Advanced Integrated Circuits

Lecture Notes 2023

Carsten Wulff

Built on Sat Jul 27 10:18:49 UTC 2024
from 56a207308234d282a7b11ade8cd8183d530d18c7

Contents

Chapter 1

Introduction

In the spring of 2023 I lectured Advanced Integrated Circuits for the second time. I have an inherent need to make things better, and the course is no different.

In the first round I noticed that very little of what I had on slides, or said in lectures, made it into the student brain. That annoyed me, and I realized that probably a few things needed to change.

I needed to make the content more available, and not assume that students actually remembered anything from previous years. I also got a suggestion from Jonathan to flip the class room.

As a result, I recorded a video, made the video available online, and encouraged students to ask questions in the lectures. In the lectures I tried, to a larger degree, to explain on the white-board, and not in slides.

Sometimes I had too much material, sometimes I had too little, sometimes I went on a tangent unrelated to the topic at hand. Sometimes I felt great after the lecture, sometimes I felt crap.

I also learnt that even though I felt crap, there could be students that found my crap lecture interesting (thanks JS).

It's hard to gauge whether 2023 was better than 2022, but I think so. At least I got one student comment that the free-form white-board lectures were better than slides (thanks J).

To prepare for the lectures, and to try and remember what to do next year, I ended up expanding on the lecture notes, which were made available online at analogicus.com/aic2023.

The lecture notes were partly for the students, but mostly as a love letter to my future self.

I realized during the first years of teaching that each `lecture` was a `race`, and each `topic` is a different `distance`. Unless I got into perfect shape that exactly matched to the distance, then the lecture would be crap.

Next year, I hope that by reading the notes, and watching the videos, I can shorten my training time for each lecture to a few hours, and at the same time make the lectures better than ever.

I love programming and automation. Not much makes me more happy than using the same source (the slide markdowns), to generate the lecture notes, to translate into the book your looking at right now.

Thanks to Fredrik and Jonathan that helped me with the course during the 2023 semester.

If you find an error in what I've made, then fork aic2023, fix , commit, push and create a pull request. That way, we use the global brain power most efficiently, and avoid multiple humans spending time on discovering the same error.

Chapter 2

Refresher

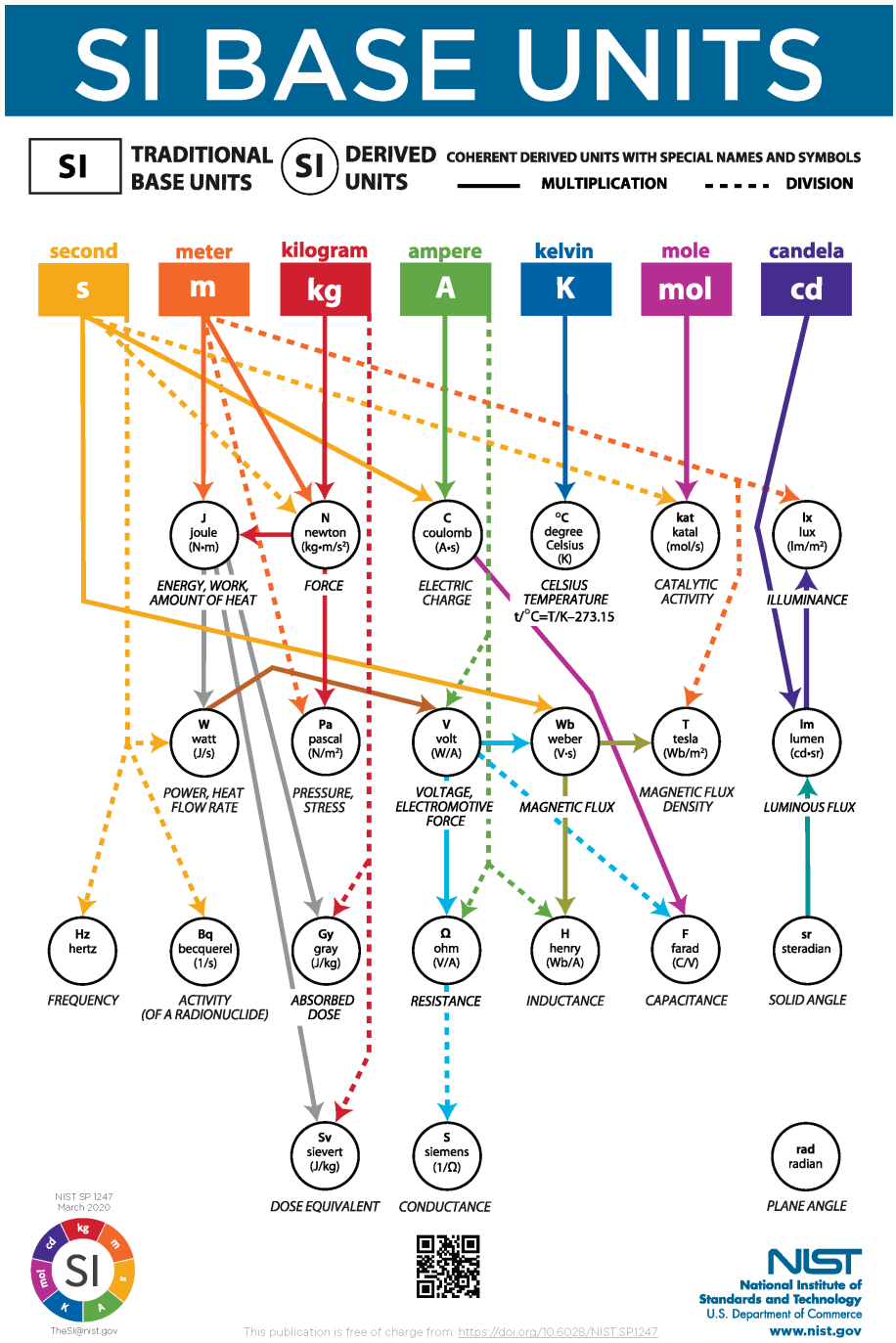
2.1 There is a standard unit of measurement

All known physical quantities are derived from 7 base units (SI units) , second (s), meter (m), kg (kilogram), ampere (A), kelvin (K), candela (cd).

All other units (for example volts), are derived from the base units.

I don't go around remembering all of them, they are easily available online. When you forget the equation for charge (Q), voltage (V) and capacitance (C), look at the units below, and you can see it's $Q = CV$[1]

[1]Although you do have to keep your symbols straight. We use "C" for Capacitance, but C can also mean Columbs. Context matters.



2.2 Electrons

Electrons are fundamental, they cannot (as far as we know), be divided into smaller parts. Explained further in the standard model of particle physics

Electrons have a negative charge of $q \approx 1.602 \times 10^{-19}$. The proton a positive charge. The two charges balance exactly! If you have a trillion electrons and a trillion protons inside a volume, the net external charge will be 0 (assuming we measure from some distance away). I find this fact absolutely incredible. There must be a fundamental connection between the charge of the proton and electron. It's insane that

the charges balance out so exactly.

All electrons are the same, although the quantum state can be different.

An electron cannot occupy the same quantum state as another. This rule that applies to all Fermions (particles with spin of 1/2)

The quantum state of an electron is fully described by it's spin, momentum (p) and position in space (r).

2.3 Probability

The probability of finding an electron in a state as a function of space and time is

$$P = |\psi(r,t)|^2$$

, where ψ is named the probability amplitude, and is a complex function of space and time. In some special cases, it's

$$\psi(r,t) = Ae^{i(kr-\omega t)}$$

, where A is complex number, k is the wave number, r is the position vector from some origin, ω is the frequency and t is time.

The energy is $E = \hbar\omega$, where $\hbar = h/2\pi$ and h is Planck Constant and the momentum is $p = \hbar k$

2.4 Uncertainty principle

We cannot, with ultimate precision, determine both the position and the momentum of a particle, the precision is

$$\sigma_x \sigma_p \geq \frac{\hbar}{2}$$

From the uncertainty principle we can actually estimate the size of the atom

2.5 States as a function of time and space

The time-evolution of the probability amplitude is

$$i\hbar\frac{d}{dt}\psi(r,t) = H\psi(r,t)$$

, where H is named the Hamiltonian matrix, or the energy matrix or (if I understand correctly) the amplitude matrix of the probability amplitude to change from one state to another.

For example, if we have a system with two states, a simplified version of two electrons shared between two atoms, as in H_2, or hydrogen gas, or co-valent bonds, then the Hamiltonian is a 2 x 2 matrix. And the ψ is a vector of $[\psi_1, \psi_2]$

Computing the solution to the Schrodinger Equation can be tricky, because you must know the number of relevant states to know the vector size of ψ and the matrix size of H. In addition, the H can be a function of time and space (I think).

Compared to the equations of electric fields, however, Schrodinger is easy, it's a set of linear differential equations.

2.6 Allowed energy levels in atoms

Solutions to Schrodinger result in quantized energy levels for an electron bound to an atom.

Take hydrogen, the electron bound to the proton can only exists in quantized energy levels. The lowest energy state can have two electrons, one with spin up, and one with spin down.

From Schrodinger you can compute the energy levels, which most of us did at some-point, although now, I can't remember how it was done. That's not important. The important is to internalize that the energy levels in bound electrons are discrete.

Electrons can transition from one energy level to another by external influence, i.e temperature, light, or other.

The probability of a state transition (change in energy) can be determined from the probability amplitude and Schrodinger.

2.7 Allowed energy levels in solids

If I have two silicon atoms spaced far apart, then the electrons can have the same spin and same momentum around their respective nuclei. As I bring the atoms closer, however, the probability amplitudes start to interact (or the dimensions of the Hamiltonian matrix grow), and there can be state transitions between the two electrons.

The allowed energy levels will split. If I only had two states interacting, the Hamiltonian could be

$$H = \begin{bmatrix} A & 0 \\ 0 & -A \end{bmatrix}$$

and the new energy levels could be

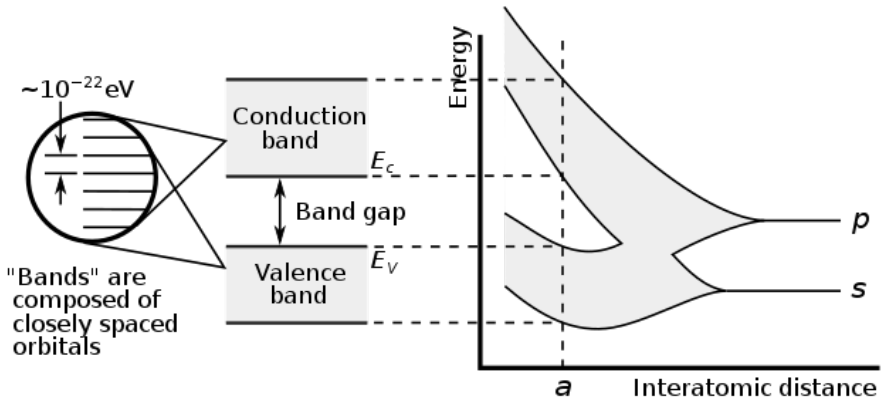
$$E_1 = E_0 + A$$

and

$$E_2 = E_0 - A$$

In a silicon crystal we can have trillions of atoms, and those that are close, have states that interact. **That's why crystals stay solids**. All chemical bonds are states of electrons interacting! Some are strong (co-valent bonds), some are weaker (ionic bonds), but it's all quantum states interacting.

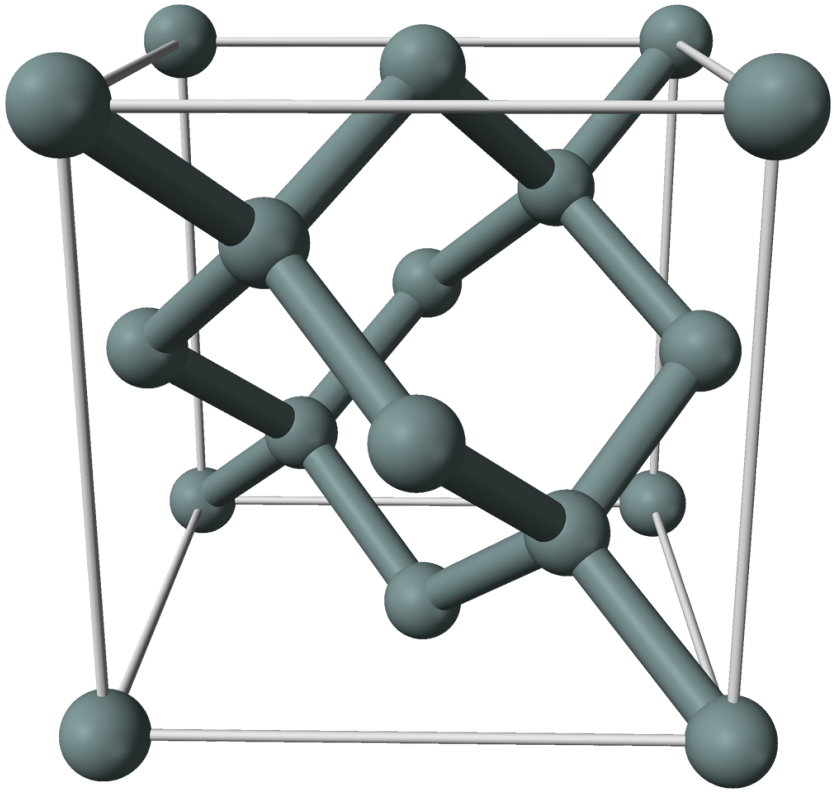
The discrete energy levels of the electron transition into bands of allowed energy states.



For a crystal, the allowed energy bands is captured in the band structure

2.8 Silicon Unit Cell

A silicon crystal unit cell is a diamond faced cubic with 8 atoms in the corners spaced at 0.543 nm, 6 at the center of the faces, and 4 atoms inside the unit cell at a nearest neighbor distance of 0.235 nm.



2.9 Valence band and Conduction band

The full band structure of a silicon unit cell is complicated. For bulk silicon we simplify, and we think of two bands. In the conduction band (E_C) is the lowest energy where electrons are free (not bound to

atoms). The valence band (E_V) is the highest band where electrons are bound to silicon atoms.

The difference between E_C and E_V is a property of the material we've named the band gap.

$$E_G = E_C - E_V$$

2.10 Metals

In metals, the band splitting of the energy levels causes the valence band and conduction band to overlap. As such, electrons can easily transition between bound state and free state. As such, electrons in metals are shared over large distances, and there are many electrons readily available to move under an applied field, or difference in electron density. That's why metals conduct well.

2.11 Insulators

In insulating materials the difference between the conduction band and the valence band is large. As a result, it takes a large energy to excite electrons to a state where they can freely move.

That's why glass is transparent to optical frequencies. Visible light does not have sufficient energy to excite electrons from a bound state.

That's also why glass is opaque to ultra-violet, which has enough energy to excite electrons out of a bound state.

Based on these two pieces of information you could estimate the bandgap of glass.

2.12 Semiconductors

In a silicon the bandgap is lower than an insulator, approximately $E_G = 1.12\ eV$ for silicon.

At room temperature, that allows a small number of electrons to be excited into the conduction band, leaving behind a "hole" in the valence band.

2.13 Fermi level

From Wikipedia's Fermi level

> In band structure theory, used in solid state physics to analyze the energy levels in a solid, the Fermi level can be considered to be a hypothetical energy level of an electron, such that at thermodynamic equilibrium this energy level would have a 50% probability of being occupied at any given time

The Fermi level is closely linked to the Fermi-Dirac distribution

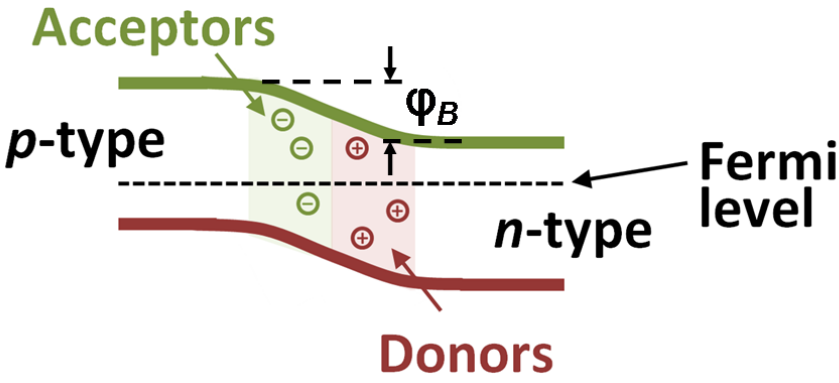
$$f(E) = \frac{1}{e^{(E-E_F)/kT} + 1}$$

If the energy of the state is more than a few kT away from the Fermi-level, then

$$f(E) \approx e^{(E_F-E)/kT}$$

2.14 Band diagrams

A band diagram or energy level diagrams shows the conduction band energy and valence band energy as a function of distance in the material.



The horizontal axis is the distance, the vertical axis is the energy.

The figure shows a PN-junction

2.15 Density of electrons/holes

There are two components needed to determine how many electrons are in the conduction band. The density of available states, and the probability of an electron to be in that quantum state.

The probability is the Fermi-Dirac distribution. The density of available states is a complicated calculation from the band-structure of silicon. See Diodes for details.

$$n_e = \int_{E_C}^{\infty} N(E) f(E) dE$$

The Fermi level is assumed to be independent of energy level, so we can write

$$n_e = e^{E_F/kT} \int_{E_C}^{\infty} N(E) e^{-E/kT} dE$$

for the density of electrons in the conduction band.

2.16 Fields

There are equations that relate electric field, magnetic field, charge density and current density to each-other.

The equations

$$\oint_{\partial\Omega} \mathbf{E} \cdot d\mathbf{S} = \frac{1}{\epsilon_0} \iiint_V \rho \cdot dV$$

,relates net electric flux to net enclosed electric charge

$$\oint_{\partial\Omega} \mathbf{B} \cdot d\mathbf{S} = 0$$

,relates net magnetic flux to net enclosed magnetic charge

$$\oint_{\partial\Sigma} \mathbf{E} \cdot d\ell = -\frac{d}{dt} \iint_{\Sigma} \mathbf{B} \cdot d\mathbf{S}$$

,relates induced electric field to changing magnetic flux

$$\oint_{\partial\Sigma} \mathbf{B} \cdot d\ell = \mu_0 \left(\iint_{\Sigma} \mathbf{J} \cdot d\mathbf{S} + \epsilon_0 \frac{d}{dt} \iint_{\Sigma} \mathbf{E} \cdot d\mathbf{S} \right)$$

,relates induced magnetic field to changing electric flux and to current

These are the Maxwell Equations, and are non-linear time dependent differential equations.

Under the best of circumstances they are fantastically hard to solve! But it's how the real world works.

The permittivity of free space is defined as

$$\epsilon_0 = \frac{1}{\mu_0 c^2}$$

, where c is the speed of light, and μ_0 is the vacuum permeability, which, in SI units, is now

$$\mu_0 = \frac{2\alpha}{q^2} \frac{h}{c}$$

, where α is the fine structure constant.

2.17 Voltage

The electric field has units voltage per meter, so the electric field is the derivative of the voltage as a function of space.

$$E = \frac{dV}{dx}$$

2.18 Current

Current has unit A and charge C has unit As, so the current is the number of charges passing through a volume per second.

The current density J has units A/m^2 and is often used, since we can multiply by the surface area of a conductor, if the current density is uniform.

$$I = A \times J$$

2.19 Drift current

Charges in an electric field will give rise to a drift current.

We know from Newtons laws that force equals mass times acceleration

$$\vec{F} = m\vec{a}$$

If we assume a zero, or constant magnetic field, the force on a particle is $\vec{F} = q\vec{E}$

The current density is then

$$\vec{J} = q\vec{E} \times n \times \mu$$

where n is the charge density, and μ is the mobility (how easily the charges move) and has units $[m^2/Vs]$

Assuming $E = V/m$, we could write

$$J = \frac{C}{m^3}\frac{V}{m}\frac{m^2}{Vs} = \frac{C}{s}m^{-2}$$

So multiplying by an area $A = Bm^2$

$$I = qn\mu BV$$

and we can see that the conductance $G = qn\mu B$, and since $G = 1/R$, where R is the resistance, we have

$$I = GV \Rightarrow V = RI$$

Or Ohms law

2.20 Diffusion current

A difference in charge density will give rise to a diffusion current, and the current density is

$$J = -qD_n\frac{d\rho}{dx}$$

,where D_n is a diffusion constant, and ρ is the charge density.

I struggled with the concepts diffusion current and drift current for a long time. Why are there two types of current? It was when I read The Schrödinger Equation in a Classical Context: A Seminar on Superconductivity I realised that the two types of current come directly from the Schrodinger equation, there is one component related to the electric field (potential energy) and a component related to the momentum (kinetic energy).

In the absence of an electric field electrons will still jump from state to state set by the probabilities of the Hamiltonian. If there are more electrons in an area, then it will seem like there is an average movement of charges away from that area. That's how I think about the equation above.

2.21 Currents in a semiconductor

Both electrons, and holes will contribute to current.

Electrons move in the conduction band, and holes move in the valence band.

Both holes and electrons can only move if there are available quantum states.

For example, if the valence band is completely filled (all states filled), then there can be no current.

To compute the total current in a semiconductor one must compute

$$I = I_{n_{drift}} + I_{n_{diffusion}} + I_{p_{drift}} + I_{p_{diffusion}}$$

where n denotes electrons, and p denote holes.

2.22 Resistors

We can make resistors with metal and silicon (a semiconductor)

In metal the dominant carrier depends on the metal, but it's usually electrons. As such, one can often ignore the hole current.

In a semiconductor the dominant carrier depends on the Fermi level in relation to the conduction band and valence band. If the Fermi level is close to the valence band the dominant carrier will be holes. If the Fermi level is close to the conduction band, the dominant carrier will be electrons.

That's why we often talk about "majority carriers" and "minority carriers", both are important in semiconductors.

2.23 Capacitors

A capacitor resists a change in voltage.

$$I = C\frac{dV}{dt}$$

and store energy in an electric field between two conductors with an insulator between.

2.24 Inductors

An inductor resist a change in current.

$$V = L\frac{dI}{dt}$$

and store energy in the magnetic fields in a loop of a conductor.

2.25 Diodes

See Diodes for a long explanation.

Assume we in the silicon lattice introduce a dopant, for example phosphorus with one more electron than silicon. Four of the electrons will be in co-valent bonds with the silicon lattice, while the last electron is loosely bound to the phosphorus atom. We call that a donor.

A donor will shift the Fermi level towards the conduction band, and as such, there will be more free electrons, as long as there is sufficient energy to break the loose electron bond.

A atom with one less electron, for example Boron, can be introduced in the same way, and is called an acceptor.

Most of the charges in a p-type (acceptors) silicon will be holes, while in n-type it will be electrons.

The interesting thing happens when p-type and n-type are in contact. Since we've shifted the Fermi level of the two silicon types in opposite direction there will be an energy difference. That energy difference must be equalized, as such, over time, the Fermi level of the two types will align, and a junction will form, with few free charges, a depletion region.

The conduction band and valence band will bend, and we now have a barrier for charge transport, a built-in electric field.

Chapter 3

Diodes

For the source of this paper, see the markdown.

3.1 Why

Diodes are a magical [1] semiconductor device that conduct current in one direction. It's one of the fundamental electronics components, and it's a good idea to understand how they work.

If you don't understand diodes, then you won't understand transistors, neither bipolar, or field effect transistors.

A useful feature of the diode is the exponential relationship between the forward current, and the voltage across the device.

To understand why a diode works it's necessary to understand the physics behind semiconductors.

This paper attempts to explain in the simplest possible terms how a diode works [2]

3.2 Silicon

Integrated circuits use single crystalline silicon. The silicon crystal is grown with the Czochralski method which forms a ingot that is cut into wafers. The wafer is a regular silicon crystal, although, it is not perfect.

A silicon crystal unit cell, as seen in Figure 1 is a diamond faced cubic with 8 atoms in the corners spaced at 0.543 nm, 6 at the center of the faces, and 4 atoms inside the unit cell at a nearest neighbor distance of 0.235 nm.

[1] It doesn't stop being magic just because you know how it works. Terry Pratchett, The Wee Free Men
[2] Simplify as much as possible, but no more. Einstein

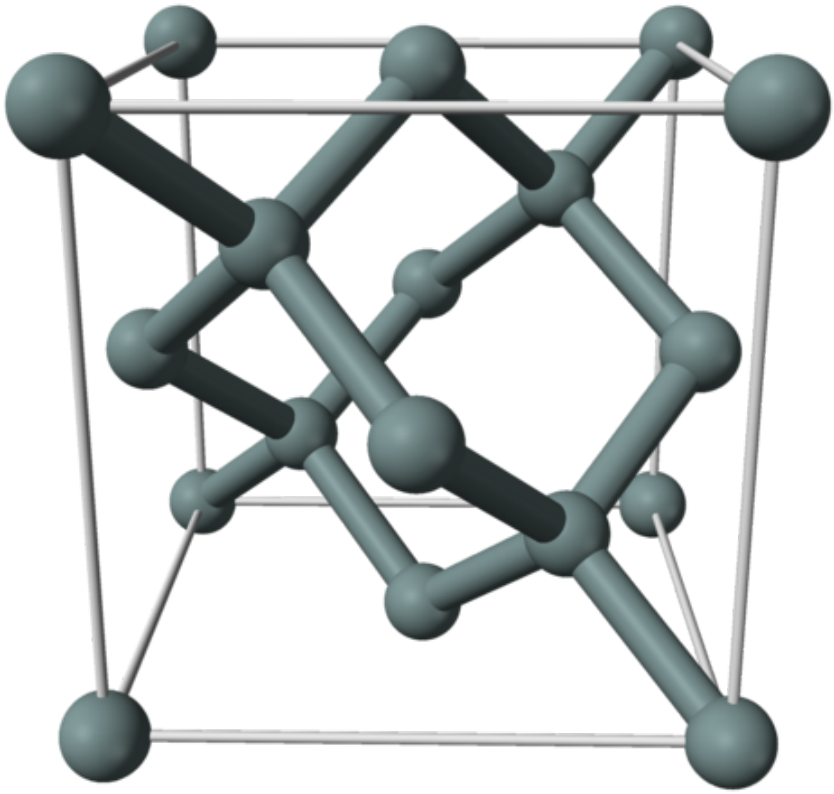


Figure 1: Silicon crystal unit cell

As you hopefully know, the energy levels of an electron around a positive nucleus are quantized, and we call them orbitals (or shells). For an atom far away from any others, these orbitals, and energy levels are distinct. As we bring atoms closer together, the orbitals start to interact, and in a crystal, the distinct orbital energies split into bands of allowed energy states. No two electrons, or any Fermion (spin of 1/2), can occupy the same quantum state. We call the outermost "shared" orbitial, or band, in a crystal the valence band. Hence covalent bonds.

If we assume the crystal is perfect, then at 0 Kelvin all electrons will be part of covalent bonds. Each silicon atom share 4 electrons with its neighbors. I think what we really mean when we say "share 4 electrons" is that the wave-functions of the outer orbitals interact, and we can no longer think of the orbitals as belonging to either of the silicon nuclei. All the neighbors atoms "share" electrons, and nowhere is there an vacant state, or a hole, in the valence band. If such a crystal were to exist, it would not conduct any current, as the charges cannot move.

In a atom, or a crystal, there are also higher energy states where the carriers are "free" to move. We call these energy levels, or bands of energy levels, conduction bands. In singular form "conduction band", refers to the lowest available energy level where the electrons are free to move.

Due to imperfectness of the silicon crystal, and non-zero temperature, there will be some electrons that achieve sufficient energy to jump to the conduction band. The electrons in the conduction band leave vacant states, or holes, in the valence band.

Electrons can move both in the conduction band, as free electrons, and in the valence band, as a positive particle, or hole.

3.3 Intrinsic carrier concentration

The intrinsic carrier concentration of silicon, or how many free electrons and holes at a given temperature, is given by

$$n_i = \sqrt{N_c N_v} e^{-\frac{E_g}{2kT}} \tag{1}$$

where E_g is the bandgap energy of silicon (approx 1.12 eV), k is Boltzmann's constant, T is the temperature in Kelvin, N_c is the density of states in conduction band, and N_v is the density of states in the valence band.

The density of states are

$$N_c = 2\left[\frac{2\pi kTm_n^*}{h^2}\right]^{3/2} \quad N_v = 2\left[\frac{2\pi kTm_p^*}{h^2}\right]^{3/2}$$

where h is Planck's constant, m_n^* is the effective mass of electrons, and m_p^* is the effective mass of holes.

In [@cjm11] they claim the intrinsic carrier concentration is a constant, although they do mention n_i doubles every 11 degrees Kelvin. In BSIM 4.8 [@bsim] n_i is

$$n_i = 1.45e10\frac{TNOM}{300.15}\sqrt{\frac{T}{300.15}}\exp^{21.5565981-\frac{E_g}{2kT}}$$

Comparing the three models in Figure 2, we see the shape of BSIM and the full equation is almost the same, while the "doubling every 11 degrees" is just wrong.

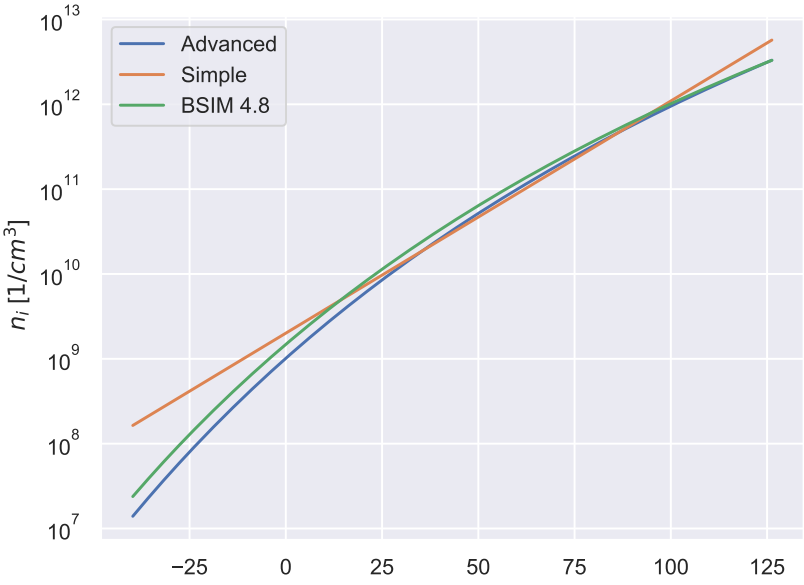


Figure 2: Intrinsic carrier concentration versus temperature

At room temperature the intrinsic carrier consentration is approximately $n_i = 1 \times 10^{16}$ carriers/m^3.

That may sound like a big number, however, if we calculate the electrons per um^3 it's $n_i = \frac{1\times10^{16}}{(1\times10^6)^3}$ carriers/μm$^3 < 1$, so there are really not that many free carriers in intrinsic silicon.

But where does Eq (**??**) come from? I find it unsatisfying if I don't understand where things come from. I like to understand why there is an exponential, or effective mass, or Planck's constant. If you're like me, then read the next section. If you don't care, and just want to memorize the equations, or indeed the number of intrinsic carrier concentration number at room temperature, then skip the next section.

3.4 It's all quantum

There are two components needed to determine how many electrons are in the conduction band. The density of available states, and the probability of an electron to be in that quantum state.

For the density of states we must turn to quantum mechanics. The probability amplitude of a particle can be described as

$$\psi = Ae^{i(k\mathbf{r}-\omega t)}$$

where k is the wave number, and ω is the angular frequency, and $\mathbf{r}$ is a spatial vector.

In one dimension we could write $\psi(x,t) = Ae^{i(kx-\omega t)}$

In classical physics we described the Energy of the system as

$$\frac{1}{2m}p^2 + V = E$$

where $p = mv$, m is the mass, v is the velocity and V is the potential.

In the quantum realm we must use the Schrodinger equation to compute the time evolution of the Energy, in one space dimension

$$-\frac{\hbar^2}{2m}\frac{\partial^2}{\partial^2 x}\psi(x,t) + V(x)\psi(x,t) = i\hbar\frac{\partial}{\partial t}\psi(x,t)$$

where m is the mass, V is the potential, $\hbar = h/2\pi$.

We could rewrite the equation above as

$$\widehat{H}\psi(x,t) = i\hbar\frac{\partial}{\partial t}\psi(x,t) = \widehat{E}\psi(x,t)$$

where

$$\widehat{H}$$

is sometimes called the *Hamiltonian* and is an operator, or something that act on the wave-function. I recently read Feynman's Lectures on Physics, and Feynman called the Hamiltonian the *Energy Matrix* of a system. I like that better. The

$$\widehat{E}$$

is the energy operator, something that operates on the wave-function to give the Energy.

We could re-arrange

$$[\widehat{H} - \widehat{E}]\psi(r,t) = 0$$

This is an equation with at least 5 unknowns, the space vector in three dimensions, time, and the energy matrix

$$\widehat{H}$$

. It turns out, that the energy matrix depends on the system. The energy matrix further up is for one free electron. For an atom, the energy matrix will have more dimensions to describe the possible quantum states.

I was watching Quantum computing in the 21st Century and David Jamison mentioned that the largest system we could today compute would be a system with about 30 electrons. So although we know exactly how the equations of quantum mechanics appear to be, and they've proven extremely successful, we must make simplifications before we can predict how electrons behave in complicated systems like the silicon lattice with approximately 0.7 trillion electrons per cube micro meter. You can check the calculation

$$\left[\frac{1\ \mu\text{m}}{0.543\ \text{nm}}\right]^3 \times 8 \text{ atoms per unit cell} \times 14 \text{ electrons per atom}$$

3.4.1 Density of states

To compute "how many Energy states are there per unit volume in the conduction band", or the "density of states", we start with the three dimensional Schrodinger equation for a free electron

$$-\frac{\hbar^2}{2m}\nabla^2\psi = E\psi$$

I'm not going to repeat the computation here, but rather paraphrase the steps. You can find the full derivation in Solid State Electronic Devices.

The derivation starts by computing the density of states in the k-space, or momentum space,

$$N(dk) = \frac{2}{(2\pi)^p}dk$$

Where p is the number of dimensions (in our case 3).

Then uses the band structure $E(k)$ to convert to the density of states as a function of energy $N(E)$. The simplest band structure, and a approxmiation of the lowest conduction band is

$$E(k) = \frac{\hbar^2 k^2}{2m^*}$$

where m^* is the effective mass of the particle. It is within this effective mass that we "hide" the complexity of the actual three-dimensional crystal structure of silicon.

The effective mass when we compute the density of states is

$$m^* = \frac{\hbar^2}{\frac{d^2E}{dk^2}}$$

as such, the effective mass depends on the localized band structure of the silicon unit cell, and depends on direction of movement, strain of the silicon lattice, and probably other things.

In 3D, once we use the above equations, one can compute that the density of states per unit energy is

$$N(E)dE = \frac{2}{\pi^2}\frac{{m^*}^{3/2}}{\hbar^2}E^{1/2}dE$$

In order to find the number of electrons, we need the probability of an electron being in a quantum state, which is given by the Fermi-Dirac distribution

$$f(E) = \frac{1}{e^{(E-E_F)/kT}+1} \tag{2}$$

where E is the energy of the electron, E_F is the Fermi level or checmical potential, k is Boltzmann's constant, and T is the temperature in Kelvin.

Fun fact, the Fermi level difference between two points is what you measure with a voltmeter.

If the $E - E_F > kT$, then we can start to ignore the $+1$ and the probability reduces to

$$f(E) = \frac{1}{e^{(E-E_F)/kT}} = e^{(E_F-E)/kT}$$

A few observiation on the Fermi-Dirac distribution. If the Energy of a particle is at the Fermi level, then $f(E) = \frac{1}{2}$, or a 50 % probability.

In a metal, the Fermi level lies within a band, as the conduction band and valence band overlap. As a result, there are a bunch of free electrons that can move around. Metal does not have the same type of covalent bonds as silicon, but electrons are shared between a large part of the metal structure. I would also assume that the location of the Fermi level within the band structure explains the difference in conductivity of metals, as it would determined how many electrons are free to move.

In an insulator, the Fermi level lies in the bandgap between valence band and conduction band, and usually, the bandgap is large, so there is a low probability of finding electrons in the conduction band.

In a semiconductor we also have a bandgap, but much lower energy than an insulator. If we have thermal equilibrium, no external forces, and we have an un-doped (intrinsic) silicon semiconductor, then the fermi level E_F lies half way between the conduction band edge E_C and the valence band edge E_V.

The bandgap is defined as the $E_C - E_V = E_g$, and we can use that to get $E_F - E_C = E_C - E_g/2 - E_C = -E_g/2$. This is why the bandgap of silicon keeps showing up in our diode equations.

The number of electrons per delta energy will then be given by $N_e dE = N(E)f(E)dE$, which can be integrated to get

$$n_e = 2\left(\frac{2\pi m^* kT}{h^2}\right)^{3/2} e^{(E_F-E_C)/kT}$$

For intrinsic silicon at thermal equlibrium, we could write

$$n_0 = 2\left(\frac{2\pi m^* kT}{h^2}\right)^{3/2} e^{-E_g/(2kT)} \tag{3}$$

As we can see, Equation (**??**) has the same coefficients and form as the computation in Equation (**??**). The difference is that we also have to account for holes. At thermal equilibrium and intrinsic silicon $n_i^2 = n_0 p_0$.

3.4.2 How to think about electrons (and holes)

I've come to the realization that to imagine electrons as balls moving around in the silicon crystal is a bad mental image.

For example, for a metal-oxide-semiconductor field effect transistor (MOSFET) it is not the case that the electrons that form the inversion layer under strong inversion come from somewhere else. They are already at the silicon surface, but they are bound in covalent bonds (there are literaly trillions of bound electrons in a typical transistor).

What happens is that the applied voltage at the gate shifts the energy bands close to the surface (or bends the bands in relation to the Fermi level), and the density of carriers in the conduction band in that location changes, according to the type of derivations above.

Once the electrons are in the conduction band, then they follow the same equations as diffusion of a gas, Fick's law of diffusion. Any charge concentration difference will give rise to a diffusion current given by

$$J_{\text{diffusion}} = -qD_n \frac{\partial \rho}{\partial x} \tag{4}$$

where J is the current density, q is the charge, ρ is the charge density, and D is a diffusion coefficient that through the Einstein relation can be expressed as $D = \mu kT$, where mobility $\mu = v_d/F$ is the ratio of drift velocity v_d to an applied force F.

To make matters more complicated, an inversion layer of a MOSFET is not in three dimensions, but rather a two dimensional electron gas, as the density of states is confined to the silicon surface. As such, we should not expect the mobility of bulk silicon to be the same as the mobility of a MOSFET transistor.

3.5 Doping

We can change the property of silicon by introducing other elements, something we've called doping. Phosphor has one more electron than silicon, Boron has one less electron. Injecting these elements into the silicon crystal lattice changes the number of free electron/holes.

These days, we usually dope with ion implantation, while in the olden days, most doping was done by diffusion. You'd paint something containing Boron on the silicon, and then heat it in a furnace to "diffuse" the Boron atoms into the silicon.

If we have an element with more electrons we call it a donor, and the donor concentration N_D.

The main effect of doping is that it changes the location of the Fermi level at thermal equilibirum. For donors, the Fermi level will shift closer to the conduction band, and increase the probabilty of free electrons, as determined by Equation (**??**).

Since the crystal now has an abundance of free electrons, which have negative charge, we call it n-type.

If the element has less electrons we call it an acceptor, and the acceptor concentration N_A. Since the crystal now has an abundance of free holes, we call it p-type.

The doped material does not have a net charge, however, as it's the same number of electrons and protons, so even though we dope silicon, it does remain neutral.

The doping concentrations are larger than the intrinsic carrier concentration, from maybe 10^{21} to 10^{27} carriers/m^3. To separate between these concentrations we use $p-, p, p+$ or $n-, n, n+$.

The number of electrons and holes in a n-type material is

$$n_n = N_D \ , \ p_n = \frac{n_i^2}{N_D}$$

and in a p-type material

$$p_p = N_A \ , \ n_p = \frac{n_i^2}{N_A}$$

In a p-type crystal there is a majority of holes, and a minority of electrons. Thus we name holes majority carriers, and electrons minority carriers. For n-type it's opposite.

3.6 PN junctions

Imagine an n-type material, and a p-type material, both are neutral in charge, because they have the same number of electrons and protons. Within both materials there are free electrons, and free holes which move around constantly.

Now imagine we bring the two materials together, and we call where they meet the junction. Some of the electrons in the n-type will wander across the junction to the p-type material, and visa versa. On the opposite side of the junction they might find an opposite charge, and might get locked in place. They will become stuck.

After a while, the diffusion of charges across the junction creates a depletion region with immobile charges. Where as the two materials used to be neutrally charged, there will now be a build up of negative charge on the p-side, and positive charge on the n-side.

3.6.1 Built-in voltage

The charge difference will create a field, and a built-in voltage will develop across the depletion region.

The density of free electrons in the conduction band is

$$n = \int_{E_C}^{\infty} N(E) f(E) dE$$

, where $N(E)$ is the density of states, and $f(E)$ is a probability of a electron being in that state (Equation (**??**)).

We could write the density of electrons on the n-side as

$$n_n = e^{E_{F_n}/kT} \int_{E_C}^{\infty} N_n(E) e^{-E/kT} dE$$

since the Fermi level is independent of the energy state of the electrons (I think).

The density of electrons on the p-side could be written as

$$n_p = e^{E_{F_p}/kT} \int_{E_C}^{\infty} N_p(E) e^{-E/kT} dE$$

If we assume that the density of states, $N_n(E)$ and $N_p(E)$ are the same, and the temperature is the same, then

$$\frac{n_n}{n_p} = \frac{e^{E_{F_n}/kT}}{e^{E_{F_p}/kT}} = e^{(E_{F_n} - E_{F_p})/kT}$$

The difference in Fermi levels is the built-in voltage multiplied by the unit charge.

$$E_{F_n} - E_{F_p} = q\Phi$$

and by substituting for the minority carrier concentration on the p-side we get

$$\frac{N_A N_D}{n_i^2} = e^{q\Phi_0/kT}$$

or rearranged to

$$\Phi_0 = \frac{kT}{q} ln \left(\frac{N_A N_D}{n_i^2} \right)$$

3.6.2 Current

The derivation of current is a bit involved, but let's try.

The hole concentration on the p-side and n-side could be written as

$$\frac{p_p}{p_n} = e^{-q\Phi_0/kT}$$

The negative sign is because the built in voltage is positive on the n-type side

Asssume that $-x_{p0}$ is the start of the junction on the p-side, and x_{n0} is the start of the junction on the n-side.

Assume that we lift the p-side by a voltage qV

Then the hole concentration would change to

$$\frac{p(-x_{p0})}{p(x_{n0})} = e^{q(V-\Phi_0)/kT}$$

while on the n-side the hole concentration would be

$$\frac{p(x_{n0})}{p_n} = e^{qV/kT}$$

So the excess hole concentration on the n-side due to an increase of V would be

$$\Delta p_n = p(x_{n0}) - p_n = p_n \left(e^{qV/kT} - 1 \right)$$

The diffusion current density, given by Equation (**??**) states

$$J(x_n) = -qD_p \frac{\partial \rho}{\partial x}$$

Thus we need to know the charge density as a function of x. I'm not sure why, but apparently it's

$$\partial \rho(x_n) = \Delta p_n e^{-x_n/L_p}$$

where L_p is a diffusion length. This equation smells to me like a simplified model of reality, I'm not sure how much it's based on fundamental physics.

Anyhow, we can now compute the current density, and need only compute it for $x_n = 0$, so you can show it's

$$J(0) = q \frac{D_p}{L_p} p_n \left(e^{qV/kT} - 1 \right)$$

which start's to look like the normal diode equation. The p_n is the minority concentration of holes on the n-side, which we've before estimated as $p_n = \frac{n_i^2}{N_D}$

We've only computed for holes, but there will be electron transport from the p-side to the n-side also.

We also need to multiply by the area of the diode to get current from current density. The full equation thus becomes

$$I = qAn_i^2 \left(\frac{1}{N_A} \frac{D_n}{L_n} + \frac{1}{N_D} \frac{D_p}{L_p} \right) \left[e^{qV/kT} - 1 \right]$$

where A is the area of the diode, D_n,D_p is the diffusion coefficient of electrons and holes and L_n,L_p is the diffusion length of electrons and holes.

Which we usually write as

$$I_D = I_S(e^{\frac{V_D}{V_T}} - 1), \text{ where } V_T = kT/q$$

3.6.3 Forward voltage temperature dependence

We can rearrange I_D equation to get

$$V_D = V_T \ln \left(\frac{I_D}{I_S} \right)$$

and at first glance, it appears like V_D has a positive temperature coefficient. That is, however, wrong.

First rewrite

$$V_D = V_T \ln I_D - V_T \ln I_S$$

$$\ln I_S = 2 \ln n_i + \ln Aq \left(\frac{D_n}{L_n N_A} + \frac{D_p}{L_p N_D} \right)$$

Assume that diffusion coefficient [3], and diffusion lengths are independent of temperature.

That leaves n_i that varies with temperature.

$$n_i = \sqrt{B_c B_v} T^{3/2} e^{\frac{-E_g}{2kT}}$$

where

$$B_c = 2 \left[\frac{2\pi k m_n^*}{h^2} \right]^{3/2} \quad B_v = 2 \left[\frac{2\pi k m_p^*}{h^2} \right]^{3/2}$$

$$2 \ln n_i = 2 \ln \sqrt{B_c B_v} + 3 \ln T - \frac{V_G}{V_T}$$

with $V_G = E_G/q$ and inserting back into equation for V_D

$$V_D = \frac{kT}{q} (\ell - 3 \ln T) + V_G$$

Where ℓ is temperature independent, and given by

$$\ell = \ln I_D - \ln \left(Aq \frac{D_n}{L_n N_A} + \frac{D_p}{L_p N_D} \right) - 2 \ln \sqrt{B_c B_v}$$

From equations above we can see that at 0 K, we expect the diode voltage to be equal to the bandgap of silicon. Diodes don't work at 0 K though.

Although it's not trivial to see that the diode voltage has a negative temperature coefficient, if you do compute it as in vd.py, then you'll see it decreases.

The slope of the diode voltage can be seen to depend on the area, the current, doping, diffusion constant, diffusion length and the effective masses.

Figure 3 shows the V_D and the deviation of V_D from a straight line. The non-linear component of V_D is only a few mV. If we could combine V_D with a voltage that increased with temperature, then we could get a stable voltage across temperature to within a few mV.

[3] From the Einstein relation $D = \mu kT$ it does appear that the diffusion coefficient increases with temperature, however, the mobility decreases with temperature. I'm unsure of whether the mobility decreases with the same rate though.

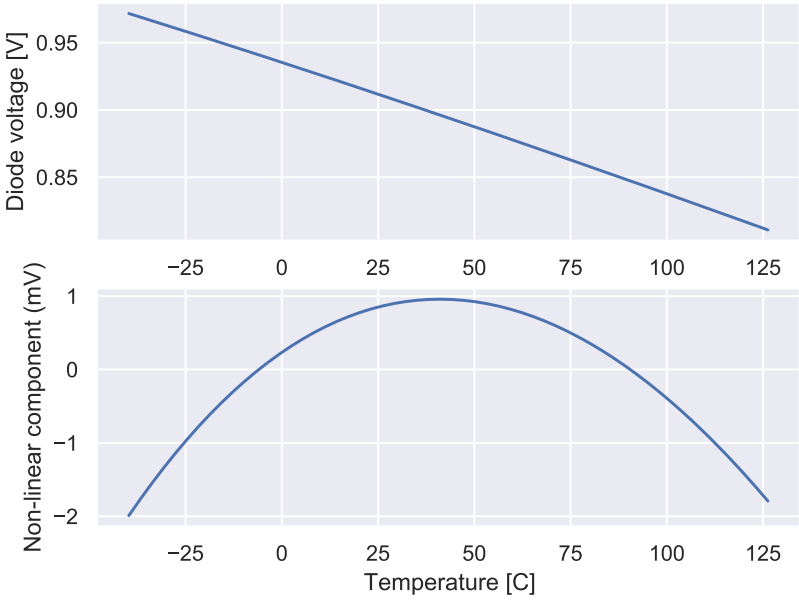


Figure 3: Diode forward voltage as a function of temperature

3.6.4 Current proportional to temperature

Assume we have a circuit like Figure 4.

Here we have two diodes, biased at different current densities. The voltage on the left diode V_{D1} is equal to the sum of the voltage on the right diode V_{D2} and voltage across the resistor R_1. The current in the two diodes are the same due to the current mirror. A such, we have that

$$I_S e^{\frac{qV_{D1}}{kT}} = N I_S e^{\frac{qV_{D2}}{kT}}$$

Taking logarithm of both sides, and rearranging, we see that

$$V_{D1} - V_{D2} = \frac{kT}{q} \ln N$$

Or that the difference between two diode voltages biased at different current densities is proportional to absolute temperature.

In the circuit above, this ΔV_D is across the resistor R_1, as such, the $I_D = \Delta V_D / R_1$. We have a current that is proportional to temperature.

If we copied the current, and sent it into a series combination of a resistor R_2 and a diode, we could scale the R_2 value to give us the exactly right slope to compensate for the negative slope of the V_D voltage.

The voltage across the resistor and diode would be constant over temperature, with the small exception of the non-linear component of V_D.

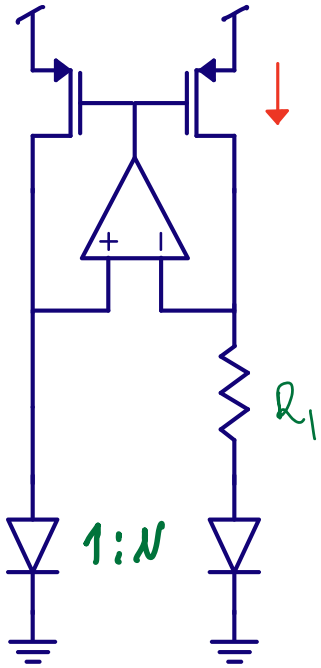


Figure 4: Circuit to generate a current proportional to kT

3.7 Equations aren't real

> Nature does not care about equations. It just is.

We know, at the fundamental level, nature appears to obey the mathematics on quantum mechanics, however, due to the complexity of nature, it's not possible today (which is not the same as impossible), to compute exactly how the current in a diode works. We can get close, by measuring a diode we know well, and hope that the next time we make the same diode, the behavior will be the same.

As such, I want to warn you about the "lies" or "simplifications" we tell you. Take the diode equation above, some parts, like the intrinsic carrier concentration n_i has roots directly from quantum mechanics, with few simplifications, which means it's likely solid truth, at least for a single unit cell.

But there is no reason nature should make all unit cells the same, and infact, we know they are not the same, we put in dopants. As we scale down to a few nano-meter transistors the simplification that "all unit cells of silicon are the same, and extend to infinity" is no longer true, and must be taken into account in how we describe reality.

Other parts, like the exact value of the bandgap E_g, the diffusion constant D_p or diffusion length L_p are macroscopic phenomena, we can't expect them to be 100 true. The values would be based on measurement, but not always exact, and maybe, if you rotate your diode, they would be different.

You should realize that the consequence of our imperfection is that the equations in electronics should always be taken with a grain of salt.

Nature does not care about your equations. Nature will easily have the superposition of trillions of electrons, and they don't have to agree with your equations.

But most of the time, the behavior is similar.

Chapter 4

ESD and IC

4.0.1 What blocks must our IC include?

The project for 2023 is to design an integrated temperature sensor. The hope is that some will tapeout on the Google/Efabless Open MPW shuttle

First, we need to have an idea of what comes in and out of the temperature sensor. Before we have made the temperature sensor, we need to think what the signal interface could be, and we need to learn.

Maybe we read Kofi Makinwa's overview of temperature sensors and find one of the latest papers, A BJT-based CMOS Temperature Sensor with Duty-cycle-modulated Output and ±0.54 °C (3-sigma) Inaccuracy from -40 °C to 125 °C.

At this point, you may struggle to understand the details of the paper, but at least it should be possible to see what comes in and out of the module. What I could find is in the table below, maybe you can find more?

Pin	Function	in/out	Value	Unit
VDD_3V3	analog supply	in	3.0	V
VDD_1V2	digital supply	in	1.2	V
VSS	ground	in	0	V
CLK_1V2	clock	in	20	MHz
RST_1V2	digital	out	0 or 1.2	V
I_C	bias	in	?	uA?
PHI1_1V2	digital	out	0 or 1.2	V
PHI2_1V2	digital	out	0 or 1.2	V
DCM_1V2	digital	out	0 or 1.2	V

This list contains supplies, clocks, digital outputs, bias currents and a ground. Let me explain what they are.

4.0.1.1 Supply

The temperature sensor has two supplies, one analog (3.3 V) and one digital (1.2 V), which must come from somewhere.

We're using Skywater, and to use the free tapeouts we must use the Caravel test chip harness.

That luckily has two supplies. It can be powered externally by up to 5.0 V, and has an external low dropout regulator (LDO) that provides the digital supply (1.8 V). See more at Absolute maximum ratings

4.0.1.2 Ground

Most ICs have a ground, a pin which is considered 0 V. It may have multiple grounds. Remember that a voltage is only defined between two points, so it's actually not true to talk about a voltage in a node (or on a wire). A voltage is always a differential to something. We've (as in global electronics engineers) have just agreed that it's useful to have a "node" or "wire" we consider 0 V.

4.0.1.3 Clocks

Most digital need a clock, and the Caravel provide a 40 MHz clock which should suffice for most things. We could probably just use that clock for our temperature sensor.

4.0.1.4 Digital

We need to read the digital outputs. We could either feed those off chip, or use a on chip micro-controller. The Caravel includes options to do both. We could connect digital outputs to the logic analyzer, and program the RISC-V to store the readings. Or we could connect the digital output to the I/O and use an instrument in the lab.

4.0.1.5 Bias

The Caravel does not provide bias currents (that I found), so that is something you will need to make.

4.0.1.6 Conclusion

Even a temperature sensor needs something else on the IC. We need digital input/output, clock generation (PLL, oscillators), bias current generators, and voltage regulators (which require a constant reference voltage).

I would claim that any System-On-Chip will always need these blocks!

I want you to pause, take a look at the course plan, and now you might understand why I've selected the topics.

4.0.1.7 One more thing

There is one more function we need when we have digital logic and a power supply. We need a "RESET" system.

Digital logic has a fundamental assumption that we can separate between a "1" and a "0", which is usually translated to for example 1.8 V (logic 1) and 0 V (logic 0). But if the power supply is at 0 V, before we connect the battery, then that fundamental assumption breaks.

When we connect the battery, how do we know the fundamental assumption is OK? It's certainly not OK at 30 mV supply. How about 500 mV? or 1.0 V? How would we know?

Most ICs will have a special analog block that can keep the digital logic, bias generators, clock generators, input/output and voltage regulators in a **safe** state until the power supply is high enough (for example 1.62 V).

4.1 Electrostatic Discharge

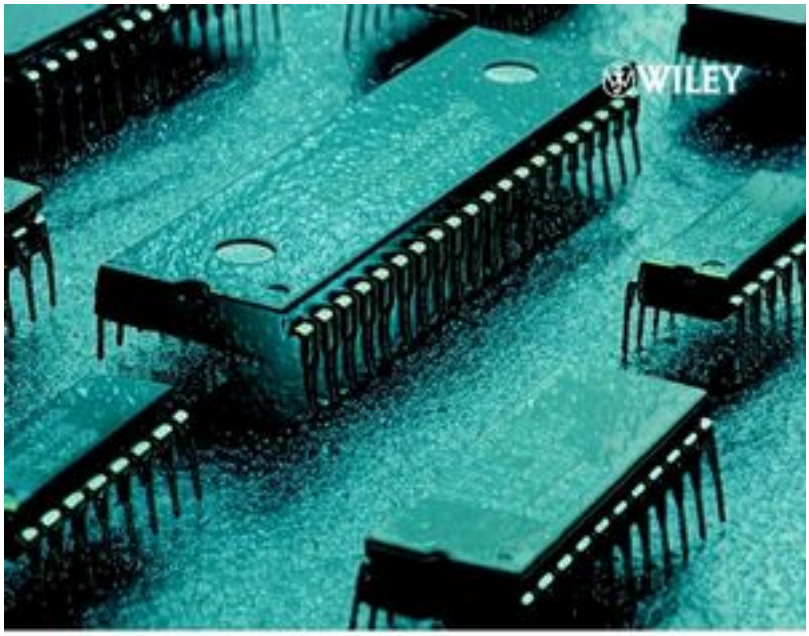
If you make an IC, you must consider Electrostatic Discharge (ESD) Protection circuits

ESD events are tricky. They are short (ns), high current (Amps) and poorly modeled in the SPICE model. Most SPICE models will not model correctly what happens to an transistor during an ESD event. The SPICE models are not made to model what happens during an ESD event, they are made to model how the transistors behave at low fields and lower current.

But ESD design is a must, you have to think about ESD, otherwise your IC will never work.

Consider a certain ESD specification, for example 1 kV human body model, a requirement for an integrated circuit. By requirement I mean if the 1 kV is not met, then the project will be delayed until it is fixed. If it's not fixed, then the project will be infinitely delayed, or in other words, canceled.

Now imagine it's your responsibility to ensure it meets the 1 kV specification, what would you do? I would recommend you read one of the few ESD books in existence, shown below, and rely on you understanding of PN-junctions.



ESD in Silicon Integrated Circuits

Second Edition

AJITH AMERASEKERA | CHARVAKA DUVVURY

The industry has agreed on some common test criteria for electrostatic discharge. Test that model what happens when a person touches your IC, during soldering, and PCB mounting. If your IC passes the test then it's probably going to survive in volume production

Standards for testing at JEDEC

4.1.1 When do ESD events occur?

4.1.2 Before/during PCB

Human body model (HBM)

Models a person touching a device with a finger.

Charged device model (CDM)

An IC left alone for long enough will equalize the Fermi potential across the whole IC.

Not entirely a true statement, but roughly true. One exception is non-volatile memory, like flash, which uses Fowler-Norheim tunneling to charge and discharge a capacitor that keeps it's charge for a very, very long time.

I'm pretty sure that if you leave an SSD hardrive to the heat death of the universe in maybe $10^{10^{10^{56}}}$ years, then the charges will equalize, and the Fermi level will be the same across the whole IC, so it's just a matter of time.

Assume there is an equal number of electrons and protons on the IC. According to Gauss' law

$$\oint_{\partial\Omega} \mathbf{E} \cdot d\mathbf{S} = \frac{1}{\epsilon_0} \iiint_V \rho \cdot dV$$

So there is no external electric field from the IC.

If we place an IC in an electric field, the charges inside will redistribute. Flip the IC on it's back, place it on an metal plate with an insulator in-between, and charge the metal plate to 1 kV.

Inside the IC electrons and holes will redistribute to compensate for the electric field. Closest to the metal plate there will be a negative charge, and furthest away there will be a positive charge.

This comes from the fact that if you leave a metal inside an electric field for long enough the metal will not have any internal field. If there was an internal field, the charges would move. Over time the charges will be located at the ends of the metal.

Take a grounded wire, touch one of the pins on the IC. Since we now have a metal connection between a pin and a low potential the charges inside the IC will redistribute extremely quickly, on the order of a few ns.

During this Charged Device Model event the internal fields in the IC will be chaotic, but at any given point in time, the voltage across sensitive devices must remain below where the device physically breaks.

Take the MOSFET transistor. Between the gate and the source there is an thin oxide, maybe a few nm. If the field strength between gate and source is high enough, then the force felt by the electrons in co-valent bonds will be $\vec{F} = q\vec{E}$. At some point the co-valent bonds might break, and the oxide could be permanently damaged. Think of a lighting bolt through the oxide, it's a similar process.

Our job, as electronics engineers, is to ensure we put in additional circuits to prevent the fields during a CDM event from causing damage.

For example, let's say I have two inverters powered by different supply, VDD1 and VDD2. If I in my ESD test ground VDD1, and not VDD2, I will quickly bring VDD1 to zero, while VDD2 might react slower, and stay closer to 1 kV. The gate source of the PMOS in the second inverter will see approximately 1 kV across the oxide, and will break. How could I prevent that?

Assuming some luck, then VDD1 and VDD2 are separate, but the same voltage, or at least close enough, I can take two diodes, connected in opposite directions, between VDD1 and VDD2. As such, when VDD1 is grounded, VDD2 will follow but maybe be 0.6 V higher. As a result, the PMOS gate never sees more than approximately 0.6 V across the gate oxide, and everyone is happy.

Now imagine an IC will hundreds of supplies, and billions of inverters. How can I make sure that everything is OK?

CDM is tricky, because there are so many details, and it's easy to miss one that makes your circuit break.

4.1.3 After PCB

Human body model (HBM)

System level ESD

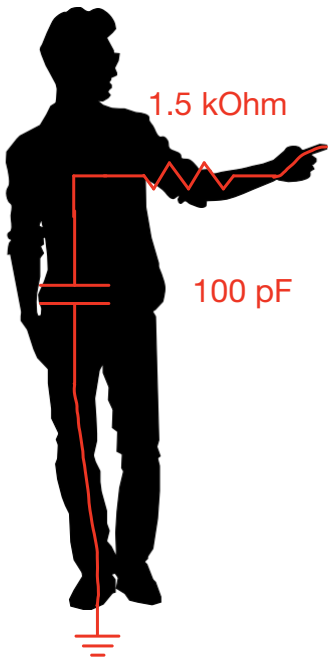
Once mounted on the PCB, the ICs can be more protected against ESD events, however, it depends on the PCB, and how that reacts to a current.

Take a look at your USB-A connector, you will notice that the outer pins, the power and ground, are made such that they connect first, The $D+$ and $D-$ pins are a bit shorter, so they connect some μs later. The reason is ESD. The power and ground usually have a low impedance connection in decoupling capacitors and power circuits, so those can handle a large ESD zap. The signals can go directly to an IC, and thus be more sensitive.

We won't go into details on System level ESD, as that is more a PCB type of concern. The physics are the same, but the details are different.

4.1.4 Human body model (HBM)

- Models a person touching a device with a finger
- **Long** duration (around 100 ns)
- Acts like a current source into a pin
- Can usually be handled in the I/O ring
- 4 kV HBM ESD is 2.67 A peak current

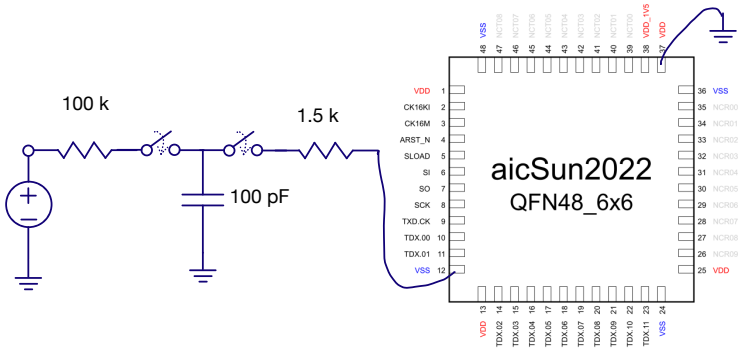


4.2 An ESD zap example

Imagine a ESD zap between VSS and VDD. How can we protect the device?

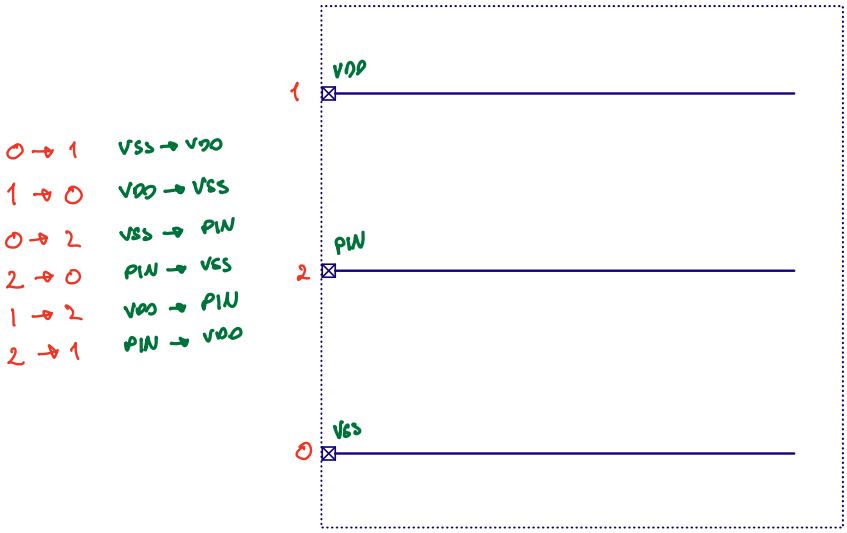
The positive current enters the VSS, and leaves via the VDD, so our supplies are flipped up-side down. It's a fair assumption that none of the circuits inside will work as intended.

But the IC must not die, so we have to lead the current to ground somehow



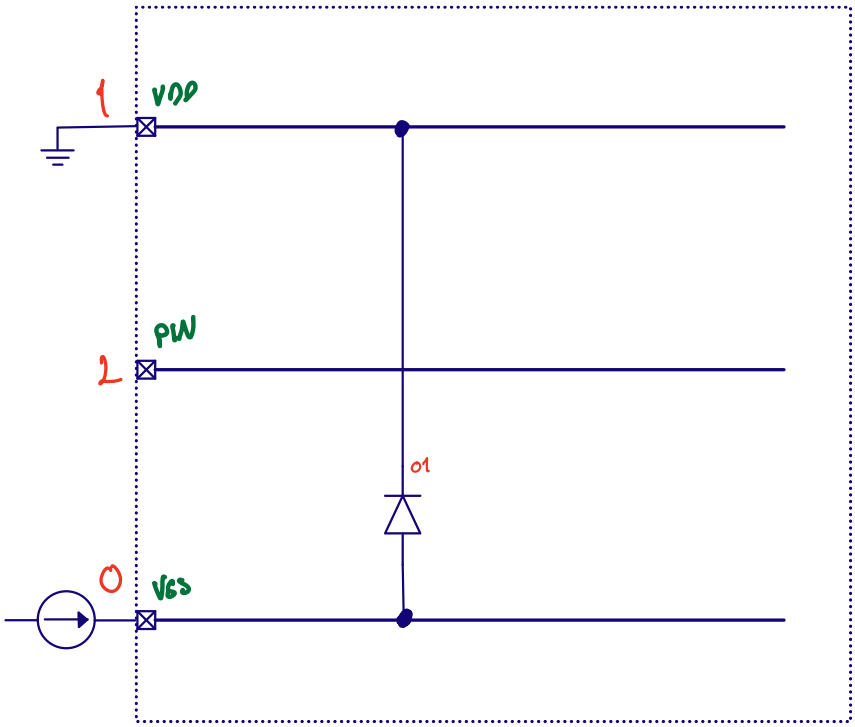
4.3 Permutations

Let's simplify and think of the possible permutations, shown in the figure below. We don't know where the current will enter nor where it will leave our circuit, so we must make sure that all combinations are covered.

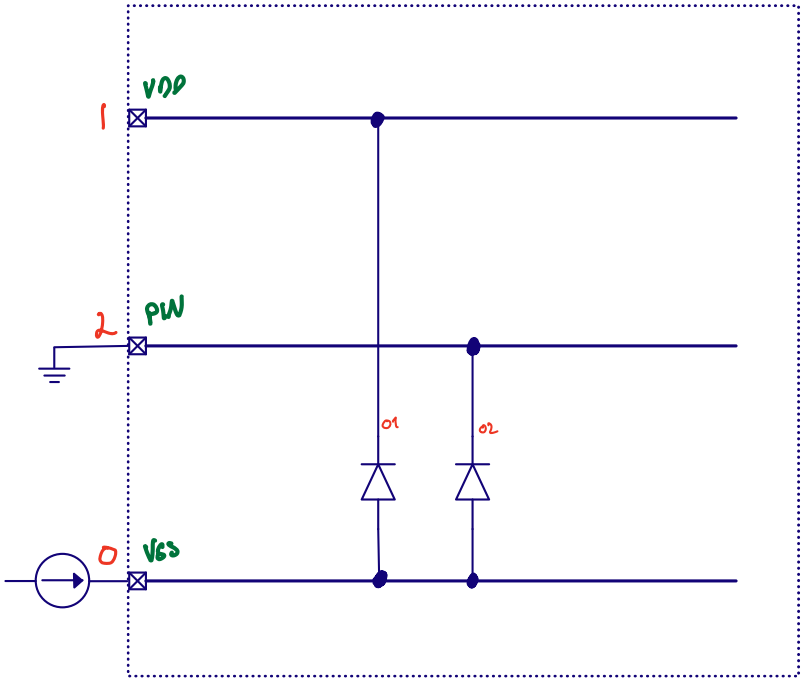


When the current enters VSS and must leave via VDD, then it's simple, we can use a diode.

Under normal operation the diode will be reverse biased, and although it will add some leakage, it will not affect the normal operation of our IC.



The same is true for current in on VSS and out on PIN. Here we can also use a diode.



For a current in on VDD and out on VSS we have a challenge. That's the normal way for current to flow.

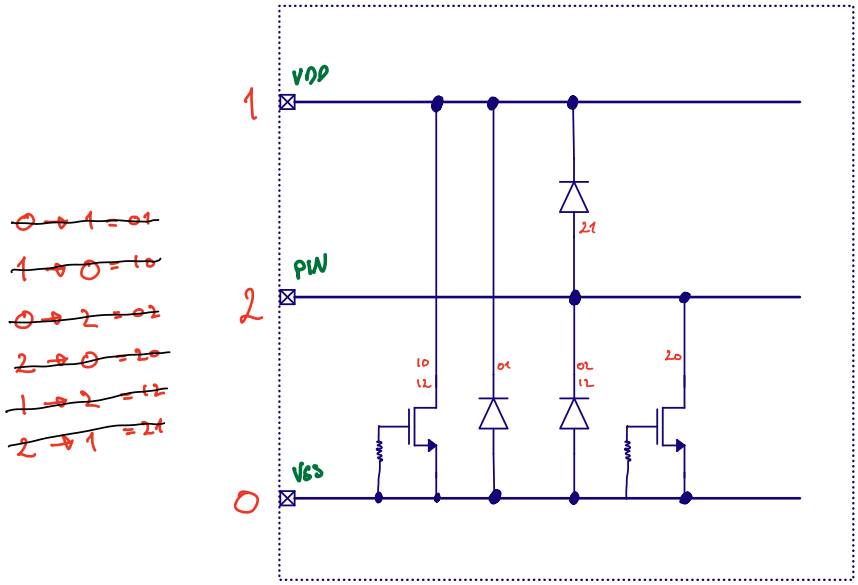
For those from Norway that have played a kids game Bjørnen sover, that's a apt mental image. We want a circuit that most of the time sleeps, and does not affect our normal IC operation. But if a huge current comes in on VDD, and the VDD voltage shoots up fast, the circuit must wake up and bring the voltage

down.

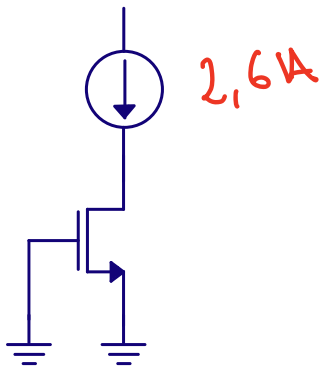
If the circuit triggers under normal operating condition, when your watching a video on your phone, your battery will drain very fast, and it might even catch fire.

As such, ESD design engineers have a “ESD design window”. Never let the ESD circuit trigger when VDD < normal, but always trigger the ESD circuit before VDD > breakdown of circuit.

A circuit that can sometimes be used, if the ESD design window is not too small, is the Grounded-Gate-NMOS in the figure below.



4.3.1 Why does this work?



If you try the circuit above in with the normal BSIM spice model, it will not work. The transistor model does not include that part of the physics.

We need to think about how electrons, holes PN-junctions and bipolars work.

4.3.1.1 Quick refresh of solid-state physics

Electrons sticking to atoms (bound electrons), can only exist at discrete energy levels. As we bring atoms closer to each-other the discrete energy levels will split, as computed from Schrodinger, into bands of allowed energy states. These bands of energy can have lower energy than the discrete energy levels of the atom. That's why some atoms stick together and form molecules through co-valent bonds, ionic bonds, or whatever the chemists like to call it. It's all the same thing, it's lower energy states that make the electrons happy, some are strong, some are weak.

For silicon the energy band structure is tricky to compute, so we simplify to band diagrams that only show the lowest energy conduction band and highest energy valence band.

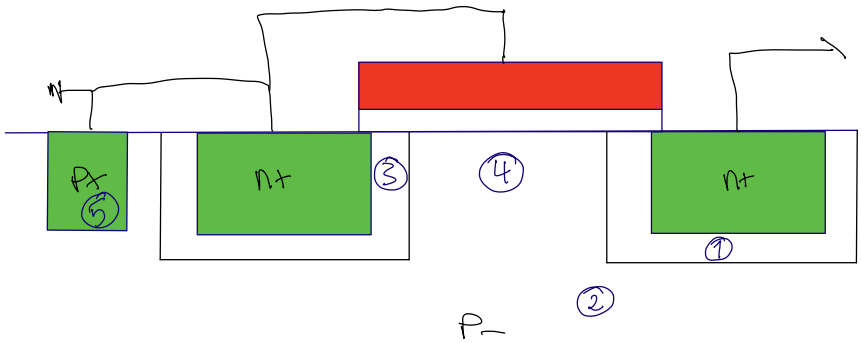
Electrons can move freely in the conduction band (until they hit something, or scatter), and electrons moving in the valence band act like positive particles, nicknamed holes.

How many free charges there are in a band is given by Fermi-Dirac distribution and the density of states (allowed energy levels).

If an electron, or a hole have sufficient energy (accelerated by a field), they can free an electron/hole pair when they scatter off an atom. If you break too many bonds between atoms, your material will be damaged.

4.3.1.2 The grounded-gate NMOS

Assume a transistor like the one below. The gate, source and bulk is connected to ground. The drain is connected to a high voltage.



4.3.1.3 Avalanche

The first thing that can happen is that the field in the depletion zone between drain and bulk (1) is large, due to the high voltage on drain, and the thin depletion region.

In the substrate (P-) there are mostly holes, but there are also electrons. If an electron diffuses close to the drain region it will be swept across to drain by the high field.

The high field might accelerate the electron to such an energy that it can, when it scatters of the atoms in the depletion zone, knock out an electron/hole pair.

The hole will go to the substrate (2), while the new electron will continue towards drain. The new electron can also knock out a new electron/hole pair (energy level is set by impact ionization of the atom), so can the old one assuming it accelerates enough.

One electron turn into two, two to four, four to eight and so on. The number of electrons can quickly become large, and we have an avalanche condition. Same as a snow avalanche, where everything was quiet and nice, now suddenly, there is a big trouble.

Usually the avalanche process does not damage anything, at least initially, but it does increase the hole concentration in the bulk. The number of holes in the bulk will be the same as the number of electrons freed in the depletion region.

4.3.1.4 Forward bias of PN-junction

The extra holes underneath the transistor will increase the local potential. If the substrate contact (5) is far away, then the local potential close to the source/bulk PN-junction (3) might increase enough to significantly increase the number of electrons injected from source.

Some of the electrons will find a hole, and settle down, while others will diffuse around. If some of the electrons gets close to the drain region, and the field in the depletion zone, they will be accelerated by the drain/bulk field, and can further increase the avalanche condition.

4.3.1.5 Bad things can happen

For a normal transistor, not designed to survive, the electron flow (4) can cause local damage to the drain. Normally there is nothing that prevents the current from increasing, and the transistor will eventually die.

If we add a resistor to the drain region (unscilicided drain), however, we will slow down the electron flow, and we can get a stable condition, and design a transistor that survives.

4.3.1.6 What have we done

Turns out, that every single NMOS has a sleeping bear. A parasitic bipolar. That's exactly what this GGNMOS is, a bipolar transistor, although a pretty bad one, that is designed to trigger when avalanche condition sets in and is designed to survive.

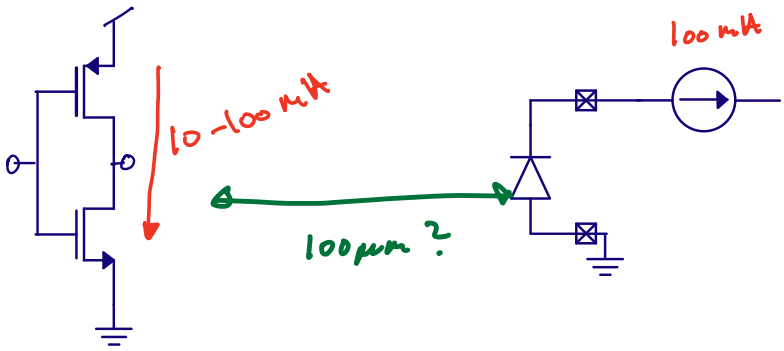
A normal NMOS, however, can also trigger, and if you have not thought about limiting the electron current, it can die, with IC killing consequences. Specifically, the drain and source will be shorted by likely the silicide on top of the drain, and instead of a transistor with high output impedance, we'll have a drain source connection with a few kOhm output impedance.

Take a look at New Ballasting Layout Schemes to Improve ESD Robustness of I/O Buffers in Fully Silicided CMOS Process for the pretty pictures you'll get when the drain/source breaks.

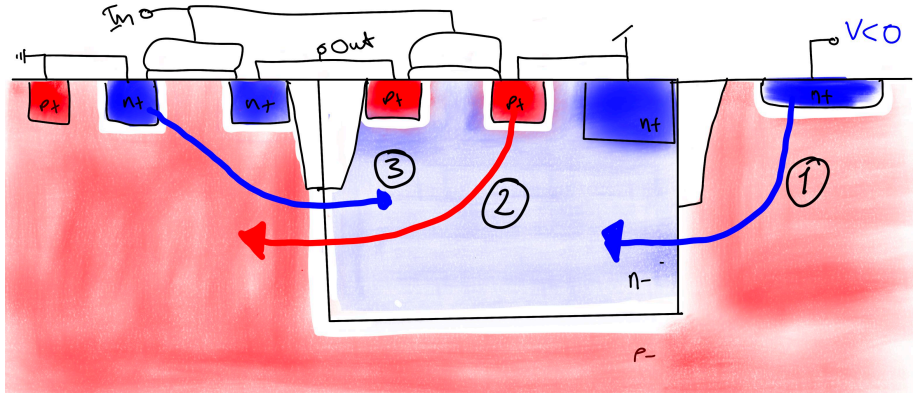
4.3.2 How can current in one place lead to a current somewhere else?

Another fun physics problem can happen in digital logic that is close to an electron source, like a connection to the real world, what we call a pad. A pad is where you connect the bond-wire in a QFN type of package with wire-bonding

Assume we have the circuit below.



We can draw a cross section of the inverter.



4.3.2.1 Electron injection

Assume that we have an electron source, for example a pad that is below ground for a bit. This will inject electrons into the substrate/bulk (1) and electrons will diffuse around.

If some of the electrons comes close to the N-well depletion region (2) they will be swept across by the built-in field. As a result, the potential of the N-well will decrease, and we can forward bias the source or drain junction of a PMOS.

4.3.2.2 Forward biased PMOS source or drain junction

With a forward biased source/bulk junction (2), holes will be injected into the N-Well, but similarly to the GGNMOS, they might not find a electron immediately.

Some of the holes can reach the depletion region towards our NMOS, and be swept across the junction.

4.3.2.3 Forward biased NMOS source or drain junction

The increase in hole concentration underneath the NMOS can forward bias the PN diode between source (or drain) and bulk. If this happens, then we get electron injection into bulk. Some of those electrons can reach the N-well depletion region, and be swept across (3).

4.3.2.4 Positive-feedback

Now we have a condition where the process accellerates, and locks-up. Once turned on, this circuit will not turn off until the supply is low.

This is a phenomena called latch-up. Similar to ESD circuits, latch-up can short the supply to ground, and make things burn.

That is why, when we have digital logic, we need to be extra careful close to the connection to the real world. Latch-up is bad.

We can prevent latch-up if we ensure that the electrons that start the process never reach the N-wells. We can also prevent latch-up by separating the NMOS and PMOS by guard rings (connections to ground, or indeed supply), to serve as places where all these electrons and holes can go.

Maybe it seems like a rare event for latch-up to happen, but trust me, it's real, and it can happen in the strangest places. Similar to ESD, it's a problem that can kill an IC, and make us pay another X million dollars for a new tapeout, in addition to the layout work needed to fix it.

Latch-up is why you will find the design rule check complaining if you don't have enough substrate connections to ground, or N-well connections to power close to your transistors.

Similar to the GGNMOS, this circuit, a thyristor can be a useful circuit in ESD design. If we can trigger the thyristor when the VDD shoots to high, then we can create a good ESD protection circuit.

See low-leakage ESD for a few examples.

You must **always handle ESD** on an IC

- Do everything yourself
- Use libraries from foundry
- Get help www.sofics.com

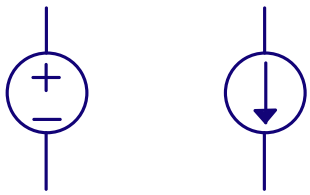
Chapter 5

References and bias

In our testbenches, and trial schematics, it's common to include voltage sources and current sources. However, the ideal voltage source, or ideal current source does not exist in the real world. There is no such thing.

We can come close to creating a voltage source, so a known voltage, with a low source impedance, but not zero. And it won't be inifinetly fast either. If we suddenly decide to pull 1 kA from a lab supply I promise you the voltage will drop.

So how do we create something that is a *good enough* voltage and current source on an IC?



5.1 Routing

Before we take a take a look at the voltage and current source, I want you to think about how you would route a current, or a voltage on an IC.

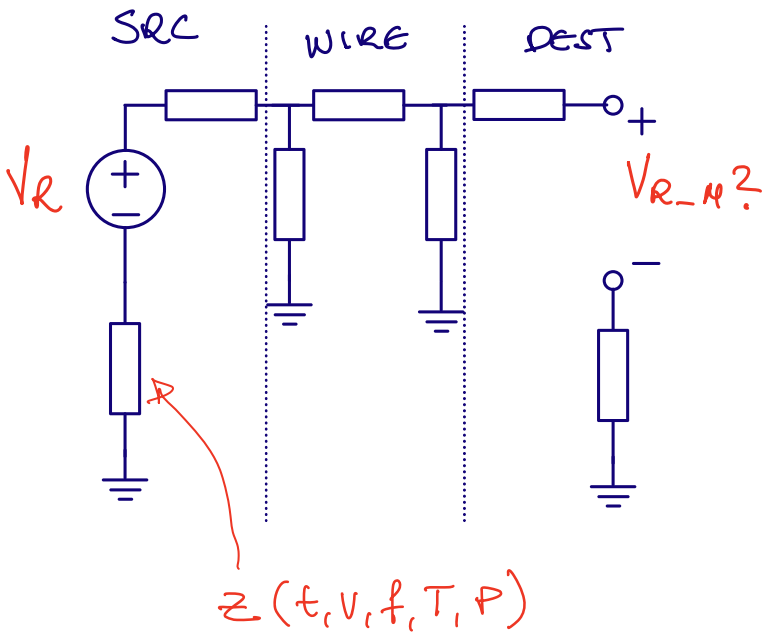
Assume we do have a known voltage on our IC. How can we make sure we can share that voltage across an IC?

A voltage is only defined between two points. There is no such thing as the *voltage at a point on a wire*, nor *voltage in a node*. Yes, I know we say that, but it's not right. What we forget is that by *voltage in a node* we always, always mean *voltage in a node referred to ground*.

We've invented this magical place called *ground*, the final resting place of all electrons, and we have agreed that all voltages refer to that point.

So when we say "Voltage in node A is 1V", what we actually mean is "Voltage in node A is 1 V referred to ground".

Now you understand why we can't just route a known voltage across the IC, the other side might not have the same ground. The *other* side might have a different impedance to ground, and the impedance might be a function of time, voltage, frequency temperature, pressure and presence of gremlins.



Most of the time, in order not to think about the ground impedance, we choose to route a known quantity as a current instead of a voltage. That means, however , we must convert from a voltage to a current, but we can do that with a resistor (you'll see later), and as long as the resistor is the same on the other side of the IC, then we'll know what the voltage is.

Resistors have finite matching across die, let's say 2 % 3-sigma variation. As a result, if we need a really accurate voltage reference, then we must distribute voltage.

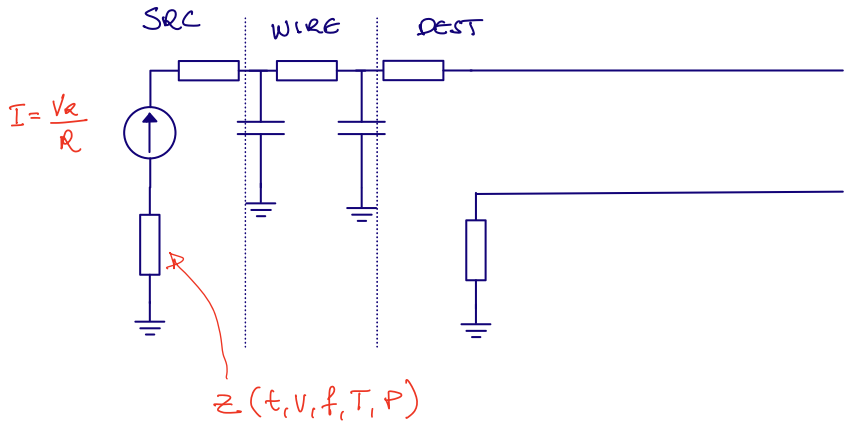
But how can "It's better to distribute a voltage as a current across the IC, it's more accurate" and "If you need something really accurate, you must distribiute voltage" both be true?

Imagine I have a 0.5 % 3-sigma accurate voltage reference at 1.22 V, that's a sigma of 2 mV. I need this reference voltage on a block on the other side of the IC, I don't want to distribute voltage, because I don't know that the ground is the same on the other side, at least not to a precision of 2 mV. I convert the voltage into a current, however, I know the R has a 2 % 3-sigma across die, so my error budget immediately increases to 2.06%. In addition, each current mirror will add more errors due to the mismatch in current.

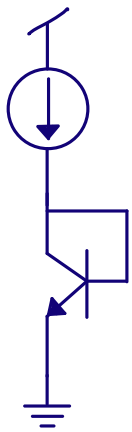
But what if I must have 0.5 % 3-sigma voltage in the block? For example in a battery charger, where the 4.3 V termination voltage must be 1 % accurate? I have no choice but to go with voltage directly from the reference, but the key point, is then the receiving block **cannot** be on the other side of the IC. Maybe I then need two references on my IC.

Maybe another question that could be asked would be "Why do we care if there is two references?" And

the answer is “Silicon area is extremely expensive, to make things cheep, we must make things small”, in other words, we should not duplicate features unless we absolutely have to.



5.2 Bandgap voltage reference



5.2.1 How does VBE change with temperature?

A diode connected bipolar transitor, or indeed a PN diode, assuming a fixed current, will have a voltage across that is temperature dependent

$$I_D = I_S \left(e^{\frac{V_{BE}}{V_T}} - 1 \right) + I_B$$

As I_S is much smaller that I_D we can ignore the -1, and we assume that the base current is much smaller than the drain current, which gives us

$$I_D \approx I_S e^{\frac{V_{BE}}{V_T}}$$

Re-arranging for V_{BE} and inserting for

$$V_T = \frac{kT}{q}$$

$$V_{BE} = \frac{kT}{q} \ln \frac{I_D}{I_S}$$

From this equation, it looks like the voltage V_{BE} is proportional to temperature

However, it turns out that the V_{BE} decreses with temperature due to the temperature dependence of I_S . If you want to understand why, take a look at Diodes

The V_{BE} is linear with temperature with a property that if you extraplotate the V_{BE} line to zero Kelvin, then all diode voltages seem to meet at the bandgap voltage of silicon (approx 1.12 V).

5.2.2 The current is proportional to temperature (PTAT), why?

If we take two diodes, or bipolars, biased at different current densities, as shown in the figure below, then

$$V_{D1} = V_T \ln \frac{I_{D1}}{I_S}$$

$$V_{D2} = V_T \ln \frac{I_{D2}}{I_S}$$

where $I_{D1} = NI_{D2}$.

The OTA will force the voltage on top of the resistor to be equal to V_{D1}, thus the voltage accross the resistor R_1 is

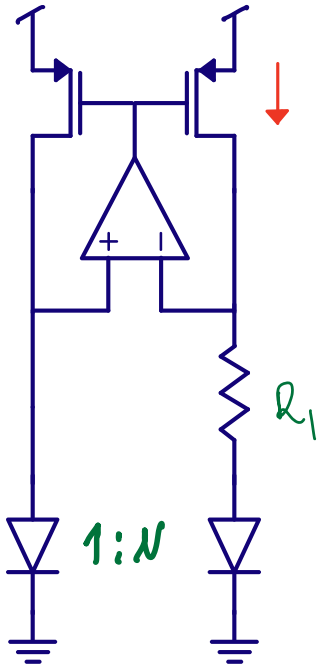
$$V_{D1} - V_{D2} = V_T \ln \frac{I_{D1}}{I_S} - V_T \ln \frac{I_{D2}}{I_S} = V_T \ln \frac{I_{D1}}{I_{D2}} = V_T \ln N$$

This is a remarkable result. The difference between two voltages is only defined by boltzmann's constant, temperature, charge, and a know current relationship.

This differential voltage can be used to read out directly the temperature on an IC, provided we have a known voltage to compare with.

We often call this voltage ΔV_D or ΔV_{BE}, and we can clearly see it's proportional to absolute temperature.

We know that the V_D decreases linearly with temperature, so if we combined a multiplum of the ΔV_{BE} with a V_D voltage, then we should get a constant voltage.



5.2.3 How can we combine a CTAT voltage with a PTAT current to get a constant voltage?

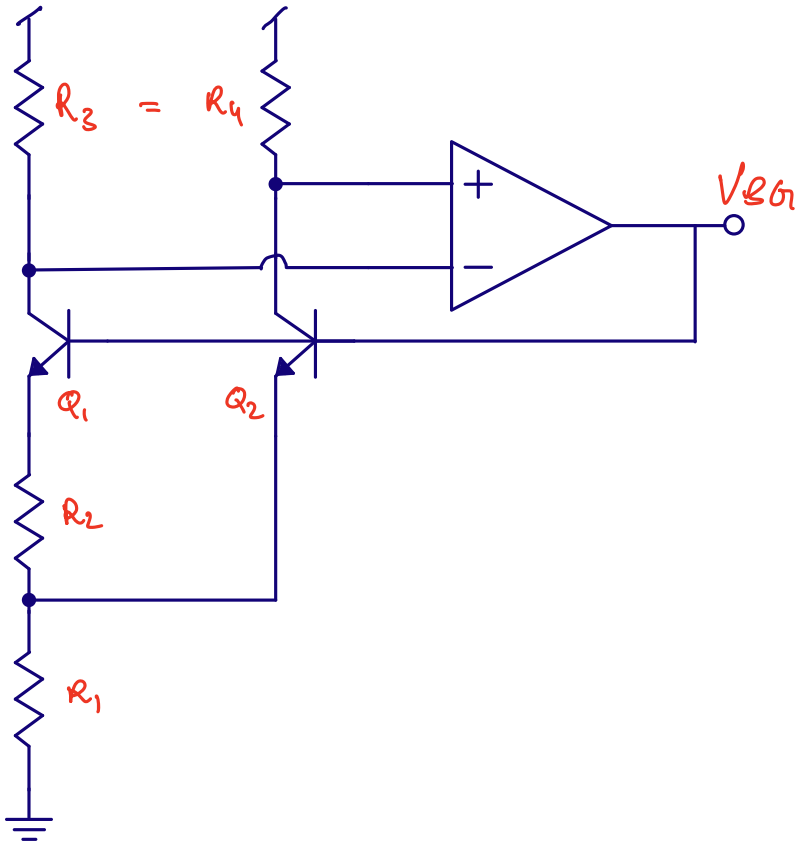
The Brokaw reference below does exactly that. The opamp ensures the two bipolars have the same current. Q_1 is larger than Q_2. The ΔV_{BE} is across the R_2, so we know the current I. We know that R_1 must then have $2I$. The voltage at the output will then be.

$$V_{BG} = V_{G0} + (m-1)\frac{kT}{q}\ln\frac{T_0}{T} + T\left[\frac{k}{q}\ln\frac{J_2}{J_1}\frac{2R2}{R1} - \frac{V_{G0}-V_{be0}}{T_0}\right]$$

where V_{G0} is the bandgap, V_{be0} is the base emitter measured at a temperature T_0 and the J's are the current densities.

To get a constant output voltage, the relationship between the resistors should be approximately

$$\frac{R2}{R1} = \frac{V_{G0}-V_{be0}}{2T_0\frac{k}{q}\ln(\frac{J_2}{J_1})}$$

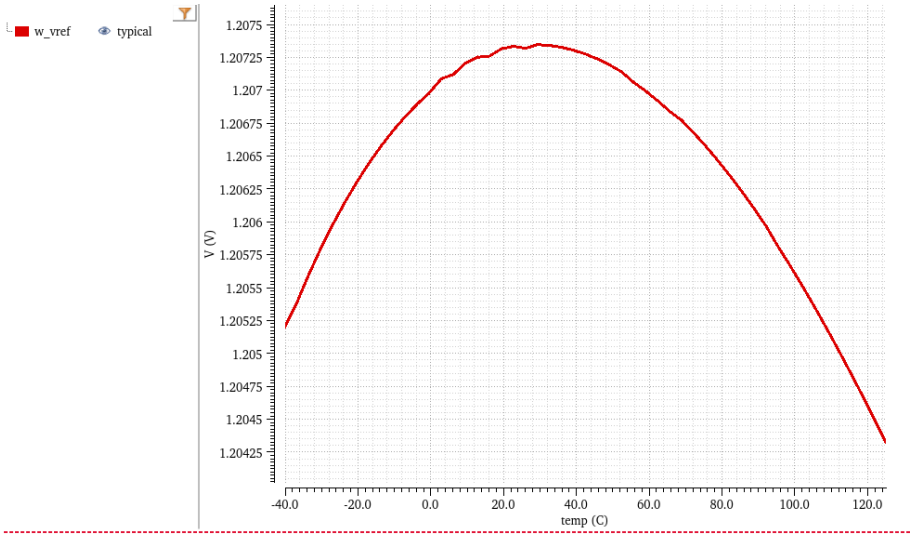


In typical simulations, the variation can be extremely low over the temperature range. The second order error is the remaining error from

$$V_{BG} = V_{G0} + (m-1)\frac{kT}{q}\ln\frac{T_0}{T} + T\left[\frac{k}{q}\ln\frac{J_2}{J_1}\frac{2R2}{R1} - \frac{V_{G0} - V_{be0}}{T_0}\right]$$

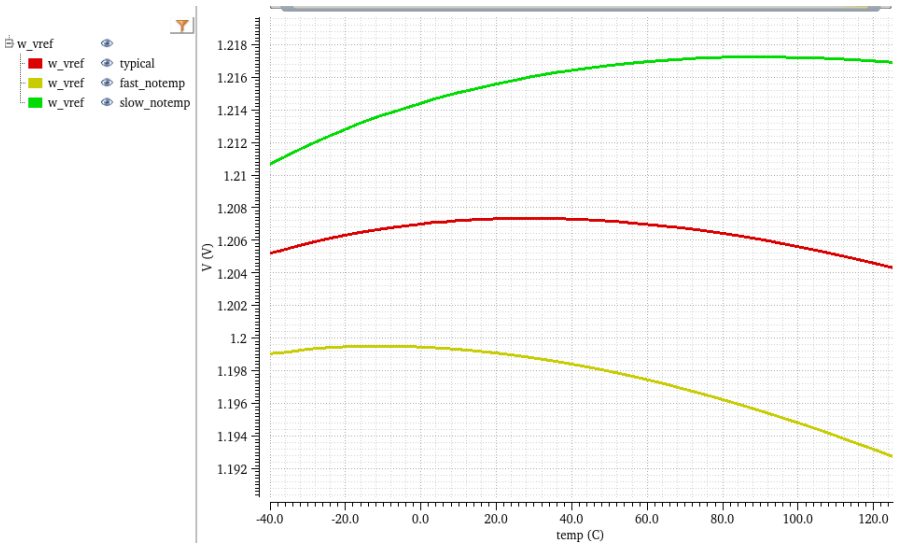
Where the last term is zero, so

$$V_{BG} = V_{G0} + (m-1)\frac{kT}{q}\ln\frac{T_0}{T}$$



Over corners, I do expect that there is variation. It may be that the V_D modeling is not perfect, which means the cancellation of the last term is incomplete.

We could include trimming of PTAT to calibrate for the remaining error, however, if we wanted to remove the linear gradient, we would need a two point temperature test of every IC, which too expensive for low-cost devices.

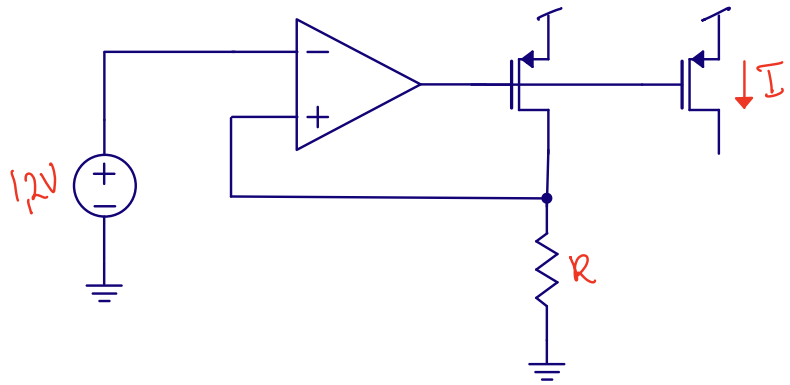


5.2.4 How does a VI converter circuit work?

With a known voltage, we can convert to a known current with the circuit below. On-chip we don't have accurate resistors, but for bias currents, it's usually ok with $+-20$ variation (the variation of R).

Across a IC, we can expect to match the R within a few percent, so we can recreate a voltage with a accuracy of a few percent difference from the original.

If we wanted to create an accurate current, then we'd trim the R until the current is what we want.



5.3 Low voltage bandgap

The Brokaw reference, and others, have a 1.2 V output voltage, which is hard if your supply is below about 1.4 V. As such, people have investigated making lower voltage references.

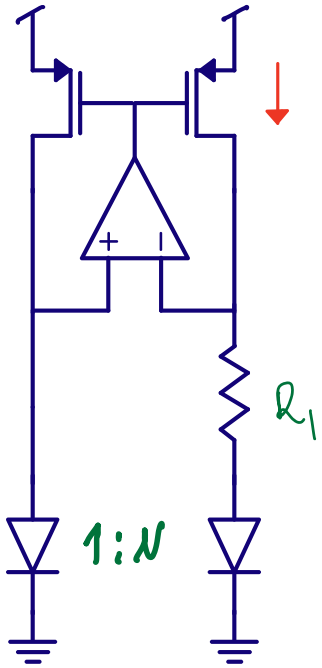
In real ICs though, you should ask yourself long and hard whether you really need these low-voltage references. Most ICs today still have a high voltage, either 1.8 V or 3.0 V.

If you do need them though, consider the circuit below. We have two diodes at different current densities. The ΔV_D will be across R_1. The voltage at the input of the OTA will be V_D and the OTA will ensure the both are equal.

The current will then be

$$I_1 = \frac{\Delta V_D}{R_1}$$

and we know the current increases with temperature, since ΔV_D increases with temperature.



5.3.1 What is the current in R1 and R2?

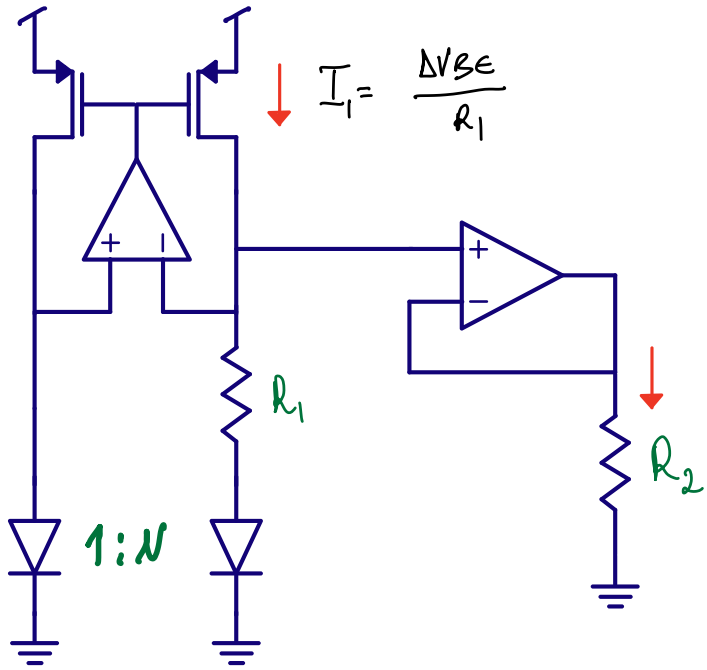
In the figure below I've used ΔV_{BE}, it's the same as ΔV_D, so ignore that error.

Assume we copy the V_D to another node, and place it across a second resistor R_2, as shown in the figure below. The current in this second resistor is then

$$I_2 = \frac{V_D}{R_2}$$

and we know the current decreases with temperature, since V_D decreases with temperature.

From before, we know the current in R_1 is proportional to temperature. As such, if we combine the two with the correct proportions, then we can get a current that does not change with temperature.

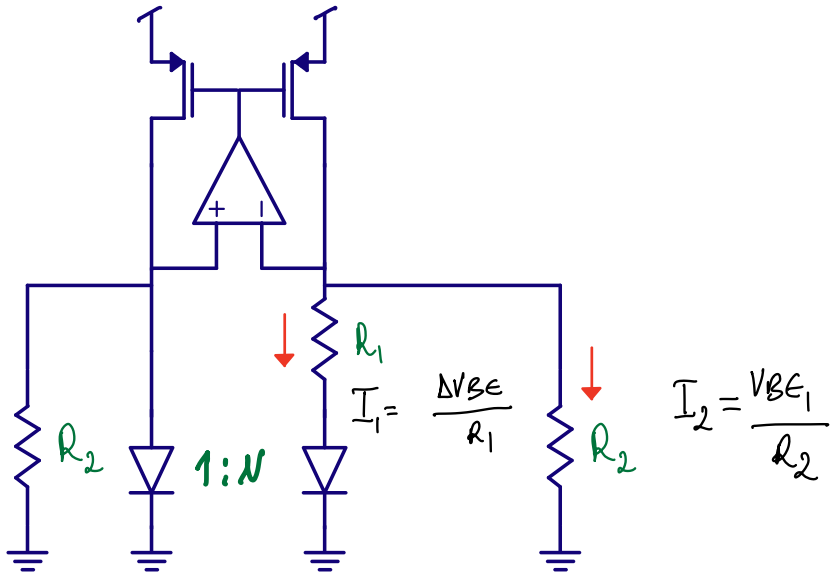


5.3.2 Is the current through R2 the same without the OTA?

Let's remove the OTA, and connect R_2 directly to V_D nodes, you should convince yourself of the fact that this does not change I_1 at all. It does, however, change the current in the PMOS.

Provided we scale R_2 correctly, then the positive I_1 can be compensated by the negative I_2, and we have a current that is independent of temperature.

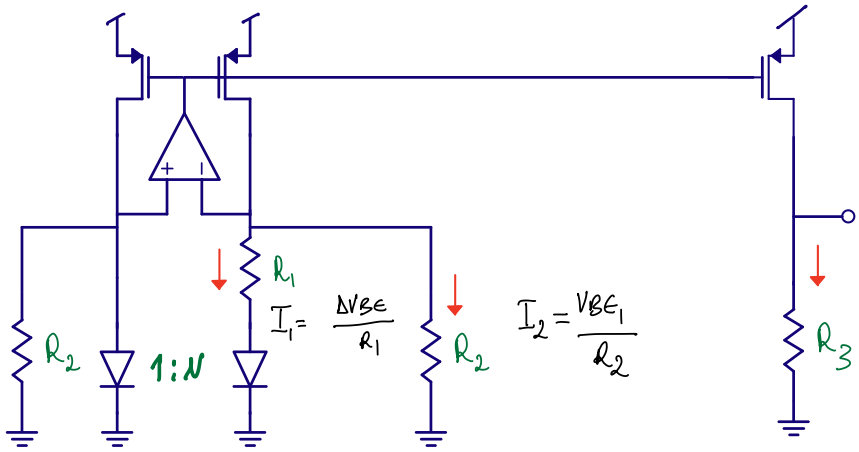
$$I_{PMOS} = \frac{V_D}{R_2} + \frac{\Delta V_D}{R_1}$$



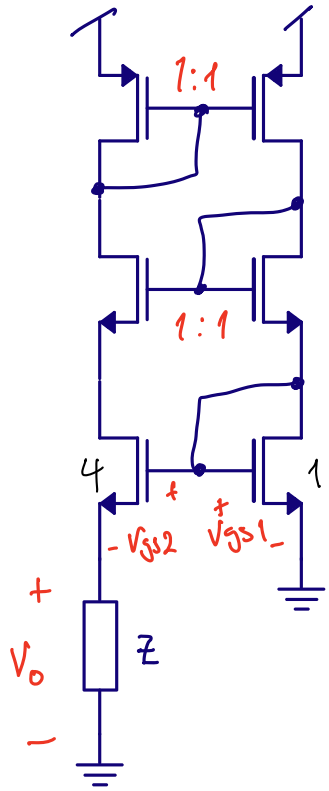
Assuming we copy the current into another resistor R_3, as shown below, we can get a voltage that is

$$V_{OUT} = R_3 \left[\frac{V_D}{R_2} + \frac{\Delta V_D}{R_1} \right]$$

Where the output voltage can be choosen freely, and indeed be lower than 1.2 V.



5.3.3 Why is 1/Z proportional to transistor transconductance?



Sometimes we don't need a full bandgap reference. In those cases, we can use a GM cell, where the impedance could be a resistor, in which case

$$V_o = V_{GS1} - V_{GS2} = V_{eff1} + V_{tn} - V_{eff2} - V_{tn} = V_{eff1} - V_{eff2}$$

Assuming strong inversion, then

$$I_{D1} = \frac{1}{2}\mu_n C_{ox}\frac{W_1}{L_1}V_{eff1}^2$$

$$I_{D2} = \frac{1}{2}\mu_n C_{ox}4\frac{W_1}{L_1}V_{eff2}^2$$

$$I_{D1} = I_{D2}$$

$$\frac{1}{2}\mu_n C_{ox}\frac{W_1}{L_1}V_{eff1}^2 = \frac{1}{2}\mu_n C_{ox}4\frac{W_1}{L_1}V_{eff2}^2$$

$$V_{eff1} = 2V_{eff2}$$

Inserted into above

$$V_o = V_{eff1} - \frac{1}{2}V_{eff1} = \frac{1}{2}V_{eff1}$$

Still assuming strong inversion, such that

$$g_m = \frac{2I_d}{V_{eff}}$$

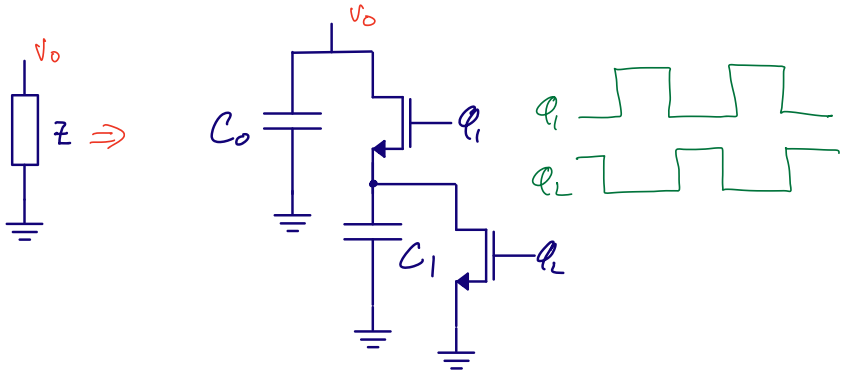
we find that

$$I = \frac{V_{eff1}}{2Z}$$

$$Z \Rightarrow \frac{1}{g_m}$$

If we use a resistor for Z, then we can get a transconductance that is proportional to a resistor, or a constant g_m bias.

We can use other things for Z, like a switched capacitor



Chapter 6

Analog frontend and filter

6.1 Introduction

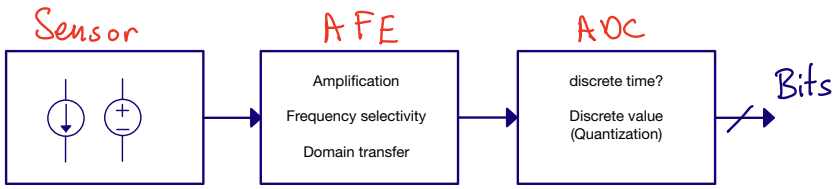
The world is analog, and these days, we do most signal processing in the digital domain. With digital signal processing we can reuse the work of others, buy finished IPs, and probably do processing at lower cost than for analog.

Analog signals, however, might not be suitable for conversion to digital.

A sensor might have a high, or low impedance, and have the signal in the voltage, current, charge or other domain.

To translate the sensor signal into something that can be converted to digital we use analog front-ends (AFE). How the AFE looks will depend on application, but it's common to have amplification, frequency selectivity or domain transfer, for example current to voltage.

An ADC will have a sample rate, and will alias (or fold) any signal above half the sample rate, as such, we also must include a anti-alias filter in AFE that reduces any signal outside the bandwidth of the ADC as close to zero as we need.



One example of an analog frontend is the recieve chain of a typical radio, shown below. The signal that arrives at the antenna, or the "sensor", can be weak, maybe -90 dBm, which is a power of $10^{-90/10} \times 1$ mW $= 1$ pW or 50 μV RMS in 50 Ohm.

At the same time, at another frequency, there could be a signal of 0 dBm, or 0 mW, or 50 mV in 50 Ohm.

As such, there is a 1000 times difference between the wanted signal and the unwanted signal. Assume for the moment we actually used an ADC at the antenna, how many bits would we need?

Bluetooth uses Gaussian Frequency Shift Keying, which is a constant envelope binary modulation, and it's ususally sufficient with low number of bits, assume 8-bits for the signal is more than enough.

But with the unwanted signal being 1000 times bigger, we would need an additional 10-bits, in total 18-bits. If we were to sample at 5 GHz, to ensure the bandwidth is sufficient for a 2.480 GHz maximum frequency we can actually compute the power consumption.

Given the Walden figure of merit of $FOM = \frac{P}{2^{ENOB} fs}$. The best FOM in literature is about 1 fJ/step, so $P = 1$ fJ/step $\times 2^{18} \times 5$GHz $= 1.31$ W, which is not an impossible number, but it's certainly not what Bluetooth ICs do, because they consume around 20 mW in recieve mode.

The radio below has multiple blocks in the AFE. First is low-noise amplifier (LNA) amplifying the signal by maybe 10 times. The LNA reduces the noise impact of the following blocks. The next is the complex mixer stage, which shifts the input signal from radio frequency down to a low frequency, but higher than the bandwidth of the wanted signal. Then there is a complex anti-alias filter, also called a poly-phase filter, which rejects parts of the unwanted signals. Lastly there is a complex ADC to convert to digital.

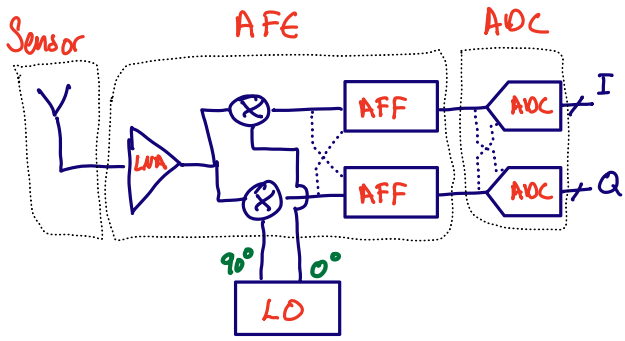
In digital we can further filter to select exactly the wanted signal. Digital filters can have high stop band attenuation at a low power and cost. There could also be digital mixers to further reduce the frequency.

The AFE makes the system more efficient. In the 5 GHz ADC output, from the example above, there's lot's of information that we don't use.

An AFE can reduce the system power consumption by constraining digital information processing and conversion to the parts of the spectrum with information of interest.

There are instances, though, where the full 2.5 GHz bandwidth has useful information. Imagine in a cellular base station that should process multiple cell-phones at the same time. In that case, it could make sense with an ADC at the antenna.

What make sense depends on the application.



6.2 Filters

A filter can be fully described by the transfer function, usually denoted by $H(s) = \frac{\text{output}}{\text{input}}$.

Most people will today start design with a high-level simulation, in for example Matlab, or Python. Once they know roughly the transfer function, they will implement the actual analog circuit.

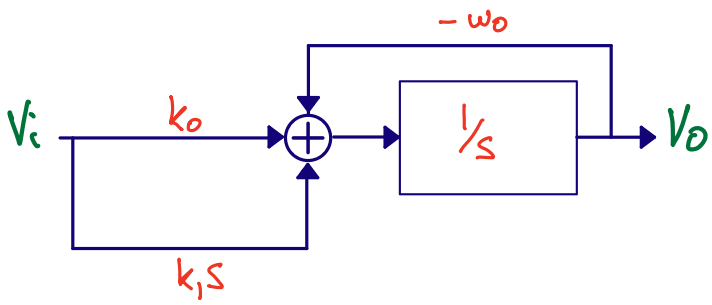
For us, as analog designers, the question becomes "given an $H(s)$, how do we make an analog circuit?"" It can be shown that a combination of 1'st and 2'nd order stages can synthesize any order filter. To break down $H(s)$ into first and second order stages we could use symbolic tools like Maple, and maybe it's even possible in Python these days.

Once we have the first and second order stages, we can start looking into circuits.

6.3 First order filter

In the book they use signal flow graphs to show how the first order stage can be generated. By selecting the coefficients k_0 ,k_1 and ω_0 we can get any first order filter, and thus match the $H(s)$ we want.

I would encourage you to try and derive from the signal flow graph the $H(s)$ and prove to your self the equation is correct.



$$H(s) = \frac{V_o(s)}{V_i(s)} = \frac{k_1 s + k_0}{s + w_o}$$

6.3.1 Try to calculate the transfer function from the figure

Signal flow graphs are useful, we can just follow the instructions.

The instructions are 1. any line with a coefficient is a multiplier 1. any box output is a multiplication of the coefficient and the input 1. any sum, well, sum all inputs 1. be aware of gremlins (a sudden -, typing k_1 instead of k_0, etc)

Let's call the $1/s$ box input u

$$u = (k_0 + k_1 s)V_i - \omega_0 V_o$$

$$V_o = u/s$$

$$u = V_o s = (k_0 + k_1 s)V_i - \omega_0 V_o$$

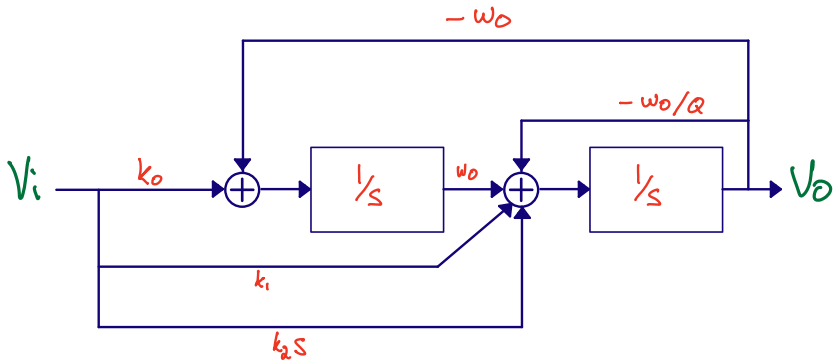
$$(s + \omega_0)V_o = (k_0 + k_1 s)V_i$$

$$\frac{V_o}{V_i} = \frac{k_1 s + k_0}{s + \omega_0}$$

6.4 Second order filter

Bi-quadratic is a general purpose second order filter.

Bi-quadratic just means "there are two quadratic equations". Once we match the k's ω_0 and Q to our wanted $H(s)$ we can proceed with the circuit implementation.



$$H(s) = \frac{k_2 s^2 + k_1 s + k_0}{s^2 + \frac{\omega_0}{Q} s + \omega_o^2}$$

6.4.1 Try to calculate the transfer function from the figure

Follow exactly the same principles as above. If you fail, and you can't find the problem with your algebra, then maybe you need to use Maple, Mathcad, or SymPy if you must find a transfer function in your future job.

I guess you could also spend hours training on examples to get better at the algebra. Personally I find such tasks mind numbingly boring, and of little value, especially since we today can solve the algebra easily with computers.

What's important is to be able to setup the equation in the first place.

6.4.2 How do we implement the filter sections?

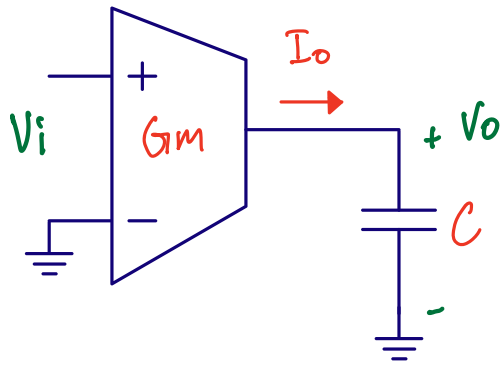
While I'm sure you can invent new types of filters, and there probably are advanced filters, I would say there is roughly three types. Passive filters, that do not have gain. Active-RC filters, with OTAs, and usually trivial to make linear. And Gm-C filters, where we use the transconductance of a transistor and a capacitor to set the coefficients. Gm-C are usually more power efficient than Active-RC, but they are also more difficult to make linear.

In many AFEs, or indeed Sigma-Delta modulator loop filters, it's common to find a first Active-RC stage, and then Gm-C for later stages.

6.5 Gm-C

In the figure below you can see a typical Gm-C filter and the equations for the transfer function. One important thing to note is that this is Gm with capital G, not the g_m that we use for small signal analysis.

In a Gm-C filter the input and output nodes can have significant swing, and thus cannot always be considered small signal.



$$V_o = \frac{I_o}{sC} = \frac{\omega_{ti}}{s} V_i$$

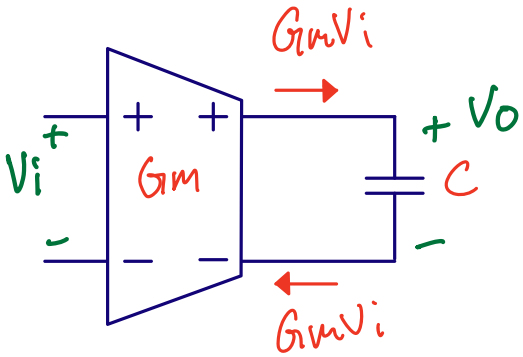
$$\omega_{ti} = \frac{G_m}{C}$$

6.5.1 gm = Gm ?

NO

6.5.2 Differential Gm-C

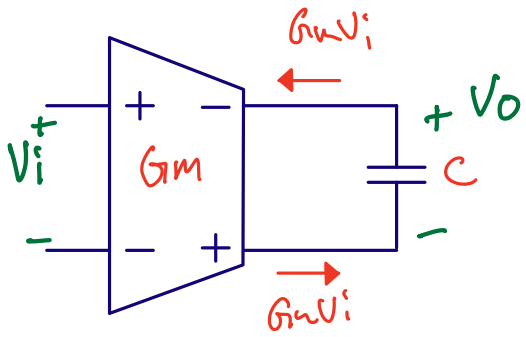
In a real IC we would almost always use differential circuit, as shown below. The transfer function is luckily the same.



$$sCV_o = G_m Vi$$

$$H(s) = \frac{V_o}{V_i} = \frac{G_m}{sC}$$

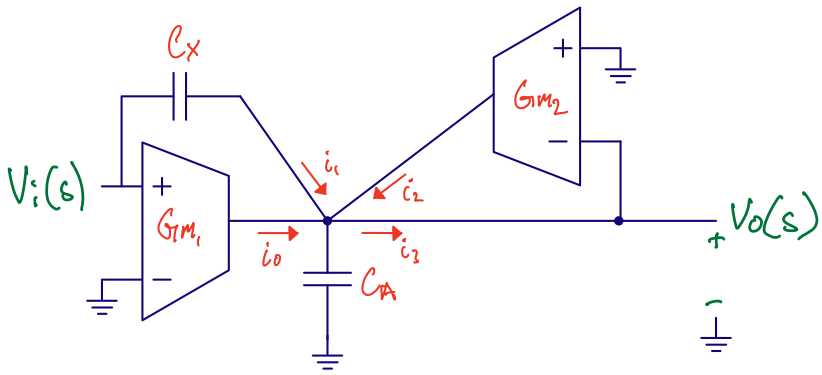
Differential circuits are fantastic for multiple reasons, power supply rejection ratio, noise immunity, symmetric non-linearity, but the qualities I like the most is that the outputs can be flipped to implement negative, or positive gain.



6.5.3 Calculate the transfer function

The figure below shows a implementation of a first-order Gm-C filter that matches our signal flow graph.

I would encourage you to try and calculate the transfer function.



$i_0 =$

$i_1 =$

$i_L =$

$i_3 =$

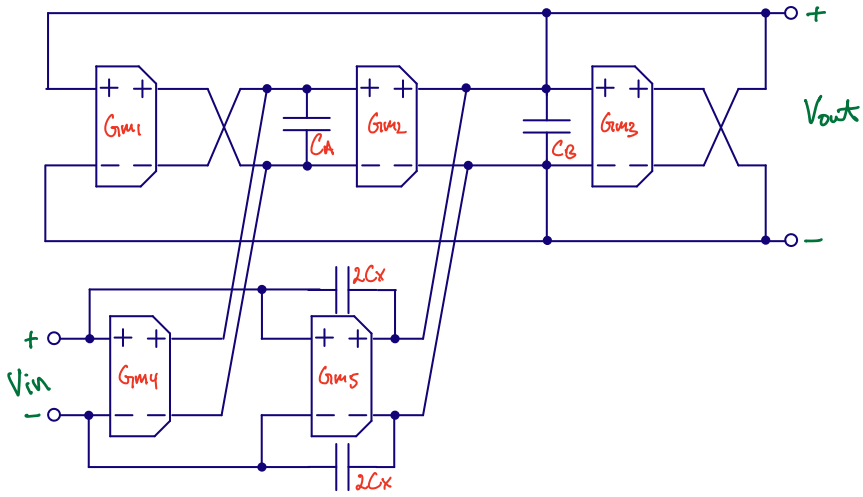
Given the transfer function from the signal flow graph, we see that we can select C_x, C_a and G_m to get the desired k's and ω_0

$$H(s) = \frac{k_1 s + k_0}{s + w_o}$$

$$H(s) = \frac{s\frac{C_x}{C_a+C_x} + \frac{G_{m1}}{C_a+C_x}}{s + \frac{G_{m2}}{C_a+C_x}}$$

6.5.4 Try and calculate the transfer function

Feel free to use a symbolic math tool. The important thing is to figure out how to setup the equations correctly.



$$H(s) = \frac{k_2 s^2 + k_1 s + k_0}{s^2 + \frac{\omega_0}{Q} s + \omega_o^2}$$

$$H(s) = \frac{s^2\frac{C_X}{C_X+C_B} + s\frac{G_{m5}}{C_X+C_B} + \frac{G_{m2}G_{m4}}{C_A(C_X+C_B)}}{s^2 + s\frac{G_{m2}}{C_X+C_B} + \frac{G_{m1}G_{m2}}{C_A(C_X+C_B)}}$$

6.5.5 Try and figure out how we could make a transconductor

Although you can start with the Gm-C cells in the book, I would acutally choose to look at a few papers first.

The main reason is that any book is many years old. Ideas turn into papers, papers turn into books, and by the time you read the book, then there might be more optimal circuits for the technology you work in.

If I were to do a design in 22 nm FDSOI I would first see if someone has already done that, and take a look at the strategy they used. If I can't find any in 22 nm FDSOI, then I'd find a technology close to the same supply voltage.

Start with IEEEXplore

I could not find a 22 nm FDSOI Gm-C based circuit on the initial search. If I was to actually make a Gm-C circuit for industry I would probably spend a bit more time to see if any have done it, maybe expanding to other journals or conferences.

I know of Pieter Harpe, and his work is usually superb, so I would take a closer look at A 77.3-dB SNDR 62.5-kHz Bandwidth Continuous-Time Noise-Shaping SAR ADC With Duty-Cycled Gm-C Integrator

And from Figure 10 a) we can see it's a similar Gm-C cell as chapter 12.5.4 in CJM.

6.6 Active-RC

The Active-RC filter should be well know at this point. However, what might be new is that the open loop gain A_0 and unity gain ω_{ta}

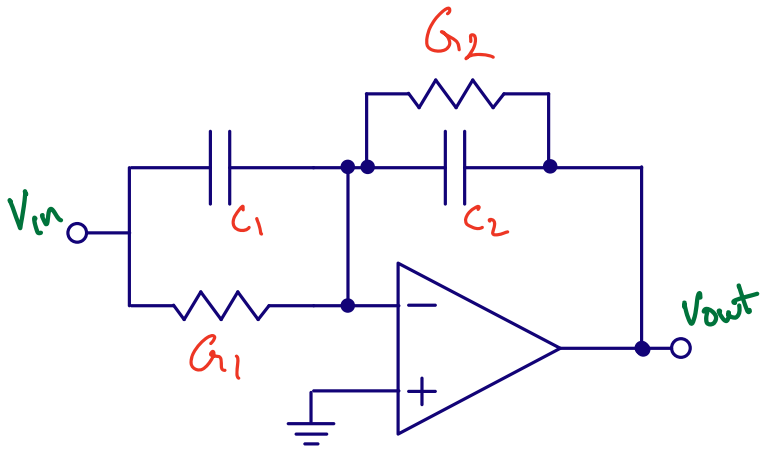
6.6.1 General purpose first order filter

Below is a general purpose first order filter and the transfer function. I've used the condutance $G = \frac{1}{R}$ instead of the resistance. The reason is that it sometimes makes the equations easier to work out.

If you're stuck on calculating a transfer function, then try and switch to conductance, and see if it resolves.

I often get my mind mixed up when calculating transfer functions. I don't know if it's only me, but if it's you also, then don't worry, it's not that often you have to work out transfer functions.

Once in a while, however, you will have a problem where you must calculate the transfer function. Sometimes it's because you'll need to understand where the poles/zeros are in a circuit, or you're trying to come up with a clever idea, or I decide to give this exact problem on the exam.



$$H(s) = \frac{k_1 s + k_0}{s + w_o}$$

$$H(s) = \frac{-\frac{C_1}{C_2} s - \frac{G_1}{C_2}}{s + \frac{G_2}{C_2}}$$

6.6.2 Try and calculate the transfer function

Let's work through the calculation.

6.6.2.1 Step 1: Simplify

The conductance from V_{in} to virtual ground can be written as

$G_{in} = G_1 + sC_1$

The feedback conductance, between V_{out} and virtual ground I write as

$G_{fb} = G_2 + sC_2$

6.6.2.2 Step 2: Remember how an OTA works

An ideal OTA will force its inputs to be the same. As a result, the potential at OTA− input must be 0.

The input current must then be

$I_{in} = G_{in}V_{in}$

Here it's important to remember that there is no way for the input current to enter the OTA. The OTA is high impedance. The input current must escape through the output conductance G_{fb}.

What actually happens is that the OTA will change the output voltage V_{out} until the feedback current , I_{fb}, exactly matches I_{in}. That's the only way to maintain the virtual ground at 0 V. If the currents do not match, the voltage at virtual ground cannot continue to be 0 V, the voltage must change.

6.6.2.3 Step 3: Rant a bit

The previous paragraph should trigger your spidy sense. Words like "exactly matches" don't exist in the real world. As such, how closely the currents match must affect the transfer function. The open loop gain A_0 of the OTA matters. How fast the OTA can react to a change in voltage on the virtual ground, approximated by the unity-gain frequency ω_{ta} (the frequency where the gain of the OTA equals 1, or 0 dB), matters. The input impedance of the OTA, whether the gate leakage of the input differential pair due to quantum tunneling, or the capacitance of the input differential pair, matters. How much current the OTA can deliver (set by slew rate), matters.

Active-RC filter design is "How do I design my OTA so it's good enough for the filter". That's also why, for integrated circuits, you will not have a library of OTAs that you just plug in, and they work.

I would be very suspicious of working anywhere that had an OTA library I was supposed to use for integrated filter design. I'm not saying it's impossible that some company actually has an OTA library, but I think it's a bad strategy. First of all, if an OTA is generic enough to be used "everywhere", then the OTA is likely using too much power, consumes too much area, and is too complex. And the company runs the risk that the designer have not really checked that the OTA works porperly in the filter because "Someone else designed the OTA, I just used in my design".

But, for now, to make our lifes simpler, we assume the OTA is ideal. That makes the equations pretty, and we know what we should get if the OTA actually was ideal.

6.6.2.4 Step 4: Do the algebra

The current flowing from V_{out} to virtual ground is

$$I_{out} = G_{fb}V_{out}$$

The sum of currents into the virtual ground must be zero

$$I_{in} + I_{out} = 0$$

Insert, and do the algebra

$$G_{in}V_{in} + G_{out}V_{out} = 0$$

$$\Rightarrow -G_{in}V_{in} = G_{out}V_{out}$$

$$\frac{V_{out}}{V_{in}} = -\frac{G_{in}}{G_{out}}$$

$$= -\frac{G_1 + sC_1}{G_2 + sC_2}$$

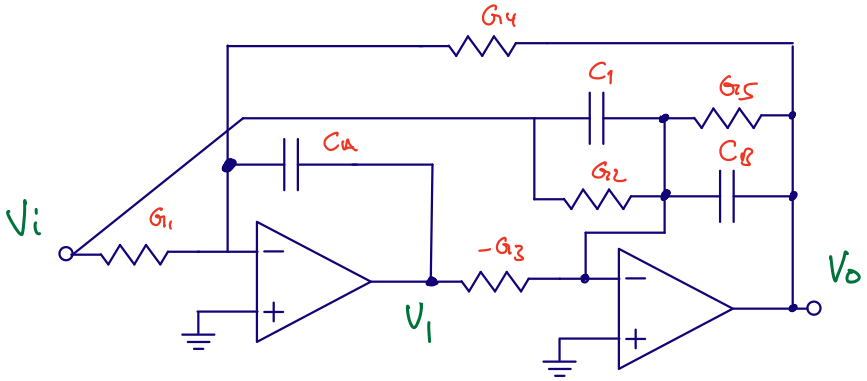
$$= \frac{-s\frac{C_1}{C_2} - \frac{G_1}{C_2}}{s + \frac{G_2}{C_2}}$$

6.6.3 General purpose biquad

A general bi-quadratic active-RC filter is shown below. These kind of general purpose filter sections are quite useful.

Imagine you wanted to make a filter, any filter. You'd decompose into first and second order sections, and then you'd try and match

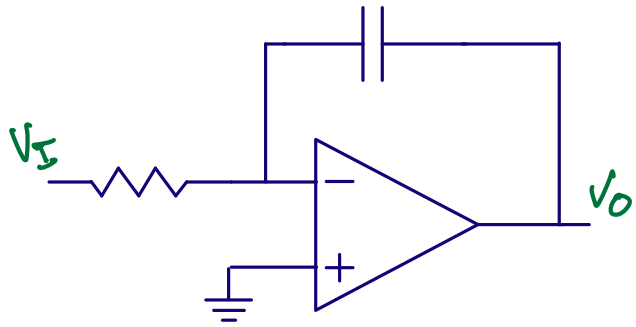
the transfer functions to the general equations.



$$H(s) = \frac{k_2 s^2 + k_1 s + k_0}{s^2 + \frac{\omega_0}{Q}s + \omega_o^2}$$

$$H(s) = \frac{\left[\frac{C_1}{C_B}s^2 + \frac{G_2}{C_B}s + (\frac{G_1 G_3}{C_A C_B})\right]}{\left[s^2 + \frac{G_5}{C_B}s + \frac{G_3 G_4}{C_A C_B}\right]}$$

6.7 The OTA is not ideal



$$H(s) \approx \frac{A_0}{(1 + sA_oRC)(1 + \frac{s}{w_{ta}})}$$

where

$$A_0$$

is the gain of the amplifier, and

$$\omega_{ta}$$

is the unity-gain frequency.

6.7.1 In what region does this equation match an ideal integrator 1/sRC response?

At frequencies above $\frac{1}{A_0RC}$ and below w_{ta} the circuit above is a good approximation of an ideal integrator.

See page 511 in CJM (chapter 5.8.1)

6.8 Example circuit

One place whwere both active-RC and Gm-C filters find a home are continuous time sigma-delta modulators. More on SD later, for now, just know that SD us a combination of high-gain, filtering, simple ADCs and simple DACs to make high resolution analog-to-digital converters.

One such an example is

A 56 mW Continuous-Time Quadrature Cascaded Sigma-Delta Modulator With 77 dB DR in a Near Zero-IF 20 MHz Band

Below we see the actual circuit. It may look complex, and it is.

Not just “complex” as in complicated circuit, it’s also “complex” as in “complex numbers”.

We can see there are two paths “i” and “q”, for “in-phase” and “quadrature-phase”. The fantasitc thing about complex ADCs is that we can have a-symmetric frequency response around 0 Hz.

It will be tricky understanding circuits like this in the beginning, but know that it is possible, and it does get easier to understand.

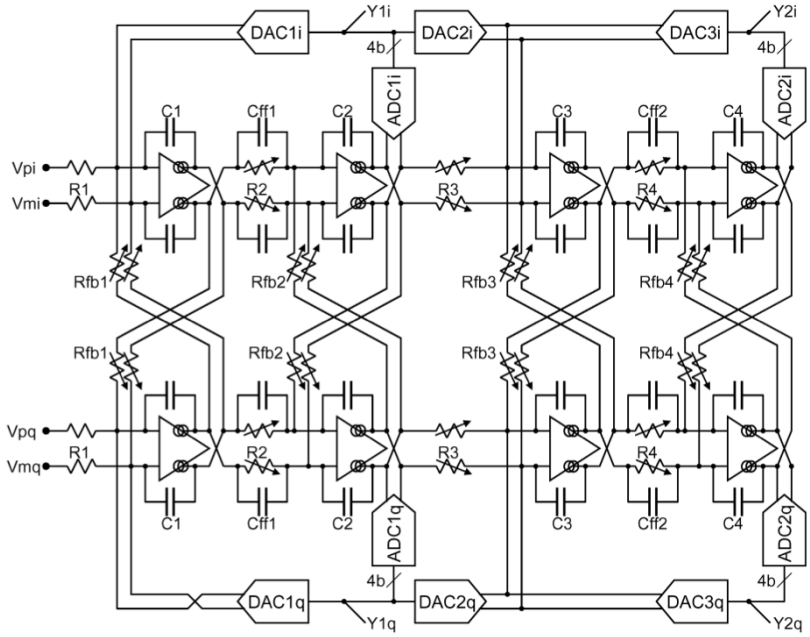
With a complex ADC like this, the first thing to understand is the rough structure.

There are two paths, each path contains 2 ADCs connected in series (Multi-stage Noise-Shaping or MASH). Understanding everything at once does not make sence.

Start with “Vpi” and “Vmi”, make it into a single path (set Rfb1 and Rfb2 to infinite), ignore what happens after R3 and DAC2i.

Now we have a continuous time sigma delta with two stages. First stage is a integrator (R1 and C1), and second stage is a filter (Cff1, R2 and C2). The amplified and filtered signal is sampled by the ADC1i and fed back to the input DAC1i.

It’s possible to show that if the gain from $V(Vpi, Vpm)$ to ADC1i input is large, then $Y1i = V(Vpi, Vpm)$ at low frequencies.



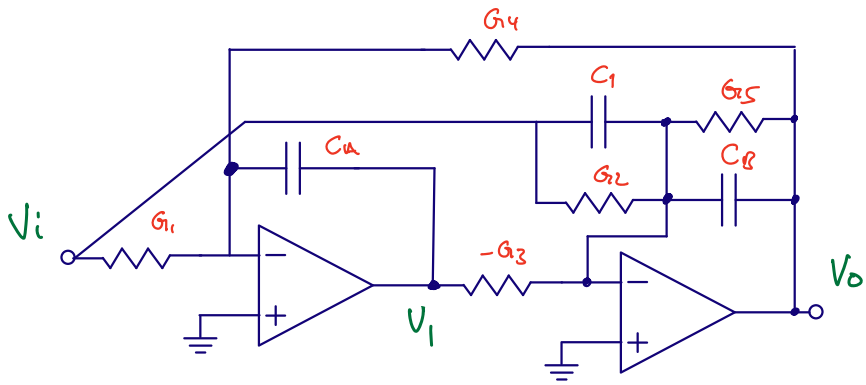
igma-delta modulator design.

Chapter 7

Switched capacitor circuits

7.1 Active-RC

A general purpose Active-RC bi-quadratic (two-quadratic equations) filter is shown below



If you want to spend a bit of time, then try and calculate the transfer function below.

$$H(s) = \frac{\left[\frac{C_1}{C_B}s^2 + \frac{G_2}{C_B}s + (\frac{G_1 G_3}{C_A C_B})\right]}{\left[s^2 + \frac{G_5}{C_B}s + \frac{G_3 G_4}{C_A C_B}\right]}$$

Active resistor capacitor filters are made with OTAs (high output impedance) or OPAMP (low output impedance). Active amplifiers will consume current, and in Active-RC the amplifiers are always on, so there is no opportunity to reduce the current consumption by duty-cycling (turning on and off).

Both resistors and capacitors vary on an integrated circuit, and the 3-sigma variation can easily be 20 %.

The pole or zero frequency of an Active-RC filter is proportional to the inverse of the product between R and C

$$\omega_{p|z} \propto \frac{G}{C} = \frac{1}{RC}$$

As a result, the total variation of the pole or zero frequency is can have a 3-sigma value of

$$\sigma_{RC} = \sqrt{\sigma_R^2 + \sigma_C^2} = \sqrt{0.02^2 + 0.02^2} = 0.028 = 28\ \%$$

On an IC we sometimes need to calibrate the R or C in production to get an accurate RC time constant.

We cannot physically change an IC, every single one of the 100 million copies of an IC is from the same Mask set. That's why ICs are cheap. To make the Mask set is incredibility expensive (think 5 million dollars), but a copy made from the Mask set can cost one dollar or less. To calibrate we need additional circuits.

Imagine we need a resistor of 1 kOhm. We could create that by parallel connection of larger resistors, or series connection of smaller resistors. Since we know the maximum variation is 0.02, then we need to be able to calibrate away +- 20 Ohms. We could have a 980 kOhm resistor, and then add ten 4 Ohm resistors in series that we can short with a transistor switch.

But is a resolution of 4 Ohms accurate enough? What if we need a precision of 0.1%? Then we would need to tune the resistor within +-1 Ohm, so we might need 80 0.5 Ohm resistors.

But how large is the on-resistance of the transistor switch? Would that also affect our precision?

But is the calibration step linear when we add the transistors. If we have a non-linear calibration step, then we cannot use gradient decent calibration algorithms, nor can we use binary search.

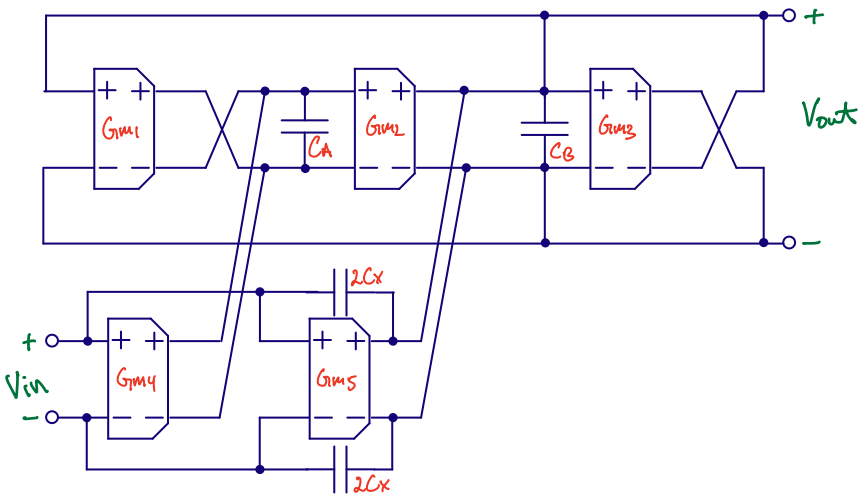
Analog designers need to deal with an almost infinite series of "But", and the experienced designer will know when to stop, when is the "But what if" not a problem anymore.

The most common error in analog integrated circuit design is a "I did not imagine that my circuit could fail in this manner" type of problem. Or, not following the line of "But"'s far enough.

But if we follow all the "But"'s we will never tapeout!

Although active-RC filters are great for linearity and easy to drive, but if we need accurate time constant, there are better alternatives.

7.2 Gm-C



$$H(s) = \frac{\left[s^2 \frac{C_X}{C_X+C_B} + s\frac{G_{m5}}{C_X+C_B} + \frac{G_{m2}G_{m4}}{C_A(C_X+C_B)}\right]}{\left[s^2 + s\frac{G_{m2}}{C_X+C_B} + \frac{G_{m1}G_{m2}}{C_A(C_X+C_B)}\right]}$$

The pole and zero frequency of a Gm-C filter is

$$\omega_{p|z} \propto \frac{G_m}{C}$$

The transconductance accuracy depends on the circuit, and the bias circuit, so we can't give a general, applies for all circuits, sigma number. Capacitors do have 3-sigma 20 % variation, usually.

Same as Active-RC, Gm-C need calibration to get accurate pole or zero frequency.

7.3 Switched capacitor circuits

We can make switches and capacitors behave as a resistor. An example of such a circuit can be found in

A pipelined 5-Msample/s 9-bit analog-to-digital converter

Shown in the figure below. You should think of the switched capacitor circuit as similar to a an amplifier with constant gain. We can use two resistors and an opamp to create a gain. Imagine we create a circuit without the switches, and with a resistor of R from input to virtual ground, and $4R$ in the feedback. Our Active-R would have a gain of $A = 4$.

You might not believe it, but the circuit below is almost the same, but we've used switched capacitor. The complete amplifier still has a $A = 4$, but not all the time.

The switches disconect the OTA and capacitors for half the time, but for the other half, at least for the latter parts of ϕ_2 the gain is four.

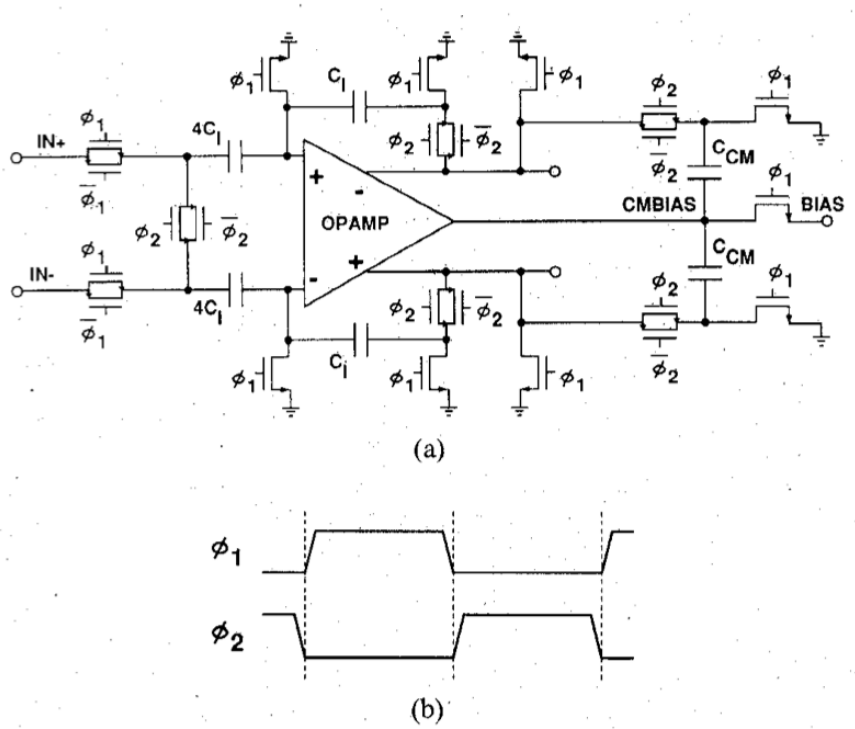


Fig. 6. (a) Schematic of S/H amplifier. (b) Timing diagram of a two-phase nonoverlapping clock.

The output is only correct for a finite, but periodic, time interval. The circuit is discrete time. As long as all circuits afterwards also have a discrete-time input, then it's fine. An ADC can sample the output from the amplifier at the right time, and never notice that the output is shorted to a DC voltage in ϕ_1

Switched capacitor circuits rely on charge transfer. We charge the capacitor $4C$ to the differential input voltage in ϕ_1

$$Q_1 = 4CV_{in}$$

Then we turn off ϕ_1, which opens all switches. The charge on $4C$ will still be Q_1 (except for higher order effects like charge injection from switches).

After a short time (overlap), we turn on ϕ_2, closing some of the switches. The OTA will start to force it's two inputs to be the same, and we short the left side of $4C$. After some time we would have the same voltage on the left side of $4C$ for the two capacitors, and another voltage on the left side of the $4C$ capacitors. The two capacitors must now have the same charge, so the difference in charge, or differential charge must be zero.

Physics tell us that charge is conserved, so our differential charge Q_1 cannot vanish into thin air. The difference in electrons that made Q_1 must be somewhere in our circuit.

Assume the designer of the circuit has done a proper job, then the Q_1 charge will be found on the feedback capacitors.

We now have a Q_1 charge on smaller capacitors, so the differential output voltage must be

$$Q_1 = 4CV_{in} = Q_2 = CV_{out}$$

The gain is

$$A = \frac{V_{out}}{V_{in}} = 4$$

Why would we go to all this trouble to get a gain of 4?

In general, we can sum up with the following equation.

$$\omega_{p|z} \propto \frac{C_1}{C_2}$$

We can use these "switched capacitor resistors" to get pole or zero frequency or gain proportional to a the relative size of capacitors, which is a fantastic feature. Assume we make two identical capacitors in our layout. We won't know the absolute size of the capacitors on the integrated circuit, whether the C_1 is 100 fF or 80 fF, but we can be certain that if $C_1 = 80$ fF, then $C_2 = 80$ fF to a precision of around 0.1 %.

With switched capacitor amplfiers we can set an accurate gain, and we can set an accurate pole and zero frequency (as long as we have an accurate cloc and a high DC gain OTA). The switched capacitor circuits do have a drawback. They are discrete time circuits. As such, we must treat them with caution, and they will always need some analog filter before to avoid a phenomena we call aliasing.

7.4 Discrete-Time Signals

An random, gaussian, continuous time, continuous value, signal has infinite information. The frequency can be anywhere from zero to infinity, the value have infinite levels, and the time division is infinetly small. We cannot store such a signal. We have to quantize.

If we quantize time to $T = 1$ ns, such that we only record the value of the signal every 1 ns, what happens to all the other information? The stuff that changes at 0.5 ns or 0.1 ns, or 1 ns.

We can always guess, but it helps to know, as in absolutely know, what happens. That's where mathemathics come in.

With mathematics we can prove things, and know we're right

7.4.1 The mathematics

Define

$$x_c$$

as a continuous time, continuous value signal

Define

$$\ell(t) = \begin{cases} 1 & \text{if } t \geq 0 \\ 0 & \text{if } t < 0 \end{cases}$$

Define

$$x_{sn}(t) = \frac{x_c(nT)}{\tau}[\ell(t - nT) - \ell(t - nT - \tau)]$$

where $x_{sn}(t)$ is a function of the continuous time signal at the time interval nT.

Define

$$x_s(t) = \sum_{n=-\infty}^{\infty} x_{sn}(t)$$

where $x_s(t)$ is the sampled, continuous time, signal.

Think of a sampled version of an analog signal as an infinite sum of pulse trains where the area under the pulse train is equal to the analog signal.

Why do this?

With a exact definition of a sampled signal in the time-domain it's sometimes possible to find the Laplace transform, and see how the frequency spectrum looks.

If

$$x_s(t) = \sum_{n=-\infty}^{\infty} x_{sn}(t)$$

Then

$$X_{sn}(s) = \frac{1}{\tau}\frac{1-e^{-s\tau}}{s}x_c(nT)e^{-snT}$$

And

$$X_s(s) = \frac{1}{\tau}\frac{1-e^{-s\tau}}{s}\sum_{n=-\infty}^{\infty} x_c(nT)e^{-snT}$$

Thus

$$\lim_{\tau\to 0} \to X_s(s) = \sum_{n=-\infty}^{\infty} x_c(nT)e^{-snT}$$

Or

$$X_s(j\omega) = \frac{1}{T}\sum_{k=-\infty}^{\infty} X_c\left(j\omega - \frac{jk2\pi}{T}\right)$$

The spectrum of a sampled signal is an infinite sum of frequency shifted spectra

or equivalenty

When you sample a signal, then there will be copies of the input spectrum at every

$$nf_s$$

However, if you do an FFT of a sampled signal, then all those infinite spectra will fold down between

$$0 \to f_{s1}/2$$

or

$$-f_{s1}/2 \to f_{s1}/2$$

for a complex FFT

7.4.2 Python discrete time example

If your signal processing skills are a bit thin, now might be a good time to read up on FFT, Laplace transform and But what is the Fourier Transform?

In python we can create a demo and see what happens when we "sample" an "continous time" signal. Hopefully it's obvious that it's impossible to emulate a "continous time" signal on a digital computer. After all, it's digital (ones and zeros), and it has a clock!

We can, however, emulate to any precision we want.

The code below has four main sections. First is the time vector. I use Numpy, which has a bunch of useful features for creating ranges, and arrays.

Secondly, I create continuous time signal. The time vector can be used in numpy functions, like `np.sin()`, and I combine three sinusoid plus some onise. The sampling vector is a repeating pattern of 11001100, so our sample rate should be 1/2'th of the input sample rate. FFT's can be unweieldy beasts. I like to use coherent sampling, however, with mutiple signals and samplerates I did not bother to figure out whether it was possible.

The alternative to coherent sampling is to apply a window function before the FFT, that's the reason for the Hanning window below.

dt.py

```
#- Create a time vector
N = 2**13
t = np.linspace(0,N,N)

#- Create the "continuous time" signal with multiple sinusoidal signals and some noise
f1 = 233/N
fd = 1/N*119
x_s = np.sin(2*np.pi*f1*t) + 1/1024*np.random.randn(N) +   0.5*np.sin(2*np.pi*(f1-fd)*t) + 0.

#- Create the sampling vector, and the sampled signal
t_s_unit = [1,1,0,0,0,0,0,0]
t_s = np.tile(t_s_unit,int(N/len(t_s_unit)))
x_sn = x_s*t_s

#- Convert to frequency domain with a hanning window to avoid FFT bin
#- energy spread
Hann = True
if(Hann):
    w = np.hanning(N+1)
else:
    w = np.ones(N+1)
X_s = np.fft.fftshift(np.fft.fft(np.multiply(w[0:N],x_s)))
X_sn = np.fft.fftshift(np.fft.fft(np.multiply(w[0:N],x_sn)))
```

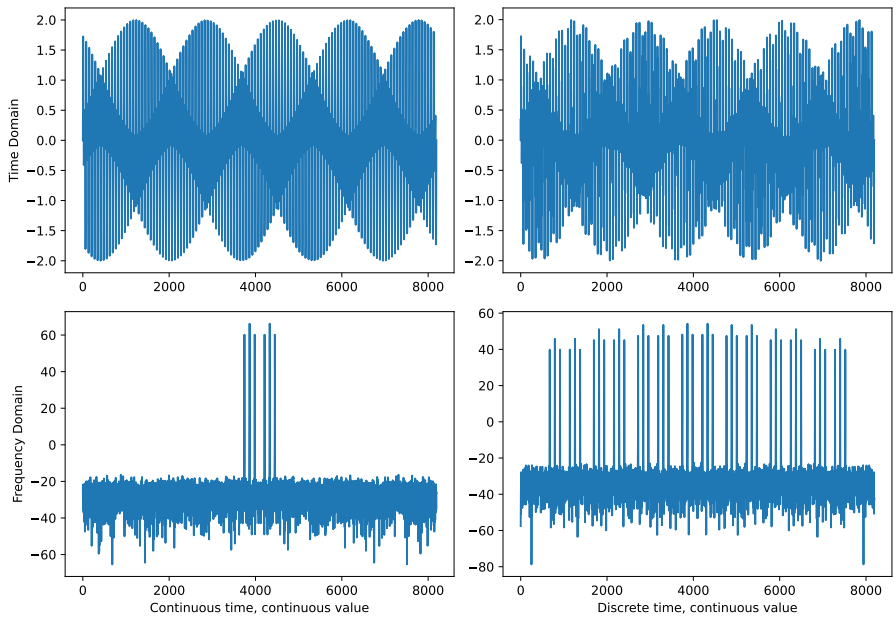
Try to play with the code, and see if you can understand what it does.

Below are the plots. On the left side is the "continous value, continous time" emulation, on the right side "discrete time, continous value".

The top plots are the time domain, while the bottom plots is frequency domain.

The FFT is complex, so that's why there are six sinusoid's bottom left. The "0 Hz" would be at x-axis index 4096 ($2^{13}/2$).

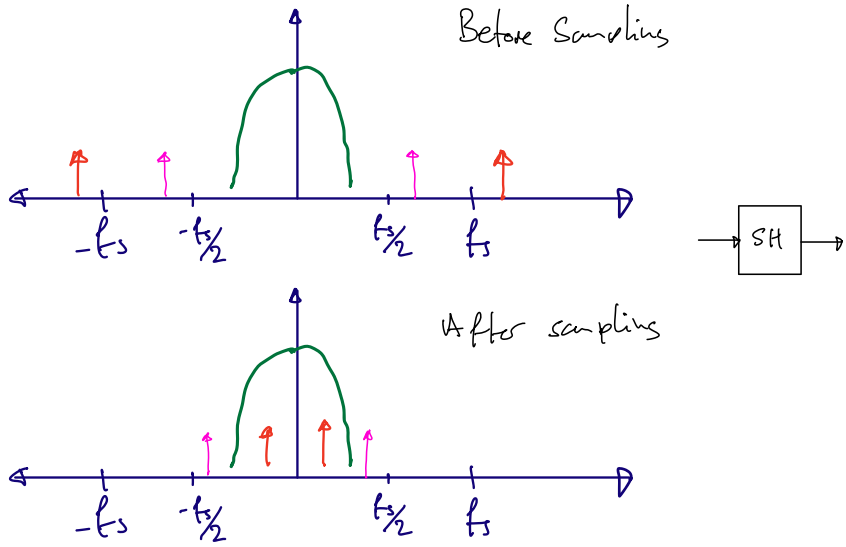
The spectral copies can be seen bottom right. How many spectral copies, and the distance between them will depend on the sample rate (length of `t_s_unit`). Try to play around with the code and see what happens.



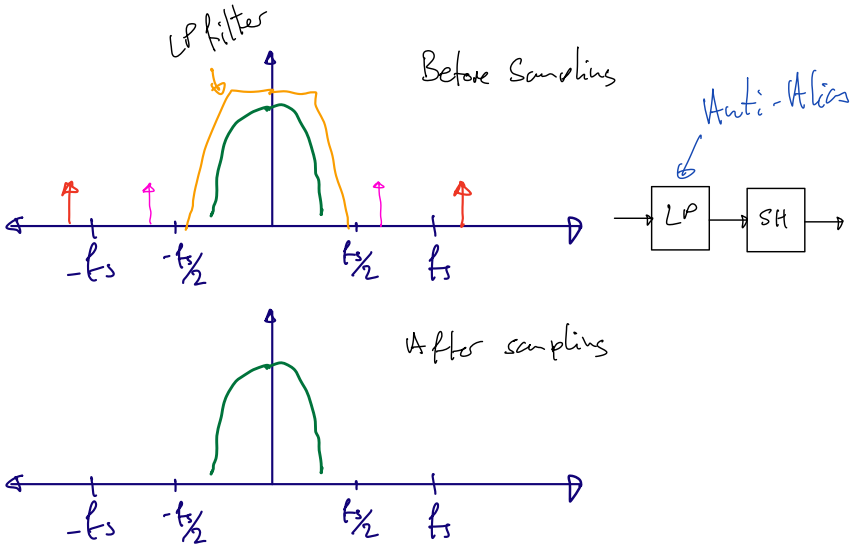
7.4.3 Aliasing, bandwidth and sample rate theory

I want you to internalize that the spectral copies are real. They are not some "mathematical construct" that we don't have to deal with.

They are what happens when we sample a signal into discrete time. Imagine a signal with a band of interest as shown below in greeen. We sample at f_s. The pink and red unwanted signals do not dissapear after sampling, even though they are above the Nyquist frequency ($f_s/2$). They fold around $f_s/2$, and in may appear in-band. That's why it's important to band limit analog signals before they are sampled.



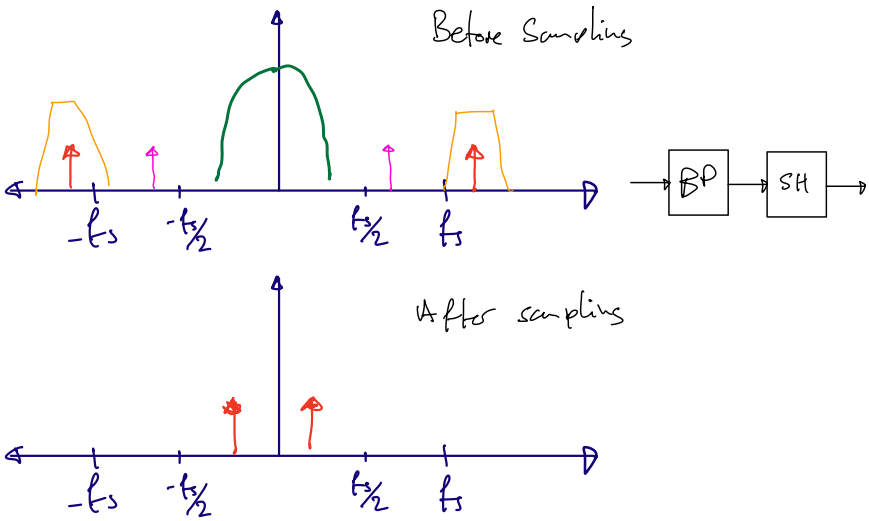
With an anti-alias filter (yellow) we ensure that the unwanted components are low enough before sampling. As a result, our wanted signal (green) is undisturbed.



Assume that we we're interested in the red signal. We could still use a sample rate of f_s. If we bandpass-filtered all but the red signal the red signal would fold on sampling, as shown in the figure below.

Remember that the Nyquist-Shannon states that a suffient no-loss condition is to sample signals with a sample rate of twice the bandwidth of the signal.

Nyquist-Shannon has been extended for sparse signals, compressed sensing, and non-uniform sampling to demonstrate that it's sufficient for the average sample rate to be twice the bandwidth. One 2009 paper Blind Multiband Signal Reconstruction: Compressed Sensing for Analog Signal is a good place to start to devlve into the latest on signal reconstruction.



7.4.4 Z-transform

Someone got the idea that writing

$$X_s(s) = \sum_{n=-\infty}^{\infty} x_c(nT)e^{-snT}$$

was cumbersome, and wanted to find something better.

$$X_s(z) = \sum_{n=-\infty}^{\infty} x_c[n]z^{-n}$$

For discrete time signal processing we use Z-transform

If you're unfamiliar with the Z-transform, read the book or search https://en.wikipedia.org/wiki/Z-transform

The nice thing with the Z-transform is that the exponent of the z tell's you how much delayed the sample $x_c[n]$ is. A block that delays a signal by 1 sample could be described as $x_c[n]z^{-1}$, and an accumulator

$$y[n] = y[n-1] + x[n]$$

in the Z domain would be

$$Y(z) = z^{-1}Y(z) + X(z)$$

With a Z-domain transfer function of

$$\frac{Y(z)}{X(z)} = \frac{1}{1-z^{-1}}$$

7.4.5 Pole-Zero plots

If you're not comfortable with pole/zero plots, have a look at

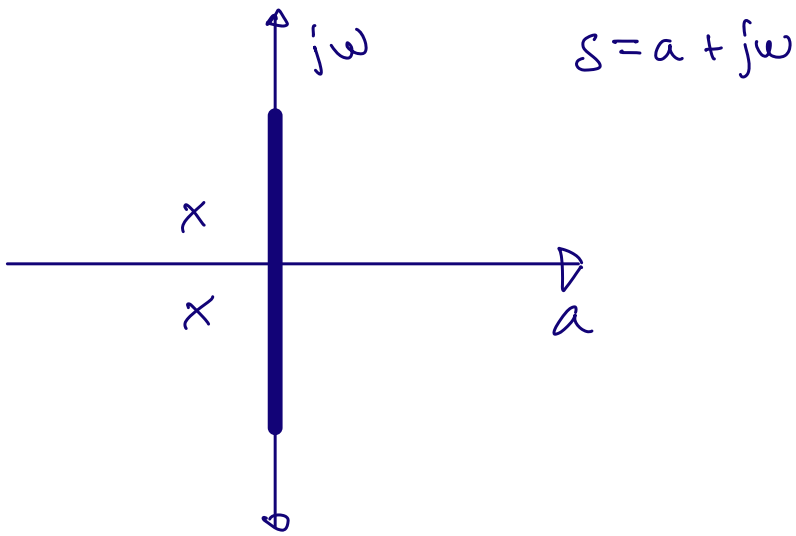
What does the Laplace Transform really tell us

Think about the pole/zero plot as a surface your looking down onto. At $a = 0$ we have the steady state fourier transform. The "x" shows the complex frequency where the fourier transform goes to infinity.

Any real circuit will have complex conjugate, or real, poles/zeros. A combination of two real circuits where one path is shifted 90 degrees in phase can have non-conjugate complex poles/zeros.

If the "x" is $a < 0$, then any pertubation will eventually die out. If the "x" is on the $a = 0$ line, then we have a oscillator that will ring forever. If the "x" is $a > 0$ then the oscillation amplitude will grow without bounds, although, only in Matlab. In any physical circuit an oscillation cannot grow without bounds forever.

Growing without bounds is the same as "being unstable".



7.4.6 Z-domain

Spectra repeat every

$$2\pi$$

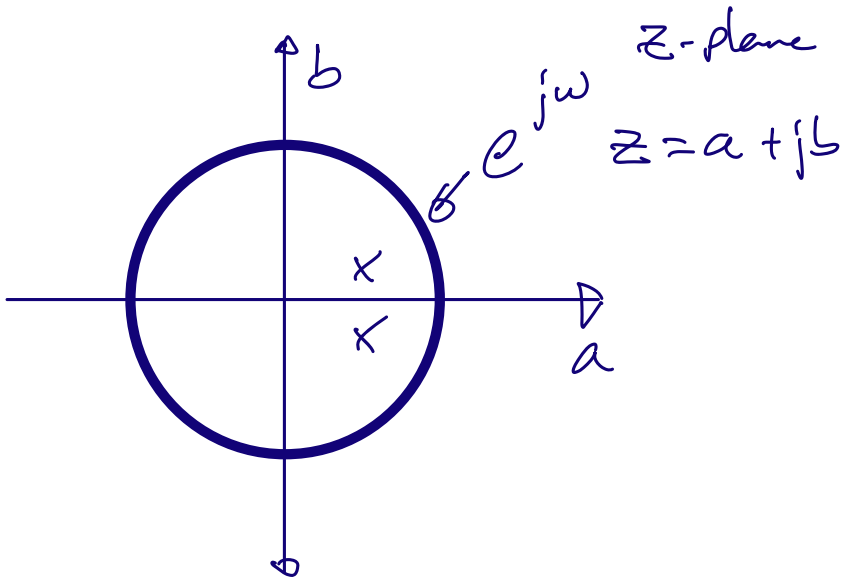
As such, it does not make sense to talk about a plane with a a and a $j\omega$. Rather we use the complex number $z = a + jb$.

As long as the poles ("x") are within the unit circle, oscillations will die out. If the poles are on the unit-circle, then we have an oscillator. Outside the unit circle the oscillation will grow without bounds, or in other words, be unstable.

We can translate between Laplace-domain and Z-domain with the Bi-linear transform

$$s = \frac{z-1}{z+1}$$

Warning: First-order approximation https://en.wikipedia.org/wiki/Bilinear_transform



7.4.7 First order filter

Assume a first order filter given by the discrete time equation.

$$y[n+1] = bx[n] + ay[n] \Rightarrow Yz = bX + aY$$

The "n" index and the "z" exponent can be choosen freely, which sometimes can help the algebra.

$$y[n] = bx[n-1] + ay[n-1] \Rightarrow Y = bXz^{-1} + aYz^{-1}$$

The transfer function can be computed as

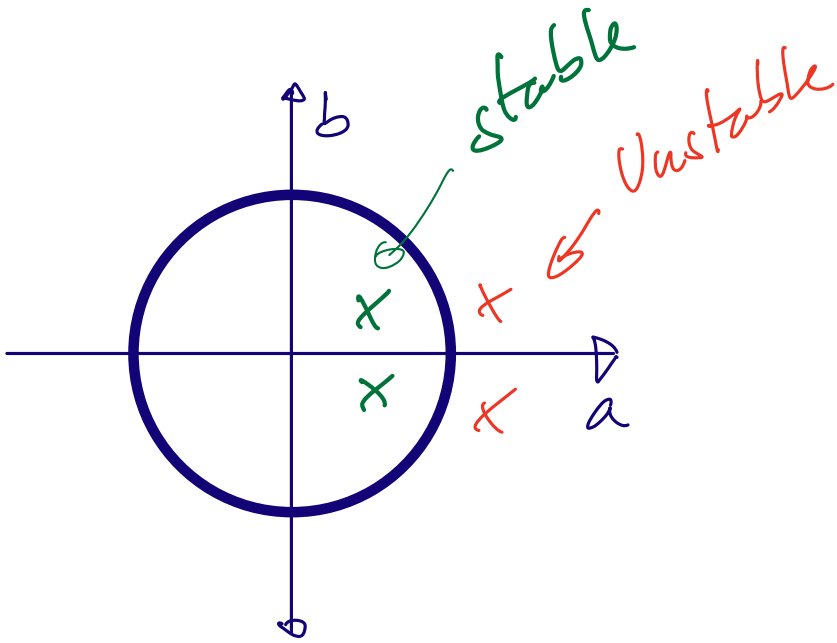
$$H(z) = \frac{b}{z-a}$$

From the discrete time equation we can see that the impulse will never die out. We're adding the previous output to the current input. That means the circuit has infinite memory. Accordingly, filters of this type are known as. Infinite-impulse response (IIR)

$$h[n] = \begin{cases} k & \text{if } n < 1 \\ a^{n-1}b + a^n k & \text{if } n \geq 1 \end{cases}$$

Head's up: Fig 13.12 in AIC is wrong

From the impulse response it can be seen that if $a > 1$, then the filter is unstable. Same if $b > 1$. As long as $|a + jb| < 1$ the filter should be stable.



The first order filter can be implemented in python, and it's really not hard. See below. The x_sn vector is from the previous python example.

There are smarter, and faster ways to do IIR filters (and FIR) in python, see scipy.signal.iirfilter

iir.py

```
#- IIR filter
b = 0.3
a = 0.25
z = a + 1j*b
z_abs = np.abs(z)
print("|z| = " + str(z_abs))
y = np.zeros(N)
y[0] = a
for i in range(1,N):
    y[i] = b*x_sn[i-1] + y[i-1]
```

From the plot below we can see the sampled time domain and spectra on the left, and the filtered time domain and spectra on the right.

The IIR filter we implemented above is a low-pass filter, and the filter partially rejects the copied spectra, as expected.

media/l5$_i$ir.svg

7.4.8 Finite-impulse response(FIR)

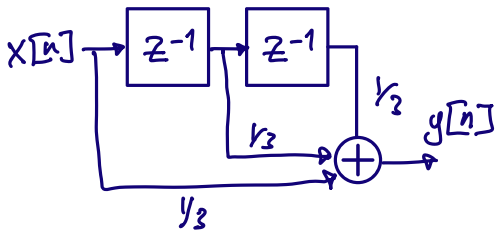
FIR filters are unconditionaly stable, since the impulse response will always die out. FIR filters are a linear sum of delayed inputs.

In my humble opinion, there is nothing wrong with an IIR. Yes, the could become unstable, however, they can be designed safely. I'm not sure there is a theological feud on IIR vs FIR, I suspect there could be. Talk to someone that knows digital filters better than me.

But be wary of rules like "IIR are always better than FIR" or visa versa. Especially if statements are written in books. Remember that the book was probably written a decade ago, and based on papers two decades old, which were based on three decades old state of the art. Our abilities to use computers for design has improved a bit the last three decades.

Try to modify the IIR filter, and implement the FIR below instead.

$$H(z) = \frac{1}{3}\sum_{i=0}^{2} z^{-1}$$



7.5 Switched-Capacitor

Below is an example of a switched-capacitor circuit. It's drawn twice, because there are usually at least 2 phases to a switched-capacitor circuit. Think of the two phases as two different configurations of a circuit, each with a specific purpose.

On the left we have the SC circuit during the sampling phase. Imagine that we somehow have stored a voltage $V_1 = \ell$ on capacitor C_1 (the switches for that sampling or storing are not shown). The charge on C_1 is $Q_1 = C_1 V_1 = C_1 \ell$

The C_2 capacitor is shorted, as such, $V_2 = 0$, which must mean that the charge on C_2 given by $Q_2 = 0$.

The voltage at the negative input of the OTA must be 0 V, as the positive input is 0 V, and we assume the circuit has settled all transients.

Imagine we (very carefully) open the circuit around C_2 and close the circuit from the negative side of C_1 to the OTA negative input. It's the OTA that ensures that the negative input is the same as the positive input, but the OTA cannot be infinetly fast. At the same time, the voltage across C_1 cannot change instantaniously. Neither can the voltage across C_2. As such, the voltage at the negative input must immediately go to $-\ell$ (ignoring any parasitic capacitance at the negative input).

The OTA does not like it's inputs to be different, so it will start to charge C_2 to increase the voltage at the negative input to the OTA. When the negative input reaches 0 V the OTA is happy again.

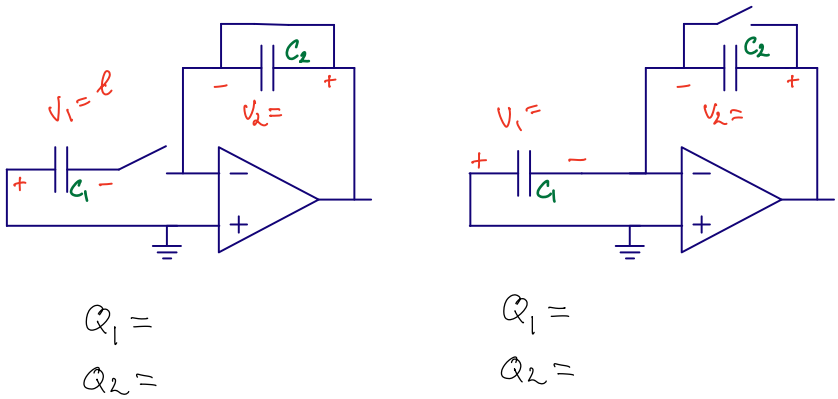
A key point is, that even the voltages now have changed, there is zero volt across C_1, and thus there cannot be any charge across C_1 the charge that was there cannot have dissapeared. The negative input of the OTA is a high impedance node, and cannot supply charge. The charge must have gone somewhere, but where?

In process of changing the voltage at the negative input of the OTA we've changed the voltage across C_2. The voltage change must exactly match the charge that was across C_1, as such

$$Q_1 = C_1\ell = Q_2 = C_2 V_O$$

thus

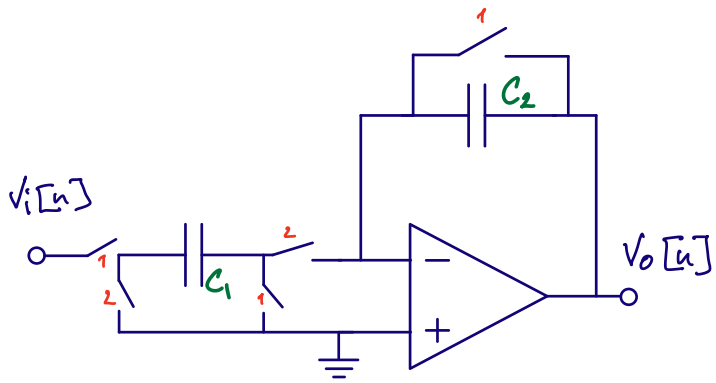
$$\frac{V_O}{\ell} = \frac{C_1}{C_2}$$



7.5.1 Switched capacitor gain circuit

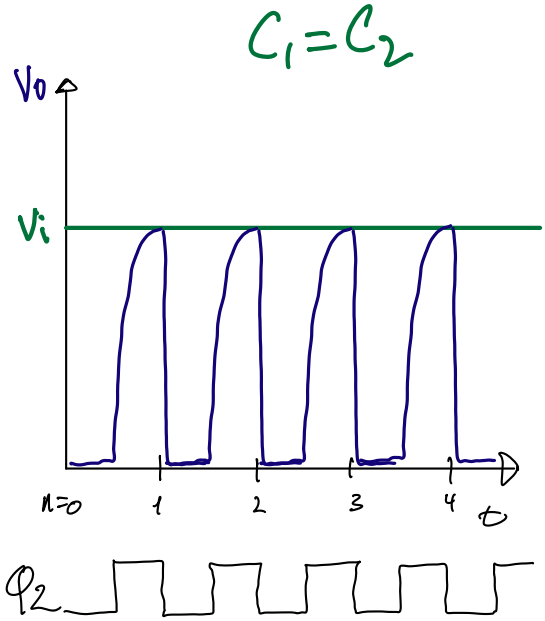
Redrawing the previous circuit, and adding a few more swithces we can create a switched capacitor gain circuit.

There is now a switch to sample the input voltage across C_1 during phase 1 and reset C_2. During phase 2 we configure the circuit to leverage the OTA to do the charge transfer from C_1 to C_2.



The discrete time output from the circuit will be as shown below. It's only at the end of the second phase that the output signal is valid. As a result, it's common to use the sampling phase of the next circuit close to the end of phase 2.

For charge to be conserved the clocks for the switch phases must never be high at the same time.



The discrete time, Z-domain and transfer function is shown below. The transfer function tells us that the circuit is equivalent to a gain, and a delay of one clock cycle. The cool thing about switch capacitor circuits is that the precision of the gain is set by the relative size between two capacitors. In most technologies that relative sizing can be better than 0.1 %.

Gain circuits like the one above find use in most Pipelined ADCs, and are common, with some modifications, in Sigma-Delta ADCs.

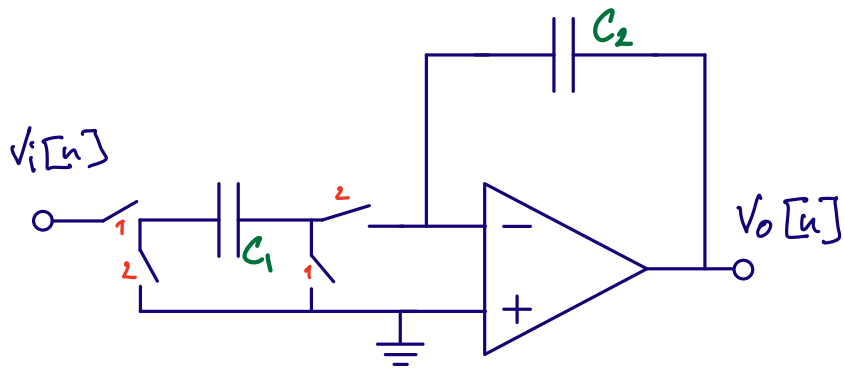
$$V_o[n+1] = \frac{C_1}{C_2} V_i[n]$$

$$V_o z = \frac{C_1}{C_2} V_i$$

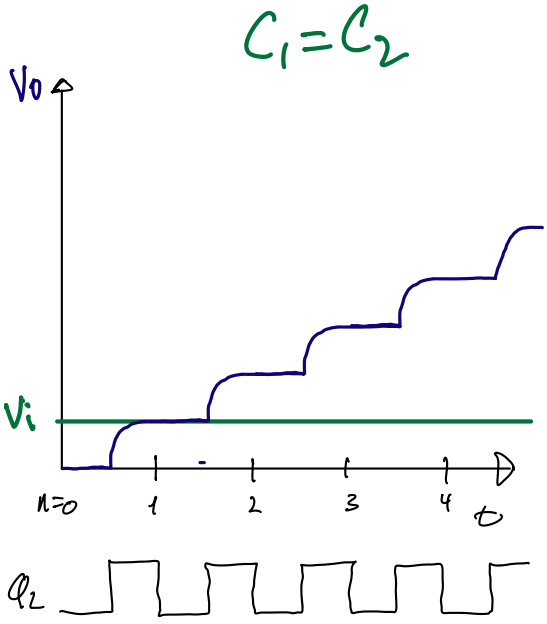
$$\frac{V_o}{V_i} = H(z) = \frac{C_1}{C_2} z^{-1}$$

7.5.2 Switched capacitor integrator

Removing one switch we can change the function of the switched capacitor gain circuit. If we don't reset C_2 then we accumulate the input charge every cycle.



The output now will grow without bounds, so integrators are most often used in filter circuits, or sigma-delta ADCs where there is feedback to control the voltage swing at the output of the OTA.



Make sure you read and understand the equations below, it's good to realize that discrete time equations, Z-domain and transfer functions in the Z-domain are actually easy.

$$V_o[n] = V_o[n-1] + \frac{C_1}{C_2} V_i[n-1]$$

$$V_o - z^{-1} V_o = \frac{C_1}{C_2} z^{-1} V_i$$

Maybe one confusing thing is that multiple transfer functions can mean the same thing, as below.

$$H(z) = \frac{C_1}{C_2}\frac{z^{-1}}{1-z^{-1}} = \frac{C_1}{C_2}\frac{1}{z-1}$$

7.5.3 Noise

Capacitors don't make noise, but switched-capacitor circuits do have noise. The noise comes from the thermal, flicker, burst noise in the switches and OTA's. Both phases of the switched capacitor circuit contribute noise. As such, the output noise of a SC circuit is usually

$$V_n^2 > \frac{2kT}{C}$$

I find that sometimes it's useful with a repeat of mathematics, and since we're talking about noise.

The mean, or average of a signal is defined as

Mean

$$\overline{x(t)} = \lim_{T\to\infty}\frac{1}{T}\int_{-T/2}^{+T/2} x(t)dt$$

Define

Mean Square

$$\overline{x^2(t)} = \lim_{T\to\infty}\frac{1}{T}\int_{-T/2}^{+T/2} x^2(t)dt$$

How much a signal varies can be estimated from the Variance

$$\sigma^2 = \overline{x^2(t)} - \overline{x(t)}^2$$

where

$$\sigma$$

is the standard deviation. If mean is removed, or is zero, then

$$\sigma^2 = \overline{x^2(t)}$$

Assume two random processes,

$$x_1(t)$$

and

$$x_2(t)$$

with mean of zero (or removed).

$$x_{tot}(t) = x_1(t) + x_2(t)$$

$$x_{tot}^2(t) = x_1^2(t) + x_2^2(t) + 2x_1(t)x_2(t)$$

Variance (assuming mean of zero)

$$\sigma_{tot}^2 = \lim_{T\to\infty}\frac{1}{T}\int_{-T/2}^{+T/2} x_{tot}^2(t)dt$$

$$\sigma_{tot}^2 = \sigma_1^2 + \sigma_2^2 + \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{+T/2} 2x_1(t)x_2(t)dt$$

Assuming uncorrelated processes (covariance is zero), then

$$\sigma_{tot}^2 = \sigma_1^2 + \sigma_2^2$$

In other words, if two noises are uncorrelated, then we can sum the variances. If the noise sources are correlated, for example, noise comes from the same transistor, but takes two different paths through the circuit, then we cannot sum the variances. We must also add the co-variance.

7.5.4 Sub-circuits for SC-circuits

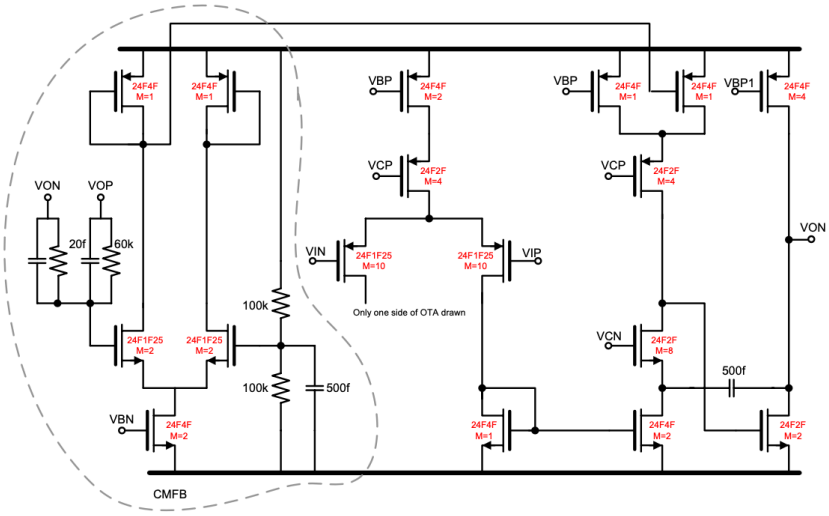
Switched-capacitor circuits are so common that it's good to delve a bit deeper, and understand the variants of the components that make up SC circuits.

7.5.4.1 OTA

At the heart of the SC circuit we usually find an OTA. Maybe a current mirror, folded cascode, recycling cascode, or my favorite: a fully differential current mirror OTA with cascoded, gain boosted, output stage using a parallel common mode feedback.

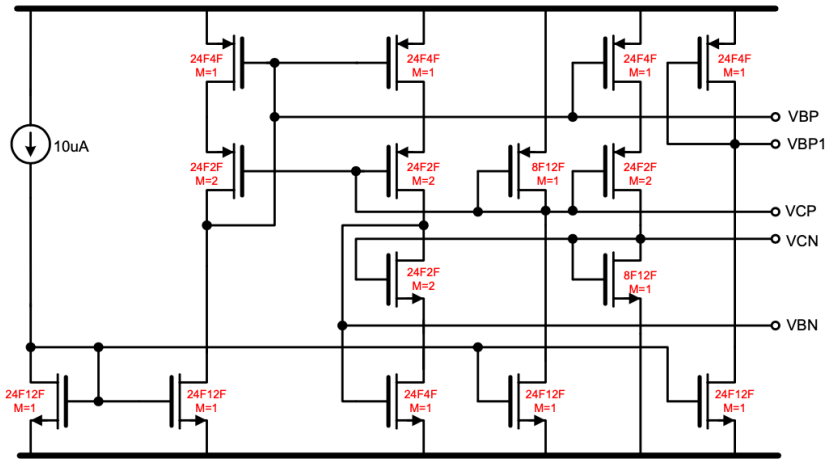
Not all SC circuits use OTAs, there are also comparator based SC circuits.

Below is a fully-differential two-stage OTA that will work with most SC circuits. The notation "24F1F25" means "the width is 24 F" and "length is 1.25 F", where "F" is the minimum gate length in that technology.



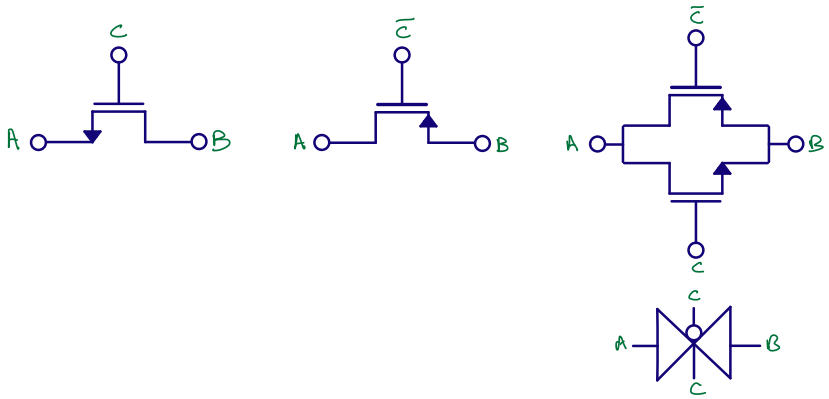
F = minimum transistor gate length. For example 24F4F => W = 24 x min gate, L= 4 x min gate

As bias circuit to make the voltages the below will work



7.5.4.2 Switches

If your gut reaction is "switches, that's easy", then you're very wrong. Switches can be incredibly complicated. All switches will be made of transistors, but usually we don't have enough headroom to use a single NMOS or PMOS. We may need a transmission gate



The challenge with transmission gates is that when the voltage at the input is in the middle between VDD and ground then both PMOS and NMOS, although they are on , they might not be that on. Especially in nano-scale CMOS with a 0.8 V supply and 0.5 V threshold voltage. The resistance mid-rail might be too large.

For switched-capacitor circuits we must settle the voltages to the required accuracy. In general

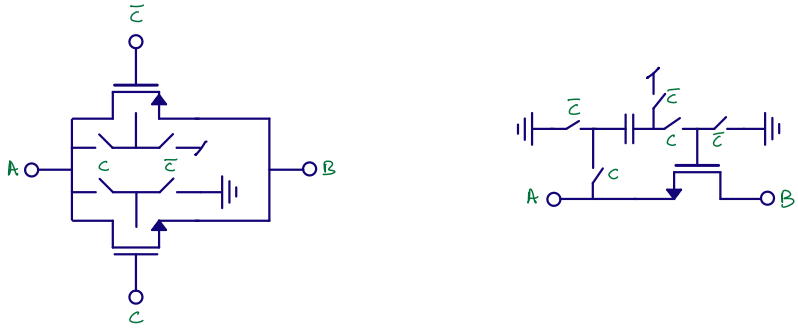
$$t > -\log \text{error} \tau$$

For example, for a 10-bit ADC we need $t > -\log(1/1024)\tau = 6.9\tau$. This means we need to wait at least 6.9 time constants for the voltage to settle to 10-bit accuracy in the switched capacitor circuit.

Assume the capacitors are large due to noise, then the switches must be low resistance for a resonable time constant. Larger switches have smaller resistance, however, they also have more charge in the inversion

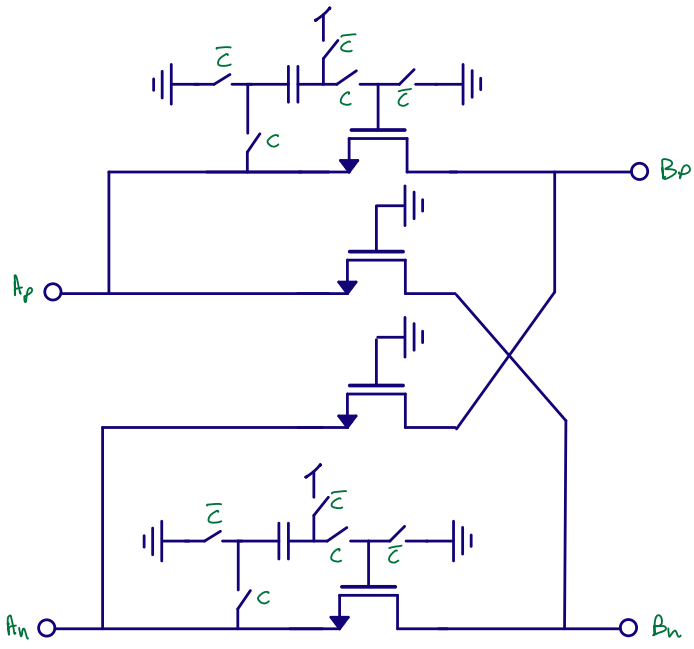
layer, which leads to charge injection when the switches are turned of. Accordingly, larger switches are not always the solution.

Sometimes it may be sufficient to switch the bulks, as shown on the left below. But more often that one would like, we have to implement bootstrapped switches as shown on the right.



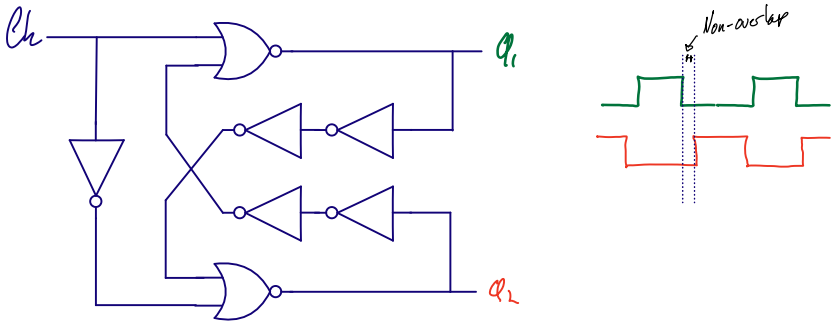
The switch I used in my JSSC SAR looks like the one below.

Do not underestimate the amount of work that the switches requires. I know a case where 3 really good designers spent 6 months on trying to design the input switches for an ADC.



7.5.4.3 Non-overlapping clocks

The non-overlap generator is standard. Use the one shown below. Make sure you simulate that the non-overlap is sufficient in all corners.


Ck
ϕ_1
ϕ_2
Non-overlap

Chapter 8

Oversampling and Sigma-Delta ADCs

8.1 ADC state-of-the-art

The performance of an analog-to-digital converter is determined by the effective number of bits (ENOB), the power consumption, and the maximum bandwidth.

The effective number of bits contains information on the linearity of the ADC. The power consumption shows how efficient the ADC is. The maximum bandwidth limits what signals we can sample and reconstruct in digital domain.

Many years ago, Robert Walden did a study of ADCs, one of the plot's is shown below.

1999, R. Walden: Analog-to-digital converter survey and analysis

There are obvious trends, the faster an ADC is, the less precise the ADC is (lower SNDR). There are also fundamental limits, Heisenberg tells us that a 20-bit 10 GS/s ADC is impossible, according to Walden.

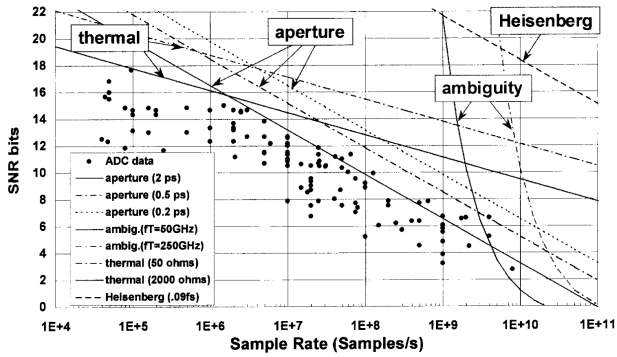


Fig. 7. Signal-to-noise ratio according to SNR-bits = (SNR(dB) − 1.76)/6.02. Three sets of curves show performance limiters due to thermal noise, aperture uncertainty, and comparator ambiguity. The Heisenberg limit is also displayed.

I would take those fundamental limits with a grain of salt. The uncertainty principle states that the precision we can determine position and the momentum of a particle is

$$\sigma_x \sigma_p \geq \frac{\hbar}{2}$$

In the paper, however, Walden quickly states that

$$\Delta E \Delta t > \frac{h}{2\pi}$$

where ΔE is the difference in energy, and Δt is the difference in time. Whether that is a valid reformulation of the uncertainty principle, I'm not sure. Also, the plot assumes 50 Ohm and 1 V full-scale. As a result, the "Heisenberg" line that appears to be unbreakable certainly is breakable. Just change the voltage to 100 V, and the number of bits can be much higher. Always check the assumptions.

A more recent survey of ADCs comes from Boris Murmann. He still maintains a list of the best ADCs from ISSCC and VLSI Symposium.

B. Murmann, ADC Performance Survey 1997-2022 (ISSCC & VLSI Symposium)

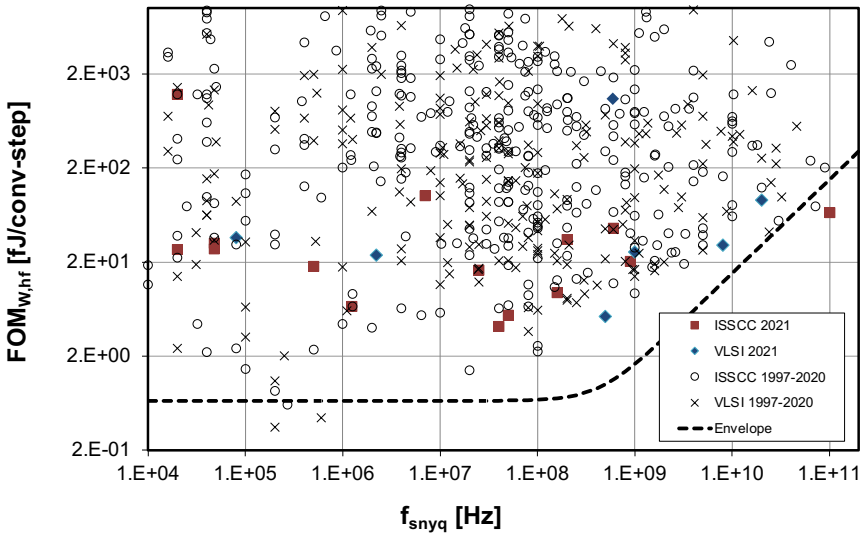
A common figure of merit for low-to-medium resolution ADCs is the Walden figure of merit, defined as

$$FOM_W = \frac{P}{2^B f_s}$$

Below 10 fJ/conv.step is good.

Below 1 fJ/conv.step is extreme.

In the plot below you can see the ISSCC and VLSI ADCs.



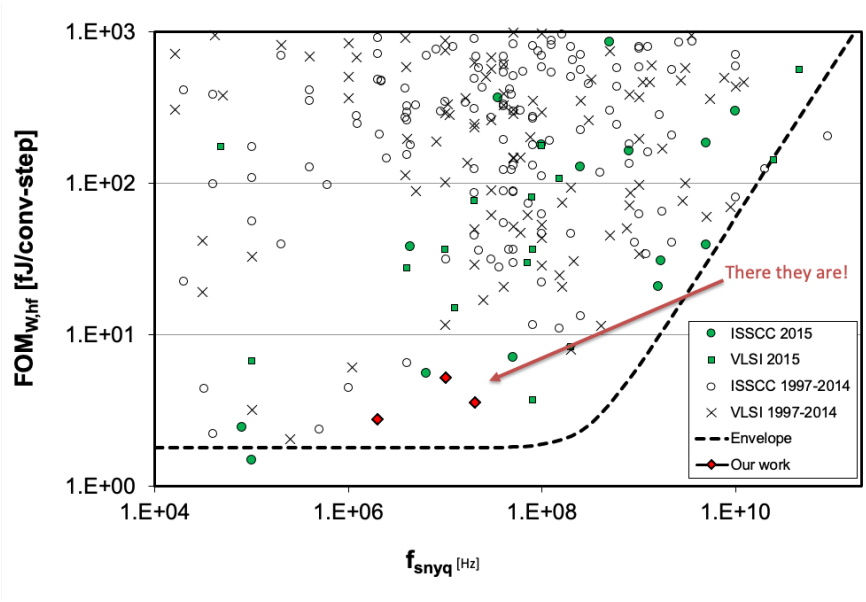
People from NTNU have made some of the worlds best ADCs

If you ever want to make an ADC, and you want to publish the measurements, then you must be better than most. A good algorithm for state-of-the-art ADC design is to first pick a sample rate with low number of data (blank spaces in the plot above), then read the papers in the vicinity of the blank space to understand the application, then set a target FOM which is best in world, then try and find a ADC architecture that can achieve that FOM.

That's pretty much the algorithm I, and others, have followed to make state-of-the-art ADCs. A few of the NTNU ADCs are:

A Compiled 9-bit 20-MS/s 3.5-fJ/conv.step SAR ADC in 28-nm FDSOI for Bluetooth Low Energy Receivers

A 68 dB SNDR Compiled Noise-Shaping SAR ADC With On-Chip CDAC Calibration



For high-resolution ADCs, it's more common to use the Schreier figure of merit, which can also be found in

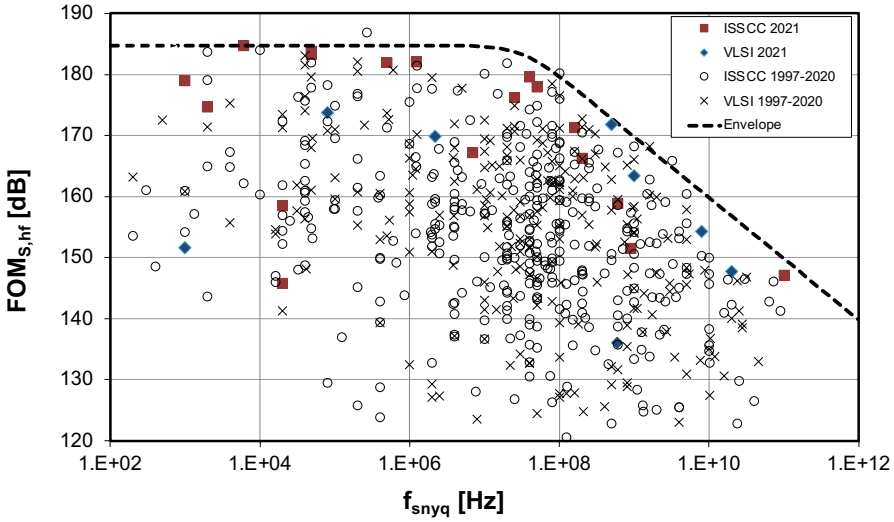
B. Murmann, ADC Performance Survey 1997-2022 (ISSCC & VLSI Symposium)

The Walden figure of merit assumes that thermal noise does not constrain the power consumption of the ADC, which is usually true for low-to-medium resolution ADCs. To keep the Walden FOM you can double the power for a one-bit increase in ENOB. If the ADC is limited by thermal noise, however, then you must quadruple the capacitance (reduce kT/C noise power) for each 1-bit ENOB increase. Accordingly, the power must also go up four times.

For higher resolution ADC the power consumption is set by thermal noise, and the Schreier FOM allows for a 4x power consumption increase for each added bit.

$$FOM_S = SNDR + 10\log\left(\frac{f_s/2}{P}\right)$$

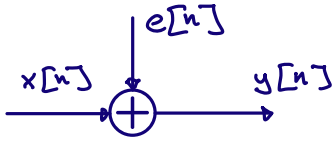
Above 180 dB is extreme



8.2 Quantization

Sampling turns continuous time into discrete time. Quantization turns continuous value into discrete value. Any complete ADC is always a combination of sampling and quantization.

In our mathematical drawings of quantization we often define $y[n]$ as the output, the quantized signal, and $x[n]$ as the discrete time, continuous value input, and we add some "noise", or "quantization noise" $e[n]$, where $x[n] = y[n] - e[n]$.



Maybe you've even heard the phrase "Quantization noise is white" or "Quantization noise is a random Gaussian process"?

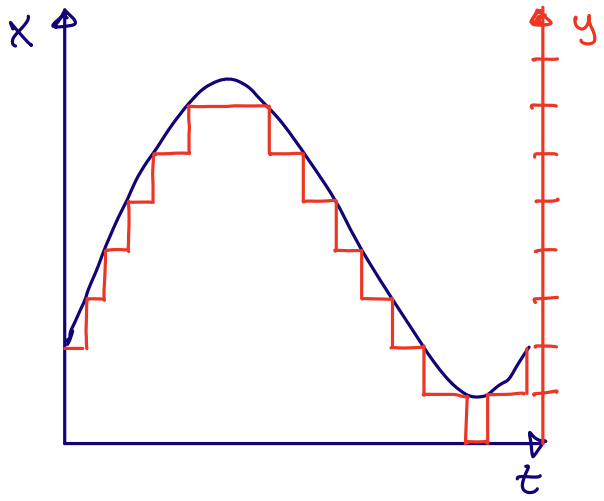
I'm here to tell you that you've been lied to. Quantization noise is not white, nor is it a Gaussian process. Those that have lied to you may say "yes, sure, but for high number of bits it can be considered white noise". I would say that's similar to saying "when you look at the earth from the moon, the surface looks pretty smooth without bumps, so let's say the earth is smooth with no mountains".

I would claim that it's an unnecessary simplification. It's obvious to most that the earth would appear smooth from really far away, but they would not surprised by Mount Everest, since they know it's not smooth.

An Alien that has been told that the earth is smooth, would be surprised to see Mount Everest.

But if Quantization noise is not white, what is it?

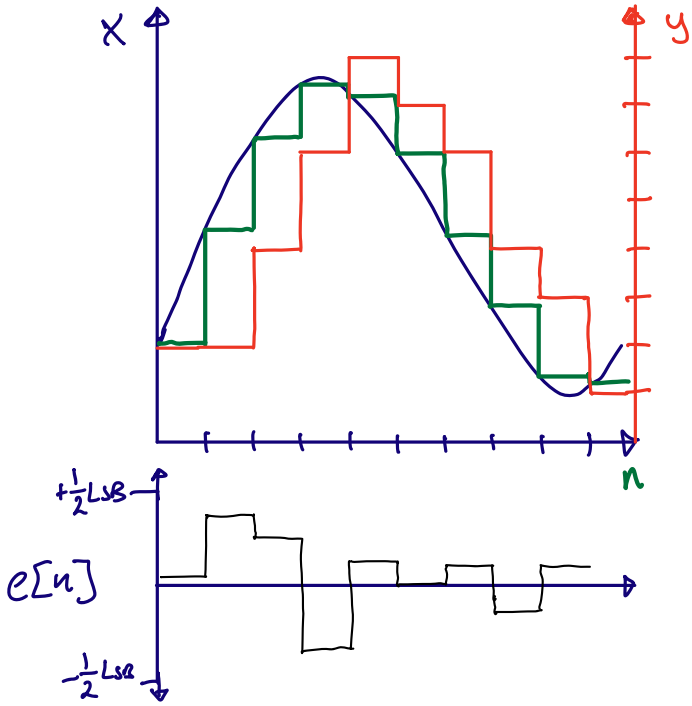
The figure below shows the input signal x and the quantized signal y.



To see the quantization noise, first take a look at the sample and held version of x in green in the figure below. The difference between the green (x at time n) and the red (y) would be our quantization noise e.

The quantization noise is contained between $+\frac{1}{2}$ Least Significant Bit (LSB) and $-\frac{1}{2}$ LSB.

This noise does not look random to me, but I can't see what it is, and I'm pretty sure I would not be able to work it out either.



Luckily, there are people in this world that love mathematics, and that can delve into the details and figure out what $e[n]$ is. A guy called Blachman wrote a paper back in 1985 on quantization noise.

See The intermodulation and distortion due to quantization of sinusoids for details

In short, quantization noise is defined as

$$e_n(t) = \sum_{p=1}^{\infty} A_p \sin p\omega t$$

where p is the harmonic index, and

$$A_p = \begin{cases} \delta_{p1} A + \sum_{m=1}^{\infty} \frac{2}{m\pi} J_p(2m\pi A) & , p = \text{ odd} \\ 0 & , p = \text{ even} \end{cases}$$

$$\delta_{p1} \begin{cases} 1 & , p = 1 \\ 0 & , p \neq 1 \end{cases}$$

and

$$J_p(x)$$

is a Bessel function of the first kind, A is the amplitude of the input signal.

If we approximate the amplitude of the input signal as

$$A = \frac{2^n - 1}{2} \approx 2^{n-1}$$

where n is the number of bits, we can rewrite as

$$e_n(t) = \sum_{p=1}^{\infty} A_p \sin p\omega t$$

$$A_p = \delta_{p1} 2^{n-1} + \sum_{m=1}^{\infty} \frac{2}{m\pi} J_p(2m\pi 2^{n-1}), p = odd$$

Obvious, right?

I must admit, it's not obvious to me. But I do understand the implications. The quantization noise is an infinite sum of input signal odd harmonics, where the amplitude of the harmonics is determined by a sum of a Bessel function.

A Bessel function of the first kind looks like this

media/Bessel.svg

So I would expect the amplitude to show signs of oscillatory behavior for the harmonics. That's the important thing to remember. The quantization noise is **odd harmonics of the input signal**

The mean value is zero

$$\overline{e_n(t)} = 0$$

and variance (mean square, since mean is zero), or noise power, can be approximated as

$$\overline{e_n(t)^2} = \frac{\Delta^2}{12}$$

8.2.1 Signal to Quantization noise ratio

Assume we wanted to figure out the resolution, or effective number of bits for an ADC limited by quantization noise. A power ratio, like signal-to-quantization noise ratio (SQNR) is one way to represent resolution.

Take the signal power, and divide by the noise power

$$SQNR = 10\log\left(\frac{A^2/2}{\Delta^2/12}\right) = 10\log\left(\frac{6A^2}{\Delta^2}\right)$$

$$\Delta = \frac{2A}{2^B}$$

$$SQNR = 10\log\left(\frac{6A^2}{4A^2/2^B}\right) = 20B\log 2 + 10\log 6/4$$

$$SQNR \approx 6.02B + 1.76$$

You may have seen the last equation before, now you know where it comes from.

8.2.2 Understanding quantization

Below I've tried to visualize the quantization process q.py.

The left most plot is a sinusoid signal and random Gaussian noise. The signal is not a continuous time signal, since that's not possible on a digital computer, but it's an approximation.

The plots are FFTs of a sinusoidal signal combined with noise. These are complex FFTs, so they show both negative and positive frequencies. The x-axis is the FFT bin (not the frequency). Notice that there are two spikes, which should not be surprising, since a sinusoidal signal is a combination of two frequencies.

$$sin(x) = \frac{e^{ix} - e^{-ix}}{2i}$$

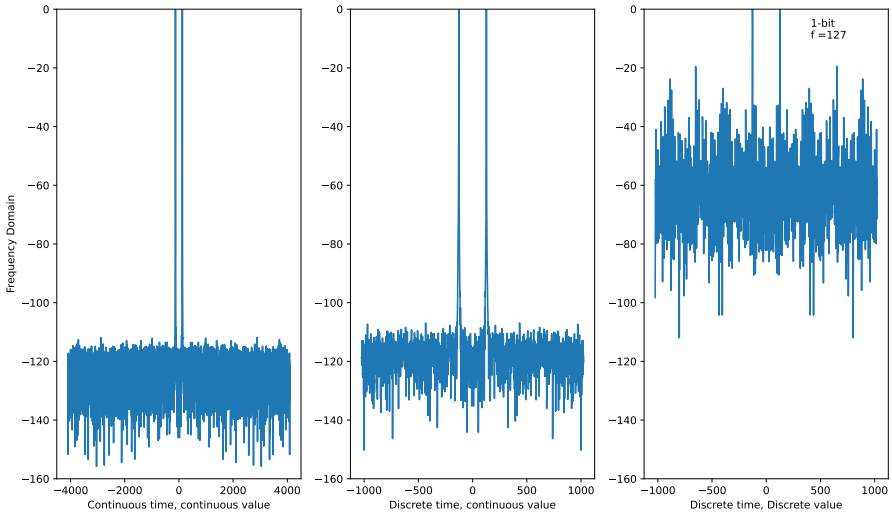
The second plot from the left is after sampling, notice that the noise level increases. The increase in the noise level should be due to noise folding, and reduced number of points in the FFT, but I have not confirmed (maybe you could confirm?).

The right plot is after quantization, where I've used the function below.

```
def adc(x,bits):
    levels = 2**bits
    y = np.round(x*levels)/levels
    return y
```

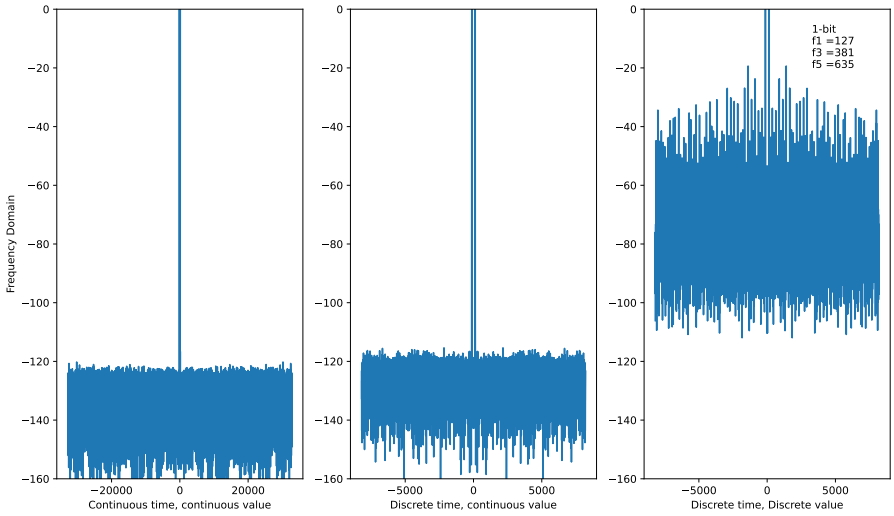
I really need you to internalize a few things from the right most plot. Really think through what I'm about to say.

Can you see how the noise (what is not the two spikes) is not white? White noise would be flat in the frequency domain, but the noise is not flat.



If you run the python script you can zoom in and check the highest spikes. The fundamental is at 127, so odd harmonics would be 381, 635, 889, and from the function of the quantization noise we would expect those to be the highest harmonics (at least when we look at the Bessel function), however, we can see that it's close, but that bin 396 is the highest. Is the math's wrong?

No, the math is correct. Never bet against mathematics. If you change the python script to reduce the frequency, `fdivide=2**9`, and increase number of points, `N=2**16`, as in the plot below, you'll see it's the 11'th harmonic that is highest.



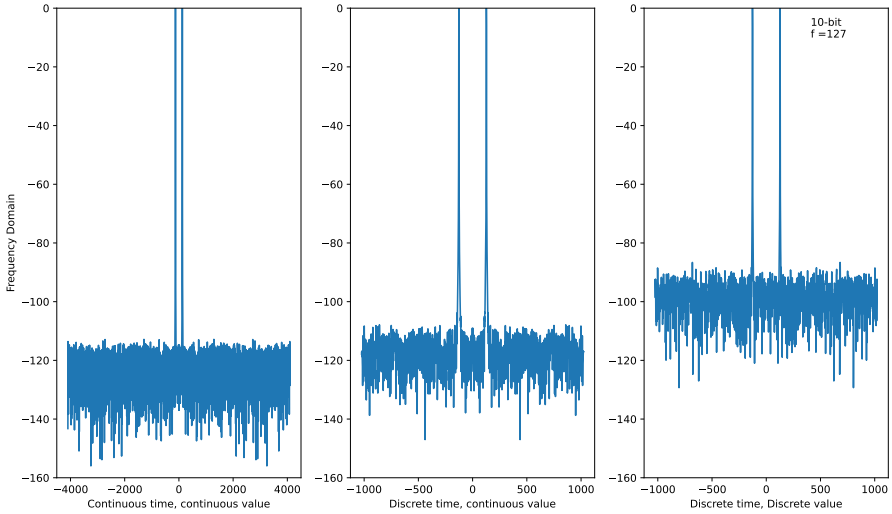
All the other spikes are the odd harmonics above the sample rate that fold. The infinite sum of harmonics will fold, some in-phase, some out of phase, depending on the sign of the Bessel function.

From the function for the amplitude of the quantization noise for harmonic indices higher than $p = 1$

$$A_p = \sum_{m=1}^{\infty} \frac{2}{m\pi} J_p(2m\pi \mathbf{2^{n-1}}), p = odd$$

we can see that the input to the Bessel function increases faster for a higher number of bits n. As such, from the Bessel function figure above, I would expect that the sum of the Bessel function is a lower value. Accordingly, the quantization noise reduces at higher number of bits.

A consequence is that the quantization noise becomes more and more uniform, as can be seen from the plot of a 10-bit quantizer below. That's why people say "Quantization noise is white", because for a high number of bits, it looks white in the FFT.



8.2.3 Why you should care about quantization noise

So why should you care whether the quantization noise looks white, or actually is white? A class of ADCs called oversampling and sigma-delta modulators rely on the assumption that quantization noise **is** white. In other words, the cross-correlation between noise components at different time points is zero. As such the noise power sums as a sum of variance, and we can increase the signal-to-noise ratio.

We know that assumption to be wrong though, **quantization noise is not white**. For noise components at harmonic frequencies the cross-correlation will be high. As such, when **we** design oversampling or sigma-delta based ADC **we** will include some form of dithering (making quantization noise whiter). For example, before the actual quantizer we inject noise, or we make sure that the thermal noise is high enough to dither the quantizer.

Everybody that thinks that quantization noise **is** white will design non-functioning (or sub-optimal) oversampling and sigma-delta ADCs. That's why you should care about the details around quantization noise.

8.3 Oversampling

Assume a signal $x[n] = a[n] + b[n]$ where a is a sampled sinusoid and b is a random process where cross-correlation is zero for any time except for $n = 0$. Assume that we sum two (or more) equally spaced signal components, for example

$$y = x[n] + x[n+1]$$

What would the signal to noise ratio be for y?

8.3.1 Noise power

Our mathematician friends have looked at this, and as long the noise signal b **is random** then the noise power for the oversampled signal $b_{osr} = b[n] + b[n+1]$ will be

$$\overline{b_{osr}^2} = OSR \times \overline{b^2}$$

where OSR is the oversampling ratio. If we sum two time points the $OSR = 2$, if we sum 4 time points the $OSR = 4$ and so on.

For fun, let's go through the mathematics

Define $b_1 = b[n]$ and $b_2 = b[n+1]$ and compute the noise power

$$\overline{(b_1 + b_2)^2} = \overline{b_1^2 + 2b_1b_2 + b_2^2}$$

Let's replace the mean with the actual function

$$\frac{1}{N}\sum_{n=0}^{N}\left(b_1^2 + 2b_1b_2 + b_2^2\right)$$

which can be split up into

$$\frac{1}{N}\sum_{n=0}^{N} b_1^2 + \frac{1}{N}\sum_{n=0}^{N} 2b_1b_2 + \frac{1}{N}\sum_{n=0}^{N} b_2^2$$

we've defined the cross-correlation to be zero, as such

$$\overline{(b_1 + b_2)^2} = \frac{1}{N}\sum_{n=0}^{N} b_1^2 + \frac{1}{N}\sum_{n=0}^{N} b_2^2 = \overline{b_1^2} + \overline{b_2^2}$$

but the noise power of each of the b's must be the same as b, so

$$\overline{(b_1 + b_2)^2} = 2\overline{b^2}$$

8.3.2 Signal power

For the signal a we need to calculate the increase in signal power as OSR increases.

I like to think about it like this. a is low frequency, as such, samples n and $n+1$ is pretty much the same value. If the sinusoid has an amplitude of 1, then the amplitude would be 2 if we sum two samples. As such, the amplitude must increase with the OSR.

The signal power of a sinusoid is $A^2/2$, accordingly, the signal power of an oversampled signal must be $(OSR \times A)^2/2$.

8.3.3 Signal to Noise Ratio

Take the signal power to the noise power

$$\frac{(OSR \times A)^2/2}{OSR \times \overline{b^2}} = OSR \times \frac{A^2/2}{\overline{b^2}}$$

We can see that the signal to noise ratio increases with increased oversampling ratio, **as long as the cross-correlation of the noise is zero**

8.3.4 Signal to Quantization Noise Ratio

The in-band quantization noise for a oversampling ratio (OSR)

$$\overline{e_n(t)^2} = \frac{\Delta^2}{12OSR}$$

And the improvement in SQNR can be calculated as

$$SQNR = 10\log\left(\frac{6A^2}{\Delta^2/OSR}\right) = 10\log\left(\frac{6A^2}{\Delta^2}\right) + 10\log(OSR)$$

$$SQNR \approx 6.02B + 1.76 + 10\log(OSR)$$

For an OSR of 2 and 4 the SQNR improves by

$$10\log(2) \approx 3dB$$

and for OSR=4

$$10\log(4) \approx 6dB$$

which is roughly equivalent to a 0.5-bit per doubling of OSR

8.3.5 Python oversample

There are probably more elegant (and faster) ways of implementing oversampling in python, but I like to write the dumbest code I can, simply because dumb code is easy to understand.

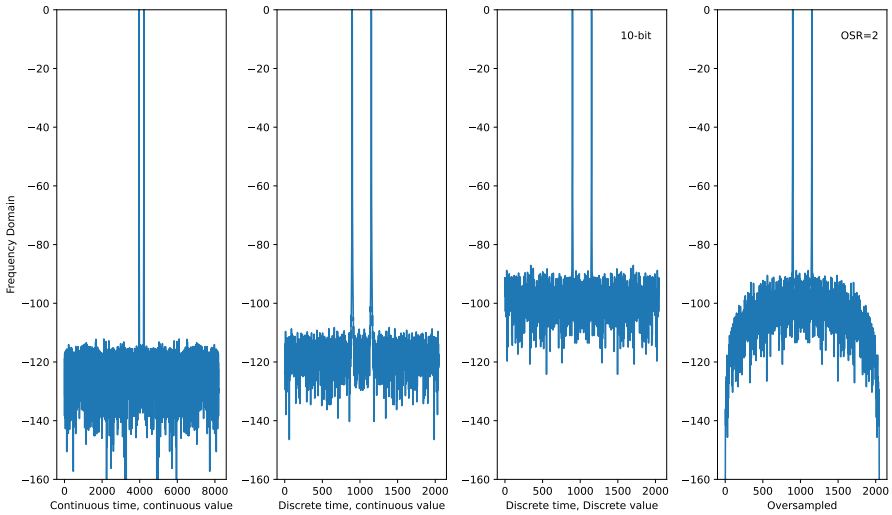
Below you can see an example of oversampling. The `oversample` function takes in a vector and the OSR. For each index it sums OSR future values.

```
def oversample(x,OSR):
    N = len(x)
    y = np.zeros(N)

    for n in range(0,N):
        for k in range(0,OSR):
            m = n+k
```

```
            if (m < N):
                y[n] += x[m]
    return y
```

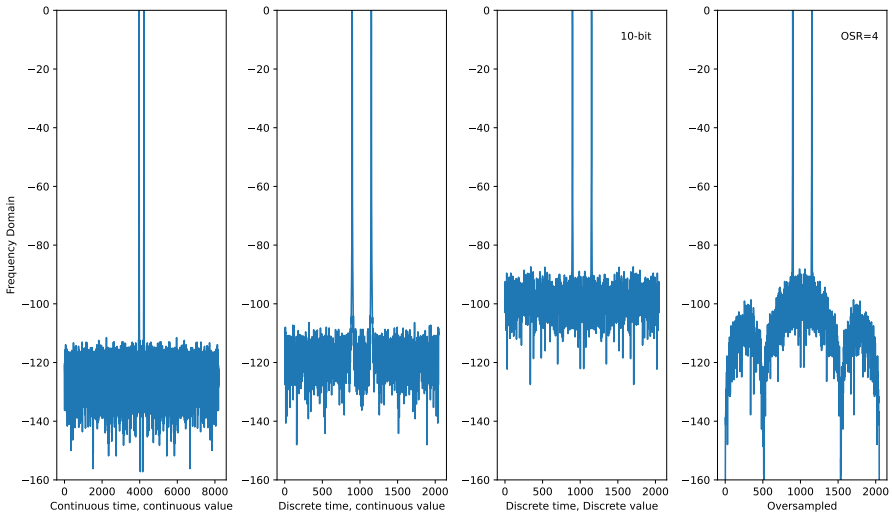
Below we can see the plot for OSR=2, the right most plot is the oversampled version.

The noise has all frequencies, and it's the high frequency components that start to cancel each other. An average filter (sometimes called a sinc filter due to the shape in the frequency domain) will have zeros at $\pm fs/2$ where the noise power tends towards zero.



The low frequency components will add, and we can notice how the noise power increases close to the zero frequency (middle of the x-axis).

For an OSR of 4 we can notice how the noise floor has 4 zero's.



The code for the plots is osr.py. I would encourage you to play a bit with the code, and make sure you understand oversampling.

8.4 Noise Shaping

Look at the OSR=4 plot above. The OSR=4 does decrease the noise compared to the discrete time discrete value plot, however, the noise level of the discrete time continuous value is much lower.

What if we could do something, add some circuitry, before the quantization such that the quantization noise was reduced?

That's what noise shaping is all about. Adding circuits such that we can "shape" the quantization noise. We can't make the quantization noise disappear, or indeed reduce the total noise power of the quantization noise, but we can reduce the quantization noise power for a certain frequency band.

But what circuitry can we add?

8.4.1 The magic of feedback

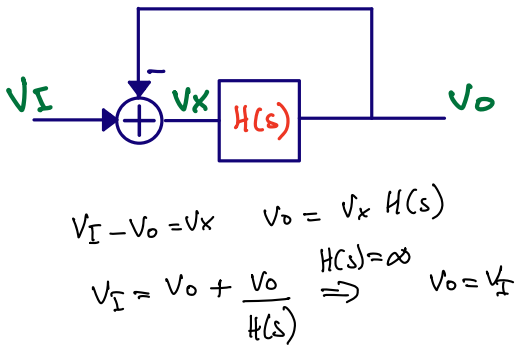
A generalized feedback system is shown below, it could be a regulator, a unity-gain buffer, or something else.

The output V_o is subtracted from the input V_i, and the error V_x is shaped by a filter $H(s)$.

If we make $H(s)$ infinite, then $V_o = V_i$. If you've never seen such a circuit, you might ask "Why would we do this? Could we not just use V_i directly?". There are many reasons for using a circuit like this, let me explain one instance.

Imagine we have a VDD of 1.8 V, and we want to make a 0.9 V voltage for a CPU. The CPU can consume up to 10 mA. One way to make a divide by two circuit is with two equal resistors connected between VDD and ground. We don't want the resistive divider to consume a large current, so let's choose 1 MOhm resistors. The current in the resistor divider would then be about 1 μA. We can't connect the CPU directly to the resistor divider, the CPU can draw 10 mA. As such, we need a copy of the voltage at the mid-point of the resistor divider that can drive 10 mA.

Do you see now why a circuit like the one below is useful? If not, you should really come talk to me so I can help you understand.

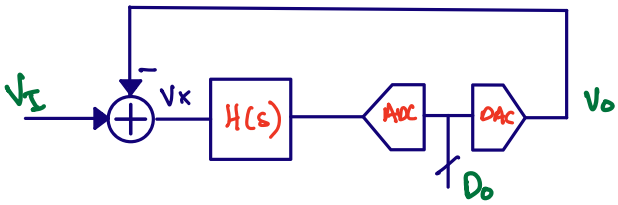


8.4.2 Sigma-delta principle

Let's modify the feedback circuit into the one below. I've added an ADC and a DAC to the feedback loop, and the D_o is now the output we're interested in. The equation for the loop would be

$$D_o = adc\left[H(s)\left(dac(D_o) - V_i\right)\right]$$

But how can we now calculate the transfer function $\frac{D_o}{V_i}$? Both *adc* and *dac* could be non-linear functions, so we can't disentangle the equation. Let's make assumptions.



8.4.2.1 The DAC assumption

Assumption 1: the *dac* is linear, such that $V_o = dac(D_o) = AD_o + B$, where A and B are scalar values.

The DAC must be linear, otherwise our noise-shaping ADC will not work.

One way to force linearity is to use a 1-bit DAC, which has only two points, so should be linear. For example

$$V_o = A \times D_o$$

, where $D_o \in (0, 1)$. Even a 1-bit DAC could be non-linear if A is time-variant, so $V_o[n] = A(t) \times D_o[n]$, this could happen if the reference voltage for the DAC changed with time.

I've made a couple noise shaping ADCs, and in the first one I made I screwed up the DAC. It turned out that the DAC current had a signal dependent component which lead to a non-linear behavior.

8.4.2.2 The ADC assumption

Assumption 2: the *adc* can be modeled as a linear function $D_o = adc(x) = x + e$, where e is **white noise source**

We've talked about this, the e is not white, especially for low-bit ADCs, so we usually have to add noise. Sometimes it's sufficient with thermal noise, but often it's necessary to add a random, or pseudo-random noise source at the input of the ADC.

8.4.2.3 The modified equation

With the assumptions we can change the equation into

$$D_o = adc\left[H(s)\left(V_i - dac(D_o)\right)\right] = H(s)\left(V_i - AD_o\right) + e$$

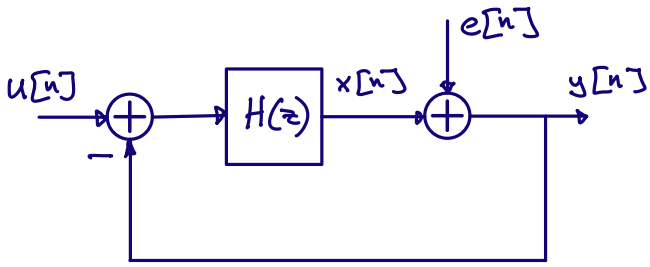
In noise-shaping texts it's common to write the above equation as

$$y = H(s)(u - y) + e$$

or in the sample domain

$$y[n] = e[n] + h * (u[n] - y[n])$$

which could be drawn in a signal flow graph as below.



in the Z-domain the equation would turn into

$$Y(z) = E(z) + H(z)\left[U(z) - Y(z)\right]$$

The whole point of this exercise was to somehow shape the quantization noise, and we're almost at the point, but to show how it works we need to look at the transfer function for the signal U and for the noise E.

8.4.3 Signal transfer function

Assume U and E are uncorrelated, and E is zero

$$Y = HU - HY$$

$$STF = \frac{Y}{U} = \frac{H}{1+H} = \frac{1}{1+\frac{1}{H}}$$

Imagine what will happen if H is infinite. Then the signal transfer function (STF) is 1, and the output Y is equal to our input U. That's exactly what we wanted from the feedback circuit.

8.4.4 Noise transfer function

Assume U is zero

$$Y = E + HY \rightarrow NTF = \frac{1}{1+H}$$

Imagine again what happens when H is infinite. In this case the noise-transfer function becomes zero. In other words, there is no added noise.

8.4.5 Combined transfer function

In the combined transfer function below, if we make $H(z)$ infinite, then $Y = U$ and there is **no added quantization noise**. I don't know how to make $H(z)$ infinite everywhere, so we have to choose at what frequencies it's "infinite".

$$Y(z) = STF(z)U(z) + NTF(z)E(z)$$

There are a large set of different $H(z)$ and I'm sure engineers will invent new ones. We usually classify the filters based on the number of zeros in the NTF, for example, first-order (one zero), second order (two zeros) etc. There are books written about sigma-delta modulators, and I would encourage you to read those to get a deeper understanding. I would start with Delta-Sigma Data Converters: Theory, Design, and Simulation.

8.5 First-Order Noise-Shaping

We want an infinite $H(z)$. One way to get an infinite function is an accumulator, for example

$$y[n+1] = x[n] + y[n]$$

or in the Z-domain

$$zY = X + Y \rightarrow Y(z-1) = X$$

which has the transfer function

$$H(z) = \frac{1}{z-1}$$

The signal transfer function is

$$STF = \frac{1/(z-1)}{1+1/(z-1)} = \frac{1}{z} = z^{-1}$$

and the noise transfer function

$$NFT = \frac{1}{1+1/(z-1)} = \frac{z-1}{z} = 1 - z^{-1}$$

In order calculate the Signal to Quantization Noise Ratio we need to have an expression for how the NTF above filters the quantization noise.

In the book they replace the z with the continuous time variable

$$z = e^{sT} \overset{s=j\omega}{\rightarrow} e^{j\omega T} = e^{j2\pi f/f_s}$$

inserted into the NTF we get the function below.

$$NTF(f) = 1 - e^{-j2\pi f/f_s}$$

$$= \frac{e^{j\pi f/f_s} - e^{-j\pi f/f_s}}{2j} \times 2j \times e^{-j\pi f/f_s}$$

$$= \sin\frac{\pi f}{f_s} \times 2j \times e^{-j\pi f/f_s}$$

The arithmetic magic is really to extract the $2j \times e^{-j\pi f/f_s}$ from the first expression such that the initial part can be translated into a sinusoid.

When we take the absolute value to figure out how the NTF changes with frequency the complex parts disappears (equal to 1)

$$|NFT(f)| = \left|2\sin\left(\frac{\pi f}{f_s}\right)\right|$$

The signal power for a sinusoid is

$$P_s = A^2/2$$

The in-band noise power for the shaped quantization noise is

$$P_n = \int_{-f_0}^{f_0} \frac{\Delta^2}{12}\frac{1}{f_s}\left[2\sin\left(\frac{\pi f}{f_s}\right)\right]^2 dt$$

and with a bunch of tedious maths, we can get to the conclusion

$$\vdots$$

$$SQNR = 6.02B + 1.76 - 5.17 + 30\log(OSR)$$

If we compare to pure oversampling, where the SQNR improves by $10\log(OSR)$, a first order sigma-delta improves by $30\log(OSR)$. That's a significant improvement.

Assume 1-bit quantizer, what would be the maximum ENOB?

OSR	Oversampling	First-Order	Second Order
4	2	3.1	3.9
64	4	9.1	13.9
1024	6	15.1	23.9

The table above shows the effective number of bits for oversampling, and sigma-delta modulators. For a 1-bit quantizer, pure oversampling does not make sense at all. For first-order and second-order sigma delta modulators, and a OSR of 1024 we can get high resolution ADCs.

8.5.1 Python noise-shaping

I want to demystify noise-shaping modulators. I think one way to do that is to show some code. You can find the code at sd_1st.py

Below we can see an excerpt. Again pretty stupid code, and I'm sure it's possible to make a faster version (for loops in python are notoriously slow).

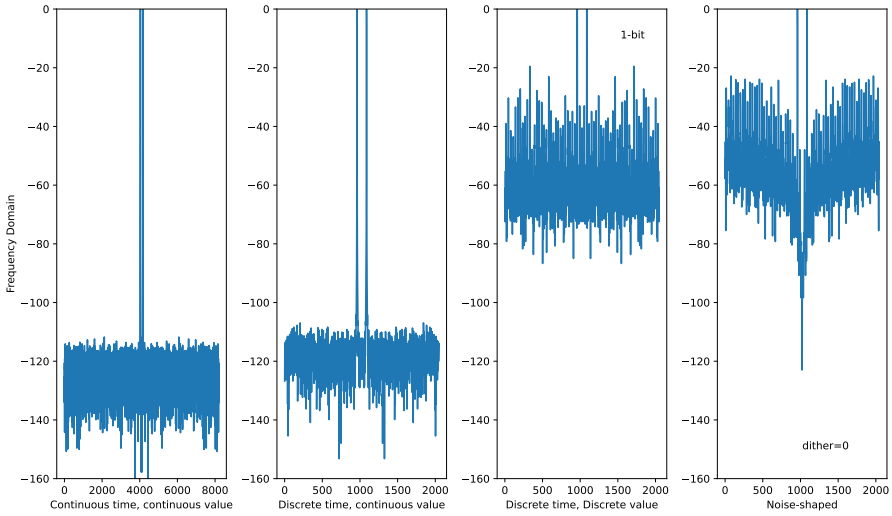
For each sample in the input vector x_{sn} I compute the input to the quantizer x, which is the sum of the previous input to the quantizer and the difference between the current input and the previous output y_{sd}.

The quantizer generates the next y_{sd} and I have the option to add dither.

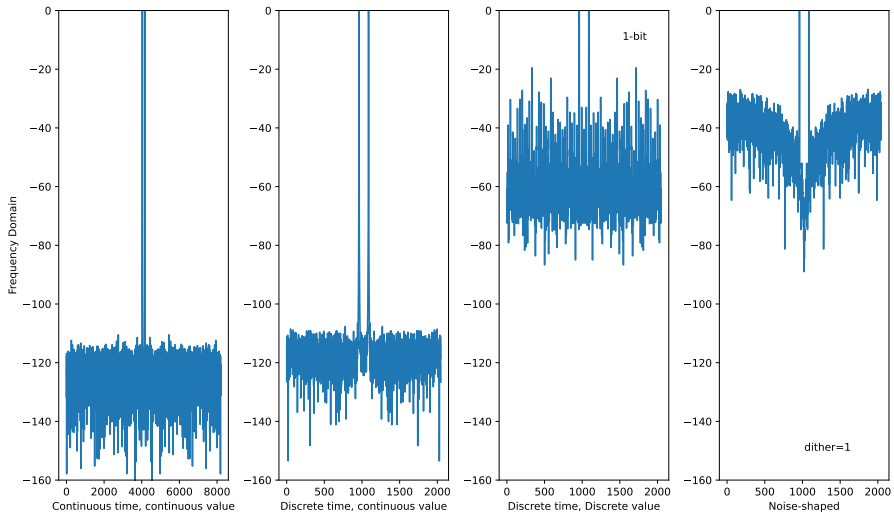
```
# x_sn is discrete time, continuous value input
dither = 0
M = len(x_sn)
y_sd = np.zeros(M)
x = np.zeros(M)
for n in range(1,M):
    x[n] = x[n-1] + (x_sn[n]-y_sd[n-1])
    y_sd[n] = np.round(x[n]*2**bits
        + dither*np.random.randn()/4)/2**bits
```

The right-most plot is the one with noise-shaping. We can observe that the noise seems to tend towards zero at zero frequency, as we would expect. The accumulator above would have an infinite gain at infinite time (it's the sum of all previous values), as such, the NTF goes towards zero.

If we look at the noise we can also see the non-white quantization noise, which will degrade our performance, which I hope by now, you've grown tired of me harping on the point that **quantization noise is not white**



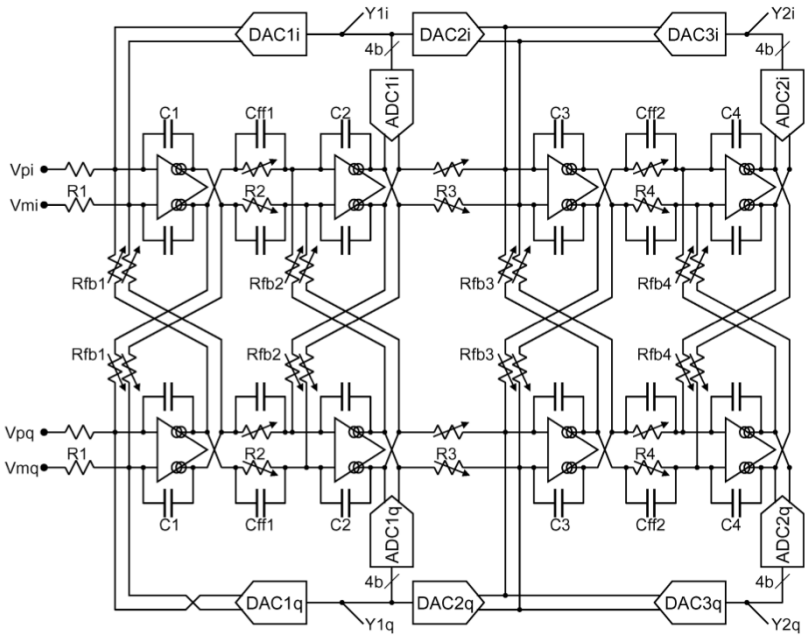
In the figure below I've turned on dither, and we can see how the noise looks "better", which I know is not a qualitative statement, but ask anyone that's done 1-bit quantizers. It's important to have enough random noise.



8.5.2 Sigma-Delta examples

There are cool sigma-delta modulators with crazy configurations and that may look like an exercise in "Let's make something complex", however, most of them have a reasonable application. One example is the one below for radio recievers

A 56 mW Continuous-Time Quadrature Cascaded Sigma-Delta Modulator With 77 dB DR in a Near Zero-IF 20 MHz Band



igma-delta modulator design.

8.5.3 My first Sigma-Delta

The first sigma-delta modulator I made in "real-life" was for the nRF51. The ADC was similar to the one shown below.

The input voltage is translated into a current, and the current is integrated on capacitor C. The R_{offset}

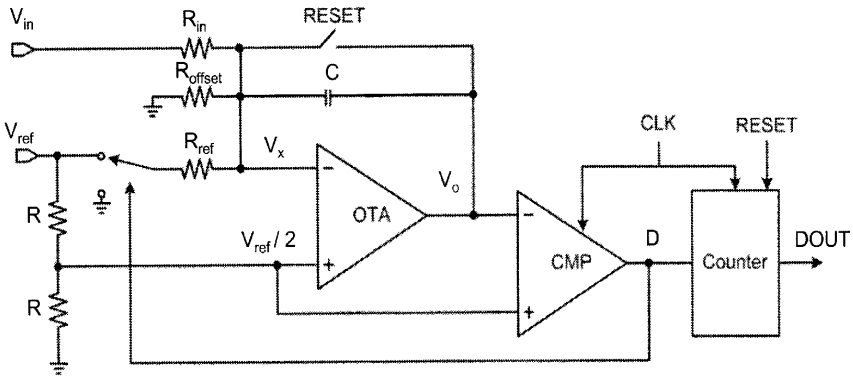
is to change the mid-level voltage, while R_{ref} is the 1-bit feedback DAC. The comparator is the quantizer. When the clock strikes the comparator compares the V_o and $V_{ref}/2$ and outputs a 1-bit digital output D

The complete ADC is operated in a "incremental mode", which is a fancy way of saying

> Reset your sigma-delta modulator, run the sigma delta modulator for a fixed number of cycles (i.e 1024), and count the number of ones at D

The effect of an "incremental mode" is to combine the modulator and a output filter so the ADC appears to be a slow Nyquist ADC.

For more information, ask me, or see the patent at Analogue-to-digital converter



Chapter 9

Voltage Regulation

9.1 Voltage source

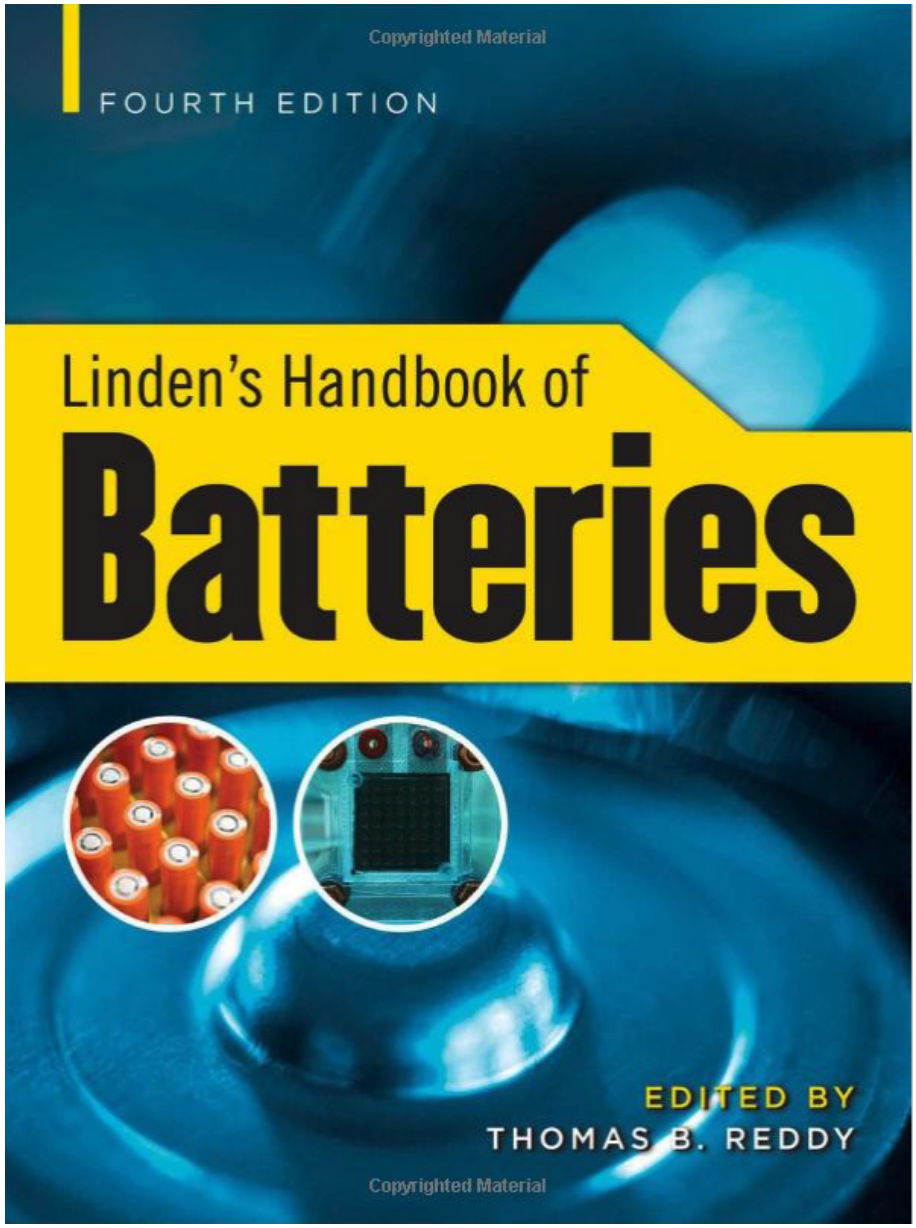
Most, if not all, integrated circuits need a supply and ground to work.

Assume a system is AC powered. Then there will be switched regulator to turn wall AC into DC. The DC might be 48 V, 24 V, 12 V, 5 V, 3 V 1.8 V, 1.0 V, 0.8 V, or who knows. The voltage depends on the type of IC and the application.

Many ICs are battery operated, whether it's your phone, watch, heart rate monitor, mouse, keyboard, game controller or car.

For batteries the voltage is determined by the difference in Fermi level on the two electrodes, and the Fermi level (chemical potential) is a function of the battery chemistry. As a result, we need to know the battery chemistry in order to know the voltage.

Linden's Handbook of Batteries is a good book if you want to dive deep into primary (non-chargeable) or secondary (chargeable) batteries and their voltage curves.



Some common voltage sources are listed below.

	Chemistry	Voltage [V]
Primary Cell	LiFeS2 + Zn/Alk/MnO2 + LiMnO2	0.8 - 3.6
Secondary Cell	Li-Ion	2.5 - 4.3
USB	-	4.0 - 6.5 (20)

The battery determines the voltage of the "electron source", however, can't we just run everything directly off the battery? Why do we need DC to DC converters or voltage regulators?

Turns out, transistors can die.

9.1.1 Core voltage

The transistors in a particular technology (from GlobalFoundries, TSMC, Samsung or others) have a maximum voltage that they can survive for a certain time. Exceed that time, or voltage, and the transistors die.

9.1.1.1 Why transistors die

A gate oxide will break due to Time Dependent Dielectric Breakdown (TDDB) if the voltage across the gate oxide is too large. Silicon oxide can break down at approximately 5 MV/cm. The breakdown forms a conductive channel from the gate to the channel and is permanent. After breakdown there will be a resistor of kOhms between gate and channel. A similar breakdown phenomena is used in Metal-Oxide RRAM.

The threshold voltage of a transistor can shift excessively over time caused by Hot-Carrier Injection (HCI) or Negative Bias Temperature Instability.

Hot-Carrier injection is caused by electrons, or holes, accelerated to high velocity in the channel, or drain depletion region , causing impact ionization (breaking a co-valent bond releasing an electron/hole pair). At a high drain/source field, and
medium gate/(source or drain) field, the channel minority carriers can be accelerated to high energy and transition to traps in the oxide, shifting the threshold voltage.

Negative Bias Temperature Instability is a shift in threshold voltage due to a physical change in the oxide. A strong electric field across the oxide for a long time can break co-valent, or ionic bonds, in the oxide. The bond break will change the forces (stress) in the amorphous silicon oxide which might not recover. As such, there might be more traps (states) than before. See Simultaneous Extraction of Recoverable and Permanent Components Contributing to Bias-Temperature Instability for more details.

9.1.1.2 What is traps?

For a long time, I had trouble with “traps in the oxide”“. I had a hard time visualizing how electrons wandered down the channel and got caught in the oxide. I was trying to imagine the electric field, and that the electron needed to find a positive charge in the oxide to cancel. Diving a bit deeper into quantum mechanics, my mental image improved a bit, so I’ll try to give you a more accurate mental model for how to think about traps.

Quantum mechanics tells us that bound electrons can only occupy fixed states. The probability of finding an electron in a state is given by Fermi-Dirac statistics, but if there is no energy state at a point in space, there cannot be an electron there.

For example, there might be a 50 % probability of finding an electron in the oxide, but if there is no state there, then there will not be any electron , and thus no change to the threshold voltage.

What happens when we make “traps”, through TDDB, HCI, or NBTI is that we create new states that can potentially be occupied by electorns. For example one, or more, broken silicon co-valent bonds and a dislocation of the crystal lattice.

If the Fermi-Dirac statistics tells us the probability of an electron being in those new states is 50 %, then there will likely be electrons there.

The threshold voltage is defined as the voltage at which we can invert the channel, or create the same density of electrons in the channel (for NMOS) as density of dopant atoms (density of holes) in the bulk.

If the oxide has a net negative charge (because of electrons in new states), then we have to pull harder (higher gate voltage) to establish the channel. As a result, the threshold voltage increases with electrons stuck in the oxide.

In quantum mechanics the time evolution, and the complex probability amplitude of an electron changing state, could, in theory, be computed with the Schrodinger equation. Unfortunately, for any real scenario, like the gate oxide of a transistor, using Schrodinger to compute exactly what will happen is beyond the capability of the largest supercomputers.

9.1.1.3 Breakdown voltages

The voltage where the transistor can survive is estimated by the foundry, by approximation, and testing, and may be like the table below.

Node [nm]	Voltage [V]
180	1.8
130	1.5
55	1.2
22	0.8

9.1.2 IO voltage

Most ICs talk to other ICs, and they have a voltage for the general purpose input/output. The voltage reduction in I/O voltage does not need to scale as fast as the core voltage, because foundries have thicker oxide transistors that can survive the voltage.

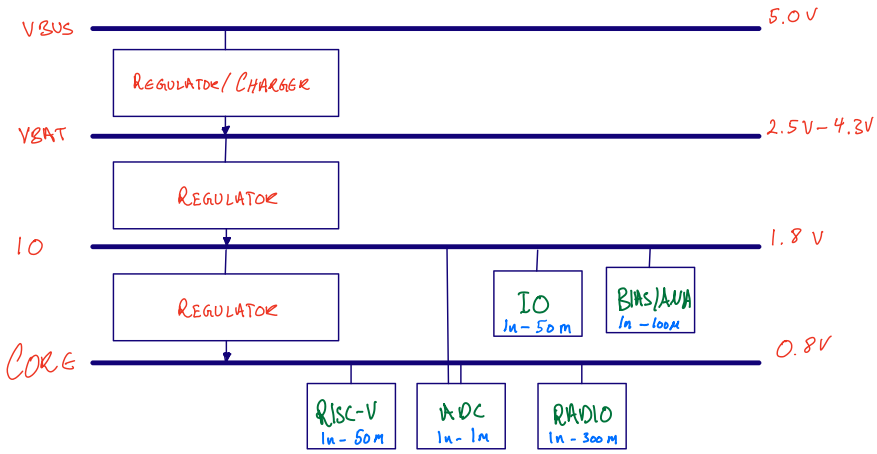
Voltage [V]
5.0
3.0
1.8
1.2

9.1.3 Supply planning

For any IC, we must know the application. We must know where the voltage comes from, the IO voltage, the core voltage, and any other requirements (like charging batteries).

One example could be an IC that is powered from a Li-Ion battery, with a USB to provide charging capability.

Between each voltage we need an analog block, a regulator, to reduce the voltage in an effective manner. What type of regulator depends again on the application, but the architecture of the analog design would be either a linear regulator, or a switched regulator.



The dynamic range of the power consumed by an IC can be large. From nA when it's not doing anything, to hundreds of mA when there is high computation load.

As a result, it's not necessarily possible, or effective, to have one regulator from 1.8 V to 0.8 V. We may need multiple regulators. Some that can handle low load (nA - μA) effectively, and some that can handle high loads.

For example, if you design a regulator to deliver 500 mA to the load, and the regulator uses 5 mA, that's only 1 % of the current, which may be OK. The same regulator might consume 5 mA even though the load is 1 uA, which would be bad. All the current flows in the regulator at low loads.

Name	Voltage	Min [nA]	Max [mA]	PWR DR [dB]
VDD_VBUS	5	10	500	77
VDD_VBAT	4	10	400	76
VDD_IO	1.8	10	50	67
VDD_CORE	0.8	10	350	75

Most product specifications will give you a view into what type of regulators there are on an IC. The picture below is from nRF5340

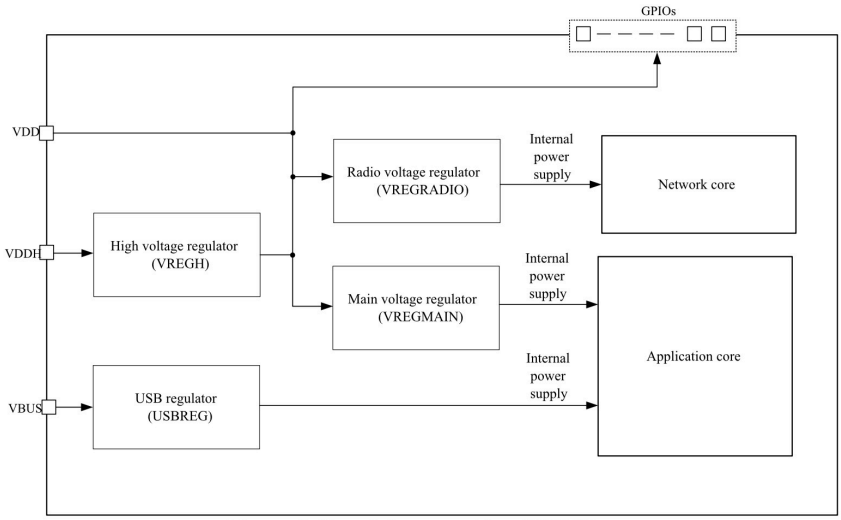


Figure 1. Regulators used in nRF5340

9.2 Linear Regulators

9.2.1 PMOS pass-fet

One way to make a regulator is to control the current in a PMOS with a feedback loop, as shown below. The OTA continuously adjusts the gate-source voltage of the PMOS to force the input voltages of the OTA to be equal.

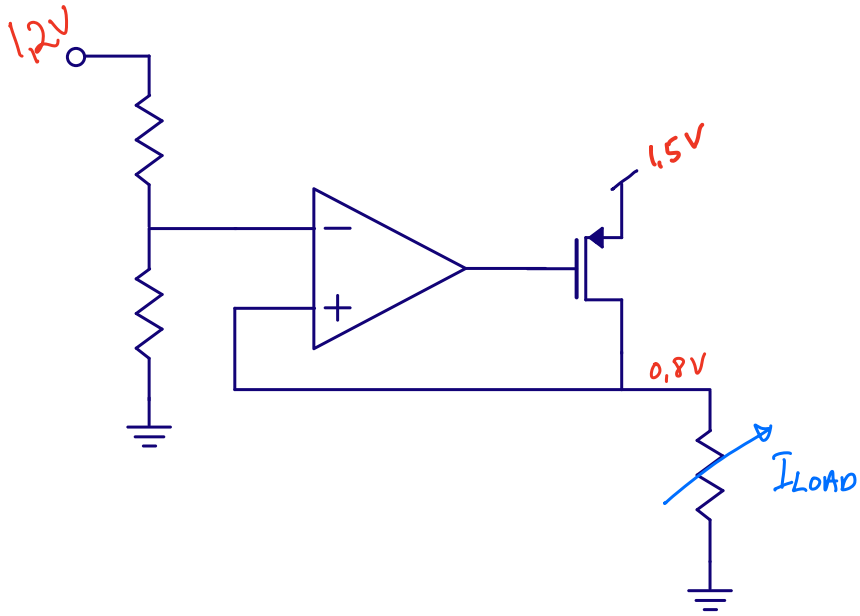
For digital loads, where I_{load} is a digital current, with high current every rising edge of the clock, it's an option to place a large external decoupling capacitor (a reservoir of charge) in parallel with the load. Accordingly, the OTA would supply the average current.

The device between supply (1.5 V) and output voltage (0.8 V) is often called a pass-fet. A PMOS pass-fet regulator is often called a LDO, or low dropout regulator, since we only need a V_{DSSAT} across the PMOS, which can be a few hundred mV.

Key parameters of regulators are

Parameter	Description	Unit
Load regulation	How much does the output voltage change with load current	V/A
Line regulation	How much does the output voltage change with input voltage	V/V
Power supply rejection ratio	What is the transfer function from input voltage to output voltage? The PSRR at DC is the line regulation	dB
Max current	How much current can be delivered through the pass-fet?	A
Quiescent current	What is the current used by the regulator	A
Settling time	How fast does the output voltage settle at a current step	s

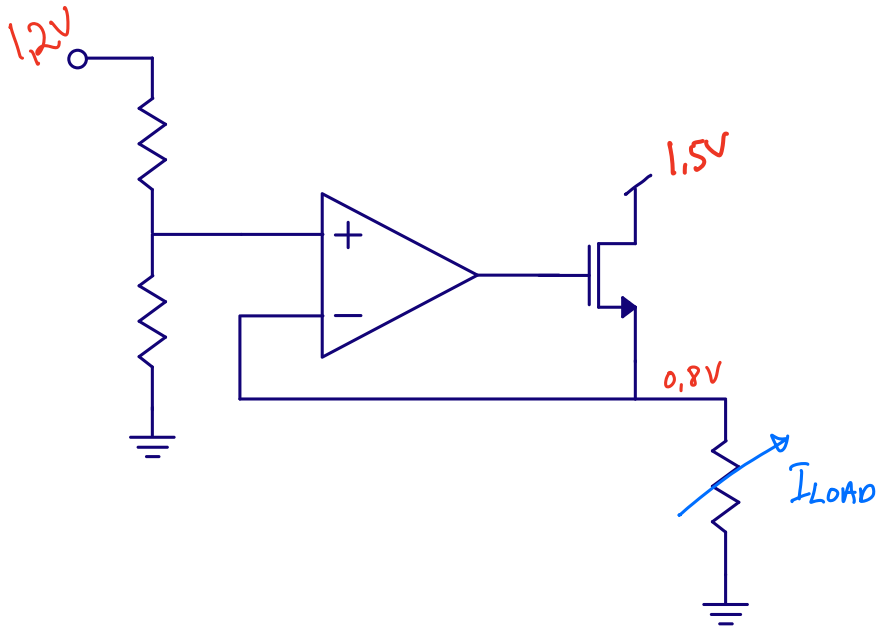
A disadvantage of a PMOS is the hole mobility, which is lower than for NMOS. If the maximum current of an LDO is large, then the PMOS can be big. Maybe even 50 % of the IC area.



9.2.2 NMOS pass-fet

An NMOS pass-fet will be smaller than a PMOS for large loads. The disadvantage with an NMOS is the gate-source voltage needed. For some scenarios the needed gate voltage might exceed the input voltage (1.5 V). A gate voltage above input voltage is possible, but increases complexity, as a charge pump (switched capacitor regulator) is needed to make the gate voltage.

Another interesting phenomena with NMOS pass-fet is that the PSRR is usually better, but we do have a common gate amplifier, as such, high frequency voltage ripple on output voltage will be amplified to the input voltage, and may cause issues for others using the input voltage.



9.3 Switched Regulators

Linear regulator have poor power efficiency. Linear regulators have the same current in the load, as from the input.

For some applications a poor efficiency might be OK, but for most battery operated systems we're interested in using the electrons from the battery in the most effective manner.

Another challenge is temperature. A linear regulator with a 5 V input voltage, and 1 V output voltage will have a maximum power efficiency of 20 % (1/5). 80 % of the power is wasted in the pass-fet as heat.

Imagine a LDO driving an 80 W CPU at 1 V from a 5 V power supply. The power drawn from the 5 V supply is 400 W, as such, 320 W would be wasted in the LDO. A quad flat no-leads (QFN) package usually have a thermal resistance of 20 °C/W, so if it would be possible, the temperature of the LDO would be 6400 °C. Obviously, that cannot work.

For increased power efficiency, we must use switched regulators.

Imagine a switch regulator with 93 % power efficiency. The power from the 5 V supply would be 80 W/0.93 = 86 W, as such, only 6 W is wasted as heat. A temperature increase of 6 W × 20 °C/W = 120°C is still high, but not impossible with a small heat-sink.

All switched regulators are based on devices that store electric field (capacitors), or magnetic field (inductors).

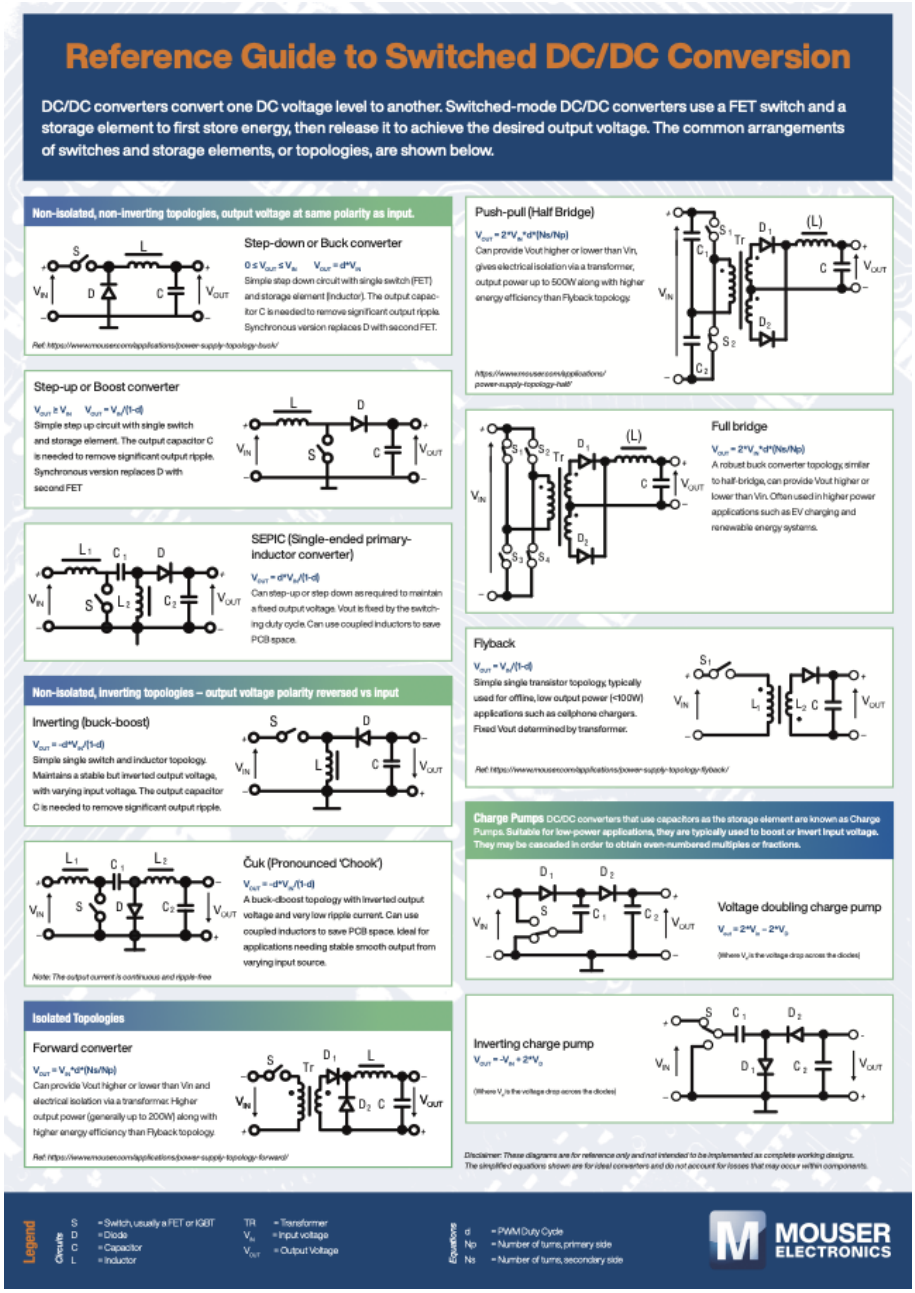
9.3.1 Types of switched-regulators

Mouser has a nice overview of different type of switched regulators. I would encourage you to download the PDF from Mouser.

A capacitor can be charged to a voltage. Once charged, one can reconfigure a capacitor circuit, for

example changing the capacitor circuit from series to parallel (convert down in voltage), or parallel to series (convert up in voltage). See "Charge pumps" in figure below.

An inductor, once charged with a current, will continue to push the current even if we change the voltage across the inductor terminals. As such, we can redirect current, either to charge a capacitor to a higher voltage ("Step-up", or "Boost" in figure below), or a lower voltage ("Step-down" or "Buck" in figure below), than the input voltage.



Reference Guide to Switched DC/DC Conversion

9.3.2 Inductive DC/DC converters

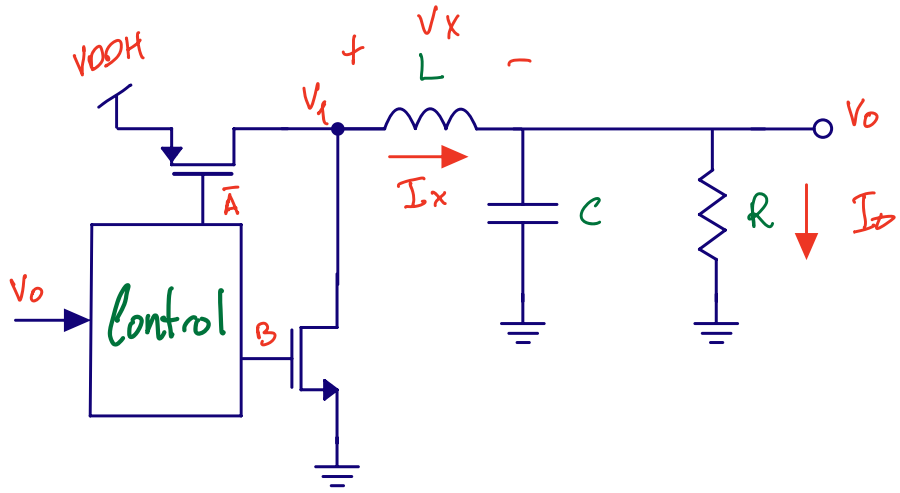
I've found that people struggle with inductive DC/DCs. They see a circuit inductors, capacitors, and transistors and think filters, Laplace and steady state. The path of Laplace and steady state will lead you

astray and you won't understand how it works.

Hopefully I can put you on the right path to understanding.

In the figure below we can see a typical inductive switch mode DC/DC converter. The input voltage is V_{DDH}, and the output is V_O.

Most DC/DCs are feedback systems, so the control will be adjusted to force the output to be what is wanted, however, let's ignore closed loop for now.



To see what happens I find the best path to understanding is to look at the integral equations.

The current in the inductor is given by

$$I_x(t) = \frac{1}{L}\int V_x(t)dt$$

and the voltage on the capacitor is given by

$$V_o(t) = \frac{1}{C}\int (I_x(t) - I_o(t))dt$$

Before you dive into Matlab, Mathcad, Maple, SymPy or another of your favorite math software, it helps to think a bit.

My mathematics is not great, but I don't think there is any closed form solution to the output voltage of the DC/DC, especially since the state of the NMOS and PMOS is time-dependent.

The output voltage also affect the voltage across the inductor, which affects the current, which affects the output voltage, etc, etc.

The equations can be solved numerically, but a numerical solution to the above integrals needs initial conditions.

There are many versions of the control block, let's look at two.

9.3.3 Pulse width modulation (PWM)

Assume $I_x = 0$ and $I_o = 0$ at $t = 0$. Assume the output voltage is $V_O = 0$. Imagine we set $A = 1$ for a fixed time duration. The voltage at $V_1 = V_{DDH}$, and $V_x = V_{VDDH} - V_O$. As V_x is positive, and roughly constant, the current I_x would increase linearly, as given by the equation of the current above.

Since the I_x is linear, then the increase in V_o would be a second order, as given by the equation of the output voltage above.

Let's set $A = 0$ and $B = 1$ for fixed time duration (it does not need to be the same as duration as we set $A = 1$). The voltage across the inductor would be $V_x = 0 - V_o$. The output voltage would not have increased much, so the absolute value of V_x during $A = 1$ would be higher than the absolute value of V_x during the first $B = 1$.

The V_x is now negative, so the current will decrease, however, since V_x is small, it does not decrease much.

I've made a

Jupyter PWM BUCK model

that numerically solves the equations.

In the figure below we can see how the current during A increases fast, while during B it decreases little. The output voltage increases similarly to a second order function.

media/l07$_b$uck$_p$wm$_f$ig$_s$tart.svg

If we run the simulation longer, see plot below, the DC/DC will start to settle into a steady state condition.

On the left we can see the current I_x and I_o, on the right you can see the output voltage. Turns out that the output voltage will be

$$V_o = V_{in} \times \text{ Duty-Cycle}$$

, where the duty-cycle is the ratio between the duration of $A = 1$ and $B = 1$.

media/l07$_b$uck$_p$wm$_f$ig.svg

Once the system has fully settled, see figure below, we can see the reason for why DC/DC converters are useful.

During $A = 1$ the current I_x increases fast, and it's only during $A = 1$ we pull current from V_{DDH}. At the start of $A = 0$ the current is still positive, which means we pull current from ground. The average current in the inductor is the same as the average current in the load, however, the current from V_{DDH} is lower than the average inductor current, since some of the current comes from ground.

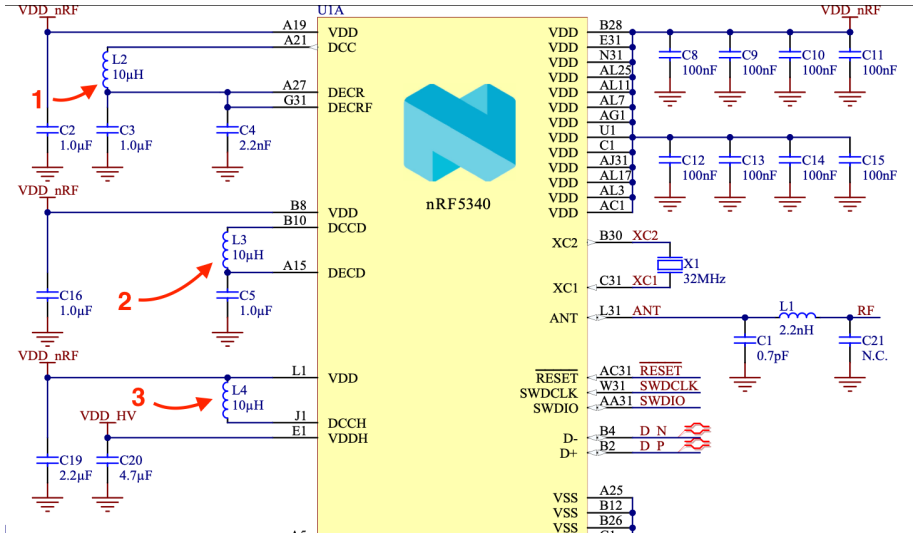
If the DC/DC was 100% efficient, then the current from the 4 V input supply would be 1/4'th of the 1 V output supply. 100% efficient DC/DC converters violate the laws of nature, as such, we can expect to get up to 9X% under optimal conditions.

media/l07$_b$uck$_p$wm$_f$ig$_s$ettled.svg

9.3.4 Real world use

DC/DC converters are used when power efficiency is important. Below is a screenshot of the hardware description in the nRF5340 Product Specification.

We can see 3 inductor/capacitor pairs. One for the “VDDH”, and two for “DECRF” and “DECD”, as such, we can make a good guess there are three DC/DC converters inside the nRF5340.



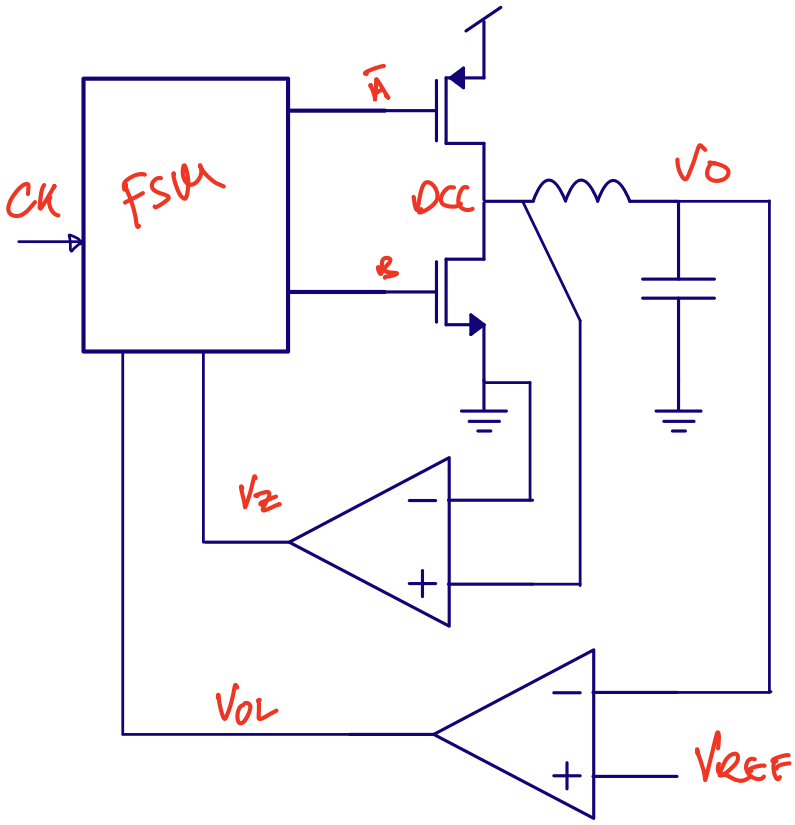
9.3.5 Pulsed Frequency Mode (PFM)

Power efficiency is key in DC/DC converters. For high loads, PWM, as explained above, is usually the most efficient and practical.

For lighter loads, other configurations can be more efficient.

In PWM we continuously switch the NMOS and PMOS, as such, the parasitic capacitance on the V_1 node is charged and discharged, consuming power. If the load is close to 0 A, then the parasitic load’s can be significant.

In pulsed-frequency mode we switch the NMOS and PMOS when it’s needed. If there is no load, there is no switching, and V_1 or DCC in figure below is high impedant.



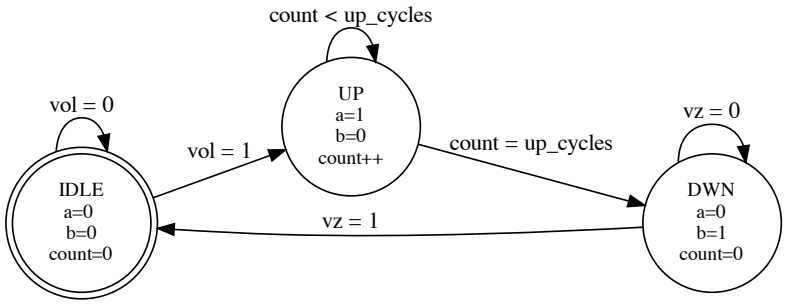
Imagine V_o is at 1 V, and we apply a constant output load. According to the integral equations the V_o would decrease linearly.

In the figure above we observe V_o with a comparator that sets V_{OL} high if the $V_o < V_{REF}$. The output from the comparator could be the inputs to a finite state machine (FSM).

Consider the FSM below. On $vol = 1$ we transition to "UP" state where turn on the PMOS for a fixed number of clock cycles. The inductor current would increase linearly. From the "UP" state we go to the "DOWN" state, where we turn on the NMOS. The inductor current would decrease roughly linearly.

The "zero-cross" comparator observes the voltage across the NMOS drain/source. As soon as we turn the NMOS on the current direction in the inductor is still from DCC to V_o. Since the current is pulled from ground, the DCC must be below ground. As the current in the inductor decreases, the voltage across the NMOS will at some point be equal to zero, at which point the inductor current is zero.

When $vz = 1$ happens in the state diagram, or the zero cross comparator triggers, we transition from the "DWN" state back to "IDLE". Now the FSM wait for the next time $V_o < V_{REF}$.



I think the name "pulsed-frequency mode" refers to the fact that the frequency changes according to load current, however, I'm not sure of the origin of the name. The name is not important. What's important is that you understand that mode 1 (PWM) and mode 2 (PFM) are two different "operation modes" of a DC/DC converter.

I made a jupyter model for the PFM mode. I would encourage you to play with them.

Below you can see a period of the PFM buck. The state can be seen in the bottom plot, the voltage in the middle and the current in the inductor and load in the top plot.

Jupyter PFM BUCK model

media/l07$_buck_pfm_fig.svg$

9.4 Want to learn more?

Search terms: regulator, buck converter, dc/dc converter, boost converter

9.4.1 Linear regulators

A Scalable High-Current High-Accuracy Dual-Loop Four-Phase Switching LDO for Microprocessors Overview of fancy LDO schemes, digital as well as analog

Development of Single-Transistor-Control LDO Based on Flipped Voltage Follower for SoC In capacitor less LDOs a flipped voltage follower is a common circuit, worth a read.

A 200-mA Digital Low Drop-Out Regulator With Coarse-Fine Dual Loop in Mobile Application Processor Some insights into large power systems.

9.4.2 DC-DC converters

Design Techniques for Fully Integrated Switched-Capacitor DC-DC Converters Goes through design of SC DC-DC converters. Good place to start to learn the trade-offs, and the circuits.

High Frequency Buck Converter Design Using Time-Based Control Techniques I love papers that challenge "this is the way". Why should we design analog feedback loops for our bucks, why not design digital feedback loops?

Single-Inductor Multi-Output (SIMO) DC-DC Converters With High Light-Load Efficiency and Minimized Cross-Regulation for Portable Devices Maybe you have many supplies you want to drive, but you don't want to have many inductors. SIMO is then an option

A 10-MHz 2–800-mA 0.5–1.5-V 90% Peak Efficiency Time-Based Buck Converter With Seamless Transition Between PWM/PFM Modes Has some lovely illustrations of PFM and PWM and the trade-offs between those two modes.

A monolithic current-mode CMOS DC-DC converter with on-chip current-sensing technique In bucks converters there are two "religious" camps. One hail to "voltage mode" control loop, another hail to "current mode" control loops. It's good to read about both and make up your own mind.

Chapter 10

Clocks and PLLs

10.1 Why clocks?

Virtually all integrated circuits have some form of clock system.

For digital we need clocks to tell us when the data is correct. For Radio's we need clocks to generate the carrier wave. For analog we need clocks for switched regulators, ADCs, accurate delay's or indeed, long delays.

The principle of a clock is simple. Make a 1-bit digital signal that toggles with a period T and a frequency $f = 1/T$.

The implementation is not necessarily simple.

The key parameters of a clock are the frequency of the fundamental, noise of the frequency spectrum, and stability over process and enviromental conditions.

When I start a design process, I want to know why, how, what (and sometimes who). If I understand the problem from first principles it's more likely that the design will be suitable.

But proving that something is suitable, or indeed optimal, is not easy in the world of analog design. Analog design is similar to physics. An hypothesis is almost impossible to prove "correct", but easier to prove wrong.

10.1.1 A customer story

Take an example.

10.1.1.1 Imagine a world

> "I have a customer that needs an accurate clock to count seconds". – Some manager that talked to a customer, but don't understand details.

As a designer, I might latch on to the word "accurate clock", and translate into "most accurate clock in the world", then I'd google atomic clocks, like Rubidium standard that I know is based on the hyperfine transition of electrons between two energy levels in rubidium-87.

I know from quantum mechanics that the hyperfine transition between two energy levels will produce an precise frequency, as the frequency of the photons transmitted is defined by $E = \hbar\omega = hf$.

I also know that quantum electro dynamics is the most precise theory in physics, so we know what's going on.

I know that as long as the Rubidium crystal is clean (few energy states in the vicinity of the hyperfine transition), the distance between atoms stay constant, the temperature does not drift too much, then the frequency will be precise. So I buy a rubidium oscillator at a cost of $ 3k.

I design a an ASIC to count the clock ticks, package it plastic, make a box, and give my manager.

Who will most likely say something like

> "Are you insane? The customer want's to put the clock on a wristband, and make millions. We can't have a cost of $ 3k per device. You must make it smaller an it must cost 10 cents to make"

Where I would respond.

> "What you're asking is physically impossible. We can't make the device that cheap, or that small. Nobody can do that."

And both my manager and I would be correct.

10.1.1.2 Imagine a better world

Most people in this world have no idea how things work. Very few people are able to understand the full stack. Everyone of us must simplify what we know to some extent. As such, as a designer, it's your responsibility to fully understand what is asked of you.

When someone says

> " I have a customer that needs an accurate clock to count seconds"

Your response should be "Why does the customer need an accurate clock? How accurate? What is the customer going to use the clock for?". Unless you understand the details of the problem, then your design will be sub-optimal. It might be a great clock source, but it will be useless for solving the problem.

10.1.2 Frequency

The frequency of the clock is the frequency of the fundamental. If it's a digital clock (1-bit) with 50 % duty-cycle, then we know that a digital pulse train is an infinite sum of odd-harmnoics, where the fundamental is given by the period of the train.

10.1.3 Noise

Clock noise have many names. Cycle-to-cycle jitter is how the period changes with time. Jitter may also mean how the period right now will change in the future, so a time-domain change in the amount of cycle-to-cycle jitter. Phase noise is how the period changes as a function of time scales. For example, a clock might have fast period changes over short time spans, but if we average over a year, the period is stable.

What type of noise you care about depends on the problem. Digital will care about the cycle-to-cycle jitter affects on setup and hold times. Radio's will care about the frequency content of the noise with an offset to the carrier wave.

10.1.4 Stability

The variation over all corners and enviromental conditions is usually given in a percentage, parts per million, or parts per billion.

For a digital clock to run a Micro-Controller, maybe it's sufficient with 10% accuracy of the clock frequency. For a Bluetooth radio we must have +-50 ppm, set by the standard. For GPS we might need parts-per-billion.

10.1.5 Conclusion

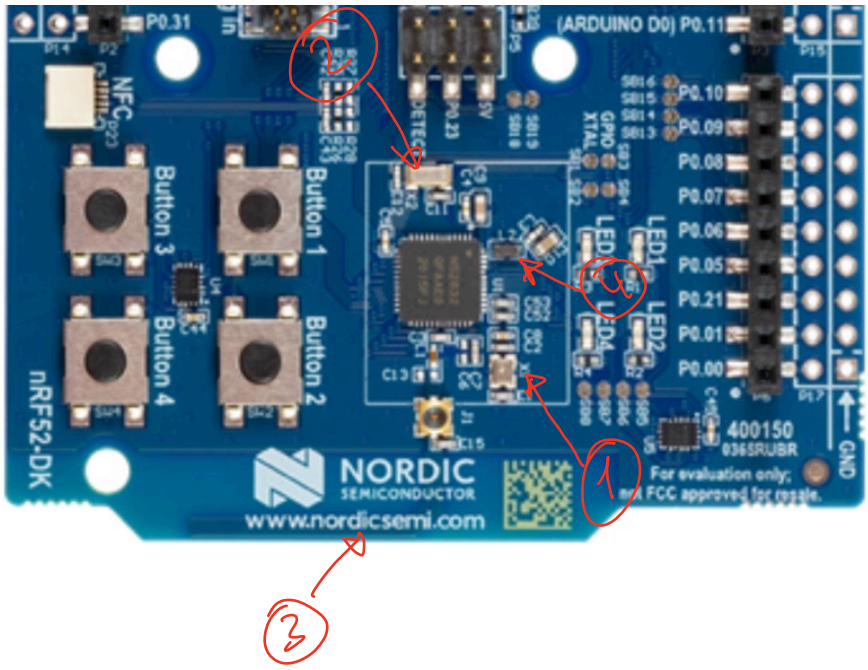
Each "clock problem" will have different frequency, noise and stability requirements. You must know the order of magnitude of those before you can design a clock source. There is no "one-solution fits all" clock generation IP.

10.2 A typical System-On-Chip clock system

On the nRF52832 development kit you can see some components that indicate what type of clock system must be inside the IC.

In the figure below you can see the following items.

1. 32 MHz crystal
2. 32 KiHz crystal
3. In PCB antenna
4. DC/DC inductor



10.2.1 32 MHz crystal

Any Bluetooth radio will need a frequency reference. We need to generate an accurate 2.402 MHz - 2.480 MHz carrier frequency for the gaussian frequency shift keying (GFSK) modulation. The Bluetooth

Standard requires a +- 50 ppm accurate timing refrence, and carrier frequency offset accuracy.

I'm not sure it's possible yet to make an IC that does not have some form of frequency reference, like a crystal. The ICs I've seen so far that have "crystal less radio" usually have a resonator (crystal or bulk-accustic-wave or MEMS resonator) on die.

The power consumption of a high frequency crystal will be proportional to frequency. Assuming we have a digital output, then the power of that digital output will be $P = CV^2f$, for example $P = 100$ fF $\times$ 1 V^2 $\times$ 32 MHz $= 3.2$ μW is probably close to a minimum power consumption of a 32 MHz clock.

10.2.2 32 KiHz crystal

Reducing the frequency, we can get down to minimum power consumption of $P = 100$ fF$\times$1 V$^2\times$32 KiHz $=$ 3.2 nW for a clock.

For a system that sleeps most of the time, and only wakes up at regular ticks to do something, then a low-frequency crystal might be worth the effort.

10.2.3 PCB antenna

Since we can see the PCB antenna, we know that the IC includes a radio. From that fact we can deduce what must be inside the SoC. If we read the Product Specification we can understand more.

10.2.4 DC/DC inductor

Since we can see a large inductor, we can also make the assumption that the IC contains a switched regulator. That switched regulator, especially if it has a pulse-width-modulated control loop, will need a clock.

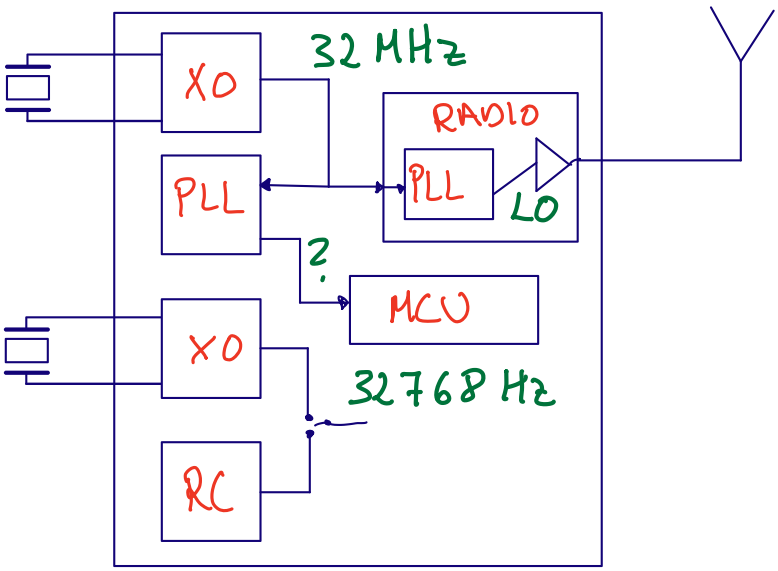
From our assumptions we could make a guess what must be inside the IC, something like the picture below.

There will be a crystal oscillator connected to the crystal. We'll learn about those later.

These crystal oscillators generate a fixed frequency, 32 MHz, or 32 KiHz, but there might be other clocks needed inside the IC.

To generate those clocks, there will be phase-locked loops (PLL), frequency locked loops (FLL), or delay-locked loops (DLL).

PLLs take a reference input, and can generate a higher frequency, (or indeed lower frequency) output. A PLL is a magical block. It's one of the few analog IPs where we can actually design for infinite gain in our feedback loop.



Most of the digital blocks on an IC will be synchronous logic, see figure below. A fundamental principle of sychnronous logic is that the data at the flip-flops (DFF, rectangles with triangle clock input, D, Q and $\overline{Q}$) only need to be correct at certain times.

The sequence of transitions in the combinatorial logic is of no consequence, as long as the B inputs are correct when the clock goes high next time.

The registers, or flip-flops, are your SystemVerilog “always_ff” code. While the blue cloud is your “always_comb” code.

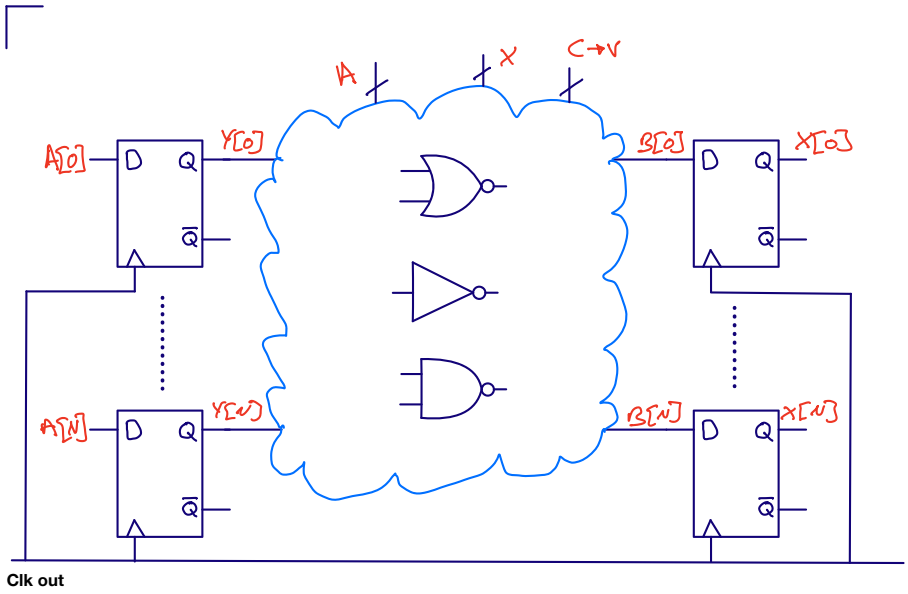
In a SoC we have to check, for all paths between a Y[N] and B[M] that the path is fast enough for all transients to settle until the clock strikes next time. How early the B data must arrive in relation to the clock edge is the setup time of the DFFs.

We also must check for all paths that the B[M] are held for long enough after the clock strikes such that our flip-flop does not change state. The hold time is the distance from the clock edge to where the data is allowed to change. Negative hold times are common in DFFs, so the data can start to change before the clock edge.

In an IC with millions of flip-flops there can be billions of paths. The setup and hold time for every single one must be checked. One could imagine a simulation of all the paths on a netlist with parasitics (capacitors and resistors from layout) to check the delays, but there are so many combinations that the simulation time becomes unpractical.

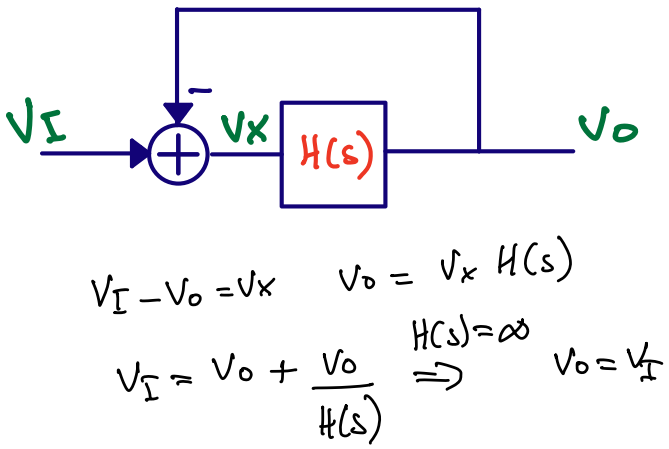
Static Timing Analysis (STA) is a light-weight way to check all the paths. For the STA we make a model of the delay in each cell (captured in a liberty file), the setup/hold times of all flip-flops, wire propagation delays, clock frequency (or period), and the variation in the clock frequency. The process, voltage, temperature variation must also be checked for all components, so the number of liberty files can quickly grow large.

For an analog designer the constraints from digital will tell us what’s the maximum frequency we can have at any point in time, and what is the maximum cycle-to-cycle variation in the period.



10.3 PLL

PLL, or it's cousins FLL and DLL are really cool. A PLL is based on the familiar concept of feedback, shown in the figure below. As long as we make $H(s)$ infinite we can force the output to be an exact copy of the input.



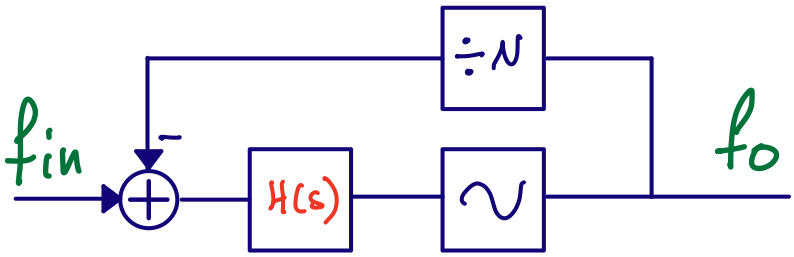
For a frequency loop the figure looks a bit different. If we want a higher output frequency we can divide the frequency by a number (N) and compare with our reference (for example the 32 MHz reference from the crystal oscillator).

We then take the error, apply a transfer function $H(s)$ with high gain, and control our oscillator frequency.

If the down-divided output frequency is too high, we force the oscillator to a lower frequency. If the down-divided output frequency is too low we force the oscillator to a higher frequency.

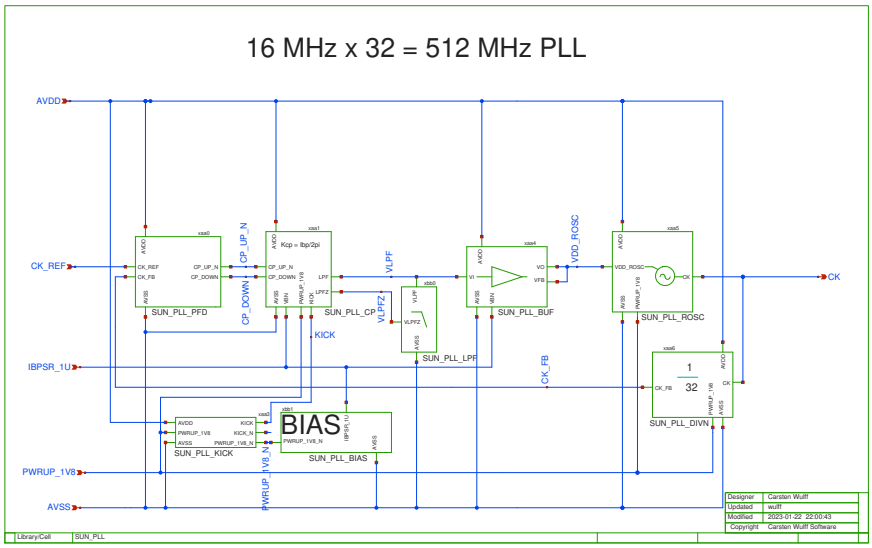
If we design the $H(s)$ correctly, then we have $f_o = N \times f_{in}$

Sometimes you want a finer freuqnecy resolution, in that case you'd add a divider on the reference and get $f_o = N \times \frac{f_{in}}{M}$..



I've made an example PLL that you can download and play with. I make no claims that it's a good PLL. Actually, I know it's a bad PLL. The ring-oscillator frequency varies to fast with the voltage control. But it does give you a starting point.

A PLL can consist of a oscillator (SUN_PLL_ROSC) that generates our output frequency. A divider (SUN_PLL_DIVN) that generates a feedback frequency that we can compare to the reference. A Phase and Frequency Detector (SUN_PLL_PFD) and a charge-pump (SUN_PLL_CP) that model the +, or the comparison function in our previous picture. And a loop filter (SUN_PLL_LPF and SUN_PLL_BUF) that is our $H(s)$.



10.4 PLLs need calculation!

#noCowboyDesign

Read any book on PLLs, talk to any PLL designer and they will all tell you the same thing. **PLLs require calculation**. You must setup a linear model of the feedback loop, and calculate the loop transfer function to check the stability, and the loop gain. **This is the way!** (to quote Mandalorian).

But how can we make a linear model of a non-linear system? The voltages inside a PLL must be non-linear, they are clocks. A PLL is not linear in time-domain!

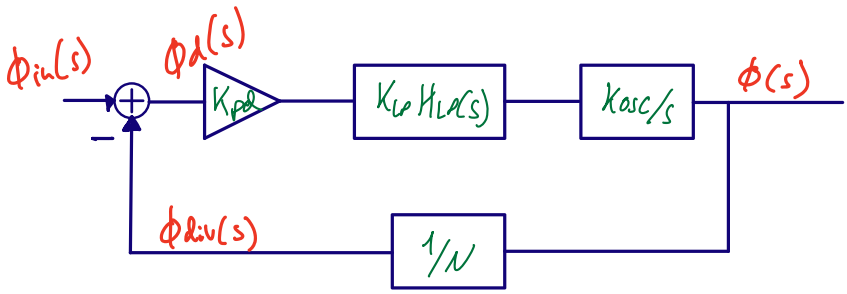
I have no idea who first thought of the idea, but it turns out, that one can model a PLL as a linear system if one consider the phase of the voltages inside the PLL. Where the phase is defined as

$$\phi(t) = 2\pi \int_0^t f(t)dt$$

As long as the bandwidth of the $H(s)$ is about $\frac{1}{10}$ of the reference frequency, then the linear model below holds (at least is good enough).

The phase of our input is $\phi_{in}(s)$, the phase of the output is $\phi(s)$, the divided phase is $\phi_{div}(s)$ and the phase error is $\phi_d(s)$.

The K_{pd} is the gain of our phase-frequency detector and charge-pump. The $K_{lp}H_{lp}(s)$ is our loop filter $H(s)$. The K_{osc}/s is our oscillator transfer function. And the $1/N$ is our feedback divider.



10.4.1 Loop gain

The loop transfer function can then be analyzed and we get.

$$\frac{\phi_d}{\phi_{in}} = \frac{1}{1 + L(s)}$$

$$L(s) = \frac{K_{osc}K_{pd}K_{lp}H_{lp}(s)}{Ns}$$

Here is the magic of PLLs. Notice what happens when $s = j\omega = j0$, or at zero frequency. If we assume that $H_{lp}(s)$ is a low pass filter, then $H_{lp}(0) =$ constant. The loop gain, however, will have a $L(0) \propto \frac{1}{0}$ which approaces infinity at 0.

That means, we have an infinite DC gain in the loop transfer function. It is the only case I know of in an analog design where we can actually have infinite gain. Infinite gain translate can translate to infinite precision.

If the reference was a Rubidium oscillator we could generate any frequency with the same precision as the frequency of the Rubidium oscillator. Magic.

For the linear model, we need to figure out the factors, like K_{vco}, which must be determined by simulation.

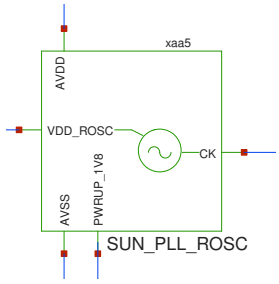
10.4.2 Controlled oscillator

The gain of the oscillator is the change in output frequency as a function of the change of the control node. For a voltage-controlled oscillator (VCO) we could sweep the control voltage, and check the frequency. The derivative of the f(V) would be proportional to the K_{vco}.

The control node does not need to be a voltage. Anything that changes the frequency of the oscillator can be used as a control node. There exist PLLs with voltage control, current control, capacitance control, and digital control.

For the SUN_PLL_ROSC it is the VDD of the ring-oscillator (VDD_ROSC) that is our control node.

$$K_{osc} = 2\pi \frac{df}{dV_{cntl}}$$



10.4.2.1 SUN_PLL_SKY130NM/sim/ROSC/

I simulate the ring oscillator in ngspice with a transient simulation and get the oscilator frequency as a function of voltage.

tran.spi

```
let start_v = 1.1
let stop_v = 1.7
let delta_v = 0.1
let v_act = start_v
* loop
while v_act le stop_v
alter VROSC v_act
tran 1p 40n
meas tran vrosc avg v(VDD_ROSC)
meas tran tpd trig v(CK) val='0.8' rise=10 targ v(CK) val='0.8' rise=11
let v_act = v_act + delta_v
end
```

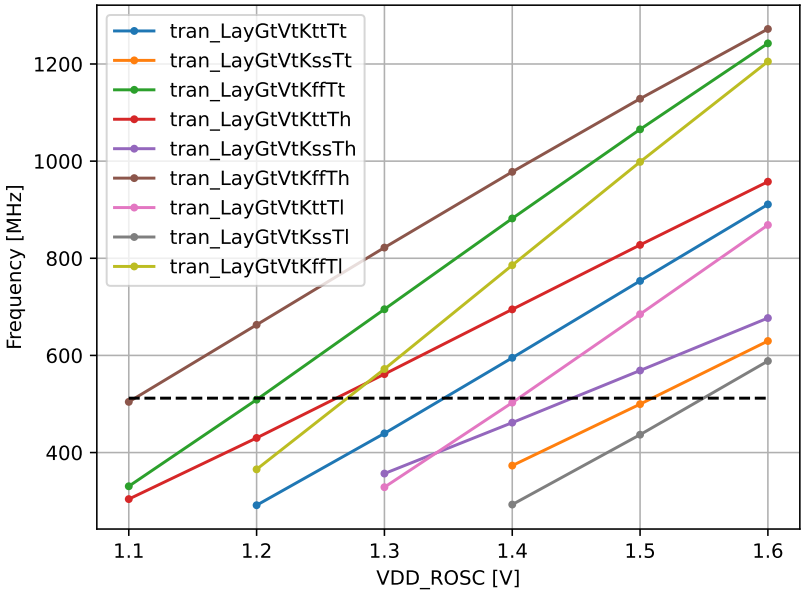
I use `tran.py` to extract the time-domain signal from ngspice into a CSV file.

Then I use a python script to extract the K_{osc}

kvco.py

```
df = pd.read_csv(f)
freq = 1/df["tpd"]
kvco = np.mean(freq.diff()/df["vrosc"].diff())
```

Below I've made a plot of the oscillation frequency over corners.

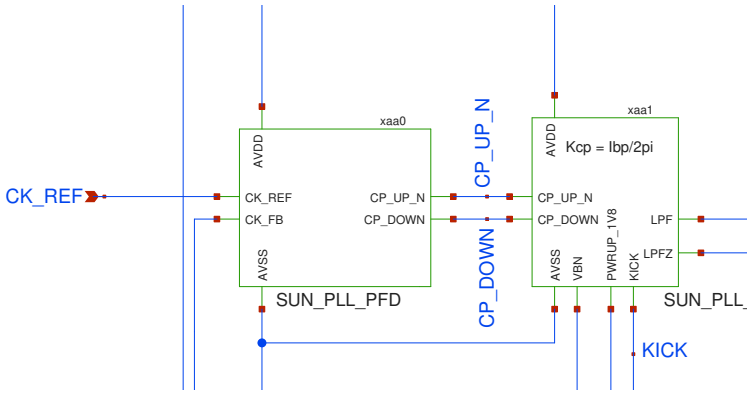


10.4.3 Phase detector and charge pump

The gain of the phase-detector and charge pump is the current we feed into the loop filter over a period. I don't remember why, check in the book for a detailed description.

The two blocks compare our reference clock to our feedback clock, and produce an error signal.

$$K_{pd} = \frac{I_{cp}}{2\pi}$$



10.4.4 Loop filter

In the book you'll find a first order loop filter, and a second order loop filter. Engineers are creative, so you'll likely find other loop filters in the literature.

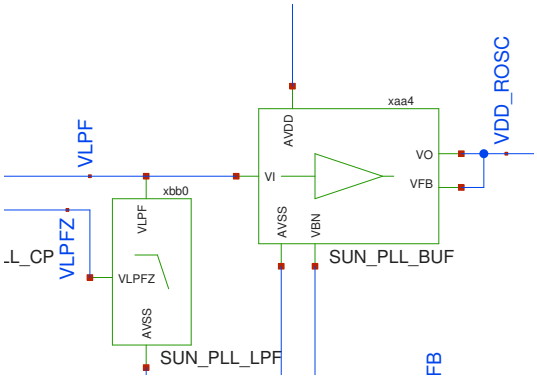
I would start with the "known to work" loop filters before you explore on your own.

If you're really interested in PLLs, you should buy Design of CMOS Phase-Locked Loops by Behzad Razavi.

The loop filter has a unity gain buffer. My oscillator draws current, while the VPLF node is high impedant, so I can't draw current from the loop filter without changing the filter transfer function.

$$K_{lp}H_{lp}(s) = K_{lp}\left(\frac{1}{s} + \frac{1}{\omega_z}\right)$$

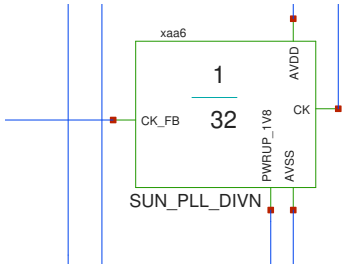
$$K_{lp}H_{lp}(s) = \frac{1}{s(C_1 + C_2)} \frac{1 + sRC_1}{1 + sR\frac{C_1C_2}{C_1+C_2}}$$



10.4.5 Divider

The divider is modelled as

$$K_{div} = \frac{1}{N}$$



10.4.6 Loop transfer function

With the loop transfer function we can start to model what happens in the linear loop. What is the phase response, and what is the gain response.

$$L(s) = \frac{K_{osc}K_{pd}K_{lp}H_{lp}(s)}{Ns}$$

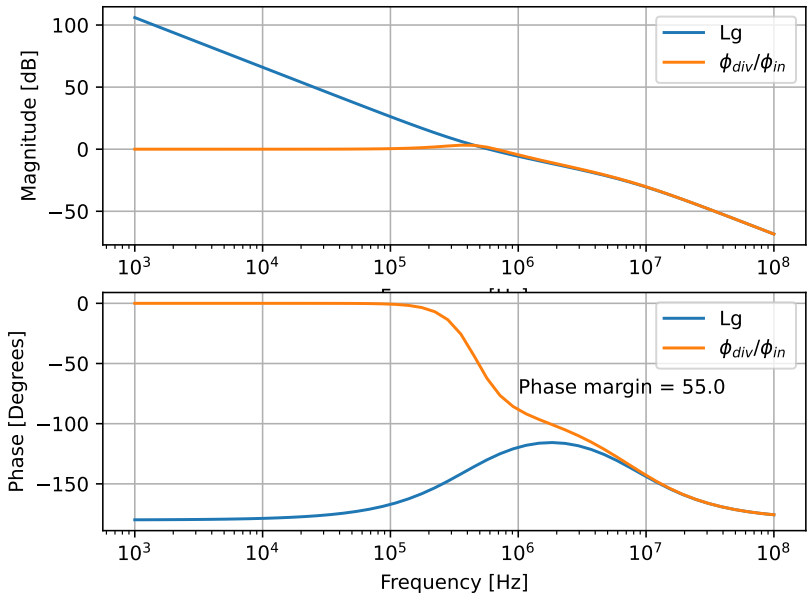
10.4.6.1 Python model

I've made a python model of the loop, you can find it at sun_pll_sky130nm/py/pll.py

Below is a plot of the loop gain, and the transfer function from input phase to divider phase.

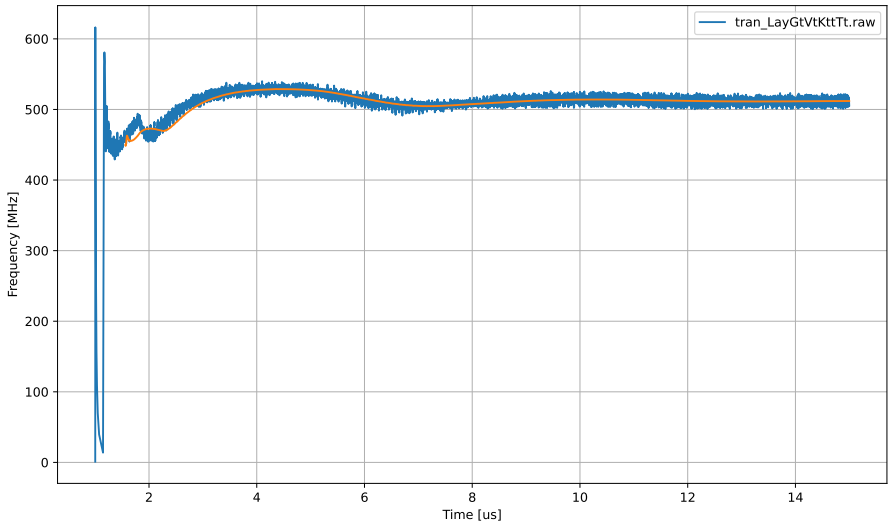
We can see that the loop gain at low frequency is large, and proportional to $1/s$. As such, the phase of the divided down feedback clock is the same as our reference.

The closed loop transfer function ϕ_{div}/ϕ_{in} shows us that the divided phase at low frequency is the same as the input phase. Since the phase is the same, and the frequency must be the same, then we know that the output clock will be N times reference frequency.



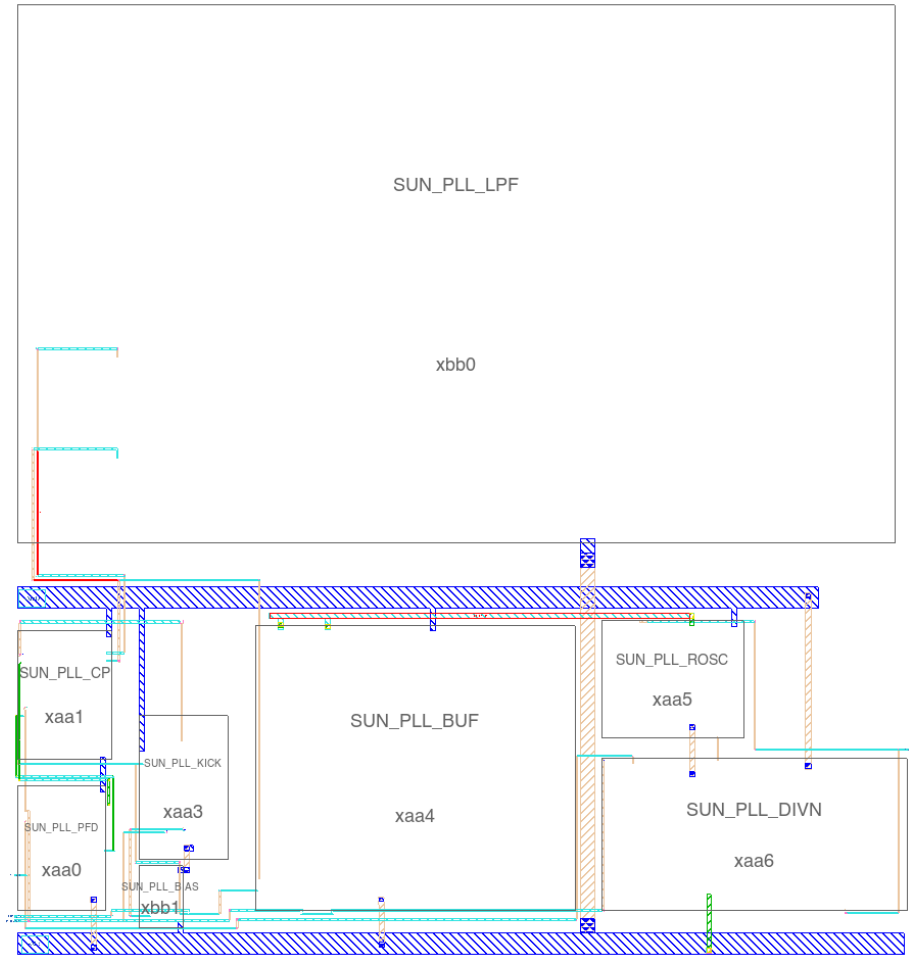
The top testbench for the PLL is tran.spi.

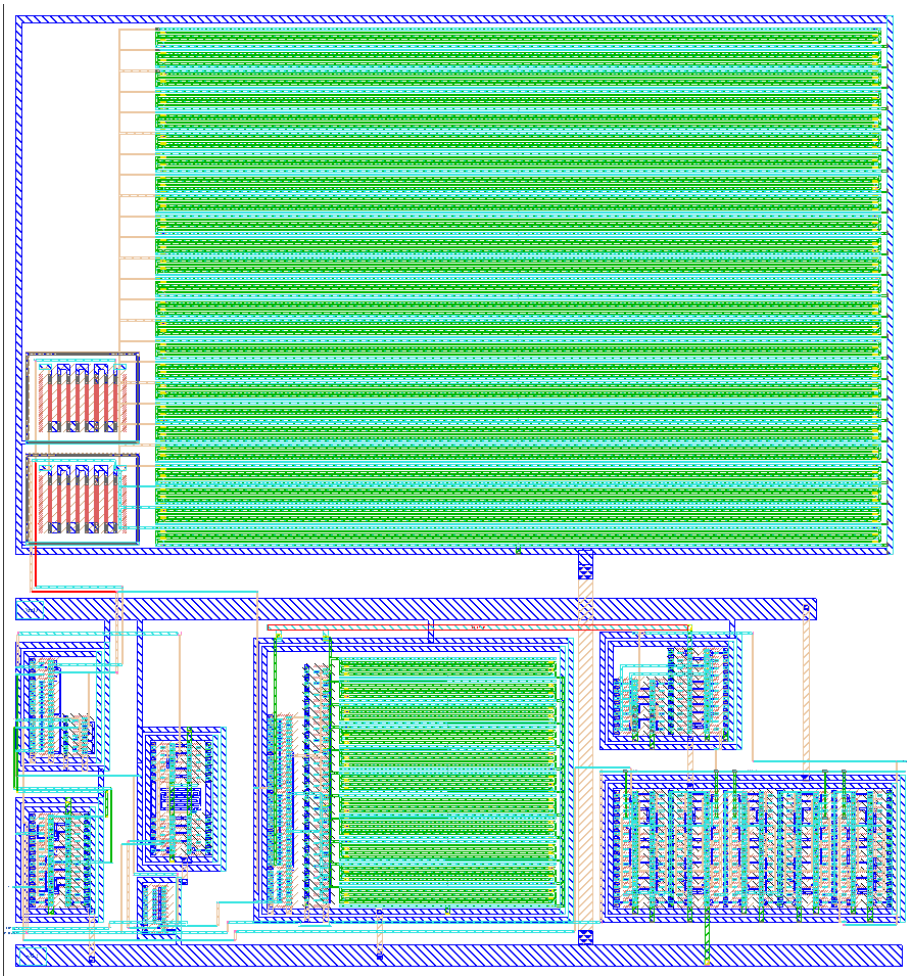
I power up the PLL and wait for the output clock to settle. I use freq.py to plot the frequency as a function of time. The orange curve is the average frequency. We can see that the output frequency settles to 512 MHz.



You can find the schematics, layout, testbenches, python script etc at SUN_PLL_SKY130NM

Below are a couple layout images of the finished PLL





10.4.7 Want to learn more?

Back in 2020 there was a Master student at NTNU on PLL. I would recommend looking at that thesis to learn more, and to get inspired Ultra Low Power Frequency Synthesizer.

A Low Noise Sub-Sampling PLL in Which Divider Noise is Eliminated and PD/CP Noise is Not Multiplied by N2

All-digital PLL and transmitter for mobile phones

A 2.9–4.0-GHz Fractional-N Digital PLL With Bang-Bang Phase Detector and 560-fsrms Integrated Jitter at 4.5-mW Power

Chapter 11

Oscillators

11.1 Goal

11.2 Crystal oscillators

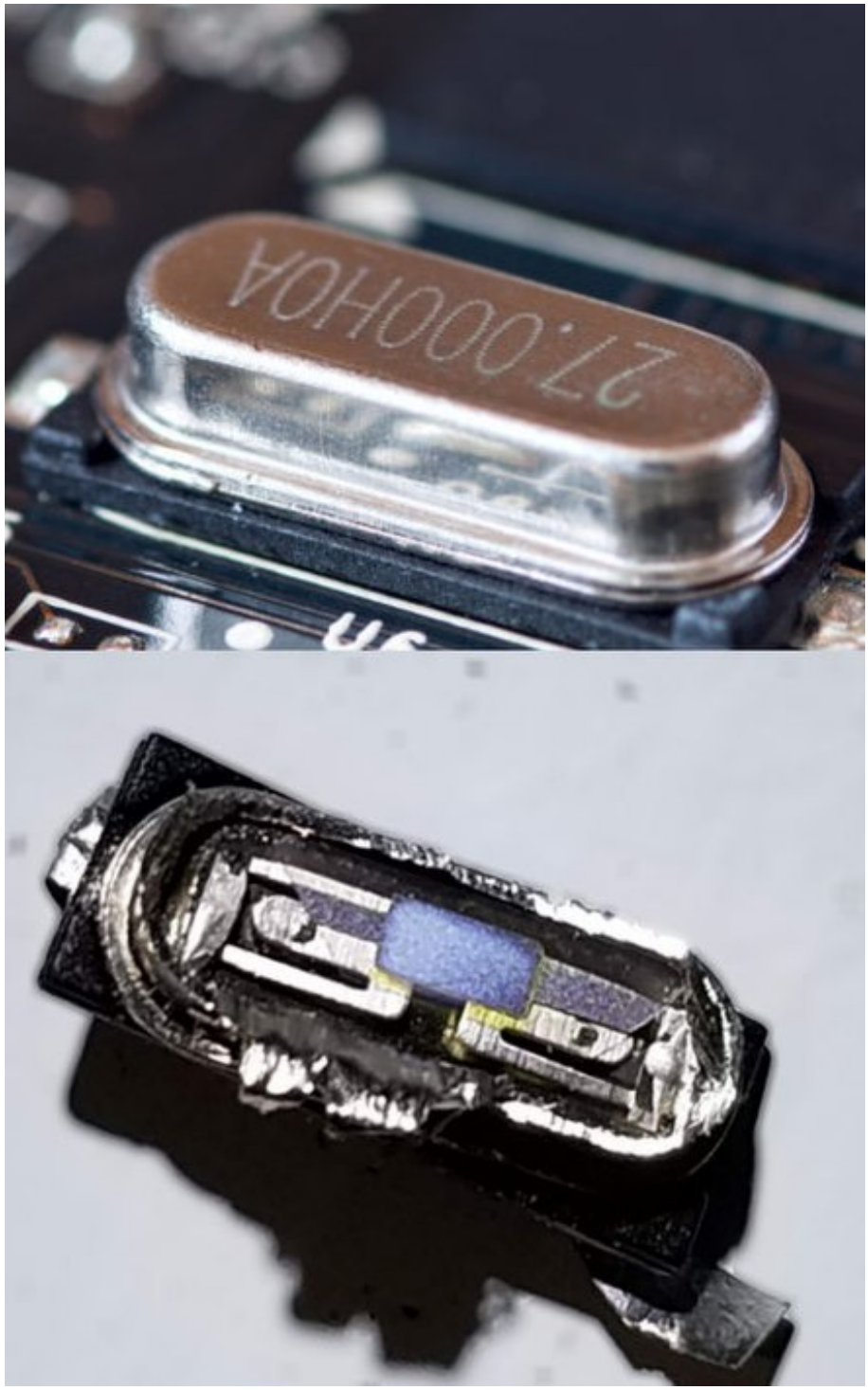
The world depends on accurate clocks. From the timepiece on your wrist, to the phone in your pocket, they all have a need for an accurate way of counting the passing of time.

Without accurate clocks an accurate GPS location would not be possible. In GPS we even correct for Special and General Relativity to the tune of about $+38.6\mu s$/day .

The most accurate clocks, like Rubidium standard, use the hyper-fine splitting of energy levels in atoms, however, most clocks don't need to be accurate to parts per billion.

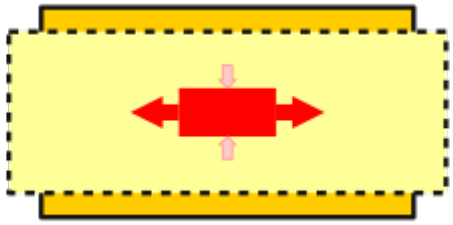
For accuracy's of parts per million, which is sufficient for your wrist watch, or most communication, it's possible to use crystals.

A quartz crystal can resonate at specific frequencies. If we apply a electric field across a crystal, we will induce a vibration in the crystal, which can again affect the electric field. For some history, see Crystal Oscillators

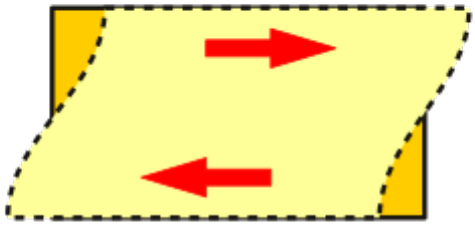


The vibrations in the crystal lattice can have many modes, as illustrated by figure below.

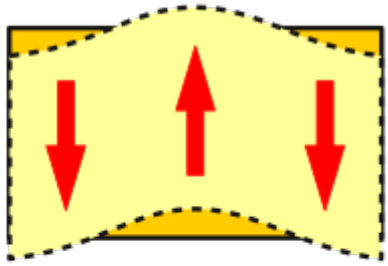
All we need to do with a crystal is to inject sufficient energy to sustain the oscillation, and the resonance of the crystal will ensure we have a correct enough frequency.



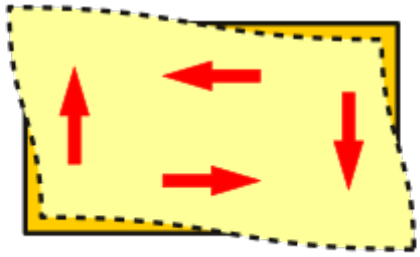
Longitudinal mode



Thickness shear mode



Flexural mode



Face shear mode

Tuning fork

11.2.1 Impedance

The impedance of a crystal is usually modeled as below. A RLC circuit with a parallel capacitor.

Our job is to make a circuit that we can connect to the two pins and provide the energy we will loose due to R_s.

media/xosc$_m$*odel.svg*

Assuming zero series resistance

$$Z_{in} = \frac{s^2 C_F L + 1}{s^3 C_P L C_F + s C_P + s C_F}$$

Notice that at $s = 0$ the impedance goes to infinity, so a crystal is high impedant at DC.

Since the 1/(sCp) does not change much at resonance, then

$$Z_{in} \approx \frac{L C_F s^2 + 1}{L C_F C_p s^2 + C_F + C_P}$$

See Crystal oscillator impedance for a detailed explanation.

In the impedance plot below we can clearly see that there are two "resonance" points. Usually noted by series and parallel resonance.

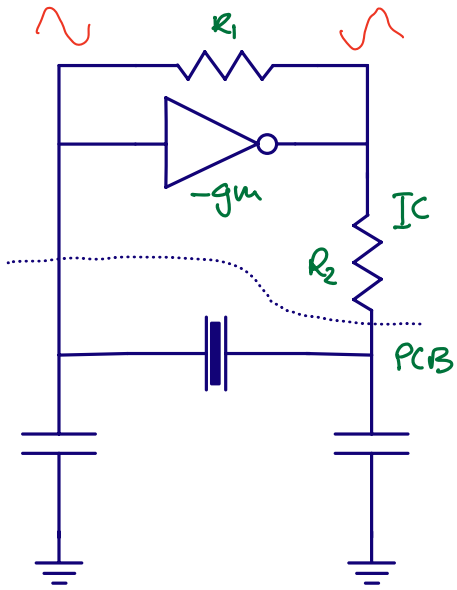
I would encourage you to read The Crystal Oscillator for more details.

media/xosc$_r$*es.svg*

11.2.2 Circuit

Below is a common oscillator circuit, a Pierce Oscillator. The crystal is the below the dotted line, and the two capacitance's are the on-PCB capacitance's.

Above the dotted line is what we have inside the IC. Call the left side of the inverter XC1 and right side XC2. The inverter is biased by a resistor, R_1, to keep the XC1 at a reasonable voltage. The XC1 and XC2 will oscillate in opposite directions. As XC1 increases, XC2 will decrease. The R_2 is to model the internal resistance (on-chip wires, bond-wire).



Negative transconductance compensate crystal series resistance

The transconductance of the inverter must compensate for the energy loss caused by R_s in the crystal model. The transconductor also need to be large enough for the oscillation to start, and build up.

I've found that sometimes people get confused by the negative transconductance. There is nothing magical about that. Imagine the PMOS and the NMOS in the inverter, and that the input voltage is exactly the voltage we need for the current in the PMOS and NMOS to be the same. If the current in the PMOS and NMOS is the same, then there can be no current flowing in the output.

Imagine we increase the voltage. The PMOS current would decrease, and the NMOS current would increase. We would pull current from the output.

Imagine we now decrease the voltage instead. The PMOS current would increase, and the NMOS current would decrease. The current in the output would increase.

As such, a negative transconductance is just that as we increase the input voltage, the current into the output decreases, and visa versa.

Long startup time caused by high Q

The Q factor has a few definitions, so it's easy to get confused. Think of Q like this, if a resonator has high Q, then the oscillations die out slowly.

Imagine a perfect world without resistance, and an inductor and capacitor in parallel. Imagine we initially store some voltage across the capacitor, and we let the circuit go. The inductor shorts the plates of the capacitor, and the current in the inductor will build up until the voltage across the capacitor is zero. The inductor still has stored current, and that current does not stop, so the voltage across the capacitor will become negative, and continue decreasing until the inductor current is zero. At that point the negative voltage will flip the current in the inductor, and we go back again.

The LC circuit will resonate back and forth. If there was no resistance in the circuit, then the oscillation would never die out. The system would be infinite Q.

The Q of the crystal oscillator can be described as $Q = 1/(\omega R_s C_f)$, assuming some common values of

$R_s = 50$, $C_f = 5e^{-15}$ and $\omega = 2\pi \times 32$ MHz then $Q \approx 20$ k.

That number may not tell you much, but think of it like this, it will take 20 000 clock cycles before the amplitude falls by 1/e. For example, if the amplitude of oscillation was 1 V, and you stop introducing energy into the system, then 20 000 clock cycles later, or 0.6 ms, the amplitude would be 0.37 V.

The same is roughly true for startup of the oscillator. If the crystal had almost no amplitude, then an increase e would take 20 k cycles. Increasing the amplitude of the crystal to 1 V could take milliseconds.

Most circuits on-chip have startup times on the order of microseconds, while crystal oscillators have startup time on the order of milliseconds. As such, for low power IoT, the startup time of crystal oscillators, or indeed keeping the oscillator running at a really low current, are key research topics.

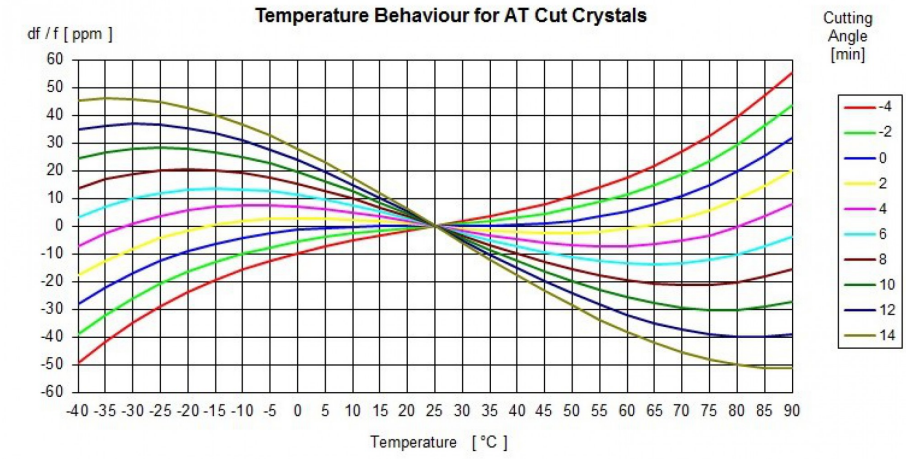
Can fine tune frequency with parasitic capacitance

The resonance frequency of the crystal oscillator can be modified by the parasitic capacitance from XC1 and XC2 to ground. The tunability of crystals is usually in ppm/pF. Sometimes micro-controller vendors will include internal load capacitance's to support multiple crystal vendors without changing the PCB.

11.2.3 Temperature behavior

One of the key reasons for using crystals is their stability over temperature. Below is a plot of a typical temperature behavior. The cutting angle of the crystal affect the temperature behavior, as such, the closer crystals are to "no change in frequency over temperature", the more expensive they become.

In communication standards, like Bluetooth Low Energy, it's common to specify timing accuracy's of +-50 ppm. Have a look in the Bluetooth Core Specification 5.4 Volume 6, Part A, Chapter 3.1 (page 2653) for details.



11.3 Controlled Oscillators

On an integrated circuit way may need multiple clocks, and we can't have crystal oscillators for all of them. We can use frequency locked loops, phase locked loops and delay locked loops to make multiples of the crystal reference frequency.

All phase locked loops contain an oscillator where we control the frequency of oscillation.

11.3.1 Ring oscillator

The simplest oscillator is a series of inverters biting their own tail, a ring oscillator.

The delay of each stage can be thought of as a RC time constant, where the R is the transconductance of the inverter, and the C is the gate capacitance of the next inverter.

$$t_{pd} \approx RC$$

$$R \approx \frac{1}{gm} \approx \frac{1}{\mu_n C_{ox} \frac{W}{L}(VDD - V_{th})}$$

$$C \approx \frac{2}{3} C_{ox} W L$$

media/osc$_r ing.svg$

One way to change the oscillation frequency is to change the VDD of the ring oscillator. Based on the delay of a single inverter we can make an estimate of the oscillator gain. How large change in frequency do we get for a change in VDD.

$$t_{pd} \approx \frac{2/3 C_{ox} W L}{\frac{W}{L}\mu_n C_{ox}(VDD - V_{th})}$$

$$f = \frac{1}{2N t_{pd}} = \frac{\mu_n (VDD - V_{th})}{\frac{4}{3} N L^2}$$

$$K_{vco} = 2\pi \frac{\partial f}{\partial VDD} = \frac{2\pi \mu_n}{\frac{4}{3} N L^2}$$

The K_{vco} is proportional to mobility, and inversely proportional to the number of stages and the length of the transistor squared. In most PLLs we don't want the K_{vco} to be too large. Ideally we want the ring oscillator to oscillate close to the frequency we want, i.e 512 MHz, and a small K_{vco} to account for variation over temperature (mobility of transistors decreases with increased temperature, the threshold voltage of transistors decrease with temperature), and changes in VDD.

To reduce the K_{vco} of the standard ring oscillator we can increase the gate length, and increase the number of stages.

I think it's a good idea to always have a prime number of stages in the ring oscillator. I have seen some ring oscillators with 21 stages oscillate at 3 times the frequency in measurement. Since $21 = 7 \times 3$ it's possible to have three "waves" of traveling through the ring oscillator at all times, forever. If you use a prime number of stages, then sustained oscillation at other frequencies cannot happen.

As such, then number of inverter stages should be $\in [3, 5, 7, 11, 13, 17, 19, 23, 29, 31]$

11.3.2 Capacitive load

The oscillation frequency of the ring oscillator can also be changed by adding capacitance.

$$f = \frac{\mu_n C_{ox} \frac{W}{L} (VDD - V_{th})}{2N \left(\frac{2}{3} C_{ox} WL + C \right)}$$

$$K_{vco} = \frac{2\pi \mu_n C_{ox} \frac{W}{L}}{2N \left(\frac{2}{3} C_{ox} WL + C \right)}$$

Assume that the extra capacitance is much larger than the gate capacitance, then

$$f = \frac{\mu_n C_{ox} \frac{W}{L} (VDD - V_{th})}{2NC}$$

$$K_{vco} = \frac{2\pi \mu_n C_{ox} \frac{W}{L}}{2NC}$$

And maybe we could make the K_{vco} relatively small.

The power consumption of an oscillator, however, will be similar to a digital circuit of $P = C \times f \times VDD^2$, so increasing capacitance will also increase the power consumption.

media/osc$_r ing_c .svg$

11.3.3 Realistic

Assume you wanted to design a phase-locked loop, what type of oscillator should you try first? If the noise of the clock is not too important, so you don't need an LC-oscillator, then I'd try the oscillator below, although I'd expand the number of stages to fit the frequency.

The circuit has a capacitance loaded ring oscillator fed by a current. The $I_{control}$ will give a coarse control of the frequency, while the $V_{control}$ can give a more precise control of the frequency.

Since the $V_{control}$ can only increase the frequency it's important that the $I_{control}$ is set such that the frequency is below the target.

Most PLLs will include some form of self calibration at startup. At startup the PLL will do a coarse calibration to find a sweet-spot for $I_{control}$, and then use $V_{control}$ to do fine tuning.

Since PLLs always have a reference frequency, and a phase and frequency detector, it's possible to sweep the calibration word for $I_{control}$ and then check whether the output frequency is above or below the target based on the phase and frequency detector output. Although we don't know exactly what the oscillator frequency is, we can know the frequency close enough.

It's also possible to run a counter on the output frequency of the VCO, and count the edges between two reference clocks. That way we can get a precise estimate of the oscillation frequency.

Another advantage with the architecture below is that we have some immunity towards supply noise. If we decouple both the current mirror, and the $V_{control}$ towards VDD, then any change to VDD will not affect the current into the ring oscillator.

Maybe a small side track, but inject a signal into an oscillator from an amplifier, the oscillator will have a tendency to lock to the injected signal, we call this "injection locking", and it's common to do in ultra high frequency oscillators (60 - 160 GHz). Assume we allow the PLL to find the right $V_{control}$ that corresponds to the injected frequency. Assume that the injected frequency changes, for example frequency shift keying (two frequencies that mean 1 or 0), as in Bluetooth Low Energy. The PLL will vary the $V_{control}$ of the PLL to match the frequency change of the injected signal, as such, the $V_{control}$ is now the demodulated frequency change.

Still today, there are radio recievers that use a PLLs to directly demodulate the incoming frequency shift keyed modulated carrier.

media/osc$_r ing_a dv.svg$

We can calculate the K_{vco} of the oscillator as shown below. The inverters mostly act as switches, and when the PMOS is on, then the rise time is controlled by the PMOS current mirror, the additional $V_{control}$ and the capacitor. For the calculation below we assume that the pull-down of the capacitor by the NMOS does not affect the frequency much.

The advantage with the above ring-oscillator is that we can control the frequency of oscillation with $I_{control}$ and have a independent K_{vco} based on the sizing of the $V_{control}$ transistors.

$$I = C\frac{dV}{dt}$$

$$f \approx \frac{I_{control} + \frac{1}{2}\mu_p C_{ox}\frac{W}{L}(VDD - V_{control} - V_{th})^2}{C\frac{VDD}{2}N}$$

$$K_{vco} = 2\pi\frac{\partial f}{\partial V_{control}}$$

$$K_{vco} = 2\pi\frac{\mu_p C_{ox} W/L}{C\frac{VDD}{2}N}$$

11.3.4 Digitally controlled oscillator

We can digitally control the oscillator frequency as shown below by adding capacitors.

Today there are all digital loops where the oscillator is not really a "voltage controlled oscillator", but rather a "digital control oscillator". DCOs are common in all-digital PLLs.

Another reason to use digital frequency control is to compensate for process variation. We know that mobility affects the K_{vco}, as such, for fast transistors the frequency can go up. We could measure the free-running frequency in production, and compensate with a digital control word.

media/osc$_r ing_c ap.svg$

11.3.5 Differential

Differential circuits are potentially less sensitive to supply noise .

Imagine a single ended ring oscillator. If I inject a voltage onto the input of one of the inverters that was just about to flip, I can either delay the flip, or speed up the flip, depending on whether the voltage pulse increases or decreases the input voltage for a while. Such voltage pulses will lead to jitter.

Imagine the same scenario on a differential oscillator (think diff pair). As long as the voltage pulse is the same for both inputs, then no change will incur. I may change the current slightly, but that depends on the tail current source.

Another cool thing about differential circuits is that it's easy to multiply by -1, just flip the wires, as a result, I can use a 2 stage ring differential ring oscillator.

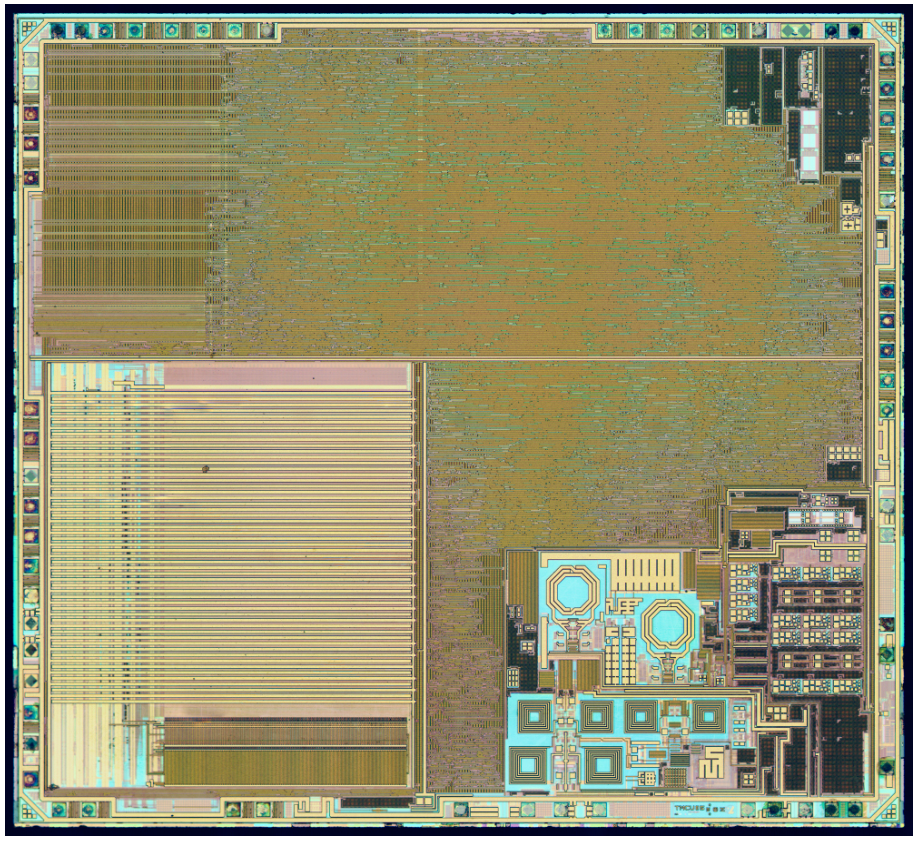
media/osc$_{r}ing_{d}iff.svg$

11.3.6 LC oscillator

Most radio's are based on modulating information on-to a carrier frequency, for example 2.402 GHz for a Bluetooth Low Energy Advertiser. One of the key properties of the carrier waves is that it must be "clean". We're adding a modulated signal on top of the carrier, so if there is noise inherent on the carrier, then we add noise to our modulation signal, which is bad.

Most ring oscillators are too high noise for radio's, we must use a inductor and capacitor to create the resonator.

Inductors are huge components on a IC. Take a look at the nRF51822 below, the two round inductors are easily identifiable. Actually, based on the die image we can guess that there are two oscillators in the nRF51822. Maybe it's a multiple conversion superheterodyne reciever



Below is a typical LC oscillator. The main resonance is set by the L and C, while the tunability is provided by a varactor, a voltage variable capacitor. Or with less fancy words, the gate capacitance of a transistor, since the gate capacitance of a transistor depends on the effective voltage, and is thus a "varactor"

The NMOS at the bottom provide the "negative transconductance" to compensate for the loss in the LC tank.

media/lcosc.svg

$$f \propto \frac{1}{\sqrt{LC}}$$

11.4 Relaxation oscillators

A last common oscillator is the relaxation oscillator, or "RC" oscillator. By know you should be proficient enough in circuits to figure out how the below circuit works.

If you can't figure it out, then ask me.

media/rcosc.svg

11.5 Additional material

11.5.1 Crystal oscillators

The Crystal Oscillator - A Circuit for All Seasons

High-performance crystal oscillator circuits: theory and application

Ultra-low Power 32kHz Crystal Oscillators: Fundamentals and Design Techniques

A Sub-nW Single-Supply 32-kHz Sub-Harmonic Pulse Injection Crystal Oscillator

11.5.2 CMOS oscillators

The Ring Oscillator - A Circuit for All Seasons

A Study of Phase Noise in CMOS Oscillators

An Ultra-Low-Noise Swing-Boosted Differential Relaxation Oscillator in 0.18-um CMOS

Chapter 12

Low Power Radio

Radio's are all around us. In our phone, on our wrist, in our house, there is Bluetooth, WiFi, Zigbee, LTE, GPS and many more.

A radio is a device that receives and transmits light encoded with information. The frequency of the light depends on the standard. How the information is encoded onto the light depends on the standard.

Assume that we did not know any standards, what would we do if we wanted to make the best radio IC for gaming mice?

There are a few key concepts we would have to know before we decide on a radio type: Data Rate, Carrier Frequency and range, and the power supply.

12.1 Data Rate

12.1.1 Data

A mouse reports on the relative displacement of the mouse as a function of time, some X and Y displacement. A mouse has buttons. There can be many mice in a room, so PCs would like to tell them apart, they must have an address.

A mouse must be low-power. As such, the radio cannot be on all the time. The radio must start up and be ready to receive quickly. We don't know how far away from the PC the mice might be, as such, we don't know the dB loss in the communication channel. As a result, the radio needs to have a high dynamic range, from weak signals to strong signals. In order for the radio to adjust the gain of the reciever we should include a pre-amble, a known sequence, for example 01010101, such that the radio can adjust the gain, and also, recover the symbol timing.

All in all, the packets we send from the mouse may need to have the following bits.

What	Bits	Why
X displacement	8	
Y displacement	8	
CRC	4	Bit errors
Buttons	16	On-hot coding. Most mice have buttons
Preamble	8	Synchronization
Address	32	Unique identifier

What	Bits	Why
Total	76	

12.1.2 Rate

Gamers are crazy for speed, they care about milliseconds. So our mice needs to be able to send and receive data quite often.

Assume 1 ms update rate

12.1.3 Data Rate

To compute the data rate, let's do a back of the envelope estimate of the data, and the rate.

Application Data Rate $>$ 76 bits/ms = 76 kbps

Assume 30 % packet loss

Raw Data Rate $>$ 228 kbps

Multiply by

$$\pi$$

$>$ 716 kbps

Round to nearest nice number = 1Mbps

The above statements are a exact copy of what happens in industry when we start design of something. We make an educated guess. More optimistic people would multiply with e.

12.2 Carrier Frequency & Range

12.2.1 ISM (industrial, scientific and medical) bands

There are rules and regulations that prevent us from transmitting and receiving at any frequency we want. We need to pick one of the ISM bands, or we need to get a license from governments around the world.

But how should we pick? There are at least two criteria that should be investigated. Antenna and Range.

6 765-6 795 kHz	(centre frequency 6 780 kHz)	FN 5.138
13 553-13 567 kHz	(centre frequency 13 560 kHz)*	FN 5.150
26 957-27 283 kHz	(centre frequency 27 120 kHz)	FN 5.150
40.66-40.70 MHz	(centre frequency 40.68 MHz)	FN 5.150
433.05-434.79 MHz	(centre frequency 433.92 MHz)in Region1**	FN 5.138
902-928 MHz	(centre frequency 915 MHz) in Region 2	FN 5.150
2 400-2 500 MHz	(centre frequency 2 450 MHz)	FN 5.150
5 725-5 875 MHz	(centre frequency 5 800 MHz)	FN 5.150
24-24.25 GHz	(centre frequency 24.125 GHz)	FN 5.150
61-61.5 GHz	(centre frequency 61.25 GHz)	FN 5.138
122-123 GHz	(centre frequency 122.5 GHz)	FN 5.138
244-246 GHz	(centre frequency 245 GHz)	FN 5.138

12.2.2 Antenna

For a mouse we want to hold in our hand, there is a size limit to the antenna. There are many types of antenna, but a

assume

$$\lambda/4$$

is an OK antenna size (

$$\lambda = c/f$$

)

The below table shows the ISM band and the size of a quarter wavelength antenna. Any frequency above 2.4 GHz may be OK from a size perspective.

ISM band	$\lambda/4$	Unit	OK/NOK
40.68 MHz	1.8	m	:x:
433.92 MHz	17	cm	:x:
915 MHz	8.2	cm	
2450 MHz	3.06	cm	:white_check_mark:
5800 MHz	1.29	cm	:white_check_mark:
24.125 GHz	3.1	mm	:white_check_mark:
61.25 GHz	1.2	mm	:white_check_mark:

12.2.3 Range (Friis)

One of the worst questions a radio designer can get is "What is the range of your radio?", especially if the people asking are those that don't understand physics, or the real world. The answer to the question is incredibly complicated, as it depends on exactly what is between two devices talking.

If we assume, however, that there is only free space, and no real reflections from anywhere, then we can make an estimate of the range.

Assume no antenna gain, power density p at distance D is

$$p = \frac{P_{TX}}{4\pi D^2}$$

Assume reciever antenna has no gain, then the effective aperture is

$$A_e = \frac{\lambda^2}{4\pi}$$

Power received is then

$$P_{RX} = \frac{P_{TX}}{D^2}\left[\frac{\lambda}{4\pi}\right]^2$$

Or in terms of distance

$$D = 10^{\frac{P_{TX} - P_{RX} + 20log_{10}\left(\frac{c}{4\pi f}\right)}{20}}$$

12.2.4 Range (Free space)

If we take the ideal equation above, and use some realistic numbers for TX and RX power, we can estimate a range.

Assume TX = 0 dBm, assume RX sensitivity is -80 dBm

Freq	$20log_{10}(c/4\pi f)$ [dB]	D [m]	OK/NOK
915 MHz	-31.7	260.9	:white_check_mark:
2.45 GHz	**-40.2**	**97.4**	:white_check_mark:
5.80 GHz	-47.7	41.2	:white_check_mark:
24.12 GHz	-60.1	9.9	:x:
61.25 GHz	-68.2	3.9	:x:
160 GHz	-76.52	1.5	:x:

12.2.5 Range (Real world)

In the real world, however, the

path loss factor,

$$n \in [1.6, 6]$$

,

$$D = 10^{\frac{P_{TX} - P_{RX} + 20log_{10}\left(\frac{c}{4\pi f}\right)}{n \times 10}}$$

So the real world range of a radio can vary more than an order of magnitude. Still, 2.4 GHz seems like a good choice for a mouse.

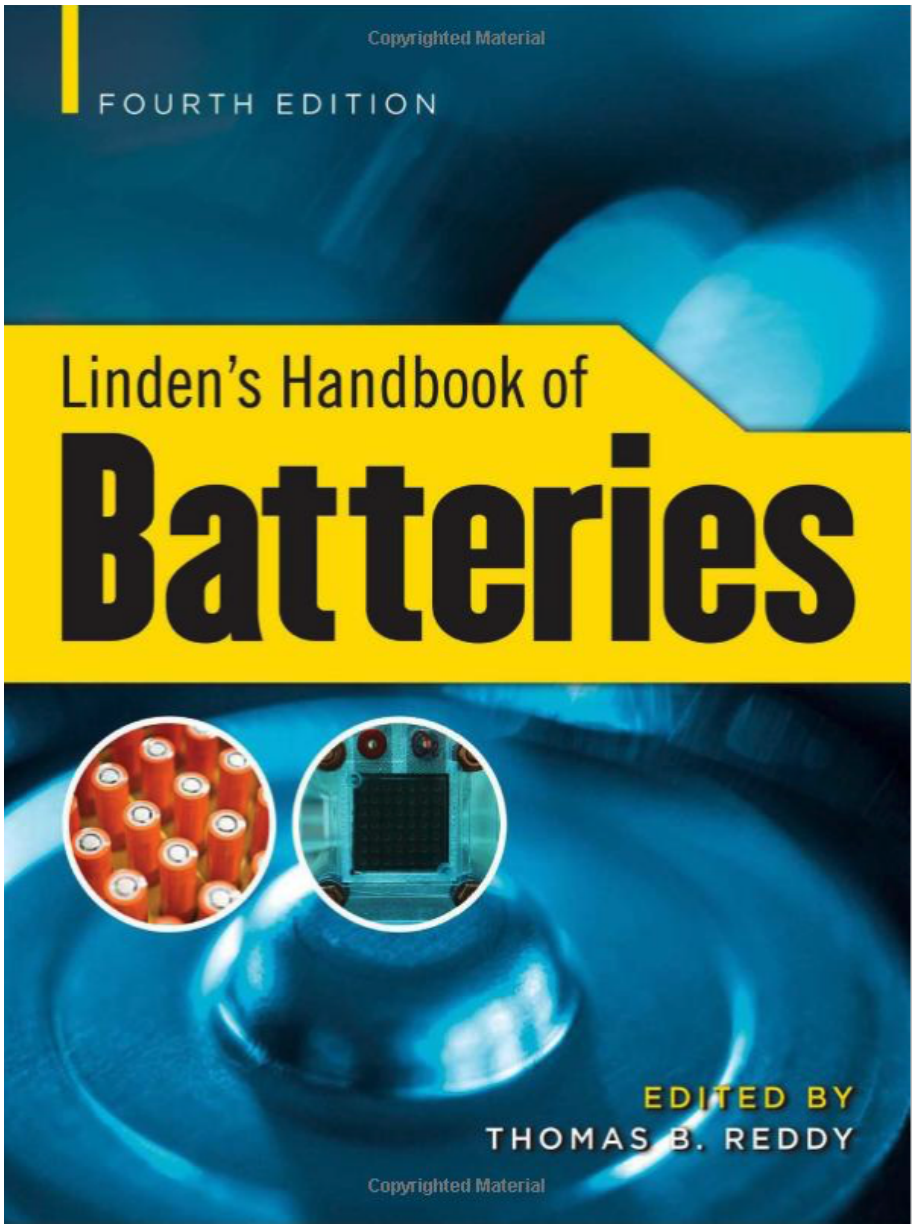
Freq	$20log_{10}(c/4\pi f)$ [dB]	D@n=2 [m]	D@n=6 [m]	OK/NOK
2.45 GHz	**-40.2**	**97.4**	**4.6**	:white_check_mark:
5.80 GHz	-47.7	41.2	3.45	:white_check_mark:
24.12 GHz	-60.1	9.9	2.1	:x:

12.3 Power supply

We could have a wired mouse for power, but that's boring. Why would we want a wired mouse to have wireless communication? It must be powered by a battery, but what type of battery?

There exists a bible of batteries, see picture below. It's worth a read if you want to dive deeper into chemistry and properties of primary (non-chargeable) and secondary (chargeable) cells.

12.3.1 Battery



Mouse is maybe AA, 3000 mAh

Cell	Chemistry	Voltage (V)	Capacity (Ah)	
AA	LiFeS2	1.0 - 1.8		3
2xAA	LiFeS2	2.0 - 3.6		3
AA	Zn/Alk/MnO2	0.8 - 1.6		3
2xAA	Zn/Alk/MnO2	1.6 - 3.2		3

12.4 Decisions we must make

Now we know that we need a 1 Mbps radio at 2.4 GHz that runs of a 1.0 V - 1.8 V or 2.0 V - 3.6 V supply.

Next we need to decide what modulation scheme we want for our light. How should we encode the bits onto the 2.4 GHz carrier wave?

12.4.1 Modulation scheme

People have been creative over the last 50 years in terms of encoding bits onto carriers. Below is a small excerpt of some common schemes.

Scheme	Acronym	Pro	Con
Binary phase shift keying	BPSK	Simple	Not constant envelope
Quadrature phase-shift keying	QPSK	2bits/symbol	Not constant envelope
Offset QPSK	OQPSK	2bits/symbol	Constant envelope with half-sine pulse shaping
Gaussian Frequency Shift Keying	GFSK	1 bit/symbol	Constant envelope
Quadrature amplitude modulation	QAM	$>$ 1024 bits/symbol	Really non-constant envelope

12.4.2 BPSK

In binary phase shift keying the 1 and 0 is encoded in the phase change. Change the phase 180 degrees and we've transitioned from a 0 to a 1. Do another 180 degrees and we're back to where we were.

It's common to show modulation schemes in a constellation diagram with the real axis and the complex axis. For the real light we send the phase and amplitude is usually real.

I say usually, because in quantum mechanics, and the time evolution of a particle, the amplitude of the wave function is actually a complex variable. As such, nature is actually complex at the most fundamental level.

But for now, let's keep it real in the real world.

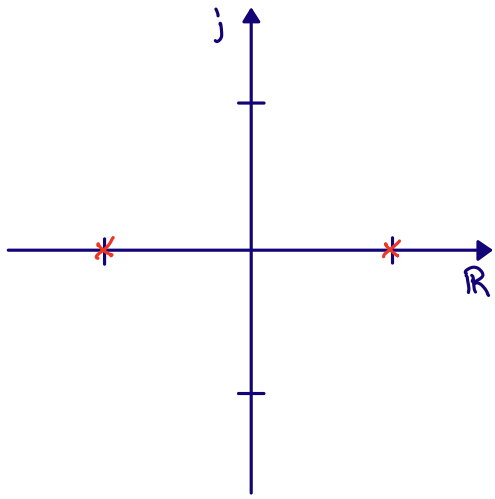
Still, the maths is much more elegant in the complex plane.

The equation for the unit circle is $y = e^{i(\omega t+\phi)}$ where ϕ is the phase, and ω is the angular frequency.

Imagine we spin a bike wheel around at a constant frequency (constant ω), on the bike wheel there is a red dot. If you keep your eyes open all the time, then the red dot would go round and round. But imagine that you only opened your eyes every second for a brief moment to see where the dot was. Sometimes it could be on the right side, sometimes on the left side. If our "eye opening rate", or your sample rate, matched how fast the "wheel rotator" changed the location of the dot, then you could receive information.

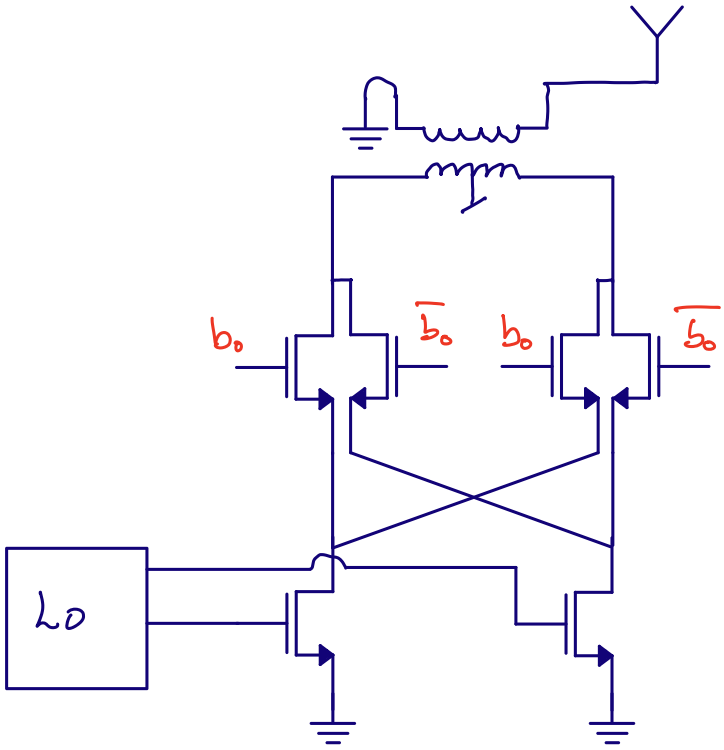
Now imagine you have a strobe light matched to the "normal" carrier frequency. If one rotation of the wheel matched the frequency of the strobe light, then the red dot would stay in exactly the same place. If the wheel rotation was slightly faster, then the red dot would move one way around the circle at every strobe. If the wheel rotation was slightly slower, the red dot would move the other way around the circle.

That's exactly how we can change the position in the constellation. We increase the carrier frequency for a bit to rotate 180 degrees, and we can decrease the frequency to go back 180 degrees. In this example the dot would move around the unit circle, and the amplitude of the carrier can stay constant.

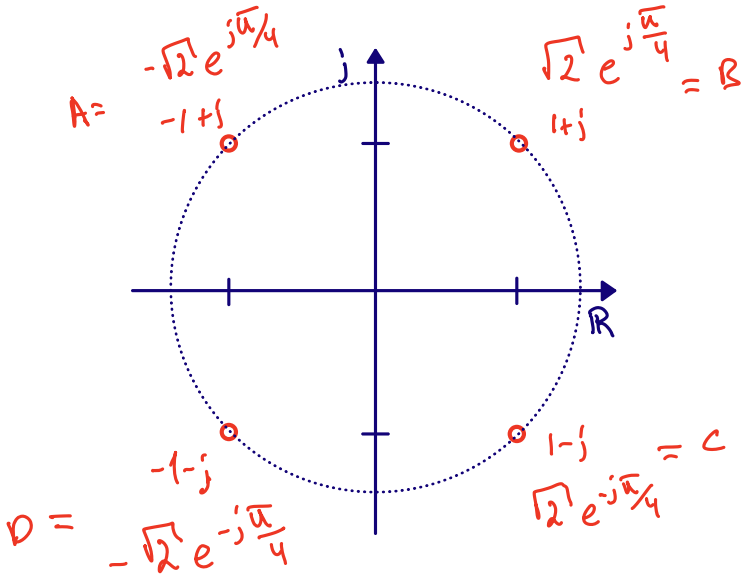


There is another way to change phase 180 degrees, and that's simply to swap the phase in the transmitter circuit. Imagine as below we have a local oscillator driving pseudo differential common source stages with switches on top. If we flip the switches we can change the phase 180 degrees pretty fast.

A challenge is, however, that the amplitude will change. In general, constant envelope (don't change amplitude) modulation is less bandwidth efficient (slower) than schemes that change both phase and amplitude.

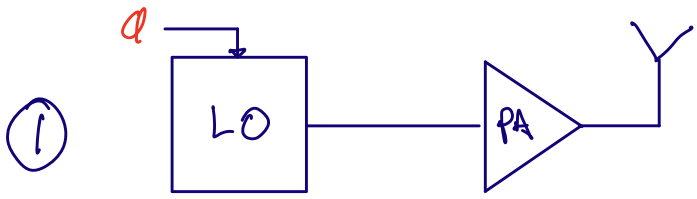


Standards like Zigbee used offset quadrature phase shift keying, with a constellation as shown below. With 4 points we can send 2 bits per symbol.

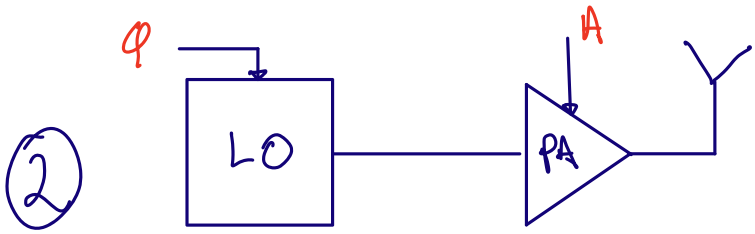


In ZigBee, or 802.15.4 as the standard is called, the phase changes is actually done with a constant envelope.

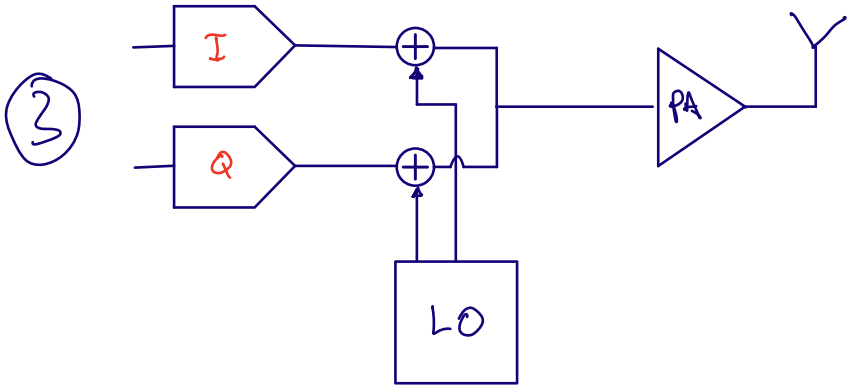
The nice thing about constant envelope is that the radio transmitter can be simple. We don't need to change the amplitude. If we have a PLL as a local oscillator, where we can change the phase (or frequency), then we only need a power amplifier before the antenna.



For phase and amplitude modulation, or complex transmitters, we need a way to change the amplitude and phase. What a shocker. There are two ways to do that. A polar architecture where phase change is done in the PLL, and amplitude in the power amplifier.

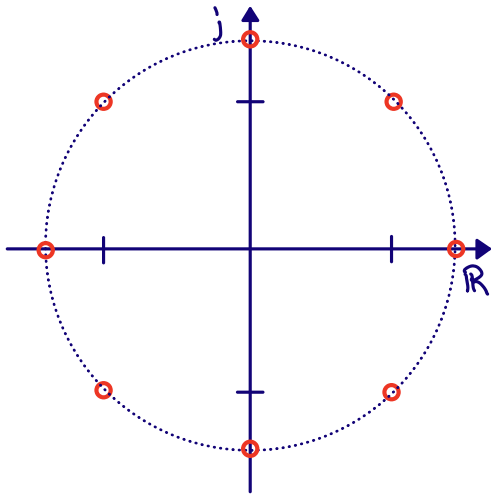


Or a Cartesian architecture where we make the in-phase component, and quadrature-phase components in digital, then use two digital to analog converters, and a set of complex mixers to encode onto the carrier. The power amplifier would not need to change the amplitude.



We can continue to add constellation points around the unit circle. Below we can see 8-PSK, where we can send 3-bits per symbol. Assuming we could shift position between the constellation points at a fixed rate, i.e 1 mega symbols per second. With 1-bit per symbol we'd get 1 Mbps. With 3-bits per symbol we'd get 3 Mbps.

We could add 16 points, 32 points and so on to the unit circle, however, there is always noise in the transmitter, which will create a cloud around each constellation point, and it's harder and harder to distinguish the points from each other.



Bluetooth Classic uses $\pi/4$-DQPSK and 8DPSK.

DPSK means differential phase shift keying. Think about DPSK like this. In the QPSK diagram above the symbols (00,01,10,11) are determined by the constellation point $1+j$, $1-j$ and so on. What would happen if the constellation rotated slowly? Would $1+j$ turn into $1-j$ at some point? That might screw up our decoding if the received constellation point was at $1+0j$, we would not know what it was.

If we encoded the symbols as a change in phase instead (differential), then it would not matter if the constellation rotated slowly. A change from $1+j$ to $1-j$ would still be 90 degrees.

Why would the constellation rotate you ask? Imagine the transmitter transmits at 2 400 000 000 Hz. How does our reciever generate the same frequency? We need a reference and a PLL. The crystal-oscillator reference has a variation of +-50 ppm, so $2.4e9 \times 50/1e6 = 120$ kHz.

Assume our receiver local oscillator was at 2 400 120 000 Hz. The transmitter sends 2 400 000 000 Hz + modulation. At the reciever we multiply with our local oscillator, and if you remember your math, multiplication of two sine creates a sum and a difference between the two frequencies. As such, the low frequency part (the difference between the frequencies) would be 120 kHz + modulation. As a result, our constellation would rotate 120 000 times per second. Assuming a symbol rate of 1MS/s our constellation would rotate roughly 1/10 of the way each symbol.

In DPSK the rotation is not that important. In PSK we have to measure the carrier offset, and continuously de-rotate the constellation.

Most radios will de-rotate somewhat based on the preamble, for example in Bluetooth Low Energy there is an initial 10101010 sequence that we can use to estimate the offset between TX and RX carriers, or the frequency offset.

The $\pi/4$ part of $\pi/4 - DQPSK$ just means we actively rotate the constellation 45 degrees every symbol, as a consequence, the amplitude never goes through the origin. In the transmitter circuit, it's difficult to turn the carrier off, so we try to avoid the zero point in the constellation.

GFSK modulation mode, whereas the subsequent synchronization sequence, payload, and trailer sequence are transmitted using the Enhanced Data Rate PSK modulation mode.

3.2.1 Modulation characteristics

During access code and packet header transmission the Basic Rate GFSK modulation mode shall be used. During the transmission of the synchronization sequence, payload, and trailer sequence a PSK type of modulation with a data rate of 2 Mb/s or optionally 3 Mb/s shall be used. The following subsections specify the PSK modulation for this transmission.

3.2.1.1 Modulation method overview

The PSK modulation format defined for the 2 Mb/s transmission shall be π/4 rotated differential encoded quaternary phase shift keying (π/4-DQPSK).

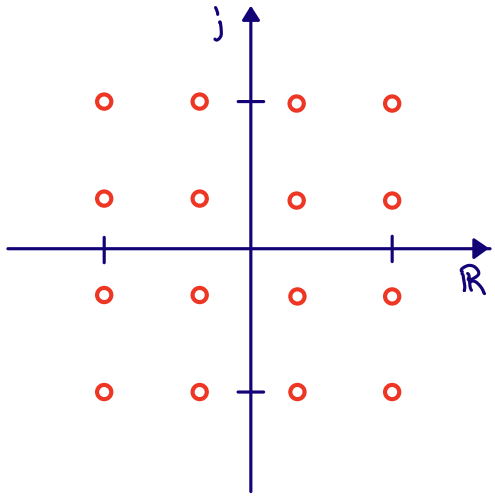
The PSK modulation format defined for the 3 Mb/s transmission shall be differential encoded 8-ary phase shift keying (8DPSK).

The modulation shall employ square-root raised cosine pulse shaping to generate the equivalent lowpass information-bearing signal v(t). The output of the transmitter shall be a bandpass signal that can be represented as

$$S(t) = Re\left[v(t)e^{j2\pi F_c t}\right] \quad \text{(EQ 1)}$$

I don't think 16PSK is that common, at 4-bits per symbol it's common to switch to Quadrature Amplitude Modulation (QAM), as shown below. The goal of QAM is to maximize the distance between each symbol. The challenge with QAM is the amplitude modulation. The modulation scheme is sensitive to variations in the transmitter amplitude. As such, more complex circuits than 8PSK could be necessary.

If you wanted to research "new fancy modulation schemes" I'd think about Sphere packing.



12.4.3 Single carrier, or multi carrier?

Assume we wanted to send 1024 Mbps over the air. We could choose a bandwidth of a about 1 GHz with 1-bit per symbol, or have a bandwidth of 1 MHz if we sent 1024 QAM at 1MS/s.

In both cases we get problems with the physical communication channel, the change in phase and amplitude affect what is received. For a 1 GHz bandwidth at 2.4 GHz carrier we'd have problems with the phase. At 1024 QAM we'd have problems with the amplitude.

Back in 1966 Orthogonal frequency division multiplexing was introduced to deal with the communication channel.

In OFDM we modulate a number of sub-carriers in the frequency space with our wanted modulation scheme (BPSK, PSK, QAM), then do an inverse fourier transform to get the time domain signal, mix on to the carrier, and transmit. At the reciever we take an FFT and do demodulation in the frequency space.

There are more details in OFDM than the simple statement above, but the details are just to fix challenges, such as "How do I recover the symbol timing? How do I correct for frequency offset? How do I ensure that my time domain signal terminates correctly for every FFT chunk"

The genius with OFDM is that we can pick a few of the sub-carriers to be pilot tones that carry no new information. If we knew exactly what was sent in phase and amplitude, then we could measure the phase and amplitude change due to the physical communication channel, and we could correct the frequency space before we tried to de-modulate.

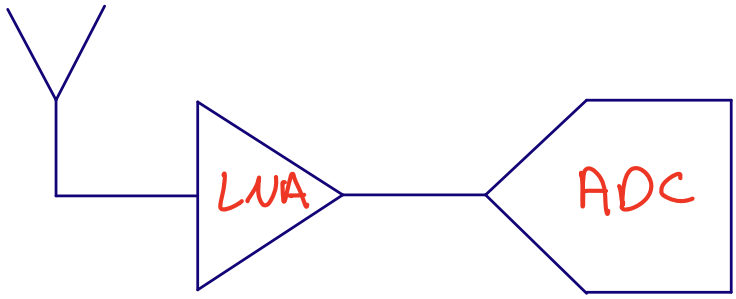
It's possible to do the same with single carrier modulation also. Imagine we made a 128-QAM modulation on a single carrier. As long as we constructed the time domain signal correctly (cyclic prefix to make the FFT work nicely, some preamble to measure the communication channel, then we could take an FFT at the reciever, correct the phase and amplitude, do an IFFT and demodulate the time-domain signal as normal.

In radio design there are so many choices it's easy to get lost.

12.4.4 Use a Software Defined Radio

For our mouse, what radio scheme should we choose? One common instances of "how to make a choice" in industry is "Delay the choice as long as possible so your sure the choice is right".

Maybe the best would be to integrated a software defined radio reciever? Something like the picture below, an antenna, low noise amplifier, and a analog-to-digital converter. That way we could support any transmitter. Fantastic idea, right?



Well, lets check if it's a good idea. We know we'll use 2.4 GHz, so we need about 2.5 GHz bandwidth, at least. We know we want good range, so maybe 100 dB dynamic range. In analog to digital converter design there are figure of merits, so we can actually compute a rough power consumption for such an ADC.

ADC FOM

$$= \frac{P}{2BW2^n}$$

State of the art FOM

$$\approx 5 \text{ fJ/step}$$

$$BW = 2.5 \text{ GHz}$$

$$DR = 100 \text{ dB} = (96 - 1.76)/6.02 \approx 16 \text{ bit}$$

$$P = 5 \text{ fF} \times 5 \text{ GHz} \times 2^{16} = 1.6 \text{ W}$$

At 1.6 W our mouse would only last for 2 hours. That's too short. Maybe we could use a WiFi 6 radio, those are good! #racetoidle

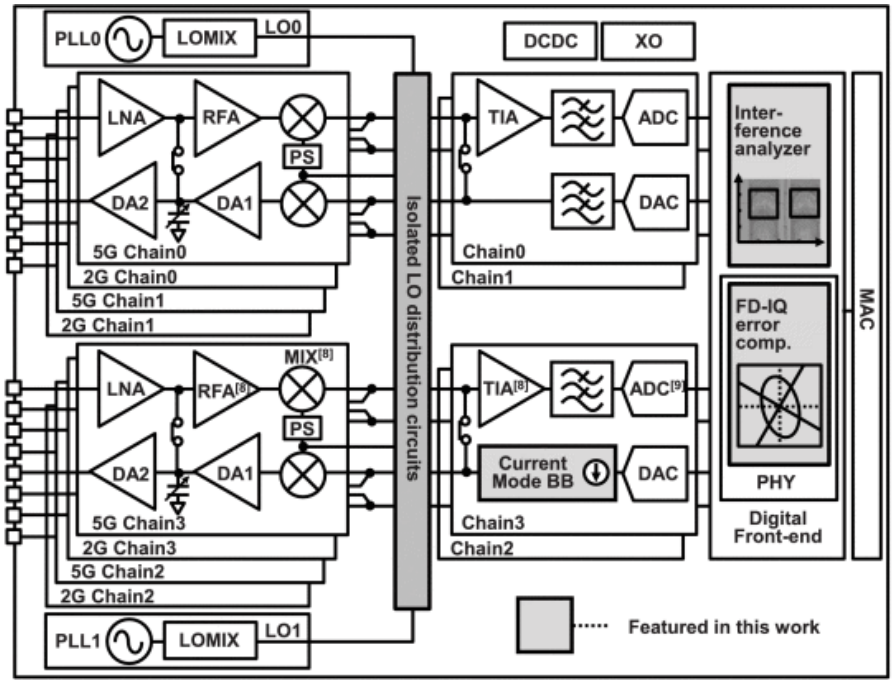
12.4.5 WiFi 6

It's fun sometimes to read really high end radio papers. The complexity of the radios are so high it's hard to understand how it could be made.

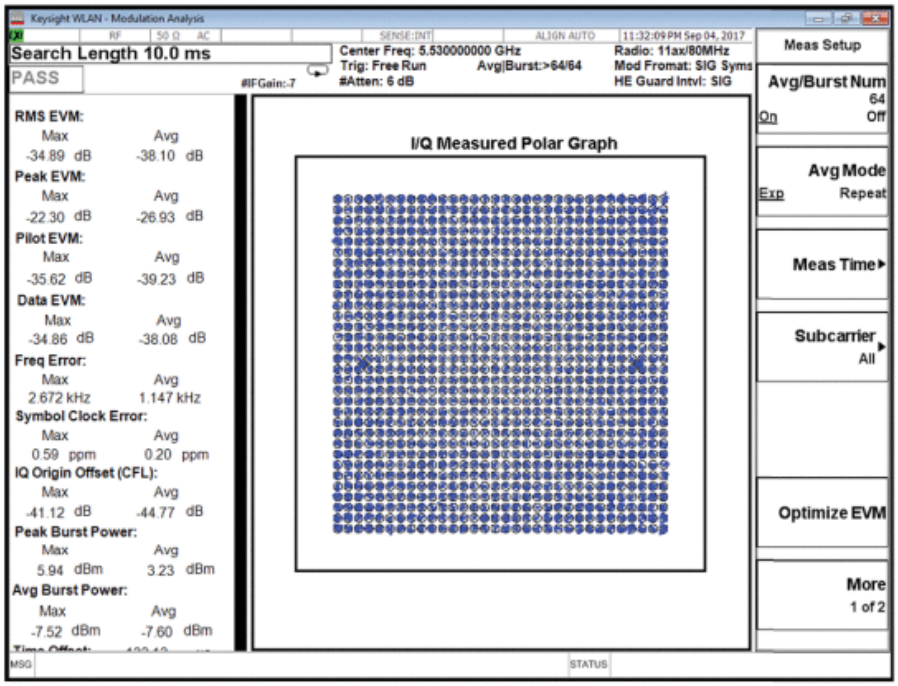
One example is the one below

An 802.11ax 4 × 4 High-Efficiency WLAN AP Transceiver SoC Supporting 1024-QAM With Frequency-Dependent IQ Calibration and Integrated Interference Analyzer

There are multiple PLLs, multiple recievers, multiple ADCs and dacs, with a bunch of calibration and compensation loops.



And an insane 1024 QAM constellation. Notice how small space there is between the points.



Again, though, the current consumption is a bit too high. Almost 1 Watt. Maybe something simpler is needed for our wireless mouse?

		This work	ISSCC2017[4]	JSSC2017[6]	CICC2015[3]	RFIC2014[5]	ISSCC2014[7]
WLAN standards		4x4 11abgn/ac/ax	4x4 11abgn/ac	2x2 11abgn/ac	1x1 11abgn/ac	2x2 11abgn/ac	3x3 11abgn/ac
Process [nm]		28	40	40	55	55	40
TX EVM [dB]	2.4G	-42.1(n,64QAM*3,-5dBm) -42.5(ax,40M,1KQAM*1,-5dBm)	NA	-40 (20M, Floor)	NA	-38(20M, Floor)	-41 (HT40, -5dBm)
	5G	-38.4(ac,80M,256QAM*2,-5dBm) -38.1(ax,80M,1KQAM*1,-5dBm)	-36.5(ac,80M,MCS9, Floor)	-38 (20M,Floor)	-37.8(80MHz, Floor)	-37.5(ac, 80M, Floor)	-37 (-5dBm)
RX sensitivity [dBm]	2.4G	-78.4(g,54M) -64.2(ax,40M,1KQAM*1)	-77(LG,54M)	-78.3(54Mbps)	-77.5(LG,54M)	-77.5(LG, 54M)	NA
	5G	-65.4(ac,80M,256QAM*2) -57.7(ax,80M,1KQAM*1)	-62(ac,80M,MCS9)	-66(MCS9)	-63.5(80M, MCS9)	-62.5 (ac, 80M, MCS9)	NA
RX NF [dB]	2.4G	2.9	N/A	2.9	3	3.8	3.0
	5G	3.2	N/A	4.5	3.7	4	4.3
RF power consumption [mW]	TX 2.4G	844(4SS+1LO, -5dBm)	3863 (4SS+1LO,21dBm)	1460 (2SS+1LO,20dBm)	705(1SS+1LO, 20.5dBm)	1588 (2SS+1LO, 20dBm)	1080*4 (3SS+1LO)
	TX 5G	832(4SS+2LO, -5dBm)	4164 (2SS+2LO,22dBm)	1750 (2SS+1LO,18dBm)	996 (2SS+1LO, 18.5dBm)	1722 (2SS+1LO, 17.5dBm	1520*4 (3SS+1LO)
	RX 2.4G	354(4SS+1LO)	297(4SS+1LO)	179 (2SS+1LO)	84 (1SS+1LO)	303 (2SS+1LO)	1170*4 (3SS+1LO)
	RX 5G	447(4SS+2LO)	474(2SS+2LO)	243 (2SS+1LO)	156 (1SS+1LO)	317 (2SS+1LO)	2080*4 (3SS+1LO)
Image rejection ratio after cal. [dB]		-53(RX, Ave. over 80M) -58(RX, at 5MHz) -61(TX, Ave. over 80M) -64(TX, at 5MHz)	-61(TX,at 5MHz)	NA	-45(RX)	-45(RX)	NA
RF chip area[mm^2]		12.0	11.4	8.6	3.4*6	7.7	21.5*5
1K QAM		Yes	No	No	No	No	No
FD-IQ amp. mismatch calibration		Yes	No	No	No	No	No
Non-contiguous		Yes	Yes	No	No	No	No
Integrated interference analyzer		Yes	No	No	No	No	No

*1 Modulation and coding scheme is MCS11 *2 Modulation and coding scheme is MCS9 *3 Modulation and coding scheme is MCS7 *4 SoC power *5 SoC area *6 Including BT RF area

12.5 Bluetooth

Bluetooth was made to be a "simple" standard and was introduced in 1998. The standard has continued to develop, with Low Energy introduced in 2010. The latest planned changes can be seen at Specifications in Development.

12.5.1 Bluetooth Basic Rate/Extended Data rate

- 2.400 GHz to 2.4835 GHz
- 1 MHz channel spacing
- 78 Channels
- Up to 20 dBm
- Minimum -70 dBm sensitivity (1 Mbps)
- 1 MHz GFSK (1 Mbps), pi/4-DQPSK (2 Mbps), 8DPSK (3 Mbps)

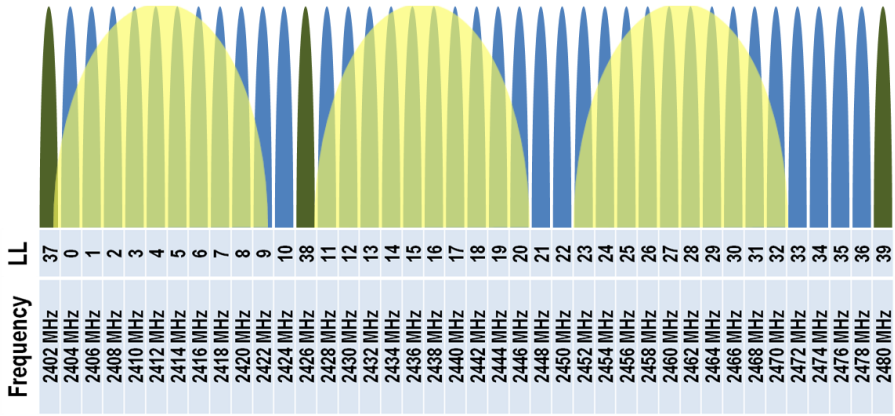
You'll find BR/EDR in most audio equipment, cars and legacy devices. For new devices (also audio), there is now a transition to Bluetooth Low Energy.

12.5.2 Bluetooth Low Energy

- 2.400 GHz to 2.480 GHz
- 2 MHz channel spacing
- 40 Channels (3 primary advertising channels)
- Up to 20 dBm
- Minimum -70 dBm sensitivity (1 Mbps)
- 1 MHz GFSK (1 Mbps, 500 kbps, 125 kbps), 2 MHz GFSK (2 Mbps)

Below are the Bluetooth LE channels. The green are the advertiser channels, the blue are the data channels, and the yellow is the WiFi channels.

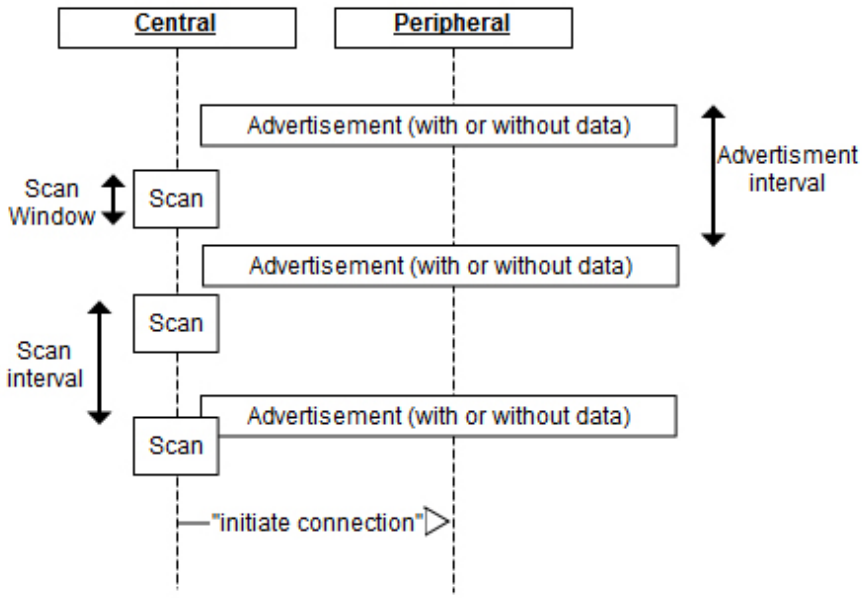
The advertiser channels have been intentionally placed where there is space between the WiFi channels to decrease the probability of collisions.



Any Bluetooth LE peripheral will advertise it's presence, it will wake up once in a while (every few hundred milliseconds, to seconds) and transmit a short "I'm here" packet. After transmitting it will wait a bit in receive to see if anyone responds.

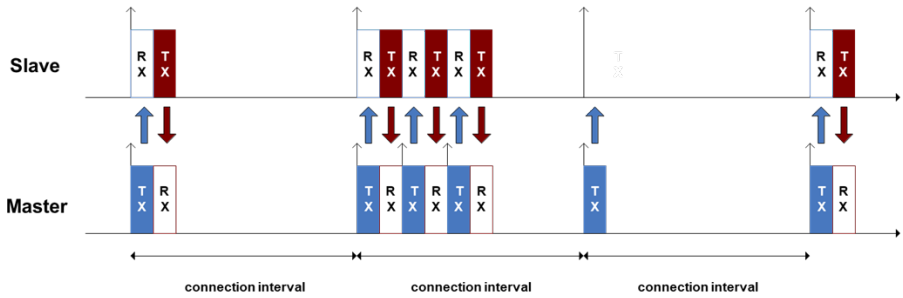
A Bluetooth LE central will camp in receive on a advertiser channel and look for these short messages from peripherals. If one is observed, the Central may choose to respond.

Take any spectrum analyzer anywhere, and you'll see traffic on 2402, 2426, and 2480 MHz.



In a connection a central and peripheral (the master/slave names below have been removed from the spec, that was a fun update to a 3500 page document) will have agreed on an interval to talk. Every "connection interval" they will transmit and receive data. The connection interval is tunable from 7.5 ms to seconds.

Bluetooth LE is the perfect standard for wireless mice.

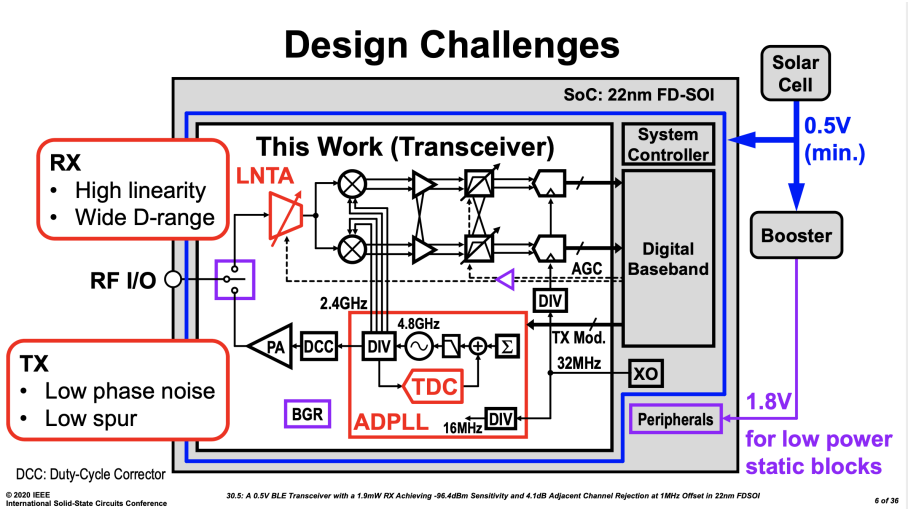


For further information Building a Bluetooth application on nRF Connect SDK

12.6 Algorithm to design state-of-the-art LE radio

- Find most recent digest from International Solid State Circuit Conference (ISSCC)
- Find Bluetooth low energy papers
- Pick the best blocks from each paper

A typical Bluetooth radio may look something like the picture below.



In the typical radio we'll need the blocks below. I've added a column for how many people I would want if I was to lead development of a new radio.

Blocks	Key parameter	Architecture	Complexity (nr people)
Antenna	Gain, impedance	??	<1
RF match	loss, input impedance	PI-match?	<1
Low noise amp	NF, current, linearity	LNTA	1
Mixer	NF, current, linearity	Passive	1
Anti-alias filter	NF, linearity	TIA + AFIR	1
ADC	Sample rate, dynamic range, linearity	NS-SAR	1 - 2
PLL	Freq accuracy, phase noise, current	AD-PLL	2-3
Baseband	Eb/N0, gate count, current.	SystemVerilog	> 10

12.6.1 LNTA

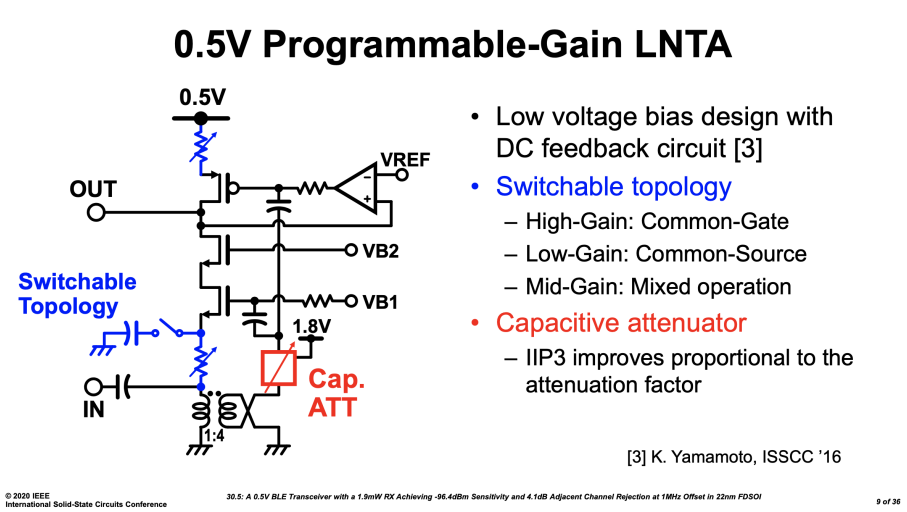
The first thing that must happen in the radio is to amplify the noise as early as possible. Any circuit has inherent noise, be it thermal-, flicker-, burst-, or shot-noise. The earlier we can amplify the input noise, the less contribution there will be from the radio circuits.

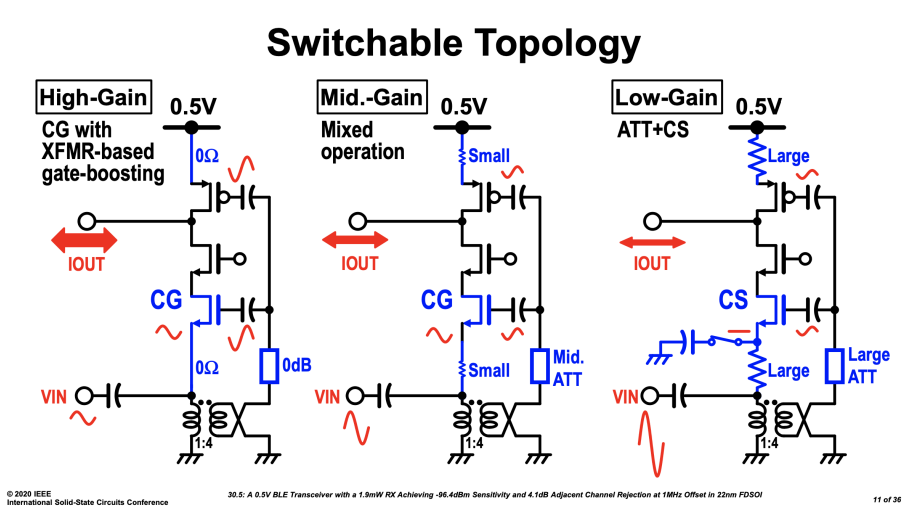
The challenges in the low noise amplifier is to provide the right gain. If there is a strong input signal, then reduce the gain. If there is a low input signal, then increase the gain.

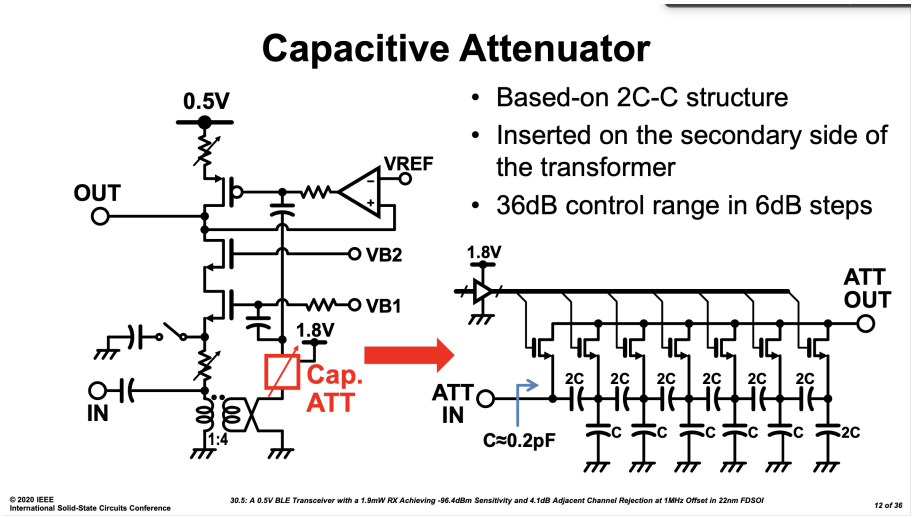
One way to implement variable gain is to reconfigure the LNA.

The example below combines a NMOS common gate, common source and a PMOS common source. Depending on the required gain there is also attenuation introduced.

One second challenge in the LNA is that the gain must be tuned without change to the input impedance. At RF it matters that the impedances are matched, otherwise power would be lost to reflections.



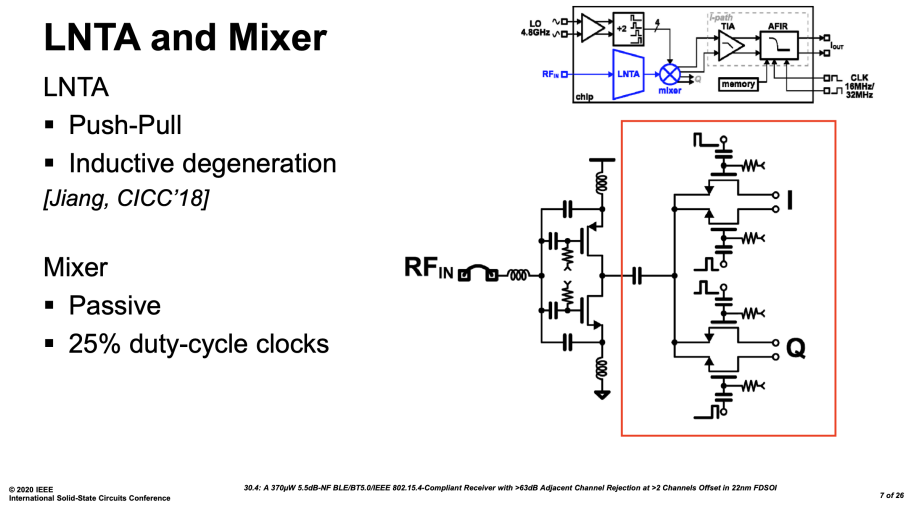




12.6.2 MIXER

In the mixer we multiply the input signal with our local oscillator. Most often a complex mixer is used. There is nothing complex about complex signal processing, just read Complex signal processing is not complex.

In order to reduce power, it's most common with a passive mixer as shown below. A passive mixer is just MOS that we turn on and off with 25% duty-cycle.



To generate the quadrature and in-phase clock signals, which must be 90 degrees phase offset, it's common to generate twice the frequency in the local oscillator (4.8 GHz), and then divide down to 4 2.4 GHz clock signals.

I liked the "Windmill" divider in the picture below (I have not tried it though, but it looked cool).

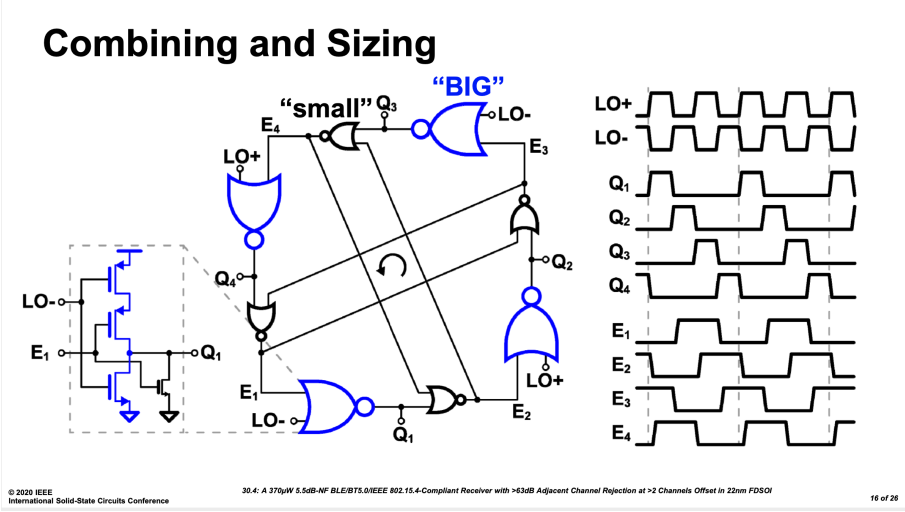
If the LO is the same as the carrier, then the modulation signal will be at DC, often called direct conversion.

The challenge at DC is that there is flicker noise, offset, and burst noise. The modulation type, however, can impact whether low frequency noise is an issue. In OFDM we can choose to skip the sub-carriers around 0 Hz, and direct conversion works well. An advantage with direct conversion is that there is no "image frequency" and we can use the full complex bandwidth.

For FSK and direct conversion the low frequency noise can cause issues, as such, it's common to offset the LO from the transmitted signal, for example 4 MHz offset. The low frequency noise problem disappears, however, we now have a challenge with the image frequency (-4 MHz) that must be rejected, and we need an increased bandwidth.

There is no "one correct choice", there are trade-offs that both ways. KISS (Keep It Simple Stupid) is one of my guiding principles when working on radio architecture.

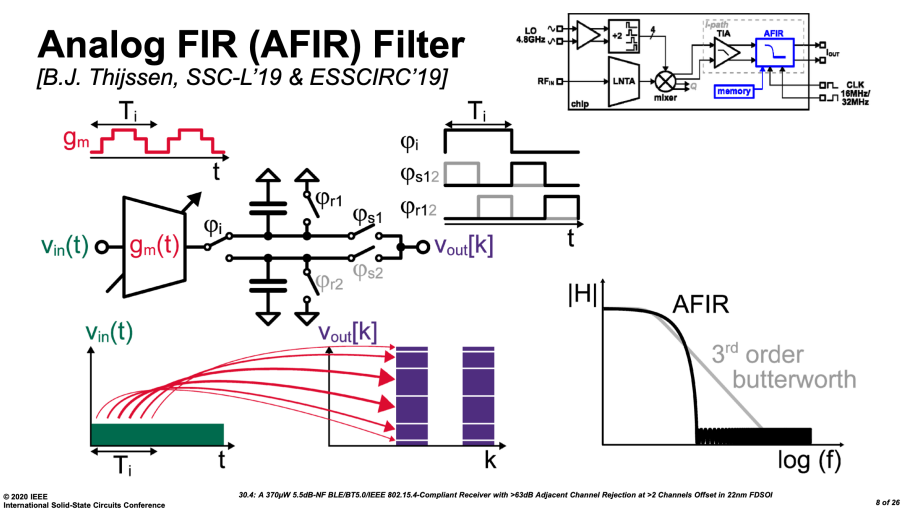
These days most de-modulation happens in digital, and we need to convert the analog signal to digital, but first AAF.

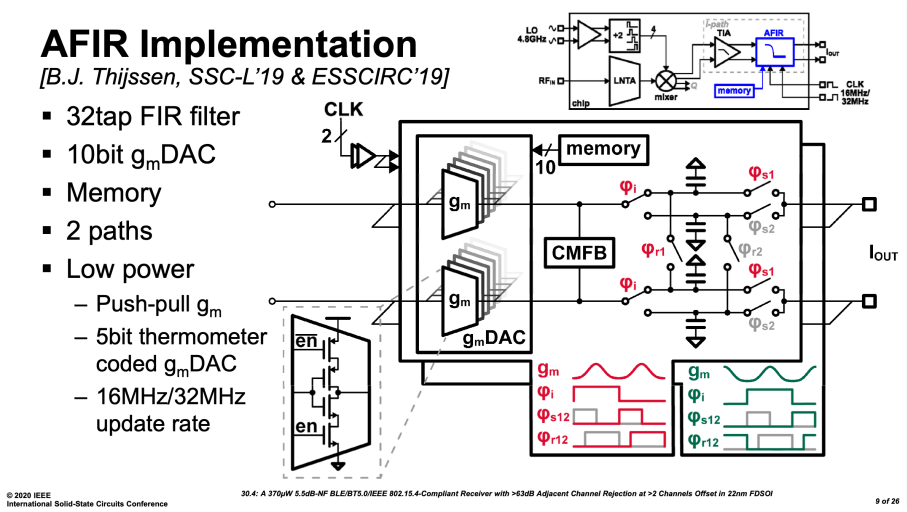


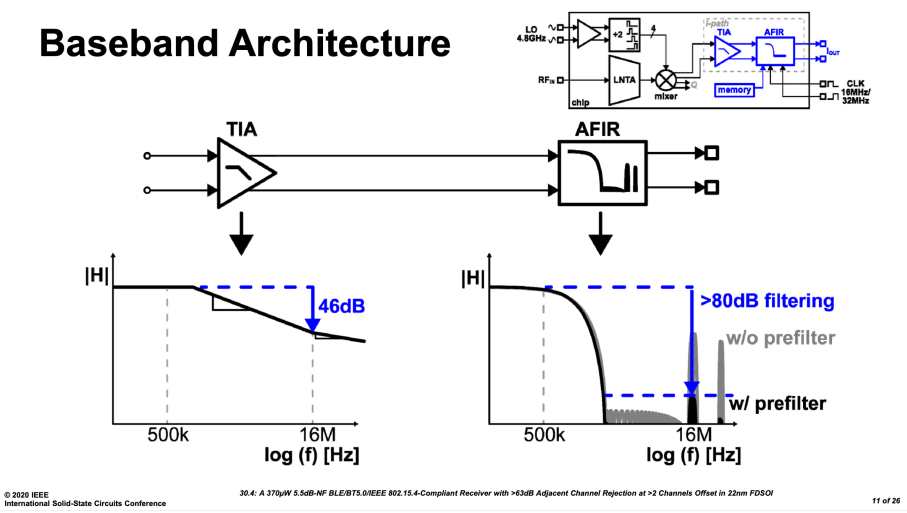
12.6.3 AAF

The anti alias filter rejects frequencies that can fold into the band of interest due to sampling. Below is a fancy anti-alias filter, but quite often, simple active-RC filters are perfect.

We often need gain in the AAF, as the LNA does not have sufficient gain for the weakest signals. -100 dBm in 50 ohm is 6.2 nV RMS, while input range of an ADC may be 1 V. Assume we place the lowest input signal at 0.1 V, so we need a voltage gain of $20\log(0.1/6.2e-9) = 76$dB in the reciever.







12.6.4 ADC

Aaah, ADCs, an IP close to my heart. I did my Ph.d and Post-Doc on ADCs, and the Ph.D students I've co-supervised have worked on ADCs.

At NTNU there have been multiple students through the years that have made world-class ADCs, and there's still students at NTNU working on state-of-the-art ADCs.

These days, a good option is a SAR, or a Noise-Shaped SAR.

If I were to pick, I'd make something like A 68 dB SNDR Compiled Noise-Shaping SAR ADC With On-Chip CDAC Calibration.

A 68 dB SNDR Compiled Noise-Shaping SAR ADC With On-Chip CDAC Calibration

Harald Garvik, Carsten Wulff and Trond Ytterdal
Department of Electronic Systems
Norwegian University of Science and Technology (NTNU), Trondheim, Norway
Email: harald.garvik@ntnu.no

***Abstract*—This paper presents a noise-shaping SAR ADC with an on-chip, foreground capacitive DAC calibration system. At start-up, the ADC uses the smallest DAC capacitors to measure and digitize the errors of the largest ones. A synthesized digital module accumulates the noise-shaped measurements, computes calibration coefficients, and corrects ADC codes at run-time. The loop filter implements two optimal zeros and two poles, and achieves 27.8 dB in-band attenuation at an oversampling rate of 4. The prototype implemented in 28 nm FDSOI achieves 68.2 dB SNDR at 5 MHz bandwidth while using 108.7 µW. The Walden figure-of-merit is 5.2 fJ/conv.-step. The layout of the ADC is compiled from a netlist, a rule file, and an object definition file.**

***Index Terms*—Noise-shaping, SAR ADC, CDAC calibration, analog layout synthesis.**

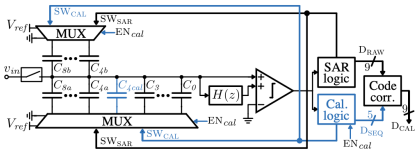


Fig. 1. Proposed noise-shaping SAR architecture. Blue blocks and paths are only active in calibration mode.

Or if I did not need high resolution, I'd choose my trusty A Compiled 9-bit 20-MS/s 3.5-fJ/conv.step SAR ADC in 28-nm FDSOI for Bluetooth Low Energy Receivers. That particular ADC have been ported to multiple technologies (22 nm FDSOI, 22 nm, 28 nm, 55 nm, 65 nm and 130nm), and can be found at sun_sar9b_sky130nm.

A Compiled 9-bit 20-MS/s 3.5-fJ/conv.step SAR ADC in 28-nm FDSOI for Bluetooth Low Energy Receivers

Carsten Wulff, *Member, IEEE*, and Trond Ytterdal, *Senior Member, IEEE*

***Abstract*— This paper presents a low-power 9-bit compiled successive-approximation register (SAR) analog-to-digital converter (ADC) for Bluetooth low energy receivers. The ADC is compiled from a SPICE netlist, a technology rule file, and an object definition file into a design rule check and layout versus schematic clean layout and schematic in 28-nm FDSOI. The compiled SAR ADC reduces the design time necessary to port to a new technology, and to demonstrate technology porting the same SAR ADC architecture is compiled in 28-nm FDSOI with Input/Output (IO) transistors. This paper also includes a comparator clock generation loop that uses the bottom plate of the capacitive digital-to-analog converter. The proposed compiled core transistor SAR ADC achieves the state-of-the-art Figure of Merit (FoM) of 2.7 fJ/conv.step at 2 MS/s, and 3.5 fJ/conv.step at 20 MS/s with an area of 0.00312 mm^2.**

***Index Terms*— Analog layout, analog layout synthesis, analog-to-digital conversion, analog-to-digital converter (ADC), Bluetooth low energy, Fully-depleted Silicon-on-insulator (FDSOI), low power, successive approximation.**

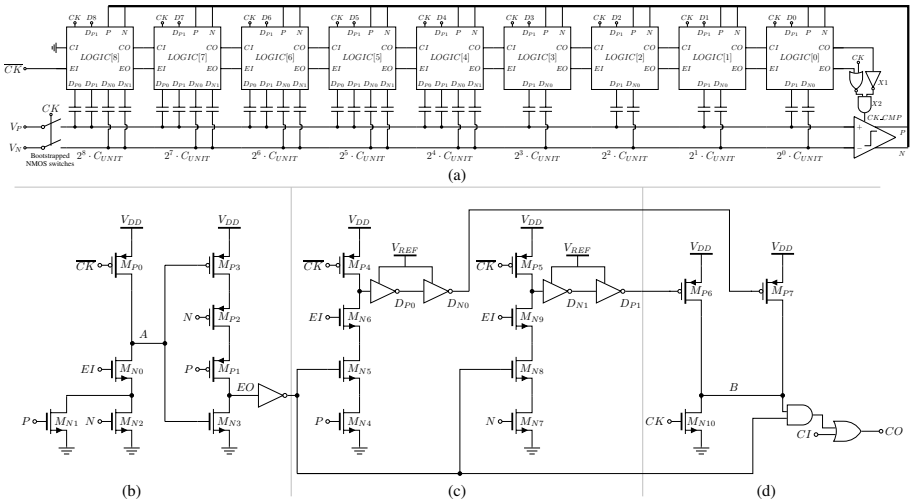
layout. For example, the schematic does not contain placement information for layout, since the optimum layout placement might be different from optimal schematic placement.

Analog layout generation has a long history with works from the previous century [2], [3], but the state-of-the-art analog layout generation, as reviewed in [4], is not widely adopted. More promising research avenues avoid the challenge of analog layout generation from schematics, by not having a schematic. Recently, ADCs have been compiled in a digital flow [5], [6], and although the Effective Number of Bits (ENOB) was less than 6 bit, it is an interesting approach. A similar approach has been used successfully for all-digital Phase Locked Loops [7].

This paper, first introduced at a conference [8], presents a method where the layout is not generated based on drawn schematics. The ADC is described using an approach borrowed from object oriented programming. A custom compiler

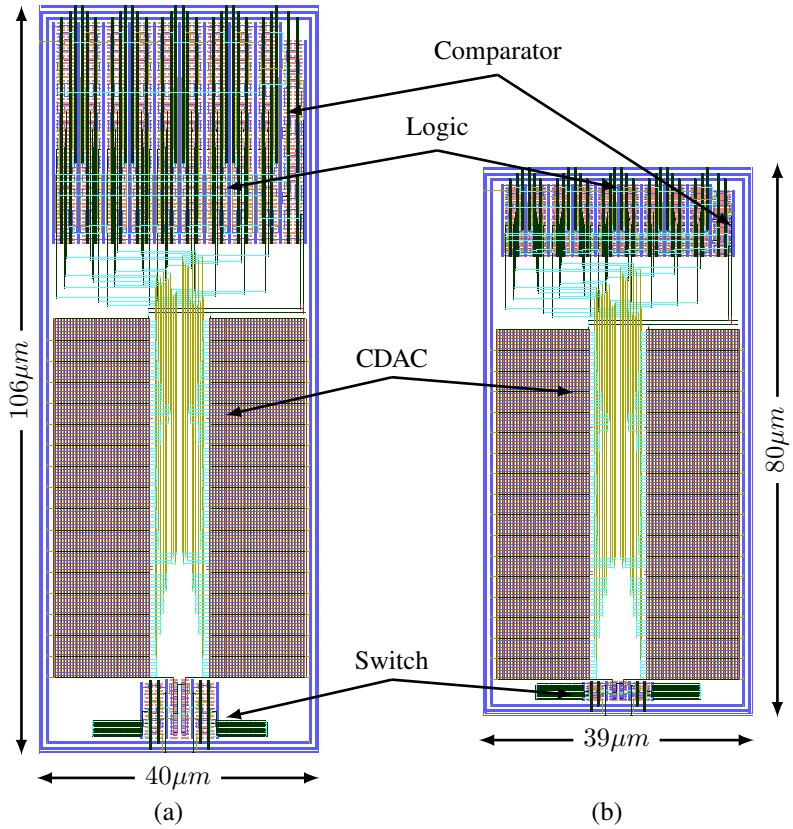
The main selling point of the ADC was that it's compiled from a JSON file, a SPICE file and a technology file into a DRC/LVS clean layout.

I also included a few circuit improvements. The bottom plate of the SAR capacitor is in the clock loop for the comparator (DN0, DP1 below), as such, the delay of the comparator automatically adjusts with capacitance corner, so it's more robust over corners



The compiled nature also made it possible to quickly change the transistor technology. Below is a picture with 180 nm FDSOI transistors on the left, and 28 nm FDSOI transistors on the right.

I detest doing anything twice, so I love the fact that I never have to re-draw that ADC again. I just fix the technology file (and maybe some tweaks to the other files), and I have a completed ADC.



12.6.5 AD-PLL

The phase locked loop is the heart of the radio, and it's probably the most difficult part to make. Depends a bit on technology, but these days, All Digital PLLs are cool.

You can spend your life on PLLs.

#Phase Locked loops

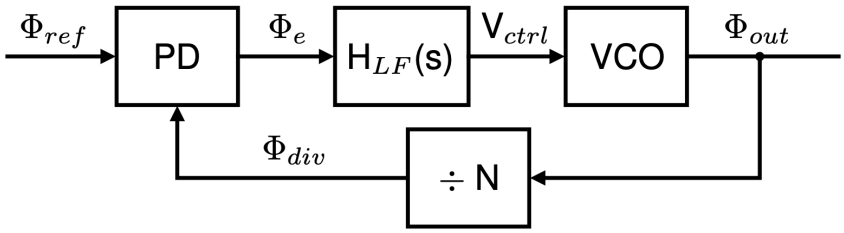
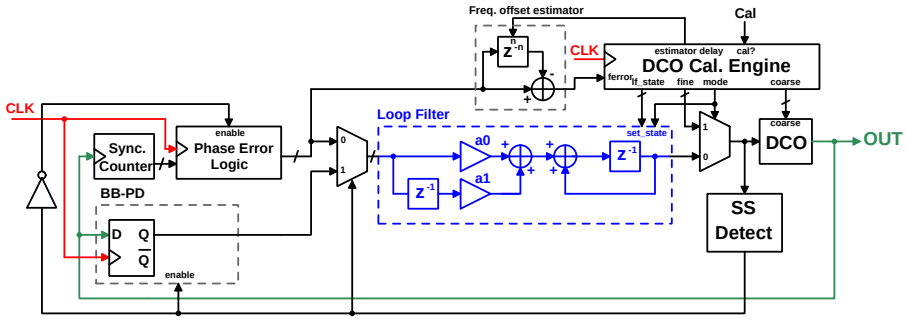


Figure 2: Basic PLL.

- Read Razavi's PLL book

AD-PLL with Bang-Bang phase detector for steady-state



12.6.6 Baseband

Once the signal has been converted to digital, then the de-modulation, and signal fixing start. That's for another course, but there are interesting challenges.

Baseband block	Why
Mixer?	If we're using low intermediate frequency to avoid DC offset problems and flicker noise
Channel filters?	If the AAF is insufficient for adjacent channel
Power detection	To be able to control the gain of the radio
Phase extraction	Assuming we're using FSK
Timing recovery	Figure out when to slice the symbol
Bit detection	single slice, multi-bit slice, correlators etc
Address detection	Is the packet for us?
Header detection	What does the packet contain
CRC	Does the packet have bit errors
Payload de-crypt	Most links are encrypted by AES
Memory access	Payload need to be stored until CPU can do something

12.7 What do we really want, in the end?

The reciever part can be summed up in one equation for the sensitivity. The noise in a certain bandwidth. The Noise Figure of the analog reciever. The Energy per bit over Noise of the de-modulator.

$$P_{RX_{sens}} = -174dBm + 10 \times log10(DR) + NF + Eb/N0$$

for example, for nRF5340

$$P_{RX_{sens}} + 174 - 60 = NF + Eb/N0 = 17dB$$

- *Bluetooth*® 5.1, IEEE 802.15.4-2006, 2.4 GHz transceiver
  - -97.5 dBm sensitivity in 1 Mbps Bluetooth low energy mode
  - -20 to +3 dBm configurable TX power
  - On-air compatible with nRF52, nRF51, nRF24L, and nRF24AP Series
  - Supported data rates:
    - *Bluetooth* 5.1: 2 Mbps, 1 Mbps, 500 kbps, and 125 kbps
    - IEEE 802.15.4-2006: 250 kbps
    - Proprietary 2.4 GHz: 2 Mbps, 1 Mbps
  - Single-ended antenna output (on-chip balun)
  - 128-bit AES/ECB/CCM/AAR co-processor (on-the-fly packet encryption)
  - 3.2 mA run current in TX (0 dBm)
  - 2.6 mA run current in RX
  - RSSI (1 dB resolution)

In the block diagram of the device the radio might be a small box, and the person using the radio might not realize how complex the radio actually is.

I hope you understand now that it's actually complicated.

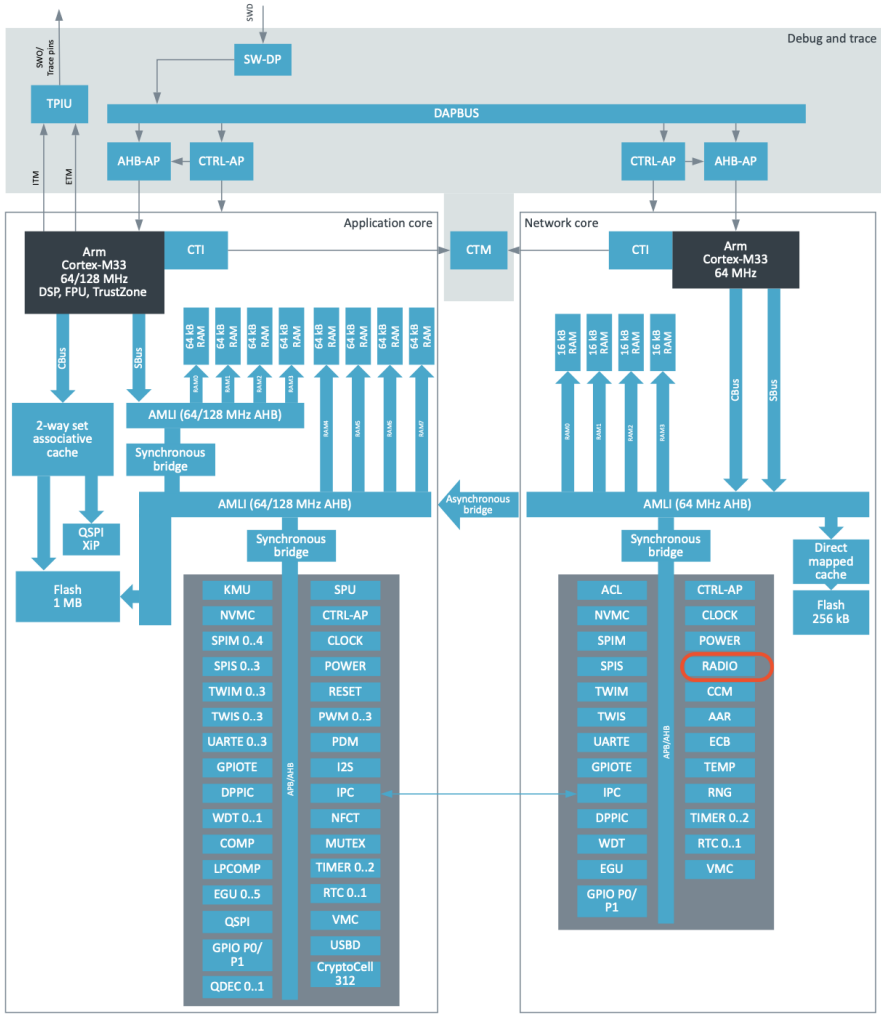


Figure 1: Simplified block diagram

12.8 References

"A 0.5V BLE Transceiver with a 1.9mW RX Achieving -96.4dBm Sensitivity and 4.1dB Adjacent Channel Rejection at 1MHz Offset in 22nm FDSOI", M. Tamura, Sony Semiconductor Solutions, Atsugi, Japan, 30.5, ISSCC 2020

"A 370uW 5.5dB-NF BLE/BT5.0/IEEE 802.15.4-Compliant Receiver with >63dB Adjacent Channel Rejection at >2 Channels Offset in 22nm FDSOI", B. J. Thijssen, University of Twente, Enschede, The Netherlands

"A 68 dB SNDR Compiled Noise-Shaping SAR ADC With On-Chip CDAC Calibration", H. Garvik, C. Wulff, T. Ytterdal

"A Compiled 9-bit 20-MS/s 3.5-fJ/conv.step SAR ADC in 28-nm FDSOI for Bluetooth Low Energy Recievers", C. Wulff, T. Ytterdal

Cole Nielsen, https://github.com/nielscol/thesis_presentations

"Python Framework for Design and Simulation of Integer-N ADPLLs", Cole Nielsen, https://github.com/nielscol/tfe4580-report/blob/master/report.pdf

Design of CMOS Phase-Locked Loops, Behzad Razavi, University of California, Los Angeles

Chapter 13

Analog SystemVerilog

Design of integrated circuits is split in two, analog design, and digital design.

Digital design is highly automated. The digital functions are coded in SystemVerilog (yes, I know there are others, but don't use those), translated into a gate level netlist, and automatically generated layout. Not everything is push-button automation, but most is.

Analog design, however, is manual work. We draw schematic, simulation with a mathematical model of the real world, draw the analog layout needed for the foundries to make the circuit, verify that we drew the schematic and layout the same, extract parasitics, simulate again, and in the end get a GDSII file.

When we mix analog and digital designs, we have two choices, analog on top, or digital on top.

In analog on top we take the digital IP, and do the top level layout by hand in analog tools.

In digital on top we include the analog IPs in the SystemVerilog, and allow the digital tools to do the layout. The digital layout is still orchestrated by people.

Which strategy is chosen depends on the complexity of the integrated circuit. For medium to low level of complexity, analog on top is fine. For high complexity ICs, then digital on top is the way to go.

Below is a description of the open source digital-on-top flow. The analog is included into GDSII at the OpenRoad stage of the flow.

The GDSII is not sufficient to integrate the analog IP. The digital needs to know how the analog works, what capacitance is on every digital input, the propagation delay for digital input to digital outputs , the relation between digital outputs and clock inputs, and the possible load on digital outputs.

The details on timing and capacitance is covered in a Liberty file. The behavior, or function of the analog circuit must be described in a SystemVerilog file.

But how do we describe an analog function in SystemVerilog? SystemVerilog is simulated in an digital simulator.

media/dig$_d$*es.svg*

13.1 Digital simulation

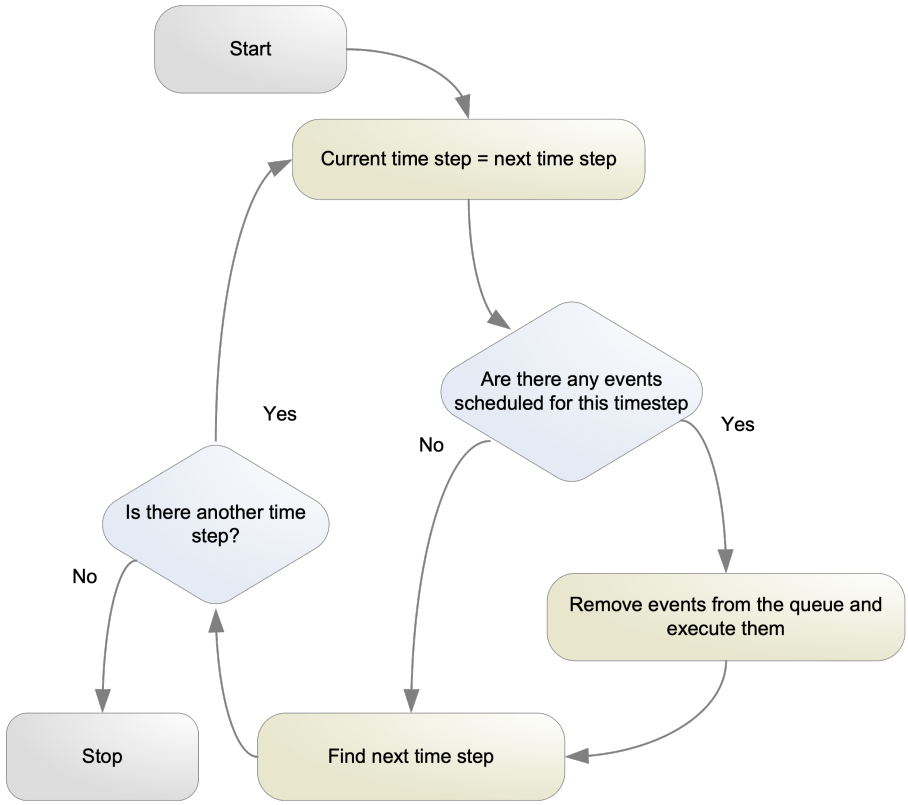
Conceptually, the digital simulator is easy.

- The order of execution of events at the same time-step do not matter

- The system is causal. Changes in the future do not affect signals in the past or the now

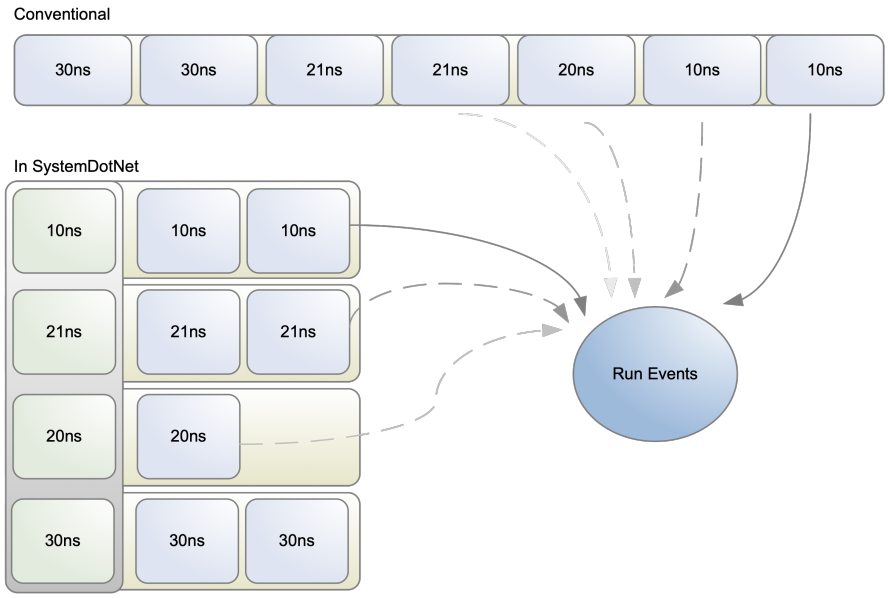
In a digital simulator there will be an event queue, see below. From start, set the current time step equals to the next time step. Check if there are any events scheduled for the time step. Assume that execution of events will add new time steps. Check if there is another time step, and repeat.

Since the digital simulator only acts when something is supposed to be done, they are inherently fast, and can handle complex systems.



It's a fun exercise to make a digital simulator. On my Ph.D I wanted to model ADCs, and first I had a look at SystemC, however, I disliked C++, so I made SystemDotNet

In SystemDotNet I implemented the event queue as a hash table, so it ran a bit faster. See below.



13.1.0.1 Digital Simulators

There are both commercial an open source tools for digital simulation. If you've never used a digital simulator, then I'd recommend you start with iverilog. I've made some examples at dicex.

Commercial

- Cadence Excelium
- Siemens Questa
- Synopsys VCS

Open Source - iverilog/vpp - Verilator - SystemDotNet

13.1.0.2 Counter

Below is an example of a counter in SystemVerilog. The code can be found at counter_sv.

In the always_comb section we code what will become the combinatorial logic. In the always_ff section we code what will become our registers.

```
module counter(
               output logic [WIDTH-1:0] out,
               input logic              clk,
               input logic              reset
               );

   parameter WIDTH = 8;

   logic [WIDTH-1:0]                    count;
   always_comb begin
      count = out + 1;
   end
```

```
  always_ff @(posedge clk or posedge reset) begin
    if (reset)
      out <= 0;
    else
      out <= count;
  end

endmodule // counter
```

In the context of a digital simulator, we can think through how the event queue will look.

When the clk or reset changes from zero to 1, then schedule an event where if the reset is 1, then out will be zero in the next time step. If reset is 0, then out will be count in the next time step.

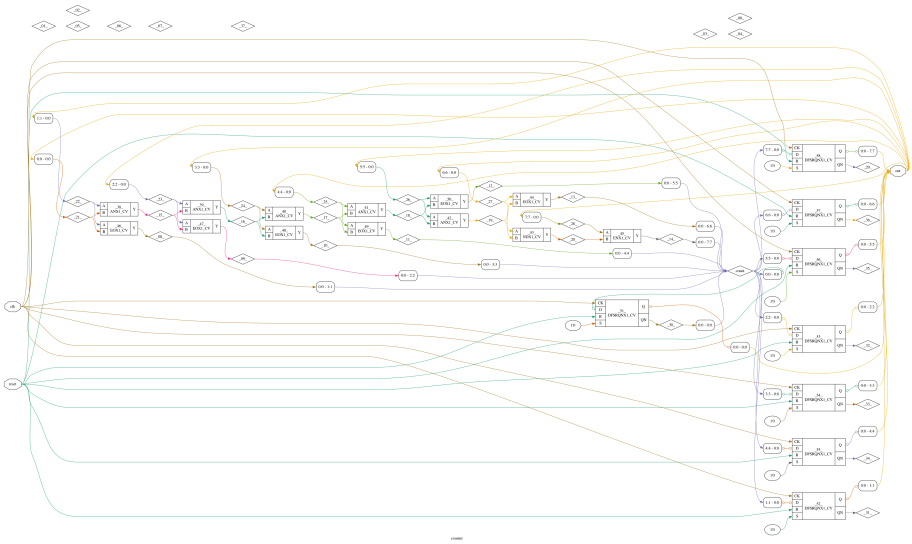
In a time-step where out changes, then schedule an event to set count to out plus one. As such, each positive edge of the clock at least 2 events must be scheduled in the register transfer level (RTL) simulation.

For example:

```
Assume `clk, reset, out = 0`

Assume event with `clk = 1`

0: Set `out = count` in next event (1)

1: Set `count = out + 1` using
   logic (may consume multiple events)

X: no further events
```

When we synthesis the code below into a netlist it's a bit harder to see how the events will be scheduled, but we can notice that clk and reset are still inputs, and for example the clock is connected to d-flip-flops. The image below is the synthesized netlist

It should feel intuitive that a gate-level netlist will take longer to simulate than an RTL, there are more events.



13.2 Transient analog simulation

Analog simulation are different. There is no quantized time step. How fast "things" happen in the circuit is entirely determined by the time constants, change in voltage, and change in current in the system.

It is possible to have a fixed time-step in analog simulation, for example, we say that nothing is faster than 1 fs, so we pick that as our time step. If we wanted to simulate 1 s, however, that's at least 1e15 events, and with 1 event per microsecond on a computer it's still a simulation time of 31 years. Not a viable solution for all analog circuits.

Analog circuits are also non-linear, properties of resistors, capacitors, inductors, diodes may depend on the voltage or current across, or in, the device. Solving for all the non-linear differential equations is tricky.

An analog simulation engine must parse spice netlist, and setup partial/ordinary differential equations for node matrix

The nodal matrix could look like the matrix below, i are the currents, v the voltages, and G the conductances between nodes.

$$\begin{pmatrix} G_{11} & G_{12} & \cdots & G_{1N} \\ G_{21} & G_{22} & \cdots & G_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ G_{N1} & G_{N2} & \cdots & G_{NN} \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_N \end{pmatrix} = \begin{pmatrix} i_1 \\ i_2 \\ \vdots \\ i_N \end{pmatrix}$$

The simulator, and devices model the non-linear current/voltage behavior between all nodes

as such, the G's may be non-linear functions, and include the v's and i's.

Transient analysis use numerical methods to compute time evolution

The time step is adjusted automatically, often by proprietary algorithms, to trade accuracy and simulation speed.

The numerical methods can be forward/backward Euler, or the others listed below.

- Euler
- Runge-Kutta
- Crank-Nicolson
- Gear

If you wish to learn more, I would recommend starting with the original paper on analog transient analysis.

SPICE (Simulation Program with Integrated Circuit Emphasis) published in 1973 by Nagel and Pederson

The original paper has spawned a multitude of commercial, free and open source simulators, some are listed below.

If you have money, then buy Cadence Spectre. If you have no money, then start with ngspice.

Commercial - Cadence Spectre - Siemens Eldo - Synopsys HSPICE

Free - Aimspice - Analog Devices LTspice - xyce

Open Source - ngspice

13.3 Mixed signal simulation

It is possible to co-simulate both analog and digital functions. An illustration is shown below.

The system will have two simulators, one analog, with transient simulation and differential equation solver, and a digital, with event queue.

Between the two simulators there would be analog-to-digital, and digital-to-analog converters.

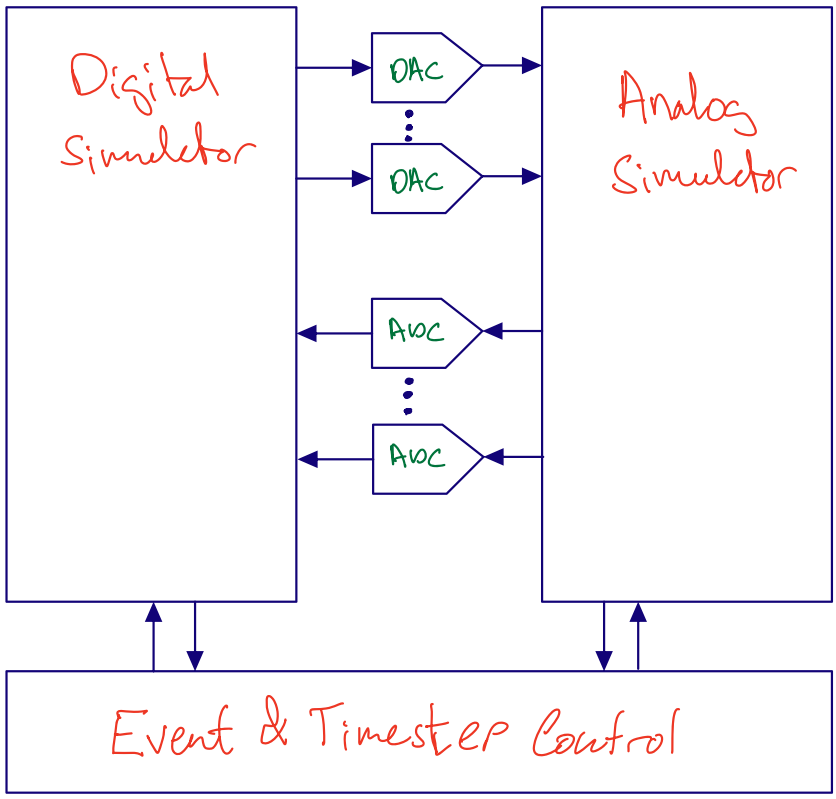
To orchestrate the time between simulators there must be a global event and time-step control. Most often, the digital simulator will end up waiting for the analog simulator.

The challenge with mixed-mode simulation is that if the digital circuit becomes to large, and the digital simulation must wait for analog solver, then it does not work.

Most of the time, it's stupid to try and simulate complex system-on-chip with mixed-signal , full detail, simulation.

For IPs, like an ADC, co-simulation works well, and is the best way to verify the digital and analog.

But if we can't run mixed simulation, how do we verify analog with digital?



13.4 Analog SystemVerilog Example

The key idea is to model the analog behavior to sufficient detail such that we can verify the digital code. I think it's best to have a look at a concrete example.

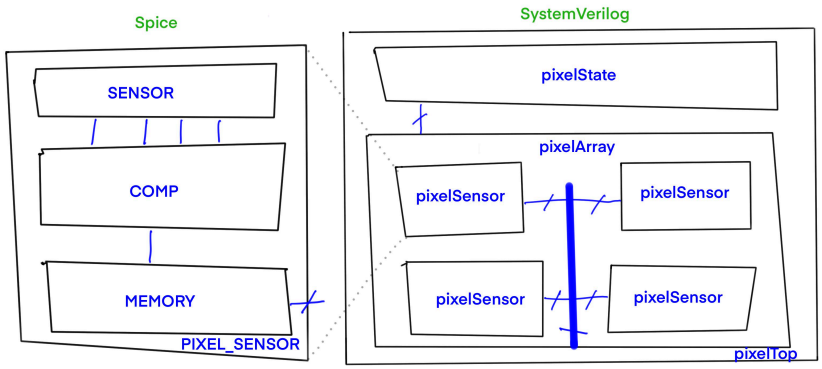
Take a high-speed camera IC, as illustrated below. Each pixel consists of a sensor, comparator and a memory, which are analog functions.

The control of each pixel, and the readout of each pixel is digital.

The image below is a model of the system in A 10 000 Frames/s CMOS Digital Pixel Sensor.

The principle of the paper is as follows: - in each pixel, sample the output of the light sensor - from top level, feed a digital ramp, and an analog ramp to all pixels at the same time - in each pixel, use comparator to lock the memory when the analog ramp matches the sampled output from the sensor

As such, the complete image is converted from analog to digital in parallel.



The verilog below shows how the “Analog SystemVerilog” can be written.

In SystemVerilog there are “real” values for signals and values, or floating point.

The output of the sensor is modeled as the `tmp`` variable. First tmp is reset tov_eraseon theERASE‘ signal.

```
module PIXEL_SENSOR
  (
   input logic      VBN1,
   input logic      RAMP,
   input logic      RESET,
   input logic      ERASE,
   input logic      EXPOSE,
   input logic      READ,
   inout [7:0] DATA

   );

   real             v_erase = 1.2;
   real             lsb = v_erase/255;
   parameter real   dv_pixel = 0.5;

   real             tmp;
   logic            cmp;
   real             adc;
```

```
logic [7:0]      p_data;

//-----------------------------------------------------------------
// ERASE
//-----------------------------------------------------------------
// Reset the pixel value on pixRst
always @(ERASE) begin
   tmp = v_erase;
   p_data = 0;
   cmp  = 0;
   adc = 0;
end
```

When `EXPOSE` is enabled to collect light, we reduce the `tmp` value proportional to an lsb and a derivative `dv_pixel`. The `dv_pixel` is a parameter we can set on each pixel to model the light.

The RAMP input signal is the analog ramp on the top level fed to each pixel. However, since I did not have real wires in `iverilog` I had to be creative. Instead of a continuous increasing value the `RAMP` is a clock that toggles 255 times. The actual "input" to the ADC is the `adc` variable, but that is generated locally in each pixel. When the `adc` exceeds the `tmp` we set `cmp` high.

The previous paragraph demonstrates an important point on analog systemVerilog models. They don't need to be exact, and they don't need to reflect exactly what happens in the "real world" of the analog circuit. What's important is that the behavior from the outside resembles the real analog circuit, and that the digital designer makes the correct design choices.

The comparator locks the `p_data`, and, as we can see, the `DATA` is a tri-state bus that is used both for reading and writing.

```
//-----------------------------------------------------------------
// SENSOR
//-----------------------------------------------------------------
// Use bias to provide a clock for integration when exposing
always @(posedge VBN1) begin
   if(EXPOSE)
     tmp = tmp - dv_pixel*lsb;
end

//-----------------------------------------------------------------
// Comparator
//-----------------------------------------------------------------
// Use ramp to provide a clock for ADC conversion, assume that ramp
// and DATA are synchronous
always @(posedge RAMP) begin
   adc = adc + lsb;
   if(adc > tmp)
     cmp <= 1;
end

//-----------------------------------------------------------------
// Memory latch
```

```
   //-----------------------------------------------------------------
   always_comb  begin
      if(!cmp) begin
         p_data = DATA;
      end

   end

   //-----------------------------------------------------------------
   // Readout
   //-----------------------------------------------------------------
   // Assign data to bus when pixRead = 0
   assign DATA = READ ? p_data : 8'bZ;

endmodule // re_control
```

For more information on real-number modeling I would recommend The Evolution of Real Number Modeling

Chapter 14

Energy Sources

Electronic circuits are wasteful of energy. Digital circuits charge transistor gates to change states, and when discharged, the charges are dumped to ground. In analog circuits the transconductance requires a DC current, a continuous flow of charges from positive supply to ground.

Electronic circuits are incredibly useful though. Life without would be different.

A continuous effort from engineers like me have reduced the power consumption of both digital and analog circuits by order of magnitudes since the invention of the transistor 75 years ago.

One of the first commercial ADCs, the DATRAC, was a 11-bit 50 kSps that consumed 500 W. That's Walden figure of merit of 4 μJ/conv.step. Today's state-of-the-art ADCs in the same sampling range have a Walden figure of merit of 0.6 fJ/conv.step.

4 μ / 0.6 f = 8.1e9, a difference in power consumption of almost 10 billion times !!!

Improvements to power consumption become harder and harder, but I believe there is still far to go before we cannot reduce power consumption any more.

Towards a Green and Self-Powered Internet of Things Using Piezoelectric Energy Harvesting has a nice overview of power consumption of technologies, seen below.

As devices approach average power consumption of μW it becomes possible to harvest the energy from the environment.

I'd like to give you an introduction to harvesting sources, and my opinion on the circuits needed. There are at least 5 sources, thermoelectric, photovoltaic, piezoelectric, ambient RF and triboelectric.

14.1 Thermoelectric

Apply heat to one end of a metal wire, what happens to the free electrons? As we heat the material we must increase the energy of the free electrons at the hot end of the wire. Think of the electrons at the hot side as high energy electrons, while on the cold side there are low energy electrons.

There will be diffusion current of electrons in both directions in the material, however, if the mobility of electrons in the material is dependent on the energy, then we would get a difference in current of low energy electrons and high energy electrons. A difference in current would lead to a charge difference at the hot end and cold end, which would give a difference in voltage.

Take a copper wire, bend it in half, heat the end with the loop, and measure the voltage at the cold end. Would we measure a voltage difference?

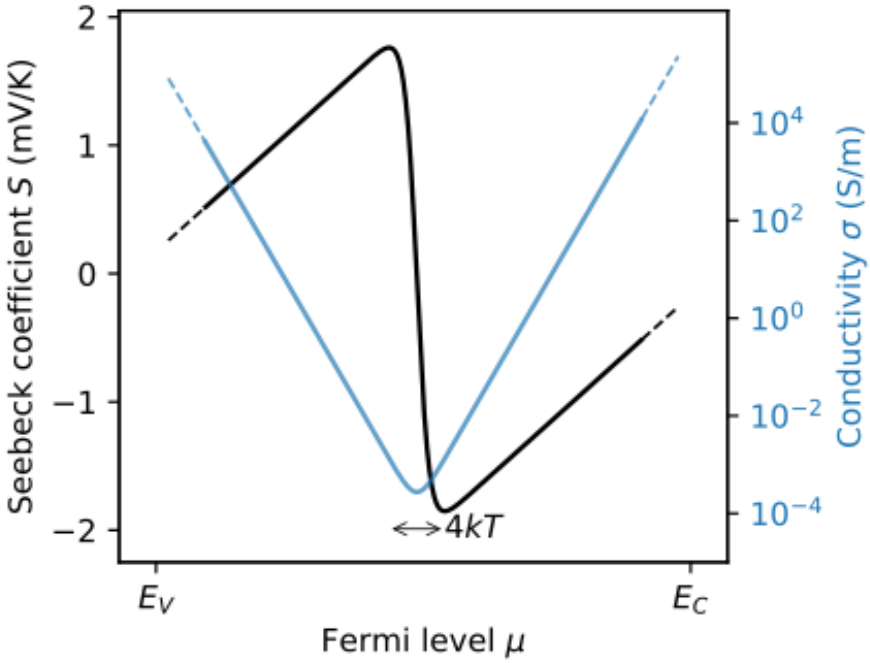
NO, there would be no voltage difference between the two ends of the wire. The voltage on the loop side would be different, but on the cold side, where we have the ends, there would be no voltage difference.

Gauss law tell us that inside a conductor there cannot be a static field without a current. As such, if there was a voltage difference between the cold ends, it would quickly dissipated, and no DC current would flow.

The voltage difference in the material between the hot and cold end will create currents, but we can't use them if we only have one type of material. The voltage difference at the hot and cold end is described by the Seebeck coefficient.

Imagine two parallel wires with different Seebeck coefficients, one of copper ($6.5\ \mu V/K$) and one of iron ($19\ \mu/K$). We connect them at the hot end. The voltage difference between hot and cold would be higher in the iron, than in the copper. At the cold end, we would now measure a difference in voltage between the wires!

In silicon, the Seebeck coefficient can be modified through doping. A model of Seebeck coefficient is shown below. The value of the Seebeck coefficient depends on the location of the Fermi level in relation to the Conduction band or the Valence band.

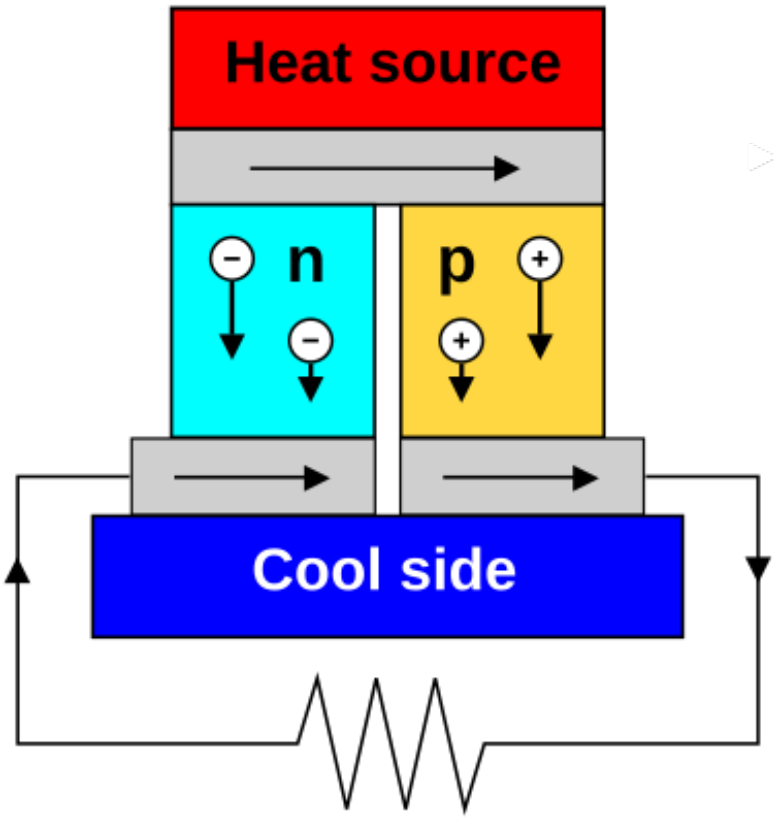


In the picture below we have a material doped with acceptors, and one with donors.

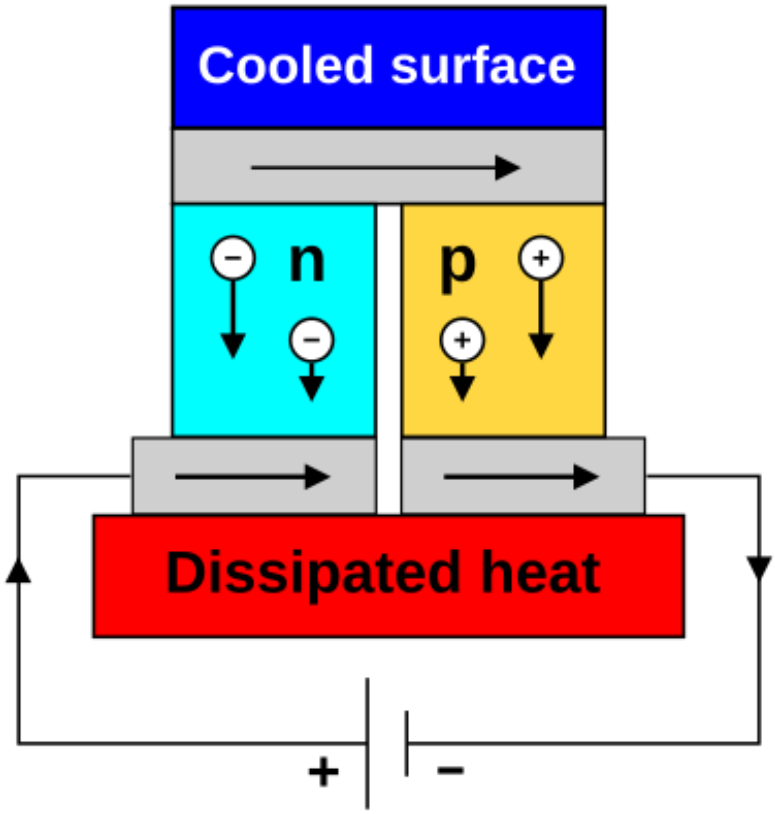
Assume we dope with acceptors, that shifts the Fermi level closer to the Valence band (E_V), and the dominant current transport will be by holes, maybe we get 1 mV/K from the picture above.

For the material doped with donors the Fermi level is shifted towards the Conduction band (E_C), and the dominant charge transport is by electrons, maybe we get -1 mV/K from the picture above.

Assume we have a temperature difference of 50 degrees, then maybe we could get a voltage difference at the cold end of 100 mV, not much, but possible to use.



The process can be run in reverse. In the picture below we force a current through the material, we heat one end, and cool the other. Maybe you've heard of Peltier elements.



14.1.1 Radioisotope Thermoelectric generator

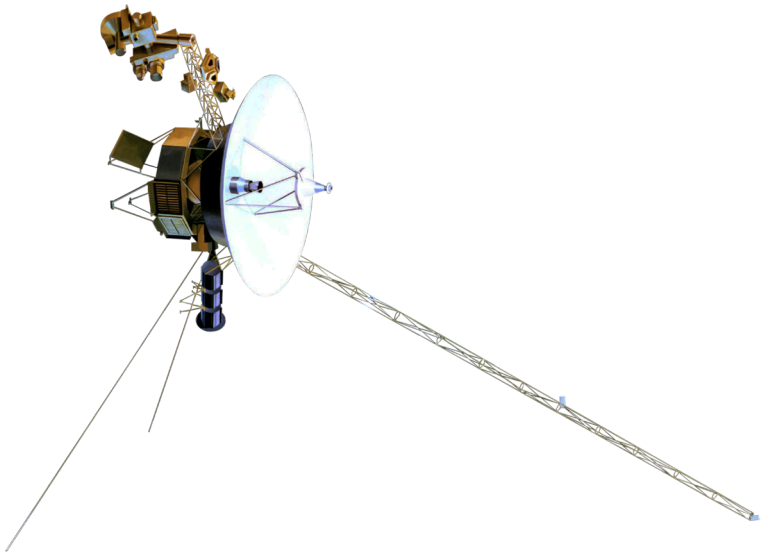
Maybe you've heard of a nuclear battery. Sounds fancy, right? Must be complicated, right?

Not really, take some radioactive material, which generates heat, stick a thermoelectric generator to the hot side, make sure you can cool the cold side, and we have a nuclear battery.

Nuclear batteries are "simple", and ultra reliable. There's not really a chemical reaction. The nucleus of the radioactive material degrades, but not fast. In the thermoelectric generator, there are no moving parts.

In a normal battery there is a chemical reaction that happens when we pull a current. Atoms move around. Eventually the chemical battery will change and degrade.

Nuclear batteries were used in Voyager, and they still work to this day. The nuclear battery is the round thing underneath Voyager in the picture below. The radioisotopes provide the heat, space provides the cold, and voila, 470 W to run the electronics.



14.1.2 Thermoelectric generators

Assume a we wanted to drive a watch from a thermoelectric generator (TEG). The skin temperature is maybe 33 degrees Celsius, while the ambient temperature is maybe 23 degrees Celsius on average.

From the model of a thermoelectric generator below we'd get a voltage of 10 mV to 500 mV, too low for most integrated circuits.

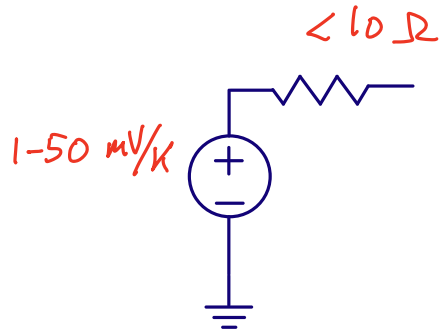
In order to drive an integrated circuit we'd need to boost the voltage to maybe 1.8 V.

The main challenge with thermoelectric generators is to provide a cold-boot function where the energy harvester starts up at a low voltage.

In silicon, it is tricky to make anything work below some thermal voltages (kT/q). We at least need about 3 – 4 thermal voltages to make anything function.

The key enabler for an efficient, low temperature differential, energy harvester is an oscillator that works at low voltage (i.e 75 mV). If we have a clock, then we can boost with capacitors

In A 3.5-mV Input Single-Inductor Self-Starting Boost Converter With Loss-Aware MPPT for Efficient Autonomous Body-Heat Energy Harvesting they use a combination of both switched capacitor and switched inductor boost.



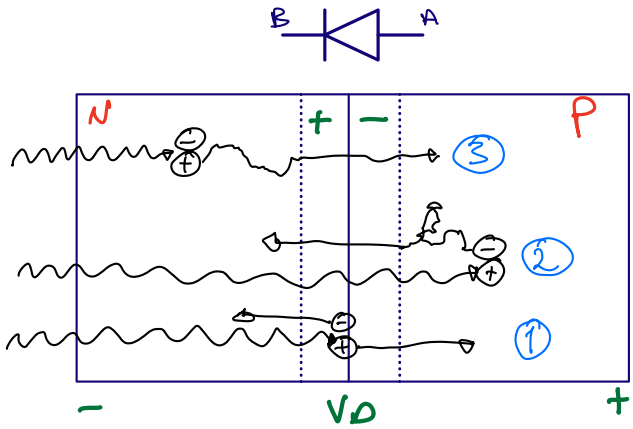
14.2 Photovoltaic

In silicon, photons can knock out electron/hole pairs. If we have a PN junction, then it's possible to separate the electron/holes before they recombine as shown in figure below.

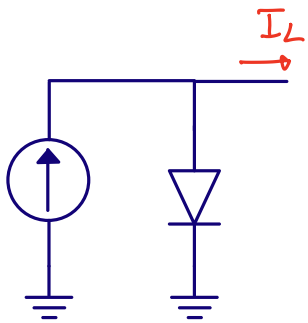
An electron/hole pair knocked out in the depletion region (1) will separate due to the built-in field. The hole will go to P and the electron to N. This increases the voltage VD across the diode.

A similar effect will occur if the electron/hole pair is knocked out in the P region (2). Although the P region has an abundance of holes, the electron will not recombine immediately. If the electron diffuses close to the depletion region, then it will be swept across to the N side, and further increase VD.

On the N-side the same minority carrier effect would further increase the voltage (3).



A circuit model of a Photodiode can be seen in figure below, where it is assumed that a single photodiode is used. It is possible to stack photodiodes to get a higher output voltage.



As the load current is increased, the voltage VD will drop. As the photo current is increased, the voltage VD will increase. As such, there is an optimum current load where there is a balance between the photocurrent, the voltage VD and the load current.

$$I_D = I_S \left(e^{\frac{V_D}{V_T}} - 1 \right)$$

$$I_D = I_{Photo} - I_{Load}$$

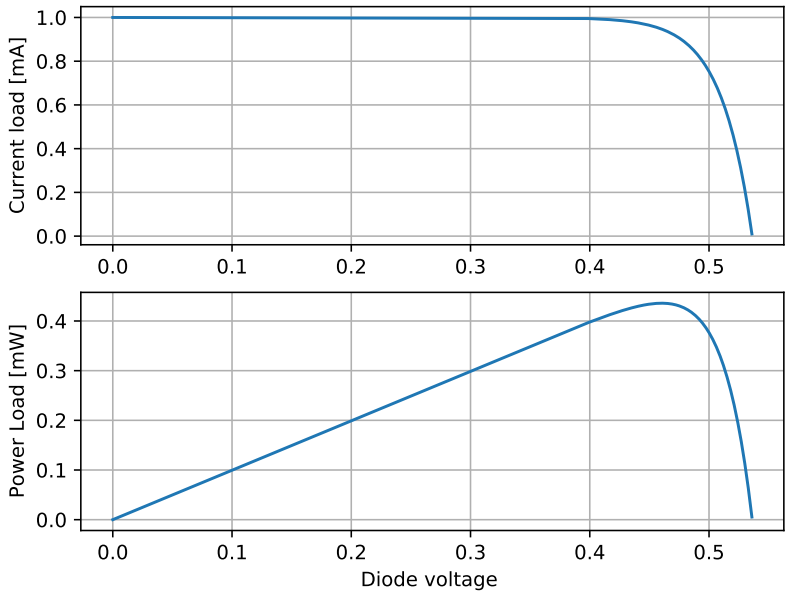
$$V_D = V_T ln\left(\frac{I_{Photo} - I_{Load}}{I_S} + 1\right)$$

$$P_{Load} = V_D I_{Load}$$

Below is a model of the power in the load as a function of diode voltage

```
#!/usr/bin/env python3
import numpy as np
import matplotlib.pyplot as plt

m = 1e-3
i_load = np.linspace(1e-5,1e-3,200)

i_s = 1e-12  # saturation current
i_ph = 1e-3  # Photocurrent

V_T = 1.38e-23*300/1.6e-19  #Thermal voltage

V_D = V_T*np.log((i_ph - i_load)/(i_s) + 1)

P_load = V_D*i_load

plt.subplot(2,1,1)
plt.plot(i_load/m,V_D)
plt.ylabel("Diode voltage [mA]")
plt.grid()
plt.subplot(2,1,2)
plt.plot(i_load/m,P_load/m)
plt.xlabel("Current load [mA]")
plt.ylabel("Power Load [mW]")
plt.grid()
plt.savefig("pv.pdf")
plt.show()
```

From the plot below we can see that to optimize the power we could extract from the photovoltaic cell we'd want to have a current of 0.9 mA in the model above.



Most photovoltaic energy harvesting circuits will include a maximum power point tracker as the optimum changes with light conditions.

In A Reconfigurable Capacitive Power Converter With Capacitance Redistribution for Indoor Light-Powered Batteryless Internet-of-Things Devices they include a maximum power point tracker and a reconfigurable charge pump to optimize efficiency.

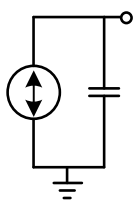
14.3 Piezoelectric

A property of some materials is that they polarize when placed under physical stress, and although I don't exactly understand the physics, this somehow induces a current.

A model of a piezoelectric transducer can be seen below.

The voltage on the transducer can be on the order of a few volts, but the current is usually low (nA – µA). The key challenge is to rectify the AC signal into a DC signal. It is common to use tricks to reduce the energy waste due to the rectifier.

An example of piezoelectric energy harvester can be found in A Fully Integrated Split-Electrode SSHC Rectifier for Piezoelectric Energy Harvesting



14.4 Ambient RF Harvesting

Extremely inefficient idea, but may find special use-cases at short-distance.

Will get better with beam-forming and directive antennas

There are companies that think RF harvesting is a good idea.

AirFuel

I think that ambient RF harvesting should tingle your science spidy senses.

Let's consider the power transmitted in wireless standards. Your cellphone may transmit 30 dBm, your WiFi router maybe 20 dBm, and your Bluetooth LE device 10 dBm.

In case those numbers don't mean anything to you, below is a conversion to watts.

dBm	W
30	1
0	1 m
-30	1 u
-60	1 n
-90	1 p

Now ask your self the question "What's the power at a certain distance?". It's easier to flip the question, and use Friis to calculate the distance.

Assume

$$P_{TX}$$

= 1 W (30 dBm) and

$$P_{RX}$$

= 10 uW (-20 dBm)

then

$$D = 10^{\frac{P_{TX} - P_{RX} + 20log_{10}\left(\frac{c}{4\pi f}\right)}{20}}$$

In the table below we can see the distance is not that far!

Freq	$20log_{10}\left(c/4\pi f\right)$ [dB]	D [m]
915M	-31.7	8.2
2.45G	-40.2	3.1
5.80G	-47.7	1.3

I believe ambient RF is a stupid idea.

Assuming an antenna that transmits equally in all direction, then the loss on the first meter is 40 dB at 2.4 GHz. If I transmitted 1 W, there would only be 100 µW available at 1 meter. That's an efficiency of 0.01 %.

Just fundamentally stupid. **Stupid, I tell you!!!**

Stupidity in engineering really annoys me, especially when people don't understand how stupid ideas are.

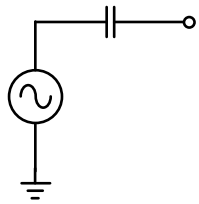
14.5 Triboelectric generator

Although static electricity is an old phenomenon, it is only recently that triboelectric nanogenerators have been used to harvest energy.

An overview can be seen in Current progress on power management systems for triboelectric nanogenerators.

A model of a triboelectric generator can be seen in below. Although the current is low (nA) the voltage can be high, tens to hundreds of volts.

The key circuit challenge is the rectifier, and the high voltage output of the triboelectric generator. Take a look in A Fully Energy-Autonomous Temperature-to-Time Converter Powered by a Triboelectric Energy Harvester for Biomedical Applications for more details.



14.6 Comparison

Imagine you're a engineer in a company that makes integrated circuits. Your CEO comes to you and says "You need to make a power management IC that harvest energy and works with everything".

Hopefully, your response would now be "That's a stupid idea, any energy harvester circuit must be made specifically for the energy source".

Thermoelectric and photovoltaic provide low DC voltage. Piezoelectric and Triboelectric provide an AC voltage. Ambient RF is stupid.

For a "energy harvesting circuit" you must also know the application (wrist watch, or wall switch) to know what energy source is available.

Below is a table that show's a comparison in the power that can be extracted.

The power levels below are too low for the peak power consumption of integrated circuits, so most applications must include a charge storage device, either a battery, or a capacitor.

media/yoo4-3080383-large.png

14.7 References

Towards a Green and Self-Powered Internet of Things Using Piezoelectric Energy Harvesting

A 3.5-mV Input Single-Inductor Self-Starting Boost Converter With Loss-Aware MPPT for Efficient Autonomous Body-Heat Energy Harvesting

A Reconfigurable Capacitive Power Converter With Capacitance Redistribution for Indoor Light-Powered Batteryless Internet- of-Things Devices

A Fully Integrated Split-Electrode SSHC Rectifier for Piezoelectric Energy Harvesting

Current progress on power management systems for triboelectric nanogenerators

A Fully Energy-Autonomous Temperature-to-Time Converter Powered by a Triboelectric Energy Harvester for Biomedical Applications