Computer Network Time Synchronization using a Low Cost GPS Engine

M. H. Refan* and H. Valizadeh**

Abstract: Accurate and reliable time is necessary for financial and legal transactions, transportation, distribution systems, and many other applications. Time synchronization protocols such as NTP (the Network Time Protocol) have kept clocks of such applications synchronized to each other for many years. Nowadays there are many commercial GPS based NTP time server products at the market but they almost have a high price. In this paper we are going to use a Low Cost GPS engine to build a time server to provide time synchronization with accuracy of a few milliseconds. This time server is relatively very cheap and it can be used in almost all typical applications. We also proposed a software based NTP time server implemented in MATLAB as well.

Keywords: Time synchronization, Time synchronization Protocols, NTP (the Network Time Protocol), GPS Timing, Computer network, Time Server.

1 Introduction

We may usually set our computer's time by our wristwatch to within a minute or two, but on the other side accurate and reliable time is necessary for financial and legal transactions, transportation, distribution systems, and many other applications involving widely distributed resources. To make sense, as an example, in a distributed airline reservation system a seat can be sold twice or not at all if the distributed computers vary in time or there may be legal consequences when an online stock trade is completed, before it is bid [1]. In this regard, coordination to an international time scale and clock synchronization have been developed. The basis for this international level has been refined throughout history and sidereal time, earth rotation based time and atomic time have been developed [2]. Some important time scales with a brief description are presented in Table 1 and a recorded example of them on April 27, 2011 is shown in Table 2 [3].

Clock synchronization deals with the idea that internal clocks of several computers may differ Even when initially set accurately, real clocks will differ after some amount of time due to clock drift [4], caused by clocks counting time at slightly different rates so there is always need for keeping these drifty clock synchronous to a reference clock or with another more accurate clock.

Table 1 A brief description of some important time scales.

Time scales	Description
TAI	International Atomic Time[a], is the international atomic time scale based on a resonance frequency between selected energy levels of Cesium atom to an accuracy of a few parts in 10^{12} [5]
UTC	Coordinated Universal Time[a]. UTC is presently slow relative to TAI by a fraction of a second per year
LT	Local time differs from UTC by the number of hours of a time zone.
GPS	Global Positioning System time is the atomic time scale implemented by the atomic clocks in the GPS ground control stations and the GPS satellites themselves. The general GPS system time is expressed as a week number and the number of elapsed seconds in that week.

a. Conventional cultural sensibilities require descriptive terms in English and abbreviations in French.

Table 2 Recorded example of important time scales recorded on April 2011.

LT	2011-04-27 16:50:19	Wednesday	Day117	Time zone UTC+4.5
UTC	2011-04-27 12:20:19	Wednesday	Day117	MJD 55678.5141
GPS	2011-04-27 12:20:34	Week 1633	303634 s	Cycle1, week 0609, day 3
TAI	2011-04-27 12:20:53	Wednesday	Day 117	34 leap seconds

Iranian Journal of Electrical & Electronic Engineering, 2012.
Paper first received 17 Sep. 2011 and in revised form 4 July 2012.
* The Author is with the Department of Electrical and Computer Engineering Faculty, Shahid Rajaee Teacher Training University, and MAPMA Electrical and Control Engineering & Manufacturing Co. (MECO).
E-mail: refan@srttu.edu; refan@mapnaec.com.
** The Author is with the Department of Electrical And Computer Engineering, Shahid Rajaee Teacher Training University, Tehran, Iran.
E-mail: hs_valizadeh@yahoo.com.

Synchronization directly to UTC requires a specialized radio or satellite receiver, or telephone modem source. Such sources are available for many governmental and industrial dissemination services, including the Global Positioning System (GPS), WWV/H and WWVB radio time/frequency stations [6] and [7]. U.S. Naval Observatory (USNO) and National Institute of Science and Technology (NIST) telephone modem services in the United States [8], DCF77 long wave radio time station in Germany, JJY radio time station in Japan, as well as similar systems and services in other countries [9]. If every computer be equipped with one of these clocks, the entire above mentioned problems would be solved, but for reasons of cost, unavailability in some places and their complexity it is not possible to equip every computer with a reference clock. Furthermore, the reliability requirements for time synchronization may be so strict that a single clock cannot always be trusted. Therefore, for time synchronization in practice, a structure similar to Fig. 1 is being used.According to that, some numbers of computers are getting time from reference clocks themselves and then act as primary time servers to feed a much larger group of secondary servers and clients connected with a common network with an accurate and reliable time.

Reference clocks at the top of the hierarchy should be very accurate; Nowadays Thanks to the many progresses in Global Positioning System, its time accuracy over radio stations (GPS: short-term accuracy of ± 1 microsecond, while radio signal accuracy is: +5 to +25 millisecond [10]), Noise Immunity, and worldwide availability for free, GPS based Clocks are used very often as the reference clocks over the other clock recourses.

At the present time there are many commercial GPS time synchronization products at the market but they almost have a high price, for instance a typical one is about two thousand dollars, therefore in this paper we are going to use a low cost GPS engine to build a precise clock for time synchronization. This clock is relatively very cheap and it can be used in almost all typical applications.

The remainder of this document is organized as follows. Section 2 describes some important time synchronization protocols. In Section 3 we explain how a Computer Network Time is Synchronized Using the Network Time Protocol (NTP). A brief explanation of timing data in a GPS receiver is clarified in section 4. Section 5 proposes a MATLAB based and a standalone time server board for synchronizing computer networks time. Finally computer network time synchronization results for the two proposed time servers and conclusion are presented in sections 6, 7 respectively.

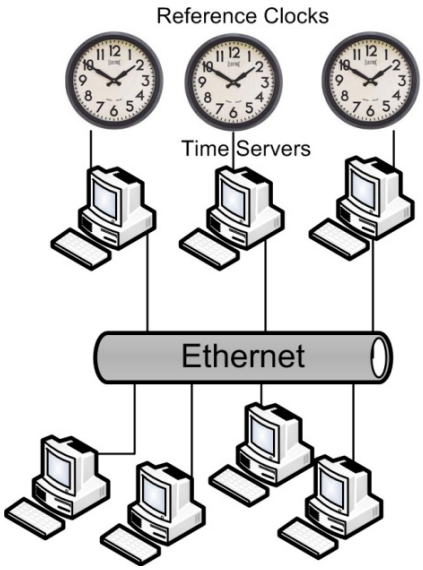

Fig. 1 A typical time synchronization structure.

2 Time Synchronization Protocols

To keep time of computers synchronized to the primary time servers in a distributed network, clock synchronization protocol is required that can read a server clock, transmit the reading to one or more clients, and adjust each client clock as required.

The various synchronization protocols in use today provide different means to time synchronization, but they all follow the same general model. The client sends a request to the server and the server responds with its current time and for the best accuracy, the client needs to measure the server-client propagation delay to determine the true time offset relative to the server.

Some Important standard time synchronization protocols are as follows:

2.1 Time Protocol

Time protocol, specified in RFC868 [11], provides a site-independent, machine readable date and time. This simple protocol returns a 32-bit unformatted binary number that represents the time in Universal Time Coordinate (UTC) seconds since January 1, 1900. The server listens for time protocol requests on port 37, and responds in either TCP/IP or UDP/IP formats. Since the TIME protocol sends timestamps in seconds, it can provide only ±1 second accuracy.

2.2 Daytime Protocol

Daytime Protocol, specified in RFC867 [12] is widely used by small computers that run MS-DOS and similar operating systems. The server listens on port 13, and responds to requests in either Transmission Control Protocol/Internet Protocol (TCP/IP) or User Datagram Protocol/Internet Protocol (UDP/IP) formats. The standard does not specify an exact format for the daytime protocol, but requires that the time is sent using standard ASCII characters. Similar to TIME protocol

daytime protocol sends timestamps in seconds and it can provide only ±1 second accuracy too.

2.3 Network Time Protocol

Network Time Protocol (NTP) originally specified in RFC958 [13] and later in RFC1059 [14], RFC1119 [15], RFC1305 [16], and current version of RFC5905 [17], is the most complex and sophisticated of the time protocols for synchronizing computer clocks across a network. Because NTP software is often bundled with the operating system it is the most common used protocol for computer network time synchronizations. The NTP client software runs continuously as a background task that periodically receives updates from one or more servers. The client software ignores responses from servers that appear to be sending the wrong time and averages the results from those that appear to be correct. The NTP servers listen for a NTP request on port 123, and respond by sending a UDP/IP data packet in the NTP format. The time stamps in this time protocol are in 64-bit, consist of 32-bit second part and 32-bit fractional seconds part allowing theoretical resolution of 2^{-32} second (233 picoseconds), but in practice the accuracy of NTP depends on the network environment. In most places of the Internet of today, NTP provides time accurate to the order of 10-100 msec. while Under good conditions on a LAN without too many routers synchronization to within a few milliseconds is normal [18].

2.4 Simple Network Time Protocol

Simple Network Time Protocol (SNTP) originally specified in RFC1361 [19] and later in RFC1769 [20], RFC2030 [21], and the current version RFC4330 [22], is a less complex implementation version of NTP. It provides a simplified access strategy for servers and clients that do not require the degree of accuracy of the NTP protocol. The network packet formats of both NTP and SNTP protocols are identical, and the two are interoperable. The main difference between the two is missing the complex filtering algorithms to maintain an accurate time that NTP provides and the accuracy is around tens of milliseconds [18].

2.5 Precision Time Protocol

The Precision Time Protocol (PTP), as defined originally in the IEEE 1588-2002[23] and then with IEEE 1588-2008[24] standard, provides a method to precisely synchronize computers over a Local Area Network requiring accuracies beyond those attainable using NTP. An existing LAN, PTP is capable of synchronizing multiple clocks to better than 10 microseconds RMS, but on the other hand it is more expensive in implementation than NTP [25].

3 Computer Network Time Synchronization Using NTP

For being open source, having sufficient accuracy for typical applications and the ability to work on large networks, NTP is the one widely in use on the public Internet and numerous private networks for over almost three decades. NTP comes with most flavors of Windows as well as all flavors of UNIX. About 25 million clients implode on the NTP time servers at NIST alone [18].

3.1 Computer Clocks and NTP

Most computers have quartz or surface acoustic wave (SAW) resonator stabilized oscillator and a hardware counter that interrupts the processor at intervals of a few milliseconds, called the tick [18]. At each tick interrupt, this value is added to a system variable representing the clock time. Clock errors are due to systematic (offset) variations in network delays and latencies in computer hardware and software (jitter), as well as clock oscillator wander. The time of a computer clock relative to ideal time can be expressed as Eq. (1) [18]:

$$T(t) = T(t_0) + R(t - t_0) + D(t - t_0)^2 + x(t) \qquad (1)$$

where t is the current time, $t0$ is the time at the last measurement update, T is the time offset, R is the frequency offset, D is the drift due to resonator aging, and x is a stochastic error term.

The first two terms include systematic offsets that can be bounded by some analysis and NTP estimate these two. The third term is usually dominated by errors in the first two terms and the last random variations that cannot estimated because of its stochastic characteristics.

3.2 Network Time Protocol Principles

NTP has three major parts: the NTP software program, called a daemon in UNIX and a service in Windows; a protocol that exchanges time values between servers and clients; and a suite of algorithms that processes the time values to advance or retard the system clock [18]. For instant, we are not going to cover all the three but we are intending to describe the Protocol which is in need for designing a NTP time server. Further details can be found in the formal specifications [14-17].

The most important field in the NTP packet is the time stamp field, as it is shown in Fig. 2 an NTP timestamp is a 64-bit unsigned fixed-point number, with the integer part in the first 32 bits showing the past seconds from 0h 1 January 1900 and the fraction part in the last 32 bits.

The precision of this representation is about 2^{-32} second (233 picoseconds), which should be adequate for even the most exotic requirements.

Second since 1900(32bit)	Fraction of second(32bit)

Fig. 2 NTP packet timestamp format.

Table 3 NTP packet timestamp conversion example.

Time	Apr 26,2011 20:05.563181
Difference time from 0h 1 January 1900	111 years, 3 months, 3 weeks, 4 days
Total seconds difference	3512764800 seconds
Fractional part	0.56318 seconds
NTP time stamp second field	3512837107 seconds (Hex: D161A3F3)
NTP time stamp fraction of second field	0.563181 seconds (Hex:902CA4C0)

To convert a time to this format we should calculate the seconds past since 0h 1 January 1900, leap years should be also considered. An example of this conversion is shown in Table 3.

Note that since some time in 1968 the most significant bit of the 64-bit field has been set and that the field will overflow some time in 2036, for making NTP work even from that time on, there will be 128-bit time stamps format in the next NTP versions in which the years can span the age of the universe.

Fig. 3 shows how the timestamps are numbered and exchanged between server B and client A. First, client A sends the current time, T1, to server B. Upon arrival, B saves T1 along with the current time T2. Server B does not have to respond immediately, because it may have other duties. Sometime later, B sends the current time T3, along with the saved T1 and T2 to A. Upon arrival, A reads its clock, T4, and proceeds to compute both time offset θ and round-trip delay δ relative to B according to Eq. (2) and Eq. (3):

$$\theta = \frac{1}{2}[(T2 - T1) + (T3 - T4)] \quad (2)$$

$$\delta = (T4 - T1) + (T3 - T2) \quad (3)$$

These values are processed in client by a suite of three concatenated algorithms, including the selection, clustering, and combining algorithms [18], the protocols also provide a way to detect duplicate and bogus packets. The result of the algorithms is a single time value representing the best guess of the system clock offset then the adjustment is implemented by the system clock.

The NTP packet is a UDP datagram [26]. The NTP packet header shown in Table 4 has 12 words followed by optional extension fields and an optional message authentication code (MAC). Following is a short description of the various fields. A complete description is given in [14-17].

Leap Indicator (LI): Warns of an impending leap second to be inserted or deleted in the UTC timescale at the end of the current day.

Version Number (VN): Identifies the NTP version.

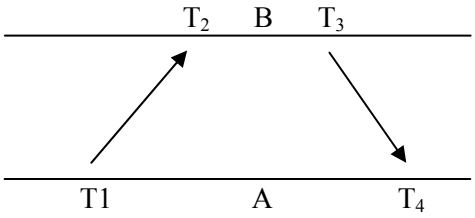

Fig. 3 Client (A) and server (B) NTP packet exchange.

Table 4 NTP Packet Header.

LI	VN	Mode	Stratum	Poll	Precision
Root Delay					
Root Dispersion					
Reference ID(32 bit)					
Reference Timestamp (64 bit)					
Originate Timestamp (64 bit)					
Receive Timestamp (64 bit)					
Transmit Timestamp (64 bit)					
Extension Field 1 (optional)					
Extension Field 2 (optional)					
MAC (optional)					

Mode, Stratum, and Precision: Indicate the current operating mode, stratum and local-clock precision.

Poll Interval (Poll): The current desired interval between NTP messages sent.

Root Delay: Total round-trip delay to the reference clock.

Root Dispersion: Total dispersion to the reference clock.

Reference ID: 32-bit ASCII [27] string code identifying the particular server or reference clock.

Reference Timestamp: Time when the system clock was last set or corrected, in NTP timestamp format.

Origin Timestamp: Time at the client when the request departed for the server, in NTP timestamp format.

Receive Timestamp: Time at the server when the request arrived from the client, in NTP timestamp format.

Transmit Timestamp: Time at the server when the response left for the client, in NTP timestamp format.

Destination Timestamp: Time at the client when the reply arrived from the server, in NTP timestamp format.

Extension Field 1and 2: used to add optional capabilities for example, the Autokey security protocol [28].

Message Authentication Code (MAC): consisting of the Key Identifier field and Message Digest field [17].

4 GPS Timing

Recall from the introduction, GPS receivers can play the role of reference clocks; here we explain time data in a typical GPS receiver in brief. The GPS receiver we used here is NEO-5Q GPS receiver module [29] which is a family of stand-alone GPS receivers featuring the

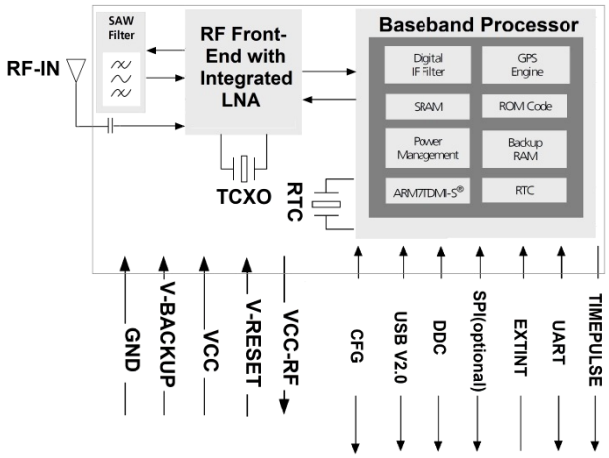

Fig. 4 In use GPS receiver block diagram.

high performance ublox-5 positioning engine from U-BLOX company available in a development board for only about 50$ at [30].

As the block diagram of this GPS receiver in Fig. 4 is depicting [31], it supports four communication interfaces consist of USART, USB, SPI and DDC. Simply stated the GPS gets coded signals via an RF antenna from GPS satellites in view and after decoding and processing that signals in its Baseband Processor, provides navigation and timing data via the four selectable communication interfaces.

Many GPS receivers communicate with other devices using NMEA 0183 (first released in March of 1983 [32] and recently replacing by NMEA 2000® [33]) standard protocol messages, but there are also Binary protocols which are invented by different GPS receiver manufacturing companies to provide higher data rates and in detail data compared to the NMEA protocol. U-BLOX Company uses both NMEA and a proprietary UBX binary protocol [34] in its products. Table 5 shows an example of the two protocols containing time data captured from the GPS. As it is seen there, the binary protocol provides time to an accuracy of a microsecond while NMEA provides it to millisecond accuracy.

Most GNSS receivers generate a time pulse every second, referred to as 1 PPS (1 Pulse Per Second), which is synchronized to UTC. The accuracy of 1PPS in GPS receivers is tens of nanoseconds; some specifications of the GPS receiver in use, including 1PPS accuracy is shown in Table 6.

The in use GPS receiver provides a hardware-synchronized time pulse pin with a selectable time pulse with period of 1S to 4S (0.25 to 1 Hz). Every rising/falling edge (configurable) of this pulse is happening in an UTC time with very a high accuracy of tens of nanoseconds (Table 6). Fig. 5(a) shows 1PPS signal configured with pulse period of 1 second with every rising edge happening on UTC seconds. The related time data for each pulse is achievable by reading the TIM-TP binary message from GPS. TIM-TP message contains time information of the next time pulse and is available a few millisecond after each time pulse via a communication interfaces. Fig. 5(b) illustrates two continues serial output data containing this message captured by an oscilloscope.

Table 5 Example of NMEA and binary protocol time messages in the GPS receiver.

Message	ZDA	NAV-TIMEUTC
Protocol	NMEA	UBX
Structure	$GPZDA, hhmmss. sss, day, month, year, ltzh, ltzn*cs<CR><LF>	Header, ID, Length, iTOW, nano, year, month, day, hour, min, sec, Validity Flags
Example	$GPZDA,082710.232,16,09,2002,00,00*64	B5 62 01 21 14 00 EE 7B 5E 12 0C 00 00 00 EF FB D9 FF D3 07 02 13 0D 24 09 07 0D 69
Decoded Time Data	Date: 16.9.2002 Time:08:27:10 Millisecond:232	Date:19.2.2003 Time:13:36:09 Millisecond:998 Nanosecond:2491409

Table 6 In use GPS receiver specifications summery

Parameter	**Specification**	
Receiver type	50 Channels GPS L1 frequency, C/A Code GALILEO Open Service L1 frequency	
Tracking & Navigation Sensitivity	-160 dBm	
Horizontal position accuracy	< 2.0 m	
Accuracy of Time pulse signal	RMS 99%	30 ns <60 ns

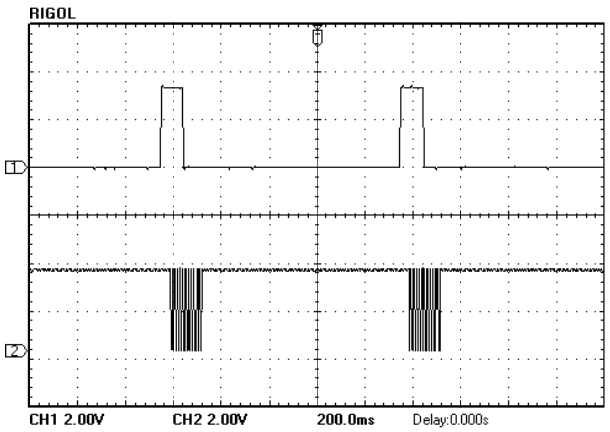

Fig. 5 (a, upper: ch1) GPS receiver 1PPS signal (b, lower: ch2) GPS receiver serial output data for TIM-TP message (200 msec./div,2V/div).

5 Implementing a Time Server

5.1 MATLAB Based Time Server

In this part we present a MATLAB program which acts like a GPS master clock, synchronizing the time of a computer network to an accuracy of 80±20 milliseconds. Although this accuracy is not so much but it is usually enough for a typical computer network within an office which most computer clocks are set by eyeball-and-wristwatch to within a minute or two and rarely checked after that.

Fig. 6 shows the overall scheme for this work. A MATLAB program is developed that communicate with GPS receiver via USB interface. As the Algorithm in Fig. 7 shows, first, the program waits for an NTP packet by listening to 123 port and upon arrival of a client request, it stores the request packet in a temporary variable and polls the NAV-TIMUTC massage (by sending a request to the GPS receiver and getting GPS response containing the current time).Then it converts current time data in to NTP time stamp and fills NTP packet fields according to Table 7 and finally sends the response back to the client address. Although MATLAB has a built-in UDP/IP Packet send/receive block in xPC Target™ toolbox but it can just be run on MATALB SIMULINK program, and both Host and Client should run Simulink program to establish their communication, so we made use of some functions of a free toolbox provided by MATLAB central file exchange [35] to establish UDP/IP connection between the host PC running MATLAB and any other computer connecting it via Ethernet notwithstanding it runs MATLAB or not.

Regarding the fact that whenever MATLAB time server wants to have the current time it polls a time message, there is a problem with MATLAB time server. According to Fig. 8, when we poll a message from GPS receiver (polling serial data is shown in Fig. 8(a)), it response about 42 milliseconds later and the response duration is about 80 milliseconds (response serial data is shown in Fig. 8(b)), so when MATLAB time server gets the polling response, the time data included in the response has always a 80 milliseconds error regarded to the current time. Therefore, when MATLAB sends a NTP packet, the data points to 80 msec. back in time. This error can be compensated by adding 80 msec. time offset to the packets before sending them in the program.

To overcome this issue, we should made use of the time pulse signal and its related time data message in the GPS receiver. What we need to do is to configure the time pulse to trigger every one second with a rising/falling edge then read the related message, make the needed time stamps and answer to the clients requests exactly when the time pulse is at its rising/falling edge. This means we need to utilize 1PPS as hardware interrupt which is not an easy task in MATLAB and there should be a data acquisition interface installed. Meanwhile, a small low price microcontroller can do the job!

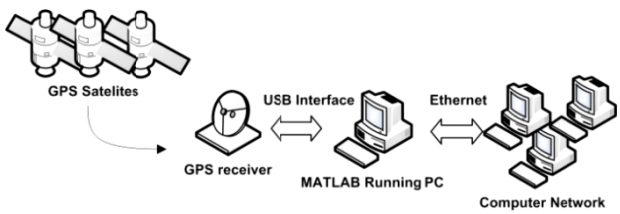

Fig. 6 Overall scheme of MATLAB time server.

Table 7 Needed actions for building the NTP packet in different working modes.

	Multicast mode						Unicast/Anycast mode					
LI	0						00					
VN	4						copied from request					
Mode	5						2 or 4					
Stratum	1						1					
Poll	10						10					
Precision	-10						-10					
Root delay	0 seconds						0 seconds					
Root dispersion	0.5 seconds						0.5 seconds					
Reference ID	GPS						GPS					
Reference timestamp	Set to the current GPS time						Set to the current GPS time					
Originate timestamp	0 seconds						Copied from transmit timestamp					
Receive time stamp	0 seconds						Set to the current GPS time					
Transmit timestamp	Set to the current GPS time						Set to the current GPS time					

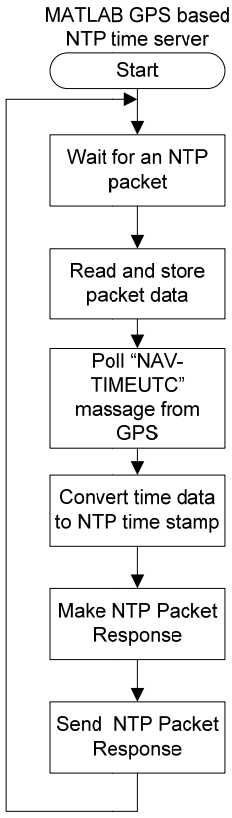

Fig. 7 MATLAB time server program algorithm

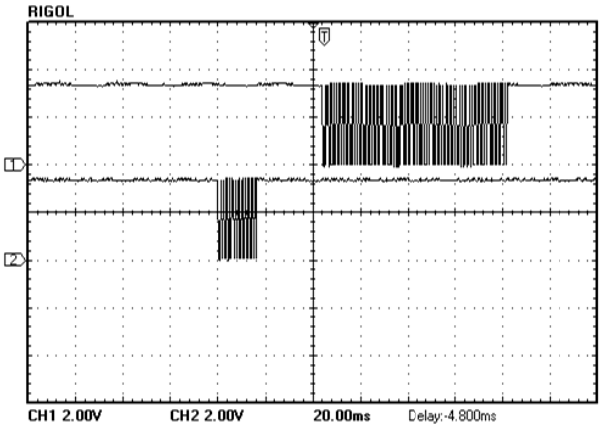

Fig. 8 (a, upper: ch1) GPS receiver response serial data (b, lower: ch2) polling serial data (20 msec./div,2V/div).

5.2 Standalone Time Server Board

Here we are going to design a Standalone time synchronization server which can be used to synchronize time in precise time needing applications. The board will connect to GPS receiver, Ethernet interface, and provides accurate time for the clients using NTP.

As the overall scheme of this Time server in Fig. 9 shows, it consists of a GPS receiver, an Ethernet Controller IC, which acts like a Network Ethernet Card under IEEE 802.3, a microcontroller to Process GPS data and Ethernet Packets, and a graphical LCD to show current Time.

The implemented algorithm in the microcontroller is presented in Fig. 10, it consist of three sub algorithms, main loop, Ethernet controller Interrupt Service Routine (ISR) and Time pulse ISR. At the first step, the Main loop initializes Ethernet controller, GPS receiver and graphical LCD, then it enters to an endless loop, waiting for Interrupts from GPS time pulse and the Ethernet controller. The Ethernet controller ISR is called every time a Packet is received by the Ethernet controller IC, Packet is checked then to see if it is an NTP Packet with time server's IP address in its destination field, and if it be the packet that should be answered. A reply packet is made by using "next-pulse-time" variable's time data, then a flag is set letting the microcontroller to send the packet upon arrival of the next Time pulse. The Time pulse ISR is set to be called at the rising edge of time pulse; this ISR has higher priority than the Ethernet controller ISR and does a simple job. It just checks the "Send–flag" and if it be set, sends the NTP packet then reads TIM-TP message and stores it in "next-pulse-time" variable.

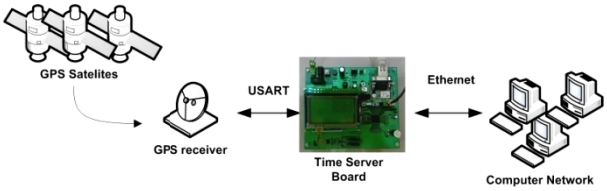

Fig. 9 Overall scheme of MATLAB time server.

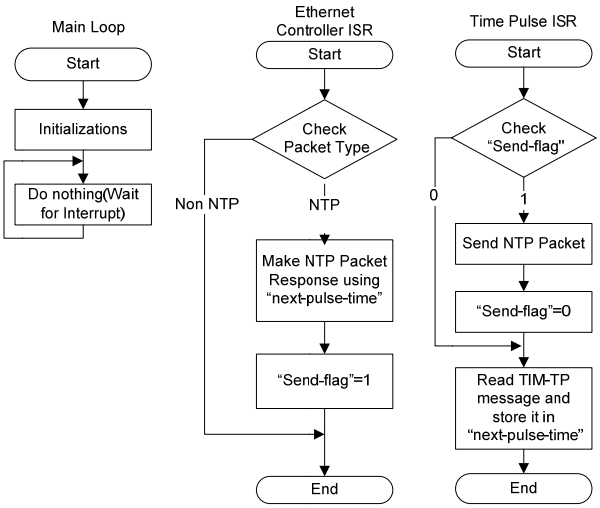

Fig. 10 standalone time server program algorithm.

6 Results

6.1 MATLAB Based Time Server

To examine if our program works correctly or not, we tested it on a simple computer network consist of two PCs connected directly with an Ethernet cable. PC-1 has IP: 192.168.1.10 with Windows XP installed, and the PC-2 has IP: 192.168.1.2 with Win 7.

Windows XP and Windows 7 both have an integrated time synchronization service (w32time service) installed by default, which can synchronize to a NTP Time Server.

We disabled the PC-2's W32time service to let MATLAB work instead. Moreover, by manipulating registry settings [36] in PC-1, we made it act as an NTP client.In order to PC-1 be synchronized by PC-2,it sends NTP request to PC-2's IP address, which our program was running there, then ran MATLAB program and it could successfully synchronize that.

Fig. 11 displays a screenshot of "date and time". Fig. 12 shows "event viewer" of PC-1 showing the successful time synchronization. Fig. 13 shows a screenshot of Wireshark (a free network protocol analyzer [37]) capturing the NTP packets transmitted between MATLAB and PC-2.

Whenever a NTP packet is received in the client, client synchronizes its local clock according to the packet, meanwhile the installed Wireshark network protocol analyzer timestamps the receiving packets to 1 microsecond accuracy.

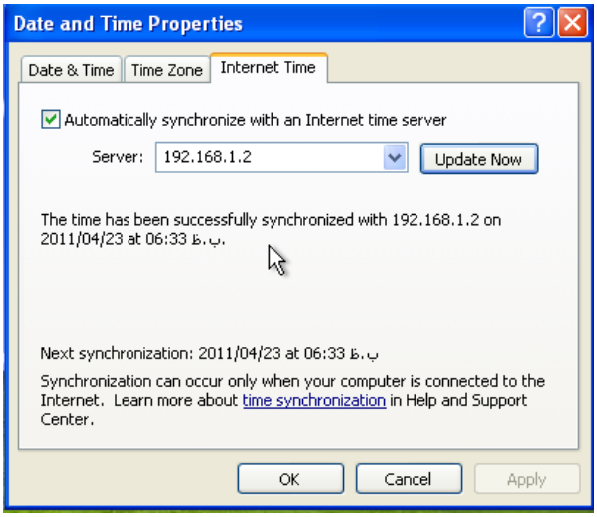

Fig. 11 Screenshots of "date and time" in the Client PC.

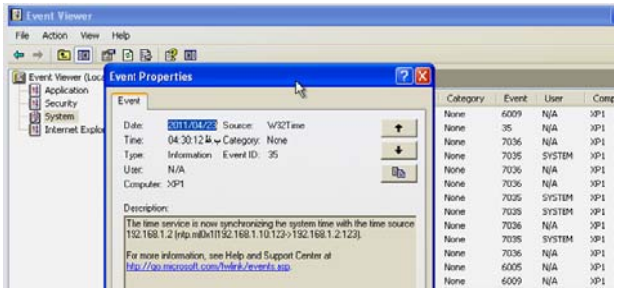

Fig. 12 Screenshots of "event viewer" in the Client PC.

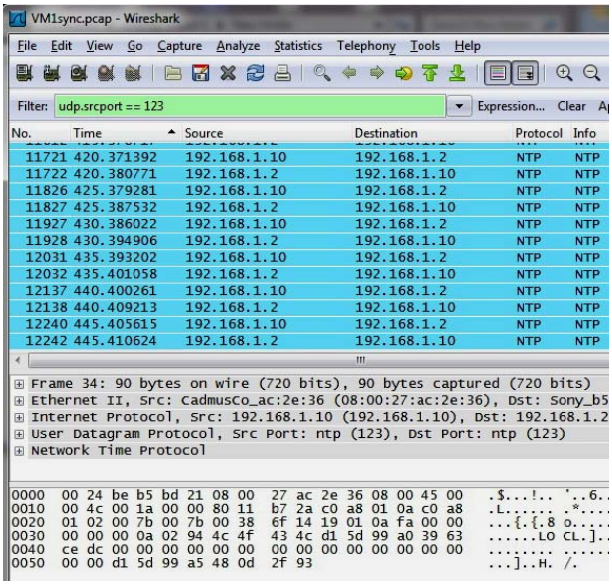

Fig. 13 Screenshot of Wireshark [37] software capturing the NTP packets transmitted between MATLAB and PC-2.

To get the synchronization accuracy, we can use time differences between 2 continuous received NTP packets in the client. These difference values should be identical in zero accuracy. The difference between these time differences is a measure of the time sever's floating accuracy. Fig. 14 shows a chart of these time

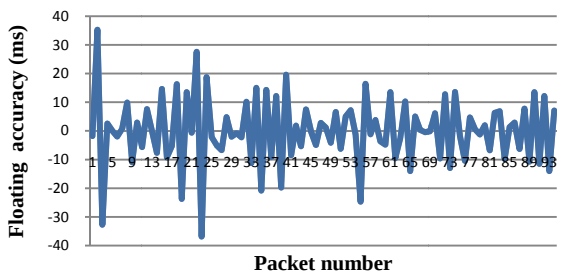

Fig. 14 MATLAB time server floating accuracy.

values for 100 NTP packets received by the client, as it is seen there, they are almost limited to ±20 milliseconds.

Recall from 5.1 the 80 millisecond error in sending packets was compensated, so it won't harm the accuracy and therefore the accuracy of synchronization will be around ±40 milliseconds.

6.2 Standalone Time Server Board

We used a product of ATMEL microcontrollerand an Ethernet controller in implementing the NTP server in practice and implemented a prototype board shown in Fig. 15.

In order to test the board, it was connected to a computer with "Sun virtual box" [38] virtual Machine software installed in it. Then made two virtual machine, installed Windows XP in each, networked all three together and configured W32 time service in each to be synchronized from Standalone NTP server (with IP: 192.168.1.20). All computers successfully synchronized to the Time server, Fig. 16 shows a screenshot of "date and time" of three computers. Fig. 17 shows screenshot of Wireshark software capturing the NTP packets transmitted between three computers and the standalone time server in which client requests and Time server reply are traceable.

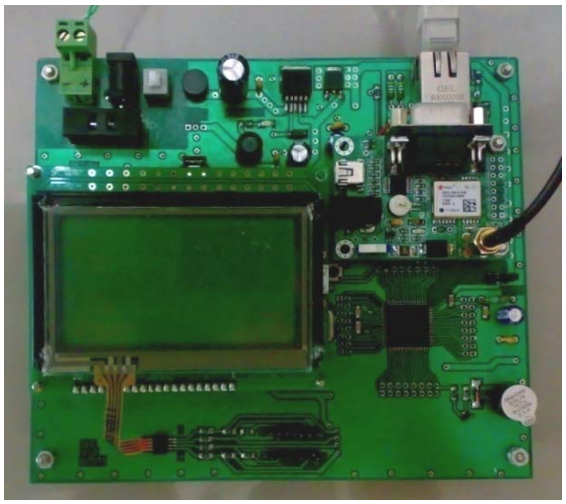

Fig. 15 NTP time server prototype board.

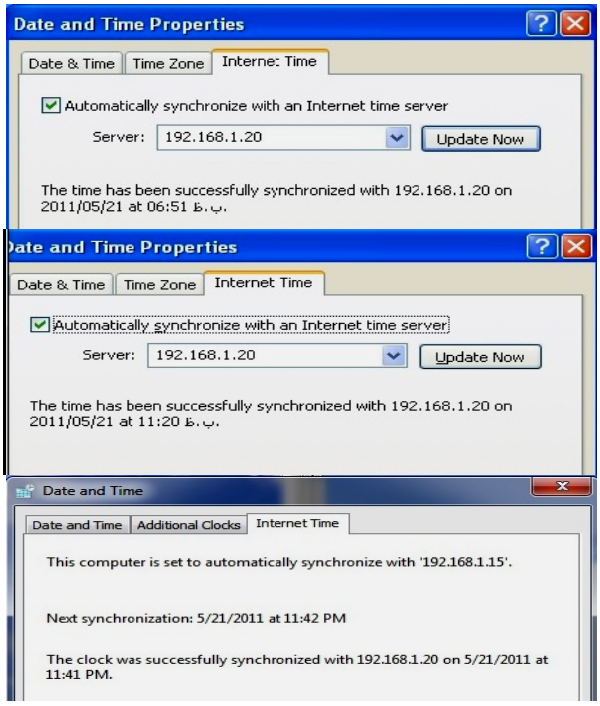

Fig. 16 Screenshot of "date and time" of three computers synchronized to the standalone time server.

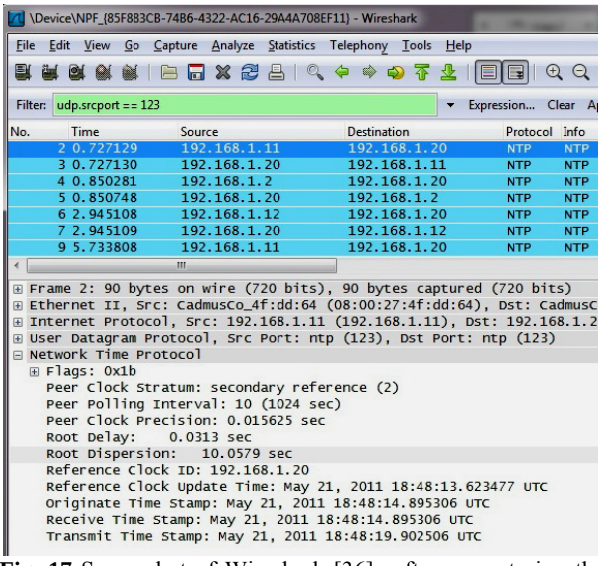

Fig. 17 Screenshot of Wireshark [36] software capturing the NTP packets transmitted between three computers and the standalone time server.

Fig. 18 shows a chart of client clock floating accuracy for 100 NTP packets received by a client, as it is seen there, the floating accuracy of synchronization will be around ±10 milliseconds.

According to Fig. 19 that shows an oscilloscope screenshot of 1PPS signal and "send-flag" status (see Fig. 10) there is about 15 msec. time difference between the GPS 1PPS signal and the packet sending time. This msec. offset every time. Similarly this error is

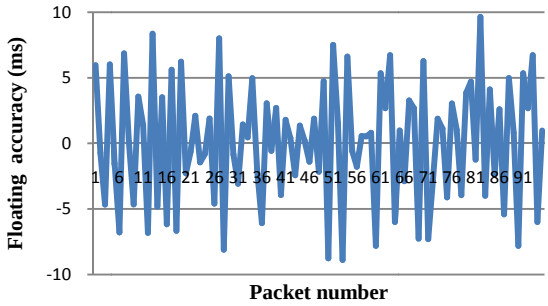

Fig. 18 Standalone time server floating accuracy.

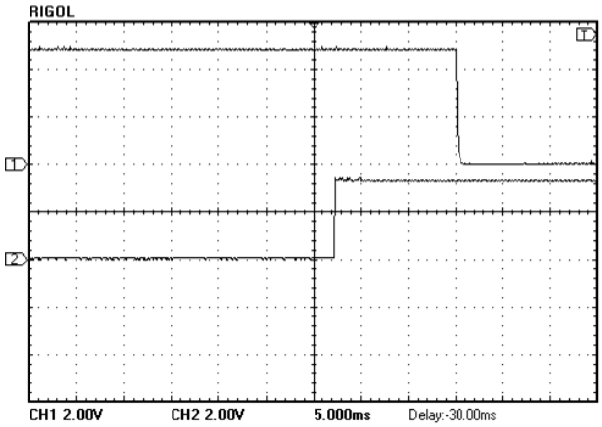

Fig. 19 (a, upper: ch1) standalone NTP server "send-flag" status (b, lower: ch2) GPS receiver 1PPS signal (5 msec./div,2V/div).

compensated by adding a time offset in the packet sending routine program, therefore the total accuracy will be remain around ±10 msec.

7 Conclusion

Time Synchronization of clocks is a vital need in many modern networks especially in the computer networks. Generally in a time synchronization procedure, an external time source is used for spreading timedata to the time needing applications via some time synchronization protocols. Today among many different types of time sources, the GPS is more often used as a time synchronization source due to its significant advantages over other sources.

In this study, GPS was used as the external time source. Meanwhile, because of great accuracy and worldwide popularity of NTP protocol, it was chosen as the time synchronization protocol.

A MATLAB software based and a standalone time server board have been designed and implemented. These two, both use GPS timing signals to get precise time and then spread it in NTP packet message to the clients. Implemented Time servers have a few millisecond accuracy. The standalone time synchronization board is relatively very cheap and can be developed as a commercial product.

References

[1] Liskov B., "Practical uses of Synchronized Clocks in Distributed Systems", *Distributed Computing*, Vol. 6, pp. 211-219, Apr. 1993.

[2] The University of California Observatories, *Time scales*, in URL: http://www.ucolick.org/~sla/leapsecs/timescales.html, accessed on July18, 2012.

[3] *GPS, UTC, and TAI Clocks*, in URL: http://www.leapsecond.com/java/gpsclock.htm, accessed on July 18, 2012.

[4] Mills D. L., "On the accuracy and stability of clocks synchronized by the network time protocol in the internet system", *ACM SIGCOMM Computer Communication Review*, Vol. 20, pp. 65-75, 1989.

[5] Allan D. W., Gray J. E. and Machlan H. E., "The National Bureau of Standards Atomic Time Scale: Generation, Stability, Accuracy and Accessibility". In: Blair, B.E. (Ed.). *Time and Frequency Theory and Fundamentals*. National Bureau of Standards Monograph 140, U.S. Department of Commerce, pp. 205-231, 1974.

[6] Lombardi M. L., "NIST Time and Frequency Services", *NIST Special Publication 432*. Jan. 2002. Available online at: http://www.nist.gov/timefreq/general/pdf/1383.pdf

[7] Nelson G. K., Lombardi M. A., and Okayama D. T., "NIST Time and Frequency Radio Stations: WWV, WWVH, and WWVB," *NIST Special Publication 250-67,* 161 pages, Jan. 2005.

[8] Levine J., Weiss M., Davis D.D., Allan D.W. and Sullivan D.B., "The NIST automated computer time service." *J. Research National Institute of Standards and Technology*, Vol. 94, pp. 311-321, Sep. 1989.

[9] Standard Time Signal Broadcast Channels, *SW Time Signal Broadcasts*, in URL: http://www.dxinfocentre.com/time.htm, accessed on July 18, 2012.

[10] Accuracy between DCF77 and GPS, *hopf Elektronik - Accuracy between DCF77 and GPS,* in URL: http://www.hopf.com/en/dcf-gps.htm, accessed on July 18, 2012.

[11] Postel J. and Harrenstien k., "Time protocol", *DARPA Network Working Group Report RFC-868, USC Information Sciences Institute*, May. 1983.

[12] Postel J., "Daytime protocol.", *DARPA Network Working Group Report RFC-867, USC Information Sciences Institute*, May. 1983.

[13] Mills D. L., "Network Time Protocol (version 0)", *DARPA Network Working Group Report RFC-958, M/A-COM Linkabit*, Sep. 1985.

[14] Mills D. L., "Network Time Protocol (Version 1) - specification and implementation", *DARPA Network Working Group Report RFC-1059, University of Delaware*, July 1988.

[15] Mills D. L., "Network Time Protocol (Version 2) - specification and implementation", *DARPA Network Working Group Report RFC-1119, University of Delaware*, Sept. 1989.

[16] Mills D. L., "Network Time Protocol (Version 3) - Specification, Implementation and Analysis", *DARPA Network Working Group Report RFC-1305, University of Delaware*, Mar. 1992.

[17] Mills D. L., "Network Time Protocol (Version 4) - Protocol and Algorithms Specification", *DARPA Network Working Group Report RFC-5905, University of Delaware*, June 2010.

[18] Mills D. L., *Computer Network Time Synchronization: the Network Time Protocol on Earth and in Space*, Second Edition, CRC Press, 2011.

[19] Mills D. L., "Simple Network Time Protocol (SNTP)", *DARPA Network Working Group Report RFC-1361, University of Delaware*, Aug. 1992.

[20] Mills D. L., "Simple Network Time Protocol (SNTP)", *DARPA Network Working Group Report RFC-1769, University of Delaware*, Mar. 1995.

[21] Mills D. L., "Simple Network Time Protocol (SNTP)", *DARPA Network Working Group Report RFC-2030, University of Delaware*, Oct. 1996.

[22] Mills D. L., "Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI, RFC 4430", *University of Delaware*, Jan. 2006.

[23] IEEE *Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, IEEE Standard1588-2002, 2002.

[24] IEEE *Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, IEEE Standard1588-2008, July. 2008.

[25] Eidson J., *Measurement Control and Communication Using IEEE 1588*, London, UK, Springer, 2006.

[26] Postel J., "User Datagram Protocol", *DARPA Network Working Group Report RFC-768, USC Information Sciences Institute*, Aug. 1980.

[27] Simonsen K., "Character Mnemonics and Character Sets", *DARPA Network Working Group Report RFC-1345*, June 1992.

[28] Mills D. L., "The Autokey security architecture, protocol and algorithms. Electrical and Computer Engineering Technical Report 06-1-1", *NDSS*, Jan. 2006.

[29] u-blox 5 ROM-based GPS receiver module, *NEO-5Q: GPS receiver module with KickStart*, in URL: http://www.u-blox.com/en/neo-5q.html, accessed on July 18, 2012.

[30] Rf Telecommunication Electronics, *RFPhone*, in URL: www.rfphone.com, accessed on July 18, 2012.
[31] u-blox AG. "*NEO-5 u-blox 5 GPS Modules Data Sheet,*" Document number: GPS.G5-MS5-07025-B3, 2009.Available online at: http://www.u-blox.com/images/downloads/Product_Docs/NEO-5x_DataSheet_%28GPS.G5-MS5-07025%29.pdf.
[32] *NMEA 0183 standard for interfacing marine electronics devices*, National Marine Electronics Association, 1983.
[33] *NMEA 2000®standard for interfacing marine electronics devices*, National Marine Electronics Association, 2000.
[34] u-blox AG. "*u-blox 5 Receiver Description Including Protocol Specification*", Document number: GPS.G5-X-07036-G,2009 Available online at: http://www. u-blox.com/images/downloads / Product_Docs/u-blox5_Protocol_Specifications (GPS.G5-X-07036).pdf.
[35] MATLAB CENTRAL, *TCP/UDP/IP Toolbox 2.0.6 - File Exchange - MATLAB Central,* in URL: http://www.mathworks.com/matlabcentral/fileexchange/345, accessed on July 18, 2012.
[36] Microsoft Support, *Registry entries for the W32Time service*, in URL: http://support.microsoft.com/kb/223184, accessed on July 18, 2012.
[37] The world's foremost network protocol analyzer, *Wireshark-Go deep,* in URL: http://www.wireshark.org/, accessed on July 18, 2012.
[38] virtualbox.org, *VirtualBox*, in URL: http://www.virtualbox.org/, July 18, 2012.

Mohammad Hossein Refan received his B.Sc. in Electronics Engineering from Iran University of Science and Technology, Tehran, Iran in 1972. After 12 years working and experience in industry, he started studying again in 1989 and received his M.Sc. and Ph.D. in the same field and the same University in 1992 and 1999 respectively. He is currently Professor Assistance of the Faculty of Electrical and Computer Engineering, Shahid Rajaee Teacher Training University, Tehran, Iran, and also working with MAPNA Electrical and Control Engineering & Manufacturing Company (MECO), Tehran, Iran. He is the author of about 50 scientific publications on journals and international conferences. His research interests include GPS, DCS, and Automation System.

Hossein Valizadeh received the B.Sc. degree in electrical engineering from Karaj Islamic Azad University (KIAU), Karaj, Iran in 2008. He also got his M.Sc. degree in Electrical Engineering in Shahid Rajaee University (SRTTU), Tehran, Iran in 2012. His research interests are in the area of Digital Electronic, Microcontrollers and distributed control systems (DCS).