



European Organisation  
for Astronomical  
Research in the  
Southern Hemisphere

Organisation Européenne  
pour des Recherches  
Astronomiques  
dans l'Hémisphère Austral

Europäische Organisation  
für astronomische  
Forschung in der  
südlichen Hemisphäre

## DIRECTORATE OF ENGINEERING

### WP345 IEEE1588 PROJECT - TECHNICAL REPORT

VLT-TRE-ESO-17240-5539 Issue 1

20/03/2012

Author

M. Kiekebusch

20/03/2012

WP Manager

C. Lucuix

20/03/2012

DOE Director

M. Peron

27/03/2012

Name	Affiliation
------	-------------

[illegible]

# Contents

<b>1.1</b>	<b>Abbreviations and Acronyms .....</b>	<b>4</b>
<b>1.2</b>	<b>Scope.....</b>	<b>5</b>
<b>1.3</b>	<b>Applicable documents .....</b>	<b>5</b>
<b>1.4</b>	<b>Reference documents .....</b>	<b>5</b>
<b>2</b>	<b>EXECUTIVE SUMMARY .....</b>	<b>6</b>
<b>3</b>	<b>OVERVIEW.....</b>	<b>7</b>
<b>3.1</b>	<b>IEEE 1588 .....</b>	<b>7</b>
<b>3.2</b>	<b>Time Requirements .....</b>	<b>7</b>
<b>3.3</b>	<b>COTS Solutions .....</b>	<b>8</b>
3.3.1	EtherCat Distributed Clocks .....	8
3.3.2	Beckhoff Oversampling .....	9
3.3.3	Component List .....	9
	Embedded PC CX1030 .....	10
	Digital Output Terminal ES2262 .....	10
	Terminal Servers.....	12
	EtherCATextension.....	12
	EtherCAT coupler .....	12
	Windows PC .....	12
	Linux WS.....	12
<b>4</b>	<b>FUNCTIONAL TESTS.....</b>	<b>13</b>
<b>4.1</b>	<b>Fast I/O Control.....</b>	<b>13</b>
4.1.1	Description .....	13
4.1.2	Test Setup .....	13
4.1.3	Results .....	13
<b>4.2</b>	<b>High I/O Time Resolution.....</b>	<b>14</b>
4.2.1	Description .....	14
4.2.2	Test Setup .....	14
4.2.3	Results .....	15
<b>4.3</b>	<b>DC Synchronization .....</b>	<b>16</b>
4.3.1	Description .....	16
4.3.2	Test Setup .....	16
4.3.3	Results .....	17
<b>4.4</b>	<b>UTC Synchronization .....</b>	<b>18</b>
4.4.1	Description .....	18
4.4.2	Test Setup .....	18
4.4.3	Results .....	21
	4.5.3.1 Terminal Version.....	21
	4.5.3.2 Time Server Connection .....	21
	4.5.3.3 Reading DC Time .....	21
	4.5.3.4 UTC Time.....	22
	4.5.3.5 Time Synchronization .....	23
<b>4.5</b>	<b>Timer Definition .....</b>	<b>23</b>
4.5.1	Description .....	23
4.5.2	Results .....	23
<b>4.6</b>	<b>High Level Time Synchronization.....</b>	<b>24</b>
4.6.1	Description .....	24
4.6.2	Test Setup .....	25
4.6.3	Results .....	25
	4.6.3.1 PTPd.....	25
	4.6.3.2 PTPv2 from InEs ZHAW.....	26
<b>4.7</b>	<b>Tracking Device .....</b>	<b>28</b>
4.7.1	Description .....	28

4.7.2	Results .....	28
<b>4.8</b>	<b>VLTSW Demo Application .....</b>	<b>28</b>
<b>5</b>	<b>CONCLUSIONS .....</b>	<b>30</b>
<b>5.1</b>	<b>Time Servers .....</b>	<b>30</b>
<b>5.2</b>	<b>Time Synchronization .....</b>	<b>30</b>
<b>5.3</b>	<b>High-level Synchronization .....</b>	<b>30</b>
<b>5.4</b>	<b>Next Steps .....</b>	<b>30</b>

# Abbreviations

## 1.1 Abbreviations and Acronyms

In addition to acronyms given in applicable document AD1 the following acronyms apply to this document.

COTS	Commercial Off-The-Shelf
DC	Distributed Clocks
FB	Function Block
NTP	Network Time Protocol
PLC	Programmable Logic Controller
PPS	Pulse Per Second
PTP	Precision Time Protocol
SL	Scientific Linux
ST	Structured Text
TAI	International Atomic Time
TDEM	Telescope Demonstrator
TRS	Time Reference System
UTC	Coordinated Universal Time
WP	Work Package

## 1.2 Scope

This document presents the results obtained in the scope of the WP345 project related to the integration of the time protocol over Ethernet (IEEE1588) into the VLT control system.

## 1.3 Applicable documents

The following applicable documents form a part of the present document to the extent specified herein. In the event of conflict between applicable documents and the content of the present document, the content of the present document shall be taken as superseding.

AD1 E-ESO-SPE-313-0066 Issue 1, Common definitions and acronyms

## 1.4 Reference documents

- RD1 IEEE Std 1588-2008, Precision clock synchronization protocol for networked measurement and control systems
  - RD2 E-TRE-ESO-449-0288, Issue 1, Technology Demonstrator Report – Timing System
  - RD3 E-TRE-ESO-449-0288, Issue 2, Technology Demonstrator Report – Timing System
  - RD4 VLT-MAN-ESO-17300-0473, 2.0 --- TIM Technical Manual
  - RD5 VLT-MAN-ESO-17300-1254, Issue 3, VLT Central Time Standard - Manual
  - RD6 Beckhoff Application Note DK9222-0909-005 – XFC technology oversampling
  - RD7 Design Considerations for Software Only Implementations of the IEEE1588 Precision Time Protocol – Kendall Correl, Nich Barendt and Michael Branicky.
  - RD8 Beckhoff Website – [www.beckhoff.de](http://www.beckhoff.de)
-

## 2 Executive Summary

The WP345 IEEE1588 mini project has been defined on April 2011 with the purpose to evaluate the integration of the IEEE1588 time protocol into the VLT control system specifically for controlling instrument functions using fieldbus technology. The replacement of LCU by PLCs is subordinated to their capabilities to provide similar features in various areas of control including the time synchronization.

Ethernet based time protocol (IEEE1588) has been already studied in the context of the E-ELT Telescope Demonstrator (TDEM) where it was demonstrated a sub-microsecond synchronization accuracy of the version 2 of this protocol [RD3]. Consequently the IEEE1588 became the baseline solution for the E-ELT and it has been the main driver for the ongoing changes of the VLT time reference system [RD5].

However having the availability of a time network is not enough to satisfy all high-end time related functionalities. So it is required to demonstrate how this time protocol can be used in combination with the PLCs to fulfil the synchronization requirements of future VLT instruments. This has been the focus of this project and in order to achieve this task a set of COTS components were employed in connection to the PTP time network with the purpose to evaluate them as a replacement for some of the LCU TIM board capabilities.

The outcome of this work demonstrated the following:

- Internal time synchronization of EtherCat, based on distributed clocks, is very good. It has been measured a difference of less than 10 [nsec] between two output signals within the same EtherCat network. The synchronization precision goes down when using an external time reference (IEEE1588) but still is better than 1 [μs] in the worst case. This precision satisfy by large the known VLT requirements for hardware synchronization of infrared instruments.
- It was possible to access the absolute time from the PLC through a gateway terminal (EL6688) and program one-shot action inside the PLC program with 1 [μs] time resolution.
- It was possible to implement a continuous pulse train with a time resolution up to 1 [μs] achieving a frequency of 0.5 MHz. This allows the implementation of timers that can be used for continuous hardware synchronization which is one of the typical requirements for infrared instruments.
- It was possible to deploy and test the IEEE1588 protocol stack under Linux. It has been tested two solutions both using PTP version 2: one open source (PTP daemon) and one commercial from the Institute of Embedded Systems (InES) from the Zurich University of Applied Sciences (zhaw).
- It was possible to implement a prototype of a tracking device (derotator) running in the WS, using the absolute time coming from the IEEE1588 network and sending continuously position corrections to the PLC during tracking. It has been tested with a Beckhoff stepper motor achieving a tracking loop frequency up to 20 Hz. Telescope coordinates were obtained from the Telescope Simulator and encoder corrections computed using the *slalib*. More complete performance tests could not be done due to the limitations of the motor and encoder but preliminary results are promising validating the implementation of tracking loops at higher level with the fieldbus extension. However we cannot conclude about the performance of this kind of devices running in the WS until we can carry out proper tests with suitable hardware.

### 3 Overview

#### 3.1 IEEE 1588

The IEEE1588 standard or known as well as Precision Time Protocol (PTP) is used to synchronize clocks throughout a computer network. The standard has been reviewed few years ago and new version formally endorsed by the IEEE and IEC which is known as IEEE 1588-2008. This new version, also known as PTP2, is supposed to improve in accuracy, precision and robustness [RD1].

This protocol has been declared as the baseline solution for time synchronization for the E-ELT. This has been the reason for the update of the VLT time reference system toward to provide the IEEE1588 time network in combination with the actual time reference system and thus allowing the VLT upgrade activities. The Figure 1 represents the current VLT time reference system after the upgrade toward the support of IEEE1588 protocol [RD5].

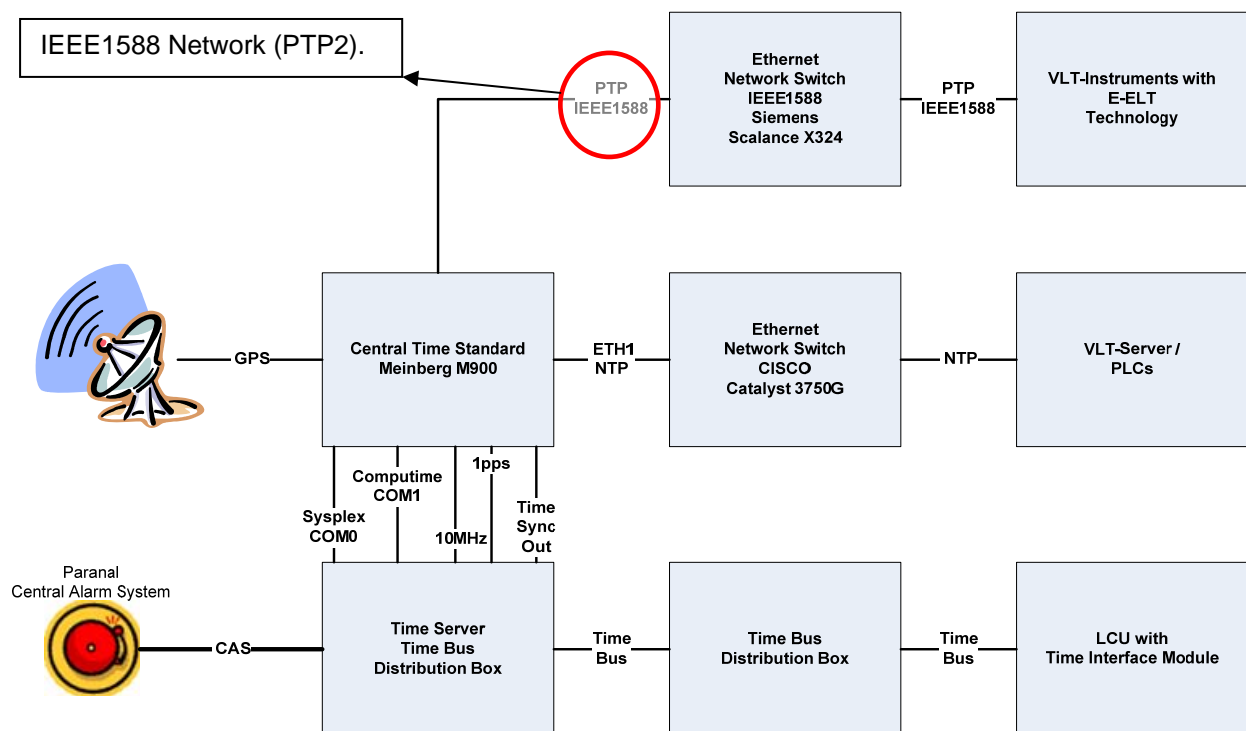


Figure 1: VLT Time Reference System Overview [RD5].

#### 3.2 Time Requirements

There are no special requirements besides the ones provided by the current VLT Time Interface Module (TIM). These requirements will represent our baseline to crosscheck if they can be fulfilled with PLCs

- Access to the Universal Time Coordinated (UTC) provided by the VLT Time Reference System (TRS).
- A set of general purpose timers where the user can specify the starting count value, in absolute time, and the count rate (period). The maximum supported frequency is 1 Mhz.

- <sup>1</sup> Requires a hardware connection to the TIM board P2 connector.



The above figure shows an EtherCAT IEEE1588 gateway (e.g. EL6688) connected to an external clock (grandmaster clock). The external synchronization interface consists of standardized CoE objects, which can be read by the EtherCAT master acyclically per SDO or cyclically via the process data. The External Synchronization Interface also encompasses one time stamp each from the local and the global master clock allowing the calculation of the time control values within the PLC application. The EtherCAT slave with the external synchronization Interface can be located at any desired position in the EtherCAT network.

### 3.3.2 Beckhoff Oversampling

Maximum PLC cycle times are normally equivalent to the fieldbus communication cycle which is a limiting factor. In order to overcome this limitation and obtain finer resolution vendors have implemented some additional features. This is the case of EtherCat distributed clocks and Beckhoff terminals with support for oversampling. Oversampling means writing and reading with significant higher time resolution than the communication cycle time. Digital input and output terminals currently enable resolutions of up to 1  $\mu\text{s}$ , with analog signals the smallest possible interval is 10  $\mu\text{s}$ .

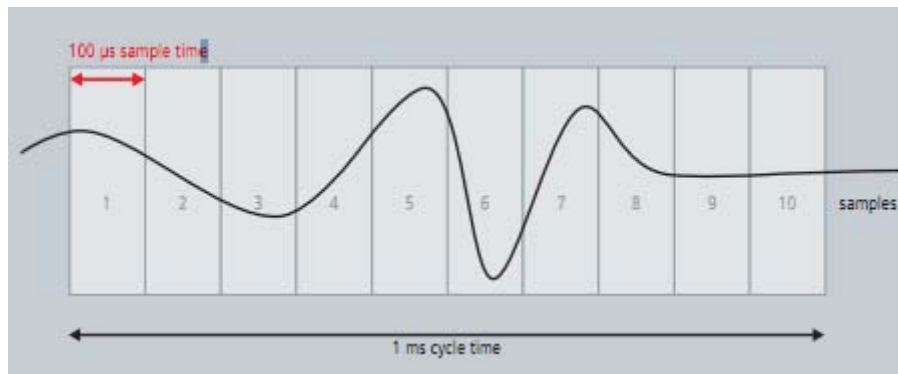
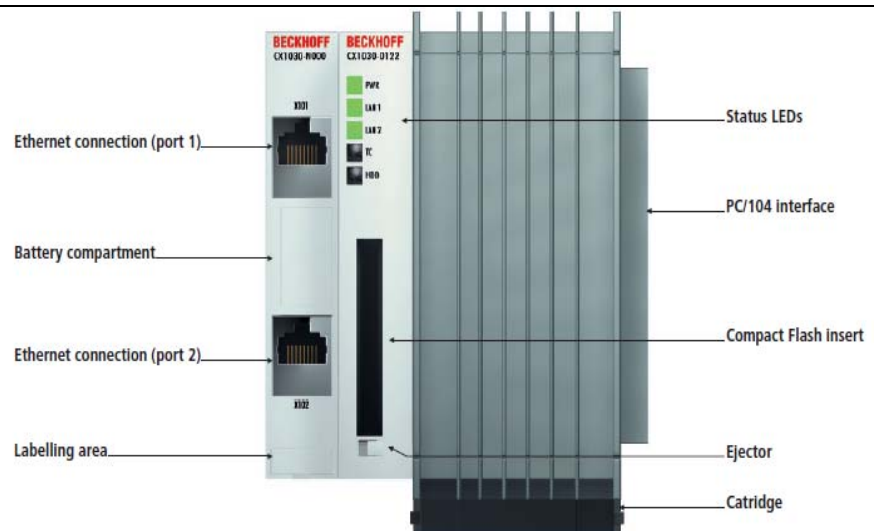
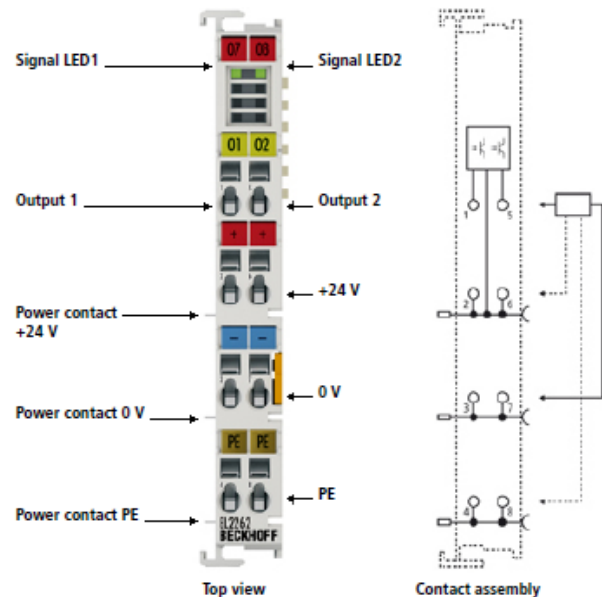
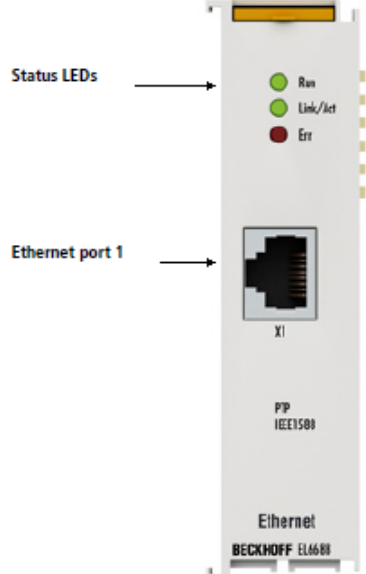
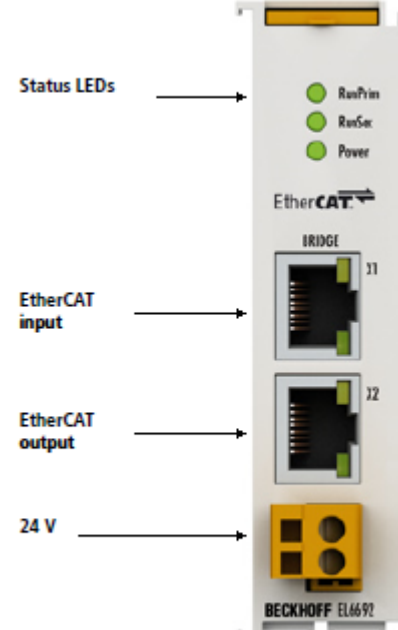





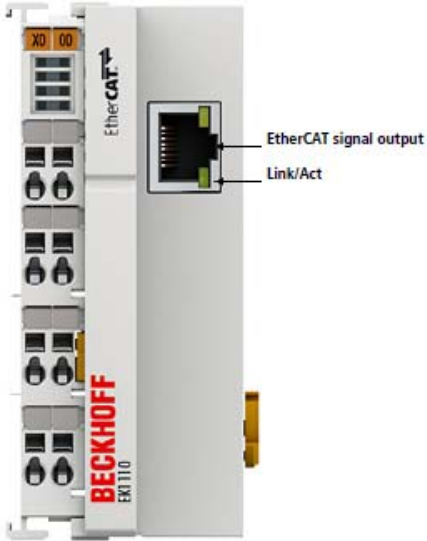
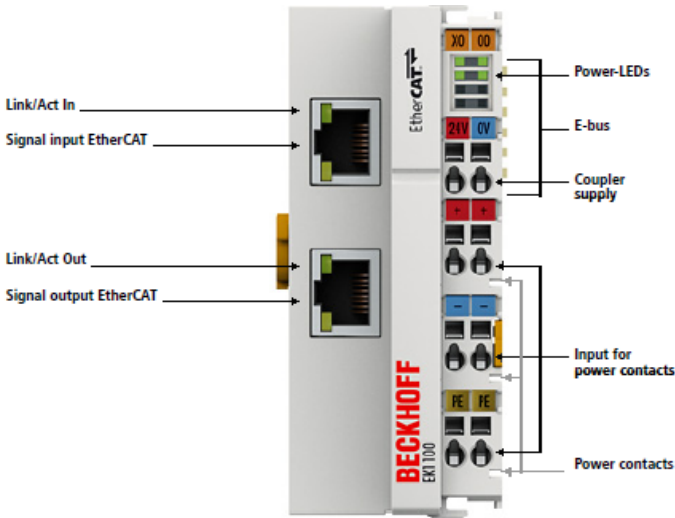
Figure 3: **Oversampling Concept – Cycle time vs sample time [RD6].**

### 3.3.3 Component List

The following table shows the list of main components used to test the time synchronization with PLCs. Not all the components have been used at the same time, they have been combined depending of the purpose of the test.

*Embedded PC  
CX1030**Digital Output  
Terminal  
ES2262*

IEEE1588 Terminal EL6688	 <p>Status LEDs →</p> <p>Ethernet port 1 →</p> <p>X1</p> <p>PP IEEE1588</p> <p>Ethernet BECKHOFF EL6688</p> <p>Top view</p>
EtherCAT bridge	 <p>Status LEDs →</p> <p>EtherCAT input →</p> <p>EtherCAT output →</p> <p>24 V →</p> <p>RunPrim RunSec Power</p> <p>EtherCAT IRIDGE</p> <p>X1</p> <p>X2</p> <p>BECKHOFF EL6692</p> <p>Top view</p>
Network Switch	

<i>Terminal Servers</i>	Symmetricom TimeProvider 5000	
	Meinberg Time Server	
<i>EtherCAT extension</i>		
<i>EtherCAT coupler</i>		
<i>Windows PC</i>	Normal PC used for programming the PLC and configuring the TwinCAT project.	
<i>Linux WS</i>	Standard VLT WS with Scientific Linux (SL) having a dedicated network interface for connecting to the IEEE1588 network.	

## 4 Functional Tests

### 4.1 Fast I/O Control

#### 4.1.1 Description

One of our goals was to validate the fast I/O control from COTS solution that would make possible the synchronization through hardware triggers. This test consisted of activating a digital output and measuring the transition time from Off (0 V) to On (24 V) state. It was used one channel of the Beckhoff terminal ES2262 together with a dummy PLC program implemented in Structure Text (ST) language which is just switching Off/On a digital output signal.

#### 4.1.2 Test Setup

For this test it has been used only one terminal (digital output) additionally to the EtherCAT master in the setup network.

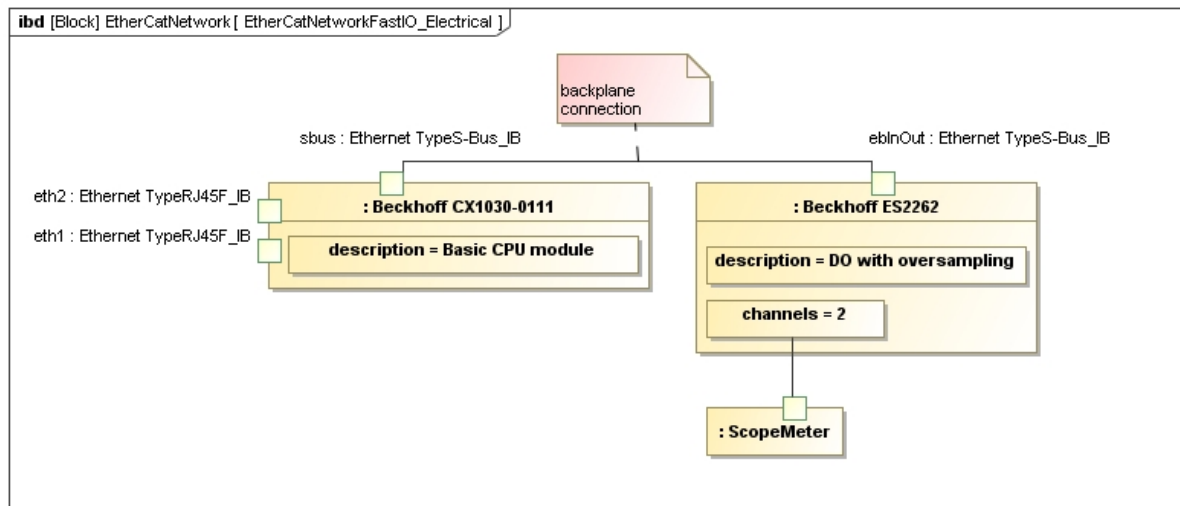


Figure 4: **Fast I/O hardware setup.**

Using the *PLC Control* tool it was configured the interval of the task to 1 [ms]. Then using the *System Manager* tool it was configured the base time to 1 [ms] in the real-time settings. These general settings were kept across all the tests performed in the scope of this report.

#### 4.1.3 Results

The nominal performance from the vendor says that switching time of terminal EL2262 should be less than 1 [μs]. During our tests, it was measured with the scopeMeter a switching time less than 10 [ns]. The results are shown in Figure 5 where the two vertical lines define a range of 10 [nsec].

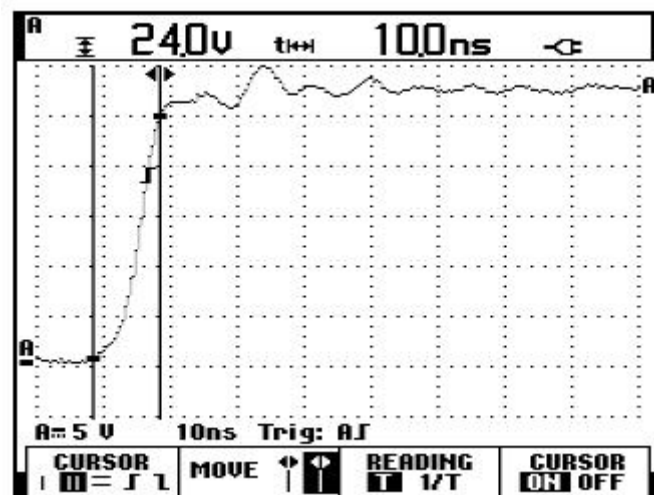


Figure 5: Scope view of DO switching time with terminal ES2262.

## 4.2 High I/O Time Resolution

### 4.2.1 Description

The test consisted in validating Beckhoff oversampling mechanism. A PLC program has been implemented to write the pulse train of the entire oversampling period at each PLC cycle for the two output channels. This information is transferred by the EtherCAT master to the terminal and then, at each terminal cycle, the signals are activated in the digital output module according to the information written in the last PLC cycle.

### 4.2.2 Test Setup

The EtherCAT network setup is the same used for the previous test, see Figure 4. The EL2262 terminal was configured as follows: maximum oversampling factor that allows reaching up to 500 kHz frequency and bytes as operation mode, see Figure 6. This must be taken into account when writing the PLC program to define properly the pulse train for the digital output module. The start time of the hardware trigger shall be used to compute the time, at least, one cycle in advance to be able to keep the synchronization with respect to the absolute time.

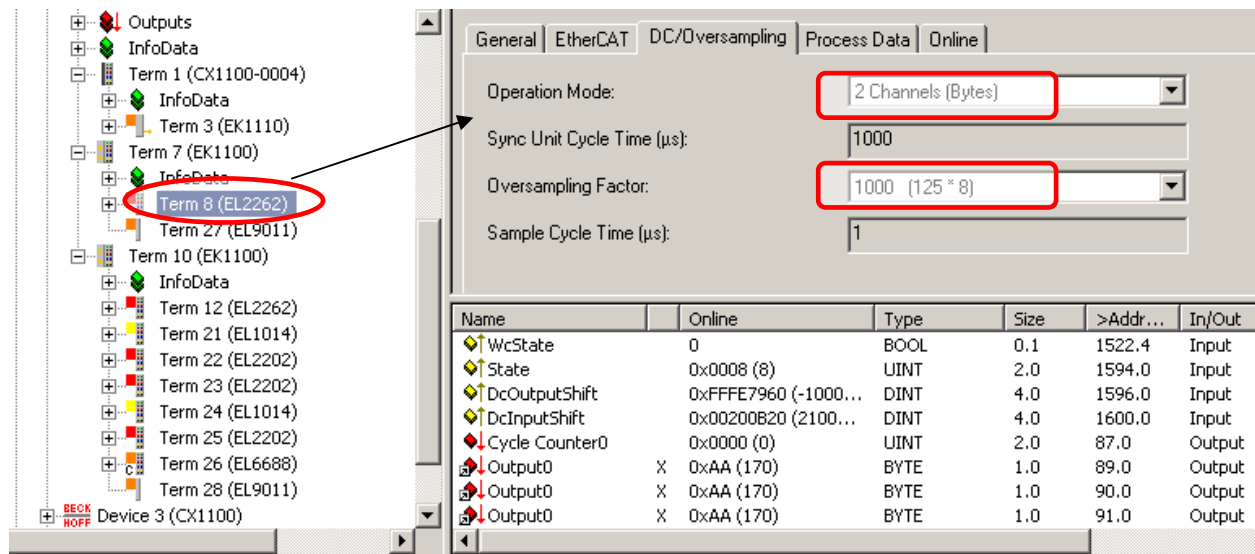


Figure 6: Oversampling configuration

### 4.2.3 Results

Measuring the signals with the scopeMeter it was demonstrated that oversampling mechanism works. We could achieve up to 0.5 [MHz] frequencies when using the maximum oversampling capabilities. The results look stable over time and it was not observed any misbehaviour during the tests. Nevertheless this has not been tested yet systematically to verify the stability of the frequencies over long periods of time.

The Figure 7 shows the two signals with a period of 2 [µs] defined by the distance between the two vertical lines.

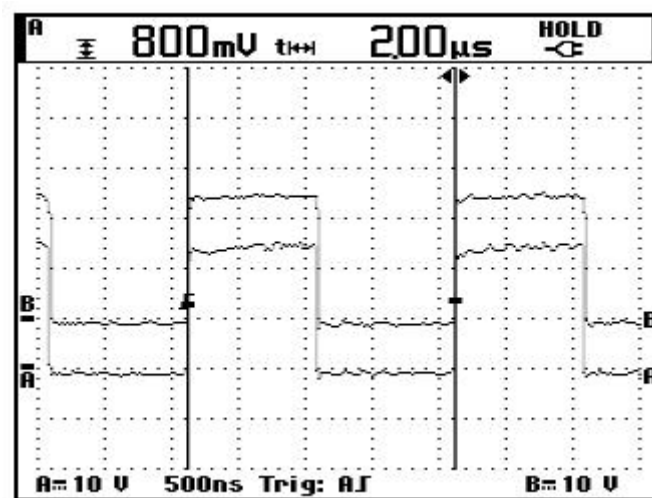


Figure 7: Scope view of the pulse train generated with oversampling.

## 4.3 DC Synchronization

### 4.3.1 Description

This test was aiming to verify the capabilities of the EtherCat time synchronization mechanism (DC).

### 4.3.2 Test Setup

In order to test this it was setup two digital output terminals (ES2262), each one in a different segment of the EtherCat network. The network is divided in two segments by using the EtherCat extension (EK1110) and the EtherCat Coupler (EK1100), see list of components in section 3.3.3. Then it was implemented a PLC program to switch Off/On one the output signals on each of the terminals and a ScopeMeter was used to measure their synchronization.

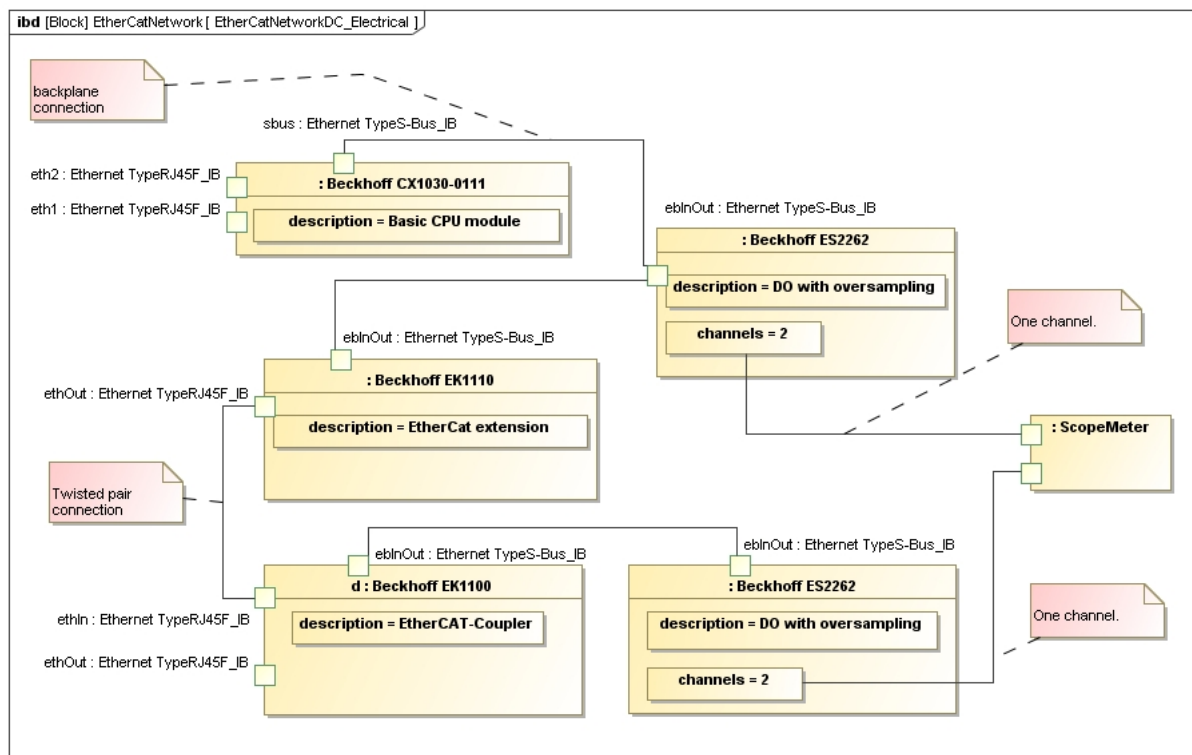


Figure 8: ***EtherCat distributed clocks network setup.***





Figure 9: Test hardware Setup

#### 4.3.3 Results

The synchronization between the two signals coming from different terminals located in different segments of the network is very good, less than 10 [ns]. At this point, this is achieved using the internal synchronization provided by DC but not using any external time source.

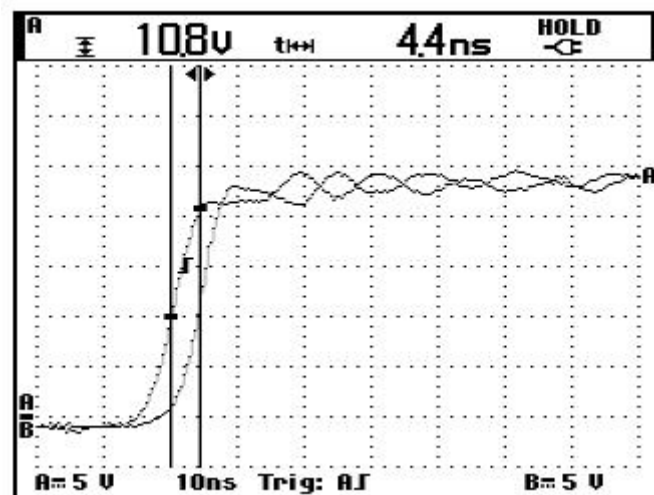


Figure 10: Scope view of the synchronization of two output signals with DC.

## 4.4 UTC Synchronization

### 4.4.1 Description

The purpose of this test is to evaluate the access to the external time source (IEEE1588-2008) activating a signal based on the absolute time rather than the internal DC. This test also measures the time synchronization between two output signals but using the external time reference instead of the DC internal synchronization.

### 4.4.2 Test Setup

Beckhoff has a terminal (EL6688) that allows connecting distributed clocks with an external reference time coming from a IEEE1588 network. It has been used two test configurations, one using the Symmetricom timer server (TimeProvider 5000) connected directly to the EL6688 terminal and another one using the Meinberg time server connected through a Cisco switch supporting the PTP protocol.

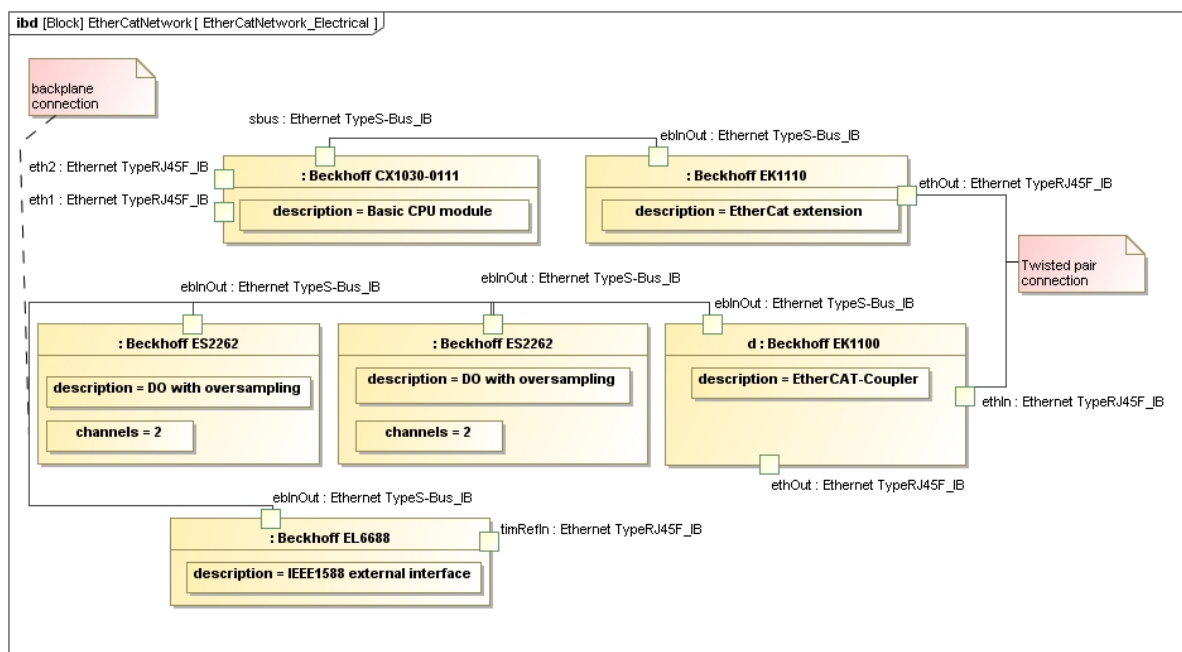


Figure 11: *Re-arranged EtherCat network.*

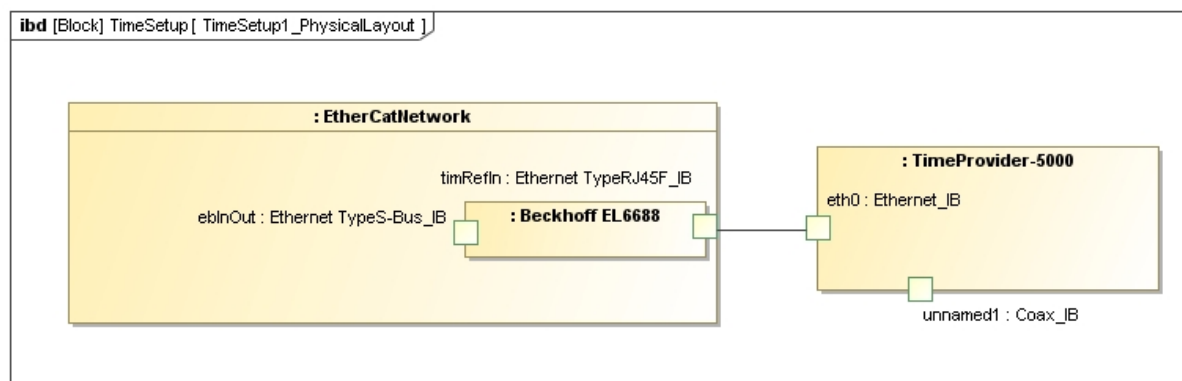


Figure 13: *Direct time server communication.*

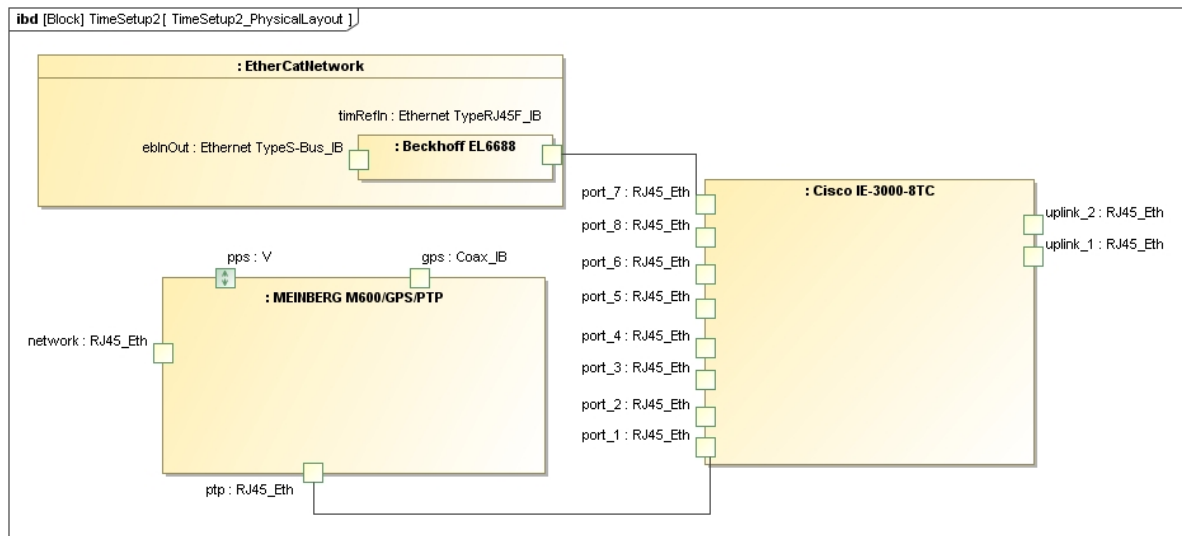


Figure 14: Time server communication through a Cisco switch.

Before start getting the time signal in the PLC it is required to configure few parameters of the EL6688 terminal. These parameters are shown in Figure 15, the other ones were not touched and they kept their default values.

Index	Name	Flags	Value	
F008	Code word	R/W	0x00000000 (0)	
F880:0	PTP Common	R/W	> 1 <	
F880:01	Precision Time Protocol	R/W	IEEE1588-2008 (PTPv2) - Slave ...	Definition of the protocol. We have used version 2 for our tests (Slave Only Clock).
F881:0	PTPv1 Settings	R/W	> 7 <	
F882:0	PTPv2 Settings	R/W	> 9 <	
F882:01	Transport Layer	R/W	LAYER 3 (PTP over UDP) (1)	Definition of the transport layer. We have selected UDP instead of Ethernet
F882:02	Domain Number	R/W	0x0000 (0)	
F882:03	Sync Interval	R/W	1 Second (0)	
F882:04	Delay Request Interval	R/W	1 Second (0)	
F882:05	Delay Mechanism	R/W	DISABLED (0)	
F882:06	Announce Interval	R/W	2 Seconds (1)	
F882:07	Announce Interval Timeout	R/W	3 Intervals (3)	
F882:08	Priority1	R/W	0x0080 (128)	
F882:09	Priority2	R/W	0x0080 (128)	
F8E0:0	Ethernet Settings	R/W	> 4 <	
F8E0:01	Address Type	R/W	FIXED (1)	
F8E0:02	IP Address	R/W	0xC0A8010F (-1062731505)	Ethernet settings in hexadecimal.
F8E0:03	Subnetmask	R/W	0xFFFF0000 (-65536)	
F8E0:04	Gateway	R/W	0x00000000 (0)	
F8F0:0	Vendor data	R/W	> 1 <	
FA80:0	PTP Diag	RO	> 15 <	

Figure 15: EL6688 Relevant configuration parameters.

It is also required to map the some input variables that will allow the computation of the time offset between the external source and the DC time as is shown in Figure 17. The *timestamp toggle* variable is used to make sure that there are two time updates before offset is computed.

With the *TwinCat System Manager* tool it is possible to verify the functioning of the EL6688 module and the external synchronization. The default list of process data objects includes the synchronization mode, the status of the external connection and the received time stamps among other parameters that can be monitored.

Mapping of terminal input variables allowing the computation of the TAI/DC time offset and the control of the time updates.

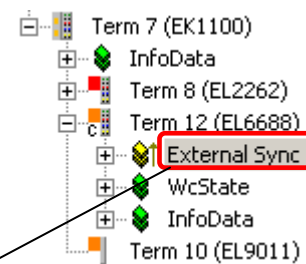


Figure 16: **EL6688 input values.**

>Name	Online	Type	Size	Address	In/Out	Us
Control Value for DC Master Clock	0x00000080 (128)	DINT	4.0	379.0	Input	0
Control value update toggle	0	BOOL	0.1	362.5	Input	0
External device not connected	0	BOOL	0.1	362.7	Input	0
External time stamp	X 0x01090000000000...	ULINT	8.0	371.0	Input	0
Internal time stamp	X 0x00000000000000...	ULINT	8.0	363.0	Input	0
Sync Mode	0x0 (0)	BIT2	0.2	361.0	Input	0
Time stamp update toggle	X 0	BOOL	0.1	362.6	Input	0

Figure 17: **External synchronization parameters.**

#### 4.4.2.1 EtherCAT Configuration

In order to use the time from the IEEE1588 network it was required to configure master accordingly. The above means that DC must be in use and the option 'DC time controlled by External Sync Device' must be selected.

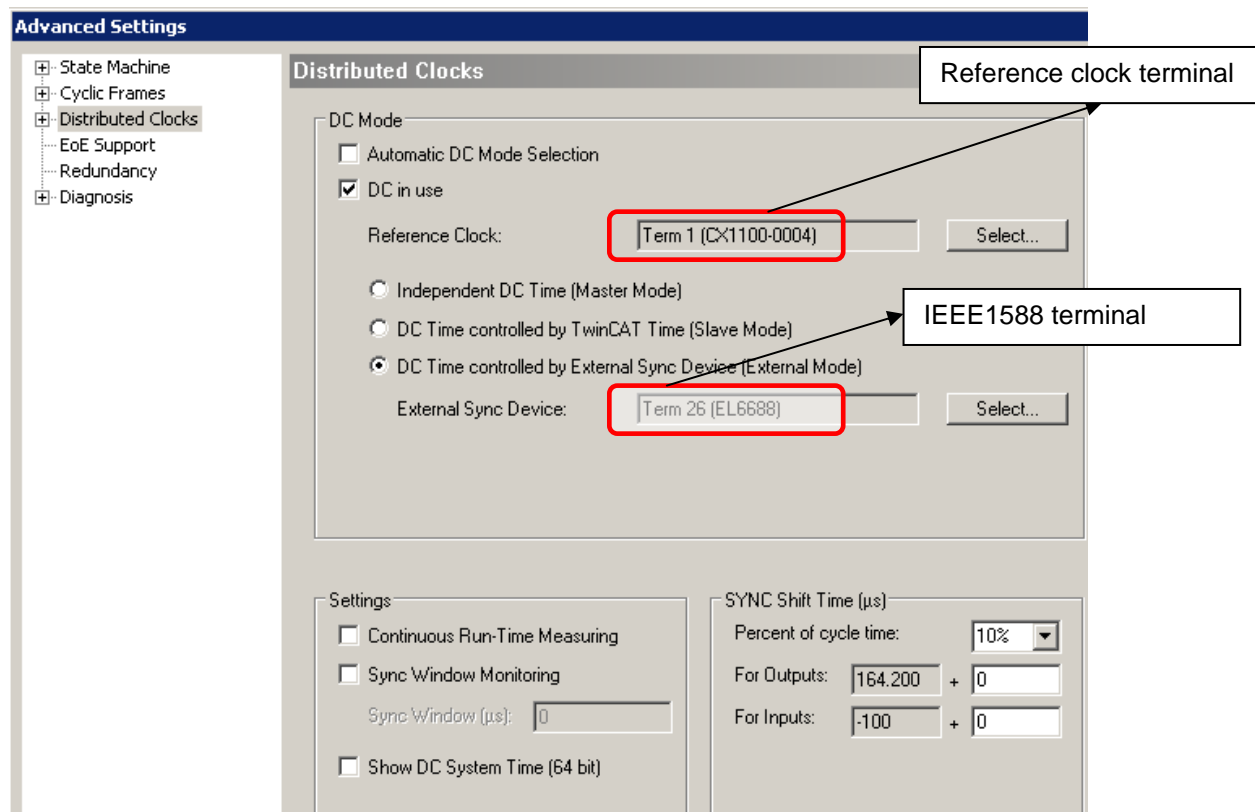




Figure 18: **EtherCAT DC configuration.**

### 4.4.3 Results

#### 4.5.3.1 Terminal Version

One of the problems found during the tests was that only the newest EL6688 terminal worked properly. An old one with a different hardware and software version never received the time signal. Even though after installing the latest firmware version on the old terminal, the module still did not work so no further attempts were tried.

Terminal	HW version	SW version	Status
EL6688	5	5	
EL6688	3	3	

#### 4.5.3.2 Time Server Connection

It was first tried to access the absolute time connecting the EL6688 directly to the Symmetricom time server, see Figure 13. After having a slow update rate of the time in the PLC it was investigated with Wireshark the traffic of PTP packets. It was discovered that time updates delivered by the time server were every 128 seconds. The configuration of the time server was checked through the serial interface using HyperTerminal tool but all parameters were found correct. Additionally we have tried to use the Symmetricom time server in combination with the Cisco switch IE-3000 but in that case the time signal did not arrive to the Beckhoff terminal.

Later, it was used the Meinberg time server and this one worked immediately after it was connected, directly or through the Cisco switch. The test setup using the Meinberg time server is depicted in Figure 14.

#### 4.5.3.3 Reading DC Time

In the PLC there are 4 possibilities to access the current DC time during task runtime (PLC cycle).

- Readout of the **DcSysTime** input variable in the input data of the EtherCAT device. This input data comes from the EtherCAT master but is copied from the reference clock.
- **F\_GetActualDcTime**: Get the actual DC time. In the case of a repeated call within a task, this function returns different values.
- **F\_GetCurDcTickTime**: Moment of the last base time. A repeated call only returns different values if there is at least 1 base time in between.  
*F\_GetCurDcTaskTime*
- **F\_GetCurDcTaskTime**: It delivers the start time of the task.

The evaluation of the above last three FB methods are depicted in Figure 19:

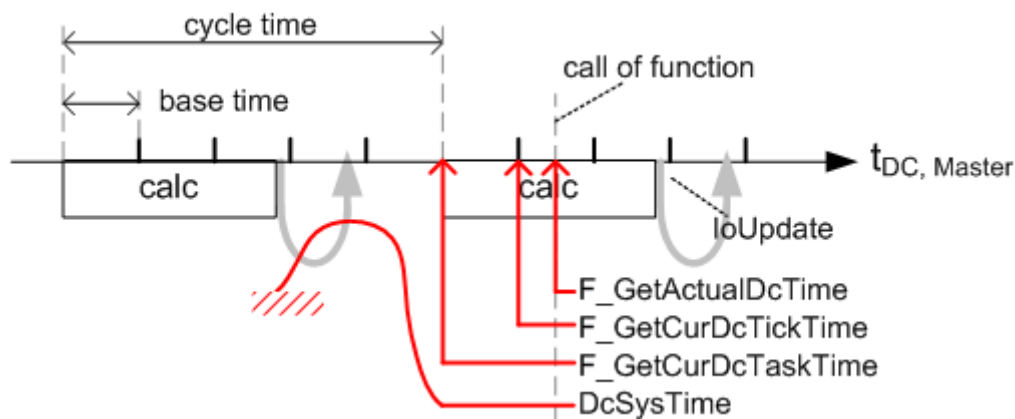


Figure 19: **PLC functions for reading DC time.**

Unfortunately none of these three functions worked initially so it has been used the contents of the `DcSysTime` variable that comes from the reference clock to obtain the DC time inside the PLC cycle. When doing that, we faced another problem, the reference clock must support 64 bits otherwise the variable can store only 4.2 seconds and therefore cannot be used for computing the absolute time. So it was required to re-arrange our EtherCat network and put a 64 bit DC capable terminal as the first one in the segment, see Figure 11. Please note that EtherCAT selects automatically the first DC capable terminal as reference clock so modules must be configured to be used as potential reference clocks.

After contacting Beckhoff support it was communicated to us that those functions only work with TwinCat version 2.11 above build 1545 which was not so clear from the Beckhoff documentation. It was also recommended not to use the `DcSysTime` input variable to obtain the DC time since it could be one cycle older than a call of the functions inside the PLC program. Finally after installing a newer build of TwinCat 2.11 (build 2033) it was possible to use the function `F_GetCurDcTaskTime` and thus obtain the DC time from the PLC program. This is the function recommended by Beckhoff to be used as the DC base time.

#### 4.5.3.4 UTC Time

The absolute time is obtained by getting the offset between the external time, coming from the IEEE1588 network, and the internal DC time of the same module. Since the IEEE1588 time protocol delivers TAI time, the time has to be adjusted to UTC by subtracting the leap second, which currently is 34 seconds. For simplicity, the leap second has been hardcoded in the PLC program. So at each PLC cycle, the absolute time is computed as follows:

$$\text{Time Offset} = \text{Internal DC time} - \text{External time} + \text{Leap time}$$

$$\text{Absolute Time (UTC)} = \text{DC time} - \text{Time Offset}$$

It was implemented a FB (`FB_DcAbsoluteTime`) in ST language that converts the DC time in absolute UTC time. Since internally DC time is stored as two 32 bits words it is required to use special functions to operate these 64 bits numbers.



It was observed that synchronization takes few seconds when the external time signal is connected for the first time. The input variable 'Time stamp update toggle' can be used to determine when a time updated has occurred inside the PLC program, see Figure 17.

#### 4.5.3.5 Time Synchronization

It was repeated the test to measure the time synchronization between two output signals but this time using the external time reference from terminal EL6688. The two signals had the same precision as measured before with DC (< 10 [nsec]) but only when they were coming from the same terminal. Few other combinations were configured and the results are described in Table 1.

Table 1: Time synchronization results

Signals	Method	Precision
Two signals from the same terminal	DC	< 10 [nsec]
Two signals from the same terminal	External time reference	< 10 [nsec]
Two signals from different terminals and from different segments	DC	< 10 [nsec]
Two signals from contiguous terminals (same segment)	External time reference	< 200 [nsec]
Two signals from different terminals and each terminal located in different segments.	External time reference	< 1 [μs]

These results are consistent with nominal precision specification from Beckhoff [RD8]:

DC Precision: < 100 [nsec]

DC + External synchronization Precision: < 1 [μs]

## 4.5 Timer Definition

### 4.5.1 Description

This tests aims to verify how to define timers that mimic the functionality of the TIM board timers.

### 4.5.2 Results

It has been implemented a GUI using the TwinCat GUI tool to provide user interaction to define up two timers. The display shows the absolute time, DC time and individual status of each timer. The user can specify the period and the start time (UTC) of the timers. This small GUI works in combination with the PLC program that implements the oversampling and the internal conversion to absolute time.

The screenshot shows a graphical user interface titled "TIMER\_GUI". Inside, there's a section titled "PLC TIMER Example". Under "Status:", there are two time fields: "Absolute Time:" with value "2011-09-20-06:19:14.079838784" and "DC Time:" with value "2011-09-20-06:17:39.689030791". Below these are two timer status boxes. "Timer 1:" shows "Status: Not Armed Not Active". "Timer 2:" shows "Status: Not Armed Not Active". Under "Settings:", there are two more fields: "Period [microseconds]:" with value "250" and "Start Time:" with value "2011-09-08-15:40:00.0".

Status:	
Absolute Time:	2011-09-20-06:19:14.079838784
DC Time:	2011-09-20-06:17:39.689030791
Timer 1:	Timer 2 :
Status:	Status:
Not Armed	Not Armed
Not Active	Not Active

Settings:	
Period [microseconds]:	250
Start Time:	2011-09-08-15:40:00.0

Figure 20: *Timer example application.*

## 4.6 High Level Time Synchronization

### 4.6.1 Description

One of the advantages of having an Ethernet based time synchronization network is that high-level computers like Workstations can connect directly to this time network. This is an advantage compared with the current VLT scheme where it is used an additional protocol (NTP) to synchronize all WS in the control network. The purpose for this test is to investigate PTP implementations for Linux and the implementation of tracking devices directly in the WS instead of the PLC. To achieve the above we have added an additional network card to our linux WS (te67) that it is connected to the dedicated time network.



## 4.6.2 Test Setup

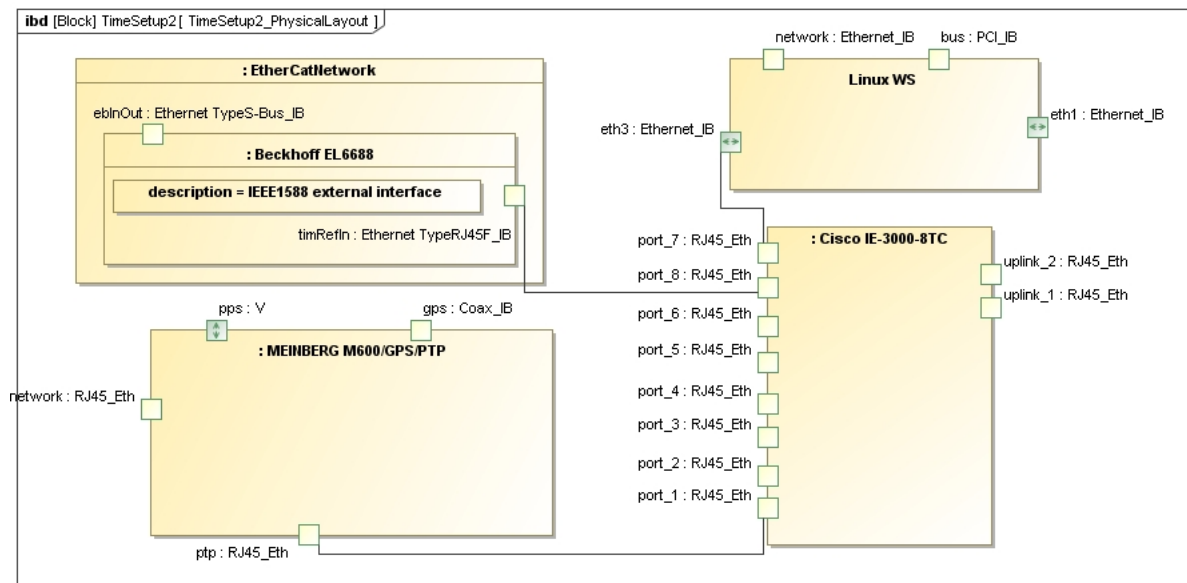


Figure 21: **High-level network setup.**

## 4.6.3 Results

Two different software implementation of the IEEE1588 protocol stack have been deployed and tested:

- PTPd
- PTPv2 implementation from InEs ZHAW.

### 4.6.3.1 PTPd

PTPd is open source software. The source code is available under same BSD-style license as NTP. The project is hosted on SourceForge at [ptpd.sourceforge.net](http://ptpd.sourceforge.net). It was not easy to find the PTPd that implements IEEE1588-2008 version. Most of the available ptpd on internet supports only PTP version 1. PTPd software, including the protocol stack and the clock servo, runs as a background user-space process and requires to be launched with root privileges.

PTPd interfaces with the kernel through standard Linux system calls. Received time stamps are recorded in the Network Interface Card (NIC) driver, in or close to the receive interrupt handler. The receive time stamps are passed to user-space through an `ioctl()`. PTPd uses the Linux kernel's software clock along with the `adjtimex()` interface for clock tick-rate adjustment.

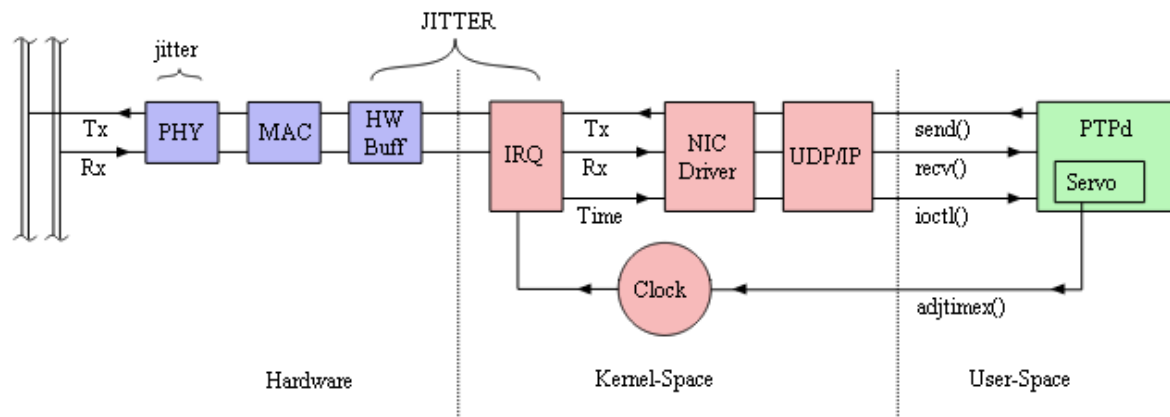


Figure 22: **PTPd data flow diagram.**

Its main advantages are:

- It is very simple and can run as a background process without the need of using extra libraries from the applications.
- Its code is open and it can eventually be adapted.
- It is now available in linux kernels.

Its disadvantages are:

- The support is very limited since this is an open source project which is currently not very active.
- It is only a software solution which limits the time accuracy.

#### 4.6.3.2 PTPv2 from InEs ZHAW

PTPv2 is an API implementing the IEEE1588-2008 protocol stack developed by the InEs of ZHAW. ESO has procured the source code of this implementation and this code has been compiled under Linux. It has been used a sample application that uses the API but runs in background mode. It was used just the soft version of the API which means no access to HW timestamps.

In order to use the sample application you have to specify the IP number of the subnet connected to the IEEE1588 network, for instance:

```
./ptp2_linux 192.162.1.10
```

The application must run with root privileges and it has an interactive menu with a number of options allowing the reading of the current time and debugging protocol messages among other capabilities.

```

/-----\
| IEEE1588 PTPv2 Application for ZHAW / InES IEEE1588nic |
| (c) 2008 Zurich University of Applied Sciences         |
|               Institute of Embedded Systems           |
|               Contact: <1588support.ines@zhaw.ch>      |
|-----|
| Specify the IEEE1588 capable PCI Ethernet Card's IP-address |

```

```
| as parameter. Example: ptp2_linux_1588nic 192.168.100.11 |
|-----|
| Press 'h' for help, 'v' for verbosity levels, 'q' to quit |
\-----/
```

Searching interface for 192.168.1.25...

found eth2 with IP 192.168.1.25

Version 1.1\_release\_release

Receive thread started

Timer thread started

Keyboard listener thread started

\*\*\* PTPv2 for IEEE1588nic up and running \*\*\*

t

\*\*\* Current Time \*\*\*

sec: 1324023866 nsec: 987040000

### Protocol Debugging (option: v 8)

```
PTP MSG: RX sync with Seq_ID: 39232 from      0 60 6E FF FF 7C 23 1C on port 1
PTP MSG: RX follow up with Seq_ID: 39232 from  0 60 6E FF FF 7C 23 1C on port 1
PTP MSG: RX follow up -> Synchronise local clock
PTP MSG: RX announce from 0 60 6E FF FF 7C 23 1C on port 1
PTP MSG: RX sync with Seq_ID: 39233 from      0 60 6E FF FF 7C 23 1C on port 1
PTP MSG: RX follow up with Seq_ID: 39233 from  0 60 6E FF FF 7C 23 1C on port 1
PTP MSG: RX follow up -> Synchronise local clock
PTP MSG: RX sync with Seq_ID: 39234 from      0 60 6E FF FF 7C 23 1C on port 1
PTP MSG: RX follow up with Seq_ID: 39234 from  0 60 6E FF FF 7C 23 1C on port 1
PTP MSG: RX follow up -> Synchronise local clock
PTP MSG: RX announce from 0 60 6E FF FF 7C 23 1C on port 1
PTP MSG: RX sync with Seq_ID: 39235 from      0 60 6E FF FF 7C 23 1C on port 1
PTP MSG: RX follow up with Seq_ID: 39235 from  0 60 6E FF FF 7C 23 1C on port 1
PTP MSG: RX follow up -> Synchronise local clock
PTP MSG: RX sync with Seq_ID: 39236 from      0 60 6E FF FF 7C 23 1C on port 1
PTP MSG: RX follow up with Seq_ID: 39236 from  0 60 6E FF FF 7C 23 1C on port 1
PTP MSG: RX follow up -> Synchronise local clock
```

Its main advantages of this implementation are:

- Availability of a quality support considering the deep knowledge of the protocol of the people from InEs.
- It can work with a NIC supporting HW time stamps which would increase the time synchronization. It requires a specific linux driver which is also delivered when buying the NIC from them.

- It provides good debugging capabilities.

Its disadvantages are:

- The solution is more complex than PTPd and there are not many examples how to use the API.
- The API only works for C/C++ applications.

## 4.7 Tracking Device

### 4.7.1 Description

The aim of this test was to implement a tracking device running in the WS and using the absolute time obtained from the IEEE1588 network to compute the motor corrections that will compensate for the field rotation during the tracking loop.

### 4.7.2 Results

The two PLC platforms currently being evaluated for replacing the LCUs have a technical limitation which is that they cannot run C/C++ code, at least not from the PLC run-time. The above means that we cannot use some critical libraries like *slalib* to compute the field rotation compensation based on the telescope coordinates and the absolute time. On the other hand having a fast communication mechanism between the WS and the PLC provided by OPC UA and accessing the absolute time directly from the WS would allow the implementation of these devices in the WS instead of doing it at lower level (PLC).

A prototype of a derotator device was implemented using the fieldbus extension and tested with a stepper motor. The device interface and functionality is similar to the LCU version. The tracking frequency achieved was up to 20 Hz which is higher than the ones currently used by the same devices in the LCU (maximum 10 Hz). SL is not a real-time linux distribution so it is expected some jittering in the tracking loop and some latency sending the updated position to the PLC via OPC UA. However this shall not affect significantly the overall performance of the tracking for these small optical devices. Unfortunately we did not have a high-resolution motor that could be used for performance tests to validate the accuracy of the tracking loop at low tracking speed and crossing the meridian. Nevertheless preliminary tests showed that this could be a valid solution to overcome the limitation of the PLCs until the integration of C/C++ code will be provided by the PLC vendor solutions.

The prototype code has been encapsulated in a VLTSW module (*liifbi*) and archived in SVN with the Laboratory Instrumentation SW (LINS).

## 4.8 VLTSW Demo Application

In order to simplify the control of the time synchronization and the operation of the tracking device it was implemented a graphical interface using the VLTSW. This GUI allows setting the interval and the start time for the timers and defining the operational mode and position for the derotator device, see Figure 23.

---

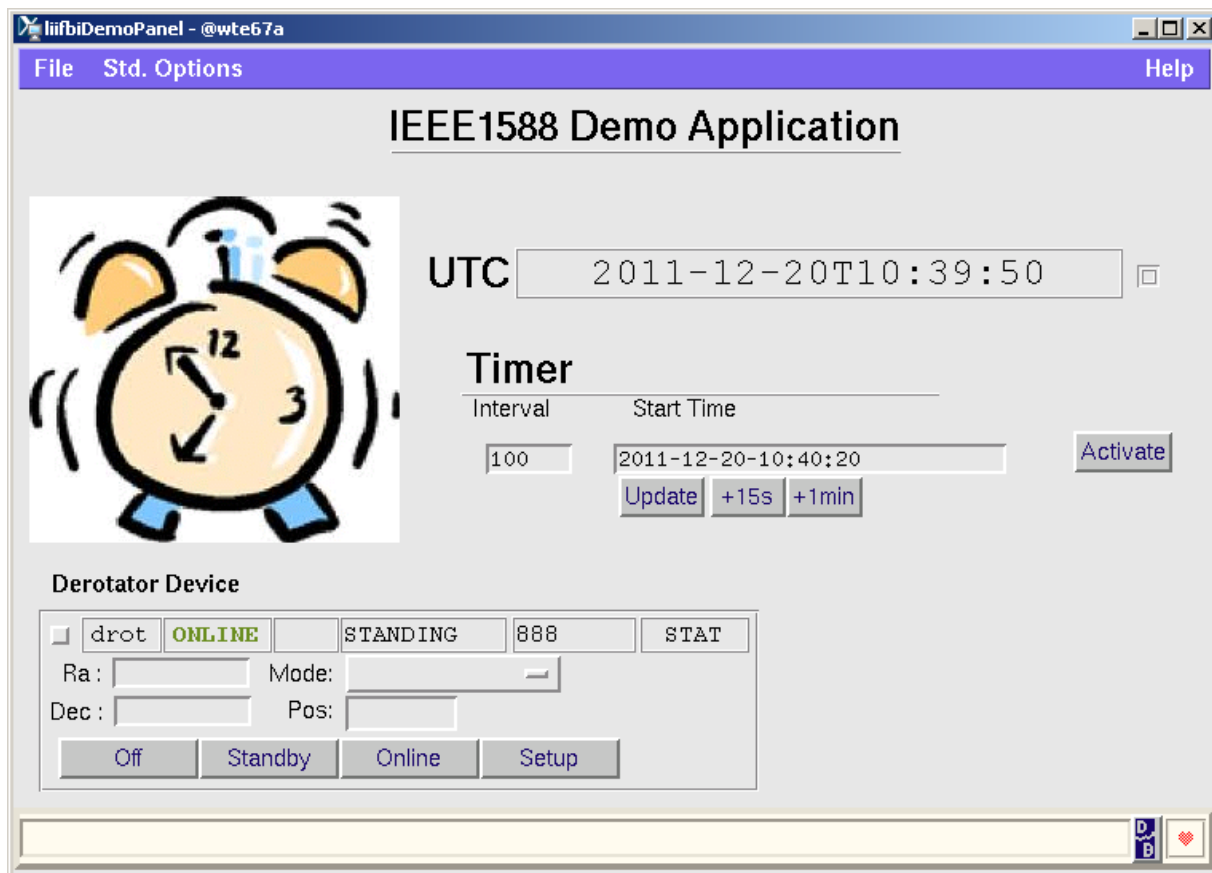


Figure 23: *Test GUI application.*

## 5 Conclusions

### 5.1 Time Servers

- The Symmetricom time server did not perform very well and it could not be used in combination to the Cisco switch while Meinberg time server worked without any problem.

### 5.2 Time Synchronization

- Internal time synchronization of EtherCat, based on distributed clocks, is very good. It has been measured a difference of less than 10 [nsec] between two output signals within the same EtherCat network. The synchronization precision goes down when using an external time reference (IEEE1588) but still is better than one microsecond in the worst case. This precision is more than enough to satisfy the requirements for hardware synchronization defined by MATTISE instrument and probably for all future instruments.
- It was possible to access the absolute time from the PLC through a gateway terminal (EL6688) and program one-shot action inside the PLC program with 1 [μs] time resolution.
- It was possible to implement a continuous pulse train with a time resolution up to 1 [μs] achieving a frequency of 0.5 MHz. This allows the implementation of timers which can be used for continuous hardware synchronization.

### 5.3 High-level Synchronization

- It was possible to deploy and test the IEEE1588 protocol stack under Linux. It has been tested two solutions both using PTP version 2: one open source (PTP daemon) and one commercial from the Institute of Embedded Systems (InES) from the Zurich University of Applied Sciences (zhaw).
- It was possible to implement a prototype of a tracking device (derotator) running in the WS, using the absolute time coming from the IEEE1588 network and sending continuously position corrections to the PLC during tracking. It has been tested with a Beckhoff stepper motor achieving a tracking loop frequency up to 20 Hz.

### 5.4 Next Steps

- It would be recommended to setup a similar architecture as described in Figure 1 and measure the hardware synchronization between a TIM board and a PLC digital output signal controlled by a PLC. This would be an important milestone to verify the synchronization between old time reference system and the one based on IEEE1588.
  - Measure external time synchronization between two Ethernet networks connected through a EtherCAT bridge.
  - It would be desirable to have a high resolution motor setup to carry out performance tests with tracking devices running in the WS and thus validate the feasibility of implementing tracking loops at high-level.
  - Test the IEEE1588 protocol under linux in combination with a PTP capable NIC so that it would be possible to achieve sub-microsecond time resolution. It was investigated providers of Ethernet controllers with support for IEEE1588 and it was found that it is not so difficult to find them on the market, few examples are listed here:
    - IEEE1588 enabled PCI Ethernet NIC from InEs
-

- Intel Ethernet controllers supporting hardware-enabled IEEE 1588, for instance NIC 82574, 82576, and 82580.