A comparison of Distributed Data Communications using Ethernet in Aircraft

This paper was downloaded from TechRxiv (<https://www.techrxiv.org>).

LICENSE

CC BY 4.0

SUBMISSION DATE / POSTED DATE

22-11-2020 / 31-12-2020

CITATION

Hamid, Owais; Vaqar, Sayyid Anas (2020): A comparison of Distributed Data Communications using Ethernet in Aircraft. TechRxiv. Preprint. <https://doi.org/10.36227/techrxiv.13270742.v1>

DOI

[10.36227/techrxiv.13270742.v1](https://doi.org/10.36227/techrxiv.13270742.v1)

A comparison of Distributed Data Communications using Ethernet in Aircraft

Owais Hamid
Systems Design Engineering
University of Waterloo
omabdulh@uwaterloo.ca

Anas Vaqar
Training and Research
HTN Networks Inc
anas.vaqar@gmail.com

Abstract—Distributed control in the aviation industry provides the backbone for reliable information communication. The precise nature and the complexities dealt by a control system in the air require reliable communication of data in a timely manner. This has led to communication protocols defined specifically for the aviation industry. This paper endeavors to provide an overview of specific networks and data communication standards in the aviation industry, the focus being on aviation standards that require working under rigorous time critical environments. Specifically it focuses on the more modern techniques used for data bus communications in aerospace, with a comparative study between the *Time Triggered Ethernet (TTE)* and *Avionics Full Duplex Switched Ethernet (AFDX)*. While several studies have explained the evolution of Aircraft Data Networks (ADNs), this study compares two Ethernet based data communication protocols for reasons pertaining to the trend of usage of *Commercial Off the Shelf (COTS)* Components. A detailed and comprehensive understanding of all components of the two types of networks is provided here, in addition to clear comparisons using a table.

I. INTRODUCTION

Message communication can largely be subdivided into two categories based on the time they are sent. **Event triggered communications** are systems that only generate communication messages based on a specific event when a new information request is sent out or some request needs to be completed by the user for the users' own request. A good example of such a system is what is used between computer nodes in a network, specifically an Ethernet. The requests sent out by users from a browser, for example, are based upon the users' own needs and do not reflect a time based communication system.

On the other hand, if it is known when the specific times are required to transmit messages in a process that is based on well known sequential dynamics, **time triggered communications** can be used. In such communications, every node is given a specific agreed upon time when it will transmit whatever it needs to communicate. The definition of the agreed upon time depends on the architecture. Some of them derive their timings from a master clock. Others can combine clock messages from several nodes to create a master-less approach for generating a time base. The advantage of the latter is that a fault tolerant clock algorithm based on various nodes keeps correcting errors that might creep in from specific nodes. Besides, in case of a failure, the latter approach does not crash. It merely continues to rely

on others nodes for fault correction [6].

Added to all this is the management of processing power. Initially, as Aircraft sensors started incorporating sensors that required higher processing capability, dedicated processors were used for a task, example: navigation. These are called *Federated Architectures*. Federated architectures require a lot more space, weight and the number of components used are much higher. Moreover, redundancy requires even more components. Cabling lengths, weights and costs increase dramatically, and [10] attributes cabling, processors and spare costs to as much as 50%.

For the various tasks that require computing, this architecture has proved to be unsatisfactory, and therefore the push has been towards *Integrated Modular Avionics (IMA)* [11]. The IMA Architecture calls for sharing of hardware and computing power. Modern computers can easily deal with concurrent tasks in a parallel fashion, as shall be seen in Section IV-D, whereas faults can be contained using partitioning spaces in the computer.

As a push towards better maintenance and utilization of COTS components is also a consideration, the usage of Ethernet and devices used for it contribute to lower maintenance costs. A part of the problem with IMA is fault tolerance. While federated architectures prove costlier, bulkier and unwieldy, they can easily isolate faults due to independent processing of different components. IMA architectures similarly need to have robust fault tolerance for functions of varying criticality. This means that errors from low priority communications must not propagate to highly critical functions and their communications [9].

Therefore, the IMA architecture has the objective of sharing hardware and computing resources with the same reliability that Federated architectures provide in terms of fault isolation. Admittedly, this requires separating the levels of criticality all on one computing platform, which makes it a challenge in itself.

II. WHY UTILIZE ETHERNET?

When it comes to Ethernet as a medium for a good Quality of reliable transmission for real time purposes, Ethernet by itself cannot provide a rigorous Quality of Service since it is entirely event based, and the latency, the time taken for a message to reach across the network, is not deterministic. Rather, it is best effort [9]. To remedy this, several solutions

have been proposed. These include Industrial Ethernet such as Profibus, Foundation Fieldbus and EthernetIP. Variations of the Ethernet such as the Advanced Full Duplex Switched Ethernet (AFDX) modifies Ethernet for quick real time situations. However, classical Ethernet has had limited adaptation to these Data network systems. Adapted Ethernet solutions cannot be utilized for the already present infrastructure since they are specifically tailored.

This study is motivated towards understanding how commonly available technology can be leveraged to provide rigorous Quality of Service as required by the aerospace industry. This case study utilizes the adaptation of Ethernet for a specialized purpose. It is hoped that by understanding this strategy of using the concept of Commercial off the Shelf Components (COTS), and similar strategies for cost reduction, an example can be had for utilizing other commonly used technology for specialized purposes.

There are several examples where the COTS philosophy has been implemented to reduce costs. Notably, Airbus' aircrafts, especially the A series including the Airbus A380 utilizes COTS components that are mass produced for many other products and are available [3]. This shows how important cost effective COTS components are, and why it is important to build step by step on commercially available technology. The study endeavors to understand the differences that are created by the philosophy embodied in the event trigger driven Avionics Full Duplex Switched Ethernet (ARINC 664 P7) and the time trigger driven TTEthernet.

As a result, the factors under consideration are strongly influenced by how easily a network can be maintained and how well it can be adapted to architecture that can be found, i.e. legacy systems.

Since the two data networks and their operations need to be described in order to grasp the differences between them, the following comparison categories are utilized in this work.

- Network Topology
- Physical layer characteristics
- Redundancy
- Frame Structure/Word format
- Fault Tolerance

III. TIME TRIGGERED ETHERNET (TTE/SAE AS6802)

TTEthernet was an effort made to bridge the very widely available Ethernet without a definitive Quality of Service and the adapted solutions to Ethernet such as those mentioned above. The project and experiences come out of the applications of the TTP/C in aerospace, automotive and railway domains [8].

A. Network Philosophy

Networking trends usually incorporate two philosophies. **Statically configured** networks (Eg. Fieldbus) are definitely reliable in communication and offer latencies suitable for various real time purposes. However they are specifically fashioned for that task. **Dynamically configured** networks

Fig. 1. TTE combines Open and closed world communications, from [1]

are open and provide free-form communication, the drawback is message sending may sometimes fail and Quality of Service (QoS) is required to ensure such free form communication is delivered on time. Determinism is then sometimes lost.

Time Triggered Ethernet combines the flexibility of free form communication and the determinism of statically configured communication [8].

Central to the TTE is the switch that allows communication between various parts of the Network. The switch is called Ste06. Thus, the TTE consists of end systems that need to communicate and receive, and switches that connect them both ways to the controller.

The TTE is designed to be compatible with the IEEE 802.3 standards. It is used as a backbone network for applications in auto-industry and aerospace. It can take care of Navigation and guidance system, entertainment and flight control. To do this, the network is partitioned so that critical applications receive higher priority and different applications do not interfere with each other.

TTE can use deterministic communication to achieve fixed latencies to realize real time control. It can then use other dynamic configurations for less critical applications.

In this way, TTE combines the best of two worlds, Time Triggered (TT) and Event Triggered (ET) communication [7].

1) *What makes a Time Triggered Ethernet:* According to SAE International, Time Triggered Ethernet is an addition on top of Ethernet that implements at least the following [12]:

- The AS 6802 synchronization architecture for fail operational and safety critical systems. The AS 6802 defines fault tolerant synchronization for distributed systems including switches and end components as defined in IEEE 802.3
- Time triggered data flow with message scheduling
- Timing strategies for datagram flows
- Traffic partitioning for fault detection

2) *TTE Characteristics:* Among other things, TTE can provide [7]:

- Fault Tolerant Clock Synchronization: TTE uses something similar to the IEEE 1588 Precision time protocol, but with multiple active grandmaster clocks. It has a global time that synchronizes between various components.
- Deterministic message transport

Fig. 2. TTE in the OSI architecture

- Startup service
- Fault detection and diagnosis: **Temporal Partitioning** is a concept for fault isolation where a temporal firewall prevents the faulty component from affecting the determinism of critical traffic flows.
- Precise Diagnostics: Global time stamps simplify reconstruction of events. Synchronous capturing of events can help in understanding the state of a system that is distributed over several kilometers.

TTE operates on the Data Link layer which is OSI layer 2. The physical layer does not matter, however wireless communication is not recommended as it depends on signal strength [8].

B. Network Topology of the TTEthernet

It is important to understand the key parts of a TTEthernet. The end systems are made up of network adapters loaded on to standard CompactPCI/PCIe devices or FPGAs. Message transmission is conducted on a schedule where every end system knows when it has to transmit. This schedule is important and is loaded on to Switches and End systems so that it is known before hand when the messages will arrive. This plays a key role in determinism of the time required for a message to propagate (also called the latency) [9]. The schedule has to follow an established global clock that shall be described later.

TTE has three different priority classes [8]:

- **Time triggered (TT) traffic:** Every node is assigned a transmit schedule and every TTE switch also has a receive and forward schedule.
- **Rate Constrained (RC) traffic:** These messages are sent on a reserved bandwidth. No clock synchronization is needed.
- **Best effort (BE) traffic** is compatible with IEEE 802.3. There are no timing guarantees with Best effort messages.

TTE operates on layer 2 but can also allow the usage of layer 3, and upper layers built on top of it. Messages from other protocols such as IP v4/v6 and TCP/UDP can be sent without any modifications.

TT messages are deterministic in nature. They are sent at determined times and take the highest precedence. This keeps the *Jitter*, the delay variation of a frame, to less than $1\mu s$ [15].

These TT messages are suited for fly by wire systems in aviation where control loops require tight control by the

Fig. 3. An example of data flow within the TTE architecture. Taken from [8]

flight controllers.

Switches play a central role in communication. TT messages are handled by a switch with the help of a defined schedule. RC messages have less strict requirements regarding both time and jitter (message quality). They can be used for message descriptions that require reliable communication but are not urgent to the functioning of the plant. However, when it comes to switches, TT messages can delay RC messages, and these may get queued.

BE methods follow regular Ethernet based practices. They use the remaining bandwidth and the switch only transmits them when no RC or TT messages are using up the bandwidth.

Even if there are physical links that connect all end systems to their switches, and all switches to each other, message routing between two end points is conducted through a concept called *Virtual Links (VLs)*. Each Time Triggered frame consists of a Critical Traffic Identifier (CTID), that occupies two bytes within the Ethernet header as shown in Figure 5.

C. Startup and Synchronization

It is also important to understand synchronization in TTE. TTE is built to tolerate failures with end systems and switches. If a system has to work in real time, it needs to be assured that only one synchronization framework exists. With this in mind, three types of synchronizations exist.

Data Source Synchronization assumes all TT data are externally synchronized. The assumption is that TT messages in conflict with each other will not be sent and are detected externally. In this case, an ET message in conflict with TT will be delayed until the TT message has been cleared.

Time Synchronization uses an externally synchronized timing clock. IEEE 1588 based timing sources can be utilized in this context.

After startup, a *configuration state* needs to be determined. This state informs the switch on how to identify a TT message. The configuration message can be sent as a regular ET message. Fault diagnosis can also happen here. The end component can send the current configuration state if the switch detects a fault, and the system can recover quickly from that fault.

A TT message will be sent at the beginning of every cycle, and any message that arrives in the interim is stored until cycle start. Any failures in this particular domain are

Fig. 4. An example of communications in the TTE architecture

contained by the switch via a firewall, and do not propagate beyond this part.

For clock synchronization and after, three roles are assigned to the various components. The switches are called Compression Masters (CMs), while the end systems are called Synchronization Masters (SMs). Synchronization clients (SCs) can be assigned to both end systems or Switches.

After the system is powered on, two protocols are activated. The coldstart protocol establishes real time upon powering up the system.

TTE components transmit synchronization messages to keep themselves synchronized. Components are termed as either masters or slaves, in which slaves ask for time sync, whereas all masters keep simultaneous track of time to prevent fault. There are two steps to the synchronization. First, SMs send Protocol Control Frames (PCFs) to CMs at the same local time. The PCFs are the primary entities responsible for communication of time details. Since there has been a drift in the absolute time, the CMs calculate the difference in time that these have been received by them. This variation is used to calculate a new global synchronized time. This time is relayed by another PCF to the SMs and SCs which then get synchronized accordingly [9].

D. Protocol

As mentioned before, two types of messages can be handled, TT-messages (Time triggered) and ET messages (Event Triggered). TTE deals with both TT and ET messages in similar formats. The only difference is in how they are dealt with at the switch.

Every sender has a pre-defined time sequence where a frame starts as shown in Figure 3. TT messages may be sent at any point in time. However, at the time a new cycle starts, TT messages are prioritized and ET messages are either delayed by the switch, or are pre-empted. At the start of this cycle, TT messages are forwarded by the switch to its recipient. Then the band is open to best effort messages. As stated in [7], in a 100 MB implementation, the time taken using this strategy for a TT message to be delivered is at most $5 \mu s$.

Therefore the delay for a TT message is known and calculated a-priori. Once the TT message has been transmitted at the start of its cycle, the ET message which had been stored in the switch buffer and was waiting, is then transmitted.

Fig. 5. An Ethernet frame as compared to a TTEthernet frame, from [9]

Since messages are transmitted on the basis of priorities, TT frames are scheduled to go at fixed intervals or cycles apart from each other as can be seen in Figure 4. If a low priority frame occurs (under an Event trigger) just before the TT based high priority communication should occur, there are a few ways to deal with it.

The *Pre-emption* method cancels the ongoing ET message if it is time for the TT to transmit.

Timely Block mode means that since it is known when a TT message will be initiated, the ET message is not transmitted at all, and is held back until the TT message is through.

Lastly, *Shuffling* can be induced, which means the high priority frame will be queued and will only be transmitted after the low priority message is through [8].

Finally, the usage of Virtual Links (VLs) ensures that a single message from an end system can easily propagate to multiple end systems located elsewhere providing state information to multiple components that might need it [9].

E. Frame Structure

Since TTE is fully compatible with Ethernet, or the IEEE 802.3 protocol, the switch dealing with messages needs to identify which messages are TT, RC or BE categories. As can be seen in Figure 5, there is a 2 byte representation for Ethernet type frame. One strategy that can be configured for use is that this Ethernet type can be utilized to indicate the type of message, whether TT, RC or BE. The Ethernet type field is two bytes long. According to Ethernet standards, if the value here is between hex 0000 and hex 05DC, the type field denotes length of message. If it is larger than hex 0200, a special message is indicated. IEEE has assigned 88d7 for a TT message [8].

Another strategy is to examine the contents of address fields in the message.

Lastly, the time at which the message arrives can be used to determine the type. If a message arrives at the predefined time it was supposed to, that message would then be TT.

In the frame structure shown in Figure 5, the frame starts with a preamble of 7 bytes that allows the receiver to synchronize with the global clock. The Start Frame Delimiter (SFD) allows the identification of the start of a new frame. The CTID identifies the VL along which the frame should get routed, and the 2 byte length identifies what sort of message (TT, RC or BE) the frame is. The Frame Check Sequence (FCS) allows for error detection, and the Inter Packet Gap (IPG) is the minimum gap needed between two frames. The data payload can be between 46 and 1500 bytes.

Fig. 6. Redundancy implementation in TTE, taken from [1]

F. Safety and Redundancy

Two different priorities guide failure resumption operations under TTE. In the **Fail safe** operations, the system must reach a safe state in case of failure. In **Fail operational** applications, the system must remain fully functional despite failure [9].

TTE is usually setup with redundant switches, end components and links. Arbitrary and inconsistent omissions along with end system and switch failures should not degrade the critical services that run TTE such as clock synchronization [9]. TTE also allows for the 'guardian' system where components are checked for the values they output. The output values need to be within certain parameters, and a fault can be detected if it isn't within the specified parameters.

Just as was mentioned in Section III-A.1, there are certain categories that need to be fulfilled for a Time triggered Ethernet to be time triggered, while working on top of an Ethernet system. These include time triggered traffic flow control which form an essential part of TTE. However, when no such capability is configured, rugged TTE devices perform as full duplex switched Ethernet with IEEE 802.3 and 802.1 standards.

A NASA study [9] conducted rigorous fault analysis on two different missions that were conducted: the Asteroid Redirect Mission (ARM) and the Orion Ascent Abort 2 (AA-2).

For the ARM (which is no longer active), the computers should communicate with two external devices outside the control loop, a crew display and an Environment Control Life Support System. For the second, Orion AA-2 is a sequence that aborts the launch of the Orion space vehicle due to any major failure using the Launch Abort System (LAS) including the point where the vehicle undergoes the highest aerodynamic stress or Max Q. TTE was used in both cases, the AA-2 requiring more rigorous testing where TTE should ensure the communication of messages between various modules using redundancy.

In essence, a failover mechanism is instituted where should

the main (primary agent) communication fail, a secondary (backup) procedure ensures the delivery of messages. Critically, the performance should remain the same as in regular conditions. Previously, failover for NASA's spacecrafts was implemented using the existing IP-based Ethernet backbone. For the Orion based AA-2 project, TTEthernet was selected because, other than the benefits described previously on determinism, TTE's VL based message forwarding, which includes forwarding to multiple addresses is a facility not provided by simple Ethernet. TTE channels allow failovers to occur more seamlessly, the backup and main components receiving the same data at the same time. The switches can be hotswapped without any negative impact to the overall system.

TTE drivers and ethernet adapters for TTE were integrated into the cFS (core Flight System), and the ARINC 664/P7 was also used for the Rate Constrained (RC) traffic (which will be introduced in the next section). The testing was done to determine what the maximum effective throughput would be, i.e. the highest possible rate of transmission without loss of any information. TTE was tested for extremely tight timing requirements with a need for high throughput given the safety procedures' need to be enacted instantaneously. The requirements called for 200 cycles per second (200 Hz) with each cycle running for 5ms. Each time triggered message was transmitted for $250\mu s$ allowing for a total of $5/0.25 = 20$ interrupts, i.e. 1 TT message and the rest for other RC and BE messages.

G. Implementation and hardware design

Ethernet switches play an integral role in TTEthernet design. They have the capability to incorporate copper wires for electrical signals or fiber wires for optical communication. The hardware is based on Intel Altera Commercial off the shelf (COTS) mother boards and PHY daughter boards.

IV. ADVANCED FULL DUPLEXED SWITCHED ETHERNET (AFDX/ARINC 664, PART 7)

A. History of Aircraft Data Networks

The Aeronautical Radio Incorporated (ARINC) produces standards and Systems Engineering solutions. As an early entrepreneur for Aircraft Data Networks, the ARINC 429 was released and is the most widely used data link system in aircrafts presently. However, since it operates on a point to point basis, it is heavy and costs much more due to added wiring. It operated at a speed of 100kbits/s at the highest and could communicate with 20 receivers.

In response to this, the ARINC 629 was released and implemented on the Boeing 777. However, it was adopted by very few companies because of the proprietary nature of the hardware. Furthermore, despite reduced cabling, there was an addition in complexity, and hence the cost. Commercial off the shelf components were unavailable making the costs higher. ARINC 629 operated at 2 Mbits/s and could communicate with 120 terminals [13].

B. The need for AFDX

Due to these problems, a need was identified for a Data Network that would have the following properties.

- Commercial off the Shelf (COTS) hardware that was easily available and replaceable
- Standardized communication protocol
- Quality of Service that ensured timely delivery of messages
- Utility of higher bandwidth available due to advancing technology to address better functionality and higher safety.
- Utilization of the Integrated Modular avionics design

The IEEE 802.3 (Ethernet) standard was chosen due to the wide implementation and experience with the usage of this particular Network. Since the AFDX is implemented as a modification of the Ethernet, COTS parts are readily available.

There are several problems with the Ethernet that need to be addressed to provide a high Quality of Service and impart determinism to the messages being sent. Lack of determinism is the primary concern. Packets are usually lost and Ethernet is configured to handle these problems by asking for the lost packets again. Collision is another major issue that adds non-determinism due to the tactic of random delays while the Carrier is sensed to be busy. In the worst case scenario, multiple such collisions can lead to a jammed bus [4]. The Ethernet architecture also consists of switches that can re-order the sequence of packets from source to destination. While this flexibility is well suited for non time critical tasks, packet re-ordering may lead to delays due to switch hopping further adding to non-determinism. No hard quality of service is provided by the regular Ethernet, and there are some scheduling protocols such as IEEE 802.1q that allow a certain amount of quality of service at the MAC level, this is far from ideal for safety critical applications. All these reasons require a new deterministic standard.

All the above mentioned issues are addressed under the deterministic Ethernet termed ARINC specification 664, part 7 or AFDX. AFDX is proprietary and registered to Airbus.

C. Network Topology

The architecture of the AFDX consists of certain principal components that define its behavior.

- The **Avionics Subsystem** comprises all the traditional sensors such as tire pressure and computer systems such as flight control and global positioning and navigation.
- **End system** The End system provides an interface between the Subsystem and the switch. It also consists of an Application Programming Interface (API) that connects the sub-system's readings to the main bus and allows it to communicate with other subsystems.
- **AFDX Interconnect** is an interconnection of switches that communicate with different areas of the network

One of the drawbacks of the Ethernet is the way messages are handled using half duplex wiring. Since there is only one cable for receive and transmit purposes, the possibilities of

Fig. 7. Network Topology in AFDX

collision become much higher. Several techniques have been introduced to avoid this, and these include the CSMA/CD, CSMA/CA etc. This contention is tolerable for non time-critical applications, but needs overhaul for time critical applications.

The AFDX uses full duplex switching. A key change in this architecture is that the Transmit and receive wires are separated. The end systems, which are connected to a particular sensor or sub-system such as autopilot, are in turn connected to two wires communicating transmit and receive messages to and from the switch. The switch in turn connects this information to the Input/ Output processing unit (CPU) as shown in Figure 7.

Since switches are the primary receivers and senders of frames in the network, it is also important to deal with frames that overflow. If a specific switch receives too many frames, they will be buffered at the switch. However, this might create congestion based delays that increase the time it takes for messages to reach their destinations, i.e. it increases the *jitter*. Jitter is defined as the maximum amount of time it takes for messages to reach their destination from source. The time taken for message transfer is called **Latency**. Hence, jitter is the maximum latency.

Since determinism in time is crucial to keep up real time communications for critical applications, a detailed analysis of jitter needs to be performed.

AFDX specifies jitter under the following framework.

$$MaxJitter \leq 40\mu s + \frac{\sum_{i \in set of VLS} (20 + LMax_i) \cdot 8}{N_{bw}}$$

$$MaxJitter \leq 500\mu s \quad (1)$$

N_{bw} is the speed of the Ethernet link in bits per second [13].

D. Frame Structure

The concept of **Virtual Links** is central to message communication between nodes in the AFDX architecture as shown in Figure 8. Since two channels are used, namely transmit and receive, collision rates are drastically reduced. However, further Quality of Service is required to ensure timely delivery of frames of different importance levels. AFDX accomplishes this using three properties integrated

Fig. 8. Virtual Links and Routing in AFDX

in every frame. A virtual channel is setup every time a frame needs to translate across the network. This channel has three properties, **Bandwidth Allocation Bag (BAG)**, **Maximum L2 Frame Size** and **Bandwidth limit**.

Maximum L2 frame size is also called LMAX, and is used to define the maximum frame size. The complimentary LMIN defines the smallest size a frame may take. Ideally, information is being sent as a single frame with size $LMIN < size < LMAX$ and does not need to be split into separate frames. If calibrated properly on a case by case basis, there is no re-ordering of packets, and the frame reaches its destination in one single container, removing considerable non-determinism introduced through packet switching through the routers [2].

The other problem that needs resolving is buffer overflow in switches and end systems. When too many messages are waiting to be routed, they are queued in a buffer. If the buffer overflows, the messages that come after the buffer overflow are discarded. For critical applications, this is obviously very inconvenient. To address this issue, the Bandwidth Allocation Gap (BAG) was introduced [4].

BAG simply defines the time period between two frames for a specific Virtual Link. The delay can be defined between 1-128 milliseconds. The introduction of BAG based messaging has been reported to dramatically increase efficiency and reduces network throughput [4] [14]. Even if BAGs and LMAX are used inefficiently, i.e. if the LMAX (amount of transmitted data per frame) is low and the BAG (time between frame transmission) is high, low network throughput is observed.

Since the VL has been defined between a source and target, no MAC Address is required. This space is taken up by the a combination of legacy MAC IDs from pre-existing Ethernet structure based devices and a *Virtual Link Identifier (VLID)*. The VLID occupies the final 16 bits of the space designated to MAC addresses. This integrability is of considerable financial importance. Commercial off the Shelf (COTS) for regular Ethernet devices can then be utilized to make

a Deterministic Ethernet structure under this framework, reducing the cost.

Only any one subsystem can send data using the VLID making communication unidirectional, although the frame can be sent to multiple recipients.

One detail that needs to be resolved is the communication apparel between End systems and Sub-systems and how frame data originates. The communications between Subsystems and associated instruments occur through an ARINC standard, ARINC 653, which uses an Application Programming Interface (API), called the APplication EXecutive (APEX). The APEX has been designed to decouple real time data communication with the applications that utilize them. Very briefly, ARINC 653 partitions memory spaces for each application (instrument). Multi-tasking, the capability of a memory space or computer to run several tasks on a set period of time and interrupt each other, is also allowed. In effect, the arrows that are shown communicating between Controllers, sensors and actuators with the Subsystem in Figure 7 are handled by the ARINC 653 [16].

Two types of ports, sampling and queuing ports, collectively called *communication ports*, are assigned using the ARINC 653 standard. Sampling ports require 8 KB of data to be handled. For transmission, both act the same way. For receiving, the sampling port over writes the current message being held, while the queuing port has a FIFO queue and the message gets added on to the queue. Communication ports communicate between the sub-systems. The subsystems then communicate with end systems that are designed to deliver messages outside or receive messages from other end systems. Each communication port is attached to a VL, and this helps the transfer of information, creating a virtual pathway for delivery of messages. BAGs and LMAX need to be assigned as needed for a specific network and its applications and the requirements of the nodes of each link. After this, scheduling is performed on the basis of the BAGs and LMAX set before at each end systems independently. End systems also have another function, which is to ensure that VLs do not exceed bandwidth limits. Once a VL has been created, it will remain in the system until network devices are re-initialized [5].

Every AFDX frame has to transmit at least 84 bytes no matter what type of communication. In Equation 1, the term 20 refers to the added bytes added on to any frame, 8 bytes for the preamble and Start Frame Delimiter (SFD), and 12 bytes for the Inter Frame Gap (IFG). $40\mu s$ is the minimum fixed technical jitter. Therefore, a maximum AFDX frame has 1538 bytes for a payload of 1471 bytes of information.

ARINC 664 describes that it is the duty of the system integrator to enforce the maximum $500\mu s$ jitter hard limit. Once the end system has sent the packet out, the physical layer transports the frame to the switch it is connected to. The switch adds to the latency since the frame has to wait in the FIFO (First In First Out) type queue for other frames to be transmitted first.

Fig. 9. An AFDX frame and its constituent data byte distribution, taken from [5]

Fig. 10. How redundancy is scheduled in AFDX, taken from [2]

Finally, jitter is calculated for two scenarios with respect to frame size and message communication.

- If the frame length is less than LMAX and are produced at frequencies lower than BAG of the VL, the maximum latency,

$$MaxLatency \leq BAG + MaxJitter + L_T$$

- If the message is too large for a frame and needs to split up, i.e. the LMAX is lesser than the communication size, there could be other packets at the switch waiting to be transmitted. If p is the number of packets waiting, then the Latency is

$$MaxLatency(p) \leq p \cdot BAG + MaxJitter + L_T$$

ARINC 664-7 specifies that the value of $L_T < 150\mu s$ and should be bounded at the switch. In the exact same way, on the receiving end, the value of L_R (Receiving Latency) should also be less than $150\mu s$

E. Safety and Redundancy

Both availability and reliability are key concepts in the AFDX configuration. AFDX provides sophisticated redundancy using end system and cabling redundancy using which two networks, A and B are provided. Every sub-system is connected to two end systems and the end systems deliver separate frames using separate networks. Hence, two frames arrive at the destination bearing the same information, given no error has occurred. This is shown better in Figure 10. This means duplicate frames need to be discarded. To detect the duplicate, a *Sequence number* of 1 byte is added on, as shown in Figure 9 as SN. [4]

V. COMPARISONS

An analysis of the implementation of the two types of Networks shows us different adaptation to the Ethernet philosophy.

- While hot swap of COTS components is supported by both TTEthernet and AFDX, there are varying degrees of extra work that are needed to implement determinism in latency. TTE requires a new schedule in order to add a new component. The AFDX requires a new schedule to make the switch aware of the added component. All this is in order to maintain and certify time determinism as pertains to ARINC 664/P7 standards.
- In AFDX, Rate Constrained traffic establishes a theoretical upper bound, equivalent to the RC traffic in TTE. TTEthernet establishes three classes, TT, RC and BE of varying criticality. BE is the equivalent of Layer 2 Ethernet, without guarantees for delivery. TT is used for highly critical applications and RC and BE are event driven components
- Demarcation of TT messages to Event driven ones in TTE allow critical messages to stay independent of Event triggered messages (RC and BE) which are usually handled by cheap COTS components and Line Replaceable Units (LRUs).
- TTEthernet utilizes the same frame structure as that of the Ethernet. Legacy Ethernet architectures can be more easily modified for critical tasks. The AFDX utilizes a specialized frame that would require specialized hardware for transmitting information.

A detailed comparison is also further provided in Section VII for added comparative understanding of the two protocols.

VI. CONCLUSION

In this paper, we sought to explain in a holistic manner, how AFDX and TTEthernet based Distributed control architectures are implemented. We consider these protocols to be of importance and better than other protocols due to the fact that these protocols incorporate maintenance life cycle concerns by basing themselves on the publicly well known Ethernet protocol utilizing Commercial Off The Shelf components.

However, these protocols differ in their philosophy and implementation, which is why a detailed comparative analysis of their strengths and weaknesses is important for optimal implementation specific to the use case. In terms of the philosophy of message forwarding, however, it is important to note that TTE with its tiered system of Time Triggered (TT) and Rate Constrained (RC) at tier 1, and Best Effort (BE) at tier 2, is a highly robust system that encompasses AFDX's philosophy, and can be considered superior to AFDX. In short, TTE combines the best of both Time triggered and Even triggered communication. This makes it highly suitable to extremely critical communication with redundancy, thus its implementation in space crafts such as Orion and Ariane-6.

VII. TABLE OF COMPARISON

Feature	AFDX	TTEthernet
Description	It is a standard that specifies electrical and Protocol specifications using IEEE 802.3 and ARINC 664/Part 7 for QoS	Synchronous and Fault tolerant Distributed Systems Architecture for safety critical systems Governed by specifications from TTTech and ARINC 664/Part 7
Application	Developed for Aircraft Data Networks	Aerospace Applications for Spacecraft (Orion) and Ariane-6 (Launcher)
Deployed Systems	Airbus A380, A400M Boeing 787, Augusta Westland Helicopters	NASA Orion and ESA Ariane-6 Launcher
Communication Type	Event Triggered, Bandwidth defined by hard limits	Time Triggered, independent of host, masterless
Max. data rate	100 MBps	100 MBps / 1GBps(TT-GbE)
Media Access	Filtering at Switch, frames are shaped by defining BAG and VL	TT messages are pre-defined. RC and BE follow standard Ethernet techniques
Clock Synchronization	No	Follows masterless synchronization protocol that is unique
Latency Jitter	Less than 500 micro-seconds	Less than 1 micro-second for 1GBps TT message implementation
Physical Layer/ Independence(Y/N)	Twisted Pair, Ethernet 100BaseT MAC. Not Independent	Ethernet using COTS, RS 485, Copper Wire, Fibre Optics. Independent
Topology	Switched Fabric	Multi-drop and Point to Point
Physical Layer Length	100 m between Tx and Rx	30 nodes for 100 m, cable length of up to 1 km
Supports Hot Swap of same type node	Yes	Yes
Fault Containment	Yes if the fault is contained to one dual redundant switched network	Yes. For Hardware, message status, cabling
Babbling Idiot Avoidance	Yes, Detected and suppressed by Switch	Yes, by the bus guardian
Node Failure Reporting	Yes	Only nodes expecting data are aware at application level
Error Handling Approach	Message loss is taken care of by redundancy and duplication. Fail silence for protocol error, Restart with test	Smart Switch for failure localization and discarding bad messages. Switch enters quiet mode for catastrophic failure
Ease of Expansion	Requires design of new switch configuration data table	Requires new schedule
IP Available for ASIC/FPGA Implementation	Yes but limited	Yes
Notes	Trademark of Airbus	Open source implementation available. Main supplier of hardware and software implementation is TTTech

REFERENCES

- [1] Astrit Ademaj. Time-triggered ethernet – a powerful network solution for multiple purpose. *TTTECH*.
- [2] Len Buckwalter. *Avionics Databases*. Avionics Communications Incorporated, 2008.
- [3] Bruno Cattelan and Steffen Bondorf. On delay bounds and measurements: A cots testbed for network performance experimentation. In *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 1133–1138. IEEE, 2020.
- [4] Christian M Fuchs et al. The evolution of avionics networks from arinc 429 to afdx. *Innovative Internet Technologies and Mobile Communications (IITM), and Aerospace Networks (AN)*, 65:1551–3203, 2012.
- [5] J Javier Gutiérrez, J Carlos Palencia, and M González Harbour. Response time analysis in afdx networks with sub-virtual links and prioritized switches. *XV Jornadas de Tiempo Real*, pages 1–18, 2012.
- [6] DA Gwaltney and JM Briscoe. Comparison of communication architectures for spacecraft modular avionics systems. 2006.
- [7] Hermann Kopetz. The rationale for time-triggered ethernet. In *2008 Real-Time Systems Symposium*, pages 3–11. IEEE, 2008.
- [8] Hermann Kopetz, Astrit Ademaj, Petr Grillinger, and Klaus Steinhammer. *The time-triggered ethernet (TTE) design*. IEEE, 2005.
- [9] Andrew Loveless. Ttethernet for integrated spacecraft networks. 2015.
- [10] Quang Nguyen, William Yuknis, Scott Pursley, Dennis Albaijes, Noosha Haghani, and Omar Haddad. A high performance command and data handling system for nasa’s lunar reconnaissance orbiter. In *AIAA SPACE 2008 Conference & Exposition*, page 7926, 2008.
- [11] John Rushby. Partitioning in avionics architectures: Requirements, mechanisms, and assurance. Technical report, SRI INTERNATIONAL MENLO PARK CA COMPUTER SCIENCE LAB, 2000.
- [12] Nov SAE AS6802. Sae standards, 2011.
- [13] Nour El-Din Safwat, MA El-Dakroury, and Abdelhalim Zekry. The evolution of aircraft data networks. *International Journal of Computer Applications*, 94(11), 2014.
- [14] Teresa Schuster and Dinesh Verma. Networking concepts comparison for avionics architecture. In *2008 IEEE/AIAA 27th Digital Avionics Systems Conference*, pages 1–D. IEEE, 2008.
- [15] Wilfried Steiner, Flavio Bonomi, and Hermann Kopetz. Towards synchronous deterministic channels for the internet of things. In *2014 IEEE world forum on Internet of Things (WF-IoT)*, pages 433–436. IEEE, 2014.
- [16] Steven H VanderLeest. Arinc 653 hypervisor. In *29th Digital Avionics Systems Conference*, pages 5–E. IEEE, 2010.