**Module**: Processor.sv

**Inputs**: Clk, Reset\_Load\_Clear, Run, [9:0] SW

**Outputs**: [6:0] HEX(0-6), [7:0] Aval, [7:0] Bval, Xval

**Description**: This module is the top level of this circuit and is used to instantiate all the modules within it. This includes the control unit, mux, registers, the adder, and the hex drivers.

**Purpose**: This module takes the input signals from the FPGA, directs the signals to perform the necessary operations, and displays the outputting signals on the hex displays.

**Module**: control.sv

**Inputs**: Clk, Reset\_Load\_Clear, Run

**Outputs**: Loadc, ADD, SUB, Shift

**Description**: This module is the control unit for the design. By steeping through the states when called upon, it performs the correct operations in the right order.

**Purpose**: This module starts running when the Run signal turns active low. It then steps through 20 states, alternating between shifting and adding to perform multiplication of 2 numbers.

**Module**: full\_adder.sv

**Inputs**: A, B, cin

**Outputs**: S, cout

**Description**: This is the baseline one-bit full adder. It is composed of 3 inputs, A, B and cin that are all XORed ( *A ^ B ^ cin*) together to obtain S. The operation *(A & B) | (A & cin) | (B & cin)* is used to obtain cout.

**Purpose**: This module is the basis of all the adders above it. To scale the adders to 16 bits, a 4x4 hierarchy of the full adder is used.

**Module**: HexDriver.sv

**Inputs**: [3:0] In0

**Outputs**: [6:0] Out0

**Description**: This module pairs every unique case of 4-bits (16 in total) into a 7-bit value used to illuminate the hex displays.

**Purpose**: This was used to output the values we were working with to the hex displays, both for debugging and for the demo itself.

**Module**: ripple\_adder.sv

**Inputs**: [7:0] A, [7:0] B, operation, subtract

**Outputs**: [8:0] S, Cout

**Description**: This is a 9 bit adder that is used to do both addition and subtraction. A and B are 8-bit inputs to be combined. Operation is a 1-bit value that tells the adder whether to add values, or simply 0. Subtract tells the adder whether to do addition or subtraction.

**Purpose**: This module is called after every shift in order to properly implement multiplication. The subtract 1-bit input allows it to also do subtraction, and the operation input allows it to add o on command.

**Module**: reg\_8.sv

**Inputs**: Clk, Reset, Load, [7:0] D

**Outputs**: [7:0] Data\_Out

**Description**: This module instantiates a 8-bit register using the 8-bit D signal to load the register when Load is high.

**Purpose**: This register is instantiated once in the top-level module and is used to store the values of A and B.

**Module**: xreg (in reg\_4.sv)

**Inputs**: Clk, Reset, Load, D

**Outputs**: Data\_Out

**Description**: This module instantiates a 1-bit register using the 1-bit D signal to load the register when Load is high.

**Purpose**: This register is instantiated once in the top-level module and is used to store the value of X.