verilogA/va_counter.va

```
// VerilogA for wk_carrerobardon, va_t_counter_17_e, veriloga

`include "constants.vams"
`include "disciplines.vams"

module va_t_counter_17_e(clock,n_reset,mask,enable_count);

output clock;
electrical clock;

output [4:0] mask;
electrical [4:0] mask;

output n_reset;
electrical n_reset;

output enable_count;
electrical enable_count;

parameter real vlogic_high = 1.2;
parameter real vlogic_low = 0;
parameter real vtrans = 0.6;
parameter real tdel = 1e-12;
parameter real tfall = 50e-12;
parameter real trise = 50e-12;

parameter real clk_period = 50n;
integer mask_bit_enabled = 0;
parameter real reset_period = 45u;
parameter real enable_count_period = 20u;
real next_mask_change = 0.0003000;

genvar k;
integer j = 0;
integer i = 0;
integer mask_out [4:0];
integer clock_state = 1;
integer n_reset_state = 1;verilog
integer enable_count_state = 1;
real n = 0;

analog begin


@(initial_step) begin
    for(k = 0; k < 5 ; k = k + 1) begin
        mask_out[j] = j == mask_bit_enabled ? 1 : 0; // generate correct mask
        j = j + 1;
    end
    mask_bit_enabled = -1;
end

@(timer(0,clk_period / 2)) begin
    clock_state = !clock_state; // generate clock
end



//@(timer(0,reset_period / 2)) begin
//  n_reset_state = !n_reset_state; // generate reset signal for test
//end

//@(timer(0,enable_count_period / 2)) begin
//  enable_count_state = !enable_count_state; // generate eable count for test
//end

@(timer(0,next_mask_change)) begin
    j = 0;
    mask_bit_enabled = mask_bit_enabled + 1;
    for(k = 0; k < 5 ; k = k + 1) begin
        mask_out[j] = j == mask_bit_enabled ? 1 : 0; // generate correct mask
        j = j + 1;
    end // generate mask for test
```

```
        next_mask_change = 0.0003 * (2 ** (n)) - $abstime ;
        n = n + 1;

    end



    V(clock) <+ transition(clock_state*vlogic_high, tdel, trise, tfall);
    V(n_reset) <+ transition(n_reset_state*vlogic_high, tdel, trise, tfall);
    V(enable_count) <+ transition(enable_count_state*vlogic_high, tdel, trise, tfall);
    generate i (0, 4, 1 ) V(mask[i]) <+ transition(mask_out[i]*vlogic_high, tdel, trise,
    tfall);


    end

    endmodule
```