Evaluation of Ethernet over EtherCAT Protocol Efficiency

Mladen Knežić and Željko Ivanović
Faculty of Electrical Engineering
University of Banja Luka
Banja Luka, Bosnia and Herzegovina
{mladen.knezic, zeljko.ivanovic}@etfbl.net

Abstract—**Real-Time Ethernet networks are commonly found in today's factory automation systems. Beside requirements regarding the real-time communication, these networks often must satisfy conditions for efficient transport of standard Ethernet frames (e.g. TCP/IP traffic). This paper aims to provide evaluation of the efficiency of Ethernet over EtherCAT protocol that is used for tunneling standard Ethernet frames through an EtherCAT network segment.**

Keywords-mailbox; non real-time ethernet traffic; performance indicators; real-time ethernet; throughput

I. INTRODUCTION

Nowadays, Real-Time Ethernet (RTE) networks, which are used for connecting automation devices (sensors and actuators) distributed in the field, are commonly found in factory automation systems. These networks offer many advantages in comparison to the older fieldbus systems of which the most important are: increased installation and maintenance costs efficiency, increased bandwidth, and easier integration with standard computer networks which improves system interoperability [1].

Several RTE protocols, which support communication in real-time, are developed in the last decade by different manufacturers. The most of these protocols is covered by the international standards [2]. Some of them use conventional Ethernet components at the expense of somewhat worse real-time communication capability, while the others enable stringent real-time communication and short cycle times by modifying standard Ethernet on data-link layer. In that way, the direct compatibility with standard Ethernet devices that communicate in non real-time context is not fully preserved, which means that special devices for transfer of standard Ethernet frames must be used. One such protocol, which offers high efficiency of cyclic process data exchange, is EtherCAT.

To specify the capabilities of an RTE device and an RTE communication network, a consistent set of performance indicators (PIs), defined in [3], is used. The supplier of RTE end devices and RTE communication networks should provide at least one consistent set of PIs. However, PIs specified by supplier are often ambiguous and cannot be easily reproduced by the end user. To address this issue, many papers that deal with the evaluation of the PIs for various RTE protocols are published. For example, in [4], several PIs for EtherCAT and Ethernet Powerlink protocols are evaluated. In [5], relevant PIs are evaluated for coordinated motion control application.

One of the PIs, named *non-RTE bandwidth*, is used to indicate the percentage of bandwidth that can be used for non-RTE (NRT) communication on one link (total link bandwidth should be specified as well). However, this PI does not take into account additional protocol overhead (specific to the RTE network) that is used by flow control mechanisms for consistent and reliable transport of standard Ethernet frames over an RTE segment. In that sense, percentage of available bandwidth does not provide realistic figures of the NRT traffic transport efficiency. This paper aims to address this issue and provide evaluation of the efficiency of Ethernet over EtherCAT (EoE) protocol that is used for tunneling standard Ethernet frames through an EtherCAT network segment.

The paper is organized as follows. In Section II, we provide basics of EtherCAT and briefly describe EoE protocol. Section III gives an overview of EoE protocol performance analysis. In Section IV, both simulation and experimental results are presented and discussed, while Section V contains some concluding remarks.

II. ETHERCAT PROTOCOL

EtherCAT is an RTE protocol developed by Beckhoff, which is currently supported and maintained by EtherCAT Technology Group (ETG) organization [6]. EtherCAT uses the concept of summation frame for delivering both cyclic and acyclic data. This concept enables high bandwidth utilization even when large number of networked devices is used. Given that process data in automation systems takes only a few bytes that must be exchanged frequently, and that minimum sized Ethernet frame is 64 bytes, summation frame method introduces much smaller overhead in comparison to the individual frame approaches (e.g. Profinet or Ethernet Powerlink), which leads to better bandwidth utilization.

EtherCAT follows centralized master/slave communication paradigm, which means that one central device, called master, initiates and controls all communication in the network. EtherCAT frames, sent by master, pass through all slaves in the network and return to the master over the same path. This means that EtherCAT always has a logical ring topology

regardless of actual physical topology used in the network. Input and output data are processed in hardware while the frame passes the slave control unit. This method, known as "processing-on-the-fly", enables very short propagation delay (a few hundreds of nanoseconds) of the EtherCAT slave device.

EtherCAT slave unit supports two physical layer interfaces: E-Bus and MII (Media Independent Interface). The former is compatible with the standard Ethernet and is typically used as a backbone for connecting modular devices. It does not require additional components on physical layer and can be used only for short distances. The latter requires additional components (PHYs and transformers) for connecting with the standard 100BASE-TX Ethernet equipment. These components introduce additional propagation delays in the system.

In order to differentiate slave devices, EtherCAT protocol defines data units called EtherCAT datagrams (or telegrams). Every datagram contains header with the information (device address and command) about the device that consumes data. By using so-called logical addressing, it is possible to deliver process data to more than one slave device using only one EtherCAT datagram. This is possible because EtherCAT uses the concept similar to a memory management unit in computer systems. Namely, data placed at certain 32-bit logical address is mapped to the actual physical memory location within the slave using the FMMU (Fieldbus Memory Management Unit). By exploiting symmetry feature and logical addressing, frame size can be further reduced as reported in [7]. Logical addressing is typically used for process (cyclic) data exchange. In case of acyclic data or network initialization, device (node) and position addressing modes are used.

The structure of the EtherCAT frame is shown in Fig. 1. As can be seen from the figure, process data are exchanged without additional overhead, usually using a few EtherCAT datagrams (by means of logical addressing). Mailbox data are exchanged using dedicated datagram with the device address (node addressing), usually in separate Ethernet frame. To enable reliable data transfer, mailbox protocols introduce additional overhead that is used for data flow control. EtherCAT master and application processor in the addressed slave device, access the shared slave memory buffer using the mailbox concept. When producer writes to the buffer, the consumer is not allowed to access the buffer until the write operation is finished. EtherCAT master cyclically polls the state of the slave buffer to check if there is new data, and then initiate the mailbox communication. The maximum size of the mailbox buffer depends on available NRT bandwidth and is configured by EtherCAT master on system initialization.

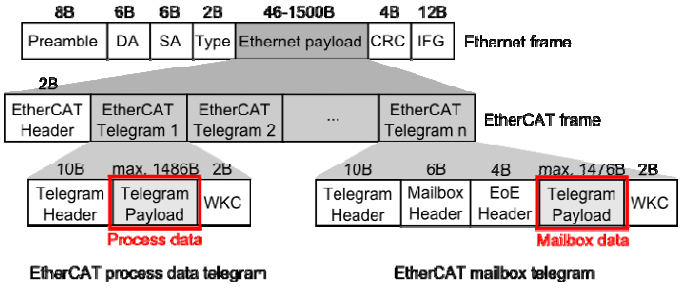

Figure 1. EtherCAT frame structure.

Mailbox protocols are typically used for configuration data exchange with the slave devices that support mailbox communication (usually more complex devices) and for tunneling NRT traffic (e.g. standard Ethernet frames) through an EtherCAT segment. Tunneling of NRT frames is suitable for accessing the field device parameters using the standard protocols used in computer networks (e.g. TCP/IP). In EtherCAT system, EoE mailbox protocol is used for this purpose.

III. Performance Analysis

Non-RTE bandwidth depends on many factors. The most important one is the minimum cycle time of the network. This time is defined as the minimum time necessary for exchange of cyclic input/output data between all networked devices. Minimum cycle time (T_{MCT}) is consisted of the time necessary for frame transmission (T_{Frame}) and overall propagation time in the network ($T_{Network}$),

$$T_{MCT} = T_{Frame} + T_{Network} . \quad (1)$$

Both frame transmission and network propagation times depend on the number of devices in the network. Frame transmission time additionally depends on the scheduling policy used in the system (e.g. number of EtherCAT telegrams used for cyclic data exchange). Cyclic data usually can be exchanged using only a few EtherCAT datagrams by means of logical addressing. In case of ideally symmetrical network (all devices have equal input and output data size), data can be exchanged using only one EtherCAT datagram [7]. In more complex systems, in addition to this datagram, additional datagrams must be used (e.g. for checking the state of the slave, synchronization and checking the mailbox state). General formula for calculating the frame transmission time (for 100 Mb/s system) is

$$T_{Frame} = T_P + T_{EH} + T_{ECATH} + k \cdot T_{DH} + \sum_{i=1}^{k} T_{Di} + T_{IFG} , \quad (2)$$

where: T_P – Ethernet preamble transmission time (640 ns), T_{EH} – Ethernet header transmission time including frame check sequence (1.44 µs), T_{ECATH} – EtherCAT header transmission time (160 ns), k – number of EtherCAT datagrams within Ethernet frame, T_{DH} – EtherCAT datagram header transmission time including working counter (960 ns), T_{Di} – payload transmission time of the i^{th} EtherCAT datagram (depends on process data size and number of slaves), and T_{IFG} – interframe gap time (960 ns).

Network propagation time depends on number of slaves and slave forwarding delay and can be calculated as

$$T_{Network} = 2 \cdot T_{Cables} + m \cdot T_{EBUS} + n \cdot T_{MII} , \quad (3)$$

where: T_{Cables} – total delay introduced by cables (about 5 ns/m), m – number of slaves with E-Bus interface, T_{EBUS} – E-Bus slave forwarding delay (about 300 ns), n – number of slaves

with MII interface, and T_{MII} – MII slave forwarding delay (about 1.2 µs). More details about calculation of transmission and network propagation times and the actual values can be found in [8] and [9].

Actual cycle time (T_{CT}) in the system is set on system start up by system configuration tool. Percentage of total bandwidth available for NRT communication (*NRTBw*) is then

$$NRTBw = \frac{T_{CT} - T_{MCT}}{T_{CT}} \cdot 100 \ [\%]. \quad (4)$$

Maximum mailbox buffer size (in number of bytes), which can also be set by system configuration tool, depends on *NRTBw* and can be calculated as

$$MBXSize = \min\left(\left\lfloor \frac{NRTBw \cdot T_{CT} - T_{OVH}}{T_{BT}} \right\rfloor, \overline{MBX}\right) \quad (5)$$

where: T_{OVH} = 4.16 µs is total overhead transmission time of one EtherCAT frame that contains one datagram (used for mailbox communication), T_{BT} = 80 ns is the time needed for transmission of one byte, and $\overline{MBX} = 1486$ is the maximum mailbox data (including mailbox and EoE headers) that can be transferred using single EtherCAT frame (see Fig. 1).

To address efficiency of EoE protocol, it is important to estimate throughput available for NRT data transfer. EoE protocol divides large chunks of data into so-called fragments whose size must be multiple of 32 and cannot be greater than preset mailbox size. If we assume that only one EoE fragment can be sent within one cycle, NRT throughput (*NRTTp*) can be calculated as

$$EoEPSize = \left\lfloor \frac{MBXSize - EoE_{OVH}}{FragMult} \right\rfloor \cdot FragMult, \quad (6)$$

$$NRTTp = \frac{EoEPSize}{T_{CT}} \ [octets/s], \quad (7)$$

where: *EoEPSize* – EoE payload size, EoE_{OVH} – length of total EoE protocol overhead including mailbox header (10 bytes), and *FragMult* – EoE protocol fragment multiple (32).

IV. Results and Discussion

To address the influence of number of slaves and cycle time on NRT data throughput, we performed a number of simulations and experiments. This section presents some of these results and points out the most important factors for efficient EtherCAT system design in presence of NRT traffic.

A. Simulation Results

By using equations derived in Section III, we performed several simulations in Matlab R2008b software with different scenarios that take into account number of slaves, cycle time and mailbox buffer size. In all simulations we used linear topology, which is commonly found in factory automation networks.

The following assumptions are made:

- Number of slaves is in the range from 1 to 100 (medium sized networks)
- Each slave has 4 bytes input and 4 bytes output data (ideally symmetrical network)
- Average slave propagation delay is 1 µs
- Average cable length between two slaves is 10 m
- All slaves support mailbox communication
- Every cyclically exchanged EtherCAT frame contains 5 datagrams (two for local clock synchronization of slaves, one for process data exchange, one for checking the mailbox state, and one for checking slave state)
- Only one slave supports EoE protocol
- Only one EoE fragment can be sent within one cycle
- Maximum mailbox buffer size calculated using (5) is used.

NRT throughput and bandwidth as a function of number of slaves are shown in Fig. 2. Mailbox buffer size is fixed at 1024 bytes (maximum value supported by off-the-shelf devices, e.g. [10]) and cycle time set to 200 µs.

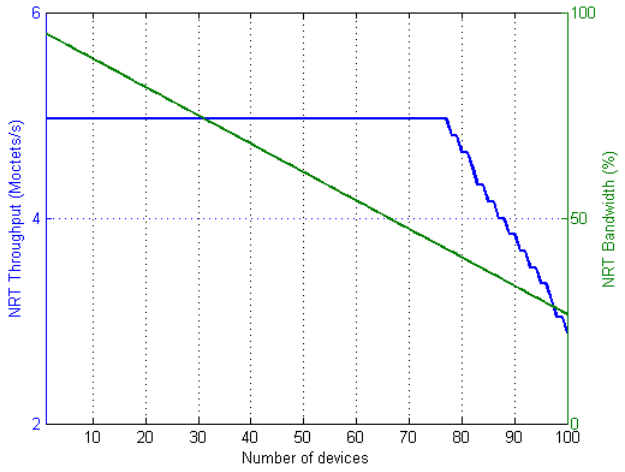

Figure 2. NRT throughput and bandwidth as a function of number of slaves.

As can be seen from this figure, percentage of the bandwidth available for NRT traffic is decreasing with the increase of number of slaves, and is about 25% for 100 slaves. NRT throughput is constant (about 5 Moctets/s) when number of slaves is less than 78 and decreases for larger networks.

To address the influence of the cycle time to NRT throughput and bandwidth, we made a simulation for a case of small network that contains 10 slaves (one with EoE support). The simulation results are shown in Fig. 3.

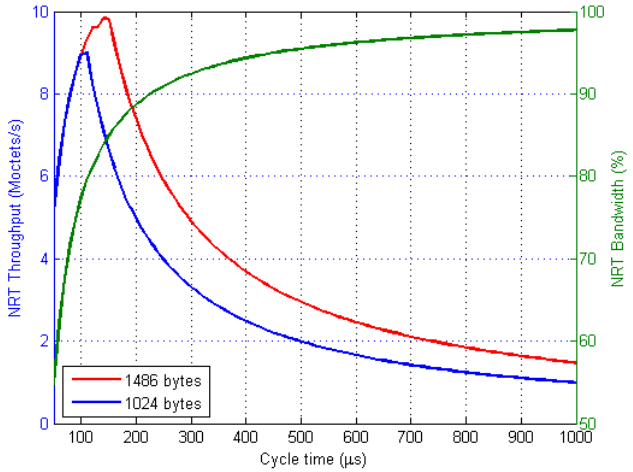

Figure 3. NRT throughput and bandwidth as a function of cycle time.

In this simulation, we used two maximum mailbox buffer size values: theoretical maximum value (red trace) obtained using (5), and maximum value supported by off-the-shelf devices (blue trace). Mailbox size does not affect NRT bandwidth (green trace).

As can be seen from the figure, NRT throughput takes maximum value for cycle times between 100 and 150 µs. In case of longer cycle times, throughput gradually decreases to about 1□1.5 Moctets/s. Mailbox buffer size also affects NRT throughput significantly.

B. *Experimental Results*

Simulation results presented in the previous section give the performance figures in case of the ideal network conditions that do not take into account protocol stack delays introduced by software implemented in devices. To investigate performance of the EoE protocol in real operating conditions, we made some measurements on real network setup. Tested network contains 23 slaves with different functionalities (couplers, digital inputs/outputs, analog inputs/outputs, complex devices with mailbox support, etc.) and physical interface types. For measurement, we used industrial Ethernet probe (Hilscher netAnalyzer) that logs Ethernet frames with 10 ns accuracy. The probe is connected between network and master as illustrated in Fig. 4. In this way, all frames sent by master are logged in both forward and return direction (as mentioned earlier, EtherCAT always has logical ring topology).

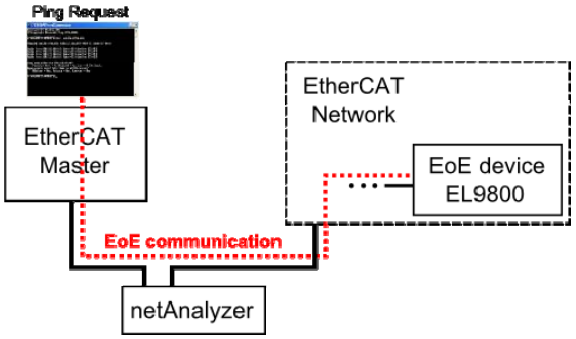

Figure 4. Tested network setup.

To evaluate EoE protocol efficiency, at least one device in the network must support this protocol. We used EL9800 evaluation board from Beckhoff for this purpose. This board is delivered with PIC18F452 microcontroller as an integral part, which is used as an application processor. Microcontroller is connected with EtherCAT slave controller over SPI (Serial Peripheral Interface) bus.

As a master, we used TwinCAT software delivered by Beckhoff, which is commonly used in EtherCAT systems. This software creates virtual Ethernet interface that is used for tunneling Ethernet frames over EtherCAT network segment to the slave which supports EoE protocol.

First, we measured network propagation time for given network setup (Fig. 5). For this purpose, we logged over a million frames with timestamps in both forward and return directions.

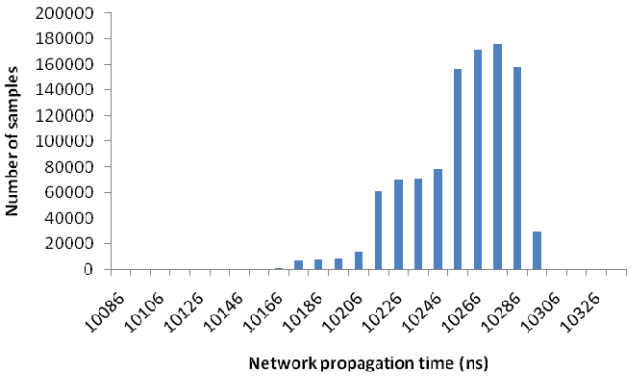

Figure 5. Measured network propagation time.

Average network propagation time is 10.257 µs with 250 ns jitter. For given network setup, EtherCAT frame contains 7 datagrams and its length without preamble, interframe gap and frame check sequence is 284 bytes (this information is obtained from TwinCAT System Manager that automatically generates necessary datagrams for EtherCAT communication). This gives 24.64 µs frame transmission time. It can be concluded that average minimum cycle time for used network is 34.897 µs. This value is important for calculation of *NRTBw* and *MBXSize* parameters using (4) and (5). *NRTBw* obtained by measurements for given network setup is shown in Fig. 6.

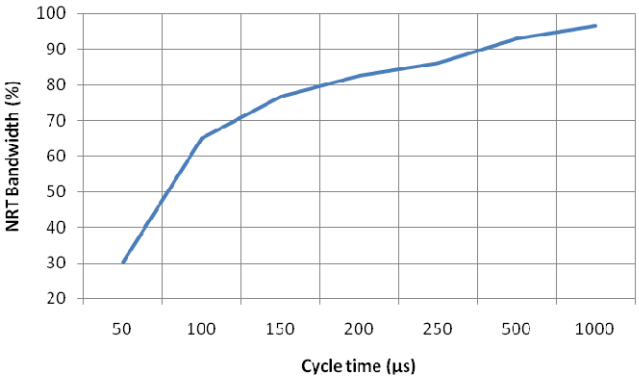

Figure 6. NRT bandwidth for real network setup.

As can be seen, qualitative results are similar to those obtained by simulation (Fig. 3). To be honest, there are some differences when compared to the simulation results regarding actual values. This is mainly due to the fact that in our network setup, we have more slaves although network propagation time of both setups is almost equal (about 10 µs). This leads to the conclusion that average slave propagation delay in real networks is less than previously assumed (1 µs). On the other hand, EtherCAT frame is somewhat larger in comparison to simulation, even though we have fewer data than assumed. This is due to TwinCAT frame scheduling algorithm that uses separate datagrams for digital inputs and outputs, which requires more datagrams in comparison to scenario when symmetrical network is used.

To measure protocol stack delays in devices, we generated NRT traffic by initiating multiple ping echo requests over the virtual Ethernet port in master that can be processed by the application software implemented in EL9800 slave. All EoE traffic is logged by netAnalyzer and analyzed using Wireshark tool. From timestamps of captured frames, we derived stack delays of master and slave. The results are reported in Table I.

TABLE I. MASTER AND SLAVE RESPONSE TIMES

Cycle time [µs]	Master stack delay [ms]	Slave stack delay [ms]
1000	2	6
500	1	7
250	1	16
200	1	48

As can be seen, slave stack delay increases when cycle time is decreased. When cycle time is below 200 µs, device stops to respond to the ping requests. This happens because slave's software cannot service incoming EoE communication since application processor is busy doing other, high priority, tasks that process cyclic data. We must also note that master stack delay is somewhat greater when 1 ms cycle time is set. This relates to the internal timing of the master software.

To measure stack delays when only EoE traffic is serviced by the application processor, we put slave device into state where only mailbox communication is allowed. In this case, stack delays are fixed at 0.6 ms (master) and 4.5 ms (slave). If we use results obtained experimentally in the analysis derived in Section III, NRT throughput changes significantly (Fig. 7).

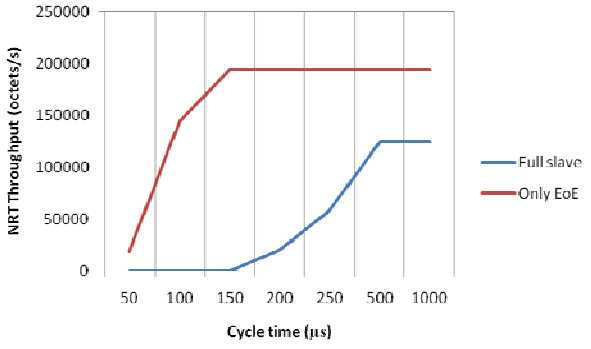

Figure 7. NRT throughput for real network setup.

Honestly speaking, results shown in Fig. 7 can be improved so that NRT throughput qualitatively matches one presented in Fig. 3. This is possible if software in application processor is designed more efficiently so that slave stack delay is reduced.

V. CONCLUSION

Modern factory automation systems require easy and efficient integration with standard computer networks without affecting real-time communication channel used by controller and field devices (sensors and actuators). Because of this, it is of major importance to estimate how efficiently NRT traffic is handled by various RTE protocols developed in the last decade.

This paper tried to address some important aspects of one such solution, known as Ethernet over EtherCAT, which is used by EtherCAT protocol. Theoretical analysis showed how cycle time and number of devices can affect NRT throughput in the network. It is also shown that large percentage of bandwidth available for non-RTE traffic does not always lead to high throughput. Results obtained experimentally shows that special attention must be paid to the design of the slave's software in order to achieve small stack delays, which significantly affects NRT throughput.

REFERENCES

[1] T. Sauter, "The Continuing Evolution of Integration in Manufacturing Automation," IEEE Industrial Electronics Magazine, Vol. 1, No. 1, pp. 10□19, 2007.

[2] M. Felser, T. Sauter, "Standardization of Industrial Ethernet – The Next Battlefield?," in Proc. of the 5th IEEE International Workshop on Factory Communication Systems (WFCS 2004), pp. 413□420, Austria, September 2004.

[3] International Standard IEC 61784-2. Industrial communication networks □ Profiles – Part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC 8802-3, 2007.

[4] L. Seno, S. Vitturi, C. Zunino, "Real-Time Ethernet Networks Evaluation Using Performance Indicators," in Proc. of the 14th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2009), pp. 1□8, Spain, September 2009.

[5] S. Vitturi, L. Peretti, L. Seno, M. Zigliotto, C. Zunino, "Real-Time Ethernet networks for motion control," Computer Standards & Interfaces, Vol. 33, No. 5, pp. 465□476, 2011.

[6] EtherCAT Technology Group (ETG). Available online: http://www.ethercat.org

[7] M. Knezic, B. Dokic, Z. Ivanovic, "Increasing EtherCAT Performance Using Frame Size Optimization Algorithm," in Proc. of the 16th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2011), pp. 1□4, France, September 2011.

[8] M. Knezic, B. Dokic, Z. Ivanovic, "Toplogy Aspects in EtherCAT Networks," in Proc. of the 14th International Power Electronics and Motion Control Conference (EPE/PEMC 2010), pp. T1-1□T1-6, Macedonia, September 2010.

[9] F. Häfele, "Topologievarianten von EtherCAT und deren Einfluss auf die Systemeigenschaften," Kommunikation–Wissenswert, pp. 38–42, December 2008. Available online: http://www.ethercat.org/pdf/german/atp_122008_etg.pdf

[10] Beckhoff Automation GmbH. EL6601, EL6614 – Ethernet switch port terminals. Available online: http://www.beckhoff.com