Verilog (contd..)

Behavioral modelling

- It describes the digital system at algorithmic level or abstract level or functional level.
- It describes what to do with circuit rather than how it is built on hardware.
- It enables design of complex ASIC consisting of millions of gates

Procedural blocks in behavioural modelling

- always block – it is executed in a loop subjected to some condition. It is used to describe the behaviour of both combinational and sequential circuits.
- Initial block – It is executed once, and is used to set the initial conditions in test bench code.
- In procedural assignments, the left hand side (target output) must be reg data type.
- Two keywords – begin and end are used in defining the procedural block.
- Multiple always blocks can be written, and they run concurrently with all other continuous assignment statements (using assign keyword).

Wire vs. Reg

Wire

- Cannot store value without being driven.
- assign keyword is used in LHS.
- Usually used to model combinational circuits.

Reg

- Can store the value until a new value is assigned.
- Can be used in procedural block.
- Can be used to model both combinational and sequential circuits.

Procedural Block Describing a Majority Circuit

```
module maj3 ( input a, b, c, output reg y );
    always @( a or b or c )
    begin
        y = (a & b) | (b &c) | (a & c);
    end
endmodule
```

```
module mux2_1 (input i0, i1, output reg s, y );
   always @( i0 or i1 or s ) begin
      if ( s==1'b0 )
         y = i0;
      else
         y = i1;
   end
endmodule
```

Figure 3.42 Procedural Multiplexer

```
module alu_4bit (input [3:0] a, b, input [1:0] f, output
                 reg [3:0] y );
   always @ ( a or b or f ) begin
      case ( f )
         2'b00 : y = a + b;
         2'b01 : y = a - b;
         2'b10 : y = a & b;
         2'b11 : y = a ^ b;
         default: y = 4'b0000;
      endcase
   end
endmodule
```

Figure 3.43 Procedural ALU

Combinational Rules

Completion of **case** alternatives or **if-else** conditions is an important issue in combinational circuit coding. In an **always** block, if there are conditions under which the output of a combinational circuit is not assigned a value, because of the property of **reg** variables the output retains its old value. The retaining of old value infers a latch on the output. Although, in some designs this latching is intentional, obviously it is unwanted when describing combinational circuits. With this, we have set two rules for coding combinational circuits with **always** blocks.

1. List all inputs of the combinational circuit in the sensitivity list of the **always** block describing it.
2. Make sure all combinational circuit outputs receive some value regardless of how the program flows in the conditions of **if-else** and/or **case** statements. If there are too many conditions to check, set all outputs to their inactive values at the beginning of the **always** block.