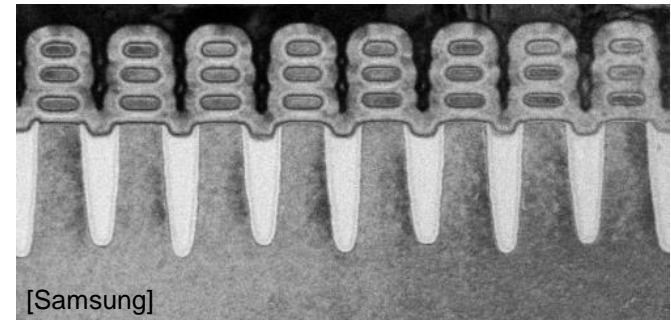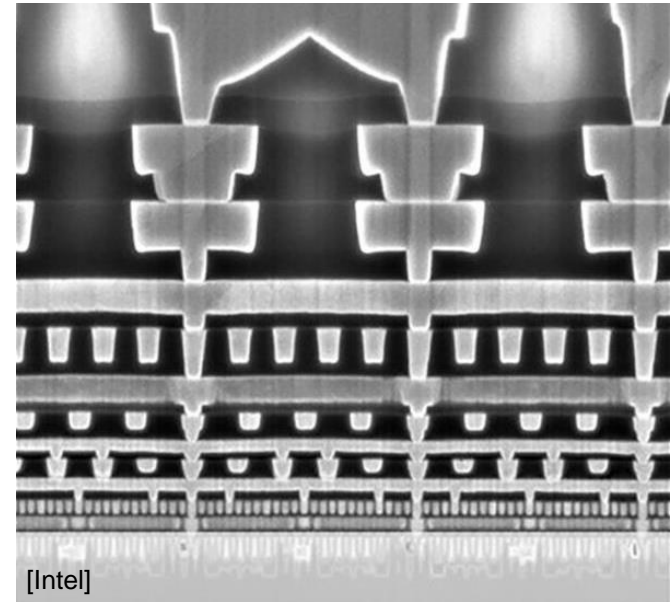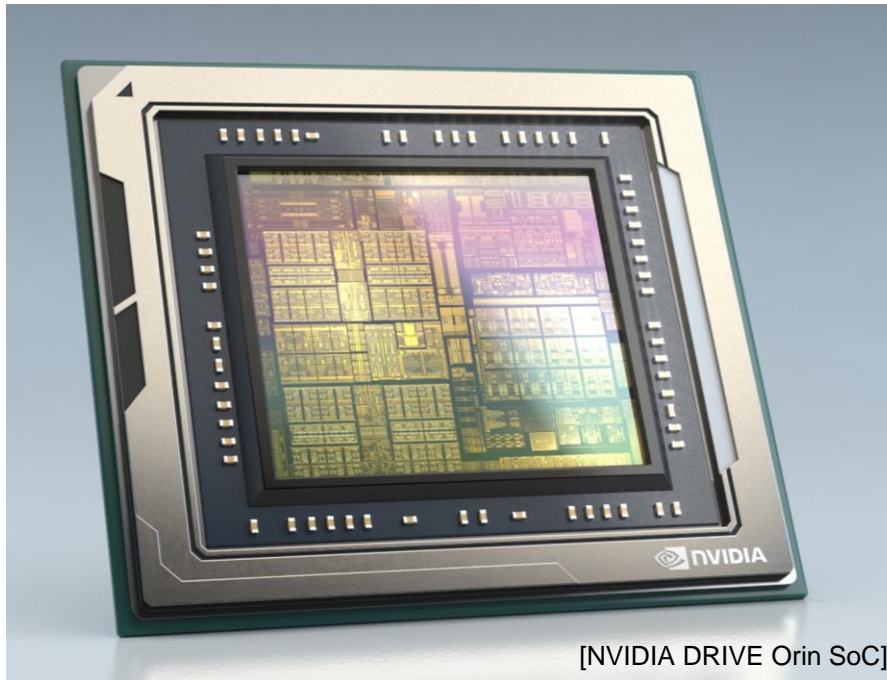# Teaching Mixed-Signal Design Using Open-Source Tools

Boris Murmann

bmurmann@hawaii.edu

Today's Chip Technology: Truly Amazing, But not "Cool"?

[NVIDIA DRIVE Orin SoC]

[Intel]

[Samsung]

Competition for Tech Talent

- Today's tech talents are drawn to higher levels of abstraction
  - Near instant gratification, e.g., in machine learning applications

- Today's tech stack is fueled by chips
  - But few are interested in designing the chips

Abstraction Layer Sandwich

**Software Systems
Algorithms**

**Hardware Systems
Circuits**

**Devices
Materials**

Traditional learning
trajectory for chip
designers

- The current education system requires far too may prerequisite courses before exposing students to chip design (especially mixed-signal)

- The field was created bottom-up, but innovation is progressively shifting to higher levels of abstraction

- We must adjust to this trend to re-energize chip design education

Starting From the Top

Software Systems
Algorithms

Hardware Systems
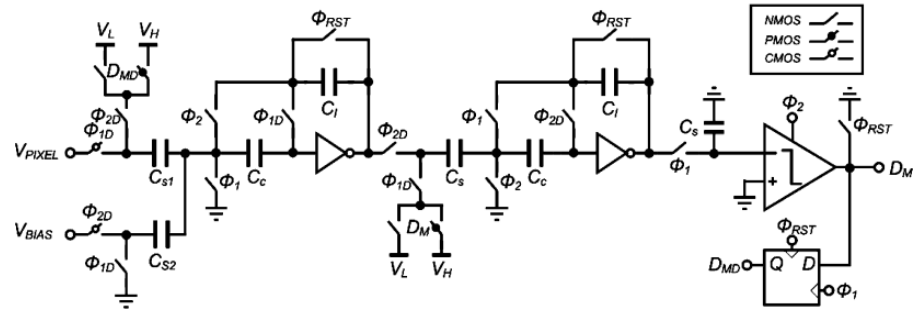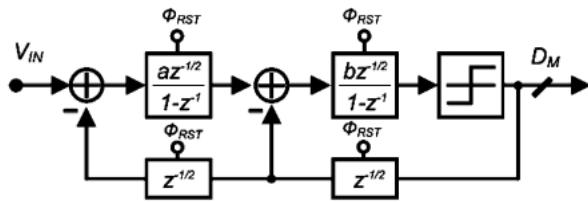**Circuits**

Devices
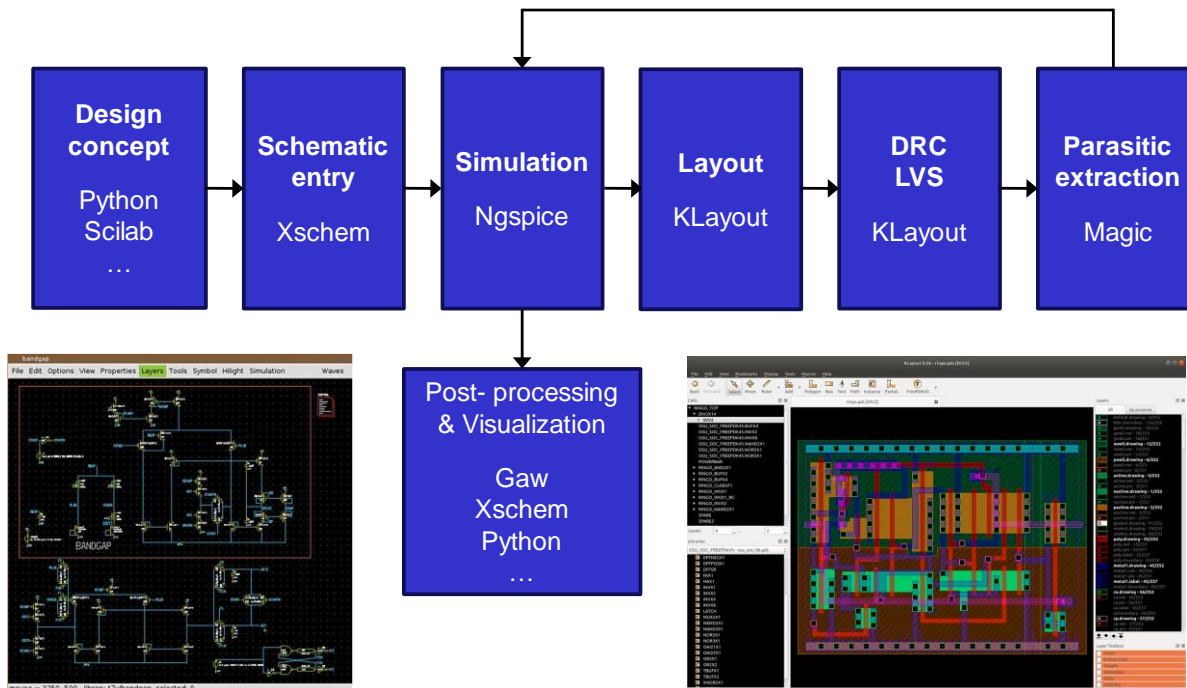**Materials**

"Just-in time"
exploration
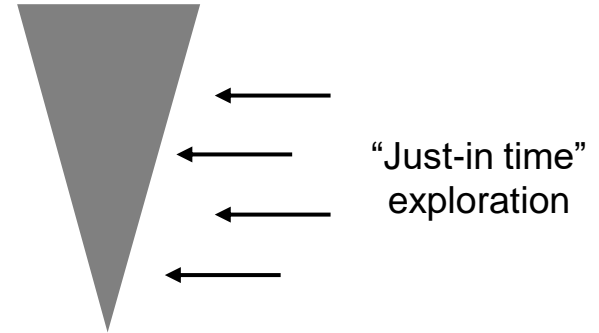
- It is not necessary to understand the entire sandwich to learn the basics of chip design (including mixed-signal ICs)

- Possible approaches for university teaching
  - Follow along as the instructor creates a "template" design
  - Form teams of students with complementary skill sets
    - Some may understand transistors, some excel at software, etc.

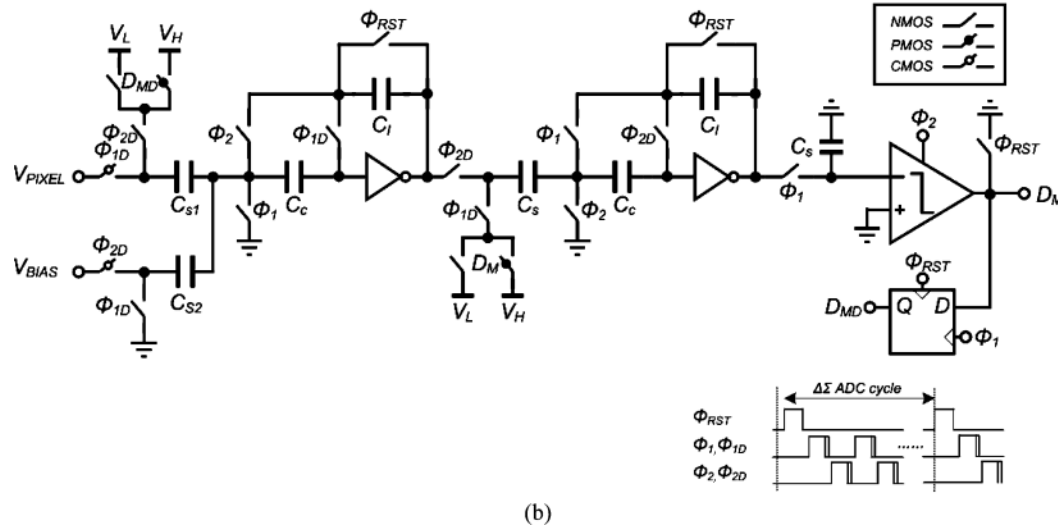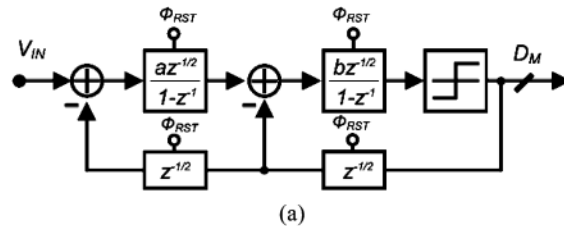# Overview of EE 628 "Tape-out Course" (University of Hawaii)

[Chae, JSSC 1/2011]

High-level model → Design and layout of complete transistor-level circuit

Design concept	Schematic entry	Simulation	Layout	DRC LVS	Parasitic extraction
Python Scilab …	Xschem	Ngspice	KLayout	KLayout	Magic

Post- processing & Visualization

Gaw
Xschem
Python
…

https://github.com/bmurmann/EE628

Template Project: Incremental Delta-Sigma A/D Converter

(a)

(b)

- Attractive features
  - Can study operation & nonidealities in software
  - Can do a gradual transition to real circuits & transistors
  - Given implementation has low overhead (no opamps)
  - Circuit has low pin count, easy to fit many copies in one package

Y. Chae et al., "A 2.1 M Pixels, 120 Frame/s CMOS Image Sensor With Column-Parallel ADC Architecture," in IEEE Journal of Solid-State Circuits, Jan. 2011. https://ieeexplore.ieee.org/document/5641589
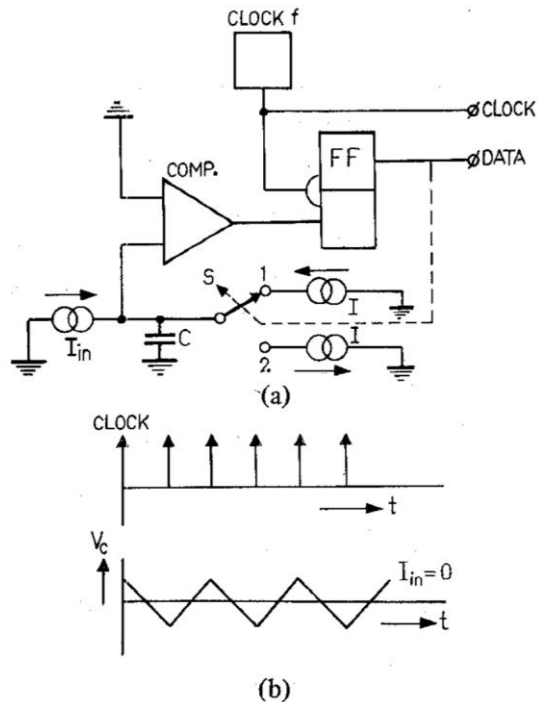
Application Context – Digital Voltmeters

Fig. 1. (a) Basic sigma-delta modulator. (b) Pulse patterns as a function of time.

$$N = n_{up} + n_{down}$$
$$n = n_{up} - n_{down}$$

$$\frac{I_{in}}{I_{ref.}} = \frac{n}{N}$$

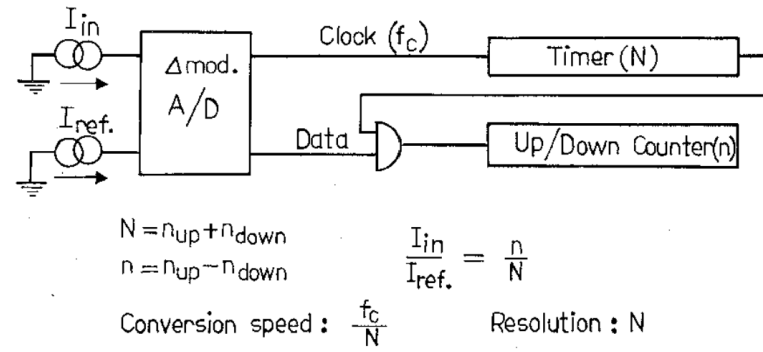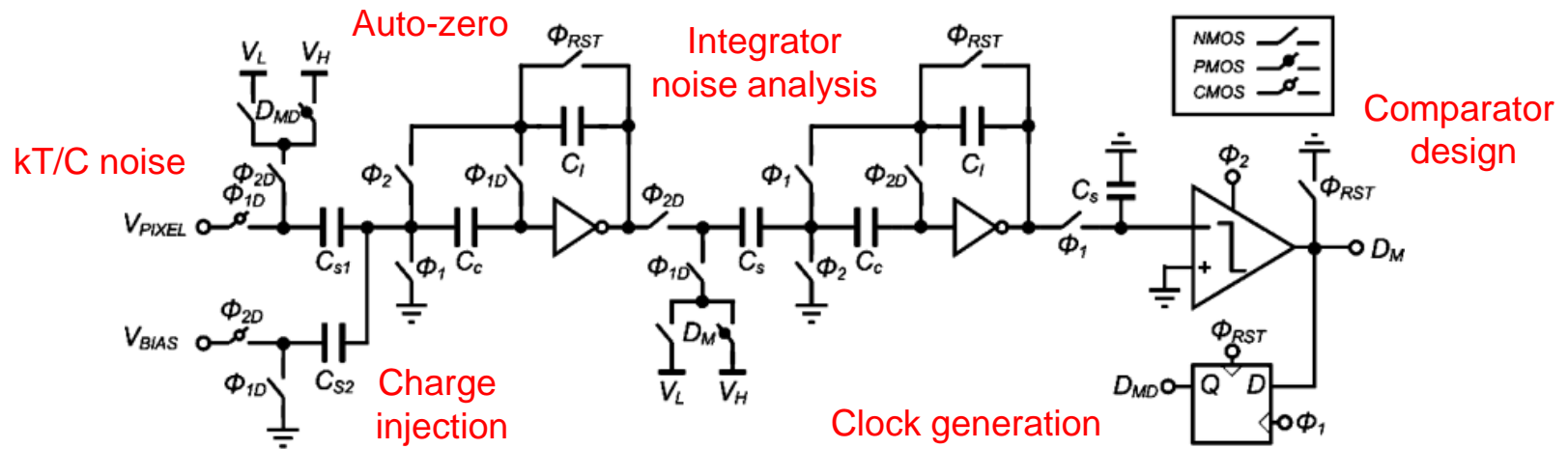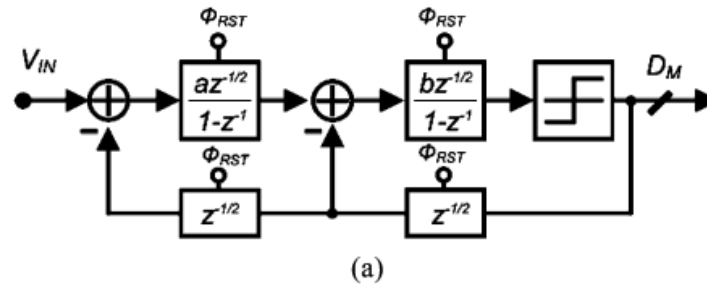Conversion speed: $\frac{f_C}{N}$ Resolution: N

Fig. 3. Digital controller circuit.


R. van de Plassche and R. E. J. van Der Grift, "A five-digit analog-digital converter," in IEEE Journal of Solid-State Circuits, Dec. 1977. https://ieeexplore.ieee.org/document/1050975

Lots of Interesting Things to Learn

How does the ideal model work?

(a)

kT/C noise

Auto-zero

Integrator noise analysis

Comparator design

Charge injection

Clock generation

Possible to complete project without understanding all details…
But students may want to take the next course to learn more…
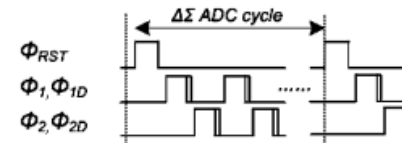
EE 628 Course Outline (15 Weeks)

- **High-level analysis and simulation of the template project**
  - Using Scilab, Simulink, etc.

- **Build and simulate the idealized spice-level circuit**
  - Using ideal switches and controlled sources (no transistors)

- **Build, analyze and simulate the transistorized circuits**
  - Switches, integrator, comparator, clock generator

- **Mid-semester review & team presentations**

- **Assemble the complete circuit**
  - Insert components one by one and verify operation

- **Layout, DRC, LVS**
  - First using a trivial example, then for the designed blocks & chip level

- **Final review & presentations**

- **Tapeout!**

Lecture Structure
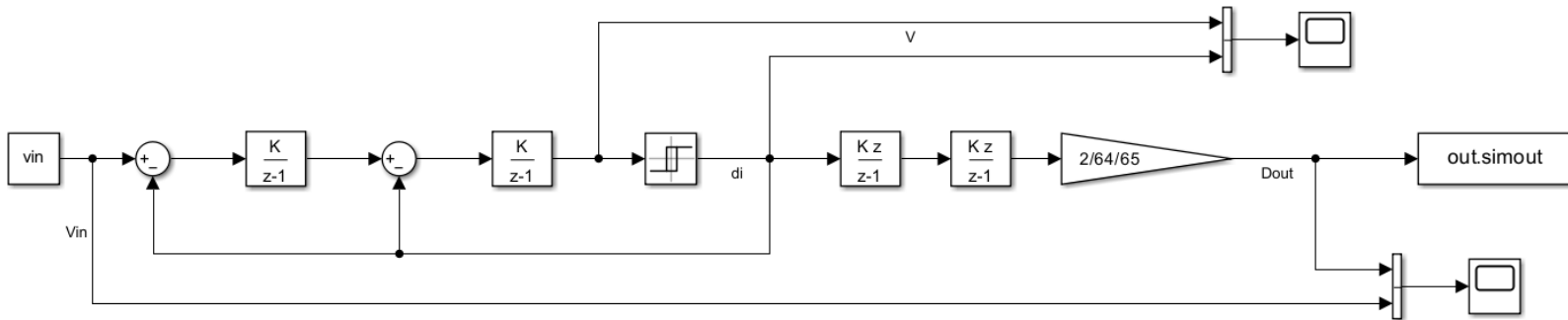
Classical lecture material

Circuit design
Circuit simulation
Analysis of nonidealities
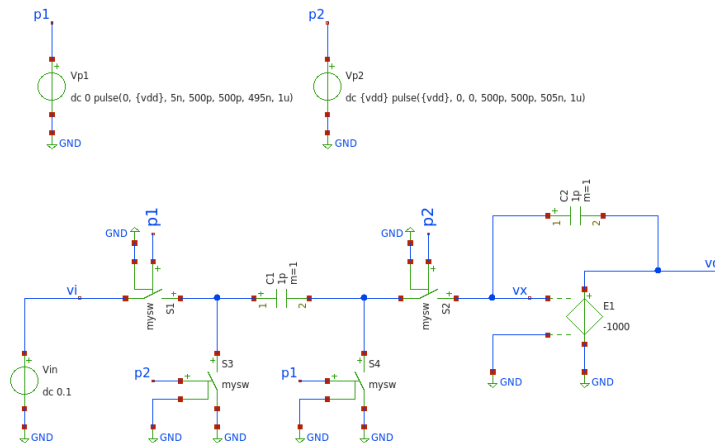Technology aspects
Layout basics

…

Demo & Discussion Time

Logistics
Tool demos
Troubleshooting
Student presentations

…

Progression

Conceptual integrator circuit

Complete "Transistorized" Stage

U.S.-Japan Collaborative Workshop on Advanced
Integrated Circuit Design (Phase 2)
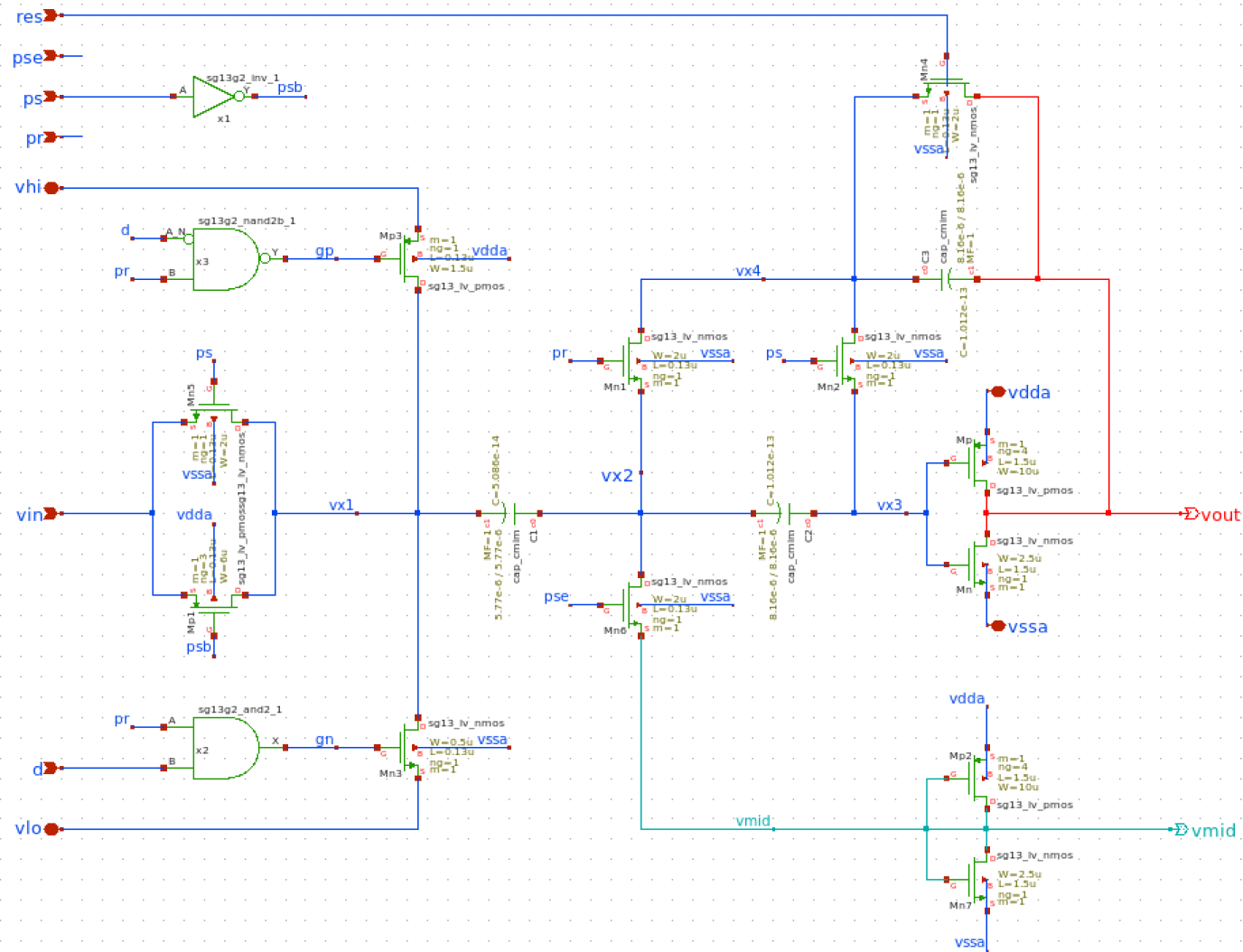
Complete Delta-Sigma Modulator
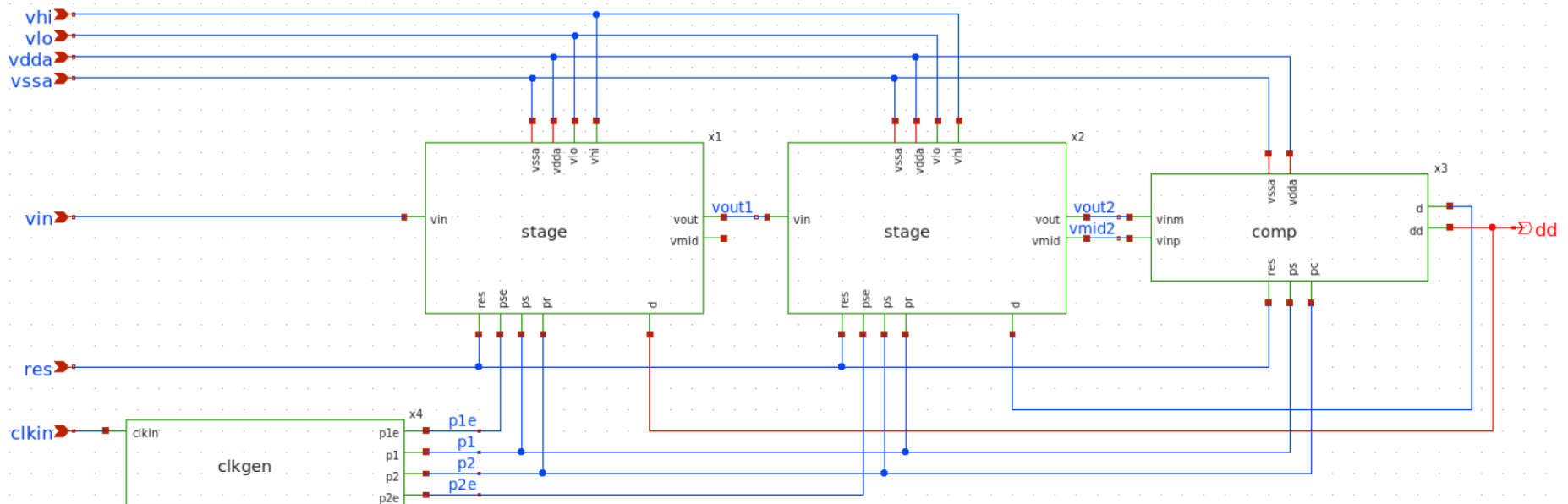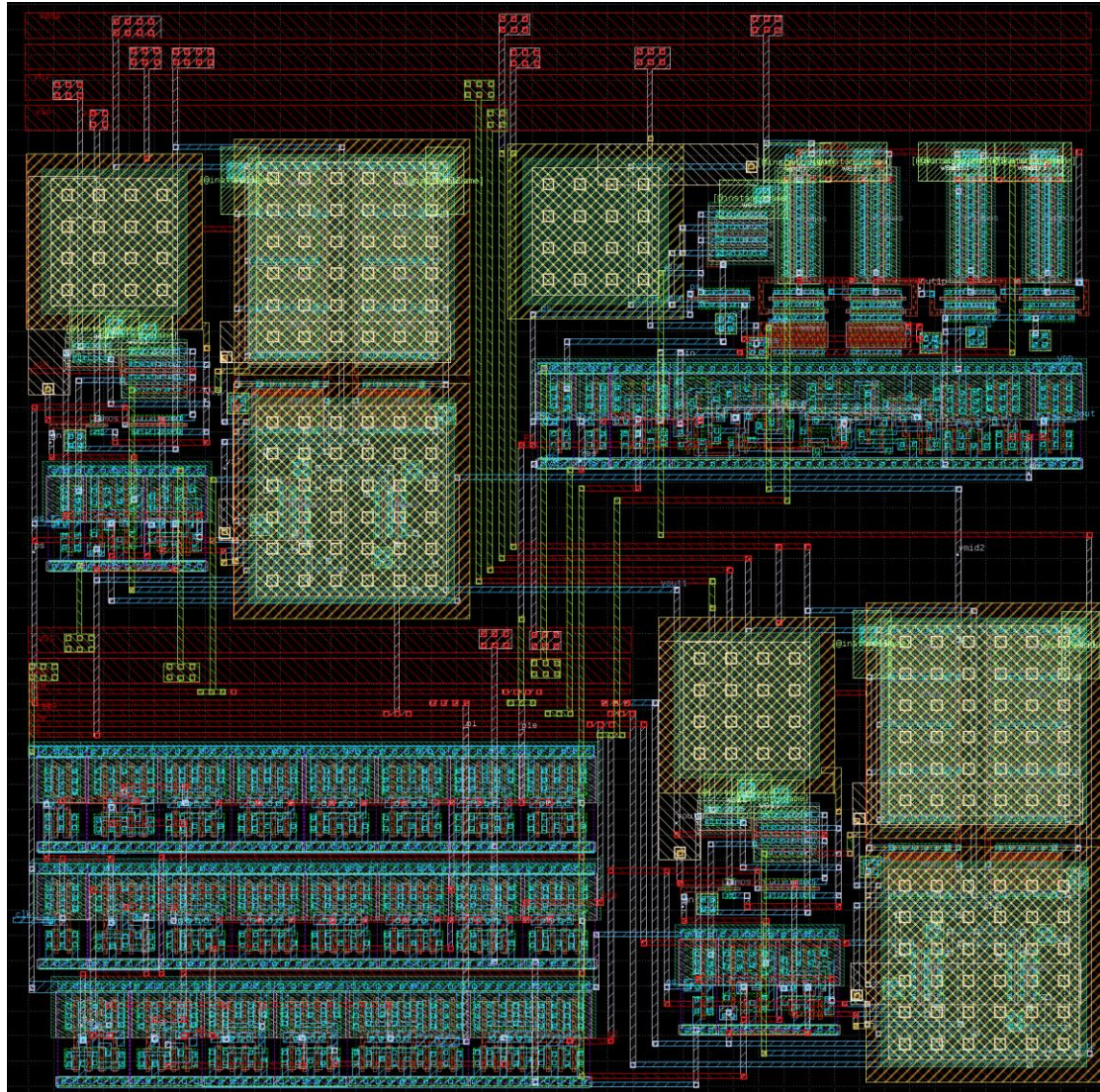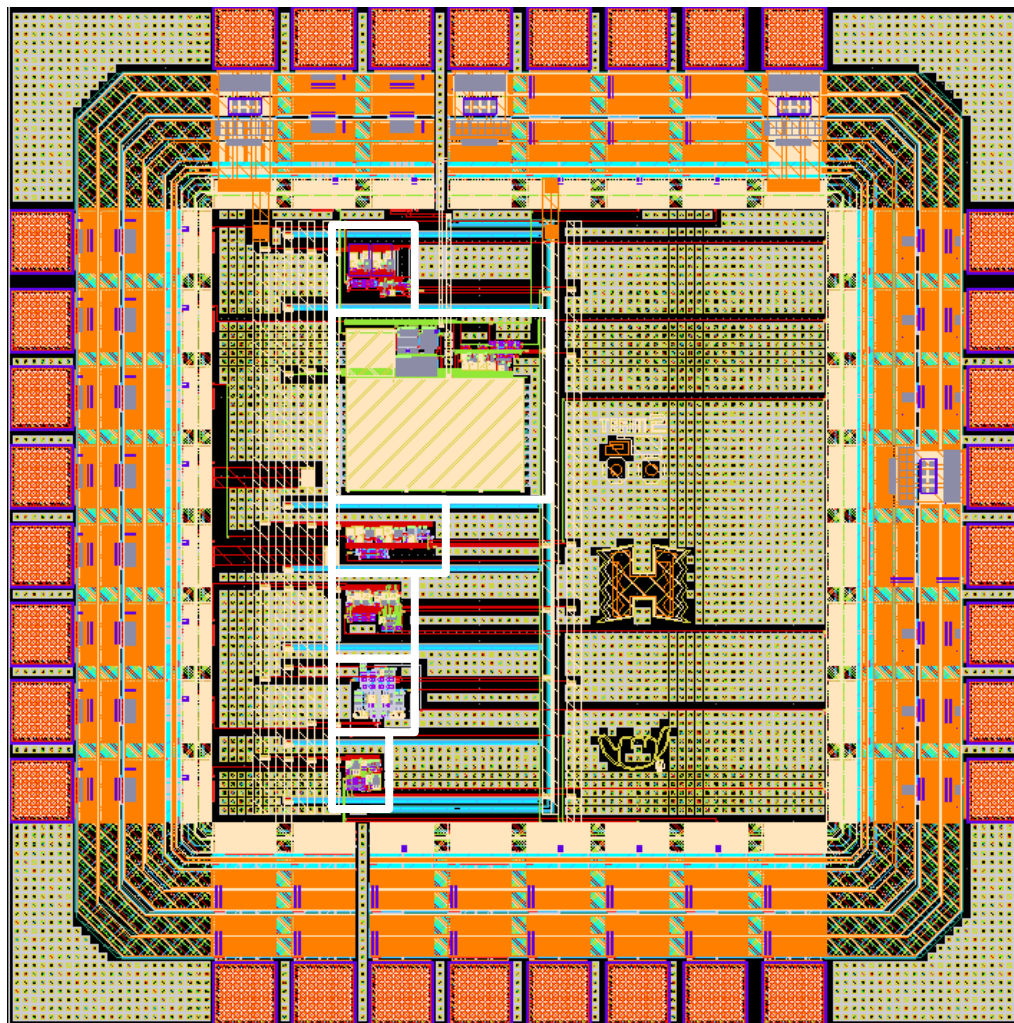
Technology: IHP SG13G2 (Open-Source PDK)
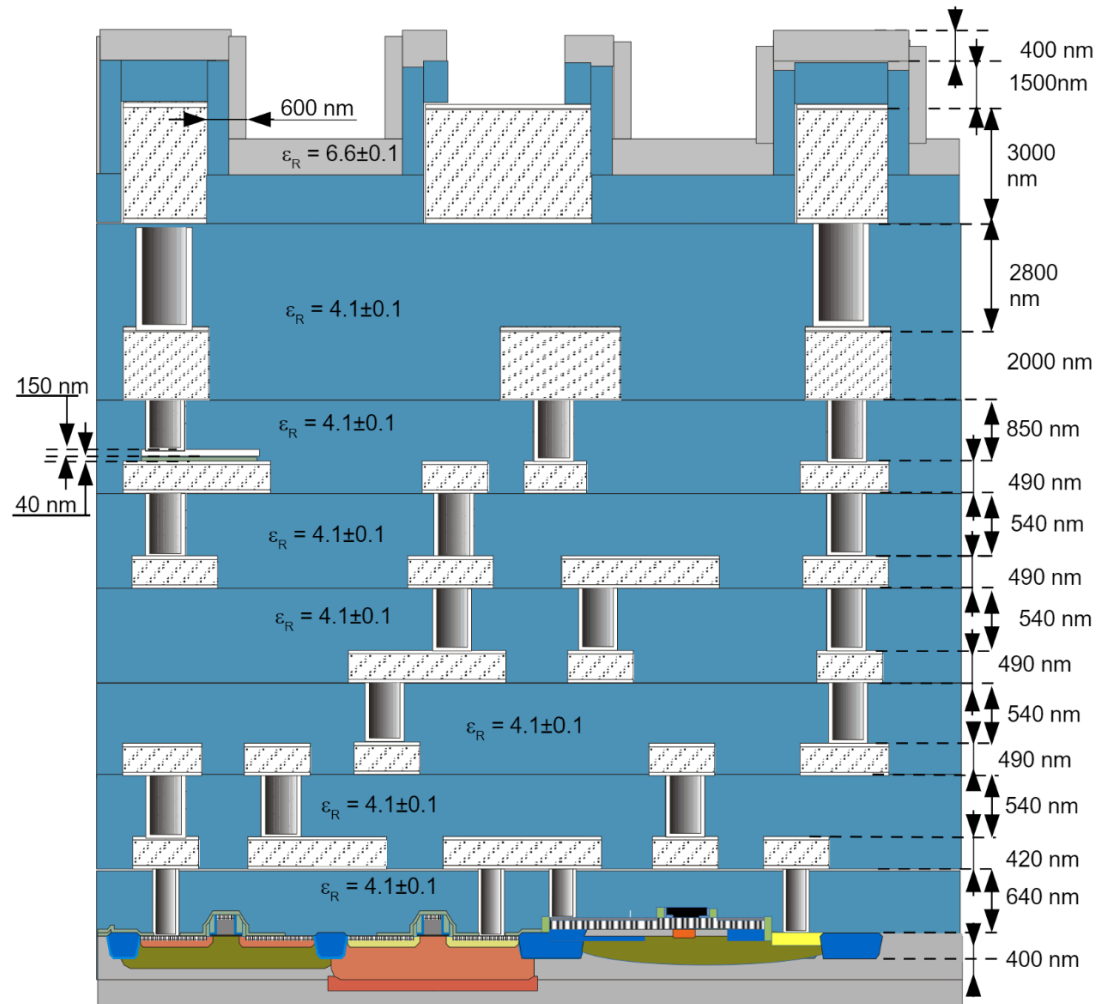
https://github.com/IHP-GmbH/IHP-Open-PDK/blob/main/ihp-sg13g2/libs.doc/doc/SG13G2_os_process_spec.pdf
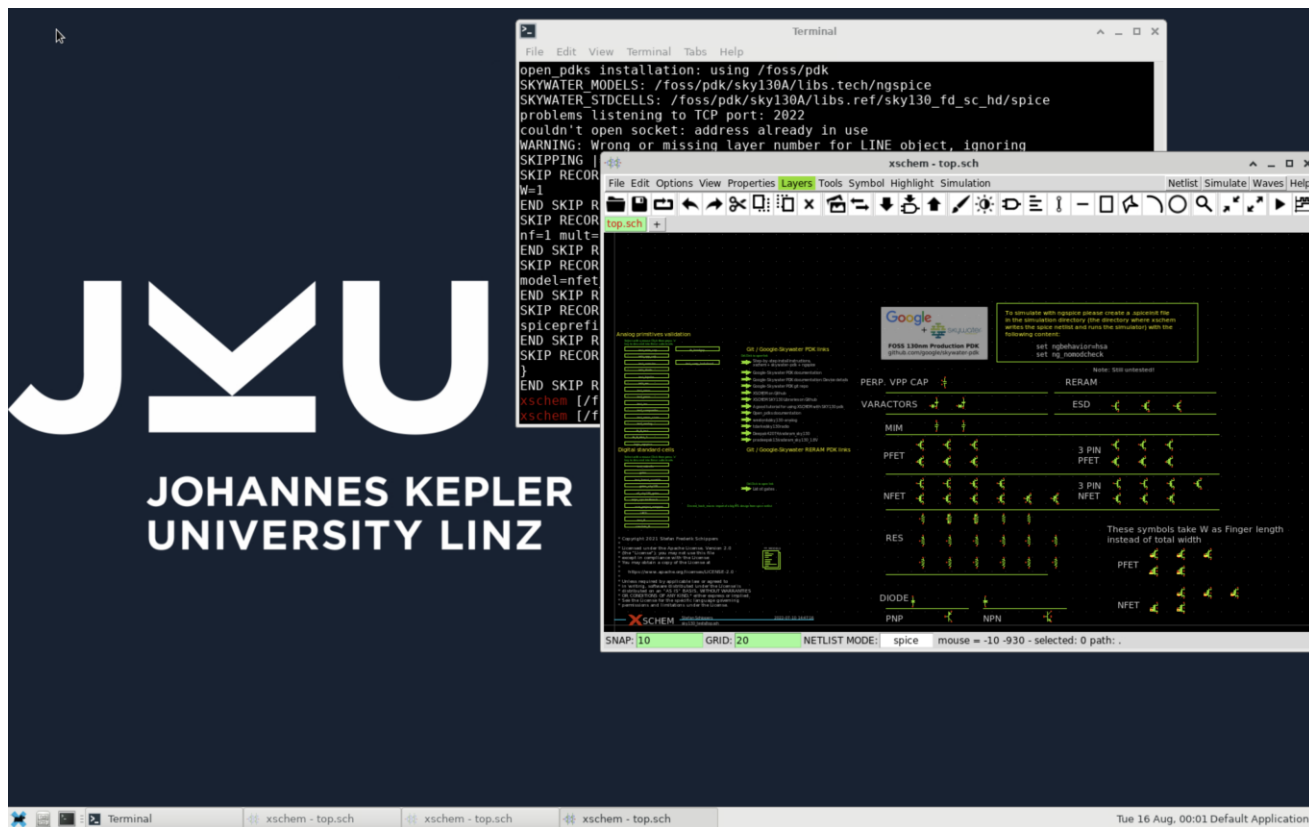
Sample Layout

U.S.-Japan Collaborative Workshop on Advanced
Integrated Circuit Design (Phase 2)

Final Chip (Shared by 6 Teams)

6 Incremental $\Delta\Sigma$
1 LDO

Open-Source Tools via Docker Container

- https://github.com/hpretl/iic-osic-tools

- Not the highest performance option, but easy entry point for students
  - Can transition to self-installation of all tools as a second step

The Power of Open-Source PDKs and Tools

- No license limitations

- Convenient sharing of all course materials through GitHub

- Easy to interface with Python (students typically have prior exposure)

- Students learn basic data ops and GitHub collaboration
  - Important skillset, regardless of specialization

Demo 1: "How To" Examples Using Notebooks

- How to evaluate distortion in a basic track-and hold circuit?

SKY130 Track and Hold Circuit

Tool setup

```
In [37]:   import pandas as pd
           import matplotlib.pyplot as plt
           import numpy as np
           import os

           CONDA_PREFIX = os.environ.get('CONDA_PREFIX', None)
           if not CONDA_PREFIX:
             import condacolab
             CONDA_PREFIX = condacolab.PREFIX
```

```
In [38]:   %%writefile environment.yml
           channels:
             - litex-hub
           dependencies:
             - open_pdks.sky130a
             - ngspice
```

Overwriting environment.yml

```
In [39]:   !mamba env update -n base -f environment.yml
```
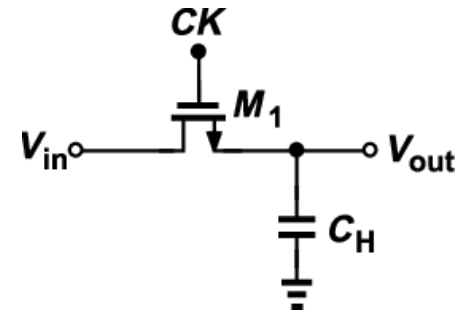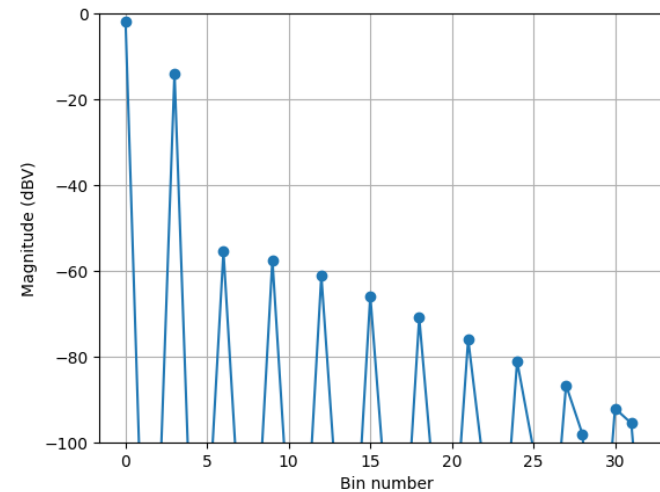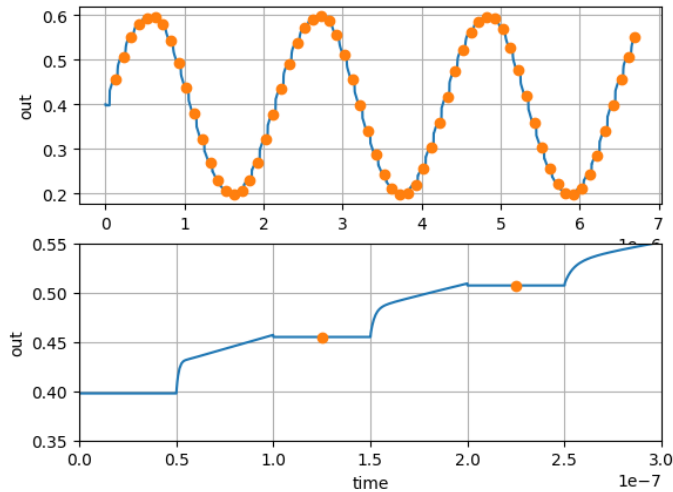
GitHub link

Circuit

```
%%writefile netlist.spice
* Track-and-hold circuit using single NMOS
.lib "/usr/local/share/pdk/sky130A/libs.tech/ngspice/sky130.lib.spice" tt
x1 in clk out 0 sky130_fd_pr__nfet_01v8_lvt w=5 l=0.15
cl out 0 100f
vin  in 0 sin (0.4 0.2 {fin})
vclk clk 0 pulse (1.2 0 0 100p 100p {per/2} {per})
.param nfft=64 fclk=10Meg per=1/fclk cycles=3 fin=fclk*cycles/nfft
```

Raw simulation data

```
df1 = pd.read_csv("output1.txt", delim_whitespace=True)
df1
```

	time	out
0	0.000000e+00	0.400000
1	1.000000e-12	0.399961
2	2.000000e-12	0.399922
3	4.000000e-12	0.399845
4	8.000000e-12	0.399695
...	...	...

Postprocessing

U.S.-Japan Collaborative Workshop on Advanced Integrated Circuit Design (Phase 2)

Demo 2: EE 628 Simulation Flow (Xschem/Ngspice/Jupyter)

U.S.-Japan Collaborative Workshop on Advanced
Integrated Circuit Design (Phase 2)