



## **End-to-end delay analysis in an Integrated Modular Avionics architecture**

Nesrine Badache, Katia Jaffrès-Runser, Jean-Luc Scharbarg, Christian Fraboul

### **► To cite this version:**

Nesrine Badache, Katia Jaffrès-Runser, Jean-Luc Scharbarg, Christian Fraboul. End-to-end delay analysis in an Integrated Modular Avionics architecture. 18th Conference on Emerging Technologies and Factory Automation (ETFa 2013), Sep 2013, Cagliari, Italy. pp. 1-4. <hal-01212948>

**HAL Id: hal-01212948**

**<https://hal.science/hal-01212948v1>**

Submitted on 7 Oct 2015

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



HAL Authorization



## Open Archive TOULOUSE Archive Ouverte (OATAO)

OATAO is an open access repository that collects the work of Toulouse researchers and makes it freely available over the web where possible.

This is an author-deposited version published in : <http://oatao.univ-toulouse.fr/>  
Eprints ID : 12742

Official URL: <http://dx.doi.org/10.1109/ETFA.2013.6648155>

**To cite this version** : Badache, Nesrine and Jaffres-Runser, Katia and Scharbarg, Jean-Luc and Fraboul, Christian *End-to-end delay analysis in an Integrated Modular Avionics architecture*. (2013) In: 18th Conference on Emerging Technologies and Factory Automation (ETFA 2013), 10 September 2013 - 13 September 2013 (Cagliari, Italy).

Any correspondance concerning this service should be sent to the repository administrator: [staff-oatao@listes-diff.inp-toulouse.fr](mailto:staff-oatao@listes-diff.inp-toulouse.fr)

# End-to-end delay analysis in an Integrated Modular Avionics architecture

Nesrine Badache, Katia Jaffrès-Runser, Jean-Luc Scharbarg, Christian Fraboul  
University of Toulouse IRIT-INPT/ENSEEIH

2, rue Camichel, 31000 Toulouse, France

Nesrine.Badache, Katia.Jaffres-Runser, Jean-Luc Scharbarg, Christian.Fraboul@enseeiht.fr

## Abstract

*Recent modular avionics architectures have been designed to share computation and communication resources. However, such an approach creates new challenges to master the temporal properties of avionics applications.*

*In the context of IMA (Integrated Modular Avionics), it is crucial to investigate the performance gains that future integration platforms and software will propose. This paper brings to light the impact of spatial and temporal integration choices on the communication performance (e.g. message loss rate, latencies, ...). The conclusion of this investigation is that it is necessary to conduct a thorough modeling and simulation study of an IMA architecture integrating several applications during its early design stages.*

## 1. Introduction

Embedded avionics systems have evolved from a federative architecture where calculators were interconnected through dedicated mono-emitter links [2] towards a modular architecture. The Integrated Modular Avionics (IMA) has been standardized as ARINC 651 standard [4] for the definition of the hardware architecture and as ARINC 653 [5] for the corresponding software architecture. They define the APEX (APplication EXecutive interface) which ensures the spatial and temporal partitioning of the avionics functions. Thus, it is possible to design the application software independently from the target IMA physical platform.

However, such a modular architecture still necessitates a thorough configuration step at the early stages of its design, both at the physical and software levels. The first point in this article is to evaluate the impact of such configuration and integration choices on the overall performance of the system. Indeed, sharing communication and computation resources in such a modular system increases the complexity of accounting for the temporal properties of the target avionics applications (e.g. execution frequency of functions, communication delays, ...). More specifically, we show that the core of this problem is to evaluate end-to-end communication delays between remotely located and distributed avionics functions.

This article is organized as follows. Section 2 presents

the relevant assumptions for the integration of avionics applications on an IMA architecture, centered around an AFDX network. In Section 3, the impact of integration choices on end-to-end communication performance is illustrated on an example architecture. Section 4 presents the proposed modeling and simulation strategy to compare the communication performance of different architectures for different integrations and configuration choices. Concluding remarks and future works are given in Section 5.

## 2. IMA applications integration

An IMA architecture interconnects several computation modules, sensors and actuators using one or more communication networks. Computation modules communicate mainly using an AFDX network (Avionic Full Duplex Switched Ethernet) which has been standardized in ARINC 664 [6]. AFDX relies on a switched Ethernet technology and provides communication means for asynchronous computation modules. Data is transmitted using the so-called virtual links (VLs) that ensure data flow segregation. A VL defines a mono-emitter logical communication path through a sequence of switches. It is characterized by the minimum time that separates two consecutive frames transmitted by the same source, referred to as the BAG (Bandwidth Allocation Gap) and a maximum frame size SMAX.

An avionics application is composed of a set of functions or tasks, which are executed in partitions that ensure the temporal and physical segregation between applications. These partitions communicate using ports and logical APEX channels. Partitions are executed periodically on the physical modules and are statically scheduled in a cyclical frame called MAF (MAJOR time Frame). A partition hosts a set of periodic or aperiodic tasks that are scheduled based on a pre-defined policy. Two types of APEX ports exist: sampling and queuing ports. In a sampling port, a new piece of data overwrites the previously stored one, whether the old one has been consumed (i.e. read) or not. In contrast, in a queuing port, all incoming pieces of data are stored in their order of arrivals.

In the design of an IMA system, the problem is then to allocate the partitions to the computation modules (spatial allocation), to allocate the APEX channels for the various communications taking place between the tasks and to define appropriate MAFs (temporal allocation). This integration has to guaranty that end-to-end communica-

tion delays or packet losses are maintained below given limits.

This integration problem is illustrated for the application presented in Figure 1. This application is composed of 14 partitions  $P_1, P_2, P_{14}$ . Each partition  $P_i$ ,  $1 \leq i \leq 7$ , executes a periodical function  $F_i$ . At the end of its execution, each function (i.e. task)  $F_i$  creates a message  $Msg_i$  to be sent to function  $F_{i+7}$ . In this example, functions  $F_1$  to  $F_6$  have a period of  $32ms$  while  $F_7$  has a period of  $16ms$ .

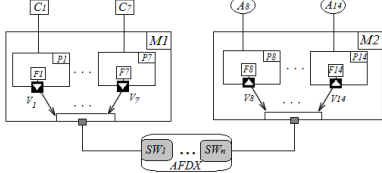


Figure 1. Illustrative IMA example.

The target physical architecture for this application is composed of computation modules interconnected via an AFDX network. A first module  $M_1$  is linked to sensors  $C_1$  to  $C_7$  whose values are read by functions  $F_1$  to  $F_7$ , respectively. These functions are hosted by partitions  $P_1$  to  $P_7$ , which are executed in module  $M_1$ . A second module  $M_2$  is linked to a set of actuators  $A_1$  to  $A_7$  which are commanded by functions  $F_8$  to  $F_{14}$ . Partitions  $P_8$  to  $P_{14}$  are executed on module  $M_2$  for functions  $F_8$  to  $F_{14}$ , respectively.

Functions  $F_i$ ,  $1 \leq i \leq 7$ , communicate with its remote functions  $F_{i+7}$  by sending at the end of each one of their activation period a message  $Msg_i$ . The resulting seven message flows are transmitted from module  $M_1$  to  $M_2$  using seven VLs,  $V_1, \dots, V_7$ , defined in the AFDX port of  $M_1$  ( $Msg_i$  is sent on  $V_i$ ). We assume that the size of a message does not exceed the maximum packet size of its VL. A reasonable design for the MAFs of  $M_1$  knowing the periodicity of  $F_1$  to  $F_7$  is to create a MAF of  $14ms$  repeating itself every  $16ms$ . Each function is executed during  $2ms$ . The MAF of  $M_1$  is thus subdivided into 7 slots of  $2ms$  each (cf. Figure 2).

The first six slots are used every two MAF cycles (i.e. every  $32ms$ ) by  $F_1$  to  $F_6$ , while slot number seven is used each cycle (i.e. every  $16ms$ ) by  $F_7$ . Accordingly, VLs  $V_1$  to  $V_6$  are assigned a BAG of  $32ms$  and  $V_7$  a BAG of  $16ms$ .

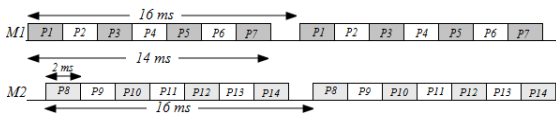


Figure 2. MAFs of modules  $M_1, M_2$

The MAF of module  $M_2$  follows the same model as  $M_1$  as shown in Figure 2.  $M_1$  and  $M_2$  modules are asynchronous, which means that MAFs can experience different offsets as the physical module is being started. Unfortunately, no simplifying assumption can be made regarding these offsets.

### 3. Impact of applications integration

This section investigates the impact of integration choices on end-to-end communication performance. To this end, we first define the end-to-end communication delay. Then, we show on the example of figure 1 how integration choices impact these delays.

In our analysis, we derive the end-to-end communication delay of the function  $F_7$  that transmits the messages  $Msg_7$  to the function  $F_{14}$  through VL  $V_7$ . The end-to-end communication latency is here defined as the duration between the time  $Msg_1$  is generated in  $F_7$  and the time it has been consumed by  $F_{14}$ . It is decomposed into three components:

–  $T_{VL}: Msg_1$  has to wait until a BAG has elapsed before being transmitted on the VL.  $T_{VL}$  is the time  $Msg_1$  is waiting for an available transmission slot in  $V_7$ .  $T_{VL}$  varies from  $0ms$  to  $V_7$  BAG size.

–  $T_{nw}$ : the latency the message experiences in the network. It is completely determined by the target network properties.

–  $T_{dest}$ : When  $Msg_1$  arrives at the incoming port of  $M_2$ , it has to wait until function  $F_{14}$  is scheduled and can consume it. Thus,  $T_{dest}$  is the duration  $Msg_1$  has to wait in  $M_2$  before being read by its destination function. These

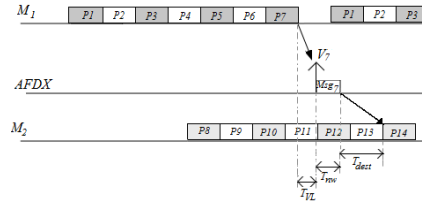


Figure 3.  $F_7$  and  $F_{14}$  communication

communication latencies are illustrated on our example in the following:

1. **Latency for an available VL slot.** In the example of figure 3, the MAF of  $M_1$  and VL  $V_7$  have an equal period ( $16ms$ ). In this case,  $T_{VL}$  is constant for each new message instance (cf.  $T_{VL}$  on figure 3). Now, what happens if the period of a function  $F_x$  does not match with a possible BAG value (BAGs values are limited to powers of two durations, starting at  $8ms$ )? Let consider a period for  $F_x$  of  $30ms$ . The corresponding partition can be allocated in a  $30ms$  MAF cycle. To ensure the availability of a BAG of the VL  $V_x$  from each message production, the BAG size has to be fixed to the largest available BAG size which is lower or equal to  $F_x$  period, i.e.  $16ms$  in this case. Thus, a message created every  $30ms$  finds an available slot for transmission in the VL of period  $16ms$ .

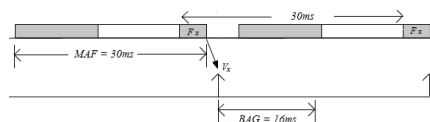


Figure 4. Availability in  $V_x$

Consequently, it is possible to control the latency for the VL availability (as long as the message sending period is not lower than the smallest BAG i.e  $1ms$ ).

2. **Latency through the network.** The latency  $T_{nw}$  consists in two parts: *i*) the multiplexing delay  $d_{mux}$  on the  $M_1$  output port and *ii*) the transmission latency  $T_p$  through the network.

*a) Multiplexing delay  $d_{mux}$ :* all the VL data frames are multiplexed in an ordered sequence according to the messages sending times in the AFDX port. Thus, a data frame of  $V_7$  arrived after a set of data frames produced earlier by other VLs will have to wait for complete transmission of previous frames. This multiplexing delay is depicted in figure 5 for three successive frames of  $V_7$  of the illustrative example. All VLs carry here a frame size of 1530 bytes. At 100 Mbps, emission duration is of  $0.122ms$ . The activation of the function  $F_1$  to  $F_6$  during the first occurrence causes a delay of  $0.732ms$  for the data frame of  $V_7$ . Considering the previously defined BAG values ( $16ms$  for  $V_7$ ,  $32ms$  for  $V_1$  to  $V_6$ ), the same behavior is repeated in the third occurrence in  $V_7$ . However, the second occurrence of  $V_7$  is not affected ( $d_{mux} = 0$ ). The difference between the minimum and the maximum  $d_{mux}$  value is denoted by *multiplexing jitter*. When the data frames generation times of the VLs are at the source module (e.i. modules and VLs offsets are known), this multiplexing jitter can be analytically computed [9], otherwise, it requires a recursive derivation through simulations.

*b) Transmission latency through the AFDX network:* it is the difference between the data frame release time from the output of the source module and the reception time in the destination module. In an AFDX network, this delay varies depending on the network traffic [10]. Different approaches to bound this variability exist [10]. Some approaches take into account the local scheduling of different VLs originating from the same source. It has also been shown that this local scheduling of VLs can significantly reduces the delays occurred in the network [11].

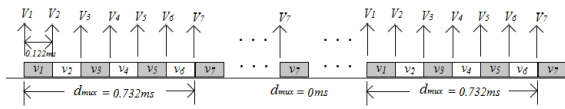


Figure 5. Multiplexing delay in  $V_7$

3. **Latency at the destination module.** The variation in the latency  $T_{nw}$  means that the successive data frames received in destination module  $M_2$  are no longer periodic. This is illustrated in figure 6.

The function  $F_{14}$  is periodic. However, the variable latency introduced by the AFDX network on  $Msg_7$  occurrences results in an additional delay  $T_{dest}$  on the reception time in the destination module. Since  $M_1$  and  $M_2$  are asynchronous, it is impossible to set execution times of  $P_{14}$  on  $M_2$  to compensate for this network jitter.

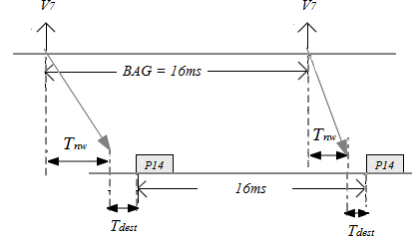


Figure 6. Latency at the destination module  $M_2$

It is therefore possible that two data frames of  $V_7$  arrive between two successive executions of  $P_{14}$ . This scenario is shown in the figure 7.

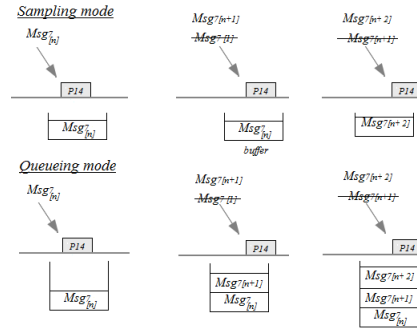
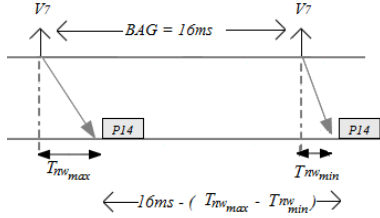


Figure 7. Reception in sampling and queueing modes

Depending on the type of destination port (queuing or sampling), such successive data receptions have a consequence on the communication performance. The message  $Msg_{7n+1}$  overwritten by the new occurrence  $Msg_{7n+2}$ . If it is a queueing port, the message  $Msg_{7n+1}$  is consumed in the next execution of  $P_{14}$  ( $n+2$ ). In this case,  $T_{dest} > 16ms$ . To avoid such loss of messages or additional delay problems, the execution period of  $P_{14}$  has to be reduced to avoid the reception of two data frames of  $V_7$  between two successive execution of  $P_{14}$ . The minimal duration between the reception of two successive data frames on  $M_2$  is determined by deducting the multiplexing jitter induced by the network ( $T_{nw} = T_{mux} + T_p$  in this case) from the execution period of  $P_{14}$ . The figure 8 illustrates the minimal required duration avoid messages loss (sampling) or additional delay (queueing).

## 4. Proposed modeling and simulation tool

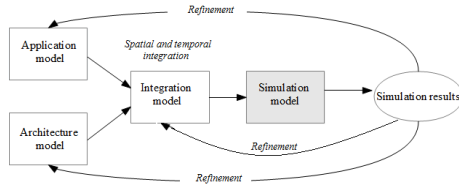
The analysis of Section 3 stresses the need for a tool capable to evaluate and compare different integration choices. Such a tool would enable the comparison and optimization of the temporal and spatial allocations needed in the design of an IMA system. For example,



**Figure 8. Minimal execution period**

it could provide optimal MAF constructions for remotely communicating applications, under given end-to-end requirements. Likewise, is needed compare the parameters of different communication architectures by changing for example the number and the characteristics of VLs (BAG, SMAX) in an AFDX network. This comparison can consider different priority levels among messages (messages loss and additional delay not allowed), or messages with less strict constraints. Overall, it is about the comparison of different IMA platform architectures (number of modules, AFDX ports, ...) that lead to different allocation spatial scenarios.

Our main objective is the modeling and simulation of an IMA platform. It consists in the description of the different characteristics (software architecture, physical architecture, integration), to generate a simulation model able to verify the application constraints and requirements. The proposed modeling approach is described in the figure 9. It consists of three modeling levels: application model, architecture model and integration model.



**Figure 9. Proposed modeling approach**

The application model describes the avionics applications in term of communicating partitions across logical communication ports and APEX channels. It also describes the embedded functions embedded onto the partitions.

The architecture model formalizes the description of an IMA platform architecture in terms of physical components (execution modules, sensors, actuators, ...) linked in heterogeneous networks.

The integration model describes the mapping of the application model onto the architecture model: spatial allocation of the partitions onto the execution modules and the APEX channels on the AFDX ports, and the temporal allocation of partitions and modules (MAF) and the virtual communication links (BAGs, SMAX).

The auto-generated simulation model is obtained from the integration model which is capable of driving the performance of the loaded IMA architecture. with respect to different communication metrics (end-to-end communication delay, jitters, messages loss, ...)

The generated simulation model can be used to evaluate the performance of application and architecture models. Moreover, we can use these results to fine tune the parameters defined in application and architecture model and re-generate the simulation model.

This iterative process can help in early design phase to decide optimal parameters for the IMA architecture. We are developing such a tool and initial results are encouraging.

## 5. Conclusion

In this paper we have presented the complexity of communication delays, analysis in the context of IMA architecture (impact of the multiplexing jitter and the communication in sampling and queueing modes, influences of source and destination execution periods and influence of the choice of the logical communication channels).

A deep analysis of those influential parameters requires a simulation tool able to consider the random aspects (offsets of the modules and networks, tasks execution duration, communication delay, ...).

The ongoing research covers the modeling approach of such an IMA architecture according to three levels (application, architecture, integration) and aims at generating a simulation tool that can be used to calculate the end-to-end communication of modeled software and physical architecture.

## References

- [1] M. Lauer and al. : *Latency and freshness analysis on IMA systems*. ETFA 2011, Toulouse, France, Sep. 2011.
- [2] Aeronautical Radio Inc. ARINC 429, *ARINC specification 429-ALL: Mark 33 Digital Information Transfer System*, 2001.
- [3] F. Martin and C. Fraboul : *Modeling and Simulation of Integrated Modular Avionics* PDP 1998, Madrid, Spain, January 1998.
- [4] Aeronautical Radio Inc. ARINC 651, *Design Guidance for Integrated Modular Avionics*, 1991.
- [5] Aeronautical Radio Inc. ARINC 653, *Avionics Application Software Standard Interface*, 1997.
- [6] Aeronautical Radio Inc. ARINC 664, *Aircraft Data Network, Part 7: Avionics Full Duplex Switched Ethernet (AFDX) Network*, 2005.
- [7] M. Lauer and al. : *Analyzing end-to-end functional delays on an IMA platform*. In Proc. of the 4th ISO/LA'10, Heraklion, Greece, page 243-257. Springer-Verlag, 2010.
- [8] J. Javier Gutiérrez and al. : *Response time analysis in AFDX networks with sub-virtual links and prioritized switches*. JTR'12, Santander, Spain, January 2012.
- [9] A. Choquet-Geniet and E. Grolleau : *Minimal schedulability interval for real time systems of periodic tasks with offsets*. Theoretical of Computer Sciences, vol. 310, pp. 117-134, 2004.
- [10] H. Bauer and al. : *Worst-case delay analysis of an industrial AFDX network* DATE 2011, Dresden, Germany, Mars 2013.
- [11] X. Li and al. : *Existing offset assignments are near optimal for an industrial AFDX network* RTN 2011, Porto, Portugal, July 2011.