2009 International Conference on Advances in Recent Technologies in Communication and Computing

Load Balancing Towards ECU Integration

Rajeshwari Hegde,
Dept of Telecommunication Engineering,
BMS College of Engineering,
Bangalore, India,
Email:rajeshwari.hegde@gmail.com.

K S Gurumurthy,
Dept of E & C,
UVCE,
Bangalore, India,
Email:drksgurumurthy@gmail.com

***Abstract*— There has been an exponential increase in the number of electronic components embedded in vehicles. Development processes, techniques and tools have changed to accommodate that evaluation. A wide range of electronic functions such as navigation, adaptive control, infotainment, traffic information, safety system etc are implemented in today's vehicles. Many of the new functions are not stand alone and hence they need to exchange information, sometimes with stringent time constraints for time critical functions such as engine management, collision warning systems etc. The complexity of the embedded architecture in a vehicle is continually increasing. Today up to 2500 signals are exchanged through up to 70 Electronic Control Units (ECUs) using 5 different buses. This paper introduces the load balancing approach across ECUs supplied by various Tier1 suppliers.**

***Keywords*— AUTOSAR, ECU, OEM**

I. INTRODUCTION

Due to increasing number of Electronic Control Units (ECU) in automotives, time management puts a premium on prioritization. Of all the automotive innovations in the future, it is forecast that 90% will be based on electronics development, rather than mechanical systems and 80% of that amount will be based on software development. Microprocessors have been introduced in various systems like Antilock Braking System,(ABS), Electronic Power Steering(ESP), Engine Management System(EMS), Safety and Infotainment Systems (Telematics and multimedia systems).[4]. The automotive industry is currently going through a dramatic increase of electronic components for on-board vehicle control. The numbers of electronic control modules in modern automobiles increased enormously within the last few years. Due to increasing number of ECUs in automotives, time management puts a premium on prioritization.[4]. Not only the raising number of the ECUs within one automobile is a challenge, but there is also a very strong increase in functionality in every single ECU due to the safety, comfort and other requirements by the customers.[3]. These facts lead to an exponential boost of complexity regarding intra- and inter- ECU behavior. This increase in electronics in modern automotives has lead to the proliferation of ECUs [2].The main challenge of the automotive industry is to come up with methods and tools to facilitate the integration of different ECUs supplied by various Tier1 suppliers into the vehicle's global electronic architecture to reduce the complexity and cost of the vehicles. Automotive OEMs (Original Equipment Manufacturer) are facing difficulties in integrating subsystems which are designed and implemented by multiple Tier-1 vendors. In the last ten years several industry wide projects have been undertaken in that direction and significant results have already been achieved. The next step is to build an accepted open software architecture, as well as the associated development processes and tools, which should allow for easy integration of different functions and ECUs provided by car makers and third party suppliers. This is ongoing work in the context of AUTOSAR.[1]. ECUs, the fundamental electronic building blocks of any automotive subsystem, used to be relatively simple, hardware-oriented systems. Today, they are multi-purpose, multi-chip computer systems where more functionality often is delivered in software than hardware. The most complex ECUs operate the power train. Simpler ones operate functions such as power windows, power seat, mirror adjustment system, etc,. But even these ECUs need to be networked so that those specific features can be exploited both from the view of power management and such other critical co-ordinations and for enhancing the utility by way of personalization, etc. Having numerous subsystems like these, a medium to high complexity automotive contains as many as 70 ECUs which are interconnected by up to five different buses using proper gateways. This trend of increasing automotive electronic content is the direct result of many new features that will greatly increase both safety and comfort but that will require more sophisticated ECUs with a large embedded software component. The safety features include steer-by-wire, brake-by-wire and drive-by-wire (collectively known as "X-by-wire"), automatic lane-following, drowsy driver detection, intelligent cruise control and airbag systems. Of all the automotive innovations in the future, it is forecast that 90% will be based on electronics development, rather than mechanical systems and 80% of that amount will be based on software development. The paper is organized as follows. Section II explains In Vehicle Embedded System. Section III deals with ECU integration. Section IV deals with AUTOSAR technical concept. Section V explains the load balancing concepts. The paper is concluded in section VI.

II. IN VEHICLE EMBEDDED SYSTEM

As all the functions embedded in cars do not have the same performance or safety needs, different QoSs (e.g., response time, jitter, bandwidth, redundant communication channels for tolerating transmission errors, efficiency of the error detection mechanisms, etc.) are expected from the communication

978-0-7695-3845-7/09 $26.00 © 2009 IEEE
DOI 10.1109/ARTCom.2009.57

systems. Typically, an in-car embedded system is divided into several functional domains that correspond to different features and constraints [6]. Two of them are concerned specifically with real-time control and safety of the vehicle's behavior: the "power train" and the "chassis" domains. The third, the "body," mostly implements comfort functions. The "telematics" , "multimedia," and "human–machine interface" (HMI) domains take advantage of the continuous progress in the field of multimedia and mobile communications. Finally, an emerging domain is concerned with the safety of the occupant.

The chassis domain gathers functions such as ABS, ESP, ASC (Automatic Stability Control), 4WD (4 Wheel Drive), which control the chassis components according to steering/braking solicitations and driving conditions (ground surface, wind, etc). Communication requirements for this domain are quite similar to those for the power train but, because they have a stronger impact on the vehicle's stability, agility and dynamics, the chassis functions are more critical from a safety standpoint. Furthermore, the "x-by-wire" technology, currently used for avionic systems, is now being introduced to execute steering or braking functions. "X-by-wire" is a generic term referring to the replacement of mechanical or hydraulic systems by fully electrical/electronic ones, which led and still leads to new design methods for developing them safely and, in particular, for mastering the interferences between functions [8. Chassis and power train functions operate mainly as closed-loop control systems and their implementation is moving toward a time-triggered approach [9], which facilitates composability (i.e., ability to integrate individually developed components) and deterministic real-time behavior of the system. Dashboard, wipers, lights, doors, windows, seats, mirrors, and climate control are increasingly controlled by software based systems that make up the "body" domain. This domain is characterized by numerous functions that necessitate many exchanges of small pieces of information among themselves. Not all nodes require a large bandwidth, such as the one offered by CAN; this lead to the design of low-cost networks such as Local Interconnect Network (LIN) and TTP/A. On these networks, only one node, termed the master, possesses an accurate clock and drives the communication by polling the other nodes—the slaves—periodically. The mixture of different communication needs inside the body domain lead to a hierarchical network architecture where integrated mechatronic subsystems based on low-cost networks are interconnected through a CAN backbone. The activation of body functions is mainly triggered according to the driver and passengers' solicitation (e.g., opening a window, locking doors, etc). Telematics functions are becoming more and more numerous: hands-free phones, car radio, CD, DVD, in-car navigation systems, rear seat entertainment, remote vehicle diagnostics, etc. These functions require a lot of data to be exchanged within the vehicle but also with the external world through the use of wireless technology.

III. NEED FOR ECU INTEGRATION

When the ECU is finished and tested by the supplier, the car manufacturer has the task to integrate all the ECUs from all the different vendors in an overall system and do more testing. This is usually done in several separate stages: module test, part system test, integration test and total system test. Integrating ECUs, i.e. getting them to work together as planned, is a challenging process and is carried out by OEM(Original Equipment Manufacturer) who determines bus topology, speed etc,. Each ECU is attached to a number of sensors and actuators. The sensor signals are thus processed and shared via a digital bus to other modules all of which ultimately control some actuators performing a wide range of tasks ranging from electric windows to fuel injectors**.** The multiplex bus is one of the most important shared resources in a vehicle electrical system, with the electrical power being the other. The utilization of this resource is a place where interaction between vehicle functions happens. If one function increases its band-width consumption, it may affect all other functions. e.g, the commonly used CAN bus. Functions transmit data on the CAN bus within frames that contain 0 to 8 bytes of data. Frames can be sent either periodically or sporadically and have unique identifiers that are used for identification and prioritization. Prioritization works via CSMA/CA and uses a model of "dominant" bits and "recessive" bits. When two nodes transmit at the same time, the node sending the dominant bit gets access to bus. When the bus load is low, it's easy to implement this even for multiple and independently designed nodes. However, this concept has some drawbacks when the bus load is high. If, for example, a node schedules a low priority frame before a high priority frame, the low priority frame doesn't get access to the bus and therefore blocks the high priority frame. This is one of the reasons why the CAN bus is often perceived as unpredictable, non-deterministic, and not well suited for real-time processes. One way to address this issue is to use only a fraction of the theoretical bandwidth of the bus, typically 30%, leading to the need for more buses and gateways in the system and therefore increasing the cost.

IV. AUTOSAR TECHNICAL CONCEPT

The ECUs are proprietary solutions-highly customized OEM and vehicle specific components. This control module centered approach to ECU development has disadvantages. A single change in requirements may make it necessary to conduct a completely new ECU development. Increase in complexity of ECU functionality is always associated with enormous expense. Further intensifying the need for a conceptually new approach are new constraints such as the need to represent an even greater number of variants and equipment options, offer options for upgrading architectures and handle nonfunctional requirements such as ECU diagnostics or availability. Therefore there is a need to replace the existing ECU centered development approach by a functionally oriented approach.

Reductions of hardware costs as well as implementations of new innovative functions are the main drivers of today's automotive electronics. Indeed more and more resources are spent on adapting existing solutions to different environments. At the same time, due to the increasing number of networked components, a level of complexity has been reached which is difficult to handle using traditional development processes. To address this problem a paradigm shift from a hardware-, component-driven to a requirement- and function-driven development process, and a stringent standardization of infrastructure elements is necessary. One central standardization initiative is the AUTomotive Open System ARchitecture (AUTOSAR), which aims at facilitating the re-use of soft- and hardware components between different vehicle platforms, OEMs and suppliers. To achieve this, AUTOSAR defines a methodology that supports a distributed, function-driven development process and standardizes the software-architecture for each ECU in such a system.[5]. AUTOSAR also specifies compatible software-interfaces at application-level and provides a technically and economically viable target for a system-level automotive E/E design methodology. [10].

V. LOAD BALANCING

Load-balancing, by definition, is dividing the amount of work that a computer has to do between one or more additional computers so that more work gets done in the same amount of time and, in general, all processing get done faster [11]. It is the assignment of work to processors and is critical in parallel simulations. It maximizes application performance by keeping processor idle time and interprocessor communication as low as possible. The problem of load balancing is much more difficult in large distributed systems. Algorithms have to minimize both load imbalance and communication overhead of the application. Additionally they should be efficient themselves and scalable.[12]. In applications with constant workloads, static load balancing can be used as a pre-processor to the computation. Other applications, such as adaptive finite element methods, have workloads that are unpredictable or change during the computation; such applications require dynamic load balancing that adjusts the decomposition as the computation proceeds. Numerous strategies for static and dynamic load balancing have been developed in embedded systems, including recursive bisection (RB) methods, space-filling curve based (SFC) partitioning and graph partitioning. In the migration strategy, each processor works out a schedule for the exact amount of load that it should send to (or receive from) its neighboring processors. Once this schedule is worked out, each processor decides which particular node it should send to or receive from its neighboring processors.[12]. The migration of load then takes place. The scheduling algorithms are mostly iterative and hence there is a startup cost which is usually very high compared with the subsequent cost of transmitting a word.

Consider the load balancing across ECUs. In static load balancing, the load on each ECU is known in advance. Hence the work is equally distributed across ECUs and no extra cost is required for balancing the load [6]. This can be explained using graph theory, where in, vertices represent individual ECU load and the edges represent the amount of load to be transferred from one ECU to another [9].Dynamic Load Balancing across ECUs can improve the utilization of CPUs and the efficiency of parallel computations through migrating workload across CPUs at runtime. Workload migration can be carried out through transferring processes across nearest neighbor ECUs. Iterative strategies have become prominent in recent years because of the increasing popularity of point-to-point interconnection networks. [11]. There are many reasons to institute load balancing across ECUs.

The two most popular are:

1.Response time- With two or more ECUs sharing the load, each of them will be running less of a load than a single ECU alone, there by keeping the response time low.

2. Redundancy-If a load is balanced across 3 ECUs and one of them dies completely, then the other two can keep running and a vehicle will not even notice any downtime.

Any load-balancing solution worth its salt will immediately stop trying to send traffic to the down ECU.

Usually, the load-balancing mechanism aim is to move the running tasks across the CPUs in order to insure that no CPU is idle while some tasks are waiting to be scheduled on other CPUs.

A. Why Load Balancing

Distributed systems such as automotives can suffer from poor performance due to a bottleneck at overloaded ECUs. To address this performance bottleneck, an adaptive load balancing is used to distribute the load from densely loaded ECUs to scarcely loaded ECUs. Not much research has been done on keeping the load balanced across ECUs. To achieve good performance, it is essential to maintain a balanced work load among all the ECUs. Sometimes the load can be balanced statically. However, in many cases, the load on each ECU cannot be predicted a priory. Dispatching tasks from densely loaded ECUs to scarcely loaded ones to improve the overall performance of the vehicle is both logical and feasible.[3] A schedule of the work load that should be moved between any two ECUs , such that each ECU will have the same load on completion is a challenging task. One way to balance the load is to dispatch the job immediately upon arrival. The best load balancing status occurs when all ECUs are at the point of full utilization, without saturation. Each ECU's work load is proportional to its capacity. Allocating more jobs to a fully utilized ECUs might cause imbalance without improving the overall throughput. Since the data movement between ECUs incurs communication cost, the schedule should give balanced load with minimal data movement. Restricting the data movement to the neighboring ECUs might reduce communication cost. According to dimension Exchange Algorithm, the ECUs can be grouped in pairs and an ECU pair (a, b) with load la and lb will exchange load, after which each will have the load (la+ lb)/2.It can be assumed that the optimal load balancing policy will try to equalize CPU utilization of each ECU.[13].

B. Load Balancing Concepts

Load balancing plays import role in ECU integration. Modern automotive system contains as many as 75 ECUs which are interconnected by up to five different buses using proper gateways. Fig.1 shows the Load balancing across six ECUs on the basis of CPU utilization. For effective load balancing, status information and CPU utilization of each ECU have to be known. [14].

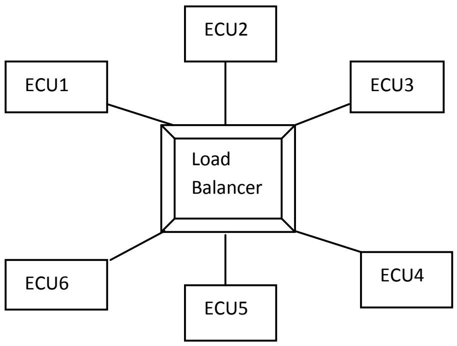

Figure1: Load balancing concepts

Load balancer attempts to ensure that loads are balanced across a group of ECUs. The fig.2 shows the Load balancer components. [14].

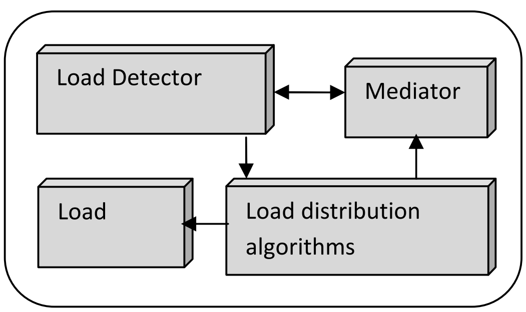

Figure2: Components of Load Balancer

The detector detects and initiates a trigger when an overload or load imbalance occurs. Load Estimator estimates the load on overloaded and under loaded ECU and informs the mediator about the imbalance. The trigger from the detector tells the mediator to establish a load balancing session between the two entities, namely offloading ECU (ECU with the higher load doing the offloading) and the load-accepting ECU (ECU accepting load from the offloading ECU). Depending on which performance metric is to be balanced, one of the offloading algorithms is invoked. Finally, the mediator is invoked to establish the load balancing session.

VI. CONCLUSION

We presented a load balancing approach based on the CPU utilization of ECUs. Load balancing plays important role in the integration of ECUs supplied by various Tier1 suppliers. The load balancing approach eases ECU integration by distributing the load equally on the ECUs and improves overall performance of the automotive system.

REFERENCES

[1].Automotive Embedded Systems Handbook by Nicolas Navet, Francoise Simonot-Lion, Industrial Information Technology Series, CRC press.

[2]. Rajeshwari Hegde, K S Gurumurthy, "Model Based Approach for the Integration of EUCs", IAENG, World Congress on Engineering, London, July 2nd -4th 2008.

[3]. Rajeshwari Hegde, K S Gurumurthy, "Challenges in Integration of ECUs in Automotives", IEEE International Advance Computing Conference (IACC 2009) Patiala, India, 6–7March 2009.

[4]. Joachim Fetzer, Dieter Lederer, Matthias Wernicke and Klaus-Jürgen Amsler, "Virtual Design of Automotive Electronic Networks". Automotive Electronics I/2004, special issue of ATZ, MTZ and Automotive Engineering Partners, Vieweg Verlag (Wiesbaden, Germany), p. 2-4

[5] .Matthias Wernicke and Jochen Rein, Integration of Existing ECU Software in the Autosar Architecture, ATZelektronik 01I2007.

[6].Thomas Heurung Mentor Graphics," In-Vehicle Network Design Methodology".

[7]. Neal Snooke and Chris Price, Department of Computer Science, University of Wales, Aberystwyth, UK, Challenges for Qualitative Electrical Reasoning in Automotive Circuit Simulation

[8]. Eugen Mayer**,** "Status quo and future of ECU communication in automobiles", Vector Informatik.

[9]. Razvan Racu, Rolf Ernst,Kai Richter, Marek Jersak, , "A Virtual Platform for Architecture Integration and Optimization in Automotive Communication Networks" SAE World Congress 2007.

[10]. Antal Rajnak, Ajay Kumar, "Computer-aided Architecture Design & Optimized Implementation of Distributed Automotive EE Systems", Annual ACM IEEE Design Automation Conference, 2007

[11]. http://webhosting.devshed.com/c/a/Web-Hosting-Articles/What-is-Load-balancing-and-Do-I-Need-It/

[12]. Ralf Diekmann, Burkhard Monien, Robert Preis, " Load Balancing Strategies for Distributed Memory Machines", In:F Karsch, H.Satz(ed.): Multi-Scale Phenomena and their Simulation, World Scientific, 1997 (to appear).

[13]. Helen D. Karatza, Ralph C. Hilzer, "Load Sharing In Heterogeneous Distributed Systems", Proceedings of the 2002 Winter Simulation Conference.

[14]. Rajeshwari Hegde, K S Gurumurthy, "Load Balancing Across ECUs in Automotives", Proceedings of ICCSN 2008, China.

[15]. Alex K. Y. Cheung, Hans-Arno Jacobsen, "Dynamic Load Balancing in Distributed Content-based Publish/Subscribe", ACM Middleware 2006.