UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
GRADUAÇÃO NO CURSO DE ENGENHARIA DE COMPUTAÇÃO

REGULADOR DE TEMPERATURAS PARA CHUVEIROS ELÉTRICOS

TRABALHO DE GRADUAÇÃO

Lucian Ribeiro da Silva

Santa Maria, RS, Brasil

2014

REGULADOR DE TEMPERATURAS PARA CHUVEIROS ELÉTRICOS

Lucian Ribeiro da Silva

Trabalho de Graduação apresentado ao curso de Engenharia de Computação da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para a obtenção do grau de

Engenheiro de Computação

Orientador: Prof. Dr. José Eduardo Baggio

Santa Maria, RS, Brasil

2014

Silva, Lucian Ribeiro da

REGULADOR DE TEMPERATURAS PARA CHUVEIROS ELÉTRICOS / por Lucian Ribeiro da Silva. – 2014.
63 f.: il.; 30 cm.

Orientador: José Eduardo Baggio
Monografia (Graduação) - Universidade Federal de Santa Maria, Centro de Tecnologia, curso de Engenharia de Computação, RS, 2014.

1. Automação. 2. Arduino. 3. Chuveiro. I. Baggio, José Eduardo. II. Título.

Universidade Federal de Santa Maria
Centro de Tecnologia
Graduação no curso de engenharia de computação

A Comissão Examinadora, abaixo assinada,
aprova o Trabalho de Graduação

REGULADOR DE TEMPERATURAS PARA CHUVEIROS ELÉTRICOS

elaborado por
Lucian Ribeiro da Silva

como requisito parcial para obtenção do grau de
Engenheiro de Computação

COMISSÃO EXAMINADORA:

José Eduardo Baggio, Dr.
(Orientador)

Cesar Augusto Prior, Dr. (UFSM)

Carlos Henrique Barriquello, Dr. (UFSM)

Santa Maria, 16 de janeiro de 2014.

AGRADECIMENTOS

À minha família e amigos pelo apoio e incentivo durante minha trajetória no curso.

Ao professor José Eduardo Baggio pela paciência e orientação durante a implementação do projeto.

Aos professores pelo conhecimento transmitido durante o curso.

Aos funcionários do NUPEDEE pelo empréstimo de ferramentas e instalação da bancada onde os testes experimentais foram realizados.

À Franciele Rovasi pelo incentivo e paciência durante o projeto.

Ao colega Thiago Biazus pelo empréstimo do Arduino.

À Dona Lúcia pelos conselhos e apoio nas horas difíceis.

À todos que contribuiram de forma direta ou indiretamente para a conclusão do trabalho.

“Uma longa viagem começa com um único passo”
— LAO-TSÉ

RESUMO

Trabalho de Graduação
Graduação no curso de engenharia de computação
Universidade Federal de Santa Maria

REGULADOR DE TEMPERATURAS PARA CHUVEIROS ELÉTRICOS
AUTOR: LUCIAN RIBEIRO DA SILVA
ORIENTADOR: JOSÉ EDUARDO BAGGIO
Local da Defesa e Data: Santa Maria, 16 de janeiro de 2014.

O projeto consiste na implementação de um regulador de temperaturas para chuveiros elétricos onde o usuário irá escolher a temperatura desejada da água e o sistema irá realizar o controle da potência do chuveiro até que a temperatura escolhida seja atingida.

O regulador faz uso de um sensor de temperatura, componentes eletrônicos e um microcontrolador. Foi utilizado um controlador Proporcional Integral, que leva o erro de regime permanente para zero. Além disso, devido à presença de atraso de transporte, foi utilizado um Preditor de Smith, que tem a principal vantagem de eliminar o tempo morto da malha de controle, melhorando a eficiência do controlador utilizado.

Com o regulador de temperaturas, a temperatura do banho fica regulada, proporcionando um conforto, independente de haver variação de temperatura na água que alimenta o chuveiro, além de reduzir os despercícios de recursos como água e energia.

Palavras-chave: Automação. Arduino. Chuveiro.

ABSTRACT

Undergraduate Final Work
Computer Engineer
Federal University of Santa Maria

TEMPERATURE REGULATOR FOR ELETRIC SHOWERS

AUTHOR: LUCIAN RIBEIRO DA SILVA
ADVISOR: JOSÉ EDUARDO BAGGIO
Defense Place and Date: Santa Maria, January 16st, 2014.

The project consists in implementing a temperature regulator for eletric showers where the user will choose the desired water temperature and the system will perform the power control of the shower until the selected temperature is reached.

The regulator uses a temperature sensor, electronic components, and a microcontroller. An Integral Proportional controller, which takes the steady state error to zero, was used. Furthermore, due to the presence of transport delay, it was used a Smith Predictor which has the main advantage of eliminating the dead time of the control loop, improving the efficiency of the regulator used.

With the temperature controller, the temperature of the bath is regulated, providing comfort, regardless of whether the temperature variation in the water feeding the shower, and reduce waste of resources like water and energy.

Keywords: Automation, Arduino, Shower.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ANEXOS

LISTA DE ABREVIATURAS E SIGLAS

I/O	*Input/Output*
A/D	Analógico/Digital
RISC	*Reduced Instruction Set Computing*
EEPROM	*Electrically-Erasable Programmabl eRead Only Memory*
SRAM	*Static Random Access Memory*
USART	*Universal Synchronous Asynchronous Receiver Transmitter*
SPI	*Serial Peripheral Interface*
PWM	*Pulse-Width Modulation*
IDE	*Integrated Development Environment*
USB	*Universal Serial Bus*
MT1	*Main Terminal 1*
MT2	*Main Terminal 2*
TRIAC	*Triode for Alternating Current*
PI	Proporcional Integral

SUMÁRIO

1 INTRODUÇÃO

Com a evolução da tecnologia da informação, a automação residencial vem deixando de ser um assunto de filmes de ficção científica e se tornando parte de nossas vidas. Ela é capaz de nos fornecer comodidade, praticidade, qualidade de vida e muitos outros atributos. Intensidade das luzes de uma sala, temperatura de ambientes, acendimento de um forno, hoje em dia, podem ser controlados por um simples comando em um *notebook*, *Smartphone* ou um *tablet*, trabalhando em conjunto de um *hardware* e um *software* adequados. Um aparelho imprescindível em qualquer residência , e que poderia ter um foco maior de pesquisas na área de automação, é o chuveiro elétrico. Ele funciona de forma bem simples: A água ao passar pela resistência, localizada na parte interna do chuveiro, aquece-se devido à alta temperatura da resistência, ocasionada pela passagem de corrente elétrica (efeito joule). Um dos problemas encontrados nesse sistema de chuveiro elétrico convencional é o tempo que se leva para ajustar a temperatura da água de acordo com nosso gosto. Normalmente há a necessidade de se regular a vazão da água e a posição da estação do chuveiro (geralmente verão, inverno e desligado). Esse ajuste de temperatura leva em torno de dois minutos, o que gera, além de certo desconforto, desperdício de recursos ambientais como água e energia.

Visando a aumentar o nível de conforto e reduzir o tempo gasto para o ajuste, o presente trabalho tem como proposta a implementação de um regulador de temperatura para chuveiros elétricos, onde o usuário irá entrar com a temperatura desejada através de uma interface com botões e displays e o microcontrolador irá realizar o controle do sistema para que a temperatura da água atinja o valor desejável.

O chuveiro utilizado no trabalho é um ThermoSystem modelo 01 que utiliza um sistema de controle de temperatura controlado pela alteração do ângulo de disparo do sinal. Esta alteração é feita pelo usuário ao girar a haste localizada na parte inferior do chuveiro.

O sistema convencional será substituído por um sistema microcontrolado para que a temperatura final seja igual, ou muito próxima à temperatura escolhida.

Um sensor de temperatura será utilizado para verificar se a temperatura da água condiz com a temperatura desejada e caso não sejam iguais será calculado o erro entre elas. Este erro servirá de parâmetro de entrada para o controlador Proporcional Integral.

O parâmetro de saída do controlador fornecerá um dado do qual possibilitará o cálculo de tempo de disparo do TRIAC. Este tempo será calculado inúmeras vezes até que o sistema

atinja a estabilidade.

1.1 Motivação

As motivações mais importantes para a realização deste trabalho foram as seguintes:

- Aplicação prática de assuntos abordados durante o curso de Engenharia de Computação como microcontroladores, sistemas de controle, circuitos eletrônicos;
- Economia e otimização do uso de recursos ambientais, como água e energia elétrica;
- Automatização de um componente comum a praticamente todas as residências;

1.2 Objetivos

Os objetivos principais deste trabalho são:

- Diminuição do tempo de ajuste de temperatura da água do banho;
- Economia de água e energia;
- Conforto na escolha da temperatura precisa em graus Celsius;

1.3 Estrutura do trabalho

Esta seção irá comentar sobre a estrutura dos capítulos deste trabalho.

O capítulo 2 irá apresentar informações sobre os assuntos que foram estudados e pesquisados em diversas fontes e os quais ofereceram conhecimento para que a implementação do projeto fosse realizada.

O capítulo 3 irá tratar dos procedimentos efetuados para o desenvolvimento do regulador.

No capítulo 4 serão mostrados as etapas que foram realizadas para que a montagem do projeto se tornasse possível.

O capítulo 5 irá apresentar os resultados obtidos, através de gráficos construídos pelo *software* matemático Matlab.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão apresentados as informações adquiridas através de pesquisas em diversas fontes, bem como os conhecimentos obtidos durante o curso de Engenharia de Computação os quais tornaram possível a implementação deste trabalho. Primeiramente será comentado sobre o funcionamento e os componentes de um chuveiro convencional e após será discutido informações sobre os dispositivos e componentes utilizados no presente trabalho.

2.1 Funcionamento de um chuveiro elétrico

O funcionamento de um chuveiro elétrico, independente do modelo ou marca, é praticamente o mesmo, na maioria dos modelos. Ao abrir o registro a água irá passar pelo diafragma pressionando-o para que haja contato entre dois terminais possibilitando assim o fornecimento de energia elétrica para o chuveiro. Com o contato feito, a corrente elétrica passa pela resistência do chuveiro esquentando-a e consequentemente esquentando a água. Este mecanismo impede que haja passagem de corrente elétrica na resistência do chuveiro caso o registro seja fechado, o que ocasionaria um superaquecimento e o comprometimento do funcionamento do chuveiro. Este fenômeno que transforma energia elétrica em energia térmica é denominado efeito Joule.

2.1.1 Componentes de um chuveiro

Os componentes principais de um chuveiro elétrico, para seu funcionamento correto são o espalhador, o diafragma e a resistência elétrica. Em duchas eletrônicas existe o componente interno chamado TRIAC do qual seu funcionamento também será comentado no presente trabalho.

2.1.2 Resistência

A resistência elétrica de um chuveiro é composta de um fio de Nicromo (material condutor constituído de Niquel e Cobre) enrolado. A corrente elétrica passa por este material e, desta forma, aquece a água para o banho. A figura 2.1 mostra um exemplo de uma resistência elétrica.

Figura 2.1 – Resistência de um chuveiro

2.1.3 Diafragma

O diafragma é um componente que funciona como uma chave para a alimentação do chuveiro. Quando o registro é aberto, a água ao passar pelo chuveiro pressiona o diafragma fazendo com que dois pontos entrem em contato conectando o chuveiro a rede elétrica e, dessa forma, tem-se corrente elétrica circulando pela resistência. Ele é uma peça crucial, pois ao desligar-se o chuveiro a tensão no chuveiro é cortada e assim não há mais passagem de corrente elétrica na resistência evitando acidentes devido ao superaquecimento da resistência elétrica.A figura 2.2 ilustra o funcionamento de um diafragma.

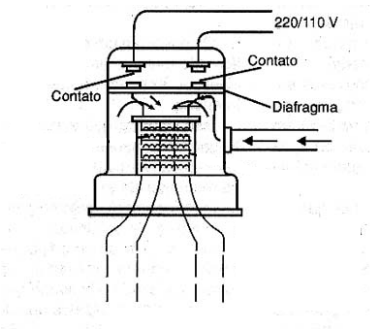

Figura 2.2 – Diafragma

2.1.4 Espalhador

O espalhador é um elemento do chuveiro onde há a saída da água aquecida pela resistência. Ele possui pequenos orifícios que dificultam a passagem da água, funcionando como um limitador de vazão e fazendo com que o diafragma fique pressionado mantendo dois pontos em contato para que haja passagem de corrente elétrica.A figura 2.3 mostra o espalhador de um

chuveiro convencional.

Figura 2.3 – Espalhador

2.2 Microcontrolador

Um microcontrolador é um dispositivo presente em inúmeros aparelhos eletrônicos do dia-a-dia, como celulares, microondas, geladeiras, dentre outros. Ele é constiuído de um processador, memória de programa, memória de dados, pinos de *I/O* (*Input/Output*) e mais alguns periféricos como *timers*, conversores A/D (analógico/digitais), dentre outros, tudo em apenas um encapsulamento, tornando assim menor a complexidade do circuito eletrônico que seria utilizado para uma determinada aplicação de controle. Por esse motivo, os microcontroladores são ótimas ferramentas para a implementação de projetos de sistemas embarcados,onde há necessidade de um controle específico e regulado.

2.2.1 Arduino

O Arduino Uno (Uno, 2013) é uma plataforma de prototipagem eletrônica gratuita que utiliza um microcontrolador da família ATmega de 8 *bits*. Ele é capaz de trabalhar com inúmeros tipos de aplicações como sensores, motores, *leds*, dentre outros. No presente trabalho foi utilizado o modelo do Arduino chamado UNO que utiliza o microcontrolador ATmega328.

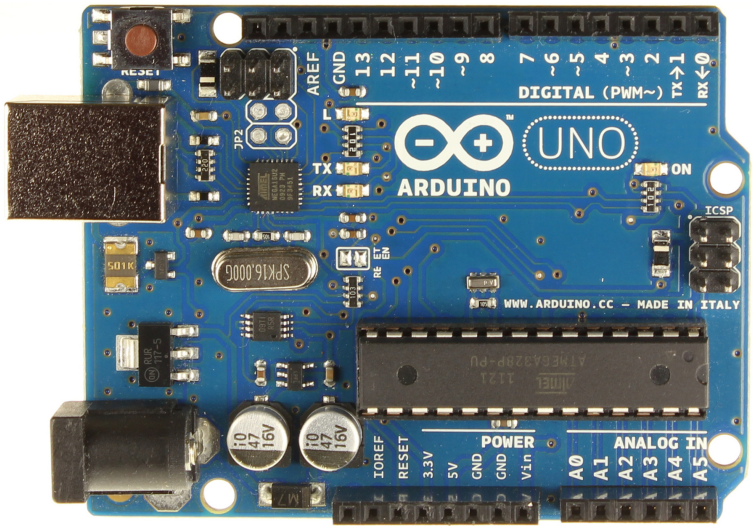

Figura 2.4 – Arduino UNO

2.2.2 ATmega328

O ATmega328 é um microcontrolador, pertencente à família AVR da empresa ATmel que é utilizado no Arduino UNO. Este modelo de microcontrolador possui um processador RISC (*Reduced Instruction Set Computing*) de 8 bits que utiliza uma arquitetura Harvard modificada, possui 32KB de memória flash, 1KB de EEPROM (*Electrically-Erasable ProgrammableRead-Only Memory*), 2KB de SRAM (*Static Random Access Memory*), 32 registradores, três *timers* (temporizadores), uma USART (*Universal Synchronous Asynchronous Receiver Transmitter*), portas para comunicação SPI (*Serial Peripheral Interface*), um conversor AD de 10 bits, saídas PWM (*Pulse-Width Modulation*) dentre outros(ATmega328 Datasheet, 2009). O ATmega328 é utilizado em alguns modelos de Arduino encontrados no mercado. Na tabela 2.1 são mostrados alguns dados do microcontrolador fornecidos pelo fabricante.

Tabela 2.1 – Caracteristicas do Arduino UNO

Microcontrolador	ATmega328
Tensão de operação	5V
Tensão de entrada (recomendado)	7-12V
Pinos digitais de I/O	14 (6 com saída PWM)
Pinos de entrada analógica	6
Corrente por pino de I/O	40 mA
Corrente pelo pino de 3,3V	50 mA
Memória Flash	32KB (ATmega328)
SRAM	2 KB (ATmega328)
EEPROM	1KB (ATmega328)
Velocidade do Clock	16 MHz

O Arduino possui uma IDE (*Integrated Development Environment*) que é uma aplicação escrita em linguagem java onde pode-se desenvolver códigos em linguagem C/C++ para controlar os pinos do microcontrolador. Esta IDE está disponível para download gratuitamente no site do fornecedor (Arduino,2013).

2.2.2.1 Pinagem

O microcontrolador ATmega328 possui, ao total, 28 pinos dos quais são divididos da seguinte forma:

- 14 pinos digitais de entrada e saída (configuráveis).
- 6 pinos de entrada analógica ou entrada e saída digital (configuráveis).
- 5 pinos de alimentação (terra, 5V, referência analógica).
- 1 pino de reset.
- 2 pinos para conexão de um cristal oscilador.

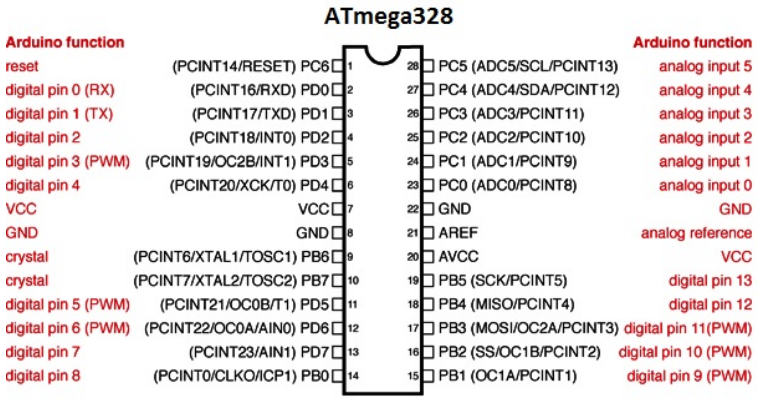

Figura 2.5 – Pinagem do ATmega328

Os pinos digitais podem ser configurados como entrada ou saída e possuem somente dois estados: ligado (*HIGH*) ou desligado(*LOW*). Quando o pino é configurado como ligado ele apresenta uma tensão de saída de 5V e quando configurado como desligado apresenta na saída a tensão de zero Volts. Esses pinos podem ser utilizados para várias aplicações como acendimento de *leds*, operações com portas lógicas, acionamento de relés, dentre outros.

Os pinos analógicos são utilizados na maioria das vezes para leituras de sensores e transdutores que convertem grandezas físicas em tensão elétrica, a qual pode ser medida pela entrada analógica no microcontrolador.

2.2.2.2 Pinos especiais

Além dos pinos analógicos e digitais, o ATmega 328 possui uma quantidade de pinos com características especiais que podem ser configuradas através da programação. São eles:

- PWM (Pulse Width Modulation – Modulação por Largura de Pulso): é gerado um sinal pulsante, com uma determinada frequência, onde se define o tempo em que o sinal fica em nível alto e o tempo que o sinal fica em nível baixo. Desta forma, variando-se a razão cíclica, que é a razão entre tempo em nível alto e o período do sinal, pode-se controlar o valor médio do sinal PWM. Os pinos que possuem saída PWM são: pinos 3,5,6,10 e 11.
- Portal Serial USART: possibilita o microcontrolador se comunicar com um computador, bluetooth ou outro dispositivo do gênero, através de envio e recebimento de dados no formato serial assíncrono (USART). Os pinos que possuem comunicação serial são: pino 0 (recebe dados) e pino 1 (envia dados).
- Interrupção externa: Existe a possibilidade de se programar um pino para que quando ativado force o processador a parar o que está fazendo e realizar outras operações pré-programadas. O ATmega328 possui 2 pinos (2 e 3) para interrupções externas. São bastante úteis para se economizar processamento em diversas aplicações.

2.2.3 Vantagens

As grandes vantagens de se utilizar a plataforma arduino são:

- Ser uma ferramenta *open-source* (*Software/Hardware*).
- A gama de utilizadores da plataforma é grande, tendo assim uma grande quantidade de informações na rede, permitindo uma constante atualização sobre as inovações.
- Não tem a necessidade de operação em conjunto com um computador (*Standalone*).
- Possibilidade de aumentar a capacidade de aplicações com a utilização de SHIELDS (placas de circuito impresso que podem ser plugadas ao pinos do microcontrolador possibilitando mais funções, como controlar motores e utilização de redes sem fio).

2.2.4 Conversor A/D

Um conversor A/D é um circuito que realiza a conversão de dados analógicos, de tensão ou corrente, para um valor digital com *n bits* Este circuito é bastante útil para diversos tipos de aplicações, como aquisição de dados de sensores, voz, áudio, entre outros.

O Arduino UNO possui pinos que utilizam um conversor A/D de 10 bits, ou seja, a tensão de referência é dividida por 1024 unidades (2^{10}). Por exemplo, suponha-se que a tensão de referência seja de 5V, teremos então 1024 valores entre a tensão de 0 à 5V, ou seja:

$$\text{Resolução} = 5V/1024 \approx 4,9mV$$

Assim, a cada aumento de 4,9mV na entrada do conversor A/D, tem-se o aumento de uma unidade digital.

2.2.5 Comunicação serial

A comunicação serial trata-se de um envio de forma sequencial de bits, por um barramento, em um certo intervalo de tempo, ou seja, de forma sequencial. Esse meio de comunicação é utilizado em diversos dispositivos como a USB (*Universal Serial Bus*),*FireWire*, RS-232 dentre outros.

A Arduino UNO possui uma porta de comunicação nos pinos digitais 0 (recebimento de sinais digitais) e 1 (Envio de sinais digitais). A IDE do Arduino oferece uma aplicação bastante interessante e útil chamada de *Serial Monitor* que quando utilizada mostra na tela os valores das portas digitais e analógicas.

2.2.6 Timers

Os *timers* são contadores internos do microcontrolador que podem ser limitados a um certo valor. A grande utilidade dos timers é de chamar uma determinada rotina num intervalo de tempo configurado (interrupção por tempo), como por exemplo realizar a leitura de um valor analógico em alguma porta do microcontrolador de 10 em 10 segundos.

Na tabela 2.2 são mostrados os timers do microcontrolador ATmega328 e suas características.

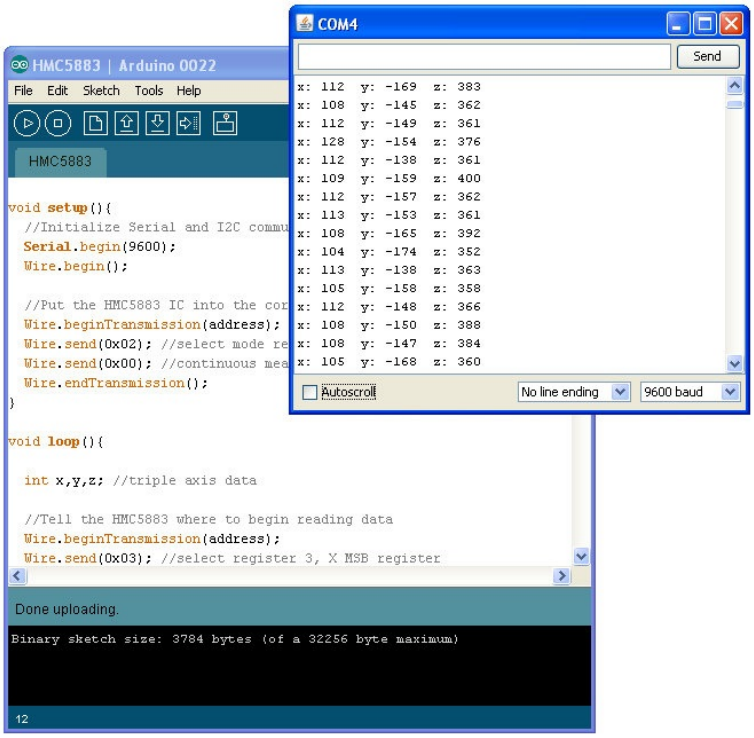

Figura 2.6 – Serial Monitor

Tabela 2.2 – Timers do ATmega328

Timer0	temporizador de 8 bits que é utilizado nas funções millis() e micros()
Timer1	temporizador de 16 bits
Timer2	temporizador de 8 bits

2.2.7 Interrupção

Uma interrupção é um sinal do *hardware* para mandar o processador suspender a tarefa que está executando no momento e executar outra determinada tarefa e, após executada, o processador retornar a tarefa inicial sem a perda de informações(TANENBAUM, 1995).

O ATmega328 possui dois métodos diferentes de chamadas de interrupção: externa ou por tempo. A interrupção externa pode ser utilizada nos pinos digitais 2 e 3. A tabela 2.3 mostra os modos de chamada de interrupção que esses pinos utilizam.

Tabela 2.3 – Modos de interrupção externa

Tipo de Interrupção	**Descrição**
Rising	Chama rotina quando há mudança de nivel baixo para nivel alto
Falling	Chama rotina quando há mudança de nivel alto para nivel baixo
Change	Chama rotina quando há qualquer mudança de nivel.
Low	Chama rotina quando há nível baixo

A interrupção por tempo trata-se de uma chamada de rotina repetidas vezes em um tempo pré configurado. Para este tipo de chamada são utilizados os *timers* internos do microcontrolador (*Timer0, Timer1* ou *Timer2*).

2.3 Sensor de temperatura

O sensor que foi utilizado no presente trabalho para a medição da temperatura da água foi o sensor LM35 fabricado pela *National Semicondutor*. Trata-se de um sensor de precisão com saída linear relativa à temperatura (Datasheet LM35, 2013). A figura 2.7 mostra o sensor e seus respectivos terminais.

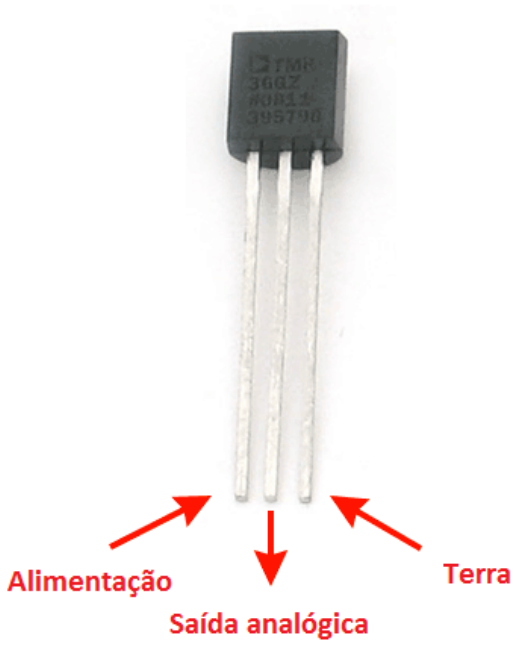

Figura 2.7 – Sensor de temperatura

O sensor possui três terminais: Alimentação, terra e saída analógica. O sensor pode ser alimentado com uma tensão contínua que varia de 4 à 30V e é capaz de medir temperaturas que variam de $-55\,°C$ à $150\,°C$. A precisão do sensor é de $\pm 0,25\,°C$ para medições de temperatura ambiente e de $\pm 0,75\,°C$ para medições entre $-55\,°C$ à $150\,°C$. A sua saída varia de 10mV para cada grau Celsius de temperatura, não havendo a necessidade de calibração (Datasheet LM35, 2013).

O sensor possui baixa impedância de saída, tensão linear e calibração inerente precisa, tornando assim a interfaceamento de leitura de temperatura simples e barato (LM35 Datasheet,

2013).

2.4 TRIAC

O TRIAC (*Triode for Alternating Current*) é um dispositivo de controle de corrente alternada. Ele é bastante semelhante ao SCR (*Silicon Controlled Rectifier*), porém ele possui a característica de conduzir corrente elétrica em dois sentidos. Ele é bastante utilizado quando há a necessidade de se controlar a potência aplicada a uma determinada carga elétrica. Ele é constituído por três terminais, MT1 (*Main Terminal 1*), MT2 (*Main Terminal 2*) e *Gate*. A figura 2.8 representa o símbolo de um TRIAC bem como seus terminais.

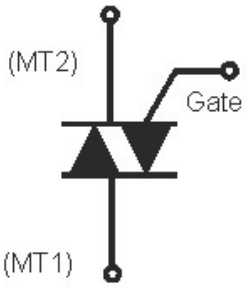

Figura 2.8 – TRIAC

Um TRIAC pode ser disparado por uma tensão positiva ou negativa aplicada ao terminal de disparo (Gate), e a corrente normalmente na casa dos miliamperes. Ao ser disparado, o TRIAC conduz corrente elétrica até que este valor de corrente se reduza para um valor muito baixo (valor de corrente próximo a zero no final do ciclo de um sinal de corrente alternada). Dessa forma, o TRIAC se torna um componente de grande utilidade para correntes alternadas pois permite controlar altas potências a partir de circuitos acionados por correntes muito baixas.

O TRIAC também possibilita o controle de ângulo de fase de um sinal por simplesmente aplicar-se um pulso em um determinado instante do ciclo de corrente alternada. A figura 2.9 mostra o funcionamento do ângulo de fase.

Quanto mais próximo do início do ciclo o disparo for dado, maior a potência será aplicada na carga.

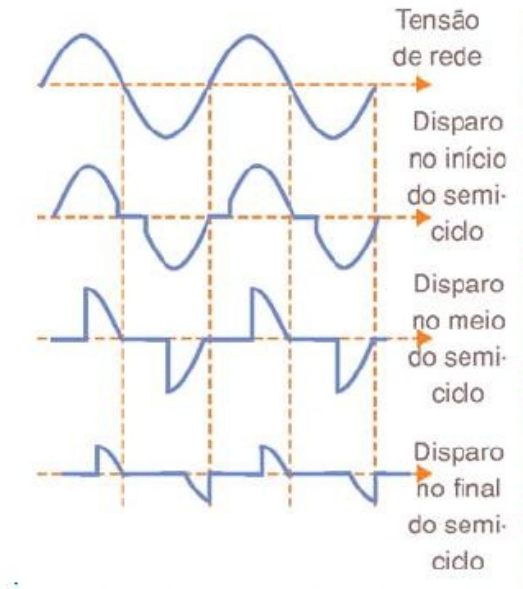

Figura 2.9 – Controle de ângulo de fase

2.5 Optoacoplador

O optoacoplador MOC 3020 é um dispositivo isolador que é constituído de um *led* infravermelho e um fotodiac. Com este dispositivo se torna possível o controle de uma alta tensão a partir de uma baixa tensão. A figura 2.10 representa o símbolo de um optoacoplador.

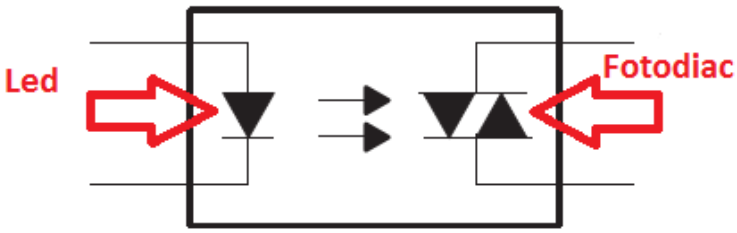

Figura 2.10 – Optoacoplador

A corrente elétrica ao passar pelo LED infravermelho aciona um fotodiac do outro lado do circuito permitindo a passagem de corrente elétrica e dessa forma isolando o lado onde há uma alta tensão do lado onde se tem uma baixa tensão.

2.6 Sistemas de controle

Um sistema de controle é um método utilizado para controlar o comportamento de um determinado sistema onde a saída seja dependente da entrada. Um sistema pode ser interpretado como uma caixa preta com uma entrada e uma saída onde sabemos somente a relação entre elas (BOLTON, 1995). A figura 2.11 mostra a ilustração da interpretação de um sistema qualquer.

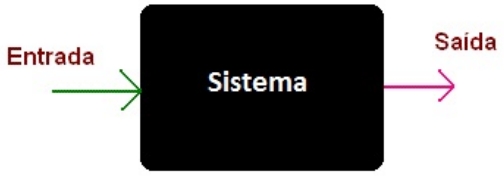

Figura 2.11 – Caixa preta

Para praticamente todos os tipos de sistemas é necessário um estudo de controle para que haja um funcionamento de acordo com o que se quer e seja alcançada a estabilidade do mesmo. Sem esse estudo o sistema pode funcionar de forma inadequada e nunca convergir para um determinado valor. A figura 2.12 mostra um gráfico como exemplo da estabilidade de um sistema em função do tempo, onde o *set point* é o valor que deseja-se na saída do sistema.

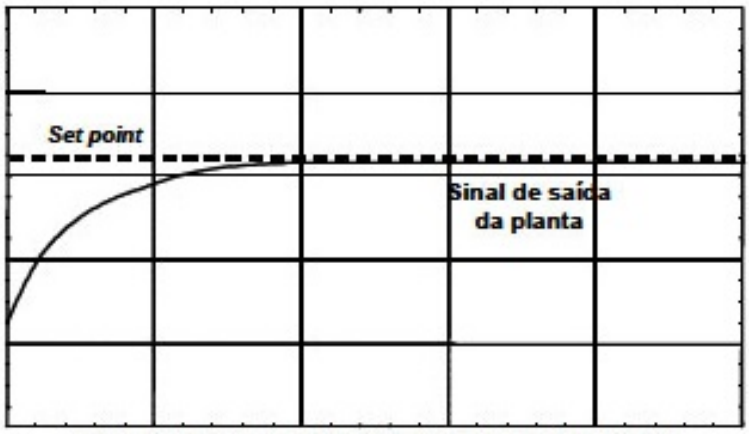

Figura 2.12 – Estabilidade de um sistema

A vantagem de se estudar sistemas de controle é que inúmeros sistemas com propriedades completamente diferentes podem ter a mesma relação de entrada e saída, como por exemplo a resposta de um capacitor em série com um resistor a uma determinada tensão. Este sistema tem o mesmo tipo de relação com um recipiente cheio de um liquido do qual é aplicado uma determinada entrada de calor (BOLTON, 1995).

2.6.1 Transformada de Laplace

Uma ferramenta matemática bastante útil no estudo de sistemas de controle é a transformada de Laplace. Esta ferramenta tem o poder de transformar equações diferenciais em equações algébricas das quais são mais facilmente solucionáveis. Operações como integrais e derivadas podem ser substituídas por equações algébricas no plano complexo. Quando aplica-se a transformada de Laplace a um sistema ele sai do domínio do tempo e passa a ser analisado no domínio s (domínio da frequência complexa) (BOLTON, 1995). Além de facilitar bastante os cálculos, a análise neste domínio nos fornece informações sobre o comportamento do sistemas em regime transitório (antes de estabilizar) e em regime permanente (após estabilizar).

A transformada de Laplace é aplicada da seguinte forma: multiplica-se cada termo na equação por e^{-sT}, onde s é uma constante com unidade $\frac{1}{tempo}$ e então integrar cada termo em relação ao tempo de zero até infinito. O resultado é o que chama-se de transformada de Laplace. Portanto, a transformada de Laplace de algum termo, que é função do tempo, é:

$$F(s) = \int_0^\infty f(t)e^{-sT}dt,$$

onde $f(t)$ é uma função no domínio do tempo e $F(s)$ é a transformada de Laplace desse termo no domínio de s (BOLTON, 1995).

2.6.2 Sistemas de controle em malha aberta

Os sistemas de controle em malha aberta são sistemas onde a saída não influencia no controle do sistema, ou seja, não é verificado um erro da resposta em relação a entrada. Um exemplo disso seria a máquina de lavar roupas. A máquina não verifica se a roupa está bem limpa no final da lavagem, ela apenas executa suas funções em sequência em um determinado tempo. Logo, não temos uma verificação do sinal de saída (roupa limpa) por um sinal de entrada (roupa limpa) (OGATA, 2003). A figura 2.13 mostra o sistema de controle em malha aberta de uma máquina de lavar roupas.

Este tipo de sistema de controle é normalmente utilizado quando se tem um conhecimento da relação entre a entrada e a saída do sistema e também com a ausência de possíveis distúrbios internos ou externos. O sistema de controle em malha aberta apresenta, normalmente, um erro em regime permanente, ou seja, um erro quando o sistema já está estabilizado.

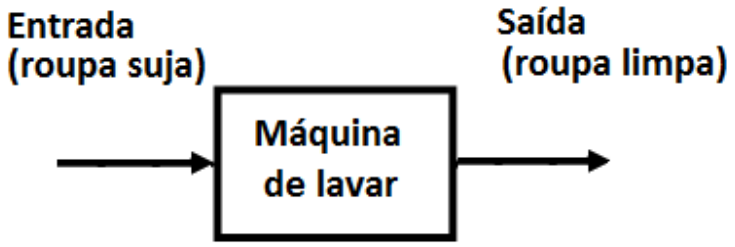

Figura 2.13 – Sistema em malha aberta

2.6.3 Sistemas de controle em malha fechada

Os sistemas de controle em malha fechada são sistemas onde a saída influencia no controle do mesmo. O erro (a diferença entre o valor de saída e o valor de entrada) excita o controlador para que este erro seja eliminado ou minimizado na próxima iteração(OGATA, 2003). Este é um sistema de controle utilizado em vários aparelhos, como por exemplo uma geladeira onde a temperatura é constantemente medida para que quando atinja um certo valor ela se desligue, economizando assim energia elétrica.A figura 2.14 um exemplo de sistema de controle em malha fechada do funcionamento de uma geladeira.

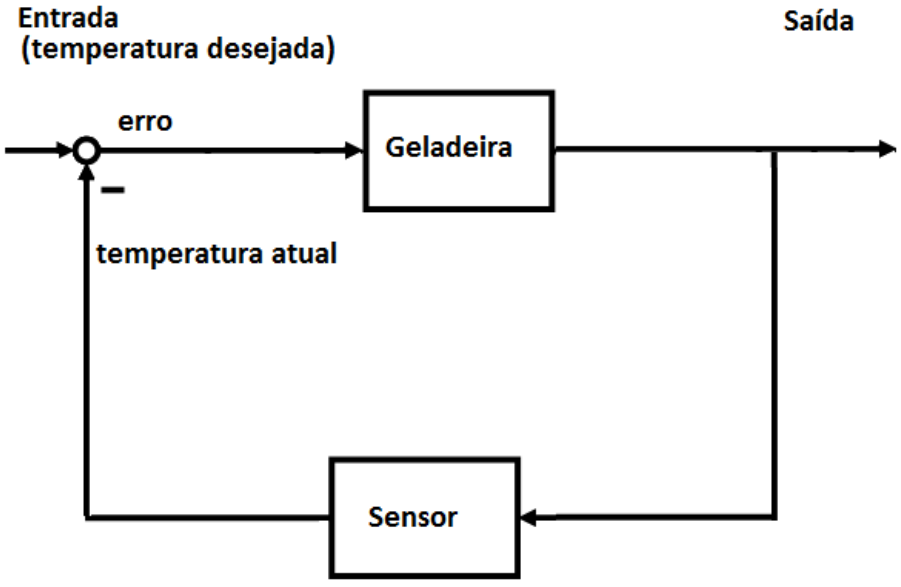

Figura 2.14 – Sistema em malha fechada

2.6.4 Comparação entre os tipos de sistemas de controle

A principal vantagem de um sistema de controle em malha fechada é a possível correção do erro do sistema o que o torna mais estável quando há perturbações externas ou internas. É

possível a utilização de componentes baratos, sem muita exatidão para se conseguir um controle preciso de um determinado processo, fato que se torna impossível na utilização de um sistema de controle em malha aberta. Mas em relação a estabilidade de um sistema, os controles em malha aberta são, normalmente, mais fáceis de serem implementados, pois o controle em malha fechada pode tender a corrigir erros além do necessário ocasionando uma oscilação com amplitude constante ou cada vez maior com o passar do tempo(OGATA, 2003).

2.6.5 Controladores

Os controladores são estratégias utilizadas para se fazer com que um sistema físico atenda as especificações de funcionamento e desempenho determinadas. Basicamente um controlador é um elemento no sistema de controle em malha fechada que tem como entrada o erro do sistema e gera uma saída que se torna a entrada para a planta do sistema (BOLTON, 1995). A figura 2.15 mostra um exemplo de planta realimentada com utilização de um controlador.

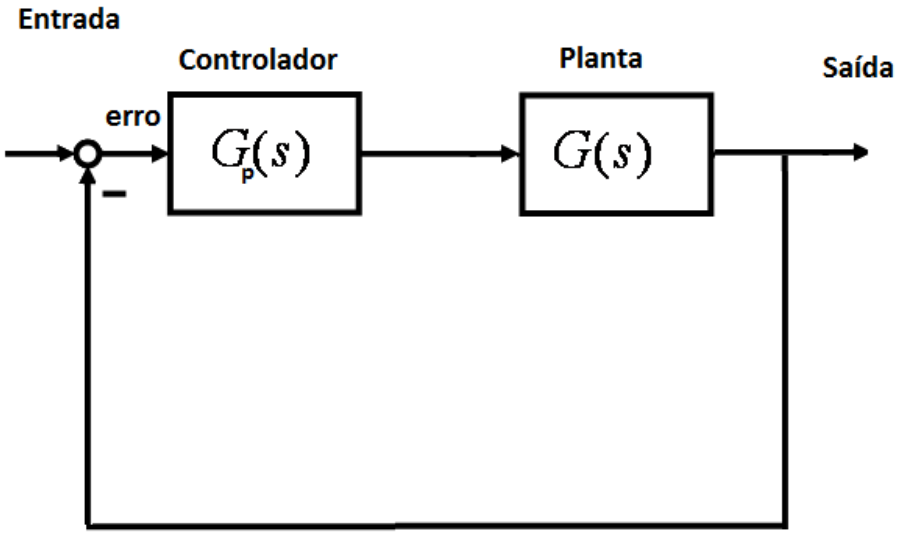

Figura 2.15 – Planta realimentada com controlador

Existem três formas principais de se implementar um controlador que utiliza o parâmetro de erro entre a referência e a saída: proporcional, integral e derivativa.

2.6.6 Controlador proporcional

Um controlador proporcional tem a relação de entrada $e(t)$ e saída $u(t)$ da seguinte forma:

$$u(t) = K_p.e(t)$$

onde pode-se notar que o sinal de saída do controlador é proporcional ao erro do sistema. Passando para o domínio de s com a transformada de Laplace obtemos a função de transferência:

$$\frac{U(s)}{E(s)} = K_p$$

onde Kp é denomidado o ganho proporcional.

Independente do mecanismo, o controlador proporcional nada mais é do que um amplificador com um ganho ajustável. A figura 2.16 representa o diagrama de blocos de um controlador proporcional (OGATA, 2003).

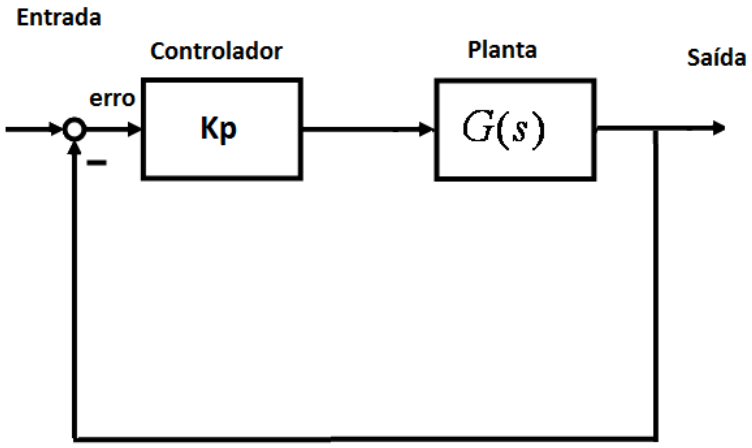

Figura 2.16 – Diagrama de blocos do controlador proporcional

2.6.7 Controlador Integral

A relação de entrada e saída de um controlador integral é dada pela seguinte equação:

$$u(t) = K_i \int_0^t e(t)dt$$

onde Ki é uma constante denominada de ganho integral. A função de transferência de um controlador integral é dada pela equação abaixo:

$$\frac{U(s)}{E(s)} = \frac{K_i}{s}$$

Desta forma, a saída em qualquer instante de tempo é proporcional ao acúmulo de efeitos do erro em instantes anteriores (BOLTON, 1995). A figura 2.17 mostra um diagrama do controlador integral.

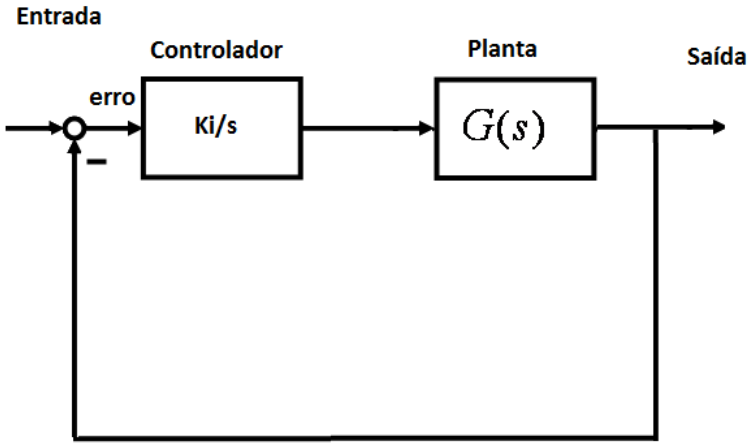

Figura 2.17 – Diagrama de blocos do controlador integral

2.6.8 Controlador Derivativo

A saída de um controlador derivativo é proporcional à taxa de variação do erro, $e(t)$, ou seja:

$$u(t) = K_d \frac{de(t)}{dt}$$

onde K_d é denominado de ganho derivativo.

A função de transferência do controlador derivativo é definida por:

$$\frac{U(s)}{E(s)} = K_d.s$$

Com o controlador derivativo, quando há sinal de erro a saída do controlador já torna-se grande pois ele é proporcional a taxa de variação do sinal de erro e não do erro em si. Isso fornece uma grande correção no erro, entretanto se o erro for um valor constante nenhuma ação será executada pelo controlador (BOLTON, 1995). A figura 2.18 mostra o diagrama de blocos de um controlador derivativo.

2.6.9 Preditor de Smith

Normalmente, em diversos tipos de processos há a presença de um *delay*, ou seja, um determinado tempo de atraso para um processo sentir a variação da entrada na saída. Em sistemas de controle este *delay* é denominado tempo morto e ele existe sempre que há transporte físico de energia ou material. Um exemplo da presença de tempo morto é mostrado na figura 2.19.

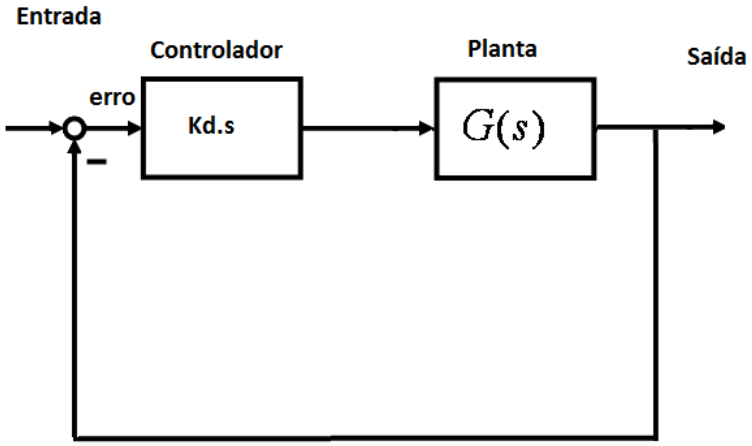

Figura 2.18 – Diagrama de blocos do controlador derivativo

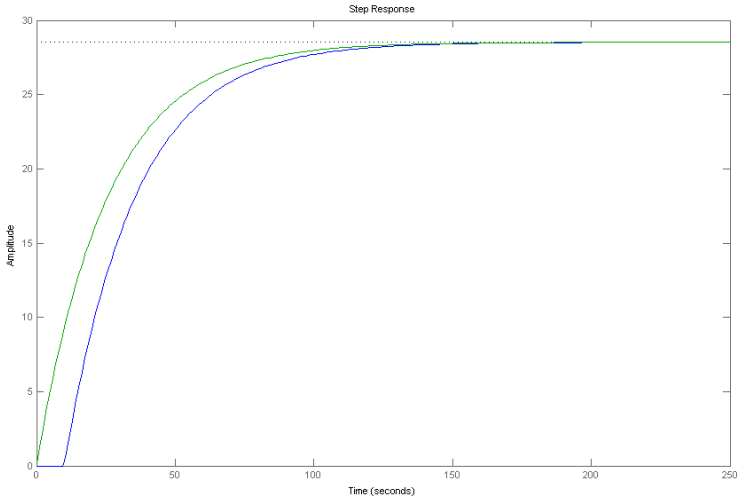

Figura 2.19 – Tempo morto

Pode-se ver que o sistema (linha azul) demorou um certo tempo para a responder a entrada (linha verde)

Quando há a presença de tempo morto em algum tipo de processo há também uma diminuição no desempenho do sistema de controle por realimentação. O ganho controlador tem que ser reduzido em relação ao sistema de controle utilizado caso o sistema não tivesse tempo morto (MACHADO, 2004).

Para driblar esse problema presente em inúmeras aplicações na engenharia, processos industriais, entre muitos outros, O. J. M. Smith (SMITH, 1959) propôs um esquema de controle bastante poderoso para melhorar a eficiência de malhas de controle com presença de tempo morto. Este esquema de controle ficou conhecido como Preditor de Smith. O Preditor de Smith

utiliza um sistema de realimentação interna em malha fechada que elimina o tempo morto do sistema a ser controlado. A figura 2.20 representa o diagrama de blocos de um Preditor de Smith.

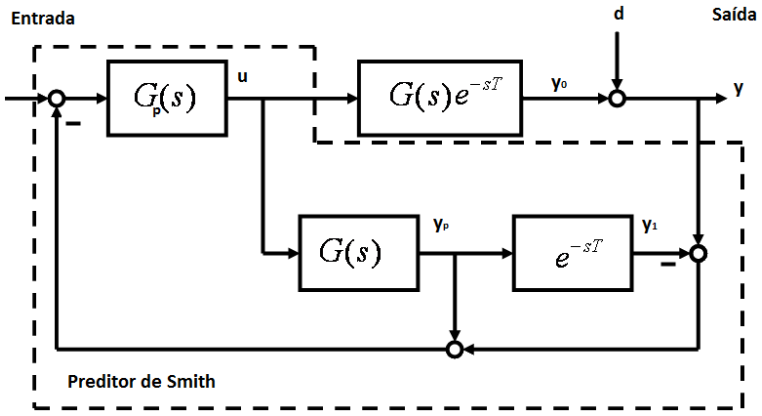

Figura 2.20 – Preditor de Smith

Na figura acima, $G(s)$ é a planta do sistema, e^{-sT} é o atraso do sistema,$G_p(s)$ representa o controlador e d representa um possível distúrbio externo. A equação abaixo mostra a função de transferência resultante da malha fechada.

$$\frac{SAIDA}{ENTRADA} = \frac{G_p(s).G(s).e^{-sT}}{1+G_p(s).G(s)}$$

Dessa maneira consegue-se eliminar o tempo de atraso e^{-sT} da equação característica do sistema e portanto tem-se uma melhora significante no controle do mesmo. Após aplicado o esquema do Preditor de Smith, podemos redesenhar o diagrama de blocos como mostra a figura 2.21.

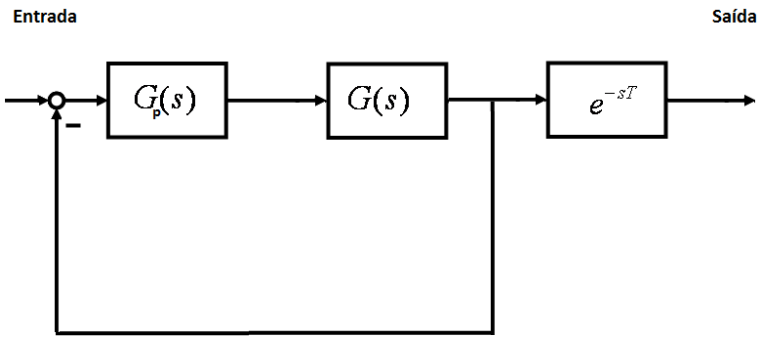

Figura 2.21 – Diagrama de blocos equivalente

Pode-se notar que o diagrama de blocos equivalente do Preditor de Smith coloca o tempo morto para fora da malha fechada o que não afetará a performance dinâmica do controlador.

2.7 Matlab

O *MathWorks* Matlab é um software poderoso destinado à manipulação de matrizes, plotagem de funções, implementação de algoritmos, dentre outros. É considerada uma linguagem de alto nível e possui um ambiente para cálculos, visualização e programação computacional (Matlab, 2013). O Matlab abrange uma gama grande de aplicações como processamentos de sinais, processamento de vídeo e imagens, biologia, economia, entre outros.

Neste trabalho, o Matlab foi utilizado para a obtenção do modelo da planta, simulação da planta e dos controladores, e projeto dos controladores.

3 DESENVOLVIMENTO

Este capítulo irá falar sobre o desenvolvimento do projeto do regulador de temperaturas para chuveiros elétricos. Primeiramente será comentado sobre o funcionamento do circuito interno da ducha ThermoSystem, após serão comentados as etapas realizadas pelo *software* do projeto.

3.1 Ducha ThermoSystem

A ducha eletrônica ThermoSystem foi a primeira ducha no Brasil a possuir um sistema de ajuste de temperatura gradual proporcionando um maior conforto para os usuários. A época de maior produção de duchas pela empresa ThermoSystem é no inverno, onde produção chega a 150.000 unidades por mês (PILATTI, 2012).

Figura 3.1 – Ducha ThermoSystem

O circuito interno da ducha ThermoSystem é bastante simples porém eficiente. Ele é constituído de um potenciômetro, um resistor, dois capacitores em paralelo, um DIAC e um TRIAC. A haste do chuveiro controla o potenciômetro. Este potenciômetro, ao aumentar ou diminuir o valor da resistência, regula o tempo de carga dos capacitores em paralelo. Os capacitores, por sua vez, ao atingirem o valor de tensão de 24V acionam o DIAC que consequentemente acionará o TRIAC. O TRIAC ao ser acionado fecha o circuito, possibilitando a passagem de corrente elétrica pela carga do chuveiro (resistência). Dessa maneira é possível regular o ângulo de disparo de um sinal possibilitando a aplicação de potências diferentes em uma carga. A figura 3.2 representa o circuito interno da ducha ThermoSystem.

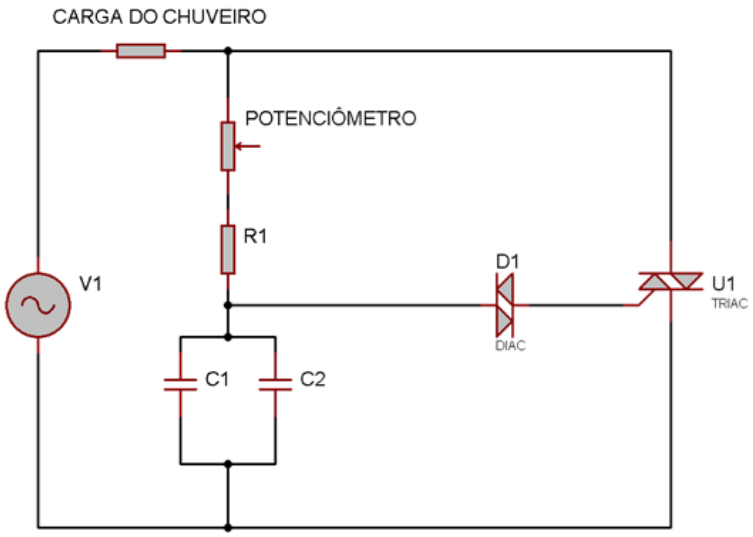

Figura 3.2 – Circuito interno da ducha ThermoSystem

3.2 Etapas

Nesta seção serão comentadas as etapas e as implementações realizadas pelo *software* do projeto do regulador de temperaturas. A figura abaixo mostra o fluxograma do *software* utilizado.

3.2.1 Detecção de passagem por zero

A detecção de passagem por zero do sinal CA (corrente alternada) foi necessário pois somente dessa forma se teria um controle preciso do ângulo de fase do sistema. A implementação da detecção de passagem por zero foi realizada com o auxílio de um transformador de 6V+6V, um retificador de onda completa, um divisor resistivo e um pino de entrada analógica do microcontrolador ATmega328. O sinal CA da saída do transformador, após ser retificado, passa pelo divisor resistivo implementado para que a tensão seja reduzida de 6V para uma tensão menor que 5V (tensão máxima suportada nas portas do microcontrolador ATmega328). A saída do divisor resistivo é conectada a porta analógica A_0 do microcontrolador do qual irá detectar toda vez que o sinal retificado e reduzido passar por zero volt.

3.2.2 Leitura da temperatura

A leitura da temperatura da água se torna necessária, uma vez que o usuário irá entrar com a temperatura desejada do banho e este valor de referência será comparado com a tem-

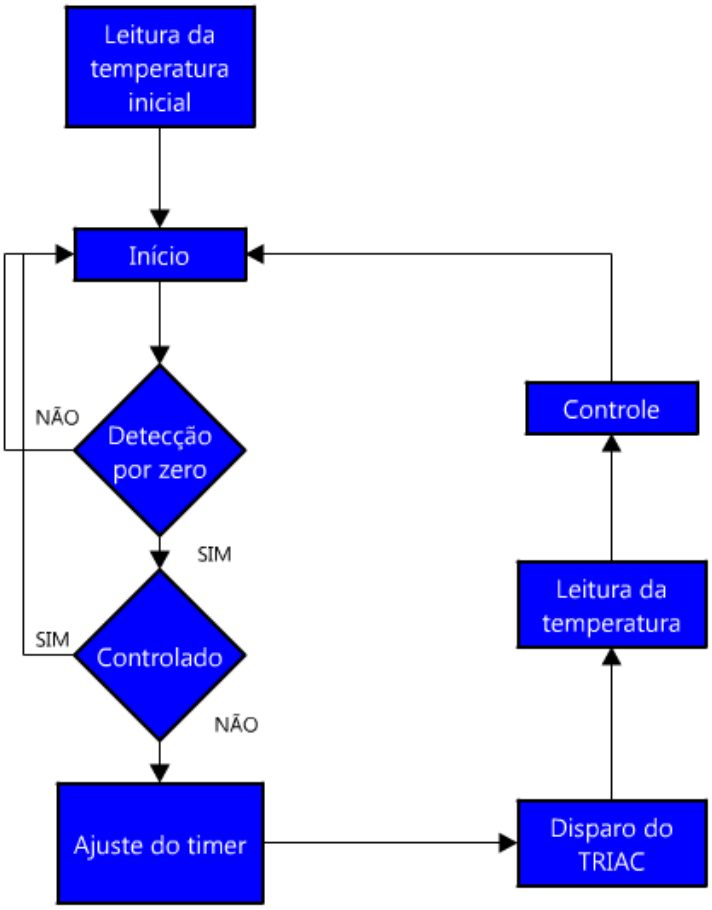

Figura 3.3 – Fluxograma do software do projeto

peratura atual da água. O sensor utilizado para a medição da temperatura da água é o LM35. Este sensor se comporta de forma linear, e fornece uma tensão de $10\frac{mV}{°C}$ sendo a temperatura máxima medida de 150°C, fornecendo assim uma tensão máxima de 1,5V. A tabela 3.1 mostra os pinos do microcontrolador em que o sensor é conectado.

Tabela 3.1 – Conexões do sensor de temperatura ao Arduino

Terminal do sensor	**Pino do Arduino**
Alimentação ($+V_S$)	5V
Saída analógica (V_out)	A1
Terra (GND)	GND

Um resistor de 2,2kΩ, conectado entre o terra e a saída do sensor, foi utilizado para a redução de ruídos e possíveis erros de medição

O sensor foi configurado para medir temperaturas entre 2°C até 110°C. Esta configuração foi realizada mudando-se a referência do conversor A/D de 5V (referência padrão) para

1,1V (referência interna do microcontrolador). Desta forma, o conversor A/D irá fornecer um valor de 1023 quando o sensor estiver marcando 110°C, diminuindo assim a influência de possíveis ruídos externos em comparação à referência de 5V. Com a referência em 1,1V, para cada aumento de uma unidade na leitura do conversor A/D temos um aumento de $\frac{1,1V}{1024}$, ou seja, 1,07mV.

Para diminuir os efeitos dos ruídos, presentes no sensor LM35, foi feita uma média aritmética de 20 leituras do sensor de temperatura, dessa forma tem-se uma melhor precisão da temperatura medida.

Com a saída do sensor de temperatura conectada a porta analógica A_1 do Arduino, utilizou-se a seguinte equação para a conversão de tensão em temperatura:

$$\text{Temperatura} = \frac{1,1}{1024}.100.AD$$

onde a variável Temperatura é dada em °C, AD representa a leitura do conversor A/D. O valor 100 é utilizado para transformar o valor de tensão em °C.

3.3 Controle

O levantamento da planta do sistema foi realizado utilizando-se o *software* Matlab.

Primeiramente foi realizada a aquisição de dados do sistema com o auxílio do sensor de temperatura LM35 e o Arduino. O sensor e seus terminais foram envolvidos com espaguete termoretrátil e silicone, para que não houvesse passagem de água na parte interna do sensor, ocasionando erros de medição.A figura 3.4 mostra o sensor com a proteção aplicada.
Os terminais do sensor foram soldados nos fios de um cabo de par trançado, mais conhecido como cabo de rede. O motivo da escolha deste cabo foi a sua tolerância a interferências, fácil manuseio e maleabilidade.

Os dados adquiridos foram os de temperatura da água em função do tempo, com o funcionamento do chuveiro em potência máxima, ou seja, a resposta à um degrau de potência. Na figura 3.5 são mostrados os dados adquiridos pelo microcontrolador e plotados com a ferramenta Matlab.

3.3.1 Planta do sistema

O tempo para o sistema atingir o regime permanente com a potência máxima foi por volta de 85 segundos. Com os dados adquiridos, uma ferramenta do Matlab chamada *System Identifi-*

Figura 3.4 – Sensor LM35 com espaguete termoretrátil

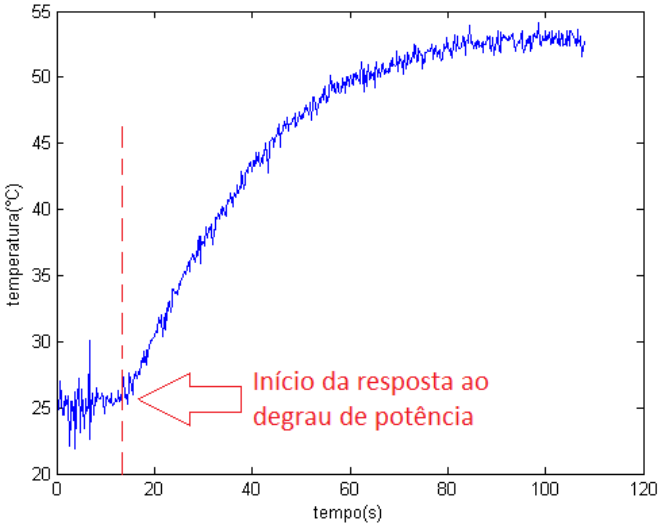

Figura 3.5 – Gráfico Temperatura X Tempo

cation Toolbox foi utilizada para a identificação da planta do sistema. Antes de se utilizar essa ferramenta foi necessário deslocar os dados da temperatura para que fossem analisados a partir do zero, ou seja, reduziu-se a temperatura inicial (ambiente), como mostrado na figura 3.6.

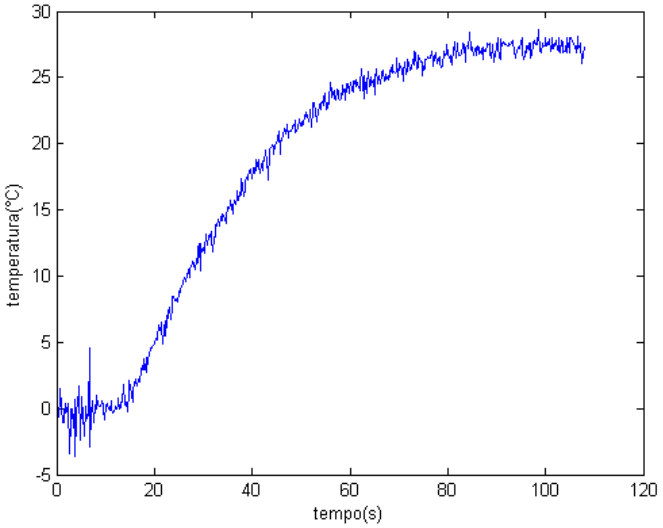

Figura 3.6 – Gráfico Temperatura X Tempo partindo do zero

O *System Identification Toolbox* pode ser acessado com o comando "*ident* dentro do ambiente de programação do Matlab. A planta a ser identificada tem como entrada a tensão CA da rede elétrica, que vai ser aplicada a carga do chuveiro (resistência), e como saída a temperatura da água em função do tempo. Os dados adquiridos foram importados para a ferramenta e desta forma a planta do sistema foi identificada com a função de transferência mostrada na equação abaixo.

$$G(s) = \frac{28{,}524}{25{,}367s+1}$$

A figura 3.7 faz uma comparação entre a planta obtida e os dados e mostra que a planta adquirida pela ferramenta representa o sistema de forma satisfatória.

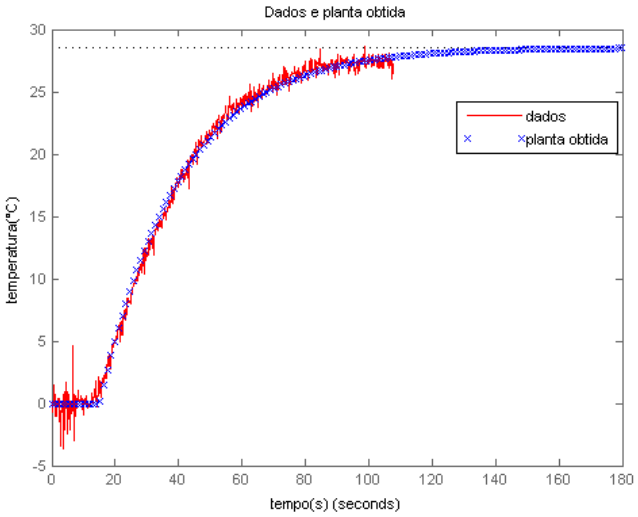

Figura 3.7 – Comparação entre dados e planta do sistema

3.3.2 Tempo Morto

Ao coletar-se os dados de temperatura em função do tempo foi notado que o sensor de temperatura LM35 apresenta um tempo de reposta de cerca de quatro segundos. Devido a este atraso, a planta do sistema foi modificada para a seguinte expressão:

$$G_d(s) = G(s).e^{-4s}$$

onde $G_d(s)$ representa a planta do sistema agora levando-se em conta o tempo de resposta de quatro segundos (e^{-4s}). Para a eliminação desse tempo morto da malha de controle foi utilizado uma estratégia de controle chamada Preditor de Smith.

3.3.3 Controlador

O controlador utilizado para a implementação do sistema foi o controlador PI (Proporcional Integral). Ele é o conjunto do controlador proporcional e o controlador integral trabalhando em paralelo. Utilizando a ferramenta do Matlab chamada *PID Tuner*, através do comando *pidtool*, foi possível calcular os coeficientes do controlador PI. Os coeficientes do controlador foram calculados para que o sistema tivesse um tempo de resposta de cerca de 12 segundos,para que o tempo até o regime permanente fosse de aproximadamente de 40 segundos e para que não houvesse *overshoot*.

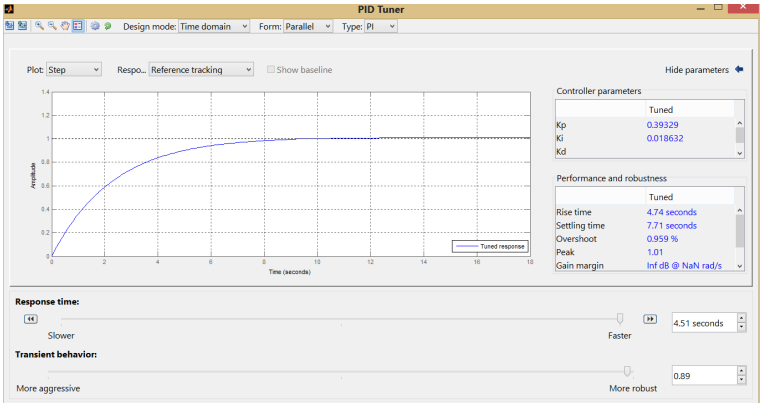

Figura 3.8 – PID Tuner

Com os coeficientes calculados, a função de transferência é representada na equação a seguir:

$$G_p(s) = \frac{0{,}143s+0{,}00566}{s}$$

Com a obtenção de todos os parâmetros necessários para a implementação do sistema de controle foi possível representar a malha de controle como mostra a figura 3.9.

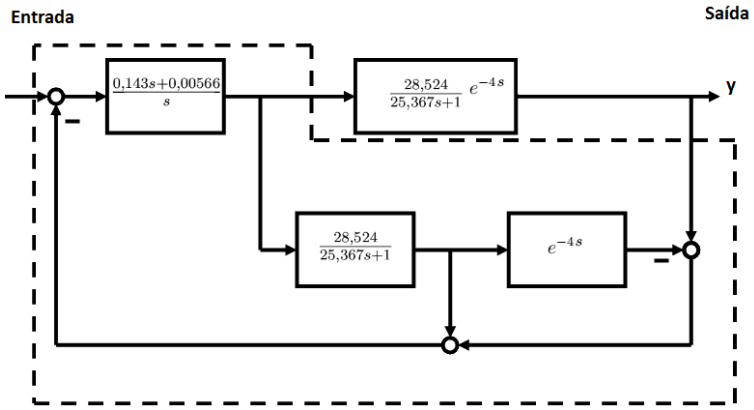

Figura 3.9 – Malha de controle do regulador

3.3.4 Discretização

Para a implementação das leis de controle em linguagem de programação para o microcontrolador foi necessário a discretização das funções de transferência uma vez que elas foram obtidas no domínio do tempo contínuo e os microcontroladores trabalham apenas com sinais amostrados, ou seja, no domínio do tempo discreto.

O *software* Matlab possui uma função chamada *c2d(sys,Ts)*, que realiza essa conversão. Os parâmetros *sys* e *Ts* representam respectivamente a função de transferência a ser discretizada

e o tempo de amostragem. O tempo de amostragem empregado neste projeto é de um segundo, que é o intervalo de tempo de coleta do dado de temperatura da água.

Com os parâmetros obtidos, foi possível realizar a conversão para o domínio do tempo discreto como mostra a tabela 3.2.

Tabela 3.2 – Discretização do sistema contínuo

Função	**Sistema contínuo**	**Sistema discreto**
Controlador	$G_p(s) = \frac{0{,}143s+0{,}00566}{s}$	$G_p(z) = \frac{0{,}143z-0{,}1374}{z-1}$
Planta do sistema	$G(s) = \frac{28{,}52}{25{,}37s+1}$	$G(z) = \frac{1{,}103}{z-0{,}9613}$
Planta do sistema com atraso	$G_{at}(s) = \frac{28{,}52}{25{,}37s+1}.e^{-4s}$	$G_{at}(z) = \frac{1{,}103}{z-0{,}9613}.z^{-4}$

3.4 Ajuste do timer e disparo do TRIAC

Para controlar o ângulo de disparo do TRIAC foi utilizado o recurso de interrupção por tempo do microcontrolador.

A corrente alternada ao passar pelo retificador de onda completa tem, na saída, uma onda com frequência e período modificados. Como a tensão presente nas residências possui uma frequência de 60Hz, ao ser retificada ela passa a ter a frequência de 120Hz. A figura 3.10 mostra como funciona a retificação do sinal.

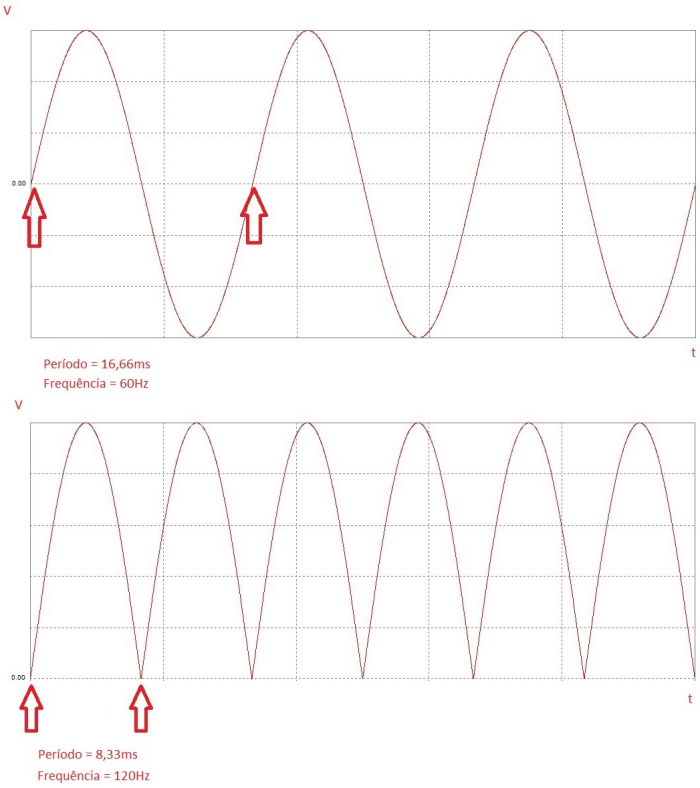

Figura 3.10 – Sinal retificado

Como pode-se observar, o período da onda passa de 16,66ms para 8,33ms. Dessa forma, o disparo do TRIAC deverá ser executado no intervalo de tempo calculado pela equação a seguir:

$$t = (1 - d).8,33$$

onde d é o *duty cycle* (valor entre 0 e 1), obtido na saída do controlador PI do sistema, e t é o intervalo de tempo, em milissegundos, em que a interrupção deverá ser executada.

A interrupção por tempo gerada pelo microcontrolador executa uma rotina da qual tem a função de gerar um pulso de 5V, procedente de um pino digital, para o optoacoplador MOC3020. O pulso gerado liga o *led* interno do optoacoplador que, por sua vez, aciona o fotodiac do componente permitindo a passagem de corrente elétrica nos terminais 4 e 6, responsáveis pelo disparo do TRIAC. Um exemplo de controle de disparo pode ser visto na figura 3.11 a seguir.

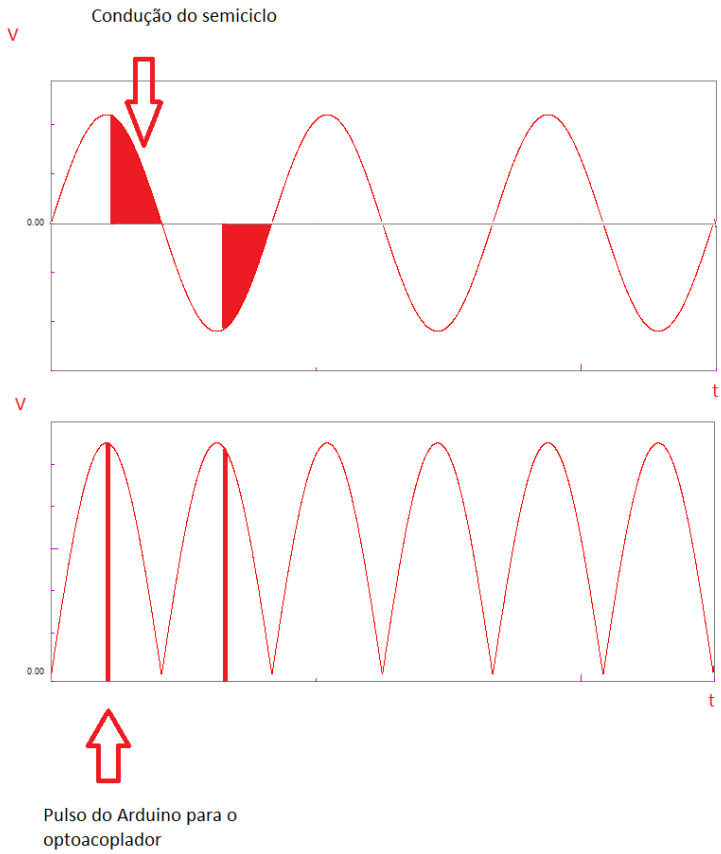

Figura 3.11 – Exemplo de controle de ângulo de condução

4 MONTAGEM DO PROJETO

Neste capítulo serão comentados os procedimentos adotados para a montagem do projeto do regulador de temperaturas, como o circuito implementado na protoboard e as conexões na parte interna do chuveiro.

4.1 Circuito de controle de temperatura

O circuito microcontrolado para testes foi montado e testado em um dos laboratórios do Núcleo de Pesquisa e Desenvolvimento em Engenharia Elétrica (NUPEDEE) localizado na Universidade Federal de Santa Maria. Na figura abaixo podemos ver o circuito implementado em uma protoboard.

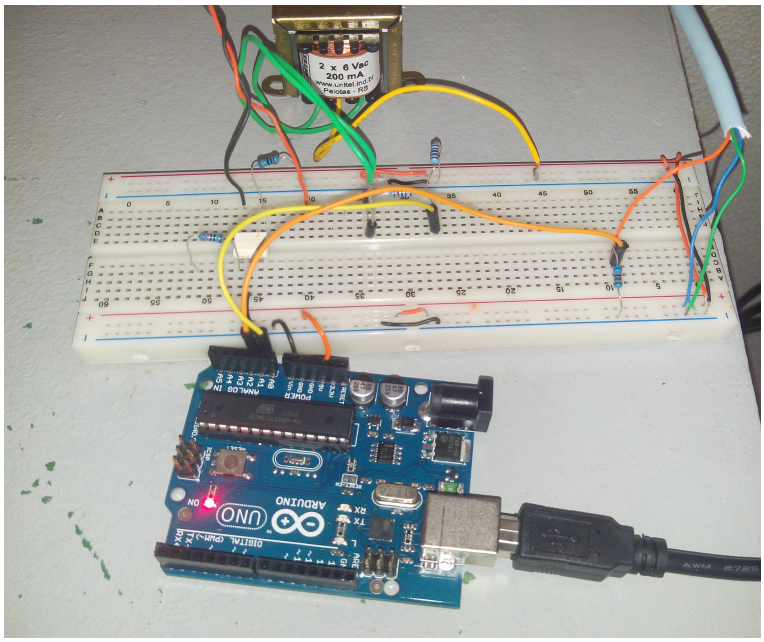

Figura 4.1 – Teste do circuito de controle

4.2 Instalação no chuveiro

Primeiramente a ducha ThermoSystem foi desmontada retirando-se o circuito de controle de temperatura convencional, cujo funcionamento foi comentado no capítulo 3. Após a retirada, foram soldados quatro fios no circuito interno do chuveiro. A tabela 4.2 representa as conexões dos fios soldados ao circuito microcontrolado.

O MT2 conectado ao terminal 4 do optoacoplador terá a função de disparar o TRIAC

Tabela 4.1 – Conexões do chuveiro ao circuito microcontrado

Fio	**Conexão**
MT2 (TRIAC)	Terminal 4 do optoacoplador
Gate (TRIAC)	Terminal 6 do optoacoplador
Fase da rede	Alimentação do Transformador
Neutro da rede	Neutro do transformador

após o microcontrolador realizar a detecção de passagem por zero e calcular a lei de controle do sistema. A fase e o neutro da rede são conectados ao transformador que só irá conduzir corrente elétrica quando houver passagem de água pelo diafragma (abertura do registro) fazendo com que dois contatos se encostem fechando a malha do circuito. A figura 4.2 mostra as conexões dos fios na parte interna do chuveiro.

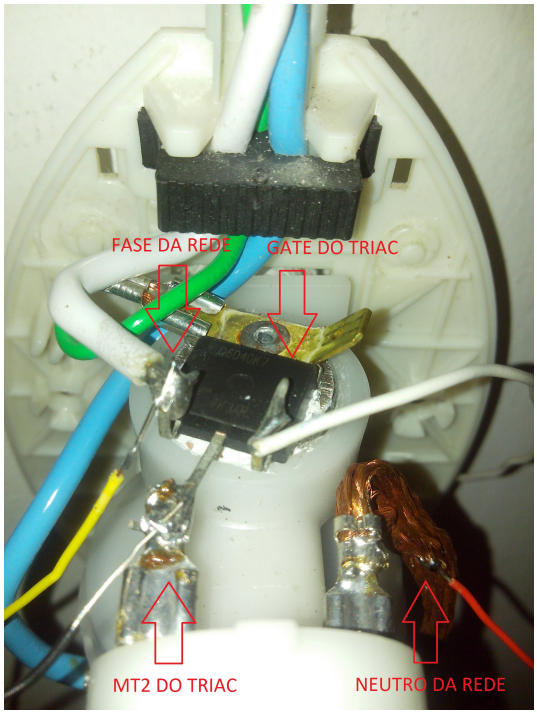

Figura 4.2 – Conexões da parte interna do chuveiro

Após a instalação na parte interior do chuveiro, foi realizada a instalação do sensor de temperatura. Ele foi posicionado abaixo da saída de água do espalhador para que a água ao passar por ele aqueça-o gerando assim um sinal analógico na saída do sensor referente a temperatura e possibilitando a leitura deste sinal na porta analógica do microcontrolador.Seus

terminais foram conectados a protoboard que, por sua vez, foram conectado aos devidos pinos do Arduino. A figura 4.3 mostra como foi realizada a instalação do sensor de temperatura.

Figura 4.3 – Instalação do sensor de temperatura

4.3 Implementação no microcontrolador

Os cálculos das leis de controle aplicadas ao sistema para que ele atinja a estabilidade são realizados toda vez que há a passagem por zero Volt na porta analógica do microcontrolador e são aplicados apenas no próximo ciclo. O fluxograma mostrado na figura 4.4 representa a lógica do sistema.

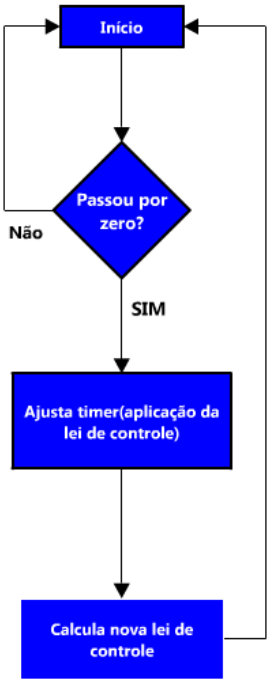

Figura 4.4 – Lógica do sistema

4.4 Circuito Impresso

Após serem realizados os testes de estabilidade do sistema, a implementação da interface e do circuito de controle foram desenvolvidas na ferramenta CAD chamada *Eagle*.

A placa da interface do circuito possui dois displays de sete segmentos e dois botões. Os displays tem a função de mostrar a temperatura desejada da água, enquanto os botões tem a função de incrementar ou decrementar essa temperatura. As figuras 4.5 e 4.6 mostram respectivamente o esquemático da placa no *Eagle* e a placa já impressa com seus componentes soldados.

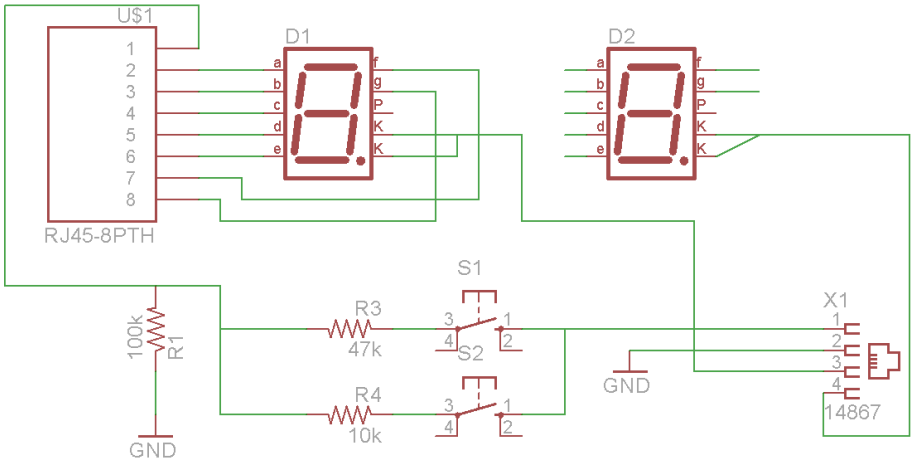

Figura 4.5 – Esquemático da placa de interface

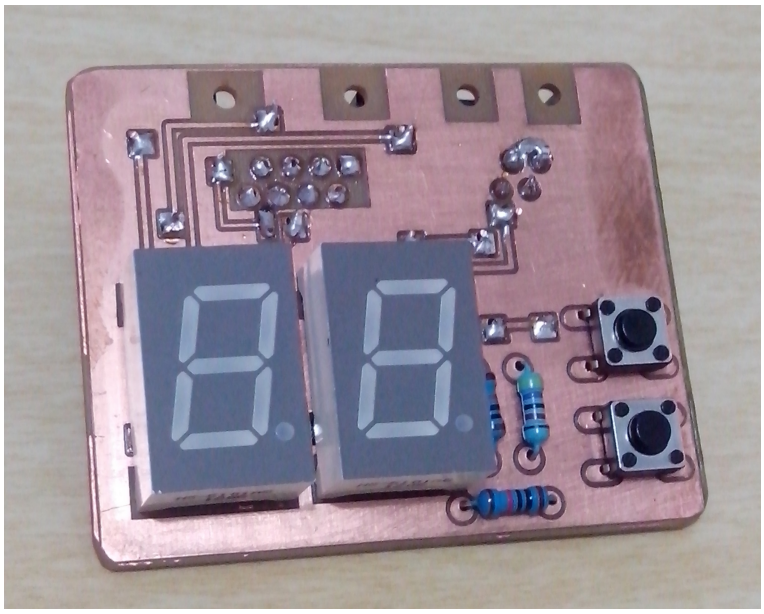

Figura 4.6 – Placa de interface

A placa do circuito de controle possui um microcontrolador ATmega328, um optoaco-

plador, componentes eletrônicos como resistores, capacitores, diodos e transistores e um regulador de tensão. Esta placa terá a funcão de substituir a plataforma de prototipagem (Arduino UNO), e realizar todo o controle do sistema. Sua localização será na parte interna do chuveiro. As figuras 4.7 e 4.8 mostram respectivamente o esquemático do circuito de controle e a placa impressa com os componentes.

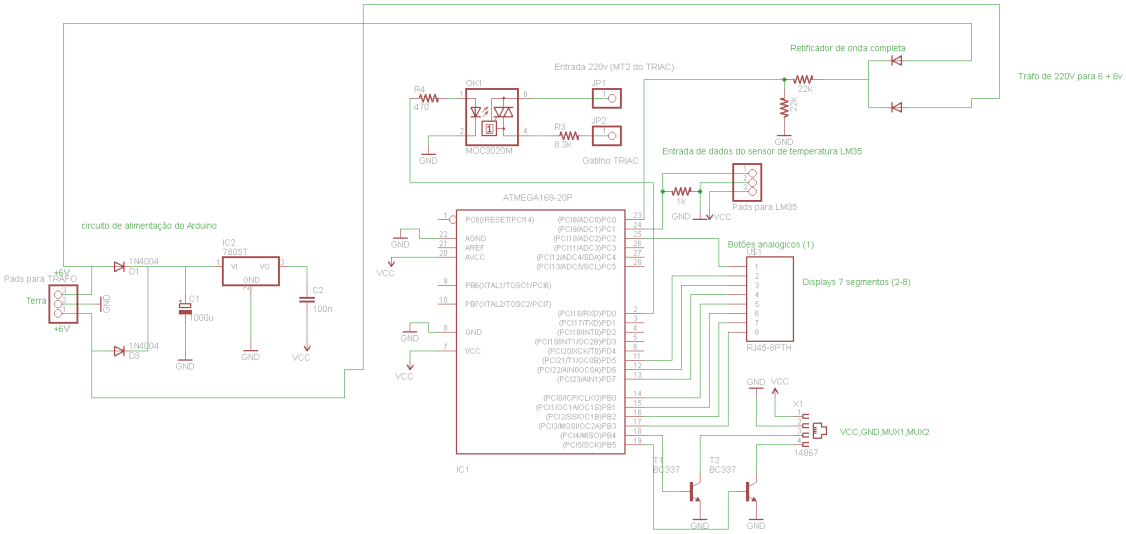

Figura 4.7 – Esquemático do circuito de controle

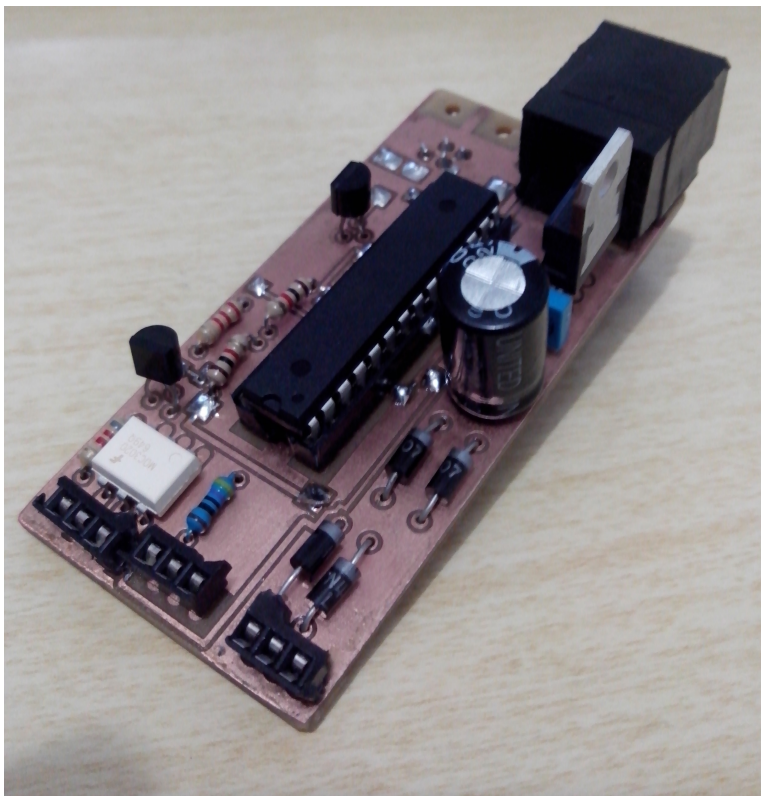

Figura 4.8 – Circuito de controle

5 RESULTADOS

Três testes com diferentes temperaturas de referência foram realizados para que fosse possível a análise da eficiência do regulador. Nos três testes, a temperatura ambiente encontrava-se por volta de 27°C. Os dados de temperatura foram adquiridos utilizando-se o próprio sensor LM35.

O primeiro teste foi realizado com a referência (temperatura desejada da água) para 34°C. A figura 5.1 mostra o gráfico da temperatura da água em função do tempo.

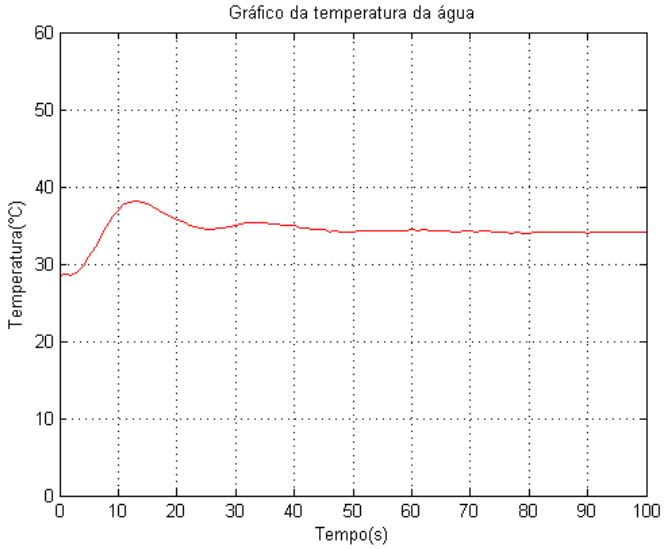

Figura 5.1 – Temperatura da água para referência em 34 graus Celsius

É possível notar uma pequena variação na temperatura após o sistema atingir o regime permanente (cerca de 40 segundos). Essa variação, apesar de não influenciar de forma relevante na variação da temperatura, provavelmente se dá devido ao ruído do sensor LM35.

Mesmo com a presença de ruídos no sensor de temperatura, o regulador teve uma eficiência satisfatória, com baixíssimas variações em relação a temperatura desejada, que dificilmente seriam notadas na pele de uma pessoa.

O segundo teste realizado foi para a temperatura desejada de 37°C. A figura 5.2 mostra o gráfico para o teste realizado.

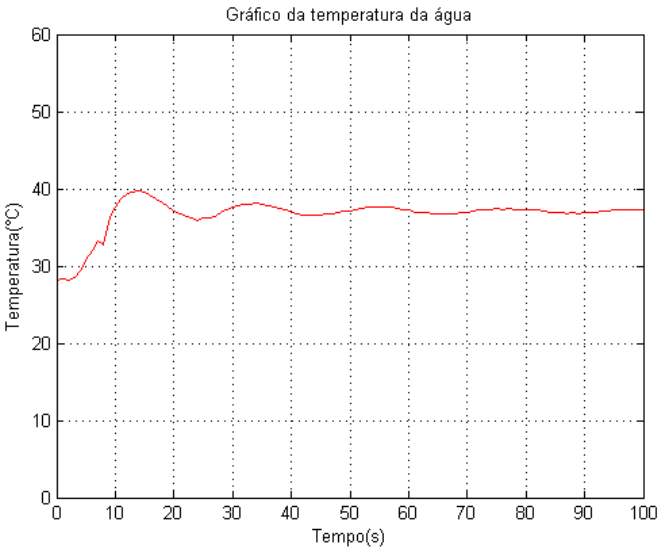

Figura 5.2 – Temperatura da água para referência em 37 graus Celsius

Neste caso pode-se perceber que o ruído no sensor teve uma influência maior, pois após o tempo de regime permanente as variações foram um pouco maiores que as analisadas no primeiro teste, porém, como o primeiro teste, essas variações seriam dificilmente notadas em um banho.

O terceiro e último teste foi realizado com a temperatura desejada de 40°C. A figura 5.3 mostra o gráfico para o último teste.

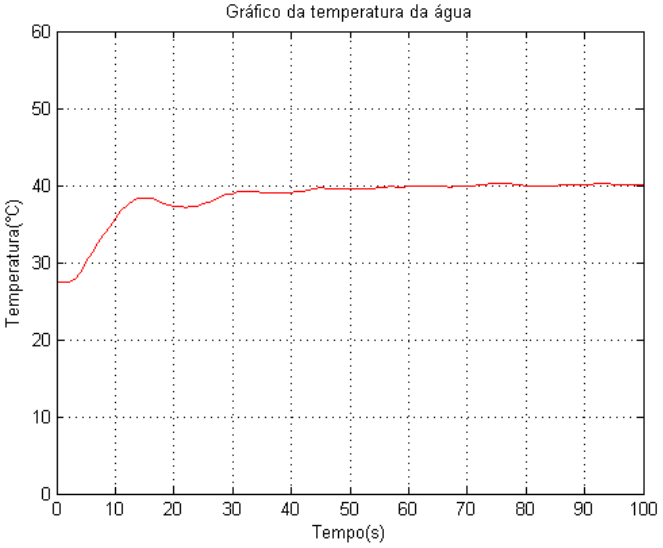

Figura 5.3 – Temperatura da água para referência em 40 graus Celsius

O comportamento deste teste foi bastante semelhante ao do primeiro. As variações de temperatura após a estabilização foram baixíssimas, em torno de 0,2°C à 0,3°C.

Com base nos gráficos apresentados para os testes, pode-se notar que o regulador de

temperaturas funciona de forma satisfatória com erros que variam em torno de 0,2°C à 0,3°C, valores que a pele humana dificilmente irá sentir em um banho. O tempo de ajuste de temperatura da água do banho foi reduzido de dois minutos (tempo médio para ajuste da temperatura ideal de um chuveiro convencional) para cerca de 40 segundos.

6 CONCLUSÃO

A possibilidade de uma precisão maior na temperatura da água e um tempo menor de ajuste é um atrativo bastante interessante que agrega um conforto maior ao banho.

A solução apresentada por esse trabalho propõe um regulador de temperaturas para chuveiros elétricos que possibilita o usuário escolher como parâmetro a temperatura em °C para a água do seu banho. O regulador foi implementado com o auxílio de um microcontrolador, um sensor de temperatura, componentes eletrônicos e conhecimentos de engenharia que envolvem sistemas de controle e circuitos eletrônicos.

Os resultados apresentados mostraram que o sistema é eficaz e consegue estabilizar a temperatura em uma faixa aceitável de erro e em um tempo menor do que os chuveiros existentes no mercado, fornecendo ao usuário conforto e economia de energia e água.

Uma grande dificuldade que o projeto apresentou foi o tempo de resposta do sensor de temperatura, o que levou a necessidade da implementação de uma estratégia que visasse a eliminar esse tempo morto da malha de controle para que o controlador não fosse afetado o que poderia causar uma não convergência da estabilidade do sistema.

Os tópicos a seguir são sugeridos para trabalhos futuros:

- Testes do sistema de controle com outros sensores de temperaturas mais precisos;
- Aplicação do sistema de controle utilizando outros métodos para eliminação do tempo morto.
- Otimização dos coeficientes do controlador para diminuir o tempo até o regime permanente.

7 REFERÊNCIAS

Arduino, I. D. E. Disponível em:< http://arduino.cc/en/main/software> Acesso em 01 de dezembro de 2013.

ATmega328P Datasheet. Disponivel em :<http://www.atmel.com> (acesso em 10 de dezembro de 2013), 2009

BOLTON, William. **Engenharia de controle**. Makron Books, 1995.

Curso de Arduino. Disponível em: <http://www.cursodearduino.com.br/apostila/apostila-rev4.pdf> (acesso em 20 de setembro de 2013).

LM35 Datasheet . Disponível em: <www.national.com/ds/LM/LM35.pdf> (Acesso em 15 de dezembro de 2013).

MACHADO, L.L. - **Controle de Processos Distribuídos na Presença de Tempo Morto**, UFSC, 2004.

Matlab. Disponível em: <http://www.mathworks.com> (acesso em 20 de novembro de 2013).

OGATA, Katsuhiko; LEONARDI, Fabrizio. **Engenharia de controle moderno**. Prentice Hall, 2003.

PILATTI, L. A. D. - **Controle da Temperatura e Vazão de um Chuveiro Usando Lógica Fuzzy**, UFSC, 2012.

SMITH, Otto JM. **A controller to overcome dead time**. ISA Journal, v. 6, n. 2, p. 28-33, 1959.

TANENBAUM, Andrew S.; MACHADO FILHO, Nery. *Sistemas operacionais moder-*

nos. Prentice-Hall, 1995.

Uno, Arduino. Disponível em:< http://arduino. cc/en/Main/arduinoBoardUno> (acesso em 10 de dezembro de 2013).

ANEXOS

ANEXO A – Código em linguagem C do regulador de temperaturas

```
1
2   //#################################################################
3   //Codigo do regulador de temperaturas para chuveiros eletricos
4   //#################################################################
5  #include <TimerOne.h>
6
7 //DECLARACAO DE VARIAVEIS************************************
8 float delta;
9 float temp_estimada = 0;
10 float tempo;
11 float temp_medida = 0;
12 float temp_desejada = 30 ;
13 float erro = 0;
14 float erro_a = 0;
15 float d;
16 float tempo_delay = 0;
17 float temp_ambiente;
18 float temp_medida_z;
19 float temp_desejada_z;
20 float tempo_botao;
21 boolean controlado = false;
22 int potencia;
23 int decimal;
24 int numeral;
25 int tensao_retificada;
26 int i;
27 int mux1 = 18;
28 int mux2 = 19;
29 int flag;
30 int flag_botao;
31 byte digitos_display[10][7] = {
32     { 1, 1, 1, 1, 1, 1, 0} // = 0
33     ,
34     { 0, 1, 1, 0, 0, 0, 0} // = 1
35     ,
36     { 1, 1, 0, 1, 1, 0, 1} // = 2
37     ,
38     { 1, 1, 1, 1, 0, 0, 1} // = 3
39     ,
40     { 0, 1, 1, 0, 0, 1, 1} // = 4
41     ,
42     { 1, 0, 1, 1, 0, 1, 1} // = 5
43     ,
44     { 1, 0, 1, 1, 1, 1, 1} // = 6
45     ,
46     { 1, 1, 1, 0, 0, 0, 0} // = 7
47     ,
48     { 1, 1, 1, 1, 1, 1, 1} // = 8
49     ,
50     { 1, 1, 1, 1, 0, 1, 1} // = 9
51 };
52
53 //FIM*****************************************************
54
55 //PRIMEIRA FUNCAO A SER EXECUTADA PELO MICROCONTROLADOR*******************
```

```
56 void setup() {
57       Serial.begin(9600);
58      analogReference(INTERNAL);
59      pinMode(2, OUTPUT); //Pulso para o optoacoplador
60      pinMode(11, OUTPUT);//Display
61      pinMode(12, OUTPUT);//Display
62      pinMode(13, OUTPUT);//Display
63      pinMode(14, OUTPUT);//Display
64      pinMode(15, OUTPUT);//Display
65      pinMode(16, OUTPUT);//Display
66      pinMode(17, OUTPUT);//Display
67      pinMode(18, OUTPUT);//mux1
68      pinMode(19, OUTPUT);//mux2
69      pinMode(A0, INPUT); //Entrada da tensao retificada
70      pinMode(A1, INPUT); //entrada da temperatura do sensor
71      pinMode(A2, INPUT); //entrada do botao de +/- temperatura
72      Timer1.attachInterrupt(aciona);//interrupcao de tempo para pulso do
            optoacoplador
73      delay(2000);
74
75      leitura();
76      temp_ambiente = temp_medida;
77      d = (temp_desejada - temp_ambiente)/28;
78      tempo_delay = 1-d;
79      tempo = 2000;
80 }
81
82 //FIM*********************************************************
83
84 //****************FUNCAO PARA PULSO NO OPTOACOPLADOR**********
85 void aciona(){
86
87      digitalWrite(2,HIGH);
88
89      if (tempo_delay < 0.1){
90   delayMicroseconds(1000);
91      }
92      else
93          delayMicroseconds(100);
94
95      digitalWrite(2,LOW);
96      Timer1.stop();
97
98 }
99
100 //FIM*********************************************************
101
102 //FUNCAO DE LEITURA DA TEMPERATURA*****************************
103 void leitura(){
104       temp_medida=analogRead(A1) * 1.10 / 1024*100;
105       for (i=0;i<20;i++)
106           temp_medida = temp_medida *0.9 + (analogRead(A1)* 1.10 / 1024*100)
                *0.1;
107 }
108 //FIM*********************************************************
109
110 //FUNCAO QUE VERIFICA BOTOES**********************************
111
```

```
112 void verifica_botao(){
113
114     if (analogRead(A2) < 2)
115   flag_botao = true;
116
117     else{
118       if(flag_botao == true){
119         if(analogRead(A2) < 819){
120           temp_desejada = temp_desejada + 1;
121           flag_botao = false;
122           tempo_botao = millis();
123         }
124         else{
125           temp_desejada = temp_desejada - 1;
126           flag_botao = false;
127           tempo_botao = millis();
128
129         }
130       }
131     }
132 }
133 //FIM*********************************************************
134
135 //FUNCAO DE ESCRITA DO DISPLAY*********************************
136 void escreve_display() {
137
138   if (millis() - tempo_botao < 5000){
139
140       decimal = temp_desejada/10;
141       numeral = temp_desejada -(decimal*10);
142   }
143   else{
144       potencia = d*99;
145       decimal = potencia/10;
146       numeral = potencia - (decimal*10);
147   }
148
149       if (flag == 1) //primeiro display
150       {
151         digitalWrite(mux2, LOW);
152
153         for (byte segCount = 0; segCount < 7; ++segCount)
154         {
155             digitalWrite(segCount + 11, digitos_display[decimal][segCount])
                  ;
156         }
157         digitalWrite(mux1, HIGH);
158         flag = 2;
159       }
160
161       else
162       { // segundo display
163         digitalWrite(mux1, LOW);
164         for (byte segCount = 0; segCount < 7; ++segCount)
165         {
166             digitalWrite(segCount + 11, digitos_display[numeral][segCount])
                  ;
167         }
```

```
168           digitalWrite(mux2, HIGH);
169           flag = 1;
170         }
171 }
172
173 //FIM*********************************************************
174
175 //FUNCAO QUE CALCULA LEI DE CONTROLE**************************
176
177 void controle(){
178
179   if (millis() > tempo)//
180   {
181    leitura();
182    temp_medida_z = temp_medida - temp_ambiente;
183    temp_desejada_z = temp_desejada - temp_ambiente;
184    erro_a = erro;
185
186    temp_estimada = temp_medida_z*0.9613 + 1.103*d;
187    erro = temp_desejada_z - temp_estimada;
188    d =  0.143*erro - 0.1374*erro_a + d;
189
190     if (d > 1) d = 1;
191     if (d < 0) d = 0;
192
193     tempo = tempo + 1000;
194     tempo_delay = 1 - d;
195   }
196 }
197
198 //FIM*********************************************************
199
200 //FUNCAO PRINCIPAL*******************************************
201 void loop() {
202
203     tensao_retificada = analogRead(A0);
204     if ((tensao_retificada < 2) && !controlado){
205
206
207           if (tempo_delay==0)
208             aciona();
209           else {
210             if (tempo_delay<1)
211               Timer1.initialize(tempo_delay * 8333 * 0.88);
212           }
213
214         controlado=true;
215         escreve_display();
216         verifica_botao();
217         controle();
218     }
219   else{ // histerese
220             if (tensao_retificada > 10)
221               controlado = false;
222           }
223 }//FIM*********************************************************
```