Tighter buffer dimensioning in AFDX networks

Nassima Benammar, Henri Bauer, Frédéric Ridouard, Pascal Richard
LIAS - ISAE/ENSMA and Université de Poitiers, France
{nassima.benammar,henri.bauer,frederic.ridouard, pascal.richard}@ensma.fr

ABSTRACT

The AFDX (Avionics Full DupleX Switched Ethernet) is the backbone network of most recent avionic communication systems. These systems require deterministic buffer dimensioning for certification reasons. As in such systems, static design is often preferred over dynamic allocation, a dimensioning in terms of frames rather than in terms of bits can be required.

A common approach consists in deriving the worst-case buffer size in terms of frames from the worst-case in bits. However, we show that this can lead to an overestimation of the buffer size. Thus, we propose a dedicated approach for a tighter dimensioning. Eventually, the two approaches are compared on a sample AFDX configuration.

Keywords

AFDX, buffer dimensioning, worst-case analysis.

1. INTRODUCTION

During the last decades, avionics systems have faced a prominent growth of the number of embedded functions and of data sharing between functions. Federal and centralized architectures have been superseded by Integrated Modular Avionics (IMA), sharing resources between functions for higher flexibility and scalability. The Avionics Full DupleX Switched Ethernet (AFDX) has been promoted [1] by major civilian manufacturers as the backbone network of such architectures.

As AFDX is intended for real-time systems, determinism is an important issue. Guaranteed worst-case transmission times (WCTT) and no frame loss are mandatory for certification in the avionics context. In this work, we will focus especially on the latter. In the nominal case, frame losses can only occur in the multiplexing points (the switch output port buffers) in a switched Ethernet network with full-duplex links. As there is no global time shared in the network, all the components are asynchronous, and the network traffic can be highly variable. Frame loss happen if buffer size cannot cope with the traffic bursts resulting from this variability. For certification purpose, AFDX buffers have thus to be carefully dimensioned.

There are however two ways of considering the worst-case occupancy for a network buffer: in terms of total amount of bits, or in terms of number of frames. The first one is suitable as long as dynamic memory allocation is not an issue: variable memory slots are allocated within memory space dimensioned in terms of bits. Nevertheless, in avionics systems, static design and fixed-size memory slots are often preferred in order to reduce the complexity of the certification process. Then, the relevant value for dimensioning a buffer is the worst case occupancy in terms of number of frames.

There is, obviously, a simple way of deriving the maximum number of frames, knowing the maximum in terms of bits, by dividing by the size of a minimal frame size. However, this can lead to an overestimation of the required memory space. Thus, we propose a more precise computation of a worst-case upper-bound in terms of number of frames, based on the knowledge of the network topology, the traffic contracts and the worst-case transmission times of the flows. Therefore, our method can be based on any WCTT computation framework available in the context of AFDX, such as the Network Calculus (NC) [5, 6] or the Forward end-to-end delay Analysis (FA) [11] (see Section 3 for more exhaustive bibliographical study). In this paper, we illustrate our method based on the bounds computed with FA.

A traffic contract for a flow v_i should define a shaping envelope at the network ingress node with a period T_i (which can be defined in a sporadic sense as a minimum inter-generation time) and a maximum transmission time C_i (a maximum frame size at the network servicing rate). Such contracts are usually available in the avionics context, since they are the foundation of WCTT analysis.

This work is organized as follows. In Section 2, we discuss the relevance of a specific per frame approach for buffer dimensioning in the context of AFDX. The main contribution of this paper is given in Section 3: the computation of the maximum number of frames in a switch buffer. The principle of the computation is detailed on a sample AFDX configuration in Section 4. We also compare our proposal to the more naive approach (deriving the bound in terms of frames from the bound in terms of bits).

2. MOTIVATION AND RELATED WORK

In the literature, only few approaches are proposed for dimensioning buffers, especially, in the context of AFDX networks (because of the certification constraint). Those methods generally focus on WCTT computation, computing the worst-case buffer occupancy in terms of bits (often referred to as the worst-case backlog) because it generates the longest waiting times in the switches.

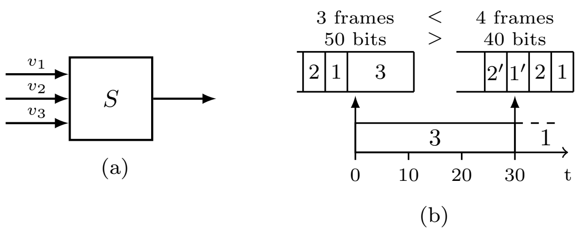

Figure 1: Example configuration where the maximum backlog in terms of bits and of frames do not coincide.

Network Calculus [5, 6, 3] has been the first method used in AFDX networks [9]. Cumulative arrival and service curves are used to compute the maximum backlog and the maximum waiting time in a servicing node (e.g., a switch output port buffer). The backlog and the delay are obtained respectively with the maximum vertical and horizontal distance between the two curves. This approach can suffer from some pessimism when using conservative envelopes for the arrival and service curves.

Another ETE (end-to-end) delay analysis approach called the Trajectory Approach (TA) [12, 13] can be used to derive a worst-case backlog in a node in AFDX networks [2]. A method to compute the number of frames per node in AFDX context with multiple priority queuing using TA approach is proposed in [4]. However, the frames are supposed to all have the same size.

In the FA approach [11, 10], a worst-case ETE delay computation method based on scheduling theory results, the worst-case buffer occupancy in terms of bits appears explicitly as a term of the computation, called *Backlog*.

In concrete terms, these methods focus on a worst-case scenario in terms of traversal time, where the backlog is maximal in terms of bits. However, it can be easily shown on a counter-example that the worst-case in terms of number of frames does not necessarily occur when the maximum is reached in bits.

Indeed, consider the configuration depicted in Fig. 1.(a). It is composed of a single node S and three flows (each coming from a different input link). v_1, v_2 and v_3 are periodic flows with frame transmission times such as $C_1 = C_2 = 10$ and $C_3 = 30$, and periods such as $T_1 = T_2 = 30$ and $T_3 = 100$. For the sake of simplicity, we consider a servicing rate of one bit per time unit (the backlog in time is equivalent to the backlog in bits). The maximum backlog in terms of bits is then obtained at time $t = 0$ and is equal to 50 bits, corresponding to the transmission time of three frames ($C_1 + C_2 + C_3$).

However, at time $t = 30$, frame 3 has left the buffer and, in the worst-case scenario, two new frames $1'$ and $2'$ (from v_1 and v_2) enter the buffer (see Fig. 1.b). Although the backlog at time $t = 30$ reaches only 40 bits, it consists in one more frame (two from v_1 and two from v_2) than the 50 bits backlog from time 0. This demonstrates that the worst-case backlog in terms of bits and of frames are two separate problems that cannot be tracked simultaneously. The worst-case backlog in frames derived from the value in bits cannot be a tight value.

Therefore, we propose to study specifically the worst-case backlog in terms of frames. This constitutes the main contribution of the paper. The principle of this approach is described in the next section.

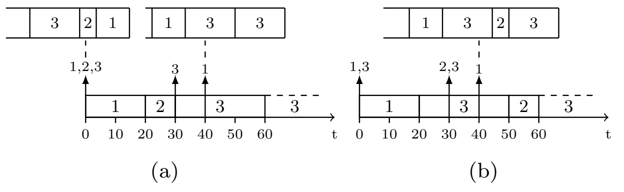

Figure 2: Arrival scenarios considering FIFO buffer.

3. CONTRIBUTION

Hereafter, we propose a new way of computing the worst case backlog of a FIFO buffer in terms of frames, denoted Nbf^h for a given network node h. The main idea of our proposal is to determine a maximum buffer occupancy by maximizing the input rate and by minimizing the output rate in terms of frames in the buffer. In a first time, we focus on the problem of getting such an arrival scenario with a FIFO buffer policy.

Let us consider again the configuration from Fig. 1.(a) with a single node S and three incoming flows but with different flow characteristics (C_i and T_i). In Fig. 2.(a), we give an arrival scenario with frames arriving at their earliest from time 0. The tiebreaker for simultaneous arrival is the lowest frame index. Due to their history on previous nodes, two consecutive frames of a flow can sometimes arrive closer than their initial period. This is why frames from v_1 and from v_3 are so close in the example of Fig. 1.(a). The maximum backlog in terms of flows is observed in S at time 0 and 40 with $NbF^h = 3$.

However, it is possible to find an even worse scenario if the arrival of frame 2 is delayed at time 30. In Fig. 2.(b), we then get a four frames backlog at time 40. To the best of our knowledge, there is no existing work about the description of a specific arrival scenario of packets leading to maximize the number of frames contained in the buffer. This is why we propose to treat separately the worst-case arrival rate of incoming frames in Section 3.1 and the strategy to minimize the servicing rate of outgoing frames in Section 3.2. The execution of the complete computation is recapped in Section 3.3.

3.1 Incoming frames

The incoming workload arriving in a node h during an interval of length t is obtained by considering the *Request Bound Function* (RBF) of each flow $v_i \in h$. For a non-preemptive sporadic flow v_i, the maximum amount of data generated during a closed time interval $[t_0, t_1]$ (with $t \geq 0$) is: $\left(1 + \left\lfloor \frac{t_1 - t_0}{T_i} \right\rfloor\right)$. However, if $[t_0, t_1]$ is the time interval to consider in h, the corresponding interval in the source node of each flow v_i expands to: $[t_0 - Smax_i^h, t_1 - Smin_i^h]$, where $Smax_i^h$ and $Smin_i^h$ are respectively the longest and the shortest times needed for a frame from v_i to reach h from its source node (*i.e.* the transmission jitter J_i^h). If we define $J_i^h = Smax_i^h - Smin_i^h$, we thus have for an interval $[0, t]$:

$$rbf_i^h(t) = \left(1 + \left\lfloor \frac{t + J_i^h}{T_i} \right\rfloor\right) C_i \quad (1)$$

The RBF function $rbf_i^h(t)$ maximizes the arrival rate of

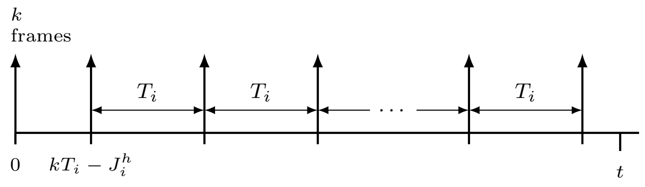

Figure 3: Minimum arrival times of frames from v_j in a FIFO buffer h, considering the function $rbf_i^h(t)$.

frames from a flow v_i in node h, so we can determine the earliest arrival time of any frame from v_i, starting from time 0:

$$rbf_i^h(0) = \underbrace{\left(1 + \left\lfloor \frac{J_i^h}{T_i} \right\rfloor\right)}_{k \text{ frames}} C_i$$

$k \in \mathbb{N}^*$ being defined such as : $(k-1)T_i \leq J_i^h < kT_i$ gives the number of frames available at time 0. The next incoming frame from v_i will then arrive at time $kT_i - J_i^h$, all the following frames arriving with an interval of T_i, as depicted in Fig. 3.

This incoming scenario, obtained from the RBF, maximizes the number of arriving frames for every flow crossing node h, as soon as possible from time 0. To determine NbF^h, we still have to minimize the number of served frames by h in the same time interval.

3.2 Outgoing frames

Due to the complexity of determining the worst case strategy with a FIFO policy, and in order to minimize the frame servicing rate, we propose to use the Largest Processing Time First (LPT) [8] as a worst-case of any non-preemptive work-conserving service policy (including FIFO). With LPT, at each decision time, the frame with the largest transmission time among all pending frames is served. We prove that LPT is optimal in order to minimize the number of transmitted frames in Theorem 1.

Theorem 1. *Given a set of independent flows generating frames with arbitrary arrival times on a node h, the servicing algorithm LPT is optimal among non-preemptive work-conserving policies to minimize the number of transmitted frames by h.*

Proof: This theorem can be proved with an interchange argument similar to the one used by Dertouzos [7] about the optimality of EDF for finding a feasible schedule in uniprocessor systems.

Consider that a node h serves frames using a servicing policy A different from LPT. At any time t, we denote by $\sigma(t)$ (*resp.* $LPT(t)$) the frame transmitted by A (*resp.* LPT). Moreover, $t_{LPT(t)}$ designs the time chosen by A to start the transmission of frame $LPT(t)$. The transmission time of frames are independent from the servicing policy.

Since A is different from LPT, there exists some time instant where the decision taken by A and LPT are different. We denote by t_1 ($t_1 \geq 0$) the first time where $\sigma(t_1) \neq LPT(t_1)$. We prove that interchanging the transmission of $LPT(t_1)$ and $\sigma(t_1)$ cannot increase the number of transmitted frames.

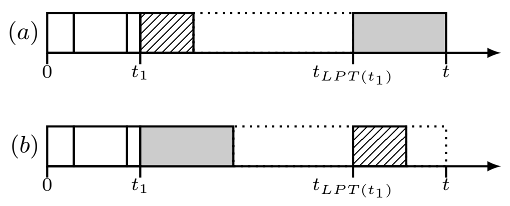

Figure 4: Optimality of LPT to minimize the number of transmitted frames

Fig. 4 illustrates the interchanging between the transmission of $LPT(t_1)$ and $\sigma(t_1)$. In Fig. 4.(a), the algorithm A serves frame $\sigma(t_1)$ (dashed frame) at time t_1 and $LPT(t_1)$ (grey frame) at time $t_{LPT(t_1)}$ whereas the transmission of these frames are interchanged in Fig. 4.(b).

During the interval $[0, t]$ (where t designs the end of transmission of frame $LPT(t_1)$ by A), in the two cases, the same duration of time (dotted area) is available to transmit others frames. Moreover, the dotted area is sliced in two in Fig. 4.(b) which can lead to transmit less frames than with A.

By iteration, at each time t_1 ($t_1 \geq 0$) such as $\sigma(t_1) \neq LPT(t_1)$, we interchange the transmission of these two frames in order to be compliant with the servicing policy LPT. At the end of the transmission of each interchanged frame, the number of transmitted frames using A is always larger than or equal to the one obtained after exchange. Therefore, LPT minimizes the number of transmitted frames. $\square$

3.3 Computation

Hereafter, we recall the principle of the algorithm in order to determine the worst-case backlog in terms of frames, NbF^h, of a node h. NbF^h is determined as the maximum difference between:

- the cumulative arrival curve of every incoming flow v_j on h, the earliest arrival date of each frame being deduced from its RBF (see Section 3.2) ;
- the cumulative service curve of the incoming frames with a LPT policy.

For any value of t, the difference between these two curves is computed. Finally, computation stops at the first intersection between these curves (empty buffer). NbF^h is the maximum computed difference.

4. EXPERIMENTATION

Hereafter, we apply our buffer dimensioning approach in terms of frames on a sample configuration. Moreover, we compare our results to a more naive technique based on the computation of the worst-case backlog in terms of bits.

The details of our experimentation are presented in Section 4.1. Section 4.2 describes a detailed example of the application of our approach to the buffer dimensioning in terms of frames of a particular node. Finally, in Section 4.3, the results of the global computation and relevant comparisons are presented.

4.1 Case study

As introduced in Section 1, to apply our approach we need a configuration with the knowledge of the network topology, the traffic contracts and WCTT for the flows.

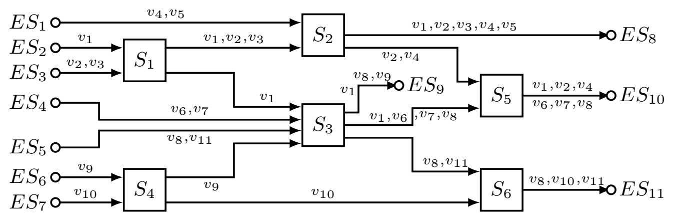

Figure 5: An AFDX configuration

An AFDX network [1] interconnects a set of End-Systems (ESs) by the mean of switches and physical links. The ES models the ingress/outgress points of the network. ESs have single output port with a FIFO buffer. An AFDX switch is composed of a set of input and output ports. Each output port is defined by a single FIFO buffer. A switch retrieves incoming frames on its input ports, applies some traffic policing and routes them on the intended output ports, according to the path of the flows. The servicing rate of all the output ports is constant and equal to $R = 100\,\text{Mbps}$, and the frame sizes are compliant with the Ethernet standard (between 64 and 1518 bytes).

The ESs exchange frames through Virtual Links (VLs). A VL is a virtual communication channel with allocated bandwidth. It defines an unidirectional link from a source ES to one or more destination ESs (multicast allowed) with static routes. Every VL enforces at its ingress point a traffic contract consisting in a minimum time interval between the emission of two consecutive frames (called Bandwidth Allocation Gap (BAG)) and a maximum frame size $Fmax$. From these parameters, we can deduced the maximum transmission time C_i as $\frac{Fmax}{R}$, whereas its period T_i is directly given by its BAG.

A typical airborne AFDX configuration is composed of about one hundred ESs, interconnected by a thousand of VLs through a dozen of switches. A sample configuration composed of 11 end systems (ES_1 to ES_{11}), exchanging 11 virtual links (v_1 to v_{11}) via 6 switches (S_1 to S_6) is illustrated in Fig. 5. A virtual link is modeled by one (in case of a unicast VL) or multiple paths (in case of a multicast VL). As an example, v_4 is a multicast flow modeled as two paths: $ES_1 - S_2 - ES_8$ and $ES_1 - S_2 - S_5 - ES_{10}$.

The maximum jitter required in order to define the RBF functions of a flow is obtained with an ETE delay analysis framework. For this experimentation, we chose the FA approach [11, 10]. The AFDX configuration from Fig. 5 is transposed into the FA model (as depicted in Fig. 6) to be analyzed. Each node represents a multiplexing point corresponding to an ES or a switch output port. Since the delay incurred in the destination ES is implementation specific, it is not included in the computation. Therefore, the destination nodes are considered as the output port of the last crossed switch. A multicast VL is modeled by a set of flows, duplicated at each fork in its path. The characteristics of the flows from the configuration depicted in Fig. 6 are given in Table 1.

4.2 Detailed computation

In order to detail the principle of the method, we focus on the worst-case backlog computation in terms of frames for node S_{31} (see Fig. 6). S_{31} is crossed by three flows: v_1, v_8 and v_9. Their characteristics are given in Table 1. The com-

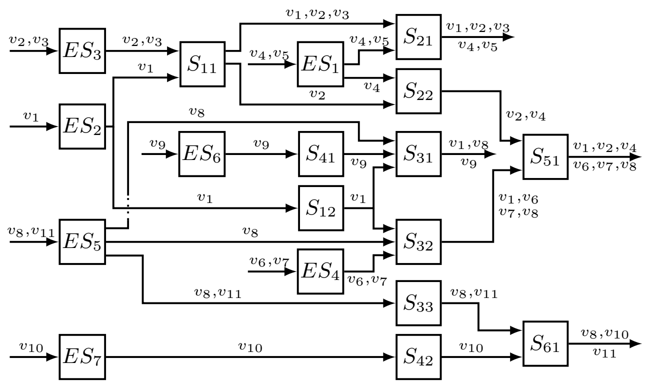

Figure 6: AFDX configuration depicted in Fig. 5 transposed in our model.

	$v_1,\dots,v_5$	v_6	v_7	v_8	v_9	v_{10}	v_{11}
C_i	10	38	12	22	64	22	22
T_i	60	320	150	80	126	48	320

Table 1: Characteristics of the flows of the sample configuration

puted jitter for these three flows at their arrival in S_{31} are: $J_1^{S_{31}} = J_9^{S_{31}} = 0\,\mu\text{s}$ and $J_8^{S_{31}} = 22\,\mu\text{s}$. The corresponding RBFs can be determined using Formula (1). The cumulative arrival curve is the sum of these RBFs. Fig. 7 shows this cumulative arrival curve and also the service curve resulting from the LPT servicing policy.

At time 0, one frame from each flow arrives in S_{31} (for each flow, $0 \times T_i \leq J_i^{S_{31}} < T_i$). The frame from v_9 (the longest one) starts its transmission immediately, according to the LPT strategy. It is fully transmitted after $64\,\mu$s. At time $58 = T_8 - J_8^{S_{31}}$, an additional frame generated by v_8 arrives in S_{31}. Similarly, a new frame from v_1 (*resp.* v_9) is accounted at time 60 (*resp.* 120). At time 64 according to the LPT policy, a frame from v_8 is transmitted. Similarly, the two curves are constructed until time 372, where the two curves intersect.

The maximum difference between these two curves, corresponding to the worst-case buffer occupancy in terms of frames, is obtained between time 60 and 64 and is equal to $NbF^{S_{31}} = 5$.

4.3 Results

The worst-case buffer dimensioning in terms of frames (NbF^h) has been conducted using our approach for each node from Fig. 6. In parallel, FA has been used to determine the maximum backlog in term of bits in each node. Another worst-case backlog value has then been determined by dividing these values in bits by the local minimum frame size in each node. This is what we refer to as the *naive* computation. The results obtained by these two approaches for each node are resumed in Table 2.

Regarding End Systems, the results obtained by the two approaches are the same except for node ES_4. In fact, in all the other ESs, all the flows have the same transmission time C_j. Thus, both methods give very close values of backlogs in terms of frames.

The highest differences between the two approaches are

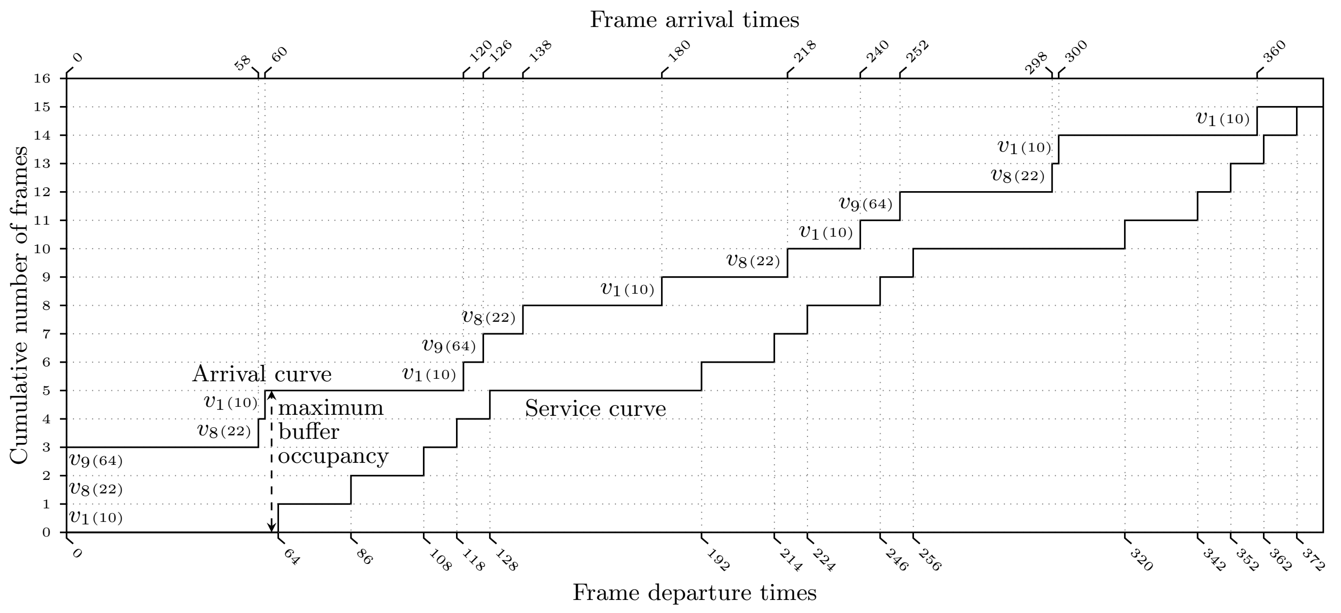

Figure 7: Worst-case buffer occupancy in S_{31}.

	Per bits approach		Per frames approach
Node	Backlog (bits)	Naive (frames)	Backlog (NbF^h) (frames)
ES_1	2000	2	2
ES_2	1000	1	1
ES_3	2000	2	2
ES_4	5000	5	2
ES_5	4400	2	2
ES_6	6400	1	1
ES_7	2200	1	1
S_{11}	3000	3	3
S_{12}	1000	1	1
S_{21}	5000	5	5
S_{22}	2000	2	2
S_{31}	9600	10	5
S_{32}	8200	9	4
S_{33}	4400	2	2
S_{41}	6400	1	1
S_{42}	2200	1	1
S_{51}	13400	14	13
S_{61}	6600	3	3

Table 2: Comparison of the two approaches for determining worst-case buffer occupancy in terms of frames.

observed in nodes S_{31} and S_{32}. This is because of the high differences in the transmission times of the flows crossing theses nodes. For example, in node S_{31}, we have a shortest (*resp.* largest) transmission time equal to 10 (*resp.* 64). Therefore, the division of the backlog (9600 bits) by the minimum frame size ($10 \times R = 1000$ bits) leads to widely overestimate the number of frames.

Furthermore, considering node S_{51}, the ratio between the shortest and largest transmission time is similar to the one observed in S_{32}. Although the results are close for S_{51} (14 frames versus 13 with our method), the difference is important for S_{32} (9 frames vs. 4 with our method). The explanation lies in the number of shorter and of longer frames in the nodes. In fact, only two flows generating frames with a transmission time lower than $12\,\mu$s are present in S_{32}, whereas there are four of them in S_{51}. Therefore the relative cost of the larger frames is decreased in the naive approach.

In order to get the total amount of bits needed for a buffer, the number of frames has still to be multiplied by the maximum frame size. Obviously, with our approach, this amount would be greater than the backlog computed initially with the FA method. However, the backlog value from FA is only suitable in the context of dynamic memory allocation. When a static design is required, the results obtained with our approach can only be compared with the higher values obtained using the naive approach.

5. CONCLUSION

Buffer dimensioning is required for certification reason in avionics networks. The worst-case backlog in terms of bits is suitable if dynamic memory allocation is allowed. But in avionics systems, static design and fixed-size memory slots are often preferred. Therefore, a worst-case buffer dimensioning in terms of frames is more relevant. We showed that it is possible to get a tighter dimensioning by using a specific approach for this computation, rather than using a more naive method, consisting in dividing the worst-case backlog in bits by the smallest frame size. Our computation uses as input the network topology, the traffic contracts and the WCTT values for the flows, which can be obtained with any appropriate framework, such as NC or FA.

We showed that tracking the worst-case buffer occupancy in terms of frames with a FIFO servicing policy is hard. Therefore, we analyze separately the arrival and transmission of frames in a node. We maximize the number of arriving frames in a node with *Request Bound Functions*, and we use the LPT as the servicing algorithm. We prove that LPT is optimal to minimize the number of served frames among non-preemptive work-conserving policies. Finally, the worst-case backlog in terms of frames is obtained by computing the difference between cumulative arrival and service curves.

We experimented our approach on a sample AFDX configuration, using FA in order to evaluate WCTT of flows. The results have been compared with the one obtained with the more naive approach based on the backlog in terms of bits. The approach we proposed leads to tighter bounds than the naive one, especially when the difference between the transmission time of frames crossing a node is important.

6. REFERENCES

[1] *ARINC 664, Aircraft Data Network, Parts 1,2 7. Technical report, ARINC specification 664.*, 2002-2005.

[2] H. Bauer, J.-L. Scharbarg, and C. Fraboul. Worst-case backlog evaluation of avionics switched ethernet networks with the trajectory approach. In *Real-Time Systems (ECRTS), 2012 24th Euromicro Conference on*, pages 78–87. IEEE, 2012.

[3] J.-Y. L. Boudec and P. Thiran. *Network calculus: A Theory of Deterministic Queuing Systems for the Internet*, volume 2050. Springer Verlag, 2001. ISBN: 3-540-42184-X.

[4] R. Coelho, G. Fohler, and J.-L. Scharbarg. Dimensioning buffers for afdx networks with multiple priorities virtual links. In *Digital Avionics Systems Conference (DASC), 2015 IEEE/AIAA 34th*, pages 10A5–1. IEEE, 2015.

[5] R. L. Cruz. A calculus for network delay, part i: Network elements in isolation. *IEEE Transactions on Information Theory*, 37(1):114–131, January 1991.

[6] R. L. Cruz. A calculus for network delay, part ii: Network analysis. *IEEE Transactions on Information Theory*, 37(1):132–141, January 1991.

[7] M. L. Dertouzos. Control robotics:the procedural control of physical processors. *Proceedings of the IFIP congress*, pages 807–813, 1974.

[8] R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal on Applied Mathematics*, 17(2):416–429, March 1969.

[9] J. Grieu. *Study and evaluation of the switched Ethernet technology for avionics systems interconnection.* PhD thesis, Institut National Polytechnique de Toulouse, 2004.

[10] G. Kemayo, N. Benammar, F. Ridouard, H. Bauer, and P. Richard. Improving afdx end-to-end delays analysis. In *Emerging Technologies & Factory Automation (ETFA), 2015 IEEE 20th Conference on*, pages 1–8. IEEE, 2015.

[11] G. Kemayo, F. Ridouard, H. Bauer, and P. Richard. A forward end-to-end delays analysis for packet switched networks. In *22nd International Conference on Real-Time Networks and Systems*, Versailles, France, October 2014. RTNS 2014.

[12] S. Martin and P. Minet. Holistic and trajectory approaches for distributed non-preemptive fp/dp* scheduling. In *ICN*, pages 296–305, Berlin Heidelberg, 2005. Springer-verlag.

[13] S. Martin, P. Minet, and L. George. End-to-end response time with fixed priority scheduling : trajectory approach versus holistic approach. *International Journal of Communication Systems*, 18(1):37–56, February 2005. Chichester, UK.