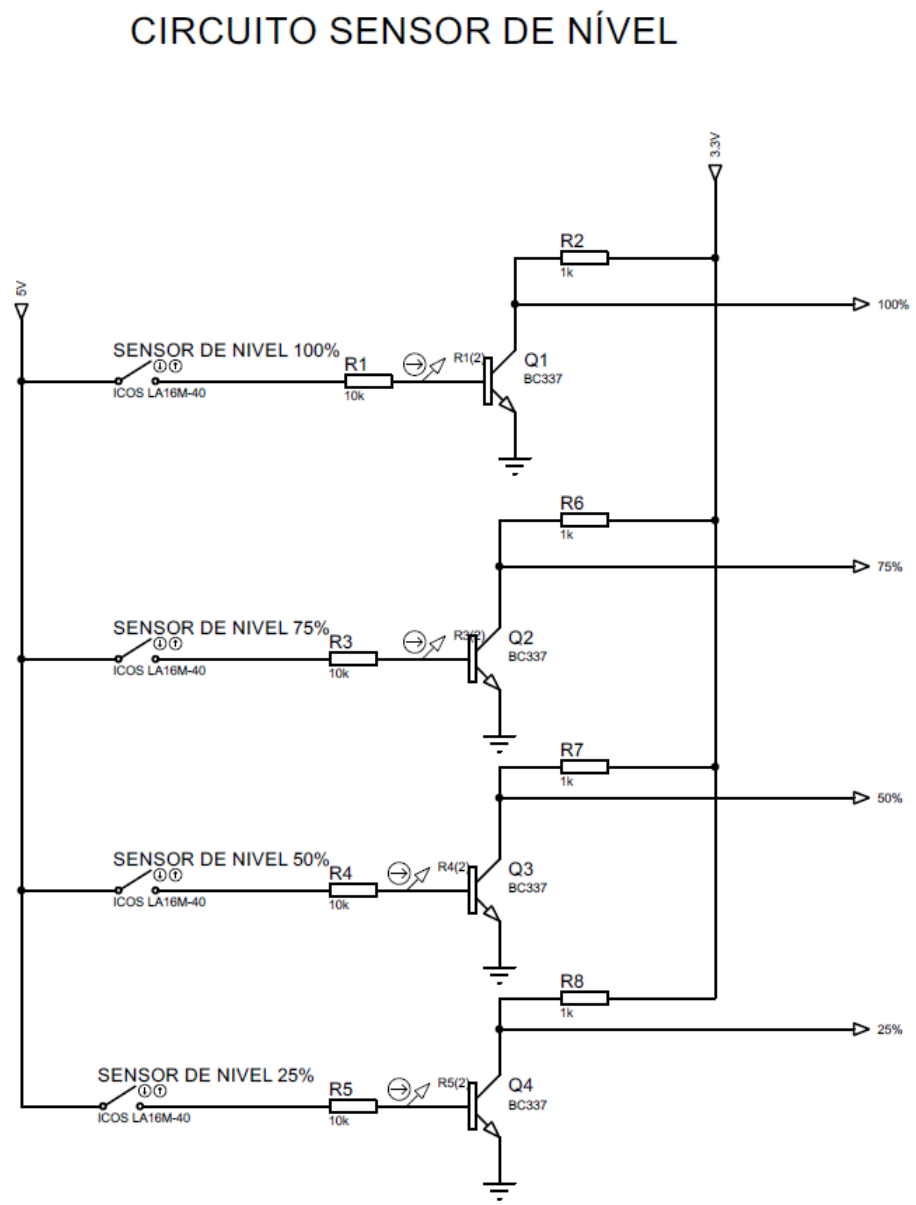


Figura 21: Circuito dos Sensores de Nível

3.3.3 Placa de Circuito Impresso

Para a produção das placas de circuito impresso do projeto foi utilizado o processo de foto-transferência, que por experiência prática é o método que entrega o melhor resultado excluindo a produção por CNC.

3.3.3.1 Material Necessário

- **Folha Transparente:** onde será impresso o circuito;
- **Placa de fenolite:** onde o circuito será montado;
- **Esponja de Aço:** para limpar a oxidação da placa;

- **Detergente:** para eliminar gordura e resíduos da placa;
- **Mini-retífica:** para cortar e furar a placa;
- **Tinta fotossensível:** tinta sensível à luz ultravioleta, que quando exposta a mesma, fixa na superfície preparada;
- **Luz ultravioleta:** utilizada para sensibilizar a tinta;
- **Bicarbonato de sódio:** utilizado em solução aquosa para a revelação do circuito;
- **Percloreto de Ferro:** utilizado para a corrosão da placa de circuito;
- **Soda Cáustica:** utilizada em solução aquosa para retirar a tinta da placa depois da corrosão;
- **Verniz:** para aplicar na parte cobreada da placa depois que a mesma estiver pronta.

3.3.3.2 Confecção da PCI

Os circuitos eletrônicos foram elaborados no *software Proteus* e foram exportados para impressão com as cores invertidas. Por padrão, o *Proteus* exporta para a impressão a separação entre trilhas na cor branca e o restante do circuito em cor preta (Figura 22).

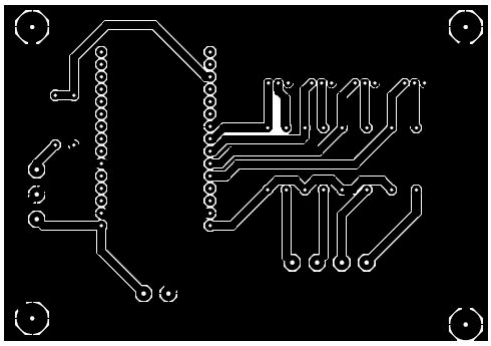


Figura 22: PCI Exportada do Proteus

Para o método de foto-transferência precisamos exatamente do contrário (Figura 23), que a separação entre trilhas fique na cor preta e o restante na cor branca. Com isso a tinta adere em toda a placa excluindo a separação entre as trilhas que deve ser corroída.

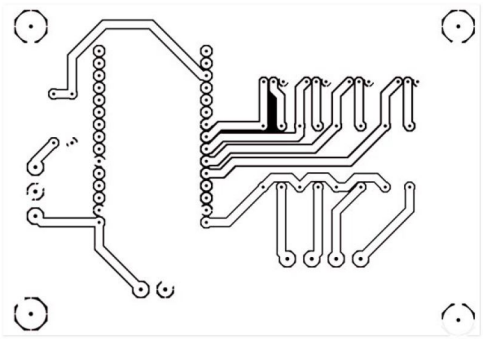


Figura 23: PCI Exportada do Proteus Invertida

3.3.3.3 Preparação do Fenolite

Afim de obter resultados mais precisos, deve-se limpar a placa com a esponja de aço para retirar qualquer traço de oxidação do cobre. Logo depois deve-se lavar a placa com detergente para a eliminação de gordura e em seguida deve-se secar. Logo após isso, já não deve-se colocar os dedos na parte de cobre (Figura 24).

Figura 24: PCI Preparada

3.3.3.4 Aplicação da Tinta no Fenolite

A placa deve ser fixada em um suporte giratório (Figura 25) como a Mini-retífica. Logo depois deve-se passar com ajuda de um pincel macio a tinta por toda a extensão de cobre da placa, evitando excessos. Em seguida liga-se a Mini-retífica em velocidade média para evitar tirar toda a tinta e através do processo de centrifugação a tinta é espalhada homogenicamente por toda a placa. Logo após a centrifugação com o auxilio de um secador, deve-se secar a tinta e aí a placa estará pronta para a etapa seguinte.

Figura 25: PCI Fixada na Mini-retífica

3.3.3.5 Transferência do Circuito para a Placa

A folha transparente com o circuito já impresso deve ser colocada na placa e presa com a ajuda de duas chapas de vidro ou acrílico. Em seguida em um local escuro, deve-se ligar a luz negra a cerca de 10 cm da placa e deixar dessa forma por cerca de 5 min. (Figura 26)

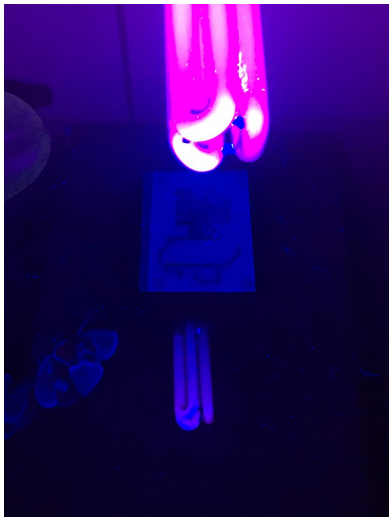


Figura 26: Transferência do Circuito para a Placa

3.3.3.6 Revelação do Circuito

Logo após a exposição da placa à luz negra, deve-se preparar uma solução aquosa com 300ml de água e uma colher de sopa de bicarbonato de sódio e mergulhar a placa nessa solução. Com a ajuda de uma esponja, deve-se esfregar levemente a placa a fim de retirar a tinta dos locais onde irão ocorrer a corrosão.(Figura 27)

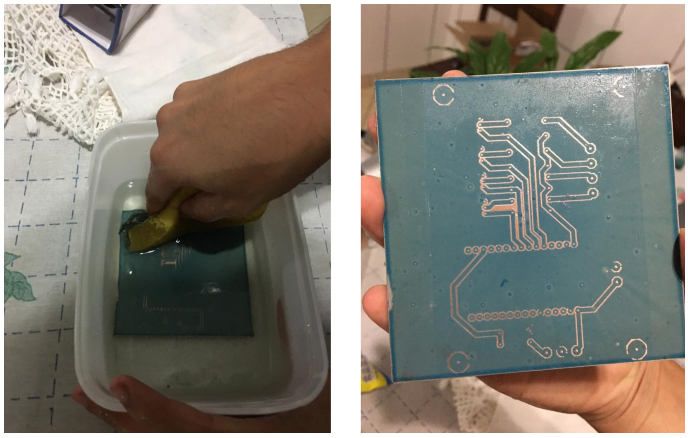


Figura 27: Revelação do Circuito

3.3.3.7 Corrosão da Placa

Após ter retirado qualquer vestígio de tinta da placa deve-se corroer a parte de cobre descoberto pela tinta. Para isso deve-se depositar a solução do Percloreto de Ferro em recipientes que não sejam metálicos, por exemplo, em garrafas de vidro ou embalagens de plástico. Em seguida mergulha-se a placa no percloreto e deixa-se a reação acontecer. É recomendado prender um pedaço de fio ou mexer com algum objeto de plástico a placa no percloreto e depois de alguns minutos deve-se retira-la sem entrar em contato com o ácido. (Figura 28)

Figura 28: Corrosão da Placa

3.3.3.8 Retirada da Tinta Restante

Depois de retirar a placa do ácido, deve-se lava-la e em uma outra vasilha, preparar a solução com 300ml de água e uma colher de sopa de soda cáustica. Mergulha-se a placa nessa solução e espera-se até que toda a tinta solte, em seguida devesse retirar e lavar a PCI e com isso a placa estará pronta para ser furada e soldada. (Figura 29)

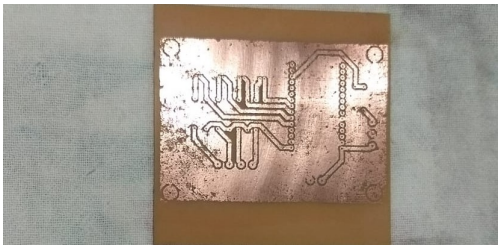


Figura 29: Placa Pronta para Solda

3.3.4 ThingSpeak

O MQTT é um protocolo comum usado em sistemas IoT para conectar dispositivos e sensores a um *Broker* na Internet conforme revisado no Capitulo 2. O ThingSpeak é um serviço que oferece um protocolo de comunicação baseado em http e serviço web para envio e recebimento de dados gerados por microcontroladores que tenham interface de comunicação via internet. O ThingSpeak adicionou recentemente um *broker* do MQTT para que os dispositivos possam enviar mensagens diretamente para sua plataforma.

As vantagens do ThingSpeak é a possibilidade de se criar uma conta gratuita que atende perfeitamente o objetivo desse trabalho. É possível também gerar visualizações de dados instantâneos e ou histórico de dados, além de poder interagir com um Aplicativo desenvolvido pela Cinetica, o ThingView. Por esses motivos a plataforma ThingSpeak foi escolhido como interface gráfica web do projeto em questão.

O ThingSpeak funciona como um *broker* MQTT que se comunica com um MQTT *Client*, no nosso caso o próprio NodeMCU (Figura 30). Os dados são enviados e mostrados em tempo real na forma de gráficos.

Figura 30: Comunicação NodeMCU e ThingSpeak

Fonte: (SCHARLER, 2017)

3.3.5 Código do NodeMCU

Para a aplicação desse projeto foi desenvolvido um código em linguagem de programação baseado em C++ para o NodeMCU ESP8266, que se encontra completo no Apêndice B.

As principais partes da lógica de programação usadas no código será exemplificada a seguir.

A primeira parte do código é dedicada às definições de portas e parâmetros de comunicação em rede Wifi e *Broker* MQTT.

```
#define D2 4
#define D5 14
#define D6 12
#define D7 13
#define D8 15

const char* wifiSSID = "NOME_DA_REDE";
const char* wifiPassword = "SENHA_DA_REDE";
const char* mqttID = "ID_MQTT";
const char* mqttUserName = "USUÁRIO_THINGSPEAK";
const char* mqttAPIKey = "SENHA_THINGSPEAK";
const char* writeAPIKey = "CHAVE_DE_ESCRITA";
const char* readAPIKey = "CHAVE_DE_LEITURA";
const char* channelID = "ID_DO_CANAL";
const char* mqttServer = "mqtt.thingspeak.com";
uint16_t mqttPort = 1883;
const char* topicRead =
↪ "channels/channelID/subscribe/fields/field1/readAPIKey";
```

Em seguida declarou-se os protótipos das funções usadas no programa. Funções são ferramentas usadas em programação que tem como objetivo reduzir o tempo de codificação e tempo de depuração. Essa prática reduz também o tamanho do programa , e melhora a legibilidade do mesmo.

```
void setWiFiConnection(void);
void setMQTTConnection(void);
void mqttCallback(char* topic, byte* payload, unsigned int length);
void checkWiFiAndMQTTconnnection(void);
void pulsesCounter(void);
void resetPulsesCounter(void);
void nivelcheck(void);
void calculateFlowRateAndConsumption(void);
void sendFlowRateConsumptionAndNivelByMQTT(void);
void sendFlowRateConsumptionAndNivelToSerial(void);
void turnOnPulsesCounter(void);
void turnOffPulsesCounter(void);
```

Os trechos seguintes mostram as principais funções do programa. A "turnOnPulses-Counter"habilita as interrupções associadas ao pino D2, sendo que a função “pulseCounter” deverá ser chamada sempre que houver um evento de borda de subida no pino D2. Já a "turnOffPulsesCounter"desabilita as interrupções para continuar a depuração do código principal.

```
void turnOnPulsesCounter(void)
{
  attachInterrupt(digitalPinToInterrupt(D2), pulsesCounter, RISING);
}

void turnOffPulsesCounter(void)
{
  detachInterrupt(digitalPinToInterrupt(D2));
}
```

A função "pulsesCounter"soma a quantidade de pulsos cada vez que ocorre uma interrupção, ou seja, toda vez que a água passa pelo sensor de vazão ela gera pulsos e esses são contabilizados pela função em questão. A "resetPulsesCounter"é chamada toda vez que o programa vai fazer uma nova medição, para isso ela deve zerar o contador de pulsos afim de gerar uma nova medida de vazão.

```
void pulsesCounter(void)
{
  *(volatile long*)&interruptionsCounter += 1;
}

void resetPulsesCounter(void)
{
  interruptionsCounter = 0;
}
```

Na função "calculateFlowRateAndConsumption", transformamos os pulsos contados pela função "pulsesCounter"em litro e litros/min. A cada 450 pulsos dado pelo sensor de vazão, significa dizer que passou 1 litro do fluido pelo sensor.

```
void calculateFlowRateAndConsumption(void)
{
  flowRate = (float) ((float)interruptionsCounter / (float)450.0);
  consumption = consumption + flowRate;
  flowRate = flowRate*60.0;
}
```

Já a "nivelCheck"verifica a cada 20 segundos o nível da caixa d'água principal da residência.

```
void nivelCheck(void)
{
  if ( digitalRead(D5)==HIGH){
     nivel=100;
  }else if (digitalRead(D6)==HIGH){
    nivel=75;
  }else if (digitalRead(D7)==HIGH){
    nivel=50;
  }else if (digitalRead(D8)==HIGH){
    nivel=25;
  }else {
    nivel=0;
  }
}
```

A "sendFlowRateConsumptionAndNivelByMQTT"é a função responsável pelo envio de dados das instancias medidas e calculadas pelo NodeMcu para o *Broker* MQTT, no nosso caso o ThingSpeak.

```
1  void sendFlowRateConsumptionAndNivelByMQTT(void)
2  {
3    String data = String("field1=" + String(consumption));
4    data = String(data + String("&field2=" + String(flowRate)));
5    data = String(data + String("&field3=" + String(nivel)));
6    int length = data.length();
7    char msgBuffer[length];
8    data.toCharArray(msgBuffer, length+1);
9    String topic = String("channels/" + String(channelID));
10   topic = String(topic + String("/publish/" + String(writeAPIKey)));
11   length = topic.length();
12   char topicBuffer[length];
13   topic.toCharArray(topicBuffer, length+1);
14   mqttClient.publish(topicBuffer, msgBuffer);
15   lastSendTime = millis();
16 }
```

4 Experimentos e discussão dos resultados

Neste Capítulo apresenta-se os resultados obtidos nas etapas 1 e 2, e discuti-se esses dados fazendo comparativos e projeções.

4.1 Primeira Etapa

A primeira etapa desse estudo teve como objetivo o teste de viabilidade e eficácia de um medidor de vazão por efeito hall na medição de consumo de água em alguns pontos de uma residência. Esses pontos escolhidos foram os maiores produtores de água cinza, água essa que pode ser reaproveitada para fins não potáveis. Sendo assim, fizemos uma relação de consumo de água potável versus produção de água para reuso.

Durante 7 dias foram levantados dados sobre a quantidade de água consumida com chuveiro, pia do banheiro e pia da cozinha numa casa onde vivem 4 pessoas. Os dados conforme apresentados no capítulo anterior, foram coletados e salvos em tempo real num planilha online do Google (Figura 31). Os dados foram enviados para a planilha em média a cada 4 segundos, esse tempo depende exclusivamente da velocidade de *upload* da internet a qual o NodeMCU ESP 8266 está conectado. Essa média de tempo é mais do que suficiente para termos uma base de dados muito precisa, podendo identificar picos de gastos proveniente de vazamentos em questão de segundos, por exemplo.

1	Carimbo de data/hora	Pia banheiro (litros consumidos)	Chuveiro	Pia Cozinha
25886	1/10/2018 14:50:53	30	785	64
25887	1/10/2018 14:50:59	30	785	64
25888	1/10/2018 14:51:05	30	785	64
25889	1/10/2018 14:51:11	30	785	64
25890	1/10/2018 14:51:16	30	785	64
25891	1/10/2018 14:51:22	30	785	64
25892	1/10/2018 14:51:28	30	785	64
25893	1/10/2018 14:51:33	30	785	64
25894	1/10/2018 14:51:39	30	785	64
25895	1/10/2018 14:51:45	30	785	64
25896	1/10/2018 14:51:50	30	785	64
25897	1/10/2018 14:51:56	30	785	64
25898	1/10/2018 14:52:01	30	785	64
25899	1/10/2018 14:52:07	30	785	64
25900	1/10/2018 14:52:13	30	785	64
25901	1/10/2018 14:52:19	30	785	64
25902	1/10/2018 14:52:25	30	785	64
25903	1/10/2018 14:52:30	30	785	64
25904	1/10/2018 14:52:37	30	785	64
25905	1/10/2018 14:52:42	30	785	64
25906	1/10/2018 14:52:48	30	785	64
25907	1/10/2018 14:52:54	30	785	64
25908	1/10/2018 14:52:59	30	785	64
25909	1/10/2018 14:53:05	30	785	64
25910	1/10/2018 14:53:11	30	785	64
25911	1/10/2018 14:53:17	30	785	64

Respostas ao formulário 1 | Página1

Figura 31: Dados Salvos em Planilha Online

A partir dos dados salvos nessa planilha geramos gráficos com o resultado do levantamento realizado nos sete dias conforme figuras 32, 33, 34.

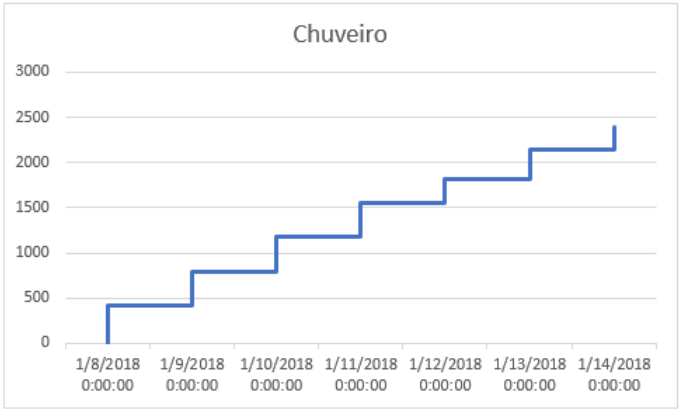


Figura 32: Água Consumida no Chuveiro

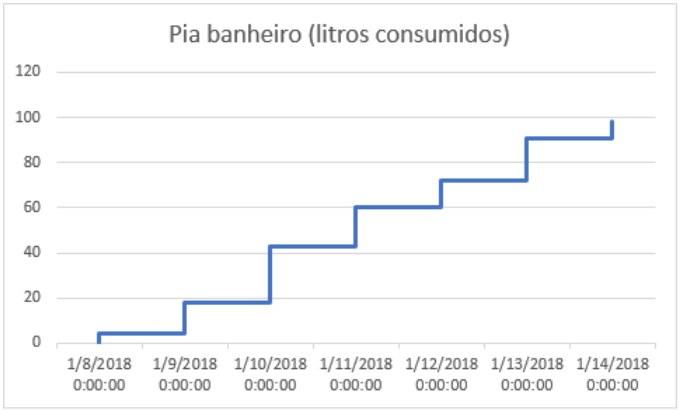


Figura 33: Água Consumida na Pia do Banheiro

Figura 34: Água Consumida na Pia do Cozinha

A partir desses dados observa-se que o consumo médio de água nos pontos medidos é de 102,07 (litros por pessoa) /dia. Fazendo a projeção por mês, tem-se um gasto de 3062 (litros por pessoa) /mês. Como na casa vivem 4 pessoas, a projeção da capacidade dessa residência de produzir água de reuso por mês é de 12249 litros, desconsiderando qualquer tipo de perda. No mês de janeiro de 2018, quando esse levantamento foi feito, o consumo total dessa casa foi de 23 mil litros d'água segundo a conta da Copasa. Ou seja, 53,25% do total de água consumido nessa residência poderia ser reaproveitado para fins não potáveis.

4.2 Segunda Etapa

A segunda parte desse estudo teve como objetivo a integração de um sistema de monitoramento do sistema hidráulico de uma residência com uma interface gráfica Web, onde é possível visualizar dados em tempo real e/ou os históricos dos mesmos. Em posse dessas informações é possível gerir de forma eficiente os gastos de água, bem como detectar desperdícios e vazamentos na rede.

Foi desenvolvido um sistema constituído de *hardware e software* onde é possível monitorar através de uma página Web ou por um aplicativo, o consumo de água de uma residência e o nível do reservatório de água da casa. Para isso foi confeccionada uma placa de circuito impresso, PCI, contendo toda a parte eletrônica do sistema que comunica com um *Broker* MQTT, onde tem-se acesso a uma interface gráfica para visualização dos dados, o ThingSpeak.

Na primeira etapa desse projeto foi utilizada a memória EEPROM como dispositivo de segurança dos dados de consumo de água em caso de queda de energia, ou *reset* inesperado do microcontrolador. Como era uma fase de testes inicial, essa estratégia se fez útil e bem eficaz. Para um sistema que funcionará de forma permanente essa estratégia não é a mais recomendada para essa aplicação. A memória EEPROM dos NodeMCU

ESP8266, na verdade é emulada sobre a memória *flash* (MELO et al., 2018). A memória *flash* utilizada por esse módulo, tem a limitação de cerca de 100.000 operações de escrita (WINBOND, 2007). Como a aplicação será de uso contínuo, não se pode projetar projetar tal sistema com uma limitação interstícia. Para contornar essa situação, foram utilizados recursos do próprio protocolo MQTT que foram resolvidos a partir da função *"callback"*. Segundo Melo et al. (2018) o papel da função *"callback"* é reagir aos eventos de publicação no(s) tópico(s) ao(s) qual(is) se está assinado, recuperando a(s) mensagem(s) publicada(s). Ou seja, a ultima informação enviada pelo NodeMCU para o ThingSpeak a respeito do consumo é recuperada pela função *"callback"*. Essa informação é carregada novamente na variável do programa responsável pelo armazenamento da mesma caso ocorra uma situação de *reset* inesperado ou desligamento anormal do sistema.

A interface visual criada no ThingSpeak, que permite monitorar o consumo, vazão e o nível de água do reservatório, além de mostrar a localização geográfica do local onde os dados estão sendo enviados, serão demonstrados a seguir.

A figura 35 mostra o gráfico gerado pela ThingSpeak onde é possível acompanhar em tempo real a vazão instantânea do sistema.

Figura 35: Vazão em L/min

Já o consumo em litros do sistema pode ser visto na figura 36. É possível também exportar esses dados para uma planilha excel ou fazer um link direto com o MatLab, para análise do perfil de consumo, linhas de tendências etc.

Figura 36: Consumo em L

A condição do nível do reservatório principal da casa é demonstrada em tempo real através de um gráfico ilustrado na figura 37.

Figura 37: Nível do Reservatório

O ThingSpeak também fornece a localização exata de onde os dados estão sendo enviados.

Figura 38: Localização do Sistema

O ThingSpeak inicialmente foi desenvolvido para *desktop*, mas hoje já conta também com o ThingView (Figura 39), aplicativo desenvolvido pela Cinética que permite dispositivos móveis terem acesso a canais do ThingSpeak. Para isso basta entrar com o número e a chave de leitura única do canal para que o aplicativo tenha acesso em tempo real as informações contidas no ThingSpeak.

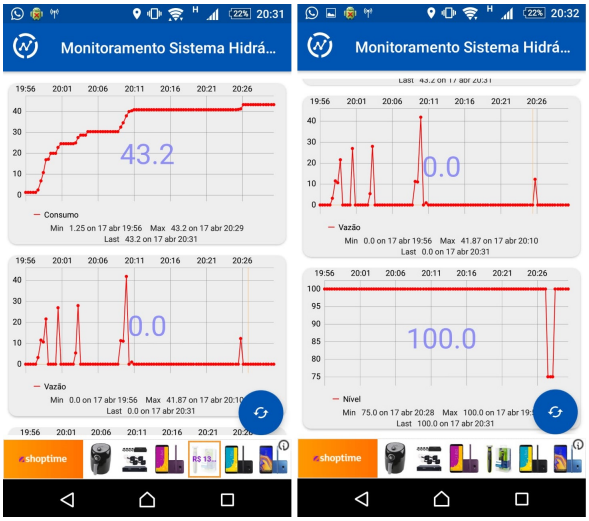


Figura 39: Aplicativo ThingView com acesso ao ThingSpeak

A montagem física do sistema pode ser vista nas figuras a seguir.

Figura 40: Sensor de Nível Instalado

Figura 41: Saída dos Cabos do Sensor de Nível

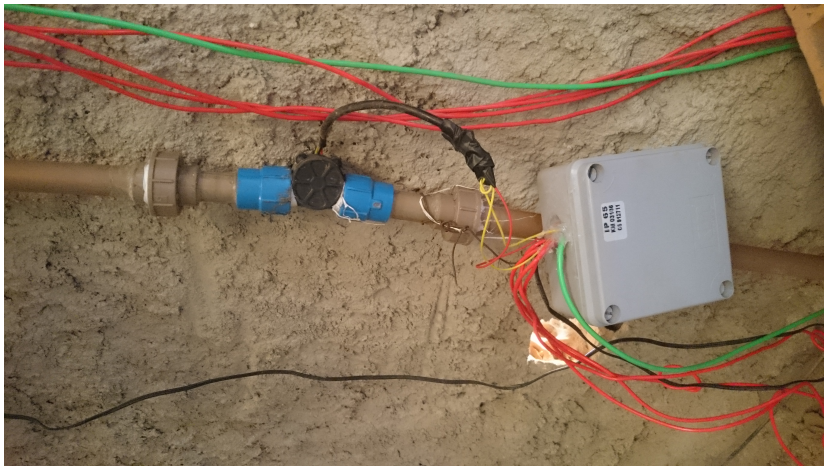


Figura 42: Sensor de vazão Instalado e Caixa Com o Microcontrolador

Figura 43: Visão Geral do Sistema Instalado

5 Conclusão

O projeto em estudo teve como motivação a criação de uma ferramenta que fosse capaz de monitorar o consumo de água de edificações em tempo real a fim de evitar desperdícios, apontar possibilidades de reúso e tornar o consumo desse bem tão importante mais eficiente e consciente. Para isso foi projetado e executado um sistema contendo *Hardware e Software*, que usa sensores eficientes e de baixo custo associados a uma interface gráfica disponível na Web que permite visualizar os dados aferidos remotamente e em tempo real.

Com os resultados obtidos na primeira parte desse trabalho foi possível concluir que uma parcela relevante da água que utilizamos no dia a dia poderia ser reaproveitada, gerando economia para as casas onde um sistema desse tipo fosse implementado. A utilização do Google Planilhas se mostrou bastante usual para guardar os dados obtidos e a possibilidade de se analisar tais valores através de gráficos no Excel dá ao usuário oportunidade de gerir melhor o seu gasto com a água.

A criação de uma interface gráfica que pudesse ser acessada de qualquer lugar via Internet, onde o usuário tivesse todas as informações relativas a vazão, consumo e nível de reservatório de sua casa foi desenvolvida na segunda etapa. O ThingSpeak atendeu as demandas propostas como *Broker*, apresentando diversas possibilidades de integração com *softwares* de análise de dados, como o Excel, Matlab e ThingView. Os dados atualizados a cada 20 segundos foram suficientes para entregar ao usuário uma informação em tempo real que dá a oportunidade identificar possíveis vazamentos de forma rápida.

Um dos principais pontos do projeto que gerou dificuldades foi a confecção da PCI, uma vez que não utilizou-se tinta já diluída com o fotossensibilizante. A proporção indicada no manual de uso da tinta não se mostrou eficaz, gerando assim diversas experiências para chegar numa formula eficaz de diluição, que foi da ordem de 7% de fotossensibilizante.

O sistema pode ser aplicado na medição de diversas grandezas, como pressão do sistema, consumo energético, e acionamento de cargas. Para trabalhos futuros a inserção de um banco de dados dedicado, um *Broker* MQTT local e uma página da web customizada são possíveis pontos de melhoria. Espera-se também que este estudo contribua para trabalhos futuros focados na monitoração do consumo de água na UFOP e em Ouro Preto, havida conta de que a falta de hidrometração induz ao desperdício de forma irresponsável deste bem cada vez mais escasso.

Referências

ALBERTIN, A. L.; ALBERTIN, R. M. de M. A internet das coisas irá muito além as coisas. **GV-Executivo**, 2017. v. 16, n. 2, p. 12–17, 2017.

ALEGRE, H.; COELHO, S. T.; ALMEIDA, M. d. C.; VIEIRA, P. Controlo de perdas de água em sistemas públicos de adução e distribuição. **Série guias técnicos**, 2005. v. 3, 2005.

ASHTON, K. That 'internet of things' thing. **RFID journal**, 2009. Jun, v. 22, n. 7, p. 97–114, 2009.

AZEVEDO, R. J. d. et al. Uma arquitetura para execução de codigo comprimido em sistemas dedicados. 2002. [sn], 2002.

BALL, S. **Embedded microprocessor systems: real world design**. [S.l.]: Elsevier, 2002.

BAZZARELLA, B. B. **Caracterização e aproveitamento de água cinza para uso não-potável em edificações**. Dissertação (Mestrado) — Universidade Federal do Espírito Santo, 2005.

BOYLESTAD, R. L. Introdução à análise de circuitos elétricos. **Tradução de José**, 2012. 2012.

CHASE, O. A. **Projeto e Construção de um Robô Móvel AGV/ROV Não-Holonômico com Habilidade para Navegação Autônoma do Tipo Wall-Following**. Tese (Doutorado) — Dissertação de Mestrado, Programa de Pós-Graduação em Engenharia Elétrica e . . . , 2009.

CORAUCCI, G. d. O. et al. Sensor de pressão microeletronico baseado no efeito piezoresistivo transversal em silicio. 2008. [sn], 2008.

FARINES, J.-M.; FRAGA, J. d. S.; OLIVEIRA, R. d. Sistemas de tempo real. **Escola de Computação**, 2000. v. 2000, p. 201, 2000.

HALLIDAY, D.; RESNICK, R.; WALKER, J. **Fundamentos de Física, vol. 2: gravitação, ondas e termodinâmica**. [S.l.]: Rio de Janeiro: LTC, 2009.

HORIGOSHI, F. M. **Uso de sensor efeito Hall para medição da vazão de fluidos no processo de produção de palatabilizantes a base de proteína animal em escala piloto**. Tese (Doutorado) — Universidade de São Paulo, 2016.

ISA, I. S. **Água nas metrópoles, o risco de escassez**. 2008. Acessado em 13/05/2018. Disponível em: <http://www.socioambiental.org/pt-br/blog/blog-do-isa/agua-nas-metropoles-o-risco-da-escassez>.

LESCHE, B. J. Fiii 5/07 divisor de tensão. **Apostila de aula Física 3 da UFJF**, 2013. 2013.

LIMA, A. S. I. Ratio of real to apparent losses in brazil. **Water Loss Conference and Exhibition - Waterloss**, 2018. 2018.

MATHIAS, R. B. **Influencia do perfil de velocidade do escoamento sobre a medição ultrassônica de vazão por tempo de trânsito**. Dissertação (Mestrado) — Universidade Federal do Espírito Santo, 2010.

MCIDADES. Sistema nacional de informações sobre saneamento. **Ministério das Cidades do Brasil**, 2018. 2018.

MECATRÔNICA, A. **Medidor de vazo tipo magnético**. [S.l.]: Editora Saber, 2005. (ED. 020).

MELO, R. C. d. S. et al. Sistema de monitoramento de consumo de água utilizando o protocolo de comunicação mqtt. 2018. Universidade Federal de Uberlândia, 2018.

MORAES, F. **Utilizando o Transistor Como Chave Eletrônica**. 2018. Acessado em 10/09/2018. Disponível em: <http://blog.baudaeletronica.com.br/transistor-chave-eletronica/>.

MORAIS, J. **Banco de dados com Google planilhas – ESP**. 2017. Acessado em 23/11/2017. Disponível em: <https://portal.vidadesilicio.com.br/banco-de-dados-com-google-planilhas-esp/>.

NBR, A. 15527: Água de chuva–aproveitamento de coberturas em áreas urbanas para fins não potáveis–requisitos. **ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS Rio de Janeiro**, 2007. 2007.

NETO, J. F.; SANTOS, L. A. d. A. Automação de redes hidráulicas e reuso de água no edifício nzeb com sistemas supervisórios. 2016. 2016.

OLIVEIRA, T. N. d. Projeto de captação e aproveitamento da água de chuva em uma indústria no município de elói mendes-mg. 2017. Fundação de Ensino e Pesquisa do Sul de Minas, 2017.

ROTAGINE. **Água: Sabendo usar não vai faltar**. 2017. Acessado em 22/04/2017. Disponível em: <http://www.rotogine.com.br/site/?page_id=205>.

SCHARLER, H. **Use MQTT to Send IoT Data to ThingSpeak**. 2017. Acessado em 06/01/2017. Disponível em: <https://blogs.mathworks.com/iot/2017/01/20/use-mqtt-to-send-iot-data-to-thingspeak/>.

SILVEIRA, B. Q. da. Reuso da /'agua pluvial em edificações residenciais. 2008. Universidade Federal de Minas Gerais, 2008.

SUNDMAEKER, H.; GUILLEMIN, P.; FRIESS, P.; WOELFFLÉ, S. Vision and challenges for realising the internet of things. **Cluster of European Research Projects on the Internet of Things, European Commision**, 2010. v. 3, n. 3, p. 34–36, 2010.

TOMAZ, P. Aproveitamento de água de chuva de cobertura em área urbana para fins não potáveis. **São Paulo: Navegar**, 2007. 2007.

TORRES, A. B.; ROCHA, A. R.; SOUZA, J. N. D. Análise de desempenho de brokers mqtt em sistema de baixo custo. In: **Anais do XXXVI Congresso da Sociedade Brasileira de Computaça o**. [S.l.: s.n.], 2016. p. 2804–2815.

WINBOND. **8M-BIT, 16M-BIT AND 32M-BIT SERIAL FLASH MEMORY WITH DUAL AND QUAD SPI**. 2007.

YUAN, M. **Conhecendo o MQTT**. 2017. Acessado em 07/02/2018. Disponível em: <https://www.ibm.com/developerworks/br/library/iot-mqtt-why-good-for-iot/index-.html>.

Apêndices

APÊNDICE A – Código do ESP8266 Sensor de Vazão Planilha Online

```
#include <ESP8266WiFi.h>
#include <EEPROM.h>

WiFiClientSecure client;//Cria um cliente seguro (para ter acesso ao HTTPS)
String textFix ="GET /forms/d/e/1FOUOLUplM_yKOQX4-vg/formResponse?ifq&entry.156818112765=";
//Essa String sera uma auxiliar contendo o link utilizado pelo GET, para nao precisar ficar re-escrevendo toda hora
const char* ssid = ""; //VARIÁVEL QUE ARMAZENA O NOME DA REDE SEM FIO EM QUE VAI CONECTAR
const char* password = ""; //VARIÁVEL QUE ARMAZENA A SENHA DA REDE SEM FIO EM QUE VAI CONECTAR
const int buttonPin = D2; // variable for D2 pin
int contagem = 0; // variable to store the "rise ups" from the flowmeter pulses
int litros = 0;
int addr = 0; //endereço eeprom
void pin_ISR()
{
  contagem++;
}
void setup()
{
  Serial.begin(115200);//Inicia a comunicacao serial
  WiFi.mode(WIFI_STA);//Habilita o modo estaçao
  delay(10); //INTERVALO DE 10 MILISEGUNDOS
  EEPROM.begin(8);
  litros=EEPROM.read(addr);
  pinMode(buttonPin, INPUT);
  Serial.println(""); //PULA UMA LINHA NA JANELA SERIAL
  Serial.println(""); //PULA UMA LINHA NA JANELA SERIAL
```

```
27    Serial.print("Conectando a "); //ESCREVE O TEXTO NA SERIAL
28    Serial.print(ssid); //ESCREVE O NOME DA REDE NA SERIAL
29
30    WiFi.begin(ssid, password); //PASSA OS PARÂMETROS PARA A FUNÇÃO QUE
      ↪ VAI FAZER A CONEXÃO COM A REDE SEM FIO
31    while (WiFi.status() != WL_CONNECTED) { //ENQUANTO STATUS FOR
      ↪ DIFERENTE DE CONECTADO
32      delay(500); //INTERVALO DE 500 MILISEGUNDOS
33      Serial.print("."); //ESCREVE O CARACTER NA SERIAL
34    }
35    Serial.println(""); //PULA UMA LINHA NA JANELA SERIAL
36    Serial.print("Conectado a rede sem fio "); //ESCREVE O TEXTO NA SERIAL
37    Serial.println(ssid); //ESCREVE O NOME DA REDE NA SERIAL
38    Serial.print("IP para se conectar ao NodeMCU: "); //ESCREVE O TEXTO NA
      ↪ SERIAL
39    Serial.print("http://"); //ESCREVE O TEXTO NA SERIAL
40    Serial.println(WiFi.localIP()); //ESCREVE NA SERIAL O IP RECEBIDO
      ↪ DENTRO DA REDE SEM FIO (O IP NESSA PRÁTICA É RECEBIDO DE FORMA
      ↪ AUTOMÁTICA)
41    attachInterrupt(digitalPinToInterrupt(buttonPin), pin_ISR, RISING);
42  }
43
44  void loop()
45  {
46     if(contagem > 440 )
47    {
48      litros++;
49      Serial.println();
50      Serial.print("Litros: ");
51      Serial.print(litros);
52      EEPROM.write(addr, litros);
53      EEPROM.commit();
54      contagem = 0;
55    }
56      if (client.connect("docs.google.com", 443) == 1)//Tenta se conectar
        ↪ ao servidor do Google docs na porta 443 (HTTPS)
57      {
58          String toSend = textFix;//Atribuimos a String auxiliar na nova
            ↪ String que sera enviada
```

```
59          toSend +=litros;
60          toSend += "&submit=Submit HTTP/1.1";//Completamos o metodo GET
            ↪ para nosso formulario.
61
62          client.println(toSend);//Enviamos o GET ao servidor-
63          client.println("Host: docs.google.com");//-
64          client.println();//-
65          client.stop();//Encerramos a conexao com o servidor
66          Serial.println("Dados enviados.");//Mostra no monitor que foi
            ↪ enviado
67      }
68      else
69      {
70          Serial.println("Erro ao se conectar");//Se nao for possivel
            ↪ conectar no servidor, ira avisar no monitor.
71      }
72
73      delay(5000);
74  }
```

APÊNDICE B – Código do NodeMCU ESP8266 Monitoramento do Sistema Hidráulico

Código Adaptado de Melo et al. (2018)

```
#include <ESP8266WiFi.h> // Biblioteca que possibilita utilizar o
#include <PubSubClient.h> // Biblioteca que permite utilizar o protocolo MQTT


// Definição de pinos/GPIOs

#define D2 4
#define D5 14
#define D6 12
#define D7 13
#define D8 15

// Parâmetros da conexão

const char* wifiSSID = "NOME_DA_REDE"; // Nome da rede na qual se deseja conectar
const char* wifiPassword = "SENHA_DA_REDE"; // Senha da rede na qual se deseja se conectar

// Parâmetros da conexão com o Broker MQTT

const char* mqttID = "ID_MQTT"; // ID para identificação da sessão de conexão ao MQTT
const char* mqttUserName = "USUÁRIO_THINGSPEAK"; // Nome de usuário para se conectar  ao ThingSpeak
const char* mqttAPIKey = "SENHA_THINGSPEAK"; // Senha para se conectar ao ThingSpeak
```

```
23 const char* writeAPIKey = "CHAVE_DE_ESCRITA"; // chave de escrita no
   ↪ canal
24 const char* readAPIKey = "CHAVE_DE_LEITURA"; // Chave de leitura do
   ↪ canal
25 const char* channelID = "ID_DO_CANAL"; // ID de identificaÃ§Ã£o do canal
   ↪ na plataforma ThingSpeak
26 const char* mqttServer = "mqtt.thingspeak.com"; // URL do MQTT que se
   ↪ deseja utilizar
27 uint16_t mqttPort = 1883; // Porta do MQTT
28
29 // Tópico de leitura
30
31 const char* topicRead =
   ↪ "channels/channelID/subscribe/fields/field1/readAPIKey";
32
33 // Inicialização das bibliotecas WiFiClient e PubSubClient
34
35 WiFiClient nodeMCUClient; // Construtor de classe "WiFiClient" utilizado
   ↪ para criar a instância "nodeMCUClient"
36 PubSubClient mqttClient(nodeMCUClient); // Construtor de classe
   ↪ "PubSubClient", utilizado para criar a instância "mqttClient", que
   ↪ recebe o objeto "nodeMCUClient"
37
38 // Variáveis globais
39
40 long interruptionsCounter;
41 unsigned long lastMillis;
42 unsigned long lastSendTime;
43 const unsigned long interval = 20000;
44 float flowRate;
45 float consumption;
46 float nivel;
47 unsigned int flag = 0;
48
49 // Protótipos das funções
50
51 void setWiFiConnection(void);
52 void setMQTTConnection(void);
53 void mqttCallback(char* topic, byte* payload, unsigned int length);
```

```
54 void checkWiFiAndMQTTconnnection(void);
55 void pulsesCounter(void);
56 void resetPulsesCounter(void);
57 void nivelcheck(void);
58 void calculateFlowRateAndConsumption(void);
59 void sendFlowRateConsumptionAndNivelByMQTT(void);
60 void sendFlowRateConsumptionAndNivelToSerial(void);
61 void turnOnPulsesCounter(void);
62 void turnOffPulsesCounter(void);
63
64 // Estabelece a conexão
65
66 void setWiFiConnection(void)
67 {
68   Serial.println();
69   Serial.println("------Conexão Wi-Fi------");
70   Serial.println();
71   Serial.print("Conectando-se na rede ");
72   Serial.println(wifiSSID);
73   Serial.print("Aguarde");
74
75   if (WiFi.status() == WL_CONNECTED) {
76     return;
77   }
78
79   WiFi.begin(wifiSSID, wifiPassword);
80
81   while (WiFi.status() != WL_CONNECTED) {
82     delay(100);
83     Serial.print(".");
84   }
85
86   Serial.println();
87   Serial.print("Conectado com sucesso na rede ");
88   Serial.print(wifiSSID);
89   Serial.println();
90   Serial.print("IP obtido: ");
91   Serial.print(WiFi.localIP());
92   Serial.println();
```

```
93  }
94
95  // Estabelece a conexão com o MQTT
96
97  void setMQTTConnection()
98  {
99    mqttClient.setServer(mqttServer,mqttPort);
100   mqttClient.setCallback(mqttCallback);
101
102   while (!mqttClient.connected()) {
103     Serial.println();
104     Serial.println("------Conexão Broker MQTT------");
105     Serial.println();
106     Serial.print("Tentando se conectar ao Broker MQTT ");
107     Serial.println(mqttServer);
108     if (mqttClient.connect(mqttID,mqttUserName,mqttAPIKey)) {
109       Serial.println("Conectado com sucesso ao broker MQTT!");
110       mqttClient.subscribe(topicRead); // realiza a assinatura do tópico
          ↪ "topicRead"
111     }else {
112       Serial.println("Falha ao reconectar no broker!");
113       Serial.println("Causa: ");
114       Serial.print(mqttClient.state());
115       Serial.println();
116       Serial.println("Haverá nova tentativa de conexão em 5s");
117       delay(5000);
118     }
119   }
120 }
121
122 // Função chamada quando há publicações no tópico ao qual se está
    ↪ assinando
123
124 void mqttCallback(char* topic, byte* payload, unsigned int length)
125 {
126   payload[length] = '\0';
127   String s = String((char*)payload);
128   float salveConsumption = s.toFloat();
129   consumption = consumption + salveConsumption;
```

```
130   Serial.println();
131   Serial.print("Mensagem da função callback: ");
132   Serial.println(salveConsumption);
133   flag = 1;
134 }
135
136 // Verifica as conexões e MQTT
137
138 void checkWiFiAndMQTTconnnection(void)
139 {
140   if (WiFi.status() != WL_CONNECTED) {
141     setWiFiConnection();
142   }if (!mqttClient.connected()) {
143     setMQTTConnection();
144   }
145 }
146
147 // Conta a quantidade de pulsos
148
149 void pulsesCounter(void)
150 {
151   *(volatile long*)&interruptionsCounter += 1;
152 }
153
154 // Reseta a variável "interruptionsCounter"
155
156 void resetPulsesCounter(void)
157 {
158   interruptionsCounter = 0;
159 }
160
161 // Calcula a vazão e o consumo de água
162
163 void calculateFlowRateAndConsumption(void)
164 {
165   flowRate = (float) ((float)interruptionsCounter / (float)450.0);
166   consumption = consumption + flowRate;
167   flowRate = flowRate*60.0;
168 }
```

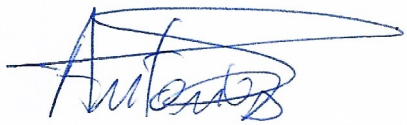
```
169
170 // Verifica o nível do reservatório
171
172 void nivelcheck(void)
173 {
174   if ( digitalRead(D5)==HIGH){
175      nivel=100;
176   }else if (digitalRead(D6)==HIGH){
177     nivel=75;
178   }else if (digitalRead(D7)==HIGH){
179     nivel=50;
180   }else if (digitalRead(D8)==HIGH){
181     nivel=25;
182   }else {
183     nivel=0;
184   }
185 }
186
187 // Envia a vazão, consumo e nível de água através do protocolo MQTT
188
189 void sendFlowRateConsumptionAndNivelByMQTT(void)
190 {
191   String data = String("field1=" + String(consumption));
192   data = String(data + String("&field2=" + String(flowRate)));
193   data = String(data + String("&field3=" + String(nivel)));
194   int length = data.length();
195   char msgBuffer[length];
196   data.toCharArray(msgBuffer, length+1);
197   String topic = String("channels/" + String(channelID));
198   topic = String(topic + String("/publish/" + String(writeAPIKey)));
199   length = topic.length();
200   char topicBuffer[length];
201   topic.toCharArray(topicBuffer, length+1);
202   mqttClient.publish(topicBuffer, msgBuffer); // Publica a vazão e o
      ↪  consumo no tópico "topicBuffer"
203   lastSendTime = millis();
204 }
205
206 // Publica a vazão, consumo e nível na porta serial
```

```
207
208 void sendFlowRateConsumptionAndNivelToSerial(void)
209 {
210   Serial.println();
211   Serial.print("Vazão: ");
212   Serial.print(flowRate);
213   Serial.println(" l/min");
214   Serial.print("Consumo: ");
215   Serial.print(consumption);
216   Serial.println(" litros");
217   Serial.print("Nível: ");
218   Serial.println(nivel);
219 }
220
221 // Habilita as interrupções
222
223 void turnOnPulsesCounter(void)
224 {
225   attachInterrupt(digitalPinToInterrupt(D2), pulsesCounter, RISING); //
      ↪ Habilita as interrupções associadas ao pino D2, sendo que a função
      ↪ "pulseCounter" deverá ser chamada sempre que houver um evento de
      ↪ borda de subida no pino D2
226 }
227
228 // Desabilita as interrupções
229
230 void turnOffPulsesCounter(void)
231 {
232   detachInterrupt(digitalPinToInterrupt(D2)); // Desabilita as
      ↪ interrupções associadas ao pino D2
233 }
234
235 void setup()
236 {
237   Serial.begin(115200); // Inicia a serial com a taxa de 115200 baud
      ↪ rate
238   delay(10);
239   WiFi.mode(WIFI_STA); // Configura o ESP como STATION, ou seja, cliente
240   setWiFiConnection();
```

```
241   setMQTTConnection();
242   resetPulsesCounter();
243   flowRate = 0.0;
244   consumption = 0.0;
245   nivel = 0.0;
246   pinMode(D2, INPUT); // Configura o pino D2 da NodeMCU como entrada
247   pinMode(D5, INPUT);// Configura o pino D5 da NodeMCU como entrada
248   pinMode(D6, INPUT);// Configura o pino D6 da NodeMCU como entrada
249   pinMode(D7, INPUT);// Configura o pino D7 da NodeMCU como entrada
250   pinMode(D8, INPUT// Configura o pino D8 da NodeMCU como entrada
251
252   turnOnPulsesCounter();
253   lastMillis = millis();
254   lastSendTime = lastMillis;
255 }
256
257 void loop()
258 {
259   checkWiFiAndMQTTconnnection(); // Verifica as conexões e MQTT
260
261   if ((millis() - lastMillis) >= 1000) { // Base de tempo para
      ↪ realização dos  cálculos de vazão e consumo (1 segundo)
262
263     if (flag == 1) {
264       mqttClient.unsubscribe(topicRead); // Cancela a assinatura ao
        ↪ tópico "topicRead"
265     }
266
267     turnOffPulsesCounter(); // Desabilita as interrupções
268     calculateFlowRateAndConsumption(); // Calcula a vazão e o fluxo
269     nivelcheck();
270
271     if ((millis() - lastSendTime) >= interval) { // Publica a vazão,
      ↪ consumo e nível a intervalos pré-determinados
272
273       sendFlowRateConsumptionAndNivelByMQTT();
274
275     }
276
```

```
277       sendFlowRateConsumptionAndNivelToSerial(); // Envia a vazão, consumo
          ↪  e nível para a  porta serial
278       lastMillis = millis(); // Atualiza o último valor que havia sido
          ↪  armazenado na variável "lastMillis"
279       resetPulsesCounter(); // Reseta a variável que armazena o número de
          ↪  interrupções
280       turnOnPulsesCounter(); // Habilita as interrupções
281
282     }
283
284     mqttClient.loop(); // Garante a manutenção da conexão ao broker e o
        ↪  recebimento de publicações
285
286   }
```

Certifico que o aluno **Gustavo Ferreira Rodrigues,** autor do trabalho de conclusão de curso intitulado "**Desenvolvimento de um sistema de automação predial para gestão e monitoramento em tempo real do sistema hidráulico**", efetuou as correções sugeridas pela banca examinadora e que estou de acordo com a versão final do trabalho.



Prof. Dr. Antonio Santos Sánchez
Orientador