GPS-based timing system for LAMOST

Xinqi Xu[1], Gangping Jin

Nanjing Institute of Astronomical Optics & Technology

National Astronomical Observatories, CAS

188 Bancang Street, Nanjing 210042, P.R.China

ABSTRACT

The control system of LAMOST telescope is highly distributed real time system, and the time base is crucial. During the motion tracking for 4000 celestial objects being simultaneously observed the alt-azimuth mount has to be driven on two axes in a servo loop to follow the motion of the objects in a timing system that has to be precise to the level of a few milliseconds. The GPS-based timing system has been developed in the lab of Nanjing Institute of Astronomical Optics & Technology (NIAOT). This paper describes a Net Time Server (NTS) that maintains the Coordinated Universal Time (UTC) derived from GPS, and distributes the time to precisely synchronize the client computer clocks across a network. The NTS is built on real time OS QNX4.25 platform. With help of a GPS receiver at hand, the NTS reaches the precision of 0.1 millisecond, and the time precision across LAN computers served by the NTS can meet the requirements for different time critical tasks in LAMOST control.

Keywords: GPS, real time, time system, UTC, NTS

1. INTRODUCTION

The GPS-based timing technique has been brought into use since the nineties of last century. The utilization of GPS-based timing system has been proved a big success for a number of contemporary large astronomical telescopes. However, China as a developing country lacks the experience in building large astronomical optical telescopes. The previously largest one made ever in China was 2.16-meter equatorial telescope, which came with no GPS at all in its timing system. As a China's ambitious new project, the Large Sky Area Multi-objects Fiber Spectroscopic Telescope (LAMOST) enjoys its fame for the telescope's unique optical configuration, which features the largest ever field of view of 5^0 in all world's existent meter-class level ground astronomical optical telescopes. It also calls for a sophisticate control system to meet all tough requirements for the distributed and real time control. The utilization of GPS-based timing system in LAMOST control is the first time step in China for large ground astronomical optical telescopes.

Four years ago, at the time of the preliminary design of LAMOST control system in 1998 the choice of GPS-based timing system has been presented for the reason of a number of merits. It was not until 2000 that a GPS receiver was really used in our site test for a model LAMOST tracking. The receiver could normally catch 6~7 satellites at one time and give off time tick for the mount drive servo. The receiver worked all right except for its accuracy. To the bottom line, only was the GPS data flow that cooperated in producing the UTC. And a PC controller polled the readiness of the GPS data flow once every 110ms, which meant in worst case the error could reach 110ms. In view of this and later on came along a topic selection for a post-graduate's thesis we decided to greatly improve the accuracy by making fully use of both 1PPS (1 Pulse Per Seconds) signal and 10KHz signal that the receiver output.

2. TIME ACCURACY REQUIREMENT FOR LAMOST

Figure 1 shows the GPS-based time sever with various kinds of hardware components in LAMOST that need time service for control. The Telescope Control System (TCS) is in command of variety of hardware components via Local Control Units (LCU) that are not sketched in figure 1. The GPS Receiver Unit provides an accurate time tick for smooth execution of all real time function necessary for star observation.

[1] Correspondence: Email: xqxu@nairc.ac.cn; Telephone & Fax: 86 25 540 5562

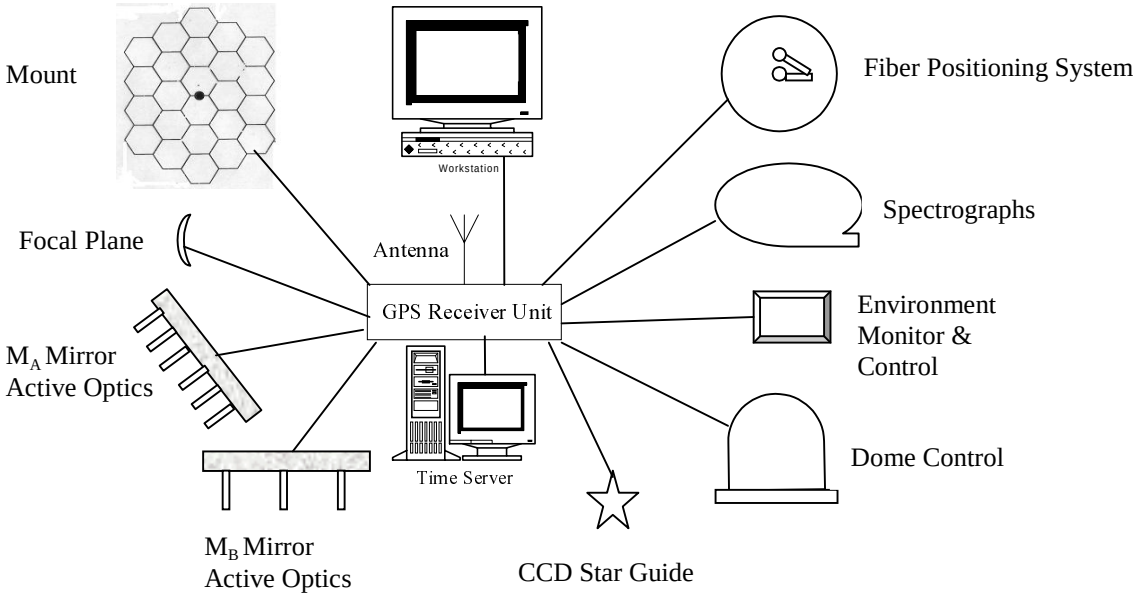

Figure1: GPS-based time sever with various kinds of hardware components

Investigation has been made to determine which function is most time critical. The following table shows the different time criticalness for the control of different hardware components.

Mount tracking
: Most time critical process takes place during the observation, in which the position servo rate could reach 50Hz.

Focal plane adjustment
: Normally there is no real time tip-tilt and focusing adjustment during the observation. However, the servo rate for image rotation compensation comes around 10Hz.

M_A mirror active optics
: The mirror surface correction by applying active force runs once every 1.5 minute during the observation. The image stacking for 24 segment-mirrors normally happens before the observation.

M_B mirror active optics
: The image stacking for 37 segment-mirrors normally happens before the observation.

CCD star guide
: The servo rate for compensating the image drift is predicted around once every minute.

Dome control
: There is no critical real time requirement.

Environment monitor & control
: Meteorological sensors normally sense once every several minutes and respond accordingly.

Spectrographs
: There are dedicated clock drives for CCDs' readout.

Fiber positioning system
: The positioning normally is implemented before the observation.

Obviously most time critical event takes place during the observation for the mount to track the object precisely. According to academician Dingqiang Su and professor Yanan Wang, both from the Nanjing Institute of Astronomical

Optics & technology, there is no blind zone for LAMOST observation area, and the tracking velocities are quite small. The azimuth tracking velocity is smaller than the diurnal velocity (15”/s), and the altitude tracking velocity is even smaller. The maximum field of view rotation is equal to the diurnal velocity. Taking the maximum tracking rate of 15”/s, we get 0.015” deviation if the mount were suddenly stopped for period of 1ms. The deviation is well below the tracking error budget by one order, which is about 0.23” for each mount axis in LAMOST. In conclusion, 1ms accuracy requirement for the GPS-based timing is enough even for most time critical control tasks in LAMOST.

3. OVERVIEW OF GPS-BASED TIMING SYSTEM FOR LAMOST

An in depth discussion of the GPS system is beyond the scope of this paper. We just present here what we have done in our practice to meet the timing requirement stated in section 2.

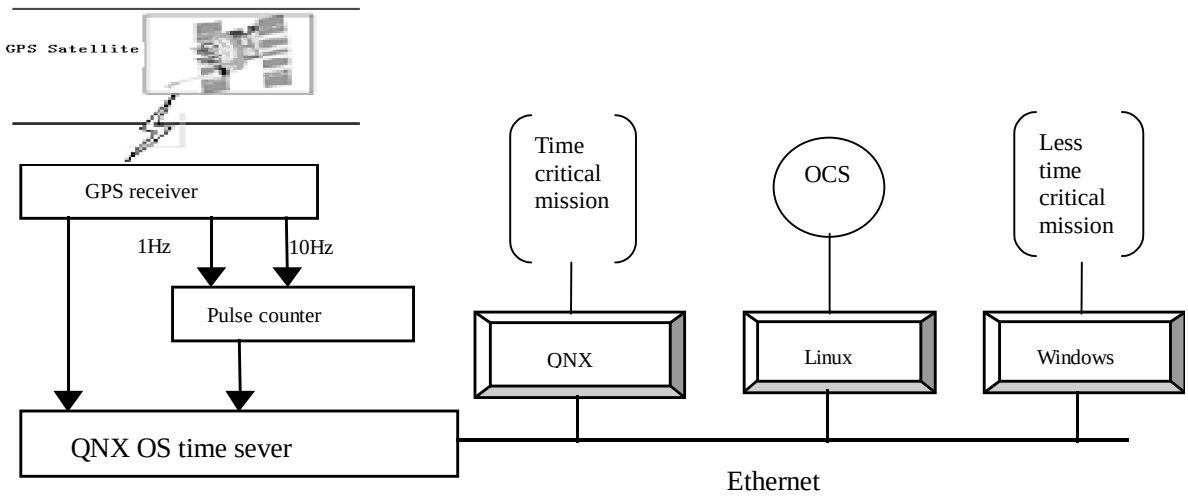

Figure2: GPS-based timing system schematic

The GPS-based timing system schematic for LAMOST is shown in figure 2. The receiver gets GPS signals from a number of GPS satellites, usually more than 3, and processes the information and again in conjunction with the 1PPS time mark pulse and 10KHz time mark pulse through a counter to produce 0.1ms accuracy UTC for the time-server on the LAN. The time-server that is built on QNX node may in turn either actively broadcast or passively get the time synchronization requests to or from various nodes across the network. Typically the timing among the real time QNX nodes can thus reach 1ms accuracy thanks to the Fleet protocol, which is enough to meet all the real time task requirements for LAMOST. Since the LAMOST network consists of multi-operating systems, which give different responding time for time synchronization service, the Linux client for OCS thus can obtain 10ms in accuracy through TCP/IP protocol. Because OCS is on the top of the LAMOST hierarchy and does not deal with hardware components directly, 10ms in accuracy is adequate. The Windows OS is slowest of the three OSs, yet the time accuracy with several tens of ms is guaranteed with Client/Server (C/S) service through TCP/IP protocol, which is enough for its real time tasks in accuracy less than 100ms.

4. HARDWARE PLATFORM FOR GPS NTS

Currently we are working on developing level-1 simulator for LAMOST, which is controlled through LAN. There are five PCs, three of which have been installed with QNX OS, and one with Linux and one with Windows. The NTS is built on one of the QNX node, and the other nodes are served as time-clients. The GPS Receive Unit is comprised of GPS receiver module, a cascaded counter and an I/O interface board. The GPS receiver module we bought is commercial type GSU-25 made in Japan. The major specifications in relation to our research purpose are shown in following table.

Reception:	12channel parallel
Reception Frequency:	1575.42MHz ± 1MHz (C/A code)
Measurement accuracy:	1Sec ± 300nS (typ)
Supply power Voltage:	+5V ± 5%, ripples under 50mV
Current consumption:	Under 200mA
Operation Temperature range:	-30^0C~$+70^0$C
In/Output Communication mode:	Serial transfer with start/stop
Time Mark Pulse output:	1PPS/10KHz

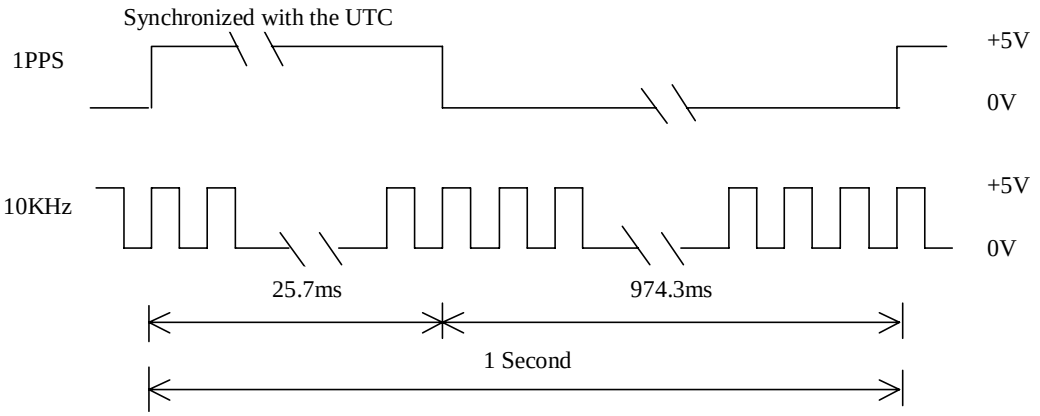

Figure3: Relation between 1PPS and 10KHz

Figure 3 shows the relation between 1PPS and 10KHz. When this receiver provides a valid navigation solution, the rising edge of TIME MARK PULSE is synchronized with the UTC one second's epochs to within ± 1μS. The 1PPS (1Hz) TIME MARK OUTPUT PULSE rise time is typically less than 2ns and the duration typically 25.6ms. The 10KHz waveform is synchronized to the UTC TIME MARK PULSE.

In the GPS world the NMEA-0183 data format with ASCII codes is easy to read and widely used for general purposes. For accurate timing purpose the so-called Rockwell binary format is better, which again includes three types of information. One of the three is called 1108-type that is our concern to generate accurate timing signals in conjunction with the 1PPS and 10KHz. The 1108-type signal is synchronized with 1PPS and output in fixed format, which contains information of 20 bytes and updates once per second. The contents of the 20 bytes are shown in following table.

1st:	00FF
2nd:	0081---------- Standard initialization
3rd:	0004
4th:	0054 ---------- H0454 = 1108
5th:	Header checksum
…….	
14th—15th:	UTC time in units of second starting from the week, 0 ~ 604799
16th:	The time offset in units of seconds between the constant rate GPS and UTC
17th—18th:	Same as above in units of ns
19th:	Checksum

In accordance with the above table we are able to pick up, from the 1108-type data flow, the number of seconds starting from the week. Now we are safe to say the time message we have obtained after the process is accurate to the degree of

one second. However, what we need is far more accurate timing in the system. Fortunately, we still have 1PPS and 10KHz signals come to our rescue. Here is what we have done. Referring to the figure 3, we know that the 1PPS pulse precisely designates the rising of each second, and the 10KHz square wave consists of a serial 0f 0.1ms pulses. These two signals are synchronized at the rising of each second. Four 74ls191 chips have been cascaded together as a 16-bit counter to count the number of 0.1ms pulses. In fact, only 14-bit is needed since the maximum number of 0.1ms pulses in each second is 10000. The 1PPS pulse clicks in and triggers the start of counting the number of 0.1ms pulse. The cascaded counter is so configured that after the number reaches 9999 the next 1PPS will reset the counter to zero, and then the circle comes once again. Therefore it is assured that the number of counter at any moment represents the time period from the start of current second to that very moment with accuracy of 0.1ms. The combination of the number of second received by the serial port and the number of 0.1ms received by an ISA I/O interface card in a slot of a QNX computer will be interpreted as the UTC to the accuracy of 0.1ms as shown in figure 4.

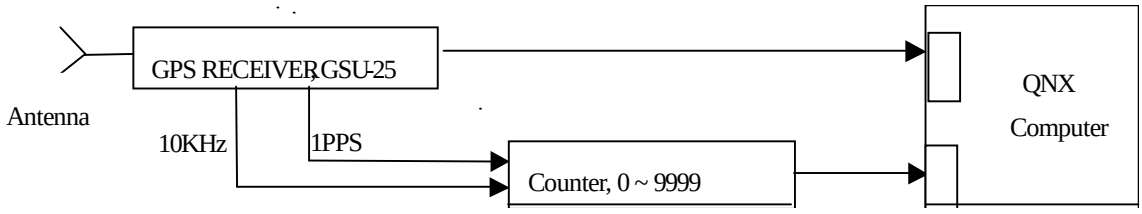

Figure4: GPS-based signal process schematic

5. SYNCHRONIZATION OF QNX TIME-SERVER WIH GPS TIMING

The time-server is built on the QNX OS node. That the QNX OS is adopted as a real time system for LAMOST is because it possesses a number of merits such as its distributed real time nature, user determinable priorities, priority driven, pre-empty schedule, low self-spending of resources and quick context switch, all of which benefit the real time applications. Also it is cost effective. In QNX OS computer the standard UTC is initialized in system hardware when the operating system is installed. Besides there are oscillator and time counter in the system hardware, which adds a certain value to the current UTC value continually and smoothly as time goes along. In addition, there is a programmable mechanism that possibly makes a fine adjustment with resolution of nanosecond level to the clock in QNX. The clock value is also accessible and reset-able by the user. In other words, from software perspective the QNX system clock is adjustable. The way to do it is to change values of two variables, *stime.tv_sec* that contains the number of seconds from the beginning of 1970 and *stime.tv_nsec* that contains the number of nanoseconds that has passed from the beginning of the current second. Of course, we also have means of measuring a time period of an event by accessing these two variables at both the event start and the event end. The difference of the two is the time period of that event measured with accuracy of one nanosecond. This is the method that we frequently use for estimating the network event

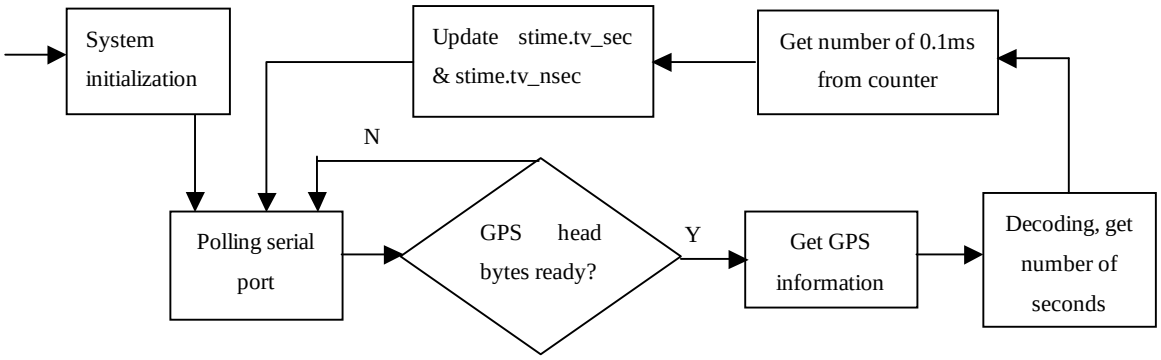

Figure5: Time server flow chart

delay. Up to now, we have got all we need to know in principle to build a time sever on the QNX OS node and synchronize it with output from the GPS Receiver Unit. Figure 5 shows the software flow block diagram for this

purpose. In this flow chart the serial port is polled in circling. For every second it gets the GPS information, then decodes it and pucks up its particular time message, the number of seconds, then again inputs the number of 0.1ms from the counter, finally updates the two time variables. The process of taking the counter number in bit parallel way plus the updating of the time variables goes fast enough in less than 0.1ms and keeps the accuracy of the time server to the degree of 0.1ms. In stead of polling, another possible way is using 1PPS interruption to alert the readiness of GPS information.

6. TIME SYNCHRONIZATION ALGORITHM ACROSS NETWORK

In previous sections we discussed the way in which the UTC is acquired from the GPS Receiver Unit and the synchronization of time-server on a QNX node with the UTC obtained. For economy reason it is not practical to do this across the network on each node. Therefore in our LAMOST control system only one time-server has been built that is the NTS. The NTS provides time synchronization service for each client across the network. The Network Time Protocol (NTP) may serve the purpose without adding more hardware, which we have adopted in our reality. The NTP determines the difference between the NTS clock and the client clock. Since there is no way to get the network delay directly for one-way trip another method has to be figured out. First the NTS transmits a data packet with its time stamp to the client. At the moment of receiving the data packet the client copies own time stamp to the packet and sends it back to the NTS. By the time the NTS receives the packet back it estimates the one-way trip delay by taking a half of the round trip time. Then the NTS transmits a data packet again with its time stamp and the estimated one-way delay to the client. The client gets the packet and corrects its own system time in accordance with the NTS time stamp plus the one-way delay, which completes the time synchronization in C/S mode. This idea is schematically shown in figure 6.

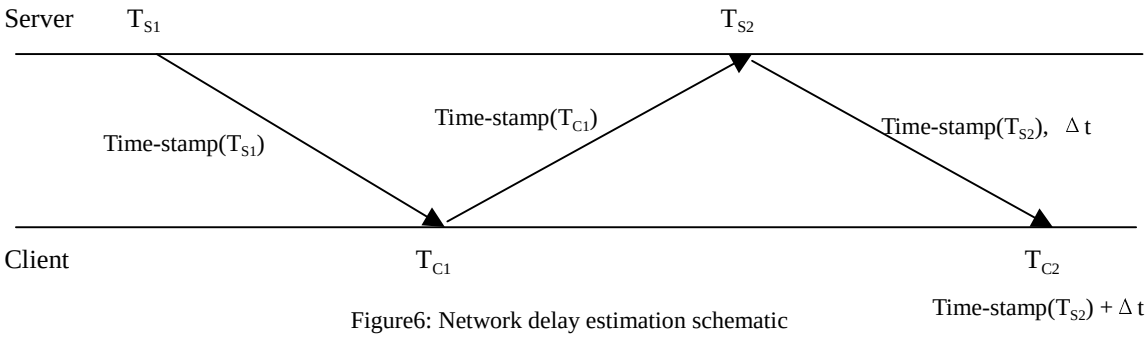

Figure6: Network delay estimation schematic

Of course the average method is just a rough estimation since many factors contribute to the network delay with various pathways and different directions. Fortunately in our practical network approach for LAMOST the network delay does not so much vary in either direction of forward and backward. Besides the estimation error could be reduced greatly if a number of such a measurement are made and the errors are averaged statistically. In practice there are two synchronization modes, one differs from the other in that who plays the active role in the process.

- Active mode. The NTS broadcasts its time message at possible highest refresh rate across the network. Each client picks up the time massage in conjunction with its network delay to synchronize its time with the NTS.
- Passive mode. As apposed to the active mode, in which the NTS actively broadcasts its time message at fixed rate and does not care each individual node's request for different refresh rate, the NTS is requested passively by individual node for time synchronization and responds accordingly.

7. TIME SYNCHRONIZATION AMONG QNX NODES THROUGH C/S MODE

The QNX is a well-known real time OS available on current software market. One of its distinguished features is that all the nodes across a QNX network is treated as a virtual computer with combined capacity of all individual node. This makes it possible to share all the resources among these nodes with transparency. The communication among QNX nodes might be carried out through proprietary FLEET protocol, which is much fast than TCP/IP. Our measurement shows the network delay among QNX nodes with FLEET protocol in our experience is far less 1ms. Therefore it is not

a problem that the time synchronization through C/S mode to 1ms accuracy with the UTC for the QNX client nodes if the QNX NTS has accuracy of 0.1ms. That again means that most critical real time tasks in LAMOST have no risk losing attention.

8. SYNCHRONIZATION BETWEEN QNX NTS AND LINUX CLIENT

Aforementioned OCS is built on Linux OS. The synchronization between QNX NTS and Linux client is still an issue, although not so critical as for the QNX nodes. The following table shows the time delay between QNX NTS and a Linux client through socket communication. In the table all the numbers are with the unit of ms. Send packet time(T_1) designates the time moment that the packet is sent. Receive packet time(T_2) designates the time moment that the packet is received. RTT(T_2-T_1) designates the Round Trip Time. The RTTs vary between 9ms and 22ms with average of

Send packet time(T_1)	Receive packet time(T_2)	RTT(T_2-T_1)
495	515	20
810	831	21
665	684	19
606	616	10
363	385	22
733	742	9
318	337	19
374	386	12
320	331	11
757	776	19

16.2ms, reflecting one-way delay about 8.1ms. Suppose either way delay for each measurement is nearly same, which is quite fair assumption in our reality, and take 8.2ms as Δt, we get the error for one-way between 2.8ms and 3.7ms. At any rate, we are quite safe to say the synchronization between the QNX NTS and Linux client with accuracy 10ms is obtainable, which is our design goal for LAMOST Linux clients.

9. SYNCHRONIZATION BETWEEN QNX NTS AND WINDOWS CLIENT

The synchronization between QNX NTS and Windows client is least critical that the accuracy of several tens of ms is adequate to meet our design goal. The following table shows the time delay between the QNX NTS and a Windows client through socket communication. In the table the RTTs vary between 119ms and 220ms with average of 152.8 ms reflecting one way delay around 76.4 ms. Suppose either way delay for each measurement is nearly same, which is quite fair assumption in our reality, and take 76.4 ms as Δt, we get the error between 16.9 ms and 33.6ms. At any rate, we are quite safe to say the synchronization between the QNX server and Windows client with accuracy 100ms is obtainable, which is our design goal for LAMOST Windows clients.

Send packet time(T_1)	Receive packet time(T_2)	RTT(T_2-T_1)
629	849	220
888	1007	119
111	251	140
534	662	128
945	1065	120
171	331	160
929	1089	160
711	881	170
315	505	190
143	264	121

10. CONCLUSION

Timescale of high precision is of great importance to the control system of LAMOST. We have successfully developed and built a GPS-based NTS by using a commercial GPS receiver added with a counter and some interface circuits to fully utilize the 1PPS and 10KHz signals of the receiver. The NTS is built on QNX OS node and provides the time synchronization across the network nodes under different operating systems such as QNX, Linux and Windows to the accuracy of 1ms, 10ms and 100ms respectively, which have satisfied our design objectives for LAMOST.

ACKNOWLEDGMENTS

We would like to thank Hai Wang for his suggestions during the course of the GPS-based NTS development. The discussion with him in this regard seems to be always beneficial and interesting. Our gratitude also goes to Yizhong Zeng for his part in various electronics work for the experience.

REFERENCES

1. Xinqi Xu, “Control system and technical requirements - preliminary design”, LAMOST Internal Technical Report, 1998.
2. Xinqi Xu, “Study of QNX real time operation system and exploration on its application in large astronomical telescopes”, ASTRONOMICAL INSTRUMENT AND TECHNOLOGY, 1999.
3. Ding-qiang Su, Ya-nan Wang, “The Tracking Motion of the Large Sky Area Multi-object Fiber Spectroscopic Telescope (LAMOST)”, ACTA ASTROPHYSICA SINICA Vol. 17 No. 3, 1997.
4. Cristian, F., “Probability clock synchronization In distributed Computing 3”, Springer Verlag, pp. 146-158,1989.
5. Kopetz, H., “Clock synchronization in distributed real-time systems”, IEEE Trans. Computers C-36, 8, pp. 933-939, August 1987.
6. “TCP/IP programmer’s Guide, Fourth Edition”, QNX Software System Ltd., 1998.
7. “Watcom Compiler & Tools, First Edition”, QNX Software System Ltd., 1996.