Preprints of the 18th IFAC World Congress
Milano (Italy) August 28 - September 2, 2011

Distributed Dynamic Scheduling of Controller Area Network Messages for Networked Embedded Control Systems

Yasser Shoukry * **Hesham Shokry** ** **Sherif Hammad** **

* *Ain Shams University, Abbasia, Cario, Egypt (e-mail: yasser_shoukry@eng.asu.edu.eg).*
** *Mentor Graphics Egypt, Heliopolis, Cairo, Egypt (e-mail: {hesham_shokry, sherif_hammad}@mentor.com).*

Abstract: This paper presents an online, dynamic, and distributed message scheduling of communication packets over the controller area network (CAN) bus. Using online, dynamic, and distributed scheduling of messages, one can obtain better temporal characteristics for networked embedded control systems. The scheduling algorithm is implemented as a hardware unit which is augmented to the communication controller on all of the communicating nodes in the system. This insures minimum change in the current platforms used in CAN-based control applications like those found heavily in automotive and aerospace applications. We provide a number of experiments held over a physical CAN bus for controlling an automotive active suspension system and showing the effect of the proposed scheduling implementation on the quality of the control loop.

Keywords: Controller Area Network (CAN), Dynamic scheduling, Earliest Deadline First (EDF), Networked Embedded Control Systems (NECS)

1. INTRODUCTION

Using networked embedded control systems (NECSs) raises several new problems which need to be resolved for successful development of such control systems. Problems related to these systems attracted considerable attention during the last decades. The reason for this interest in the research area is that the use of networks offers many advantages, such as low installation and maintenance costs, reduced system wiring, and increased flexibility of the system. However, from a control theory point of view, the presence of the network in the control loop introduces several disadvantages such as time-varying networked-induced delays, aperiodic sampling and/or packet dropouts. Typically, control engineers design a control system by implicitly assuming the timely behavior of a system. However if the timeliness assumptions are violated (for example, the feedback signal sent from a sensor and recieved by the controller after the sample period has been elpased) then either the stability of the system could be disturbed or the performance of the system could be degraded. Thus, timeliness is an essential requirement which should be supported by the platform on which the control system is implemented.

Many researchers try to model random time delays and packet dropouts in order to incorporate them as a random process in the design of the controller. For example, Yu and Shi (2008) model the network-induced random time delays and packet dropout existing in sensor-to-controller (S-C) and controller-to-actuator (C-A) links by two Markov chains. Xiao et al. (2000) propose the state feedback controller and output feedback controller design methods for a type of NCSs which can be modeled as discrete time jump linear systems with the transition jumps. In this paper, we are taking a different approach to solve this problem. Our idea is based on dynamic scheduling of the communication messages in order to ensure that messages are sent before a specified deadline. With distributed online dynamic scheduling, all nodes are competing with each other in order to win the arbitration rounds on the bus. Nodes change the priority of messages dynamically with time whenever the deadline is approaching until the message is sent. Thus, by insuring that messages will be sent before the timing constraint, one can assume a fixed latency for the network while designing control algorithms. This simplifies the design process of NECS.

Controller Area Network (CAN) is one of the widely used communication protocol especially when having hard real-time communication requirement. CAN bus fixed priority scheduling techniques, like Deadline Monotonic proposed in Leung and Whitehead (1982) or Rate Monotonic (DM, RM) originally developed in Leung and Whitehead (1982) and Liu and Layland (1973), might have minimal bus utilization. To solve this problem, other scheduling techniques should be addressed. On the other hand, more efficient scheduling techniques directly impact CPU load. In this paper, we address the implementation of dynamic scheduling technique in order to meet the timeliness requirement since many control systems are time-critical systems, with the right action being required to be executed at the right time. The effect of the proposed design is then emphasized using real-time hardware-in-the-loop simulation carried out over a physical CAN network to control an automotive active suspension system.

The rest of this paper is organized as follows. A background of dynamic scheduling algorithms along with details about one important scheduling algorithm called Earliest Deadline First (EDF) and its application to CAN networks are given in Section 2. Section 3 details the hardware implementation of the EDF scheduler. Then, real-time experiments carried out to show the effect of the proposed scheduler are given in Section 4. Finally, Section 5 concludes this work.

2. DYNAMIC SCHEDULING

Dynamic priority scheduling is a type of scheduling algorithms in which the priorities are calculated during the execution of the system. This section overviews how to apply this concept to CAN messages and details one famous dynamic scheduling algorithm called Earliest Deadline First (EDF).

2.1 Dynamic Scheduling of CAN Messages

Scheduling messages on the CAN corresponds to assigning identifiers (IDs) to messages according to their priorities as shown in Fig. 1, where the CAN message ID is divided into two parts; the first one is assigned according to the dynamic priority of the message while the other identifies the message itself. Number of bits assigned to the dynamic priority portion is denoted as n_p.

CAN bus uses priority based arbitration where CAN frames with lower IDs transmit first on bus. Thus by decreasing the dynamic portion of the ID, a node has a higher tendency to arbiter the bus and send the required information. Fig. 2 shows a typical example of two nodes sending frames in the same time slot. The first 2 bits of the ID field goes on bus with no collisions, as they are identical in both frames. Both node A and node B read back a correct value of the bit they just transmitted. When it comes to the third bit, node A will read a dominant value although it was sending a recessive value. Hence, node A realize that the bus is busy with a higher priority frame, cancel its transmission, and retries again in the next arbitration round. On the other hand, node B will detect a correct dominant value and will continue transmitting its frame. It is interesting here to notice that node B is never aware of node A frame or transmission. That powerful technique used over CAN bus surely strengthens the robustness of transmission and also preserve bandwidth as there are no incomplete transmissions on bus during collision.

2.2 Earliest Deadline First (EDF) Scheduling

The Earliest deadline scheduling (EDF) assigns higher priority to the message having the earliest absolute deadline. The algorithm implements a logarithmic relative deadline

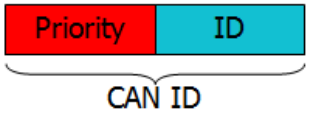


Fig. 1. Dynamic CAN ID.

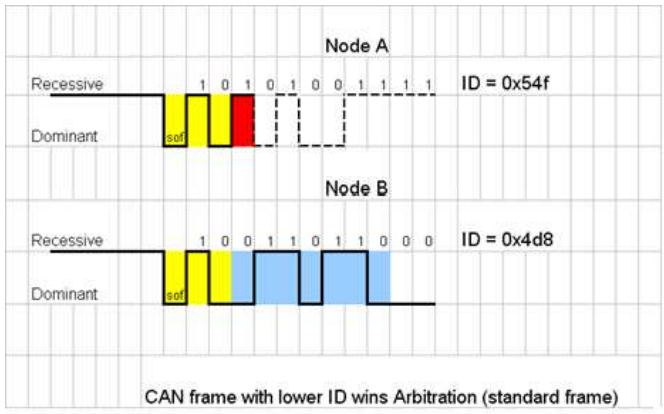


Fig. 2. Arbitration example of two CAN nodes

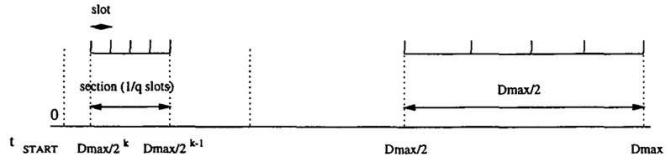


Fig. 3. Distribution of the time units in the time axis

encoding, as a wide range of deadlines can be encoded using logarithmic scale. It implies the need for CAN ID update at the start of each bus-arbitration round [Meschi et al. (1996); Natale (2000); Fuster et al. (2005)].

From scheduling point of view, a scheduling algorithm has to divide the overall time into various time sections and the scheduling algorithm objective is to allocate each of the available messages into these time sections. For most applications, a maximum number of 8 to 16 time sections of exponentially increasing length should be considered. Given a number of message deadline D_i; the largest time section must be able to contain the largest relative deadline among all messages $D_{max} = \max(D_i)$. If the number n_p of priority bits allows more than 16 priority levels, then it is convenient to divide each time section into q units (of the same size). Since we have n_p priority bits, k time sections, and 2^{n_p} available priority levels, they are related as shown in (1). Fig. 3 shows the distribution of time units along the time axis [Natale (2000)].

$$q * k = 2^{n_p} \tag{1}$$

In order to calculate the section index for a specific relative deadline d_{rel}, first a parameter h will be calculated as defined in equation (2) then the priority p representing slot index of the section containing the relative deadline to be encoded has to be calculated as defined in (3). In the equations below, d_i denotes the absolute deadline of the CAN frame.

$$h = \begin{cases} k & \text{if } d_{rel} = D_{max}, \\ k + \left\lfloor \log_2 \dfrac{d_{rel}}{D_{max}} \right\rfloor & \text{if } \dfrac{D_{max}}{2^k} < d_{rel} < D_{max}, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

$$p = h * q + \left\lfloor \frac{d_i - t_{start} - \frac{D_{max}}{2^{k-h}}}{\frac{D_{max}}{2^{k-h}}} \right\rfloor \quad (3)$$

The calculated priority p is then assigned to the dynamic portion of the CAN ID. Fig. 4 shows a detailed priority update algorithm flowchart.

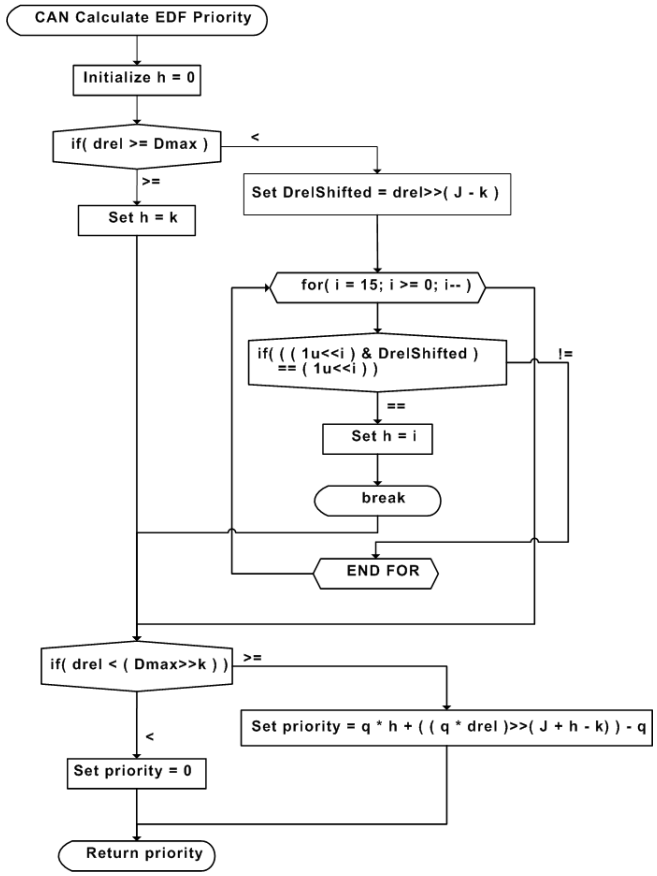


Fig. 4. EDF algorithm flowchart.

3. HARDWARE IMPLEMENTATION

By using distributed EDF scheduling, each node on the network should compete (by updating CAN ID) with other nodes in order to ensure that messages are sent before the specified deadline. This process requires intensive computations to be done when a node loses bus arbitration. Different software implementations are provided by authors' previous work [Shokry et al. (2009)]. Software profiling reveals a serious problem. Periodic scheduler triggering doubles the CPU usage with respect to loss of arbitration triggering. Loss of arbitration EDF triggering runs twice within the maximum frame time period. As the CPU usage increases with the baud-rate, the CAN bus highest baud rate (1Mbps) cannot be achieved. Even if it will fit into the 100% CPU usage, no room would be available for any other application. Therfore, hardware EDF scheduler provides a feasible solution for this problem.

The motivation of integrating EDF scheduler IP with the CAN controller is to save processing cycles for complicated control algorithm applications. It also results in timely accurate sensor/actuator signals transmission and reception.

Fig. 5 shows the EDF algorithm hardware IP block connected to the CAN FPGA SoC blocks. The EDF algorithm calculation block is mainly connected to the register file block. Both Tx_state signal, delayed version Tx_state_q, enable the EDF algorithm calculation block to detect the transmission starting point. Upon tx-signal positive edge, the scheduler algorithm is triggered. Tx_point represents the bit instance at which the controller starts transmission. EDF priority calculation takes only one clock cycle. Integrating the EDF IP leads to adding two 8-bit registers di_1 and di_2 to represent the low-byte and high-byte respectively of the frame time to deadline (relative deadline). The value of this relative deadline starts to decrease just after being written completely in the CAN controller registers di_1 and di_2.

Our hardware implementation assumed that the deadline is written in ms unit. The algorithm configuration parameters values are as follows:

- No. of priority bits = 6bits
- No. of bits for message identification = 5bits
- q = 8
- k = 8
- Dmax = 1024 ms

The proposed EDF algorithm implementation performs all calculations by hardware; no CPU overhead is introduced by the algorithm calculations. The execution time of the logarithmic EDF calculation algorithm takes only one clock-cycle.

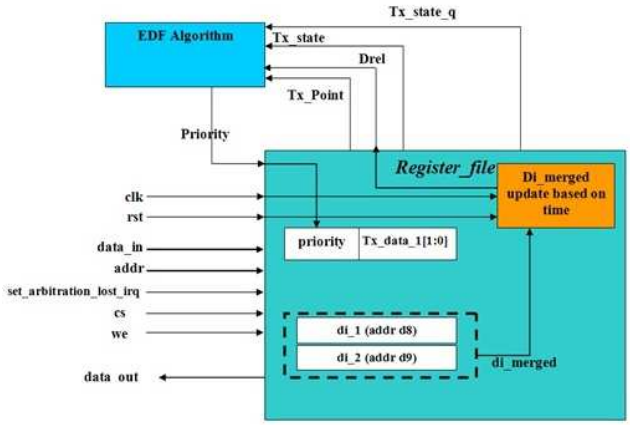


Fig. 5. CAN IP internal hardware updates.

4. CASE STUDY AND RESULTS

This section presents the case study of applying the proposed EDF scheduler inside an automotive active suspension control loop. The details of the testing environment, process and controller dynamics along with the effect of dynamic scheduling on the quality of the control-loop are all presented in this section.

4.1 Environment Setup

The EDF logarithmic algorithm have been implemented on a Spartan-3E FPGA board. The implemented SoC utilizes a Microblaze core along with a CAN IP augmented

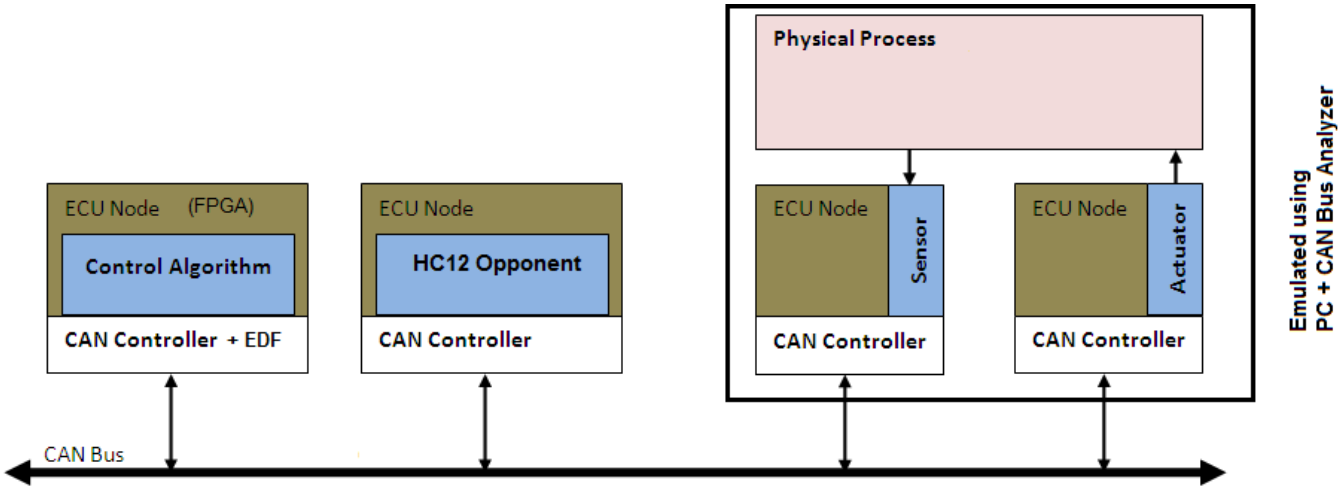


Fig. 6. Experimental environment used for testing the proposed EDF scheduler.

with the hardware EDF Algorithm component. Fig. 6 is showing the overall testing setup that has been used for testing the EDF implementation. It consists of a CAN bus connecting the FPGA board, the CAN bus analyzer used (Tellus [Mentor Graphics Corporation (2002)]) and an evaluation board opponent with an HC12 microcontroller. The autmotive active suspension system is simulated in real-time on a PC. Emulated sensor measurments are sent on the CAN bus using the CAN bus analyzer tool. Control algorithm implemented on the controller node (FPGA) utilizes this signal sent on the CAN bus in order to compute the next control signal which is sent also using the same CAN bus. This setup is actually needed to be able to test the performance of the algorithm by having another node (HC12) trying to send its messages at the same time as the node under test. HC12 node can be configured to achieve up to 98% CAN bus utilization leading to delay CAN frames holding sensor and control signals.

4.2 Automotive Active Suspension System

Automotive active suspension systems utilize electro hydraulic actuators in order to reject the vertical movements coming from deficiencies in road profiles. The wheel is connected to the unsprung mass through a spring/damper pair. The hydraulic actuator force F_A is used to control the sprung mass (chassis) vibrations x_s. A pure damping element exists parallel with the hydraulic actuator in order to remove shock vibrations. Active suspension systems have natural nonlinear behavior in addition to different high frequency response to actuator force. Analyzing this system, one can observe that the output is affected by two inputs which are the voltage input command v to the actuator and the road profile $\dot{r}$. It is suitable to represent our system as shown in Fig. 7 where $r - x_s$ is the system output, H_r and H_v are transfer functions describing dynamics between inputs and the output.

We used the quarter car model detailed in Donahue (2001). Applying system identification techniques on this automotive active suspension system is presented in Shoukry et al. (2010b). The obtained transfer functions are given in (4) and (5).

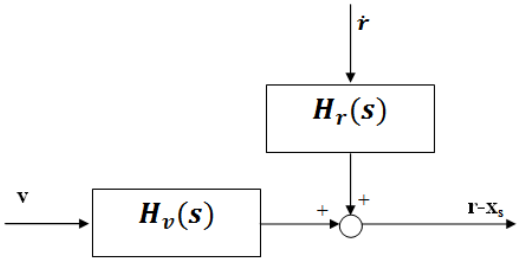


Fig. 7. System model of the automotive active suspension system.

$$H_v(q^{-1}) = \frac{H_{v1}(q^{-1})}{H_{v2}(q^{-1})} \tag{4}$$

where:

$$H_{v1}(q^{-1}) = 1 - 3.9704q^{-1} + 6.4992q^{-2} - 5.5351q^{-3} + 2.4597q^{-4} - 0.4532q^{-5}$$
$$H_{v2}(q^{-1}) = 0.0002032q^{-1}$$

$$H_r(q^{-1}) = \frac{H_{r1}(q^{-1})}{H_{r2}(q^{-1})} \tag{5}$$

where:

$$H_{r1}(q^{-1}) = 0.008862q^{-1} - 0.02498q^{-2} + 0.02143q^{-3} - 0.005318q^{-4}$$
$$H_{r2}(q^{-1}) = 1 - 3.234q^{-1} + 4.094q^{-2} - 2.466q^{-3} + 0.6097q^{-4}$$

4.3 Results

We applied Generalized Predictive Controller (GPC) [Clarke et al. (1987a), Clarke et al. (1987b)] on automotive active suspension systems in a previous work [Shoukry et al. (2010a), Shoukry et al. (2010b)]. Controller is designed while assuming a fixed unit-sample delay in the network. The resulting controller filters are given in the R-S-T canonical form as:

$$R(q^{-1}) = 3493.5392q^{-1} - 13784.8582q^{-2} + 22449.0147q^{-3} - 19032.0021q^{-4} + 8419.1817q^{-5} - 1543.9133q^{-6}$$
$$S(q^{-1}) = 1 - 1.9391q^{-1} + 1.1235q^{-2} - 0.1668q^{-3} - 0.0176q^{-4}$$
$$T(q^{-1}) = 0$$

Experiments carried out, using real-time CAN-based environments, aimed at finding appropriate GPC tuning parameters, show high sensitivity of the process against packets time-varying networked-induced delays. Fig. 8 shows the control signal for certain controller tuning parameter for an ideal network where no other opponents are connected to the network. Accordingly, no packet lost or delays in the network occur. We show three results for each experiment, the first one show the time response of the system due to changes in a road profile. The second one is the control signal from the designed GPC controller. The third response shows the quality of the controller, where the FFT of the time response is plotted against the ride comfort zone

Fig. 9 shows the response of the same system when an opponent is connected to the network while the system is not using any dynamic scheduling algorithm. The opponent node is transmitting frames perioidically each 1 msec in order to achieve a 98% overload on the network. Accordingly, the feedback loop suffers from sensor and actuator packets delay on the network. The response shown in Fig. 9 shows obvious degradation in the time response and on the quality of the control loop since many component of the FFT is now above the ride comfort specified by the red line.

Fig. 10 and Fig. 11 show the same previous scenario but this time the control node is using dynamic scheduling only for half of the experiment time, i.e., the experiment shown in Fig. 10 utilizes the EDF dynamic scheduling algorithm starting from time 12.5 sec while the experiment shown in Fig. 11 utilizes the EDF algorithm for the first half of the experiment starting from time 0 sec till 12.5 secs. These two experiments show that dynamic scheduling is able to overcome network delays. Due to the logarithmic nature of the algorithm, the EDF algorithm is able to modify CAN IDs rapidly and increases the priority of the sensor and actuator nodes so that they meets the timing constraints imposed on this signal (which is a unit-sample delay).

5. CONCLUSION

Controlling complicated time-critical systems (like automotive active suspension system) dictates a good consideration for the data loss problem. Timely sensor and actuator signals highly affect digital closed loop control stability and performance. So, scheduler design should be considered in real-time digital control implementation. This paper presented the effect of using online distributed EDF scheduler in order to achieve timely sensor/actuator signals. Experiments carried out show that online distributed EDF scheduling is an effective way to confront network problems. Experiemnts also show that by using distributed dynamic scheduling, one can assume always a fixed network delay while designing control loops.

REFERENCES

Clarke, D.W., Mohtadi, C., and Tuffs, P.C. (1987a). Generalized predictive control-part 1: The basic algorithm. *Automatica*, 23(2), 137–148.

Clarke, D.W., Mohtadi, C., and Tuffs, P.C. (1987b). Generalized predictive control-part 2: The basic algorithm. *Automatica,*, 23(2), 149–160.

Donahue, M.D. (2001). *Implementation of an Active Suspension, Preview Controller for Improved Ride Comfort.* Master's thesis, The Universirty of California at Berkeley, Berkely, California, USA.

Fuster, S., Rodriguez, F., and Bonastre, A. (2005). Software-based EDF message scheduling on CAN networks. In *Proceedings of Second International Conference on Embedded Software and System (ICESS05).* Xian, China.

Leung, J.Y.T. and Whitehead, J. (1982). On the complexity of fixed-priority scheduling of periodic, real-time tasks. *Performance Evaluation*, 2 (4), 237–250.

Liu, C.L. and Layland, J.W. (1973). Scheduling algorithms for multiprogramming in a hard real-time environment. *Journal of the ACM*, 20 (1), 46–61.

Mentor Graphics Corporation (2002). *Volcano TELLUS.*

Meschi, A., Natale, M.D., and Spuri, M. (1996). Earliest deadline message scheduling with limited priority inversion. In *Proceedings of the 4th International Workshop on Parallel and Distributed Real-Time Systems*, 87–94.

Natale, M.D. (2000). Scheduling the CAN bus with earliest deadline techniques. In *Proceedings of the 21st IEEE Real-Time Systems Symposium*, 259–268.

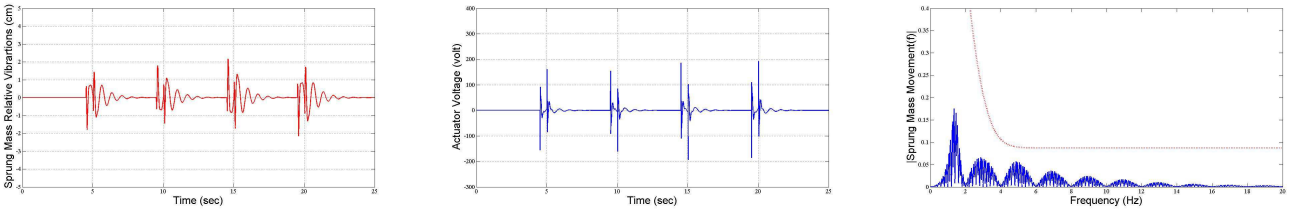
Shokry, H., Shedeed, M., Hammad, S., Shalan, M., and Wahdan, A. (2009). Hardware EDF scheduler implementation on controller area network controller. In *Proceedings of 4th International IEEE Design and Test Workshop (IDT) 2009.* Riyadh, Saudi Arabia.

Shoukry, Y., El-Kharashi, M.W., El-Shafey, M., and Hammad, S. (2010a). Towards real-time networked embedded generalized predictive control for automotive active suspension system. In *Proceedings of IFAC symposium on Advances in Automotive Technology (IFAC AAC) 2010.* Munich, Germany.

Shoukry, Y., El-Shafey, M., and Hammad, S. (2010b). Networked embedded generalized predictive controller for active suspension system. In *Proceedings of American Control Conference (ACC) 2010*, 4510–4575. Baltimore, Maryland, USA.

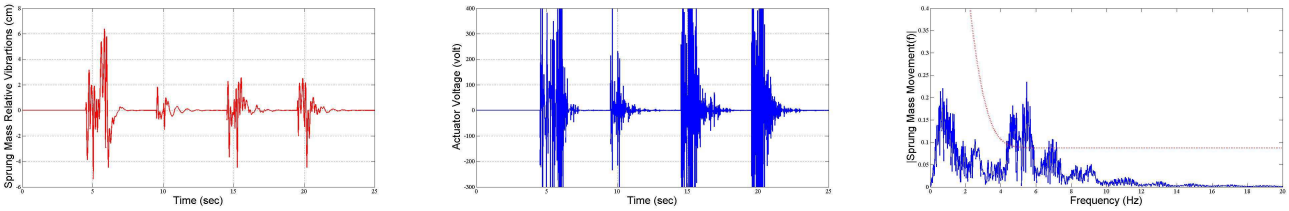
Xiao, L., A. L., H., and How, J.P. (2000). Control with random communication delays via a discrete-time jump system approach. In *Proceedings of American Control Conference*, 2199–2204.

Yu, B. and Shi, Y. (2008). State feedback stabilization of networked control systems with random time delays and packet dropout. In *Proceedings of ASME 2008 Dynamic Systems and Control Conference.* Ann Arbor, Michigan, USA.



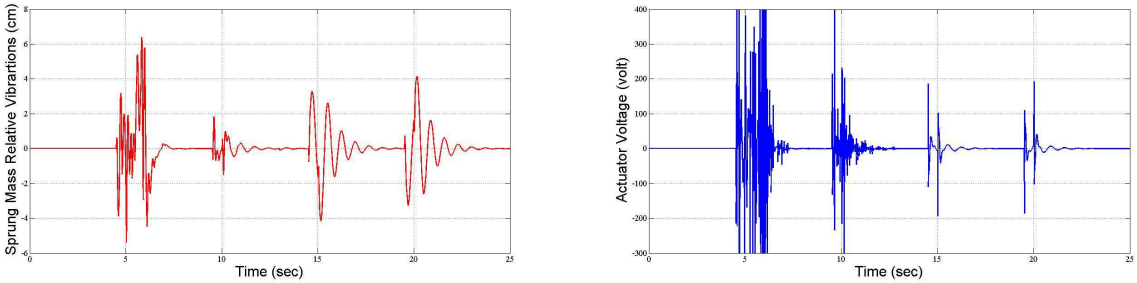
(a) Time response (b) Control signal (c) Perfromance index

Fig. 8. Results of the control-loop with a 0% loaded network.



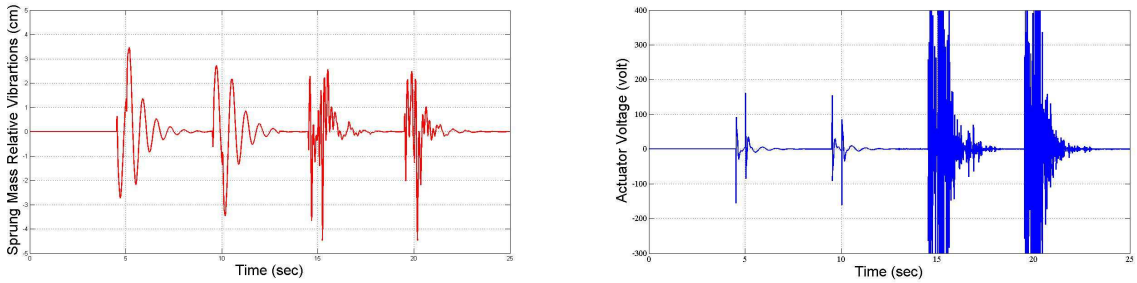
(a) Time response (b) Control signal (c) Perfromance index

Fig. 9. Results of the control-loop with a 98% loaded network with no EDF dynamic scheduling.



(a) Time response (b) Control signal

Fig. 10. Results of the control-loop with a 98% loaded network, EDF used starting from 12.5 secs.



(a) Time response (b) Control signal

Fig. 11. Results of the control-loop with a 98% loaded network, EDF used starting till 12.5 secs.