

Topic of the project:

Project name: "ASIC design with open source tools"

Developing of an low resolution ADC. As example an 4-Bit ADC with conversion of the thermometer code into binary code.

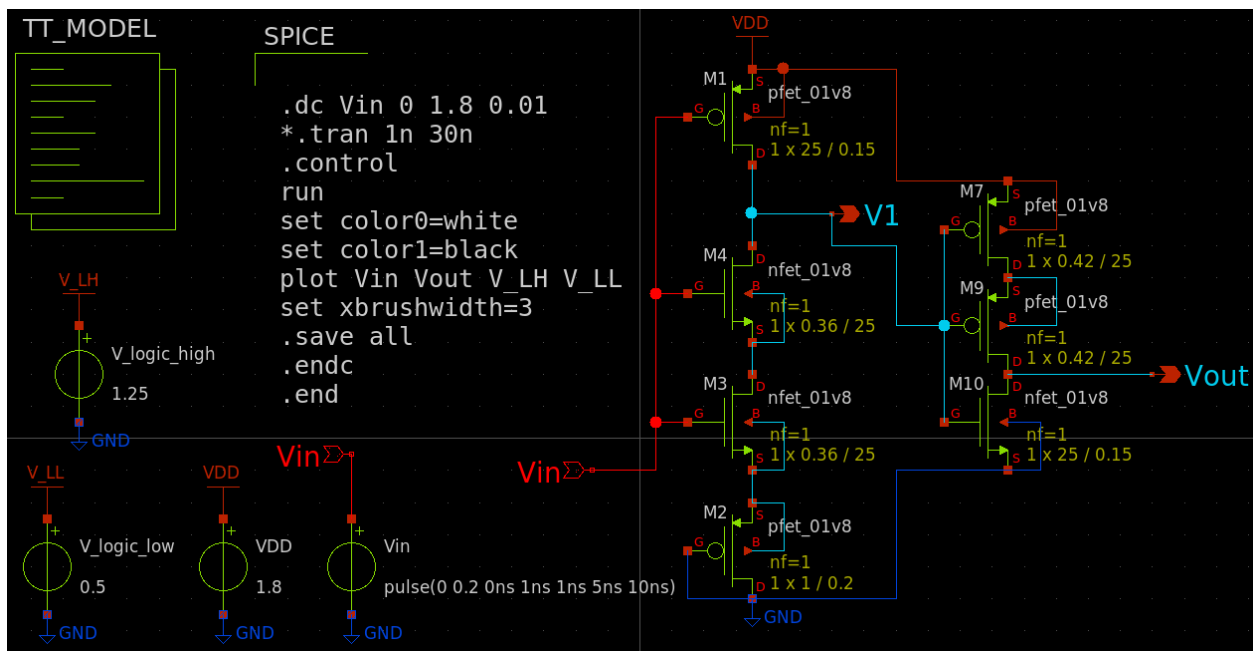
Steps:

A small CMOS inverter with xschem and the sky130a technology (PDK) and simulate the circuit with ngspice:

Binary Logic level assumption

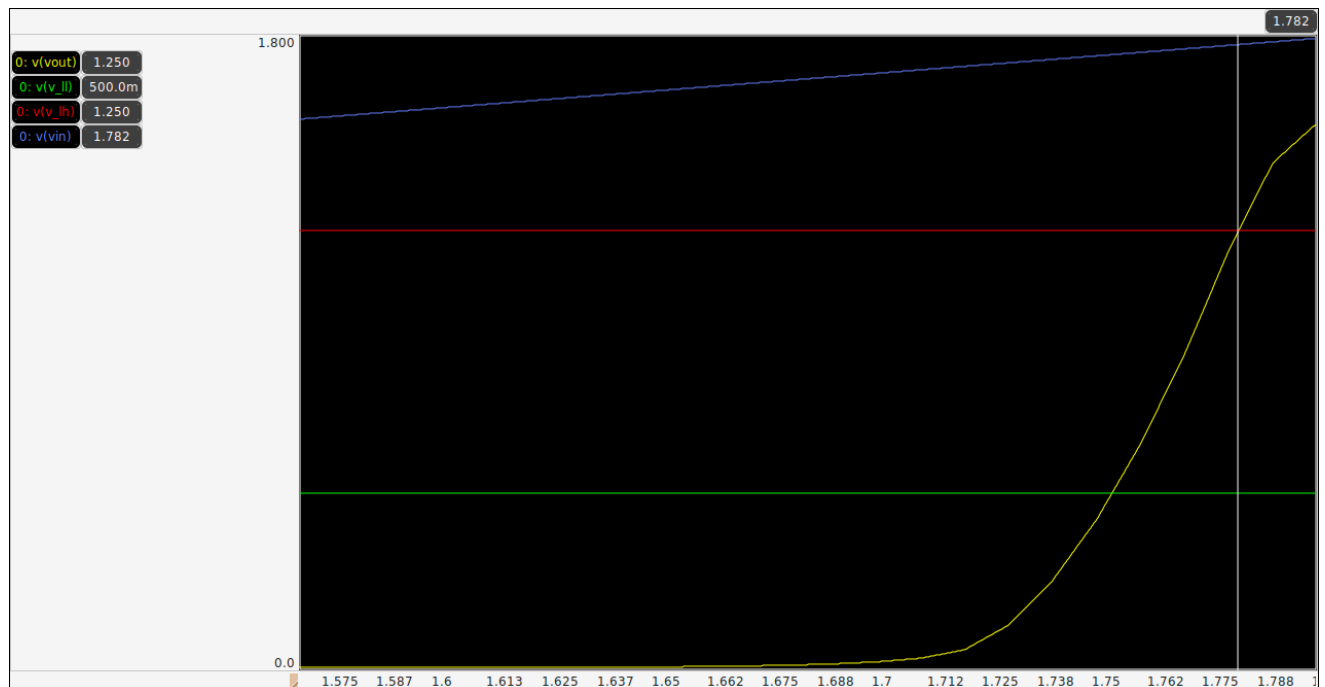
https://en.wikipedia.org/wiki/Logic_level

Technology	L Voltage	H Voltage	note
CMOS	0 to 30% of Vdd	70% of Vdd to Vdd	Vdd = Supply voltage
	≈0.5 Volt	≈1.25 Volts	Vdd = 1.8 Volts



The above figure is the schematic for 15th threshold point of the analogue part of the ADC.

The plot is as follows:



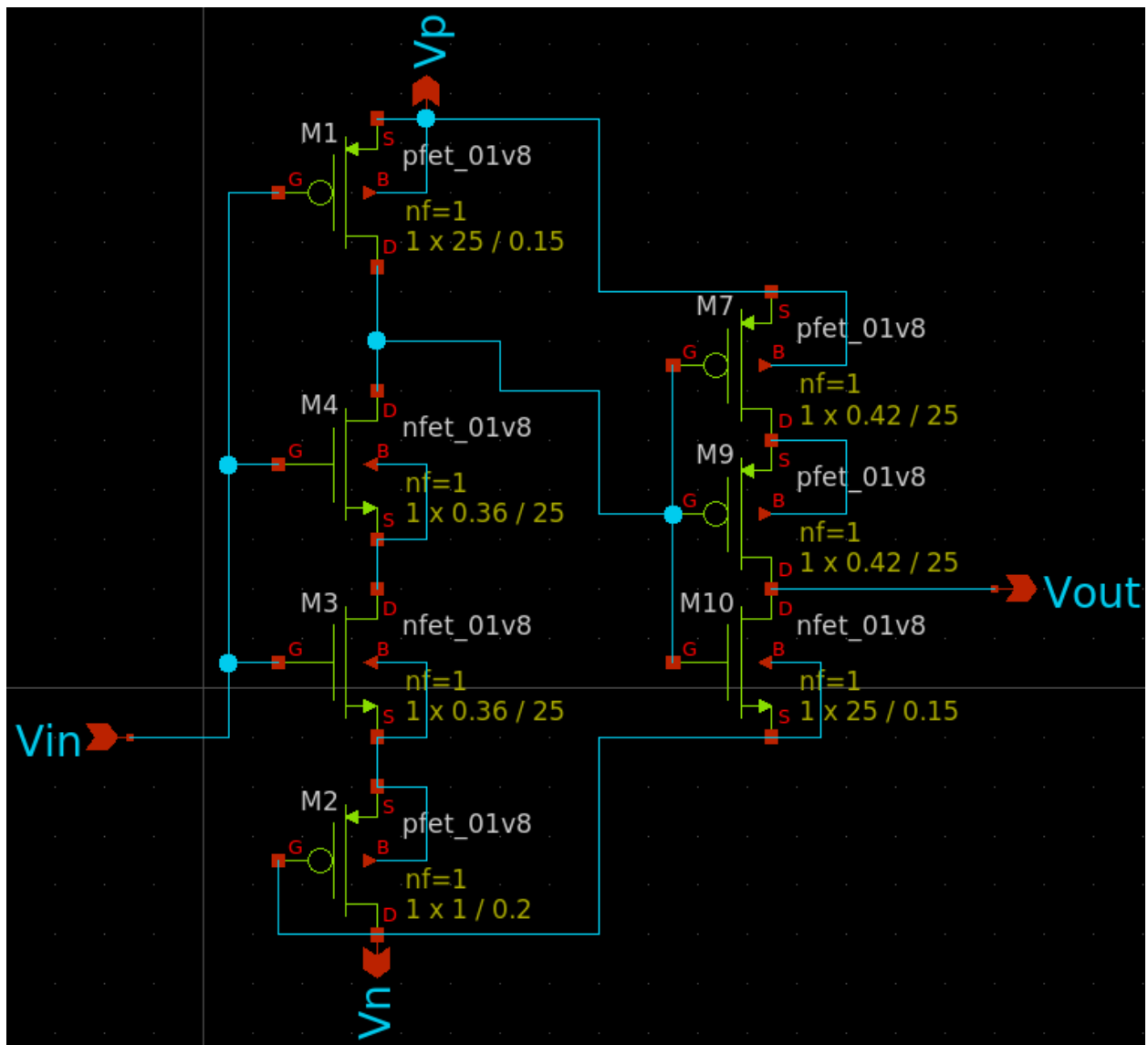
The inverter will switch from logic-low to logic-high when input voltage exceeds 1.782 volts. So, for this inverter the threshold is 1.782 volts.

For a 4-bit ADC we need 16 voltage level including zero (0~1.782 volts). The quantization should be $1.782/15=0.1188$ volts.

Vdd = 1.8 V

Th. 15: 1.782

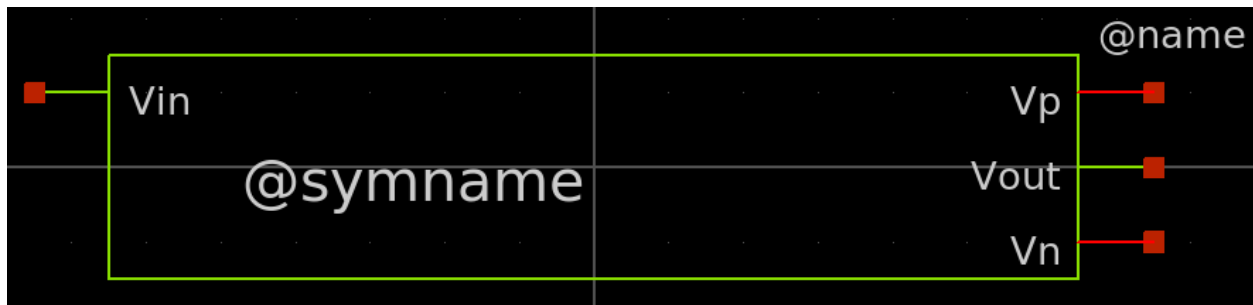
For calculation the schematic and the plot already mentioned earlier. Remove all source to make symbol from schematic. As the symbol will have to be connected to Vdd and GND, those pins are replaced with Vp and Vn accordingly.



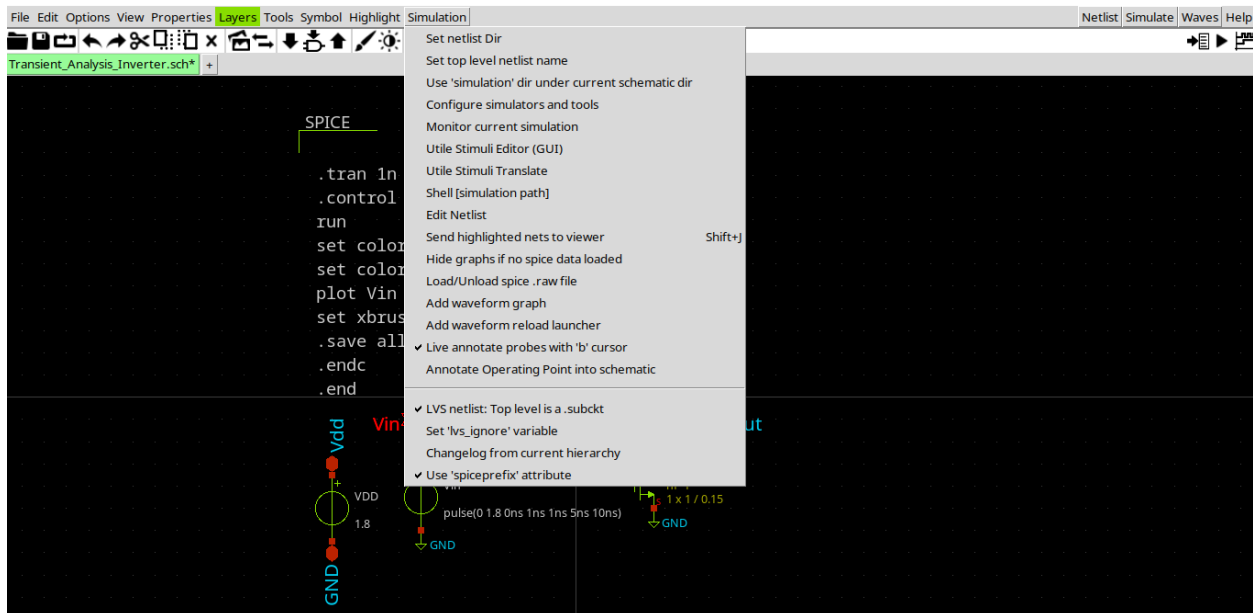
File Edit Options View Properties **Layers** Tools Symbol Highlight Simulation

Inverter-th15_sym.sch Inverter-th15_sym.sch

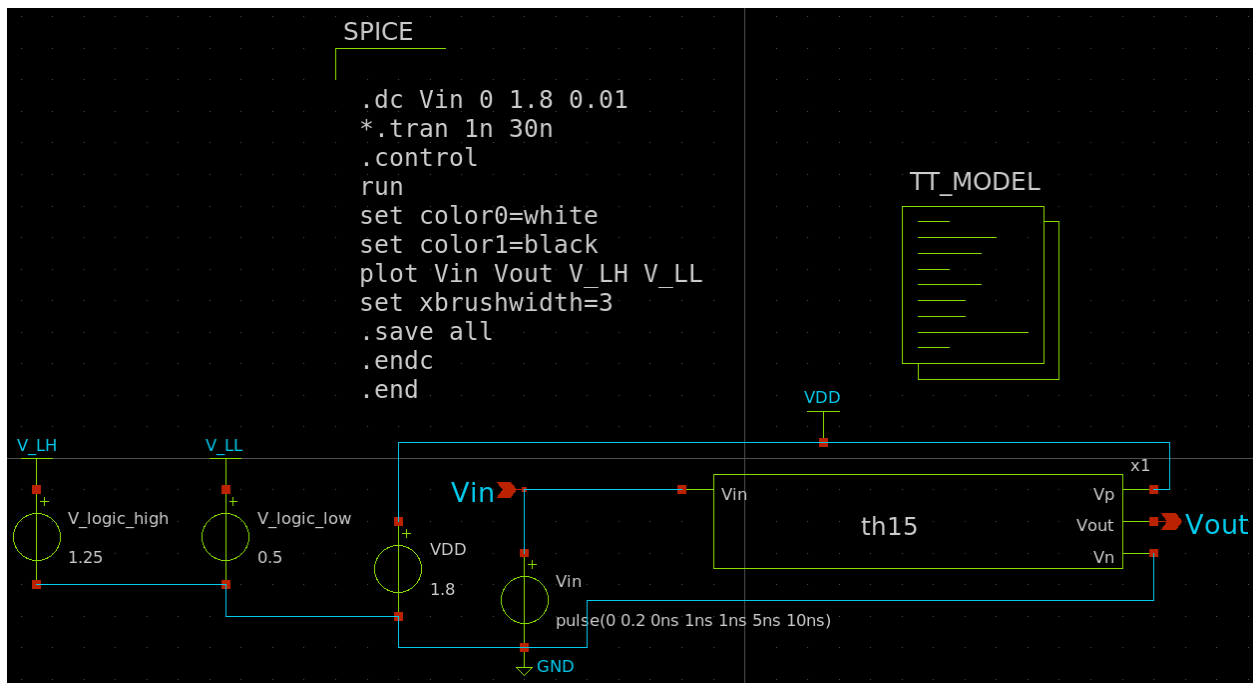
- Show symbols Alt+B
- Show instance Bounding boxes for subcircuit symbols Alt+B
- Show instance Bounding boxes for all symbols Alt+B
- Show net names on symbol pins/floater
- Set symbol width
- Make symbol from schematic A
- Make schematic from symbol Ctrl+L
- Make schematic and symbol from selected components Ctrl+Shift+H
- Attach net labels to component instance Shift+H
- Create symbol pins from selected schematic pins Alt+H
- Place symbol pin Alt+P
- Print list of highlight nets J
- Print list of highlight nets, with buses expanded Alt-Ctrl-J
- Create labels from highlight nets Alt-J
- Create labels from highlight nets with 'i' prefix Alt-Shift-J
- Create pins from highlight nets Ctrl-J
- Search all search-paths for schematic associated to symbol
- Allow duplicated instance names (refdes)



Magic=> before importing netlist to ngspice check LVS net

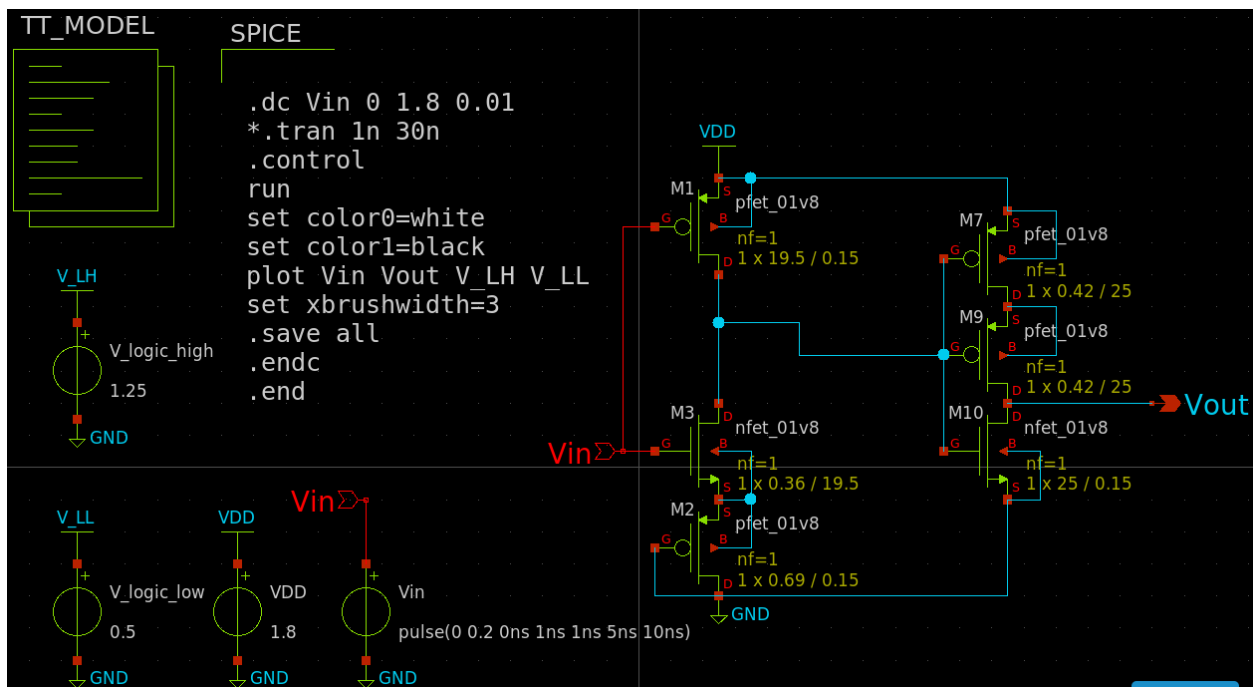


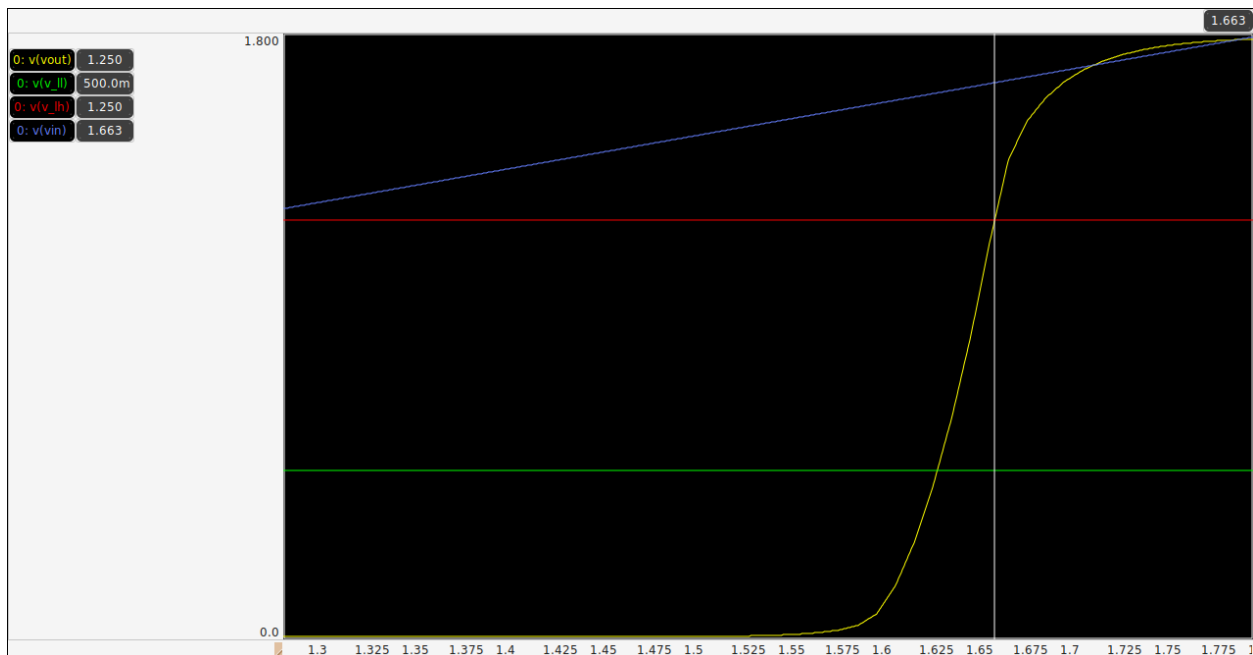
The simplified schematic is as follows:



The plot is similar to the previous figure.

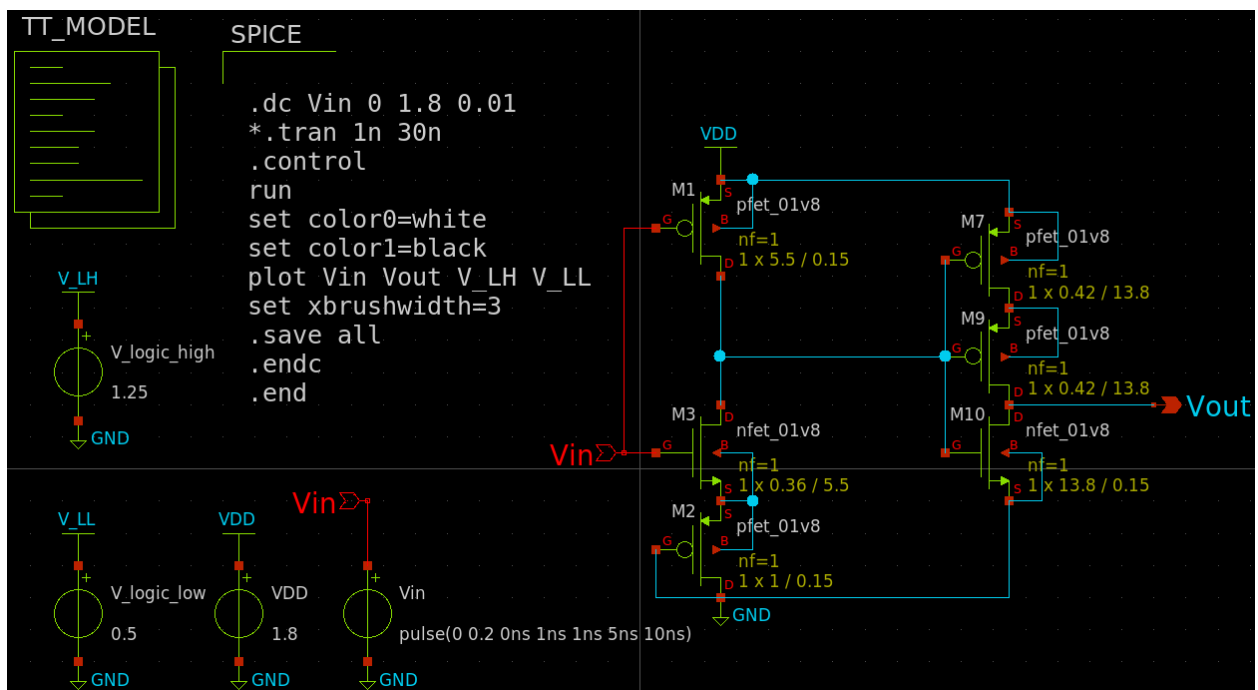
Th. 14: 1.6632

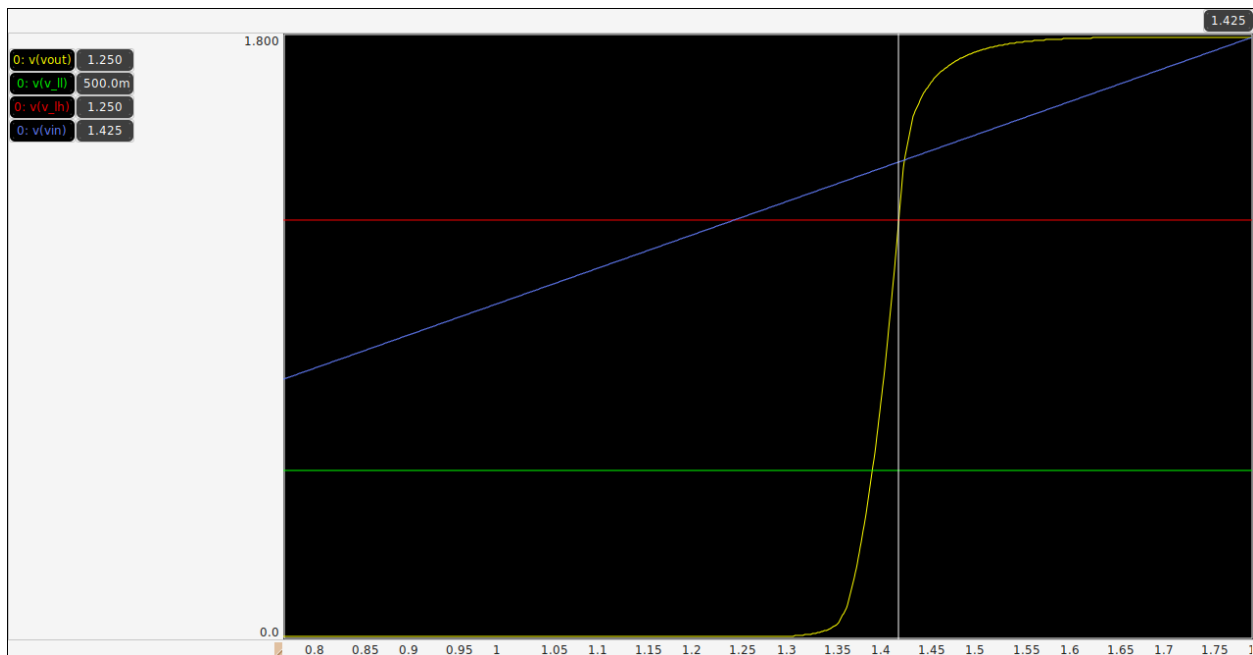




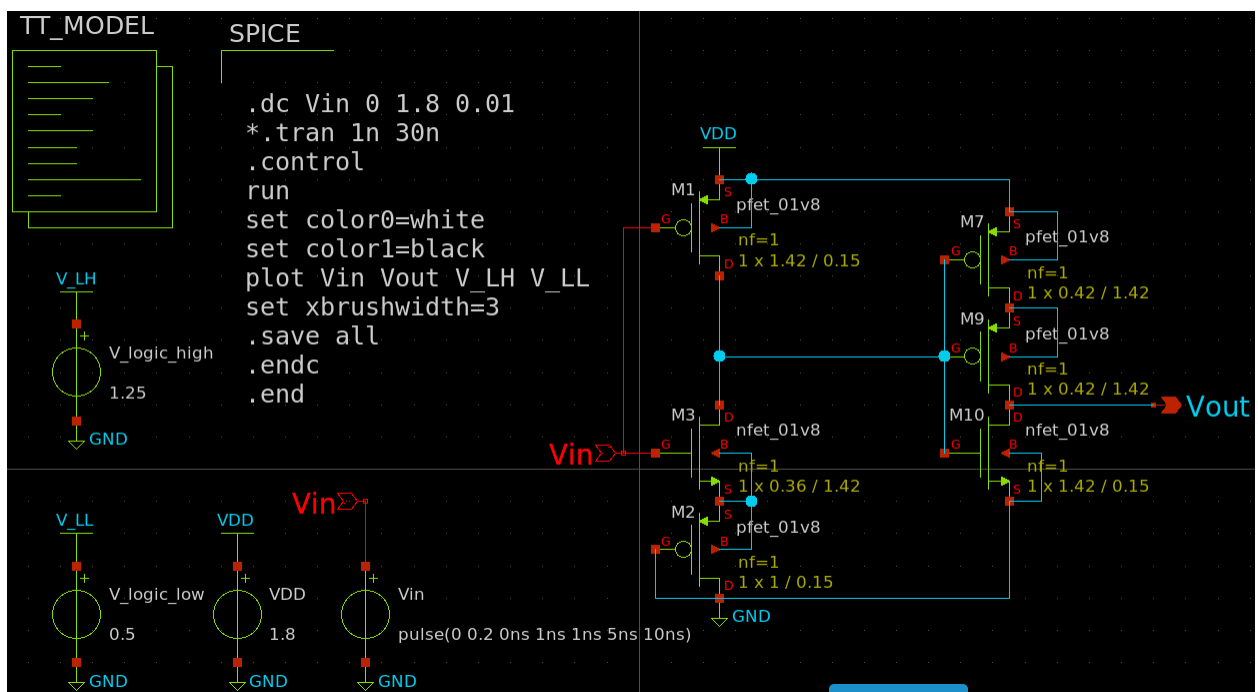
Rest of the figures are similar to the previous figures.

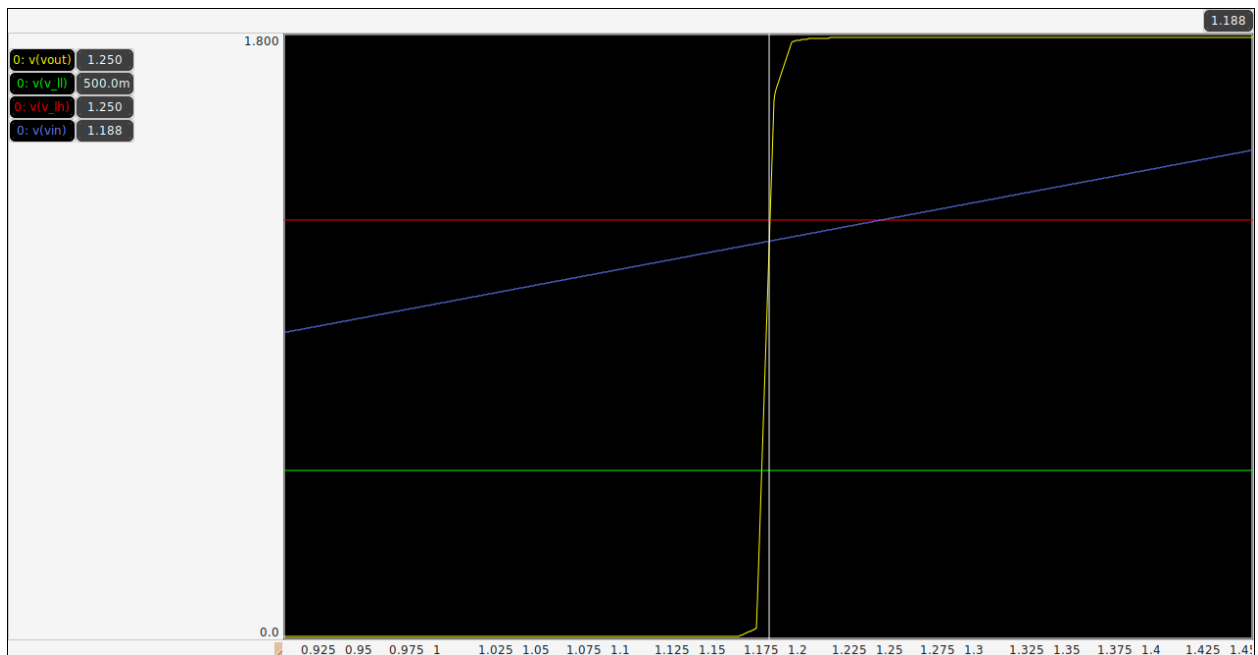
Th. 13: **1.5444:**



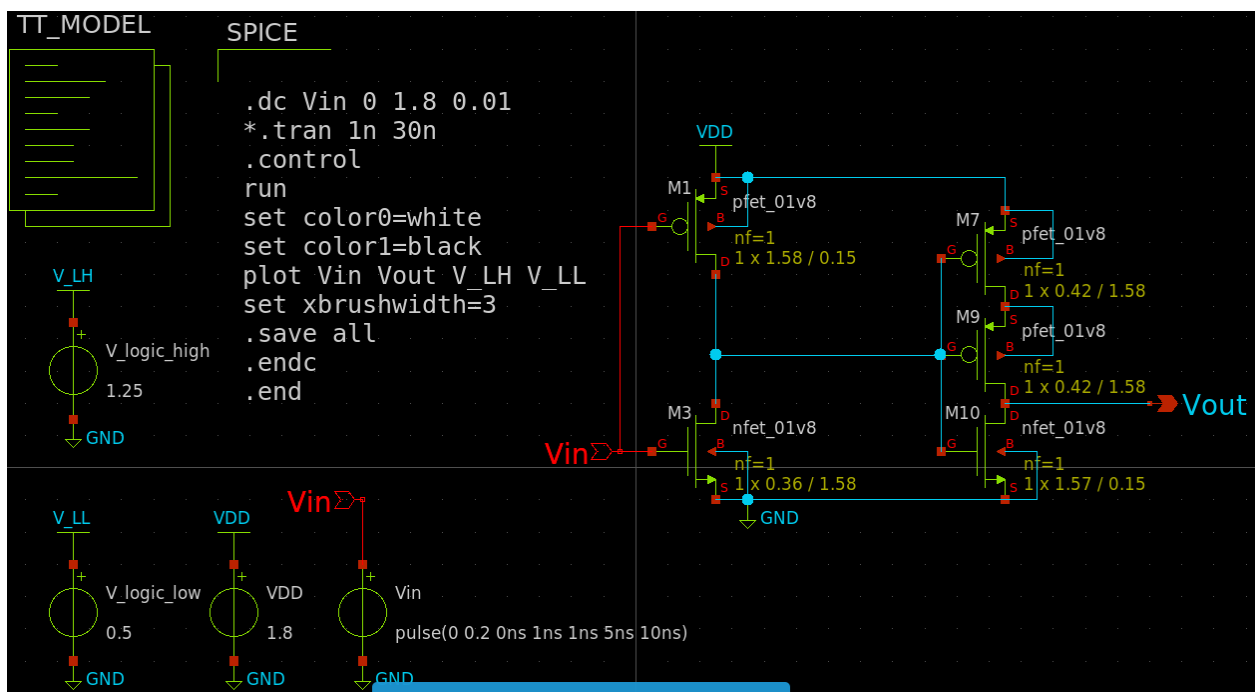


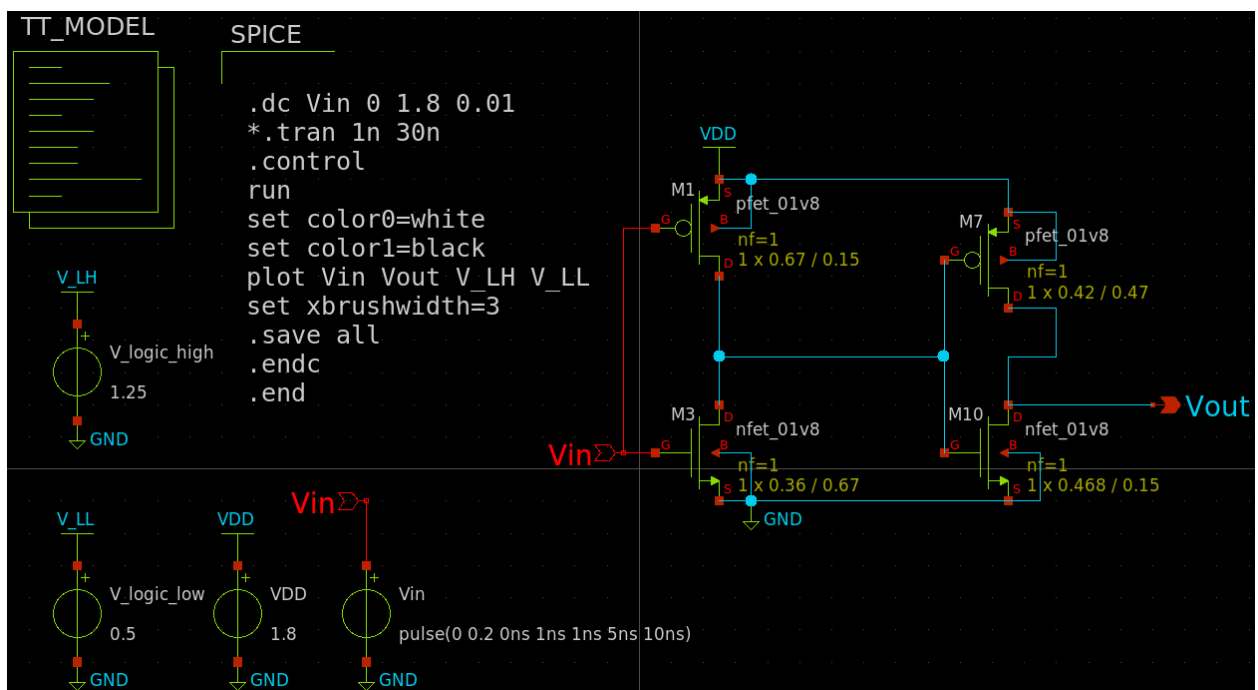
Th. 11: 1.3068

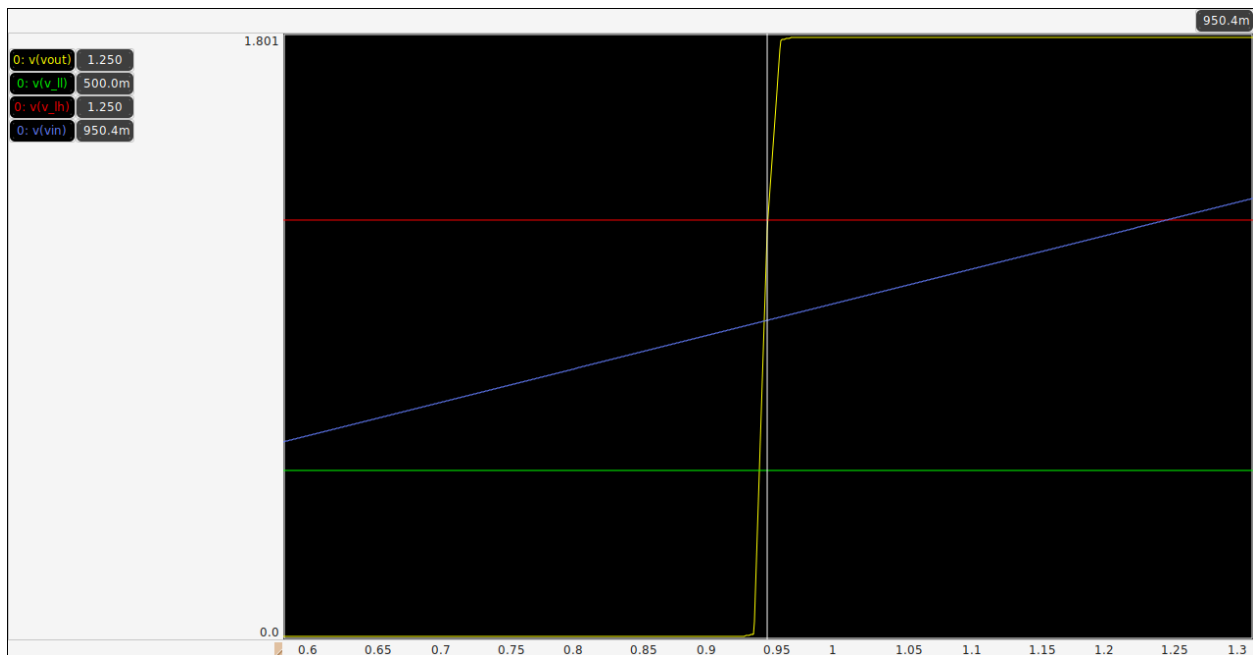




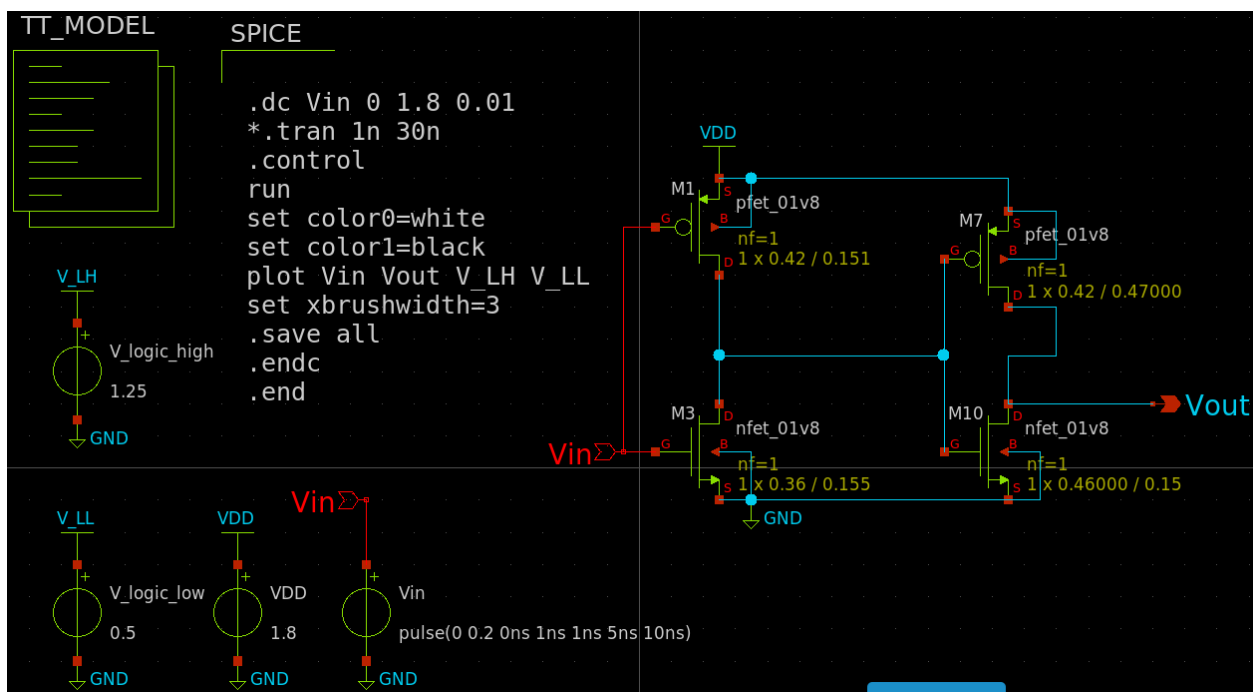
Th. 9: 1.0692

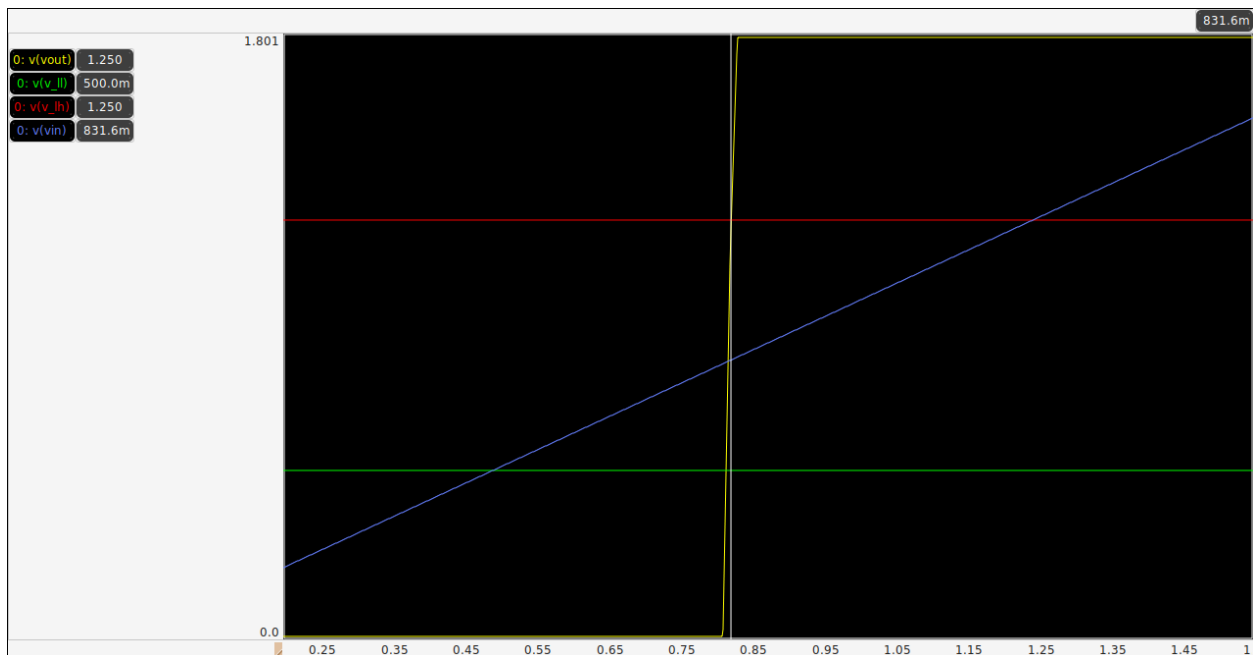




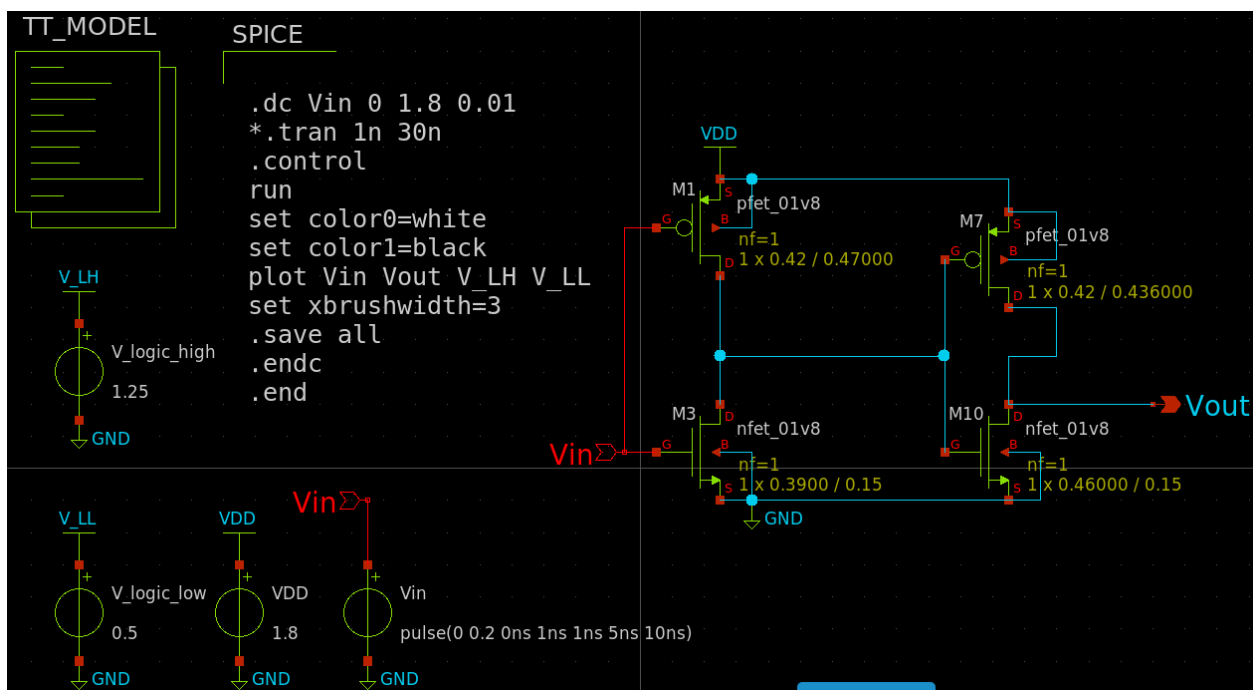


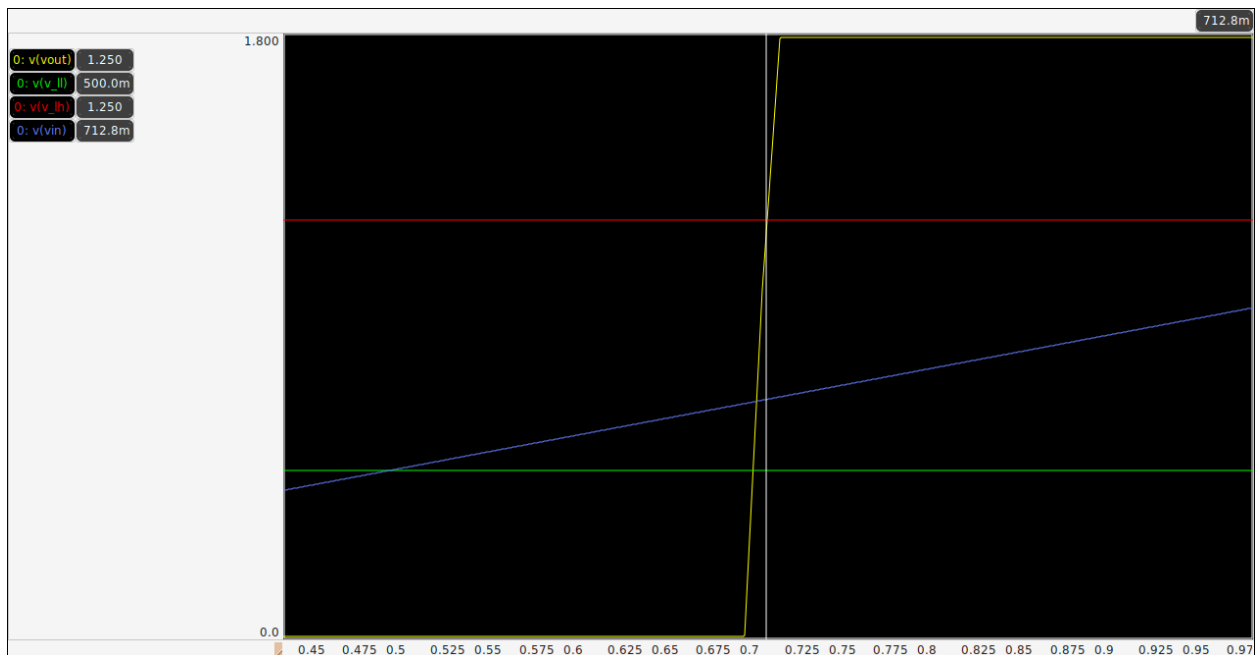
Th. 7: 0.8316



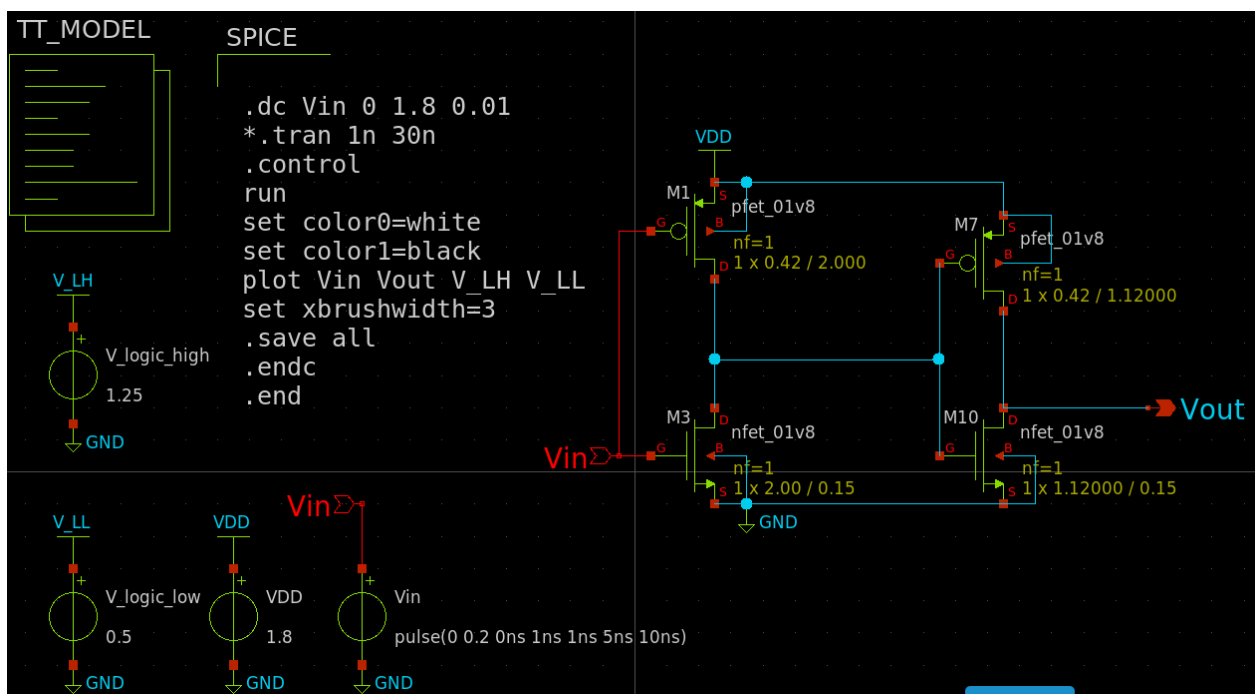


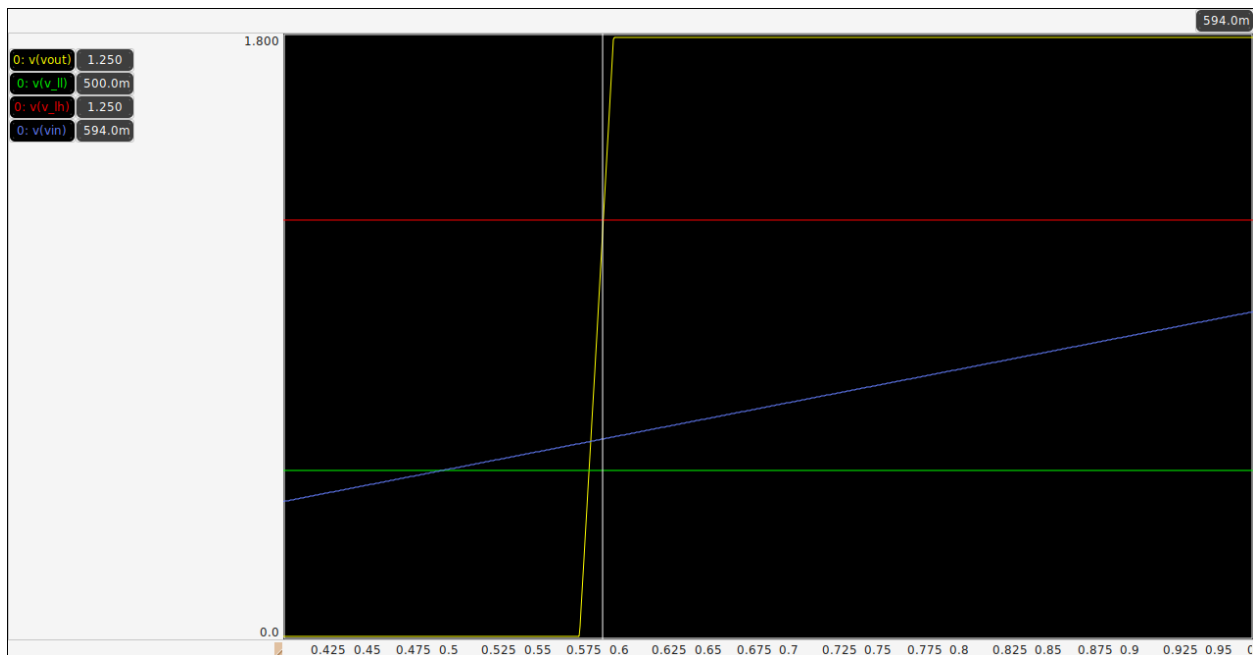
Th. 6: 0.7128



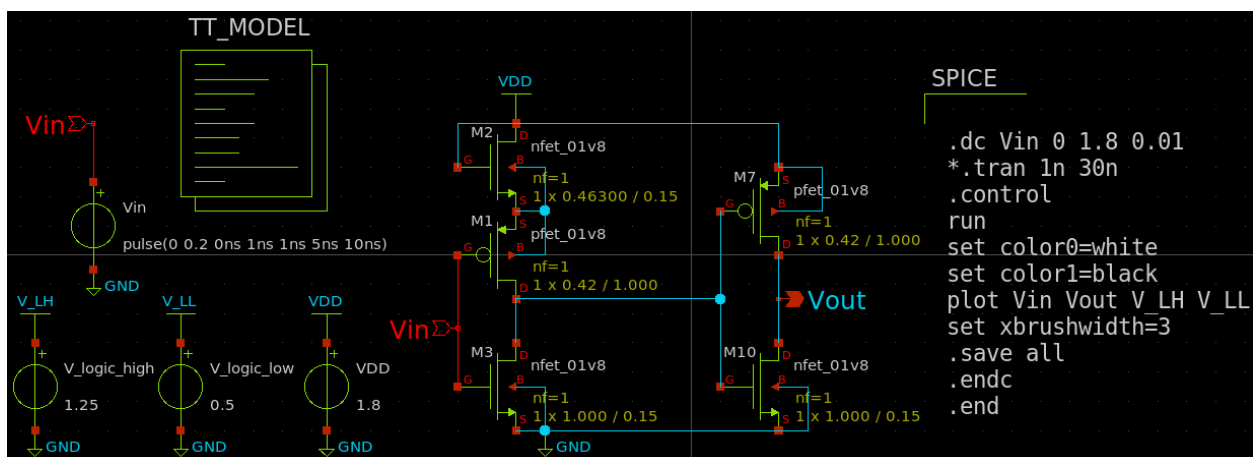


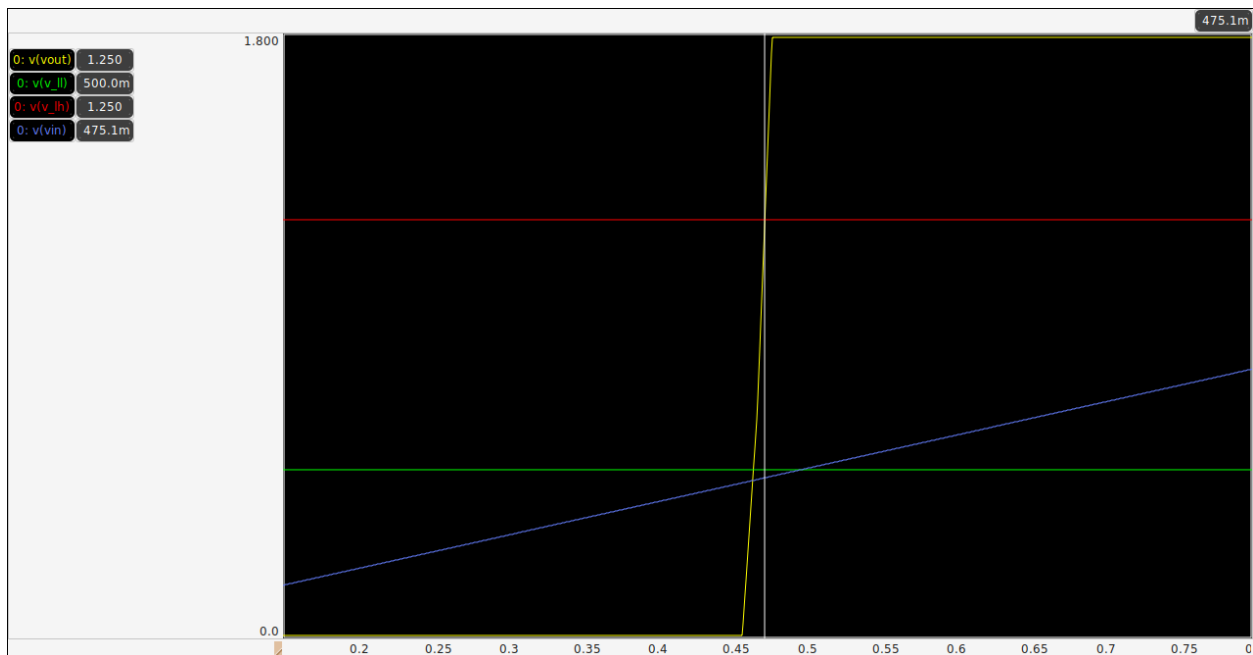
Th. 5: 0.594



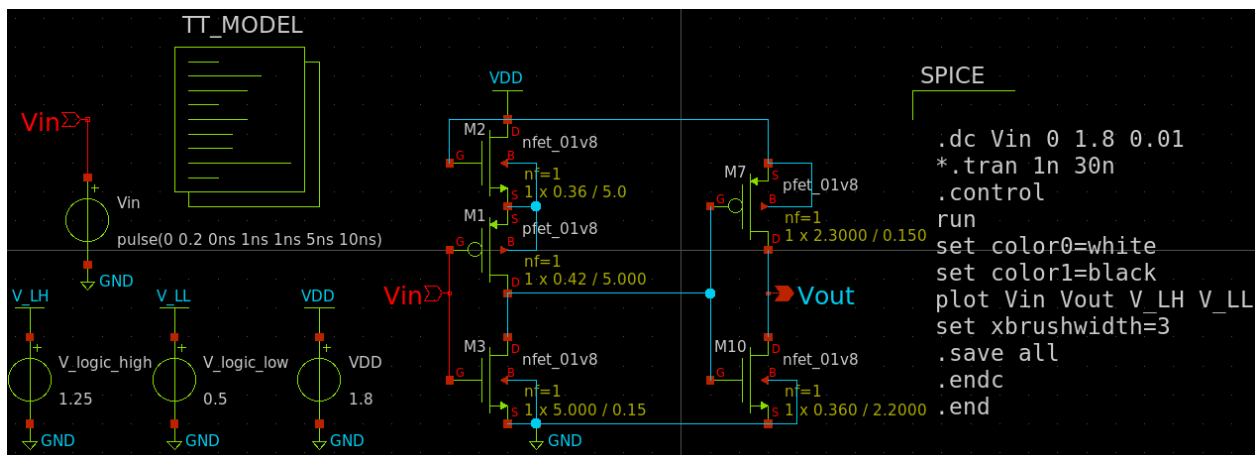


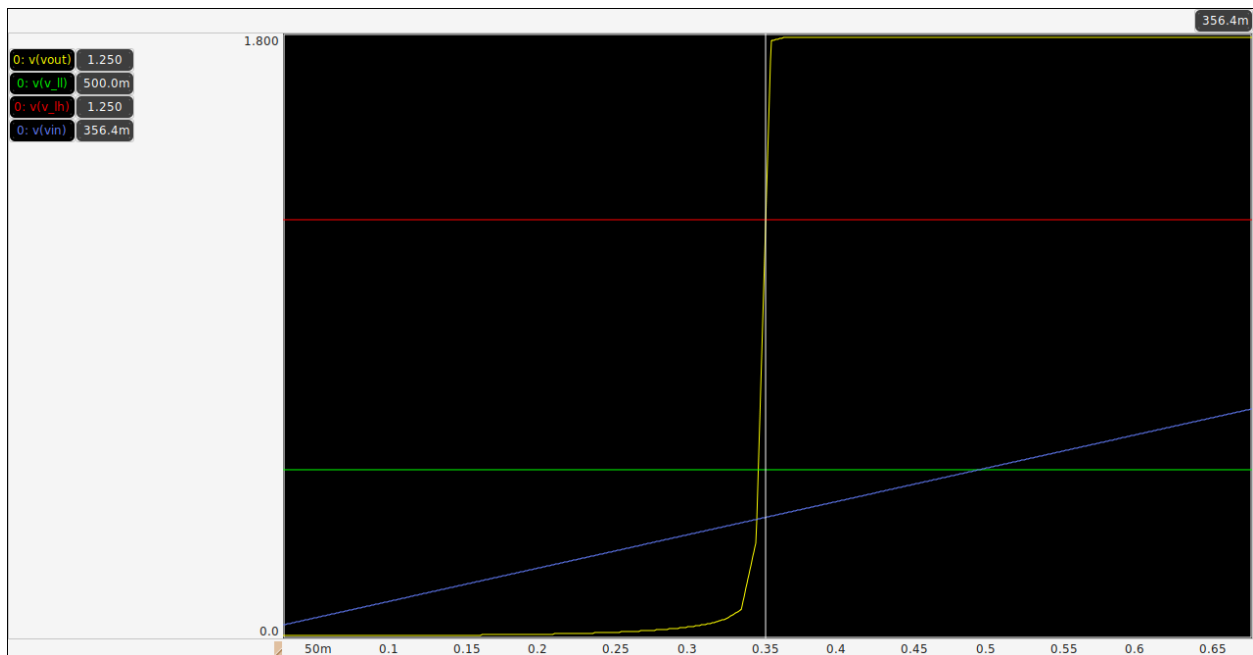
Th. 4: 0.4752



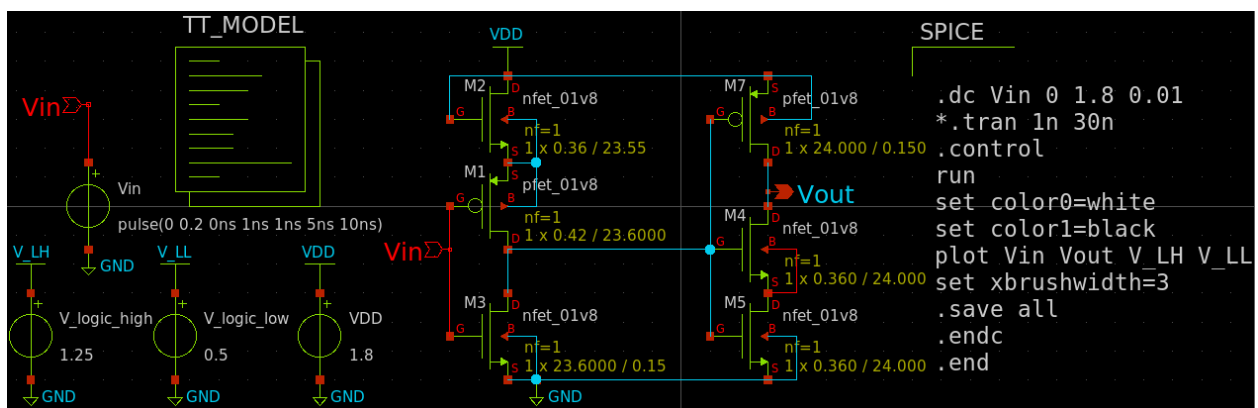


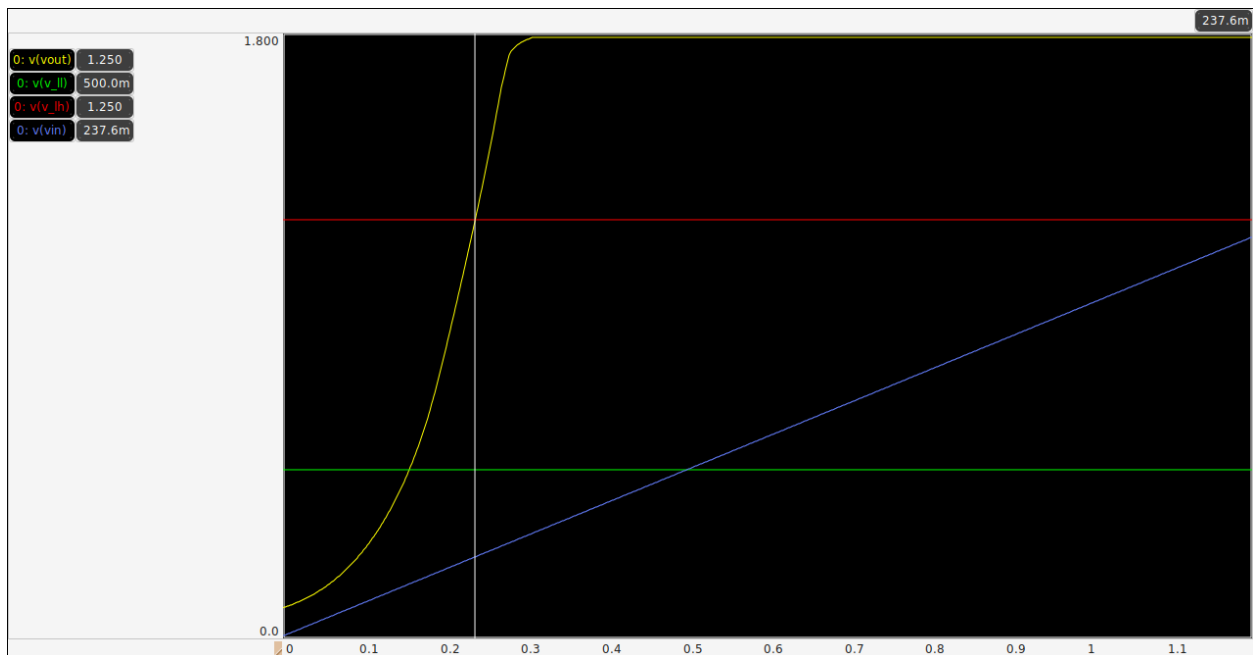
Th. 3: 0.3564



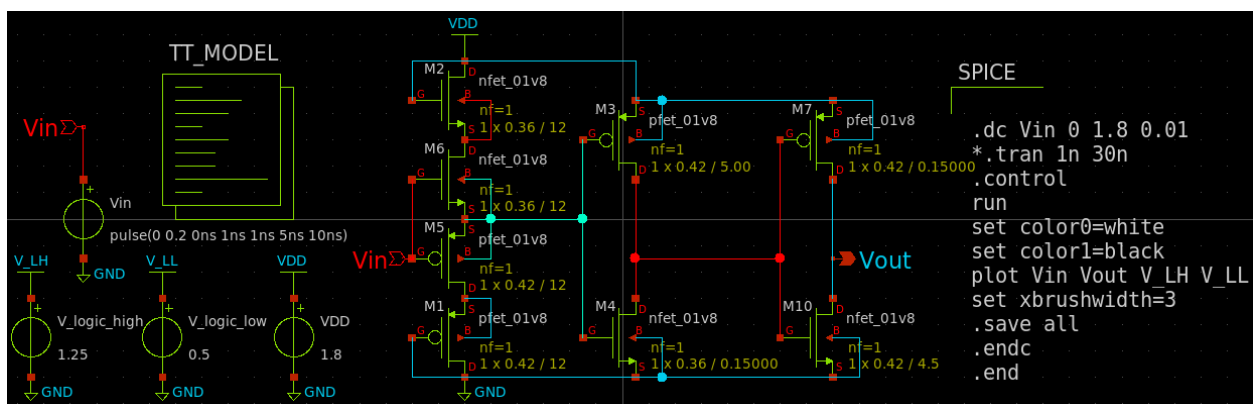


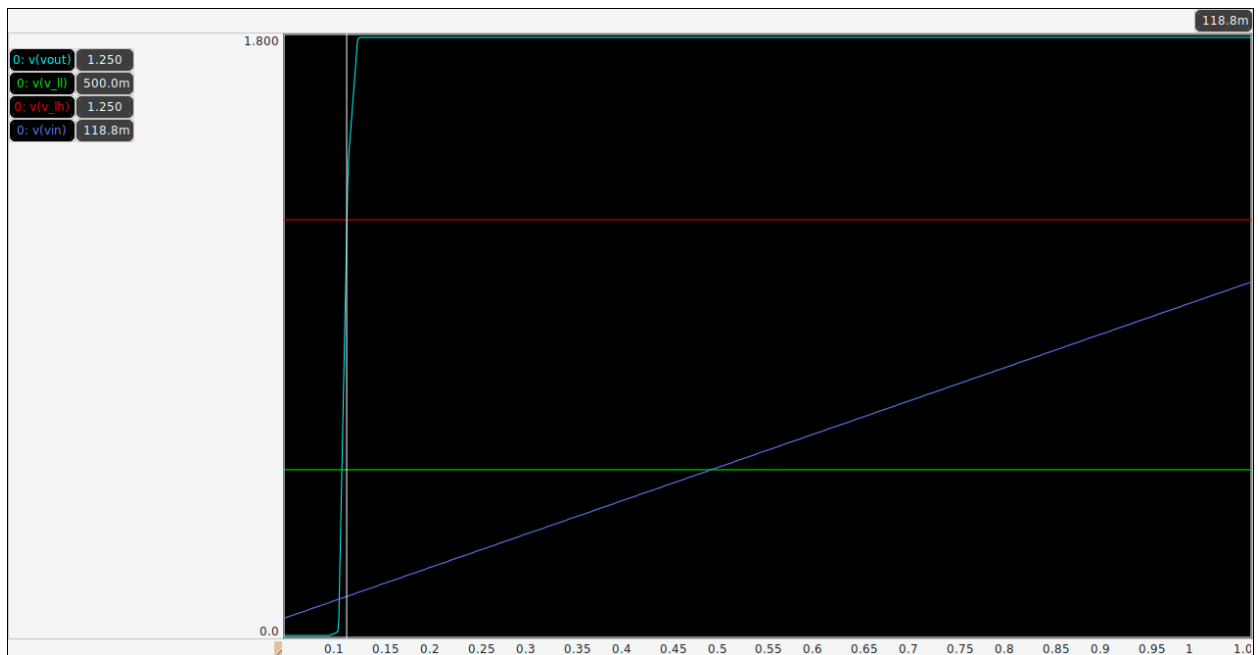
Th. 2: 0.2376



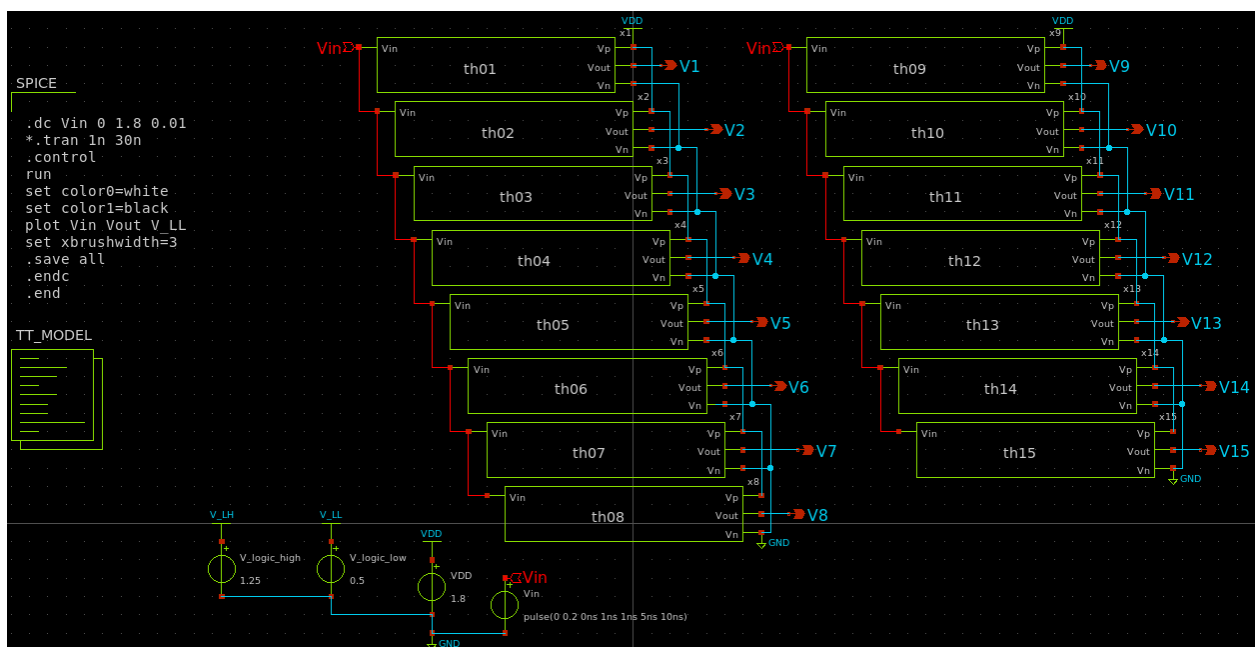


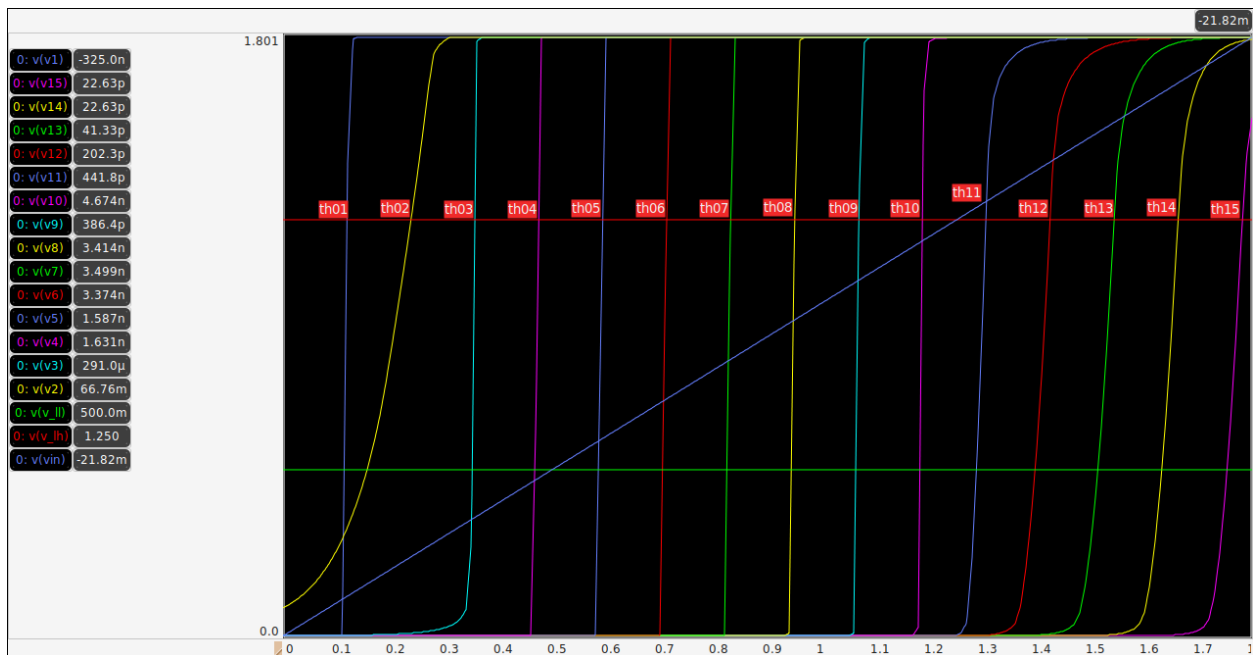
Th. 1: 0.1188





Combined Analog sch.





<http://web02.gonzaga.edu/faculty/talarico/vlsi/xschemTut.html>

<https://ngspice.sourceforge.io/ngspice-control-language-tutorial.html>

Alter subckt tutorial—

<https://sourceforge.net/p/ngspice/discussion/133842/thread/5ad086c79f>

<https://sourceforge.net/p/ngspice/discussion/133842/thread/9a75acf2/#bf90/45b3/ea48>

Pyspice ngspice interpreter

<https://pyspice.fabrice-salvaire.fr/releases/v1.3/examples/ngspice-shared/ngspice-interpreter.html>

<https://github.com/ashwith/ngspicepy>

Read spice raw output data

<https://gist.githubusercontent.com/snmishra/27dcc624b639c2626137/raw/742fe0dd59c7b2c41b71a1c8c1c2506d13affc53/rawread.py>

ngspice wrapper for python

<https://github.com/eps82/lyngspice/wiki>

write some verilog code to convert the thermometer code (from the inverter stages) into the binary code

```
module thermometer_to_binary (  
    input [15:0] ith,  
    output reg [3:0] binary  
);  
  
always @(*) begin  
    case (ith)  
        16'b0000000000000000: binary = 4'b0000; // ~0.0000  
        16'b0000000000000001: binary = 4'b0001; // ~0.1188  
        16'b0000000000000011: binary = 4'b0010; // ~0.2376  
        16'b0000000000000111: binary = 4'b0011; // ~0.3564  
        16'b0000000000001111: binary = 4'b0100; // ~0.4752  
        16'b0000000000011111: binary = 4'b0101; // ~0.5940  
        16'b0000000000111111: binary = 4'b0110; // ~0.7128  
        16'b0000000001111111: binary = 4'b0111; // ~0.8316  
        16'b0000000111111111: binary = 4'b1000; // ~0.9504  
        16'b0000001111111111: binary = 4'b1001; // ~1.0692  
        16'b0000011111111111: binary = 4'b1010; // ~1.1880  
        16'b0000111111111111: binary = 4'b1011; // ~1.3068  
        16'b0001111111111111: binary = 4'b1100; // ~1.4256  
        16'b0011111111111111: binary = 4'b1101; // ~1.5444  
        16'b0111111111111111: binary = 4'b1110; // ~1.6632  
        16'b1111111111111111: binary = 4'b1111; // ~1.7820  
        default: binary = 4'bxxxx; // Don't care  
    endcase  
end
```

endmodule

<http://www.asic-world.com/verilog/veritut.html>

<https://nandland.com/introduction-to-verilog-for-beginners-with-code-examples/>

https://www.youtube.com/playlist?list=PLfGJELQIDBN0VsXQ68_FEYyqcym8CTDN

<https://github.com/Swagatika-Meher/msvds2bitcomp>

ALIGN Tool-

https://www.youtube.com/playlist?list=PLvXKBnlvcSm30Y0zu1765oG_x-ECU8tVG

<https://learning.edx.org/course/course-v1:HarveyMuddX+ENGR85A+3T2021/home>

<https://www.vlsiuniverse.com/digital-thermometer-code-in-verilog-vhdl-flash-adc-binary-encoder/>

<https://www.classcentral.com/subject/vlsi?free=true>

<https://www.eng.biu.ac.il/temanad/digital-vlsi-design/>

https://www.youtube.com/playlist?list=PLZU5hLL_713x0_AV_rVbay0pWmED7992G

https://youtu.be/BlqLk23hE90?list=PLZU5hLL_713x0_AV_rVbay0pWmED7992G

get familiar with the yosis syntesis tool to convert the verilog code into a CMOS circuit

<https://www.mehmetburakaykenar.com/synthesis-n-bit-counter-with-open-source-yosys-synthesizer/294/>

https://github.com/spdy1895/RTL_synthesis_using_sky130

<https://github.com/lmellal/RTL-workshop-Sky130-PDK>

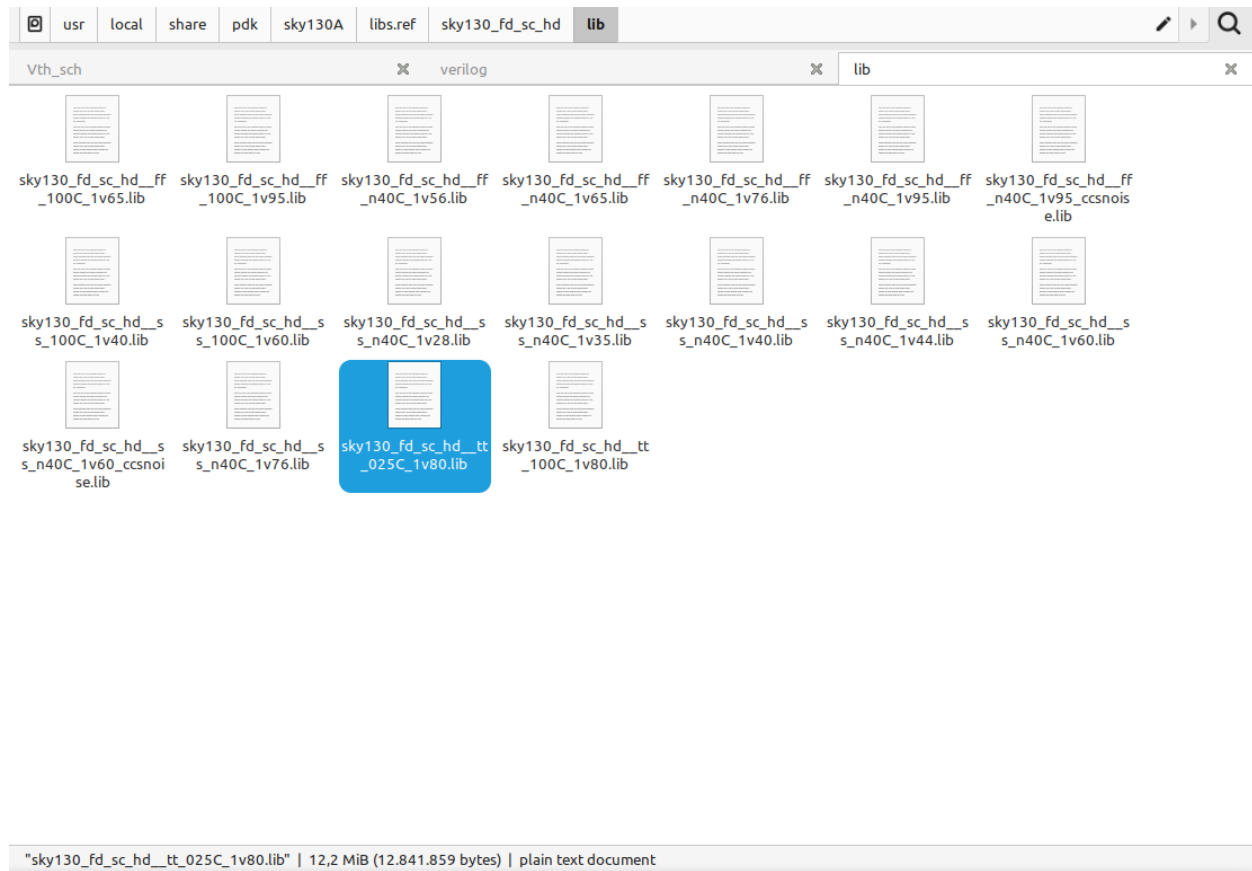
After install YoSys (see the bash file in the repository), run the following commands—

```
>> yosys          // open yosys console in terminal
```

```
>> read_liberty -lib
```

```
/usr/local/share/pdk/sky130A/libs.ref/sky130_fd_sc_hd/lib/sky130_fd_sc_hd__tt_025C_1v80.lib
```

```
// reading the suitable liberty file (see the screenshot below)
```



```
>> read_verilog therm_code.v // read the verilog file
```

```
Terminal - exotic@exotic-virtual-machine: ~/Desktop/ASIC_ADC/verilog
File Edit View Terminal Tabs Help

exotic@exotic-virtual-machine: ~/Desktop/yosys x exotic@exotic-virtual-machine: ~/Desktop/ASIC_ADC/... x

Permission to use, copy, modify, and/or distribute this software for any
purpose with or without fee is hereby granted, provided that the above
copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES
WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR
ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF
OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

-----/

Yosys 0.35+36 (git sha1 c95298225, gcc 11.4.0-1ubuntu1~22.04 -fPIC -Os)

yosys> read verilog therm_code.v
1. Executing Verilog-2005 frontend: therm_code.v
Parsing Verilog input from `therm_code.v' to AST representation.
Generating RTLIL representation for module `thermometer_to_binary'.
Successfully finished Verilog frontend.

yosys> 
```

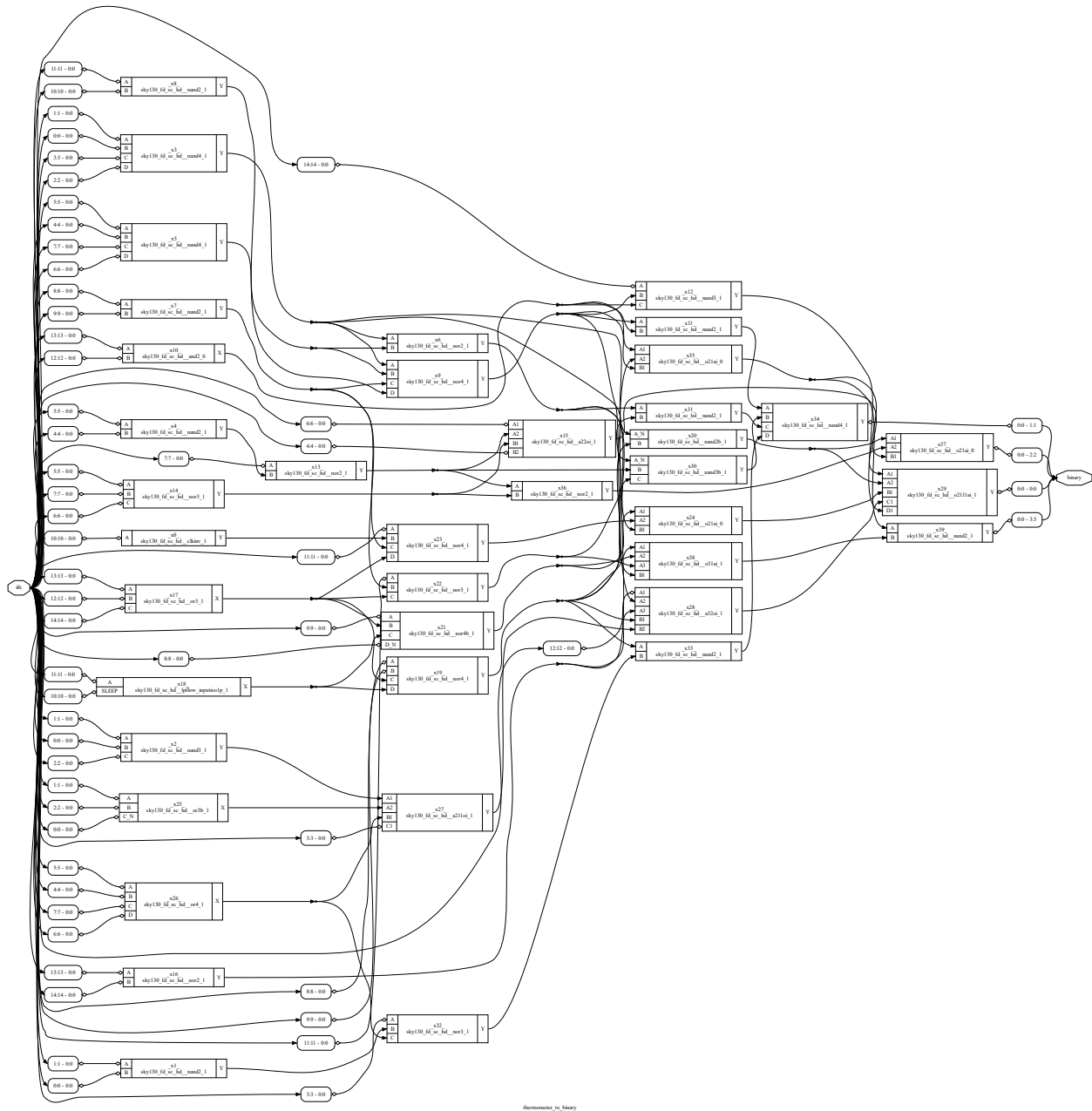
```
>> synth -top thermometer_to_binary // synthesize the module

>> abc -liberty
/usr/local/share/pdk/sky130A/libs.ref/sky130_fd_sc_hd/lib/sky130_fd_sc_hd__tt_025C_1v80.lib

>> flatten // // to invoke flat synthesis after netlist generation
// netlist generation
>> show // view graphical

>> write_verilog therm.v // write the generated netlist to a new module/verilog file
for verification
Or
>> write_verilog -noattr therm.v // // to write verilog netlist without
attributes(clean)
Optimize:
>> flatten
>> opt_clean -purge // running optimization
>> show

>> write_spice therm.spice
>> show -prefix therm -format svg // save diagram in .svg
Or
>> show -prefix therm -format dot // save diagram in .dot
Or
>> show -enum -prefix therm -format dot // save diagram in .dot with gate numbers
```

Netlist to schematic conversion:
<https://github.com/aidangoettsch/asg>

install NetlistViewer:
<https://github.com/fl8m/netlist-viewer/blob/master/NetlistViewer/build/linux/README.md>

doteditor:
<https://sourceforge.net/projects/netlistviewer/>

Iverilog with gtkwave:
<https://youtu.be/Ryt1ms8Tido>

Digital flow with qflow:

http://opencircuitdesign.com/qflow/tutorial_nogui.html

- simulate the complete (ultra small) ADC

References:

<https://github.com/bluecmd/learn-sky130/blob/main/schematic/xschem/getting-started.md>

Run iverilo/icarus

iverilog file.v

vvp a.out
