TTP and CAN Buses: Evaluation of Automotive Products Applied to the Aeronautic Products on an Aircraft Fly-By-Wire System Architecture

Herminio Duque Lustosa
Empresa Brasileira de Aeronáutica-EMBRAER

Marcelo Lopes de Oliveira e Souza
National Institute for Space Research-INPE

ABSTRACT
A major trend in modern aerospace and automotive systems is to integrate computing, communication and control into different levels of the vehicle and/or its supervision. A well fitted architecture adopted by this trend is the Common Bus Network Architecture. A Networked Control System (NCS) is called when the control loop is closed through a communication network. The presence of this communication network introduces new characteristics (sharing bus, delays, jitter etc.) to be considered at design time of a control system.

This work focuses on the influences of automotive data bus protocols on an aircraft Fly-By-Wire (FBW) networked control system. We intend to show, through simulations, the influences of sharing bus on a real time control system. To compare effects, we choose the CAN Bus protocol where the medium access control is event-triggered; and the TTP protocol where the medium access control is time-triggered, that have been used by the automotive industry in large scale. Progressively, both technologies are taking space in the aeronautic system architectures.

We intend to extend this study to a simplified model of an aircraft actuation system, composed by ACE (Actuator Control Electronics) and a “smart” Electro-Hydraulic Servo Actuator as part of a Flight Controls System based on a Fly-By-Wire technology.

Among the results of this work, we expect to obtain metrics comparing CAN Bus and TTP influences and FBW control loop for elevator, in order to trade off them and simultaneously validate the applied simulation tools.

INTRODUCTION
AN OVERVIEW OF THE SYNERGY BETWEEN THE AUTOMOTIVE AND AERONAUTIC INDUSTRIES – Traditionally, the automotive industry contemplated additional functions based on mechanical and hydraulic components. With the advance of electronic devices, these absorbed the introduction of these additional functions and become even more involved in the essential functions, like, break, steering, air-conditioning etc.

These facts fed back the evolution of the automotive products, stimulating the development of new functions with a higher level of performance and with significant cost and size reduction. In consequence, the x-by-wire system appeared, bringing together the concepts from the Aerospace and Aeronautic industries.

Progressively, the automotive industry demanded an increasing number of products with more dynamics. Components Of The Shelf (COTS), like Microprocessors (MPs), Digital Signal Processors (DSPs), and Programmable Logic Devices (PLDs) were created under a standardization process to assure the quality of the products according to the criticality of the functions executed by them.

Today, we are watching an inverted process of diffusing the technology, i.e., the technology developed by the automotive is migrating to the aeronautic industry, e.g., the COTS exemplified above, providing reduction of costs and shorter periods of development front a robust solution of engineering.

Figure 1 shows an automotive system controlled by some electronic devices, which need some functional, computational and communication requirements well defined to be integrated to the entire vehicle. A similar approach is observed in the modern and highly integrated airplanes, e.g., EMBRAER 170.

The automotive and aeronautic industries offer vehicles with large number of functions, in this way they can structure their system architectures distributed as efficient as possible, with the objectives of reducing costs, diagnostic and maintenance times, via constant improvements of the offered products.

Figure 2 shows the tendency of the number of Electronics Control Units (ECUs) used in some German vehicles, where it can be observed that the growing is exponential. And a similar statistics can be
observed in the aeronautic industry; however this research is not available for public domain.

Figure 1 – Automotive systems controlled by electronic functions [1].

Figure 2 – Tendency of the number of ECUs in German vehicles [1].

NETWORKED CONTROLS SYSTEM – To integrate all this technology it is necessary to go back to basics and understand very well how to define a Control System.

In a classical closed-loop control system, as shown in Figure 3, the sensors are directly connected to the controller which is responsible to adequate the plant to the desired input (reference).

In a classical closed-loop control system, a sensor monitors the output and feeds the data to a controller which continuously adjusts the control input as necessary to keep the control error to a minimum. Feedback on how the system is actually performing allows the controller to dynamically compensate for disturbances on the system. An ideal feedback control system cancels out all errors, effectively mitigating the effects of any forces that may or may not arise during operation; and producing a response in the system that perfectly matches the user’s wishes.

Figure 3 – A Classical Control System Diagram.

In such cases there are many concerns to specify the sensor considering type, frequency response, distance from the controller etc., to reach the desired performance of the system and keep intrinsic characteristics like stability and controllability.

In a closed loop control system, the output of the system is fed back to the reference value, through a sensor measurement. The controller takes the error (difference) between the reference and the output to change the inputs to the system under control. This is shown in the Figure 3. This is called a Single-Input-Single-Output (SISO) control system; Multi-Input-Multi-Output (MIMO) control systems, are also common. In such cases, variables are represented through vectors instead of simple scalar values.

More than ever, the requirement for the control systems are becoming more restrictive and more demanding, mainly for embedded systems, demanding more sensors, more computer capability and processor speed, which makes the systems more complex. In this scenario, the databuses have arisen as a solution to concatenate a huge amount of information. They are a very important addition to the classical architecture, giving rise to a new architecture named a Networked Control System, as shown in Figure 4. This feature increases the system in terms of complexity, costs, etc.; but it reduces weight and development/manufacturing time. Nevertheless, they introduce some characteristics like latency, jitter and new failure modes that shall be addressed in any control system design, solution and certification to be compliant with the life or mission-critical requirements.

MAIN CONCEPTS [2]

DATABUS - In Networked Control System (NCS) architectures, a bus is a subsystem that transfers data between computer components inside a computer or between computers. Unlike a point-to-point connection, a bus can logically connect several peripherals over the same set of wires. Each bus
defines its set of connectors to physically plug devices, cards or cables together.

An event can be defined as "a significant change in state". For example, the activation of an emergency button.

This architectural pattern may be applied to the design and implementation of applications and systems which transmit events among loosely coupled software components and services. An event-triggered system typically consists of event consumers and event producers. Event consumers subscribe to an intermediary event manager, and event producers publish to this manager. When the event manager receives an event from a producer, the manager forwards the event to the consumer. If the consumer is unavailable, the manager can store the event and try to forward it later. This method of event transmission is referred to in message-based systems as "store and forward".

Building applications and systems around an event-driven architecture allows these applications and systems to be constructed in a manner that facilitates more responsiveness, because event-triggered systems are, by design, more adequate to unpredictable and asynchronous environments.

Event-triggered architectures complement service-oriented architectures because services can be started by triggers such as events.

Computing machinery and sensing devices can detect state changes of objects or conditions and create events which can then be processed by a service or system. Event triggers are conditions that result in the creation of an event.

In an event-triggered databus the acknowledgment of the reception is explicit; the receivers send explicit acknowledgement messages after either receiving or not receiving a message.

Nowadays, there is a new tendency by an Automotive Consortium, called FlexRay, where they mixed both, event and time-triggered technologies, creating a new approach for NCS, which will not be exercised herein.

**DATABUS CHARACTERISTICS** - In this section we will describe the main characteristics that will be the focus of this work.

**Latency** - is the time interval between the start of transmission at the transmitter and the message delivery at the receiver, as shown in Figure 6. It can depend on many factors like: transmission speed, cable lengths, bus load, etc.
Jitter - is the variation in the time between packets sending and arriving, caused by network congestion, timing drift, or route changes, as shown in Figure 7.

The jitter introduces and additional measurement error, as shown in Figure 8. Then, the application must assume worst-case jitter for hard real-time systems.

Peak load handling - Peak load can happen when all nodes on the shared bus require communication services at the same time; or when they send maximum amount/length of data, highest priority messages. In this scenario, the actual load and the worst-case message delays shall be considered in the NCS design to avoid non-expected problems.

Then the communication system for hard real-time applications must be designed to guarantee message delivery for all hard real-time messages within the deadlines, even in a peak load scenario (which typically occurs in a critical situation anyway), and for safety-related applications even in a faulty scenario. Up to now, time-triggered communication is the only acceptable solution.

The design shall avoid the possibility of thrashing, which is an abrupt decrease of throughput that occurs with an increase of system load, see Figure 9.

The characteristics listed before may affect the data congruence, which consists in having the data congruent in amplitude and in time. If the system can manage the data congruence, it is expected to avoid some problems in NCS.

The control system shown in Figure 10 was evaluated with no congruence of data in time and amplitude, and the results are shown in Figure 11 and Figure 12.
In such example, it is important to say that the Flight Control Computer (FCC) is based on a Proportional-Integral Controller (PI). In other words, there is an integrator not resettable that integrates the error of input commands. One of the main objectives of this work is to evaluate the impact of some parameters of time and event-triggered databases in the data congruence.

ARCHITECTURE OF AN ELECTRONICS CONTROL UNIT (ECU) – An automotive or aeronautical ECU is based in a computational system where the architecture of hardware and software is composed by layers: Hardware, Real-Time Operational System, units of Inputs/Outputs (I/Os), Network and Application, where the functions are designed. Figure 13 presents an illustration of a generic architecture; however the level of software and hardware can vary in accordance with the application.

For distributed system architecture to execute the functions designed in accordance with the system requirements, the ECUs shall be connected through a communication databus, which allows the tasks to communicate among them.

APPLICATIONS – With the focus based on achieving fault tolerance for ultra-reliable real-time systems, one way to define the reliability requirements for these systems and to distinguish them from other fault-tolerant applications is to measure them in terms of a maximum acceptable probability of failure. As the applications are totally based on the correct operation of the system, the acceptable probability of failure of the system and its components (sensors, computers, actuators and databases) is very small, typically in the range of $10^{-4}$ to $10^{-10}$, depending on the consequences of the failure. They split in:

Safety-critical applications – A safety-critical application is an application whose failure or malfunction may result in:
- death of or serious injury of people, or
- loss or severe damage of equipment or
- environmental harm.

Risks of this sort are usually managed with the methods and tools of safety engineering based on ARP4761 [4]. This type of application is the most demanding for reliability, e.g., Flight Control Systems based on FBW technology applied in commercial transportation, which is designed to have a probability of failure per flight hour of less than $10^{-6}$. This type of application has to be compliant with the requirements imposed by the Airworthiness Authorities (Federal Aviation Administration (FAA) in USA, Agência Nacional de Aviação Civil (ANAC) in Brazil, European Aviation Safety Agency (EASA) in Europe and so on) to demonstrate a reasonable level of safety before entry into service. Similar applications in military aircraft are several orders of magnitude less demanding, with a probability of failure per flight hour typically around or less than $10^{-7}$ (presumably because the crew can bail out) [5].

Vehicle-critical applications – In this case, the cost of the failure is huge economic penalty rather than loss of life (such as Unmanned Aircraft Vehicles, autonomous submarines and satellites). They require probabilities of failure per hour of operation less than $10^{-6}$ to $10^{-7}$. The time response required for these applications are very demanding, e.g. a statically unstable fighter aircraft can develop divergent flight modes if correct control inputs are not applied every 40 to 100 ms [5].

Mission-critical applications – In which a failure of equipment, e.g. computer, can cause an incomplete or aborted mission. They occupy the low end of the ultra-reliable spectrum. Typical probabilities of failure per hour of mission are less than $10^{-4}$ to $10^{-5}$. These applications do not have so stringent response time as required for safety critical and vehicle-critical ones, but typically they need higher throughput to process more functions and amount of data, e.g., satellites [5].

Figure 12 - Effect of drift in amplitude of the input in an Aircraft Control System.

Figure 13 – Generic ECU architecture [1].
On-line transaction processors (OLTP’s) – by contrast this type of application demand high availability, i.e. uptime, rather than high reliability (i.e., correct operations). Incorrect operations in these applications can usually be found through audits and rolled back after the fact. OLTP applications can withstand a delay of seconds to process transactions. In any event, the penalty for slow response is far from becoming a catastrophe. The validation of these applications is not so formal, although it is relatively more expensive to fix hardware and software design errors in the field than during production [5].

TOPOLOGIES - There are two types of data communication between processors and between processors and peripherals: channels and networks. A channel provides a direct or switched point-to-point connection between communicating devices. Channels are usually hardware intensive and provide high bandwidth with low overhead. A network is a collection of processors and peripherals that interact with each other using a protocol. Networks require more software to handle the communication but can be used to a larger variety of tasks. Each communication technique has its advantages and disadvantages; and within a network different topologies, protocols and media are used [7].

The most common topologies are point-to-point, bus, star, ring and combinations of these. In avionics, point-to-point topologies are dominant because of their superior reliability, which turn it easier to be certified. Over the last decades the focus on environmental issues of the aviation industry has grown. Due to this, weight reduction is a big issue today. Large savings can be made if a sparse (partly connected) mesh or bus can be used instead of a fully connected mesh topology.

Point-to-point interconnection - All point-to-point topologies are shown in Figure 14.

Point-to-point interconnects can be configured in many different ways. A fully connected point-to-point communication system (fully connected mesh) has a topology where all nodes have point-to-point interconnections between themselves and all the other nodes in the network. The communication capability of this topology is very high because messages can be transmitted in parallel on all communication paths. It is a very robust topology that has extensive redundancy at the cost of being very hardware-intensive. The complexity increases fast with the number of nodes and a reconfiguration is difficult to achieve. To reduce weight and complexity of the cost of reliability, it is possible to use a mesh that is similar but not fully connected (partly connected mesh). This: decreases the reliability, turns some nodes intermediate between orders, and jeopardizes the determinism of the network.

A ring topology is a point-to-point connection where every node has two connections to other nodes so that all nodes form a ring. Messages are sent in one direction and repeated by all nodes until it reaches the sender again. In order to tolerate a single fault on the ring it is possible to exchange communication direction. A system with redundant rings and re-directional transmission is very robust.

An advantage with a point-to-point topology is that it creates physical separation between communication channels. Separation between the channels creates natural fault containment zones, avoiding the problem to kill the communication when one node decides to talk all the time (babbling-idiot effect).

Bus interconnection - The bus interconnection is shown in Figure 15.

Bus interconnection

The broadcast bus topology consists of a medium (which in principle can be an electric wire, optical fiber or radio link) to which all nodes are connected. Buses always transmit broadcast messages to all members on the bus. Because of this, only one member can send at a time despite the fact that there usually exist several paths within the bus. Since all nodes will see the signal at, virtually, the same time on the bus, timing analysis becomes easy compared to making an analysis of a complete point-to-point communication system.

The bus topology is flexible and composable and can easily be reconfigured without any major hardware
redesign. Buses generally utilize less wiring and interfaces than point-to-point and star topologies. Since all nodes are connected to the bus there is a single point of failure, issue that must be handled. Fault containment needs to be created using mechanisms such as membership to prevent that faulty members disrupt the communications between other nodes on the bus, e.g. the bus guardian implemented in the TTP bus [7].

The protocols supporting bus topologies in safety-critical communication systems available today are limited in bandwidth compared to point-to-point topologies. This comes from the fact that on a bus all nodes share the bandwidth while every point-to-point is dedicated to the communication between two nodes. A bus topology is considered very reliable since it is a passive component with few fault modes [7].

In addition, the bus topology is robust and makes the development and certification tasks easier, considering that you can plug a perfect listener to monitor all traffic of information which turns the lab and flight test instrumentations much simpler and cheaper.

The two types of databuses evaluated herein can be configured as bus topology and the analysis performed is based also in this configuration.

**Star interconnection** - The star interconnection is shown in Figure 16. A star-coupled system relies on one central node, a star or a hub, that is connected to all other nodes. The star is the obvious master of the communication since there are no direct connections between the other nodes. The central position puts the star in charge of all communications; faulty nodes can be excluded since it controls the access to the communication medium. In a star topology it is natural to implement the guardians in the star coupler since this makes them physically independent from the node [7].

However, if the star breaks down or becomes faulty, then the whole network will experience loss of communication. The probability that this fault mode can cause total communication blackout can be significantly reduced by using redundant stars.

Star topology is a hardware intensive topology that requires more wiring than most other topologies; an exception is the fully connected mesh. Additionally, the star coupler is much more complex and has larger failure rate than a passive component like a cable in a mesh or bus interconnection. Delays are increased when using a star coupler since the routing at the switch is not instantaneous [7].

The star topology is inflexible to hardware reconfigurations since the star couplers need to be reconfigured. One advantage a star topology has over a bus is that the physical layers that are available for star topologies today supports higher bandwidth. TTP/C using 100Base-TX has the maximum bandwidth of 25Mbit/s [7].

There is the possibility to use a passive star, which just re-adequates the signals in amplitude to compensate the losses in the harnesses.

**DATA BUSES EVALUATION**

This section provides a brief overview of the databuses that are considered for implementation into current life-critical and mission-critical control systems.

**Controller Area Network (CAN Bus)** - CAN is a network bus protocol jointly developed by Bosch, Philips Components and Intel in the mid 1980's for the automotive industry. It is now used worldwide in the automotive industry and has many applications in the aeronautical industry. It provides real-time data communication, with reduced cable size and weight, and a normalized input/output specification [8].

CAN 2.0 is divided into two parts, A and B. Part A has a standard 11-bit identifier field, no specification for message filtering, and layered architecture based on Bosch’s internal model. Part B extends the identifier to 18-bits, has some message filtering, and supports the layer description of the Open System International (OSI) model.

CAN bus has been subdivided into different layers: the object layer, the transfer layer, and the physical layer. The CAN specification focuses on the first and second layers of the OSI model (data link layer and physical layer). A variety of protocols are...
used between the CAN data link layer and the various application layers.

CAN implements five error detection mechanisms: Cyclic Redundant Checks (CRC), frame checks, acknowledgment error checks, bit monitoring, and bit stuffing.

The key attributes of CAN are [8]:

- CAN uses multi-master, priority-based, serial communications protocol;
- CAN supports distributed, real-time protocol and multiplexing, using non-destructive contention-based arbitration;
- CAN has maximum bit rates of 1 Mbps (with 40 m bus length), 500 kbps (with 90 m bus length) or 100 kbps (with 500 m bus length);
- CAN uses No Return to Zero (NRZ) encoding;
- CAN has message length of 0 to 8 bytes;
- CAN offers the following classes of service: periodic, sporadic, and time-triggered;
- CAN uses Carrier Sensing Multiple Access/Collision Resolution (CSMA/CR) for media access;
- CAN topology is a terminated differential two-wire bus;
- CAN media used is screened or unscreened twisted pair or flat pair telephone cable.

**APPLICATION**

**FLIGHT CONTROL SYSTEM (FCS)** - One of the objectives of this work is to apply the investigated concepts into a safety-critical project, chosen as a FCS with the FBW technology. As an entire FCS is too complex, it was decided to opt for a simplification as described below, adopting just the control of the elevators (that is totally representative of it) to exercise the concepts described herein.

The main goal of the system herein named Simplified Elevator Control System is to control the two elevator surfaces of a hypothetical aircraft as described by an overview representation in Figure 17.

![Figure 17 – Simplified Elevator Control System Overview.](image)

Where:
- FCC - Flight Control Computer
- OB - outboard
- IB - inboard

The system is composed of Inceptors (Sidesticks), Sensors, Controllers (FCCs) and Actuators. The system is powered by hydraulic and electrical power sources.

Inceptors are used to capture commands or intents from pilots which are processed in the controller as well as inputs from sensors such as rate gyro. After processing these signals according to a specific control algorithm, the controller sends a command signal to the actuator via electrical wiring, which in turn will drive the surface movements.
Physically speaking, the Controller, herein called Elevator Controller, is formed by two closely coupled computers and network interfaces: The COM computer reads the inputs, processes them in accordance to the specific control laws implemented in software for the elevators operation, generates the corresponding elevator angle and reads the aircraft’s pitch rate to verify if the desired aircraft command has been processed correctly. On the other hand, the MON computer processes the same information as the COM computer, reads the outputs generated by this other computer and compares them with the outputs generated by it. Disagreement between them may result in the actuator being commanded to the neutral position and both computers being disengaged.

METHODOLOGY

SIMULATION TOOLS - The effects of temporal non-determinism on control performance are often very hard, if not impossible, to investigate analytically. A natural approach is then to use simulation instead of that. However, today’s simulation tools make it difficult to simulate the true temporal behavior of control loops. What is normally done is to introduce time delays in the control loop representing average case or worst case delays [9].

In this work the new simulation toolbox TRUETIME is applied. TRUETIME, which is based on MatLab/Simulink©, refer to Figure 18, makes it possible to simulate the temporal behavior of a multitasking real-time kernel containing controller tasks. The controller tasks control processes modeled as ordinary Simulink blocks.

![TrueTime interfaces to the MatLab/Simulink©](image)

Figure 18 - TrueTime interfaces to the MatLab/Simulink©.

Different scheduling policies may be used, e.g., priority driven or deadline driven schedulers. The execution times of the controller tasks can be modeled as being constant or time varying, using some suitable probability distribution. The effects of context switching and interrupt handling are taken into account, as well as task synchronization using events and monitors. With TRUETIME it is also possible to simulate the timing behavior of communication networks used in, e.g., NCS, which is the focus of this work [9].

TRUETIME can be used for several purposes: to investigate the true effects of timing non-determinism on control performance; to develop compensation schemes that adjust the controller dynamically based on measurements of actual timing variations; to experiment with new, more flexible approaches to dynamic scheduling; and to simulate event and time-triggered control systems [10].

The TRUETIME simulation environment offers two Simulink blocks: a computer block and a network block, the interfaces of which are shown in Figure 18. The input signals are assumed to be discrete, except the signals connected to the A/D port which may be continuous.

The TrueTime Network - The TRUETIME network block simulates medium access and packet transmission in a local area network. When a node tries to transmit a message, a triggering signal is sent to the network block on the corresponding input channel. When the simulated transmission of the message is finished, the network block sends a new triggering signal on the output channel corresponding to the receiving node. The transmitted message is put in a buffer at the receiving computer node. A message contains information about the sending and the receiving computer nodes, arbitrary user data (typically measurement signals or control signals), the length of the message, and optional real-time attributes such as a priority or a deadline [10].

Six simple models of networks are supported: CSMA/CD (e.g. Ethernet), CSMA/AMP (e.g. CAN), Round Robin (e.g. Token Bus), FDMA, TDMA (e.g. TTP), and Switched Ethernet. The propagation delay is ignored, since it is typically very small in a local area network. Only packet-level simulation is supported—it is assumed that higher protocol levels in the kernel nodes have divided long messages into packets, etc.

CSMA/AMP (CAN) - CSMA/AMP stands for Carrier Sense Multiple Access with Arbitration on Message Priority. If the network is busy, the sender will wait until it occurs to be free. If a collision occurs (again, if two transmissions are being started within 1 microsecond), the message with the highest priority (the lowest priority number) will continue to be transmitted. If two messages with the same priority
seek transmission simultaneously, an arbitrary choice is made as to which is transmitted first. (In real CAN applications, all sending nodes have a unique identifier, which serves as the message priority) [10].

**TDMA (TTP)** - TDMA stands for Time Division Multiple Access and it works similarly to FDMA, except that each node has 100% of the bandwidth but only in its scheduled slots. If a full frame cannot be transmitted in a slot, the transmission will continue in the next scheduled slot, without any extra penalty. Note that overhead is added to each frame just as in the other protocols [10].

**SIMULATIONS AND DISCUSSIONS**

**SIMPLE CONTROL LOOP** - After reviewing some basic concepts about NCS, we will exercise the influence of some databus characteristics on a simple PID control loop, refer to Figure 19.

![Figure 19 - Simple control loop.](image1)

Figure 19 - Simple control loop.

Analyzing such simple control loop with a square wave inserted in the input command, the response shown in Figure 20 was obtained.

![Figure 20 - Response of the simple control loop to the square wave.](image2)

Figure 20 - Response of the simple control loop to the square wave.

Then the parameters measured to evaluate the control performance are shown in Table 1, in the column "without databus".

**INTRODUCTION OF THE NETWORKED CONTROL SYSTEM CONCEPT** – Now, the databus is introduced to compose a NCS, as shown in Figure 21. Thus, it is expected to have some disturbances in the controller characteristics, as shown in Figure 22, where the Bandwidth was varied up to the loss of control.

![Figure 21 - Introduction of the databus (network) into a simple control loop.](image3)

Figure 21 - Introduction of the databus (network) into a simple control loop.

![Figure 22 - Evaluation of the impact of Bandwidth (BW) changes in a Simple Control System - CAN Bus.](image4)

Figure 22 - Evaluation of the impact of Bandwidth (BW) changes in a Simple Control System - CAN Bus.

In Table 1 are some parameters to measure the performance of a control system for the different values of BW, where the databus is CAN Bus. Then, it was observed that the performance of the control system degrades inversely proportional to the BW, which corresponds direct to the delays introduced in the control loop chain [11].
Table 1 - Parameters evaluated in the NCS for BW variations - CAN Bus

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Without databus</th>
<th>BW for CAN Bus (CSMA/AMP)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1Mbps</td>
</tr>
<tr>
<td>Peak value [deg]</td>
<td>1.032</td>
<td>1.039</td>
</tr>
<tr>
<td>instant of the peak [s]</td>
<td>0.315</td>
<td>0.319</td>
</tr>
<tr>
<td>overshoot [%]</td>
<td>3.20%</td>
<td>3.90%</td>
</tr>
<tr>
<td>Settling time [s]*</td>
<td>0.150</td>
<td>0.157</td>
</tr>
</tbody>
</table>

* in reference to the raising edge of the input

The same exercise was repeated using TTP bus and the results are shown in Figure 23 and the performance evaluation parameters are shown in Table 2.

A disturbance node in the system was introduced, as shown in Figure 24, to evaluate the influences of the occupation rate in the characteristics of the control system.

Table 2 - Parameters evaluated in the NCS for BW variations - TTP Bus

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Without databus</th>
<th>BW for TTP Bus (TDMA)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1Mbps</td>
</tr>
<tr>
<td>Peak value [deg]</td>
<td>1.032</td>
<td>1.056</td>
</tr>
<tr>
<td>instant of the peak [s]</td>
<td>0.315</td>
<td>0.318</td>
</tr>
<tr>
<td>overshoot [%]</td>
<td>3.20%</td>
<td>5.60%</td>
</tr>
<tr>
<td>Settling time [s]</td>
<td>0.150</td>
<td>0.154</td>
</tr>
</tbody>
</table>

* in reference to the raising edge of the input

Figure 24 – Introduction of a disturbance node in the databus of the simple control loop.

Figure 25 shows the plotted curves of the evaluated control system using CAN Bus protocol as the main media of the system. The characteristics of performance were compiled in Table 3, where it is observed that the control is lost with occupation rate of 90% of the BW, which is near the trashing point of this network.

Comparing the results obtained with TTP and CAN Bus, it was observed a better performance of the CAN Bus using a BW of 100 Kbps, which is compatible with the time delays used to tuning the control loop.
Figure 25 - Evaluation of occupation rate of the BW in a simple control system using CAN Bus.

Table 3 - Evaluation of the impacts of the occupation rate of the BW in the performance of a simple control system using CAN Bus.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>% of BW occupied - CAN Bus (CSMA/AMP) - BW = 90 Kbps</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0%</td>
</tr>
<tr>
<td>instant of the peak [s]</td>
<td>0.313</td>
</tr>
<tr>
<td>overshoot [%]</td>
<td>26.00%</td>
</tr>
<tr>
<td>Settling time [s]*</td>
<td>0.243</td>
</tr>
</tbody>
</table>

* in reference to the raising edge of the input

The same exercise was repeated with the same system, replacing the network media by the TTP bus, and Figure 26 shows the plotted curves of the evaluated control system. The characteristics of performance were compiled in Table 4, where it is observed that the control is not lost with occupation rate of 90% of the BW. This was demonstrated by the earlier example using CAN Bus, which is one of more important characteristics of the TTP Bus; i.e., it is not susceptible to occupation rate of the BW, because each message has its time reserved and planned in the network schedule.

Figure 26 - Evaluation of occupation rate of the BW in a simple control system using TTP Bus.

Analyzing the data captured in Table 4, it was observed practically the same control performance characteristics for different occupation rates of BW. This makes the design of the control system easier mainly when it is necessary to integrate it to other control systems. This characteristic is called composability and it is very important for system integration purposes.

Table 4 - Evaluation of the impacts of the occupation rate of the BW in the performance of a simple control system using TTP Bus

<table>
<thead>
<tr>
<th>Parameter</th>
<th>% of BW occupied - TTP Bus (TDMA) - BW = 90 Kbps</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0%</td>
</tr>
<tr>
<td>Peak value [deg]</td>
<td>1.642</td>
</tr>
<tr>
<td>instant of the peak [s]</td>
<td>0.319</td>
</tr>
<tr>
<td>overshoot [%]</td>
<td>64.20%</td>
</tr>
<tr>
<td>Settling time [s]*</td>
<td>&gt;&gt; 0.25</td>
</tr>
</tbody>
</table>

* in reference to the raising edge of the input

Note: all simulations herein used the following software versions:

- MatLab/Simulink – Version 7.3.0.267 (R2006b)
- TRUETIME – version 1.5
CONCLUSIONS

This work traced a parallel between the automotive and aeronautic industries, where many technologies changed in different ways, during different phases of the history of their Engineerings; e.g. the by-wire technology that migrated from the aerospace to the aeronautic and then to the automotive industries; and the COTS that are migrating from the automotive to the aeronautic industries. A high level of commonality and even synergy is observed among all fields of the technologies used, and this allows a very fast evolution looking for better, faster and cheaper results.

This work compiled very important information to introduce Networked Control Systems theoretically, associated with practical cases used in the Aerospace Industry. These exemplified the influence of some communication parameters on the control system performance. Among them were Bandwidth values and Occupation Rate of the Bandwidth, which affect directly the time delays introduced by the network in the system control loop.

Taking two different approaches of communication and architecture (event-triggered and time-triggered), it was observed that the time-triggered philosophy provided more robustness to the cases used in terms of bus traffic, integration changes and redesigns. Other issues to be addressed in future works are the influence of jitter, etc., on performance. This can also be simulated by a toolbox called Jitterbug.

Other important topic addressed in this work is the importance of simulation tools to support Network Control System design to anticipate the problems before the system implementation, which saves a lot of time and money. The TRUETIME tool is very useful for these purposes; however, it has to be validated in comparison with a real system. This will be addressed in a more extensive work to be completed in a near future at EMBRAER and INPE, using a TTP cluster provided by TT Tech, as shown in Figure 24.

![Figure 27 – Cluster TTP](image)

BIBLIOGRAPHY

BIOGRAPHIES

Herminio Duque Lustosa is graduated in Electrical Engineering by São Paulo State University (UNESP-FEIS) in 1999. He has had graduate studies in Biomedical Engineering at the Campinas State University (UNICAMP-FEEC) in 2000. In 2001, he attended the EMBRAER Specialization Program in Aeronautical Engineering; and, lately, in 2006 he attended EMBRAER's Specialization Program in Flight Safety.

In 1994 he worked with industrial automation as a technical staff of key industrial applications, in projects like ink reactors, boilers fed by crushed sugarcane and plants at food industry. Since 2001, he has been at EMBRAER as a Product Development Engineer of Flight Control Systems.

From 2005 up to the beginning of 2008 he has led a team of R&D in Fly-by-Wire Systems, where the TTP bus was evaluated.

Nowadays he leads a team of Product Development Engineers in Fly-by-Wire Systems for the newest EMBRAER Executive Jet – Legacy 500. And he is Engineering Representative Member of Internal Flight Safety Committee.

In addition, he is a graduate student in the Space Engineering and Technology Master Degree Program at INPE (www.inpe.br), with research topic on the influence of data bus protocols on an aircraft and satellite control systems, supervised by Dr. Marcelo L. O. Souza. Contact him via:

Herminio Duque Lustosa
Flight Controls - Product Development Engineer - Team Leader
EMBRAER Executive Jets - Legacy 450/500
Empresa Brasileira de Aeronáutica-EMBRAER
Av. Brigadeiro Faria Lima, 2170
São José dos Campos, São Paulo,12227-970, Brazil
Phone: 55-12-39276614
E-mail: herminio.lustosa@embraer.com.br

Marcelo Lopes de Oliveira e Souza is an Electronics Engineer by the Instituto Tecnológico de Aeronáutica-ITA, at São José dos Campos, São Paulo, Brazil in 1976. Since then, he is a Registered Professional Engineer in Brazil by the Federal Council of Engineering, Architecture and Agronomy-CONFIA, Section of São Paulo-CREA-SP. He joined the National Institute of Space Research-INPE at São José dos Campos, São Paulo, Brazil in 1977 in the first group to learn and work with satellites there. He was at the first satellite technical mission of INPE/Brazil to learn and work at the Centre National D’Etudes Spatiales-CNES, in Toulouse, France in 1979. He has a Master in Space Sciences/Orbital Mechanics by INPE in 1980. He has a Ph.D. in Aeronautics and Astronautics by the Massachusetts Institute of Technology-MIT at Cambridge, Massachusetts, USA, in 1985. He is a Full Professor of the Options Space Mechanics and Control-CMC and Engineering and Management of Space Systems-CSE of the Space Engineering and Technology Graduate (M, D) Course-ETE; and he is a Senior Researcher of the Space Mechanics and Control Division-DMC of INPE since 1991, where he created and coordinates the Laboratory for Simulation, Identification and Modeling Environments-LABSIM2 of Attitude and Orbit Control Systems-AOCS since 2002. See http://www24.brinkster.com/labsim2. Since 1977 he has worked and supervised works in various aspects of AOCS; and later, on Modeling, Identification, Simulation and Control of Dynamic (mainly Aerospace) Systems, and their Development Environments. He is a senior member of prestigious societies like the AIAA, ISA, MIT Alumni Association, etc. in whose journals and congresses he has published many works on those and related areas. Contact him via:

Marcelo L. O. Souza
Professor and Senior Researcher
National Institute for Space Research-INPE
Division of Space Mechanics and Control-DMC
Av. dos Astronautas, 1758 Satellite Building, Room 27
São José dos Campos, São Paulo, 12227-010, Brazil
Phone: 55-12-39456191,
E-mail: marcelo@dem.inpe.br