Project 3: Fast Multiplier Design

1. Background
   A. Shift and add multiplier

For more information please refer to pages 268-271 and Figure 6.7.

B. Fast multiplication algorithms

In general, fast multiplication algorithm consists of three parts: 1) partial product generation (PPG), 2) partial product reduction (PPR), and 3) final summation (FS) – see Figure below. Note that all additions within substeps are done in parallel while substeps are executed sequentially. More information on parallel tree adders can be found in lecture notes and online at [http://en.wikipedia.org/wiki/Wallace_tree](http://en.wikipedia.org/wiki/Wallace_tree)
2. Project description

Step A. Implement in Verilog shift-and-add 8x8 unsigned integer multiplier with ripple carry adder

Step B. Implement in Verilog 8x8 unsigned integer parallel multiplier with simple PPG, parallel (Wallace) tree PPR, and ripple carry adder FS stages

Note that Verilog code must be structural, i.e. no behavioral constructs are allowed. Start your code by first creating structural Verilog code for the half adder, full adder, (shift) registers and then build larger components, i.e. adders etc., from these smaller blocks by explicitly connecting them.

3. Deliverables

1) Demonstrate correctness of the operation for two multipliers and answer on related questions (50%)

2) Report including (50%)
   a) general description of the project (40%) including
      - Block level diagram
      - Gate level schematics
      - Verilog code
      - Testbench code with snapshot of testing
      - Performance including gate count and clock cycle time (from timing analysis)

   In addition, report should include the discussion of
   b) potentials for pipelining and improving throughput for two multipliers (10%);
   c) delay (total time), throughput, and area (gate count) scaling as a function of N (10%)

   Up to 20% bonus points might be given for faster adder implementations and partial product encoding.