Alan Oliveira dos Santos

Projeto e Implementação de um Sistema de Controle Utilizando Lógica Programável

Porto Alegre - RS, Brasil

2023

Alan Oliveira dos Santos

Projeto e Implementação de um Sistema de Controle Utilizando Lógica Programável

Trabalho apresentado como requisito parcial para obtenção de grau de Engenheiro de Computação da Escola Politécnica da Pontifícia Universidade Católica do Rio Grande do Sul.
Orientador: Prof. Renan Caron Viero

Pontifícia Universidade Católica do Rio Grande do Sul - PUCRS

Escola Politécnica

Curso de Engenharia de Computação

Orientador: Prof. Renan Caron Viero

Porto Alegre - RS, Brasil

2023

Agradecimentos

Agradeço imensamente a todos que de alguma maneira contribuíram para minha formação, em especial a meus pais, por me apoiarem e auxiliarem nessa jornada, ao meu orientador Prof. Renan Viero por todo o acompanhamento durante o desenvolvimento deste trabalho, ao Prof. Marlon Moraes pelos ensinamentos na área de lógica programável, estendendo este cumprimento a todo o corpo docente da PUCRS, seus técnicos e funcionários.
.

Resumo

Sistemas de controle são utilizados para ditar o comportamento dos mais variados tipos de sinais, comumente tais sistemas são implementados fazendo uso de circuitos de propósito geral. Criar uma solução específica para este tipo de sistema através de lógica programável mostra-se interessante devido sua natureza de aplicação, controlar um ou mais sinais, a uma especificidade na ideia central dos sistemas de controle. Este estudo visa uma primeira abordagem de sistemas de controle utilizando lógica programável, usando FPGA para implementar as descrições de hardware desenvolvidas. Provendo uma estrutura básica para estudo da junção das áreas de controle e lógica programável, de maneira que possa ser usada e aperfeiçoada por terceiros, se valendo do paralelismo do hardware e do FPGA para sustentar uma performance superior frente as soluções já existentes. **Palavras-Chave**: Lógica Programável. FPGA. Sistemas de Controle. PID. Peltier.

Abstract

Control systems are used to dictate the behavior of the most varied types of signals, commonly such systems are implemented using general purpose circuits. Creating a specific solution for this type of system through programmable logic is interesting due to its application nature, controlling one or more signals, a specificity in the central idea of control systems. This study aims at a first approach of control systems using programmable logic, using FPGA to implement the developed hardware descriptions. Providing a basic structure for studying the combination of control and programmable logic areas, so that it can be used and improved by third parties, taking advantage of FPGA and hardware parallelism to sustain superior performance compared to existing solutions. **Keywords**:Control Systems. PID. FPGA. Peltier.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ASIC	Circuito Integrado de Aplicação Especifica (do inglês, *Application Specific Integrated Circuit*)
FPGA	Matriz de Portas Programáveis (do inglês, *Field-Programmable Gate Array*)
HDL	Linguagem de Descrição de Hardware (do inglês, *Hardware Description Language*)
IDE	Ambiente de Desenvolvimento Integrado (do inglês, *Integrated Development Environment*)
IEEE	Instituto de Engenheiros Eletricistas e Eletrônicos (do inglês,*Institute of Electrical and Electronics Engineers*)
PID	Proporcional Integral Derivativo
PWM	Modulação por largura de pulso (do inglês, *Pulse Width Modulation*)
VHDL	Linguagem de Descrição de Hardware VHSIC (do inglês, *VHSIC Hardware Description Language*)
VHSIC	Circuito Integrado de Velocidade Muito Alta (do inglês, *Very-High-Speed Integrated Circuit*)

Sumário

1 Introdução

Com a industrialização e posteriormente a digitalização da economia, criou-se uma alta demanda e dependência de sistemas digitais e seus circuitos integrados, sistemas esses com as mais variadas finalidades, de controlar a temperatura de um ambiente até sistemas de voo, por exemplo. Estabelece-se assim uma tríade composta por algo a se controlar, software, este sendo a camada de abstração, e o hardware, comumente um circuito integrado de proposito geral capaz de executar as mais diversas funções baseado nas instruções providas pelo software.

Nota-se que há uma camada entre o sistema que deve atuar e a entidade que executa os cálculos necessários para o funcionamento, esta camada é o software, que conecta essas duas partes devido a este hardware ser de propósito geral. Entretanto é possível criar um hardware de propósito específico (ASIC), livrando-se assim da camada de software e de detalhes que são necessários ao funcionamento de sistemas de propósito geral, como interpretação de instruções, escalonamento de outras tarefas, gerencia de memoria e demais fatores que resultam em atraso na atividade final.

Porém é necessário evidenciar o alto custo para fabricação de um circuito integrado, sendo inviável caso este não possua uma alta demanda que compense o investimento inicial. Entra em cena assim os dispositivos de lógica programável, como a Matriz de Portas Programável (FPGA), que utilizando a mesma descrição de hardware que seria enviada para fabricação, pode ser configurado, e passar a executar a função do hardware descrito.

Sistemas de controle digitais em sua grande maioria são implementados em sistemas de propósito geral, como microcontroladores ou similares. Controlam variáveis específicas e podem ser críticos, a junção desses dois fatores levam a uma afinidade entre o hardware específico e os sistemas de controle. Prover uma plataforma para que esse tipo de sistema seja implementado em FPGAs mostra-se interessante, pois o sistema de controle seria o único foco do hardware, além da chance de um sistema crítico ser preempitado ou atrasado por outra solicitação ser nula, adicionando uma camada de segurança e deixando de lado a implementação de algoritmos de escalonamento de tarefas e decodificação de instruções.

1.1 Objetivos

Esta seção ilustra os objetivos desejados ao final da execução deste projeto, desde os mais abrangentes até os de pequenos módulos do sistema.

1.1.1 Objetivos Gerais

Este estudo visa o desenvolvimento de uma descrição de hardware, que possibilite a execução de um controlador do tipo PID, e que essa descrição possa ser utilizada como base em estudos futuros que relacionem a área de controle e lógica programável.

Tal descrição deverá ser capaz de executar os cálculos necessários para a lei de controle de maneira coerente, baseado nos dados obtidos pelos sensores, e gerando o sinal de controle para que o atuador execute sua função na planta.

Ao finalizar todos os levantamentos e ensaios é esperado obter um sistema de controle do tipo PID utilizando um dispositivo de lógica programável, sendo utilizado FPGA. Este sistema será composto, além do controlador, por sensores, atuadores e a planta que será controlada, composta por uma pastilha de Peltier, pastilha essa que pode ser usada para controle de temperatura, e estes quatro componentes base devem trabalhar de maneira harmônica para que o sistema siga a referência estabelecida para a planta, e assim a descrição do controlador PID seja validada.

1.1.2 Objetivos Específicos

- Prover uma plataforma para implementação de controladores PID em dispositivos de lógica programável.
- Desenvolver uma aritmética de ponto fixo que possibilite os cálculos necessários ao controlador.
- Desenvolver os módulos adjacentes necessários a sensores, atuadores e afins.
- Configurar o FPGA, com a descrição do controlador e demais dispositivos necessários ao funcionamento do sistema de controle como um todo;
- Desenvolver um protótipo do sistema composto por FPGA, sensores, atuadores e Pastilha de Peltier;
- Ser possível controlar temperatura da Pastilha de Peltier, devendo essa seguir a referência designada;

1.2 Limitações

Este trabalho limita-se a ser uma prova de conceito da junção das áreas de controle e lógica programável. Não possuindo como foco superar as soluções já existentes, mas consolidando uma base que possa ser aperfeiçoada e então as soluções em FPGA's se aproximem

cada vez mais das soluções existentes. Assim como a planta utilizada é para fins didáticos, não possuindo grande complexidade em relação ao controle, mas servindo para os fins da prova de conceito.

2 Trabalhos Relacionados

Ogata (OGATA, 2009) denota as bases necessárias para o entendimento inicial da teoria de controle. Tipos de sistemas, erros associados, como trabalhar com as funções de transferência dos sistemas, além de utilizar o software MATLAB para demostrações, provendo um entendimento para execução de simulações de controle no software citado.

Zigler e Nichols (ZIEGLER; NICHOLS, 1942) apresenta o conceito de sintonização de controladores PID baseado em métodos empíricos, tais demonstrações trazidas por eles se mantém como o cerne deste tipo de sintonização até os dias de hoje. Åström e Hägglund (ÅSTRöM; HäGGLUND, 1995) transportam esses conceitos a uma linguagem mais atual e prática, os unindo com o restante da teoria de controle.

Rabaey *et al.* (RABAEY; CHANDRAKASAN; NIKOLIC, 2002) apresenta uma visão completa do projeto de circuitos digitais, desde o nível de transistores, utilização das construções básicas para desenvolvimentos de circuitos complexos, como memórias e multiplexadores, questões de temporização, circuitos aritméticos e seus algoritmos.

Halliday *et al.*(HALLIDAY; RESNICK; WALKER, 2008) explora as leis da termodinâmica e a troca de calor entre os corpos, estes conceitos auxiliam no entendimento do efeito termoelétrico, sendo esse as junções dos conceitos apresentados por Seebeck (SEEBECK, 1822) e Peltier (PELTIER, 1834), que levam a construção da pastilha de Peltier e seu comportamento térmico baseado no fluxo de elétrons.

De maneira a obter-se o entendimento completo do sistema é necessário atentar-se a documentação do fabricante tanto do sensor de temperatura AHT21 (ASAIR, 2021), quanto do circuito integrado BTS7960 que compõe a ponte H (INFINEON, 2004) utilizada como atuador, estes trabalhos elucidam o funcionamento dos circuitos e como a comunicação com os mesmos é realizada, além de suas limitações.

3 Referencial Teórico

Este capítulo denota as bases teóricas necessárias ao projeto e prototipação do sistema proposto no capítulo 1.

3.1 Teoria de Controle

Um sistema de controle é aquele projetado para possibilitar um comportamento específico a um ou mais sinais baseado em um sinal de referência. Este tipo de sistema possui uma série de componentes característicos necessários a execução do controle, inicialmente a planta, que é o elemento a ser controlado, o controlador que executará os cálculos segundo a lei estabelecida gerando o sinal de controle, o atuador que aplicará o sinal gerado pelo controlador sob a planta e o sensor, este último é responsável por manter o controlador atualizado sobre a situação da planta.

Além disso alguns sinais são comumente citados e utilizados para realizar o controle:

- $r(t)$: É a referência que deve ser seguida pela planta, o comportamento esperado.
- $e(t)$: Sinal de erro, formado pela diferença entre a referência e o valor obtido pelo sensor, é sobre este sinal que o controlador trabalha.
- $u(t)$: Sinal gerado pelo controlador baseado no sinal de erro, ele é o sinal aplicado na planta pelo atuador.
- $y(t)$: Saída da planta, é o sinal sensoreado e posteriormente usado para gerar o sinal de erro.

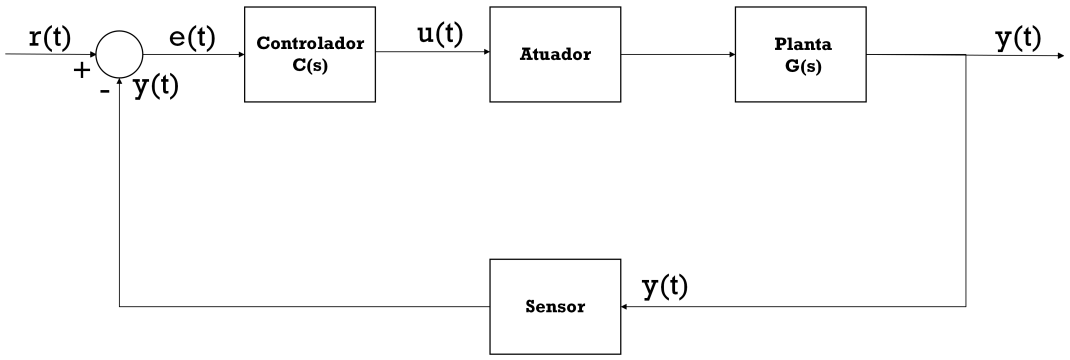

Figura 1 – Diagrama do sistema de controle e seus componentes

3.2 Ações de Controle

Dentre as ações mais comuns realizadas pelo controlador estão a ação proporcional, a ação integral e a derivativa.

3.2.1 Ação Proporcional

Sinal de controle aplicado em cada instante de tempo é proporcional à amplitude do valor do sinal de erro.

$$u(t) = K_p \; e(t) \tag{3.1}$$

Quanto maior o valor de K_p, menor o erro em regime permanente, porém este nunca pode ser zerado para plantas sem polo na origem, possuindo um erro associado denotado como erro de posição, descrito na equação 3.2 para sistemas com realimentação unitária (cujo elemento sensor possa ser aproximado por um ganho unitário).

$$e_p = \frac{1}{1 + k_p} \tag{3.2}$$

É importante frisar que a constante proporcional K_p é diferente da constante de erro de posição k_p, sendo essa definida na equação 3.3.

$$k_p = \lim_{s \to 0} G(s) \tag{3.3}$$

3.2.2 Ação Integral

Sinal de controle aplicado em cada instante de tempo é proporcional à integral do sinal de erro.

$$u(t) = \frac{1}{T_i} \int_0^{\tau} e(\tau) \tag{3.4}$$

Sendo T_i o tempo integral, definindo o tempo que a ação integral atuará sobre a resposta do sistema, trata-se de uma ação armazenadora de energia, caso $e(t) = 0$ então o sinal de controle será proporcional à energia armazenada até aquele momento. Se $e(t) = 0$ então $y(t) = r(t)$, resultando em erro nulo em regime permanente.

3.2.3 Ação Derivativa

Sinal de controle aplicado em cada instante de tempo é proporcional à variação do erro.

$$u(t) = T_d \; \frac{de(t)}{dt} \tag{3.5}$$

Ação preditiva, resultando em uma reação mais veloz do sistema, porém não podendo ser implementado de maneira isolada pois possui amplitude infinita para altas frequências, atuando apenas na resposta transitória do sistema, devido ao fato que se $e(t) = 0$ então $\frac{de(t)}{dt} = 0$.

3.3 Controlador PID

Amplamente utilizado nas mais variadas soluções industriais, o controlador Proporcional Integral Derivativo, une as três ações citadas no intuito de criar um sistema de controle robusto, que pode ser aplicado em diferentes setores e domina uma enorme fatia do mercado de controladores.

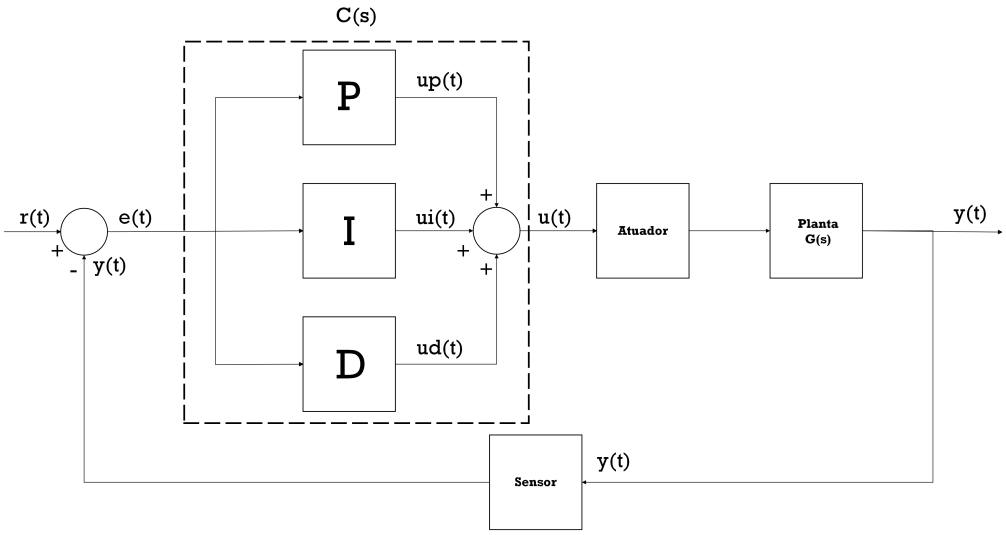

Figura 2 – Diagrama do sistema de controle PID.

Este possui as seguintes equações expandidas no domínio do tempo (3.6) e no domínio da frequência (3.7), e também possui suas versões reduzidas, que são normalmente as utilizadas, tanto no domínio do tempo (3.8), quanto no domínio da frequência (3.9).

$$u(t) = K_p\ e(t) + \frac{K_p}{T_i} \int_0^\tau e(\tau)\ d\tau + K_p\ T_d\ \frac{de(t)}{dt} \tag{3.6}$$

$$C(s) = K_p + \frac{\frac{K_p}{T_i}}{s} + K_p\ T_d\ s \tag{3.7}$$

$$u(t) = K_p\ e(t) + K_i \int_0^\tau e(\tau)\ d\tau + K_d\ \frac{de(t)}{dt} \tag{3.8}$$

$$C(s) = K_p + \frac{K_i}{s} + K_d\ s \tag{3.9}$$

3.3.1 Sintonização

Para realizar a sintonização de controladores do tipo PID, ou seja a definição dos três parâmetros do compensador. É possível utilizar tanto abordagens analíticas quanto empíricas, destacam-se entre os métodos empíricos o método do degrau de Ziegler-Nichols, onde através da resposta ao degrau da planta que se deseja controlar, é possível obter um controlador com uma resposta malha fechada aceitável a grande maioria dos sistemas.

Este método retira parâmetros da resposta ao degrau para inicialmente se obter uma função de transferência de primeira ordem que se adéque ao modelo do processo, para que possa ser realizada a simulação desta, e utiliza os mesmos parâmetros obtidos inicialmente para determinar os coeficientes do controlador PID por meio das tabelas determinadas por Ziegler-Nichols. Em posse da resposta ao degrau, os seguintes passos devem ser seguidos:

1. Ponto de Inflexão: Inicialmente é necessário traçar uma reta tangente ao ponto de inflexão da curva, este pode ser determinado através do ponto em que sua derivada assume valor máximo.

2. Atraso Aparente (L): Distância entre a origem do degrau e o cruzamento da reta tangente no eixo do tempo.

3. Ganho Integral Equivalente (a): Distância entre a origem do degrau e o cruzamento da reta tangente no eixo da amplitude.

4. Constante de Tempo 63% ($C63$): Tempo que a saída leva para atingir 63% da sua resposta máxima.

5. Constante de Tempo Dominante (T): Constante de Tempo 63%($C63$) descontado do atraso aparente(L).

6. Ganho (K): Ganho estático do sistema.

A função de transferência pode ser obtida substituindo os parâmetros encontrados na equação modelo 3.10

$$G(s) = \frac{K}{Ts+1}e^{-Ls} \tag{3.10}$$

Em um segundo momento é possível obter os parâmetros do controlador, mostrados na equação 3.9 utilizando a tabela 1:

Tabela 1 – Tabela Ziegler-Nichols

Controlador	K_p	T_i	T_d
P	$1/a$	-	-
PI	$0.9/a$	$3L$	-
PID	$1.2/a$	$2L$	$L/2$

3.4 Discretização

Devido a característica discreta da eletrônica digital, nota-se a necessidade de discretizar o controlador apresentado na equação 3.8 baseado no período de amostragem T_s, obtendo assim uma aproximação das operações de integral e derivada, já a parcela proporcional é somente convertida para o domínio discreto, como é mostrado na equação 3.11.

$$u_p[n] = K_p\ e[n] \tag{3.11}$$

3.4.1 Integrador Discreto

A operação de integração pode ser interpretada como a soma de retângulos de tamanho infinitesimal sob a curva, triângulos esses que possuem base de tamanho T_s, assim a integral pode ser descrita da seguinte maneira:

$$\int_{-\infty}^{n} x(t)\ dt \approx T_s \sum_{k=-\infty}^{n} x[k] \tag{3.12}$$

Usando a equação 3.12 de ponto de partida é possível obter a lei de controle da parcela integral por meio da equação de recorrência e assim implementar um integrador discreto que pode ser descrito utilizando-se de sua saída anterior:

$$u_i[n] = u_i[n-1] + K_i\ T_s\ e[n] \tag{3.13}$$

3.4.2 Derivador Discreto

É possível obter uma aproximação da variação baseado na amostra anterior, através da subtração da amostra atual pela anterior, levando também em conta o período de amostragem T_s, como pode ser visualizado na equação 3.14.

$$\frac{dx}{dt} \approx \frac{x[n] - x[n-1]}{T_s} \tag{3.14}$$

Desta maneira é possível obter a aproximação discreta da parcela derivativa do controlador PID:

$$u_d[n] = \frac{K_d}{T_s}\ (e[n] - e[n-1]) \tag{3.15}$$

3.4.3 Controlador PID Discreto

Utilizando as equações 3.11, 3.13 e 3.15, a seguinte lei de controle do PID pode ser obtida e posteriormente implementada em plataformas que utilizam lógica discreta:

$$u[n] = u_p[n] + u_i[n] + u_d[n] \tag{3.16}$$

3.5 Lógica Programável

Dispositivos de lógica programável, são circuitos integrados que podem ser configurados a partir de um descrição de hardware utilizando alguma linguagem de descrição de hardware (HDL), sendo assim, podem assumir as características funcionais do circuito lógico descrito e passar a executar a função deste.

FPGAs são um dispositivo que cumpre a descrição acima, sendo composto de três componentes básicos:

- Bloco Lógico: Configurado com a função lógica baseado na descrição utilizada.
- Matriz de Interconexão: Responsável pelo roteamento entre blocos lógicos, também é configurado a partir da descrição, conectando os blocos lógicos de diversas maneiras para formar funções mais complexas.
- Bloco de Entrada e Saída: Executam a conexão com o mundo externo.

3.5.1 VHDL

Criada pelo departamento de defesa dos Estados Unidos em meados da década de 80, com o intuito de padronizar projetos de circuitos da época, destaca-se até os dias de hoje como umas das principais HDLs, possibilitando as mais variadas e complexas construções de circuitos integrados, permitindo descrição, síntese, teste e simulação de projetos, além da integração com uma série de IDEs, facilitando as possibilidades citadas.

A linguagem possui as operações lógicas básicas e as operações aritméticas de adição e subtração, além de tipos básicos, que podem ser usados para construir tipos mais complexos, como vetores e matrizes. Possuindo também suporte a multiplicação e divisão, porém as ferramentas de síntese normalmente não são capazes de mapear tais operações para os dispositivos, sendo necessário desenvolver os circuitos manualmente.

Todo esse arcabouço disponibilizado pela linguagem pode ser utilizado para elaboração de memórias, registradores de deslocamento, máquinas de estados finitos, circuitos aritméticos, reconhecedores de padrões, a gama de circuitos lógicos a serem criados é infinita.

3.5.2 Representação de Ponto Fixo

Nota-se que a linguagem VHDL não possui em seu padrão suporte para representação de números racionais ou uma aproximação deste, porém todo sistema digital armazena seus dados da mesma maneira, de maneira binária, possibilita-se assim a criação de uma representação de ponto fixo utilizando um vetor binário já suportado pela linguagem.

Para desenvolver uma aritmética de ponto fixo coerente é possível utilizar sua semelhança com a aritmética de inteiros. Operações sobre dois inteiros em complemento de 2 ou dois racionais em complemento de 2, representado em ponto fixo neste estudo, equivalem-se. Assim as operações de adição e subtração binárias básicas podem ser utilizadas para possibilitar as operações de multiplicação e divisão. A tabela 2 trás exemplos em uma representação de 32 bits, com 16 para parte inteira e 16 para parte fracionária:

Tabela 2 – Exemplos de representações de ponto fixo

Base 2	**Base 16**	**Base 10**
0000000000000001.0000000000000000	0001.0000	1.0
0000000000000000.1000000000000000	0000.8000	0.5
0000000000000000.0000000000000001	0000.0001	0.000015258
0111111111111111.1111111111111111	7FFF.FFFF	32767.9999847412109375
1111111111111111.0000000000000000	FFFF.0000	-1
1111111111111111.1000000000000000	FFFF.8000	-0.5
1111111111111111.1111111111111111	FFFF.FFFF	-0.000015258
1000000000000000.0000000000000000	8000.0000	-32768

3.5.2.1 Adição

Ao realizar a soma binária de dois bits, é necessário observar sua reserva, comumente denotado como "vai-um", tanto aquela gerada por uma adição interior, quanto a que será gerada para operação subsequente. Na tabela 3 é apresentada a relação entre as entradas e as saídas geradas pelo circuito somador.

3.5.2.2 Subtração

A subtração pode ser realizada seguindo duas vertentes quando utilizado complemento de dois, a primeira visa a reutilização do circuito de soma baseado na regra de sinais clássica, a subtração de dois termos equivale a soma do primeiro com o complemento do segundo. Esta abordagem reduz a área do chip, já que utiliza o mesmo bloco que o somador, porém é

necessário multiplexadores e inversores na entrada do segundo termo de soma, então ainda a um acréscimo de área.

A segunda abordagem implementa o circuito de subtração, seguindo a tabela 4.

Tabela 3 – Tabela Verdade Soma

Entrada Vai-Um	**Termo 1**	**Termo 2**	**Soma**	**Saída Vai-Um**
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Tabela 4 – Tabela Verdade Subtração

Termo 1	**Entrada Empresta-Um**	**Termo 2**	**Subtração**	**Saída Empresta-Um**
0	0	0	0	0
0	0	1	1	1
1	0	0	1	0
1	0	1	0	0
0	1	0	1	1
0	1	1	0	1
1	1	0	0	0
1	1	1	1	1

3.5.2.3 Multiplicação

É possível visualizar a multiplicação binária sob a óptica de uma sequência de somas binárias e deslocamentos, resultando assim no algoritmo serial na figura 3 e elucidado com mais detalhes a seguir.

1. Se algum dos termos for negativo é necessário obter seu complemento de dois, ou seja, ambos os termos devem ser positivos para executar o cálculo.

2. Avalia-se o bit menos significativo do multiplicador, se este é igual à '1' então soma-se o multiplicando ao resultado, caso '0' nada é executado.

3. Desloca-se a direita o multiplicador e o resultado.

4. Retorna-se ao passo 2 até que todos os bits do multiplicador sejam percorridos.

5. Avalia-se o sinal das parcelas iniciais, se necessário executa-se o complemento de dois do resultado, baseado na regra de sinais da matemática clássica.

3.5.2.4 Divisão

A divisão binária de maneira serial, assemelha-se a multiplicação, porém baseia-se na avaliação da subtração entre o termo parcial e o divisor, seguindo o algoritmo na figura 4. Porém a uma pequena diferença em relação ao laço de repetição do algoritmo, a divisão deve se repetir pelo tamanho de bits somado dele mesmo dividido por 2. A seguir os passos necessários a operação são descritos em detalhes.

1. Se algum dos termos for negativo é necessário obter seu complemento de dois, ou seja, ambos os termos devem ser positivos para executar o cálculo.

2. Inicializa-se o termo parcial, o igualando a zero.

3. Desloca-se o dividendo e o termo parcial a esquerda.

4. Subtrai-se o divisor do termo parcial.

5. Caso a subtração resulte em um número negativo, o bit menos significativo do dividendo recebe '0'; caso este resulte em número positivo, o termo parcial recebe o resultado da subtração e o bit menos significativo do dividendo recebe '1'.

6. Neste passo a aritmética inteira e de ponto fixo se diferem, deve-se retornar ao passo 3 até que todos os bits do dividendo sejam deslocados uma vez, como na aritmética inteira, mais seu tamanho sobre 2.

7. Ao final do processo, o dividendo conterá o resultado e o termo parcial o resto da divisão.

8. Avalia-se o sinal das parcelas iniciais, se necessário executa-se o complemento de dois do resultado, baseando-se na regra de sinais.

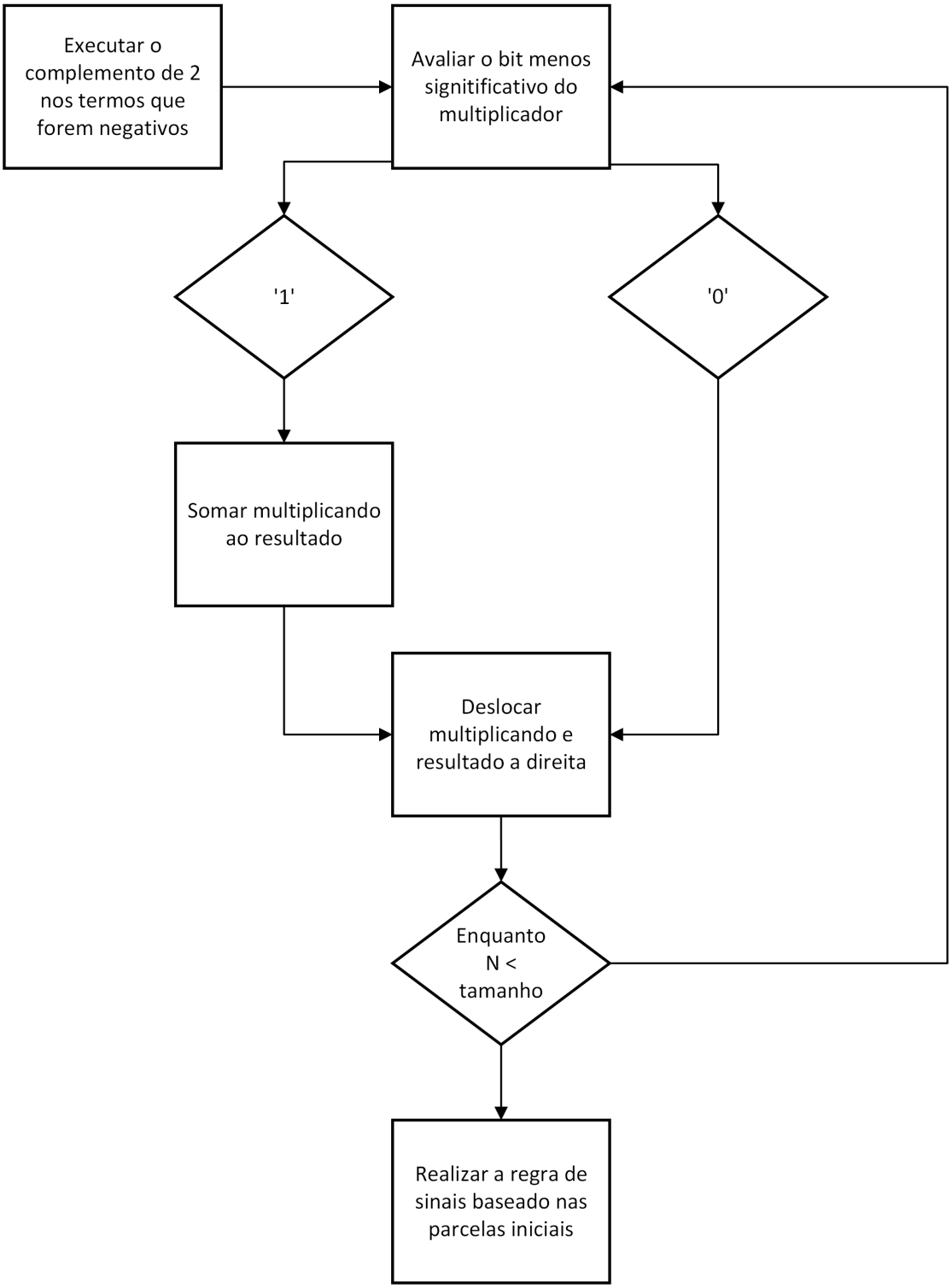

Figura 3 – Algoritmo de Multiplicação.

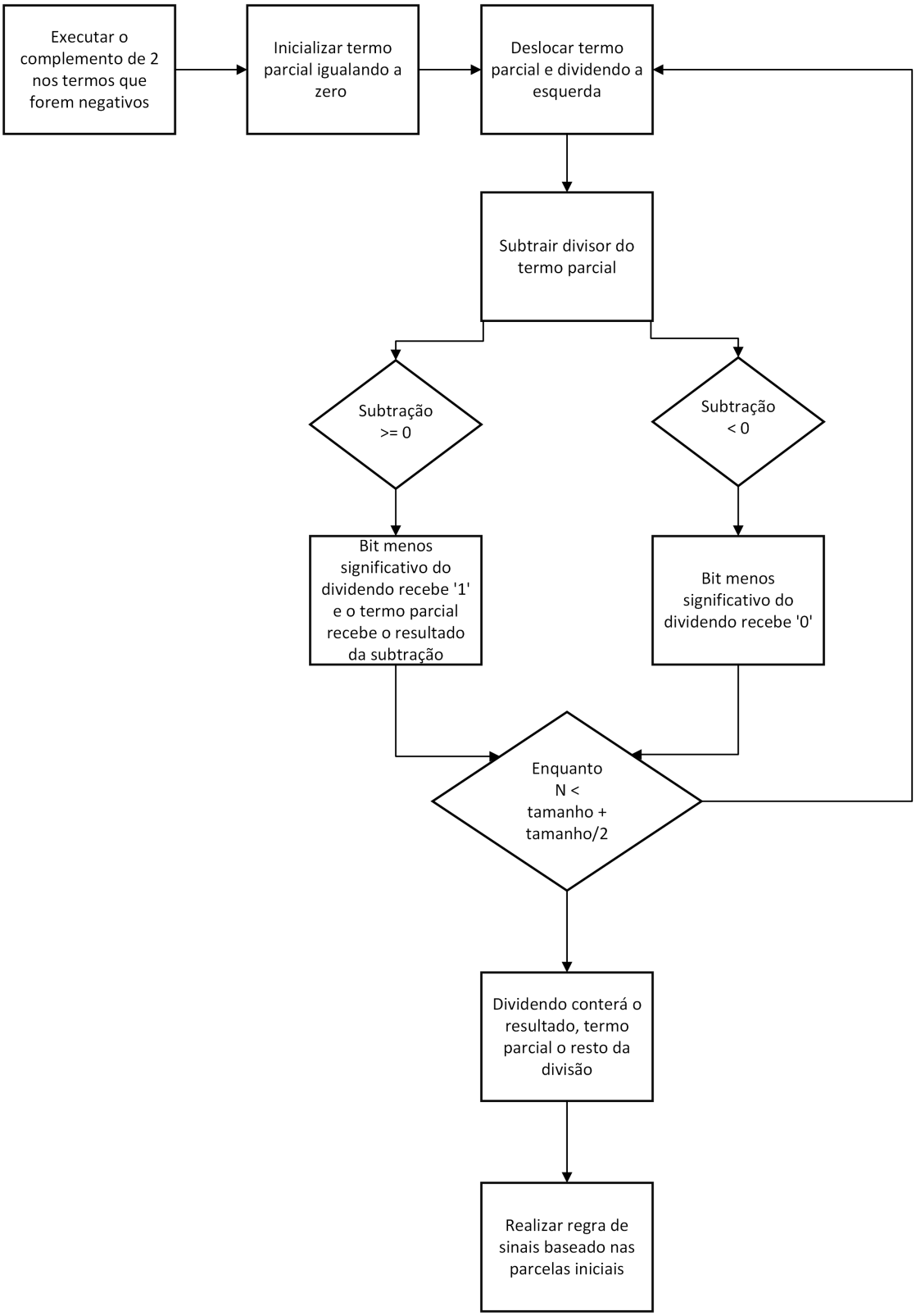

Figura 4 – Algoritmo de Divisão.

3.6 Pastilha de Peltier

Baseado nos trabalhos de Thomas Johann Seebeck(1770-1831) com o efeito Seebeck, onde observou-se que através da diferença de temperatura entre dois terminais é induzida uma corrente entre eles, Jean Charles Athanase Peltier(1785-1845) observou que é possível obter um gradiente de temperatura ao induzir uma corrente entre dois terminais condutores ou semicondutores. Estes efeitos são complementares, e juntos são denotados como efeito termoelétrico.

Utilizando-se de semicondutores do tipo N-P, com a correta organização, é possível criar um arranjo que possuirá uma face que absorverá calor(lado frio) e na superfície oposta dissipará calor(lado quente), o sentido da corrente definirá o comportamento térmico que cada face executará, este conjunto comumente é chamado de pastilha de Peltier. Este comportamento é obtido devido às características dos semicondutores N-P, semicondutores do tipo N absorvem calor no terminal negativo e dissipam no terminal positivo, já no tipo P o calor é dissipado no negativo e absorvido no terminal positivo.

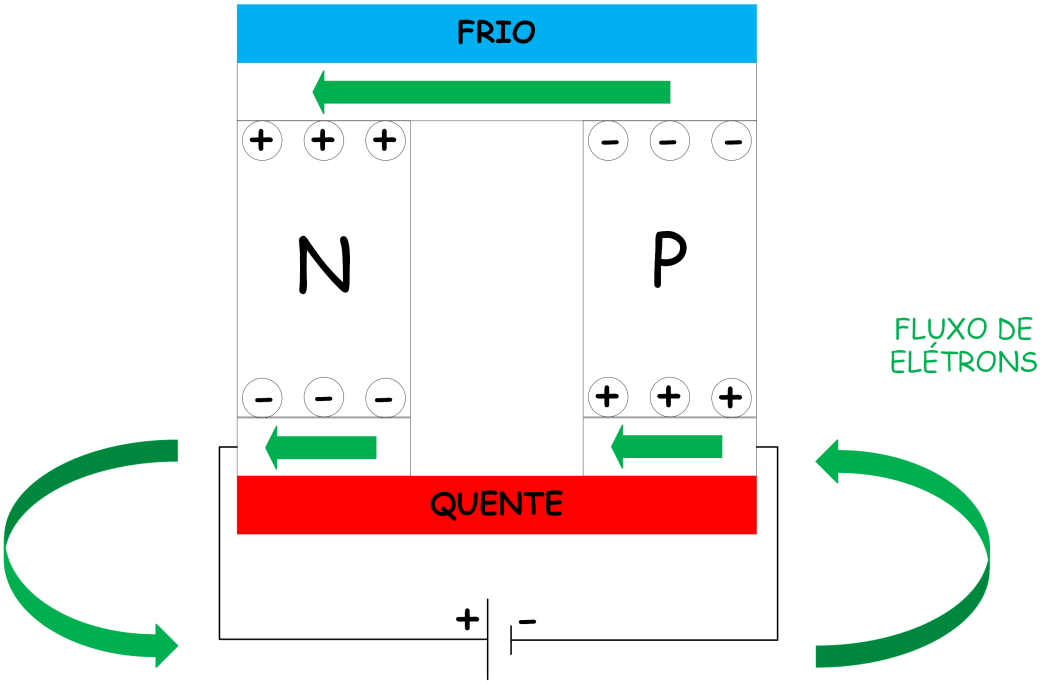

Figura 5 – Diagrama esquemático da pastilha de Peltier.

3.7 Módulo Ponte H: BTS7960

Desenvolvida inicialmente para controle da direção de rotação de motores, a ponte H pode ser vista como uma inversora do sentido da corrente de um circuito. Formada por

um arranjo de quatro transistores, a ativação dos pares corretos transportam a corrente da direita para esquerda ou da esquerda para direita, definindo assim o sentido da rotação do motor. Porém fica claro que este comportamento de direcionar o sentido da corrente de um lado a outro e a possibilidade de inverter essa conduta, é o mesmo desejado para definir a função térmica de cada superfície da pastilha de Peltier.

O circuito integrado BTS7960, trata-se de meia ponte H para cargas que necessitam de alta corrente, possuindo assim dois transistores MOSFET, um de canal N, que tem seu bloco de operação chamado de LSS e outro de canal P denotado com HSS. Relacionado a pinagem do circuito BTS7960, destacam-se IN que controla os transistores e INH que é o habilitador do circuito, além dos pinos que compõe a lógica de alimentação da carga GND, VS e OUT, que se conectam a carga baseado na ativação dos transistores.

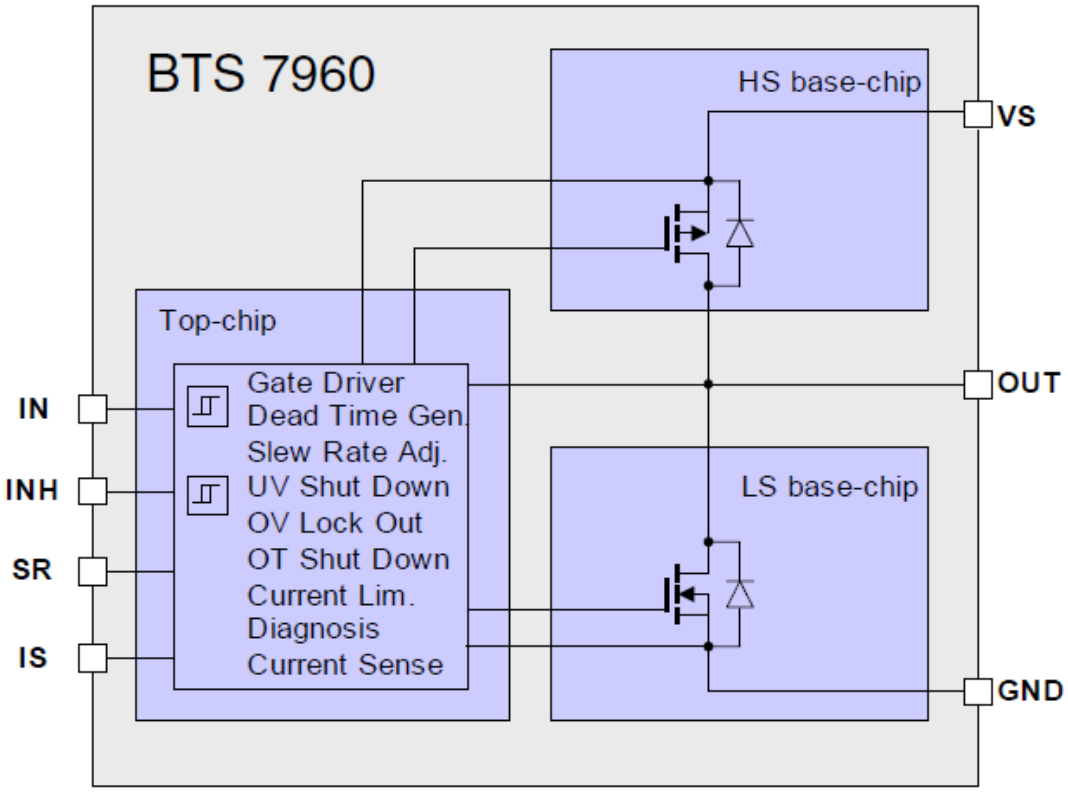

Figura 6 – Diagrama de bloco BTS7960.

Tabela 5 – Tabela Verdade BTS7960

INH	**IN**	**HSS**	**LSS**
0	X	Desligado	Desligado
1	0	Desligado	Ligado
1	1	Ligado	Desligado

Logo o módulo possui dois circuitos BTS7960 para formar uma ponte H completa e fornecer o comportamento desejado a carga, o módulo também possui dois pinos para PWM, um para o sentido à direita e outro a esquerda, além de dois pinos para habilitação do PWM dos respectivos sentidos, um conector para a carga e outro para fonte de alimentação da carga. A interconexão correta entre as entradas e ambos BTS7960 é realizada pelo buffer de três estados 74HC244D, baseado na habilitação dos pinos de PWM, este aciona a ponte H com o comportamento desejado.

3.7.1 Modulação por Largura de Pulso

O PWM é uma técnica amplamente utilizada para controle de tensão sobre uma carga, pode ser resumida como o ato de ligar e desligar a fonte de alimentação com uma temporização controlada, aplicando assim sobre a carga a tensão média baseado na porcentagem de tempo da fonte ativa, esta porcentagem é chamada de razão cíclica.

Tabela 6 – Exemplos de PWM

Tensão da Fonte (V)	**Razão Cíclica (%)**	**Tensão Média sobre a carga (V)**
5	100	5
5	50	2.5
5	0	0
12	100	12
12	75	9

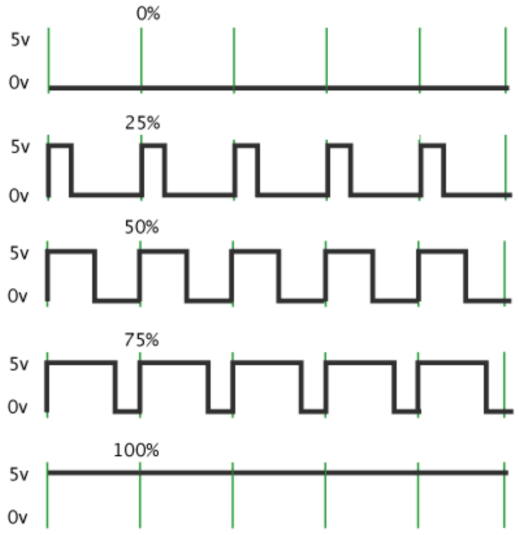

Figura 7 – Exemplos de PWM.

3.8 Módulo Sensor de Temperatura: AHT21

Módulo responsável por determinar o estado atual da planta através do sensoriamento da sua temperatura pelo sensor AHT21, este possui a capacidade de aferir temperaturas de -50 a +80 graus Celsius, com uma precisão de ±0.5 graus, cumprindo assim o requisito de projeto da pastilha de Peltier.

Comunicando-se através do protocolo I2C, também sendo capaz de obter a umidade relativa, porém esta não será utilizada neste estudo, o módulo ainda possui um capacitor de filtro ligado ao pino de alimentação e resistores de pull-up que são necessários para comunicação no protocolo I2C, mais detalhes serão demostrados na seção 3.8.1.

A sequência de solicitação de conversão de dados e leitura de dados podem ser observadas nas figuras 8 e 9 respectivamente. Além disso duas temporizações devem ser respeitadas, a primeira referente ao tempo de inicialização do sensor. Após o sensor ser ligado deve-se aguardar 100 milissegundos até a primeira solicitação, além do tempo para conversão de dados que é de 80 milissegundos.

O dado obtido do sensor, pode ser considerado um dado bruto, é necessário sua conversão utilizando a equação 3.17, obtendo assim a temperatura lida em graus Celsius.

$$Temperatura = \frac{dado}{2^{20}} * 200 - 50 \tag{3.17}$$

3.8.1 Protocolo I2C

Protocolo desenvolvido nos anos 1980, o I2C foi criado com o intuito de conectar uma série de periféricos de baixa frequência em um dispositivo, através de somente dois barramentos, a linha de dados (SDA) e a linha de sincronismo (SCL). Comumente chamado de protocolo do tipo mestre-escravo, o mestre I2C após a rotina de inicialização, escreve o endereço do escravo na linha de dados e se executará uma leitura ou escrita enquanto gera pulsos na linha de sincronismo, quando um escravo identifica seu endereço esse responde colocando o barramento de dados em nível lógico baixo devido ao reconhecimento (ACK) e ficando de prontidão para receber ou enviar dados ao mestre, desta maneira cada escravo no barramento deve ter um endereço único, ou mais de um responderá o mestre deixando a comunicação inviável.

É importante frisar que normalmente os dispositivos I2C, não possuem seu endereço completamente estático, alguns bits podem ser definidos pelo projetista e assim sensores iguais podem ser colocados no mesmo barramento I2C, porém o AHT21 possui todos os seus bits definidos, acarretando em uma escolha caso mais de 1 sensor necessite ser utilizado, ou utiliza-se mais pinos, criando outro barramento I2C, ou se faz uso de um multiplexador I2C.

Figura 8 – Temporização de inicio da conversão de dados do AHT21.

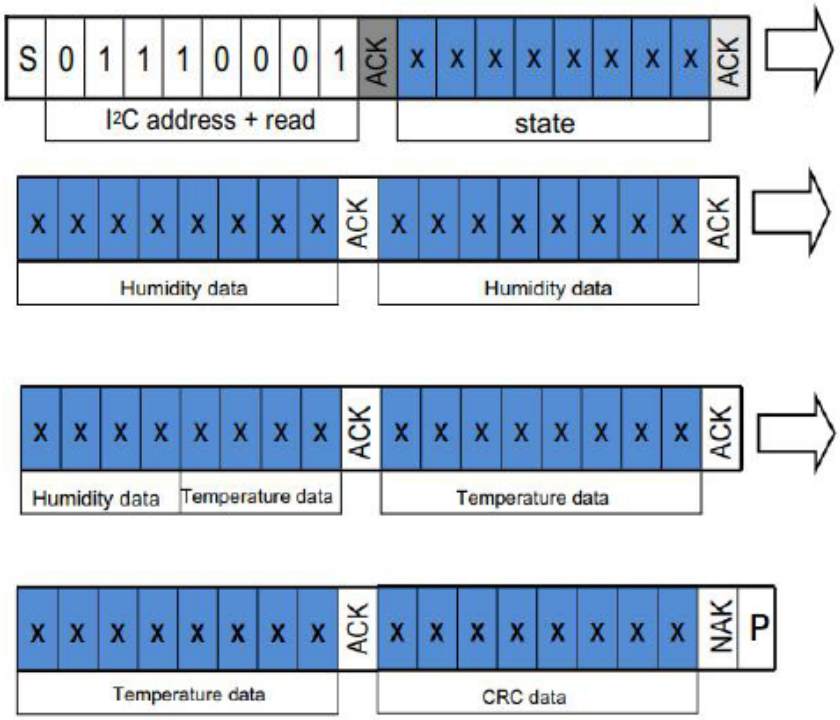

Figura 9 – Temporização de leitura de dados do AHT21.

4 Metodologia

Este capítulo elucida os métodos para construção do projeto, a ligação entre o projeto físico e os levantamentos teóricos feitos no capítulo 3, detalhes de implementação e montagem dos protótipos, além de uma explicação da descrição de hardware do controlador e demais blocos necessários para funcionamento do sistema no FPGA.

4.1 Montagem do Protótipo

O projeto físico inicia-se pela planta, com o envolvimento da pastilha de Peltier em pasta de condução térmica e posteriormente sua colocação entre duas placas metálicas, estas funcionam como dissipadores de calor, seria como se afastassem a face quente da fria, e caso isso não aconteça, baseado na lei zero da termodinâmica, onde os corpos tendem a buscar o equilíbrio térmico, a face quente transferiria calor para a face fria e a pastilha ficaria em equilíbrio térmico, não possuindo um gradiente de temperatura.

Em cada uma das placas metálicas foi fixado um sensor de temperatura AHT21 juntamente com um isolante elétrico, já que o sensor ficará em contato com um bom condutor e assim evitando curtos entre suas conexões elétricas, a diferença entre a leitura desses dois sensores é o gradiente de temperatura, a variável a ser controlada. Com o arranjo da planta finalizado, a pastilha foi colocada no conector destinado à carga da ponte H, e uma fonte de alimentação de 5 volts também foi conectada à ponte H para fornecer energia elétrica a pastilha de Peltier. Além disso todas as conexões lógicas para os sensores e para as entradas lógicas da ponte H foram preparadas para suas conexões futuras. A montagem pode ser vista na figura 10.

4.2 Ensaios Preliminares

Inicialmente dois ensaios foram realizado para se chegar a um sistema de controle mais coeso e com sua lei de controle já validado, primeiramente o ensaio da planta para extração dos parâmetros para o controlador e sua lei, e o ensaio desta lei em uma plataforma já consolidada, para que os testes no FPGA se limitem as descrições de hardware desenvolvidas.

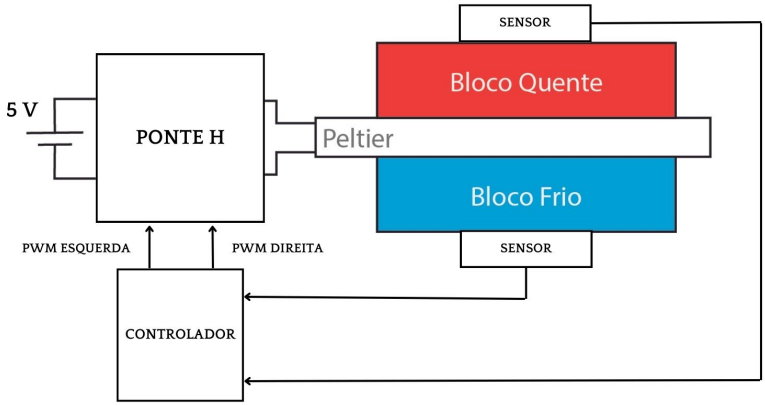

Figura 10 – Protótipo do sistema.

4.2.1 Ensaio da Planta

Para realização da sintonização os passos citados na seção 3.3.1 foram seguidos, iniciando com a aplicação de um degrau com 100% de razão cíclica sobre a pastilha, que devido a fonte conectada equivale-se a aplicação de 5 volts, extraindo a resposta em malha aberta da planta durante 500 segundos, obtendo um gradiente de temperatura máximo de aproximadamente 28 graus, que pode ser visualizado na figura 11.

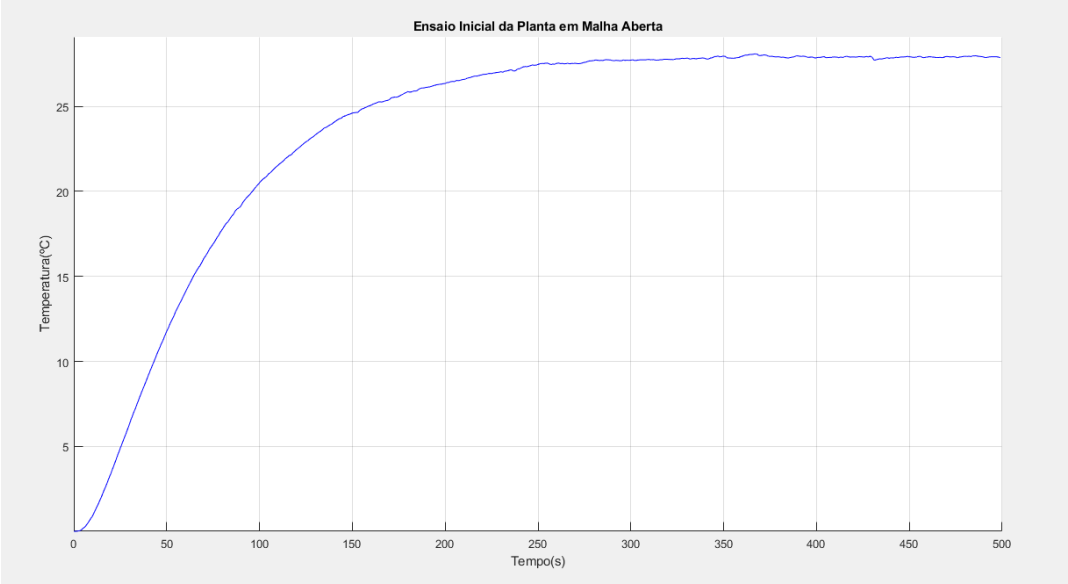

Figura 11 – Resposta ao Degrau da Planta em Malha Aberta.

Em posse da resposta em malha aberta da planta, executa-se a derivada em busca do

seu ponto de maior valor, traçando assim a reta tangente ao ponto de inflexão e retirando os parâmetros necessários, como é visualizado na figura 12.

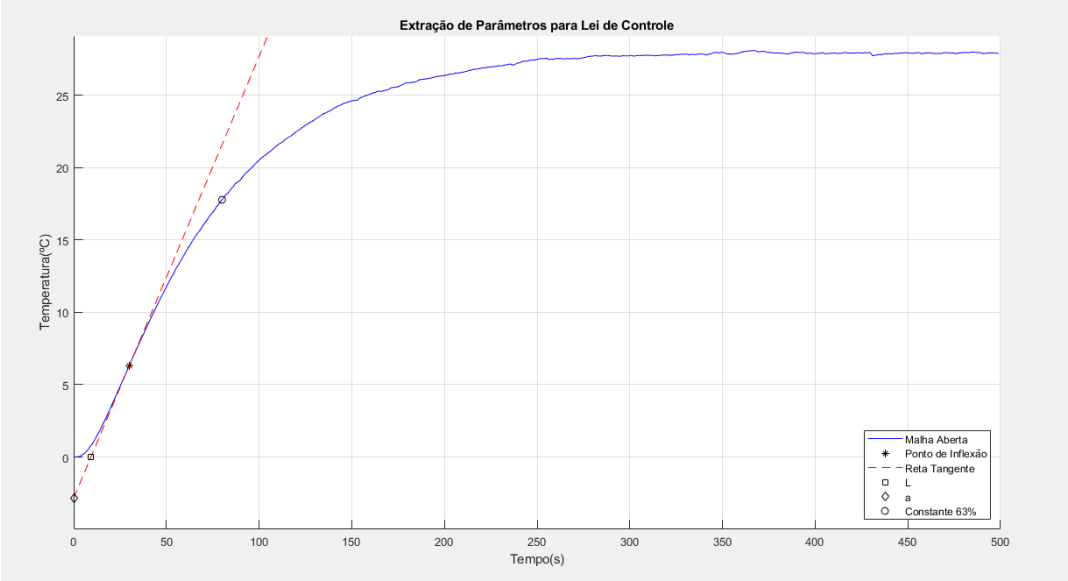

Figura 12 – Extração de parâmetros para lei de controle.

Obtém-se assim o atraso aparente como $L = 9.2787$, o ganho integral equivalente como $a = 2.83$, a constante 63% como $C63 = 80$ segundos, constante de tempo dominante $T = 70.7213$ e o ganho $K = 28$. Ao aplica-lós na equação 3.10, temos a equação 4.1 que descreve o modelo da planta e a comparação entre sua resposta em malha aberta com a resposta em malha aberta obtida através do ensaio experimental na figura 13.

$$G(s) = \frac{28}{70.7213s + 1} e^{-9.2787s} \tag{4.1}$$

Então utilizando a tabela 1, é possível obter os parâmetros de ziegler-nichols para o controlador(tabela 7) e posteriormente suas constantes(tabela 8).

Tabela 7 – Parâmetros para o controlador

Controlador	K_p	T_i	T_d
P	0.3534	-	-
PI	0.3180	27.8361	-
PID	0.4240	18.5574	4.6393

Em posse das constantes do controlador é possível aplica-lós na equação 3.9, e juntamente com a equação 4.1 simular o sistema em malha fechada com os três tipos de controladores.

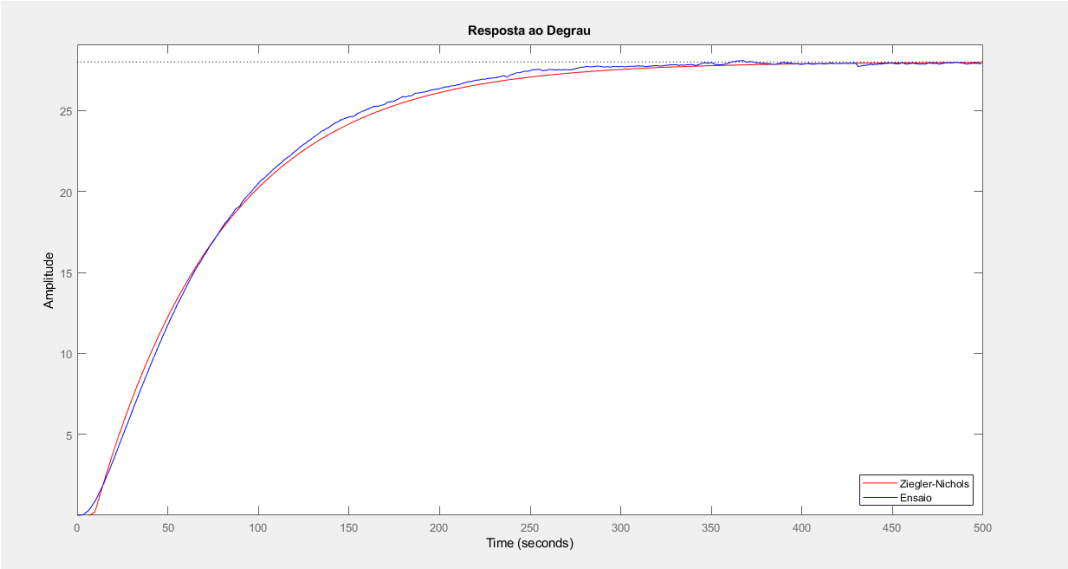

Figura 13 – Comparação do ensaio em malha aberta da planta e da sua aproximação via Zigler-Nichols.

Tabela 8 – Constantes do controlador

Controlador	K_p	K_i	K_d
P	0.3534	-	-
PI	0.3180	0.0114	-
PID	0.4240	0.0228	1.9672

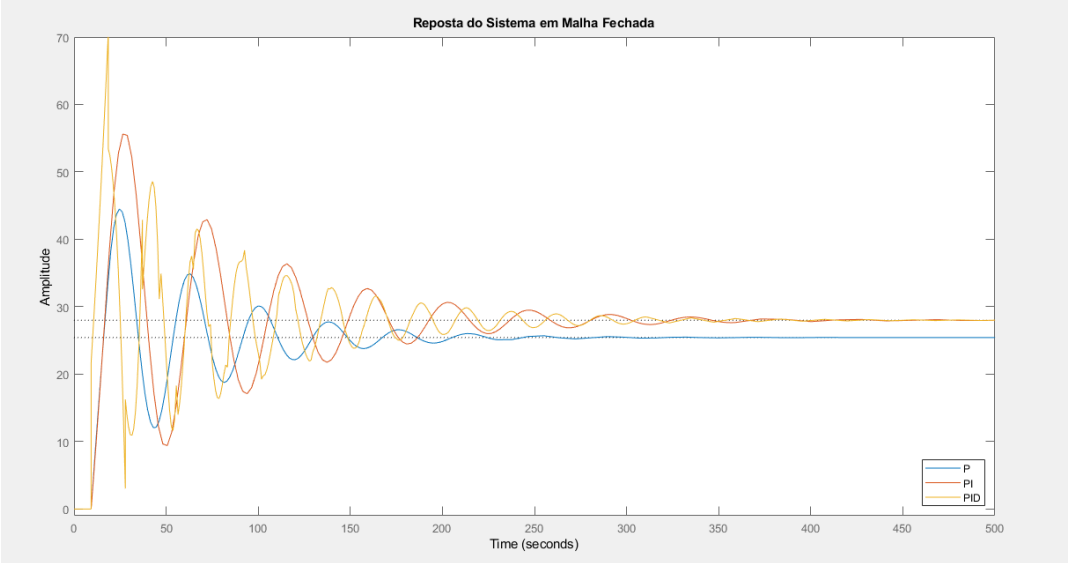

Figura 14 – Simulação da planta em malha fechada com seus respectivos controladores.

4.2.2 Ensaio do Controlador

Para validar a lei de controle obtida via ziegler-nichols e auxiliar como um comparativo com a solução no FPGA, foi utilizada a já difundida plataforma de desenvolvimento NodeMCU, que muitas vezes é a solução adotada para sistemas de controle. Este já possui bibliotecas para o sensor AHT21 e para o PWM que será aplicado na ponte H, estas bibliotecas serão utilizadas. A plataforma também possui uma biblioteca para execução de controladores PID, porém para também validar a equação discreta deduzida para o controlador PID (equação 3.16) e suas três parcelas, não serão utilizadas bibliotecas para este fim.

Figura 15 – NodeMCU.

Como trata-se de uma aplicação física, é necessário tratar o sinal de controle gerado, defini-se então o sinal aplicado no ensaio inicial da planta (seção 4.2.1) como a razão cíclica máxima para atuação na planta, ou seja, caso o controlador chegue a conclusão que é necessário uma atuação maior este deve ser saturado. Exemplificando, durante o ensaio foi utilizado uma fonte de alimentação de 5 volts e uma razão cíclica de 100%, logo se o controlador calcular 150% de razão cíclica, seria como atuar 150% do tempo, o que para temporização do PWM não faz sentido, ou atuar a 7.5 volts, que também é uma limitação física da fonte, então a todo momento que o sinal de controle for superior a 1, atua-se à 100%/5 volts. Essa definição também será utilizada no FPGA e suas consequências serão discutidas em capítulos posteriores.

Utilizando um período de amostragem $T_s = 1$ segundo, as equações anteriormente apresentadas e as constantes PID calculadas, frisando que é necessário ponderar K_i e K_d através do período de amostragem, os transportando do domínio do tempo contínuo para o tempo discreto, obtém-se as respostas práticas para os controladores P (figura 16) e PI (figura 17) que seguem as referências de maneira esperada.

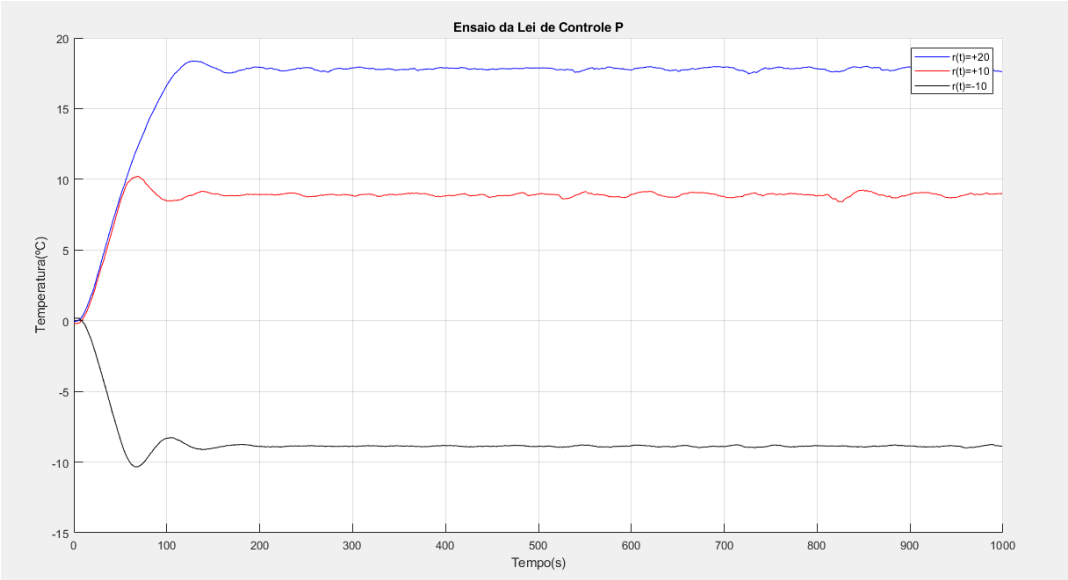

Figura 16 – Ensaio da lei de controle P na plataforma NodeMCU.

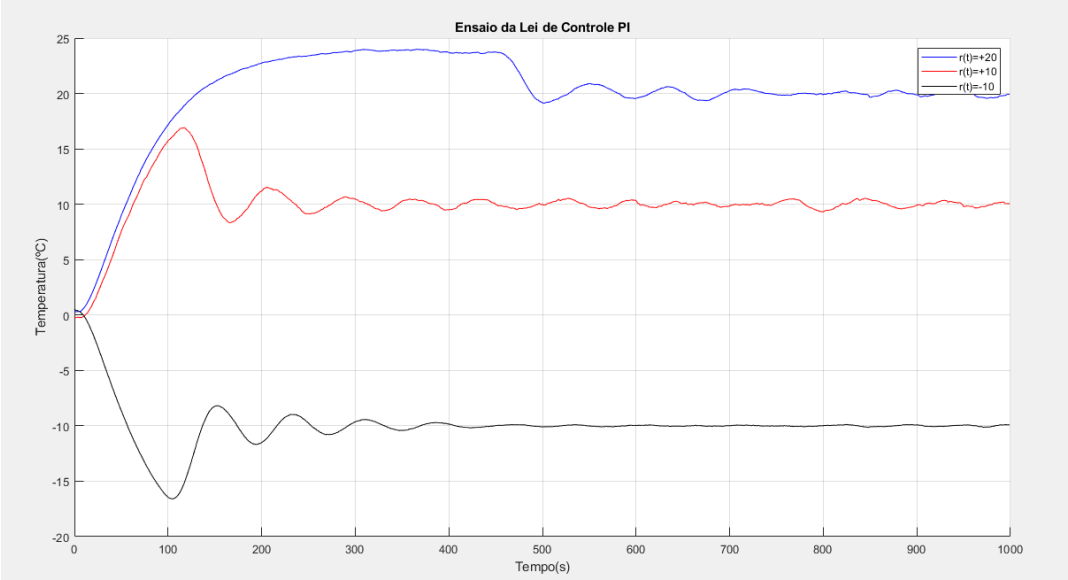

Figura 17 – Ensaio da lei de controle PI na plataforma NodeMCU.

4.3 Descrição de Hardware

O primeiro ponto do projeto de hardware é o levantamento das funções necessárias e sua divisão hierárquica. Quatro grandes blocos podem definidos inicialmente, uma descrição para o sensor, que necessita ler e converter o valor obtido, uma descrição para o controlador, outra que através do sinal de controle gere o PWM adequado, e uma ultima que seja capaz de enviar as leituras do sensor via serial para manter o mundo externo atualizado sobre a situação da planta.

Além dos já citados, outras descrições necessitam ser feitas, para suportar os quatro principais blocos, um divisor de frequência, pois nem todos os blocos utilizam a mesma, um bloco de multiplicação e outro de divisão, devendo todos os circuitos respeitarem a aritmética de ponto fixo elucidada na seção 3.5.2. Através da microarquitetura na figura 18, é possível visualizar todas as instâncias das descrições e seus principais sinais.

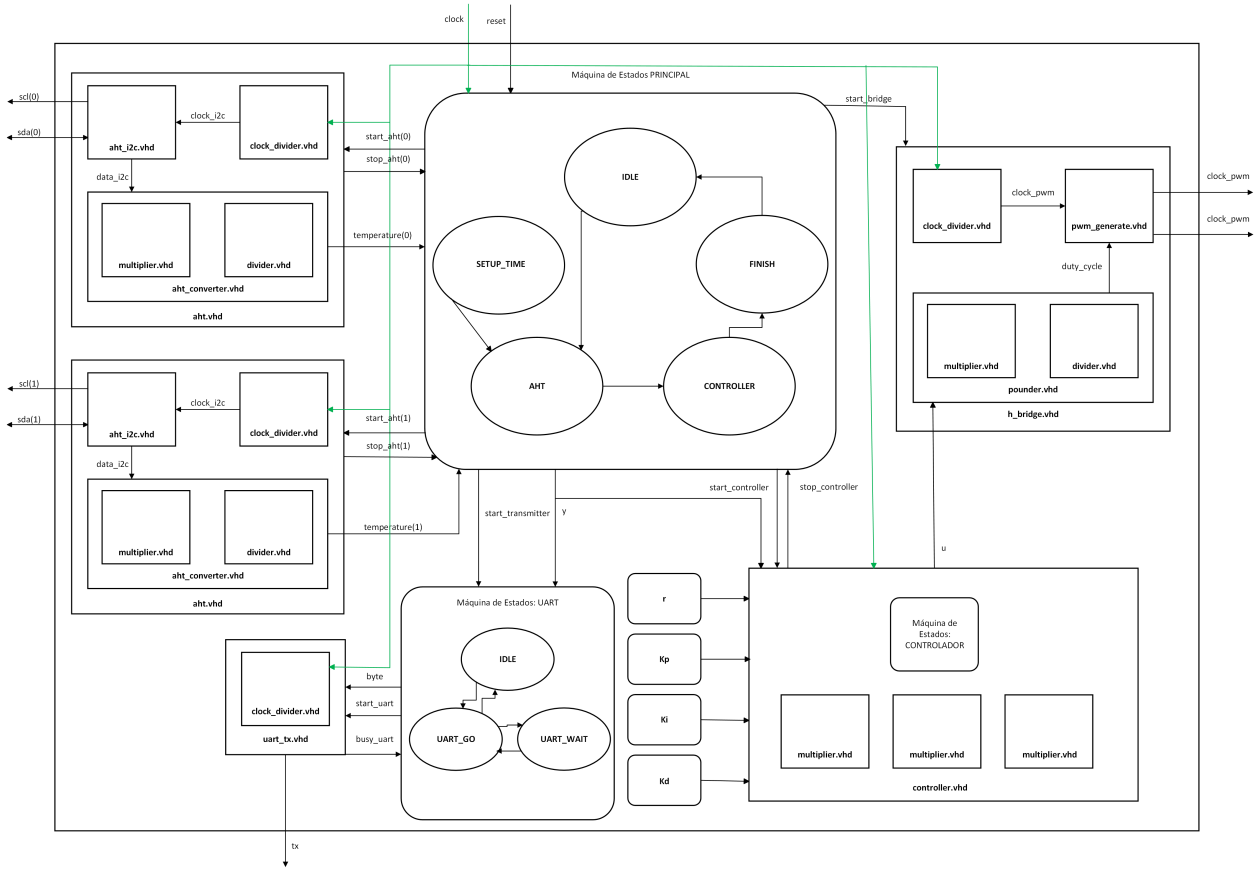

Figura 18 – Microarquitetura do hardware desenvolvido.

4.3.1 Circuitos Aritméticos

Devido a linguagem VHDL não dar suporte nativo as operações de multiplicação e divisão, é necessário desenvolver ambos seguindo os algoritmos apresentados nas seções 3.5.2.3 e 3.5.2.4. Uma simulação de ambos os circuitos pode ser vista nas figuras 19 e 20.

4.3.2 Circuito Controlador

Utilizando como suporte o circuito multiplicador, o controlador precisa executar as equações deduzidas durante a seção 3.4, recebendo como entrada as constantes do controlador

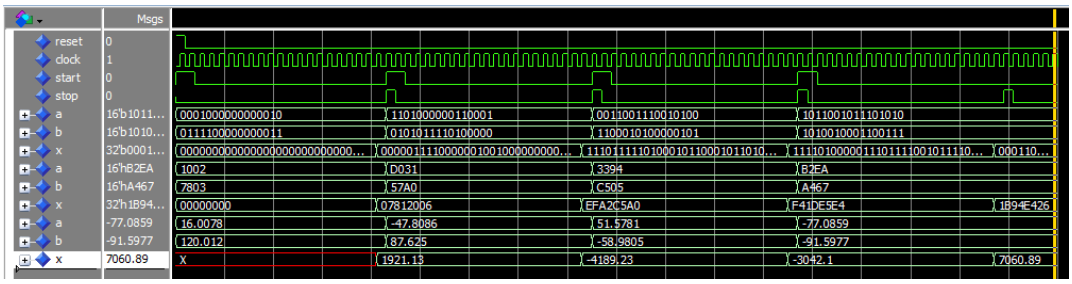

Figura 19 – Simulação do circuito de multiplicação.

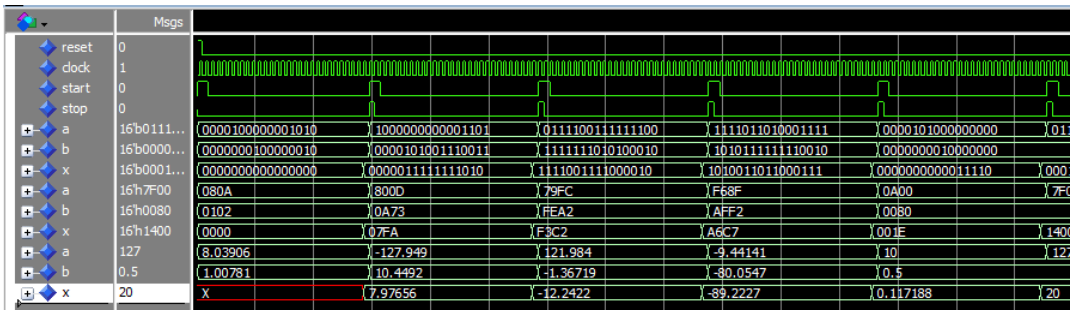

Figura 20 – Simulação do circuito de divisão.

K_p, K_i e K_d, já no domínio do tempo discreto, além do estado atual da planta e o sinal de referência, gerando então o sinal de controle $u[n]$, este terá os mesmos N bits de tamanho que as entradas. Na figura 21 é possível visualizar a estrutura interna do circuito com instanciações de outros circuitos, máquinas de estado finitos e suas transições.

4.3.3 Demais Circuitos Desenvolvidos

Iniciando pelo divisor de frequência, este se baseia em contadores para gerar uma frequência mais lenta que a frequência de entrada, é essencial a comunicação com os sensores de temperatura que utilizam um frequência de 100 KHz. Para externar o estado atual da planta, foi desenvolvida uma meia UART, esta possui somente a parte lógica referente a transmissão, e utilizando os circuitos da seção 4.3.1, forma-se um circuito de relação de valores, este executa uma regra de três.

Outro bloco relevante diz respeito ao sensor de temperatura AHT21, tal bloco deve se comunicar com os sensores através do protocolo elucidado na seção 3.8 e executar a conversão do valor obtido com o auxilio do circuitos de multiplicação e divisão. Para uma comunicação coesa ele necessita gerar a frequência necessária de 100 kHz via circuito divisor de frequência.

O ultimo bloco diz respeito a ponte H, que deve gerar o sinal de PWM, baseado na conversão do sinal de controle, o transformando na razão cíclica correta. Este bloco executa a saturação do sinal de controle e ativa a ponte no modo para direita caso o sinal de controle

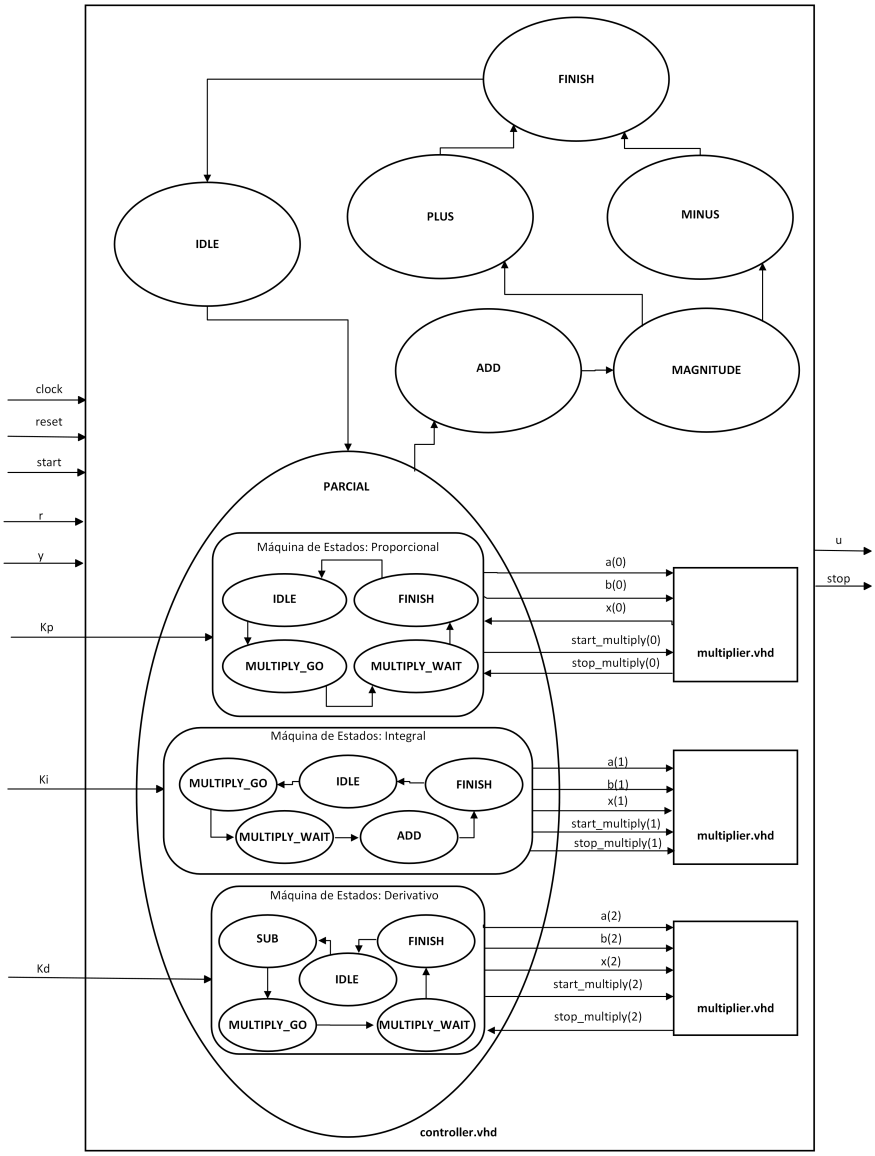

Figura 21 – Microarquitetura do controlador.

seja positivo e para esquerda caso negativo.

4.3.4 Nexys2

Para implementação dos circuitos descritos, foi utilizado a plataforma de desenvolvimento Nexys2, esta possui um FPGA Spartan-3E, e toda a estrutura de hardware necessária para o projeto, incluindo uma série de pinos para comunicação externa com os sensores e para controle da ponte H, um oscilador que gera uma frequência de 50 MHz, que será a frequência base do sistema, e um conector RS232 para comunicação serial.

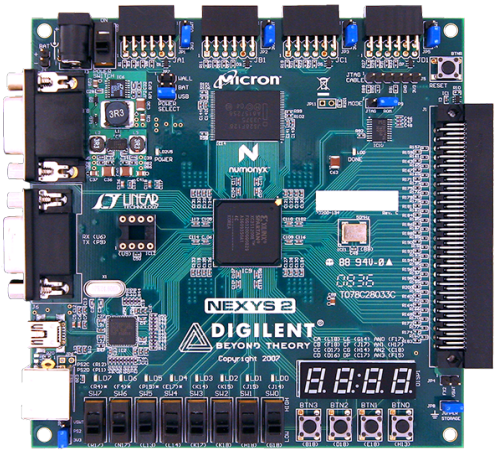

Figura 22 – Nexys2.

4.3.5 Implementação em FPGA

Em posse das constantes do controlador e as descrições finalizadas, os sensores e a ponte H foram conectados ao FPGA, cada sensor utilizou 2 pinos, totalizando 4 pinos para os sensores, 2 pinos foram utilizados para o PWM da ponte H, além do pino correspondente ao TX para realizar a comunicação serial e 1 para reinicialização assíncrona. O FPGA é capaz de fornecer 3,3 volts e até 200 mili amperes, suficiente para os sensores e entradas lógicas da ponte H, então um pino de VCC e outro de GND foi utilizado para criar um linha de alimentação, totalizando o uso de 10 pinos do FPGA.

A ferramente de síntese pode ser usada para gerar os arquivos binários de configuração, determinar sua frequência máxima de operação baseado no atraso máximo, sendo essa frequência 80 MHz, além de gerar o relatório de utilização do FPGA, visto na figura 23.

Device Utilization Summary				[-]
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	3,991	9,312	42%	
Number of 4 input LUTs	6,196	9,312	66%	
Number of occupied Slices	4,111	4,656	88%	
Number of Slices containing only related logic	4,111	4,111	100%	
Number of Slices containing unrelated logic	0	4,111	0%	
Total Number of 4 input LUTs	6,632	9,312	71%	
Number used as logic	6,196			
Number used as a route-thru	436			
Number of bonded IOBs	9	232	3%	
Number of BUFGMUXs	5	24	20%	
Average Fanout of Non-Clock Nets	3.18			

Figura 23 – Relatório de utilização do FPGA.

Desta maneira o FPGA passará a executar a função dos circuitos descritos, sendo possível acompanhar o estado atual da planta via serial. As respostas obtidas utilizando o

FPGA podem ser vistas nas figuras 24 e 25.

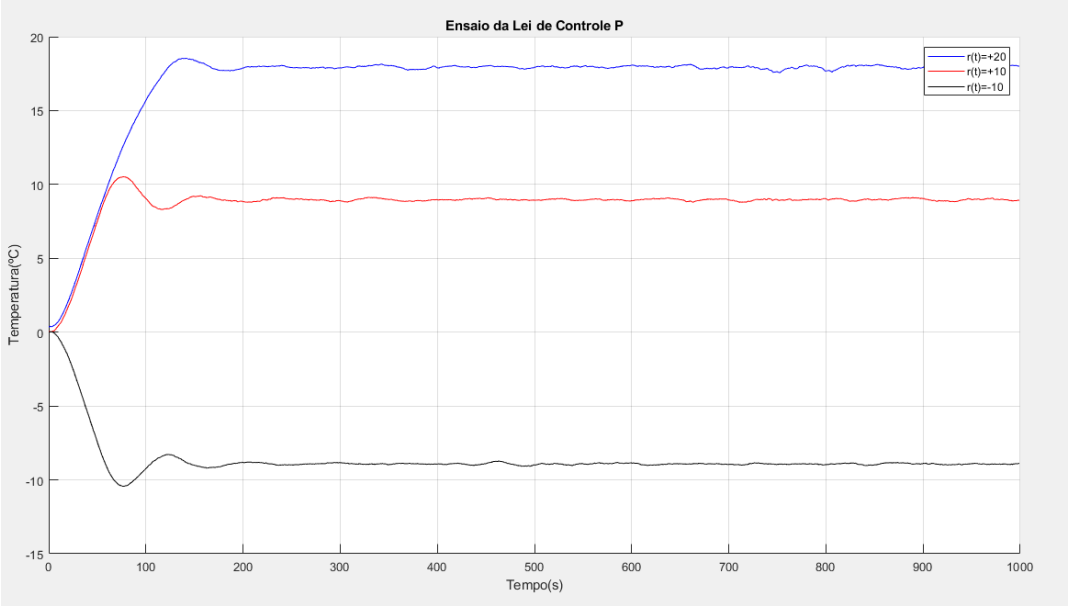

Figura 24 – Ensaio da lei de controle P em FPGA.

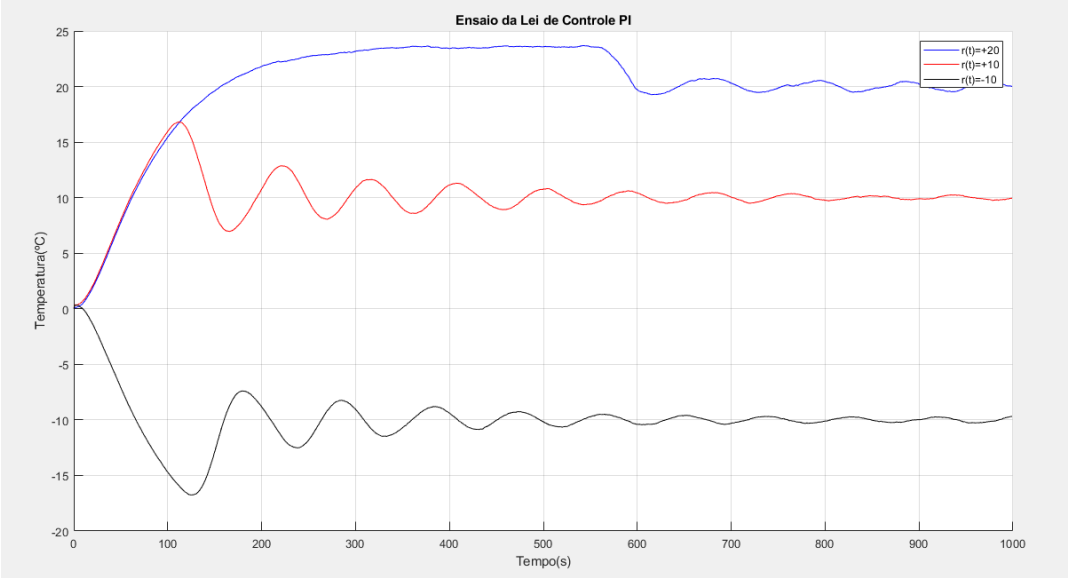

Figura 25 – Ensaio da lei de controle PI em FPGA.

5 Resultados

Este capitulo analisa os resultados obtidos durante todos os ensaios e compara as diferentes soluções utilizadas, salientando pontos fortes e pontos fracos de cada abordagem.

5.1 Respostas Práticas x Respostas Teóricas

Inicialmente as respostas práticas serão comparadas com as teóricas, avaliando o seguimento de referência e o tempo até a estabilização.

5.1.1 Controlador P

Ao observar as figuras 26 e 27, é notório que ambas as abordagens seguiram a referência, levando em consideração o erro de posição, o tempo de estabilização também é similar, a grande diferença é a oscilação. As respostas práticas oscilam levemente, diferente da teórica, tal comportamento está ligado a saturação do sinal de controle. No sistema prático a uma máxima potência que pode ser fornecida a planta, durante seu dimensionamento foi estabelecido que caso o sinal de controle seja 1 ou mais aplica-se a máxima potência. Porém o sinal de controle maior que 1 claramente é um pedido de mais potência, como não há, o sinal de controle é saturado levando ao comportamento prático divergente da simulação.

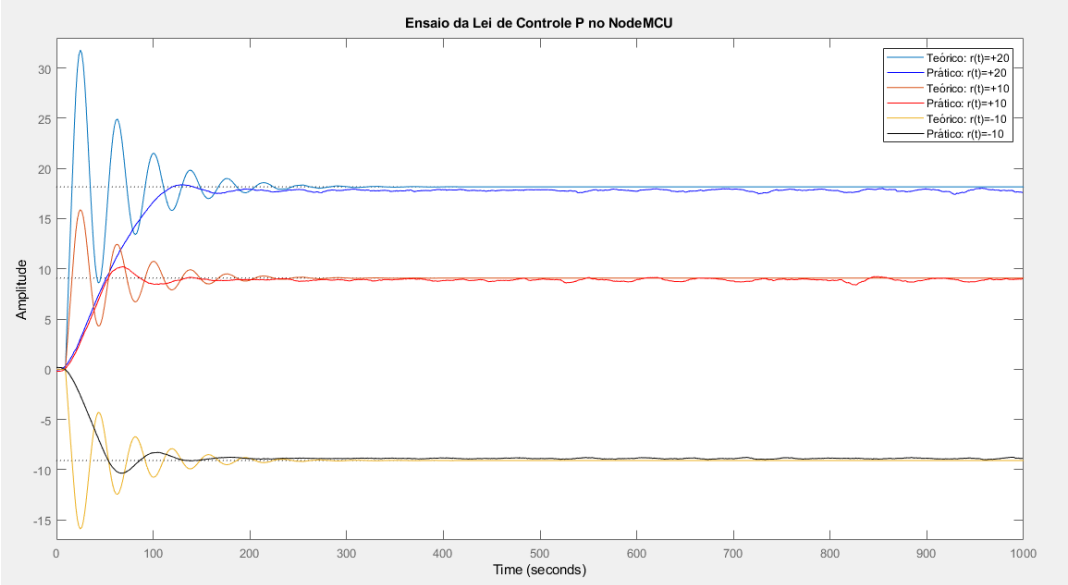

Figura 26 – Comparação da resposta teórica com a resposta prática do sistema com controlador P no NodeMCU.

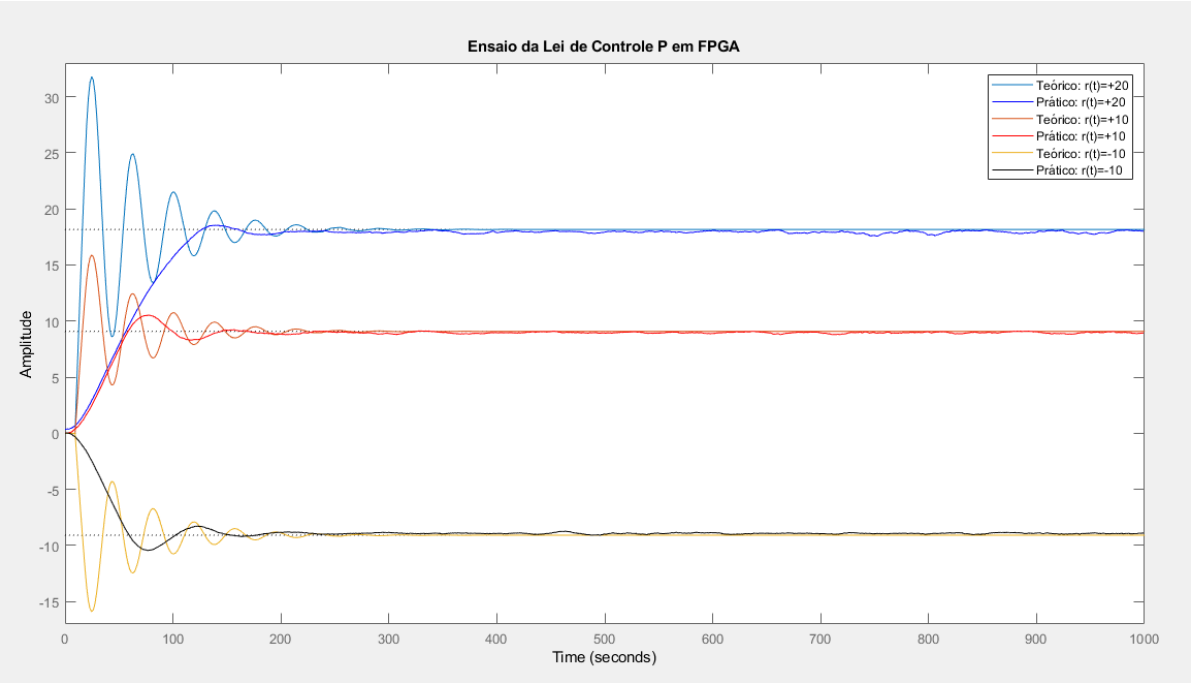

Figura 27 – Comparação da resposta teórica com a resposta prática do sistema com controlador P na Nexys2.

5.1.2 Controlador PI

Baseado nas figuras 28 e 29, iniciando pelas referências de +10 e -10 graus, ambas as soluções rumam para a estabilidade dentro do esperado teórico, e também oscilam menos, o seguimento de referência é satisfatório, e neste caso sem erro de posição devido ao integrador.

Porém ao determinar uma referência de +20 graus, esta é até seguida, porém o tempo de estabilização é violado cerca de 200 segundos, além do sistema não oscilar que é uma característica de sistemas com integrador. Isto já era esperado devido a saturação do sinal de controle, o gradiente máximo que pode ser obtido gira em torno de 28 graus, sendo que durante a resposta transitória teórica, o gradiente se aproxima de 40 graus, logo com o dimensionamento prático atual do sistema é inviável respeitar o tempo de estabilização.

O integrador também acumula erro de maneira desnecessária, pois na prática ele tenta corrigir um erro que no atual dimensionamento não é possível. A aplicação de técnicas de anti-windup auxiliariam em uma resposta mais satisfatória, a integração condicional pode ser um dos exemplos a ser aplicados.

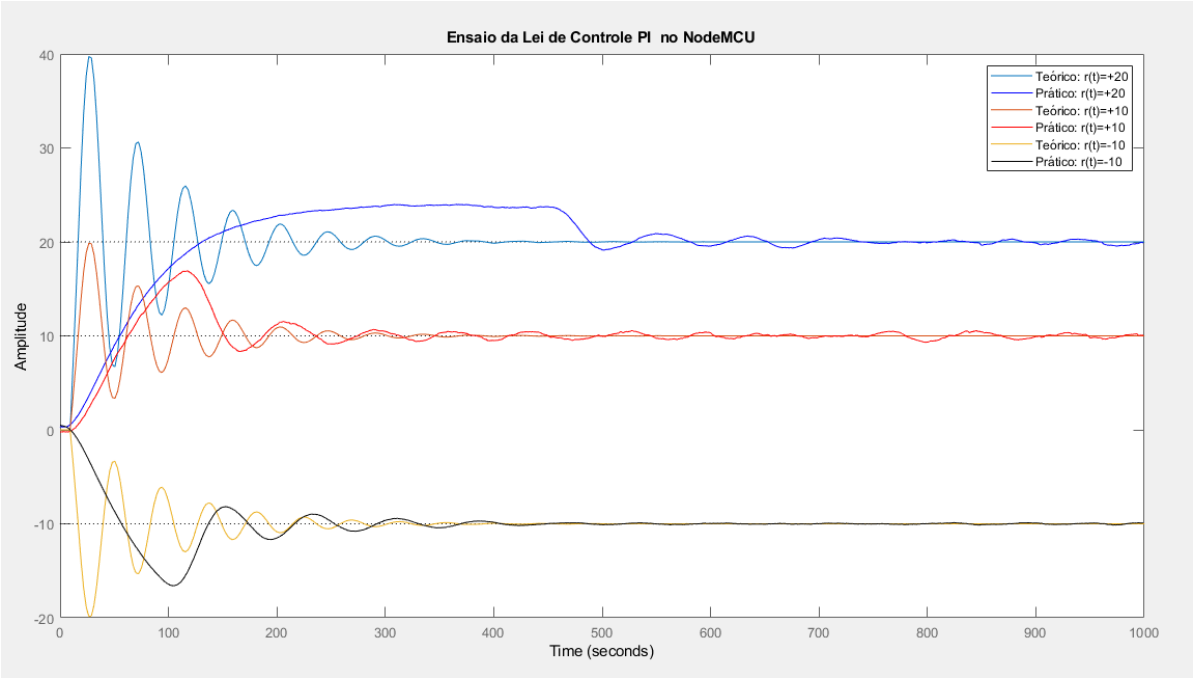

Figura 28 – Comparação da resposta teórica com a resposta prática do sistema com controlador PI no NodeMCU.

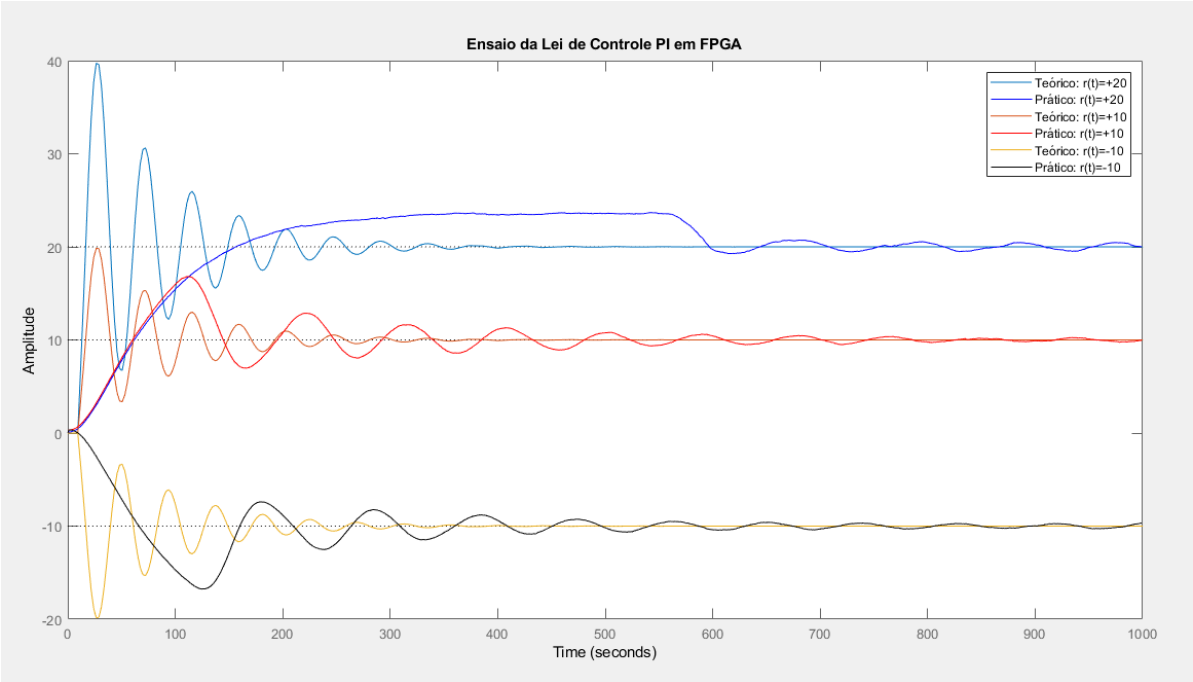

Figura 29 – Comparação da resposta teórica com a resposta prática do sistema com controlador PI no FPGA.

5.2 Nexys2 x NodeMCU

É necessário colocar a prova as duas soluções práticas, se houve ganho ao trocar um sistema de propósito geral por um sistema de propósito específico, e quais pontos podem ser melhorados para as soluções em FPGA se aproximem e até superem soluções convencionais.

5.2.1 Controlador P

Observando a figura 30, a solução utilizando a Nexys2 estabiliza com um pequeno atraso temporal, porém cerca de 10 segundos não é um atraso relevante ao ponto de apontar a superioridade da solução no NodeMCU dado a dinâmica lenta da planta.

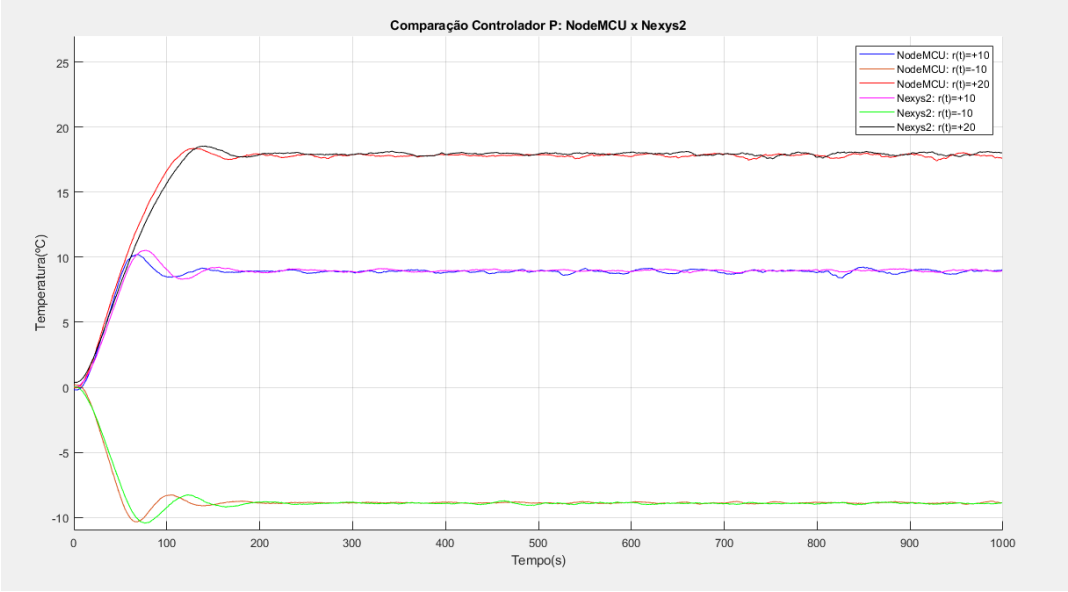

Figura 30 – Comparação da resposta no NodeMCU com a resposta na Nexys2 para o controlador P.

5.2.2 Controlador PI

Considerando a figura 31, a resposta obtida na Nexys2 é mais oscilatória, por mais que oscile em torno da referência, isso atrasa sua entrada em regime permanente. A resposta no NodeMCU passa apresentar oscilações de menos de 1 grau aos 200 segundos, já a Nexys2 após os 400 segundos, o que pode inviabilizar o uso da solução em FPGA caso o sistema necessite uma estabilização mais veloz.

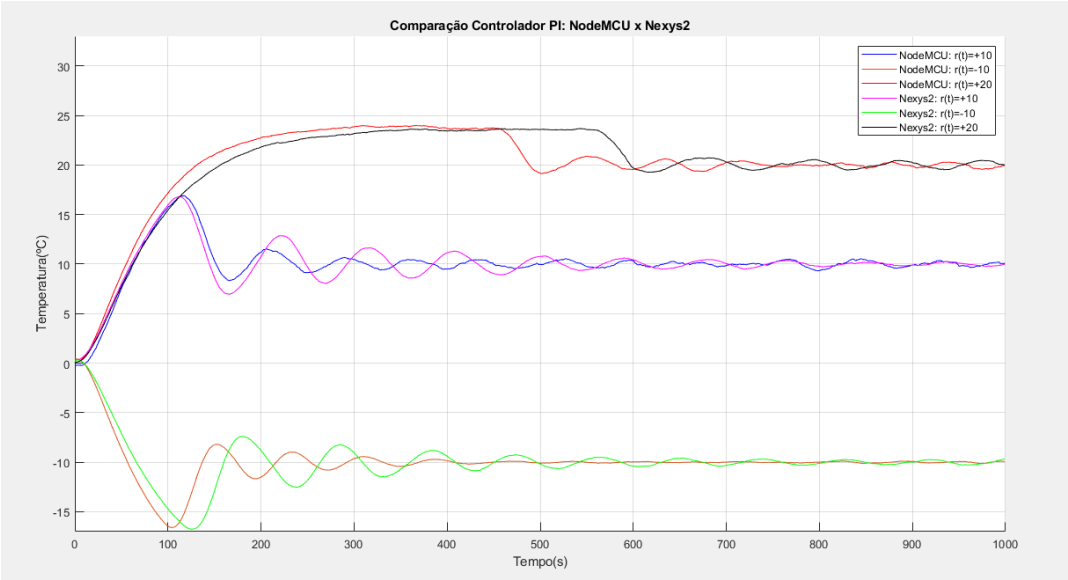

Figura 31 – Comparação da resposta no NodeMCU com a resposta na Nexys2 para o controlador PI.

5.3 Tempo e Erros de Cálculo

Inicialmente a diferença de algoritmos de multiplicação e divisão pode ser apontada como a causa do atraso na resposta, o algoritmo de ponto flutuante pode ser realizado em poucos ciclos, enquanto o de ponto fixo utilizado é serial, porém a solução no FPGA leva 101 mili segundos do inicio da leitura dos sensores de temperatura até a conclusão do cálculo da razão cíclica para o PWM, enquanto o NodeMCU fica em espera somente pela conversão dos sensores de temperatura durante 160 mili segundos, já que realiza as leituras em série.

A principal causa deste comportamento é o paralelismo, o fato de ambos sensores serem lidos ao mesmo tempo pela Nexys2, reduz o tempo de espera de conversão pela metade, levando o FPGA a superar o NodeMCU neste quesito, mesmo possuindo uma frequência menor e um algoritmo serial, também é necessário citar a comunicação em paralelo via serial, assim que o gradiente de temperatura é calculado, o bloco serial já inicia o envio dos dados e ao mesmo tempo outro bloco já inicia os cálculos da lei de controle, novamente acelerando o processamento de dados no FPGA.

Outro efeito do paralelismo é na leitura de sensores, que pode levar a uma diferença no gradiente de temperatura calculado no NodeMCU e na Nexys2, devido a ler os sensores em série, as leituras do NodeMCU possuem uma diferença no tempo e durante este tempo a atuação na planta continua, a leitura do segundo sensor sofreu atuação por mais tempo do que o primeiro e logo é diferente do que realmente deveria ser caso ambos fossem lidos no mesmo momento. Já na leitura dos sensores na Nexys2 o tempo entre leituras tende a zero,

pois devido ao paralelismo a leitura de ambos inicia ao mesmo tempo, o que em teoria leva a um gradiente de temperatura mais exato no tempo.

Porém a principal diferença entre os dois sistemas são suas aritméticas, para uma mesma quantidade de bits a representação de ponto flutuante do NodeMCU é mais precisa que a de ponto fixo utilizada na Nexys2, isso afeta todos os cálculos, desde a conversão do sensor até gerar o sinal de PWM, além das equações discretas serem uma aproximação das equações do controlador em tempo contínuo, havendo assim uma aproximação de representação numérica executando aproximações de operações matemáticas, onde a representação mais precisa acaba tendo vantagem. Um exemplo pode ser a leitura dos sensores na Nexys2, que mesmo sendo mais precisa no tempo, pode ser degradada pela falta de precisão na conversão dos valores e pode ser uma das causas da diferença nas respostas obtidas.

6 Conclusão

Este estudo apresentou o desenvolvimento de um sistema de controle desde sua fundamentação mais básica, passando por todo o arcabouço matemático utilizado, implementações práticas convencionais e seu grande foco a implementação em FPGA. Frisando as diferenças entre as implementações relacionados a algoritmos para execução de cálculos matemáticos e a mudança de paradigma que pode ser causada devido ao paralelismo proporcionado pelo FPGA. Discutindo como a saturação do sinal de controle afeta a resposta em ambas as soluções discutidas, e a diferença causada na resposta prática em relação a teórica.

Conclui-se que é possível realizar sistemas de controle por meio da utilização de FPGA's, com uma resposta satisfatória para um primeiro protótipo, mesmo com as limitações citadas. A descrição desenvolvida para o controlador cumpre sua função e pode ser utilizada em estudos futuros, visando aperfeiçoar a própria descrição, quanto a visão geral da implementação de sistemas de controle em FPGA, principalmente se valendo do paralelismo possibilitado por este.

Ficando como primeiro ponto de evolução, a troca da representação e da aritmética utilizada para ponto flutuante, além do aumento da precisão nos cálculos, aumentaria a integração com outros projetos já existentes, devido ser a representação padrão da IEEE, além dos dados oriundos do FPGA não necessitarem de conversão de ponto fixo para flutuante através de um programa auxiliar.

Referências

ASAIR. *AHT21B: Temperature and humidity sensor*. 2021. Disponível em: <http://www.aosong.com/en/products-62.html>. Citado na página 21.

HALLIDAY, D.; RESNICK, R.; WALKER, J. *Fundamentals of Physics, Part 2*. 8. ed. [S.l.]: Willey, 2008. Citado na página 21.

INFINEON. *BTS7960: High Current PN Half Bridge*. 2004. Disponível em: <https://www.infineon.com/dgdl/Infineon-BTS7960-DS-v01_01-en.pdf?fileId=db3a304412b407950112b43945006d5d>. Citado na página 21.

OGATA, K. *Modern Control Engineering*. 5. ed. [S.l.]: Person, 2009. Citado na página 21.

PELTIER, J. C. A. Nouvelles expériences sur la caloricité des courants électrique. *Annales de Chimie et de Physique*, n. 56, p. 371–386, 1834. Citado na página 21.

RABAEY, J.; CHANDRAKASAN, A.; NIKOLIC, B. *Digital Integrated Circuits: A Design Perspective*. 2. ed. [S.l.]: Pearson, 2002. Citado na página 21.

SEEBECK, T. J. Magnetische polarisation der metalle und erze durch temperatur-differenz. *Abhandlungen der Königlichen Akademie der Wissenschaften zu Berlin*, p. 265–373, 1822. Citado na página 21.

ZIEGLER, J. G.; NICHOLS, N. B. Optimum settings for automatic controllers. *ASME Digital Collection*, ASME, 1942. Citado na página 21.

ÅSTRöM, K. J.; HäGGLUND, T. *PID Controllers: Theory, Design, and Tunning*. 2. ed. [S.l.]: ISA - The Instrumentation, Systems and Automation Society, 1995. Citado na página 21.