



**Ministério  
da Educação**

Secretaria de Educação Profissional e Tecnológica  
Instituto Federal de Educação, Ciência e Tecnologia de  
Santa Catarina

**CÂMPUS FLORIANÓPOLIS  
DEPARTAMENTO ACADÊMICO DE METAL-MECÂNICA  
BACHARELADO EM ENGENHARIA MECATRÔNICA**

**LUANA NUNES DA SILVA**

# **CONTROLE PID DE TEMPERATURA COM INTEGRAÇÃO DE DADOS EM REDE**

**FLORIANÓPOLIS, DEZEMBRO DE 2018**



**INSTITUTO FEDERAL  
SANTA CATARINA**

**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA**

**CÂMPUS FLORIANÓPOLIS**

**DEPARTAMENTO ACADÊMICO DE METAL-MECÂNICA**

**BACHARELADO EM ENGENHARIA MECATRÔNICA**

**LUANA NUNES DA SILVA**

**CONTROLE PID DE TEMPERATURA COM INTEGRAÇÃO DE DADOS EM REDE**

Trabalho de Conclusão de Curso submetido ao Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina como parte dos requisitos para a obtenção do título de Bacharel em Engenharia Mecatrônica

Professor Orientador: Valdir Noll

**FLORIANÓPOLIS, DEZEMBRO DE 2018.**

Ficha de identificação da obra elaborada pelo autor.

Silva, Luana Nunes da  
Controle PID de Temperatura com Integração de Dados  
em Rede / Luana Nunes da Silva ; orientação de Valdir Noll.  
- Florianópolis, SC, 2018.  
65 p.

Trabalho de Conclusão de Curso (TCC) - Instituto Federal  
de Santa Catarina, Câmpus Florianópolis. Bacharelado  
em Engenharia Mecatrônica. Departamento  
Acadêmico de Metal Mecânica.  
Inclui Referências.

1. Gerenciamento Remoto. 2. Controle PID. 3. ESP8266.  
4. NodeMcu. 5. Internet das Coisas. I. Noll, Valdir.  
II. Instituto Federal de Santa Catarina. Departamento  
Acadêmico de Metal Mecânica. III. Título.


## **CONTROLE PID DE TEMPERATURA COM INTEGRAÇÃO DE DADOS EM REDE**

**LUANA NUNES DA SILVA**

Este trabalho foi julgado adequado para obtenção do título de Bacharel em Engenharia Mecatrônica e aprovado na sua forma final pela banca examinadora do Curso de Bacharelado em Engenharia Mecatrônica do Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina.


Florianópolis, 13 de dezembro de 2018.

Banca examinadora:



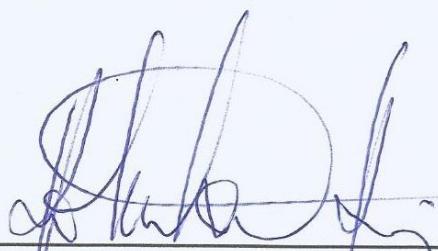
---

Prof. Valdir Noll, Dr.  
Orientador



---

Prof. Cynthia Beatriz Scheffer Dutra, Dr.  
Membro Titular



---

Prof. Roberto Alexandre Dias, Dr.  
Membro Titular

A todos aqueles que de alguma forma  
estão e estiveram próximos a mim,  
fazendo cada vez mais tudo valer a pena.

## **AGRADECIMENTOS**

Primeiramente agradeço a Deus, por ter me dado forças para ir até o final desse segundo desafio proposto a mim que foi essa segunda graduação, dentro desta instituição de ensino que me acolheu, e espero que não seja a última.

Agradeço a minha família, Renata Lu, Fernando, Joaquim, Marcos, Maria Loy, Bertolino, Nilton, Lizete, Denise, Douglas, Pedro, e ao meu gato Smile, que estiveram sempre do meu lado e me deram todo o apoio necessário para ir até o fim deste objetivo.

Ao meu namorado Carlos Eduardo, que teve paciência, carinho, compreensão, amor e sempre esteve do meu lado, me incentivando, me ajudando e me motivando a ir até o final deste desafio, e a sua família.

Ao professor Valdir, que aceitou ser meu orientador deste projeto e esteve sempre do meu lado, me desafiando e me ajudando. Espero que todas as pessoas possam ter orientadores que nem o senhor.

A professora Cynthia que conseguiu me dar uma luz para o encaminhamento do meu estudo, quando eu estava confusa.

A professora Claudia, que se prontificou a me ajudar a corrigir o documento mesmo quando quase não havia mais tempo.

Aos meus amigos, que sempre estiveram do meu lado e entenderam minhas ausências e crises, e me ajudaram, me motivaram a ir até o final.

A empresa REIVAX S.A, aos meus chefes Gilvan e Rodrigo, ao Cleber, ao Henrique e a todos meus colegas de trabalho por terem me compreendido e me apoiado neste momento complicado.

E ao final, mas não menos importante, a todos os mestres desta instituição que passaram pela minha jornada, sempre me desafiando e me instigando a ser uma melhor aluna, uma profissional melhor.

Finalizo este projeto com o coração pleno e feliz por ter conseguido vencer mais esta etapa.

A todos vocês, o meu muito obrigada.

A educação é a arma mais poderosa  
que você pode usar para mudar o mundo.  
(Nelson Mandela)

## RESUMO

Este projeto consiste em desenvolver um sistema elétrico-eletrônico para realizar o controle de temperatura de uma resistência elétrica. O modo de medir a temperatura será por termopar tipo K. O controle de temperatura será por meio da implementação de um controlador do tipo PID, cuja ação de controle será um comando PWM, através do microcontrolador ESP8266, na plataforma NodMCU. O *software* para acionar uma resistência elétrica e realizar o controle da temperatura será feito através de uma plataforma na *WEB* (uma página HTML), tentando utilizar os conceitos de internet das coisas. Nessa interação, o usuário também receberá informações da temperatura atual e do seu histórico, podendo configurar e acompanhar e o desenvolvimento do controle por meio da temperatura medida. O controlador desenvolvido permite alterar o valor de referência para o controle e ligar/desligar a ação de controle. Além disso, permite visualizar graficamente o comportamento da temperatura e do valor da ação de controle. Os resultados alcançados mostram que o gerenciamento remoto é uma boa ferramenta para administrar sistemas de controle que não exigem velocidades de respostas muito rápidos, tais como o controle de temperatura.

**Palavras-Chave:** Gerenciamento Remoto. Controle PID. ESP8266. NodeMCU. Internet das Coisas.



## **ABSTRACT**

This project consists of developing an electrical-electronic system to perform the temperature control of an electrical resistance. The temperature control will be by means of type K thermocouple. The temperature control will be by means of the implementation of a PID controller, whose control action will be a PWM command, through the ESP8266 microcontroller, in the platform NodMCU. The software to trigger an electrical resistance and perform temperature control will be done through a WEB platform (an HTML page), trying to use the concepts at internet of things. In this interaction, the user will also receive information about the current temperature and history, being able to configure and monitor and the development of the control through the measured temperature. The developed controller allows changing the reference value for the control and on / off control action. In addition, it allows graphical visualization of the behavior of the temperature and the value of the control action. The results show that remote management is a good tool for managing control systems that do not require very fast response speeds, such as temperature control.

Keywords: Remote Management. PID Controller. ESP8266. NodeMCU. Internet of Things.

## LISTA DE FIGURAS

Figura 1 - Diagrama de blocos de um sistema de controle em malha aberta .....	20
Figura 2 - Diagrama de blocos de um sistema de controle em malha fechada .....	21
Figura 3 - Esquemático simplificado do AD620 .....	23
Figura 4 - Esquemático operacional do AD620 .....	24
Figura 5 - Placa reguladora de tensão 24V-5V.....	25
Figura 6 - Protótipo inicial.....	26
Figura 7 - Placa com amplificador de instrumentação AD620 .....	28
Figura 8 - Esquemático da placa do AD620 .....	28
Figura 9 - Plataforma NodeMCU .....	29
Figura 10 – Arduino IDE .....	30
Figura 11 - Leitura do AD em malha aberta.....	31
Figura 12 - Gráfico da Relação da Temperatura do Termopar Tipo K .....	32
Figura 13 - Temperatura em Malha Aberta .....	33
Figura 14 – Gráfico em malha aberta, obtido através do levantamento de pontos do sistema.....	34
Figura 15 - Gráfico Método Ziegler - Nichols.....	36
Figura 16 - Gráfico Método Smith .....	38
Figura 17 - Gráfico Método Sundaresan & Krishnaswamy.....	39
Figura 18 - Comparação dos Métodos com Curva Real .....	41
Figura 19 - Controladores com melhores resultados com a curva em malha aberta .....	42
Figura 20 - Curva em malha fechada com coeficientes calculados.....	45
Figura 21 - Código de programação - 1 .....	47
Figura 22 - Código de programação - 2 .....	48
Figura 23 - Código de programação - 3 .....	48
Figura 24 - Código de programação - 4 .....	49
Figura 25 - Código de programação - 5 .....	50
Figura 26 - Código de programação - 6 .....	51
Figura 27 - Timers .....	51
Figura 28 - Código de programação - 7 .....	52
Figura 29 - Código de programação - 8 .....	53
Figura 30 - Código de programação - 9 .....	53
Figura 31 – Ambiente de desenvolvimento Node-RED .....	55

Figura 32 - Nó de requisição HTTP .....	56
Figura 33 - Exemplo de parametrização de interface gráfica .....	57
Figura 34 - Página HTML desenvolvida ( <i>dashboard</i> ) .....	58
Figura 35 - Valor de referência de temperatura no dashboard .....	58
Figura 36 – Inicia e Desliga Controle no <i>dashboard</i> .....	59
Figura 37 - Gráfico de temperatura do <i>dashboard</i> .....	60
Figura 38 - Gráfico do valor de controle do <i>dashboard</i> .....	60
Figura 39 – Protótipo finalizado .....	61
Figura 40 - Resultados da malha fechada.....	62

## LISTA DE ABREVIATURAS E SIGLAS

AD ou ADC – conversor analógico-digital

HTML – HyperText Markup Language

Kp – Coeficiente de controle proporcional

Ki - Coeficiente de controle integrativo

Kd- Coeficiente de controle derivativo

p – Polo da função

PID - Controlador Proporcional, Integral e Derivativo

RG – Resistor de ganho

s – Raízes da equação

Td – Tempo do controlador derivativo

Ti – Tempo do controle integrativo

Ta – Tempo de acomodamento

Vin – Tensão de entrada do AD

Vout – Tensão de saída do AD

WEB - World Wide Web

WWW – World Wide Web

## SUMÁRIO

1 INTRODUÇÃO .....	15
<b>1.1 Justificativa</b> .....	16
<b>1.2 Definições do problema</b> .....	17
<b>1.3 Objetivo geral</b> .....	17
<b>1.4 Objetivos específicos</b> .....	18
2 PRINCIPAIS CONCEITOS .....	19
<b>2.1 Acesso remoto</b> .....	19
<b>2.2 Controlador PID</b> .....	20
2.2.1 Controle Proporcional (P) .....	21
2.2.2 Controle Proporcional-Integral (PI) .....	21
2.2.3 Controle Proporcional-Derivativo (PD) .....	22
2.2.4 Controle Proporcional-Integral-Derivativo (PID) .....	22
<b>2.3 Transdutor de temperatura</b> .....	23
3 DESENVOLVIMENTO DO SISTEMA .....	25
<b>3.1 Materiais e componentes utilizados</b> .....	25
3.1.1 Placa Reguladora de Tensão 24V-5V .....	25
3.1.2 Desenvolvimento do Hardware .....	26
3.1.3 Plataforma utilizada .....	29
3.1.4 Interface de programação do NodeMCU .....	29
<b>3.2 Controlador pid</b> .....	30
3.2.1 Levantamento da curva em malha aberta .....	31
3.2.2 Controlador PID .....	33
<b>3.3 Software</b> .....	46
3.3.1 Software embarcado .....	46
3.3.2 Página de Gerenciamento WEB .....	54

4 ANÁLISE DOS RESULTADOS .....	61
5 CONCLUSÃO .....	63
REFERÊNCIAS .....	64

## 1 INTRODUÇÃO

Em uma época que caminha para a interligação dos mais diversos tipos de máquinas e sensores, tendo como horizonte a Indústria 4.0, é interessante e importante que se dominem tecnologias de comunicação de dados sem fio, gerenciamento remoto, visualização remota de dados em forma gráfica, e ação remota por meio dos operadores da indústria. O objetivo deste trabalho é afinar o controle de sistemas e análise de informações futuras, visando a otimização dos processos industriais.

O controle de temperatura se destaca nesse cenário, por ser um dos mais utilizados no meio industrial. O controle de temperatura pode ser realizado por diversos fabricantes que disponibilizam controladores dos mais variados formatos e tipos. Apesar disso, é importante estudar esses processos, aliando o conhecimento do controlador em si, e incluindo o gerenciamento remoto do controlador via *WEB*, que trazem flexibilidade ao usuário final do sistema, bem como informações em tempo real sobre o comportamento do processo, permitindo agir no sistema, evitando problemas, ou corrigindo-os antes que ocorram.

Com o aumento da disponibilidade de novos microcontroladores, tais como o ESP8266, da Espressif®, que possuem preço baixo com alto poder de processamento, e que embutem em seus sistemas a capacidade de, com pouco esforço de *software*, entrarem na rede *Wi-Fi* como servidores *WEB*, viabilizou-se ainda mais o gerenciamento remoto, bem como o acesso de dados em servidores localizados fora da própria indústria.

Tendo em mente esses dois aspectos, desenvolveu-se, neste trabalho, o controle de temperatura por um controlador PID clássico, e o gerenciamento remoto deste sistema, procurando mostrar o potencial que tem esse tipo de disponibilidade de dados e visualização gráfica remota, via *WEB*.

## 1.1 JUSTIFICATIVA

A comunicação de dados sem fio tem se tornado, cada vez mais, alvo de estudos e implementações práticas no meio industrial, por permitir ter acesso aos dados de maneira simples, intuitiva. Mantém-se, também, esses dados em banco de dados de maneira automática, permitindo a análise posterior do comportamento de sistemas industriais.

É fato que o uso de comunicação de dados sem fio tem crescido no meio industrial, especialmente com o advento da Indústria 4.0, sendo, portanto, alvo de estudos e pesquisas nessa área.

Também é notório que o problema de controle de temperatura é um dos mais importantes tipos de controle realizados atualmente, pois em quase todas as plantas industriais há a necessidade de se medir e controlar a temperatura de algum processo. Um exemplo, o controle de fornos na indústria metal-mecânica para o forjamento de metais, o cozimento de alimentos na indústria alimentícia, o controle de temperatura na fabricação de polímeros usados na indústria química, dentre outros exemplos.

Aliando-se esses dois aspectos, monitorar e controlar a temperatura por meio de uma interface *WEB* se torna um importante trabalho, no sentido de controlar um processo clássico, e de monitorar em tempo real o seu comportamento por meio de um aplicativo *WEB*, podendo, inclusive, agir no sentido de desligar e alterar variáveis de controle sem estar próximo à planta.

Nesse sentido, justifica-se desenvolver um trabalho de conclusão de curso nessa importante área, além de fornecer conhecimentos práticos importantes para um futuro engenheiro mecatrônico.



## 1.2 DEFINIÇÕES DO PROBLEMA

Conforme exposto na justificativa, este projeto visa monitoramento e controle de temperatura por meio de uma interface *WEB*. Para demonstrar os princípios de funcionamento, propõem-se controlar a temperatura de uma resistência elétrica utilizada em ferro de passar roupa, conectada à um relé de estado sólido e a uma fonte de tensão de corrente alternada. O relé de estado sólido é uma chave eletrônica que recebe um sinal de controle da plataforma de controle, que, além de controlar a temperatura, também envia informações à rede *ethernet* sem fio, para serem analisadas e informadas numa página *WEB*.

Para realimentar o sistema de controle PID, tem-se um sensor de temperatura do tipo termopar, tipo K, fixado à mesma superfície que a resistência, e um amplificador de instrumentação que recebe o valor do termopar, amplifica, e envia para o conversor ADC, para a conversão de temperatura analógica em digital. Esses dados digitais servirão de base para o controle e monitoramento da temperatura.

Por fim, deve-se realizar uma interface gráfica que permita ao usuário interagir com o sistema proposto, de maneira simples e funcional, alterando alguns parâmetros de controle e monitorando a temperatura.

## 1.3 OBJETIVO GERAL

Desenvolver o controle PID da temperatura de um sistema de medição de temperatura e gerenciar esse sistema por meio de um acesso à internet sem fio.

## 1.4 OBJETIVOS ESPECÍFICOS

Para cumprir o objetivo geral, foram traçados os seguintes objetivos específicos:

- a) Pesquisar e estudar os microcontroladores;
- b) Fazer o controle PID da temperatura de um sistema;
- c) Implementar comunicação via *Ethernet*;
- d) Criar uma página na *WEB* para comunicação e mostrar o comportamento da temperatura em forma gráfica;
- e) Interagir com o usuário pela *WEB*, configurando o controle e monitorando os resultados.

## 2 PRINCIPAIS CONCEITOS

Para compreender o funcionamento da eletrônica presente no projeto, é necessário o entendimento das tecnologias empregadas. Como o projeto proposto visa desenvolver um sistema de gerenciamento remoto de temperatura, através de um controlador PID numa plataforma que possibilita a comunicação sem fio, é necessário apresentar os principais conceitos utilizados no desenvolvimento deste projeto.

### 2.1 Acesso remoto

O acesso remoto é uma ferramenta a qual possibilita que o usuário de um determinado sistema possa interagir com ele à distância - isto é, longe fisicamente do sistema determinado. O gerenciamento remoto, então, permite controlar e monitorar esse sistema à distância. Para utilizá-lo, é necessária a conexão com dispositivos externos que permitam a comunicação sem fio (PANORAMA POSITIVO, 2018).

A comunicação sem fio, por sua vez, é uma tecnologia que permite a transmissão de dados entre dispositivos sem a necessidade de um cabo ou fio entre eles, basta que os mesmos estejam conectados a uma mesma rede.

Tais tecnologias têm sido amplamente empregadas em diferentes áreas como, por exemplo, as redes de telefonia celular, a transmissão de dados via satélite e, sobretudo, as redes sem fio com abrangência local, classificadas como *Wireless* LAN, também conhecidas como redes *Wi-Fi*, presentes em ambientes residenciais e corporativos (BOBAK, 2001).

Através de rede *wi-fi*, pode-se realizar esse gerenciamento remoto de um determinado sistema. O sistema aqui proposto possui uma resistência elétrica, a qual receberá um sinal para realizar seu aquecimento, e quem enviará esse sinal, será o controlador PID, que receberá o valor da temperatura em tempo real.

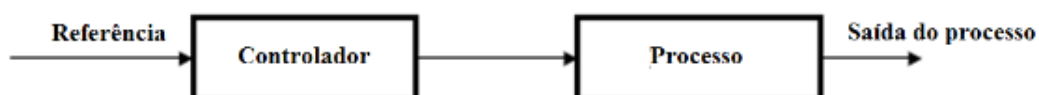
## 2.2 Controlador PID

O controlador PID, ou simplesmente PID, é uma técnica de controle de processos que une as ações derivativa, integrativa e proporcional, fazendo, assim, com que o sinal de erro seja minimizado pela ação do proporcional, zerado pela ação integral e obtido com uma velocidade antecipativa pela ação derivativa.

Para realizar a modelagem do processo, é necessário obter a curva em malha aberta do sistema. Após realizado o modelo do processo, é possível realizar os cálculos necessários para o controle.

Os sistemas de controle de malha aberta são aqueles em que o sinal de saída não exerce nenhuma ação de controle no sistema, neles o sinal de saída não é medido nem utilizado para realimentação do sistema para comparação com a entrada conforme a Figura 1 (OGATA, 2011).

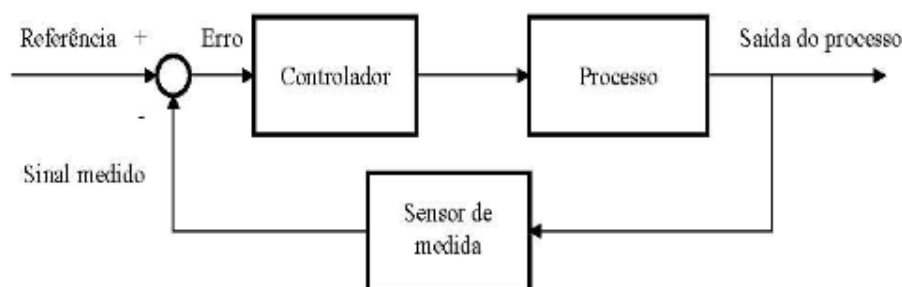
**Figura 1 - Diagrama de blocos de um sistema de controle em malha aberta**



Fonte: OLIVEIRA (1999).

Já no controle em malha fechada, o sinal de saída possui um efeito direto na ação do controle e é designado por um sistema de controle com realimentação. Neste tipo, o sinal de erro que corresponde à diferença entre os valores de referência e de alimentação, é introduzido no controlador de modo a reduzir o erro e a manter a saída do sistema num determinado valor pretendido pelo usuário (OGATA, 2011). A Figura 2 representa este processo.

**Figura 2 - Diagrama de blocos de um sistema de controle em malha fechada**



Fonte: OLIVEIRA (1999).

As características gerais de controle proporcional, integral, derivativo e suas combinações estão descritas abaixo:

### 2.2.1 Controle Proporcional (P)

O controle proporcional é normalmente aplicado em muitos processos contendo constantes de tempo simples, e responde rapidamente tanto aos distúrbios como às alterações do ponto de ajuste. Possui, entretanto, a característica normalmente indesejável de apresentar um erro residual em regime permanente (*offset*). A sintonia é relativamente fácil de ser obtida, pelo ajuste de um único parâmetro ( $K_p$ ).

### 2.2.2 Controle Proporcional-Integral (PI)

A principal característica do controle integral é a redução ou eliminação do erro em regime permanente. Possui a desvantagem de tornar a resposta muito oscilatória quando o ganho integral é muito alto ou a resolução do sensor é baixa, este inconveniente é contornado com a adição do ganho proporcional e se necessário a atenuação do ganho integral.

### 2.2.3 Controle Proporcional-Derivativo (PD)

Um controlador derivativo associado à ação proporcional, corresponde ao acréscimo de um zero ao sistema, atuando benéficamente no regime transitório, tendendo a aumentar a estabilidade relativa do sistema e reduzindo o tempo de acomodação, contudo, contrapondo-se a estas vantagens, ele aumenta o tempo de subida e, por não atuar no regime permanente, não corrige o erro de estado estacionário (ARAUJO, 2007).

### 2.2.4 Controle Proporcional-Integral-Derivativo (PID)

O Controle Proporcional-Integrativo-Derivativo combina as características de estabilidade conferida pelo controle proporcional mais derivativo com as características de eliminação do erro oferecidas pelo controle proporcional mais integral. Como a ação derivativa torna o controlador mais difícil de sintonizar, o controle de três modos (PID) deverá ser utilizado somente em determinados processos que realmente tenham seu desempenho bastante aumentado (BENTO, 1989).

O controle PID normalmente é expresso no domínio do tempo, como mostra a equação (1):

$$u(t) = K_p e(t) + K_i \int_0^t e(t) dt + K_d \frac{de(t)}{dt} \quad (1)$$

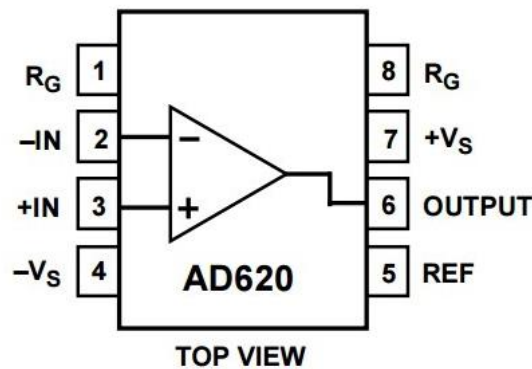


Fonte: ANALOG DEVICES (2018).

A

Figura 4 apresenta um esquemático da pinagem do amplificador AD620.

Figura 4 - Esquemático operacional do AD620



Fonte: ANALOG DEVICES (2018).

Os transistores de entrada (Q1 e Q2) formam um par diferencial bipolar de entrada para uma alta precisão e ainda assim, oferece menos corrente de polarização.

A realimentação entre os laços Q1-A1-R1 e Q2-A2-R2 fazem com que a corrente de coletor de Q1 e Q2 sejam constantes e consequentemente se tem uma diferença de potencial no resistor de ganho (RG). Isso cria um ganho diferencial entre as entradas e as saídas de A1/A2. O subtrator unitário A3 remove qualquer sinal de modo-comum, gerando um sinal simples, referenciado ao pino REF (MARCINICHEN, 2018).

Como R1 e R2 possuem um valor de 24.7kΩ, apresenta-se a Equação 2 para definir o RG em função do ganho desejado: (ANALOG DEVICES, 2018).

$$R_G = \frac{49.4k\Omega}{G-1} \quad (2)$$



### 3 DESENVOLVIMENTO DO SISTEMA

Esta seção descreve o desenvolvimento do sistema de hardware de controle de temperatura, do controlador PID, do *software* realizado.

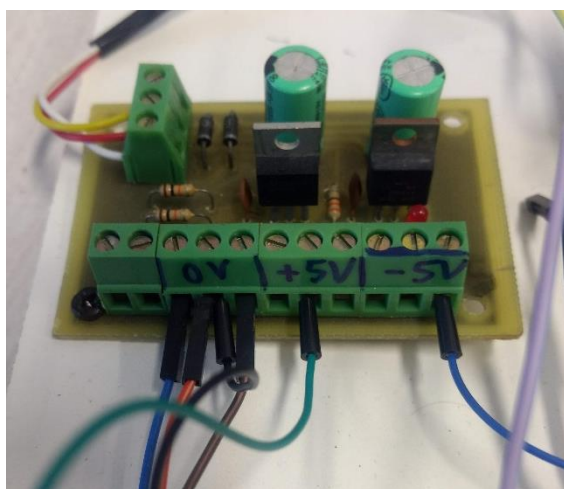
#### 3.1 Materiais e componentes utilizados

Para que o controle PID possa ser realizado, faz-se necessário o desenvolvimento de uma placa com um amplificador AD para a leitura do sensor termopar tipo K, para anexar ao protótipo inicial. Também foi necessária a utilização de uma placa reguladora de tensão 24V para 5V que havia sido desenvolvida anteriormente, para alimentação da placa do conversor AD. Para realizar a programação de todo sistema, foi utilizada a plataforma NodeMCU, que além de comandar o controle PID, faz a conexão *Wi-Fi* e possibilita a interface *WEB*.

##### 3.1.1 Placa Reguladora de Tensão 24V-5V

Foi necessária a adição de uma placa reguladora de tensão 24V-5V, conforme a Figura 5, para poder ser conectado ao conversor AD que foi utilizado.

Figura 5 - Placa reguladora de tensão 24V-5V



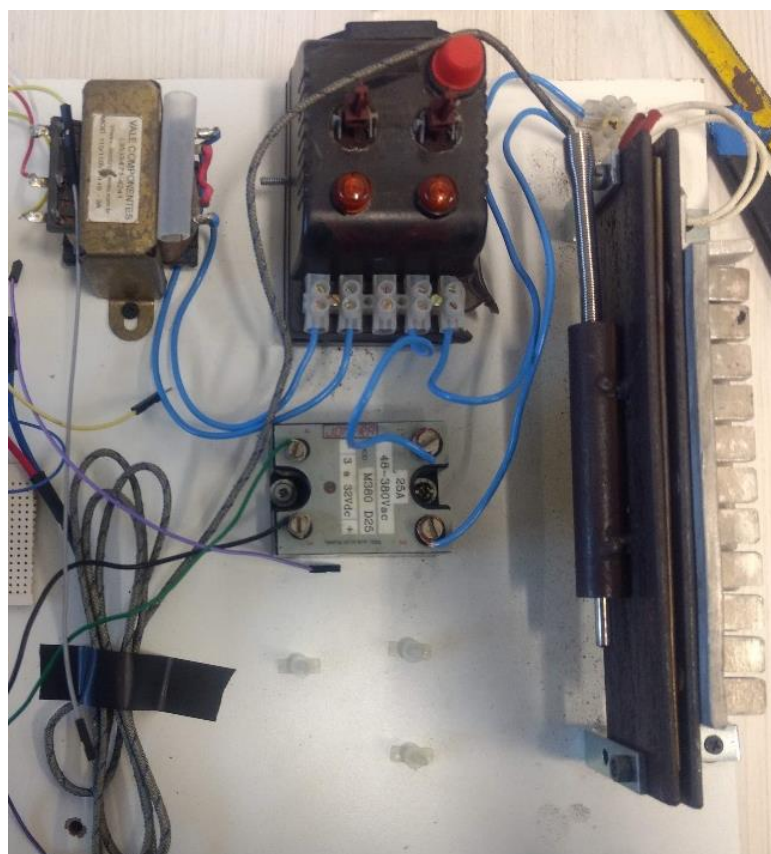
Fonte: Autoria própria (2018).

### 3.1.2 Desenvolvimento do Hardware

Foi utilizado inicialmente, como reaproveitamento de o um projeto anterior, alguns itens eletrônicos que seriam extremamente importantes para o desenvolvimento do projeto, conforme mostrado na Figura 6.

Instalados, haviam um transformador 220V/24V, uma botoeira que possui as conexões para a rede elétrica, em 220V, conexão para o transformador com um botão para acionamento e outro botão para a resistência elétrica utilizada. Também já existia instalado um relé de estado sólido, conectado a resistência elétrica de um ferro de passar roupa de  $165\Omega/220V$ , e um suporte de metal que serve como dissipador de calor, e um sensor de temperatura termopar tipo K, que faz a conversão do calor adquirido em tensão.

**Figura 6 - Protótipo inicial**



Fonte: Autoria própria (2018).

Foi construída, no início do projeto, uma segunda placa, com um amplificador AD620, para poder realizar a leitura do sensor termopar. Foi utilizado este componente pois ele possui o ganho ajustável, através de um resistor conectado entre os pinos 1 e 8, o que é uma vantagem, porque permite programar o ganho para qualquer valor desejado.

Foi calculado o ganho necessário para que a saída de tensão para o controlador fosse de até 3.3V, limitação que a plataforma de controle que será utilizada possui.

Com os resultados dos testes iniciais com o termopar, foi constatado que era necessário um ganho de aproximadamente 150 para ter uma correta amplificação da tensão, e, portanto, um resistor de  $330\Omega$ , calculados conforme as Equações 3 a 5 descrevem.

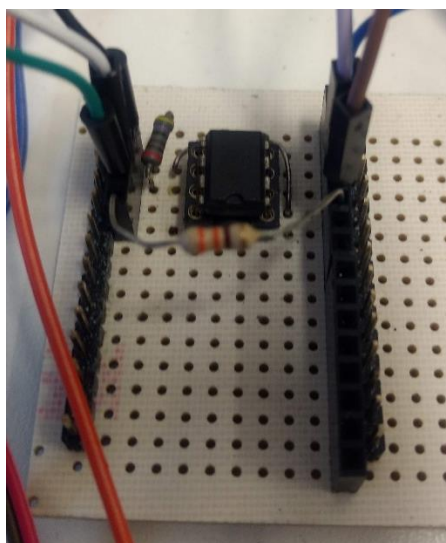
$$R_G = \frac{49.4k\Omega}{150-1} \quad (3)$$

$$R_G = \frac{49.4k\Omega}{149} \quad (4)$$

$$R_G = 331,54 \Omega \quad (5)$$

Foi necessária a adição de dois resistores com valores de  $119 \Omega$  e  $47 \Omega$  como filtros nas portas de entrada e saída do amplificador. Na Figura 7 é possível ver como a montagem em seu estado final.

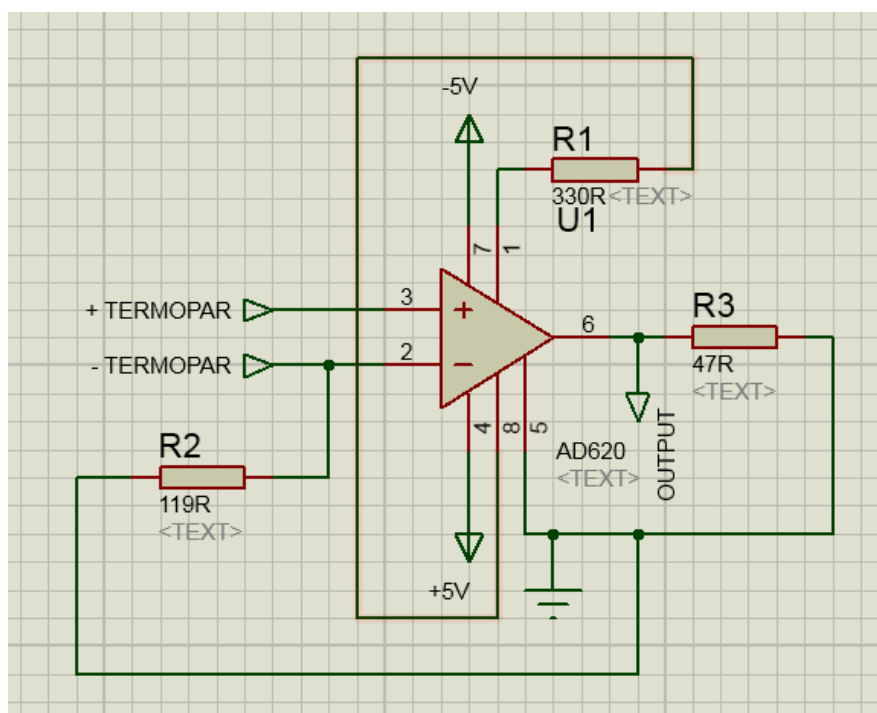
**Figura 7 - Placa com amplificador de instrumentação AD620**



Fonte: Autoria própria (2018).

O esquemático final desta montagem pode ser visto na Figura 8.

**Figura 8 - Esquemático da placa do AD620**

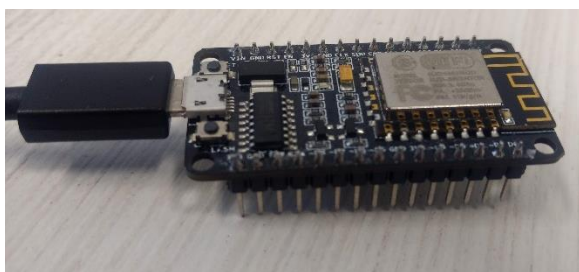


Fonte: Autoria própria (2018).

### 3.1.3 Plataforma utilizada

A plataforma escolhida, por ser de fácil acesso, foi a NODEMCU® - Figura 9, que utiliza o microcontrolador ESP8266E, da empresa Espressif®. E que tem uma forma de gravação de dados diretamente pela USB. Esta plataforma de desenvolvimento permite realizar diretamente a conexão a uma rede de dados sem fio, através da programação, sem a necessidade de adição de componentes externos.

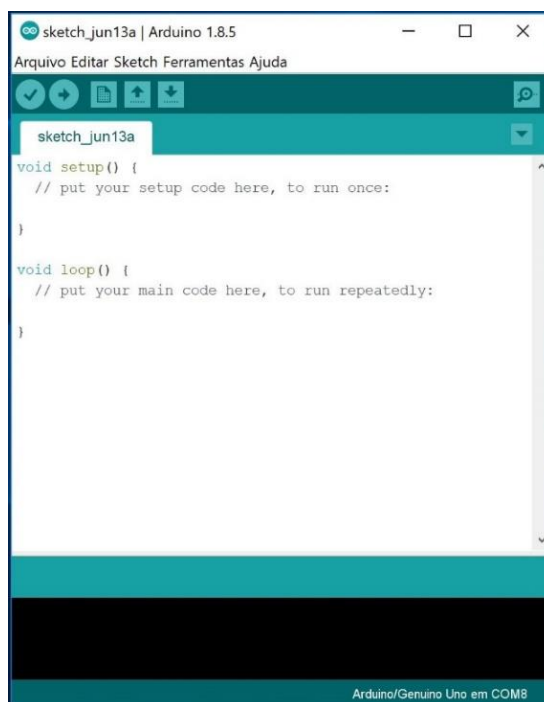
**Figura 9 - Plataforma NodeMCU**



Fonte: Autoria própria (2018).

### 3.1.4 Interface de programação do NodeMCU

Devido ao fato de a plataforma e o *software* serem abertos e conterem bibliotecas de código acessíveis, e integrações menos complicadas com a plataforma escolhida, optou-se pela utilização do Ambiente de Desenvolvimento Integrado (IDE) do *Arduino*, para realizar a programação do código completo, incluindo a configuração sem fio, controle PID e comunicação com a interface *Node-red*. A Figura 10 mostra a interface IDE *Arduino*.

**Figura 10 – Arduino IDE**

Fonte: Autoria Própria (2018).

### 3.2 CONTROLADOR PID

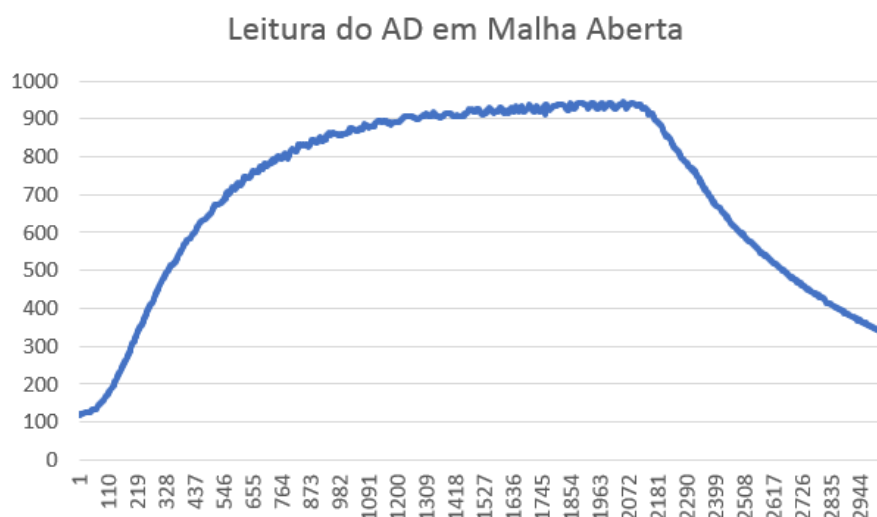
Na parte metodológica para a realização do controle, inicialmente levantou-se a curva de resposta ao degrau em malha aberta, e para isso foi necessário calcular o ganho total de conversão entre o valor lido pela ADC e o valor real da temperatura informado pelo Termopar.

Em seguida, foram levantados os valores de  $K_p$ ,  $K_i$  e  $K_d$  simulados pelo Matlab™, através de 4 métodos de identificação de processos: Ziegler–Nichols, Hägglund, Smith e Sundaresan & Krishnaswam. De posse destes resultados, é possível realizar os cálculos de controle e encontrar os coeficientes de  $K_p$ ,  $K_i$  e  $K_d$  que melhor se aproximam da resposta desejada.

### 3.2.1 Levantamento da curva em malha aberta

Inicialmente foi realizada uma programação para se ler a entrada analógica (A0), que vem do conversor AD (que está ligado ao termopar), e exibir os resultados via comunicação serial, para os registros dos dados através do *software* Terminal. A Figura 11 mostra a resposta do conversor AD mantendo-se sempre ligada à resistência até o tempo 2070 s, quando então foi desligada a resistência elétrica da fonte de energia.

**Figura 11 - Leitura do AD em malha aberta**



Fonte: Autoria Própria (2018).

Baseado no *datasheet* do termopar tipo K utilizado, foram coletadas algumas amostras dos valores de temperatura do termopar e relacionadas à tensão obtida antes do amplificador operacional. Foi possível então obter um gráfico com uma reta, que podem ser vistos na Figura 12, e com esta reta foi obtida a equação que relaciona o valor em graus centígrados, para poder realizar futuros cálculos de conversão da temperatura.

**Figura 12 - Gráfico da Relação da Temperatura do Termopar Tipo K**



Fonte: Autoria Própria (2018).

Como a equação obtida é baseada na tensão em milivolts que o termopar fornece, pode-se realizar cálculos para se poder obter uma relação entre a leitura do AD e a temperatura efetivamente medida.

É possível verificar alguns valores com a relação entre a leitura do AD e a tensão de saída e de entrada do amplificador, através de algumas equações que serão descritas abaixo, levantadas baseadas em dados obtidos.

Para se obter os valores de tensão na entrada e saída do amplificador, e o valor de temperatura convertido através da equação levantada, foram utilizadas as Equações 6 a 9, que fornecem o processo de obtenção desses valores.

$$V_{OUT} = \frac{LEITURA_{AD} * 3.3}{1023} \quad (6)$$

$$V_{IN} = \frac{V_{OUT}}{156.26} \quad (7)$$

$$Temperatura = \frac{V_{IN} + 0.0001}{0.000041} \quad (8)$$



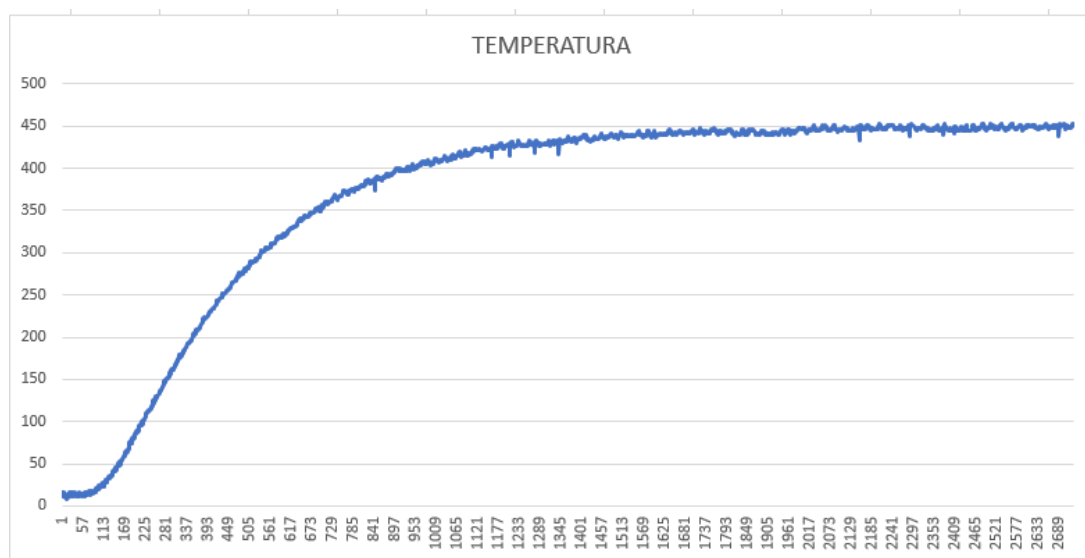
A conversão direta da leitura do AD para a temperatura em graus Celsius, foi obtida então a Equação 9:

$$Temperatura = \frac{\left( \frac{Leitura_{AD} * 3.3}{1023} \right) + 0.0001}{0.000041} \quad (9)$$

Se, por exemplo, for obtido na leitura no ADC da NodeMCU o valor 19, então nosso valor de tensão na saída do AD é 0,06129 V, então o valor de tensão na entrada do AD é 0,000392V, que representa a temperatura de 12,00568 °C.

A Figura 13 mostra a temperatura medida pelo termopar como resposta ao degrau unitário, em malha aberta, ou seja, sem controle em um tempo de 2650s.

**Figura 13 - Temperatura em Malha Aberta**



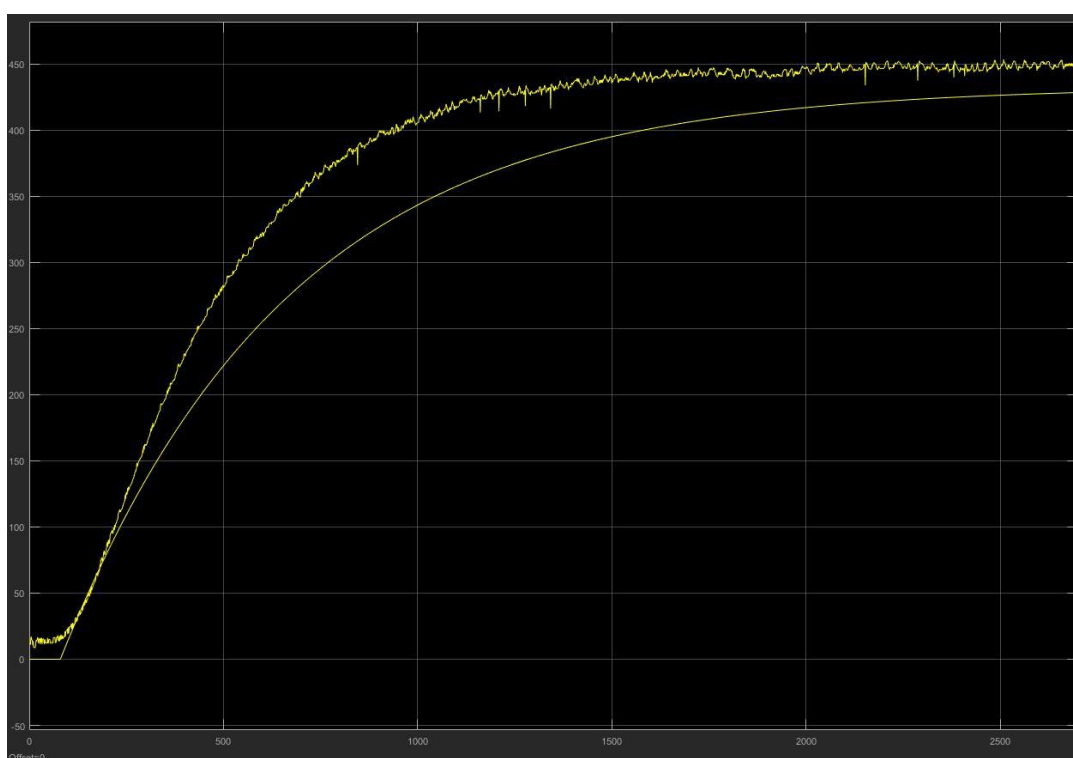
Fonte: Autoria Própria (2018).

### 3.2.2 Controlador PID

Foi obtida a curva de temperatura pelo tempo, obtendo-se cerca de 2700 pontos medidos, ao longo do tempo, e com eles, por meio do *software* MATLAB, foi obtida a curva de resposta do sistema em malha aberta. Esses dados foram utilizados para gerar a função de transferência que posteriormente será modelada pelo método que melhor se adapta ao projeto.

Os testes consistiam na alimentação máxima da tensão em 220V. Baseado nesta alimentação se adquiria a curva de resposta do sistema. Após o levantamento da curva, foram realizados os cálculos da função de transferência do sistema, e obtiveram-se os dados necessários para inserir no MATLAB®, comparando então as duas curvas, mostradas na Figura 14, sendo a superior obtida através da leitura do sensor, e a inferior calculada através da função de transferência obtida do MATLAB®.

**Figura 14 – Gráfico em malha aberta, obtido através do levantamento de pontos do sistema**



Fonte: Autoria Própria (2018).

Com essa curva pode-se obter, por meio de quatro métodos, a curva que melhor se adequa ao projeto, os quais são: método de Ziegler – Nichols, método de Hägglund, método de Smith e método de Sundaresan&Krishnaswamy.

Cada método tem como saída uma função. Outras informações que são importantes nos cálculos dos métodos são: variação da temperatura do forno ( $\Delta Y$ ), variação da alimentação do motor ( $\Delta U$ ), atraso ( $\theta$ ) e o tempo de subida ( $\tau$ ) e ganho estático ( $K$ ) (COELHO, 1999).

Para todos os métodos utiliza se a seguinte função de controle:

$$G(s) = \frac{K \times e^{-\theta}}{\tau s + 1} \quad (10)$$

K é calculado como sendo:

$$K = \frac{\Delta Y}{\Delta U} \quad (11)$$

$$K = \frac{446-12}{220-0} \quad (12)$$

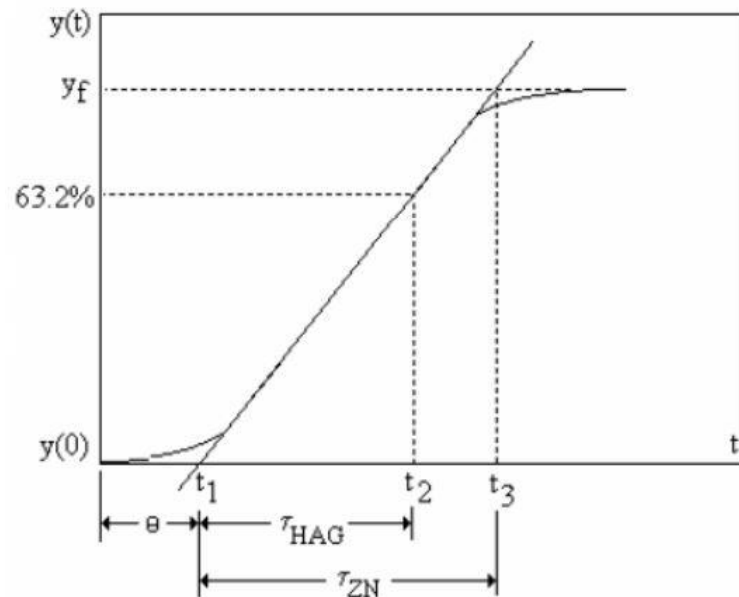
$$K = \frac{434}{220} \quad (13)$$

$$K = 1,97 \quad (14)$$

A partir dos dados mostrados na Figura 14, e o valor de K calculado, é possível, então, calcular a função de transferência em malha aberta do nosso sistema, baseado nos métodos descritos nas Figura 15, Figura 16 e Figura 17.

### 1º Método - Ziegler – Nichols:

Figura 15 - Gráfico Método Ziegler - Nichols



Fonte: COELHO & COELHO (2004).

Sendo:

$$t_1 = 80s; t_2 = 514s; t_3 = 667s. \quad (15)$$

$$\theta = t_1 \quad (16)$$

$$\theta = 80 \quad (17)$$

$$\tau = t_3 - t_1 \quad (18)$$

$$\tau = 667 - 80 \quad (19)$$

$$\tau = 587 \quad (20)$$

Obtendo a seguinte função:

$$G_{ZN}(s) = \frac{1,97 \times e^{-80}}{587s+1} \quad (21)$$

$$G_{ZN}(s) = \frac{0.0034 \times e^{-80}}{s+0.0017} \quad (22)$$

## 2º Método - Hägglund:

Utilizando a figura 15 também, pode-se realizar os cálculos de  $t_2$  e  $t_1$  pelo método de Hägglund:

$$\theta = t_1 \quad (23)$$

$$\theta = 80 \quad (24)$$

$$\tau = t_2 - t_1 \quad (25)$$

$$\tau = 514 - 80 \quad (26)$$

$$\tau = 434 \quad (27)$$

Obtendo a seguinte função:

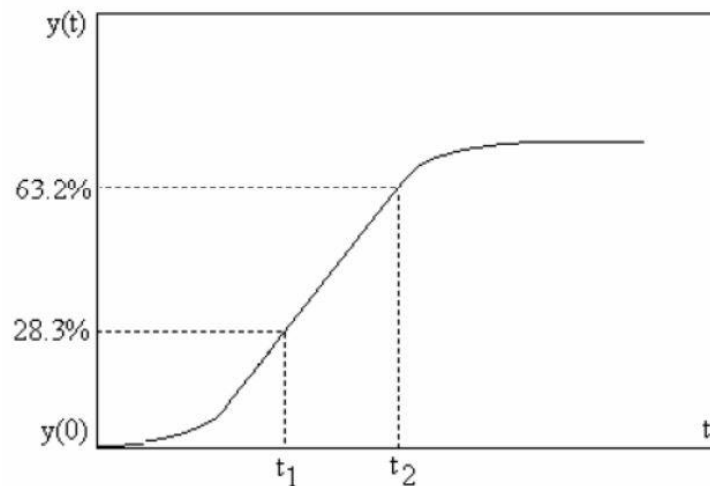
$$G_{HAG}(s) = \frac{1,97 \times e^{-80}}{434s+1} \quad (28)$$

$$G_{HAG}(s) = \frac{0.0045 \times e^{-80}}{s+0.0023} \quad (29)$$

### 3º Método - Smith:

Da mesma forma como foi feito com os outros métodos, pode-se realizar o cálculo pelo método de Smith obtendo a seguinte sequência de cálculo:

**Figura 16 - Gráfico Método Smith**



Fonte: COELHO & COELHO (2004).

Sendo:

$$t_1 = 244s; t_2 = 514s. \quad (30)$$

$$\tau = 1,5 \times (t_2 - t_1) \quad (31)$$

$$\tau = 1,5 \times (514 - 244) \quad (32)$$

$$\tau = 1,5 \times (270) \quad (33)$$

$$\tau = 405 \quad (34)$$

$$\theta = t_2 - \tau \quad (35)$$

$$\theta = 514 - 405 \quad (36)$$

$$\theta = 109 \quad (37)$$

Obtendo a seguinte função:

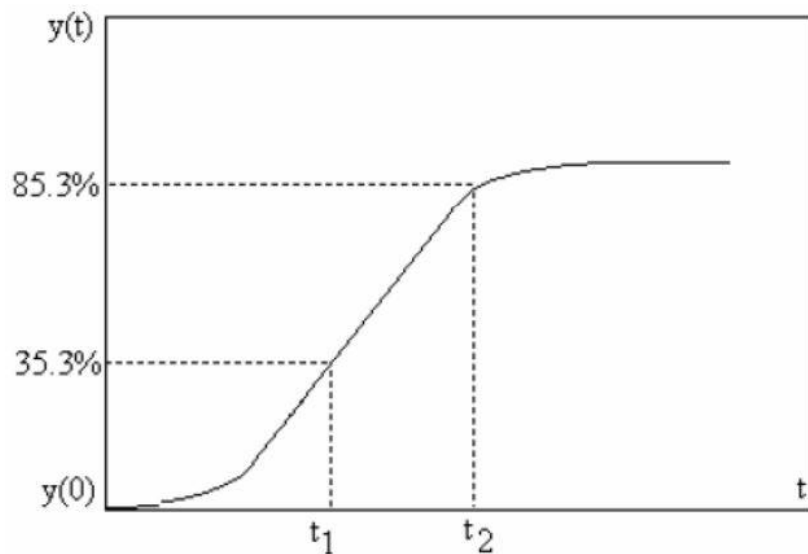
$$G_{SMT}(s) = \frac{1.97 \times e^{-109}}{405s+1} \quad (38)$$

$$G_{SMT}(s) = \frac{0.0049 \times e^{-109}}{s+0.0025} \quad (39)$$

#### 4º Método: Sundaresan & Krishnaswamy

O cálculo pelo método de Sundaresan & Krishnaswamy segue a seguinte sequência:

**Figura 17 - Gráfico Método Sundaresan & Krishnaswamy**



Fonte: COELHO & COELHO (2004).

Sendo:

$$t_1 = 289s; \quad t_2 = 811s \quad (40)$$

$$\tau = 0,67 \times (t_2 - t_1) \quad (41)$$

$$\tau = 0,67 \times (811 - 289) \quad (42)$$

$$\tau = 0,67 \times (522) \quad (43)$$

$$\tau = 350 \quad (44)$$

$$\theta = 1,3 \times t_1 - 0,29 \times t_2 \quad (45)$$

$$\theta = 376 - 235 \quad (46)$$

$$\theta = 141 \quad (47)$$

Obtendo a seguinte função:

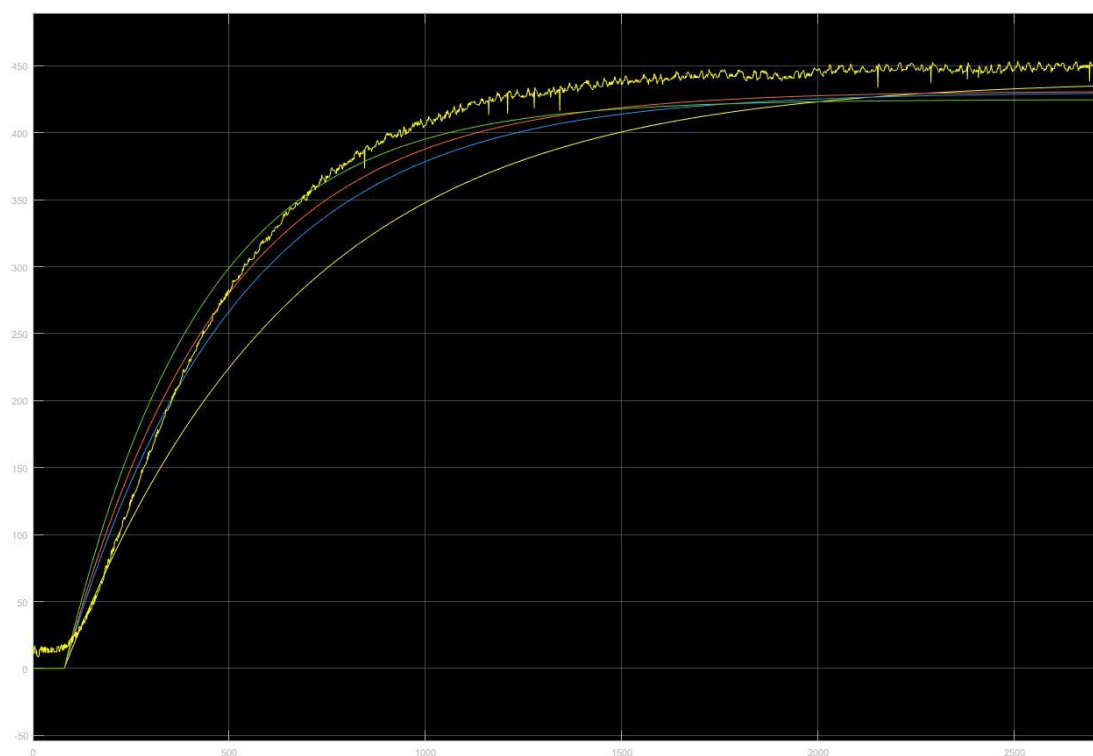
$$G_{SK}(s) = \frac{1,97 \times e^{-141}}{350s+1} \quad (48)$$

$$G_{SK}(s) = \frac{0,0056 \times e^{-141}}{s+0,0029} \quad (49)$$

Inserindo estas quatro funções dos diferentes métodos, no MATLAB e comparando-se com a curva obtida, imprimindo-as mesmas em um mesmo gráfico, pôde-se escolher a curva que mais se aproxima do nosso sistema, como mostra a Figura 18.



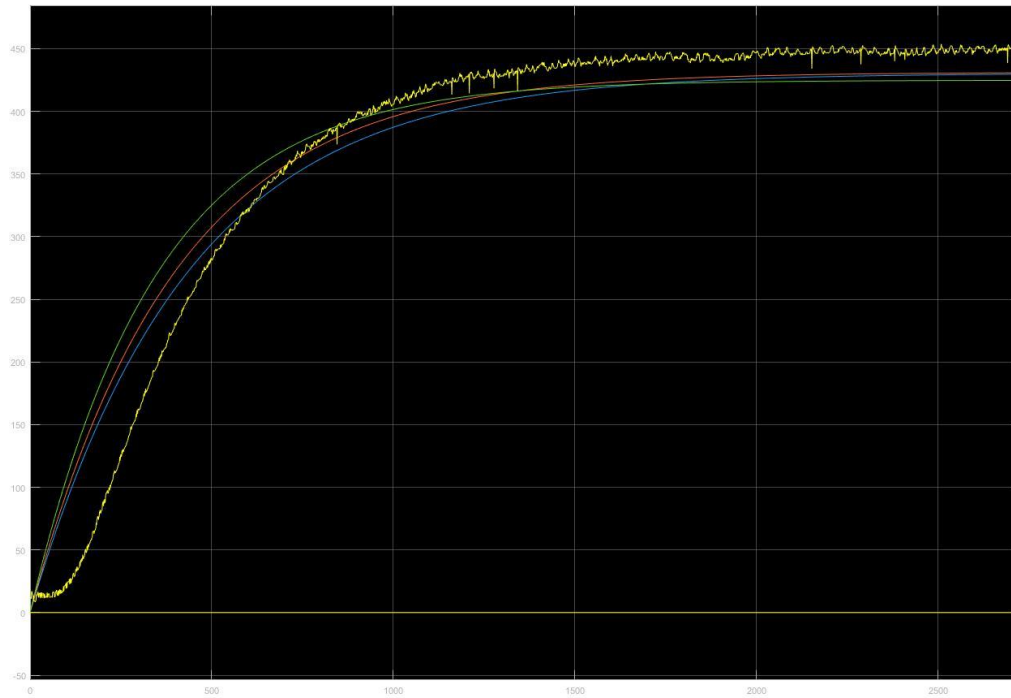
**Figura 18 - Comparação dos Métodos com Curva Real**



Fonte: Autoria Própria (2018).

Excluindo o método de Ziegler – Nichols que ficou muito lento em relação ao objetivo, restaram três curvas, que podem ser vistas na Figura 19, as quais apresentaram melhores resultados.

**Figura 19 - Controladores com melhores resultados com a curva em malha aberta**



Fonte: Autoria Própria (2018).

Conforme requisito de controle – ter um sobressinal de até 20% e menor erro possível de regime permanente possível - concluiu-se que deve ser utilizado o controlador PI, porque ao calcular o  $K_d$  do PID, verificou-se se o seu valor deve ser igual zero.

Fechando então a malha relacionada ao controle PI, obtém-se as seguintes funções gerais de cada método.

$$U(s) = \frac{K_p s + K_i}{s} * G(s) \quad (50)$$

$$G_{HAG}(s) = \frac{0.0045K_p s + 0.0045K_i}{s^2 + (0.0023 + 0.0045K_p)s + 0.0045K_i} \quad (51)$$

$$G_{SMT}(s) = \frac{0.0049K_p s + 0.0049K_i}{s^2 + (0.0025 + 0.0049K_p)s + 0.0049K_i} \quad (52)$$

$$G_{SK}(s) = \frac{0.0056Kps + 0.0056Ki}{s^2 + (0.0029 + 0.0056Kp)s + 0.0056Ki} \quad (53)$$

Definida a função geral, devem ser determinados os valores ideais de Kp e Ki. Para isso, calcula-se o tempo de acomodação ( $T_a$ ). Baseando-se nele, é encontrado um dos polos (p) da função. Considerando que os polos são iguais e reais, a função dominante é igualada com a função gerada por  $(s+p)^2$ .

Para um tempo de acomodação de 500 segundos, cerca de um minuto mais rápido do que em malha aberta, conforme requisitado um ganho de 10%, fez-se o seguinte cálculo para obter o denominador da função transferência (equação 60).

$$T_a = \frac{4}{|p|} \quad (54)$$

$$500 = \frac{4}{|p|} \quad (55)$$

$$|p| = \frac{4}{500} \quad (56)$$

$$|p| = 0,008 \quad (57)$$

$$(s + p)^2 \quad (58)$$

$$(s + 0,008)^2 \quad (59)$$

$$s^2 + 0,016s + 0,000064 \quad (60)$$

Com esta equação, é possível realizar a comparação dela com os denominadores das equações encontradas de cada método, porém foi escolhido o Método Häggglund por obter os melhores resultados, e com isso obter os valores de Kp e Ki calculados.

**a) Hägglund**

$$s^2 + 0.016s + 0.00064 = s^2 + (0.0023 + 0.0045Kp)s + 0.0045Ki \quad (61)$$

$$0.0045Ki = 0.00064 \quad (62)$$

$$Ki = 0.01422 \quad (63)$$

$$0.0023 + 0.0045Kp = 0.016 \quad (64)$$

$$Kp = 3.04 \quad (65)$$

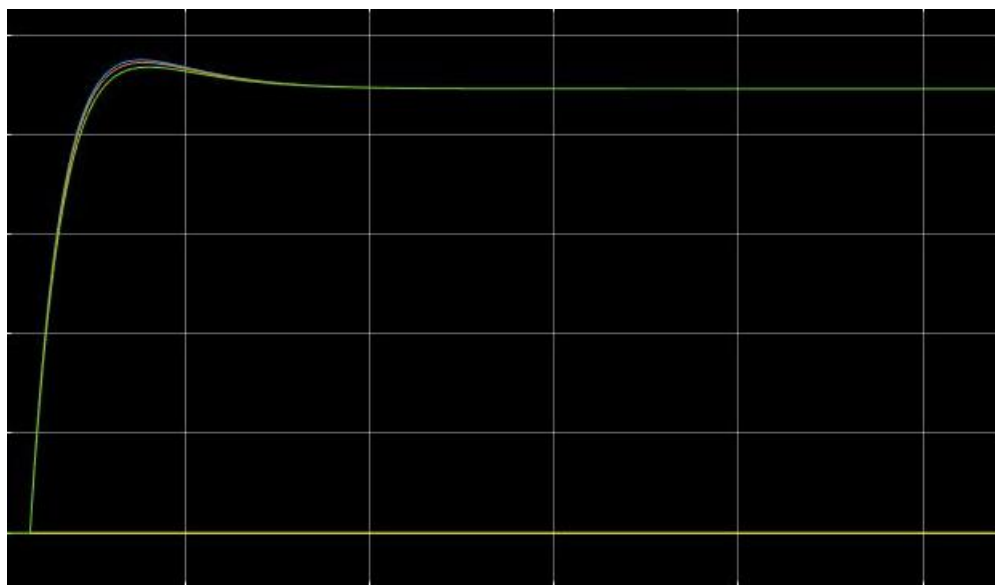
Substituindo-se os valores nas funções calculadas, é possível obter as funções de transferência final de cada método aplicado:

$$G_{HAG}(s) = \frac{(0.0045*3.04)s + (0.0045*0.01422)}{s^2 + (0.0023 + (0.0045*3.04))s + (0.0045*0.01422)} \quad (66)$$

$$G_{HAG}(s) = \frac{0.01368s + 0.000064}{s^2 + 0.01598s + 0.000064} \quad (67)$$

A função gerada foi analisada no *software* MATLAB, para verificar se apresentava uma curva dentro dos padrões esperados – Figura 20.

**Figura 20 - Curva em malha fechada com coeficientes calculados**



Fonte: Autoria Própria (2018).

Por meio da aplicação “tune” nas configurações do bloco do PID do *Simulink*, podem ser encontrados os seguintes valores para  $K_p$  e  $K_i$ :

$$K_p = 0.8 \quad (68)$$

$$K_i = 0.003 \quad (69)$$

Com os ajustes realizados, de acordo com o menor erro e o melhor tempo de acomodação, obtiveram-se os seguintes valores finais para  $K_p$  e  $K_i$  e que serão usados na parte experimental do projeto.

$$K_p = 2 \quad (70)$$

$$K_i = 0.005 \quad (71)$$

### 3.3 SOFTWARE

Para finalizar este capítulo, descreve-se o desenvolvimento do *software* embarcado na plataforma de controle, e as configurações da interface *WEB*.

#### 3.3.1 Software embarcado

A programação do microcontrolador consiste em realizar uma conexão *wi-fi*; após isso foram configuradas duas interrupções – uma para realizar o cálculo do controle PID e a outra para calcular a razão cíclica do PWM da saída do controlador, e algumas informações que estarão sendo trocadas via *wi-fi* com a plataforma que permita disponibilizar dados numa interface *WEB*. Utilizou-se, para essa função, a plataforma de desenvolvimento Node-RED, da IBM.

Um usuário que esteja acessando uma página *WEB* pré-definida, pode a qualquer momento enviar, por meio de um *software* navegador na internet, tipo Firefox, os comandos de acionar e desacionar a função de controle; permite-se, através de uma barra de rolagem informar um valor de referência para o controle ser realizado, e os valores de temperatura são mostrados já convertidos pra a escala de temperatura.

Segue, na Figura 21, uma descrição do código de programação em linguagem C, do *software* embarcado na NodeMCU:

Figura 21 - Código de programação - 1

```
#include <PID_v1.h>
#include <user_interface.h>

#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <ESP8266mDNS.h>

#define CONTROLE_LIGADO 1
#define CONTROLE_DESLIGADO 0
```

Fonte: Autoria própria (2018).

Inicialmente, incluiu-se no programa as bibliotecas PID\_v1.h, e a biblioteca user\_interface.h, que são as bibliotecas para controle PID.

Foram adicionadas também as bibliotecas ESP8266WiFi.h, WiFiClient.h, ESP8266WebServer.h, Esp8266mDNS.h que são bibliotecas de programação que permitem a configuração da placa *wi-fi* ESP8266 que está acoplada ao NodeMCU.

Definiu-se, ainda, que as palavras CONTROLE\_LIGADO corresponde a 1, e CONTROLE\_DESLIGADO corresponde a 0.

A Figura 22, mostra que foram criadas as variáveis utilizadas ao longo da programação e foram definidos seus valores iniciais.

Foi definido também o servidor da ESP para funcionar por meio da porta número 80 (porta HTTP).

Figura 22 - Código de programação - 2

```

unsigned char flagControle = CONTROLE_DESLIGADO;
const char *ssid = "Sol E Lua";
const char *password = "Kz0v3571";
ESP8266WebServer server ( 80 );

const int PERIODO_MEDICAO_MS = 500;

const int analogInPin = A0;
const int DigitalOutPin = D1;

double valor_medido_temperatura = 0;
double valor_medido = 0;
double kp = 2.0;
double ki = 0.005;
double kd = 0;
double input = 0, output = 0, setpoint = 200;
float time_high = 0;

```

Fonte: Autoria própria (2018).

É configurado o controlador PID através da função PID com as variáveis e valores que foram previamente definidos, onde em cada uma delas é adicionada o parâmetro de controle obtido – “&input” fica a porta de entrada, “&output” a porta de saída, “&setpoint” o valor de temperatura escolhido para estabilizar o controle, as variáveis de “kp”, “ki” e “kd” que foram calculadas e a direção que a saída irá se movimentar, sendo usado o DIRECT a mais comum . Foram criadas duas interrupções no sistema, e com elas calcula-se o valor do controle PID e o valor do PWM de saída, como mostra a Figura 23.

Figura 23 - Código de programação - 3

```

PID myPID(&input, &output, &setpoint, kp, ki, kd, DIRECT);
os_timer_t tmr0;
os_timer_t tmr1;

String referencia_de_controle = "";

```

Fonte: Autoria própria (2018).



Na função “setup”, definiu-se a frequência da comunicação serial, o tipo de conexão WIFI que foi usada e envia-se mensagens informando o IP em que a NodeMCU se conectou e se a conexão foi bem-sucedida, conforme mostra a Figura 24.

Figura 24 - Código de programação - 4

```
void setup() {

    Serial.begin(115200);

    WiFi.mode ( WIFI_STA );
    WiFi.begin ( ssid, password );
    Serial.println ( "" );
    // Wait for connection
    while ( WiFi.status() != WL_CONNECTED )
    {
        delay ( 500 );
        Serial.print ( "." );
    }
    Serial.println ( "" );
    Serial.print ( "Connected to " );
    Serial.println ( ssid );
    Serial.print ( "IP address: " );
    Serial.println ( WiFi.localIP() );
    delay ( 500 );
}
```

Fonte: Autoria própria (2018).

As duas funções de “server.on” servem para receber as informações da página *WEB* e executar suas respectivas funções, como chamar as interrupções respectivas, como quando o controle é ligado e desligado, função que adicionamos para uma melhor otimização do sistema. Para questões de conferência, também se liga e desliga o *LED* da placa quando o controle é ligado e desligado – Figura 25.

Figura 25 - Código de programação - 5

```

if ( MDNS.begin ( "esp8266" ) ) {
    Serial.println ( "MDNS responder started" );
}
server.on ( "/", handleRoot );
server.on ( "/inline", []() {
    server.send ( 200, "text/plain", "this works as well" );
} );
server.onNotFound ( handleNotFound );

server.on("/ligaControle", [](){
    server.send(200, "text/plain", "controle-ligado");
    digitalWrite(LED_BUILTIN, HIGH);
    flagControle = CONTROLE_LIGADO;
});

server.on("/desligaControle", [](){
    server.send(200, "text/plain", "controle-desligado");
    digitalWrite(LED_BUILTIN, LOW);
    flagControle = CONTROLE_DESLIGADO;
});

```

Fonte: Autoria própria (2018).

Na Figura 26 são mostradas as funções que serão executadas pelo servidor. Estas funções são: enviar para o servidor Node-Red o valor obtido da temperatura, o valor atual de controle, e o valor de referência para o controlador.

É configurado o pino D1 como a variável de saída digital, onde será gerado o *PWM*, e inicialmente este valor começa desligado. Em seguida define-se as configurações do PID, tais como, tipo de PID, período de amostragem e os valores limites de saída do controle (no caso de 0 a 1023).

São configurados 2 temporizadores (*timers*) para serem os responsáveis por definir os tempos alto e baixo do *PWM* gerado. A base de tempo é de 500ms, dado pelo *tmr0*, e dentro desse tempo que se calcula, com o *trm1*, a largura de pulso do *PWM*.

Figura 26 - Código de programação - 6

```

server.on("/data", handleData);
server.on("/controle", handleDataControle);
server.on("/referencia", HTTP_POST, handleReferencia);
server.begin();
Serial.println ( "HTTP server started" );

pinMode(DigitalOutPin, OUTPUT);
digitalWrite(DigitalOutPin, LOW);

myPID.SetMode(AUTOMATIC);
myPID.SetSampleTime(PERODO_MEDICAO_MS);
myPID.SetOutputLimits(0,1023);

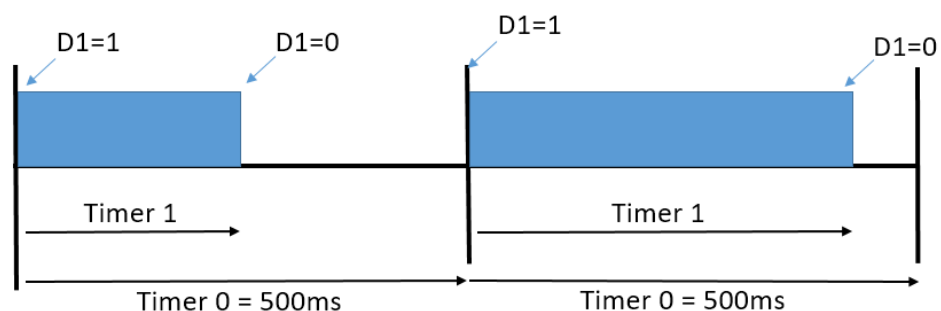
os_timer_setfn(&tmr0,PidControle, NULL);
os_timer_setfn(&tmr1,Width, NULL);
os_timer_arm(&tmr0,PERODO_MEDICAO_MS, true);
}

```

Fonte: Autoria própria (2018).

A Figura 27 ilustra o funcionamento do PWM usando 2 timers (timer 0 e timer 1):

Figura 27 - Timers



Fonte: Autoria própria (2018).

A função *handleData*, envia o valor de temperatura para a página *HTML* exibir. A função *handleDataControle*, envia o valor de controle que está sendo gerado para ser exibido também na página *HTML*. A função *handleReferencia* recebe o valor definido, de referência do controle, na página *HTML*, para o valor de temperatura, a ser usado pelo PID (no caso de 0 a 1023), conforme mostra a Figura 28.

**Figura 28 - Código de programação - 7**

```
void handleData() {
    String message = String(valor_medido_temperatura);
    server.send(200, "text/plain", message);
}

void handleDataControle() {
    String message = String(output);
    server.send(200, "text/plain", message);
}

void handleReferencia() {

    referencia_de_controle = server.arg("valor");
    double setpoint_temp = (double)referencia_de_controle.toFloat();
    setpoint = (((setpoint_temp*0.000041)-0.0001)*156.26)*1023)/3.3);

}
```

Fonte: Autoria própria (2018).

Dentro da função *loop*, executa-se a chamada do servidor e da função *handleClient* onde a mesma mantém o servidor funcionando no modo *Call-back*.

A função *PidControle* executa a leitura do pino analógico (A0) cujo valor é salvo como um parâmetro de entrada para o controle PID (variável *input*).

É calculado, em uma variável, o valor convertido para o sistema de temperatura em graus Celsius. Existem algumas funções para não estourar o valor de leitura máximo, e para chamar a função de controle – que realiza o cálculo do PID, quando o botão “inicia controle” for acionado na interface gráfica. Também são impressos na serial os valores de leitura AD e de temperatura, para possível conferência com a página, em algum momento de dúvida, mostrado pela Figura 29.

Figura 29 - Código de programação - 8

```

void loop() {
    server.handleClient();
}

void PidControle(void *z)
{
    valor_medido = analogRead(analogInPin);
    valor_medido_temperatura = (((valor_medido*3.3/1023)/156.26)+0.0001)/0.000041;
    input = valor_medido;
    if (valor_medido == 1023.0) valor_medido=0;
    if (flagControle == CONTROLE_LIGADO) myPID.Compute();

    Serial.println(valor_medido);
    Serial.println(valor_medido_temperatura);
    digitalWrite(DigitalOutPin, HIGH);

    time_high = output*PERIODO_MEDICAO_MS/1023;
    os_timer_arm(&tmr1, (int)time_high, false);
}

```

Fonte: Autoria própria (2018).

E no final do código, Figura 30, mostra-se a função que envia ao pino de saída PWM o valor zero, fazendo com que o mesmo fique desativado.

Figura 30 - Código de programação - 9

```

void Width(void *z)
{
    digitalWrite(DigitalOutPin, LOW);
}

```

Fonte: Autoria própria (2018).

### 3.3.2 Página de Gerenciamento WEB

Na intenção de maximizar e facilitar a experiência do usuário com o sistema de monitoramento, foi desenvolvida uma interface *WEB*, na qual é possível fazer o monitoramento e gerenciamento da temperatura e do controle PID realizado.

O *software* utilizado para criação da página HTML que faz a interface da placa de controle com o operador chama-se Node-RED.

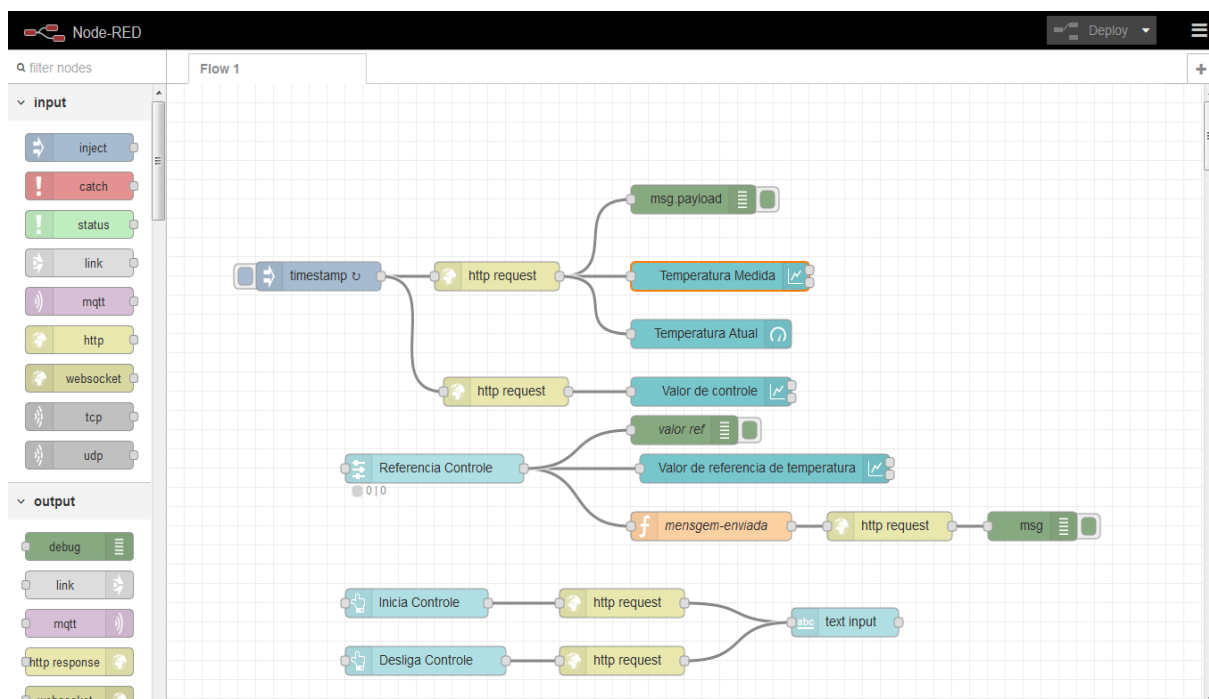
O Node-RED é uma ferramenta de desenvolvimento, originalmente criada pela IBM, hoje declarada como *software* livre, é baseada em fluxo para programação visual, para conectar dispositivos de *hardware*, pontos de acessos e serviços online como parte da Internet das coisas. (HEAT, 2018). Node-RED fornece um editor de fluxo baseado em um navegador, que pode ser utilizado para criar funções JavaScript. Elementos de aplicativos podem ser salvos ou compartilhados para reutilização. O tempo de execução é construído em Node.js e os fluxos criados no Node-RED são armazenados usando JSON.

O Node-Red também oferece a possibilidade de instalar bibliotecas adicionais que permitem uma melhor usabilidade das ferramentas de criação, fazendo que a programação da página se torne algo customizado e prático. Nesse sentido, foi instalada a biblioteca que permite criar uma página com elementos gráficos (chamada de dashboard).

Na Figura 31, tem-se a página de configuração, onde são colocados os blocos, que no *DASHBOARD* se tornam as opções gráficas.

Com esses blocos, facilitando o processo, é possível programar o dashboard para solicitar informações à placa de controle NodeMCU, e também receber informações, conectando os blocos respectivos que fazem essas funções, ao endereço de IP que a placa de controle recebe quando faz a conexão com a rede Wi-Fi local.

**Figura 31 – Ambiente de desenvolvimento Node-RED**



Fonte: Autoria própria (2018).

O bloco “*http request*” é que faz a conexão entre o servidor node-red e um servidor que está rodando na placa de controle. Estas requisições disparam serviços no servidor *web* que está rodando na NodeMCU.

O bloco “*timestamp*” tem a função de gerar o tempo de 5s, e em cada 5s há uma requisição a placa de controle. Retorna as temperaturas medidas, o valor atual do controlador PI.

Também foram criados blocos para enviar informações à placa de controle, e são enviados o valor de referência da temperatura para controle, e se deve ou não desligar o controlador (deixa o sistema no último valor de controle).

A parametrização destes blocos é facilitada, e na Figura 32 pode-se ver um exemplo de requisição, por meio do método GET, na URL 192.168.0.101, requerendo que o servidor no lado de controle interprete o comando “/data” e recebendo como resposta um valor do tipo cadeia de caracteres (string).

Figura 32 - Nó de requisição HTTP

**Edit http request node**

Delete Cancel Done

▼ node properties

Method GET

URL 192.168.0.101/data

☐ Enable secure (SSL/TLS) connection

☐ Use basic authentication

Return a UTF-8 string

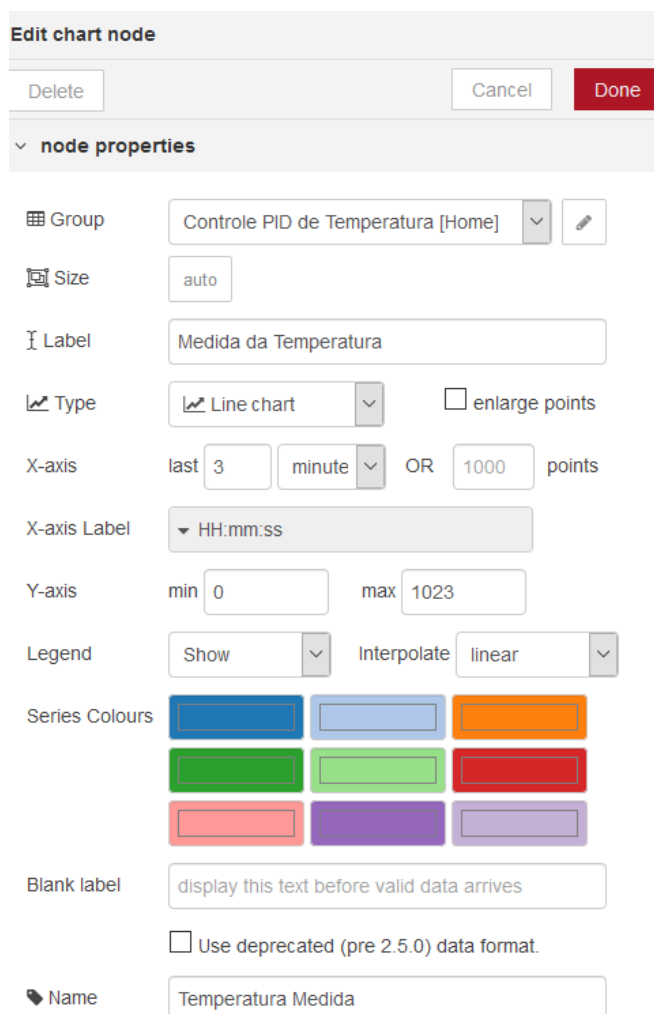
Name Name

Fonte: Autoria própria (2018).

A parte gráfica é também configurada nesse ambiente, e que depois gerará um *dashboard*. Ali se configura, por exemplo, o tipo de gráfico (linha, barras, polar, etc...), as cores, os nomes dos eixos, o título do gráfico, etc.... Um exemplo pode ser visto na Figura 33.



**Figura 33 - Exemplo de parametrização de interface gráfica**



**Edit chart node**

Delete Cancel Done

▼ node properties

Group Controle PID de Temperatura [Home] ▼

Size auto

Label Medida da Temperatura

Type Line chart ▼ ☐ enlarge points

X-axis last 3 minute ▼ OR 1000 points

X-axis Label ▼ HH:mm:ss

Y-axis min 0 max 1023

Legend Show ▼ Interpolate linear ▼

Series Colours

Blank label display this text before valid data arrives

☐ Use deprecated (pre 2.5.0) data format.

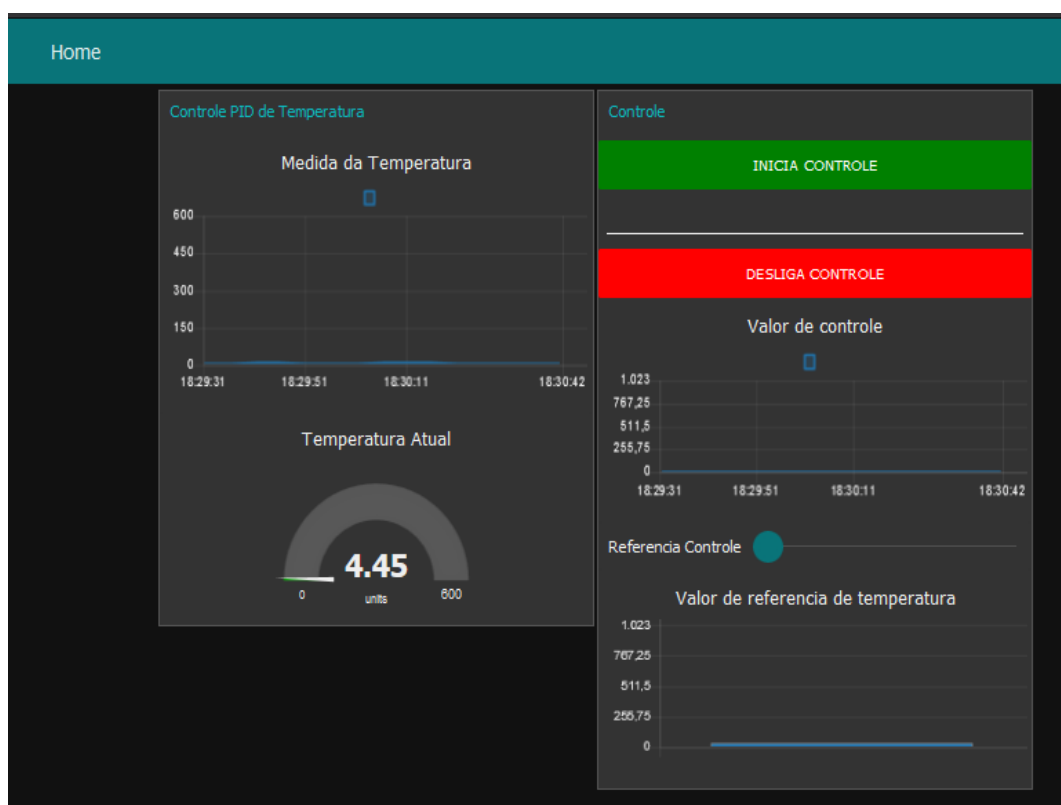
Name Temperatura Medida

Fonte: Autoria própria (2018).

Com todas as informações definidas na página de configuração, foi necessário clicar no botão “*DEPLOY*” no canto superior da página, para poder serem salvas e enviadas as modificações para o *Dashboard*, que pode ser vista no mesmo endereço *WEB*, porém acrescido da terminação “/ui/#/0”

O *Dashboard* é, portanto, a página HTML final, Figura 34, a qual permite a interação com o usuário. Através dela, é realizado o gerenciamento e monitoramento do projeto.

**Figura 34 - Página HTML desenvolvida (*dashboard*)**

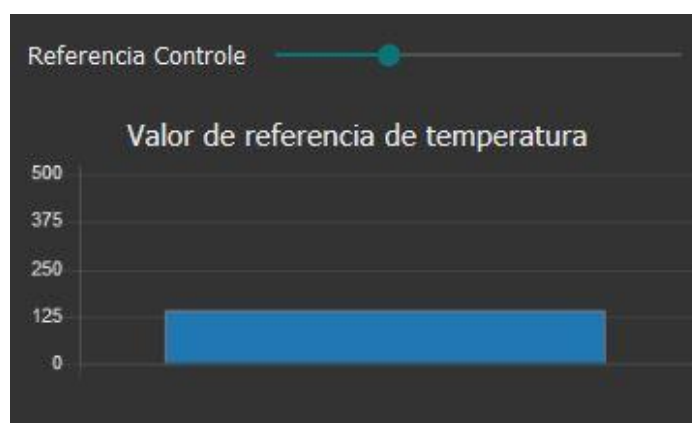


Fonte: Autoria própria (2018).

As funções apresentadas no dashboard possuem interação com o usuário, as quais podemos ver nas Figura 35 e Figura 36, são:

- a) Definir o valor de referência para o Controle PID: Através de uma barra de rolagem, é possível setar o valor de referência para o controle PID da temperatura.

**Figura 35 - Valor de referência de temperatura no dashboard**



Fonte: Autoria própria (2018).

b) Iniciar o controle: Botão para iniciar o controle, que manda um comando para a programação, que chama a interrupção de realiza o cálculo do controle PID.

c) Desligar o controle: Botão para desligar o controle, que manda um comando para a programação, que chama a interrupção de realiza o envio do comando de baixo – referente a 0, para o pino de saída da placa, no qual o sistema segue realizando o cálculo de controle do processo a partir da última referência enviada.

**Figura 36 – Inicia e Desliga Controle no *dashboard***

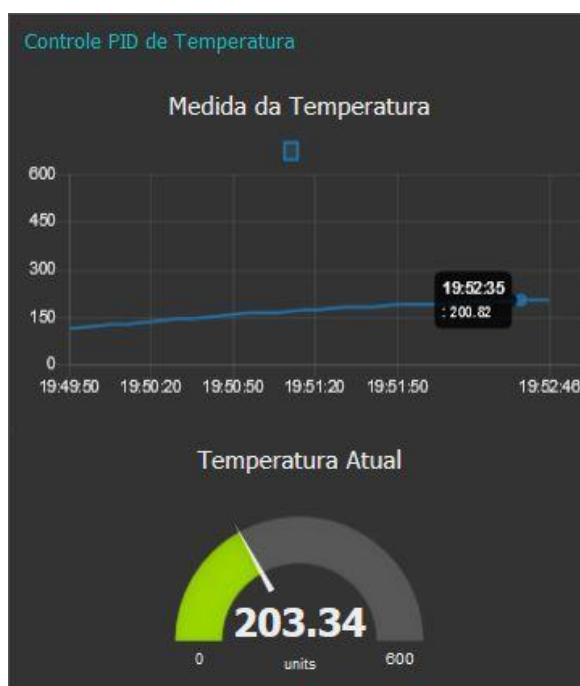


Fonte: Autoria própria (2018).

Os serviços que a página fornece ao usuário estão ilustrados nas Figura 37 e Figura 38:

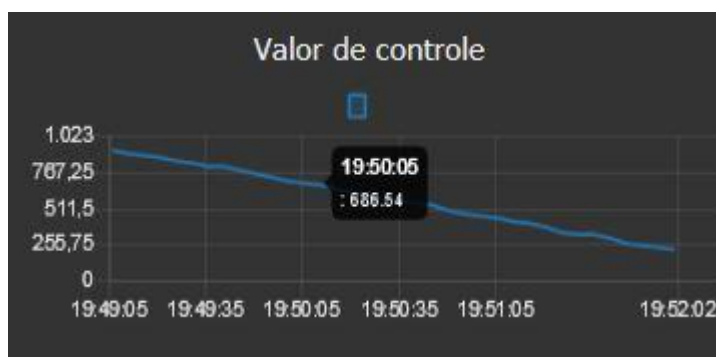
a) Dados: Gráficos da leitura da temperatura em graus Celsius e dos valores de controle são exibidos na tela, permitindo que o usuário tenha um melhor gerenciamento do processo.

**Figura 37 - Gráfico de temperatura do *dashboard***



Fonte: Autoria própria (2018).

**Figura 38 - Gráfico do valor de controle do *dashboard***



Fonte: Autoria própria (2018).

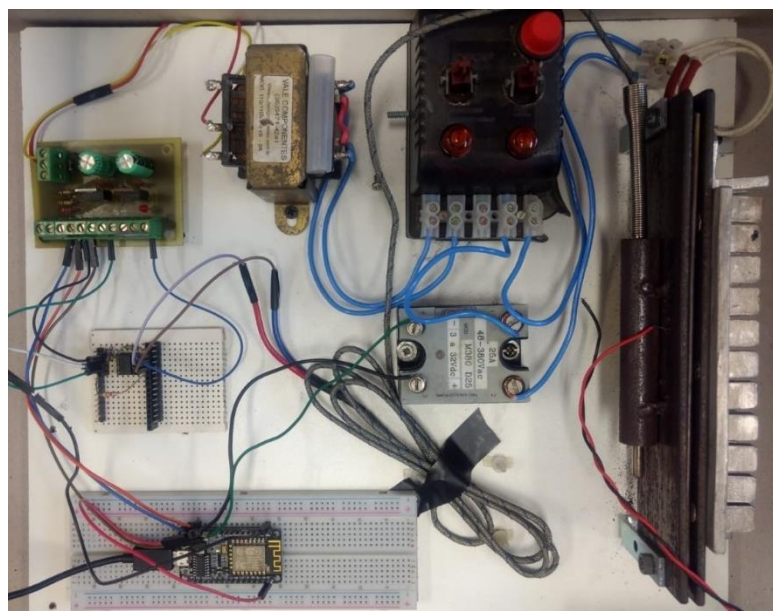
## 4 ANÁLISE DOS RESULTADOS

O objetivo do projeto foi alcançado, com uma solução considerada simples e de baixo custo, com controle PI, e gerenciamento do sistema por meio de redes sem fio.

Com o objetivo de propor uma interface amigável e de fácil acesso para o usuário, foi desenvolvida uma página em HTML que, através da plataforma utilizada de conexão remota, recebe os dados provenientes do conversor AD, o qual está recebendo valores de tensão do sensor Termopar Tipo K e os apresenta na página originada pela conexão sem fio do mesmo.

O projeto todo foi desenvolvido com recursos do laboratório NersD e apresentou resultados satisfatórios, mantendo seu funcionamento mesmo após longos períodos de teste. O protótipo final pode ser observado na Figura 39.

**Figura 39 – Protótipo finalizado**



Fonte: Autoria própria (2018).

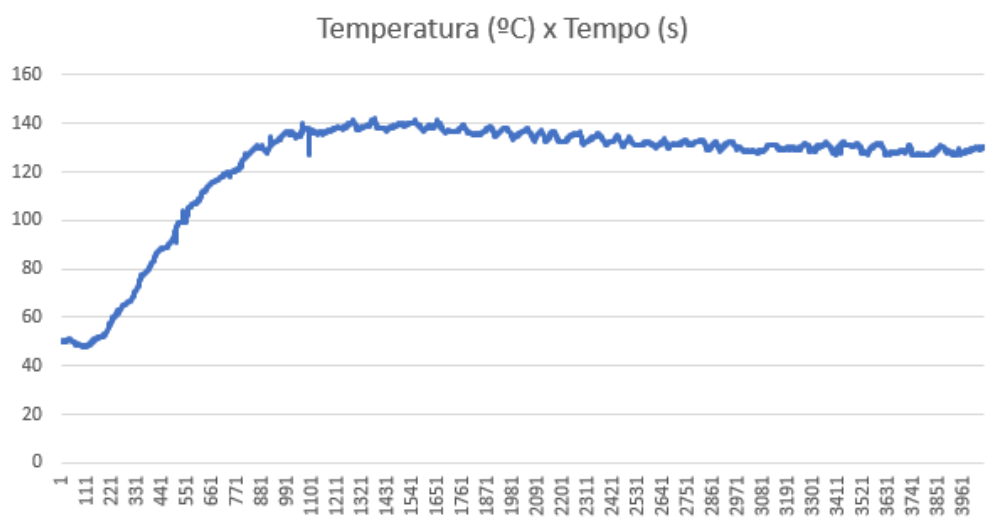
Foram realizados testes com os valores de  $K_p$  e  $K_i$  para o controle PI, calculados pela função de transferência e equação de Hägglund, que foi a melhor que

se adaptou ao sistema do projeto.

Ajustes manuais nos valores foram realizados, e o melhor resultado geral, foi com os valores de  $K_p=2$  e  $K_i= 0.005$ .

Na Figura 40, mostra-se um gráfico do comportamento do sistema em malha fechada, para uma temperatura de referência de 126 graus Celsius.

**Figura 40 - Resultados da malha fechada**



Fonte: Autoria própria (2018).

Foram alcançados resultados satisfatórios para todas as funções de gerenciamento que haviam sido previstas e que possíveis de serem desenvolvidas neste período. Desenvolveu-se a programação de uma plataforma com comunicação sem fio, o controle PI realizado, a parte mecânica construída para se poder aplicar o projeto e o desenvolvimento de uma plataforma *WEB* para o monitoramento e gerenciamento do processo.

Foram obtidos valores de temperatura controlados dentro do esperado, onde o erro de regime permanente foi de até 10% e com uma boa estabilização da temperatura desejada, com uma rampa de subida rápida, mesmo com o projeto sofrendo interferências com as temperaturas do meio ambiente, por não se encontrar em um ambiente isolado termicamente.

## 5 CONCLUSÃO

Didaticamente, foi interessante esse desenvolvimento porque foram utilizados microcontroladores, foi desenvolvido o controle de um sistema que foi necessário passar por uma série de testes e levantamentos de dados para se conseguir chegar até a função de transferência do sistema.

Pôde-se ver a importância das disciplinas de controle de processos e de programação de microcontroladores, que permitiu utilizar as bibliotecas de controle disponíveis com o *software* e plataforma utilizados, a implementação de uma comunicação via internet com o microcontrolador através de novas plataformas de desenvolvimento.

Foi necessária uma busca por uma forma de desenvolver uma página para a comunicação e interação do usuário com o sistema, de forma fácil e eficiente, tudo isso sem o conhecimento prévio em desenvolvimento *WEB* e conseguir realizar a integração dela com o sistema. Foram grandes desafios, pois era extremamente necessário para o projeto ter uma página aonde o usuário pudesse interagir com o sistema, e visualizar em forma de gráficos e tabelas as informações fornecidas no processo.

Com trabalhos desse tipo, é possível fazer com que se tenha uma boa experiência prática, com a integração de todas essas partes. Fazer o aluno resolver um problema real utilizando as habilidades adquiridas na sua formação, é uma das ações que fazem com que o aluno se sinta competente e feliz com a sua escolha para profissão.

Por fim, como melhorias, poderia ser adicionada uma forma de resfriamento do processo, fazendo também que o mesmo pudesse ser controlado automaticamente. Também a construção de uma espécie de recipiente isolado onde o mesmo pudesse ficar, para resolver o problema da interferência de temperatura do ambiente aonde se encontra, podendo melhorar os resultados de controle de temperatura. E também, pode-se trabalhar com um servidor de dados que armazene os dados em banco de dados, para registro e análise de comportamento da temperatura ao longo do tempo.

## REFERÊNCIAS

ANALOG DEVICES. **AD620 Datasheet**. Disponível em:  
[http://www.analog.com/static/importedfiles/data\\_sheets/AD620.pdf](http://www.analog.com/static/importedfiles/data_sheets/AD620.pdf) Último acesso: 20 de nov. 2018.

BENTO, Celso Roberto. **Sistemas de Controle – Teoria e Projetos**. Érica, São Paulo, 1989.

CAPGO. **Data Acquisition and Data Loggers**. Disponível em:  
<<http://www.capgo.com/Resources/Temperature/Thermocouple/Thermocouple.html>>  
Acesso em: 20 de nov. 2018.

CORRÊA, Underléa et al. **Redes Locais sem Fio: Conceito e Aplicações**. Florianópolis. Universidade Federal de Santa Catarina, 2006.

HEATH, Nick. **Como o Node-RED da IBM está invadindo a Internet das coisas**. Disponível em: <<https://www.techrepublic.com/article/node-red/>>. Acesso em: 20 de nov. 2018.

PANORAMA POSITIVO. **Tudo o que você precisa saber sobre gerenciamento remoto**. 2018. Disponível em:  
<<https://www.meupositivo.com.br/panoramapositivo/tudo-o-que-voce-precisa-saber-sobre-gerenciamento-remoto/>>. Acesso em: 20 set. 2018.

LANDO, R. A. & Alves, S. R., **Amplificador Operacional**, Livros Editora Erica Ltda., 1985, São Paulo, Brasil.



MARCINICHEN, André Ramos. MARIN, Veronica. **Amplificadores de Instrumentação**. Universidade Estadual de Santa Catarina. Disponível em: <https://www.ebah.com.br/content/ABAAAglmUAC/amplificadores-instrumentacao#>. Último acesso: 20 de nov. 2018.

OGATA, K. **Engenharia de Controle Moderno**, Prentice Hall, 5a. Edição, 2011.

OLIVEIRA, A. L. LIMA, **Instrumentação** – Fundamentos de Controle de Processo. SENAI - Serviço Nacional de Aprendizagem Industrial. Espírito Santo. 1999.

PROJECT BLOG. **Improving the Beginner's PID** – Introduction. Disponível em: <http://brettbeauregard.com/blog/2011/04/improving-the-beginners-pid-introduction/>. Acesso em: 20 de nov. 2018.

SCERVINI, Michele. **Thermoelectric materials for thermocouples**. 2009. Disponível em: <https://www.msm.cam.ac.uk/utc/thermocouple/pages/ThermocouplesOperatingPrinciples.html>. Acesso em: 10 jul. 2018.