ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Heuristic Approach for Scheduling in Flex Ray Protocol Based Communication

Jarugumallu Madhanmohan
M.Tech (VLSI System Design),
TRR Institute of Technology & Sciences, Hyd, India.

G.Devendhar
Assistant Professor,
Dept of ECE,
TRR Institute of Technology & Sciences, Hyd, India.

ABSTRACT:

FlexRay is a communication protocol heavily promoted on the market by large group of car manufactures and automotive electronic suppliers. This protocol was developed by the FlexRay consortium which was started when BMW and DaimlerChrysler work together to create a new network. However before it can be successfully used for safety critical applications that require predictability, timing analysis is necessary for providing bounds for message communication times. The heuristic techniques are used to optimize the path of FlexRay communication protocol while transfer the data from one node to another.Different techniques are used to give information about scheduling and optimization results but having limited outcome and boundaries. Here heuristic techniques are used to optimize the path of FlexRay communication protocol.The analysis can be carried out by Heuristic optimization method, wind driven optimization (WDO) which comprises of both Local and Global scheduling approaches. These local and global scheduling is based on Interpolation Techniques. The optimization of path with timing analysis results is shown in MATLAB using GUI.

Keywords:

Communication protocols, heuristic optimization, local and global scheduling methods, interpolation.

I.INTRODUCTION:

Many safety-critical applications, following physical, modularity or safety constraints, are implemented using distributed architectures composed of several different types of hardware units (called nodes), interconnected in a network. For such systems, the communication between functions implemented on different nodes has an important impact on the overall system properties, such as performance, cost and maintainability. There are several communication protocols for realtime networks. Among the protocols that have been proposed for in-vehicle communication, only the Controller Area Network (CAN) [4], the Local Interconnection Network (LIN) [17], and SAE's J1850 [30] are currently in use on a large scale [20]. Moreover, only a few of the proposed protocols are suitable for safety-critical applications where predictability is mandatory [29]. Communication activities can be triggered either dynamically, in response to an event (event-driven), or statically, at predetermined moments in time (time-driven). Therefore, on one hand, there are protocols that schedule the messages statically based on the progression of time, such as the SAFEbus [13], SPIDER [19], TTCAN [14], and Time-Triggered Protocol (TTP) [16]. The main drawback of such protocols is their lack of flexibility. On the other hand, there are communication protocols where message scheduling is performed dynamically, such as Byteflight [3] introduced by BMW for automotive applications, CAN [4], LonWorks [9] and Profibus [28]. A large consortium of automotive manufacturers and suppliers has recently proposed a hybrid type of protocol, namely the FlexRay communication protocol [11]. FlexRay allows the sharing of the bus among event-driven (ET) and time-driven (TT) messages, thus offering the advantages of both worlds.

FlexRay will very likely become the de-facto standard for in-vehicle communications.1 However, before it can be successfully deployed in applications that require predictability, timing analysis techniques are necessary to provide bounds for the message communication times [20]. FlexRay is composed of static (ST) and dynamic (DYN) segments, which are arranged to form a bus cycle that is repeated periodically. The ST segment is similar to TTP, and employs a generalized time-division multiple-access (GTDMA) scheme. The DYN segment of the FlexRay protocol is similar to Byteflight and uses a flexible TDMA (FTDMA) bus access scheme. Although researchers have proposed analysis techniques for dynamic protocols such as CAN [32], TDMA [33], ATM [10], Token Ring protocol [31], FDDI protocol [1] and TTP [24], none of these analyses is applicable to the DYN segment in FlexRay.

In [7], the authors consider the case of a hard real-time application implemented on a FlexRay bus. However, in their discussion they restrict themselves exclusively to the static segment, which means that, in fact, only the classical problem of communication scheduling over a TDMA bus [24, 12] is considered. The performance analysis of the Byteflight protocol, which is similar to the DYN segment of FlexRay, is analyzed in [5]. The authors assume a very restrictive "quasi-TDMA" transmission scheme for time-critical messages, which basically means that the DYN segment would behave as an ST (TDMA) segment in order to guarantee timeliness. In this paper we present the first approach to timing analysis of applications communicating over a FlexRay bus, taking into consideration the specific aspects of this protocol, including the DYN segment.

More exactly, we propose techniques for determining the timing properties of messages transmitted in the static and the dynamic segments of a FlexRay communication cycle. We first briefly present a static cyclic scheduling technique for TT messages transmitted in the ST segment, which extends our previous work on the TTP [23]. Then, we develop a worstcase response time analysis for ET messages sent using the DYN segment, thus providing predictability for messages transmitted in this segment. The analysis techniques for messages are integrated in the context of a holistic schedulability analysis algorithm that computes the worst-case response times of all the tasks and messages in the system. This paper is organized in eight sections. Section 2 presents the system architecture considered, and Section 3 introduces the FlexRay media access control.

II.SYSTEM MODEL:

We consider architectures consisting of nodes connected by one FlexRay communication channel1 (see Figure 1.a). Each processing node connected to a FlexRay bus is composed of two main components: a CPU and a communication controller (see Figure 2.a) that are interconnected through a two-way controller-host interface (CHI). The controller runs independently of the node's CPU and implements the FlexRay protocol services. For the systems we are studying, we have designed a software architecture which runs on the CPU of each node. The main component of the software architecture is a realtime kernel that contains two schedulers2, for static cyclic scheduling (SCS) and fixed priority scheduling (FPS), respectively.

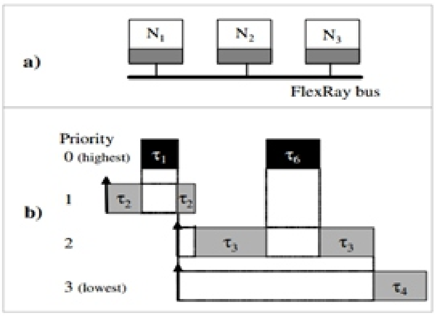


Figure 1: System Architecture Example

When several tasks are ready on a node, the task with the highest priority is activated, and preempts the other tasks. Let us consider the example in Figure 1.b, where we have six tasks sharing the same node. Tasks τ1 and τ6 are scheduled using SCS, while the rest are scheduled with FPS. The priorities of the FPS tasks are indicated in the figure. The arrival time of a task is depicted with an upwards pointing arrow. Under these assumptions, Figure 1.b presents the worst-case response times of each task. SCS tasks are non-preempt able and their start time is offline fixed in the schedule table (they also have the highest priority, denoted with priority level "0" in the figure). FPS tasks can only be executed in the slack of the SCS schedule table. FPS tasks are scheduled based on priorities. Thus, a higher priority task such as τ3 preempts a lower priority task such as τ4. SCS activities are triggered based on a local clock in each processing node. The synchronization of local clocks throughout the system is provided by the communication protocol [11].

III.FLEXRAY COMMUNICATION PROTOCOL:

In this section we will describe how messages generated by the CPU reach the communication controller and how they are transmitted on the bus. Let us consider the example in Figure 2 where we have three nodes, N1 to N3 sending messages ma, mb,... mh using a FlexRay bus. In FlexRay, the communication takes place in periodic cycles (Figure 2.b depicts two cycles of length Tbus). Each cycle contains two time intervals with different bus access policies: an ST segment and a DYN segment3. The ST and DYN segment lengths can differ, but are fixed over the cycles. We denote with STbus and DYNbus the length of these segments. Both the ST and DYN segments are composed of several slots. In the ST segment, the slots number is fixed, and the slots have constant and equal length, regardless of whether ST messages are sent or not over the bus in that cycle.

The length of an ST slot is specified by the FlexRay global configuration parameter gdStaticSlot [11]. In Figure 2 there are three static slots for the ST segment. The length of the DYN segment is specified in number of "minislots", and is equal to gNumberOfMinislots. Thus, during the DYN segment, if no message is to be sent during a certain slot, then that slot will have a very small length (equal to the length gdMinislot of a so called minislot), otherwise the DYN slot will have a length equal with the number of minislots needed for transmitting the whole message [11]. This can be seen in Figure 2.b, where DYN slot 2 has 3 minislots (4, 5, and 6) in the first bus cycle, when message me is transmitted, and one minislot (denoted with "MS" and corresponding to the minislot counter 2) in the second bus cycle when no message is sent. During any slot (ST or DYN), only one node is allowed to send on the bus, and that is the node which holds the message with the frame identifier (FrameID) equal to the current value of the slot counter. There are two slot counters, corresponding to the ST and DYN segments, respectively.

The assignment of frame identifiers to nodes is static and decided offline, during the design phase. Each node that sends messages has one or more ST and/or DYN slots associated to it. The bus conflicts are solved by allocating offline one slot to at most one node, thus making it impossible for two nodes to send during the same ST or DYN slot. In Figure 2, node N1 has been allocated ST slot 2 and DYN slot 3, N2 transmits through ST slots 1 and 3 and DYN slots 2 and 4, while node N3 has DYN slots 1 and 5. For each of these slots, the CHI reserves a buffer that can be written by the CPU and read by the communication controller (these buffers are read by the communication controller at the beginning of each bus cycle, in order to prepare the transmission of frames) The associated buffers in the CHI are depicted in Figure 2.a.

We denote with the number of dynamic slots associated to a node Np (this means that for N2 in Figure 2, = 2). We use different approaches for ST and DYN messages to decide which messages are transmitted during the allocated slots. For ST messages, we consider that the CPU in each node holds a schedule table with the transmission times. When the time comes for an ST message to be transmitted, the CPU will place that message in its associated ST buffer of the CHI. For example, ST message mb sent from node N1 has an entry "2/2" in the schedule table specifying that it should be sent in the second slot of the second ST cycle. For the DYN messages, the designer specifies their FrameID.

For example, DYN message me has the frame identifier "2". We assume that there can be several messages sharing the same DYN FrameID1.For example, messages mg and mf have both Frame ID 4. If two messages with the same frame identifier are ready to be sent in the same bus cycle, a priority scheme is used to decide which message will be sent first. Each DYN message mi has associated a priority prioritymi . Messages with the same FrameID will be placed in an output queue ordered based on their priorities. The message from the head of the priority queue is sent in the current bus cycle. For example, message mf will be sent before mg because it has a higher priority. At the beginning of each communication cycle, the communication controller of a node resets the slot and minislot counters. At the beginning of each communication slot, the controller verifies if there are messages ready for transmission (present in the CHI send buffers) and packs them into frames2.

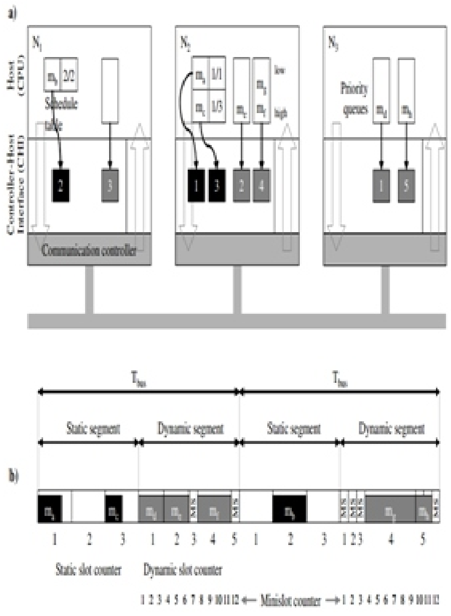


Figure 2: FlexRay Communication Cycle Example

In the example in Figure 2 we assume that all messages are ready for transmission before the first bus cycle. Messages selected and packed into ST frames will be transmitted during the bus cycle that is about to start according to the schedule table. For example, in Figure 2, messages ma and mc are placed into the associated ST buffers in the CHI in order to be transmitted in the first bus cycle. However, messages selected and packed into DYN frames will be transmitted during the DYN segment of the bus cycle only if there is enough time until the end of the DYN segment.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Such a situation is verified by comparing if, in the moment the DYN slot counter reaches the value of the FrameID for that message, the value of the minislot counter is smaller than a given value pLatestTx. The value pLatestTx is fixed for each node during the design phase, depending on the size of the largest DYN frame that node will have to send during run-time. For example, in Figure 2, message mh is ready for transmission before the first bus cycle starts, but, after message mf is transmitted, there is not enough room left in the DYN segment. This will delay the transmission of mh for the next bus cycle.

IV.PHYSICAL LAYER:

The FlexRay physical layer shall be discussed under the following headings:

- Network Topologies
- Transmission Medium
- Signal Levels and Bit Representation
- Bit Coding and Decoding
- Synchronization

Network Topologies:

As previously discussed there are several options for the layout of a FlexRay network. It can be configured as a single-channel or dual-channel bus network, a single-channel or dual-channel star network, or in various combinations of bus and star topologies. A FlexRay network consists of a maximum of two channels, Channel A and Channel B. Each node on the network can be connected to either or both of these channels. This flexibility in configuration may be used to increase bandwidth and/or introduce redundancy in to the system to increase its level of fault tolerance.

Transmission Medium:

The FlexRay protocol specification does not define cable types to be used, but does stipulate their electrical specifications. The medium in use for FlexRay busses may be shielded or unshielded cables, as long as they provide the following characteristics: Impedance of 80- 110Ω at a frequency of 10MHz, maximum line delay of 10ns/m and a maximum cable attenuation of 82dB/km at a frequency of 5MHz. These cable requirements are similar to that of CAN. A twisted-wire pair is generally used, as it helps to prevent electromagnetic interference from other electrical devices in the vicinity affecting the network.

Each channel uses two wires to connect to the bus, labeled BP (Bus Plus) and BM (Bus Minus). The cables need to be terminated at each node, and at either end of a bus. This is achieved by connecting a termination resistor, RT, in the region of 100Ω between the BP and BM wires. This prevents the signal being reflected back through the bus once it reaches the end of the system.

Signal Levels and Bit Representation:

The bus communicates using two signals BP and BM. The differential voltage between the signals, Vdiff, is used to represent the four different states which can occur on the bus: Idle LP, Idle, Data 1, Data 0. The various states and their voltages (measured to ground) are shown in Figure 2.13. The differential voltage on the bus is designed as follows:

Vdiff = VBM − VBL

• When the bus is in Idle LP (Low Power) there is no current being driven to either BP or BM and the bus driver biases both to ground

• When the bus is in Idle there is also no current being driven to BP or BM, but as we can see from Figure 2.13 however, the connected nodes bias both BP and BM to 2.5 volts

• To drive the bus to Data 1, the bus driver increases the voltage on BP by 600mV and decreases BM by 600mV. This gives us a differential voltage of 1.2V. Data 1 represents a logical HIGH

• To drive the bus to Data 0, the bus driver decreases the voltage on BP by 600mV and increases BM by 600mV. This gives us a differential voltage of -1.2V. Data 0 represents a logical LOW.

Bit Coding and Decoding:

The decoding process samples the incoming data at eight times the rate of the bit clock. These samples are forwarded to a majority voting process, which analyses the last five samples received. If at least three samples are HIGH the process outputs a value of HIGH for that bit, otherwise it outputs a value of LOW. This voting process is used to suppress glitches in the received signal, provided that the duration of the glitch is less than three samples].

Synchronization:

FlexRay is a time triggered networking system. Media access is time controlled and unlike CAN there is no collision detection or resolution mechanism in case of collisions but instead a mechanism for prevention of collisions. All nodes must be synchronized for successful and accurate communication. The clocks of the communization controllers in the network, however, can be influenced by temperature and voltage fluctuations, or production tolerances of the oscillator. This leads to differing internal time bases. To offset this, the FlexRay protocol uses a distributed clock synchronization mechanism, i.e. there is no single physical reference clock. Instead, each node individually synchronizes itself to the network by observing the timing of transmitted synchronization frames from other nodes. From this a virtual reference clock is established using a distributed fault-tolerant clock synchronization algorithm. The deviation to this reference clock is then periodically measured in regard to phase and frequency deviation in order to ensure offset and rate correction respectively. If necessary, the clock is adjusted accordingly.

V.DATA LINK LAYER:

The FlexRay data link shall be discussed under the following headings:
• Message Framing
• Communication Cycle

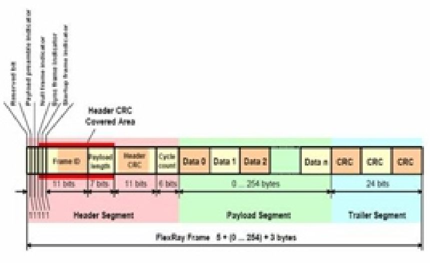


Fig 3: FlexRay Frame Format

VI.RESULTS AND ANALYSIS:

In the aeronautical or cruise control applications we can us FlexRay protocol. The currently used protocols in most of the application are event-triggered. This means that all activity is invoked by the occurrence of an event. We can measure the speed up to 10 Mb/s for example a sensor that senses a changing value immediately sends an event message to the controller of the sensor. A system designed for transmission of event messages requires a dynamic scheduling strategy because the time of an invocation of a task can't be predicted. FlexRay is having time triggered as well as an event triggered communication channel.To demonstrate the effectiveness of Optimization in scheduling the task to the processors/nodes we have used Heuristic approach called Wind Driven Optimization technique. The technique is based on flight is traveling from one place to another place for which the required time is reduced by applying the optimization heuristic method in turn having options like global and local scheduling. In those linear and spline interpolation techniques are investigated. We assume the Flexray Communication protocol of Average bandwidth of 1000 kbps and the nodes having processors executing in different times for an approximate 10 meters distance message transfer.The linear interpolation optimizations table, chart, simulation results is shown below. Due to this the average Bandwidth is increased from 811.489Kbps to 817.928Kbps, the required time is reduced from 44364sec to 44178sec in local and global optimization techniques respectively.

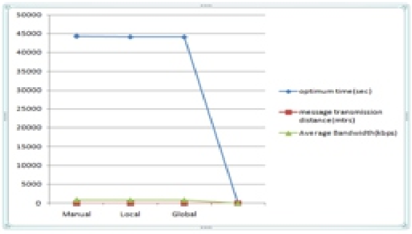


Figure 4: linear interpolation graphical chart

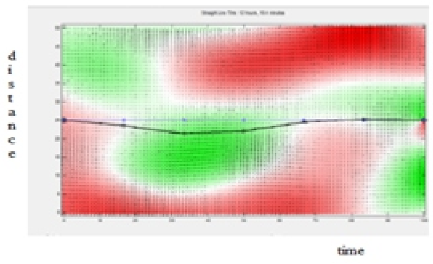


Figure 5: linear local optimization graph

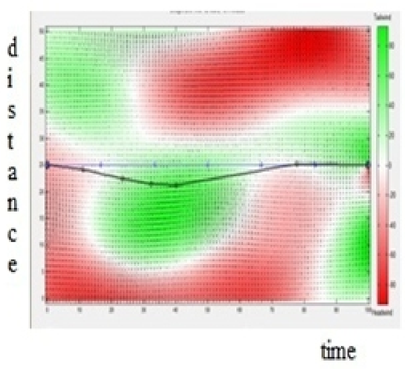


Figure 6: linear global optimization graph

The spline interpolation optimizations table, chart, simulation results is shown below. Due to this the average Bandwidth is increased from 811.489Kbps to 818.147kbps, the required time is reduced from 44178sec to 44142sec in local and global optimization techniques respectively.

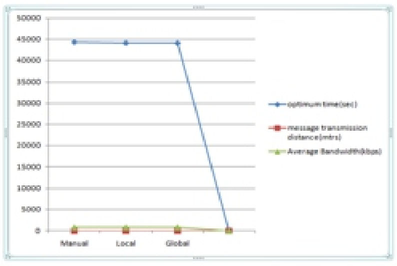
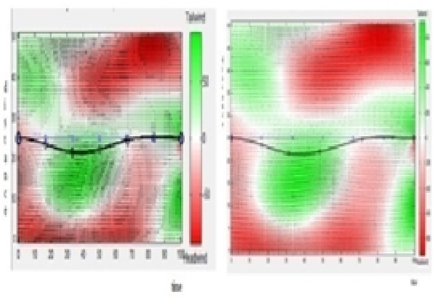


Figure 7: Spline interpolation graphical chart



(a) (b)

Figure 8(a) spline local optimization graph (b) Spline global optimization graph

As the results obtained are related to optimized movements. Similarly it can be adopted for scheduling jobs in static approach which is available in FlexRay protocol. Where in allotment is based on optimal results obtained. The adoption of WDO can enhance the allocation and execution of job.

VII.CONCLUSION:

The heuristic techniques are used to optimize the path of FlexRay communication protocol. The analysis is carried out by heuristic optimization method, wind driven optimization (WDO) which comprises of the linear and spline interpolation techniques of both local and global scheduling approaches. On comparison we found better optimum time and best Bandwidth utilization in message transmission through FlexRay of 10 meters length. The linear and spline interpolations results are approximately same in local and global approaches. However, Global approach in WDO has an edge over Local approach. The adoption of other optimization techniques can be looked into for better results. As extension of this work the other techniques of interpolation like shape preserving spline techniques can be compared for both local and global optimization results. From that we can estimate which technique gives best results.

REFERENCES:

[1]EPN online (2005) Drive-By-Wire: Electronics Vs Mechanical Engineering [online], available: http://www.epnonline. com/page/16409/drive-by-wire--lectronics-vs mechanical-engineering.html [accessed 31 Oct 2007].

[2] N. Nave, Y. Song, F. Simonot-Lion et al., "Trends in automotive communication systems", Proceedings of the IEEE, vol. 93, no. 6, pp.1204-1223, 2005.

[3]FlexRay Protocol Specification, version 2.1, revision A. 2005. http://www.flexray.com.

[4]Anderi Hagiescu , et al, " Performance Analysis of FlexRay based ECU networks ", DAC-2007, pp- 284-289.

[5]FlexRay Consortium (2007) about FlexRay [online],available: http://www.flexray.com/index.php?sid=254188fe2bd59eb7108227f0adea90f5&pid=80&lang=de [accessed 19 Oct 2007].

[6]BMW Manufacturing Co. (2006) THE NEW BMW X5 Perfect Blend of Driving Dynamics, Functionality and Exclusivity [press release], 8 August, available: http://www.bmwusfactory.com/media_center/releases/release.asp?intReleaseNum=209&strYear=2006 [accessed 2 October 2007].

[7]"On Hardware implementation of flex-ray bus guardian module" ,Proc. 12th IXDES JUNE 2007.

[8]R. Saket,and N. Navet,"Frame packing algorithms for automotive applications," Journal of Embedded Computing, vol. 2, pp. 93-102, Sept. 2006.

[9]G. Quan, and X. Hu, "Enhanced Fixed-Priority Scheduling with (m,k)- Firm Guarantee", in Proc. REAL'00, 2000, paper 10.1109 p.79

[10]R. Horst and P.M. Pardalos, editors. Handbook of Global Optimization, volume 1. Kluwer Academic Publishers, Dordrecht, 1995.

[11]R. Horst and Hoang Tuy. Global Optimization: Deterministic Approaches. Springer-Verlag, Berlin, third edition, 1996.

[12]Lam, N. S. 1983. Spatial interpolation methods review. The American Cartographer 10: 129-149.

[13]EPN online (2005) Drive-By-Wire: Electronics Vs Mechanical Engineering [online], available: http://www.epnonline.com/page/16409/drive-by-wire--electronics-vs mechanical-engineering.html [accessed 31 Oct 2007].

[14]FlexRay Consortium. FlexRay communications systems -protocol specification version 2.1 rev. a., 2005. http://www.flexray.com R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[15]National Instruments (NI) Developer Zone."FlexRay automotive communication bus overview" ,http://zone.ni.com/devzone/cda/tut/p/id/3352 Aug 21 2009

[16]T. Pop, P. Pop, P. Eles, Z. Peng, and A. Andrei, "Timing analysis of the FlexRay communication protocol," In Proc.Of the ECRTS '06, pp. 203–216, 2006.

[17]Stefan SCHNEELE, Christoph HELLER (EADS Innovation Works - Germany) Margrate MAWIRE, Haydn THOMPSON (University of Sheffield - Great Britain) Christophe MITAL (TTTech - Austria) Christian THIRY (Zodiac Aerospace - France) "Use of automotive data bus in avionics : CAN and FlexRay" MOET Consortium, 2009.