ASEE AMERICAN SOCIETY FOR ENGINEERING EDUCATION

Paper ID #35176

FPGA/MATLAB Hardware in the Loop Testbed for Stochastic Artificial Neural Networks

Mr. Matthew Carrano, Baylor University
Dr. Scott Koziol, Baylor University

SCOTT KOZIOL received the B.S.E.E. degree in electrical engineering from Cedarville University, Cedarville, OH, USA, in 1998, the M.S. degree in electrical engineering from Iowa State University, Ames, IA, USA, in 2000, and the M.S.M.E. degree in mechanical engineering and the Ph.D. degree in robotics from the Georgia Institute of Technology, Atlanta, GA, USA, in 2011 and 2013, respectively. He is currently an Associate Professor and Assistant Chair with the Department of Electrical and Computer Engineering, Baylor University, Waco, TX, USA

Dr. Eugene Chabot, University of Rhode Island

Dr. Chabot is a researcher for the Department of the Navy and an adjunct faculty member at the University of Rhode Island in the department of Electrical, Computer, and Biomedical Engineering. His research focus is on navigation, autonomous systems, and applications of neuroscience with an emphasis on cognitive processing, sensory, and perception.

Jacob Boline
John DiCecco

Session XXXX

FPGA/MATLAB Hardware in the Loop Testbed for Stochastic Artificial Neural Networks

Jacob Boline, Matthew Carrano, Scott Koziol
Department of Electrical and Computer Engineering
Baylor University

John DiCecco, Eugene Chabot
Naval Undersea Warfare Center, Newport, RI
&
Electrical, Computer and Biomedical Engineering Department
University of Rhode Island

Abstract

This paper presents a Hardware in the Loop (HWIL) testbed[1] for evaluating a Stochastic Artificial Neural Network (SANN). The SANN is implemented in a Field Programmable Gate Array (FPGA), and this testbed allows test data to be generated on a computer using MATLAB, and sent to the FPGA SANN over a Universal Asynchronous Receiver/Transmitter (UART) interface. Initial hardware results are presented.

Introduction

As interest in the research of Stochastic Artificial Neural Networks (SANN) has increased[2,3,4], so has the desire to run them on reconfigurable Field Programmable Gate Arrays (FPGAs)[5]. This comes with its own set of challenges such as interfacing between sensors and a coprocessor, or computer. This paper presents a high level overview of our solution which uses a Universal Asynchronous Receiver/Transmitter (UART) interface to communicate between a computer running MATLAB and an FPGA running a SANN. Instead of having an external sensor, a 4-by-4 pixel image (where each pixel has a dynamic range of 16-bits) was sent from MATLAB to the FPGA. After, MATLAB received a stochastic stream back for processing. This design was developed using Vivado 2019.1 and tested on a Xilinx Nexys 4 DDR board containing an Artix-7 device.

Architecture

In this section we will discuss the interface of the SANN and the architecture of the UART to SANN interface.

SANN Interface

Figure 1 shows a basic overview of the interfaces for the SANN black box. The basic theory of operation is that the network is fed 236 16-bit seeds that are fixed. These seeds are derived from training the neural network on a computer, then hard coded in the implementation on the FPGA. The network is then presented with the 4-by-4 pixel image for classification. The output of the SANN is a stochastic bitstream that represents values between -1 and 1. This bitstream represents the classification of the image input into the network based on the training seeds.

Figure 1. Overview of the SANN Interface. The SANN takes in the proper clocks, fixed seeds that are produced through training the network, and images (which the SANN will classify). The output of the SANN is a constantly running stochastic stream with values between -1 and 1.

FPGA Architecture

Figure 2 shows a high level overview of the system in place on the FPGA. This system is operated using a Mixed-Mode Clock Manager (MMCM) Module that is fed using the Nexys 4 DDR's onboard 100 MHz oscillator. This MMCM is responsible for producing the required clock signals and logic to notify the rest of the modules when the clock is stable and operation can begin. The core of the FPGA architecture is the Xilinx MicroBlaze[6], a customizable 32-bit RISC soft processor that integrates tightly with the Xilinx ecosystem, and can be easily interfaced with many pre-existing Xilinx IPs[7]. The MicroBlaze System block in Figure 2 is a system developed to contain all the required logic to support a MicroBlaze, UART interface, and AXI to SANN interface. A bidirectional interface is developed to allow the MicroBlaze System to send images to the SANN, and receive the stochastic stream from the SANN where it is processed and sent to MATLAB over UART for further processing. This system was designed to be as modular as possible allowing different SANNs of the same resolution and size to be connected without having to modify any of the MicroBlaze System code. In addition, the MicroBlaze System was designed to minimize changes required for moving to SANNs of different resolutions and sizes. This allows us to easily build and test different SANN architectures to get data for comparison.

SANN/FPGA Results

A MATLAB script was developed to send and receive data from the FPGA. This script begins by loading image data stored on the computer, and saved as decimal values between 0 and 1. They are then quantized in order to fit into the 16-bit register required for the neural network. This data is then

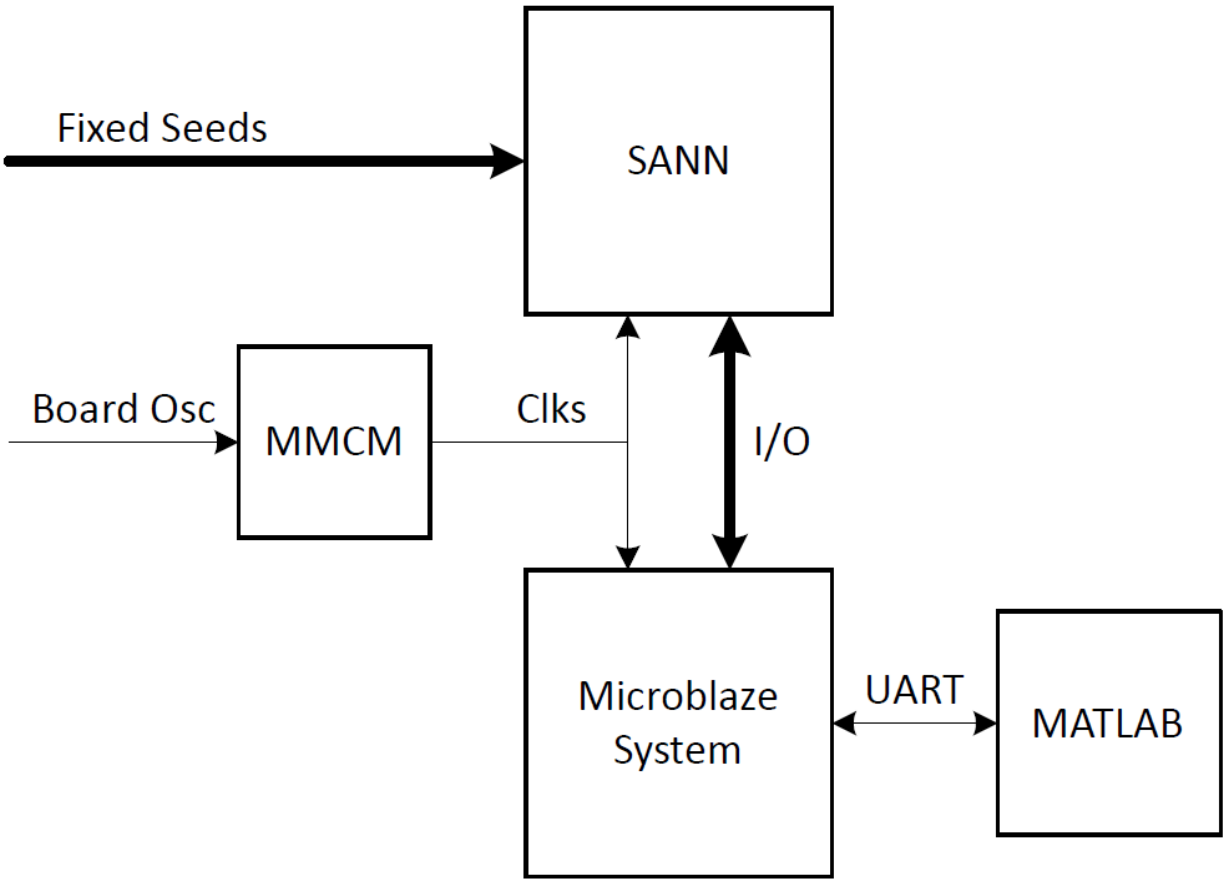

Figure 2. High-level overview of the entire system. 100 MHz comes in from the onboard crystal oscillator, while the UART interface is over a USB connection to a computer running MATLAB. The MMCM, SANN, and MicroBlaze System are contained inside of the FPGA in addition to the fixed seeds, clocks, and I/O.

sent over UART with the respective register address as the first nibble in the word. After, a string is sent to the FPGA to tell it to enter *data send* mode at which point the internal buffer is turned on and MATLAB begins reading the specified number of 28-bit packages in order to get the specified bit stream length. After the data has been read MATLAB unpacks the data into the proper bit streams, finds the stochastic average, and then scales to the proper scaling. The result is then compared to the ideal output for that image in order to verify that the FPGA SANN calculated the correct value. Table I shows an example of the SANN properly classifying and image.

Table 1. Comparison between FPGA SANN results for a bit stream length of 2048 bits and the ideal value. In this case it is shown that Class 2 is the identified class, which correctly matches with the idealized outcome

Class	FPGA (Experimental)	Ideal
1	-0.0585	-1
2	0.4776	1
3	0.1014	-1
4	-0.0214	-1
5	-0.4181	-1
6	-0.0897	-1
7	-0.2846	-1

Analysis

Our tests were done with images composed of 16 pixels, however scaling up to images that are larger should be trivial. The custom blocks were designed to be as flexible as possible, and the SANN input peripheral can be resized by changing a single setting and adjusting the connections in the top level file. One thing we did not account for was different dynamic ranges as all of our tests were on 16-bit unsigned integers. However, increasing the dynamic range should not be too difficult as long as the registers do not exceed 32-bits. Even with all of the interface logic and the SANN there is ample capacity on the FPGA for expansion of both the SANN and the MicroBlaze System.

This system was designed to be fairly basic and used for proof of concept with smaller SANNs. For systems that require much more space, or speed, this design can be customized to run on a more advanced system. For example, one could use a Xilinx Zynq line and run all of the C code on a hard ARM processor rather than a soft core in the programmable logic region of the FPGA. Different data interfaces could be used such as Ethernet, PCIe, or even fiber connections.

Discussion

In conclusion, we have shown a Hardware in the Loop (HWIL) testbed that implements bidirectional communication between a computer and a SANN on a FPGA. Because we assumed some data loss was acceptable we made a design choice in favor of a slower and simpler data interface (UART). A fielded SANN system, as opposed to our testbed, would receive data from sensors rather than from the computer. This design change is of reasonable complexity as one would just have to adjust the input to the SANN and put a sample & hold system between the sensors and the SANN. This sample & hold system could be interfaced to the MicroBlaze and work in tandem with the data output system to control the flow of data and accuracy of the final stochastic calculations.

Acknowledgement

This study is supported in whole or in part by funds from the Naval Engineering Education Consortium (NEEC).

References

1. J. Boline (2020), *Stochastic Computing & Stochastic Resonance in Digital & Analog Neuromorphic Systems*, [MSECE Thesis, Baylor University].
2. A. Coninx, P. Bessière, E. Mazer, J. Droulez, R. Laurent, M. A. Aslam, and J. Lobo, "Bayesian sensor fusion with fast and low power stochastic circuits," in 2016 IEEE International Conference on Rebooting Computing (ICRC), Oct 2016, pp. 1–8.
3. M. Faix, E. Mazer, R. Laurent, M. Othman Abdallah, R. Le Hy, and J. Lobo, "Cognitive computation: A bayesian machine case study," in 2015 IEEE 14th International Conference on Cognitive Informatics Cognitive Computing (ICCI*CC), July 2015, pp. 67–75.
4. R. Frisch, R. Laurent, M. Faix, L. Girin, L. Fesquet, A. Lux, J. Droulez, P. Bessière, and E. Mazer, "A bayesian stochastic machine for sound source localization," in 2017 IEEE International Conference on Rebooting Computing (ICRC), Nov 2017, pp. 1–8.
5. R. Frisch, "Stochastic machines dedicated to Bayesian inference for source localization and separation," Theses, Université Grenoble Alpes, Nov. 2019. [Online]. Available: https://tel.archives-ouvertes.fr/tel-02513349
6. B. Muralikrishna and K. Gnana Deepika, "Input/output peripheral devices control through serial communication using MicroBlaze processor," in 2012 International Conference on Devices, Circuits and Systems (ICDCS), 2012, pp. 532–536.
7. V. Kale, Using the MicroBlaze Processor to Accelerate Cost-Sensitive Embedded System Development, Xilinx, 2015.

JACOB BOLINE
Mr. Boline completed his MSECE degree (December 2020) in the Electrical and Computer Engineering Department at Baylor University.

MATTHEW CARRANO
Mr. Carrano is currently pursuing an MSECE degree in the Electrical and Computer Engineering Department at Baylor University.

SCOTT KOZIOL
Dr. Koziol currently serves as an Associate Professor and Assistant Chair of the Electrical and Computer Engineering Department at Baylor University.

JOHN DICECCO
Dr. DiCecco currently serves at the Naval Undersea Warfare Center in Newport Rhode Island and as Adjunct Professor in the Electrical, Computer and Biomedical Engineering Department at the University of Rhode Island.

EUGENE CHABOT
Dr. Chabot currently serves at the Naval Undersea Warfare Center in Newport Rhode Island and as Adjunct Professor in the Electrical, Computer and Biomedical Engineering Department at the University of Rhode Island.