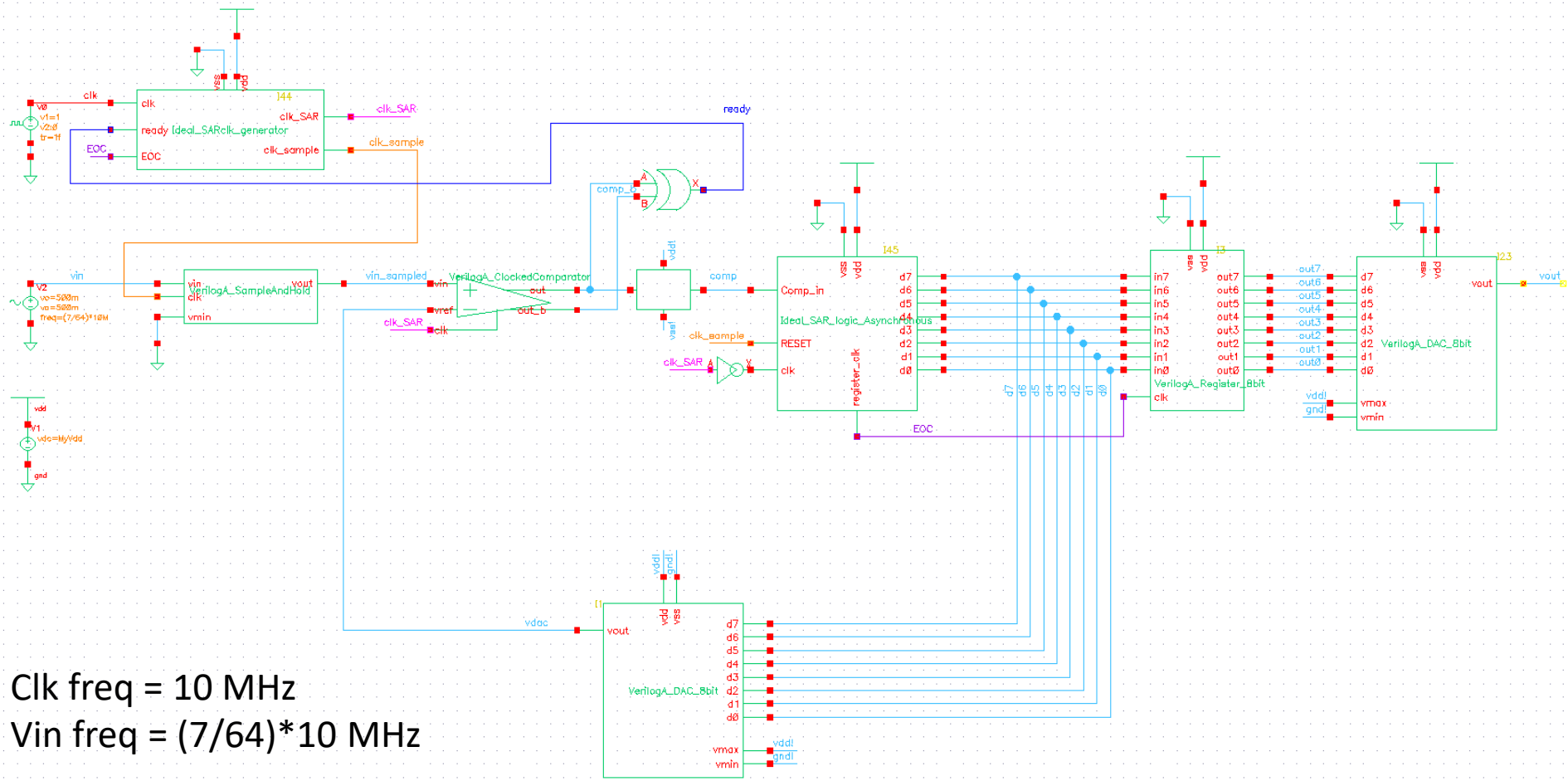
VerilogA Model for 8-bit Asynchronous SAR ADC

MUHAMMAD ALDACHER

3/8/2020

Asynchronous SAR ADC

Asynchronous SAR ADC

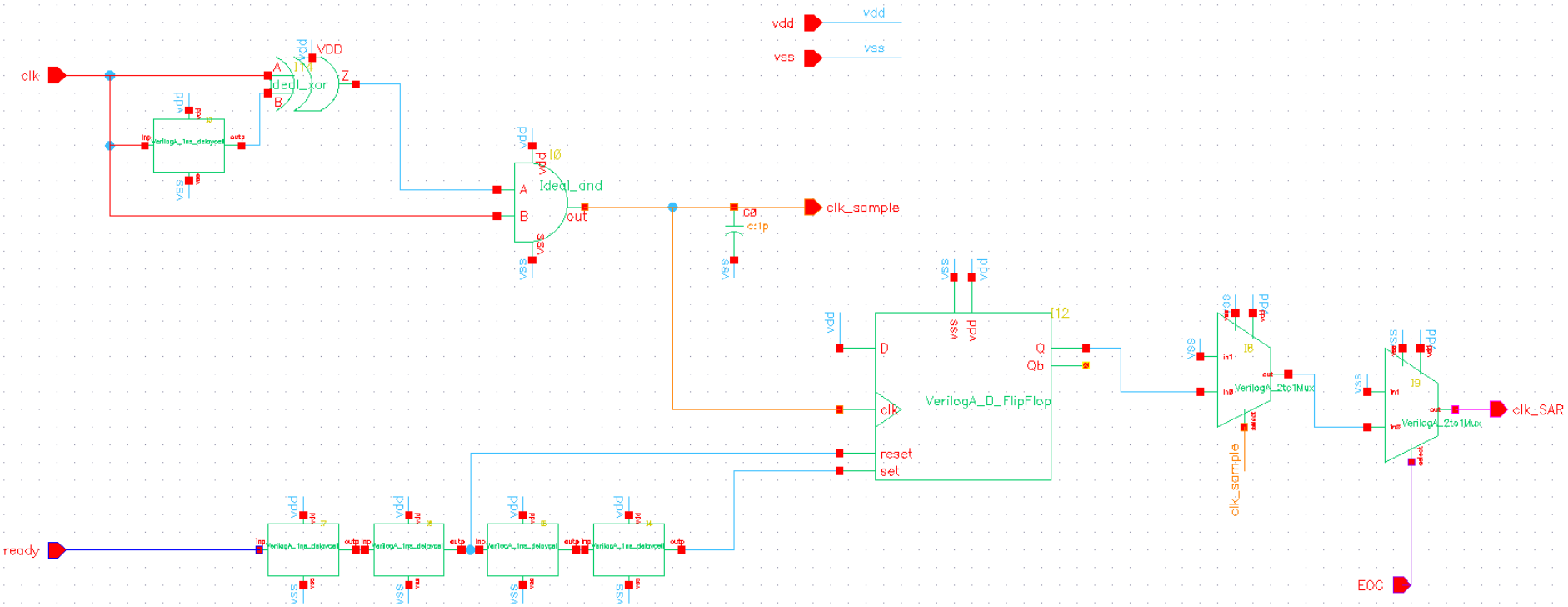


Clk freq = 10 MHz

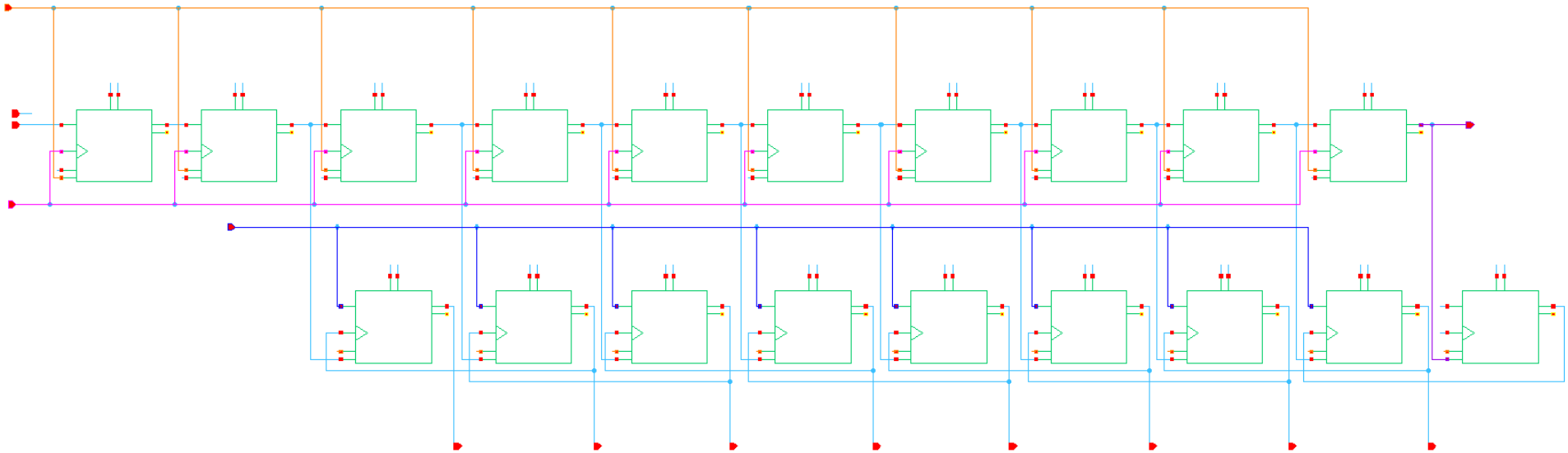
Vin freq = (7/64)*10 MHz

Output = 10 Msamples/sec

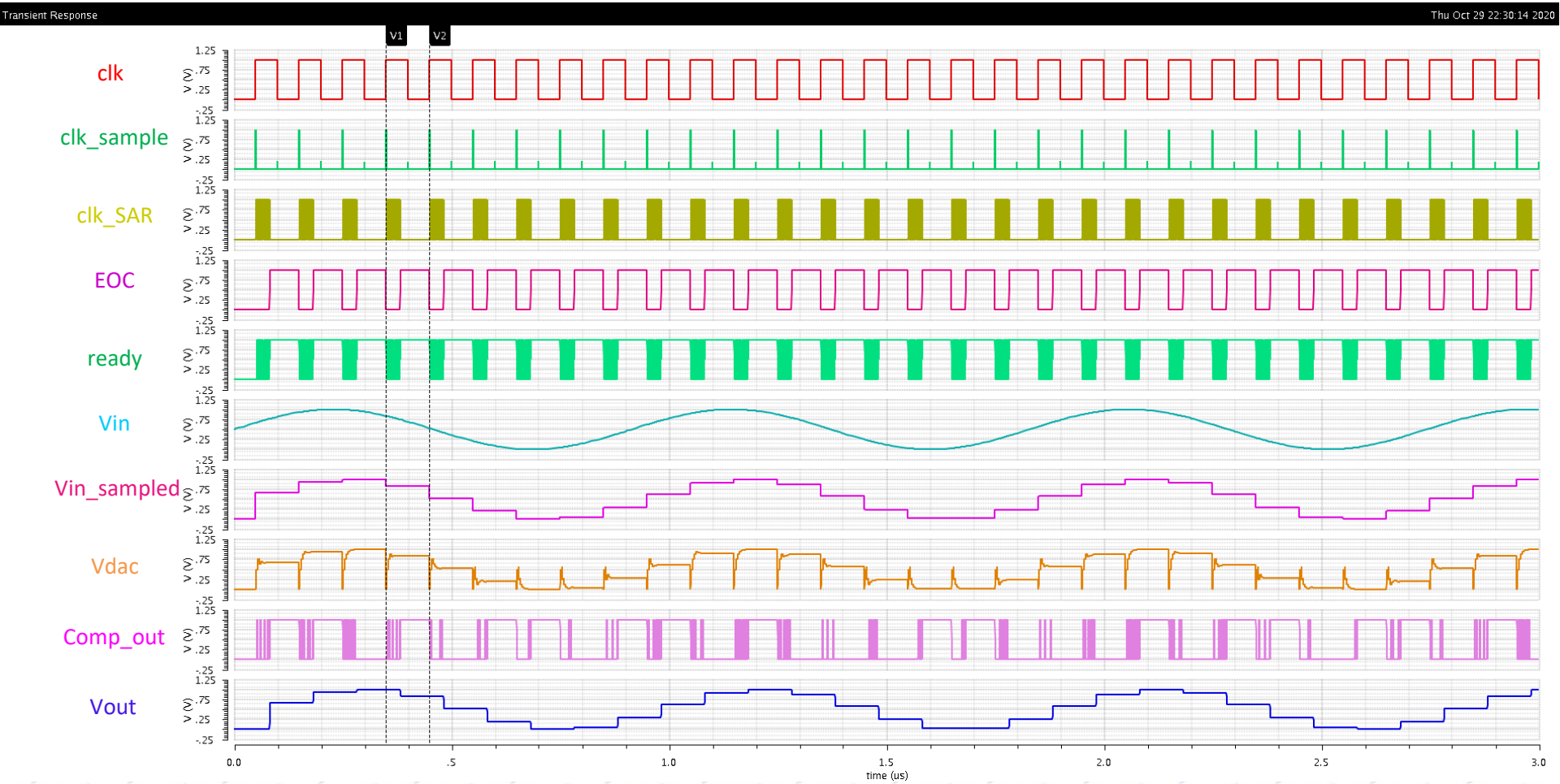
Clock-Signals Generator



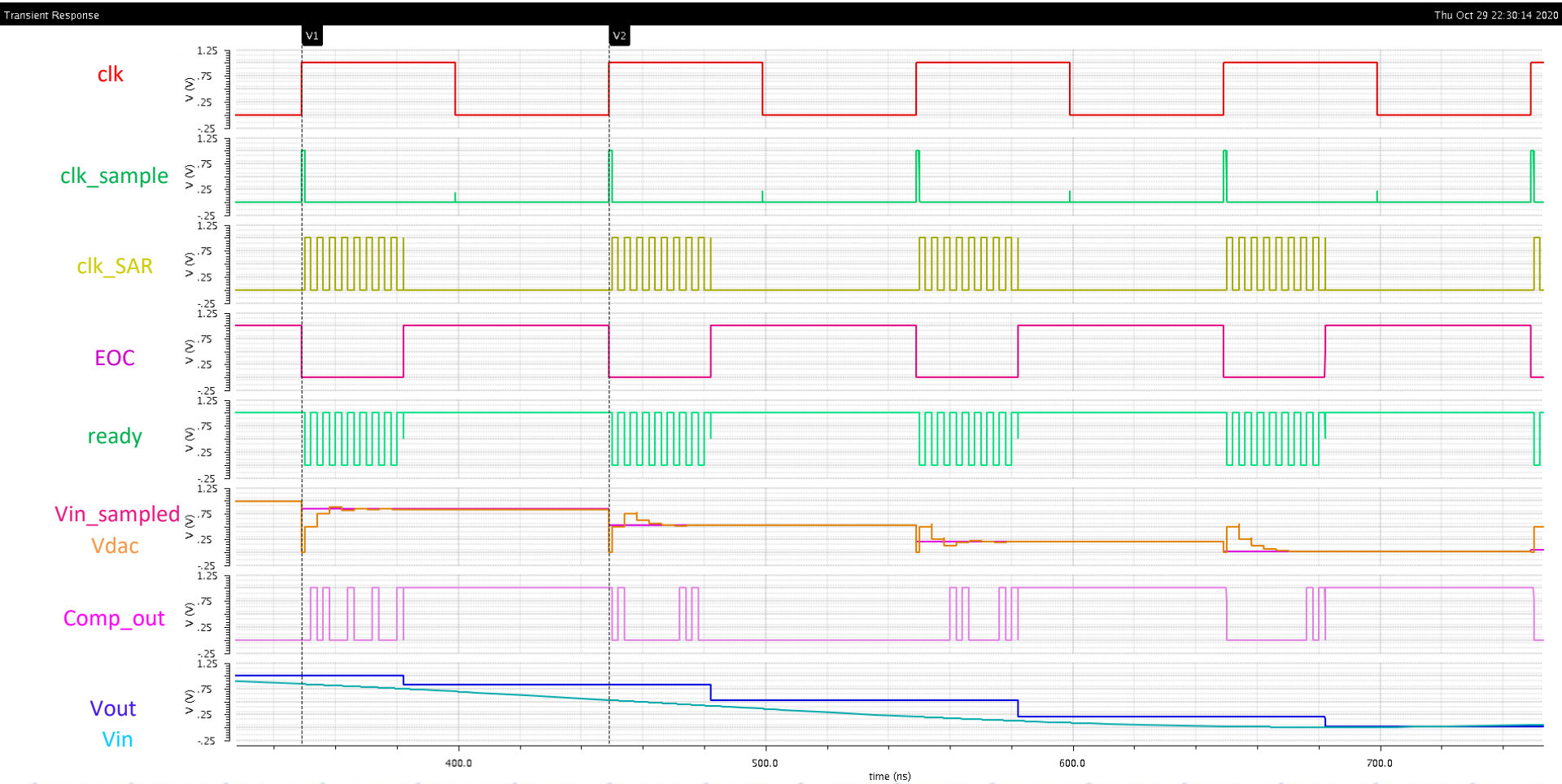
Asynchronous SAR Logic Block



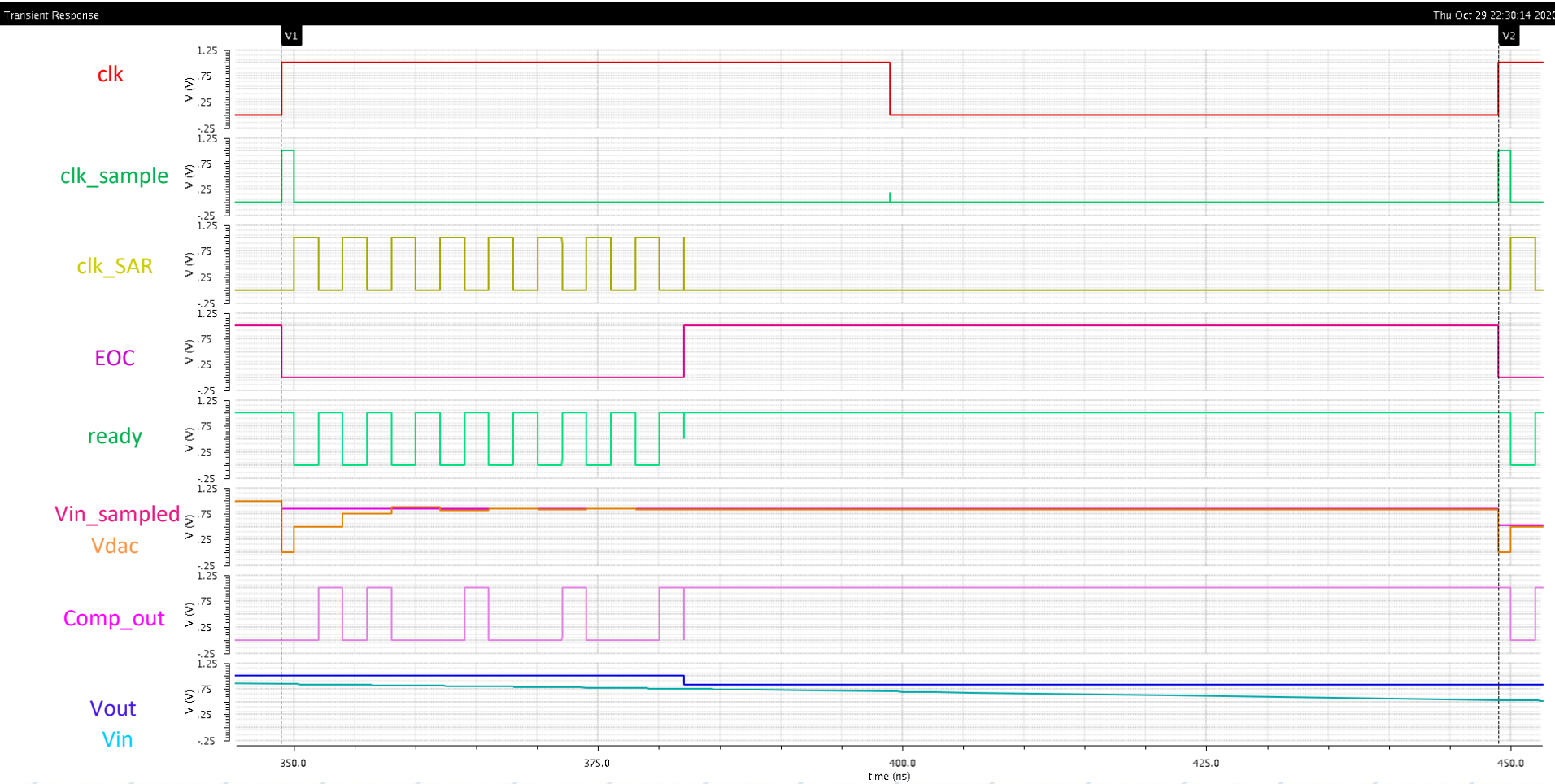
Waveforms


Transient Response
Thu Oct 29 22:30:14 2020
V1
V2
clk
clk_sample
clk_SAR
EOC
ready
Vin
Vin_sampled
Vdac
Comp_out
Vout
1.25
.75
.25
-.25
V (V)
0.0
.5
1.0
1.5
2.0
2.5
3.0
time (us)

Waveforms


Transient Response
Thu Oct 29 22:30:14 2020
V1
V2
clk
clk_sample
clk_SAR
EOC
ready
Vin_sampled
Vdac
Comp_out
Vout
Vin
V (V)
1.25
.75
.25
-.25
400.0
500.0
600.0
700.0
time (ns)

Waveforms (1 sample cycle)



VerilogA Codes

1) Sample-And-Hold

```
// VerilogA for SampleAndHold

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

module VerilogA_SampleAndHold(clk,vin,vmin,vout);

parameter real vtrans=0.5;
parameter real delay = 0;
parameter real ttime = 1p;
parameter real clk_threshold = 0.5;                    //vdd is 1v

input clk,vin,vmin;
output vout;

electrical vout,vin,vmin,clk;

real v;

analog begin
        // Sampling Phase (+1 is for rising edge, -1 is for falling edge)
        @(cross(V(clk) - clk_threshold, +1))
                        v = V(vin);

        V(vout) <+ transition(v,delay,ttime);

end
endmodule
```

2) Clocked Comparator (differential output)

```
// VerilogA for ClockedComparator

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

module VerilogA_ClockedComparator(out,out_b,vref,vin,clk);

parameter real clk_th=0.5;
parameter real delay = 0;
parameter real ttime = 1p;

input vin,vref,clk;
output out,out_b;

electrical out,out_b,vref,vin,clk;
real d_result, d_result_b;

analog begin
        @(cross(V(clk) - clk_th, +1)) begin
                if(V(vin) > V(vref)) begin
                        d_result = 1;
                        d_result_b = 0;
                end
                else begin
                        d_result = 0;
                        d_result_b = 1;
                end
        end
        @(cross(V(clk) - clk_th, -1)) begin
                        d_result = 0;
                        d_result_b = 0;
        end

        V(out) <+ transition(d_result,delay,ttime);
        V(out_b) <+ transition(d_result_b,delay,ttime);
end

endmodule
```

3) Output Register (8-bit)

```
// VerilogA for SAR_VerilogA, VerilogA_Register_8bit, veriloga

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

module
VerilogA_Register_8bit(clk,in0,in1,in2,in3,in4,in5,in6,in7,out0,out1,out
2,out3,out4,out5,out6,out7,vdd,vss);

parameter real vtrans=0.5;
parameter real delay = 0;
parameter real ttime = 1p;
parameter real clk_threshold = 0.5;

inout vdd,vss;
input clk,in0,in1,in2,in3,in4,in5,in6,in7;
output out0,out1,out2,out3,out4,out5,out6,out7;

electrical
clk,in0,in1,in2,in3,in4,in5,in6,in7,out0,out1,out2,out3,out4,out5,out6,o
ut7,vdd,vss;
real d_0,d_1,d_2,d_3,d_4,d_5,d_6,d_7;

analog begin
        @(cross(V(clk) - clk_threshold, +1))
        begin
                d_7 = V(in7);
                d_6 = V(in6);
                d_5 = V(in5);
                d_4 = V(in4);
                d_3 = V(in3);
                d_2 = V(in2);
                d_1 = V(in1);
                d_0 = V(in0);
        end

        V(out7) <+ transition(d_7,delay,ttime);
        V(out6) <+ transition(d_6,delay,ttime);
        V(out5) <+ transition(d_5,delay,ttime);
        V(out4) <+ transition(d_4,delay,ttime);
        V(out3) <+ transition(d_3,delay,ttime);
        V(out2) <+ transition(d_2,delay,ttime);
        V(out1) <+ transition(d_1,delay,ttime);
        V(out0) <+ transition(d_0,delay,ttime);

end
endmodule
```

4) DAC (8-bit)

```
// VerilogA for VerilogA_DAC_8bit

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

module
VerilogA_DAC_8bit(d0,d1,d2,d3,d4,d5,d6,d7,vout,vdd,vss,vmin,vmax);

parameter real vtrans=0.5;
parameter real delay = 0;
parameter real ttime = 1p;

inout vdd,vss;
input d0,d1,d2,d3,d4,d5,d6,d7;
input vmin, vmax;
output vout;

electrical vout,vdd,vss,d0,d1,d2,d3,d4,d5,d6,d7,vmin,vmax;

real result,d_0,d_1,d_2,d_3,d_4,d_5,d_6,d_7;

analog begin
    d_7 = V(d7)*128;
    d_6 = V(d6)*64;
    d_5 = V(d5)*32;
    d_4 = V(d4)*16;
    d_3 = V(d3)*8;
    d_2 = V(d2)*4;
    d_1 = V(d1)*2;
    d_0 = V(d0)*1;

    result = ((d_7+d_6+d_5+d_4+d_3+d_2+d_1+d_0) * ((V(vmax)-
V(vmin))/(256))) + V(vmin) ;

    V(vout) <+ transition(result,delay,ttime);
end


endmodule
```

* D Flipflop (with asynchronous Set & Reset)

```
// VerilogA for SAR_VerilogA, VerilogA_D_FlipFlop, veriloga
`include "constants.vams"
`include "disciplines.vams"

module VerilogA_DFlipFlop(D,clk,Q,Qb,set,reset,vdd,vss);

parameter real vtrans=0.5;
parameter real delay = 0;
parameter real ttime = 1p;
parameter real clk_threshold = 0.5;

inout vdd,vss;
input D,clk,set,reset;
output Q,Qb;

electrical D,clk,Q,Qb,set,reset,vdd,vss;

real d_0,d_0b;

analog begin

  @(cross(V(set) - vtrans, +1)) begin
    d_0 = 1;
    d_0b = 0; end

  @(cross(V(reset) - vtrans, +1)) begin
    d_0 = 0;
    d_0b = 1; end

  @(cross(V(clk) - clk_threshold, +1))
  begin
    if((V(set) < vtrans) && (V(reset) < vtrans) ) begin
      if(V(D) > vtrans) begin
      d_0 = 1; d_0b = 0; end

      else begin
      d_0 = 0; d_0b = 1; end
    end
  end


  V(Q) <+ transition(d_0,delay,ttime);
  V(Qb) <+ transition(d_0b,delay,ttime);
end

endmodule
```

* Inverter

```
// VerilogA for SAR_VerilogA_Asynchronous, VerilogA_inv, veriloga

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

module VerilogA_inv(A,X);

parameter real vtrans=0.5;
parameter real delay = 0;
parameter real ttime = 1p;
parameter real clk_threshold = 0.5;

input A;
output X;
electrical A,X;

real X_out;

analog begin

    if(V(A) > vtrans) begin
      X_out = 0; end

    else begin
      X_out = 1; end

    V(X) <+ transition(X_out,delay,ttime);
end

endmodule
```

* XOR

```
// VerilogA for SAR_VerilogA, VerilogA_XOR, veriloga

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

module VerilogA_XOR(A,B,X);

parameter real vtrans=0.5;
parameter real delay = 0;
parameter real ttime = 1p;
parameter real clk_threshold = 0.5;

input A,B;
output X;

electrical A,B,X;

real X_out;

analog begin

    if(V(A) == V(B)) begin
      X_out = 0; end

    else begin
      X_out = 1; end

    V(X) <+ transition(X_out,delay,ttime);
end

endmodule
```

* Delay Cell (1ns)

```
// VerilogA for SAR_VerilogA_Asynchronous, VerilogA_1ns_delaycell,
veriloga

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

module VerilogA_1ns_delaycell(inp,outp,vdd,vss);

parameter real vtrans=0.5;
parameter real delay = 1n;
parameter real ttime = 1f;

inout vdd,vss;
input inp;
output outp;

electrical inp,outp,vdd,vss;
real d_out;

analog begin
  @(cross(V(inp) - vtrans, +1))
  begin
    d_out = 1;
  end
  @(cross(V(inp) - vtrans, -1))
  begin
    d_out = 0;
  end

                V(outp) <+ transition(d_out,delay,ttime);
end

endmodule
```

* 2-to-1 Mux

```
// VerilogA for SAR_VerilogA_Asynchronous, VerilogA_2to1Mux, veriloga
`include "constants.vams"
`include "disciplines.vams"

module VerilogA_2to1Mux(in0,in1,select,out,vdd,vss);

parameter real vtrans=0.5;
parameter real delay = 0;
parameter real ttime = 1p;
parameter real clk_threshold = 0.5;

inout vdd,vss;
input in0,in1;
output out;

electrical in0,in1,select,out,vdd,vss;
real result;

analog begin
        if(V(select) < vtrans) begin
                result = V(in0); end
        else begin
                result = V(in1); end

        V(out) <+ transition(result,delay,ttime);

end
endmodule
```