Validating a Stratum 1 Network Time Protocol Server

Find the Correct Fudge

Jon Meek

Lawrenceville, NJ

meekj@ieee.org

Introduction

If you are interested in a precise network time source, such as recently described by Rudi van Drunen [1], you probably also want an accurate time source. This is especially true if you need to compare packet capture or log data, possibly to support a security investigation, from different administrative domains.

The goals of this project are to:

- Tune the NTP configuration to provide an accurate time source.
- Verify that the PPS (Pulse per Second) signal from the GPS receiver and computer agree on the start of second.
- Verify that the GPS PPS signal agrees with other standard time signals.
- Compare our time with reliable, and other, network time sources.

The test setup consists of the following:

- Garmin GPS 18x LVC Receiver
- Dell Optiplex GX150 Desktop PC with two "real" serial ports
- Agilent 54624A Digital Oscilloscope
- Sony SW7600 Shortwave Radio
- Trimble Resolution T Timing GPS Receiver

Using the Oscilloscope

We will display, in various combinations:

- The GPS PPS leading edge (within 1 µs (Garmin) or 15 ns (Trimble) of the true start of second)
- A pulse that represents the computer's start of second
- The audio second tones from CHU (Canada) and WWV (US) shortwave signals

Tuning the NTP Configuration

Step 1: Initial NTP Configuration Snippet using gpsd[2] to interface with the NTP SHM [3] clock driver

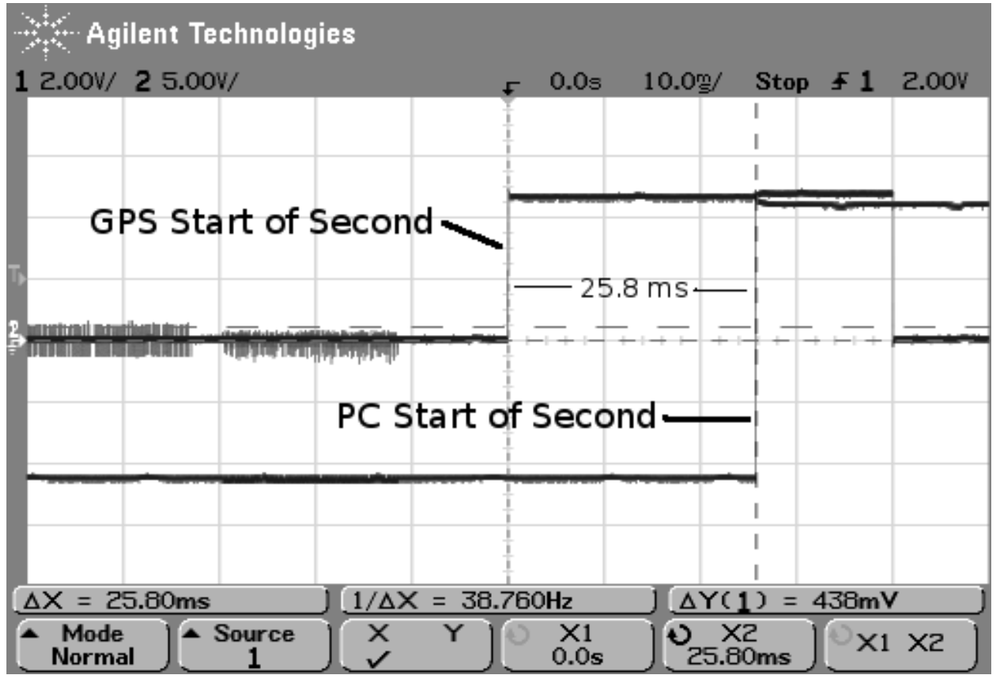
```
# Use gpsd as time source
# NMEA 0183 ASCII time, data appear long after PPS with considerable jitter
server 127.127.28.0 prefer
fudge  127.127.28.0 time1 0.633 refid GPS

# PPS - Leading edge is within 1 µs of UTC second start, try 100 µs offset
server 127.127.28.1 minpoll 4 maxpoll 4
fudge  127.127.28.1 time1 0.0001 refid PPS
```

After running for about one hour we have:

```
: dd1:~ ; ntpq -p
remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
LOCAL(0)         .LOCL.     10 l   58   64  377    0.000    0.000   0.001
xSHM(0)          .GPS.       0 l   16   16  377    0.000  107.990  21.692
xSHM(1)          .PPS.       0 l   11   16  377    0.000  -14.646   0.068
*vpn-ntp-gps.    .GPS.       1 u  129  128  377   19.521   28.036   0.744
+vpn-ntp-cdma.   .CDMA.      1 u   65  128  377   20.086   27.806   1.121
-time-C.timefreq .ACTS.      1 u    3  128  377   92.021   11.653   1.050
+rackety.udel.ed .PPS.       1 u   19  128  377   21.388   26.952   0.809
```

Delay, offset, and jitter values are reported in milliseconds.

Scope shows that PC start of second comes 25.8 ms after the GPS second which agrees closely with known good NTP servers



Step 2: We need to adjust the offset, taking a guess:

```
14.646 + (28.036 + 26.952 ) / 2 = 42.14 ms
```

Adjust the NTP configuration:

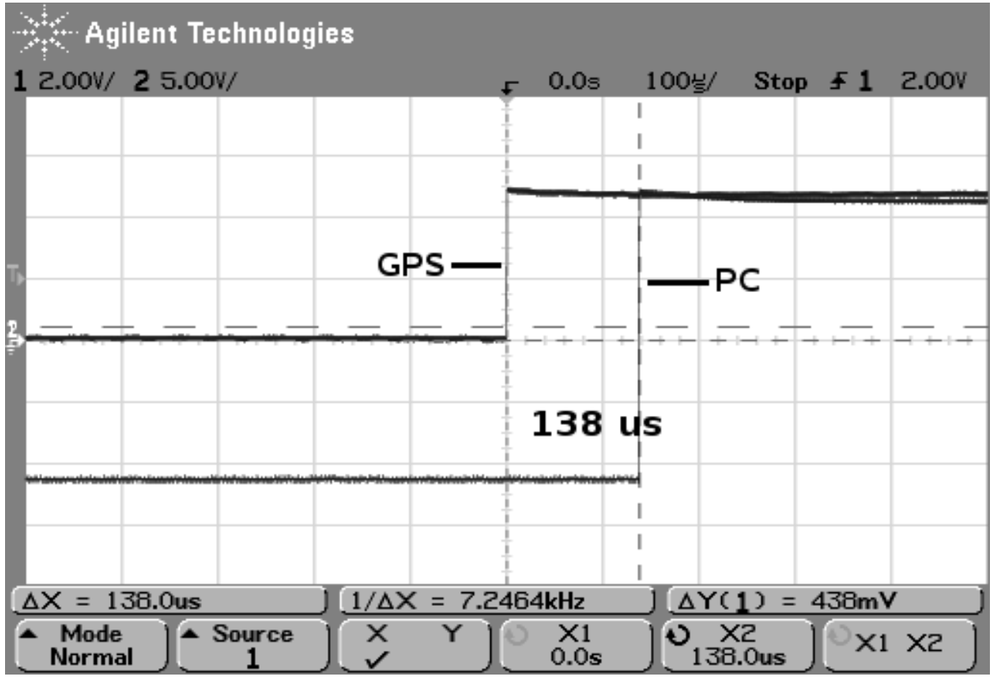
```
fudge 127.127.28.1 time1 0.04214 refid PPS
```

After running for a few hours, a second adjustment is made:

```
fudge 127.127.28.1 time1 0.040434 refid PPS
```

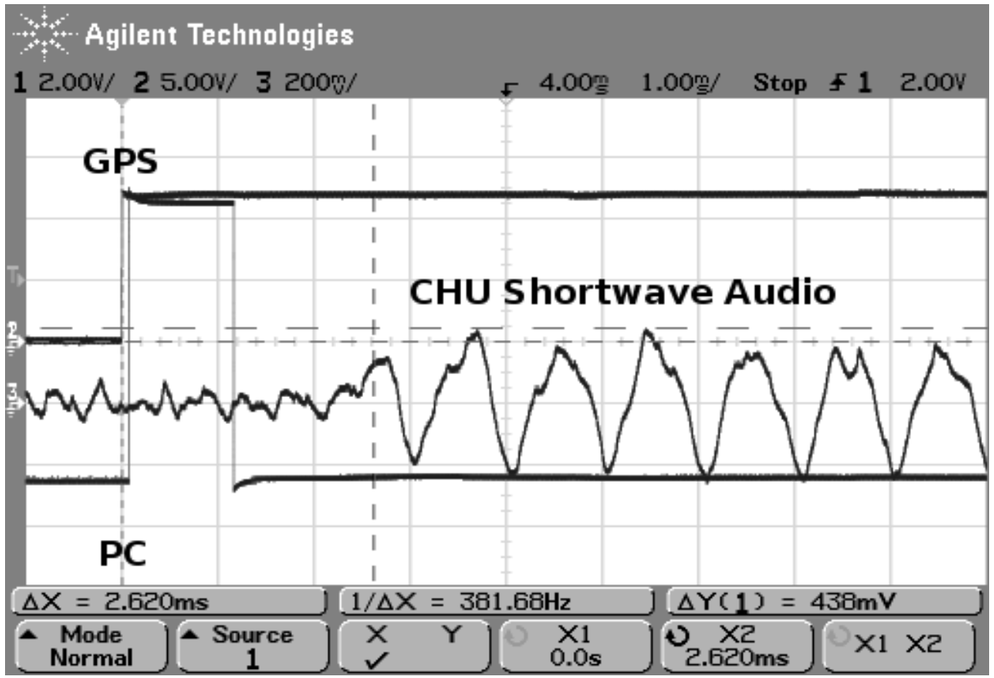
and about an hour later we have:

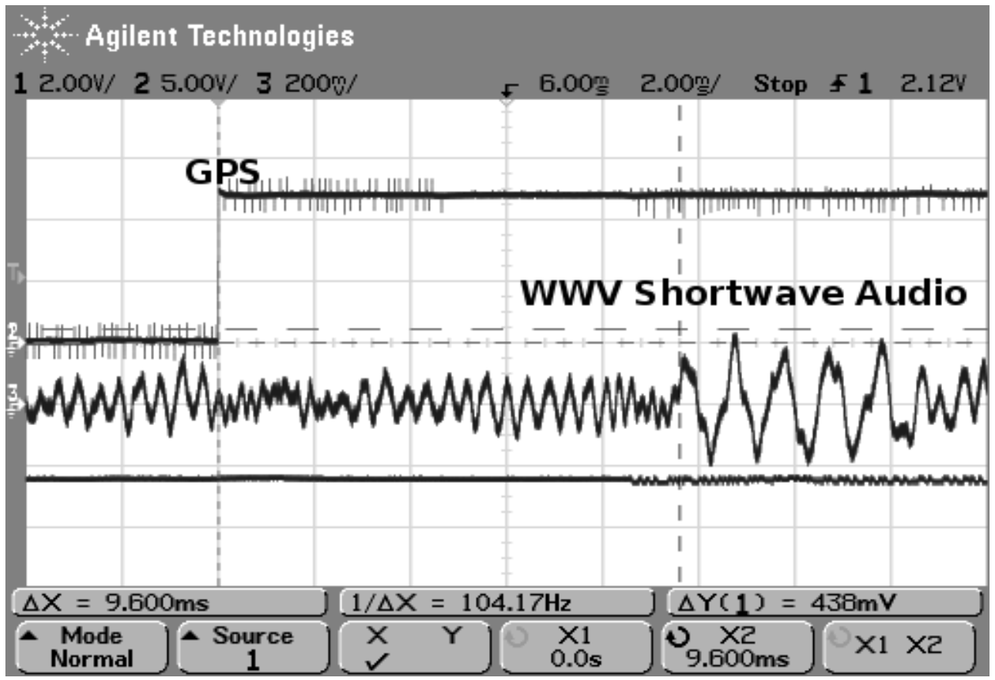
```
: dd1:~ ; ntpq -p
remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
LOCAL(0)        .LOCL.      10 l    2   64  377    0.000    0.000   0.001
xSHM(0)          .GPS.       0 l   11   16  377    0.000   68.019  15.405
*SHM(1)          .PPS.       0 l    4   16  377    0.000    0.145   0.009
+vpn-ntp-gps.    .GPS.       1 u    5   64  377   19.279    1.167   0.933
-vpn-ntp-cdma.   .CDMA.      1 u    6   64  377   21.493    2.322   0.347
xtime-C.timefreq .ACTS.      1 u   16   64  377   91.883  -15.493   0.566
+rackety.udel.ed .PPS.       1 u    6   64  377   21.415    0.225   0.772
xotc1.psu.edu    .WWV.       1 u    2   64  377   42.559  -26.780   0.193
```



The PC pulse arrives 138 µs after the GPS pulse, with jitter. This agrees reasonably with the 145 µs offset shown in the ntpq output.

Crosscheck with Independent Time Sources





First, we tune the shortwave radio to CHU Canada at 3330 kHz and feed the audio output to the oscilloscope's channel 3. The start of second pulse arrives about 2.6 ms after the GPS pulse. This agrees reasonably well with the 3 ms predicted by the propdelay program that is included with the NTP software distribution.

```
propdelay -C n40:18:03 w74:44:15
CHU summer propagation, height 350 km, hops 1, delay 0.00302628 seconds
CHU winter propagation, height 250 km, hops 1, delay 0.00253601 seconds
```

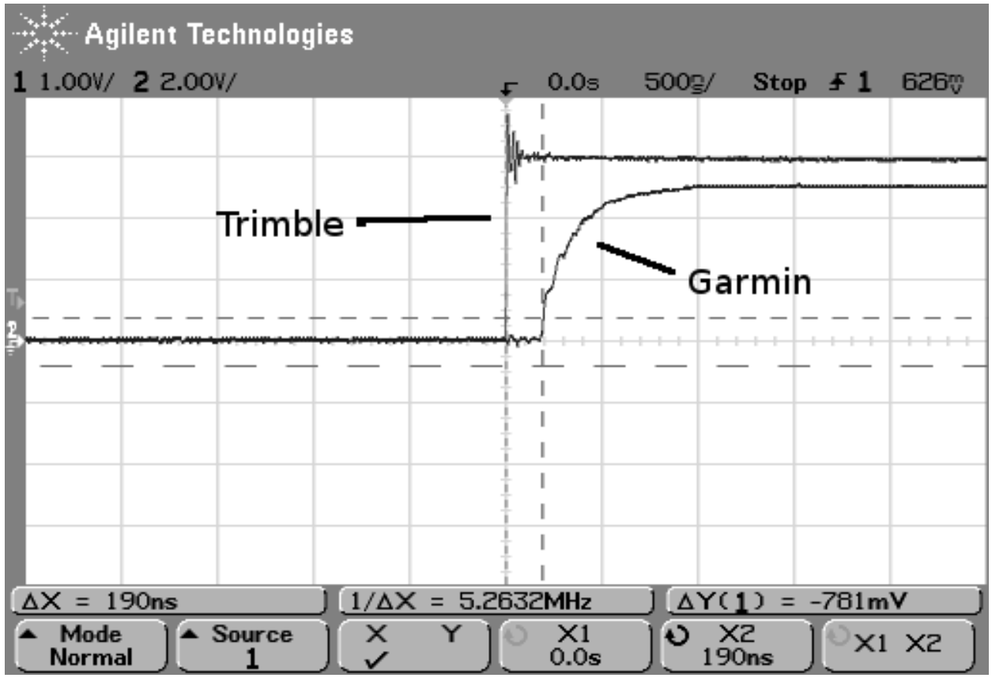
Then we check WWV US at 2500 kHz.

```
propdelay -W n40:18:03 w74:44:15
WWV  summer propagation, height 350 km, hops 1, delay 0.00908854 seconds
WWV  winter propagation, height 250 km, hops 1, delay 0.00887403 seconds
```

The 9.6 ms measured delay is a bit longer than the predicted 9.1 ms propagation delay.

As a final cross check, the Garmin PPS output is compared to the PPS output from a Trimble GPS timing receiver which is specified to be accurate to within 15 ns of UTC. The Garmin pulse (specified to be within 1 µs of UTC) usually arrives about 190 ns after the Trimble pulse, well within the specification.

The Garmin pulse has a 365 ns rise time due to the 5 meter, non-coax, signal cable. This could affect jitter at the serial port DCD input.



How are we doing after two days ?

```
: dd1:~ ; ntpq -p
remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
LOCAL(0)        .LOCL.      10 l   41   64  377    0.000    0.000   0.001
xSHM(0)          .GPS.       0 l    9   16  377    0.000   40.037  29.422
*SHM(1)          .PPS.       0 l   11   16  377    0.000   -0.002   0.001
+vpn-ntp-gps.    .GPS.       1 u   25   64  377   19.894    1.396   0.227
+vpn-ntp-cdma.   .CDMA.      1 u   46   64  377   20.017    1.763   0.504
-time-C.timefreq .ACTS.      1 u   51   64  377   92.771  -14.652   0.281
xrackety.udel.ed .PPS.       1 u   35   64  377  122.987  -49.522   1.110
xotc1.psu.edu    .WWV.       1 u   50   64  377   30.453  -20.275   0.947
```

Why Should You Run Your Own Stratum 1 NTP Server?

One reason: Sudden latency changes wreak havoc. In the ntpq output above, latency to the normally very reliable rackety.udel.edu suddenly jumped from 22 ms to 123 ms resulting in a large shift in its' apparent offset.

Tuning Other Systems

We used the NTP tuning technique with a PC Engines ALIX3D3 low power single board computer. The GPS 18x LVC was connected via a FTDI based USB serial port adapter that handles the PPS input on the DCD line (some USB adapters don't, and the ALIX3D3's onboard serial ports do not have hardware control lines). We found that

the fudge time1 value needed to be 41.4 ms, compared to 40 ms for the primary test system. The ALIX3D3 might make a low cost, rugged Stratum 1 NTP server, but the PPS jitter is about 500 µs, presumably due to USB overhead. We will need to use a GPIO pin for the PPS signal, like Rudi [1] did with Soekris hardware.

We also tested a 500 MHz Dell desktop and found that the `fudge time1` parameter did not change from the value required on the 1 GHz system. A quick test of the NTP Generic NMEA GPS Receiver Clock (Type 20) driver suggests that a significant calibration offset value is required there as well.

Conclusions & Questions

- We have shown one method to fine tune the NTP configuration of a Stratum 1 server so that it provides time accurate to about 100 µs when directly compared to the GPS PPS signal.
- We have compared Start of Second signals from the GPS Receiver, Synchronized NTP Server, CHU and WWV Shortwave.
- The Start of Second signals agree to about 1 millisecond.

- Is the GPS PPS leading edge really the correct Start of Second within 1 microsecond ? (Yes)
- Are our commercial NTP appliances (vpn-ntp-{gps,cdma} above) off by 1-2 milliseconds ? (Probably not)
- What about some Internet sources that appear to be off by 15, or more, milliseconds ? (Select carefully)

Acknowledgements

Thanks to Jim Trocki for early discussions and Rick Thomas for reviewing a draft of this poster.

References

[1] ;login: USENIX Association, August 2009, “A Home Built NTP Appliance”, Rudi van Drunen.
[2] http://gpsd.berlios.de/
[3] http://www.eecis.udel.edu/~mills/ntp/html/refclock.html

Addendum – January 2010

The PPS time1 value of 40.32 ms determined in this work seems quite large. Further investigation showed that gpsd -2.39 was using the falling, rather than the rising, edge of the PPS pulse as the start of the second. An adaptive algorithm in gpsd is supposed to select the proper PPS leading edge so that the user does not have to know the polarity (or have access to an oscilloscope). Changing to gpsd-2.90 solved this problem and good accuracy is obtained with time1 set to 3.2 ms. It would be nice if there was a way to optionally specify the proper leading edge in the gpsd configuration to eliminate uncertainty related to edge selection.

The good news is that the calibration technique described here corrected for the problem and the result was an accurate NTP server.