Bus Systems

Avionic Bus Systems

Prof. Dr. Reinhard Gotzhein, Dr. Thomas Kuhn

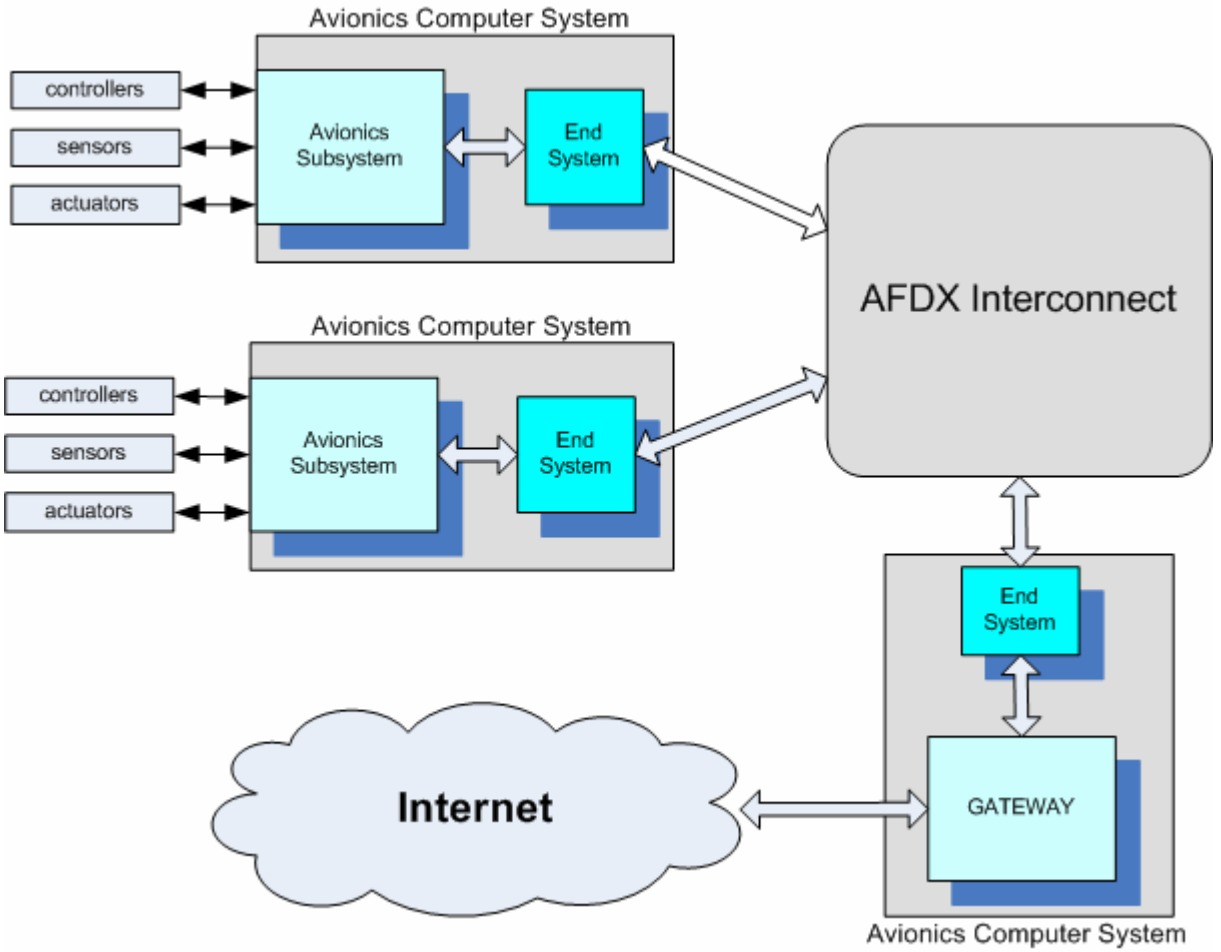
Avionic Bus Systems

Learning objectives

- Understand principles of AFDX networks
- Understand how Real-Time guarantees are achieved in AFDX networks
- Understand differences between AFDX and switched Ethernet networks

AFDX

- AFDX Systems

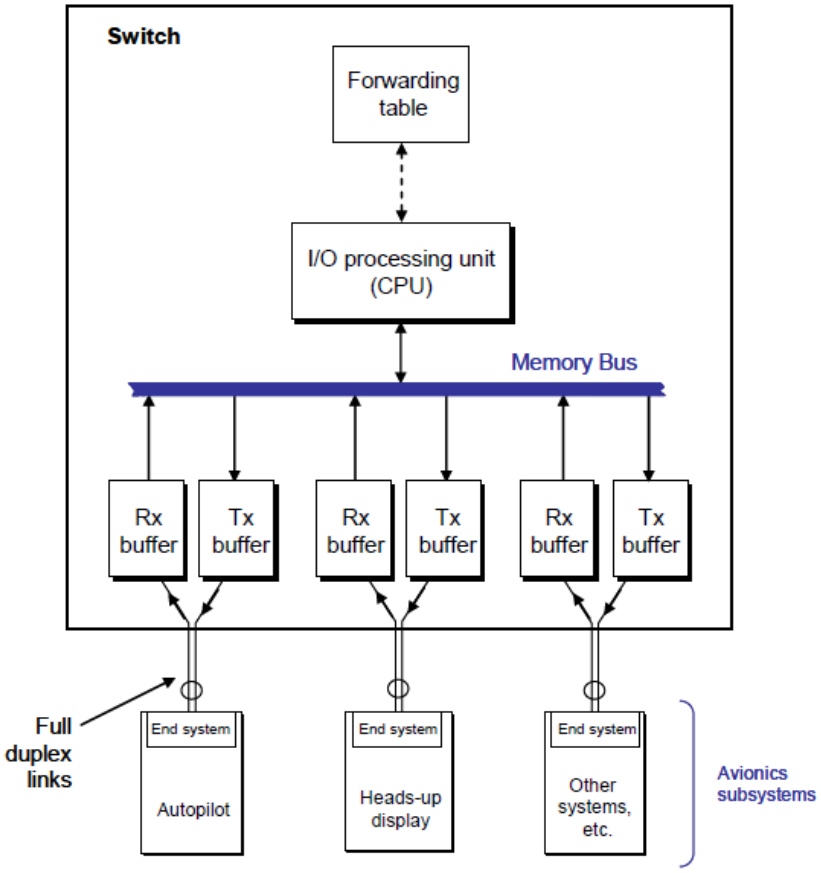


AFDX

- AFDX System Components
  - **Avionics Subsystem:** The traditional Avionics Subsystems on board an aircraft, such as the flight control computer, global positioning system, tire pressure monitoring system, etc. An *Avionics Computer System* provides a computational environment for the Avionics Subsystems. Each Avionics Computer System contains an embedded End System that connects the Avionics Subsystems to an AFDX Interconnect.
  - **AFDX End System (End System):** Provides an "interface" between the Avionics Subsystems and the AFDX Interconnect. For each Avionics Subsystem the End System interface to guarantee a secure and reliable data interchange with other Avionics Subsystems. This interface exports an application program interface *(*API) to the various Avionics Subsystems, enabling them to communicate with each other through a simple message interface.
  - **AFDX Interconnect:** A full-duplex, switched Ethernet interconnect. It generally consists of a network of switches that forward Ethernet frames to their appropriate destinations. This switched Ethernet technology is a departure from the traditional ARINC 429 unidirectional, point-to-point technology and the MIL-STD-1553 bus technology.

AFDX

- Full-Duplex switched Ethernet (AFDX switch)
  - Requires two pairs CAT5 TPI copper wire
  - Store-and-forward architecture
- Nodes are only connected via switches
  - Star topologies
- Individual TX/RX lines to nodes
  - Full-duplex
  - Collsions are prevented
  - Random Backoff cannot occur
- TX/RX buffers in switch
  - FIFO buffers
  - CPU moves frames from RX buffers to TX buffers based on forwarding table



AFDX

Achieving determinstic traffic guarantees with ethernet physical layers

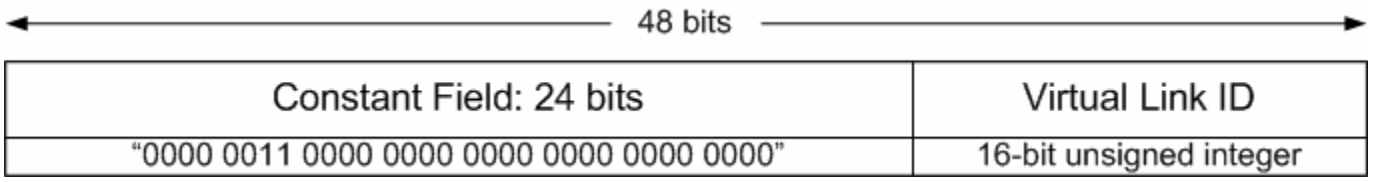
- Buffers in switches
  - Buffers may overflow if not sufficiently dimensionated
  - Maximum arrival rate of frames must be known
- Jitter introduced by switches
  - (Random) delay introduced by one frame waiting for transmission of other frames
  - Extend of introduced jitter must be controlled to provide deterministic traffic guarantees

AFDX

Ethernet Destination addresses

– Traditional Ethernet networks: 48 Bit MAC address

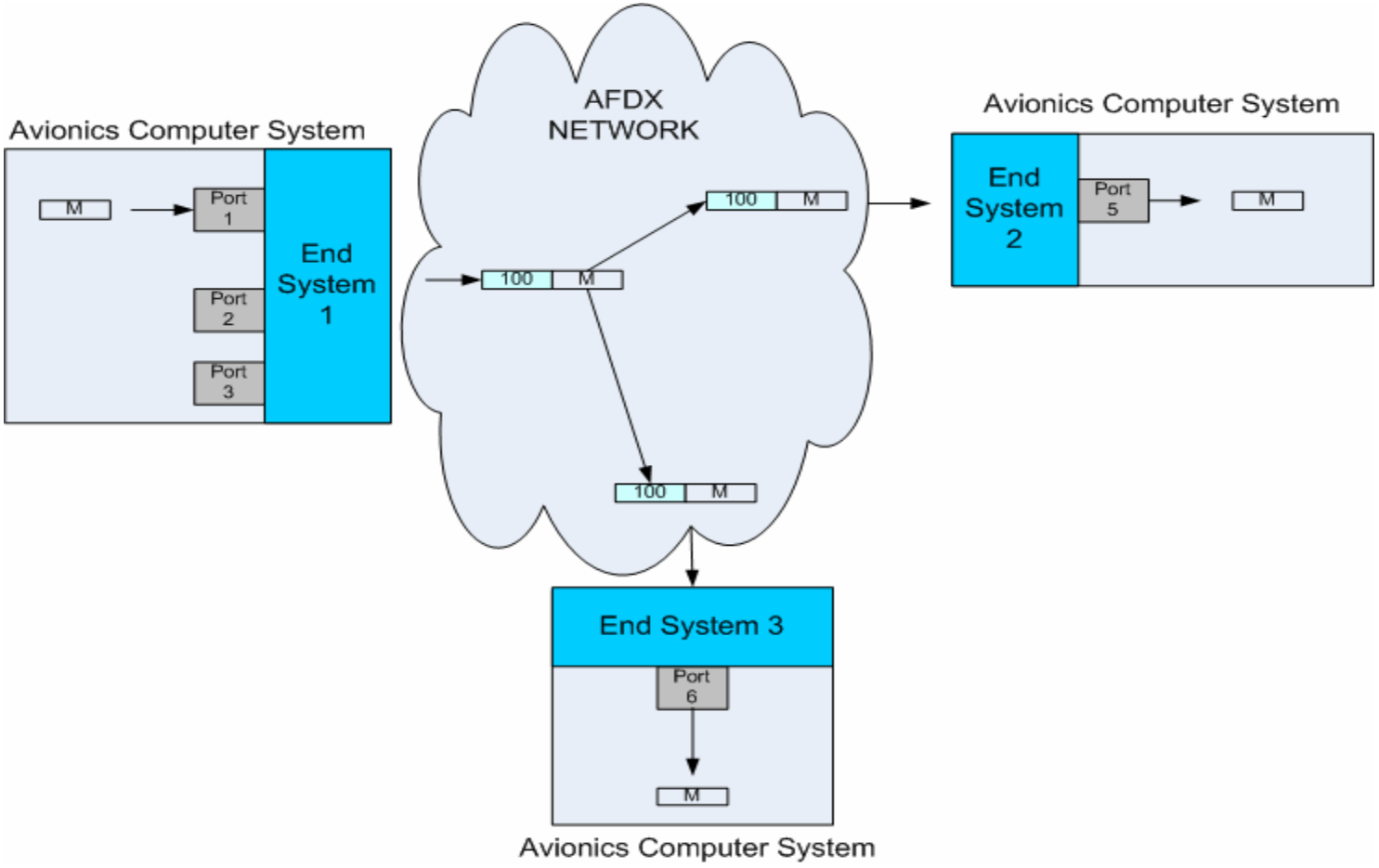
AFDX Networks:



– Using Virtual Link ID instead of MAC address
– AFDX Frames with a given virtual Link must originate at exactly one end system
– AFDX switches deliver frames with given logical link ID to predetermined outputs
– Statically configured
– Enables cascading of AFDX switches

AFDX

AFDX frame routing example



AFDX

AFDX Frame routing example

- Message M being sent to Port 1 by the Avionics subsystem.
- End System 1 encapsulates the message in an Ethernet frame and sends the Ethernet frame to the AFDX Switched Network on Virtual Link 100 (the Ethernet destination address specifies VLID 100).
- The forwarding tables in the network switches are configured to deliver the Ethernet frame to both End System 2 and End System 3.
- The End Systems that receive the Ethernet frame are configured so that they are able to determine the destination ports for the message contained in the Ethernet frame.
- In the case shown the message is delivered by End System 2 to port 5 and by End System 3 to port 6.

AFDX

AFDX Ports

- AFDX ports are logical communication endpoints of a AFDX network
- Virtual links may carry data from more than one port at the same time

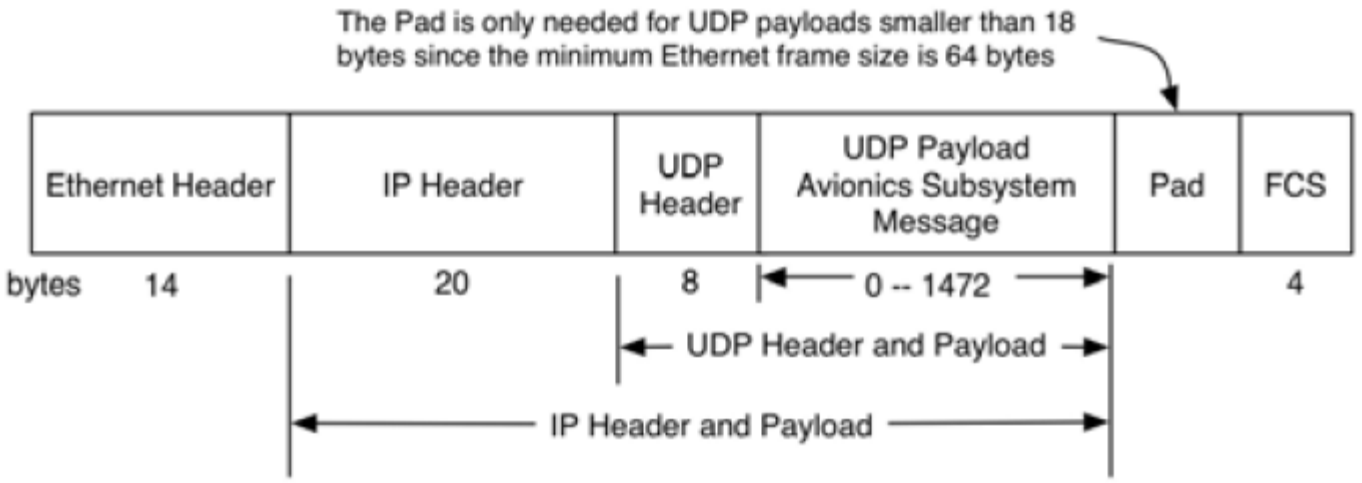
AFDX distinguishes the following port types

- Sampling ports
  - Buffers one value
  - New values overwrite old values
  - Reading a value does not remove it
  - Requires freshness indicator to indicate whether a value is recent or not
- Queuing ports
  - Provides a fixed queue size for incoming messages
  - New messages are appended to queue
  - Read messages are removed from queue

AFDX

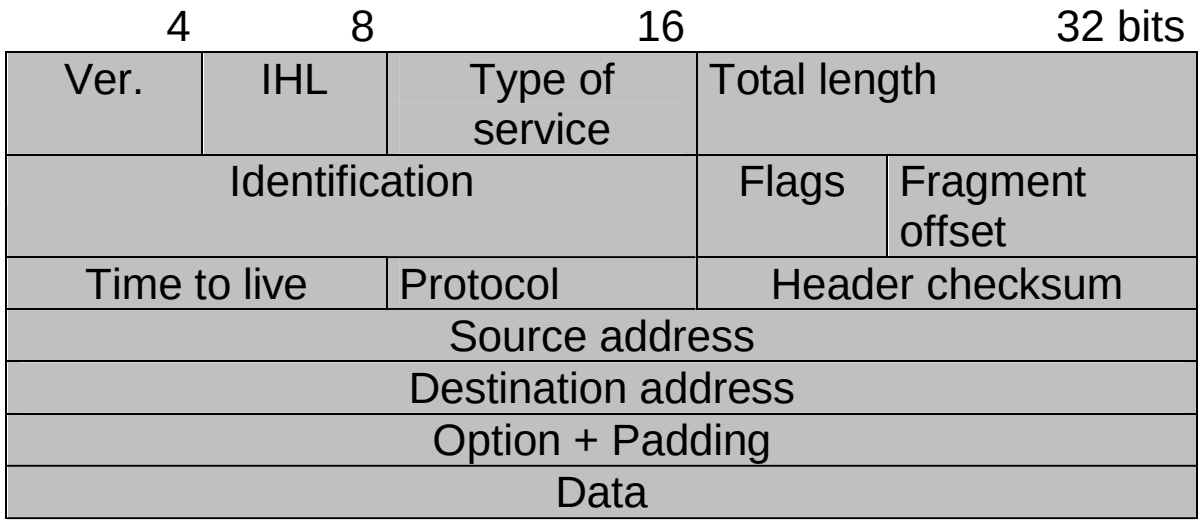
AFDX Frame format

- Based on UDP/IP protocol headers



AFDX

AFDX Frame format – IP Header



AFDX

AFDX Frame format – IP header

- **Version: (Not used in AFDX)**
  - Version field indicates the format of the Internet header.

- **IHL (Not used in AFDX)**
  - Internet header length is the length of the Internet header in 32-bit words. Points to the beginning of the data. The minimum value for a correct header is 5.

- **Type of service (Not used in AFDX)**
  - Indicates the quality of service desired. Networks may offer service precedence, meaning that they accept traffic only above a certain precedence at times of high load. There is a three-way trade-off between low delay, high reliability and high throughput. Unused for AFDX.

AFDX

AFDX Frame format – IP header

- **Time to live (Not used in AFDX)**
  - Indicates the maximum time the datagram is allowed to remain in the Internet system. If this field contains the value zero, the datagram must be destroyed.

- **Protocol**
  - Indicates the next level protocol used in the data portion of the Internet datagram. AFDX only permits ICMP (1), TCP (6), UDP (17)

- **Header checksum**
  - A checksum on the header only. Since some header fields change, e.g., Time To Live, this is recomputed and verified at each point that the Internet header is processed.

AFDX

AFDX Frame format – IP header

- **AFDX IP addressing**
  - AFDX frames may be addressed to one particular receiver of a virtual link (unicast) or to all nodes that are connected to a virtual link (multicast)

IP Unicast Addressing Format 32 bits							
Class A 1 bit	Private IP address 7 bits	Network ID 8 bits		Equipment ID 8 bits		Partition ID 8 bits	
"0"	"0001010"	Constant field "0000"	Domain ID 4 bits	Side ID 3 bits	Location ID 5 bits	Constant field 3 bits	Partition ID 5 bits

- Domain: Domain ID of sender/receiver
- Side ID: Side where Equipment is located (left/right/front…)
- Location ID: Location specifier within Side
- Partition ID: Partition within Equipment
- Assigned by AFDX network administrator

AFDX

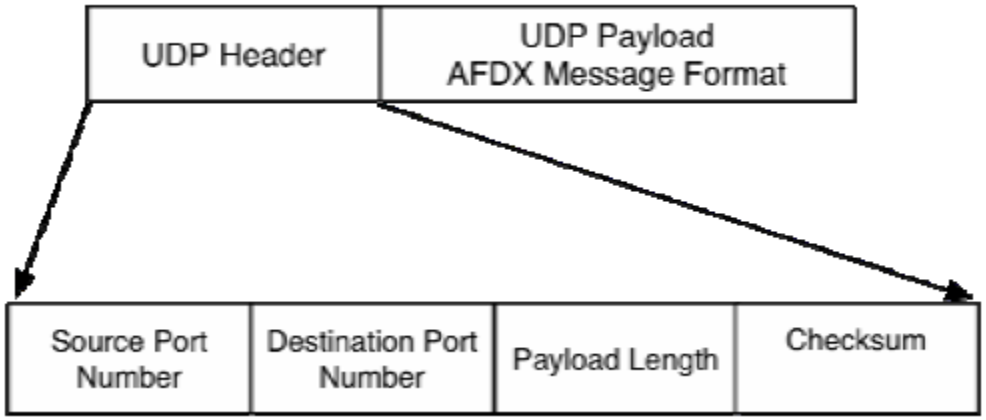
AFDX Frame format – IP header

- **AFDX IP addressing**
  - AFDX frames may be addressed to one particular receiver of a virtual link (unicast) or to all nodes that are connected to a virtual link (multicast)

IP Multicast Identifier Addressing Format 32 bits		
4 bits	28 bits	
Class D	Constant field12 bits =	Virtual Link Identifier Same value as “Virtual Link identifier “ used in the MAC address destination of the VL
"1110"	"0000 1110 0000"	16 bits

AFDX

AFDX Frame format – UDP header



- Source ports and destination ports address AFDX applications
  - Mapped to AFDX ports
  - Should be high/not reserved in case of the presence of a gateway

AFDX

AFDX Frame format – Ethernet header

Ethernet format 84-1538 bytes (with mainly used UDP protocol)

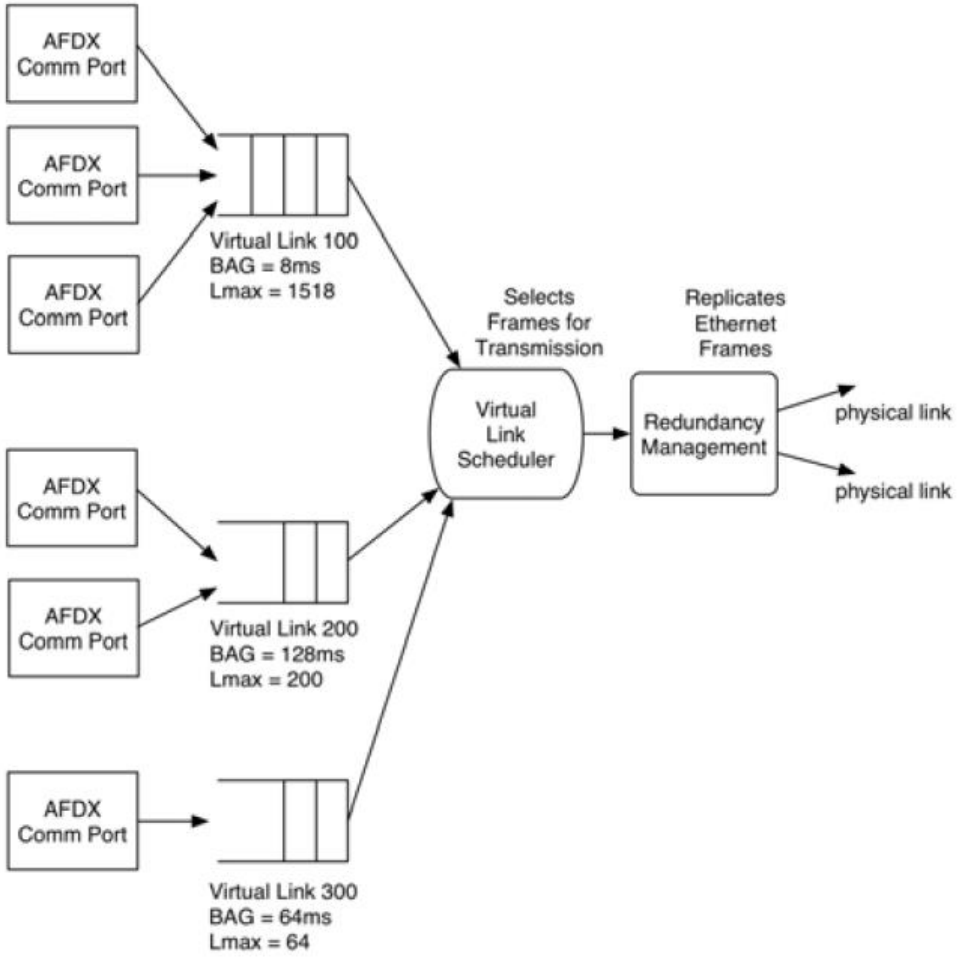
7 bytes	1 byte	6 bytes	6 bytes	2 bytes	Ethernet payload 46-1500 bytes				4 bytes	>12 bytes (soit 0.96us)
Preamble 10101010...	start frame delimiter 10101011	@ MAC dest.	@ MAC source	Ethernet type IP V4 0x0800	IP header 20 bytes	UDP header 8 bytes	AFDX payload 17-1471 bytes	Tag (SN) 1 byte	Frame Check Sequence (CRC)	Inter Frame Gap

- Both sender & receiver MAC addresses contain Virtual Link identifiers in case of AFDX frames

AFDX

AFDX Frame format – Virtual Link Scheduling

- Isolates virtual links
  - BAG (Bandwidth Allocation Gap)
    - Powers of 2 between 1 to 128 milliseconds
    - Defines minimum interval between frames on this virtual link
  - LMax
    - Defines maximum Ethernet frame size on this link (bytes)



AFDX

AFDX Frame format – Virtual Link Scheduling

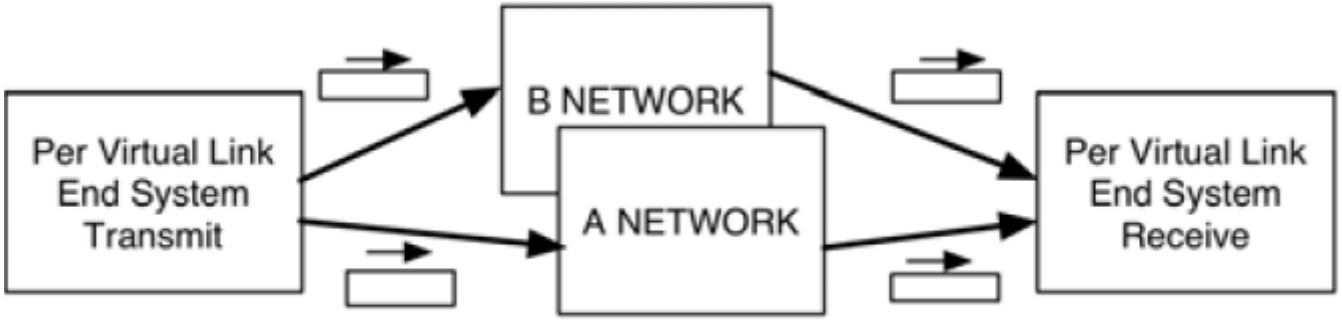
- **BAG**
  - Should be set to a lower value than the period between message transmissions on the virtual link. E.g. when period between message transmissions is 13ms, a BAG value of 8 is to be selected
  - Source end system is required to enforce BAG restrictions

BAG milliseconds	Hz
1	1000
2	500
4	250
8	125
16	62.5
32	31.25
64	15.625
128	7.8125

AFDX

AFDX Redundancy

- Each frame is transmitted over two physically independent networks



A and B Networks

AFDX

AFDX End-to-End delay

- AFDX permits predictable upper bounds for delays
  - Big advantage over standard Ethernet
  - Achieved by assuring rate constraints, i.e. minimum transmission rate for each virtual link (VL)

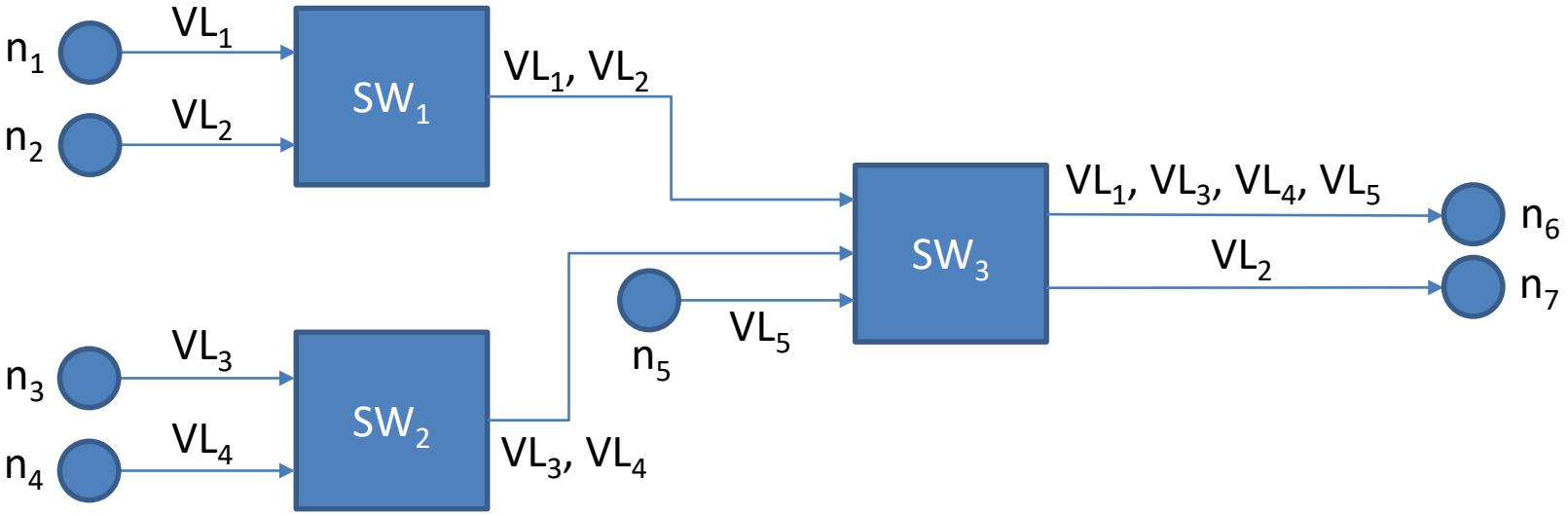
- AFDX does not guarantee transmission times
  - Planning concurrent streams is therefore a challenge for the system engineer as VL transmissions influence each other

- Predicting transmission times requires calculation of worst-case end-to-end transmission delays

AFDX

AFDX End-to-End delay

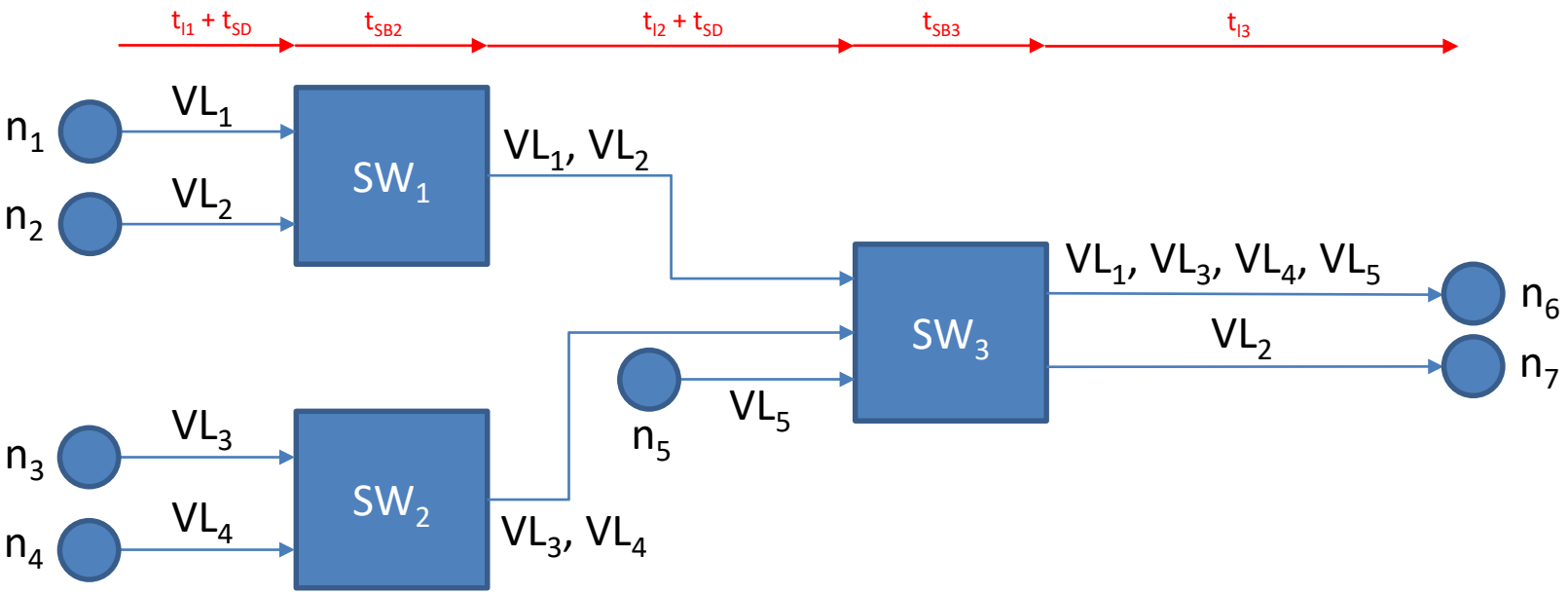
- Example: When will frame in VL_3 from node n_3 reach destination node n_6?



AFDX

AFDX End-to-End delay

- Calculation using the trajectory approach
  - t_{lx} Transmission time on link segment x
  - t_{SBx} Buffering time at switch X
  - t_{SD} Switch processing delay



AFDX

AFDX End-to-End delay

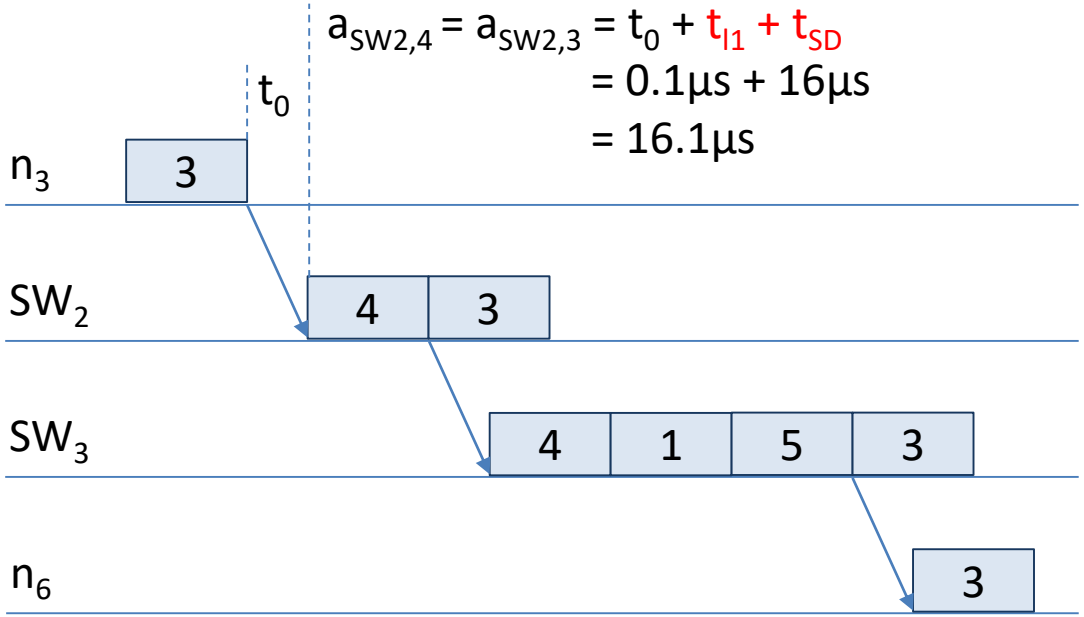
- Evaluation of timing
  - We assume worst case timing at critical instant for transmitted frame
  - All possibly concurrent frames are queued in front of evaluated frame
  - Evaluation needs to be performed for each arrival node

- Remember
  - Switches use Store-and-Forward strategy

AFDX

AFDX End-to-End delay

- Evaluation of timing



$a_{SW2,4}$: Arrival of frame 4 at SW_2
$a_{SW2,3}$: Arrival of frame 3 at SW_2

Both frames arrive at same time
→ Worst case for 3: 4 is processed first

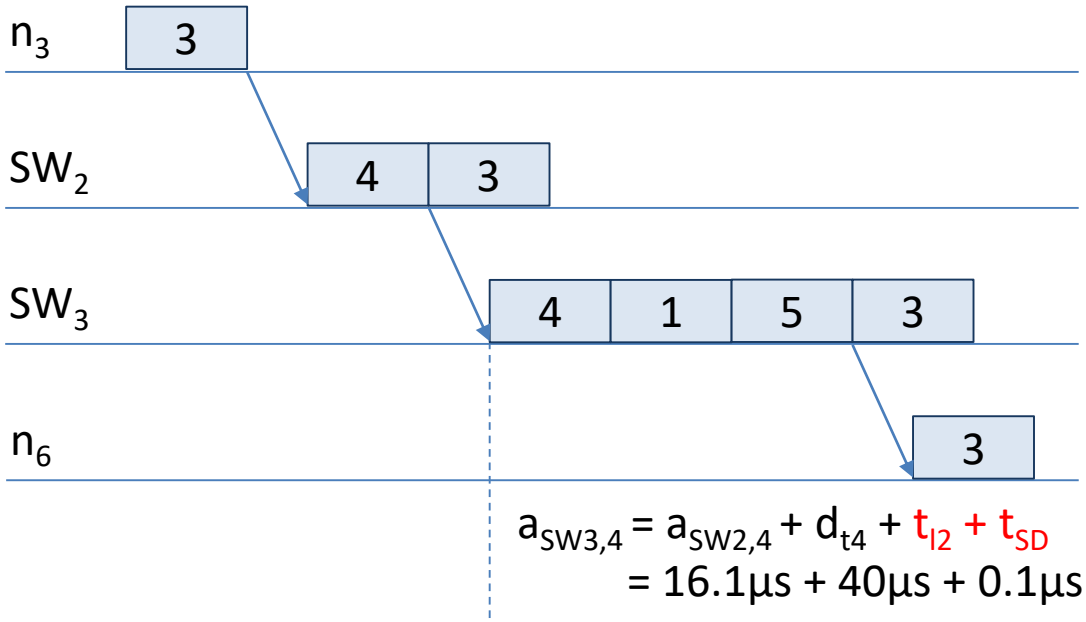
t_{l1}: Bit propagation time on link 1
→ We assume 0.1µs

t_{SD}: Switch delay
→ We assume 16µs per switch

AFDX

AFDX End-to-End delay

- Evaluation of timing
  - Switches use store and forward
  - Frame 4 transmission starts after it has been fully received



$a_{SW3,4}$: Arrival of frame 4 at SW_3
$a_{SW3,3}$: Arrival of frame 3 at SW_3

d_{t4}: Worst case transmission duration for frame 4

→ L_{max} = 500Bytes
→ Speed = 100 Mbit/s

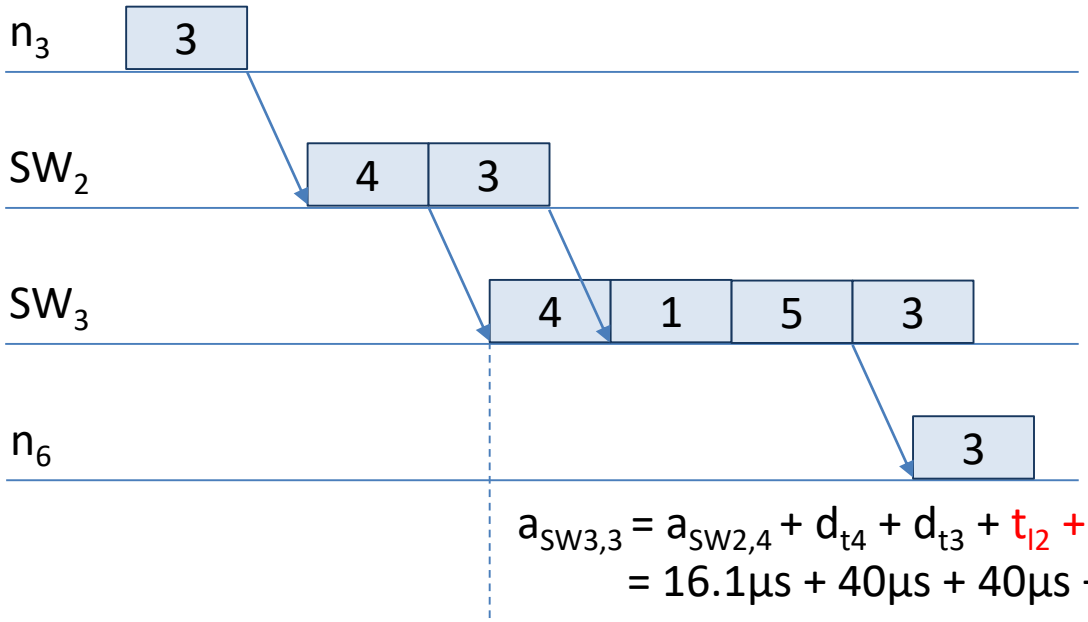
$$d_{t4} = 500 * 8 * \frac{1}{100\,000\,000 s} = 40 \mu s$$

$a_{SW3,4} = a_{SW2,4} + d_{t4} + t_{l2} + t_{SD}$
$= 16.1\mu s + 40\mu s + 0.1\mu s + 16\mu s$
$= 72.2\mu s$

AFDX

AFDX End-to-End delay

- Evaluation of timing
  - Frame 3 transmission starts after transmission of frame 4 finished
  - Switch 2 receives Frame 3 in parallel to frame 4



$a_{SW3,4}$: Arrival of frame 4 at SW_3
$a_{SW3,3}$: Arrival of frame 3 at SW_3

d_{t3}: Worst case transmission duration for frame 3
→ L_{max} = 500Bytes
→ Speed = 100 Mbit/s

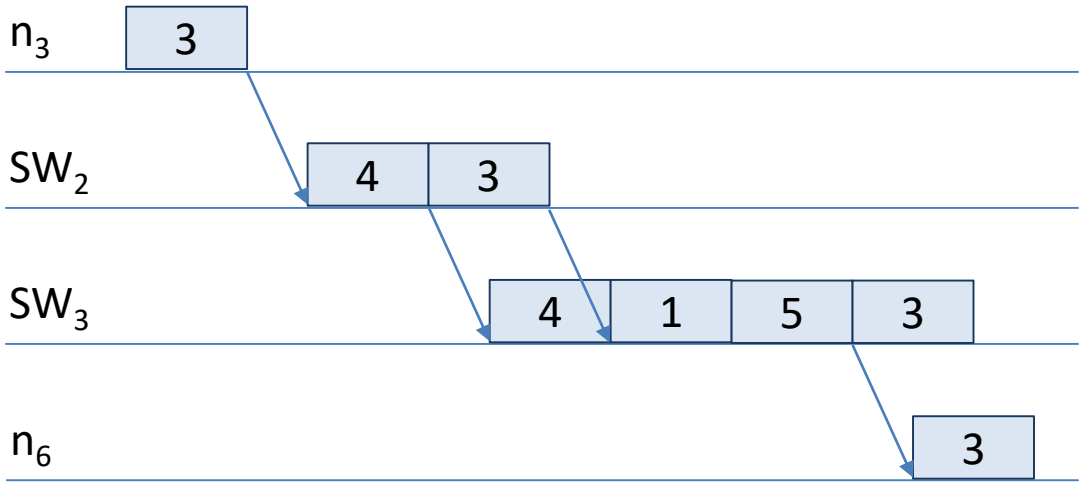
$$d_{t3} = 500 * 8 * \frac{1}{100\,000\,000 s} = 40 \mu s$$

$a_{SW3,3} = a_{SW2,4} + d_{t4} + d_{t3} + t_{l2} + t_{SD}$
$= 16.1\mu s + 40\mu s + 40\mu s + 0.1\mu s + 16\mu s$
$= 112.2\mu s$

AFDX

AFDX End-to-End delay

- Evaluation of timing
  - Switch buffering time t_{SB2} for frame 3 at switch 2 is 40µs
  - The time that frame 3 has to wait for transmission while frame 4 is transmitted



AFDX

AFDX End-to-End delay

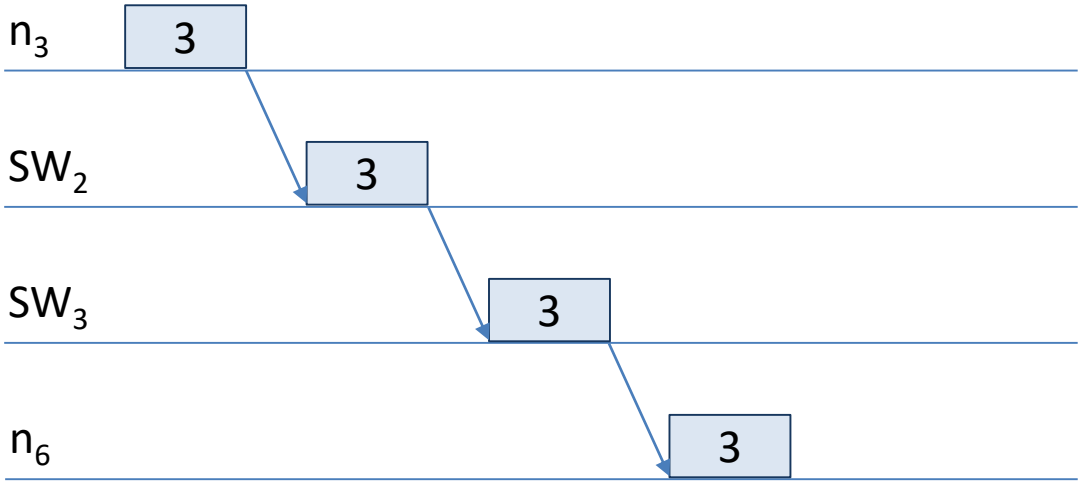
- Evaluation of timing
  - Switch buffering time at switch 3 is $d_{t4} + d_{t1} + d_{t5} + d_{t3} = 160\mu s$
  - Arrival at node 6 $a_{n6,3}$ is $a_{n6,3} = a_{SW3,4} + d_{t4} + d_{t1} + d_{t5} + d_{t3} + t_{l3} = 232.3\mu s$
  - We assume again same maximum frame length for frames 1 and 5

  - Trajectory approach only needs to calculate arrival time of first frame
    - Calculation of $a_{SW3,3}$ was not necessary for evaluation
    - But helps with understanding of approach

AFDX

AFDX End-to-End delay

- Second example: only one virtual link in system
  - Again, mind that switches use store-and-forward strategy
  - $a_{n6} = t_0 + t_{l1} + t_{SD} + d_{t3} + t_{l2} + t_{SD} + d_{t3} + t_{l3} = 112.3\ \mu s$



AFDX

AFDX End-to-End delay

- Evaluation of timing
  - AFDX timing evaluation depends on knowledge about frame arrival times
    - First possible time to transmit
  - Actual frame transmission time is worst case transmission time
    - All possible other frames are transmitted first

- BAG for VLs may differ
  - The set of worst case frames that must be considered change in this case for each frame instance
  - Pessimistic assumption: Assume longest set of preceding frames
  - Better approach: Calculate worst case duration for each frame instance until preceding frame instances repeat