Evaluating TSN Preemption for Cyclic EtherNet/IP Communication

Agenda

- Ethernet switch overview
- Simulating Ethernet switch behavior, including preemption
- Simulation example and results
- Conclusions

Ethernet Switch Behavior

- Quality of Service (QoS)
- Store-and-forward switching
- Cut-through switching
- Physical interface (PHY)
- Preemption

QoS Strict Priority

- Default queuing mechanism for Ethernet bridges (switches)
- Higher numbered queues have priority over lower numbered queues
- In general, highest numbered queue with packet is transmitted next
- Shaping mechanisms can affect priority queuing

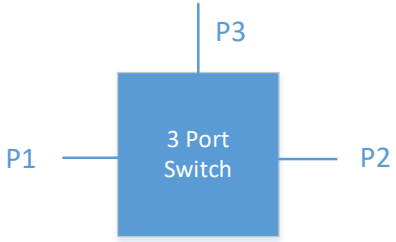
Cut-through Switching

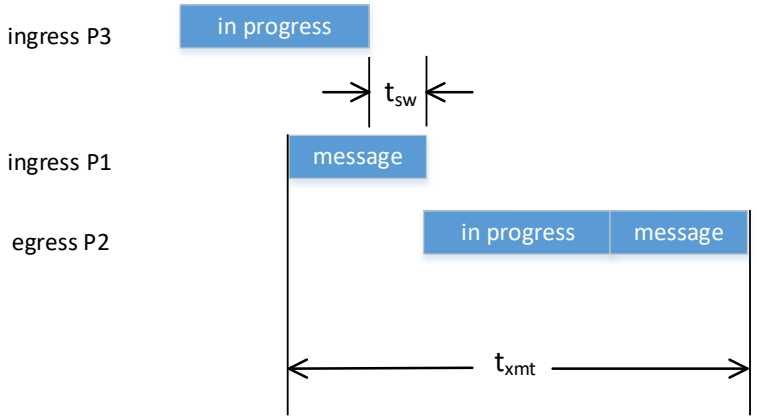
- Advantages
  - Switch begins forwarding before fully receiving messages
  - Offers lower latencies over store-and-forward switching
- Caveats
  - Behavior not specified in standards
  - Propagates corrupted messages
- Congestion and port speed differences cause fallback to store-and-forward behavior

Physical Interface (PHY)

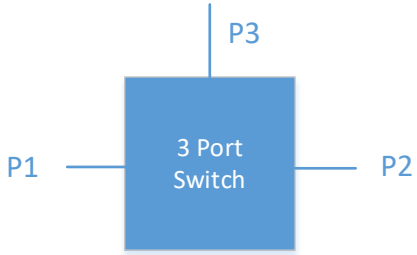
- performs encoding, transmission, reception, decoding and provides galvanic isolation
- translates logical communications into hardware-specific operations
- Transmitter Latency may be different compared to receiver latency
- Simulator uses single average, since transmitter always coupled to a receiver
- conversion increases packet latencies

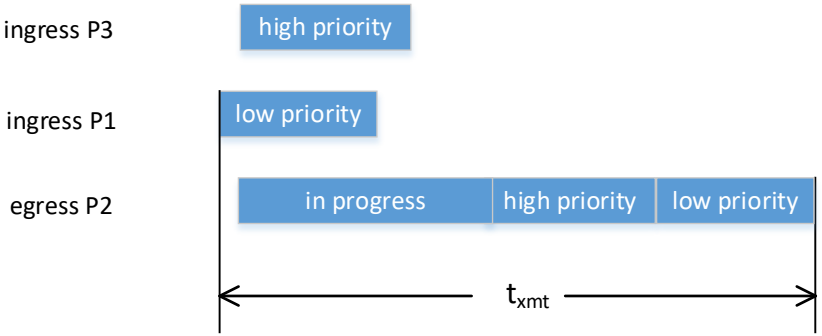
Interference





Interference due to in-progress message

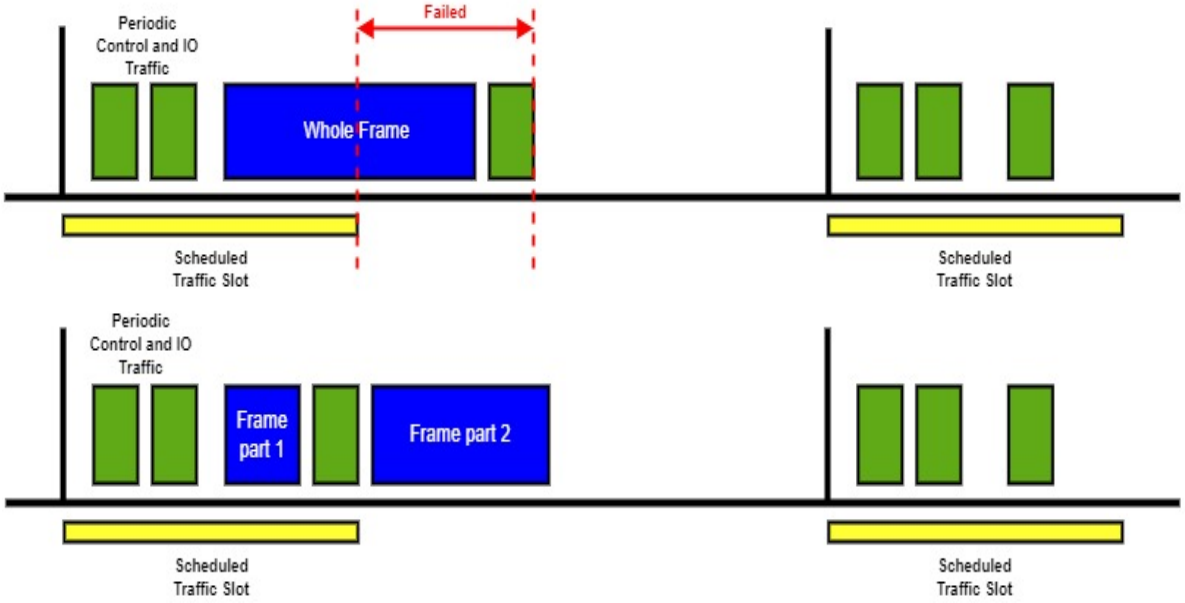




High-priority Interference

Frame Preemption and Interspersing Express Traffic

- Ability to interrupt a low priority (non-express) packet with a higher priority (express) during transmission



Simulator Overview

- Written in Python programming language
- Focus on preemption, QoS, Cut-through and store-and-forward switching
- Main classes to represent
  - ethernetSwitch
  - ethernetCable
  - ethernetEndpoint
  - ethernetPort
  - ethernetPacket and ethernetPacketState
  - egressMonitor
- Bandwidth analyzer

Setting Up the Simulator

- Modify main python file to define simulation
  - Define packet traffic patterns
  - Define endpoints, endpoint addresses, and switches
  - Define connections
  - Configure egress monitors
  - Queue packets into endpoints
- Run simulation
- Produce results

Simulator Variables

- QueueingTime, processingTime, preemptionTime, cutThroughTime
- Preempt
- cutThrough
- cutThroughOctets
- phyDelay
- simulationTime
- simulationStep
- portSpeed
- cableLength

Ethernet Packets

- Simulate the movement of packets from one class to another
- Static portion in ethernetPacket class (size, priority, destination address, Dynamic portion stored in EthernetPacketState class (start and end time, network time, in and out ports, state, and so on)
- Static portion only changes if preempted
- Same packet can be in several state instances during simulation (egressing endpoint1 while ingressing switchA, and so on)

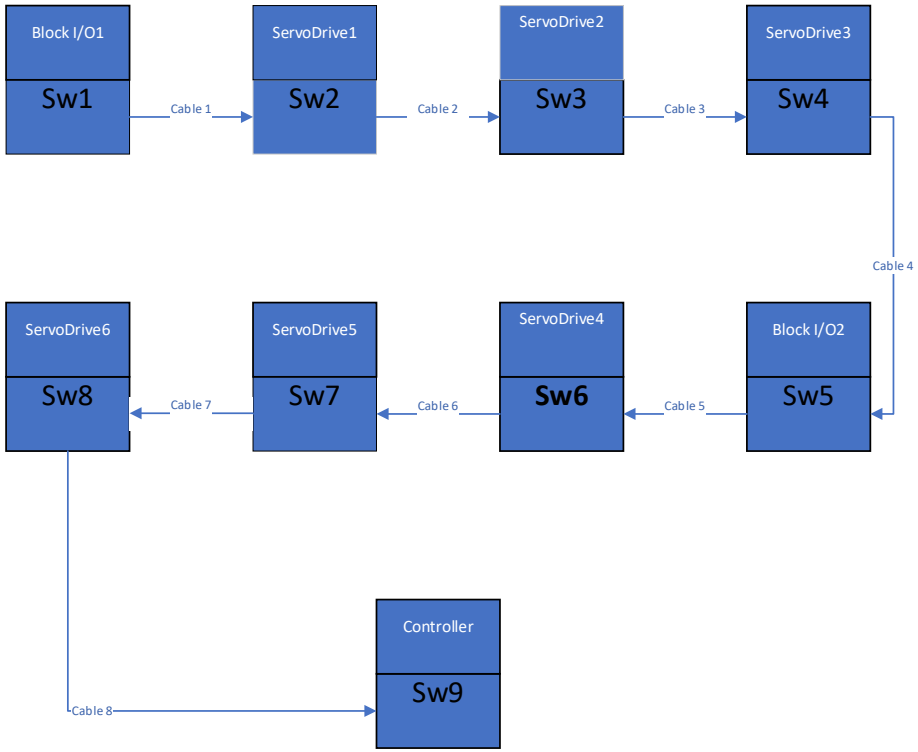
Simulation Operation

- Time slice, increment time and
- Loop through all class instances, evaluates packet state instances to determine if a change of state is needed (for example, packet completes ingress, move to queue pending state)
- Update egress monitors
- Upon completion, perform bandwidth analysis and print results

Simulation Example

- Line topology
  - 6 drives, 16 octets payload, 1 ms cycle
  - 2 rack I/O, 251 octets payload, 4 ms cycle
  - 1 controller
- Device to controller only
- All devices send data simultaneously at beginning of cycle
- 30 msec simulation duration

Simulation Network Diagram



Device Characteristics

Device Type	Cycle time (us)	num dev	payload	Pkt Size	Priority
Servo Drive	1000	6	16	90	7
Block I/O	4000	2	251	325	6

Baseline Simulation Results

start time	gap time	Latency	Pkt Name
19680	19680	27520	ServoDrive6-pkt1
30020	1540	37860	ServoDrive5-pkt1
40360	1540	48200	ServoDrive4-pkt1
61040	11880	68880	ServoDrive3-pkt1
145040	75200	171680	BlockI/O2-pkt1
172640	0	180480	ServoDrive2-pkt1
181440	0	189280	ServoDrive1-pkt1
261260	71020	287900	BlockI/O1-pkt1
1019680	730820	27520	ServoDrive6-pkt2
1030020	1540	37860	ServoDrive5-pkt2
1040360	1540	48200	ServoDrive4-pkt2
1061040	11880	68880	ServoDrive3-pkt2
1071380	1540	79220	ServoDrive2-pkt2
1081720	1540	89560	ServoDrive1-pkt2

Baseline Min & Max Latencies

Pattern	Min	max
ServoDrive1-pkt*	89560	189280
BlockI/O1-pkt*	287900	287900

Max Latencies with Interfering Packets

Pattern	5-123	50-123	1-750	1-1500
ServoDrive1-pkt*	199540	216180	191520	190360
BlockI/O1-pkt*	318660	343620	297940	291200

Bandwidth Analysis – No Interference

Start Time	End Time	SW1	SW2	SW3	SW9
0	1	2.76	3.64	4.52	10.8
1	2	0	0.88	1.76	5.28
2	3	0	0.88	1.76	5.28
3	4	0	0.88	1.76	5.28

Bandwidth Analysis – with Interference

Start Time	End Time	SW1	SW2	SW3	SW9
0	1	59.96	98.802	98.912	95.998
1	2	0	77.318	100	100
2	3	0	58.08	100	100
3	4	0	58.08	100	100

Simulation Results Overview

- Even without interference, drive packet latency affected when in same cycle as I/O traffic
- Packets arriving at controller not in same order as endpoints when cycles coincide
- Nominal bandwidth utilization when no interfering traffic present
- Increasing quantity of interfering packets per cycle and longer simulation times increase latencies towards theoretical prediction

Conclusions

- Presented simulation as one method for analyzing cyclic EtherNet/IP traffic in a network.
- Simulator demo, line topology, two types of traffic at different priorities and cyclic rates.
- Observations not intuitively obvious; that even without interference, the packets received at the controller are not in the order of the devices on the network
- Attempting to analyze using Excel would be extremely complex, especially if one wanted to examine variance such as store-and-forward vs. cut-through, preemption vs. no preemption, packet size and priority variations, and so on.
    - ESS allows analyzing complex network topologies and cyclic data patterns by changing main program file and re-running the simulation.

2022 ODVA

INDUSTRY CONFERENCE AND 21ST ANNUAL MEETING