Real-time Open Source Solution for Industrial Communication Using OPC UA PubSub over TSN

Diesilva Sagaya James[1], Jayachandran Ramesh Babu[2], Melvin Francis[3], Pavithra Karuppusamy[4]
Kalycito Infotech Private Limited, Coimbatore, India.
[1]diesilva.s@kalycito.com
[2]jayachandran.r@kalycito.com
[3]melvin.f@kalycito.com
[4]pavithra.k@kalycito.com

***Abstract*— Industry 4.0 induced the OEMs to migrate towards a scalable and interoperable architecture with demand for a vendor neutral communications standard. Open Platform Communication Unified Architecture (OPC UA) is a platform independent, service-oriented modeling framework that is capable of meeting this demand from the OEMs. The OPC UA Publisher/Subscriber (PubSub) standard together with the emerging time aware Ethernet standard Time Sensitive Networking (TSN) will enable real-time networking capabilities as well. TSN is expected to be established as a standard with wide availability of network interface chipsets in two years from now. Meanwhile, this whitepaper looks at the performance possibilities and readiness of the technology architecture that is available today for OEMs to kickstart their projects developing next generation systems.**

***Keywords*— Industrie 4.0, M2M, M2C, PubSub, OPC UA, TSN**

I. Introduction

Manufacturers and users of automation components such as sensors, drives and PLCs were predominantly using analog signaling before moving to legacy fieldbuses protocols such as Modbus, Profibus, DeviceNet over physical medium such as RS485, CAN, etc for the last 40 years. In the last 5 to 10 years, the industry has seen the trend of migrating to Ethernet based communication protocols to increase bandwidth, reduce wiring complexity and reduce costs.

There are multiple Ethernet based communications protocols promoted by different PLC vendors in the market. This has created a fragmented market where device manufacturers constantly worry about cost effective and quick go-to-market plans to implement support for as many protocols as possible into their automation products. The same worry is there for end users such as machine builders or factory automation system integrators. They need frictionless flow of data from the field to the cloud which is difficult to obtain, if products from different vendors that use different standards need to talk to each other.

In such a scenario, TSN is an IEEE 802.1 standard that is fast emerging as the upgrade to the IEEE 802.3 Ethernet standard to solve the interoperability problem for manufacturers and end users alike while still promising high performance in terms of guaranteed bandwidth and timing at the layer 2 – data link. Going up to the higher layers, IEC 62541 OPC UA is fast emerging as the application data modeling standard to enable interoperable data exchange between devices from different manufacturers.

II. Full Stack Perspective

To fully exploit the potential of IoT, it must be possible for data to flow securely to wherever in the system it can add value: between edge gateway devices for control purposes, data aggregation/ingestion and local analytics, to cloud-based applications and enterprise systems for big data analytics, IT-OT and supply chain integration. Figure 1 shows the full stack perspective of end-to-end real-time industrial systems according to OSI layers.

Machine to Cloud (M2C) architectures will be defined by security requirements and Machine to Machine (M2M) architectures will be defined by performance requirements. While the focus of this paper is on OPC UA PubSub over TSN for M2M architectures, M2C is also considered when including traffic for secure Client/Server as part of next phase. This is the basis for approaching the implementation architecture from a full stack perspective to visualize all the layers that can impact the performance between the applications on different devices. Section IV of this paper contains the results of i) measuring the real-time capability of the test systems, ii) the quality of the time synchronization between systems using Precision Time Protocol (PTP) and iii) the quality of the time synchronization between the network and the systems. A revised version of PTP is standardized as IEEE 802.1AS and will be referred to as such where appropriate. The subsequent sections V and VI describe the performance of an OPC UA TSN PubSub application.

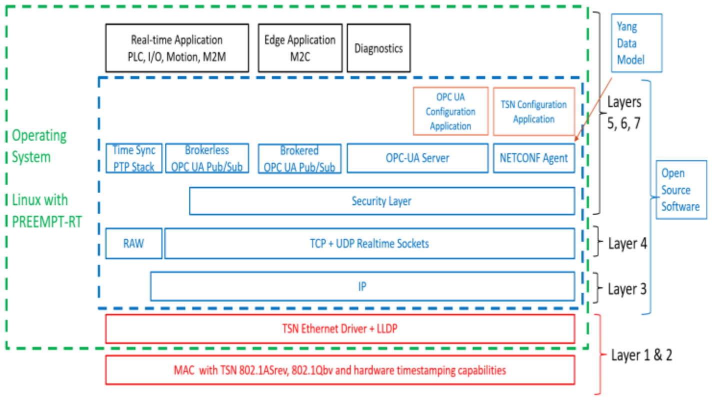
Real-time Application PLC, I/O, Motion, M2M
Edge Application M2C
Diagnostics
Yang Data Model
OPC UA Configuration Application
TSN Configuration Application
Layers 5, 6, 7
Operating System
Linux with PREEMPT-RT
Time Sync PTP Stack
Brokerless OPC UA Pub/Sub
Brokered OPC UA Pub/Sub
OPC-UA Server
NETCONF Agent
Open Source Software
Security Layer
RAW
TCP + UDP Realtime Sockets
Layer 4
IP
Layer 3
TSN Ethernet Driver + LLDP
Layer 1 & 2
MAC with TSN 802.1ASrev, 802.1Qbv and hardware timestamping capabilities

Figure 1: Full stack perspective for end-to-end real-time industrial systems

Table 1 lists the related TSN standards both released and under development for audio-video, industrial and automotive use cases. This paper is focusing on the time synchronization standard IEEE 802.1AS that is important for industrial use cases. The standards for scheduled traffic IEEE 802.1Qbv and configuration IEEE 802.1Qcc also are important topics for interoperability but are beyond the scope of this paper.

Table 1

TSN STANDARDS

Standard	Description
IEEE 802.1AS-rev (a specific profile of IEEE 1588 under development)	Timing and Synchronization for Time-Sensitive Applications
IEEE 802.1Qav	Forwarding and Queuing Enhancements for Time-Sensitive Streams
IEEE 802.1Qbu and IEEE 802.3br	Frame Preemption
IEEE 802.1Qbv	Enhancements for Scheduled Traffic
IEEE 802.1Qca	Path Control and Reservation
IEEE 802.1Qcc	Enhancements and Performance Improvements
IEEE 802.1Qci	Per-stream Filtering and Policing
IEEE 802.1CB	Seamless Redundancy
More to come as the standards evolve	...

III. TEST SETUP

A. Hardware setup

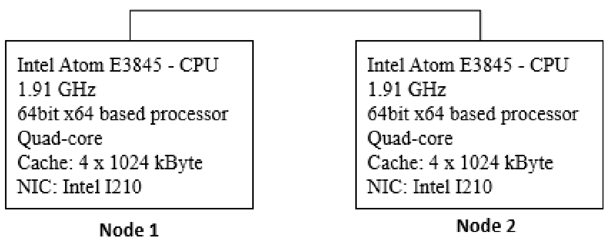

Figure 2: Hardware setup for performance measurements

The hardware setup consists of two identical Atom processor systems with I210 network interface cards connected via PCIe. Both the systems are connected peer to peer with an Ethernet cable at a link speed of 1 Gb/s.

B. Software setup

All the software packages used in the system are open source and available to download from the respective developer forums. Intel’s Time Based Scheduling (TBS) patch is used for IEEE 802.1Qbv implementation whereas IEEE 802.1Qav is already present in the I210 solution.

Table 2

SOFTWARE PACKAGE INFORMATION

Software package	Versions
Linux	Debian v9.5, Kernel 4.16.15-rt7
PTP	lptp 1.7
OPC UA stack	open62541 v0.3-rc1
Intel TBS	I210 igb driver

IV. REAL-TIME PERFORMANCE

The real-time performance of the system is determined using the *cyclictest* application. Briefly, *cyclictest* consists of a main task that initially starts a number of child tasks. The latter install cyclic alarms using a real-time capable timer at a given interval and repeatedly wait for the expiration of the timer. When a timer expires the current time is obtained and compared to the theoretically expected wake-up time. The difference that represents the inability of the system to exactly wake up in time, but a little later represents the so-called system latency. If the measurement is repeated frequently enough and under as many conditions as possible, the longest latency ever obtained may be used to describe a system's worst-case latency.

A. Worst-case latency

The longest acceptable worst-case latency of a real-time system depends on the system requirements such as cycle time and sample rate and may be as short as several tens of microseconds. Based on this observation and with the goal in mind that the results presented in this paper should be applicable to the vast majority of industrial use cases, it was requested that the real-time systems used for the tests described in this paper should have a worst-case latency of less than 40 µs.

The *cyclictest* application is run for a period of 12 hours in both the nodes without applying any system or network load. While running the *cyclictest* application, the two nodes are time synchronized as described in the next section. The observed worst-case latency of Node 1 is 36 µs and Node 2 is 34 µs.

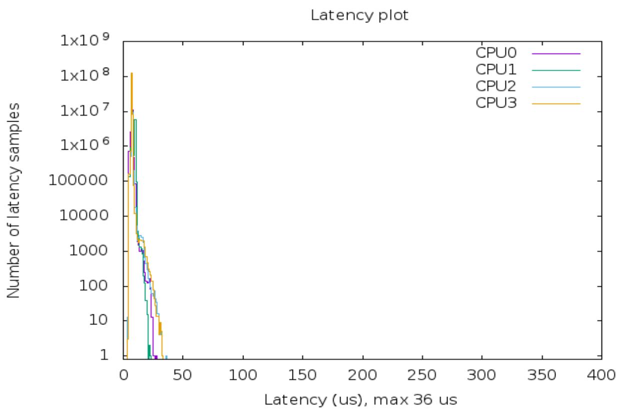

Figure 3: Worst case latency of Node 1 over 12 hours

B. PTP synchronization

Node 1 is configured to be the PTP master and Node 2 is configured to be the PTP slave. The hardware clock of the PTP hardware clock (PHC) of the PTP master serves as the clock for this network. PTP slave adjusts its PHC to that in the PTP master. This ensures that the two nodes are synchronized.

Figure 4 shows the results of the PTP accuracy measurement between the two nodes over a period of 12 hours; the observed clock offset is in the range of ±40 ns and, thus, close to the performance that is theoretically achievable.

Note: It is seen that the PTP synchronization is lost and re-established once during the test period of 12 hours.

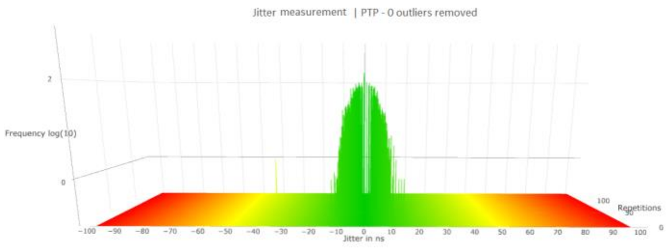

Figure 4: PTP clock offset

C. *PTP hardware clock to system clock synchronization*

The system clock is used by most user-space applications, hence the system clock needs to be synchronized between nodes on the network. This can be achieved by synchronizing the system clock with the PHC by using the *phc2sys* utility.

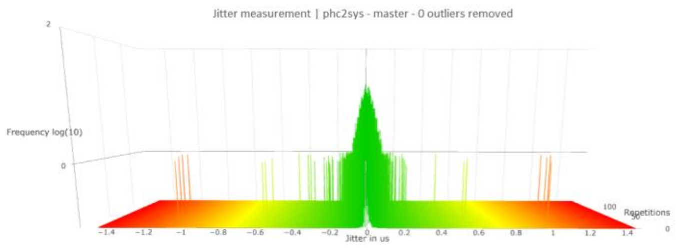

Figure 5: phc2sys accuracy on Master System (Node 1)

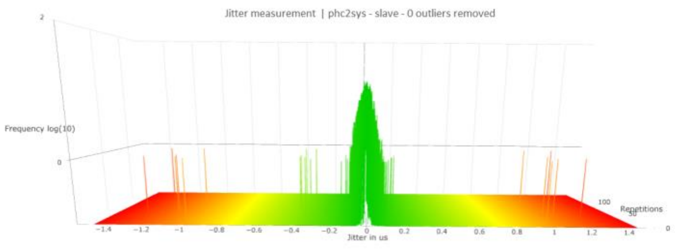

Figure 6: phc2sys accuracy on Slave System (Node 2)

Referring to Figure 5 and 6, *phc2sys* jitter values of PTP master and slave nodes are at ±1.3 µs, it is evident that the system can synchronize the system clock to hardware clock at single-digit microsecond range which ideally matches the requirements of most, if not all, industrial real-time applications.

V. PUBSUB NETWORK PERFORMANCE

The OPC UA PubSub application is configured to publish UADP packets at 100 µs cycle time. To measure the network performance of OPC UA TSN application the packet to packet jitter is calculated on the receive (Rx) side of the nodes. Figure 7 and 8 show the jitter performance of the OPC UA TSN application for 2 million samples without TBS and with TBS respectively.

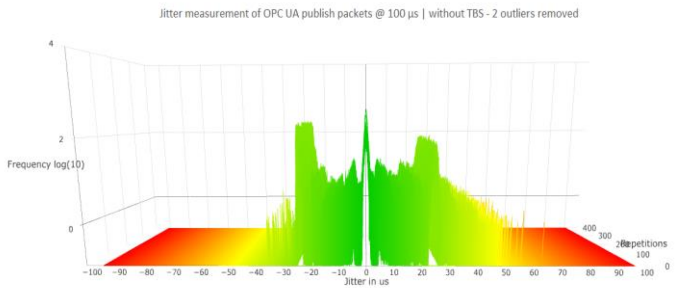

Figure 7: Jitter of OPC UA UADP traffic – without TBS

The jitter of OPC UA UADP traffic without TBS functionality is in the range of -40 µs to +138 ms. The samples which occurred beyond 100 µs are removed and are not plotted in Figure 7.

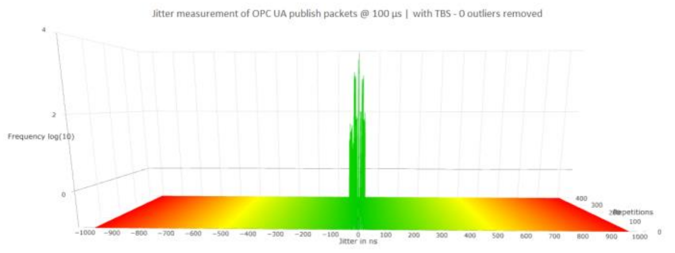

Figure 8: Jitter of OPC UA UADP traffic – with TBS

The packet to packet jitter of the OPC UA UADP traffic with TBS functionality is at ±40 ns.

From the above comparison, the performance of OPC UA PubSub from Linux user-space promises real-time guarantee with minimum jitter in nanosecond range when using TBS.

TBS ensures that the transmission (Tx) of high priority traffic is prioritized through the high priority hardware queue. It also helps to buffer packets and make sure that the packets are sent out in the configured time before their deadline (Tx times). If TBS is not enabled, the network device will not be able to distinguish between priorities of different traffics. So, no absolute guarantee for an end-to-end delivery time can be given when TBS is not enabled.

VI. PUBSUB ROUND TRIP TIME MEASUREMENTS

The previous section looked at performance of an application in meeting the deadline to place a packet on the network. This section goes further and looks at the application 'Round trip time' (RTT) measured using the OPC UA PubSub application over TSN. This is important for the communication latency from a PLC to an I/O or motion control node scan and back to the PLC.

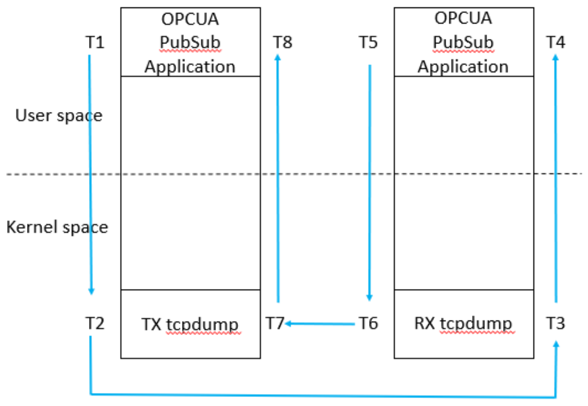

Figure 9: Trace point setup

T1 – UADP packet publish timestamp
T2 – tcpdump outgoing packet timestamp
T3 – tcpdump incoming packet timestamp
T4 – UADP packet subscribe timestamp
T5 – UADP loopback packet publish timestamp
T6 – tcpdump loopback packet outgoing timestamp
T7 – tcpdump loopback packet incoming timestamp
T8 – UADP loopback packet subscriber timestamp

The round trip time is the time taken for the application in Node 1 to encode and publish a packet to Node 2 whose application decodes the received publish packet and loops back the same to make the published data finally available to the application on Node 1. From Figure 9, T8-T1 provide us the absolute round trip time of the application.

Absolute round trip time = 2 * (Tx latency + Rx latency)

Tx latency – Time taken to encode and send data from one node to reach the network interface of another node (includes wire delay).

Rx latency – Time taken to receive and decode the subscribed packets from the network interface.

A. Round trip time without TBS

The maximum RTT measured for half a million samples is 900 ms.

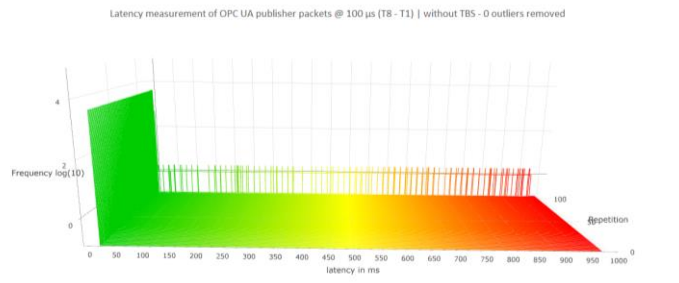

Figure 10: Round trip time without TBS

B. Round trip time with TBS

The maximum RTT measured for 2 million samples is 447 µs.

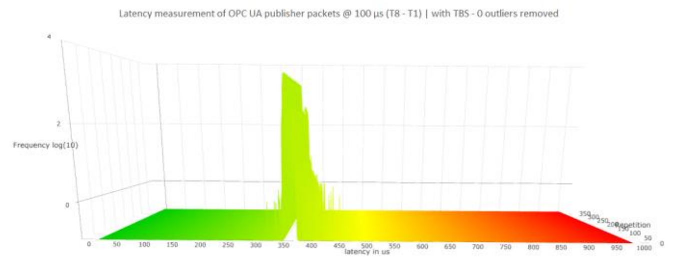

Figure 11: Round trip time with TBS

VII. Network Errors

During round trip time measurements, it was observed that in the measurements taken at T8, certain number of packets were dropped i.e. packets were reaching the kernel but did not arrive in user-space. The underlying mechanism of this network error remains to be investigated and fixed.

VIII. Summary

From the series of measurements done in the fully integrated stack it is inferred that the open source OPC UA TSN PubSub solution is realizable with 100 µs cycle time performance and overall network jitter at ±40 ns. Also, end-to-end round trip time is possible at 500 µs which matches the requirements of a large part of industrial use cases.

Acknowledgment

The authors would like to thank the industry consortium supporting the joint project for developing the PubSub extension for the open62541 OPC UA SDK. Special thanks go to Carsten Emde of the Open Source Automation Development Lab (OSADL) for assistance to setup the real-time systems and to Julius Pfrommer of Fraunhofer IOSB for providing continuous support on the open62541 stack development and maintenance.

References

[1] "Hardware Assisted IEEE 1588 Clock Synchronization Under Linux" By Bálint Ferencz, Budapest University of Technology and Economics

[2] "The Road Towards a Linux TSN Infrastructure", Jesus Sanchez-Palencia

[3] "OPC UA TSN A new Solution for Industrial Communication" by D. Brucknery, R. Blair, M-P. Stanica, A. Ademajz, W. Skeffingtonx, D. Kutscher, S. Schriegel, R. Wilmes, K. Wachswender, L. Leurs, M. Seewald, R. Hummen, E-C. Liux and S. Ravikumar