# Design Considerations for Optimizing Real-Time Control and Communication for the Kiira Electric Vehicle

Gerald Baguma, Sandy Stevens Tickodri-Togboa, Paul Isaac Musasizi, Nancy Senabulya, and Pauline Korukundo

***Abstract*—The Vehicle Design Project based at the School of Engineering, Makerere University embarked on development of a two-seater Electric Vehicle (EV) code named KIIRA as a proof of concept. This paper presents the design considerations in developing the vehicle's control and communication systems. It highlights the hardware and software, architectural and structural concerns that were considered, in order to achieve high response times while maintaining the predictability and optimal performance of the system. The architectural perspectives for putting together eligible control nodes are also presented, with emphasis on the compositionality and composability requirements of the system. The paper elucidates how the adopted architectural framework and software policies have provided a schedulable and dependable distributed system for the vehicle, thereby achieving a high level of performance and Real Time network optimization in the Kiira Electric Vehicle.**

***Index Terms*—Control network, determinism, electric vehicle, real time, responsiveness, priority scheduling**

## I. Introduction

Automotive Safety Critical Systems require high levels of responsiveness and determinism. Thus, the implementation of dependable automotive systems requires electronic control units and buses that can respond timely and consistently to the automotive environment. Since automotive control networks require a high level of predictability and reliability [1], [2], the Electronic Control Units in the Kiira EV have been designed to achieve low granularity with simple and fast algorithms. Vehicle Control nodes are highly autonomous with optimal cooperation implemented over the Controller Area Network (CAN) communication protocol, and because of the robustness and reliability of CAN [3], hard and soft deadlines in the system are guaranteed to be met.

Electronic control units are integrated in such a way that they address compositionality and composability concerns, and thus, the vehicle electronic control units are autonomous. The communication is synchronous and asynchronous; asynchronous because the control units have to capture sensor inputs at the respective interfaces, and synchronous for periodic and timely transfer of vehicle control commands. Because of this type of communication, generative event triggered architecture is most suitable [4]. The main ECU executes a parallel task execution mechanism with bound deadlines to increase both responsiveness and speed [5], [2]. Sensor and actuator controls of the main ECU for the Kiira EV have been implemented using the National Instrument Compact Reconfigurable input/output (NI CRio) that is designed for applications that require high performance and reliability. The programming pattern of the CRio utilizes Field Programmable Gate Arrays (FPGAs) which are significantly faster than the traditional Application Specific Integrated Circuits (ASIC) [6] and can significantly achieve high levels of high-speed hardware reliability and tight determinism [7]. Since the reliability of networked embedded systems depends significantly on the execution environment [8] the Kiira EV network has been implemented with the LabVIEW Real Time(RT) platform which uses the LabVIEW Real Time Engine that is designed to implement deterministic and real-time performance for data acquisition and control applications [9], [10].

Paper Outline: Section II describes the System Architecture of the ECUs for which optimization considerations have been made. . Section III examines the processes through which the ECUs optimize network wide and ECU centric level responsiveness. Section IV discusses the considerations taken to establish a high level of determinism in the distributed system and minimize jitter. Section V discusses considerations taken to achieve optimal network message priorities in the distributed computing environment and Section VI discusses the conclusions made on the used policies in the KIIRA EV environment.

## II. System Description

The distributed control network was targeted for the power train applications, and as such, the network is designed to encompass four major nodes; the Motor Controller, Battery Management System (BMS), Instrument Cluster Display and the Vehicle Control Unit.

The BMS monitors the state of the battery, calculates secondary data and reports the data to other network nodes; it protects the battery pack and also controls its environment.

The Motor control module not only provides the communication to the front drive train, but also oversees and controls all the activities of the motor, inverter and transmission systems using the embedded algorithms therein. The motor, inverter and transmission systems initiate the physical dynamic forces in the EV system. The BMS monitors the voltage, current, speed, torque and environment variables through sensor interfaces and avails this information to the CAN bus for notification of other networked nodes. This ensures that sufficient power levels

Manuscript received April 14, 2012; revised June 2, 2102. This work was supported in part by the Center for Research in Transportation Technologies.

The authors are with the Department of Electrical and Computer Engineering at Makerere University. (e-mail: gbaguma@tech.mak.ac.ug, stogboa@tech.mak.ac.ug, pim@tech.mak.ac.ug, nsenab@umich.edu, pkorukundo@tech.mak.ac.ug)

DOI: 10.7763/IJIEE.2012.V2.203

are generated by the motor through the drive by wire interface, to match the drivers' mode of operation, for example, power output should match high speed mode, an uphill climb, slow speed mode and parking mode.

Instrument Cluster Display provides the car with a "head" to perform vehicle on-board diagnostics and provides warning and emergency help functionality. The Vehicle Control Unit monitors the entire vehicle control and communication systems. Its design is based on the NI Compact RIO, a programmable automation controller (PAC) produced by National instruments. The VCU provides a rugged, reconfigurable embedded system containing three components – a real-time controller, a reconfigurable FPGA, and industrial grade I/O modules, making it versatile and best suited for noisy automotive conditions.

In an attempt to design a viable system, the project team concentrated on a distributed system policy that ensures a high level of responsiveness, good determinism, reduced jitter and optimal priority scheduling. The next sections describe how these have been achieved in the prototype

## III. Macroscopic and Microscopic Responsiveness

The close deadline coupled with the need to develop a reliable and fault tolerant system resulted into the choice of the NI Real Time reconfigurable frame work for the EV prototype. This enabled building optimized and flexible electric circuitry for the vehicle without actually building custom circuitry. This architecture combines a low power consumption processor with a high performance Reconfigurable I/O (RIO) FPGA that provides a matrix of configurable-logic blocks which are surrounded by hot swappable C series industrial modules to support I/O and communication, all in one chassis.

The RIO core has in-built data transfer mechanisms to pass data to the embedded processor for Real Time analysis and communication to other networked nodes [10], [11]. This heterogeneous hardware architecture chosen provided the team with a reliable reconfigurable embedded system for both testing and deployment within the project window. Additionally it "compressed" a myriad of ECUs, as is the current trend with automotive industries [12], on a single platform thus saving power and improving network performance given the Low CAN bandwidth [13].

One of the software design policies specified in the requirements engineering was a system whose tasks meet their deadlines with 100% certainty. This encompasses both system level tasks (macroscopic) and ECU centric tasks (microscopic). Networked tasks, which require distributed processing, were treated as critical irrespective of whether the task has to meet a hard deadline or a soft deadline. Based on this assumption, the design formulated a fixed cyclic schedule for these tasks since the state change delays and process delays in the firmware for the networked nodes is known apriori. Each of the tasks in the periodic schedule is considered to have its deadline at the beginning of the other task in the fixed schedule. Similarly each of the tasks begins to execute at the beginning of its period. All the pre-emption costs are taken care of by estimating an appropriate laxity that does not conflict with the deadline of other tasks. This provided network principles which fully guarantee that all the deadlines will be met. To ascertain this level of responsiveness at the software level, a layer by layer approach was adopted as suggested in [2]. As noted in [1] the state space conditions of an RT multi-core system are too large to explore all the possibilities of Best Case Execution Time (BCET) and Worst Case Execution Time (WCET). By estimating bounds for the execution times of every execution thread in the firmware, threads were set at optimal performance limits. It is against this that an overall fixed cyclic schedule for the system was determined, following the system policies thus established. Since the real-time technology used offers reliable, deterministic performance for time-critical applications [7], [9], the set execution bounds in software loops are guaranteed to be timely. This is additionally enhanced by the RTOS running on the RT processor core in the CRio embedded system and the FPGA module that provides high speed access to the targets on the platform.

This implementation is characterized by parallel execution loop architecture, with each loop having an established execution window. These loops can be classified as threads. The RT processor thus picks each execution thread, runs it for its pre-specified window and halts it for the next task to execute. It then picks another execution thread and runs it. As thus the software is best on a pre-emptive algorithm. This way each execution thread on the ECU modules is kept from hogging the CPU and denying CPU access to other threads, and the network is fast enough to respond to both system wide and ECU centric commands.

## IV. Determinism and Jitter

Parallel loop execution architecture was proposed in order to achieve the design policies earlier established, but this presented another design challenge of determinism. How certain would we be that the repeatability of timing in each of the control loops would be accurate or at least within tolerable limits of jitter? To get around the impasse, a hardware platform that would support this kind of software architecture had to be used to provide fast and consistent response times. This is because automotive actuator devices have to respond in almost no time in case of time critical applications as discussed in [15]. This response has to be timely and within tolerable delays.

The NI CRio embedded System provides a reconfigurable FPGA chassis which is the center of the embedded system designed for the Vehicle Control Unit and is directly connected to the targets (C series modules).The FPGA circuitry designed is a parallel processing reconfigurable computing engine that executes the vehicle control applications in silicon circuitry on chip and provides unlimited timing, triggering and synchronization. This is because each module is connected directly to the FPGA rather than through a bus. The system experiences almost no latency each time a trigger for an execution thread is initiated. It provides a platform that can access sensor and actuator interfaces of the Vehicle with less than 500ns of jitter as discussed in [15]. For control loops that run on the RT processor, the hardware contains an industrial grade

processor with a core built to support parallelism and therefore deterministically executes LabVIEW RT applications and offers a reliable engine for multi-rate loop control. It also implies that the platform is scalable since addition of more computation does not necessarily mean reduced speed for the FPGA application.

## V. Priority Scheduling

Another system policy established in the requirements engineering phase was to integrate as much functionality as possible on each ECU, to improve ECU autonomy and reduce bus contention during CAN arbitration as a result of increase in traffic.

As an extension to this policy, an assumption was made that task execution times would only be stretched marginally by arbitration issues; therefore in an attempt to improve message timeliness and ensure that deadlines are met, the traffic on the bus had to be considerably reduced to provide low contention delays. This effect ripples to bandwidth optimization on the underlying CAN bus. Since the networked tasks (those that poll the CAN bus to receive and send updates) use a fixed cyclic scheduler as discussed in III, they rely on the CAN protocol as discussed in [17] to resolve the necessary message priority issues. Consequently, the Real Time network relies entirely on the fixed priority non-pre-emptive scheduling which is embedded in the CAN controllers. Time critical tasks such as Break by Wire, Motor Control, and BMS management functionality were given message identifiers with high priority arbitration IDs as discussed in [15], [16]. This ensures that deadlines for delivery of the required information to these nodes are met.

## VI. Conclusion

The complexity of modern automotive systems requires an efficient way to achieve real time communication and control. This paper has described both hardware and software related policies that were considered in the development of the Kiira EV in a bid to match the systems' performance requirements. First, by building optimized circuitry on FPGAs to increase actuator response times and avoid the bottlenecks caused by traditional ASICs, an almost zero delay and reliable hardware frame work was provided. By utilizing a fixed periodic schedule built on the LabVIEW RT frame work, it is guaranteed that all networked tasks meet their deadlines, and in so doing ensure a high level of responsivity in the real time control network. This, reinforced with fixed priority non-pre-emptive scheduling on the CAN bus provides a robust distributed vehicular system.

## References


[1] C. Cullman, C. Ferdinand, G. Gebhard, D. Grund, C. Maiza, J. Reineke, B. Triquet, and R. Wilhelm, "Predictability Considerations in the Design of Multi-Core Embedded Systems," 2010.

[2] J. A. Stankovic and K. Ramamritham, "What is predictability of Real Time-Systems," *Amherst: s.n*, 1993.

[3] M. A. Livani, J. Kaiser, and W. Jia, "Scheduling Hard and Soft Real-Time Communication in a Controller Area Network," 1999.

[4] J. Kaiser and M. Mock, *Implementing the Real-Time Publisher/Subscriber Model on the Controller Area Network (CAN)*.

[5] J. M. Turja, K. Hanninen, and M. Nolin, *Efficient Development of Real Time Systems Using Hybrid Scheduling*.

[6] R. Wain, I. Bush, M. Guest, M. Deegan, I. Kozin, and C. Kitchen, *An Overview of FPGAs and FPGA programming*, 2000.

[7] National Instruments. NI LabVIEW FPGA. www.ni.com. [Online]. Available: http://www.ni.com/fpga.

[8] P. H. Feiler, B. Lewis, and S. Vestal, "Improving Predictability in Real-Time Embedded Systems," Pittsburgh, 2000.

[9] National Instruments, LabVIEW Real-Time. www.ni.com. [Online]. Available: National Instruments, July 08, 2011. http://sine.ni.com/nips/cds/view/p/lang/en/nid/2381.

[10] Work In Progress, Introductory LabVIEW Real Time Data Acquisition Laboratory Activities, 2010.

[11] J. S. Falcon, "Model Development and Measurements for Real-Time Systems," 2011.

[12] National Instruments, What is Compatriot? www.ni.com. [Online]. Available: National Instruments, August 2, 2011. http://www.ni.com/compactrio/whatis/

[13] L. Peter, Trends in Embedded Development. Stuttgart, Germany: *Vector*, 2009.

[14] J. Stroop and R. Stolpe, Prototyping of Automotive Control Systems in a Time-Triggered Environment Using FlexRay.

[15] R. I. Davis, A. Burns, R. J. Bril, and J. J. Lukkien, "Controller Area Network (CAN) Schedulability Analysis," *Refuted, Revisited and Revised. 3, s.l: Springer Netherlands*, vol. 35, 2007.

[16] National Instruments, *A Primer to Machine Control*, National Intruments, 2009.

[17] CAN in Automation, can Specification 2.0, Part B. Germany.



**Gerald Baguma** was born in Kabarole District, Uganda. Baguma attended Makerere University in Kampala, Uganda and on the 17th January 2010, he obtained a Bachelors degree in Telecommunications Engineering. He has since specialized in designing Software and Hardware for embedded systems. He has worked as a Research and Teaching Assistant in the Department of Electrical and Computer Engineering for two years. He has worked as a Lead Researcher and Developer for electric vehicle systems since 2009, on the Vehicle Design Project. He has served as an intern in; Bukasa Telecom Company LTD, Spear Motors Uganda LTD and Mühlbauer High Tech international company. Gerald was a member of a team of students who prototyped the first ever Electric Vehicle in East and Central Africa, the KIIRA EV. He is now the Lead Firmware developer in an ongoing solar-electric bus project code named KAYOOLA, at the Center For Research in Transportation Technologies (CRTT) Mr. Gerald is now the Assistant Projects Manager at the Center for Research in Transportation Technologies. He together with his team were awarded the College of Engineering, Design, Art and Technology Research Innovation Award for the KIIRA EV on 17th-February 2012.