**Question 1.**

Design an 8-bit adder/subtracter to add/subtract two eight-bit two's complement numbers. The bottom module will have a one-bit adder/subtracter. It will have four inputs as follows:  
a) I0  
b) I1  
c) carry\_in  
d) mode

And two output signals:  
a) Sum  
b) carry\_out.

mode=1 indicates that it is an addition operation and mode=0 indicates that it is a subtraction operation.

The second module implements adder/subtracter functionality using the one-bit adder/subtracter module as mentioned above. It will take three inputs:  
a) 8-bit Data0  
b) 8-bit Data1  
c) mode

And three output signals:  
a) final\_sum  
b) final\_carry\_out  
c) overflow

Write a test bench to test the eight-bit adder/subtractor. Make sure to display your inputs, sum, and carry out. Your test bench must have fifteen different inputs. Put five-time unit delay between consecutive inputs. Place one module in one Verilog file i.e., you will have three Verilog files.

**Question 2.**

Linear Feedback Shift Registers (LFSRs) with primitive connection polynomials as feedback functions are used as building blocks for many stream ciphers and other cryptographic primitives. They are widely used to generate pseudo-random numbers, in fast digital counters, whitening sequences etc.

An LFSR is a shift register whose input bit is a linear function of its previous state. We generally use XOR as the linear function of multiple bits of the register. The initial value of the LFSR is called the seed. And all possible states of an LFSR can be pre-determined from the given seed as the operation on the register is deterministic. Additionally, an n-bit register can have a finite number of states possible. Hence the LFSR will also have a finite number of states, following a cycle to return back the seed value.

Given this background, first design a D-flipflop to model a 1-bit register. The inputs to the module will be:  
a) D  
b) clk  
c) reset

And the outpu will be:  
a) Q

Then write a top module to implement a 16 bit LFSR using the above mentioned module as shown in the below figure. The inputs to the module will be:  
a) 16 bit seed  
b) clock  
c) reset  
And the output signal will be:  
a) 16-bit state

Write a test bench to go through all possible states of the LFSR. Make sure to display the seed, clock, reset and the state. Put two-time unit delay between every state. Stop the program when the LFSR reaches the initial state, i.e. its seed value. Place one module in one Verilog file i.e., you will have three Verilog files.

