QMAC HF-90 ombouw

RF Seminar 22 juni 2025

Wouter Jan Ubbels PE4WJ

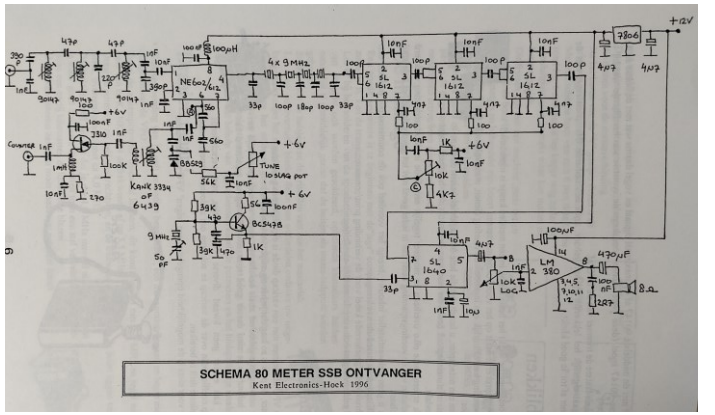
PE4WJ

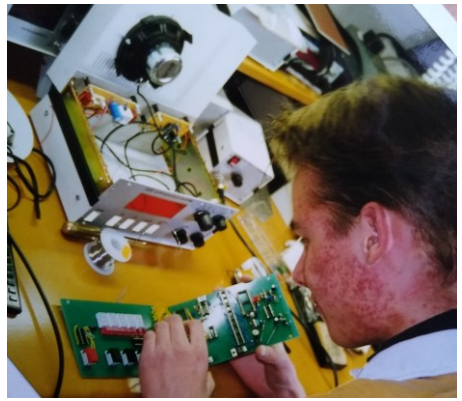
- PE4WJ sinds 2001, daarvoor NL12801. Volgend jaar lid van de OTC ☺
- Delfi-C3 satelliet
- Amateur van het jaar 2007
- Mede-oprichter en tegenwoordig Lead RF Engineer ISISPACE Group www.isispace.nl
- HF zelfbouw sinds 1996, ARDF, SOTA, CW contesten (als gast bij PI4CC en thuis), satellieten, EMC, analoge elektronica, SDR

1997 – NL12801

- Geinfecteerd met het HF-virus
- Kent Electronics 80m SSB RX
- Daar **MOEST** een PLL synthesizer in komen
- 1.6-7.384 MHz in 1 kHz stappen, USB & LSB
- MC145151 PLL IC uit TV-omzetter
- Veel van geleerd
  - PLL loop filters & stabiliteit (locktijd: ~2 sec!)
  - Rotary encoder uitlezen zonder microcontroller
  - Dynamisch bereik & intermodulatie
  - Systeemontwerp

Stichting
DE JONGE ONDERZOEKERS AMERSFOORT
Als techniek **jouw** hobby is!





1997 - QMAC HF-90

VK6MH Australia

Login is required for additional detail.

Ham Member **Lookups: 1187**

1997 - QMAC HF-90

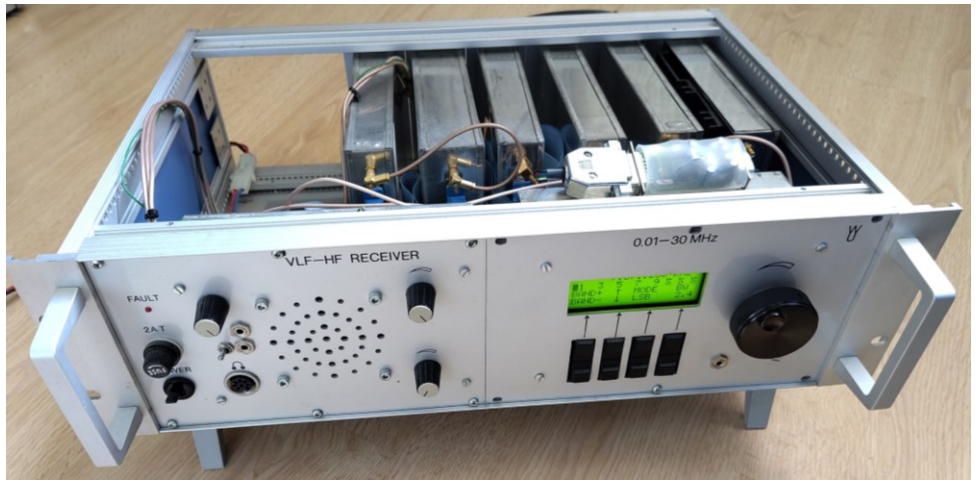
- Mijn droomradio!
- Naar (mijn) schatting zo'n 10000 geproduceerd tussen 1995 en 2012
- Civiele versie voor landmobiel gebruik
- Ook een militaire versie (HF-90M) – frequency hopping
- "Eurokaart" formaat, modulair, 3 PCB's
- "Channelized"
- Specificaties
  - 2-30 MHz
  - USB & LSB (& FSK)
  - Max 50 W PEP (op 10m nog maar 5 W…)
  - SELCALL (CCIR-493-4)
  - 12-28 V DC

Onbetaalbaar…

Bron: https://www.hfradio.com.au

2001-2005 Zelfbouw HF set

- Eerst een ontvanger, later transceiver van gemaakt
- PA0KSB DDS-PLL, later: SI570
- PIC18F452, geprogrammeerd in JAL
- 45 MHz 1e MF, 455 kHz 2e MF
- High level mixers
- SSB, CW, AM (RX)
- 100 mW exciter output, aparte PA0SSB IRF830 PA
- Modulair opgebouwd



2016

YouTube NL

Search

Q-MAC HF-90 shortwave HF radio design

Roderick Macduff
269 subscribers

Subscribe

82 Share Save

Jaren ’90 Technologie


MOTOROLA
SEMICONDUCTOR TECHNICAL DATA

Order this document by MC145159–1/D

MC145159-1

Serial-Input PLL Frequency Synthesizer with Analog Phase Detector

Interfaces with Dual–Modulus Prescalers

The MC145159–1 has a programmable 14–bit reference counter, as well as fully programmable divide–by–N/divide–by–A counters. The counters are programmed serially through a common data input and latched into the appropriate counter latch, according to the last data bit (control bit) entered. When combined with a loop filter and VCO, this device can provide all the remaining functions for a PLL frequency synthesizer operating up to the device's frequency limit. For higher VCO frequency operations, a down mixer or a dual–modulus prescaler can be used between the VCO and the PLL.

- Operating Temperature Range = –40 to 85°C
- Low Power Consumption Through Use of CMOS Technology
- 3.0 to 9.0 V Supply Range
- On– or Off–Chip Reference Oscillator Operation
- Compatible with the Serial Peripheral Interface (SPI) on CMOS MCUs
- ÷R Range = 16 to 16383
- ÷N Range = 16 to 1023, ÷A Range = 0 to 127
- High–Gain Analog Phase Detector
- See Application Note AN969

P SUFFIX
PLASTIC DIP
CASE 738

DW SUFFIX
SOG PACKAGE
CASE 751D

VF SUFFIX
SSOP
CASE TBD

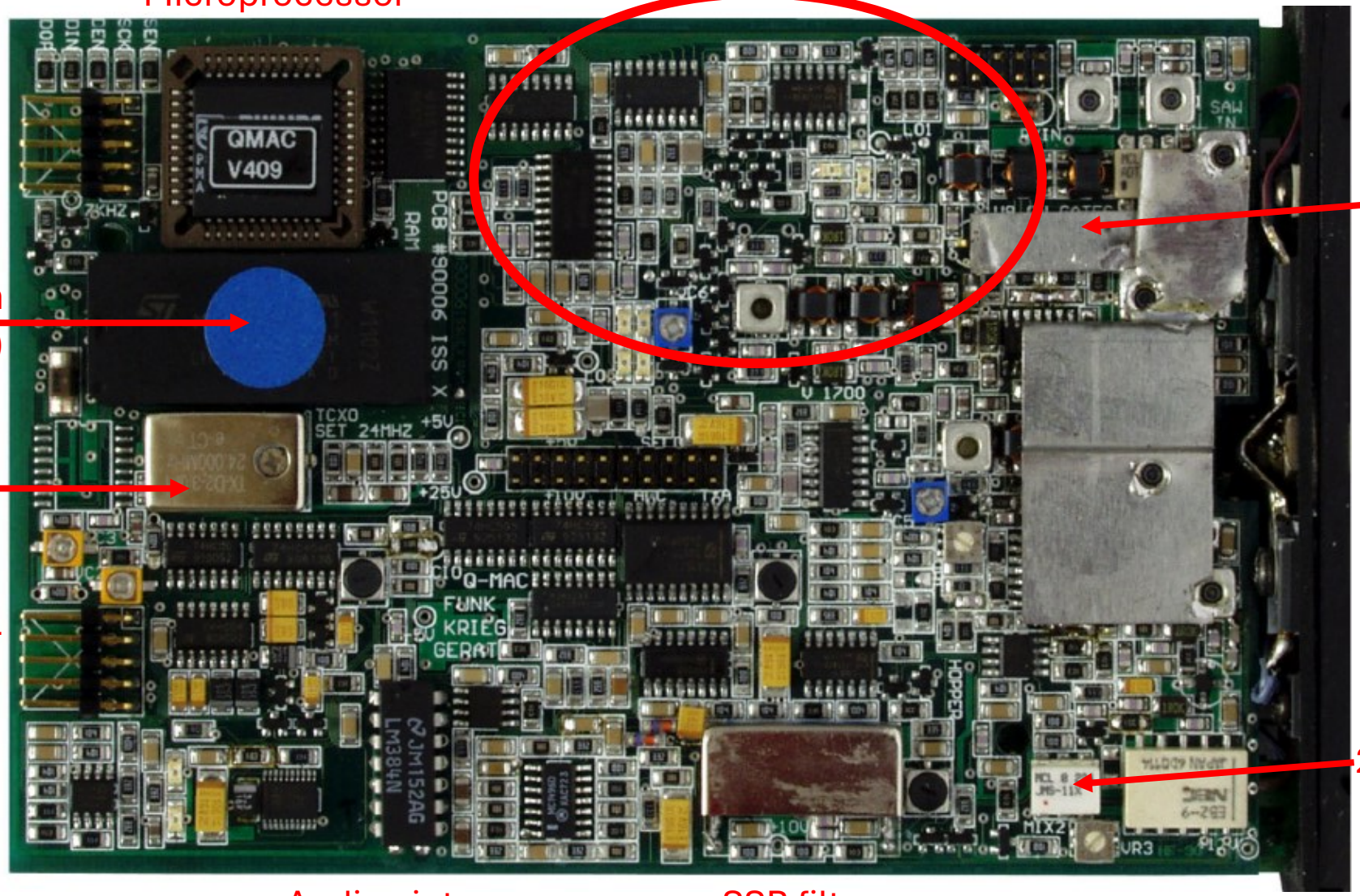
ORDERING INFORMATION

MC145159P1	Plastic DIP
MC145159DW1	SOG Package
MC145159VF1	SSOP

Bron: Cryptomuseum

- Motorola PLL Synthesizer familie
- IRF830 in eindtrap (later ARF463)
- Current feedback opamps (!)
- High level GaAsFET mixer
- Upconversie frontend – 83.16 MHz 1^{e} MF, 30 kHz SAW filter (!)
- 8051-afgeleide microprocessor
- Bourns rotary encoder – voor volume…

RXMP print



Microprocessor

1e en 2e LO synthesizer

Pin header naar front

1e mixer + SAW filter

Kanaal geheugen (batterij backup!)

24 MHz TCXO

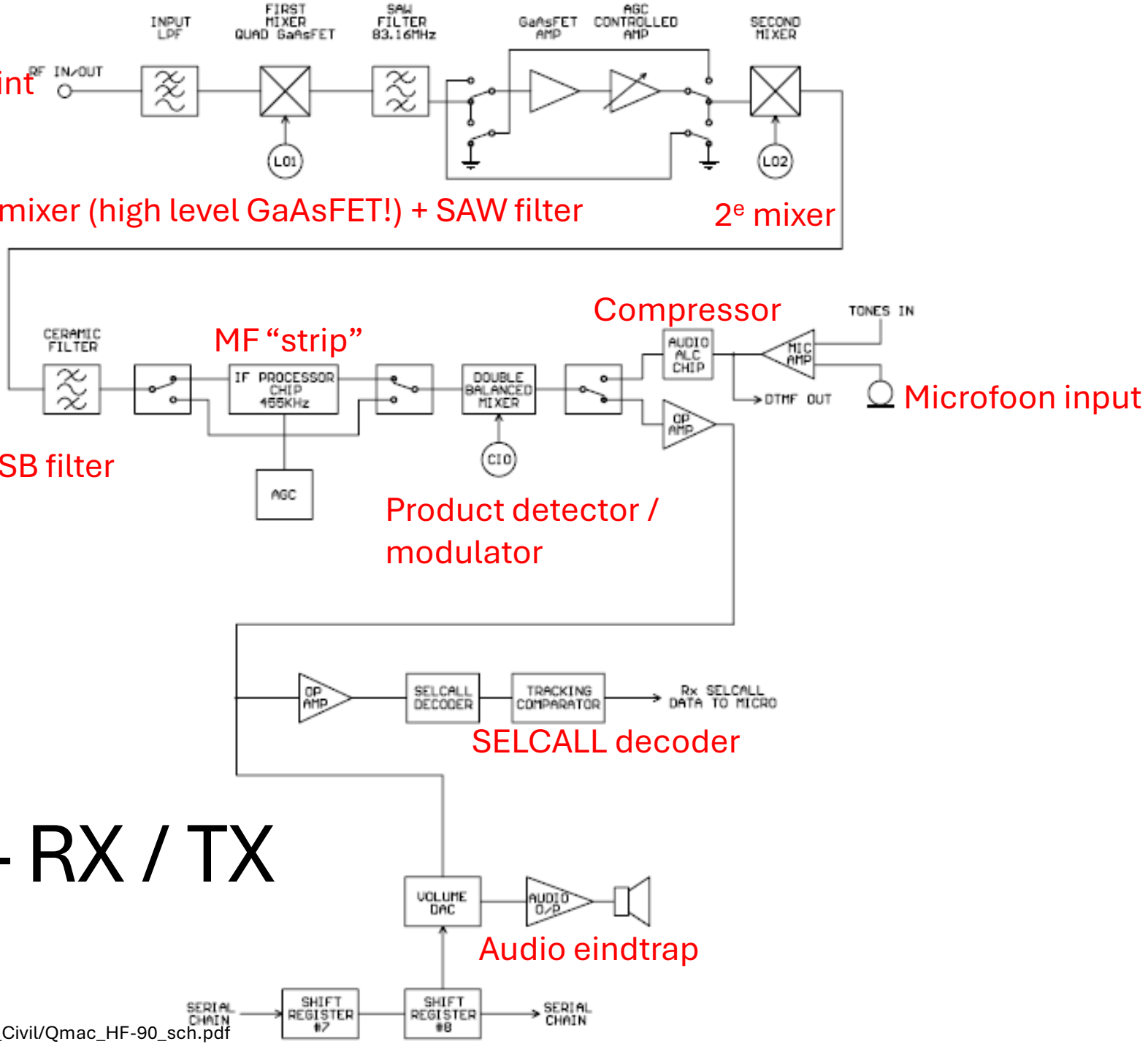
Pin header naar front

2e mixer

Audio eintrap

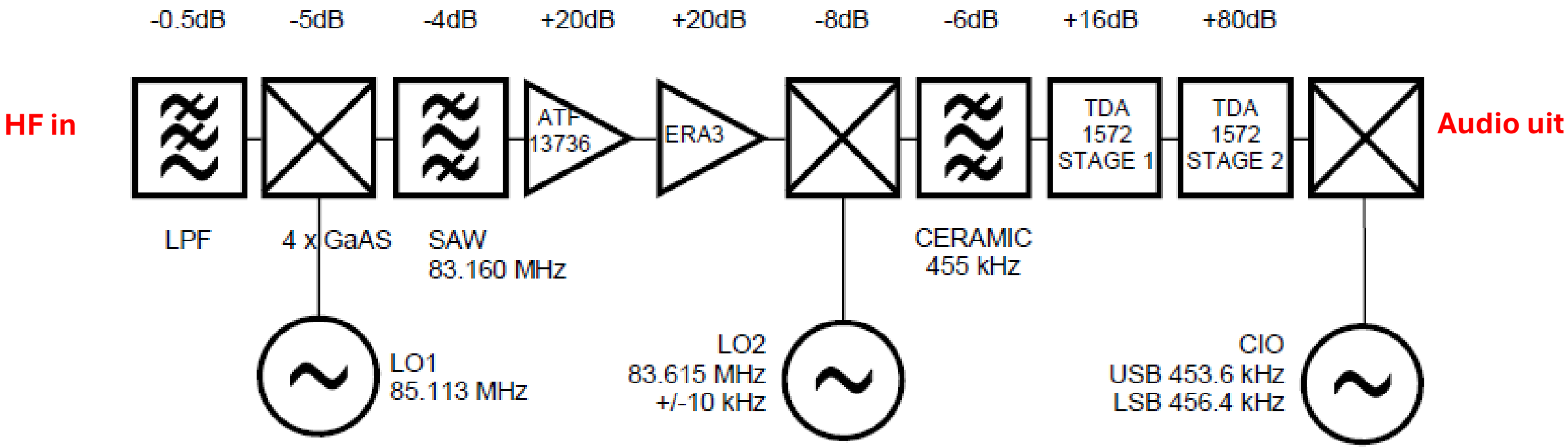
SSB filter

RXMP print – RX / TX



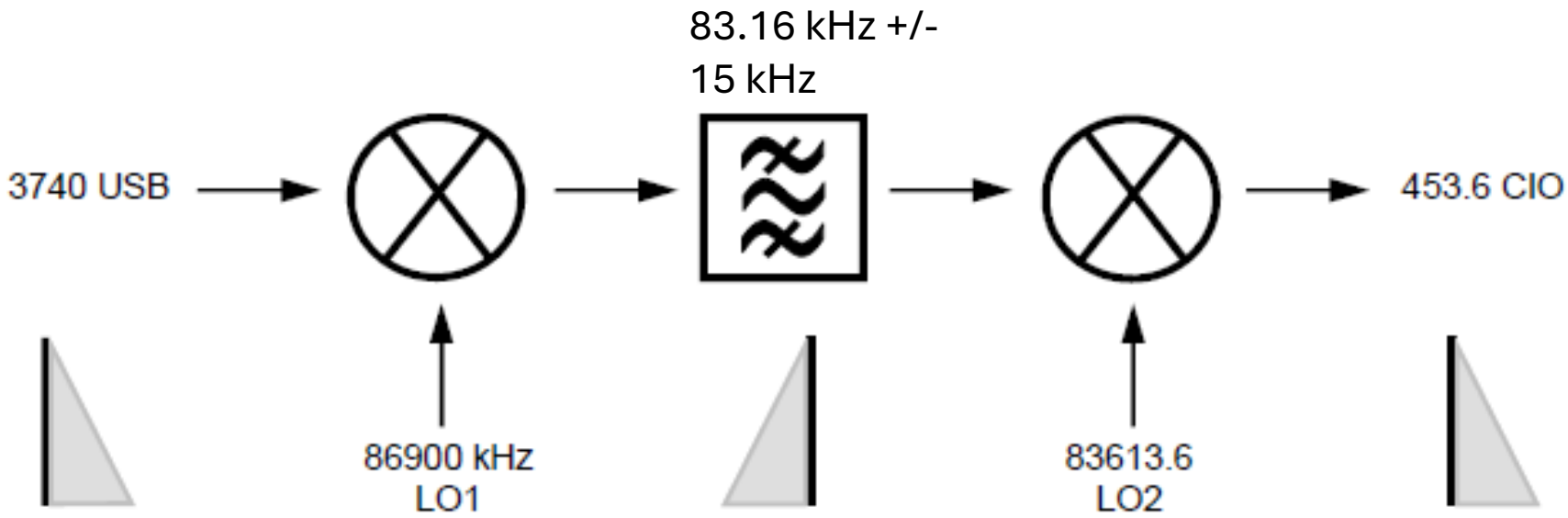
Bron: http://www.radiomanual.info/schemi/Surplus_Civil/Qmac_HF-90_sch.pdf

RX Gain distributie



Bron: QMAC HF-90 Technical Manual – https://fccid.io/RK2HF90

Traditionele superhet, met een hoge 1e MF

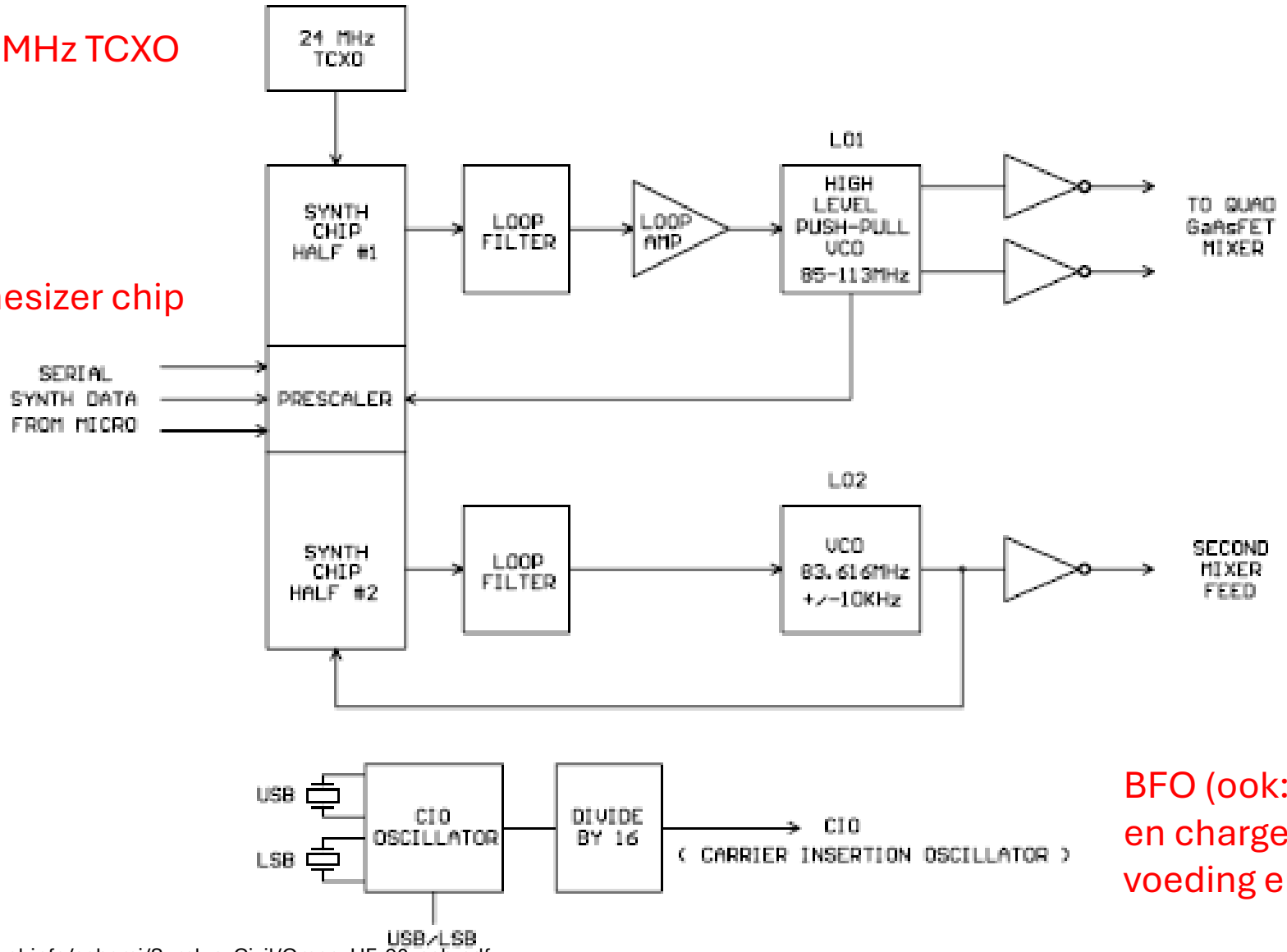


Bron: QMAC HF-90 Technical Manual – https://fccid.io/RK2HF90

RXMP print – Frequentiefabriek

24 MHz TCXO

Dual synthesizer chip



85-113 MHz 1e LO

83.616 MHz +/- 10 kHz 2e LO

BFO (ook: display multiplex klok en charge pump voor negatieve voeding en +25V voor synthesizer)

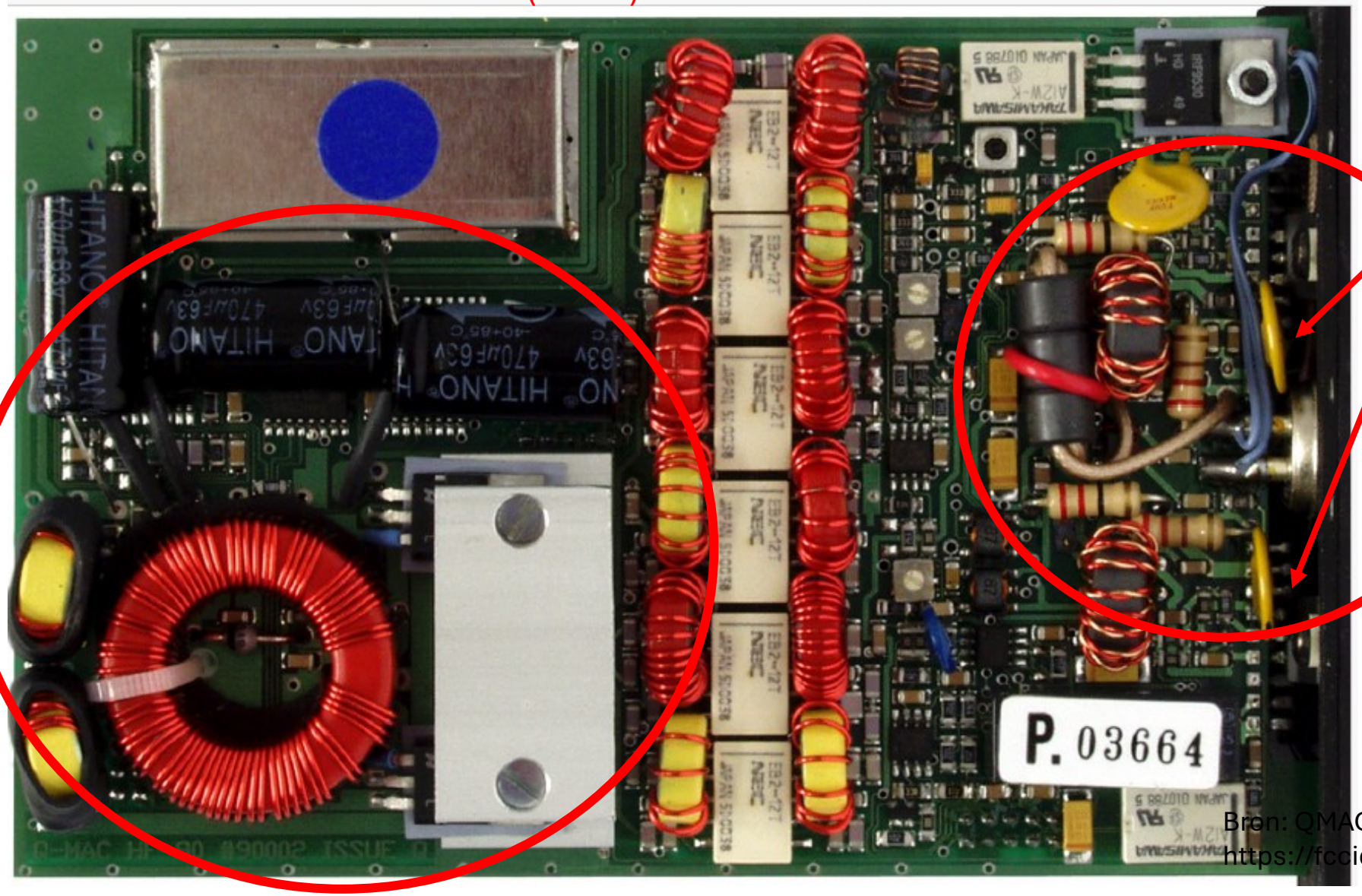
Bron: http://www.radiomanual.info/schemi/Surplus_Civil/Qmac_HF-90_sch.pdf

PASW print

5V DC/DC converter (in blik)

LPF

Driver & eindtrap

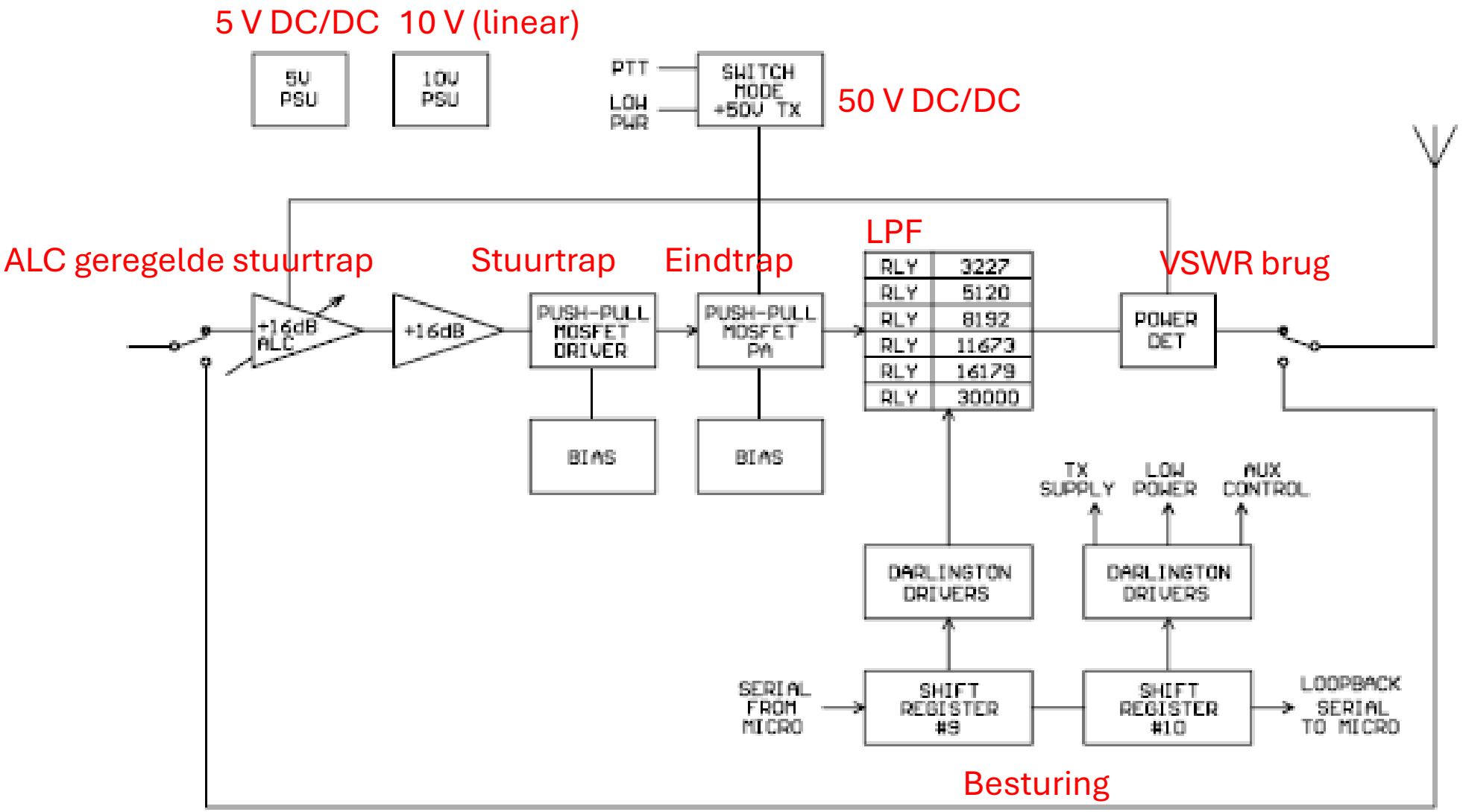


PTC's in de source van de eindFETs

50V DC/DC converter voor eindtrap

Bron: QMAC HF-90 Schematics – https://fccid.io/RK2HF90

PASW print



Bron: http://www.radiomanual.info/schemi/Surplus_Civil/Qmac_HF-90_sch.pdf

Display print

6x 7 segment display

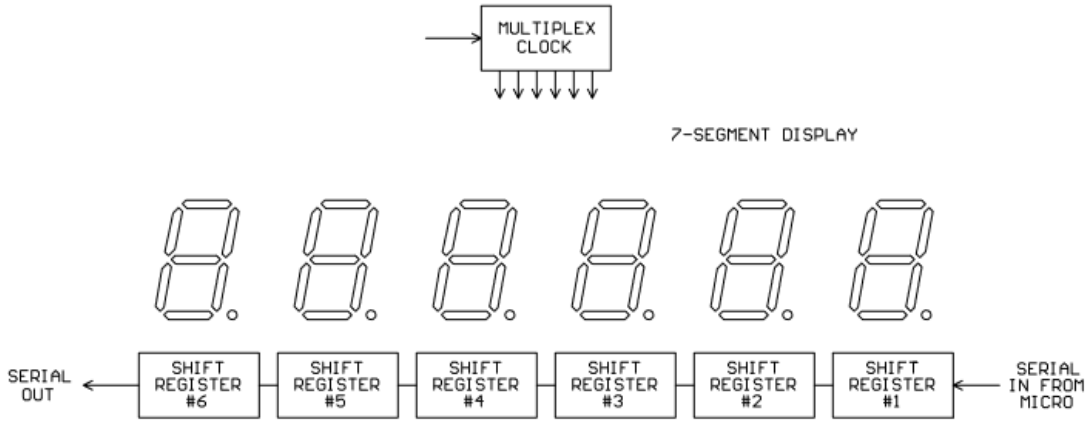
Volume encoder

Keypad

Header naar PASW print

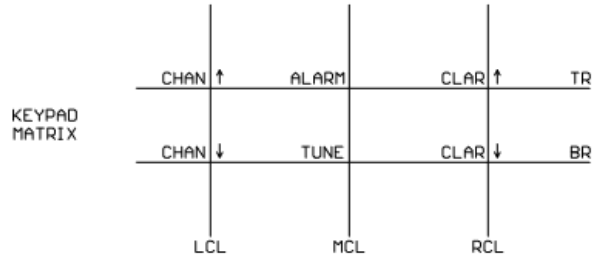
Header naar PASW print

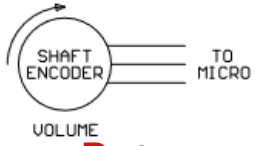
Display print



Display + schuifregisters

Matrix keypad





Rotary encoder

Bron: http://www.radiomanual.info/schemi/Surplus_Civil/Qmac_HF-90_sch.pdf

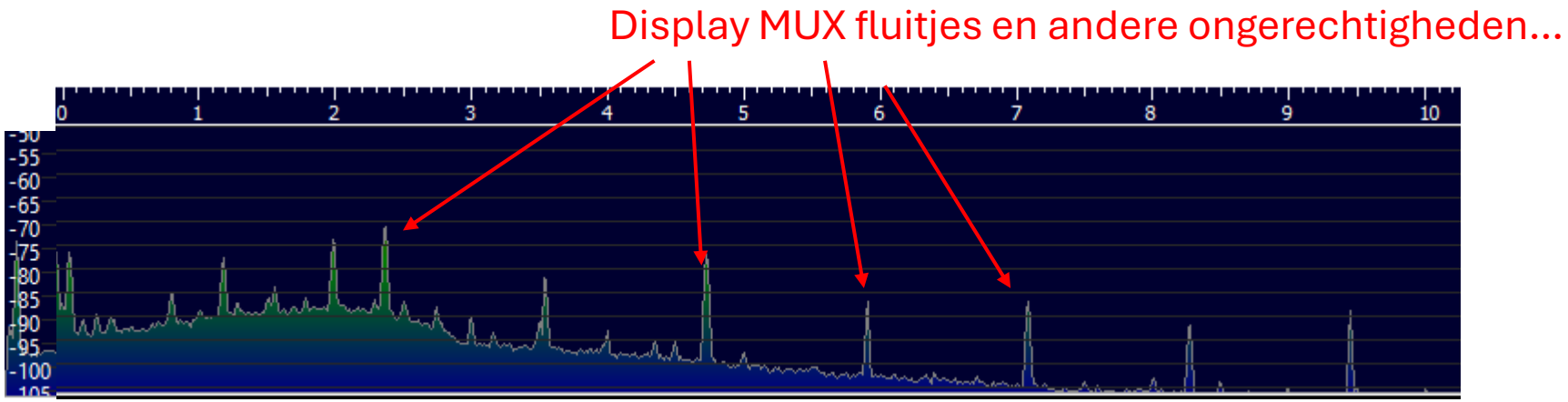
2018

- HF-90 te koop! Inclusief handmicrofoon
- Serie nummer in de 6000
- Afkomstig van de Australian Antarctic Division – Macquarie island

Bron: https://en.wikipedia.org/wiki/Macquarie_Island

De deceptie…

- Zeer hard volume na aanzetten (prima voor in een rijdende jeep, veel te hard voor de shack)
- Vreselijk audio, “geknepen” en behoorlijk vervormd (clipping)
- “Channelized operation” volkomen onpraktisch voor amateurgebruik
- Paar verbindingen mee gemaakt en daarna belandde de set op de plank…

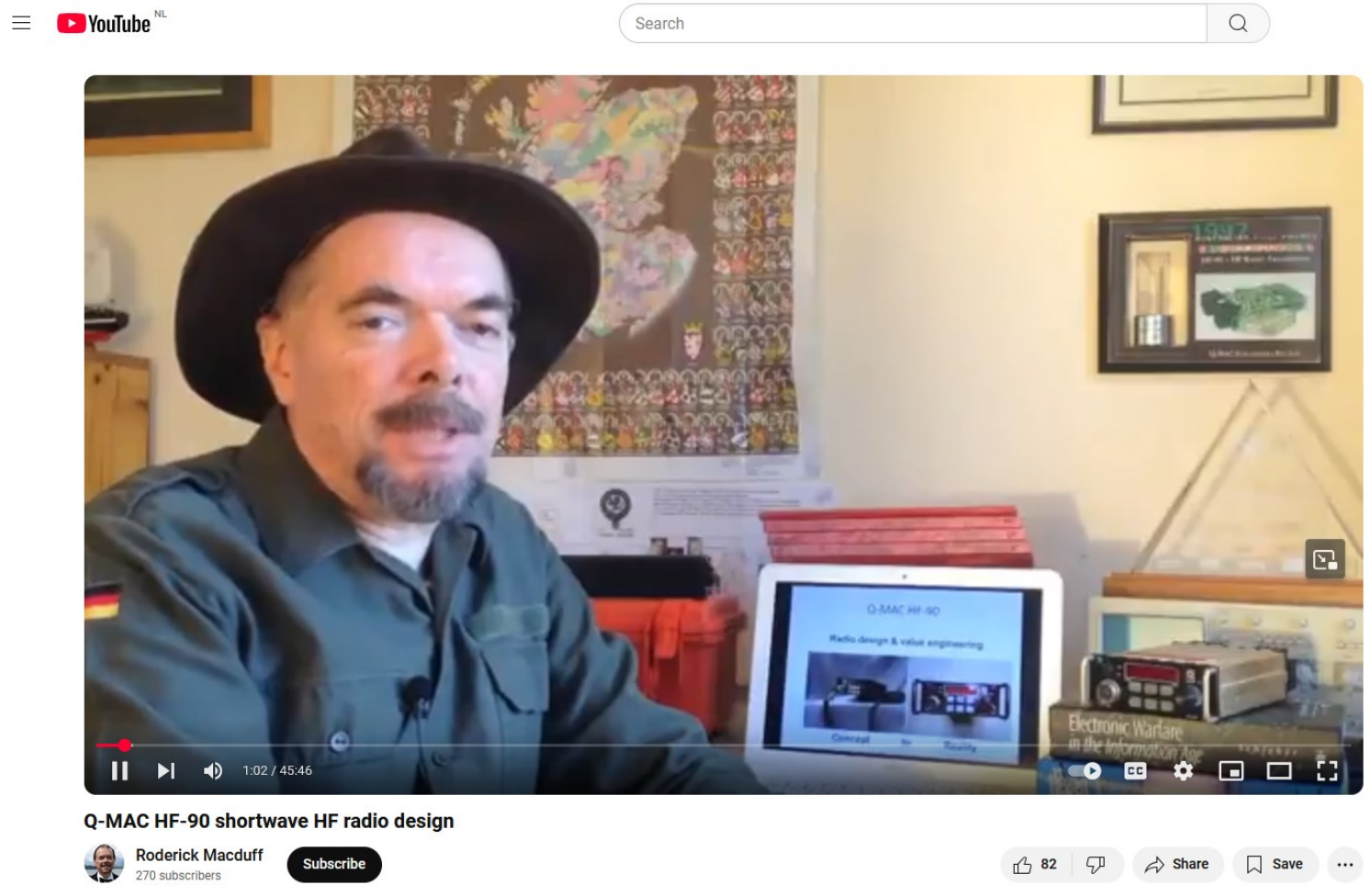


Originele set, audio spectrum zonder antenne aangesloten

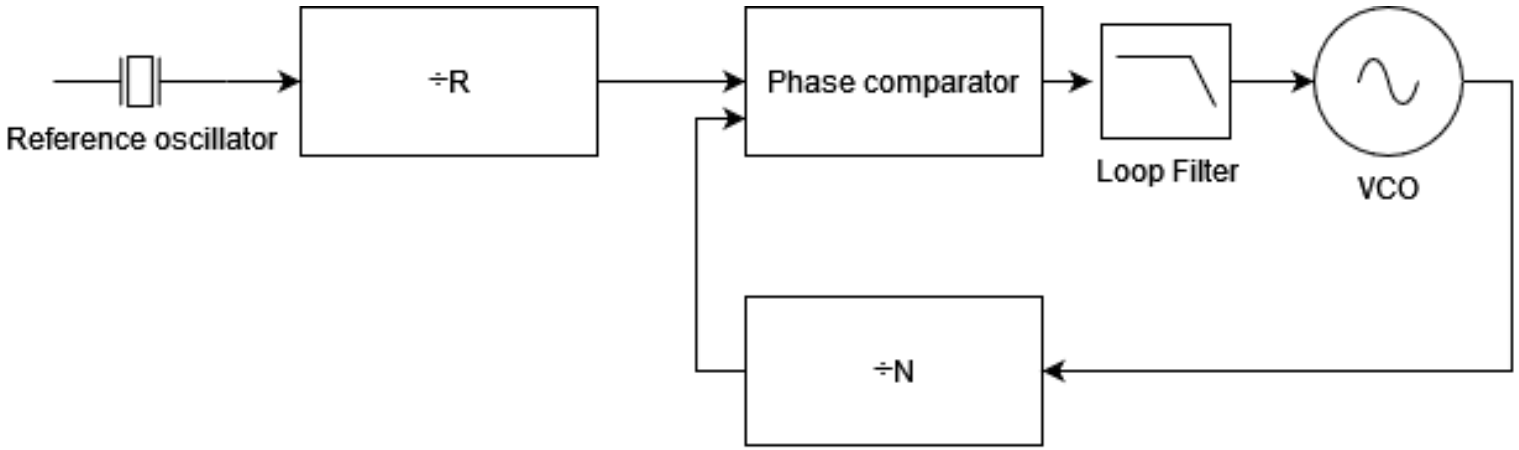
Ondertussen…

Nog maar eens de videos van Rod VK6MH bekeken, twee aspecten liet me niet los…

- **Hoe konden ze nou die 25 Hz frequentie stapjes maken van 2-30 MHz, en tegelijkertijd heel snel locken?**
- **Kan ik de set VFO afstembaar maken?**



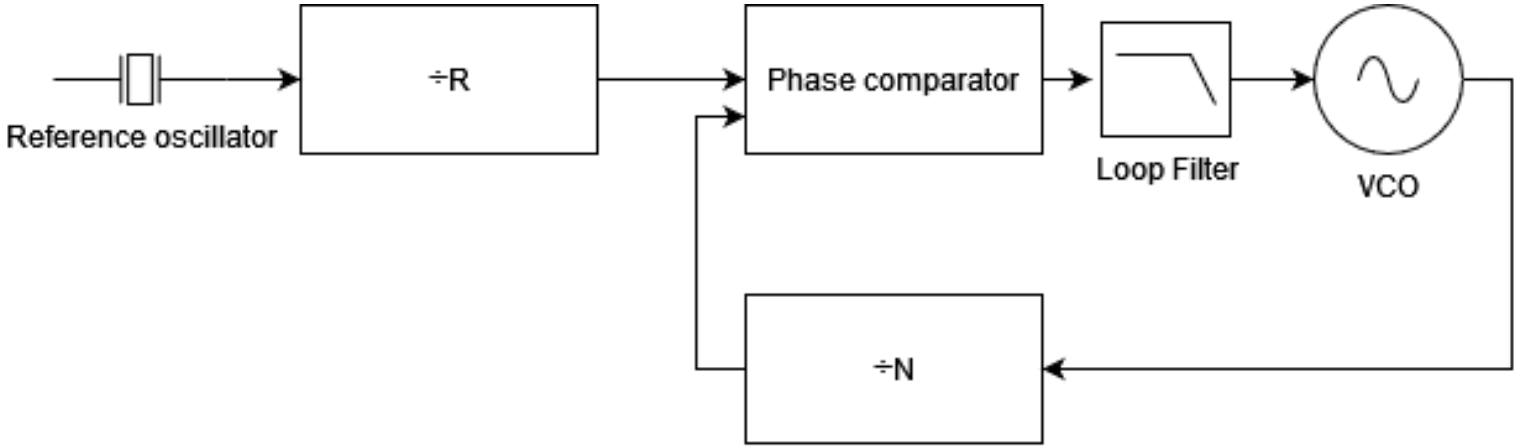
Conventionele PLL



$$f_{vco} = \frac{N}{R} f_{ref}$$

- R heeft vaste waarde, hiermee wordt het frequentie raster bepaald (bijv. 25 kHz)
- Voor een 100 Hz raster betekent dat een 100 Hz fase-vergelijkings frequentie → zeer lage lusbandbreedte → lange locktijd en slechte faseruis (net als bij mijn eerste PLL in 1997!)

“Magic number” PLL

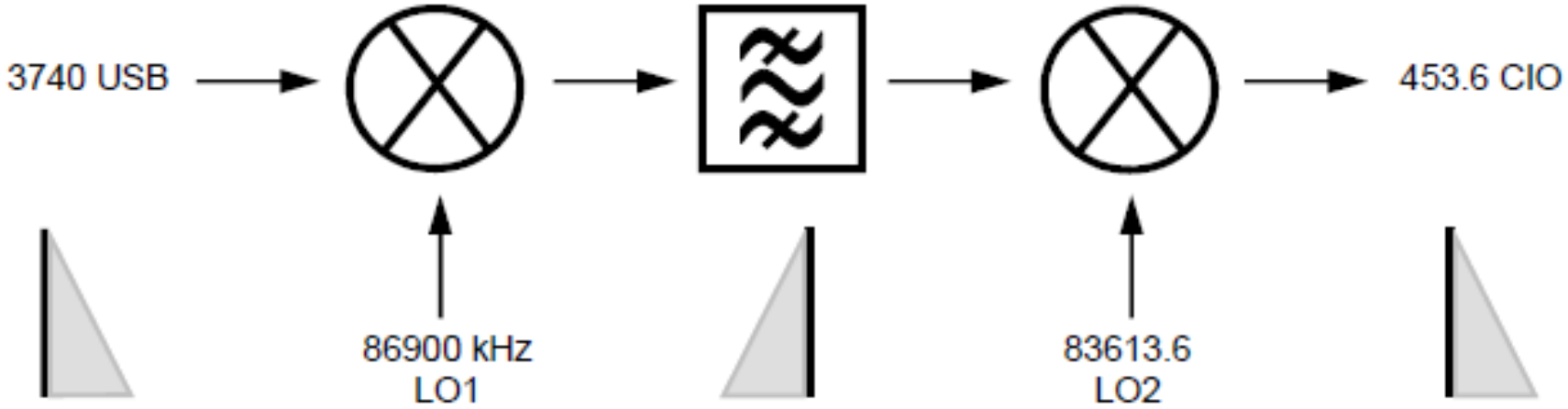


$$\left| \frac{N}{R} f_{ref} - f_{vco,wanted} \right| < \varepsilon$$

Volgens de presentatie van Rod:

- Varieer zowel N als R, en zorg dat de fase-vergelijkingsfrequentie zo groot mogelijk blijft, en in elk geval niet kleiner wordt dan een bepaalde waarde (voor de HF-90 is dat 40 kHz) → **snelle locktijd en goede faseruis! → belangrijk voor frequency hopping**
- Dit leidt tot een (varierende) frequentie fout
- Voor elke gewenste VCO frequentie bestaat er minimaal 1 combinatie van R en N die leiden tot een acceptabele frequentiefout (bijvoorbeeld 25 Hz)
- Dat laatste bleek later niet helemaal waar te zijn…

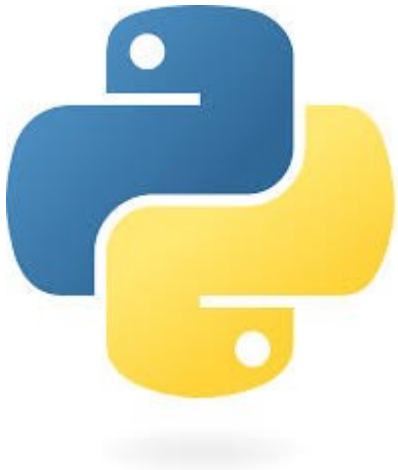
HF-90 meng schema



- LO1 wordt afgestemd in 20 kHz stappen
- LO2 in 25 Hz stappen, over een 20 kHz bereik. Het 1e MF filter is 30 kHz breed.
- Dit leidt tot 25 Hz stapjes over 2 – 30 MHz
- Op deze manier blijft het aantal deeltallen wat moet worden opgeslagen in het geheugen van de microprocessor acceptabel
- Beide synthesizers werken volgens het “dual modulus” principe

```python
import numpy as np
from tqdm import tqdm
import pandas as pd


# Input parameters
F_REF = 24_000_000  # master reference frequency
LO1_STEP = 20000  # Hz frequency step of LO1
LO2_STEP = 25  # Hz frequency step of LO2

ALLOWED_ERROR_LO1 = (
    3000
)
ALLOWED_ERROR_LO1_GAPS = 3000 # Hz max allowed frequency error of LO2 in frequencies which ca

ALLOWED_ERROR_LO2 = 15#12.5  # Hz max allowed frequency error of LO2
ALLOWED_ERROR_LO2_GAPS = 50#12.5  # Hz max allowed frequency error of LO2 in frequencies whic

F_LO2_CENTER = 83616000  # Hz Center frequency of LO2
SIDEBANDMARGIN = (
    2400
)  # Hz margin to allow sideband (BFO) correction to be carried out using LO2

# min and max phase comparison frequencies per loop
FR_MIN_LO1 = 40_000
FR_MIN_LO1_GAPS = 3_000

FR_MAX_LO1 = 202_000

FR_MIN_LO2 = 1_000
FR_MAX_LO2 = 14_000

F_IF = 83160000  # Hz 1st IF frequency
F_RF_MIN = 2000000  # Hz min receive frequency
F_RF_MAX = 30020000  # Hz max receive frequency

R_MAX = 8191

def fast_find_magic_numbers(f, f_ref, fr_min, fr_max, allowed_error, maximize_fr = False):
    N_min = f / fr_max
    N_min_round = int(np.round(N_min) - 1)

    N_max = f / fr_min
```

Python



Tijd om het magic number schema in Python te implementeren, en te kijken of het klopt...

https://www.python.org/

Script stappen

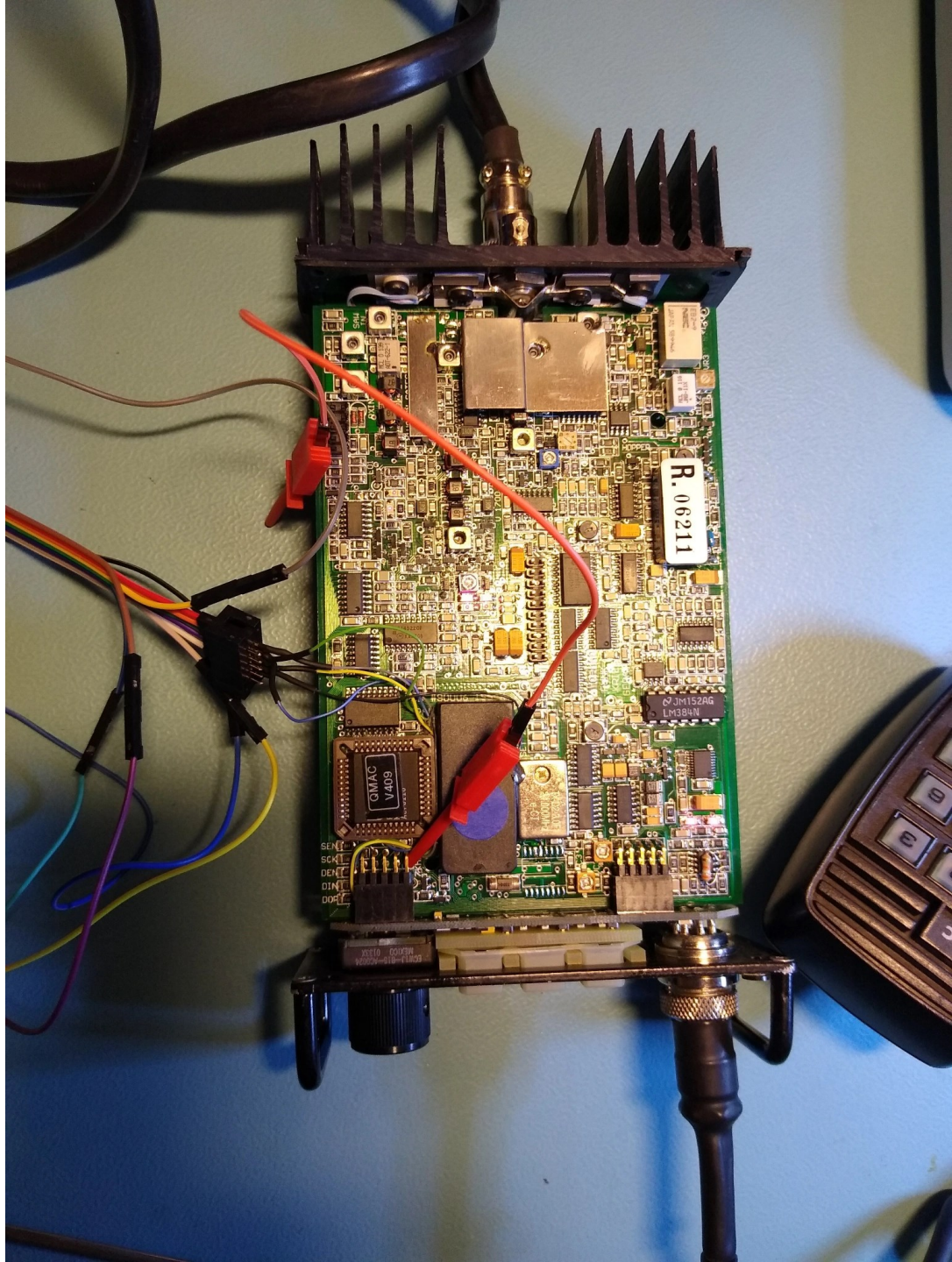
1. LO1: bepaal PLL deeltallen die nodig zijn, de volgende beperkingen in acht nemende:
    - Fasevergelijkingsfrequentie zo hoog mogelijk, doch altijd tussen 40 kHz en 202 kHz
    - Maximale frequentiefout 3 kHz
2. LO2: bepaal PLL deeltallen die nodig zijn, de volgende beperkingen in acht nemende:
    - Fasevergelijkingsfrequentie zo hoog mogelijk, doch altijd tussen 1 kHz en 14 kHz
    - Maximale frequentiefout 15 Hz
3. Bepaal hoeveel 25 Hz stapjes nodig zijn om LO1 frequentiefout te compenseren met LO2
4. Sla de resultaten op in een aantal .csv bestanden (tabellen)

Script resultaten – wat blijkt:

- *Bijna* alle 20 kHz raster frequenties zijn te maken met LO1 met een minimale fasevergelijkingsfrequentie van 40 kHz zolang we accepteren dat de fout tot **3 kHz** kan bedragen.
- *Bijna* alle 25 Hz raster frequenties zijn te maken met LO2 (over +/- 20 kHz) met een minimale fasevergelijkingsfrequentie van 1 kHz zolang we accepteren dat de fout tot **15 Hz** kan bedragen
- Er zijn bij beide LO's **4** frequenties waar dit niet lukt, hier **moeten** we een lagere fasevergelijkingsfrequentie accepteren. Het is niet anders. Dit is ongetwijfeld te verklaren vanuit de getallentheorie, ik ben er niet ingedoken ☺
- De (tot 3 kHz grote) frequentiefout die wordt geintroduceerd door LO1 wordt gecompenseerd door LO2 overeenkomstig "de andere kant" af te stemmen. Dit levert een "overall" frequentie fout op van <= 15 Hz!
- Bovenstaande is bevestigd dmv spectrum analyzer metingen met de originele microcontroller op de print
- Voor USB/LSB bedrijf wordt LO2 + of – 1.5 kHz verstemd, om zodoende weer zero-beat te zitten. Er is immers maar 1 MF filter, op 455 kHz + / 2.4 kHz
- Let wel: dit werkt bij gratie van de breedte van het 1^{e} MF filter!
- Dit filter is 30 kHz breed – dat betekent dus "aan weerszijden" 5 kHz marge. 3 kHz (compensatie frequentiefout) + 1.5 kHz (USB/LSB offset) = 4.5 kHz. Dus in totaal 20 + 2x4.5 = 29 kHz nodig → **dat past precies!**

Seriele bus

- Seriele bus van microprocessor naar:
  - Synthesizer (direct)
  - 10 In serie geschakelde74HC595 Schuifregisters
    - Display (6 registers)
    - AGC, USB/LSB
    - Audio volume (door een DAC)
    - LPF
    - High/low power, selcall aan/uit, externe 12V aan/uit
- 4 bus signalen:
  - Data
  - Clock
  - Schuifregister Latch Enable
  - Synthesizer Latch Enable

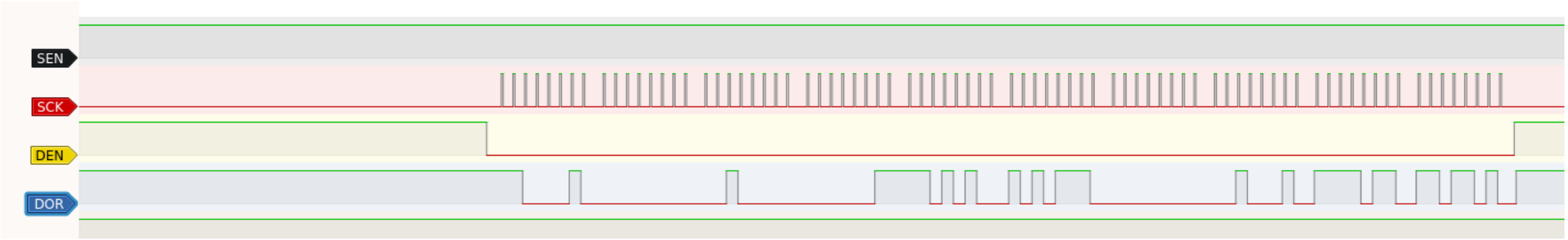


Bus sniffen met Sigrok en de “$10 Logic Analyzer”

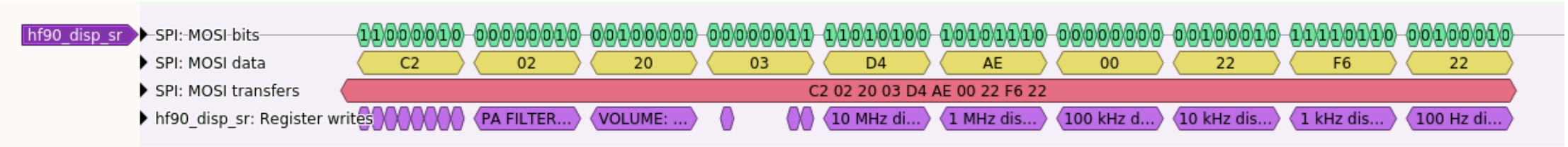
- https://sigrok.org/wiki/Main_Page
- Open source Logic Analyzer software, met professionele functionaliteit
- Custom protocol decoders, geschreven in Python
- Werkt met zeer goedkope Logic Analyzers (bijv. Tinytronics, ebay, Ali…)

Bron: www.tinytronics.nl

Schuifregister databus transactie



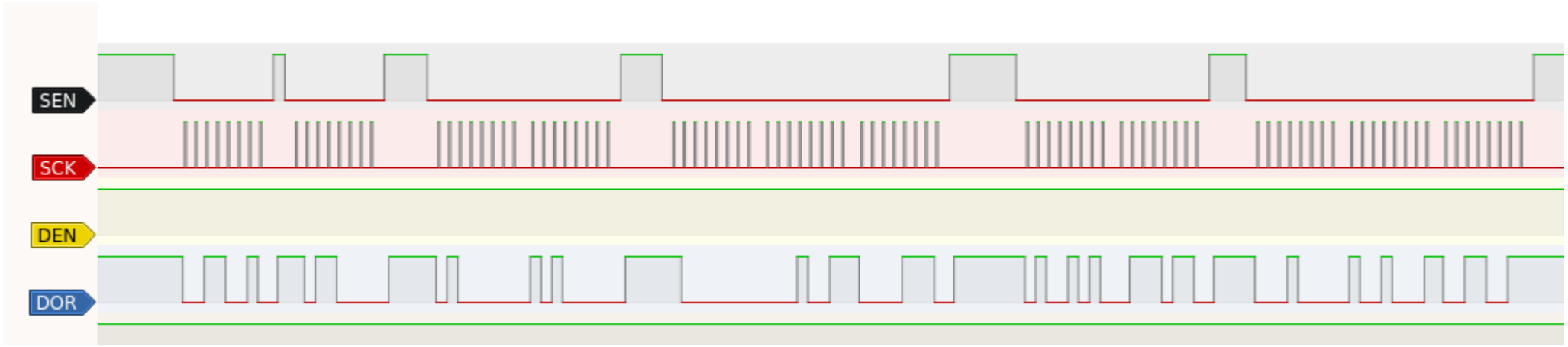
Gedecodeerde schuifregister data



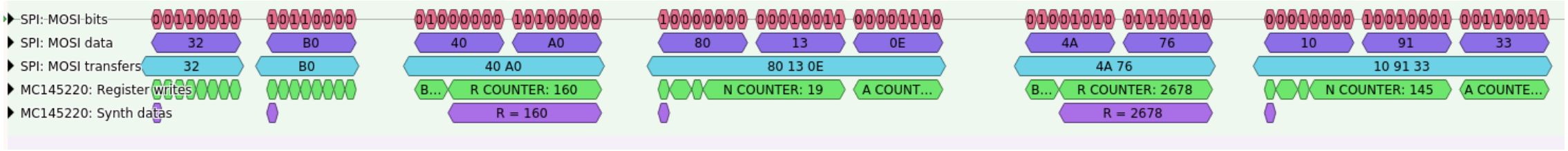
Zelfde in tabelvorm...

Sample	Time	Decoder	Ann Row	Ann Class	Value
31404239	3.926 s	hf90_disp_sr	Register writes	Register write	DISABLE LOOPBACK: 1
31404287	3.926 s	SPI	MOSI bits	MOSI bit	1
31404287	3.926 s	hf90_disp_sr	Register writes	Register write	DISABLE SELCALL: 1
31404335	3.926 s	SPI	MOSI bits	MOSI bit	0
31404335	3.926 s	hf90_disp_sr	Register writes	Register write	NIL: 0
31404383	3.926 s	SPI	MOSI bits	MOSI bit	0
31404383	3.926 s	hf90_disp_sr	Register writes	Register write	NIL: 0
31404431	3.926 s	SPI	MOSI bits	MOSI bit	0
31404431	3.926 s	hf90_disp_sr	Register writes	Register write	+50V OFF: 0
31404479	3.926 s	SPI	MOSI bits	MOSI bit	0
31404479	3.926 s	hf90_disp_sr	Register writes	Register write	Low Power: 0
31404527	3.926 s	SPI	MOSI bits	MOSI bit	1
31404527	3.926 s	hf90_disp_sr	Register writes	Register write	ATU ON: 1
31404575	3.926 s	SPI	MOSI bits	MOSI bit	0
31404575	3.926 s	hf90_disp_sr	Register writes	Register write	NIL: 0
31404659	3.926 s	SPI	MOSI data	MOSI data	00
31404659	3.926 s	SPI	MOSI bits	MOSI bit	0
31404659	3.926 s	hf90_disp_sr	Register writes	Register write	PA FILTER BAND SELECT: NULL
31404707	3.926 s	SPI	MOSI bits	MOSI bit	0
31404755	3.926 s	SPI	MOSI bits	MOSI bit	0
31404803	3.926 s	SPI	MOSI bits	MOSI bit	0
31404851	3.926 s	SPI	MOSI bits	MOSI bit	0
31404899	3.926 s	SPI	MOSI bits	MOSI bit	0
31404947	3.926 s	SPI	MOSI bits	MOSI bit	0
31404995	3.926 s	SPI	MOSI bits	MOSI bit	0
31405079	3.926 s	SPI	MOSI data	MOSI data	20
31405079	3.926 s	SPI	MOSI bits	MOSI bit	0
31405079	3.926 s	hf90_disp_sr	Register writes	Register write	VOLUME: 32
31405127	3.926 s	SPI	MOSI bits	MOSI bit	0
31405175	3.926 s	SPI	MOSI bits	MOSI bit	1
31405223	3.926 s	SPI	MOSI bits	MOSI bit	0
31405271	3.926 s	SPI	MOSI bits	MOSI bit	0
31405319	3.926 s	SPI	MOSI bits	MOSI bit	0
31405367	3.926 s	SPI	MOSI bits	MOSI bit	0
31405415	3.926 s	SPI	MOSI bits	MOSI bit	0
31405499	3.926 s	SPI	MOSI data	MOSI data	03
31405499	3.926 s	SPI	MOSI bits	MOSI bit	0
31405547	3.926 s	SPI	MOSI bits	MOSI bit	0
31405547	3.926 s	hf90_disp_sr	Register writes	Register write	MIC INHIBIT: NO INHIBIT
31405595	3.926 s	SPI	MOSI bits	MOSI bit	0
31405643	3.926 s	SPI	MOSI bits	MOSI bit	0
31405691	3.926 s	SPI	MOSI bits	MOSI bit	0
31405739	3.926 s	SPI	MOSI bits	MOSI bit	0
31405787	3.926 s	SPI	MOSI bits	MOSI bit	1
31405787	3.926 s	hf90_disp_sr	Register writes	Register write	AGC: SLOW
31405835	3.926 s	SPI	MOSI bits	MOSI bit	1
31405835	3.926 s	hf90_disp_sr	Register writes	Register write	MODE: USB
31405919	3.926 s	SPI	MOSI data	MOSI data	D4
31405919	3.926 s	SPI	MOSI bits	MOSI bit	1

Synthesizer databus transactie



Gedecodeerde synthesizer register data:



In tabelvorm...

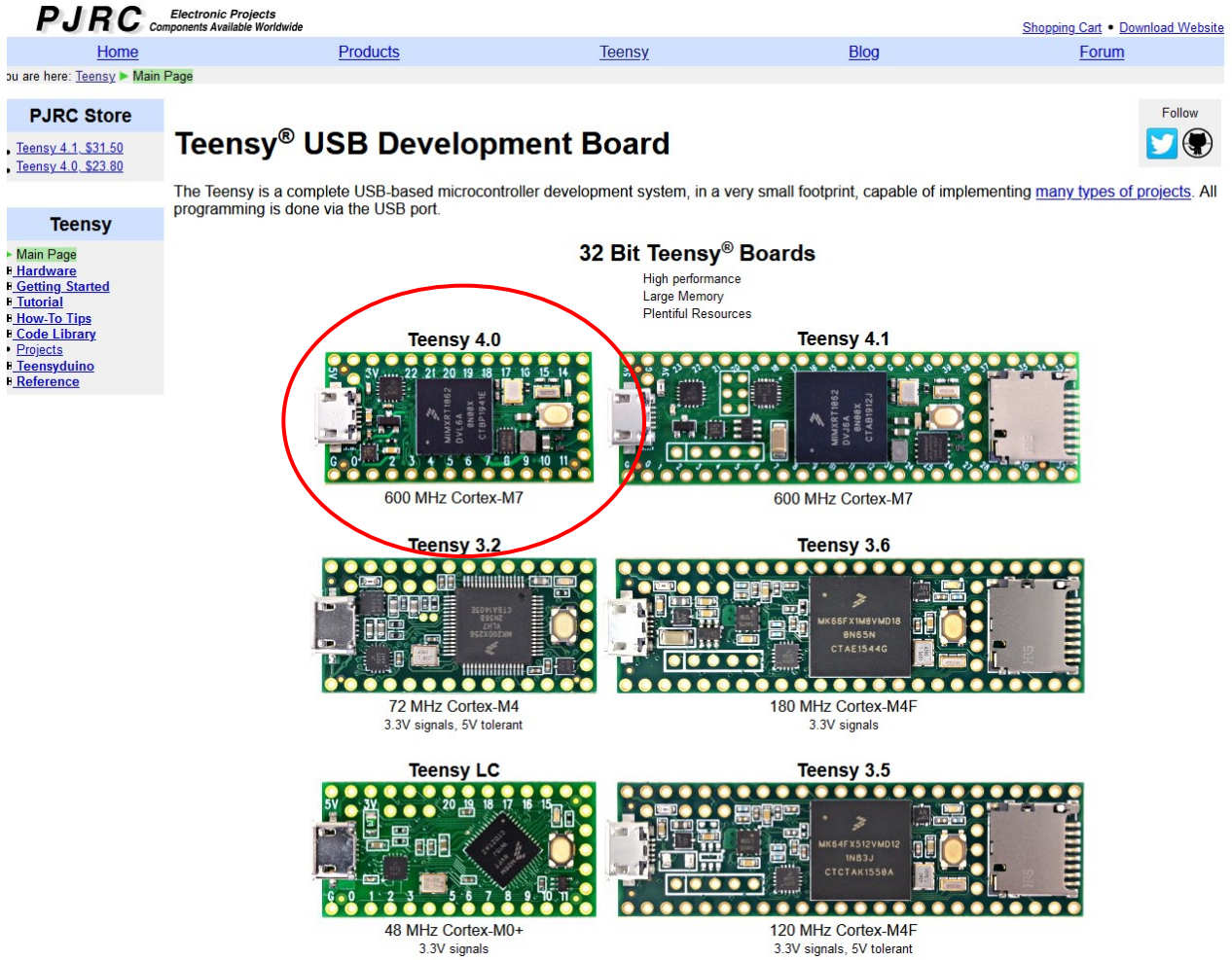
Sample	Time	Decoder	Ann Row	Ann Class	Value
33116120	4.140 s	MC145220	Register writes	Register write	STEER: SUB
33116120	4.140 s	MC145220	Synth datas	Synth data	flo1 = 93300000.0 Hz
33116168	4.140 s	SPI	MOSI bits	MOSI bit	0
33116168	4.140 s	MC145220	Register writes	Register write	OUTPUT A FUNCTION: 0
33116216	4.140 s	SPI	MOSI bits	MOSI bit	0
33116264	4.140 s	SPI	MOSI bits	MOSI bit	0
33116264	4.140 s	MC145220	Register writes	Register write	PRESCALE RATIO: 32/33
33116312	4.140 s	SPI	MOSI bits	MOSI bit	0
33116312	4.140 s	MC145220	Register writes	Register write	N COUNTER: 19
33116360	4.140 s	SPI	MOSI bits	MOSI bit	0
33116408	4.140 s	SPI	MOSI bits	MOSI bit	0
33116456	4.140 s	SPI	MOSI bits	MOSI bit	0
33116540	4.140 s	SPI	MOSI data	MOSI data	13
33116540	4.140 s	SPI	MOSI bits	MOSI bit	0
33116588	4.140 s	SPI	MOSI bits	MOSI bit	0
33116636	4.140 s	SPI	MOSI bits	MOSI bit	0
33116684	4.140 s	SPI	MOSI bits	MOSI bit	1
33116732	4.140 s	SPI	MOSI bits	MOSI bit	0
33116780	4.140 s	SPI	MOSI bits	MOSI bit	0
33116828	4.140 s	SPI	MOSI bits	MOSI bit	1
33116876	4.140 s	SPI	MOSI bits	MOSI bit	1
33116960	4.140 s	SPI	MOSI data	MOSI data	0E
33116960	4.140 s	SPI	MOSI bits	MOSI bit	0
33116960	4.140 s	MC145220	Register writes	Register write	A COUNTER: 14
33117008	4.140 s	SPI	MOSI bits	MOSI bit	0
33117056	4.140 s	SPI	MOSI bits	MOSI bit	0
33117104	4.140 s	SPI	MOSI bits	MOSI bit	0
33117152	4.140 s	SPI	MOSI bits	MOSI bit	1
33117200	4.140 s	SPI	MOSI bits	MOSI bit	1
33117248	4.140 s	SPI	MOSI bits	MOSI bit	1
33117296	4.140 s	SPI	MOSI bits	MOSI bit	0
33117652	4.140 s	SPI	MOSI transfers	MOSI transfer	4A 76
33117700	4.140 s	SPI	MOSI data	MOSI data	4A
33117700	4.140 s	SPI	MOSI bits	MOSI bit	0

Stoute schoenen

- Ik wist genoeg! De oude microprocessor mag met welverdiend pensioen.
- Het “ding” met de blauwe sticker is een “battery backed RAM” (tijdbom?) met alle kanalen erin… Deze wordt ook niet meer gebruikt.

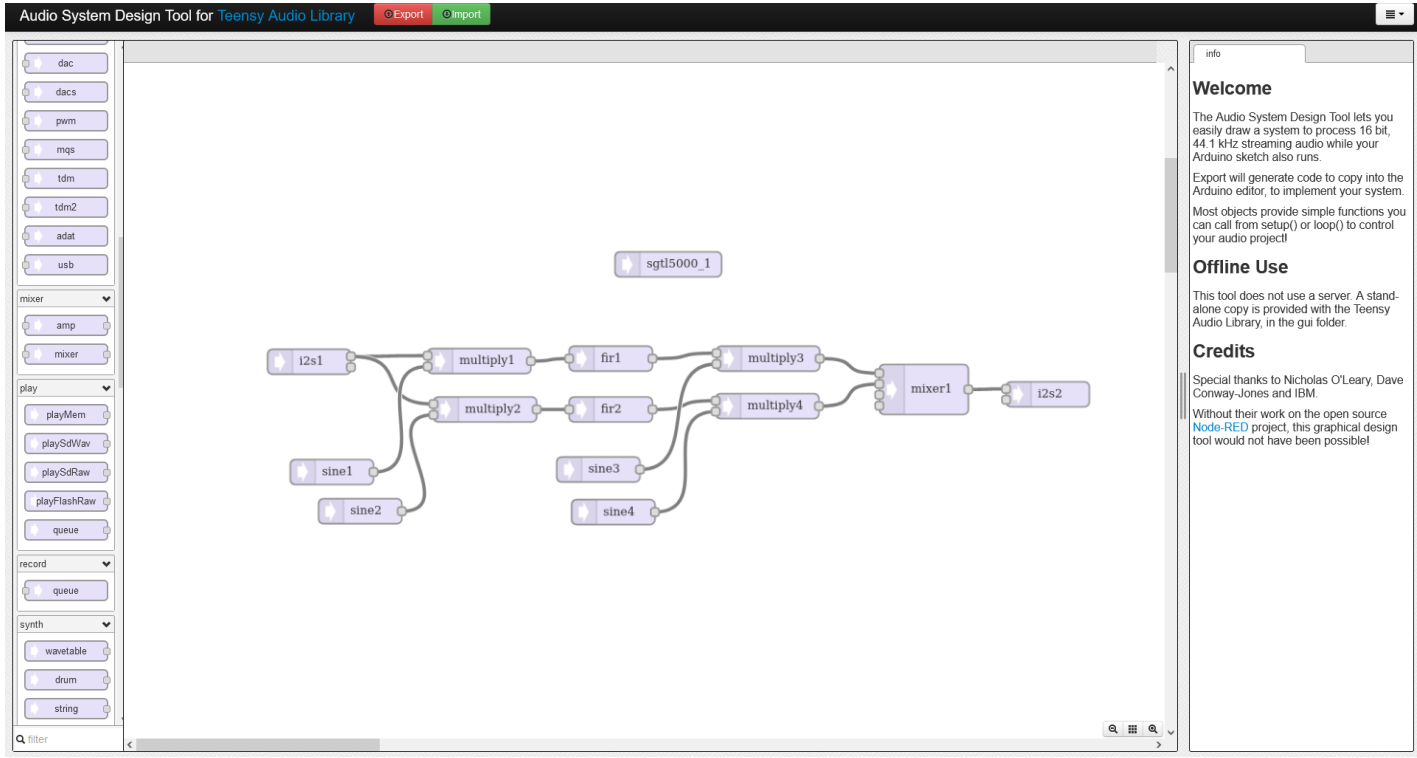
Teensy 4.0 Microcontroller

- ARM Cortex-M7, 600 MHz
- Audio library
- Zie ook bijvoorbeeld DD4WH convolution SDR project: https://github.com/DD4WH/Teensy-ConvolutionSDR
- Te programmeren met behulp van de Arduino ontwikkelomgeving



Teensy Audio system design tool

- Vermenigvuldigers
- Versterkers
- FIR & IIR filters
- Optellers
- Sinus / cosinus generatoren
- → Hiermee kun je een hele SDR maken ☺…



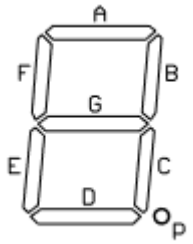
Broncode inkloppen…

- “Gewoon” in de Arduino IDE
- Automatische generatie van Arduino broncode voor Lookup Tabellen dmv Python:
  - Tabellen met PLL deeltallen op basis van de “magic number” getallen

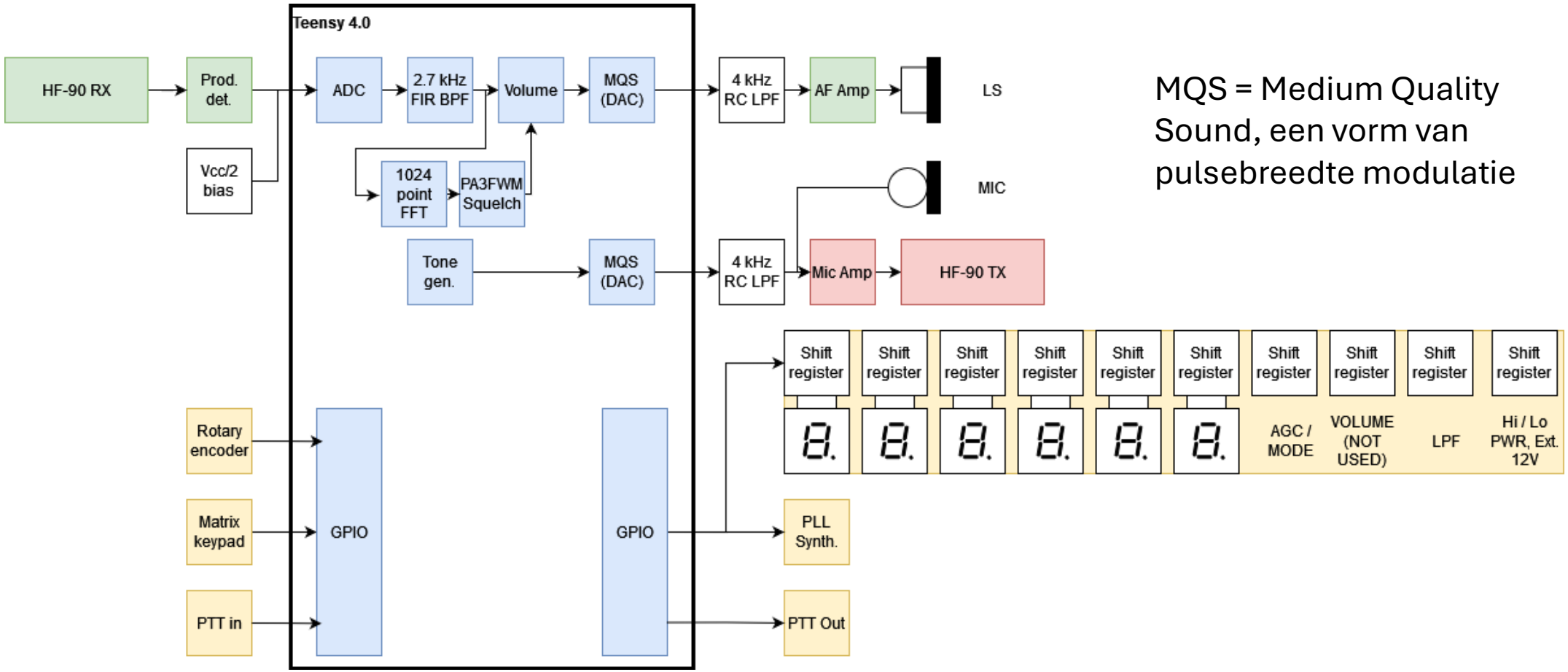
```
#ifdef DUAL_SYNTH
/*
 Synthesizer Look Up Tables
 */

const int LO2_MID_IDX = 616; // the index in the LO2 LUT that corresponds to 83616000 Hz
const int LO1_STEPSIZE = 20000; // Hz
const int LO2_STEPSIZE = 25; // Hz
/*
 LO1 / SUB N
 */#define LO1_NPTS 1401

const short Ns_LUT[LO1_NPTS] = {
66,
66,
```

  - Opzoektabellen voor 7-segments display (dus: welke segmenten moeten aangestuurd worden om een bepaald karakter weer te geven)

```
/*
 Seven segment Look Up Tables
 */

/*
 use an array of structs in a dict-type fashion to look up 7 segment display segments for each alphanumeric input character
 */
typedef struct { // frequency memory
String alphanum; // corresponding alphanumeric character
uint8_t segments; // segments which are lit
} segdict;

const uint8_t nalphas = 69;

const segdict lookup_segments[] = {
{"0", 246},
{"1", 34},
{"2", 218},
{"3", 122},
{"4", 46},
{"5", 124},
{"6", 252},
{"7", 50},
{"8", 254},
```



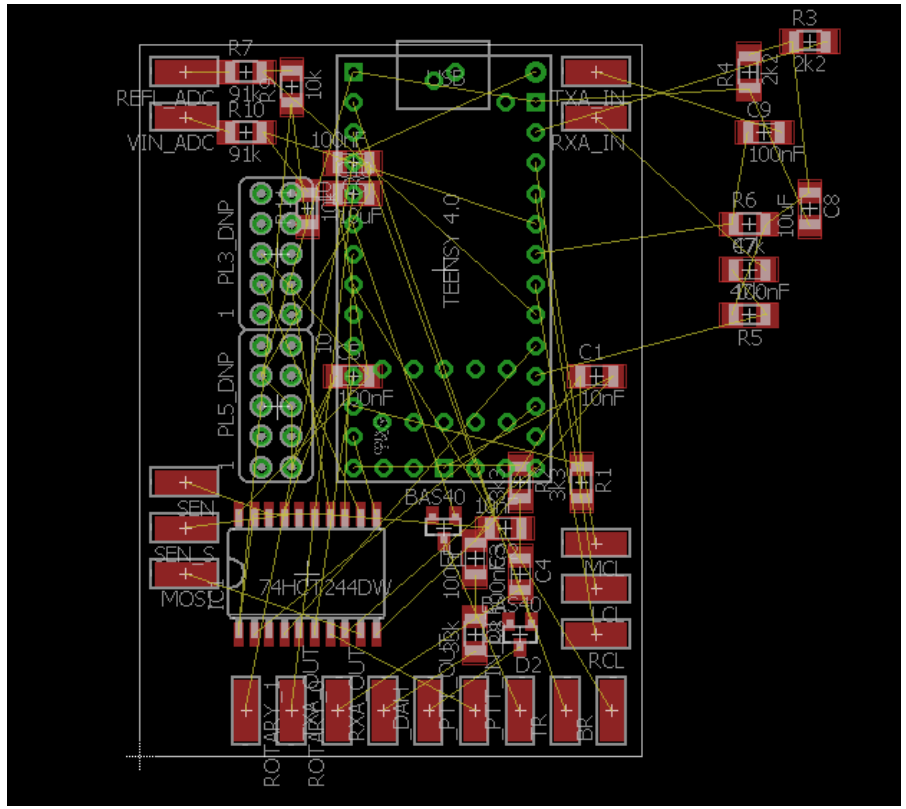
Blokschema

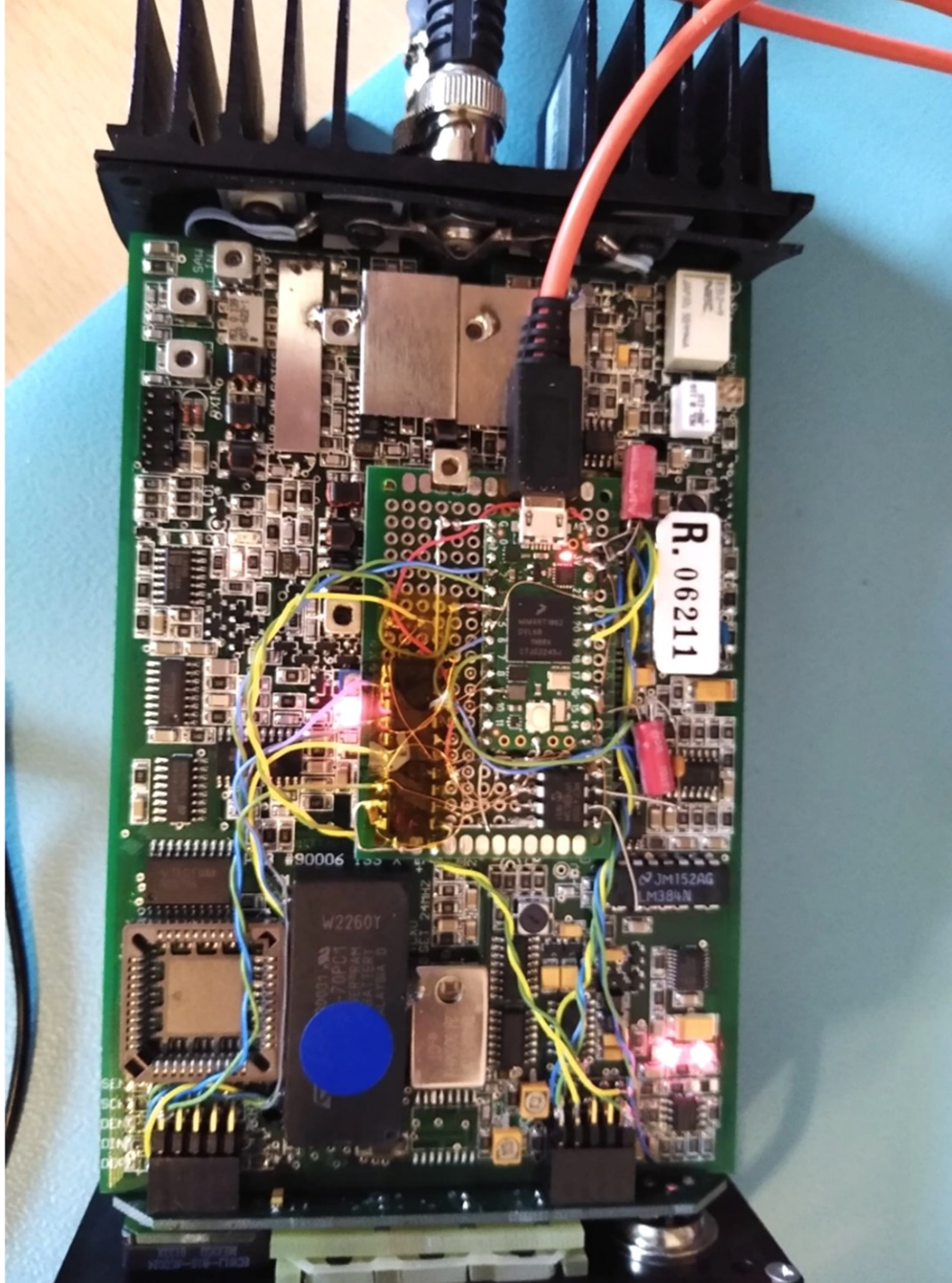


MQS = Medium Quality Sound, een vorm van pulsebreedte modulatie

Is er ook een printje van?

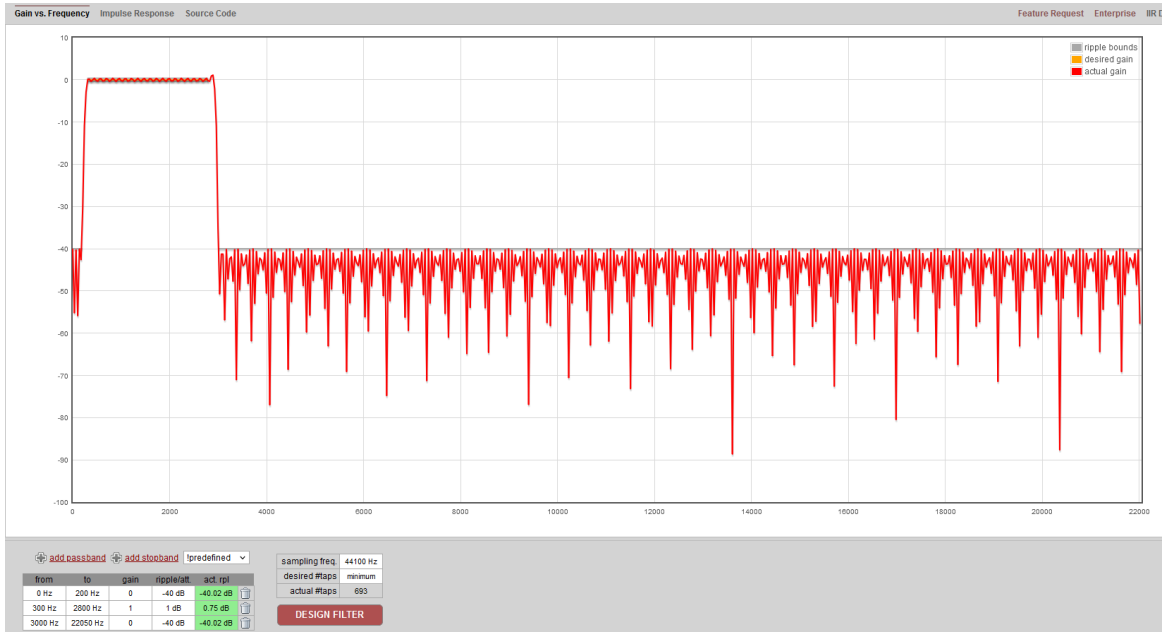
- *Nog* niet, wordt aan gewerkt in de wintermaanden ☺

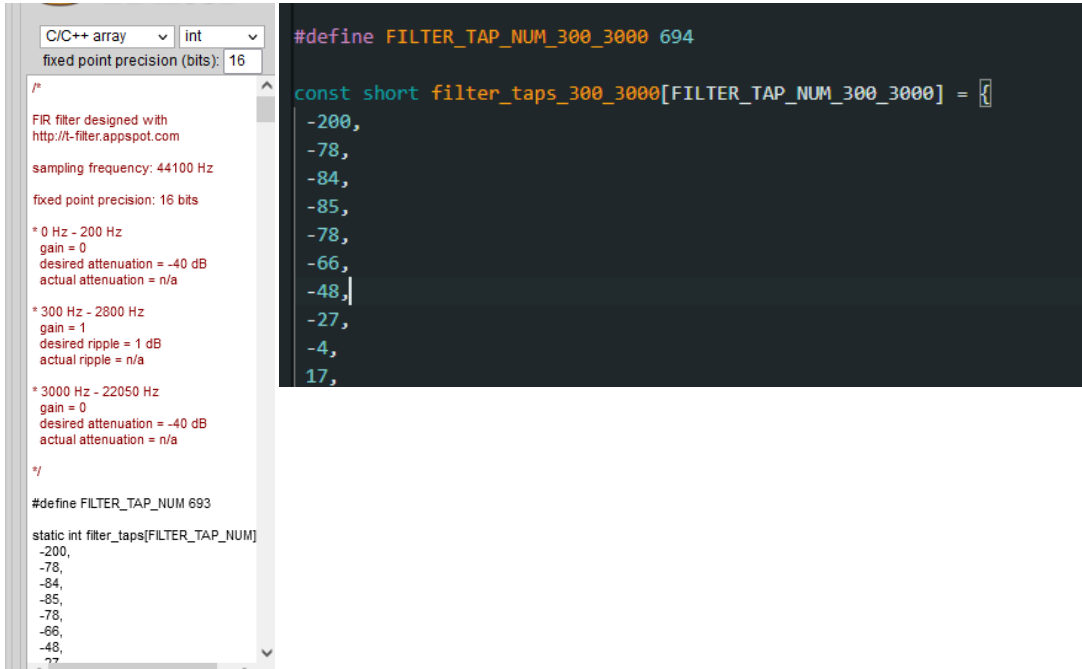




300 – 3000 Hz audio FIR Bandpass filter

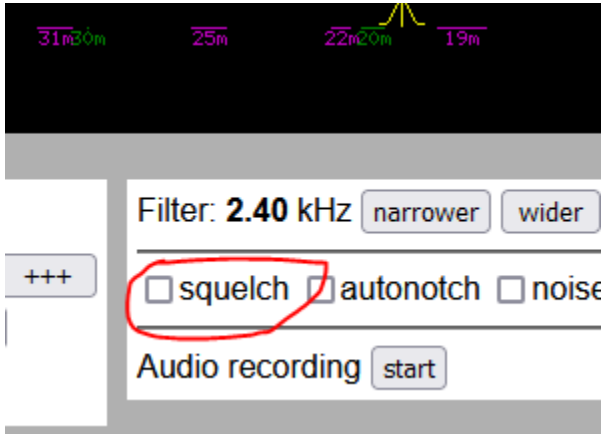
- Coefficienten berekend met T-filter design tool: http://t-filter.engineerjs.com/
- 44.1 ksps, decimeren naar bijv. 8 ksps zou beter zijn!
- Filter coefficienten zijn direct te plakken in de Arduino broncode





PA3FWM SSB squelch

- https://www.pa3fwm.nl/technotes/tn16f.html
- Berekent de zogenaamde “relatieve variantie” van een FFT van het ontvangen (SSB) audio
- dit werkt ZEER goed! (Probeer het maar eens op de Twente WebSDR)
- Leent zich goed voor “stand-alone” implementatie



New squelch algorithm for the WebSDR

Pieter-Tjerk de Boer, PA3FWM web@pa3fwm.nl

(This is an adapted version of part of an article I wrote for the Dutch amateur radio magazine *Electron*, January 2018.)

For the WebSDR I recently developed a new squelch algorithm, which in practice doesn't need a manually set threshold. This squelch looks at the signal in the frequency domain.

See the figure. At the left, we see part of a waterfall diagram, in which occasionally a USB signal is present (someone calling CQ contest, in this case). We immediately recognize the moments at which the signal is there, because those are brighter: there is more power, which principle (1) of the previous page uses. We also see that during the transmissions the audio frequencies between 200 and 600 Hz are most prominent (this is a USB signal on 21114 kHz, so those frequencies land between 21114.2 and 21114.6 kHz), which principle (4) uses. But what is also clear is that when there's no transmission, all pixels on a line are approximately equally bright (dark blue in this case), while during the transmissions there are large differences (from dark blue to white). My idea is to use this: if the differences are small, the squelch must close, and it must open otherwise. This general idea still needs to be translated into an algorithm that can be programmed in a computer.

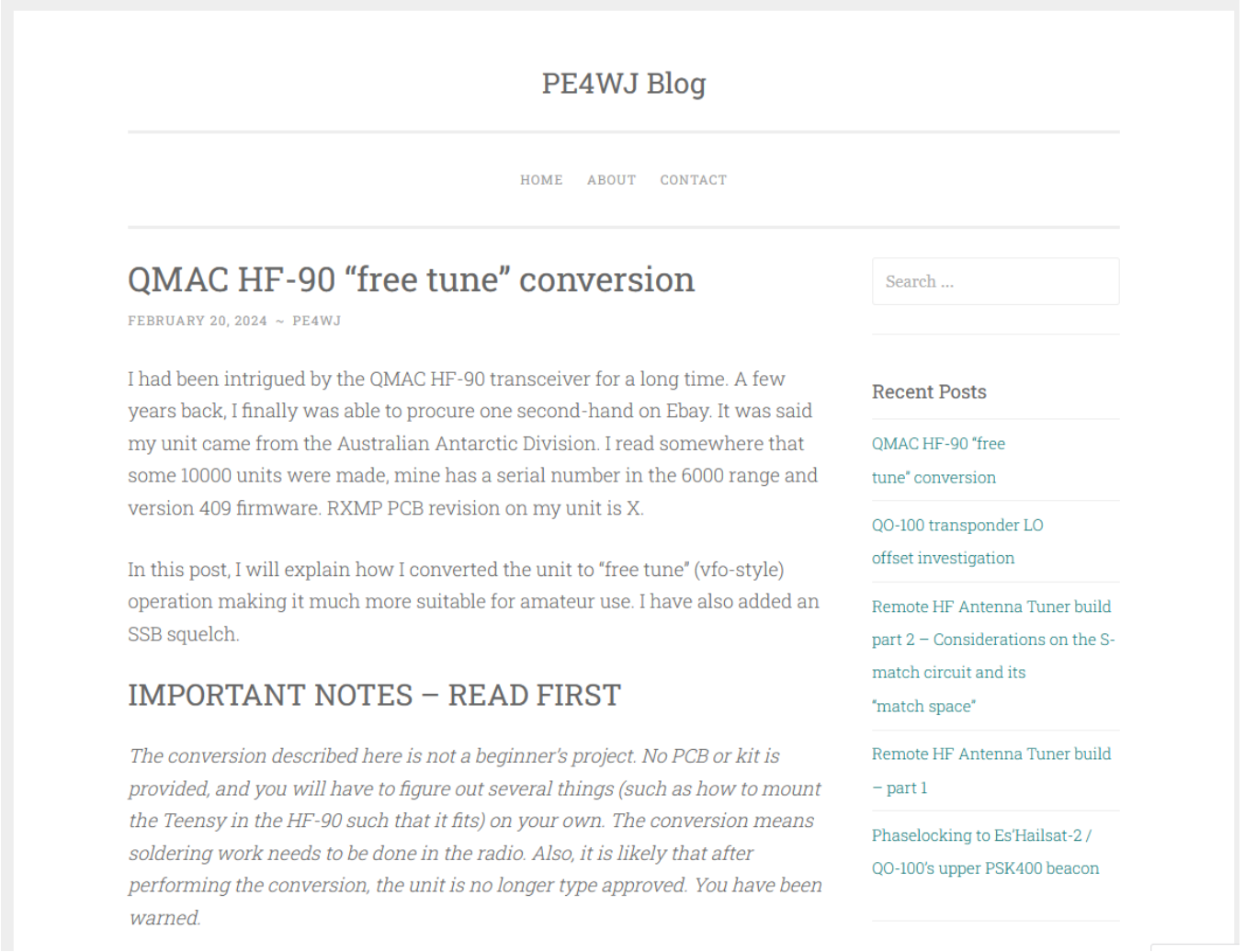
To measure the amount of variation in the spectrum, we'll use a quantity which I'll call the relative variance: this is the variance divided by the square of the average. It's a number that indicates how much a set of measurement values are spread out, but relative with respect to their average. Practically, this means that the number only depends on the signal/noise *ratio*, not on the absolute strength of both signal and noise. So we'll repeatedly take one horizontal row from the waterfall diagram (i.e., one set of FFT output bins, one line marked in green in the diagram), and compute the relative variance of the brightness of the pixels on that line. The results is shown at the right.

PA3FWM SSB squelch

```
if (myFFT.available()) {
  // each time new FFT data is available

  // calculate the mean of the data
  sum = 0;

  n_bins = high_idx - low_idx + 1;  // the number of bins to be processed in the FFT for a 1024 point FFT
  for (i = low_idx; i <= high_idx; i++) {
    sum += myFFT.read(i);  // sum over all bins in the FFT
  }
  mean = sum / float(n_bins);  // divide the sum by the number of datapoints

  // calculate the sum of the squares of the differences from the mean
  sum = 0;  // re-initialize sum
  for (i = low_idx; i <= high_idx; i++) {
    sum += pow((mean - myFFT.read(i)), 2);  // sum over all bins in the FFT
  }

  // from that, calculate the variance
  variance = sum / float(n_bins);  // divide the sum by the number of datapoints

  // now calculate the relative variance by dividing the variance by the mean squared
  relvar = variance / (mean * mean);

  // simple state machine - determine if a signal is present, but only when the squelch is not hanging on a previous signal

  if (relvar > thresh) {
```

Tijd om te publiceren!

- https://pe4wj.wordpress.com/
- Alle broncode, scripts: https://github.com/pe4wj/hf90f



PE4WJ Blog

HOME ABOUT CONTACT

QMAC HF-90 “free tune” conversion

FEBRUARY 20, 2024 ~ PE4WJ

I had been intrigued by the QMAC HF-90 transceiver for a long time. A few years back, I finally was able to procure one second-hand on Ebay. It was said my unit came from the Australian Antarctic Division. I read somewhere that some 10000 units were made, mine has a serial number in the 6000 range and version 409 firmware. RXMP PCB revision on my unit is X.

In this post, I will explain how I converted the unit to “free tune” (vfo-style) operation making it much more suitable for amateur use. I have also added an SSB squelch.

IMPORTANT NOTES – READ FIRST

The conversion described here is not a beginner’s project. No PCB or kit is provided, and you will have to figure out several things (such as how to mount the Teensy in the HF-90 such that it fits) on your own. The conversion means soldering work needs to be done in the radio. Also, it is likely that after performing the conversion, the unit is no longer type approved. You have been warned.

Search …

Recent Posts

QMAC HF-90 “free tune” conversion

QO-100 transponder LO offset investigation

Remote HF Antenna Tuner build part 2 – Considerations on the S-match circuit and its “match space”

Remote HF Antenna Tuner build – part 1

Phaselocking to Es’Hailsat-2 / QO-100’s upper PSK400 beacon

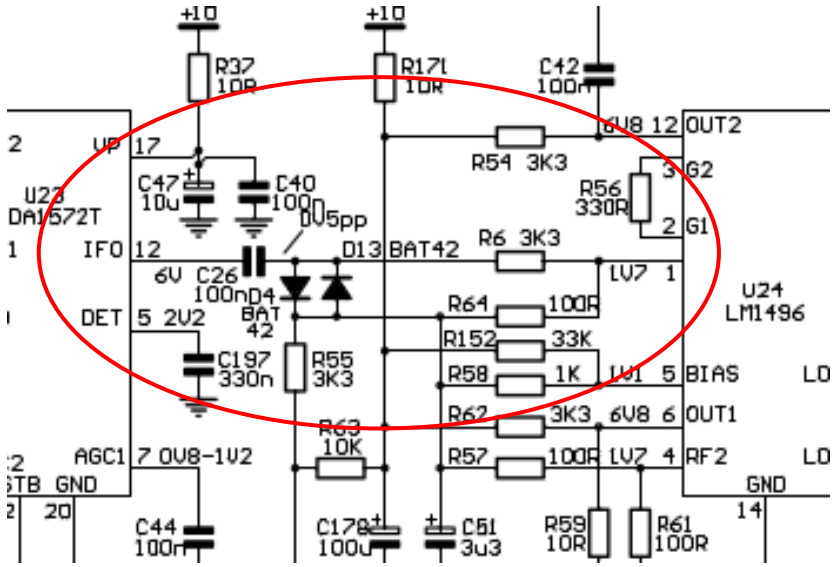
E-mail van Mr. QMAC himself ☺

Comment:
Halo Wouter Jan,
My old friend and colleague Keith Perry VK6QA sent me a link to your blog on HF-90 mods as he thought I would be interested.
He was right!
What a wonderful blog and sterling work done on making the HF-90 more ham friendly.
Could you send me a pdf of the "article" as I'd love to peruse it offline?
As you say it is not a beginners project. It must have taken you many hours.
The radio was designed to have as few user settings as possible for use by aid agencies, 4WD outback types, expeditioners, emergency services and soldiers.
The intention always was to make a frequency hopping version which could command a higher price. The original HF-90 was launched in 1996 with the first unit being used by an expedition on the Kokoda Trail in Papua New Guinea. It can be heard in the documentary that was made:
https://www.youtube.com/watch?v=Af55iC3Lq44
By 1999 the software was ready for the hopping version and that increased sales significantly. Manufacture continued for a couple of runs beyond the sale of Q-MAC to Barrett in 2009. Barrett was bought by Motorola a year or two back.
Originally the radio was to use the TCA440 IF chip which was made by Philips and Siemens and is a better chip all round but sadly this was discontinued before HF-90 reached production. Later the TCA440 in SMD version was revived by a former ex DDR chip manufacturer and it was my intention to use it but by then it was 2009 and other events transpired...
I see from your QRZ page that you have visited Western Australia.
I have been to Utrecht and Amsterdam but never to Voorburg.
I still use a couple of HF-90s, one on FT8 which works amazingly well and another for occasional expeditions.
Anyway enough rambling.
73 de VK6MH GM4AWB Rod

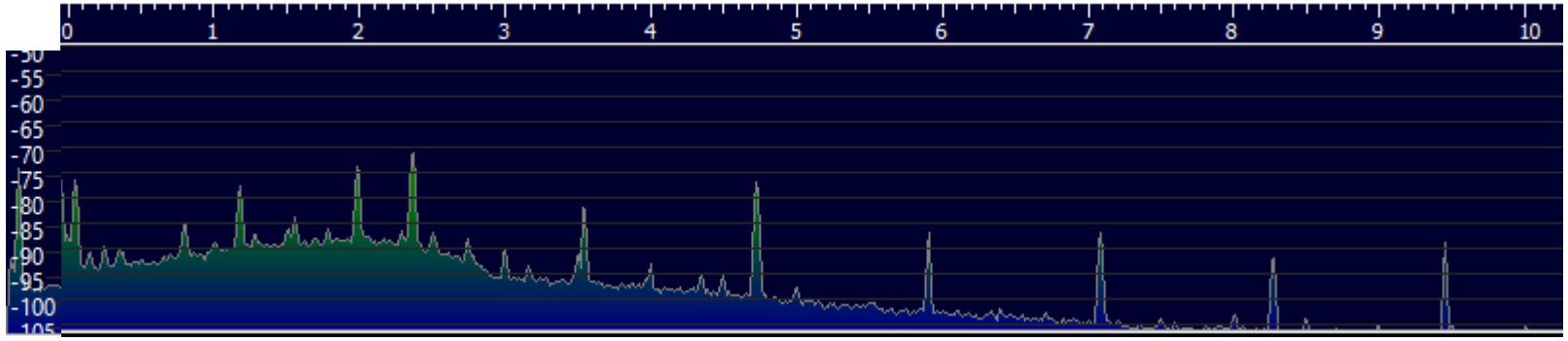
Intussen goed email contact en veel achtergrond informatie gekregen, inclusief de originele “magic numbers” zoals uitgerekend door zijn toenmalige XYL!

Oh ja, dat slechte audio…

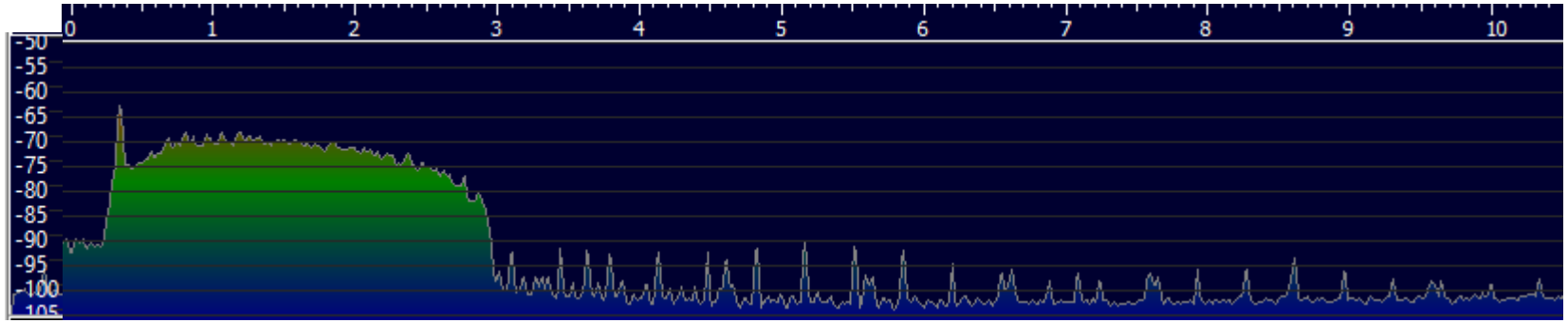
- Vervorming in:
  - DAC0800 die wordt gebruikt voor volumeregeling
  - HEF4053 voor TX/RX omschakeling
  - 2 antiparallele diodes in 455 kHz MF – de AGC is te graag!
  - Weerstanden aangepast in AGC circuit, werkt nu een stuk beter
- Display multiplex klok wordt opgepikt door audio voortrap, deze wordt nu niet meer gebruikt, er komt genoeg spanning uit de Teensy DAC
- RX audio door een 2.7 kHz DSP audio FIR filter in de Teensy, zo kan ik ook het volume regelen zonder de originele volume regeling DAC.
- Audio is nu prima op een paar zachte toontjes na die worden gemaskeerd door de bandruis



Audio voor / na vergelijking



Originele set, audio spectrum zonder antenne aangesloten



Omgebouwde set, audio spectrum zonder antenne aangesloten

Einde?

- De transceiver is hier nu in gebruik voor portabel werk op met name 80, 40 en 20m
- Inmiddels van PA0M / NL199 een defecte HF-90 ontvangen, oudere revisie met **2 aparte synthesizer chips**. Deze werkt inmiddels weer, na *alle* SMD Elco's te hebben vervangen… De software behoefde wel veel aanpassing omde andere synthesizer te ondersteunen…
- Interesse vanuit het buitenland (VK), en 1 amateur (G1HMO) heeft de ombouw succesvol gedaan
- PCB ontwerp + ombouwbeschrijving in de maak
- Intussen toegevoegd:
  - Roger piep
  - CW IAMBIC keyer en 500 Hz audio filter
  - Geheugenkanalen
  - Frequency hopping + FSK synchronisatie…

Voorbeeld menustructuur

Frequentie, CHAN knoppen stellen de afstemstap in

Mode

VFO / Memory

Squelch aan / uit

Dankwoord

- **Rod MacDuff VK6MH** voor een prachtig ontwerp, vol met “engineering trade-offs”
- **Pieter-Tjerk de Boer PA3FWM** voor het uitvinden van de “relatieve variantie” squelch
- **Paul Stoffregen** & consorten voor het hele Teensy ecosysteem en de audio library
- **Thieu Mandos, PA0M / NL199** voor het doneren van een 2e HF-90