# LAB # 02:MORE ON VERILOG SYNTAX

## Lab Objective:

This lab is in continuation of previous lab to practice Behavioral and data flow modelling using MODELSIM.

## Lab Description:

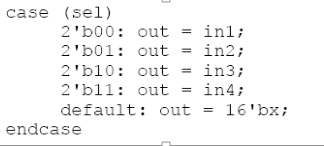
**Initial and Always Procedural Block:**

A procedural block contains one or multiple statements per block. An assignment statement used in a procedural block is called procedural assignment. Initial block executes only once starting at t=0 simulation time, whereas an always block executes continuously at t = 0 and repeatedly thereafter. Characteristics of an initial block are as follows:



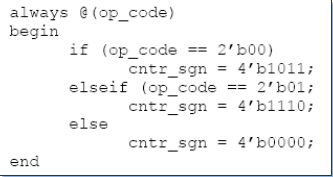
**Case statement:**

Like C and other high level programming languages, Verilog supports switch and case statements for multi way decision support.



**Conditional (if-else) statement:**

Verilog supports the use of conditional statements in behavioral modeling. The if-else statement evaluates the expression and branches to execute the statements in if block if expression is true and in case the expression is false, 0, x or z, the statements in else block is executed.



**The assign statement:**

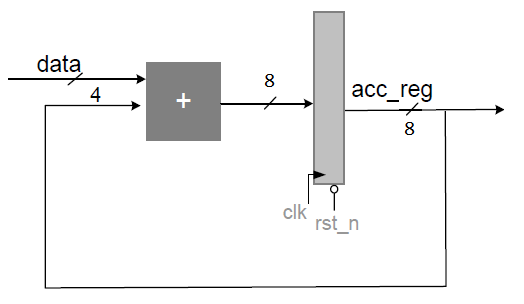
The assign statement is used to make continuous assignment in the dataflow modeling. The assign statement usage is given below:

*Assign out = in0 + in1; // in0 + in1*

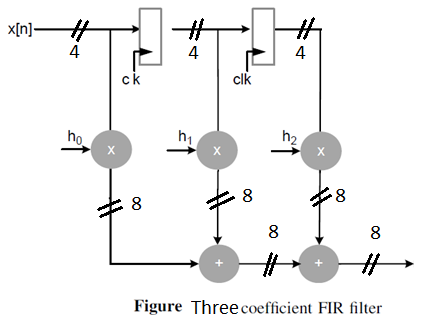
This is evaluated and then assigned to out. Please note that the LHS of assign statement must always be a scalar or vector net or a concatenation, it cannot be a register. Continuous statements are always active statements, which mean that if any value on the RHS changes the value of LHS changes automatically. The RHS expression is evaluated whenever one of its operands changes. Then the result is assigned to the LHS.

## Lab Tasks:

1. Design the accumulator in the figure bellow. Create the test bench module of it and simulate and show the results.

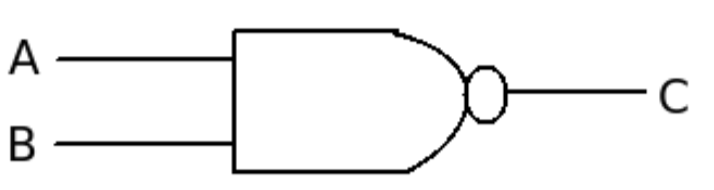
****

1. Design (3 coefficients) FIR filter in figure bellow. Where are the 4-bit inputs, clk is the clock signal, and is 8-bit output. Simulate the design module and show the timing diagrams. Use [ho = 1, h1 = 2, h2 = 1] during the simulation, change the input values and observe the output.

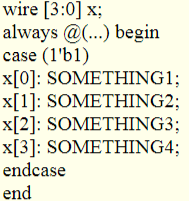


## Test Plans:

1. Write the correct way of entity representation for the two input NAND gate shown below?



1. What will be the output of following piece of code?



## THINK!!

1. What is the difference between:

c = foo ? a : b;

and

if (foo) c = a;

1. What is the difference between wire and reg?
2. To understand and practice VERILOG syntax and simulation process.