IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)
e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197
PP 83-88
www.iosrjournals.org

Design For High Performance Flexray Protocol For Fpga Based System

E. Singaravelan[1], S. Kalaiselvy[2], A. Anbarasan [3]
[1]*PG Student, VLSI Design, Department of ECE, Surya group of institutions, India*
[2] *Assistant professor, Department of ECE, Surya group of institutions,, India*

Abstract: *In recent most advanced applications we can use the benefit from tight coupling of the embedded computing units with the communication interface, thereby providing functionality beyond the Flex Ray standard. Such an approach is highly suited to implementation on reconfigurable architectures. In this paper we presents highly pipelined FSM based flex relay protocol that consists of a reconfigurable processing module, including reconfigurable compute units and output control logic, CAM memory units, and peripheral circuits. Flex Ray communication controller (CC) which integrates configurable extensions that augment the CC's capabilities beyond those defined by the Flex Ray standard. The main core comprises of Protocol Engine (PE) which implements the protocol behavior, and the Controller Host Interface (CHI) which interfaces to the host ECU. Here we describe an FPGA-based communication controller which features configurable extensions to provide functionality that is unavailable with standard implementations. Finally for custom interfacing purposes such as in embedded controller applications Nios II based system build and core bus in-between the main processor and the I/O interface are used.*

I. INTRODUCTION

The Processor design highly optimized (area, power & speed) electronic control units (ECUs) for soft core embedded processer and prove the efficiency through SOPC builder based hardware synthesize and verify the functionality through Modelism. Flex-Ray is a new communication protocol designed to provide large bunches of data to be exchanged in real-time and with high dependability between electronic control units (ECU). It features data-rates up to 10 Mb/s. It is accounting for both time and event triggered transmissions. (Beyond CAN)[1] .The host checks wakeup event, based on this decides Modes of Operation by state will be next. In the form of static slots for synchronous time-triggered communication and dynamic slots for burst mode event-triggered (priority based) data transfer. In modern processor widely used CAN protocol for internal communication.CAN bus speed 10mb/s and operation under event triggered method. Area, size and power requirement is large. Speed is limited (10mb/s) and conflict occur in modes of operation. In this paper, we present an architecture-optimized Flex Ray communication controller (CC) [4] which integrates configurable extensions that augment the CC's capabilities beyond those defined by the Flex-Ray standard. The controller provides enhancements to the data path, like programmable width time stamping, data filtering, header insertion and processing functions, which are abstracted away from the host function. Our flexible architecture can be used to design advanced ECUs on reconfigurable hardware that consume less power and offer increased consolidation, while providing enhanced capabilities that are impossible to implement using standard controllers or IP cores[2] We also quantify the potential of the proposed controller using case studies based on existing and evolving automotive applications that are safety-critical and data-intensive. Our experiments show that advanced features like high-speed mode switching for fault-tolerant ECUs, low latency data handling for high performance gateways, timeliness and security for messages can be efficiently achieved by integrating such extensions within the controller data path rather than offloading them to the processing logic.

.

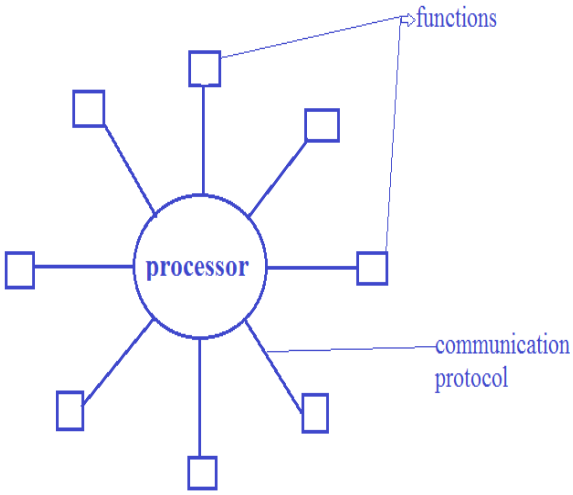


Fig .1 Block diagram of Felx Ray protocol

II. RELATED WORK

The investigate the scheduling problem for the dynamic segment of Flex-Ray. We formulate a nonlinear integer programming problem (NIP) that minimizes the duration of the dynamic segment. In order to obtain efficient schedules. While the CC independently implements the Flex-Ray protocol services. The CHI[5] of each node contains a buffer for each related FID. The host implements a periodic scheduling table (PST) per allocated FID. Each node consists of a host and a communication controller (CC) that is connected by a controller-host interface (CHI).While the CC independently implements the Flex-Ray protocol services. An arrow indicates the current message to be transferred to the respective transmit buffer in the CHI. The flex-ray network splits the bus into separate branches and operates as a selective central switch. The proposed device is increasing both available bandwidth and safety of existing Flex-Ray networks. Schedule Execution (SE) module implements the clock synchronization algorithm. Coding And Decoding (CODEC) [4] module extracts synchronization information. The MTG stores the current micro- and macro tick and updates the time base accordingly .Measurement of the deviation values and calculation of the correction values is done by the clock synchronization process (CSP). Switched Flex-Ray networks use star coupler called a switch, thereby increasing the effective network bandwidth. Branch parallelism and multiplexing is available. Safety-critical in-vehicle electronic control units(ECUs) demand high levels of determinism and isolation. It uses FPGA partial reconfiguration and a customized bus controller to offer fast recovery from faults. Results show that such an integrated design is better than alternatives that use discrete bus interface modules. To an increase in the number of computational nodes and the need for faster in-vehicle networks. The Predictability and isolation of safety-critical function .Providing both static and dynamic flexibility with high computational capabilities at lower power consumption. FPGAs improve upon processor-based ECUs by providing better determinism and segregation. The customizable and reconfigurable nature of the fabric can be exploited by implementations addressing a wide range of applications for modern and future in-vehicle systems. Non-safety-critical applications like multimedia and driver assistance can leverage the high computational capability of FPGAs [2] while taking advantage of dynamic re-configurability to multiplex functions. FPGAs also enable techniques for implementing multiple levels of fault-tolerance and redundancy to support FPGA-based safety-critical ECUs for the drive train and drive-by-wire systems.

III. ARCHITECTURE DESIGN

Flex-Ray is a new communication protocol designed to provide large bunches of data to be exchanged in real-time. It features data-rates up to 10 Mb/s .It is accounting for both time and event triggered transmissions.(beyond CAN). Much work has been done on scheduling communication on the shared bus. Optimizations of the static and dynamic segment of the Flex-Ray protocol has been widely addressed in among others. The survey of scheduling algorithms and provides a comparison between optimization strategies like simulated annealing, genetic, hybrid-genetic and probabilistic approaches applied by various algorithms. Given a set of communication requirements, all algorithms try to optimize the number of communication slots and cycles that are required to schedule the different messages, satisfying all requirements. The optimization in most cases is to find the minimum number of communication slots that can solve the problem, hence consuming minimum bandwidth. Alternatively, the problem can be formulated to maximize the number of unused slots, which provides flexibility for future expansion.

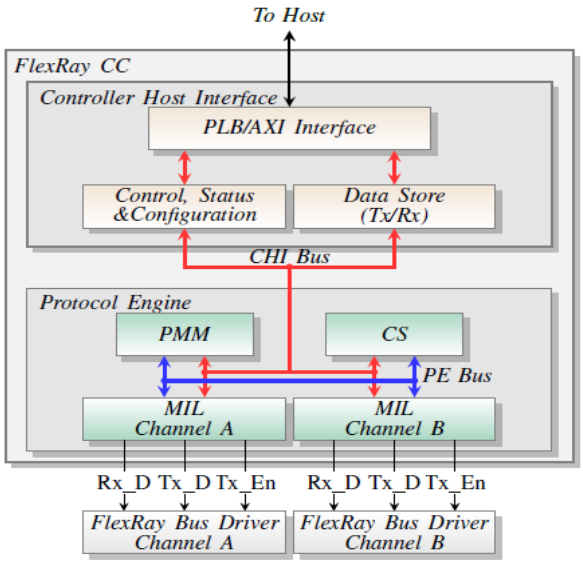


Fig 2 Architecture of custom Flex-ray communication controller.

The Flex-Ray CC switches between different operating states, based on network conditions and/or host commands, ensuring conditions defined by the Flex-Ray protocol are met at all times. The CC architecture [5], as shown in Fig. 3, comprises the Protocol Engine (PE) which implements the protocol behavior, and the Controller Host Interface (CHI) [3] which interfaces to the host ECU. The CHI module communicates with the host and handles commands and configuration parameters for the Flex-Ray node. These parameters are defined for the particular cluster the node is operating on, and are initialized during the node's configuration phase. The CHI feeds the current state and operational status to the host for corrective action if necessary. There are transmit and receive buffers and status registers for the data-path to isolate control and data flow. The CHI may also incorporate clock domain crossing circuitry to enable the different interfaces to work in distinct clock domains.

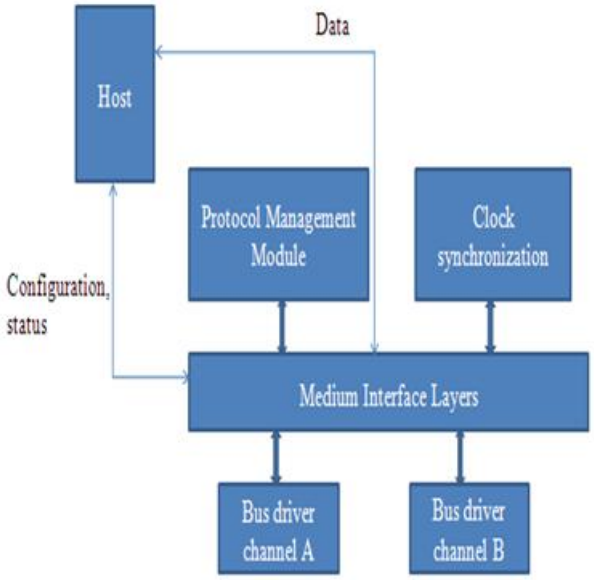


Fig 3 Functional block for PMM

These corrections must be applied in the same way at all nodes and must full fill the following conditions:
1) Rate correction is continuously applied over the entire cycle.
2) Offset correction is applied only during the NIT in an odd cycle and must finish before the start of the next communication cycle.
3) Rate correction is computed once per double cycle, following the static segment in an odd cycle. The calculation is based on values measured in an even-odd double cycle.
4) Calculation of offset correction takes place every cycle but is applied only at the end of odd cycle.

It has PMM,CS and MIL [4]for major constituent part.PMM is to control and coordinate the clock synchronization and medium interface layer. Macro and micro timer are synchronized by use of clock synchronization.MIL uses encoding and decoding for transmission. Through hardware synthesis area , power & speed of PMM, CS and MIL modules are analyzed. Improve the overall throughput rate. It features data-rates increase up to 10 Mb/s. It is accounting for both static and dynamic transmissions.

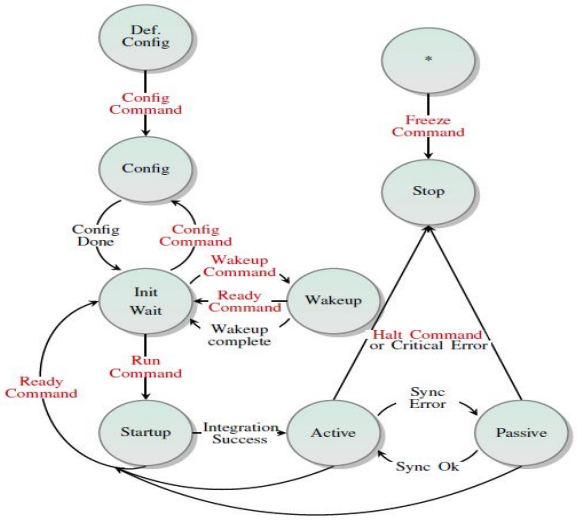
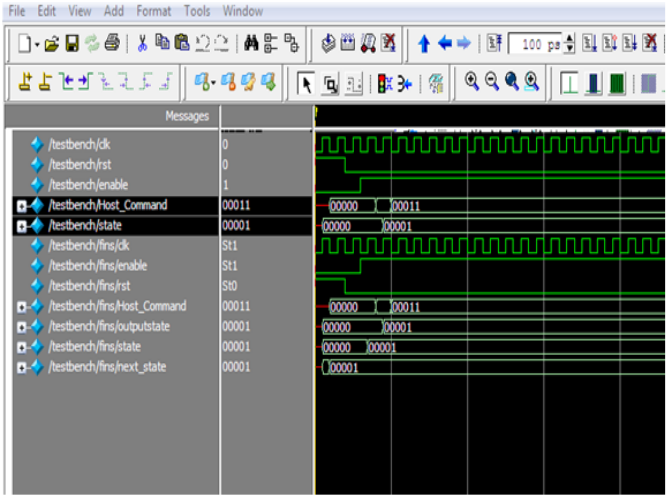


Fig 4 Flex-ray cc modes

Give configuration command, states moves to wait then wakeup. Give run command from wait state, entire process will done in active state .If synchronization error arises, it will moves to passive and critical command in the state of halt mode. Communication Controller is designed using finite state machine (FSM) [5] methodology. Speed is independent number of states, and depends only on the number of transitions into a particular state. Traditional controllers depend on the host processor to read the received data and determine the usefulness of it. The controller issues a data interrupt, to which the processor responds with a status register read followed by a data read request, subsequently receiving the data. These overheads are wasted in the case of frames with irrelevant data (like obsolete or untimely data) or multi-cycle data frames where the processor cannot process the received fragment until more data is available. In the case of critical data frames like error state that require immediate attention, the latency introduced by the traditional scheme limits the performance of safety critical systems which rely on host-triggered recovery.

IV. SOFTWARE IMPLEMENTATION RESULTS

a) Functional verification for CC Modes



b) Performance of area

Flow Summary

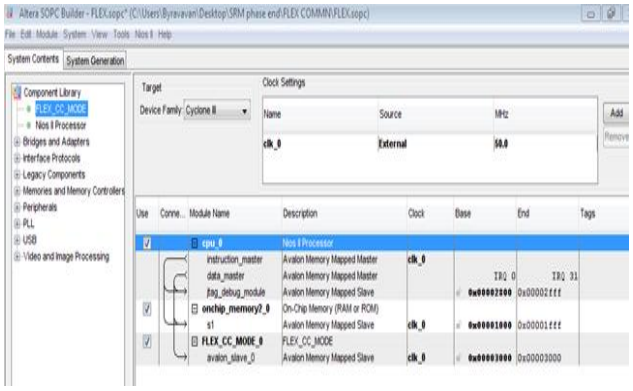
Flow Status	Successful - Wed Sep 10 22:22:47 2014
Quartus II Version	9.0 Build 132 02/25/2009 SJ Web Edition
Revision Name	QTOP
Top-level Entity Name	FLEX_CC_MODE
Family	Cyclone III
Device	EP3C16F484C6
Timing Models	Final
Met timing requirements	N/A
Total logic elements	46 / 15,408 (< 1 %)
Total combinational functions	46 / 15,408 (< 1 %)
Dedicated logic registers	12 / 15,408 (< 1 %)
Total registers	12
Total pins	13 / 347 (4 %)
Total virtual pins	0
Total memory bits	0 / 516,096 (0 %)
Embedded Multiplier 9-bit elements	0 / 112 (0 %)
Total PLLs	0 / 4 (0 %)

c) Power analyzer

PowerPlay Power Analyzer Summary

PowerPlay Power Analyzer Status	Successful - Wed Sep 10 22:22:47 2014
Quartus II Version	9.0 Build 132 02/25/2009 SJ Web Edition
Revision Name	QTOP
Top-level Entity Name	FLEX_CC_MODE
Family	Cyclone III
Device	EP3C16F484C6
Power Models	Final
Total Thermal Power Dissipation	65.19 mW
Core Dynamic Thermal Power Dissipation	0.00 mW
Core Static Thermal Power Dissipation	51.73 mW
I/O Thermal Power Dissipation	13.45 mW
Power Estimation Confidence	Low: user provided insufficient toggle rate data

d) IP Core Integration



V. CONCULSION

In this paper, we have given an overview of the Flex Ray protocol and the generic architecture of the communication controller, as defined by the specification. By identifying and extracting operations which are mutually exclusive or natively parallel, we have designed a custom controller which takes advantage of the heterogeneous resources on modern FPGAs, resulting in reduced logic footprint, and low power consumption, while providing a host of features beyond those described by the standard and to give an overview of the Flex Ray protocol and the generic architecture of the communication controller to Reconfiguration to provide flexible

use of the FPGA. The future scope is to improve the overall throughput rate by integrating control layer within the controller data part and All peripherals is integrated as a IP core. Finally High pipelined FSM is used in controller circuits for better performance

REFERENCES

[1]. S. Chakraborty, M. Lukasiewycz, C. Buckl, S. Fahmy, N. Chang, S. Park, Y.Kim, P. Leteinturier, and H. Adlkofer, "Embedded Systems and Software Challenges in Electric Vehicles," in Proc. Design, Automation and Test in Europe (DATE) Conference, 2012.

[2]. S. Shreejith, S. A. Fahmy, and M. Lukasiewycz, "Reconfigurable Computing in Next-Generation Automotive Networks," IEEE Embedded Systems Letters, vol. 5, No. 1, pp. 12–15, 2013.

[3]. I. Sheikh, M. Hanif, and M. Short, "Improving information throughput and transmission predictability in Controller Area Networks," in Proc. International Symposium on Industrial Electronics (ISIE). IEEE, 2010, pp. 1736–1741.

[4]. J. K ¨ otz and S. Poledna., "Making FlexRay a Reality in a Premium Car," in Proc. of the SAE International, 2008.

[5]. FlexRay Communications System, Protocol Specification Version 2.1 Revision A, FlexRay Consortium Std., December 2005.