EtherCAT Network Latency Analysis

Vinh Quang Nguyen
Department of Electrical and Computer Engineering
Sungkyunkwan University
Suwon, Korea
ngqvinh@skku.edu

Jae Wook Jeon
Department of Electrical and Computer Engineering
Sungkyunkwan University
Suwon, Korea
jwjeon@yurim.skku.ac.kr

***Abstract*—EtherCAT, a popular communication protocol based on Ethernet, with a mechanism to transfer data on the fly that enables high speed with high efficiency, plays an essential role in real-time Ethernet-based communication protocols. However, in network-based control systems, it is impossible to transmit a control command to multiple devices at the same time due to network latency. This problem is of particular importance in motion control. In the present work, we analyzed several aspects concerning EtherCAT network latency and carried out some related experiments. As the speed at which the embedded programmable logic controller (PLC) gave commands was increased, the EtherCAT I/O became more likely to respond with signal errors. Moreover, some incorrect signals appeared when short-duration PLC commands were given.**

Keywords—Industrial nework; Real-time Ethernet; EtherCAT communication; EtherCAT distributed clock; network lantency; PLC.

I. INTRODUCTION

Ethernet-based control systems are widely used in industrial and factory automation fields due to their advantages including low cost, maintainability, and scalability. In addition to Ethernet IP, Ethernet Power-Link, … EtherCAT is one of the real-time networks that allows effective use of Ethernet frame in the field or on the shop-floor. Despite its reliance on conventional Ethernet frames, this protocol exploits a peculiar mechanism to access multiple- slaves on the fly. Under this mechanism, the data that coming to or from each slave are encapsulated separately into packets in Ethernet frames, so that the EtherCAT network can transfer data to multiple slaves within a short cycle time [1] [3].

In a distributed system, besides allowing short cycles, synchronization is important because it enables a master to control multiples slaves at the same time, for example motion control systems. In network-based control systems, it is impossible to transmit a control command to multiple devices at the same time because of the network latency [4].

In some EtherCAT devices, network latency compensation is addressed by means of hardware (ET1100, FPGA, …) by using a distributed clock (DC) mechanism based on the IEEE 1588 algorithm. In this way, the master sends a write/read command to the appropriate registers in each slave, and then collects time information from them and calculates the propagation time and offset for each slave during the next cycle. The system time is synchronized with a reference clock (normally the clock in the first slave) [2] [5] [6-12].

Additionally, some devices do not support DC mechanisms, such as the EL1004 digital input module and the EL2008 digital output module, etc., which use EBUS for processing and forwarding EtherCAT datagrams [14-16]. The network latency is significant when using an embedded PLC and these devices.

In this paper, we analyze and evaluate the effects of EtherCAT latency upon system performance by using TwinCAT PLC software and the TwinCAT configuration as the EtherCAT master. The network latency is evaluated based on the changing PLC programming that is encapsulated into EtherCAT datagrams and is processed and forwarded at modules without supported the DC mechanism through the Ethernet cable and EBUS serial medium.

The remainder of this paper is organized as follows: In Section II, the basic EtherCAT protocol and EtherCAT network latency are introduced. Section III presents the concept of the embedded PLC and explains the closed loop motion control mechanism based on the network. Finally, some experiments and conclusions are presented in Section IV and Section V, respectively.

II. BASIC ETHERCAT PROTOCOL

A. Overview

EtherCAT networks are based on an open ring topology in which all of the slaves are connected. A master is connected with the network at the open end. Almost all messages are processed completely by the master. EtherCAT uses standard Ethernet frames (IEEE 802.3) or UDP/IP as shown in Figure 1 and Figure 2 below:

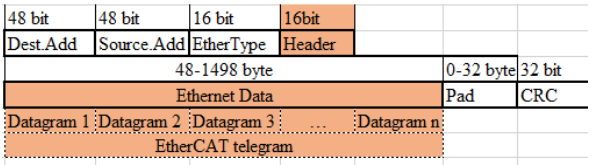

Figure 1. EtherCAT embedded in Standard Ethernet

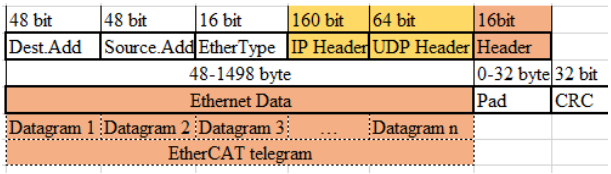

Figure 2. EtherCAT embedded in IP/UDP Ethernet

ISBN:978-1-5090-1666-2/16/$31.00 ©2016 IEEE

Up to 65535 slaves can be connected in an EtherCAT network. Also, EtherCAT permits short cycle time and high bandwidth based on an "on the fly" mechanism that ensures that the master and multiple slaves can exchange data during the very short time of a single Ethernet frame.

The master can be implemented easily as a PC with a full-duplex Ethernet network interface controller. In this case, Beckhoff supported TwinCAT software as the major tool used for developing and monitoring the EtherCAT data frame. The TwinCAT software includes three tools: TwinCAT System Manager, TwinCAT PLC Control, and TwinCAT Scope View. The TwinCAT manager helps to configure the device through the field bus or I/O, by mapping and linking variables between the TwinCAT PLC Control and real I/O devices. The TwinCAT PLC Control is an PC-based embedded PLC that supports multiple PLC programming languages in accordance with the IEC 61131-3 standard. The TwinCAT Scope View is an oscilloscope tool for monitoring I/O signals from the embedded PLC.

In EtherCAT, in addition to Ethernet, a second kind of physical layer called the EBUS is defined, which is based on low voltage differential signaling (LVDS) for connecting modular slaves through a suitable backplane. Because of this, EtherCAT has lower implementation costs than Ethernet.

B. EtherCAT network latency calculation

Cycle time and network latency have an importance influence on performance in real time network. The shorter latency is, the better the system performance is. In this section, we determine the minimum cycle time and compute the network latency between the master and the multiple-slaves in an EtherCAT network.

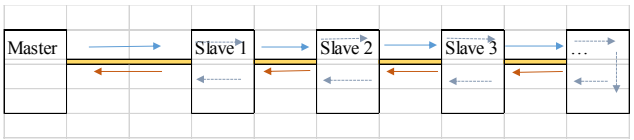

Figure 3. EtherCAT transmission cycle time

Figure 3 shows the transmission cycle time during cyclic communication with EtherCAT. The transmission cycle time includes the processing time, delay time, internal processing time, and forwarding time. In EtherCAT, data (commands) are encapsulated and transmitted by the master. During the cycle time, each slave exchanges (reads/writes) its data into the EtherCAT frame by mean of the "on the fly" mechanism. The master receives all the data from each slave response after the transmission time cycle. The processing time is the amount of time it takes for data to be transferred to each slave from themaster. The delay time is the time that it takes for the data to travel along the cable length. EtherCAT supports two options for the physical layer, Ethernet cable and EBUS, so the delay time depends on the chosen media. EBUS is used for modular slaves. It is low cost, supports Gigabyte Ethernet, and is faster than Ethernet cable. The internal processing time is the time during which the data is processed at each slave. The forwarding time is the time during which the data is forwarded from the slave to the master.

In a system with only one master and one slave, the transmission cycle time of EtherCAT is:

$$T_{cycle} = T_{pro} + 2T_{delay} + 2T_{inter} + T_{forward} \quad (1)$$

Where T_{cycle} is total time of one transmission cycle. T_{pro}, T_{delay}, T_{inter}, and forward are the processing time, delay time, internal processing time, and forwarding time, respectively.

If the distributed system is spatially symmetric, the processing time is almost identical to the forwarding time (the difference in time is very small), so they can be considered equal; (1) can be re-written as:

$$T_{cycle} = 2\,(T_{pro} + T_{delay} + T_{inter}) \quad (2)$$

Applying equation (2) for a spatially symmetric system with N slaves, givens the following transmission time cycle:

$$T_{cycle} = 2\sum_{i=1}^{N} T_{t_i} \quad (3)$$

Where T_t is the total delay time at one slave, equal to $T_{pro}+T_{delay}+T_{inter}$.

The EtherCAT network latency depends not only upon the transmission time, but also on the delay time and the jitter protocol stack. Each EtherCAT device includes an Application layer, Data-link layer (Driver, MAC), and Physical layer (PHY/LVDS), respectively. Additionally, in order to synchronize the clocks between the master and multiple-slaves and to support uniform time in the network, EtherCAT uses the Network layer (IP) and Transport layer (UDP) for multicast purposes, based on IEEE 1588 standard. Details about distributed clock (DC) of the EtherCAT protocol are available elsewhere [3]. Figure 4 shows the mechanism of message exchange between the master and multiple-slaves by means of the delay and jitter protocol stack based on the OSI model:

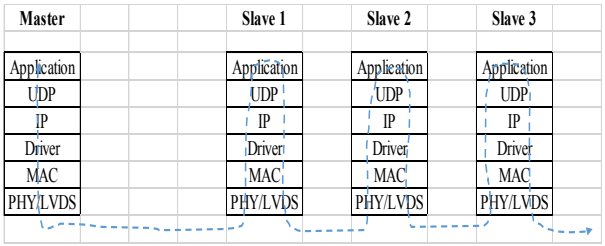

Figure 4. OSI layers based EtherCAT transmission cycle

In the master, the queue frames can appear to create delay time in the data processing phase. In the slave, the delay time arising from queue frames is negligible because the data in the frames sent by the master can be read and written on-the-fly by the slave in a short amount of time.

$$T_{latency} = T_{master} + T_{queue} + T_{cycle} + \sum_{i=1}^{N} T_{slave} \quad (4)$$

Where $T_{latency}$ is the total time during which one frame is transmitted and received at the master. T_{master}, T_{queue}, T_{slave} are the delay time and jitter protocol stack in the master, the

duration of queue frames in the processing phase, and the delay and jitter protocol stack in the slave, respectively.

From equations (3) and (4) the network latency of the N slaves in the EtherCAT network can be expressed as:

$$T_{latency} = T_{Master} + 2\sum_{i=1}^{N} T_{t_i} + \sum_{i=1}^{N} T_{slave} \quad (5)$$

Where T_{Master}, which equal to the sum of T_{master} and T_{queue}, is the time it takes for frames to be processed by the master.

III. EMBEDDED PLC

In motion control, a programmable logic controller (PLC) plays an essential role in the control system. This is also true for a distributed system based on a network for closed loop motion control. In these cases, PLC is the main device for producing data, creating motion profiles and processing data from the lower layer. Normally, in a control network, one master controls multiple slaves by means of PLC programs. The difference between embedded PLC and real PLC is the sample time for the computing algorithms. In the embedded PLC, the sample time depends on the network latency, and on the jitter and delay times in the Operating System (OS). Figure 5 shows a closed loop motion control system based on the EtherCAT network.

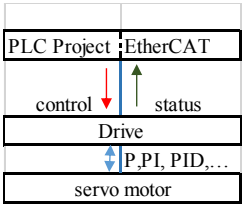

Figure 5. Close loop motion control based on network.

In the first phase, the drive parameters are read by the master through the EtherCAT network. In the second phase, the drive receives a "control" command (reference velocity, reference position, and reference torque) from the master through the EtherCAT network. After receiving commands, the drive uses them for closed loop motion control using P, PI, PD, PID, …. In the next phase, the status parameters are fed-back from the drive into master through the EtherCAT network. These steps are repeated during each PLC cycle.

TwinCAT software includes an embedded PLC that is called TwinCAT PLC Control. In order to monitor and configure the I/O hardware, embedded PLC I/O variables are mapped into Inputs and Outputs in the TwinCAT manager. Due to network latency, the speed command embedded PLC (speed I/O) does not achieve high speed even though the network and hardware support a high baud rate and high speed. The command PLC time can be expressed as:

$$T_{PLC} \geq T_{latency} \quad (6)$$

Where T_PLC is the duration of the PLC I/O needed for cycle communication. If T_PLC is very short, some problems may occur in the system. Using the "high speed" PLC command can cause malfunctions in the control system.

IV. EXPERIMENTS AND RESULTS

In this section, some experiments are proposed to evaluate the degree of influence of the EtherCAT network latency upon the system performance. In addition, the ability of the embedded PLC to response to a high speed network is also considered. The experimental setup is shown in Figure 6:

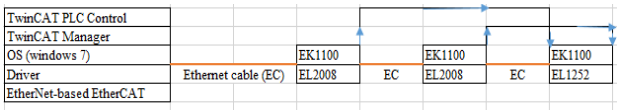

Figure 6. Experiment setup

An EtherCAT coupler connected the EtherCAT master to the EtherCAT terminals (EL2008, EL1252). It converted the passing telegrams from the Ethernet 100BASE-TX to the E-bus signal representation [14-16]. Two programmable digital output (EL2008) terminals were used to generate two square waves. A pair of wires connected the outputs of the two digital outputs to the two input ports of the same digital input. Because these wires had the length, it was assumed that both electrical signals had the same propagation delays.

To compute latency time and to compare the output and input PLC signals, we generated two square waves with varying periods, which were then mapped into the TwinCAT Manager. The digital input module received the same signal from the two digital outputs and gave feedback to the embedded PLC through the serial Ethernet cable-EBUS.

To measure and monitor the PLC signal, we used an oscilloscope connected to the master using the 25-pin printer I/O port. The mappings between all the devices, including the embedded PLC, TwinCAT Manager, Digital Outputs, Digital Inputs and Printer I/O are shown in Figure 7.

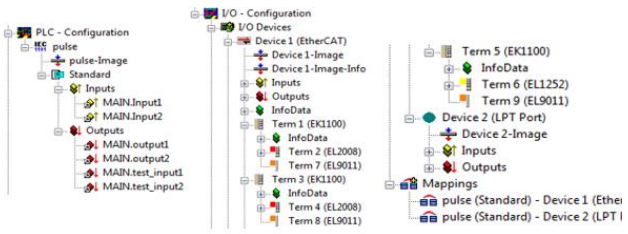

Figure 7. Embedded PLC mapping

To evaluate the influence of the EtherCAT network latency, a series of experiments was conducted in which the square wave output from the I/O PLC was varied in period; each pulses had the ratio of Ton/Toff = 1. The periods used were 400, 200, 100, 60, and 40ms. The corresponding results are shown in Figures 8 through 12, respectively.

Figure 8. Embedded PLC I/O with Ton=200ms, Toff =200ms.

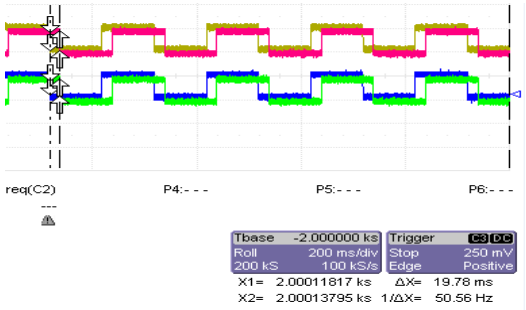

Figure 9. Embedded PLC I/O with Ton=100ms, Toff =100ms

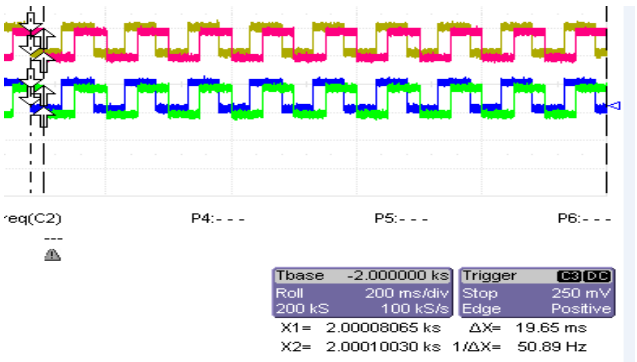

Figure 10. Embedded PLC I/O with Ton=50ms, Toff=50ms

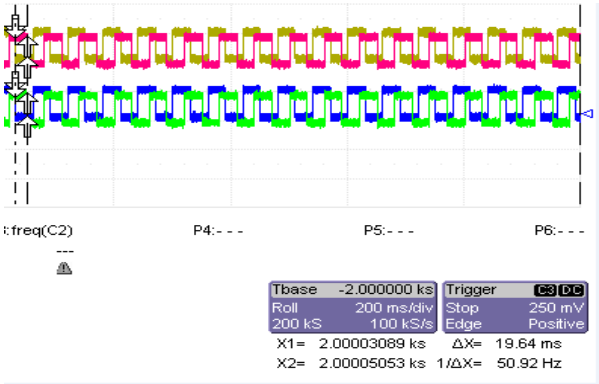

Figure 11. Embedded PLC I/O with Ton=30ms, Toff =30ms

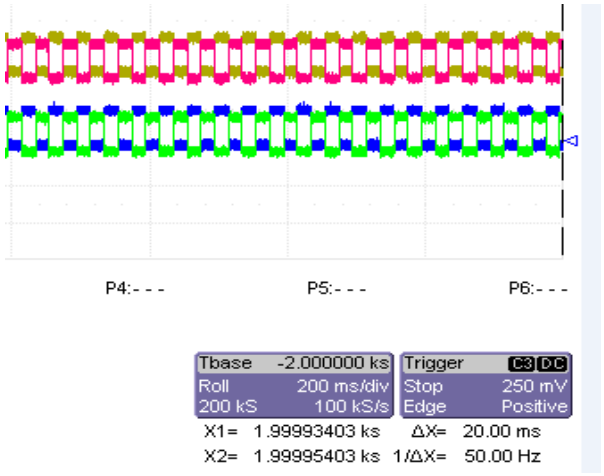

Figure 12. Embedded PLC I/O with Ton=20ms, Toff =20ms

We recognized that the EtherCAT network latency (ΔX) was fairly large (≈20ms) and was constant when we used the embedded PLC. In conditions of longer pulses (400 and 200ms) the I/O slave modules responded to the master more precisely. The deviation between the PLC output signals and the I/O hardware increased when the PLC pulse periods were reduced (to 100 or, 60ms). Moreover, the signals from the slave modules can differ from the original PLC signals (master signals) if the pulse period from the master is less than or equal to twice the network latency ($T_{period} \leq 2$ x 20ms). The effects of network latency upon accuracy of signals are shown in Figure 13:

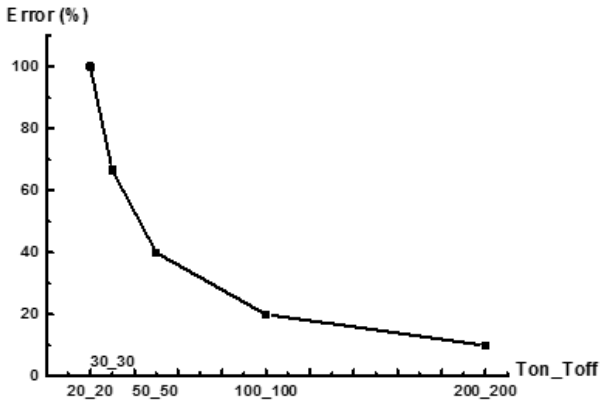

Figure 13. The effects of network latency to EtherCAT system

Equation (3) can be used to compute the cycle time of EtherCAT. The cycle time is very small (≈12µs) [14-16]. Based on Equation (6), the network latency depends primarily on T_{Master}. The time it takes for mapping and processing data from the embedded PLC to the I/O hardware using TwinCAT

software is longer than the cycle time. Therefore, the embedded PLC should be replaced by PLC hardware (CX1020) to ensure adequate performance in a real time network.

V. CONCLUSION

In this paper, we have analyzed the EtherCAT network latency and its influence on real time network performance. We have also presented an overview of embedded PLCs. An experimental model has been established, including one EtherCAT master, that uses PLC software as the application layer, and three EK1100 EtherCAT couplers. Experiments showed that the network latency is significant when using an embedded PLC. The system becomes unstable when the PLC command processing speed is shorter than the network latency. This limitation can be overcome by using a PLC hardware, applying distributed clock (DC) for synchronization, and reducing the network latency time. In the future, this problem will be considered in detail in order to improve the performance of the system.

ACKNOWLEDGMENT

This research was supported by the MOTIE and KEIT through the ATC program.

REFERENCES

[1] Max Felser, member. Real-Time Ethernet –Industry Prospective. Proceeding of the IEEE, vol. 93, No. 6, June 2005.

[2] Kun Ji, Won-jong Kim. Real-Time Control of Networked Control Systems via Ethernet. International Journal of Control, Automation, and Systems, December 2005

[3] EtherCAT Technology Group. EtherCAT Communication: Communication Principle.

[4] Trung Vien Tu, Jin Ho Kim, Kien Minh Le, Jae Wook Jeon. Method for Increasing the Accuracy of Synchronous Control by Compensating for Network Latency. Mecatronics, Tokyo, Japan, 2014.

[5] EtherCAT Technology Group. EtherCAT: The Ethernet Fieldbus.

[6] Jin Ho Kim, Suk Hyung Seo, Jung Hoon Chun, Jae Wook Jeon. Distributed clock synchronization for Industrial. 8th IEEE World Conference on Factory Communication Systems (WFCS), 2010.

[7] Gianluca Cena, Ivan Cibrario Bertolotti, Stefano Sacnzio, Adriano Valenzano, Claudio Zunino. Evaluation of EtherCAT distributed clock Performance. IEEE Transactions on Industrial Informatics, Feb 2012.

[8] Gianluca Cena, Ivan Cibrario Bertolotti, Stefano Sacnzio, Adriano Valenzano, Claudio Zunino. On the Accuracy of the Distributed Clock Mechanism in EtherCAT. 8th IEEE World Conference on Factory Communication Systems (WFCS), 2010.

[9] Gianluca Cena, Ivan Cibrario Bertolotti, Stefano Sacnzio, Adriano Valenzano, Claudio Zunino. Performance Evaluation of the EtherCAT Distributed Clock Algorithm. IEEE International Symposium Industrial Electrics (ISIE), 2010

[10] Hans Webel. High Precision Clock Synchronixation according to IEEE 1588 Implementation and Performance Issues.

[11] Jeremy Robot, Jean-Philippe Gearges, Eric Rondeau, Thierry Divoux. Minimum cycle time analysis of Ethernet-based realtime protocols. International Journal of Computers, Communications and Control, 2012.

[12] Multi-Axis Position Control by EtherCAT real-time Networking.

[13] Gunnar Prytz. A Performance Analysis of EtherCat and ProfiNet IRT. IEEE International Conference Emerging technologies and Factory Automation (ETFA), 2008.

[14] EK1100 | EtherCAT Coupler.

[15] EL1252 | 2-channel digital input terminal with time stamp.

[16] EL2008 | 8-channel digital output terminal 24 V DC, 0.5 A.