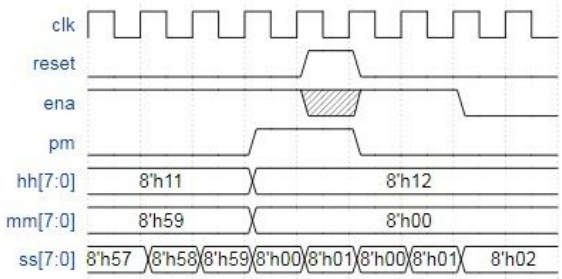
Count clock

countbcd Previous Next shift4

Create a set of counters suitable for use as a 12-hour clock (with am/pm indicator). Your counters are clocked by a fast-running `clk`, with a pulse on `ena` whenever your clock should increment (i.e., once per second).

`reset` resets the clock to 12:00 AM. `pm` is 0 for AM and 1 for PM. `hh`, `mm`, and `ss` are two BCD (Binary-Coded Decimal) digits each for hours (01-12), minutes (00-59), and seconds (00-59). Reset has higher priority than enable, and can occur even when not enabled.

The following timing diagram shows the rollover behaviour from `11:59:59 AM` to `12:00:00 PM` and the synchronous reset and enable behaviour.



https://hdlbits.01xz.net/wiki/Count_clock HDLbits-Counter Clock

```
module top_module(
    input clk,
    input reset,
    input ena,
    output pm,
    output [7:0] hh,
    output [7:0] mm,
    output [7:0] ss);

    wire ena_ss, ena_mm, ena_hh, ena_pm;
    assign ena_ss = ena;
    assign ena_mm = ena & ss==8'h59;   //sec=59
    assign ena_hh = ena_mm & mm==8'h59;//min,sec=59'59
    assign ena_pm = ena_hh & hh==8'h11;//hhmmss 115959

    always@(posedge clk)begin
        if(reset) begin
            ss[7:0]<=8'h0;
        end
        else if(ena_ss) begin
            ss[3:0]<= (ss[3:0]==4'h9)?4'h0:ss[3:0]+4'h1;
            ss[7:4]<= (ss[3:0]==4'h9)?(ss[7:4]==4'h5)?4'h0:ss[7:4]+4'h1:ss[7:4];
        end
        else begin
            ss <= ss;
        end
    end

    always@(posedge clk)begin
```

```
        if(reset) begin
            mm[7:0]<=8'h0;
        end
        else if(ena_mm) begin
            mm[3:0]<= (mm[3:0]==4'h9)?4'h0:mm[3:0]+4'h1;
            mm[7:4]<= (mm[3:0]==4'h9)?(mm[7:4]==4'h5)?4'h0:mm[7:4]+4'h1:mm[7:4];
        end
        else begin
            mm <= mm;
        end
    end

    always@(posedge clk)begin
        if(reset) begin
            hh[7:0]<=8'h12;
        end
        else if(ena_hh) begin
            hh[3:0]<= (hh[3:0]==4'h9)?4'h0:(hh[7:0]==8'h12)?4'h1:hh[3:0]+4'h1;
            hh[7:4]<= (hh[3:0]==4'h9)?4'h1:(hh[7:0]==8'h12)?4'h0:hh[7:4];
        end
        else begin
            hh <= hh;
        end
    end

    always@(posedge clk)begin
        if(reset) begin
            pm<=1'b0;
        end
        else if(ena_pm) begin
            pm <= ~pm;
        end
        else begin
            pm <= pm;
        end
    end

endmodule
```