Model Based Approach for the Integration of ECUs

Rajeshwari Hegde, K S Gurumurthy

Abstract— **A modern automotive system is a complex electro-mechanical system, whose comfort, safety and performance requirements have warranted their implementation by way of multiple number of Electronic Control Units (ECUs) in it. ECUs, the fundamental 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 is often delivered in software than in hardware. An attempt has been made in this paper to introduce the concept of model based approach for the integration of ECUs supplied by multiple Tier1 vendors.**

Index Terms—**AUTOSAR, ECU, OEM**

I INTRODUCTION

Importance of electronics is increasing day by day in modern automotive systems and the same has lead to the proliferation of ECUs. The number of Electronic Control Units (ECUs) and the amount of functionality is increasing with every new car coming on the road [1]. Automotive OEMs(Original Equipment Manufacturer) are facing difficulties in integrating subsystems which are designed and implemented by multiple Tier-1 vendors. Most of the ECUs currently used are “one box solution” for each application, resulting in ECUs of different complexities and capabilities being supplied by different vendors functioning in a single vehicle, further adding to the complexity. To make the situation worse, these vendors also reserve their design philosophy and details as proprietary assets. The whole automotive supply chain, including car manufacturers, electronic subsystem providers and component providers, struggles to cope with an ever increasing functionality implemented on a staggering number of ECUs. Further, ECU software has to satisfy constraints like tight cost, performance and safety. Today, 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 greatly increase both safety and comfort which 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, that can adjust deployment based upon passenger weight and the specific nature of an accident.

The integration of ECUs in Automotives has to deal with 1000’s of signals that are exchanged among up to 70 ECUs.
The noticeable increase in driving comfort and increased driver safety are important competitive factors in the automobile market. The Integration of ECUs plays an even
more important role for both goals. Recently, ECUs with many functions have been introduced due to semiconductor integration, and advances in hybrid IC technology. Therefore many automotive manufacturers are researching high performance, fault-tolerant ECUs. Managing the complexity of the ECU systems is a challenging task.

Following factors must be considered while designing an ECU.

- Real Time Requirements.
- Resource consumption-CPU, Memory, Power, Physical space.
- Dependability-safety, reliability, availability.
- Life cycle properties(Long Life systems)-Maintainability, expandability.
- Interoperability.

All these aspects of architecture have to be captured precisely by a modeling approach. These issues are closely related and are interdependent. Model-based approach has been introduced in the automotive industry in an effort to keep the ECU software development process manageable. This approach provides an executable model that can be used not only as a functional specification for implementation, but also as a new basis for testing. Starting tests through simulation, as soon as the model becomes available means functional failure is detected early, saving time and cost in the overall development process [11].

The paper is organized as follows. Section II answers the question as to why is model based approach required.
Section III gives an overview of Auto electronics innovations. Section IVdeals with the integration of ECUs. Section V looks at the general tool requirements. Section VI deals with the AUTomotive Open System ARchitecture (AUTOSAR). The paper is concluded in section VII.

Manuscript received February 10, 2008. Rajeshwari Hegde is with Department of Telecommunication Engineering, BMS College of Engineering, Bangalore, India(phone:918026622130,Extension:3068,Mobile-919242411560,;email:rajeshwari.hegde@gmail.com).K S Gurumurthy is with University Vishwesharaiya College of Engineering, Bangalore, India.(phone:918022961821;email:drksgurumurthy@gmail.com)

II WHY MODEL BASED APPROACH?

A very promising approach for the development of Embedded Software in Automotive systems is the so-called conceptual model-based design process which is supported by the Matlab/Simulink/Stateflow Tools [3]. Its nature is depicted in Figure 1. Requirements given in a requirements document are first translated into a formal specification (Simulink/Stateflow model)representing the system under design. By formally modeling and specifying a design, formal verification technologies can be used to completely validate whether this formal design specification meets its intended requirements.

Automatic Model Validation offers a technology for performing model-based automatic formal verification by model checking for reactive embedded systems designed using Matlab/Simulink/Stateflow. After developing a model, it has to be transferred into C code and implemented on the ECU. Automatic code generation is used for this purpose. In contrast to earlier versions of code generators today's methods of code generation are mature enough to bridge the gap between model based function design and production quality ECU code. Designs, such as those for avionics and automotive systems, have become too complex to develop and coordinate without the creation of a design environment common to all involved developers.

Model-based design provides a single design environment that enables developers to use a single model of their entire system for data analysis, model visualization, testing and validation, and ultimately product deployment, with or without automatic code generation.[2]

I.

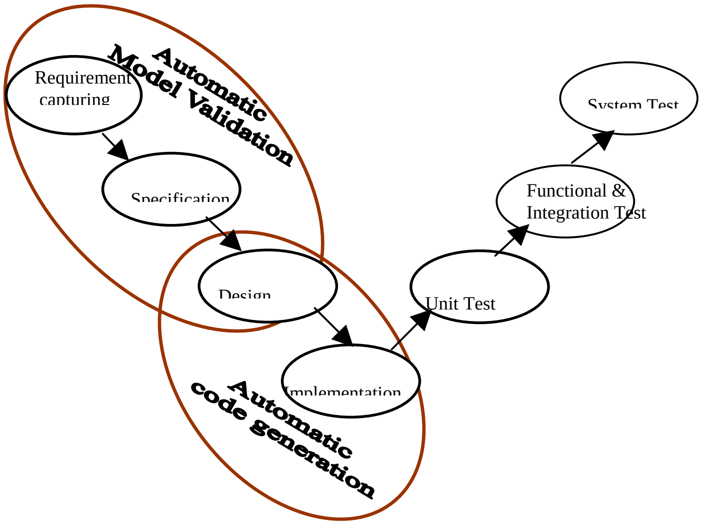
Requirement capturing
Automatic Model Validation
Specification
Design
Implementation
Automatic code generation
Unit Test
Functional & Integration Test
System Test

Figure 1:Automatic Model Validation and Automatic Code Generation in a Model Based Approach for Embedded System

Model-based design works in the following manner:

- The entire system model is visualized via block diagrams and state charts to describe knowledge and implementation details.
- Design options can be evaluated and systems performance predicted via simulation of the system model.
- Algorithm and behavioral models are optimized and refined yielding a fully tested specification.
- Production quality software is automatically created for real-time testing and deployment from the fully tested specification [2].

The model based design approach consists of various phases as shown in figure 2.

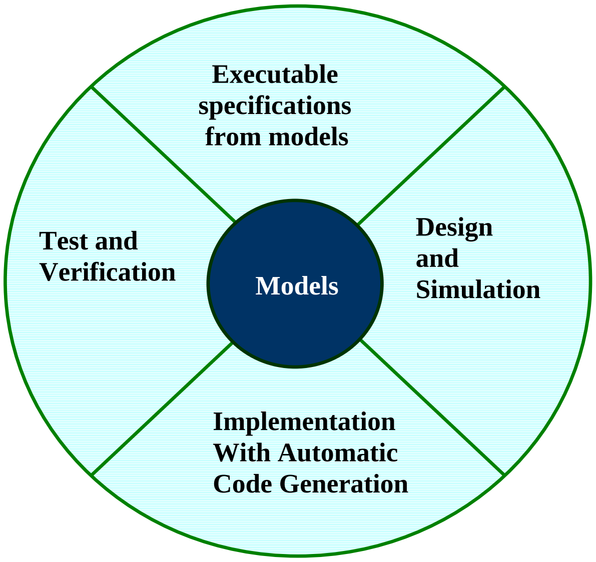
Executable specifications from models
Test and Verification
Models
Design and Simulation
Implementation With Automatic Code Generation

Figure2: Model Based Design approach

The models in this Model-Based approach are used in multiple ways: to provide executable specifications, to analyze the system's dynamic behavior, to simulate system components and environmental conditions that reduce or eliminate the need for costly physical prototypes, and to design the algorithms. Furthermore, automatic code generation from these models has become an accepted way to arrive at the production ECU software. In the coming years, it is expected to become the primary approach for implementing embedded control software in many automotive companies.[4]. The model based approach allows the design engineers to graphically specify the behavior of a system and simulate and execute it in a very early development stage. The tool environment offered by Mathworks is a widespread model based development tool for different industrial domains such as automotive and avionics systems. A precise knowledge of the architecture of the ECU is the prerequisite for the model based design methodology.

III AUTOELECTRONICS INNOVATIONS

The automotive electronics market has been growing faster than the overall electronics market and much faster than actual vehicle production. For the next several years, research predicts that automotive electronics will grow at a rate of more than seven percent. Over the course of this decade, the worldwide market for automotive electronics is expected to double. The percentage of an automobile's value attributable to the electronic content is growing and is projected to reach 30 percent by 2008 end. A generic presentation of any automotive subsystem shall be as depicted in Figure 3.

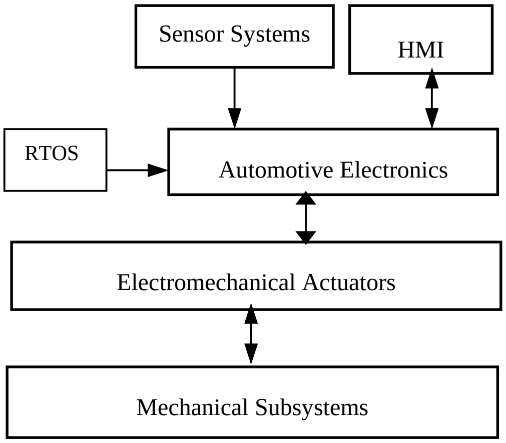

Figure 3: A generic Automotive Subsystem

In addition, a larger portion of the electronic content will be embedded software, tripling by the end of the decade. By 2010, more than 40 percent of the total value of automotive electronics will be in the software. Electronic Control Units (ECUs), the fundamental electronic building block of any automotive subsystem, used to be relatively simple, hardware-oriented systems. Today, they are multi-purpose, multi-chip computer systems where more functionality is often delivered in software than hardware. The most complex ECUs operate the powertrain. 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 or 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 which themselves are connected by proper gateways. Improving fuel efficiency is an important goal: hybrid (internal combustion and battery) and fuel-cell electric drive place high demands on software. Telematics and in-car entertainment will further increase the electronic content of cars, requiring the combination of such technologies as wireless connectivity, global positioning, digital radio and Internet access, all with hands-free voice activation whenever possible.

IV NEED FOR INTEGRATION

A clear trend in automotive networking is reduction in the number of ECUs in a vehicle. ECU is the most important resource that is being shared in automotive system. With the computing power that is available at present, it is efficient to share this resource. Consistent efforts must be made towards running as many applications as possible within the same control module (ECU). Several standardization initiatives happening in the automotive world are aimed at supporting such integration. In the 1990s automotive OEMs such as Daimler-Chrysler made great efforts to establish the OSEK embedded operating system as a binding standard for in-house developments and supplier components. Today automotive OEMs and suppliers use the real-time and multitasking capable operating system as the basis for improved code quality, good structuring and integration of the components of different suppliers. Working committees have also been established in the areas of software, software testing, process assessment, simulation and tools as well as flash programming. Latest and the most important initiative being AUTOSAR (Automotive Open System ARchitecture), which is a Consortium responsible for the standards of future vehicle generations.

Reduction in number of ECUs provides huge opportunity towards saving cost, reducing complexity and possibility of adding new features using existing computing resources available across ECUs in the vehicle. Advances in the hardware technology, like advent of multi-core processors, make it possible to provide enough computing resources on a single ECU to integrate multiple functionalities.

V GENERAL TOOL REQUIREMENTS

Each tool should provide some specific needs for modeling and simulation. Verifiability is a requirement to the design tools. Closely related to this is the need that they should be conformable to IEC 61508 [8], the standard defining the functional safety, electronic safety-related systems. The reasons are quite obvious: If a product should be certifiable, the tools which are used to produce it must meet the same requirements. This reproducibility is achieved through several prerequisites [1]. As in other application domains, *single source* is preferable to multi source in the automotive domain too. Multi source can be a consequence of different wrong techniques used: A model created using one tool may have to be re-modeled if another tool is used for creating the same model, because there is no link between them, which introduces the possibility of conversion mistakes and the constraint to use only one tool after the conversion. Another reason for unwanted multi source is that if an automated import from one model type into a tool is realized, but there will be no link back to the original application. This is not a problem as long as changes are made only in the original program and then re-imported into the second one. But if any design or parameter change is made in the second tool, the system gets inconsistent because it is very difficult to ensure that the original model is modified the same way. The Mathworks's Simulink and Stateflow and Target Link, the production code generator from dSPACE can be used for

Model based design. Ex:Power Window System can be modeled using model based design with Simulink, Stateflow, SimMechanics, and SimPowerSystems. Simulink provides an environment where we model our physical system and controller as a block diagram. SimPowerSystem and SimMechanics are used to model the electrical and mechanical components of the system.

VI AUTOSAR STANDARD

Using the traditional, ECU centric approach do design E/E architectures, the vehicle manufacturer writes a specification of all the software involved, sometimes given to the supplier as an executable prototype [10]. To ensure the interoperability of all functions, a communication matrix of all CAN messages is derived and given in parts to the appropriate supplier. Then, every supplier involved will start its own development cycle according to the V-model.[9] shown in Figure 4. The first integration attempt (Integration stage1) with the final ECUs at the vehicle manufacturers premises usually leads to a failure. As a rule, some suppliers have to redesign their system only to see it failing again at the next generation (Integration stage 2), which causes integration problem with another ECU. Further, an ECU centric approach prevents the reuse of functionality across different vehicle platforms. Failure in the integration phase exhibits timing problems, interface problems and communication problems between the ECUs.

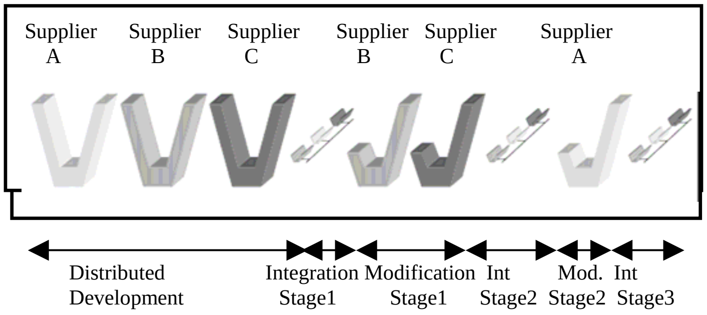

Figure4: Integration problem in ECU Networks

The only problem which can be addressed during the integration phase at CAN message level are communication problems. Interface as well as timing problems are associated to application design problems and can be sorted out much earlier in the design cycle. 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. AUTOSAR also specifies compatible software-interfaces at application-level.
The AUTOSAR partnership is an alliance in which the majority of OEMs, suppliers, tool providers and semiconductor companies work together to develop and establish a de-facto open industry standard for automotive E/E(Electrical/Electronic) architecture and to manage the growing E/E complexity. Its goal is the fulfillment of future vehicle requirements, such as availability and safety, software upgrades/updates and maintainability, as well as increased scalability and flexibility to integrate and transfer functions.
The AUTOSAR standard aims to prepare developers for upcoming technologies and to improve cost efficiency without compromising quality. The AUTOSAR partnership was set up with the objective of defining an open standard for an automotive E/E architecture. The major issues that need to be addressed are

- Managing the growing complexity of automotive E/E systems.
- Flexibility for product modification, upgrade and update.
- Scalability of solutions within and across product lines.
- Quality and reliability of E/E systems.

The standard for automotive software under development will be a first step towards tackling these problems. The concept for the standard is a layered software architecture with standard APIs. The standardization will apply not only to the software, but also to the whole development process from functional description to software testing. Introducing the AUTOSAR process would be a breakthrough in automotive E/E design. It would be a radical step and its repercussions would change the industry forever.[5]

With the increasing distribution of functions over several ECUs in a car, the importance of end-to-end timing (and deadlines) is also increasing. Industrial standardization efforts such as AUTOSAR have already defined models for capturing such “timing chains” composed of communicating “software components”, illustrated in Figure 5.[6] Similar models are known from data-flow theory, where clear semantics relate the execution of nodes (here:software components) with timing behavior of the stream.

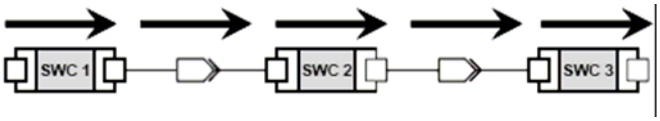

Figure 5: AUTOSAR View on “ Timing Chains”

The primary goal of AUTOSAR is to define a software infrastructure for application and basic software. Figure 6 illustrates the architecture of an AUTOSAR ECU.

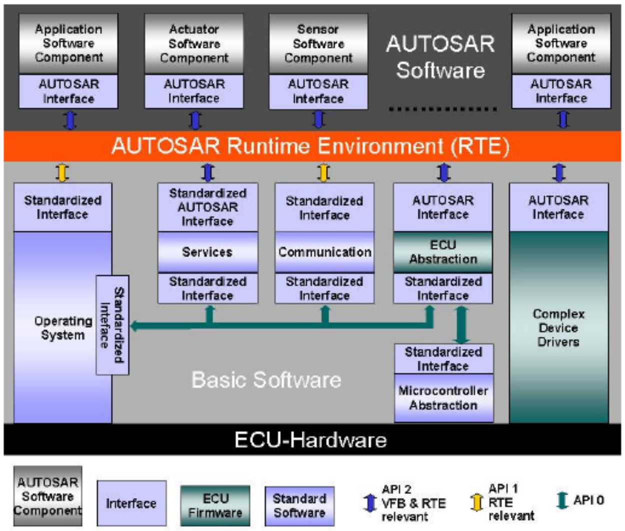

Figure 6:Architecture of an AUTOSAR ECU

The AUTOSAR software architecture can be subdivided into three layers:

- AUTOSAR software components (SWC)
- Run-time environment (RTE)
- Basic Software (operating system, communication, interfaces,...)

AUTOSAR provides a procedure model to generate the RTE automatically from descriptions, including SWC descriptions. At the same time the modules of the basic software are configured. There is no direct interaction between the SWCs and the basic software in the AUTOSAR architecture. Each SWC interacts with the RTE via standardized interfaces that the RTE provides for this. The RTE is scalable and is created statically for the applications of the particular ECU, thus allowing resources to be conserved. With regard to SWCs, the RTE interacts with the basic software via its standardized interface or arranges communication between different SWCs. In AUTOSAR, portability is built into the design.[16]

The goal is to be able to exchange parts of the system's software without rebuilding everything. This shall enable modularity, scalability, transferability and re-usability of software among projects, variants, suppliers, customers, etc. Hence, timing is not in the center of AUTOSAR but has later been recognized as an "important issue" that requires further consideration.[7]. AUTOSAR pushes the paradigm shift from an ECU based to a function based approach in automotive software development [13] which enables the exchange and reuse of ECU software components between vehicle manufacturers and suppliers as well as their us within different vehicle platforms and variants.

Benefits of AUTOSAR

- Increased reuse of software.
- Increased design flexibility.
- Clear design rules for integration.
- Reduction of costs for software development and service in the long term.
- OEM overlapping reuse of non-competitive software modules.
- Focus on protected, innovative and competitive functions.

VII CONCLUSION

In this paper, general information about model based approach for the integration of ECUs in automotive system is presented. Active research is going on in this field to reduce the complexity of the automotive system by reducing total number of ECUs. Embedding more functions in a single ECU will reduce the number of ECUs required. Mechanisms to improve fuel efficiency, reduce emissions, and optimize the cost and similar such mechanisms are finding their way into automotive subsystems.

REFERENCES

[1] Joachim Schlosser, **"**Requirements for Automotive System Engineering Tools", Proceedings of the 2002 IEEE International Conference on Computer Design: VLSI in Computers and Processors (ICCD'02)

[2] Jerry Krasner, "Model-Based Design and Beyond: Solutions for Today's Embedded Systems Requirements", EMBEDDED MARKET FORECASTERS, American Technology International. January 2004.

[3] Dr. Udo Brockmeyer, Guido Sandmann, Michael Beine, "Formal Verification Techniques in a Model- Based Development Process based on TargetLink generated C-Code", ERTS 2006.

[4] Jim Tung, The MathWorks, "Using model-based esign to test auto embedded software", EDA Design Line 2007.

[5] NEC Electronics Activities in Automotive: AUTOSAR | Web Magazine – Innovation Channel- | News & Events | NEC Electronics:Available: http://www.eu.necel.com/applications/automotive/index.html

[6] Dr. Kai Richter, Dr. Marek Jersak, Prof. Dr. Rolf Ernst, "Applying Real-Time Network Research in the Automotive Industry: Lessons Learned and Perspectives", Proceedings RTN'06.

[7] Autosar. Automotive Open System Architecture 2006 [cited 10/11/2006]; Available from: http://www.autosar.org/find02.php.

[8] IEC 61508:1998, Functional safety of E/E/PE safety- related systems. International electro technical Commission (IEC),Geneva, 1998.

[9] Schauffele J, Zurawka T, "Automotive Software Engineering, Vieweg Verlag, Wiesbaden, 2003.

[10] igel Tracey, Ulrich Lefarth, Hans-Jörg Wolff, Ulrich Freund, "ECU Software Module Development Process Changes in AUTOSAR".

[11] Christian Wewetzer, Klaus Lamberg ,Rainer Otterbach, "Creating Test Patterns for Model-based Development of Automotive Software", SAE International 2006.

[12] Paul F. Smith, Sameer M. Prabhu, Jonathon Friedman, "Best Practices for Establishing a Model-Based Design Culture", The Mathworks Inc.2007.

[13] Proceedings from The MathWorks International Automotive Conferences: www.mathworks.com/industries/auto/iac06/presenations

[14] Virtual Prototyping boosts Model Driven Design for Six Sigma. Available: www.Automotivedesignline.com

[15] Markus Maier (ETAS GmbH), "Impacts auf AUTOSAR on the future development of ECU Software".

[16] Thierry Rolina, "Past, Present, and Future of Real-Time Embedded Automotive Software: A Close look at basic concepts of AUTOSAR", SAE World Congress 2005.