Application Note

EtherCAT® Slave Controller Development Products

Frequently Asked Questions and Troubleshooting

ET1100
ET1200
EtherCAT IP Core

Version 1.1
Date: 2011-12-15

BECKHOFF

DOCUMENT HISTORY

Version	Comment
1.0	Initial release
1.1	• Added chapter on missing RSA key in Xilinx® EDK • Added error counter interpretation guide • Added appendix on showing error counters in TwinCAT

Trademarks
Beckhoff®, TwinCAT®, EtherCAT®, Safety over EtherCAT®, TwinSAFE® and XFC® are registered trademarks of and licensed by Beckhoff Automation GmbH. Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

Patent Pending
The EtherCAT Technology is covered, including but not limited to the following German patent applications and patents: DE10304637, DE102004044764, DE102005009224, DE102007017835 with corresponding applications or registrations in various other countries.

Disclaimer
The documentation has been prepared with care. The products described are, however, constantly under development. For that reason the documentation is not in every case checked for consistency with performance data, standards or other characteristics. In the event that it contains technical or editorial errors, we retain the right to make alterations at any time and without warning. No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

CONTENTS

1   Introduction

Purpose of this document is to answer common questions regarding EtherCAT development products and to avoid common implementation mistakes which came up in the past.

This document is aiming at the following Beckhoff ESCs:

- ET1200 (up to ET1200-0002)
- ET1100 (up to ET1100-0002)
- EtherCAT IP Core for Altera$^®$ FPGAs (up to V2.4.0)
- EtherCAT IP Core for Xilinx$^®$ FPGAs (up to V2.04a)

2   Frequently unasked questions

2.1   What information should I provide when I need support?

Sometimes the documentation and this FAQ do not offer a solution for your problem and you need support. You can improve your support experience by providing precise and complete information, which helps us identifying the problem faster.

Please provide the following information when contacting the support:

- Detailed error description with the following background information (if appropriate):
    - network topology
    - how often does the error occur
    - ESC type (ET1100, ET1200, IP Core, other)
    - ESC version (ASICs: e.g. -0002, IP Core: Altera/Xilinx and version)
    - are frames lost, are error counters increasing (refer to chapter 3.12)
- IP Core: IP Core configuration (Altera: MegaFunction source, Xilinx: .eccnf file)
- **IP Core: complete report log files: do not extract single lines – the context of the error is very important.**
- Schematics of the ESC/FPGA and the PHYs
- IP Core: top level schematics/sources of the FPGA which shows EtherCAT IP Core connections.
- EtherCAT master type and version (if applicable)

3   General Issues

3.1   Where can I find documentation updates?

Documentation updates are available for download at the BECKHOFF website:
http://www.beckhoff.de/english/download/ethercat_development_products.htm

3.2   ESC clock source accuracy: Is 25 ppm necessary?

Since standard PCs can be used as a master, the clock source deviation of the master's Ethernet PHY can be up to 100ppm. This is tolerated by the ESCs (according to IEEE 802.3) if the FIFO size is default. Nevertheless, EtherCAT requires 25 ppm clock accuracy for the EtherCAT slaves in order to enable FIFO size reduction.

FIFO size reduction is possible in any slave with high accuracy except for the first one directly attached to the master. This can reduce network latency significantly, so, 25 ppm is mandatory. Additionally, the clock accuracy has influence on the DC settling time and the forwarding latency jitter.

3.3   Why should port 0 never be an unused port?

Refer to the special features of port 0: If a frame travels through a slave with port 0 automatically closed (because it is unused, the link state is fixed at no link), the Circulating Frame bit is set in the frame. If this frame comes to a second device which has also port 0 automatically closed (because unused), the frame will be destroyed.

So, any network containing more than one of the slaves with port 0 unused will experience impossible communication. Additionally, in case of network trouble, masters might try to close ports in order to isolate faulty network segments and to maintain communication in the rest of the network. If port 0 is not used, the master is connected via port 1, 2 or 3. The master might try to close ports 1, 2, or 3 because it expects to be located at port 0, which results in network isolation.

Refer to the ESC datasheet section I for more information on port 0 special behaviour.

3.4   Link/Activity LEDs shows strange behavior

LED is on without anything attached, LED is off with other device attached. If all ports are connected, communication with the slave is possible, although following slaves are not seen.

This happens if the LINK_MII polarity is incorrect, i.e. the link information is inverted for the ESC. The communication with the slave is possible if all ports are connected, because the ESC detects that no port has a link, and then it automatically opens port 0. You can check this in ESC DL status register 0x0110: no physical link at all ports (although you can obviously read through port 0).

3.5   Can slaves communicate without SII EEPROM / invalid SII EEPROM content?

Yes! EtherCAT communication is possible; all frames can be forwarded without a problem. Register reading/writing is possible, but process data exchange is impossible. The ESC blocks Process Data memory until the EEPROM is successfully loaded (typically no PDI is configured unless the SII EEPROM is loaded successfully). You can do the initial programming of the SII EEPROM via EtherCAT (e.g. for hundreds of slaves with "empty" SII EEPROM in parallel (broadcast).

3.6   Do I need the complete XML ESI description for simple PDI read/write tests?

No, you do not need a complete ESI for PDI read/write tests. Only the first 8 words of the SII EEPROM need to be written (e.g. via the network with values from the pinout configurator, the IP core configurator, or the ESI EEPROM CRC calculator). This enables the PDI interface. Such a device cannot be correctly identified by TwinCAT, but basic frame exchange (register view etc.) is possible.

3.7   What do I do with unused ports (EBUS/MII)

EBUS
Attach RX resistor, leave TX signals open. If EBUS is not used at all, the tolerance of the RBIAS resistor is increased to 1kΩ (i.e., you could set RBIAS to 10 kΩ).

MII
Set LINK_MII to indicate no link (value depends on link polarity configuration), leave TX signals open, pull-down RX signals (should not be left floating). Leave pull-up for MDC/MDIO in the design, so that the interface can detect the state correctly..

3.8   Resetting ESC, PHYs, and µController

All Ethernet PHYs should be hold in reset state until the ESC leaves reset state. With the EtherCAT ASICs, make use of the nRESET signal. With FPGAs, you should route the nRESET signal out of the FPGA. Make sure the PHYs are in reset state while the FPGA is loading. If FX transceivers are used, make also sure that the transceivers are disabled (powered down) while the ESC is not operational (refer to the AN "PHY selection guide" for more details).

The reason is that otherwise the PHYs will establish a link with the neighbours, these neighbours will open their link, but communication is not possible because the ESC is not operational. The result will be frame loss, which is not desired.

If EEPROM emulation is used, the µController should be able to process EEPROM commands once communication is possible, otherwise an EtherCAT master might time-out.

A µController attached to the PDI should also be reset by the ESC (e.g. by a soft reset via register 0x0040:0x0041).
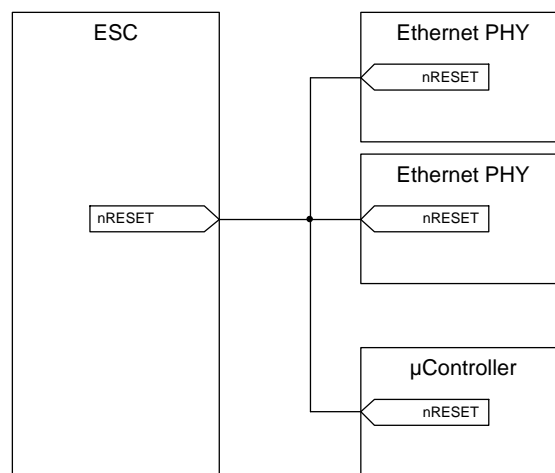
Figure 1: Reset connection principle

Refer to the ESC datasheet and the AN PHY selection guide for more information.

3.9   Should I enable Enhanced Link Detection?

For MII ports, a precondition for Enhanced Link Detection is that the MII management interface is operating with all PHYs (PHY address configuration ok), which is strongly recommended anyway. If the ESC has only MII ports, Enhanced Link Detection is recommended regardless of the PHY link loss reaction time. If the Enhanced Link Detection can be enabled port-wise, it is recommended to enable it for each MII port.

If the ESC has one or more EBUS ports, Enhanced Link Detection must remain disabled for ET1200 and ET1100. This is also true if the PHY offset for ET1200 or ET1100 is not zero.

Refer to the ESC datasheet section I for more information about Enhanced Link Detection.

3.10   Why must I configure the PHYs for auto-negotiation instead of forcing 100 Mbit+FD?

EtherCAT requires that the Ethernet PHYs are configured to use auto-negotiation, and not 100 Mbit + full duplex in force mode. The reason is interoperability.

While two devices which are forced to 100 Mbit + full duplex (FD) are perfectly operating together with EtherCAT, a problem arises if one device uses auto-negotiation, and the other one forced mode. In this case, the auto-negotiation device will fall back to 100 Mbit + half duplex (HD), because it has no knowledge of the capabilities of the link partner. The half-duplex operation can lead to communication issues especially at the master, but probably also at the slaves. That is why auto-negotiation is mandatory for EtherCAT PHYs.

Another issue is the Auto-MDIX feature of the PHYs, which often depends on auto-negotiation being enabled. So, without auto-negotiation, some EtherCAT links would require a cross-over cable, others a straight cable, which complicates the situation even more.

Refer to the AN PHY selection guide for more information.

3.11   What is TX Shift and Auto TX Shift?

Beckhoff ESCs are not incorporating transmit (TX) FIFOs for the MII interfaces to reduce the latency. Nevertheless, the PHYs have certain timing requirements for the TX signals (setup/hold), which have to be fulfilled. The TX Shift configuration is used to move the TX signal timing to a valid window manually. With Auto TX Shift, the ESC automatically figures out the correct shift value and shifts the TX signals accordingly. Nevertheless, Auto TX Shift is not the same as a TX FIFO, it still requires synchronous clocking of the ESC and the PHY.

Refer to the ESC datasheet for more information about TX Shift and Auto TX Shift.

3.12 Frames are lost / communication errors are counted

During power-up, it is a normal behaviour of the ESCs to count a few communication errors. You can typically ignore them, only new errors which occur during operation are significant. An exception is the PDI error counter, which is expected to be zero after power-up.

Generally, no real bus system is completely error-free, i.e., the bit error rate (BER) is never zero. A few individual errors over time are tolerated by EtherCAT. It needs several errors or lost frames in a row to interrupt the operation of a part of the network. Again, the PDI error counter is an exception, it has to be zero all the time.

If frames are lost or error counters are increasing, configure TwinCAT to show the ESC error counters and use the error counter interpretation guide in this chapter.

Refer to the ESC datasheet section I and II for more information on the ESC error counters.

3.12.1 Configuring TwinCAT to show the ESC error counters

Refer to the appendix (chapter 6.1) for details on configuring TwinCAT for displaying all the ESC error counters. This helps in identifying the problem.

3.12.2 Reduce the complexity

Try reducing the complexity of the problem, which helps in localizing the problem cause:

- Try reducing the network size to a minimum, while the error is still present.
- Disable Enhanced Link Detection for testing to improve error visibility.
- Swap devices in the network to isolate the error source.

3.12.3 Basic error counter interpretation

At first, check the ESC configuration, e.g. by viewing the POR registers 0x0E00:0x0E01 or the IP Core configuration in the User RAM 0x0F80. If the configuration is wrong, check the strapping options and power-on signal values during POR sampling.

Then, check the link status in the ESC DL Status register 0x0110. If there is something wrong with the link status, check the LINK_MII signal. If this is also ok, the problem is in the area PHY – connector – cable – connector – PHY.

If an RX Error counter (0x0301, 0x0303, 0x0305, 0x0307) is increasing, there is a problem in the area PHY – connector – cable – connector – PHY. Measure the RX_ERR signal from the PHY to prove the error counter: if it is sometimes high, check the cabling, connectors, and PHY. Measure the RX_ERR signal from the PHY. The ESC is not the source of the problem.

If there is no RX Error, there is probably a master issue. Use a proven master, attach also proven slaves. Remove DUT; move the DUT in the network by swapping positions and check if the error is moving with the DUT.

If the issue is not a master issue, it must be in the area PHY – ESC – PHY, i.e., check the interface configuration of the PHY, the connections itself, the configuration of the ESC, and – in case of the IP Core –, if the ESC is correctly constrained and the timing is achieved.

Also think of power supply break-down, clock jitter, and glitches on the reset signal.

3.12.4 Error counter interpretation guide

The following diagram can be used to identify possible error causes. At first, find out which error counters are increasing in your network (or device), and at the master. Compare this with each of the cases shown in the main table of Figure 2 and find the most similar case. Follow the arrows at the bottom of each column to the possible error causes. Use Table 1 for comments and error localization.

Error counter	Register	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Lost frame	Master			+	+		+			+	+		+	+	+	+		
TwinCAT RX Err	Master					+	+											
TwinCAT TX Err	Master	+																
ESC CRC Err	0x0300 (+2/4/6)				--					+	+		+				+	
ESC RX Err	0x0301 (+2/4/6)				--			+	+	+	+							
ESC Forw. Err	0x0308 (+1/2/3)		(+)		--					(+)	(+)		(+)	(+)	(+)	+	(+)	(+)
ESC EPU Err	0x030C		(+)		--					(+)	(+)		(+)	(+)			(+)	
ESC Lost link	0x0310 (+1/2/3)				--				(+)		(+)	(+)						

+	Increasing
(+)	Might be increasing/ available
--	No data / no communication
●→	Common cause
●--→	Unusual cause
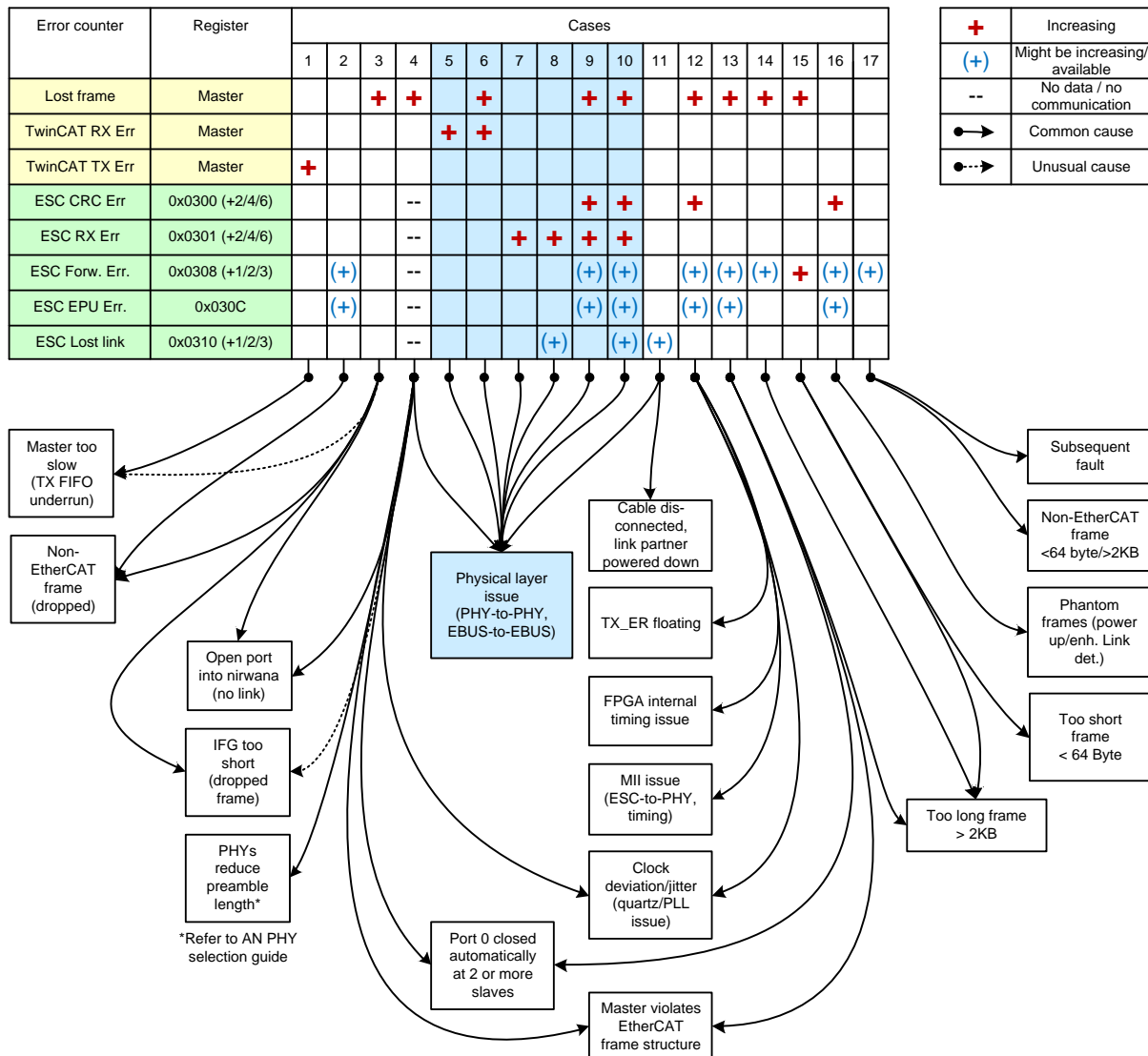
Figure 2: Error counters interpretation guide

Table 1: Error counters interpretation comments

Case	Comment	Localization
1	Master issue	Master
2		Master/slave to slave with increasing EPU error cnt., or between the two slaves in front of slave with increasing Forw. error cnt.
3		
4	no communication to the slave, error counters cannot be read	Last slave with stable communication to slave without communication
5	RX_ER during IDLE	from 1st slave back to master
6	RX_ER during frame	from 1st slave back to master
7	RX_ER during IDLE	Slave to slave with RX err. cnt. increasing
8	RX_ER during IDLE, link down by Enhanced Link Detection	Slave to slave with RX err. cnt. increasing, or last slave with stable communication to slave without communication
9	RX_ER during frame	Slave to slave with RX err. cnt. increasing
10	RX_ER during frame, link down by Enhanced Link Detection	Slave to slave with RX err. cnt. increasing, or last slave with stable communication to slave without communication
11		Slave to slave with Lost link cnt. increasing
12		Slave to slave with CRC err. cnt. increasing, or between the two slaves in front of slave with increasing Forw. error cnt.
13		Slave to slave with increasing or Forw. error cnt., or between the two slaves in front of slave with increasing Forw. error cnt.
14		Any slave to slave at or before the slave with increasing Forw. error cnt.
15		Any slave to slave at or before the slave with increasing Forw. error cnt.
16		Slave to slave with increasing or CRC error cnt
17	Subsequent fault, typically not source of errors	Between the two slaves in front of slave with increasing Forw. error cnt.

3.13 PDI Performance

The PDI interface performance cannot be compared with the performance of a simple dual-ported memory, mainly because of the SyncManagers. The SyncManager buffer mechanism coordinates EtherCAT and PDI access to the memory and registers by the means of buffer re-mapping, enabling/disabling accesses, and interrupt/error handling. Especially the dependency between EtherCAT frame processing and PDI reduces PDI performance.

The theoretical maximum throughput of any PDI is (1 Byte / 80 ns) = 12.5 Mbyte/s (equal to the maximum Ethernet throughput). Additional latency is required for synchronization (clock domain crossing, EtherCAT frame start) and read/write collision detection.

The datasheet figures are worst case timing, min./max. and average read/write times for the asynchronous µC PDI while using the BUSY signal are (ET1100, preliminary):

Read 16 bit: min. 195 ns,   average 235 ns,   max. 575 ns

Write 16 bit: min. 160 ns,   average 200 ns,   max. 280 ns

The maximum read time includes a postponed 16 bit write access, a 16 bit read access, phase alignment and read/write collision. A µController has to cope with this maximum latency, but it is not the average latency! If large amounts of data are to be transferred in one direction (either read or write), the latency is coming close to the average value.

All PDIs are using the same internal interface, thus the maximum throughputs of the PDIs are very similar, and even the on-chip bus PDIs are comparable. That's because they are all limited by the internal PDI/SyncManager realization.

The synchronous µC interface is based on the asynchronous uC interface, plus additional output synchronization, i.e., it is slightly slower. The SPI PDI achieves the maximum throughput for accesses with a large number of bytes. The maximum times for the on-chip bus interfaces (Avalon/OPB/PLB) are slightly better, because there is no need of synchronization (the throughput degrades with smaller access width and with a faster bus clock), although the average throughput is similar.

In most situations, the PDI performance is sufficient, at least if the read and write accesses are properly grouped and timed in relation to the network cycle time.

3.14 Interrupts

3.14.1 µControllers with edge-triggered Interrupt / only the first interrupt is processed

The Beckhoff EtherCAT ESCs generate a level-triggered interrupt signal, which combines many internal interrupt sources (AL Event Request) into one interrupt signal (according to the AL Event Mask).

If a µController with edge-triggered interrupt input is connected to this ESC, care has to be taken that all interrupts are processed by the interrupt service routine before it is left. It is especially possible that additional interrupts are coming up while the ISR is processed. If the ISR does not check the AL Event Request to be empty before it is left, the ISR might not be called anymore at all, because the interrupt signal is continuously active, without any further triggering edge. In such a case, the ISR has to check the AL Event Request register before the ISR is left, to guarantee that the interrupt signal is cleared.

3.14.2 Polling and Interrupt handling

Some functions of EtherCAT slave controllers can be used either by polling or by interrupting. This could be:

- SyncManager reading or writing (Mailbox, 3 buffer mode)
- AL Control changes
- Distributed Clocks Sync events
- etc.

Typically, only one of the two methods (polling or interrupting) should be chosen. Using both methods simultaneously is potentially dangerous, because interrupts could be missed when the polling access "acknowledges" the interrupt immediately when it occurs, and the interrupt service routine does not get called (because the interrupt is already gone or was never seen on the interrupt signal).

3.15 Distributed Clocks: Resolution, Precision, Accuracy

The Distributed Clocks system is operating internally with a 100 MHz clock source, i.e. 10 ns cycle time. Nevertheless, most DC values are represented by multiples of 1 ns (only the pulse length of DC SyncSignals is counted in multiples of 10 ns).

Question: How good is DC, 10 ns or 1 ns?

To answer this question, different terms have to be considered. The short answer is that the resolution of most values is 1 ns, but the precision is basically 10 ns. By averaging, the accuracy can be increased so that it can come close the 1 ns resolution. This is a more detailed answer:

Resolution
The resolution is the smallest value you can read or set in a DC register. The resolution of the DC values is 1 ns, only the pulse length of DC SyncSignals has a resolution of 10 ns.

Precision
The precision is somehow the "quality" of a single event or measurement, i.e. the deviation between actual time and the ideal time. The precision of the DCs is mainly caused by jitter, e.g. due to the 10 ns / 100 MHz operating clock of the DCs.

Accuracy
The accuracy is like the "average of the precision", i.e. the average deviation between a couple of measurements and the ideal value. The accuracy of most DC values gets better and better the more measurements are made. This is a statistical effect, with better accuracy for an increased number of measurements.

Ideal reference
For most DC functions, the precision and jitter reference is the ideal and continuous time value of the reference clock, the global system time.
The system time value read from the reference clock registers is already subject to a 10 ns jitter because of the 100 MHz operating clock for DC. So the precision of the system time read from the reference clock is 10 ns, the accuracy of the system time at the reference clock is 0 ns.

Synchronization
This system time value is spread over the network into the slaves, which requires knowledge of the propagation delays. These delays are calculated from the DC receive times. The precision of the receive times at the ports and the core depends on the physical layer. For EBUS, the precision is 10 ns, for MII it is 40 ns (because of the 25 MHz MII receive clock). By averaging the calculated delays, their accuracy can get close to the 1 ns resolution.

The system time of a synchronized slave is generated by averaging system time differences. The accuracy of a synchronized slave's system time depends on the accuracy of the delay measurement (down to 1 ns), but it can only be read with a precision of 10 ns due to the 100 MHz clock source. This is nearly the same for all synchronized DC slaves.

SyncSignal start time and LatchSignal time stamps
The SyncSignal generation and the LatchSignal time stamps depend on the local representation of the system time. The start time of a SyncSignal is subject to a jitter of 10 ns. This adds to the jitter of the local system time, so the total precision of the SyncSignal start time is about 20 ns. Averaging many SyncSignal start times will lead to accuracy near the resolution limit. The precision of LatchSignal time stamps is also about 20 ns, because of the input synchronization stage with 10 ns jitter and the local system time with another 10 ns jitter. The accuracy of multiple latch events gets down to the resolution limit.

NOTE: The precision/jitter and accuracy values are based on the algorithms; real hardware and real clock sources further increase the jitter by a small fraction, and inaccuracies can sum up.

3.16   Hardware not working

If the hardware is not working, the following functions should be tested, e.g., by external measurement equipment.

General
- Are all power lanes supplied correctly?
- Is the clock running?
- Reset: Is the reset signal released?
- Is the ESC accessing the EEPROM after power-up? No: check connection, power, clock, reset, ESC strapping options/configuration again.

Communication
- LINK_MII: Are the PHYs signaling a link? Is the link speed correct? Is the LINK_POLARITY for the ASIC/FPGA correct?
- The SII EEPROM is not required for communication. Reading registers from EtherCAT master is possible without SII EEPROM at all.
- Turn off Enhanced Link detection for testing.
- Force the LINK_MII signal to indicate link up for port 0 and try again.
- Is PDI access possible? Have a look at the registers to solve the problem. Try reading out the PHY status via MI management.

PDI
- Check if the PDI is active and that the settings are correct (PDI Control, PDI Configuration)
- Check the PDI error counter (if available)
- Try User RAM access (0x0F80:0FFF), because the Process RAM might be blocked if the EEPROM is not loaded correctly or if SyncManagers are active.
- Watch the RAM area via EtherCAT network while accessing it with the PDI.
- Try writing to registers, e.g., AL event mask. Registers have additional write checks in contrast to RAM. If only register writing is impossible, check for access errors (data sheet).
- Check if the signal state during an access (are the appropriate signals driven by the uC, does the ESC drive the appropriate signals), maybe there is an issue with floating signals.

4 EtherCAT IP Core for Altera FPGAs

4.1 Licensing issues

4.1.1 Check license status

You can check if your license is valid by invoking Altera Quartus II, menu Tools – License Setup…

The EtherCAT IP Core license has Vendor "Beckhoff (745C)", and Product "1810". Full license and OpenCore Plus license cannot be distinguished in this window, only during compilation.
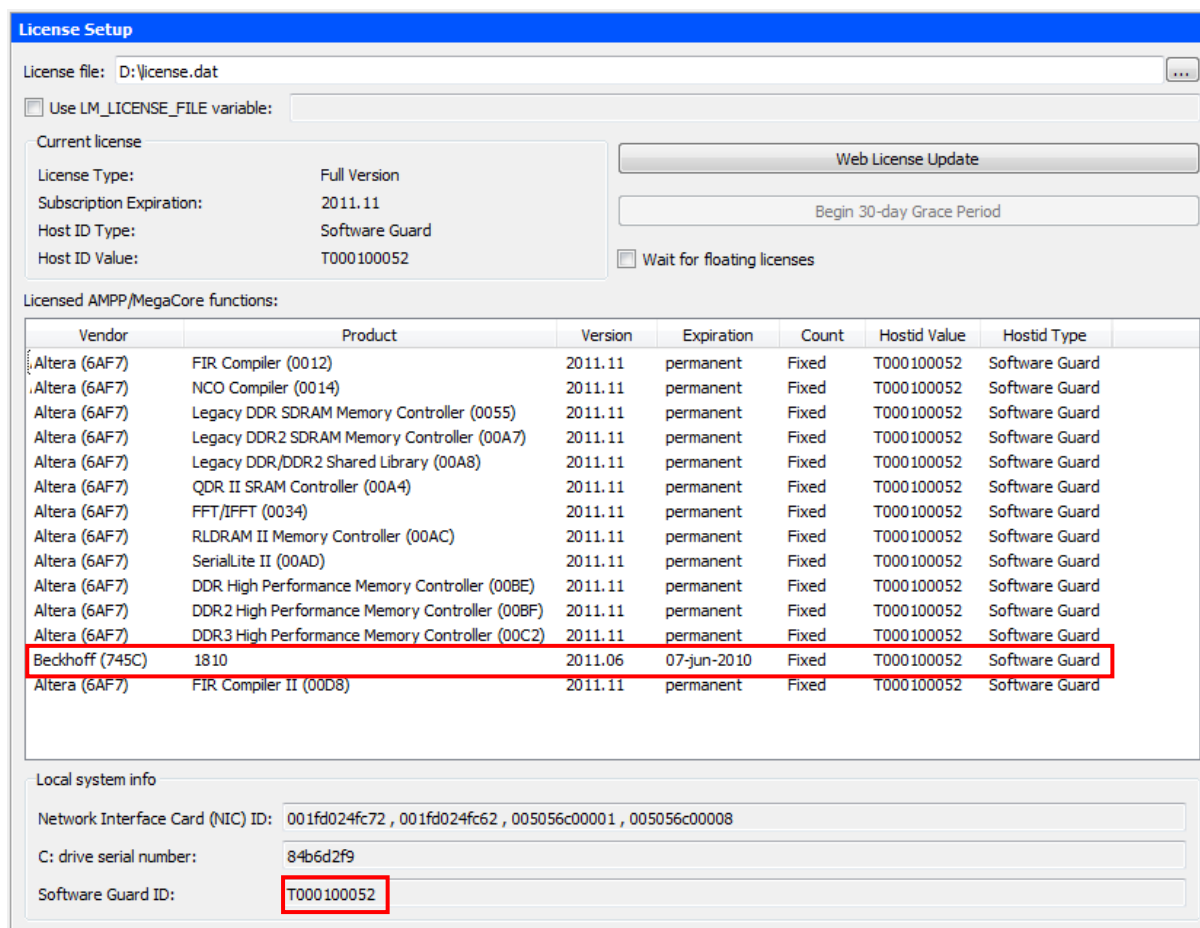
Figure 3: Example license

1. The version column states the latest release month of an EtherCAT IP Core for which the license is valid. In the example, the license is valid as long as the release month of the IP Core is June 2011 or earlier.
2. The Expiration column shows the date when the license expires. The example license is valid until 2011/06/07. This is the latest date at which you can synthesize the IP Core with this license.
3. The Count column is basically for floating licenses[1] (number of seats). If the value is "Fixed", you have a node-locked license with one seat.

---

[1] Floating licenses are not supported by the EtherCAT IP Core.

4. The Hostid Value column contains the dongle ID/MAC ID of the license.
    For node-locked licenses (Count=Fixed), this host value must occur in the lower Local system info area:
    - Hostid values beginning with "T….." are for Dongles, the corresponding Dongle ID must occur in the Software Guard ID field, otherwise the license is not valid.
    - Hostid value is a hex numer for Ethernet MAC ID, the corresponding MAC ID must occur in the Network Interfaces Card (NIC) ID field, otherwise the license is not valid.
    For floating licenses[1], the Ethernet MAC ID must belong to the MAC IDs of the license server.
5. The Hostid Type is "Software Guard" for dongles, and "NIC ID" for Ethernet MAC IDs.

4.1.2   Compare license file and Quartus II License Setup

If the Quartus II License Setup (menu Tools – License Setup…) does not report the license as expected, or if you want to find out which license is inside a license file, just open the license file with a text editor. For the EtherCAT IP Core, there should be a similar line like this:

```
FEATURE 745C_1810 alterad 2011.06 7-jun-2011 uncounted 637F8760B8D1
VENDOR_STRING="k7ke4[...]" HOSTID=GUARD_ID=T000100052 SIGN="2DE7 A4DD[...]"
```

This feature line contains a license with this information:

- It is a license for feature 745C_1810, which says it is a feature from Beckhoff (vendor ID 745C) for the EtherCAT IP Core (product ID 1810).
- It is a license for Altera, because it uses the Altera license deamon (alterad).
- The license is valid for products which are released until 06/2011.
- The license expires at June 7th, 2011
- The number of seats is uncounted (relevant for floating licenses[1])
- The following hex code and the VENDOR_STRING (shortened) are interpreted by the license software
- The HOSTID is important, it contains a MAC ID or, like this one, a Dongle ID (GUARD_ID=T000100052).
- The last field (SIGN, shortened) is interpreted by the license software.

With this information, you can compare the license file contents with the License Setup information. If the license is not displayed, compare the HOSTID of the license file against the NIC ID/Software Guard ID shown in the Quartus License Setup.

For MAC IDs, make sure the network adapter is enabled. Some adapters enter a power down mode if nothing is connected, try to connect some Ethernet device with the adapter.

From the feature line, it is not possible to determine if this is an OpenCore Plus license or a full license. This can only be determined from the filename of the original license file from Beckhoff, or during analysis in Quartus II.

4.1.3   No license found

Message

```
Error: Can't find valid feature line for core EtherCAT_VendorID (745C_1810) in current license
Error (10003): Can't open encrypted VHDL or Verilog HDL file
"C:/EDA/Altera/Quartus101/ip/beckhoff/ethercat_2.4.0/reference_designs/EL9800_DIGI_EP3C25F256/
ethercat_digitalio/ETHERCAT_VENDORID.VHD" -- current license file does not contain a valid
license for encrypted file
Info: Found 0 design units, including 0 entities, in source file
ethercat_digitalio/ethercat_vendorid.vhd
Error (10003): Can't open encrypted VHDL or Verilog HDL file
"C:/EDA/Altera/Quartus101/ip/beckhoff/ethercat_2.4.0/reference_designs/EL9800_DIGI_EP3C25F256/
ethercat_digitalio/ETHERCAT_IPCORE_TOP.VHD" -- current license file does not contain a valid
license for encrypted file
Info: Found 0 design units, including 0 entities, in source file
ethercat_digitalio/ethercat_ipcore_top.vhd
Error (10003): Can't open encrypted VHDL or Verilog HDL file
"C:/EDA/Altera/Quartus101/ip/beckhoff/ethercat_2.4.0/reference_designs/EL9800_DIGI_EP3C25F256/
ethercat_digitalio/ETHERCAT_IPCORE.VHD" -- current license file does not contain a valid
license for encrypted file
Info: Found 0 design units, including 0 entities, in source file
ethercat_digitalio/ethercat_ipcore.vhd
Info: Found 2 design units, including 1 entities, in source file el9800_digi_ep3c25f256.vhd
Error: Quartus II 64-Bit Analysis & Synthesis was unsuccessful. 4 errors, 0 warnings
Error: Quartus II Full Compilation was unsuccessful. 6 errors, 0 warnings
```
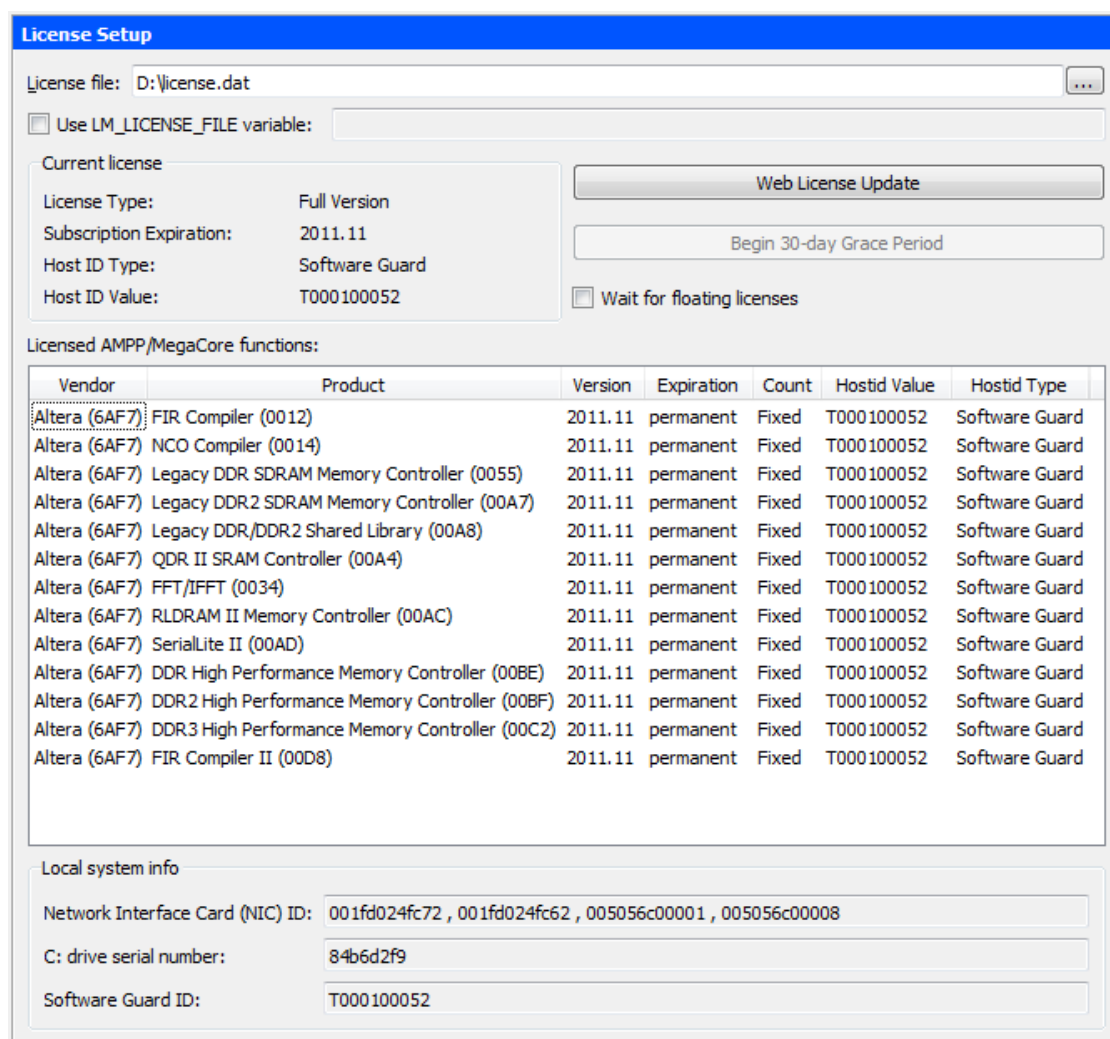

Figure 4: License setup: No license available, expecting "Beckhoff (745C) / 1810"

Solution
Install license according to EtherCAT IP Core datasheet.

4.1.4 License expired

Messages

```
Warning: License for core EtherCAT_VendorID (745C_1810), version 0000.00 is expired
Error: Can't find valid feature line for core EtherCAT_VendorID (745C_1810) in current license
Error (10003): Can't open encrypted VHDL or Verilog HDL file
"C:/Altera/Quartus101/ip/beckhoff/ethercat_2.4.0/reference_designs/EL9800_DIGI_EP3C25F256/ethe
rcat_digitalio/ETHERCAT_VENDORID.VHD" -- current license file does not contain a valid license
for encrypted file
Info: Found 0 design units, including 0 entities, in source file
ethercat_digitalio/ethercat_vendorid.vhd
Error (10003): Can't open encrypted VHDL or Verilog HDL file
"C:/Altera/Quartus101/ip/beckhoff/ethercat_2.4.0/reference_designs/EL9800_DIGI_EP3C25F256/ethe
rcat_digitalio/ETHERCAT_IPCORE_TOP.VHD" -- current license file does not contain a valid
license for encrypted file
Info: Found 0 design units, including 0 entities, in source file
ethercat_digitalio/ethercat_ipcore_top.vhd
Error (10003): Can't open encrypted VHDL or Verilog HDL file
"C:/Altera/Quartus101/ip/beckhoff/ethercat_2.4.0/reference_designs/EL9800_DIGI_EP3C25F256/ethe
rcat_digitalio/ETHERCAT_IPCORE.VHD" -- current license file does not contain a valid license
for encrypted file
Info: Found 0 design units, including 0 entities, in source file
ethercat_digitalio/ethercat_ipcore.vhd
Info: Found 2 design units, including 1 entities, in source file el9800_digi_ep3c25f256.vhd
Error: Quartus II 64-Bit Analysis & Synthesis was unsuccessful. 4 errors, 1 warning
Error: Quartus II Full Compilation was unsuccessful. 6 errors, 1 warning
```


Figure 5: License Setup: License expired

Solution
Acquire a valid license.

4.1.5 OpenCore Plus License Identification

The only way to determine if an OpenCore Plus license or a full license is used, is during compilation. For an OpenCore Plus license is used, Quartus II will produce the following additional messages:

```
Warning: OpenCore Plus Hardware Evaluation feature is turned on for the following cores
        Warning: "EtherCAT_VendorID (745C_1810)" will use the OpenCore Plus Hardware Evaluation
        feature
Warning: Messages from megafunction that supports OpenCore Plus feature
        Warning: Messages from megafunction that supports OpenCore Plus feature ETHERCAT_IPCORE
                Warning: EtherCAT Megafunction will be disabled after the time-out is reached
Warning: Megafunction that supports OpenCore Plus feature will stop functioning in 1 hour
after device is programmed
Info: Evaluation period of megafunction that supports OpenCore Plus feature can be extended
indefinitely by using tethered operation
```

For OpenCore Plus, only a time-limited SOF programming file can be produced. It is called

<project_name>_time_limited.sof

This is reported by the Assembler like this:

```
Info: Assembler is generating device programming files
Info: Design contains a time-limited core -- only a single, time-limited
programming file can be generated
Warning: Can't convert time-limited SOF into POF, HEX File, TTF, or RBF
Info: Quartus II 64-Bit Assembler was successful. 0 errors, 1 warning
```

4.2   Implementation issues

4.2.1   MegaWizard generation

Message

> Error: Error (10481): VHDL Use Clause error at ETHERCAT_IPCORE.VHD(15906): design library "work" does not contain primary unit "ECAT_VENDOR" File: C:/altera/quartus101/ip/beckhoff/ethercat_2.4.0/lib/ETHERCAT_IPCORE.VHD Line: 15906
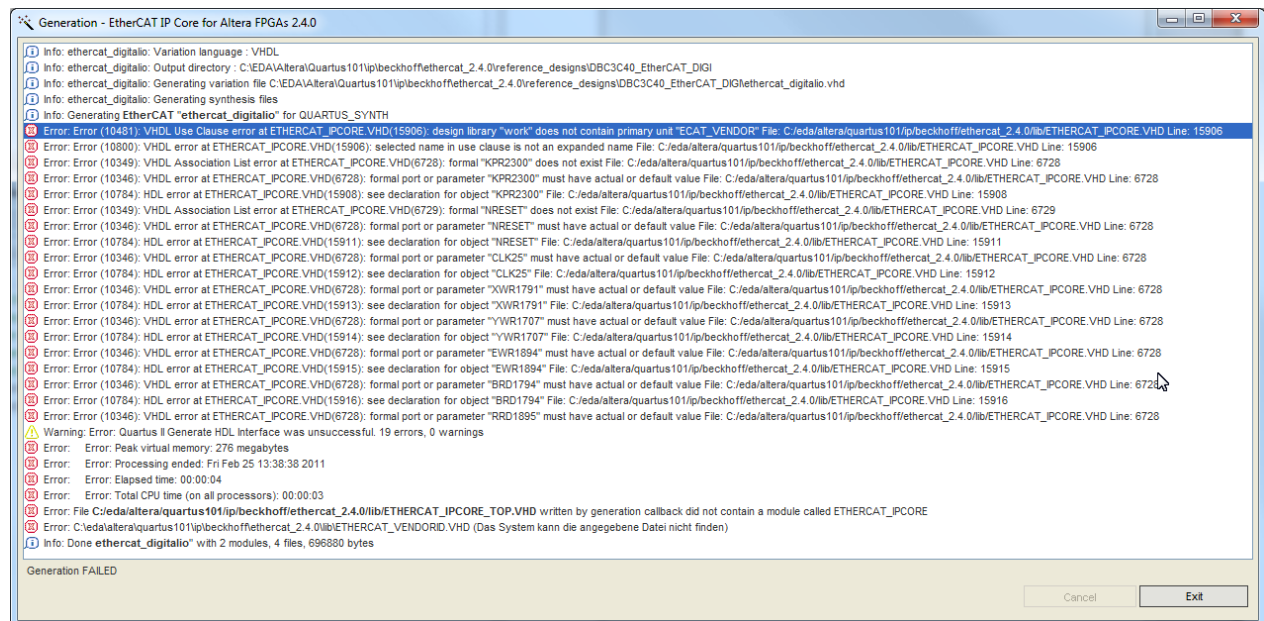
Message window

Figure 6: MegaWizard generation issue

Solution

If the vendor ID package is missing in the installation folder (e.g., because it was not installed/found during installation), it has to be added manually to the installation folder. Refer to the EtherCAT IP Core datasheet for instructions. Regenerate the EtherCAT IP Core afterwards.

4.2.2   Analysis & Synthesis

4.2.2.1   Checking the actual EtherCAT IP Core configuration

The EtherCAT IP Core configuration made in the MegaWizard can be checked during Analysis & Synthesis. The IP Core will print out it's configuration in the message window like this:

```
Info (10544): VHDL Assertion Statement at ETHERCAT_IPCORE_TOP.VHD(496):
assertion is false - report "
#####################################################" (NOTE)
#####################################################
########## ETHERCAT IP CORE V2.4.0        ##########
#####################################################
########## 2 PORTS RMII                   ##########
########## FMMU        : 2                ##########
########## SYNCMANAGER : 4                ##########
########## RAM         : 1 KBYTE          ##########
########## NO DISTRIBUTED CLOCKS          ##########
########## REGISTER SET : SMALL           ##########
########## TRISTATE BUFFERS: FUNC_DISABLED ##########
########## ESC FEATURES (0X0F81 ..):      ##########
##########   0X0000_0001_2D80_800D        ##########
########## PDI: PDI_DIGITAL_IO            ##########
##########   DIGITAL I/O BYTES: 4         ##########
##########   DIRECTION BYTE 3..0: OOII    ##########
##########   INPUT MODE 0X0150[5:4]= 0    ##########
##########   OUTPUT MODE 0X0150[7:6]= 0   ##########
########## NO GPIO                        ##########
########## PRODUCT ID: 0000:0000:0000:001C ##########
#####################################################
########## ETHERCAT IP CORE V2.4.0        ##########
#####################################################
#####################################################
```

Additionally, most of the IP Core configuration can be checked in the register space and in user RAM.

Table 2: Registers which reflect EtherCAT IP Core configuration

Register	Comment
0x0004	FMMUs supported
0x0005	SyncManagers supported
0x0006	RAM size
0x0007	Port Descriptor
0x0008:0x0009	ESC Features supported
0x0140	PDI Control
0x0141	ESC Configuration
0x0150	PDI Configuration
0x0151	SYNC/LATCH PDI Configuration
0x0152:0x0153	Extended PDI Configuration
0x0502	EEPROM Status
0x0510	MII Management Status
0x0980	Cyclic Unit Control
0x0982:0x0983	Pulse Length of SyncSignals
0x0E00:0x0E07	Product ID
0x0F80:0x0FFF	Extended ESC Features

4.2.2.2 Vendor ID package is in the project, but not on the disk

Message

Warning: Can't analyze file -- file ethercat_digitalio/ETHERCAT_VENDORID.VHD is missing

Message window

Figure 7: Vendor ID package is missing

Solution

Regenerate EtherCAT IP Core, if the vendor ID package is correctly installed, it should be copied to the project folder again.

4.2.2.3 Vendor ID package is not in the project

Message

Error (10481): VHDL Use Clause error at ETHERCAT_IPCORE.VHD(15906): design library "work" does not contain primary unit "ECAT_VENDOR"

Message window

Figure 8: Vendor ID package is missing

Solution

Regenerate EtherCAT IP Core, if the vendor ID package is correctly installed, it should be copied to the project folder again.

4.2.2.4 Important logic parts or I/O signals are optimized away, hardware does not work

If important logic blocks or I/O signals are optimized away, please check if every signal of the EtherCAT IP Core is connected, especially the input signals. Altera Quartus II adds default values for all input signals of a MegaFunction. If such an input signal remains unconnected, no error will be generated, but some logic up to I/Os will be optimized away because the input is static. This is an Altera Quartus II issue.

As an example, have a look at the generated VHDL source code of an EtherCAT MegaFunction:

```vhdl
entity ethercat_digitalio is
port (      NRESET              : in  std_logic:= '0';
            CLK25               : in  std_logic:= '0';
            CLK100              : in  std_logic:= '0';
            PROM_CLK            : out std_logic;
            PROM_SIZE           : in  std_logic:= '0';
            PROM_DATA_IN        : in  std_logic:= '0';
[...]
```

Figure 9: MegaFunction entity with default values

If *nRESET* would be left unconnected, the whole IP Core would remain in reset state. If *CLK100* would be left unconnected, only parts of the design would be functional. Depending on the signal, many different error scenarios can occur.

4.2.3   Library files are not copied to project (reference designs)

Messages

```
Warning: Can't analyze file -- file ethercat_spi/ETHERCAT_VENDORID.VHD is missing
Warning: Can't analyze file -- file ethercat_spi/ETHERCAT_IPCORE_TOP.VHD is missing
Warning: Can't analyze file -- file ethercat_spi/ETHERCAT_IPCORE.VHD is missing
[…]
Warning: Using design file ethercat_spi.vhd, which is not specified as a design file for the
current project, but contains definitions for 2 design units and 1 entities in project
Info: Elaborating entity "EtherCAT_SPI" for hierarchy "EtherCAT_SPI:ETHERCAT_INST"
Error: Node instance "ethercat_spi_inst" instantiates undefined entity "ETHERCAT_IPCORE"
Error: Quartus II 64-Bit Analysis & Synthesis was unsuccessful. 1 error, 4 warnings
Error: Quartus II Full Compilation was unsuccessful. 3 errors, 4 warnings
```

Solution

These messages typically appear if you compile a reference design for the first time without generating the EtherCAT IP Core first. The errors occur because important library files need to be copied to the project folder. Open MegaWizard Plug-In for the EtherCAT IP Core and press "Finish" (refer to the datasheet: reference design/implementation chapter). Quartus II will copy the required files to the project folder. Compile the project again, the error messages should disappear.

4.2.4 Additional signals (SIM_FAST, PHY_OFFSET) in the pinout report

Sometimes Altera Quartus II routes internal signals of the EtherCAT IP Core up to the top level design and finally to FPGA pins if OpenCore Plus evaluation is used. This issue does not occur with the full license, and it was not observed in recent Quartus II versions. It is not an EtherCAT IP Core issue, but a Altera Quartus II issue.

Typically, the names of these input signals look like this (Compilation report – Fitter – Resource Section – Input Pins) – note the extra path information, which is not common to other FPGA I/Os:

- EtherCAT_DigitalIO:inst1|ETHERCAT_IPCORE:EtherCAT_IPCore_inst|**SIM_FAST**
- EtherCAT_DigitalIO:inst1|ETHERCAT_IPCORE:EtherCAT_IPCore_inst|**PHY_OFFSET**

The pin location is assigned by the fitter, so the actual values of these signals are not predefined. The intentional default value of these internal signals is '0'. Depending on the actual pin state, the signals might become '1', which causes unwanted behavior of the IP Core. The SIM_FAST signal will e.g. disturb EEPROM and MII interface functions, if it is '1'.

This is a known bug of Altera Quartus, which is currently not fixed. The bug was observed e.g. with Quartus II 7.2 SP3, and EtherCAT IP Core V2.0.0, and the EL9800_DIGI_EP1C12F256 reference design. Other Quartus versions and other IP Core versions have also been subject to this issue.

Work-around

You have to edit the *.vhd wrapper file generated by the MegaWizard Plugin, e.g. "EtherCAT_DigitalIO.vhd" for the EL9800_DIGI_EP1C12F256 reference design, with a text editor.

In the following example, SIM_FAST and PHY_OFFSET occur as additional unwanted FPGA inputs, the actual names of the additional signals have to be taken from the Compilation report (Fitter – Resource Section – Input Pins).

1. Locate the COMPONENT declaration of the EtherCAT_IPCore (**COMPONENT**), and add the additional input signals in the **PORT** section declaration:

```
COMPONENT EtherCAT_IPCore
GENERIC (
     PRODUCT_ID0     : STD_LOGIC_VECTOR := X"1234";
     PRODUCT_ID1     : STD_LOGIC_VECTOR := X"5678";
     [...]
     USERSET2        : STD_LOGIC_VECTOR := X"0000";
     USERSET3        : STD_LOGIC_VECTOR := X"0000";
     PDI_GPIO_WIDTH  : NATURAL
     );
PORT (
     SIM_FAST        : IN STD_LOGIC;
     PHY_OFFSET      : IN STD_LOGIC;
     nRESET          : IN STD_LOGIC;
     CLK25           : IN STD_LOGIC;
     [...]
```

2. Locate the instantiation of the EtherCAT_IPCore (**PORT MAP**), and set the additional inputs to '0':

```
EtherCAT_IPCore_inst : EtherCAT_IPCore
GENERIC MAP (
     PRODUCT_ID0      => X"1234",
     PRODUCT_ID1      => X"5678",
     [...]
     USERSET2         => X"0000",
     USERSET3         => X"0000",
     PDI_GPIO_WIDTH   => 0
     )
PORT MAP (
     SIM_FAST         => '0',
     PHY_OFFSET       => '0',
     nRESET           =>  nRESET,
     CLK25            =>  CLK25,
     [...]
```

3. Save the changes and start Compilation in Quartus. Check the Fitter reports: no additional signals with path information and locations assigned by the fitter should be left.

You have to repeat these steps each time you generate the EtherCAT IP Core again, because this will overwrite the changes.

4.2.5   Timing closure issues

4.2.5.1   OpenCore Plus logic does not achieve timing (altera_reserved_tck)

Sometimes timing requirements are not met with OpenCore Plus. Experience shows that timing violations related to the clock altera_reserved_tck can be ignored. They do not occur with the full version.
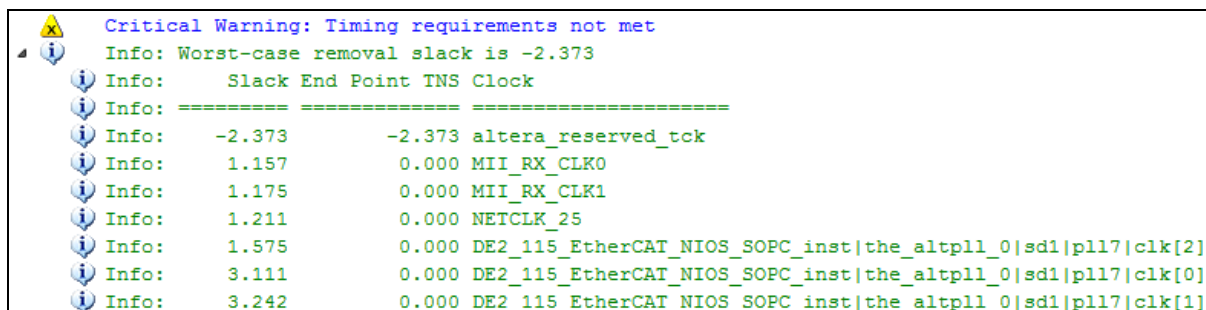
```
  ⚠️  Critical Warning: Timing requirements not met
 ▲ ⓘ    Info: Worst-case removal slack is -2.373
     ⓘ Info:    Slack End Point TNS Clock
     ⓘ Info: ========= ============== ======================
     ⓘ Info:    -2.373         -2.373 altera_reserved_tck
     ⓘ Info:     1.157          0.000 MII_RX_CLK0
     ⓘ Info:     1.175          0.000 MII_RX_CLK1
     ⓘ Info:     1.211          0.000 NETCLK_25
     ⓘ Info:     1.575          0.000 DE2_115_EtherCAT_NIOS_SOPC_inst|the_altpll_0|sd1|pll7|clk[2]
     ⓘ Info:     3.111          0.000 DE2_115_EtherCAT_NIOS_SOPC_inst|the_altpll_0|sd1|pll7|clk[0]
     ⓘ Info:     3.242          0.000 DE2_115_EtherCAT_NIOS_SOPC_inst|the_altpll_0|sd1|pll7|clk[1]
```

Figure 10: Timing requirement of OpenCore Plus logic not met

4.2.5.2   General timing closure issues

Regarding constraints, we can only give general advice on increasing the synthesis effort. The default synthesis and implementation features are a good starting point, the following additional options might increase timing closure:

Analysis & Synthesis Settings
Optimization Technique = Area
Timing-Driven Synthesis = On
Advanced:
        Auto Resource Sharing = Off
        Remove Duplicate Registers = Off

Compilation Process Settings – Physical Synthesis Optimizations – Optimize for performance (physical synthesis)
Perform physical synthesis for combinational logic = On
Perform register retiming = On
Effort level = Normal (or even Extra)

Fitter Settings
Fitter effort = Standard

General advice
Over-constraining was not found useful with current synthesis, it might cause the tools to put too much effort in paths which would originally pass, and consequently making other paths slower.

Find out if the violating path consists basically of logic or routing delays. If the logic delays are too high, reduce the number of features, increase optimization effort, or use a faster speed grade.

Generally, the more features are used, the tighter becomes the timing. Especially 64 bit Distributed Clocks are very demanding, due to the high register count. The low-cost FPGAs are often not able to use 64 bit DC at all. In many cases using 32 bit DC solves the problem.

If many FPGA resources are available, the design is sometimes spread widely across the FPGA, resulting in long connections with high routing delay. Reduce the available FPGA resources/area for the EtherCAT IP Core to improve timing.

Generally, it is impossible to predict timing closure for custom user logic in combination with all IP Core configurations and FPGA devices. Final certainty can only be reached by implementing the design – that is what the evaluation versions are for. There will be combinations which are not able to reach timing closure at all.

4.3   Hardware issues

If the hardware is not working, the following functions should be tested, e.g., by routing signals to LEDs or using SignalTap II or external measurement equipment.

- Reset: polarity issues are common, probe exactly the signal which enters the IP Core.
- Clock: Are all clocks connected, valid and synchronous (25 MHz and 100 MHz)? Is the PLL locked?
- Turn off MI link detection and configuration for testing if the PHY does not need to be re-configured.
- Check in the reports if all FPGA pins are located correctly.
- Is the Timing report error-free?

4.4   OpenCore Plus design stops operating too early

If you want to test the OpenCore Plus design for about one hour without JTAG connection, follow these steps:

1. Program the FPGA (the OpenCore Plus window appears after programming)
2. Disconnect the JTAG interface.
3. Accept the OpenCore Plus window notice.

If you accept the OpenCore Plus window notice before disconnecting the JTAG, Quartus II will disable the logic immediately. By following the above steps, the one hour operation is possible.

5   EtherCAT IP Core for Xilinx FPGAs

5.1   Project navigator/EDK crashes

Xilinx ISE/EDK are subject to crash without any notice when a project with EtherCAT IP Core is opened or migrated to a newer version, while the folder containing the decryption keys (*%APPDATA%\RSA or $HOME/.rsa*) is not accessible/present and/or the decryption keys themselves are missing.

Solution

Install the RSA decryption keys according to the data sheet. Please note that there are three different keys:

- *rsa_ethercat_base_pvt.pem*
- *rsa_ethercat_ip_<version>_eval_pvt.pem*   (Evaluation license, version specific)
- *rsa_ethercat_ip_<version>_full_pvt.pem*   (Full license, version specific)

5.2   Licensing issues

5.2.1   Check license status

You can check if your license is valid by invoking Xilinx ISE, menu Help – Manage License…
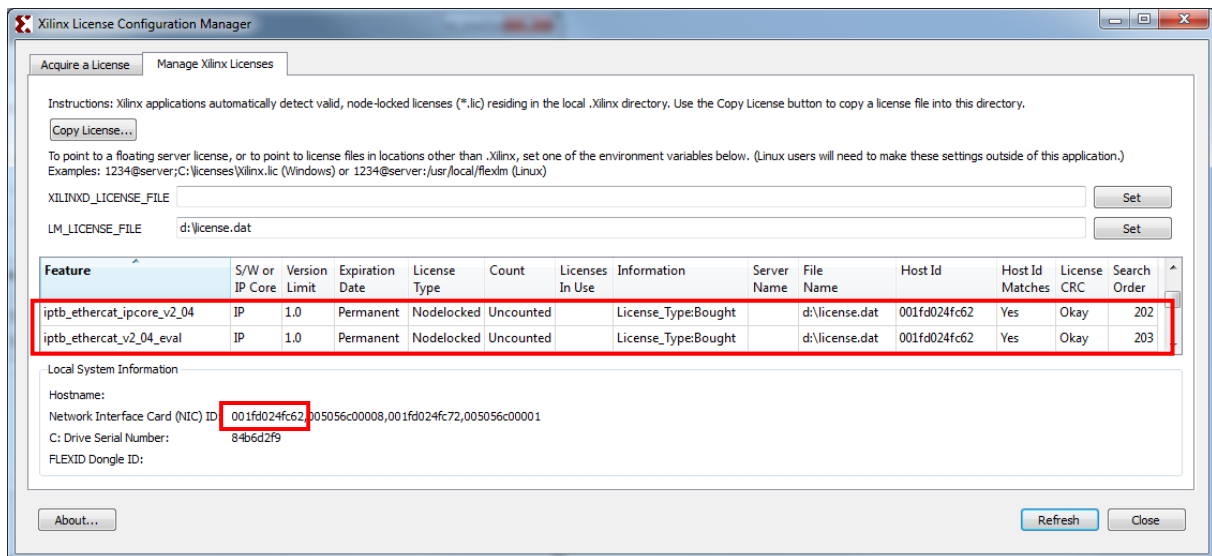

Figure 11: Example license

1. The EtherCAT IP Core license has one of the following features:
    *   iptb_ethercat_ipcore_v2_04   for the full version (IP Core Version 2.04a-z)
    *   iptb_ethercat_v2_04_eval   for the evaluation version (IP Core Version 2.04a-z)
2. The "S/W or IP Core" column gives information if the license is for a tool or for an IP Core.
3. The "Version Limit" column is not used for the EtherCAT IP Core, it is always 1.0.
4. The "Expiration date" column shows the date when the license expires. The example license is permanently valid, otherwise this is the latest date at which you can synthesize the IP Core with this license.
5. "License Type" is either Nodelocked (for a single license), or Floating for server licenses.
6. The "Count" column is basically for floating licenses[2] (number of seats). For nodelocked licenses, the count is always "Uncounted".
7. "Licenses in Use" is for floating licenses[2].
8. The "Information" column shows information from the license feature.
9. The server name used by floating licenses[2] is shown in the "Server Name" column.
10. The filename with the license feature is shown in the File Name column.
11. The "Host Id" column contains MAC ID of the license.
    For node-locked licenses, this host value must occur in the lower Local System Information area in the Network Interfaces Card (NIC) ID field, otherwise the license is not valid.
    For floating licenses[2], the Ethernet MAC ID must belong to the MAC IDs of the license server.
12. The column "Host Id Matches" says "Yes" if the MAC ID was found, otherwise the value is "No".
13. The "License CRC" column shows the result of the license check
14. The "Search Order" enumerates all the licenses.

---

[2] Floating licenses are not supported by the EtherCAT IP Core.

5.2.2 Compare license file and Xilinx License Configuration Manager

If the License Configuration Manager does not report the license as expected, or if you want to find out which license is inside a license file, just open the license file with a text editor. For the EtherCAT IP Core full edition, there should be a similar line like this:

```
FEATURE iptb_ethercat_ipcore_v2_04 xilinxd 1.0 7-jun-2011 uncounted
1B68DE7FAB61 VENDOR_STRING=License_Type:Bought HOSTID=001fd024fc62
ISSUER=Xilinx TS_OK
```

The EtherCAT IP Core evaluation edition has such a license entry:

```
FEATURE iptb_ethercat_v2_04_eval xilinxd 1.0 7-jun-2011 uncounted
1B68DE7FAB61 VENDOR_STRING=License_Type:Bought HOSTID=001fd024fc62
ISSUER=Xilinx TS_OK
```

These feature lines each contain a license with this information:

- It is a license for feature *iptb_ethercat_ipcore_v2_04*, i.e., for the EtherCAT IP Core with version 2.04a-z (full license), or
  it is a license for feature *iptb_ethercat_v2_04_eval*, i.e., for the EtherCAT IP Core with version 2.04a-z (evaluation license).
- It is a license for Xilinx, because it uses the Xilinx license deamon (xilinxd).
- The version number 1.0 is fixed, regardless of the EtherCAT IP Core version.
- The license expires at June 7th, 2011
- The number of seats is uncounted (relevant for floating licenses[2])
- The following hex code and the VENDOR_STRING (shortened) are interpreted by the license software
- The HOSTID is important, it contains the MAC ID (001fd024fc62).
- The license ISSUER is Xilinx (via Beckhoff)
- The TS_OK states that remote sessions are allowed.

With this information, you can compare the license file contents with the License Configuration Manager information. If the license is not displayed, compare the HOSTID of the license file against the NIC ID shown in the Xilinx License Configuration Manager.

Make sure the network adapter is enabled. Some adapters enter a power down mode if nothing is connected, try to connect something with the adapter.

5.2.3 No license found

This error occurs if either no valid license was found, or if the type of the license file (full, evaluation) is not the same as the EtherCAT IP Core. The error message shows which license is required by the IP Core (highlighted).

Messages

```
Analyzing hierarchy for entity <EtherCAT_DigitalIO> in library <work> (architecture <SYN>).
INFO:coreutil - No license for component <IPTB_ETHERCAT_IPCORE_V2_04> found. You may use the
customization GUI for this component but you will not be able to generate any implementation
or simulation files.

   For license installation help, please visit:
   www.xilinx.com/ipcenter/ip_license/ip_licensing_help.htm

   For ordering information, please refer to the product page for this component on:
www.xilinx.com FLEXlm Error: No such feature exists. (-5,21)
ERROR:Xst:1484 - A core is unlicensed !
-->
Total memory usage is 348576 kilobytes

Number of errors   :    1 (   0 filtered)
Number of warnings :    0 (   0 filtered)
Number of infos    :    1 (   0 filtered)

Process "Synthesize - XST" failed
```
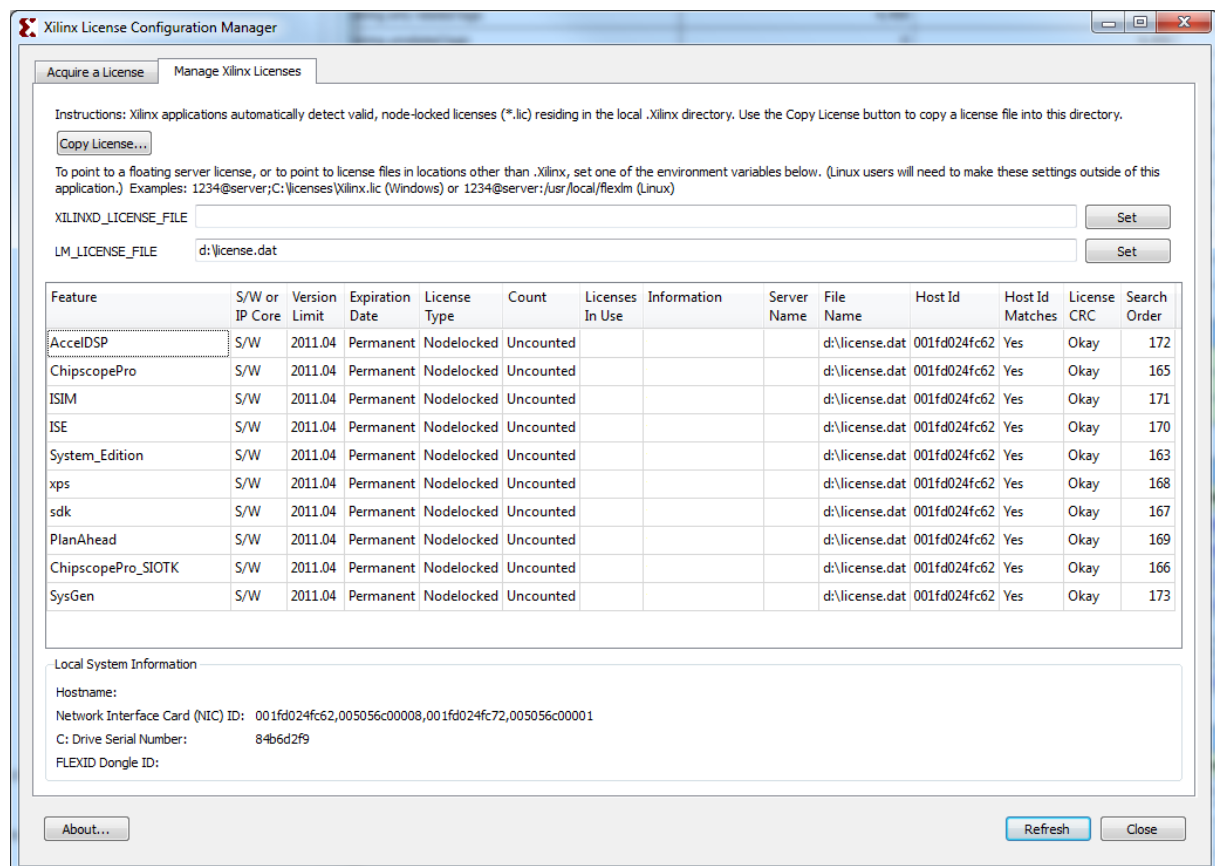

Figure 12: License Configuration Manager: No license available (expecting *iptb_ethercat_...*)

Solution

Install the required license according to the EtherCAT IP Core datasheet or re-install EtherCAT IP Core if the wrong license type is used (full/evaluation).

5.2.4   Evaluation License Identification

5.2.4.1   Installed license file

If you want to find out if you have the full license or the evaluation license installed, run Xilinx ISE, menu Help – Manage License…

Check the feature line for the EtherCAT IP Core:

- iptb_ethercat_ipcore_v2_04   for the full version (IP Core Version 2.04a-z)
- iptb_ethercat_v2_04_eval   for the evaluation version (IP Core Version 2.04a-z)

5.2.4.2   Installed EtherCAT IP Core

There are two ways to find out which type of EtherCAT IP Core is installed (full/evaluation).

- Remove all EtherCAT IP Core licenses and synthesize a reference design. The error message gives the type of the license:
  "No license for component <IPTB_ETHERCAT_IPCORE_V2_04> found" for the full license, and
  "No license for component <IPTB_ETHERCAT_V2_04_EVAL> found" for the evaluation license
- Browse the installation folder <*IPInst_dir*>\lib. There should be a text file indicating the type of the license:
  The full version of EtherCAT_IPCore.vhd was installed.txt or
  The evaluation version of EtherCAT_IPCore.vhd was installed.txt
- If the EtherCAT IP Core is synthesizable, watch for the following XST messages in the console (the first one is the full version, the last one is the evaluation version):

```
INFO:Xst:1749 - "C:/BECKHOFF/ethercat-v2.04a/lib/EtherCAT_IPCore.vhd" line
26524: note:
####################################################
####################################################
########## ETHERCAT IP CORE V2.04A          ##########
####################################################
########## 2 PORTS MII                      ##########
[…]
```

or

```
INFO:Xst:1749 - "C:/BECKHOFF/ethercat-v2.04a/lib/EtherCAT_IPCore.vhd" line
26524: note:
####################################################
####################################################
########## ETHERCAT IP CORE V2.04A          ##########
####################################################
########## EVALUATION VERSION               ##########
########## 2 PORTS MII                      ##########
[…]
```

5.2.5   Vendor ID package missing

If the vendor ID package is missing, the library unit ECAT_VENDOR cannot be found.

Messages

```
[…]
Entity <MDI1630> compiled.
Entity <MDI1630> (Architecture <XRT2219>) compiled.
ERROR:HDLParsers:3014 - "C:/BECKHOFF/ethercat-
v2.04a/lib/EtherCAT_IPCore.vhd" Line 13059. Library unit ECAT_VENDOR is not
available in library work.
-->


Total memory usage is 296892 kilobytes

Number of errors   :    1 (   0 filtered)
Number of warnings :    0 (   0 filtered)
Number of infos    :    0 (   0 filtered)
```

Solution
Add the vendor ID package to the project. This happens if the vendor ID package was not added to
the IP Core installation during setup. Refer to the EtherCAT IP Core datasheet on adding the IP Core
vendor ID package and integrating it into the reference designs. Also check if the environment variable
for the IPCore_Config tool is set (datasheet). Regenerate EtherCAT IP Cores for the EDK with the
IPCore_Config tool to integrate the vendor ID package afterwards.

5.3 Implementation issues

5.3.1 XST

5.3.1.1 RSA decryption keys missing (ISE)

The ISE/EDK design software might crash upon opening a project file or migrating a project file to a newer version if the RSA decryption keys are not properly installed. Another expression of missing RSA decryption keys is the following message:

Messages (1)

```
[…]
========================================================================
*                          HDL Compilation                             *
========================================================================
input in flex scanner failed

Process "Synthesize - XST" failed
```

Messages (2)

```
INFO:HDLCompiler:1061 - Parsing VHDL file
   "/opt/beckhoff/ethercat_v2.04a/lib/EtherCAT_CLK.vhd" into library work
input in flex scanner failed
```

Solution

Install the RSA decryption keys according to the EtherCAT IP Core data sheet (at least two). Please note that there are three different keys:

- *rsa_ethercat_base_pvt.pem* (Base, version independent)
- *rsa_ethercat_ip_<version>_eval_pvt.pem* (Evaluation license, version specific)
- *rsa_ethercat_ip_<version>_full_pvt.pem* (Full license, version specific)

5.3.1.2 RSA decryption keys missing (EDK)

Final Message (error message for many kinds of issues)

```
ERROR:EDK:440 - platgen failed with errors!
make: *** [implementation/system.bmm] Error 2
Done!
```

Detail Message above final message (hinting at missing RSA keys)

```
ERROR:EDK:546 - Aborting XST flow execution!
INFO:EDK:2246 - Refer to
   C:\<path>\synthesis\plb_ethercat_user_0_wrapper_xst.srp for details

Running NGCBUILD ...
IPNAME:microblaze_0_wrapper INSTANCE:microblaze_0 -
C:\<path>\system.mhs
line 47 - Running NGCBUILD
input in flex scanner failed
```

The .srp file ends with the following entry, with no more messages or explanations regarding the error:

```
==========================================================================
*                         HDL Compilation                                *
==========================================================================

[end of file]
```

Solution

Install the RSA decryption keys according to the EtherCAT IP Core data sheet (at least two). Please note that there are three different keys:

- *rsa_ethercat_base_pvt.pem*   (Base, version independent)
- *rsa_ethercat_ip_<version>_eval_pvt.pem*   (Evaluation license, version specific)
- *rsa_ethercat_ip_<version>_full_pvt.pem*   (Full license, version specific)

5.3.1.3   Checking the actual EtherCAT IP Core configuration

The hardware configuration is printed out during synthesis like this:

```
INFO:Xst:1749 - "C:/BECKHOFF/ethercat-v2.04a/lib/EtherCAT_IPCore.vhd" line
26524: note:
####################################################
####################################################
########## ETHERCAT IP CORE V2.04A        ##########
####################################################
########## EVALUATION VERSION             ##########
########## 2 PORTS MII                    ##########
########## FMMU        : 2                ##########
########## SYNCMANAGER : 3                ##########
########## RAM         : 1 KBYTE          ##########
########## NO DISTRIBUTED CLOCKS          ##########
########## REGISTER SET : SMALL           ##########
########## TRISTATE BUFFERS: ALL_ENABLED  ##########
########## ESC FEATURES (0X0F81 ..):      ##########
##########   0X0000_0081_2D90_800D        ##########
########## PDI: PDI_DIGITAL_IO            ##########
##########   DIGITAL I/O BYTES: 4         ##########
##########   DIRECTION BYTE 3..0: OOII    ##########
##########   INPUT MODE 0X0150[5:4]= 0    ##########
##########   OUTPUT MODE 0X0150[7:6]= 0   ##########
########## NO GPIO                        ##########
########## PRODUCT ID: 0000:0000:0000:0019  ##########
####################################################
########## ETHERCAT IP CORE V2.04A        ##########
####################################################
####################################################
```

Additionally, most of the IP Core configuration can be checked in the register space and in user RAM.

Table 3: Registers which reflect EtherCAT IP Core configuration

Register	Comment
0x0004	FMMUs supported
0x0005	SyncManagers supported
0x0006	RAM size
0x0007	Port Descriptor
0x0008:0x0009	ESC Features supported
0x0140	PDI Control
0x0141	ESC Configuration
0x0150	PDI Configuration
0x0151	SYNC/LATCH PDI Configuration
0x0152:0x0153	Extended PDI Configuration
0x0502	EEPROM Status
0x0510	MII Management Status
0x0980	Cyclic Unit Control
0x0982:0x0983	Pulse Length of SyncSignals
0x0E00:0x0E07	Product ID
0x0F80:0x0FFF	Extended ESC Features

5.3.2 Place & Route

5.3.2.1 CLOCK_DEDICATED_ROUTE=FALSE with SPI reference design

The SPI_DI and SPI_SEL signals are used as clock inputs for a few control logic registers. Since these signals are not placed on dedicated clock inputs – which is not required –, Xilinx ISE sometimes issues an error, which needs to be suppressed.

Messages

```
Phase 4.2  Initial Clock and IO Placement

................
ERROR:Place:1018 - A clock IOB / clock component pair have been found that are not placed at an optimal clock IOB /
    clock site pair. The clock component <PDI_SPI_SEL_BUFGP/BUFG> is placed at site <BUFGMUX_X2Y0>. The IO component
    <PDI_SPI_SEL> is placed at site <PAD167>.  This will not allow the use of the fast path between the IO and the Clock
    buffer. If this sub optimal condition is acceptable for this design, you may use the CLOCK_DEDICATED_ROUTE constraint
    in the .ucf file to demote this message to a WARNING and allow your design to continue. However, the use of this
    override is highly discouraged as it may lead to very poor timing results. It is recommended that this error
    condition be corrected in the design. A list of all the COMP.PINs used in this clock placement rule is listed below.
    These examples can be used directly in the .ucf file to override this clock rule.
    < NET "PDI_SPI_SEL" CLOCK_DEDICATED_ROUTE = FALSE; >

ERROR:Place:1018 - A clock IOB / clock component pair have been found that are not placed at an optimal clock IOB /
    clock site pair. The clock component <PDI_SPI_DI_BUFGP/BUFG> is placed at site <BUFGMUX_X1Y1>. The IO component
    <PDI_SPI_DI> is placed at site <PAD199>.  This will not allow the use of the fast path between the IO and the Clock
    buffer. If this sub optimal condition is acceptable for this design, you may use the CLOCK_DEDICATED_ROUTE constraint
    in the .ucf file to demote this message to a WARNING and allow your design to continue. However, the use of this
    override is highly discouraged as it may lead to very poor timing results. It is recommended that this error
    condition be corrected in the design. A list of all the COMP.PINs used in this clock placement rule is listed below.
    These examples can be used directly in the .ucf file to override this clock rule.
    < NET "PDI_SPI_DI" CLOCK_DEDICATED_ROUTE = FALSE; >
```

Solution for ISE/EDK 10/11/12 with Spartan-3x devices
Add the following constraints to the UCF file:

```
Net PDI_SPI_SEL CLOCK_DEDICATED_ROUTE = FALSE;
Net PDI_SPI_DI CLOCK_DEDICATED_ROUTE = FALSE;
```

Solution for ISE/EDK 11.3 with Spartan-6 devices
Add the following constraints to the UCF file:

```
PIN "PDI_SPI_DI_BUFGP.O" CLOCK_DEDICATED_ROUTE = FALSE;
PIN "PDI_SPI_SEL_BUFGP.O" CLOCK_DEDICATED_ROUTE = FALSE;
```

Solution for ISE/EDK 11.4-12.4 with Spartan-6 devices
Add the following constraints to the UCF file:

```
PIN "PDI_SPI_DI_BUFGP/BUFG.O" CLOCK_DEDICATED_ROUTE = FALSE;
PIN "PDI_SPI_SEL_BUFGP/BUFG.O" CLOCK_DEDICATED_ROUTE = FALSE;
```

Solution for ISE/EDK 13.1
In this version, no CLOCK_DEDICATED_ROUTE=FALSE constraint is required, since the SPI_CLK location is not considered an error any more.

5.3.3 PlanAhead

5.3.3.1 PlanAhead implementation/floorplaning/analysis is not possible

PlanAhead can only be used pre-synthesis with the EtherCAT IP Core (e.g. for pin planning), because PlanAhead does not support the security attributes in the generated netlist like ISE. Xilinx is aware of this issue.

Messages

```
WARN: [HD-EDIFIN 5] Could not read top design file 'C:\BECKHOFF\ethercat-
v2.04a\reference_designs\EL9800_DIGI_XC3S1200E\EL9800_DIGI_XC3S1200E_VHDL.n
gc' because ngc2edif command failed with the following message:
ERROR:NetListWriters:415 - The design contains secured core(s). Creation of
the
    output netlist is prohibited.

ERROR: Failed to open design - please see the console for details
```

5.3.4   General timing closure issues

Regarding constraints, we can only give general advice on increasing the synthesis effort. The default synthesis and implementation features are a good starting point, the following additional options might increase timing closure:

XST – Synthesis Options
Optimization Goal = Speed
Optimization Effort = High

XST – Xilinx Specific Options
Register Duplication = On
Equivalent Register Removal = Off

Map Properties
Perform Timing-Driven Packing and Placement = On
Map Effort = Standard (or even High or High + Extra Effort)
Combinatorial Logic Optimization = On
Register Duplication = On
Allow Logic Optimization Across Hierarchy = On
Optimization Strategy (Cover Mode) = Speed

Place & Route Properties
Place & Route Effort Level (Overall) = High (or even Extra Effort = Normal)

General advice
Over-constraining was not found useful with current synthesis, it might cause the tools to put too much effort in paths which would originally pass, and consequently making other paths slower.

Find out if the violating path consists basically of logic or routing delays. If the logic delays are too high, reduce the number of features, increase optimization effort, or use a faster speed grade.

Generally, the more features are used, the tighter becomes the timing. Especially 64 bit Distributed Clocks are very demanding, due to the high register count. The low-cost FPGAs are often not able to use 64 bit DC at all. In many cases using 32 bit DC solves the problem.

If many FPGA resources are available, the design is sometimes spread widely across the FPGA, resulting in long connections with high routing delay. Reduce the available FPGA resources/area for the EtherCAT IP Core to improve timing.

Generally, it is impossible to predict timing closure for custom user logic in combination with all IP Core configurations and FPGA devices. Final certainty can only be reached by implementing the design – that is what the evaluation versions are for. There will be combinations which are not able to reach timing closure at all.

5.4   Hardware issues

If the hardware is not working, the following functions should be tested, e.g., by routing signals to LEDs or using ChipScope or external measurement equipment.

- Reset: polarity issues are common, probe exactly the signal which enters the IP Core.
- Clock: Are all clocks connected, valid and synchronous (25 MHz and 100 MHz)? Is the PLL locked?
- Turn off MI link detection and configuration for testing if the PHY does not need to be programmed.
- Check if all FPGA pins are located correctly.
- Is the Timing report error-free?

6 Appendix

6.1 Logging Error Counters in TwinCAT

By default, TwinCAT reads out the ESC error counter registers 0x0300/0x0302/0x0304/0x0306 (CRC error port 0-3) and clears them. The results are accumulated in software and displayed in the CRC column for each port of the ESC (separated by commas).
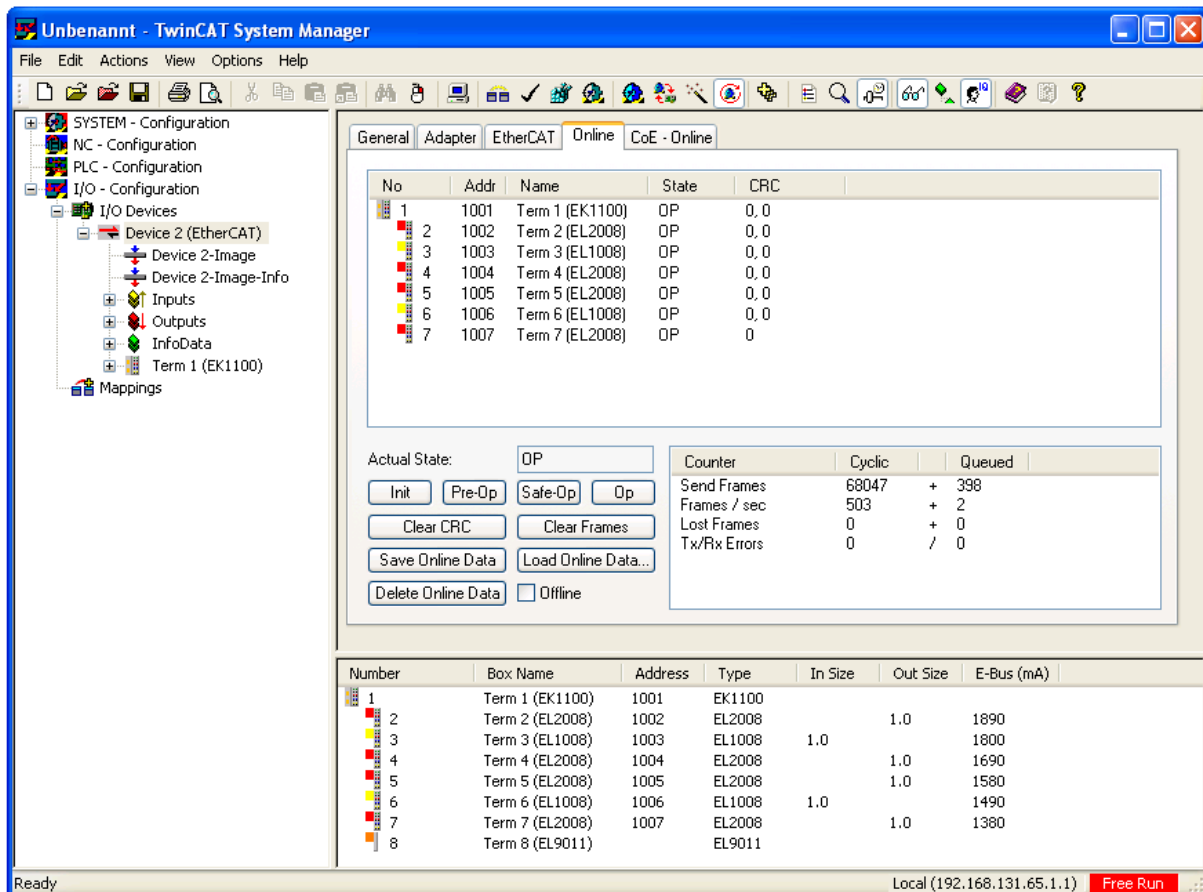
Figure 13: TwinCAT logging CRC error counters

By clearing the CRC error counters, TwinCAT also clears the RX error counter registers 0x0301/0x0303/0x0305/0x0307. The other error counters are also not evaluated by TwinCAT. For a better identification of the error source, it is necessary to disable the CRC error logging and have a look at all the error counters. Follow these steps to disable CRC error logging and displaying the error counters.

1. Go to the "EtherCAT" tab and open "Advanced Settings…":
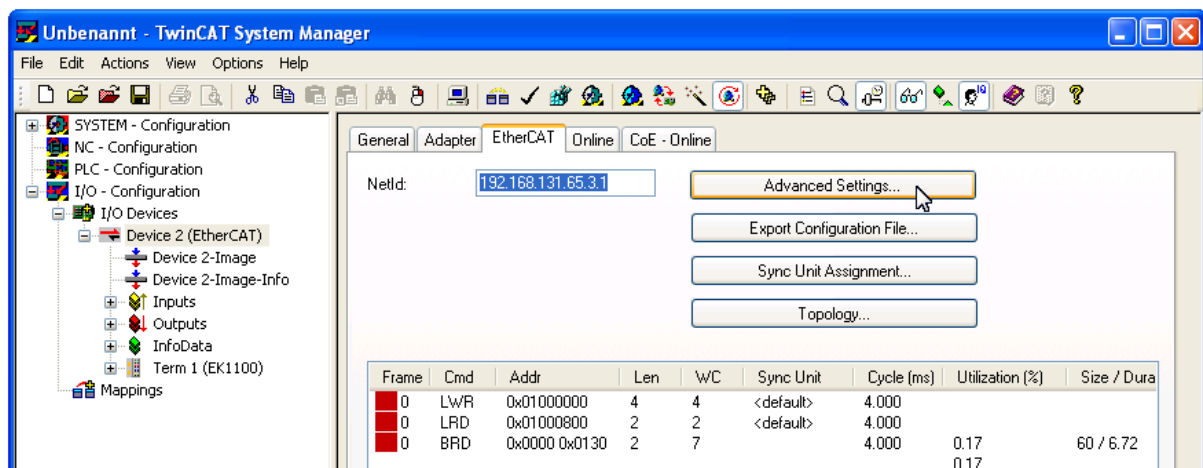
Figure 14: Go to EtherCAT Advanced Settings

2. Unselect "Log CRC Counters" and press OK.
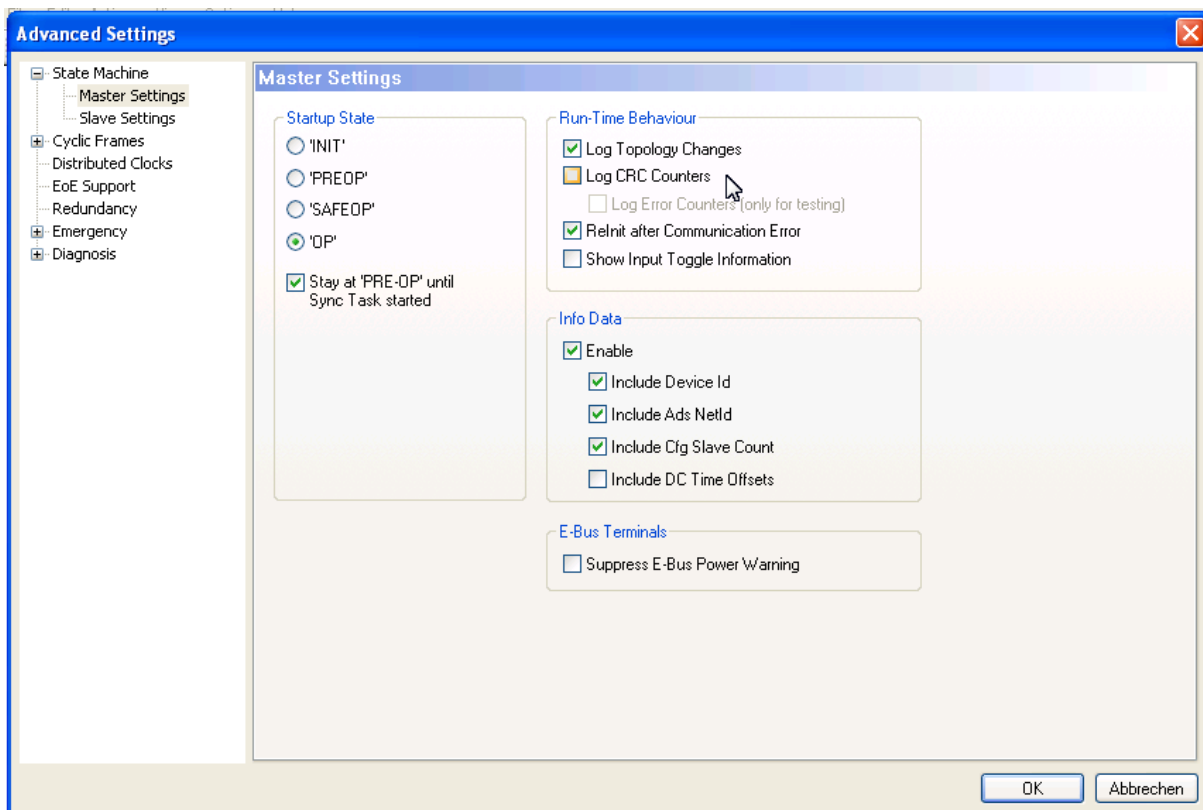
Figure 15: Unselect CRC Counter logging

3. Although the CRC column is not visible anymore, TwinCAT still reads out and clears the CRC error counter registers. Activate the new settings by reloading I/O devices (press F4 or use the button).

It is very important to reload I/O devices (F4) after disabling "Log CRC Counters", otherwise the error counters will still be cleared automatically by TwinCAT.
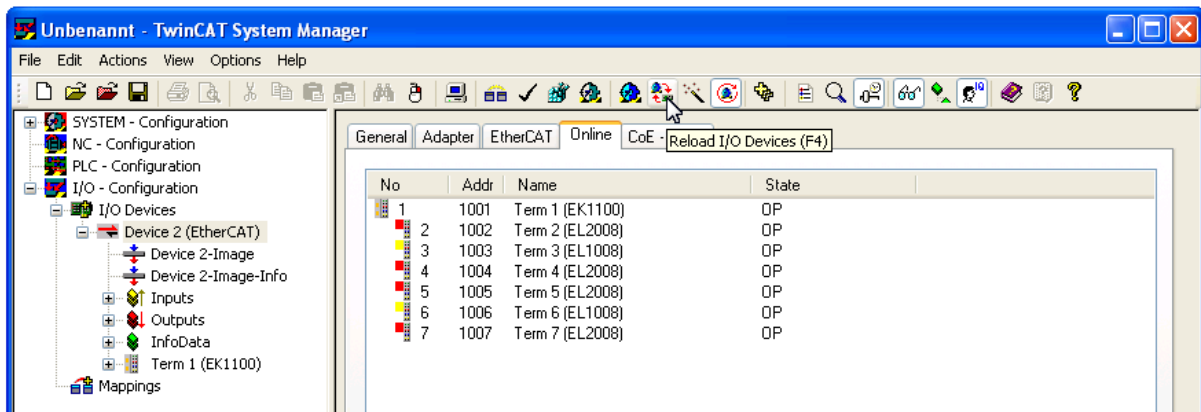
Figure 16: Reload I/O Devices (F4) after disabling CRC logging

4. Now you can add the error counters to the online view. Right-click into the Online view area and select "Properties…"
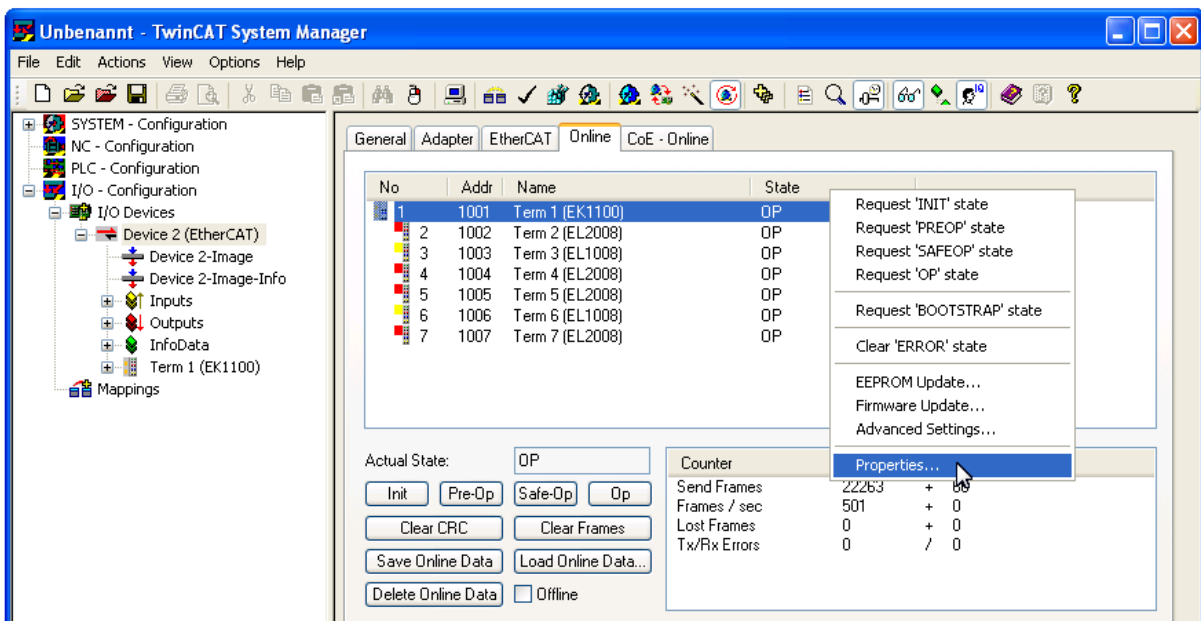
Figure 17: Online view properties

5. Select the following registers for online viewing, and press OK:
   2 port ESCs:
   - "0300 CRC A",
   - "0302 CRC B",
   - "0308 Forw. CRC A/B",
   - "030C Proc. CRC/PDI Err", and
   - "0310 Link Lost A/B"

   3-4 port port ESCs: additionally select
   - "0304 CRC C",
   - "0306 CRC D",
   - "030A Forw. CRC C/D", and
   - "0312 Link Lost C/D"

   Optionally Enable
   - "0000 ESC Rev./Type" and
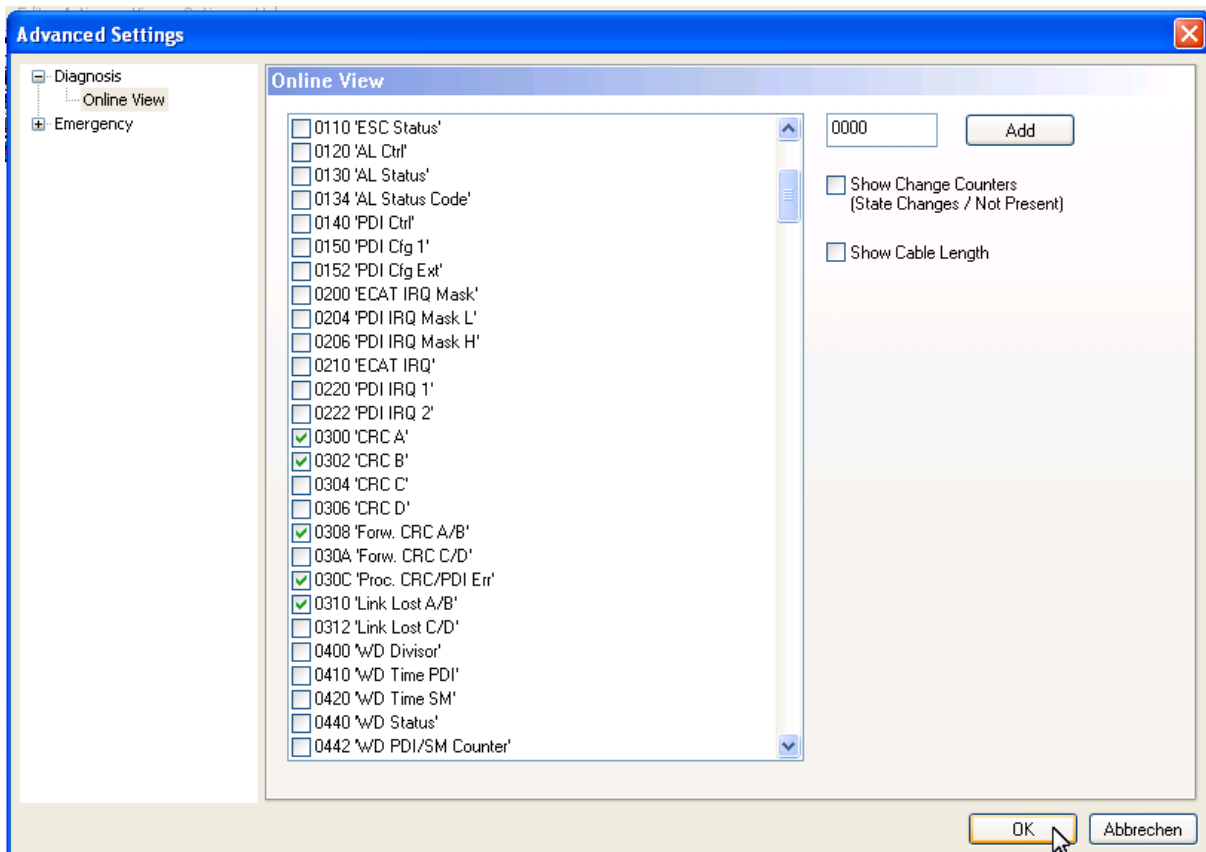   - "0002 ESC Build"
   to identify the type of ESC.

Figure 18: Select registers for online view

NOTE: TwinCAT uses A/B/C/D to enumerate the ESC ports 0/1/2/3.

6. Now you might see some errors in registers 0x030C-0x0313 resulting from power-up. These errors have no significance, only errors increasing during operation are important.
TwinCAT reads out and clears only the error counters in register 0x0300-0x0307, and pressing the button "Clear CRC" also clears only these counters. The other counters are still accumulating from power-up until now.
Please note that the error counters are shown as 16 bit values, i.e., the column called "Reg. 0300" shows in fact registers 0x0300 (lower 8 bit, right side) and 0x0301 (higher 8 bit, left side).

Press the buttons "Clear CRC" and "Clear Frames" to clear the TwinCAT counters in the lower right area (Lost frames, Tx/Rx Errors).
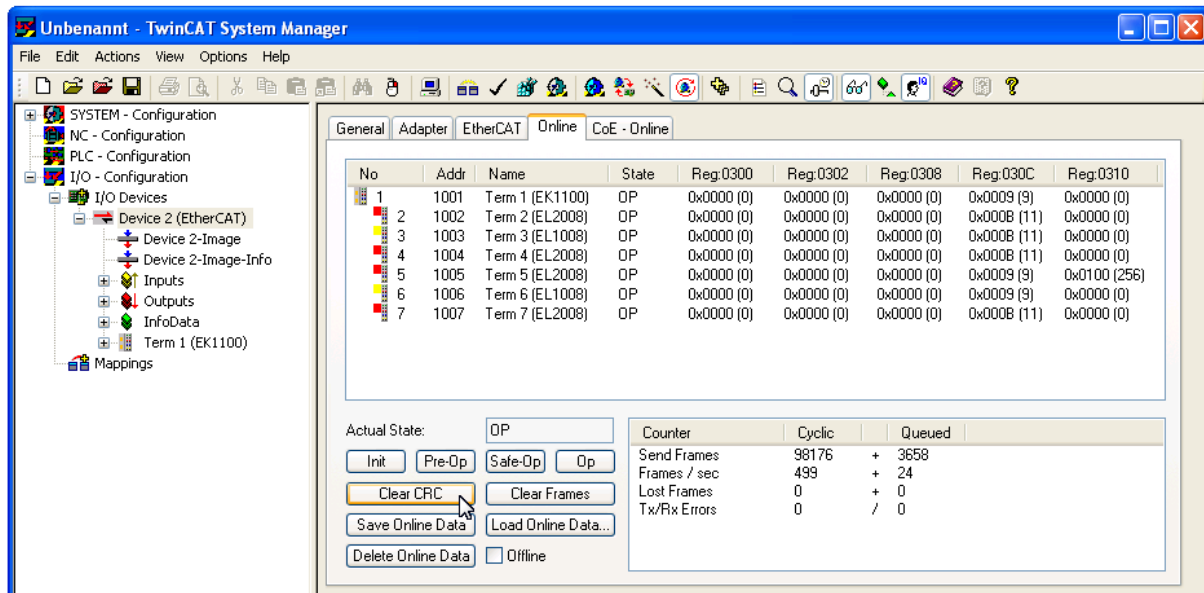

Figure 19: Some power-up errors without significance

7. If you want to clear all the error counters, follow these steps for each ESC.
At first, highlight the ESC with the error counters you want to clear. Right-click into the row and select "Advanced Settings…" from the context menu.
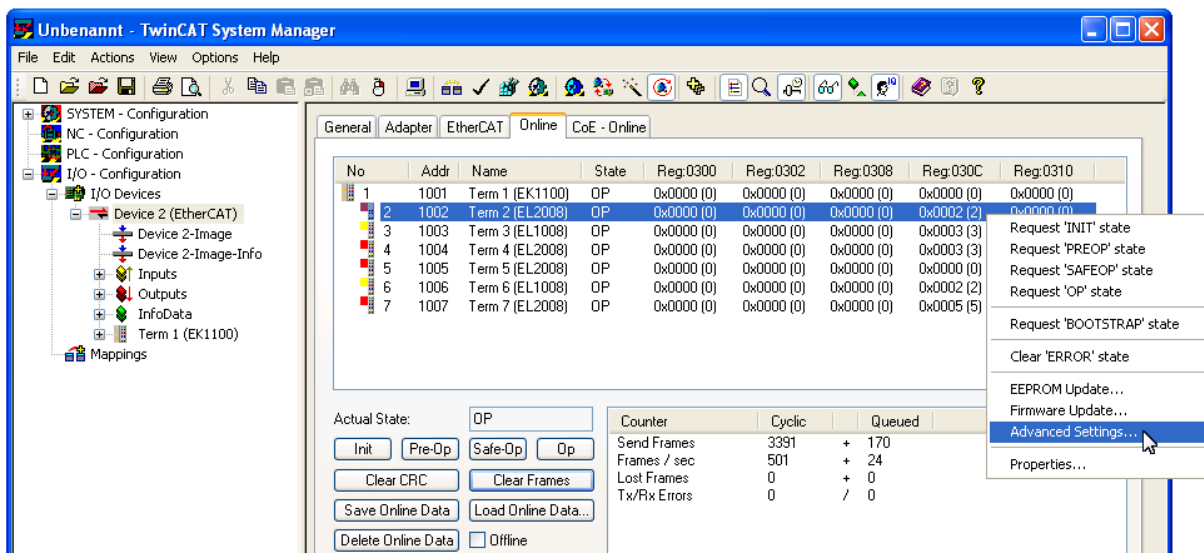

Figure 20: Clear error counters manually in the advanced settings

8. Enter "Start Offset" of 300, activate "Auto Reload" and "Compact view". Press the "Reload" button to see the error counters from 0x0300 ff. The error counters are again shown as 16 bit values.
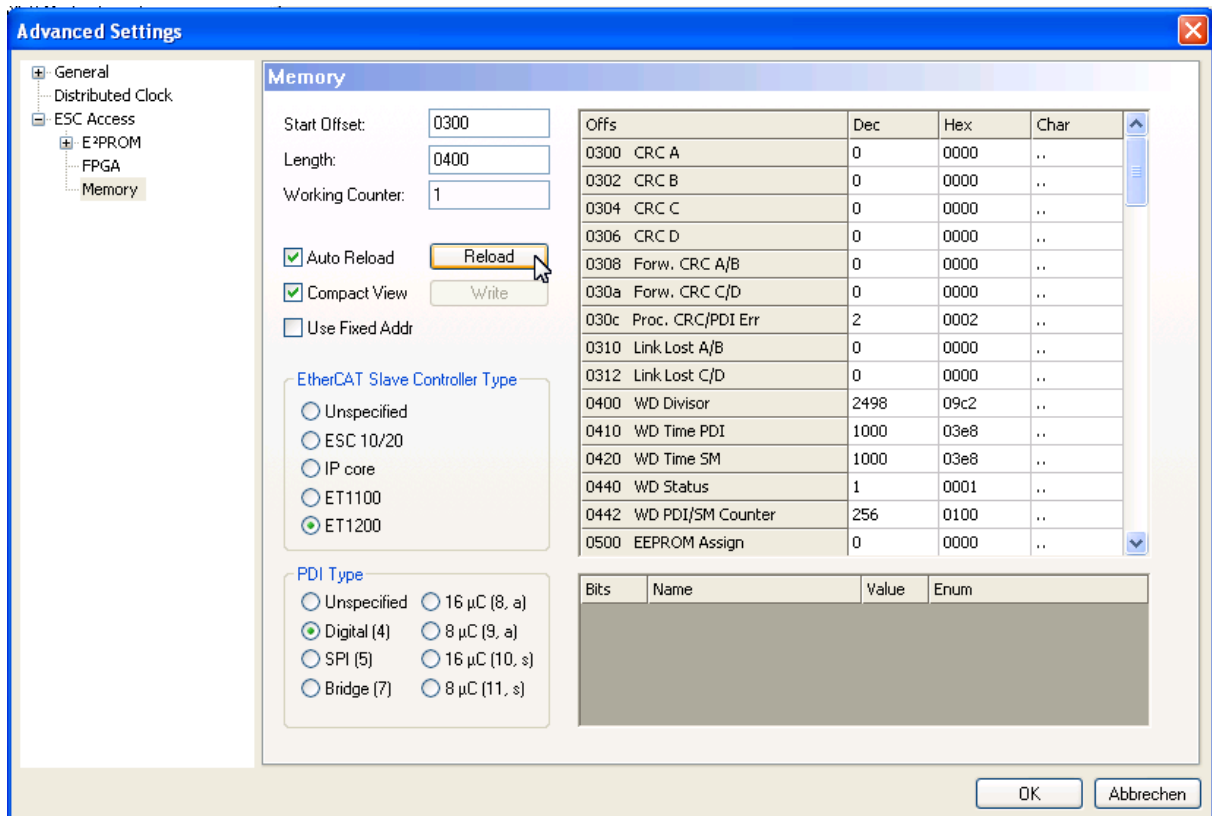

Figure 21: Error counters in the Advanced Settings window

9. To clear an error counter, write any value into it, it is the write accesss and not the value which clears the error counter. Refer to the ESC data sheet section II to find out which error counters are cleared in a group.

Figure 22: Write any value into an error counter to clear it.

10. Press the "Write" button to write the new value, which in fact clears the error counter.
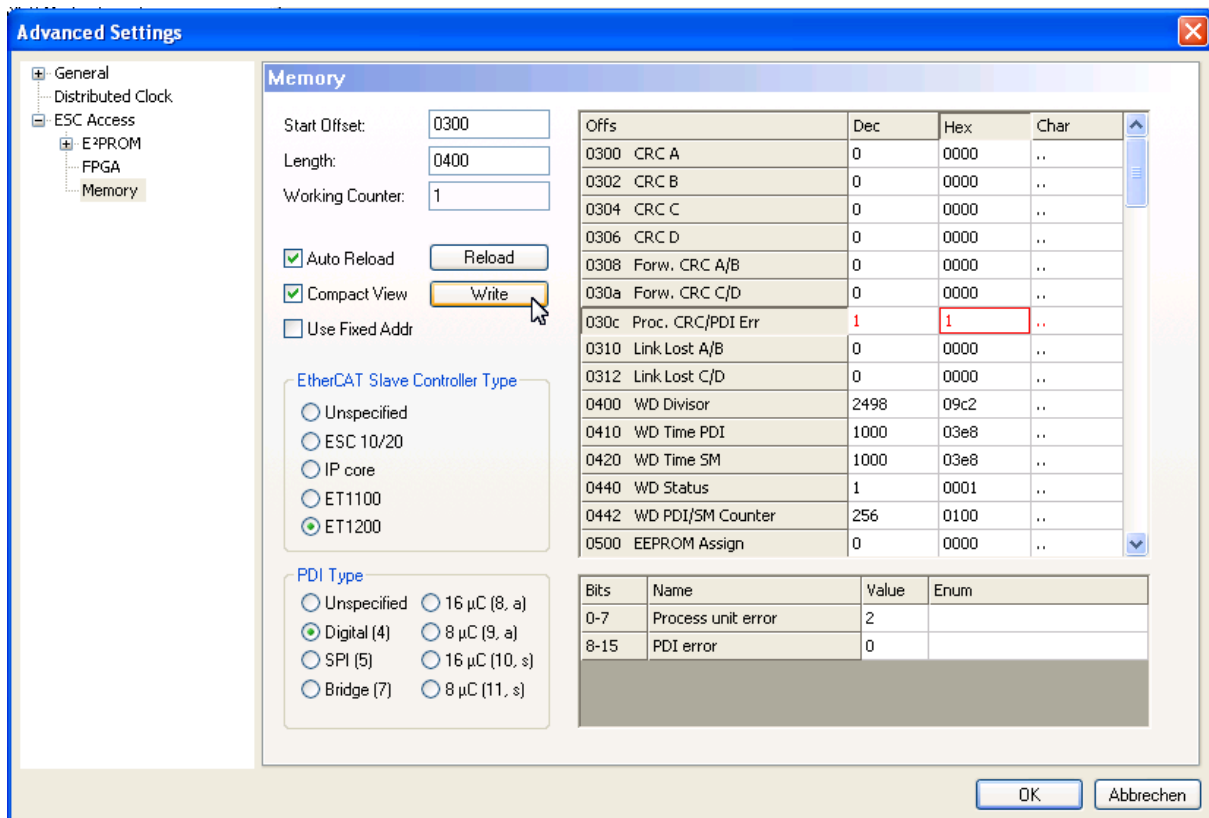

Figure 23: Write to the error counter to clear it

6.2 TwinCAT hints

If communication errors are present, TwinCAT tries to close ports automatically to maintain stable network operation. If you want to observe the instable network part, you have to re-open the ports manually (e.g., by resetting the devices, or writing to the appropriate ESC registers). If TwinCAT closes the ports again, try setting TwinCAT to INIT mode.

6.3   Support and Service

Beckhoff and their partners around the world offer comprehensive support and service, making available fast and competent assistance with all questions related to Beckhoff products and system solutions.

6.3.1   Beckhoff's branch offices and representatives

Please contact your Beckhoff branch office or representative for local support and service on Beckhoff products!

The addresses of Beckhoff's branch offices and representatives round the world can be found on her internet pages: http://www.beckhoff.com

You will also find further documentation for Beckhoff components there.

6.4   Beckhoff Headquarters

Beckhoff Automation GmbH
Eiserstr. 5
33415 Verl
Germany

phone: + 49 (0) 5246/963-0

fax: + 49 (0) 5246/963-198

e-mail: info@beckhoff.com

web: www.beckhoff.com

Beckhoff Support

Support offers you comprehensive technical assistance, helping you not only with the application of individual Beckhoff products, but also with other, wide-ranging services:

- world-wide support
- design, programming and commissioning of complex automation systems
- and extensive training program for Beckhoff system components

hotline: + 49 (0) 5246/963-157

fax: + 49 (0) 5246/963-9157

e-mail: support@beckhoff.com

Beckhoff Service

The Beckhoff Service Center supports you in all matters of after-sales service:

- on-site service
- repair service
- spare parts service
- hotline service

hotline: + 49 (0) 5246/963-460

fax: + 49 (0) 5246/963-479

e-mail: service@beckhoff.com