XpressEth: Concise and Efficient Converged Real-Time Ethernet

Kun Qian, Fengyuan Ren, Danfeng Shan, Wenxue Cheng, Bo Wang
Tsinghua National Laboratory for Information Science and Technology(TNList), Beijing, China
Department of Computer Science and Technology, Tsinghua University, Beijing, China
{qk15,sdf13, chengwx14,wang-b14}@mails.tsinghua.edu.cn, renfy@tsinghua.edu.cn

***Abstract*— Owing to Ethernet's low cost, high bandwidth and architecture openness, much attention has been paid to develop converged Ethernet to support both time-critical services and conventional communication services on a unified network infrastructure. The greatest challenge here is providing low and deterministic latency for time-critical packets. Recently, the IEEE time sensitive networking task group is launched to address it. However, their framework is complex and unsuitable for commodity switch architecture. In this paper, we propose a concise and efficient converged real-time Ethernet framework called XpressEth, which leverages Dual Preemption mechanism to minimize the delay of time-critical packets, and employs a lightweight Slot Assignment Scheduler to minimize the conflicts among time-critical packets at sources. XpressEth cuts off great burden from both forwarding and scheduling. The simulation results verify that XpressEth can provide ultra-low and deterministic latency for time-critical packets (1.024μ s per hop and zero jitter in 1Gbps network), which is $13\times$ better than time sensitive networking solution, and the side-effect on conventional communication traffic is negligible.**

I. Introduction

With the fast development of Industrial Internet and Internet of Things, many networked systems (such as factory process control, medical equipment, avionics, and transportation) involve both industrial real-time services (e.g. state monitoring and industrial control) and conventional communication services (e.g. human machine interface, camera and infotainment). These two categories of services have different requirements and conventionally run in different kinds of networks. Real-time services require strict end-to-end message delay (a few milliseconds) and jitter (a few microseconds) [15, 17], which are ordinarily supported by fieldbus. Conventional communication services, on the other hand, usually need high bandwidth, which are generally run on Ethernet. These two kinds of networks are incompatible. Hence, both fieldbus and Ethernet need to be deployed along the entire system and specific gateways are required to exchange data between different networks. This heterogeneous structure makes the entire networked system complex and expensive. Therefore, there is a trend to converge two kinds of traffic in a unified network.

Due to limited bandwidth (1∼100Mb/s) [10, 12, 20] and complex structure [16, 22], fieldbus is an unsatisfied candidate for this convergence. On the other hand, because of low cost, high bandwidth and open infrastructure, Ethernet attracts much manufacturers' attention to merging real-time services into it. However, simply adopting conventional Ethernet cannot meet the requirements of real-time services, because standard Ethernet allows data exchange at any time; in this circumstance, the time-critical traffic needs to compete for the shared resource with the non-time-critical traffic which may appear at any time. As a result, it may add a great variance to end-to-end delay of time-critical traffic. Although some QoS mechanisms defined in IEEE 802.1Q [6] (such as priority queue) can be adopted, it still has a long way to meet the requirements of real-time services.

In order to enhance Ethernet to provide the deterministic end-to-end delay, many modified Ethernet solutions have been proposed (e.g. SERCOS [7], RT-Ethernet [13], Profinet [14] and FTT-Ethernet [19]). The basic idea is reserving time slice periodically for time-critical traffic. These solutions restrict the sending mode awkwardly and then need all end hosts to modify the MAC layer, but just can achieve coarse-grained latency bound. In 2012, IEEE 802.1 set up a new task group to develop Time Sensitive Networking (TSN) [9]. The technical goal is to guarantee the transmission delay for time-critical scheduled traffic (see details in Section II). However, the frame preemption enhancement proposed in TSN framework implicitly assumes that all switches are output-queued, which is an unpopular architecture owing to high implementation cost. Most commodity switches today adopt the Combined Input and Output Queue (CIOQ) architecture [21]. Without considering the forwarding competition in input buffer, the performance of frame preemption is far from ideal in the CIOQ architecture. Furthermore the scheduling solution requires all switches store tedious control table and cooperating accurately with global synchronized timer, which is complex and error-prone, and hence it presents great challenges to actual deployments.

In this paper, we propose XpressEth, a concise and efficient converged real-time Ethernet framework, to support both industrial control traffic and conventional communication traffic in converged Ethernet. XpressEth mainly comprises of two components: (1) *Dual Preemption*, a concise switching enhancement, which employs preemption mechanism at both ingress and egress to minimize the influence from non-time-critical packets, and then provides ultra-low and deterministic delay for time-critical packets; (2) *Slot Assignment Scheduler*, which is designed to guarantee zero conflict among time-critical scheduled packets. The scheduling solutions are only

978-1-5386-2704-4/17/$31.00

TABLE I: The characteristics of different traffic in real-time Ethernet.

Category	Pattern	Requirement
Scheduled	Time-triggered	Low and deterministic latency
Stream-reservation	Fixed bandwidth	Reserved bandwidth
Best-effort	—	—

deployed on source nodes, which cuts off scheduling-related burden on switches and makes the entire framework robust. Leveraging these two components, XpressEth can meet the requirements of time-critical traffic without deteriorating the performance of non-time-critical services. We build a byte-level simulator to evaluate the performance of entire XpressEth framework with realistic traffic traces. The results show that XpressEth reduces 92.3% end-to-end latency for time-critical packets compared with TSN framework, and induces negligible side-effect to non-time-critical services.

The rest of paper is organized as follows: Section II introduces the framework of TSN and motivation. Section III presents our basic idea and elaborates on the framework of XpressEth. Section IV evaluates the performance of XpressEth and Section V concludes this paper.

II. Background and Motivation

A. TSN Framework

Time sensitive networking task group is launched in 2012 for enabling the support of time-critical streaming service in Ethernet. As listed in Table I, TSN divides the traffic in converged real-time Ethernet into three categories: scheduled traffic, stream-reservation traffic and best-effort traffic. The scheduled traffic is time-critical (TC) and requires low end-to-end latency (a few milliseconds) and jitter (a few microseconds) [15, 17], and all flows in scheduled traffic (hereinafter referred to as scheduled flow) send packets according to a predefined cycle. According to [13], the length of all TC packets is the Ethernet minimum transmission unit (64B). Flows in stream-reservation traffic requires end-to-end reserved bandwidth.

In TSN framework, a central scheduler arranges each scheduled packet in advance. Since scheduled flows send packets in predefined cycles, the scheduler can arrange the exact time when one scheduled packet enters each switch. At each output port, the scheduler sets a guard band time slot to drain ongoing packet and forbid the transmitting of other non-time-critical (stream-reservation and best-effort) packets exactly before the upcoming of each scheduled packet. So the scheduled packet can be transmitted as fast as possible. By employing IEEE 802.1AS [1], all TC end nodes and switches maintain the global high-accuracy synchronized timer to execute this time-based scheduling.

Figure 1 shows the architecture of switch complying with TSN. The priority queues are defined in IEEE 802.1Q to provide different services for different traffic categories, and TSN framework retains the credit based shaper defined by

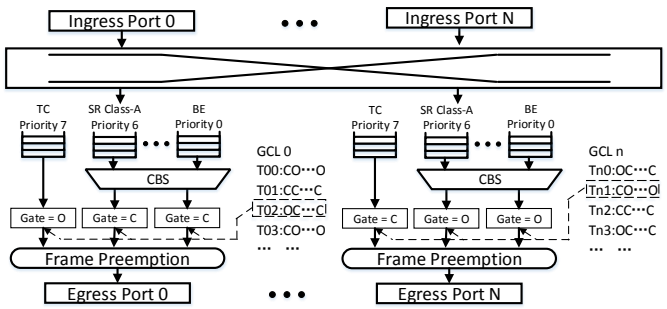


Fig. 1: TSN switch architecture. In GCL, 'C' means closed and 'O' means opened.

Audio/Video Bridge task group. After arranging all scheduled flows, the scheduler generates a gate control list (GCL) for each port in each switch. The GCL is a table recording the port operations (opened or closed) and the duration of corresponding operations [3]. In each output port, the transmission of different queues is controlled according to GCL. For making sure draining ongoing non-time-critical transmission (maximum transmission unit in the worst case) before the upcoming of TC packet, the size of the guard band is very large, which likely causes bandwidth wastage. Frame preemption [2] is employed in the output port to minimize this side-effect by splitting the ongoing non-time-critical packet. Frame preemption suspends the ongoing non-time-critical packet transmission and allows TC packet to be transmitted, therefore it greatly minimizes the delay suffered by TC packets. The transmission of preempted packet will be resumed after the finish of TC packet transmission. In addition, frame preemption keeps the fragment format compatible with Ethernet frame format. By employing frame preemption, the guard band just needs to wait the transmission of 64B (fragment cannot be less than 64B otherwise it will be dropped in the next switch).

B. Understanding and Motivation

The advantages of TSN framework are: (1) guaranteeing theoretically minimum delay and zero jitter for scheduled packet, and (2) achieving low side-effect to the non-time-critical traffic. However, there are some flaws in it:

- This framework is error-prone. One scheduled packet is assigned with an elaborate time slot, and all gate operations are controlled by the pre-calculated GCL. If a subtle mistake occurs, the further executions are permanently disordered.
- The entire framework is complex. The scheduler needs to arrange the operations of all switches and end nodes.
- This framework cannot achieve optimal performance in the commodity switch architecture. However, most COTS switches use the CIOQ architecture [21]; in which, the forwarding competition in the input buffer will introduce great delay variance, which is not considered by TSN.

Therefore, TSN solution is not perfect for the converged real-time Ethernet. This motivates us to design a concise and

efficient framework to support all kinds of traffic in the unified Ethernet.

III. XpressEth Framework

A. Overview of XpressEth

We propose a concise and efficient converged real-time Ethernet framework called XpressEth, which comprises of two components: (1) Dual Preemption and (2) Slot Assignment Scheduler (SAS). The first focuses on fast forwarding each TC packet and the second concentrates on the scheduling of all scheduled traffic. Specifically, Dual Preemption is an enhancement in the switch, which leverages the preemption on both ingress and egress to eliminate all possible interferences from non-time-critical packets. And with our optimizations, it can provide ultra-low latency jitter for every single TC packet. SAS guarantees zero conflict among all scheduled flows by scheduling the starting slots at sources. By leveraging some enhancements, it can also tolerate the occurrence of interspersing traffic. The combination of these two components can meet the requirements of all TC traffic. Furthermore, XpressEth has some special features for converged real-time Ethernet:

1) XpressEth keeps the entire network and scheduler concise. Unlike TSN employs complicated GCLs on each switch, XpressEth gets rid off the scheduling burden on switches. And SAS only schedules the start time of each scheduled flow at sources, which greatly reduces scheduling complexity and improves the robustness.
2) Rather than the frame preemption proposed in TSN, which only applies to output buffer switch, the Dual Preemption proposed in XpressEth can minimize the interference from non-time-critical traffic in the CIOQ switch architecture, which is more widely adopted in COTS switches.
3) XpressEth has no assumption or restriction on the non-time-critical services. It keeps all Ethernet features (e.g. plug and play) unchanged for non-time-critical ends. In addition, those enhanced mechanisms (e.g. credit based shaper) for supporting non-time-critical traffic can be seamlessly integrated into XpressEth.
4) Owing to the suspend-and-resume mechanism of preemption, the convergence of TC traffic has negligible side-effect on the link utilization.

Next, we present the details of these two components.

B. Dual Preemption

1) Design: For eliminating the interference of non-time-critical packets, a general switch forwarding enhancement should be employed to safeguard low and deterministic delay for TC packet. Preemption is a good methodology. However, the frame preemption in TSN is only suitable for the output port buffer switch. As shown in Figure 2, we propose Dual Preemption to provide ultra-low and deterministic delay for TC packets, which can apply to the widely used CIOQ switch.

The large variance of input buffer delay is the major reason why the frame preemption is inadequate for the CIOQ switch.

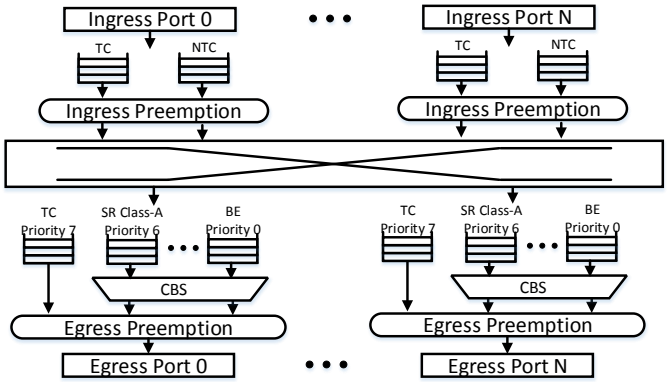


Fig. 2: XpressEth switch architecture.

We employ the first preemption between the input buffer and the crossbar, which is called ingress preemption, to eliminate the unwanted delay in the input buffer. Another preemption in the output port is called egress preemption, which does the same job as frame preemption and thus is not described again in this part.

The operation of ingress preemption is similar to the frame preemption. When a new TC packet enters into the ingress buffer, after the necessary pre-processing over received TC packet, the TC queue triggers a preemption signal to the crossbar controller and the controller does the preemption operations: (1) If the path in switch fabric needed by the TC packet is in use and the length of current fragment which has been sent is less than minimum transmission unit, it waits until the length achieves minimum transmission unit. (2) Preempts the current non-time-critical transmission. Then the TC packet can pass through the crossbar. (3) After finishing TC packet transmission, the controller resumes the transmission of preempted non-time-critical packet. This ingress preemption can greatly decrease the unwanted interference from the non-time-critical packet in the input buffer. Combining the ingress preemption and the egress preemption, Dual Preemption can greatly reduce the TC packet latency in commodity switch architecture.

2) Optimization: Due to the necessary waiting for non-time-critical fragments, each preemption introduces $\frac{m}{B}$ latency in the worst case, where m represents the minimum transmission unit. In fact, when a TC packet enters into the switch, triggering preemptions one after another is unnecessary. We just need one preemption signal to trigger both the ingress and egress preemption. When the TC packet is fully received and checked in the input queue, it triggers one preemption signal to the crossbar controller and the corresponding output port. Then two preemptions start at the same time and the entire delay jitter is decreased to $\frac{m}{B}$. Furthermore, in order to achieve deterministic delay, we introduce the Talker Scheduled Traffic Support (TSTS) [8]. Its basic idea is simple: when a TC packet is fully received in the input buffer, it always waits the worst case preemption delay. Although the worst case transmission delay does not decrease, the delay jitter will be theoretically

zero, which is critical for scheduled traffic.

3) Delay analysis: For lack of space, we employ a brief worst case delay analysis to deduce the performance of Dual Preemption theoretically. We assume that scheduler already makes all TC packets conflict free. First, for a switch, it needs $\frac{m}{B}$ to fully receive a TC packet. With only strict priority, in worst case, TC packet needs to wait the transmission of maximum transmission unit (M) in both ingress port and egress port. As a result, the worst case delay of TC packet in strict priority T_{SP} is shown in Equ. 1.

$$T_{SP} = \frac{m}{B} + \frac{2M}{B} + t_o \tag{1}$$

where t_o represents the time for other processing (e.g. CRC check and output port look up). t_o is usually a small value.

With frame preemption enhancement, the waiting time in output buffer can be reduced to $\frac{m}{B}+t_p$, where t_p represents the delay introduced by frame preemption (several clock cycles). So the worst case delay of TC packet in frame preemption T_{FP} is shown in Equ. 2.

$$T_{FP} = \frac{2m}{B} + \frac{M}{B} + t_p + t_o \tag{2}$$

With Dual Preemption, the waiting time in input buffer can be reduced to $\frac{m}{B}+t_p$, so the delay time will be $\frac{3m}{B}+2t_p+t_o$. And with optimization to trigger two preemptions at the same time, we merge $\frac{2m}{B}+2t_p$ to only $\frac{m}{B}+t_p$. Finally, the delay of TC packet in frame preemption T_{DP} is shown in Equ. 3.

$$T_{DP} = \frac{2m}{B} + t_p + t_o \tag{3}$$

Noticing that in real Ethernet M (1518B) is about 23× larger than m (64B), so our Dual Preemption mechanism decreases about 92.3% delay suffered by TC packet compared with frame preemption.

C. Slot Assignment Scheduler

The scheduler applied in TSN solution is complicated and error-prone, which involves both end nodes and switches. Generally, TC traffic takes a small proportion of the whole traffic [3] (0.07% in [23] and 0.05% in [5]), hence scheduling TC traffic can be done conveniently. In XpressEth, the lightweight SAS is proposed to guarantee zero conflict among scheduled flows.

1) Slot Assignment Scheduler: Thanks to the deterministic delay of one TC packet guaranteed by Dual Preemption, once one scheduled packet leaves end host, its time slots in any switches are determined. So the scheduler just needs to set the sending time of each scheduled packet to guarantee zero conflict. Noticing that the scheduled flow sends packets periodically, actually we just need to set the start time of each scheduled flow, and following packets are naturally conflict-free. With this basic idea, the SAS schedules the start time of each scheduled flow and makes sure the time slots taken by all scheduled flows are conflict-free. Specifically, assuming there are N scheduled flows, and each flow F_i has its own sending period T_i and forwarding path P_i. $T = LCM\{T_i | i = 1, 2, ..., N\}$, where LCM denotes the lowest common multiple. Let T_{DP} represent the time slot of a TC packet in single switch by employing Dual Preemption. There are total $\frac{T}{T_{DP}}$ time slots in T. As long as we can make sure that no one time slot in T is taken by more than one scheduled flow in any links, there is no conflict among all scheduled flows. Any specific scheduling algorithms can be used in SAS. Our SAS decides the sending time of TC packet, and let Dual Preemption in the switch to guarantee the deterministic delay. So the entire system is robust to small operation errors. If a TC packet is delayed by some reason, it does not make any influence on the succeeding scheduling decisions.

2) Random Assignment Algorithm: In this part, we propose a lightweight slot assignment algorithm: Random Assignment Algorithm for SAS. The main purpose is to exemplify that the scheduling employed by XpressEth can be done in relatively low cost. At the beginning, all scheduled flows are assigned in the unscheduled flow set $\boldsymbol{U}$. For each scheduled flow F_i in $\boldsymbol{U}$, Random Assignment Algorithm randomly generates an integer $t_i \in [1, \frac{T_i}{T_{DP}}]$ as the starting time slot. After generating the starting time for all scheduled flows, for each scheduled flow F_i, Random Assignment Algorithm checks along the path P_i to verify the feasibility of this assignment. The checking process is as follows: for each edge E_j on P_i, examining the slot sequence, if all slots in this sequence are not taken by other flows, then F_i is conflict free on E_j. If F_i does not conflict with any other scheduled flows on all edges in P_i, the scheduling of this flow is completed and F_i will be removed from $\boldsymbol{U}$. Otherwise, F_i remains in $\boldsymbol{U}$. Then Random Assignment Algorithm simply repeats the random generating and checking process for unscheduled flows until $\boldsymbol{U}$ is empty, which means all starting time slots are legal. Due to the fact that scheduled traffic shares small part of the entire bandwidth resources, Random Assignment Algorithm is quite efficient.

3) Time Complexity Analysis: In this part, we analyze the time complexity of the Random Assignment Algorithm. N_i represents the number of unscheduled flows after $(i-1)^{th}$ iteration ($i \geq 1$ and $N_1 = N$). The generating of single time slot takes $O(1)$, and the time complexity of the conflict free checking of one time slot is $O(E)$, where E represents the number of links. So the time complexity of i^{th} iteration is $O(N_i E)$. The time complexity of entire Random Assignment Algorithm is $O(\sum_{i=1}^{n} N_i E)$, where n is the number of iterations. Since the entire bandwidth share of scheduled traffic is small, the conflict of two scheduled flows is infrequent, which means $N_{i+1} << N_i$. Thus according to master theorem, the time complexity of mere random assignment procedure is $O(NE)$, and the incremental update complexity is $O(E)$.

IV. Evaluation

In this section, we conduct simulations to verify that XpressEth achieves our goals and to evaluate its performance under actual real-time system traffic patterns. First we evaluate the performance of two main components (Dual Preemption and

Slot Assignment Scheduler), respectively. Then we leverage a comprehensive simulation with real industry workloads to verify the overall XpressEth performance.

Since Dual Preemption processes packets at a byte-level granularity, commonly used packet-level simulators (e.g. ns-2, ns-3 and OMNET++) are inappropriate. Therefore, we built our XpressEth simulator based on the CIOQ Switch Simulator [4]. CIOQ Switch Simulator is a slot-level simulator for a single CIOQ switch. We let the clock run at 125 MHz, in hence 8 bits can be processed in each slot, namely it is a byte-level switch simulator. The Dual Preemption is add into the CIOQ switch simulator and we extend it to simulate the actual network topology with multiple switches.

A. Performance of Dual Preemption

This subsection focuses on evaluating the performance of Dual Preemption. We use a 24-port 1Gbps switch with combined input and output buffer. The size of both input buffer and egress buffer on each port is 500KB.

1) Guarantee deterministic latency: In this simulation, the scheduled traffic occupies 0.1% bandwidth (it is a common setting in the conventional real-time system [5, 23]). The source ports of half of the scheduled packets are uniformly distributed from 1 to 23, and the destination ports are 24, which simulates the pattern that real-time devices send data to the controller. Another half of the scheduled packets are assigned with reversed source-destination pairs, which simulates that instructions are sent from the controller to different real-time devices. All scheduled packets are arranged to be zero-conflict. The non-time-critical packets are generated with random source and destination ports. The scheduled packet length is 64B and the non-time-critical packet length is 1518B. With different non-time-critical traffic input intensities, we first compare the scheduled packet latency in Dual Preemption with the strict priority, native frame preemption and TSN solution. For each input port, the bandwidth taken by non-time-critical traffic increases from 10% to 99%.

As shown in Figure 3(a), the performance of Dual Preemption is extremely better than all other solutions and achieves 1.024μs delay for TC packet regardless of the variance background traffic. In comparison, although TSN can provide deterministic latency, the one hop latency itself (12.656μs) is relatively high for TC services. This simulation verifies the correctness of our modeling analysis and shows that the TSN solution is not an ideal framework for COTS Ethernet switch.

2) Minimize bandwidth side-effect: Another important goal we want to achieve is maximizing bandwidth utilization. Figure 3(b) shows the wasted bandwidth proportion in different proportion of scheduled traffic (varies from 0.01% to 10%). In all four mechanisms, Strict Priority is a baseline since it actually introduces zero side-effect to link utilization. When scheduled traffic is less than 1% (which is the common case in the practical industrial scenario), the side-effect introduced by Dual Preemption is negligible.

TABLE II: Simulation traffic pattern setting

Service	Bandwidth[MB/s]	Class
Control	736×10^{-4}	TC
DAC	25	SR-Class A
A/V	48	SR-Class B
TV	20	SR-Class B

B. Performance of SAS

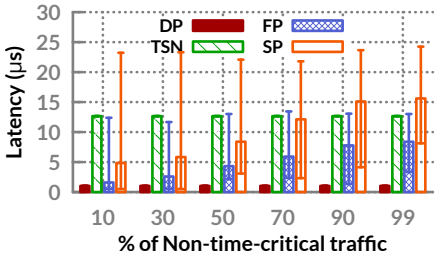
In this part, we evaluate the performance of SAS. As shown in Figure 4, we take the widely used double-star topology [11, 18] with $m = 5$ and $n = 10$, which is a large scale topology in the real industry scenario [11]. The number of controllers connected to the network is varied from 1 to 4. The sending cycles of scheduled flows are uniformly distributed from 100μs to 1ms with 100μs granularity. The entire bandwidth taken by scheduled traffic varies from 0.01% to 10%. SAS is used to schedule the start time of all scheduled flows. After our scheduling, no conflict exists in any evaluation settings. The calculation time is shown in Figure 5. The computation time of SAS is linearly correlation with the input scale, which coincides with our complexity analysis. SAS needs about 2.3ms to schedule 10% scheduled flows, which is fast and efficient.

C. Comprehensive simulation

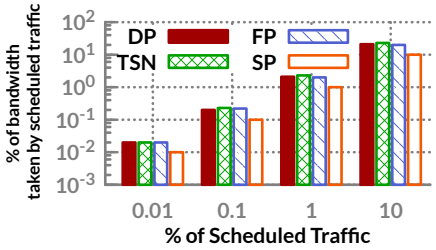
In this simulation, the comprehensive performance of XpressEth is evaluated. The double-star topology is used, with 1 controller, 2 core switches, 4 edge switches and 24 end nodes. And the traffic pattern is derived from the real configurations and traces of the communication in a BMW series car [23]. The characteristics of the traffic pattern are listed in Table II. DAC is the abbreviation of Digital Assistance Camera, and A/V represents the Audio/Video service. The sending cycles of TC traffic vary from 5ms to 1s. The credit based shaper is used for non-time-critical traffic. The driver assistance camera is allocated to SR-Class A and other non-time-critical traffic are allocated to SR-Class B. We compare the performance of XpressEth, TSN and Strict Priority. SAS is used to schedule all scheduled flows. And the GCLs in TSN solution is generated according to the scheduling results of SAS. Figure 6 shows the end-to-end latency of both scheduled and interspersing packets. XpressEth can guarantee ultra-low and deterministic delay for all TC packets with multiple hops. In comparison, 13$\times$ larger delay for scheduled traffic is achieved with TSN framework, and the delay of interspersing traffic is even worse and unsteady. Figure 7 shows the achieved bandwidth of different non-time-critical services, respectively. All three solutions can meet the bandwidth requirement of non-time-critical traffic.

V. Conclusion

In this paper, we propose XpressEth, a framework of concise and efficient converged real-time Ethernet, to support both TC and non-time-critical traffic on a unified network. The Dual



(a) TC packet latency



(b) Bandwidth side-effect

Fig. 3: Dual Preemption performance analysis, where DP, FP and SP represent Dual Preemption, Frame Preemption and Strict Priority, respectively.

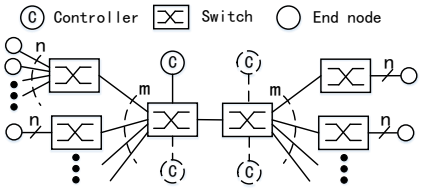


Fig. 4: Double star (m, n) with 2 core switches, $2m$ edge switches and $2mn$ end nodes.

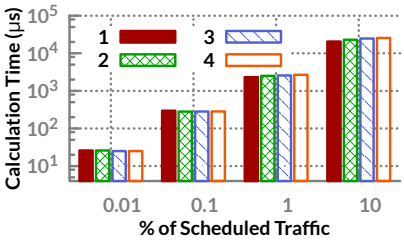


Fig. 5: The performance of SAS.

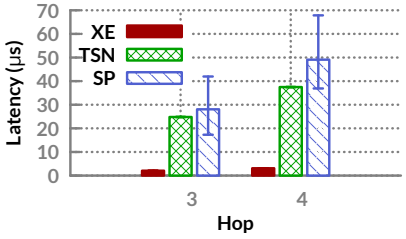


Fig. 6: End-to-end delay of TC packets.

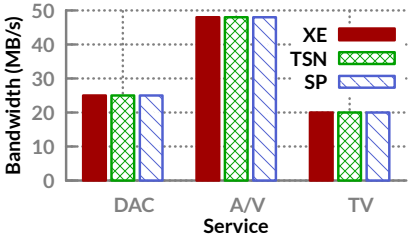


Fig. 7: Achieved bandwidth of services.

Preemption is devised to forward TC packets fast and determinately in the commodity switch architecture. The lightweight Slot Assignment Scheduler is designed to schedule all scheduled flows conflict-free. With both analysis and simulations, we verify the efficiency and performance of XpressEth, which meets all requirements of TC traffic and introduces negligible side-effect to non-time-critical services.

VI. Acknowledgments

The authors gratefully acknowledge the anonymous reviewers for their constructive comments. This work is supported in part by 863 Plan under Grant No. 2015AA020101,and Suzhou-Tsinghua Special Project for Leading Innovation.

References

[1] 802.1as-rev - timing and synchronization for time-sensitive applications. http://www.ieee802.org/1/pages/802.1AS-rev.html.

[2] 802.1qbu - frame preemption. http://www.ieee802.org/1/pages/802.1bu.html.

[3] 802.1qbv - enhancements for scheduled traffic. http://www.ieee802.org/1/pages/802.1bv.html.

[4] Cioq switch simulator homepage. http://www.cs.technion.ac.il/Courses/Computer-Networks-Lab/projects/spring2004/cioq1/CIOQ%20site/.

[5] Current state of ieee 802.1 time-sensitive networking task group. http://www.ieee802.org/1/files/public/docs2014/tsn-nfinn-IEEE-TSN-Status-for-IETF-1114-v02.pdf.

[6] Ieee 802.1q media access control (mac) bridges and virtual bridged local area networks. http://www.ieee802.org/1/files/private/q-edition-drafts/2012%20edition/IEEE802-1Q-2012-Edition.pdf.

[7] Sercos. http://www.sercos.com/.

[8] Talker scheduled traffic support for ultra low latency. http://www.ieee802.org/1/files/public/docs2013/new-tsn-specht-talker-scheduled-traffic-support-20130318.pdf.

[9] Time-sensitive networking task group. http://www.ieee802.org/1/pages/tsn.html.

[10] G. Carvajal, C. W. Wu, and S. Fischmeister. Evaluation of communication architectures for switched real-time ethernet. *IEEE Transactions on Computers*, 63(1):218–229, Jan 2014.

[11] S. S. Craciunas and R. S. Oliver. Smt-based task-and network-level static schedule generation for time-triggered networked systems. In *Proceedings of the 22nd International Conference on Real-Time Networks and Systems*, page 45. ACM, 2014.

[12] J. D. Decotignie. Ethernet-based real-time and industrial communications. *Proceedings of the IEEE*, 93(6):1102–1117, June 2005.

[13] R.-T. Ethernet. Ethernet control automation technology (ethercat): Proposal for a publicly available specification for real-time ethernet, 2004.

[14] J. Feld. Profinet-scalable factory communication for all applications. In *Factory Communication Systems, 2004. Proceedings. 2004 IEEE International Workshop on*, pages 33–38. IEEE, 2004.

[15] M. Felser. Real-time ethernet–industry prospective. *Proceedings of the IEEE*, 93(6):1118–1129, 2005.

[16] M. Felser and T. Sauter. The fieldbus war: history or short break between battles? In *Factory Communication Systems, 2002. 4th IEEE International Workshop on*, pages 73–80, 2002.

[17] K. H. *Real-time systems: design principles for distributed embedded applications*. Springer Science & Business Media, 2011.

[18] H.-T. Lim, K. Weckemann, and D. Herrscher. Performance study of an in-car switched ethernet network without prioritization. In *International Workshop on Communication Technologies for Vehicles*, pages 165–175. Springer, 2011.

[19] P. Pedreiras, L. Almeida, and P. Gai. The ftt-ethernet protocol: Merging flexibility,timeliness and efficiency. In *Proceedings of the 14th Euromicro Conference on Real-Time Systems*, ECRTS '02, pages 152–, Washington, DC, USA, 2002. IEEE Computer Society.

[20] T. Sauter. Integration aspects in automation - a technology survey. In *2005 IEEE Conference on Emerging Technologies and Factory Automation*, volume 2, pages 9 pp.–263, Sept 2005.

[21] R. Seifert and J. Edwards. *The All-New Switch Book: The Complete Guide to LAN Switching Technology*. John Wiley & Sons, 2008.

[22] T. Steinbach, F. Korf, and T. C. Schmidt. Real-time ethernet for automotive applications: A solution for future in-car networks. In *2011 IEEE International Conference on Consumer Electronics-Berlin (ICCE-Berlin)*, pages 216–220. IEEE, 2011.

[23] T. Steinbach, H. T. Lim, F. Korf, T. C. Schmidt, D. Herrscher, and A. Wolisz. Tomorrow's in-car interconnect? a competitive evaluation of ieee 802.1 avb and time-triggered ethernet (as6802). In *Vehicular Technology Conference (VTC Fall), 2012 IEEE*, pages 1–5, Sept 2012.