Time Keeping with the Network Time Protocol

Whitham D. Reeve

Abbreviations in this article:
CPU: Central Processing Unit
GNSS: Global Navigation Satellite System
IP: Internet Protocol
LAN: Local Area Network
NIST: National Institute of Standards and Technology
PC: Personal Computer
ppm: Parts Per Million
NTP: Network Time Protocol
UTC: Coordinated Universal Time
WAN: Wide Area Network

Note: References in brackets [ ] and internet links in braces { } are provided in **section 10**.

1. Introduction

I previously produced papers discussing the importance and problems of proper time keeping in radio astronomy observations; for example, see {Reeve12-1}, {Reeve12-2} and {Reeve12-3}. As a follow-up, in early 2015, I investigated the Network Time Protocol (NTP) in greater detail and eventually installed NTP on all my PCs. At about this time I also produced an NTP time server called GpsNtp-Pi based on the Raspberry Pi platform and a GNSS receiver. The GpsNtp-Pi uses NTP and a pulse-per-second reference clock; see {ReeveGps}.

This paper updates and supplements the earlier papers. Here I review clock accuracy and synchronization, the Network Time Protocol, the Windows Time service and the Meinberg NTP Time Server Monitor software tool. These all are directly related to proper time-keeping and associated data time-stamping in an observatory PC. In addition, for reasons described later, the Network Time Protocol is the only method I recommend for PC time-keeping purposes.

2. Clock Accuracy and Synchronization

Clock performance is specified by its accuracy and stability. Accuracy is how well the clock matches a time reference (for example, Coordinated Universal Time, UTC) and stability is a measure of clock deviation and drift once it has been synchronized. Clocks can be broadly categorized based on their accuracy and stability (figure 1).

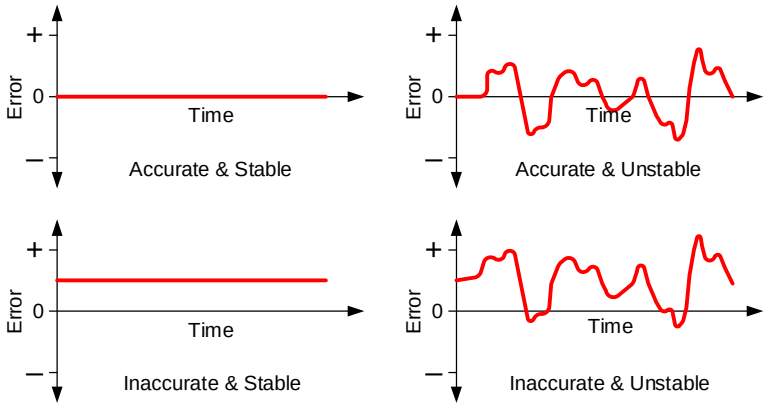


Figure 1 ~ Four basic categories of clock performance: accurate and stable, accurate and unstable, inaccurate and stable, and inaccurate and unstable. A PC clock can fall within any category at any given time. Image from {Reeve02}

Synchronization is the process a clock uses to obtain and set accurate time from a reference time source. Synchronization at most levels of detail is not simple. It requires protocols, algorithms, estimations and interfaces to negotiate with a reference clock, which usually is in another location. When a computer process is called to update the clock, it must know or at least estimate or assume when it sent the update request, the time required for the request to get to the reference, the time required for processing at the reference, the time required for the response to get back and the time required to process the response and set the PC clock. To

some extent a PC clock must depend on its own accuracy to set itself accurately. Small errors in assumptions and estimates accumulate to become larger errors. Before long the accumulated error can have a magnitude of seconds, minutes or hours.

A PC uses a quartz crystal oscillator that has good short term performance but is sensitive to temperature and drifts with age. Crystals used in PCs typically have 10 to 100 ppm tolerance when new so the associated clock may gain or lose 4 s/d or more before environmental effects are even considered (figure 2). A clock based on this technology must be regularly updated or resynchronized if useful accuracy is to be achieved.

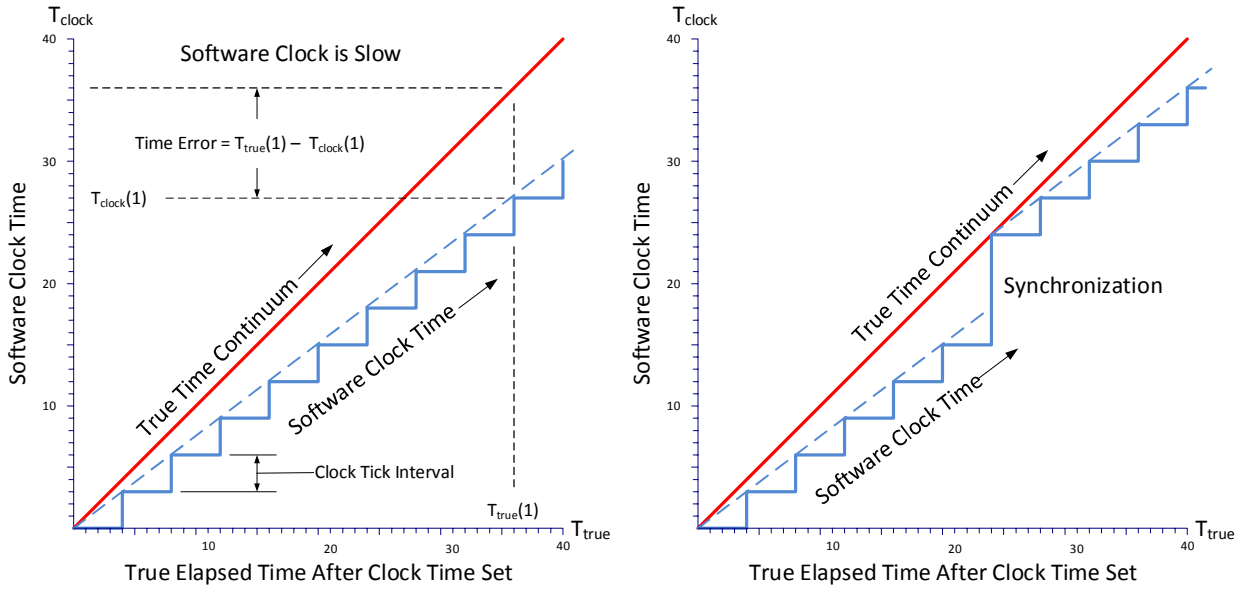


Figure 2 ~ Left: A stable software clock derived from a hardware oscillator with a frequency or period error will drift away from true time. The clock shown here is slow with respect to true time because the clock tick interval is slightly too small. Right: Synchronization takes place when the software clock is corrected to shift the clock time to true time. Since this clock has a slightly small tick interval, it will again drift away and need re-synchronization. The synchronization can occur at fixed intervals (say every 15 minutes) or be determined dynamically based on the clock error history. The synchronization process becomes much more complicated when the clock is unstable. Image © 2015, W. Reeve.

3. Network Time Protocol ~ NTP

The Network Time Protocol is a set of procedures used by computers and PCs to exchange time information and then to compensate for the error in a PC's real-time (or time-of-day) clock. NTP runs as a background process or daemon. The underlying software is called NTP daemon or just NTPD, but I will simply use NTP in the following discussions. The NTP software currently is version 4 (NTPv4). NTPv4 as well as tutorials and configuration information can be downloaded from the NTP website {NTPOrg}, and that information should be used to supplement this guide. The easiest way to install and configure NTP on a windows PC is to follow the installation guide that I prepared and is available here: [ReeveNTP].

NTP was developed as an internet protocol for synchronizing the clocks in distributed time servers and clients to a standard reference time (UTC). It is widely used for time distribution and to ensure time accuracy in local area

networks (LAN) and wide area networks (WAN). The National Institute of Standards and Technology (NIST) recommends NTP above all other time-keeping methods as stated in the following at {NIST}:

1. *We* (NIST) *will continue to support the "TIME" protocol that uses tcp port 37 for the forseeable [sic] future. However, this protocol is very expensive in terms of network bandwidth, since it uses the complete tcp machinery to transmit only 32 bits of data. Users are *strongly* encouraged to upgrade to the network time protocol (NTP), which is both more accurate and more robust.*
2. *Users of the NIST "DAYTIME" protocol on tcp port 13 are also strongly encouraged to upgrade to the network time protocol, which provides greater accuracy and requires less network bandwidth. The NIST time client (nistime-32bit.exe) supports both protocols.*

Note: The DAYTIME and TIME protocols mentioned above by NIST are described in Request for Comments RFC 867 and 868, respectively. An RFC is a document produced by the Internet Engineering Task Force (IETF) and the Internet Society, which are the technical development and standards bodies for the Internet. These RFCs may be easily found by internet search.

NTP is publicly documented and does not require special application software to run. Versions are available for Windows and Linux and several other operating systems. Most other time-keeping software applications that are freely available on the internet use proprietary code and few provide any useful description of how they work or even have a working help menu. My experience is that these other applications work most of the time but are inconsistent. I have never had any inconsistency or problem with NTP. Therefore, NTP is a known-good, well documented time-keeper and using it to control a PC time-of-day clock involves very little, if any, risk.

NTP exchanges messages between a time server and client. These are used to calculate time offsets and delays. NTP is capable of providing synchronization at the sub-tens of millisecond level in a wide area network. Thousands of NTP servers exist in the internet and many (but not all) of them are accessible by public users from their PC. Besides the servers that are part of the NTP project (discussed more later), a list of NIST public time servers in the USA can be found at {NIST}. This list also shows the real-time status of these servers and whether they are recommended for use at the time of viewing.

NTP can operate as a time client or server, and it always uses Coordinated Universal Time (UTC). The PC's time-of-day clock can be set to local time but it will use an offset from UTC. When used as a client on a PC, NTP synchronizes and regulates the PC clock by periodically obtaining time information from an external NTP server, either on the same LAN or over a WAN such as the internet. NTP also can use a *reference clock*, which is a clock that obtains accurate time information from an external source such as a GPS receiver or atomic clock. NTP cannot operate on a client PC unless it has at least one source of time information, either a reference clock or the URL or IP address of an NTP server.

In both server and client applications, NTP can be setup to use a pool of time servers, thus providing redundancy and guards against a server failure or a server being too busy to service time requests. The pool points to a random set of time servers that change at least every hour. When given a choice, NTP always will select the best performing server and reject a failed or poorly performing server (called "falseticker"). NTP can be setup to use a

preferred time server (*prefer* option in the NTP configuration file), which always will be used if it is found acceptable by NTP.

There may be considerable variability in the networks between NTP clients and servers, leading to variable delays in synchronization messages. NTP's job is to monitor these delays in real time and then estimate what actions are needed at the client to reduce the difference between the client PC's clock time and the NTP server's time. As the NTP client runs, it speeds up or slows down the PC's clock until synchronization is achieved. It compensates for clock instability by correcting the clock more or less often.

NTP has an ultimate precision of about 200 ps but PCs and most other computer systems cannot measure time to this resolution. Generally, after the PC's clock is set, NTP will maintain it within 128 ms even in the face of extreme network path congestion and jitter. Under normal network conditions, NTP's time keeping typically is an order of magnitude better. NTP slews the client's clock in very small steps, effectively providing a continuous timescale. The maximum slew rate under most conditions is limited to 500 parts per million. A simple calculation shows that the maximum slew rate is 2000 s for each 1 s error. The NTP has provisions for initial time bursts (*iburst* option in the NTP configuration file) to increase the polling rate when first activated so that a PC can be brought into synchronization faster after reboot or start. Also, NTP keeps a record of clock drift that is updated at regular intervals. If the PC is restarted after a shutdown, NTP uses the drift record to initially correct the clock, saving the time and trouble of relearning as the PC warms up.

To meet the goal of 128 ms error limit, the polling and correction interval by the PC's NTP client cannot be too long. For example, if the PC's clock drifts 250 ms/h (equivalent to 70 ppm or 6 s/d) and the NTP polling interval is 24 h, the goal never will be achieved. Windows PCs that have the built-in *Internet Time* feature enabled use a 7 day update interval by default (described in the next section) , much too long to be of any use in applications that time-stamp data. On the other hand, a polling interval that is too short usually serves no useful purpose and increases network load and potentially manifesting as clock instability.

While preparing this guide, I was surprised to discover that not all Windows installations have the Network Time Protocol pre-installed. Where NTP is not used and the user must specify an update interval in a time-keeping software tool, I have found that between 15 min and 2 h works best for most PCs. Generally, the shorter interval is used on PCs subject to considerable short-term temperature variations and longer interval for PCs in stable environments. It should be noted that PCs running many CPU-intensive processes throughout the day will experience higher internal temperature variations and thus larger clock variations even though they may be in a stable ambient environment. However, when NTP is run on a PC and it has access to a reference clock or time server, it will determine the best update interval based on what it learns about the PC's clock. This update interval is in in multiples of 2 such as ... 16 s, 32 s, ... 1024 s, and so on, to a maximum of 2^{17} s (36.4 h). I have noticed that NTP uses 1024 s (about 17 min) after the clocks have stabilized on all my technical PCs.

4. Microsoft Windows Default Time Environment

In the following discussion, I will concentrate on Windows XP and Windows 7. Most PCs running Windows XP and 7 use the Windows Time service by default (figure 3). In Windows a "service" is a piece of software that

performs a certain task and usually automatically starts or stops as needed. If the PC is connected to a local area network, the Windows Time service starts automatically.

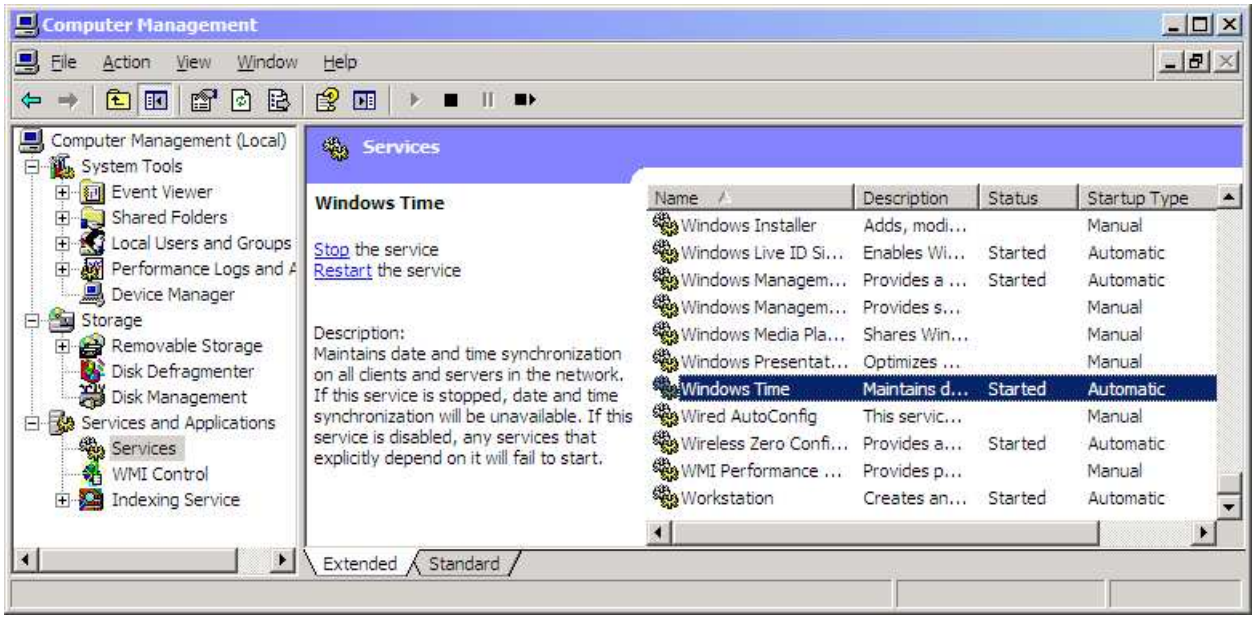


Figure 3 ~ Windows Time service is an underlying service on all modern Windows-based PCs. It automatically starts whenever the PC is connected to a network.

If the PC is connected to the internet, and Windows Time service is properly configured, it will attempt to contact a time server to set the clock accurately. However, if the PC is not part of a network or is not connected to the internet, the Windows Time service may not be started, in which case the user must figure out how to use it or to use some other method. Additional information on the Windows Time service can be found here: http://technet.microsoft.com/en-us/library/bb490605.aspx.

One of the problems with the default settings for the Windows time service is that the time server it is setup to use normally is very busy, resulting in failed time updates. Many public time servers are so busy that it becomes a waste of time trying to use them. To compound the problem, a time server that is not busy today or this morning may be busy later, which makes choosing a time server an ongoing but easily forgotten task. While it is possible to change the time servers used by the Windows Time service, a better solution is to use your own NTP time server such as the GpsNtp-Pi {GpsNtp-Pi} and then run NTP and the Meinberg NTP Timer Server Monitor on the client PCs.

5. Overview of Meinberg NTP Time Server Monitor

The Network Time Protocol does not have a familiar windows-type interface for user management. However, as a Windows service, it is accessible through the service management facilities built into all modern Windows operating systems but user actions are limited to starting and stopping the service (figure 4). NTP also may be controlled using a command line interface (figure 5), which provides more flexibility in terms of access to operating statistics and configuration. On the other hand, the Meinberg NTP Time Server Monitor software

briefly described in this section provides a familiar Windows graphical user interface for setting up and using NTP.

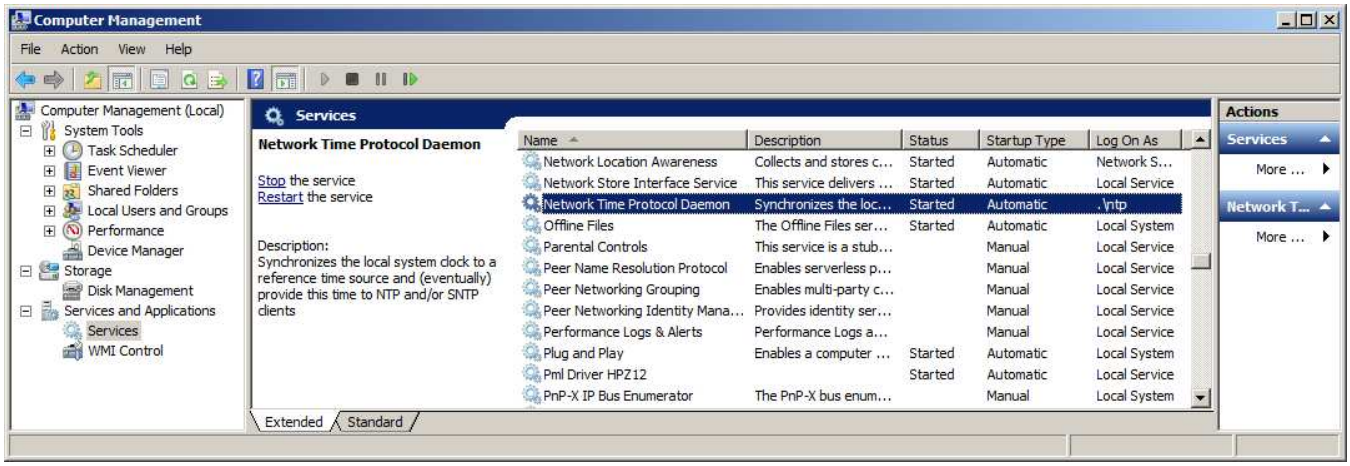


Figure 4 ~ As a Windows service, NTP can be started and stopped though the Computer Management window, but there are no other controls within easy reach of the service interface.

The Meinberg NTP Time Server Monitor allows the user to easily control, monitor and analyze NTP actions through a tabular Windows interface. For a step-by-step installation guide covering the Meinberg NTP Time Server Monitor and the Network Time Protocol see [ReeveNTP]. For a much more detailed description of the monitor than is provided in this paper see [ReeveMNTSM]. It should be noted that the Network Time Protocol and the Meinberg NTP Timer Server Monitor are two separate programs. NTP does not require the Meinberg monitor to run, but the Meinberg monitor requires NTP to run.

The monitor always opens to the NTP Service tab (figure 6), which provides basic Service Information and controls. The NTP service can be started, stopped and restarted using the buttons in the Service Configuration frame on the right side of the window. These are equivalent to the NTP controls in the Windows services management console previously described. NTP normally is setup to start automatically whenever the PC is started or rebooted, and the start, stop and restart buttons usually are used only after configuration changes.

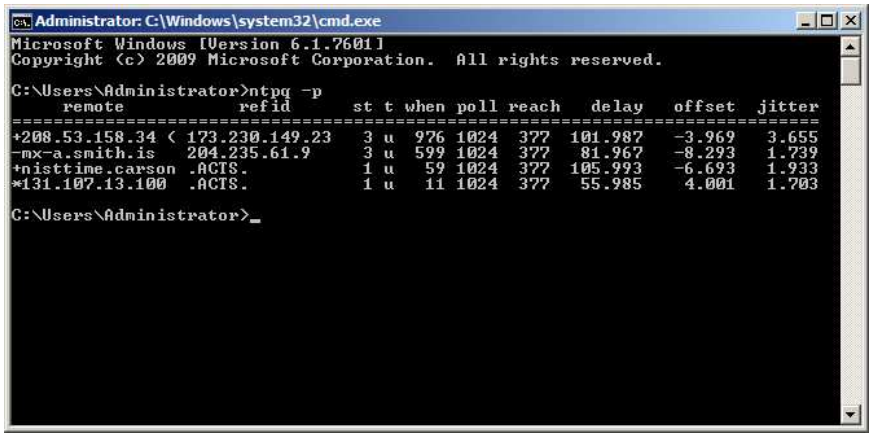


Figure 5 ~ The NTP Service can be accessed through the Windows command line interface to send commands and read statistics. The various commands can be found at {NTPOrg}.

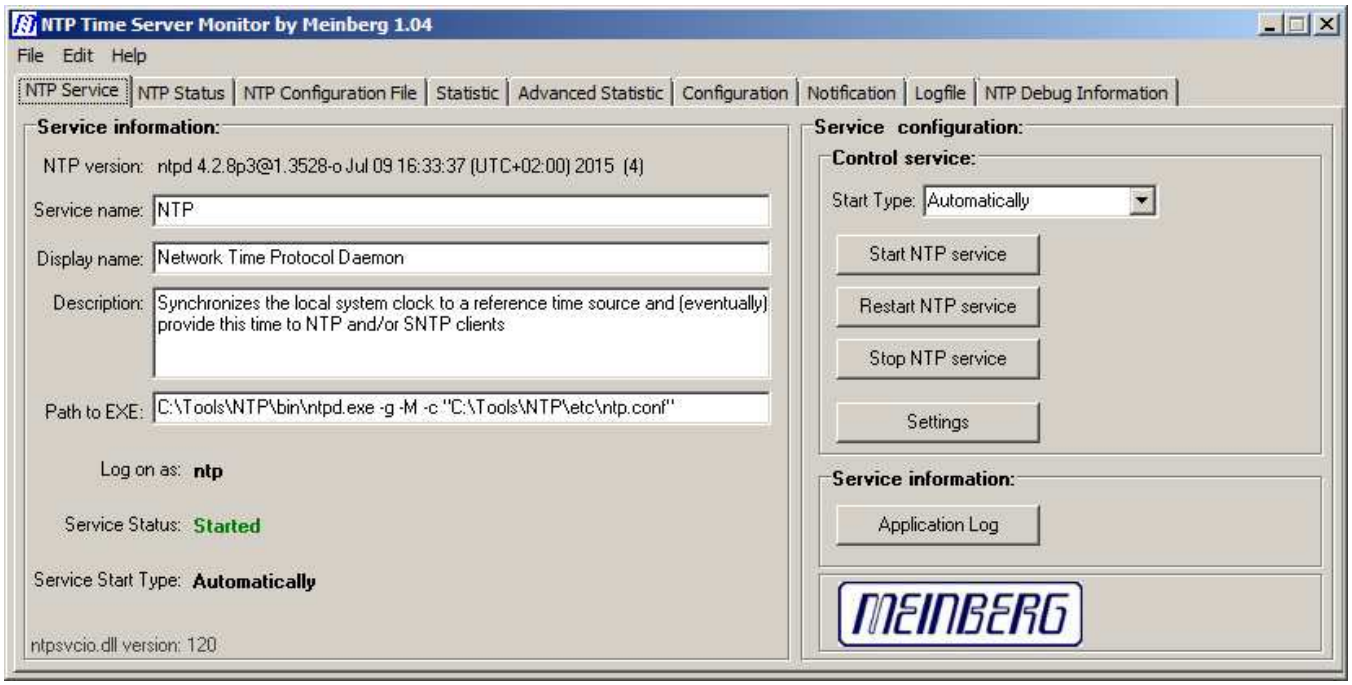
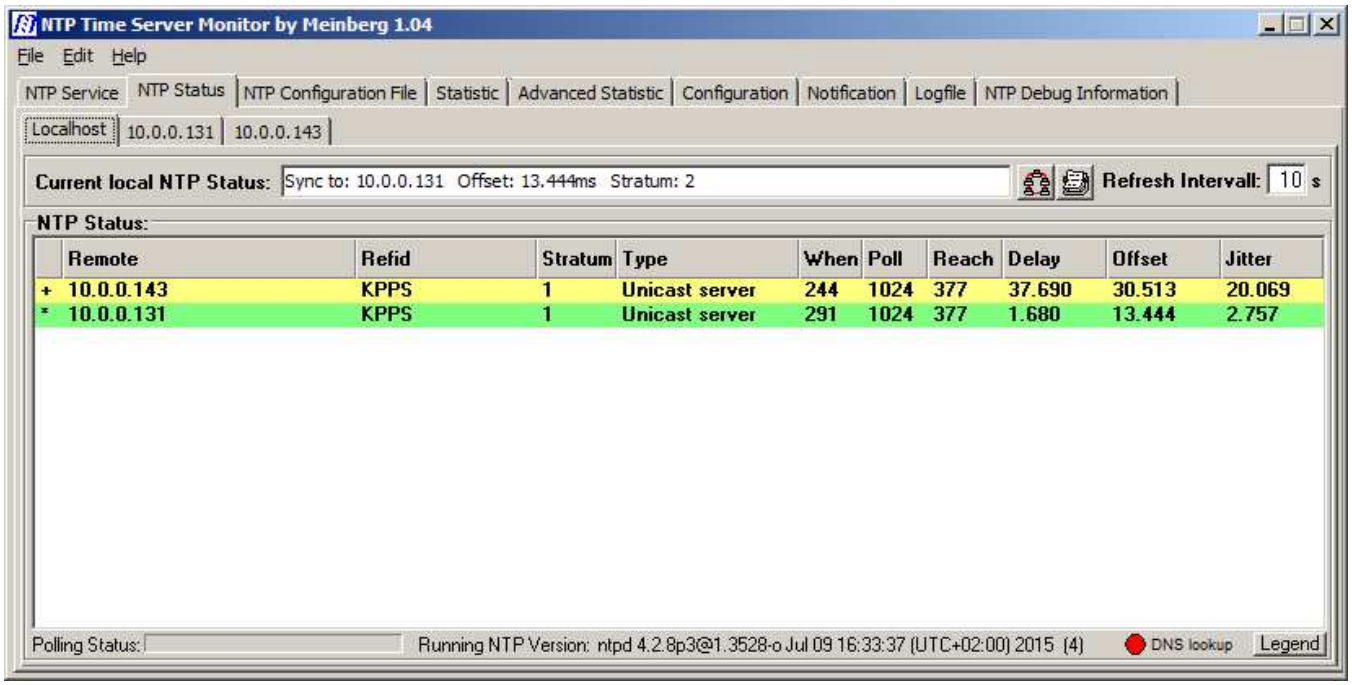


Figure 6 ~ The NTP Service tab provides detailed information about version, path and Service Status as well as control buttons for starting, restarting (stop and start combined) and starting the NTP service. NTP usually is setup to load and start automatically so that user intervention is not required when a PC reboots or is started.

Another important tab is the NTP Status tab, which shows the status of all time servers that have been configured in the PC's NTP service (figure 7). When NTP is started, it queries the configured servers, learns the environment and slowly adjusts the PC clock. NTP learns about the PC clock average drift rate and stores and periodically updates the information in a "driftfile". NTP uses this file to quickly synchronize the PC clock after a restart. NTP can be setup to use local time servers, a pool of servers or combination. Various time server pools can be configured such as by country (for example United States), continent (for example, North America), or custom. There are almost 4 000 pooled time servers (the majority in Europe); more details may be found at {NTPPool}.

Basic statistics of the NTP messaging loop between the NTP client and time server can be viewed in the Meinberg monitor Statistics tab. These statistics show how the local clock is being disciplined by NTP (figure 8). The Advanced Statistics tab has sub-tabs for Selected Peer, Stratum value, Delay and Polling interval (figure 9). Only Delay and Polling interval are of interest for most setups. They show the roundtrip messaging delay in ms between the time server and client and the interval that is used by NTP for polling a time server or reference clock, respectively.



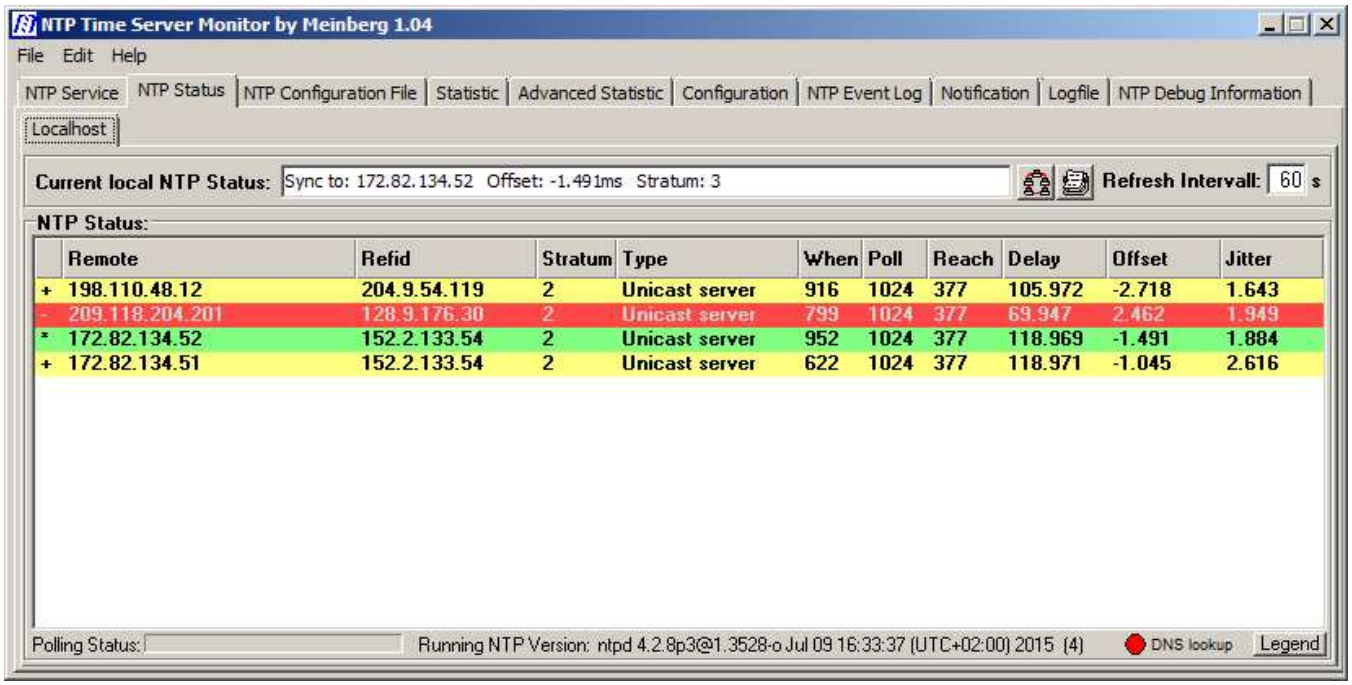


Figure 7 ~ Top: Two local GpsNtp-Pi time servers (IP addresses 10.0.0.131 and 10.0.0.143) have been setup on the PC. Only one server is selected at any given time and it is the one with the best performance as determined by NTP. The status of the selected server is shown in the field *Current local NTP Status*. In this case, the PC time-of-day clock is 13.444 ms faster than UTC. The refresh interval for the displayed data is 10 s and can be changed by the user. Bottom: Four servers from the United States NTP pool have been setup. The "*" character on the far left (and green shading) indicates the selected time server and the "+" character (and yellow shading) indicates a server acceptable for synchronization. The "-" (and red shading) indicates a server that has been found to be unacceptable for synchronization or that will not be used. The symbols are standardized and fixed but the colors may be changed by the user in the Configuration tab. The refresh interval for the displayed data in this example is 60 s.

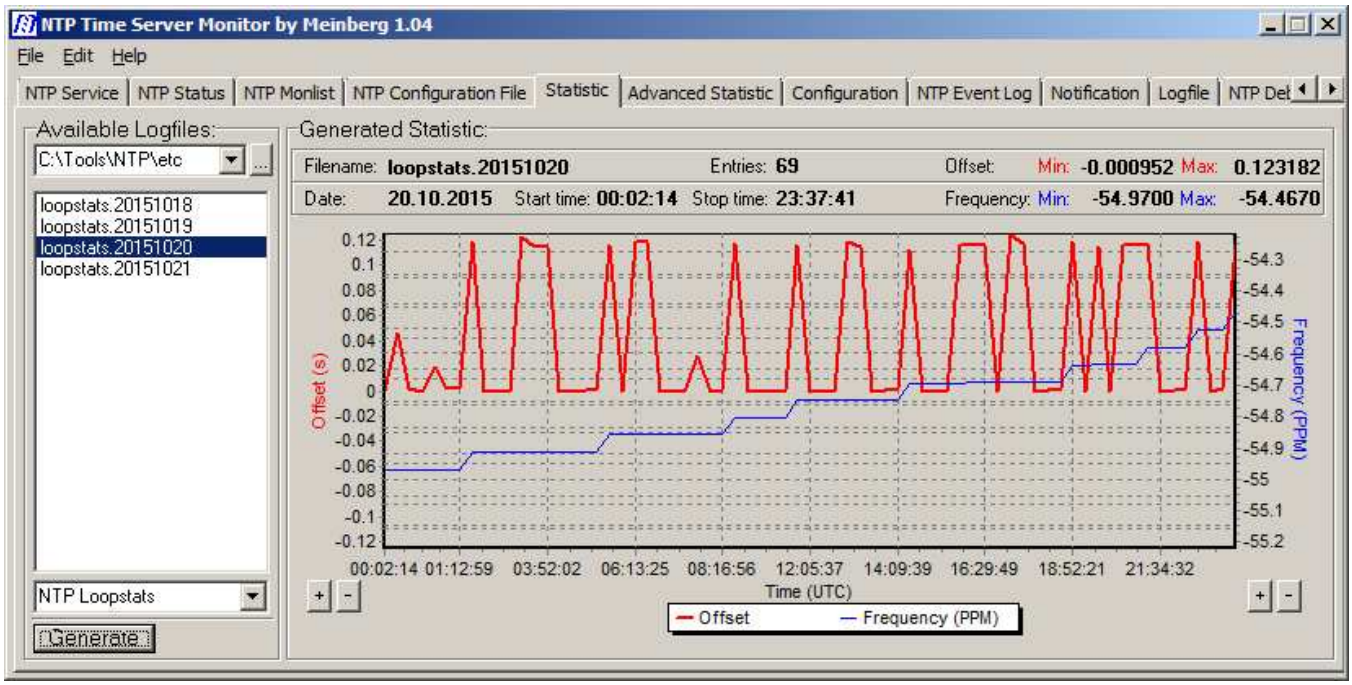


Figure 8 ~ The Statistic tab shows 24-hour plots of the clock offset adjustments (red trace) and frequency offset (blue trace). It is seen in this example that NTP is slowly stepping the frequency while the clock offset adjustments change more rapidly. The plots look different each day. Autoscaling is used on both vertical axes.

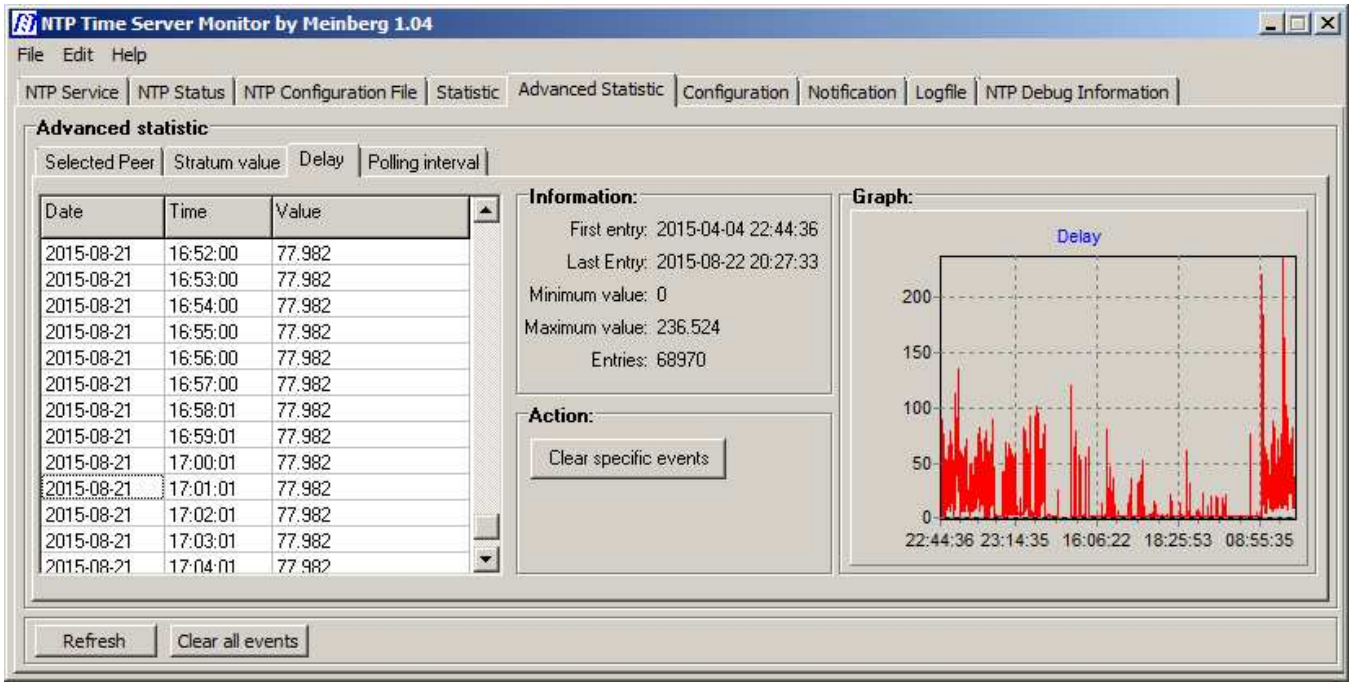


Figure 9 ~ The Advanced Statistics tab provides plots of the Selected Peer, Stratum value, Delay and Polling Interval over the most recent 24 h period. Generally, only the Delay and Polling Interval are of interest. They indicate the measured time characteristics s of the NTP messages that are exchanged between an NTP server and client. Note the Delay plot on the right, which in this example shows variability ranging from near 0 to over 250 ms presumably caused by buffering in the LAN router and the variable latency in the Wi-Fi connection. NTP compensates for this variability.

A PC that synchronizes via a wireless LAN usually shows more variation (jitter) than a PC on a wired LAN. Also, the jitter characteristics of a local time server such as the GpsNtp-Pi usually are more stable than, say, servers accessed through the public internet. The statistics for several identical PCs on the same LAN and with the same type of connection should be roughly comparable but will vary due to each PC's clock characteristics and environment.

7. Measurements

I made some measurements with three time server configurations (figure 10). One configuration used only local NTP time servers (two GpsNtp-Pi time servers), one used only a pool of servers in the United States and another used a combination of local and pool servers. I ran each setup on each of three PCs for 72 h. As mentioned above, when NTP has a choice it always will use the server with the best time characteristics. In the setup where one of the local GpsNtp-Pi servers was connected via a wired network and the other was connected via a wireless (Wi-Fi) network, NTP always used the local wired GpsNtp-Pi server. Also, when a combination of local and pool servers was available, NTP always chose the local wired GpsNtp-Pi server. Nevertheless, the clock and frequency offset statistics were similar for all setups (table 1). For time-stamping purposes, it is important to note the PC clock offset. In the three PCs analyzed here, NTP kept the clock within 10 ms of UTC, an order of magnitude better than NTP's goal of 128 ms.

8. Additional Comments and Leap Second

PCs used to produce time-stamped observations should be set to UTC and not local time, thus eliminating the possible 1 h time error twice a year at most locations. If it is desired to have local time readily available, observatory operators can use the TinyBen clock {TinyBen}. It can be placed anywhere on the Desktop and set to display both analog and digital formats with UTC and local time. The image right (actual size) shows analog local Alaska Standard Time and digital UTC. Installation of TinyBen requires only that it be copied to a convenient location on a user's PC and run from that location. The TinyBen's job is to simply read the PC's clock, apply the offsets specified in its configuration and display the results.

When a PC clock is set to UTC as recommended and local time is to be displayed on the TinyBen analog clock, set it as follows: Right-click the clock-face and click on UTC in the menu and then select Analog. Right-click again and click on UTC Offset and set it to your time zone (for example, Alaska Standard Time is –9 and Eastern Standard Time is -5). To also display UTC on the digital overlay, check Digital Overlay in the menu. Only TinyBen versions 1.2.1 and later support this setup.

Integral to Coordinated Universal Time is the addition or deletion of a leap second to adjust for irregularities in Earth's rotation with respect to UTC so that it remains within ±0.9 seconds of mean solar time (UT1). See {Reeve12-2} for a more complete discussion of the leap second. As of this document update (January 2016), a leap second was added most recently on 30 June 2015 at 23:59:60 UTC. The use of leap seconds to adjust UTC will continue to at least 2023 according to a decision made in November 2015 at the ITU World Radiocommunication Conference (WRC-15) in Geneva, Switzerland.

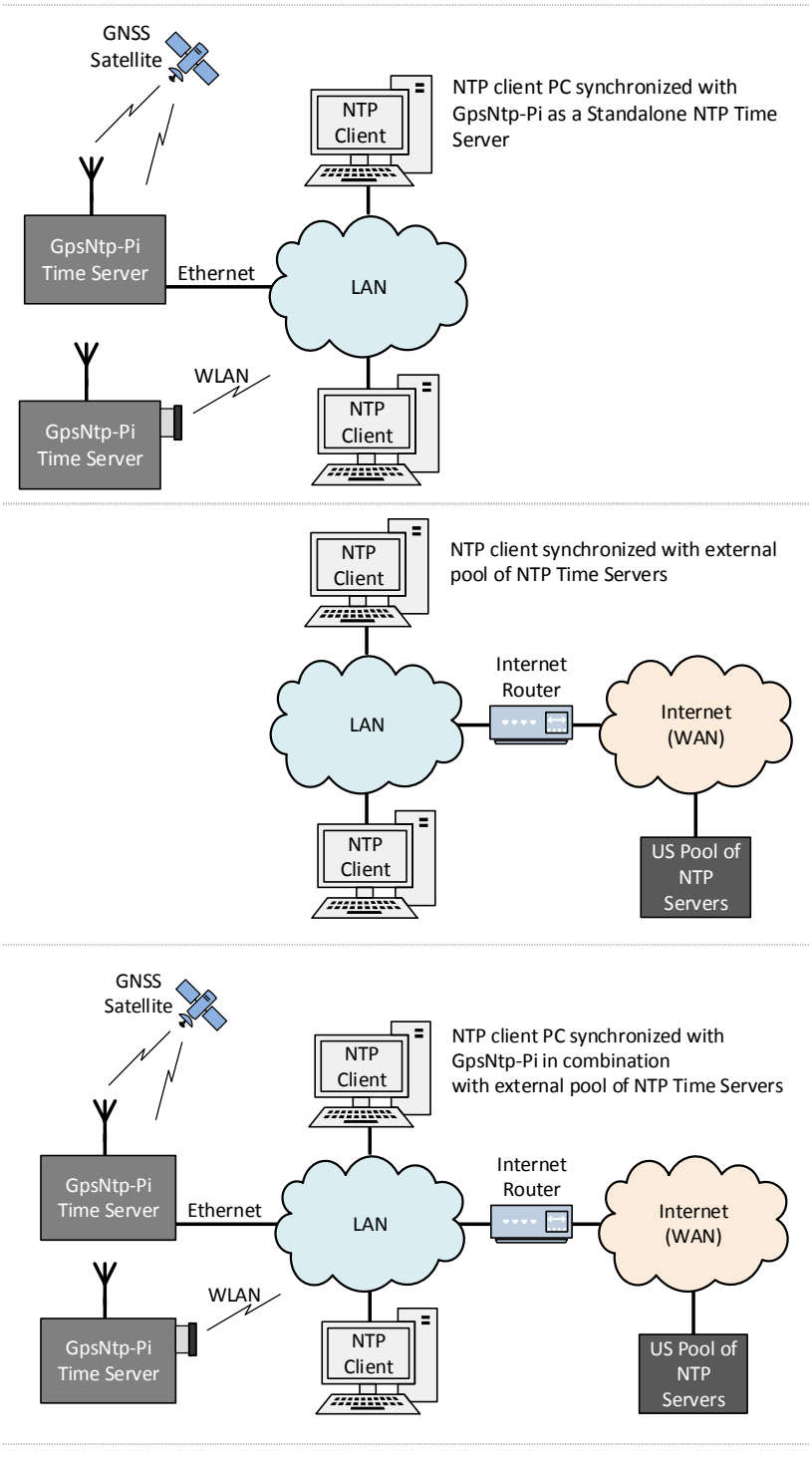


Figure 10 ~ Configurations for time server measurements.

Upper: GpsNtp-Pi as standalone time server

Middle: United States server pool only

Lower: GpsNtp-Pi in combination with a US server pool

Table 1 ~ Summary of minimum and maximum clock adjustments offsets and frequency offsets for three PCs using various NTP time servers. The clock offsets were marginally better when the GpsNtp-Pi time server was used. When the combination of GpsNtp-Pi and server pool was used, NTP always selected the GpsNtp-Pi as the best server; therefore, three clock offset averages are calculated from the data, one for all measurements (sections 1, 2 and 3), another where the GpsNtp-Pi was involved (sections 1 and 3) and where on the US pool was used (section 2).

Client PC	Clock offset Min. (ms)	Clock offset Max. (ms)	Pk-Pk Clock Adj. (ms)	Frequency offset Min. (Hz)	Frequency offset Max. (Hz)
1. GpsNtp-Pi only					
W7 A61e (Lab)	–3.337	1.526	4.863	19.8750	19.8850
W7 A61e (Rx)	–3.968	0.943	4.911	19.9930	20.0210
WXP SG41 (Lab)	–2.588	1.163	3.751	–38.3140	–37.9910
2. US Pool only					
W7 A61e (Lab)	–6.684	11.399	18.083	19.8880	19.9060
W7 A61e (Rx)	–4.236	1.720	5.956	19.9320	19.9510
WXP SG41 (Lab)	–1.337	3.158	4.495	–38.0510	–38.0350
3. Combination					
W7 A61e (Lab)	–1.374	3.411	4.785	19.8010	19.8210
W7 A61e (Rx)	–2.675	2.469	5.144	19.9450	19.9590
WXP SG41 (Lab)	–4.611	2.991	7.602	–38.3250	–38.1830
Average all (1,2,3)	–3.423	3.198	6.621		
Average GpsNtp-Pi (1,3)	–3.092	2.084	5.176		
Average pool (2)	–4.086	5.426	9.511		

9. Conclusions

The Network Time Protocol is recommended for use on all PCs and especially those PCs used to time-stamp radio observational data. The Network Time Protocol and the Meinberg NTP Time Server Monitor enable a user to set and maintain a PC clock within a few ms of UTC under most network conditions and within 128 ms under worst network conditions using either a local or internet time server.

10. References and Web Links

{Reeve02} Reeve, W., Telecommunications Synchronization Overview, August 2002, available here: http://www.reeve.com/Documents/Synchronization.pdf

{Reeve12-1} Reeve, W., Maintain Your Time, 2012, available here: http://www.reeve.com/Documents/Articles%20Papers/MaintainTime_Reeve.pdf

{Reeve12-2} Reeve, W., Is Time Broken (or, Will It Be Y2K All Over Again)?, 2012, available here: http://www.reeve.com/Documents/Articles%20Papers/IsTimeBroken_Reeve.pdf

{Reeve12-3} Reeve, W., Time Differences in Charted Solar Observations at High Frequencies, 2012, available here: http://www.reeve.com/Documents/Articles%20Papers/TimeDifference_Reeve.pdf

{ReeveGps} Reeve, W., GpsNtp-Pi: GPS Network Time Server on Raspberry Pi, 2015, available here: http://www.reeve.com/Documents/Articles%20Papers/Reeve_GpsNtp-Pi.pdf

{ReeveMNTSM} Reeve, W., Meinberg NTP Time Server Monitor Guide, 2015, available here: http://www.reeve.com/Documents/Articles%20Papers/Reeve_MeinbergMonGuide.pdf

{ReeveNTP} Reeve, W., Network Time Protocol and Meinberg NTP Time Server Monitor ~ Installation Guide, 2015, available here: http://www.reeve.com/Documents/Articles%20Papers/Reeve_NTP-MeinMon_Install.pdf

{GpsNtp-Pi} http://www.reeve.com/RadioScience/Raspberry%20Pi/GpsNtp-Pi.htm
{NIST} http://tf.nist.gov/tf-cgi/servers.cgi
{NTPOrg} http://www.ntp.org/
{NTPPlot} http://www.satsignal.eu/software/net.htm
{NTPPool} http://www.pool.ntp.org/en/
{TinyBen} http://www.satsignal.eu/software/disk.html#TinyBen

Document Information

Author: Whitham D. Reeve, Anchorage, Alaska USA

Revision history: Iss. 0.0 (Initial draft started based on original paper, 4 Apr 2015)
0.1 (Major additions to NTP and Meinberg, 18 Oct 2015)
0.2 (Split from NTP and Meinberg monitor installation doc, 20 Oct 2015)
0.3 (Add'l analyses, 23 Oct 2015)
0.4 (Cleanup, 28 Oct 2015)
0.5 (Added DNS option and final images, 2 Nov 2015)
1.0 (Distribution, 2 Nov 2015)
1.1 (Added basic operation, 7 Nov 2015)
1.2 (Prepare for split into separate parts, 15 Nov 2015)
1.3 (Split, 16 Nov 2015)
1.4 (Cleanup, 19 Nov 2015)
2.0 (Redistribution after split, 24 Nov 2015)
2.1 (Minor updates, added Leap Second, 14 Jan 2016)
2.2 (Final cleanup, 2 Feb 2016)
2.3 (Clarified table 1, added fig. 10, 19 Feb 2016)

Total word count: 4427
File size: 2677248B