Integrating end-system frame scheduling for more accurate AFDX timing analysis

Authors :
Marc Boyer, Luca Santinelli – ONERA, The French Aerospace Lab – F31055 Toulouse
Nicolas Navet – Université of Luxembourg – L1359 Luxembourg
Jörn Migge – RealTime-at-Work – F54600 Villers-lès-Nancy
Marc Fumey – Thales Avionics – F31100 Toulouse

Keywords : Timing verification, AFDX, Network Calculus, worst-case traversal times, offsets.

Abstract: Avionics systems distributed on AFDX networks are subject to stringent real-time constraints that require guaranteeing the Worst-Case Traversal Time (WCTT) on the network for each of the data flows. Over the last 10 years, since the initial use of Network Calculus in certification, important progresses have been made in AFDX timing verification. The maximum pessimism for the latencies is now known to range from 10 to 25% on realistic systems. Further progresses towards more accurate timing analysis can still be made by considering additional temporal information. In this paper, we show that integrating the knowledge of the scheduling of the frames that is done within an end-system in the timing analysis enables to dramatically reduce the WCTT bounds computed by Network Calculus. Indeed, in our experiments performed on a realistic configuration provided by Thales Avionics, this technique reduces the WCTT upper bound by 40% on average over all flows. The reason is that the scheduling of the frames shapes the outgoing traffic, reducing thus peaks of load on the outgoing traffic, which can be accounted for in the timing analysis. Importantly, because the scheduling of the frames within the end-systems is in the scope of the network supplier, unlike the scheduling of tasks done at the application level, the approach presented here does not imply major changes in the design process.

1 Introduction

1.1 Context of the study

Avionics Full DupleX (AFDX - see[8]) is an aeronautic-specific switched Ethernet technology that supports data exchanges among avionics sub-systems with bounded latencies, without requiring the sub-systems to share a common global clock like in TDMA networks. In AFDX, virtual links (VLs) define the unicast and multicast connections there exist between end-systems. The predictable latencies of AFDX can be achieved because the standard enforces appropriate switching mechanisms within the communication switches and because the workload submitted to the network by each sub-system is upper bounded and known in advance. Indeed, to each VL is associated a maximum frame size and a minimum time between the transmission of two successive frames of the same VL. This latter quantity is called the Bandwidth Allocation Gap (BAG) and has to take a value that is a power of two in the range 1 to 128ms.

With the increasing amount of critical data exchanged with real-time constraints in on-board aerospace systems, the computation of accurate upper bounds on network traversal times is an industrial requirement. Indeed, it is needed in the certification process to convince the certification authorities that the real-time and safety constraints are met and this should be achieved without over-provisioning the hardware resources.

If, for realistic AFDX networks, it is in practice not possible to compute the exact Worst-Case Traversal Time (WCTT), conservative upper bounds on the WCTT can be computed in reasonable time (a few seconds using state-of-the-art Network Calculus). The accuracy of these bounds is crucial since over-approximation leads to over-provisioning of hardware resources and is a hindrance to system evolutions. Over the last 10 years, since the initial use of Network Calculus in certification, important progresses have been made in timing verification and the maximum pessimism for the latencies is now known to range from 10 to 25% on realistic systems (see experiments in [1] for timing verification with Network Calculus and [2] with the trajectory approach).

1.2 Contribution and related works

Further progresses towards more accurate timing behavior analysis can still be made by considering additional temporal information. In particular, as done in [6,7], taking into account the scheduling of the tasks provides more accurate bounds on the quantity of work that can possibly be submitted to the network, and thus enable to more precisely evaluate the maximum interference that can be suffered by a flow. But in aircraft design, linking network analysis to task scheduling will create strong dependencies and complexity to the global design.
The approach presented in this paper is similar in the spirit with the work in [3,4] where AFDX virtual links are desynchronized with offsets so as to better balance network load over time, in a similar manner to what has been done in automotive CAN networks for almost the last 10 years [5]. The approach discussed here do not required a number of assumptions placed in [3,4] that make those proposals non-practical without changes in the design flow. Importantly, the knowledge of the scheduling of tasks is not needed, and the technique discussed is usable with non-periodic streams as well as with streams with a deadline smaller than their period. The rationale and limitations of existing approaches are further discussed in Section 3.
In this study, we show experimentally that integrating the knowledge of the scheduling of the frames that is done within an end-system in the timing analysis enables to importantly reduce the WCTT bounds computed by Network Calculus, up to 40% on average. Importantly, because the scheduling of the frames in an end-system is in the scope of the network supplier, unlike the scheduling of tasks done at the application level, this does not imply major changes in the design process.

2 Decomposition of the end-to-end latency from the sending to the receiving application

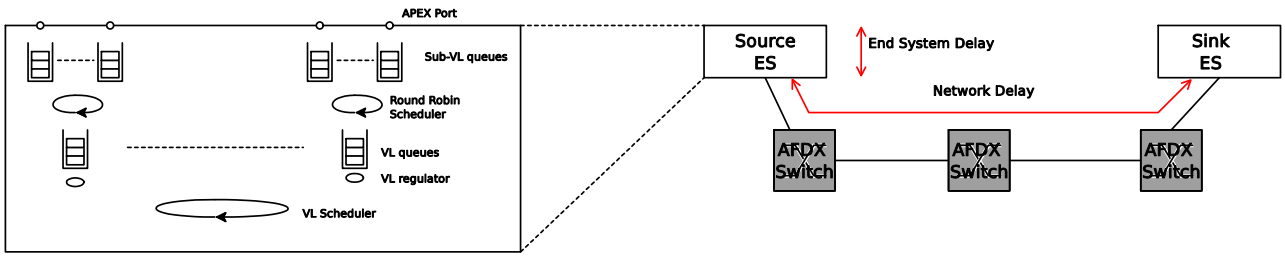

Figure 1: Communication delays: end-system latency and network traversal latency.

The ARINC 664P7 specification defines 3 delays [§ 2.1, §3.2.4.1, 8]: the transmission latency, at source and receiver ends, the network latency, and the sum of the latencies in the switches that are crossed. The latency at the receiver end is mainly a technological latency, and is not further discussed here.

2.1 Network latency

The network latency is the delay inside the network, i.e. the sum of the delays inside the network switches, divided into packetisation at input port (the frame size divided by the link speed), a filtering delay (to reject out-of-contract frames), a switch commutation latency (sum of both must less than 100µs, [§4.11.13, 8]), and waiting time in the two-level priority output ports that is due to link sharing among flows.

This network latency can typically be upper bounded using the network calculus theory [9] which is a technique used in certification. The evaluation of the network bounds, with the latest advances in network calculus, have been shown to be moderately pessimistic (typically less than 25%) on experiments made on realistic large AFDX networks [1].

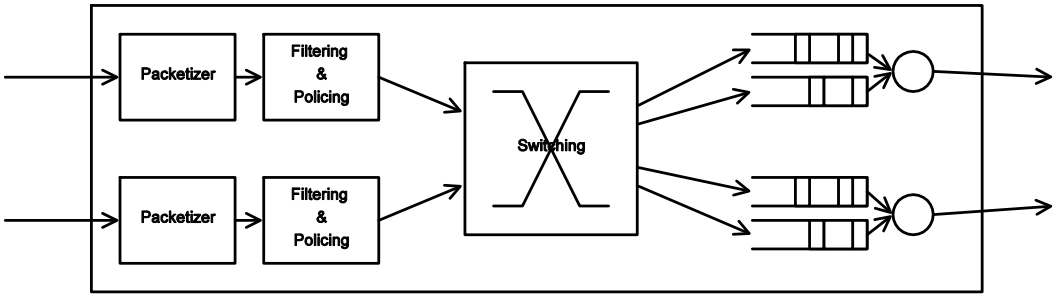

Figure 2: Generic AFDX switch architecture

2.2 Source end-system latency

The latency at source starts when an application writes a data in an APEX port. After some UDP/IP handling, an AFDX frame is placed into a sub-VL queue. Sub-VL queues are then multiplexed into the VL queue using a Round-Robin scheduler, knowing that sub-VLs are optional (an APEX port can be directly linked to a VL queue). A traffic regulator is associated to each VL queue. It ensures that there is at least duration of one BAG between any two successive frames. Lastly, a VL scheduler is used to share the network bandwidth between the VLs. This scheduler must also ensure that output flows still respect the per-VL BAG, up to some jitter *MAX_Jitter* that must be less than 500µs (see §3.2.3 and §3.2.4.3 in [8]). That is to say, the time interval between the first bits of two frames of the same VL, with BAG *B*, must be not less that *B* – *MAX_jitter*.

To ensure this constraint, the per-VL scheduler cannot implement a simple FIFO or Round Robin policy. As an illustration, consider the end-system shown in Figure 2 with 3 VLs: two frames m_2 and m_3 are placed into empty VL queues while frame m1 is being sent. Assume that m_2 is sent before m_3 (as a result of applying the FIFO or Round Robin policy). Then, m_3 is sent after m_1 and m_2. Let us call t_3 this instant. Then, assume that VL3 queues a new frame m'_3, with strict BAG respect. Assume also that when m'_3 is sent, all other VL queues are empty. Then, if m'_3 is sent immediately, the interval between m_3 and m'_3 will be less than the BAG. Let t'_3 be the time when m'_3 is sent. The value t'_3-t_3-BAG_3 is called the jitter. With only three VL, this jitter is always less than 500µs, but with 5 VL of maximal frame size, at 100Mb/s, the condition no more holds.

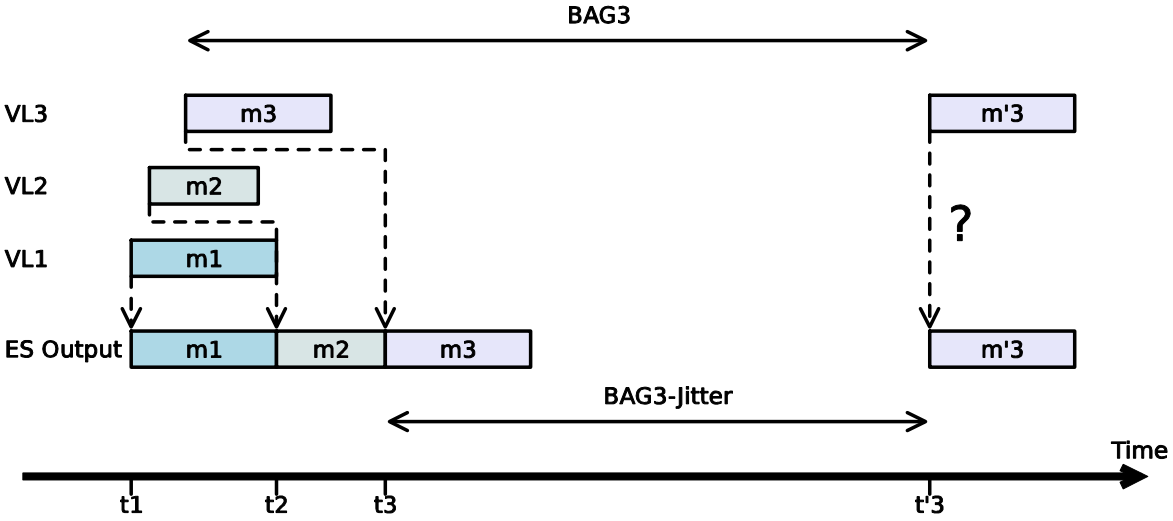

Figure 3: Jitter at end-System output.

The specification requires also that "if the system has no other data to proceed on this virtual link", the latency must at the source end-system respects

$$MAX\_latency_i \leq BAG_i + MAX\_Jitter + Technological\_latency$$

It is up to the end-system provider to implement a scheduler that respects these two requirements, on jitter and latency.

3 Modeling the sending end-system: a state-of-the-art

3.1 From task scheduling to data production windows

The AFDX network is a shared resource: several virtual links cross it. While no information is given, one must assume that is it possible for all VLs to send their frames simultaneously. But the data to be transmitted in the VLs are produced by tasks, and on mono-processor systems, only one task is active at a given time. In [6,7], a system with periodic tasks (with variable execution times) is considered, where tasks send frames only at end of execution (assuming a typical execution model: data acquisition/control computation/actuation), and assuming a static priority scheduler. Under these assumptions, the traffic generated by the tasks (known as "arrival curves" in network calculus theory) is computed, considering all possible executions. To do so, common scheduling analysis must be adapted to consider tasks instances. These papers show, on hundreds of generated configurations, that the burst size at system output decreases linearly with the number of tasks. Nevertheless, the computation is not propagated through the network, and no evaluation of the impact of the approach on the network delay is given.
This study in [6,7] was a theoretical one, and cannot be directly applied to aircraft design: to facilitate the re-use of components and ease system evolutions during the development cycle, it is may not be possible to make assumptions about the scheduling of tasks while designing the network. Nevertheless, the method in [6,7] is the basis for this study.

3.2 Scheduling virtual links with offsets

Another way to reduce the interferences between the flows in the sending end-system and the switches is to desynchronize the flows in such a way as to better uniformize the load over time. This approach, called scheduling with offsets is routinely used today in the design of vehicles [5]. Its use in the context of AFDX is explored in [3,4]. The experiments in these studies show that offsets are very efficient to reduce the latencies of the packets in the network. In [3], it is assumed that the VL are send in a purely periodic way by the end systems, and that per-end systems offsets are assigned to VL (*i.e.* local synchronization on an end-system between flows, but no synchronizations between the end-systems). Then a method is given to take into account the offsets between flows, to propagate them throughout the network, and to compute a better bound on network delays. Different strategies to assign offsets are compared in [4], and evaluated on a realistic case study. It is shown that offsets can reduce the network delay up to 51% (average of the VLs). Another contribution of the paper is an upper bound on the possible gain: whatever the offsets are, without synchronization between end-systems, each frame can compete in each queue against one frame of each other end-systems. Using this upper bound, it is shown on the case study of the paper that the *maximal possible gain* would be of 53%.

However there is a downside if data production and data transmission are not explicitly synchronized: before being transmitted, a packet may have to wait for a time that can be up to one transmission period (i.e., one BAG) in the end-systems. It means that this method decreases the network delay but increases the end-system delay. Since it is commonly assumed that the network delay in AFDX is in general less than the BAG, it means that, without synchronization of task and network scheduling, this approach may actually increase the global communication latency, by increasing the "transmission latency" more than the reduction of the "network latency".

3.3 The End-System VL scheduler is a scheduler

As presented in Section 2, there is a VL scheduler in each end-system. The implementation of this scheduler is up to the system provider. In this study, the behavior of the Thales end-system has been modeled. For confidentiality reasons, the exact behavior of this scheduler will not be presented here. It just have to be noticed that the same kind of method that the one of [6,7] is being used. Nevertheless, it can be mentioned that this scheduler has been configured so as to ensure the respect of the 500µs maximum jitter. It can also be mentioned that this scheduler offers some way to "prioritize" the flows into the end-system, independently of the priority of the VL inside the network. A VL with a local high priority is ensured to have a end-system latency less than 1ms. Of course, if too many VLs are placed into this local high priority class, the 1ms constraint cannot be satisfied anymore.

4 Experimental assessment

4.1 Case-study and toolset

The experiments are performed on a realistic large AFDX configuration provided by Thales Avionics, whose main characteristics are summarized in the table below.

Entities	Number
End Systems	104
Routers	8
Virtual Links	974
Latency constraints	6501

As can be seen in the following table, each Virtual Link (VL) has on average 6 destination end systems. This explains the 6501 latency constraints shown in the first table, which means also that 6501 WCTT bounds need to be computed.

	Virtual Link destinations	BAG (minimum interarrival time)	Maximal Packet Size	Traversed Routers	Latency Constraints
minimum	1.0	2 ms	100 bytes	1	1000 µs
average	6.6	60 ms	380 bytes	1.3	10040 µs
maximum	84.0	128 ms	1500 bytes	4	30000 µs

The computations are performed with the RTaW-Pegase tool [10] that implements AFDX and switched Ethernet timing analysis with network calculus. Unlike most other timing analysis tools, RTaW-PEGASE is no black-box in the sense that the algorithms are documented and have been proven correct in peer-reviewed publications (see [11,13,14] and, more recently, with the help of formal methods in [12] to assert the correctness of the computation). When information about the scheduling of frames in the sending end-system is available, RTaW-Pegase makes use of a more fine-grained model that accounts for the resulting desynchronization between flows, and thus leads to more accurate latency bounds.

4.2 Results with several End-System configurations

The gain is computed on a per-VL basis: for each VL, the bound on the latency is computed first without considering the end-system scheduling using the state-of-the-art of network calculus as most often used in the industry now, that is with piecewise linear functions [9]. The corresponding bound is denoted by a_i for VL i. Then, for a given configuration of the VL scheduler on the sending end-system, the new delay bound b_i is computed, using the generic ultimately pseudo-periodic functions [11]. The per-VL gain is $g_i=(a_i-b_i)/a_i$. It should be noted that the gains g_i combine the benefits of two recent advances in network calculus: better computing functions and better end-system modeling[1]. Finally, the average of all gains is computed.

The first experiment was done to compare the current approach with the previous works of [3,4]: a purely periodic scheduler was assumed, with the same offset assignment algorithms. It leads to an average gain of 42%. This must be compared with the 51% of [3,4], while we do not know if this is because the case study is different, or because of the analysis method itself.

In the second experiment, all VLs are placed in the local high priority class. This class tries to minimize the waiting time into the end-system, while still maintaining the 500µs jitter constraint. Of course, if the end-system is too loaded, the 1ms of local delay cannot be satisfied. In this case, the average gain is equal to 26% only.

Finally, only the VLs with a BAG no greater than 8ms are placed into this local high priority class. The rational is that VLs with smaller BAGs are more often used by avionics systems with stringent delay requirements. In this case, the gain is 38%, which is almost as good as for the purely periodic configuration.

[1] The choice to combine the gain of two improvements in the same value is done to show their impact in an industrial context. The reader can refer to [1] for an evaluation of the impact of the use of Ultimately Pseudo-Periodic (UPP) functions alone, without modeling the frame scheduling at the source end-system.

End-systems configuration	Average gain
Purely periodic	42%
All VLs in local high-priority class	26%
VLs with BAG ≤ 8 in local high-priority class	38%

The average gain is presented in the above table, but the table does not inform about how this gain is distributed among the VLs. This question is answered by Figure 3 by plotting the bounds with and without scheduling for all VLs. For readability, VLs are sorted by increasing value of latency with frame scheduling taken into account.

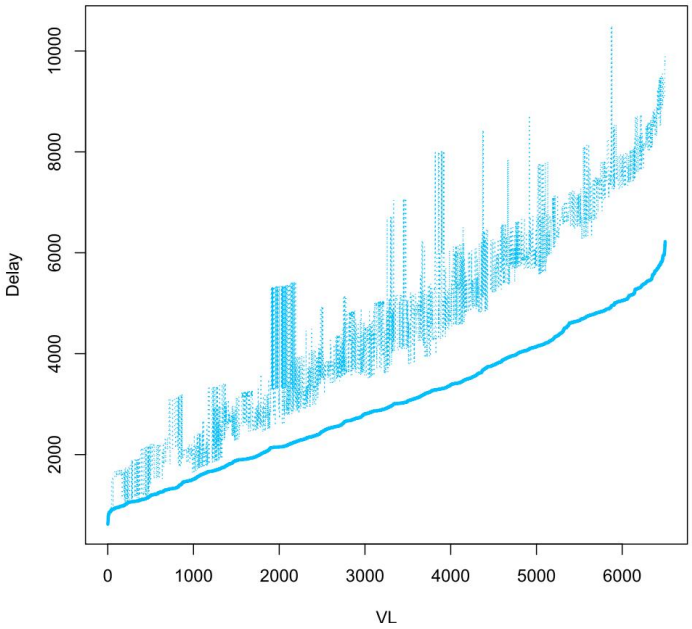

Figure 3: Upper bounds on the worst-case traversal times (WCTT in us) with (lower curve) and without the knowledge of the frame scheduling within the sending end-system, placing VLs with BAG not greater than 8ms in local high priority class. Virtual Links are sorted by increasing latencies. The VLs on the graph are sorted by increasing latencies computed with frame scheduling knowledge, which explains why the lower curve is monotonous unlike the upper curve.

5 Conclusion and future work

The network calculus theory is used to provide upper bounds the network latencies since the A380 [9]. However, measures on real networks were dramatically smaller than the theoretical bounds. It was first supposed that the theory was overly pessimistic. Since the work in [2] on the estimation of lower bounds on the latencies, it was experimentally shown in [1] that the pessimism of the method was less than 25%. This means that, without taking into account new information on the real system in the model, no large improvement of the bounds can be achieved.

It has been shown in previous theoretical studies [3,4,6,7] that considering the offsets between the flows in an AFDX network can dramatically reduce the evaluation of the network traversal times. Because these studies assume a strong link between the task scheduling and the network scheduling, they would impose an unaffordable overhead in aircraft design. This study however does not consider the applicative scheduling, but only the frame scheduling done in Thales end-systems. The system that is analyzed remains in the network perimeter and can be managed by the network designer.

In our experiments performed on a realistic case-study, integrating the frame scheduling done at the sending end in the timing analysis allow to reduce the upper bound on the network delay by an average of 38%. This improvement opens the door to a more efficient use of the network bandwidth and may ease the incrementally of existing systems.

6 References

[1] M. Boyer, N. Navet, M. Fumey, "Experimental assessment of timing verification techniques for AFDX", Embedded Real-Time Software and Systems (ERTS 2012), Toulouse, France, February 1-3, 2012

[2] H. Bauer, "Improving the worst-case delay analysis of an AFDX network using an optimized trajectory approach", IEEE TII, vol. 6 n° 4, pp521-533, 2010.

[3] X. Li, J.-L. Scharbarg, C. Fraboul, "Improving end-to-end delay upper bounds on an AFDX network by integrating offsets in worst-case analysis", IEEE ETFA 2010, Bilbao, Spain, Sep. 2010.

[4] X. Li, J.-L. Scharbarg, F. Ridouard, and C. Fraboul, "Existing offset assignments are near optimal for an industrial AFDX network", ACM SIGBED Review, vol. 8 n° 4, pp49-54, December 2011.

[5] M. Grenier, L. Havet, N. Navet, "Pushing the limits of CAN – Scheduling frames with offsets provides a major performance boost", Proc. of the 4th European Congress Embedded Real Time Software (ERTS 2008), Toulouse, France, January 29 – February 1, 2008.

[6] M. Boyer, D. Doose, "Collaboration entre méthode d'ordonnancement et calcul réseau", Actes des journées du GDR Génie de la Programmation et du Logiciel (GPL), 10-12 march 2010, Pau, France

[7] M. Boyer, D. Doose, "Combining network calculus and scheduling theory to improve delay bounds". RTNS 2012: 51-60, November 2012.

[8] "Aircraft data network part 7: Avionics Full DupleX switched Ethernet (AFDX) network", ARINC specification 664P7, June 2005.

[9] J. Grieu, "Analyse et évaluation de techniques de commutation Ethernet pour l'interconnexion des systèmes avioniques". PhD Thesis, Institut National Polytechnique de Toulouse (INPT), 2004.

[10] M. Boyer, J. Migge, and M. Fumey, "PEGASE – a robust and efficient tool for worst-case network traversal time evaluation on AFDX", SAE Aerotech 2011, Toulouse, France, 2011.

[11] A. Bouillard, E. Thierry, "An Algorithmic Toolbox for Network Calculus, in Journal of Discrete Event Dynamic Systems", Vol. 18(1), pages 3-49, 2008.

[12] E. Mabille, M. Boyer, L. Fejoz, and S. Merz, "Certifying Network Calculus in a Proof Assistant", 5th European Conference for Aeronautics and Space Sciences (EUCASS), Munich, Germany, 2013.

[13] M. Boyer, G. Stea, W. Mangoua Sofack, "Deficit Round Robin with network calculus". 6th International ICST Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS'12), 2012.

[14] M. Boyer, J. Migge, N. Navet, "A simple and efficient class of functions to model arrival curve of packetised flows", First International Workshop on Worst-case Traversal Time (WCTT), in conjunction with the 32nd IEEE Real-time Systems Symposium (RTSS), Vienna, Austria, 2011.