the EXTENSION

A Technical Supplement to Control Network

JULY–AUGUST
Volume 5 Issue 4

Introduction to Real-Time Ethernet II

By Paula Doyle, a doctoral researcher with the Circuits and Systems Research Centre at the University of Limerick in Ireland

INTRODUCTION

In "Real-Time Ethernet I", we introduced the basic concepts of Ethernet's capacity to deliver a real-time (RT) communication system. "Real-Time Ethernet II" introduces some of the RT solutions available to industry today*: PROFInet, EtherCAT and ETHERNET Powerlink. It also provides an introduction to a single standard, IEEE 1588 that is growing in popularity amongst RT Ethernet developers to provide sub-microsecond synchronization accuracy of distributed clocks over Ethernet.

* EtherNet/IP is included in the full article available at **http://www.ccontrols.com/pdf/volume5n4.pdf**

IEEE 1588

IEEE 1588 [1] specifies "A protocol to synchronize independent clocks running on separate nodes of a distributed measurement or control system to a high accuracy and precision." IEEE 1588 is, or will be, incorporated into EtherNet/IP, ETHERNET Powerlink, EtherCAT and PROFInet—making it a popular standard for delivering RT over Ethernet.

In IEEE 1588, all network nodes down to the transducer level contain an IEEE 1588 clock, synchronized with all network peers (see **Figure 1**) using Precision Time Protocol (PTP). At device level, sensors can timestamp their data locally and actuators can operate at a precise time, avoiding stack and application delays between transducer and controller. The accuracy of the system depends on the synchronization of local RT clocks.

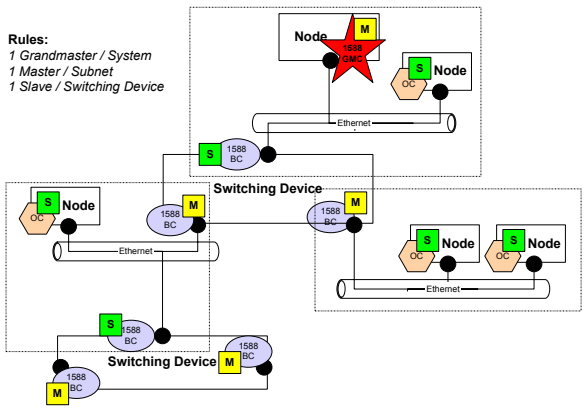

Figure 1—IEEE 1588 Configuration

IEEE 1588 defines two separate types of clocks: ordinary and boundary. Boundary clocks (BC) are employed in devices such as hubs or switches—where more than one PTP communication path (port) exists. Ordinary clocks exist in devices having a single port—e.g., normal network devices. Each BC port can act as a master or ordinary clock in its own segment.

PTP is for networks that support multicasting but keep multicasts within a subnet and where each local clock fulfills exacting requirements. The grandmaster clock (GMC) is the best clock in the system—with the best inherent stability, accuracy, resolution, etc. defined by the standard [2]. The Best Master Clock Algorithm (BMC), run by every live node, determines clock quality. Within each subnet, the BMC determines the master clock; in a single-subnet system the master is the GMC.

The GMC determines system synchronization; system clocks synchronize their subnet clocks to the system. There is only one GMC per system, and only one master clock per subnet.

Synchronization is performed as follows. All masters periodically broadcast "Sync" messages containing an estimate of the time the message will physically leave the master. The precise receipt time of these messages is noted at the slaves. The precise sending time of the message is noted at the grandmaster. All precise timing measurements are performed as close to the physical layer as possible—to eliminate the delays from the network stack and operating system—while the estimated times are calculated by the IEEE 1588 code at the Application Layer (see **Figure 2**). Following the Sync message, the master transmits a related "Follow_Up" message containing the precise sending time of the Sync message. A slave uses the transmission and reception times to calculate its offset and can initiate synchronization with the delay measurement, which is not periodic and not performed as often as the protocol synchronization. Sync messages do not propagate beyond their originating subnet.

The resolution of the system clock is the resolution of the GMC. If required, the GMC can be synchronized to an external source such as GPS.

IEEE 1588 is a highly precise system for synchronizing distributed nodes for applications such as motion control and robotics. It was designed for multicasting networks but with the popularity of Industrial Ethernet, Annex D was included for an

Ethernet implementation of PTP. Although IEEE 1588 does not alter Ethernet or make it more deterministic or reliable, it does provide a method for other protocols to do so. A highly synchronized system of distributed nodes—coupled with an application for handling resolution and controlling traffic—could deliver hard, deterministic RT over Ethernet.

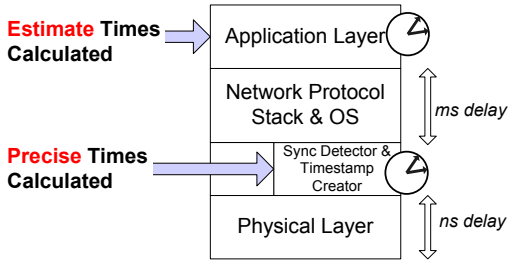

Figure 2—IEEE 1588 node timing

PROFInet

PROFInet [3] is a plant-wide fieldbus standard for distributed automation systems. It uses object-orientation and available IT standards (TCP/IP, Ethernet, XML, COM). PROFInet is also built on IEEE 802.3 and is interoperable with TCP/IP—allowing it to be implemented on existing Ethernets. It is compatible with PROFIBUS-DP.

PROFInet V1, has a response time of 10-100 ms. PROFInet-SRT (Soft Real-Time) allowed PROFInet to work with a factory automation cycle time of 5-10 ms, achieving RT solely in software. It uses TCP/IP and a dedicated software channel for RT communications. PROFInet-IRT brings a hard-RT element to the PROFInet protocols. The three PROFInet protocols allow differing degrees of RT. PROFInet for hard RT is PROFInet-IRT.

PROFInet-IRT

PROFInet IRT (Isochronous RT) was developed for systems requiring sub-microsecond synchronization, typically high-performance motion control systems. The benchmark for such a system is 1 ms cycle time, 1 µs jitter accuracy, and guaranteed determinism [4]—which IRT fulfills.

Since software introduces jitter above 1 µs, IRT (unlike SRT) is a hardware solution with highly synchronized Ethernet nodes. Using full-duplex switched Fast Ethernet, it divides the communication cycle into a standard TCP/IP open channel and a deterministic RT channel. The channel ratio is system-dependent and is chosen by the systems engineer.

Each PROFInet-IRT device has a special ASIC (Application-Specific Integrated Circuit) for handling node synchronization and cycle subdivision and incorporates an intelligent 2 or 4 port switch.

The PROFInet switch in every node is highly synchronized, contains a schedule of bus access and can deal with RT and non-RT traffic. It prioritizes RT traffic and provides full-duplex links for all ports. Contemporary switches (even cut-through) add jitter that would impact on determinism. PROFInet switches minimize jitter to where it has a negligible effect. The PROFInet communication model allows both RT and non-RT traffic to co-exist on one network without additional precautions.

By 2005, PROFInet-IRT and SRT will incorporate PROFISafe, the PROFIbus safety solution for manufacturing and processing industries.

PROFInet, of all the solutions discussed here offers the greatest determinism—and since this is built into the PROFInet-IRT device, the systems engineer is spared from the burden of configuration to guarantee RT communication.

EtherCAT

EtherCAT (Ethernet for Control Automation Technology) is the motion-control RT solution from Beckhoff. It can process 1000 I/Os in 30 µs [5], but requires full-duplex. It can use copper or fiber optic cables. EtherCAT is based on the master/slave principal and can interoperate with normal TCP/IP-based networks and other Ethernet-based solutions such as PROFInet. It also supports any Ethernet topology, including the bus.

The EtherCAT master processes RT data via dedicated hardware and software (Beckhoff currently use their PC-based TwinCAT OS and TwinCAT Y driver). In the future, further variations will be introduced that will also provide the same guarantees. The current master prioritizes EtherCAT frames over normal Ethernet traffic, which is transmitted in gaps. The master controls traffic by initiating all transmissions.

The telegrams are standard Ethernet, and the data field encapsulates the EtherCAT frame (an EtherCAT header and one or more EtherCAT commands). Each command contains a header, data and Working Counter (WC) field. Each Ethernet telegram can contain many EtherCAT commands—realizing a higher bandwidth and more efficient use of the large Ethernet data field size and header (see **Figure 3**). The standard Ethernet CRC is used to verify message correctness.

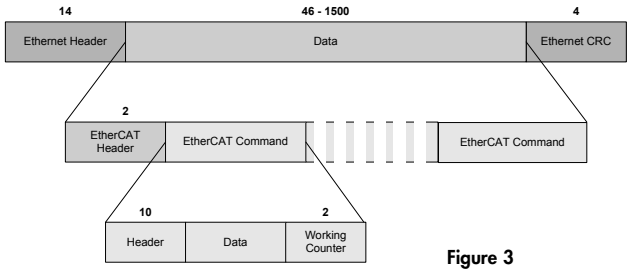

Figure 3
EtherCAT Encapsulation

The EtherCAT master fully controls its slaves. Its commands only elicit responses; slaves do not initiate transmissions. The two EtherCAT communication methods used are "Ether Type" or UDP/IP encapsulation.

The "Ether Type" uses the type field (defined in Ethernet II), which is more commonly known as the length field in IEEE 802.3. The Ether Type implementation does not use IP, thus limiting EtherCAT traffic to the originating subnet. Encapsulating commands using UDP/IP allows EtherCAT frames to traverse subnets, but has drawbacks. The UDP/IP header adds 28 (20: IP, 8: UDP) bytes to the Ethernet frame and undermines RT performance through its non-deterministic stack. EtherCAT slaves range from intelligent nodes to 2-bit I/O modules and are networked via 100Base-TX, fiber optic cable or E-bus (depending on distance requirements). E-bus is an EtherCAT physical layer for Ethernet offering a LVDS (Low Voltage Differential Signal) scheme. Slaves are hot pluggable in any topology of branches or stubs. Multiple "slave rings" can exist on a single network if connected by a switch.

EtherCAT slaves have integrated memory from 2 bits to 64 Kbytes. They appear to the Ethernet as a single device though actually comprising up to 65,535 devices. They are configured in an open-ring topology, with the Ethernet interface at the open end. Masters transmit commands to the MAC address of the first device. When the signal reaches the Ethernet/slave interface, it is converted to E-bus specifications (if E-bus is employed) and forwarded.

A slave receives a telegram, processes it (in hardware) then forwards it to the next slave on the ring. This processing delays the telegram by an order of nanoseconds. The last slave returns the completed telegram, via the ring, to the master. On the return route, each slave amplifies and regenerates the signal. Each slave has two Tx & Rx interfaces, so bi-directional communication occurs without contention.

In each EtherCAT command, the WC increments when a slave processes a command addressed to it, allowing the master to determine if each addressed slave is exchanging data, although correct data is not guaranteed.

The FMMU (Field Memory Management Unit) of each configurable slave converts a logical address to a physical one, and that information is available to the master at initialization. Thus, each slave needs a special ASIC. On telegram reception, a slave determines if it is addressed and then passes data to/from the telegram—incurring a delay of some nanoseconds. EtherCAT is also internally synchronized by a distributed clock algorithm (a simplified version of IEEE 1588) although external synchronization is achievable with IEEE 1588.

EtherCAT is a fast RT Ethernet solution and deterministic if not used with UDP/IP or intermediate switches or routers between master and slaves.

ETHERNET Powerlink (EPL)

EPL [6] is a hard-RT protocol based on Fast Ethernet. Like EtherCAT, it uses the Ethernet II Frame type field. EPL devices use standard Ethernet hardware with no special ASICs. EPL can deliver a cycle time of 200 µs with jitter under 1 µs. Its frame is encapsulated as illustrated in **Figure 4**.

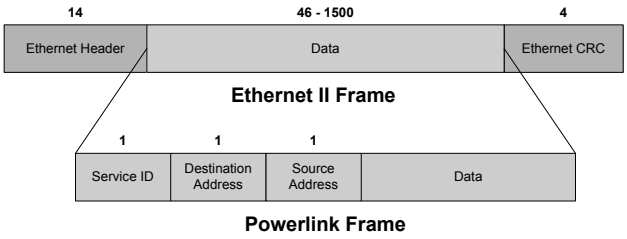

Figure 4—PowerLink Encapsulation

EPL uses cyclic communication with time-slot division and the master/slave model. One master (manager) is allowed per network. The master schedules all transmissions and is the only active station—slaves transmitting on request.

The four EPL cycle subdivisions are illustrated in **Figure 5**.

During the Start Period, the EPL master broadcasts the "Start-of-Cyclic" (SoC) frame which synchronizes the slaves. The timing of this frame provides the only time base for the network synchronization: all other frames are purely event-driven.

After transmitting the SoC frame, the Cyclic Period occurs as the manager polls each station with a "Poll Request" frame. Only then does the slave respond with a "Poll Response" frame containing data— hence, collisions are avoided. The slave broadcasts its response to all devices; thus, inter-slave communication can occur.

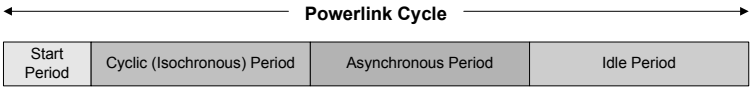

Figure 5—PowerLink Cycle

After successful polling of all slaves, the master broadcasts the "End-of-Cyclic" (EoC) frame, informing each slave that the cyclic traffic progressed correctly.

The Asynchronous Period allows non-cyclic data transfers under master control. To transmit during this period, a slave must have informed the master in its "Poll Response" during the Cyclic Period. The master builds a list of waiting slaves and employs a scheduler to guarantee that no send request will be delayed indefinitely. During the Asynchronous Period, standard IP datagrams can be transferred.

Unlike PROFInet, EPL does not employ switches to avoid collisions or to provide the network synchronization; the master controls this. EPL networks can be built with standard hubs. It is proposed that each device incorporate a hub for ease of bus implementation. Switches, although not prohibited, are not recommended for EPL because they add jitter and

reduce determinism. Since the EPL network avoids collisions via time-controlled bus access, up to 10 hubs can be cascaded (an allowable exception to the 5-4-3 Ethernet rule).

Currently, EPL devices demanding RT communication cannot co-exist on the same segment as non-RT Ethernet devices. However, EPL devices can operate as normal Ethernet devices. In Protected Mode, the RT segment must be separated from normal traffic by a bridge or router. In Open Mode, RT traffic shares the segment with normal traffic, but RT communication is compromised. In the next Powerlink version (V3), IEEE 1588 will be used to synchronize traffic across multiple RT segments—providing a more distributed EPL implementation, but true RT segments will still contain only EPL devices. Unlike PROFInet where normal Ethernet and RT devices can co-exist and not affect RT traffic, EPL must be protected from non-RT communication through bridges or routers. Unlike PROFInet or EtherCAT, which need special ASICs, EPL employs standard Ethernet hardware.

CONCLUSION

Real-Time Ethernet is a fast-growing, exciting development of the Ethernet protocol. The ability to have RT control segments running on the same network as office applications brings many new possibilities for industrial applications. With different protocols offering different levels of RT service, it is vital to understand the RT requirement of the system before choosing a solution. Sub-microsecond synchronization accuracy, with IEEE 1588, along with an RT protocol can provide an Ethernet capable of delivering hard, fast and deterministic RT for applications such as motion control, while other solutions cater for softer applications. Therefore, when choosing RT Ethernet, it is vital to consider the real-time, interoperability and flexibility requirements of your system along with all possible solutions before making a commitment.

REFERENCES

1 IEEE Std. 1588 -2002, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems."

2 Eidson, J. et al. "Synchronizing Measurement and Control Systems." Sensors, November 2002.

3 PROFInet available at http://www.profibus.com/

4 Baumann, G. "Solving the Compatible Real-Time Ethernet Conundrum," The OnLine Industrial Ethernet Book, Issue 16, September 2003.

5 Beckhoff EtherCAT available at http://www.beckhoff.com/english/default.htm?ethercat/default.htm.

6 ETHERNET Powerlink available at http://www.ethernet-powerlink.org

Look for Paula Doyle's complete article as Elective IE402 in the Curriculum of the virtual Industrial Ethernet University at **www.industrialethernetu.com**. If you wish to contact Paula, her e-mail address is: Paula.Doyle@ul.ie.

Paula Doyle is a doctoral researcher with the Circuits and Systems Research Centre at the University of Limerick in Ireland. She has a first class honors B. Engineering degree in Computer Engineering from the University of Limerick. Her research interests include embedded real-time systems, with emphasis on control networks for smart transducers with applications in the field of industrial automation. Paula is currently in the final stages of her Ph.D. research, based on the time-triggered transducer clusters in an Ethernet networking environment.