

# PLL Design and Verilog-A Behavioral Modeling

Followed the PLL design procedure described in the “pll design survey.pdf”

SOFTWARE TOOL: CADENCE VIRTUOSO

Omama Elrefaei | 1 May 2020

## Verilog-A model for the PFD:

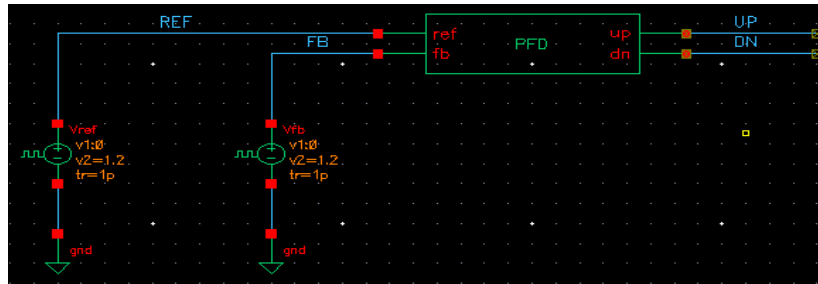
// VerilogA for Lab\_5, PFD, veriloga

```
`include "constants.vams"
`include "disciplines.vams"
module PFD(ref,fb,up,dn);
  input ref, fb;
  output up, dn;
  electrical ref, fb, up, dn;
  parameter real vh = 1.2;
  parameter real vl = 0;
  parameter real vth = 0.6;
  parameter real ttol = 5p from [0:inf);
  parameter real td = 0 from [0:inf);
  parameter real tt = 1p from (0:inf);
  parameter real ton = 20p from[0:inf);
  integer state, hide_state;
  real Toff;
  analog begin
    @(initial_step) begin
      state=0;
      hide_state = 0;
      Toff = 0;
    end
    @(cross(V(ref)-vth, 1, ttol)) begin
      if (state == -1) begin
        hide_state = 1;
        Toff = ton + $realtime;
      end
      else state = 1;
    end
    @ (cross(V(fb)-vth, 1, ttol)) begin
      if (state == 1) begin
        hide_state = 1;
        Toff = ton+$realtime;
      end
      else state = -1;
    end
    @ (timer(Toff)) begin
      if (hide_state == 1) begin
        state = 0;
        hide_state = 0;
      end
    end
    V(up) <+ transition((state == 1)|| (hide_state == 1) ? vh:vl, td, tt);
    V(dn) <+ transition((state == -1)|| (hide_state == 1) ? vh:vl, td, tt);
  end
endmodule
```



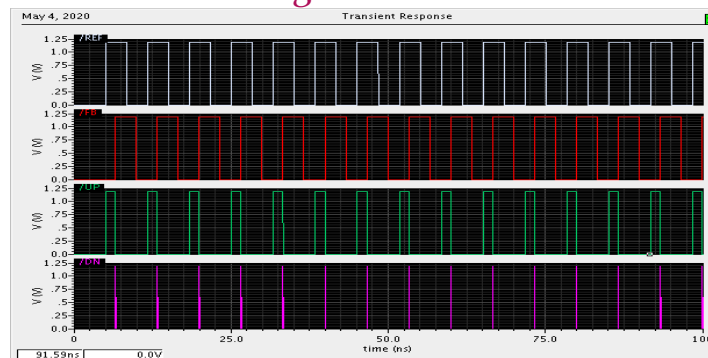
PFD Symbol

## A simple testbench to test the PFD:

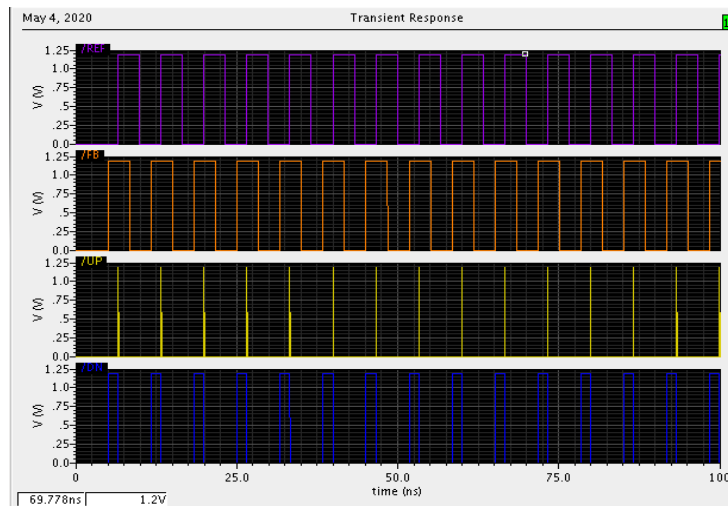


$V_1 = 0\text{ V}$ ,  $V_2 = 1.2\text{ V}$ ,  $\text{Freq} = 150\text{MHz}$ ,  $\text{trise} = \text{tfall} = 1\text{ ps}$ .

## PFD timing when REF is leading:



## PFD timing when FB is leading:



## Verilog-A model for the CHP:

// VerilogA for Lab\_5, CHP, veriloga

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

```

`include "disciplines.vams"
module CHP(siginc, sigdec, vout, vsrc);
  input siginc, sigdec;
  inout vout, vsrc;
  electrical siginc, sigdec, vout, vsrc;
  parameter real iamp = 10u from [0:inf];
  parameter real vtrans = 0.6;
  parameter real tdel = 0 from [0:inf];
  parameter real trise = 1p from (0:inf);
  parameter real tfall = 1p from (0:inf);
  real iout_val;

```

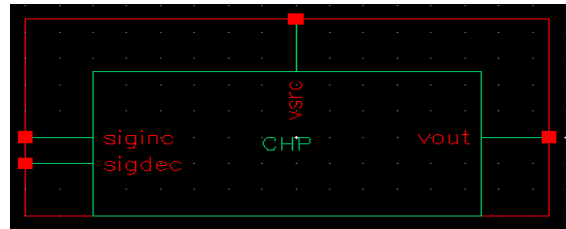
// Current multiplier - returns direction that charge should be pumped

```

analog function real i_mult;
  input inc;
  input dec;
  input vtrans;
  real inc;
  real dec;
  real vtrans;
  integer inc_high;
  integer dec_high;
begin
  inc_high = inc > vtrans;
  dec_high = dec > vtrans;
  i_mult = 0.0;
  if (inc_high == dec_high) begin
    i_mult = 0.0;
  end else if (inc_high) begin
    i_mult = 1.0;
  end else if (dec_high) begin
    i_mult = -1.0;
  end
end
endfunction
analog begin
  @ ( initial_step ) begin
    iout_val = iamp*i_mult(V(siginc), V(sigdec), vtrans);
  end
  @ (cross(V(siginc) - vtrans, 0)) begin
    iout_val = iamp*i_mult(V(siginc),V(sigdec),vtrans);
  end
  @ (cross(V(sigdec) - vtrans, 0)) begin
    iout_val = iamp*i_mult(V(siginc),V(sigdec),vtrans);
  end
end

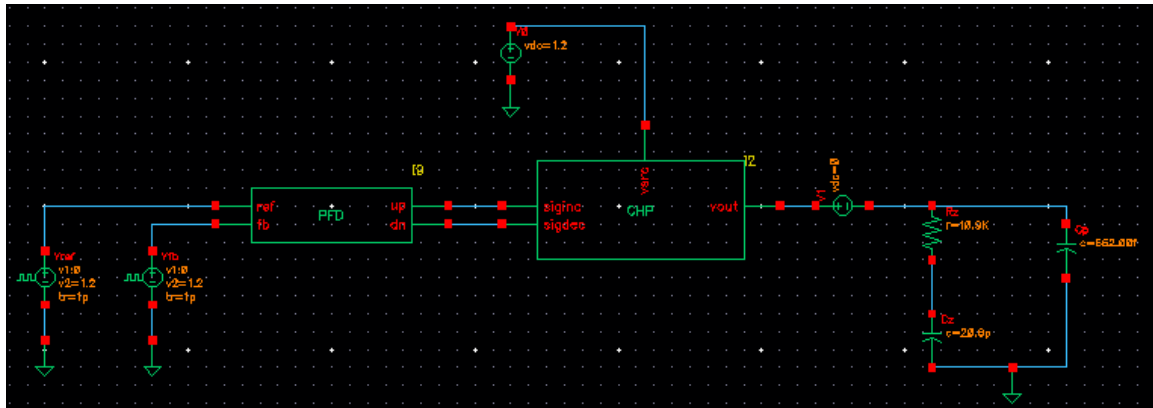
I(vsrc, vout) <+ transition(iout_val,tdel,trise,tfall);
end
endmodule

```



CHP Symbol

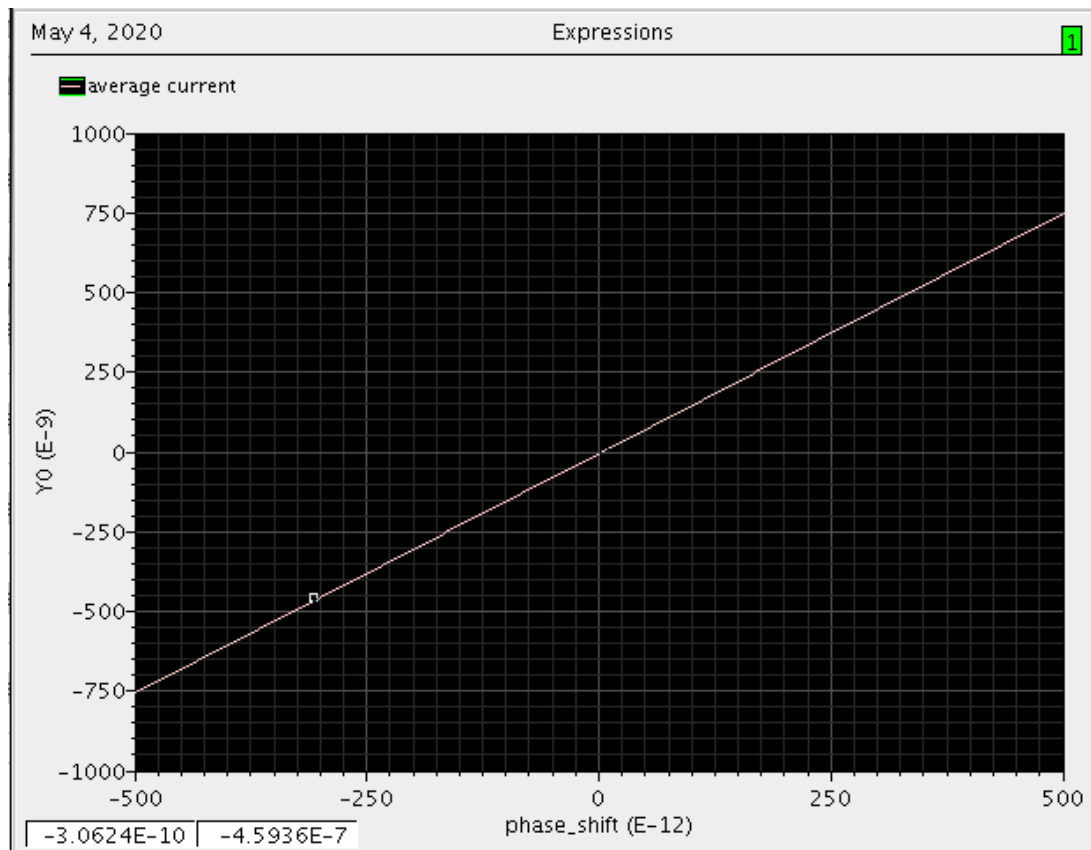
A simple testbench to test the CHP:



$V_1 = 0\text{ V}$ ,  $V_2 = 1.2\text{ V}$ , Freq = 150MHz, trise = tfall = 1 ps.

$R_z = 10.9\text{ Kohm}$ ,  $C_z = 20.6\text{ pf}$ ,  $C_p = 0.662\text{ pf}$ .

CHP average output current vs. phase error:



## Verilog-A model for the VCO:

// VerilogA for Lab\_5, VCO, veriloga

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

module VCO (out, in);
  input in; voltage in;           // input terminal
  output out; voltage out;       // output terminal
  parameter real vl = 0;         // low output voltage
  parameter real vh = 1.2;       // high output voltage
  parameter kvco = 1.9G;         //VCO gain
  parameter real tt = 1p from (0:inf); // output transition time
  parameter real ttol = 0.1p from (0:inf); // time tolerance
  parameter center_freq = 1k;
  //assuming that at Vin = 0, the VCO will work at low frequency=1KHz
  real freq, phase;
  integer n;

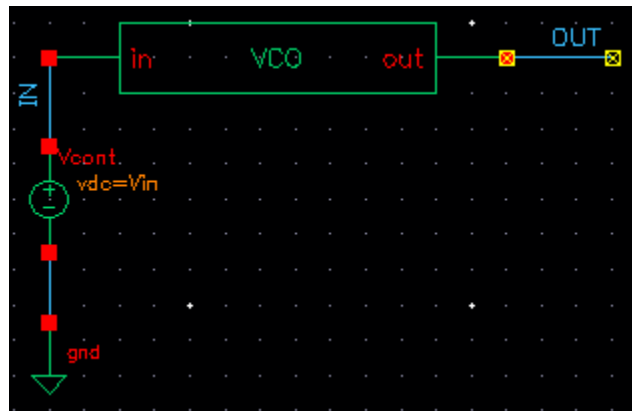
  analog begin
    // compute the freq from the input voltage, it depends on Kvco
    //rather that Vmax, Vmin, Fmax, Fmin
    freq = (kvco*V(in)) + center_freq;
    // bound the time step to assure no cycles are skipped
    $bound_step(0.1 / freq);
    // phase is the integral of the freq modulo 2pi
    phase = 2*M_PI*idtmod(freq, 0.0, 1.0, -0.5);

    // identify the point where switching occurs
    @(cross(phase + `M_PI/2, +1, ttol) or cross(phase - `M_PI/2, +1, ttol))
      n = (phase >= -`M_PI/2) && (phase < `M_PI/2);
    // generate the output
    V(out) <+ transition(n ? vh : vl, 0, tt);
  end
endmodule
```

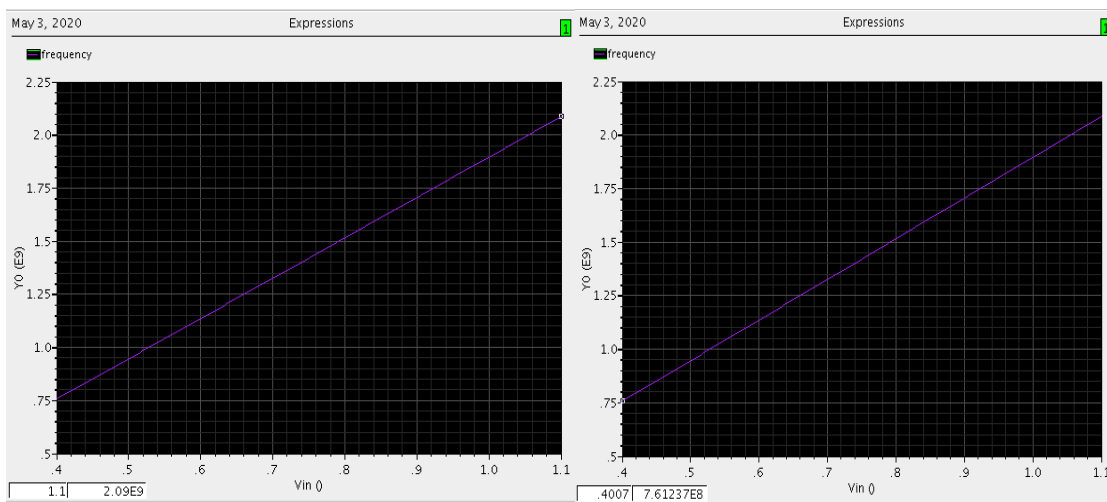


VCO Symbol

A simple testbench to test the VCO:



VCO tuning curve:



$$\text{Slope} = (2.09\text{E}9 - 7.61237\text{E}8) / (1.1 - 0.4001) = 1.90013299\text{E}9 = K_{\text{VCO}}$$

Verilog-A model for the divider:

// VerilogA for Lab\_5, Divider, veriloga

```

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

module Divider (out, in);

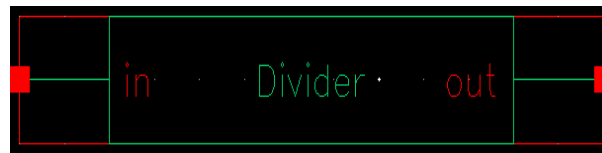
    output out; voltage out;                // output
    input in; voltage in;                   // input (edge triggered)
    parameter real vh = 1.2;                // output voltage in high state
    parameter real vl = 0;                  // output voltage in low state
    parameter real vth = 0.6;               // threshold voltage at input
    parameter integer ratio = 8 from [2:inf); // the required divider ratio

    // dir=1 for positive edge trigger, dir=-1 for negative edge trigger
    parameter integer dir = 1 from [-1:1] exclude 0;
    parameter real tt = 1p from (0:inf);    // transition time of output signal
    parameter real td = 0 from [0:inf);     // average delay from input to output
    integer count, n;

    analog begin
        @(cross (V(in) - vth, dir)) begin
            count = count + 1; // count input transitions
            if (count >= ratio)
                count = 0;
            n = (2*count >= ratio);
        end
        V(out) <+ transition(n ? vh : vl, td, tt);
    end

endmodule

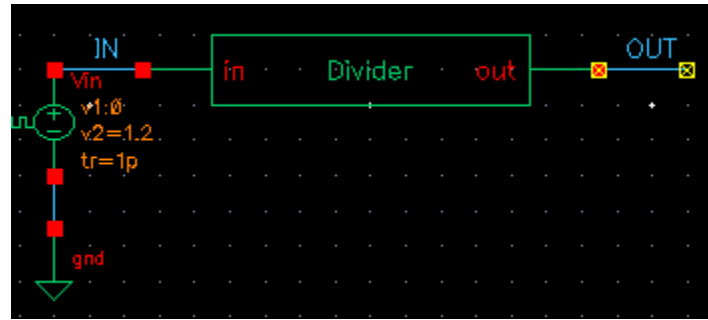
```



Divider Symbol

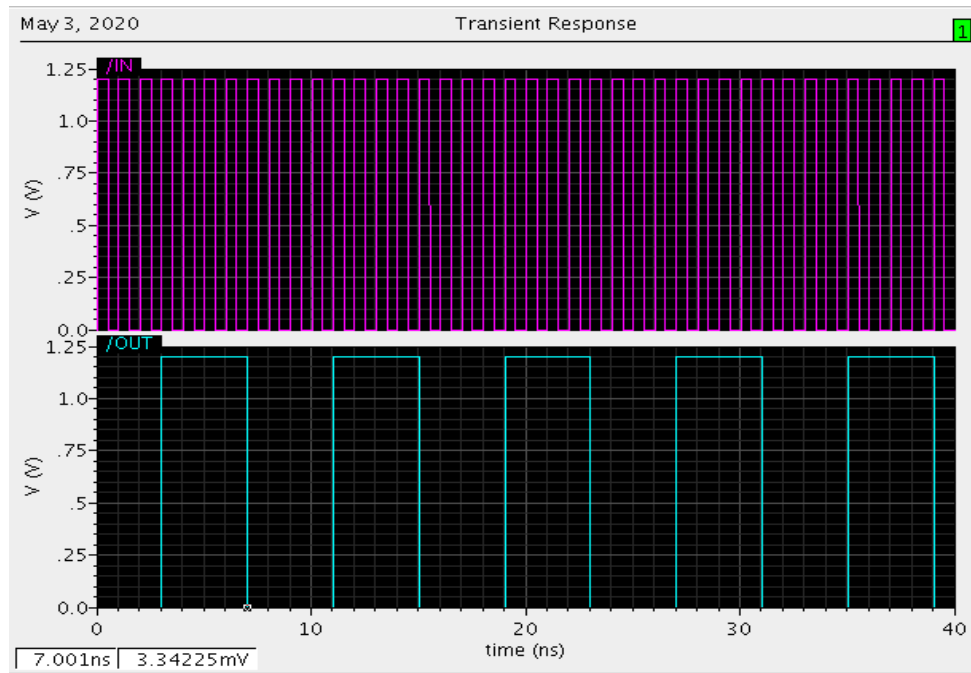
A simple testbench to test the divider:





$V_1 = 0\text{ V}$ ,  $V_2 = 1.2\text{ V}$ ,  $\text{Freq} = 100\text{MHz}$ ,  $\text{trise} = \text{tfall} = 1\text{ ps}$ .

Frequency divider transient simulation:

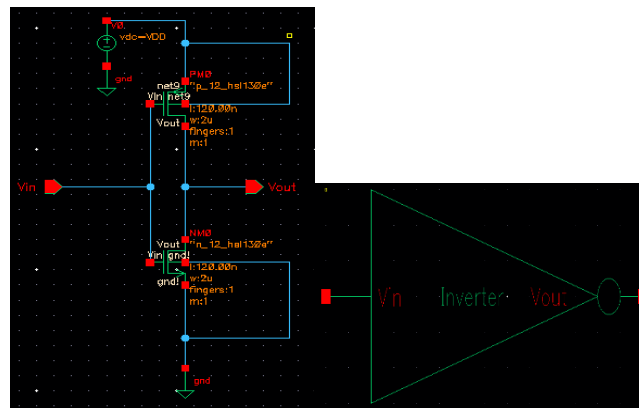


$\text{Period (out)} = 8 * \text{Period (in)}$

$\text{Freq (out)} = 1/8 * \text{Freq (in)}$

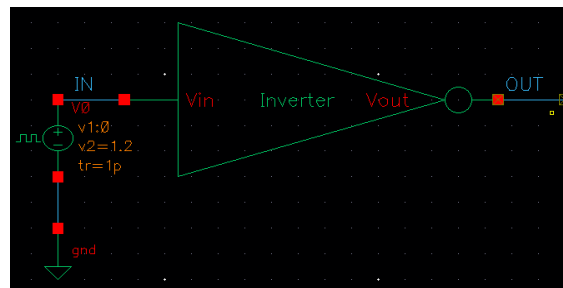
## The divider at the transistor level:

### Inverter:



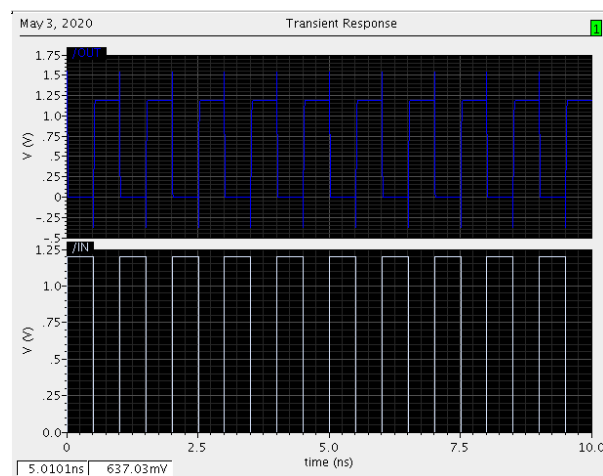
Symbol

### Inverter Testbench:



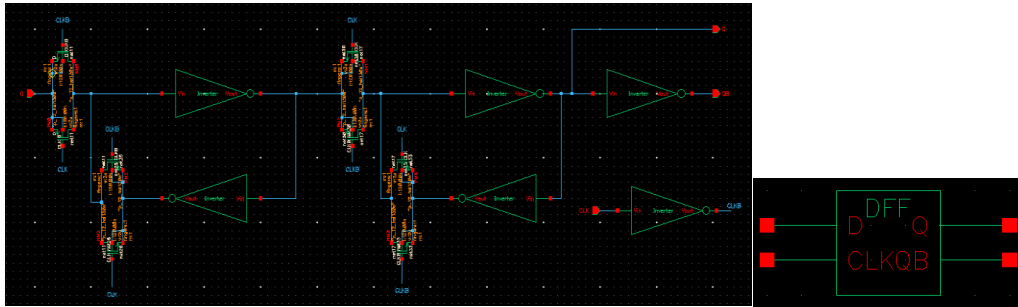
Schematic

### Inverter transient simulation:



## The divider at the transistor level:

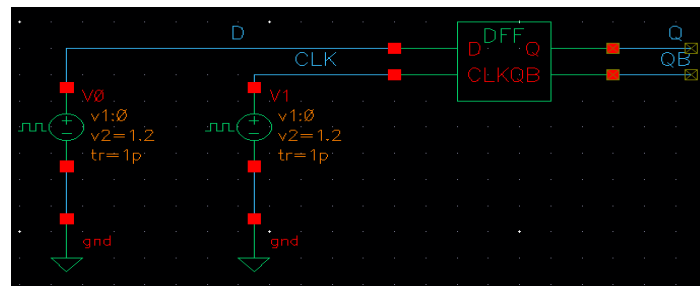
### DFF:



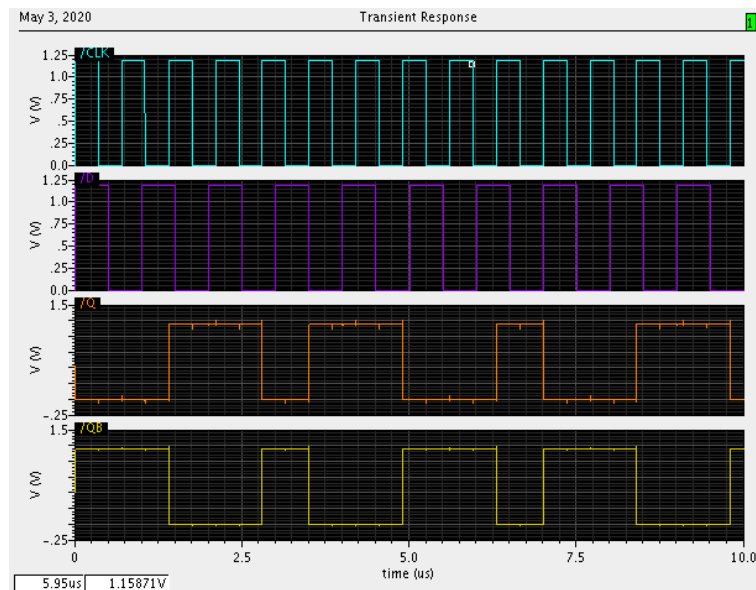
Schematic

Symbol

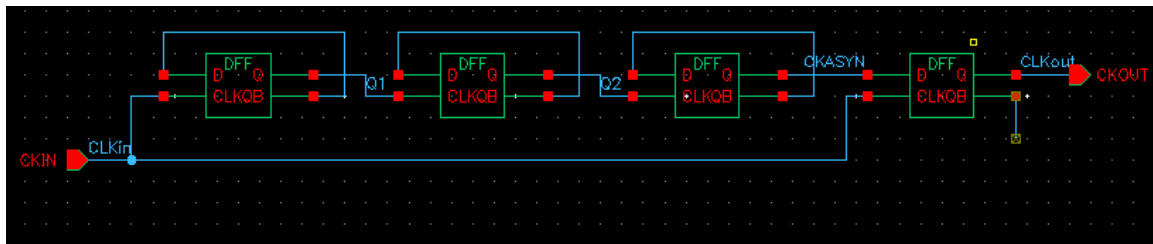
### DFF Testbench:



### DFF transient simulation:



## The divider at the transistor level:

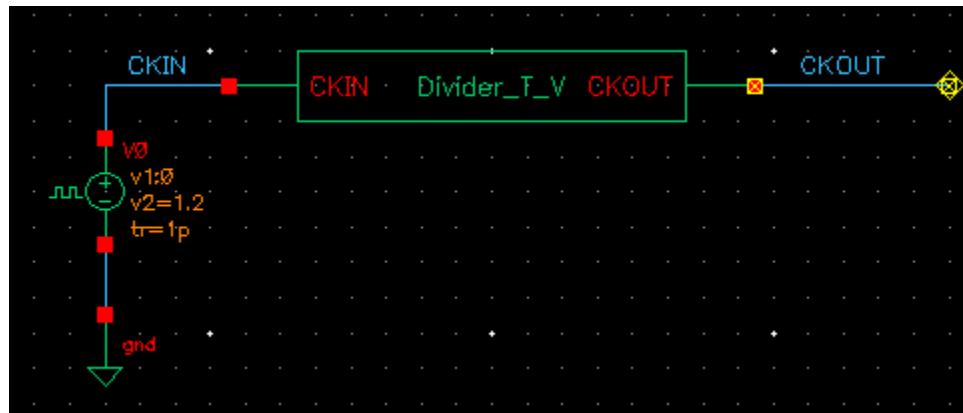


Schematic



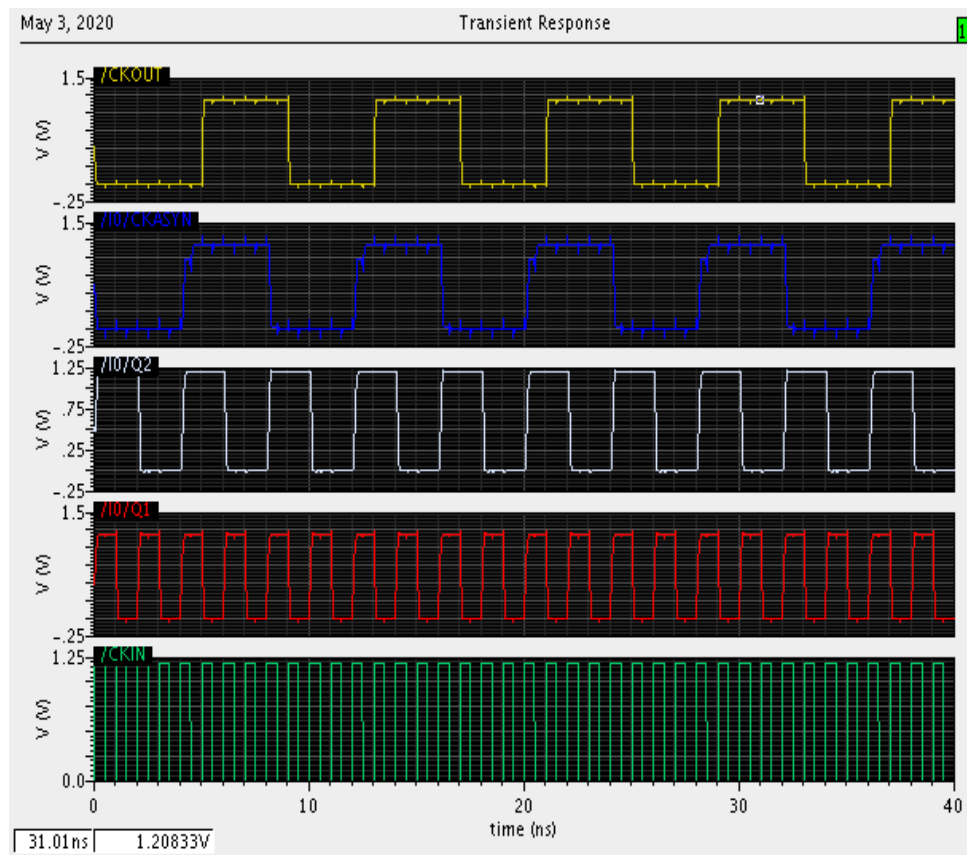
Symbol

## Testing the transistor level divider:

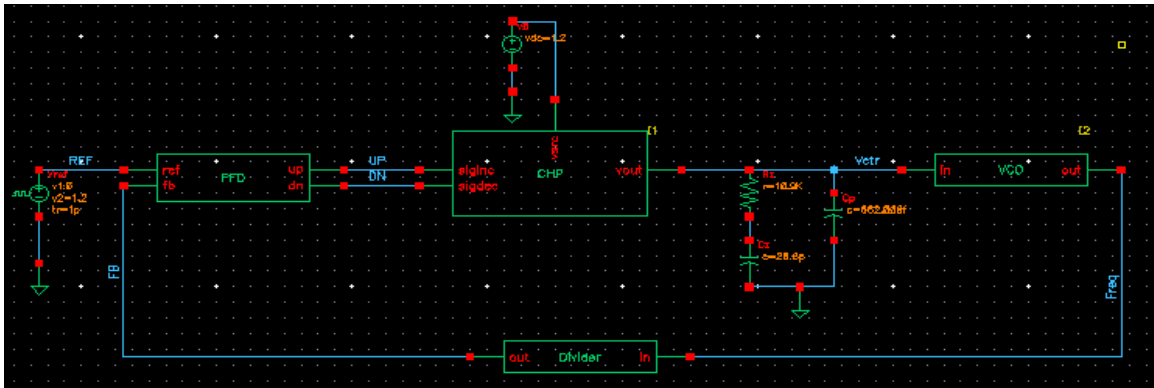


$V_1 = 0$  V,  $V_2 = 1.2$  V, Freq = 100MHz, trise = tfall = 1 ps.

## Frequency divider transistor level transient simulation:



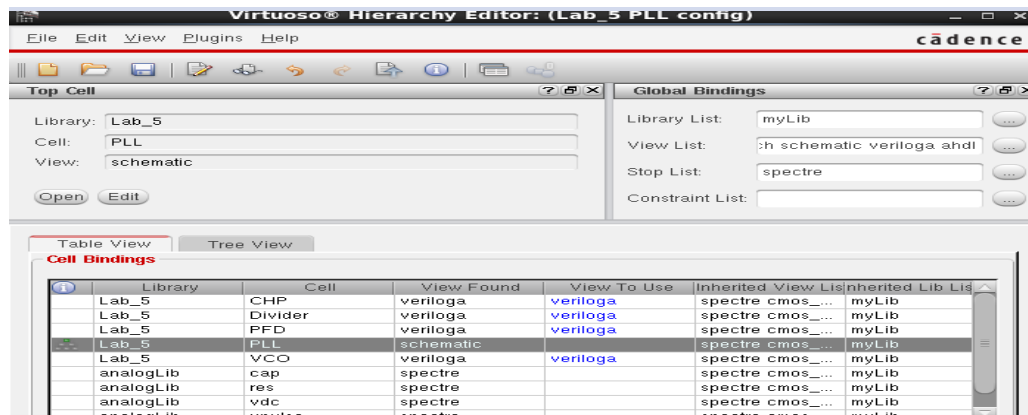
## The PLL simulation using config view and hierarchy editor:



$V_1 = 0\text{ V}$ ,  $V_2 = 1.2\text{ V}$ ,  $\text{Freq} = 150\text{ MHz}$ ,  $\text{trise} = \text{tfall} = 1\text{ ps}$ .

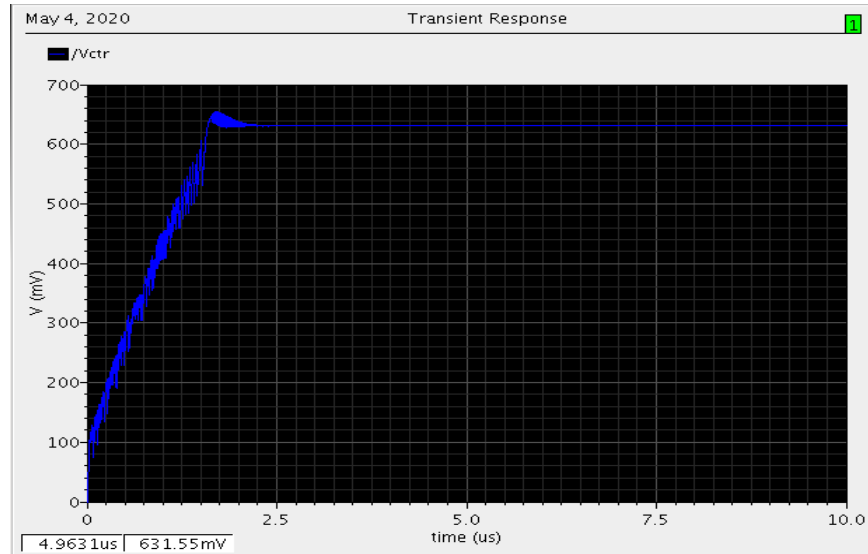
$R_z = 10.9\text{ Kohm}$ ,  $C_z = 20.6\text{ pf}$ ,  $C_p = 0.662\text{ pf}$ .

## Config view and hierarchy editor using Verilog-A models only:



## The PLL simulation using config view and hierarchy editor:

### VCO control voltage (behavioral simulation):

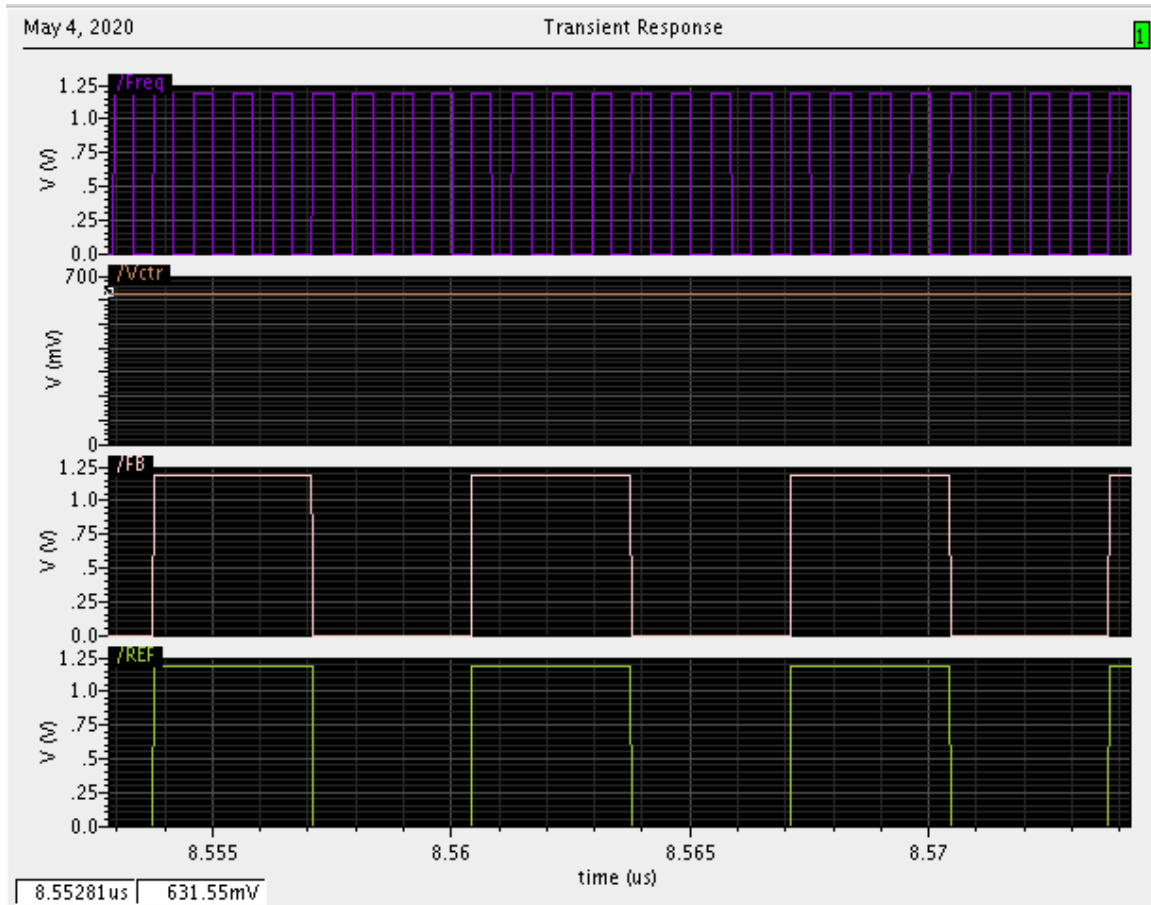


At  $V_{ctr} = 631.55 \text{ mV}$ .

$\text{Freq(out)} = K_{vco} * V_{ctr} = 1.2 \text{ MHz}$ .

The PLL simulation using config view and hierarchy editor:

PLL in locked state of for VCO (behavioral simulation):



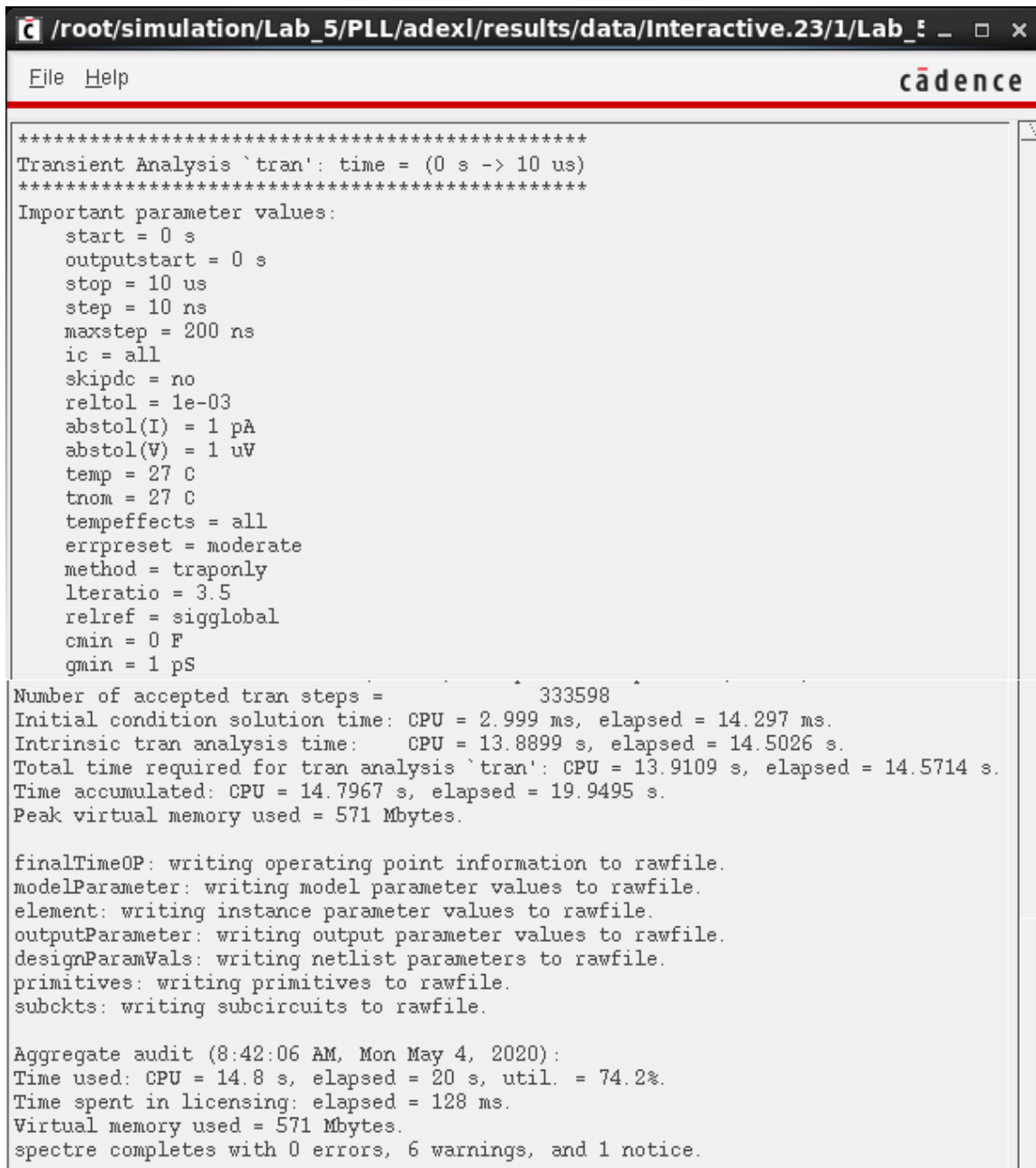
The reference and the output of the divider (Feedback) are the same.

Control volt ( $V_{ctr}$ ) of VCO is 631.55mV.

$Freq(out) = K_{vco} * V_{ctr} = 1.90013299E9 * 631.55E-3 = 1.2 \text{ GHz}$



The log file of the PLL simulation using config view and hierarchy editor:



The screenshot shows a Cadence Spectre log window with the title bar "/root/simulation/Lab\_5/PLL/adexl/results/data/Interactive.23/1/Lab\_5\_ \_ □ ×". The window has a menu bar with "File" and "Help", and the Cadence logo in the top right corner. The log content is as follows:

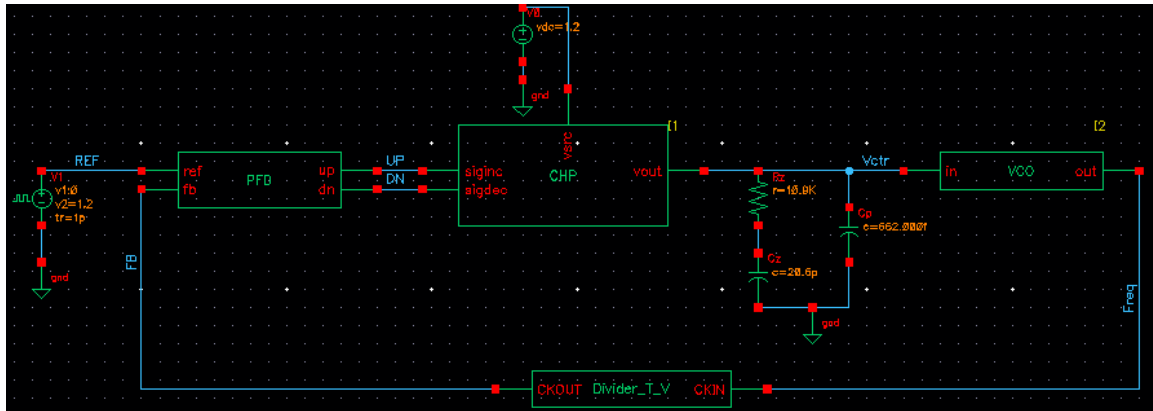
```
*****
Transient Analysis `tran': time = (0 s -> 10 us)
*****
Important parameter values:
  start = 0 s
  outputstart = 0 s
  stop = 10 us
  step = 10 ns
  maxstep = 200 ns
  ic = all
  skipdc = no
  reltol = 1e-03
  abstol(I) = 1 pA
  abstol(V) = 1 uV
  temp = 27 C
  tnom = 27 C
  tempeffects = all
  errpreset = moderate
  method = traponly
  lteratio = 3.5
  relref = sigglobal
  cmin = 0 F
  gmin = 1 pS

Number of accepted tran steps = 333598
Initial condition solution time: CPU = 2.999 ms, elapsed = 14.297 ms.
Intrinsic tran analysis time: CPU = 13.8899 s, elapsed = 14.5026 s.
Total time required for tran analysis `tran': CPU = 13.9109 s, elapsed = 14.5714 s.
Time accumulated: CPU = 14.7967 s, elapsed = 19.9495 s.
Peak virtual memory used = 571 Mbytes.

finalTimeOP: writing operating point information to rawfile.
modelParameter: writing model parameter values to rawfile.
element: writing instance parameter values to rawfile.
outputParameter: writing output parameter values to rawfile.
designParamVals: writing netlist parameters to rawfile.
primitives: writing primitives to rawfile.
subckts: writing subcircuits to rawfile.

Aggregate audit (8:42:06 AM, Mon May 4, 2020):
Time used: CPU = 14.8 s, elapsed = 20 s, util. = 74.2%.
Time spent in licensing: elapsed = 128 ms.
Virtual memory used = 571 Mbytes.
spectre completes with 0 errors, 6 warnings, and 1 notice.
```

## The PLL simulation using config view and hierarchy editor:

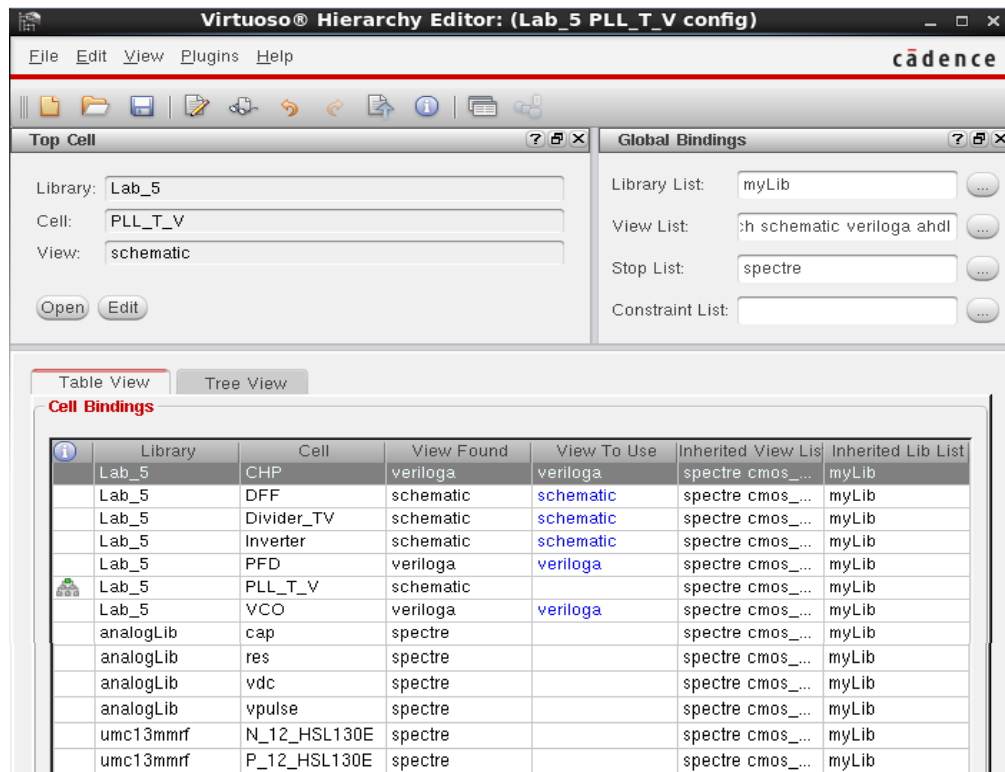


$V_1 = 0\text{ V}$ ,  $V_2 = 1.2\text{ V}$ ,  $\text{Freq} = 150\text{ MHz}$ ,  $\text{trise} = \text{tfall} = 1\text{ ps}$ .

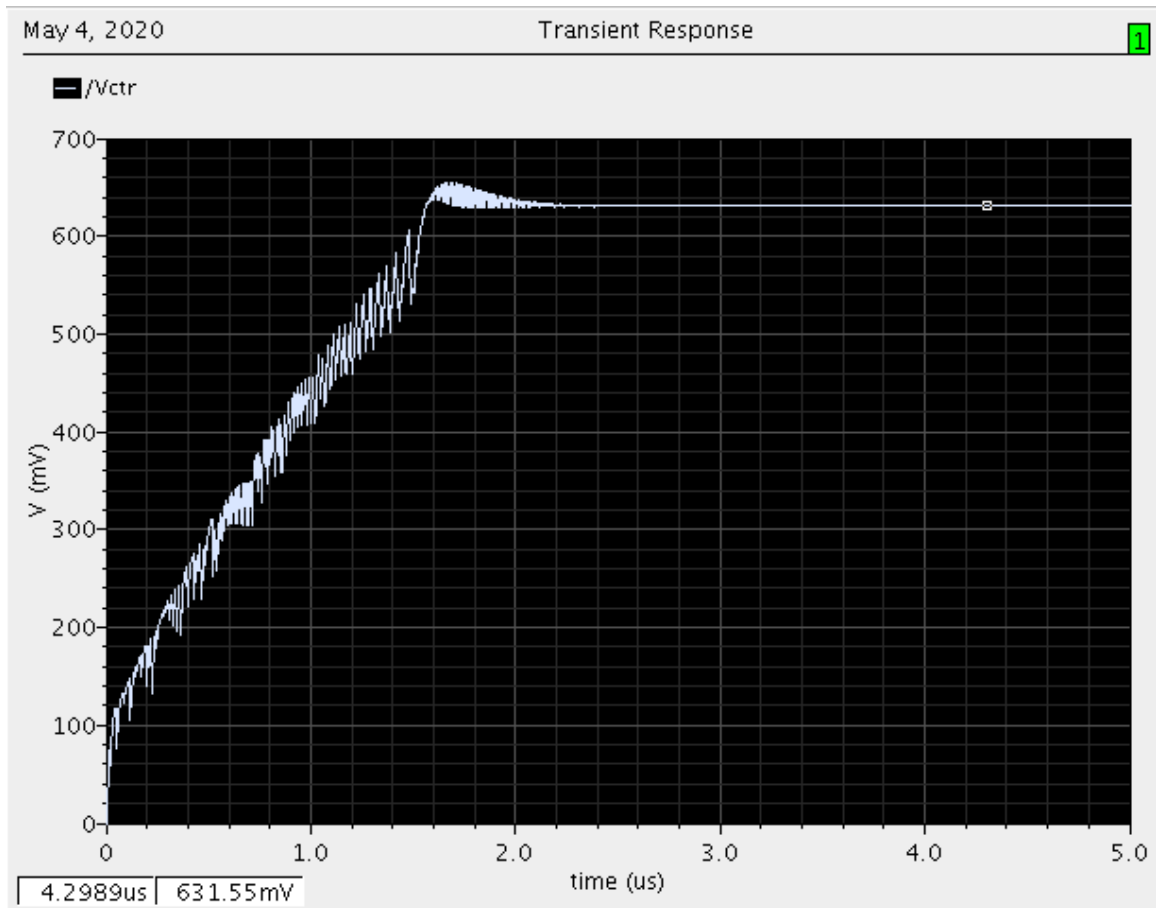
$R_z = 10.9\text{ Kohm}$ ,  $C_z = 20.6\text{ pf}$ ,  $C_p = 0.662\text{ pf}$ .

$V_{DD} = 1.2\text{ V}$

## Config view and hierarchy editor using the divider with the schematic:



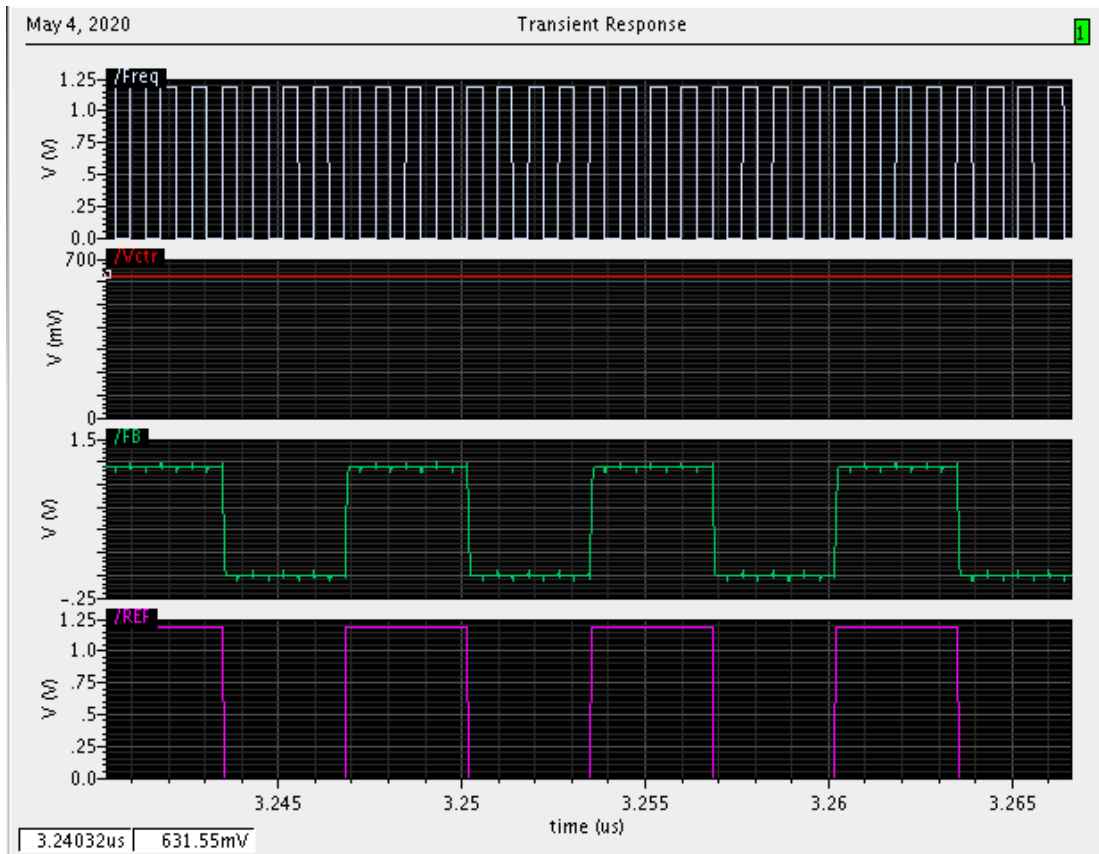
## VCO control voltage for full transistor level simulation:



At  $V_{ctr} = 631.55$  mV.

$Freq (out) = K_{vco} * V_{ctr} = 1.2$  MHz.

## PLL in locked state for full transistor level simulation:

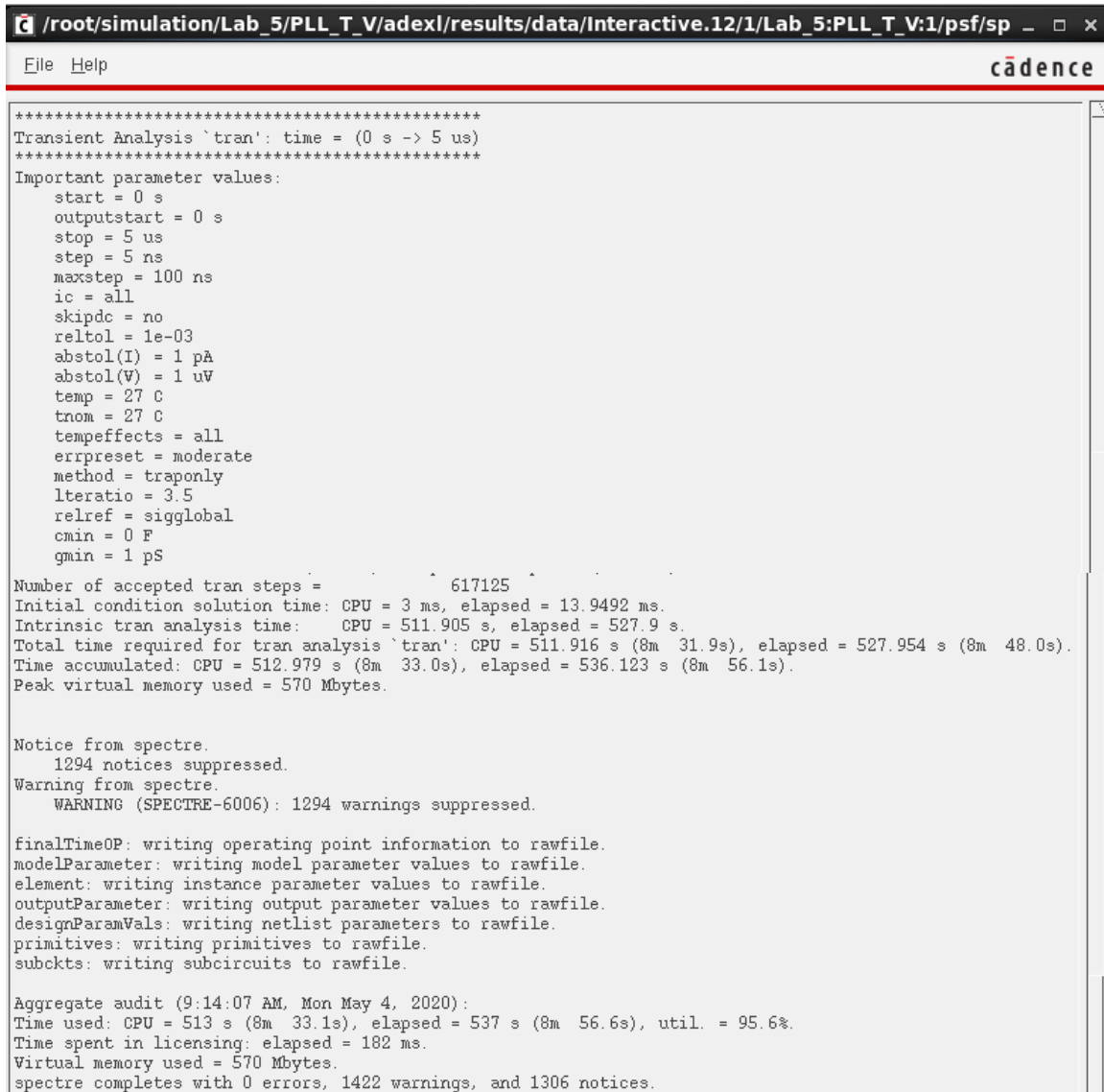


The reference and the output of the divider (Feedback) are almost the same

Control volt (Vctr) of VCO is 631.55mV.

$\text{Freq (out)} = K_{vco} * V_{ctr} = 1.90013299\text{E}9 * 631.55\text{E}-3 = 1.2 \text{ GHz}.$

## The log file of the PLL simulation using config view and hierarchy editor:



The screenshot shows a window titled `/root/simulation/Lab_5/PLL_T_V/adexl/results/data/Interactive.12/1/Lab_5:PLL_T_V:1/psf/sp` with a menu bar containing `File` and `Help`, and a `cadence` logo in the top right corner. The main area displays the following log text:

```
*****
Transient Analysis `tran': time = (0 s -> 5 us)
*****
Important parameter values:
  start = 0 s
  outputstart = 0 s
  stop = 5 us
  step = 5 ns
  maxstep = 100 ns
  ic = all
  skipdc = no
  reltol = 1e-03
  abstol(I) = 1 pA
  abstol(V) = 1 uV
  temp = 27 C
  tnom = 27 C
  tempeffects = all
  errpreset = moderate
  method = trapezoidal
  lteratio = 3.5
  relref = sigglobal
  cmin = 0 F
  gmin = 1 pS

Number of accepted tran steps = 617125
Initial condition solution time: CPU = 3 ms, elapsed = 13.9492 ms.
Intrinsic tran analysis time: CPU = 511.905 s, elapsed = 527.9 s.
Total time required for tran analysis `tran': CPU = 511.916 s (8m 31.9s), elapsed = 527.954 s (8m 48.0s).
Time accumulated: CPU = 512.979 s (8m 33.0s), elapsed = 536.123 s (8m 56.1s).
Peak virtual memory used = 570 Mbytes.

Notice from spectre.
  1294 notices suppressed.
Warning from spectre.
  WARNING (SPECTRE-6006): 1294 warnings suppressed.

finalTimeOP: writing operating point information to rawfile.
modelParameter: writing model parameter values to rawfile.
element: writing instance parameter values to rawfile.
outputParameter: writing output parameter values to rawfile.
designParamVals: writing netlist parameters to rawfile.
primitives: writing primitives to rawfile.
subckts: writing subcircuits to rawfile.

Aggregate audit (9:14:07 AM, Mon May 4, 2020):
Time used: CPU = 513 s (8m 33.1s), elapsed = 537 s (8m 56.6s), util. = 95.6%.
Time spent in licensing: elapsed = 182 ms.
Virtual memory used = 570 Mbytes.
spectre completes with 0 errors, 1422 warnings, and 1306 notices.
```

## Comparison:

The simulation time of PLL using Verilog-A models only	The simulation time of PLL using Verilog-A models and only the divider with transistor level
Time used: CPU = 14.8 s, elapsed =20 s, util = 74.2%.	Time used: CPU = 513 s, elapsed =537 s, util = 95.6%.
Time spent in licensing: elapsed =128 ms.	Time spent in licensing: elapsed =182 ms.

## Comment:

It's clear that on simulating all blocks with Verilog-A models is much faster than using transistor level blocks. When we just replaced the divider with its transistor-level model, the simulation became much slower.

## Reference:

- <http://rt2innocence.net/integrated-circuit/pfd-veriloga-model>
- ahdl library
- <https://designers-guide.org/verilog-ams/functional-blocks/vco/vco.va>
- <https://designers-guide.org/verilog-ams/functional-blocks/freq-divider/freq-divider.va>