Time synchronization in switched Ethernet

Øyvind Holmeide and Tor Skeie

Abstract - Achieving accurate time synchronization in a switched Ethernet infrastructure is not an easy task due to the variable latencies through the Ethernet switches. The switch latency depends on the network load and the switch architecture. However, this problem can be solved if the time servers are integrated in the Ethernet switches. This article describes the basic principles of the well established internet SNTP/NTP time protocol, and how this protocol can be implemented in an Ethernet switch and in an Ethernet enabled end node in order to achieve a timing accuracy better than one microsecond (us) regardless of the network load.

Index terms - Time synchronization, switched Ethernet, SNTP/NTP, extreme accuracy.

I. INTRODUCTION

Distributed systems with real time requirements have often been based on fieldbuses as the communication solution, but switched Ethernet is now a preferred alternative for many applications due to the never ending price decrease driven by the office Ethernet marked, high bandwidth, priority features (e.g. VoIP) and the availability of Ethernet switches and Ethernet enabled products fulfilling industrial environmental requirements. However, the switch latency will vary depending on the switch load. A variable switch latency means that raw data sent from two different data acquisition nodes to the same receiving node may be delayed differently. This is a problem, but can be solved if the raw data are time stamped. The receiving node can resample the incoming raw data based on the time stamps, and raw data from several sources can be correctly compared. This requires time synchronization.

The timing accuracy that can be achieved in a LAN based on switched Ethernet, where time synchronization data is distributed via this infrastructure, depends on two factors:

1. Time stamping of incoming and outgoing time packets. Time stamping shall preferably be performed at lowest possible level in the OSI protocol stack in order to avoid the variable latency through the protocol stack.
2. Variable network latency. The switch latency depends on the network load, drop link speed, packet sizes and the switch architecture

Øyvind Holmeide is president in OnTime Networks, a spin-off company from ABB that develops deterministic industrial Ethernet switches. Øyvind can be contacted at oeyvind@ontimenet.com.

Tor Skeie is a senior scientist at ABB Corporate Research. Tor can be contacted at tor.skeie@no.abb.com.

Which protocol that is used for time synchronization is of less importance as long as the above listed challenges are handled properly. We do, however, recommend SNTP/NTP as the time synchronization protocol since this is an open internet standard with few limitations. Source code for this protocol is also available.

II. SNTP/NTP

The Simple Network Time Protocol (SNTP), RFC2030, and the Network Time Protocol (NTP), RFC1305, are popular and well established protocols for distribution of time in a IP based network. The SNTP contains only a subset of the NTP functionality. E.g. advanced filtering techniques are not part of a typical SNTP implementation. These filtering techniques are used in order to cope with variable latency through the network elements between the server and the client and the receive and transmit protocol stacks of the server and the client. Both the SNTP and the NTP servers use 123 as their UDP port number. The SNTP/NTP server holds the system reference clock. This clock is often based on GPS, a radio source or a free running clock as its time base.

A. SNTP/NTP time updates

An SNTP/NTP server/client implementation is either based on unicast or multicast distribution of time.

1) Unicast time updates:

Unicast time updates are initiated by a time request sent from the client to the server. The server sends back a time reply. The time client generates a reference time stamp, T_0, and puts this time stamp in the time request packet. The server copies the reference time stamp from the time request into the time reply. This time stamp is used for pairing the time request and the time reply. The client stores two other time stamps based on its local clock. I.e. the local time when time request is sent from the client, T_1, and the local time when time reply is received at the client, T_4. The server generates a time stamp when the time request is received, T_2, and when the time reply is sent, T_3. These two time stamps are generated based on the server clock, which is the system reference clock. The time stamps are put into the time reply. The client can calculate the propagation delay, t_{pd}, and the local clock offset, t_{offset}, based on these four time stamps according to the following formulas:

$$t_{pd} = ((T_4-T_1) - (T_3-T_2))/2$$
$$t_{offset} = ((T_2 - T_1) + (T_3 - T_4)) / 2$$

The client decides how often it wants to perform time updates.

2) Multicast time updates:

SNTP/NTP multicast time updates are based on time reply packets sent on regular intervals to an IP multicast group address. A multicast server expects no requests from the clients. A multicast client join the pre-defined IP multicast group, and listens for time reply packets. The clients are not able to calculate the propagation delay, t_{pd}, based on this method unless the multicast server also accept unicast time request.

B. Time stamping

The timing accuracy depends on where the time stamping of incoming and outgoing time packets are performed, and the variation in the network latency (see chapter III). Time stamping can be performed in the SNTP/NTP application layer, Ethernet driver level (software) or Ethernet data link/physical layer (hardware).

1) Time stamping in the SNTP/NTP application layer

Most SNTP/NTP implementations perform all time stamping in the application layer, see Figure 1 for details. This means that the timing accuracy that can be achieved at the clients will suffer from the variable latency through the UDP/IP protocol stack. This latency depends on the UDP/IP implementation, platform load and the Real Time Operating System (RTOS) implementation. The impact of this jitter can be reduced by using the NTP or similar filtering techniques.

The timing accuracy that can be achieved at the time clients by performing time stamping at application layer will typically be in milliseconds range even though NTP or similar filtering is used, and a direct wire (cross-over cable) are used between the server and the client.

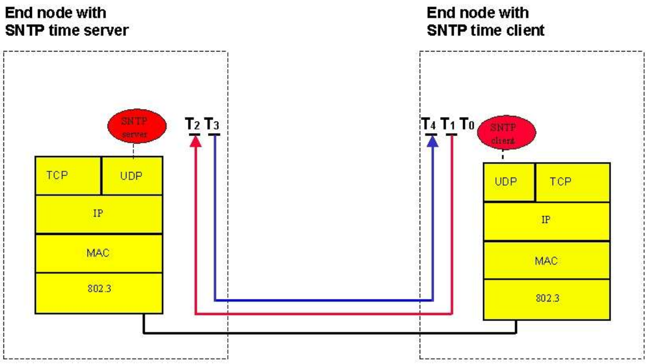

Figure 1, Time stamping at application layer

The server adds both the receive time stamp of the time request, T_2, and the transmit time stamp of the time reply, T_3, in the time reply packets. Thus, the T_3 timestamp is put into the packet before the packet is sent.

2) Time stamping at the Ethernet driver

The timing accuracy can be significantly improved if the time stamping is performed at the Ethernet software driver level. The receive time stamps, i.e. T_2 and T_4, are performed in the Ethernet Interrupt Service Routine (ISR), while the transmit time stamps of the client, T_1, is performed in the Ethernet send routine of the Ethernet driver. The T_3 time stamp is somewhat more tricky, since this time stamp is already included in the reply packet that is generated in the application layer. How to achieve an accurate T_3 time stamp is further discussed in chapter IV.

It is possible to store the server clock when the time reply was sent from the server, T_3, and send this time stamp in a second packet later. However, this approach will be in conflict with existing patents, and not compliant with the SNTP/NTP standard.

Transmit time stamping can be very accurate, while the corresponding receive time stamps may suffer from jitter in the interrupt latency. This jitter depends on the RTOS implementation and the platform load. An alternative software approach that can be used in order to overcome this problem is to implement an RTOS independent ISR that performs the time stamping of incoming time packets.

The timing accuracy that can be achieved at the time clients by performing time stamping at the Ethernet driver level will typically be in the order of a few tens of microseconds to 100 microseconds even though NTP or similar filtering techniques are used, and a direct wire (cross-over cable) are used between the server and the client.

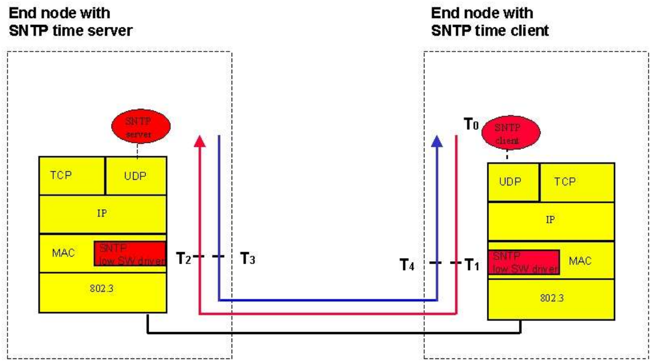

Figure 2, Time stamping in Ethernet level

3) Time stamping at the Ethernet data link/physical layer

The latency through the protocol stacks can be removed if time stamping is performed in the data link/physical layer. The time stamping can be implemented in HW in the Ethernet controller, or in e.g. a separate FPGA that is interfacing the Media Independent Interface (MII) between the Ethernet controller and the Ethernet PHY chip. Time synchronization can be extremely accurate if time stamping is performed in HW. The accuracy at the time clients can be better than one microsecond if this method is used and a direct wire is used between the server and the client. (Achieving an accurate T3 time stamp is also a challenge when using this method, see chapter III for how this can be done.)

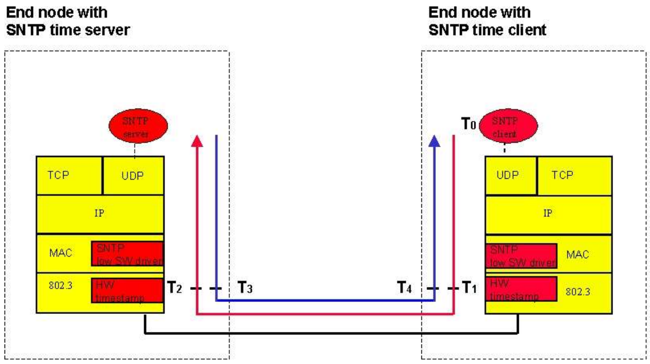

Figure 3, Time stamping at Ethernet data link/physical layer

III. WHY IS SWITCH LATENCY A PROBLEM?

Figure 4 shows a traditional time synchronization implementation, where time packets are sent through a switched Ethernet infrastructure.

The network latency depends on the network load, drop link speed, packets sizes, the switch architecture and the number of switches between the server and the client. The switch latency may vary from a few tens of microseconds up to several milliseconds.

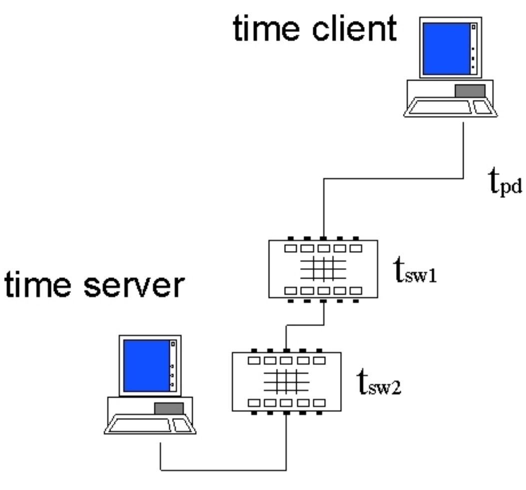

Figure 4, Traditional time synchronization

Most new Ethernet switch designs are based on "store-and-forward" technology. This means that an Ethernet packet must be completely received on an input port before it is checked for bit errors (the switch calculates the Frame Check Sequence (FCS), and compares it with the FCS found at the end of the packet) and forwarded to the respective output port. Thus, the latency depends on the drop link speed and the packet sizes. E.g. an Ethernet packet of maximum Ethernet packet size (1522 bytes) received on a 10 Mbps drop link will be delayed by 1.2 milliseconds due to the "store-and-forward" mechanism. A corresponding 100 Mbps drop link represents a delay of 122 microseconds.

The packet may be further delayed if other packets are queued for transmission on the same output port. Protecting the time packets by using priority does not improve the situation, because the transmission of another packet may already be started when the time packet arrives the output port. Figure 5 shows an example of the Probability Density Function (PDF) of the switch latency taken from actual measurements in case other packets are sent to the same output port as the time packets. The measurement is based on 50% load on the receiving drop link. The PDF has a normal distribution for all switch latency measurements, where extra delay is introduced due to other packets.

Note: extra latency due to other traffic is also a problem in case the switch is based on "cut-through" packet forwarding.

The switch latency may also vary due to general switch load. I.e. packets sent and received on ports not used for time updates. This unwanted switch property is relevant on some switch architectures.

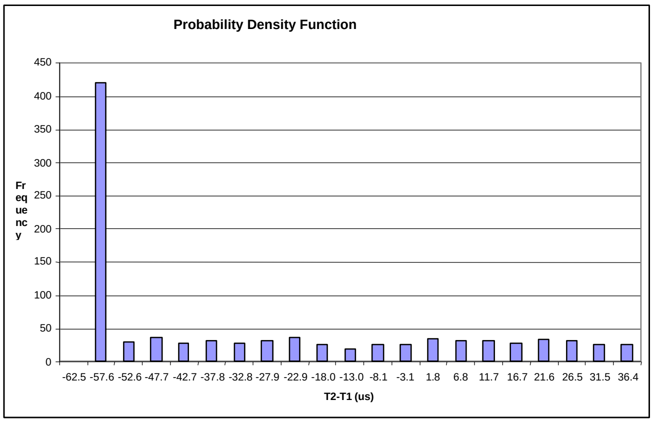

Figure 5, Example of switch latency measurements

The NTP or similar filtering techniques can be used to reduce the impact of variable network latency. However, it is hard to predict the accuracy that can be achieved due to network load dependency, the drop link speed, traffic patterns (e.g. cyclic or burst type of traffic) and the network complexity (i.e. number of switches between the server and the clients). The accuracy may be degraded up to several hundred microseconds depending on these factors.

IV. TIME SERVER IN THE ETHERNET SWITCH

The switch latency problem can be removed if the time server is integrated in the Ethernet switch. There will only be one drop link (direct wire) between the server and the client if this implementation is used[1], and the timing accuracy will be independent of the network load if the time packets are time stamped in hardware. An accuracy better than one microsecond can be achieved at the time client if the time client also performs time stamping in hardware. Figure 6 shows this implementation, where an integrated GPS receiver is used as the time base.

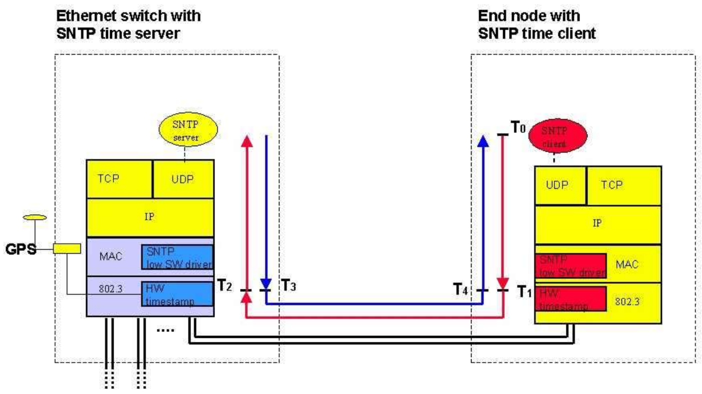

Figure 6, Time server integrated in the switch

The timing accuracy that can be achieved on the client can still be very good, i.e. in the range of 10 microseconds or better, if the client is based on a software implementation, where time stamping is performed in software in the Ethernet driver as described in chapter II.B.2) and the server is integrated in the switch where time stamping is done in hardware.

A time synchronization solution based on these principles does not need the NTP or similar filtering techniques, because the jitter in the protocol and network latencies is very low. Thus, SNTP is a good protocol alternative.

1) Redundancy

A time synchronization solution, where a time server is implemented in each Ethernet switch, is very convenient with respect to redundancy. A time server can act as a time client towards other servers in order to synchronize the server clocks, and true redundancy is achieved if double switched infrastructure is used, i.e. each end node has two network interfaces. A client can perform time updates in both network's.

2) How to achieve an accurate T_3 time stamp[2]

The SNTP time reply contains both the receive time stamp of the time request packet, T_2, and the transmit time stamp, T_3, of the time reply packet. Thus, actual time must be equal to T_3, when the time reply packet hit the drop link. This means that the server must have deterministic access to the media. I.e. T_3 is a future time when this time stamp is put into the time reply, and the packet will then be sent on a pre-determined point in time. The following implementations can be used:

1. The flow control feature of the Ethernet switch, ref. IEEE802.3x, can be used in case of full duplex connectivity in order to hold back the reply packet until absolute time is equal to the T_3 time stamp given in the time reply payload. The time reply packet must be ready for transmission when the flow control is turned off.
2. The back pressure feature of the Ethernet switch can be used in case of half duplex connectivity in order to hold back the reply packet until absolute time is equal to the T_3 time stamp given in the time reply payload. The time reply packet must be ready for transmission when the back pressure signal (JAM patter) is turned off
3. Sending a dummy packet of a given length can be used for both half and full duplex connectivity drop links. The time reply packet must be sent immediately after the dummy packet (only minimum Inter Packet Gap (IPG) between the packets). The time reply packet is granted access to the media at a given time, T_3, by using this technique.

Note: the time reply packet may experience a collision in case of half duplex connectivity, and a re-transmission of this packet will then contain a wrong T_3 time stamp. This must be handled either at the server (i.e. abort re-transmissions of time reply packets) or at the client (the client generates an erasure in case a collision is detected prior to the reception of the time reply packet).

The above described techniques can also be used on a standalone time server.

V. Conclusions

A timing accuracy better than one microsecond (us) independent of the network load is possible to achieve by performing time synchronization via a switched Ethernet infrastructure. This is possible if the servers are implemented in the Ethernet switches and time stamping of incoming and outgoing time packets are performed at lowest possible level in the OSI protocol stack of both the server and the client implementations.

The accuracy will be in the range of 10 microseconds (us) or better if the time client implementation is based on time stamping in the Ethernet software driver.

Time synchronization redundancy can easily be implemented if the servers are implemented in the Ethernet switches.

The well established internet SNTP protocol can be used as the time synchronization protocol. Thus, no advanced filtering techniques are required.

---

[1] Patent pending

[2] Patent pending