PROPOSTA DE ARQUITETURA PARA EXECUÇÃO DE REDES NEURAIS ARTIFICIAIS EM DISPOSITIVOS FPGA

IGOR O. FONSECA[1]; WELBERT H. L. CASTRO[2]; MILTON R. HEINEN[3], BRUNO S. NEVES[4]

[1] *Universidade Federal do Pampa – igorfonseca@unipampa.edu.br*
[2] *Universidade Federal do Pampa – welberthime@gmail.com*
[3] *Universidade Federal do Pampa – milton.heinen@unipampa.edu.br*
[4] *Universidade Federal do Pampa – brunoneves@unipampa.edu.br*

1. INTRODUÇÃO

A utilização de redes neurais tem atraído grande atenção nos últimos anos. Desde que foi proposto o primeiro modelo de neurônio artificial, em 1943, novas e mais sofisticadas propostas são feitas ano a ano. Hoje, a Inteligência Artificial (IA) tem aplicações como automação de rotinas, interpretação de textos e imagens, diagnósticos médicos e suporte para pesquisas científicas, resolvendo rapidamente problemas difíceis para humanos, especialmente quando o prazo para geração da solução é exíguo, porém fáceis para computadores (GOODFELLOW; BENGIO; COURVILLE, 2016).

Uma Rede Neural Artificial (RNA) é um sistema projetado para emular o comportamento que o cérebro humano exerce para executar determinadas tarefas. Com ela, é possível realizar o processamento distribuído e paralelo através de unidades de processamento simples, denominadas neurônios artificiais, com propósito de armazenar conhecimento experiencial e disponibilizá-lo para utilização (HAYKIN, 2003). O propósito de uma RNA é ser um sistema capaz de aprender através de um conjunto de exemplos (NIELSEN, 2015). As RNAs são adequadas para problemas nos quais os dados de treinamento correspondem a dados de sensores complexos e ruidosos, como entradas de câmeras e microfones, além de serem aplicáveis a problemas para os quais são usadas representações simbólicas, como tarefas de aprendizagem com árvores de decisão (MITCHELL, 1997).

Existem algumas alternativas para implementação de RNAs, sendo as duas principais: (*i*) execução em software, com computadores convencionais, e (*ii*) solução em hardware específico, capaz de decrementar o tempo de execução (ROJAS, 1996). Na prática, implementações em software têm sua velocidade limitada em processadores sequenciais. Este problema é resolvido pelo alto paralelismo que pode ser alcançado em circuitos VLSI (BOSER et al., 1991), enquanto a implementação em dispositivos FPGA dispõe do paralelismo e das adaptações possíveis em software (OMONDI; RAJAPAKSE; BAJGER, 2006). Outras implementações possíveis para RNAs são feitas com uso de software sobre processadores de propósito geral (OH; JUNG, 2004), com uso de software sobre processadores com conjunto de instruções específicas (ASIP) (SHAPIRO et al., 2011) e com implementação híbrida entre software e hardware (utilização de aceleradores/coprocessadores) (ZHAO et al., 2017).

Este trabalho propõe uma arquitetura que possibilite a execução em dispositivos FPGA de RNAs já treinadas, com pesos e funções de ativação previamente definidas. Para isso, é proposto um método de codificação das camadas e os respectivos pesos de seus neurônios para alocação em memória com a implementação de funções de ativação, seu cálculo e alimentação nas camadas seguintes. A arquitetura paraleliza as operações dos neurônios e acessos a memória em uma mesma camada, formando um pipeline a partir da

reutilização da mesma estrutura em todas as camadas. Assim, o atraso de propagação das entradas até as saídas se torna linearmente dependente do número de camadas até o preenchimento do pipeline, após, o hardware passará a produzir novas saídas a cada ciclo.

2. METODOLOGIA

Este trabalho foi desenvolvido em seis etapas, descritas a seguir, que envolvem a revisão da literatura e proposta de uma arquitetura para execução de RNAs em dispositivos FPGA. (i) Estudo do funcionamento de uma RNA, com foco na implementação dos neurônios baseada no modelo de perceptron; (ii) Apresentação de uma visão geral da arquitetura proposta para processamento de RNAs, dando ênfase a descrição da codificação da rede e de seus parâmetros de configuração; (iii) Descrição com maior aprofundamento a organização das memórias necessárias para suporte ao processamento dos neurônios artificiais. (iv) apresentação da arquitetura do neurônio utilizado neste trabalho; (v) Apresentação, em linhas gerais, da unidade de controle da arquitetura desenvolvida; (vi) Apresentação de uma análise sobre a precisão dos dados para processamento na arquitetura;

3. RESULTADOS E DISCUSSÃO

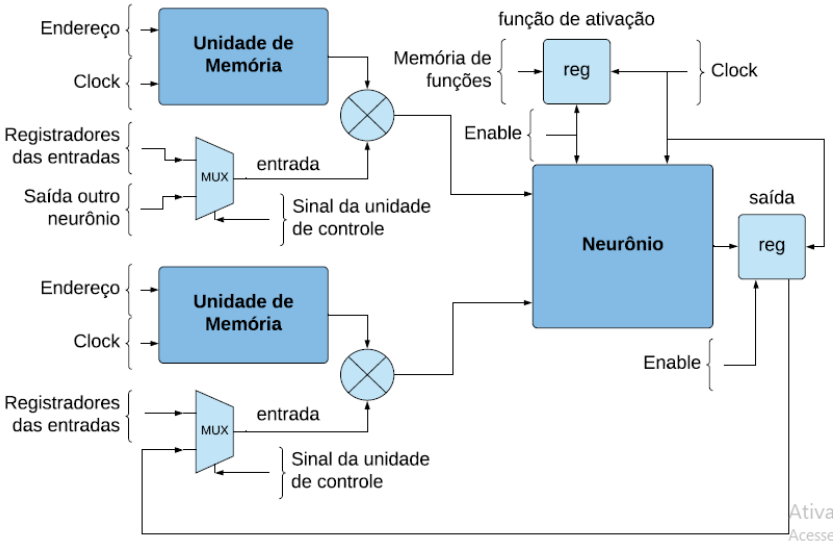
Algumas informações importantes são necessárias para a parametrização de uma rede neural. Sua execução necessita do conhecimento de: (*i*) as entradas, (*ii*) os pesos utilizados em cada neurônio, (*iii*) as funções de ativação, (iv) o número de neurônios em cada camada, (*v*) o número de camadas e (*vi*) o número de saídas. Nas redes perceptron todos os neurônios de uma camada alimentam as entradas da camada seguinte com suas saídas, assim, todos os neurônios recebem os dados gerados pela camada anterior. Esta característica permite que a implementação em hardware reutilize a mesma estrutura para gerar os dados de cada camada, intercalando entre as transições das camadas, acessos as memórias buscando os parâmetros para o próximo ciclo de execução.

Embora ao adicionar mais neurônios e camadas as RNAs possam executar funções mais poderosas (GOODFELLOW; BENGIO; COURVILLE, 2016), o número de camadas e neurônios se torna um fator crucial para a área ocupada e a latência total na produção de respostas da arquitetura proposta. Além disso, o número máximo de camadas, neurônios e a precisão dos dados calculados podem ser ajustados incrementando ou decrementando a largura de palavra utilizada na arquitetura (ROJAS, 1996), de modo que a solução se torne escalável e capaz de suportar redes densas, respeitando as limitações do hardware utilizado.

Implementações de redes neurais em dispositivos FPGA para processamento de imagens em tempo real utilizam memórias individuais para cada neurônio a fim de possibilitar a paralelização das operações entre eles (FAN YANG; PAINDAVOINE, 2003), uma vez que o uso de uma única uma unidade de memória compartilhada entre todos os neurônios de uma camada inviabilizaria a leitura em paralelo de todos os pesos de neurônio. A estratégia de utilização de memórias individuais dedicadas à configuração de cada neurônio se torna custosa em área, porém ela permite a configuração em paralelo dos pesos para os neurônios, viabilizando um processamento mais natural e efetivo do pipeline entre as camadas da RNA.

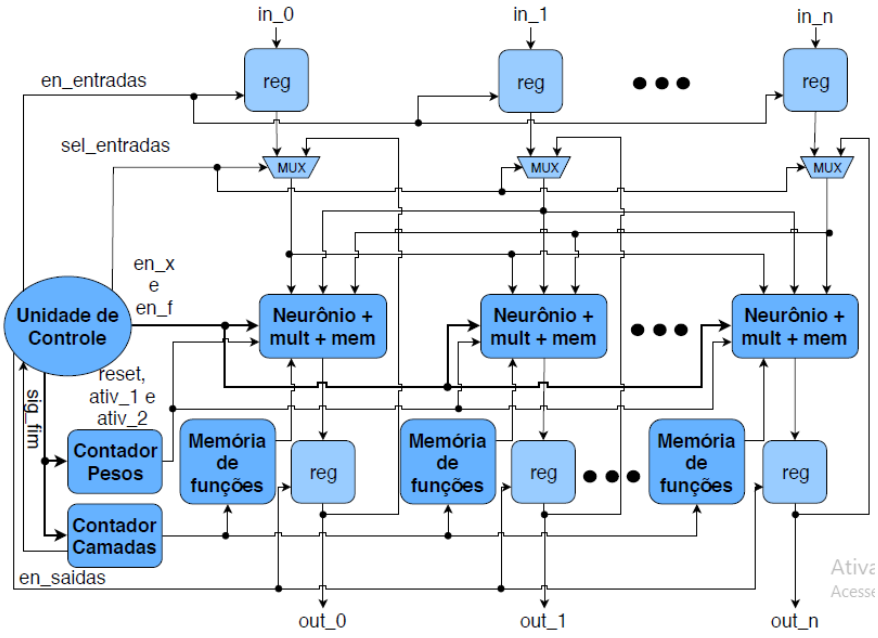
A Figura 1 ilustra a interação entre as memórias dedicadas aos pesos e um neurônio em uma RNA hipotética. As possíveis entradas dependem do tipo de camada. Camadas de entrada recebem os registros lidos na entrada da arquitetura, enquanto uma camada oculta ou de saída recebe o valor de todos os neurônios (inclusive o valor armazenado no registrador de saída do próprio neurônio). O valor recebido na entrada do neurônio é especificado por um multiplexador com sinal seletor recebido da unidade de controle. Cada neurônio, então, recebe uma memória dedicada a memorizar sua função de ativação naquela camada.

Figura 1. Unidades de memória dedicadas a um neurônio



A Figura 2 traz um *datapath* da arquitetura com Unidade de Controle, porém sem incluir as Unidades de Memória e multiplicadores já trazidos na Figura 1, porém apresentando os contadores importantes para emissão do sinal de finalização da execução.

Figura 2. Datapath da arquitetura.



4. CONCLUSÕES

Foram apresentadas decisões para implementação de uma arquitetura capaz de executar RNAs em dispositivos FPGA. A utilização de dispositivos FPGA permite a velocidade inerente a uma implementação feita em hardware e, ao mesmo tempo, a versatilidade para alteração da arquitetura (baseado no objetivo final de uso) típica de implementações em software.

Para trabalhos futuros é interessante a implementação de um módulo capaz de realizar a inicialização de pesos e a adaptação deles através de algoritmos de treinamento como *Backpropagation*. Desta forma, não só a execução da rede é acelerada, mas também seu aprendizado, exigindo a leitura de amostras e cálculos para o incremento das taxas de acerto das RNAs. Outra linha de trabalho futuro a ser desenvolvida é a realização das adequações necessárias para criar suporte para uma RNC (Rede Neural Convolucional), buscando realizar customizações para uso em aplicações de visão computacional.

5. REFERÊNCIAS BIBLIOGRÁFICAS

BOSER, B. et al. **An Analog Neural Network Processor with Programmable Topology**. [S.l.]: IEEE Journal of Solid-State Circuits (Volume: 26 , Issue: 12, Dec1991), 1991. 2017-2025 p.

FAN YANG; PAINDAVOINE, M. Implementation of an RBF neural network on embedded systems: real-time face tracking and identity verification. **IEEE Transactions on Neural Networks**, 14(5):1162–1175, 2003.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep learning**. [S.l.]: MIT press, Series: Adaptive computation and machine learning series, 2016.

HAYKIN, S. S. **Neural networks and learning machines**. Third. Upper Saddle River,NJ: Pearson Education, 2009.

MITCHELL, T. M. **Machine Learning**. [S.l.]: McGraw-Hill Science/Engineering/Math, 1997.

NIELSEN, M. A. **Neural networks and deep learning**. [S.l.]: Determination press SanFrancisco, CA, USA:, 2015. v. 25.

OMONDI, A. R.; RAJAPAKSE, J. C.; BAJGER, M. FPGA neurocomputers. In: **FPGA Implementations of Neural Networks**. Boston, MA: Springer US, 2006. p. 1–36. ISBN 978-0-387-28487-3. Disponível em: <https://doi.org/10.1007/0-387-28487-7_1>.

ROJAS, R. **Neural networks: a systematic introduction**. [S.l.]: Springer Science &Business Media, 1996.

SHAPIRO, D. et al. Asips for artificial neural networks. **2011 6th IEEE InternationalSymposium on Applied Computational Intelligence and Informatics (SACI)**, p.529–533, 2011.

ZHAO, R. et al. Hardware acceleration for machine learning. **2017 IEEE Computer Society Annual Symposium on VLSI**, 2017.