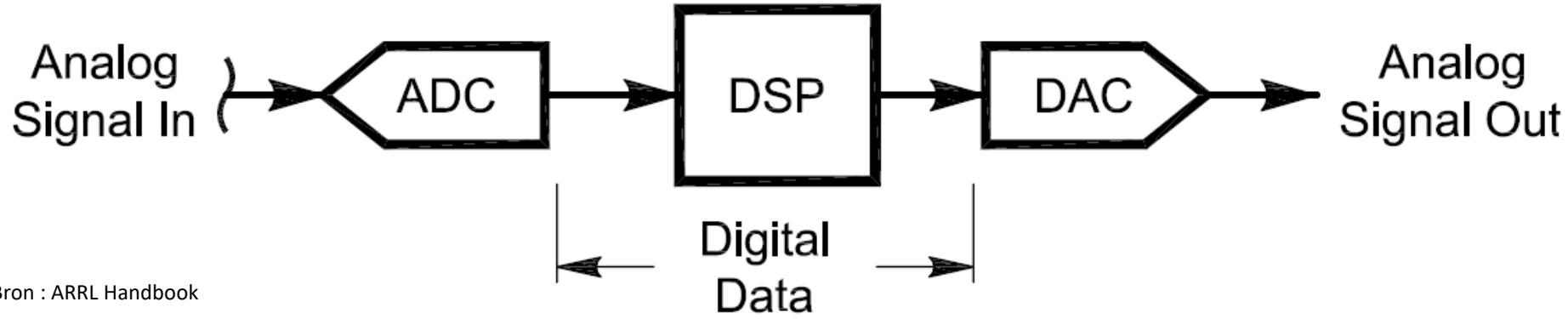
DSP en SDR voor de Ham



Bron : ARRL Handbook

DSP en SDR voor de Ham

Belangrijk doel: Enig gevoel krijgen op de gebruikte DSP terminologie

Veel kretologie enigszins verklaard

Wat achtergronden van SDR ontwerpen

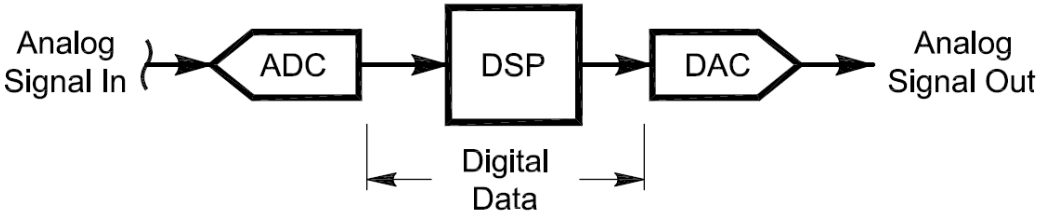
Maar…. DSP, de Processing is wiskunde.
Inhoudelijk slaan we dat over.

Author:
Arie Kleingeld PA3A

Inhoud

Met wat grote stappen door de **terminologie** van het digitale werk (de ‘basis’)

1. Van analoog naar Digitaal (ADC) en weer terug (DAC)
   - Nyquist theorema
   - ADC (uitgebreid), DAC
   - Te kiezen sample-rates
   - Dithering

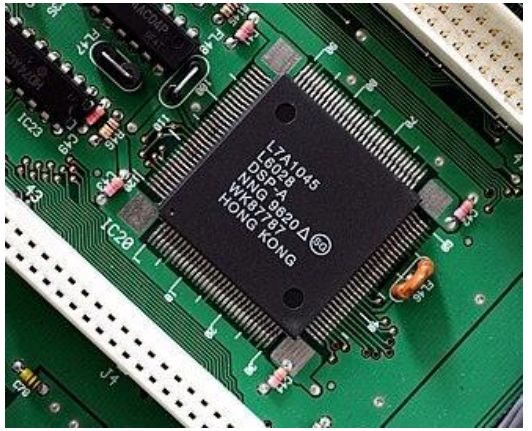


2. Waar het wonder gebeurt: DSP
   - FFT (Fast Fourier Transform) perikelen
   - Verschillende sample snelheden in de DSP (zgn. Multi-rate)
   - Nauwkeurigheid van het rekenen
   - Positieve en negatieve frequenties?
   - Filters: FIR en Fast Convolution m.b.v. FFT, IIR en CIC
3. Wat zijn de stappen van het signaal in een SDTRX
   - De digitale uitvoering in mode SSB vergeleken met analoge trx’s van het F-examen
   - Enkele populaire transceivers in blokken

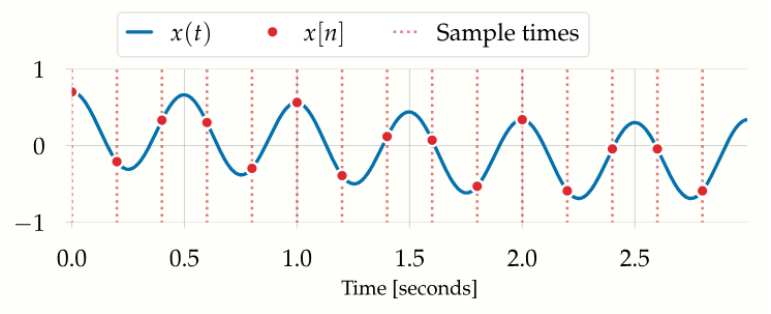
Met wat (grote) stappen langs de DSP-velden…

Dit is pittig dus…
Riemen vast!

DSP – Digital Signal Processing



- DSP is het bewerken (of analyseren) van informatie die is gemeten als een concrete serie gehele getallen (0, 1 , 2 tot ….)
- Signalen worden ***op gezette tijden gemeten en omgezet naar getallen***.
  Dit zijn de zgn. ***samples*** = **steekproeven**.
  Tussen die tijden zien we dus niks, met alle gevolgen van dien.

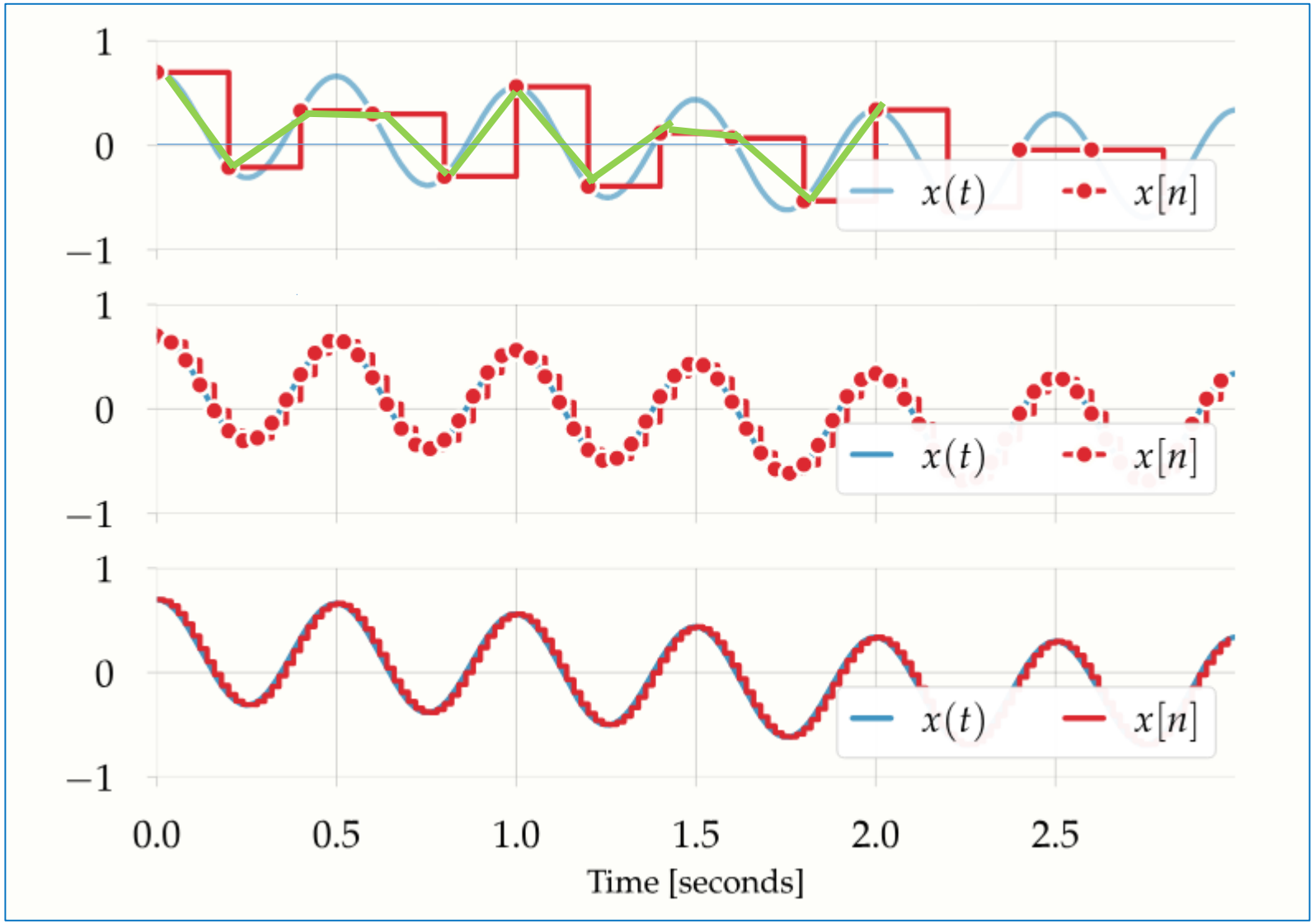


bijvoorbeeld

18002, 8, 24, 645, 2, 25786, etc

- Verwerkingstijd in onze ontvangers van antenne tot luidspreker…
  een beetje snel alstublieft, b.v. liefst binnen 30 ms.
  Dit stelt eisen aan de hard- en software. Dat moet je slim doen.

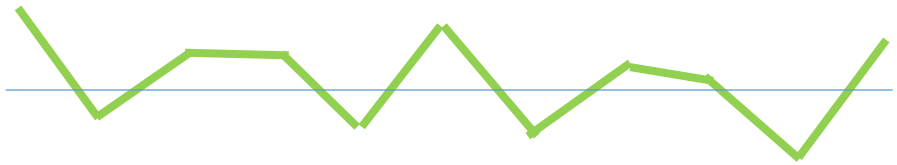
Signaal in samples



x(t) = signaal in de tijd

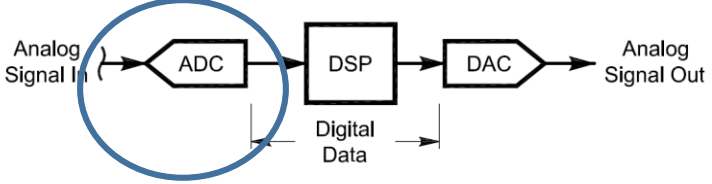
x(n) = signaal in samples

Signaal in samples

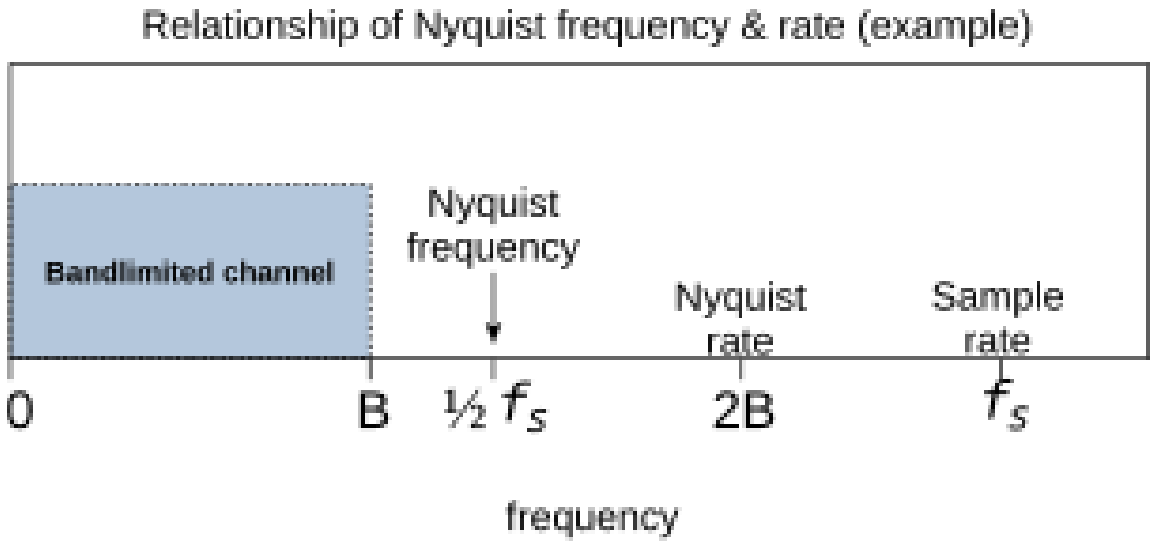


- Ziet er niet meer uit als die oorspronkelijke sinus…
- Is die oorspronkelijke sinus uit deze samples nog terug te halen?
- Wat heb je nodig?

Sampling is niet zo maar sampling



- **Meten** van een (analoog) signaal **op vastgestelde tijdstippen** en het **digitaal vastleggen** van de gemeten waarde doen we met een **Analog – Digital Converter** - of kortweg **ADC**.



- **Claude Shannon**: er is geen informatie verlies van het signaal als de ***Sample rate*** groter is dan de ***Nyquist rate,*** dit komen we ook tegen als het ***Nyquist Sampling Criterium***.
  - De ***Nyquist rate*** is 2x bandbreedte van het gesampelde signaal
  - De ***Nyquist frequentie*** *= ½ x* ***Sample rate***
- ***Om de bandbreedte van het gesampelde signaal te beperken gebruiken we vooraf een analoog filter. We noemen dit straks een 'anti-alias filter'.***

Sampling

Hoge sampling frequentie voor grote bandbreedte is zalig makend?

- Ja
  - grote bandbreedte - vinden wij amateurs erg leuk
  - technisch positieve invloed op SNR (verklaring volgt later)
- Nee
  - erg veel data om te verstouwen - er zijn limieten…
  - stel dat je een frequentie van 1 Hz nog wil zien moet je 1 seconde lang samplen en die als geheel bekijken – bij elkaar heel veel samples om te analyseren

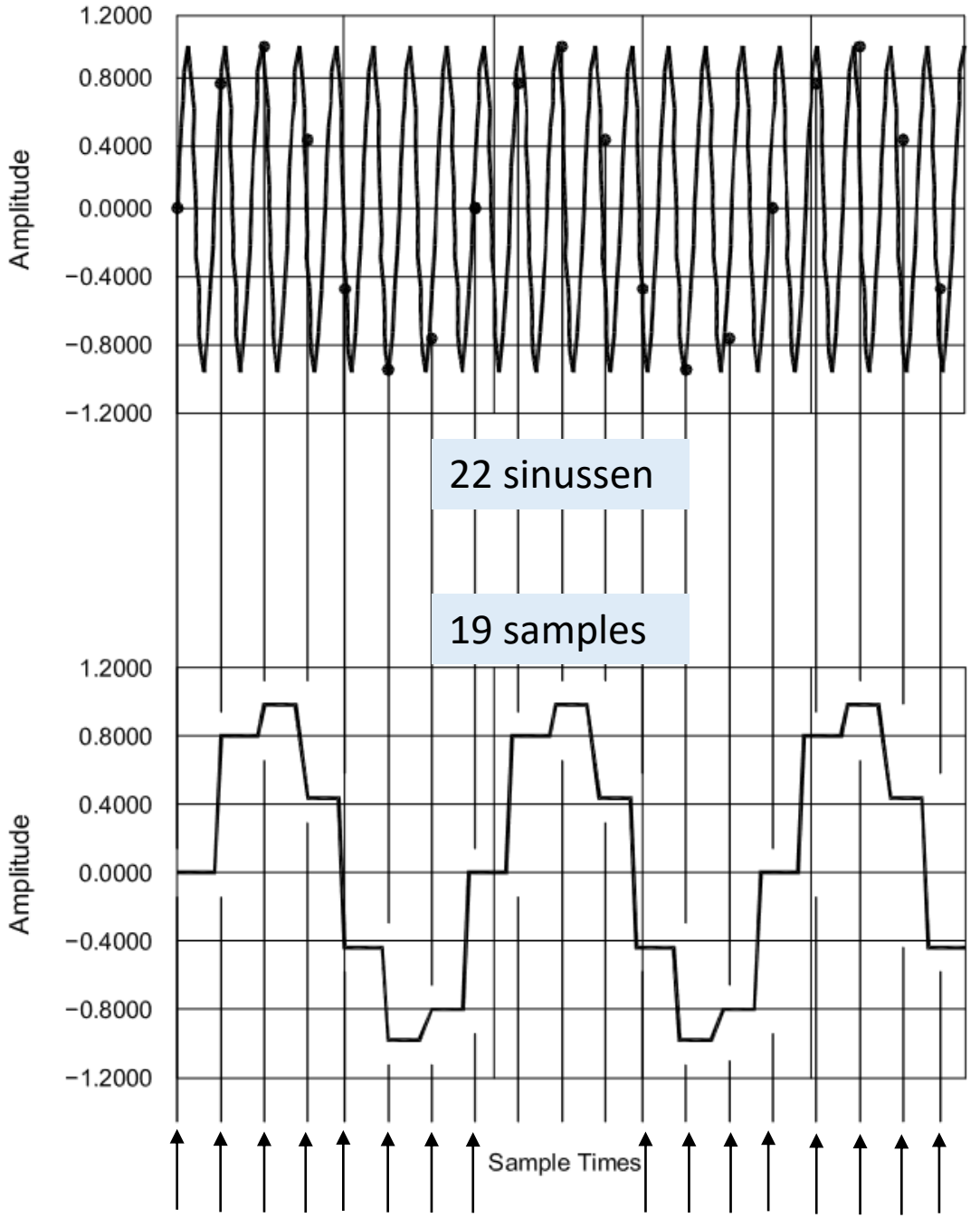
Sampling foutjes

- Te traag samplen = te lage sample rate
  Het beeld van de samples is anders dan werkelijk.
  Dit heet **aliasing**. 22 Hz doet zich hier b.v. voor als ongeveer 3 Hz (**de alias**).

- Het digitale getal ≠ analoge werkelijkheid
  De afronding in het gemeten getal geeft **random noise** t.o.v. het werkelijke signaal.

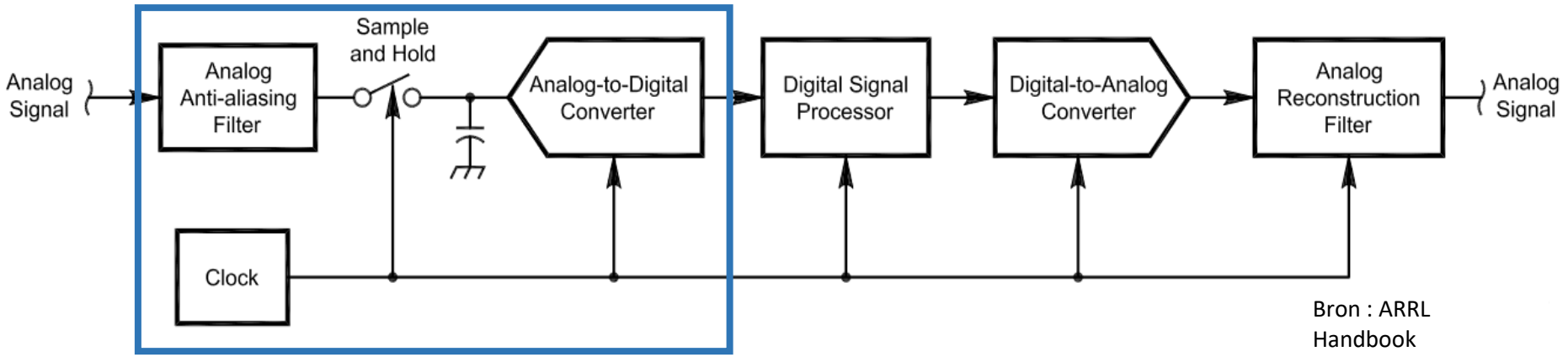
- Sample tijdstip is nooit 100% stabiel
  Er worden a.h.w. verkeerde waarden gemeten.
  Ook dit is in het algemeen random noise.

Bron : ARRL Handbook



Nog eens samenvattend

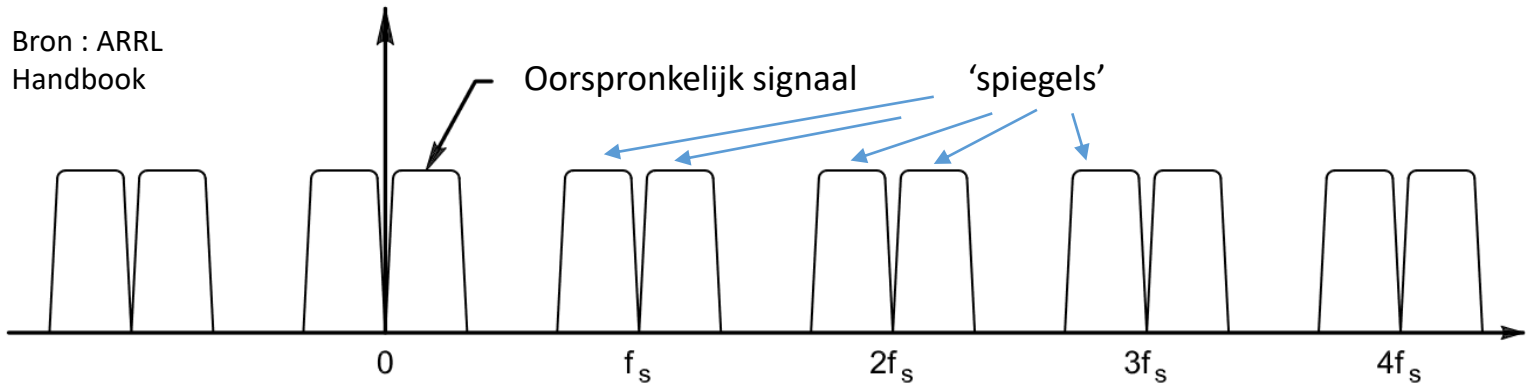
- Bij baseband sampling, d.w.z. van 0 Hz tot X Hz, zorgt het anti-aliasfilter ervoor dat er geen hogere frequenties kunnen worden gesampled dan de Nyquist frequentie (= ½ x sample frequentie)
- Anti-alias filter zorgt hiervoor en is analoog maar nooit ideaal. Het is onmogelijk om de stopband-demping oneindig hoog te krijgen en de doorlaatband helemaal vlak.
- De Clock is nooit 100% stabiel, samplemoment kan ietsje afwijken
  (vraag: is dat in ns, ps, of zelfs fs?)
- De Sample&Hold switch houdt een waarde vast voor de ADC, dus wat er tot de volgende sample gebeurt, weten we niets van en wordt in de berekeningen dus niet meegenomen!



Bron : ARRL Handbook

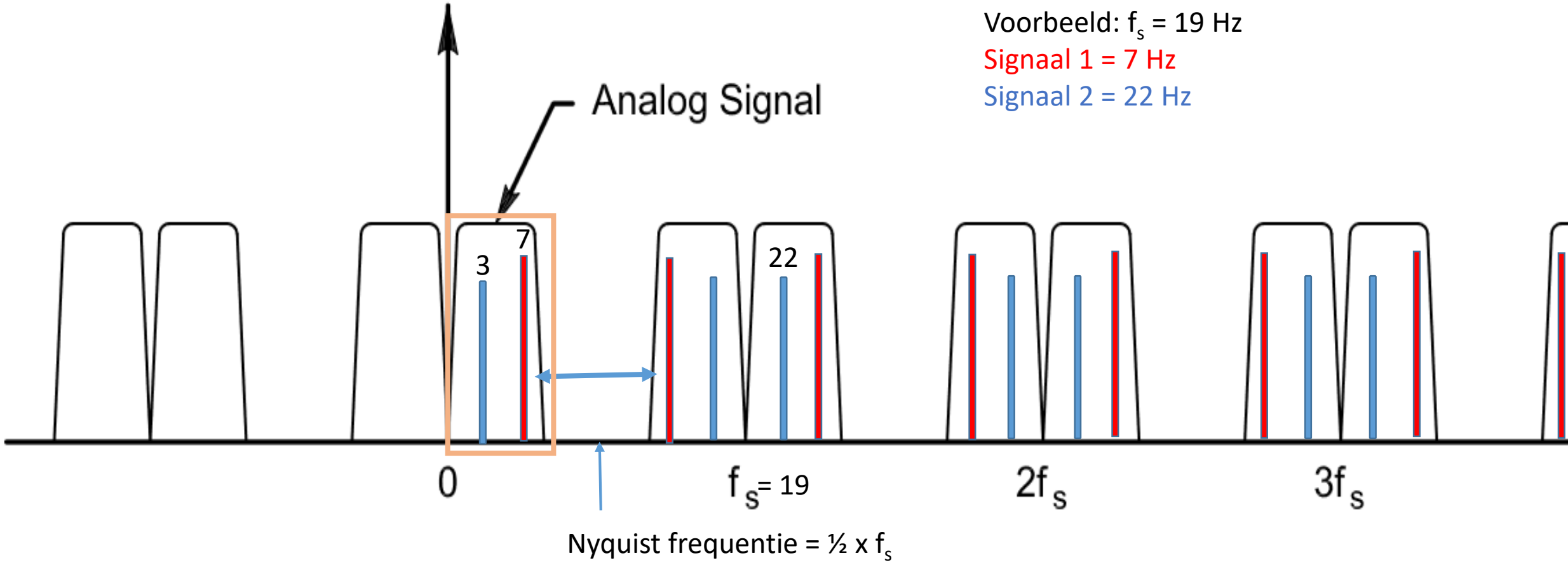
Een beetje onder de (*wiskundige*) motorkap: de samples

- Samples zijn **pulsen** met een variërende waarden zijn dus geen blokjes. Immers de waarden zijn bepaald **per *tijdstip*, niet per tijdsperiode**
- Dat heeft voor een serie samples tot gevolg:
  - Een oneindig aantal harmonischen *in het (wiskundige) digitale domein*, allemaal zelfde sterkte met frequenties: f_s, $2f_s$, $3f_s$, $4f_s$ enz. waarbij f_s de sample frequentie is
  - De pulsen worden in hoogte 'gemoduleerd' (de sample-waarden), in dit geval levert dit wiskundig een **soort 'zijbanden zonder draaggolf'**, of beter gezegd **'meerdere spiegels'** die met periode f_s (de samplefrequentie) herhaald worden



Alias

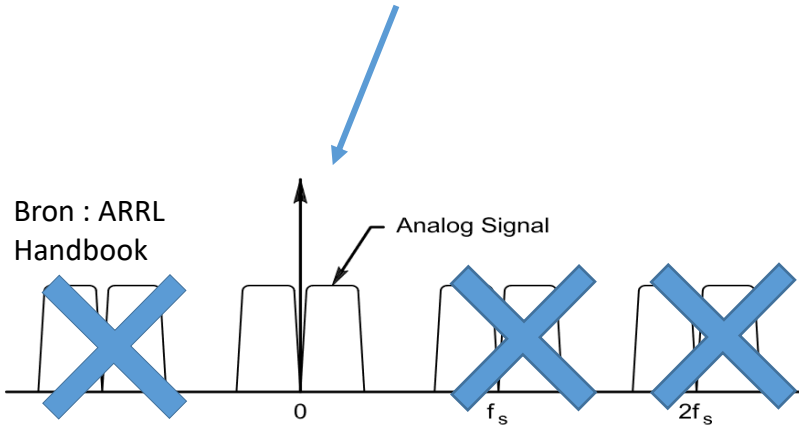
Het zgn. anti-alias filter zorgt voor het beperken van de bandbreedte zodat de spectra elkaar niet gaan overlappen bij de Nyquist frequentie
Een anti-alias filter is echter nooit ideaal)



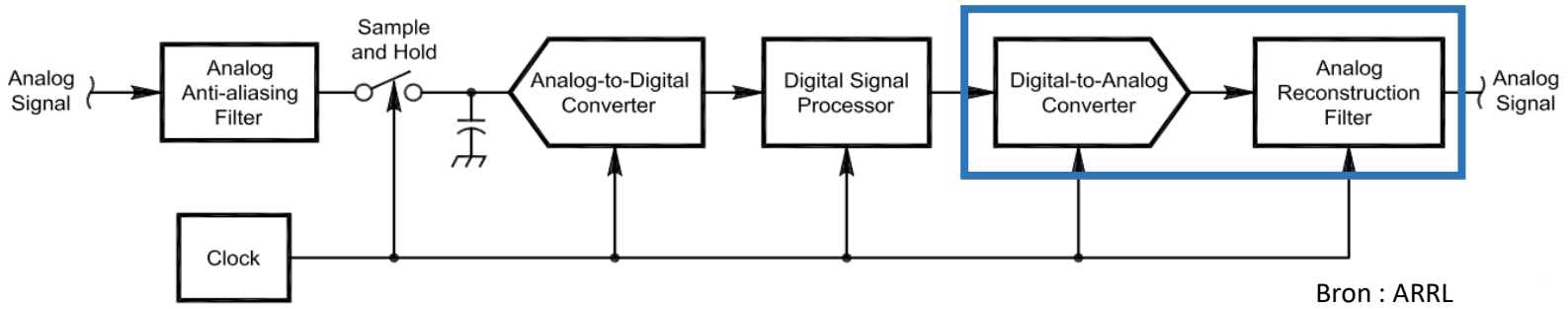
Voorbeeld: f_s = 19 Hz
Signaal 1 = 7 Hz
Signaal 2 = 22 Hz

Na het DSP terug van wiskunde naar onze analoge werkelijkheid

- Zodra we verder naar de analoge output gaan (met een DAC), terug in de werkelijkheid, dan dienen alle harmonische en de hogere ‘mengproducten’ te worden onderdrukt.
- Dit is de taak van het zgn. *reconstructie* filter. Dit is analoog.
  Vaak is dit een laag-doorlaat filter.
- Voor de volledigheid: de digitale signalen (=getallen) boven en onder de ‘0 Hz’ die overblijven kunnen we bij elkaar optellen tot het reële analoge signaal, onze werkelijkheid.



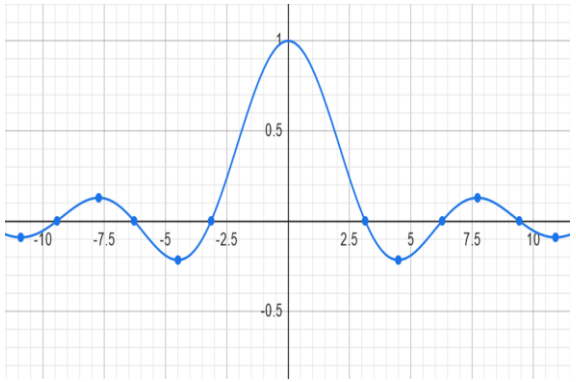
Bron : ARRL Handbook

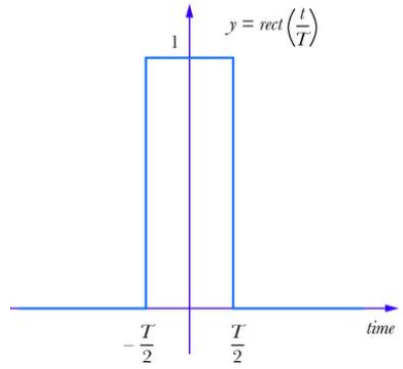


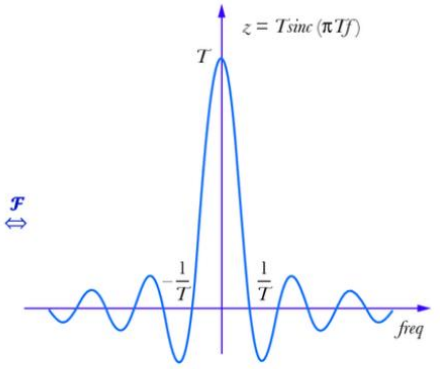
Bron : ARRL Handbook

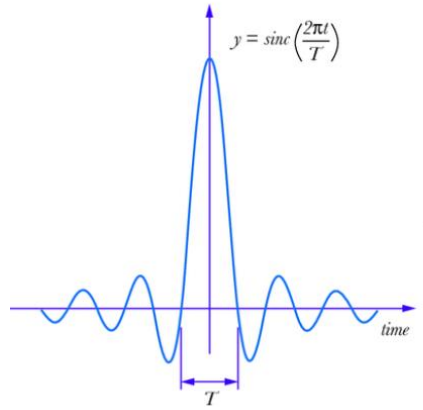
Belangrijk uitstapje tot verdieping: de sinc-funtie

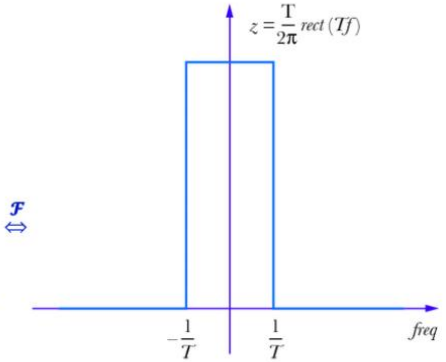
- Sinc = sin(x) / x
- Lijkt op een sinusvorm die vanaf 0 steeds kleiner wordt. Speelt een belangrijke rol in de DSP
- Een constante waarde signaal met bepaalde tijdlengte (in het tijdsdomein) heeft een spectrum (in het frequentiedomein) dat er uit ziet als de sinc-functie (simpel gezegd)
- En andersom ook (de wiskunde is heel geduldig):
  Een blokvormig spectrum heeft in de tijd een sinc-functie als vorm die overigens oneindig lang duurt.

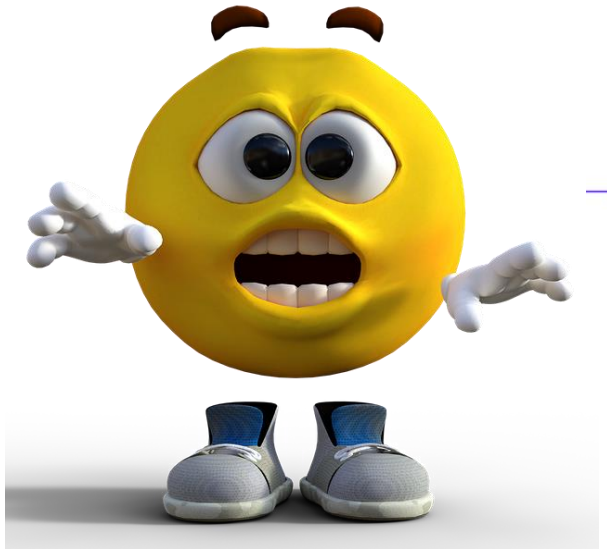






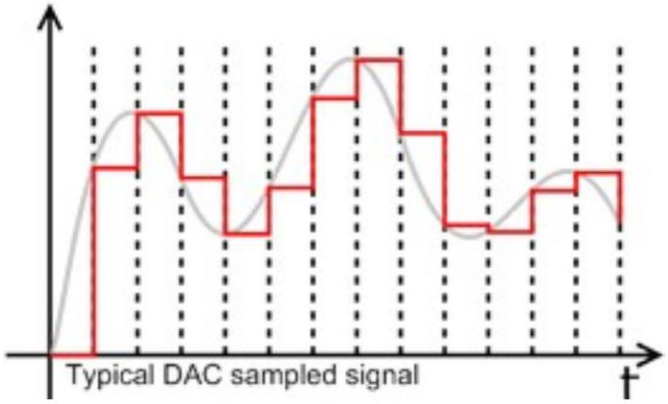




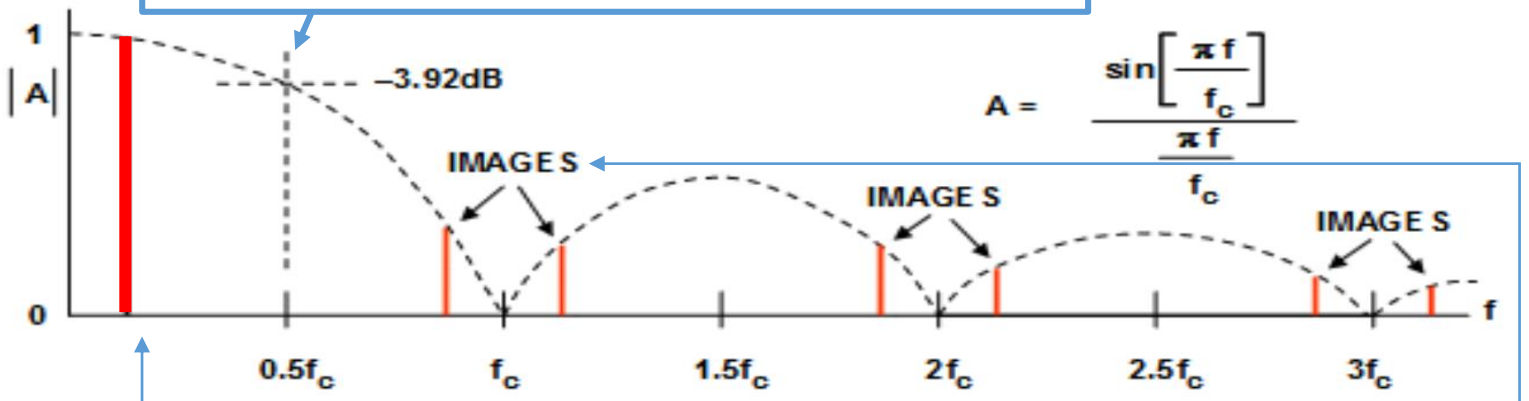


Straks hierover meer

DAC (‘zero order hold’) en het reconstructiefilter
(weer onder de motorkap)



DAC is dus niet lineair als het gaat om output.
In de buurt van Nyquist frequentie bijna 4 dB verzwakking.



- DAC output en bijbehorend output-spectrum en heeft de vorm van de zgn. sinc-funtie.
- f_c is de sample frequentie.
- Zichtbaar is de verzwakking van de spiegels (images) rondom de harmonischen van de sample frequentie.
- **Hoe hoger de sample frequentie t.o.v. frequentie die gereconstrueerd moet worden, des te beter de onderdrukking.**
- Het analoge ‘reconstructiefilter’ hoeft maar een deel te doen (!)

Reken voorbeeld m.b.v. de sinc-functie:

- Sample rate is 48 kHz, het audio signaal is 300 – 3000 Hz
- T.o.v. de max (0dB):
  - 300 Hz : -0,001 dB
  - 3000 Hz : -0,056 dB
  - Spiegelinderdrukking: 48000-3000 = 45000 Hz: -23,6 dB
    48000- 300 = 47700 Hz: -44 dB
    (Ingeval van 192 kHz sample rate: resp. -36dB en -56 dB)
- **Enz. Maar hoe hoger de audio frequentie t.o.v. de sample frequentie, des te sterker de spiegels** en v.v.
  (zie ook volgende slide)

Anekdote: cd-spelers

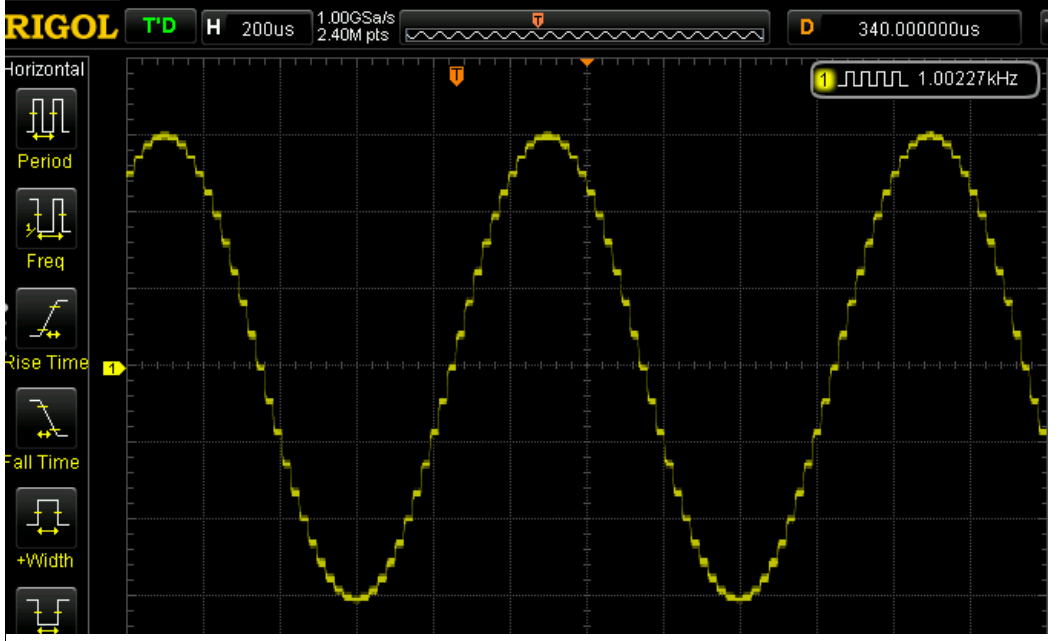
Gevolgen van deze DAC:

- Hoge sample rate bij de DAC zorgt voor minder vervorming binnen het gewenste frequentiegebied. Conform vorige slide

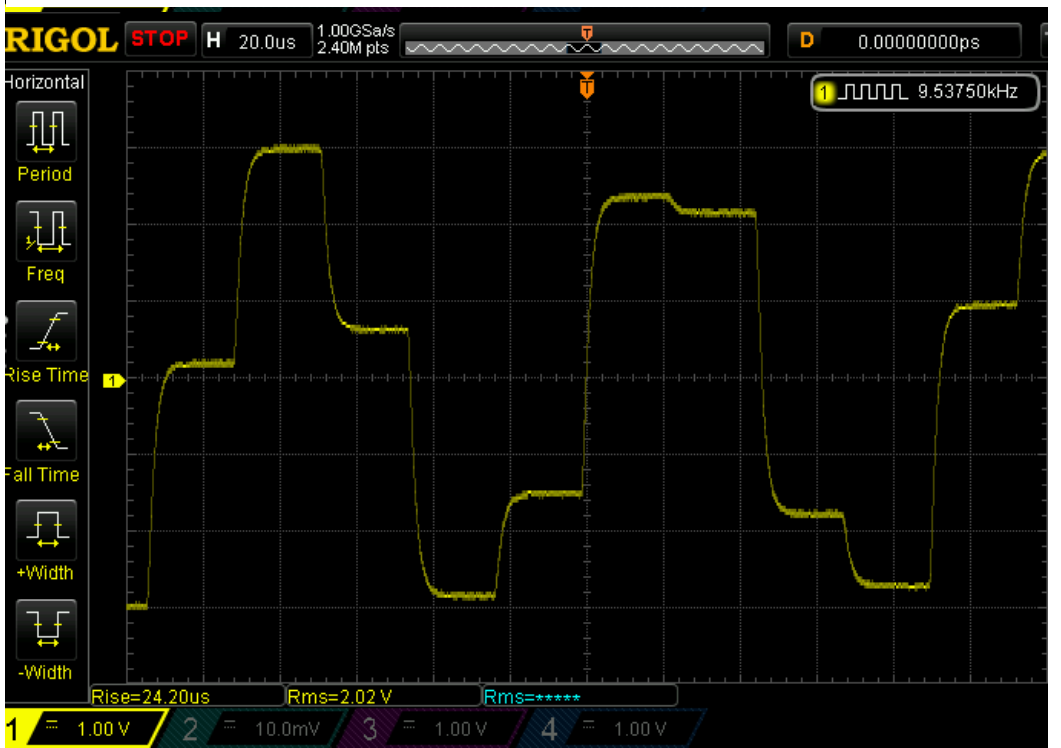
- Vervorming van het bedoelde analoge signaal, waarbij hogere frequenties tot 4dB worden verzwakt (vorige plaatje met de sinx / x functie)

  Remedie: een compenserend *digitaal filter* voor de DAC om de verzwakte frequentiecomponenten te corrigeren

- Daarna analoog het reconstructie filter. Tegenwoordig zijn er ook goede analoge brick-wall filters, kant en klaar in geluidskaart.

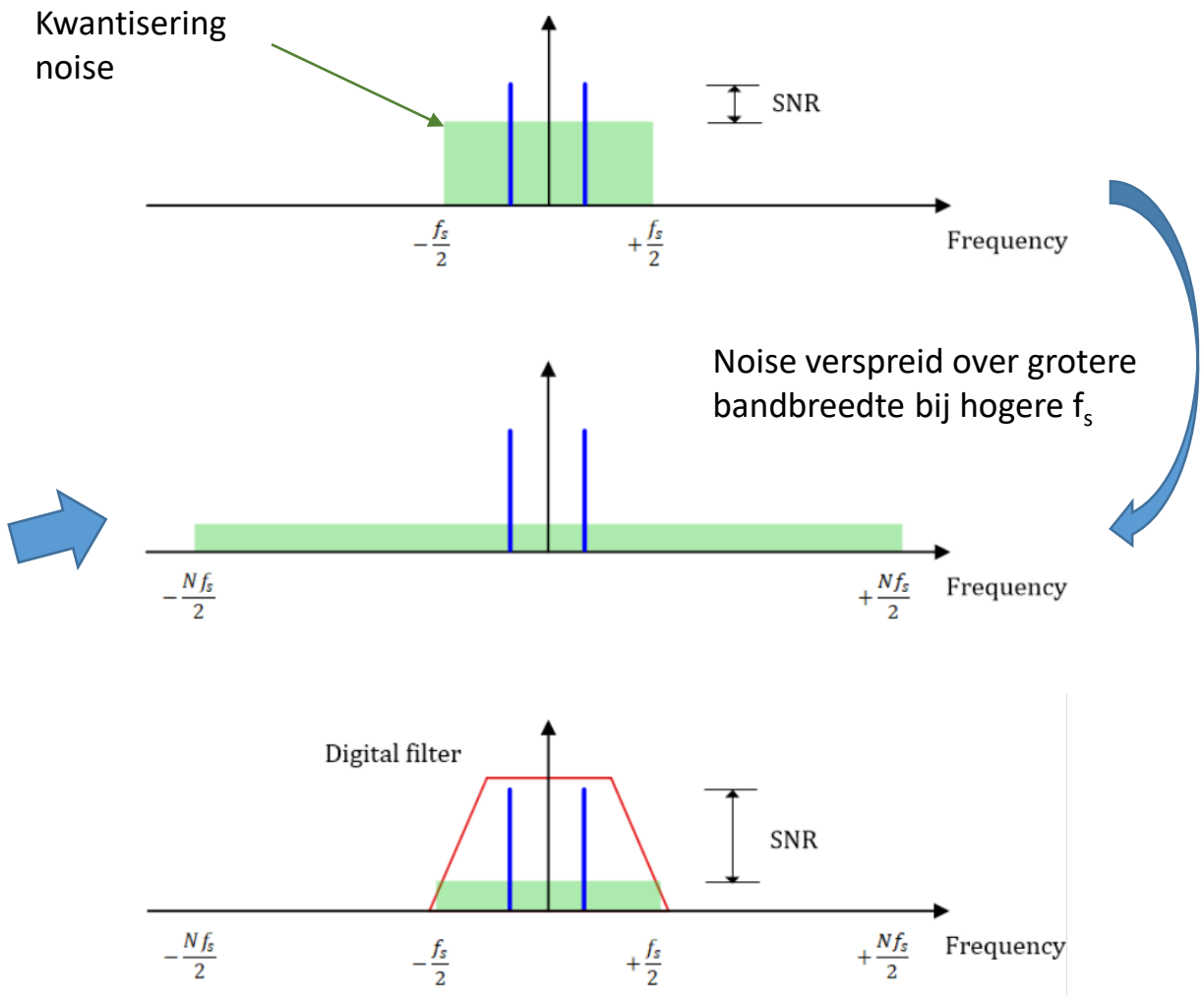


Bron: addictedtoaudio.com.au

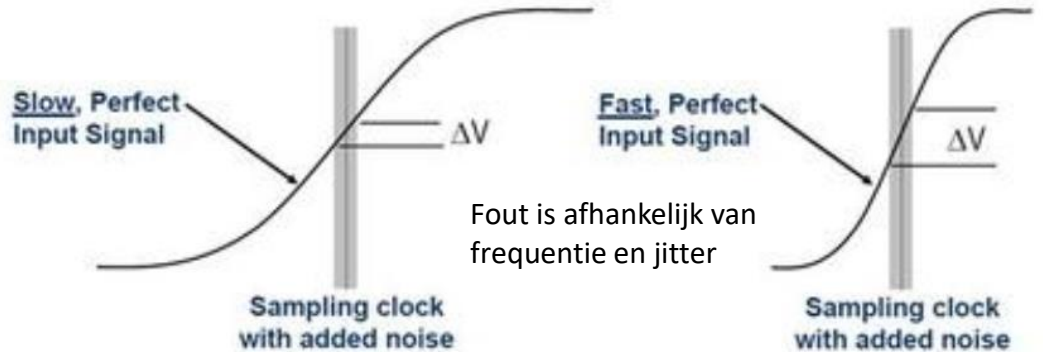


Signaal / Ruisverhouding in de ADC – aantal bits en sample rate

- **Max S/N ratio =**
  Full Scale signal/ kwantiseringsruis
  (zie sampling)
- Voor **IDEALE ADC**: Max **S/N ratio =**
  6,02 x aantal bits + 1,76 dB
  16 bit → S/N = 98 dB
- Iedere bit extra levert 6dB extra S/N ratio op
- Iedere verviervoudiging van de samplerate Fs levert 6dB extra SNR op door noise verspreiding. Dit is a.h.w 1 bit extra!
- *Maar nogmaals, een ADC is nooit ideaal* ☺



SNR in de ADC en klok-jitter



- Door de jitter op de ADC clock ontstaan fouten in de gemeten waarden en dit uit zich in ruis.
- Uit de figuur blijkt:
  - Hoe meer jitter, des te meer foutruis
  - Hoe hoger de signaalfrequentie, des te meer foutruis
- Volgens analog.com is de maximale SNR van de ADC wat betreft jitter:
  max SNR = -20 x log($2\pi f_{signaal}$ x t_{jitter})
  $f_{signaal}$ = **gesampelde** frequentie (niet de sample frequentie), t_{jitter} de hoeveelheid **jittertijd (RMS)**

- Bij $f_{signaal}$ = 15 MHz en t_{jitter} = 80 fs (= 80 x 10^{-15} !) is de max SNR gelijk aan:
  -20 x log(6,28 x 15e6 x 80 e-15) = 102 dB
- Bij een slechte klok jitter kan dit een limiterende factor worden.
  B.v.: $f_{signaal}$ = 50MHz en t_{jitter} = 100 fs dan is de max SNR = 90 dB

De stapgrootte van de ADC is niet constant

- D.w.z. als per 0,1V verschil de ADC 1 LSB waarde moet veranderen, dan gebeurt dat niet. Het varieert een beetje.
  Afwijking t.o.v. ideale curve: DNL en INL

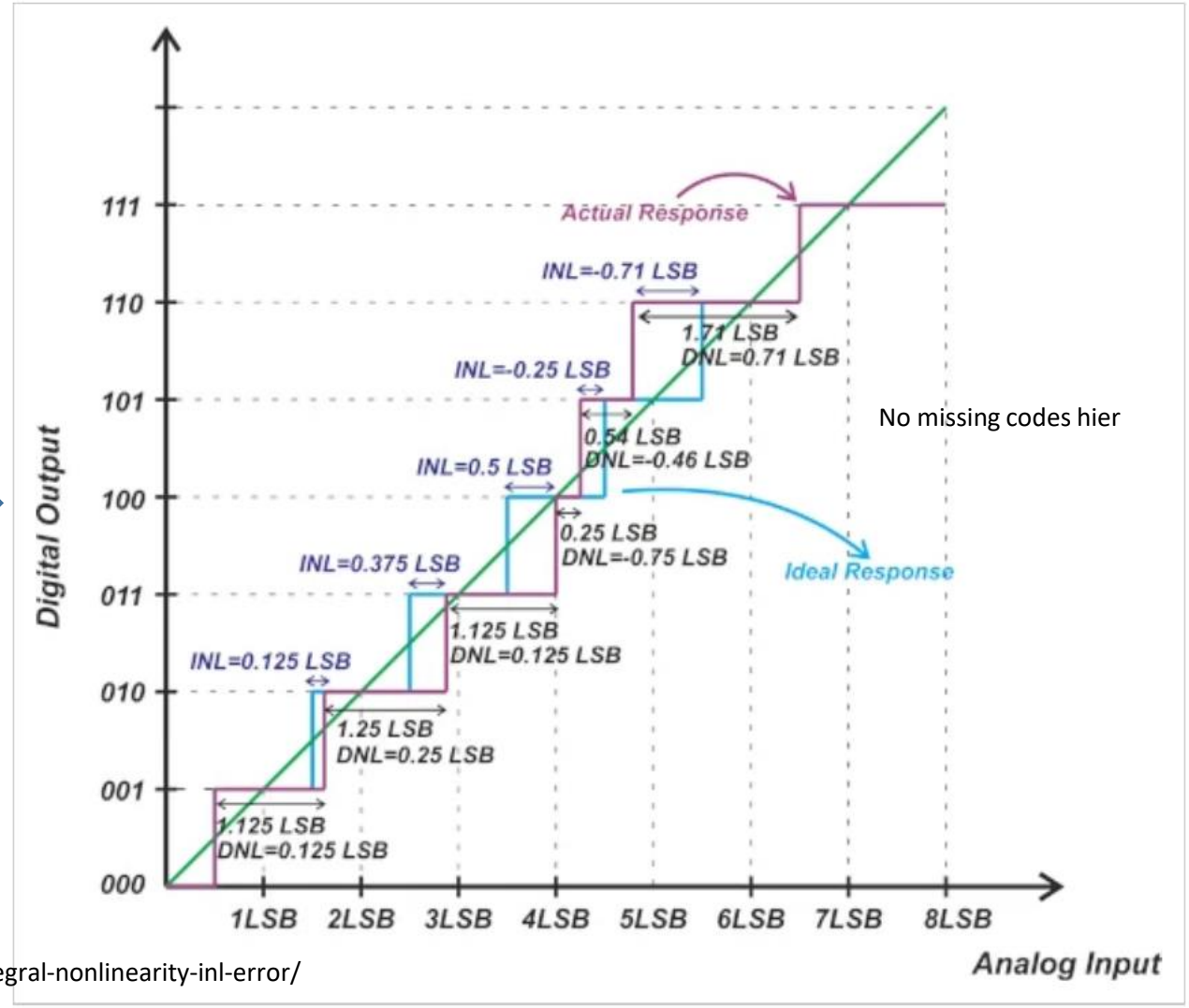
- ADC Differential Non Linearity (DNL)

(Stap afwijking) $$DNL(k) = \frac{W(k) - W_{ideal}}{W_{ideal}}$$

- ADC Integral Non Linearity (INL)

(totale afwijking) $$INL(k) = \frac{T_a(k) - T_{ideal}(k)}{Ideal\ Step\ Size}$$

- Al met al weer een bijdrage aan meetruis



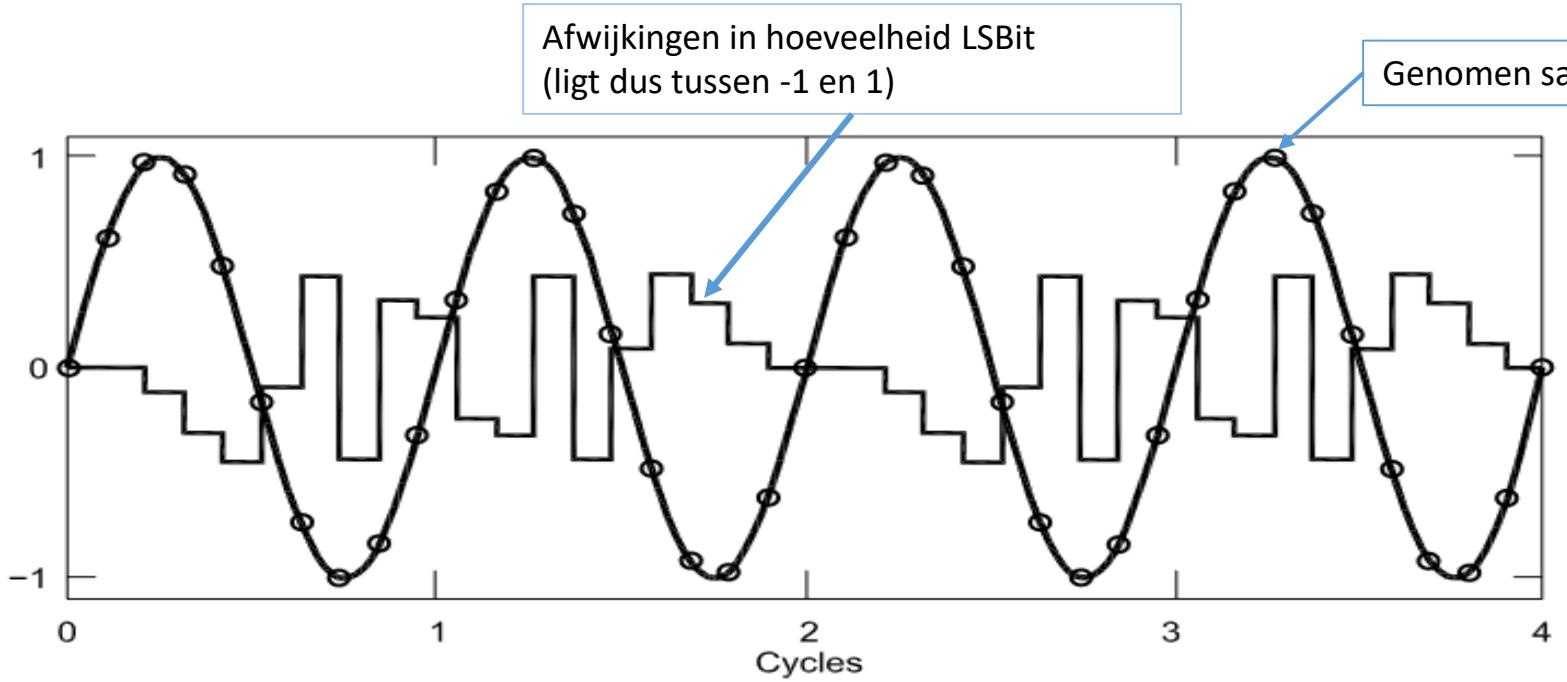
No missing codes hier

https://www.allaboutcircuits.com/technical-articles/understanding-analog-to-digital-converter-integral-nonlinearity-inl-error/

Dithering (mooie kreet) en het helpt ergens tegen

Maar wat is het en hoe werkt het *(Op Icom IC-7300 heet het b.v. **IP+**)*

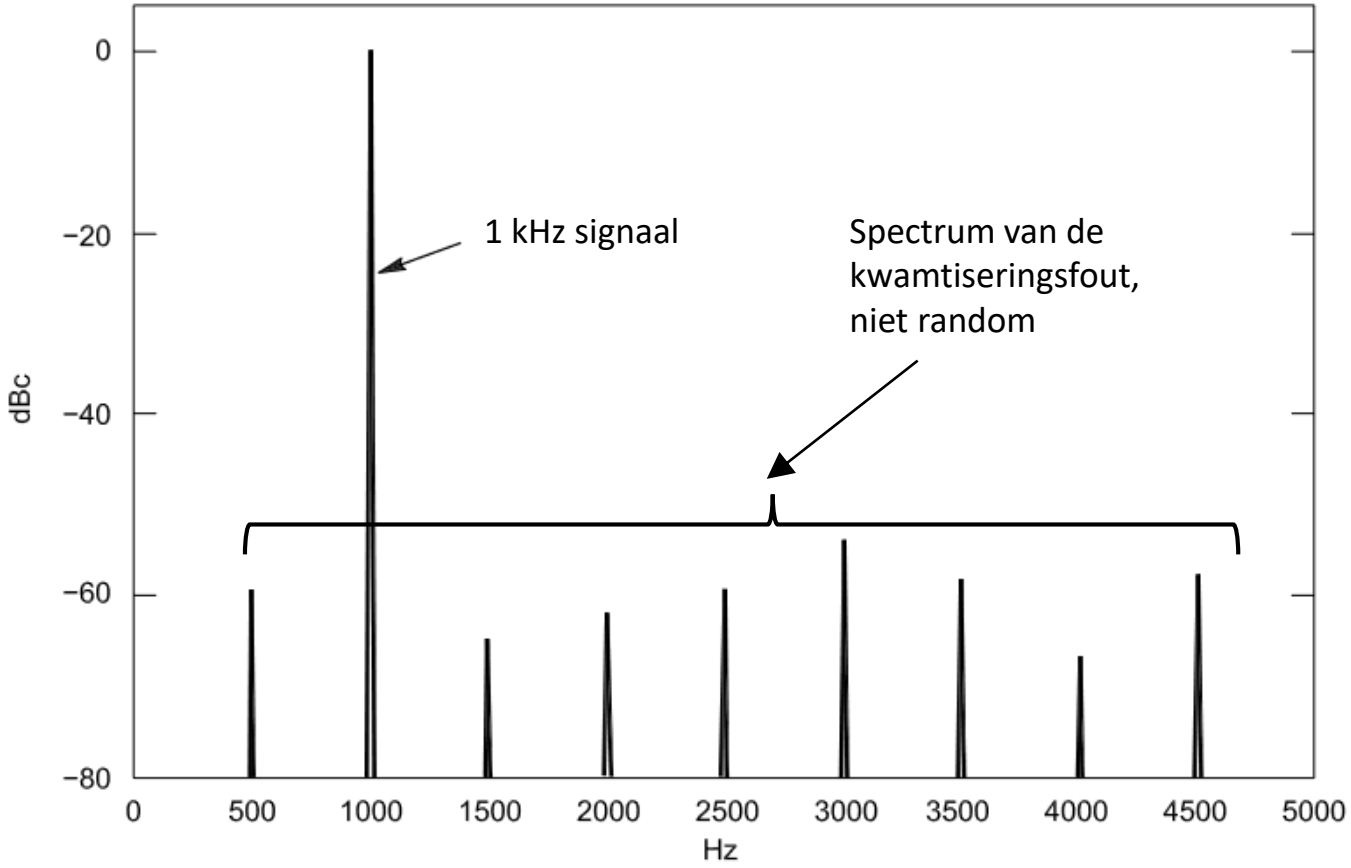
- Normaliter is kwantiseringsruis random en dus breed verdeeld over het spectrum, zie de eerdere opmerkingen over SNR
- Stel dat er een heel hard signaal of meerderde signalen de ADC domineren → de kwantiseringsfout wordt dan voornamelijk hierdoor bepaald en is *niet meer random maar periodiek*!
- In de figuur:
  De sinus wordt gesampeld. Het bloksignaal geeft de kwantiseringsfout weer in aantal LSBit. Je ziet een zich herhalend foutsignaal (periodiek). Dit is géén random noise meer en het heeft dus een bepaald spectrum!



Signaal van 1 kHz
gesampeld met 9,5 kHz
ADC: 8 bits

Dithering 2

- De periodieke kwantiseringsruis heeft een eigen spectrum gekregen dat duidelijk zichtbaar is
- Dus: Spurious… **Schande!**
- Het niveau dat nog geen spurious boven de ruisvloer laat zien, heet Spurious Free Dynamic Range: ‘SFDR’, de nu vaste kwaliteitsparameter voor SDR i.p.v. de analoge ‘IP3’.
- Door extra randomruis toe te voegen aan de ADC, ordergrootte ca. 2 LSBits groot, wordt de kwantiseringsfout weer random gemaakt
- **Weg Spurious**… (weer een plekje omhoog in de Sherwood list) maar wel een hogere ruisvloer, b.v. bij een ic-7610 +3dB
- Dit proces heet DITHERING



SFDR van een ADC… die van de IC-7610

LINEAR TECHNOLOGY

LTC2208

16-Bit, 130Msps ADC

FEATURES

- **Sample Rate: 130Msps**
- **78dBFS Noise Floor**
- **100dB SFDR**
- **SFDR >83dB at 250MHz (1.5V_{P-P} Input Range)**
- **PGA Front End (2.25V_{P-P} or 1.5V_{P-P} Input Range)**
- **700MHz Full Power Bandwidth S/H**
- **Optional Internal Dither**
- **Optional Data Output Randomizer**
- LVDS or CMOS Outputs
- Single 3.3V Supply
- Power Dissipation: 1.25W
- Clock Duty Cycle Stabilizer
- Pin Compatible 14-Bit Version
  130Msps: LTC2208 (16-Bit), LTC2208-14 (14-Bit)
- 64-Pin (9mm × 9mm) QFN Package

DESCRIPTION

The LTC®2208 is a 130Msps, sampling 16-bit A/D converter designed for digitizing high frequency, wide dynamic range signals with input frequencies up to 700MHz. The input range of the ADC can be optimized with the PGA front end.

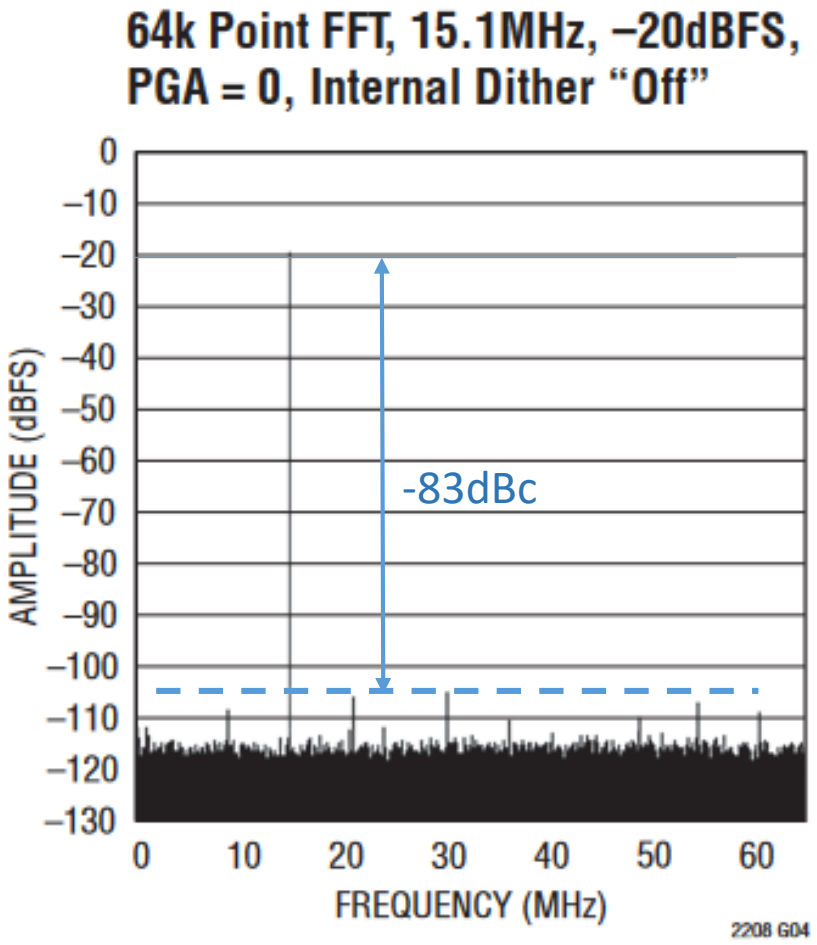
The LTC2208 is perfect for demanding communications applications, with AC performance that includes 78dBFS Noise Floor and 100dB spurious free dynamic range (SFDR). Ultra low jitter of 70fs_{RMS} allows undersampling of high input frequencies with excellent noise performance. Maximum DC specs include ±4LSB INL, ±1LSB DNL (no missing codes).

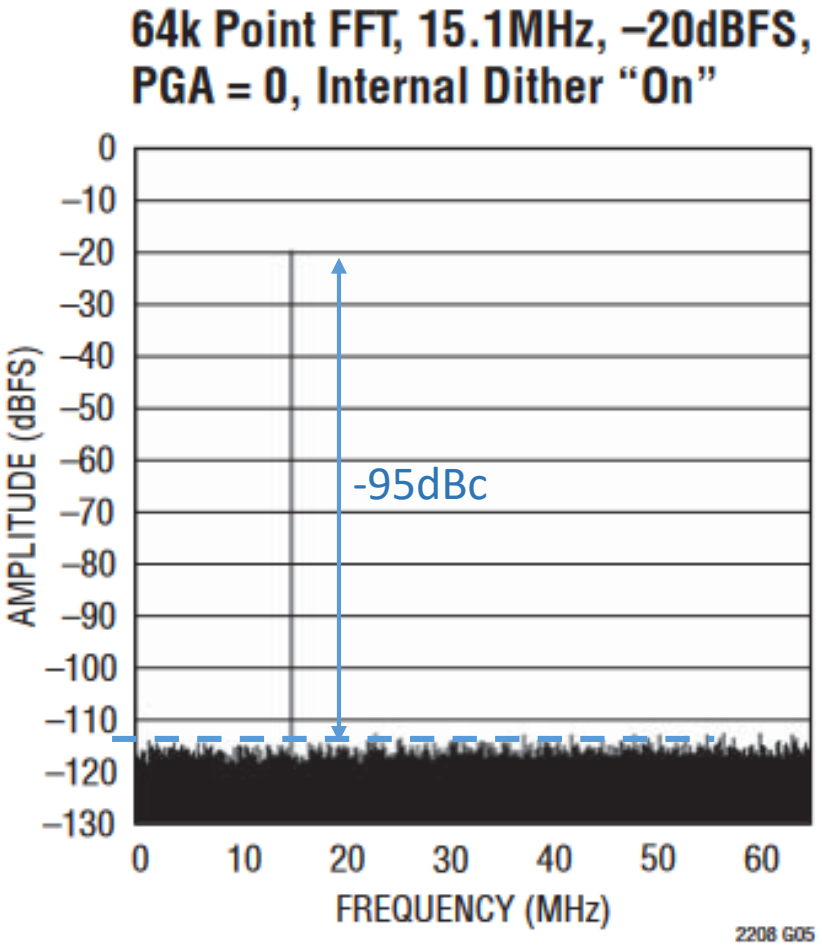
The digital output can be either differential LVDS or single-ended CMOS. There are two format options for the

ADC van de IC-7610

Dithering on/off vergelijking

- PGA = programmable Gain amplifier
- dBFS = dB t.o.v. Full Scale
- dBc = dB t.o.v. de gestuurde carrier





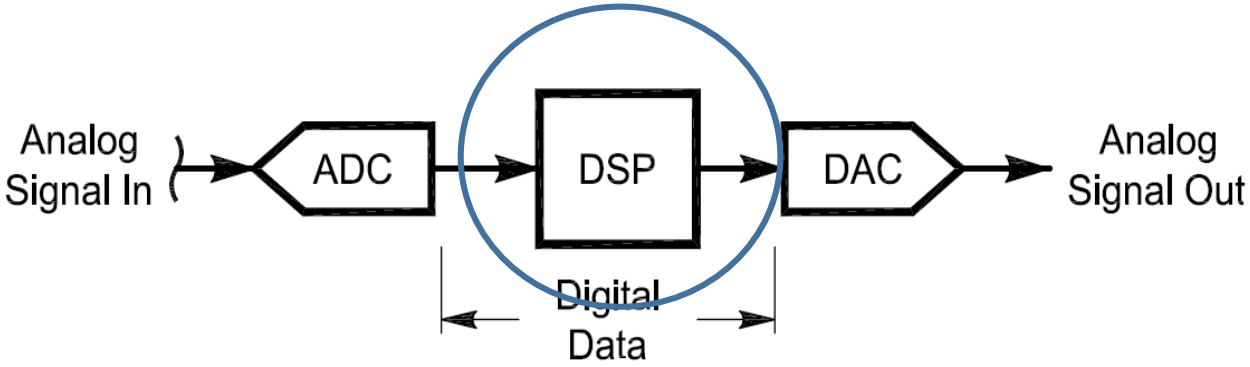
Take away: ADC is dus zeker niet ideaal…

- Aantal detectie bits is een gegeven.
- De ADC meet niet 100% juist, zie o.a. sheetje over sampling fouten, het jitter verhaal en de non-linearity, etc.
- De anti aliasfiltering is niet ideaal
- Dithering helpt het Spurious Free Dynamisch bereik van de ADC

- Extra ruis door de hardware om de ADC heen (filters, pre-amps, etc)
- Praktijk meting voor S/N: **ENOB** – Effective Number Of Bits
- Definitie = (de echte gemeten SINAD (in dB) – 1,76dB) / 6,02
  zo kan een 16 bits DAC een ENOB hebben van 14,5

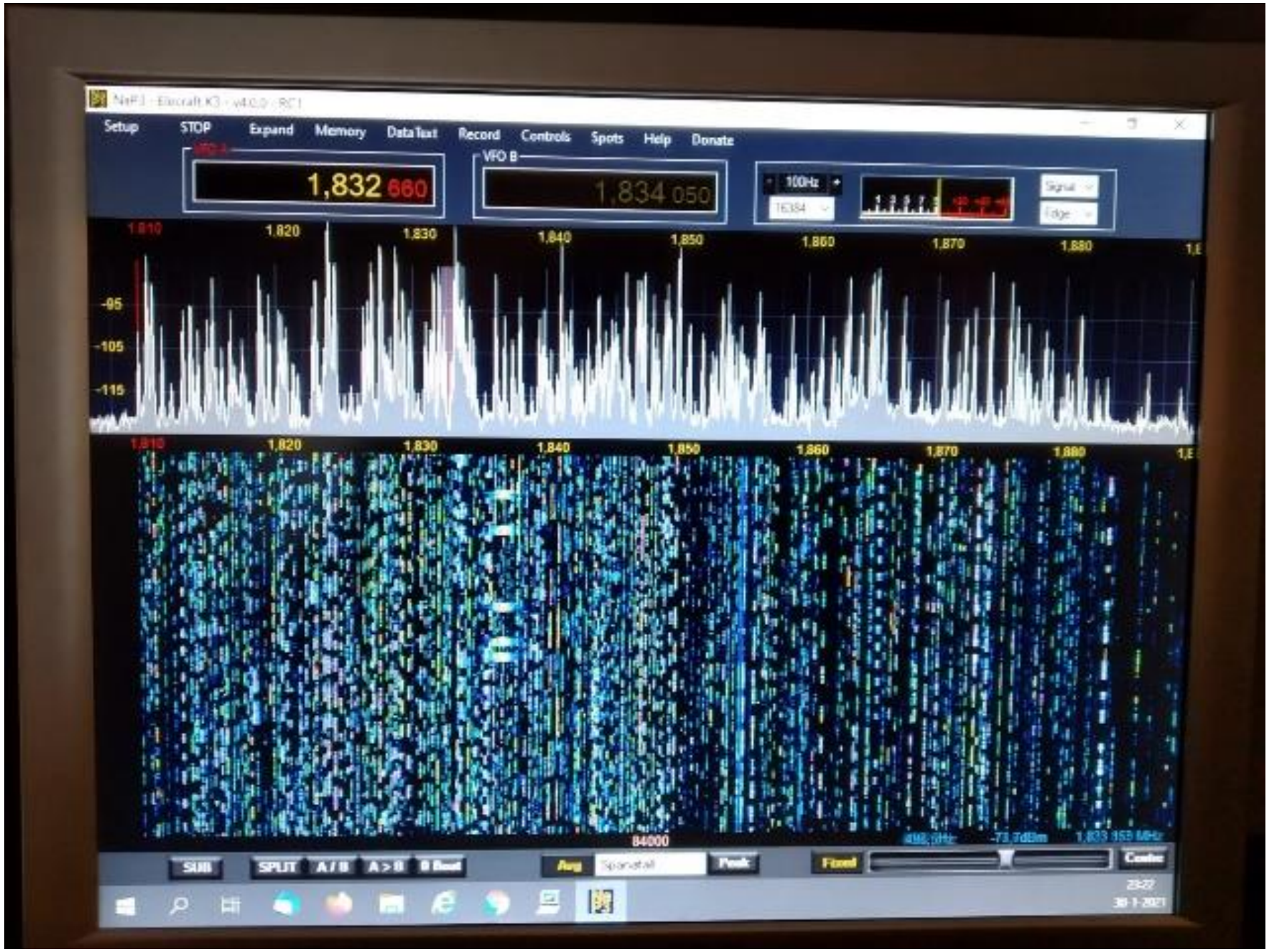
2. SDR → de DSP

(d.w.z. de DSP berekening bepaalt ***voor een groot deel*** wat voor radio je hebt)

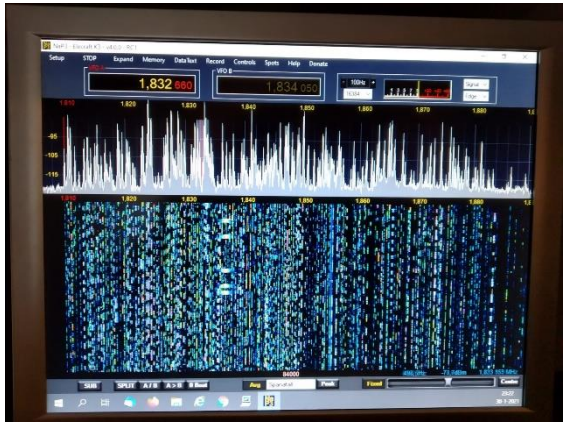


- De FFT – panadapter –
  (deze vinden we mooi)
- Verschillende sample rates in de DSP
- Rekennauwkeurigheid
- I-Q signaal, wat is dat
- Positieve en negatieve frequenties
- Verschillende filter typen

Een FFT scherm



De Fast Fourier Transform - FFT - (wie kent hem niet)



Fourier Transform:
Vanuit een beschreven signaal in de tijd (tijdsdomein) naar een frequentie spectrum.
Andersom heet het de Inverse Fourier Fransform

De Familie van Fourier transformaties:

1. Fourier Transform – voor niet-periodieke signalen
2. Fourier Series – voor periodieke signalen (zich oneindige herhalende signalen)
3. Discrete Time Fourier Transform – digitaal, niet periodiek
4. Discrete Fourier Transform (DFT) – *digitaal wel periodiek* (= onze interesse)
   Een slimmere en snellere rekenmethode van de DFT is die FFT maar doet verder hetzelfde.

- Het principe van de **FFT** werd al gevonden door Gauss in 1805; in 1965 publiceerden Cooley & Tukey de methode voor het gebruik met een computer.
- De FFT is zo belangrijk geworden dat het rekenwerk tegenwoordig hardware gecodeerd in chips zit.

Even inzoomen…

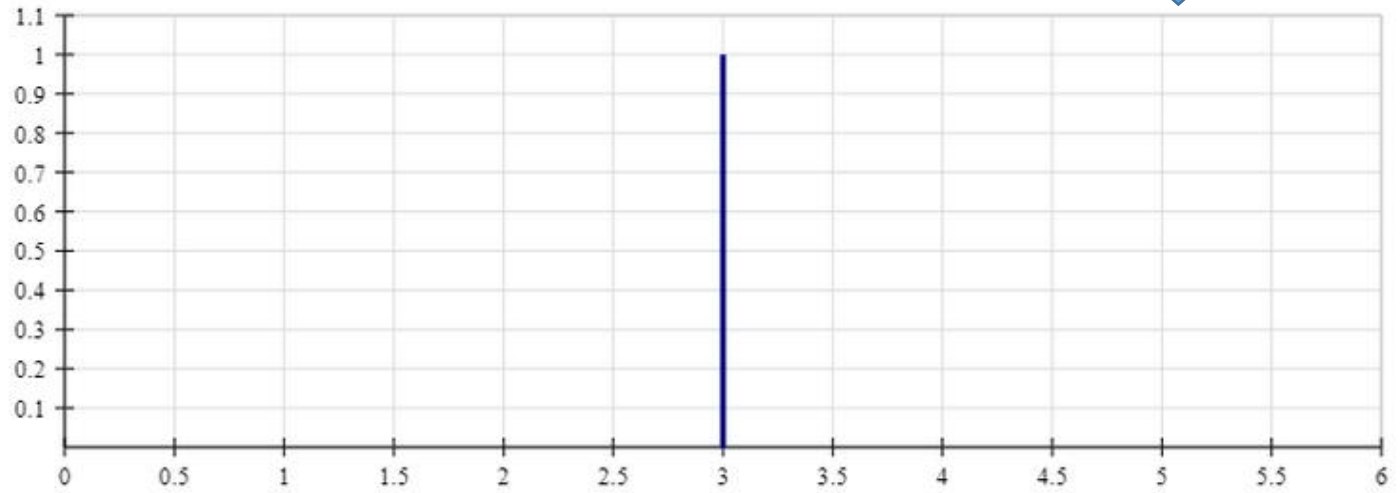
- Signaal: 3 Hz
- 512 samples
- Begin sluit aan bij eind

- Precieze uitkomst:
  - 256 bins gevuld (= ½ x sample rate)
  - 1 lijn op 3 Hz.

- Hoe meer samples, des te meer en des te smaller de bins (en des te dieper de ‘FFT scherm ruisvloer’)

Un-windowed 3Hz Cosine Wave

FFT

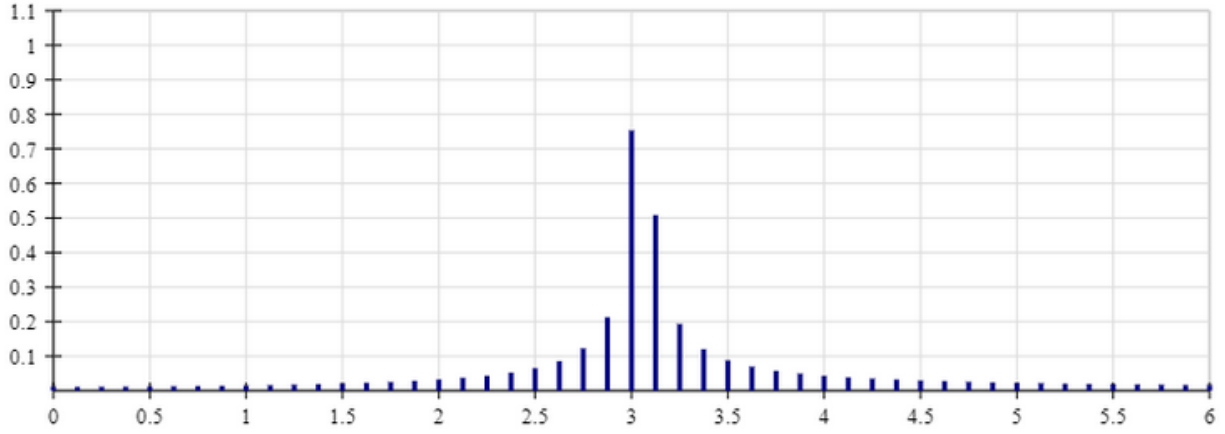


Frequency Spectrum of the Un-windowed signal

- Signaal: 3,05 Hz
- Ook 512 samples
- Begin sluit NIET aan bij eind

- Precieze uitkomst:
  niet helemaal zichtbaar en
  erg breed
- Dit heet
  ‘Spectral Leaking’

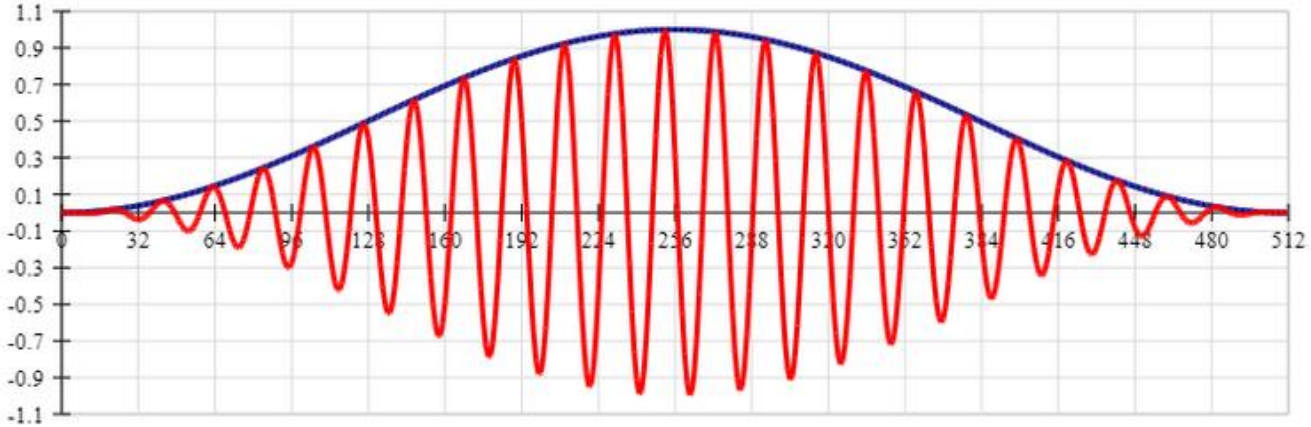
A 3.05Hz Cosine Wave in the Time Domain



Spectral Leakage in a 3.05Hz Cosine Wave in the Frequency Domain

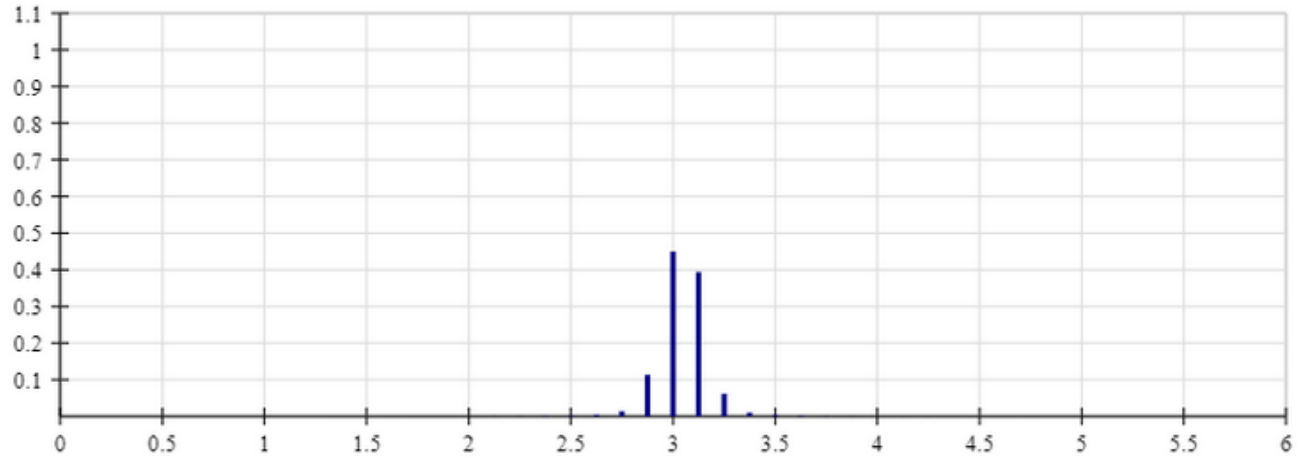
Toepassing van een ‘window’

- Signaal: 3,05 Hz
- Ook 512 samples
- Begin sloot NIET aan bij eind
- We passen een ‘Hanning Window’ toe → nu sluit begin wel aan bij einde



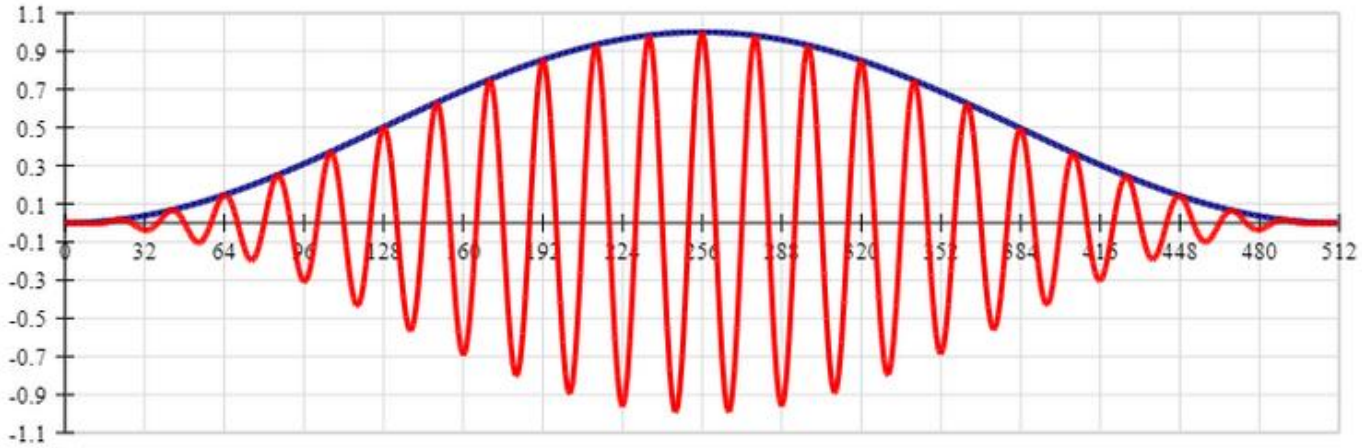
A 3.05Hz Cosine Wave with a Hanning Window applied

- Precieze uitkomst:
  niet helemaal zichtbaar, maar beter dan zonder window
- Spectral Leakage is opgelost
- De top is verbreed. Dit heet: ‘Smearing’



The frequency spectrum of a 3.05Hz Cosine Wave with a Hanning Window applied

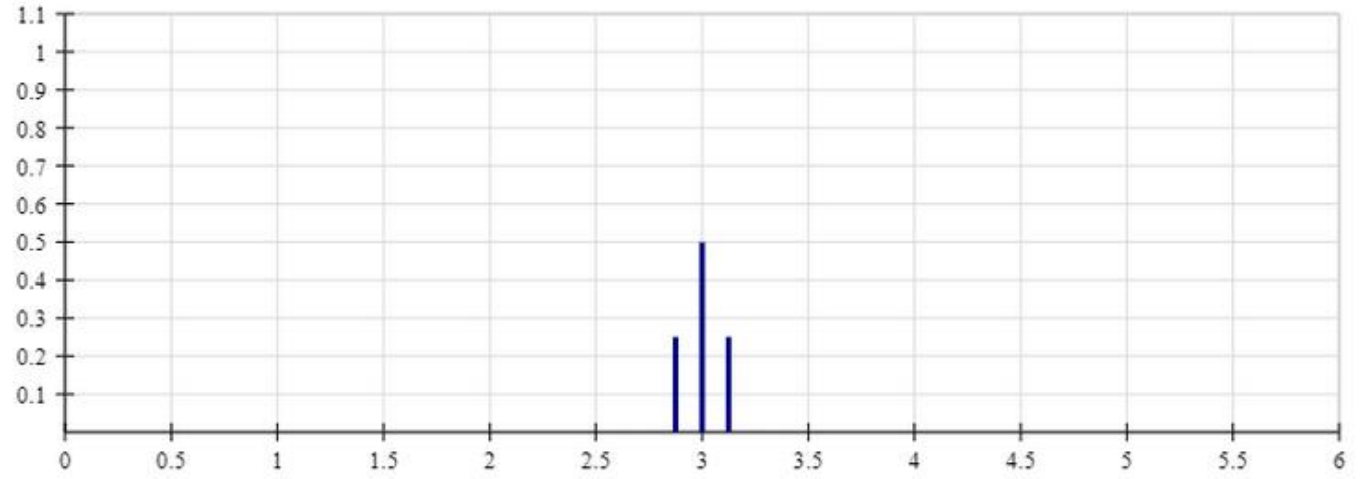
- Signaal: 3 Hz
- 512 samples
- Begin sluit aan bij eind dus goed voor FFT
- Nog steeds met Hanning window



- Redelijk precieze uitkomst: 1 lijn op 3 Hz, plus 2 kleine
- Het window zorgt onnodig voor Smearing in dit geval

Windowed 3Hz Cosine Wave

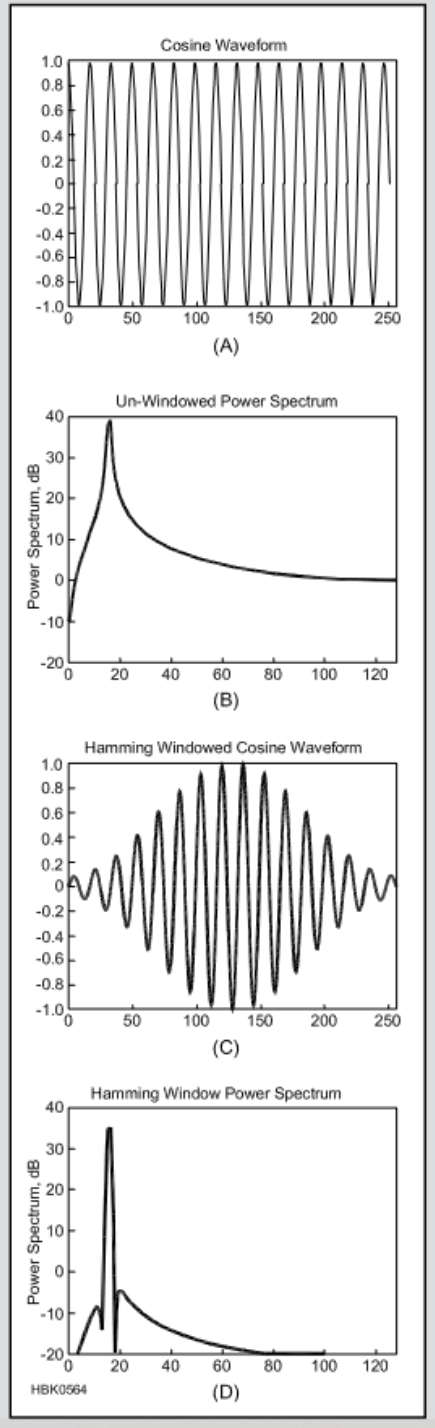
"Ieder voordeel hep zun nadeel"



Frequency Spectrum of the Windowed signal

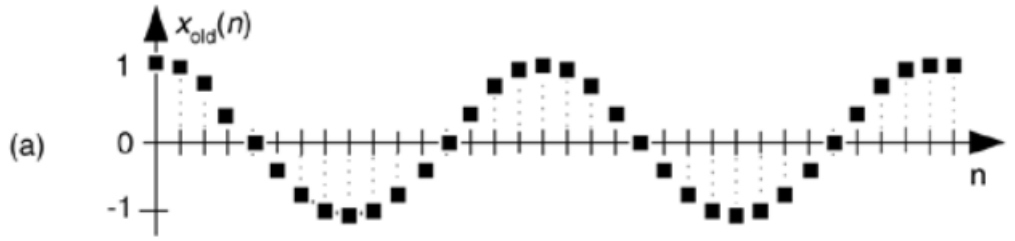
Take away: FFT en foutcorrecties m.b.v. ‘*windows*’

- De FFT gaat uit van periodieke signalen die zich tot in het oneindige herhalen. Onze sample-series voldoen daar niet aan. De laatste samples sluiten niet mooi aan op de eerste. Gevolg is een onnauwkeurige/foute weergave van het frequentiespectrum (b.v. ‘spectral leakage’)
- Om dat te compenseren zijn er zgn. ‘windows’ die de uiteinden van de sample-serie in waarden naar elkaar brengt voordat een FFT berekening wordt gedaan. Er zijn er verschillende windows met ieder eigen bij-effecten.
- Bekende windows zijn (genoemd naar de ontwerpers):
  - Blackman
  - Hamming
  - Hanning
  - Bartlett
  - Flattop
- Hoe meer samples de FFT gebruikt, des te nauwkeuriger is het frequentiespectrum dat eruit komt.

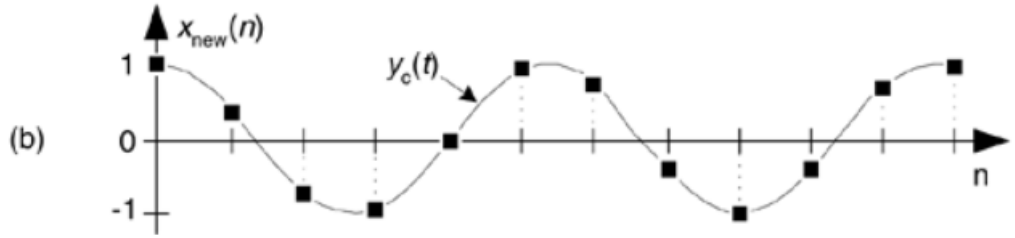


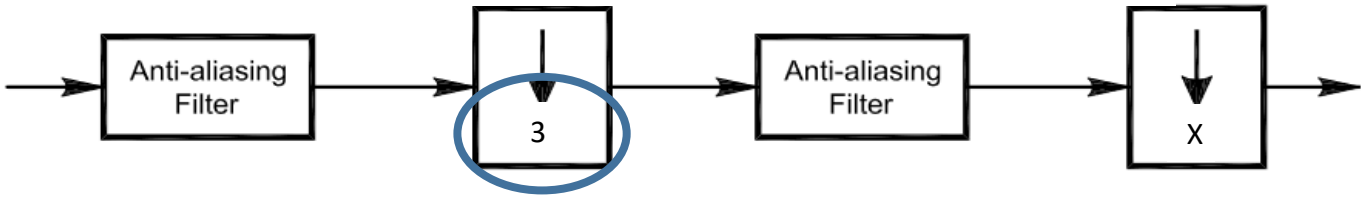
Sample rates: hoe hoog of hoe laag

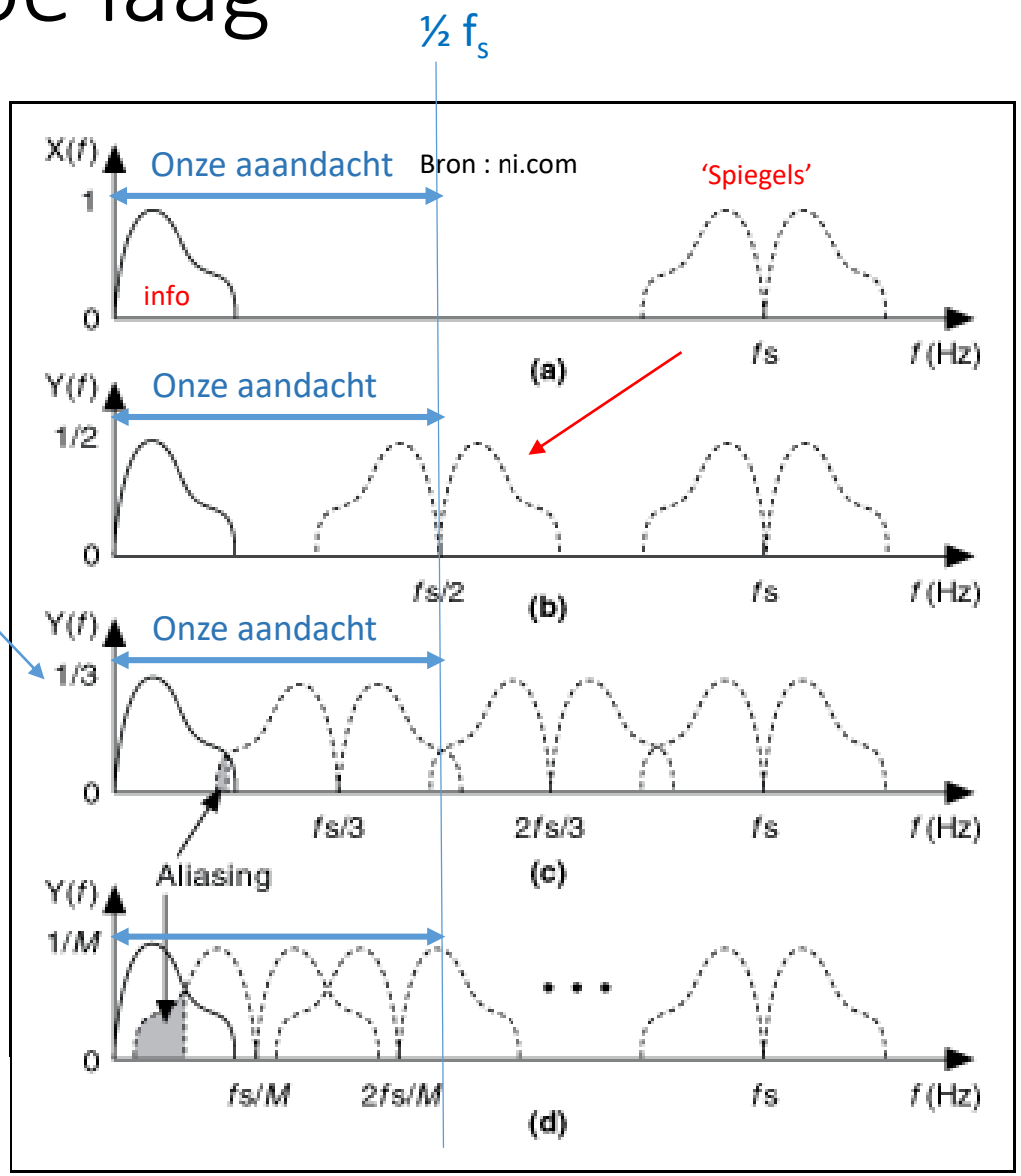
- Hoge sample rate: niet handelbaar bij uitgebreide berekeningen, daarom intern na eerste sampling z.s.m. de sample rate omlaag → '***Decimate***' of 'Decimeren'
- ECHTER Let op: er kan aliasing ontstaan
  Dit kan ondervangen worden door een digitaal low pass filter (dus extra rekenwerk) **vooraf** aan de Decimate.



Delen door 3 → 2 op de 3 samples laten vallen

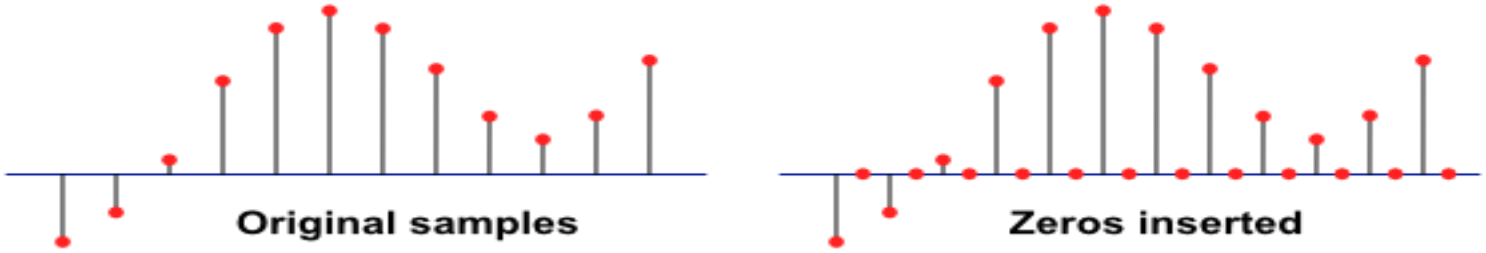


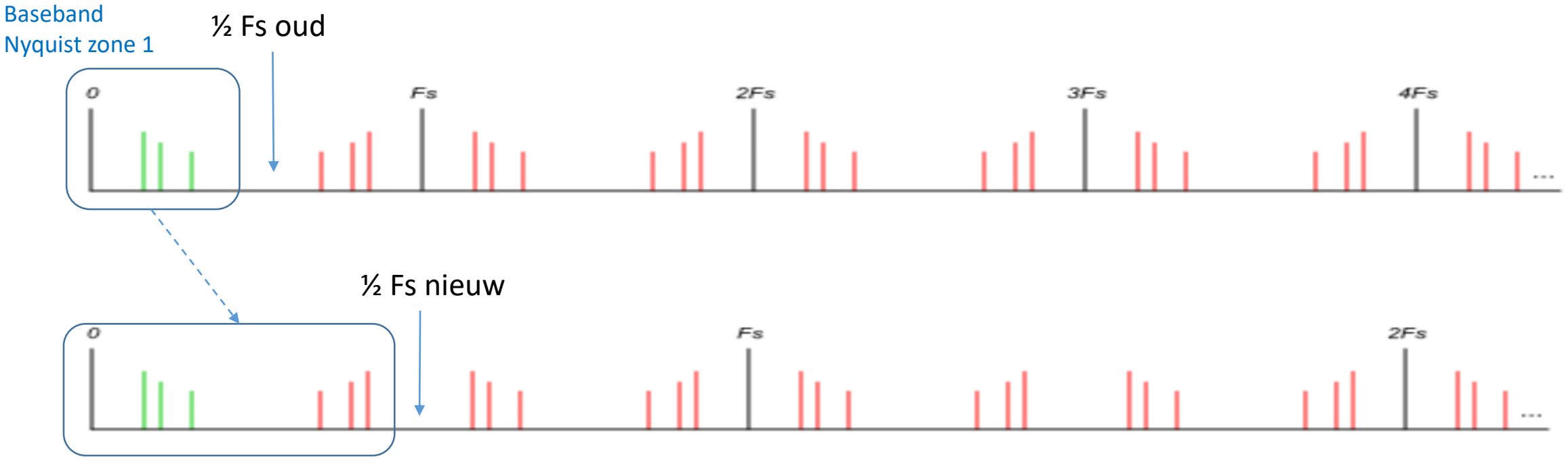




Samplerate omhoog: Interpoleren / **Interpolate**

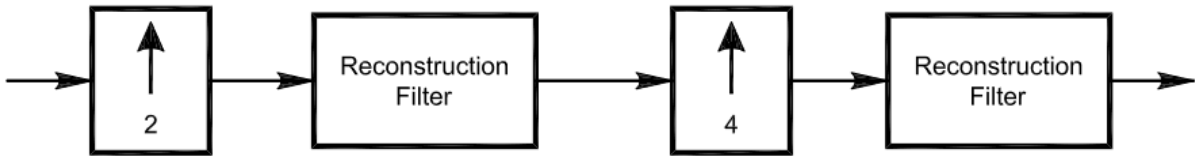
- Soms wil je een hogere samplerate maken (**Interpolate**), b.v. om een audio signaal met een lage samplerate te mengen met een digitaal hf signaal met een veel hogere samplerate.
- Er worden dan tussen iedere twee audiosamples extra samples toegevoegd met waarde = 0
  Dit heet *zero-stuffing* en heeft effect op het spectrum

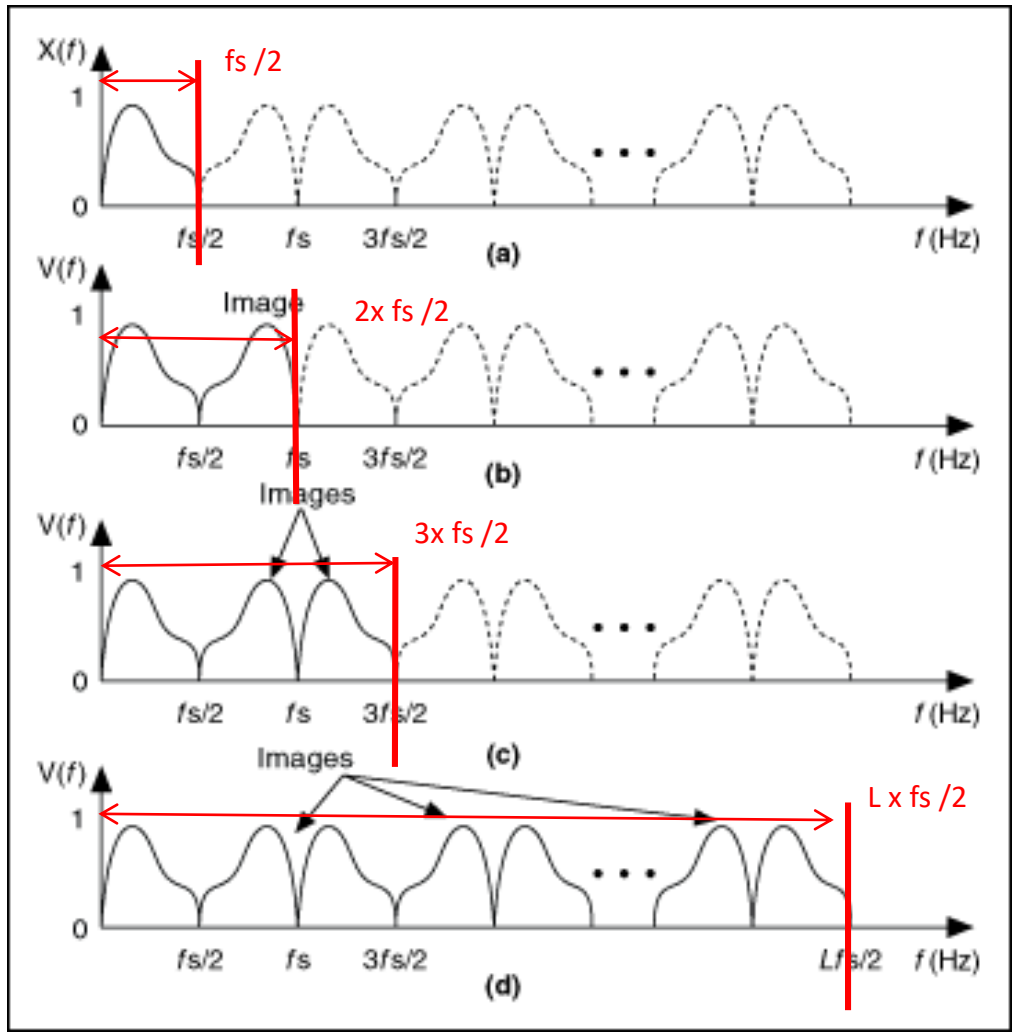




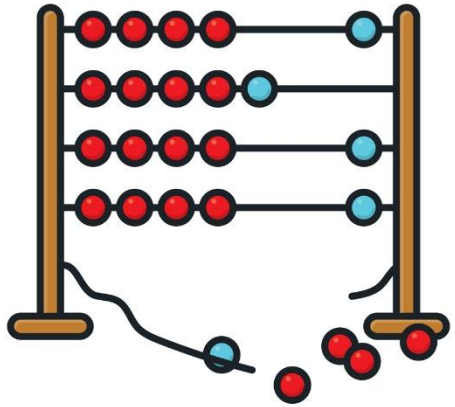
Samplerate omhoog: Interpoleren / **Interpolate 2**

- Effect op het digitale spectrum.
  Spiegels waar je eerst geen last van had komen zo vanzelf in het baseband spectrum (Nyquist zone 1)
- Filtering is daarom nodig **na** de Interpolatie





Onnauwkeurigheid

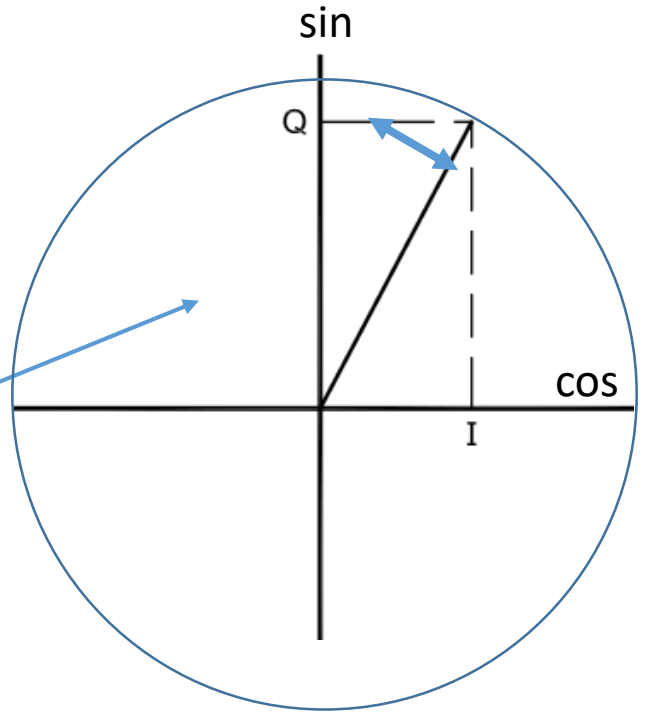


Meerdere bronnen van onnauwkeurigheid in DSP:

- De reeds **kwantiseringsfouten** beschreven ADC en DAC processen
- Het rekenwerk DSP wordt gelimiteerd door de zgn. ***woordlengte***
  - In den beginne al door het aantal bits waarmee wordt gesampeld
  - Het aantal bits / de nauwkeurigheid waarmee wordt gerekend /afrondingen
  - Eventuele grenzen aan de processor
  - Het aantal bits voor de weg terug naar analoog ADC

  Uiteindelijk komen de fouten terug als random noise
- Het **aantal samples waarmee wordt gerekend**:
  hoe meer samples in de berekening, des te beter onderscheid in de verschillende frequenties (b.v. FFT: hoe meer samples in de berekening, des te gedetailleerder het beeld)

I en Q (= In fase en Quadratuur - 90gr verschoven)



- Waarom is dat van belang?
  - **I = In fase Q = Quadratuur fase** (90 graden verschoven)
  - **I** = COSINUS, **Q** = SINUS
  - Combinatie van de twee: I + jQ
    (j betekent: 90 graden verschoven)

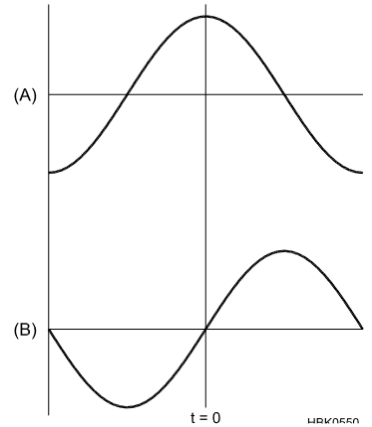
  - $S(t) = I(t) + jQ(t) = \cos(\omega t) + j \sin(\omega t) = e^{j\omega t}$
    Dit is draaiende vector die met tempo ωt rond draait.

  - Wiskunde: I/Q maakt zo onderscheid tussen positieve en negatieve frequenties, dus linksom en rechtsom draaiende vector. Hierover later meer als we inzoomen op SDR.

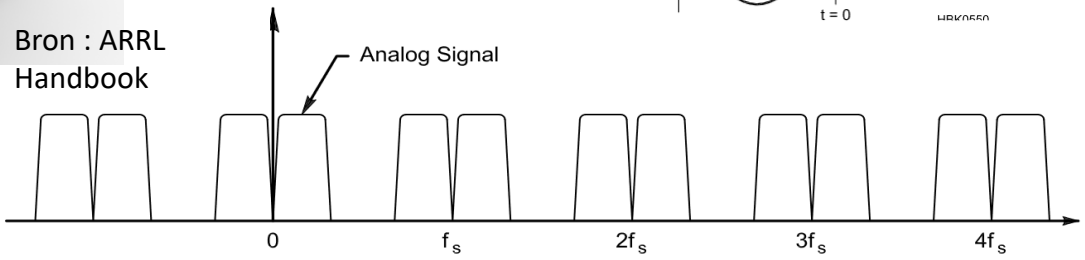
  - In audio I en Q kun je hiermee een heel spectrum doorgeven, b.v. naar een stereo geluidskaart, waarbij 192kHz samplerate in de praktijk een bandbreedte geeft van bijna 192kHz (theoretisch). De eerste SDR's werkten zo. (Theorie: 2 x 192 kS/s = 192 kHz BW en in feite in dit geval -96kHz tot + 96 kHz)

  - Een -90° fasedraaiing over een geheel spectrum is in DSP eenvoudig gemaakt, dankzij meneer Hilbert. Deze methode heet dan ook de '**Hilbert Transform**'.

Negatieve frequenties (wiskundig van belang)



- Denk nog eens na bij een superhet:
  trx IF = 9 MHz en VFO = 5.5 MHz
  Output mixer: 14.5 en 3.5 MHz
  f1+f2 en f1-f2… welke is f1??



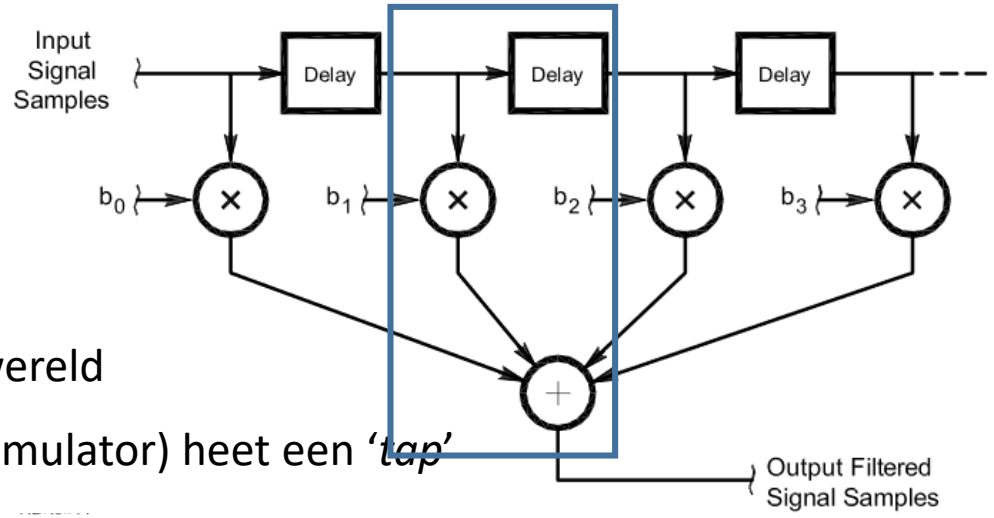
- Plat gezegd: dat maakt niets uit.
  - Een cosinus is symmetrisch rondom de y-as (x=0)
    cos(ωt) = cos (-ωt), tel ze op dan krijg je 2 cos(ωt).
    Een sinus is symmetrisch rondom het punt (0,0)
    sin(-ωt) = - sin(ωt). Tel je deze waarden op, dan is de som nul.

- De DSP rekent voor het gemak en snelheid met ‘complexe getallen’.
  B.v. **$\cos(\omega t) = \frac{1}{2} (e^{j\omega t} + e^{-j\omega t})$** , waarbij die positieve en negatieve frequenties (+ω en -ω) meteen in de berekeningen komen, vandaar die tweezijdige spectra.
  Zoals we al gezien hebben kunnen we positieve en negatieve frequenties in I/Q onderscheiden.

FILTERING

- FIR
- FFT (Fast convolution)
- IIR
- CIC

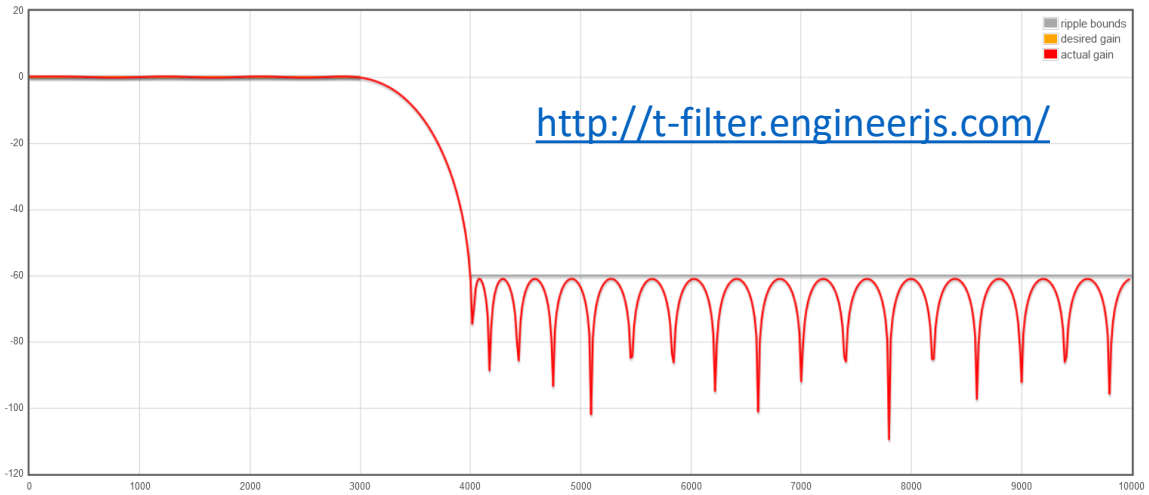
FIR – filter (rekenwijze heet ‘convolutie’)



- **FIR**: **F**inite (=EINDIGE) **I**mpuls **R**esponse i.p.v. ONEINDIGE in de analoge wereld
- Eén delay plus vermenigvuldiging en opteller (**MAC** – **M**ultiplicator/ **AC**cumulator) heet een ‘*tap*’
- Hiervoor heb je nodig de zgn. filter coëfficiënten b_0, b_1, b_2, …etc.
- Deze rekenwijze (schuif het signaal langs de verschillende MAC’s en tel op) heet **‘Convolutie’**
- **Hoe smaller het filter t.o.v. de samplerate**, **des te meer taps** zijn er nodig om een bijna volledige impulsresponsie kwijt te kunnen in de coëfficiënten b_0, b_1, b_2, ….
  De eigenlijke oneindige reeks wordt altijd ingekort (‘truncated’) tot ‘eindige’. Hiervoor wordt weer een ‘window’ gebruikt, net als bij de FFT.

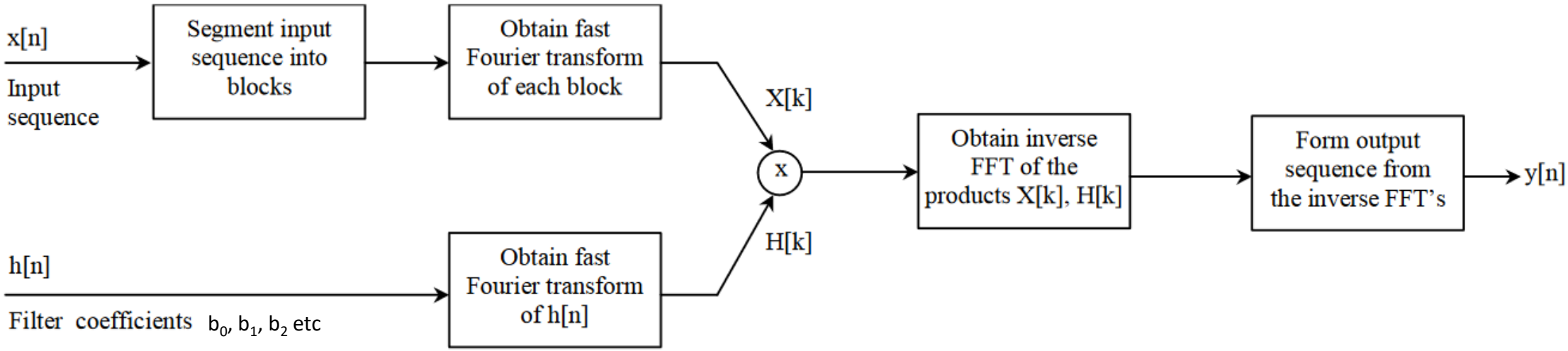
- Voorbeeld:
  F_{sample} = 20 kHz, LPF 0 - 3kHz, stopband begint op 4 kHz, stopbandverzwakking = -60dB

- Benodigd aantal taps ordegrootte: 49



Filtering m.b.v FFT en IFFT (zgn. ‘Fast Convolution’)

- In feite zien we hier de relatie tussen tijd en frequentie.
- Als we het analoge signaal omzetten naar het frequentie domein (FFT), en vermenigvuldigen met de filterkarakteristiek van het filter ontstaat het gefilterd signaal
- Dat kan worden terug gezet naar het tijdsdomein (IFFT)
- Kun je zo een ideaal filter maken met rechte zijden? Nee!
  Er is nog steeds de beperking van de niet oneindige reeksen in de berekeningen.
- Het blijkt in de praktijk (bij veel taps) wel sneller dan de standaard FIR Convolutie berekening



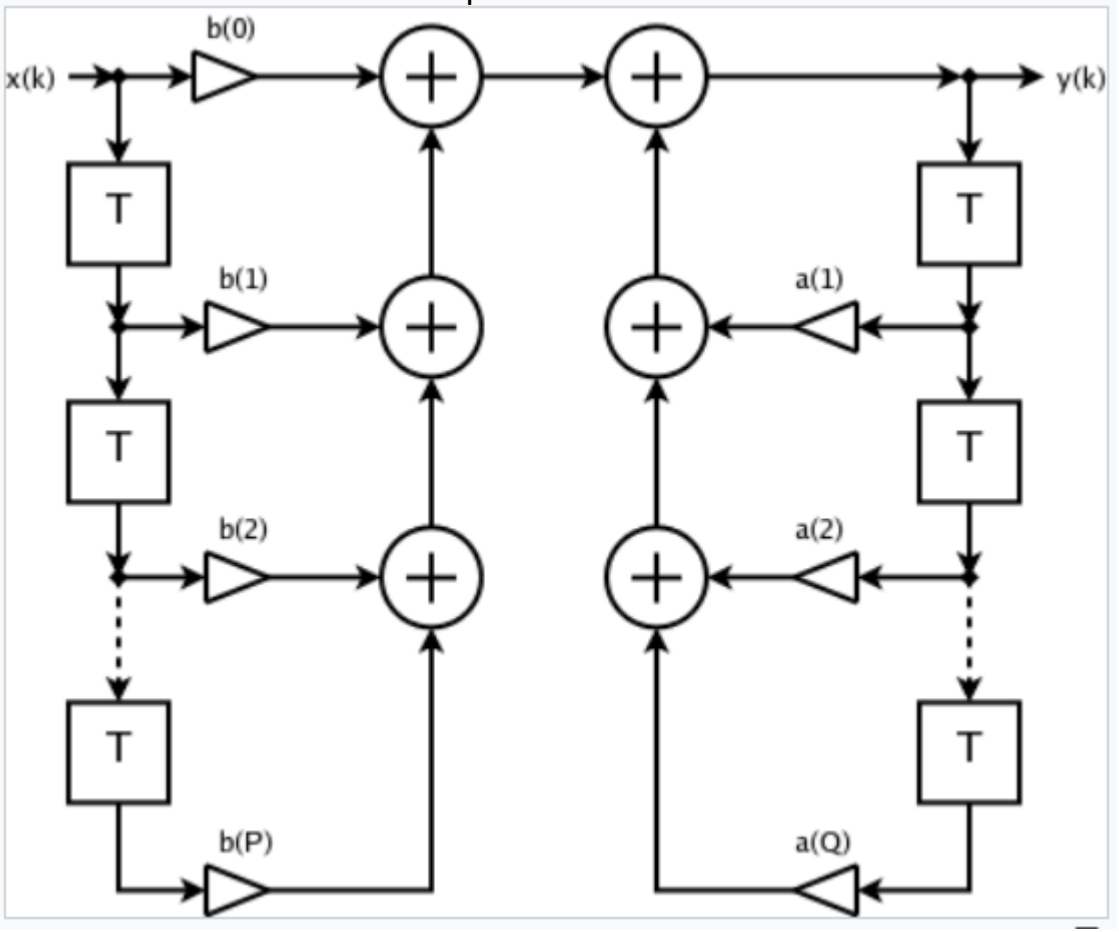
IIR – filter

(**I**nfinite (=ONEINDIG) **I**mpulse **R**esponse)

- Ingewikkelder ontwerpproces dan FIR
- Uitgebreid rekenwerk gaat vooraf
  (Laplace- en Z-transformaties). Dit ligt ver buiten het bestek van de lezing.
- Tools: o.a. Matlab

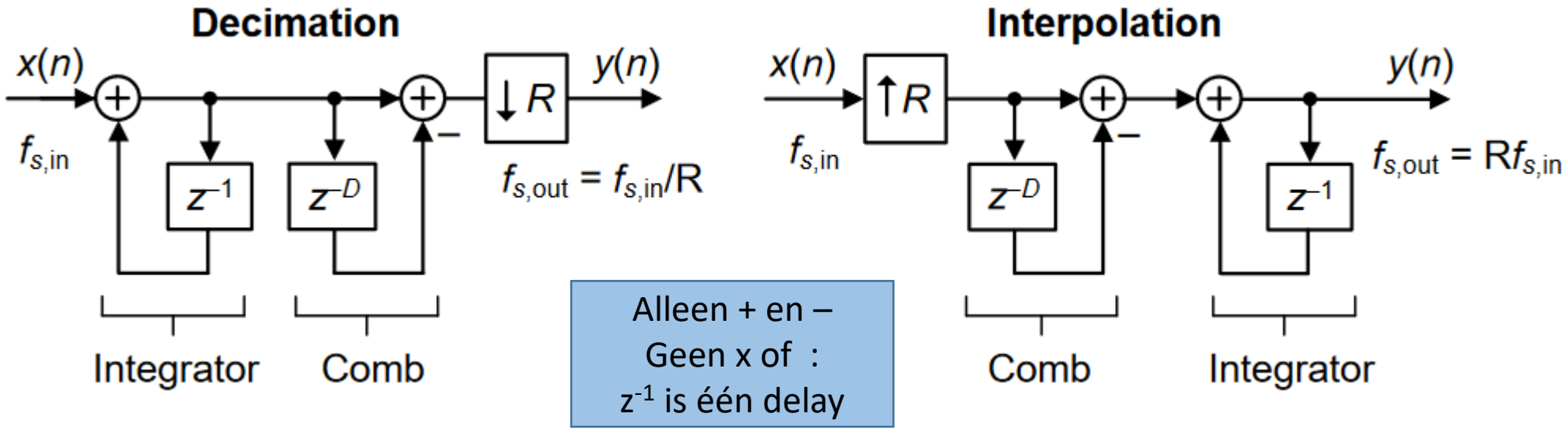
- Je kunt met weinig MAC's een scherp filter maken
- Nadeel: Kans op instabiliteit, immers de output wordt deels als input gebruikt en kan oscillatie ontstaan.
  Dit kan gebeuren bij grote signalen, maar ook bij kleine signalen door simpele afrondingsfouten.

Bron: Wikipedia



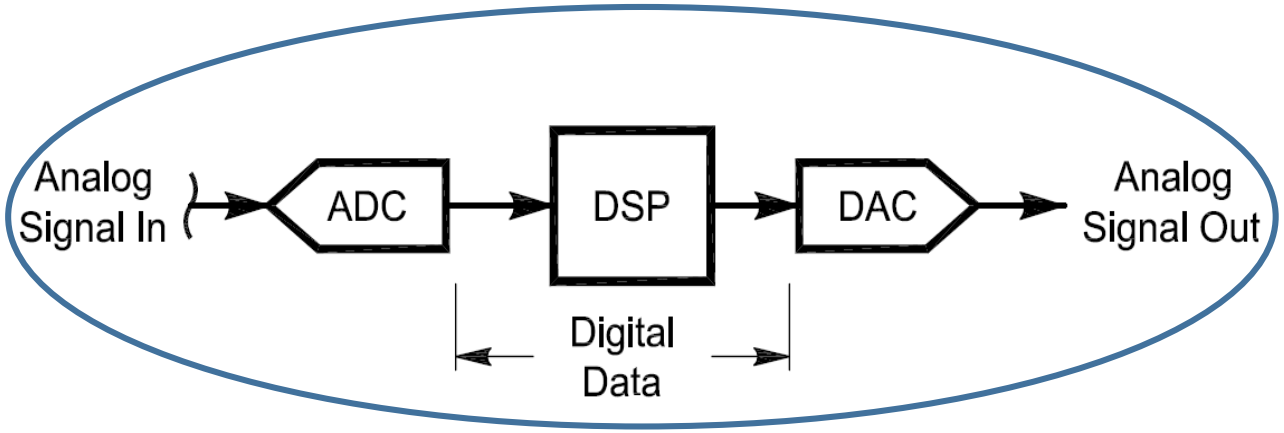
Zgn. ‘CIC’ filters bij decimeren interpoleren

- CIC staat voor **C**ascaded **I**ntegrator **C**omb, een efficiënt **Low Pass** Filter
- Zeer goed voor anti-alias bij decimeren en anti-spiegel bij interpoleren
- Vaak staan er meerdere filters in serie.
- **Voordeel: alleen + en - , geen x of : en dus weinig processorcapaciteit nodig.**
- Omdat deze filters een sinc-achtige response hebben, wordt een simpel FIR filter toegevoegd om dat te compenseren aan de kant waar de sample rate *het laagst* is!



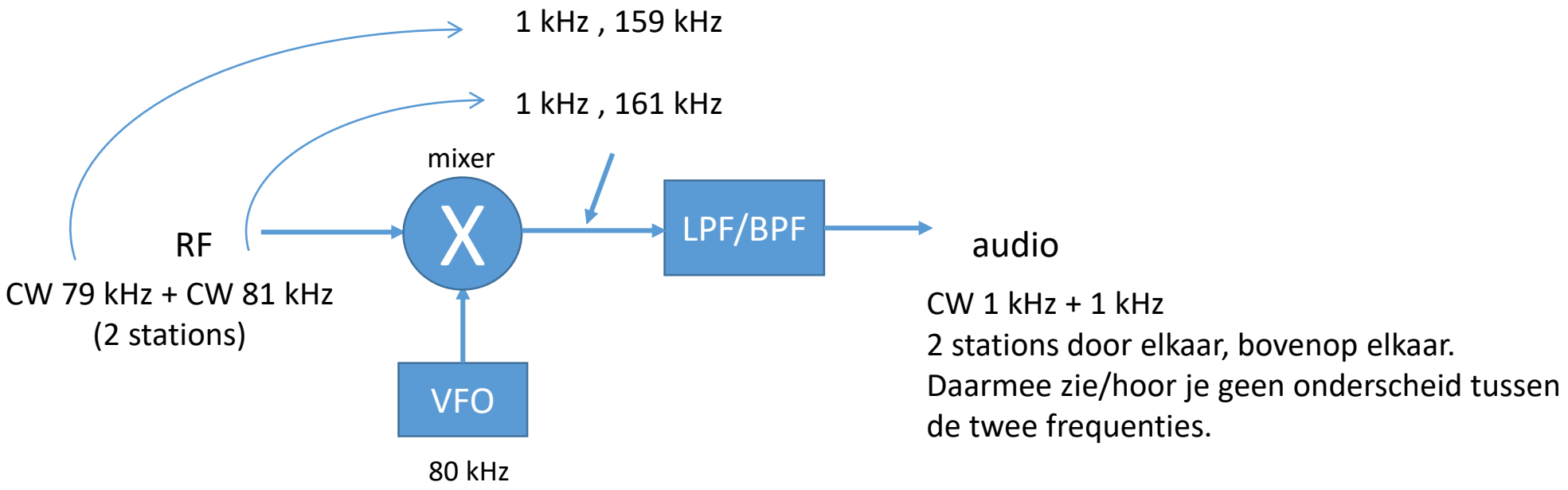
3. De TRX, wat er in een SD-RADIO gebeurt

- Voor ons interessant: communicatie-modes SSB, CW en FM detectie
- Verschillende TRX:
  - RTL SDR stick
  - Yaesu FTDX-10
  - Yaesu FT-710
  - ICOM IC-7300
  - Flexradio Flex-6700 – 144 MHz



Een Direct Conversion (DC) analoge ontvanger 1

- Essentie: meng de RF naar audio met een VFO en filter het audiosignaal.
- Simpel en doeltreffend, maar geen spiegelonderdrukking b.v. tussen signaal +1 kHz en -1 kHz t.o.v. de VFO frequentie.



Een DSP versie van de DC ontvanger - 1

DSP maakt gebruik van I en Q signalen (Q signaal is 90° verschoven)

Dus met cosinus en sinus, die zijn ook 90° verschoven.
Klein beetje belangrijke wiskunde uit de hogere DSP rekenschool:

$$\sin \omega t = (e^{j\omega t} - e^{-j\omega t})/2j \quad \text{en} \quad \cos \omega t = (e^{j\omega t} + e^{-j\omega t})/2$$

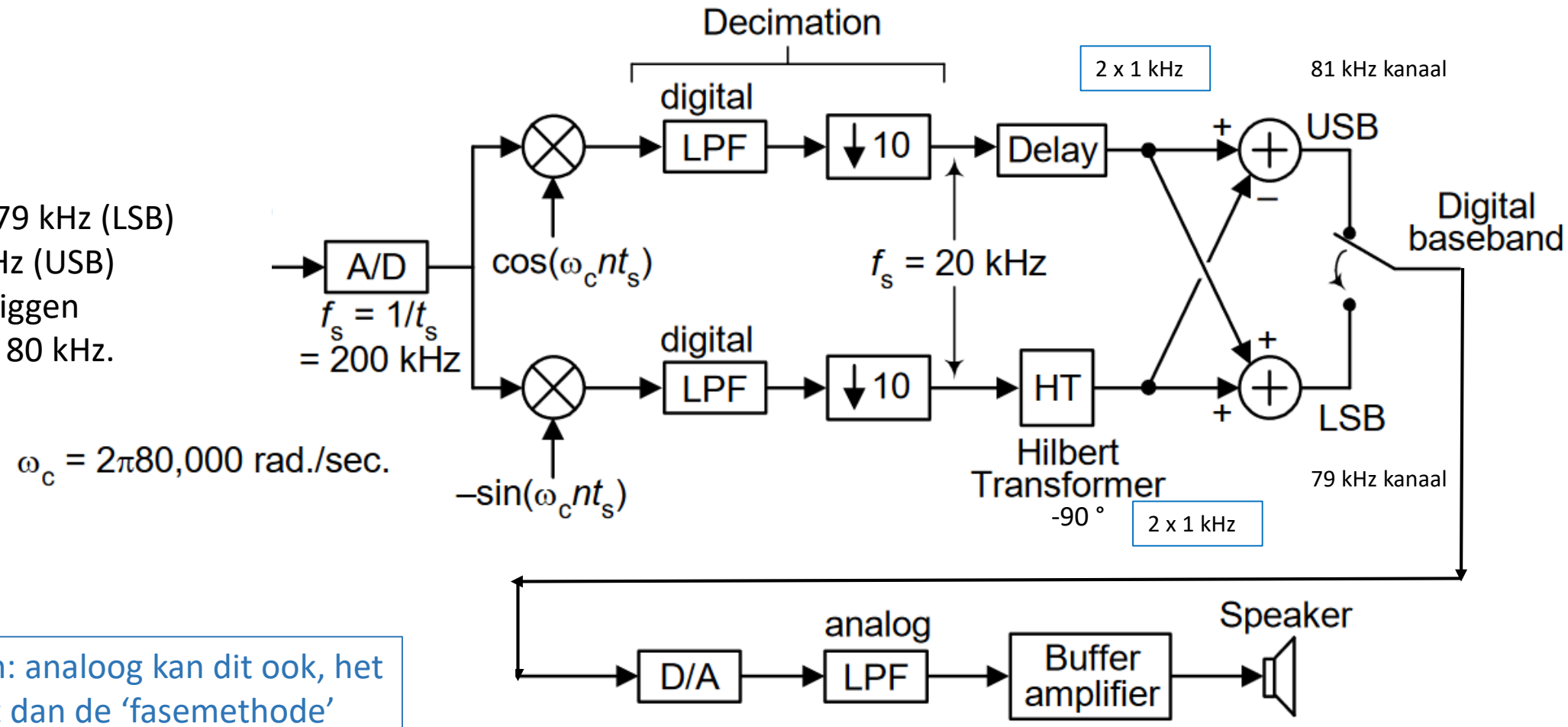
$$(\omega = 2\pi f)$$

Met al deze plussen en minnen met j = 90° draaien
en j x j = -1, kun je je dan voorstellen dat:

Een DSP versie van de DC ontvanger - 2

- De toepassingsvoorbeeld in DSP

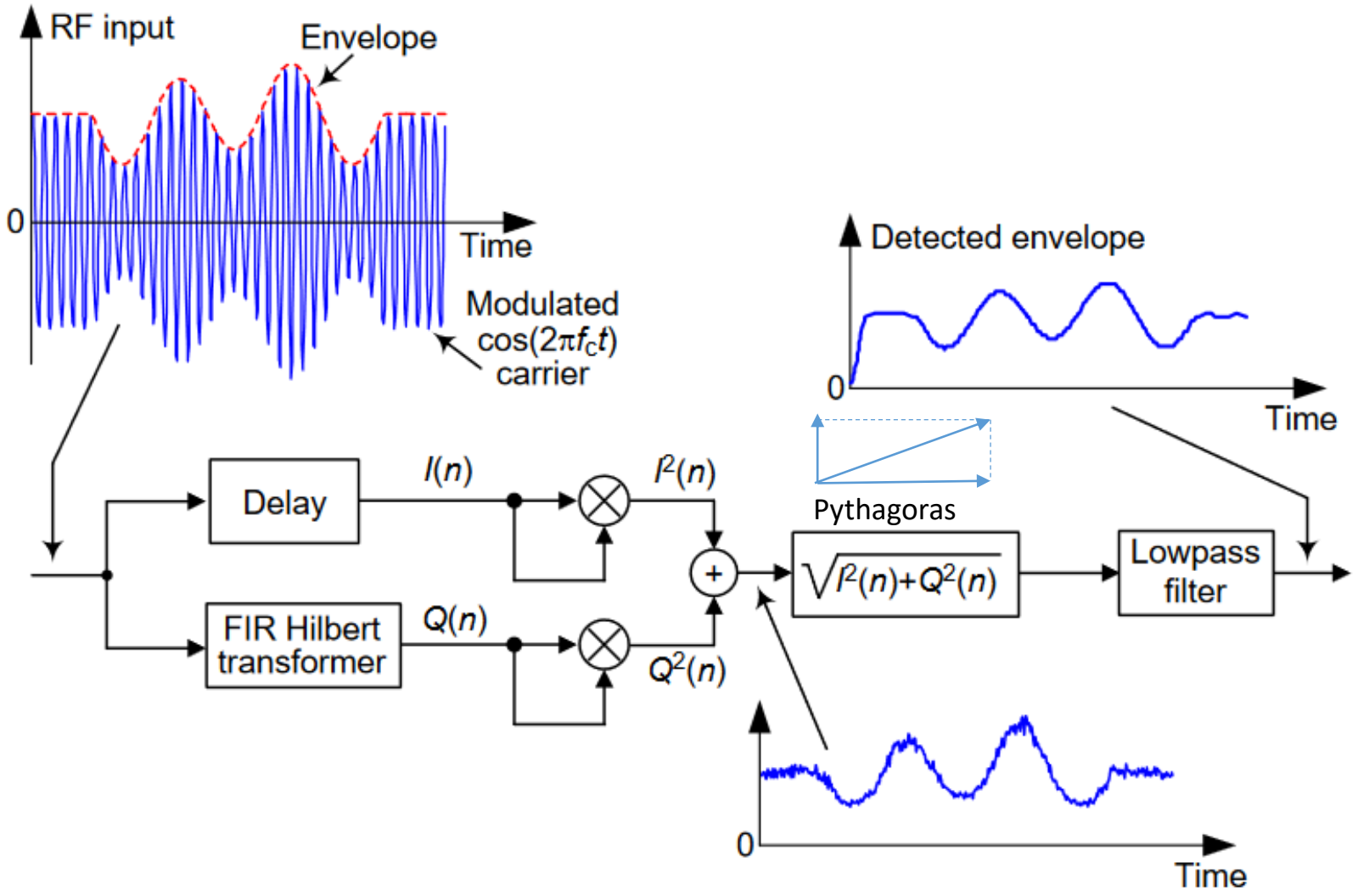
Signaal 79 kHz (LSB)
en 81 kHz (USB)
Beiden liggen
rondom 80 kHz.



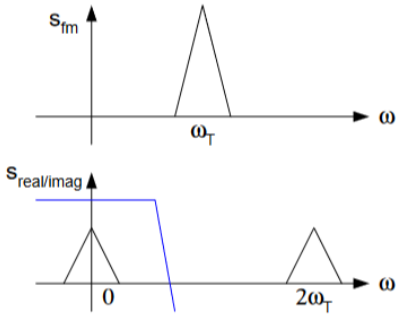
Opm: analoog kan dit ook, het heet dan de ‘fasemethode’

DSP AM detector

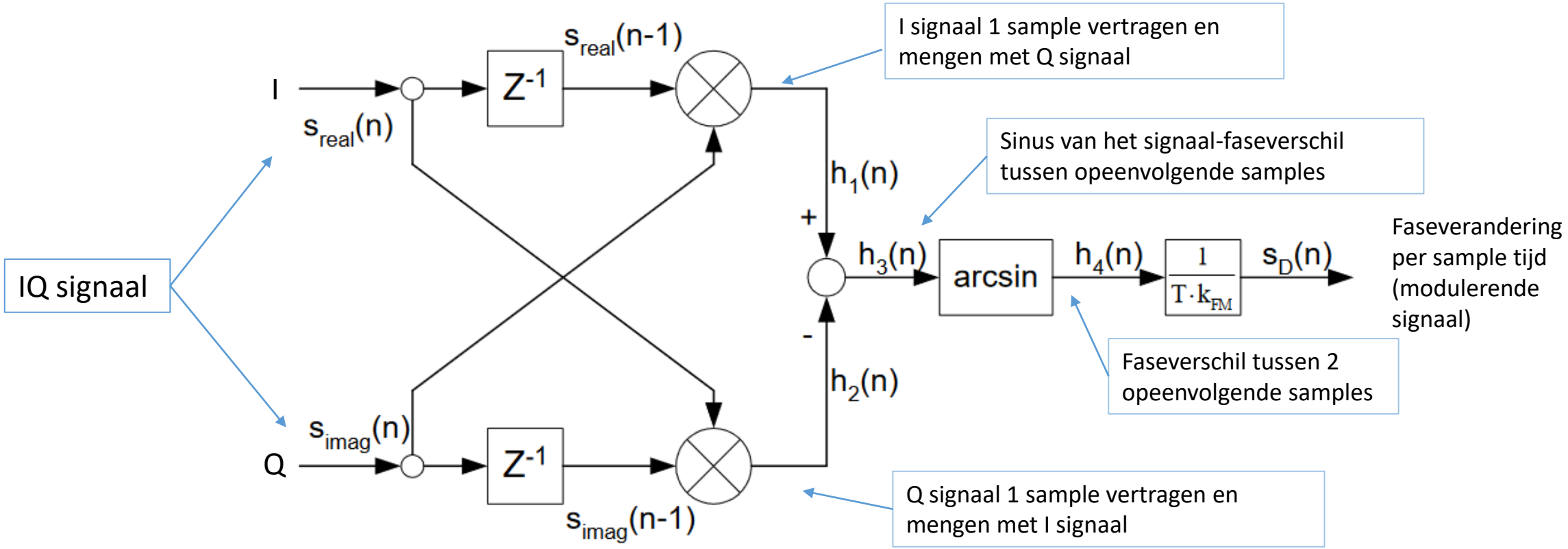
Pythagoras did it again! ☺



DSP FM detector



- Deze detector werkt in ‘baseband’, dus is al quadratuur voorgemengd, zie ook zijbandontvangst in de Direct Conversion ontvanger.

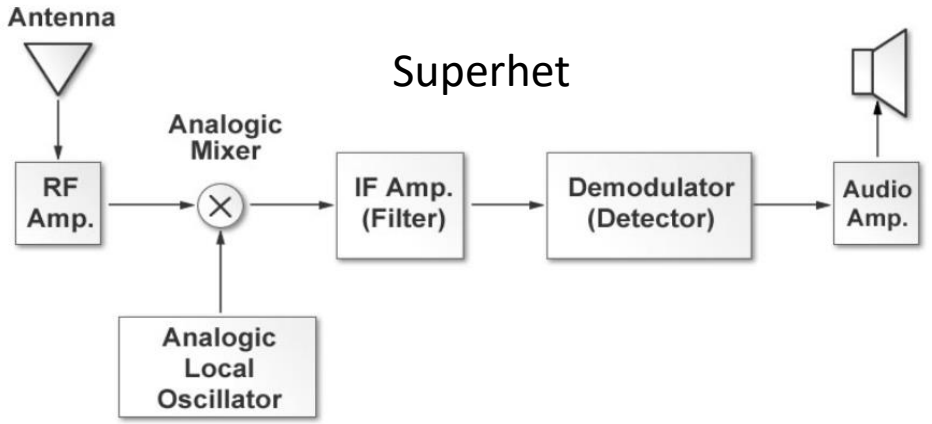


Bron: Implementation of FM Demodulator Algorithms on a High Performance Digital Signal Processor, Franz Schnyder en Christoph Haller

Onze praktijk, verschillende (T)RX’s

- RTL-SDR
- FTDX-10
- FT-710
- IC-7300
- Flex-6700 (2m)

RTL-SDR en de Superhet

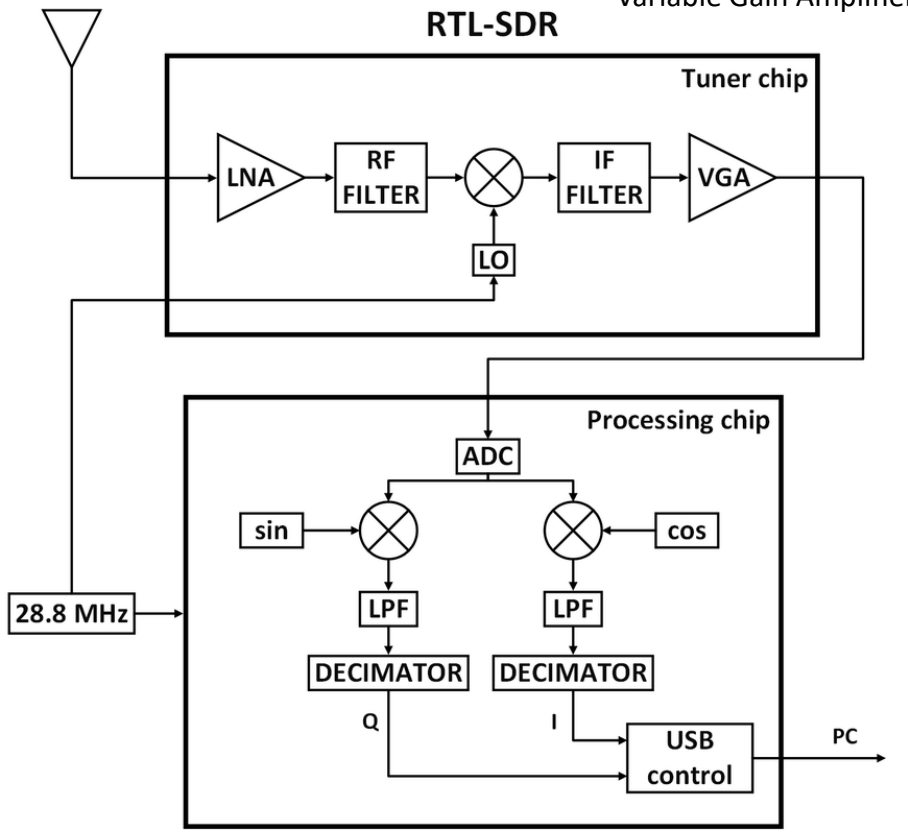


1e stap RTL-SDR is analoog á la een superhet.

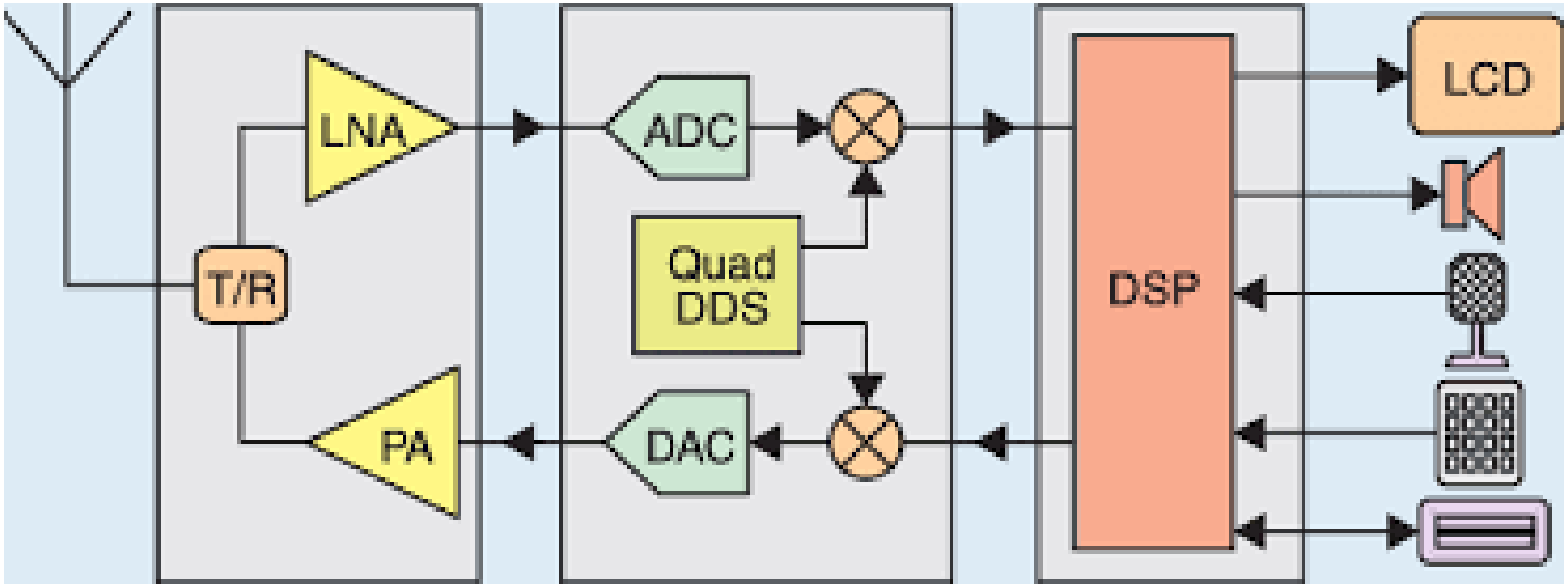
Wie doen dit b.v. ook?
Elecraft K3 (IF = 8.2MHz + 15 kHz)
Yaesu FTDX-101 (IF = 9 MHz)
Yaesu FTDX-10 (IF = 9 MHz)

Bij de RTL-SDR wordt het digitale baseband signaal doorgezet in IQ vorm naar de pc.

VGA =
Variable Gain Amplifier



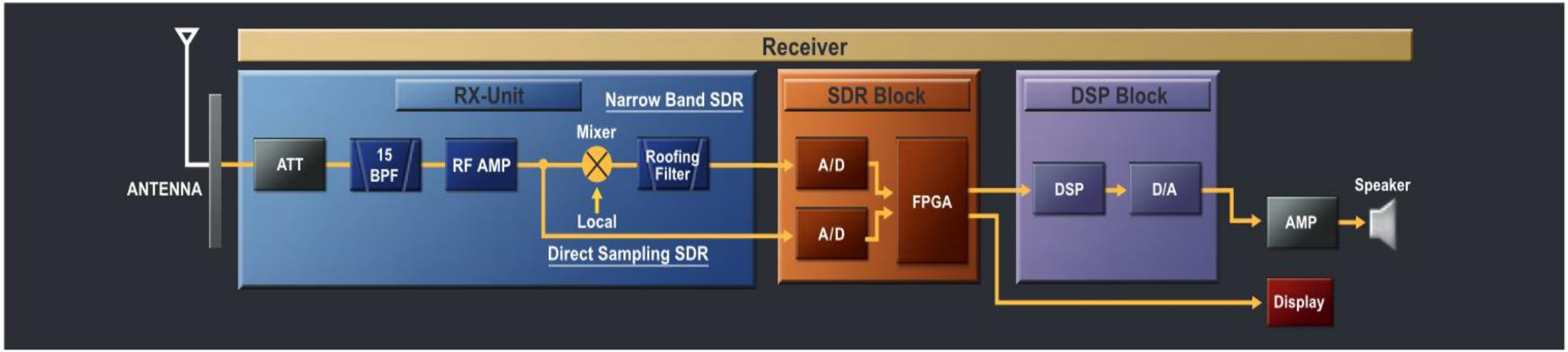
Full digital SDR (erg grof)



Yaesu FTDX-10 (hybrid: heterodyne + SDR/DSP)

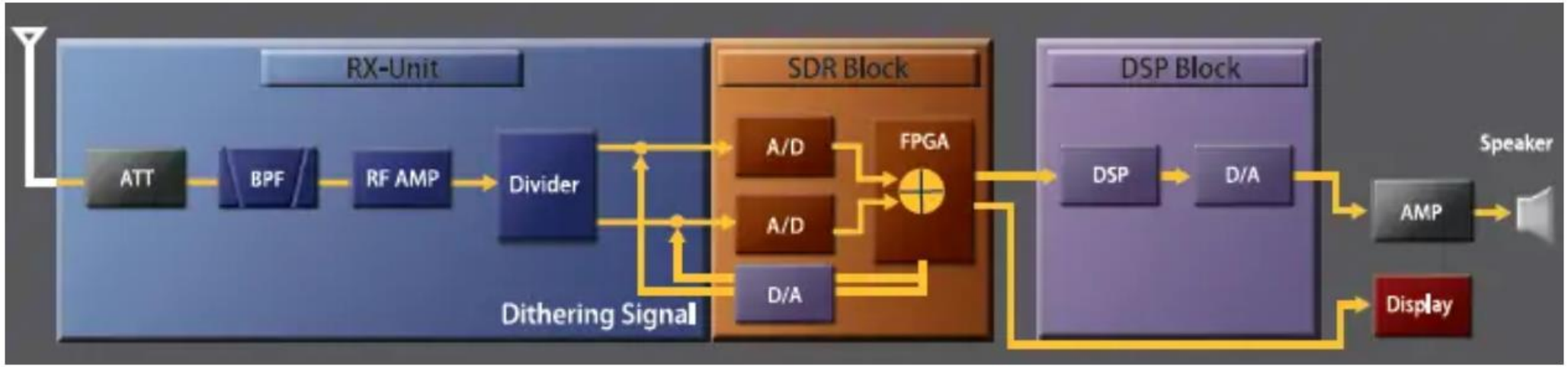
- Is een superhet met deel SDR, vanaf IF (= 9MHz) → dit wordt tegenwoordig **'IF Direct Sampling'** genoemd. (B.v. de Elecraft K3 had dit al in 2006 met een 2e IF van 15 kHz.)
- IF filters: 3 kHz, 500 Hz, 300 Hz
- Het tweede ADC-pad zorgt voor het spectrum display (FFT)
- De FPGA bevat het algoritme dat het te bewerken / te decoderen signaal naar baseband brengt in IQ-format met tegelijk een (veel) lagere samplerate

FPGA: Field Programmable Gate Array



Yaesu FT-710 ontvanger (redelijk low-cost)

- Bijzonder: 2 x 14 bit ADC's in de Yaesu's (zowel FTDX-101 als FTDX-10 als FT-710 heeft deze)
- Wat gebeurt er:
  - De Divider voedt een TI ADS62P45, dit is een dual-channel, 14-bit ADC
  - De 2 ADC's samplen in I/Q
  - De FPGA levert signalen voor:
    - De DSP in I/Q format in baseband met lagere sample-rate
    - FFT bandscope
    - Pseudo Random Digital Dithersignaal voor de ADC's.



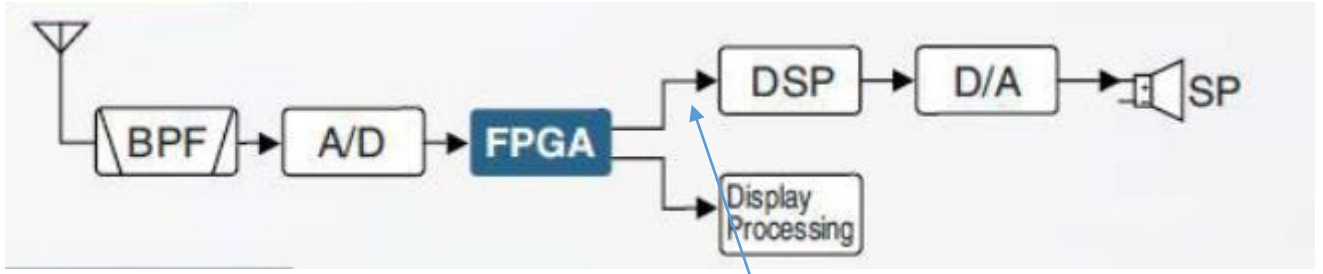
Yaesu FT-710 receiver block diagram

Zie: https://www.qsl.net/ab4oj/test/docs/ft710notes.pdf

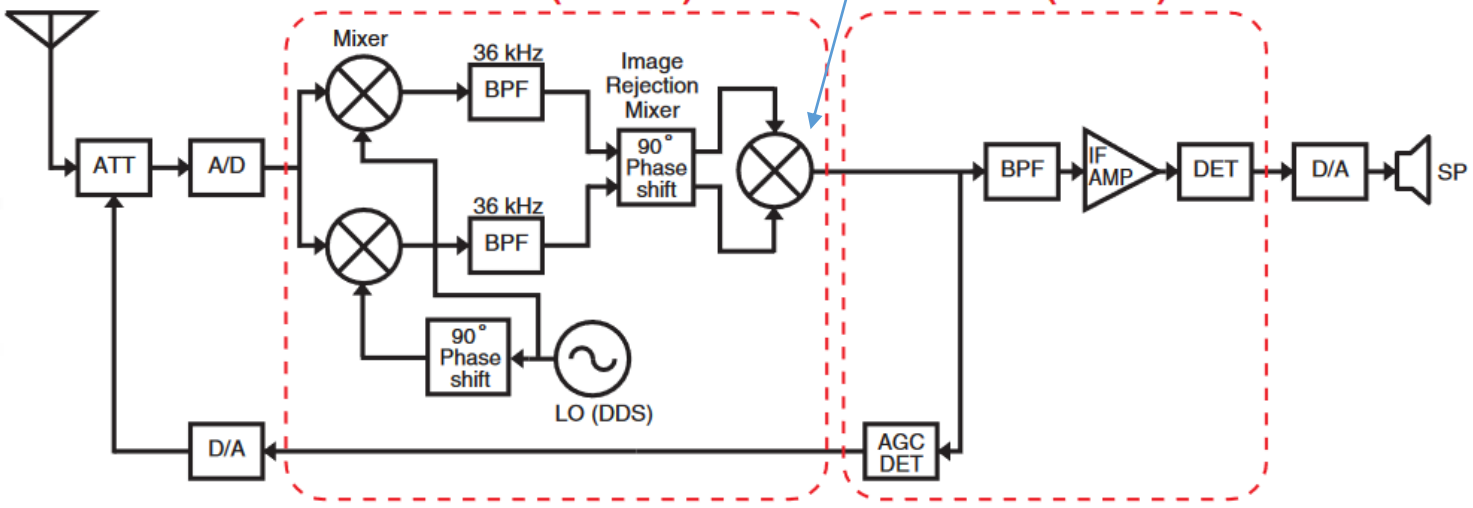
Icom IC-7300 (de game changer naar SDR)

- ADC: 14 bit,
  F_{sample}= 124 MS/s
- IP+ = Dithering aan

- FPGA → DSP:
  36 kHz 12kHz breed
  (zelfde als de ic-7100)



36 kHz – breedte 12 kHz



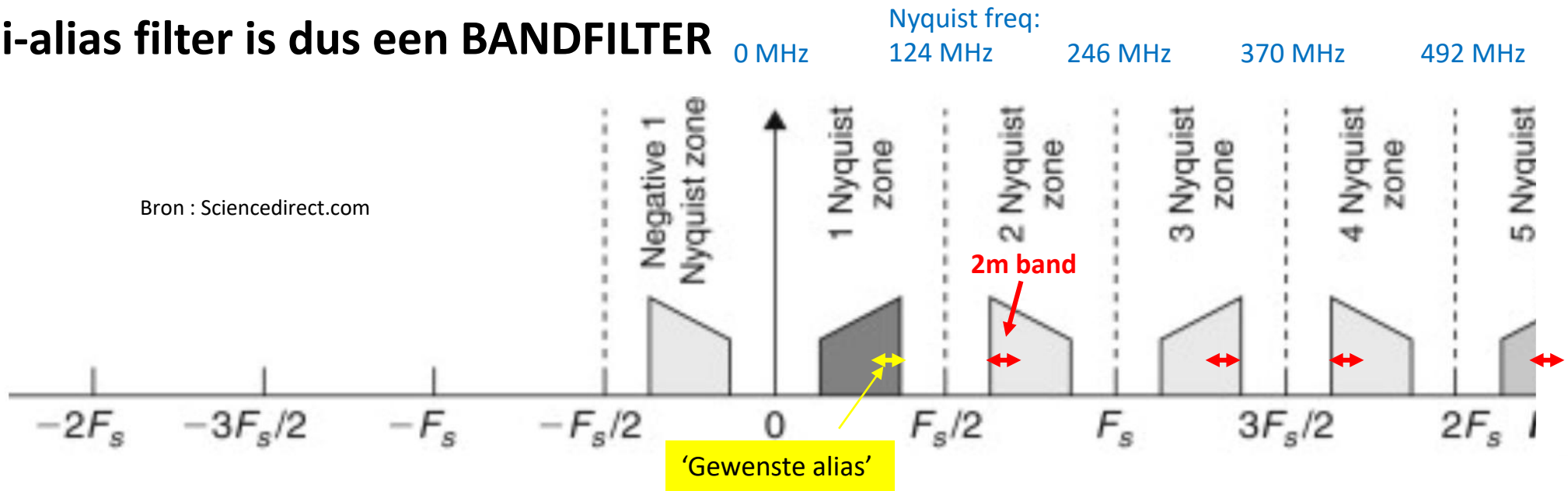
Flex 6700 met ‘harmonic’ sampling

- Flexradio 6700 met F_s = 246 Ms/s kan ook de 2m-band samplen. Maar 145MHz > Nyquist frequentie.
- **Nyquist zones**
  De 2m-band ligt binnen de zgn. 2e Nyquist zone, d.w.z. tussen ½ F_s en F_s
  Dit zien we wiskundig meteen ook in de base band (Nyquist zone 1) terug waar we normaliter ons signaal zien. Hiermee wordt dan weer verder gewerkt.

  Belangrijk: de gehele 2m band mag niet over een grens van de zone’s heen liggen, dan treedt meteen foute aliasing (overlapping) op.

Het analoge anti-alias filter is dus een BANDFILTER

Bron : Sciencedirect.com



Take away

- Na de samping wordt het te ontvangen deel van het spectrum geselecteerd en gedecimeerd (lagere sample rate) met het blokje FPGA zodat het verder verwerkt kan worden in het blokje DSP
- Bewerkingen op het signaal vinden plaats in IQ-vorm
  Met IQ wordt b.v. ook het voorbewerkte signaal van een device doorgegeven via USB aan de computer met een DSP (‘SDR’) programma
  Met IQ kun je onderscheid maken tussen positieve en negatieve frequentie
- In het blokje DSP vindt de filtering en detectie plaats
- De panadapter berekening volgt vaak een aparte weg (grotere bandbreedte en simpele berekening m.b.v. FFT)
- Alias is niet altijd verkeerd als je gebruik maakt van de zgn. Nyquist zones.
  De sample rate van de ADC bepaalt *niet de hoogste frequentie* die gesampled kan worden *maar de grootste bandbreedte*

Vragen ?

Opmerkingen?

Bronnen:

- ARRL Handbook (2015)
- DSP Basics, Dr Chris Bore
- Sciencedirect.com
- Atx7006.com
- Wikipedia.org
- Ni.com
- Addictedtoaudio.com.au
- Binnenlandsbestuur.nl
- sharetechnote.com
- lntwww.de
- digitalsoundandmusic.com
- Windowing & the FFT - A Guide for the Perplexed, Mark Newman
- http://t-filter.engineerjs.com/
- https://brianmcfee.net/dstbook-site/content/ch02-sampling/Sampling.html
- Understanding the 'Phasing Method' of SingleSideband Demodulation, Richard Lyons
- Quadrature Signals: Complex, But Not Complicated, Richard Lyons
- Design Of Digital Down Converter And Signal Detection Techniques For Software Defined Radio, Venugopal and Nemo
- A Beginner's Guide To Cascaded Integrator-Comb (CIC) Filters, Richard Lyons
- Analog.com
- IC-7610 User Evaluation &Test Report, Adam Farson VA7OJ/AB4OJ
- Yaesu FT-710 User Evaluation & Test Report, Adam Farson VA7OJ/AB4OJ
- Researchgate.net
- Yaesu.com
- Allaboutcircuits.com
- UNIVERSITY OF NEWCASTLE UPON TYNE – DSP
- Service manual IC-7300
- https://www.dutchaudioclassics.nl/philips-oversampling-system-for-compact-disc-decoding/