Exercise 9 (Inferring Flops)(HW2 prob 5):

Below is the implementation of a D-latch

```
module latch(d,clk,q);
  input d, clk;
  output reg q;

  always @(clk)
    if (clk)
      q <= d;

endmodule
```

Is this code correct? If so why does it correctly infer and model a latch? If not, what is wrong with it?

Submit to the Exercise09 dropbox a single file called **HW2_prob5.sv** with as much of this as you can finish during class. You will have to complete this for HW2 dropbox.

a. The comments **should answer** the questions about the latch posed above.

b. The file should contain the model of a D-FF with an active high synchronous reset.

c. The file should contain the model of a D-FF with asynchronous active low reset and an active high enable.

(continued next slide)

Exercise 9 (Inferring Flops)(HW2 prob 5):

d. The file should contain the model of a SR FF with active high synchronous reset, and an active high synchronous set. SR in this case **does not mean** some crazy cross coupled NOR gate thing you learned in a previous digital design course.

SR meaning it has a **S** input that will set the flop, and a **R** input that will reset the flop, and it maintains state if neither **S** or **R** are high. If both **S** and **R** were asserted **R** would have priority. It **also has active low async reset**. This is a handy style flop that we will use frequently.

e. I would like you to use the **always_ff** construct of System Verilog. The file should contain *(as comments)* the **answer to this question**: Does the use of the **always_ff** construct ensure the logic will infer a flop? Looking for a little more than a simple yes/no answer.