Hardware Accelerator Test Bench for Error-Correcting Algorithms

Mike Thomson, Dr. Elizabeth Brauer

NORTHERN ARIZONA UNIVERSITY
College of Engineering & Natural Sciences
Department of Electrical Engineering
Flagstaff, AZ 86011

Background / Project Rationale

This project focuses on producing a hardware-based test bench for rapid testing and development of error-correcting algorithms. The test bench needs to create repeatable, normally-distributed "white noise" to emulate real-world interference. The test bench introduces the "noise" into data that is then sent to the error-correcting module. The performance of the test bench and the success rate of the error-correcting algorithms are recorded. The Gaussian noise generator module of the project is implemented in hardware on a Cyclone FPGA from Altera and is able to generate 50 million samples per second.

- Mentor: Dr. Elizabeth Brauer: circuits
  - VHDL
  - FPGA
- Dr. Sheryl Howard: error-correcting codes
  - Correct errors from data transmission
  - Currently implemented and tested in software
  - Trillions of test cases
  - Software not fast enough
- This project: move to hardware-based
  - Faster than software – at least 10x
  - Needs hardware-generated "white noise"
  - Needs error-correcting codes (ECC) in hardware

Introduction

When data is transmitted, noise and interference are always present. Examples of noise-inducing channels are:

- The air that cell phone signals travel through
- Wires that connect computers and their peripherals
- Plastic on CDs (and scratches in the plastic) which CD player lasers must pass through
- Water that sonar must propagate through

Interference in these "noisy" channels can cause the transmitted data to become corrupted (Figure 1).

Figure 1 – A simple model of the effects noise can introduce into a transmitted signal.

Preliminary Research / Work

Prior work:

- Gaussian noise generator [1,3] (Figure 2.)
- Howard, error-correcting codes

My work:

- ECC studied in prior courses
- Modeled noise generator in Matlab & VHDL

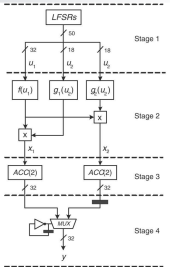

Figure 2 - **Gaussian noise generator architecture.** (Dong-U Lee, et. al. "A Gaussian Noise Generator for Hardware-Based Simulations," IEEE Trans. on Comp., vol. 53, no. 12, 2004.)

Hardware-based Implementation

- Initially modeled in Matlab
- Converted code to VHDL
  - VHDL – **V**HSIC **H**ardware **D**escription **L**anguage
  - FPGA – **F**ield-**p**rogrammable **G**ate **A**rray
- Simulate VHDL code
  - Compare with Matlab results
- Program chip from VHDL code
- Using Altera development board (Figure 3.)

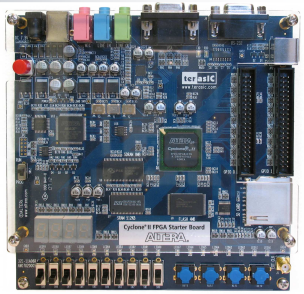

Figure 3 - The Altera FPGA board I am developing with.

Results: Noise Generator

Noise generated as expected – pseudo random (see Figure 4.)

- Normally distributed
- Near-zero correlation
- Repeatable testing
- Matlab = 500 noise samples per second
- FPGA = 50 million noise samples per second
- FPGA usage: area = 7%, memory = 62%

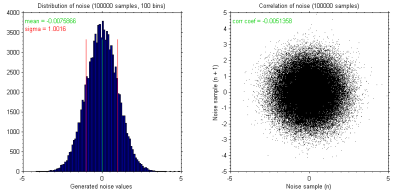

Figure 4 – VHDL output distribution and correlation.

Results: Error-Correcting Codes

Hamming (7,4) decoder tests:

- Decoder already implemented from prior coursework
- Uses small codewords (8 bits)
- Tested as proof-of-concept

Wrote a statistics-recording module:

- Number of codewords received
- Number of codewords with 1, 2, and 3 errors
- Number of errors corrected
- Number of false-positive corrections

Future Work / Conclusions

Complete VHDL Min-Sum decoder [4]

- Matlab version already completed
- Initial testing with small codewords (16 bits)
- Further testing with industry-length codewords (2048 bits)
- Speed vs. area tradeoff analyses
- These are the types of algorithms Dr. Howard will be testing

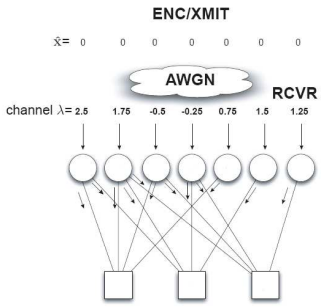

Figure 5 – Example Min-Sum factor graph[4] (algorithm visual representation)

References

[1] A. Alimohammad, *et al.* "Area-Efficient Parallel White Gaussian Noise Generator," *Elec and Comp Eng, Canadian Conf.* May 2005.
[2] A. Alimohammad, *et al.* "An Iterative Hardware Gaussian Noise Generator," *PACRIM conf on comm, comp, and SP.* 2005.
[3] D. Lee, *et al.* "A Gaussian Noise Generator for Harware-Based Simulations," *IEEE Trans on Comp.* Dec 2004.
[4] S. Howard. "Low-Density Parity-Check (LDPC) Decoder," Course slide presentation. Fall 2007.

Collaborators

- Dr. Sheryl Howard, Electrical Engineering, Error Correcting Codes.
- Jeremy Danny and Harlan Greeley, EE, Undergraduate