Performance Evaluation of Ethernet for Supporting the Vehicle Body and the Multimedia Domains ant its comparison to FlexRay Bus

Azer Hasnaoui, Ikbel Mejri, Manel Takrouni, Tahar Ezzeddine, Salem HASNAOUI
University of Tunis El-Manar
National Engineering School of Tunis - ENIT
Communication Systems Research Laboratory SYSCOM - LR-99-ES21, Tunis-Belvédère, BP 1002, Tunisia
Hasnaoui.Azer@gmail.com, ikbel.mejri@gmail.com, manel_takrouni@yahoo.fr, Tahar.Ezzeddine@enit.rnu.tn, Salem.Hasnaoui@enit.rnu.tn

Abstract—we designed a SIMULINK vehicle blockset that corresponds to the Society of Automotive Engineers (SAE) benchmark as a first level of CBSE architecture. We introduced a second level of abstraction where we connect each module to a Simulink DDS blockset to take advantage of the QoS parameters, events generated in response to faults or exceeding of fixed parameters and calls to callback functions, managed within the DDS middleware itself without involving the user tasks realized by the modules in the application level.

The powertrain and chassis modules are connected to FlexRay bus. We chose to use the FlexRay network for its fault-tolerant dual channel bus (physically independent cables), where a node can be connected to one or both of the busses. A node connected to both busses can send the same or different messages on the two busses. Sending the same message on both busses increases the fault-tolerance. However, FlexRay does not meet the bandwidth and scalability requirements of next-generation advanced driver Assistance systems, V2X and RadCom communications. Giga-Ethernet and Wireless High-speed communications are the emerging technologies in the automotive domain, specifically in the body and multimedia domains. In this paper, we are interested to computations related to Ethernet and we demonstrated that the Body and Multimedia domains can be connected to it with a minimum latency budget.

Keywords—GBE; FlexRay; V2V; V2I; QoS; SAE Benchmark; Vehicle Blockset; DDS

1. INTRODUCTION

THEoriginal SAE benchmark is limited to the automotive kernel model. We implemented a Simulink Blockset corresponding to the different blocks and integrated the original and the extended models studied by Utayba [1] and by our team [2]. The authors added to the original benchmark a number of nodes and messages to better represent the complexity of today's vehicles and to model some options responsible for improving vehicle safety and reliability.

The most of the vehicle nodesare until now connected by the CAN bus. We eliminated from our studies this bus and we replaced it by the FlexRay for the powertrain and chassis domains and by the Ethernet for the body and multimedia domains.

Future vehicles using V2V (Vehicle-to-Vehicle) and V2I (Vehicle-to-Infrastrucre) technologies will move one step closer to reality by using MIMO radars in order to enable a variety of applications for safety and traffic efficiency. By integrating V2I technology, into all vehicles having V2V, systems might even reduce all target vehicle crashes up. V2I technology would essentially permit a car to request information to access to the best possible road routes to a particular destination [3, 4]. Dangerous intersections would also be made safer through the use of V2I. Infrastructures would be able to warn vehicles to slow down or communicate the status of a traffic light from a given distance.

It is expected that car sensors will generate up to 1 TB of data in a single trip, so, FlexRay does not meet the bandwidth and scalability requirements of next-generation advanced driver assistance systems. Giga-bit Ethernet and high-speed wirelesscommunications will play the leading role in the near future. In this paper we will done some computations to demonstrate that Ethernet (GBE) is a serious candidate for vehicle communications. We compare its performances to FlexRay bus for some vehicle modules. We take into account the DDS parameters and we proved that using Ethernet combined with the DDS middleware is a promising alternative for FlexRay or CAN.

The paper is organized in four sections. In the second section we recall briefly our implementation of SAE vehicle and DDS blocksets.

In the third section we done some computations related to Ethernet, specifically the time before the occurrence of an error or a collision. We showed that this time is enormously large comparatively to the car's life.

In the fourth section, we calculated the worst-case response time based on the full scheduling model, and we introduced it into the DDS QoS to further prove that SAE benchmark can be best insured by DDS and the FlexRay network in powertrain and chassis domains and DDS and Ethernet in the other domains. We proved that the DDS related timing QoS are guaranteed; especially the deadline and the minimum separation, taking account of latencies induced by the FlexRay vs Ethernet buses.

2. DESIGN AND IMPLEMENTATION OF AVEHICLE AND DDS SIMULINK BLOCKSETS

2.1 Vehicle SIMULINK Blockset

In order to test if the exchanged messages are valid as described by the standard without regarding the targets

where the modules will be implemented, we applied the Model-Based Design (MBD) to develop the vehicle blockset.The resulting architecture is composed of 15 nodes connected by the FlexRay/ Ethernet buses.We implemented the Suspension model, the Wheels model, the Active-Frame-Steering model and Electronic-Brake-Control model. The Suspension model consists of three sub-blocks: Passive-Suspension block, Active-Suspension block and Active-Suspension-Control-Force block. Whereas, the wheels model gathers four sub-blocks: the Front-Left-Wheel, Front-Right-Wheel, Rear-Left-Wheel, and Rear-Right-Wheel [5, 6]. The figure 1 presents the blocksets of the modules cited above.

Fig. 1.Developed Vehicle SIMULINK Library

2.2 DDS Middleware SIMULINK Blockset

The most popular classes of middleware are: RPC, RMI, CORBA and DCOM. They offer a remote method invocation and are familiar with the OO programming model. They use synchronous invocations, have limited QoS and have cascading points of failure, typically built on top of TCP. They are best-suited to smaller and closely-coupled systems.

As it is known, the DDS (Data Distribution Service) became de-facto the standard in embedded systems that address the challenges in data-centric real-time applications.

The DDS is an open standard managed by the Object Management Group (OMG) and representing the first general-purpose middleware standard that addresses challenging real-time requirements. It has a large number of QoS configuration parameters that give developers complete control of each object in the system and maintainability of its state. Its Data-Centric Publish-subscribe layer (DCPS) consists of the following Entities: domain-Participant, DataWriter, DataReader, Publsiher, Subscriber and topic. The objective is to transmit data directly from a publisher to all its subscribers with no intermediate servers. This allows the application to communicate by publishing the data it produces and to have access to the type of data it consumes.

We argue our choice of the DDS middleware to interface the low-level infrastructure and to realize a SIMULINK blockset for it by the following reasons:

- No single point of failure: DDS requires any "special" nodes, so it can be implemented with no single-points-of-failure due to the redundancy of publishers and subscribers.
- Self-healing communication: If the network is severed into two halves, each half will continue to work with the available nodes. If the network is repaired, the network, by the built-in discovering entities, will quickly rediscover the new nodes, and once again function as a whole.
- Support for custom fault-tolerance: Implementations are free to add further fault tolerance as well like FFT [2]. The support of multiple network interfaces like channel-A and Channel-B in a FlexRay controller and redundant Data Writers and Data Readers on every node leads to completely separate networks. Even if one network fails completely, the system will continue operation.

The figure 2 depicts the DDS SIMULINK library blocks. The topic block is for assembling some output signals within a structure where we can attribute an unique key for this topic. The DataWriter_write() or the DataRead_Read() or the DataRead_Take() operation use the instance of "Data" to send or to receive the actual data, in occurrence into or from queues [7, 8, 9].

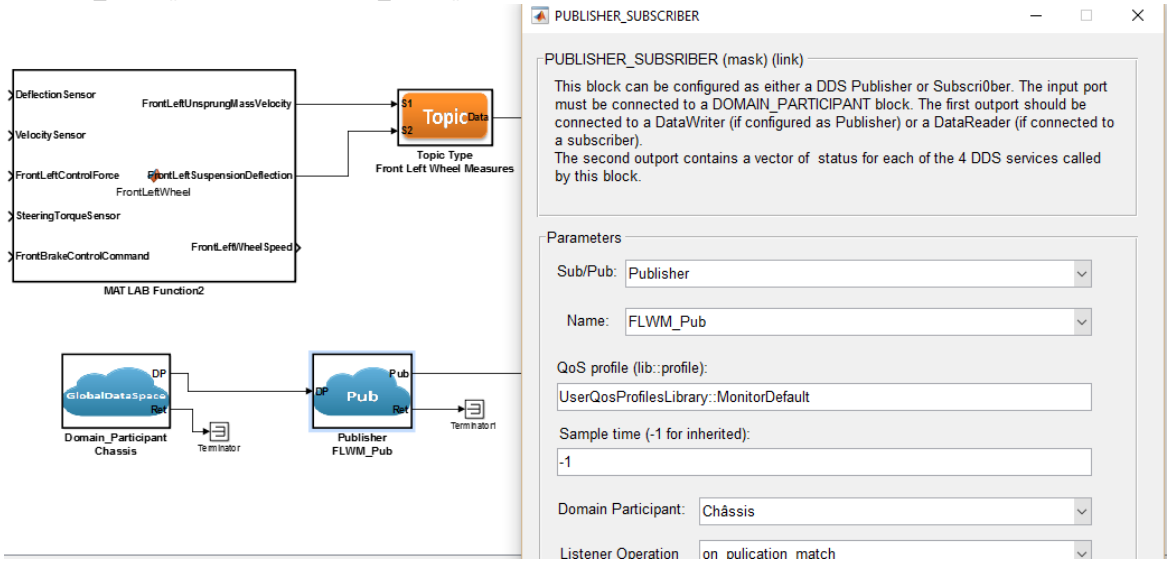

Fig.2. DDS blockset applied to Front Left Wheel Module (chassis domains)

3. Analysis of real-time capabilities of the Ethernet for speeds: 10Mbps, 100Mbps, 1Gbps and 10Gbps [10]

31 Algorithm for resolving the access contentions: the Ethernet case

When a collision occurred, the node that detected the collision draws randomly a number of time slots that it must decline before retransmitting the same frame. This initial value varies between 0 and 2 (0, 1 or 2 time slots). A time slot for the Ethernet is set to 51.2 microseconds when the throughput used is 10 Mbps. The slot value is reduced to 5.12 microseconds when the throughput is 100 Mbps. let$W_{min} = 2$.This value is called backoff, noted bc $bc \epsilon[0,\ W_{min}]$. If the collision happens again, the node in question doubles the drawing interval. The second time belongs to$[0,\ 2W_{min}]$.The third draw$bc \epsilon[0,\ 2^2W_{min}]$and so on. For the nth draw$bc \epsilon[0,\ 2^{n-1}W_{min}]$.

The"backoff" algorithm stops only when the channel is acquired. As CSMA/CA for WLAN networks, the sampling interval does not increase indefinitely but is limited to ten successive draws before concluding the non-possibilityto use of the medium. The maximum interval is$[0,\ 2^9W_{min}]$or $[0,\ 1024]$.The sampling interval increases exponentially.

For real-time systems that hard determinism is required, the access time should be very limited to ensure a very small latency before the deadline ofthe sending of periodic messages.

Latency should always be less than the transition time of any system from one state to another state. As far as the evolution of the system is faster as far as the latency should be limited for this reason, the real-time critical systems are generally limited to physical, MAC and LLC layers.

Analysis of the possibilities of Ethernet determinism to is based on the calculation of the following parameters:

• The calculation of the probability of collisions. When this probability is higher, the backoff grows exponentially and the latency is greater.

• The penalty caused by the presence of collisions on the whole system, expressed in terms of the probability of failure for a specified period.

For the calculation of probabilities mentioned above we will make the following assumptions:

1. The packets exchanged have fixed size and short length.
2. Each subnet (Powertrain, Chassis, Body and Multimedia) is lightly loaded. The number of messages sent per time unit is not high.

The packet arrival law follows the Poisson distribution ; i.e. the arrival process is Markovian.

32 Calculating the waiting probability of the two nodes and its generalization to M nodes

We adopt for the following the following notations:
S_p : Packet size in bytes (or$8S_p$in bits);
λ :The arrival rate in packets per second (packets/ sec) ;
B_w : The bandwidth in bits per second (link speed);
t_p :The mean propagation time in the network (in sec);
T_{max} : The maximum acceptable delay (in sec);
N : The number of collisions that may occur during T_{max} ;
P_{err} : The error probability for N produced collisions;
P_{Succ} :The likelihood of sending packets and whose delay does not exceedT_{max} ;

M : The number of packets sent before the occurrence of an error;
T_{err} : The time elapsed before to produce an error.
P_{col} : The collision probability

The arrival law according to the Poisson process is defined by:

$$p_n(t) = e^{-\lambda t}\frac{(\lambda t)^n}{n!} \text{(3-1);}$$

This function is the probability density function of the Poisson process. The probability distribution function of the Poisson distribution is defined as:

$$F_x(x) = P(X > x) = \int_x^{\infty} p_n(t)dt \quad \text{(3-2),}$$

(i.e. the probability that the random variable X exceeds the value x)
In the discrete case where t is a finite time interval denotedT, divided into time slotsΔt, the probability distribution function becomes:

$$F_x(x) = e^{-f}\sum_{k=0}^{\infty}\frac{f}{k!}u(k-x) \text{(3-3),}$$

Where $u(k-x)$is the unitary function and

$$f = \lambda T = \lambda\frac{8S_p}{B_w} \text{(3-4);}$$

$\frac{8S_p}{B_w}$represents the size of information in bits divided by the speed in bits/ sec($\sim$ Bandwidth).

If the number of nodes in a subnet is lower than or equal to 2; there are no collisions. Both nodes can develop agreements to not send at the same time just by managing they protocol FSMs (Finite State machine). However, when more nodes exist in the network, this technique cannot be applied and collisions may occur.

The probability that three nodes have their packets in output queues according to the principle of CSMA/CD, outside the contention for access to the medium, is that one node transmits and the other two nodes wait to win the media access and then transmit.
The two nodes waiting probability is represented by:$P_{wait}^2 = 1 - F_x(2) = 1 - e^{-f}\sum_{k=0}^{2}\frac{f}{k!}u(k-x)$ (3-5)

$= 1 - e^{-f}(1 + f + \frac{f^2}{2})$ (3-6)

The probability that M nodes wait to transmit becomes:$P_{wait}^M = 1 - e^{-f}(1 + f + \frac{f^2}{2} + \cdots + \frac{f^M}{M!})$(3-7)

We note thatP_{wait}^M is slightly greater thanP_{wait}^2. This remark is very important when discussing scenarios in the case of two nodes and then we generalize to multiple nodes, since the waiting time is equivalent (slightly higher).

33 Scenario for stations in repeat collisions and collision probability calculation

Consider the case where the two nodes each wants to win the support but they collide.

First pulling

- Both nodes start the backoff procedure for pulling bc $\epsilon[0, W_{min}]$,interval of integer values; with$W_{min} = 2$. Unfortunately, they pull the same bc value. The probability of pulling the same value ofbc est $\frac{1}{3}$;a value among$\{0, 1, 2\}$
- After the bcinterval, they return in collision.

Secondpulling

- Both nodes start the backoff procedure for a second bc draw which now belongs to the set of values$\{0, 1, 2, 3, 4\}$. i.e bc $\epsilon[0, 2W_{min}]$. Unfortunately they pull the same bc value for the second time. The probability of drawing the samebc value is$\frac{1}{5}$; a value among$\{0, 1, 2, 3, 4\}$.
- After the bc interval they collide again.

Third pulling

- bc is among one of the values$\{0, 1, 2, 3, 4, 5, 6, 7, 8\}$ i.e bc $\epsilon[0, 2^2W_{min}]$and the probability of having the same value of bc is$\frac{1}{9}$.
- The probability to reach the third draw is simply:$P_{col} = \frac{1}{3}$ x $\frac{1}{5}$ x $\frac{1}{9} = \prod_{k=1}^{3}\left(\frac{1}{2^{k-1}W_{min}+1}\right)$ (3-8)

- - - - - - - - - - - -

Nthpulling

- The probability to get to the Nth draw (i.e bc $\epsilon[0, 2^{N-1}W_{min}]$) is simply:

$$\boxed{P_{col} = \prod_{k=1}^{N}\left(\frac{1}{2^{k-1}W_{min}+1}\right) = \prod_{k=1}^{N}\left(\frac{1}{2^k+1}\right)} \quad \text{(3-9)}$$

34 $T_{max} = 2^{N-1}W_{min}$ x *Time Slots* and the CollisionProbability P_{col} Computation

The probability to reach the 10th draw is very low ($1.2\,x\,10^{-17}$). This is why the maximum number of pullings allowed is 10 and the number of slots that can be lost without any network node is able to connect is 1024. The value$T_{max} = 2^{N-1}W_{min}$ *time slots*expresses the maximum acceptable time there without success because repetitions of collisions. However, we notice that every timeT_{max} increases the probability of collision is very low.

The table 2 below summarizes the computation of$T_{max} = 2^{N-1}W_{min}$ and the probability of collisionP_{col}

The time slot is the time required to wait for the medium to be free from transmissions. This time slot is fixed to51.2µs for the Ethernet when the throughput used is 10 Mbps;however, it could be replaced by any positive value. Slot time is only applicable to half-duplex transmissions, there is no time required to wait for full-duplex transmissions.10 Gbit/ s is a full duplex technology, so slot time is not applicable. Table 1 summarizes the time slots used for Ethernet.

TABLE 1 ETHERNET TIME SLOTS

Speed	Slot time	Time Interval
10 Mbit/s	512 bit times	51.2 µs
100 Mbit/s	512 bit times	5.12 µs
1 Gbit/s	4096 bit times	4.096 µs
10 Gbit/s	Not applicable	Not applicable

$T_{max} = 2^{N-1}W_{min}$ is the maximum acceptable delay before successfully transmitting a new frame. This time is considered the latency imposed by the access technology. It goes without saying that this latency should be much less than the deadline for transmitting a new message. For powertrain and chassis domains, the deadline is less than $5\,ms$, thus Ethernet at $10\,MBPS$ don't match with theses constraints.

TABLE 2: $T_{max} = 2^{N-1}W_{min}$ AND P_{col}

N number of attempts (pulling)	Number of time slots for connections attempts : $2^{N-1}W_{min}$	Time (msec) taken for speed =10 MBPS	Time (msec) taken for speed =100 MBPS	Time (msec) taken for speed =1 GBPS	$\prod_{k=1}^{N}\left(\frac{1}{2^{k-1}W_{min}+1}\right)$ collision probability for N retries
1	2	2x51.2 µs ≅ 0.1 msec	2x5.12 µs = 0.01	2x4.096 µs =	3.33×10^{-1}
2	4	4x51.2 µs ≅ 0.2	0.02	4x4.096 µs =0.016 msec	6.66×10^{-2}
3	8	0.4	0.04	0.032	7.407×10^{-3}
4	16	0.8	0.08	0.064	4.0×10^{-4}
5	32	1.6	0.16	0.131	1.3×10^{-5}
6	64	3.2	0.32	0.256	2.0×10^{-7}
7	128	6.4	0.64	0.524	1.5×10^{-9}
8	256	12.8	1.28	1.048	6.0×10^{-12}
9	512	25.6	2.56	2.096	1.1×10^{-14}
10	1024	1024x51.2 µs = T_{max}≅ 51.2 msec	= T_{max}≅ **5.12**	4.194	1.2×10^{-17}

35 Probability of Error, Probability of Success and Time between Two Consecutive Errors Computation

The probability of error P_{err} is equal to the product of the probability of waiting connectionby the probability of loss of connections for N attempts.

$$P_{err} = P_{wait}^{M}\ x\ \prod_{k=1}^{N}\left(\frac{1}{2^{k-1}W_{min}+1}\right) \approx P_{wait}^{2}\ x\ \prod_{k=1}^{N}\left(\frac{1}{2^{k-1}W_{min}+1}\right) \quad (3\text{-}10)$$

$$= [1 - e^{-\lambda\frac{8S_p}{B_w}}(1 + \lambda\frac{8S_p}{B_w} + \frac{\left(\lambda\frac{8S_p}{B_w}\right)^2}{2})]\ x\ \prod_{k=1}^{N}\left(\frac{1}{2^{k-1}W_{min}+1}\right) \quad (3\text{-}11)$$

The probability of success transmission is the probability of having no errors $(1 - P_{err})$but it is possible to have Ssuccessfully transmissions before reachingT_{max}.

The probability of success transmission is then$P_{succ} = (1 - P_{err})^{S}$ (3-12)

Srepresents the number of transmissions before an error can occur i.e. S can be converted to time by:$T_{err} = \frac{S}{\lambda}$ (3-13)

Scan be calculated from the equation 3-12thus the time of successful transmission (or elapsed timeT_{err} before the occurrence of an error) is calculated as follows:

$$T_{err} = \frac{1}{\lambda}\ x\ \frac{Ln(P_{succ})}{Ln(1-P_{err})} \quad (3\text{-}14)$$

with

$$P_{err} = [1 - e^{-\lambda\frac{8S_p}{B_w}}(1 + \lambda\frac{8S_p}{B_w} + \frac{\left(\lambda\frac{8S_p}{B_w}\right)^2}{2})]\ x\ \prod_{k=1}^{N}\left(\frac{1}{2^{k-1}W_{min}+1}\right) \quad (3\text{-}15)$$

This equation is used to calculate the probability of error in terms of link parameters. The values$T_{err} = \frac{S}{\lambda}$will be calculated if we impose a value ofP_{succ}.In what follows we impose that $P_{succ} = 99\%$.

The following table 3 gives an idea of the average time between two errors and dependent on communications parameters: Flow rate, packet size, transfer rate and the

maximum acceptable delay (deadline) for the speed 100 Mbps, which is the minimum speed to be used in automotive and aeronautic domains.

TABLE 3: COMPUTATION OF $T_{err} = \frac{1}{\lambda} x \frac{Ln(P_{succ})}{Ln(1-P_{err})}$

Bandwidth B_W (Mbits/s)	Packet size S_p (bytes)	Transmission rate λ in packets/s	$T_{max} = 2^{N-1}W_{min}$ (msec)	mean time between two errors (P_{succ} = 99%)
100	64	1000	3.0	1000000 years
100	128	1000	2.0	300000 years
100	128	1000	1.0	1140 years
100	128	1000	0.5	9 années
100	**128**	**5000**	**1.5**	**483** years
100	1024	1000	2.0	604 years
100	1024	1000	1.0	2 years
100	1024	2000	1.5	40 years
100	1024	3000	1.5	8 years

The values of T_{err} give a good idea on the occurrence of collisions resulting in errors. The mean time between two consecutive errors is relatively high. We can say for certain cases there are no collisions !!.

We conclude from this deep study that Ethernet when the speed is higher than 100 Mbps and FlexRay are both deterministic. When the speed of Ethernet reaches 10 Gbps, we can apply Ethernet also for powertrain and chassis domains without fear of the access time, which is due to the occurrence of collisions.

4. VALIDITY OF DDS TIMING ON TOP OF ETHERNET/FLEXRAY WHEN APPLIED FOR VEHICLE DOMAINS.

In this section we will use the calculated worst case response time to evaluate if the DDS QoS real-time parameters can be met in the SAE benchmark implemented in SIMULINK by our team. We will focus our interest on two real-time policies, the Deadline QoS policy represented by the deadline parameter D, and the time based filter policy represented by the parameter minimum_seperation period, Min_Sep.

41 Worst Case Response Time Evaluation of DDS QOS in SAE application using FlexRay Network Based on full scheduling model

In our previous researches [8] we were interested in scheduling for the Data Distribution Service (DDS) architecture over CAN. In this paper, we focus our interest on the scheduling on the FlexRay and Ethernet. We have proposed a new scheduling method that handles all the delay sources. We have used the FPS (Fixed Priority Scheduling) approach, which is the most widely used approach in the computing world. In this case, each task has a fixed, static, priority, which is ECU pre-run-time. The runnable tasks are executed in the order determined by their priority, knowing that in real-time systems, the "priority" of a task is derived from its temporal requirements, not its importance to the correct functioning of the system or its integrity. The full model was conceived to be used in an industrial context. In this case, the response time equation is rather than:

$$R_i = C_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j \quad (4\text{-}1)$$

Where $hp(i)$ is the set of tasks with priority higher than task i, C_i is the worst case computation time of the task i and T_j is the minimum time between task releases, jobs or task period. The new equation is:

$$R_i = CS^1 + C_i + B_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil (CS^1 + CS^2 + C_j) \quad (42)$$

Where the new terms CS^1 and CS^2 are the cost of switching to the task, and the cost of switching away from it. The term B_i is the cost of the task worst case blocking time. The cost of handling interrupts is as flowing:

$$\sum_{k \in \Gamma_s} \left\lceil \frac{R_i}{T_k} \right\rceil IH \quad (4\text{-}3)$$

Where Γ_s is the set of sporadic tasks and IH is the cost of a single interrupt (which occurs at maximum priority level). There is also a cost per clock interrupt, a cost for moving one task from delay to run queue and a reduced cost of moving groups of tasks. CT_c is the cost of a single clock interrupt, Γ_p be the set of periodic tasks, and CT_s be the cost of moving one task the following equation can be derived

$$R_i = CS^1 + C_i + B_i + \sum_{j \epsilon\, hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil \left(CS^1 + CS^2 + C_j\right) + \sum_{k \epsilon \Gamma_s} \left\lceil \frac{R_i}{T_k} \right\rceil IH + \left\lceil \frac{R_i}{T_{clk}} \right\rceil CT_c + \sum_{g \epsilon \Gamma_p} \left\lceil \frac{R_i}{T_g} \right\rceil CT_s \quad (4\text{-}4)$$

42 Full model applied on the static segment tasks

In the static segment, all communication slots have identical, statically and configured duration and all frames have identical, statically and configured length. In order to schedule transmissions each node maintains a slot counter state variable $vSlotCounter$ for channel A and a slot counter state variable $vSlotCounter$ for channel B. Both slot

counters are initialized with 1 at the start of each communication cycle and incremented at the end boundary of each slot. In the Implementations of the FlexRay bus, the periodic and safety-critical data is scheduled on the static time-triggered segment. In the static segment tasks are periodic, having the same priority per communication cycle. Considering these facts the equation (4-4) applied on the static segment context becomes:

$$R_i = CS^1 + C_i + B_i + \sum_{k \epsilon \Gamma_s} \left\lceil \frac{R_i}{T_k} \right\rceil IH + \left\lceil \frac{R_i}{T_{clk}} \right\rceil CT_c + g \epsilon \Gamma p Ri Tg CTs \quad (4\text{-}5)$$

43 Full Model applied on the dynamic segment tasks

In the dynamic segment, the duration of communication slots vary in order to accommodate frames of varying length. In order to schedule transmissions each node continues to maintain the two slot counters - one for each channel - throughout the dynamic segment [11, 12, 13]. The slot counters for channel A and B are incremented simultaneously within the static segment. In the Implementations of the FlexRay bus, the dynamic segment is mainly used for maintenance and diagnosis data. Tasks are event triggered sporadic having different priority by bus communication cycle. Considering these facts, the equation applied on the static segment context becomes:

$$R_i = CS^1 + C_i + B_i + \sum_{j \epsilon hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil \left(CS^1 + CS^2 + C_j\right) + \sum_{k \epsilon \Gamma_s} \left\lceil \frac{R_i}{T_k} \right\rceil IH + \left\lceil \frac{R_i}{T_{clk}} \right\rceil CT_c + \sum_{g \epsilon \Gamma_p} \left\lceil \frac{R_i}{T_g} \right\rceil CT_s \quad (4\text{-}6)$$

Since R_i appears in both parts of the equation, we must solve the problem by forming a recurrence relation:

$$W_i^{n+1} = CS^1 + C_i + B_i + \sum_{j \epsilon hp(i)} \left\lceil \frac{W_i^n}{T_j} \right\rceil \left(CS^1 + CS^2 + C_j\right) + \sum_{k \epsilon \Gamma_s} \left\lceil \frac{W_i^n}{T_k} \right\rceil IH + \left\lceil \frac{W_i^n}{T_{clk}} \right\rceil CT_c + \sum_{g \epsilon \Gamma_p} \left\lceil \frac{W_i^n}{T_g} \right\rceil CT_s \quad (4\text{-}7)$$

The set of values $w_i^0, w_i^1, w_i^2, \ldots, w_i^n$ constitutes a non-decreasing monotone sequence. When we have equality, the solution of the equation (4-7) is found. The process for calculating the response time is described by the following algorithm.

44 Worst Case Response Time Pseudo-Algorithm Computation.

```
For i in 1..N loop
    n := 0
    loop
Calculate Ci for periodic tasks
    n := n + 1
    end loop
    end loop
for i in 1..N loop
    n := 0
    Wi^n=Ci
    loop
calculate new Wi^n+1

if Wi^n+1= Wi^n then Ri= Wi^n
    exit value found
    endif
    if Wi^n+1 >Ti then
exit value not found
    end if
    n := n + 1
    end loop
```

Applying the previous algorithm with Ethernet bus speed = 100 M bit/ s, we obtain the following results presented in table 4.

TABLE 4: SAE BENCHMARK RESULTS

Vehicular module	Message ID	Size (bytes)	D (ms)	Min separation (ms)	T (ms)	Task priority	WCRT R(ms)
Body Control module	1	1	5	0.0153	50	1	0.0170
	3	1	5	0.0153	5	3	0.0339
	13	1	5	0.0153	10	13	0.0509
	17	1	10	0.0153	10	17	0.0678
	18	2	10	0.0153	10	18	0.0848
	31	4	100	0.0153	100	31	0.1017
	34	3	320	0.0153	320	34	0.1187
Engine controller module	4	2	5	0.0153	5	4	0.0171
	6	2	5	0.0153	5	6	0.0340
	19	6	10	0.0153	10	19	0.0510
	20	2	10	0.0153	10	20	0.0680
	21	3	20	0.0153	20	21	0.0849
	35	1	300	0.0153	300	35	0.1019
Front control Unit	2	2	5	0.0153	5	2	0.0161
	30	1	20	0.0153	50	30	0.0804
	32	1	100	0.0153	100	32	0.0965
	5	1	5	0.0153	5	5	0.0322
	33	1	100	0.0153	100	33	0.1125
	36	1	320	0.0153	320	36	0.1286
	7	1	5	0.0153	5	7	0.0483
	29	3	10	0.0153	10	29	0.0643
Left Wheel Unit	9	1	5	0.0153	5	9	0.0170
	23	2	10	0.0153	10	23	0.0510
	11	1	5	0.0153	5	11	0.0340
	25	2	10	0.0153	10	25	0.0679
Righ Wheel Unit	10	1	5	0.0153	5	10	0.0170
	24	2	10	0.0153	10	24	0.0510
	12	1	5	0.0153	5	12	0.0340
	26	2	10	0.0153	10	24	0.0679
Central Control Unit	27	2	10	0.0153	10	27	0.0171
	22	2	10	0.0153	10	22	0.0341
	14	4	5	0.0153	5	14	0.0510
	8	1	5	0.0153	5	8	0.0680
	15	4	5	0.0153	5	15	0.0849
	16	4	5	0.0153	5	16	0.1019
	28	5	10	0.0153	10	28	0.1188

We have noticed that the deadline has been met and the equation below is verified.

$T \geq D \geq R$(4-8)

For Ethernet, we can assume that the DDS Deadline QoS Policy always be reached.

As for the Time Based Filter we have approximated the minimum_separation parameter to be the reception delay which is for Ethernet case the transmission delay C.

Same as the DDS Deadline QoS Policy, we can assume that the Time Based Filter QoS Policy is verified.

$R \geq Min\_Sep$(4-9)

5. COMPARATIVE STUDY

To make the comparison between FlexRay and Ethernet, we propose to base our evaluation on the $WCRT$.

The goal is to determine which of the two buses (Ethernet or FlexRay) is most suitable for real-time, and to know the impact of bus speed on performance of the system.

Figures 2, 3 and 4 present the comparison between the $WCRT$(20 Mhz)and best case response time ($BCRT$, 5 MHz) of the bus flexray [12] and the $WCRT$of bus Ethernet.This comparison is made for 3 modules: front

Control Module, body central control Module and central control Module.

The responses obtained by the same Ethernet calculate scheduling.

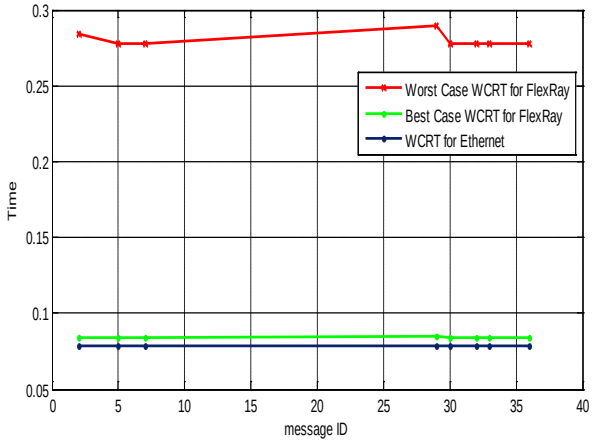

Fig.3. Comparison between $BCRT$, $WRCT$ for Flexray and $WCRT$ for Ethernet for front Control Module

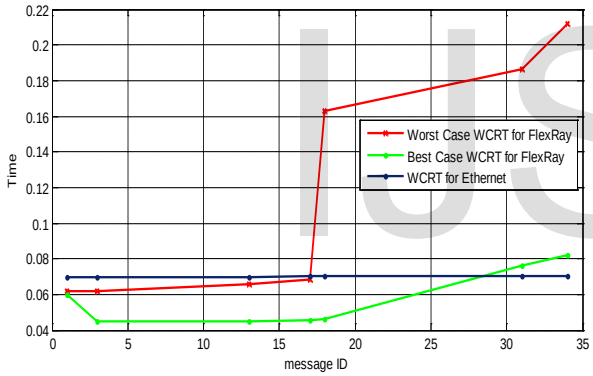

Fig.4. Comparison between $BCRT$, $WRCT$ for Flexray and $WCRT$ for Ethernet for body control Module

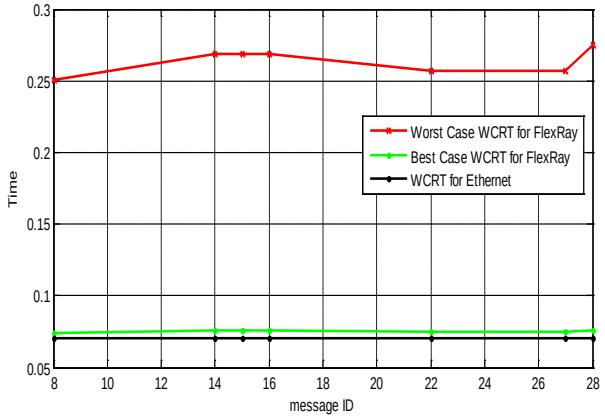

Fig.5. Comparison between $BCRT$, $WRCT$ for Flexray and $WCRT$ for Ethernet for central control Module

We note that in the case of central control module and front Control Module, the 100Mbit/s Ethernet gives better performance than the FlexRay because the bus speed is much most significant and allows a scheduling with this technique to route messages more quickly.

In the case of body control Module, we note that the temporal performances are penalized by the access technique, which handles messages according to their priority.However, it still remains very close and exceed the temporal performances of FlexRay.

6. CONCLUSIONS

In this paper, we have proposed to use DDS on top of the real-time network Ethernet to enhance the timing QoS. To do so, we have tested using vehicular applications based on the SAE benchmark. The tests have proven that using Ethernet combined with the DDS middleware is promising alternative for FlexRay or CAN. We conclude that for body and multimedia domains the Ethernet at 100 Mbps can be applied to these domains without any modification. In a further paper we demonstrate that the minimum Ethernet throughputs for powertrain and chassis domains is one Gbps and higher, however when the throughput exceeds 10 Gpbs the Ethernet becomes strictly deterministic without any modification (ie use of virtual links, etc.)

ACKNOWLEDGMENT

The researchers presented in this paper would not have been possible without the support of many people. We wish to express our gratitude to the SYSCOM ENIT members for their help and assistance.

REFERENCES

[1] M. Utayba Mohammad, N. Al-Holou, "Development of an Automotive Communication Benchmark", Canadian Journal on Electrical and Electronics Engineering, Vol. 1, No. 5, August 2010.).

[2] Pedreiras, P., Gai, P., Almeida, L., Buttazzo, G., "FTT-Ethernet: a flexible real-time communication protocol that supports dynamic QoS management on Ethernet-based systems", IEEE Transactions on Industrial Informatics, Vol.1, N°3, 2005, pp 162-172.

[3] Hirose, T., Ohtsuka, Y., and Gokan, M., "Activation Timing of a Collision Avoidance System with V2V Communication," SAE Technical Paper 2017-01-0039, 2017.

[4] Rand Raheem , Aboubaker Lasebae,,Mobile femtocell utilisation in LTE vehicular environment: Vehicular penetration loss elimination and performance enhancement, Vehicular Communications, Elseiver, Volume 9, July 2017, Pages 31–42

[5] Houda Jaouani, "Etude, Modélisation et Implémentation sous MATLAB de Modules SAE Benchmark en tant qu'entités DDS", PhD dissertation, ENIT/ Al-Manar University, December 2015,

[6] Rim BOUHOUCH, " Implémentation de DDS et des connecteurs de la radio logicielle et évaluation de leurs performances, PhD dissertation, ENIT/ Al-Manar University, March 2014,

[7] Azer Hasnaoui, Prof. Tahar EZZEDDINE, "Génération de code C des entités DDS à partir de fichiers XML", Rapport de recherche, décembre 2015.

[8] Azer Hasnaoui, "Conception et développement d'une carte à base du contrôleur FlexRay MB88121C et de son driver sous la plateforme LPC2294 et µC-OSII", End-Studies Project, TELNET company, June 2012.

[9] Manel TAKROUNI, Marwa GDHAFI, Azer HASNAOUI, Salem HASNAOUI, Design and Implementation of a Simulink DDS Blockset and its Adaptation to an SAE ElectricVehicle Blockset, SETIT 2016.

[10] Salem Hasnaoui,"Réseaux Industriels et de Télécommunications, second edition, CPU, 2013.

[11] Ben Gaid, M-M; çela, A.; Diallo, S.; Kocik, R.; Hamouche, R.; Reama A., "Performance Evaluation of the Distributed Implementation of a Car Suspension System", In proceedings of the IFAC Workshop on Programmable Devices and Embedded Systems (PDeS 2006). Brno, Czech Republic, February 2006

[12] Rim Bouhouch, Houda Jaouani, Wafa Najjar, Salem Hasnaoui, FlexRay Static Section Scheduling Using Full Model, InfoComp 2012.

[13] Marisol GarcíaValls, Patricia UriolResuela, Felipe IbáñezVázquez, Pablo BasantaVal: Low complexity reconfiguration for real time data intensive service oriented applications. Future Generation Comp. Syst. 37: 191-200 (2014).