

# Introduction

Carsten Wulff, carsten@wulff.no

**Status:** 1.0

## I. WHO

My name is

Carsten Wulff [carstenw@ntnu.no](mailto:carstenw@ntnu.no)

I finished my Masters in 2002, and did a Ph.D on analog-to-digital converters finished in 2008.

Since that time, I've had a three axis in my work/hobby life.

I work at [Nordic Semiconductor](#) where I've been since 2008. The first 7 years I did analog design (ADCs, DC/DCs, GPIO). The next 7 years I was the Wireless Group Manager. The Wireless group make most of the analog and RF designs for Nordic's short-range products. Now I'm the IC Scientist, and focus on technical issues with our integrated circuits that occur before we go into volume production.

I work at [NTNU](#) where I did a part time postdoc from 2014 - 2017. From 2020 I've been working on and teaching [Advanced Integrated Circuits](#)

I have a hobby trying to figure out how to make a new analog circuit design paradigm. The one we have today with schematic/simulation/layout/verification/simulation is too slow

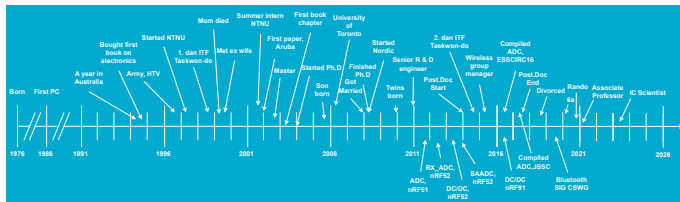


Figure 1: My life

## II. HOW I SEE OUR ROLES

In Figure 2 you can see how I think about the research universe. There are things we know to be possible, things that actually are impossible (travel back in time, breaking thermodynamics, travel with a speed beyond light).

Between the impossible, and the possible, lies the unknown. I consider our roles as follows:

**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

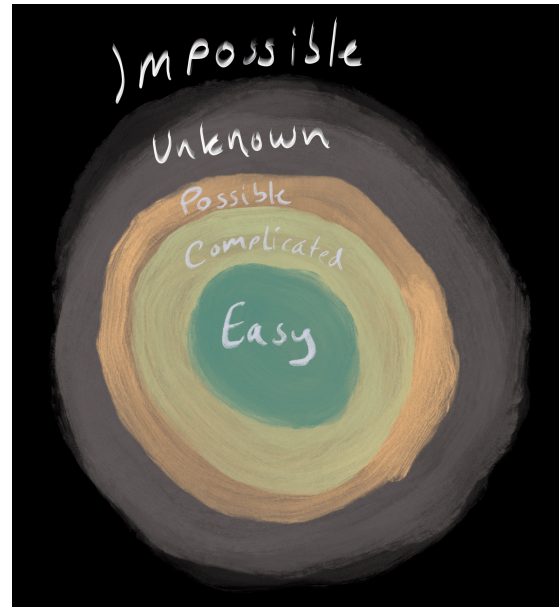


Figure 2: Research Universe

## III. I WANT YOU TO LEARN THE SKILLS NECESSARY TO MAKE YOUR OWN ICS

In 2020 the global integrated circuit market was **437.7 billion dollars!** The market is expected to grow to 1136 billion in 2028. Integrated circuits enable all technologies.

I will be dead in approximately 50 years, and will retire in approximately 20 years. Everything I know will be gone (except for the small pieces I've left behind in videos or written word)

Someone must take over, and to do that, they need to know most of what I know, and hopefully a bit more.

That's were some of you come in. Some of you will find integrated circuits interesting to make, and in addition, you have the stamina, patience, and brain necessary to learn some of the hardest topics in the world.

Making integrated circuits (that work reliably) is not rocket science, it's much harder.

## IV. THERE WILL ALWAYS BE ANALOG CIRCUITS, BECAUSE THE REAL WORLD IS ANALOG

In this course, we'll focus on analog ICs, because the real world is analog, and all ICs must have some analog components, otherwise they won't work.

The steps to make integrated circuits is split in two. We have an analog flow, and a digital flow, as shown in Figure 3.

It's rare to find a single human that do both flows well. Usually people choose, and I think it's based on what they like and their personality.

If you like the world to be ordered, with definite answers, then it's likely that you'll find the digital flow interesting.

If you're comfortable with not knowing, and an insatiable desire to understand how the world *really* works at a fundamental level, then it's likely that you'll find analog flow interesting.

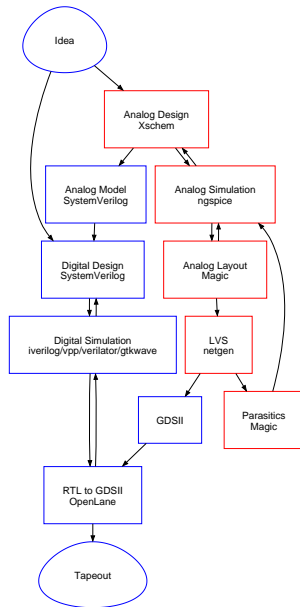


Figure 3: Analog and Digital design process

→

## V. WILL YOU TAPE-OUT AN IC?

Something that would make me really happy is if someone is able to tapeout an IC in this course.

It's now possible without signing an NDA or buying expensive software licenses.

In 2020 Google and Skywater joined forces to release a 130 nm process design kit to the public. In addition, they have fueled a renaissance of open source software tools.

Together with [Efabless](#) there are cheap alternatives, like [tinytapeout](#), which makes it possible for a private citizen to tape-out their own integrated circuit.

### A. What the team needs to know to design ICs

There are a multitude of tools and skills needed to design professional ICs. It's not likely that you'll find all the skills in one human, and even if you could, one human does not have sufficient bandwidth to design ICs with all it's aspects in a reasonable timeline

That is, unless we can find a way to make ICs easier.

The skills needed are

- *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**, build 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

### B. Zen of IC design (stolen from Zen of Python)

When you learn something new, it's good to listen to someone that has done whatever it is before.

Here is some guiding principles that you'll likely forget.

- 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.

### C. IC design mantra

To copy an old mantra I have on learning programming

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

And run the perl program

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

#### D. 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*

#### VI. MY GOAL

Don't expect that I'll magically take information and put it inside your head, and you'll suddenly understand everything about making ICs.

**You are the one that must teach yourself everything.**

I consider my role as a guide, similar to a mountain guide. I can't carry you up the mountain, you need to walk up the mountain, but I know the safe path to take and increase the likelihood that you'll come back alive.

I want to:

- 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

I'm not a mind reader, I can't see inside your head. That means, you must ask questions. Only by your questions can I start to understand what pieces of information is missing from your head, or maybe somehow correct your understanding.

At the same time, and similar to a mountain guide, you should not assume I'm always right. I'm human, and I will make mistakes. And maybe you can correct my understanding of something. All I care about is to *really* understand how the

world works, so if you think my understanding is wrong, then I'll happily discuss.

#### VII. SYLLABUS

The syllabus will be from Analog Integrated Circuit Design (CJM) and Circuits for all seasons.

These lecture notes are a supplement to the book. I try to give some background, and how to think about electronics. It's not my goal to repeat information that you can find in the book.

Buy a hard-copy of the book if you don't have that. Don't expect to understand the book by reading the PDF.

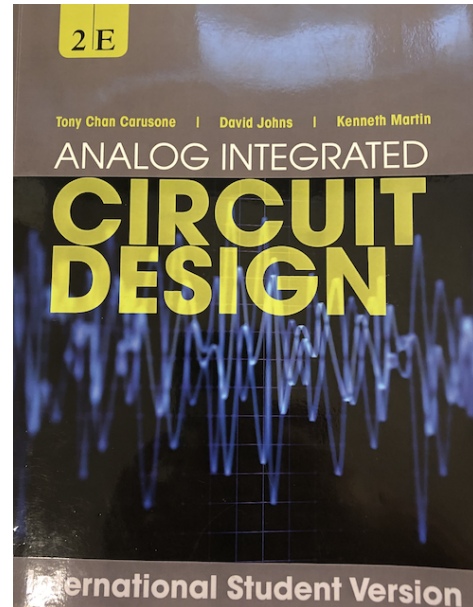


Figure 4: The book we'll use

#### VIII. PROJECT JNW (2025)

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

The project for 2025 is to

**Design a integrated temperature sensor with digital read-out**

An outline of the plan is shown below.

At the end of the project you will have a function that converts temperature to a digital value.

$$D = f_0(T)$$

I've broken down the challenge into three steps, first convert Temperature into a current

$$I = f_1(T)$$

Then convert current into a time

$$t = f_2(I)$$

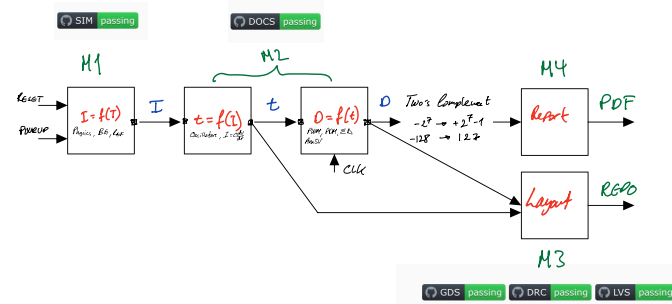
then time to digital

$$D = f_3(t) = f_3(f_2(f_1(T))) = f_0(T)$$

The third milestone is the layout, while the fourth milestone is the report.

You can find an example of last years designs at [cnr\\_gr02\\_sky130nm](#)

You will be using a repository on github for all your design data. In that repository I’ve made it possible to run github actions, or github workflows. For each of the milestones there are associated workflows (SIM/DOCS/GDS/DRC/LVS).



**Milestone 0:** The zero milestone is not really part of the project, but it does introduce you too how you will work with the files in the project. It’s important that you do this right away. To complete the milestone, upload a link to blackboard with your github repository for the tutorial [Skywater 130 nm Tutorial](#)

**Milestone 1:** The first milestone is to make a circuit that can convert from a temperature, to a current that is proportional to temperature. You will run a simulation on github that demonstrates that the circuit works. That is the SIM workflow.

**Milestone 2:** In the second milestone you will complete the schematic design of the circuit, and possibly also do some SystemVerilog to demonstrate that you get a digital value out that is proportional to temperature. Here, the simulations on github may be too long, so it’s sufficient to describe the circuit, and how it works in detail in the documentation. This is the DOC workflow.

**Milestone 3:** The third milestone, making the layout, is optional, however, it will be impossible to get an A without getting some points from the layout milestone. Once the layout is complete, I expect that the design rule checks (DRC), Layout versus Schematic (LVS), and GDS (stream out to a [GDSII](#) file) is passing on github.

**Milestone 4:** I will force you to work in groups. As such, it may be that some contribute more than others. To ensure that the grading is fair, the report will be individual. It’s OK to share figures, tables, and so on, but the PDF shall be written by you and you alone.

A. Grading

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

B. Group dynamics

How you work together is important. No-one can do everything by them self. I know from experience it can be magical when bright brains come together. The collective brain can be smarter, better, faster, than anyone in the group.

That’s why I think it’s important not to just work in groups, but also focus on how we work in groups.

A group shall be maximum 4 members. There must be at least 3 that don’t know each-other that well.

The group will meet once per week in the exercise hours.  
1) Check-in: All group session must start with a Check-in (10 minutes)

Some example questions could be

- Share one thing that is going on in your life (personal or professional.)
- What is one thing that you are grateful for right now?
- What is something funny that happened?

Some examples answers could be:

- My dog died yesterday, so I’m not feeling great today.
- I woke up early, had an omelet, and went running, so I feel motivated and fantastic.
- I feel *blaaaah* today, motivation is lacking.
- I went running yesterday and did not discover before I got home that I’d forgotten to put my pants on, even though it was -10 C.

The point of this exercise is to get to know each other a bit, and attempt to create psychological safety in the group.

IX. SOFTWARE

We’ll use professional Open source software (xschem, ngspice, sky130A PDK, Magic VLSI, netgen)

I’ve made a rather detailed (at least I think so myself) tutorial on how to make a current mirror with the open source tools. I strongly recommend you start with that first.

[Skywater 130 nm Tutorial](#)

I’ve also made some more complex examples, that can be found at the link below. There are digital logic cells, standard transistors, and few other blocks.

[aicex](#)



**Carsten Wulff** received the M.Sc. and Ph.D. degrees in electrical engineering from the Department of Electronics and Telecommunication, Norwegian University of Science and Technology (NTNU), in 2002 and 2008, respectively. During his Ph.D. work at NTNU, he worked on open-loop sigma-

delta modulators and analog-to-digital converters in nanoscale CMOS technologies. In 2006-2007, he was a Visiting Researcher with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, Canada. Since 2008 he's been with Nordic Semiconductor in various roles, from analog designer, to Wireless Group Manager, to currently Principle IC Scientist. He's also an Adjunct Associate Professor at NTNU. His present research interests includes analog and mixed-signal CMOS design, design of high-efficiency analog-to-digital converters and low-power wireless transceivers. He is the developer of Custom IC Compiler, a general purpose integrated circuit compiler.