Synchronous Ethernet

An introduction to Synchronized Ethernet

Over the past two decades Ethernet has become the dominant technology for data transmission, in particular with telecom and wireless providers, due to its simplicity and low cost. However, the asynchronous nature of Ethernet provides certain transmission challenges.

For example, Time Division Multiplexing (TDM) services such as T1/E1 and SONET/SDH require synchronized clocks at both the source and destination nodes. Similarly, wireless base stations require synchronization to a common clock to ensure a smooth call hand-off between adjacent cells.

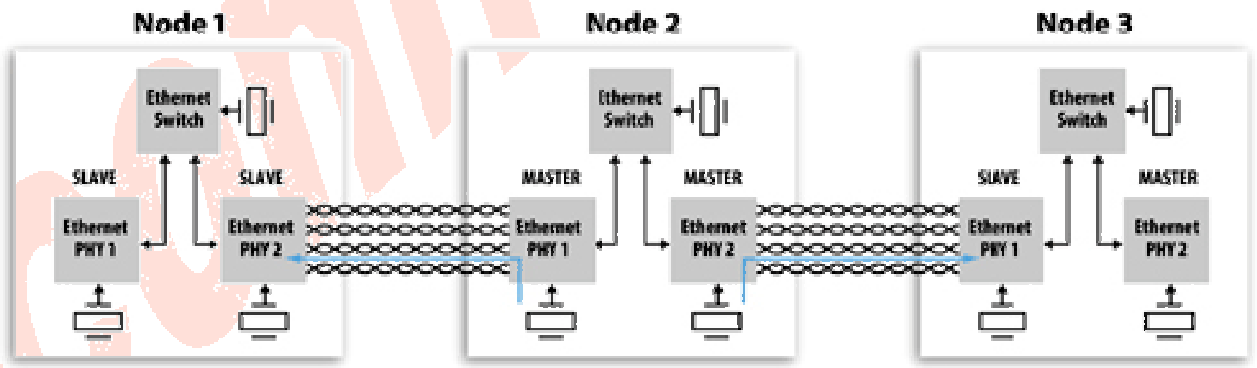
While there are several ways to achieve synchronization over Ethernet, one gaining momentum is Synchronous Ethernet (SyncE). SyncE uses the physical layer interface to pass timing from node to node in the same way timing is passed in SONET/SDH or T1/E1. This gives telecom and wireless providers confidence that networks based on SyncE will be not only cost-effective, but also as highly reliable as SONET/SDH and T1/E1 based networks.

Traditional versus Synchronized Ethernet

Traditional Ethernet was originally intended for transmission of asynchronous data traffic, meaning there was no requirement to pass a synchronization signal from the source to destination. In fact, the old 10Mbps (10Base-T) Ethernet is not even capable of synchronization signal transmission over the physical layer interface because a 10Base-T transmitter stops sending pulses during idle periods.

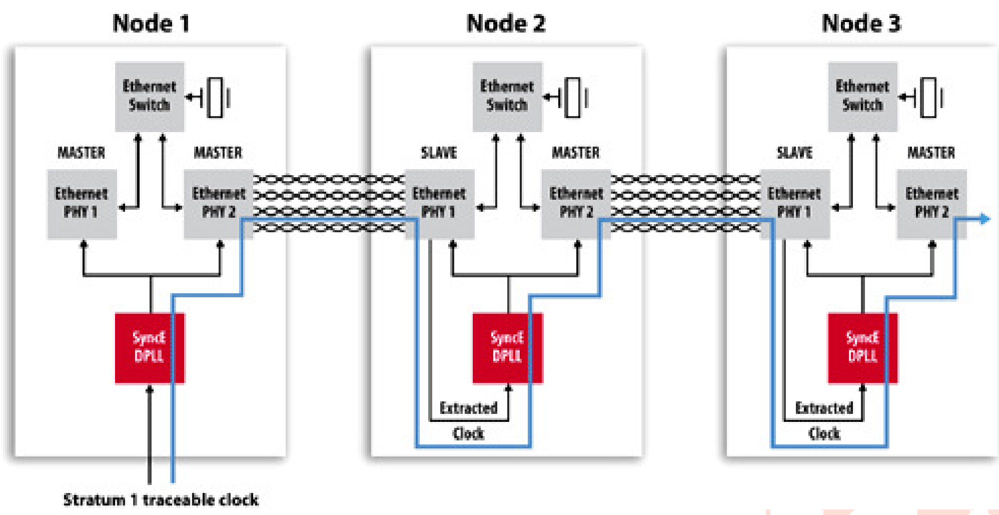
Idle periods in faster Ethernet flavours (100Mbps, 1Gbps and 10Gbps) are continuously filed with pulse transitions, allowing continuous high-quality clock recovery at the receiver.

The master generates a transmit clock locally from free-running crystal oscillator and the slave recovers the master clock from the received data and uses this recovered clock to transmit its own data. Master and slave are determined during the auto-negotiation process. The master is generally assigned randomly using a seed value but it can also be set manually.



Physical layer timing in traditional Ethernet

Synchronization does exist in Ethernet on each hop between two adjacent nodes, but it is not passed from hop to hop. Passing synchronization is relatively simple – take the recovered clock from the node receiving synchronization, and with this clock, feed all nodes that are transmitting synchronization



Physical layer timing in synchronized Ethernet

Synchronized Ethernet requirements

From the discussion so far, it appears that the only requirement for a PLL used in SyncE is to clean jitter from the recovered clock, which can be accomplished with general purpose PLLs. However, the PLL used in SyncE must provide additional functions beyond jitter cleaning.

For example, if the receiving PHY device gets disconnected from the line, the recovered clock frequency will either stop or start to drift depending on the implementation of the clock recovery circuit. The general purpose PLL will pass this big frequency change to the transmitting PHY device. As a result, not only is the transmission of synchronization signal going to fail, but the data transmission could fail as well.

The PLL used in SyncE must be able to detect failure of the recovered clock and switch the PLL to either another good reference in the system or into holdover mode. Requirements for SyncE are outlined in the timing characteristics of synchronous Ethernet equipment clock (ITU G.8262/Y1362) specifications. These specifications are based on ITU-T G.813 specification for SDH clocks.

The major requirements of ITU-T G.8262/Y1362 are:

- **Free-run accuracy:** The accuracy of PLL output when it is not driven by a reference should be equal or better than ±4.6 ppm (part per million) over a time period of one year. This is a very accurate clock relative to the clock accuracy for traditional Ethernet (±100 ppm).
- **Holdover:** The PLL constantly calculates the average frequency of the locked reference. If the reference fails and no other references are available, the PLL goes into holdover mode and generates an output clock based on a calculated average value. Holdover stability depends on the resolution of the PLL averaging algorithm and the frequency stability of the oscillator used as the PLL master clock.
- **Reference monitoring:** The PLL needs to constantly monitor the quality of its input references. If the reference deteriorates (disappears or drifts in frequency), then the PLL raises an alarm (interrupt) and switches to another valid reference.
- **Hitless reference switching:** If the PLL's reference fails, then it will lock to another available reference without phase disturbances at its output.
- **Jitter and wander filtering:** The PLL can be viewed as a jitter and wander filter. The narrower the loop bandwidth, the better the jitter and wander attenuation.
- **Jitter and wander tolerance:** The PLL should tolerate large jitter and wander at its input and still maintain synchronization without raising any alarms.

Precision Time Protocol (PTP) – IEEE 1588

Introduction

Precise time information is especially important for distributed systems in automation technology. With the Precision Time Protocol (PTP) described in IEEE 1588, it is possible to synchronise distributed clocks with an accuracy of less than 1 microsecond via Ethernet networks. The demands on the local clocks and the network and computing capacity are relatively low.

There have previously been different ways to synchronise distributed clocks through a network. The most common of these are the Network Time Protocol (NTP) and the simpler Simple Network Time Protocol (SNTP) derived from it. These methods are widely distributed in LANs (Local Area Networks) or in the Internet and allow accuracies into the millisecond range.

Another possibility is the use of radio signals from GPS satellites. However, this necessitates relatively expensive GPS receivers in every clock as well as the appropriate antennae.

Another solution is to send a high-precision time pulse (e.g. pulse per second signal) to every device on separate lines. However, this entails an enormous additional wiring effort.

This is where the Precision Time Protocol (PTP) described in IEEE 1588 comes in. It has been developed with the following aims:

- Synchronization accuracy in the sub-microsecond range
- Minimum requirements of the processor performance and network bandwidth which enables it to be implemented on simple and low-cost devices
- Low administration effort
- Use via Ethernet networks but also via other networks
- Specification as an international standard

However special attention must be paid to the network infrastructure. Since hubs have almost no influence on the accuracy of the protocol due to their practically constant throughput time, the run times must be taken into account when using switches or routers. PTP measures the run times in the network and measures the clocks accordingly. So run time variations, as they always occur in switches and routers, lead to inaccuracy.

Implementation

In PTP version 1 (IEEE 1588-2002) four types of messages have been defined:

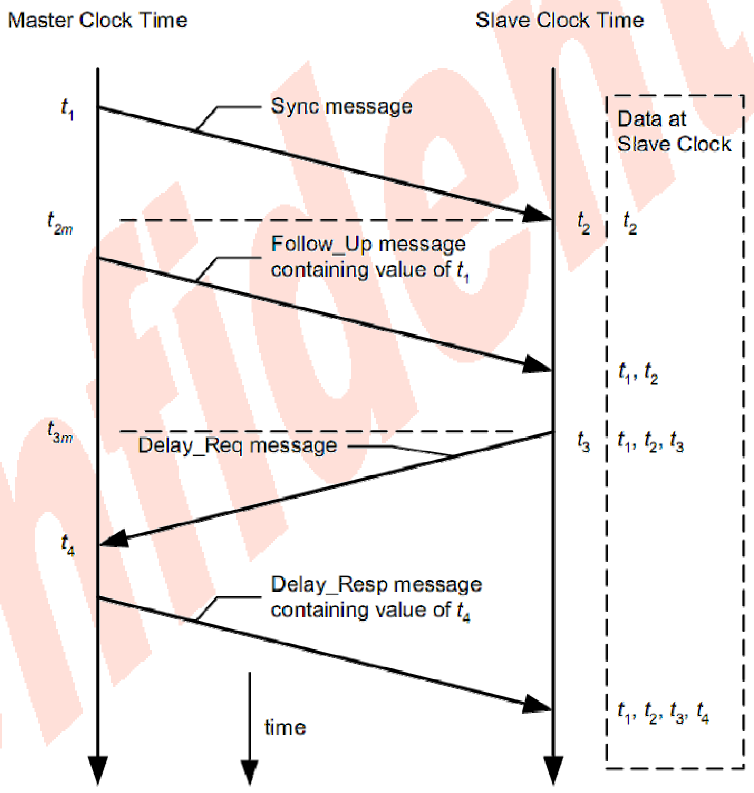
1. Sync	(from master to slaves)
2. Follow Up	(from master to salves)
3. Delay Request	(from slave to master)
4. Delay Response	(from master to slave)

In practice the Follow Up message can be avoided if a timestamp can be directly inserted in the Sync Message.

If we call t_1 the time when the packet leaves the master interface and t_2 the time when it reaches the slave interface. t_3 when it leaves the slave interface before coming back to master and t_4 when it arrives at the master site.

So we can obtain:

$$t_{owd} = \frac{(t_4 - t_1) - (t_3 - t_2)}{2}$$

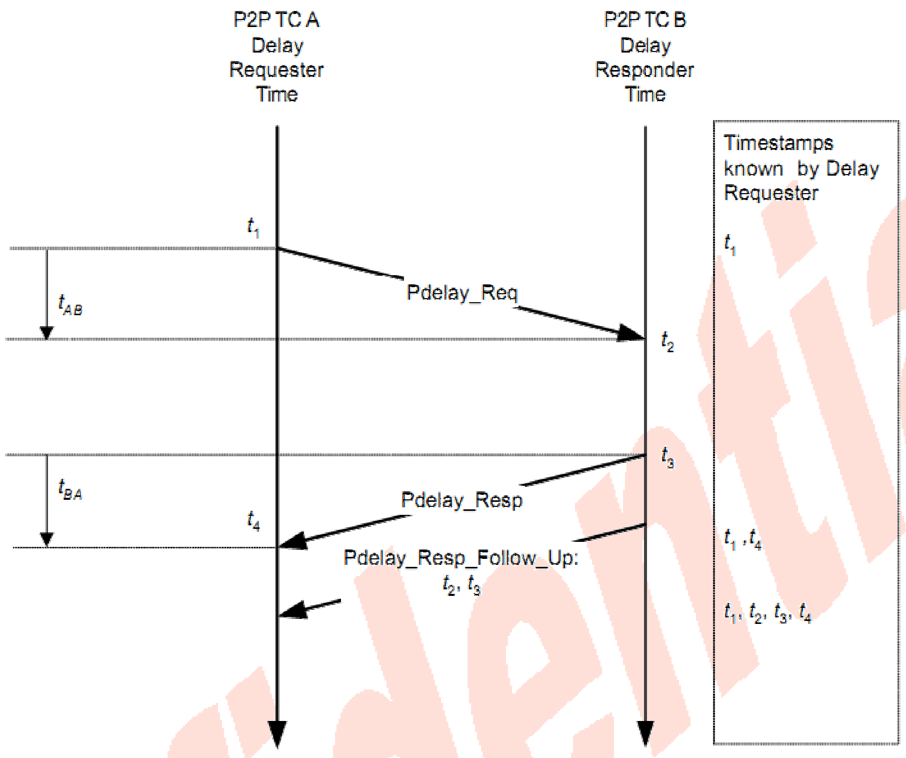


The master to slave offset can then used to phase align a slave to its master and discipline its internal clock.

PTP version 2 (IEEE 1588-2008) splits *timing information* and *master-slave hierarchy* determination: best master selection information is sent in an *Announce* message. This enables a much smaller *Sync* message (44 bytes), with higher Sync message rates using less bandwidth.

Furthemore new *peer delay* messages have been added (*Pdelay_Req*, *PDelay_Resp* and *PDelay_Follow_Up*).

The *peer delay mechanism* measures the port-to-port propagation time, i.e., the link delay, between two communicating ports supporting the peer delay mechanism.
The peer delay mechanism is limited to point-to-point links between two ordinary clocks, boundary clocks, and/or peer-to-peer transparent clocks and so the peer delay messages are not forwarded, contrary to the "Delay request-response" ones.


P2P TC A Delay Requester Time
P2P TC B Delay Responder Time
Timestamps known by Delay Requester
t_1
t_{AB}
Pdelay_Req
t_2
t_3
t_{BA}
Pdelay_Resp
t_4
Pdelay_Resp_Follow_Up: t_2, t_3
t_1
t_1, t_4
t_1, t_2, t_3, t_4

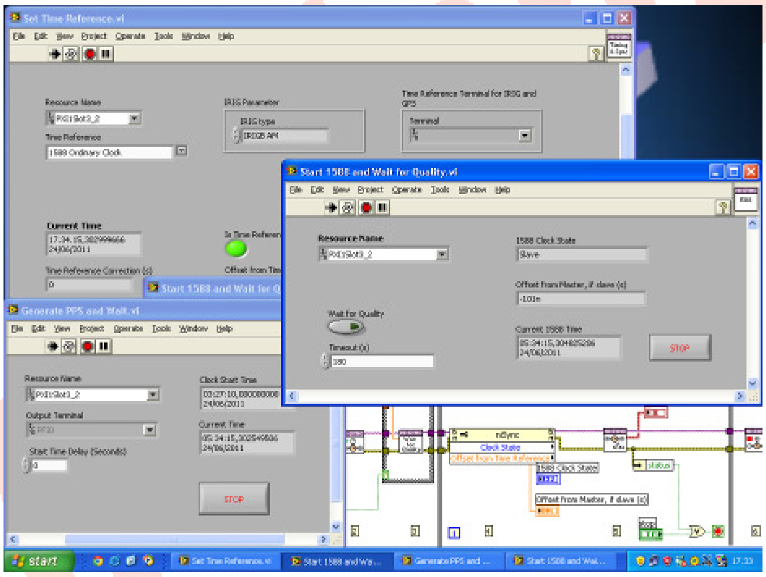
Propagation time computation requires that the two clocks are synchronized to the same timebase, so we can get:

$$t_{prop} = \frac{(t_2 - t_1) + (t_4 - t_3)}{2}$$

Hardware

In order to test our PTP version 2 (IEEE 1588-2008) protocol compliancy we used a PXI-6682H board from National Instruments.

The board can be tuned from GPS and acts as a PTP master or slave and can output its internal PPS via a SMB connector. The programming and configuration of the various functionalities can be done with LabView.



Set Time Reference.vi

Resource Name: PXI1Slot3_2
Time Reference: 1588 Ordinary Clock
IRIG Parameter – IRIG type: IRIGB AM
Time Reference Terminal for IRIG and GPS – Terminal:
Current Time: 17:34:15,302999666 24/06/2011
Time Reference Correction (s): 0

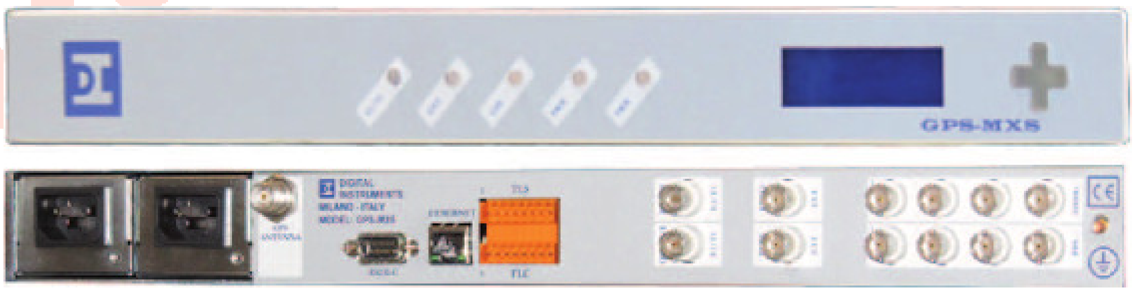
Start 1588 and Wait for Quality.vi

Resource Name: PXI1Slot3_2
1588 Clock State: Slave
Offset from Master, if slave (s): -101n
Wait for Quality
Timeout (s): 180
Current 1588 Time: 05:34:15,304825206 24/06/2011
STOP

Generate PPS and Wait.vi

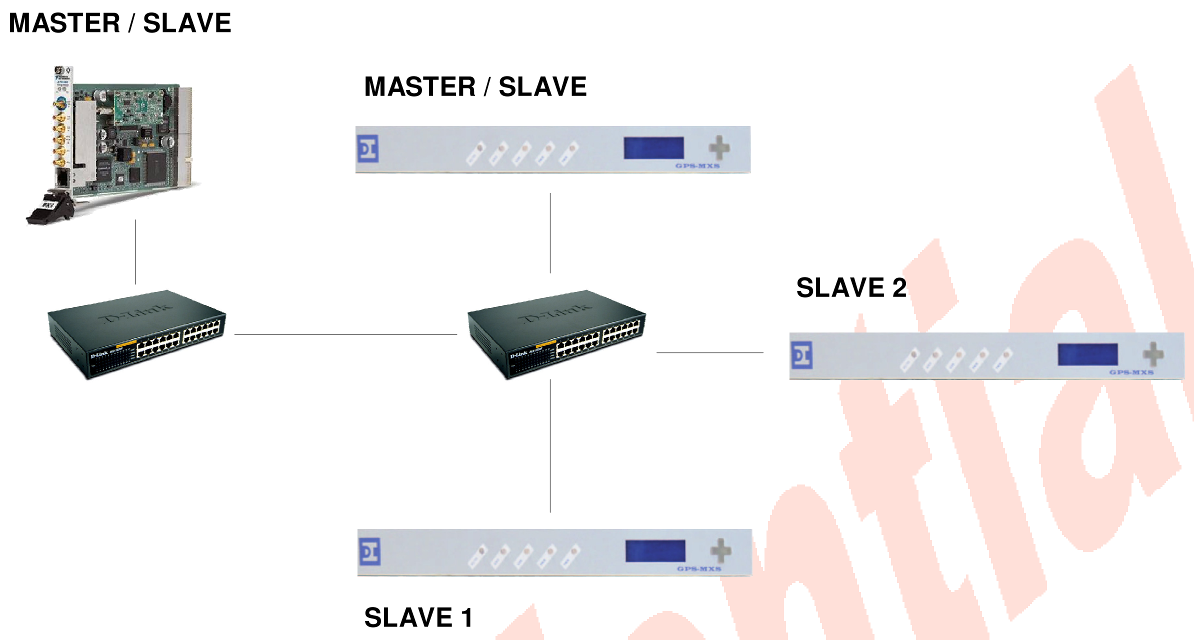
Resource Name: PXI1Slot3_2
Output Terminal: PFI0
Start Time Delay (Seconds): 0
Clock Start Time: 03:27:10,000000000 24/06/2011
Current Time: 05:34:15,302549606 24/06/2011
STOP

We choosed to test first the PTPv2 implementation on the GPS-MXS, but the same applies to the various implementations of the PTP protocol shipped with the other devices built and distributed by Digital Instruments.



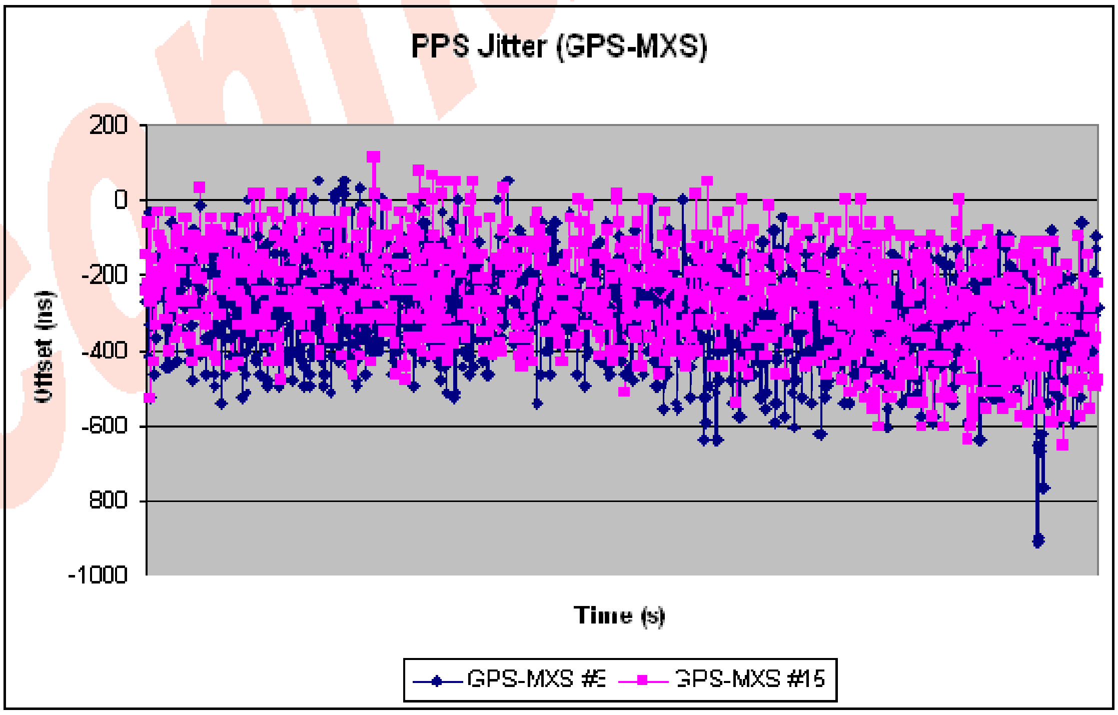
Tests

The network infrastructure consists of two normal switches (not PTP compliant) placed between the master and the slaves and a normal traffic generated mostly by web browsing and data logging.

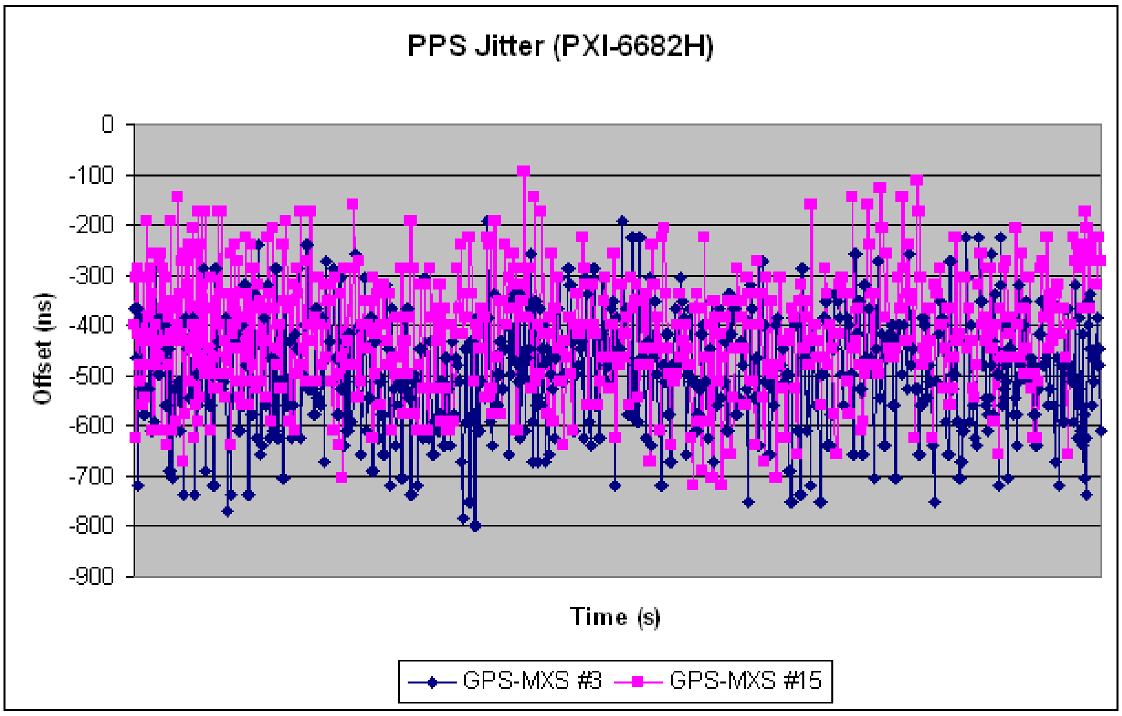


In the following tests the GPS-MXS are locked via GPS or another equivalent mean of synchronization and they are programmed to log the distance of the PPS reconstructed from the network (via the PTP protocol) from the internal PPS that is being outputted on the BNC connectors placed on the rear.

At first we have configured a third GPS-MXS as a PTP master and collected some data on how the PPS is being reconstructed on the two slaves.

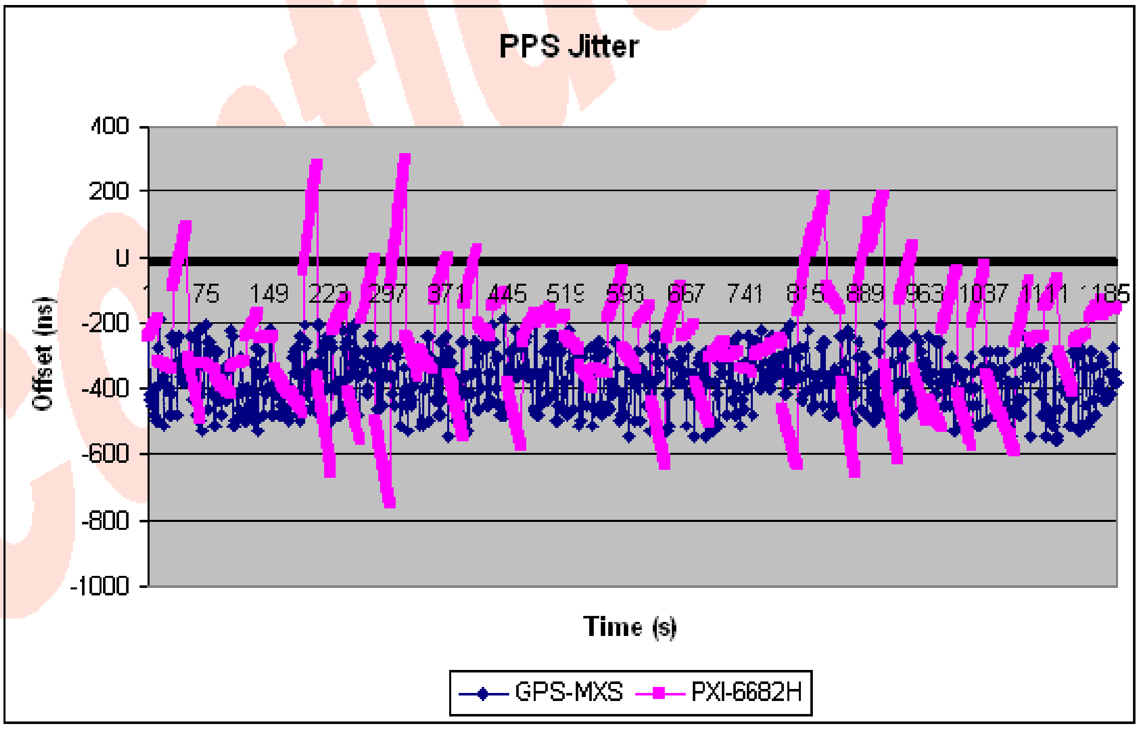


Then we configured the PXI-6682H as a PTP master and the same data acquisition has been done once more.



The results are very similar, with a mean distance less than ±1 μs.

In the end we configured the third GPS-MXS as a PTP master and compared the different PPS reconstructions done by the GPS-MXS and the PXI-6682H.

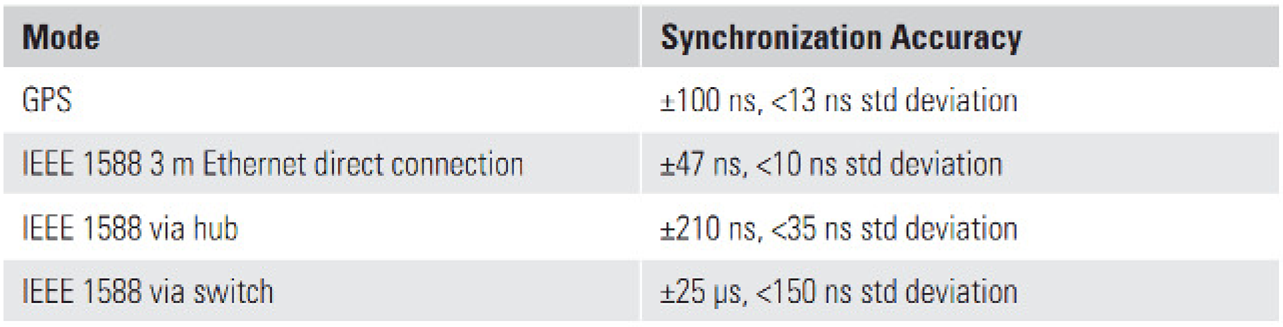


Once again the PPS jitter is less than ±1 μs.

Results

These test have demonstrated the compliancy of our implementation with a third party PTPv2 device and its operability on a normal network, not PTP compliant.

Please be careful when not using a PTP compliant network infrastructure, since no prevision of performance can be done. The following table (from the National Instruments PXI-6682H datasheet) can be taken as reference.



Mode	Synchronization Accuracy
GPS	±100 ns, <13 ns std deviation
IEEE 1588 3 m Ethernet direct connection	±47 ns, <10 ns std deviation
IEEE 1588 via hub	±210 ns, <35 ns std deviation
IEEE 1588 via switch	±25 µs, <150 ns std deviation

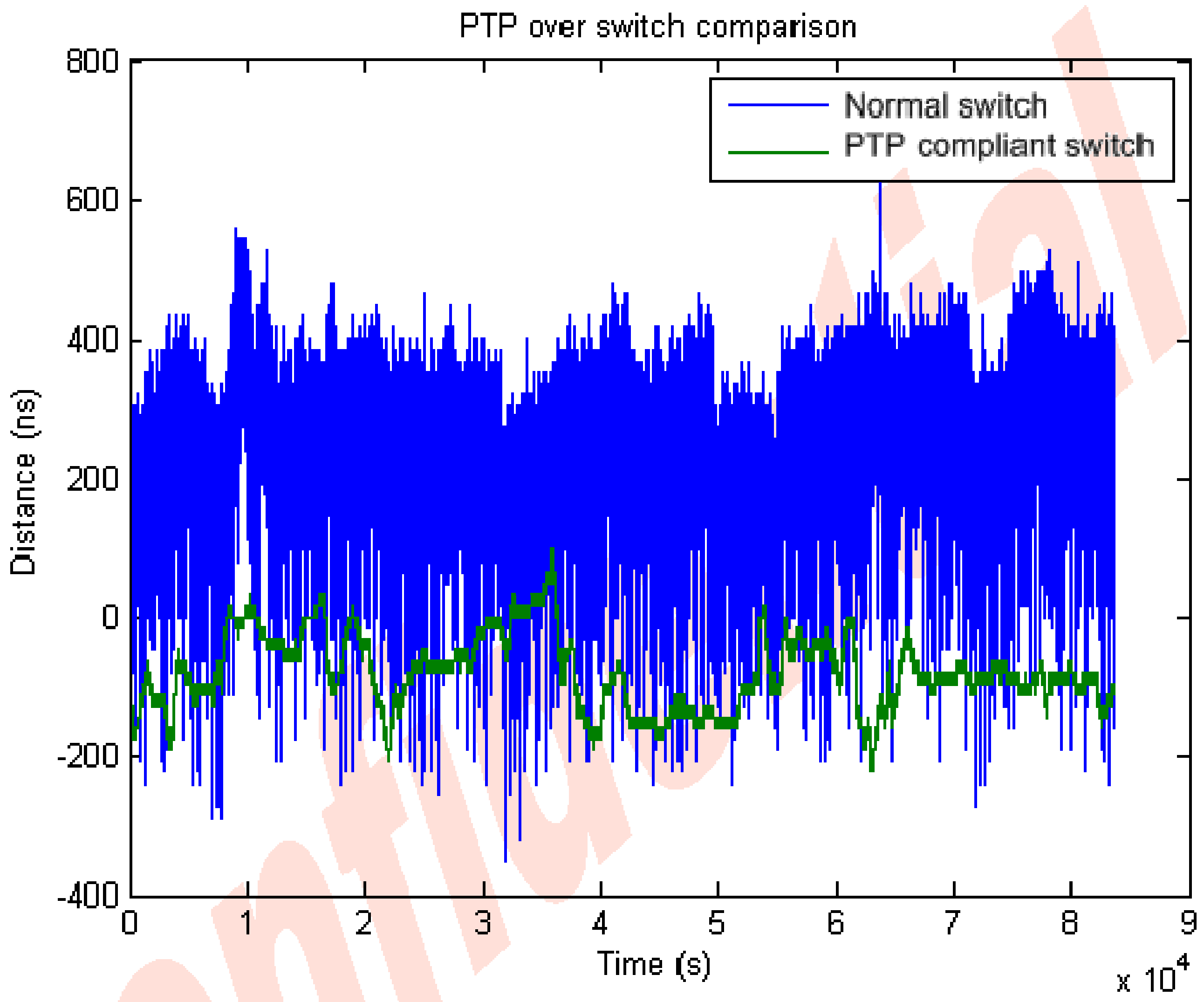
Improvements

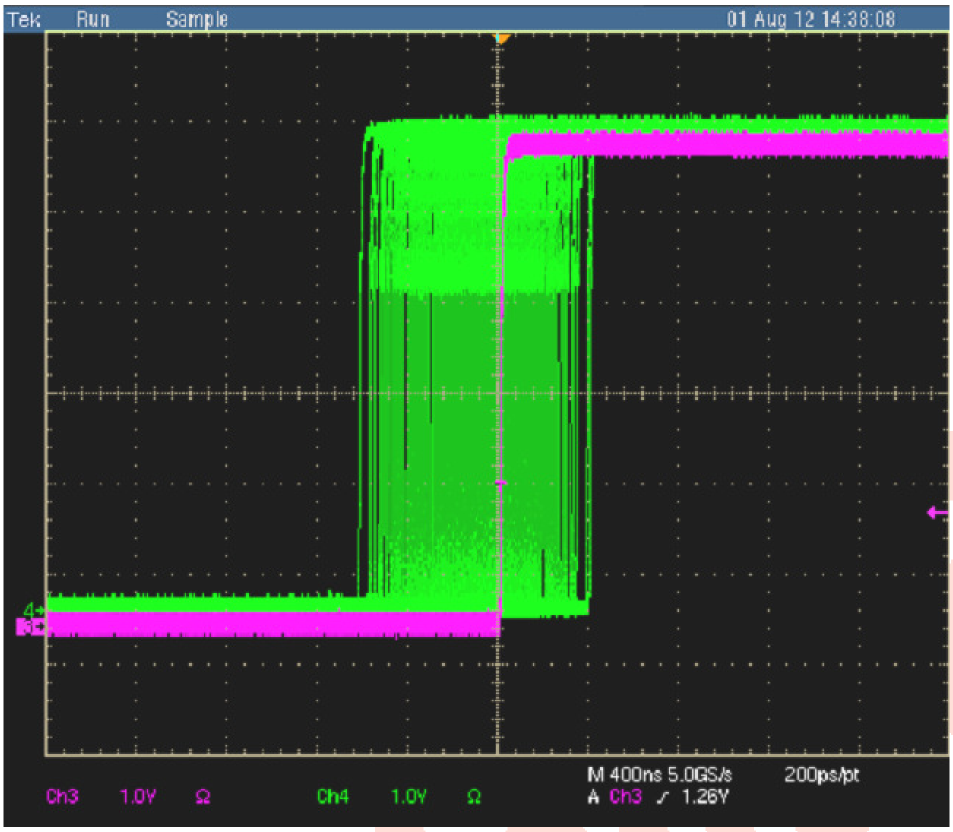
To further improve the precision to less than 1 µs (as low as 100 ns or less) special PTP compliant hardware is needed.

Various brands produce routers and switches that can recognize IEEE 1588 (2002 or 2008) packets, thus eliminating the uncertainty due to propagation time. Care must be taken in order to avoid mixing the two versions of the protocol (PTPv1 and PTPv2), since they are not compatible.
Furthemore there are various types of clock implemented on routers and switches. Notably boundary clocks and transparent clocks. The latters seem to provide slightly better results (most of the time the modality can be configured via software).

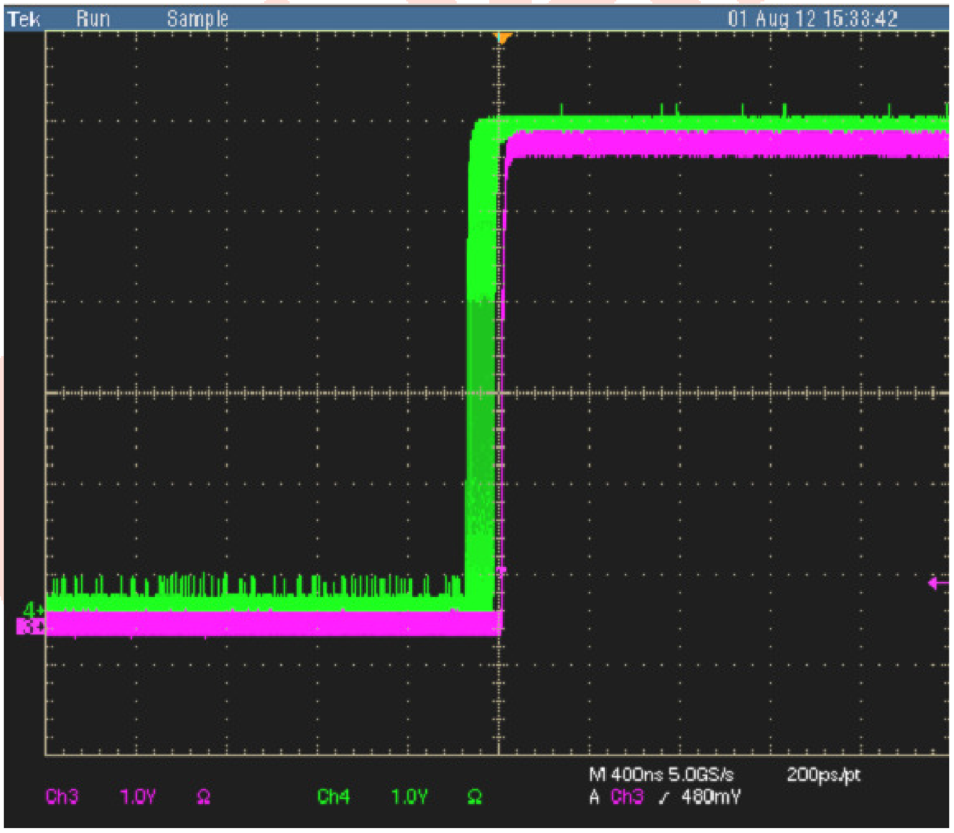
PTP compliant switch

With a Transparent Clock configuration it is possible to lower the jitter and improve the overall performance, as can be seen from the following graphs.





Normal Switch



PTP compliant switch

Equipment

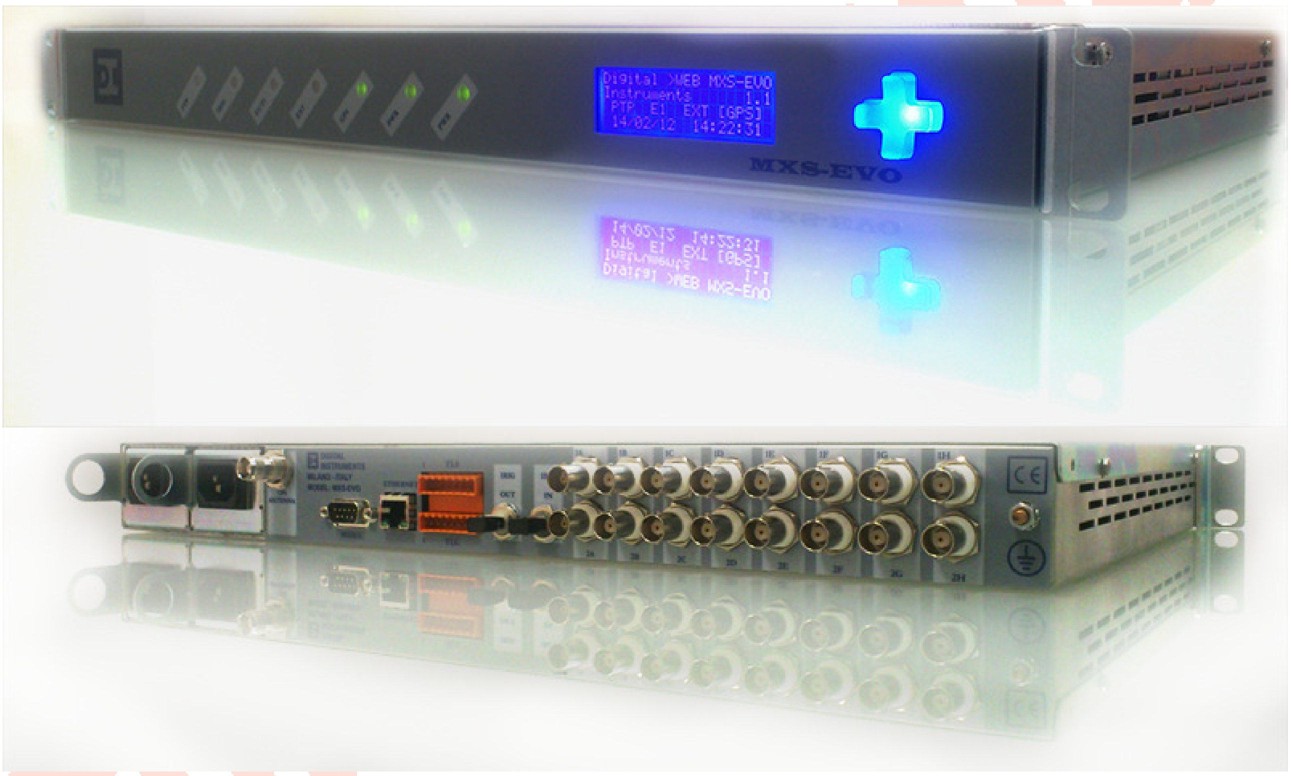
MXS-EVO

MXS-EVO is a multi-output generator for high stability Time & Frequency signals, aimed to synchronization of systems and devices in many areas like Broadcast, Defence, Space, Telecommunication etc.

The unit has 12 programmable outputs designed to make the equipment adaptable to di_erent situation and meet user's needs. Furthermore it has an Ethernet interface for Time Protocol Synchronization (NTP or PTPv2 Grandmaster Clock).

The unit is also capable of two (1 input and 1 output) optical ST connectors designed for IRIG B.

MXS-EVO can get external reference input from GPS receiver, E1/T1, PPS, 1 to 10 MHz analogue, IRIG-B Time code, PTPv2 IEEE 1588-2008, in order to have maximum reliability that is completed by dual independent Power Supply Unit.



<u>*Features*</u>

- Internal high stability OCXO aging rate of ± 1*10^{-10}/day
- 12 channels GPS receiver with automatic tracking and timing error management
- New generation DPLL fast lock with holdover
- Multi reference inputs:
  - GPS
  - E1 (G.703/9) or T1
  - PPS
  - 1, 2, 2.048, 5, 10 MHz
  - IRIG-B Time Code
  - PTPv2 (IEEE 1588-2008)

- 1x Fast-Ethernet interface for NTP and/or PTP synchronization
- 1x Optical multimode I/O via ST connectors
- 12x programmable outputs configurable between:
    - PPS
    - IRIG B DCLS
    - IRIG B AM
    - E1 (G.703/9) / T1 with SSM
    - 2.048 MHz (G.703/13)
    - 10 MHz (Low Phase Noise)
- 2x PSU (AC or DC)

ETS-EVO

ETS-EVO is a very flexible solution to generate ultra-stable Time (PPS, Time Codes, NTP/PTP Serial Time Telegrams, etc…) and Frequency (10 MHz Low Noise and 2.048 MHz square wave output).

The unit is a multi reference input equipment that can accept various reference inputs from (GPS, NTP/PTP as well as IRIG B Time Code both Electrical and Optical).

ETS-EVO has Event Time input capability via Dry Contacts.

Furthermore the unit can be easily remotely managed via SNMP or a user friendly GUI on a web interface.

Features

- Internal high stability OCXO aging rate of ± 1*10^{-10}/day
- 12 channels GPS receiver with automatic tracking and timing error management
- New generation DPLL fast lock with holdover
- Multi reference inputs:
    - GPS

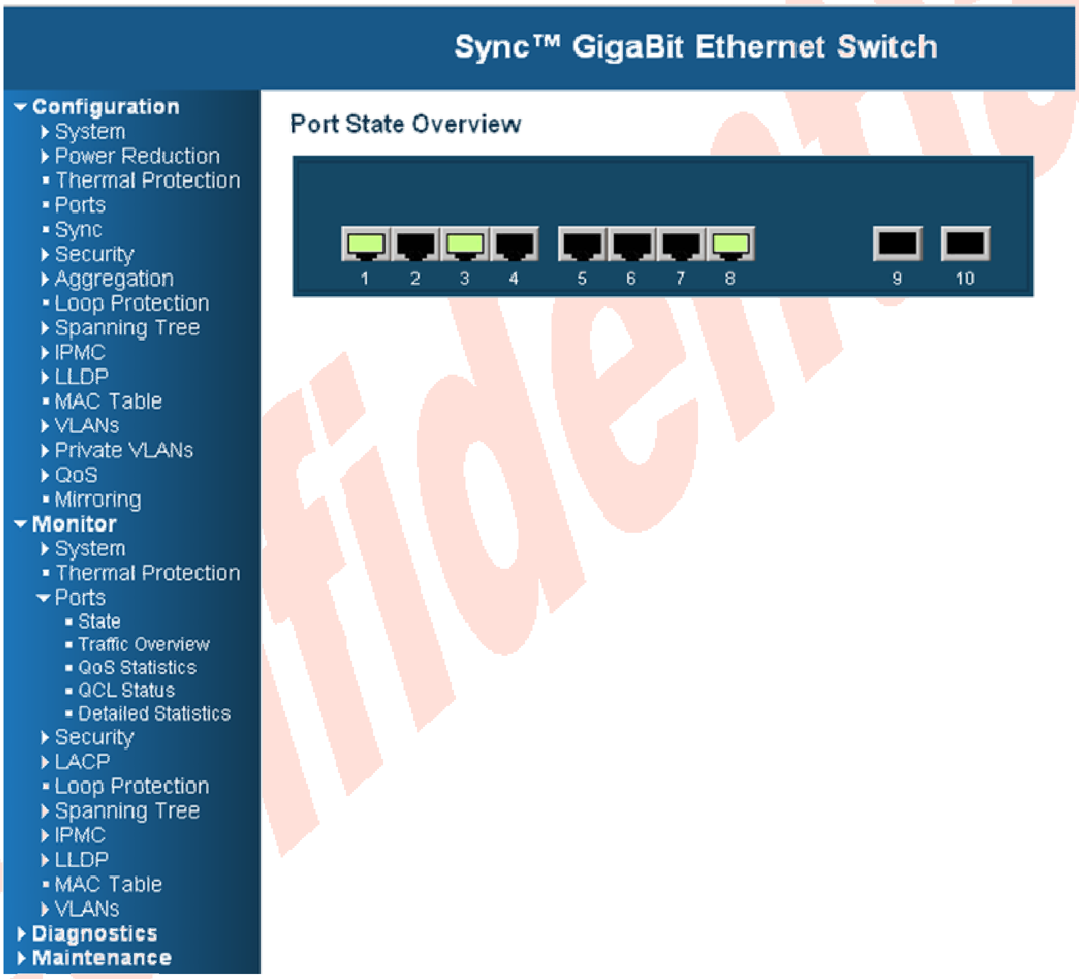
- IRIG-B Time Code
- PTPv2 (IEEE 1588-2008)

- 2x Fast-Ethernet interface for NTP and/or PTP synchronization
- 1x Optical multimode I/O via ST connectors
- 2x PSU (AC or DC)

SYNC-SWITCH

The device is an IEEE 1588-2008 compliant 10-port Gigabit switch capable of acting as a Transparent Clock and, with aid of the Synchronous Ethernet protocol, to achieve synchronization in the ns range.

It is equipped with the latest technology and may be operated via a comfortable web interface.



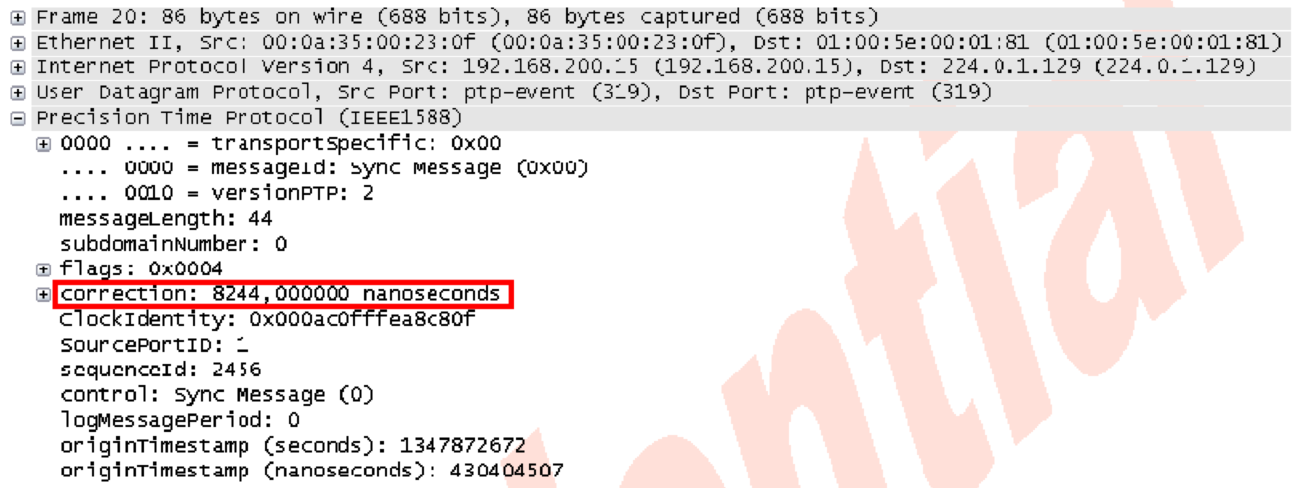
Features

- 8 Gigabit ports
- 2 FSP Ports
- High Stability OCXO
- IEEE 1588-2008 compliant
- End-to-End Transparent Clock
- GPS Radio Receiver for Grandmaster role
- Synchronous Ethernet

The job of a *Transparent Clock* (also referred to as TC) switch is very simple to understand.

It just modifies PTP messages as they pass through the device. Timestamps in the messages are corrected for time spent traversing the network equipment. This approach improves distribution accuracy by compensating for delivery variability across the network (called *Packet Delay Variation* - PDV).

The device does not alter any other message other than *Sync* and *Delay_Req* packets and is completely transparent both to the PTP Master and to the PTP slaves.



```
⊞ Frame 20: 86 bytes on wire (688 bits), 86 bytes captured (688 bits)
⊞ Ethernet II, Src: 00:0a:35:00:23:0f (00:0a:35:00:23:0f), Dst: 01:00:5e:00:01:81 (01:00:5e:00:01:81)
⊞ Internet Protocol Version 4, Src: 192.168.200.15 (192.168.200.15), Dst: 224.0.1.129 (224.0.1.129)
⊞ User Datagram Protocol, Src Port: ptp-event (319), Dst Port: ptp-event (319)
⊟ Precision Time Protocol (IEEE1588)
  ⊞ 0000 .... = transportSpecific: 0x00
    .... 0000 = messageId: Sync Message (0x00)
    .... 0010 = versionPTP: 2
    messageLength: 44
    subdomainNumber: 0
  ⊞ flags: 0x0004
  ⊞ correction: 8244,000000 nanoseconds
    ClockIdentity: 0x000ac0fffea8c80f
    SourcePortID: 1
    sequenceId: 2456
    control: Sync Message (0)
    logMessagePeriod: 0
    originTimestamp (seconds): 1347872672
    originTimestamp (nanoseconds): 430404507
```

Sync message modified by a Transparent Clock

This approach is particularly needed because the time it takes for a network switch to process a packet greatly varies depending on network load.

No Traffic	
Mean:	16.8 µs
Peak to Peak:	310.0 ns
Standard Deviation:	70.1 ns
10% Load	
Mean:	17.9 µs
Peak to Peak:	121.4 µs
Standard Deviation:	11.5 µs
25% Load	
Mean:	19.6 µs
Peak to Peak:	122.6 µs
Standard Deviation:	17.6 µs
50% Load	
Mean:	48.0 µs
Peak to Peak:	122.8 µs
Standard Deviation:	50.9 µs

Packet delay based on network load

Test cases

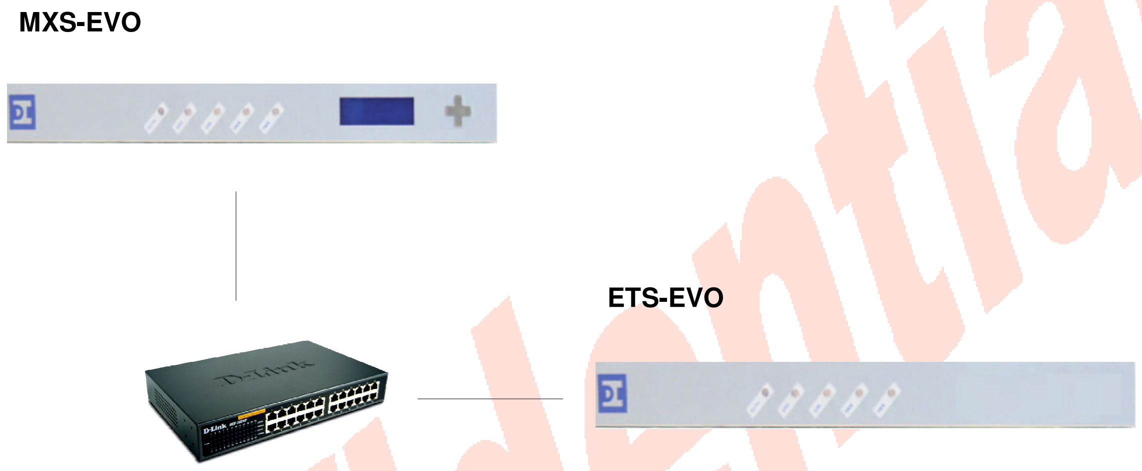
Synchronous Ethernet

Goal

- to test the distribution of a clock over an Ethernet network

Requirements

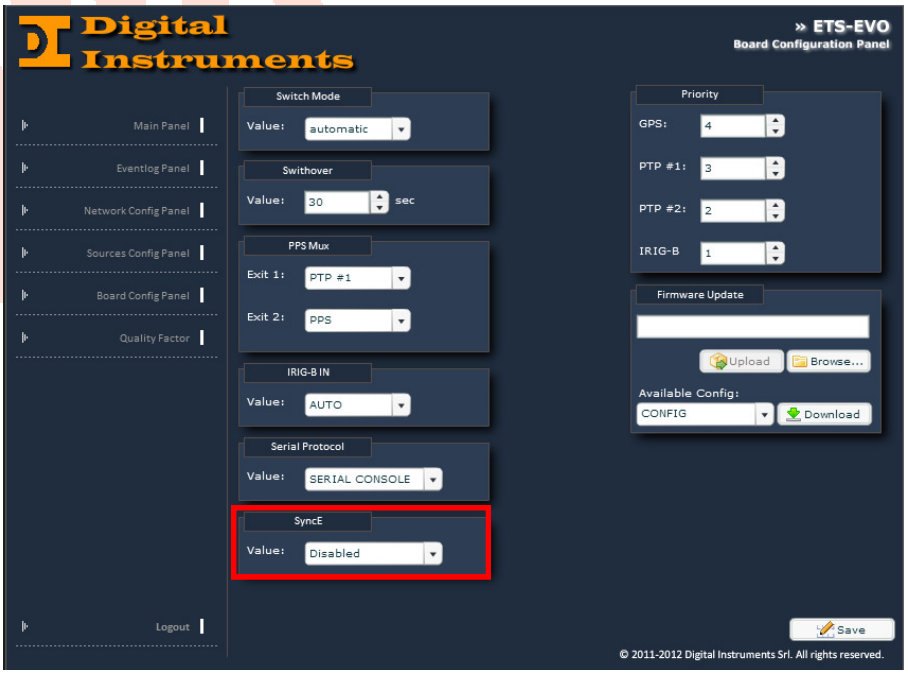
- MXS-EVO
- ETS-EVO
- SyncE compliant network
- Oscilloscope



Test #1

In this test we just want to lock the MXS-EVO with an ETS-EVO via the clock distributed on the Ethernet network and test the 10 MHz outputs.

1. First we should let the ETS-EVO oscillator to free-run (or to lock on any available source) And we must make sure that the *SyncE* protocol is *Disabled*



Switch Mode
Value: automatic

Swithover
Value: 30 sec

PPS Mux
Exit 1: PTP #1
Exit 2: PPS

IRIG-B IN
Value: AUTO

Serial Protocol
Value: SERIAL CONSOLE

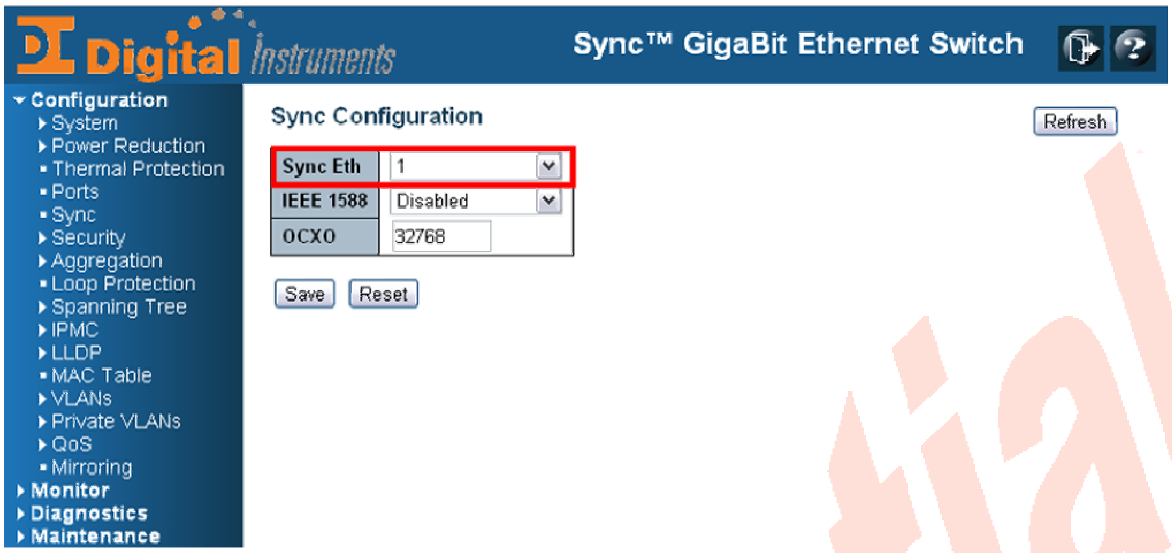
SyncE
Value: Disabled

Priority
GPS: 4
PTP #1: 3
PTP #2: 2
IRIG-B 1

Firmware Update
Available Config: CONFIG

© 2011-2012 Digital Instruments Srl. All rights reserved.

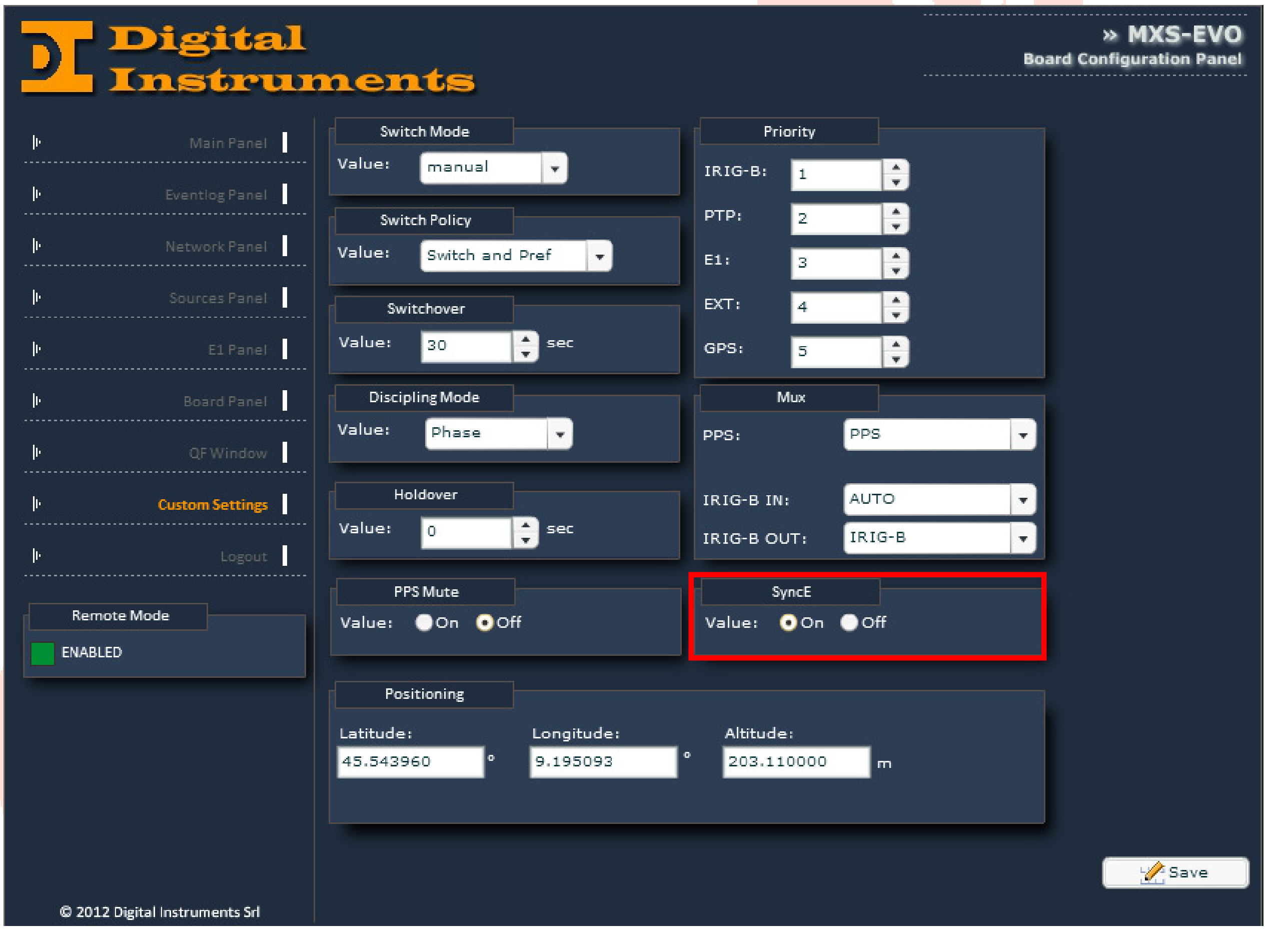
2. Then we need to configure the switch to select the proper *Sync Eth* port that is physically connected to the *master* (ETS-EVO)



Sync Configuration

Sync Eth	1
IEEE 1588	Disabled
OCXO	32768

3. Finally we must enable the *SyncE* protocol on the MXS-EVO



Remote Mode: 🟩 ENABLED

Switch Mode — Value: manual

Switch Policy — Value: Switch and Pref

Switchover — Value: 30 sec

Discipling Mode — Value: Phase

Holdover — Value: 0 sec

PPS Mute — Value: On [✓ Off]

Priority — IRIG-B: 1; PTP: 2; E1: 3; EXT: 4; GPS: 5

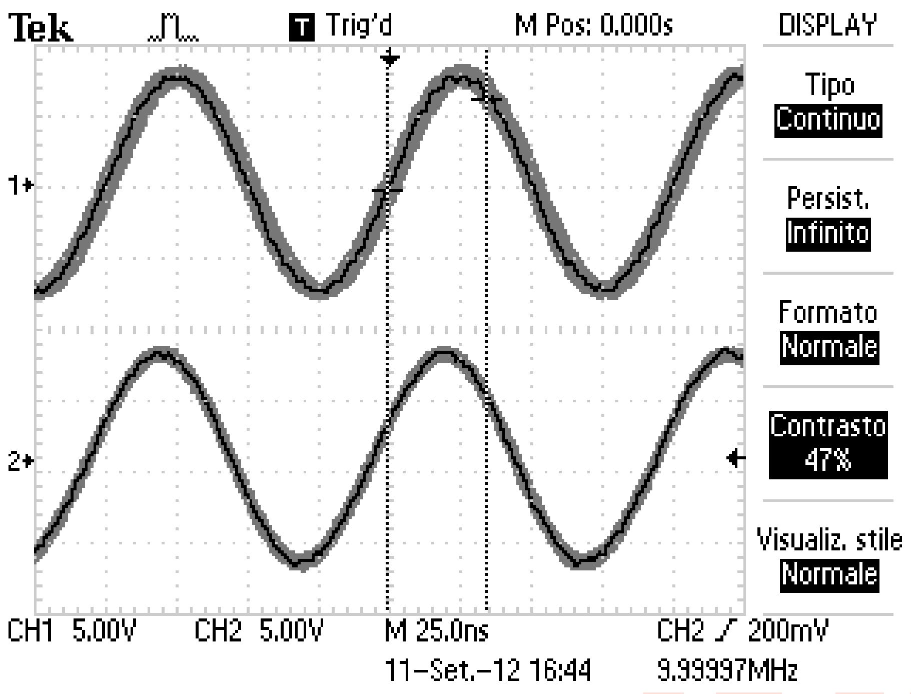
Mux — PPS: PPS; IRIG-B IN: AUTO; IRIG-B OUT: IRIG-B

SyncE — Value: [✓ On] Off

Positioning — Latitude: 45.543960 °; Longitude: 9.195093 °; Altitude: 203.110000 m

It is possible to connect the 10 MHz outputs of the ETS-EVO, the SYNC-SWITCH and the MXS-EVO to an oscilloscope.

After a few seconds the network connectors have been plugged the 10 MHz signals should start slowing down and lock.

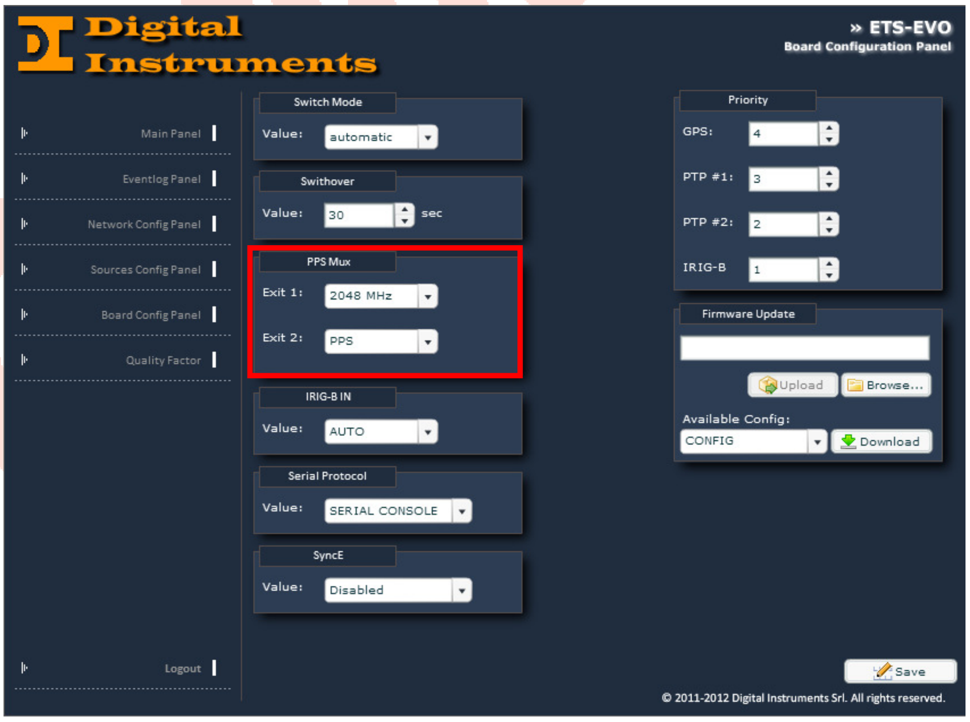


Frequency locked 10 MHz outputs

<u>Test #2</u>

In this test we just want to lock the MXS-EVO with an ETS-EVO via the clock distributed on the Ethernet network and test the 2.048 MHz outputs.

1. Just operate as in Test #1
2. We need to change the output of the PPS connector on the ETS-EVO in order to enable the 2.048 MHz



Switch Mode
Value: automatic

Swithover
Value: 30 sec

PPS Mux
Exit 1: 2048 MHz
Exit 2: PPS

IRIG-B IN
Value: AUTO

Serial Protocol
Value: SERIAL CONSOLE

SyncE
Value: Disabled

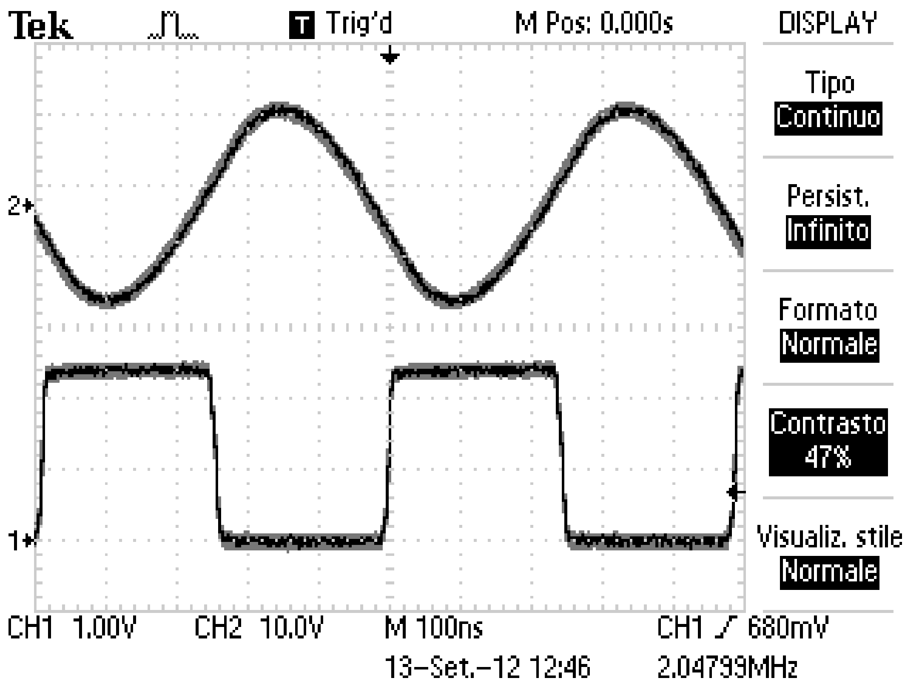
Priority
GPS: 4
PTP #1: 3
PTP #2: 2
IRIG-B 1

Firmware Update

Available Config:
CONFIG

© 2011-2012 Digital Instruments Srl. All rights reserved.

It is possible to connect the 2.048 MHz outputs of the ETS-EVO and the MXS-EVO to an oscilloscope.

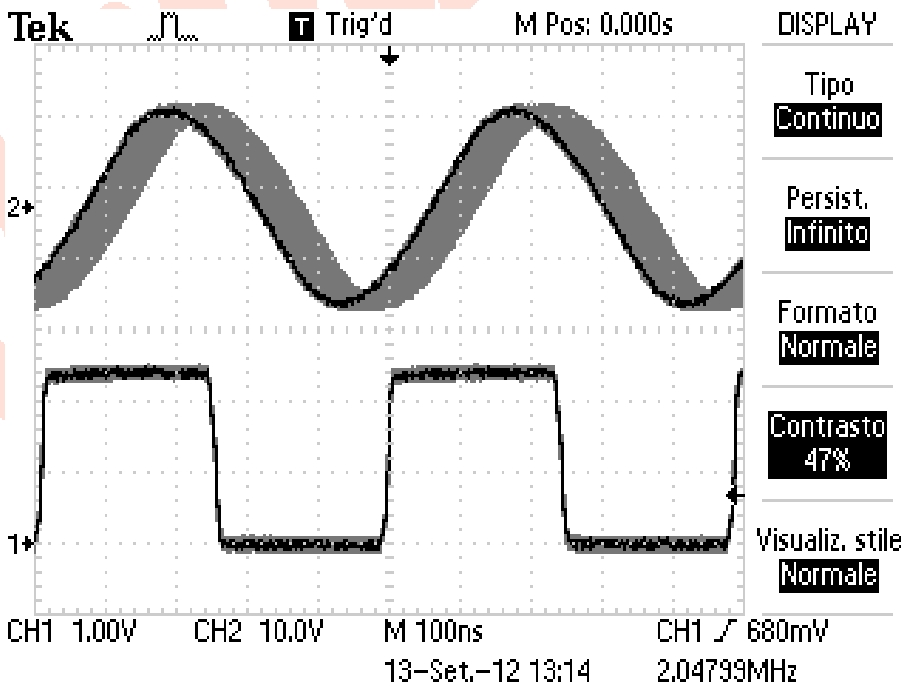


Frequency locked 2.048 MHz outputs

Test #3

In this test we want to to test the holdover capabilities of the Synchronous Ethernet.

1. Just operate as in Test #1 or Test #2
2. After a few minutes of locked status just unplug the network cable and check that the output frequency is still locked, even if probably moving with a slow wonder (can be viewed via the persistence feature of the oscilloscope)



Frequency drift during the holdover

In this case the oscillator moved of 500 ns in 75 seconds with a drift of 6.667 ns/s, equals to 0.00667 ppm

3. Re-plug the network cable and check that the frequency locks again

Test #4

It is also possible to repeat the same tests inverting the roles of the devices.
MXS-EVO can act as a master and ETS-EVO as a slave.
It is also important to change the master port in the switch (or swap the network connectors).

Precision Time Protocol (PTP) – IEEE 1588

Goal

- to test the distribution of a time source over an Ethernet network

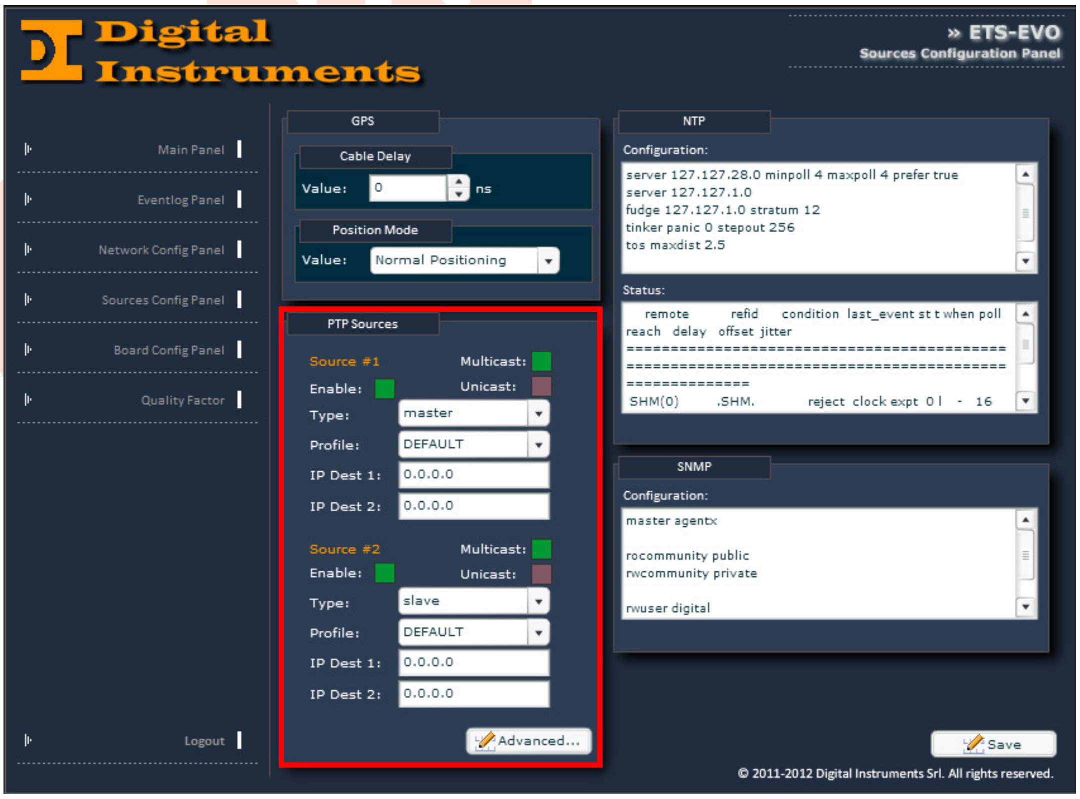
Requirements

- MXS-EVO
- ETS-EVO
- PTP compliant network
- Oscilloscope

Test #1

In this test we want to test the distribution of a time pulse (PPS) over the network.

1. First we should let the ETS-EVO oscillator to lock on any available source (GPS or IRIG-B) Then we must make sure that the *SyncE* protocol is *Disabled* and that the device is configured as *PTP Master*



Value: 0 ns
Position Mode
Value: Normal Positioning

Source #1		Multicast:	🟩
Enable:	🟩	Unicast:	🟥
Type:	master		
Profile:	DEFAULT		
IP Dest 1:	0.0.0.0		
IP Dest 2:	0.0.0.0		
Source #2		Multicast:	🟩
Enable:	🟩	Unicast:	🟥
Type:	slave		
Profile:	DEFAULT		
IP Dest 1:	0.0.0.0		
IP Dest 2:	0.0.0.0		

Configuration:
```
server 127.127.28.0 minpoll 4 maxpoll 4 prefer true
server 127.127.1.0
fudge 127.127.1.0 stratum 12
tinker panic 0 stepout 256
tos maxdist 2.5
```

Status:
```
   remote        refid     condition  last_event st t when poll
reach   delay   offset  jitter
==============================================
==============================================
==============
 SHM(0)        .SHM.          reject  clock expt  0 l   -   16
```

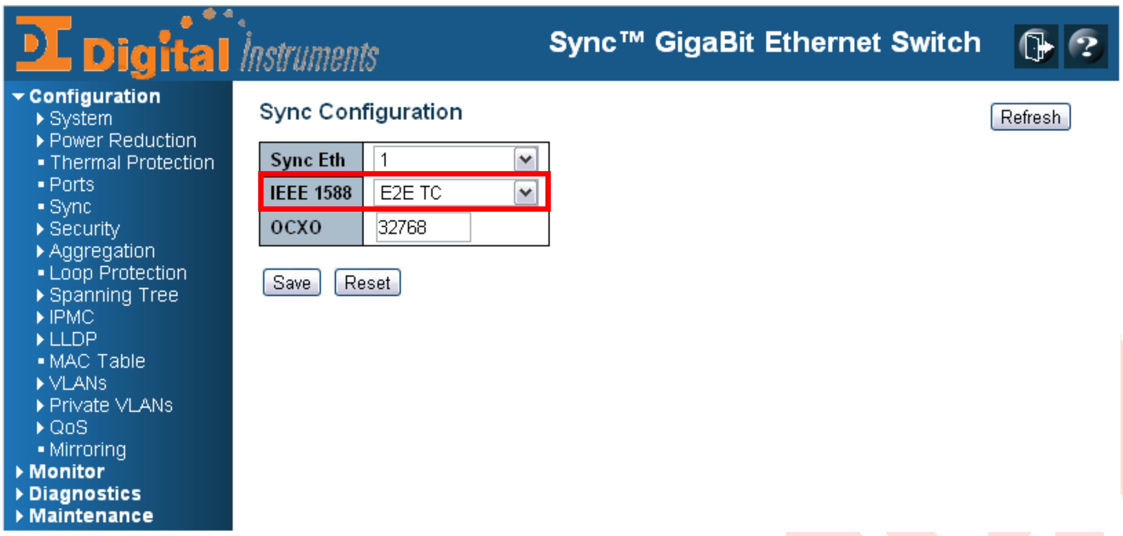
Configuration:
```
master agentx

rocommunity public
rwcommunity private

rwuser digital
```

© 2011-2012 Digital Instruments Srl. All rights reserved.

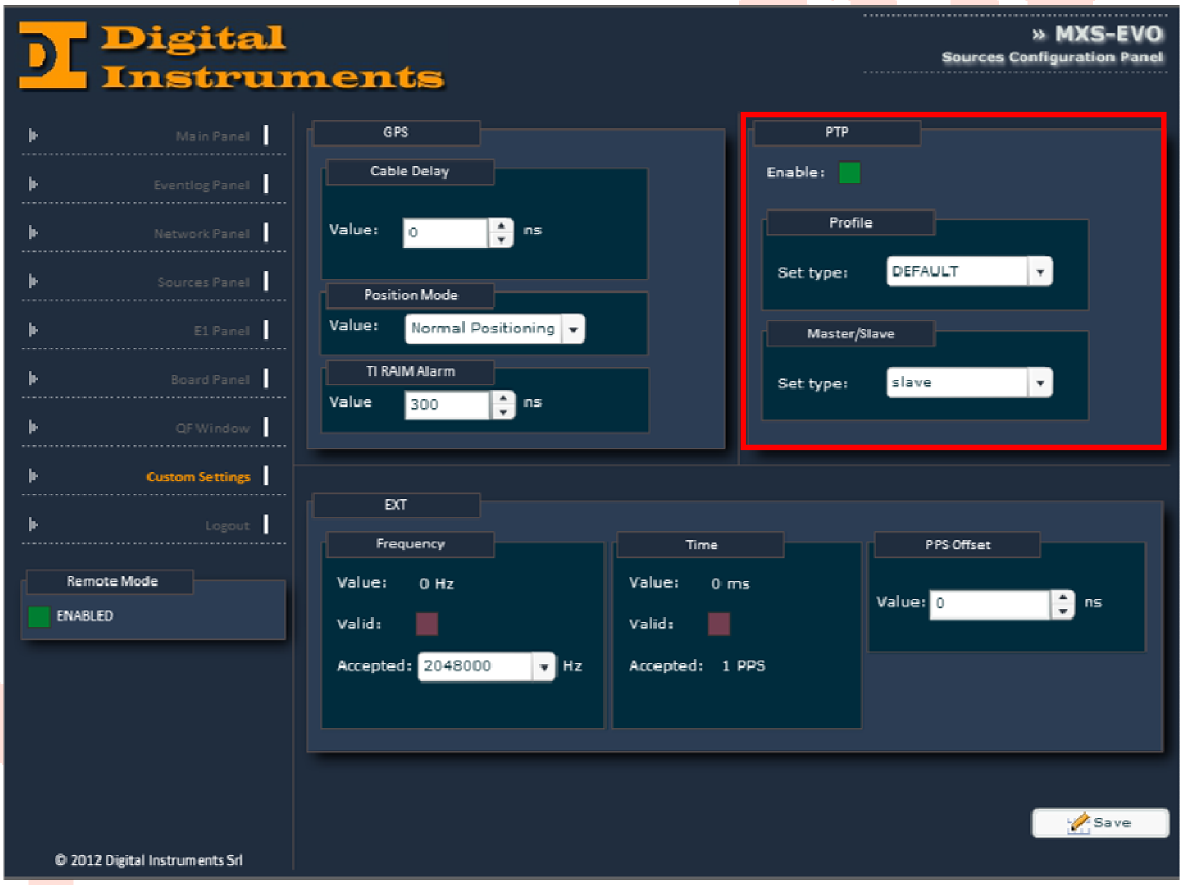
2. Then we must configure the switch as a *PTP Transparent Clock*



Sync Configuration

Sync Eth	1
IEEE 1588	E2E TC
OCXO	32768

3. Finally we must configure the MXS-EVO as a *PTP Slave*



Remote Mode: ENABLED

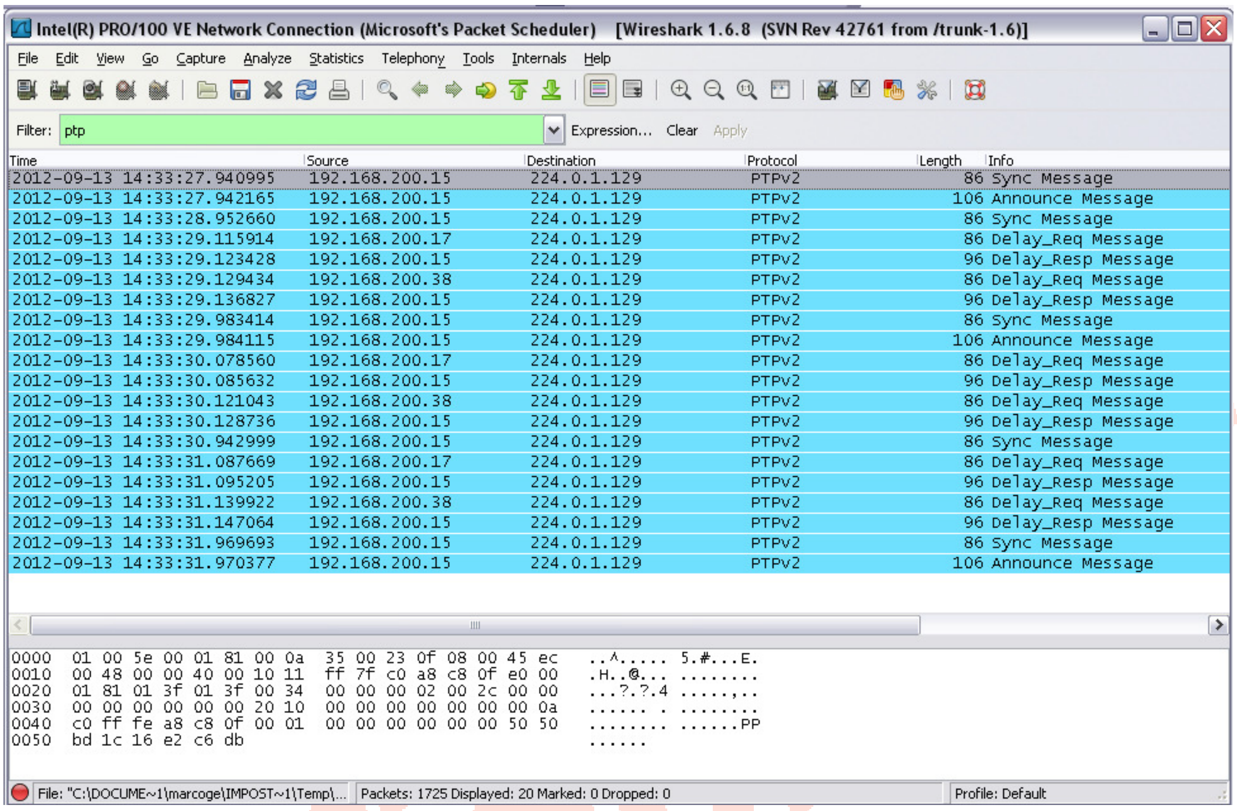
GPS
Cable Delay — Value: 0 ns
Position Mode — Value: Normal Positioning
TI RAIM Alarm — Value 300 ns

PTP
Enable: 🟩
Profile — Set type: DEFAULT
Master/Slave — Set type: slave

EXT
Frequency — Value: 0 Hz; Valid: 🟥; Accepted: 2048000 Hz
Time — Value: 0 ms; Valid: 🟥; Accepted: 1 PPS
PPS Offset — Value: 0 ns

After the setup is completed we can make sure that the master node is transmitting data (*Sync* and *Announce* packets) and that the slave node is answering (*Delay_Req* packets).

A powerful tool for testing the PTP functionality is Wireshark[1]

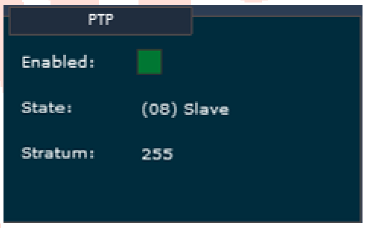

Intel(R) PRO/100 VE Network Connection (Microsoft's Packet Scheduler) [Wireshark 1.6.8 (SVN Rev 42761 from /trunk-1.6)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: ptp
Expression... Clear Apply

```
Time                        Source            Destination    Protocol  Length Info
2012-09-13 14:33:27.940995  192.168.200.15    224.0.1.129    PTPv2         86 Sync Message
2012-09-13 14:33:27.942165  192.168.200.15    224.0.1.129    PTPv2        106 Announce Message
2012-09-13 14:33:28.952660  192.168.200.15    224.0.1.129    PTPv2         86 Sync Message
2012-09-13 14:33:29.115914  192.168.200.17    224.0.1.129    PTPv2         86 Delay_Req Message
2012-09-13 14:33:29.123428  192.168.200.15    224.0.1.129    PTPv2         96 Delay_Resp Message
2012-09-13 14:33:29.129434  192.168.200.38    224.0.1.129    PTPv2         86 Delay_Req Message
2012-09-13 14:33:29.136827  192.168.200.15    224.0.1.129    PTPv2         96 Delay_Resp Message
2012-09-13 14:33:29.983414  192.168.200.15    224.0.1.129    PTPv2         86 Sync Message
2012-09-13 14:33:29.984115  192.168.200.15    224.0.1.129    PTPv2        106 Announce Message
2012-09-13 14:33:30.078560  192.168.200.17    224.0.1.129    PTPv2         86 Delay_Req Message
2012-09-13 14:33:30.085632  192.168.200.15    224.0.1.129    PTPv2         96 Delay_Resp Message
2012-09-13 14:33:30.121043  192.168.200.38    224.0.1.129    PTPv2         86 Delay_Req Message
2012-09-13 14:33:30.128736  192.168.200.15    224.0.1.129    PTPv2         96 Delay_Resp Message
2012-09-13 14:33:30.942999  192.168.200.15    224.0.1.129    PTPv2         86 Sync Message
2012-09-13 14:33:31.087669  192.168.200.17    224.0.1.129    PTPv2         86 Delay_Req Message
2012-09-13 14:33:31.095205  192.168.200.15    224.0.1.129    PTPv2         96 Delay_Resp Message
2012-09-13 14:33:31.139922  192.168.200.38    224.0.1.129    PTPv2         86 Delay_Req Message
2012-09-13 14:33:31.147064  192.168.200.15    224.0.1.129    PTPv2         96 Delay_Resp Message
2012-09-13 14:33:31.969693  192.168.200.15    224.0.1.129    PTPv2         86 Sync Message
2012-09-13 14:33:31.970377  192.168.200.15    224.0.1.129    PTPv2        106 Announce Message
```

```
0000  01 00 5e 00 01 81 00 0a  35 00 23 0f 08 00 45 ec   ..^..... 5.#...E.
0010  00 48 00 00 40 00 10 11  ff 7f c0 a8 c8 0f e0 00   .H..@... ........
0020  01 81 01 3f 01 3f 00 34  00 00 00 02 00 2c 00 00   ...?.?.4 .....,..
0030  00 00 00 00 00 00 20 10  00 00 00 00 00 00 00 0a   ...... . ........
0040  c0 ff fe a8 c8 0f 00 01  00 00 00 00 00 00 50 50   ........ ......PP
0050  bd 1c 16 e2 c6 db                                 ......
```

File: "C:\DOCUME~1\marcoge\IMPOST~1\Temp\... | Packets: 1725 Displayed: 20 Marked: 0 Dropped: 0 | Profile: Default

We can make sure that the MXS-EVO has locked to the master node in the *PTP Status* menu



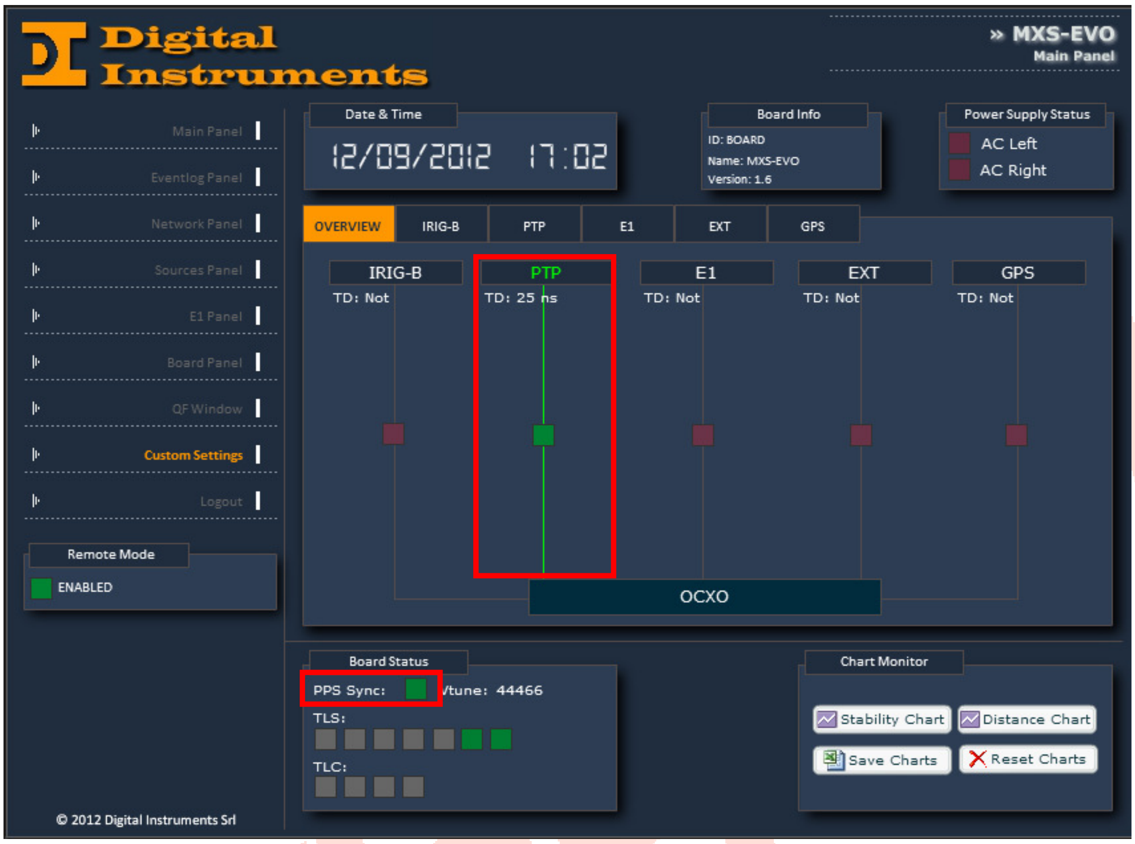
Enabled: 🟩

State: (08) Slave

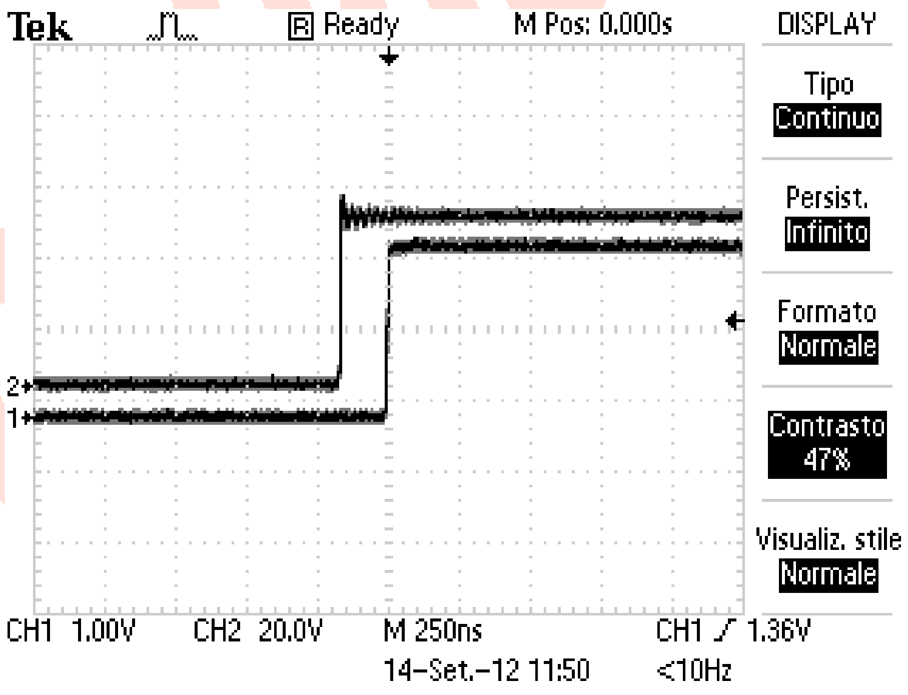
Stratum: 255

[1] http://www.wireshark.org/

We should also make the PTP to discipline that clock, if it is not already so and let the MXS-EVO to synchronize (PPS sync).



Is now possible to check the locking of the PPS from the two devices.



In this way we are transferring also the phase information in addition to the frequency and the two devices are also synchronized in time.

Test #2

In this test we want to check the timing information provided by the two devices.

1. Just operate as in Test #1
2. Then we need to setup a NTP daemon[2]
3. Just add a line for every device (MXS-EVO and ETS-EVO). It is also possible to specify an external public NTP server if the master is disciplined to a UTC source (e.g. GPS) in order to compare the time stamp with an external source
```
## ntpd.conf
server ntp1.ien.it
server ntp2.ien.it
server 192.168.200.14
server 192.168.200.15
```
4. Ask the timing information with `ntpq -c peer`
```
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
+ntp1.inrim.it   .CTD.            1 u    4   64  377   73.870   -4.482   0.868
+ntp2.inrim.it   .CTD.            1 u   32   64  377   74.691   -3.401   1.008
+192.168.200.14  .GPS.            1 u   31   64  377  122.112   48.593  33.547
*192.168.200.15  .SHM.            1 u   21   64  377   20.853  -25.797   4.407
```

The time stamps should be coherent each other and, eventually, also to the external servers.

Test #3

In this test we want to check how to improve the PTP accuracy by combining it with the Synchronous Ethernet feature aswell.

1. Just operate as in Test #1
2. Enable the SyncE feature as previously done

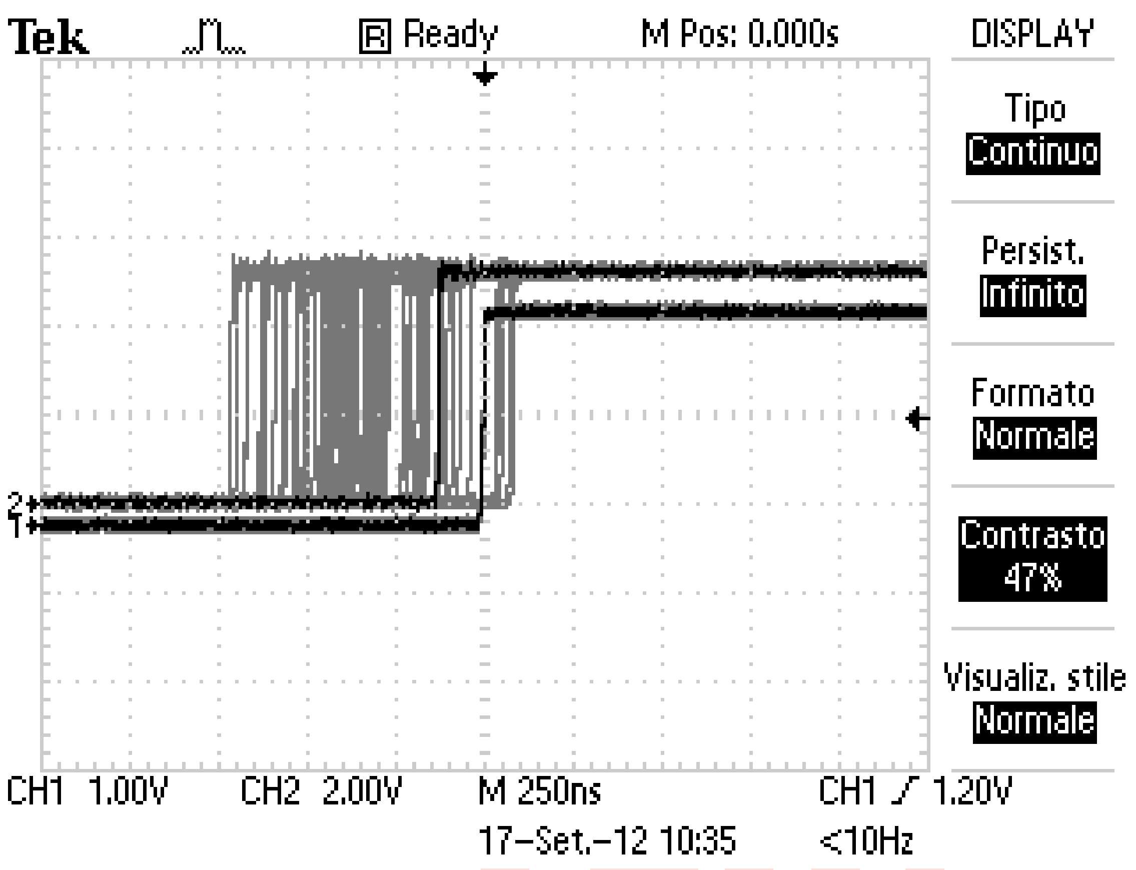
In this way the clock is frequency locked via the Synchronous Ethernet protocol and the PPS is phase locked via the IEEE 1588 protocol (PTP).
In fact the two protocols together represent a very powerful solution.

It is still needed to bear in mind that in this way there could be no strict relationship between the 10 MHz clock and the PPS output (the PPS may not fall every 10.000.000 cycles, but may jitter a bit around that value).

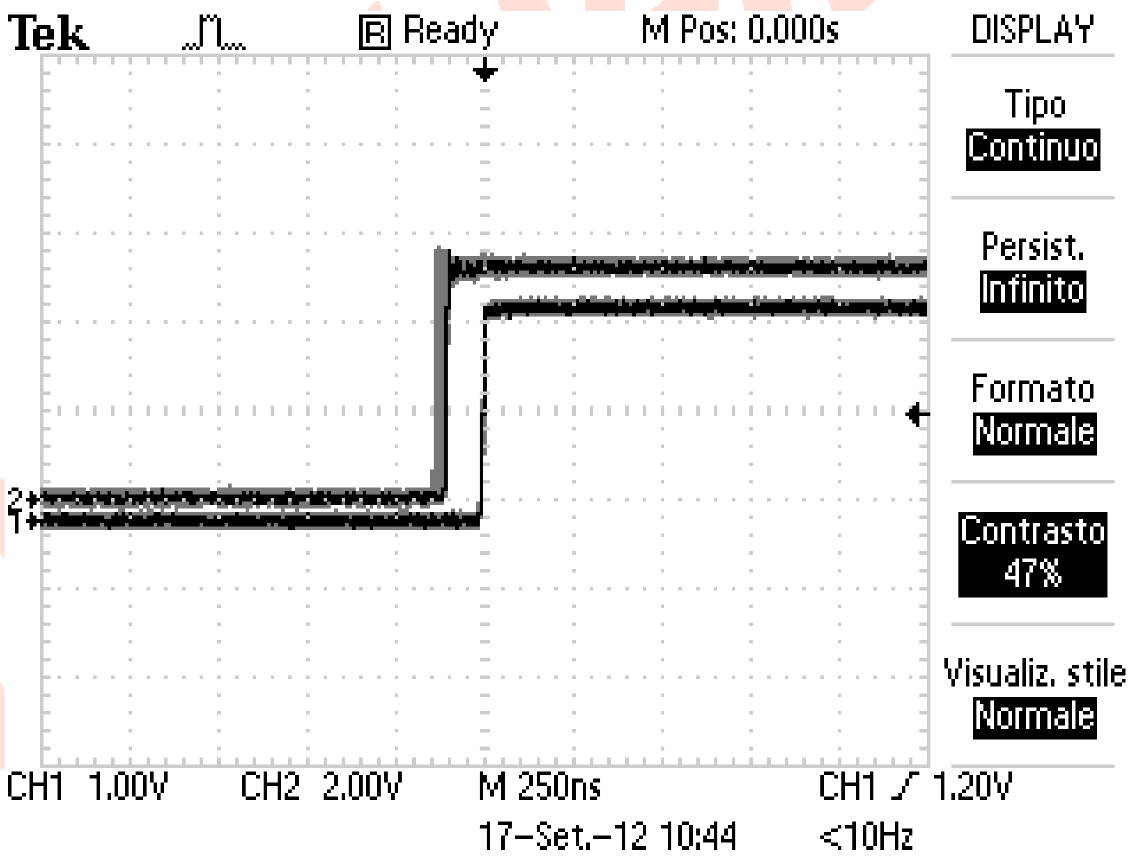
It is also valuable to compare the way a PTP compliant network behaves in respect to a normal network.

The following graphs show how a PTP Transparent Clock may help improving the overall precision of the PPS reconstruction. They have been taken by just enabling or disabling the Transparent Clock feature in the SYNC-SWITCH (as already previously done) and by letting the devices run with an infinite persistence for a few minutes.

[2] GNU/Linux version can be downloaded from http://www.ntp.org/downloads.html
Microsoft Windows version can be downloaded from http://norloff.org/ntp/ntp-4.2.6p3-RC8.zip



TC feature disabled



TC feature enabled

Pulse Generation

Goal

- to test the distribution of a time source over a wired interface with time triggered user programmable pulses

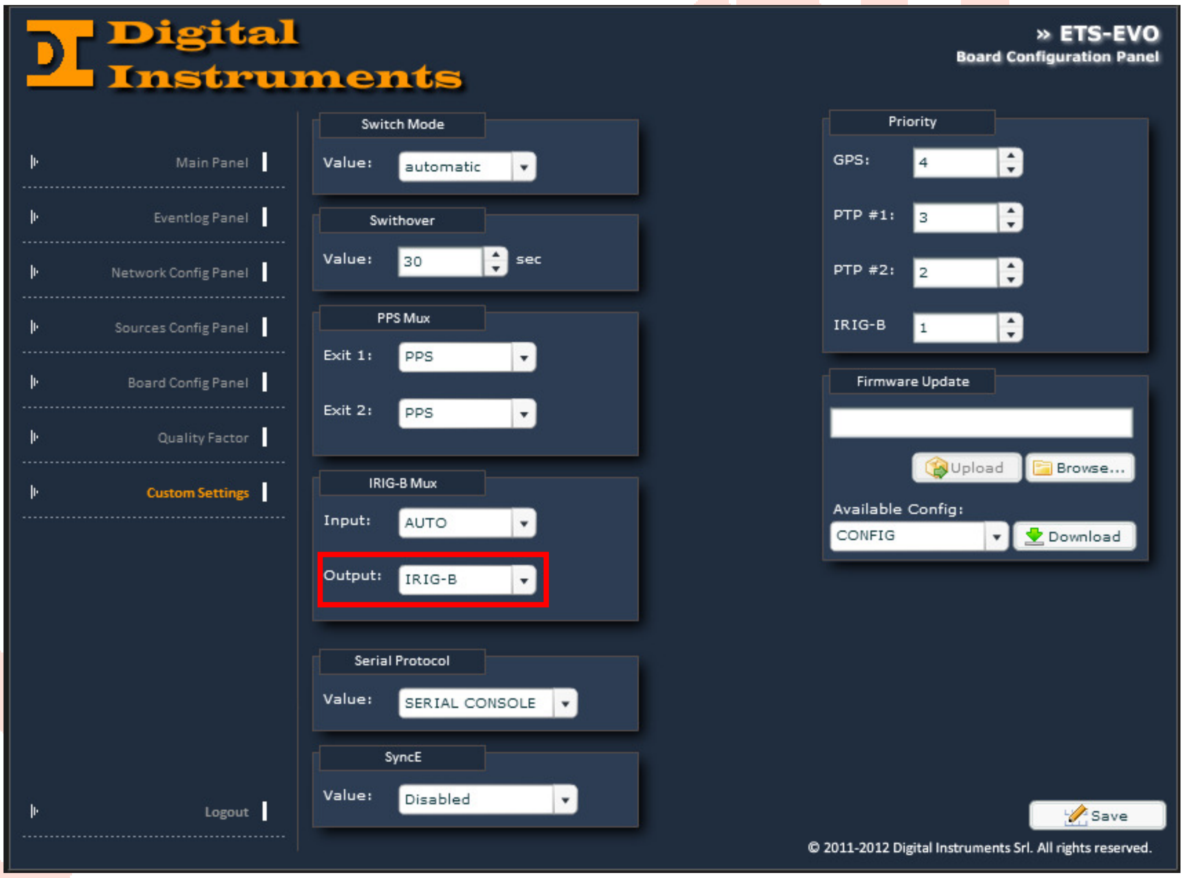
Requirements

- MXS-EVO
- ETS-EVO
- Oscilloscope

Test #1

In this test we want to test the distribution of a time signal (IRIG-B) over a wired interface.

1. First we should let the ETS-EVO and the MXS-EVO to lock on any available source (GPS or PTP)
2. Then we have to check that the IRIG-B output mux is outputting the IRIG-B time signal



Switch Mode
Value: automatic

Swithover
Value: 30 sec

PPS Mux
Exit 1: PPS
Exit 2: PPS

IRIG-B Mux
Input: AUTO
Output: IRIG-B

Serial Protocol
Value: SERIAL CONSOLE

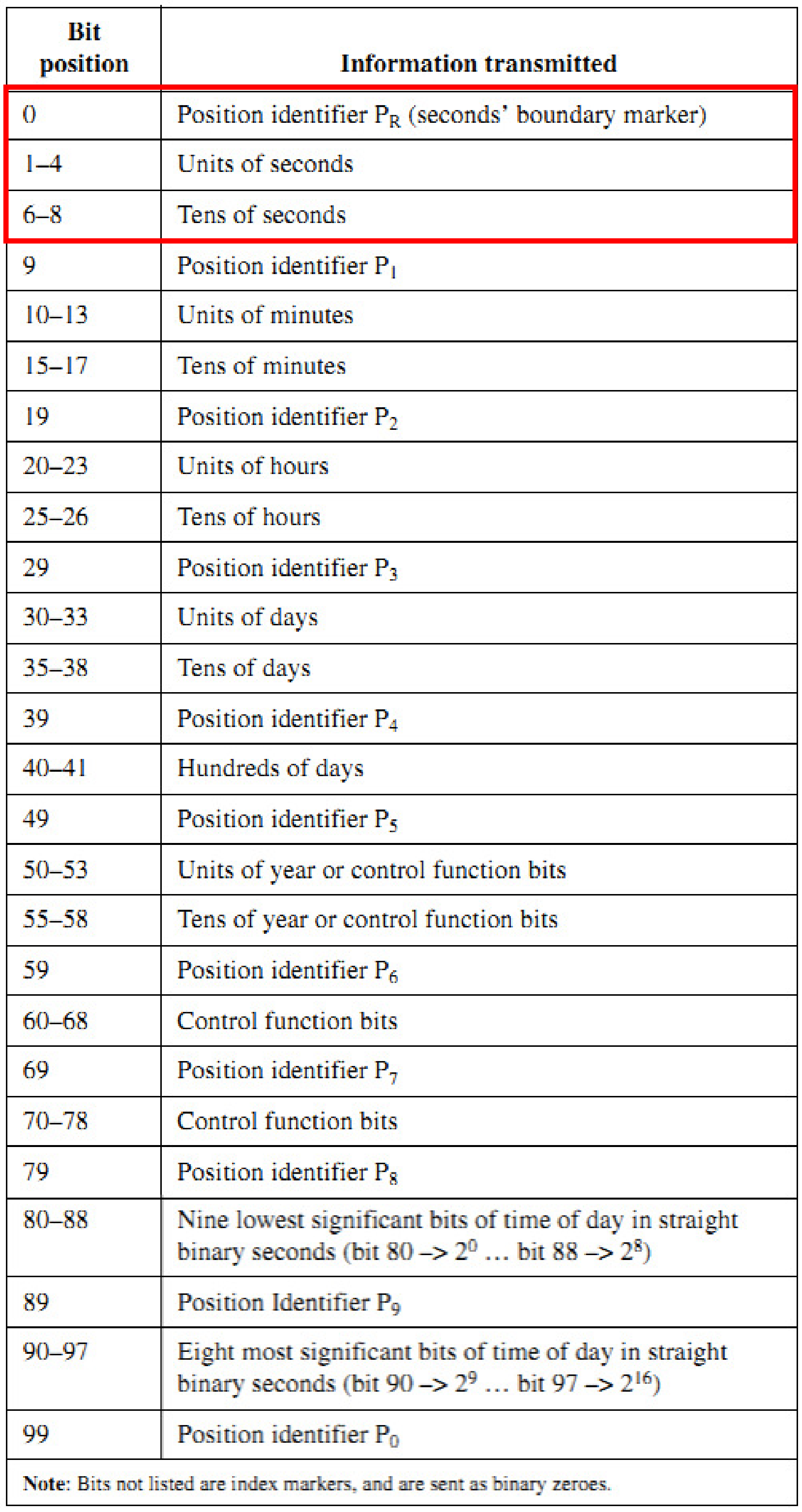
SyncE
Value: Disabled

Priority
GPS: 4
PTP #1: 3
PTP #2: 2
IRIG-B 1

Firmware Update
Available Config: CONFIG

© 2011-2012 Digital Instruments Srl. All rights reserved.

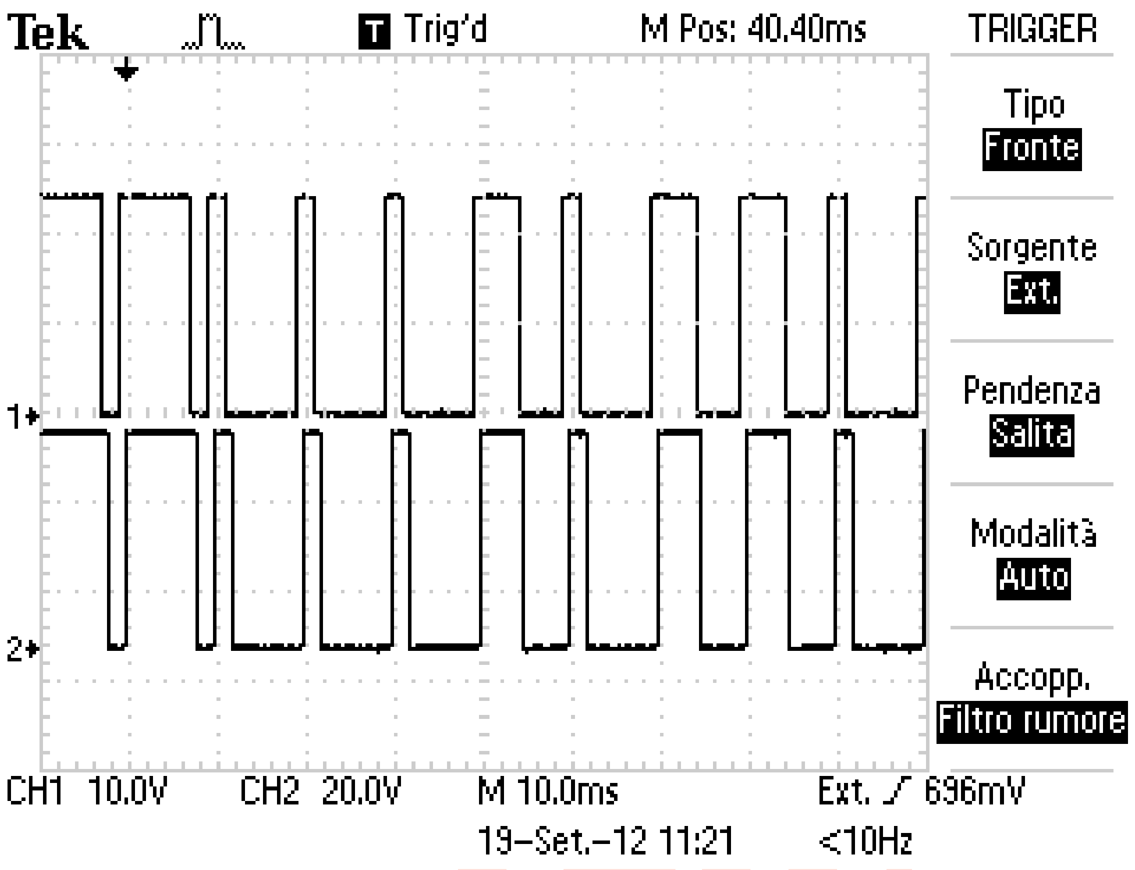
With an oscilloscope is possible to compare the output signals from the two devices.
The IRIG-B signal carries information about date and time, as shown in the following table.



Bit position	Information transmitted
0	Position identifier P_R (seconds' boundary marker)
1–4	Units of seconds
6–8	Tens of seconds
9	Position identifier P_1
10–13	Units of minutes
15–17	Tens of minutes
19	Position identifier P_2
20–23	Units of hours
25–26	Tens of hours
29	Position identifier P_3
30–33	Units of days
35–38	Tens of days
39	Position identifier P_4
40–41	Hundreds of days
49	Position identifier P_5
50–53	Units of year or control function bits
55–58	Tens of year or control function bits
59	Position identifier P_6
60–68	Control function bits
69	Position identifier P_7
70–78	Control function bits
79	Position identifier P_8
80–88	Nine lowest significant bits of time of day in straight binary seconds (bit 80 –> 2^0 … bit 88 –> 2^8)
89	Position Identifier P_9
90–97	Eight most significant bits of time of day in straight binary seconds (bit 90 –> 2^9 … bit 97 –> 2^{16})
99	Position identifier P_0

Note: Bits not listed are index markers, and are sent as binary zeroes.

It is quite easy to recognize the time passing by triggering a PPS and observing the first few bits indicating the seconds.



The above picture represents the time 38 seconds.

<u>Test #2</u>

In this test we want to to test the generation of a user defined time triggered pulse.

The MXS-EVO and ETS-EVO devices can generate user defined programmable pulses.

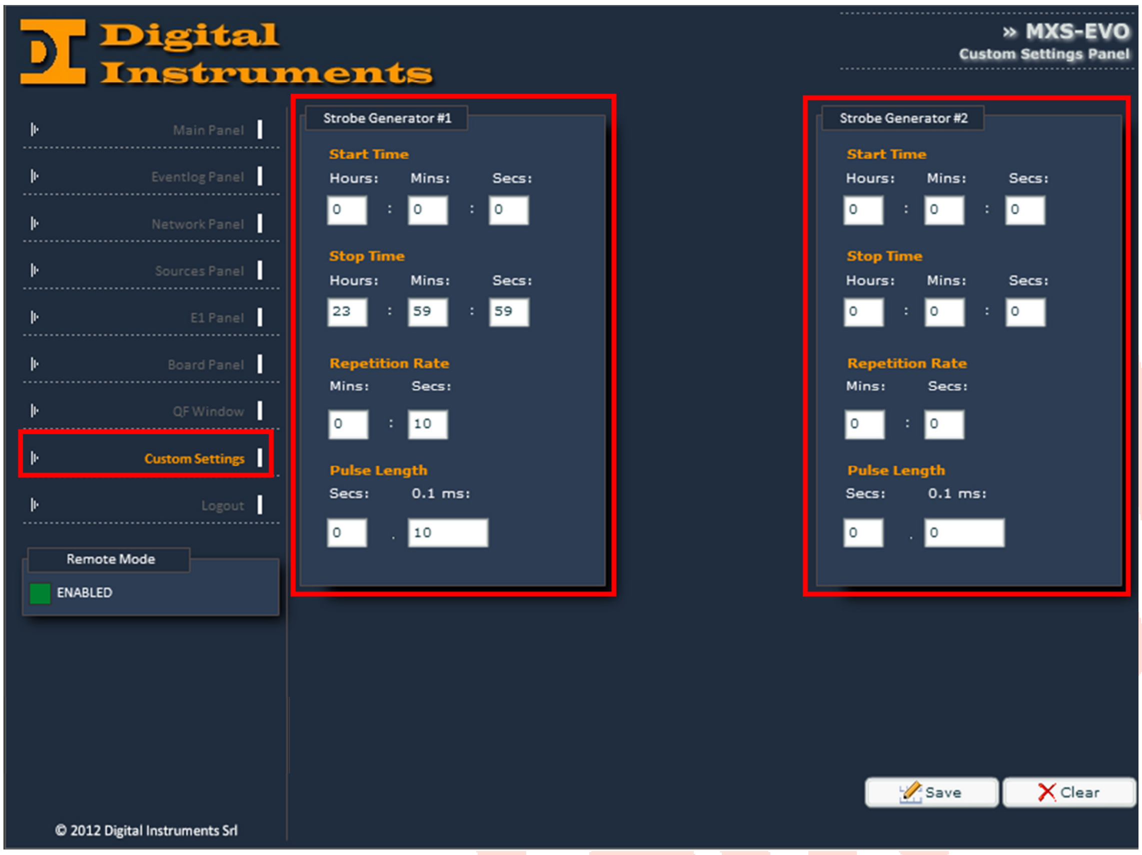
It is possible to set the following parameters:
- <Start Time> HH:MM:SS
- <Stop Time> HH:MM:SS
- <Repetition Rate> MM:SS
- <Pulse Length> SS.ssss

The system begins to generate a sequence of pulses of <Pulse Length> length and repeating every <Repetition Rate>, starting at <Start Time> and finishing at <Stop Time>.

Please note that the <Start Time> always falls synchronous to the internally generated PPS.

1. Just operate as in Test #1
2. Configure a periodic pulse of 1 ms every 10 seconds

a.	Start time:	00:00:00
b.	Stop time:	23:59:59
c.	Repetition Rate:	00:10
d.	Pulse Length:	10



Remote Mode: 🟩 ENABLED

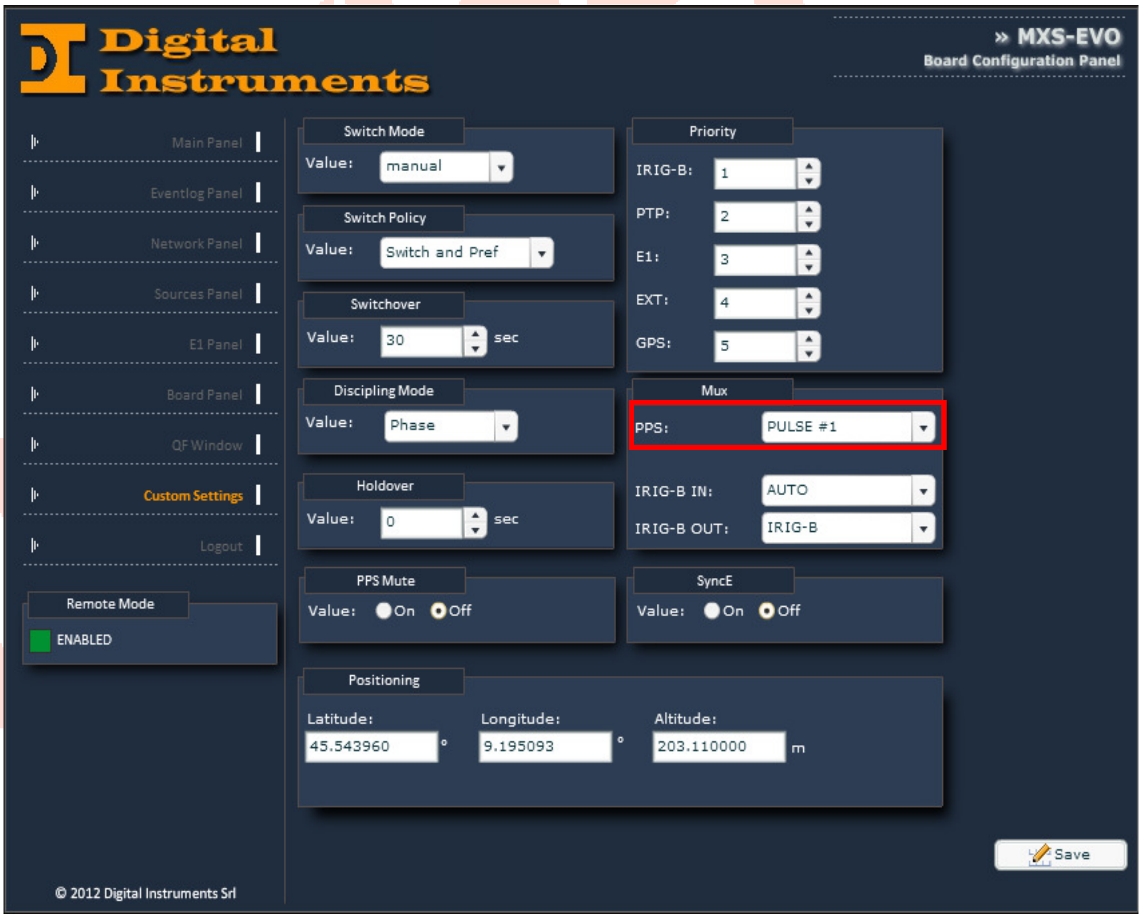
Strobe Generator #1

	Hours/Mins	Mins/Secs	Secs
Start Time (Hours : Mins : Secs)	0	0	0
Stop Time (Hours : Mins : Secs)	23	59	59
Repetition Rate (Mins : Secs)	0	10	
Pulse Length (Secs . 0.1 ms)	0	10	

Strobe Generator #2

	Hours/Mins	Mins/Secs	Secs
Start Time (Hours : Mins : Secs)	0	0	0
Stop Time (Hours : Mins : Secs)	0	0	0
Repetition Rate (Mins : Secs)	0	0	
Pulse Length (Secs . 0.1 ms)	0	0	

3. Set the Pulse #1 to be outputted on the PPS connector



Remote Mode: 🟩 ENABLED

Switch Mode — Value: manual

Switch Policy — Value: Switch and Pref

Switchover — Value: 30 sec

Discipling Mode — Value: Phase

Holdover — Value: 0 sec

PPS Mute — Value: On [✓ Off]

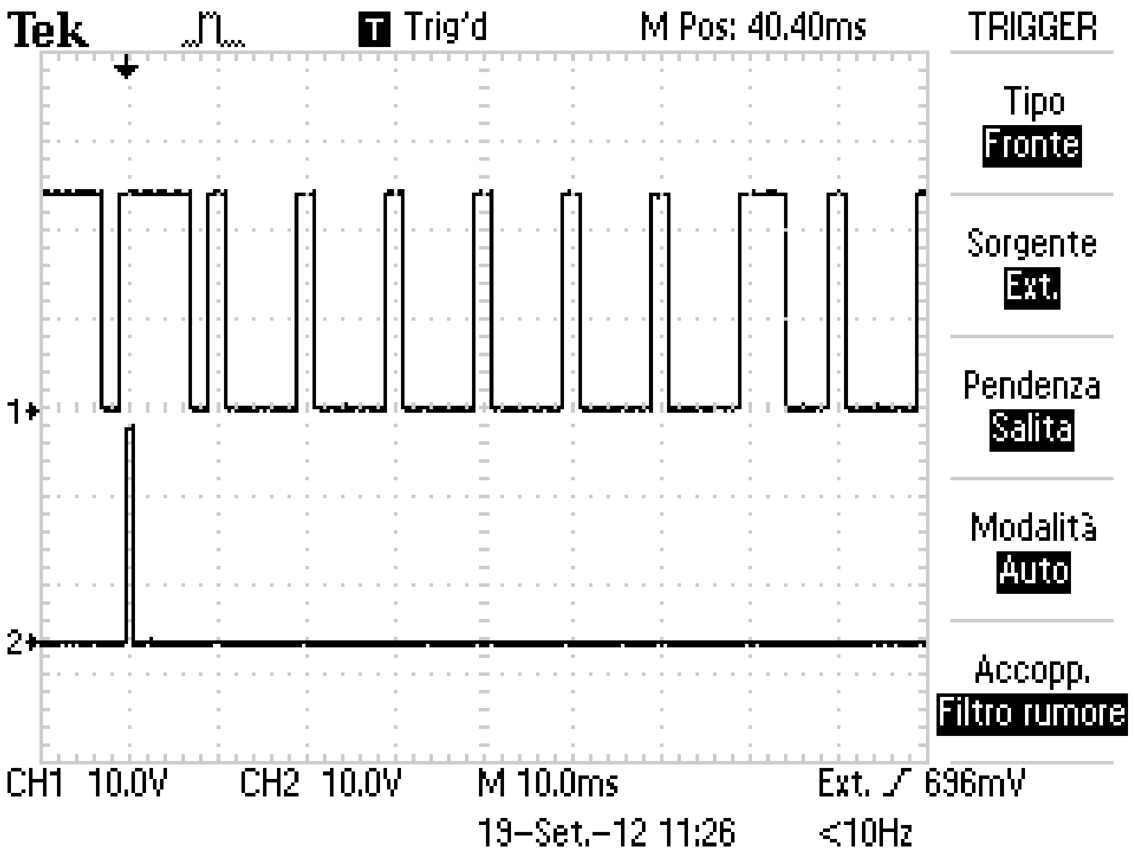
Priority — IRIG-B: 1; PTP: 2; E1: 3; EXT: 4; GPS: 5

Mux — PPS: PULSE #1; IRIG-B IN: AUTO; IRIG-B OUT: IRIG-B

SyncE — Value: On [✓ Off]

Positioning — Latitude: 45.543960 °; Longitude: 9.195093 °; Altitude: 203.110000 m

It is possible to check that the custom pulse is generated every tenth second in the minute (0, 10, 20, 30, 40 and 50) by looking at the IRIG-B output.



The above picture represents the time 20 seconds and a 1 ms pulse is correctly generated.

Conclusions

Distributed systems with real time requirements have been often based on field-buses 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 market, high bandwidth, priority features (e.g. VoIP) and the availability of Ethernet switches and Ethernet enabled products fulfilling industrial environmental requirements.

However, as we have seen, 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 can be solved if the raw data are time stamped and if the devices are synchronized in time.

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

In this paper we have presented a few methods for synchronizing remote devices via an Ethernet connection, both in frequency (**SyncE**) and time (**PTP**), with precision ranging from nanoseconds (**SyncE** and **PTP**) to milliseconds (**NTP**).

Digital Instruments is up-to-date on the topic and is manufacturing its very own solutions, by having full control over the hardware and software components involved in the process.

We are particularly willing to let our customers get used to these technologies and let them understand what is the best solution that fits their needs.

If you would like to test our solutions or ask some questions about these or other subjects please do not hesitate to contact us!

Digital Instruments S.r.l.
Via Parco degli Scout,13
20091 BRESSO (MI) - ITALY
Tel: +39-0266506250
Fax: +39-0266506103
Info e-mail: info@digital-instruments.it
Commercial e-mail: marco.muritti@digital-instruments.it
http://www.digital-instruments.it