UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

DEPARTAMENTO ACADÊMICO DE ELETRÔNICA

BACHARELADO EM ENGENHARIA ELETRÔNICA

CAIKE RODRIGO ALBERTIN

IMPLEMENTAÇÃO DE REDES NEURAIS ARTIFICIAIS UTILIZANDO FPGA PARA APLICAÇÕES DE CONTROLE INTELIGENTE EM ROBÓTICA

TRABALHO DE CONCLUSÃO DE CURSO

CAMPO MOURÃO – PR

2016

CAIKE RODRIGO ALBERTIN

IMPLEMENTAÇÃO DE REDES NEURAIS ARTIFICIAIS UTILIZANDO FPGA PARA APLICAÇÕES DE CONTROLE INTELIGENTE EM ROBÓTICA

Trabalho de Conclusão de Curso de graduação do curso de bacharelado em Engenharia eletrônica do Departamento Acadêmico de Eletrônica(DAELN) da Universidade Tecnológica Federal do Paraná, como requisito final para obtenção do título de Engenheiro em Eletrônica.

Orientador: Prof. Dr. Márcio Rodrigues da Cunha

CAMPO MOURÃO

2016

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Campus Campo Mourão
Coordenação do Curso de Engenharia Eletrônica

TERMO DE APROVAÇÃO DO TRABALHO DE CONCLUSÃO DE CURSO INTITULADO

Implementação de redes neurais artificiais utilizando FPGA para aplicações de controle inteligente em robótica

por

Caike Rodrigo Albertin

Trabalho de Conclusão de Curso apresentado no dia 23 de Junho de 2016 ao Curso Superior de Engenharia Eletrônica da Universidade Tecnológica Federal do Paraná, Campus Campo Mourão. O Candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

_____

Prof. Heber Miguel dos Santos

(UTFPR)

_____

Prof. Roberto Ribeiro Neli

(UTFPR)

_____

Prof. Marcio Rodrigues da Cunha

(UTFPR)

Orientador

A Folha de Aprovação assinada encontra-se na Coordenação do Curso

RESUMO

ALBERTIN, Caike Rodrigo. **Implementação de redes neurais artificiais utilizando fpga para aplicações de controle inteligente em robótica**. 2016. 72 f. Trabalho de Conclusão de Curso – Bacharelado em Engenharia Eletrônica. Universidade Tecnológica Federal do Paraná. Campo Mourão, 2016.

O trabalho a seguir trata da elaboração de uma rede neural artificial (RNA) *feedforward* de múltiplas camadas, descrita em linguagem de *hardware VHDL* para a aplicação em robótica móvel*,* desde sua elaboração como um sistema de blocos, até os ajustes finos como a inicialização dos pesos ideais para o problema. Foi utilizado *software* auxiliar matemático para os ajustes necessários, na base de dados, pesos iniciais e taxa de aprendizado. Deixado para trabalhos futuros o algoritmo de treinamento descrito em *VHDL,* interface entre usuário/*hardware* e o projeto do sistema como um todo.

ABSTRACT

ALBERTIN, Caike Rodrigo. **Implementation of artificial neural networks using fpga for intelligent control applications in robotic**. 2016. 72 f. Trabalho de Conclusão de Curso – Bacharelado em Engenharia Eletrônica. Universidade Tecnológica Federal do Paraná. Campo Mourão, 2016.

In the following work project deals with the elaboration of a Artificial neural network, Multi-layers feedforward, described in hardware language VHDL for the application in mobile robotics, since this elaboration as a block system, until the fine adjustments like the initialization of the ideal weights for the problem. Mathematical auxiliary software was used for the necessary adjustments in the database, initial weights and learning rate. The training algorithm described in VHDL, the user/hardware interface and the overall system design is left for future work.

Lista de Figuras

Figura 1 – Crescimento de Vendas de Robôs 7
Figura 2 – Arquitetura Cyclone IV 10
Figura 3 – Modelo de McCulloch e Pitts 14
Figura 4 – Rede feedforward de uma camada 15
Figura 5 – Rede feedforward de duas camadas 15
Figura 6 – Rede recorrente sem neurônios ocultos 16
Figura 7 – Rede recorrente com camadas ocultas 17
Figura 8 – Porta limiar quadrática 19
Figura 9 - Roda fixa 21
Figura 10 - Roda orientável centralizada 22
Figura 11 - Roda orientável não centralizada 22
Figura 12 - Disposição das rodas 23
Figura 13 - Sistema de Controle 26
Figura 14 - Diagrama de montagem do ADC 27
Figura 15 - Bloco de Regulagem 28
Figura 16 - Bloco de saída 28
Figura 17 - Bloco da RNA 29
Figura 18 - Bloco de blocos da RNA 29
Figura 19 - Bloco de Controle 30
Figura 20 - Bloco de Neurônio da primeira camada 31
Figura 21 - Bloco de neurônio da camada oculta 31
Figura 22 - Bloco final do sistema 34
Figura 23 - Diagrama de comando do servo 34
Figura 24 - Compilation Report 37
Figura 25 - Simulação para o sistema com 50 amostras 38
Figura 26 - Simulação para o sistema de 50 amostras 39
Figura 27 - Compilation Report do sistema todo 40
Figura 28 - Resposta do banco de dados utilizando o nntools 41
Figura 29 - Erro Quadrático Médio para uma nova função de ativação 42
Figura 30 - Erro quadrático médio para uma nova base de dados 43
Figura 31 - Saída da RNA com os pesos treinados em Matlab 46
Figura 32 - Utilização do Switch de menu 47
Figura 33 - Utilização do Switch de menu 48
Figura 34 - Utilização da chave seletora de entrada desejada 48
Figura 35 - Utilização da chave seletora de peso 49
Figura 36 - Utilização da chave seletora de peso 49
Figura 37 - Utilização da chave seletora de iteração 50

Lista de Tabelas

Tabela 1 – Sensores para robôs móveis 24
Tabela 2 – Atuadores para robôs móveis 24
Tabela 3 - Pesos Iniciais do sistema utilizado 44
Tabela 4 - Pesos antes e depois do treinamento 45

LISTA DE SIGLAS E ABREVIAÇÕES

ADC	*Analogic Digital Converter*
ASIC	*Application specific integrated circuit*
CLB	*Configurable Logic Blocks*
DAC	*Digital Analogic Converter*
DSP	*Digital signal program*
FPGA	*Field programmable array*
IA	Inteligência artificial
IAC	Inteligência artificial conexionista
IAS	Inteligência artificial simbólica
IEEE	*Institute of Electrical and Electronics Engineers*
I/O	*Input/Output*
LDR	*Light Diode Resistor*
PLL	*Phase locked loop*
RAM	*Random acess memory*
RNA	Rede neural artificial
ROM	*Read only memory*
SRAM	*Static random acess memory*
VHDL	*VHSIC hardware description language*
VHSIC	*Very high speed integrated circuit*

Sumário

1. INTRODUÇÃO 7
2. OBJETIVO 9
3. ESTADO DA ARTE 10
   3.1. FPGA 10
      3.1.1. FPGA's e Microcontroladores 10
   3.2. VHDL 11
      3.2.1. Vantagens 11
      3.2.2. Conceitos básicos 12
      3.2.3. Ponto Flutuante 12
   3.3. RNA's 12
      3.3.1. Modelo de neurônio 13
      3.3.2. Topologia de Redes Neurais 14
      3.3.3. Aprendizado 17
      3.3.4. Redes Perceptron de camada única 18
   3.4. Robôs móveis 20
      3.4.1. Morfologia 20
      3.4.2. Sensores e atuadores 23
      3.4.3. Controle Inteligente 25
4. METODOLOGIA 26
5. RESULTADOS OBTIDOS 36
6. IMPLEMENTAÇÃO EM HARDWARE 47
7. CONCLUSÃO 51
8. REFERÊNCIAS 52
9. APÊNDICE 54

1. INTRODUÇÃO

O desenvolvimento dos sistemas de robótica móvel, seja no Brasil ou no exterior, tem crescido consideravelmente. Para controlar os atuadores o controle inteligente tem sido amplamente utilizado, técnicas como controle fuzzy e redes neurais artificiais.

Entre os fatores que impulsionam a modernização da robótica pode-se citar a competitividade crescente, a rápida alteração dos produtos oferecidos ao mercado e o avanço tecnológico, entre outros, que visam aumentar a produtividade, a qualidade e a confiabilidade dos produtos.

Além da área de manufatura e transporte de materiais, outras aplicações destes sistemas incluem o trabalho em ambientes perigosos ou insalubres e a exploração espacial (1).

Dados da Federação Internacional de Robótica, mostram que o número de máquinas vendidas em 2014 foi recorde, atingindo a marca de 229261 unidades o que corresponde a um crescimento de 29% em relação a 2013. A evolução das vendas pode ser visualizada na figura 1.

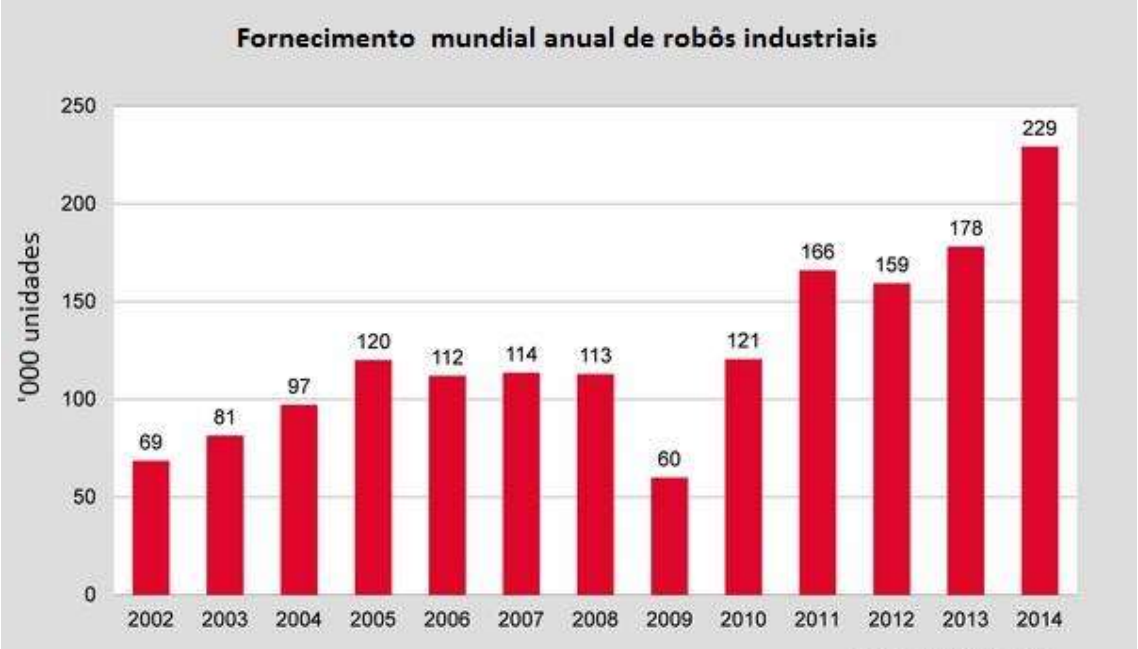


Figura 1 – Evolução de Vendas de Robôs

Fonte: Adaptada de International Federation of Robotics.

Em particular, Redes Neurais Artificiais (RNA) é um modelo de abordar solução de problemas da Inteligência Artificial (IA). Neste caso, ao invés de tentar imitar um comportamento inteligente procura-se construir um software com um comportamento inteligente, aprendendo novas tarefas, errando, fazendo generalizações e descobertas. Desta forma estes softwares podem se auto organizar aos estímulos externos afim de possuírem um comportamento que não se pode prever (2).

Projetos de robótica móvel inteligente envolve um projeto de *hardware* e *software*. O *hardware* é responsável pela implementação física de sensores, atuadores e do processamento de dados através de um processador embarcado. O *software* fica responsável pelo sistema de controle robótico, pela análise de sinais recebidos pelos sensores, planejamento e tomada de decisões (3).

Grande parte do desenvolvimento de controle inteligente vem de aplicações que exigem grande paralelismo, por isso a utilização de descrição de hardware, utilizando linguagens como *VHDL* para *FPGAs* e *ASICs* têm crescido nessa área, pois permite o projeto concorrente de *hardware* e de *software* com um único circuito integrado.

Por sua vez, os *FPGAs* são circuitos integrados que contém um grande número de unidades lógicas idênticas. Neste aspecto as unidades lógicas podem ser vistas como componentes padronizados que podem ser configurados e interconectados a partir de uma linguagem de descrição de hardware, como *ADHL*, *SystemC*, *Verilog* ou *VHDL*. Um arquivo binário é gerado para a configuração da *FPGA*. Esse arquivo contém as informações necessárias para especificar a função de cada unidade lógica e para chavear seletivamente os *switches* da matriz de interconexão (4).

Neste contexto, o escopo deste trabalho é aproveitar as vantagens da descrição de hardware, usando *VHDL* e *FPGA*, para implementação de RNA's aplicáveis a robótica móvel.

2. OBJETIVO

O objetivo deste trabalho é o desenvolvimento de uma rede neural artificial, utilizando a linguagem de descrição de hardware *VHDL* e o circuito integrado *FPGA*, aplicável ao controle inteligente de um robô móvel a partir da leitura de sensores.

3. ESTADO DA ARTE

3.1. FPGA

Os *FPGA's* foram lançados pela Xilinx por volta de 1980. Sua arquitetura é formada por matrizes de blocos lógicos e não possuem planos de portas *OR* ou *AND*, em vez disso, possuem células reconfiguráveis que podem implementar funções lógicas (4).

Blocos lógicos programáveis são denominados *CLBs*, que são circuitos construídos pela união de *flip-flops* e a utilização de lógica combinacional. Além dos indispensáveis *CLBs*, também podem ser incorporados aos *FPGAs* outros blocos, úteis para projetos grandes e complexos, tais como blocos *SRAM*, blocos *DSP* e circuitos *PLL* (4). A figura 2 ilustra a arquitetura que foi utilizada.

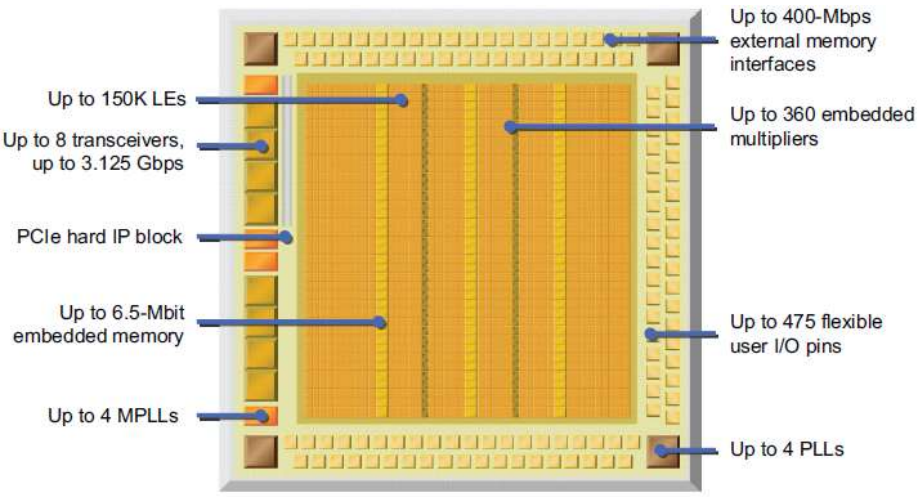


Figura 2 – Arquitetura Cyclone IV

Fonte: (ALTERA, 2014)

3.1.1. *FPGA's* e Microcontroladores

Em sistemas de controle ou sistemas embarcados, diversos tipos de dispositivos de processamento são utilizados, tais como processadores de propósito geral, processadores de propósito especifico e Circuitos Integrados para Aplicações Específicas.

Geralmente surgem dúvidas sobre como escolher o dispositivo de processamento e quais suas principais características. Para obter uma visão mais ampla do assunto, algumas vantagens e desvantagens, particularmente de *FPGA* e Microcontroladores, são apresentadas a seguir:

FPGA's são amplamente utilizados na indústria, geralmente escolhidos onde é necessário processar uma grande quantidade de dados, onde tempo real e paralelismo são parâmetros cruciais. Por se tratar de programação de *hardware*, é possível realizar uma instrução por ciclo de *clock*. Comumente utilizados em processamento digital de sinais, *switches*, roteadores de alto desempenho, setor automotivo, dentre outros. Embora o custo seja um pouco elevado, foi lançado em 2014 a família ICE40 da Lattice que por possuírem uma quantidade baixa de *I/Os* e elementos lógicos, possuem um preço acessível e podem substituir microcontroladores nesta faixa de preço (6).

Microcontroladores por sua vez são controlados por software, sendo assim, torna mais lento seu processamento, porém, seu baixo custo o torna viável para utilização em aplicações às quais não é realmente necessária uma rapidez de processamento.

3.2. VHDL

Em 1980, o departamento de defesa dos Estados Unidos pretendia desenvolver um circuito unificado e que pudesse ser reutilizável. Ficou claro a necessidade de uma linguagem de programação padronizada, para descrever a estrutura de circuitos digitais para circuitos integrados. Criaram então o *VHDL* (4).

Em 1987, ocorreram dois fatos marcantes, o departamento de defesa dos estados unidos designou que todos os circuitos digitais fossem descritos em *VHDL*, e o *IEEE* ratificou o *VHDL* como *IEEE* padrão 1076 (4).

Em 1993, a linguagem foi revisada, definindo o padrão 1076'93. Em 1996, as ferramentas de simulação foram incorporadas pelo padrão *IEEE*. No padrão *IEEE* 1076.4 foram agregados modelos *ASIC* e bibliotecas para *FPGA* em *VHDL*. Em dezembro de 1997, foi publicado o manual de referência da linguagem (4).

3.2.1. Vantagens

As principais vantagens em utilizar o *VHDL* é que permite uma maior integração e consequentemente uma maior complexidade de circuitos. Além do maior nível de abstração, vale ressaltar a facilidade na atualização de projetos e uma redução no custo/tempo de projeto (4).

3.2.2. Conceitos básicos

A entidade é composta pela claúsula *port* e *generic*. A seção *port* é utilizada para listar as portas do sistema, juntamente com seus modos, *in* (entrada), *out* (saída), *inout* (entrada e saída) ou *buffer* (buffer). A seção *generic* não é obrigatória, sendo utilizada quando se quer declarar uma constante (4).

A arquitetura é responsável pela descrição do hardware de forma estrutural ou comportamental, pode possuir códigos sequenciais, que têm de ser inseridos dentro de processos (*process)* ou subprogramas (*function* ou *procedure)*.

O projeto hierárquico é uma forma de tratar a complexidade de circuitos grandes com a divisão do circuito em blocos ou módulos menores que são projetados separadamente e depois interligados para compor o sistema completo. Cada bloco em um projeto hierárquico é um subprojeto que também possui um diagrama de captura esquemática. Esta hierarquia pode conter vários níveis, até que a complexidade de um bloco seja adequada para um projeto tradicional usando componentes digitais básicos (4).

3.2.3. Ponto Flutuante

Em arquitetura digital, os números são representados usando sequencias finitas que levam a erros de quantização e erros de aproximação em operações aritméticas. Desta forma, tanto para representar números grandes, quanto pequenos, necessita-se de uma grande quantidade de bits. Uma solução do problema é a utilização da aritmética do ponto flutuante (5).

Na robótica, atualmente, há uma alta demanda de aplicações que operam a altas velocidades, procurando um melhor desempenho. Logo, a utilização de *FPGAs* permitiria aproveitar o paralelismo intrínseco, porém a falta de uma biblioteca completa de ponto flutuante acaba limitando o uso de *FPGAs* em aplicações que requerem alta precisão.

3.3. RNA’s

Em inteligência artificial existem dois grandes paradigmas. O primeiro é a Inteligência Artificial Superficial (IAS) que o comportamento inteligente é simulado sem considerar mecanismos responsáveis por esse comportamento. E o segundo paradigma é a Inteligência Artificial Conexionista (IAC) que se

acredita que construindo uma máquina, que imite as características do cérebro, apresentará inteligência.

As duas correntes de estudos de IA se separaram e os avanços com inteligência artificial conexionista, em particular redes neurais artificiais evoluíram lentamente, enquanto a manipulação da inteligência artificial simbólica se acelerou.

Em 1943, Warren e McCulloch Pitts conseguiram representar o primeiro neurônio artificial utilizando ferramentas matemáticas, sem se preocupar em técnicas de aprendizado. Em 1949, Donald Hebb apresentou sua teoria de aprendizado baseado na variação dos pesos de entrada, como uma forma de se atingir o objetivo proposto.

Em 1951, minsky construiu o primeiro neurocomputador com capacidade de aprendizado, onde os pesos entre as sinapses eram ajustados automaticamente. Em 1958, Frank Rosenblatt criou o modelo de *Perceptron*, incrementando os pesos entre as sinapses. Em 1969 Minsky e Papert constataram que o *perceptron* não conseguia analisar problemas não linearmente separáveis.

Após um recesso, em 1982, John Hopfield publicou um artigo que abordava as propriedades associativas das RNAs, assim estimulou novamente os interesses de pesquisadores. Em 1986, surgiu o algoritmo de *backpropagation*, aplicado a redes de múltiplas camadas, criado por Rumelhart, Hilton e Williams (7).

3.3.1. Modelo de neurônio

O modelo de neurônio artificial proposto por McCulloch e Pitts é uma simplificação do conhecimento sobre neurônio biológico. Em sua formação existem n terminais de entradas, que simulam os dendritos e apenas um terminal de saída, representando o axônio.

Para simular o comportamento das sinapses, suas entradas têm pesos acoplados, onde a soma ponderada da multiplicação das entradas com seus respectivos pesos (7), gera uma saída limitada pela função de ativação, que tipicamente o intervalo da amplitude de saída é escrito como um intervalo unitário fechado [0;1] (8).

A figura 3 representa o modelo básico de neurônio utilizado, criado por McCulloch e Pitts.

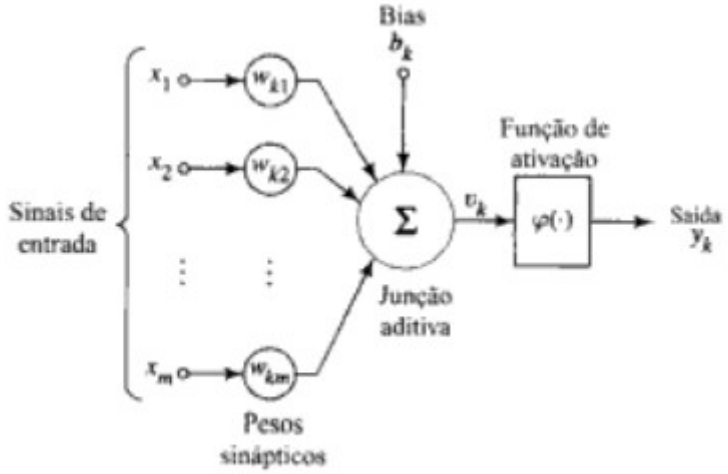


Figura 3 – Modelo de McCulloch e Pitts

Fonte : (HAYKINS, 2001)

O equacionamento deste modelo é dado por:

$$u_k = \sum_{i=1}^{n} w_i x_i \qquad (1.1)$$

Onde, w_i representa o peso referente àquela iteração, x_i é a entrada referente àquela iteração e u_k é a saída do neurônio.

Para simplificação se deseja uma saída binária, então se adota a função de ativação limiar, onde sua saída é binária e dada por:

$$F(u) = \begin{cases} 1 & u_k \geq \theta \\ 0 & u_k < \theta \end{cases} \qquad (1.2)$$

Onde, θ é o limiar da função de ativação.

3.3.2. Topologia de Redes Neurais

Independente da função de ativação definida, neurônios individuais possuem capacidade computacional limitada, por isso formam-se redes de neurônios.

A figura 4 é a topologia mais simples de uma RNA, que corresponde a uma rede neural de única camada, alimentada para frente. São capazes de resolver problemas multivariáveis, mas com algumas restrições de complexidade. É considerada estática, pois sua saída depende apenas de sua entrada (7).

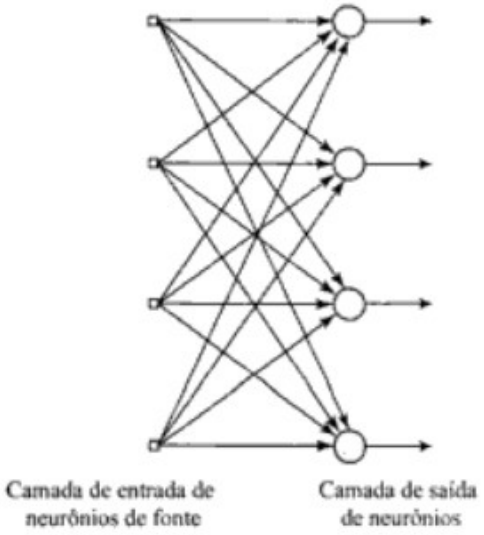


Figura 4 – Rede *feedforward* de uma camada

Fonte: (Haykins, 2001)

A RNA apresentada na figura 5 apresenta as redes *feedforwards* de múltiplas camadas. As camadas de neurônios ocultos acrescentam capacidade computacional e universalidade na aproximação de funções continuas (8). Assim como a anterior, também são estáticas (7).

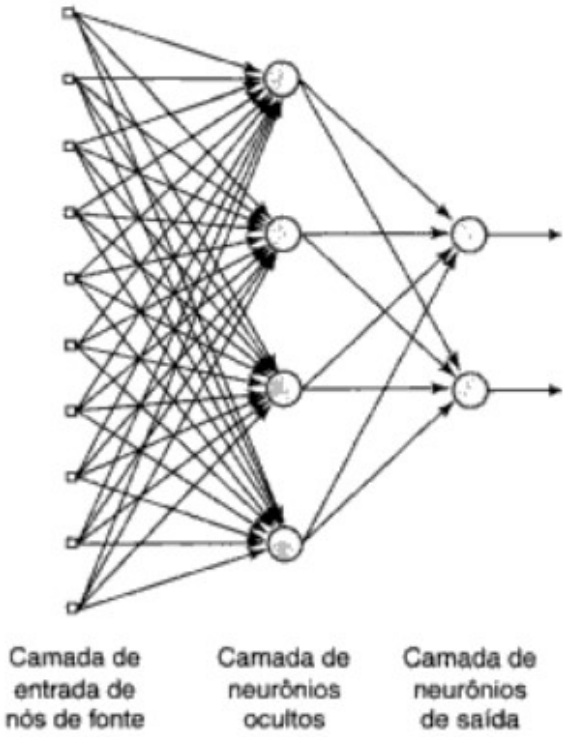


Figura 5 – Rede *feedforward* de duas camadas

Fonte: (Haykins, 2001)

As próximas topologias são consideradas redes recorrentes, por possuírem ao menos uma realimentação. São utilizadas para problemas não lineares, essas realimentações possuem grande impacto em seu aprendizado e desempenho.

A RNA apresentada na figura 6 é a topologia realimentada mais simples, que possui apenas uma camada, implicando uma limitação de complexidade.

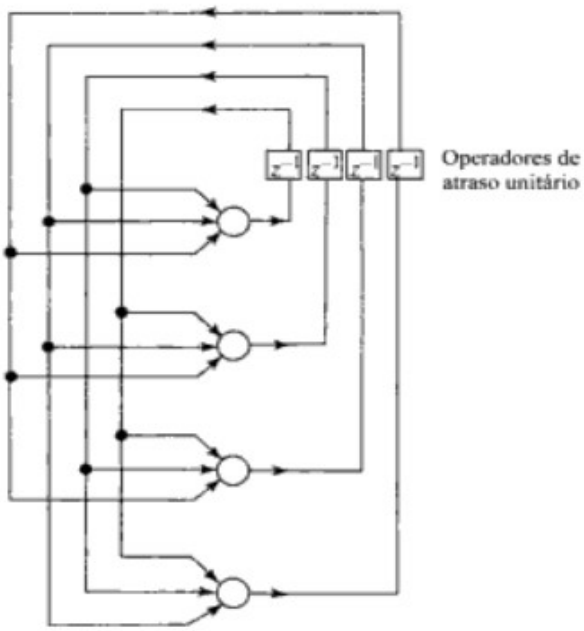


Figura 6 – Rede recorrente sem neurônios ocultos

Fonte: (Haykins, 2001)

Na rede apresentada na figura 7, observa-se camadas ocultas, quando suas saídas não dependem apenas de suas entradas, e sim da saída de outros neurônios. Nesse caso a saída dos neurônios ocultos tem impacto no treinamento dos neurônios das camadas de entrada (8). São muito utilizadas na resolução de problemas que envolvam processamento temporal, como previsão de eventos futuros (7).

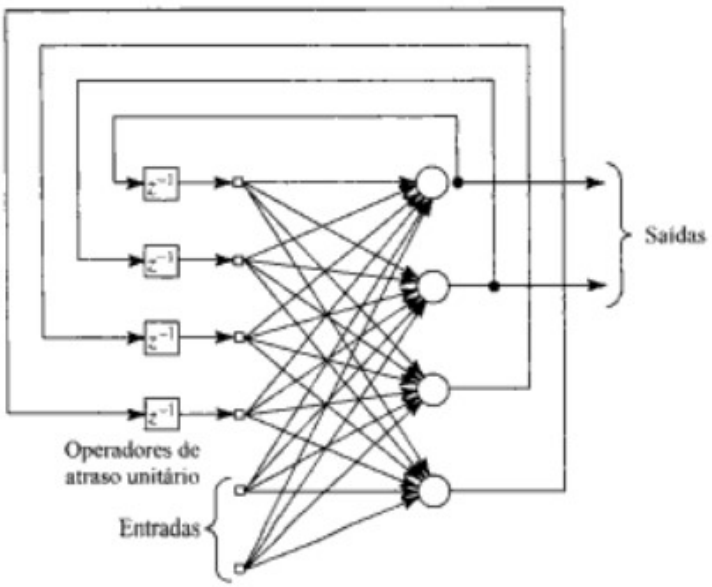


Figura 7 – Rede recorrente com camadas ocultas

Fonte: (Haykins, 2001)

3.3.3. Aprendizado

A principal característica das RNAs é sua capacidade de aprender por meio de exemplos. Como a abordagem é conexionista, o aprendizado não é imposto por meio de regras explícitas, como na IAS, mas sim através do ajuste das intensidades dos neurônios. Consiste em um processo iterativo de parâmetros da rede, e ao final desse processo guardam o conhecimento que a rede adquiriu do ambiente externo (7).

Esse conhecimento do mundo externo consiste em dois tipos de informação, o estado conhecido do mundo externo, neste caso é uma informação prévia dada pelo treinador da rede. E as observações do mundo que são obtidas por meio de sensores projetados para sondar o ambiente no qual a rede neural deve operar. Deve-se atentar, pois estas observações podem ser ruidosas (8).

Vale ressaltar que o aprendizado está relacionado com a melhoria do desempenho da rede segundo algum critério preestabelecido.

Existem vários algoritmos de treinamento de RNA’s, agrupados em dois paradigmas principais: aprendizado supervisionado e aprendizado não-supervisionado. No aprendizado supervisionado é necessário um professor externo, responsável por estimular as entradas da rede por meio de padrões de entrada e observar a saída calculada pela mesma, comparando-a com a saída

desejada. No aprendizado não-supervisionado esse professor externo não existe, a saída da rede fica a cargo de agrupamento e reconhecimento de padrões semelhantes dentre as amostras. (7)

No aprendizado supervisionado, a saída da RNA é uma função que depende dos valores atuais do seu conjunto de pesos, então, tais pesos são reajustados afim da rede neural se aproximar da saída desejada. No aprendizado supervisionado a rede tem sua saída corrente comparada com a saída desejada pelo professor, que por sua vez fornece informações sobre a direção de ajuste dos pesos. A minimização do erro é incremental, já que pequenos ajustes são feitos nos pesos a cada etapa de treinamento, de tal forma que a cada etapa de treinamento caminhe para uma convergência (7).

O exemplo típico é o aprendizado por correção de erros. Onde se procura diminuir o erro da resposta da rede em relação ao do professor, representado pela equação (1.3).

$$e(t) = y_d(t) - y_o(t) \tag{1.3}$$

A equação (1.3) mostra o erro em função do tempo e_t, onde parâmetro $y_d(t)$ é o resultado desejado e $y_o(t)$ é o resultado obtido.

Para obter um ajuste no resultado obtido altera-se o peso das entradas, através equação (1.4):

$$w_i(t+1) = w_i(t) + \eta e(t) x_i(t) \tag{1.4}$$

O erro calculado na equação (1.3) é utilizado na equação (1.4). Também é utilizado o peso anterior e η que é a taxa de aprendizado, para obter o novo valor do peso.

Estas equações são utilizadas tanto no algoritmo de treinamento do *perceptron* quanto no algoritmo de treinamentos de redes Adaline.

Por outro lado, no aprendizado não supervisionado, somente os padrões de entrada estão disponíveis para a rede. Durante o processo de aprendizado, padrões de entrada são apresentados continuamente à rede, e a existência de regularidades nesses dados faz com que o aprendizado seja possível. Desta forma, regularidade e redundância nas entradas são características essenciais para haver aprendizado não supervisionado (7). Se aplica a problemas que visam à descoberta de características estatisticamente relevantes nos dados de entrada, como a descoberta de agrupamentos e classes.

3.3.4. Redes *Perceptron* de camada única

As redes perceptron representam o modelo mais simples de uma rede neural usada para classificações de padrões ditos linearmente separáveis. Basicamente consiste de um único neurônio com pesos sinápticos ajustáveis e

função de ativação do mesmo. O algoritmo para ajustar os parâmetros livres desta rede, originou de um procedimento de aprendizado criado por Rosenblatt. Provando que os padrões usados para treinar o *perceptron* são retirados de duas classes linearmente separáveis, então o algoritmo converge(8).

Portas do tipo limiar podem ser divididas em três tipos: linear, quadrática e polinomial. Suas funções são basicamente a mesma, comparação da soma ponderada das entradas com um valor limiar. Caso a soma exceda o limiar, a saída é ativada (7).

O equacionamento da porta limiar linear é dado acima pela equação (1.2 restrita apenas à solução de problemas linearmente separáveis. As entradas são divididas em vetores que levam a saída γ para 1, e outras que levam γ para 0 (7).

Apesar da limitação à resolução de problemas linearmente separáveis, as funções de ativações limiares correspondem a uma parcela de 2^{2^n} funções booleanas possíveis, com n sendo o número de entradas. As portas limiares são mais poderosas que as portas lógicas convencionais. Com uma porta limiar pode-se implementar qualquer uma das funções booleanas *and, or, nand, nor,* entre outras, alterando apenas os parâmetros dos pesos. Com tanta flexibilidade, para a implementação de funções não linearmente separáveis como a porta *xor,* são necessárias pelo menos duas camadas de portas limiares. (7)

Além das portas limiares lineares outra bastante utilizada é a porta limiar quadrática, pois para grandes valores de n, a relação entre os números de funções linearmente separáveis e o número total de funções booleanas tende a zero, restringindo a utilização de portas lineares. Sua função de saída é dada da seguinte forma.

$$\gamma = \begin{cases} 1 & \sum w_i x_i + \sum w_{ij} x_i x_j > \theta \\ 0 & \sum w_i x_i + \sum w_{ij} x_i x_j < \theta \end{cases} \tag{1.5}$$

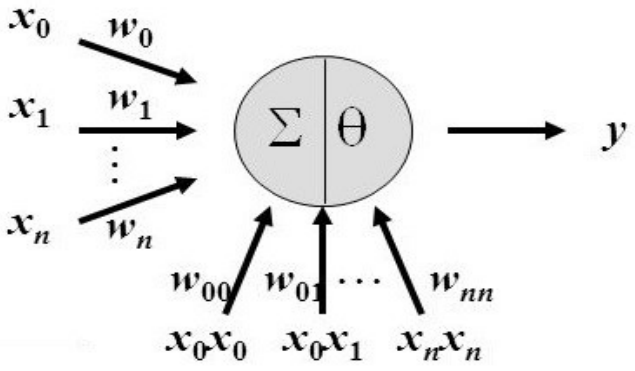


Figura 8 – Porta limiar quadrática

Fonte:(Braga, 2000)

O termo adicional na equação (1.5) é o que confere à porta limiar quadrática maior poder computacional, já que possui mais parâmetros livres ajustáveis. A equação da superfície de decisão pode assumir uma forma não-linear, devido aos termos produtos adicionais (7).

Uma RNA é composta por um conjunto de neurônios com capacidade de processamento local, uma topologia de conexão e uma regra de aprendizado. Após a apresentação dos modelos de neurônio e suas possíveis conexões, o algoritmo de aprendizado chamado correção de erros, será discutido neste trabalho. De forma geral, se deseja obter durante o processo de aprendizado no instante futuro $w(n+1)$ uma saída esteja mais próxima da solução do que no instante atual $w(n)$.

A implementação do algoritmo de treinamento do *perceptron* constitui-se em 4 passos:

1. Inicialize n.
2. Inicialize o vetor de pesos w com valores aleatórios.
3. Aplique a regra de atualização dos pesos, dadas pelas equações (1.3) e (1.4), para todos os pares(x^i, γ_d^i) do conjunto de treinamento $\Gamma = \{(x^i, \gamma_d^i)\}_{i=1}^{p}$
4. Repita o passo anterior até que o erro seja nulo, ou um valor considerado aceitável, para todos os elementos de Γ.

Algumas considerações sobre este algoritmo são importantes. O valor de n inicializado pelo usuário nem sempre converge em um tempo finito, selecionando um n muito pequeno pode levar a um tempo de convergência muito grande, e valores muito grandes n pode levar a instabilidades no treinamento. Na inicialização do vetor de pesos w, uma regra geral é inicia-los com valores amostrados em distribuição uniforme com valores pequenos, próximos a zero. Essa recomendação faz-se necessária para evitar a saturação do neurônio, o que traria dificuldades para a convergência do algoritmo.

3.4. Robôs móveis

3.4.1. Morfologia

Em geral, os robôs móveis distribuem seus sistemas de tração e direção sobre os eixos de suas rodas de acordo com as exigências de velocidade, manobrabilidade e características do terreno. A precisão e rapidez com que o robô deve alcançar seu destino fazem com que determinado sistema de tração e de direção seja adotado. A confiabilidade e a manobrabilidade que deve ter um robô móvel determinam as características do sistema de tração e direção, não

apenas em relação à técnica, mas também em relação ao número, ao tipo e à disposição das rodas necessárias para constituir uma estrutura mecânica estável (9).

O meio em que o robô está incluso pode se denominar estruturado ou não estruturado. É estruturado quando possui apenas obstáculos estáticos, ou seja, que com o passar do tempo não mudam de forma e posição. É não estruturado quando o entorno é dinâmico (9).

Possuem-se três tipos de roda convencionais: roda fixa, roda orientável centralizada e roda orientável não centralizada.

A roda fixa está associada a tração do robô, pois seu eixo está fixado na estrutura dele. A figura 9 ilustra a morfologia de roda fixa em dois ângulos de visão, tanto superior quanto lateral.

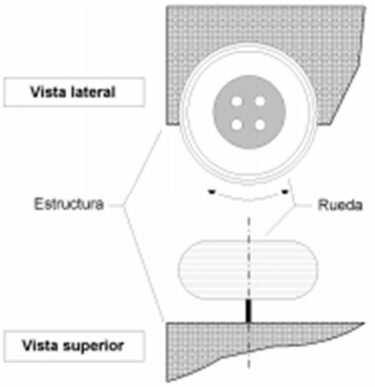


Figura 9 - Roda fixa

Fonte : (Secchi, 2008).

Roda orientável centralizada é aquela em que o movimento do plano da roda com respeito à estrutura é uma rotação ao redor de um eixo vertical que passa através do centro da roda (9). A figura 10 mostra uma visão lateral dessa morfologia e uma visão superior do eixo da roda orientável.

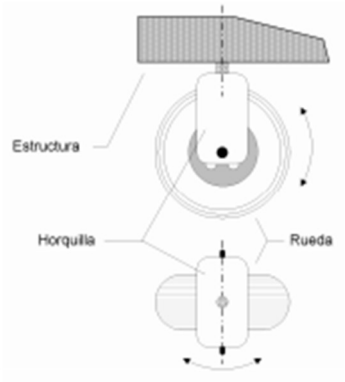


Figura 10 - Roda orientável centralizada

Fonte: (Secchi, 2008).

Roda orientável não centralizada é uma roda orientável com relação à estrutura tal que a rotação do plano da roda ocorre ao redor de um eixo vertical que não passa através do centro da roda, sua principal função é dar estabilidade à estrutura mecânica do robô como roda de direção (9). A figura 11 ilustra essa morfologia com visão lateral e superior.

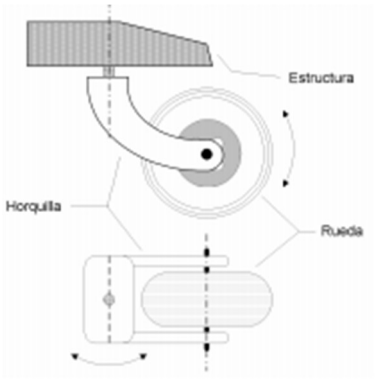


Figura 11 - Roda orientável não centralizada

Fonte: (Secchi, 2008).

Além do tipo de roda, a orientação delas também é de muita importância, pois é isto que diferencia o grau de manobrabilidade. Existem diversas formas de disponibilizar as rodas, mas será tratado apenas o modelo que se pretende trabalhar neste projeto, chamado uni ciclo. Este modelo é usado por pesquisadores por possuir uma cinemática simples, que consiste em duas rodas fixas independentes sobre um mesmo eixo e uma roda orientável não centralizada para dar estabilidade, conforme mostrado na figura 12.

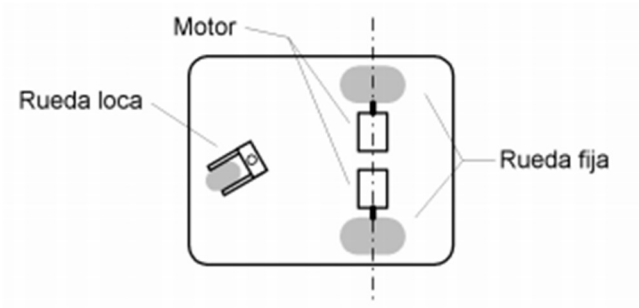


Figura 12 - Disposição das rodas

Fonte: (Secchi, 2008).

3.4.2. Sensores e atuadores

Um sensor pode ser definido como um transdutor que altera sua característica física interna devido a um fenômeno físico externo, como por exemplo presença de luz, som, gás, campo elétrico, campo magnético, etc. Um sensor muda seu comportamento sob a ação de uma grandeza física, pode fornecer direta ou indiretamente essa grandeza, convertendo em um sinal elétrico (5).

Do ponto de vista da robótica, um robô que não possui sensores é simplesmente uma máquina, isto é um robô precisa de sensores para deixar de ser uma máquina qualquer, para passar a ser um dispositivo que percebe o mundo a sua volta e reage às mudanças de situações. Os sensores detectam parâmetros do ambiente que o cerca e dos objetos a serem manipulados.

Sensores de proximidade são largamente utilizados em processos automatizados para detectar a presença de um objeto. Os mais empregados na automação de máquinas e equipamentos industriais são o tipo chaves de fim de curso, capacitivos, indutivos, ópticos, magnéticos e ultrassônicos.

A tabela 1 apresenta os principais sensores utilizados em robótica móvel, demonstrando o quão complexo pode ser o projeto de um sistema robótico que envolve a especificação e seleção de diversos componentes em um sistema autônomo.

Tabela 1 – Sensores para robôs móveis

Fonte : (Osorio, 2009).

Sensor	Principal função	Exemplos
	Determinar a posição	GPS
De posição	absoluta ou direção de	Bússola
e orientação	orientação do robô	Inclinômetro
	Determinar a distancia	Sensor infravermelho
De obstáculos	até um objeto ou	Ultrassom
	obstáculo	radar
	Determinar o contato	Sensores de contato
De contato	Com um objeto ou	Antenas e “bigodes”
	Posição de contato com	Marcações
	marcação	
		Inercial
De Deslocamento	Medir o deslocamento	Odômetro
E velocidade	do robô	Potenciômetros
	Envio e recepção de	Sistemas de visão
Para comunicação	dados e sinais externos	Sensores óticos
		Sistemas de comunicação(RF)

A tabela 2 apresenta os principais atuadores utilizados na robótica móvel, indicando sua função e dando exemplos de robôs largamente difundidos no mercado.

Tabela 2 – Atuadores para robôs móveis

Atuador	Principal tipo/função	Exemplos
Base Fixa	Braço robótico com base fixa	Robos industriais PUMA
	2 rodas independentes	Robos Khepera e Pioneer P3-DX
Base móvel: Rodas	3 rodas	Robô Brainstem PPRK
	4 rodas	Stanley – Stanford
Base móvel : esteira	Esteira(Slip/Skid locomotion tracks)	Tanques e veículos militares
Base móvel:	Bípedes	Robôs humanoides
Articulações	4 patas	Robôs Sony Aibo, Bigdog
	6 patas	Robôs inseto

Fonte: (Osorio, 2009).

O grande desafio da robótica é justamente ligar os dados vindos destes sensores para gerar comandos e controlar estes atuadores, para garantir que a tarefa seja executada sem colocar em risco o próprio robô ou aqueles que o cercam. Um robô só deve agir sobre um determinado alvo caso tenha sido programada sua ação sobre o mesmo.

3.4.3. Controle Inteligente

Na Inteligência artificial, um *software* contemporâneo é visto como um sistema dinâmico, onde a percepção e a ação constituem processos simultâneos e inseparáveis (10).

O princípio de aprendizado que será utilizado neste trabalho é implementado através de alterações nos pesos dos neurônios. Estas alterações podem ser realizadas de várias formas. Não existe ainda uma teoria que diga qual é a melhor forma de aprendizagem robótica. Apesar disso, pode-se enumerar muitas características desejáveis e necessárias em um algoritmo de aprendizagem, como por exemplo, tolerância a ruídos, convergência rápida, adaptabilidade, dependência apenas de informações que possam serem extraídas dos sensores de entrada (10).

Outro princípio é o dos processos paralelos e fracamente acoplados, que consiste em descentralizar o controle, distribuir tarefas para certo número de processos em paralelo que estão acoplados diretamente ao aparato sensório-motor do agente.

O princípio do agente absoluto envolve três componentes interconectadas e interdependentes, que são a fixação do nicho onde o sistema irá atuar, estabelecimento desejado ou tarefas a serem cumpridas e a determinação do agente propriamente dito. E o agente deve priorizar quatro aspectos, autonomia, autossuficiência, localidade e corporificação (10).

4. METODOLOGIA

A fundamentação teórica foi utilizada para a discussão de uma rede recorrente de camada única, com algoritmo de aprendizado baseado na correção de erros, para a atuação de um servo motor para orientar a direção de um robô de base móvel de roda centralizada. Todas as teorias sobre estas etapas estão descritas no estado da arte, e sua metodologia será detalhada passo a passo nesta seção.

Os conceitos foram aplicados com algumas alterações em relação ao previsto durante o pré-projeto do trabalho. Foram feitas alterações no modo de interface entre *hardware* e treinador durante a etapa de treinamento, o que acarretou uma maior facilidade na mesma, porém uma complexidade muito maior quanto ao controle, e ao *hardware* criado. O sistema descrito na figura 13.

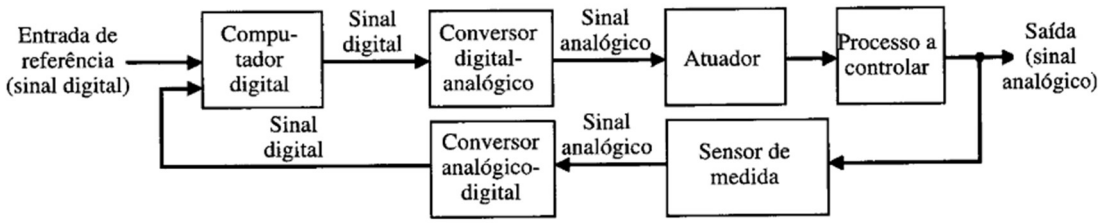


Figura 13 - Sistema de Controle

Fonte: Autoria Própria.

Na primeira etapa é mostrada a leitura analógica do ambiente externo através de um sensor. Na segunda etapa a transformação de sinal analógico para sinal digital é realizada através de um conversor A/D. Na terceira etapa, o processamento digital dos sinais sensores foi realizado utilizando o circuito integrado *FPGA* como controlador, baseado em RNA's. Neste estudo de caso, descarta-se a utilização do conversor D/A, pois o servo motor utilizado é controlado por *PWM,* onde a própria saída da *FPGA* fornece a lei de controle do atuador, no caso o servo motor. A seguir, será analisada cada etapa deste processo, detalhando desde o funcionamento até o componente utilizado.

Na primeira etapa foram utilizados três sensores do tipo *LDR (Light Diode Resistor),* é um resistor dependente da luz, ou seja, a variação de sua resistência ocorre de acordo com a intensidade luminosa incidente.

A relação entre resistência e intensidade luminosa é inversamente proporcional, ou seja, quanto maior a intensidade luminosa menor sua resistência. Os valores típicos de resistência para um ambiente completamente

iluminado são de 100Ω, na ausência total de luz alcança sua resistência máxima de aproximadamente 100M Ω.

Na segunda etapa, conversor analógico digital utilizado é o ADC0804, um conversor A/D de 8 bits de saída em paralelo. O que possibilita mapear bit a bit do mais significativo (MSB) até o menos significativo (LSB). À exemplificação retirada do *Datasheet* do componente um diagrama de ligação, demonstrado na figura 14. Este circuito não chegou a ser montado, é apenas uma sugestão para trabalhos futuros.

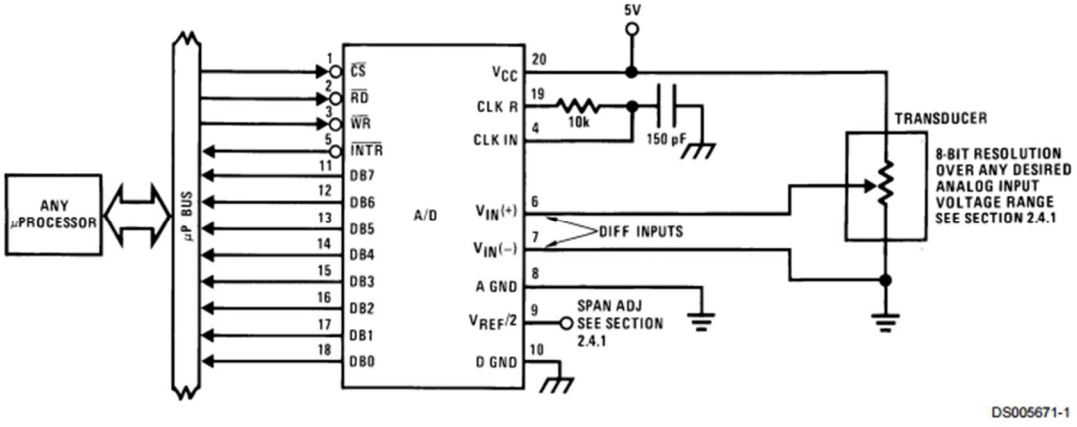


Figura 14 - Diagrama de montagem do *ADC*

Fonte : *Datasheet* do ADC0804 (11).

Observa-se na figura 14, a utilização de um sensor para medição de tensão, com entrada em diferencial de tensão (entre os pinos 6 e 7). Uma tensão de referência de 5V, é fixada, conforme apresentado pelo *datasheet*. No pino 9, a tensão de referência é dividida por 2, resultando em 2,5V.

Na terceira etapa, o bloco de processamento digital composto por unidades de entrada e seus respectivos formatos para o devido tratamento, unidades de saída, módulo *PWM* para atuação do sistema através do servo motor.

O processamento digital foi realizado com o circuito integrado FPGA. O projeto do controle inteligente baseado em RNA foi feito usando a linguagem de descrição de hardware VHDL. Foram construídos quatro blocos principais, denominados: Regulagem, SaídaPWM, RNA e Controle.

O bloco de regulagem nada mais é do que o tratamento da saída da RNA para uma classificação no bloco SaídaPWM. A figura 15 ilustra o bloco de regulagem, como dito anteriormente este bloco trata a saída da RNA e converte para uma classificação no bloco de saída. Possui duas variáveis de entrada e uma variável de saída, como a saída é limitada de 1 a 10, utiliza-se apenas 3 bits.

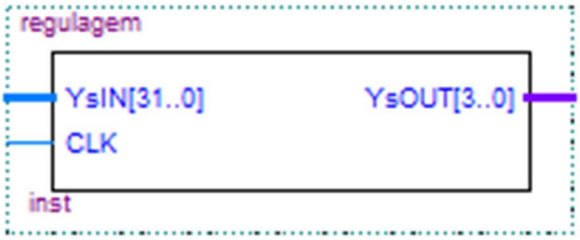


Figura 15 - Bloco de Regulagem
Fonte: Autoria própria

O bloco SaídaPWM faz a classificação entre o sinal de saída da RNA regulado para um sinal *PWM* para atuação do sistema. Duas análises simples devem ser feitas para este bloco, a utilização do clock interno da *FPGA* de 50MHz e a faixa de operação do *PWM* do servo motor a ser atuado. A figura 16 ilustra o bloco de saidaPWM, tem como entrada a saída regulada e como saída um sinal de *PWM.*

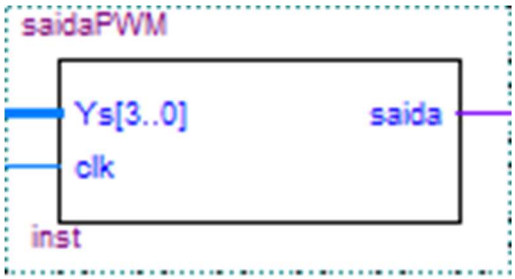


Figura 16 - Bloco de saída
Fonte: Autoria própria

O bloco RNA é um conjunto de blocos, criados a partir da topologia de rede recorrente de múltiplas camadas, onde forma-se uma RNA de 3 neurônios de entrada e 1 camada oculta, aumentando assim a capacidade computacional da rede, levando o sistema a ser capaz de resolver problemas não-lineares. Um dos principais desafios na hora de projetar esse bloco, são as interconexões entre controle e RNA, para auxiliar no treinamento e na leitura de dados internos, como pesos. A figura 17, ilustra o bloco formado pela RNA, possui 7 variáveis de entrada e 13 variáveis de saída. Sendo as entradas designadas tanto de sinais de controle, como leituras do ambiente externo. As saídas são todos os pesos da RNA e a saída da RNA propriamente dita.

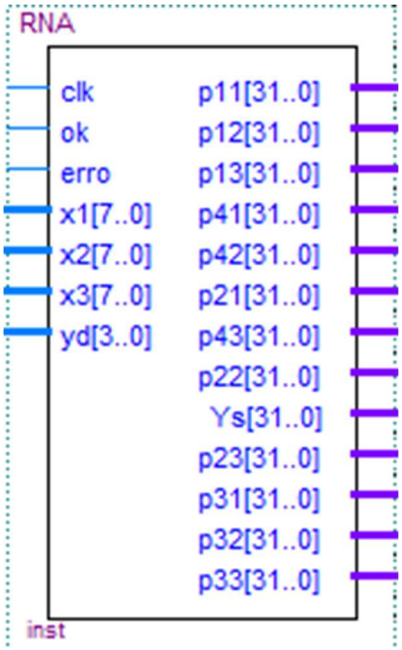


Figura 17 - Bloco da RNA
Fonte: Autoria própria

A figura 18 ilustra o bloco da RNA, demonstrando a ligação, pinos de entrada e saída e blocos contidos, como dito anteriormente possui 7 pinos de entrada e 13 pinos de saída.

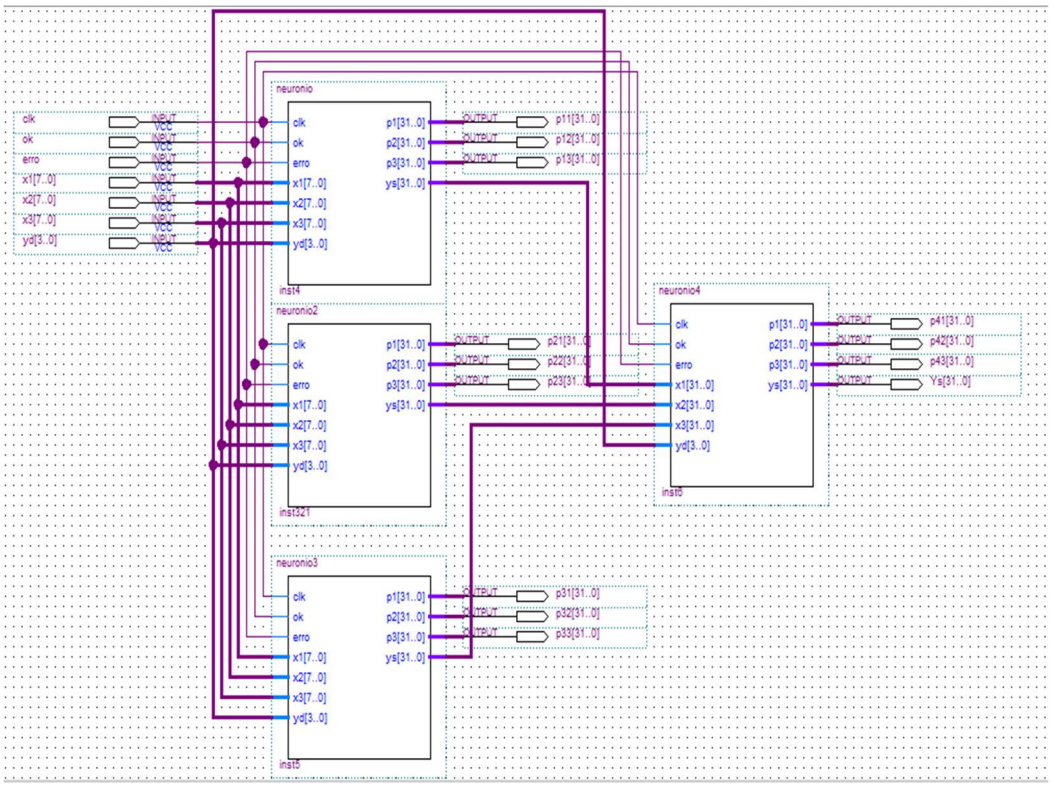


Figura 18 - Bloco de blocos da RNA
Fonte: Autoria Própria

O bloco de controle é responsável por monitorar o treinamento da rede, tal como tratar a interface usuário/*hardware* durante a mesma, disponibiliza a possibilidade de um controle entre treinamento/aplicação do *hardware*, além de disponibilizar ao “treinador” o monitoramento dos pesos de cada neurônio, fazendo apenas uma alternância a partir de uma chave seletora. O monitoramento dos pesos faz-se necessário por conta da *FPGA* ser dinâmica, ou seja, quando for reinicializada o programa voltará com os pesos iniciais, fazendo-se assim incapaz da recuperação do treinamento que fora efetuado até o dado momento. Tal abordagem poderia ser feita de outra maneira, dado que a *FPGA* possui uma memória externa acessível.

A figura 19 ilustra o bloco de controle, o bloco mais complexo pois nele passa todas as variáveis de controle e ainda faz a interface entre usuário/*hardware*. Possui 18 variáveis de entrada e 10 variáveis de saída. As variáveis de entrada podem ser descritas por variáveis de monitoramento da rede, tal como pesos e saída e variáveis de interface de usuário, tal como as chaves para manipulação dos valores a serem inseridos. A saída do bloco também pode ser dividida em dois gêneros, variáveis de controle para a rede e variáveis de interface entre usuário, tais como displays de hexadecimal.

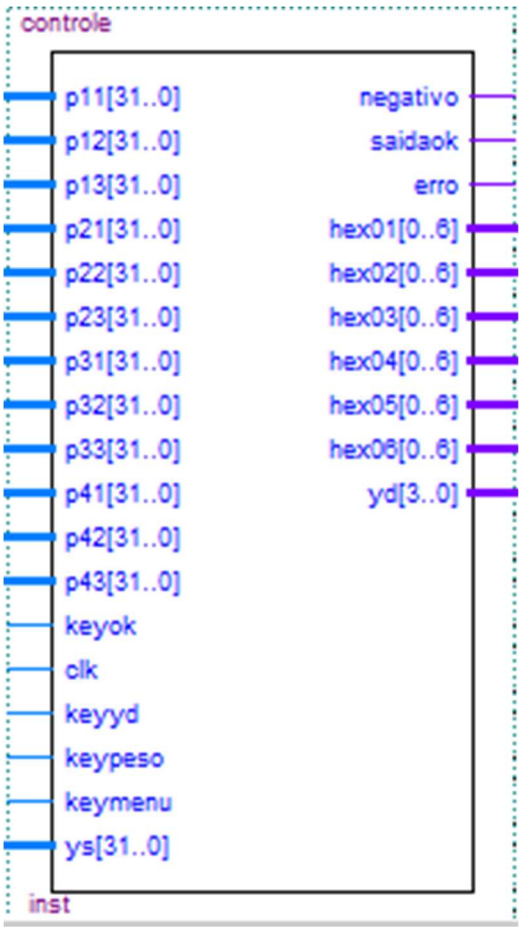


Figura 19 - Bloco de Controle
Fonte: Autoria própria

A figura 20 ilustra o bloco de neurônio artificial, possui 7 variáveis de entrada e 4 variáveis de saída. Como espera-se as entradas podem ser descritas por leituras do ambiente externo e por variáveis de controle. As saídas são os pesos e a saída propriamente dita do neurônio.

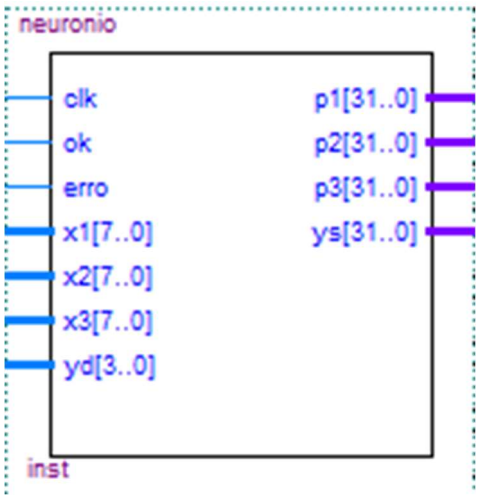


Figura 20 - Bloco de Neurônio da primeira camada
Fonte: Autoria própria

A figura 21 ilustra o neurônio de camada oculta, como o bloco de neurônio de primeira camada, possui 7 variáveis de entrada e 4 variáveis de saída, porém nota-se entre colchetes o tamanho da variável utilizada, isto explica a necessidade de criar um bloco em separado para o neurônio da camada oculta em questão, como suas entradas são variáveis baseadas na saída dos neurônios de primeira camada, não devem ser limitados pois pode-se perder informação, limita-se o tamanho de uma variável ou entrada de um sistema para diminuir custo computacional, logo não é viável deixar a saída ilimitada dos neurônios de primeira camada.

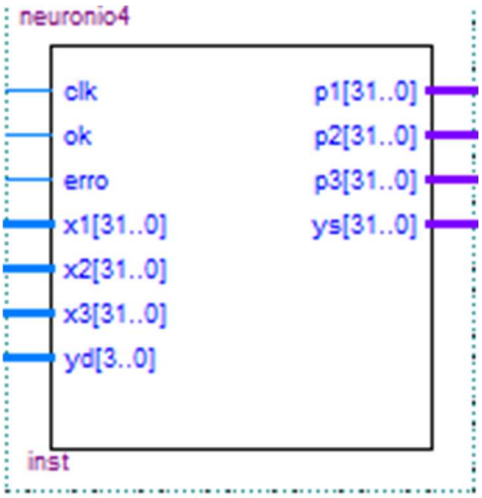


Figura 21 - Bloco de neurônio da camada oculta
Fonte: Autoria própria

Como dito anteriormente a conexão entre RNA/controle fora trabalhosa, pois o treinamento da rede deve ser feito como um todo, ou seja, analisado de fora e passado um comando de alteração ou não de peso, assim alterando todos os pesos dos neurônios da rede.

Será detalhado as entradas e saídas do sistema de processamento digital, as entradas do sistema podem ser vistas como:

x_1, é dado por um sinal inteiro, vindo do conversor A/D variante de 0 a 255;

x_2, é dado por um sinal inteiro, vindo do conversor A/D variante de 0 a 255;

x_3, é dado por um sinal inteiro, vindo do conversor A/D variante de 0 a 255;

$KeyOk$, é um sinal em forma de *bit*, quando está em '0', significa que as leituras das entradas e os sinais a serem colocados pelo "treinador" estão "OK", e pode ser efetuada uma etapa de treinamento. Como é um algoritmo recursivo, algo deve efetuar uma certa pausa para o treinador ajustar os parâmetros.

$KeyYd$, é um sinal em forma de bit, tem a função de dar ao "treinador" uma opção para alterar a saída desejada antes de ser dado como "OK" o treinamento da rede, sempre que for lido como '0', fará uma soma no contador que varia de 1 a 9.

$KeyPeso$, da mesma forma que o $KeyYd$, é um sinal do tipo *bit*, e constitui-se de forma parecida, dá ao usuário uma certa alternância para vagar entre os pesos dos neurônios, para serem demonstrado no display, variando de 1 a 15.

$KeyMenu$, também um sinal em forma de *bit*, tem a função de dar uma alternância de leitura ao treinador, quando em '0' mostra em seis display de 7 *segmentos* o neurônio lido e o peso referente àquele neurônio, tanto quanto o valor absoluto de 0 a 9999. Quando em '1' mostra ao usuário a saída desejada que está sendo inserida.

Clk, sinal básico de elementos em *VHDL*, novamente do tipo *bit*, utilizou-se o próprio *clock* interno da *FPGA* para este sinal, como trata-se de um algoritmo recursivo, precisa-se de um sinal alternante para rodar a função *process*.

Acima são as entradas do sistema de controle digital, abaixo serão detalhadas as saídas do mesmo:

$negativo$, sinal em forma de *bit,* onde é '1' caso o peso analisado no momento é negativo e '0' caso contrário.

Hex_{01}, sinal em forma de *bit_vector* de sete posições, é responsável pelo display de sete *segmentos* onde fica responsável por demonstrar o neurônio lido no momento ou um 'Y' variando de acordo com a opção menu.

Hex_{02}, sinal em forma de *bit_vector* de sete posições, é responsável pelo display de sete *segmentos* onde fica responsável por demonstrar o peso lido no momento ou um 'd' variando de acordo com a opção menu.

Hex_{03}, sinal em forma de *bit_vector* de sete posições, é responsável pelo display de sete *segmentos* o qual fica a cargo de demonstrar o algarismo mais significativo do valor absoluto do peso lido no momento ou o valor absoluto da saída desejada que está sendo inserida pelo "treinador" variando de acordo com a opção menu.

Hex_{04}, sinal em forma de *bit_vector* de sete posições, é responsável pelo display de sete *segmentos* o qual fica a cargo de demonstrar a centena do peso lido no momento.

Hex_{05}, sinal em forma de *bit_vector* de sete posições, é responsável pelo display de sete *segmentos* o qual fica a cargo de demonstrar a dezena do peso lido no momento.

Hex_{06}, sinal em forma de *bit_vector* de sete posições, é responsável pelo display de sete *segmentos* o qual fica a cargo de demonstrar a unidade do peso lido no momento.

Ys_{out}, sinal em forma de inteiro, apenas para a verificação do valor de saída da rede, trata de uma saída inutilizada ficando apenas para simulação, facilitando assim a verificação do treinamento da rede.

$Saida$, sinal em forma de bit, é um sinal em forma de *PWN* que é a lei de controle para atuar o servo motor.

Nota-se um grande número de variáveis de comunicação entre blocos, elas são dadas devido a necessidade de monitoramento previamente citada, e ao algoritmo de treinamento ter necessidade de ser monitorado de forma externa à rede. Sendo:

P_{11}, P_{12}, P_{13}, P_{21}, P_{22}, $P_{23}, P_{31}, P_{32}, P_{33}, P_{41}, P_{42}, P_{43}$, valores inteiros, que trafegam do bloco da RNA para o bloco de controle que faz deles um monitoramento, previamente citado para o treinador acompanhar esses valores em um display.

Y_s, também um sinal inteiro, que trafega do bloco da RNA para o bloco de controle e da RNA para o bloco de regulagem, nada mais é que a saída absoluta da RNA em relação às entradas oferecidas. Faz-se necessário esta transmissão para ser efetuado um cálculo de erro gerando um sinal de comando para a própria RNA e para a transmissão da saída obtida para ser feita a regulagem.

$erro$, sinal em forma de *bit*, é um sinal que transmite um comando de positivo ‘1’ ou negativo ‘0’ do bloco de controle para o bloco de RNA, fazer ou não a substituição do peso durante a etapa de treinamento.

Ys_{out}, sinal em forma de inteiro limitada entre 1 e 9, o sinal conecta a saída regulada para a geração de uma saída em formato de *PWM* no bloco de saída. É limitada pois existem 10 posições em que o servo será mapeado, entre 0º e 180º, como é saída do bloco de regulagem os valores serão limitados dentro do mesmo.

A figura 22 demonstra o bloco final do sistema, com todos os blocos desenvolvidos, os pinos de entrada e saída e suas ligações. Possui 8 pinos de entrada e 8 pinos de saída. Os pinos de entrada podem ser divididos em comandos do usuário e leituras do ambiente externo. Os pinos de saída são praticamente todos de interface usuário/*hardware* e um pino de saída propriamente dita do sistema.

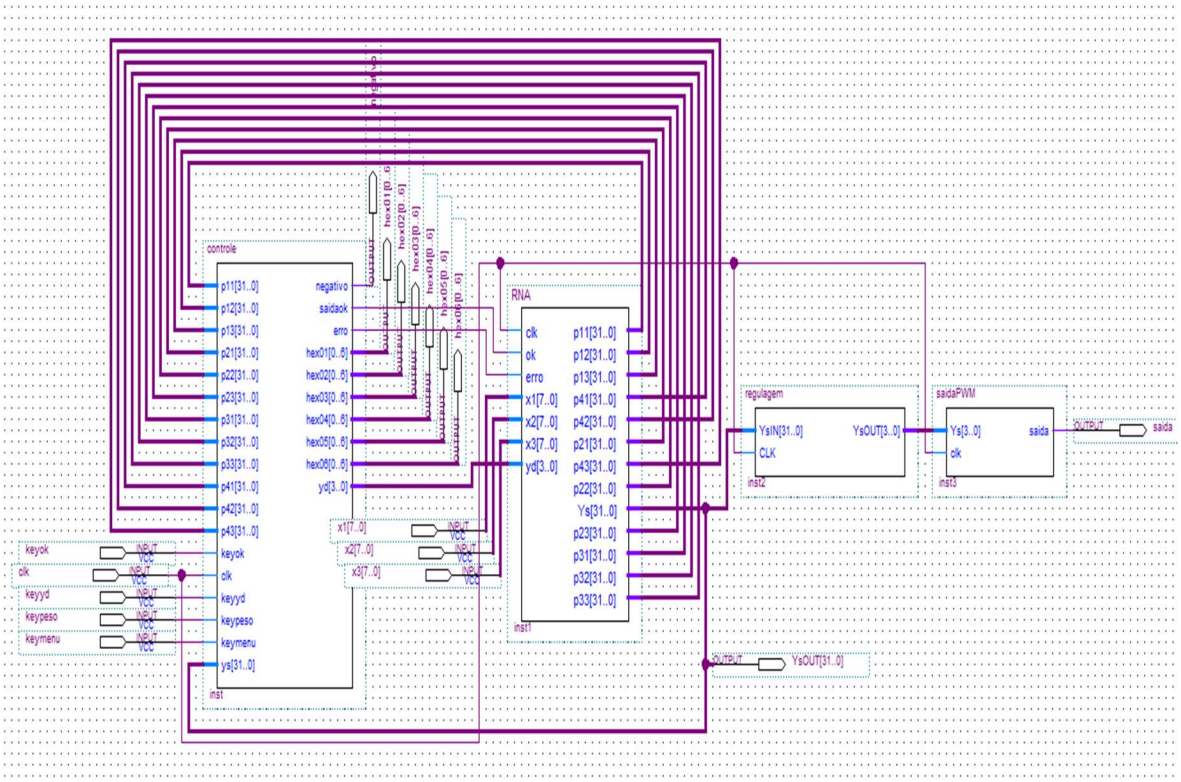


Figura 22 - Bloco final do sistema
Fonte: Autoria Própria

A figura 23 demonstra o diagrama de comando do servo, onde a amplitude do *PWM* deve ser de 5V, período de 50Hz, e largura de pulso variante de $1ms$ para -90º e $2ms$ para 90º. Como a norma de controle do atuador do sistema é feita de forma digital, dispensa-se conversor Digital/Analógico da figura 12.

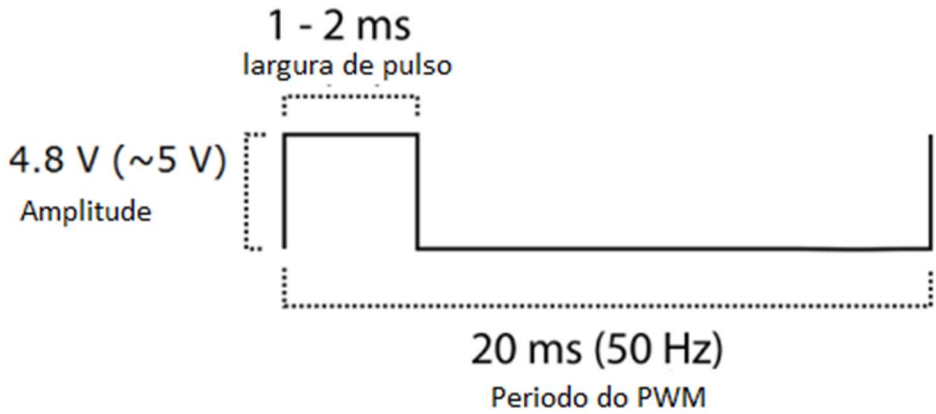


Figura 23 - Diagrama de comando do servo
Fonte: Adaptada de *Datasheet* do servomotor 9g SG90 (12).

Algo notório durante o projeto foram as inúmeras alterações durante a etapa de treinamento, visando melhorar o desempenho do sistema e a comodidade do treinador durante esta etapa. Primeiramente a ideia era criar um banco de dados

de treinamento em *excel,* a partir destas amostras fazer um bloco de leituras de dados executando todas iterações fazendo o treinamento automático. Porém tal processo utilizava muito consumo de *hardware,* além de limitar o treinador ao número de instâncias de treinamento, sendo assim colocado de lado pelo novo método que fora considerado mais iterativo além de proporcionar ao treinador maior conhecimento da rede.

Para o acompanhamento do que estava sendo criado no *Quartus II* foi utilizado um software matemático auxiliar, onde possui-se um número de ferramentas amplo, o *software Matlab* foi escolhido por possuir facilidade na criação de scripts idênticos ao que foram utilizados para a criação de cada bloco, assim tornando-se capaz de replicar os resultados em um *software* que passa mais rápido dado que precisa de uma quantidade menor de processamento por tratar apenas da matemática por trás da rede.

Além de replicar a RNA desenvolvida anteriormente em *VHDL,* o *Matlab* possui uma ferramenta específica, chamada *nntool* própria para redes neurais, que foi utilizada para acompanhar o que vinha sendo feito e possibilitar uma visão ampla da convergência ou não do sistema.

Dada a não convergência da base de dados que estava sendo utilizada com funções de ativações limiares, foi alterada a base de dados, para uma possível convergência sem a necessidade de uma matemática mais pesada, dada que a implementação no *Quartus II*, limita o sistema matematicamente.

Após a convergência da nova base de dados visualizada no *nntool,* a utilização de um script que replica a RNA criada no *Quartus II* possibilitou uma visão ampla do sistema, dado que a simulação no *Matlab* é mais simples, pois utiliza-se apenas de ferramentas matemáticas e não simulam o comportamento de milhares de elementos lógicos.

O script foi projetado para comportar mudanças rápidas, visando melhorar o desempenho. O que possibilitou analisar por tentativa e erro a melhor iniciação para as variáveis do sistema.

5. RESULTADOS OBTIDOS

Dado o alto custo computacional do *hardware* desenvolvido a simulação do sistema no *Quartus II* tornou-se inviável. Foram feitas diversas rotinas de treinamento, com diversas inicializações de pesos e alternâncias da taxa de aprendizado visto na equação (1.4), até a própria rede foi alterada, obteve-se assim diversos códigos semelhantes.

Os resultados demonstrados a seguir foram gerados a partir do mesmo código disponibilizado em anexo deste trabalho, porém sem funções de controle tais como displays e auxiliares do usuário. Tais funções foram retiradas para reduzir o custo computacional assim fazendo o maior número de iterações possível.

É de suma importância ressaltar que o treinamento de uma RNA está sempre a cargo do seu "treinador" então colocar certos padrões são de suma importância para a convergência da rede. Para efetuar tal simulação via *Quartus II*, foi criado uma tabela de entradas aleatórias e a partir desta tabela de entradas aleatórias uma determinada saída desejada da rede, procura-se ser o mais fiel possível a um ambiente não-controlado no qual o sistema seria inserido as entradas foram variadas de 0 a 255, e suas saídas mapeadas de 1 a 9. Primeiramente serão demonstradas aqui as iterações executadas na simulação e explicadas como foram escolhidas.

Os quadros 1, 2 e 3 estão disponíveis em apêndice e demonstram os bancos de dados criado em primeira instância, onde os valores de x_1, x_2, x_3 são variantes de 0 a 255 e os valores de y_d são variantes de 1 a 9. Onde 9 seria totalmente a esquerda no caso a saída em *PWM* com uma largura de banda de 1ms, o servo motor estaria com uma angulação de -90º e 1 seria o servo motor a 90º, com a saída em *PWM* com uma largura de banda de 2ms. Entende-se que os resultados esperados sejam sempre para o lado com menor luminosidade, ou seja, com maiores valores de leitura.

Dado um pequeno número de amostragem, 100 amostras no melhor caso já se tende a uma saída falha. Como citado anteriormente o simulador do *Quartus II* chamado *Modelsim,* impossibilitou treinamento com mais amostras. E mesmo para um número baixo de amostras ele indica possíveis falhas na hora da simulação dado a complexidade do sistema.

A figura 24 ilustra o *Compilation Report* do programa de simulação, com as informações de *hardware* do sistema criado para tal, nota-se um baixo número de elementos lógicos, menor que 7% da capacidade da *FPGA* utilizada. Nota-se um número de elementos lógicos reduzidos quando comparado ao sistema com diagrama de controle completo, reduzindo para 8492 elementos lógicos. Uma visível queda no número de pinos, dado a retirada dos displays de sete segmentos.

Flow Status	Successful - Fri Jun 03 00:03:13 2016
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	Simulacao2
Top-level Entity Name	Simulacao2
Family	Cyclone IV E
Device	EP4CE115F29C7
Timing Models	Final
Total logic elements	8,492 / 114,480 (7 %)
Total combinational functions	8,453 / 114,480 (7 %)
Dedicated logic registers	738 / 114,480 (< 1 %)
Total registers	738
Total pins	62 / 529 (12 %)
Total virtual pins	0
Total memory bits	0 / 3,981,312 (0 %)
Embedded Multiplier 9-bit elements	90 / 532 (17 %)
Total PLLs	0 / 4 (0 %)

Figura 24 - *Compilation Report*
Fonte: Autoria própria

A figura 25 ilustra a simulação do sistema como um todo, onde para cada subida de *clock* possui uma alteração em todas as entradas e a variável “keyok” é uma entrada do usuário que denomina se a RNA está em fase de treinamento (valor 0) ou não está em fase de treinamento (valor 1). Além dessa variável de entrada, a simulação contém como entrada x_1, x_2, x_3, y_d, que são os valores de leitura externa e o valor de saída desejada. E como variável de saída a saída real obtida pela RNA.

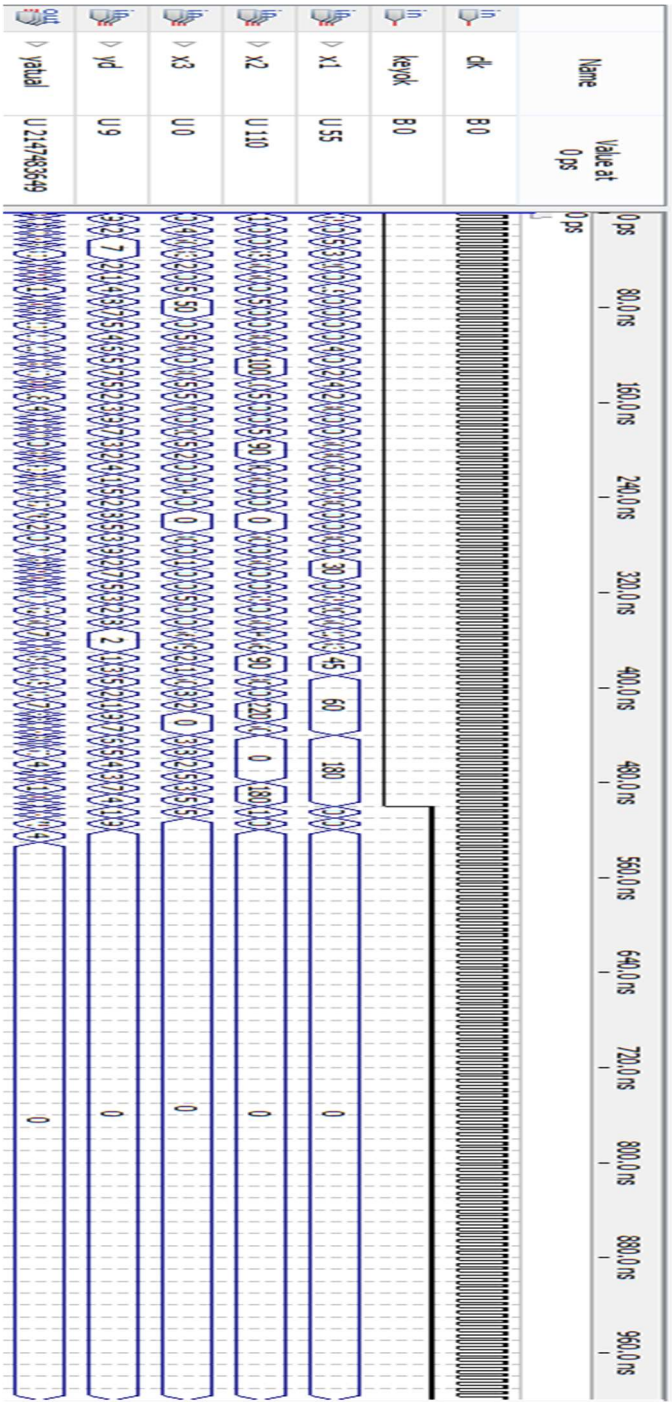


Figura 25 - Simulação para o sistema com 50 amostras
Fonte: Autoria própria

A figura 26 ilustra um zoom na parte final da figura 25, capacitando assim visualizar os valores inseridos como entrada a cada pulso de clock, além de observar a saída obtida da rede.

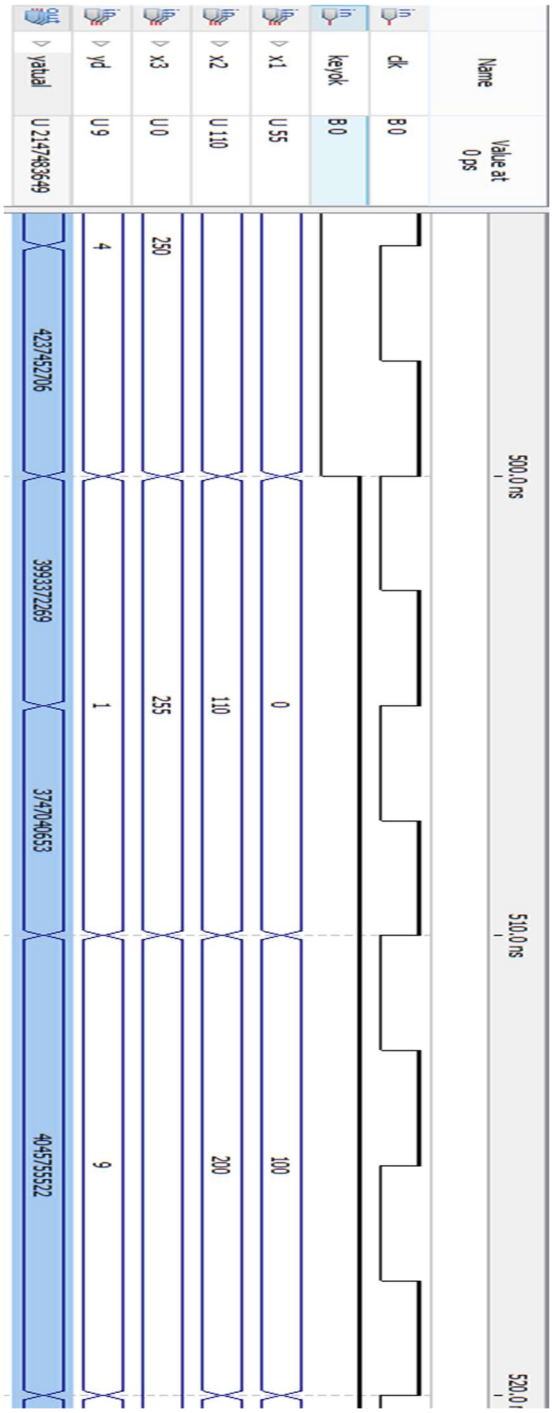
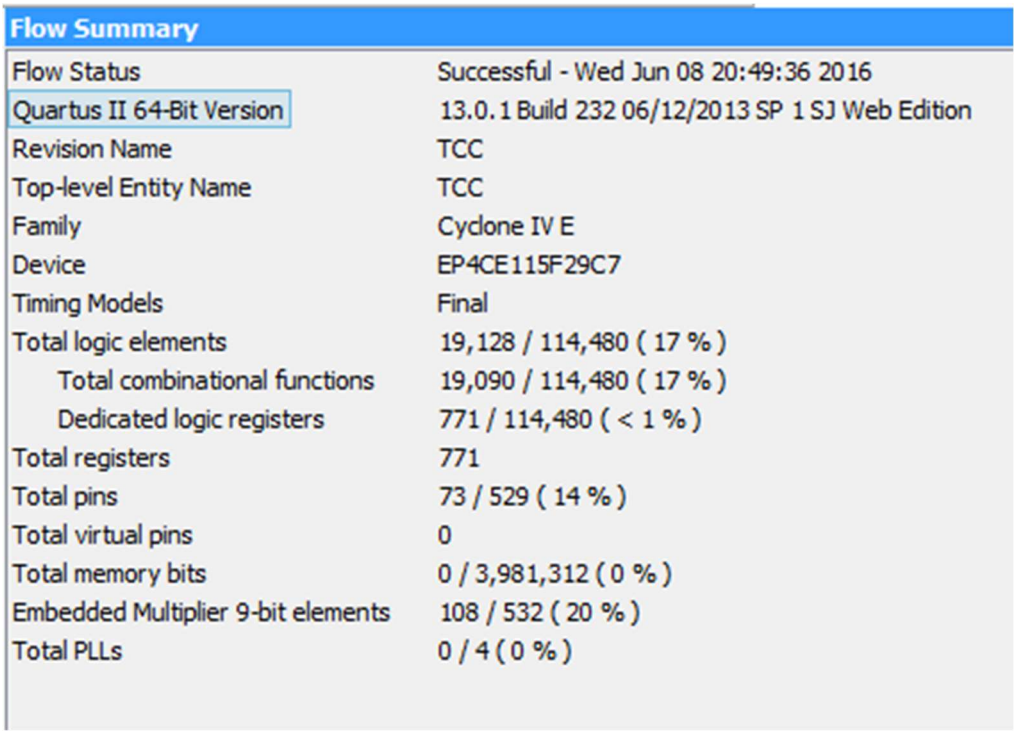


Figura 26 - Simulação para o sistema de 50 amostras
Fonte: autoria própria

Lembrando que a saída do sistema está sem o bloco de regulagem, ou seja, ainda tem de ser dividida por 1000. Quando a saída desejada do sistema é 9, sua saída real seria 9000.

Nota-se visível divergência do sistema. As simulações de 20 e 100 amostras também foram montadas e simuladas, não foi visualizado diferenças entre as simulações.

A figura 27 ilustra o *compilation report* para o sistema de controle como um todo, que foi mostrado durante a metodologia. Nota-se um visível acréscimo dado o sistema de controle entre usuário/*hardware* que foi desenvolvido e focado neste trabalho. Abaixo serão demonstradas imagens do funcionamento deste sistema na *FPGA.* Seu desenvolvimento é de suma importância para o treinamento da rede.



Flow Summary	
Flow Status	Successful - Wed Jun 08 20:49:36 2016
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	TCC
Top-level Entity Name	TCC
Family	Cyclone IV E
Device	EP4CE115F29C7
Timing Models	Final
Total logic elements	19,128 / 114,480 (17 %)
Total combinational functions	19,090 / 114,480 (17 %)
Dedicated logic registers	771 / 114,480 (< 1 %)
Total registers	771
Total pins	73 / 529 (14 %)
Total virtual pins	0
Total memory bits	0 / 3,981,312 (0 %)
Embedded Multiplier 9-bit elements	108 / 532 (20 %)
Total PLLs	0 / 4 (0 %)

Figura 27 - *Compilation Report* do sistema.
Fonte: Autoria própria

Dado a visível divergência da rede, foi utilizado o *software* auxiliar para fazer um acompanhamento da rede. O *Matlab* foi escolhido como dito anteriormente, e uma ferramenta dele chamada *nntool* foi de suma importância para a continuação desse projeto. A primeira verificação foi quanto a base de dados utilizada, dado que essa ferramenta utiliza uma matemática mais pesada e algoritmos de treinamentos melhores elaborados, a não convergência nesta ferramenta provaria a incapacidade da base de dados estudada.

A figura 28 ilustra perfeitamente a divergência da base de dados criada para o sistema efetuado. Pode visualizar um gráfico de erro médio quadrático por épocas de treinamento, vale ressaltar que cada época de treinamento é o conjunto inteiro da base de dados, ou seja, 100 iterações por época.

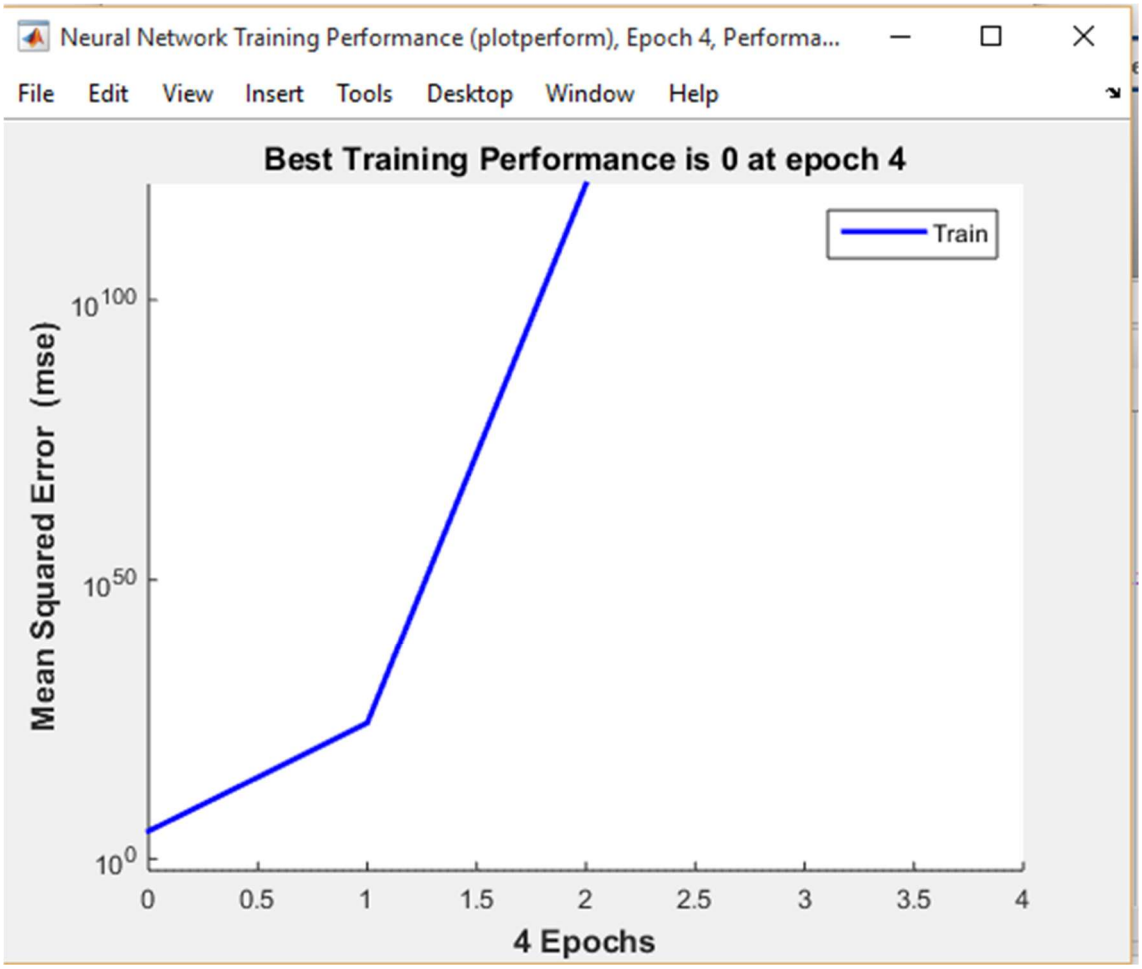


Figura 28 - Resposta do banco de dados utilizando o *nntools*
Fonte: Autoria Própria

Dado que é uma base de dados não linearmente separável, as funções de ativação da camada oculta não poderiam ser limiares como as utilizadas. Então a primeira alternativa seria o teste da mesma base de dados que é puramente sensível ao treinador, porém com uma rede que tenha possibilidade de resolver problemas não linearmente separáveis.

A figura 29 ilustra a convergência da base de dados com a mesma topologia de rede e apenas funções de ativações diferentes. Porém a convergência é lenta, após 100 épocas e o erro quadrático médio não é completamente aceitável.

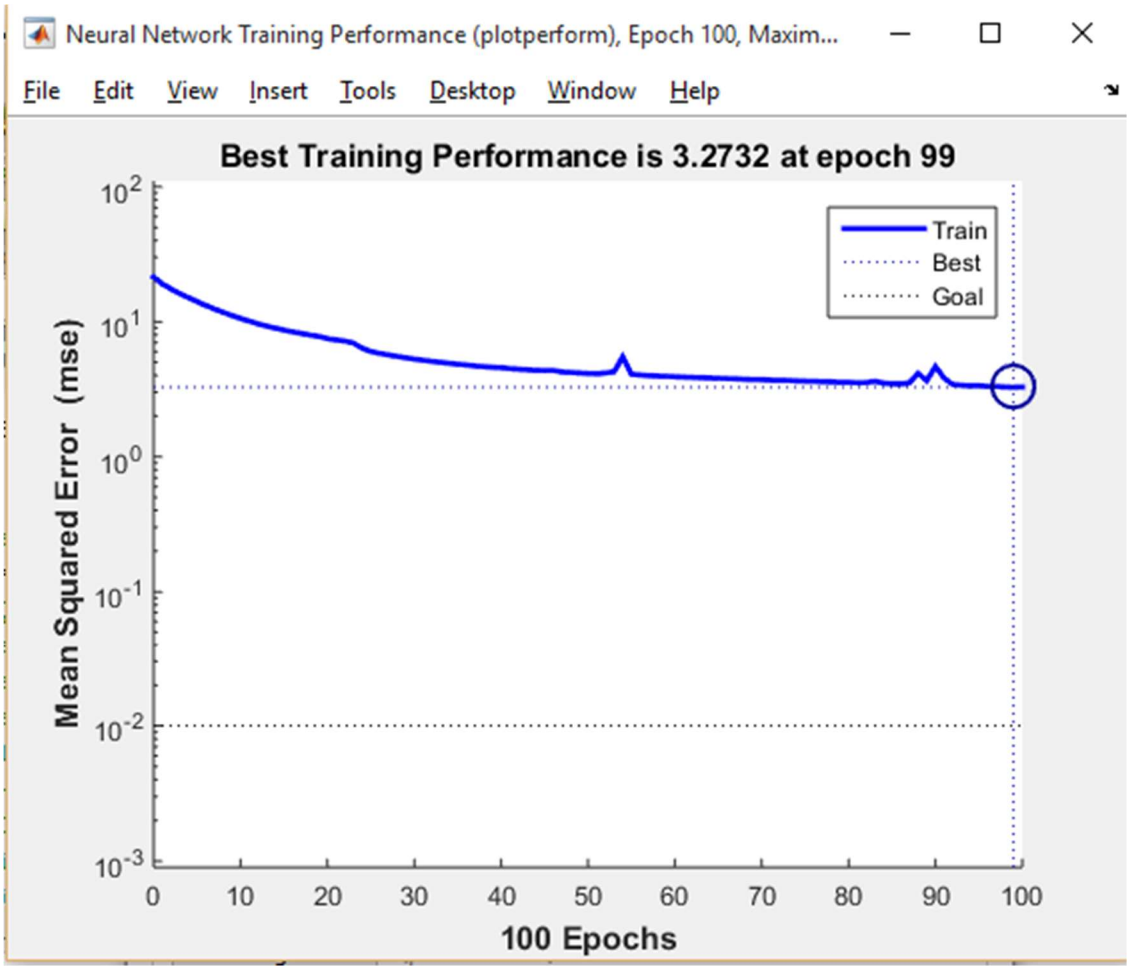


Figura 29 - Erro Quadrático Médio para uma nova função de ativação
Fonte: Autoria Própria

Foi feito alterações na base de dados, principalmente na saída desejada, essas alterações foram anexadas juntamente com a primeira base de dados. Para uma possível convergência em sistemas com funções de ativações limiar e sem demandar tanto tempo de treino.

A nova base de dados tem variância de [-30;30], nota-se uma enorme diminuição do erro quadrático médio. A figura 30 ilustra exatamente o erro quadrático médio em função do número de épocas de treinamento.

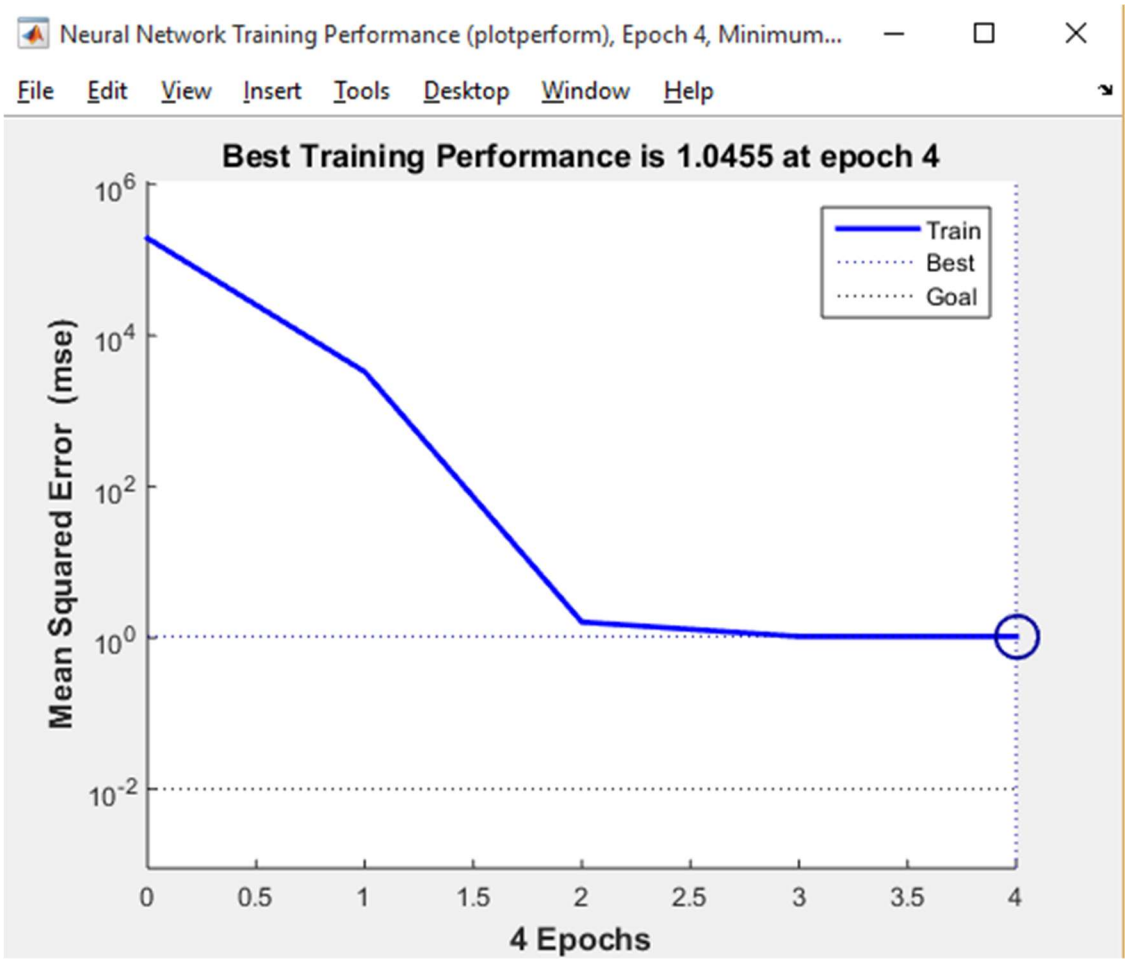


Figura 30 - Erro quadrático médio para uma nova base de dados
Fonte: Autoria Própria

O erro quadrático médio caiu de 3,27 para 1,05, além da taxa de amostragem de saída aumentar de [1;9] para [-30;30], o treinamento passa a convergir na 4ª iteração. O que prova uma ineficácia enorme da antiga base de dados em relação a nova.

Após a alteração na base de dados, foi utilizado o *Matlab* para fazer um script idêntico ao desenvolvido no *Quartus II,* dado que o *Matlab* se preocupa apenas com a matemática e não com o *hardware* desenvolvido, logo, possibilita uma gama de testes muito maior.

A RNA desenvolvida para este estudo possuía 3 neurônios de entrada, 2 neurônios na camada oculta e apenas um neurônio de saída. Pois o principal objetivo era verificar e comprovar o método de treinamento utilizado e fazer o estudo dos parâmetros iniciais.

O quadro 4 anexado a este trabalho ilustra exatamente a importância da escolha dos pesos iniciais para a convergência do treinamento e o seu número de iterações necessárias. As 4 primeiras colunas são as entradas do sistema e cada alteração de peso foram efetuadas 3 etapas de treinamento com diferente número de iterações, que foram divididas na figura 4a, 4b, 4c. Notou-se que quanto maior o peso inicial utilizado menor é sua eficiência quanto à

convergência, porém, se o sistema não divergir ela é alcançada mais rapidamente.

A tabela 3 ilustra os pesos iniciais de cada um dos quadros anexados no trabalho.

Tabela 3 - Pesos Iniciais do sistema utilizado

Peso	Quadro 4a	Quadro 4b	Quadro 4c
Peso 1	0.1	-0.01	0.001
Peso 2	0.2	0.03	-0.002
Peso 3	0.7	-0.04	0.007
W1	0.2	0.01	0.002
W2	0.5	-0.05	0.005
W3	0.8	0.01	-0.008
W4	0.1	-0.05	-0.001
W5	0.3	-0.03	-0.003
W6	0.7	0.01	0.007
U1	0.1	-0.03	0.001
U2	0.2	0.02	-0.002

A escolha dos pesos demonstrados na tabela acima foi feita por tentativa e erro e a maior importância da escolha de três diferentes é ilustrar a importância do valor deles na convergência ou não do sistema.

Para os segundos e terceiros pesos iniciais mostrados no quadro 4, nota-se um erro praticamente nulo para todas amostras do banco de dados, com exceção da linha 10, que estranhamente acusa um erro considerável. Nota-se uma dependência de um treinamento maior no último caso, o que pode ser explicado com o próximo quadro em anexo.

O quadro 5 ilustra a importância e funcionalidade da variável de ajuste de peso no sistema, os valores indicados são os valores de erro para cada amostra. Para criação deste quadro foram utilizados os menores pesos iniciais do quadro 4. Quanto menor os pesos iniciais, maior vai ser o ajuste ideal para o treinamento da rede.

O quadro 6 tem a função de ilustrar a importância do casamento entre pesos iniciais e de ajuste de peso, foram utilizados os maiores pesos contidos no quadro 4. Quando os pesos iniciais não são bons nota-se uma divergência praticamente imediata para valores de ajustes maiores. Os valores observados como NaN, vem do termo inglês utilizado pelo *matlab* para *not a number,* querendo dizer que não é um número válido para ser expressado, caracterizando assim a notável divergência da rede com as variáveis utilizadas.

A partir de tais simulações, pode-se afirmar com certeza que o método de treinamento, a base de dados e os valores iniciais são eficazes, dado o acompanhamento da convergência iteração a iteração do sistema.

O script do *Matlab* criado foi utilizado para fazer o treinamento da rede, assim como verificar a funcionalidade e eficácia do sistema. Uma vez a rede treinada, sua implementação no *Quartus II* é muito facilitada. Para tal implementação foi retirado os valores dos pesos de cada neurônio após as simulações que obteve mais sucesso. A tabela 4 ilustra exatamente estes pesos antes e depois de efetuado os treinamentos no *script* criado.

Tabela 4 - Pesos antes e depois do treinamento

	Inicial	Final	Inicial	Final
Peso 1	-0,01	-0,4066	0,001	-0,3930
Peso 2	0,03	-0,3666	-0,002	-0,3960
Peso 3	-0,04	-0,4366	0,007	-0,3870
W1	0,01	-0,3866	0,002	-0,3920
W2	-0,05	-0,0084	0,005	0,0129
W3	0,01	0,3430	-0,008	0,3773
W4	0,05	-0,3466	-0,001	-0,3950
W5	-0,03	0,0116	-0,003	0,0049
W6	0,01	0,3430	0,007	0,3923
U1	-0,03	-0,4266	0,001	-0,3930
U2	0,02	-0,3766	-0,002	-0,3960

Embora como vimos nos quadros 4b e 4c, ambas saídas são extremamente semelhantes, os pesos finais possuem algumas diferenças. Porém suas convergências tendem a ser no mesmo ponto.

Após todo o treinamento utilizando o software *Matlab* foi utilizado os pesos finais da etapa de treinamento para a elaboração de uma RNA no modelo citado, para acompanhar a resposta do sistema no *Quartus II*. A figura 31 ilustra a saída da rede e como esperado a saída é praticamente a mesma do *Matlab* apenas com alguns problemas de aproximação na saída dos sinais. Onde x_1, x_2 e x_3 são as entradas do sistema e todas as variáveis são expressas em números inteiros.

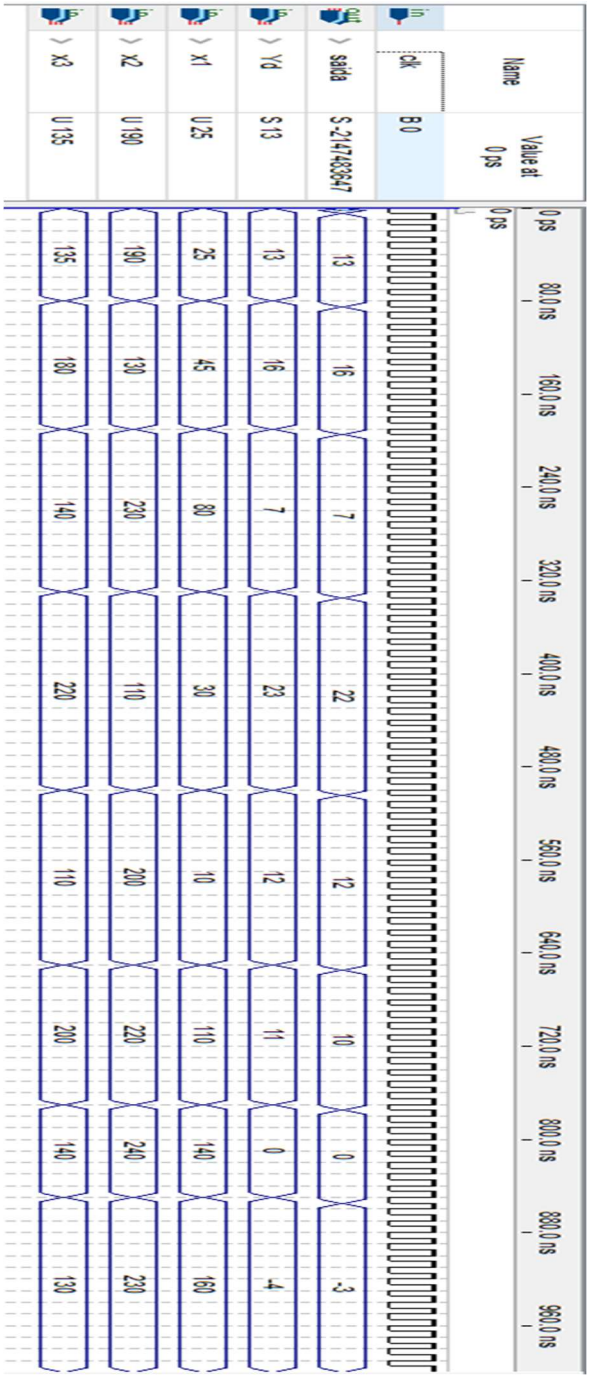


Figura 31 - Saída da RNA com os pesos treinados em *Matlab*
Fonte: Autoria própria

6. IMPLEMENTAÇÃO EM HARDWARE

Para a continuação deste trabalho e em contribuição para trabalhos futuros, foi criado uma interface usuário/*hardware* para fazer o treinamento da rede neural. Tanto para este caso específico como para outras aplicações. A seguir serão dadas explicações de seu funcionamento.

A figura 32 ilustra a utilização e localidade física da chave de alteração do tipo de menu escolhido pelo usuário. A alteração desta chave indica a leitura dos pesos atuais da RNA ou a leitura da saída desejada que está sendo inserida. No primeiro caso, os dois primeiros *displays* indicam o neurônio lido e qual o peso deste neurônio respectivamente, e os quatro últimos *displays* são responsáveis pelo valor absoluto do peso naquela iteração. No segundo caso o último *display* é responsável pela leitura do valor de saída desejada inserida.

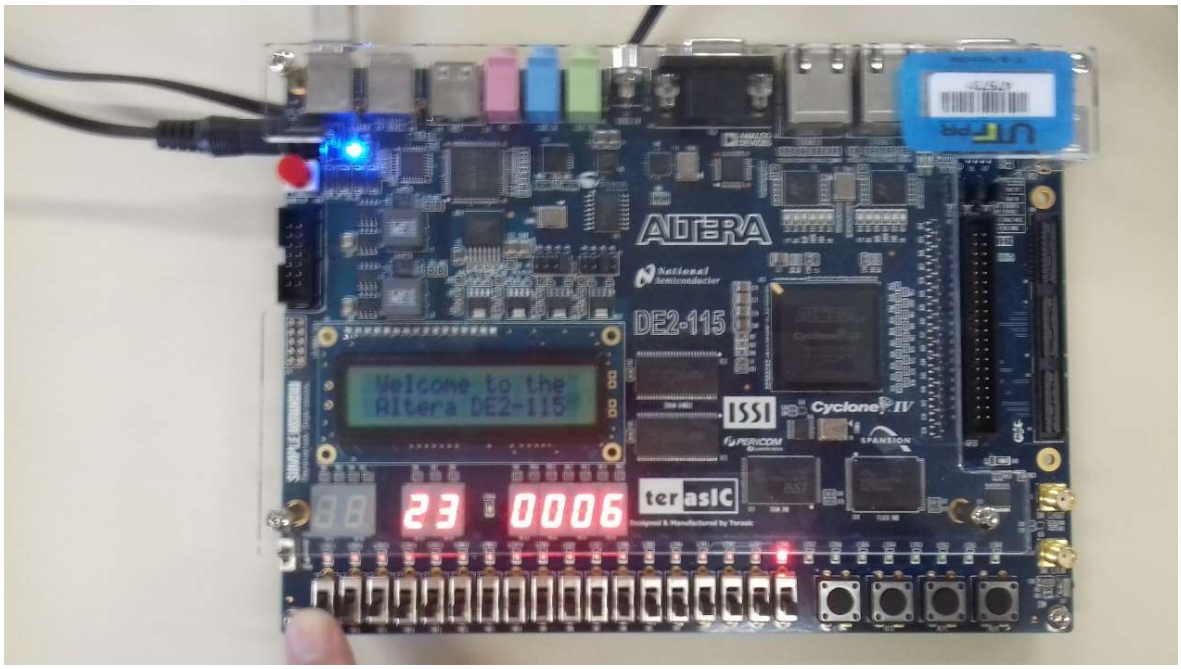


Figura 32 - Utilização do *Switch* de menu
Fonte: Autoria própria

A figura 33 ilustra o sistema com o modo de menu em alta, mostrando assim a saída desejada inserida ao sistema na iteração atual.

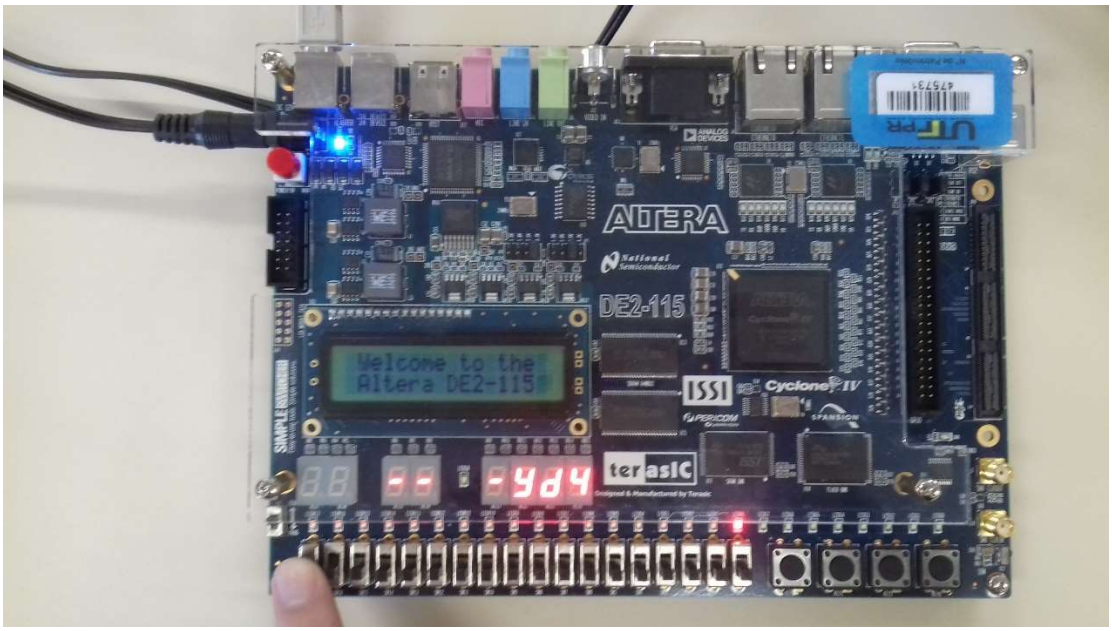


Figura 33 - Utilização do *Switch* de menu
Fonte: Autoria própria

A figura 34 mostra a localização física mapeada e a utilização da chave *key2* da *FPGA* que no sistema é utilizado para alterar a saída desejada durante a etapa de treinamento.

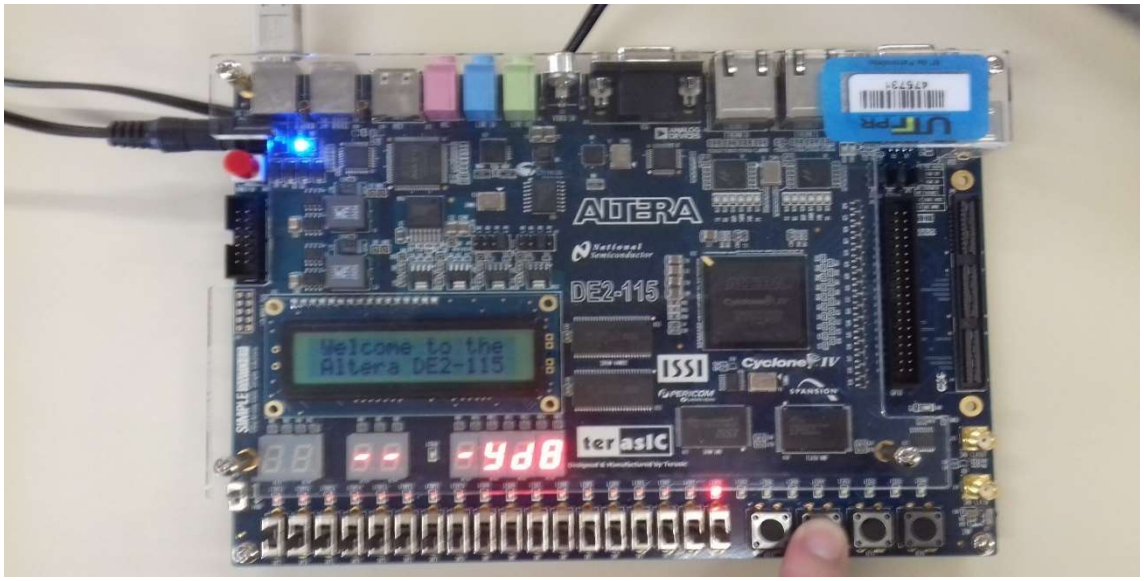


Figura 34 - Utilização da chave seletora de entrada desejada
Fonte: Autoria própria

A figura 35 tem função similar a 32, onde ilustra a localização física e a utilização da *key1* da *FPGA* que é responsável pela alteração do peso escolhido pelo usuário. Durante a leitura do valor absoluto do peso nota-se o *LedR0* da *FPGA* que indica se o valor do peso é positivo (apagado) ou negativo (aceso).

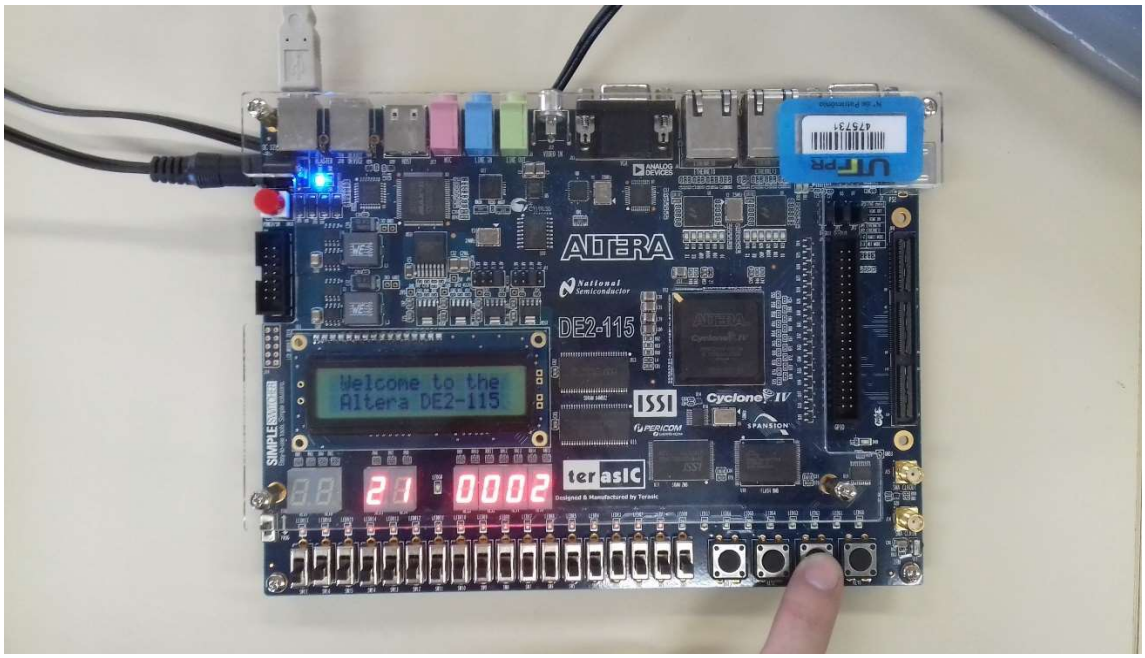


Figura 35 - Utilização da chave seletora de peso
Fonte: Autoria própria

Similar a figura 36, a figura 32 ilustra a utilização da *key1* da *FPGA* e mostra o LedR0 aceso.

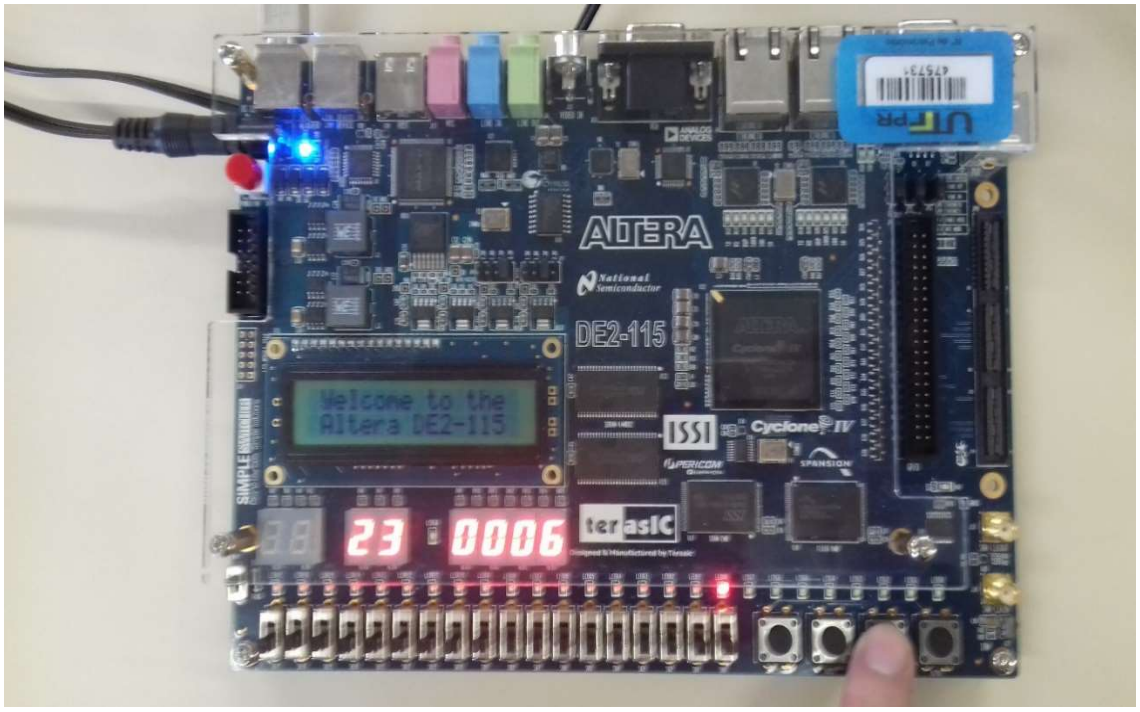


Figura 36 - Utilização da chave seletora de peso
Fonte: Autoria própria

A figura 37 ilustra a localidade física e a utilização da *key0* da *FPGA* que tem a função de checagem, que é o sinal encaminhado do supervisor à rede que a

saída desejada está configurada e está na hora de efetuar uma iteração de treinamento.

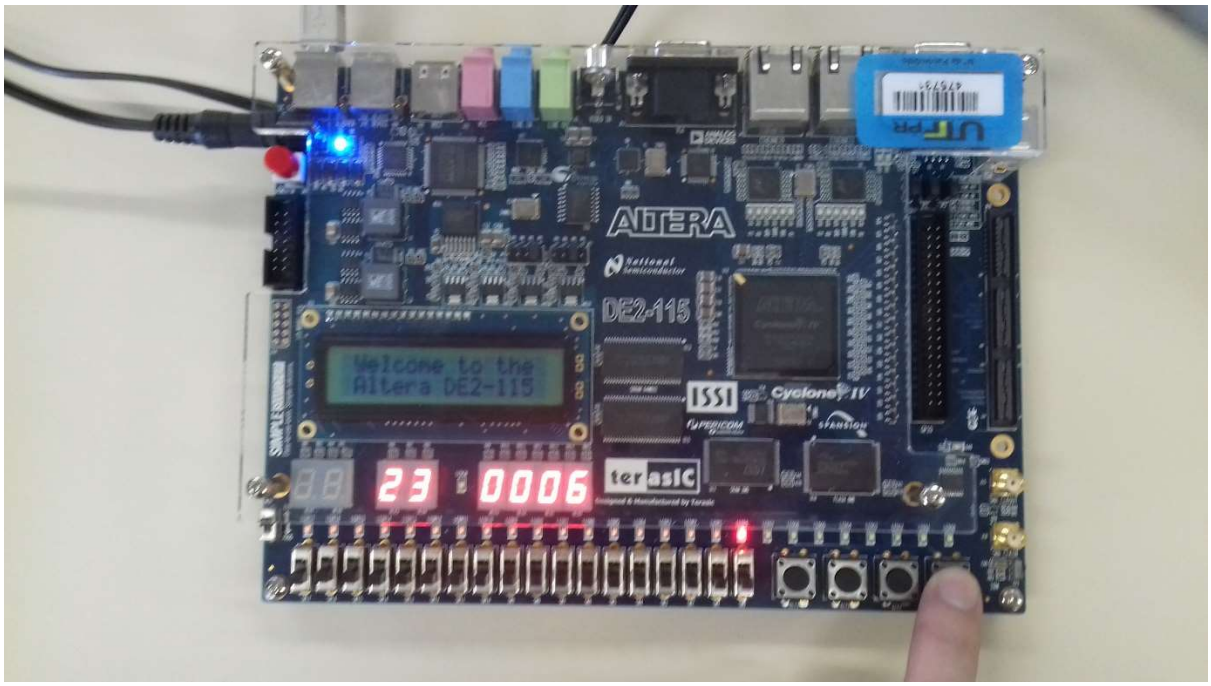


Figura 37 - Utilização da chave seletora de iteração

Fonte: Autoria própria

7. CONCLUSÃO

O principal foco deste trabalho estava na elaboração de uma RNA em *VHDL*, este objetivo foi alcançado, porém o ganho deste trabalho ficou expresso nitidamente nas etapas que não foram funcionais e o que foi feito para sua melhoria. Como o estudo dos pesos iniciais aliados à taxa de aprendizagem, a base de dados a ser utilizada para a topologia da RNA a ser criada, entre outros aspectos que teriam impactos negativos se não fossem devidamente adequados.

A elaboração de um sistema em *VHDL* para o treinamento e simulação da RNA como um todo não obteve êxito e soluções tiveram que ser tomadas, a utilização do *software Matlab* foi de suma importância para a continuação do trabalho.

Com a utilização do *Matlab* foi efetuado testes na base de dados e avaliado sua convergência ou não para a RNA que estava sendo desenvolvida, foram feitas atualizações na base de dados até garantir a convergência. Como os recursos em *VHDL* e no *FPGA* são limitados, principalmente matematicamente não se podia desenvolver uma RNA com complexidade tão grande, o que limitava a base de dados a problemas teoricamente mais simples.

Após a escolha da base de dados a utilização do *Matlab* para testar o algoritmo de treinamento utilizado neste trabalho foi de suma importância, por estar interligado com o *hardware* criado a simulação no *Quartus II* torna-se inviável para esta complexidade matemática. Com a criação de um script idêntico ao desenvolvido no *Quartus II* pode-se garantir a convergência do algoritmo de treinamento.

Com a garantia da convergência da base de dados, o script criado foi utilizado para fazer as alterações nos pesos iniciais e na taxa de aprendizado, pois estes parâmetros não podiam ser visualizados na ferramenta utilizado anteriormente chamada *nntool,* juntamente com a base de dados, a escolha dos pesos iniciais ideais melhora consideravelmente a saída do sistema e como é algo completamente imprevisível a melhor forma de escolha é através da tentativa e erro.

Como os pesos após o treinamento se tornou de fácil acesso, a criação de uma RNA no *Quartus II* foi possível, utilizando os pesos do treinamento feito em *Matlab* a simulação da RNA no *Quartus II* foi um sucesso, com apenas alguns erros de arredondamento matemático, mas isso não afetaria em nada a funcionalidade do sistema.

Para trabalhos futuros o algoritmo de treinamento e a interface entre usuário/*hardware* foi criada, possibilitando assim o treinamento direto no *FPGA.*

8. REFERÊNCIAS

COSTA, Eduardo R.; GOMES, Marcel L.; BIANCHI, Reinaldo AC. **Um mini robô móvel seguidor de pistas guiado por visão local.** VI Simpósio Brasileiro de Automação Inteligente, Bauru, Brasil, p. 175-186, 2003.

Jorge M. Barreto, **Introdução às Redes Neurais Artificiais** Laboratório de Conexionismo e Ciências Cognitivas UFSC. Disponível: http://www.inf.ufsc.br/~barreto/tutoriais/Survey.pdf Acesso: 27/04/15.

Wolf, D. F., do Valle Simões, E., Osório, F. S., & Junior, O. T. (2009). **Robótica móvel inteligente: Da simulação às aplicações no mundo real**. , *Mini-Curso: Jornada de Atualização em Informática (JAI), Congresso da SBC*.

PEDRONI, Volnei A. **Eletrônica digital moderna e VHDL**. Rio de Janeiro, RJ: Elsevier, 2010

Diego Felipe, Sánchez Gómez, **Implementação em VHDL de uma biblioteca parametrizável de operadores aritméticos em ponto flutuante para ser utilizada em problemas de robótica**, Dissertação de mestrado Brasilia, 2009.

FPGAs substituindo microcontroladores simples. Disponível : http://www.embarcados.com.br/fpgas-substituindo-microcontroladores-simples/ Acesso: 27/04/15.

BRAGA, Antônio de Pádua; CARVALHO, André Carlos Ponce de Leon Ferreira; LUDERMIR, Teresa Bernarda. **Redes neurais artificiais: teoria e aplicações.** 2ª ed Rio de Janeiro, 2000.

HAYKIN, Simon. **Redes neurais: princípios e prática.** 2. ed. Porto Alegre, 2009.

Humberto Alejandro Secchi. **Una Introducción a los Robots Móviles**. 2008. Disponível em http://www.aadeca.org/pdf/CP_monografias/monografia_robot_movil.pdf Acesso: 02/05/15.

Heinen, Milton Roberto. **Controle inteligente do caminhar de robôs móveis simulados.** Dissertação de mestrado Universidade do Vale do Rio dos Sinos (UNIsInOs), São Leopoldo, RS, Brasil 2007.

ADC0804 Datasheet (PDF) - National Semiconductor. Disponível em: http://pdf1.alldatasheet.com/datasheet-pdf/view/8105/NSC/ADC0804.html. Acesso em: 2016-02-02.

SG90 9 g Micro Servo Datasheet. Disponível em: http://www.micropik.com/PDF/SG90Servo.pdf. Acesso em 2016-02-02.

9. APÊNDICE

Quadro 1 – Banco de dados para 20 iterações

$x_1(esquerda)$	$x_2(centro)$	$x_3(direita)$	y_d
55	110	0	9
100	50	240	2
150	200	50	7
133	205	33	7
20	180	120	2
0	180	200	1
15	150	150	4
200	100	50	8
100	150	50	7
200	250	200	5
100	220	150	4
140	190	30	6
100	150	100	5
120	150	80	7
140	240	150	5
120	0	255	2
30	250	70	3
200	150	0	9
100	0	20	7
50	160	150	3

Quadro 2 – Banco de dados para 50 iterações

$x_1(esquerda)$	$x_2(centro)$	$x_3(direita)$	y_d
55	110	0	9
100	50	240	2
150	200	50	7
133	205	33	7
20	180	120	2
0	180	200	1
15	150	150	4
200	100	50	8
100	150	50	7
200	250	200	5
100	220	150	4
140	190	30	6
100	150	100	5
120	150	80	7
140	240	150	5
120	0	255	2

30	250	70	3
200	150	0	9
100	0	20	7
50	160	150	3
30	160	220	2
50	210	100	4
0	200	200	1
55	255	44	5
30	150	200	2
100	255	0	8
0	255	0	5
30	180	80	3
100	150	0	9
30	180	110	2
30	50	0	7
100	250	100	5
55	205	255	3
20	150	200	2
60	190	0	8
22	44	66	2
33	66	99	2
45	160	220	1
45	160	110	3
60	210	60	5
60	150	180	2
60	30	220	1
60	30	0	9
60	190	0	7
180	250	130	6
180	250	180	5
180	250	220	4
180	250	250	3
180	70	130	7
0	70	250	1

Quadro 3 – Banco de dados para 100 iterações

$x_1(esquerda)$	$x_2(centro)$	$x_3(direita)$	y_d
25	190	135	2
45	130	180	2
80	230	140	4
30	110	220	1
10	200	110	2
110	220	200	3

140	240	140	5
160	230	130	6
160	170	130	7
170	110	70	8
200	60	30	9
25	230	25	5
25	230	100	4
25	230	200	3
25	150	150	2
25	100	200	1
100	230	25	7
100	230	60	6
100	230	100	5
100	230	130	4
100	230	200	3
100	190	25	8
100	190	60	6
100	190	100	5
100	190	200	3
100	150	25	8
100	150	60	6
100	150	100	5
100	150	200	3
100	100	25	8
100	100	60	7
100	100	200	3
150	30	25	9
150	30	60	8
200	30	25	9
200	30	60	8
200	120	100	7
10	100	200	1
10	160	200	2
50	140	200	3
70	50	200	2
90	160	200	3
90	240	220	3
110	240	200	4
140	220	160	5
200	230	100	7
200	230	160	5
200	150	150	6
220	140	100	7
220	50	40	9
220	70	10	9

220	70	100	7
220	120	50	8
130	120	10	8
100	100	200	3
120	230	120	5
140	220	140	5
140	230	80	6
110	220	90	5
200	240	200	5
230	150	10	9
100	210	200	3
60	210	200	2
100	140	200	4
10	150	200	1
30	50	100	1
10	50	90	1
20	100	150	2
20	50	200	1
200	100	10	9
200	50	20	9
150	50	10	9
100	50	30	8
30	150	150	2
30	190	150	3
45	130	180	2
100	230	160	4
40	90	230	1
20	180	140	2
120	220	250	3
120	240	120	5
160	230	130	6
150	150	30	8
150	170	30	7
180	130	40	8
160	230	100	6
230	90	40	9
140	180	20	8
250	220	120	7
180	230	180	5
160	230	130	6
100	120	10	8
200	120	10	9
150	100	20	8
130	70	20	8
130	140	50	7

150	220	100	6
130	210	110	5
100	230	80	6
120	220	150	4

Quadro 4a – Saída do sistema no *Matlab*

x1	x2	x3	Yd	10 it.	erro	1000 it.	erro
25	190	135	13	25,04	-12,04	21,71	-8,71
45	130	180	16	30,72	-14,72	26,79	-10,79
80	230	140	7	26,91	-19,91	23,25	-16,25
30	110	220	23	36,38	-13,38	31,81	-8,81
10	200	110	12	21,35	-9,35	18,46	-6,46
110	220	200	11	36,04	-25,04	31,27	-20,27
140	240	140	0	27,37	-27,37	23,59	-23,59
160	230	130	-4	25,69	-29,69	22,12	-26,12
160	170	130	-4	24,37	-28,37	21,06	-25,06
170	110	70	-12	13,86	-25,86	11,90	-23,90
200	60	30	-20	6,73	-26,73	5,68	-25,68
25	230	25	0	9,00	-9,00	7,53	-7,53
25	230	100	9	20,53	-11,53	17,68	-8,68
25	230	200	21	35,92	-14,92	31,21	-10,21
25	150	150	15	26,47	-11,47	23,03	-8,03
25	100	200	21	33,06	-12,06	28,91	-7,91
100	230	25	-9	9,30	-18,30	7,74	-16,74
100	230	60	-5	14,68	-19,68	12,48	-17,48
100	230	100	0	20,83	-20,83	17,89	-17,89
100	230	130	4	25,45	-21,45	21,95	-17,95
100	230	200	12	36,22	-24,22	31,42	-19,42
100	190	25	-9	8,42	-17,42	7,03	-16,03
100	190	60	-5	13,80	-18,80	11,77	-16,77
100	190	100	0	19,95	-19,95	17,18	-17,18
100	190	200	12	35,34	-23,34	30,71	-18,71
100	150	25	-9	7,54	-16,54	6,32	-15,32
100	150	60	-5	12,92	-17,92	11,06	-16,06
100	150	100	0	19,08	-19,08	16,47	-16,47
100	150	200	2	34,46	-32,46	30,01	-28,01
100	100	25	-9	6,44	-15,44	5,44	-14,44
100	100	60	-5	11,82	-16,82	10,17	-15,17
100	100	250	12	33,36	-21,36	29,12	-17,12
150	30	25	-15	5,10	-20,10	4,34	-19,34
150	30	60	-11	10,49	-21,49	9,07	-20,07
200	30	25	-21	5,30	-26,30	4,48	-25,48
200	30	60	-17	10,69	-27,69	9,21	-26,21

200	120	100	-12	18,82	-30,82	16,22	-28,22
10	100	200	23	33,00	-10,00	28,87	-5,87
10	160	200	23	34,32	-11,32	29,93	-6,93
50	140	200	18	34,04	-16,04	29,69	-11,69
70	50	200	16	32,14	-16,14	28,15	-12,15
90	160	200	13	34,64	-21,64	30,15	-17,15
90	240	220	16	39,47	-23,47	34,28	-18,28
110	240	200	11	36,47	-25,47	31,63	-20,63
140	220	160	2	30,00	-28,00	25,94	-23,94
200	230	100	-12	21,23	-33,23	18,17	-30,17
200	230	160	-5	30,46	-35,46	26,29	-31,29
200	150	150	-6	27,17	-33,17	23,52	-29,52
220	140	100	-14	19,33	-33,33	16,63	-30,63
220	50	40	-22	8,13	-30,13	6,92	-28,92
220	70	10	-25	3,95	-28,95	3,21	-28,21
220	70	100	-14	17,80	-31,80	15,39	-29,39
220	120	50	-20	11,20	-31,20	9,51	-29,51
130	120	10	-14	4,69	-18,69	3,84	-17,84
100	100	200	12	33,36	-21,36	29,12	-17,12
120	230	120	0	23,99	-23,99	20,65	-20,65
140	220	140	0	26,92	-26,92	23,24	-23,24
140	230	80	-7	17,91	-24,91	15,29	-22,29
110	220	90	-2	19,11	-21,11	16,39	-18,39
200	240	200	0	36,83	-36,83	31,88	-31,88
230	150	10	-26	5,75	-31,75	4,66	-30,66
100	210	200	12	35,77	-23,77	31,07	-19,07
60	210	200	17	35,62	-18,62	30,95	-13,95
100	140	200	12	34,24	-22,24	29,83	-17,83
10	150	200	23	34,10	-11,10	29,75	-6,75
30	50	100	8	16,60	-8,60	14,50	-6,50
10	50	90	10	14,98	-4,98	13,09	-3,09
20	100	150	16	25,35	-9,35	22,13	-6,13
20	50	200	22	31,94	-9,94	28,01	-6,01
200	100	10	-23	4,53	-27,53	3,69	-26,69
200	50	20	-22	4,97	-26,97	4,15	-26,15
150	50	10	-17	3,23	-20,23	2,66	-19,66
100	50	30	-8	6,11	-14,11	5,23	-13,23
30	150	150	14	26,49	-12,49	23,04	-9,04
30	190	150	14	27,36	-13,36	23,75	-9,75
45	130	180	16	30,72	-14,72	26,79	-10,79
100	230	160	7	30,06	-23,06	26,01	-19,01
40	90	230	23	37,51	-14,51	32,83	-9,83
20	180	140	14	25,57	-11,57	22,19	-8,19
120	220	250	16	43,76	-27,76	38,07	-22,07
120	240	120	0	24,21	-24,21	20,83	-20,83

160	230	130	-4	25,69	-29,69	22,12	-26,12
150	150	30	-14	8,51	-22,51	7,14	-21,14
150	170	30	-14	8,95	-22,95	7,49	-21,49
180	130	40	-17	9,73	-26,73	8,22	-25,22
160	230	100	-7	21,07	-28,07	18,06	-25,06
230	90	40	-23	9,05	-32,05	7,65	-30,65
140	180	20	-14	7,59	-21,59	6,29	-20,29
250	220	120	-16	24,29	-40,29	20,84	-36,84
180	230	180	0	33,46	-33,46	28,94	-28,94
160	230	130	-4	25,68	-29,68	22,12	-26,12
100	120	10	-11	4,57	-15,57	3,76	-14,76
200	120	10	-23	4,97	-27,97	4,04	-27,04
150	100	20	-16	5,87	-21,87	4,90	-20,90
130	70	20	-13	5,13	-18,13	4,31	-17,31
130	140	50	-10	11,28	-21,28	9,61	-19,61
150	220	100	-6	20,81	-26,81	17,85	-23,85
130	210	110	-2	22,05	-24,05	18,97	-20,97
100	230	80	-2	17,75	-19,75	15,18	-17,18
120	220	150	4	28,38	-24,38	24,53	-20,53

Quadro 4b – Saída do sistema no *Matlab*

x1	x2	x3	Yd	10 it	erro	1000 it.	erro	1000000 it.	erro
25	190	135	13	10,08	2,92	13,29	-0,29	13,29	-0,29
45	130	180	16	12,43	3,57	16,28	-0,28	16,29	-0,29
80	230	140	7	5,34	1,66	7,27	-0,27	7,27	-0,27
30	110	220	23	18,11	4,89	22,86	0,14	22,86	0,14
10	200	110	12	9,91	2,09	12,05	-0,05	12,04	-0,04
110	220	200	11	8,98	2,02	10,84	0,16	10,84	0,16
140	240	140	0	0,57	-0,57	0,02	-0,02	0,02	-0,02
160	230	130	-4	-2,41	-1,59	-3,59	-0,41	-3,59	-0,41
160	170	130	-4	-2,72	-1,28	-3,60	-0,40	-3,60	-0,40
170	110	70	-12	-9,93	-2,07	-12,09	0,09	-12,09	0,09
200	60	30	-20	-17,22	-2,78	-20,51	0,51	-20,50	0,50
25	230	25	0	0,59	-0,59	0,00	0,00	-0,01	0,01
25	230	100	9	7,75	1,25	9,02	-0,02	9,02	-0,02
25	230	200	21	17,56	3,44	21,05	-0,05	21,05	-0,05
25	150	150	15	12,58	2,42	15,03	-0,03	15,03	-0,03
25	100	200	21	17,57	3,43	21,04	-0,04	21,04	-0,04
100	230	25	-9	-6,89	-2,11	-9,05	0,05	-9,05	0,05
100	230	60	-5	-3,44	-1,56	-4,84	-0,16	-4,84	-0,16
100	230	100	0	0,36	-0,36	-0,03	0,03	-0,03	0,03
100	230	130	4	3,21	0,79	3,58	0,42	3,58	0,42
100	230	200	12	10,21	1,79	12,00	0,00	12,00	0,00
100	190	25	-9	-7,32	-1,68	-9,04	0,04	-9,05	0,05
100	190	60	-5	-3,84	-1,16	-4,83	-0,17	-4,84	-0,16
100	190	100	0	0,06	-0,06	-0,02	0,02	-0,03	0,03
100	190	200	12	10,03	1,97	12,00	0,00	12,00	0,00
100	150	25	-9	-7,64	-1,36	-9,04	0,04	-9,05	0,05
100	150	60	-5	-4,15	-0,85	-4,83	-0,17	-4,84	-0,16
100	150	100	0	-0,19	0,19	-0,03	0,03	-0,02	0,02
100	150	200	2	9,89	-7,89	12,00	-10,00	12,00	-10,00
100	100	25	-9	-7,92	-1,08	-9,16	0,16	-9,16	0,16
100	100	60	-5	-4,66	-0,34	-4,83	-0,17	-4,83	-0,17
100	100	250	12	9,67	2,33	12,00	0,00	12,01	-0,01
150	30	25	-15	-13,74	-1,26	-15,07	0,07	-15,08	0,08
150	30	60	-11	-10,23	-0,77	-10,86	-0,14	-10,87	-0,13
200	30	25	-21	-19,46	-1,54	-21,10	0,10	-21,12	0,12
200	30	60	-17	-15,99	-1,01	-16,88	-0,12	-16,90	-0,10

200	120	100	-12	-11,99	-0,01	-12,08	0,08	-12,10	0,10
10	100	200	23	19,72	3,28	22,85	0,15	22,84	0,16
10	160	200	23	19,74	3,26	22,85	0,15	22,85	0,15
50	140	200	18	15,54	2,46	18,05	-0,05	18,04	-0,04
70	50	200	16	13,41	2,59	15,63	0,37	15,63	0,37
90	160	200	13	11,56	1,44	13,23	-0,23	13,22	-0,22
90	240	220	16	14,12	1,88	15,64	0,36	15,63	0,37
110	240	200	11	10,02	0,98	10,82	0,18	10,81	0,19
140	220	160	2	2,72	-0,72	2,45	-0,45	2,44	-0,44
200	230	100	-12	-10,13	-1,87	-12,00	0,00	-12,02	0,02
200	230	160	-5	-3,88	-1,12	-4,77	-0,23	-4,79	-0,21
200	150	150	-6	-5,54	-0,46	-5,99	-0,01	-6,00	0,00
220	140	100	-14	-13,36	-0,64	-14,41	0,41	-14,43	0,43
220	50	40	-22	-20,16	-1,84	-21,63	-0,37	-21,64	-0,36
220	70	10	-25	-23,56	-1,44	-25,28	0,28	-25,29	0,29
220	70	100	-14	-14,07	0,07	-14,41	0,41	-14,42	0,42
220	120	50	-20	-19,35	-0,65	-20,35	0,35	-20,36	0,36
130	120	10	-14	-13,47	-0,53	-14,31	0,31	-14,32	0,32
100	100	200	12	10,66	1,34	12,06	-0,06	12,06	-0,06
120	230	120	0	-0,09	0,09	0,13	-0,13	0,12	-0,12
140	220	140	0	-0,04	0,04	0,13	-0,13	0,12	-0,12
140	230	80	-7	-6,56	-0,44	-7,08	0,08	-7,09	0,09
110	220	90	-2	-2,08	0,08	-2,30	0,30	-2,30	0,30
200	240	200	0	-0,24	0,24	0,15	-0,15	0,14	-0,14
230	150	10	-26	-24,61	-1,39	-26,25	0,25	-26,26	0,26
100	210	200	12	11,07	0,93	12,09	-0,09	12,08	-0,08
60	210	200	17	15,59	1,41	16,87	0,13	16,87	0,13
100	140	200	12	10,94	1,06	12,07	-0,07	12,06	-0,06
10	150	200	23	21,25	1,75	22,83	0,17	22,82	0,18
30	50	100	8	7,77	0,23	8,42	-0,42	8,42	-0,42
10	50	90	10	8,95	1,05	9,61	0,39	9,61	0,39
20	100	150	16	14,61	1,39	15,62	0,38	15,62	0,38
20	50	200	22	19,97	2,03	21,62	0,38	21,62	0,38
200	100	10	-23	-20,85	-2,15	-22,62	-0,38	-22,63	-0,37
200	50	20	-22	-20,15	-1,85	-21,48	-0,52	-21,49	-0,51
150	50	10	-17	-15,87	-1,13	-16,76	-0,24	-16,76	-0,24
100	50	30	-8	-7,99	-0,01	-8,38	0,38	-8,39	0,39
30	150	150	14	13,95	0,05	14,52	-0,52	14,51	-0,51

30	190	150	14	14,09	-0,09	14,53	-0,53	14,53	-0,53
45	130	180	16	15,52	0,48	16,29	-0,29	16,29	-0,29
100	230	160	7	7,28	-0,28	7,24	-0,24	7,23	-0,23
40	90	230	23	21,57	1,43	22,85	0,15	22,85	0,15
20	180	140	14	14,08	-0,08	14,43	-0,43	14,43	-0,43
120	220	250	16	15,06	0,94	15,63	0,37	15,62	0,38
120	240	120	0	0,52	-0,52	-0,02	0,02	-0,03	0,03
160	230	130	-4	-3,01	-0,99	-3,64	-0,36	-3,58	-0,42
150	150	30	-14	-13,49	-0,51	-14,47	0,47	-14,42	0,42
150	170	30	-14	-13,53	-0,47	-14,45	0,45	-14,40	0,40
180	130	40	-17	-16,10	-0,90	-16,79	-0,21	-16,74	-0,26
160	230	100	-7	-6,72	-0,28	-7,19	0,19	-7,14	0,14
230	90	40	-23	-22,32	-0,68	-22,80	-0,20	-22,76	-0,24
140	180	20	-14	-13,78	-0,22	-14,40	0,40	-14,36	0,36
250	220	120	-16	-15,40	-0,60	-15,57	-0,43	-15,51	-0,49
180	230	180	0	-0,28	0,28	0,02	-0,02	0,07	-0,07
160	230	130	-4	-3,60	-0,40	-3,58	-0,42	-3,62	-0,38
100	120	10	-11	-10,44	-0,56	-10,81	-0,19	-10,84	-0,16
200	120	10	-23	-22,38	-0,62	-22,87	-0,13	-22,93	-0,07
150	100	20	-16	-15,41	-0,59	-15,67	-0,33	-15,71	-0,29
130	70	20	-13	-13,15	0,15	-13,28	0,28	-13,30	0,30
130	140	50	-10	-9,53	-0,47	-9,66	-0,34	-9,69	-0,31
150	220	100	-6	-6,03	0,03	-6,06	0,06	-6,10	0,10
130	210	110	-2	-2,50	0,50	-2,44	0,44	-2,47	0,47
100	230	80	-2	-2,29	0,29	-2,43	0,43	-2,45	0,45
120	220	150	4	3,36	0,64	3,63	0,37	3,61	0,39

Quadro 4c – Saída do sistema no *Matlab*

x1	x2	x3	Yd	10 it.	erro	1000 it.	erro	1000000 it.	erro
25	190	135	13	-2,49E-05	13,00	-2,45E-05	13,00	13,34	-0,34
45	130	180	16	-3,04E-05	16,00	-2,99E-05	16,00	16,02	-0,02
80	230	140	7	-2,63E-05	7,00	-2,59E-05	7,00	7,33	-0,33
30	110	220	23	-3,62E-05	23,00	-3,56E-05	23,00	22,50	0,50
10	200	110	12	-2,13E-05	12,00	-2,10E-05	12,00	12,26	-0,26
110	220	200	11	-3,52E-05	11,00	-3,46E-05	11,00	10,69	0,31
140	240	140	0	-2,63E-05	0,00	-2,58E-05	0,00	0,03	-0,03
160	230	130	-4	-2,44E-05	-4,00	-2,40E-05	-4,00	-3,61	-0,39
160	170	130	-4	-2,31E-05	-4,00	-2,27E-05	-4,00	-3,78	-0,22

170	110	70	-12	-1,25E-05	-12,00	-1,23E-05	-12,00	-12,22	0,22
200	60	30	-20	-5,14E-06	-20,00	-4,99E-06	-20,00	-20,72	0,72
25	230	25	0	-8,81E-06	0,00	-8,71E-06	0,00	0,52	-0,52
25	230	100	9	-2,04E-05	9,00	-2,01E-05	9,00	9,34	-0,34
25	230	200	21	-3,58E-05	21,00	-3,52E-05	21,00	21,09	-0,09
25	150	150	15	-2,63E-05	15,00	-2,59E-05	15,00	14,99	0,01
25	100	200	21	-3,29E-05	21,00	-3,23E-05	21,00	20,73	0,27
100	230	25	-9	-8,51E-06	-9,00	-8,39E-06	-9,00	-8,63	-0,37
100	230	60	-5	-1,39E-05	-5,00	-1,37E-05	-5,00	-4,52	-0,48
100	230	100	0	-2,01E-05	0,00	-1,97E-05	0,00	0,18	-0,18
100	230	130	4	-2,47E-05	4,00	-2,43E-05	4,00	3,71	0,29
100	230	200	12	-3,55E-05	12,00	-3,49E-05	12,00	11,94	0,06
100	190	25	-9	-7,63E-06	-9,00	-7,52E-06	-9,00	-8,74	-0,26
100	190	60	-5	-1,30E-05	-5,00	-1,28E-05	-5,00	-4,63	-0,37
100	190	100	0	-1,92E-05	0,00	-1,89E-05	0,00	0,07	-0,07
100	190	200	12	-3,46E-05	12,00	-3,40E-05	12,00	11,83	0,17
100	150	25	-9	-6,75E-06	-9,00	-6,64E-06	-9,00	-8,85	-0,15
100	150	60	-5	-1,21E-05	-5,00	-1,19E-05	-5,00	-4,74	-0,26
100	150	100	0	-1,83E-05	0,00	-1,80E-05	0,00	-0,04	0,04
100	150	200	2	-3,37E-05	2,00	-3,31E-05	2,00	11,71	-9,71
100	100	25	-9	-5,65E-06	-9,00	-5,54E-06	-9,00	-8,99	-0,01
100	100	60	-5	-1,10E-05	-5,00	-1,08E-05	-5,00	-4,88	-0,12
100	100	250	12	-3,26E-05	12,00	-3,20E-05	12,00	11,58	0,42
150	30	25	-15	-3,91E-06	-15,00	-3,80E-06	-15,00	-15,28	0,28
150	30	60	-11	-9,30E-06	-11,00	-9,09E-06	-11,00	-11,17	0,17
200	30	25	-21	-3,71E-06	-21,00	-3,58E-06	-21,00	-21,39	0,39
200	30	60	-17	-9,10E-06	-17,00	-8,88E-06	-17,00	-17,27	0,27
200	120	100	-12	-1,72E-05	-12,00	-1,69E-05	-12,00	-12,32	0,32
10	100	200	23	-3,30E-05	23,00	-3,24E-05	23,00	22,56	0,44
10	160	200	23	-3,43E-05	23,00	-3,37E-05	23,00	22,73	0,27
50	140	200	18	-3,37E-05	18,00	-3,31E-05	18,00	17,79	0,21
70	50	200	16	-3,16E-05	16,00	-3,11E-05	16,00	15,10	0,90
90	160	200	13	-3,40E-05	13,00	-3,34E-05	13,00	12,96	0,04
90	240	220	16	-3,88E-05	16,00	-3,82E-05	16,00	15,53	0,47
110	240	200	11	-3,56E-05	11,00	-3,50E-05	11,00	10,74	0,26
140	220	160	2	-2,89E-05	2,00	-2,84E-05	2,00	2,33	-0,33
200	230	100	-12	-1,97E-05	-12,00	-1,93E-05	-12,00	-12,02	0,02
200	230	160	-5	-2,89E-05	-5,00	-2,84E-05	-5,00	-4,97	-0,03
200	150	150	-6	-2,56E-05	-6,00	-2,51E-05	-6,00	-6,36	0,36
220	140	100	-14	-1,76E-05	-14,00	-1,73E-05	-14,00	-14,71	0,71

220	50	40	-22	-6,38E-06	-22,00	-6,20E-06	-22,00	-22,01	0,01
220	70	10	-25	-2,20E-06	-25,00	-2,10E-06	-25,00	-25,48	0,48
220	70	100	-14	-1,61E-05	-14,00	-1,57E-05	-14,00	-14,90	0,90
220	120	50	-20	-9,46E-06	-20,00	-9,25E-06	-20,00	-20,64	0,64
130	120	10	-14	-3,66E-06	-14,00	-3,58E-06	-14,00	-14,36	0,36
100	100	200	12	-3,26E-05	12,00	-3,20E-05	12,00	11,58	0,42
120	230	120	0	-2,31E-05	0,00	-2,27E-05	0,00	0,09	-0,09
140	220	140	0	-2,58E-05	0,00	-2,54E-05	0,00	-0,02	0,02
140	230	80	-7	-1,68E-05	-7,00	-1,65E-05	-7,00	-7,05	0,05
110	220	90	-2	-1,83E-05	-2,00	-1,80E-05	-2,00	-2,24	0,24
200	240	200	0	-3,53E-05	0,00	-3,47E-05	0,00	-0,24	0,24
230	150	10	-26	-3,92E-06	-26,00	-3,81E-06	-26,00	-26,48	0,48
100	210	200	12	-3,50E-05	12,00	-3,44E-05	12,00	11,88	0,12
60	210	200	17	-3,52E-05	17,00	-3,46E-05	17,00	16,76	0,24
100	140	200	12	-3,35E-05	12,00	-3,29E-05	12,00	11,69	0,31
10	150	200	23	-3,41E-05	23,00	-3,35E-05	23,00	22,70	0,30
30	50	100	8	-1,64E-05	8,00	-1,61E-05	8,00	8,23	-0,23
10	50	90	10	-1,49E-05	10,00	-1,47E-05	10,00	9,49	0,51
20	100	150	16	-2,52E-05	16,00	-2,48E-05	16,00	15,46	0,54
20	50	200	22	-3,18E-05	22,00	-3,13E-05	22,00	21,20	0,80
200	100	10	-23	-2,94E-06	-23,00	-2,84E-06	-23,00	-22,95	-0,05
200	50	20	-22	-3,38E-06	-22,00	-3,26E-06	-22,00	-21,92	-0,08
150	50	10	-17	-2,04E-06	-17,00	-1,96E-06	-17,00	-16,99	-0,01
100	50	30	-8	-5,32E-06	-8,00	-5,20E-06	-8,00	-8,54	0,54
30	150	150	14	-2,63E-05	14,00	-2,59E-05	14,00	14,38	-0,38
30	190	150	14	-2,72E-05	14,00	-2,67E-05	14,00	14,49	-0,49
45	130	180	16	-3,04E-05	16,00	-2,99E-05	16,00	16,02	-0,02
100	230	160	7	-2,93E-05	7,00	-2,88E-05	7,00	7,24	-0,24
40	90	230	23	-3,72E-05	23,00	-3,66E-05	23,00	22,40	0,60
20	180	140	14	-2,54E-05	14,00	-2,50E-05	14,00	14,51	-0,51
120	220	250	16	-4,29E-05	16,00	-4,21E-05	16,00	15,34	0,66
120	240	120	0	-2,33E-05	0,00	-2,29E-05	0,00	0,12	-0,12
160	230	130	-4	-2,44E-05	-4,00	-2,40E-05	-4,00	-3,61	-0,39
150	150	30	-14	-7,32E-06	-14,00	-7,18E-06	-14,00	-14,36	0,36
150	170	30	-14	-7,76E-06	-14,00	-7,62E-06	-14,00	-14,31	0,31
180	130	40	-17	-8,30E-06	-17,00	-8,13E-06	-17,00	-16,91	-0,09
160	230	100	-7	-1,98E-05	-7,00	-1,95E-05	-7,00	-7,14	0,14
230	90	40	-23	-7,22E-06	-23,00	-7,04E-06	-23,00	-23,12	0,12

140	180	20	-14	-6,48E-06	-14,00	-6,37E-06	-14,00	-14,24	0,24
250	220	120	-16	-2,23E-05	-16,00	-2,19E-05	-16,00	-15,80	-0,20
180	230	180	0	-3,21E-05	0,00	-3,15E-05	0,00	-0,18	0,18
160	230	130	-4	-2,44E-05	-4,00	-2,40E-05	-4,00	-3,61	-0,39
100	120	10	-11	-3,78E-06	-11,00	-3,71E-06	-11,00	-10,70	-0,30
200	120	10	-23	-3,38E-06	-23,00	-3,28E-06	-23,00	-22,90	-0,10
150	100	20	-16	-4,68E-06	-16,00	-4,57E-06	-16,00	-15,68	-0,32
130	70	20	-13	-4,10E-06	-13,00	-4,00E-06	-13,00	-13,32	0,32
130	140	50	-10	-1,03E-05	-10,00	-1,01E-05	-10,00	-9,60	-0,40
150	220	100	-6	-1,96E-05	-6,00	-1,93E-05	-6,00	-5,95	-0,05
130	210	110	-2	-2,10E-05	-2,00	-2,07E-05	-2,00	-2,36	0,36
100	230	80	-2	-1,70E-05	-2,00	-1,67E-05	-2,00	-2,17	0,17
120	220	150	4	-2,75E-05	4,00	-2,70E-05	4,00	3,59	0,41

Quadro 5 – Comparação entre taxas de aprendizado

x1	x2	x3	Yd	10 it.	1000 it.	1000000 it.	10 it.	1000 it.	1000000 it.
			Taxa = 0.00000000005				Taxa = 0.00005		
25	190	135	13	13,00	13,00	-0,34	-0,47	-0,46	-0,46
45	130	180	16	16,00	16,00	-0,02	-0,37	-0,36	-0,36
80	230	140	7	7,00	7,00	-0,33	-0,18	-0,19	-0,19
30	110	220	23	23,00	23,00	0,50	0,40	0,40	0,40
10	200	110	12	12,00	12,00	-0,26	0,15	0,16	0,16
110	220	200	11	11,00	11,00	0,31	0,28	0,27	0,27
140	240	140	0	0,00	0,00	-0,03	-0,29	-0,31	-0,31
160	230	130	-4	-4,00	-4,00	-0,39	-0,74	-0,76	-0,76
160	170	130	-4	-4,00	-4,00	-0,22	-0,29	-0,28	-0,28
170	110	70	-12	-12,00	-12,00	0,22	0,06	0,06	0,06
200	60	30	-20	-20,00	-20,00	0,72	0,29	0,29	0,29
25	230	25	0	0,00	0,00	-0,52	0,14	0,15	0,15
25	230	100	9	9,00	9,00	-0,34	-0,03	-0,02	-0,02
25	230	200	21	21,00	21,00	-0,09	0,04	0,05	0,05
25	150	150	15	15,00	15,00	0,01	0,03	0,04	0,04
25	100	200	21	21,00	21,00	0,27	0,07	0,07	0,07
100	230	25	-9	-9,00	-9,00	-0,37	-0,27	-0,27	-0,27
100	230	60	-5	-5,00	-5,00	-0,48	-0,43	-0,43	-0,43
100	230	100	0	0,00	0,00	-0,18	0,07	0,07	0,07
100	230	130	4	4,00	4,00	0,29	0,48	0,48	0,48
100	230	200	12	12,00	12,00	0,06	0,08	0,08	0,08
100	190	25	-9	-9,00	-9,00	-0,26	-0,04	-0,04	-0,04
100	190	60	-5	-5,00	-5,00	-0,37	-0,22	-0,22	-0,22
100	190	100	0	0,00	0,00	-0,07	0,01	0,01	0,01
100	190	200	12	12,00	12,00	0,17	0,03	0,02	0,02
100	150	25	-9	-9,00	-9,00	-0,15	-0,02	-0,02	-0,02
100	150	60	-5	-5,00	-5,00	-0,26	-0,21	-0,21	-0,21
100	150	100	0	0,00	0,00	0,04	0,00	0,00	0,00
100	150	200	2	2,00	2,00	-9,71	-9,99	-10,00	-10,00
100	100	25	-9	-9,00	-9,00	-0,01	1,13	1,13	1,13
100	100	60	-5	-5,00	-5,00	-0,12	-0,28	-0,29	-0,29
100	100	250	12	12,00	12,00	0,42	-0,06	-0,06	-0,06
150	30	25	-15	-15,00	-15,00	0,28	-0,05	-0,05	-0,05
150	30	60	-11	-11,00	-11,00	0,17	-0,21	-0,21	-0,21
200	30	25	-21	-21,00	-21,00	0,39	0,16	0,16	0,16
200	30	60	-17	-17,00	-17,00	0,27	-0,24	-0,24	-0,24
200	120	100	-12	-12,00	-12,00	0,32	0,12	0,13	0,13

10	100	200	23	23,00	23,00	0,44	0,17	0,16	0,16
10	160	200	23	23,00	23,00	0,27	0,09	0,08	0,08
50	140	200	18	18,00	18,00	0,21	-0,25	-0,25	-0,25
70	50	200	16	16,00	16,00	0,90	0,12	0,12	0,12
90	160	200	13	13,00	13,00	0,04	-0,44	-0,44	-0,44
90	240	220	16	16,00	16,00	0,47	0,01	0,02	0,02
110	240	200	11	11,00	11,00	0,26	-0,07	-0,07	-0,07
140	220	160	2	2,00	2,00	-0,33	-0,63	-0,62	-0,62
200	230	100	-12	-12,00	-12,00	0,02	0,18	0,18	0,18
200	230	160	-5	-5,00	-5,00	-0,03	-0,31	-0,31	-0,31
200	150	150	-6	-6,00	-6,00	0,36	-0,03	-0,04	-0,04
220	140	100	-14	-14,00	-14,00	0,71	0,41	0,41	0,41
220	50	40	-22	-22,00	-22,00	0,01	-0,58	-0,58	-0,58
220	70	10	-25	-25,00	-25,00	0,48	0,72	0,72	0,72
220	70	100	-14	-14,00	-14,00	0,90	0,11	0,10	0,10
220	120	50	-20	-20,00	-20,00	0,64	0,40	0,40	0,40
130	120	10	-14	-14,00	-14,00	0,36	0,21	0,22	0,22
100	100	200	12	12,00	12,00	0,42	-0,17	-0,17	-0,17
120	230	120	0	0,00	0,00	-0,09	-0,45	-0,44	-0,44
140	220	140	0	0,00	0,00	0,02	-0,27	-0,26	-0,26
140	230	80	-7	-7,00	-7,00	0,05	-0,07	-0,06	-0,06
110	220	90	-2	-2,00	-2,00	0,24	0,16	0,17	0,17
200	240	200	0	0,00	0,00	0,24	-0,36	-0,35	-0,35
230	150	10	-26	-26,00	-26,00	0,48	0,17	0,18	0,18
100	210	200	12	12,00	12,00	0,12	-0,12	-0,12	-0,12
60	210	200	17	17,00	17,00	0,24	0,11	0,11	0,11
100	140	200	12	12,00	12,00	0,31	-0,11	-0,11	-0,11
10	150	200	23	23,00	23,00	0,30	0,24	0,24	0,24
30	50	100	8	8,00	8,00	-0,23	-0,41	-0,41	-0,41
10	50	90	10	10,00	10,00	0,51	0,41	0,41	0,41
20	100	150	16	16,00	16,00	0,54	0,43	0,43	0,43
20	50	200	22	22,00	22,00	0,80	0,31	0,30	0,30
200	100	10	-23	-23,00	-23,00	-0,05	-1,07	-1,05	-1,05
200	50	20	-22	-22,00	-22,00	-0,08	-0,46	-0,46	-0,46
150	50	10	-17	-17,00	-17,00	-0,01	-0,54	-0,53	-0,53
100	50	30	-8	-8,00	-8,00	0,54	0,35	0,36	0,36
30	150	150	14	14,00	14,00	-0,38	-0,99	-0,99	-0,99
30	190	150	14	14,00	14,00	-0,49	-1,11	-1,10	-1,10
45	130	180	16	16,00	16,00	-0,02	-0,31	-0,31	-0,31

100	230	160	7	7,00	7,00	-0,24	0,10	0,11	0,11
40	90	230	23	23,00	23,00	0,60	0,03	0,02	0,02
20	180	140	14	14,00	14,00	-0,51	-0,27	-0,27	-0,27
120	220	250	16	16,00	16,00	0,66	0,51	0,52	0,52
120	240	120	0	0,00	0,00	-0,12	0,46	0,47	0,47
160	230	130	-4	-4,00	-4,00	-0,39	-0,70	-0,69	-0,69
150	150	30	-14	-14,00	-14,00	0,36	0,42	0,42	0,42
150	170	30	-14	-14,00	-14,00	0,31	0,11	0,12	0,12
180	130	40	-17	-17,00	-17,00	-0,09	-0,46	-0,46	-0,46
160	230	100	-7	-7,00	-7,00	0,14	0,02	0,03	0,03
230	90	40	-23	-23,00	-23,00	0,12	-0,49	-0,49	-0,49
140	180	20	-14	-14,00	-14,00	0,24	0,37	0,38	0,38
250	220	120	-16	-16,00	-16,00	-0,20	-0,94	-0,93	-0,93
180	230	180	0	0,00	0,00	0,18	-0,35	-0,35	-0,35
160	230	130	-4	-4,00	-4,00	-0,39	-0,65	-0,64	-0,64
100	120	10	-11	-11,00	-11,00	-0,30	0,18	0,18	0,18
200	120	10	-23	-23,00	-23,00	-0,10	0,14	0,14	0,14
150	100	20	-16	-16,00	-16,00	-0,32	-0,39	-0,39	-0,39
130	70	20	-13	-13,00	-13,00	0,32	0,31	0,31	0,31
130	140	50	-10	-10,00	-10,00	-0,40	-0,38	-0,38	-0,38
150	220	100	-6	-6,00	-6,00	-0,05	0,15	0,15	0,15
130	210	110	-2	-2,00	-2,00	0,36	0,48	0,48	0,48
100	230	80	-2	-2,00	-2,00	0,17	0,36	0,36	0,36
120	220	150	4	4,00	4,00	0,41	-0,16	-0,16	-0,16

Quadro 6 – Comparação entre taxas de aprendizado

x1	x2	x3	Yd	10 it.	1000 it.	1000000 it.	10 it.	1000 it.	1000000 it.
				Taxa = 0.00000000005			Taxa = 0.00005		
25	190	135	13	-12,04	-8,71	21,28	-1,E+201	NaN	NaN
45	130	180	16	-14,72	-10,79	26,56	-3,E+201	NaN	NaN
80	230	140	7	-19,91	-16,25	15,69	-1,E+202	NaN	NaN
30	110	220	23	-13,38	-8,81	35,72	-2,E+202	NaN	NaN
10	200	110	12	-9,35	-6,46	18,92	-1,E+202	NaN	NaN
110	220	200	11	-25,04	-20,27	23,00	-8,E+202	NaN	NaN
140	240	140	0	-27,37	-23,59	8,72	-6,E+203	NaN	NaN
160	230	130	-4	-29,69	-26,12	4,11	-4,E+204	NaN	NaN
160	170	130	-4	-28,37	-25,06	3,88	-3,E+205	NaN	NaN
170	110	70	-12	-25,86	-23,90	-7,71	-7,E+205	NaN	NaN
200	60	30	-20	-26,73	-25,68	-18,15	6,E+205	NaN	NaN
25	230	25	0	-9,00	-7,53	2,28	5,E+205	NaN	NaN
25	230	100	9	-11,53	-8,68	15,47	7,E+205	NaN	NaN
25	230	200	21	-14,92	-10,21	33,06	1,E+206	NaN	NaN
25	150	150	15	-11,47	-8,03	23,96	2,E+206	NaN	NaN
25	100	200	21	-12,06	-7,91	32,56	3,E+206	NaN	NaN
100	230	25	-9	-18,30	-16,74	-6,75	8,E+206	NaN	NaN
100	230	60	-5	-19,68	-17,48	-0,79	3,E+207	NaN	NaN
100	230	100	0	-20,83	-17,89	6,45	1,E+208	NaN	NaN
100	230	130	4	-21,45	-17,95	12,13	7,E+208	NaN	NaN
100	230	200	12	-24,22	-19,42	24,04	5,E+209	NaN	NaN
100	190	25	-9	-17,42	-16,03	-6,90	1,E+210	NaN	NaN
100	190	60	-5	-18,80	-16,77	-0,94	3,E+210	NaN	NaN
100	190	100	0	-19,95	-17,18	6,30	1,E+211	NaN	NaN
100	190	200	12	-23,34	-18,71	23,89	9,E+211	NaN	NaN
100	150	25	-9	-16,54	-15,32	-7,05	2,E+212	NaN	NaN
100	150	60	-5	-17,92	-16,06	-1,10	4,E+212	NaN	NaN
100	150	100	0	-19,08	-16,47	6,14	1,E+213	NaN	NaN
100	150	200	2	-32,46	-28,01	13,74	7,E+213	NaN	NaN
100	100	25	-9	-15,44	-14,44	-7,25	8,E+213	NaN	NaN
100	100	60	-5	-16,82	-15,17	-1,29	1,E+214	NaN	NaN
100	100	250	12	-21,36	-17,12	23,54	7,E+214	NaN	NaN
150	30	25	-15	-20,10	-19,34	-13,53	-5,E+214	NaN	NaN
150	30	60	-11	-21,49	-20,07	-7,57	-2,E+214	NaN	NaN
200	30	25	-21	-26,30	-25,48	-19,54	4,E+214	NaN	NaN
200	30	60	-17	-27,69	-26,21	-13,58	-2,E+214	NaN	NaN
200	120	100	-12	-30,82	-28,22	-6,00	-7,E+214	NaN	NaN
10	100	200	23	-10,00	-5,87	34,57	-4,E+214	NaN	NaN
10	160	200	23	-11,32	-6,93	34,80	-3,E+214	NaN	NaN
50	140	200	18	-16,04	-11,69	29,71	-8,E+214	NaN	NaN

70	50	200	16	-16,14	-12,15	27,36	-3,E+215	NaN	NaN
90	160	200	13	-21,64	-17,15	24,78	-1,E+216	NaN	NaN
90	240	220	16	-23,47	-18,28	29,20	-9,E+216	NaN	NaN
110	240	200	11	-25,47	-20,63	23,08	-7,E+217	NaN	NaN
140	220	160	2	-28,00	-23,94	11,76	-5,E+218	NaN	NaN
200	230	100	-12	-33,23	-30,17	-5,58	-4,E+219	NaN	NaN
200	230	160	-5	-35,46	-31,29	4,78	-4,E+220	NaN	NaN
200	150	150	-6	-33,17	-29,52	2,91	-3,E+221	NaN	NaN
220	140	100	-14	-33,33	-30,63	-7,93	-1,E+222	NaN	NaN
220	50	40	-22	-30,13	-28,92	-19,63	2,E+222	NaN	NaN
220	70	10	-25	-28,95	-28,21	-24,23	-3,E+222	NaN	NaN
220	70	100	-14	-31,80	-29,39	-8,20	-8,E+222	NaN	NaN
220	120	50	-20	-31,20	-29,51	-16,80	-1,E+223	NaN	NaN
130	120	10	-14	-18,69	-17,84	-13,02	-1,E+223	NaN	NaN
100	100	200	12	-21,36	-17,12	23,54	-5,E+223	NaN	NaN
120	230	120	0	-23,99	-20,65	7,56	-3,E+224	NaN	NaN
140	220	140	0	-26,92	-23,24	8,64	-2,E+225	NaN	NaN
140	230	80	-7	-24,91	-22,29	-1,68	-1,E+226	NaN	NaN
110	220	90	-2	-21,11	-18,39	3,85	-5,E+226	NaN	NaN
200	240	200	0	-36,83	-31,88	12,05	-6,E+227	NaN	NaN
230	150	10	-26	-31,75	-30,66	-24,93	-2,E+227	NaN	NaN
100	210	200	12	-23,77	-19,07	23,97	-1,E+228	NaN	NaN
60	210	200	17	-18,62	-13,95	28,98	-5,E+228	NaN	NaN
100	140	200	12	-22,24	-17,83	23,70	-3,E+229	NaN	NaN
10	150	200	23	-11,10	-6,75	34,76	-2,E+229	NaN	NaN
30	50	100	8	-8,60	-6,50	13,78	-1,E+229	NaN	NaN
10	50	90	10	-4,98	-3,09	15,22	-4,E+228	NaN	NaN
20	100	150	16	-9,35	-6,13	24,77	-3,E+228	NaN	NaN
20	50	200	22	-9,94	-6,01	33,37	-3,E+228	NaN	NaN
200	100	10	-23	-27,53	-26,69	-22,11	2,E+228	NaN	NaN
200	50	20	-22	-26,97	-26,15	-20,74	-3,E+228	NaN	NaN
150	50	10	-17	-20,23	-19,66	-16,29	3,E+228	NaN	NaN
100	50	30	-8	-14,11	-13,23	-6,16	1,E+228	NaN	NaN
30	150	150	14	-12,49	-9,04	22,96	2,E+228	NaN	NaN
30	190	150	14	-13,36	-9,75	23,11	3,E+228	NaN	NaN
45	130	180	16	-14,72	-10,79	26,56	7,E+228	NaN	NaN
100	230	160	7	-23,06	-19,01	16,81	4,E+229	NaN	NaN
40	90	230	23	-14,51	-9,83	36,20	1,E+230	NaN	NaN
20	180	140	14	-11,57	-8,19	22,52	1,E+230	NaN	NaN
120	220	250	16	-27,76	-22,07	30,80	1,E+231	NaN	NaN
120	240	120	0	-24,21	-20,83	7,60	6,E+231	NaN	NaN
160	230	130	-4	-29,69	-26,12	4,11	5,E+232	NaN	NaN
150	150	30	-14	-22,51	-21,14	-11,79	1,E+233	NaN	NaN
150	170	30	-14	-22,95	-21,49	-11,71	2,E+233	NaN	NaN
180	130	40	-17	-26,73	-25,22	-14,31	4,E+233	NaN	NaN

160	230	100	-7	-28,07	-25,06	-0,57	3,E+234	NaN	NaN
230	90	40	-23	-32,05	-30,65	-20,48	-5,E+233	NaN	NaN
140	180	20	-14	-21,59	-20,29	-12,23	-1,E+234	NaN	NaN
250	220	120	-16	-40,29	-36,84	-8,51	-1,E+235	NaN	NaN
180	230	180	0	-33,46	-28,94	10,90	-1,E+236	NaN	NaN
160	230	130	-4	-29,68	-26,12	4,11	-7,E+236	NaN	NaN
100	120	10	-11	-15,57	-14,76	-10,01	-7,E+236	NaN	NaN
200	120	10	-23	-27,97	-27,04	-22,03	1,E+235	NaN	NaN
150	100	20	-16	-21,87	-20,90	-14,54	8,E+234	NaN	NaN
130	70	20	-13	-18,13	-17,31	-11,65	2,E+234	NaN	NaN
130	140	50	-10	-21,28	-19,61	-6,70	5,E+234	NaN	NaN
150	220	100	-6	-26,81	-23,85	0,40	3,E+235	NaN	NaN
130	210	110	-2	-24,05	-20,97	4,92	2,E+236	NaN	NaN
100	230	80	-2	-19,75	-17,18	3,33	6,E+236	NaN	NaN
120	220	150	4	-24,38	-20,53	13,20	4,E+237	NaN	NaN