

HW/SW Codesign for Automotive Applications: Challenges on the Architecture Level

Jakob Axelsson
Carlstedt Research & Technology AB
SE-411 21 Göteborg, SWEDEN
E-mail: jakob.axelsson@crt.se

Abstract

The latest generations of road vehicles have seen a tremendous development in onboard electronic systems, which control increasingly large parts of the vehicle's functionality. HW/SW codesign techniques seem very useful for designing the nodes of the onboard computer network, and it is tempting also to apply it to the level of the overall electronic architecture. This paper discusses the differences and similarities of codesign on the node level and architecture level, and identifies some reasons why traditional assumptions of HW/SW codesign will not work when developing the complete architecture. It also outlines a suitable approach for the architecture level, based on techniques from the area of systems engineering.

1. Introduction

The automotive industry is currently going through a dramatic increase of electronic equipment for on-board vehicle control. In modern cars, an advanced distribution network connects tens of electronic control units (ECUs) implementing sophisticated real-time control functionality.

The ECUs often have very tight cost margins, and at the same time demanding requirements on e.g. performance and reliability. They are produced in large volumes, and it is therefore often cost-effective to seek special-purpose solutions. This makes HW/SW codesign (hereafter simply codesign, for short) attractive since it can help the designers find better trade-offs between HW and SW.

Most of the research in codesign is focusing on the ECU level, but there are also important trade-offs to be made at the level of the complete distributed architecture. This paper discusses what the similarities and differences are when working on the architecture level and the ECU level of a distributed real-time system. First, a short review is given of some research trends in codesign, followed by an introduction to the current and future characteristics of automotive electronic systems. Then, in Section 4, the comparison between the architecture and component level is made, and

in the last section, the conclusions are summarized.

2. HW/SW codesign

HW/SW codesign is a systematic technique for synthesizing designs consisting of both SW, running on general-purpose microprocessors, and HW in the form of application-specific integrated circuits (ASICs). The area started attracting attention in the early 1990s, when tools for automatically synthesizing ASICs from high-level behavioural descriptions became useful. This meant that the ASIC design process approached the SW design process, where high-level languages and compilers are used. It then became interesting to investigate whether the same behavioural input description could be used to generate both HW and SW, and to find ways of partitioning the description automatically into HW and SW parts, according to some optimization criterion.

In the early days, much focus was on designs consisting of one ASIC and one pre-selected microprocessor, with the goal of partitioning the behaviour onto that architecture in such a way that a maximum acceleration of the SW was achieved. Later, more flexibility in the choice of overall design was allowed, and other optimization criteria were considered. Also, systems-on-chip, where the microprocessor was included in the ASIC as a synthesizable core, attracted a lot of attention. In this perspective, the possibility to reuse functionality from external suppliers, in the form of "intellectual property" (IP), becomes important, entailing both architectural and methodological considerations.

For real-time systems, the aim is not to achieve a maximal acceleration of the SW, but rather to find an overall solution that meets performance requirements (and other constraints) at a minimal cost. In [1], an approach is described that investigates ways of choosing the overall design and partitioning the behavioural description based on schedulability of real-time tasks.

When studying the applicability of codesign, it is important to be aware of the approach's underlying assumptions:

- A detailed high-level description of the complete behaviour is made prior to design selection and HW/SW partitioning.
- Accurate estimations of important characteristics, such as cost and performance, of an implementation are made based on the behavioural description and these estimations serve as input to the optimization procedure used to select the overall design and partitioning.
- Behavioural descriptions are written independently of the intended implementation, but are still assumed to be synthesizable into an optimal implementation.
- The interfaces to the system's environment are known in detail, and stable.

Essentially, these assumptions mean that codesign has to be applied fairly late in the overall system development process, since in practice a sufficiently detailed behavioural description is not available earlier. The interface issue is often tricky for embedded control systems, involving sensors and actuators, and redesigns due to lack of information about the environment are not uncommon. Codesign also does not deal with the design of the boards and casings of the physical ECUs and these issues often add to the lead time and the cost of the ECU. Thus, for embedded automotive systems, codesign cannot be restricted to HW and SW, but must also include the mechatronics. An object-oriented (OO) approach to such multidisciplinary modelling is described in [4].

3. Automotive electronics

The electronic system in a vehicle is closely connected to the vehicle itself, and it is important to bear this in mind. Ultimately, the requirements on the electronics are derived from the requirements on the complete vehicle, since the electronics are never used outside that context.

3.1. Vehicle requirements

The mission of the vehicle is to transport passengers and goods safely and efficiently. This means that the following factors are of key importance:

- **Cost.** The total life-cycle cost should be minimized.
- **Safety.** The vehicle should minimize the risks both to the persons travelling in it and to the surrounding traffic.
- **Weight.** The vehicle should be as light as possible to

reduce the fuel consumption.

- **Availability.** The vehicle should be available to the customer as much as possible.
- **Environment.** The negative influence on the environment should be minimized.
- **Comfort.** The passengers should have an enjoyable environment inside the vehicle, in terms of a smooth ride, an agreeable climate, a low noise level, etc.
- **Information.** The driver and passengers should have access to all the information they need.

Given that the electronics implement a larger and larger portion of the vehicle functionality, the connection to the complete vehicle becomes an increasingly important issue.

3.2. Current systems

Until recently, the electrical system in a vehicle was mostly concerned with lights, starter motors, windshield wipers, and the like. The revolutionary change came when developers began to use electronics to attack the requirements stated in the previous section. Electronic control units were first introduced in the engine management system, to better control the fuel injection in order to reduce fuel consumption (and thus cost) and pollution. Also, anti-lock brakes (ABS) were introduced to increase safety, which led to the need for electronic control. On the comfort side, automatic climate control units are used to maintain a constant temperature.

The early ECUs were fairly autonomous, exchanging very little information with each other. However, as the number of ECUs started to increase, it became possible to implement advanced functions that require the cooperation between multiple units. This made the interconnections between them more complex and added significantly to the weight of the vehicle, and therefore multiplex networks were introduced. Current vehicles usually contain up to about 20 ECUs that are placed as closely as possible to the parts they control, and that are connected to each other via a few distribution buses. Figure 1 shows a typical electronics architecture in a contemporary vehicle. The networks usually run the CAN protocol [5], with speeds up to 250 kbit/s.

3.3. Future development

Among the current trends, it is interesting to notice that the drive to replace or enhance traditional mechanical solutions with electronics continues. This includes safety systems,

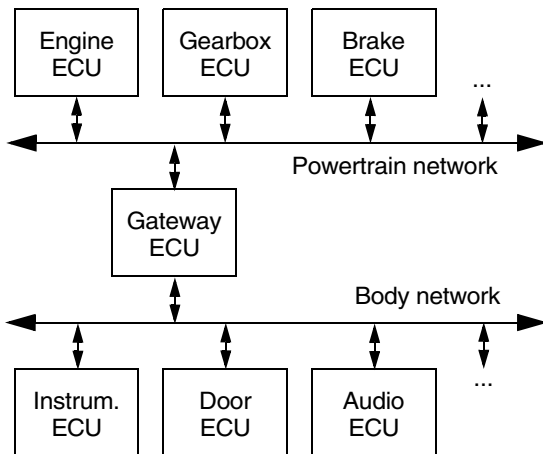


Figure 1. Sketch of a typical automotive electronics architecture.

such as stability control, but also fundamental vehicle functionality such as brakes and steering, where electronics can provide increased functionality and at the same time reduce cost and weight, compared to current technology. However, new electronic systems, such as fault-tolerant buses, are likely to be needed to reach sufficient safety levels for these crucial functions. To reduce service times and unscheduled stops, improved vehicle diagnostics is also necessary.

Another area where large changes are anticipated is in information to the driver and passengers, and also in entertainment functionality. This will affect the technology used in the electronic systems since current networks have insufficient bandwidth. Most likely, fibre optical solutions will be required to reach speeds in the Mbit range.

3.4. System integrators and suppliers

In the automotive industry, there is a long tradition to work with suppliers, and a number of suppliers exist that are comparable in size to the vehicle manufacturers themselves and deliver similar systems to all of them, thereby reducing the cost. The role of the vehicle manufacturer is then to specify

the subsystems, and integrate them in the vehicle.

So far, the suppliers have mostly developed isolated subsystems, e.g. consisting of an ECU, its SW, and possibly mechanical parts, too. However, in the future it is expected that the number of ECUs will not grow drastically, simply because it becomes too expensive and heavy to have separate HW for each functionality, and the available space in the vehicle is also limited. Therefore, ways must be found to integrate SW modules that participate in the implementation of different functionality in the same ECU. The SW modules may be delivered by different suppliers, and the integration must ensure the integrity and performance of all the functions despite the sharing of HW resources. This is a particular challenge when it comes to safety-critical functionality, and the requirements on the SW modules are thus not limited to their functional interface, but touch their overall behaviour.

4. Architectures vs. components

The future of automotive system integration, where HW and behaviour are developed separately is very similar to the ideas of IP mentioned in Section 2, and many of the issues on how to handle IP in systems-on-chip also apply to the automotive systems. One issue is how to trade IP in such a way that a reasonable compromise is found between the supplier's need to protect the information about the design of the IP, and the integrator's need to access the information to ensure the functionality of the entire system.

4.1. Integration structure

To handle development which relies heavily on the use of IP, it is necessary to have a generic system structure which determines how the IPs are interfacing to each other, and how resources are shared between them. For automotive systems, a layered architecture similar to the one in Figure 2 is suitable. The functional layer is likely to be specified by the vehicle manufacturer, since it defines the functionality of the vehicle, and suitable OO methods for this task are described in [2]. The specification is partitioned into behav-

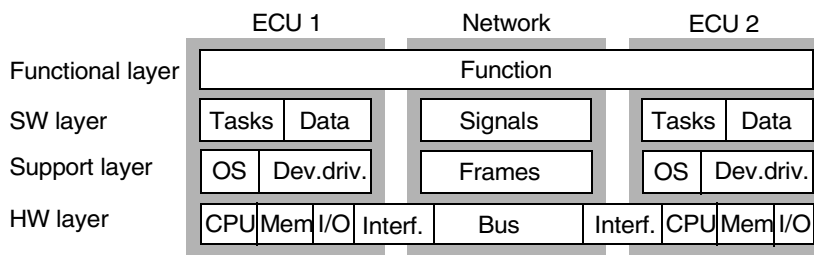


Figure 2. A layered view on the implementation of an automotive function.

behavioural modules that are allocated to different ECUs. The behavioural modules are refined by the suppliers, resulting in SW or synthesizable blocks that can be integrated in ASICs, and the ECU HW is developed by, possibly, other suppliers. The vehicle manufacturer integrates the SW modules, and to facilitate the integration, a standardized support layer is used, which hides the details of the HW from the SW modules, and defines interfaces for intermodule communication. The communication works equally well internally in an ECU and externally, over the network, and it is the role of the vehicle manufacturer to maintain the signal information.

The suppliers are likely to reuse IP components between systems delivered to different vehicle manufacturers, which reduces the development cost for everyone. To make this possible, the support layer must be standardized, and such efforts are under way with OSEK [7], which covers operating system, network communication, etc. Of course, nothing will prevent the supplier from applying codesign to the ECU, to put parts of its behaviour in ASICs and thereby reduce cost or increase performance. However, this will require a similar support structure, in the form of e.g. synthesizable VHDL code.

4.2. Systems engineering

To turn into efficient system integrators, it is clear that the vehicle manufacturers must become better at specifying the functionality of the system, and at making trade-offs in order to find the best system architecture to suit the needs expressed in the requirements. This challenge is met by improving the systems engineering (SE) capability [6], since SE addresses exactly these issues. The SE teams must be cross-disciplinary, covering many aspects of the system simultaneously, since the electronics are so tightly integrated with the rest of the vehicle.

In a sense, codesign is SE specialized to a particular class of problems, and the two disciplines share the view that the overall functionality of the system should be studied before selecting the solution. However, on the electronic architecture level, one cannot make the assumption that the behaviour is completely specified, since the architecture is decided at an early stage of the complete vehicle project. Rather, an informal functional specification needs to be developed, and partitioned into parts that can be handed out to different suppliers for implementation. The informal nature of the specification means that very accurate estimations of cost and performance are impossible, and less detailed methods such as those described in [3] must be applied instead.

This also means that automatic synthesis methods are less useful. Instead, means must be found that allow the system architects to rapidly develop the architecture using their

creativity. Thus, tools for handling the complexity and interactions between all the involved parties are needed.

5. Conclusions

In this paper, we have compared HW/SW codesign with architecture level systems engineering for automotive real-time systems. There are many similarities, in particular the holistic philosophy that underlies both approaches, and the ideas about reusing functionality in the form of IP, with the resulting architectural and methodological implications. However, there are also some fundamental differences:

- It is not feasible to assume that a complete behavioural description is available at the time of architecture selection.
- Therefore, only very rough estimations of performance and cost are possible.
- Automatic methods are hence also less useful.

By merging the process ideas of SE with OO description techniques and tool support inspired by codesign, an efficient development environment for the early architecture selection phases is possible. Such an environment must be based on a combination of the designer's creativity and automatic methods for analysis and synthesis. A predefined layered structure is necessary to help the designer in this task.

References

- [1] Axelsson, J. *Analysis and Synthesis of Heterogeneous Real-Time Systems*. PhD thesis, Linköping University, 1997.
- [2] Axelsson, J. Holistic Object-Oriented Modelling of Distributed Automotive Real-Time Control Applications. In *Proc. 2nd Intl. Symposium on Object-Oriented Real-Time Distributed Computing*, 1999.
- [3] Axelsson, J. Cost Models for Electronic Architecture Trade Studies. In *Proc. 6th Intl. Conf. on Engineering of Complex Computer Systems*, 2000.
- [4] Axelsson, J. Unified Modeling of Real-Time Control Systems and their Physical Environments Using UML. In *Proc. 8th Intl. Conf. on the Engineering of Computer Based Systems*, 2001.
- [5] Bosch. *CAN Specification*. Version 2.0. Robert Bosch GmbH, 1991.
- [6] IEEE Standard for Application and Management of the Systems Engineering Process. IEEE Std. 1220, 1998.
- [7] Kiencke, U. and D. John. On the way to an international standard for automotive applications - OSEK/VDX. *Proc. Intl. Congress on Transportation Electronics (Convergence)*, pp. 95-100, 1998.