Analysis of Converged Network Traffic Using Time Sensitive Networking (TSN)

Rick Blair
Principal Network System Architect
Schneider Electric

Presented at the ODVA
2018 Industry Conference & 18th Annual Meeting
October 10, 2018
Stone Mountain, Georgia, USA

Abstract

In modern industrial control architectures, Ethernet has become ubiquitous as a means of communicating between devices. In today's Ethernet implementations, network hierarchy, non-standard solutions segmentation and traffic restricting techniques are often used to manage real-time network traffic flows on Ethernet networks. Time Sensitive Networking (TSN) is a set of released and emerging IEEE standards that promises convergence of these networks, thus flattening the hierarchal architectures, while retaining the requirements of the varied traffic flows.

This paper will examine several real-time industrial traffic types, such as those used for motion control, remote I/O and events and describe various TSN mechanisms that could be used to allow their operation in a converged network. The paper focuses on the interaction between scheduled mechanism described in IEEE Qbv and Quality of Service (QoS). Key traffic parameters, such as latency and jitter, will be calculated using the various TSN mechanisms for a particular traffic type. Effects of converging several traffic types on the same network will also be considered.

Keywords

Time Sensitive Networking (TSN), latency, Quality of Service (QoS), convergence, interfence, store-and-forward switching, cut-through switching.

Definition of terms

Table 1 provides definitions for some terms used within this document.

Table 1: Definition of Terms

Term	Definition
Controller	A controller is an industrial digital computer which is used for the control of manufacturing processes, such as assembly lines, or robotic devices. It reads inputs, performs logic and writes outputs in real time. Modern controllers also can produce alarms, send emails and communicate with SCADA and HMI systems as well as amongst themselves

Term	Definition
	It is sometimes referred to as a master when deployed in a master/slave context
Drive/IO	A device that exchanges data with the controller. In the context of other technologies this can be referred to as a slave.
Jitter	Variation (difference between the maximum and the minimum) of latency over a time interval.
Latency	The time interval needed to forward one frame through the network from one sender to one receiver.
Scheduled Traffic	The set of data-streams which are exchanged by using the Qbv (scheduling) mechanisms.
Traffic Pattern	Application-derived characteristics of data streams. Traffic patterns enable classification of data transmission scheme, which require similar network guarantees/behaviors.
Traffic Type	Synonymous with Traffic Pattern

Introduction

Recent work by IEEE 802, the Internet Engineering Task Force (IETF), and other standards groups has extended the number of applications that Ethernet networks can serve [1]. These standards define standard Ethernet mechanisms for creating distributed, synchronized, real-time systems. TSN provides mechanisms to solve industrial control applications using standard Ethernet technologies in a manner that enables convergence between and among Operational Technology (OT) and Information Technology (IT) data on a common physical network.

This paper will explore several OT traffic types (motion, I/O and events) and the effects of converging them on a single wire. The techniques described can be extended to apply to convergence of IT and OT traffic.

System Overview

Figure 1 shows a simple industrial control system that will be used to analyze converged network traffic flows. This example contains two separate processes that need to exchange information, such as one might find in conveying or baggage handling systems.

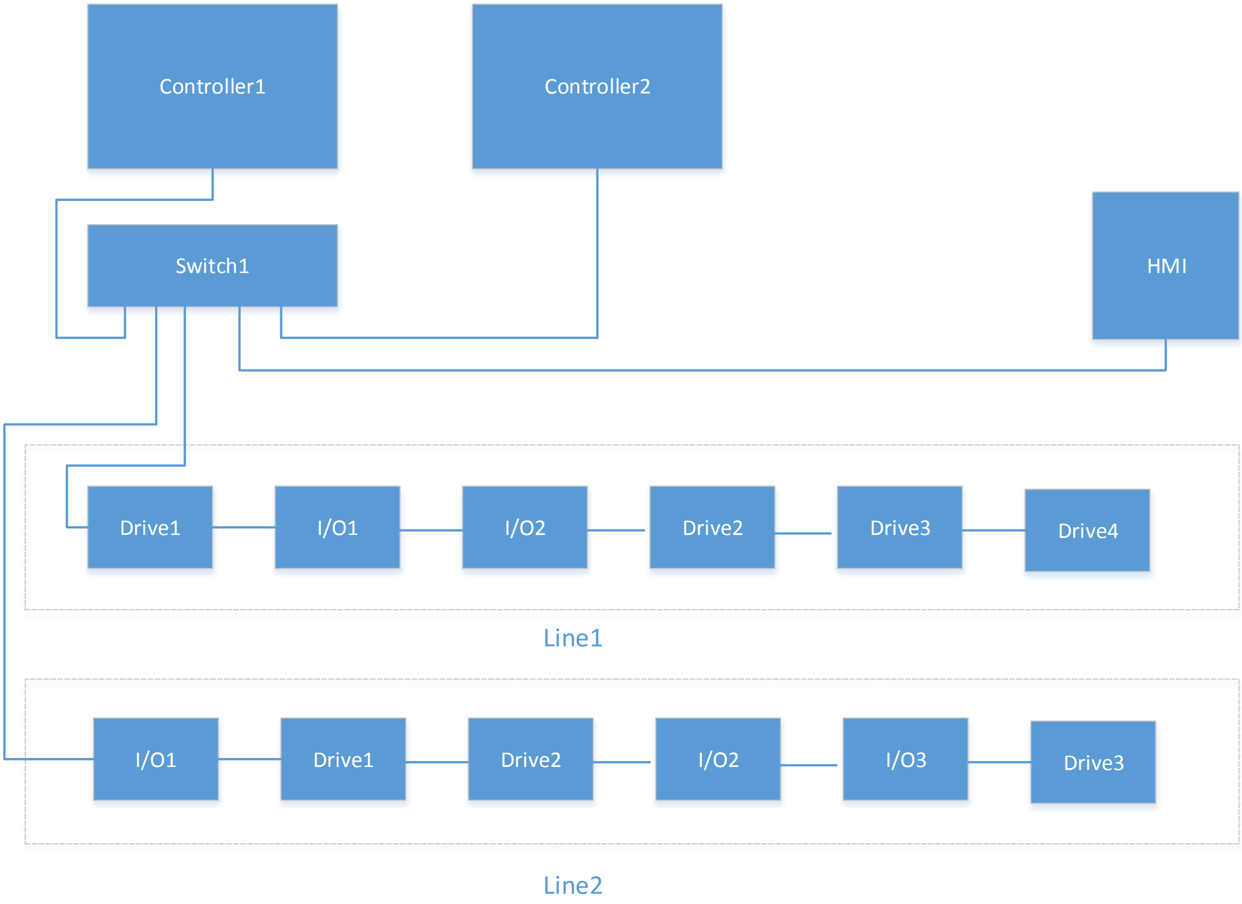


Figure 1: Converged Industrial Control System

In this example, Controller1 manages the drives and I/O modules in Line1 and Controller2 controls the devices in Line2. Both Motion and I/O traffic are present between Controller1 and Line1 devices and the same is true for Controller2 and Line2 devices. Event based traffic flows between Controller1 and Controller2.

Traffic Patterns

Network traffic between industrial controllers and devices like drives and I/O devices typically follow specific patterns and have requirements on various characteristics like periodicity and latency. This chapter introduces three traffic types (patterns) that will be evaluated individually and finally, when converged on a single wire.

Isochronous (Motion)

Isochronous traffic is mainly found in motion control applications. This traffic pattern is cyclic. Unlike the cyclic traffic pattern described in the following section, this traffic pattern has much more stringent requirements related to message latency, has application time synchronized to network time and the cycle times are typically shorter as well.
A controller, which could be controlling multiple axis, sends commands each cycle to the various drives under its control. In the same cycle, the drives return information to the motion controller (feedback). The motion controller is responsible for computing the next command while the drive is responsible for execution of the commands and supplying feedback. This cycle is repeated, often at a very high rate (1 ms or less). If the drives are expected to operate in a coordinated manner, each component of the system

must have some sense of the same time, which is strictly monotonic and steadily increasing, without jumps or leaps. This allows that each drive will apply its commands and sample their feedback at the same instance of time. This further requires that the applications in the motion controller and drives are also synchronized to network time for TSN networks.

For tight control loops, reception jitter must be minimal, with no interference from other traffic. Messages need a guaranteed delivery time. If they arrive later than this deadline, they are ignored for that cycle or discarded, thus potentially affecting the control loop. Message sizes are typically fixed at design time and remain constant for each cycle. Payload sizes are typically under 100 bytes per device. This type can be used for controller-to-controller, controller-to-drive and drive-to-drive communication.

Figure 2 illustrates a typical 1-Cycle Timing Model. The critical timing elements are described in Table 2. This diagram is for illustration purposes only. The order in which some of the timing elements occurs may vary across implementations and timing models.

Table 2: Motion Cycle Timing Elements

Parameter	Description
t_0	Start of cycle, controller sends commands to drives.
t_r	Time at which all commands must arrive at all the drives being controlled.
t_c	Time at which all drives apply new command data. Note, in some implementations, this may occur multiple times per cycle.
t_s	Time at which all drives sample their feedback from encoders, resolvers, etc.
t_f	Time at which all feedback messages must be received by the controller.

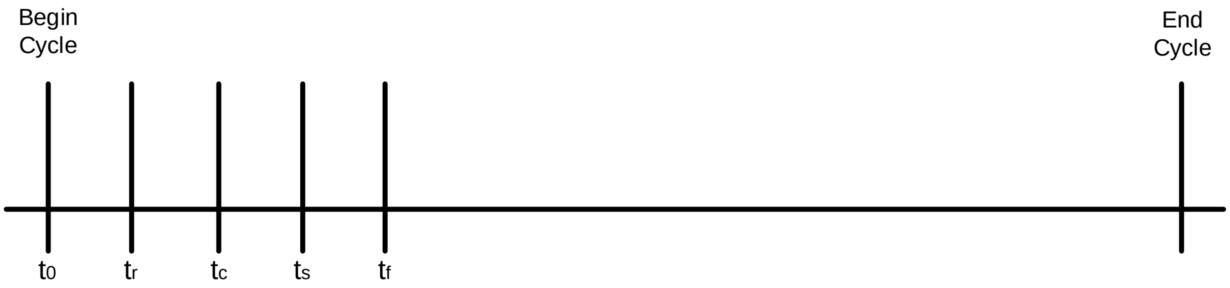


Figure 2: Diagram of 1-Cycle Motion Timing

Many models exist for synchronizing communication in a control system and the order in which the various timing elements can vary based upon the model. What is critical is that there are periods of time in which controllers and devices are sending messages and that these messages must be received before a certain time within each cycle.

IEEE Qbv defines a scheduling mechanism that ideally lends itself to manage the above need. Schedules can be defined in network infrastructure devices such that only the desired traffic can flow during these times, thus preventing interruption from ancillary traffic.

Figure 3 illustrates how isochronous traffic is distributed on the Ethernet wire between a controller (e.g. Controller1) and the Ethernet switch (Switch1) in Figure 1.

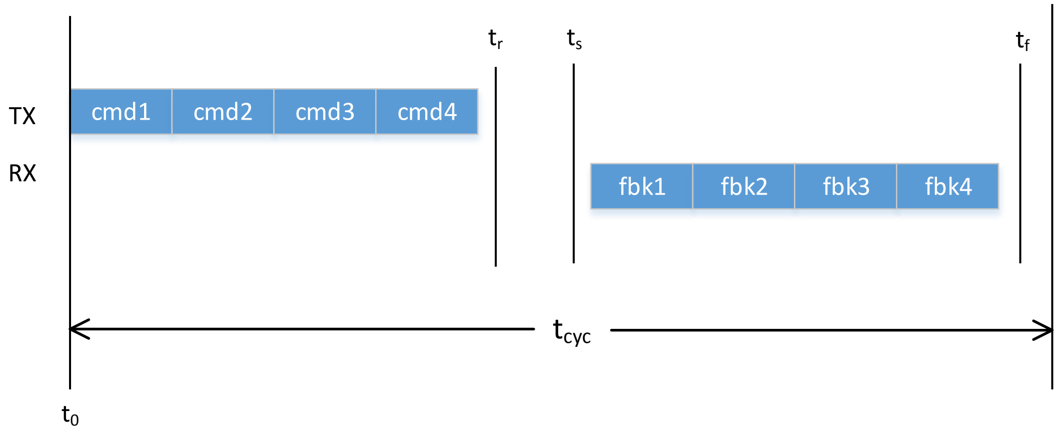


Figure 3: Isochronous Message Timing Diagram

Cyclic (I/O)

Cyclic traffic that is not isochronous is typically used for remote I/O communications. The applications in devices are not synchronized to a common time. Devices sample inputs and apply outputs cyclically, which may or may not be the same as the data transmission cycle. Controllers typically send outputs to all devices one after the other. When using a client-server protocol (e.g. MODBUS ®), output and input messages will be clustered while in a publish/subscribe environment (e.g. EtherNet/IP I/O) output messages from the controller will be clustered while input messages from the devices will be distributed over the cycle time, since these devices are not synchronized to each other. This paper will only consider publish/subscribe cyclic traffic. For best control, the time between a device sending a message and its reception should be minimized, with predictable interruptions from other traffic. Messages require a defined maximum latency time. Data message sizes are fixed at design time and remain constant for each cycle. This traffic type can be used for controller to controller, controller to I/O and I/O to I/O communication.

Figure 4 illustrates how the cyclic traffic is distributed on the Ethernet wire between a controller (e.g. Controller1) and the Ethernet switch (Switch1) in Figure 1.

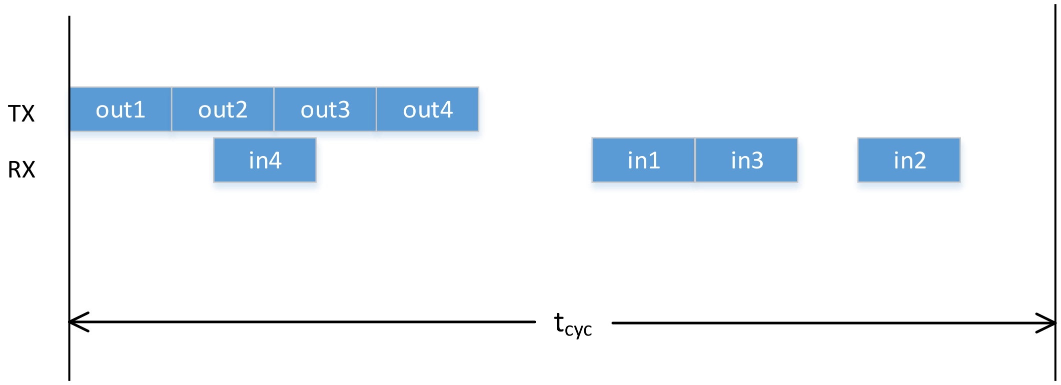


Figure 4: Cyclic Message Timing Diagram

Alarm and Control Events

In a system when an input or output variable change occurs (control event), or an occurrence needs to be announced (alarm event), event messages are generated. Depending upon the event, this might be a single message, or a flurry of messages (domino effect). While the messages may be directed to different end-devices, as in the case of control events, alarm event messages are typically directed at a single device, like an HMI or SCADA system. The network must be able to handle a burst of messages without loss, up to a certain number of messages over a defined period.

For alarm messages, after this period, messages can be lost until the allowed bandwidth quantity has been restored. Applications are designed to compensate for message loss situations.

Figure 5 illustrates how the Event / Alarm traffic is distributed on the Ethernet wire between a controller (e.g. Controller1) and the Ethernet switch (Switch1) in Figure 1.

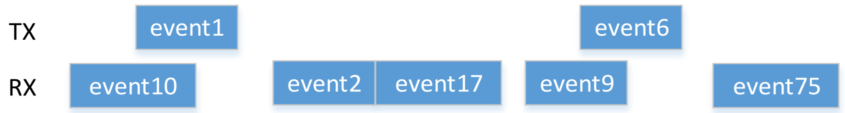


Figure 5: Control Event / Alarm Message Timing Diagram

TSN Overview

This section will discuss various aspects of TSN that will be considered in the analysis of converged networks.

TSN enhances Ethernet (specifically IEEE 802.1 and 802.3), a foundational piece of the “internet of things” (IoT). TSN adds a range of functions and capabilities to Ethernet to make it more applicable to industrial applications that require more deterministic characteristics than possible in previous Ethernet implementations. Table 3 summarizes those enhancements.

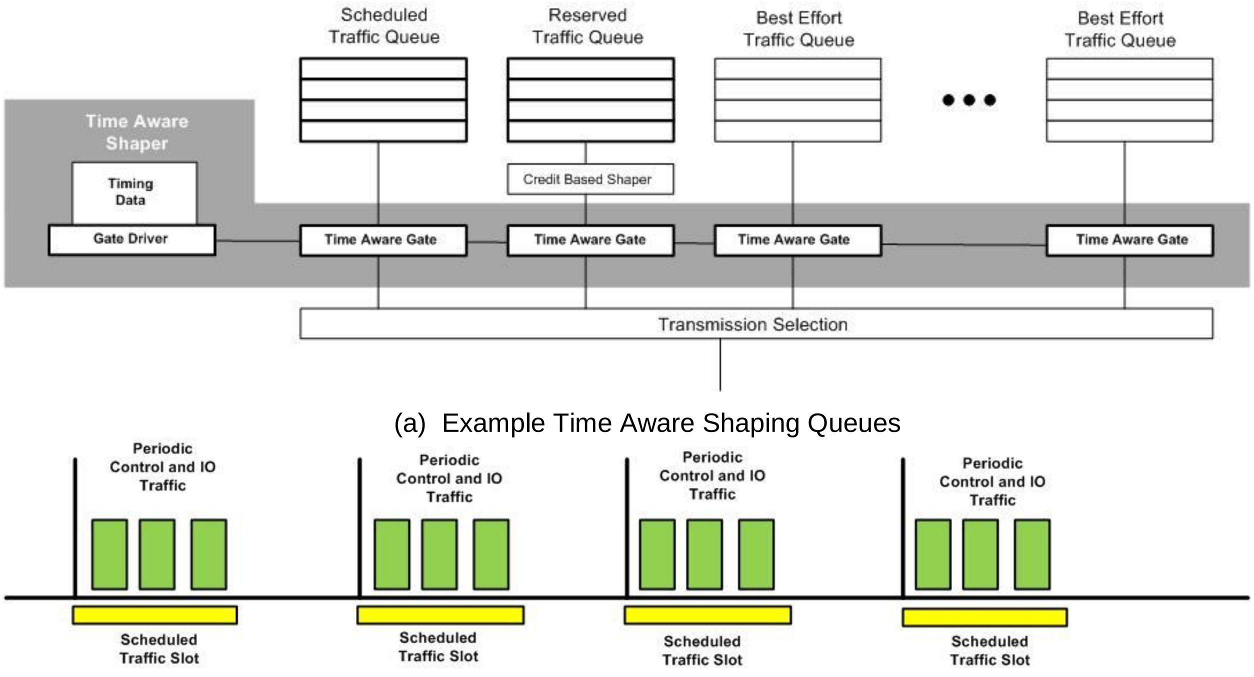
Table 3: Set of IEEE TSN Enhancements

Standard	Title
IEEE 802.1Qav	Forwarding and Queuing Enhancements for Time-Sensitive Streams
IEEE 802.1AS-Rev	Timing and Synchronization for Time-Sensitive Applications
IEEE 802.1Qbu & IEEE 802.3br	Frame preemption
IEEE 802.1Qbv	Enhancements for Scheduled Traffic
IEEE 802.1Qca	Path Control and Reservation
IEEE 802.1Qcc	Stream Reservation Protocol (SRP) Enhancements and Performance Improvements
IEEE 802.1Qci	Per-Stream Filtering and Policing
IEEE 802.1CB	Frame Replication & Elimination for Reliability

Automation and control systems require devices, including the network, to perform in a deterministic way. One of TSN’s benefits is the ability to converge applications and traffic on a single, open network, while retaining deterministic behavior.

Time Aware Traffic Shaping [2]

Time Aware Traffic Shaping allows for scheduled traffic. In order to enforce the schedule throughout a network, the interference with lower priority traffic has to be prevented, as this would not only increase the latency but also the delivery variation (jitter). The Time Aware Shaper blocks the non-Scheduled Traffic (i.e. non-scheduled traffic is queued), so that the port is idle when the Scheduled traffic is scheduled for transmission.



(a) Example Time Aware Shaping Queues

(b) Example Shaped Traffic

Figure 6: Time Aware Traffic Shaping

Figure 6 illustrates Time Aware Traffic Shaping of scheduled traffic. Figure 6-(a) illustrates example Time aware shaping Queues, and Figure 6-(b) illustrates example shaped Traffic.

QoS - Strict priority.

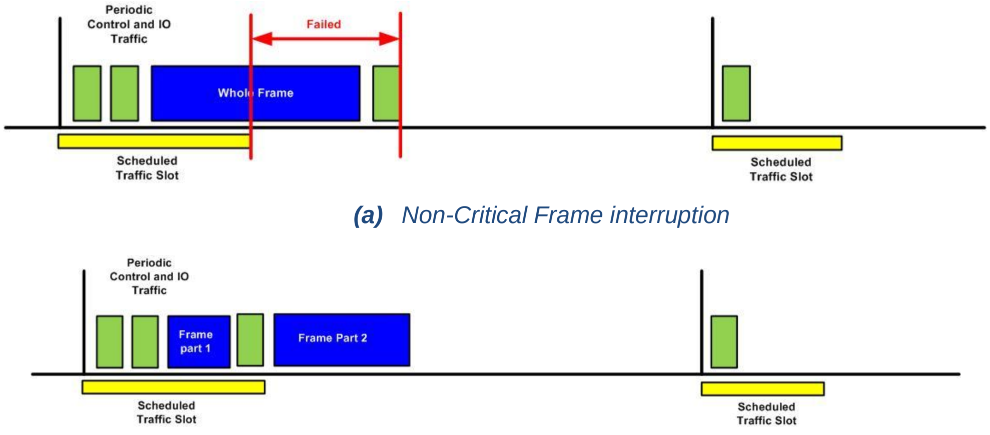
Strict priority is the default queuing mechanism for Ethernet bridges (switches). In strict priority queueing the queue with the highest number has priority over the remaining queues. When multiple Ethernet messages are queued on an interface for transmission, the queue with the highest priority having an Ethernet frame ready for transmission will transmit. Ethernet frames in lower priority queues are held until the priority of their queue becomes the highest queue with a ready Ethernet frame.
Shaped queues may have a numerically higher traffic class, but the transmission selection algorithms bound to the corresponding queues determine if such a queue is served before a lower priority queue (e.g. a queue bound to the credit-based transmission selection algorithm could have a numerically higher traffic class but would not be served if it has insufficient credits (i.e. does not have a frame ready for transmission)).

Frame Preemption and Interspersing Express Traffic [2]

The Frame Preemption amendment specifies procedures, managed objects, and protocol extensions that:

- Define a class of service for time-critical frames that requests the transmitter in a bridged Local Area Network to suspend the transmission of a non-time-critical frame, and allows for one or more time-critical frames to be transmitted. When the time-critical frames have been transmitted, the transmission of the preempted frame is resumed. A non-time-critical frame could be preempted multiple times.
- Provide for discovery, configuration, and control of preemption service for a bridge port and end station.
- Ensure that preemption is only enabled on a given link if both link partners have that capability.

The purpose of this amendment is to provide reduced latency transmission for scheduled, time-critical frames in a bridged LAN.



(a) *Non-Critical Frame interruption*

(b) *Frame Preemption*

Figure 7: Frame Preemption and Interspersing Express Traffic

Figure 7-(a)shows that a large, non-time-critical frame (in blue) may start transmission ahead of the desired transmission time of time-critical frame (in green). This condition leads to excessive latency for the time-critical frame. Transmission preemption preempts the non-time-critical frame to allow the time-critical frames to be transmitted as shown in Figure 7-(b). This provides the capabilities of an application that uses scheduled frame transmission to implement a real-time control network.

Cut-through Switching

Cut-through switching [3] is where a switch starts forwarding a message (packet) before the whole message has been received. Compared to store-and-forward switching, cut-through switching offers lower latency but, because the frame check sequence appears at the end of a message, the switch is not able to verify message integrity before forwarding it. Cut-through switching will forward corrupted packets, whereas a store-and-forward switch will drop them.

Pure cut-through switching is only possible when the speed of the outgoing interface is equal to or lower than the incoming interface speed.

A switch may buffer (acting in a store-and-forward manner) a packet instead of using cut-through under certain conditions:

- Speed: When the outgoing port is faster than the incoming port, the switch must buffer the entire message received from the lower-speed port before the switch can start transmitting that message out the high-speed port, to prevent underrun. (When the outgoing port is slower than the incoming port, the switch can perform cut-through switching and start transmitting that message before it is entirely received, although it must still buffer some of the message).
- Congestion: When a cut-through switch decides that a message from one incoming port needs to go out through an outgoing port, but that outgoing port is already busy sending a message from a second incoming port, the switch must buffer some or all of the message from the first incoming port.

Traffic Analysis

Industrial control systems require predictability of certain communications for optimal control. In today’s architectures, this is accomplished using methods such as proprietary and IEC standardized fieldbus technologies, network segmentation and isolation, over provisioning, etc. With TSN’s deterministic shaping mechanisms, these segmented traffic types can be converged in a single network while retaining each traffic’s delivery requirements.

There are several factors that affect the time it takes for a message to exit one device and be received by another, such as network data rate, message length, whether cut-through or store-and-forward switching is used, interrupting traffic and the number of switches in the path between source and destination. Before one can begin to understand the effects of converged traffic on a particular traffic type, each traffic type must be analyzed in isolation without the presence of any potential disturbances. This will establish a baseline upon which comparisons can be made.

Table 4 defines the variables used in subsequent formulas to calculate various aspects of message latencies.

Table 4: Variable Definitions

Variable Name	Description
dr	Ethernet data rate in bits per second.
l_{msg}	Length of message, in bytes. Includes preamble,start of header and inter frame gap.
l_{ct}	Number of bytes needed to be received to make cut through decision. Includes preamble and start of header.
t_{msg}	Amount of time it takes to transmit a message across the Ethernet wire. Includes preamble, Start of header and inter frame gap
t_{ct}	Time to determine cut through decision. Includes preamble and start of header.
n_{sw}	Number of switches in the network between two devices exchanging a message.
t_{sw}	Time required by an Ethernet switch to forward a received message. This time may be different for store-and-forward vs. cut-through and also for network speeds.
t_{xmt}	Amount of time elapsed from transmission of first bit of message until last bit of message arrives at receiving device (latency).
t_{int}	Interfering message time. Subscripts may be used to indicate a particular interfering message
t_{sch}	Time for which only scheduled traffic may flow. Considered an interruption for traffic using strict priority.

Equation 1 can be used to calculate the length of time it takes to transmit a message at a specific data rate. It is assumed that the message length is in bytes. Note, for completeness, the message length should include the preamble, start of header and inter-frame gap fields.

$$t_{msg} = \frac{l_{msg} * 8}{dr} \quad (1)$$

Equation 2 calculates the time required to receive the requisite number of bytes in the start of a message before a switch can make a cut-through forwarding decision. Note, for completeness, the length should include the preamble and start of header fields.

$$t_{ct} = \frac{l_{ct} * 8}{dr} \quad (2)$$

The time it takes for a message to be delivered to its intended destination depends upon several factors:

- message length;
- Ethernet data rate;
- number of switches the message must pass through;
- whether the switches use cut-through or store-and-forward switching.

The effects of store-and-forward switching are shown in Figure 8. A line topology is used for this demonstration. Each row in the figure represents an ingress or egress time of a message to or from a particular device or switch in the line. A message originator (or talker, in TSN terms), three switches plus an end-device (which is the message destination or target and listener in TSN terms) are shown.

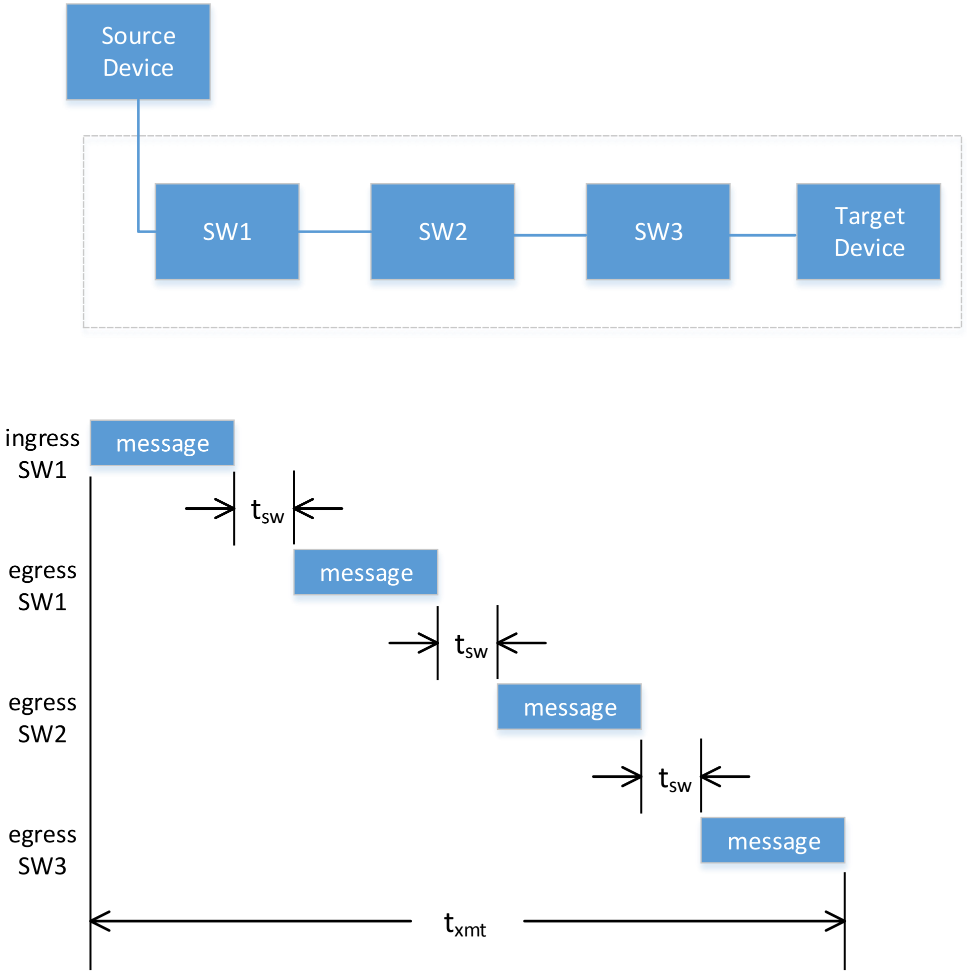


Figure 8: Store-and-Forward Message Timing

Since the format of Figure 8 is used throughout this paper to represent message timings, a brief explanation follows. The top of the figure represents the topology of the network being analyzed. On the left side, labels are used to describe whether the messages are ingress to or egress from a particular port on a specific switch. Rectangles, with message labels inside, are used to represent message times, with time flowing from left to right. Message labels like HP_0, HP_1, etc. or ‘high priority’ are used to represent high priority messages while terms like int_0, int_1, etc. or ‘low priority’ are used to represent lower priority messages. A dimension line is used to show the total latency (t_{xmt}) for the message of interest. When two switches’ ports are directly connected (e.g. P2 of SW1 connected to P1 of SW2) egress from one port is equivalent to the ingress of the other. Wire speed times are not considered in any diagrams or calculations. Switches take time to calculate which message should be forwarded next, represented by t_{sw}. The timing of t_{sw} is from the completion of a message ingress to the start of a message egress or, in the case of cut-through, after the requisite data has been received. To demonstrate maximum interference, all interfering messages in this document’s figures are depicted at the latest possible arrival time to still cause interference.

Equation 3 calculates the time from the start of a message’s transmission from the message originator (source) to its complete reception time at its destination (target) using store-and-forward switching:

$$t_{xmt} = t_{msg} + n_{sw} * \left(t_{msg} + t_{sw}\right) \qquad (3)$$

While one might envision the flow of messages in this type of architecture being analogous to water flowing in a pipe, it is more like traffic flow on a city street with stop signs at each intersection. Each message must be fully received at each switch prior to being forwarded to the next switch in line as it makes its way towards its ultimate destination.

As can be seen by Figure 8 and Equation 3, message delivery time across multiple switches using store-and-forward switching is a multiple of the message transport time. Using the same configuration, Figure 9 shows the timing using cut-through switching.

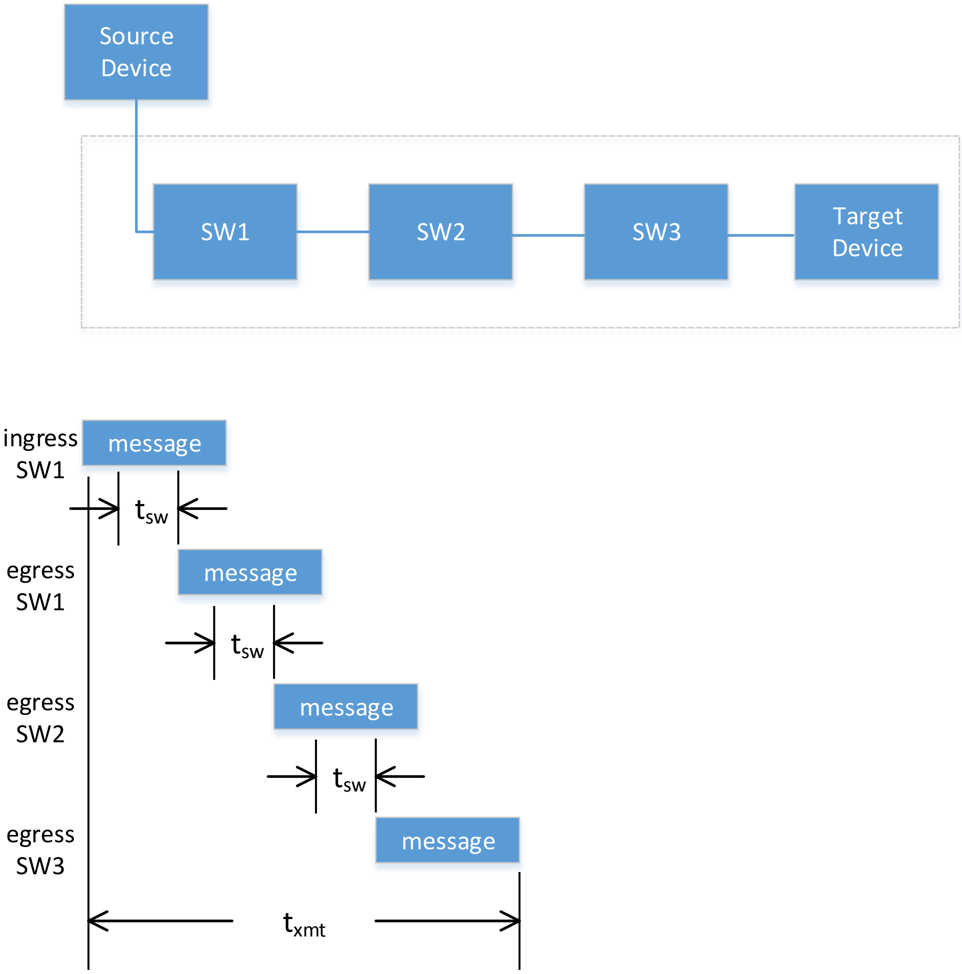


Figure 9: Cut-Through Message Timing

Equation 4 is used to calculate message timing for cut-through switching:

$$t_{xmt} = t_{msg} + n_{sw} * (t_{ct} + t_{sw}) \qquad (4)$$

The above formulas predict the best-case scenarios, with no interfering traffic.

Figure 10 and

Figure 11 show the differences between message latency for various message sizes for store-and-forward and cut-through switching. In the case of store-and-forward, the message size affects the latency at each switch while in cut-through, the latency through each switch is constant, regardless of message size, so message size is only considered once. Cut-through switching benefits are clear as the number of switch hops increases between source and destination as well as for larger message sizes. Of course, as data rates increase, if forwarding times remain similar, the benefits of cut-through switching diminishes, especially for short messages.

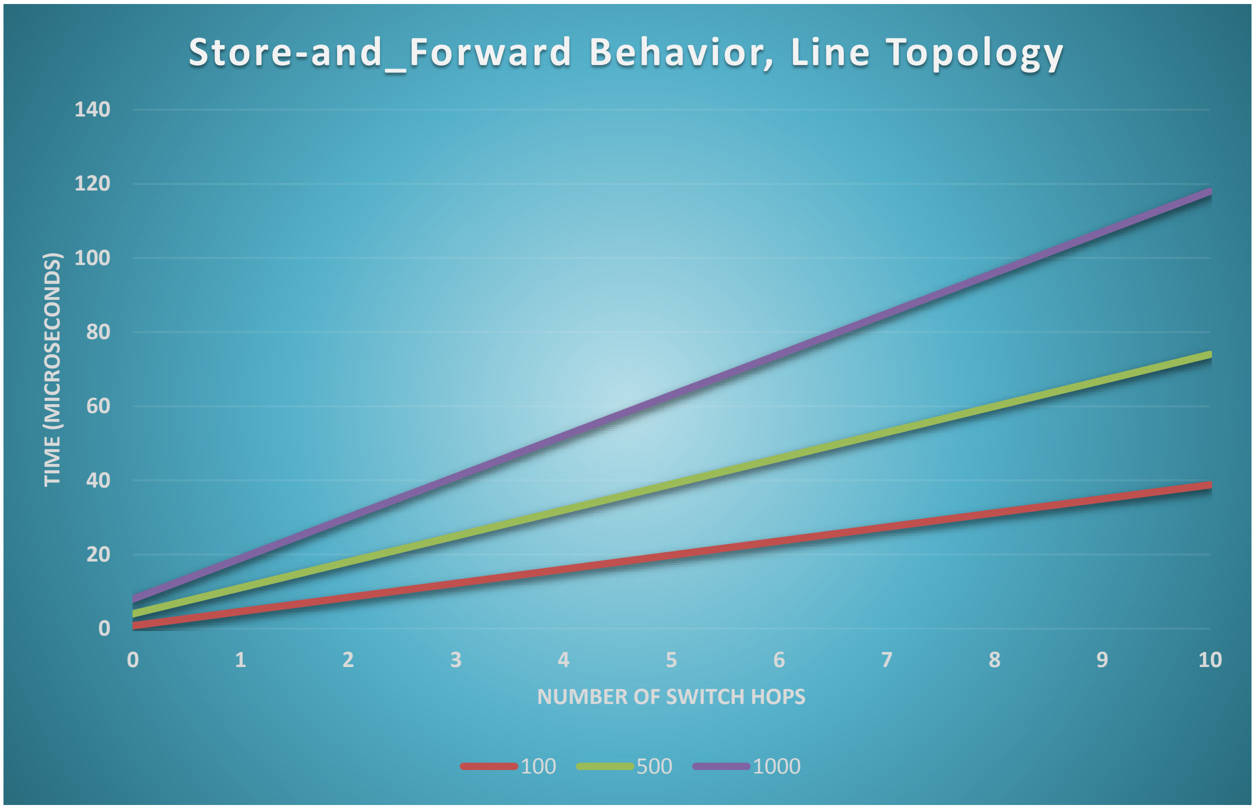


Figure 10: Store-and-Forward Message Size Impact

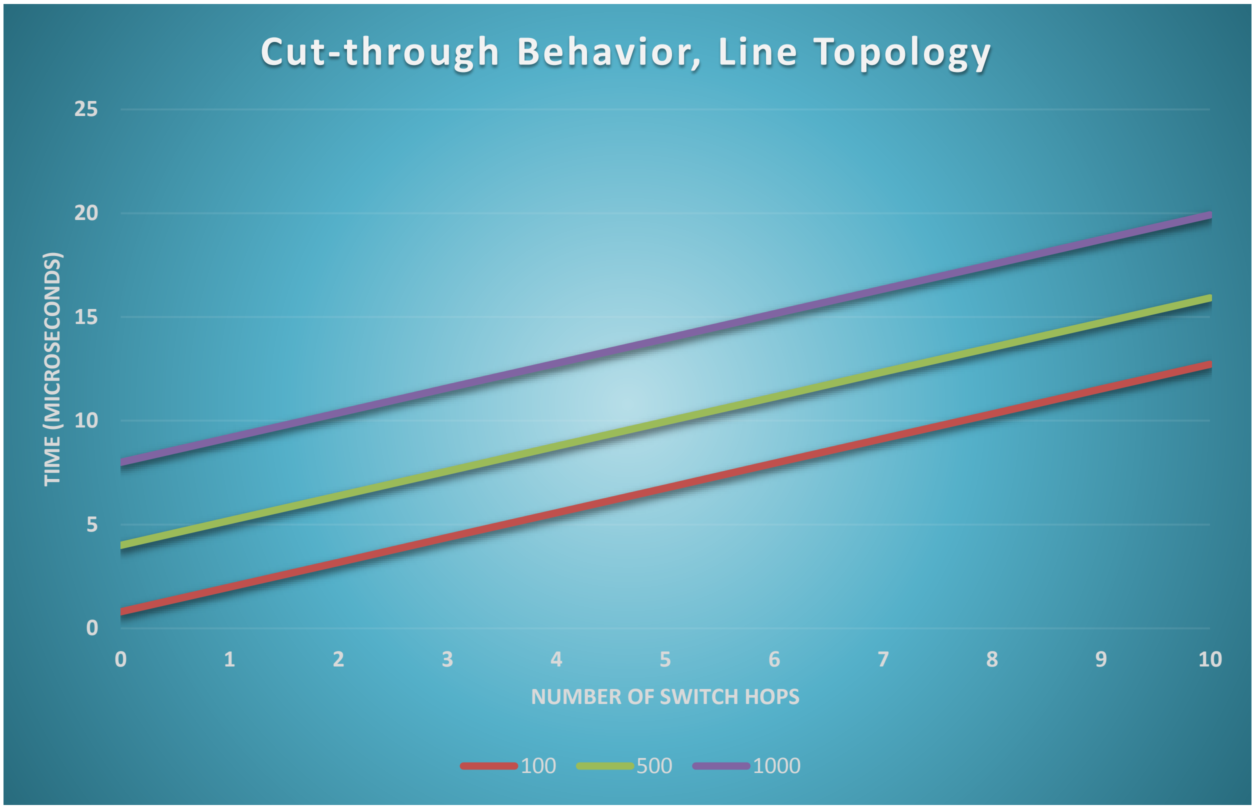


Figure 11: Cut-Through Message Size Impact

Interference

This section will consider the effects of messages that interfere with another message as it travels to its intended destination. Only store-and-forward switching scenarios are considered.

In today's industrial control environment, devices with two external switched ports (enabling easy daisy-chaining of devices) are quite common. These devices are constructed using a three-port switch, two that are exposed to outside connections while the third is typically connected to the Central Processing Unit (CPU) of the device. Figure 12 demonstrates this architecture and will be used in subsequent examples which demonstrate various interference examples.

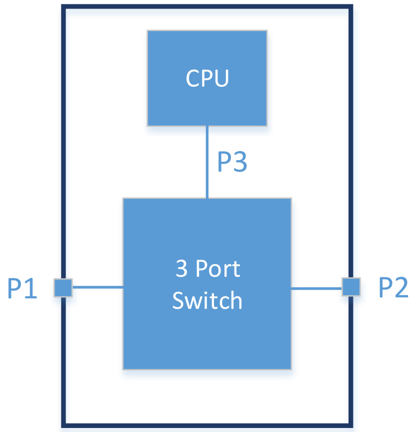


Figure 12: Three Port Switch Diagram

Interfering traffic can occur when a switch is already transmitting a message out a particular port and another message, destined for the same port, arrives. This newer message will have to wait for the completion of the already in progress message before it can be sent, as shown in Figure 13. This form of interference can happen regardless of message priorities.

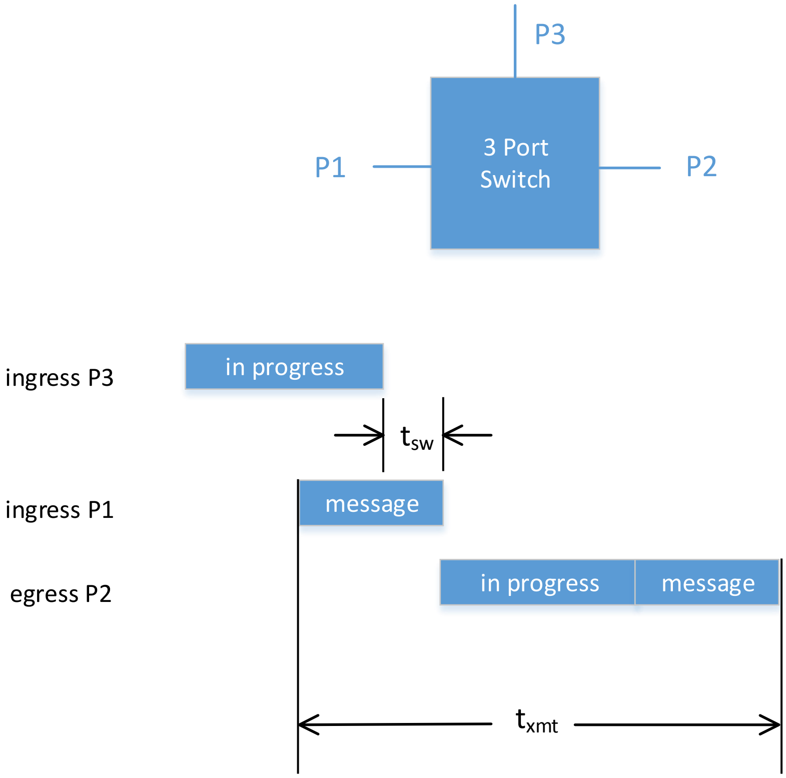


Figure 13: Interference by Message Already In Progress

Another form of interference is if a higher priority message is already queued in a switch when a lower priority message arrives (is queued). The switch will choose to send the higher priority message first, thus delaying the lower priority message, as shown in Figure 14.

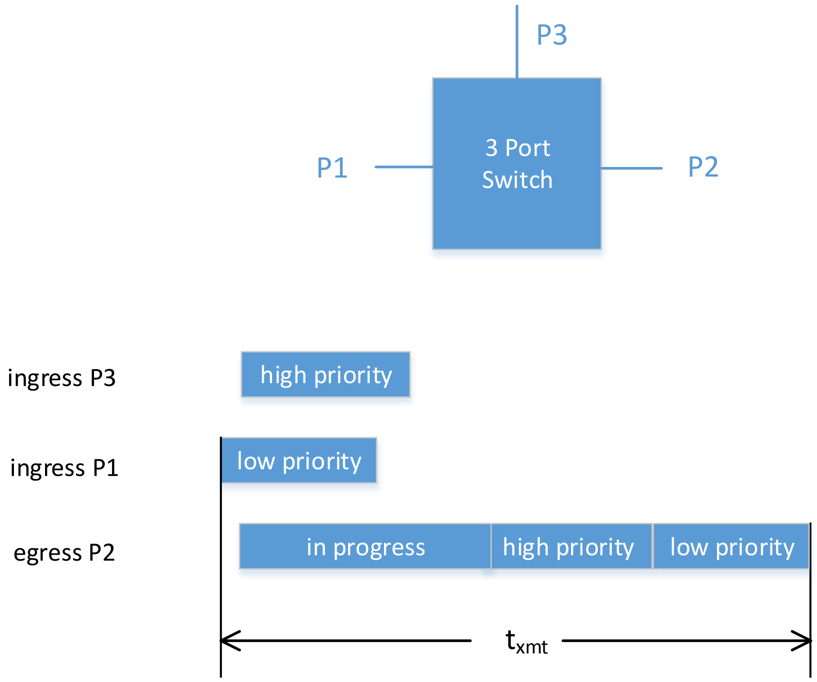


Figure 14: Interference by Higher Priority Message

The above examples demonstrate how a message travelling through a switch can get delayed due to interference. In order to determine maximum message latencies, the time when messages arrive (ingress) and message priorities must be taken into account.

In-progress Interference

In-progress interference occurs when a message destined for a particular port arrives after the switch has already decided to send another message out that same port. Message priority has no effect in this case, since the decision was made prior to the arrival of a potentially higher priority message. Assume a high priority message (HP_0) is arriving on port 1 (P1) destined for port 2 (P2) and a lower priority interfering message (int_0) is arriving from port 3 (P3) or has already been queued or is currently being transmitted out of P2.

The worst-case scenario occurs when the interfering message is queued just prior to the switch deciding which message to send next and the high priority message is queued just after the switch makes this decision. In this case, the lower priority message will be sent prior to the high priority message. This will cause a delay of the high priority message equivalent to the amount of time remaining to send a message already in progress, which could be the time for the entire message in the case of starting transmission just before the arrival (queuing) of the high-priority message. If the interfering message is shorter than the high-priority message, this initial delay is the only delay the message will experience as it travels through more switches to its destination (see Figure 15).

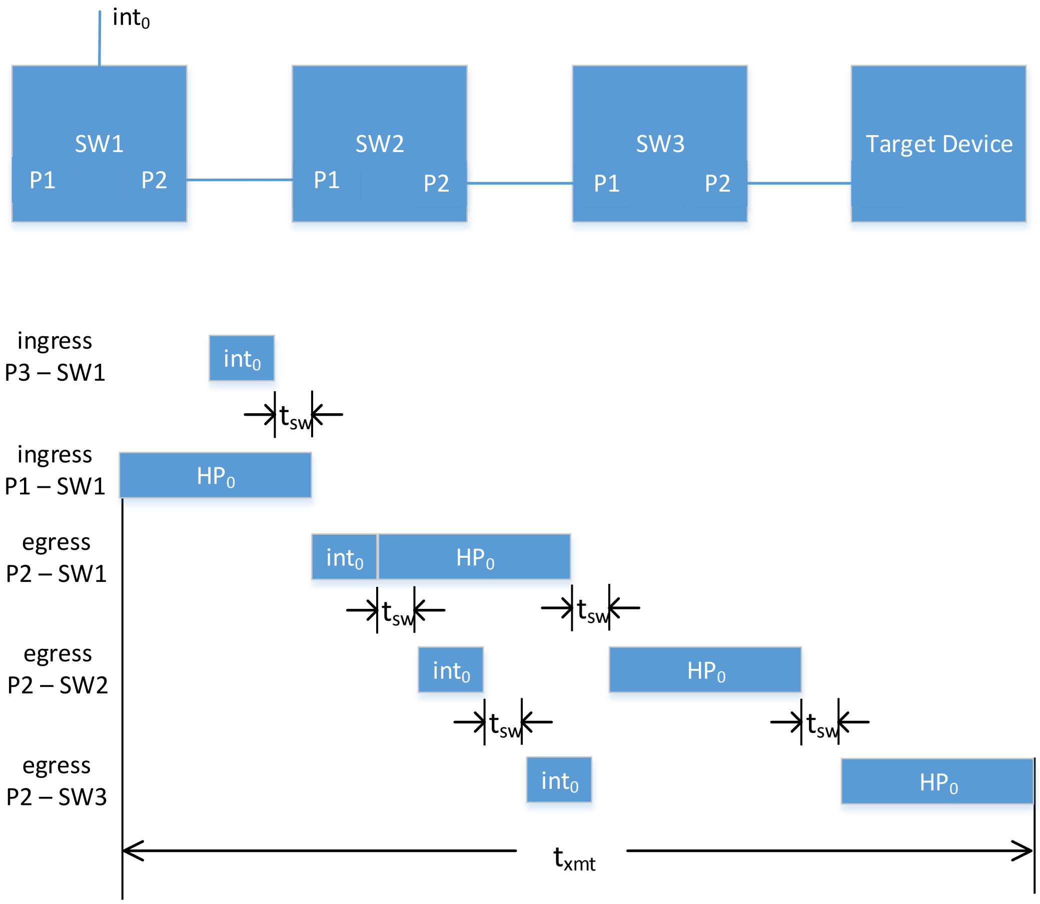


Figure 15: Single Small Interfering Frame Scenario

Equation 5 can be used to calculate the effect of a single shorter interfering message. The number of switch hops (n_{sw}) represents the number of switches between the source device that injected the interfering message and the message destination (target).

$$t_{xmt} = t_{int} + t_{msg} + n_{sw} * \left(t_{msg} + t_{sw}\right) \qquad (5)$$

However, if the interfering message is longer than the high-priority message, a delay equivalent to the time of the interfering message will occur at each subsequent switch hop as shown in Figure 16.

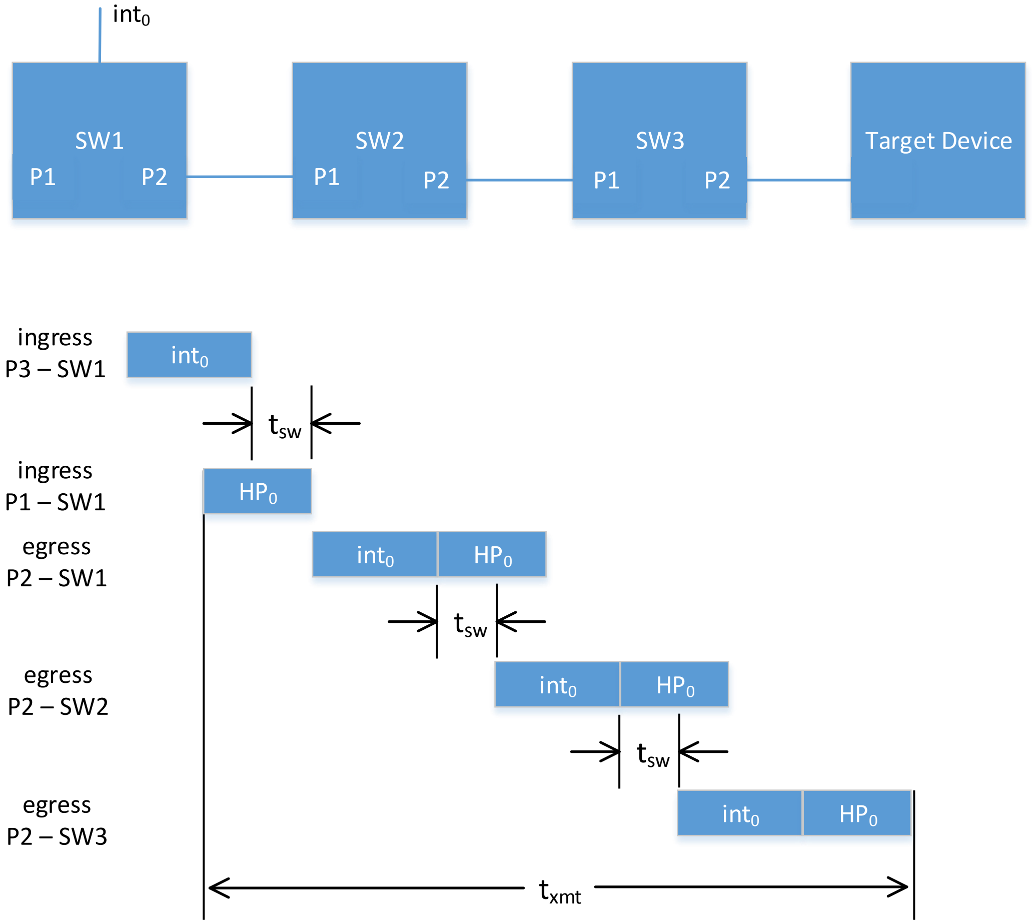


Figure 16: Single Large Interfering Frame Scenario

Equation 6 can be used to calculate the effect of a single longer interfering message. The number of switch hops (n_{sw}) represents the number of switches between the source device that injected the interfering message and the message destination.

$$t_{xmt} = t_{int} + n_{sw} * (t_{int} + t_{sw}) + t_{msg} \tag{6}$$

Subsequent Interfering Messages

If a message must travel through multiple switches before reaching its destination, additional interference at each of these switches is possible. This section will analyze what happens when additional interfering messages are present at subsequent switches. The first case is when the additional interfering messages are of lower priority than the high priority message and the second case is when they are of equal or higher priority.

Subsequent Interfering Messages, Lower Priority

While it is interesting to examine the case where the initial lower priority message is shorter than the high priority message, it does not represent the worst-case possibility. In fact, the worst-case scenario is when the initial lower priority message is the maximum length permitted by the network. Hence, the following examples will only consider the case where the initial lower priority message (int_0) is longer than the high priority message (HP_0).

On a subsequent switch in the line (SW2), there are two messages arriving on port 1 (P1) destined for port 2 (P2), an initial lower priority interfering message (int_0) followed by a high priority message (HP_0). As in the previous example, the new lower priority interfering message (Int$_1$) should be queued just prior to the arrival (queuing) of the first interfering message (int$_0$) so the switch will send int$_1$ as its next message and queue int$_0$. Two cases need to be considered. The first is that HP$_0$ is queued prior to the switch completing its sending of int$_1$. In this case, when the switch performs its evaluation of which message to send next, both HP$_0$ and int$_0$ are already queued and the switch will decide to send HP$_0$, since it has a higher priority, instead of int$_0$, thus HP$_0$ will hop in front of int$_0$ (See Figure 17).

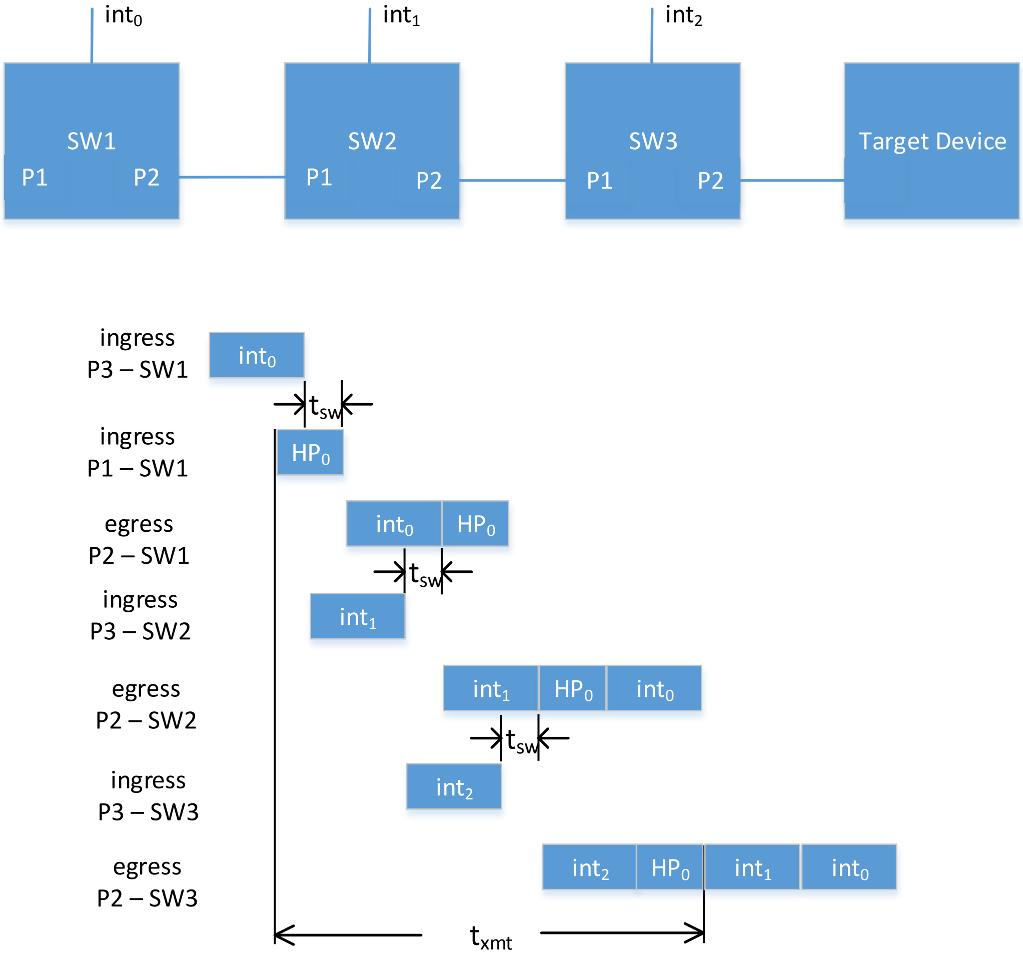


Figure 17: Leap Frog Effect Due to Priority Switching

Equation 7 can be used to calculate maximum latency when all interfering messages are larger than the high priority message.

$$t_{xmt} = t_{int_0} + \sum_{n=1}^{n_{sw}} \left(t_{int_n} + t_{sw}\right) + t_{msg} \tag{7}$$

which collapses to equation 6 when all interfering messages are the same size and larger than the high priority message.

When the second, and subsequent, interfering messages (int$_1$, int$_2$, etc.) arrive prior to int$_0$, they will be sent before int$_0$. If this new interfering message completes egressing the switch before the arrival of the high priority message, int$_0$ will egress ahead of the high priority message as well. The additional delay can be as long as the transmission time of the high priority message (See Figure 18).

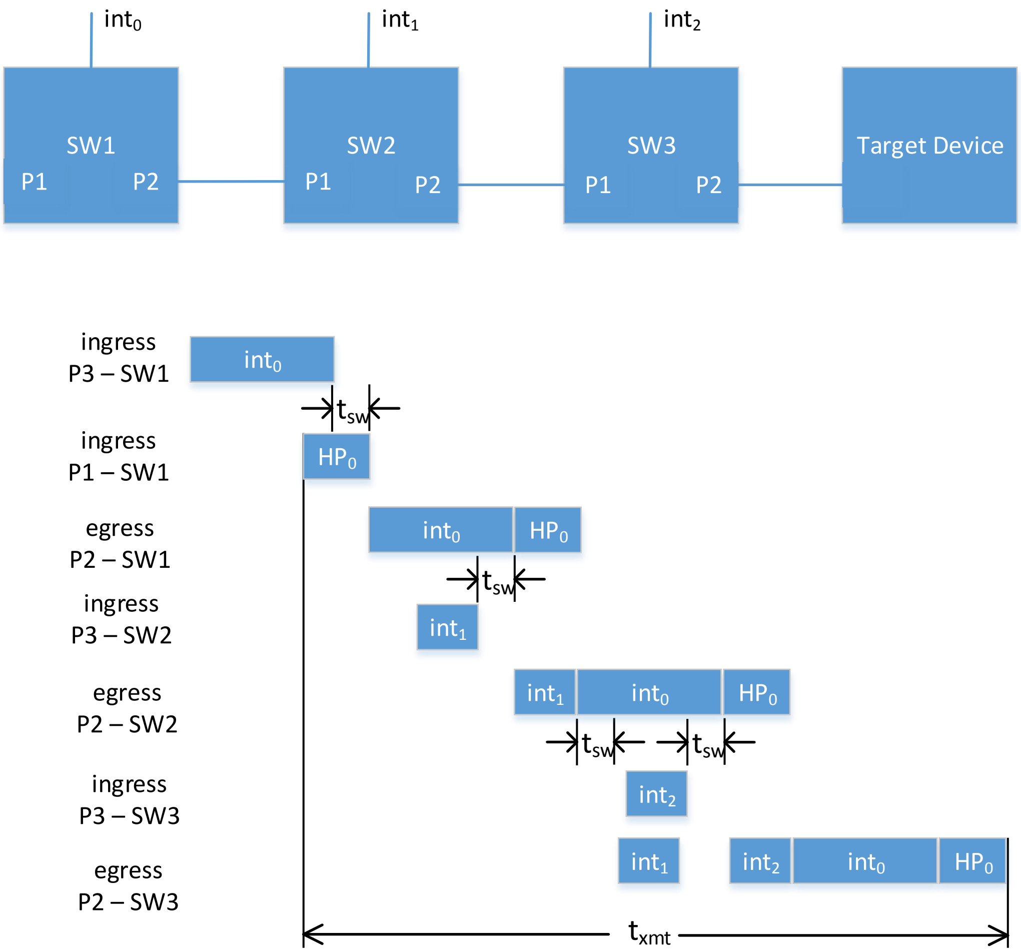


Figure 18: Additional Delay Due to Short Interfering Messages

Equation 8 describes the effect of subsequent interfering messages being shorter than the high-priority message.

$$t_{xmt} = t_{int_0} + \sum_{n=1}^{n_{sw}} \left(t_{int_0} + t_{int_n} + t_{sw}\right) + \ t_{msg} \qquad (8)$$

which collapses to equation 9 when all subsequent interfering messages are the same size and equal in length to the high priority message.

$$t_{xmt} \ = \ t_{int_0} + \ n_{sw} * \ \left(t_{msg} + \ t_{int_0} + \ t_{sw}\right) + \ t_{msg} \qquad (9)$$

Higher Priority Interfering Messages

In the previous example, all interfering messages were lower in priority than the high priority message. If the initial interfering message is, instead, at the same or higher priority as the initial high priority message, the order of messages egressing the switch will change. For a message equal in priority to the initial high priority message, it just needs to be queued prior to the initial high priority message to cause interference.

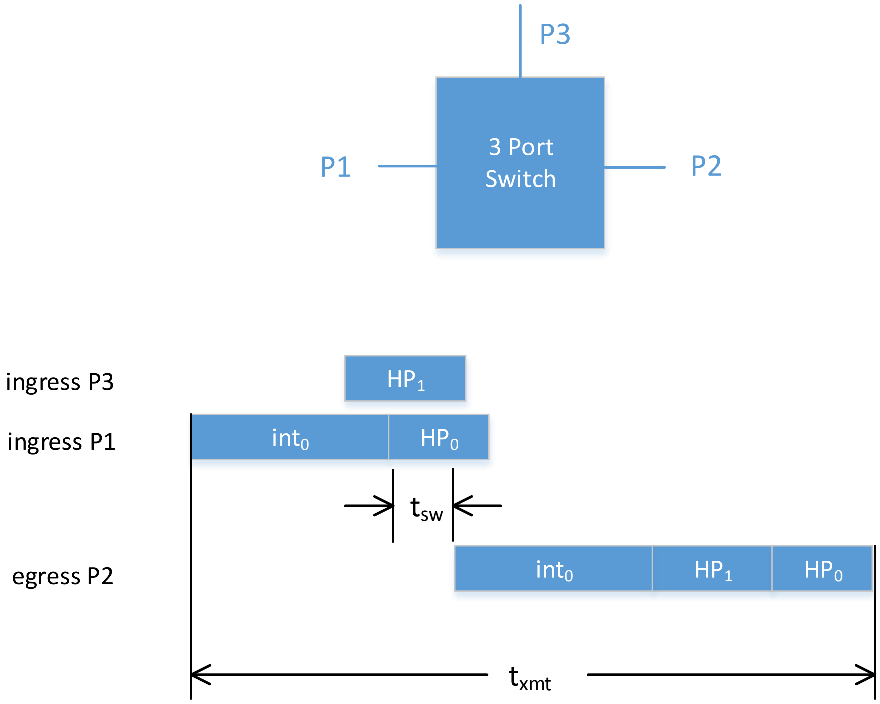


Figure 19 illustrates the scenario where a message of equal priority (HP$_1$) interferes with an already interfered with high priority message (HP$_0$).

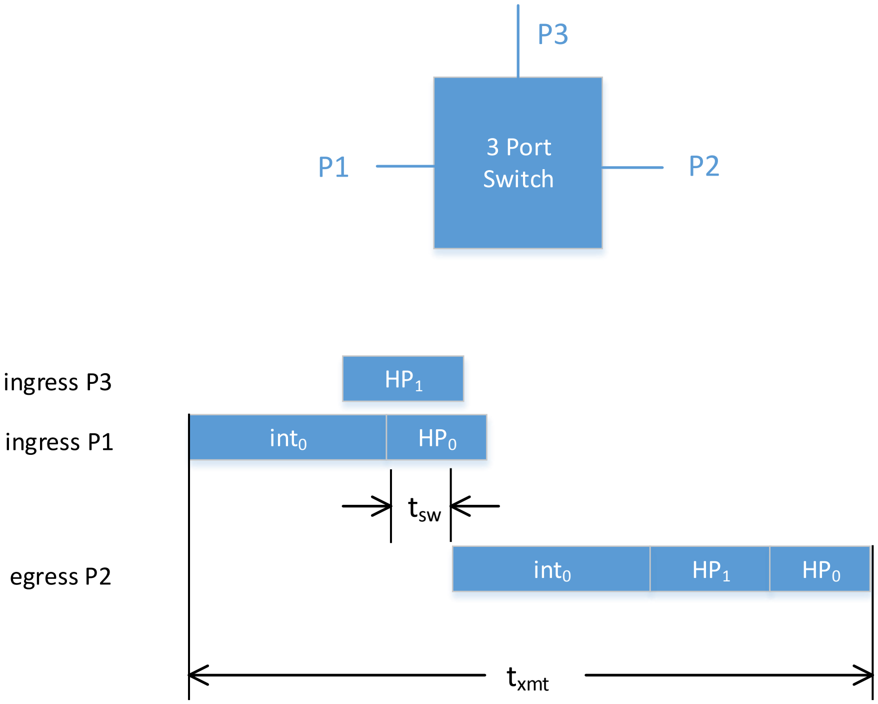


Figure 19: Equal Priority Interference

For a message of higher priority than the initial high priority message, it just needs to be queued prior to the switch deciding which message to send next.

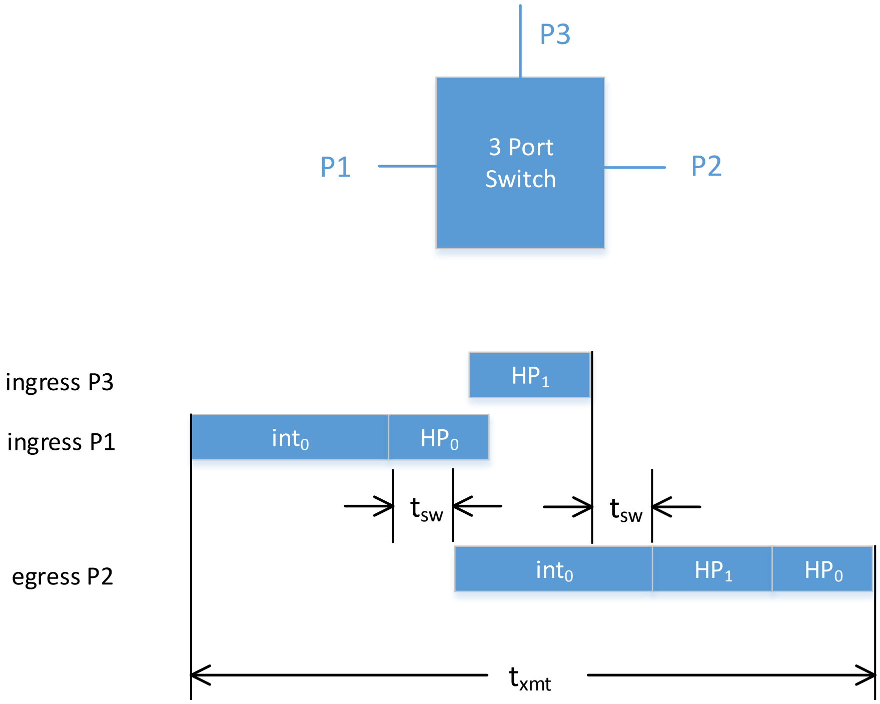


Figure 20 illustrates the scenario where a message of higher priority (HP_1) interferes with an already interfered with high priority message (HP_0).

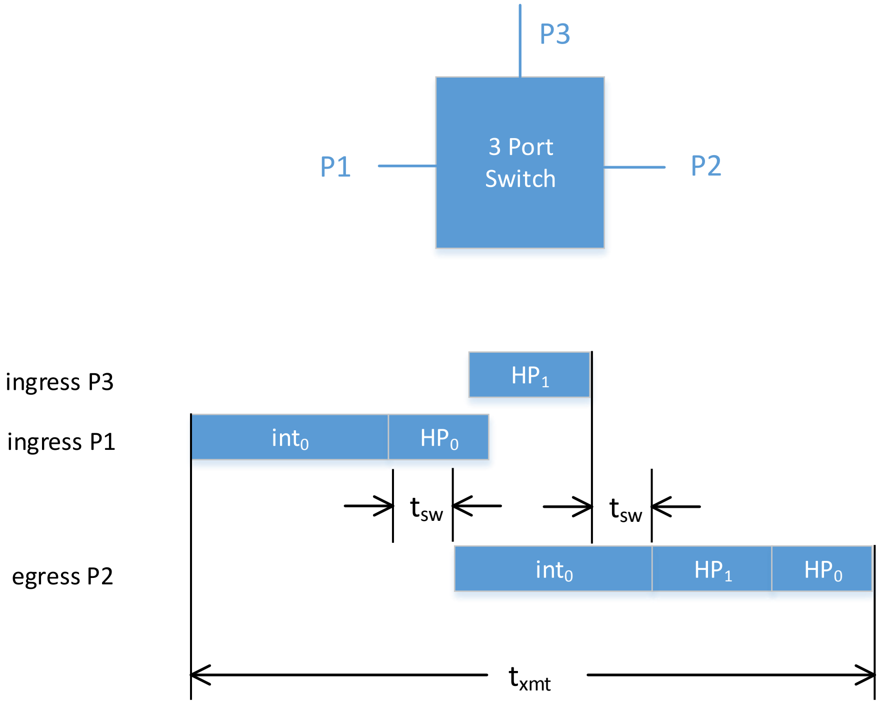


Figure 20: Higher Priority Interference

Deriving a general formula for this use case is quite difficult. Factors such as variation of interfering message lengths, priorities and order coupled with the store-and-forward behavior potentially introducing gaps contribute to the difficulty. Hence, only a formula assuming that all interfering messages are of equal length is provided. Equation 10 assumes only one interfering message per switch.

$$t_{xmt} = t_{int_0} + n_{sw} * \left(t_{int} + t_{int_0} + t_{sw}\right) + t_{msg} \qquad (10)$$

Cut-through Interference

Analyzing the behavior of interfering traffic in a cut-through situation is much more complicated. First and foremost, cut-through behavior is not defined by the IEEE and, as such, is up to a vendor's discretion regarding implementation. For example, if a switch begins receiving a message destined for a particular port and that port is already occupied sending another message, should the switch truncate the current message, wait for the completion of the current message and begin transmitting (depending upon priority) or should it fully complete storing the message before deciding on a course of action? Additionally, if a series of messages is being received on port 1, destined for port 2 and another message of equal priority is queued, is there an opportunity for this queued message to be transmitted or does the switch give preference to the cut-through messages? With all of these unknowns, and that the IIC document on traffic types [4] does not recommend the use of cut-through for strict priority traffic, this document will not explore interference behaviors of cut-through switching for strict priority traffic. Cut-through switching will only be considered for scheduled traffic.

Scheduling Interference

Finally, in a converged TSN network, schedules may exist that provide exclusive access or that may restrict access to a network. When a switch port allocates exclusive time for one of its queues, it will do so in a cyclic fashion. During a portion of that cycle, the queue is opened and only messages in that queue can egress the port. Therefore, messages in other queues will need to wait for that period of exclusive access to expire and for their queues to open. Additionally, as a message traverses a line of switches, it may encounter multiple scheduled windows. Figure 21 shows what happens when multiple messages of differing priorities that are not part of the scheduled stream arrive during an exclusive schedule window. The messages are queued until the schedule expires, after which, they egress according to priority.

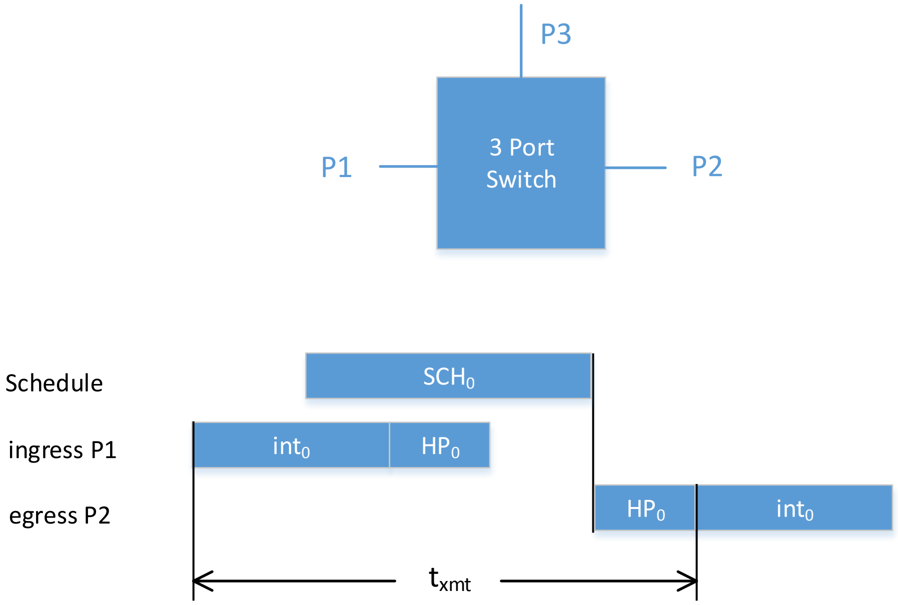


Figure 21: Schedule Interference – blocked Messages

Figure 22 demonstrates what happens if a message is in-progress when the schedule expires and other messages were queued during the schedule window. The queued messages egress the switch while the in-progress message is queued, after which normal priority queuing takes place.

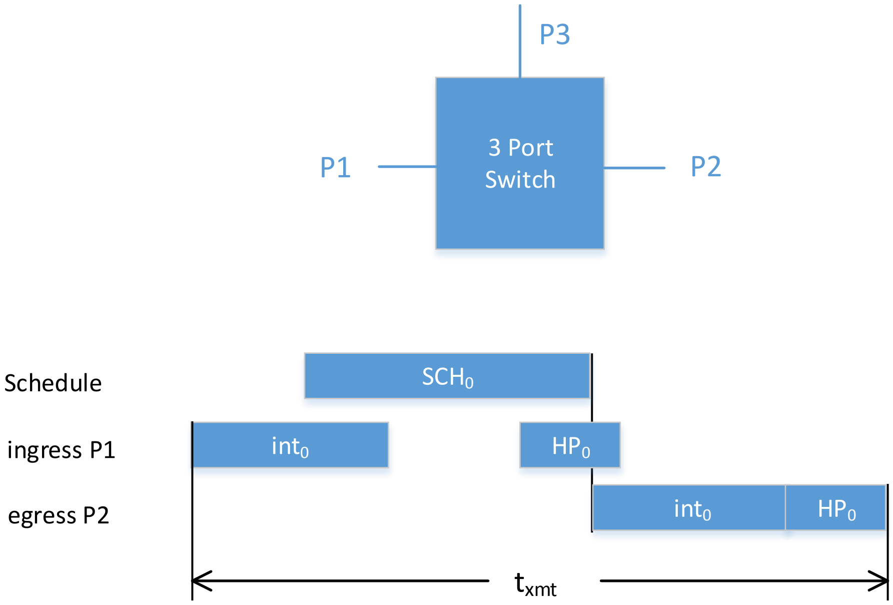


Figure 22: Schedule Interference – In-progress Messages

Equation 11 shows what effect scheduled traffic can have on a message's latency, where t_{sch} is the amount of interference time due to scheduled traffic.

$$t_{xmt} = t_{msg} + n_{sw} * \left(t_{msg} + t_{sw}\right) + t_{sch} \tag{11}$$

Converged Traffic

The previous sections described how messages on a switched Ethernet network could be delayed due to interference. Even with strict priority switching, interference can occur due to lower priority traffic. This section will look at three traffic types prevalent in Industrial Control Systems and demonstrate the effects of converging these traffic types on a single Ethernet network using TSN features. Using the system diagram presented in Figure 1, coexistence of all three traffic types (motion, I/O and event) only occurs on the wires between the controllers and the switch. Line1 and Line2 only contain motion and I/O traffic for their respective controllers.

Motion Traffic

While strict priority could be used for motion traffic, optimum performance is achieved when no interrupting traffic is present during motion communication messages. Hence, this paper will assume that scheduled traffic shaper with exclusive access will be used for motion traffic in a converged network. This shaped window must be synchronized with the motion applications within the various devices so that the messages from the motion applications arrive coincidently with the shaped schedule. Thus, the only variations in network latencies will be due to items like variances in clock synchronization, switch processing times, etc., which are not considered in this paper.

I/O Traffic

This traffic type requires bounded latency where a limited amount of interference can be tolerated. Parameter requirements like period, latency and bandwidth suggest the use of strict priority. To evaluate that latency guaranties are met, a network analysis (e.g., network calculus) is necessary. The network calculus needs to analyze the effect of different data streams of the same traffic type as well as the effect of the Isochronous (motion) traffic.

Frame preemption mechanisms, which were not analyzed in this paper, may optionally be deployed to satisfy the latency requirements. This reduces the interference effect of lower priority traffic.

Event Traffic

This traffic type contains traffic with two application categories (alarm events and control events) which have similar characteristics but differ in latency requirements. While both application categories send the data in a non-periodic manner, some upper bound for the worst-case bandwidth usage needs to be defined in the application. Additionally, two priority levels should be considered, one for the control events and second for alarm events. In a converged network, the priority of the control events compared to the I/O traffic may give precedence to one or the other or put both at the same priority.

Putting It All Together

Figure 23 illustrates the three traffic types independent of convergence, followed by a representation of the result of convergence.

The figure assumes event traffic is a lower priority than I/O traffic and that motion traffic is scheduled. As a result, motion traffic remains unaffected by convergence while some I/O and event messages are delayed. However, TSN provides mechanisms for making sure desired bandwidth and latency requirements are met.

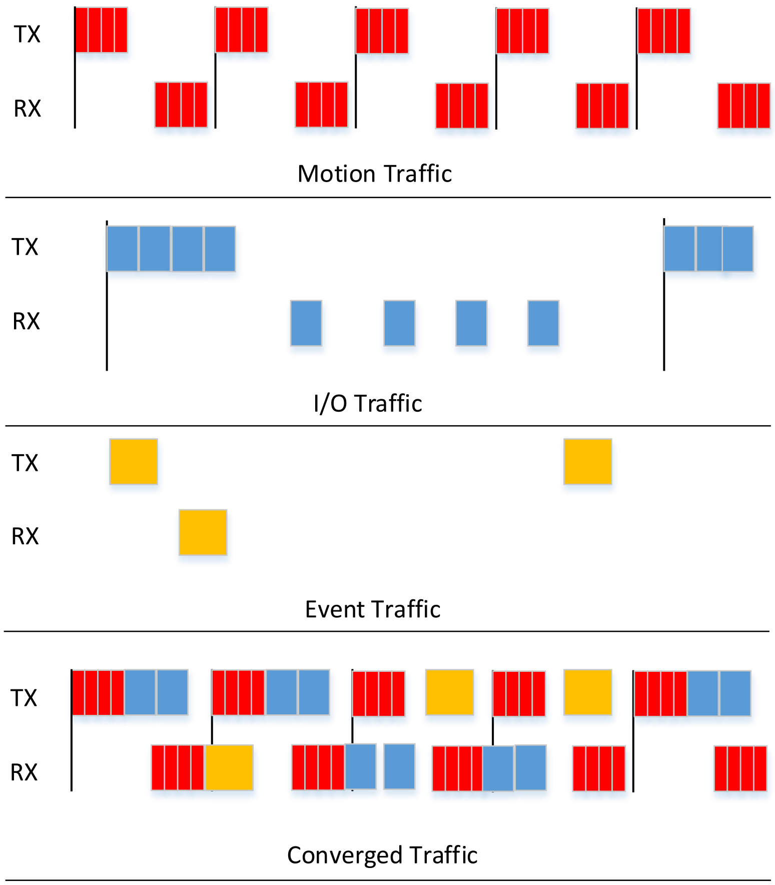


Figure 23: Diagram of Traffic Convergence

Summary and Conclusion

There are many different types of Ethernet traffic present in industrial control systems [4]. Currently, various techniques are used to guarantee delivery of these various traffic types for proper system operation. This paper summarized several Time Sensitive Networking (TSN) features that could be used to converge traffic types on a single network. A small, representative system with three traffic types (motion, I/O and event) was presented to analyze traffic type convergence. It showed how different types of interfering traffic can affect the latency of an Ethernet message, even in the presence of strict priority queuing. An overview of frame preemption was presented, which could be used to reduce latencies for certain traffic types. Cut-through switching was described along with its caveats. Figures and formulas were provided, showing various interfering traffic scenarios. The interference scenarios provided general use cases. Fringe use cases exist and are left as an exercise for the reader. One key takeaway is that, with the exception of scheduled traffic using exclusive gating, TSN cannot prevent interference of one message by another. However, TSN provides the capability, along with network calculus, to determine if messages can be delivered within a certain latency, which is a huge improvement over current Ethernet technology.

References

[1] Gardiner, Eric et al, "*Avnu Alliance™ Best Practices Theory of Operation for TSN-enabled Systems Applied to Industrial Markets*"
[2] Ditzel, George Didier, Paul, 2015 ODVA Industry Conference and 17th Annual Meeting, "*Time Sensitive Network (TSN) Protocols and use in EtherNet/IP Systems*"
[3] https://en.wikipedia.org/wiki/Cut-through_switching, "*Cut-through Switching*," Retrieved 22-Jul-2018
[4] Didier, Paul et al, Time Sensitive Networks for Flexible Manufacturing Testbed, "*Description of Converged Traffic Types*"

****

The ideas, opinions, and recommendations expressed herein are intended to describe concepts of the author(s) for the possible use of ODVA technologies and do not reflect the ideas, opinions, and recommendation of ODVA per se. Because ODVA technologies may be applied in many diverse situations and in conjunction with products and systems from multiple vendors, the reader and those responsible for specifying ODVA networks must determine for themselves the suitability and the suitability of ideas, opinions, and recommendations expressed herein for intended use.