ComS / CprE 583 – Reconfigurable Computing
Homework #3
Assigned: September 25
Due: October 9 (12:00pm)

[Note from Joe: The first three questions here relate directly to systolic computing, since that has been the majority focus of what will be covered in class over the next two weeks. The final question has to do with FPGA synthesis and again requires the use of Xilinx ISE (Webpack or Full). Since there are other deadlines for this course between now and the October 9 due date I have made this considerably shorter than both HW #1 and HW #2.]

1) Take a look at any flavor of the Secure Hash Standard (SHA-1, SHA-256, SHA-384, or SHA-512) as specified in FIPS pub 180-2 that can be found at the following web address: http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf.

(a) Provide the pseudocode for the hashing algorithm that you have selected. What does it take as input, what is the expected output, and how is the input transformed into the output? What is the run-time of the basic algorithmic implementation as a function of the size of the input? *[Any CLR-style pseudocode is acceptable here. If you're not comfortable with this feel free to provide a simplified C/C++/Java implementation instead.]*

(b) Would SHA be well-suited to a systolic implementation? If your answer is yes, briefly sketch out a sample systolic structure and flow. If you answer is no, please be specific as to why a systolic implementation would not be a good choice. In either case, what expected speedup could one hope to achieve? *[This is not a trick question – the answer you get for the expected speedup should provide justification for your 'yes' or 'no' answer. As always, just provide the assumptions and reasoning for arriving at your solution.]*

2) In class we discussed various different systolic computing structures, including the one-dimensional linear array, the two-dimensional mesh, and the hexagonal array. We also discussed how various applications could be implemented on each structure based on their dataflow and computation requirements. Provide a rough schematic of what a three-dimensional mesh might look like. How does a three-dimensional mesh differ from a hexagonal array? Provide an example of a computational flow that would make sense on a three-dimensional mesh but not on a hexagonal array, and vice versa. *[I am not looking for an algorithm here, just a description of a base computing element and dataflow.]*

3) Performing a LU factorization is a popular method of solving linear equations. After matrix A is factored into $L \cdot U$, where L is unit lower triangular and U is upper triangular, then solutions for arbitrary x vectors given A, b, and $A \cdot x = b$ can be inexpensively computed. Part of the solution process is a procedure called *back-substitution.* Back-substitution is a method of solving a lower-triangular set of equations such as those shown below:

$$q \left\{ \begin{bmatrix} a_{11} & & & & & 0 \\ a_{21} & a_{22} & & & & \\ a_{31} & a_{32} & a_{33} & & & \\ a_{41} & a_{42} & a_{43} & a_{44} & & \\ & a_{52} & a_{53} & a_{54} & a_{55} & \\ 0 & & a_{63} & a_{64} & a_{65} & \ddots \end{bmatrix} \right. \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ \vdots \\ y_n \end{bmatrix}$$

Here we have assumed that the matrix is banded with a lower bandwidth equal to q. The solution for a dense matrix is the same as that for a banded matrix with a bandwidth of n.

(a) We can see that for this equation given above, solving for x_1 is relatively easy ($x_1 = y_1/a_{11}$). After that point it gets a bit trickier, as x_2 is now a function of x_1, a_{21}, a_{22}, and y_2, x_3 is a function of x_1, x_2, a_{31}, a_{32}, a_{33}, y_3, etc. Provide a general solution for x as a function of A and y. *[Assume that q=4 as given in the above depiction. The correct solution requires a series of recursive relations.]*

(b) Design a one-dimensional linear array that can be used to implement the solution for part a) above. Your solution should describe the both the computational blocks and the how the data input/output patterns as a function of time. *[One correct solution will look somewhat similar to the banded matrix-vector product systolic implementation discussed in class.]*

4) See attached file *hw3.zip* which contains four different hardware benchmarks from the ITC'99 repository (http://www.cad.polito.it/tools/itc99.html). Specifically, you should find the following files:

- *b09.vhd* – a VHDL RT-level implementation of a serial-to-serial converter
- *b11.vhd* – a VHDL RT-level implementation of a string scrambling cipher
- *b14_1.vhd* – a VHDL RT-level implementation of a subset of a VIPER processor (RISC)
- *b15_1.vhd* – a VHDL RT-level implementation of a subset of a 80386 processor (CISC)

(a) Provide a brief description of each of the benchmarks by analyzing the VHDL source files.

(b) Synthesize the four designs onto a Xilinx XC2V250 FPGA using ISE with default settings. Provide a chart detailing the area and performance characteristics of each of the designs. Repeat this analysis for a Xilinx XC4VLX25. *[Same as in the previous assignment.]*

(c) Fully implement the designs (place-and-route) targeting both of the Xilinx FPGAs. Report the maximum operating clock frequencies obtainable. *[This can be found in the place-and-route report, typically the value is given in nanoseconds.]*

(d) How do the clock frequencies change between the estimates given after synthesis and those given after place-and-route? Does ISE tend to over-estimate or under-estimate? Is there a noticeable trend for the accuracy of the estimate as a function of the size of the circuit? Is there a trend between the estimates given for circuits synthesized to the Virtex-II as opposed to the Virtex 4? What other generalizations can you make based on these results?

(e) Design a mechanism by which a clock frequency estimate could be created during synthesis that is consistent with the results from part d). *[This could be in the form of a heuristic algorithm or just a couple of examples of block diagrams with calculated estimates.]*