Hoe laat is het?

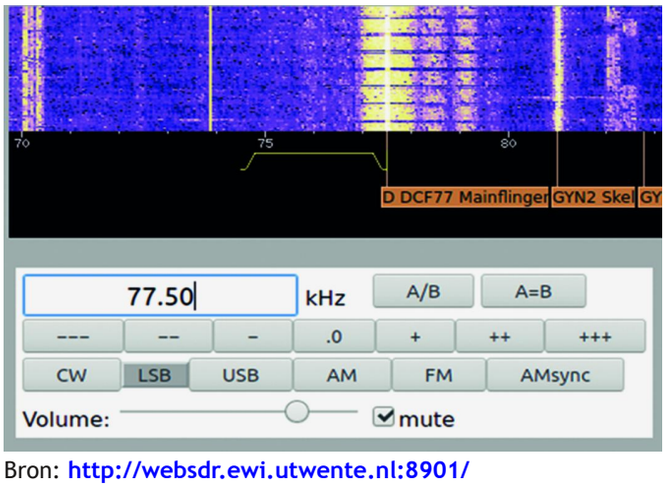
Johan Swenker

Als me die vraag gesteld wordt, dan kijk ik op mijn horloge en rond de tijd af op vijf minuten. Dat is meestal voldoende, ook als je ergens op tijd moet zijn. Zelfs het Zaanse klokje, dat ik van mijn ouders geërfd heb, loopt hiervoor nauwkeurig genoeg. Sinds ik met Oud-en-Nieuw het klokje aangezet heb, is het nog maar net een paar minuten voor gaan lopen.

DCF77

In mijn werkkamer hangt niet alleen dat Zaanse klokje, maar ik heb ook een 'atoomklok'. Zo kan ik tot op de seconde zien hoe ver het klokje voor loopt. Nu is die atoomklok niet echt een atoomklok, maar slechts een ontvanger voor de atoomkloktijd die vanuit Duitsland wordt uitgezonden.

In Mainflingen bij Frankfurt staat een zender die continu de wettelijke Duitse tijd uitzendt. Je hebt wel een speciale radio nodig om dit signaal te ontvangen. Mijn transistorradio kan de 77,5 kHz van deze zender niet ontvangen. Bij de universiteit Twente hebben ze echter een software defined radio (SDR) die deze frequentie wel kan ontvangen, en ook zichtbaar en hoorbaar kan maken.



Bron: http://websdr.ewi.utwente.nl:8901/

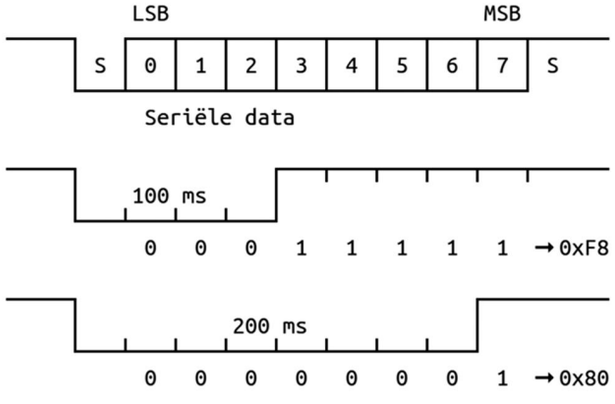
De witte band, midden in het paarse gebied, geeft het heel karakteristieke patroon van de DFC77-zender weer: 0,1 of 0,2 seconde rust, en de rest van de seconde tamelijk luide ruis. In de duur van de rust is de huidige tijd gecodeerd.

Details over de DCF77-codering kun je bij Wikipedia vinden: https://nl.wikipedia.org/wiki/DCF77. De Engelse Wikipedia legt uit dat ook ín die ruis informatie over de huidige tijd verborgen is.

DCF77-ontvanger voor de pc

Als je je pc nauwkeurig op tijd wilt laten lopen, kun je een DCF77-ontvanger aan je PC koppelen. Deze geeft de pulsen van 0,1 en 0,2 seconde door. Mijn oude ontvanger deed dat via de seriële poort.

Seriële data bestaat uit een start bit (0), 8 data bits (0 or 1), en een stop bit (1).



DCF77 codeert de '0' als een puls van 100 milliseconden. Op 40 baud ziet de seriële poort dan 0xF8 (decimaal 248).

De '1' is gecodeerd als een puls van 200 milliseconden. Op 40 baud ziet de seriële poort dan 0x80 (decimaal 128).

Dus zonder ingewikkelde digitale elektronica kan de ontvanger toch een digitaal signaal maken voor de seriële poort.

Op de software om deze ontvanger te kunnen gebruiken kom ik later terug.

Omgebouwde wekker

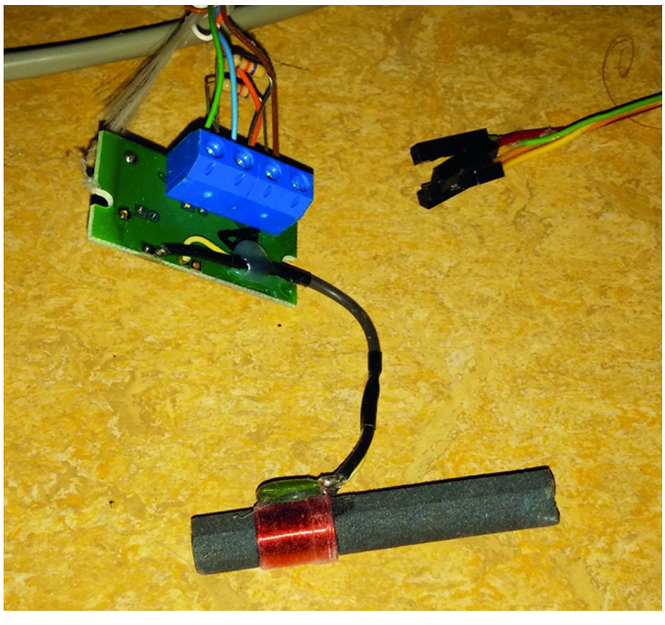
Ook mijn wekker wordt door de DCF77-zender aangestuurd; ik hoef de wekker nooit meer om te schakelen van wintertijd naar zomertijd of terug. Ook die informatie zit immers in het DCF77-signaal opgeborgen.

Als je zo'n wekker nu eens open maakt, en er met een analoge multimeter in gaat zitten zoeken, dan heb je kans dat je het karakteristieke DCF77-patroon vindt: de wijzer zwiept elke seconde omhoog of omlaag. Ik heb een keer speciaal voor dit doel een wekker gekocht, en het heeft gewerkt!

Conrad-module

Moderne computers hebben geen seriële poort meer. Als alternatief kun je een Raspberry Pi gebruiken. De GPIO-pennen van de Raspberry Pi zijn heel geschikt om te koppelen aan een DCF77-ontvanger. Dat gaat prima met de omgebouwde wekker.

In een restpartijenzaak in Groningen kocht ik een paar jaar geleden onderstaande module.



Dit is artikel nummer 641138 van Conrad, maar ik heb er op internet geen leverancier meer van kunnen vinden.

DCF77-software

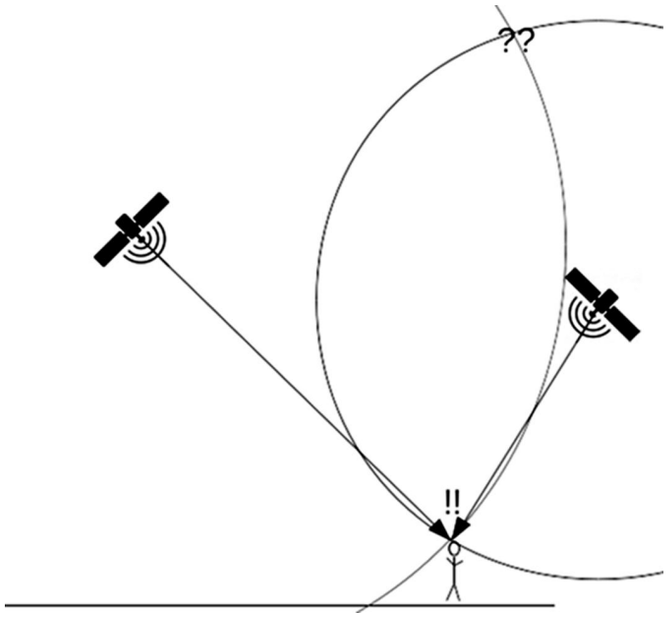
De pc moet de reeksen van 0xF8 en 0x80 omrekenen naar de exacte tijd, en dit daarna ook gebruiken om de klok van de pc bij te stellen. Op zich is dat best een leuke programmeerklus. Voor Linux heb ik dat wel eens gedaan. De computer zal dan ruim binnen een seconde gelijk lopen met de officiële tijd.

Voor MS Windows moet ik je verwijzen naar de leverancier van de hardware of naar internet. De volgende software heeft een zoekmachine voor mij gevonden. Maar let op, ik heb niet zelf getest!
https://download.cnet.com/Freeware-DCF77-Decoder/3000-20418_4-77875165.html of voor heel oude Windows: **http://www.rrs-web.net/in3her/dcf77_32.html**

GPS (Global Positioning System)

Deze sectie had natuurlijk GNSS (Global Navigation Satellite System) moeten heten, want er is meer dan alleen het Amerikaanse GPS. Ze werken allemaal volgens hetzelfde principe, of ze nu uit Europa (Galileo), Rusland (Glonass) of China (BeiDou) komen.

De satellieten zenden continu uit hoe laat het is. De lichtsnelheid is bekend, dus als je weet hoe laat je het signaal ontvangen hebt, dan kun je de afstand tot de satelliet uitrekenen. Het plaatje hieronder laat zien dat je in twee dimensies slechts twee afstanden nodig hebt om uit te kunnen rekenen waar je bent, vooropgesteld dat je niet ergens boven die satellieten zweeft.



Omdat we in een 3-dimensionale ruimte leven hebben we de afstand tot drie satellieten nodig om te bepalen waar we zijn. We hebben zelfs nog een vierde satelliet nodig.

We moeten niet alleen de positie in x, y en z uitrekenen. De GPS-ontvanger moet ook uitrekenen hoe laat het is. Dat uitrekenen van de tijd moet heel nauwkeurig gebeuren, want in één microseconde legt een radiogolf 300 meter af!

Location service op smartphone

Elke smartphone heeft tegenwoordig wel een GNSS-ontvanger. Deze wordt gebuikt door navigatieprogramma's die je locatie nodig hebben. Mijn smartphone kan GPS en Glonass ontvangen.

GPS-muis

Er zijn ook losse GPS-ontvangers die je kunt gebruiken met een navigatie-applicatie op je laptop. De GPS-ontvanger zelf weet heel nauwkeurig hoe laat het is. Maar dat vertelt 'ie niet altijd even precies aan de buitenwereld.

Deze GPS-muis gebruikt NMEA0193 om positie, datum en tijd door te geven. NMEA0183 is een standaard van de National Marine Electronic Association. Ik heb met veel plezier http://esr.ibiblio.org/?p=801 gelezen. Hierin vertelt Eric S. Raymond waarom NMEA0183 eigenlijk heel slecht is.

Jupiter

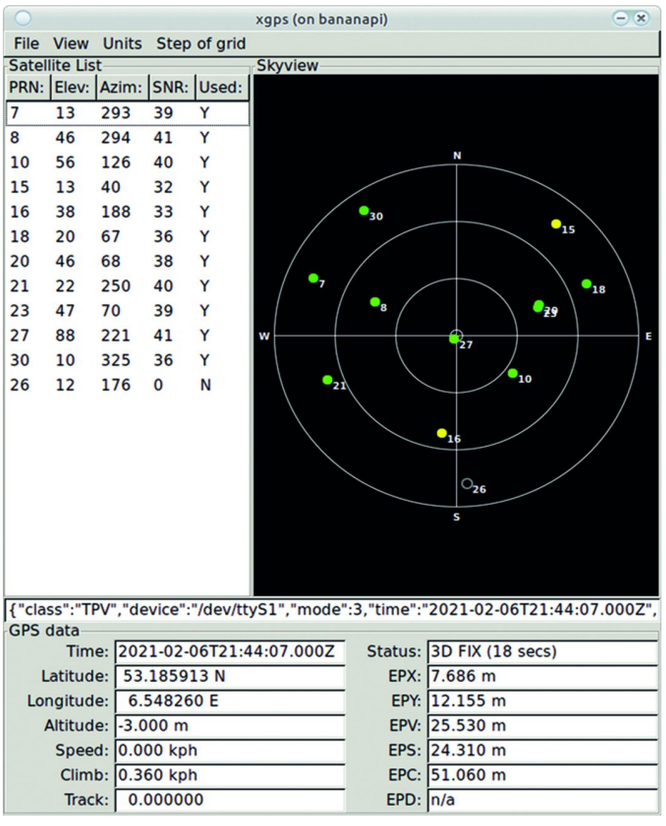
Jaren geleden heb ik bij http://www.gpskit.nl/ een Rockwell Jupiter GPS-ontvanger gekocht, compleet met kastje, antenne en handleiding. Deze werkt nog steeds, maar er is een klein probleempje met de firmware. De ontvanger beweert dat het nu 22 juni 2001 is. GPS verstuurt het weeknummer als getal van 10 bits. Daardoor zit de datum er nu precies 1024 weken naast. Door er op de pc weer 1024 weken bij te tellen, komt alles toch nog goed.

De Jupiter-module, en ook de gpskit, geven uiteraard positie, datum en tijd door. Ze hebben echter nog 2 extra signalen: een signaal van 10 kHz en een Pulse Per Second (PPS). Beide signalen zijn gesynchroniseerd met de GPS-tijd.

NMEA0183 en PPS

De NMEA0183-zinnen worden aan het begin van de seconde verstuurt. Dit kun je dus net als de DCF7-pulsen gebruiken om de pc tot op de seconde gelijk te laten lopen, en zelfs beter dan dat.

Maar het kan nóg veel beter. En ik moet toegeven, dat is vooral leuk omdat het kan. Ook dat PPS signaal kun je doorgeven aan de computer. In oudere computers werd hiervoor de Data Carrier Detectpen van de seriële poort gebruikt. Bij de Raspberry Pi wordt een van de vele GPIO-pennen gebruikt.



xgps geeft een grafische weergave van de NMEA-zinnen

Het programma ppstest laat zien dat de pulse per seconde meestal binnen de 10 microseconden nauwkeurig binnenkomt.

```
> ppstest /dev/pps0
trying PPS source "/dev/pps0"
found PPS source "/dev/pps0"
ok, found 1 source(s), now start fetching data...
source 0 - assert 1612647384.000021294, sequence: 533221 - clear  0.000000000, sequence: 0
source 0 - assert 1612647384.999986908, sequence: 533222 - clear  0.000000000, sequence: 0
source 0 - assert 1612647386.000007657, sequence: 533223 - clear  0.000000000, sequence: 0
source 0 - assert 1612647387.000001835, sequence: 533224 - clear  0.000000000, sequence: 0
source 0 - assert 1612647388.000013441, sequence: 533225 - clear  0.000000000, sequence: 0
source 0 - assert 1612647389.000008351, sequence: 533226 - clear  0.000000000, sequence: 0
source 0 - assert 1612647389.999986648, sequence: 533227 - clear  0.000000000, sequence: 0
source 0 - assert 1612647391.000011743, sequence: 533228 - clear  0.000000000, sequence: 0
source 0 - assert 1612647392.000007991, sequence: 533229 - clear  0.000000000, sequence: 0
source 0 - assert 1612647392.999997206, sequence: 533230 - clear  0.000000000, sequence: 0
source 0 - assert 1612647393.999986927, sequence: 533231 - clear  0.000000000, sequence: 0
source 0 - assert 1612647395.000002569, sequence: 533232 - clear  0.000000000, sequence: 0
^C
```

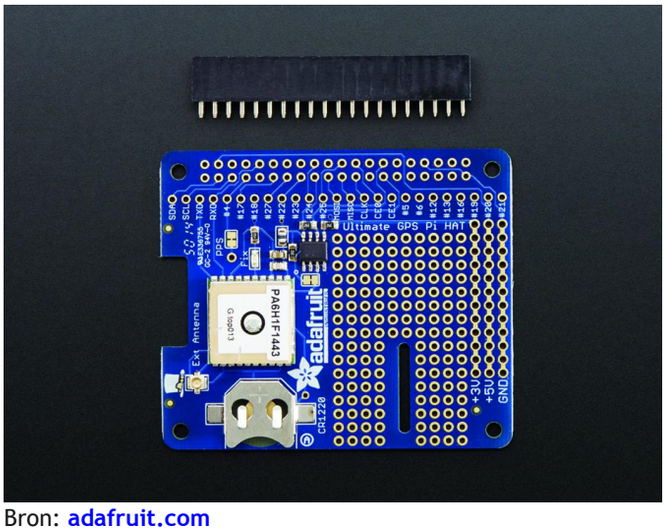
Losse GPS-module

Ik heb ook een keer met een wat modernere GPS-module gespeeld. Net zoals bij de GY GPS6MV2 op onderstaande foto, was die module bedoeld voor navigatie, en dan zijn de twee draadjes voor een seriële verbinding voldoende. De chip zelf levert echter ook de PPS. De pootjes op de chip zitten heel dicht op elkaar, maar het is me toch gelukt om een draadje aan het PPS-pootje te solderen, om daarmee mijn Raspberry Pi van een PPS-signaal te voorzien.

De chip op deze GPS-module is van de firma Ublox. Ublox heeft op haar website: **https://www.u-blox.com/en/product/u-center** GNSS evaluatiesoftware voor MS Windows staan. Ook deze software heb ik niet zelf getest, maar Ublox is een gerenommeerd merk, wat me enig vertrouwen geeft.

GPS HAT voor Raspberry Pi

Je kunt het nog makkelijker maken met een HAT (Hardware Attached on Top) die boven op de Raspberry Pi past. De Adafruit Ultimate GPS HAT for Raspberry Pi is daar een voorbeeld van. Bij Adafruit staat uitgebreide documentatie: **https://learn.adafruit.com/adafruit-ultimate-gps-hat-for-raspberry-pi/**
Deze HAT wordt ook geleverd door Nederlandse bedrijven, zoals Kiwi Electronics.



Bron: **adafruit.com**

NTP Network Time Protocol

Als je computer een internetverbinding heeft, dan kun je ook aan het internet vragen hoe laat het is. De standaard daarvoor is het Network Time Protocol. MS Windows gebruikt dit protocol op een heel simpele manier: een keer per dag vragen hoe laat het is, en dan de klok aanpassen. En meestal is dat goed genoeg.

Het protocol, en dan vooral de standaard client, kan veel beter dan dat. Mijn Linux-computer vraagt bij een aantal servers op internet op hoe laat het is, zoekt uit welke de beste is, en past daarop zijn eigen klok aan. De klok wordt niet alleen vooruitgezet of achteruit, maar ook de frequentie van de klok wordt aangepast. Het is dan niet meer voldoende om één keer per dag de tijd op de halen. NTP schakelt dynamisch tussen elke minuut of eens per kwartier.

De standaard NTP-client kan ook de tijd verwerken die door DCF77, NMEA0183 of PPS wordt aangeleverd. De standaard NTP-client is bovendien een NTP-server. Dus één Raspberry Pi die aan een GPS-ontvanger gekoppeld is, zorgt ervoor dat alle computers bij mij thuis nauwkeurig gelijk lopen met de 'echte' tijd.

```
$ ntpq -c pe
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
oPPS(0)          .PPS0.           0 l    3    8  377    0.000   -0.008   0.011
 PPS(1)          .PPS1.           0 l    -    8    0    0.000    0.000   0.000
*SHM(0)          .GPS0.           5 l   13   64  377    0.000    7.725   3.385
 SHM(1)          .GPS1.           5 l    -   64    0    0.000    0.000   0.000
+ntp1.time.nl    .MRS.            1 u   11   64  377   14.911   -0.429   0.114
 nano.swenker.or 10.20.40.27      2 u   46   64  377    0.292   -0.019   0.028
 zot.swenker.org 10.20.40.27      2 u   14   64  377    0.436    0.831   0.041
-raspi0.swenker. 10.20.40.14      3 u   10   64  377    0.435    0.157   0.019
+ntp.dicode.nl   193.79.237.14    2 u   46   64  377   12.163   -0.444   0.066
```

Kolommen delay, offset en jitter zijn in msec

Het programma ntpq laat met de parameters -c pe informatie zien over de peers (bronnen) die de NTP-server raadpleegt. PPS(0) is de PPS. De beide internetservers, die met een delay van iets meer dan 10 ms, bevestigen dat deze computer heel precies gelijk loopt met de echte tijd. SHM(0) is NMEA0183. Aan de kolom jitter is te zien dat NMEA0183 niet zo heel nauwkeurig is.

In een bedrijfsomgeving gebruik je geen Raspberry Pi, maar apparatuur die in een 19-inch rek past. Twee bekende leveranciers van dit soort apparatuur zijn Meinberg en Hopf. Zij leveren bovendien de standaard NTP-software, maar dan gecompileerd voor MS Windows: **https://www.meinbergglobal.com/english/sw/ntp.htm** en **https://www.hopf.com/ntp_en.php.**

Ik ga ervan uit dat die software, net als op mijn Linux-computers, in staat is om NMEA en DCF77 te verwerken.

Echt atoomklok-polshorloge

De website **http://leapsecond.com/** is helemaal toegewijd aan de speurtocht naar de nauwkeurigste klok. Helemaal over de top is natuurlijk het echte atoomklok-polshorloge, dat beschreven is onder het kopje *Most Accurate Wristwatch*.