

CESNET technical report 18/2004

Experience with Precise Timekeeping in End-hosts

Vladimír Smotlacha
vs@cesnet.cz
CESNET, Prague, Czech Republic

December 10, 2004

1 Introduction

This report demonstrates issues of highly precise timekeeping in a Linux workstation using external source of time. It describes the kernel patch, NTP installation + configuration and the connection of a GPS receiver. The aim of an precise timekeeping is to provide internal events by an accurate timestamp. In principle, it is similar to setup of primary NTP server but the document does not deal with network time synchronization and time distribution via the network (e.g. an authentication, a logging, peer to peer communication between servers).

2 Source of external clock

There is a lot of physical clocks that are able to discipline the computer clock. Most of them are supported by drivers in NTP package. It includes both proprietary products and general standards of communications:

- atom clocks,
- GPS receivers,
- receivers of other time information (DCF, VWW, ACTS, ...).

Clocks can be categorized also according to the way how the signal is delivered to the computer:

- a card is installed in the computer, driver reads time from its registers,
- external clock output is connected to a standard periphery (serial, parallel, audio) using a communication protocol.

We have got practical experience with following clocks:

- cesium clock providing PPS (Pulse Per Second, e.g. the active edge of the pulse marks the beginning of second) and label of second,
- GPS receivers providing PPS and label of second (e.g. Motorola ONCORE, Garmin GPS 35 and GPS 18, Trimble ACUTIME 2000). ACUTIME 2000 has also a special protocol based on request and response,
- GPS receiver Mainberg GPS169PCI. It consists of an external antenna with a frequency converter and a special embedded card,
- DCF77 receiver. Cheap but less accurate AM receiver of DFC77 signal.

Label of second is usually encoded in a serial line protocol. Some of these protocols are proprietary but there exists also a general protocol NMEA. All of GPS receivers mentioned in previous paragraph (except of the Mainberg) support NMEA, too. There is a practical advantage, we can have one general NTP configuration for all of these GPS receivers.

As the antenna of GPS receiver is usually installed on the roof, cable requirements might be important and should be taken into account.

Garmin GPS 35 and GPS 18 - the receiver has RS-232 serial interface but PPS signal is provided in TTL level. Input circuits compatible with TIA-232-F are able to process such PPS signal directly. When the cable is longer than about 25m, conversion to/from RS-422 should be applied.

Trimble ACUTIME 2000 - the receiver has all signals in RS-422. Therefore, there is no strict cable length limitation but only a conversion to RS-232 at computer site is necessary.

Motorola ONCORE consists of a separate antenna and a receiver which are connected by a coaxial cable. High quality UHF cable has to be used as it conducts GPS signal (frequency 1.575 GHz and 1.227 GHz). Receiver has RS-232 interface.

Mainberg GPS169PCI consists of a separate antenna with frequency converter and an embedded PCI card. Both units are connected by a coaxial cable. Cheap RG-58 type (which is well known as the medium of 10Base-2 Ethernet) is good enough due to low frequency of the signal.

The NMEA protocol can be used unidirectionally as it is based on periodical sending of selected information. Only one-time configuration has to be sent to the GPS receiver. Therefore, the NMEA output can be distributed from one source to several destinations. PPS signal can be split, too. We use several distribution units to provide production NTP servers, network measurement workstation and laboratory test boxes by the time signal from our GPS receivers.

Another issue of the signal distribution is an utilization of standard CAT5 cabling to deliver both the serial signal and the PPS signal. Our practical experience is that about 80m of CAT5 cable does not cause any problem.

The picture 1 shows one of our distribution units. It has a GPS interface with physical level RS-422, as it is designed for direct connection to Trimble ACUTIME 2000. There is one BNC output of PPS signal and 8 outputs providing serial signal and PPS signal in level RS-232.

3 Software

This section describes the NTP software, its installation and configuration. We are going to aim at the NMEA driver (`recflock_nmea.c`) as it is supported by majority of GPS receivers. Alternatively, drivers for native protocols of Motorola or Trimble can be used, too.

3.1 Nanokernel

Standard Linux kernel (2.4.x) has to be extended by new features to support precise timekeeping. It includes following properties:

- time is internally evaluated in nanoseconds instead of microseconds,
- time resolution is equal to CPU clock (by the TSC register reading),
- driver for PPS (pulse per second) signal processing,
- PLL (phase-lock loop) implemented in kernel.

Time to time, new version of the **PPSkit** is released. The most important file of the kit is the kernel patch (called nanokernel patch). The PPSkit contains also useful README and INSTALL texts.

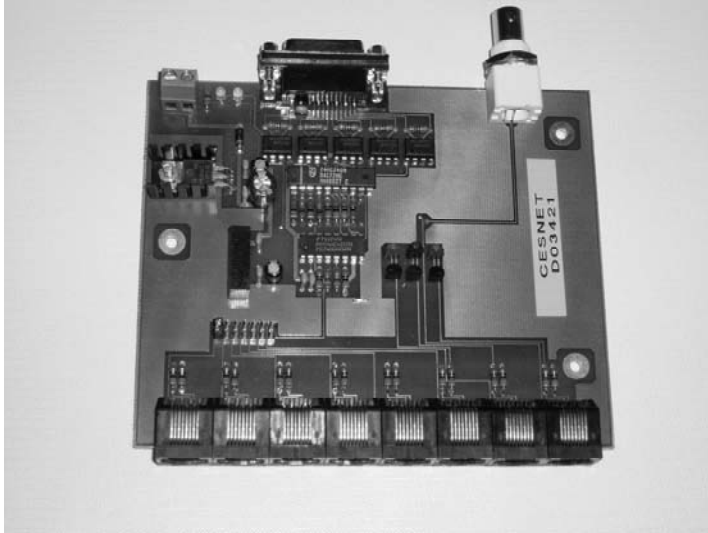


Figure 1: PPS distribution unit

The patch exists only for selected kernels as new versions of the kernel are released to rapidly. The latest PPSkit currently available (November 2004) is 2.1.4 with patch for kernel 2.4.21. PPSkit can be downloaded from <ftp://ftp.kernel.org/pub/linux/daemons/ntp/PPS>.

When nanokernel patch is applied, several new configuration options are available and they should be activated:

- enable *Development drivers* in *Code maturity level options*,
- enable *NTP kernel support* and *NTP PPS support* in the *Processor type* section,
- enable *NTP PPS support on serial port* in the *Character devices* section.

The kernel should be compiled and installed as usual. We can check useful installation in the boot log `/var/log/dmesg`. The attribute **NANO** has to appear in the first line containing kernel identification. Successful implementation of PPS support is indicated by an attribute **PPSAPI** in the line concerning Serial driver.

3.2 NTP package

The distribution of NTP contains *ntpd* daemon, a lot of utilities, and a documentation. There are also drivers for large variety of external clocks. The last available NTP version is 4.2.0. Source code of NTP package can be downloaded from <http://www.ntp.org>.

According to our experiences with Debian and RedHat, we strongly discourage anybody from using default kernel and *ntpd* (or even the old *xntpd*) of the Linux distribution for precise timekeeping as there is no support for advanced nanokernel features. Instead, the suggested method is to boot a nanokernel. Than NTP has to be configured, compiled from source and installed.

3.3 NTP installation

In order to activate nano feature and PPS support, new versions of header files *timepps.h* and *timex.h* created by patch has to be found by C compiler. Specially, new version of *timex.h* has to be used instead

of the old one in `/usr/include/sys`.

Following command is suggested to get access to new versions of header files:

```
for i in ppstime.h timex.h
do
  ln -s $LINUX/include/linux/$i /usr/include/sys/$i
done
```

where `$LINUX` is the path to the current kernel directory.

NTP should be compiled and installed by the standard sequence:

```
./configure
make
make install
```

According to the type of the external clock, new device file might be required to access the PPS source. In case of NMEA driver, the `gps0` and `gps1` has to be created:

```
ln -s /dev/ttyS0 /dev/gps0
ln -s /dev/ttyS1 /dev/gps1
```

3.4 NTP configuration

The default name and location of the configuration file is `/etc/ntp.conf`. When NMEA driver is selected, the magic lines are:

```
server 127.127.20.0
fudge 127.127.20.0 refid GPS flag2 1 flag3 1
```

Where 127.127.20.0 represents pseudo IP to activate the NMEA driver on the first serial line (`ttyS0`). For `ttyS1`, it has to be changed to 127.12.20.1. Default serial line format is 4800 baud, 8 bits, 1 stopbit, no parity. When the assert edge is the active edge of PPS signal, it is identified by value 0 (default) of `flag2`, while value 1 represents the clear edge. Value 1 of `flag3` says that PPS discipline of the kernel is enabled.

The line `server` can be used more time and can identify any clock driver or external real NTP server. However, for most precise timekeeping, we have to avoid external servers to eliminate the *selection algorithm*, which tries to find the best server. While the *selection algorithm* provides very robust solution for NTP server, it degrades the achievable accuracy of the internal clock, which is our primary goal.

Other row of the configuration file defines, where to find the file recording the precise clock frequency. Usually, it is:

```
driftfile /etc/ntp/drift
```

System generates several statistics files. It includes *loopstats* (record of loop filter information), *clockstats* (clock drivers statistics) and *peerstats* (record of peers activities).

Following fragment of configuration file activates all three statistics, defines file names and directory. Each file represents one month statistics.

```
statistics loopstats peerstats clockstats
statsdir /var/log/ntp/
filegen loopstats file ntp-loop type month enable
filegen peerstats file ntp-peer type month enable
filegen clockstats file ntp-clock type month enable
```

3.5 NTP utilities

The NTP package contains a lot of utilities to check, control and debug the *ntpd* process:

ntpdate is designed for one-time set of local clock according to the server time. As it represents a simple NTP client communicating on the same port as *ntpd* daemon, it can not be used when the *ntpd* process is active. It is often used (periodically invoked by cron) on systems without the *ntpd* process.

ntpq is a general purpose utility to check the *ntpd* process.

ntpdc is another general purpose utility to query the *ntpd* process and to set its parameters. Detailed description of both *ntpq* and *ntpdc* is beyond the scope of this text.

ntptrace is a simple utility to trace the synchronization path between local clock and primary NTP server.

ntptime presents returned values of functions *ntp_gettime()* and *ntp_adjtime()*. In principle, it displays actual status of local clock.

Example of *ntptime* output:

```
ntp_gettime() returns code 0 (OK)
  time c5683d4d.2592c8b8 Mon, Dec 13 2004 16:09:49.146, (.146771924),
  maximum error 10263 us, estimated error 5 us, TAI offset 0
ntp_adjtime() returns code 0 (OK)
  modes 0x0 (),
  offset 0.005 us, frequency -0.288 ppm, interval 256 s,
  maximum error 10263 us, estimated error 5 us,
  status 0x2107 (PLL,PPSFREQ,PPSTIME,PPSSIGNAL,NANO),
  time constant 6, precision 1.555 us, tolerance 496 ppm,
  pps frequency -0.288 ppm, stability 0.000 ppm, jitter 0.068 us,
  intervals 173893, jitter exceeded 184565, stability exceeded 707, errors 4.
```

The function *ntp_adjtime()* reports variables of both the *ntpd* process and kernel part of the time synchronization algorithm. If there is no kernel support for time synchronization, relevant variables (last 3 lines in previous example) are blank. Detailed interpretation of *ntptime* output is important to know the status of internal clock:

return code - 0 means that returned data all valid. Any other code value indicates that *ntpd* status is not reported correctly.

time - hexadecimal value represents current time in seconds since January 1, 1900.

maximal error of reported time is calculated according the clock model. The value is usually very high.

estimated error is evaluated according to the recent internal variables of the *ntpd* process. It gives much reasonable value than maximal error but real difference between accurate time and local clock time is usually several time lower.

offset is the last known difference between local time and time of master clock (e.g. GPS time). We verified that this reported value has very good correspondence with real offset between local clock and UTC time.

frequency is a difference between the real rate of local oscillator and its nominal value (i.e. 14.318MHz in PC). The unit **ppm** means 'parts per million', i.e. it is dimension free number.

interval is the length of the clock control loop.

status of the *ntpd* process introduces several flags:

NANO declares that time is internally evaluated in nanoseconds instead of microseconds. It is probably the only way how to get interpretation of function *gettimeofday()*, which return either **struct timeval** (in normal case, i.e. without NANO) or **struct timespec** (with NANO flag). When nanokernel is used but there is not set the NANO flag, we can conclude that the *ntpd* was not compiled correctly.

PPSSIGNAL announces the presence of PPS on input port.

PPSFREQ - PPS is used to control frequency of internal clock.

PPSTIME - PPS is used to control time of internal clock.

PLL - means Phase-Locked Loop and declares which quantity (phase or frequency) is controlled. Alternative flag is **FLL** - Frequency-Locked Loop.

time constant - polling interval (\log_2).

precision is a duration of *gettimeofday()* call. It represents the resolution of system clock.

tolerance is always fixed value 496 ppm representing maximal allowed difference between the real and the nominal oscillator frequency.

pps frequency is similar to item **frequency** but its is evaluated by the kernel part of the PPS driver.

stability is the weighted average of recent frequency changes.

jitter is calculated from last three captured PPS signal.

intervals - number of kernel PPS control loops.

jitter exceeded - number of high jitter events.

stability exceeded - number of intervals with bad stability of frequency.

errors - number of observed errors in kernel synchronization loop.

3.6 NTP logs

This paragraph describes the structure of statistics records. We should explain the item MJD (Modified Julian Day). It is an enumeration of days (expressed by last 5 digits), which is widely used in physics and astronomy. It is worth to remember that MJD of January 1, 1900 (time 0 in NTP protocol) is 15020, MJD of January 1, 1970 (time 0 in Unix universe) is 40587 and MJD of January 1, 2005 is 53371.

loopstats

1. MJD (Modified Julian Day),
2. time past UTC midnight [s],
3. time offset [s],
4. frequency offset [ppm],
5. jitter [s],
6. Allan deviation [ppm],
7. clock time constant [\log_2 of seconds].

peerstats

1. MJD,
2. time past UTC midnight [s],
3. peer IP address (driver of external clock has assigned also address in the form 127.127.x.x),
4. status [hex number],
5. time offset [s],
6. delay [s],
7. jitter [s].

clockstats

1. MJD,
2. time past UTC midnight [s],
3. clock IP address,
4. timecode received from the clock (format is clock dependent).

4 Accuracy, resolution and precision

There are models to describe various types of clock, however these theoretical issues are out of scope of this article. Let us define three quantitative parameters, that are often misunderstood:

accuracy is the absolute difference between the local time and some reference time (usually UTC),

resolution is the smallest step by which the clock is updated (i.e. the smallest increment of time),

precision is granularity by which the time is represented.

For instance, time in NTP protocol is coded by fixed point 64 bits word, where 32 bits represent the fraction of second. Therefore, the precision is 2^{-32} s, i.e. about 230 ps. In nanokernel, the source of time is a counter of CPU ticks (TSC register of Pentium). Therefore, the resolution is 1 ns for Pentium with clock rate 1 GHz.

Accuracy is affected by several phenomena. Let us discuss an example of the Pentium workstation with clock disciplined by GPS via PPS signal. In this case the phenomena include:

GPS - inaccuracy of provided PPS signal,

cable - delay of the signal on the cable,

input circuit - the decision level of edge of the PPS signal might vary,

interrupt latency - timestamp of PPS signal edge is evaluated in an interrupt routine.

Among these phenomena, the most important is the interrupt latency as it is higher than all others, and it varies.

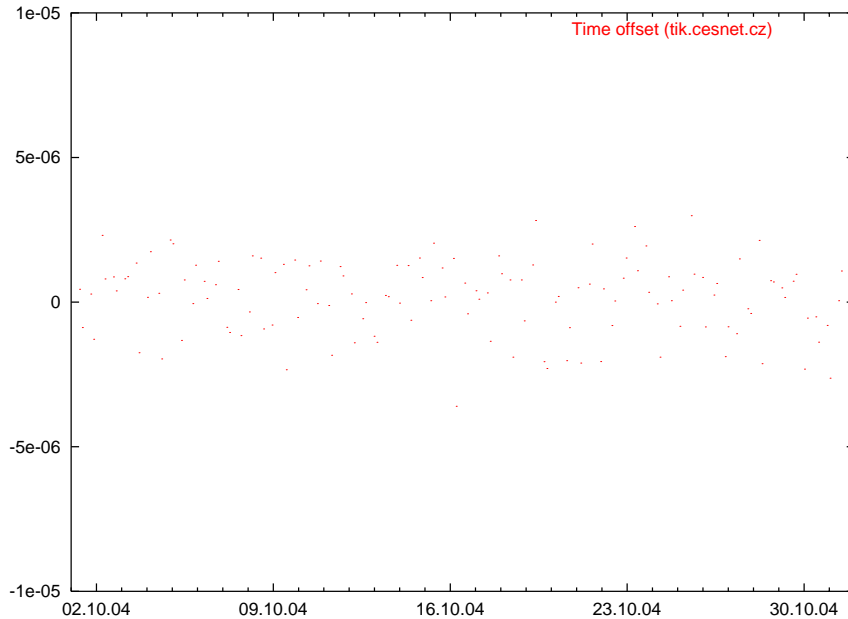


Figure 2: NTP server *tik.cesnet.cz*

4.1 Accuracy of real workstations

We did many measurements of accuracy and we operate several primary NTP servers. Following graphs display the time offset (i.e. accuracy) of these workstations as it was recorded in October 2004.

Figure 2 shows the time offset of our primary NTP server *tik.cesnet.cz*. It is an old no-name Pentium 1 (150 MHz) computer, where the original quartz was replaced by a top quartz with lower temperature dependence. The workstation is operated in air-conditioned server room. It is disciplined by the receiver Garmin GPS 35.

Figure 3 shows the time offset of another NTP primary server, *tak.cesnet.cz*. It is DELL 1400 with 860 MHz Pentium-III. The workstation is disciplined by Trimble ACUTIME 2000. It is operated in the same computer room as *tik.cesnet.cz* but the temperature dependence of its original quartz is probably worse.

Figure 4 represents our most accurate NTP server based on no-name Celeron 1 GHz computer. It is disciplined by atom clock, the Czech national standard of time. Original quartz has been replaced by a TCX oscillator and PPS signal is captured by a special PCI card which eliminates the interrupt delay and the latency down to 50 ns. Figure 4 proves that the offset of internal time is far below 1 μ s.

Figure 5 gives an example of workstation unsuitable for precise timekeeping. The time offset is about 100 times higher than the offset of previous NTP servers. This bad behaving workstation was DELL 1850 with Xeon CPU. Origin of the problem is probably similar to that one described in next paragraph.

Figures 6 and 7 show the influence of bios to the precise timekeeping. Both figures present the same data with different scale of y-axis. The hardware is DELL PE2600, where we replaced the original bios (left part of figure) by a special bios version called NORTC (right part), which was kindly provided by the Czech DELL representation. The problem was identified as a bios-level support for workstation management which invokes some non-standard interrupts.

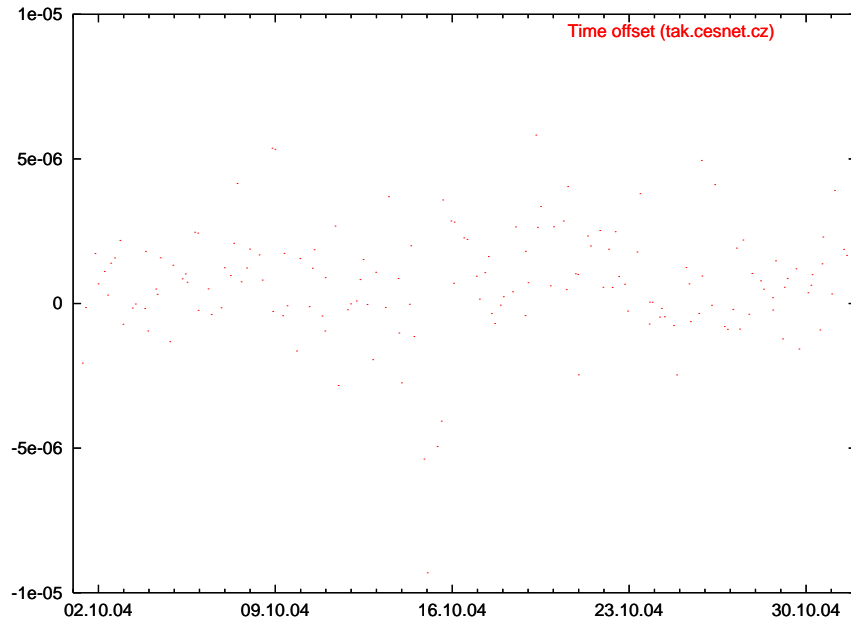


Figure 3: NTP server tak.cesnet.cz

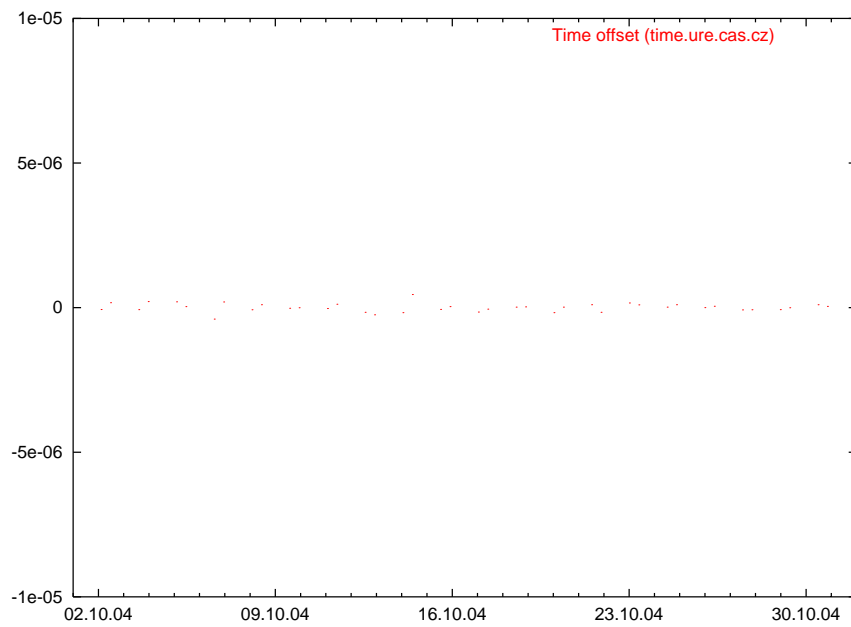


Figure 4: NTP server time.ure.cz

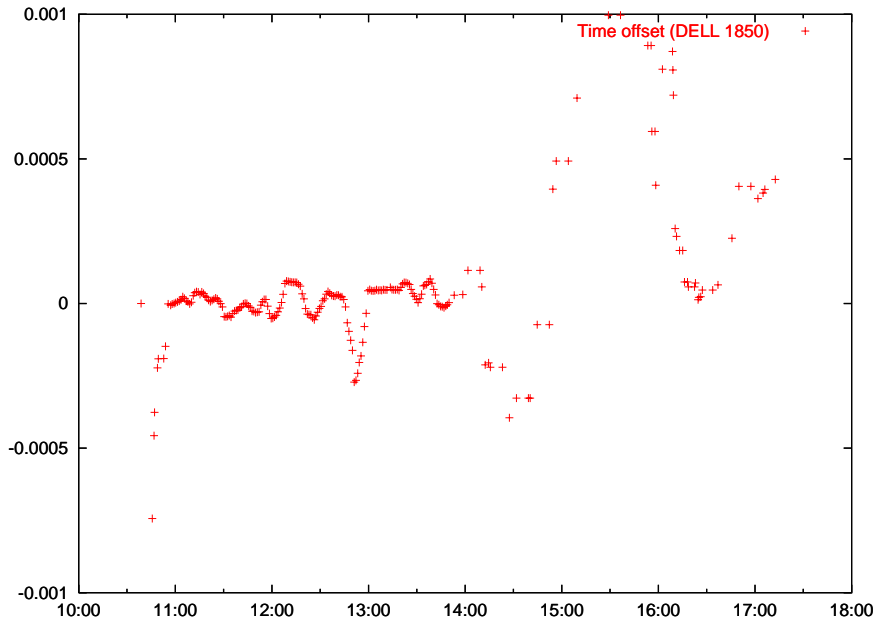


Figure 5: DELL 1850

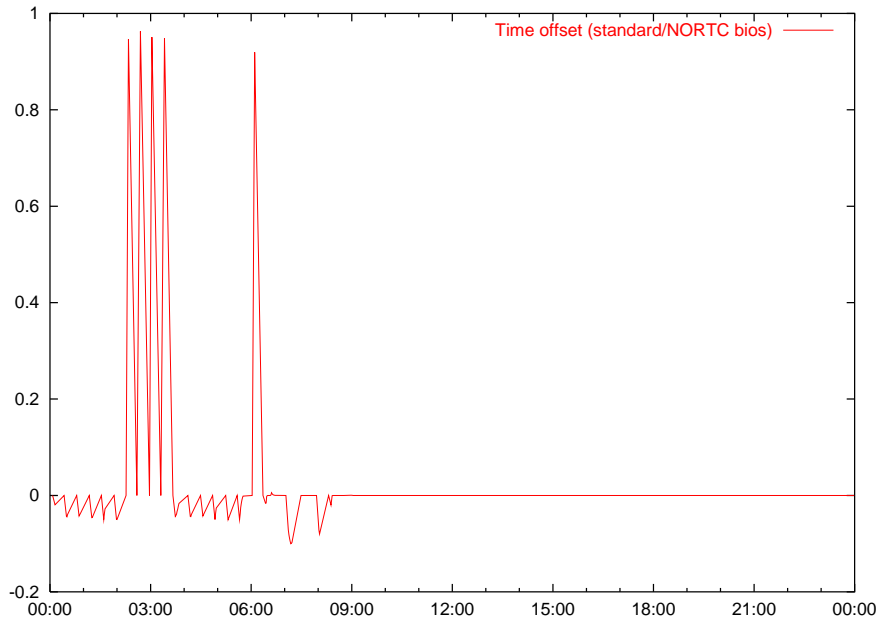


Figure 6: DELL 2600 - NORTC bios

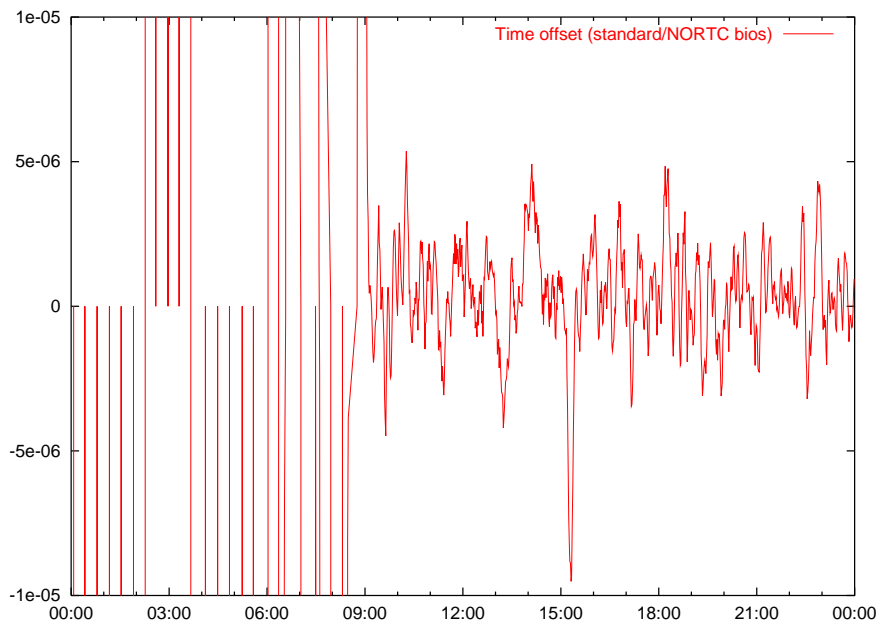


Figure 7: DELL 2600 - NORTC bios (detail)