International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue 2, February - 2014

Flexray Protocol in Automotive Network Communications

[1.] Anjan Kumar B S, [2.] Arpitha Rani R, [3.] Keya Priyambada, [4.] Arti Kumari
[1.] Asst.Professor, Department of Instrumentation Technology, Bangalore Institute of Technology, K R. Road, Bangalore-560 004
[2,3,4.] Department of Instrumentation Technology, Bangalore Institute of Technology, K R. Road, Bangalore-560 004

***Abstract*— Automotive systems play a vital part of life all over the world. Driven by the ongoing shift from mechanical to electrical systems in vehicles, the Flexray consortium defined communication systems which made many automotive industries to become their members to bring out changes in their production. Flexray is developed to fulfil the increasing demand in automotives for higher safety and comfort. The main feature for which it overtook CAN, and LIN protocols are because of the following: high speed serial communication, wake on bus command feature common to both controllers and to active stars, and the ability to have a redundant bus, which offers increased fault tolerance communication between electronic devices. In addition this gateway communication can realize bidirectional data transmission on Flexray bus and henceforth this protocol is widely being implemented in leading automobile industries. Flexray is focussed around a set of core needs which will be outlined in this paper.**

I. INTRODUCTION

Flexray is a fast, deterministic and fault-tolerant bus system for automotive use, based on the experience of well-known OEMs (Original Equipment Manufacturer) with the development of prototype applications and the byteflight communication system. Byteflight was developed especially for use in passive safety systems (airbags). In order to fulfil the requirements of active safety systems, byteflight was further developed by the Flexray consortium in particular in relation to time-determinism and fault tolerance. CAN (controller area network) was first developed for use in the automotive industry but was found to be useful in other areas such as industrial control applications. The CAN networking scheme uses a priority driven bus arbitration system. This means that a message with a higher priority message ID will be given access to the network if a lower priority message is also looking for access to the bus. The resulting message transmission delays can lead to problems for safety systems and because of this a TDMA (time division multiple access) method was chosen for the Flexray protocol. [1][2]

II. ARCHITECTURE

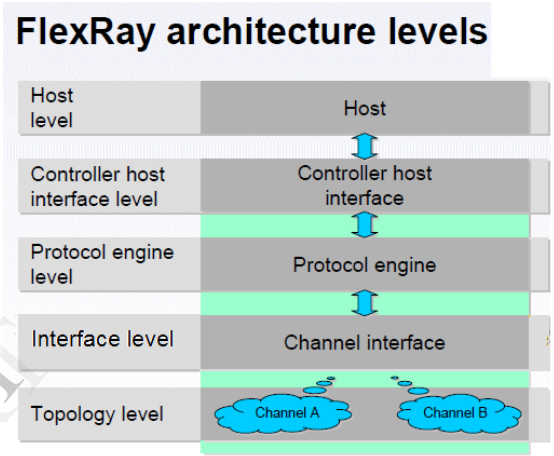

Figure 1: Flexray architecture levels

A. *Topology Level*

The Flexray protocol defines a two channel network, Channel A and channel B. A node can be attached to one or both of these channels. If a node is attached to a single channel it does not matter if it is channel A or channel B. The Flexray protocol allows for various bus topologies. These can be a point to point connection, passive star, linear passive bus, active star network, cascaded active stars, hybrid topologies and dual channel topologies. The Flexray protocol will support hybrid topologies as long as the limits of each topology which makes up the hybrid topology (i.e. the star and bus topologies) are not exceeded. [3]

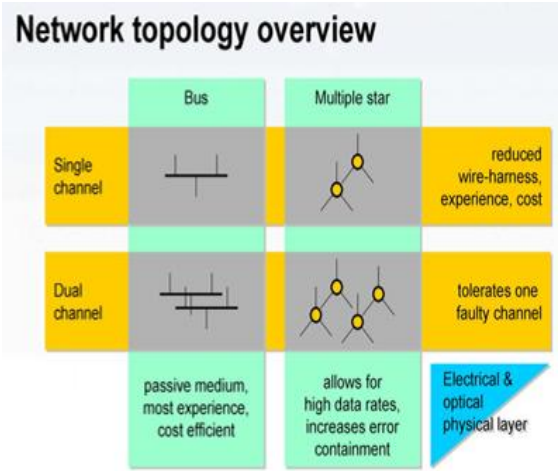

Figure 2: Network Topology Overview

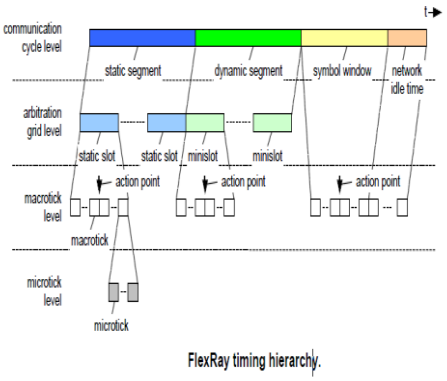

Figure 3: Flexray Timing Hierarchy

B. Interface Level

Flexray supports bus guardian at physical interface. It enforces error containment in the time domain, and performs error detection in the time domain. Bus guardian interacts with communication controller (signal monitoring and synchronization) and host processor (configuration, activation/deactivation, error signalling) [4].

The Bus Guardian- The bus guardian is used in Flexray to protect the communication channel against faulty behaviours of communication controllers. Each communication controller has a bus guardian. On the one side, the bus guardian should prevent the communication controller from accessing the communication channel outside its pre-allocated slots. On the other side, the bus guardian should

Guarantee that messages from non-faulty communication controllers are correctly relayed. We identify four properties of the bus guardian:

• Correct Relay. If a correct communication controller sends a message, its non-faulty bus guardian relays the message.

• Validity. If a non-faulty bus guardian relays a message, then all correct communication controllers receive the message.

• Agreement. If a non-faulty communication controller receives a message, then all non-faulty communication controllers receive the message.

• Integrity. If a message is received by a non-faulty communication controller, the message must have been sent by another non-faulty communication controller.

C. Protocol Engine Level

Static Segment

The static segment is broken down into smaller Sections called static slots. Every static slot is of the same duration. During transmission each slot is assigned to a specific message and only that message can transmit during that slot time.

Dynamic Segment

The Dynamic segment is an optional section of the Communication cycle. It is broken down into smaller sections known as mini slots. If a node wishes to communicate it must wait until its mini slot comes around. If no transmission occurs after a given period the mini slot counter is incremented and the node with the next message/frame id may begin transmission of data. The data to be sent will only be sent if there is enough time left in the dynamic segment. In this way the dynamic segment is priority driven with the message with the lowest ID having the highest priority, just like CAN.

Symbol Window

A symbol is used to signal a need to wake up a cluster amongst other things. This depends on the symbol sent and the status of the controller at the time. Within the symbol window a single symbol may be sent. If there is more than one symbol to be sent then a higher level protocol must determine which symbol gets priority as the Flexray protocol provides no arbitration for the symbol window.

Network Idle Time

The network idle time is used to calculate clock adjustments and correct the node's view of the global time. It also performs communication specific tasks and uses up the remaining time of the communication cycle.

Conceptual hierarchy of the communications system layers

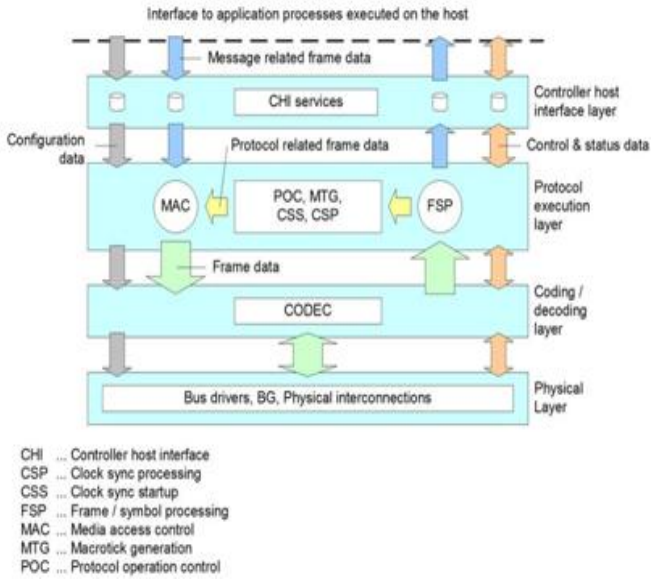

Figure 4: Interface to application processes executed on the host

The central layer of this hierarchy is the protocol execution layer. Within the protocol execution layer outgoing frame data is sent to the physical layer according to the time-driven media access strategy defined for media access control (MAC). Frame data contains not only 0-254 bytes of message data that is held in the payload section of the frame, but also 5 bytes of protocol related data that is held in the header section of the frame. The frame is secured using a 24 bit cyclic redundancy check (CRC) that is stored in the trailer segment of the frame. Incoming frame data is checked by frame / symbol processing (FSP) against a set of syntactical and semantical acceptance criteria. The message contained in an accepted frame is passed to the controller host interface (CHI) for storage while the protocol related frame data is provided to the core processes of the protocol execution layer. These core processes consist of the protocol operation control (POC), the macrotick generation (MTG), the clock synchronization start-up (CSS) and the ongoing clock synchronization processing (CSP) that executes the synchronization algorithm.

On the one hand the protocol execution layer interfaces to the controller host interface layer that contains storage means for all interface data and the controller host interface services, on the other hand the protocol execution layer interfaces to the coding / decoding layer that performs the non-return to zero (NRZ) coding and decoding of frames. The frames are exchanged among nodes on the physical layer, which forms the lowest level of the hierarchy. The physical layer contains the bus drivers, the bus guardians and the physical interconnections including any star couplers or other hubs that are located in the interconnection path.

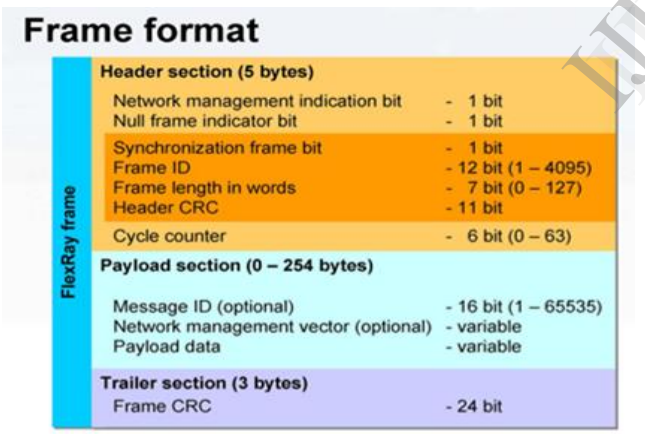

Figure 5: Frame Format

Header

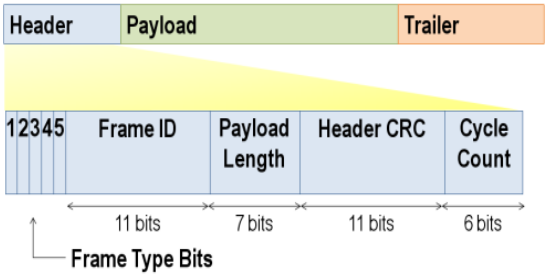

Figure 6: Bit-level breakdown of a Flexray Frame

The Frame ID defines the slot in which the frame should be transmitted and is used for prioritizing event-triggered frames. The Payload Length contains the number of words which are transferred in the frame. The Header CRC is used to detect errors during the transfer. The Cycle Count contains the value of a counter that advances incrementally each time a Communication Cycle starts.

Payload

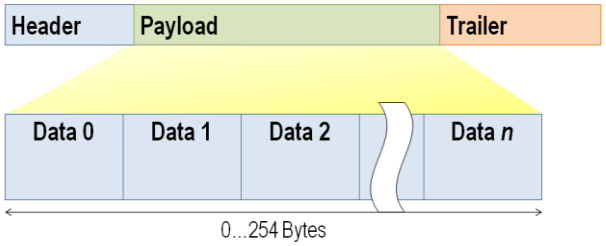

Figure 7: Payload of a Flexray Frame

The payload contains the actual data transferred by the frame. The length of the Flexray payload or data frame is up to 127 words (254 bytes), which is over 30 times greater compared to CAN.

Trailer

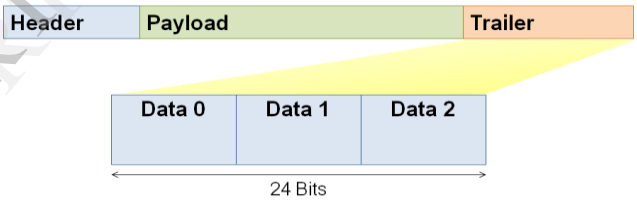

Figure 8: Trailer

The trailer contains three 8-bit CRCs to detect errors.

D. Controller Host Interface Level and Host Level

Functions currently defined include a timer service, an interrupt service, a message ID filtering service, and a network management service. The timer service allows time-outs to be defined for the application based on the synchronized time base maintained by the protocol. Upon reaching a time-out the interrupt service raises an interrupt to the host. The message ID filtering service provides means for selecting receive buffers based on a message ID that is exchanged in frames that are enabled for message ID filtering. This allows using a message selection concept such as in CAN, where message IDs embed the semantical meaning of the associated message. The network management service supports an application-level functional network.

III. HARDWARE PARAMETERS

Hardware parameters let the integrator customize the design to remove unused hardware. For a Flexray device, there could be several system-dependent parameters like bus and data width, and architectural parameters like the maximum number of

message buffers and payload length. The maximum number of message buffers (4 to 256) has a big impact on the area and the clocking requirement of the CHI, whose frequency requirement can range from 20 to 140 MHz The freedom to implement only the required message buffers eases the way to a design optimized for cost, area, power Optimization and topology analysis validation.

IV. SOFTWARE DESCRIPTION

Vector supports the user with software components and Individualized service for universal development of Flexray systems up to series production. Development is simplified by improved tools that are tuned to one another, like for instance DaVinci Network Designer for all Flexray-typical design tasks or CANoe.Flexray 6.0 for simulation and stimulation of a network, integration tests and rest-of-bus simulation as well as analysis of the finished Flexray network. CANape 6.0 is used to access to all internal parameters of the Flexray ECU via the standardized measurement and XCP-on-Flexray calibration protocol. The Flexray Evaluation Bundle provides for quick and flexible implementation of a Flexray network. This integrated environment of software components and tools also includes a sample application for a Flexray system with two nodes.

A. Embedded Software for Flexray Systems

To fully exploit the advantages of Flexray-based communication, it makes sense to fundamentally develop the associated basic software according to the AUTOSAR specification. AUTOSAR specifies a new development methodology, software architecture and basic software. OEMs may gradually introduce it step-by-step on new vehicle models. The standard-conformant ECU-specific software is modularly structured. This enables partitioning of the software components above the RTE (Remote Terminal Emulator) and the basic software below the RTE. The basic software has a modular internal structure and is specified by clearly defined interfaces, so that software from any source can be used in integration. In addition, the standard defines which exchange formats can be used and how the interfaces between individual modules need to operate. This modularity makes it easier to scale software features to the specific requirements of a vehicle variant.

B. Flexray Software Examples Included in the SK-91F467-Flexray Starter Kit

91460_template_91467d example

Within the delivery there is also an example called "91460_template_91467d". This template example should be used, when starting a new project. All required files (e.g. start-up file, header ...) and Tool settings (e.g. Assembler, C/C++-Compiler, and Linker) are included. Also, two Configurations are included, STANDALONE and MONDEB_INTERNAL, to switch between Debugging and final application. For details about the template example and the Softune Workbench Monitor Debugger refer to the USER GUIDE of the SK-91F467-FLEXRAY starter kit.

91460_templateFR_91467d

This template_FR example is a special template version for Flexray application. In addition to the template example following is prepared: The workspace contains two Projects, called "Node1" and "Node2". All additional files for the DECOMSYS::COMSYSTACK are already inside the "src" folder, "src_shared" or the "Generated_files" folder. The start91460.asm file is already adapted to SK-91F467-FLEXRAY
starter kit. CS1 enabled, 32-bit data width, address starting 0x80.0000 for external SRAM. CS3 enabled, 16-bit data width, address staring at 0x50.0000 for Flexray CC (MB88121) when starting a new Flexray application it is recommend copying the template_FR workspace and renaming the files accordingly. [6]

V. FEATURES

The main features of the Flexray protocol are as follows:

- Two channel each of which capable of 10 Mbps data rate; the two channels can be used to implement a redundancy mechanism or in stand-alone way, reaching an aggregate data rate of 20Mbps (twenty times as faster as CAN bus).

- Deterministic behaviour: during the Flexray system configuration, it is necessary to set the communication cycle period length, which is divided in a static and in a dynamic time window. The first one is reserved for synchronous communication and is able to guarantee a specified frame latency and jitter through a mechanism of fault tolerant clock synchronization; in other words, static time windows is suitable for time triggered messages. The latter one is instead reserved for event triggered messages, prioritized in a way similar to CAN bus, that is by setting specific bits in the message header. Flexray messages, moreover, can have a frame length from 2 to 254 bytes, which means a significant increase compared to the 8 byte length of CAN bus. CAN adopts priority arbitration for message delivery; that means that low priority messages will always be delayed by high priority messages, and only the highest priority message has a guaranteed delivery. Determinism means that nothing happens by chance: the correct output is always determined by its input and this behaviour can be extended to the entire network, making it a predictable system. Clock synchronization is very important in the Flexray architecture to guarantee a deterministic behaviour. Each communication cycle period, a synch message is transmitted by each synchronization node on the network (usually a Flexray network includes at least four synchronization nodes). When each node receives a synch message, compares its clock with that transmitted by the synchronization node and makes the necessary correction to match them. If doing so one node fails, the others can continue to work since all of them have been correctly synchronized.
- Fault tolerance: it is achieved both implementing redundancy at system configuration level and, on the physical layer, with the Bus Guardian, an

independent circuit which is able to protect a channel from interference caused by data not aligned with the protocol schedule. Fast error detection and signalling is also provided, as well. Nevertheless, collisions on bus are significantly reduced with Flexray.

- Support of electrical and optical physical layer.
- Support for bus, star, and multiple star topologies.[5]

VI. APPLICATIONS

The Flexray logo depicts the ray fish. With its characteristics, Flexray makes it possible to apply the so called x-by-wire technology (where x stands for drive, steer, and brake) to the automotive world. By-wire means that the hydraulic systems traditionally used to perform those tasks are eliminated and replaced by lightweight, non toxic, more efficient and more maintainable electro-mechanical systems.

Industrial applications

In the industrial area the industrial Ethernet is being propagated to solve these problems. This interface suffers from poor real time capability and an enormous overhead by the needed hardware. Not to mention the extensive software drivers and the huge variety of different available protocols.

Flexray is offering exactly the needed capabilities. Through a redundant communication path, an extremely high security can be achieved. The real time ability will be provided by a time triggered system and data will be distributed to an exact guaranteed time to the involved receiver. The needed hardware and software expenditure is relatively small, which allows using also lower cost controllers. The high performance, the jitter and collision free transmission and the little overhead allow an easy design of complex control systems. A possible scenario could be a simple sensor/actor satellite construct, which is linked to a powerful central control unit. Such a system excellently suits motor control and its synchronization. There are innumerable other applications realizable with decentralized intelligence, which are not conceivable with Ethernet or other field buses [7].

Flexray in Airplanes

This technology was firstly introduced in the avionics sector (where it is known as flight-by-wire) to assist the pilot in the flight of supersonic airplanes. On military airplanes such as the F16, when flying at more than 1 mach with some g of acceleration, the pilot does not have to push with strength on the pedals or on the control stick: it can just control the airplane by means of a joystick-like control electrically connected to electro-mechanical actuator and sensors.

Flexray in Automobiles

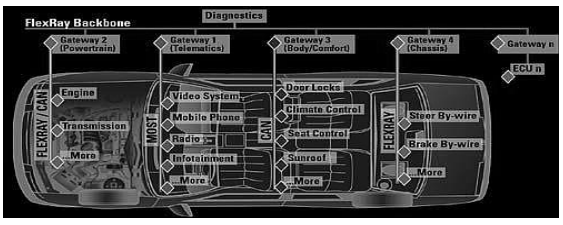

Figure 9: Flexray/CAN Model of Automobile

In 2000, BMW, DaimlerChrysler, Motorola (now Free scale), and Philips Semiconductors founded the Flexray Consortium with the declared purpose to develop a high-speed, safe, and reliable communication protocol for automotive applications, and to make that technology available also to other competitors (no royalties have to be paid for automotive applications). Flexray is becoming a standard for advanced power train, chassis, and x-by-wire systems. It is not going to substitute current protocols and networks: CAN, LIN, MOST, and J1850 will continue to exist, but they will be integrated and work in conjunction with Flexray.

All these properties explain why Flexray is an excellent solution for x-by-wire applications, where real-time, high speed, and fault tolerance are mandatory requirements. As mentioned before, a typical Flexray application is represented by the brake-by-wire system. The idea beneath this solution is to eliminate the dependence on hydraulic systems, increasing the vehicle stability control and safety. ABS (Anti-lock Braking System) is today adopted on many types of vehicles, but stability control is still a complex and expensive option.

Brake-by-wire, also called as EMB (Electro-Mechanical Braking), generates the braking force on each separated wheel by means of powerful and efficient electric motors, connected to an electrical control unit which receives the command from an electronic brake pedal unit. In this application Flexray plays a dominant role, providing the communication protocol with fast-speed, fault-tolerant, and deterministic behaviour. Since each wheel can be controlled independently, there are potentially no limits to the stability control algorithm: additional sensors could be added to detect, for instance, the weight distribution, the passengers' allocation, the tire pressure, and the external terrain conformance, so that the fastest, more precise and comfortable braking action might be executed.

Moreover, hydraulic braking systems use toxic fluid, and their faults are not easily detectable: a brake-by-wire solution is more environment-friendly and permits the usage of test and diagnostic tools. Flexray protocol has already been chosen to implement some x-by-wire solutions on commercial automobiles. For instance, the BMW flagship SUV X5 was the first vehicle on which Flexray has been commercially used; one application of Flexray on this car is the ability to choose, in a real-time manner, the correct shock absorption setting in order to achieve the best stability. [5]

VII. CONCLUSION

For automobiles to continue to improve safety, increase performance, reduce environmental impact, and enhance comfort, the speed, quantity and reliability of data communicated between a car's electronic controls units (ECU) must be improved. Advanced control and safety systems--combining multiple sensors, actuators and electronic control units--are beginning to require synchronization and performance more than what the existing standard, Controller Area Network (CAN), can provide. Coupled with growing bandwidth requirements, today's advanced vehicles utilize over five separate CAN busses, for which automotive engineers are demanding a next-generation, embedded network. After years of partnership with OEMs, tool suppliers

and end users, the Flexray standard has emerged as the in-vehicle communications bus to meet these new challenges.

REFERENCES

[1] Flexray Communications System – Protocol Specification, v2.1 Revision A, Flex Ray Consortium, Dec. 2005.
[2] Flex Ray Consortium, http:// www.flexray.com.
[3] http://repository.wit.ie/898/2/CF-005622.pdf.
[4] http://www.flexray.com/products/protocol%20overview.pdf
[5] http://dev.emcelettronica.com/flexray-protocol-x-wire-becomes-reality.
[6] http://www.vector.com/portal/medien/cmc/press/Vector/FlexRayOS_ElektronikAutomotive_200609_PressArticle_EN.pdf.
[7] IEEE Paper on Automotive Communications - Past, Current and Future by Thomas Noltey, Hans Hanssony and Lucia Lo Belloz