Homework 1

RTL Design of a 4x4 Multiplier

# Purpose

* The objectives of this homework are:
  + RTL implementation of a 4x4 multiplier for unsigned numbers using combinational logic
  + RTL implementation of the conversion of binary numbers to binary-coded-decimal (BCD) encoder (binary to BCD conversion) using the "shift-add-3" algorithm
  + RTL implementation of the a seven-segment decoder (SSD) using combinational logic.
  + Integration of the above three blocks as shown in the figure below:



Figure 1: Block diagram for the integrated multiplier

* + Validation of the design using testbenches and simulation

# Required tools

Notepad++ (source code editing)

ModelSim PE student editions (simulation)

# Binary to BCD Conversion

Another part of this HW is the binary to BCD conversion. This will be implemented using shift-add-3 algorithm.

## Shift and Add-3 Algorithm

1. Shift the binary number left one bit.
2. If 8 shifts have taken place, the BCD number is in the *Hundreds*, *Tens*, and *Units* column.
3. If the binary value in any of the BCD columns is 5 or greater, add 3 to that value in that BCD column.
4. Go to 1.

**Example 1: Convert hex E to BCD**



**Example 2: Convert hex FF to BCD**



**Truth table for Add-3 Module**



Figure 4: Add3 block

**Binary-to-BCD Converter Module**



Figure 5: 8-bit Binary to BCD Converter

# Seven-segment Decoder (SSD)

Develop the truth table for converting BCD to the 7-segment display decoder. For example, BCD=0000 will correspond to SSD=100000, BCD=0001 will correspond to SSD=1111001, etc.

# Integration

Integrate all components as shown in Figure 1. This should include:

1. One instance of mult4x4,
2. One instance of bcd8
3. And three instances of ssd

## Submission

Name the integrated module as mult4x4\_top.v. Create at testbench the verify the functionality of mult4x4\_top using self-checking testbench. Name this testbench mult4x4\_top\_tb.v