

LSI design training with SKY130

Analog LSI design Part 1

Mizuki Mori , <https://github.com/3zki>

Last update: 2024/12/31

Log

2024/12/31 Initial release of English version

2025/01/03 Move Vds-Ids curve section to Part 2.

Lab data

- Development Environment Setup Script (Windows WSL + Ubuntu 22)
 - https://github.com/3zki/wsl_osic
- Analog LSI design Part 1
 - <https://github.com/3zki/lab240921>

Please download these materials by “git clone” etc...

Analog LSI design Part 1

Make a CMOS inverter (NOT gate)

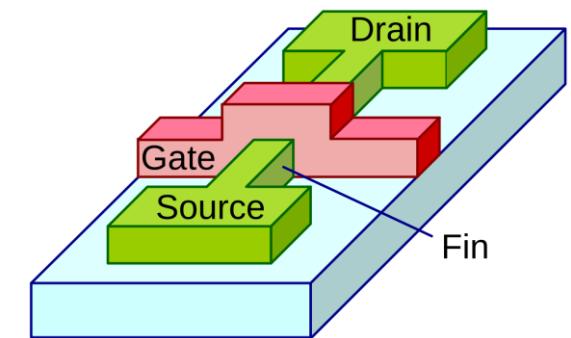
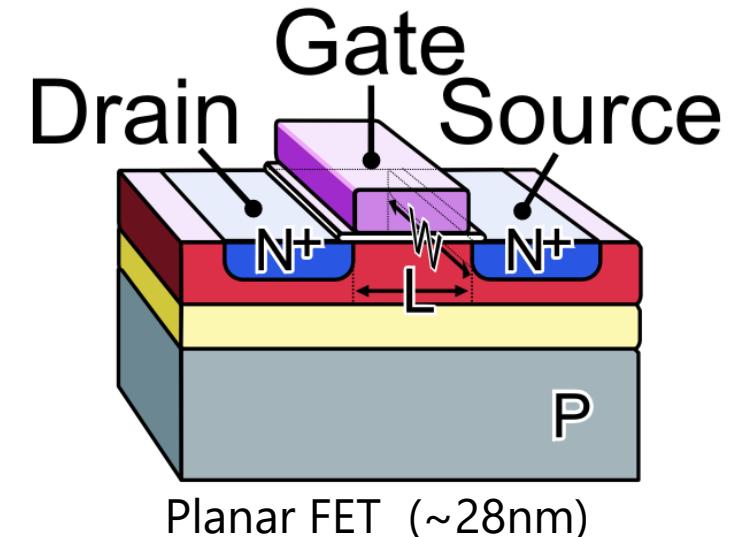
- Background knowledge
- Schematic design
- Layout design
- Post layout simulation (Parasitic extraction)

Background knowledge of LSI

Planar FET structure

What is LSI?

- There are various types of LSIs.
Design methods and characteristics differ among them.
- SKY130 and GF180MCU are **planar MOSFET process** 
We can make Pch MOSFET and Nch MOSFET
- The manufacturing limit for planar FET is about 28 nm.
High-density logic circuits such as CPU use completely different process such as FinFET and GAA-FET.
- Planar 130nm is obsoleted for logic circuits,
but cheaper than the FinFET and still be used for analog LSI.



https://commons.wikimedia.org/wiki/File:Multigate_models.png

[https://commons.wikimedia.org/wiki/File:MOS-FET_gate_model_\(n-channel\)_E.PNG](https://commons.wikimedia.org/wiki/File:MOS-FET_gate_model_(n-channel)_E.PNG)

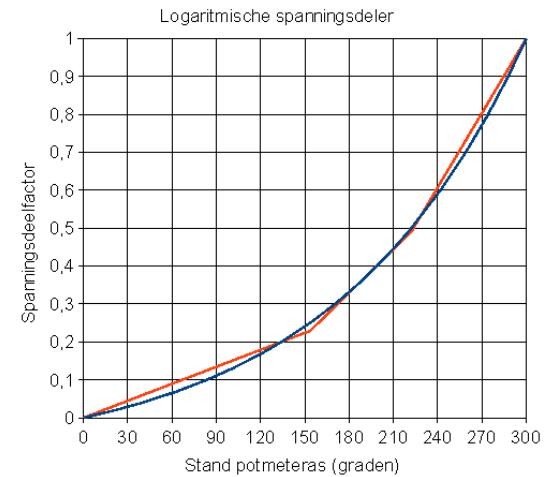
What can we make with the planar FET process?

- Planar MOSFET: **PMOS**, **NMOS**
 - SKY130 supports 1.8V, 3.3V 5.0V, 10.5V, 16V, 20V devices
- Diode
- Resistor: Polysilicon resistor, Active Resistor (Diffusion Resistor), Nwell Resistor
- Capacitor: MOS capacitor, Diffusion Capacitor, **MIMCAP***,
MOMCAP(Metal-Oxide-Metal)
- Varactor diode(Varicap)*
- Lateral bipolar transistor*: PNP, NPN

* = Not supported for some processes. SKY130 supports them.

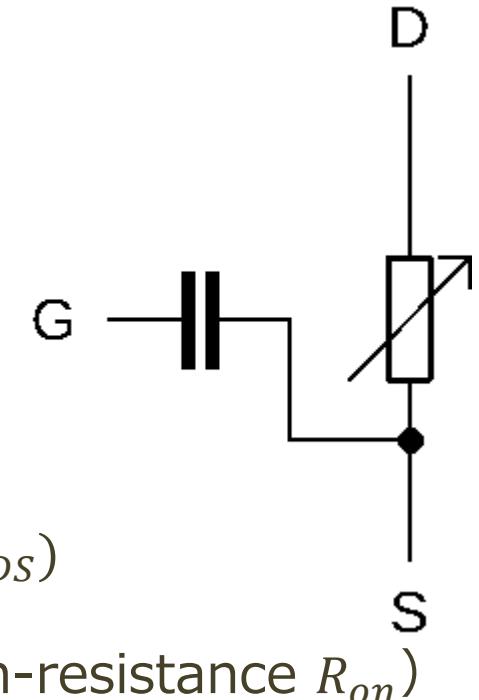
What is the MOSFET?

- Similar to a potentiometer.
 - Turning the knob changes the resistance from $\approx 0\Omega$.
 - Main characteristics are taper and maximum resistance value.
 - Taper: relation between the angle of knob and the resistance.
(For example: log, linear, inverse-log)



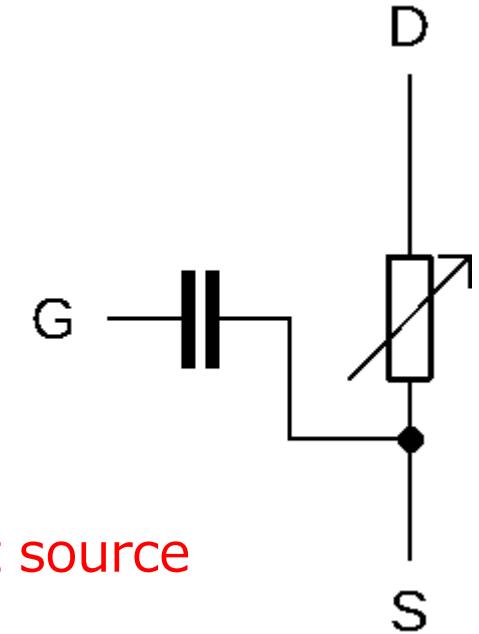
What is the MOSFET?

- In MOSFET, the characteristics are based on current.
(Basically, we never think in terms of resistance.)
- Difference from potentiometer
 - Turning the Gate voltage V_{GS} changes the current between Drain and Source from $\div 0A$ ($\div \infty\Omega$)
 - Main characteristics are current curves (Drain-Source current I_{DS})
 - Minimum resistance is calculated from the current curve (on-resistance R_{on})
 - Characteristics of current variation is process-dependent
(Threshold voltage V_{th} , Transconductance g_m)

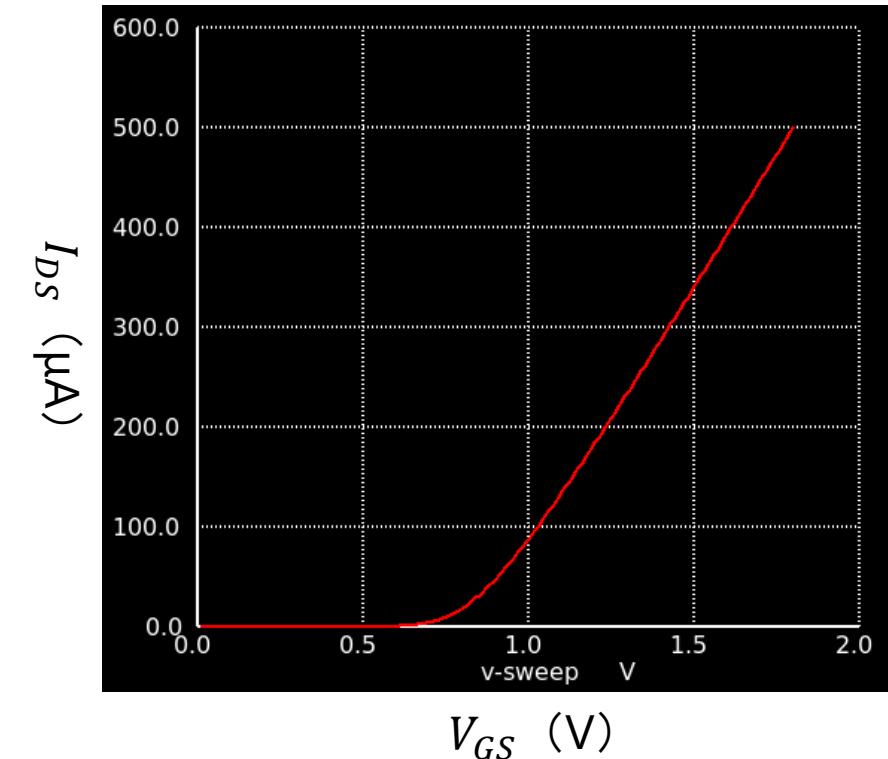
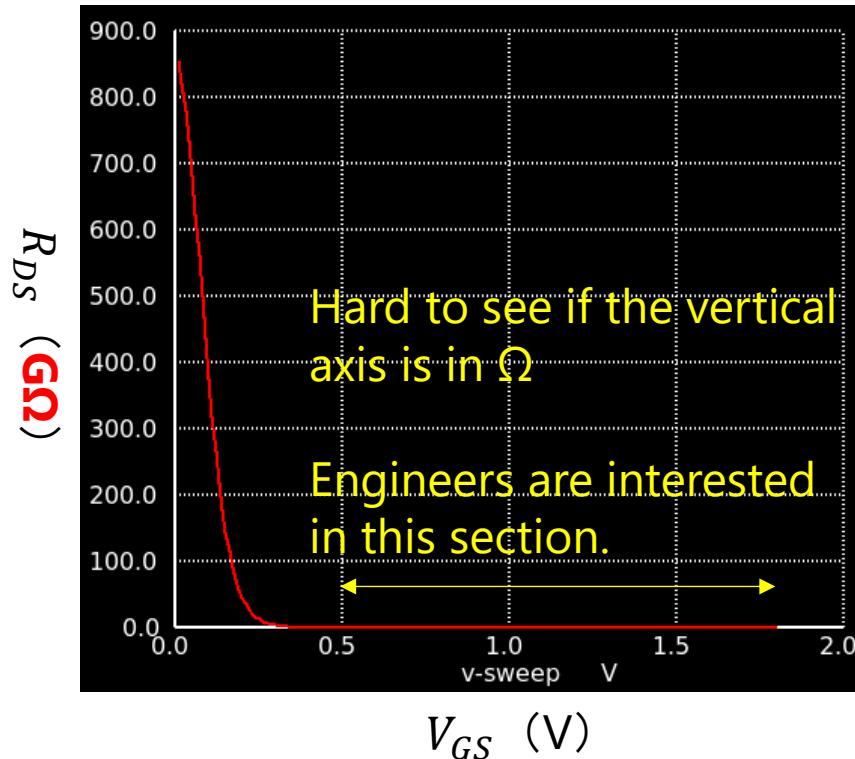
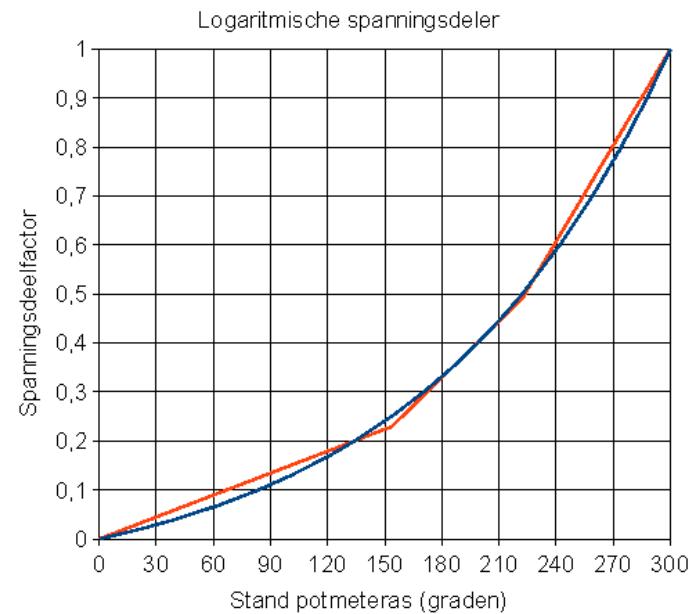


What is the MOSFET?

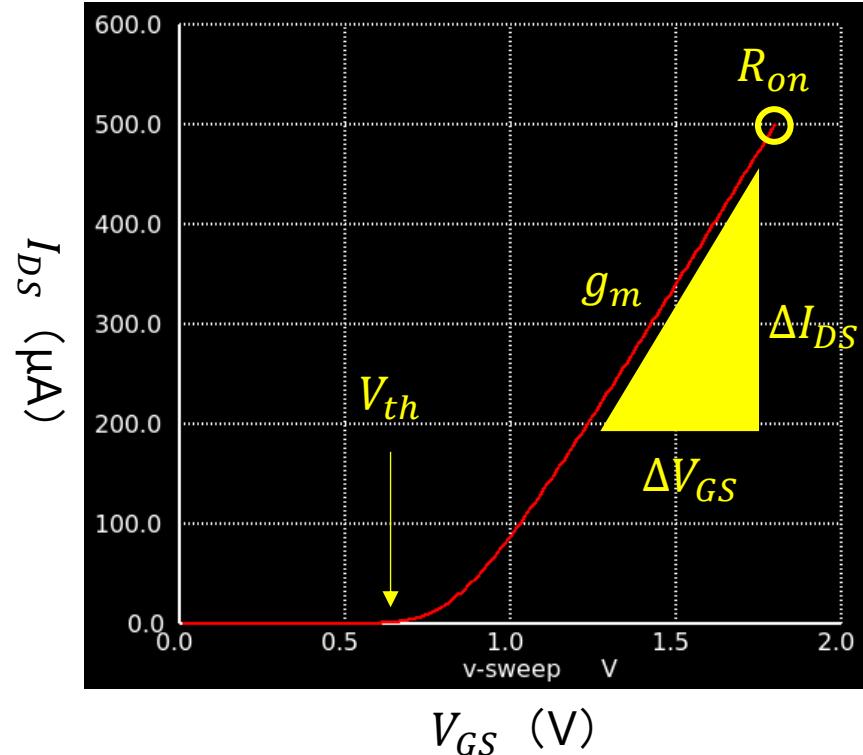
- Equivalent circuit representing a MOSFET as a passive element
- The variable resistor between drain and source should be a current source. (but it is also inaccurate though.)
- The main characteristic of MOSFETs is **transconductance**.
$$I_{DS} = \frac{1}{2} \times g_m \times (V_{GS} - V_{th})$$
- Another main characteristic is **saturation** which acts as a current source in a specific Drain-Source voltage (V_{DS}) region.
- Gate capacitance also exists between gate-drain and gate-body, but only gate-source is drawn due to visual issues.



Potentiometer vs. nMOS



Characteristics of nMOS



- V_{th} Threshold Voltage
Threshold gate-source voltage at which current begins to conduct
- g_m Transconductance
Gate-source voltage variation vs. drain-source current variation
- R_{on} on-resistance
Drain-source resistance in on-state

Design flow of Analog LSI

How to design Analog LSI

Design flow of Analog LSI

1. Draw schematics and test benches
2. Simulation
3. Draw layouts based on schematics
4. Layout verification
5. Post-layout simulation: Simulation with parasitic components
6. (Place on frame)

Draw the schematics and test benches

MODELS

VDD

A

Y

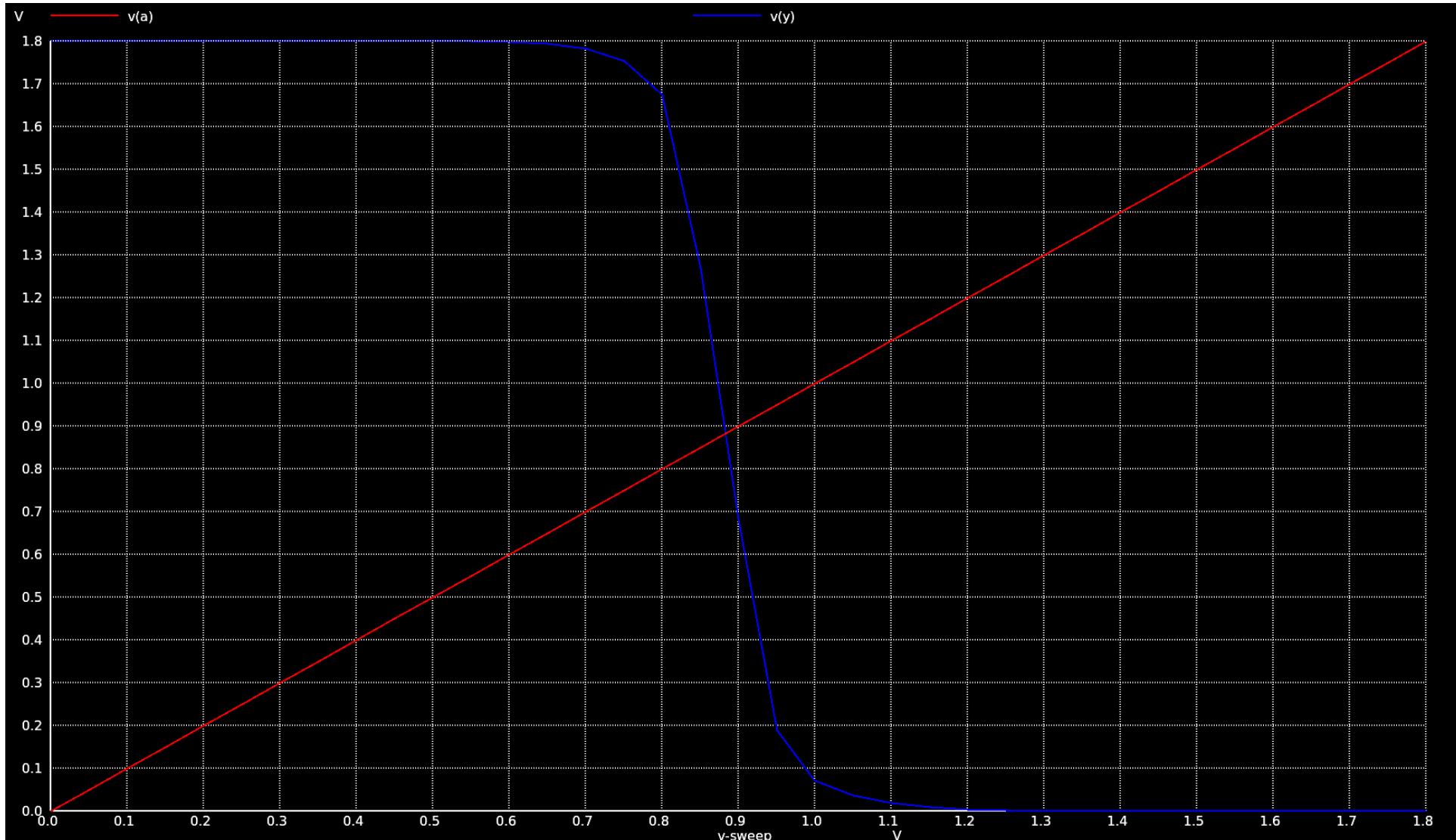
GND

COMMANDS

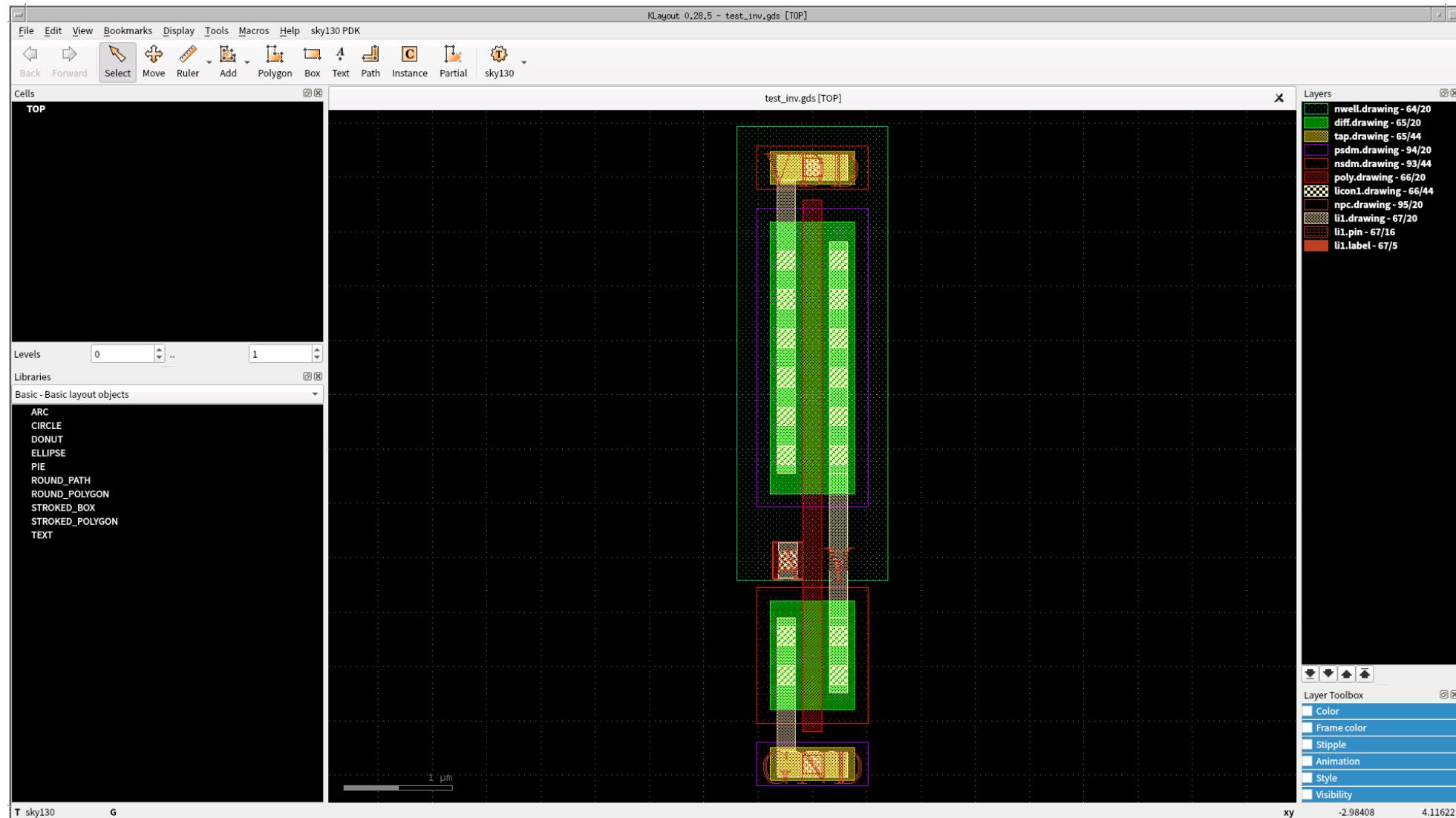
SIM=ngspice

```
VD VDD 0 dc 1.8
VA A 0 dc 0
.control
save all
dc VA 0 1.8 0.05
plot v(a) v(y)
.endc
```

Simulate



Draw the layouts



Extract parasitic components

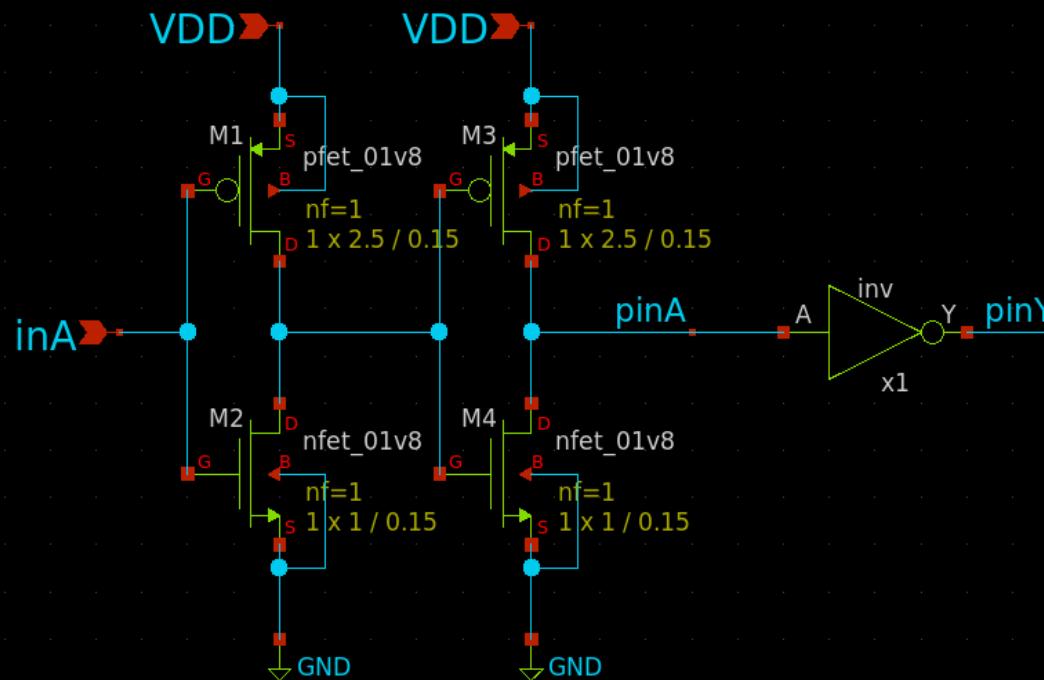
```
Magic-PEX

File Edit View Search Terminal Help
Loading sky130A Device Generator Menu ...
Loading "/home/user/.klayout/macros/sky130_magic_pex.tcl" from command line.
Warning: Calma reading is not undoable! I hope that's OK.
Library written using GDS-II Release 6.0
Library name: LIB
Reading "TOP".
CIF file read warning: CIF style sky130(): units rescaled by factor of 5 / 1
Extracting TOP into TOP.ext:
exttosim finished.
exttospice finished.
exttospice finished.
* NGSPICE file created from TOP.ext - technology: sky130A

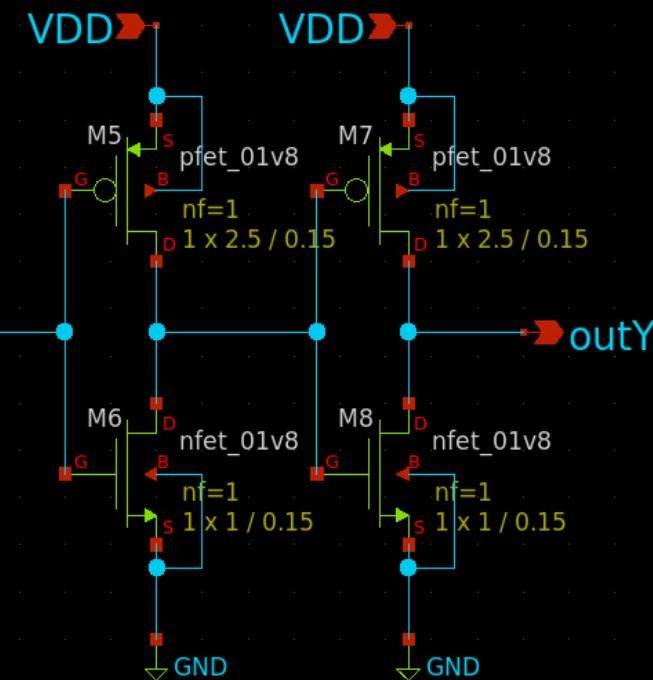
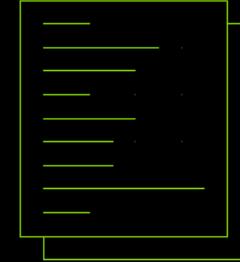
.subckt TOP A Y VDD GND
X0 Y A VDD VDD sky130_fd_pr_pfet_01v8 ad=7.5e+11p pd=5.6e+06u as=7.5e+11p ps=5.
6e+06u w=2.5e+06u l=180000u
X1 Y A GND GND sky130_fd_pr_nfet_01v8 ad=3e+11p pd=2.6e+06u as=3e+11p ps=2.6e+0
6u w=1e+06u l=180000u
C0 A Y 0.05fF
C1 VDD Y 0.17fF
C2 A VDD 0.18fF
.ends
```

Make test benches again

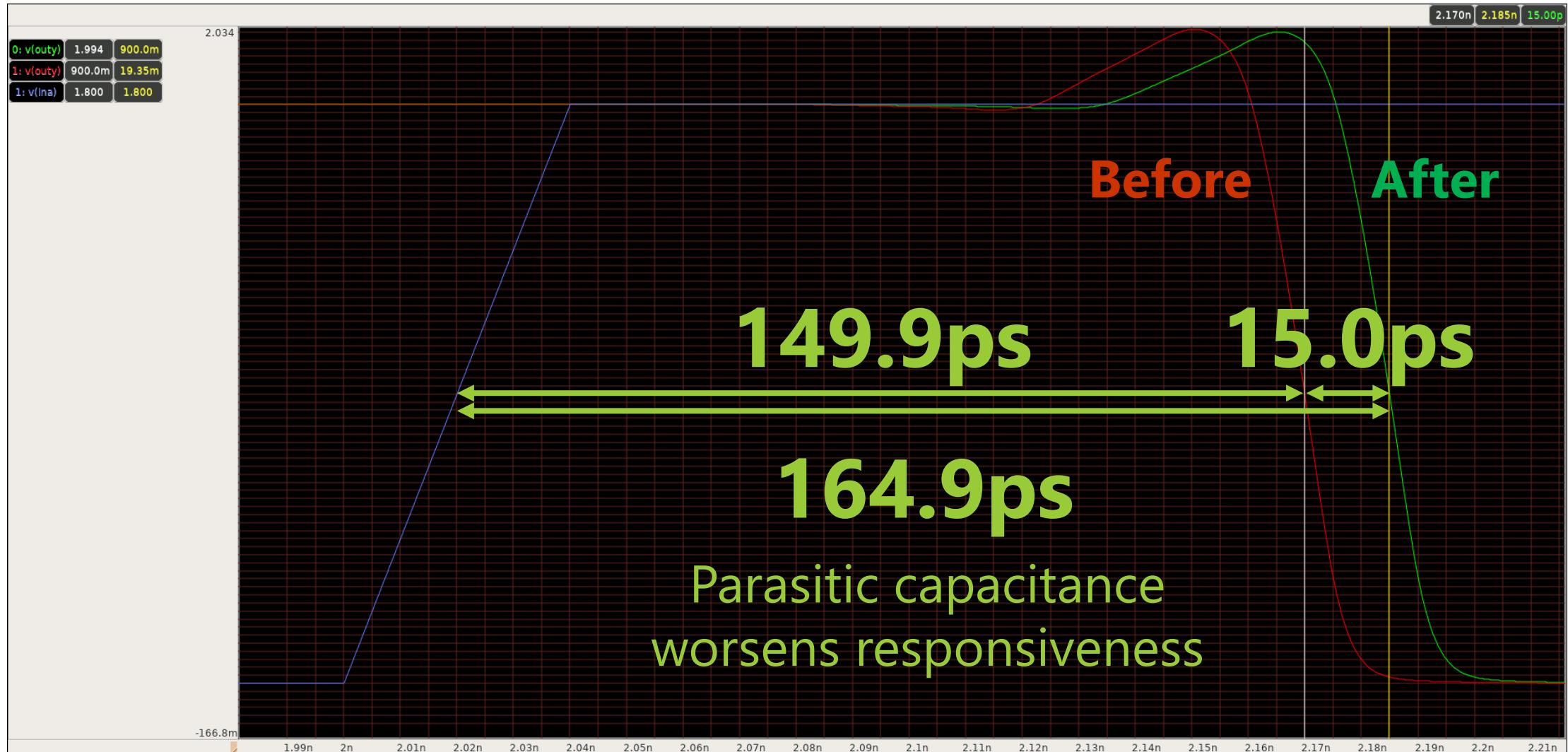
```
ngspice_
VA inA 0 pulse(0 1.8 0 40p 40p 1n 2n) dc 0
VD VDD 0 dc 1.8
.include ~/TOP_pex_extracted.spice
.control
.tran 1p 4n
.wrdata ~/inv_bench.txt v(ina) v(outy)
.write ~/inv_test_pex.raw
.endc
```



MODELS



Simulate with parasitic components (Post-layout simulation)



Background knowledge of LSI

Know the characteristics of the process

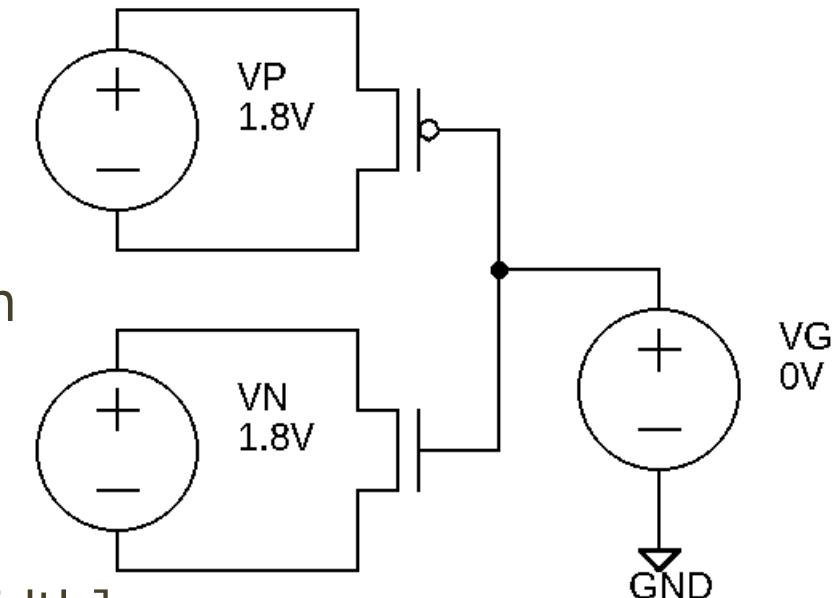
Before designing...

- We want to know the characteristics of CMOS!
 - V_{gs} – I_{ds} curve
 - Operational Point analysis
 - ~~V_{ds} – I_{ds} curve~~ will explain in Analog LSI design Part 2

V_{gs} – I_{ds} curve / on-resistance measurement

Simulate the on-resistance of CMOS ($L=0.15\mu m$, $W=1.0\mu m$, $V_{ds}=1.8V$)

1. Place voltage sources v_p ($V=1.8$) and v_n ($V=1.8$) between drain and source
2. Place voltage source $v_g(V=0)$ between gate and GND
3. Sweep the voltage of v_g between 0V and 1.8V by **DC analysis**
4. Find current consumption of voltage sources v_p and v_n



Format of DC analysis

dc [voltage source] [start voltage] [end voltage] [step width]

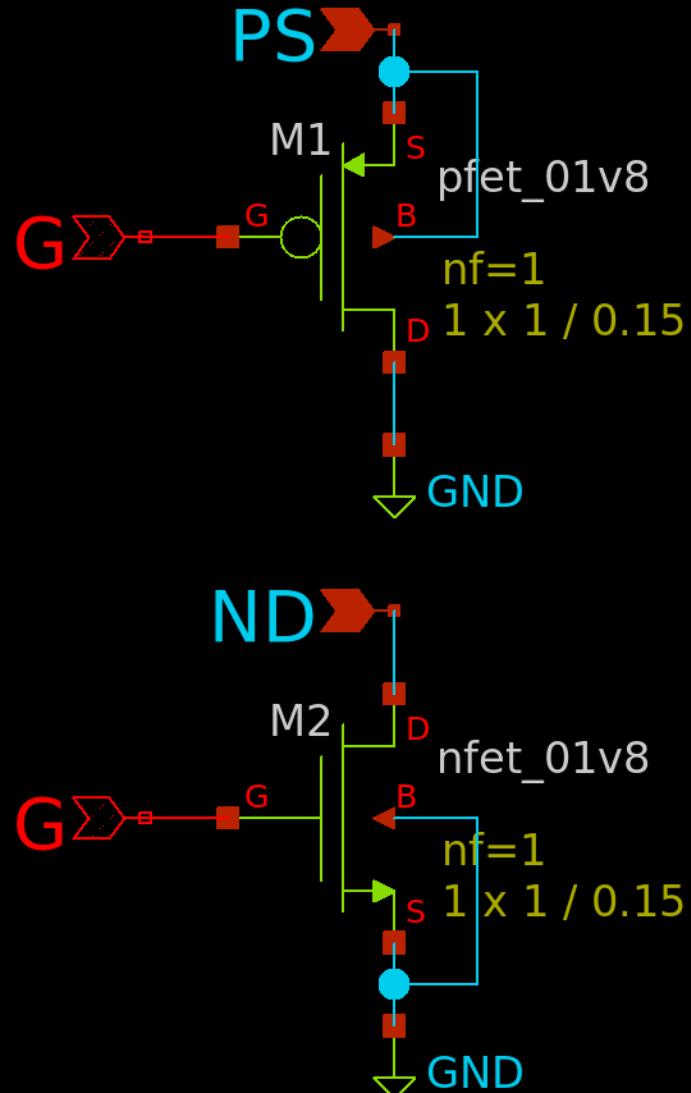
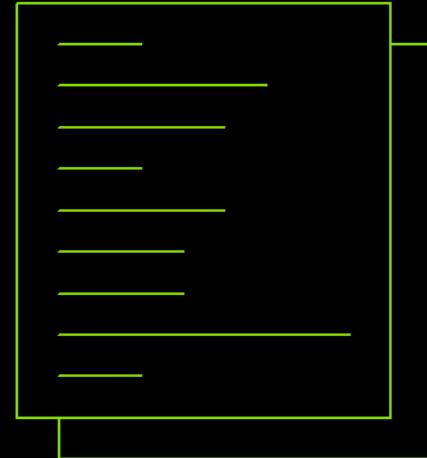
Current consumption of voltage sources

ngspice: v_p #branch, v_n #branch or gaw(waveform viewer): $i(v_p)$, $i(v_n)$

Example of test bench ($L=0.15\mu\text{m}$)

vgsbench.sch

MODELS

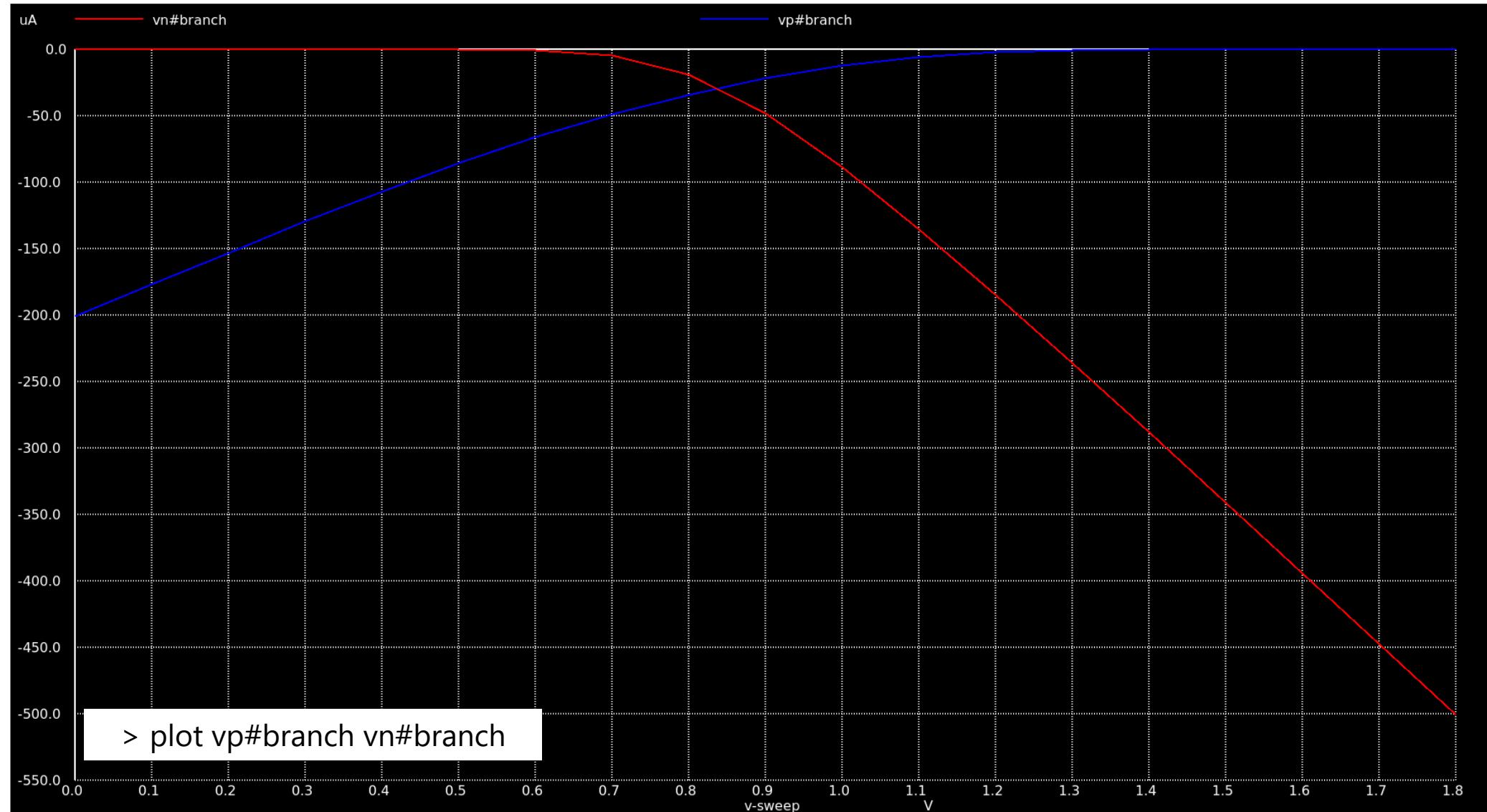


ngspice

```
VG G 0 dc 0
VP PS 0 dc 1.8
VN ND 0 dc 1.8
.control
save all
dc VG 0 1.8 0.1
write trbench.raw
.endc
```

Be sure to include GND or 0
as a reference.
"VSS" is not converted to 0

Simulation results



Output data

wrdata command saves specific data

*) hoge#branch -> current consumption of voltage source hoge

vp.txt

0.0000000e+00	-2.00726126e-04
1.0000000e-01	-1.76783868e-04
2.0000000e-01	-1.52856174e-04
3.0000000e-01	-1.29370810e-04
4.0000000e-01	-1.06779404e-04
5.0000000e-01	-8.55349063e-05
6.0000000e-01	-6.60656564e-05
7.0000000e-01	-4.87492484e-05
8.0000000e-01	-3.38904626e-05
9.0000000e-01	-2.17083481e-05
1.0000000e+00	-1.23365952e-05
1.1000000e+00	-5.82743752e-06
1.2000000e+00	-2.07958988e-06
1.3000000e+00	-5.36969830e-07
1.4000000e+00	-1.19564554e-07
1.5000000e+00	-2.81003906e-08
1.6000000e+00	-6.92577344e-09
1.7000000e+00	-1.63614705e-09
1.8000000e+00	-3.19909099e-10

vn.txt

0.0000000e+00	-2.03659312e-12
1.0000000e-01	-4.80682161e-12
2.0000000e-01	-3.99689171e-11
3.0000000e-01	-4.84302376e-10
4.0000000e-01	-5.99945693e-09
5.0000000e-01	-7.04327370e-08
6.0000000e-01	-6.97912479e-07
7.0000000e-01	-4.74592920e-06
8.0000000e-01	-1.91451559e-05
9.0000000e-01	-4.83465908e-05
1.0000000e+00	-8.87674453e-05
1.1000000e+00	-1.35141610e-04
1.2000000e+00	-1.84593052e-04
1.3000000e+00	-2.35816015e-04
1.4000000e+00	-2.88115929e-04
1.5000000e+00	-3.41048569e-04
1.6000000e+00	-3.94303222e-04
1.7000000e+00	-4.47655558e-04
1.8000000e+00	-5.00941446e-04

```
ngspice
    VG G 0 dc 0
    VP PS 0 dc 1.8
    VN ND 0 dc 1.8
    .control
    dc VG 0 1.8 0.1
    wrdata ~/vp.txt vp#branch
    wrdata ~/vn.txt vn#branch
    .endc
```

On-resistance

Characteristics of on-state at L=0.15μm, W=1.0μm

PMOS: 0.0000000e+00 -2.00726126e-04

$$V_{ds} = 1.8V \rightarrow 1.8/2.01e-4 = 9.0k\Omega$$

NMOS: 1.8000000e+00 -5.00941446e-04

$$V_{ds} = 1.8V \rightarrow 1.8/5.01e-4 = 3.6k\Omega$$

On-resistance and W are roughly inversely related

$$\rightarrow \text{aligned at } W_P : W_N = 2.5 : 1$$

We usually change W instead of L.

Operational Point analysis (OP)

opbench.sch

- **op** command calculates parameters such as vth and gm.
- Specify the proper voltage conditions.
 - gm, vdsat etc... are voltage dependent.

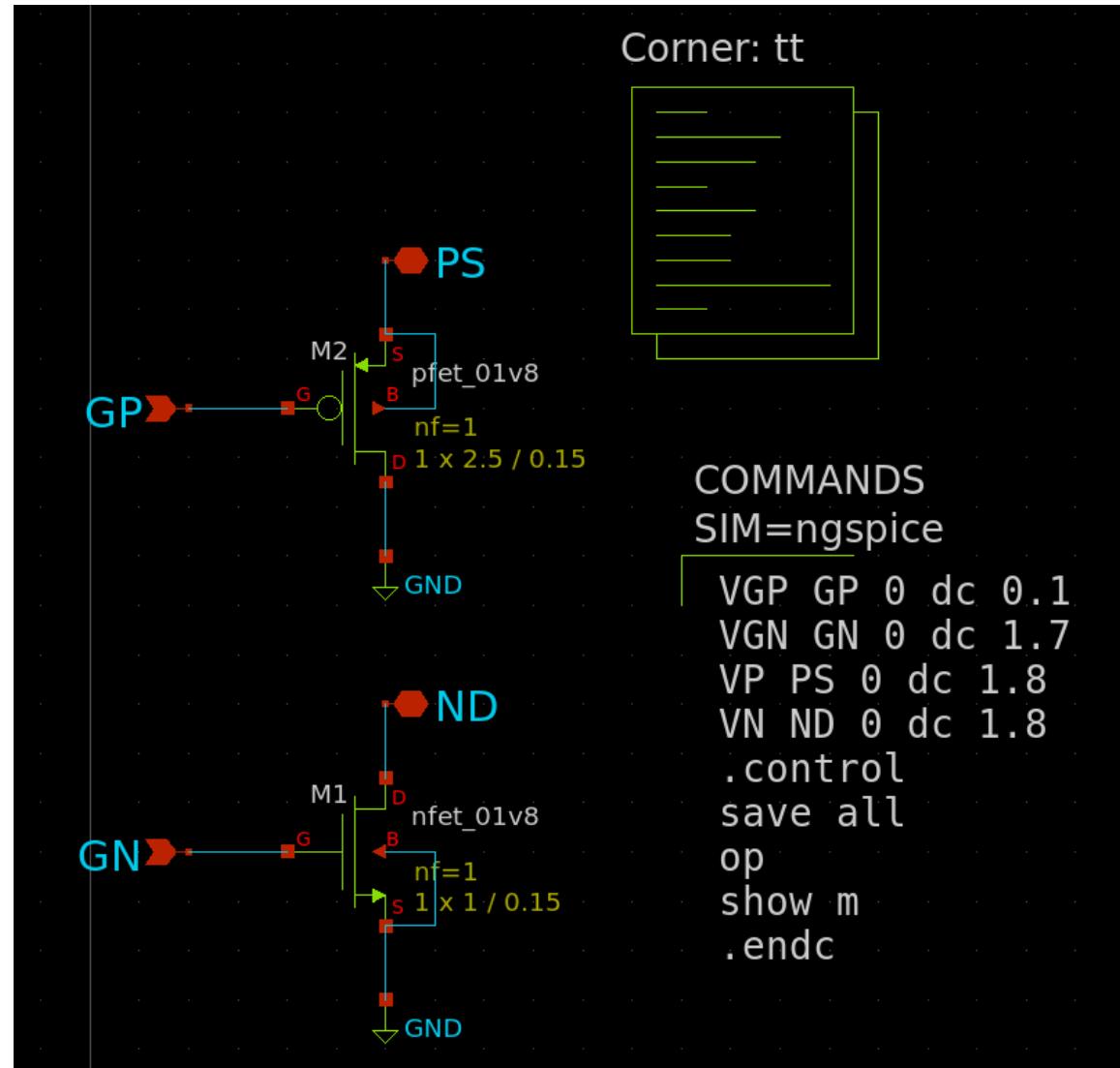
Off-state

gm	3.79481e-08	7.63463e-11
gds	4.38991e-09	2.43751e-12
vdsat	0.0447906	0.0430537
vth	0.561294	0.769268

On-state

gm	0.00060858	0.000533543
gds	9.80226e-05	4.93687e-05
vdsat	0.755181	0.343415
vth	0.561319	0.769291

- gm is useful for calculating currents by hand.



Note: transconductance gm

- Have you ever seen these formulas before?

$$I_{DS} = \frac{W}{L} \mu_n C_{ox} \left[(V_{GS} - V_{th}) V_{DS} - \frac{1}{2} V_{DS}^2 \right] \quad \text{in linear region } (V_{DS} \leq V_{GS} - V_{th})$$

$$I_{DS} = \frac{W}{L} \mu_n C_{ox} \left[\frac{1}{2} (V_{GS} - V_{th})^2 \right] \quad \text{in saturation region } (V_{DS} \geq V_{GS} - V_{th})$$

- gm is the derivative of the above equation by $(V_{GS} - V_{th})$, that is,

$$g_m = \frac{W}{L} \mu_n C_{ox} V_{DS} \quad \text{in linear region } (V_{DS} \leq V_{GS} - V_{th})$$

$$g_m = \frac{W}{L} \mu_n C_{ox} (V_{GS} - V_{th}) \quad \text{in saturation region } (V_{DS} \geq V_{GS} - V_{th})$$

$$\rightarrow I_{DS} = \frac{1}{2} \times g_m \times (V_{GS} - V_{th}) \quad \text{in saturation region}$$

ngspice command: show m

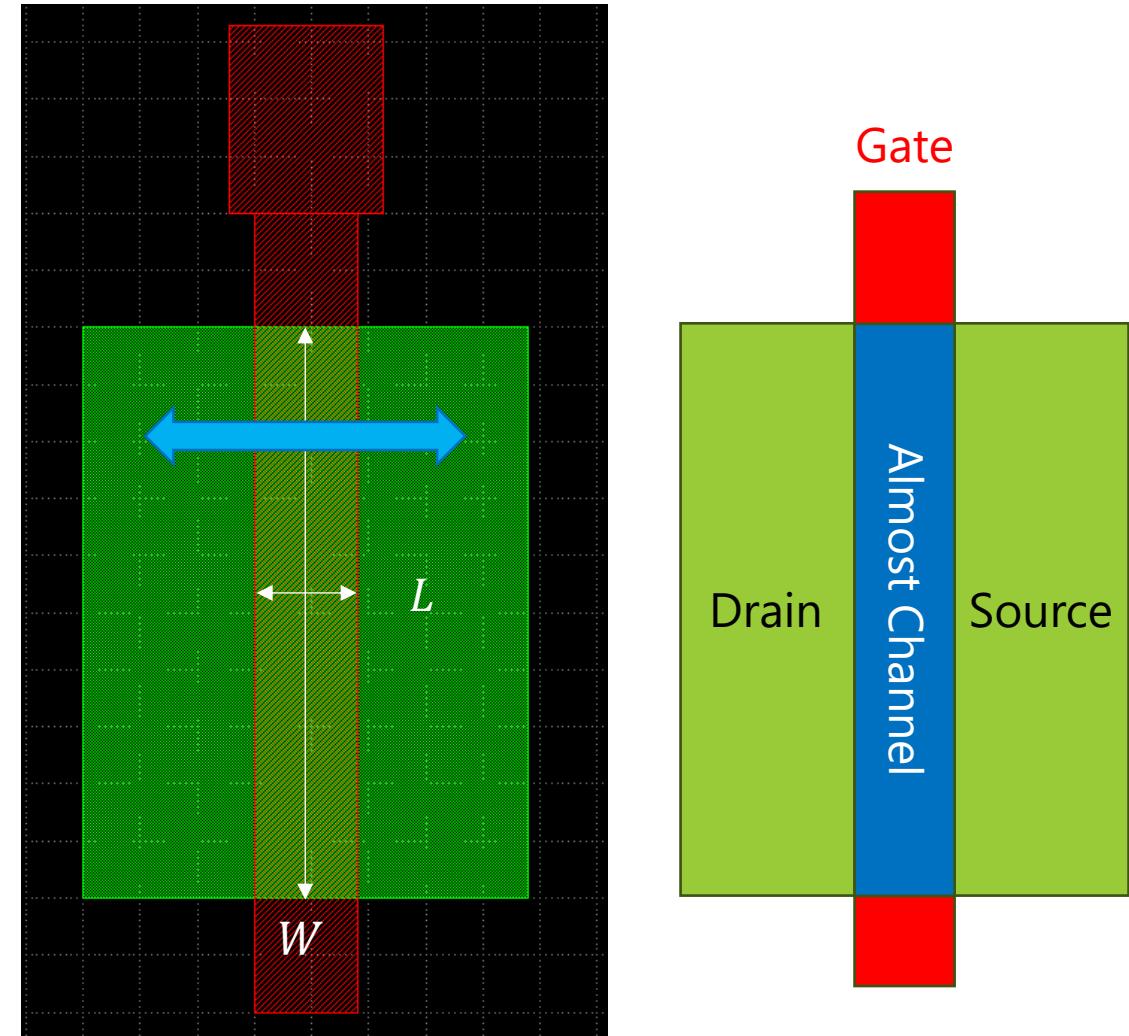
The results can be saved to plain text.

- show m > ~/test.txt

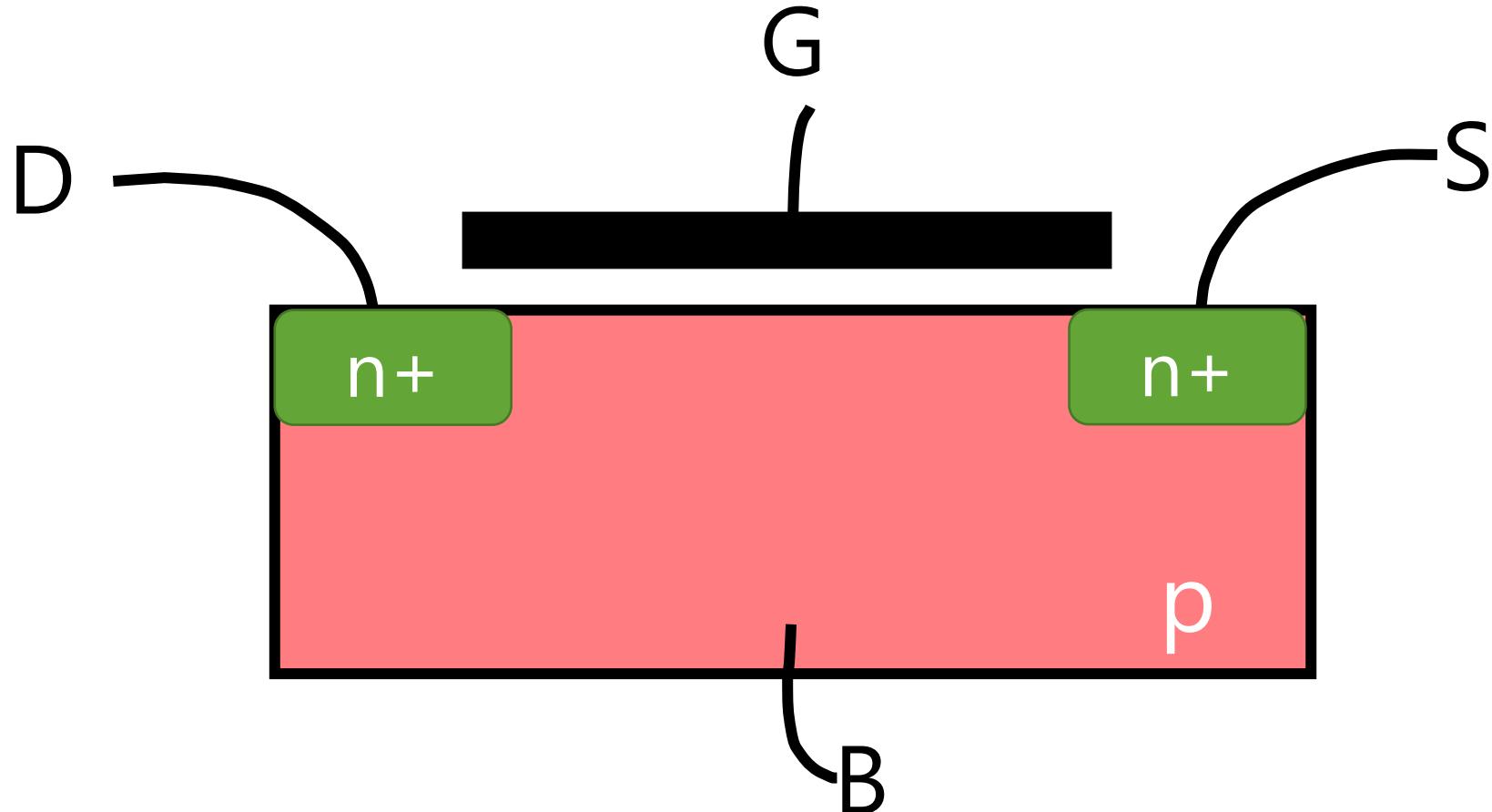
device	m.xm2.msky130_fd_pr__	m.xm1.msky130_fd_pr__
model	xm2:sky130_fd_pr_pfe	xm1:sky130_fd_pr_nfe
l	1.5e-07	1.5e-07
w	1e-06	1e-06
nf	1	1
sa	0	0
sb	0	0
sd	0	0
sca	0	0
scb	0	0
scc	0	0
sc	0	0
min	0	0
ad	2.9e-13	2.9e-13
as	2.9e-13	2.9e-13
pd	2.58e-06	2.58e-06
ps	2.58e-06	2.58e-06
nrd	0.29	0.29
nrs	0.29	0.29
off	0	0
rbdb	50	50
rbsb	50	50
rbpb	50	50
rbps	50	50
rbpd	50	50
delvto	0	0
mulu0	1	1
xgw	0	0
ngcon	1	1
trnqsmod	0	0
acnqsmod	0	0
rbodymod	1	1
rgatemod	0	0
geomod	0	0
rgeomod	0	0
gmbs	4.87021e-05	1.57575e-12
am	0.000238192	6.00366e-12

Parameter of transistor

- Basically, we change (set) W and L
 - W : Channel width
 - L : Channel length
 - *) Actual channel length is a bit short
- **The overlapped area of the polysilicon gate and diffusion is the “transistor”**
When sufficient potential is given to the gate, a **channel** is formed. ($V_{gs} > V_{th}$)
- m, nf, mf, etc... specify the number of multi-finger gate.
(Described later in the layout description)

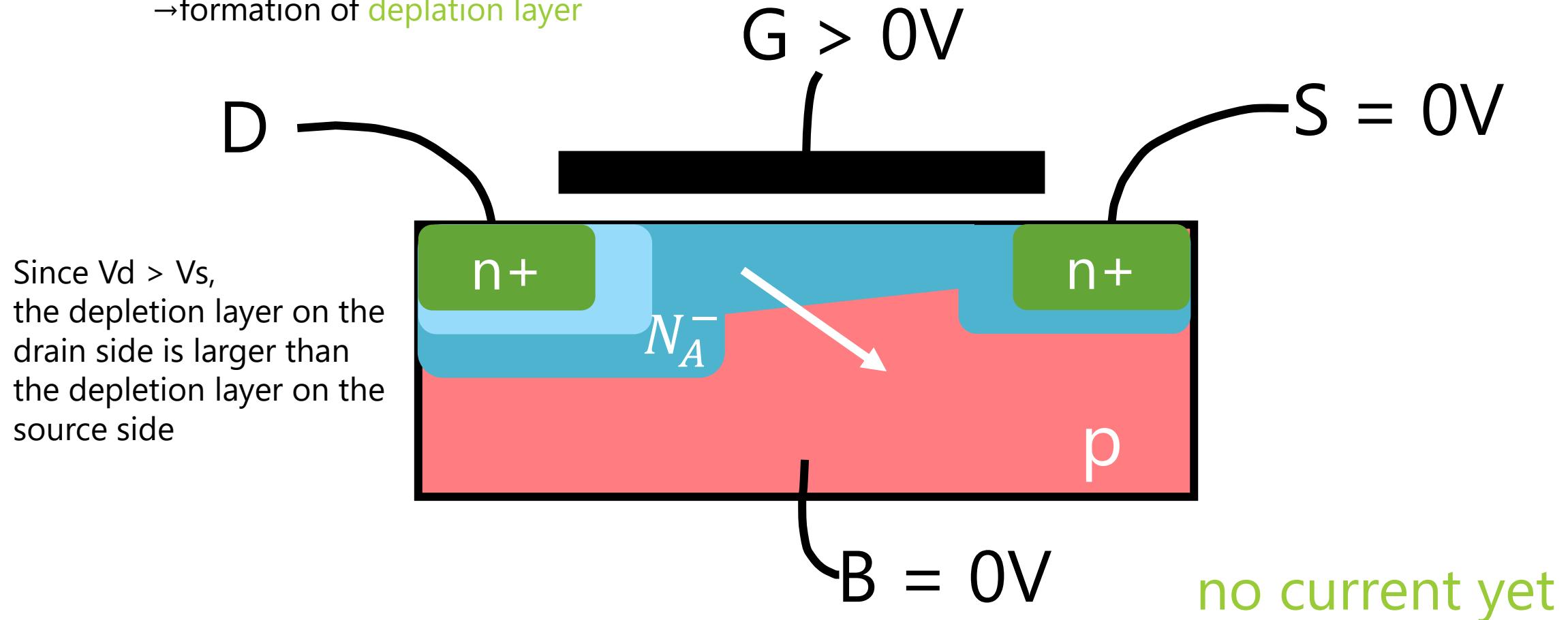


nMOS diagram



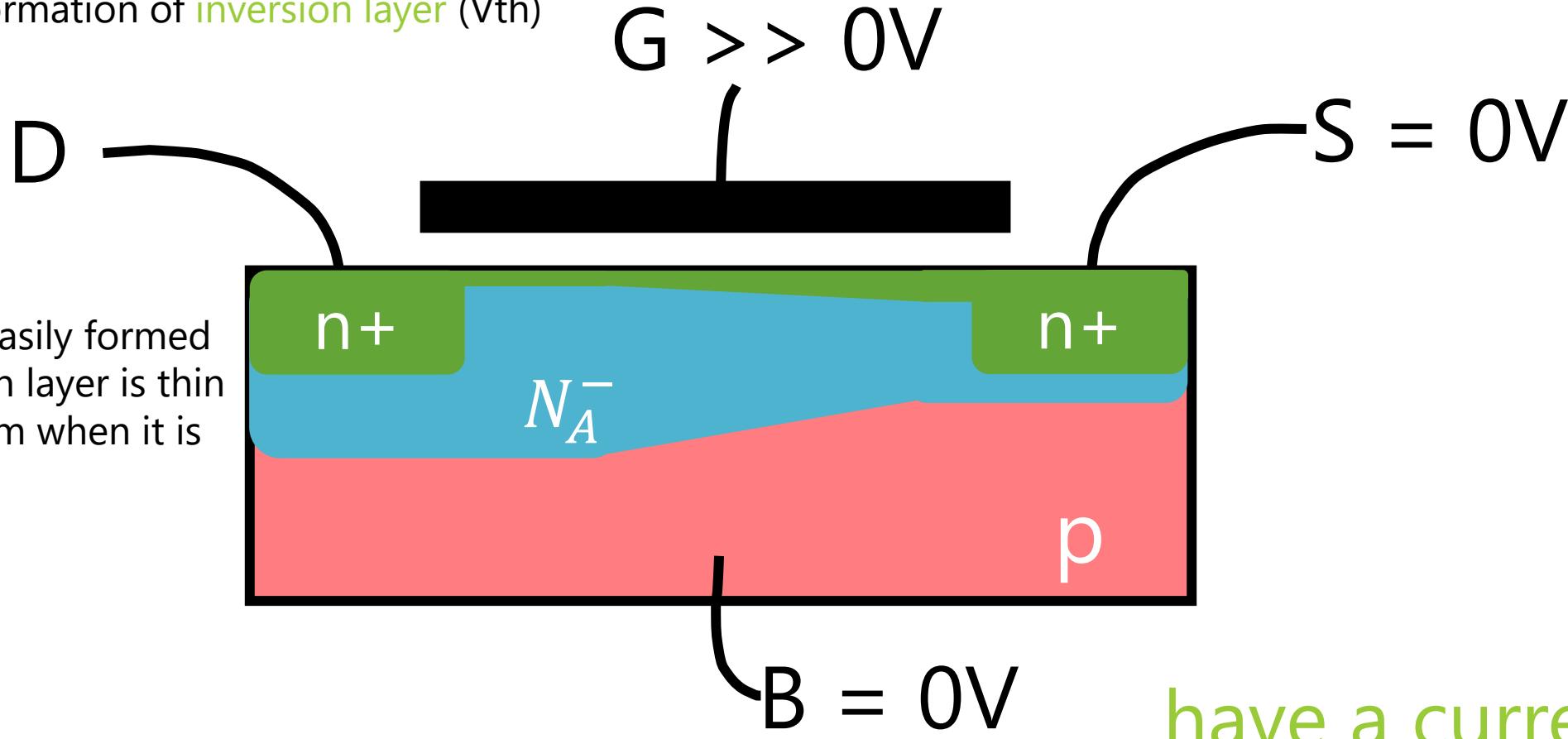
nMOS diagram

Hole on substrate moves to gate at $V_g > 0$
→ formation of **depletion layer**



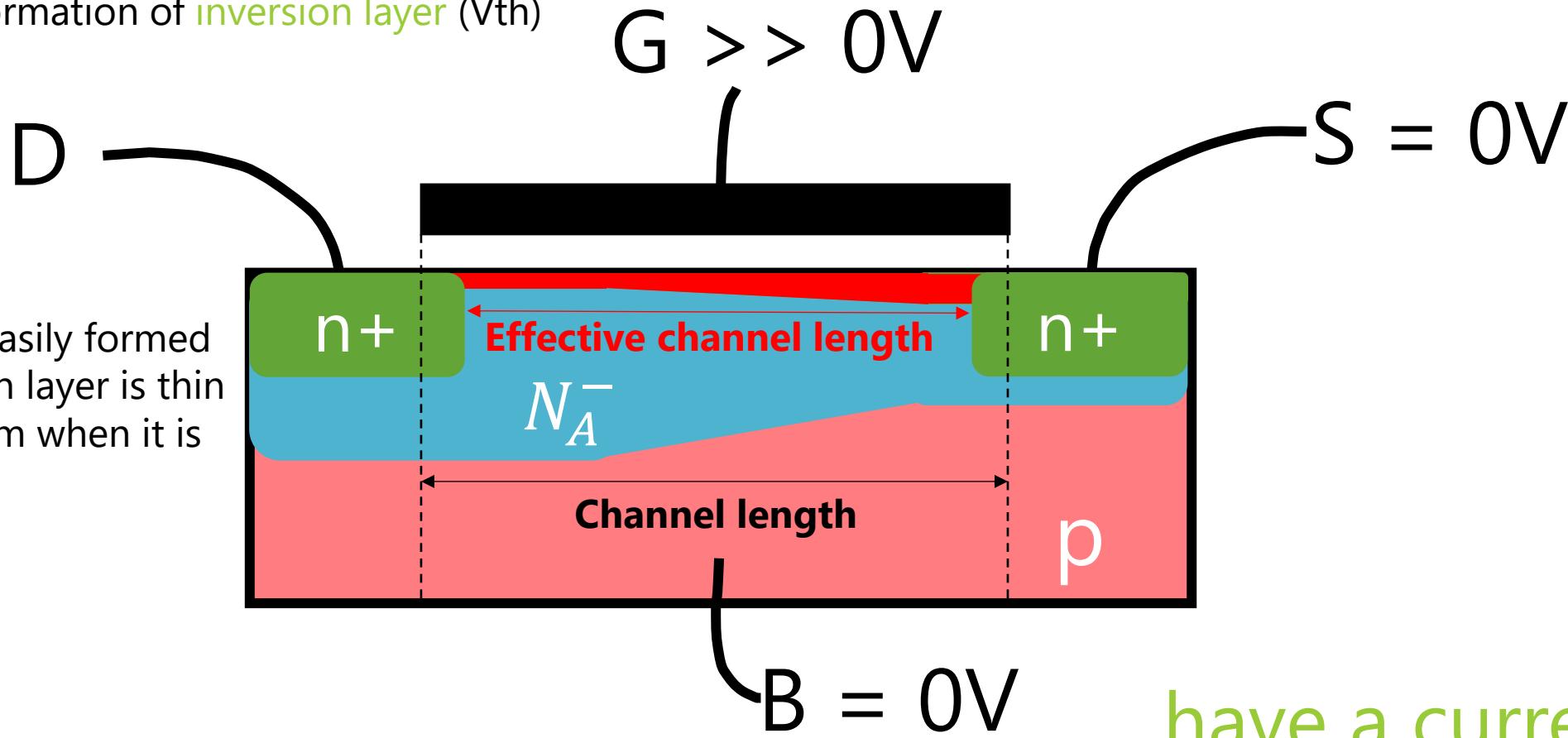
nMOS diagram

Conduction electrons become majority carriers at $V_g >> 0$
→ Formation of **inversion layer** (V_{th})



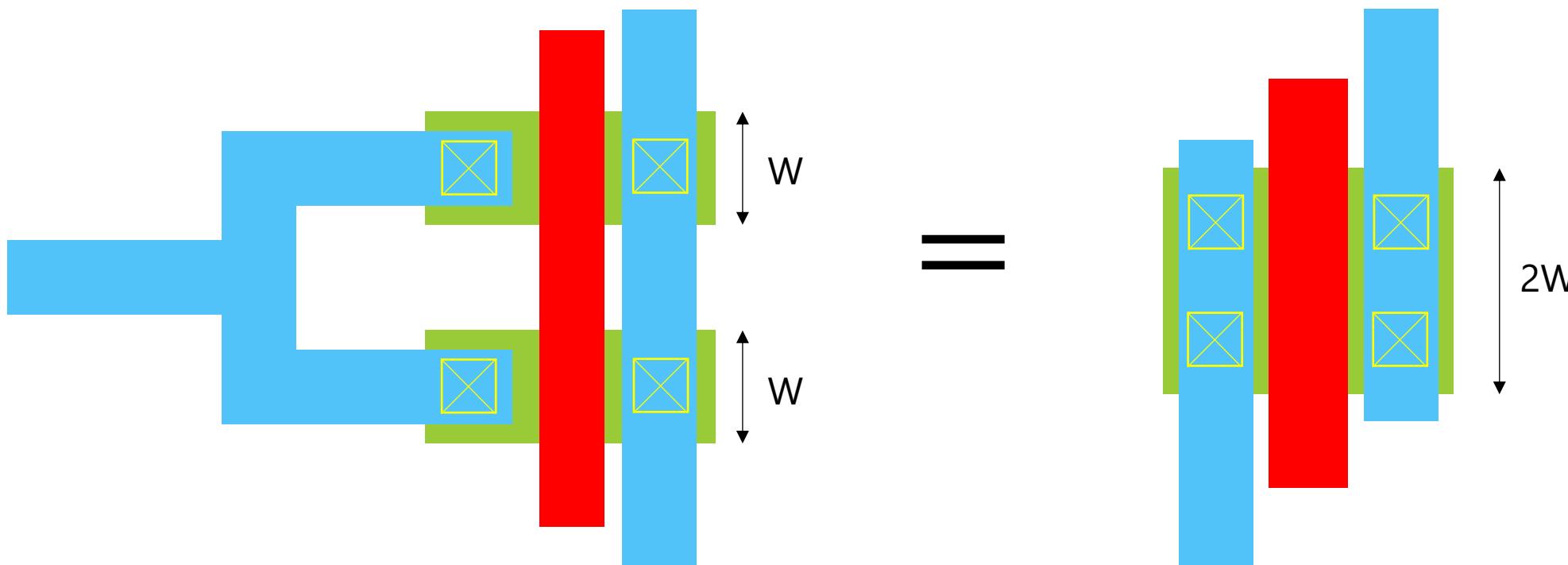
nMOS diagram

Conduction electrons become majority carriers at $V_g >> 0$
→ Formation of **inversion layer** (V_{th})



What happens if channel width W is changed?

- A larger channel width results in a wider current path, which increases the drain-source current.
- For transistors with the same channel length,
the channel width is proportional to the drain-source current.



What happens if channel length L is changed?

- The effect of the **short channel effect** completely changes the characteristics.
Basically, I_{ds} increases as channel length L decreases,
but short-channel effects also occur...
- **DIBL** (Drain Induced Barrier Lowering) decreases V_{th} for an increase in V_{ds} .
- Increasing **subthreshold swing** increases punch-through in subthreshold region
 - Leakage current in the off-state increases, and I_{ds} variation relative to V_{gs} variation becomes smaller (slower) → Switching characteristics deteriorate
- **Channel length modulation effect** increases I_{ds} variation relative to V_{ds} variation and **does not saturate (The current is not constant on the V_{ds} - I_{ds} curve.)**

What happens if channel length L is changed?

- The effect of the **short channel effect** completely changes the characteristics.
Basically, I_{ds} increases as channel length L decreases,
but short-channel effects also occur...

Logic gates are usually made with the **shortest length**
as higher density, lower V_{th} and higher I_{ds} are preferred.

We will not focus on channel length in this part.

Demonstration of Analog LSI design

Make a CMOS inverter

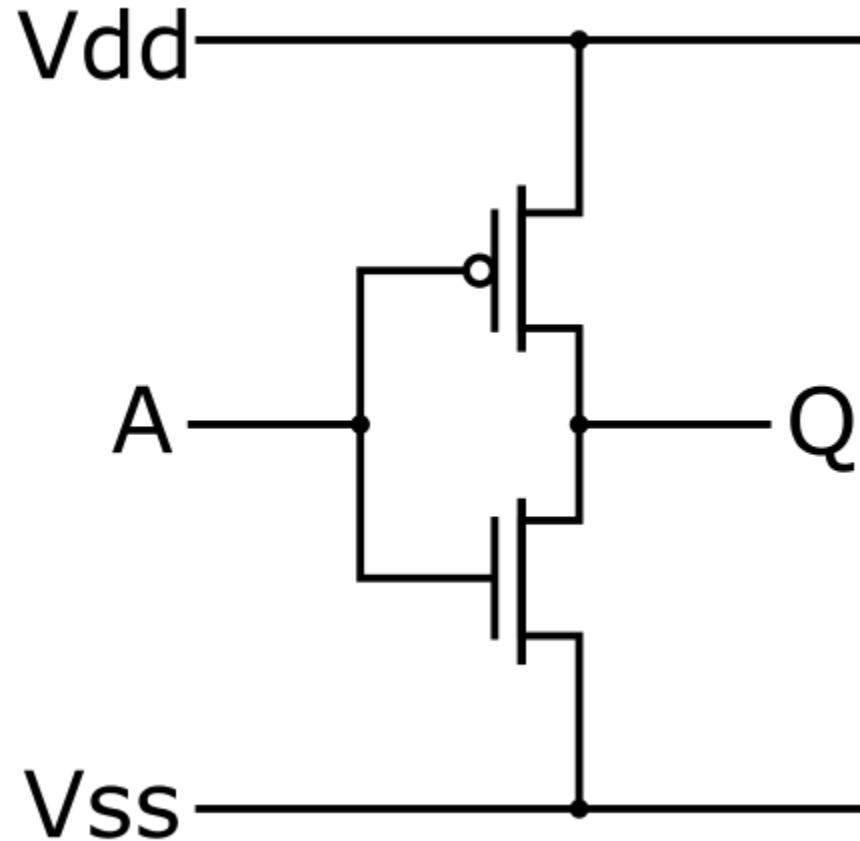
Design flow of Analog LSI

1. Draw schematics and test benches
2. Simulation
3. Draw layouts based on schematics
4. Layout verification
5. Post-layout simulation: Simulation with parasitic components
6. (Place on frame)

Design of CMOS inverter

- Topology verification
- What is the performance of CMOS inverter?
- Elmore delay model
 - Input / Output conditions
- Delay time simulation
- meas command
- Model corner simulation

CMOS inverter, NOT gate



Input A	Output Q
L	H
H	L

↓

Input A	Output Q
Vss	Vdd
Vdd	Vss

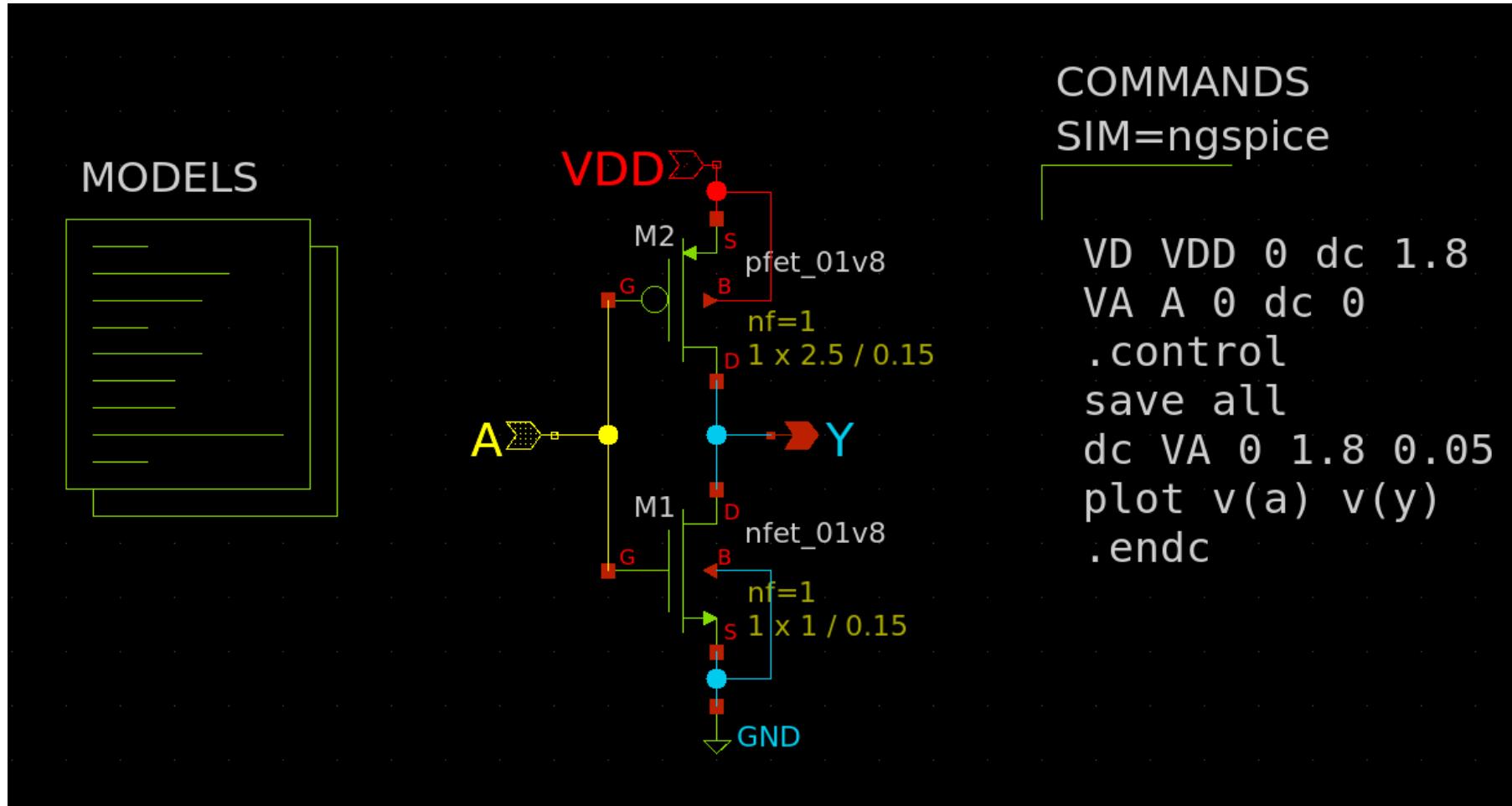
https://commons.wikimedia.org/wiki/File:CMOS_Inverter.svg

Logic verification (Topology verification)

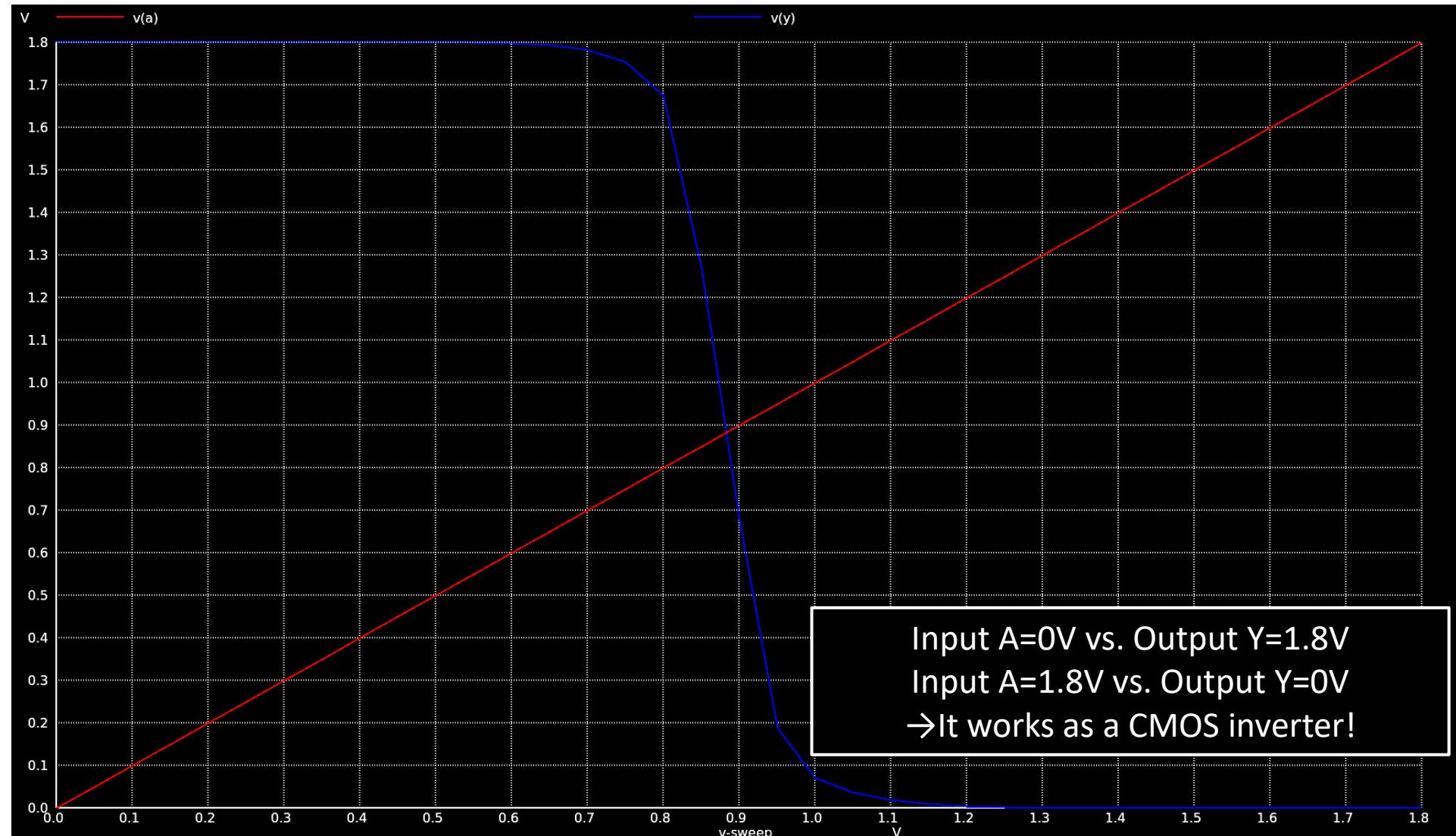
- In a CMOS inverter, the output should be inverted with the input
 - Input = 0 V → Output = Vdd V, Input = Vdd V → Output = 0 V
 - See how a CMOS inverter works with DC analysis
- DC analysis Steady-state response when voltage is varied
- tran analysis Transient response when time is varied
- AC analysis Steady-state response when frequency is varied

Example of test bench ($L=0.15\mu\text{m}$)

inv_dc.sch



Simulation results



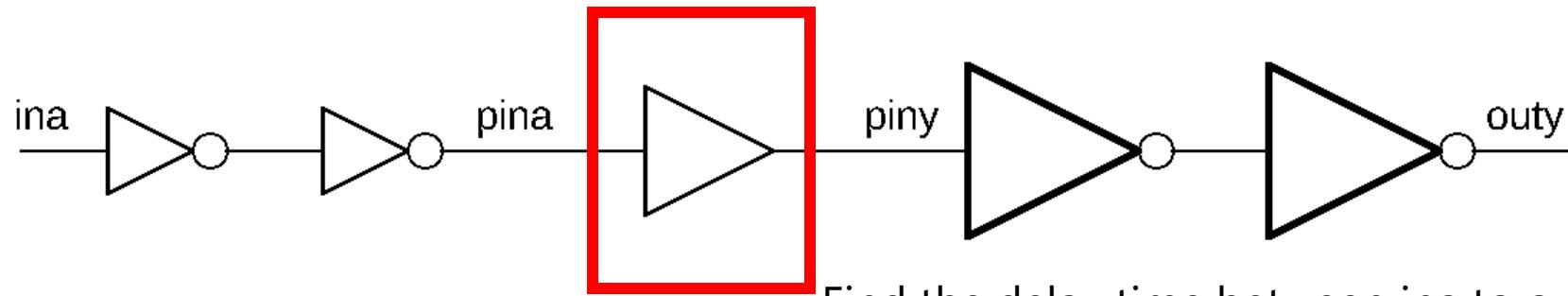
What is the performance of logic gate, CMOS inverter?

- Topology has been determined but the size of CMOS have not yet been determined.
→ How to determine?
- Performance of logic gate
 - Drivability (output current)
 - fan-out (F.O.)
 - **delay time** (rise delay / fall delay)

Fanout: Inverter/buffer delay

Thinking about fan-out

$$P=2.5\mu / N=1\mu$$



$$P=25\mu / N=10\mu$$

Find the delay time between *ina* to *outy*

- Load of input stage buffer: $w_p = 2.5 \mu\text{m}$, $w_n = 1 \mu\text{m}$
- Load of output stage buffer is 10 times of input load
- Place a buffer (2 inverters x 2) in the middle. **Consider the fan-out of this buffer.**
- Aim to minimize the delay time from *ina* to *outy*.

Delay time of buffer

Simulate the 6-stage inverter chain

Do the transient analysis because we want to see the time vs. voltage waveforms.

Format of transient analysis

tran [time step] [analysis time]

Example of testbench ($L=0.15\mu\text{m}$)

6inv_test.sch

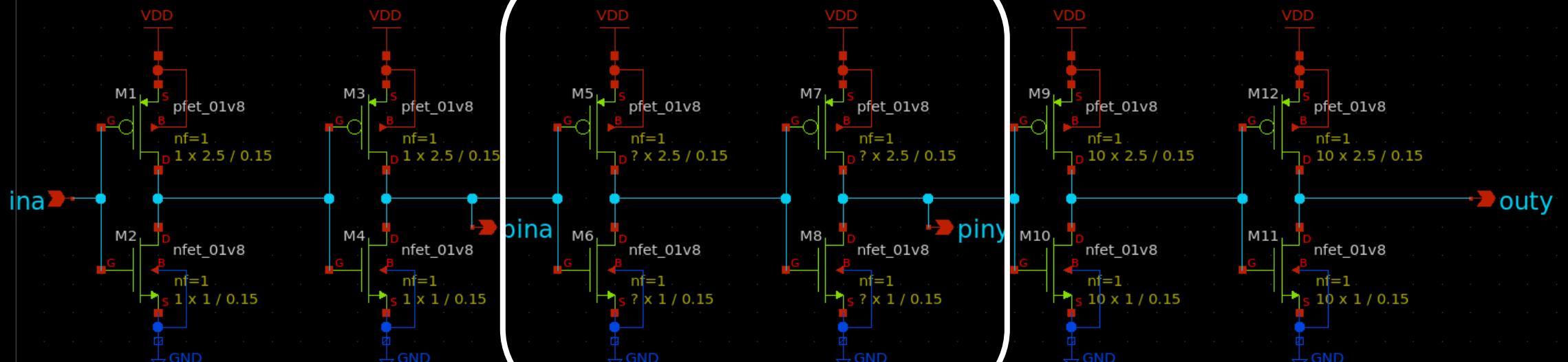
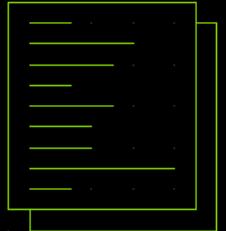
COMMANDS

SIM=ngspice

```
VA ina 0 pulse (0 1.8 0 40p 40p 0.5n 1n) dc 0  
VD VDD 0 dc 1.8  
.control  
tran 1p 2n  
plot v(ina) v(outy) v(outy)  
write 6inv_test.raw  
.endc
```

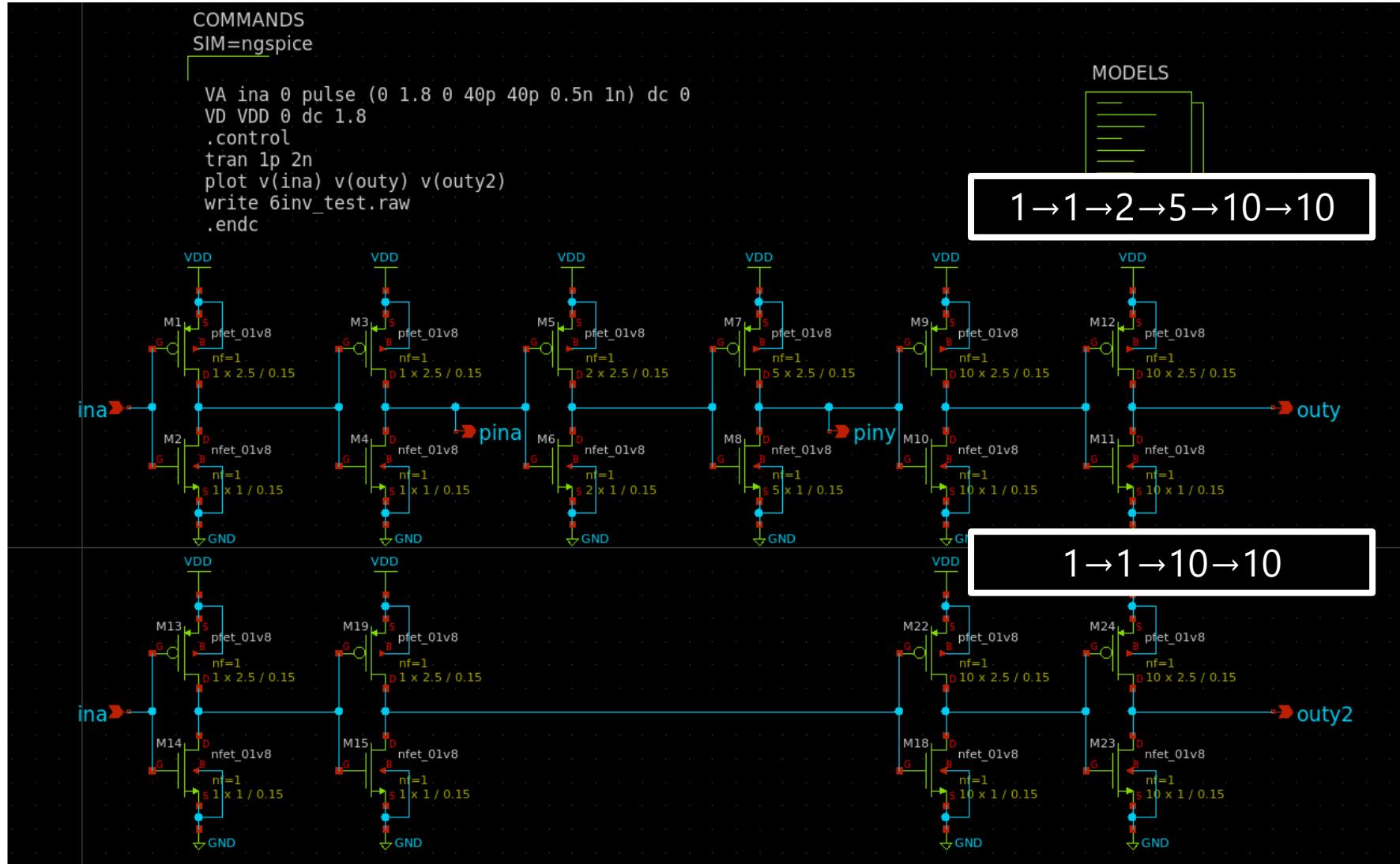
How do you determine
the fan-out here?

MODELS

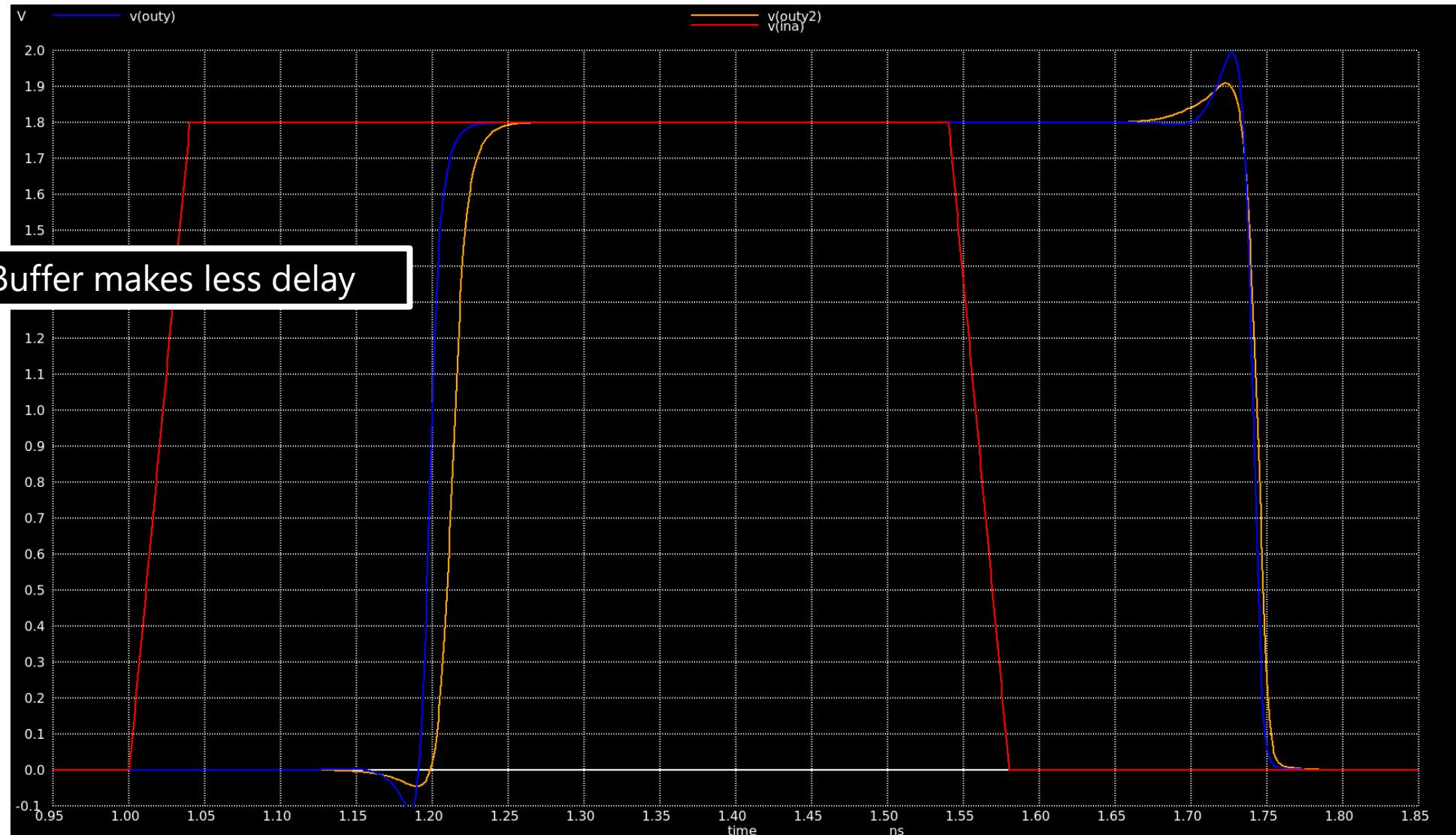


Example of testbench ($L=0.15\mu\text{m}$)

6inv_test2.sch



Simulation results



Note: fan-out of buffers in standard cells

buf_2 1→2

buf_4 1→4

buf_8 3→8

clkbuf8 2→8

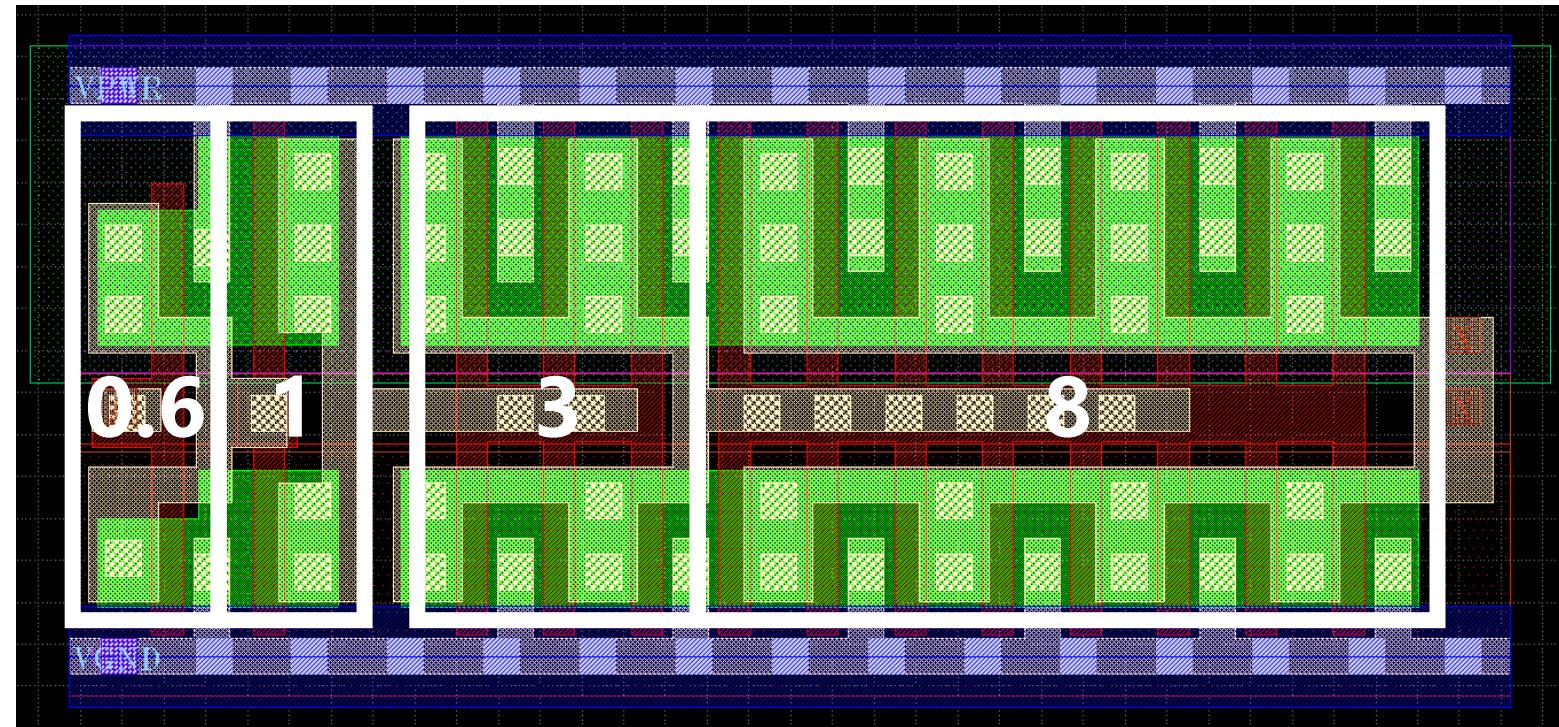
bufbuf8 **0.6→1→3→8**

buf_12 4→12

buf_16 6→16

clkbuf16 4→16

bufbuf16 **1→3→6→16**

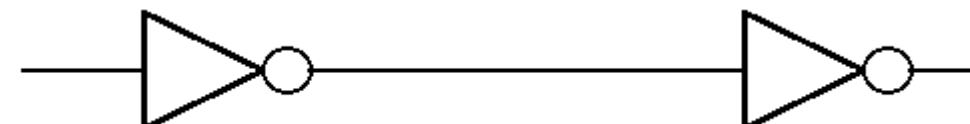


If more than 3 - 4 times, insert additional buffer(s)

What is delay time?

- Delay time = (dis)charge time of parasitic capacitance
(Input gate capacitance + Output drain capacitance vs. Output current)
RC time constant of 0.75 and a fixed resistance value
- In the Elmore delay model, the delay time is calculated by the following approximate formula
- $t_{PHL} = 0.75 \times (C_{dd.n} + C_{dd.p} + C_{gg.n} + C_{gg.p})R_n$
Input rise, output fall delay
 C_{dd} = drain capacitance on output side
 C_{gg} = gate capacitance on input side
 R = on-resistance on output side
- $t_{PLH} = 0.75 \times (C_{dd.n} + C_{dd.p} + C_{gg.n} + C_{gg.p})R_p$
Input fall, output rise delay

If $R_n = R_p$ do the rise and fall delays match?



C_{dd}, C_{gg} can be simulated by op analysis

You may find the following formula on the internet

MOSFET Input capacitance (C_{iss}): C_{iss} = C_{gs} + C_{gd}

In LSI CMOS, gate-to-body capacitance must be considered too.

$$C_{gg} = |C_{gs} + C_{gd} + C_{gb}|$$

- Gate capacitance is represented by the parameter C_{gg}
- Output capacitance = Drain capacitance C_{dd} = |C_{dg} + C_{ds} + C_{db}|
 - C_{gd} and C_{dg} both represent drain-gate parasitic capacitance.
C_{dg} is the parasitic capacitance that is affected when the **drain voltage changes**.
C_{gd} is the parasitic capacitance that is affected when the **gate voltage changes**.
Usually, C_{dg} ≠ C_{gd}

To find the input / output capacitance,
see C_{gg} / C_{dd} in op analysis

Transient response, delay time

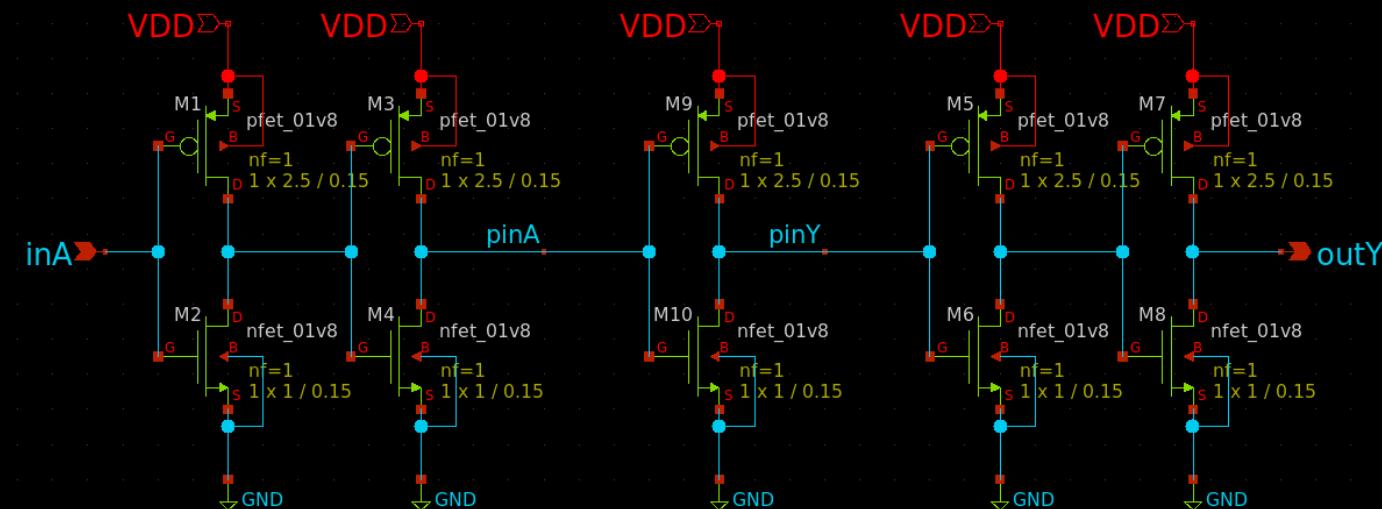
- Use transient (tran) analysis if you want to find the delay time
- **To obtain the correct transient response (delay), set the loads on the input and output stages.**
(load = logic gate; usually, CMOS buffer)
- Assuming we design a CMOS inverter with $FO=1$, we connect buffers of the same size.

Example of test bench

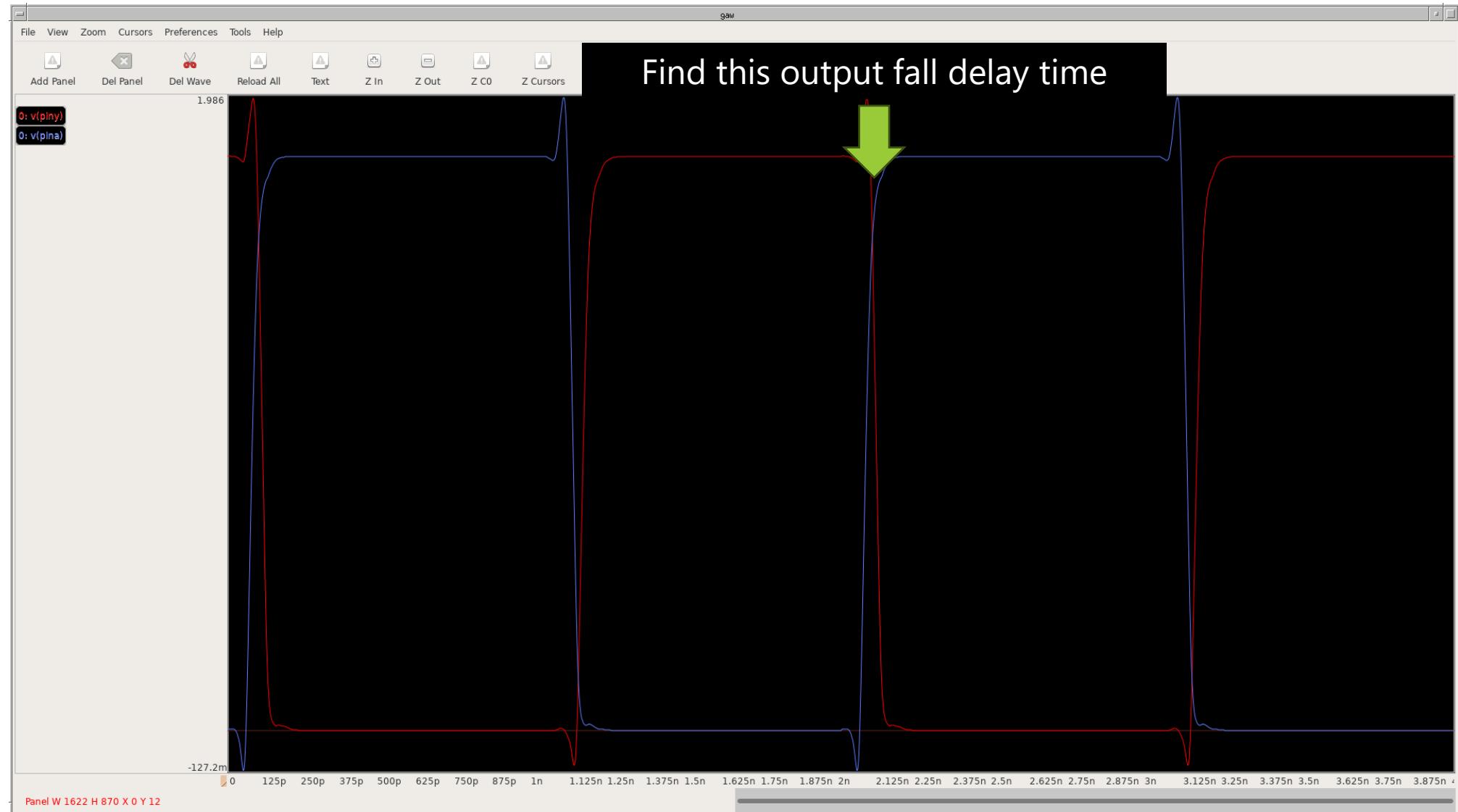
inv_meas.sch

```
ngspice
VA inA 0 pulse(0 1.8 0 40p 40p 1n 2n) dc 0
VD VDD 0 dc 1.8
.control
tran 1p 4n
meas tran delayA1 find time WHEN v(pinA)=0.9 RISE=2
meas tran delayY1 find time WHEN v(pinY)=0.9 FALL=2
let delay1 = delayY1 - delayA1
echo $&delay1
.endc
```

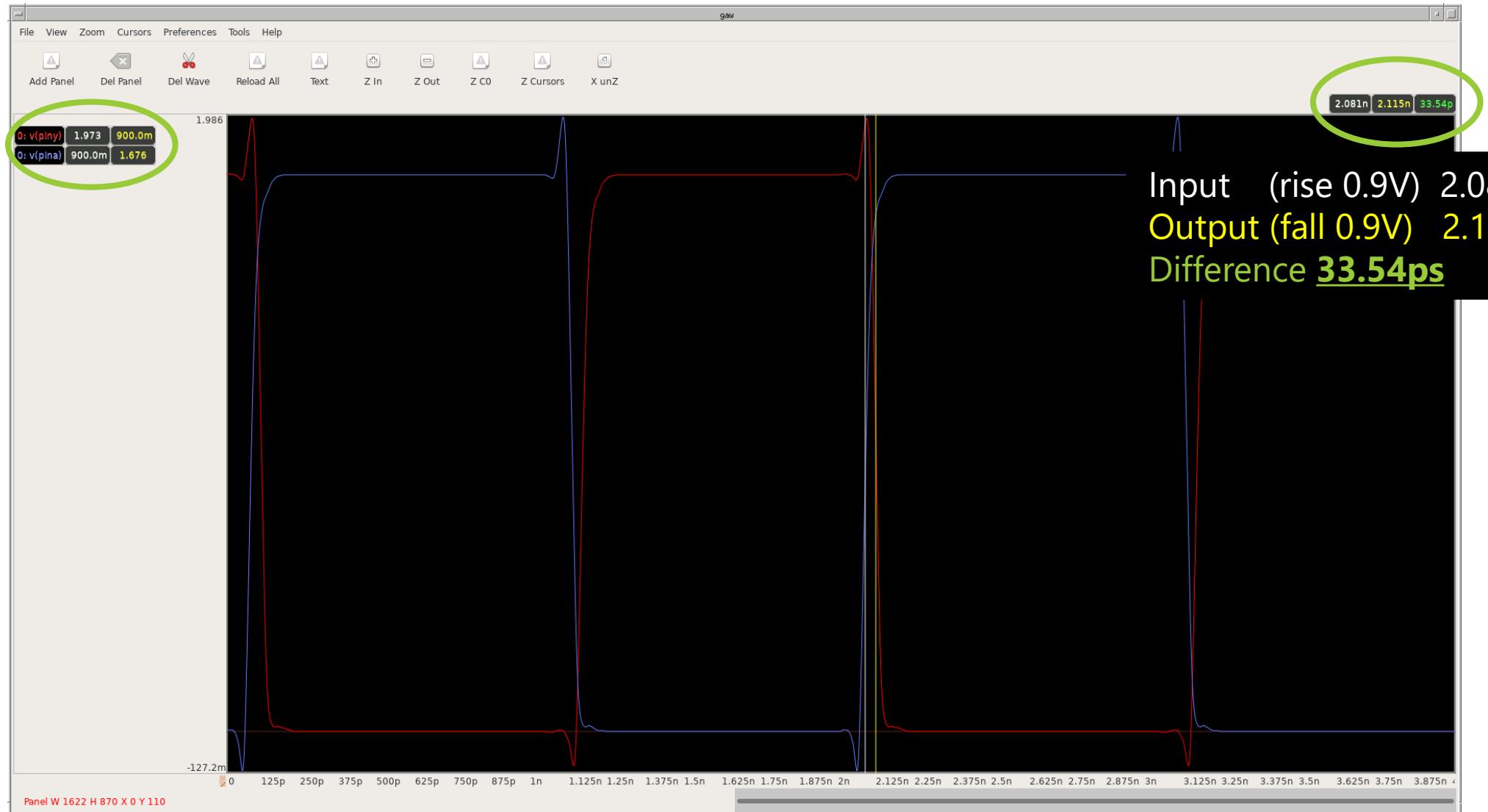
MODELS



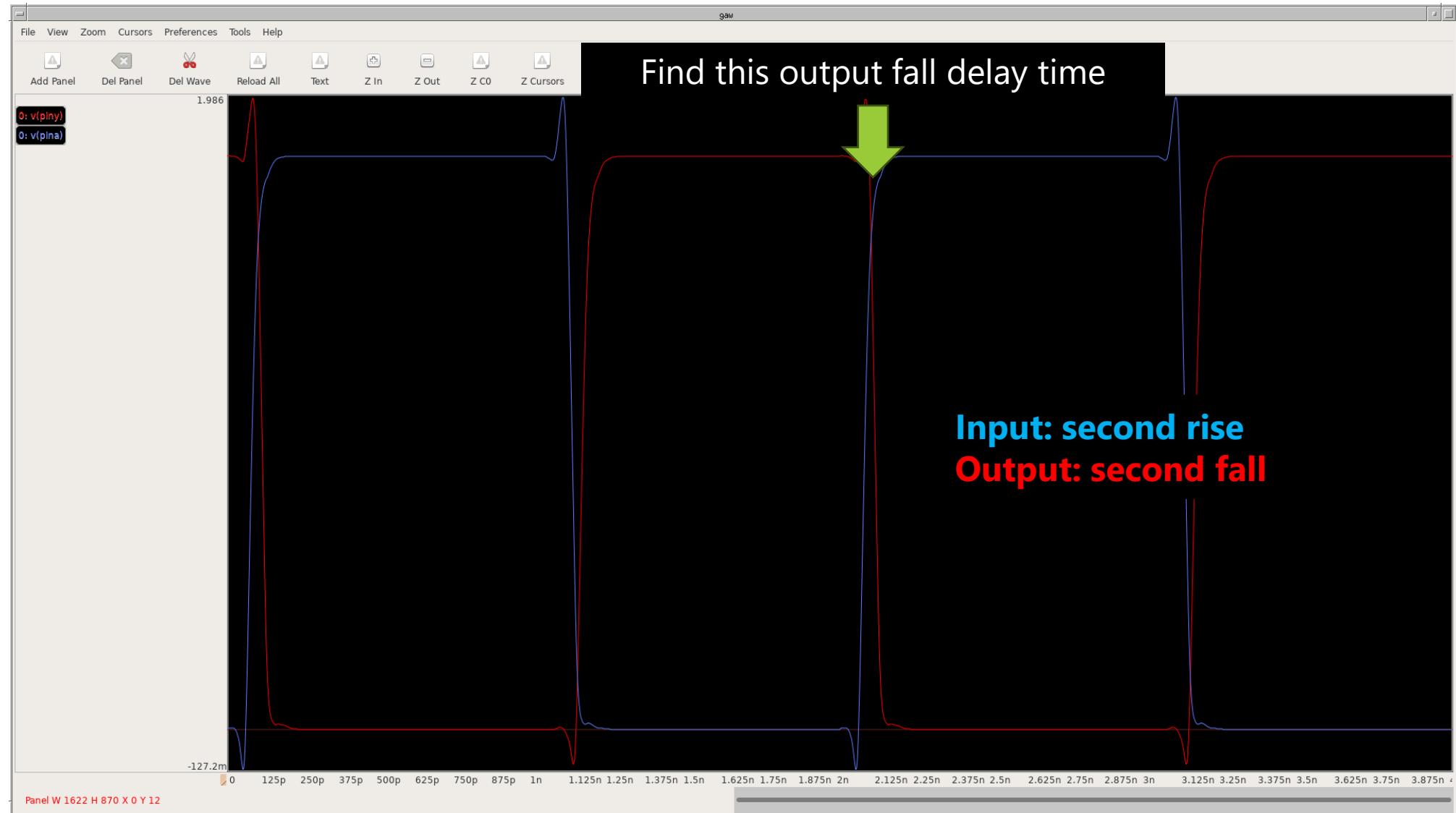
Measuring delay time with waveform viewer



Measuring delay time with waveform viewer



Measuring delay time with commands



meas command

Calculating delay time from waveforms and graphs is bothersome → meas

This command is multifunctional, but this time we will focus on delay time calculation.

Format of meas command

meas tran [variable] FIND time WHEN [condition] [command]

- [variable] – to assign the result to
- [condition] – “ $v(\text{net}) = Vdd/2$ ”
- [command] – “[RISE / FALL / CROSS] = [number of count]”

meas command

Calculating delay time from waveforms and graphs is bothersome → meas

This command is multifunctional, but this time we will focus on delay time calculation.

meas command example

meas tran **delay1** FIND time WHEN v(vin)=0.9 RISE=2

Assign the time to **delay1** when v(vin) is 0.9V at the second rise

meas tran **delay2** FIND time WHEN v(vout)=0.9 FALL=2

Assign the time to **delay2** when v(vin) is 0.9V at the second fall

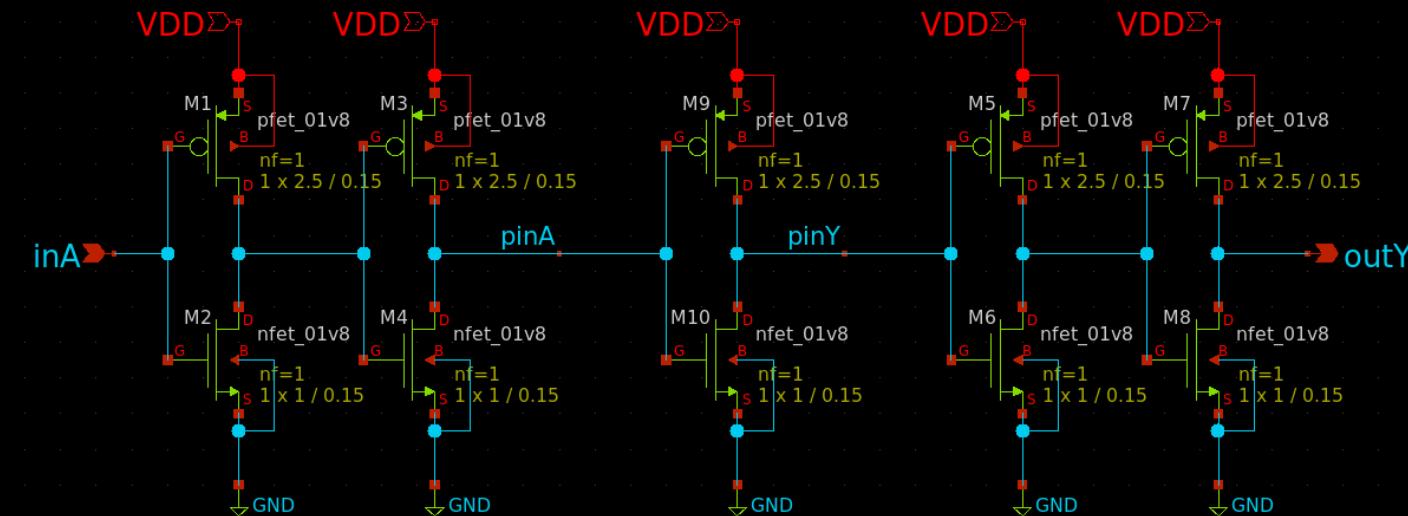
Find the fall delay time with meas command

inv_meas.sch

```
ngspice
VA inA 0 pulse(0 1.8 0 40p 40p 1n 2n) dc 0
VD VDD 0 dc 1.8
.control
tran 1p 4n
meas tran delayA1 find time WHEN v(pinA)=0.9 RISE=2
meas tran delayY1 find time WHEN v(pinY)=0.9 FALL=2
let delay1 = delayY1 - delayA1
echo $&delay1
.endc
```

MODELS

Input: second rise
Output: second fall



Find the fall delay time with meas command

inv_meas.sch

```
ngspice
VA inA 0 pulse(0 1.8 0 40p 40p 1n 2n) dc 0
VD VDD 0 dc 1.8
.control
tran 1p 4n
meas tran delayA1 find time WHEN v(pinA)=0.9 RISE=2
meas tran delayY1 find time WHEN v(pinY)=0.9 FALL=2
let delay1 = delayY1 - delayA1
echo $&delay1
.endc
```



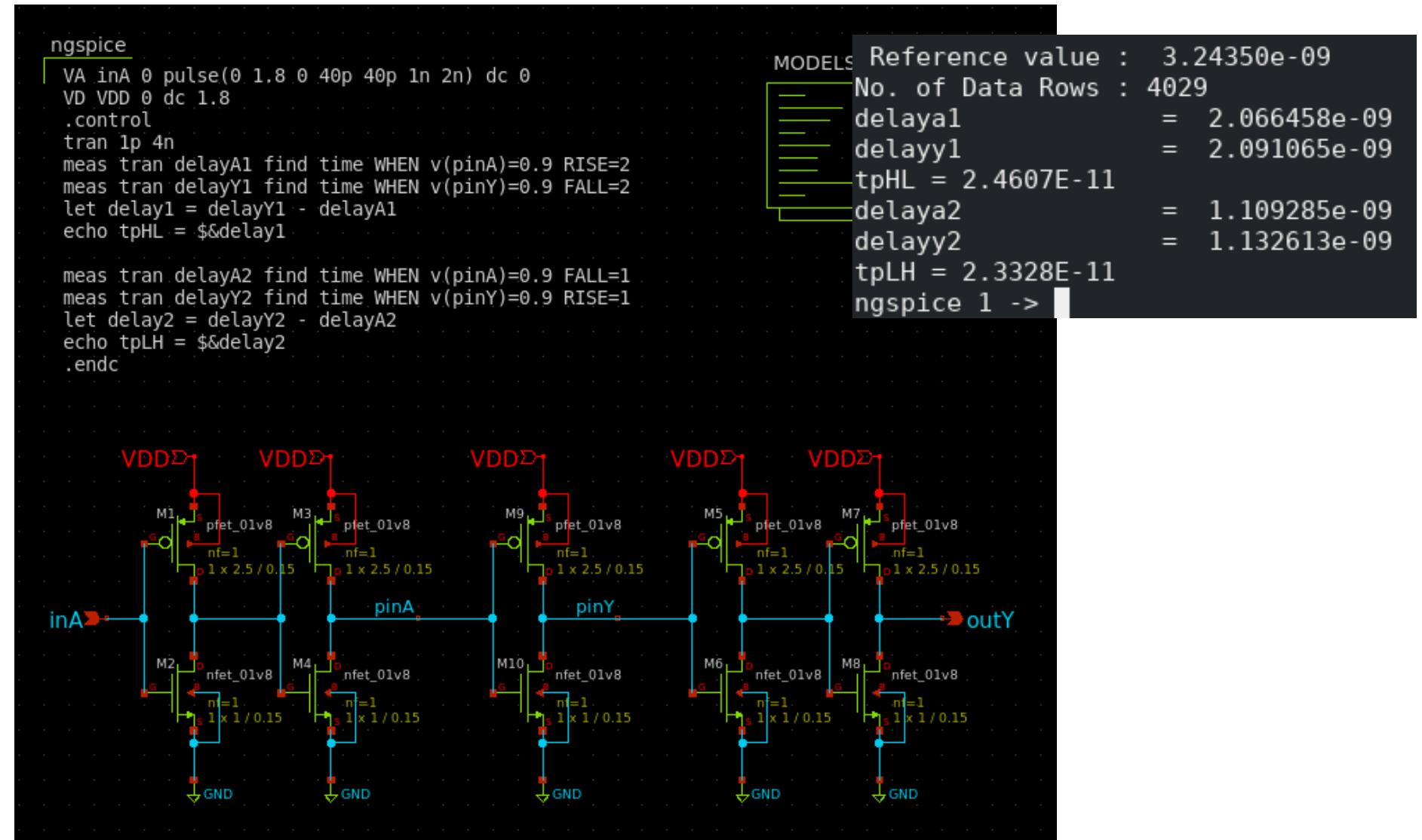
Input: second rise
Output: second fall

```
Reference value : 2.18050e-09
No. of Data Rows : 4029
delayA1           = 2.066458e-09
delayY1           = 2.091065e-09
2.4607E-11 ← Delay time is automatically calculated.
ngspice 1 ->
```



Also find the rise delay time in the same way

inv_meas.sch



Rise times and fall times

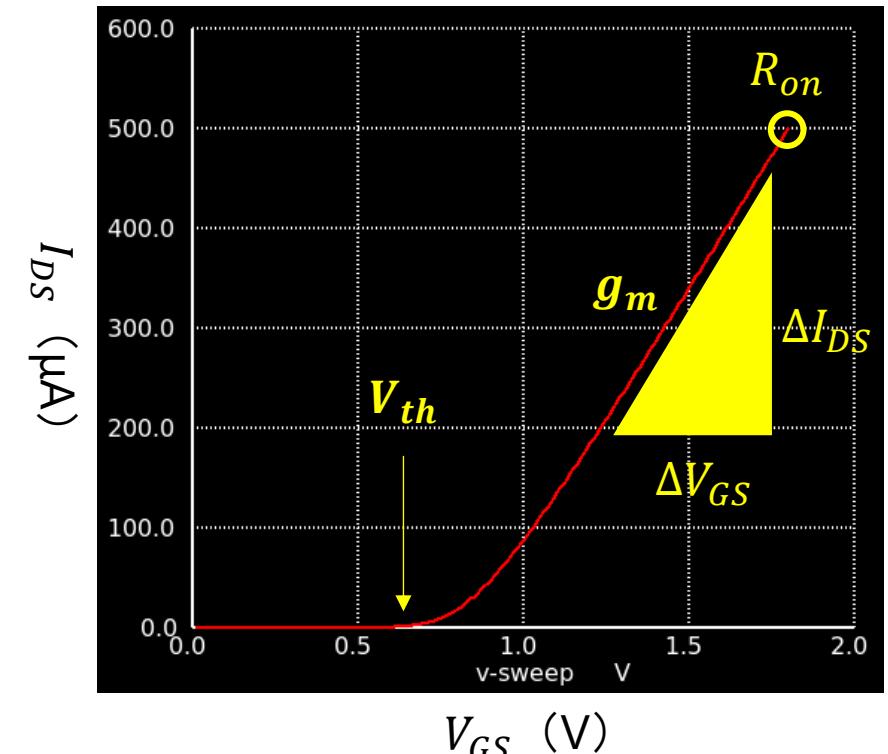
Fixed to N=1u, changed W of PMOS including buffers of input and output stages

	2.0u	2.1u	2.2u	2.3u
t_{PHL}	23.238ps	23.496ps	23.764ps	24.035ps
t_{PLH}	23.889ps	23.725ps	23.591ps	23.482ps

- For N=1u, P=2.1u or 2.2u seems better

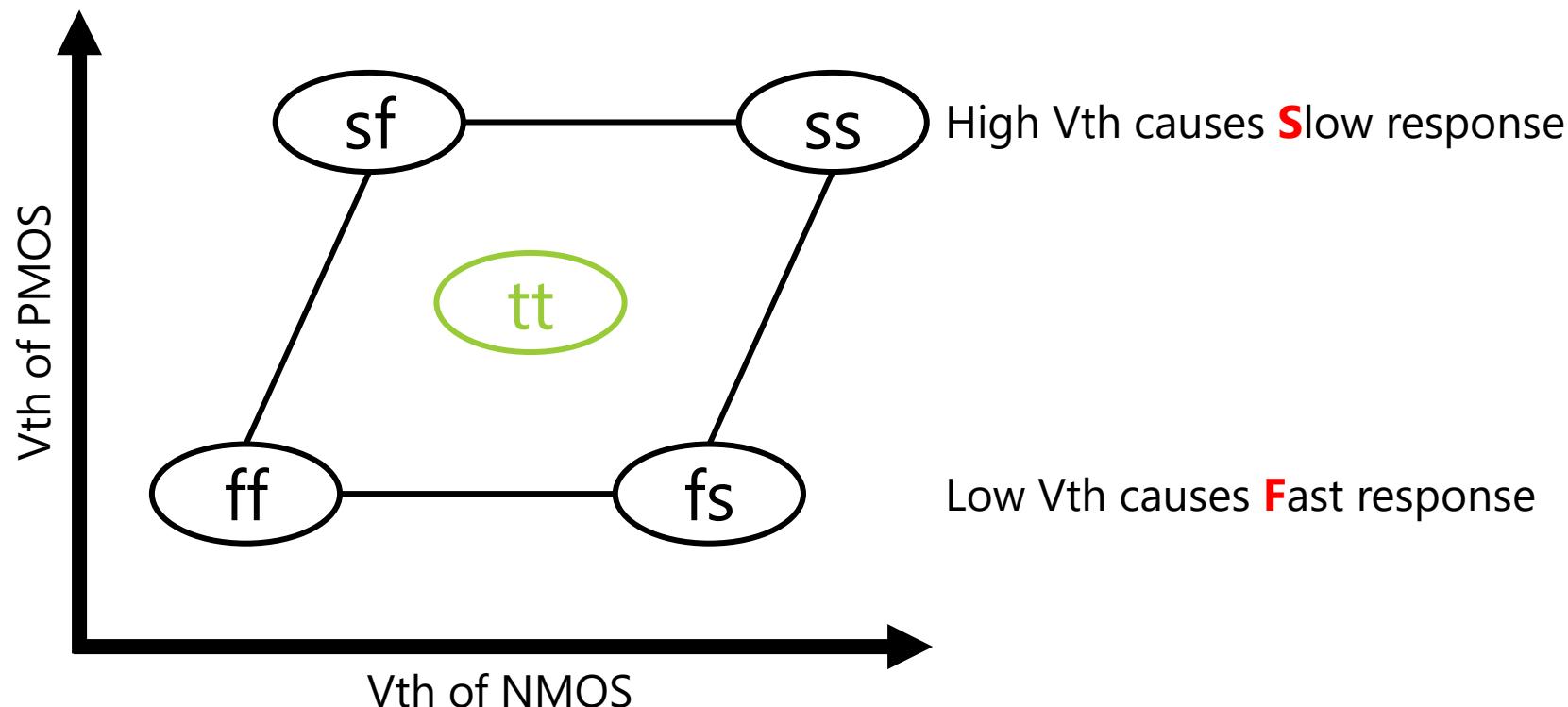
Rise and fall delays do not match at $R_n = R_p$

- V_{th} and g_m in PMOS and NMOS do not match.
- The resistance of the MOSFET (R_{ds}) varies with V_{gs} .

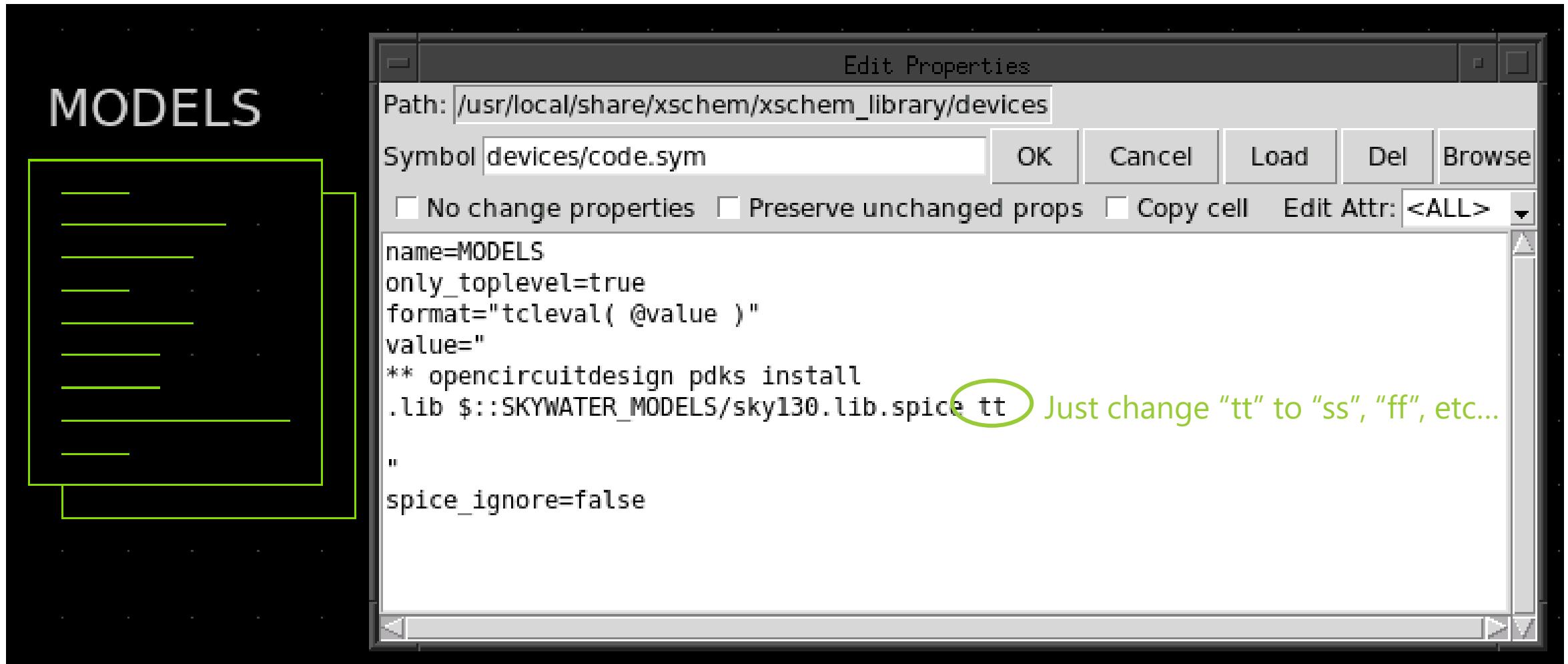


Corner of model

- ss – Model for slow responsiveness (slow-slow)
- tt – Model for typical responsiveness (typical-typical)



Switch of model corner



Rise times and fall times

- Simulation with different corner model (ss) changes the results.

	2.4u	2.5u	2.6u	2.7u
t_{PHL}	33.242ps	33.538ps	33.856ps	34.191ps
t_{PLH}	33.845ps	33.754ps	33.724ps	33.750ps

- For N=1u, P=2.5u or 2.6u seems better.

Rise and fall delays do not match at $R_n = R_p$ (On-resistance ratio is P:N = 3:1 in ss)

The resistance of the MOSFET varies with Vgs.

Background knowledge of LSI

Layout

Design flow of Analog LSI

1. Draw schematics and test benches
2. Simulation
- 3. Draw layouts based on schematics**
4. Layout verification
5. Post-layout simulation: Simulation with parasitic components
6. (Place on frame)

Planar MOSFET transistor

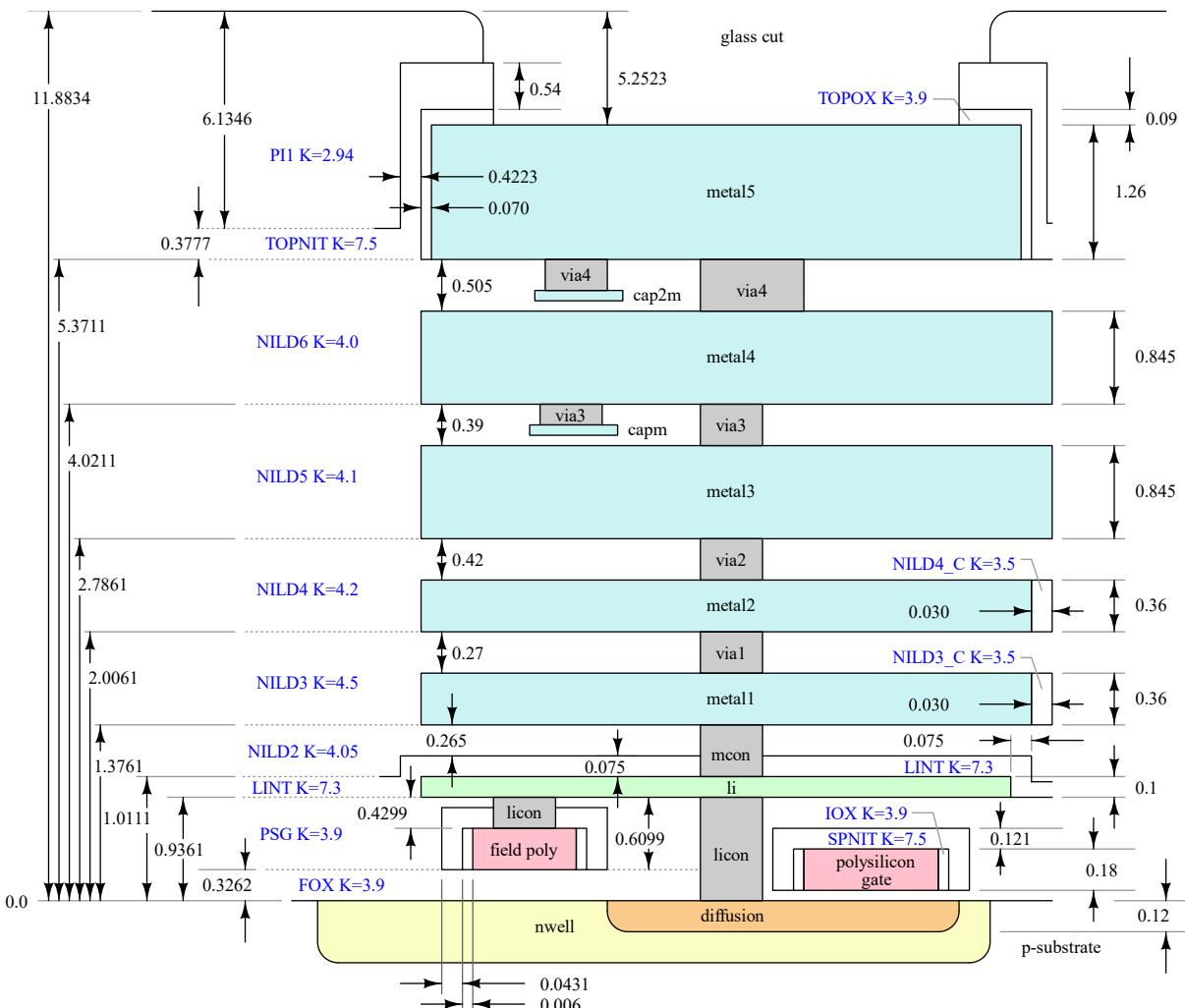
Substrate is stacked as the bottom layer.

MOSFET on substrate (well, diffusion, poly)

Wiring above the transistor(metal*, via*)

MIMCAP can be made by sandwiching
an insulator between metals
(metal3-capm and metal4-cap2m)

(Diagram not to scale!)

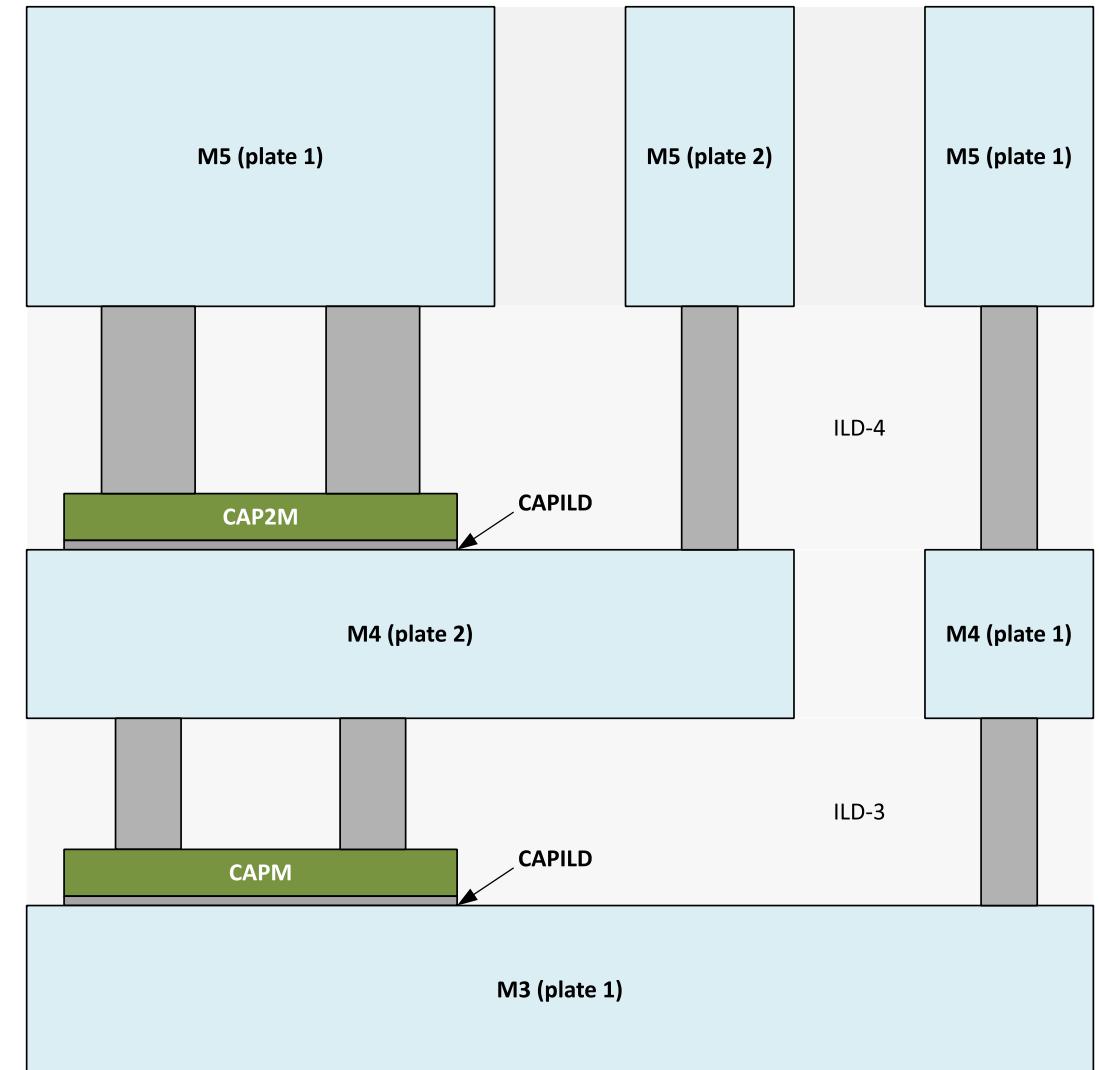
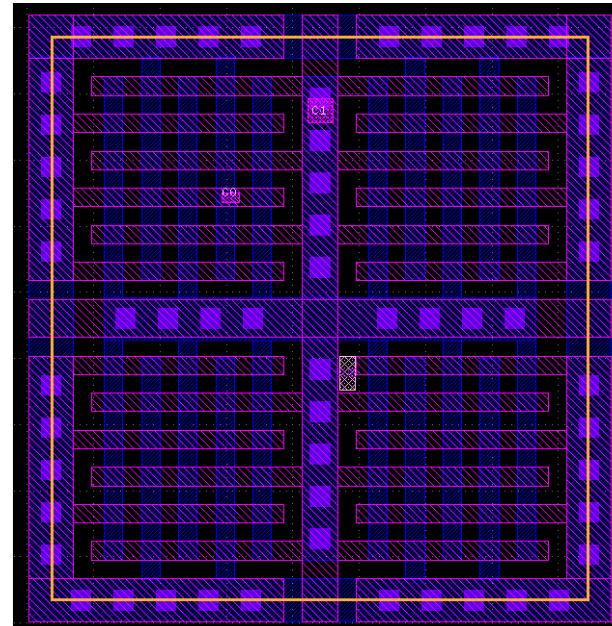


MIMCAP

2 MIMCAPs exist between M5-M4 and M4-M3

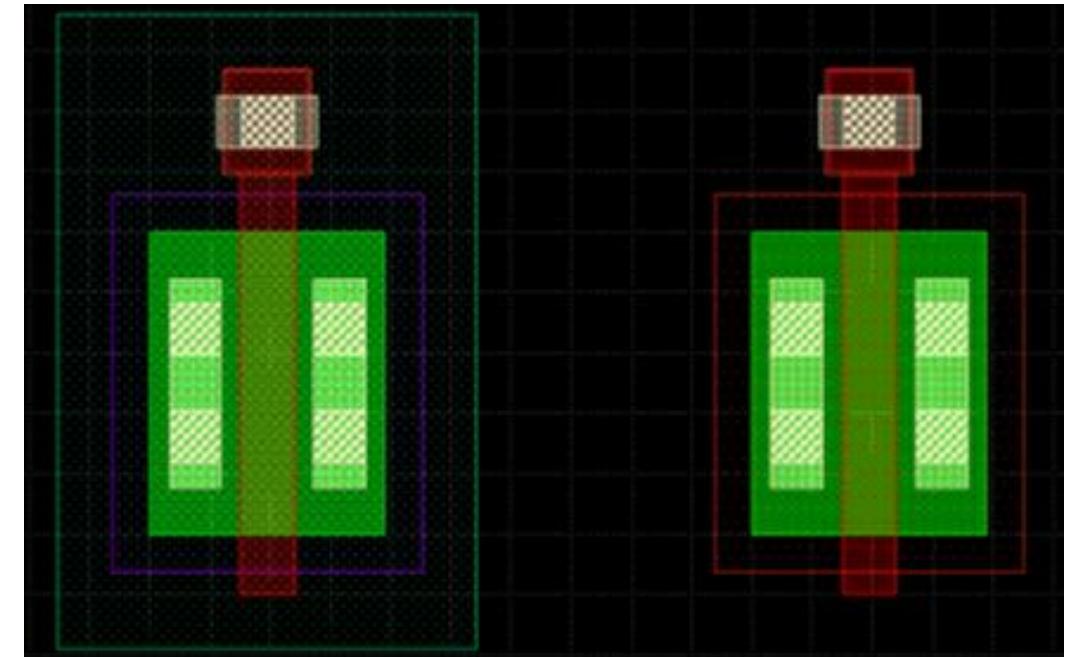
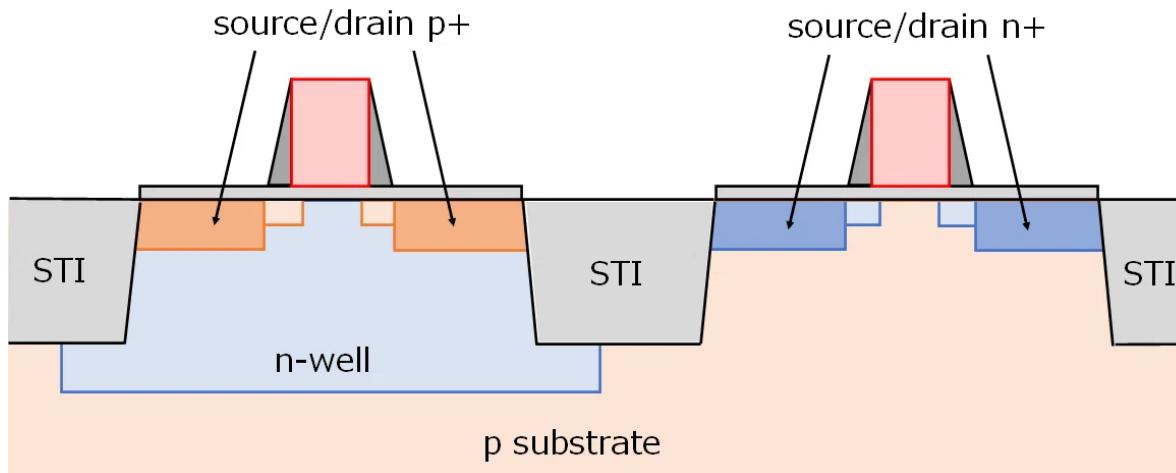
Crossed wiring layers also make a capacitor, but small capacitance. (so-called MOMCAP)
→Cause of parasitic capacitance

Capacitor Consisting only
of Metal1 and Metal2



Double well, Twin well, Dual well

CMOS transistors are composed of N-well and P-substrate
Pcell generates double-well CMOS by default.

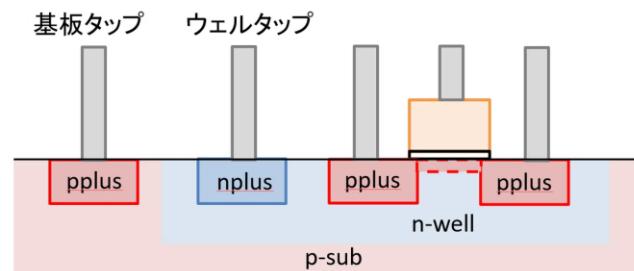


https://note.com/akira_tsuchiya/n/n416b7f74b701

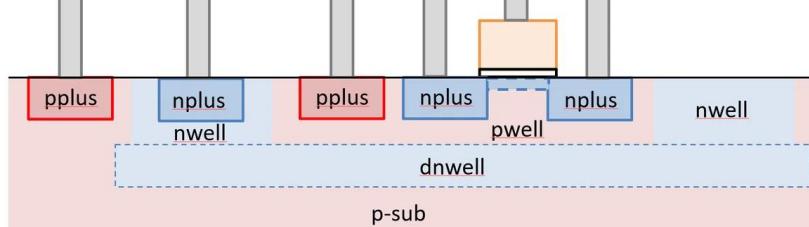
PMOS has an individual well so body voltage is variable
Body of all double well NMOS are connected via substrate
(Lowest voltage in the LSI chip, GND, VSS)

Triple well

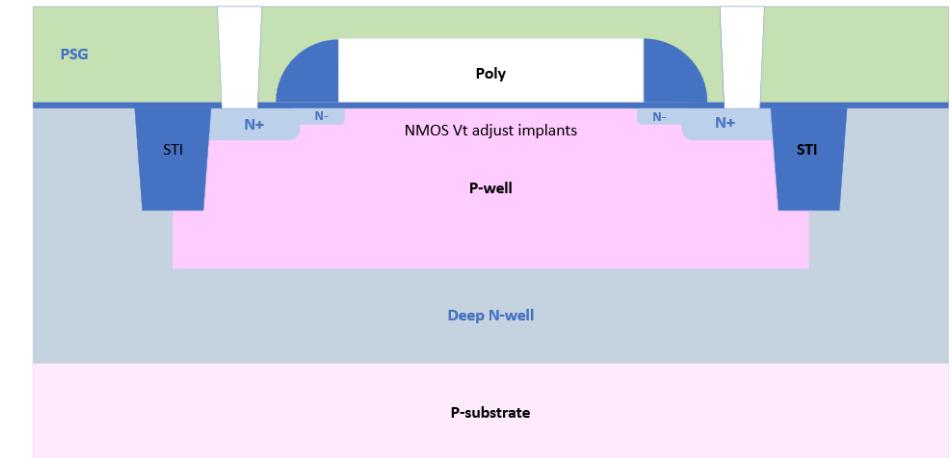
CMOS transistors are composed of Deep N-well, P-well and N-well
The cross section described in the PDK documents is triple well.



PMOS



NMOS



https://note.com/akira_tsuchiya/n/nd4444304f013

NMOS has an individual well so body voltage is variable
Example of applications

- Adjust Vth (**body effect**)
- Cascode connection with body connected to source
- Block noise from substrate

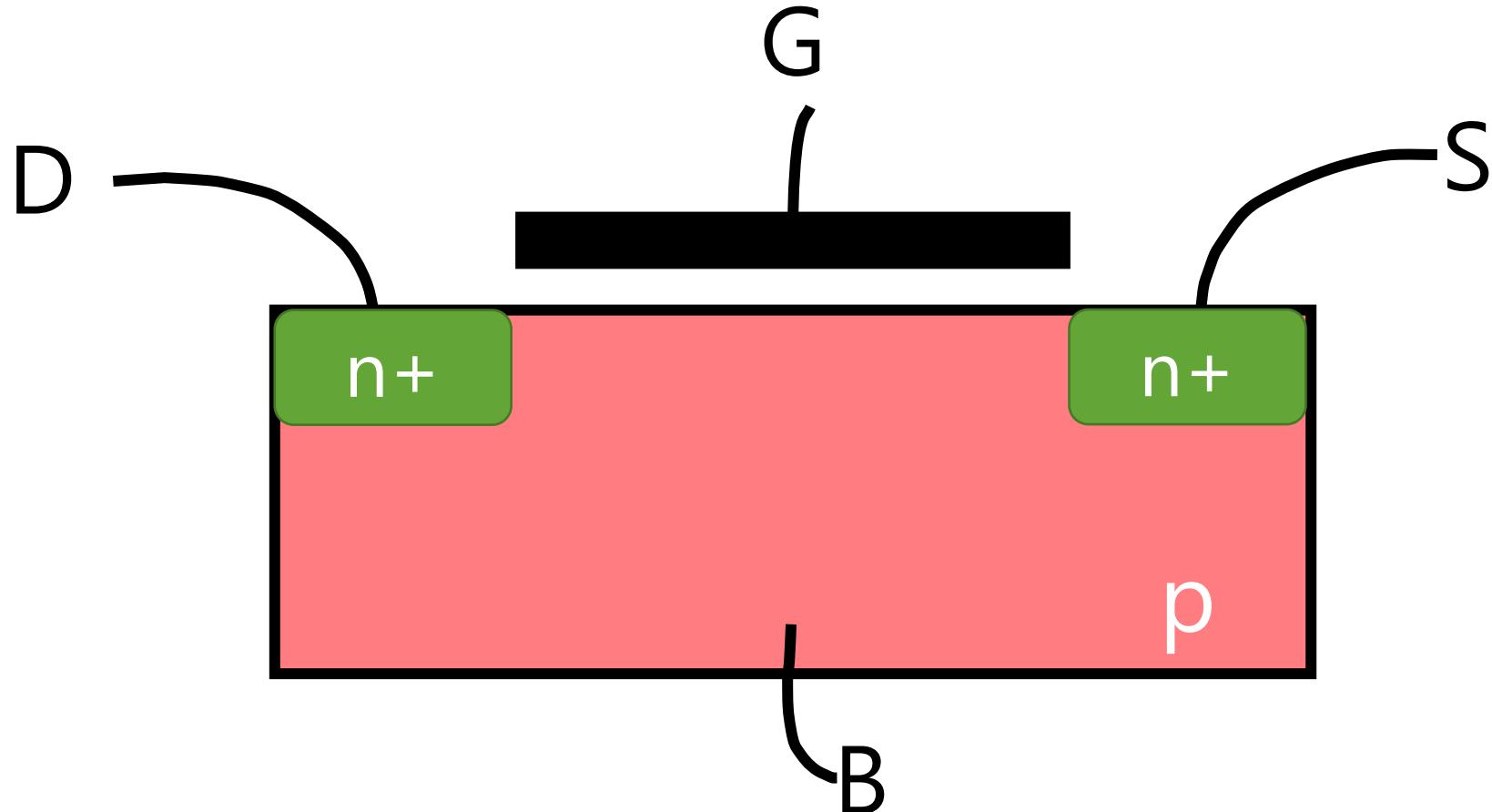
etc...

Body effect

Bias voltage applied to the body changes transistor characteristics.

- Threshold voltage (V_{th}) varies
- Drain-source current (I_{ds}) varies (when compared with fixed gate voltage)

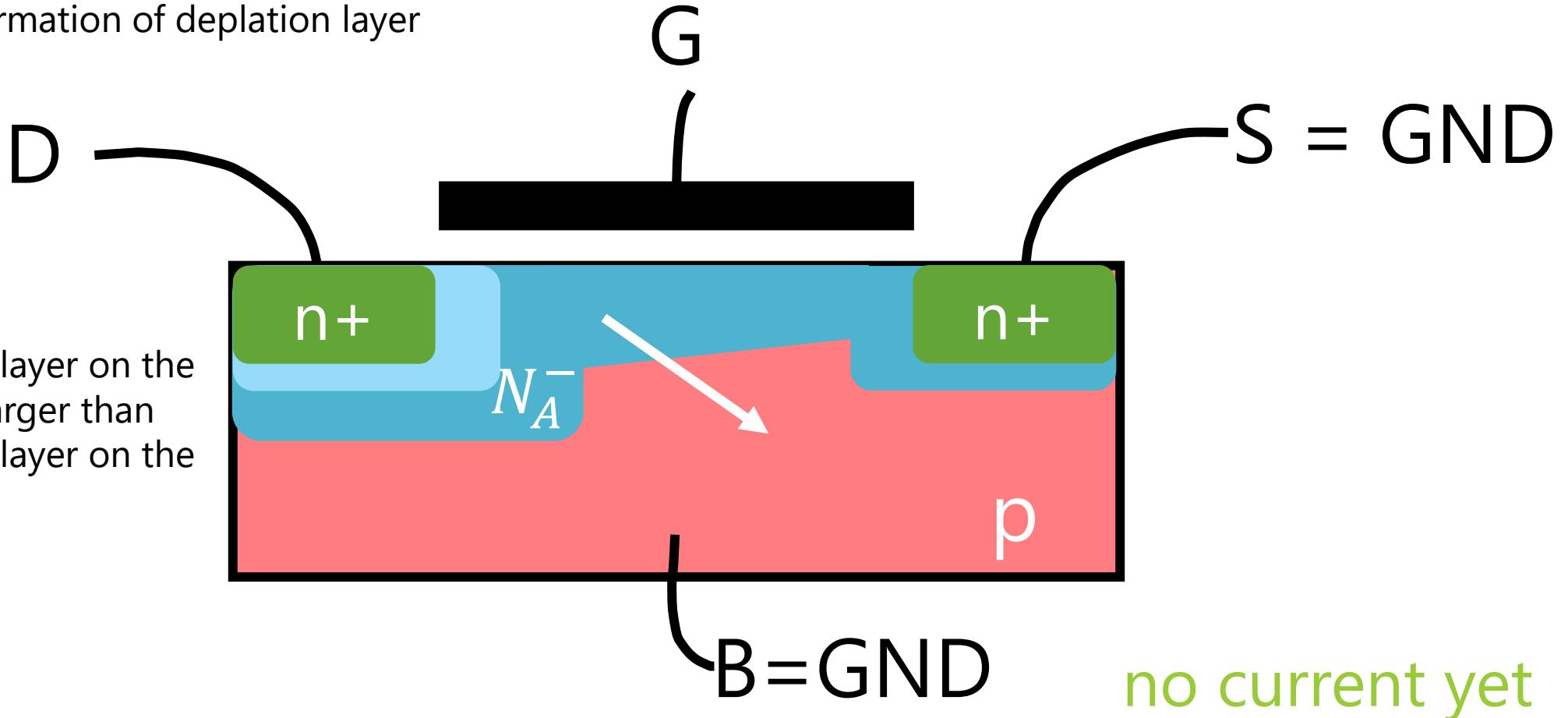
nMOS diagram



nMOS diagram

Hole on substrate moves to gate at $V_g > 0$
→ formation of depletion layer

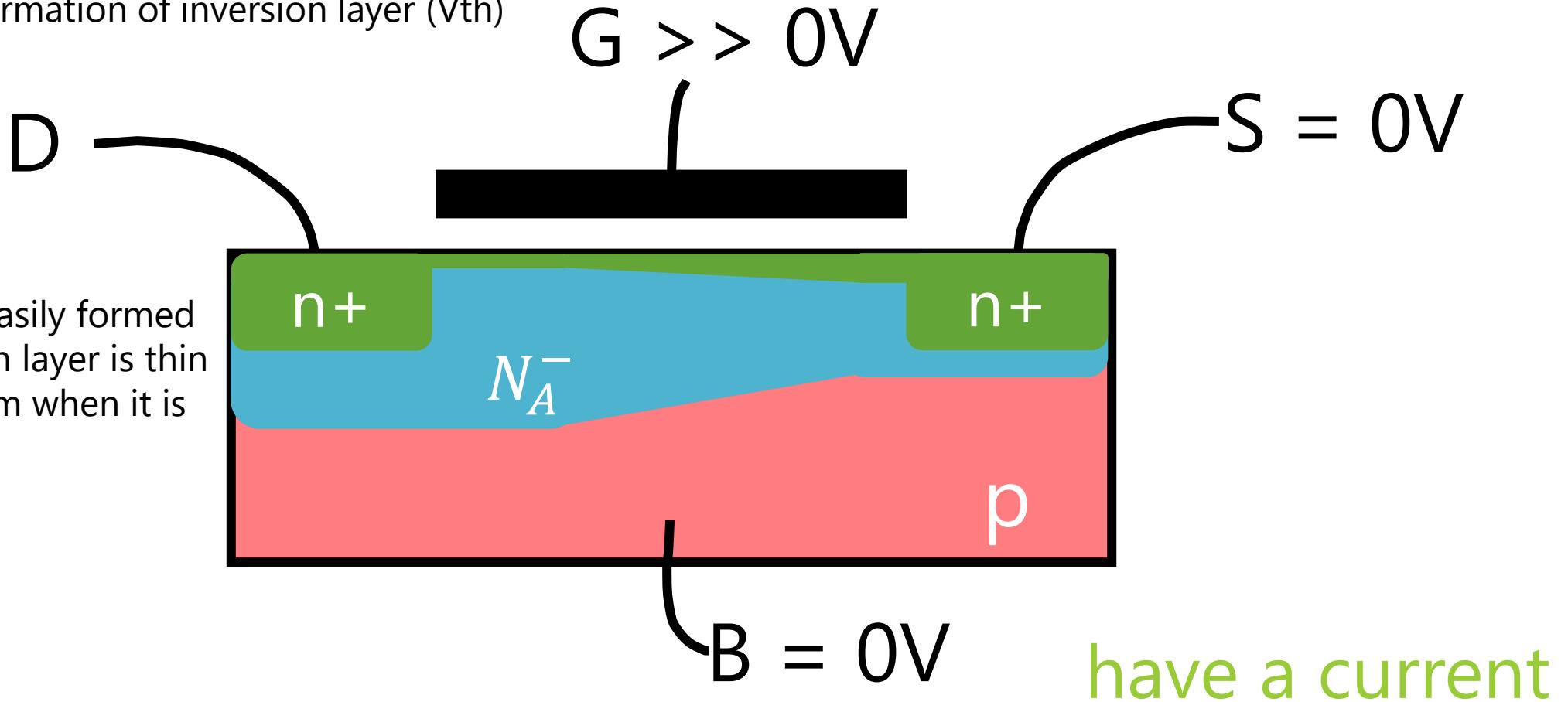
Since $V_d > V_s$,
the depletion layer on the
drain side is larger than
the depletion layer on the
source side



no current yet

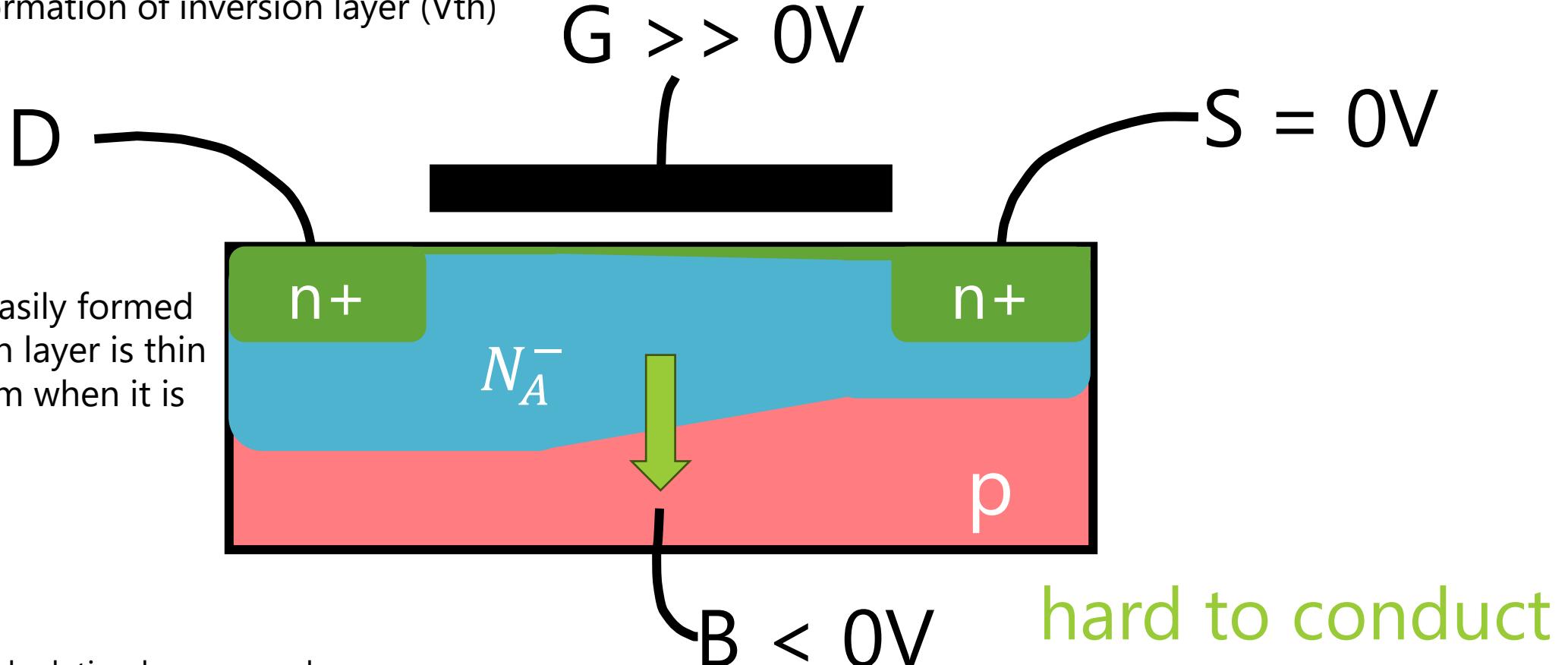
nMOS diagram

Conduction electrons become majority carriers at $V_g >> 0$
→ Formation of inversion layer (V_{th})



nMOS diagram

Conduction electrons become majority carriers at $V_g >> 0$
→ Formation of inversion layer (V_{th})

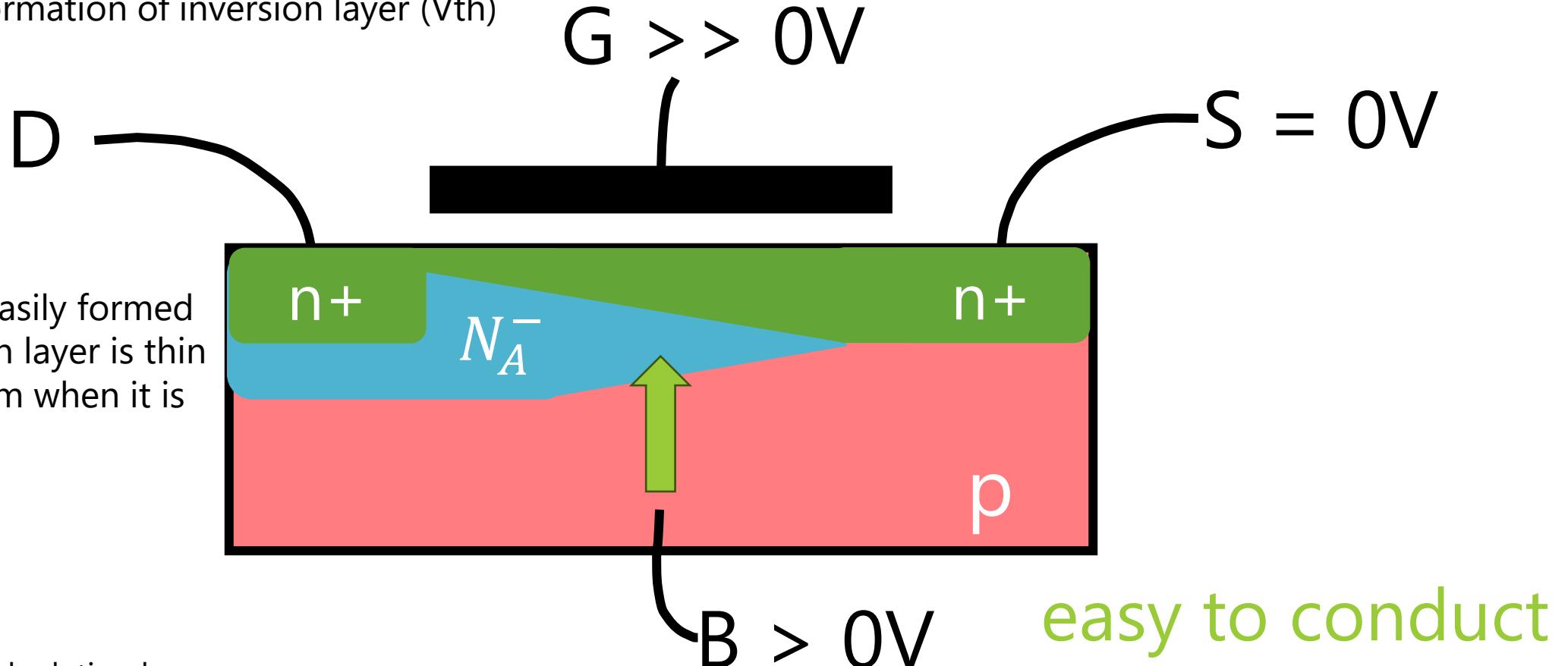


Inversion layer is easily formed
when the depletion layer is thin
and difficult to form when it is
thick

When $V_b < 0$, the depletion layer spreads
→ Inversion layer becomes harder to form (V_{th} increases) and thinner (on-resistance increases)

nMOS diagram

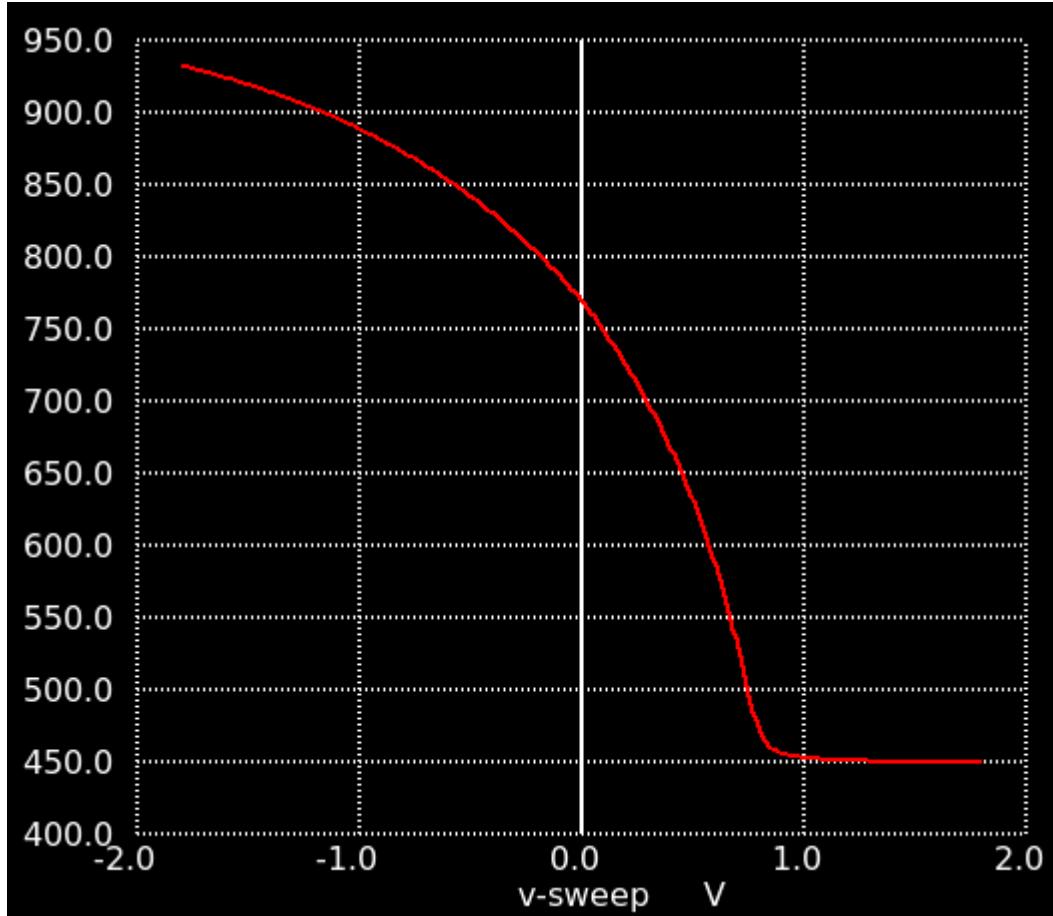
Conduction electrons become majority carriers at $V_g >> 0$
→ Formation of inversion layer (V_{th})



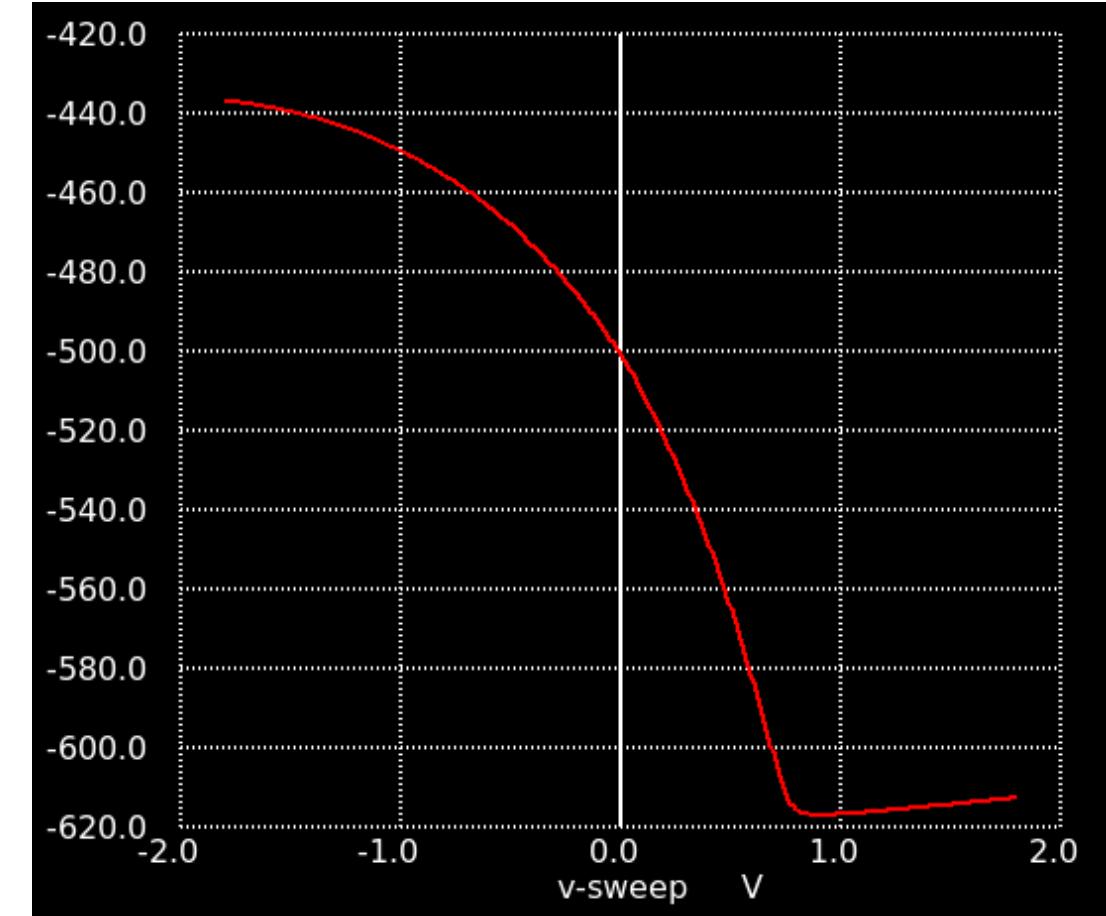
When $V_b > 0$, the depletion layer narrows
→ Inversion layer becomes easier to form (V_{th} decreases) and thicker (on-resistance decreases)

Body effect in nMOS

trbench_body.sch



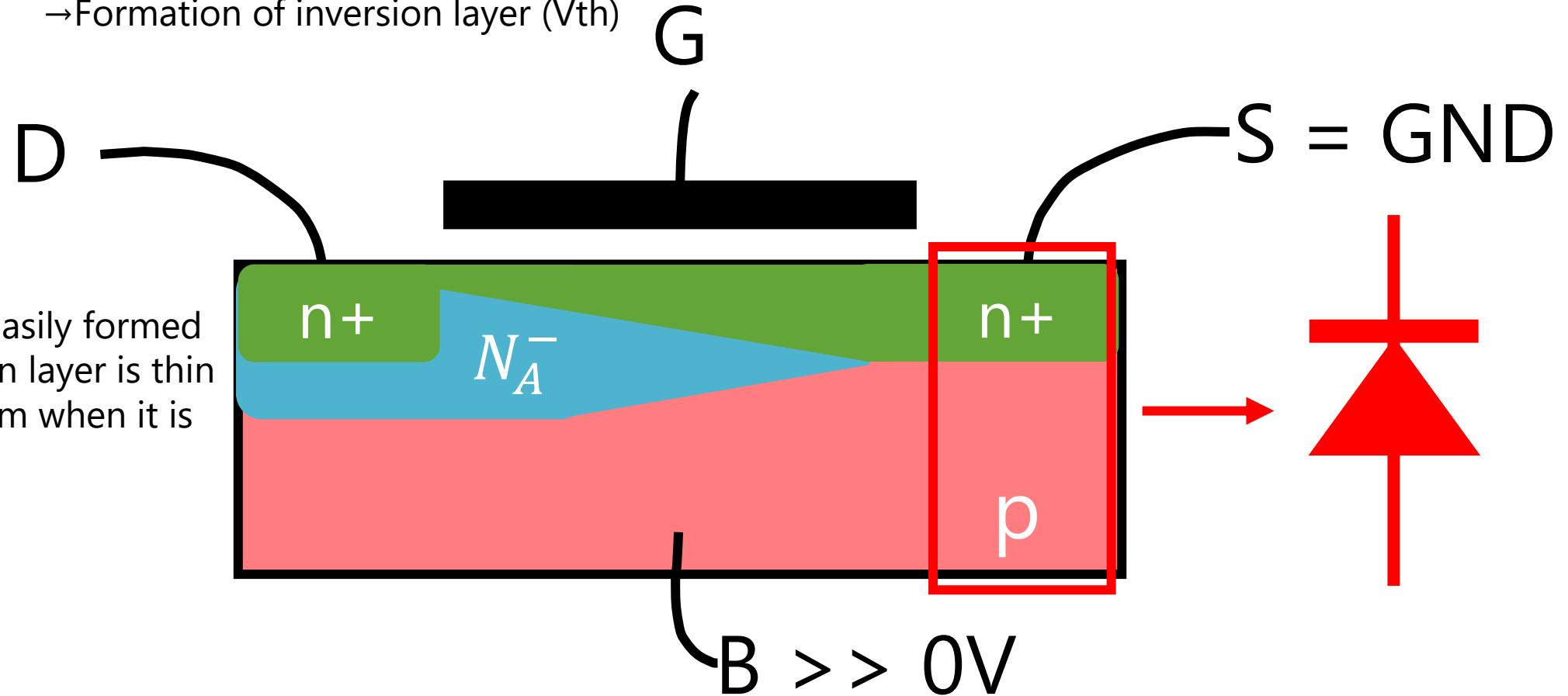
$$V_{BS} - V_{TH}$$



$$V_{BS} - I_{DS}$$

nMOS diagram

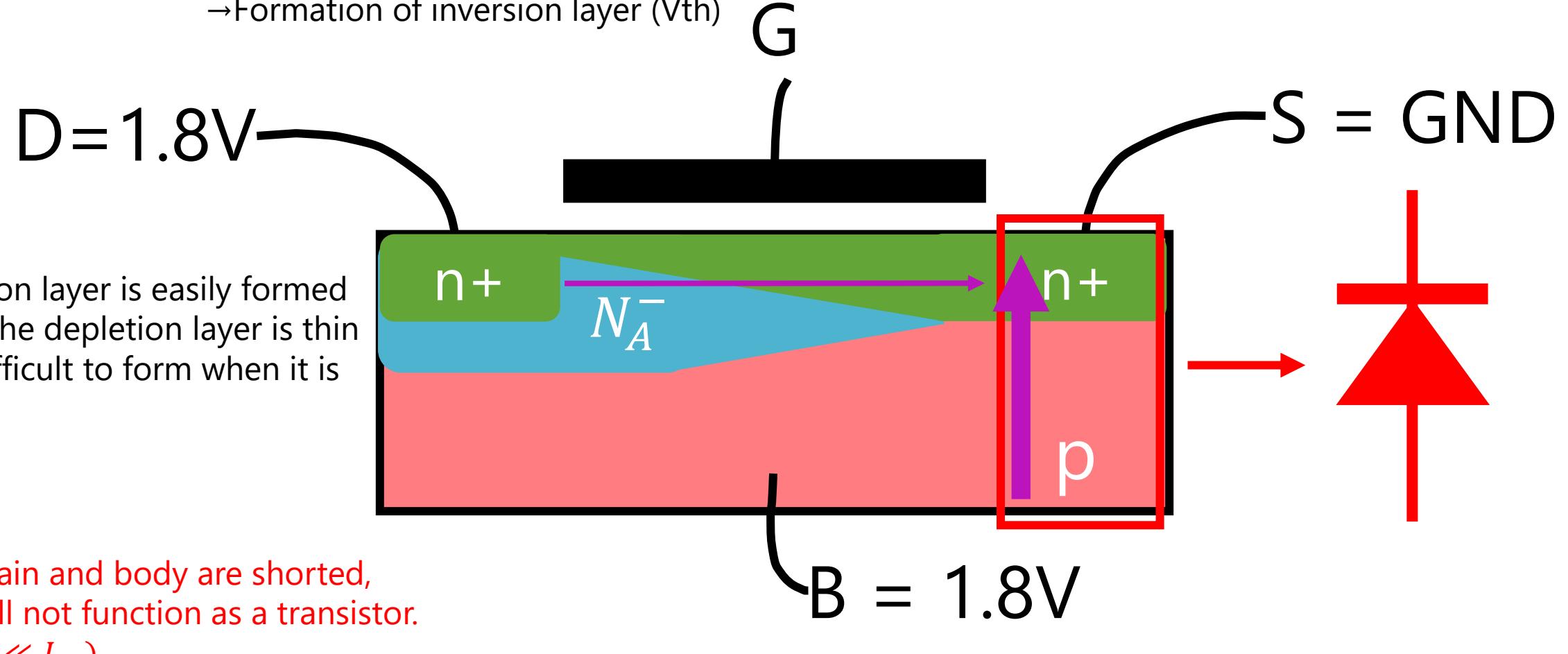
Conduction electrons become majority carriers at $V_g >> 0$
→ Formation of inversion layer (V_{th})



$B >> 0V$
Current conducts from B to S

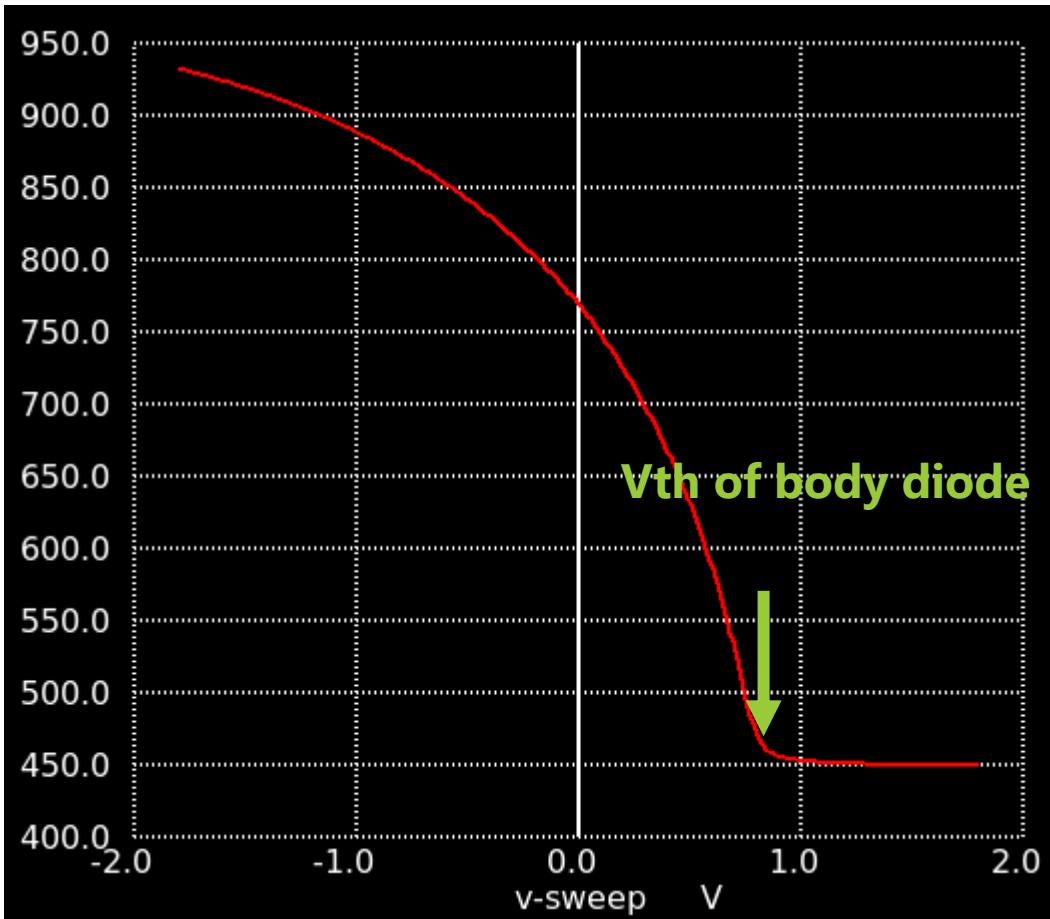
nMOS diagram

Conduction electrons become majority carriers at $V_g >> 0$
→ Formation of inversion layer (V_{th})

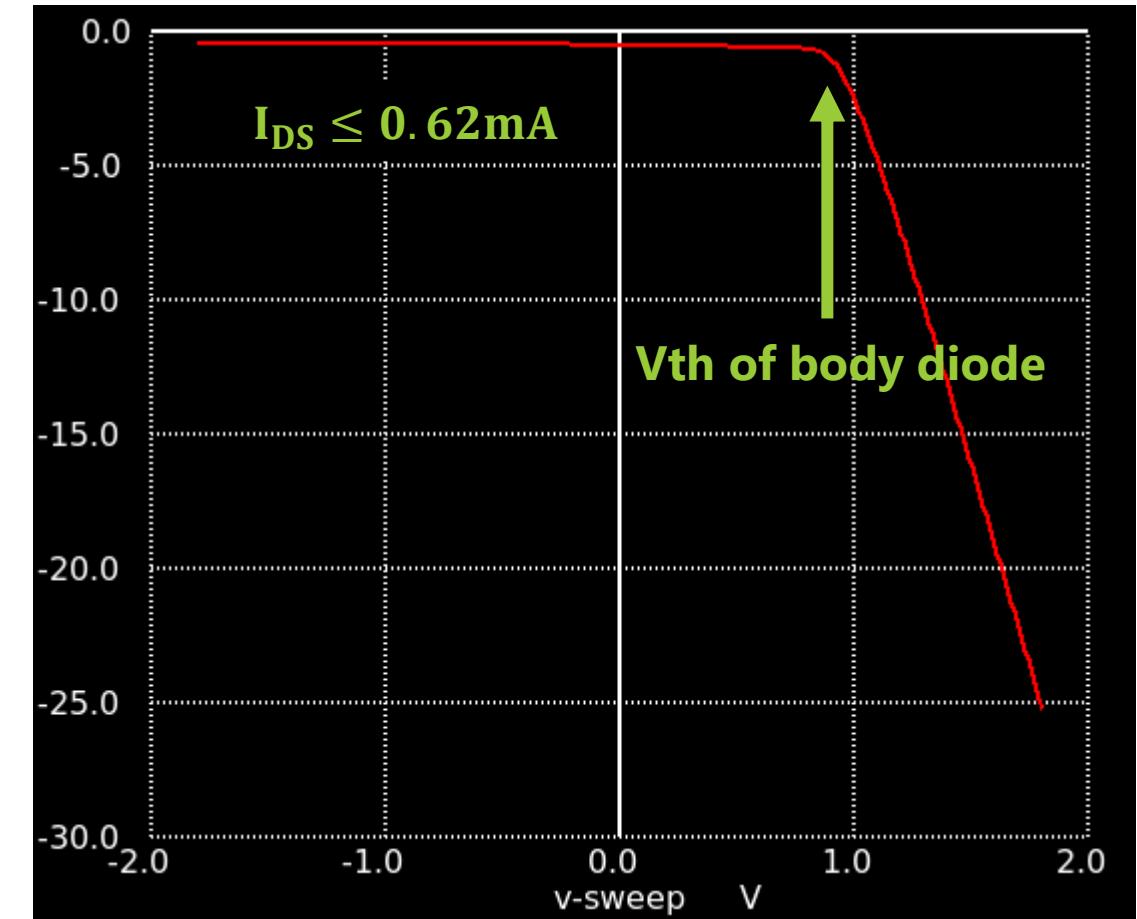


Body effect in nMOS

trbench_body.sch



V_{BS} vs V_{TH}



V_{BS} vs $(I_{DS} + I_{BS})$

To apply bias voltage to the body

Since the transistor will conduct and fail if the biasing of body diode exceeds V_{TH} , Bias voltage is applied as long as it does not exceed that value.

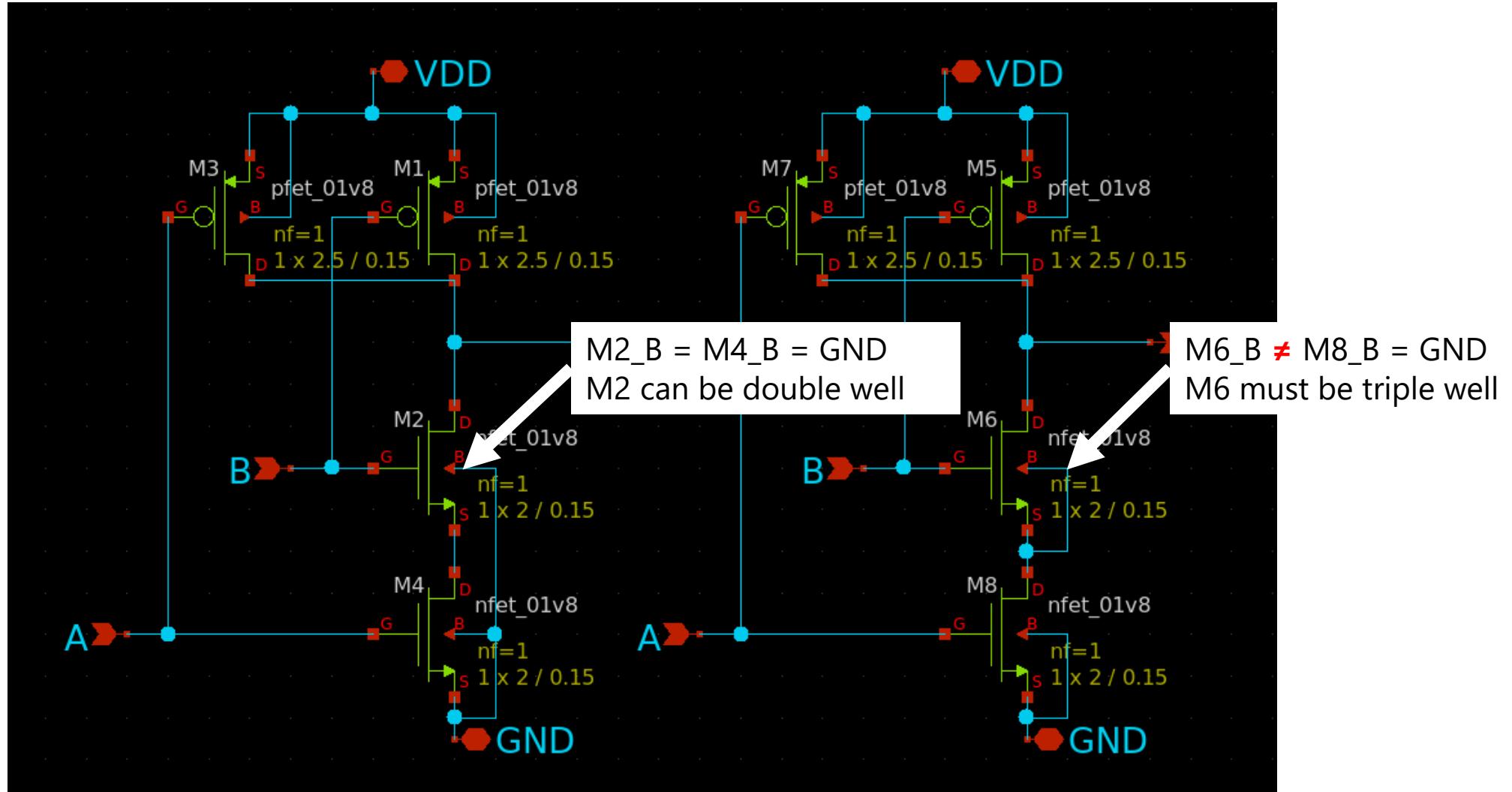
Triple well is required to bias body in nMOS.

- More complicated to draw and larger area than double well.
- Triple-well is disadvantageous in logic circuits.
The larger the area, the lower the density.

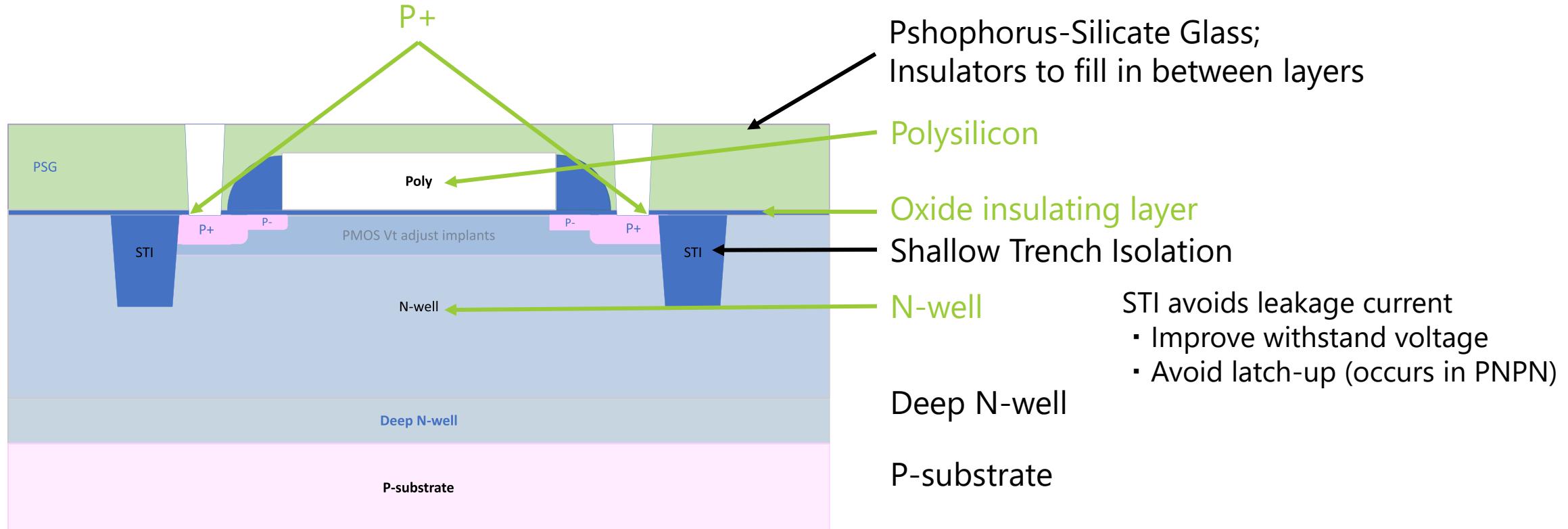
Double well is recommended if there is no preference since it is disadvantageous in terms of integration density.

Connect the body of PMOS to VDD and the body of NMOS to GND (VSS, $\pm 0V$)

Example: 2-input NAND



Planar Pch MOSFET transistor (triple well)



Planar Pch MOSFET transistor

nwell
diff / active
psdm
poly

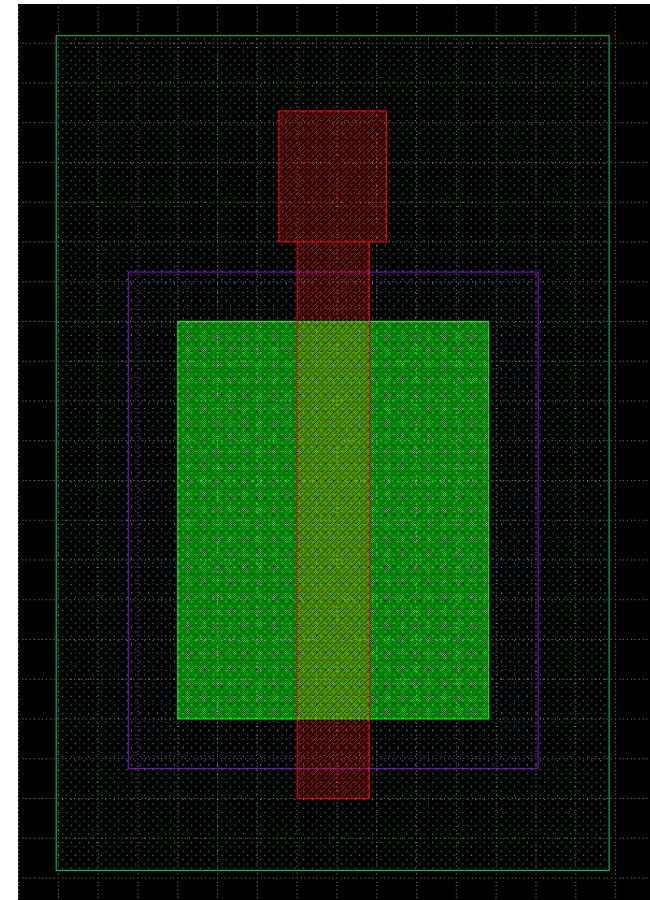
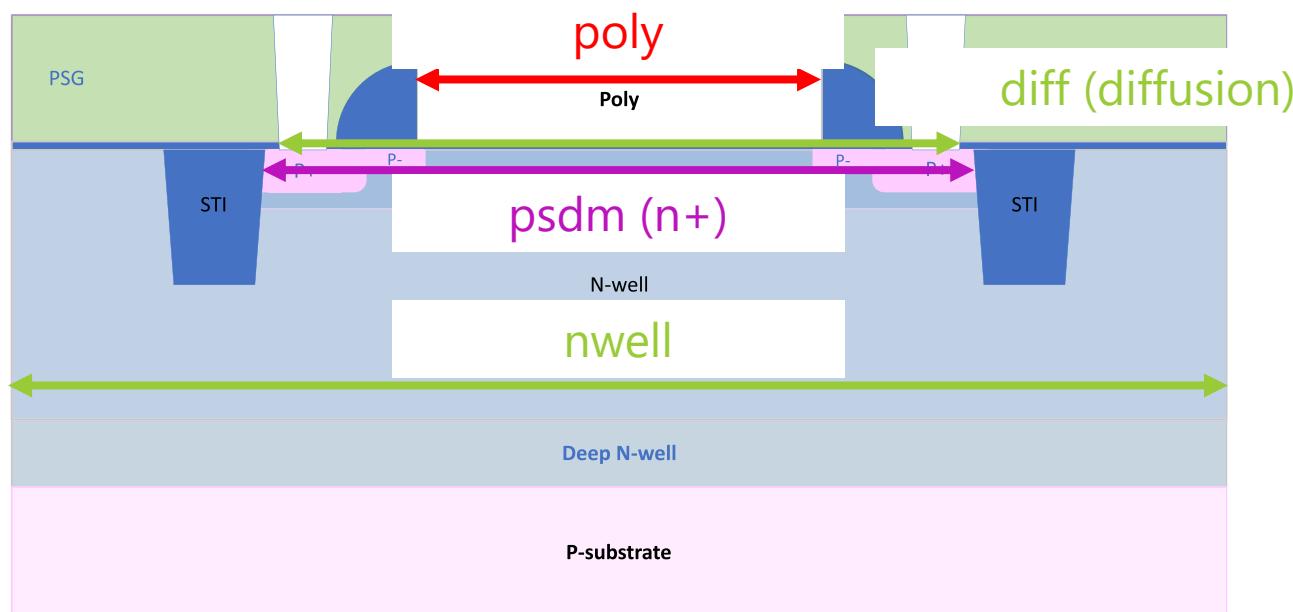
N-well

Diffusion layer of different polarity from well or substrate

P+ implant

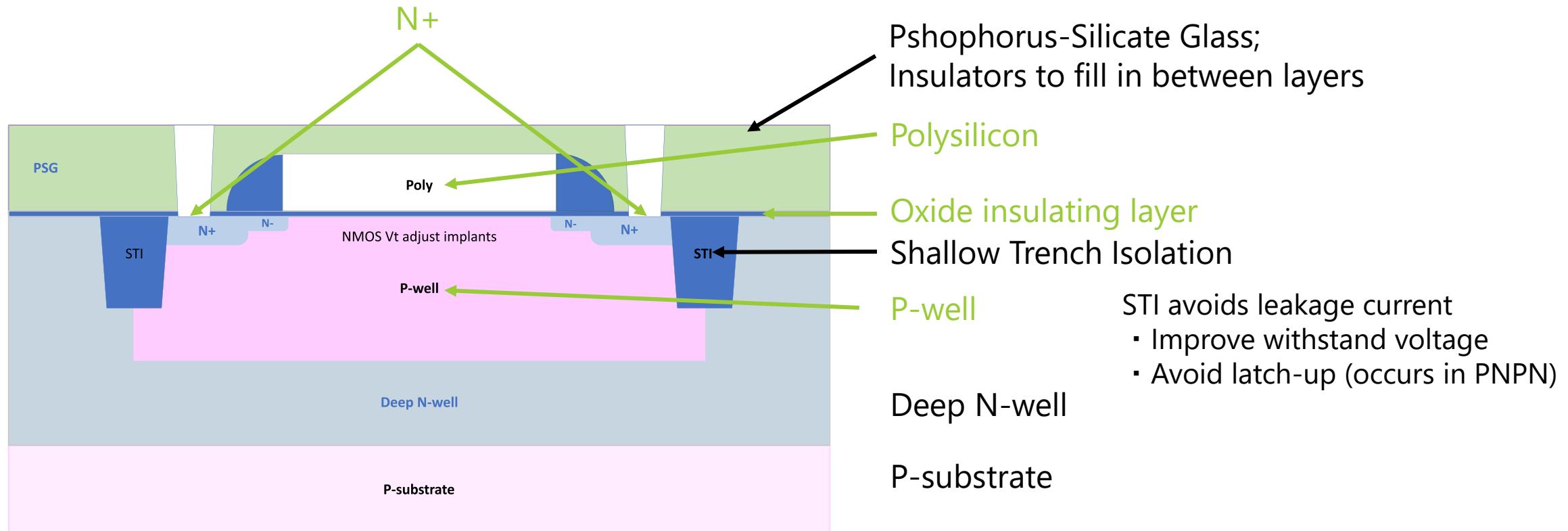
Polysilicon

dnwell is not drawn by default. (not necessary even for triple well)



white	nwell.drawing - 64/20
green	diff.drawing - 65/20
purple	psdm.drawing - 94/...
red	poly.drawing - 66/20

Planar Nch MOSFET transistor (triple well)

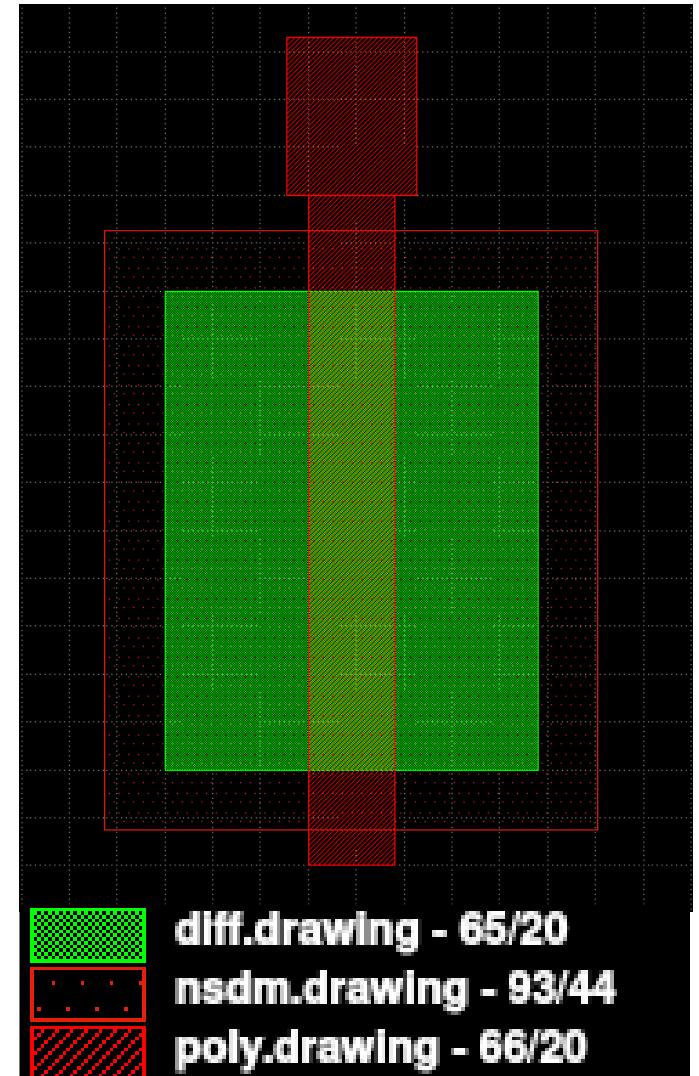
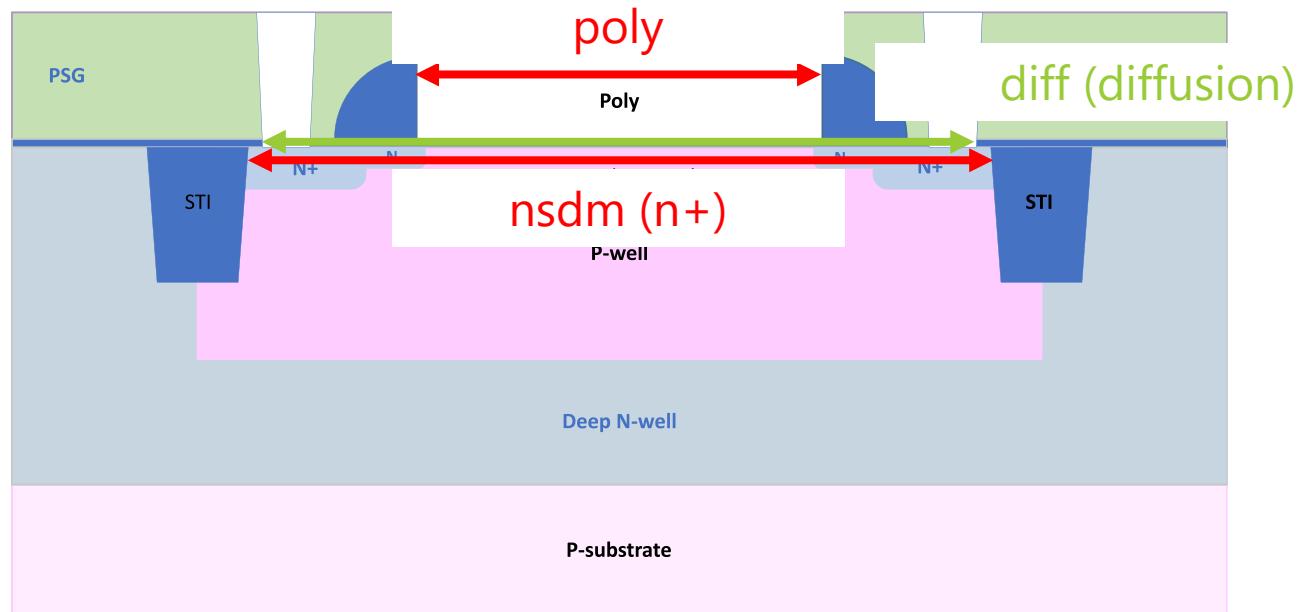


Planar Nch MOSFET transistor

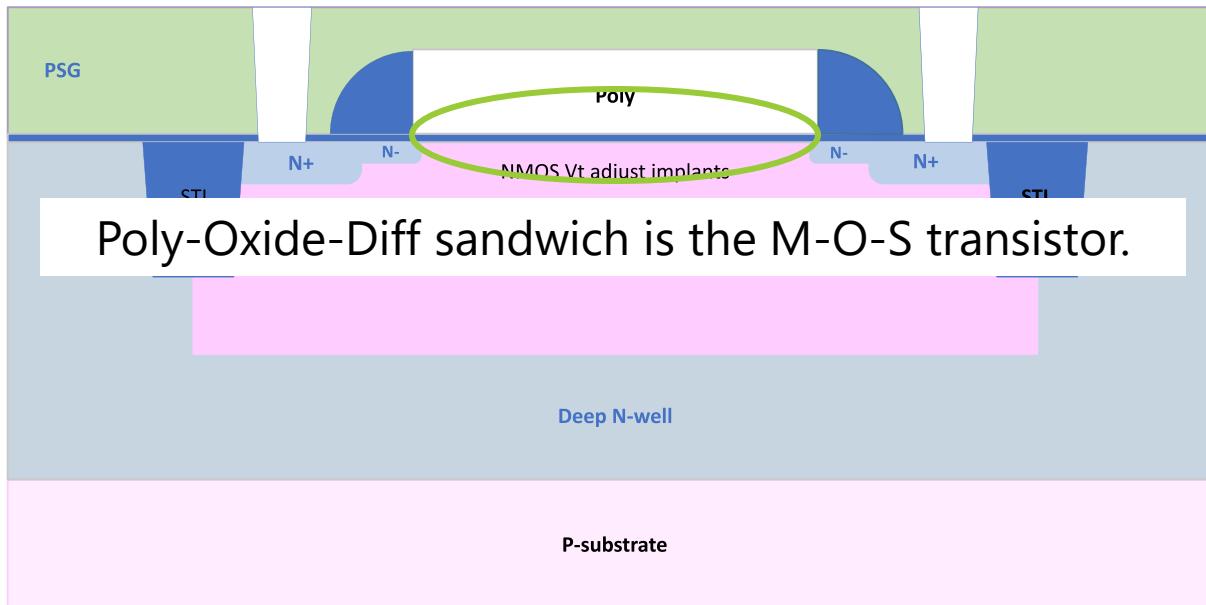
diff / active
nsdm
poly

Diffusion layer of different polarity from well or substrate
N+ implant
Polysilicon

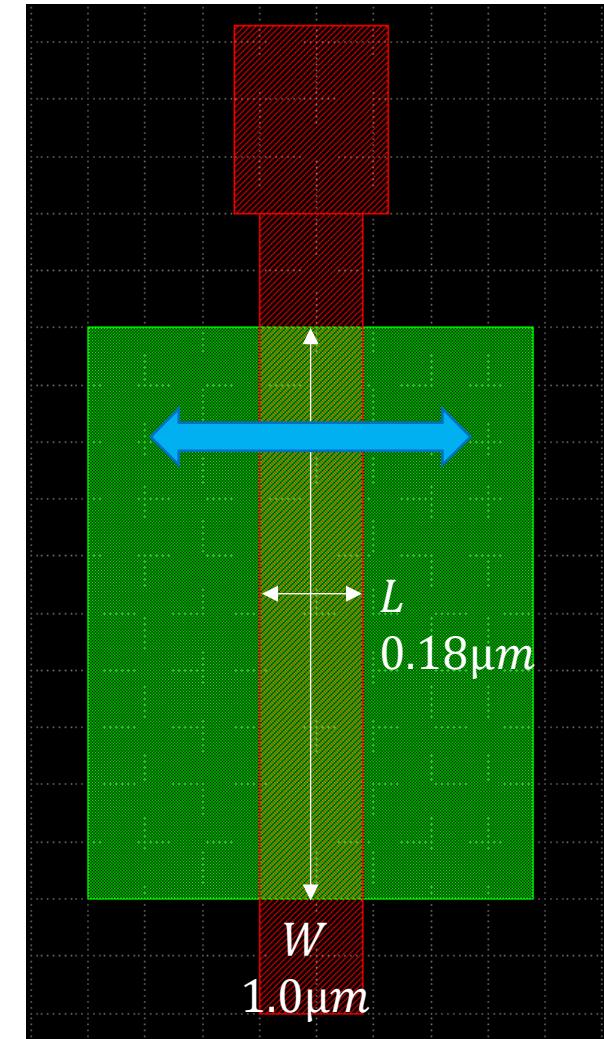
pwell and dnwell are not drawn by default.



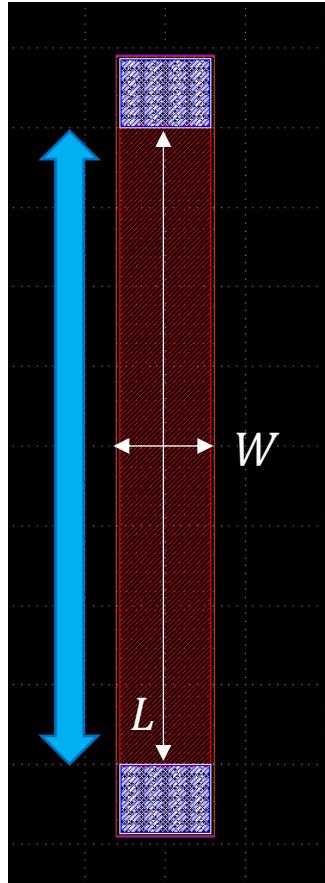
Planar Nch MOSFET transistor



**The direction of L and W is determined
by the direction of current**

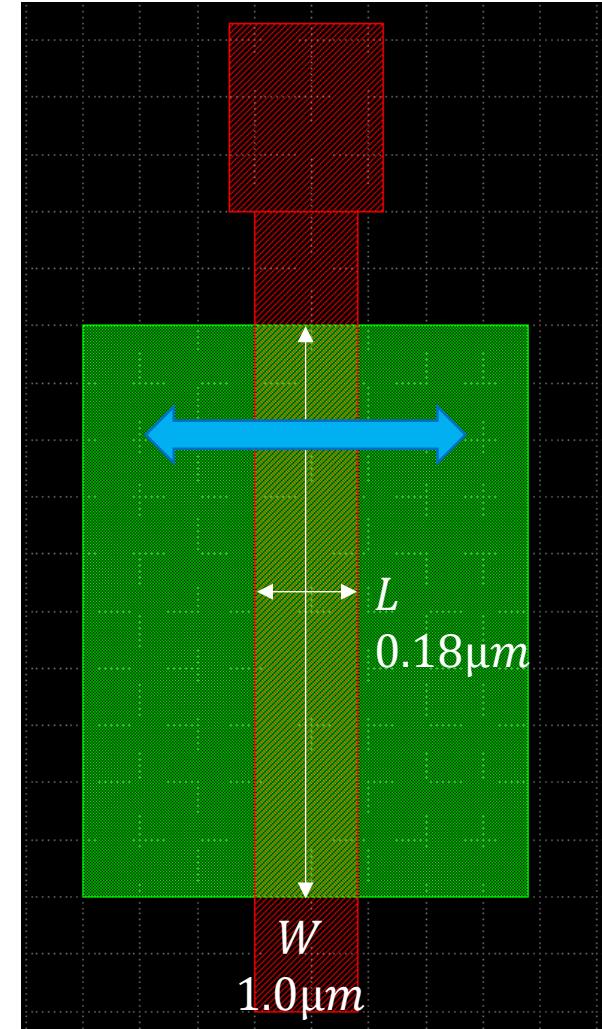


Poly resistor



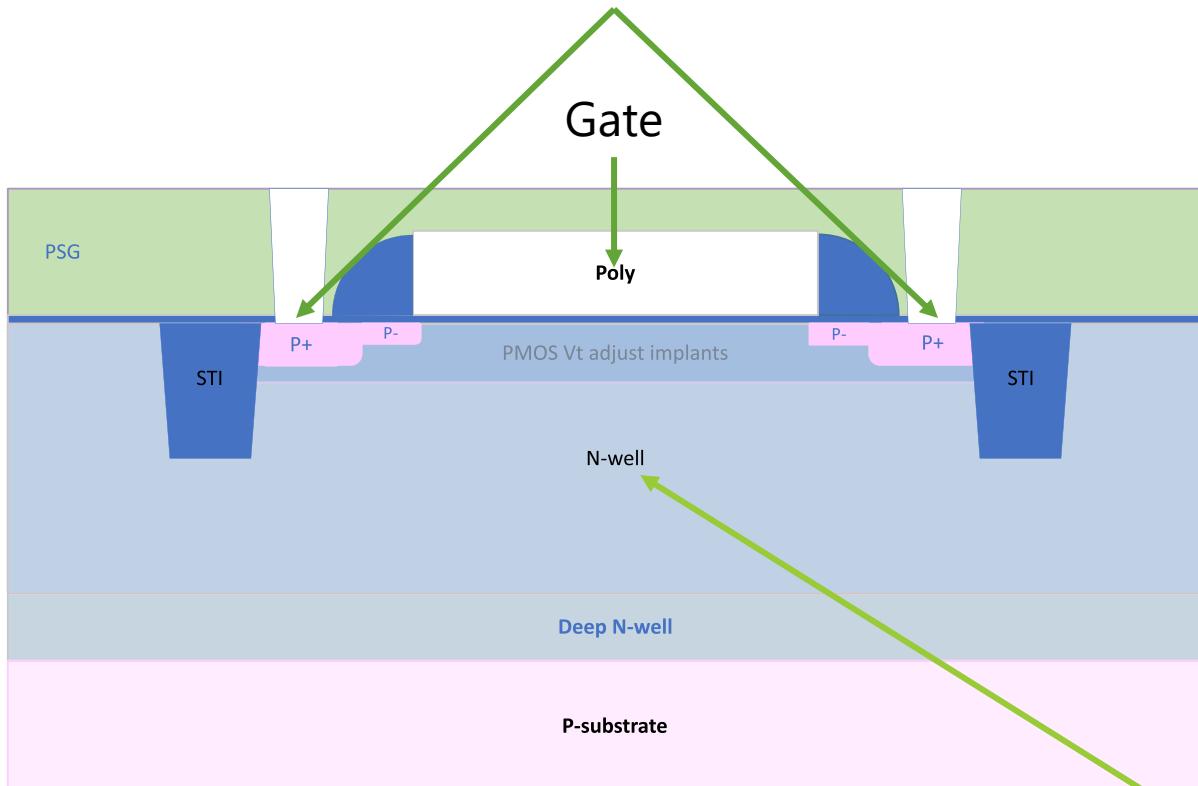
**W/L of poly resistor is
opposite to MOSFET**

**The direction of L and W is determined
by the direction of current**

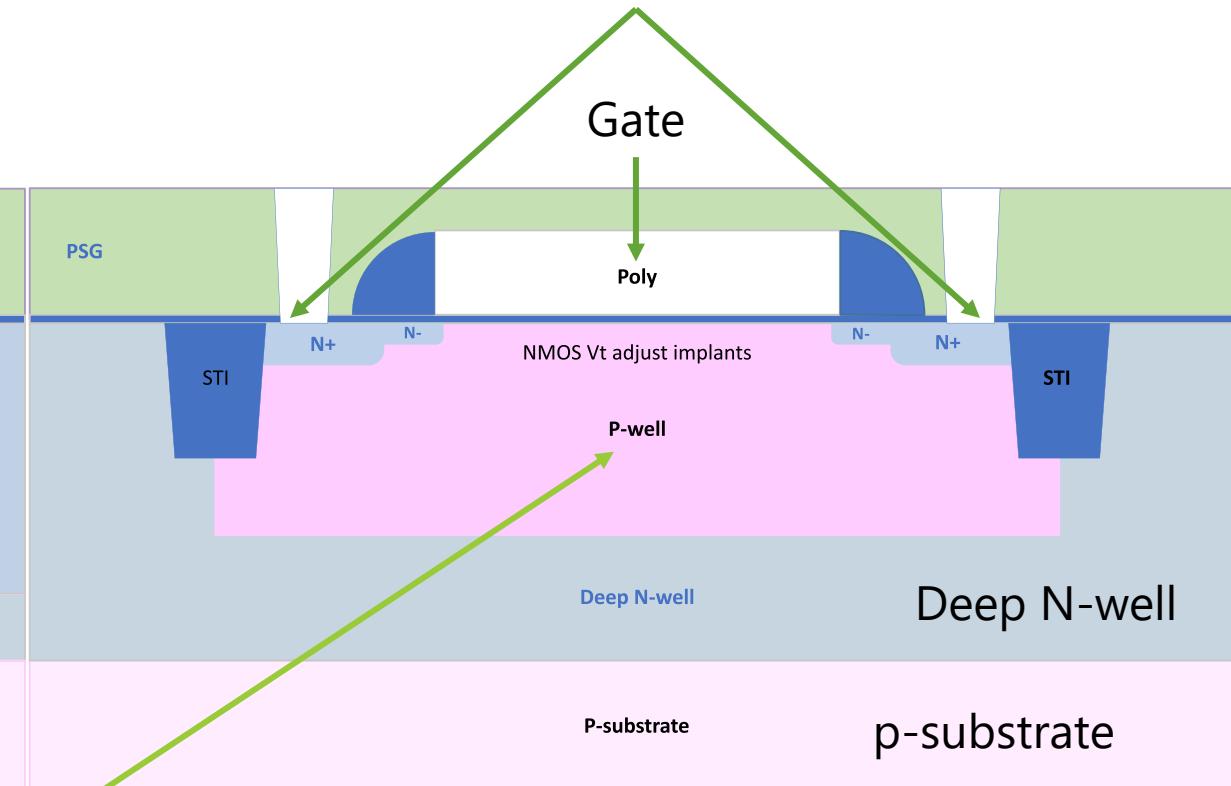


Planar MOSFET transistor (Triple well)

High voltage side is Source, low voltage side is Drain in pMOS



High voltage side is Drain, low voltage side is Source in nMOS



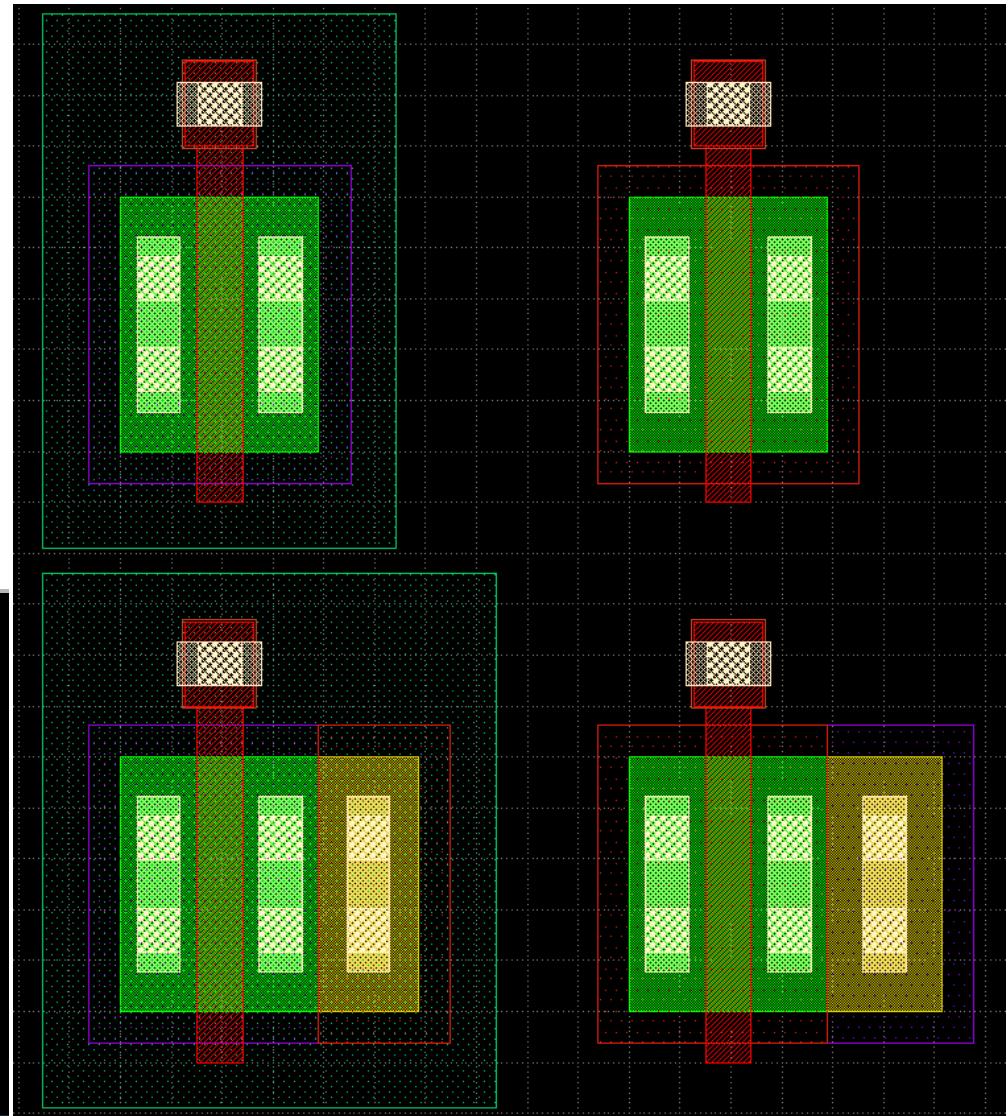
Well of each transistor is a body in the triple-well.
The P-well of nMOS is isolated from the P-substrate by the Deep N-well.

Body connection: tap

Default PFET and NFET are 3-terminal devices and have no body terminals.

Use TAP layer to connect to the body.

	nwell.drawing - 64/20
	diff.drawing - 65/20
	tap.drawing - 65/44
	psdm.drawing - 94/20
	nsdm.drawing - 93/44
	poly.drawing - 66/20
	licon1.drawing - 66/44
	npc.drawing - 95/20
	ll1.drawing - 67/20



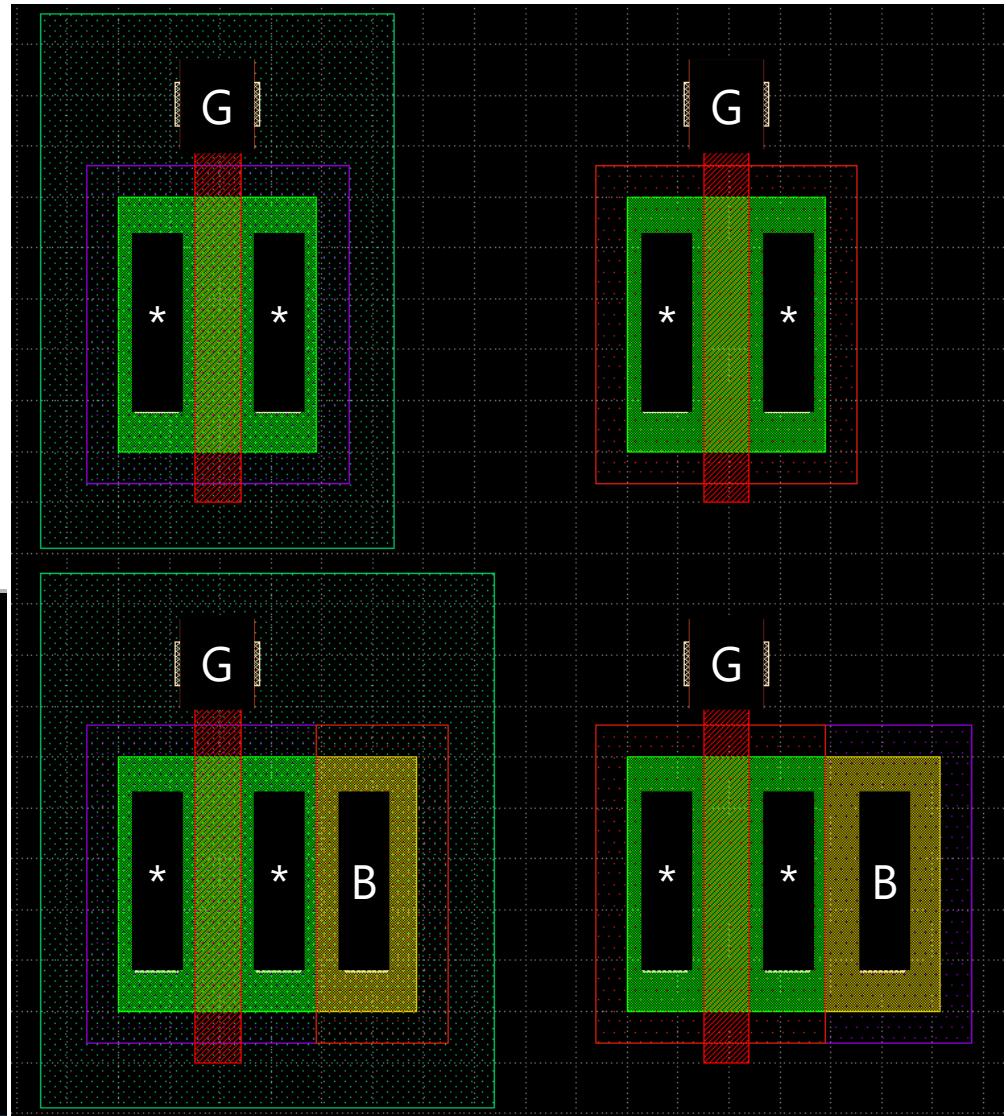
Body connection: tap

Default PFET and NFET are 3-terminal devices and have no body terminals.

Use TAP layer to connect to the body.

* = Drain or Source

	nwell.drawing - 64/20
	diff.drawing - 65/20
	tap.drawing - 65/44
	psdm.drawing - 94/20
	nsdm.drawing - 93/44
	poly.drawing - 66/20
	licon1.drawing - 66/44
	npc.drawing - 95/20
	ll1.drawing - 67/20

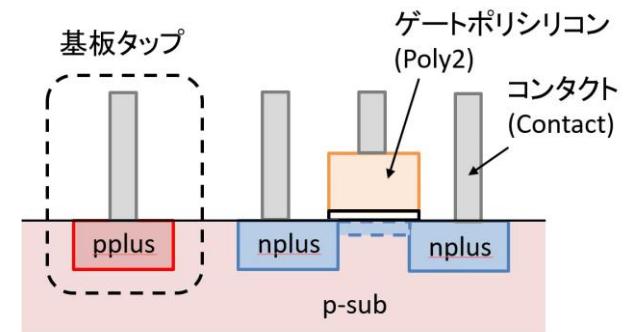
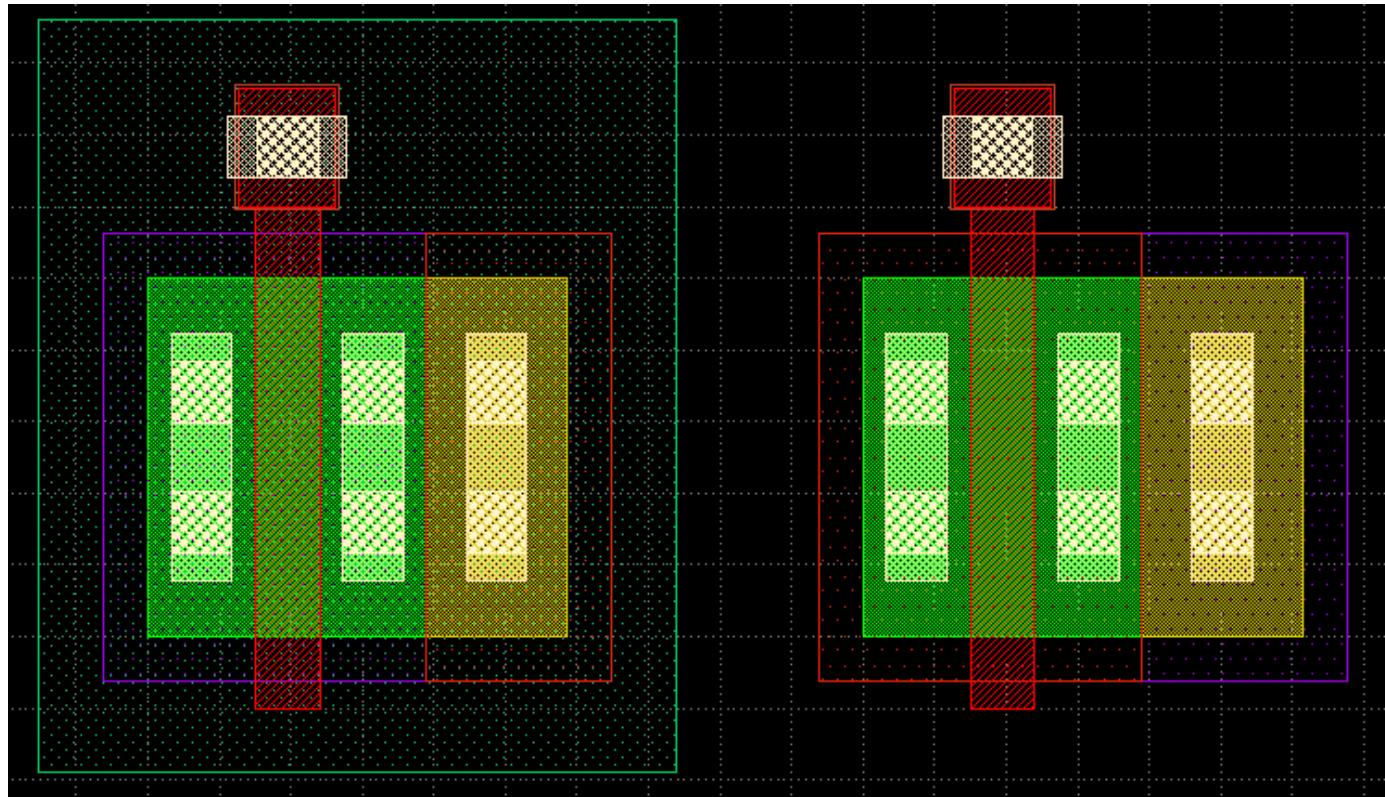


Body connection in Pcell

PFET is connected to N-well via **N+**.

NFET is connected to P-substrate via **P+**.

+ indicates a high doping concentration,
which has the effect of reducing contact resistance



https://note.com/akira_tsuchiya/n/nd4444304f013

	nwell.drawing - 64/20
	diff.drawing - 65/20
	tap.drawing - 65/44
	psdm.drawing - 94/20
	nsdm.drawing - 93/44
	poly.drawing - 66/20
	licon1.drawing - 66/44
	npc.drawing - 95/20
	ll1.drawing - 67/20

Layer names and meanings

nwell N-well

diff Diffusion layer of different polarity from well or substrate

tap Diffusion layer of the same polarity as the well or substrate

psdm P+ implant

nsdm N+ implant

poly Polysilicon

licon1 Contact (Via) to li1 (metal layer)

npc Nitride Poly Cut (polysilicon exposed area)

li1 local interconnect (metal layer)

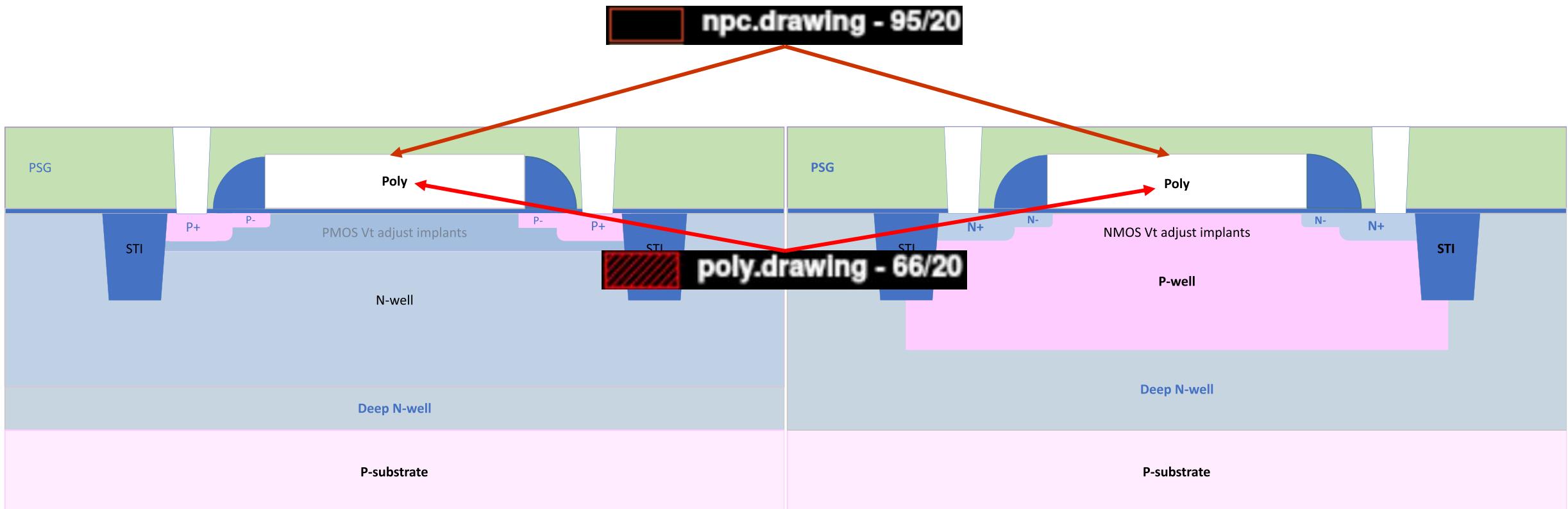
	nwell.drawing - 64/20
	diff.drawing - 65/20
	tap.drawing - 65/44
	psdm.drawing - 94/20
	nsdm.drawing - 93/44
	poly.drawing - 66/20
	licon1.drawing - 66/44
	npc.drawing - 95/20
	li1.drawing - 67/20

Layer names and meanings

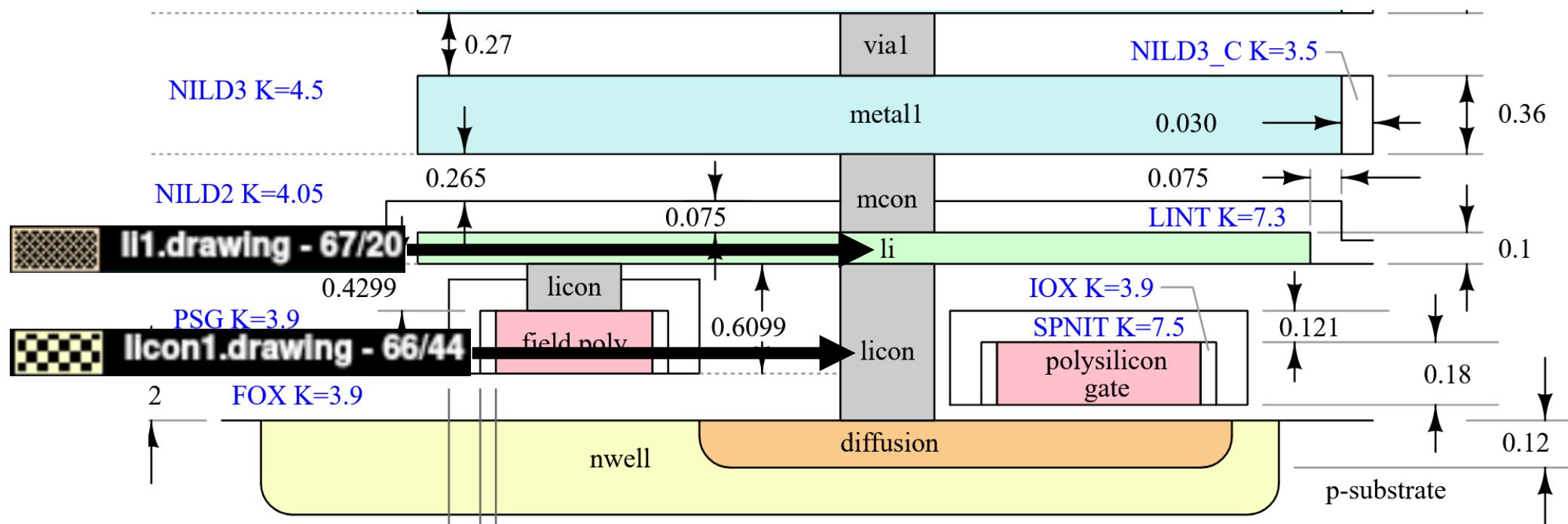
The top of polysilicon is insulated with nitride (not PSG).

npc is a nitride masking layer.

When connecting licon1 (contact to li1), it is necessary to draw npc as well.



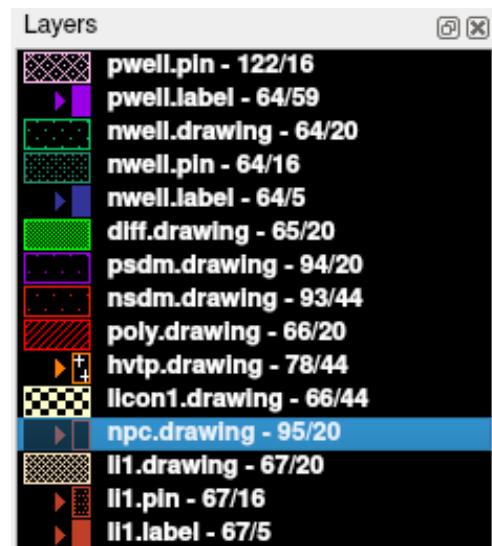
Layer names and meanings



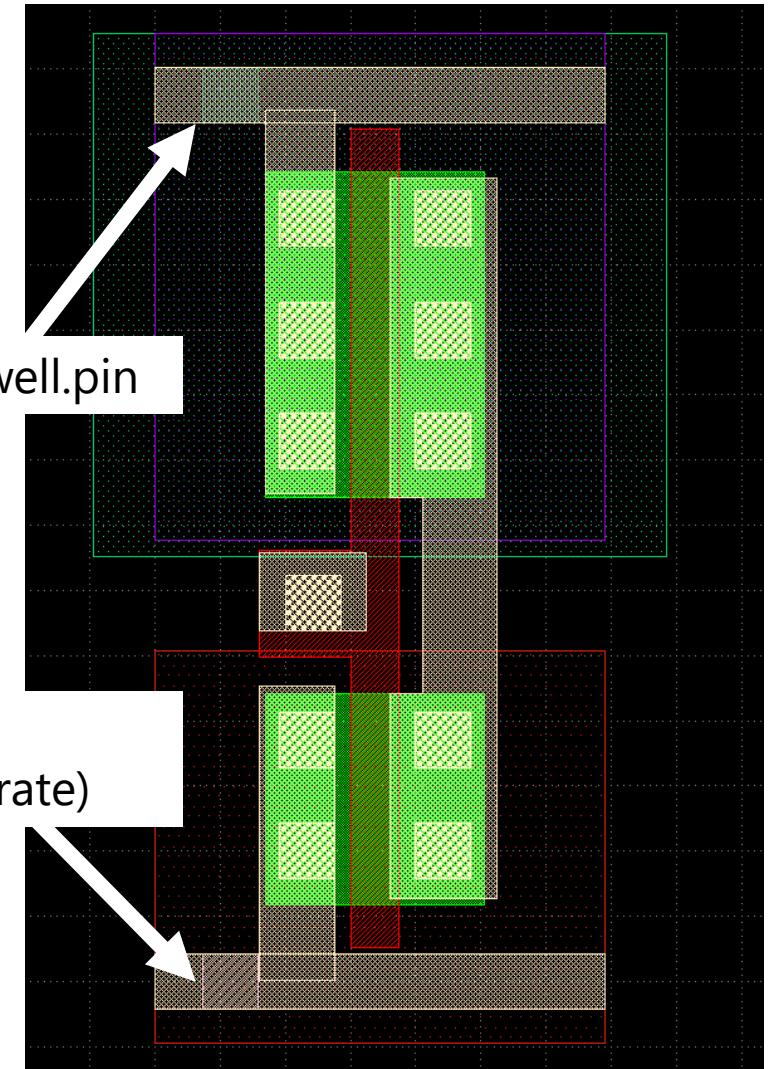
Layer to pass LVS check in tapless design: well.pin

When designing the layout without taps (contacts to body), well.pin can avoid body connection violations in LVS.

well.pin specifies the net name of the substrate / well.
Don't forget that you need to connect the body with tap cells.



pwell.pin
(even with p-substrate)



Layout

Layout: Planar FET

How to draw the layout

1 . Make a LVS schematic (Xschem)

- Delete non-schematic symbols such as power source (idc, vdc...) and commands
- Replace Global net symbol (vdd, gnd...) with iopin.sym

2 . Draw the layout

- You should run DRC every couple of operations until you get used to it.

3 . DRC(Design Rules Check)

- If you don't pass DRC, foundries will not make a chip.

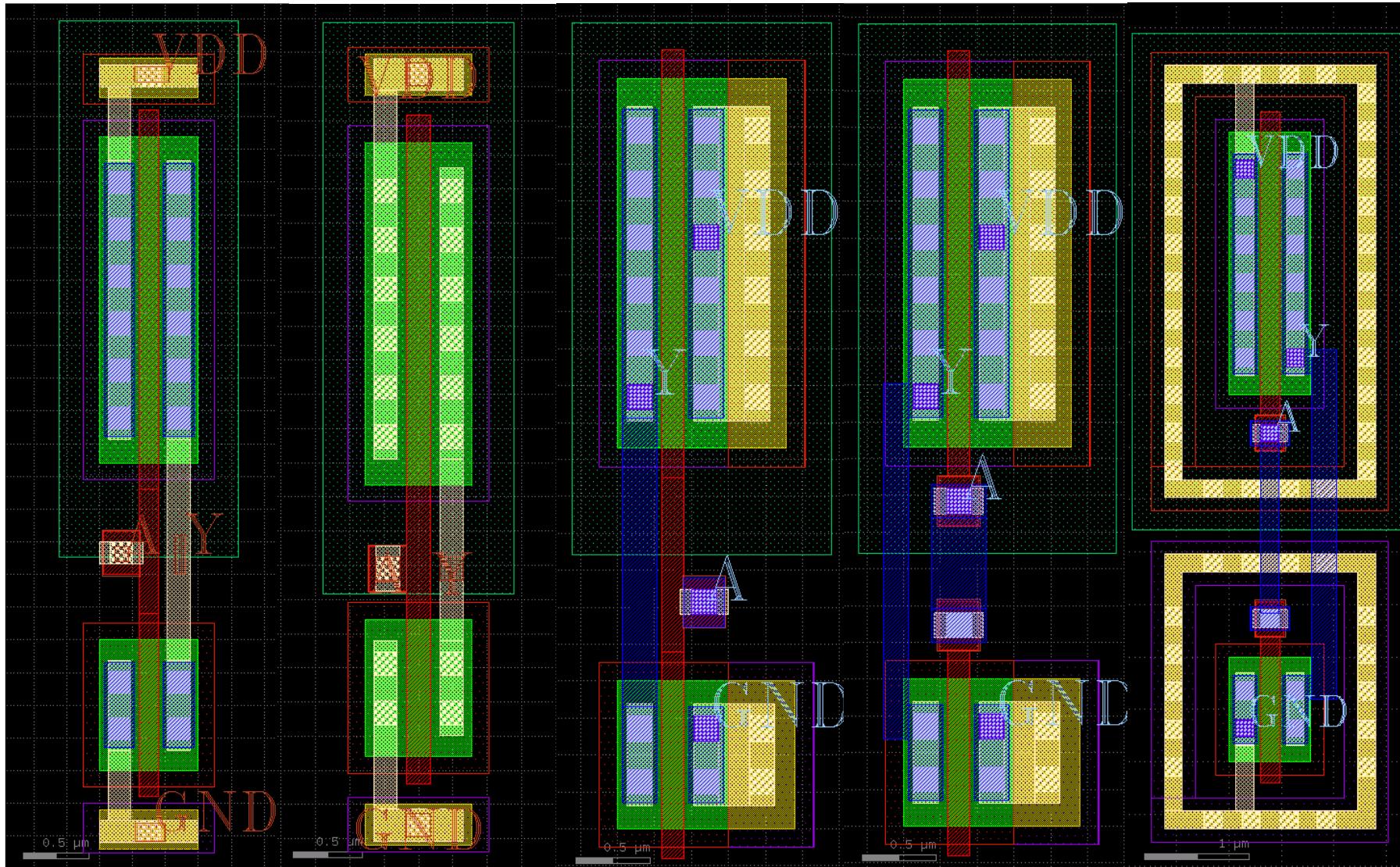
4 . LVS (Layout Versus Schematic)

- Compare your layout with the LVS schematic and check if it is drawn correctly.

5 . PEX (Parasitic Extraction)

Layout
verification

Example layouts of CMOS inverter (NOT gate)



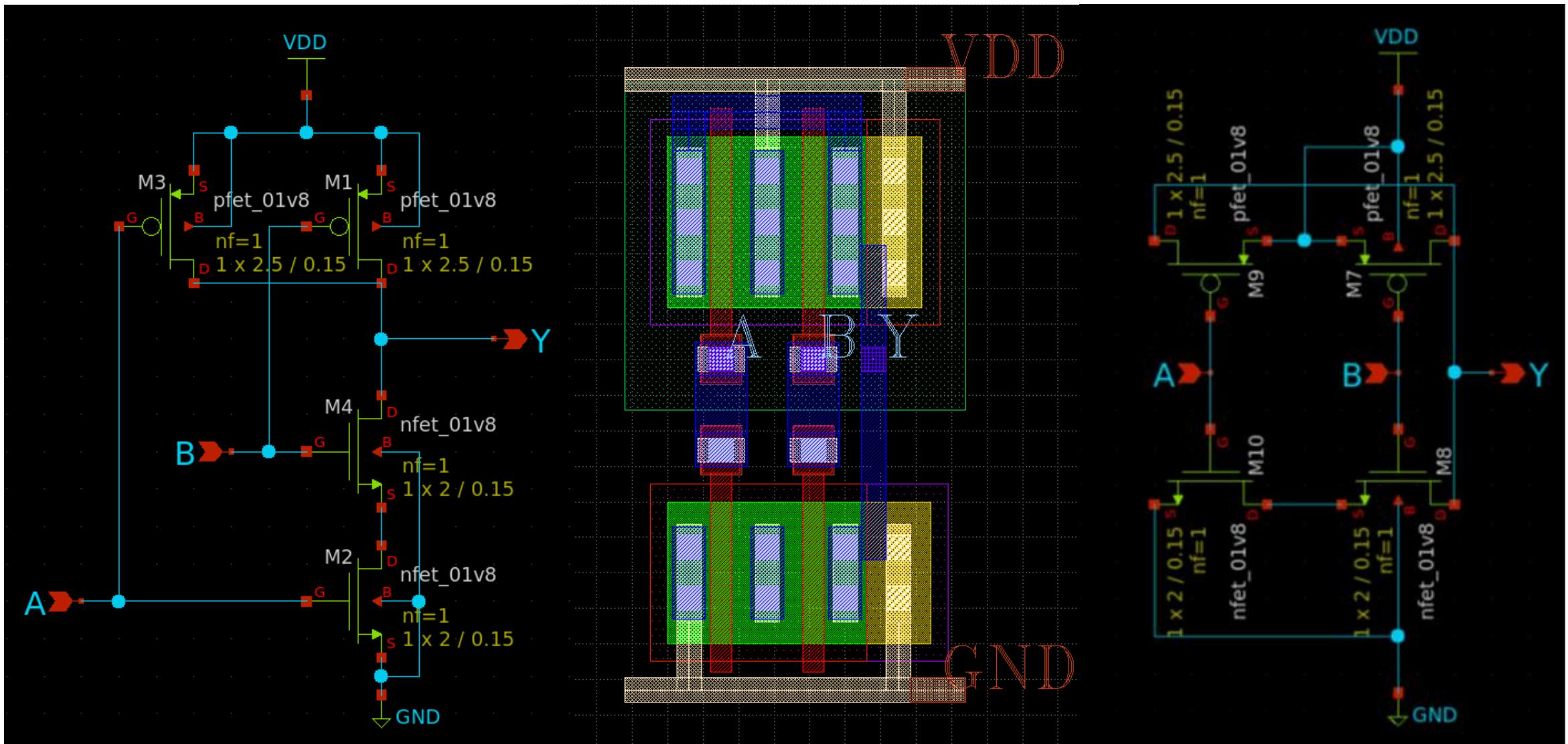
Notes

- **Do not violate design rules**

<https://skywater-pdk.readthedocs.io/en/main/rules/periphery.html>

- Run Klayout DRC (Caravel) to verify that there are no errors and no violations.
- Run Klayout LVS to verify that your layout is equal to the LVS schematic.
 - If they do not match, post-layout simulation will not work properly.
Of course, real chip may not work too.
 - Magic DRC/LVS is also available.

2-input NAND layout based on Pcell



Major design rules

SKY130

Design rules

All design rules are here

- <https://skywater-pdk.readthedocs.io/en/main/rules/periphery.html>
- Here are some important design rules for drawing CMOS inverters

Transistors are automatically generated by Pcell, so we will focus on the metal layer.

GRID / OFFGRID

x.1b: All layers must be on a grid of 0.005μm.

Some layers (met1, via, met2) are available down to 0.001μm (x.1a),
however, the Klayout DRC is unified at 0.005μm for all layers.

- For example, when you draw the rectangle,
the coordinates of 4 points must be 0.005μm steps.

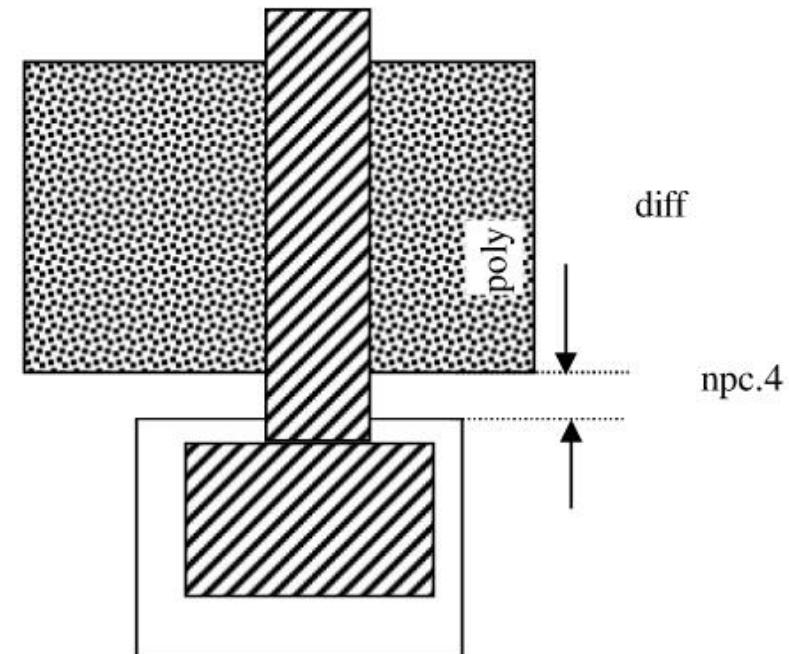
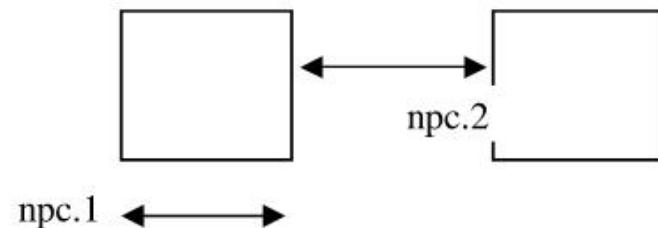
OFFGRID error occurs when the coordinate finer than 0.005μm is specified

npc.drawing : Nitride Poly Cut

npc.1 : Minimum width : 0.270 μm

npc.2 : Minimum space of npc to npc: 0.270 μm

npc.4 : Minimum space of npc to diff/tap: 0.090 μm



licon1.drawing : Local Interconnect Contact

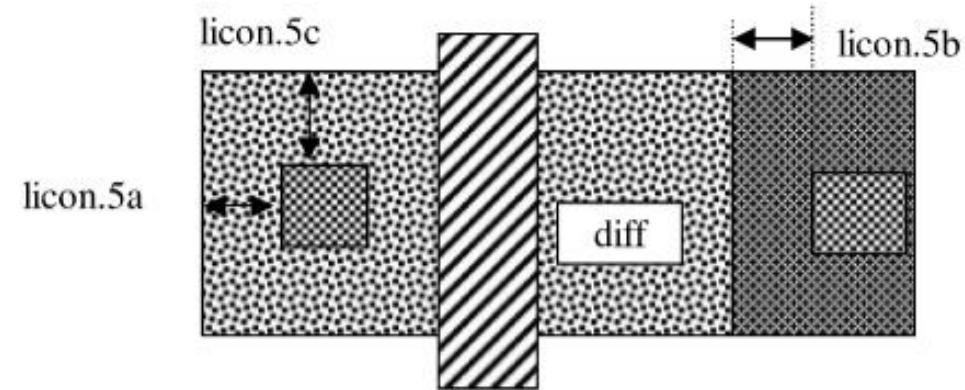
licon.1 : Contact size must be $0.170 \mu\text{m} \times 0.170 \mu\text{m}$

licon.2 : Minimum space of licon to licon: $0.170 \mu\text{m}$

Minimum enclosure of licon by diff:

licon.5a : In either one of the x,y axes : $0.040 \mu\text{m}$

licon.5c : In the other of the x,y axes : $0.060 \mu\text{m}$



Minimum enclosure of licon by tap:

licon.7 : In either one of the x,y axes : $0.120 \mu\text{m}$

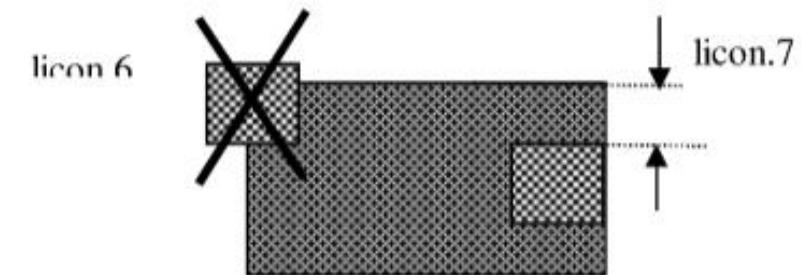
licon.6 : May contact in the other axis, but must not overhang

licon.5b : Min space between tap_licon and diff-abutting tap edge: $0.060 \mu\text{m}$ from diff

Minimum enclosure of licon by poly:

licon.8 : In either one of the x,y axes : $0.050 \mu\text{m}$

licon.8a : In the other of the x,y axes : $0.080 \mu\text{m}$



li1.drawing : Local Interconnect

li.1 : Minimum space : 0.170 μm

li.2 : Maximum L/W ratio without licon or mcon : 10 **DRC cannot check li.2 rule**

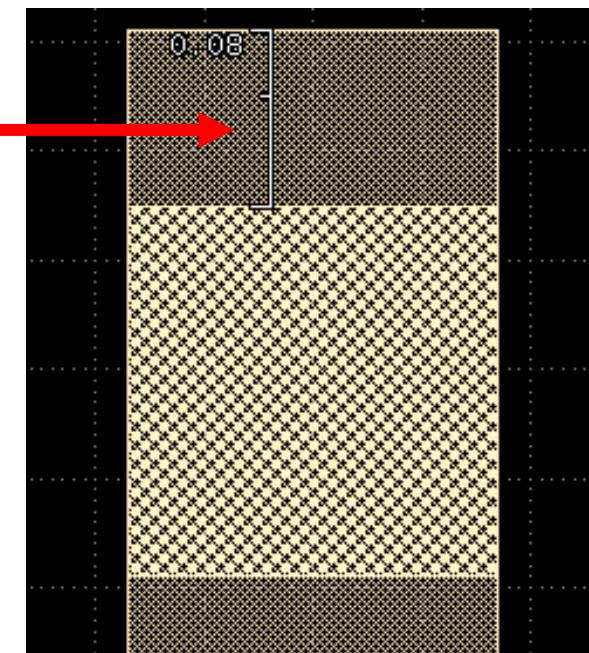
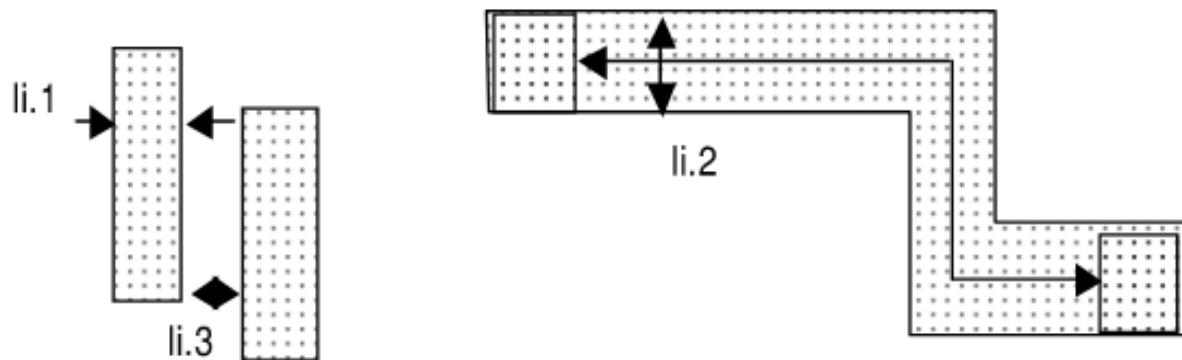
li.3 : Minimum space of li to li : 0.170 μm

li.6 : Minimum area : 0.0561 μm^2

Minimum enclosure of li1 by licon:

li.5 : In either one of the x,y axes : 0.080 μm

licon.6 : May contact in the other axis, but must not overhang



mcon.drawing : Metal Contact (li1 - met1)

ct.1 : Contact size must be $0.170 \mu\text{m} \times 0.170 \mu\text{m}$

ct.2 : Minimum space of mcon to mcon : $0.190 \mu\text{m}$

Minimum enclosure of mcon by li:

ct.4 : mcon must not overhang li1.drawing ($0.000 \mu\text{m}$)

Minimum enclosure of met1 by mcon:

m1.4 : In either one of the x,y axes : $0.030 \mu\text{m}$

m1.5 : In the other of the x,y axes : $0.060 \mu\text{m}$

met1/2/3.drawing : Metal

met1.drawing :

m1.1 : Minimum width : 0.140 μm

m1.2 : Minimum space of met1 to met1: 0.140 μm

m1.6 : Minimum area : 0.083 μm^2

met2.drawing :

m2.1 : Minimum width : 0.140 μm

m2.2 : Minimum space of met2 to met2 : 0.140 μm

m2.6 : Minimum area : 0.0676 μm^2

met3.drawing :

m3.1 : Minimum width : 0.300 μm

m3.2 : Minimum space of met3 to met3 : 0.300 μm

m3.6 : Minimum area : 0.240 μm^2

via.drawing : Via (met1 - met2)

via.1 : Contact size must be $0.150 \mu\text{m} \times 0.150 \mu\text{m}$

via.2 : Minimum space of via to via : $0.170 \mu\text{m}$

Minimum enclosure of via by met1:

via.4 : In either one of the x,y axes : $0.055 \mu\text{m}$

via.5 : In the other of the x,y axes : $0.085 \mu\text{m}$

Minimum enclosure of met2 by via:

m2.4 : In either one of the x,y axes : $0.055 \mu\text{m}$

m2.5 : In the other of the x,y axes : $0.085 \mu\text{m}$

via2.drawing : Via2 (met2 – met3)

via.1 : Contact size must be $0.200 \mu\text{m} \times 0.200 \mu\text{m}$

via.2 : Minimum space of via to via : $0.200 \mu\text{m}$

Minimum enclosure of via2 by met2:

via.4 : In either one of the x,y axes : $0.040 \mu\text{m}$

via.5 : In the other of the x,y axes : $0.085 \mu\text{m}$

Minimum enclosure of met3 by via2:

m3.4 : $0.065 \mu\text{m}$ on all four sides.

Post-layout simulation

Parasitic extraction

Layout Validation: PEX, Parasitic Extraction

RC extraction

Make a netlist with parasitic resistance and parasitic capacitance

```
File Edit View Search Terminal Help
Nets output: 4 (1.000000)
exttospice finished.
* NGSPICE file created from TOP.ext - technology: sky130A

.subckt TOP A Y VDD GND
X0 Y.t1 A.t0 VDD.t1 VDD.t0 sky130_fd_pr_pfet_01v8 ad=0p pd=0u as=0p ps=0u w=2.5
e+06u l=180000u
X1 Y.t0 A.t1 GND.t1 GND.t0 sky130_fd_pr_nfet_01v8 ad=0p pd=0u as=0p ps=0u w=1e+
06u l=180000u
R0 A.n0 A.t0 472.895
R1 A.n0 A.t1 207.794
R2 A A.n0 79.296
R3 VDD VDD.t0 1385.67
R4 VDD VDD.t1 254.38
R5 Y Y.t1 270.032
R6 Y Y.t0 134.44
R7 GND GND.t0 4820.4
R8 GND GND.t1 150.035
C0 Y A 0.05fF
C1 A VDD 0.18fF
C2 Y VDD 0.17fF
.ends
```

C extraction

Make a netlist with parasitic capacitance only

```
File Edit View Search Terminal Help
Loading sky130A Device Generator Menu ...
Loading "/home/user/.klayout/macros/sky130_magic_pex.tcl" from command line.
Warning: Calma reading is not undoable! I hope that's OK.
Library written using GDS-II Release 6.0
Library name: LIB
Reading "TOP".
CIF file read warning: CIF style sky130(): units rescaled by factor of 5 / 1
Extracting TOP into TOP.ext:
exttosim finished.
exttospice finished.
exttospice finished.
* NGSPICE file created from TOP.ext - technology: sky130A

.subckt TOP A Y VDD GND
X0 Y A VDD VDD sky130_fd_pr_pfet_01v8 ad=7.5e+11p pd=5.6e+06u as=7.5e+11p ps=5.
6e+06u w=2.5e+06u l=180000u
X1 Y A GND GND sky130_fd_pr_nfet_01v8 ad=3e+11p pd=2.6e+06u as=3e+11p ps=2.6e+0
6u w=1e+06u l=180000u
C0 A Y 0.05fF
C1 VDD Y 0.17fF
C2 A VDD 0.18fF
.ends
```

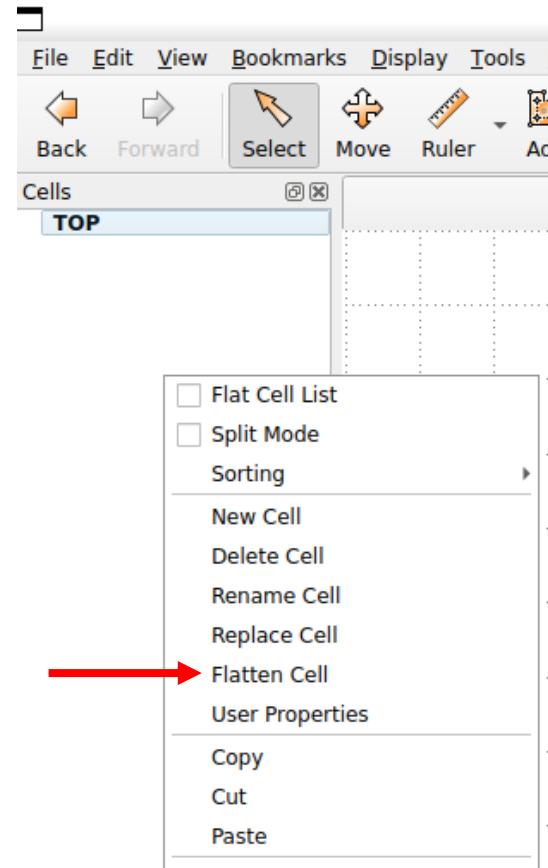
Preparation before PEX: Flatten

1. Save the copy of your layout

- Using UPPERCASE LETTERS in file name may cause some bugs.
 - Change the extension from .GDS to .gds

2. Run Cells > Flatten Cell > All hierarchy levels

3. Make sure that “Cells” is TOP (or the name you set) only and save it.



Parasitic extraction

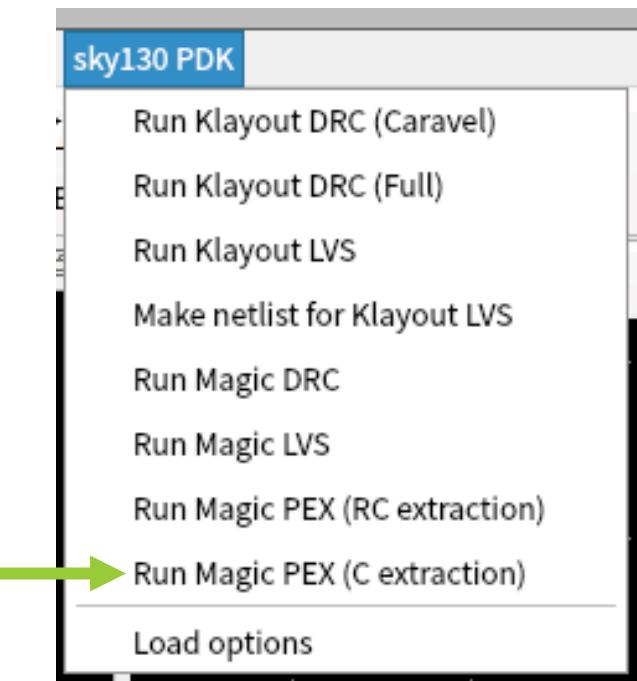
Run the parasitic capacitance extraction using Magic PEX.

- Run Magic PEX (C extraction)

A netlist with parasitic capacitance is generated

Filename : {layout name}_pex_extracted.spice

- Default: TOP_pex_extracted.spice

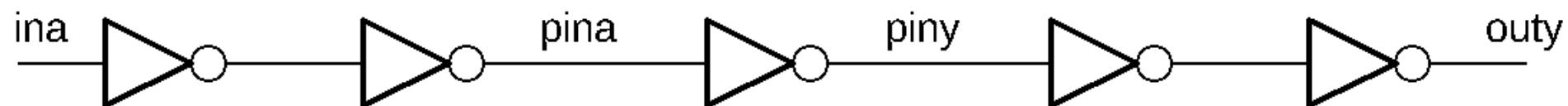


Example of a delay time simulation

CMOS inverter

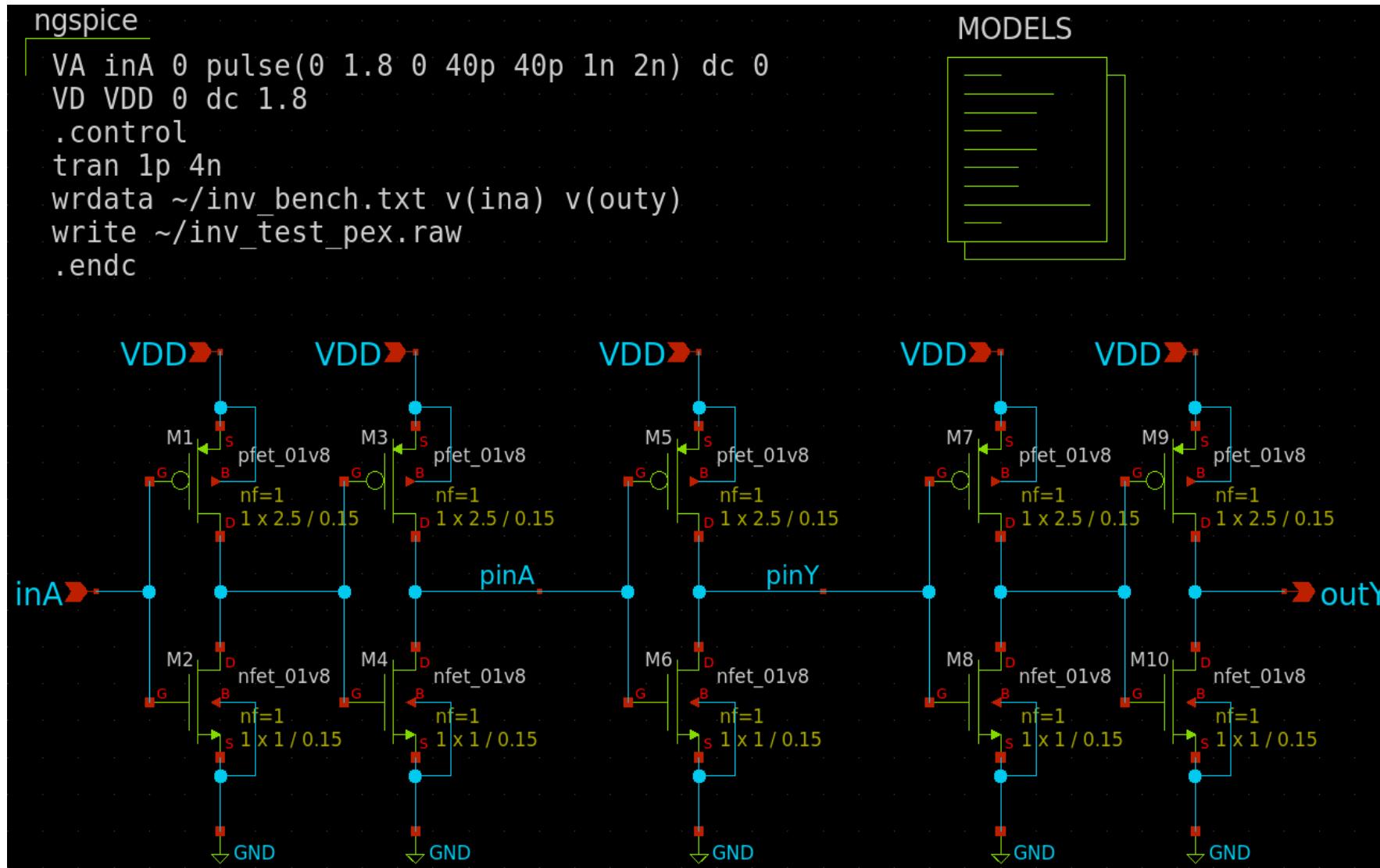
$$P=2.5\text{u} / N=1\text{u}$$

$$P=2.5\text{u} / N=1\text{u}$$



- The size of first stage is $w_p = 2.5 \mu\text{m}$, $w_n = 1 \mu\text{m}$
- Output load capacitance is the same as input: $w_p = 2.5 \mu\text{m}$, $w_n = 1 \mu\text{m}$
- Simulate the delay time from ina to outy

Example of a test bench for measuring ideal delay time



Modify the test bench to post-layout simulation

The following changes are required for post-layout simulation

1. PATH to the extracted netlist
2. Symbol for the extracted netlist

Set the PATH to the extracted netlist

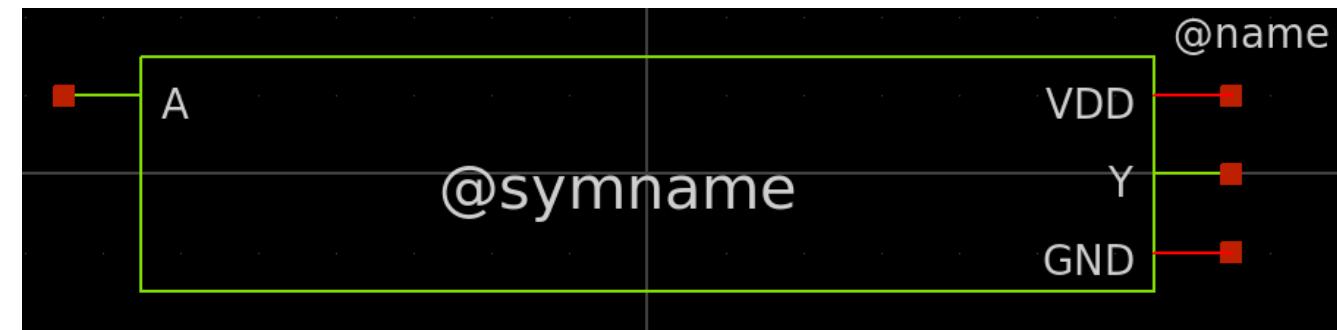
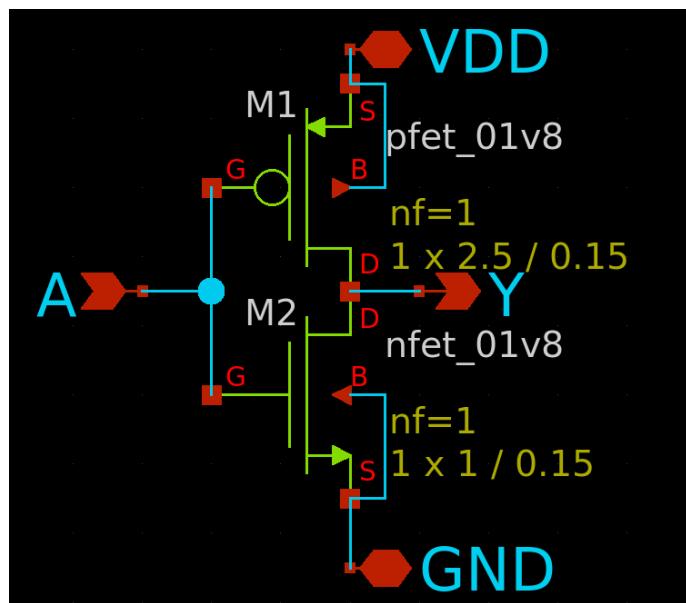
- Set the PATH to the extracted netlist by .include
 - Extracted netlist is in the same directory as the layout (.gds)

```
ngspice
  VA inA 0 pulse(0 1.8 0 40p 40p 1n 2n) dc 0
  VD VDD 0 dc 1.8
  →.include ~/T0P_pex_extracted.spice
  .control
    tran 1p 4n
    wrdata ~/inv_bench.txt v(ina) v(outy)
    write ~/inv_test_pex.raw
  .endc
```

How to make a symbol for the extracted netlist

Make a symbol from LVS schematic

- Menu Symbol > Make symbol from schematic



Generated symbol refer to the LVS schematic.
We have to modify this to refer to the extracted netlist.

How to make a symbol for the extracted netlist

Open the symbol's property and set like this (key binding: Q)

```
type=primitive  
format="@name [pins in netlist] @prefix"  
template="name=x1 prefix=[title of subckt]"  
extra="prefix"  
highlight=true
```

See http://repo.hu/projects/xschem/xschem_man/symbol_property_syntax.html

How to make a symbol for the extracted netlist

Example of CMOS inverter

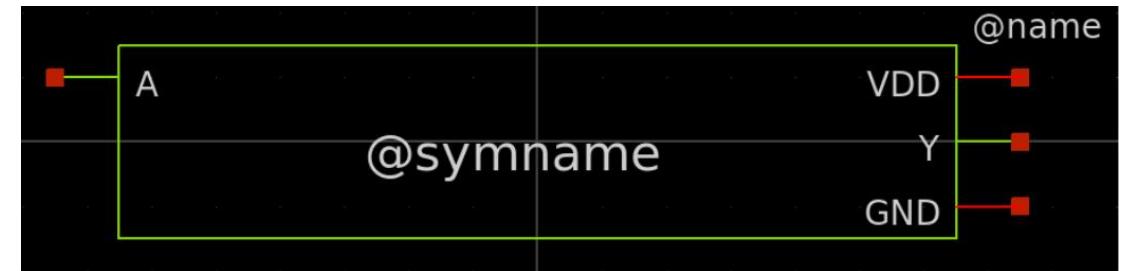
```
type=primitive  
format="@name @@A @@Y @@VDD @@GND @prefix"  
template="name=x1 prefix=TOP"  
extra="prefix"  
highlight=true
```

Describe in the same order
as the pins in the extracted netlist

```
.subckt TOP A Y VDD GND
X0 Y A VDD VDD sky130_fd_pr_pfe
X1 Y A GND GND sky130_fd_pr_nfe
C0 Y A 0.05fF
C1 Y VDD 0.17fF
C2 A VDD 0.18fF
.ends
```

How to make a symbol for the extracted netlist

Example of CMOS inverter



```
type=primitive  
format="@name @@A @@Y @@VDD @@GND @prefix"  
template="name=x1 prefix=TOP"  
extra="prefix"  
highlight=true
```

Add "@@" prefix to the pins.
Now the symbol works.

How to make a symbol for the extracted netlist (2)

Open the symbol's property and set like this (key binding: Q)

```
type=primitive  
format="@name [pins in netlist] @prefix"  
template="name=x1 [Pins to specify net in property] prefix=[title of subckt]"  
extra=" [Pins to specify net in property] prefix"  
highlight=true
```

See http://repo.hu/projects/xschem/xschem_man/symbol_property_syntax.html

How to make a symbol for the extracted netlist (2)

Remove VDD and GND pins from the symbol and describe them in the property

```
type=primitive  
format="@name @@A @@Y @VDD @GND @prefix"  
template="name=x1 VDD=VDD GND=0 prefix=TOP"  
extra="VDD GND prefix"  
highlight=true
```

To specify a net of pin in the property, add "@" prefix in **format**,
add pins in **extra** and add default valueds in **template**.

How to make a symbol for the extracted netlist (2)

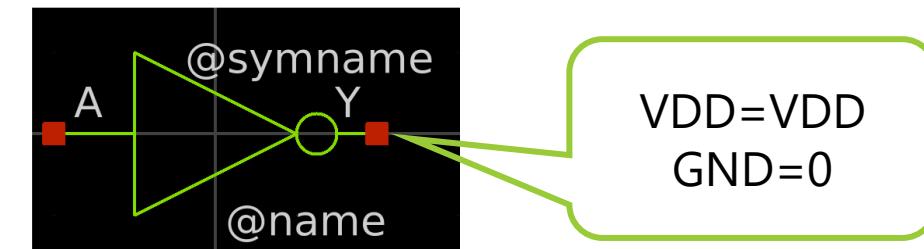
inv.sym

Remove VDD and GND pins from the symbol and describe them in the property

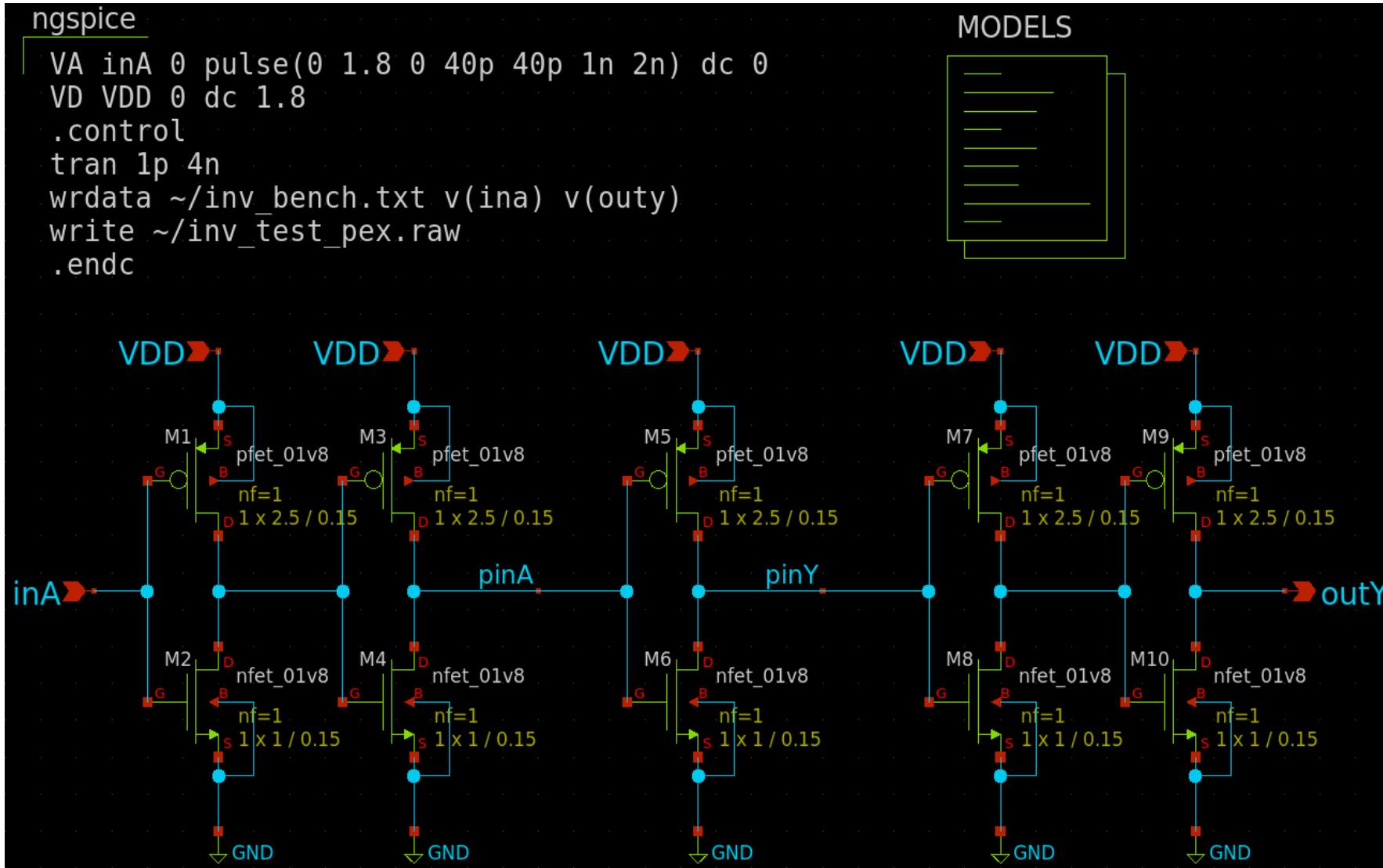
```
type=primitive  
format="@name @@A @@Y @VDD @GND @prefix"  
template="name=x1 VDD=VDD GND=0 prefix=TOP"  
extra="VDD GND prefix"  
highlight=true
```



easier to see!



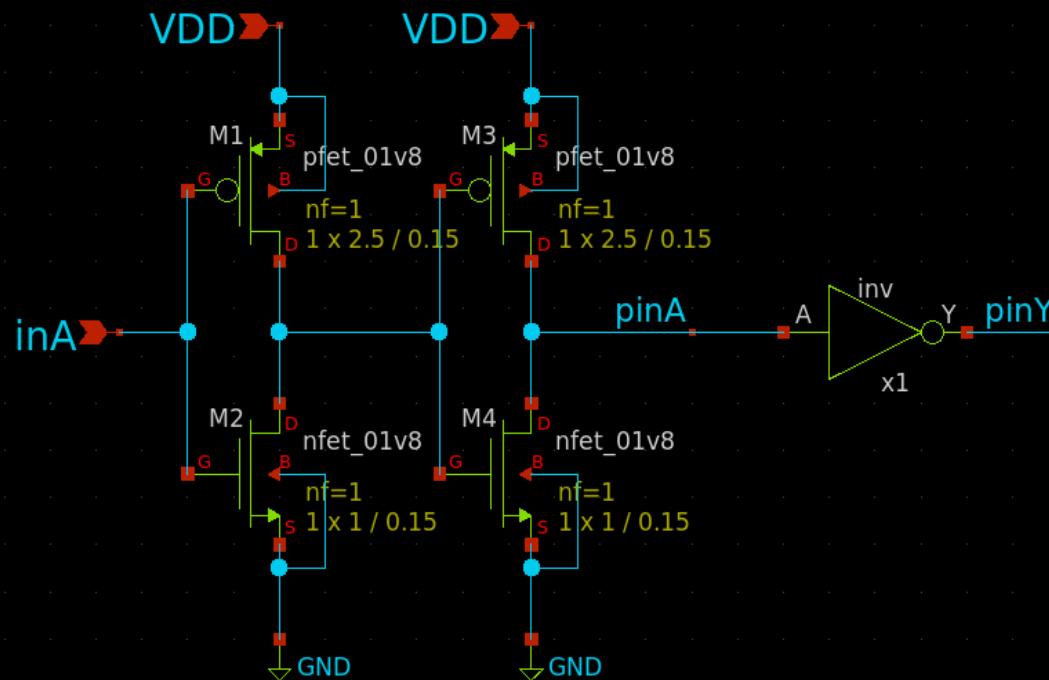
Test bench modification (before)



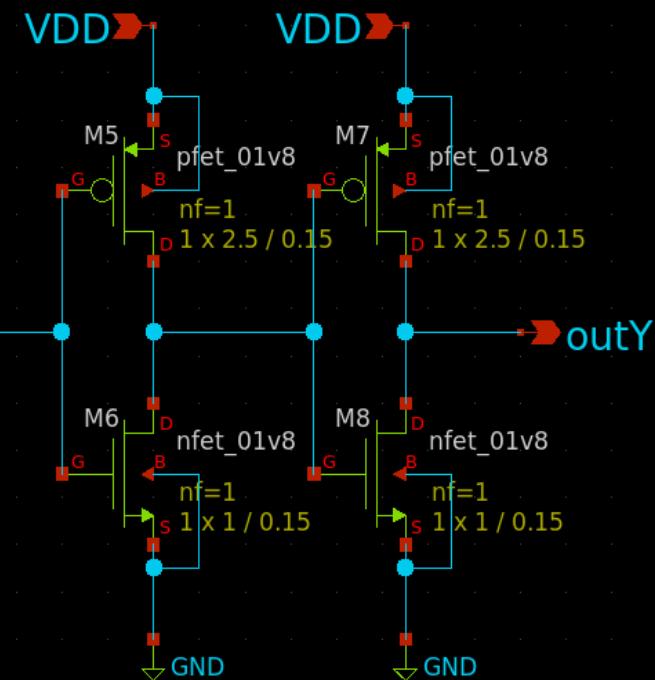
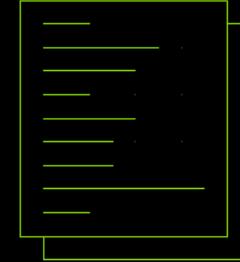
Test bench modification (after)

inv_pex.sch

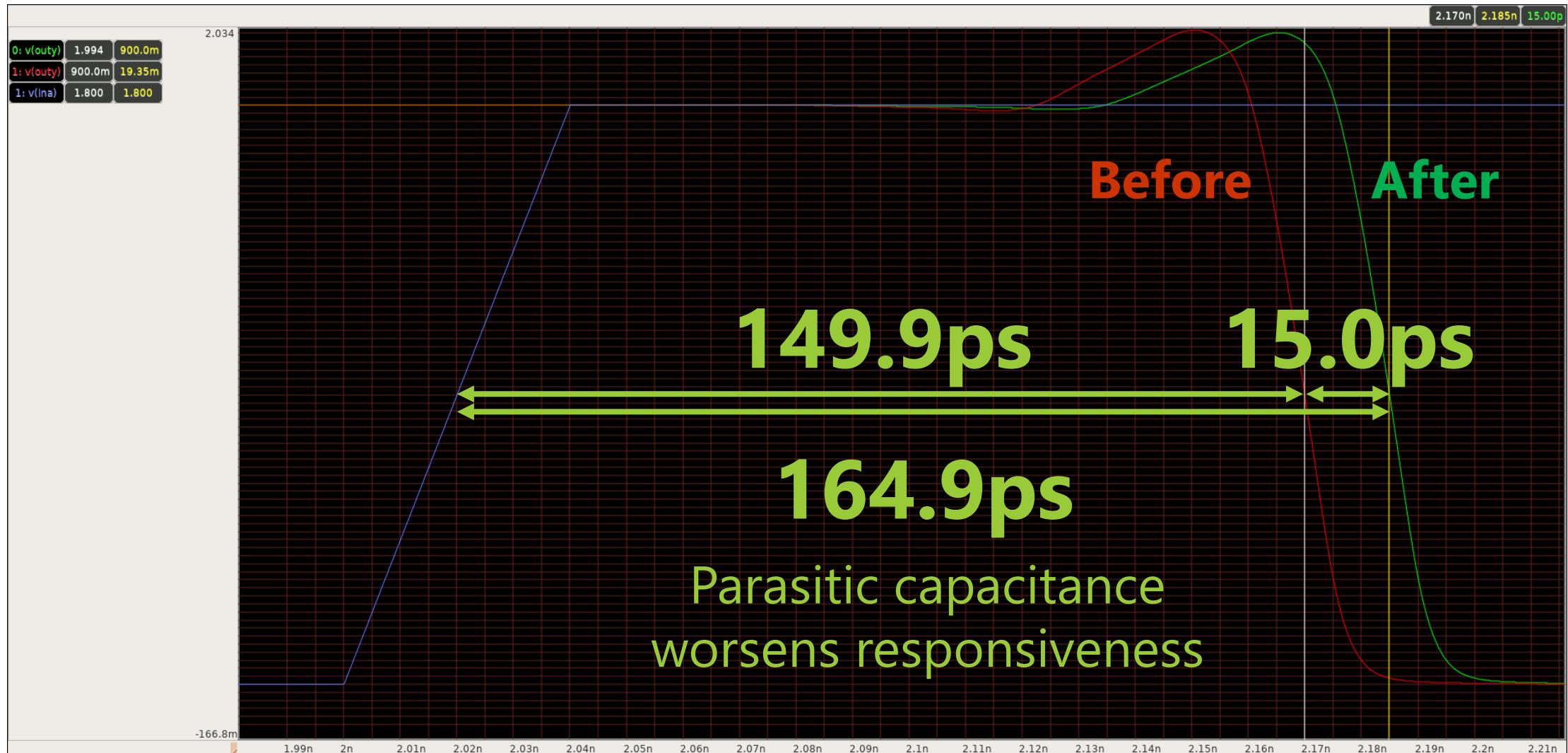
```
ngspice_
VA inA 0 pulse(0 1.8 0 40p 40p 1n 2n) dc 0
VD VDD 0 dc 1.8
.include ~/TOP_pex_extracted.spice
.control
.tran 1p 4n
.wrdata ~/inv_bench.txt v(ina) v(outy)
.write ~/inv_test_pex.raw
.endc
```



MODELS



Simulation results



Tips

Tips

- Multi finger and Mismatch simulation
- Monte Carlo simulation
- Analog switch and Pass Transistor Logic
- D latch and D flip-flop

Multi finger

Layout Techniques

FO of buffers in standard cells

buf_2 1→2

buf_4 1→4

buf_8 3→8

clkbuf8 2→8

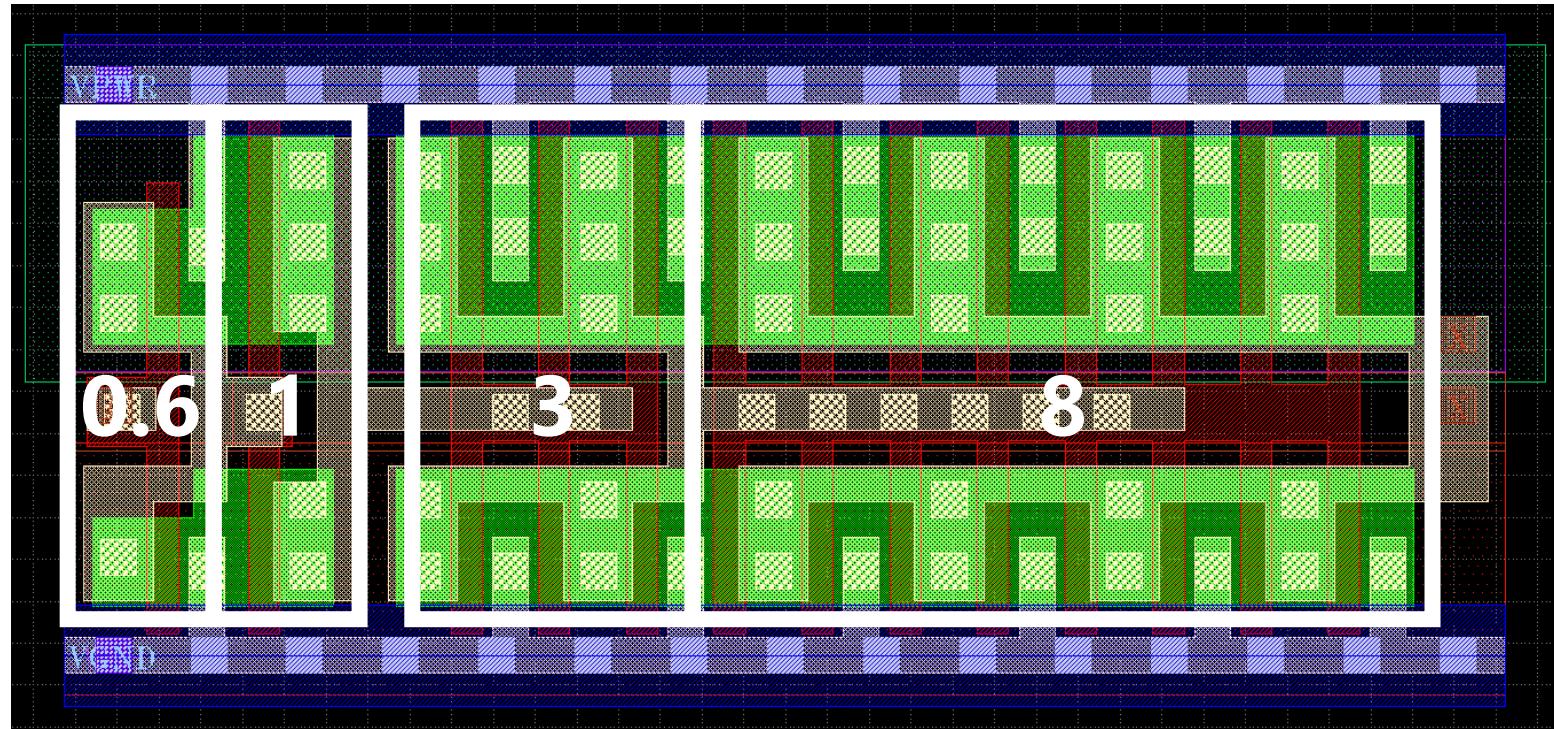
bufbuf8 0.6→1→3→8

buf_12 4→12

buf_16 6→16

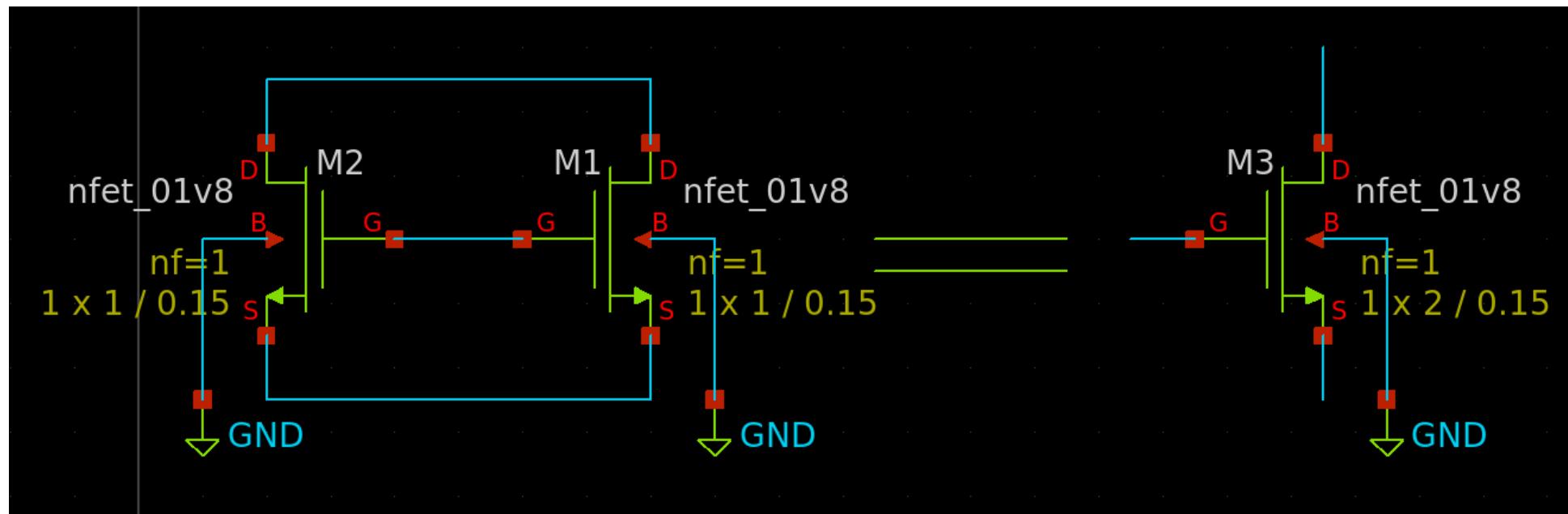
clkbuf16 4→16

bufbuf16 1→3→6→16



3 stage inverter

Two transistors with the same node and the same size equals a transistor with twice W

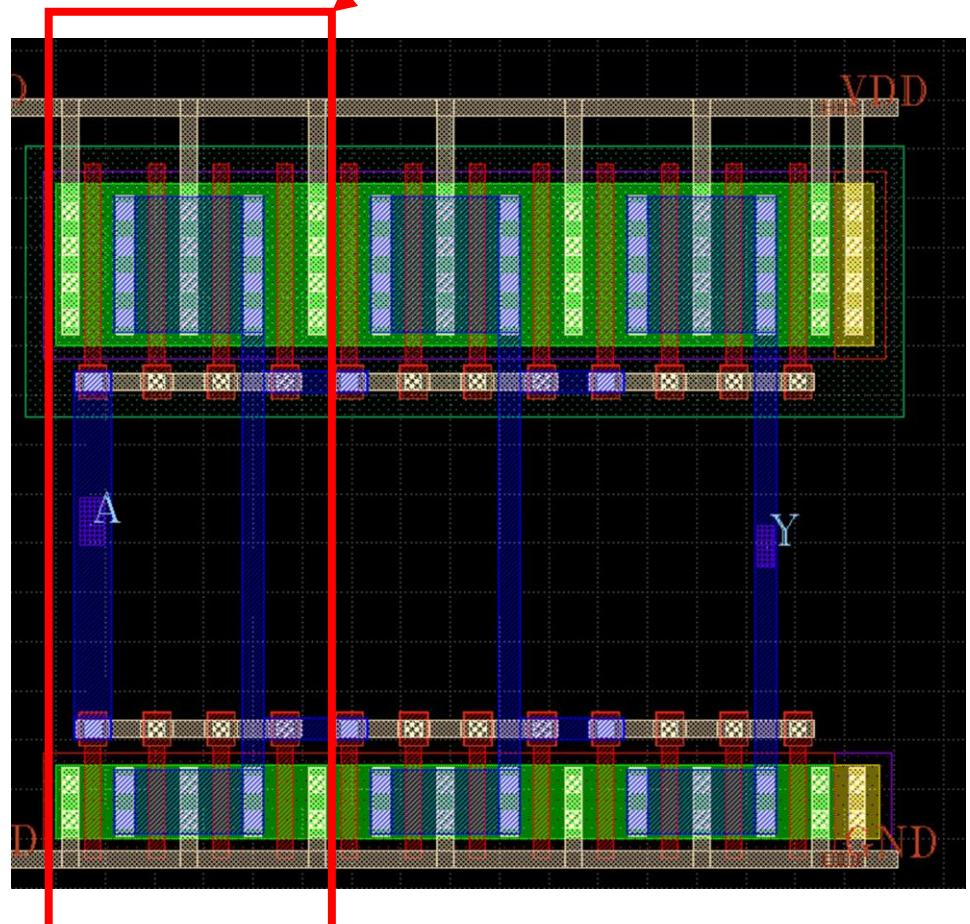
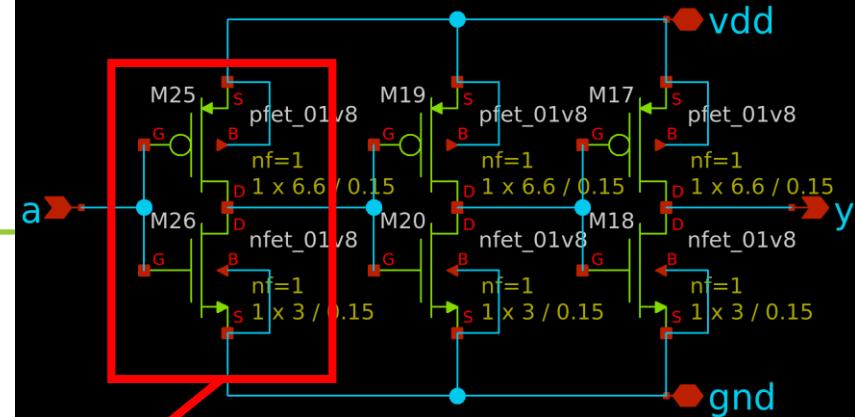


3 stage inverter

6.6 / 3 divided into $(1.65 / 0.75) * 4$

- $(6.6/3) * 3 \rightarrow (1.65 / 0.75) * 12$
- **Flatten** before edit

LVS will pass without schematic modification.

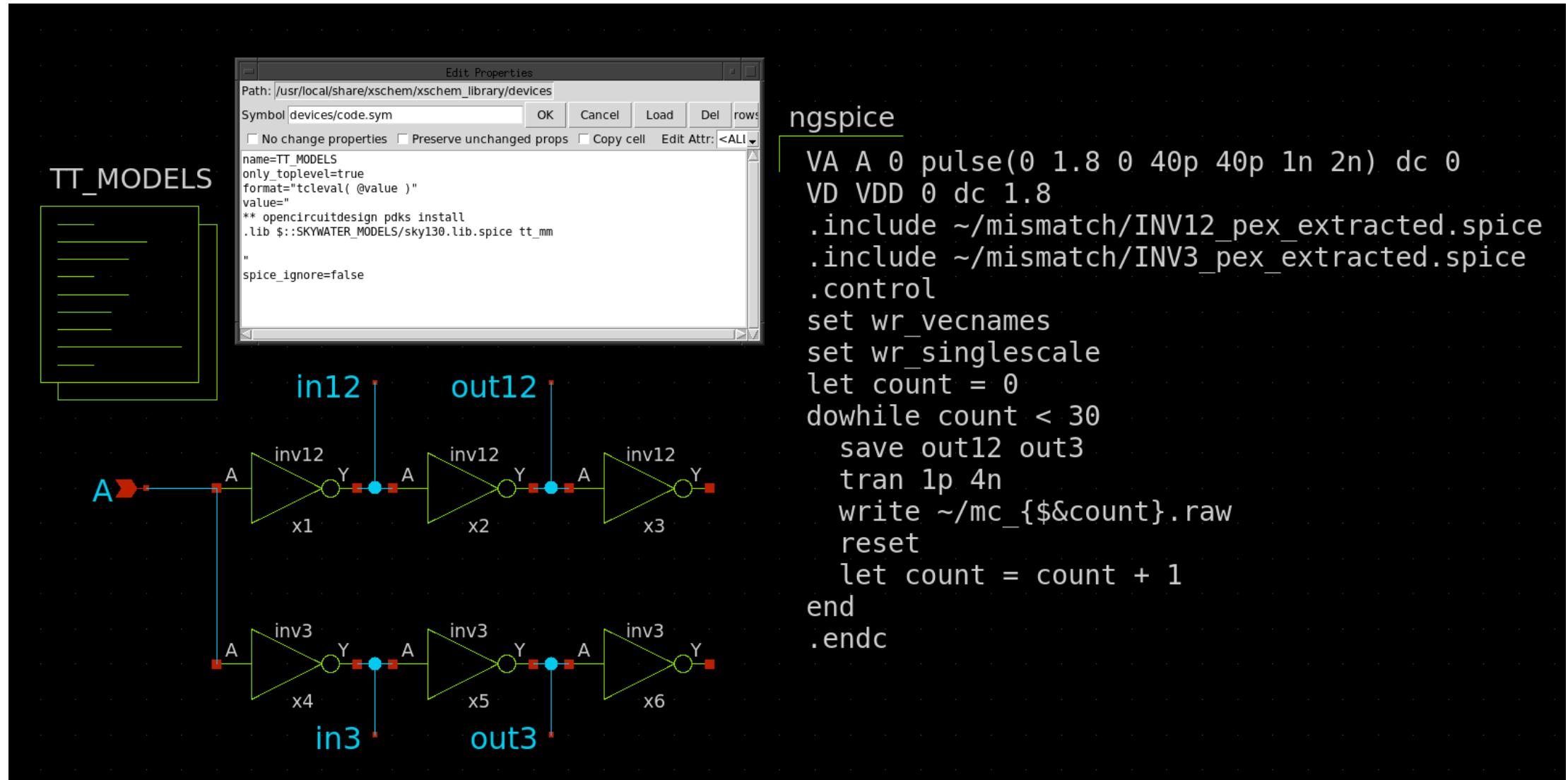


Mismatch simulation

- Multi-finger has the effect of reducing mismatch (variation between elements)
- Mismatch simulations are available to look at element-to-element variation.
- Assign a random V_{th} to each element
- Append _mm to the end of the corner such as "tt_mm"

Test bench example

mm_sim.sch

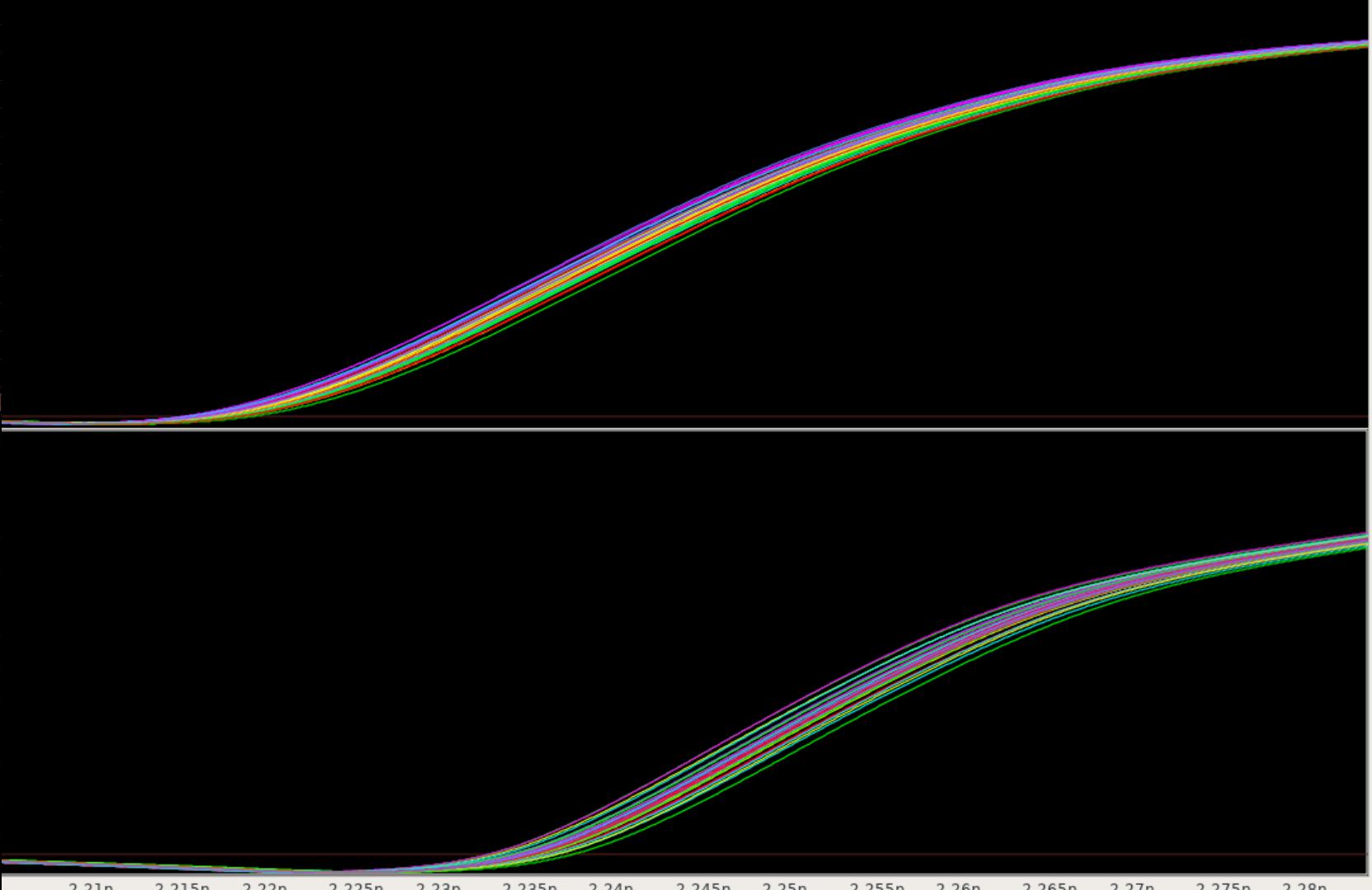
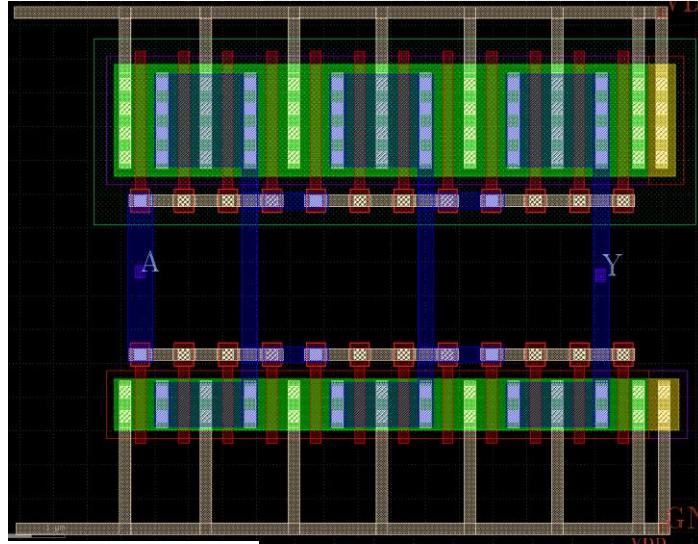


ngspice

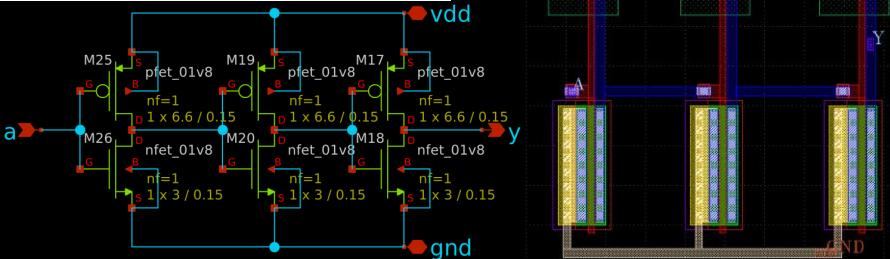
```
VA A 0 pulse(0 1.8 0 40p 40p 1n 2n) dc 0
VD VDD 0 dc 1.8
.include ~/mismatch/INV12_pex_extracted.spice
.include ~/mismatch/INV3_pex_extracted.spice
.control
set wr_vecnames
set wr_singlescale
let count = 0
dowhile count < 30
    save out12 out3
    tran 1p 4n
    write ~/mc_{$&count}.raw
    reset
    let count = count + 1
end
.endc
```

Variation in 30 runs with the same seed

inv12.gds, inv12.sym
inv3.gds, inv3.sym



Both LVS schematics
are identical.

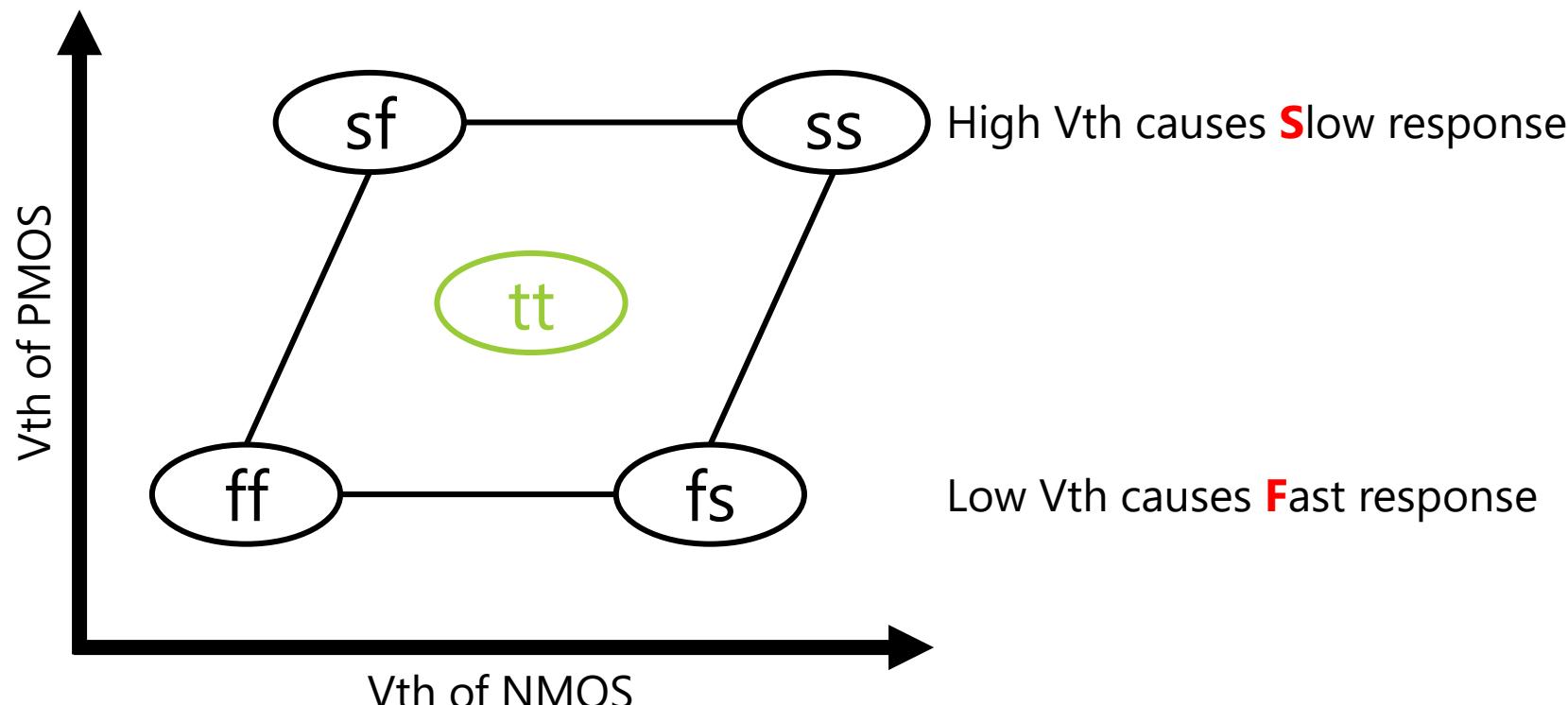


Monte Carlo simulation

Simulation by random characteristics

Monte Carlo simulation

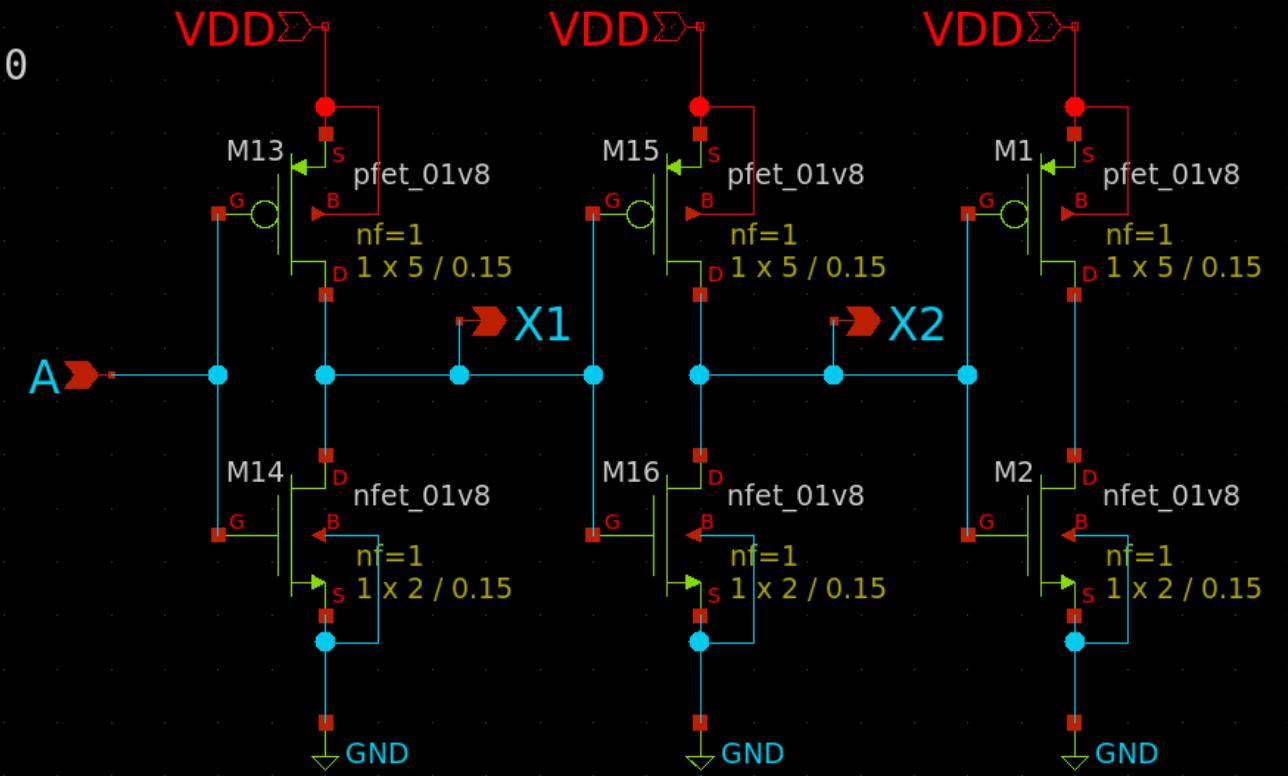
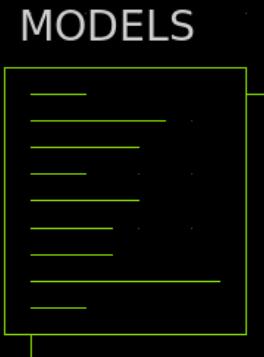
- Usually, we specify the corner of model in the testbench.
- Monte Carlo simulation runs multiple simulations with random characteristics
- In SKY130, “mc” corner is a Monte Carlo method model.



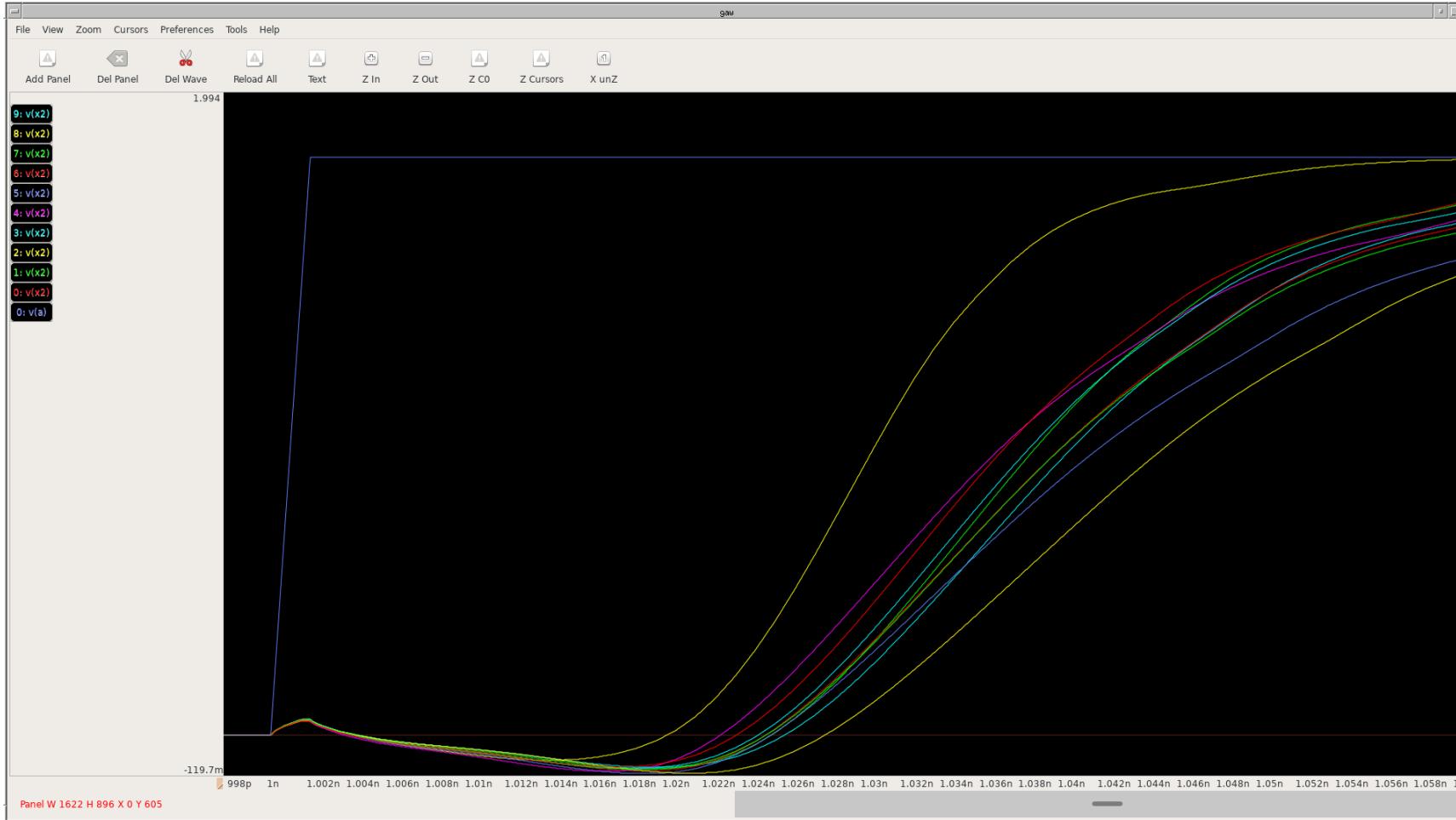
Xschem testbench (mc, 10 times, L=0.15μm)

test_inv_mc.sch

```
ngspice
VA A 0 pulse(0 1.8 0 2p 2p 0.5n 1n) dc 0
VD VDD 0 dc 1.8
.control
set wr_vecnames
set wr_singlescale
let count=0
dowhile count < 10
tran 1p 2n
write ~/mc_{$&count}.raw
reset
let count = count + 1
end
.endc
```



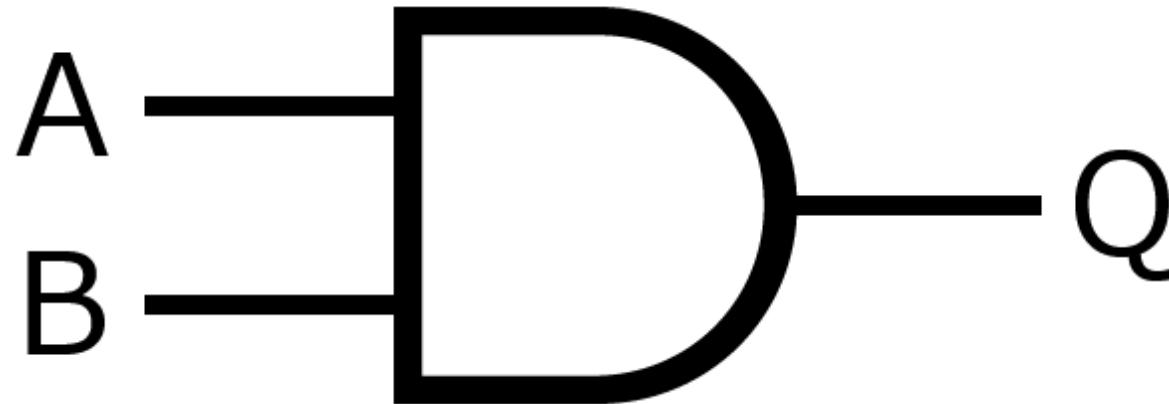
Simulation results (mc)



Pass transistor logic

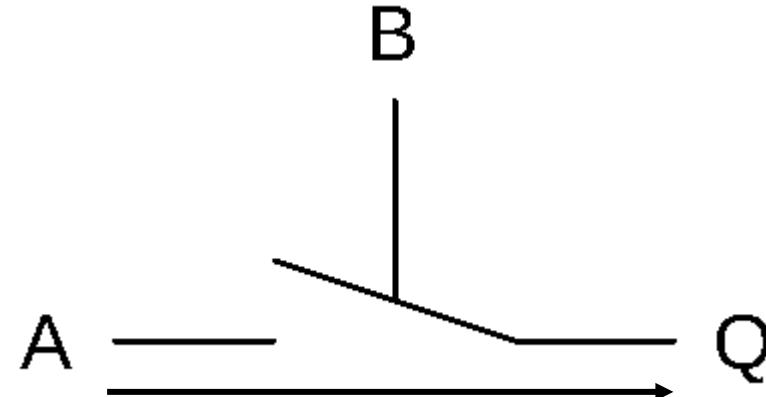
Transmission gate, Analog switch

2-input AND

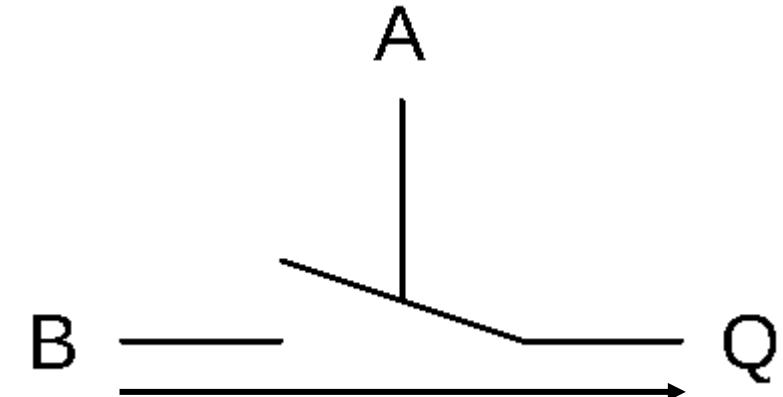


A	B	Q
L	L	L
L	H	L
H	L	L
H	H	H

2-input AND as one-way switch



B	A	Q
L	L or H	L
H	L	L
H	H	H

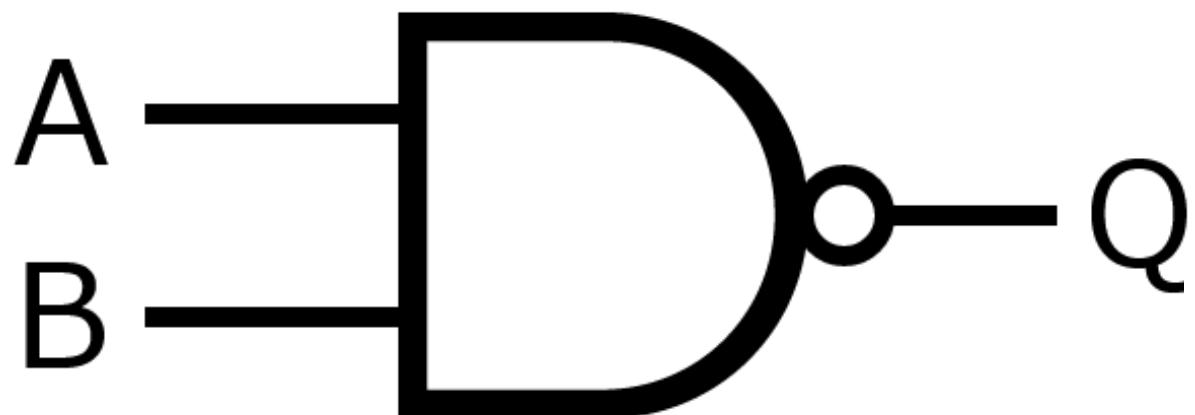


A	B	Q
L	L or H	L
H	L	L
H	H	H

When OFF, the switch is in a circuit-breaking state in the real switch,
but in the AND gate, it is connected to GND.

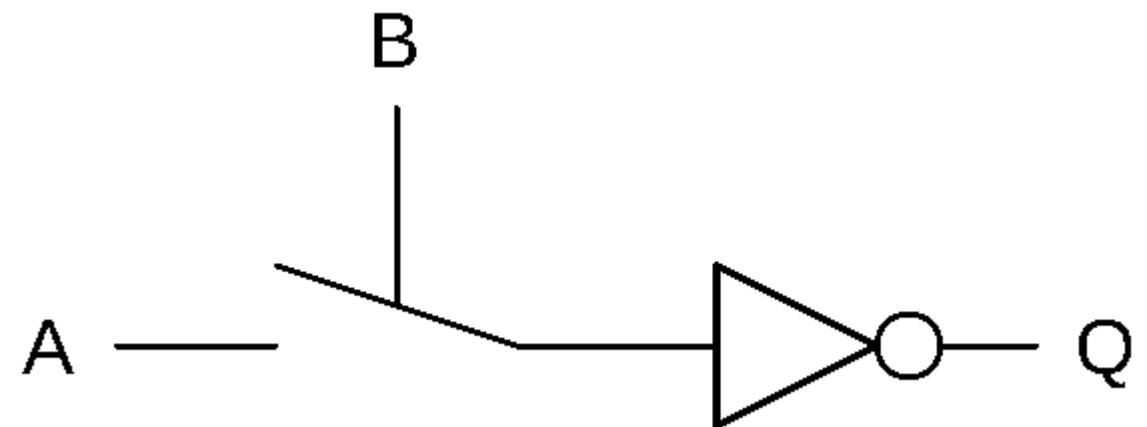
2-input NAND

Inverted output of 2-input AND

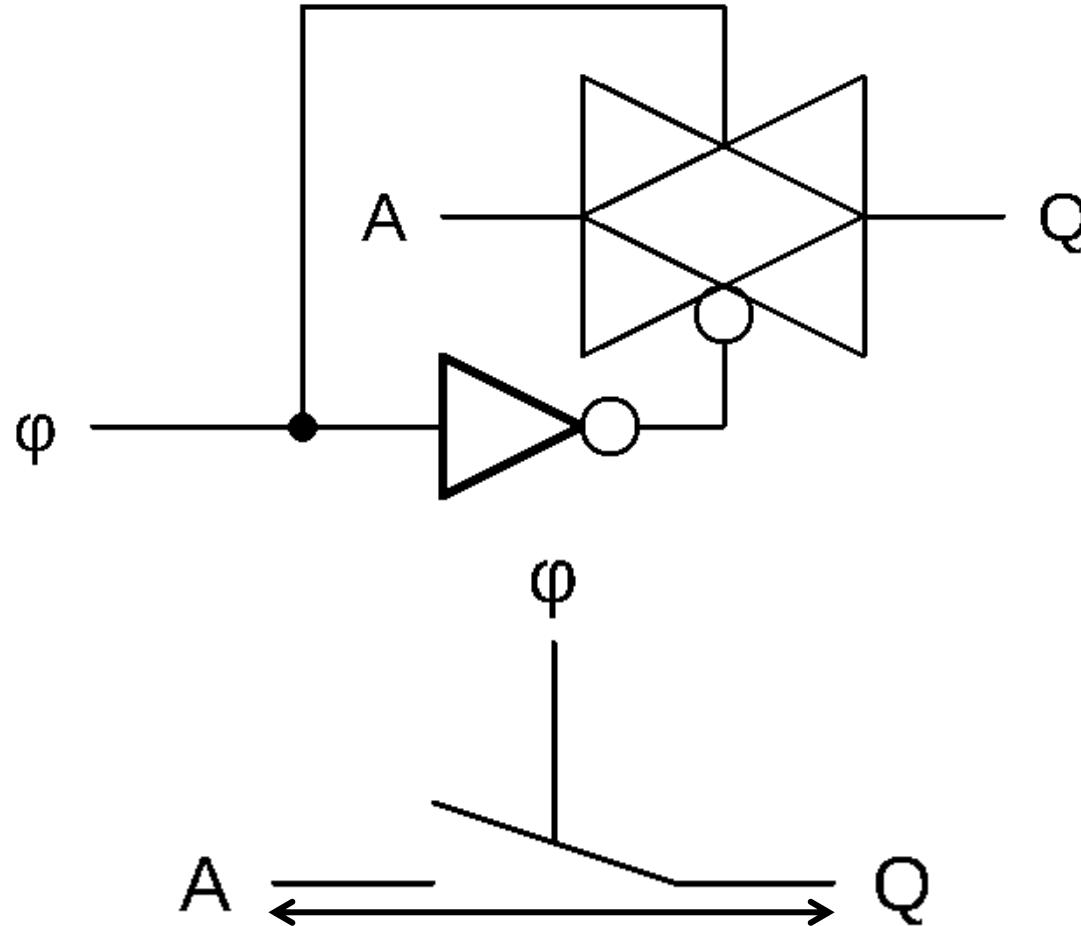


A	B	Q
L	L	H
L	H	H
H	L	H
H	H	L

Inverted NAND is treated as AND in CMOS circuits.

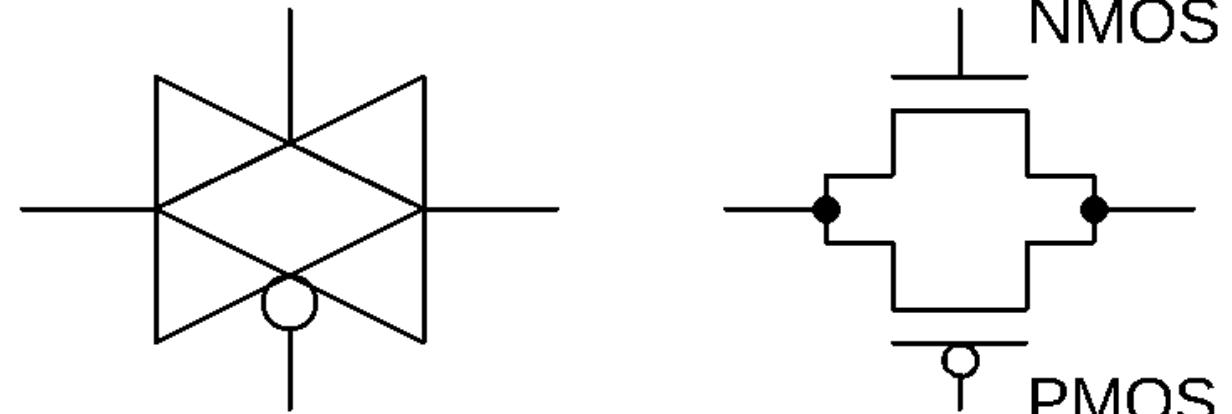


Analog switch (transmission gate + inverter)

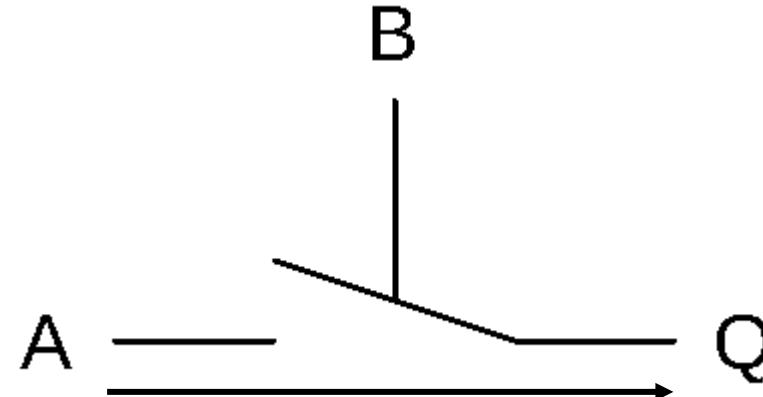


φ	A	Q
L		High resistance (Hi-Z)
H		<u>Almost conduction (PASS)</u>

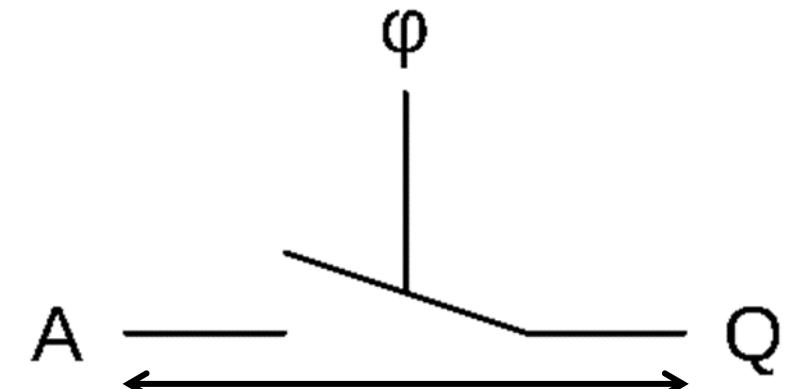
On-resistance (R_{ds})



2-input AND (left) vs. analog switch (right)



B	A	Q
L	L or H	L
H	L	L
H	H	H

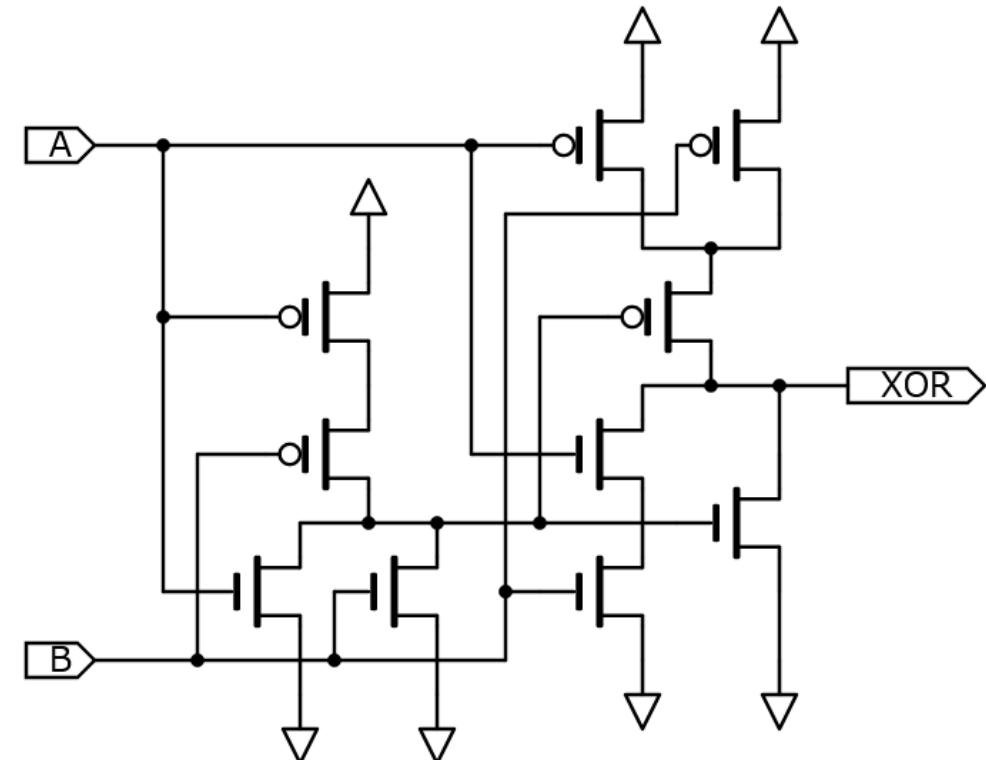
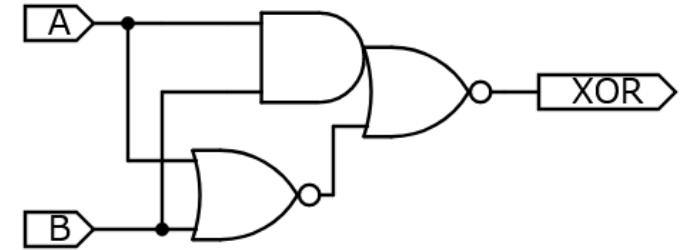


φ	A	Q
L		Hi-Z
H		PASS

In analog switches, when φ = Low, the output is almost insulated (high resistance, Hi-Z)
In AND gate, when B=Low, it is connected to GND (Low)

2-input XOR

Logic circuits prohibit short-circuiting of outputs.
XOR design with logic circuits is a bit complicated.
(NOR (4) + AOI (6) = 10 transistors)

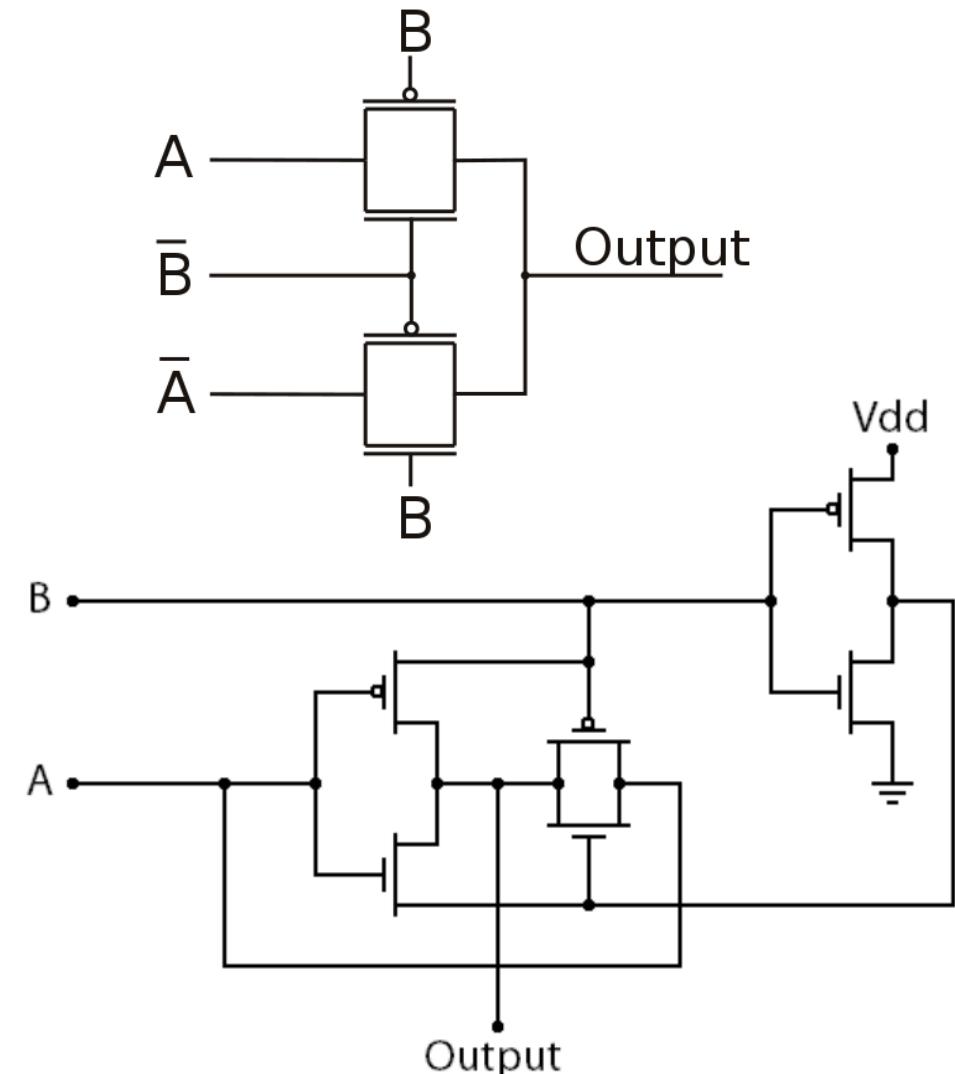


2-input XOR

Analog switch is Hi-Z at $\varphi=\text{Low}$, that is,
output can be shorted if controlled carefully.

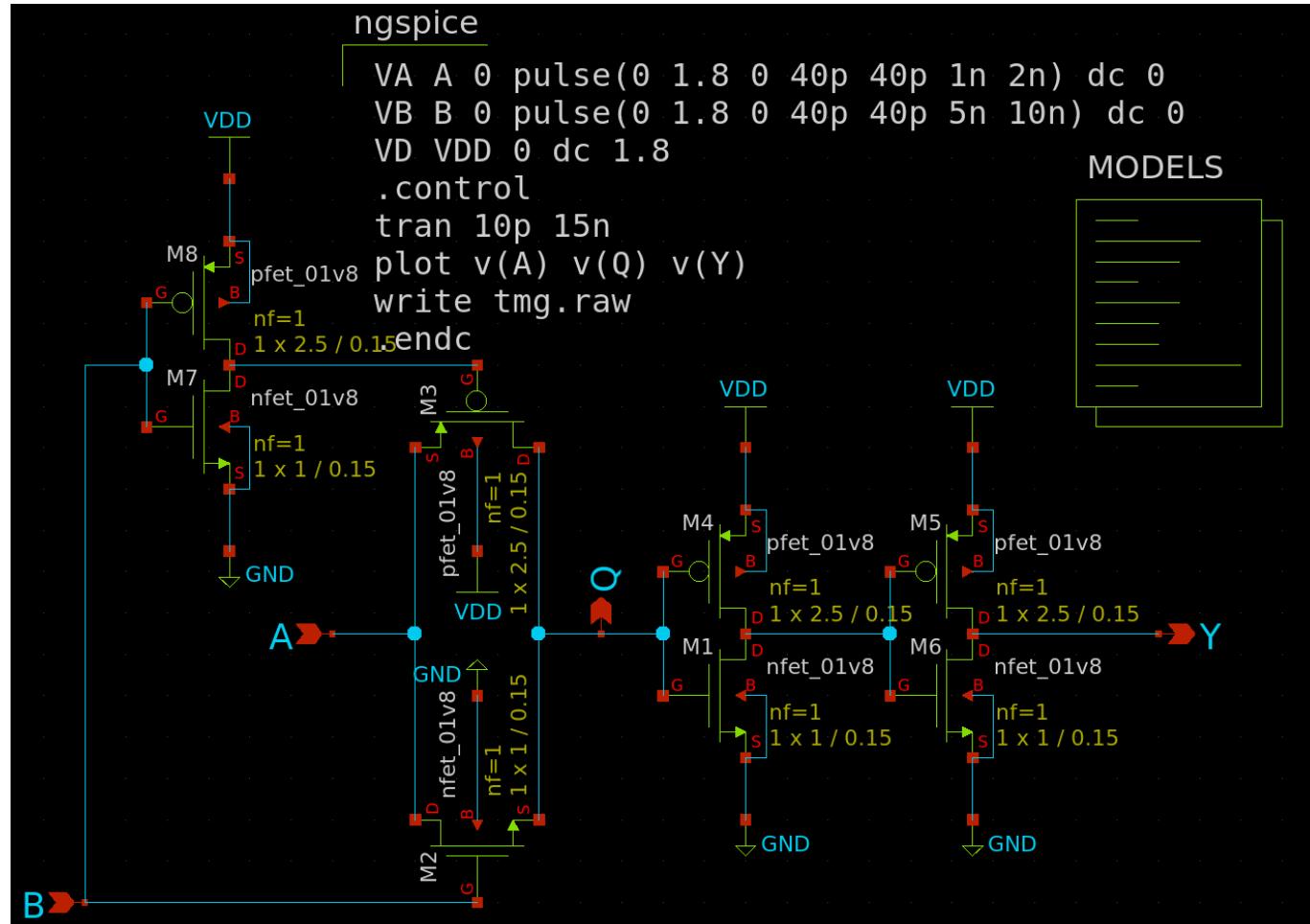
Number of transistors may be reduced.

Unintentional Hi-Z (floating) input is undesirable.
The logic output must be always High or Low.

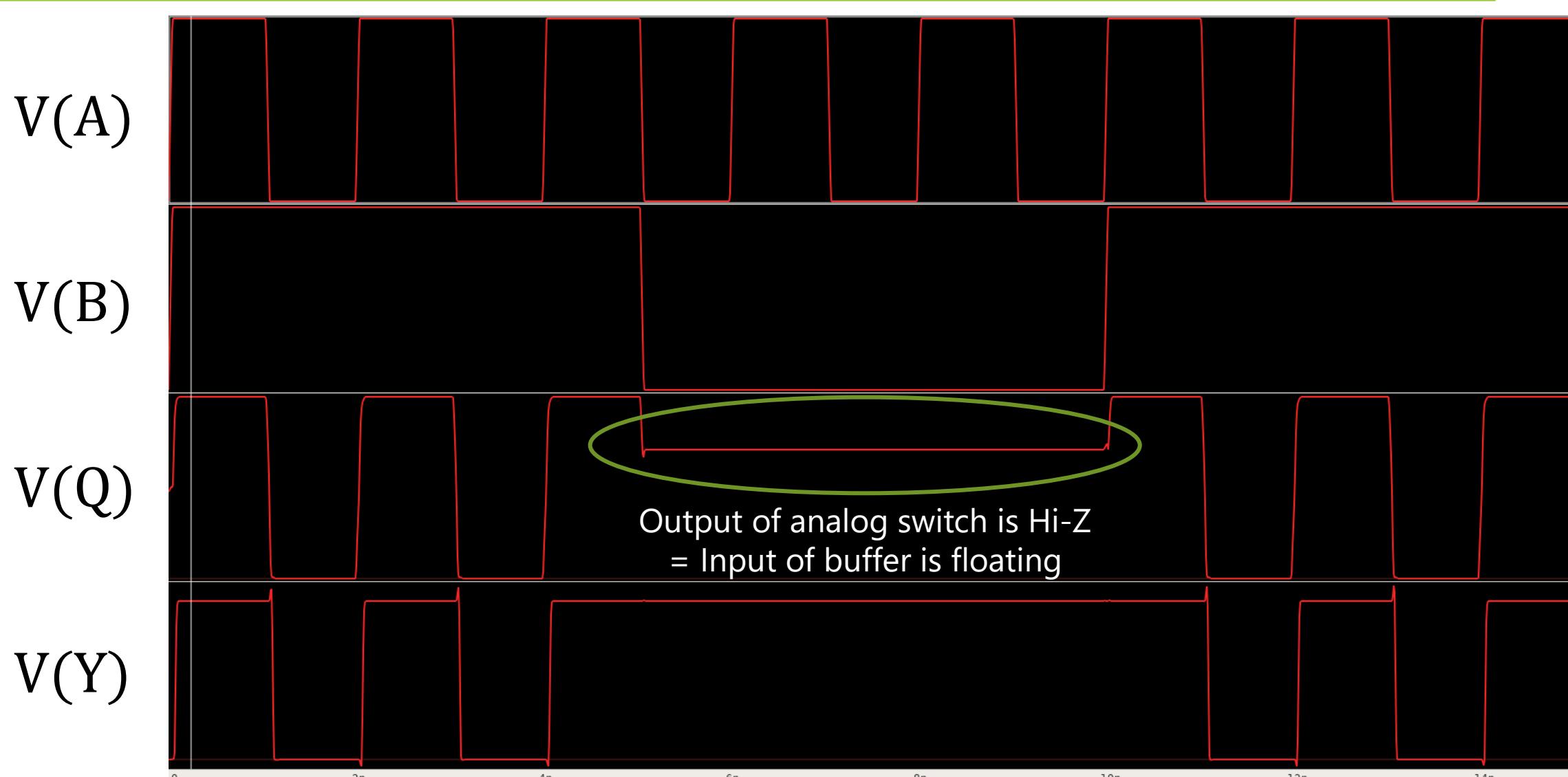


https://en.wikipedia.org/wiki/XOR_gate

Simulation of the analog switch



Simulation of the analog switch

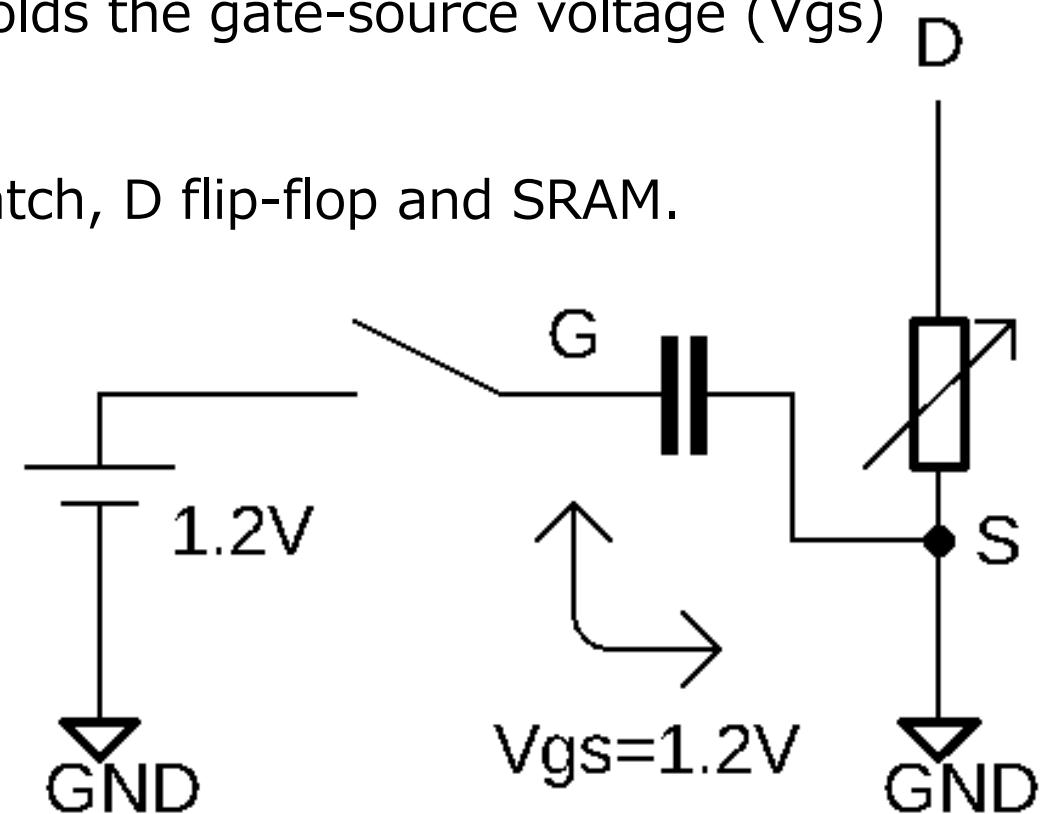


D latch and D flip-flop

Logic gates with floating gates

Floating gate

- The drain-source current (I_{ds}) is determined by the gate-source voltage (V_{gs}), but **the gate has a parasitic capacitance**.
- When the gate is in the floating state, the gate holds the gate-source voltage (V_{gs}) that was input immediately before.
- It is called a **floating gate** and is used in the D latch, D flip-flop and SRAM.

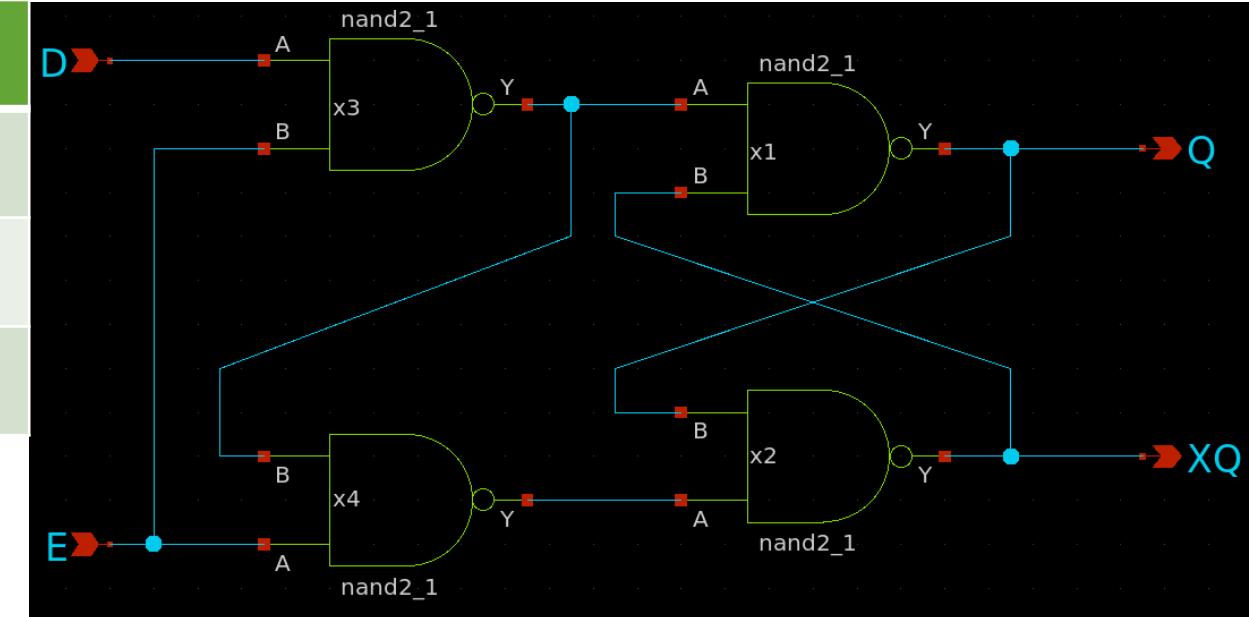


Gated D flip-flop / D latch

Holds the value of D when E goes from High to Low

- When E=High, the state of D is reflected in Q (PASS)
- E is sometimes expressed as CLK.

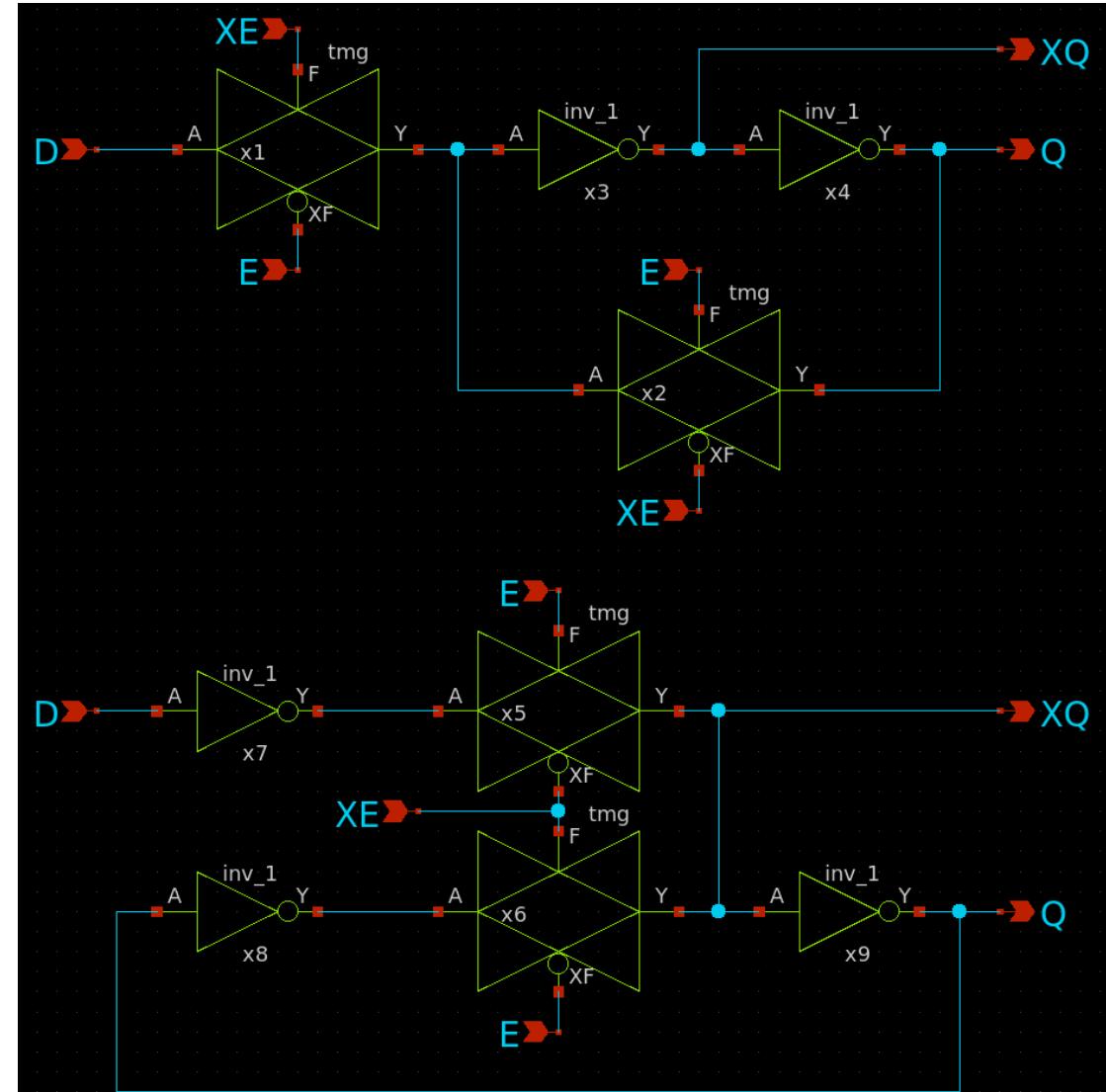
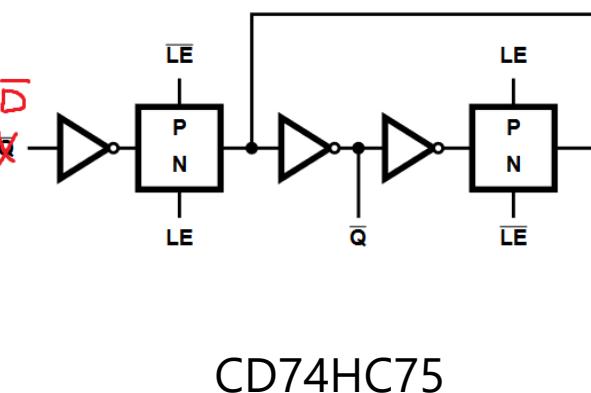
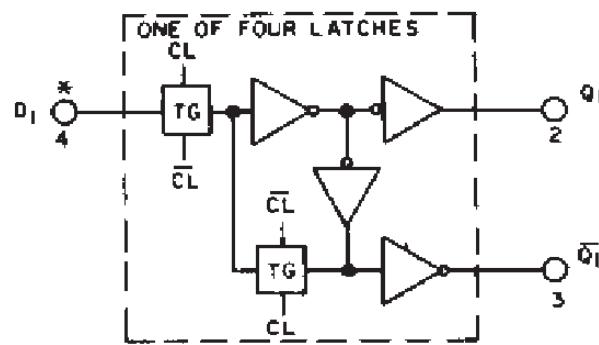
E (CLK)	D	Q
L	X	<u>Hold</u>
H	L	L
H	H	H



Gated D flip-flop / D latch

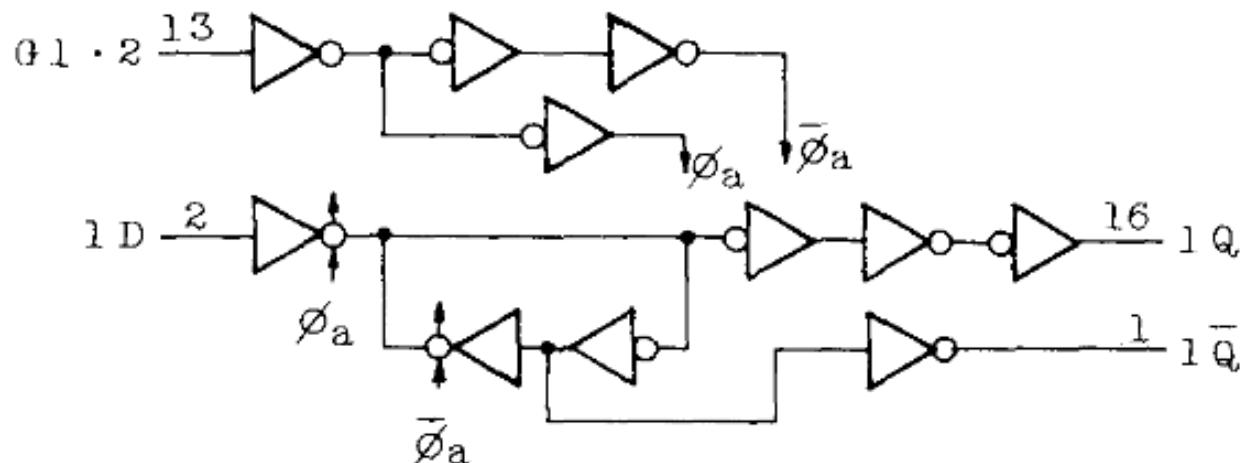
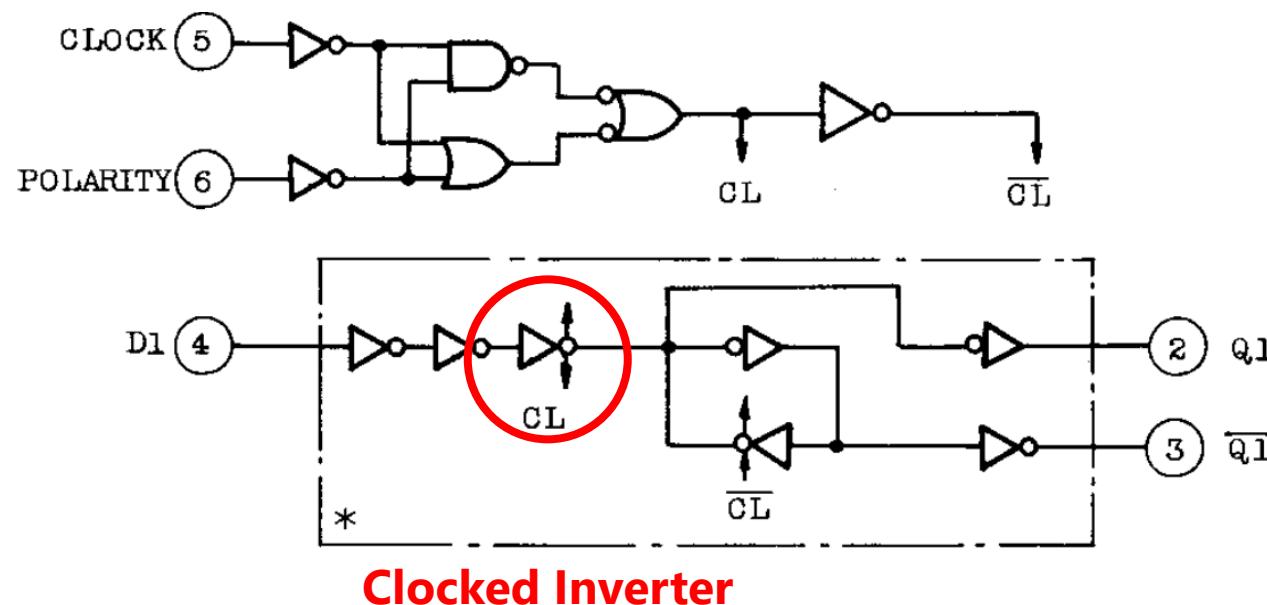
↓Need an inverter to invert E (=XE)

- Gated D flip-flop / D latch using transmission gates
- Some logic ICs are using transmission gates (TG) to make a D latch

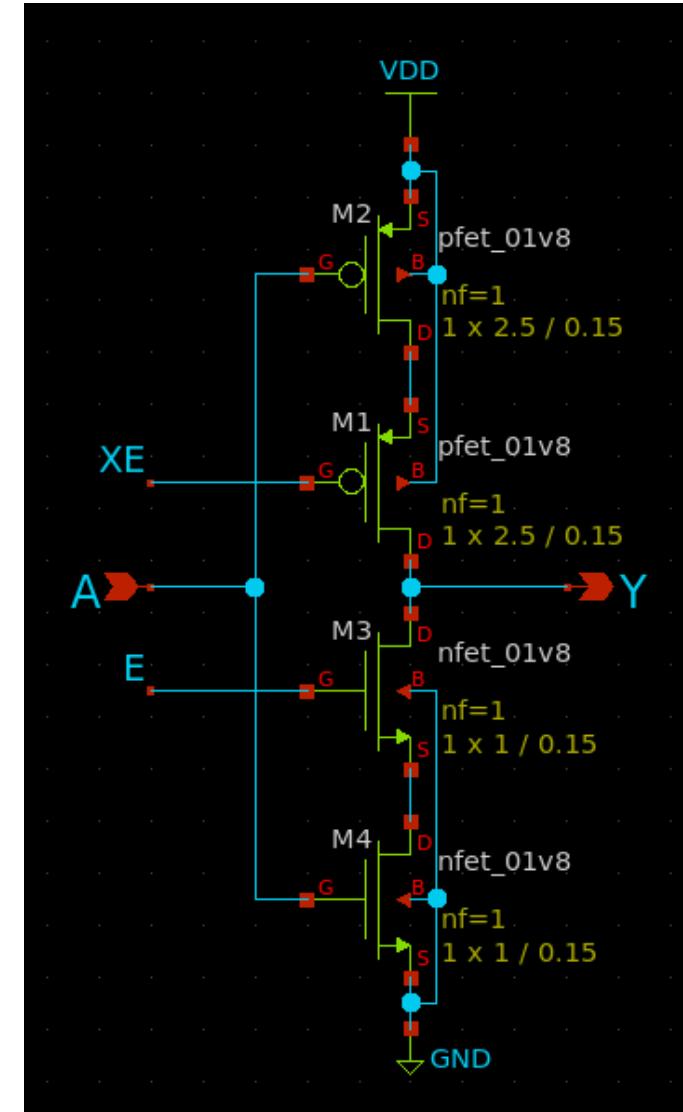
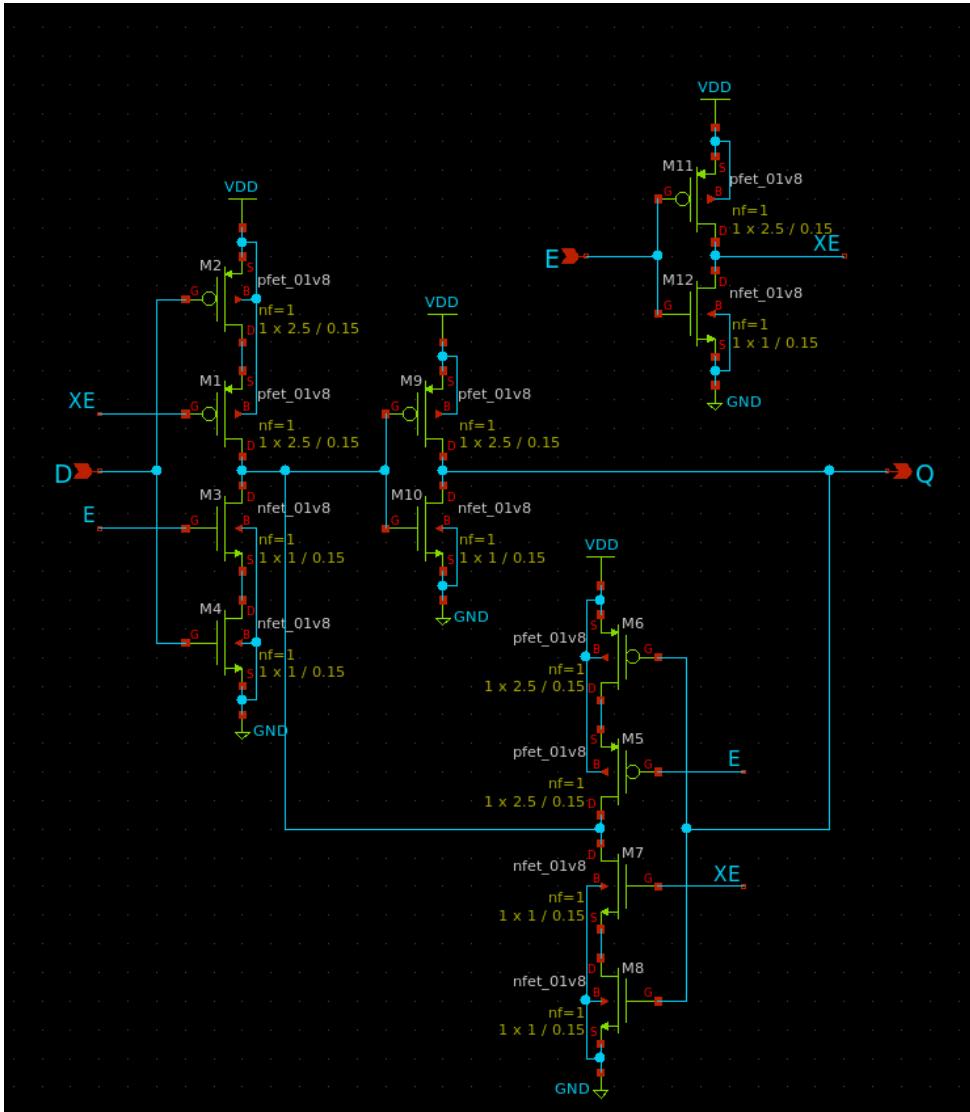


Gated D flip-flop / D latch

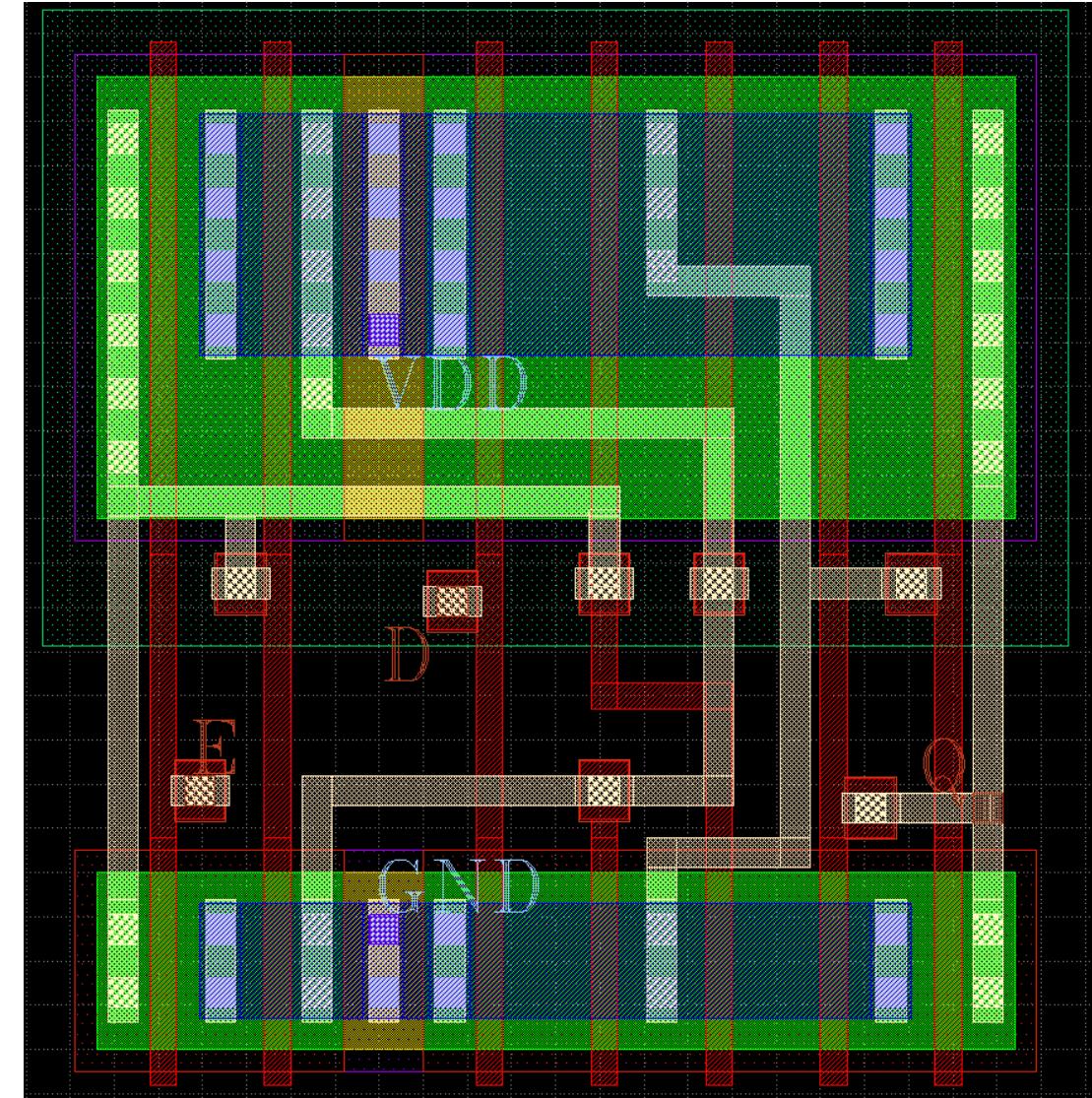
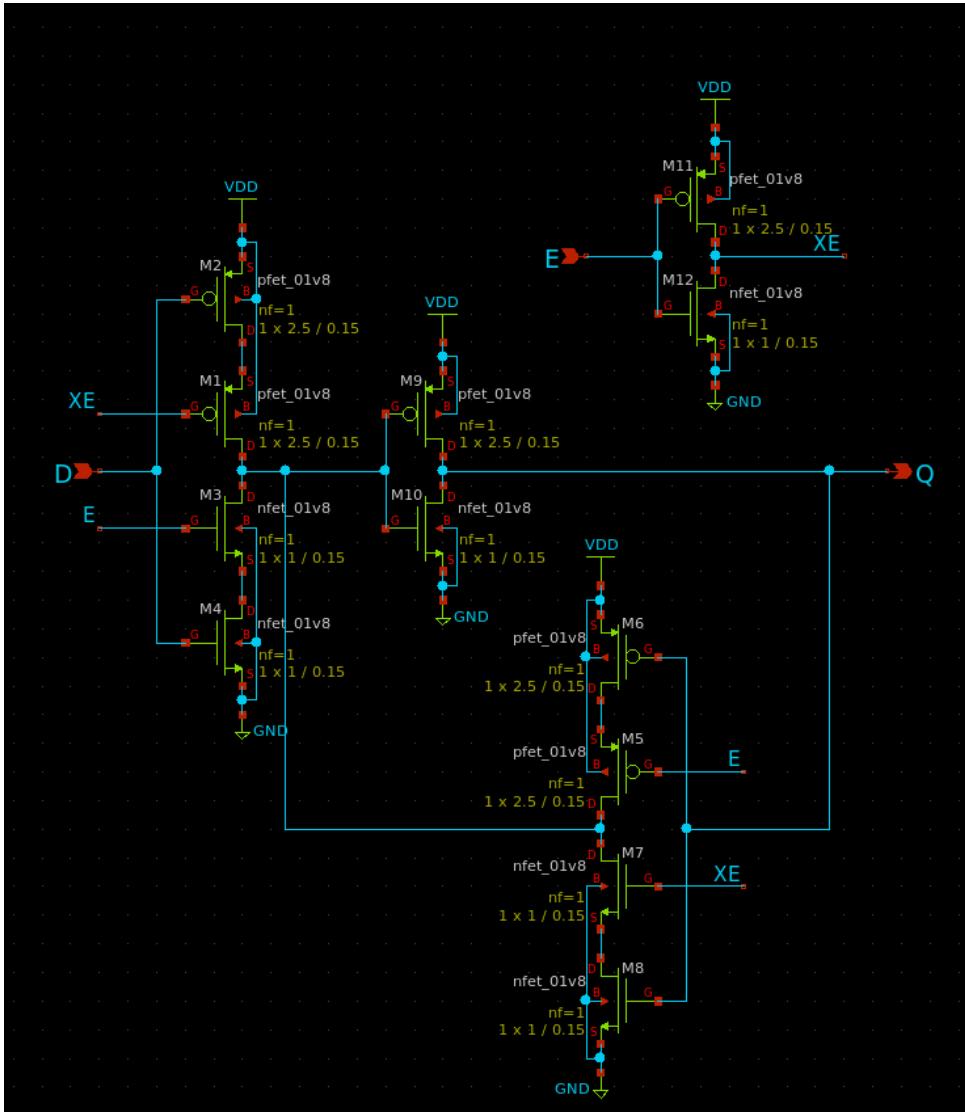
Left: TC4042B Right: TC74HC75



Example of D latch with clocked inverter

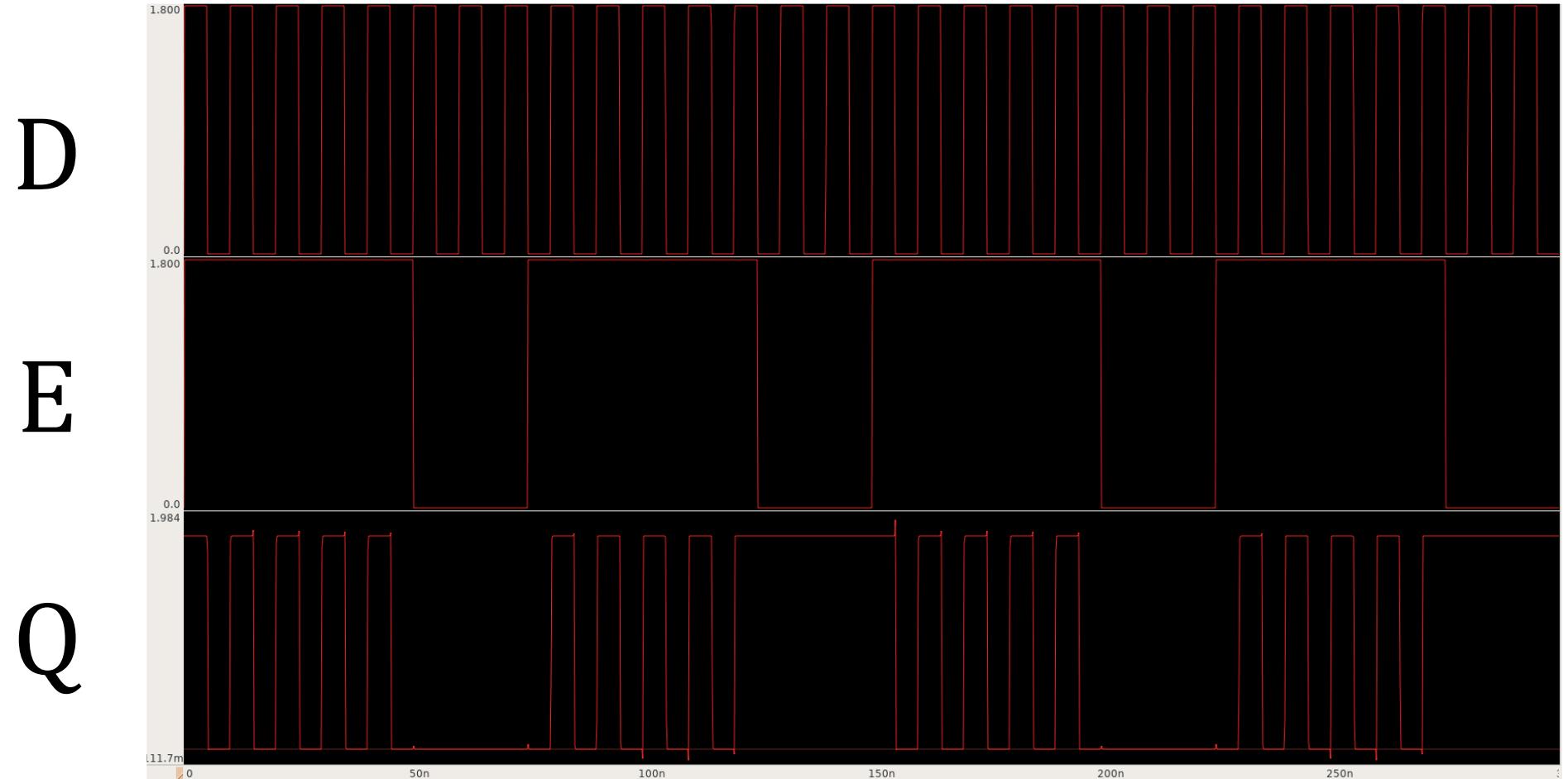


Example of D latch with clocked inverter



Waveform example of logical operation check

dLatch_bench2.sch

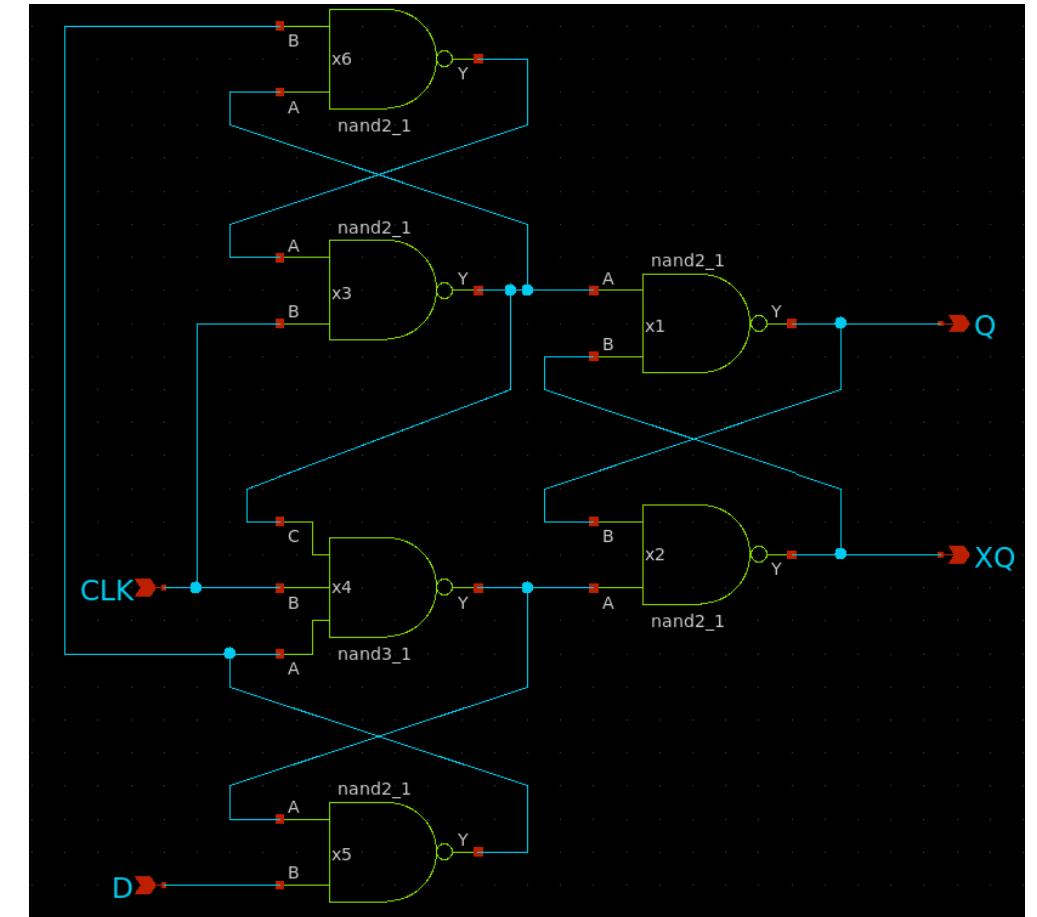


Positive-edge-triggered D flip-flop / D latch

In a D latch, when CLK(E) was High, output Q was a *pass*.

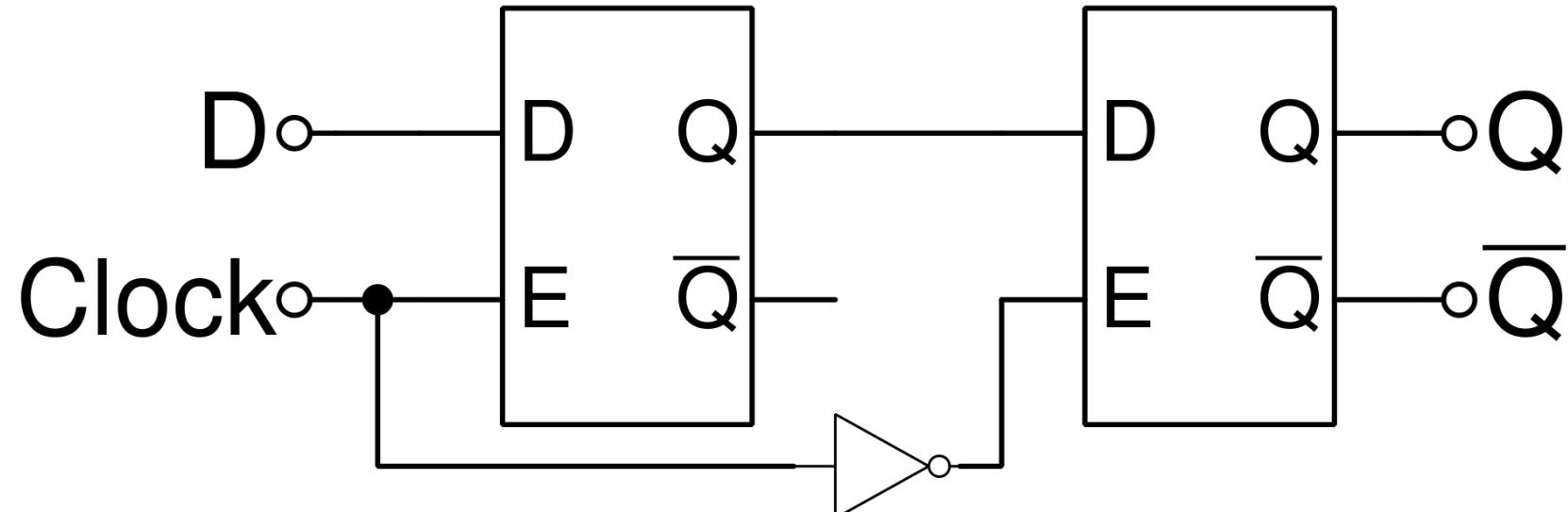
In D flip-flop, output Q changed only when CLK goes from Low to High

CLK	D	Q
L→H	L	L
L→H	H	H
otherwise	X	hold



Positive-edge-triggered D flip-flop / D flip-flop

- It can be made with two D latches.
(so-called **Master-slave edge-triggered D flip-flop**)
- Logic ICs often use two D latches to make a D flip-flop



Negative-edge-triggered D flip-flop

Positive-edge-triggered D flip-flop / D flip-flop

Left: CD4013B Right: CD74HC74

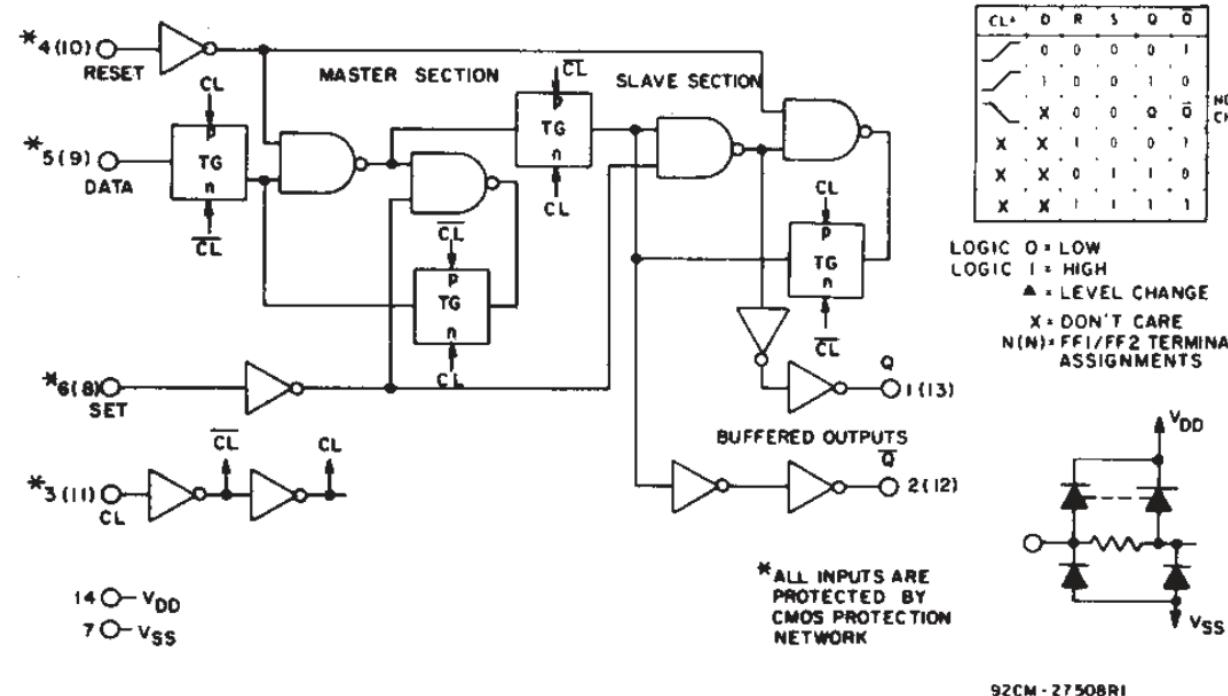
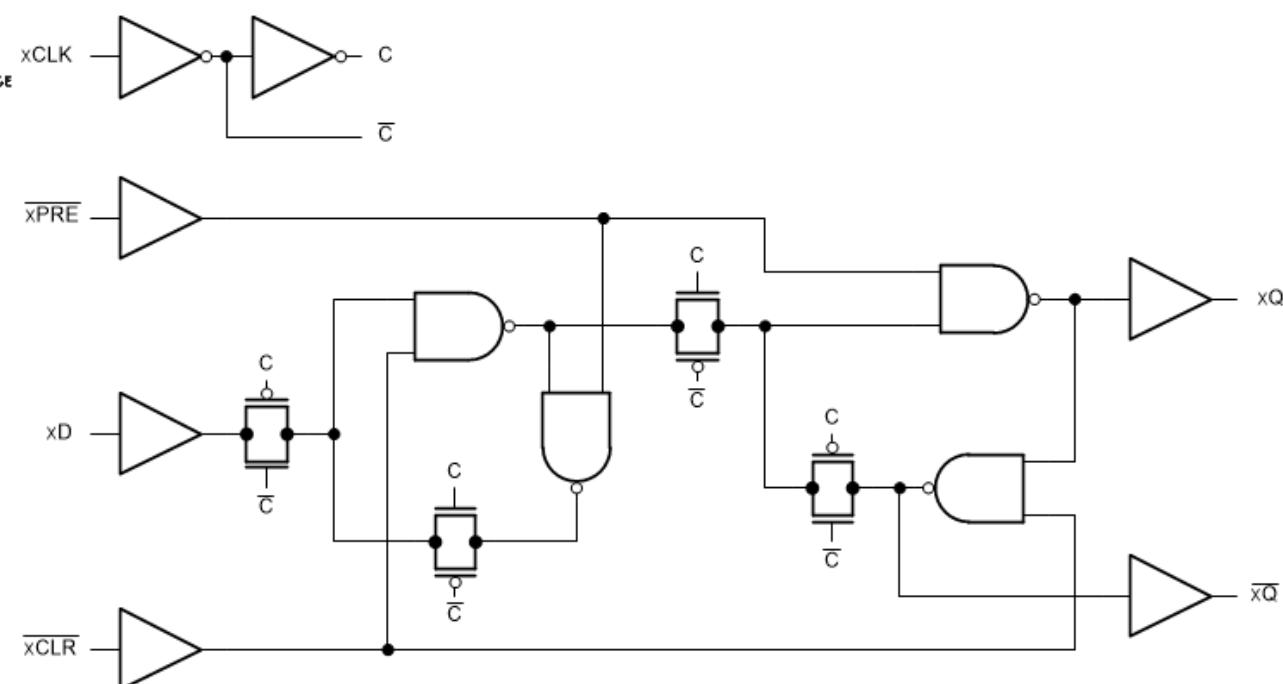


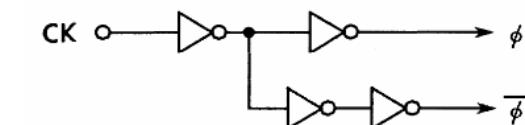
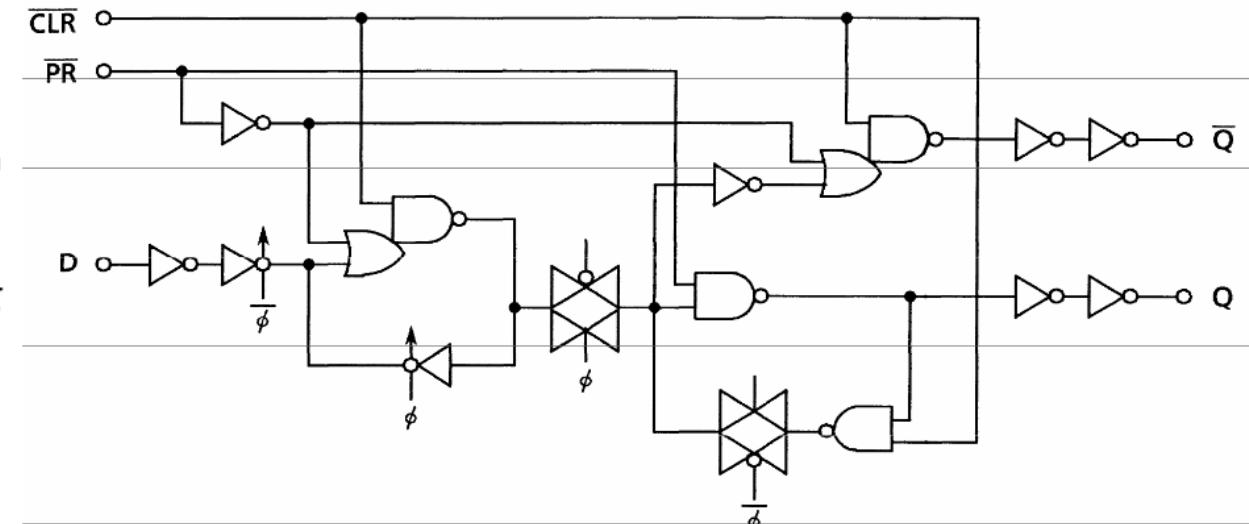
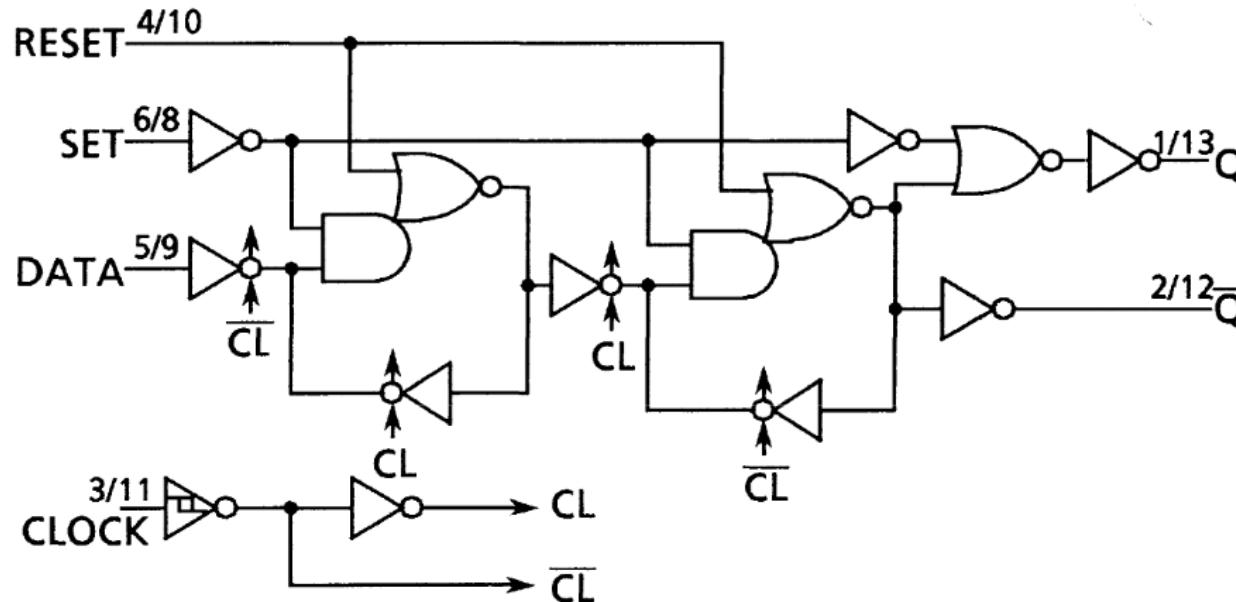
Fig. 7 Logic diagram and truth table for CD4013B (one of two identical flip-flops).



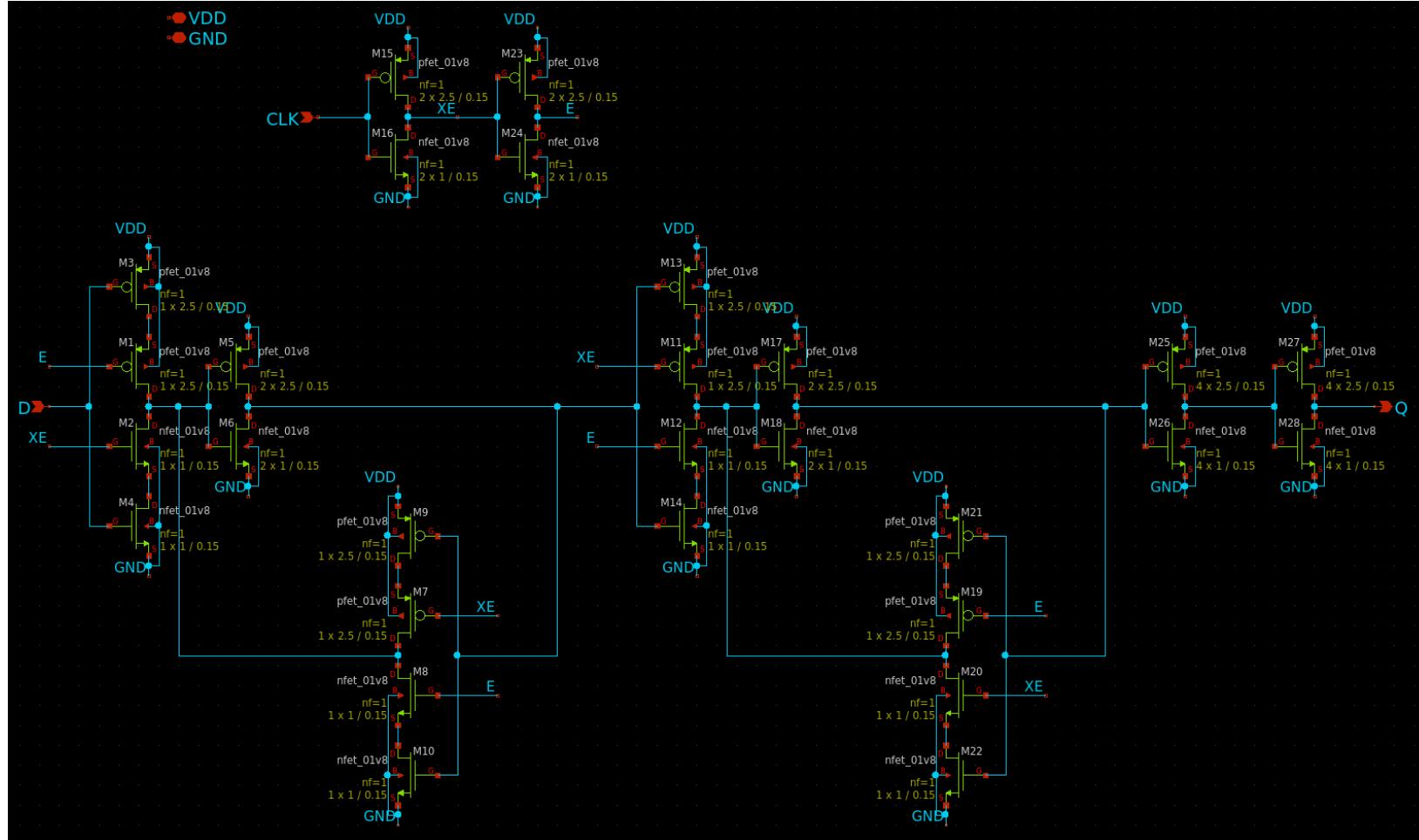
Positive-edge-triggered D flip-flop / D flip-flop

Left: TC4013BP

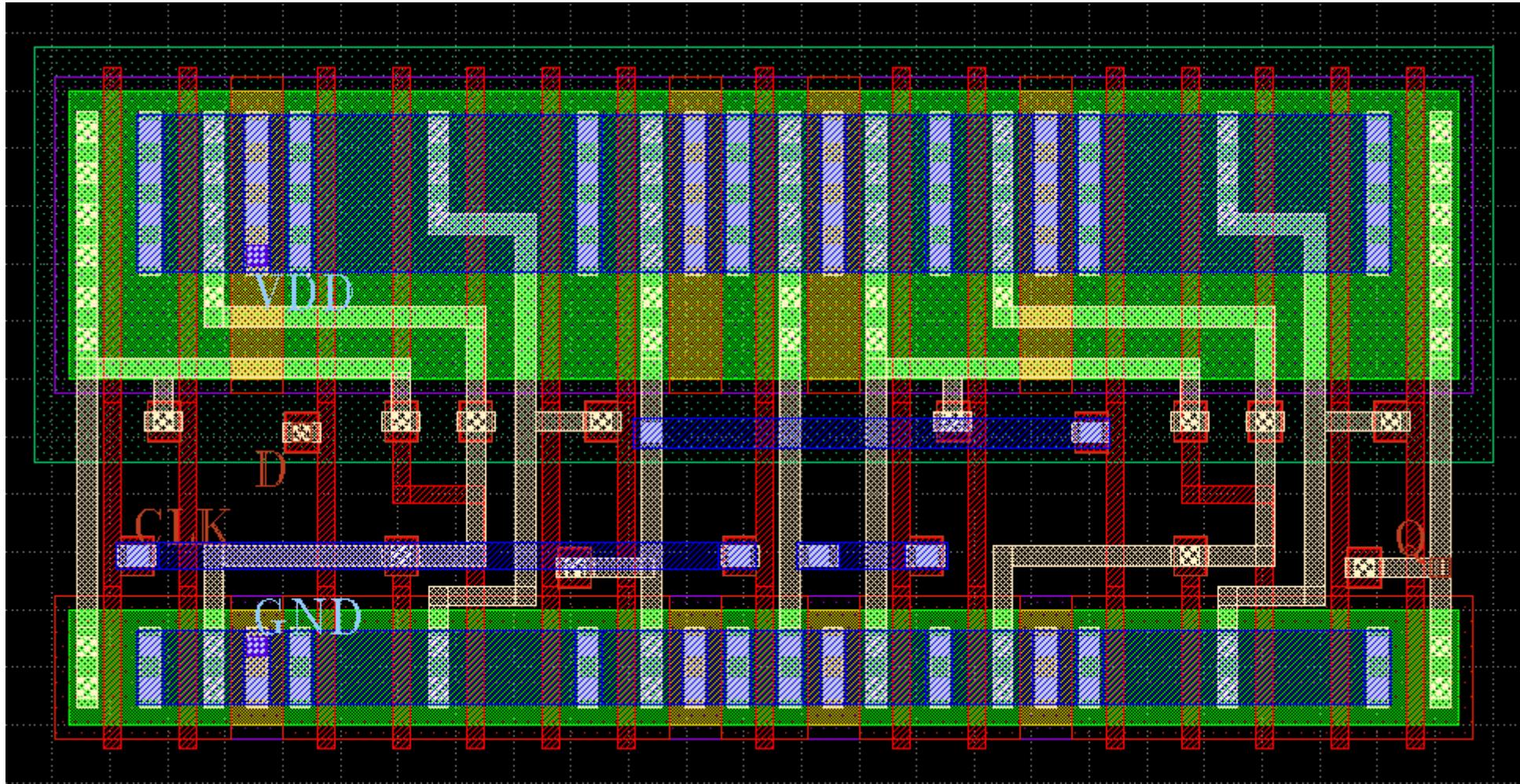
Right:TC74HC74



Example of a D flip-flop with two D latches

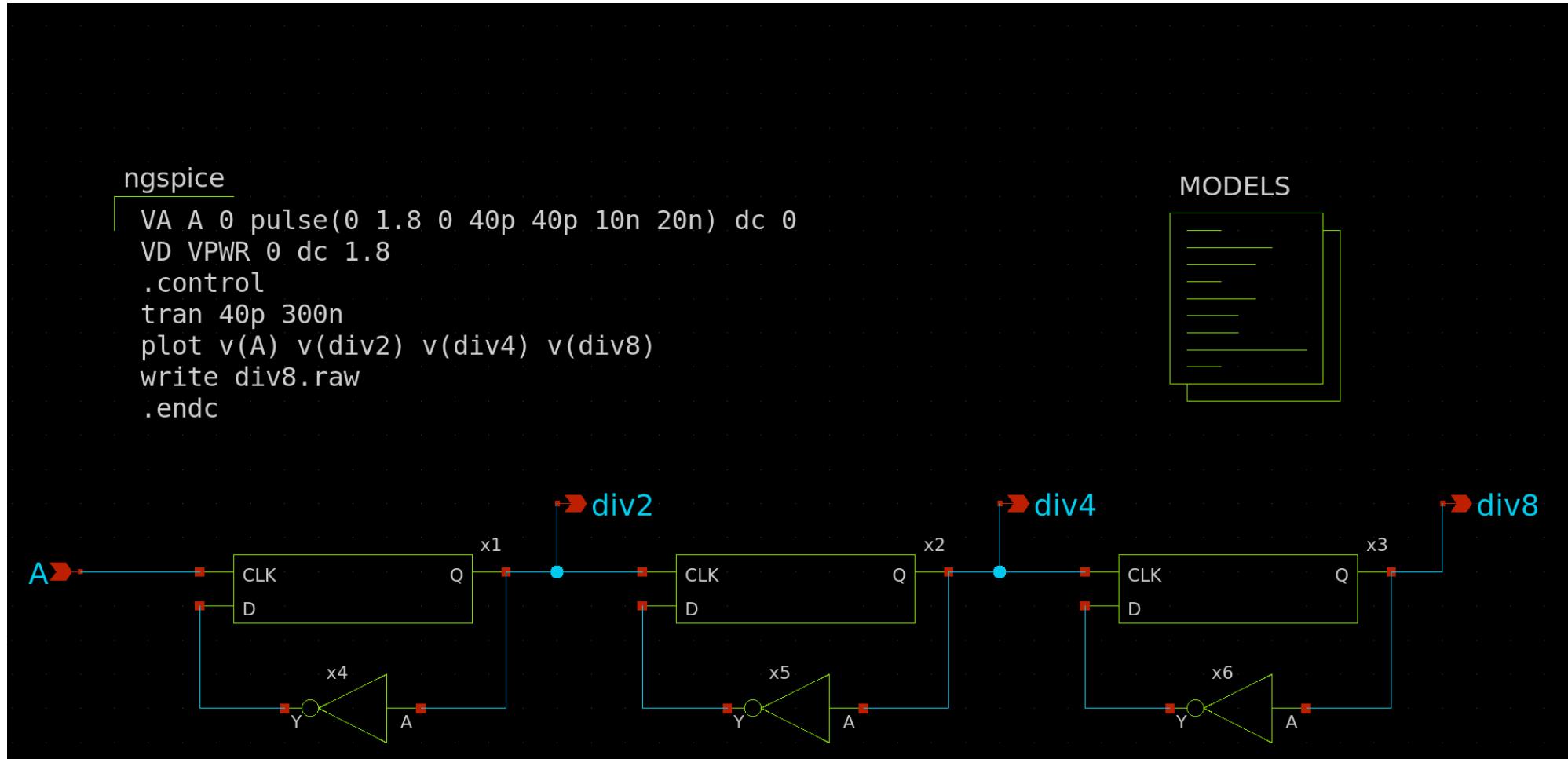


Example of a D flip-flop with two D latches

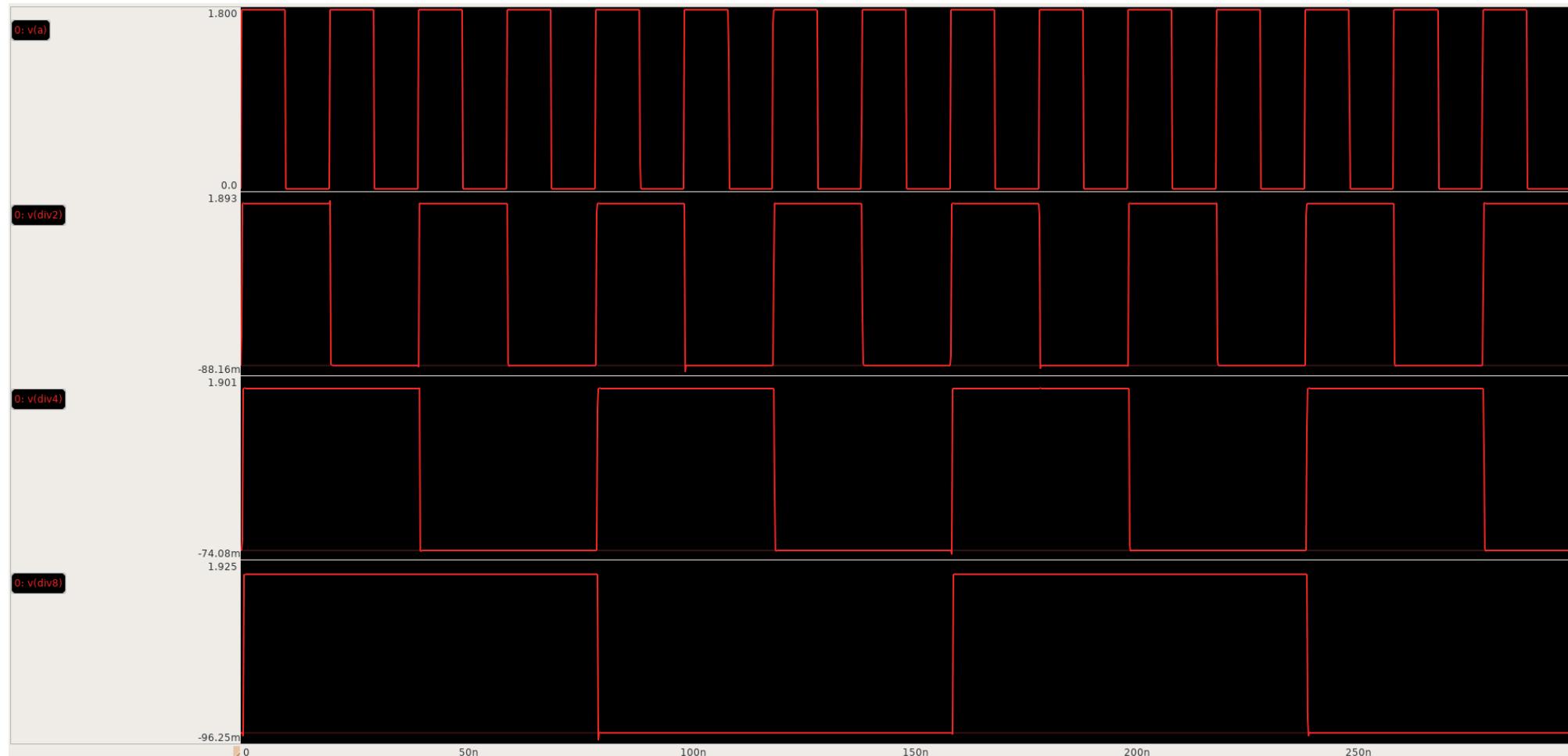


f/2, f/4, f/8 Clock divider

fdiv.sch

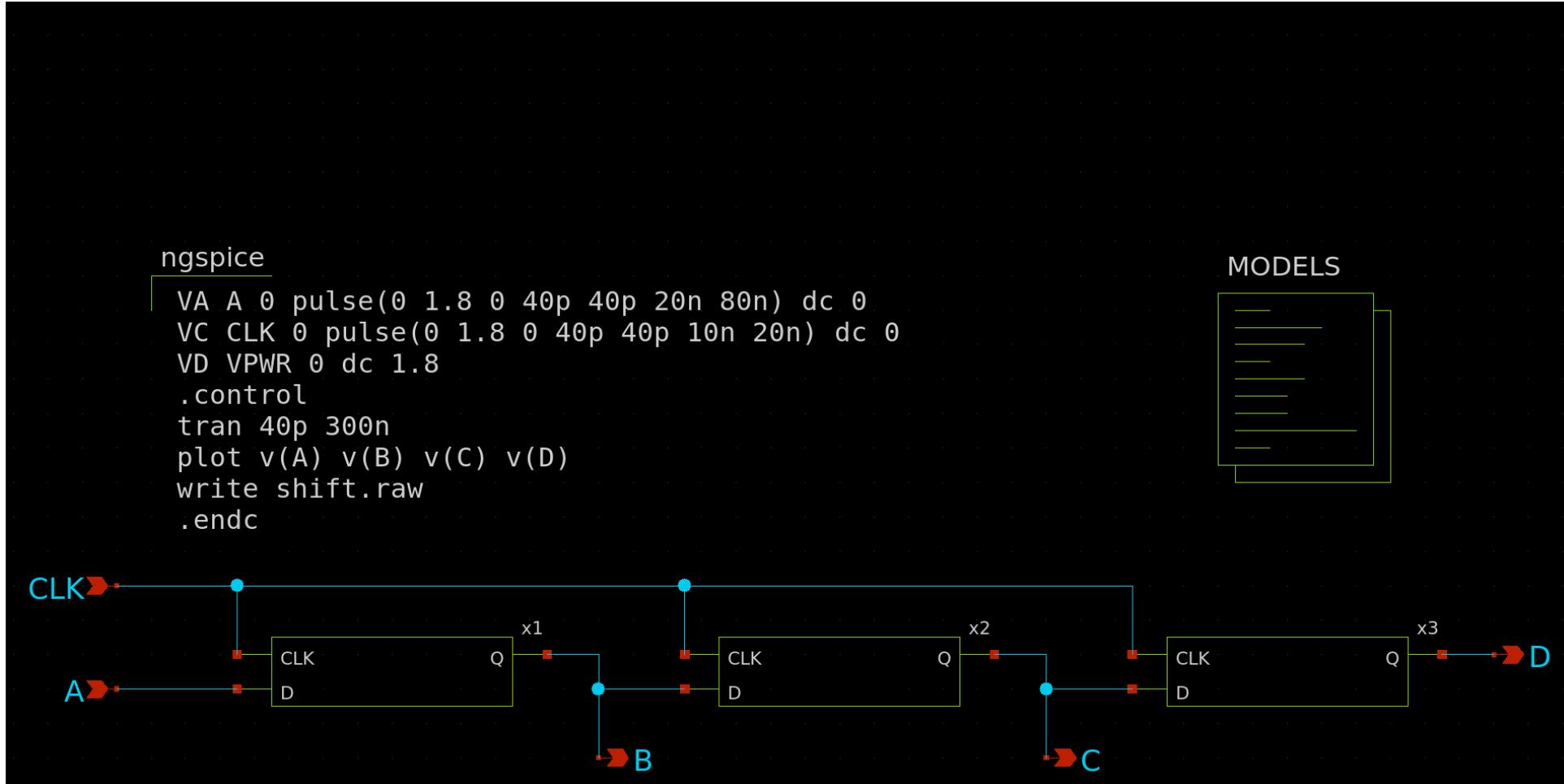


f/2, f/4, f/8 Clock divider

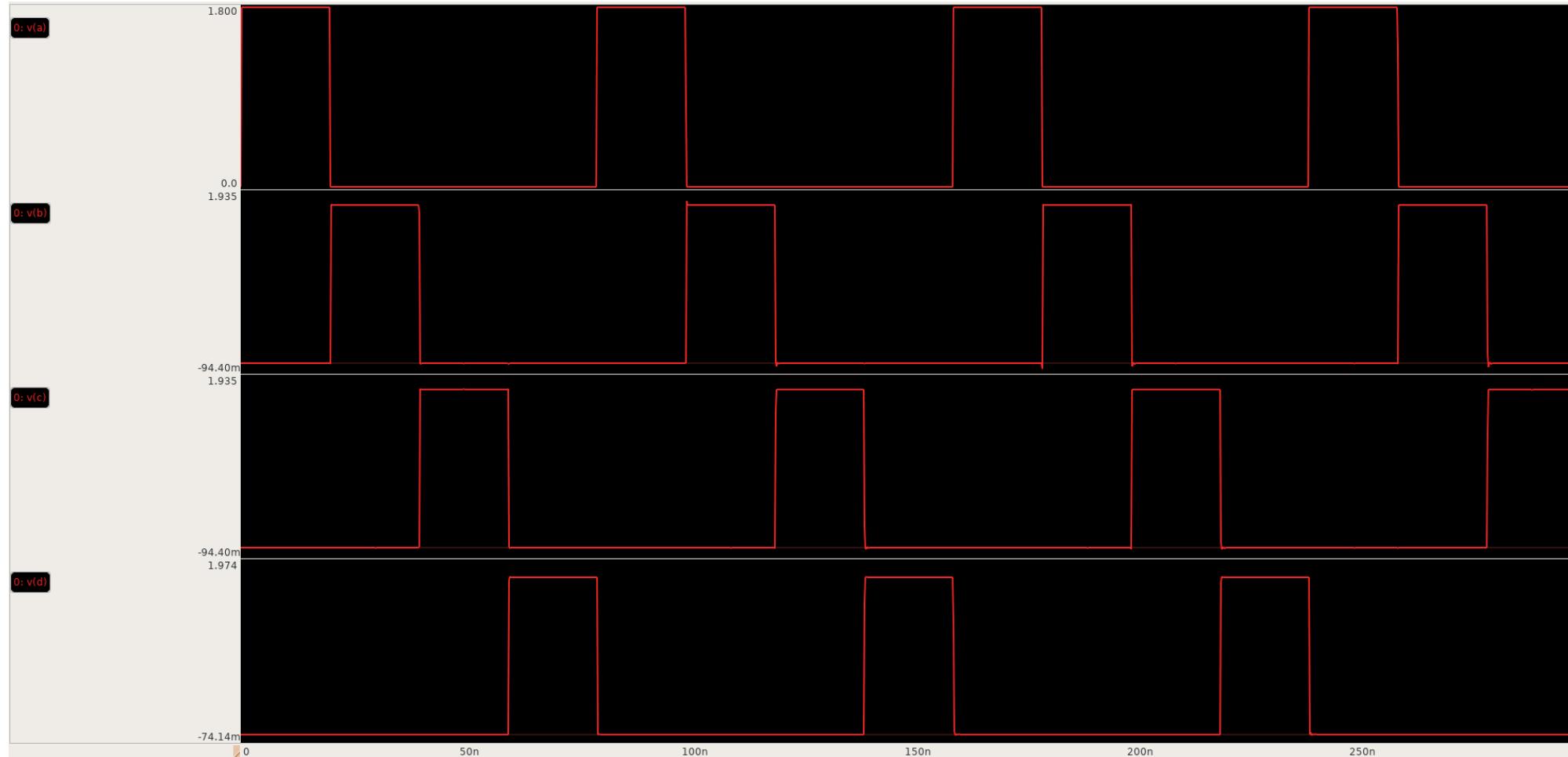


3-bit Shift register

shift.sch

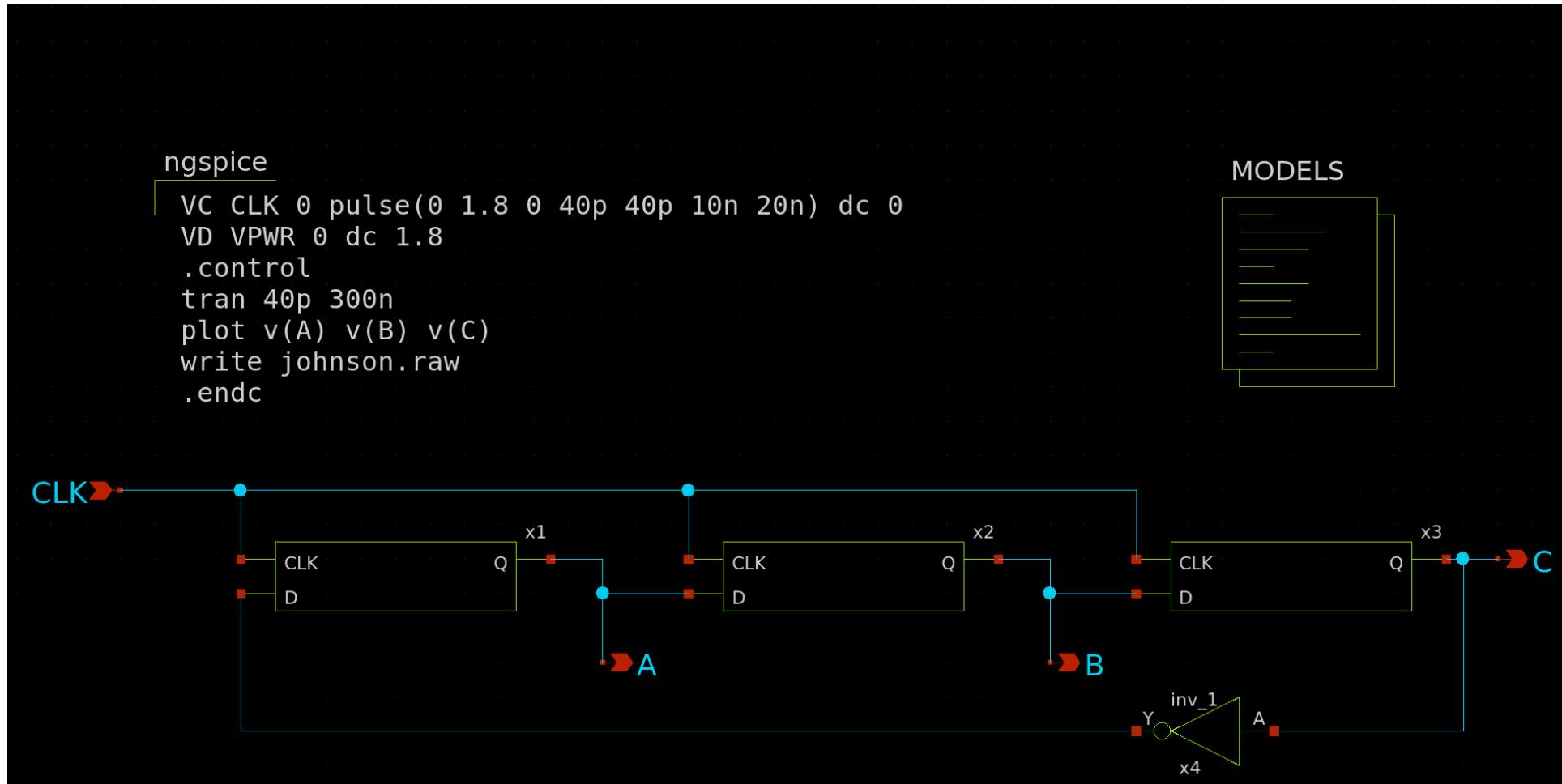


3-bit Shift register



3-bit Johnson ring counter

johson.sch



3-bit Johnson ring counter

