



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



Thailer Pereira Gun

## **CONTROLE DE TEMPERATURA UTILIZANDO ARDUINO E PID EM UM PROTÓTIPO DE EDIFICAÇÃO REDUZIDO**

Monografia de Graduação em Engenharia de Controle e Automação

Ouro Preto, 2017

Thailer Pereira Gun

**CONTROLE DE TEMPERATURA UTILIZANDO ARDUINO EPID  
EM UM PROTÓTIPO DE EDIFICAÇÃO REDUZIDO**

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

Orientador: prof. Henor Artur de Souza

Ouro Preto, 2017

G975c

Gun, Thailer Pereira.

Controle de temperatura utilizando arduino e PID em um protótipo de edificação reduzido [manuscrito] / Thailer Pereira Gun. - 2018.

36f.: il.: color; tabs.

Orientador: Prof. Dr. Henor Artur de Souza.

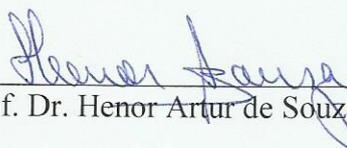
Monografia (Graduação). Universidade Federal de Ouro Preto. Escola de Minas. Departamento de Engenharia de Controle e Automação e Técnicas Fundamentais.

1. Controladores PID. 2. Controle automático. 3. Controle de Temperatura. 4. Conforto térmico. I. de Souza, Henor Artur. II. Universidade Federal de Ouro Preto. III. Título.

CDU: 681.5

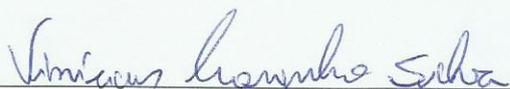
Catálogo: [ficha@sisbin.ufop.br](mailto:ficha@sisbin.ufop.br)

Monografia defendida e aprovada, em 05 de setembro de 2017, pela comissão avaliadora constituída pelos professores:



---

Prof. Dr. Henor Artur de Souza - Orientador



---

Vinícius Marinho Silva - Professor Substituto Convidado



---

Profª. Dra. Karla Boaventura Pimenta Palmieri – Professora Convidada

*Dedico este trabalho aos mestres que tive, que muito mais do que me transmitir conhecimento técnico, me ensinaram a pensar de forma crítica e objetiva.*

# Agradecimentos

A Deus que tornou possível minha caminhada e alcance do objetivo, apesar dos empecilhos. Aos meus pais e irmãos pela paciência, compreensão e preocupação com as noites mal dormidas em que este trabalho me tirava o sono.

Aos amigos troianos por me mostrarem que o descanso também era necessário para que a confecção deste projeto pudesse render ainda mais.

Agradeço ainda a Sá, Pitoco, Laura, Paulinha e Caio por não deixarem que me faltasse apoio e conhecimento específico quando necessário.

À Dona Lúcia (in memoriam), pelo ombro amigo e abraço de avó, os quais nunca me faltaram e por isso deixam agora saudade.

Muito obrigado a esta Universidade e seus docentes que moldaram meu conhecimento para que ao final eu pudesse realizar este trabalho, especialmente Professor Henor, pela paciência, disponibilidade e orientação fornecida.

*"Aprender é a única coisa de que a mente nunca se cansa, nunca tem medo e nunca se arrepende." (Leonardo Da Vinci)*

# Resumo

Controlar a temperatura de ambientes construídos vêm se tornando um desafio cada vez mais necessário e complexo, devido ao aumento da temperatura ambiente global devido a fatores como efeito estufa acelerado e grande presença de poluição nas cidades. É conhecido que os seres humanos possuem temperaturas ideais para sua sobrevivência e melhor rendimento em tarefas do dia a dia, como no trabalho ou na prática de esportes. O alto nível de desenvolvimento da tecnologia aplicada à engenharia torna essa tarefa mais simples e fornece resultados cada vez mais satisfatórios. Baseado nesses fatores, este trabalho apresenta uma solução para o controle de temperatura em ambientes contruídos para regiões tropicais e subtropicais, onde a temperatura é na maior parte, mais elevada. Para tanto, conseguir refrescar estes ambientes é muitas vezes o suficiente para manter a temperatura ideal, sem a necessidade de aquecer estes ambientes. O projeto em questão se mostra uma alternativa eficiente por se utilizar de poucos recursos, de forma controlada. Pesquisa-se, projeta-se e implementa-se um sistema envolvendo controle PID (Proporcional Integral derivativo) e o método de PWM (Modulação por Largura de Pulso) para regulagem da potência do atuador, um cooler de 12V. Utiliza-se também a plataforma de prototipagem Arduino, a qual realiza todo o processamento das informações obtidas pelo sensor e controla a ação do atuador, a partir de configuração prévia no software (interface) integrado. Testa-se e aplica-se diferentes valores para o controle PID e obtém-se um resultado otimizado para o sistema. Os valores de temperatura obtidos a partir deste controle são satisfatórios e fornecem um ambiente confortável neste protótipo de moradia em escala reduzida. O tempo de reação e estabilização mostram-se ideais para o projeto proposto, resultados esperados em um sistema PI(Proporcional Integral).

**Palavras-chaves:** Controle PID, PWM, Controle de temperatura, domótica, conforto térmico.

# Abstract

Controlling the temperature of built environments has become an increasingly necessary and complex challenge due to the increase in global ambient temperature, because of factors such as accelerated greenhouse effect and large presence of pollution in big cities. It is known that human beings have ideal temperatures for their survival and better performance in everyday tasks, such as working or practicing sports. The high level of development of engineering technology makes this task simpler and yields increasingly satisfactory results. Based on these factors, this work presents a solution for temperature control in built environments for tropical and subtropical regions, where the temperature is mostly higher. To do so, being able to refresh these environments is often enough to maintain the ideal temperature without the need of heating these places. The project in question is an efficient alternative because it uses few resources, in a controlled way. A system involving PID control and the PWM method to regulate the actuator power, a 12V cooler, is designed and implemented. It is also used the Arduino prototyping platform, which performs all the information obtained by the sensor and controls the actuator action, from previous configuration in the integrated software. Different values for the PID control are tested and applied and an optimized result is obtained for the system. The values of temperature obtained from this control are satisfactory and provide a comfortable environment in this prototype of small scale dwelling. The reaction time and stabilization are ideal for the proposed project, expected results in a PI system.

**Key-words:** PID Control, PWM, Temperature control, domotics, thermal confort.

# Lista de ilustrações

Figura 1	Diagrama de Blocos: controle com realimentação.....	13
Figura 2	Diagrama de Blocos: controle PID.....	15
Figura 3	Prótipo em vista lateral e superior (sem o teto).....	17
Figura 4	Arduino UNO.....	18
Figura 5	Interface do Arduino.....	19
Figura 6	LM35.....	20
Figura 7	Cooler .....	21
Figura 8	TIP31C .....	21
Figura 9	Circuito de controle simulado no Proteus .....	22
Figura 10	Conversão de tensão para temperatura e impressão no display LCD. .	23
Figura 11	Utilização da função PID do Arduino .....	24
Figura 12	Sistema de Controle de Temperatura instalado no protótipo e tela de apresentação de dados.....	25
Figura 13	Ensaio 1 .....	27
Figura 14	Ensaio 2 .....	27
Figura 15	Ensaio 3 .....	28
Figura 16	Ensaio 4 .....	28
Figura 17	Ensaio 5 .....	28
Figura 18	Gráfico 1: Curvas de reação dos ensaios .....	29

# Lista de tabelas

Tabela 1	Classificação de controladores convencionais e avançados.....	14
Tabela 2	Datasheet do LM35 .....	20
Tabela 3	Datasheet do TIP31C .....	21
Tabela 4	Valores da Sintonia PID .....	29

# Sumário

1 Introdução. ....	10
1.1 Justificativas e Relevância. ....	11
1.2 Objetivos. ....	11
1.3 Metodologia. ....	11
1.4 Estruturação do Trabalho. ....	12
2 REVISÃO DE LITERATURA. ....	13
2.1 O Controle PID. ....	14
3 MATERIAIS E MÉTODOS. ....	17
3.1 O Protótipo de Edificação Reduzido. ....	17
3.2 O Sistema de Controle. ....	18
3.2.1 O Microcontrolador Arduino UNO. ....	18
3.2.2 LM35. ....	20
3.2.3 O Cooler. ....	20
3.2.4 O Transistor TIP 31C. ....	21
3.3 O Circuito de Controle. ....	22
3.4 A Sintonia do PID. ....	25
4 RESULTADOS. ....	27
5 CONCLUSÃO. ....	30
REFERÊNCIAS .....	31
APÊNDICE A - Código utilizado no Arduino.....	33

# 1 INTRODUÇÃO

Devido às constantes modificações climáticas às quais o planeta está sujeito, tem-se buscado cada vez mais o conforto em ambientes fechados a partir do controle de temperatura. O aquecimento global é hoje uma realidade, o que estabelece uma situação de aumento de temperatura em todo o mundo. Essa realidade exige então que se tomem novas diretrizes na construção de edificações, como a utilização de inovações tecnológicas e uma análise sistêmica das construções (SOUZA et al. 2003).

A temperatura também é pauta quando se trata de equipamentos na indústria. Nesta, vem se tornando cada vez mais crítica a necessidade de controlar a temperatura de funcionamento de equipamentos, ao passo que estes podem apresentar defeitos em sua função se sua temperatura interna estiver baixa ou alta demais. Com isso um sensoriamento e atuação referentes às suas temperaturas tem sido uma tônica em congressos e simpósios da indústria (ref.).

O ser humano possui condições fisiológicas ideais para cada tipo de atividade que realiza, o que inclui condições de ventilação, poluição sonora, iluminação, e também temperatura e umidade relativa do ar. Tendo isso em vista, controlar a temperatura de um ambiente se torna fundamental para que o ser humano atinja seus melhores resultados na atividade que realiza.

É possível realizar um controle de temperatura em ambientes construídos, como casas e apartamentos, utilizando-se de modelagem matemática e elementos eficientes de controle e aquisição de dados. Ao se conhecer a atividade e as condições do indivíduo que irá realizá-la, pode-se desenvolver um projeto de controle, que a partir de alguns pré-requisitos, ao serem seguidos, pode proporcionar um ambiente adequado.

É preciso lembrar que, na maioria dos casos, ao se projetar um sistema de controle, é preciso conhecer o modelo matemático que rege a situação a ser controlada. Além disso, quando o modelo matemático é estabelecido para se fazer um controle que envolva as forças da natureza, como o controle de temperatura, ocorrem diversos arredondamentos que proporcionam incertezas ao modelo. Isso ocorre porque a natureza não é controlável pelo homem, e pode trazer situações inesperadas que fogem da compreensão. Por esse motivo, pode acontecer de um operador, conhecendo apenas as variáveis de entrada e saída, conseguir controlar diversos sistemas sem compreender a matemática ali envolvida, ou todos os detalhes físicos envolvidos. Essa capacidade deu origem a um novo conceito, o conceito de “inteligência artificial”, evidenciando novo enfoque na teoria de processos industriais (complexos)

, emulando características do comportamento do ser humano, surgindo como nova alternativa de controle e modelagem. (SIMOES; SHAW, 2007)

Atualmente, ao se controlar variáveis de um processo, estabilidade e precisão são vistos como pré-requisitos básicos e indispensáveis. Afim de atingir avanços tecnológicos para alcançar tais necessidades, o ser humano adquiriu inúmeras técnicas que tornaram possível a análise e o desenvolvimento de projetos cada vez mais avançados neste sentido. Graças a isso, houve o surgimento da técnica de controle PID, hoje uma das mais utilizadas no controle de variáveis contínuas. Esta técnica consiste na utilização de um modelo matemático, o qual pode tomar como base o comportamento do sistema num tempo passado, presente e estimar seu comportamento futuro, realizando um controle bastante preciso e com grande estabilidade, mesmo em meio a distúrbios ou variações.

## 1.1 Justificativas e Relevância

No Brasil, um país tropical, as temperaturas predominantes são referentes ao verão. Em função disso, a frequência e precisão na análise do conforto em edificações se tornam ainda mais necessárias. As grandes cidades abrigam ainda uma quantidade significativa de prédios e grandes construções, as quais necessitam de um controle de temperatura interno, pois abrigam residências e escritórios empresariais onde o fluxo de pessoas é intenso.

Para um controle ótimo de temperatura, se faz necessária uma avaliação da construção, o que permite então a instalação de tecnologias que visa à adequação da temperatura do ambiente ao gosto do usuário ou aos padrões de conforto atuais (SOUZA et al, 2003).

## 1.2 Objetivos

Desenvolver e implementar um controle de temperatura utilizando a plataforma de prototipagem Arduino e sua IDE, bem como o método de controle PID num ambiente em escala reduzida.

## 1.3 Metodologia

As tarefas desenvolvidas neste trabalho são:

- Aquisição de dados utilizando um sensor apropriado;
- Implementação de um código que permite definir a temperatura desejada;
- Instalação dos equipamentos e monitoramento do ambiente em função de uma temperatura pré-estabelecida.

## 1.4 Estruturação do Trabalho

Este trabalho está composto por cinco capítulos.

No capítulo 1, apresenta-se uma introdução com um breve histórico do controle de temperatura, além da descrição do problema a ser tratado e a metodologia e objetivos do trabalho.

No capítulo 2, abordam-se algumas aplicações de PID e Arduino no controle de temperatura, além da apresentação do software utilizado e das principais dificuldades encontradas ao utilizar-se tal controle.

No capítulo 3 detalha-se a utilização do controle PID neste problema, e o sistema de controle desenvolvido para atingir os objetivos do trabalho.

No capítulo 4 são mostrados os resultados obtidos pela aquisição de dados.

No capítulo 5 encontra-se a conclusão à qual se pode chegar a partir da análise dos resultados.

Ao final encontra-se as referências e o apêndice contendo o código utilizado no Arduino.

## 2 REVISÃO DE LITERATURA

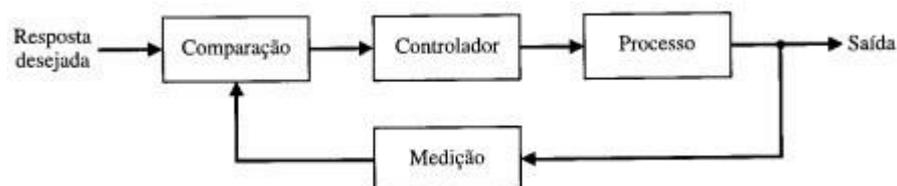
É papel do engenheiro conhecer e controlar grandezas físicas, visando sempre o benefício humano. Dessa forma, ao lidar com sistemas de controle, o engenheiro deve conhecer e buscar o controle dos segmentos ao seu redor, a fim de levar a sociedade produtos que forneçam utilidade e economia.

Existem diversos estudos e aplicações já realizadas, referentes ao controle de temperatura. Temperatura é uma grandeza física de base do Sistema Internacional (SI), que possui um conceito diferente de tempo, massa e comprimento. A combinação de dois corpos de mesmo comprimento resulta em um comprimento total igual a dobro do original. O mesmo ocorre com duas massas ou dois intervalos de tempo iguais. Esses padrões podem ainda ser indefinidamente multiplicados ou divididos gerando tamanhos arbitrários. Isso os define como grandezas extensivas. A temperatura é uma grandeza intensiva, já que a combinação de dois corpos à uma mesma temperatura resultará na mesma temperatura. (BENEDICT, 1977)

O controle de temperatura exige um sistema com retroação, ao passo que não existem sistemas perfeitos. Dessa forma, é provável que se encontre sempre uma diferença, por menor que seja, entre a temperatura desejada (entrada) e a real temperatura do ambiente (saída).

O controle com retroação (realimentação) consiste basicamente na aplicação de comparação e compensação em um processo. Dessa forma, aplica-se uma entrada desejada, obtém-se os resultados na saída e faz-se a comparação dos dois. Essa diferença deve então ser compensada por um controlador aplicado ao processo, calibrado de forma a minimizar a discrepância entre entrada e saída. Essa dinâmica é demonstrada no diagrama de blocos apresentado na figura 1.

Figura 1: Diagrama de Blocos: controle com realimentação



Fonte: DORF; BISHOP, 1998.

Na literatura, são propostos diversos modelos de controle de temperatura, utilizando os mais diversos recursos. Assim como em qualquer processo, deve-se avaliar o capital disponível, a complexidade exigida e a viabilidade de implementação e desenvolvimento.

Visando aplicar o controle da melhor forma possível, a escolha correta do controlador se torna indispensável. Alguns diferentes tipos de controladores indicados para diferentes projetos, podem ser vistos na tabela 1.

Tabela 1: Classificação de controladores convencionais e avançados.

<b>Projeto</b>	<b>Tipo de Controlador</b>
convencional	manual, PID, proporcional, em cascata e <i>feedforward</i>
avançado: clássico	escalonamento de ganhos, compensação do atraso de transporte, controle desacoplado
avançado: com aplicações	preditivo, modelo interno, adaptativo, nebuloso
avançado: poucas aplicações	não-linear, ótimo, robusto, inteligente (neural e evolutivo)

Fonte: COELHO et al., 2003

Uma lógica de controle possível de ser aplicada em sistemas de controle de temperatura é o controle PID (Proporcional Integral Derivativo). Buscando novas e melhores formas de se obter esse controle, aplicar métodos convencionais ou clássicos neste caso poderia não trazer resultados satisfatórios.

## 2.1 O Controle PID

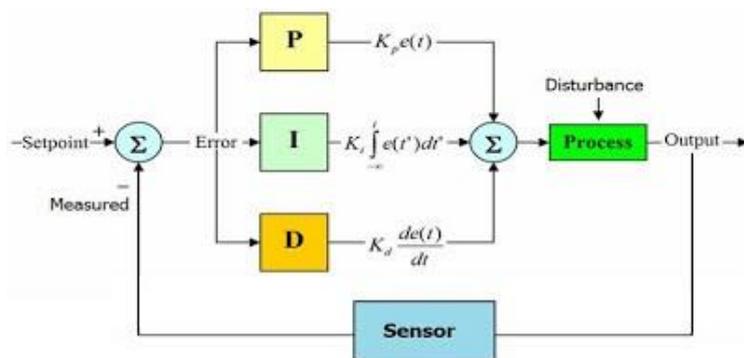
Dos tipos de controle utilizados em plantas residenciais e industriais, o método PID é o mais preciso e indicado. O controle PID baseia-se na modelagem matemática de um problema, representado por uma malha de processo a ser controlada.

Esse tipo de controle trabalha com a diferença entre o valor esperado de uma variável do processo e seu real valor, medido por meio de um sensor. Possibilita um controle com baixo erro de *offset* e boa estabilidade, e por isso é bastante utilizado em projetos acadêmicos e na indústria. Combina 3 modos de controle, P (proporcional), I (integral) e D (derivativo) (GUERRA, 2009).

Pode-se observar na figura 2 uma malha fechada de um processo, no qual atua um controlador PID. Estipula-se um *setpoint*, ou seja, um valor desejado para a variável controlada. Na saída da malha encontra-se um sensor que monitora o valor da mesma variável. O sistema calcula a diferença entre o *setpoint* e o valor medido, e a repassa para o controlador PID, que atua no sentido de reduzir essa diferença a zero.

Observando o esquema mostrado na figura 2, pode-se nomear o controlador (composto por P, I e D) como  $G_c(s)$ . Considerando-se o ganho da realimentação igual a um, a função de transferência do sistema pode ser representada pela equação 2.1.

Figura 2: Diagrama de Blocos: controle PID



PROJETO..., 2013

$$T(s) = \frac{C(s)}{R(s)} = G_c(s) \cdot \frac{G_p(s)}{1 + G_c(s) \cdot G_p(s)} \quad (2.1)$$

O objetivo, como já descrito, é tornar C(s) (variável do processo) igual a R(s) (setpoint), ou seja, reduzir o erro a zero. Pode-se concluir que, uma possibilidade seria fazer com que o termo  $G_c(s) \cdot G_p(s)$  fosse muito maior que um, de forma a se poder simplificar a equação como mostrado na equação (2.2).

$$T(s) = G_p(s) \cdot \frac{G_c(s)}{G_p(s) \cdot G_c(s)} \quad (2.2)$$

O controlador proporcional tem esse princípio de funcionamento. Apesar de parecer bastante simples e facilmente utilizável, pode apresentar grandes aumentos no que se diz respeito ao *overshoot* do sistema, ou seja, altos valores de pico atingidos pela variável no início do processo.

Para amenizar os efeitos negativos do *overshoot* elevado, matematicamente, introduz-se um componente derivativo ( $K_d$ ) que representa a taxa de mudança do erro. Isso pode ser visto na equação (2.3).

$$G_c(s) = K_p + K_d \cdot s \quad (2.3)$$

Na prática, é conhecido que, naturalmente, os valores obtidos na saída de um sistema, apresentam certa inércia em relação às variações na entrada. O resultado disso é uma ineficiência na correção do erro, por parte do controlador, já que este recebe como sinal de saída da planta um valor que, no exato momento, já não demonstra a realidade da planta. É justamente para tentar diminuir tal ineficiência que se combina ao controle proporcional um componente derivativo. O termo derivativo faz, a partir de resultados passados e comportamentos repetidos, uma previsão do próximo sinal de saída, fazendo com que a correção do erro seja feita mais precisamente. Este sistema, apesar de bastante eficiente para o controle de diversas plantas, pode apresentar um problema chamado *offset* (erro em regime permanente). Isso faz com que, ao se anular o desvio, a saída do controlador

também se torne nula, dependendo da constante de proporcionalidade e do sistema. O resultado disso é uma estabilização que, apesar de próxima do valor de *setpoint*, não consegue alcançá-lo, o que seria ideal. Para solucionar o problema causado pelo *offset*, é possível adicionar um termo integrativo ao sistema, de forma a se obter uma função de transferência do sistema igual à (2.4).

$$G_c(s) = K_p + K_d \cdot s + \frac{K_i}{s} \quad (2.4)$$

Com este termo adicionado, adiciona-se também um pólo ao sistema, o que diminui sua estabilidade. Além disso, a resposta transitória do sistema é mais lenta. Contudo, são problemas que podem ser contornados a partir da correta sintonia do controlador, ou seja, utilizando os valores corretos para os coeficientes proporcional, integral e derivativo.

## 3 MATERIAIS E MÉTODOS

Neste capítulo apresenta-se os materiais utilizados para a confecção deste trabalho, bem como os métodos empregados no controle desta planta.

### 3.1 O Protótipo de Edificação Reduzido

Para a implementação do sistema de controle proposto, foi utilizado um protótipo de edificação reduzido, construído em madeira (figura 3). As dimensões do protótipo são 62,5 cm de largura, 62 cm de comprimento e 37 cm de altura.



Figura 3: Protótipo em vista lateral e superior (sem o teto)

O protótipo mostrado na figura 3 é uma maquete (escala 1:600) de uma moradia real, construída pelo programa Usiteto das Usinas Siderúrgicas de Minas Gerais S.A - USIMINAS (USITETO..., 2001).<sup>1</sup>

<sup>1</sup> O principal objetivo deste programa é a redução do déficit habitacional brasileiro, e as moradias construídas são destinadas à população com renda de até três salários mínimos. Se comparada com as construções de habitações populares convencionais, a casa Usiteto é uma alternativa simples, feita em menor tempo e com menos gastos. A casa Usiteto, é uma alternativa mais econômica, simples e rápida, em comparação com o sistema convencional de construção de casas populares. (FRANSOZO, 2001)

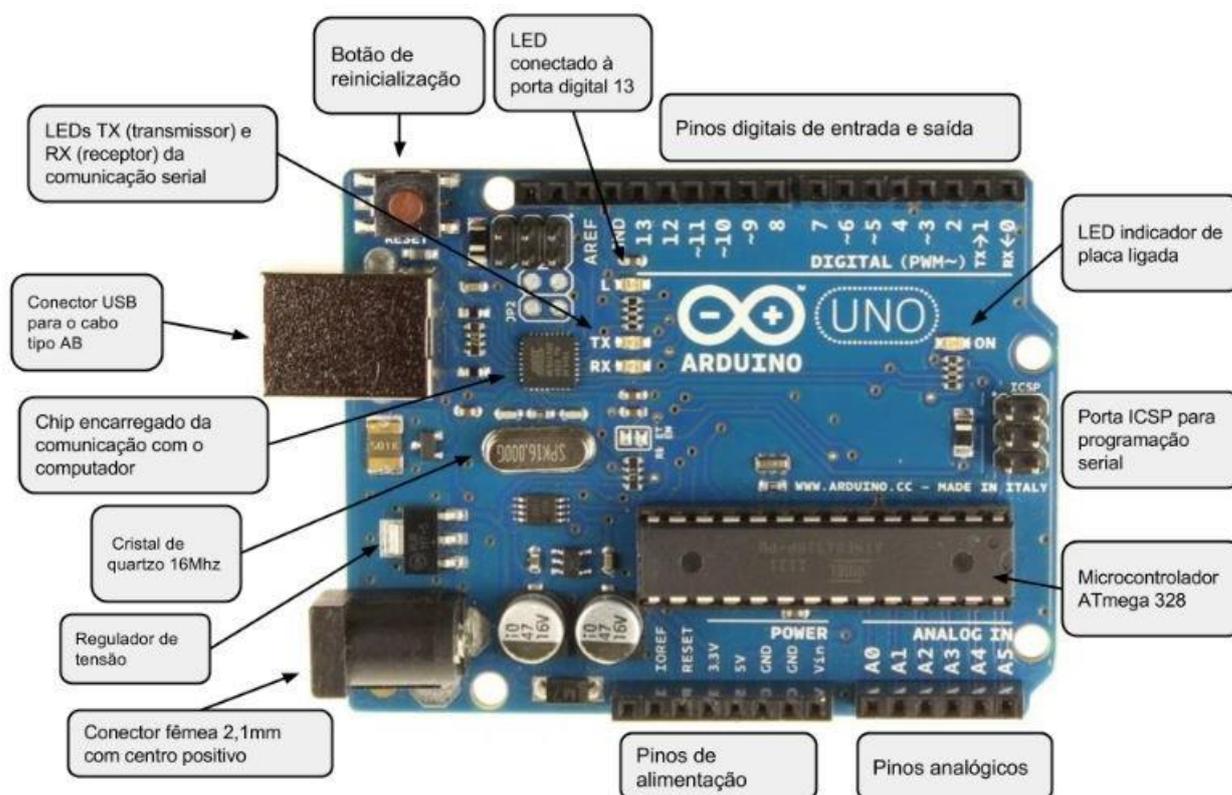
## 3.2 O Sistema de Controle

Apresenta-se aqui os principais componentes eletrônicos utilizados para a confecção deste trabalho.

### 3.2.1 O Microcontrolador Arduino UNO

O Arduino UNO (Figura 4) é um microcontrolador de prototipagem, que vem sendo utilizado tanto por amadores quanto profissionais da área de controle e automação. Apresenta fácil programação e baixo custo.

Figura 4: Arduino UNO



FONTE: CONHECENDO..., 2015.

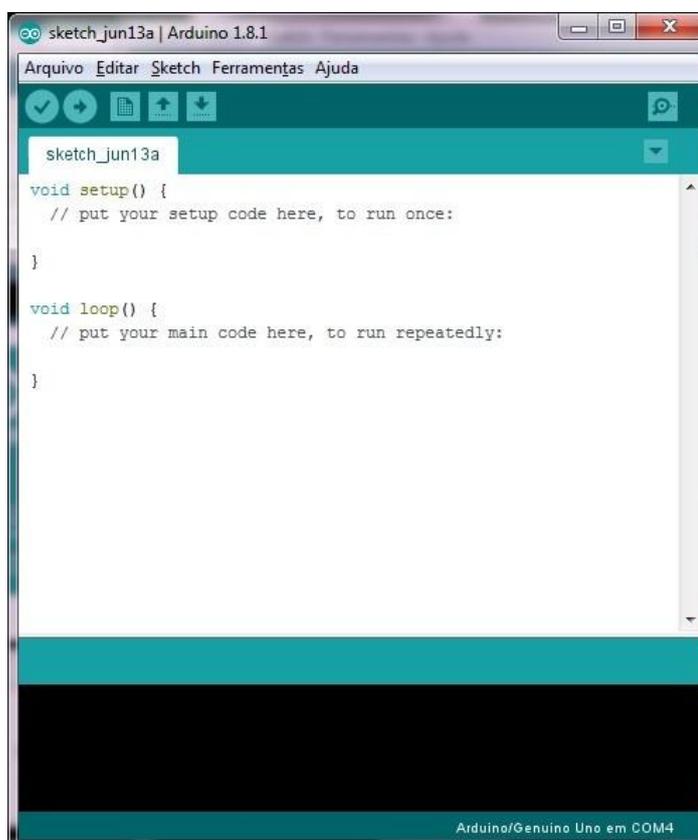
Os desenvolvedores do Arduino disponibilizam de forma gratuita um *software* de interface gráfica bastante simples (Figura 5) para sua programação, podendo esta ser feita em uma linguagem chamada de *processing*, baseada em C/C++. Esta linguagem não exige a utilização de compiladores ou *hardwares* adicionais, por possuir um *bootloader*, que dispensa o uso de programadores para o chip (BARBOSA, 2016). Para melhorar a experiência do usuário com o Arduino, são desenvolvidas diversas placas que facilitam a implementação dos mais variados projetos utilizando-o. Entre elas pode-se destacar as placas *WI-FI*, placas

de acionamento de motores DC, relés e outros atuadores. Estas placas são as chamadas *shields*.

A placa utilizada neste projeto possui as seguintes características técnicas:

1. Microcontrolador: Atmega328p;
2. Tensão de operação: 5V;
3. Tensão de entrada (recomendada): 7-12V;
4. Tensão de entrada (gimite): 6-20V;
5. Pinos de I/O digital: 14 (6 podem ser usadas como saídas PWM);
6. Pinos de entrada analógica: 6;
7. Corrente CC por pino de I/O: 40 mA;
8. Corrente CC para pinos de 3.3V: 50 mA;
9. Memória Flash: 32 KB onde 8 KB é usado pelo *bootloader*;
10. SRAM: 8 KB;
11. EEPROM: 4 KB;
12. Velocidade de clock: 16 MHz.

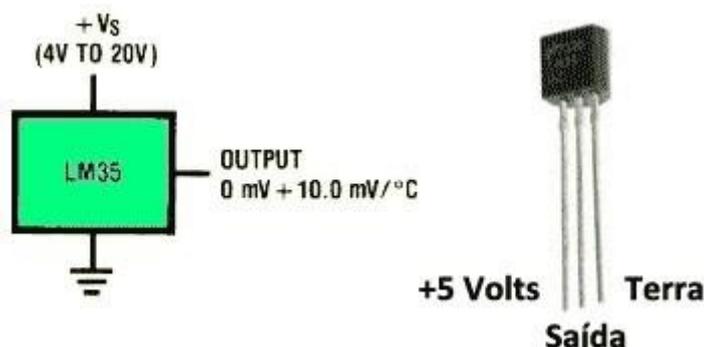
Figura 5: Interface do Arduino



### 3.2.2 LM35

O LM35 (figura 6) é um sensor de temperatura, bastante utilizado no meio microeletrônico por possuir uma ótima resposta a variações e uma boa precisão (Tabela 2).

Figura 6: LM35



FONTE: O SENSOR..., 2016

Este sensor funciona a partir da variação de tensão em seus terminais, de modo que cada 10mV de variação corresponde a 1°C. Sendo assim, a utilização deste sensor nos mais diversos projetos é geralmente uma ótima opção também pela simplicidade de funcionamento e precisão que ele oferece.

Tabela 2: Datasheet do LM35

<b>Modelo</b>	<b>LM35</b>
Fonte de Alimentação	4V - 20V CC
Sinal de saída	Sinal digital via single-bus
Faixa de Operação	Temperatura: -60 - 180 °C
Precisão	< +/- 0,5 °C
Resolução ou sensibilidade	0,1 °C
Repetibilidade	+/- 0,2 °C
Período de captação do sensor	Média: 2 segundos

Sua alimentação pode ser feita utilizando tensão de 4 a 20 Vdc, drenando apenas 60  $\mu$ A da fonte de alimentação. Isso faz com que este possua um baixo auto aquecimento.

### 3.2.3 O Cooler

O cooler (figura 7) é o atuador deste sistema de controle. Sua potência é controlada por PWM (modulação por largura de pulso), o que faz com que sua intensidade de trabalho altere de acordo com a necessidade de alteração da temperatura do ambiente onde este está inserido, sempre em busca de alcançar o valor de *setpoint*. A tensão de funcionamento do Cooler utilizado é de 12V.

Figura 7: Cooler




### 3.2.4 O Transistor TIP 31C

O TIP31C (figura 8) é um transistor NPN de junção bipolar (Tabela 3).

Figura 8: TIP31C



FONTE: DNATECHINDIA..., 2017

Tabela 3: Datasheet do TIP31C

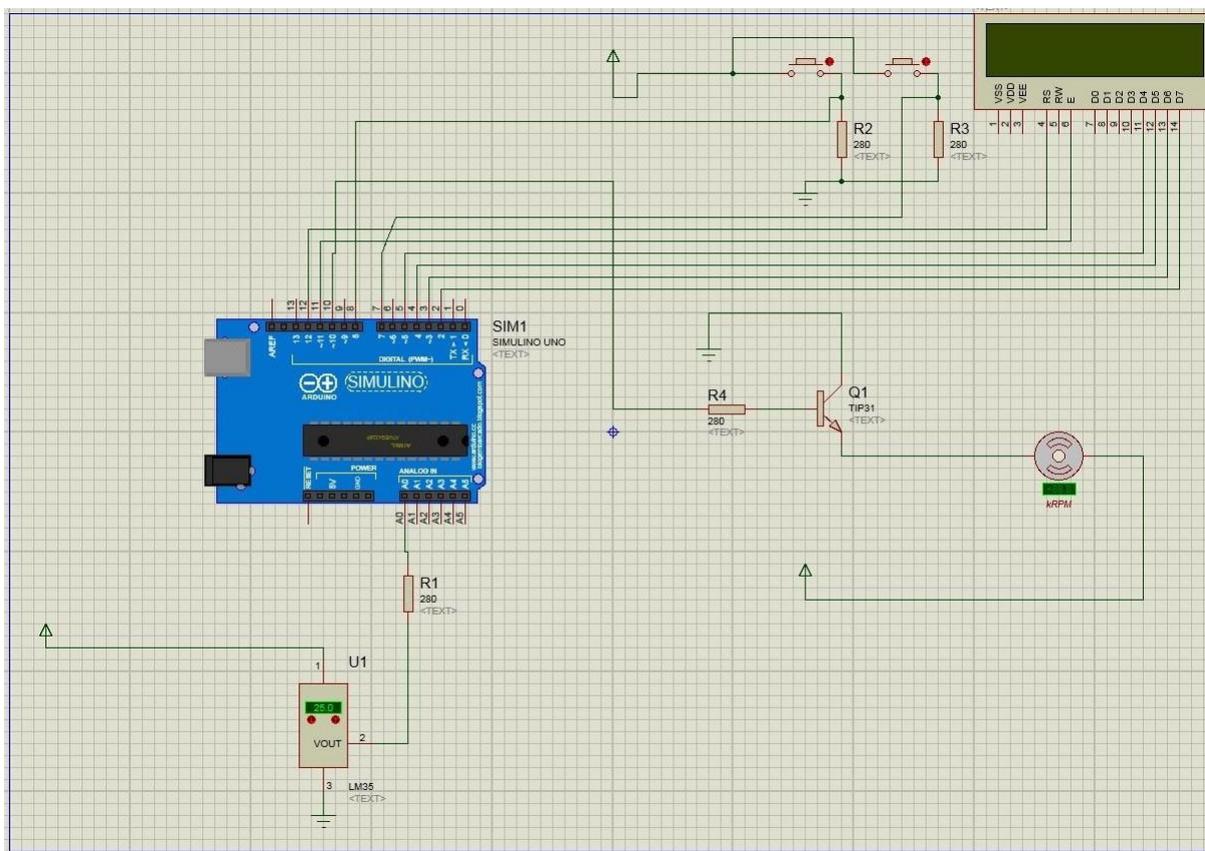
<b>Modelo</b>	<b>TIP31C</b>
Voltagem Coletor-Emissor	100 V
Voltagem Coletor-Base	100 V
Voltagem Emissor-Base	5 V
Pico de corrente contínua no coletor	5 A
Corrente da base	1 A
Dissipação de potência (25°C)	40 W
Temperatura de atuação	- 65° C à 150°C

Aqui neste trabalho, ele funciona como uma chave reguladora de tensão. Recebe em sua base um sinal proveniente do Arduino, o qual fornece a ele determinada tensão. Dessa forma, ao curto-circuitar seu coletor e emissor, o transistor deixa passar para o Cooler uma tensão proporcional ao sinal recebido pelo Arduino.

### 3.3 O Circuito de Controle

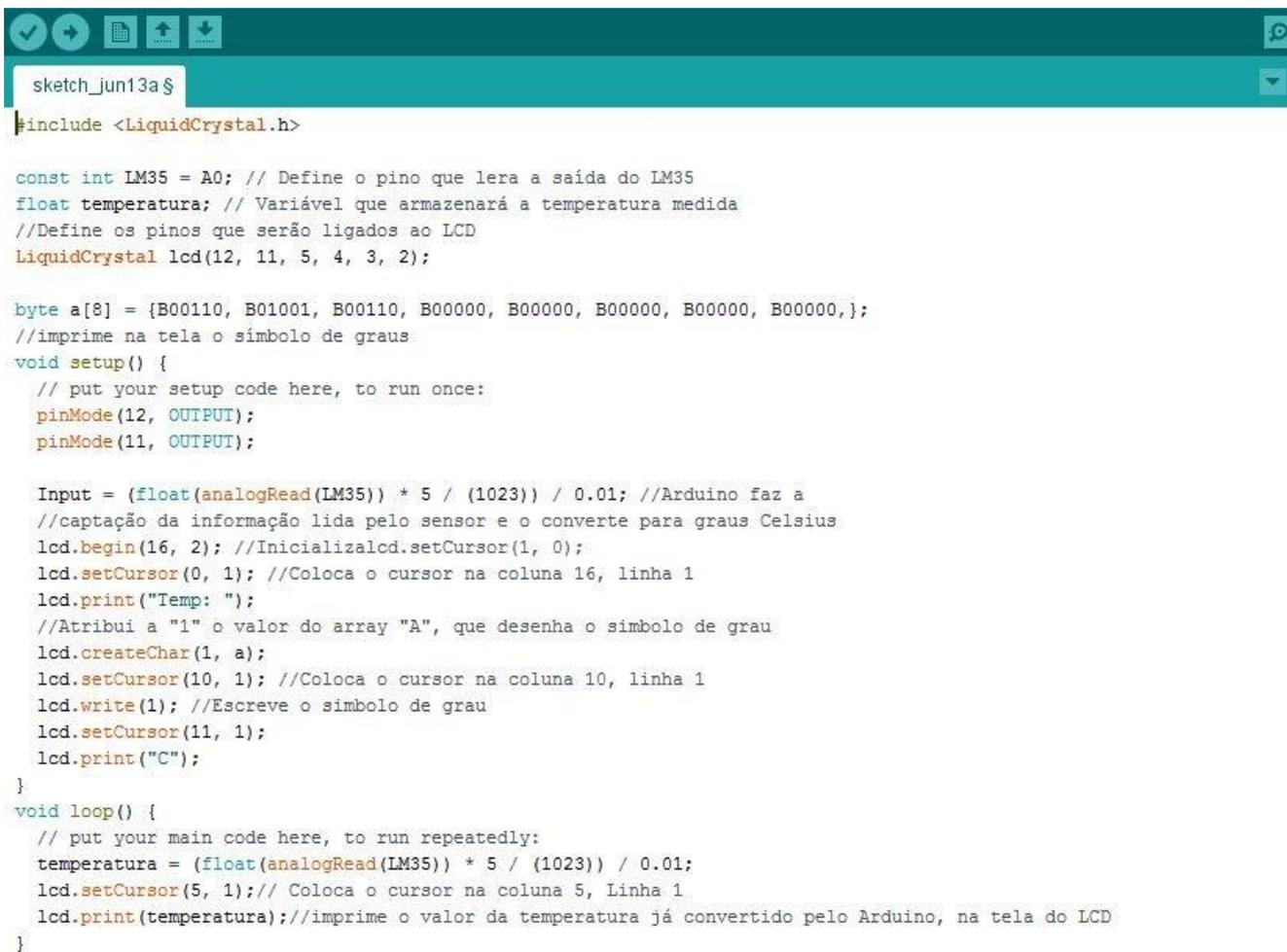
A interligação dos componentes descritos no capítulo 3 dá o circuito de controle. Este circuito foi graficamente exemplificado utilizando o *software* Proteus 8 Professional, o qual permite simular e demonstrar melhor como o projeto foi desenvolvido na prática (Figura 9). Nem todas as conexões estão representadas fielmente pois o *software* possui algumas limitações e a implementação do Arduino no *software* Proteus, que ainda é recente e não conta com o uso de todas as possibilidades reais que o Arduino oferece.

Figura 9: Circuito de controle simulado no Proteus



Primeiramente, é inicializado o monitor LCD, que apresenta as seguintes informações: temperatura ambiente atual, temperatura desejada e potência percentual do cooler. O sensor de temperatura LM35 funciona como uma resistência, que permite a passagem de determinada tensão de acordo com a temperatura ambiente (Figura 10). Dessa forma, a cada leitura, envia para a porta AO do Arduino a tensão atual registrada. O Arduino, por sua vez, realiza a conversão da informação recebida para uma temperatura em graus Celcius. Além disso, está alimentado pelos 5V do Arduino e aterrado ao GND do mesmo hardware. Pode-se ver na figura 11 a programação interna feita por meio do software Arduino para a realização do processo de acionamento do LCD, sensoriamento, leitura e conversão feitas pelo Arduino.

Figura 10: Conversão de tensão para temperatura e impressão no display LCD



```

sketch_jun13a $
#include <LiquidCrystal.h>

const int LM35 = A0; // Define o pino que lera a saída do LM35
float temperatura; // Variável que armazenará a temperatura medida
//Define os pinos que serão ligados ao LCD
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

byte a[8] = {B00110, B01001, B00110, B00000, B00000, B00000, B00000, B00000,};
//imprime na tela o símbolo de graus
void setup() {
  // put your setup code here, to run once:
  pinMode(12, OUTPUT);
  pinMode(11, OUTPUT);

  Input = (float(analogRead(LM35)) * 5 / (1023)) / 0.01; //Arduino faz a
  //captação da informação lida pelo sensor e o converte para graus Celsius
  lcd.begin(16, 2); //Inicializa lcd.setCursor(1, 0);
  lcd.setCursor(0, 1); //Coloca o cursor na coluna 16, linha 1
  lcd.print("Temp: ");
  //Atribui a "1" o valor do array "A", que desenha o simbolo de grau
  lcd.createChar(1, a);
  lcd.setCursor(10, 1); //Coloca o cursor na coluna 10, linha 1
  lcd.write(1); //Escreve o simbolo de grau
  lcd.setCursor(11, 1);
  lcd.print("C");
}

void loop() {
  // put your main code here, to run repeatedly:
  temperatura = (float(analogRead(LM35)) * 5 / (1023)) / 0.01;
  lcd.setCursor(5, 1); // Coloca o cursor na coluna 5, Linha 1
  lcd.print(temperatura); //imprime o valor da temperatura já convertido pelo Arduino, na tela do LCD
}

```

A partir do momento em que a temperatura é mensurada, é possível colocar em ação o controle PID presente na biblioteca do Arduino, cujos detalhes foram expostos na seção 2.3 deste trabalho. A função PID presente na biblioteca do *software* necessita apenas que se passe alguns parâmetros para que esta função retorne o valor de realimentação para a planta, da seguinte forma:

PID (Input, Output, Setpoint,  $K_p$ ,  $K_i$ ,  $K_d$ , Direction)

PID -> chama a função PID;

Input -> passa-se para o controlador a variável que deseja-se controlar, no caso, a temperatura ambiente;

Output-> valor fornecido pelo controlador, que por meio da programação deve-se fazer com que o mesmo controle o atuador (o Cooler) para se atingir o Setpoint;

Setpoint -> fornece-se por meio deste parâmetro o valor ideal de temperatura a ser atingido pelo sistema. É a partir deste que o sistema se adequará para atingí-lo.

$K_p$ ,  $K_i$ ,  $K_d$  -> valores das variáveis que determinam a influência que cada componente do controlador terá sobre a saída;

Direction-> Informa ao controlador se o resultado de realimentação deve ser passado de forma direta ou invertida ao sistema. Pode-se observar a declaração e o uso da função PID na figura 11.

Figura 11: Utilização da função PID do Arduino

```
1  #include <LiquidCrystal.h>
2
3  double Setpoint, Input, Output;//declaração dos parâmetros da função PID
4  int tempset;//variável de Setpoint da temperatura
5  //Especificacao dos links e parametros iniciais de ajuste
6  PID myPID(&Input, &Output, &Setpoint, 1, 0, 0, DIRECT);
7  float temperatura; // Variável que armazenará a temperatura medida
8  int fan = 9; //Declara Pino do Cooler
9
10 void setup() {
11     //ligando a função PID do Arduino
12     myPID.SetMode(AUTOMATIC);
13 }
14
15 void loop() {
16     Setpoint = tempset;
17     Input = temperatura;
18     myPID.Compute();
19     int saida = map(Output, 0, 30, 0, 255);
20
21     int velocidad = map(temperatura, 20, 32, 0, 255);
22     //analogWrite(fan, velocidad);
23     analogWrite(9, velocidad + saida);
24
25 }
26
27
28
```

Para o controle do *cooler* a partir da função PID, utiliza-se a porta 9 do Arduino que possui função de PWM. O PWM (*Pulse width Modulation* ou modulação por largura de pulso) atua em um sinal ou, como neste caso, em uma fonte de alimentação, modulando sua função cíclica e permitindo o controle da potência que alimenta o atuador (*Cooler*). Dessa forma, o resultado da função PID é enviado em forma de valor previamente convertido pelo Arduino, para a porta PWM que emite sinal proporcional ao mesmo. A porta PWM pode emitir sinais de com valores de 0 a 255, de acordo com a necessidade de ajuste do sistema.

Na linha 19 do código mostrado na figura 11, é possível ver o mapeamento para que a saída do PID, que estará entre 0 e 30, seja convertida para a porta PWM, que emitirá sinal entre 0 e 255. Essa conversão é feita a partir de uma regra de três realizada internamente pelo Arduino. O mesmo tipo de mapeamento ocorre na linha 21, mas dessa vez fazendo com que a temperatura lida pelo sensor (entre 20 e 32 graus) seja convertida para valores PWM, entre 0 e 255.

Na figura 12 é possível ver o sistema já implementado e em funcionamento, acoplado ao protótipo de edificação reduzido. A tela apresenta o ciclo de trabalho do *cooler* em relação ao máximo que pode atingir (Fan), além do *setpoint* de temperatura escolhido pelo usuário (Set) e a temperatura atual lida pelo sensor (Temp).



Figura 12: Sistema de Controle de Temperatura instalado no protótipo e tela de apresentação de dados

### 3.4 A Sintonia do PID

Os ganhos ou parâmetros para ajuste do PID podem ser obtidos por diferentes métodos. Neste trabalho adotou-se o método de tentativa e erro. Após entender o significado de cada parâmetro, este método torna a sintonia relativamente fácil. O primeiro passo consiste em definir os termos I e D para 0. Então pode-se aumentar o ganho proporcional P até se perceber uma oscilação na resposta, que deve girar em torno do *Setpoint*. Percebe-se um ganho na velocidade com que o sistema reage à mudança de *Setpoint*. Deve-se tomar cuidado com a estabilidade do sistema. A partir disso, pode-se alterar o termo integral do controlador. Um aumento neste termo proporciona uma redução no erro estacionário, ou seja, na diferença entre a temperatura atual e o *Setpoint*, mas pode gerar um *overshoot*

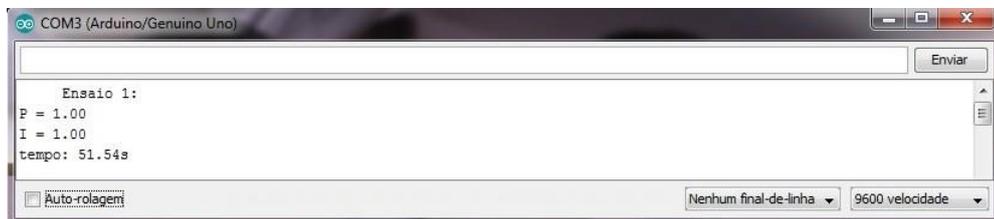
indesejado.

Por fim, pode-se alterar ainda o valor do termo derivativo, que gera uma diminuição do *overshoot*, um aumento no ganho e mantém a estabilidade, proporcionando assim um sistema bastante sensível a ruídos. Este último parâmetro costuma ser utilizado apenas em sistemas que exigem maior precisão e controle, e que lidam com mudanças bruscas e rápidas de parâmetros externos.

## 4 RESULTADOS

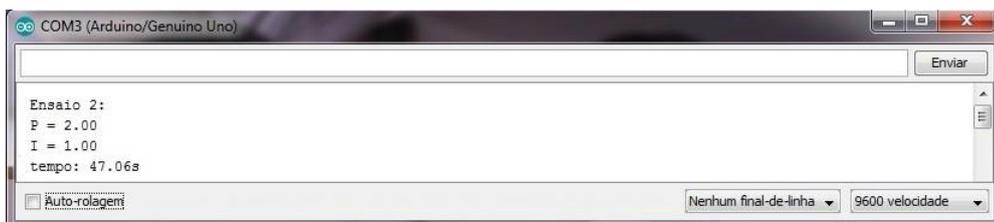
Como visto na seção 3.4 deste trabalho, após a montagem do sistema físico e programação do código controlador do sistema, utilizou-se o método de tentativa e erro para sintonizar o PID. A alteração do parâmetro Derivativo foi descartada, pois o controle de temperatura ambiente, neste caso, não lida com grandes alterações em um curto espaço de tempo. Realizou-se testes utilizando-se objetos que fizessem a temperatura aumentar e diminuir consideravelmente, como gelo e a mão humana. A partir disso, diversos valores para as variáveis Proporcionais e Integrais foram testados, com valores que variaram de 0,1 à 5 para  $K_p$  e de 0 a 200 para  $K_d$ , a partir da observação de trabalhos anteriores similares. Estabeleceu-se o objetivo de conseguir atingir a temperatura desejada em 30 segundos ou menos. Os ensaios mostrados nas figuras 13, 14, 15, 16 e 17 mostram a evolução do sistema com a alteração dos parâmetros, de acordo com o tempo gasto para atingir a temperatura de 25°C após se ter elevado a temperatura até 30°C. Na figura 13 mostra-se os valores do primeiro ensaio.

Figura 13: Ensaio 1



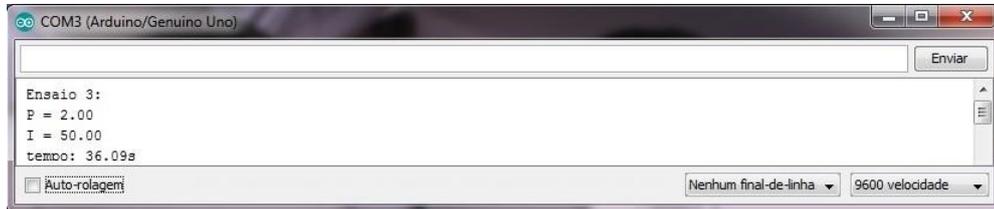
No segundo ensaio (figura 14), duplicou-se o valor de P e notou-se uma sensível diminuição no tempo de resposta.

Figura 14: Ensaio 2



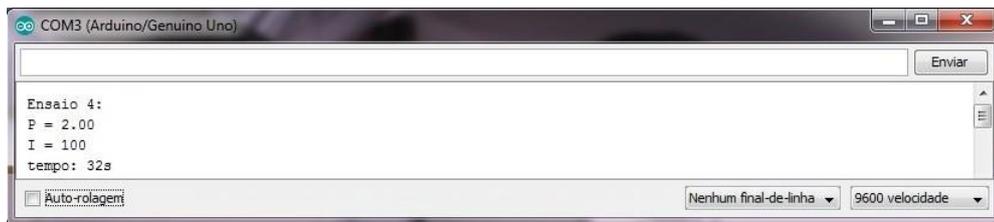
No terceiro ensaio (figura 15), alterou-se consideravelmente o valor do parâmetro integral (I), e obteve-se uma boa melhora no tempo de resposta e estabilização. Em um quarto

Figura 15: Ensaio 3



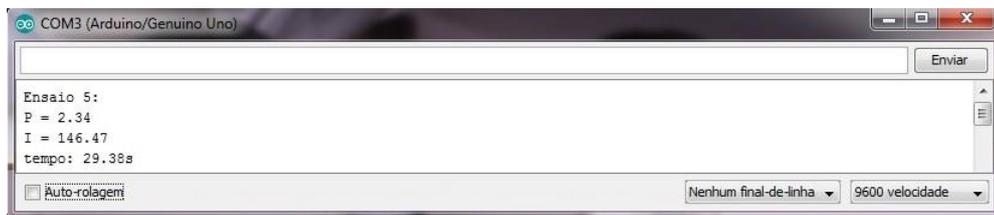
ensaio (figura 16), dobrou-se o valor do parâmetro integral (I), melhorando ainda mais o tempo de resposta e estabilização, aproximando-se do valor ideal fixado (30s).

Figura 16: Ensaio 4



No ensaio 5 (figura 17), aumentou-se com maior precisão o valor de I, alcançando-se a temperatura desejada em um tempo estabelecido como ótimo, abaixo dos 30 segundos.

Figura 17: Ensaio 5



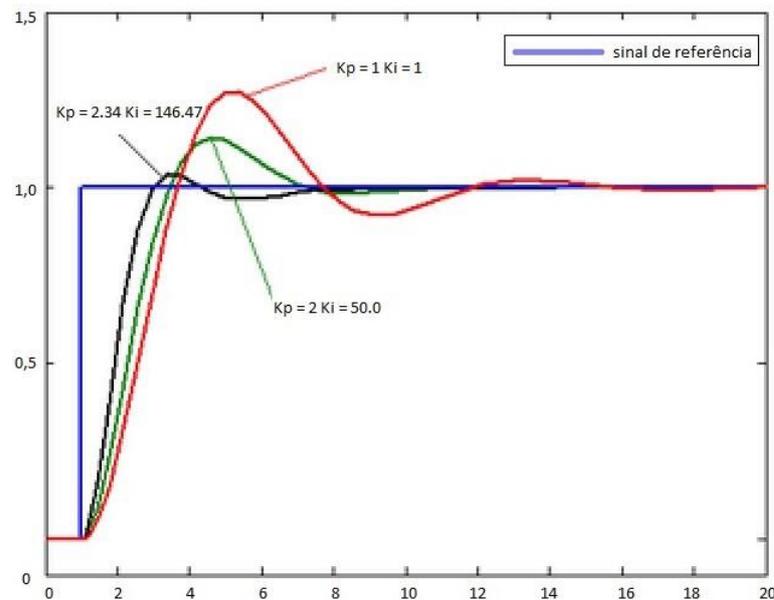
Os valores dos parâmetros eram alterados não somente em busca de um tempo ideal, mas também porque os resultados por vezes geravam grandes *overshoots*, grande instabilidade ou um grande erro estacionário. A partir disso, os valores definidos a partir do ensaio 5, para o uso nesta situação, estão descritos na tabela 4.

O sistema apresentou, com esses valores, um ótimo tempo de resposta e grande estabilidade ao atingir a temperatura desejada. No gráfico apresentado na figura 18 pode-se observar a curva de reação para os ensaios 1, 3 e 5 e perceber a evolução do controle, apresentando ao fim uma curva mais suave, com um tempo de resposta desejável, pequeno overshoot e tempo de acomodação satisfatório.

Tabela 4: Valores da Sintonia PID

$K_p$	$K_i$	$K_d$
2,34	146,47	0,0

Figura 18: Curvas de reação dos ensaios



Ao se atingir os valores finais de calibragem utilizados, realizou-se mais alguns testes, considerando uma temperatura ambiente igual a 30°C. A princípio escolheu-se um *setpoint* igual a 23°C e esperou-se pela estabilização do sistema (aproximadamente 300 segundos). O *cooler* apresentou um ciclo de trabalho igual a 33% da sua capacidade. Após certo tempo de estabilidade (1000 segundos aproximadamente), alterou-se o *setpoint* para 20°C, e esperou-se outra vez pela estabilidade do sistema (aproximadamente 400 segundos). Dessa vez, apesar de apresentar maior oscilação (algo em torno de 3°C), o sistema ainda atendeu as necessidades deste processo, e o *cooler* apresentou um ciclo de trabalho igual a 100% da sua capacidade. Voltou-se o *setpoint* para 20°C e novamente obteve-se um bom resultado. Por fim retornou-se o *setpoint* para 25°C e a resposta foi ainda melhor, com o processo se estabilizando em aproximadamente 200 segundos, e mantendo uma temperatura bastante próxima à 25°C, com o *cooler* novamente atuando com 33% de sua capacidade.

No início dos testes para se alcançar valores satisfatórios para o sistema de controle, alguns problemas foram encontrados, como um tempo de resposta muito grande e *overshoot* alto demais. Ao fim chegou-se a valores dentro do esperado, com os quais o sistema realmente demonstrou eficiência.

## 5 CONCLUSÃO

Este trabalho foi desenvolvido com o intuito de criar um protótipo de um sistema de controle de temperatura, capaz de regular o resfriamento do ambiente onde está inserido. Utilizou-se para isso a biblioteca PID do Arduino e uma porta PWM, ferramentas que facilitam a implementação e fornecem resultados bastante expressivos, precisos e eficientes.

O microcontrolador utilizado foi o Arduino, específico para prototipagem de projetos deste tipo, pela facilidade que traz na programação, montagem e conexão com demais *hardwares*. É uma poderosa ferramenta para o início de projetos de controle, automação e robótica, entre outros. Neste caso, ele teve a importante função de interpretar e converter dados, bem como receber sinais como do sensor de temperatura, e emitir sinais, como para o transistor que faz o controle do atuador (*cooler*). Além disso, foram as bibliotecas existentes e possíveis de serem utilizadas no Arduino que permitiram um controle eficiente como inicialmente foi proposto.

A matemática envolvida no uso do PID, bem como no mapeamento de parâmetros, que fazem parte da metodologia utilizada neste trabalho, são ferramentas que podem ser utilizadas no desenvolvimento de diversas outras aplicações. Além disso, a possibilidade de mudança do *setpoint* do sistema o torna ainda mais dinâmico e eficaz, eficiente para diversos tipos de projetos.

O uso de um microcontrolador como o Arduino se mostra por muitas vezes mais prático e eficaz do que o uso de PICs, os quais exigem programação mais complexa e ainda o uso de compiladores mais complexos e gravadores físicos. O Debug destes é também mais difícil de ser realizado.

Para trabalhos futuros, sugere-se a implementação de uma fonte de aquecimento também, para que o controlador seja ainda mais robusto e possa ser útil também para temperaturas ambientes frias, aquecendo o ambiente. Sugere-se ainda a utilização de valores diferentes para os parâmetros  $K_p$ ,  $K_i$  e  $K_d$ , observando-se diferenças no tempo de resposta e na precisão do controle, na medida em que se usa um, dois ou três destes parâmetros. Finalmente, a sugestão é que a partir do protótipo criado desenvolva-se um sistema mais robusto e comercializável.

# REFERÊNCIAS

AGUIRRE, L.A. **Introdução à Identificação de Sistemas: Técnicas Lineares e Não-Lineares Aplicadas a Sistemas Reais**. 3. ed. rev. e ampliada. Belo Horizonte: UFMG, 2007.

AMERICANLAB – Equipamentos para Laboratórios. **Manual do Usuário**. – Banho Maria Refrigerado Com Circulação Ultratermostatizado. Modelo AL152/10/D, 2007.

ÅSTRÖM, K. J; HÄGGLUND, T. **PID Controllers: Theory, Design and Tuning**. Instrument Society of America, 2 ed. 1995

BENEDICT, R.P., **Fundamentals of Temperature, Pressure and Flow Measurements**., New York, John Wiley, 1977.

DORF, R. C.; BISHOP, R. H. **Sistemas de Controle Modernos**. 8 ed. : Editora LTC, 1998.

FRANSOZO, H. L. **Avaliação do desempenho de habitações de baixo-custo estruturadas em aço**. 2001, 244p. Dissertação (Mestrado em Engenharia Civil) – Programa de Pós Graduação em Engenharia Civil, Departamento de Engenharia Civil, Universidade Federal de Ouro Preto, Ouro Preto, 2001.

GUERRA, W.A. **Implementação de Controle Proporcional, Integral e Derivativo Digital em Controladores Lógico Programáveis**. Recife, 2009.

SIMÕES, M.G.; SHAW, I.S. **Controle e Modelagem Fuzzy**. 2 ed. : Editora Blucher. São Paulo, 2007.

SOUZA, H. A.; FRANSOZO, H.L.; BRAGA, G. A.; FERNADES, W. L. Avaliação do desempenho térmico dos sistemas de fechamento pré-fabricados utilizados na construção estruturada em aço. In: ENCONTRO NACIONAL SOBRE CONFORTO NO AMBIENTE CONSTRUÍDO, 7, 2003, Curitiba. **Anais...** Curitiba: ENCAC-COTEDI, 2003. p.1236-1244.

BARBOSA, T.S. **Eficiência térmica de habitação de baixo custo estruturada em aço**. Poços de Caldas, 2016.

COELHO, A. A. R. et al. **Projeto e estudo de caso da implementação de um sistema de controle nebuloso**. Campinas, 2003.

**CONHECENDO** as partes do Arduino, 2015. Blog Natal Makers. Disponível em:

<<http://natalmakers.blogspot.com.br/2015/08/dispositivo-conhecendo-as-partes-do.html>>.

Acesso em: 25 mar 2017.

**PROJETO** Básico de Controle, 2013. Blog Arduino by myself. Disponível em:

<<https://arduino.bymyself.blogspot.com.br/2013/04/control-de-luminosidade-com-arduino-pid.html>>. Acesso em: 20 abr. 2017.

**O SENSOR** LM35, 2016. Blog Nova Eletrônica. Disponível em:

<<http://blog.novaeletronica.com.br/img/sensor-lm35.gif>> Acesso em: 28 nov. 2016.

**TIP31C** Power transistor, 2016. DNA Tech India. Disponível em <<http://www.dnatechindia.com/tip31c-npn-power-transistor.html>>.

Acesso em: 20 jun 2017.

USITETO – Solução Usiminas para habitação popular: detalhamento técnico. Belo Horizonte: USIMINAS. Gerência de Desenvolvimento da Aplicação do Aço-DGA. 2001



## APÊNDICE A - Código utilizado no Arduino

```

include <LiquidCrystal.h>
include <PID_v1.h> //bibliotecaquepermiteutilizaçãodoPID
double Setpoint, Input, Output; //declaração dos parâmetros da função PID

int tempset; //variável de Setpoint da temperatura

//Especifica as conexões e os parâmetros de sintonia
PID myPID(Input, Output, Setpoint, 1, 0, 0, DIRECT);

//Define os pinos que serão ligados ao LCD
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

const int LM35 = A0; // Define o pino que lerá a saída do LM35

float temperatura; // Variável que armazenará a temperatura medida

byte a[8] = B00110, B01001, B00110, B00000, B00000, B00000, B00000, B00000;

int fan = 9; //Declara pino PWM de controle do do Cooler

void setup() {

  pinMode(8, INPUT_PULLUP);
  pinMode(10, INPUT_PULLUP);
  pinMode(12, OUTPUT);
  pinMode(11, OUTPUT);

  Input = (float(analogRead(LM35)) * 5 / (1023)) / 0.01; //Inicializa o PID
  myPID.SetMode(AUTOMATIC);

  lcd.begin(16, 2); //Inicializa lcd
  lcd.setCursor(1, 0);
  lcd.print("Fan: ");
  lcd.setCursor(12, 0);
  lcd.print("lcd.setCursor(0, 1); //Coloca o cursor na coluna 16, linha 1
  lcd.print("Temp: ");
  //Atribui a "1"o valor do array "A", que desenha o símbolo de grau

  lcd.createChar(1, a);

```

```
lcd.setCursor(10, 1); //Coloca o cursor na coluna 16, linha 1
lcd.write(1); //Escreve o simbolo de grau
//lcd.setCursor(10, 1);
//lcd.print("C");

}

void loop() {

if (digitalRead(8) == LOW & tempset < 30) {
tempset++;
delay(500);
}
else if (digitalRead(10) == LOW & tempset > 15) {
tempset--;

delay(500);
}
Setpoint = tempset;
temperatura = (float(analogRead(LM35)) * 5 / (1023)) / 0.01;
lcd.setCursor(5, 1);
lcd.print(temperatura);
Input = temperatura;
myPID.Compute();
int saida = map(Output, 0, 30, 0, 255);
lcd.setCursor(5, 1);
lcd.print(temperatura);

Serial.println(temperatura);
int velocidad = map(temperatura, 20, 32, 0, 255);
//analogWrite(fan, velocidad);
analogWrite(9, velocidad + saida);
int pcent = map(velocidad, 0, 255, 0, 100);
lcd.setCursor(6, 0);
lcd.print(pcent);
delay(2000);
lcd.setCursor(11, 0);
lcd.print();
```

```
lcd.setCursor(11, 1);  
lcd.print(tempset);  
}
```