ISSN 2395-695X (Print)
ISSN 2395-695X (Online)

International Journal of Advanced Research in Biology Engineering Science and Technology (IJARBEST)

Vol. 2, Special Issue 10, March 2016

INDUSTRIAL STANDARD COMMUNICATION OF APERIODIC REAL-TIME MESSAGES OVER ETHERCAT NETWORKS

Arun D

PG Student [VLSI Design], Department of Electronics and Communication Engineering

TRP Engineering College, Trichy, Tamil Nadu, India.

ABSTRACT: **EtherCAT is a real-time Ethernet protocol mainly designed for periodic real-time transmission in factory automation. Real-time Ethernet (RTE) technologies provide high bandwidth and are able to meet the requirements of industrial real-time communications. Among RTE protocols, the EtherCAT standard is suitable for motion control and closed-loop control applications, which require very short cycle times. As EtherCAT was specifically devised for periodic traffic, aperiodic real-time transmissions are far from being efficient, as they entail long cycle times. To overcome this limitation, a general framework for aperiodic real-time messages over EtherCAT networks, which uniformly covers both static and dynamic priority and allows for very short cycle times. INET framework provides a schedulability analysis for both static priority and dynamic priority scheduling, and simulative assessments obtained through OMNeT++ simulations.**

INDEX TERMS: **EtherCat, High Bandwidth, Short Cycle Time, Real-Time Ethernet(RTE), INET framework, OMNeT++**

I. INTRODUCTION

ETHERCAT is a real-time Ethernet (RTE) network that is becoming increasingly popular in factory automation environments, in particular, at the shop-floor level. Although it relies on the conventional Ethernet technology, the communication protocol exploits a peculiar approach to access slave devices, that resembles closely the summation frame of INTERBUS. Moreover, a logical addressing scheme is defined which permits small-sized process data to be packed further. As a consequence, communication efficiency is very high (up to 90%) which achieves very short cycle times. This makes this solution particularly attractive for connecting decentralized peripherals (i.e., remote I/O devices) to the application master (either a real or Linux PC). Besides efficiency, another feature that makes EtherCAT appealing in a number of application domains such as, e.g., motion control, is the availability of a simple yet effective mechanism that enables devices to operate in a synchronized way. In particular, modern industrial networks must offer support for both time-driven and event-driven control applications. In time-driven applications, messages are periodically transmitted and control actions are taken at constant rate while in event-driven applications, messages are transmitted when one or more trigger events occur. For example, closed-loop control applications typically generate periodic messages with deadlines of approximately 1ms. However, these application may also require the transmission of aperiodic real-time messages that have to be accommodated in the overall traffic schedule without affecting periodic messages. Recently, real-time Ethernet (RTE) technologies have become increasingly popular, as they offer high bandwidth, meet the requirements of industrial real-time communication, and allow for vertical integration of different levels in the automation pyramid. Precise clock synchronization is a feature which is becoming more and more important in control networks. As pointed out in unsynchronized networks usually suffer from non-negligible jitters. A number of approaches have been described in the literature aimed at supporting synchronized operations in popular networks such as, e.g., CAN, new-generation automotive networks, and even WLANs. Many real-time Ethernet solutions rely on precision time protocol (PTP) or its variants, such as the precision transparent clock protocol (PTCP) used in PROFINET. The performance of these approaches are well-known and prove to be satisfactory even in s/w implementations. Modifications of PTP have been defined in order to improve, e.g., fault tolerance or interoperability.

Thanks to its ring topology, EtherCAT provides a daisy-chain topology and a master/slave architecture in which the master periodically transmits a standard Ethernet frame that embeds an EtherCAT frame containing multiple telegrams (as shown in Fig. 1). Slaves read and/or write data in the telegram by processing the frame “on-the-fly,” so when a byte arrives to a slave, it is processed and transmitted to the next slave without waiting for the complete reception of the Ethernet frame. The last slave in the chain transmits the frame back to the master by exploiting the full-duplex capability of Ethernet. The main goal of this paper is to focuses on the

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)

International Journal of Advanced Research in Biology Engineering Science and Technology (IJARBEST)

Vol. 2, Special Issue 10, March 2016

EtherCAT protocol, assess several properties of this mechanism and, in particular, the accuracy and precision with which coordinate actions can be carried out by devices connected through the network. To this extent, a number of thorough measurement campaigns were carried out on real-world devices, in order to determine to which degree the actual performance matches the figure provided by OMNet++. All these components can be implemented in hardware in the case of FPGA/ASIC ESC. The minimum required component for a fully working slave consist of a programmable FPGA and two EEPROMs together with a power supply and other auxiliary components and connectors. The FPGA itself contains the ESC, which is configured to use the microcontroller PDI interface. Therefore EtherCAT master will be an Linux based PC to control the FPGA slave design.

This paper is structured as follows. Sections II describes the EtherCAT protocol basics and structure. Section III presents the analytic assessment and frame processing. Section IV deals with simulative assessments of omnetpp. Finally, Section V conclude the paper with simulation results.

II. ETHERCAT PROTOCOL BASICS

The basic EtherCAT protocol only supports single-master network configurations, where the master communicates with the other devices (namely, the slaves) by sending them suitable telegrams. Each telegram encodes exactly one EtherCAT command, that is, read, write or some combination of these operations, as well as the address of the item to be accessed. In this context, an item can be either a register or a memory location on one or more devices. Two addressing modes are foreseen, that is logical and physical. The latter, in turn, can be either configured or positional. More than one telegram can be included in the same EtherCAT frame, which yields high network throughput. In turn, each EtherCAT frame can be encapsulated in either an Ethernet frame or an UDP message. Only the first option was considered in the following, in that it ensures the highest degree of real-time performance.

EtherCAT networks are based on a physical ring topology. All EtherCAT frames are generated by the master, to which they come back after they have passed through all the slaves. Each slave processes and propagates the frame onward in a very short time (usually less than 1µs). When an EtherCAT slave recognizes a command of its interest, it executes the related actions on-the-fly, by reading and/or changing directly parts of the Ethernet frame which is being relayed.

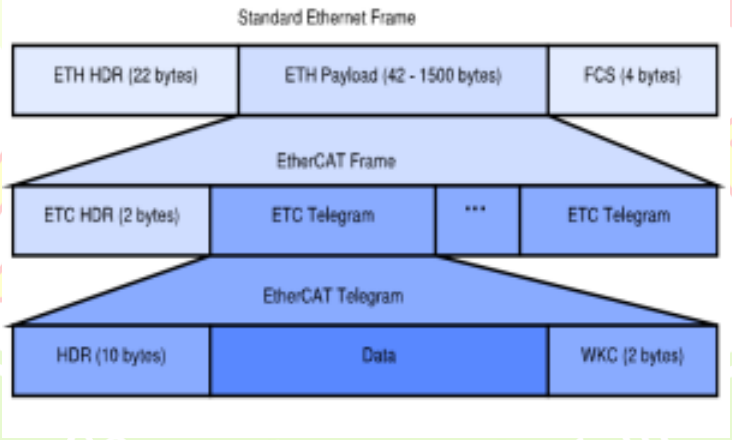

Fig. 1. Structure of an EtherCAT frame containing multiple telegrams.

However, such a mechanism would entail long cycle times, as the master must provide room in the EtherCAT frame for any slave that has the potential to transmit aperiodic real-time traffic, regardless of whether such a slave actually has traffic to transmit. For instance, in a network with 20 slaves, each with the potential for transmitting 32-byte-long aperiodic real-time messages, the master should provide 20 telegrams for each cycle.

III. ANALYTIC ASSESSMENT

A. Frame Propagation and Timing

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)

International Journal of Advanced Research in Biology Engineering Science and Technology (IJARBEST)

Vol. 2, Special Issue 10, March 2016

In EtherCAT networks, if the data to be embedded in the Ethernet frame exceed the maximum payload size (i.e., 1500 bytes), multiple Ethernet frames will have to be trans mitted by the master to complete a cycle. However, this paper addresses the case of very short cycle times (e.g., in the order of 100 µs), and a maximum payload Ethernet frame has a cycle time of 150 µs.

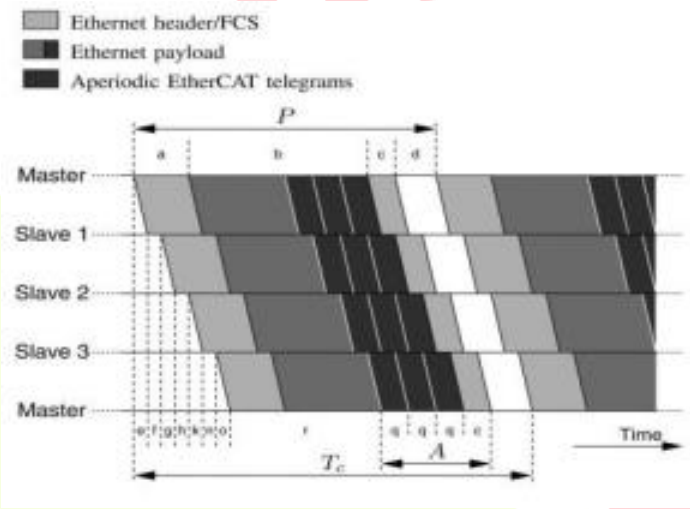

Fig. 2. EtherCAT frame processing sequence.

Hence, here only a single frame cycle is considered. Fig. 2 illustrates the propagation of the Ethernet frame and the related terminology and notations (also summarized in Table I). This figure shows a scenario with an Ethernet frame that is transmitted by the master to a chain of three slaves, Fig. 2. EtherCAT frame processing sequence and then goes back to the master according to the daisy chain topology. In Fig. 2, the master is represented twice to illustrate separately the transmission (top-side) and the reception (bottom-side). The propagation of the Ethernet header and the Frame Check Sequence (FCS) field is drawn in light gray. The Ethernet payload is drawn in two shades of gray (gray/dark gray). In the payload, we highlight in dark gray the EtherCAT telegrams (three in the figure) dedicated to the transmission of ApMs. All the notations are described in Table II. The master periodically sends an Ethernet frame with period P. Each frame then propagates to the slaves through the network. Slaves are labeled following the frame reception order, so slave 1 is the one that receives the frame first, while slave m is the last. The time elapsing from the transmission of a frame by the master to its reception, as a response is equal to the time needed by the signal to propagate through the medium (Tpr), plus the time needed by the m slaves to process the frame (Tde). From Fig. 2, it is possible to observe that the slaves process the frame "on-the-fly," i.e., each frame starts to be transmitted before it has been fully received from the preceding node. This allows a low end-to-end latency and enables the transmission of an ApM even if the external event generating the ApM arrives during the reception of the Ethernet frame. The availability of aperiodic telegrams must be carefully analyzed, as illustrated in the following.

TABLE I

SUMMARY OF NOTATION

Symbol	Definition
T_{et}	Sum of the transmission times of the Ethernet header and Frame Check Sequence (FCS) fields.
T_{ec}	Time necessary to transmit the EtherCAT frame.
T_{pr}	Propagation delay over the communication medium that is equal to $T_{pr} = u \sum_{k=0} l_k$
T_{de}	Frame delay, that is, mT_{sv}
T_{if}	Inter-frame gap, i.e., the time between the end of the transmission of the Ethernet frame and the start of the transmission of the next one.
T_{ap}	Time between the start of the transmission of the Ethernet frame and the start of the transmission of the first aperiodic telegram.

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)

International Journal of Advanced Research in Biology Engineering Science and Technology (IJARBEST)

Vol. 2, Special Issue 10, March 2016

B. EtherCAT Mechanism

Rather than transmitting data to each slave node within the network, EtherCAT passes the Ethernet frame through each salve node. When passing through, the data is read and written in units of several nanoseconds to each corresponding area within the frame at each slave node. Upon passing through all EtherCAT slaves without being stopped along the way, the Ethernet frame transmitted from the EtherCAT master is sent back by the last slave, and returns to the EtherCAT master after passing through all frames again by achieving daisy- chain topology in the Fig.3 . With this mechanism, the high-speed data transmission and realtime capability are achieved.

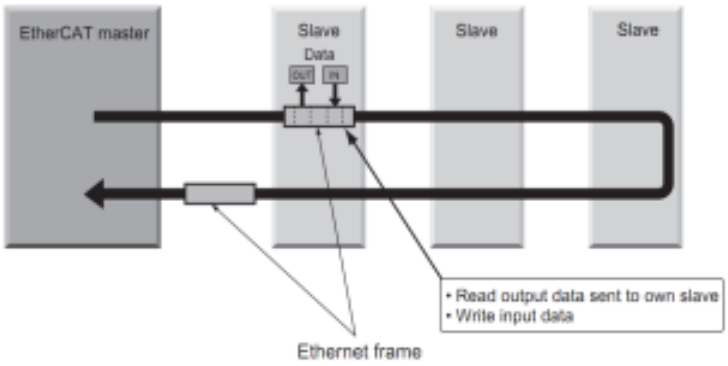

Fig. 3. Daisy-chain topology of data transmission.

Periodic data exchange between the EtherCAT master and the EtherCAT slaves is performed by the "EtherCAT telegram" that is stored directly in the Ethernet frame. Each "EtherCAT telegram" consists of 1 or multiple slave addresses, data, and working counters(check bits). If an Ethernet frame is compared to a "train," an EtherCAT telegram would be a "compartment."

IV. SIMULATIVE ASSESSMENTS

A. Simulation Model Assessment

To assess the performance of the proposed approach, a suitable simulation model was developed using the OMNeT++ framework. In the simulation model, two kinds of nodes are implemented, i.e., 1) the EtherCAT Master and 2) the EtherCAT Slave. The EtherCAT Master is composed of a EtherHost module. The first module periodically generates Ethernet frames and collects statistics. The Master transmits the frame in one-byte long packets and transmits each byte every 0.08 μs (i.e., the byte time of the 100 Mb/s Ethernet).

In this way, the timing of the simulation model is compliant with that of the EtherCAT standard. The EtherCAT Slave module provides several functionalities. Among them, the EtherCAT, which supports both the periodic telegrams foreseen by the standard and the aperiodic telegrams of the proposed PdS approach, the forwarding mechanisms for incoming packets, the read/write and management functions of the ApMs local queue, and the slave application layer.

B. Simulation Perspective

The OMNeT++ IDE defines the Simulation Perspective so that it is specifically geared towards the design of simulations. The Simulation Perspective is simply a set of conveniently selected views, arranged to make the creation of NED, INI and MSG files easier. If you are working with INI and NED files a lot, we recommend selecting this perspective showen in the flow chart of Fig.4. Other perspectives are optimized for different tasks like C++ development or debugging.

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)

International Journal of Advanced Research in Biology Engineering Science and Technology (IJARBEST)

Vol. 2, Special Issue 10, March 2016

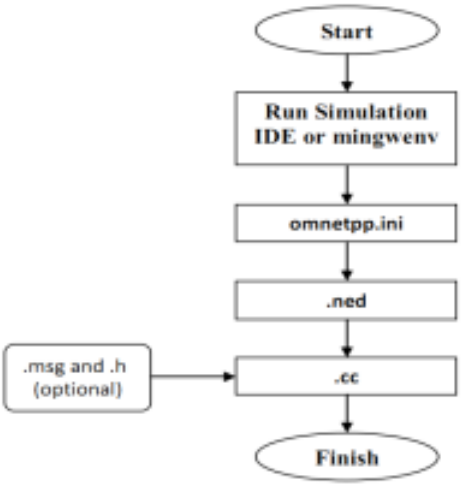

Fig.4. Flow chart of OMNeT++ works.

C. Simulation Result

Telegrams are transmitted from ethercat_master to ethercat_slave through the medium at which the packet contains request and response token along with the telegram. On the other end, slave receives the telegrams of packet and response to the master and transmit the telegrams to the following slaves as shown in the Fig.5. The process continuous like daisy-chain topology.

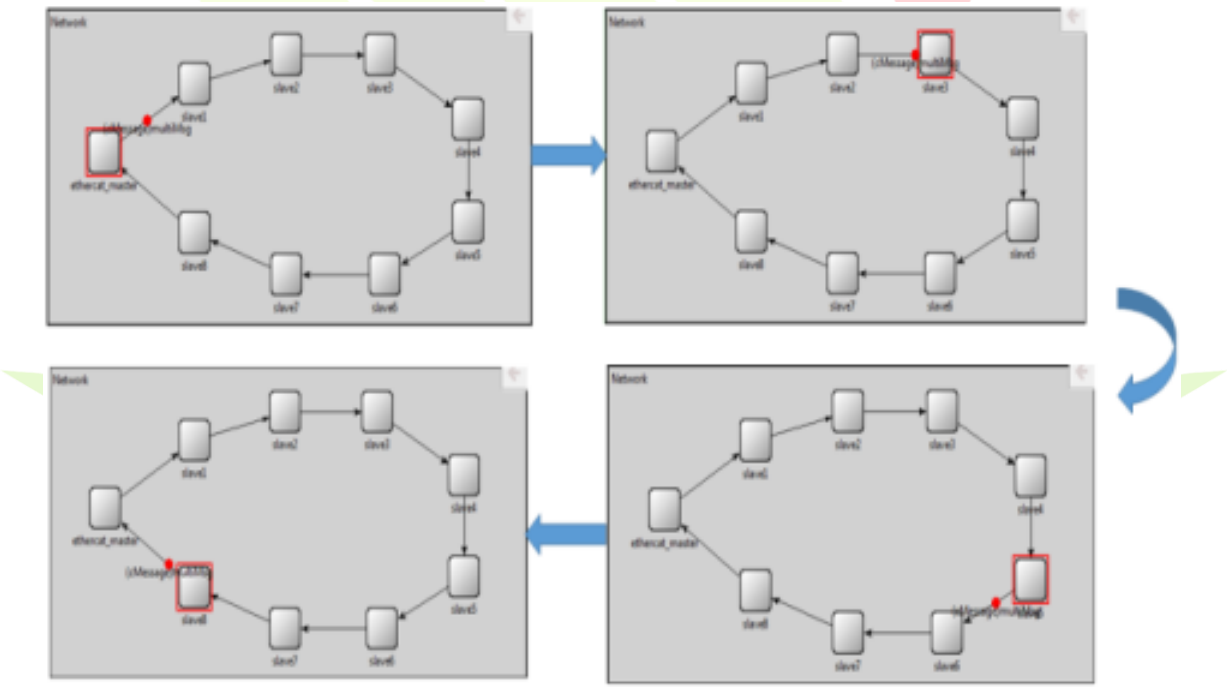

Fig.5. OMNeT++/TKenv- Simulation Output

Number of event that the EtherCAT master/slave transmission was calculated and its response time and delay between the nodes are specified. Time taken to send and receive the packets are also calculated and mentioned in the Table II. In the process of transmission Event, Time, Source/Destination, Name and Resp/Req Information's are generated.

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)

International Journal of Advanced Research in Biology Engineering Science and Technology (IJARBEST)

Vol. 2, Special Issue 10, March 2016

TABLE II
PARAMETERS OF SIMULATION

S.No	Parameter	EtherCAT Protocol
1	Bandwidth	Up to 1 Gbps
2	Data bytes/frame	128
3	Flexibility	High
4	Error Detection	32 bit CRC
5	Latency Jitter	< 1µs
6	Messaging	Event & Time triggered
7	Payload	1500 bytes

V. CONCLUSION

In this work, a mechanism to deal with the problem of providing support for aperiodic real-time traffic over EtherCAT networks was presented. The proposed mechanism provides the possibility for slaves to transmit aperiodic real-time messages under static and dynamic priorities, respectively, while maintaining the compatibility with the EtherCAT standard. The proposed approach is suitable for event-triggered applications, as the ApMs can be transmitted while maintaining short cycle times. The paper proposed a general analysis for the response times(i.e., in the order of less than1µs) which can be used to assess the feasibility of a static priority message set. As far as dynamic priorities are concerned. This paper provided extensive simulative assessment of transferring data between master and slave with high bandwidth and fastest network at the response time.

REFERENCES

[1] Lucia Lo Bello, Enrico Bini, Gaetano Patti, "Priority-Driven Swapping-Based Scheduling of Aperiodic Real-Time Messages Over EtherCAT Networks," *IEEE Transactions on industrial informatics, vol. 11, no. 3, june 2015*
[2] K.W. Schmidt and E. G. Schmidt, "Distributed real-time protocols for industrial control systems: Framework and examples," *IEEE Trans. Parallel Distrib. Syst., vol. 23, Oct. 2012.*
[3] A.Pawlowski, A.Cervin, J. L.Guzman, and M.Berenguel, "Generalized predictive control with actuator deadband for event-based approaches,*" IEEE Trans. Ind. Informat., vol. 10, Feb. 2014.*
[4] E. Moradi-Pari et al., "Design, modeling, and simulation of on-demand communication mechanisms for cyber-physical energy systems," *IEEE Trans. Ind. Informat., vol. PP, 2014.*
[5] S.Bose et al., "Shipboard power systems reconfiguration: A cyber physical framework for response time analysis," *IEEE Trans. Ind. Informat., vol. 10, Feb. 2014.*
[6] D.M.E. Ingram, P. Schaub, R. R. Taylor, and D. A. Campbell, "Performance analysis of IEC 61850 sampled value process bus networks," *IEEE Trans. Ind. Informat., vol. 9, Aug. 2013.*
[7] W. Kang, K. Kapitanova, and S. H. Son, "RDDS: A real- time data distribution service for cyber-physical systems," *IEEE Trans. Ind. Informat., vol. 8, May 2012.*
[9] T. Sauter, "The three generations of field-level networks Evolution and compatibility issues," *IEEE Trans. Ind. Electron., vol. 57, Nov. 2010.*

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)

International Journal of Advanced Research in Biology Engineering Science and Technology (IJARBEST)

Vol. 2, Special Issue 10, March 2016

[10] P. Gaj, J. Jasperneite, and M. Felser, "Computer communication within industrial distributed environment — A survey," *IEEE Trans. Ind. Informat., vol. 9, Feb. 2013.*