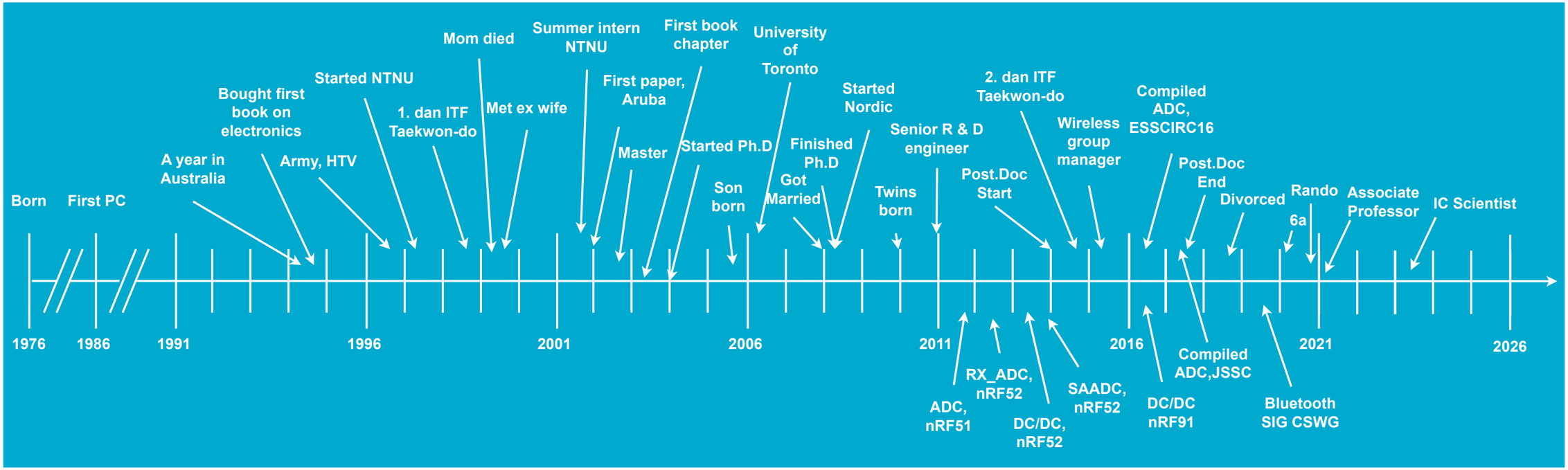
date: 2025-01-09

TFE4188 - Advanced Integrated Circuits

Lecture 1 - Introduction

Who

Carsten Wulff carstenw@ntnu.no



Teaching assistant

- Jonathan Sæthre Ege

How I see our roles

Professors: Guide students on what is impossible, possible, and hints on what might be possible

Ph.D students: Venture into the unknown and make something (more) possible

Master students: Learn all that is currently possible

Bachelor students: Learn how to make complicated into easy

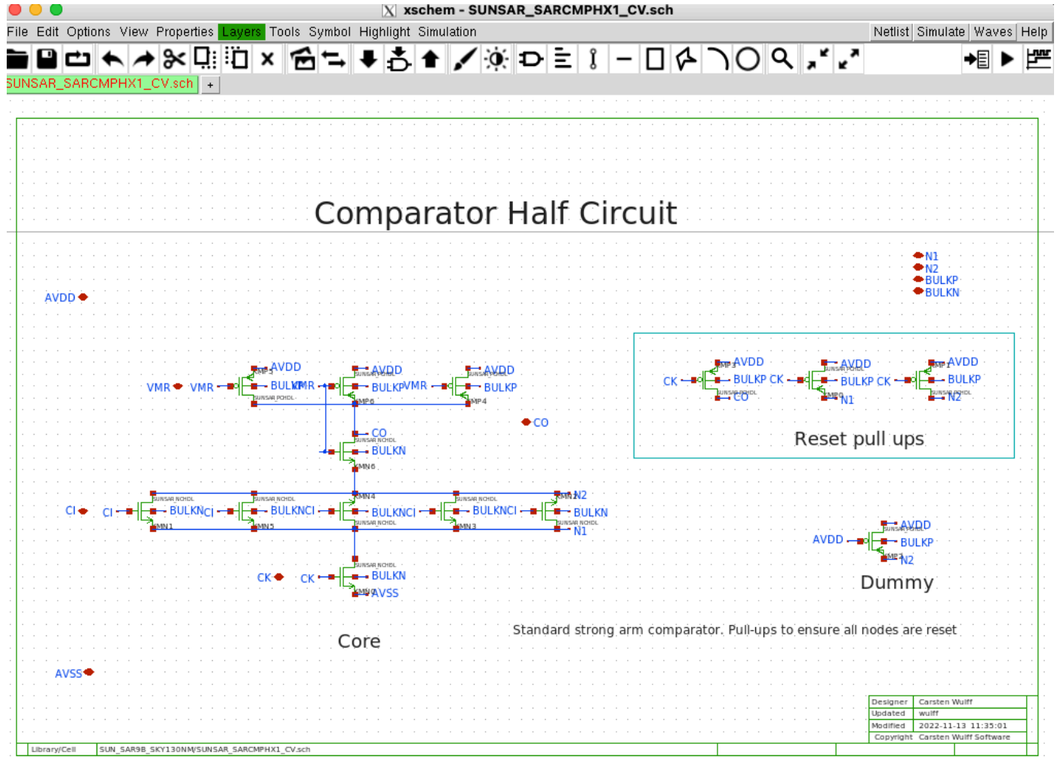
Industry: Take what is possible, and/or complicated, and make it easy

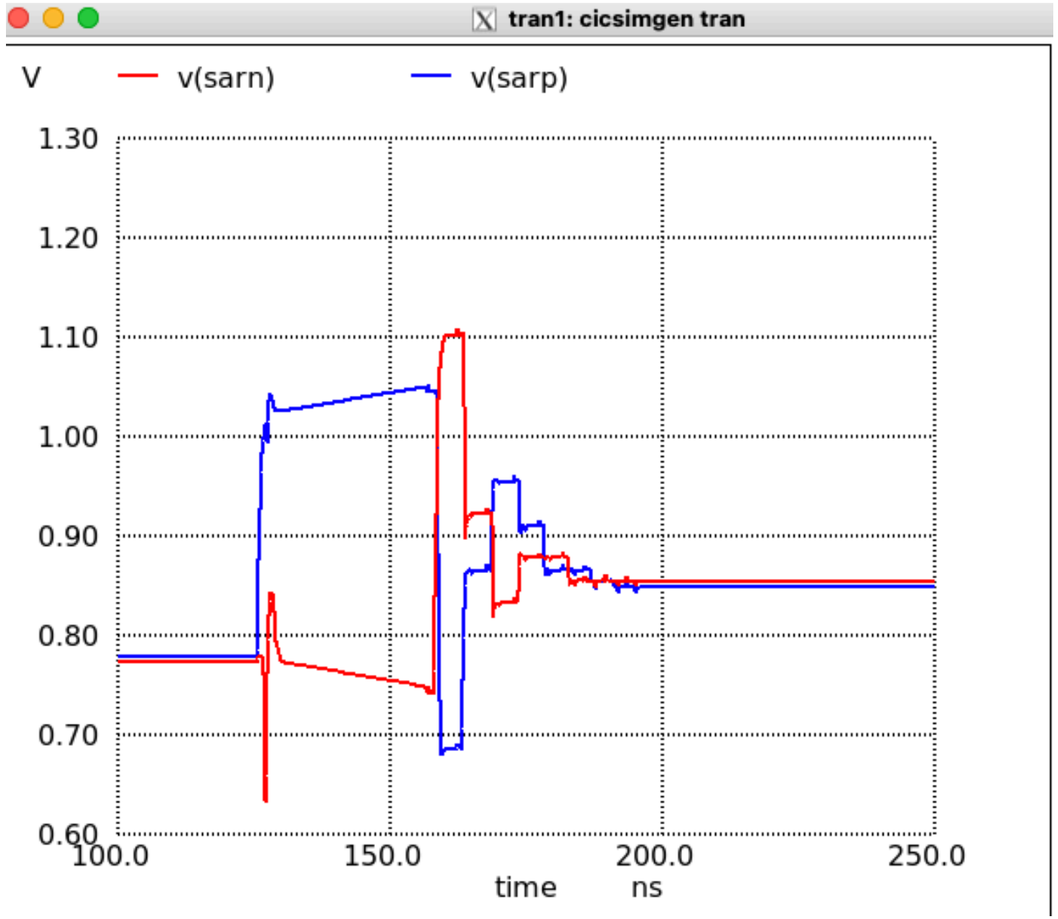
Why

I want you to learn the skills necessary to make your own ICs

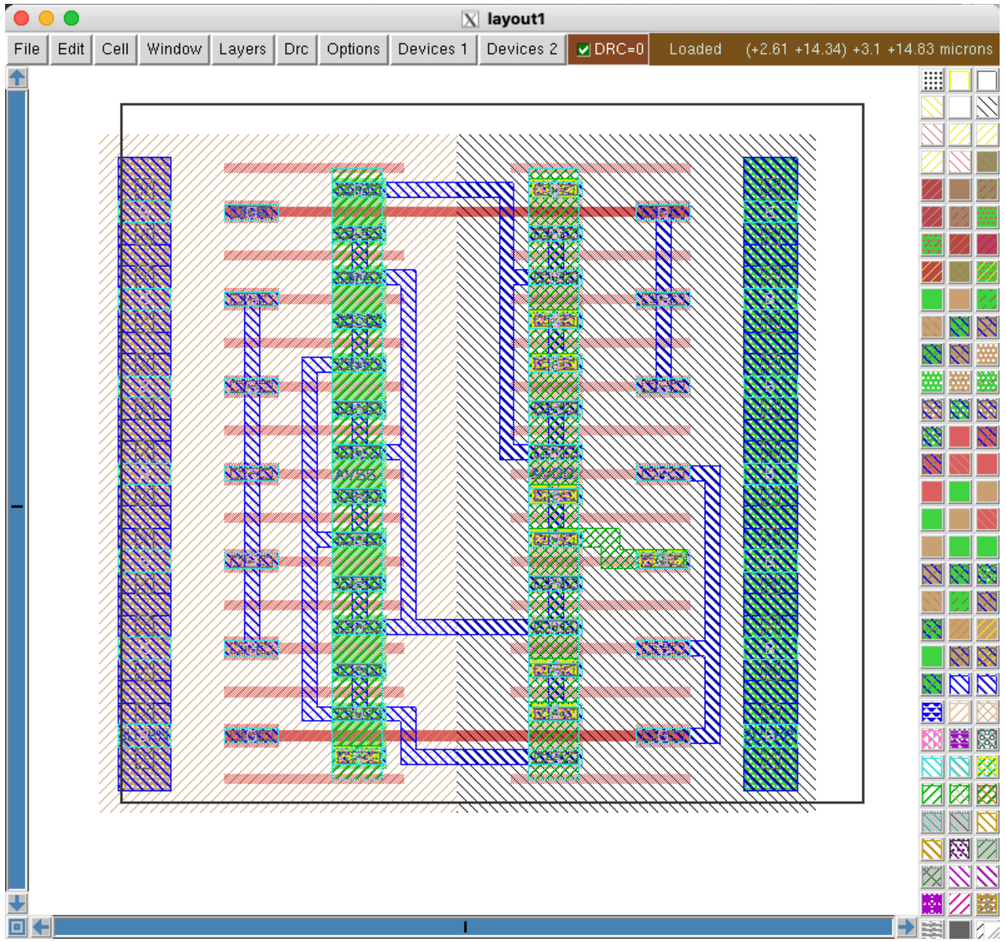
There will always be analog circuits, because the real world is analog

Life of an analog designer: Schematic Design





Life of an analog designer: Layout Design



Status	Abstraction	Design	Layout	Why
🚧	Chip	SystemVerilog	digital	Complex connections, few analog interfaces
🚧	Module	SystemVerilog	digital	Large amount of digital signals, few analog signals
⚠	Block	Schematic	programmatic	Large amount of critical analog interfaces, few digital
✅	Cell	Netlist/JSON	compiled	Few analog interfaces, few digital interfaces
✅	Device	JSON	compiled	Polygon pushing
✅	Technology	JSON/Rules	compiled	Custom for each technology

Insights · Tech The Future

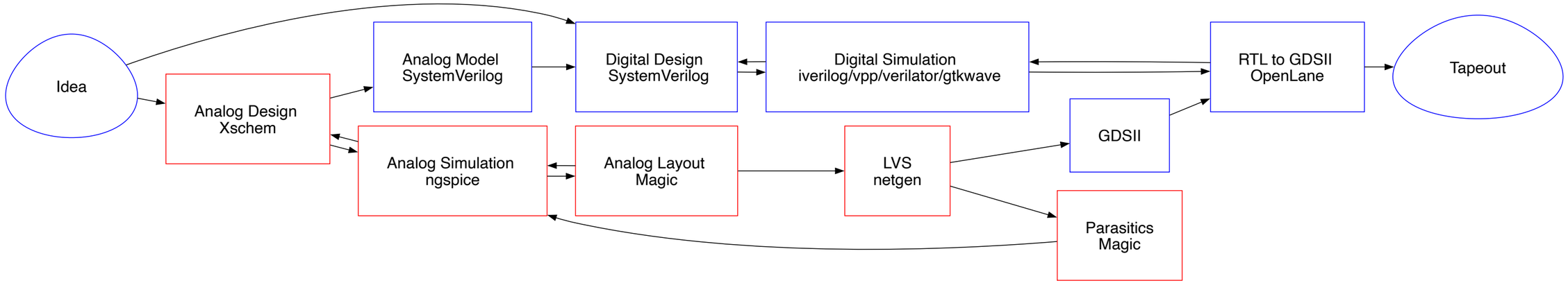
The World Is Analog

10/28/2014

Written by Peter Kinget

The world we live in is analog. We are analog. Any inputs we can perceive are analog. For example, sounds are analog signals; they are continuous time and continuous value. Our ears listen to analog signals and we speak with analog signals. Images, pictures, and video are all analog at the source and our eyes are analog sensors. Measuring our heartbeat, tracking our activity, all requires processing analog sensor information.

https://circuitcellar.com/insights/tech-the-future/kinget-the-world-is-analog/


Idea
Analog Design
Xschem
Analog Model
SystemVerilog
Digital Design
SystemVerilog
Digital Simulation
iverilog/vpp/verilator/gtkwave
RTL to GDSII
OpenLane
Tapeout
Analog Simulation
ngspice
Analog Layout
Magic
LVS
netgen
GDSII
Parasitics
Magic

Will you tape-out an IC?

https://tinytapeout.com/runs/

Current chips

Run	Launched	Closed	Shuttle	Designs	Chips expected	Estimated delivery date
TT01	2022-08-17	2022-09-01	MPW7	152	2024-01-30	None - test shuttle
TT02	2022-11-09	2022-12-02	2211Q	165	2023-10-17	Shipped 2024-01-25
TT03	2023-03-01	2023-04-23	2304C	100 *	2024-01-15	Shipped 2024-03-28
TT04	2023-07-01	2023-09-08	2309	143	2024-03-26	Shipped 2024-05-24
TT05	2023-09-11	2023-11-04	2311	174	2024-06-10	Shipped 2024-07-05
TT06	2024-01-30	2024-04-19	2404	238	2024-11-04	2024-12-15
TT07	2024-04-22	2024-06-01	2406	120	2024-12-15	2025-01-30
TT08	2024-06-10	2024-09-06	2409	135	2025-04-01	2025-05-12
TT IHP 0.2	2024-10-22	2024-11-04	IHP	95	2025-03-30	None - test shuttle
TT09	2024-09-07	2024-11-10	2411	TBD	2025-05-15	2025-06-30

* TT03 also includes 149 designs from TT02 for a total of 249 designs.

Future chips

The following chips are planned for the future. The dates are estimates and may change.

Run	Submission deadline	Estimated delivery date
TT10	2025-03-07	Nov 2025
TT11	2025-06-02	Jan 2026

https://analogicus.com/aic2025

- *Project flow support*: **Confluence**, JIRA, risk management (DFMEA), failure analysis (8D)
- *Language*: **English**, **Writing English (Latex, Word, Email)**
- *Psychology*: Personalities, convincing people, presentations (Powerpoint, Deckset), **stress management (what makes your brain turn off?)**
- *DevOps*: **Linux**, bulid systems (CMake, make, ninja), continuous integration (bamboo, jenkins), **version control (git)**, containers (docker), container orchestration (swarm, kubernetes)
- *Programming*: Python, C, C++, Matlab Since 1999 I've programmed in Python, Go, Visual BASIC, PHP, Ruby, Perl, C#, SKILL, Ocean, Verilog-A, C++, BASH, AWK, VHDL, SPICE, MATLAB, ASP, Java, C, SystemC, Verilog, Assembler, and probably a few I've forgotten.
- *Firmware*: signal processing, algorithms, software architecture, security
- *Infrastructure*: **Power management**, **reset**, **bias**, **clocks**
- *Domains*: CPUs, peripherals, memories, bus systems
- *Sub-systems*: **Radio's**, **analog-to-digital converters**, **comparators**
- *Blocks*: **Analog Radio**, Digital radio baseband
- *Modules*: Transmitter, **receiver**, de-modulator, timing recovery, state machines
- *Designs*: **Opamps**, **amplifiers**, **current-mirrors**, adders, random access memory blocks, standard cells
- *Tools*: **schematic**, **layout**, **parasitic extraction**, synthesis, place-and-route, **simulation**, (System)Verilog, **netlist**
- *Physics*: transistor, pn junctions, quantum mechanics

Zen of IC design (stolen from Zen of Python)

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Complex is better than complicated.
- Readability counts (especially schematics).
- Special cases aren't special enough to break the rules.
- Although practicality beats purity.
- In the face of ambiguity, refuse the temptation to guess.
- There should be one **and preferably only one** obvious way to do it.
- Now is better than never.
- Although never is often better than *right* now.
- If the implementation is hard to explain, it's a bad idea.
- If the implementation is easy to explain, it may be a good idea.

Find a problem that you really want to solve, and learn programming to solve it. There is no point in saying "I want to learn programming", then sit down with a book to read about programming, and expect that you will learn programming that way. It will not happen. The only way to learn programming is to do it, a lot.

— Carsten Wulff

```
s/programming/analog design/ig
```

Analog Design Process

- Define the problem, what are you trying to solve?
- Find a circuit that can solve the problem (papers, books)
- Find right transistor sizes. What transistors should be weak inversion, strong inversion, or don't care?
- Write a verification plan. Plan to simulate everything that could go wrong.
- Check operating region of transistors (.op)
- Check key parameters (.dc, .ac, .tran)
- Check function. Exercise all inputs. Check all control signals
- Check key parameters in all corners. Check mismatch (Monte-Carlo simulation)
- Do layout, and check it's error free. Run design rule checks (DRC). Check layout versus schematic (LVS)
- Extract parasitics from layout. Resistance, capacitance, and inductance if necessary.
- On extracted parasitic netlist, check key parameters in all corners and mismatch (if possible).
- If everything works, then your done.

On failure, go back as far as necessary

My Goal

- Enable you to read the books on integrated circuits
- Enable you to read papers (latest research)
- Correct misunderstandings on the topic
- Answer any questions you have on the chapters

Plan

Lectures:
Thursday at 10:15 - 12:00

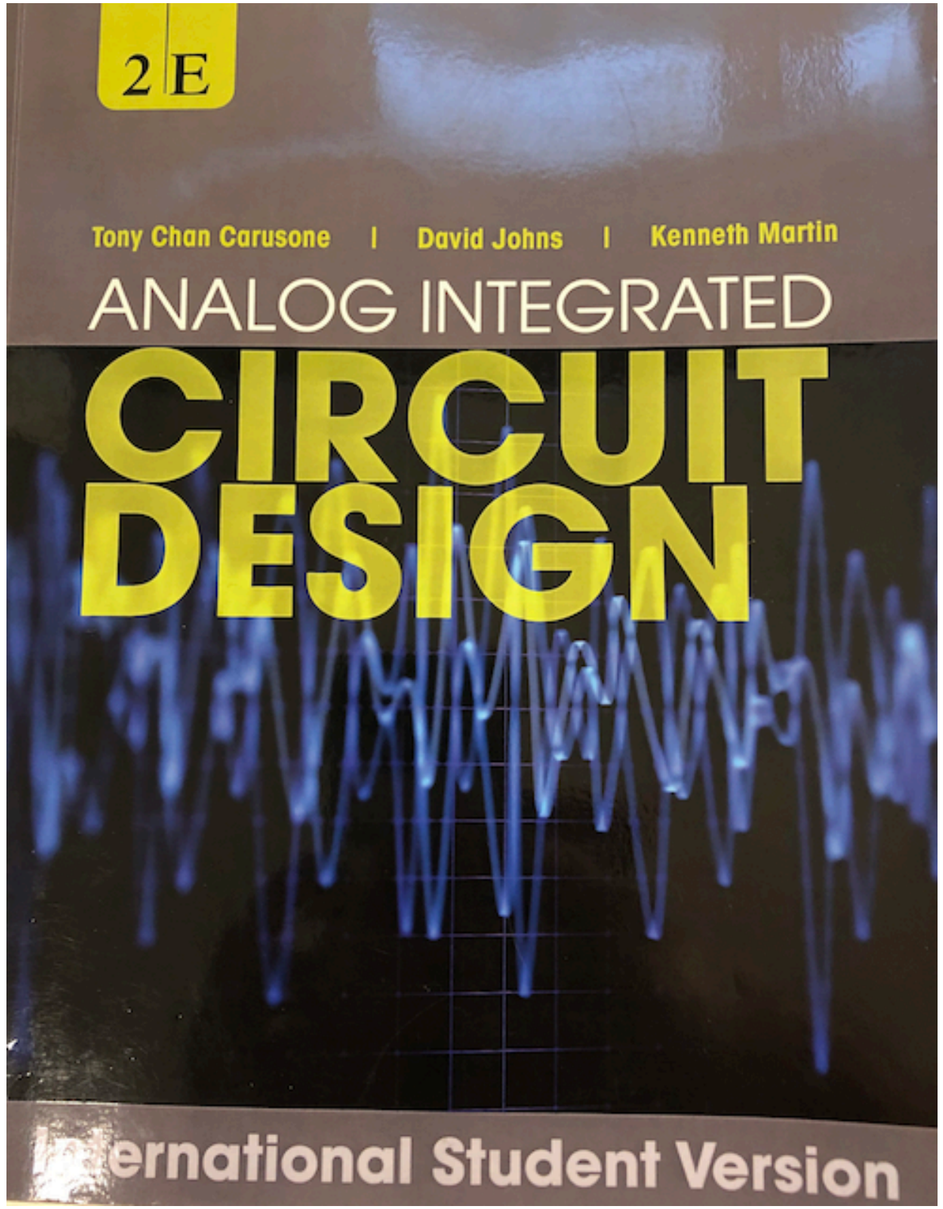
Read the introduction before the lectures at aic2025

The "lectures" will be Q & A's on the topic. If no questions, then I'll ramble on.

Project Hours:
Thursday at 12:15 - 14:00

Groups meet, and work on project.

- Description
- Time schedule
- Lecture plan
- Syllabus
- Youtube Videos
- AIC2024


2E
Tony Chan Carusone | David Johns | Kenneth Martin
ANALOG INTEGRATED
CIRCUIT
DESIGN
International Student Version

Exam

- May/June?
- 4 hours
- A - F grade (F = Fail)
- Counts for 55 % of the grade

Compulsory exercises

Follow: Sky130nm Tutorial

Submit link to your github repository on blackboard

For example, my repository:
JNW_EX_SKY130A

The exercise is designed to teach you everything the skills you need to do the project

Project

Counts for 45 % of the grade

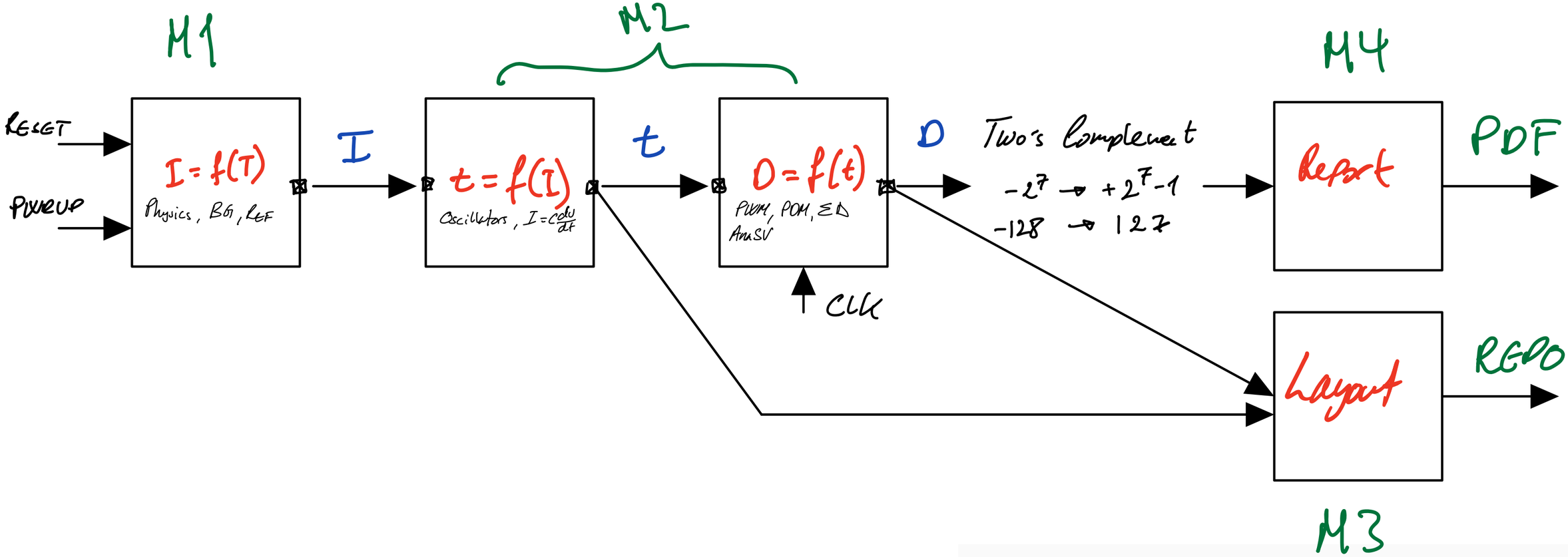
No exam without project.

Strict deadline 30'th of April. If you hand in 1'th of May at 00:00:01, then you fail the course.

JNW (2025)

"You can use logic to justify almost anything. That's its power. And its flaw." - Kathryn Janeway, Star Trek Voyager: Prime Factors

Design a integrated temperature sensor with digital read-out



Grading

Milestone	What does it mean	Condition for more than 0 points	Possible Points
M1 I=f(T)	Circuit that can convert a temperature into a current	SIM passing	10
M2 D=f(T)	Circuit that can convert from temperature into a digital value	DOC passing	20
M3 Layout	Layout of your circuit	DRC/LVS/GDS passing	20
M4 Report	Individual report	Uploaded to blackboard	48
Coolness	Extra points that I may choose to award		10
Total			108

Software

Open source software (xschem, ngspice, sky130A PDK, Magic VLSI, netgen)

Skywater 130 nm Tutorial

aicex

Lower your expectations on EDA software

Expect that you will spend at least 2π times more time than planned *(mostly due to software issues)*

Questions

Do

- google
- ask a someone in your class
- use the "øvingstime and labratorieøvelse" to talk to teaching assistants and hopefully me.
- come to the office (B311) on Thursday's

Thanks!