UNIVERSIDADE FEDERAL DE GOIÁS
ESCOLA DE ENGENHARIA ELÉTRICA, MECÂNICA E COMPUTAÇÃO
GRADUAÇÃO EM ENGENHARIA ELÉTRICA

DIEGO SOARES DOS SANTOS

PROTÓTIPO DE SISTEMA DE CONTROLE E MONITORAMENTO PARA CHUVEIRO COM AQUECIMENTO SOLAR

GOIÂNIA
2019

sistema de bibliotecas ufg

EMC

UFG

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TRABALHO DE CONCLUSÃO DE CURSO DE GRADUAÇÃO NO REPOSITÓRIO INSTITUCIONAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio do Repositório Institucional (RI/UFG), regulamentado pela Resolução CEPEC nº 1204/2014, sem ressarcimento dos direitos autorais, de acordo com a Lei nº 9610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou *download*, a título de divulgação da produção científica brasileira, a partir desta data.

1. Identificação do Trabalho de Conclusão de Curso de Graduação (TCCG):

Nome completo do autor: Diego Soares dos santos

Título do trabalho: Protótipo de Sistema de Controle e Monitoramento para chuveiro com aquecimento Solar

2. Informações de acesso ao documento:

Concorda com a liberação total do documento [X] SIM [ ] NÃO[1]

Havendo concordância com a disponibilização eletrônica, torna-se imprescindível o envio do (s) arquivo (s) em formato digital PDF do TCCG.

Diego Soares dos Santos
(Nome completo do autor)[2]

Ciente e de acordo:

(Nome completo do orientador) [2]

Data: 05 / 08 / 2019

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. A extensão deste prazo suscita justificativa junto à coordenação do curso. Os dados do documento não serão disponibilizados durante o período de embargo.

Casos de embargo:

- Solicitação de registro de patente;
- Submissão de artigo em revista científica;
- Publicação como capítulo de livro;
- Publicação da dissertação/tese em livro.

[2] As assinaturas devem ser originais sendo assinadas no próprio documento. Imagens coladas não serão aceitas.

DIEGO SOARES DOS SANTOS

PROTÓTIPO DE SISTEMA DE CONTROLE E MONITORAMENTO PARA CHUVEIRO COM AQUECIMENTO SOLAR

Trabalho submetido à Universidade Federal de Goiás como parte dos requisitos necessários para obtenção do Grau de Bacharel em Engenharia Elétrica. Sob orientação do Professor Dr. José Wilson Nerys.

GOIÂNIA
2019

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFG.

dos Santos, Diego Soares
PROTÓTIPO DE SISTEMA DE CONTROLE E MONITORAMENTO PARA CHUVEIRO COM AQUECIMENTO SOLAR [manuscrito] / Diego Soares dos Santos. - 2019.
80 f.: il.

Orientador: Prof. Dr. José Wilson Nerys .
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Goiás, Escola de Engenharia Elétrica, Mecânica e de Computação (EMC), Engenharia Elétrica, Goiânia, 2019.
Bibliografia. Apêndice.
Inclui siglas, fotografias, abreviaturas, algoritmos, lista de figuras.

1. Controle de temperatura. 2. Misturador eletrônico. 3. Economia. 4. Interface. 5. Monitoramento. I. , José Wilson Nerys, orient. II. Título.

CDU 621.3

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE GOIÁS
ESCOLA DE ENGENHARIA ELÉTRICA, MECÂNICA E DE COMPUTAÇÃO
COORDENAÇÃO DE ESTÁGIO E PROJETO FINAL

ATA DE AVALIAÇÃO DE PROJETO FINAL

CURSO

(X) Eng. Elétrica ( ) Eng. Mecânica ( ) Eng. de Computação ( ) Projeto Final 1 ( ) Projeto Final II

AVALIAÇÃO DE PROJETO FINAL

Título do projeto: PROTÓTIPO DE SISTEMA DE CONTROLE E MONITORAMENTO PARA CHUVEIRO COM AQUECIMENTO SOLAR

BANCA AVALIADORA

Membro 1: JOSÉ WILSON LIMA NERYS
Membro 2: LOURENÇO MATIAS
Membro 3: GUSTAVO SOUTO DE SÁ E SOUZA

ESTUDANTES

Matrícula	Nome
2016 09827	DIEGO SOARES DOS SANTOS

NOTAS

Matrícula	Membro 1				Membro 2				Membro 3				Média
	NPT	NTE	NAA	NF	NPT	NTE	NAA	NF	NPT	NTE	NAA	NF	
2016 09827	8,0	6,6	5,0	6,0	8,0	6,6	5,0	6,0	8,0	6,6	5,0	6,0	6,0

NPT – Nota plano de trabalho; NTE – Nota do trabalho escrito; NAA – Nota de apresentação e arguição
Para Eng. Elétrica, Mecânica e PFC2 da Eng. Da Computação: NF = 0,1 x NPT + 0,45xNTE + 0,45 x NAA
Para PFC1 da Eng. Da Computação: NF = 0,3 x NPT + 0,7 x NAA

Goiânia, 05 de AGOSTO de 2019.

Membro 1

Membro 2

Membro 3

DEDICATÓRIA

Dedico este trabalho à minha família e amigos que sempre estiveram presentes nos momentos complicados e me fortaleceram com sua presença e auxílio.

RESUMO

Este trabalho teve por objetivo estudar e desenvolver um protótipo de equipamento destinado ao controle da temperatura de água por meio de sistemas com aquecimento solar, em que o sistema inclui, além do controle de temperatura, o monitoramento via *web* das grandezas envolvidas, tais como: o consumo de água e o consumo de energia elétrica usado para aquecer a água até que o processo de aquecimento seja realizado por parte do controle. Para o projeto, foram levadas em consideração: alta capacidade de processamento pelo fato das variáveis envolvidas estarem em constante análise, baixo custo de montagem e fabricação e capacidade de se conectar à rede global de internet a fim de que os dados monitorados sejam apresentados ao usuário por meio de uma interface gráfica. Este trabalho reforça a ideia de controle e automação como as principais áreas da engenharia que estarão em contínuo progresso, com suas tecnologias e o modo como estas se relacionam sendo aprimorado.

Palavras-chave: Controle de temperatura; Aquecimento solar; Misturador eletrônico; Interface; Monitoramento; Economia.

ABSTRACT

The objective of this work was to study and develop a prototype of equipment for the control of water temperature through solar heating systems, in which the system includes, besides temperature control, web monitoring of the quantities involved, such as: the consumption of water and the consumption of electric energy used to heat the water until the heating process is performed by the control. For the project, the following were considered: high processing capacity because the variables involved are constantly under analysis, low assembly and manufacturing costs and the capacity to connect to the global internet network so that the monitored data is presented to the user by through a graphical interface.This work is in the development and automation in the progress as technology, and the mode as in the progress them, with its technologies and the mode as in the progress as a technology, in the progress as a technologies.

Keywords: Temperature control; Solar heating; Electronic mixer; Interface; Monitoring; Economy.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

A/D	Analógico/Digital
AC	Corrente Alternada
CPU	Central Processing Unit - Unidade Central de Processamento
DC	Corrente Contínua
I/O	Input/Output - Entrada/Saída
I²C	Inter-Integrated Circuit - Circuito Interno Integrado
MCU	Micro Controller Unit - Unidade de Micro Controlador
MRV	Máquina de Relutância Variável
mV/A	Milivolts por Ampere
PWM	Pulse Witdth Modulation - Modulação por Largura de Pulso
RF	Rádio Frequência
SPI	Serial Peripheral Interface - Periférico de Interface Serial
TTL	Transistor-Transistor Logic - Lógica transistor-transistor
USART	Universal Asynchronous Receiver Transmitter - Transmissor Receptor Assíncrono Universal
V	Volt
VAC	Tensão em corrente alternada
WiFi	Wireless Fidelity - Rede sem fio
μA	Micro Ampere

SUMÁRIO

1 INTRODUÇÃO

Ao longo da evolução da espécie o ser humano esteve sempre buscando conforto e comodidade para seus momentos pessoais e, associado a isso, usufruir de seus bens de consumo de modo eficiente, acarretando na busca incessante pelo aprimoramento das técnicas de produção, controle e medida dos gastos associados a tais necessidades. Diante disso, os métodos de engenharia se encontram cada vez mais elaborados no que tange à aplicação de dispositivos tecnológicos aprimoradas e capazes de oferecer resultados de excelência, tanto para aqueles que projetam tais dispositivos, quanto para aqueles que irão desfrutar da praticidade e eficiência que seu uso proporciona.

Desta forma, a ideia da Internet das Coisas (do inglês, IoT, *Intertnet of Things*) associada ao uso de dispositivos microcontroladores gera uma ferramenta poderosa de associação de evolução e praticidade a fim de serem obtidos resultados eficientes e controláveis em processos de automação industrial, comercial e até mesmo residencial.

1.1 Problema

Com o aumento gradativo das tarifas relacionadas aos bens de consumo como energia e água, faz-se crescer nas pessoas um sentimento de indignação e de necessidade em aprimorar os métodos de utilização de tais recursos. Estabelecimentos e residências que consomem em quantidades significativas ficam interessadas em produtos capazes de acarretar resultados idênticos e, em acréscimo, garantir uma economia relacionada aos custos envolvidos tanto a longo quanto a curto prazo.

Sendo assim, hotéis e residências que utilizam sistemas de aquecimento de água por meio da energia proveniente do sol já estão adiantados em relação à maioria dos outros cidadãos que ainda se encontram à mercê da dependência de dispositivos que se consomem essencialmente energia elétrica para produzir os mesmos resultados e ainda sem ter um conhecimento do quanto se está gastando em ambos os produtos elucidados.

Diante disso, faz-se necessário associar as tecnologias de automação disponíveis a fim se obter benefícios relacionados à economia financeira. Contudo, realizar apenas o controle não se torna tão eficiente quanto em outros tempos. Atualmente, é preciso saber, em tempo real, o

que ocorre durante tais processos e diante disso tomar decisões elaboradas a fim de garantir a eficácia de tais processos.

Desta maneira, a utilização da Internet das Coisas associada ao controle desses recursos por meio de automação se mostra uma solução plausível a curto e a longo prazo, uma vez que está ferramenta faz com que cada dia nos tornemos mais comprometidos com dispositivos eletrônicos capazes de nos manter informados e, por consequência, um recurso poderoso para gerenciamento dos gastos envolvendo o consumo de água e energia elétrica.

1.2 Objetivos

1.2.1 Objetivo geral

O projeto tem por objetivo o desenvolvimento de um protótipo capaz de realizar o controle de temperatura de água de sistemas com aquecimento solar. Nele, devem ser incluídos o controle de temperatura, o monitoramento via *web* de grandezas tais como consumo de água e consumo de energia elétrica usada para o aquecimento inicial desta água até a chegada de uma mistura aquecida pelo sol.

1.2.2 Objetivos específicos

Este trabalho apresenta uma interdependência de seus componentes, uma vez que se trata de um processo controle associado à transmissão de dados. Sendo assim, um processo gerando dados com erros poderá acarretar na total inutilização do mesmo, ainda que uma outra parte se encontre em perfeita codificação. Desta forma, os objetivos específicos se inter-relacionam a fim de se chegar ao resultado pretendido.

Estes objetivos são:

- Criar rotinas de códigos capazes de realizar a leitura e a interpretação de dados analógicos inseridos pelas portas correspondentes.

- Obter, por meio dessas rotinas, dados de temperatura da água aquecida, da água que está à temperatura ambiente, do sensor de corrente elétrica e do potenciômetro utilizado como chave de seleção.
- Criar uma rotina de controle que associe os dados de entrada e as leituras de temperatura e então execute a correção da temperatura da água ao valor pretendido.
- Criar uma rotina capaz de comunicar o dispositivo microcontrolador à rede de internet local e em seguida enviar os dados gerados à um servidor *web* para que tais dados sejam utilizados por um aplicativo desenvolvido para apresentar as informações de consumo de água e energia elétrica durante os banhos do dia.

1.2.3 Justificativa

A Internet das Coisas é usada para conectar objetos físicos entre si, estabelecendo uma comunicação através de processos inteligentes e softwares que transmitem informações para uma rede.

Diversas aplicações utilizando o princípio da IoT estão sendo desenvolvidas diariamente, permitindo a difusão de tais sistemas, permitindo sua implementação em diversas áreas da automação, tanto industrial com residencial.

Pelo exposto, verifica-se a aplicabilidade da realização deste projeto.

2 REVISÃO BIBLIOGRÁFICA

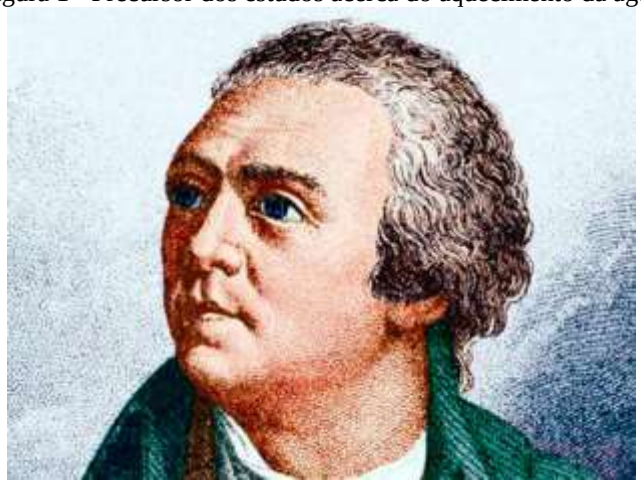
2.1 Um breve histórico sobre o aquecedor solar

O suíço Horace de Saussure (Figura 1) foi o primeiro cientista a documentar experiências que tentavam compreender a capacidade do sol em aquecer a água em 1767. Para isto, o mesmo executou várias experiências com uma caixa revestida por um isolante térmico. Posteriormente, o americano Clarence Kemp criou um aquecedor feito com tanques de cobre alocados no interior de uma caixa de madeira com isolante térmico e vidro na cobertura, patenteando-o. Porém, este sistema perdia calor a noite, limitando sua eficácia.

Contudo, foi o americano William Bailey quem conseguiu realizar este feito, pateando um aquecedor solar parecido com os modelos utilizados na atualidade.

Estes realizam a conversão da radiação eletromagnética proveniente do sol em energia térmica por meios coletores (ou painéis) solares, conhecidos como coletores de Bailey, feitos de cobre ou alumínio, em formato de aletas, e pintadas numa cor escura para maior absorção da radiação solar, proporcionando uma captação mais eficiente da radiação. Assim, o calor gerado é absorvido pelo líquido presente no interior destes painéis e que em seguida é transportada por bombeamento através de tubos isolados até chegar ao depósito de água quente (reservatório térmico ou boiler), como os mostrados na Figura 2.

Figura 1 - Precursor dos estudos acerca do aquecimento da água

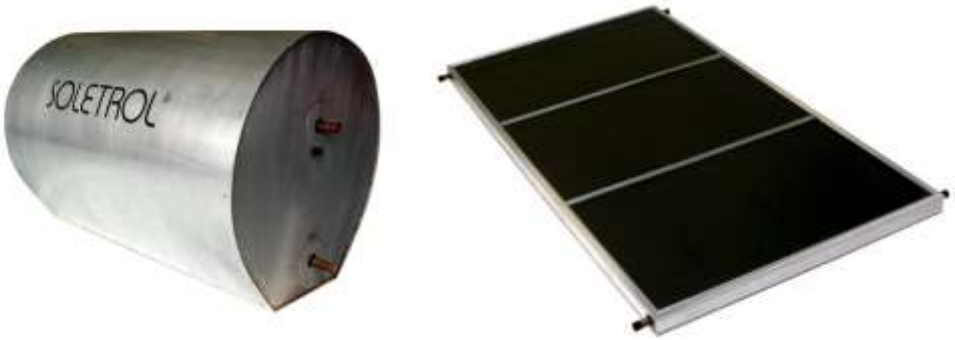


Fonte: https://www.soletrol.com.br/extras/historia-do-aquecedor-solar

Esses sistemas apresentam duas variações sendo elas diferenciadas pela forma como a água circula por ele, sendo (a) sistemas com circulação em termossifão e (b) sistema com circulação forçada. Naquele a circulação é promovida pela utilização da termodinâmica e da

força gravitacional, que fazem com a água quente suba naturalmente para o reservatório e que a água fria desça para os painéis dispensando o uso de bombas elétricas, sendo, portanto, mais econômicos e simples de instalar. Já na circulação forçada, é possível posicionar o tanque em locais separados dos painéis, portanto podendo ser colocado ao nível do chão ou em qualquer localidade da residência (na configuração termossifão deve estar posicionado acima dos destes).

Figura 2 - Reservatório térmico e coletor solar



Fonte: https://www.soletrol.com.br/extras/historia-do-aquecedor-solar/

2.2 Dispositivos utilizados no protótipo

2.2.1 Microcontroladores

Um computador é composto por um microprocessador, memória e periféricos. Este é capaz de efetuar o processamento de informações inseridas por um usuário via dispositivos de entrada (mouse e teclado, por exemplo) e em seguida devolver o resultado ao mesmo por meio de um dispositivo de saída, em geral monitores de vídeo. Assim, o que um computador faz com tais informações depende do programa armazenado em sua memória.

Microcontroladores são dispositivos que também possuem microprocessador, memória e periféricos, contudo tudo pertencente a um mesmo encapsulamento, o que faz este ser chamado comumente de computador de um único chip (SOUZA,2007).

Começaram a ser fabricados a partir da década de 1980, com destaque ao microcontrolador 8051 fabricado pela Intel Corporation®, e à medida em que as tecnologias evoluíram foram ficando cada vez mais robustos e com maior capacidade de processamento recebendo dispositivos de I/O integrados a recursos presentes nos chips, tais como: conversor A/D, PWM, oscilador interno, SPI™, I²C™, USART, módulos RF, etc. (ZANCO,2005).

Figura 3 - Modelos de Microcontroladores disponíveis no mercado

Fonte: Introduction to Microcontrollers (2018)

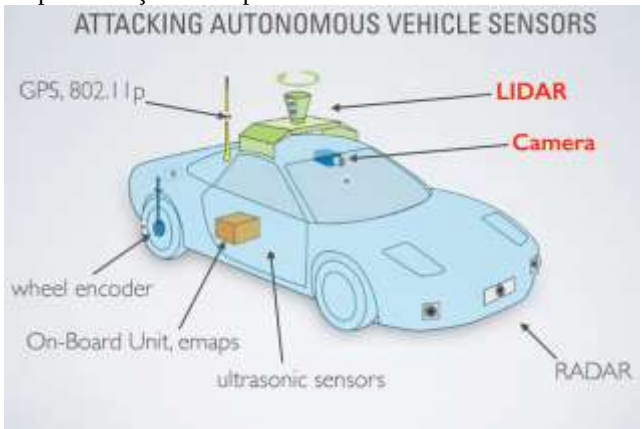
2.2.1.1 Aplicações de um microcontrolador

Sistemas microcontrolados estão inseridos em diversas áreas de produção, se destacando a automação industrial e comercial, residencial e predial, nas áreas automobilística, produtos manufaturados, eletrodomésticos, telecomunicações, agrícola, etc. (NETO, 2012).

É estimado que 63 milhões de veículos sejam fabricados anualmente, sendo que cada um deles possuem cerca de 30 microcontroladores para controle de funções básicas, chegando aplicação de 70 dispositivos em modelos mais completos, agregando conforto, segurança e eficiência ao controle de freios, direção eletrônica, controle de suspensão e injeção eletrônica de combustível, travas elétricas, etc. (CORTELLETI,2006).

A indústria de eletrodomésticos também se destaca na implementação de equipamentos com funcionalidades adicionais atribuídas ao uso de sistemas microcontrolados que conferem otimização na execução e trazem eficiência no dia a dia dos usuários (NETO, 2012).

Figura 4 - Implementação de dispositivos controlados em um veículo autônomo



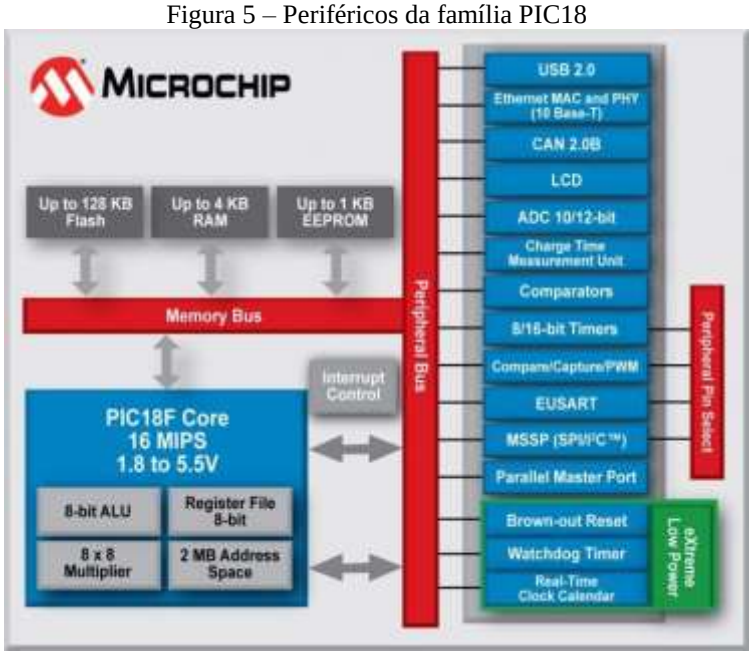
Fonte: Jonathan Petit,2015

2.2.1.2 Microcontroladores da família PIC®

Os microcontroladores PIC são fabricados pela empresa Microchip Tecnology e se utilizam da arquitetura RISC que permite que estes tenham um set de instruções reduzidos e chegam a possuir 40 MHz de frequência. São separados em três grupos, 12, 14 e 16 bits, que se diferenciam pela capacidade de armazenamento em cada localidade da memória (ZANCO, 2005).

O conjunto de instruções RISC fornecem ao PIC um caminho de migração de 6 para 80 pinos e de 384 bytes para 128 kbytes de memória de programa através da cominação de características RISC com uma arquitetura modificada do barramento duplo de Harvard (Microchip, 2006).

Figura 5 – Periféricos da família PIC18



Fonte: PIC18 e seu sistema de clock, 2005

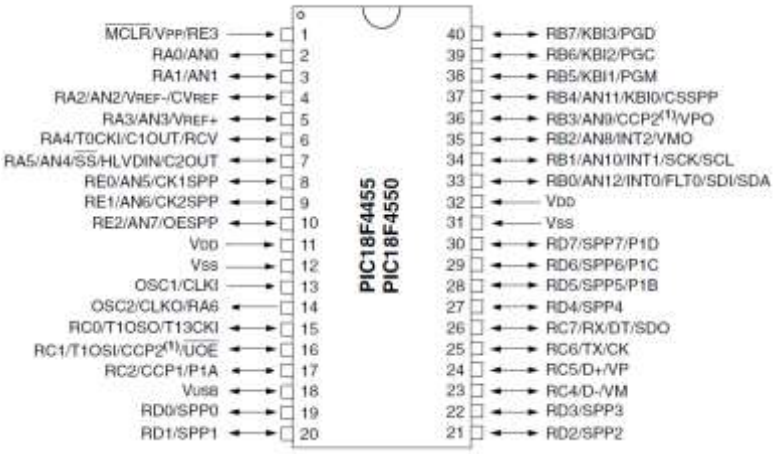
2.2.1.3 Microcontrolador PIC18F4550

Modelo da família 18F possuindo 40 pinos, tem por características principais alta performance; enhanced flash, permite ao usuário resultados mais confiáveis já que executa de 10 a 100 vezes mais operações de read/write, menos propenso a falhas e com menos corrente

operante, apresentando uma relação voltagem /temperatura melhorada (*nanoWatt Technology* – Tecnologia nanoWatt).

Este controlador fornece ao projeto pinos suficientes para implementação de todos os dispositivos utilizados, garantindo sua realização.

Figura 6 - Diagrama de pinos do 18F4550



Fonte: Dasheet PIC18F2455/2550/4455/4550, 2006

2.2.1.4 Programando um microcontrolador PIC

Para que o microcontrolador execute instruções estas devem ser escritas e então alocadas em alguma unidade de armazenamento para que este as leia, interprete e processe. Em um PIC, essas instruções são armazenadas em uma memória do tipo não volátil, ou seja, mesmo que a alimentação seja desligada, elas permanecem armazenadas e intactas para ser executadas novamente quando o sistema foi religado. Desta forma esta memória não serve para salvar dados, apenas instruções.

Os primeiros dispositivos programáveis tinham seus programas escritos em códigos conhecidos como códigos de máquina. Estes consistem em códigos binários inseridos por dispositivos de entrada de dados, principalmente por leitora de cartões, e em seguida são executadas pela máquina (PEREIRA, 2003).

Por serem códigos de difícil programação, fez surgir uma nova forma de programar sistemas, sendo esta chamada de *Assembly*. Esta linguagem consiste em uma forma de representação dos códigos utilizando mnemônicos, ou seja, abreviações de termos usuais que descrevem a operação efetuada pelo comando em código de máquina. Já a conversão desses mnemônicos em códigos binários executáveis pela máquina é realizada por um programa

chamado *Assembler* (montador). Esta linguagem é tida como linguagem de baixo nível, sendo assim não possuem nenhum comando, função ou instrução além das que estão definidas no conjunto de instruções pertenceste ao microcontrolador utilizado. Desta forma, o trabalho do programador se torna custoso e faz com este tenha um grande conhecimento acerca do hardware do dispositivo (PEREIRA, 2003).

Ocorre que, para proporcionar uma escrita com menos dificuldade, são utilizadas as linguagens de alto nível que utilizam comandos de mesmo nome e que posteriormente são convertidos para linguagem de baixo do processador utilizado. Uma linguagem bastante utilizada na codificação é a linguagem C, criada em 1972, por Dennis Ritchie, consistindo em uma linguagem de nível intermediário entre o Assembly e as linguagens de alto nível. Sua implementação é poderosa, que foi escolhida para desenvolver sistema operacionais como UNIX, e até mesmo o WINDOWS® e o LINUX® (PEREIRA, 2003).

Um exemplo de utilização de linguagem Assembly para inicializar uma saída na PORTA do 18F4550 é dado abaixo.

```
CLRF PORTA      ; Initialize PORTA by
                ; clearing output
                ; data latches
CLRF LATA       ; Alternate method
                ; to clear output
                ; data latches
MOVLW 0Fh       ; Configure A/D
MOVWF ADCON1    ; for digital inputs
MOVLW 07h       ; Configure comparators
MOVWF CMCON     ; for digital input
MOVLW 0CFh      ; Value used to
                ; initialize data
                ; direction
MOVWF TRISA     ; Set RA<3:0> as inputs
                ; RA<5:4> as outputs
```

Já em linguagem C utiliza-se o seguinte código:

```
set_tris_a(0xF0);      //Portas RA<3:0> como entradas e RA<5:4> como saídas
setup_adc_ports(AN0_TO_AN4|VSS_VDD); //Define RA<5:4> como portas analógicas
```

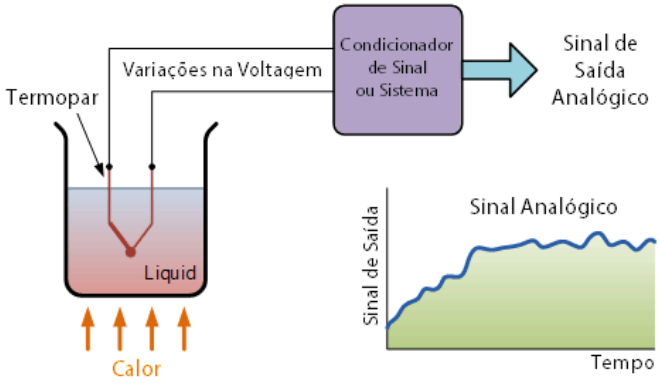
Contudo, vale ressaltar que a melhor linguagem para realizar a programação é aquela em que o programador se sente confortável para produzir seus códigos com eficiência. No caso

deste trabalho foi utilizada a implementação por meio da linguagem C, pois esta exige apenas o conhecimento prévio do comportamento dos periféricos do PIC, já que o compilador se responsabiliza pela tradução das funções do código para linguagem de máquina.

2.2.2 Sensores analógicos

Sensores analógicos, também conhecidos como transdutores, são capazes de detectar diferentes formas de energia produzindo um sinal de tensão contínuo e proporcional à quantidade a ser medida, tais como velocidade, radiação, energia térmica e também magnética.

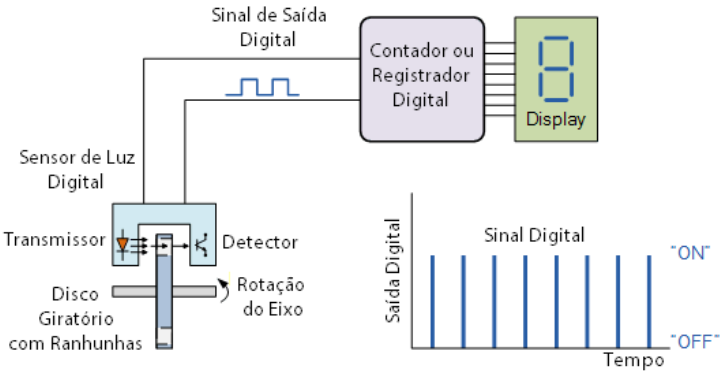
Figura 7 – Termopar utilizado para medição de temperatura em líquidos



Fonte: https://www.electronics-tutorials.ws/io/io_1.html

Diferem dos sensores digitas pela maneira em que a conversão do sinal a ser medido é analisada, enquanto aqueles respondem produzindo sinais contínuos no tempo, estes produzem um sinal de saída binário mostrados em nível lógico 1 ou 0 (ligado ou desligado), o que é conhecido no meio técnico como discretização do sinal.

Figura 8 - Sensor digital utilizado para medir velocidade de rotação de um eixo em rotação



Fonte: https://www.electronics-tutorials.ws/io/io_1.html

2.2.2.1 Sensor de temperatura LM35DZ

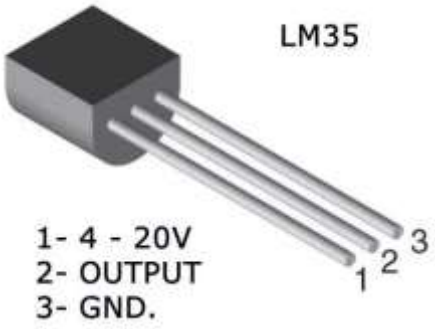
Temperatura é tida como a grandeza que caracteriza o estado térmico das moléculas de um corpo ou sistema. Desta forma, aumentando o estado de agitação das mesmas, se tem, por consequência, a elevação da temperatura e este sistema será chamado de quente, enquanto que a ação inversa, de diminuir tal agitação, faz com que seu estado térmico apresente temperaturas mais baixas e vai diminuindo a medida que as moléculas ficam cada vez mais estáticas.

A medição da temperatura pode ser feita de muitas formas desde a utilização de termômetros convencionais à aplicação de sensores térmicos como termopares, termistores, etc. Sendo assim, o monitoramento de equipamentos elétricos, máquinas e ambientes como interior de prédios acaba por ser facilitado diante da generalidade de equipamentos que podem atender essas necessidades. Um deles é o sensor analógico de temperatura LM35.

Os sensores desta série não necessitam de nenhuma calibração externa ou ajuste para prover medidas de temperatura na faixa de típica de ± ¼ ºC à temperatura ambiente e ± ¾ °C ao medir temperaturas em uma faixa de -55 a 150 ºC.

São dispositivos feitos de circuito integrado de saída em baixa voltagem e linearmente proporcional à unidade de temperatura em Celsius. Estes dispositivos apresentam uma vantagem sobre sensores de temperatura calibrados em Kelvin pelo fato os cálculos de conversão para a unidade de medida que seus usuários necessitam, uma vez que esta tem frequência de utilização em ambientes científicos (TEXAS INSTRUMENTS, 2015).

Figura 9 - Circuito integrado LM35DZ e seus pinos de alimentação



Fonte: easytronics.com.br

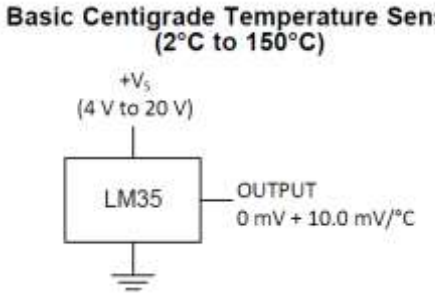
Possui uma saída de baixa impedância associada à linearização do sinal de saída, fazendo com que o LM35 apresente um bom controle de leitura, proporcionando ao usuário medições rápidas e precisas.

É um dispositivo que pode ser alimentado diretamente e em uma tensão de 5V, drenando correntes baixíssimas na ordem de 60 μA, apresentando um aquecimento de operação, cerca de 0,1 ºC ou menos, mesmo ao ar livre. Vê-se que este dispositivo possui uma grande eficiência para atender às necessidades operação deste projeto.

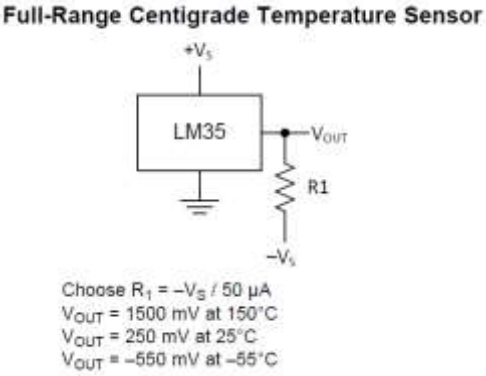
Os dispositivos da série LM35 estão disponibilizados no mercado na forma de circuitos integrados encapsulados hermeticamente em pacotes TO para transistores, enquanto que os LM35C, LM35CA e LM35D em transistor plástico TO-92.

É importante dizer que o LM35 apresenta dois tipos de configuração, sendo uma para mensurar valores apenas positivos e outra para utilizar a faixa de valores negativos. Para isto é necessário a instalação de um resistor conectado à saída de temperatura e este sendo alimentado por uma tensão $-V_S$ como detalhado no esboço a seguir.

Figura 10 - Configurações do sensor de temperatura LM35



(a) Medida básica de valores (variação de 2 a 150 ºC)



(b) Configuração para medidas de temperatura negativas e positivas
Fonte: TEXAS INSTRUMENTS, 2015

2.2.2.2 Sensor de corrente ACS712-30A

O ACS712 é um dispositivo produzido pela Allegro MycroSystems INC. capaz de proporcionar soluções precisas em medições de correntes elétricas tanto em AC quanto em DC,

sendo utilizado em aplicações industriais, comerciais e até mesmo em sistemas de comunicação. É uma família de instrumentos de medição capaz de medir correntes de 5A, 20A e 30A (modelos ACS712ELCTR-XXA-T). Por ser projetado e implementado em uma estrutura compacta, acaba sendo um componente eletrônico de fácil utilização. Suas principais aplicações são voltadas ao controle de motores, detecção e gerenciamento de cargas, a fontes de alimentação em modo comutado e proteção para faltas de sobrecorrente.

Figura 11 - Sensor de efeito Hall ACS712-30A

Fonte: Ponto da Eletrônica, 2019

Consiste em um sensor linear de efeito Hall com baixo offset na medida, possuindo um condutor de cobre localizado próximo à superfície do seu substrato. No momento em que ocorre o fluxo de corrente através do condutor de cobre é gerado um campo magnético e este é percebido pelo circuito integrado de efeito Hall que em seguida converte, para sua saída, uma voltagem proporcional a corrente medida.

Sua eficiência está relacionada ao fato de que este sinal magnético se encontra razoavelmente próximo ao transdutor de efeito Hall, e assim, a tensão de saída gerada pelo CI BicMOS Hall é convertida de forma precisa e proporcional ao seu baixo offset.

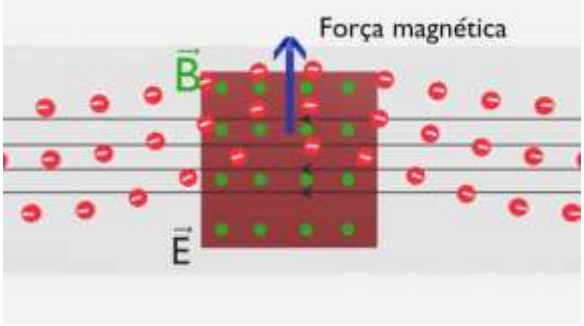
Os terminais para passagem da corrente a ser medida são eletricamente isolados dos componentes de sensoriamento, o que permitindo que ele seja utilizado em aplicações que exigem isolamento sem ter de ser utilizados optoisoladores ou qualquer outro típico de técnica dispendiosa (ACS712 DATASHEET,2007).

A característica do efeito Hall está no aparecimento de um campo elétrico transversal em um condutor que, percorrido por uma corrente elétrica, é mergulhado um campo magnético. Os sensores, como este, que utilizam deste efeito, constituídos por dispositivos semicondutores que sofrem a influência de um campo magnético.

Quando o dispositivo é alimentado por uma fonte de carga, por meio dele ocorre a passagem de uma corrente elétrica. Em seguida, é adicionado um campo magnético perpendicular ao movimento desta corrente, fazendo com que seus elétrons em movimento

sofram a ação de uma força de Lorentz que faz com os mesmos não se movimentem mais de forma retilínea, mas se concentrando na direção da força magnética aplicada.

Figura 12 - Concentração dos elétrons livres na parte superior do material semicondutor



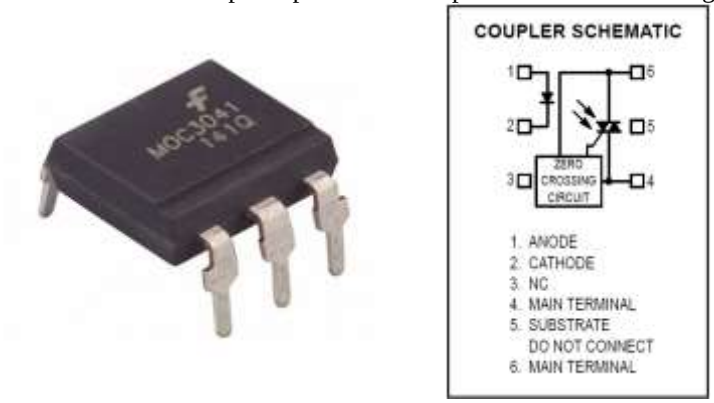
Fonte: Vida de Silício, 2018

Por fim, tal concentração faz surgir um potencial negativo na região em que existe a falta de elétrons, o que consequentemente gera uma diferença de potencial entre as extremidades do material que é proporcional ao valor da corrente que está passando por ele e, desta forma, se torna possível o cálculo da corrente elétrica que gera o campo magnético capaz de fazer com haja o surgimento desta diferença de potencial.

2.2.3 Optoacoplador MOC3041M

O Optoacoplador MOC3041 consiste em um diodo emissor de infravermelho de arseneto de gálio opticamente acoplados a um detector monolítico que desempenha a função de um TRIAC bilateral de cruzamento de tensão zero. Foram inicialmente projetados para uso com um TRIAC na interface de sistemas lógicos para equipamentos alimentados em linhas a partir de 115 VAC, como reles de estado solido, controles, solenoides, motores e aparelhos de consumo (MOTOROLA SEMICONDUCTOR TECHNICAL DATA, 1995).

Figura 13 - Optoacoplador MOC3041M de 6 pinos para tensão de pico de 400 V e seu diagrama esquemático

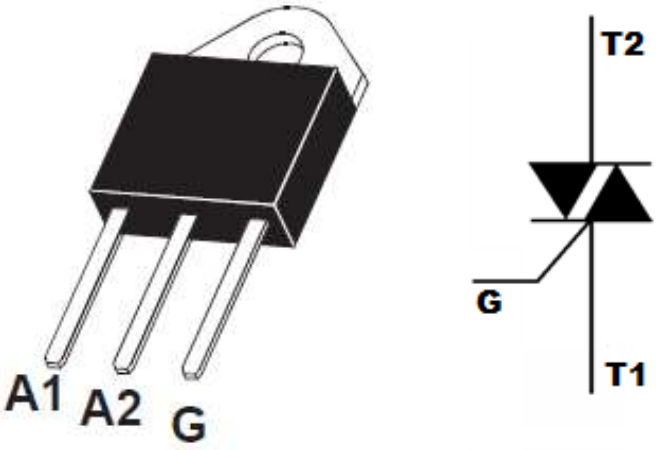


Fonte: Baú da Eletrônica, 2019

2.2.4 TRIAC BTA26- 600B

A série de dispositivos TRIAC BTA/BTB24-25-26 é adequada para comutação de uso geral em correntes alternadas. Podem ser utilizados com função LIGA/DESLIGA para serem aplicados como reles estáticos, regulação de aquecimento, aquecedores de água, em circuitos de partida de motores de indução, para operação de controle de fases de controladores de velocidade no motor de alta potência e também em circuitos de partida suave.

Figura 14 - Dispositivo TRIAC-BTA26 e seu diagrama esquemático



Fonte: BTA/BTB24, BTA25, BTA26 and T25 Series, 2002

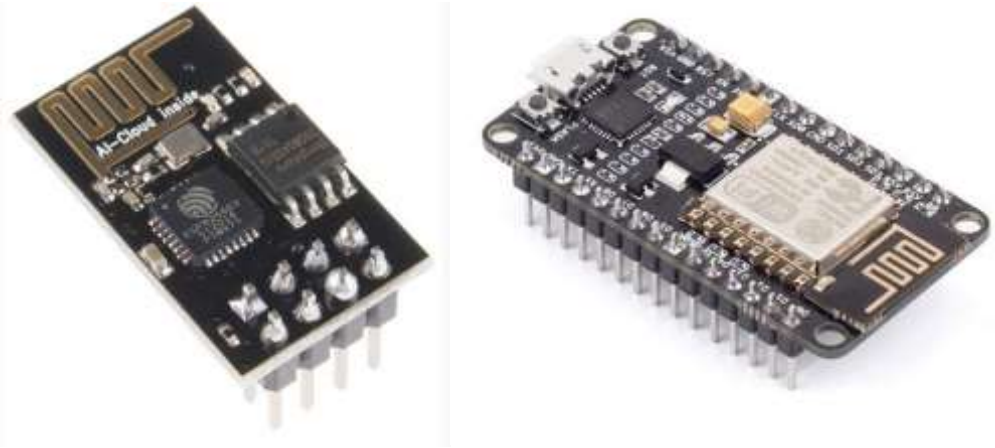
Para o uso em cargas indutivas são especialmente recomendas as versões sem suspensão devido aos seus altos desempenhos de comutação. Sendo elas a série BTA/BTBxxW e T25. Ao ser utilizada uma almofada de cerâmica interna

2.2.5 Módulo Wireless ESP8266-ESP01

Este dispositivo é um módulo que oferece uma solução para rede Wifi completa e autônoma, permitindo que o usuário hospede informações em si ou descarregue a mesma em outros servidores de aplicativos. Quando o ESP8266 armazena tais dados, os mesmos poderão ser inicializados diretamente de um flash externo, já que este módulo possui cache integrado para melhorar o desempenho do sistema e também minimizar os requisitos de memória.

Com stack TCP/IP integrado, tem como ponto forte o fato de sua placa ser integrada a um MCU, o que possibilita que códigos sejam embarcados na mesma permitindo que seus pinos possam ser controlados, funcionando sem um MCU externo.

Figura 15 - Módulo Wifi ESP8266-ESP32 e Wifi ESP8266 NodeMCU ESP-12



Fonte: BR-Ain Eletrogate,2019

Para este trabalho é utilizado o ESP-01. Este módulo conversa com o microcontrolador pela interface serial, possuindo dois pinos GPIO que permitem que o mesmo seja programado diretamente via adaptador.

As capacidades de processamento e armazenamento do ESP8266-ESP01 permitem que ele seja integrado a sensores e outros dispositivos específicos de aplicativos por meio das suas GPIOs, necessitando de um mínimo de desenvolvimento inicial e baixo carregamento durante o tempo de execução.

Possui alto grau de integração on-chip, incluindo o switch de antena balun, conversores de gerenciamento de energia, o que o faz requerer o mínimo do circuito externo, e outras soluções, incluindo o módulo front-end, projetado para ocupar uma área mínima de PCB (ESP8266 802.11BGN SMART DEVICE,2013).

Figura 16 - Módulo Wifi ESP8266-ESP01

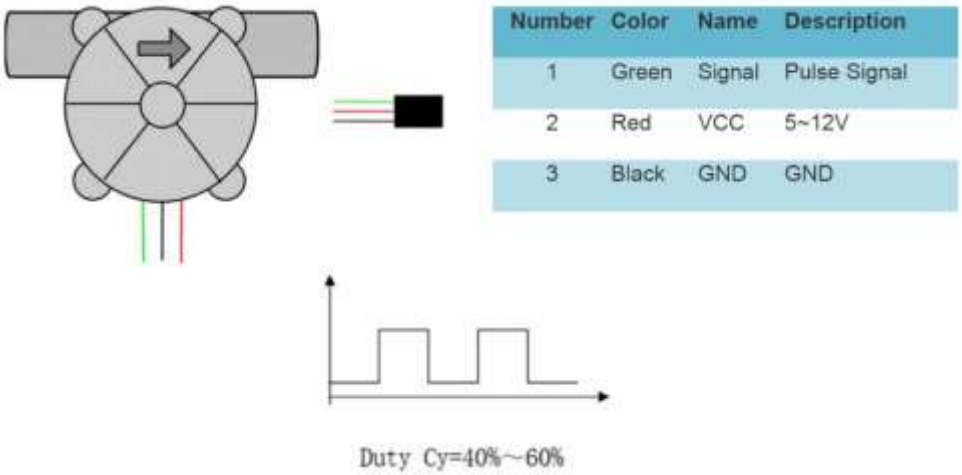
Fonte: BR-Ain Eletrogate,2019

2.2.6 Sensor de vazão YF-S201

O sensor YF-S201 consiste em uma válvula de plástico possuindo rotor de fluxo e sensor de efeito Hall. Em geral é instalado ao fim do sistema hidráulico para que desta forma o fluxo de água possa ser detectado.

No momento em que o líquido flui, seu rotor magnético irá girar e a velocidade de rotação irá variar de acordo com a taxa de fluxo fazendo com que seu sensor de efeito Hall emita um pulso de sinal PWM, permitindo que um microcontrolador possa tratar esses dados de acordo com as formas de cálculo informada no datasheet do ANEXO 1.

Figura 17 – Esboço de um dispositivo YF-S201 e seu sinal de saída PWM



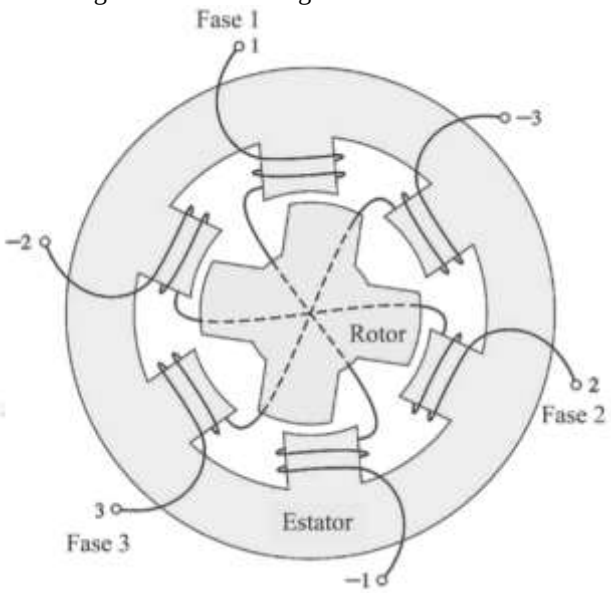
Fonte: Water Flow Sensor - 1/2", 2017

2.2.7 Motores de passo

Motores de passo são máquinas elétricas que possuem um estator com enrolamentos de excitação e um rotor magnético com saliências, sendo enquadrados na definição de máquina de relutância variável, ou MRV. Nestas máquinas não são necessários condutores em seu rotor, pois seu conjugado é produzido pela tendência do mesmo em se alinhar com a onda fluxo produzida pelo estator de modo a maximizar os fluxos concatenados que resultam da aplicação de uma dada corrente de estator.

Nessas máquinas, a produção de conjugado por ser analisada usando técnicas de conversão de eletromecânica de energia e pelo fato de que as indutâncias do enrolamento do estator estarem em função da posição angular do rotor.

Figura 18 - Fluxo magnético em uma MRV



Fonte: FITZGERALD, 2003

Contudo, apesar de ser um conceito há muito tempo difundido, as MRVs começaram a ter ampla aplicação em engenharia apenas recentemente, devido ao fato de apresentarem construção simples, porém são um tanto difíceis de ser controladas (FITZGERALD, 2003).

Ao se excitar as fases da MRV de modo sequencial, ocorre a rotação de seu rotor na forma de uma sequência de passos, girando um ângulo especifico a cada passo. Desta forma, motores de passo são projetados para tirar vantagem de tal característica, combinando uma geometria de relutância variável com ímãs permanentes, a fim de produzir aumentos de conjugado e precisão de posicionamento (FITZGERALD, 2003).

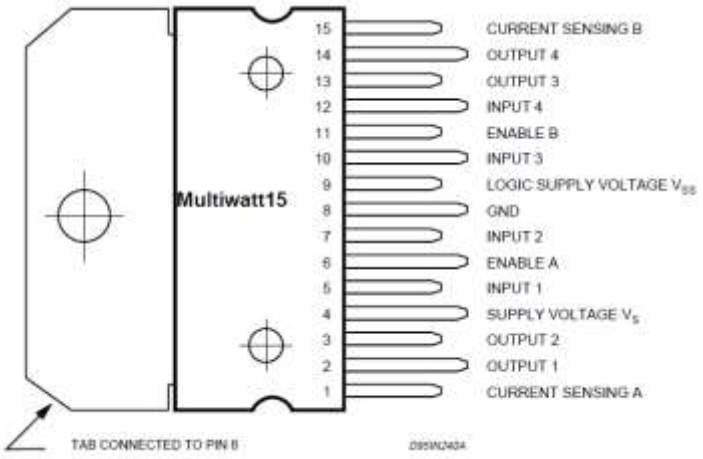
2.2.8 Driver de corrente L298

Este driver é um dispositivo monolítico integrado *multiwatt* de 15 pinos. É um driver de alta tensão, alta corrente de ponte dupla completa, projetado para aceitar níveis lógicos TTL (*Transitor-transistor logic*) padrão e unidade de cargas indutivas, tais como reles, solenoides, motores CC e motores de passo.

Nele, duas entradas *enable* são fornecidas para ativar ou desativar cada ponte do dispositivo independentemente. Os emissores dos transistores inferiores estão ligados entre si

e o terminal externo correspondente pode ser utilizado para a conexão de uma resistência de detecção externa.

Figura 19 - Driver de corrente L298 e seu diagrama esquemático



Fonte: STMICROELECTRONICS DATA, 2000

Uma entrada adicional é fornecida de modo realizar uma lógica em tensão baixa.

2.2.9 Display LCD-LM16016L

O dispositivo LCD utilizado no projeto é o RT162-7 de 16 caracteres (duas linhas de 16 caracteres) da fabricante Xiamen Amotec. Possui um total de 16 pinos, sendo dois deles utilizados para configurar a função de *backlight*.

Figura 20 - Display LCD 16x2

Fonte: Santy.cz, 2019

2.2.10 MIT App Inventor

O *App Inventor* é um software *web* criado pela universidade norte-americana Massachusetts Institute of Technology (MIT) capaz de desenvolver aplicativos Android utilizando o navegador *web* e um telefone celular ou emulador conectados. Por meio dele, o desenvolvedor cria aplicativos selecionando componentes e montando blocos que especificam como tais componentes devem se comportar.

A combinação de um ou mais comandos forma uma ação completa. E para facilitar a construção das ações, os comandos são estruturados como peças de quebra-cabeças. Apenas funções compatíveis se encaixam.

Toda a criação é feita visualmente, juntando peças com fosse um quebra-cabeças, tudo de forma interativa, pois é possível conectar o celular à plataforma por meio de um *QR Code* e assim, à medida que as peças estão sendo implementadas, o aplicativo é mostrado a fim de que seja possível testar o projeto.

Figura 21 - Blocos de lógica para desenvolvimento no App Inventor

Fonte: AndroidPro, 2018

Ao término do projeto, o desenvolvedor pode compilar todo o desenvolvimento e produzir um aplicativo Android executável para ser instalado em outros celulares.

3 METODOLOGIA

Este trabalho consiste no projeto e simulação de um sistema de controle de temperatura com microcontrolador associado à uma interface para desenvolvimento das grandezas envolvidas e posterior implementação para análise do comportamento predeterminado por programação em linguagem C.

O projeto será executado em etapas, inicialmente teóricas, passando para implementação em laboratório. As etapas previstas são:

- Etapa 1: Estudo de bibliografias sobre os temas: aquecimento solar com controle de temperatura, monitoramento de grandezas via *web*, transdutores de temperatura, vazão e corrente.
- Etapa 2: Desenvolvimento teórico/modelagem do sistema de controle de temperatura
    - Desenvolver códigos computacionais para o controle de vazão da água, sendo esta vazão determinada por um potenciômetro.
    - Desenvolver código para entrada do valor de temperatura.
    - Desenvolver códigos que recebam a leitura dos dados inseridos pelos sensores de temperatura LM35DZ e os converta para informações legíveis ao usuário.
    - Desenvolver um código para receber a leitura dos dados de corrente elétrica medidos pelo sensor ACS712-30A.
    - Desenvolver código que seja capaz de verificar quando deve ser armazenado os dados de corrente elétrica.
    - Desenvolver código para acionamento dos motores de vazão a fim de que estes sejam responsáveis pela regulagem da vazão predeterminada pelo usuário.
    - Desenvolver um código capaz de interpretar os dados gerador pelo sensor de vazão YF-S201 e associado ao seu uso, estabelecer o tempo de banho do usuário.
    - Com o tempo de banho do usuário determinar o consumo de energia que estaria sendo utilizado caso não houvesse controle.
    - Desenvolver código capaz de configura o módulo *Wifi* a se conectar com um roteador local e em seguida ser capaz de enviar os dados medidos para o aplicativo *web*.
    - Desenvolver um aplicativo *web* que receba as informações geradas pelo controle.
- Etapa 3: Simulação do sistema completo.

- Etapa 4: Testes do sistema em laboratório e análise de resultados
- Etapa 5: Escrita e apresentação de monografia.

3.1 Programação e gravação de um microcontrolador PIC

Para programação do microcontrolador PIC18F4550 será utilizado o *software* CCS C Compiler para dispositivos PIC12/14/16 bit desenvolvido para sistemas operacionais Windows. Para gravar, no chip, o código hexadecimal gerado é utilizado o Topmax II Expert Universal Device Programmer® for PC USB juntamente com o *software* gravador MaxLoader ®.

3.1.1 Compilador PCWH IDE

O CCS PCWH IDE Compiler for Microchip PIC, é um compilador com *suites* de ferramentas integradas para o desenvolvimento e depuração de aplicações embarcadas em execução no PIC Microchip® MCUs e dsPIC® DSCs. Esta *suite* inclui uma IDE para gerenciamento de projetos, um editor C sensível a contexto, ferramentas para desenvolvimento e depurador em tempo real, que ajudam os desenvolvedores a criar, analisar e depurar o código do projeto de documento (CCS MANUAL<, 2019).

Figura 22 - Interface do CCS Compiler

Fonte: CCS Inc, 2019

Seus principais recursos são:

- O PCB, o PCM e o PCH são compiladores separados. PCB é para opcodes de 12 bits, PCM é para opcodes de 14 bits e PCH é para microcontroladores opcode PIC® de 16 bits.
- Esses compiladores são projetados especificamente para atender às necessidades exclusivas do microcontrolador PIC® permitindo que os desenvolvedores criem aplicativos de software rapidamente e em uma linguagem mais legível e de alto nível.
- Compiladores IDE (PCW, PCWH e PCWHD) possuem o exclusivo ambiente de desenvolvimento integrado C Aware para compilar, analisar e depurar em tempo real.

A eficácia deste software está no conjunto de ferramentas de desenvolvimento se deve ao código CCS inteligente otimizando para compilador C, o que libera os desenvolvedores para concentrar-se na funcionalidade do projeto, em vez de ter de se tornar um especialista em arquitetura de MCU.

3.1.2 Topmax II Expert® for PC USB e software MaxLoader

O Topmax II Expert Universal Device Programmer® for PC USB um programador de dispositivos universais de alta velocidade capaz de programar uma memória flash de 64 Mbits (AM29LV641) em 42 segundos, e associado a sua eficiência se tem o baixo custo relacionado à sua aquisição (TOPMAX II DATASHEET, 2018).

Figura 23 – Dispositivo TopMax II Expert Universal Device Programmer®

Fonte: Mouse Eletronics Brasil, 2019

Seus principais recursos são:

- Um FPGA on-board para comunicação extremamente rápida.

- Suporte para baixa tensão: 5, 3,3, 2,7, 1,8, 1,5 V de alimentação na programação.
- Detecta todos os locais dos pinos para contatos de pinos ruins ou danificados.
- Possui a tecla START externa permite o modo de programação de produção.
- Fonte de alimentação universal interna, 110-240 VAC (não é necessária nenhuma fonte de alimentação separada no país estrangeiro).
- A limitação de corrente protege o circuito de hardware de erros ou erros de operação incorretamente inseridos ou defeituosos.

MaxLoader realiza a programação de microcontroladores e dispositivos de memória de série facilmente, utilizando programadores EETools. Com ele é possível adicionar, apagar e modificar dados do chip, a verificação ou bloqueio de conteúdo, automatização de limpeza e carregamento de arquivos HEX, etc (E.

4 ELABORAÇÃO E SIMULAÇÃO DO PROGRAMA

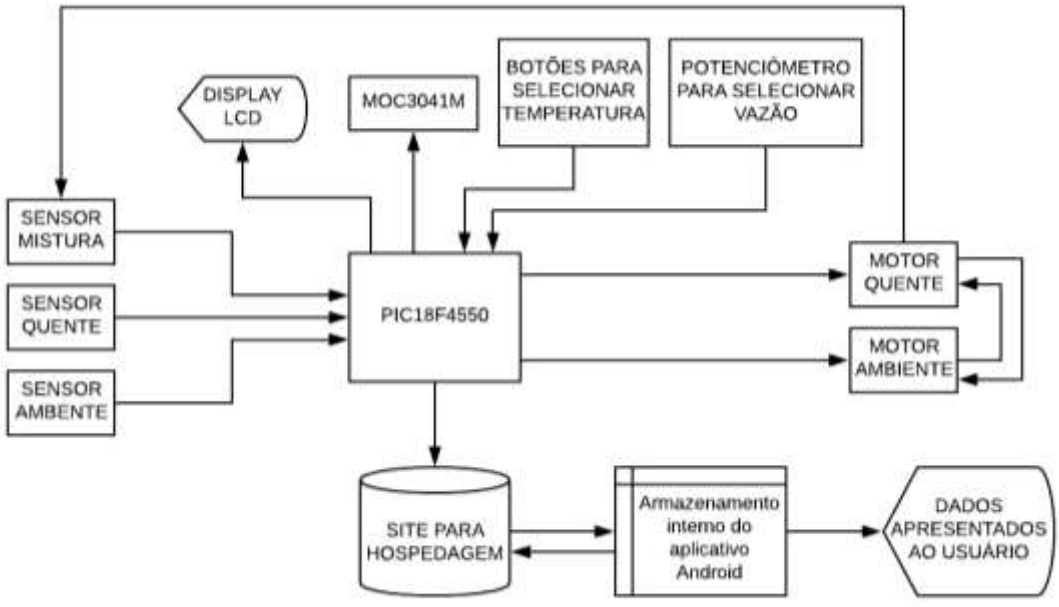
4.1 Fluxograma de funcionamento

O fluxograma apresentado representa o funcionamento desejado a este sistema (Figura 24).

A CPU receberá os valores de medida dos três sensores de temperatura, a temperatura desejada pelo usuário por meio de dois botões de escolha e um potenciômetro para determinação da vazão do sistema.

Em seguida, a CPU irá controlar os dois motores de passo e a abertura e/ou fechamento destes irá aumentar ou diminuir a vazão da água aquecida e da temperatura ambiente em uma razão cuja soma terá um total de 100%, ou seja, os motores não vão permitir que ocorra a abertura total e simultânea das válvulas, fazendo com que a relação de vazão de cada parte, ao serem somadas, não ultrapasse o valor determinado pelo usuário. Por conseguinte, de acordo com essas temperaturas, a CPU será capaz de determinar se irá enviar um sinal de acionamento para o MOC3041M, permitindo que ocorra o chaveamento do chuveiro até que o controle de temperatura alcance resultados satisfatórios.

Figura 24 – Fluxograma de operação dos dispositivos



Fonte: Próprio autor

Ao fim de cada banho, a CPU é programada para enviar os dados medidos para um site de hospedagem. Este site, estará em link com o aplicativo *web* que será responsável por analisar

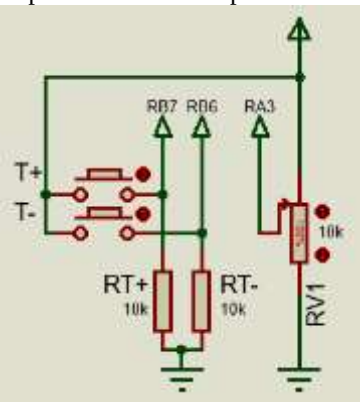
os novos dados enviados e então tratá-los para serem apresentados ao usuário por meio de gráfico de barra.

4.2 Rotinas de entradas para seleção

4.2.1 Código para escolha da temperatura

É utilizado o método de interrupção por entrada nos pinos B4 a B7 associado à interrupção pelo temporizador TIMER0 para realizar a seleção de temperatura por meio de botões. Para sua montagem é preciso a associação de um resistor de *pull-up* de 10 kΩ

Figura 25 – Botões de temperatura T+ e T- e potenciômetro para seleção de vazão



Fonte: Próprio autor

A mudança de temperatura ocorre cada vez que um dos botões T+ ou T- é pressionado, acionando a sua rotina de interrupção pelas portas B4 a B7. Nesta se encontra o comando para acionar a contagem do temporizador do TIMER0 e ao fim da contagem deste ocorre outra interrupção e nesta interrupção está o código para alteração do valor da temperatura.

4.2.2 Código para seleção de percentual de vazão

O código para leitura analógica do potenciômetro apresenta duas funções no projeto. A primeira delas é que por meio dele é escolhida a vazão percentual a ser aplicada ao sistema

hidráulico. Já a segunda, é de dar início à operação de controle por parte dos equipamentos utilizados no projeto.

Quando o potenciômetro for lido e em sua saída for obtido o ***valor de 10%*** de vazão, as outras rotinas de operação serão iniciadas e no instante que o usuário o movimentar para um valor inferior a este, o sistema será novamente fechado. Desta maneira, o potenciômetro tem a função de liga e desliga, funcionando como uma válvula de abertura e fechamento para o chuveiro.

Para obtenção de valores precisos de medição, foi utilizada a conversão A/D na capacidade máxima do microcontrolador, 10 bits, pois assim os dados lidos para uma alimentação de 5 V aplicada ao potenciômetro de 10 kΩ irão apresentar uma gama de valores.

Para a polarização deste dispositivo, foi decidido que o mesmo deve ser usado em sua resistência máxima, pois, deste modo, permite que ocorra a proteção do sistema para um caso em que o mesmo sofra um curto-circuito e assim será evitado que o chuveiro fique ligado até que o reparo seja feito.

O modo de instalação deste componente está exposto na Figura 24 e seu código de implantação pode ser consultado no APÊNDICE E.

4.3 Rotinas para leitura e interpretação das entradas de sensores

4.3.1 Código para sensores *de temperatura*

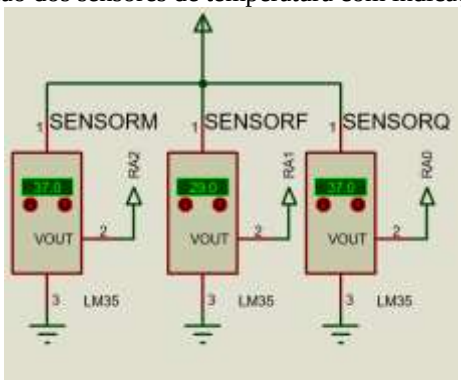
Este projeto trabalha com faixas de valores variando da temperatura ambiente (28 ºC, em média) à cerca de 80 ºC, que é quando a água se encontrará aquecida. Desta forma, a utilização da configuração disposta na Figura 10a é suficiente para atender às necessidades do mesmo.

Pelo fato do sensor fornecer em sua saída uma resposta analógica, esta deve ser interpretada pelo microcontrolador por meio do seu conversor A/D, o qual converte este sinal para uma faixa de valores entre 0 e 1023 (conversor A/D de 10 bits) e, assim, o programador transpõe essa medida o valor em ºC por meio da expressão

$$T = \left(5 * \frac{(Valor\ lido\ pelo\ microcontrolador)}{1023} * \frac{1}{0{,}01}\right)\ \ ^{\circ}C$$

com faixa de precisão de ±1 ºC.

Figura 26 – Esquema de ligação dos sensores de temperatura com indicação de suas portas de conexão



Fonte: Próprio autor

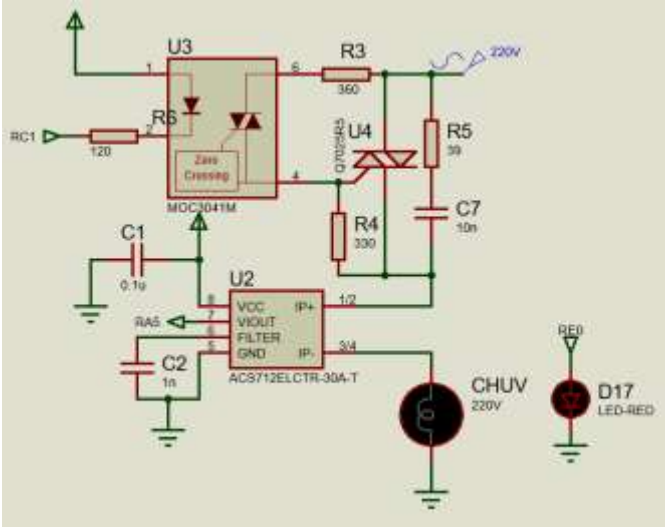
Os detalhes da programação do leitura dos sensores podem ser consultados no APÊNDICE A.

4.3.2 Código para sensor de corrente ACS712-30A

Alguns detalhes devem ser considerados ao programar este sensor, pois o mesmo apresenta uma tensão de offset em sua medida o que faz necessário o ajuste na expressão para conversão para unidades de corrente elétrica.

Quando uma corrente aplicada ao primário é zero, o ACS712 irá apresentar um valor de $V_{CC}/2$ em sua saída. Tal valor é chamado de tensão quiescente de saída ($V_{IOUT(Q)}$) e é atribuída à resolução do ajuste linear da tensão quiescente derivada ao efeito térmico do circuito integrada da Allegro ACS712 DATASHEET,2007).

Figura 27 – Esquema de ligação do dispositivo ACS712 ligado em série ao chuveiro



Fonte: Próprio autor

Já a tensão de saída é chamada de tensão de compensação elétrica (V_{OE}), em que está é enviada com um valor de offset igual à tensão quiescente de saída. Para a conversão de voltagem para amperes, faz-se necessário dividir o valor pela sensibilidade do dispositivo (para este modelo, a sensibilidade é de 66 mV/A). A expressão para obter o valor da corrente em amperes fica:

$$\text{Corrente elétrica} = \left(\frac{5*(Valor\ lido)}{1023} - 2.5 + 0.007\right) * \frac{1}{0.066}\ A$$

Os detalhes DO programa da leitura deste sensor podem ser consultados no APÊNDICE E.

4.3.3 Código para *sensor de vazão YF-S201*

Como este sensor envia sua resposta em sinais de pulsos em PWM fez-se necessária a utilização do módulo CCP1 (Capture/Compare/PWM) do microcontrolador.

Seu funcionamento é simples: quando um pulso é detectado na entrada CCP1 (por *default* é aplicada à porta RC2 do PIC18F4550) ocorre uma interrupção por CCP e desta forma a execução dos métodos escritos nela. Sendo assim, o código realizado após a ocorrência dessa é interrupção será feito um cálculo para incrementar o volume de água utilizado seguindo as informações estabelecidas no datasheet deste sensor (ANEXO 1).

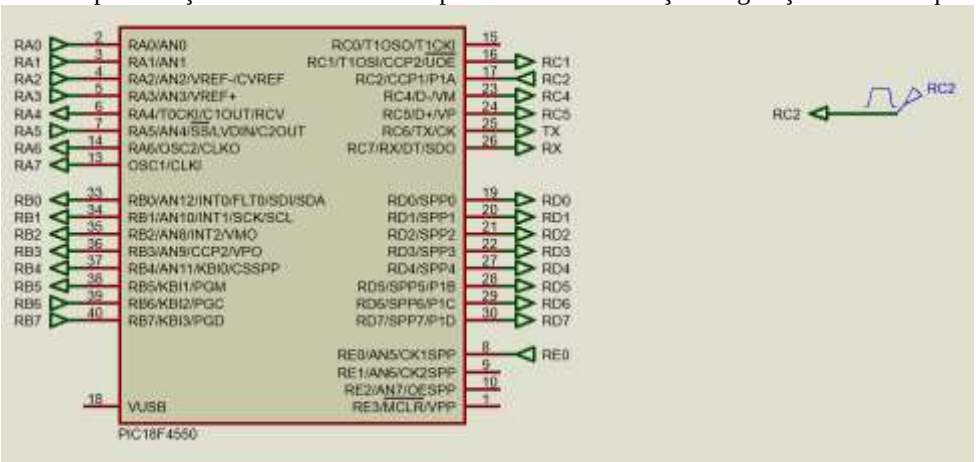
Segundo seu manual, há duas formas se calcular o volume de água gasto:

- Determinando a frequência dos pulsos gerados pelo giro do rotor presente no equipamento;
- Usando a relação pulso-litro que diz que para cada 450 pulsos ocorreu a passagem de 1 L de líquido.

Como o cálculo da frequência de operação do YF-S201 seria trabalho devido as alterações de volume de água passando pelo seu rotor, optou-se por usar a segunda expressão (relação ciclos – litros) e assim já obter o valor direto do consumo sem a necessidade de exigir do microcontrolador espaço de memória para efetuar tais cálculos.

Este dispositivo não possui um bloco de simulação para ser utilizado no *software* de simulação ISIS Proteus, contudo, foi utilizado um geral de pulsos para representá-lo.

Figura 28 - Representação do sensor YF-S201 por meio de uma função de geração de ondas quadradas



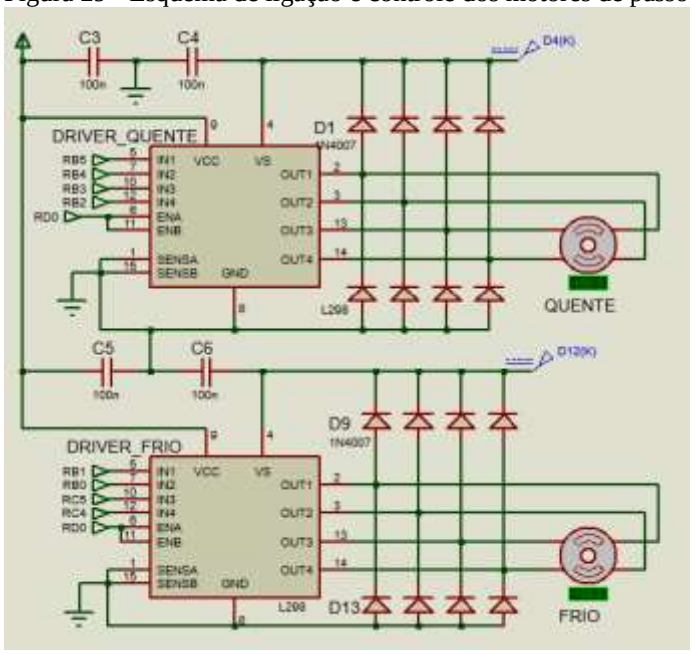
Fonte: Próprio autor

No projeto, foi optado por utilizar a interrupção CCP1 pelo fato destas estarem associadas aos temporizadores TIMER1 e TIMER3. O código para contabilizar o volume de água consumido pode ser consulta no APÊNDICE D.

4.4 Código para acionamento dos motores

Os pinos associados ao controle dos motores de passo foram pinos RB2 a RB5 sendo utilizados para o motor de controle da água aquecida e os pinos RB1, RB0, RC5 e RC4 associados ao motor para controle da água à temperatura ambiente.

Figura 29 - Esquema de ligação e controle dos motores de passo



Fonte: Próprio autor

O pino D0 é utilizado para habilitar e desabilitar os drivers de corrente, sendo então conectado, de forma simultânea, aos pinos *enable* A e *enable* B destes drivers. Após cada grupo de quatro pinos ter sido conectado aos seus respectivos drivers, foi realizada a conexão das saídas OUT1 à OUT4 nas entradas dos motores.

São necessários dois diodos conectados em série em cada uma das ligações que alimenta as bobinas do motor de passo, como mostrado na figura 29, somando um total de 16 diodos para os dois equipamentos. Isso se deve à necessidade de uma proteção contra sobrecorrente, o que pode ser feito por meio de uma ponte de diodos externa com quatro elementos de recuperação rápida (t_{rr} = 200 ns), devendo ser escolhidos de um VF (*Foward Voltage*: tensão de ruptura do diodo) tão baixo quanto o pior caso de corrente de carga (L298N DATASHEET, 2000).

As rotinas de controle dos motores de passo se encontram no APENDICE F.

4.5 Acionamento do chuveiro

O chuveiro elétrico não pode ser ligado pelo microcontrolador de uma maneira direta devido às suas diferenças de alimentação. Desta forma, foi necessário utilizar o dispositivo optoacoplador MOC3041M a fim de que este acione o dispositivo TRIAC por meio de uma corrente aplicada ao seu *gate* (Figura 27), permitindo a passagem de corrente elétrica para alimentação do chuveiro.

Seu acionamento ocorre por meio do pino RC1 ao qual está configurado para utilizar a função CPP2 para gerar um pulso PWM que irá chavear o optoacoplador e, por conseguinte, ligar ou não o chuveiro.

O acionamento se dá mediante a diferença de temperatura entre a aquela medida pelo sensor instalado na tubulação de água aquecida e a temperatura de entrada fornecida pelo usuário.

Diante disso, foram escolhidos três estados de operação para o funcionamento deste componente, sendo eles:

- Desligado: quando a temperatura da água aquecida está em condições satisfatórias à desejada pelo usuário. Para isso, foi determinada uma faixa de controle em que, quando a diferença entre a temperatura da água aquecida e a escolhida pelo usuário sejam inferiores à 6 ºC, o chuveiro permaneça desligado.

- Ligado por PWM a 90%: que é quando a diferença entre as temperaturas citadas esteja entre os valores de 6 ºC a 10 ºC. Nesta configuração, optoacoplador irá permitir passagem durante 10% do ciclo do PWM (período em que este está em nível lógico baixo).
- Ligado por PWM a 60%: que quando a temperatura apresenta uma diferença entre 10 e 16 ºC.
- Ligado: será quando a variação de temperatura exceder o valor de 16 ºC irá fazer o sinal de PWM seja desativado e o chuveiro ficará ligado diretamente.

O código gerado pode ser consultado no APÊNDICE G.

4.6 Rotinas relacionadas ao módulo ESP-01

4.6.1 Conectando o módulo à rede local

O módulo ESP-01 é conectado ao microcontrolador através das portas de transmissão serial do mesmo. Sua programação é realizada por meio de sua sequência de comandos AT preestabelecida de fábrica.

Figura 30 – Alguns comandos AT para configuração do módulo ESP-01

Comando AT	Função
AT+GMR	verifica versão do Firmware
AT+SYSGPIOWRITE	liga/desliga um pino do módulo
AT+SYSGPIOREAD	lê o valor de um pino
AT+UART_CUR	configura comunicação UART
AT+CWMODE_CUR	configura o modo WiFi(access point, cliente)

Fonte: ESP8266 DATASHEET, 2019

Os principais comandos utilizados para efetuar sua conexão foram:

- AT+CWMODE=1: configuração para modo STATION.
- AT+CWJAP: estabelece a conexão do módulo com uma rede local.
- AT+CIPMUX: configura para conexão em conexão *single* ou *multilpes*.
- AT+CIPSTART: estabelece conexão do módulo com o site hospedeiro.

- AT+CIPSEND: comando que enviar os dados ao endereço *http* estimulado pelo comando AT+CIPSTART.
- AT+CIPCLOSE: comando para fechar conexão *http* após envio de informação.

O código para conexão e envio de dados pode ser consultado no APÊNDICE H.

4.6.2 Gerando o JSON de dados

JSON é um modelo de armazenamento e transmissão de informação em formato de texto que tem sem sido amplamente utilizado em aplicações *web* por ser capaz de estruturar informações de forma simples e compacta. O formato de um dado JSON segue o modelo abaixo.

{
"Corrente":[22.1253,0.0036,14,0.000785,16.236514,0.00069,21.6395,0.001356,21.32153,0.00326,18.96,0.00412],
"Agua": [36.4412,0.256075]
}

Nele, as chaves delimitam o dado JSON e os argumentos entre aspas são chamados de rótulos, enquanto que os dados entre colchetes são os seus valores. Optou-se pelo envio de dados por este formato devido a sua grande difusão com ferramenta prática e eficiente para o envio de dados carregados de informações.

Apesar de ter sua visualização complexa, os aplicativos atuais o decodifica e interpreta facilmente, possibilitando sua manipulação em forma de *array* (listas).

4.7 Programação do aplicativo Android

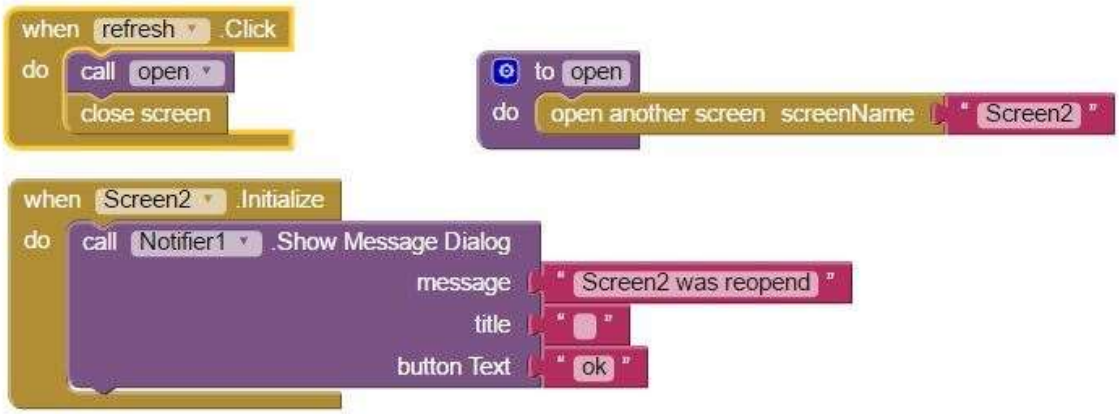
Para o desenvolvimento do aplicativo que receberá os dados de corrente, tempo de banho e consumo de água foi o utilizado a plataforma de desenvolvimento Thunkable®.

O aplicativo foi desenvolvido de modo a apresentar uma interface simples e capaz de disponibilizar ao usuário as informações mais importantes de um modo legível a este e completa.

Para isto, o aplicativo foi dividido em cinco *screens* (telas) em que a principal é responsável por permitir a escolha de qual arquivo será visto e as outras são responsáveis por mostrar os dados referentes ao consumo de energia e seu custo, o consumo de água e seu custo.

Por ser um software programado em estruturação por meio de blocos, se torna uma ferramenta que oferece praticidade e desenvoltura na programação de um aplicativo

Figura 31 - Aplicativo codificado por meio da plataforma Thunkable



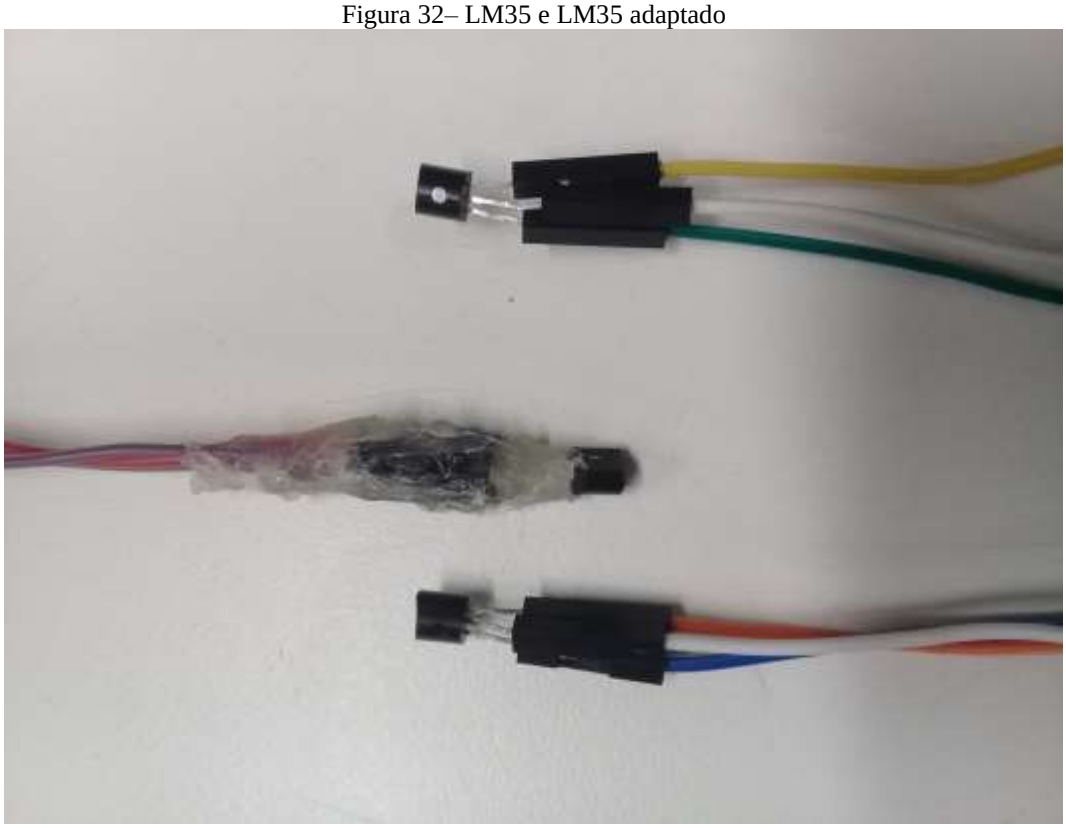
Fonte: Thunkable Website, 2019

5 RESULTADOS

5.1 Sensores de temperatura

Os sensores LM35 são produzidos na forma de pastilha de circuito integrado encapsulado em material plástico, apresentado características estéticas semelhando à um transistor e foi por isso que ele precisou ser modificado a fim de ser utilizado submerso em água. Para isto foi necessário envolvê-lo em cola de silicone de material termo retrátil, evitando que seus contatos possam entrar em contato com a água e ocorram possíveis curtos-circuitos.

Figura 32– LM35 e LM35 adaptado



Fonte: Próprio autor

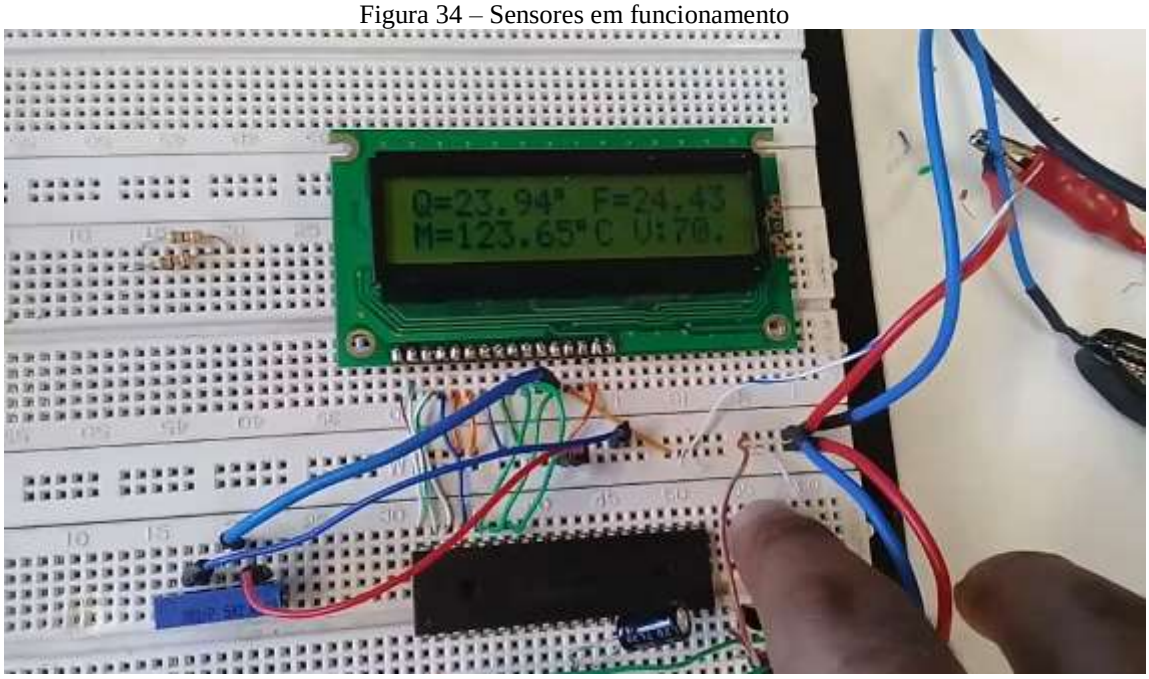
Cores quentes e frias foram utilizadas para a alimentação e terra, respectivamente, enquanto cores de tons acinzentadas tiveram preferência em ser utilizadas como o cabo para comunicar os sensores com a entradas analógicas do PIC18F4550.

Os dispositivos foram testados após adaptação e o resultado por ser conferido nas imagens da Figura 33 e Figura 34. A temperatura é representa pelas letras F e Q aparecendo no display LCD.

Figura 33 – Adaptação completa dos sensores de temperatura

Fonte: Próprio autor

Figura 34 – Sensores em funcionamento

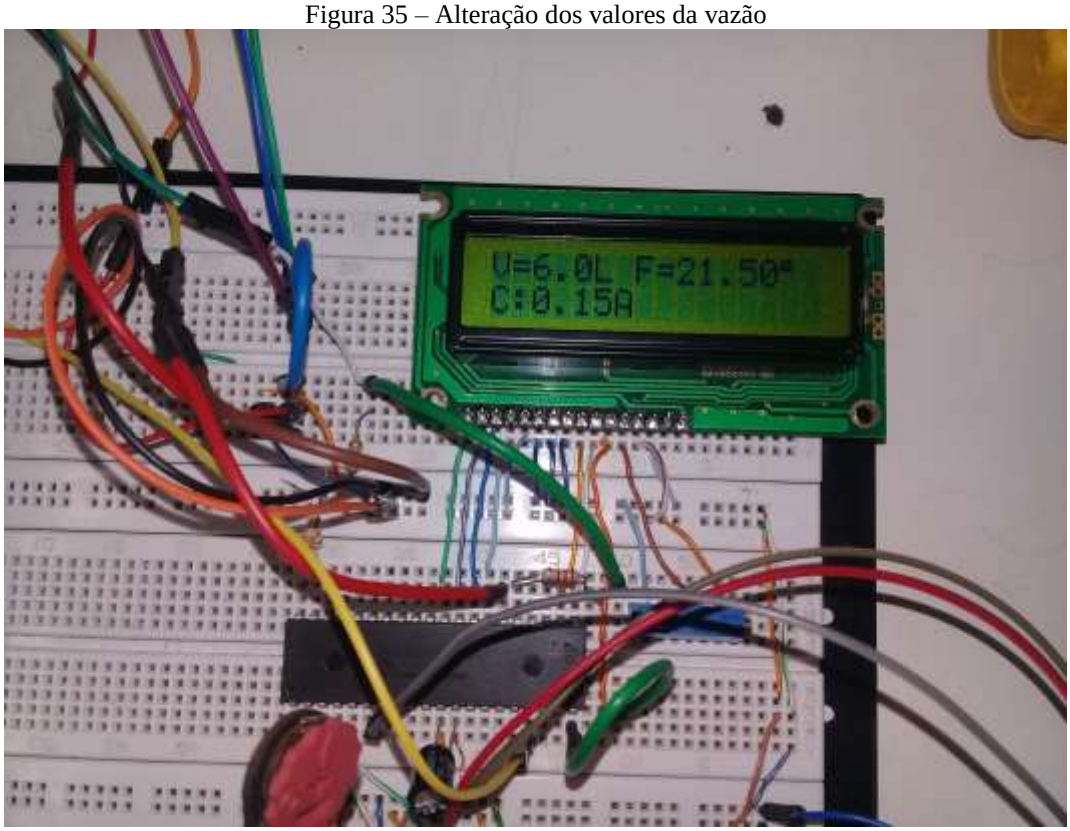


Fonte: Próprio autor

5.2 Sensor de vazão

O sensor de vazão foi testando mediante rotação forçada de seu rotor durante o tempo de 1 minuto, entregando o valor abaixo.

Figura 35 – Alteração dos valores da vazão



Fonte: Próprio autor

5.3 Aplicativo desenvolvido

O aplicativo desenvolvido apresenta 5 *screens* de utilização, sendo a primeira utilizada para acessar as outras. O mesmo foi programado para buscar as informações em um servidor *web* a cada 3 segundos, verificando se o arquivo JSON foi modificado, comparando suas informações, apresentando o consumo ao usuário por meio de gráficos.

Para os gráficos envolvido consumo de energia elétrica, é possível verificar o consumo com controle, a estimativa de consumo sem a introdução do mesmo e a diferença do gasto relacionado à tarifa de energia.

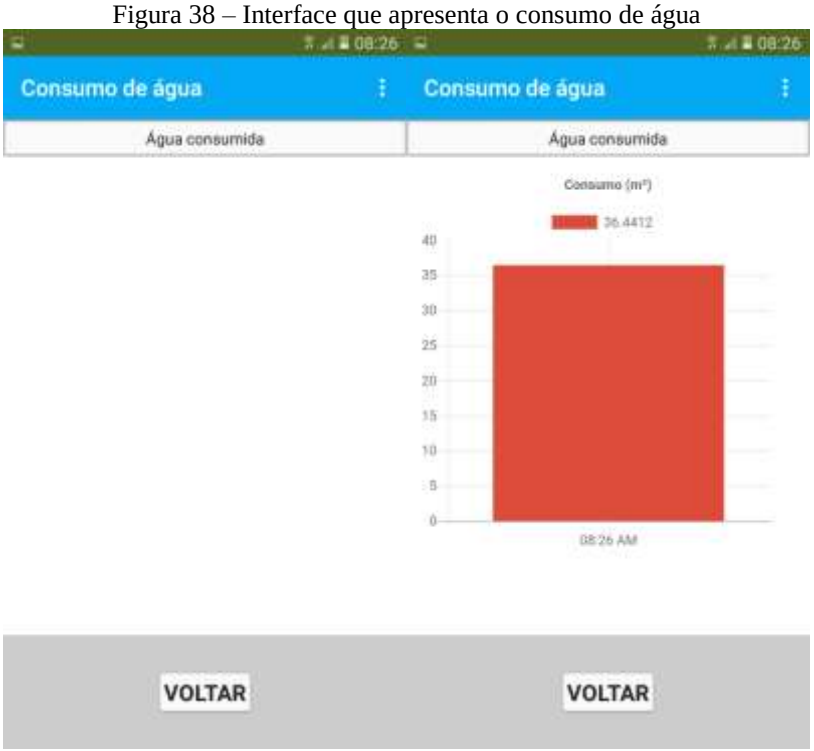
Figura 36 – Camada principal do aplicativo web

Fonte: Próprio autor

Figura 37 – Interface que apresenta consumo de ao longo do dia

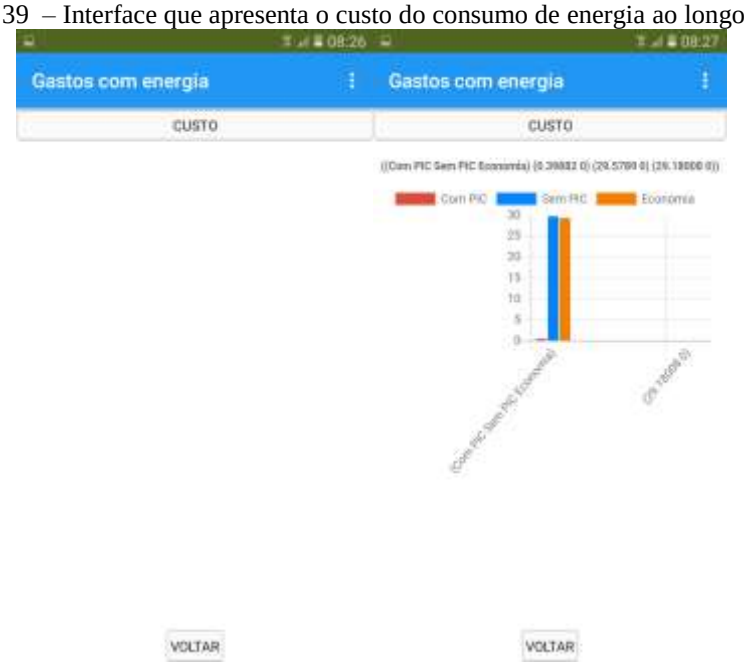
Fonte: Próprio autor

Figura 38 – Interface que apresenta o consumo de água



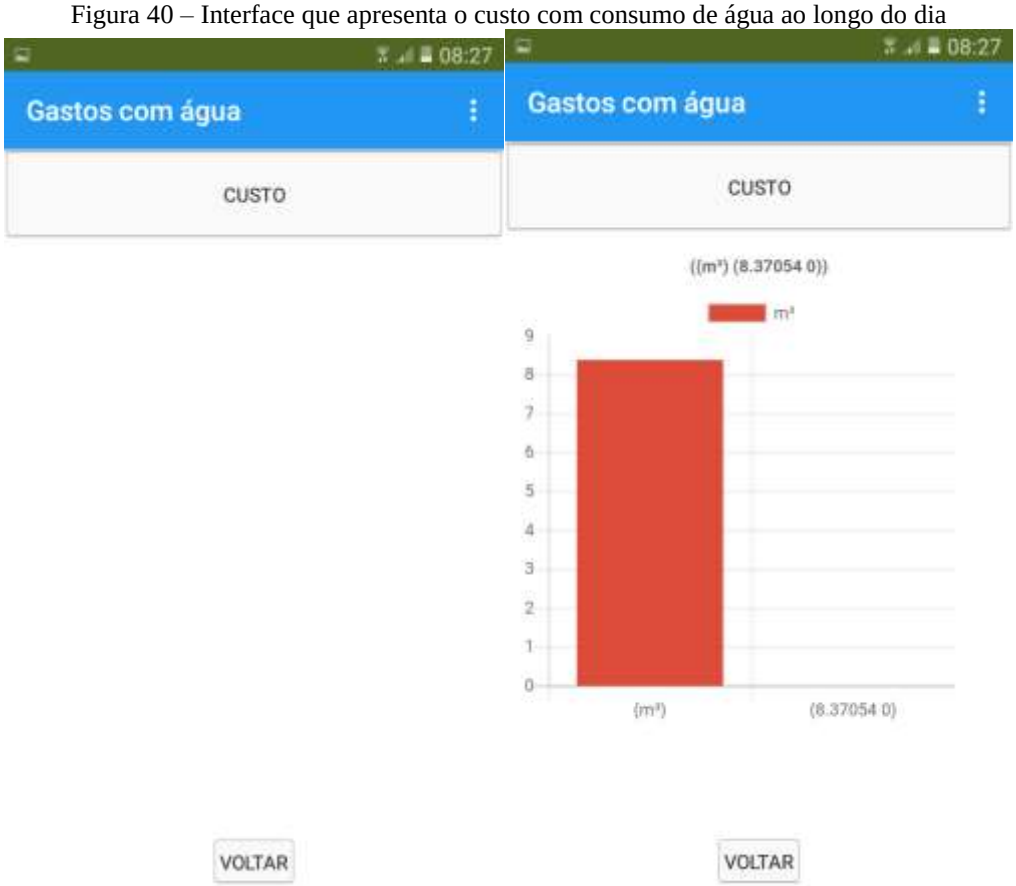
Fonte: Próprio autor

Figura 39 – Interface que apresenta o custo do consumo de energia ao longo do dia



Fonte: Próprio autor

Figura 40 – Interface que apresenta o custo com consumo de água ao longo do dia



Fonte: Próprio autor

6 CONCLUSÕES

Este trabalho reforça a ideia de controle e automação como as principais áreas da engenharia que estarão em contínuo progresso, com suas tecnologias e o modo como estas se relacionam sendo aprimorado.

Os dispositivos que possibilitam a integração de microcontroladores com rede mundial de internet estão cada vez mais acessíveis, possibilitando que, além destes automatizarem o controle dos bens de consumo de um estabelecimento, possibilitando o acesso às informações que estão sendo administradas durante a dinâmica do sistema e, desta forma, o usuário conseguirá estabelecer parâmetros e conclusões acerca do que está ocorrendo.

A Internet das Coisas associada à constante evolução dos dispositivos microcontroladores e sistemas embarcados é a ferramenta de engenharia mais dinâmica e agente potencializador da integração entre usuário e sistemas de controle e automação que, em outros remotos, apenas teriam sido administrados por si só, mediante a programação sua CPU, sem a presença intensiva da observação e ação humana.

Em suma, o controle de aquecimento de água pela associação de microcontrolador com placas de aquecimento solar juntamente com integração dos mesmos com as ideias da Internet das Coisas acarreta na economia no que tange aos custos envolvendo o aquecimento da mesma por meio de chuveiros convencionais, sendo aplicadas, principalmente, em estabelecimentos e residências que já possuem o aquecedor solar e associado a ele, apresentam consumo considerado de água e energia.

APÊNDICES

APÊNDICE A – BIBLIOTECA DAS ROTINAS PARA OS SENSORES DE TEMPERATURA

```
float tempQ,tempF,tempM;


float ler_quente()
{
  set_adc_channel(0); //Define a porta RA0 que será entrada analógica do percentual de vazão
  delay_us(10);
  tempQ = read_adc();
  tempQ = 5*tempQ/(0.01*1023);

  return tempQ;
}

float ler_frio()
{
  set_adc_channel(1); //Define a porta RA1 que será entrada analógica do percentual de vazão
  delay_us(10);
  tempF = read_adc();
  tempF = 5*tempF/(0.01*1023);

  return tempF;
}

float ler_mistura()
{
  set_adc_channel(2); //Define a porta RA2 que será entrada analógica do percentual de vazão
  delay_us(10);
  tempM = read_adc();
  tempM = 5*tempM/(0.01*1023);

  return tempM;
}
```

APÊNDICE B – BIBLIOTECAS E DEFINIÇÕES DOS PINOS UTILIZADOS

```
#include<18F4550.h>

#device ADC=10

#use delay (clock=8000000)

#include <STDLIB.H>
#include <string.h>

    #FUSES HS            //High speed (alta velocidade)
    #FUSES NOMCLR        //Sem master-clear
    #FUSES NOWRT         //Sem watchdog timer
    #FUSES NOPROTECT     //Sem proteção contra cópias
    #FUSES NOLVP         //Sem low voltage programming
    #FUSES NOPUT

#use rs232(baud=9600,parity=N,xmit=PIN_C6,rcv=PIN_C7,bits=8,ERRORS)

    #define LCD_RS_PIN PIN_D1
    #define LCD_RW_PIN PIN_D2
    #define LCD_ENABLE_PIN PIN_D3
    #define LCD_DATA4 PIN_D4
    #define LCD_DATA5 PIN_D5
    #define LCD_DATA6 PIN_D6
    #define LCD_DATA7 PIN_D7

#include <lcd.c>
#define use_portd_lcd TRUE

    #define Q0 PIN_B2
    #define Q1 PIN_B3
    #define Q2 PIN_B4
    #define Q3 PIN_B5

    #define F0 PIN_C4
    #define F1 PIN_C5
    #define F2 PIN_B0
    #define F3 PIN_B1

    #define EN PIN_D0     //Pino que habilita os Motores
    #define RED PIN_E0    //LED que indica que o chuveiro está acionado

    #include <SENSORES.c>

    //Temperatura ao fim do trabalho será salva no endereço 10 da EEPROM
    #define ultima_temperatura 10

#priority INT_TIMER0, INT_CCP1, INT_TIMER1
```

APÊNDICE C –VARIÁVEIS GLOBAIS UTILIZADAS

```
float
  //Variaveis para receber entradas analogicas
  pot_vazao, temp_Q, temp_F, temp_M,

  //Contadores para os passos dos motores
  m1=1, m2=1, contador_motor=0,

  variacao_chuveiro,

  //Variáveis auxiliares para salvar corrente e tempo mensurados
  aux_t=0,aux_c=0,

  corrente, corrente_vetor[], tempo=0 ,tempo_corrente[],

  //Recebe a conversao para volume de água gasta em Litros
  agua_consumida=0,tempo_BANHO=0,
  tempo_minutos,

  //Diferenção entre a vazão escolhida e a atual de funcionamento
  dif_vazao,
  //Diferença entre a temperatura escolhida e a temperatura da mistura
  dif_temperatura,
  //Valor absoluto da diferença de temperatura
  temp_abs,
  //Valor que indica a soma das vazões do sistema aquecido e à temperatura ambiente
  vazao_total,
  //Valor absoluto da diferença da vazão pretendida e a atual do sistema
  vazao_abs;

int1
   gravado_eeprom=0,

   //Indica se chuveiro LIGADO/CHUVEIRO
   chuveiro=0,
   //Bit de controle para saber se válvula QUENTE está totalmente aberta
   motor1_max=0,
   //Bit de controle para saber se válvula QUENTE está totalmente fechada
   motor1_min=0,

   //Bit de controle para saber se válvula FRIO está totalmente aberta
   motor2_max=0,
   //Bit de controle para saber se válvula FRIO está totalmente fechada
   motor2_min=0,
   //Flags para acionar ocontrole de vazão e temperatura
   flag_T, flag_V=0;

int
  //Variável de controle para rotina de interrupção
```

```
  pb,
  //Tempo de delay para alternância de polos do motor de passo
  time = 6,

  temperatura,

  contador_corrente,

  count=0,
  //Variável para gravação na EEPROM
  temperatura_ao_fim;

int16

  aplica_vazao,

  //Determina a abertura máxima para as válvulas em cada motor (50%)
  MAX_ROTACAO = 800;

int32
  //recebe quantidade de pulsos do sensor YF-S201 durante o banho
  pulsos=0;
```

APÊNDICE D – ROTINAS DE INTERRUPÇÃO UTILIZADAS

```
#INT_RB
void RB_isr()
{
  pb=input(PIN_B7); /*É necessário uma leitura na porta B para poder limpar a
  flag de interrupção*/

  clear_interrupt(INT_TIMER0);     //Limpa flag de overflow TIMER0
  set_timer0(6);                   //Inicia contagem do TIMER0 em 6
  enable_interrupts(INT_TIMER0);   //Inicia o TIMER0
  clear_interrupt(INT_RB);         //Limpa flag de de interrupção
}

#INT_TIMER0
void TIMER0_isr()
{ //Método para usuário escolher temperatura
  clear_interrupt(INT_TIMER0); //Limpa flag de overflow TIMER0
  disable_interrupts(INT_TIMER0); //Para contagem do TIMER0

  if(input(PIN_B7))
  {
   temperatura++;//Acrescenta 1 °C à temperatura que o usuário deseja
  }

  if (input(PIN_B6))
  {
   temperatura--;//Diminui 1 °C à temperatura que o usuário deseja
  }

  if(temperatura>60) temperatura=60;
  if(temperatura<temp_F) temperatura=temp_F;
}

#INT_TIMER1
void TIMER1_isr()
{
    //Limpa flag de overflow TIMER1
    clear_interrupt(INT_TIMER1);

    //Contador de tempo de banho a cada 0,2s
    tempo_BANHO=tempo_BANHO+0.2;
    tempo_minutos = tempo_BANHO/60;

    set_timer1(15536);

    //Contador para acionar controles a cada 0,6s
    count++;

    if(count>2)
```

```
    {
      flag_T=1;
      flag_V=1;
      count=0;
    }

    if(chuveiro==1)
    {
      tempo=tempo+0.2;
      aux_c=corrente;
      aux_t=tempo;
    }

    if(chuveiro==0&&corrente<=0.5)
    {
      if(aux_t!=0)
      {
        corrente_vetor[contador_corrente]=aux_c;
        tempo_corrente[contador_corrente]=aux_t;

        contador_corrente++;
        tempo=0;
      }
      else aux_t=0;
    }
}

#INT_CCP1
void conta_pulsos_isr()
{
  pulsos++;
  agua_consumida=(float) (pulsos/450);
}

#INT_TIMER2
void TIMER2_isr()
{
  clear_interrupt(INT_TIMER2);
}
```

APÊNDICE E – PROTÓTIPOS PARA LEITURA DO POTENCIÔMETRO, CORREÇÃO DE TEMPERATURA E MEDIDA DE CORRENTE ELÉTRICA

```
float def_vazao() //Leitura do potenciômetro
{
  set_adc_channel(3); //Define a porta RA3 que será entrada analógica do percentual de vazão
  delay_us(10);
  float vazaoADC = read_adc();
  float pot = (vazaoADC/1023)*100; // Define o percentual de vazão dos servo motores
  return pot;
}
void corrige_temperatura()
{
  if(temperatura<temp_F) temperatura=temp_F;
}

float ACS712_30A_AC()
{
  set_adc_channel(4);
  delay_us(10);
  float tensao_lida, vetor_tensao[100] , correnteAC;
  float maior=0;

  for(int i=0;i<100;i++)
  {
    vetor_tensao[i]=read_adc();
    //Delay de 166 microssegundos para evitar flutuação na entrada do PIC
    delay_us(166.68);
  }
  for(int j=0;j<100;j++)
  {
    if(maior<vetor_tensao[j])
    {
      maior=vetor_tensao[j];
    }
  }
  if(maior>=512)
  {
    tensao_lida=(maior*5/1023)-2.5+0.007;
    correnteAC=(tensao_lida/0.066)/1.414213562373095;
  }
  else
  {
    tensao_lida=2.5-(maior*5/1023)+0.007;
    correnteAC=(tensao_lida/0.066)/1.414213562373095;
  }

  return correnteAC;
}
```

APÊNDICE F – PROTÓTIPOS PARA ACIONAMENTO DOS MOTORES

```
void abrir_quente(int16 passos)
{
  if(motor1_max==0)
  {
    while(passos>contador_motor)
    {
      if((m1+contador_motor)>=800)
      {
        motor1_max=1;
        break;
      }

      //Step (3)
      output_high(Q1);
      output_low(Q0);
      output_high(Q2);
      output_low(Q3);

      delay_ms(time);

      //Próximo step (2)
      output_low(Q1);
      output_high(Q0);

      delay_ms(time);

      contador_motor=contador_motor+2;

      if((m1+contador_motor)>=800)
      {
        motor1_max=1;
        contador_motor=0;
        break;
      }

      //Próximo step (1)
      output_low(Q2);
      output_high(Q3);
      delay_ms(time);

      //Próximo step (0)
      output_high(Q1);
      output_low(Q0);
      delay_ms(time);

      contador_motor=contador_motor+2;

     if((m1+contador_motor)>=800)
```

```
        {
          motor1_max=1;
          contador_motor=0;
          break;
        }
      }

      //Informa com quantos passos o motor está
      m1=m1+contador_motor;

      if((m1+contador_motor)>=800)
      {
        motor1_max=1;
      }
      else
      {
        motor1_min=0;
      }
      contador_motor=0;

      //num=0;
      }
}

void fechar_quente(int16 passos)
{

  if(motor1_min==0) // Verifca se válvula está totalmente fechada
  {
    while (passos>contador_motor)
    {
      //No caso da válvula estar no mínimo projetado, sai do laço de execução
      if((m1-contador_motor)<=0)
      {
        motor1_min=1;
        contador_motor=0;
        break;
      }

      //Step (0)
      output_high(Q1);
      output_low(Q0);
      output_high(Q3);
      output_low(Q2);

      delay_ms(time);

      //Próximo step (1)
      output_low(Q1);
      output_high(Q0);
```

```
      delay_ms(time);

      contador_motor=contador_motor+2;

      if((m1-contador_motor)<=0)
      {
        motor1_min=1;
        contador_motor=0;
        break;
      }

      //Próximo step (2)
      output_low(Q3);
      output_high(Q2);
      delay_ms(time);

      //Próximo step (3)
      output_high(Q1);
      output_low(Q0);
      delay_ms(time);

      contador_motor=contador_motor+2;

      if((m1-contador_motor)<=0)
      {
        motor1_min=1;
        contador_motor=0;
        break;
      }
    }

    //Informa com quantos passos o motor está
    m1=m1-contador_motor;

    if((m1-contador_motor)<=0)
    {
      motor1_min=1;
    }
    else
    {
      motor1_min=0;
    }
    contador_motor=0;

    //num=0;
  }
}

void abrir_frio(int16 passos)
```

```
{
  if(motor2_max==0)
  {
    while(passos>contador_motor)
    {
      if((m2+contador_motor)>=MAX_ROTACAO)
      {
        motor2_max=1;
        break;
      }

      //Step (3)
      output_high(F1);
      output_low(F0);
      output_high(F2);
      output_low(F3);

      delay_ms(time);

      //Próximo step (2)
      output_low(F1);
      output_high(F0);

      delay_ms(time);

      contador_motor=contador_motor+2;

      if((m2+contador_motor)>=MAX_ROTACAO)
      {
        motor2_max=1;
        contador_motor=0;
        break;
      }

      //Próximo step (1)
      output_low(F2);
      output_high(F3);
      delay_ms(time);

      //Próximo step (0)
      output_high(F1);
      output_low(F0);
      delay_ms(time);

      contador_motor=contador_motor+2;

     if((m2+contador_motor)>=MAX_ROTACAO)
      {
        motor2_max=1;
        contador_motor=0;
```

```
        break;
      }
    }

    //Informa com quantos passos o motor está
    m2=m2+contador_motor;

    if((m2+contador_motor)>=MAX_ROTACAO)
    {
      motor2_max=1;
    }
    else
    {
      motor2_min=0;
    }
    contador_motor=0;

    }
}

void fechar_frio(int16 passos)
{

  if(motor2_min==0) // Verifca se válvula está totalmente fechada
  {
    while (passos>contador_motor)
    {
      //No caso da válvula estar no mínimo projetado, sai do laço de execução
      if((m2-contador_motor)<=0)
      {
        motor2_min=1;
        contador_motor=0;
        break;
      }

      //Step (0)
      output_high(F1);
      output_low(F0);
      output_high(F3);
      output_low(F2);

      delay_ms(time);

      //Próximo step (1)
      output_low(F1);
      output_high(F0);

      delay_ms(time);

      contador_motor=contador_motor+2;
```

```
      if((m2-contador_motor)<=0)
      {
        motor2_min=1;
        contador_motor=0;
        break;
      }

      //Próximo step (2)
      output_low(F3);
      output_high(F2);
      delay_ms(time);

      //Próximo step (3)
      output_high(F1);
      output_low(F0);
      delay_ms(time);

      contador_motor=contador_motor+2;

      if((m2-contador_motor)<=0)
      {
        motor2_min=1;
        contador_motor=0;
        break;
      }
    }

    //Informa com quantos passos o motor está
    m2=m2-contador_motor;

    if((m2-contador_motor)<=0)
    {
      motor2_min=1;
    }
    else
    {
      motor2_min=0;
    }
    contador_motor=0;

  }
}
```

APÊNDICE G – PROTÓTIPOS PARA CONTROLE DO CHUVEIRO E PARA CONTROLE DE TEMPERATURA E VAZÃO

```
void chuveiro_acionamento()
{
  //Delta de varição da temperatura aplicada ao chuveiro:
  variacao_chuveiro = (float) temperatura - temp_Q;

  if(variacao_chuveiro>=6)
  {
    output_high(RED); //Chuveiro acionado pela rede do sistema
    delay_ms(500);

    //Ativa os drivers L298 que controlam os motores
    output_high(EN);

    //Indica que o chuveiro foi ligado
    chuveiro = 1;

    //Estipula a vazão a ser aplicada ao motor
    aplica_vazao = (int16) abs((m1/16-pot_vazao)*8);

    fechar_frio(max_ROTACAO);

    //Condicional verificando se vazão atual está maior que a escolhida
    if((m1/16)>pot_vazao)
    {
      //Diminue vazão
      fechar_quente(aplica_vazao);

    }
    else if ((m1/16)<pot_vazao)
    {
      abrir_quente(aplica_vazao);
    }

    if(variacao_chuveiro>=6&&variacao_chuveiro<=10)
    {
      //Seleciona um duty-cycle de 90%
      set_pwm2_duty(749);
      //OPTOACOPLADOR IRÁ PERMITIR 10% de ciclo de passagem de corrente pelo
TRIAC
    }
    if(variacao_chuveiro>10&&variacao_chuveiro<16)
    {
      //Seleciona um duty-cycle de 40%
      set_pwm2_duty(334);
      //OPTOACOPLADOR IRÁ PERMITIR 60% de ciclo de passagem de corrente pelo
TRIAC
    }
```

```
    if(variacao_chuveiro>16)
    {
      //Seleciona um duty-cycle de 0%
      set_pwm2_duty(0);
      //OPTOACOPLADOR IRÁ PERMITIR 100% de ciclo de passagem de corrente pelo
TRIAC
    }

    output_low(EN);

    vazao_total=((m2+m1)/16); //Estima a porcentagem de vazão total do sistema

    output_low(RED);
  }
  else
  {
    chuveiro=0;
    //Seleciona um duty-cycle de 100%
    set_pwm2_duty(1023); //Desliga o chuveiro
    //OPTOACOPLADOR IRÁ PERMITIR 0% de ciclo de passagem de corrente pelo TRIAC
  }

}

void controle_temperatura()
{
  if(chuveiro==0)
  {
    output_high(EN);

    if(temp_M<temperatura)
    {
      fechar_frio(10);
      abrir_quente(10);
    }
    else
    {
      fechar_quente(10);
      abrir_frio(10);
    }

    output_low(EN);
  }
}

void controle_vazao()
{
  if(chuveiro==0)
  {
    //Controle operando em uma faixa DELTA=6°C
```

```
if(temp_abs<=6)
{
  //Habilita os motores
  output_high(EN);

  //Divide o valor da vazao em duas partes (cada uma aplica a um motor)
  aplica_vazao=(int16) (vazao_abs*8);


  if(dif_vazao>0)
  {
    //Verifica se motores estão aberto no máximo projetado
    if(motor1_max==0&&motor2_max==0)
    {
      abrir_quente(aplica_vazao);
      abrir_frio(aplica_vazao);
    }
  }

  if(dif_vazao<0)
  {
    //Verifica se motores estão totalmente fechados
    if(motor1_min==0&&motor2_min==0)
    {
      fechar_quente(aplica_vazao);
      fechar_frio(aplica_vazao);
    }
  }

  //Verifica se motor 1 está no seu mínimo para controlar apenas o motor 2 (FRIO)
  if(motor1_min==1)
  {
    aplica_vazao = (int16) abs(((m2/16)-pot_vazao)*8);

    if((m2/16)>pot_vazao)
    {//Verifica se motor FRIO está com vazão superior à estipulada
      fechar_frio(aplica_vazao);
    }
    else
    {
      abrir_frio(aplica_vazao);
    }
  }

   //Verifica se motor 2 está no seu mínimo para controlar apenas o motor 2 (FRIO)
  if(motor2_min==1)
  {
    aplica_vazao = (int16) abs(((m1/16)-pot_vazao)*8);

    if((m1/16)>pot_vazao)
```

```
          {//Verifica se motor frio está com vazão superior à estipulada
            fechar_quente(aplica_vazao);
          }
          else
          {
            abrir_quente(aplica_vazao);
          }
        }

        //Desabilita os motores
        output_low(EN);
      }
    }

    vazao_total=(m1+m2)/16;
}

void atualiza_motores()
{
    if(m1<MAX_ROTACAO) motor1_max=0;
    else motor1_max=1;

    if(m1>0) motor1_min=0;
    else motor1_min=1;

    if(m2<MAX_ROTACAO) motor2_max=0;
    else motor2_max=1;

    if(m2>0) motor2_min=0;
    else motor2_min=0;

    vazao_total=(m1+m2)/16;
}
```

APÊNDICE H – PROTÓTIPOS PARA UTILIZAÇÃO DO MÓDULO ESP-01

```
char json_A_ENVIAR[250];

//Rotina gera um dado JSON a ser enviado para website e posteriormente ser analisado pelo aplicativo
void JSON()
{

  char corrente_dados[250] = "{{\"Corrente\":[";
  char concatenador_corrente[20] = "20.4807,0.00478596"; //Valor de inicialização

  char agua_dados[30] = "{\"Agua\":[";
  char concatenador_agua[20] = "38.6314,14.4714"; //Valor de inicialização

  for(int i=0; i<sizeof(corrente_vetor);i++)
  {
    if(i==0)
    {
      sprintf(concatenador_corrente,"%.4f,%.8f", corrente_vetor[i],tempo_corrente[i]);
      strcat(corrente_dados,concatenador_corrente);
    }
    else if(i!=0&&i<(sizeof(corrente_vetor)-1))
    {
      sprintf(concatenador_corrente,",%.4f,%.8f", corrente_vetor[i],tempo_corrente[i]);
      strcat(corrente_dados,concatenador_corrente);
    }

    if(i==(sizeof(corrente_vetor)-1))
    {
      sprintf(concatenador_corrente,",%.4f,%.8f", corrente_vetor[i],tempo_corrente[i]);
      strcat(corrente_dados,concatenador_corrente);

      char fim = "]},";

      strcat(corrente_dados,fim);
    }
  }

  sprintf(concatenador_agua,"%.4f,%.4f]}}",agua_consumida,tempo_horas);
  strcat(agua_dados,concatenador_agua);

  json_A_ENVIAR[250] = strcat(corrente_dados,agua_dados);
}

void INICIA_ESP01()
{
  /*Seta o ESP-01 em modo Station
  Resposta será STA*/
  char comando_MODO_DE_OPERACAO[] = "AT+cwmode=1\r\n\0";
```

```
  char comando_CONECTA_AO_ROTEADOR[] =
"AT+CWJAP=\"DiegoSS\",\"iwnv5844\"\r\n\0";

  //Seta o ESP-01 com conexão única (Se 1: múltiplas conexões)
  char comando_CONEXOES[] = "AT+CIPMUX=0\r\n\0";

  puts(comando_MODO_DE_OPERACAO);
  printf(lcd_putc,"%s",getc());
  delay_ms(500);

  puts(comando_CONECTA_AO_ROTEADOR);
  printf(lcd_putc,"%s",getc());
  delay_ms(500);

  puts(comando_CONECTA_AO_ROTEADOR);
  printf(lcd_putc,"%s",getc());
  delay_ms(500);

  puts(comando_CONEXOES);
  printf(lcd_putc,"%s",getc());
  delay_ms(500);
}

void VERIFICA_CONEXAO()
{
  char comando_WORKING[] = "AT\r\n\0"; //Se funcionando, respota será OK
  char comando_GET_IP[] = "AT+CIFSR\r\n\0";

  puts(comando_WORKING);
  printf(lcd_putc,"%s",getc());
  delay_ms(500);

  puts(comando_GET_IP);
  printf(lcd_putc,"%s",getc());
  delay_ms(500);
}

void CONECTA_AO_WEBSITE()
{
  char comando_CONECTA_AO_ARMAZENAMENTO[] =
"AT+CIPSTART=\"TCP\",\"http://netfuzzer.000webhostapp.com/\",80\r\n\0";

  puts(comando_CONECTA_AO_ARMAZENAMENTO);
  printf(lcd_putc,"%s",getc());
  delay_ms(500);
}

void ENVIAR_EPS01()
{
```

```
  //INFORMA AO ESP-01 A QUANTIDADE DE CARACTERES
  char comando_TAMANHO_ENVIO[] = "AT+CIPSEND="; //Manda o tamanho da string dados

  char comando_ENVIAR_DADOS[] = "POST /salvar.php HTTP/1.1\r\nHost:netfuzzer.000webhostapp.com\r\nContent-Length:";

  char comando_FECHA_COMUNICACAO = "AT+CIPCLOSE\r\n\0";

  char ENVIANDO_DADOS[380];

  char comando_INFORMA_TAMANHO[25];

  int16 tamanho_dados;

  int  tam_json;

  tam_json=strlen(json_A_ENVIAR);

  memset(ENVIANDO_DADOS,0,sizeof(ENVIANDO_DADOS));

  sprintf(ENVIANDO_DADOS,"%s%u\r\nContent-type:application/json\r\n\r\n%s",comando_ENVIAR_DADOS,tam_json,json_A_ENVIAR);

  memset(comando_INFORMA_TAMANHO,0,sizeof(comando_INFORMA_TAMANHO));

  tamanho_dados=strlen(ENVIANDO_DADOS);

sprintf(comando_INFORMA_TAMANHO,"%s%lu\r\n\0",comando_TAMANHO_ENVIO,tamanho_dados);


  puts(comando_INFORMA_TAMANHO);
  printf(lcd_putc,"%s",getc());
  delay_ms(500);

  puts(ENVIANDO_DADOS);
  printf(lcd_putc,"%s",getc());
  delay_ms(500);

}
```

APÊNDICE I – ROTINA PRINCIPAL DO SISTEMA

```
void main()
{
  setup_comparator(NC_NC_NC_NC);

  setup_adc_ports(AN0_TO_AN4|VSS_VDD);
  setup_adc(ADC_CLOCK_INTERNAL);

  enable_interrupts(GLOBAL);
  enable_interrupts(INT_RB);

  disable_interrupts(INT_TIMER0);
  setup_timer_0(RTCC_INTERNAL|RTCC_DIV_4|RTCC_8_bit);

  disable_interrupts(INT_TIMER1);
  setup_timer_1(T1_INTERNAL|T1_DIV_BY_8);
  setup_ccp1(CCP_CAPTURE_RE);
  enable_interrupts(INT_CCP1);

  setup_ccp2(CCP_PWM);
  setup_timer_2(T2_DIV_BY_16,207,1);
  enable_interrupts(INT_TIMER2);
  set_pwm2_duty(1023);

  disable_interrupts(INT_EXT);

  set_tris_a(0x2F); //Seta as portas AN0, AN1,AN2,AN3 e AN4 como entradas.

  lcd_init();
  printf(lcd_putc,"\f***Iniciando***\n");
  delay_ms(500);

  output_low(EN); //Desabilita motores

  INICIA_ESP01();

  VERIFICA_CONEXAO();

  while(TRUE)
  {
    temp_Q = ler_quente(); //Leitura de temperatura na AN0
    temp_F = ler_frio();   //Leitura de temperatura na AN1
    temp_M = ler_mistura();//Leitura de temperatura na AN2

    pot_vazao = def_vazao();//Leitura de potenciômetro na AN3

    corrige_temperatura();

    while(pot_vazao>=5.24)
```

```
{
  temp_Q = ler_quente();
  temp_F = ler_frio();
  temp_M = ler_mistura();

  pot_vazao = def_vazao();

  enable_interrupts(INT_TIMER1);

  dif_temperatura= (float) temperatura-temp_M;

  dif_vazao=pot_vazao-vazao_total;

  temp_abs = abs(dif_temperatura);

  vazao_abs = abs(dif_vazao);

  corrige_temperatura();

  atualiza_motores();

  if(temp_abs>4&&flag_T==1)
  {
    chuveiro_acionamento();
    controle_temperatura();

    corrente = ACS712_30A_AC();//Leitura de corrente na AN4

    flag_T=0;
  }

  //Controla a vazão para o caso das mesmas estarem com 5% de diferença
  if(vazao_abs>5&&flag_V==1)
  {
    chuveiro_acionamento();
    controle_vazao();

    corrente = ACS712_30A_AC();//Leitura de corrente na AN4

    flag_V=0;
  }

  switch(gravado_eeprom)
  {
    case 0: break;
    case 1: gravado_eeprom=0;
          break;
  }
```

```
      printf(lcd_putc,"\fQ=%f%c  F=%f%c\nC:%fA",  agua_consumida,223,  temp_M,223,
tempo_BANHO);
      delay_ms(150);

    }//Fim da rotina do acionamento do sistema por Potenciômetro

    disable_interrupts(INT_TIMER1);
    clear_interrupt(INT_TIMER1);
    count=0;

    set_pwm2_duty(1023); //Impede chuveiro de ser acionado

    output_high(EN); //Habilita os motores

    if(motor1_min==0)
    {
      fechar_quente(MAX_ROTACAO);
    }

    if(motor2_min==0)
    {
      fechar_frio(MAX_ROTACAO);
    }

    output_low(EN); //Desabilita os motores

    printf(lcd_putc,"\fChuveiro                    OFF\nTempo:%f",tempo_BANHO);//Q=%f%c
F=%f%c\nC:%fA", agua_consumida,223, temp_M,223, tempo_BANHO);//,045);
    delay_ms(150);

    if(gravado_eeprom==0)
    {
      temperatura_ao_fim = (int8) temperatura;
      write_eeprom(ultima_temperatura,temperatura_ao_fim);
    }

    CONECTA_AO_WEBSITE();
    ENVIAR_EPS01();

  }  //Fim da rotina de execução do programa
}  //Fim da rotina MAIN
```

REFERÊNCIAS

CCS INC. **CCS C Compiler: PCB, PCM, PCH and PCD**. 2019, ALL RIGHTS RESERVED. Copyright Custom Computer Services, Inc..

CORDEIRO, Fillipe. App Inventor: Guia de criação de Apps. **AndroidPro**, 2018. Disponível em: https://www.androidpro.com.br/blog/desenvolvimento-android/app-inventor/. Acesso em: 08/07/2019.

DEMETRAS, Ezequiel. Módulo ACS712 – Medindo Corrente Elétrica Alternada e Contínua com Arduino. **Vida de Silício**, 2018. Disponível em: https://portal.vidadesilicio.com.br/acs712-medindo-corrente-eletrica-alternada-continua/. Acesso em: 16/05/2019

DEVMEDIA. Uma introdução ao JSON. **DEVMEDIA**, 2019. .https://www.devmedia.com.br/json-tutorial/25275. Acesso em: 07/07/2019

FITZGERALD, A. E.; KINGSLEY JR, Charles; USMANS, Stephen D.. **Máquinas Elétricas.** 6ª ed. São Paulo: McGraw-Hill, 2006.

Max Loarder por E.E.Tools. **FDM lib**, 2019. Disponível em: https://pt.freedownloadmanager.org/Windows-PC/Max-Loader-GRATUITO.html. Acesso em: 03/08/2019.

NETO, Benjamin Batista de; MONTEIRO, Priscila de França; QUEIROGA, Sandro Lino de. **Aplicabilidade dos Microcontroladores em Inovações Tecnológicas.** In: Congresso Norte Nordeste de Pesquisa e Inovação, 7°, 2012, Palmas-TO. p. 4-5.

PEREIRA, Fábio. **Microcontroladores PIC: Programação em C.** São Paulo: Érica, 2003.

SACCO, Francesco. Comunicação SPI – Parte 1. **Embarcados**, 2014. Disponível em: https://www.embarcados.com.br/spi-parte-1/. Acesso em: 20/05/2019.

SENSORES ANALÓGICOS. Disponível em: <https://automacoes.net/2008/11/20/sensores-analogicos/>. Acesso em 18/05/2019.

Significado de Internet das coisas. **Significados**, 2016. Disponível em: https://www.significados.com.br/internet-das-coisas/. Acesso em: 03/08/2019.

SOLETROL, Aquecedores Solares de Água. **História do Aquecedor Solar.** Disponível em: https://www.soletrol.com.br/extras/historia-do-aquecedor-solar/ Acesso em: 15/05/2019.

SOUZA, David José. **Desbravando o PIC: ampliando e atualizado para PIC 16F628A.** 11ª ed. São Paulo: Érica, 2007.

SOUZA, David José; LAVINIA, Nicolás César. **Conectando o PIC 16F877A: Recursos Avançados.** 1ª ed. São Paulo: Érica, 2003.

ZANCO, Wagner da Silva. **Microcontroladores PIC16F628A/648A: uma abordagem prática e objetiva.** 1ª ed. São Paulo: Érica, 2005.