AFDX® / ARINC 664

Tutorial

techSAT

mastering
realtime
complexity

Imprint

Document Title	AFDX® / ARINC 664 Tutorial
Document ID	700008_TUT-AFDX-EN_1000
Issue	1000
Classification	Public
Project	AFDX® / ARINC 664
Realeased by	TechSAT GmbH, Poing
Release Date	29/08/2008

Copyright Notice

Trademark Notice

techSAT
mastering
realtime
complexity

Contents

List of Figures

List of Tables

List of Formulas

List of Abbreviations

Abbreviation	Text
ADN	Aircraft Data Network
AFDX	Avionics Full Duplex Switched Ethernet
ARINC	Aeronautical Radio Incorporated
BAG	Bandwidth Allocation Gap. BAG is a property of a VL and is given in ms
BER	Bit Error Ratio - The number of erroneous bits to the total bit number of bits
COTS	Commercial Of The Shelf
CRC	Cyclical Redundancy Checking. A data integrity detection method
ES	End System - LRU or part of an LRU implementing the network communication interface
FCS	Frame Check Sequence
FIFO	First In First Out
IEEE802.3	Ethernet protocol standard from "Institute of Electrical & Electronics Engineers"
IFG	Inter Frame Gap
IP	Internet Protocol
Jitter	Network transmit latency caused by data contention
Latency	Network transmit latency caused by software and hardware layers in the ES
LRU	Line Replaceable Unit
MAC	Media Access Controller
MTC	Message Transmit Cycle. MTC is a property of a message and is given in ms
MTU	Maximum Transfer Unit. MTU is a property of a VL and is given in bytes
PL	Message payload data. PL is a property of a message and is given in bytes
QoS	Quality of Service
RFC	Request for Communication. Standard for protocols on top of IEEE802.3
RSN	Redundancy Sequence Number
SAP	Service Access Point
SFD	Start Frame Delimiter
TCP	Transmission Control Protocol
TFTP	Trivial File Transfer Protocol
UDP	User Datagram Protocol
VL	Virtual Link. A virtual or logical communication channel

Scope

The purpose of this tutorial is to give a description of the Avionics Full Duplex Switched Ethernet (AFDX®) which is a communication standard for an Aircraft Data Network (ADN). The description will provide the reader with an overview of the fundamentals of the AFDX network architecture, network protocol, network communication concept and network reliability.

The tutorial is based on the standard ARINC 664 which is an AFDX standard defined by Aeronautical Radio, Incorporated (ARINC) and Airbus. Meanwhile Boeing has defined a special version of AFDX which is based on the ARINC 664 standard and is backwards compatible with this. However, the specialties of the Boeing standard are not described herein.

It is assumed that the reader, as a minimum, is familiar with the principles of data communication. In particular, knowledge of the Ethernet communication standard is a definite advantage to understand the contents of this tutorial.

Overview

Aircraft Data Network (ADN) Characteristics

The most important characteristics of an ADN are Quality of Service (QoS), available bandwidth, weight and the cost of its development and deployment.

Various ADN attributes such as bandwidth guarantee, jitter, transmit latency and Bit Error Ratio (BER) determine the QoS. A guaranteed bandwidth, limited jitter, upper bounded transmit latency and a low BER (typically 10^{-12}, i.e. one error bit in a trillion) are imperative attributes of a reliable and deterministic ADN.

New generation aircraft such as the A380, A350, B787 and A400M are required to feature more sophisticated functions than previous aircraft generations. For reasons of weight savings, less required space as well as reduced maintenance costs it is desired to implement as much functionality as possible in avionics. This leads to more complex avionics systems that need to process more data than legacy systems, and consequently a need for ADNs with higher available bandwidths arises.

Another important attribute of an ADN is the required wiring. The less wiring required, the less its weight which leads to a more fuel efficient aircraft.

Finally, the cost of an ADN's development and deployment is an important factor as well. Traditionally, ADNs have been based on new technologies specifically developed for the purpose, thereby making the ADN development very expensive. Meanwhile, it has become much more desirable to utilize already existing commercial technologies more or less adapted to the requirements of ADNs. The purpose of this is not only to benefit from the lower costs of COTS equipment, but also to take advantage of the fact that COTS equipment is already field-proven.

Emerging of AFDX

Prior to the Airbus 380 Aircraft, the three main ADNs were ARINC 429, MIL-STD-1553 and ARINC 629 with a max bandwidth of 100 Kbps, 1 Mbps and 2 Mbps, respectively. For the new generation A380, none of these ADNs would fulfill the aircraft's demanding requirements to a high available bandwidth, minimum wiring to reduce the weight and low development cost. As a consequence, the Avionics Full Duplex Switched Ethernet (AFDX) was conceived by Airbus and first implemented on the A380.

Meanwhile AFDX is not only used on the A380 but also on the Airbus 400M military transport aircraft and the Boeing 787 Dreamliner, the latter, however, with some minor extensions to the standard. Furthermore, AFDX is foreseen as the ADN backbone in the planned Airbus A350. This shows a broad appliance and acceptance of the AFDX technology leading to reduced cost of AFDX equipment, thus making it even more attractive to deploy this technology.

AFDX Characteristics

The AFDX standard was originally defined by Airbus in the "AFDX Detailed Functional Specification (DFS)" standard. Meanwhile the same standard also exists as an ARINC standard which is called "ARINC 664".

As mentioned earlier, Boeing has based the ADN backbone of their 787 aircraft on the ARINC 664 standard, however with some minor extensions. The Boeing AFDX standard is called "Interoperability Specification for the 787 End System".

AFDX is a serial data transfer method based on conventional Ethernet defined in the standard IEEE802.3. AFDX allows for transfer rates of either 10 or 100 Mbps over either a copper or fiber transmission medium. Since conventional Ethernet is not a deterministic network, AFDX had to be extended to ensure a deterministic behavior and a high reliability in order to comply with the stringent requirements to ADNs.

AFDX ensures a deterministic behavior through traffic control. Traffic control is achieved by guaranteeing the bandwidth of each logical communication channel, called a Virtual Link (VL), thereby limiting the jitter and transmit latency.

To improve reliability, the AFDX standard requires each AFDX channel to be a dual redundant channel, i.e. two channels transmitting the same data stream and at the same time. At any one time AFDX will only forward one data stream to the upper layers, and automatically exclude an erroneous data stream from being forwarded.

With these characteristics AFDX ensures a BER as low as 10^{-12} while providing a bandwidth up to 100 Mbps thereby fulfilling the requirements of new generation aircraft avionics in terms of reliability and available bandwidth.

AFDX Network Architecture
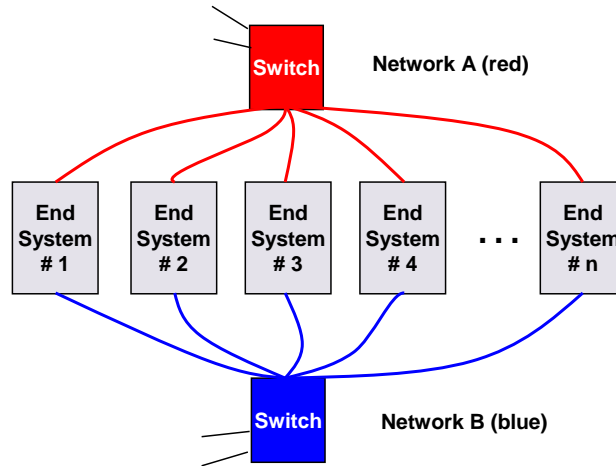
As depicted in Figure 1 each AFDX ES is connected to two independent networks namely network A (red) and network B (blue). The heart of each AFDX network is the switch which establishes physical links between all the ESs connected to the switch. The switch is capable of forwarding data from any connected ES to one or more other ESs connected to the switch. However, which incoming data will in fact be forwarded to which ESs depends on the switch configuration which establishes the logical communication links between ESs.

Based on its configuration, the switch also polices that the bandwidth allocated to each communication link is not exceeded. If the switch detects that the bandwidth of a communication link is exceeded, data is dropped (i.e. discarded and therefore not forwarded) until the bandwidth again is within its specified limits. This ensures that correctly working ESs not violating the switch configuration are guaranteed their allocated bandwidth.

In a properly configured and working AFDX network, a violation of the switch configuration will never occur. However, if an ES for one reason or another starts to malfunction, it could cause the switch to get flooded with data which it will then handle according to its bandwidth limitation policing.

As an example, a malfunction of an ES could cause it to send an erroneous data stream on network A, while an identical but faultless data stream is also sent on network B. Since all ESs eavesdropping on this data receive both the erroneous as well as the valid data stream, one of the two has to be eliminated in any case. Due to the inherent data integrity checking of all eavesdroppers, the erroneous data stream is automatically eliminated and only the valid data stream is passed on to the upper layers. In this example the network A data is eliminated while the network B data is forwarded and further processed.

According to the standard, an AFDX switch must be equipped with at least 20 ports, i.e. it must be able to interconnect at least 20 ESs.

Physical Topology
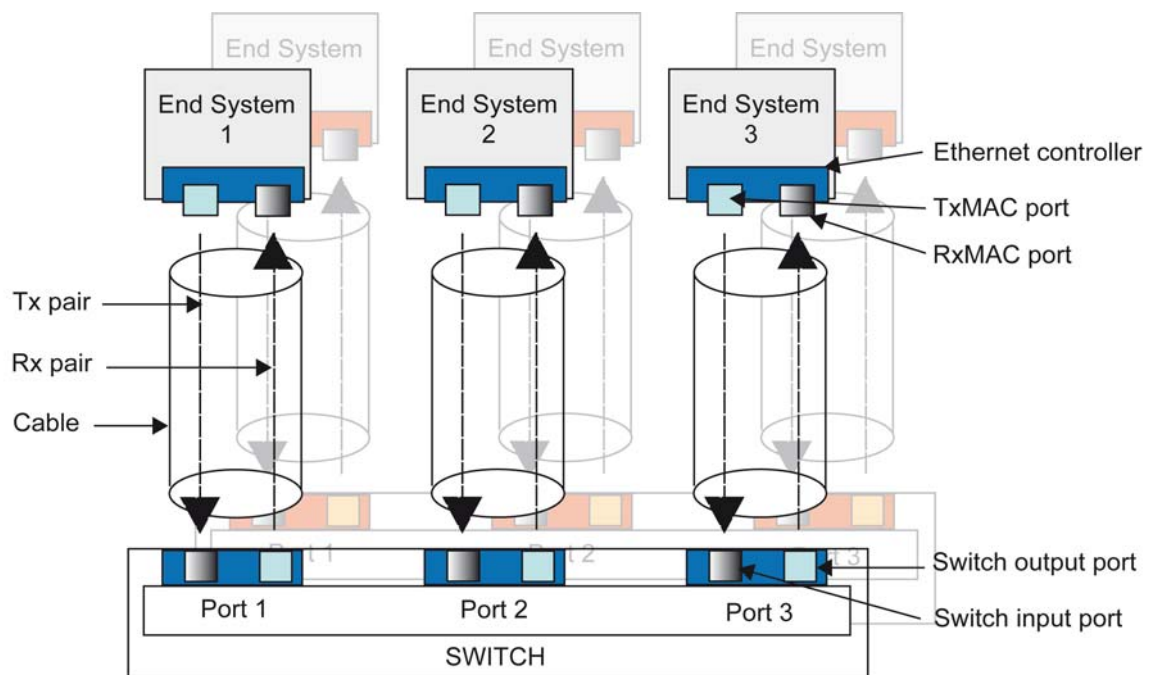
Figure 2: Physical topology of AFDX

As depicted in **Figure 2**, each channel of an ES is connected to a switch port via a cable containing two twisted pair wires interconnecting the input and output ports of the ES and switch.

Since AFDX is based on the Ethernet standard IEEE 802.3, it uses standard Ethernet (MAC) controllers for the communication.

For reasons of simplicity **Figure 2** only depicts one of the two networks, i.e. the interconnections between one of the ES's channels and the switch. The second network has an analogue interconnection; however, it uses the ES's second channel and a second switch.

Logical Topology

Figure 3 shows an example of the logical topology of two ESs communicating with each other. ES 1 transmits the messages 1, 2 and 3 to ES 2 using a common virtual link (VL) which is encoded in the destination MAC address. An AFDX message is uniquely identified by its UDP source and destination port numbers, its IP source and destination addresses as well as its MAC destination address which encodes the VL. In the example, the three messages are uniquely identified as follows:

- Msg 1: UDP Port x Src + IP Src + VL + IP Dst + UDP Port a Dst
- Msg 2: UDP Port y Src + IP Src + VL + IP Dst + UDP Port b Dst
- Msg 1: UDP Port z Src + IP Src + VL + IP Dst + UDP Port c Dst

**Figure 3:
Logical
topology of
AFDX**
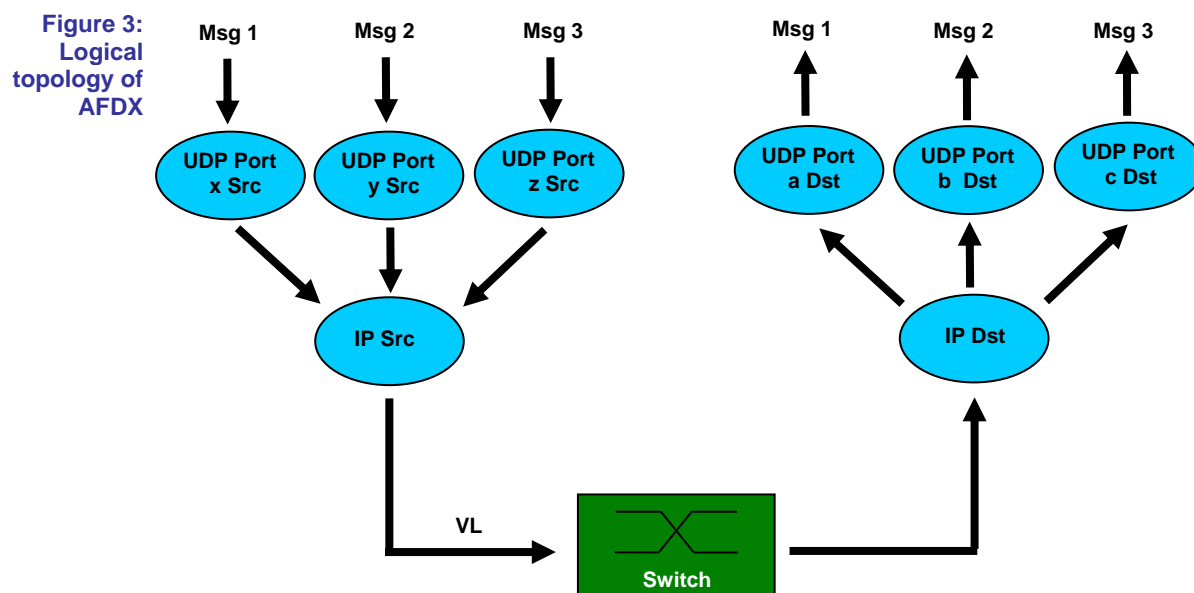
AFDX Communication Concept

The Virtual Link (VL)

Each network of an AFDX ES is connected to the switch via a single cable. This means that the communication between two AFDX ESs takes place over a single physical communication link. However, from a system or application point of view, it is possible to establish many logical communication links, called Virtual Links (VL), which behave like physical links (hence the name Virtual Link) although they all make use of one and the same physical link. Thus, VLs make it possible to establish a sophisticated network communication while ensuring a deterministic behavior through VL bandwidth policing carried out by the switch.

AFDX implements transmit VLs as well as receive VLs. Each transmit VL can only be assigned to one (1) ES, and an ES is only allowed to transmit on assigned VLs. The same receive VLs, however, can be assigned to several ESs meaning that these can be eavesdropping on the same data.

Virtual Link Scheduling

The AFDX traffic shaping is managed by the ES's VL scheduler which multiplexes the frames of all transmit VLs of an AFDX ES onto the physical link. Each VL is like a flow of frames and with multiple VLs there are multiple flows of frames that have to be multiplexed into a single flow of frames. The multiplexing is regulated by the Bandwidth Allocation Gap (BAG) which is unique to each VL.

The BAG is a timeslot confining the VL's bandwidth by defining the minimum gap time between two consecutive frames. The BAG value must be in the range 1 - 128 ms and must be a power of 2.

The jitter is an upper bounded transmit latency appearing as a frame time offset within the BAG. The frame time offset is introduced by the contention of data during VL scheduling, as well as by delays caused by the switch processing.

Figure 4 illustrates the scheduling of three frames with different jitter on the same VL.

Figure 4:
VL scheduling
with jitter
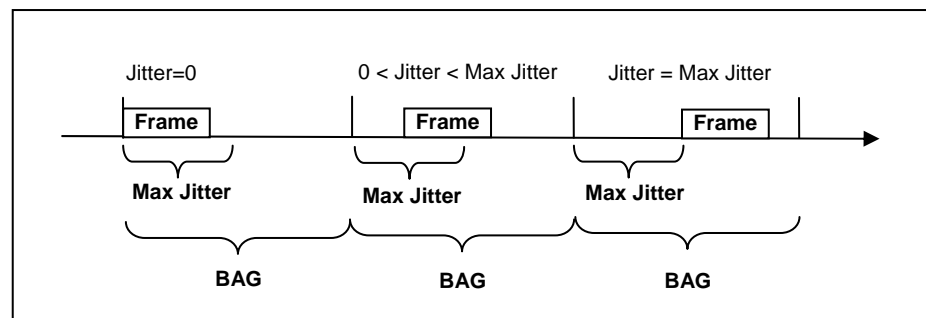
Sub Virtual Links

As the VL is a means for regulating the flow of frames onto the physical link, the Sub-VL is a means for regulating the flow of frames onto the VL. Thereby the Sub-VL makes it possible to optimize the VL's bandwidth utilization. E.g. Sub-VLs can be used to reduce latency of short messages, such as sampling messages, against bulk messages, such as queuing messages.

A VL must be capable of handling at least four Sub-VL FIFO queues. Each Sub-VL queue is read in a round-robin fashion by the VL scheduler. This ensures that the frames in the Sub-VL queues are evenly distributed onto the VL queue and later onto the physical link.

Figure 5: Sub-VL Concept
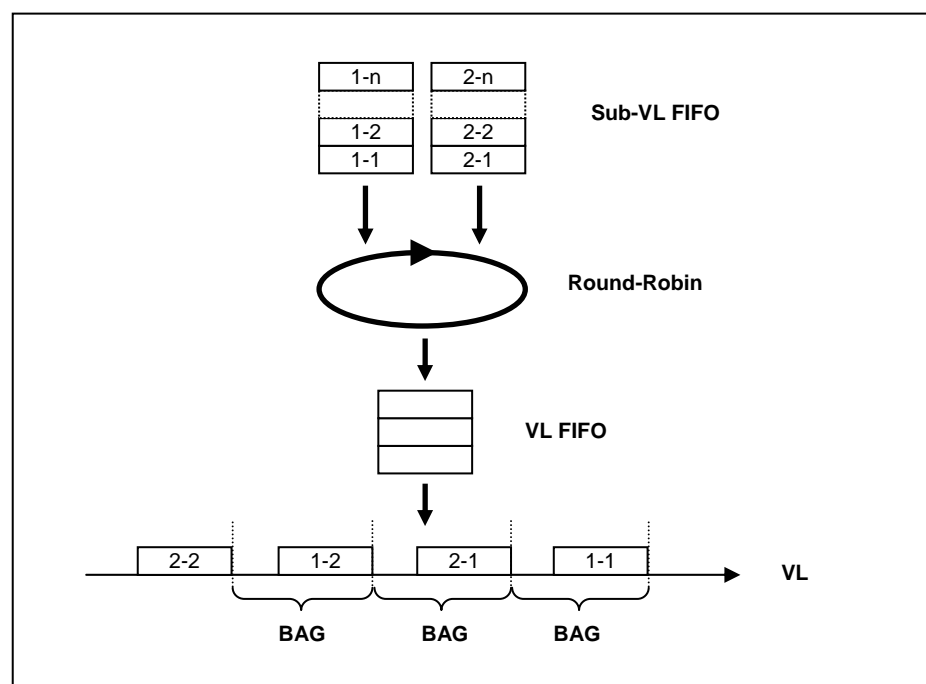
AFDX Port Types

The AFDX ES provides two different port types for data transfers - the communication port and the Service Access Point (SAP). The communication port provides two types of services, namely the queuing and the sampling services which are both based on UDP. The communication ports as well as the SAP ports can be defined as either receive or transmit ports.

AFDX Communication Ports

Receive Sampling Ports

A receive sampling port places received messages in a buffer with storage capacity for a single message. As a consequence all incoming messages on a sampling port overwrite the previous message regardless of whether it was already retrieved by the host application or not.

The host application can retrieve the sampling port's message buffer at any time and will always retrieve the last message received by the sampling port. When no new messages are received, the host application will get the last message received every time it's retrieving the message buffer.

Transmit Sampling Ports

A transmit sampling port continuously transmits the last message written to the port by the host application. When the host application writes a new message to the sampling port, the previous message on the sampling port is overwritten by the new message which from then on is continuously transmitted according to the sampling port's schedule.

Receive Queuing Ports

A receive queuing port appends received messages to the port's message queue. As opposed to the sampling port it guarantees that no messages are overwritten. The host application can retrieve messages from the message buffer at any time, and upon doing so the respective message is removed from the message queue.

The message queue is a FIFO queue, thereby ensuring that the host application will retrieve the messages in the same order they were received by the queuing port.

Transmit Queuing Ports

A transmit queuing port transmits messages on a FIFO based scheme. The messages are transmitted as fast as possible and then removed from the message queue. New transmit messages written by the host application to the queuing port are appended to the message queue. An exhausted message buffer stops further transmission.

Service Access Ports (SAP)

SAPs are used for TFTP transfers and for communication between compliant networks, e.g. other AFDX networks or LANs. When an ES receives a communication request on a SAP, it is passed the IP source address (IP Src) as well as the source UDP port number (UDP Src). When replying to the request, the IP Src is specified as the IP destination address, and the UDP Src is specified as the destination UDP port number.

Frame Fragmentation

AFDX frames transmitted on the wire are confined to be in the range 64 - 1518 bytes. However, it is possible to define AFDX frames that exceed the maximum transmit frame size. Frames conveying sampling port messages are not allowed to exceed the 1518 bytes limit, thus no fragmentation is required. However, frames conveying queuing port messages are allowed to be up to 8KBytes large, thus requiring that these frames (when exceeding the limit) must be divided into fragments transmitted one after the other.

In addition, the AFDX protocol allows limiting the maximum size of a transmitted frame even further by means of the parameter Maximum Transfer Unit (MTU). Each VL has a MTU associated with it, and the frames exceeding the MTU will be fragmented prior to transmission and subsequently reassembled by the receiving ES. Fragmentation and re-assembly is inherent in the AFDX implementation and is thus not managed by the host application.

AFDX Network Protocol

AFDX Protocol Stack

As a protocol derived from Ethernet, the AFDX Media Access Control (MAC) data link layer is almost identical to the Ethernet MAC layer. The only difference is the so-called Sequence Number (SN) which can optionally be inserted as the last byte of the Ethernet payload. The function of the SN is explained in later chapters.

On top of the Ethernet layer, AFDX implements the Internet Protocol (IP) layer which manages frame fragmentation and re-assembly as well as packet forwarding. The latter, however, is not being used in AFDX since the routing is carried out on a VL basis.

The last protocol layer of the AFDX protocol stack is the User Datagram Protocol (UDP) which is connectionless with no transmission error control. UDP was chosen since it is more efficient than the alternative Transmission Control Protocol (TCP). Although TCP is a connection-oriented protocol providing transmission error control, this is not required since the AFDX bandwidth policing and redundancy management ensures a very low frame loss probability. The UDP payload holds the ADFX user data.

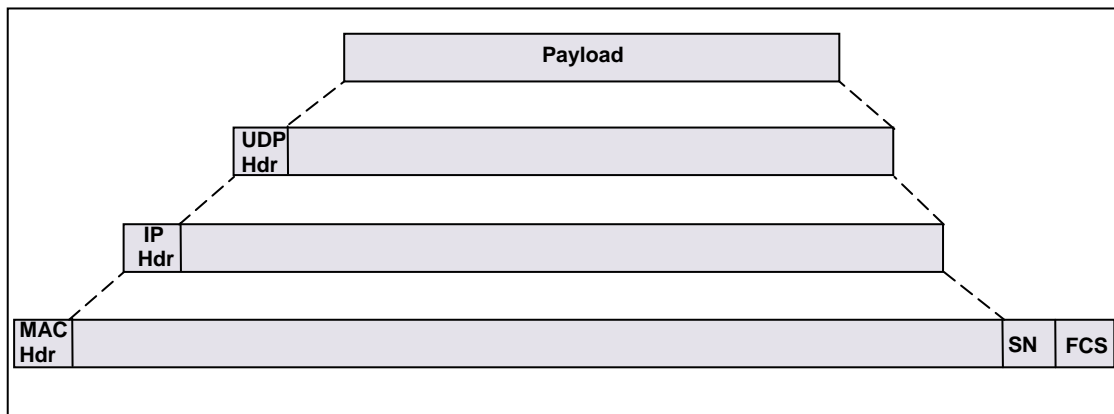Figure 6 depicts the AFDX protocol stack with its different layers.

Figure 6: AFDX protocol stack

AFDX Frame Structure

Ethernet Preamble

To signal the transmission of a new message on the network, the transmitting ES sends out a stream of bytes, called the preamble, prior to transmission of the actual frame. The preamble consists of alternating 0 and 1 bits that give the receiving ESs time for synchronization and otherwise prepare for reception of the actual frame. At the end of the preamble, the transmitting ES sends out the Start Frame Delimiter (SFD) to break this pattern and signal the beginning of the actual frame immediately after the SFD (see Figure 7).

Ethernet Protocol

The first part of the Ethernet frame is the MAC destination address where AFDX encodes the VL identifier in the last two bytes. Following the destination address is the MAC source address, where the ES can encode information such as the network ID, the equipment ID and the Interface ID. After the MAC addresses follows the EtherType field which is used to indicate which protocol type is transported in the Ethernet frame. In AFDX this 2 byte field always has the value 0x0800 meaning Internet Protocol, Version 4 (IPv4).

Ethernet Payload

Following the EtherType field is the Ethernet payload which contains the IP structure, the UDP structure as well as the AFDX payload followed by the Sequence Number (SN). The IP and UDP structures are 20 and 8 bytes long, respectively and the SN is 1 byte long. Since the Ethernet frame is specified to be in the range of 64 to 1518 bytes, the ADFX payload must consequently be in the range 17 to 1471 bytes. This calculation is done by simply subtracting the protocol overhead ($6 + 6 + 2 + 20 + 8 + 1 + 4 = 47$) from the max. and min. frame sizes.

Furthermore, by using padding it's possible to specify the AFDX payload down to 0 bytes.

Ethernet Error Control

The last field of the Ethernet frame is the Frame Check Sequence (FCS) which is 4 bytes long. The transmitting ES uses the Cyclic Redundancy Checksum (CRC) algorithm to calculate a checksum over the entire frame which is then appended as trailing data in the FCS field. The receiving ES uses the same algorithm to calculate the checksum and compare it with the received checksum. If the two checksums are not identical the receiving ES discards the frame.

Ethernet Postamble

Ethernet specifies a minimum idle period between transmissions of frames called the Interframe Gap (IFG), which is not strictly required by AFDX. However, for reasons of compatibility, the IFG also applies to AFDX. The IFG is specified to be 96 bit times, i.e. the time it takes to transmit 96 bits on the network. On a 10 Mbit/s network, the IFG idle time is thus 9.6 us. On a 100 Mbit/s network, the IFG idle time is 960 ns.
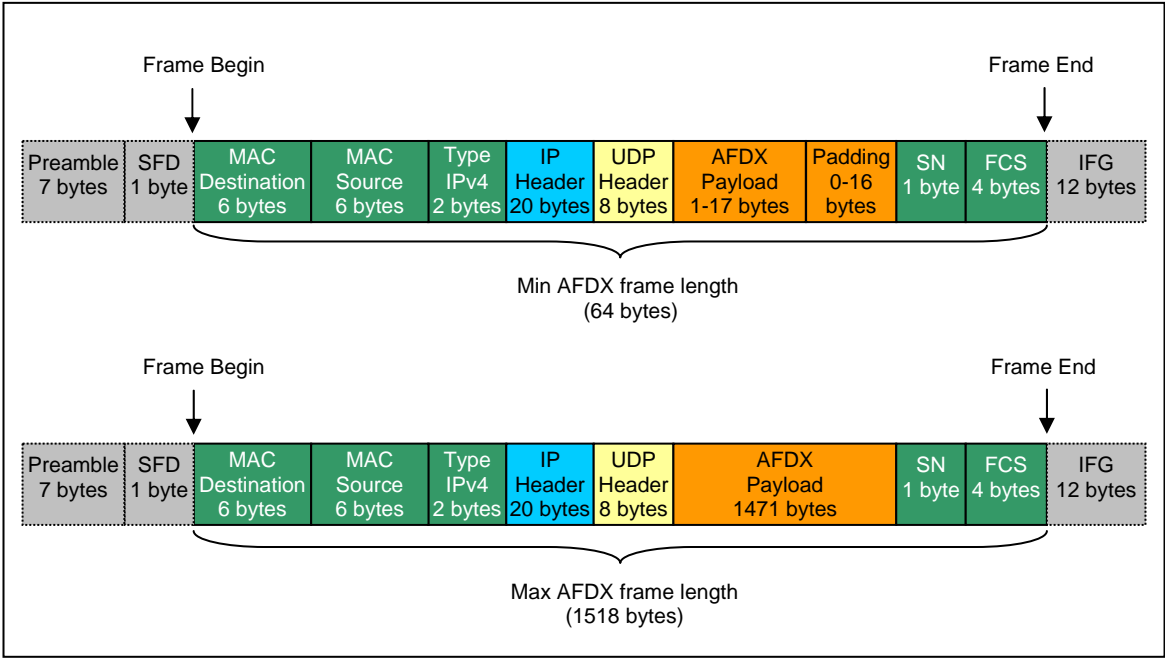
Figure 7: AFDX frame structure

Network Reliability

The inherent AFDX features such as traffic shaping (bandwidth management and bandwidth policing) and limited transmit latency and jitter, make AFDX a deterministic and reliable network. However, in order to improve reliability even further, the AFDX network has a double, i.e. a redundant network transmitting the exact same data. The two AFDX networks are called "network A" (also "red network") and "network B" (also "blue network").

The purpose of the redundant network is to mitigate the consequences of potential network failures caused by e.g. damaged cables and connectors or devices (e.g. switches) generating babbling data.

Since ultimately only one valid data stream is required by the ES application, a handling of the redundant streams is required.

As depicted in Figure 8, the ES implements Integrity Checking (IC) and Redundancy Management (RM) to ensure data integrity and that only one data stream is forwarded to the upper protocol layers and from there to the application. The IC and RM services are provided by the ES without the involvement and knowledge of the application.
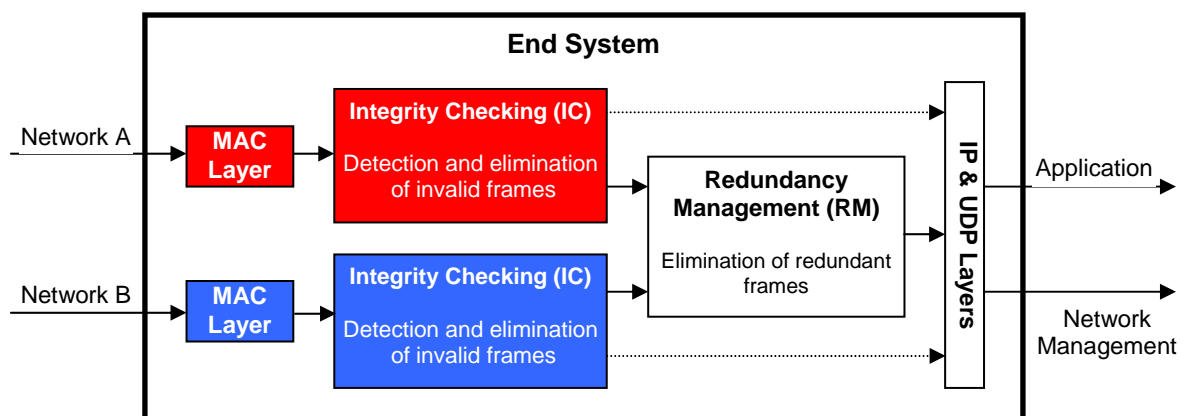
Figure 8: AFDX Integrity Checking and Redundancy Management

Integrity Checking (IC)

The first step for handling the redundant data streams is the IC, which is done separately for each network and on a per VL basis. The IC is always enabled and is done independently of the RM, also if the RM is turned off and both networks are used independently of each other.

The IC is applied on the MAC layer, i.e. on the Ethernet frame which contains a one byte Sequence Number (SN) as the last byte of the payload as illustrated in **Figure 7**.

The SN is the basis for the IC algorithm and is used differently in transmitting and receiving mode.

SN Usage in Transmitting Mode

The SN is a value in the range 0 - 255 and is handled separately for each VL on each of network A and B. Prior to transmission, the SN is incremented by one for each consecutive frame (whether fragmented or not) on the same VL. With SN = 255 in the last transmitted frame, the SN is wrapped around to 1 in the following frame. Upon a reset or start-up of the transmitting ES, the SN is set to 0 in the first transmitted frame.

SN Usage in Receiving Mode

In receiving mode, the IC uses the SN to determine if frames have been lost or whether a babbling switch is causing the same frame (with the same SN) to be transmitted over and over again.

The IC algorithm accepts all frames that comply with one of the following criteria:

- SN = 0 (The transmitting ES is started or reset)
- SN = Previous SN + 1
- SN = Previous SN + 2

All frames not complying with these criteria are discarded.

Redundancy Management (RM)

The purpose of the Redundancy Management (RM) is to evaluate the two frame sequences delivered by the IC (see Figure 7), discard possible duplicate frames, and forward only one copy of each frame to the upper protocol layers.

The RM makes use of the configurable SkewMax parameter which is given in ms and must be specified for each receive-VL defined in the ES. SkewMax defines the maximum allowed time between the reception of two redundant frames (i.e. with the same SN), one received on network A and the other on network B.

If SkewMax is not exceeded, the RM applies a "first-valid-wins" policy on the two frames, i.e. the first received frame is forwarded whereas the later received frame is discarded. However, if SkewMax is exceeded, the RM considers the two frames to be different from each other and hence forwards both.

In the case where the RM is disabled, both frame sequences are forwarded directly from the IC to the upper layers.

The AFDX Switch

Figure 9: AFDX Switch Architecture
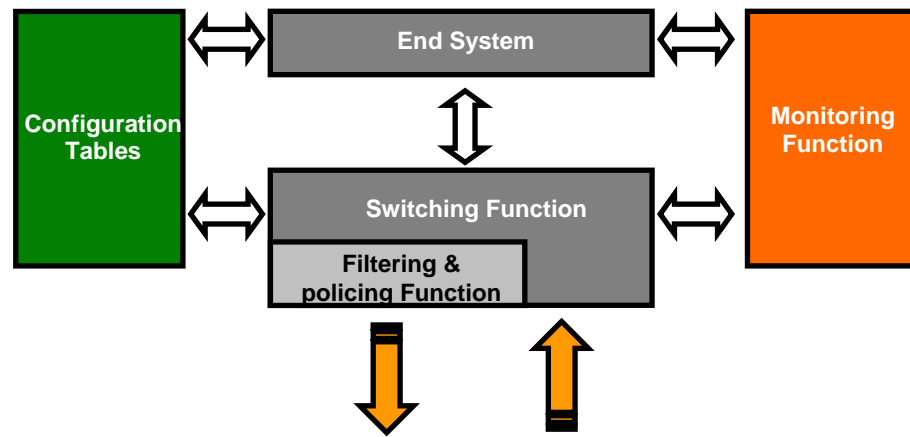
The purpose of the AFDX switch is to physically interconnect the ESs and police that the communication takes place according to the network configuration.

As depicted in Figure 9, the switch consists of various components each performing a certain task of the switch.

The main component of the switch is the switching function which implements a filtering and policing function to ensure that only valid incoming frames are forwarded to the right physical ports.

The setup of the switching function is done using configuration data held in static configuration tables.

The purpose of the monitoring function is to monitor and log all switch operations and events such as frame arrivals and invalid frames. The Monitoring function communicates with the network management function for operational and health related information.

The purpose of the switch ES is to provide a means for functions that are external to the network to communicate with the switch. For example dataloading and network management functions communicate with the switch via the embedded ES.

Frame Filtering

The switch filtering functions examines all incoming frames and ensures that only valid frames are forwarded to selected destinations. The filtering function verifies the following:

- Ethernet destination address validity, i.e. the validity of the VL identifier including the constant part

- The VL is received on an allowed destination port according to the configuration table

- FCS Validity

- Ethernet frame size alignment, i.e. the frame size is a 1-byte integral

- Ethernet frame size range, i.e. the frame size must be in the range 64 -1518 bytes

- Adherence to the MTU on a per VL basis

Incoming frames not complying with these conditions are automatically dropped.

Traffic Policing

Traffic policing can be implemented using two different algorithms of which one uses byte-based policing and the other frame-based policing. The byte based traffic policing filters out the VL in terms of bandwidth usage expressed in bits per second. The frame-based traffic policing filters out the VL in terms of bandwidth usage expressed in frames per second. The switching function may implement one or both of the two algorithms.

The implemented algorithm(s) operates on the basis of the VL identifier contained in the MAC destination address. The VL defines a traffic flow and is characterized by certain properties such as BAG, jitter and group of recipients. The properties of each VL are contained in the configuration tables.

Traffic policing ensures containment of faults caused by ESs. Incoming frames that do not comply with the configuration of the traffic flow (VL) to which they belong are automatically discarded by the policing function.

AFDX Network Configuration

Transmit Scheduling

For periodically transmitted messages on a VL it is essential to know whether the given transmit intervals are possible to implement with the given VL. The time constraints for a periodically transmitted message are defined by the properties of the message itself and of the VL to which the message is assigned.

If the message is not fragmented, the complete message is transmitted in just one time slot, defined by the BAG. However, if the message is fragmented, i.e. divided into several pieces, each fragment will get transmitted in its own time slot.

Whether a message will get fragmented depends on its payload (PL) and on the Maximum Transfer Unit (MTU) of the associated VL. In particular, the number of message fragments is calculated as PL_{msg}/MTU_{VL} rounded up to the next integer.

Formula 1 shows the inequation which all messages associated with the same VL must obey. The left hand side of the inequation is a sum of the frequencies of all message fragments transmitted on the VL. The right hand side is the maximum frame frequency feasible with the given BAG.

If the inequation is disobeyed, the network cannot cope with the transmit schedule, meaning that some messages will never be transmitted or cannot always be transmitted as per their transmit interval. Aperiodically transmitted messages (queuing port messages) are affected first, since they are lower prioritized as periodically transmitted messages (sampling port messages).

$$\sum_{msg=1}^{n} \frac{\left\lceil PL_{msg}/MTU_{VL} \right\rceil}{MTC_{msg}} \leq \frac{1}{BAG_{VL}}$$

Formula 1: Bandwidth constraints of messages on the same VL

Transmit Scheduling Example

Formula 2 shows the calculations for an example of two messages, msg_1 and msg_2 being transmitted on the same VL. Since the payload of msg_1 exceeds the MTU, the message is divided into two fragments. The payload of $msg_2,$ however, fits the MTU so it will not get fragmented. When inserting into Formula 1, it can be deduced that the inequation is disobeyed, hence the transmit schedule is not feasible.

To get a working transmit schedule, its necessary to change one or more of the four properties used in the inequation. Often it's easier and better to change the message properties than the VL properties, since changing the latter affect all messages on that particular VL. For example, the transmit schedule would work if MTC_{msg2} is changed to 10 ms. Alternatively, MTC_{msg1} and MTC_{msg2} can be changed to 8 ms and 4 ms, respectively, which would also ensure that the formula is adhered to. In both cases the

left side of the inequation will exactly equal the right side, meaning that the VL bandwidth is fully exploited.

$$BAG_{VL} = 2ms, \quad MTU_{VL} = 100bytes$$
$$PL_{msg1} = 200bytes, \quad MTC_{msg1} = 5ms$$
$$PL_{msg2} = 100bytes, \quad MTC_{msg2} = 3ms$$

$$\frac{\lceil PL_{msg1}/MTU_{VL} \rceil}{MTC_{msg1}} + \frac{\lceil PL_{msg2}/MTU_{VL} \rceil}{MTC_{msg2}} \leq \frac{1}{BAG_{VL}}$$

$$\Downarrow$$

$$\frac{\lceil 200/100 \rceil}{5} + \frac{\lceil 100/100 \rceil}{3} \leq \frac{1}{2}$$

$$\Downarrow$$

$$\frac{11}{15} \leq \frac{1}{2}$$

$$\Downarrow$$

Inequation disobeyed!!

Formula 2: Example using Formula 1 for two messages on the same VL

Bandwidth Allocation

As previously mentioned, the purpose of the switch is to police that the network's maximum bandwidth is not exceeded. The network load can be calculated by the sum of the load of each message on the network. Since a message consists of a protocol part as well as a payload part, it is necessary to take both parts into consideration when performing the calculation. The AFDX protocol overhead is shown in **Table 1**.

Protocol Type	Protocol Element	Size (bytes)
IEEE802.3	Preamble	7
IEEE802.3	SFD (Start Frame Delimiter)	1
IEEE802.3	MAC (Media Access Controller) Destination	6
IEEE802.3	MAC Source	6
IEEE802.3	Ethernet Type	2
RFC	IP (Internet Protocol) Header	20
RFC	UDP (User Datagram Protocol) Header	8
AFDX	RSN (Redundancy Sequence Number)	1
IEEE802.3	FCS (Frame Check Sequence)	4
IEEE802.3	IFG (Inter Frame Gap)	12
	Total:	67

Table 1: AFDX protocol overhead including leading and trailing bytes

In other words, every time an AFDX message is transmitted, 67 bytes alone is protocol information which is a significant amount considering that the smallest allowable AFDX payload size is merely 17 bytes.

With this information it is now possible to establish a formula for the network load of an AFDX message as shown in Formula 3.

$$Message\ Network\ Load\ =\ \frac{\left(67 + PL_{msg}\right)}{MTC_{msg}}\ bytes/ms$$

$$\Updownarrow$$

$$Message\ Network\ Load\ =\ 8\frac{\left(67 + PL_{msg}\right)}{MTC_{msg} * 1000}\ Mbps$$

Formula 3: AFDX message network load

To establish a formula for calculating the maximum allowable total network load, it is necessary to take into consideration that messages can be fragmented as was also the case in Formula 1. Formula 4 shows the inequation which the network configuration must adhere to in order not to exceed the maximum network bandwidth. The left hand side of the inequation is the sum of the network loads caused by the transmitted message fragments each having a payload of MTU bytes. The right hand side is the max allowed bandwidth on the AFDX network which is either 10 Mbps or 100 Mbps.

$$8 \sum_{msg} \frac{\lceil PL_{msg} / MTU_{VL} \rceil (67 + MTU_{VL})}{MTC_{msg} * 1000} \ \leq \ \ Max \ Bandwidth \ (Mbps)$$

Formula 4: Bandwidth constraints from a message point of view

The inequation in Formula 4 is actually not quite exact and is merely an estimation of what the network configuration must adhere to. To be absolutely exact, the inequation would also have to take the following into consideration:

1. The UDP header (see Table 1) appears only in the first fragment of the message's fragment sequence.

2. Depending on the message size, the last fragment in the sequence may be considerably smaller than the MTU.

3. Fragments always carry a multiple of 8 bytes.

However, to also consider these aspects of fragmentation in Formula 4 would make it impractical to use. Nevertheless, although merely an estimation, the formula is accurate enough to ensure that a network configuration adhering to it is always a valid configuration.

Finally, it is also possible to establish a formula for the network configuration based on the bandwidth allocation of each VL as shown in Formula 5.

$$8 \sum_{VL} \frac{MTU_{VL} + 67}{BAG_{VL} * 1000} \ \leq \ \ Max \ Bandwidth \ (Mbps)$$

Formula 5: Bandwidth constraints from a VL point of view

The inequation in Formula 5 can be derived from Formula 4 using the information of Formula 1.