ELD Lab 3
Design of 8-bit Counter

Objective

- Design 8-bit Up counter using behavioral modelling
- For counter to increment every second, design 1 Hz clock from input 100 MHz clock using clock divider
- Verify the counter using virtual input and output (VIO).
- **Lab Homework:** Change the counter to count downwards

Verilog Revision

Lab

Proposed Approach

8-bit Counter

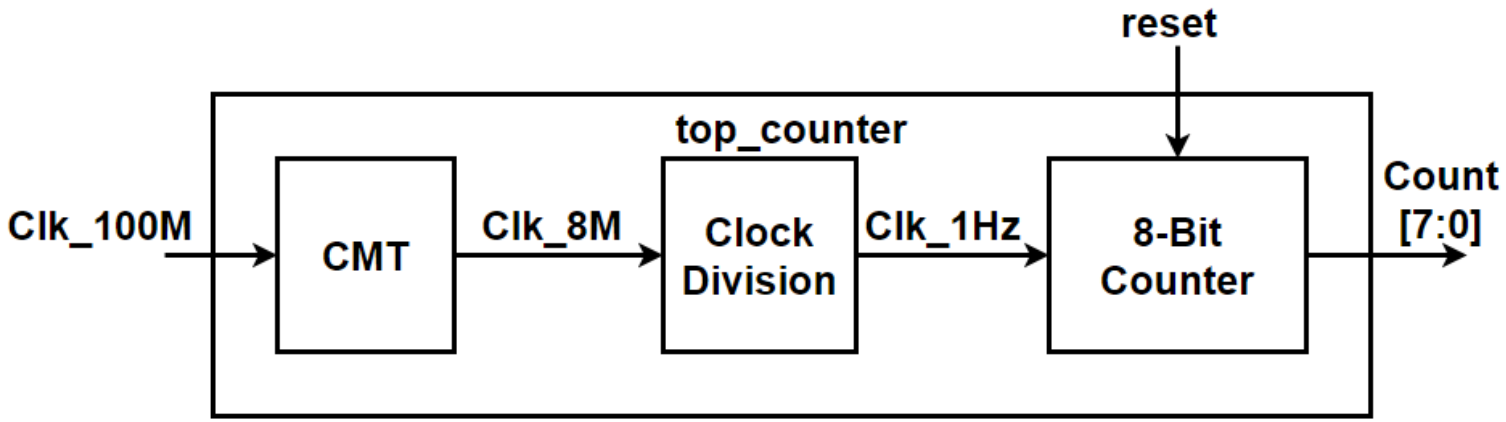
```
module Counter_8bit(
    input Clk_1Hz,
    input reset,
    output [7:0] Count
    );
    reg [7:0] Count_reg=0;
    reg [7:0] Count_next;
    always@(posedge Clk_1Hz or posedge reset)
    begin
        if(reset)
            Count_reg <= 0;
        else
            Count_reg <= Count_next;
    end
    always@(*)
    begin
        Count_next = Count_reg + 1;
    end
    assign Count = Count_reg;
endmodule
```

Counter with Clock Division

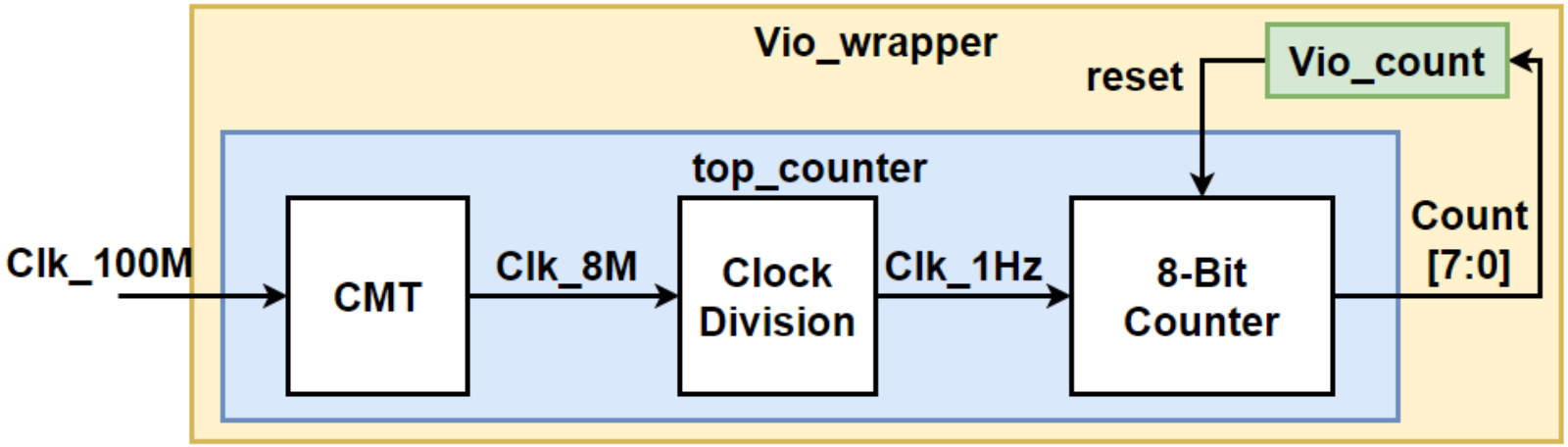
```
module top_counter(
    input Clk_100M,
    input reset,
    output [7:0] Count
    );
    wire Clk_8M;
    clk_div_cmt cd
   (
    // Clock out ports
    .Clk_8M(Clk_8M),     // output Clk_8M
   // Clock in ports
    .Clk_100M(Clk_100M));

     wire Clk_1Hz;
    // This modules divide the input clock by 2^(COUNT_DIV_FACTOR+1)
    // (8x10^6)/2^(23) -> 1 Hz
    clk_div_rtl #(.COUNT_DIV_FACTOR(22)) clk_div_rtl1(.reset(reset),.clk_in(Clk_8M),.clk_out(Clk_1Hz))
    Counter_8bit Cn(.Clk_1Hz(Clk_1Hz), .reset(reset),.Count(Count));

endmodule
```



VIO Wrapper




```
module Vio_wrapper(
    input Clk_100M
    );

    wire reset;
    wire [7:0] Count;
    vio_count v1 (
  .clk(Clk_100M),                  // input wire clk
  .probe_in0(Count),     // input wire [7 : 0] probe_in0
  .probe_out0(reset)  // output wire [0 : 0] probe_out0
);


  top_counter tc(.Clk_100M(Clk_100M),.reset(reset),.Count(Count));

endmodule
```

Demo

- Add XDC file and generate bitstream
- Verify the functionality using VIO