SC1005 Digital Logic
Tutorial 9

Q1. (a) Explain what is wrong with the following code for a counter, and correct it:

```
module countwrong (input clk, rst, output [5:0] cnt_out);

always@*
begin
cnt_out = cnt_out + 1'b1;
end
endmodule
```

(b) The following combinational module is to be converted into a synchronous module. Add a register after each combinational stage in the original description, by rewriting the module using only a single synchronous always block:

```
module arch1 (input [6:0] a, b, output [13:0] total);

wire [6:0] int1;

assign int1 = a + b;
assign total = int1 * int1;

endmodule
```

Q2. A monitoring circuit has an input, *evnt*, that is high whenever a certain condition is met. It has an internal counter that counts the number of cycles in which *evnt* is high. When this count exceeds a threshold, determined by the 6-bit *thresh* input, it sounds an alarm by asserting the alarm output and stops the counter. The human operator can then check for problems and reset the system by asserting *rst*. Design a Verilog module that implements this circuit. (Hint: alarm should be a combinational circuit determined from the count value and threshold. It can be used to determine whether or not the counter counts in any given cycle).

Q3. (a) Write a Verilog description of a 5-bit binary counter that counts up to 20 and then wraps round to zero.

(b) Modify the counter in (a), adding a new 5-bit input, *countmax*. The counter should now wrap around at the *countmax* value.

(c) Write a Verilog module that implements a 6-bit counter that counts down from an initial value. The initial value should be loaded on reset from a 6-bit input, *start_val*.