Pulse With Modulation

```
:ale 1ns/1ps
 pwm_tb;
 clk;
 rst;
 [7:0] duty;
'e pwm_opt;

 Instantiate DUT (Device Under Test)
 #(8) uut (
  .clk(clk),
  .rst(rst),
  .duty(duty),
  .pwm_opt(pwm_opt)

.tial clk = 0;
ways #10 clk = ~clk;

.tial begin
  // Initialize dump
  $dumpfile("pwm_tb.vcd");
  $dumpvars(0, pwm_tb);

  // Reset
  rst = 1;
  duty = 8'd0;
  #20 rst = 0;

  // Apply different duty cycles
  duty = 8'd64;   // 25% duty
  #6000;
  duty = 8'd128;  // 50% duty
  #6000;
  duty = 8'd192;  // 75% duty
  #6000;
  duty = 8'd255;  // ~100% duty
  #6000;
  $finish;
.tial begin
  $monitor("Time=%0t clk=%b rst=%b duty=%d pwm=%b",
 clk, rst, duty, pwm_opt);
ile
```

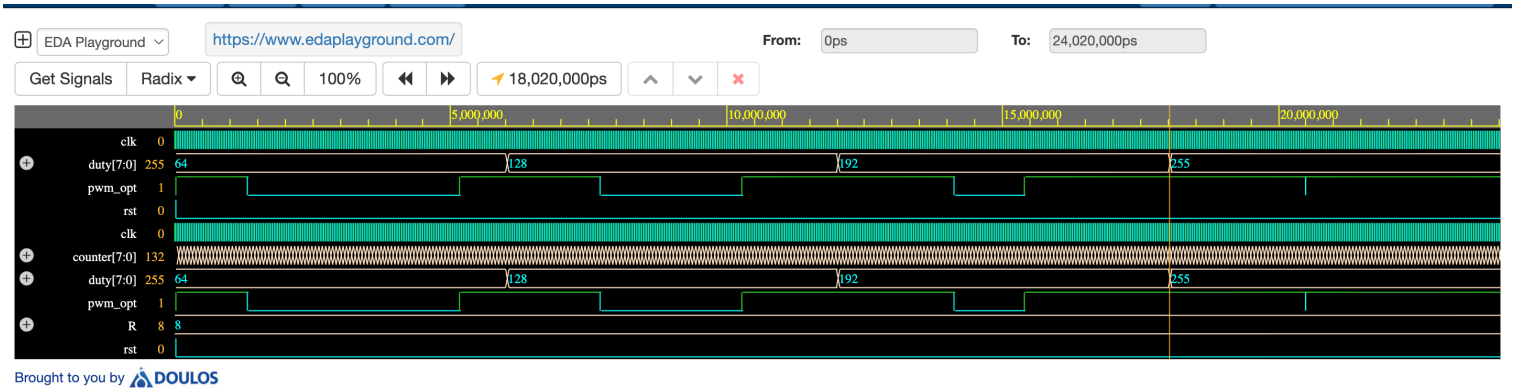
Playground LIVE! SystemVerilog Static and Automatic Lifetimes".

```
module pwm #(parameter R=8)(
  input clk,
  input rst,
  input [R-1:0]duty,
  output reg pwm_opt
);
  reg [R-1:0]counter;

   always @(posedge clk or posedge rst) begin
        if (rst) begin
            counter  <= {R{1'b0}};
            pwm_opt  <= 1'b0;
        end else begin
            // Counter increment
            if (counter == {R{1'b1}})
                counter <= {R{1'b0}};
            else
                counter <= counter + 1;

            // PWM generation
            pwm_opt <= (counter < duty) ? 1'b1 : 1'b0;
        end
    end

endmodule
```



Brought to you by DOULOS