Final Project Proposal: Configurable AES Accelerator

ECE 571, Spring 2016

Alex Pearson, Daniel Collins, Scott Lawson

Project Description

Our project will consist of creating and verifying an AES encryption and decryption accelerator that is configurable at compilation time to AES-128, AES-192, and AES-256. The three versions of AES differ primarily in the key size and the number of rounds applied, so one of the challenges will be architecting our modules to maximize design reuse and configurability. Our accelerator will have a 128-bit input bus that both the keys and 128-bit blocks are streamed into. The system will include a synchronous reset that flushes the accelerator data in progress, separate command and data input FIFOs, and an output data FIFO. Accompanying the command FIFO will be a set of commands for the processor, including set key, data write, and data read. The slice configuration will be set at compile time and will select the desired AES variant.

Implementation and Verification Strategies

To begin, we will work together to define the top-level block diagram and the sub-modules that need to be implemented. Each round during the AES encryption process is broken up into several stages. These stages will be distributed to the team members and each team member will create a module and testbench for at least 1 stage. We will then divide up the top level modules, SystemVerilog testbench, and Python encryption/decryption scripts for initial implementation. At this point, we will need to work together to integrate, debug, and validate the correctness of our encryption and decryption modules.

We will verify our design through a combination of directed and random testing. For directed testing, some modules within the hierarchical design will have their own testbenches. Our top-level modules will undergo directed testing for correct behavior when we toggle the reset and new key signals at different times. We will also test using the published AES Known Answer Test Vectors. For random testing, we will use a Python script to generate plaintext, keys, and a corresponding golden model ciphertext (using a TBD third-party library), and we will use this data inside of the SystemVerilog testbench to compare to our accelerator's output. Since we will have both an encryption and decryption accelerator we can also test the 2 blocks by using the output of one as input to the other and comparing to the original input .

We believe that it will be most appropriate to use the TBX mode of the Veloce emulator to test our design.

Planned Design Sources

We will create our design based on algorithms and AES specifications from the following sources:

Pub, NIST FIPS. "197: Advanced encryption standard (AES)." *Federal Information Processing Standards Publication* 197 (2001): 441-0311.
(http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf)

S.-M. Yoo, D. Kotturi, D.W. Pan, J. Blizzard, An AES crypto chip using a high-speed parallel pipelined architecture, Microprocessors and Microsystems, Volume 29, Issue 7, 1 September 2005, Pages 317-326, ISSN 0141-9331, http://dx.doi.org/10.1016/j.micpro.2004.12.001.
(http://www.sciencedirect.com/science/article/pii/S0141933104001632)

https://en.wikipedia.org/wiki/Advanced_Encryption_Standard