University of Tehran
Electrical and Computer Engineering Department

Computer Aided Design of Digital Systems (CAD)

CA1: Review on logic design and introduction to FPGA bit-stream generation-phase 1

In this assignment, you will review the concepts of programmable logic design. This assignment has three phases: (1) designing a combinational and sequential logic, (2) modeling the logic with Verilog, and (3) implementing the final design on FPGA-based programmable logic cells.

INTRODUCTION

Multipliers are perhaps the most widely-used unit in digital systems. In this assignment, you first design a basic 2x2 multiplier. It takes two 2-bit unsigned numbers as input and generates a 4-bit result as output.

Like many arithmetic operators, multipliers are fusible: a high order multiplier (e.g. a 4x4 multiplier) can be implemented by fusing (combining) a set of low-order multipliers (e.g. a 2x2 multiplier). Figure 1 shows how a 4x4 multiplier is built with four 2x2 multipliers. The partial products generated by each of the 2x2 units must be shifted and then summed to generate an 8-bit result.

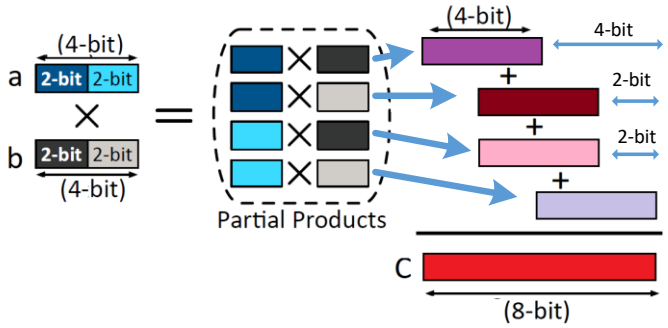

Figure 1. Using four 2x2 multipliers to multiply two 4-bit numbers (a and b) to generate an 8 bit result (c)

Phase 1

PROJECT DESCRIPTION

1. Design a circuit for the 2x2 multiplier. You should consider it as a combination logic with four inputs and four outputs. Use the combinational logic design method your learned in the Logic Design course to implemented it directly with basic gates (and minimized by Karnaugh map). Do not use adders and shifters and other macroblocks.

Figure 2. 2x2 multiplier block diagram

2. Design a 4x4 multiplier using the 2x2 multiplier. For this design, you have to use just one 2x2 multiplier. The 4-bit operands will use this multiplier in four steps. At each step, one half of the first operand and one half of the second operand are multiplied, shifted properly, and accumulated in a register. A control logic will manage the entire procedure. We have two 4-bit registers that hold the input operands and one 8-bit register that holds the multiplication results. A "start" input signal says when the operation must be started. The operation starts as soon as a 0→1 edge on the "start" signal is detected. Once the operation starts (and the system is busy), any 0→1 edge on the "start" signal is ignored until the operation ends.

   Design the control logic by an FSM and then design the entire architecture using the controller, 2x2 multiplier, registers, and other macroblocks if needed (multiplexers, decoders,…).

DELIVERABLES

1. The 2x2 multiplier design in the form of logic functions
2. The FSM of the controller of the 4x4 multiplier (handwritten figure)
3. The architecture of the 4x4 multiplier (handwritten figure). It must be something like Figure 3. Please note Figure 3 is just an example to show you in which details you should draw the architecture.

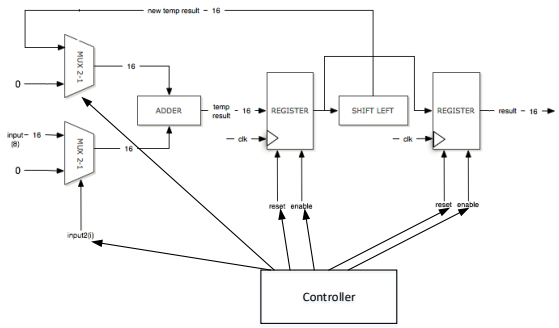

Figure 3. The figure of an example architecture