International Journal of Communication Networks and Security

Volume 1 | Issue 4 Article 16

April 2012

A SIMPLE AUTOMOTIVE APPLICATION USING FLEXRAY™ PROTOCOL

AVINASH K R
Dept of Telecommunication, R.V. College of Engineering, Bangalore, India, avinash@gmail.com

P NAGARAJU
Dept of Telecommunication, R.V. College of Engineering, Bangalore, India, NAGARAJU@gmail.com

SURENDRA S
IFIN VV, Infineon Technologies India Pvt. Ltd, Bangalore, India, SURENDRA@gmail.com

SHIVAPRASAD S
IFIN VV, Infineon Technologies India Pvt. Ltd, Bangalore, India, SHIVAPRASAD@gmail.com

Recommended Citation

R, AVINASH K; NAGARAJU, P; S, SURENDRA; and S, SHIVAPRASAD (2012) "A SIMPLE AUTOMOTIVE APPLICATION USING FLEXRAY™ PROTOCOL," *International Journal of Communication Networks and Security*: Vol. 1 : Iss. 4 , Article 16.
DOI: 10.47893/IJCNS.2012.1055
Available at: https://www.interscience.in/ijcns/vol1/iss4/16

A SIMPLE AUTOMOTIVE APPLICATION USING FLEXRAY™ PROTOCOL

AVINASH K R[1], P NAGARAJU[2], SURENDRA S[3] & SHIVAPRASAD S[4]

[1&2] Dept of Telecommunication, R.V. College of Engineering, Bangalore, India
[3&4] IFIN VV, Infineon Technologies India Pvt. Ltd, Bangalore, India

Abstract - FlexRay™ protocol is emerging as the next generation automotive communication protocol which offers high data rate, deterministic, fault tolerant, flexible in-vehicle data communication. This protocol supports both time triggered and event triggered data communication. The network that uses FlexRay™ protocol is called FlexRay™ network. The need for FlexRay™ protocol is the substantial demand for the high capacity in-vehicle data communication between the electronic components. In this work, we used Infineon SoCs as FlexRay™ nodes and establish communication between multiple nodes using FlexRay™ protocol. A simple automotive application is developed with temperature and magnetic field sensor being connected to a node and the sensor data is being communicated over the FlexRay™ network.

Keywords - *FlexRay™ Protocol; Automotive; Data communication; Multiple nodes; Infineon SoC.*

I. INTRODUCTION

Modern Vehicles are becoming more and more sophisticated, with more functions being controlled by Electronic Control Unit (ECU).As new functions are developed, there is not only demand on ECU, but there is a substantial demand on Communication networks placed in the automobile. There is also growing need for fast and dependable networks for new safety features such as X-by-wire applications. As the implementations of these technologies are getting increased, the number of Electronic Control Units (ECUs) used in automotive systems and signals communicated between them have been increased [1].Hence, this has led to the development of bus-based ECU networks. To mention few of the In-vehicle network protocols that have been developed-Local Interconnect Network (LIN), Controller Area Network (CAN), TTP(Time Triggered Protocol), FlexRay™ Protocol , MOST (Media Oriented System Transport) etc [1].

II. FLEXRAY™ PROTOCOL

FlexRay™ is a scalable, flexible, high-speed, deterministic, error-tolerant communication technology that is designed to meet growing safety related challenges in the automobile industry [3]. Mainly, this technology is concentrated for data communication in very safety critical use areas in automobile [3]. It was originally developed by FlexRay™ Consortium [4].

A FlexRay™ communication system is made up of a number of FlexRay™ nodes and a physical transmission medium (FlexRay™ Bus) interconnecting all of the FlexRay™ nodes. FlexRay™ node is nothing but a Electronic Control Unit (ECU) which is connected to a FlexRay™ bus via a communication controller and one or two bus drivers depending on the number of channels. Communication controller is the electronic component in a node where FlexRay™ protocol is implemented. Basically there are two channels available. System designer can choose between single channel or dual channel configuration [5].

FlexRay™ protocol is a unique time-triggered protocol [6-9], that provides options for deterministic data that arrives in a predefined time slot as well as CAN-like dynamic event-driven data [6-9] to handle a large variety of frames. FlexRay™ accomplishes this hybrid of core static frames and dynamic frames with a communication cycle that provides a predefined space for static and dynamic data [9]. The main physical topologies are bus, star and ring [5]. The hybrid topology of the mentioned basic topologies is also possible.

FlexRay™ protocol uses Time Division Multiple Access or TDMA scheme [5]. Every FlexRay™ node is synchronized to the same clock. Each node writes on the bus when its turn comes. Because the timing is consistent in a TDMA scheme, FlexRay™ is able to guarantee the consistency of data deliver to nodes on the network [3], [8]. Hence this helps the networks which work on up to date information.

The periodically repeating communication cycle is composed of static segment, dynamic segment, symbol window and NIT (network idle time). Within the static segment the TDMA (time division multiple access) method is used. Payload is same for all the frames in this segment. This segment is usually designed for the high priority frames. Static segment is divided into static slots where each node can transmit its frame. Static slot is divided into macroticks and each macrotick is further divided into microticks. The macrotick represents the smallest granularity unit of the global cluster time. With respect to microticks, different nodes can have different duration [9]. Several static slots can be assigned to one node, but each of these static slots may be used just by one node. The dynamic segment is intended for the non-critical messages with variable length. Dynamic segment is divided into minislots which are further divided into macroticks. An access to the bus in this segment is based on the flexible TDMA. In the symbol window, the CAS (Collision Avoidance Symbol), MTS (Media Access Test Symbol) or WUS (Wake Up Symbol) can be sent [10]. The WUS serves only for wakeup of the cluster. The CAS is used by a coldstart node (is the node, which initializes the cluster communication. The initialization of the startup process is called coldstart) to initiate the start-up procedure. During the NIT of communication cycle each node can correct its timing offsets.

III. APPLICATION

The automobiles like cars usually implement a sensor network to display the characteristics of the environment like temperature, rainfall, magnetic field etc. The application being developed considered such a sensor which senses the temperature and magnetic field near it. The nodes at various places within an automobile would be requiring characteristics of the environment for various reasons like automatic closing of window of the car, informing the user of the car when a severe temperature is observed etc. To implement that in this work, the sensor data is being communicated to the coldstart node using FlexRay™ protocol. This is achieved by connecting the sensor to a non-coldstart node and the sensed data being received at the sensor module of the non-coldstart node is transmitted to coldstart node using FlexRay™ protocol. As an extension of this application, one can use the sensor data at a coldstart node to develop a real time scenario of an automobile. Here, even though there is no master slave concept, if we could consider the coldstart nodes as the masters and the non-coldstart nodes as slaves, then communicating the sensor data from the non-coldstart node to the coldstart node exhibits real scenario where in slave updating the data to the master. This is an extended work of – [11]

IV. IMPLEMENTATION

As mentioned in the abstract, we used Infineon SoC as an ECU (Node) which includes FlexRay™ Communication Controller (CC) and Sensor module is as shown in the Fig.1

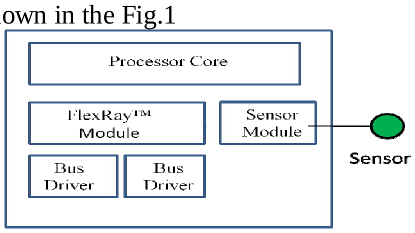

Fig. 1.Electronic Control Unit(Node)

Some of the specifications considered are as follows:

1. Number of nodes in the cluster : 04
2. Data rate :10Mbps
3. Number of coldstart nodes:02
4. Number of channels : 2 (coldstart nodes)
   : 1 (Non-coldstart nodes)
5. Physical Medium : Flexible Ribbon Cable
6. One cycle : 1ms
7. One cycle : 1000 Macroticks
8. One cycle : 40000 Microticks
9. Frame transmission : Static and Dynamic
10. Number of Static slots : 10
11. Static slot duration : 50 Macroticks
12. Number of Minislots : 100
13. Minislot duration : 4 Macroticks
14. Maximum data payload used : 8bytes
15. Number of buffers used : 8; 5 static; 3 dynamic

A. *Scenario considered*

We are considering Node 0 and Node 1 as Coldstart nodes and Node 2 and Node 3 as Non-Coldstart nodes. Node 0 and Node 1 are connected to both the channels- Channel A and Channel B; whereas Node 2 and Node 3 are connected to single channel- Channel A. The sensor is connected to Node 2. This is shown in the Fig.2.

The transmission and reception schedule of the nodes is given in Table 1.

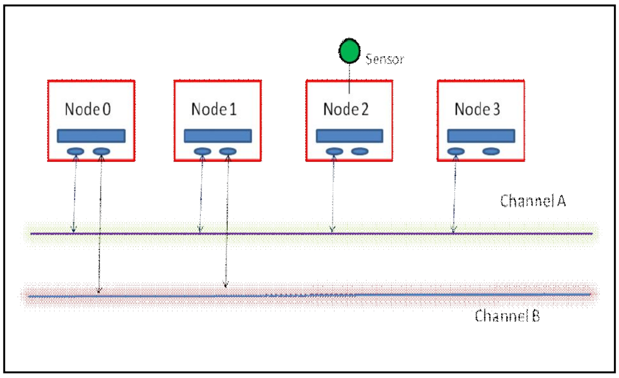

Fig. 2 : Nodes connected respective channel(s)

TABLE 1. TRANSMISSION AND RECEPTION SCHEDULE OF NODES

Node	Type of frame	Mode	Slot	Buffer	Cycle
Node 0	Sync	tx	3	0	0
Node 0	Dyn	tx	13	5	0
Node 1	Sync	tx	4	0	0
Node 1	Dyn	tx	15	6	0
Node 1	Dyn	tx	16	7	0x46
Node 2	Static	tx	8	2	0x42
Node 3	Static	tx	7	3	0
Node 0	Sync	rx	4	1	0
Node 0	Dyn	rx	15	6	0
Node 0	Dyn	rx	16	7	0x46
Node 1	Sync	rx	3	1	0

Node 1	Dyn	rx	13	5	0
Node 1	Static	rx	8	2	0x 42
Node 1	Static	rx	7	3	0
Node 2	Sync	rx	3	1	0
Node 2	Sync	rx	4	4	0
Node 2	Dyn	rx	13	5	0
Node 2	Dyn	rx	16	7	0x 46
Node 3	Sync	rx	3	1	0
Node 3	Sync	rx	4	4	0
Node 3	Dyn	rx	13	5	0
Node 3	Dyn	rx	15	6	0

V. ALGORITHM OF CODE FLOW

A. Algorithm for Coldstart node code flow:

Step 1: Begin
Step 2: Configure the System and Peripheral Clock for for FlexRay™ module.
Step 3: Initialization of interrupts for FlexRay™ module.
Step 4: Initialization of Ports for FlexRay™ module.
Step 5: Initialize the Node to Node 0 Schedule:
Configure the parameters, Baud rate, Static and Dynamic buffers, Transmission and Reception Schedule etc
Step 6: Allow the node to Coldstart for start up of the cluster
Step 7: Check whether the node is in Normal Active?
If NO? Goto Step 8.
If YES? Goto Step 10.
Step 8: Check whether CAS is received?
If NO? Goto Step 7.
If YES? Goto Step 9.
Step 9: Initialize the Node to Node 1 Schedule:
Configure the parameters, Baud rate, Static and Dynamic buffers, Transmission and Reception Schedule etc.
Goto Step 7.
Step 10: Get input from the user
Step 11: Display the temperature, magnetic field and the range of the Sensor.
Display no. of times various interrupts occurred.
Step 12: End

B. Algorithm for Non-Coldstart node code flow:

Step 1: Begin
Step 2: Configure the System and Peripheral Clock
Step 3: Initialization of interrupts for FlexRay™ module
Step 4 : Initialization of ports of FlexRay™ module
Step 5: Configure the Peripheral Clock for Sensor module.
Step 6: Initialization of interrupts for Sensor module.
Step 7: Initialization of Ports for Sensor module.
Step 8: Initialize the Node to Node 'arbitrary' Schedule:
Configure the parameters, Baud rate, Static and Dynamic buffers, Transmission and Reception Schedule etc
Step 9: Check whether the node is in Normal Active?
If NO? Goto Step 9.
If YES? Goto Step 10.
Step 10: Check whether Synchronization frames received?
If NO? Goto Step 10
If YES? Goto Step 11
Step 11: Check whether Node 2 data available?
If NO? Goto Step 12
If YES? Goto Step 13
Step 12 : Check whether the wait for Node 2 data is over?
If NO? Goto Step 11
If YES? Goto Step 14
Step 13: Initialize the Node to Node 3 Schedule:
Configure the parameters, Baud rate, Static and Dynamic buffers, Transmission and Reception Schedule etc. Goto Step 15
Step 14: Initialize the Node to Node 2 Schedule:
Configure the parameters, Baud rate, Static and Dynamic buffers, Transmission and Reception Schedule etc
Step 15 : Check whether the Node is in Normal Active?
If NO ? Goto Step 15.
If YES? Goto Step 16
Step 16: Get input from the user
Step 17: Display the temperature, magnetic field and the range of the Sensor.
Display no. of times various interrupts occurred.
Step 18: End

VI. RESULT ANALYSIS

Fig.5 gives the overview of actual communication taking place between the nodes.

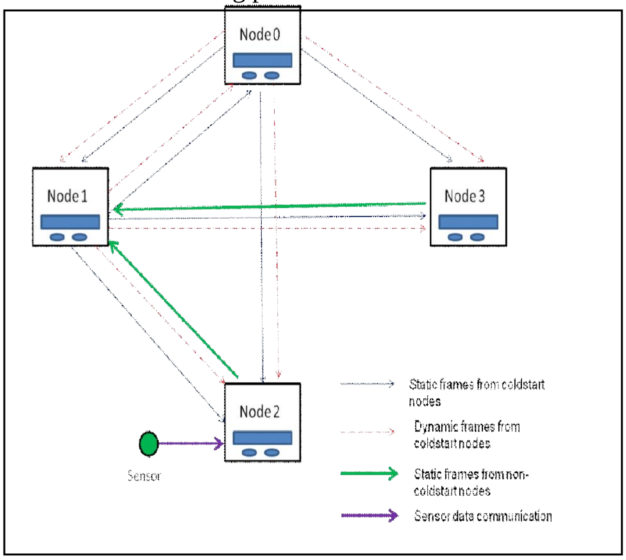

Fig. 5 : Actual communication taking place

Summary of the application:

1. Cluster communication initiation was successful by exchanging startup frames and synchronization was established by using the data present in startup frames. Interrupts generated indicated the correct transmission and reception of frames between node 0 and node 1.
2. The non-coldstart node then joined the cluster. The new node in the cluster initialized to node arbitrary schedule and collected sync frames from node 0 and node 1. This node is only connected to channel A. This node then waited for the frames from node 2 if existed. When this was not successful, this node was initialized to node 2.
3. Successful transmission and reception of frames between the node 0, node 1 and node 2 is observed through interrupts.
4. Switching off the node 2 did not affect the transmission and reception of frames between the node 0 and node 1.
5. The non-coldstart node then joined the cluster. This new node in the cluster initialized to node arbitrary schedule and collected sync frames from node 0 and node 1. This node is only connected to channel A. This node then waited for the frames from node 2 if existed. When this was successful, this node was initialized to node 3.
6. Successful transmission and reception of frames between the node 0, node 1, node 2 and node 3 is observed through interrupts.
7. Switching off the node 2 did not affect the transmission and reception of frames between the node 0, node 1 and node 3.Adding another new node to the cluster when node 2 was switched off, joined the cluster being node 2.
8. Similarly, switching off the node 3 did not affect the transmission and reception of frames between the node 0, node 1 and node 2.Adding another new node to the cluster when node 3 was switched off, joined the cluster being node 3.
9. Sensor connected to node 2, senses the environment continuously and sends the data to the sensor module. Upon the new data being received in the sensor module, the new data is moved into the register.
10. Upon the reception of new sensor data in the sensor module of node 2, the new data stored in the register is sent to node 1 using FlexRay™ frame.
11. Upon the user input, room temperature, magnetic field near the sensor and the range of the magnetic field of the sensor displayed at respective node i.e., node 1 and node2.

VII. CONCLUSIONS

FlexRay™ multinode communication is established successfully with both broadcast and node specific communication on Infineon SoCs. The sensor data being received continuously at node 2, is transmitted successfully to node 1 using FlexRay™ Protocol, which is verified by displaying the temperature and magnetic field at node 1.

VIII. FUTURE WORK

To develop an efficient safety critical real time application using FlexRay™ multinode cluster and hence evaluate its performance.

IX. ACKNOWLEDGMENT

We would like to thank Infineon India Pvt. Ltd, Bangalore and R V. College of Engineering, Bangalore for giving us support and guidance to carry out this work successfully.

REFERENCES

[1] Bongjun Kim and Kiejin Park “Probabilistic Delay Model of Dynamic Message Frame in FlexRay Protocol, ”IEEE Conference, 2009.

[2] S. P. Mane S. S. Sonavane P. P. Shingare, “A Review on Steer-By-Wire System Using Flexray”, IEEE Conference, 2011.

[3] Yi-Nan Xu, I. G. Jang, Y. E. Kim, J. G. Chung and Sung-Chul Lee, “Implementation of FlexRay Protocol with An Automotive Application,” IEEE SoC Design Conference, 2008.

[4] FlexRay Consortium, http:// www.flexray.com.

[5] Richard Murphy, “A CAN to FlexRay Migration Framework,” Msc. Thesis, Waterford Institute of Technology, Ireland.

[6] K. Schmidt and E. G. Schmidt, “Message scheduling for the FlexRay protocol: The Dynamic segment,” IEEE Trans. Veh. Technol., vol. 58, no. 5, pp. 2160–2169, Jun. 2009.

[7] K. Schmidt and E. G. Schmidt, “Message scheduling for the FlexRay protocol: The Static segment,” IEEE Trans. Veh. Technol., vol. 58, no. 5, pp. 2170–2179, Jun. 2009.

[8] Yasser Sedaghat and Seyed Ghassem Miremadi, “A Low-Cost On-Line Monitoring Mechanism for the FlexRay Communication Protocol,” Fourth Latin-American Symposium on Dependable Computing, 2009

[9] FlexRay Communication Protocol Specification V2.1 Rev A- FlexRay Consortium

[10] Denis Waraus, “Steer-by-wire system based on FlexRay protocol"

[11] Avinash K R, P Nagaraju, Surendra S, Shivaprasad S,” Communication between Multiple Nodes Using FlexRay™ Protocol”, Proc.International Conference on Electronics and Communication Engineering(ICECE 2012), Chennai,India, April 2012.

❖❖❖