Design Process of Embedded Automotive Systems — Using Model Checking for Correct Specifications

Peter Jansen

BMW AG, 80788 München, Germany
Peter.Jansen@bmw.de

Abstract. The number of Embedded Control Units (ECUs) in the car is permanently increasing. Also complexity and interconnection is increased. Conventional design processes can not cope with this complexity. In the first part of this paper we show the current development-process at BMW, the second part deals with our results of using model checking to verify STATEMATE-models.

1 Introduction

The increasing demand for dynamically controlled safety features, driving comfort and operational convenience in cars require an intensive use of Embedded Control Units (ECUs). The number of ECUs in the car increases permanently. Also complexity and interconnection increase rapidly. Customary design processes can not cope with this complexity. Systems Engineering at BMW describes the process for transforming a product idea into a real system using (semi-)formal methods for specification, analysis, rapid prototyping and support for test and diagnostics. The use of CASE-Tools for designing ECUs is such a semiformal method. It can also help to reduce the development time and costs. Another advantage of using CASE-Tools is the possibility of detection of errors in an early phase of the development process. However, the use of CASE-Tools alone can not guarantee safety-critical properties of the system. New techniques are necessary. Model checking is such a technique. It is an automatic method for proving that an application satisfies its specification as represented by a temporal-logic formula. It offers a mathematical rigid proof. In contrast to *testing* a system, *model checking* allows to check the system under all possible inputs, a test can only check a limited set of inputs. For large systems, an exhaustive test may not be feasible.

2 The Design-Process for Embedded Control Units Software

There is a large amount of ECUs in today's high-end vehicles. Fig. 1 shows some of them, e.g. an engine control or a park-distance-control. All of these systems

L. Pierre and T. Kropf (Eds.): CHARME'99, LNCS 1703, pp. 2–7, 1999.

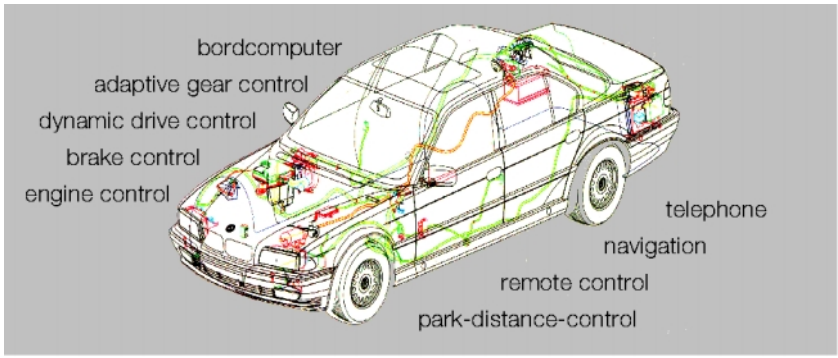

Fig. 1. View of electronic components in a car

are connected via several bus systems. Also the responsibility of these systems and with this the need for reliable systems. As shown in Fig. 2, tomorrow's ECUs could take full command of the car.

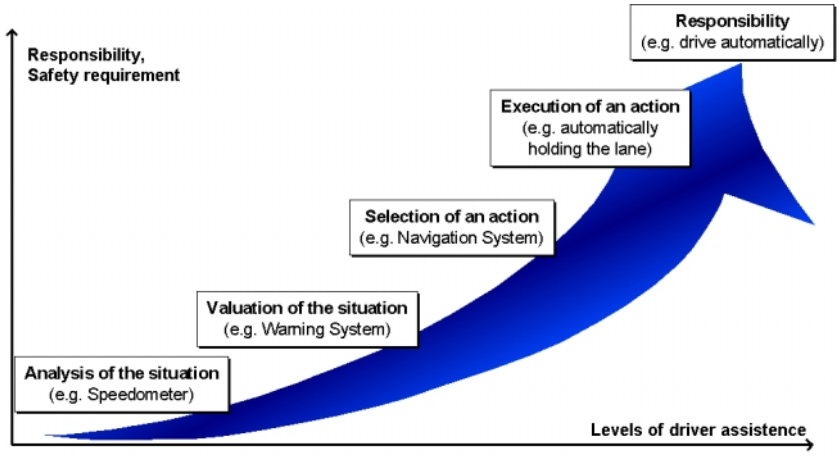

Fig. 2. Increasing responsibility of ECUs in automotive

To develop such systems, traditional design methods are inadequate. New methods, such as the use of CASE-Tools, are needed to create safe systems. Fig. 3 gives an overview on some of the CASE-Tools used at BMW. One such tool, used in an early phase of the development process is STATEMATE[1]. It is used to design

[1] STATEMATE is a registered trademark of i-Logix, Inc.

state-based systems. With STATEMATE the user can design and simulate the system. STATEMATE also offers the ability to generate C-code. A tool-set to verify STATEMATE-designs has been developed by OFFIS (Oldenburger Forschungs- und Entwicklungsinstitut für Informatik-Werkzeuge und -Systeme, Oldenburg, Germany) and used for two years at BMW. One problem with STATEMATE is that its generated code is too inefficient to be used with micro-controllers (e.g. a Siemens C167). Therefore, the STATEMATE-Design has to be coded by hand. This is usually done by a supplier like Bosch or Siemens. For the supplier the STATEMATE-Design becomes a specification.

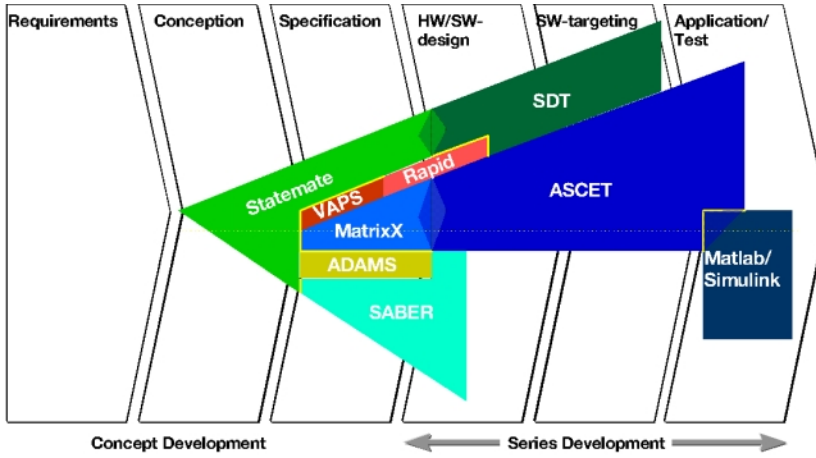

Fig. 3. Overview on CASE-Tools

3 Using Model Checking to Verify Statemate Designs

Model checking is an automatic method for proving that an application satisfies its specification represented by a temporal-logic formula. This offers a mathematical rigid proof. In contrast to testing a system, model checking allows to check the system under *all* possible inputs. No test can do this, because a test can only check a limited set of inputs. A tool-set for automatic verification of STATEMATE-Designs has been developed by OFFIS. With this tool-set it is possible to model check STATEMATE designs. For this a STATEMATE-design has to be translated into a mathematically equivalent model, called FSM (Finite State Machine). Another specification for this design has to be done in temporal-logic formulas. For this, a graphical formalism, called Symbolic Timing Diagrams (STD), has been developed. These diagrams represent an easy to understand way to express properties of a reactive system. The Siemens Model-checker SVE finally verifies whether the STATEMATE design satisfies its specification. Fig. 4 shows the

tool-environment. Model checking assists the engineer in creating software for controlling systems in an early stage of development, so cost for the development can be reduced, while making the software "safe".

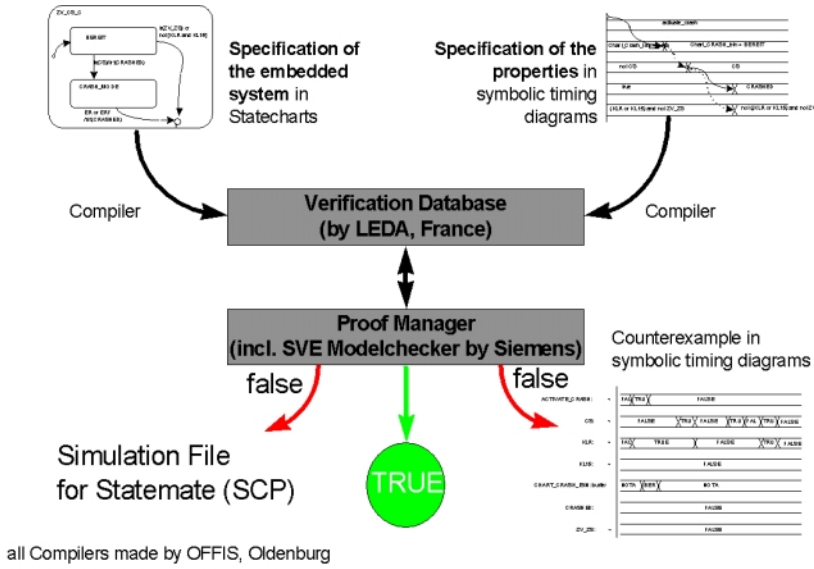

Fig. 4. Overview of CASE-Tools

3.1 Example "Electronic Brake Management"

A recently developed new feature of the brake management is a highly safety-critical application. Roughly, the main function is the following: If the driver stops at a red traffic light, he does not have to push the brake-pedal any longer; the brake is being held automatically. Fig. 5 shows the top-level activity-chart of the STATEMATE-design. There is a switch in the car to disable this function. One property for this system is: Whenever the "hold-function" is enabled and the switch is pressed, the hold-function has to be cancelled immediately. With a normal test it is not possible to guarantee this property: It can only be checked, whether the function is cancelled in *some* situations. Model-checking can guarantee that this function is cancelled in *all* situations. Fig. 6 shows the symbolic timing diagram for this property. The hold-function is represented by the variable *DECELERATION*. The value 10.0 means minimum deceleration, 0 would mean maximum deceleration.

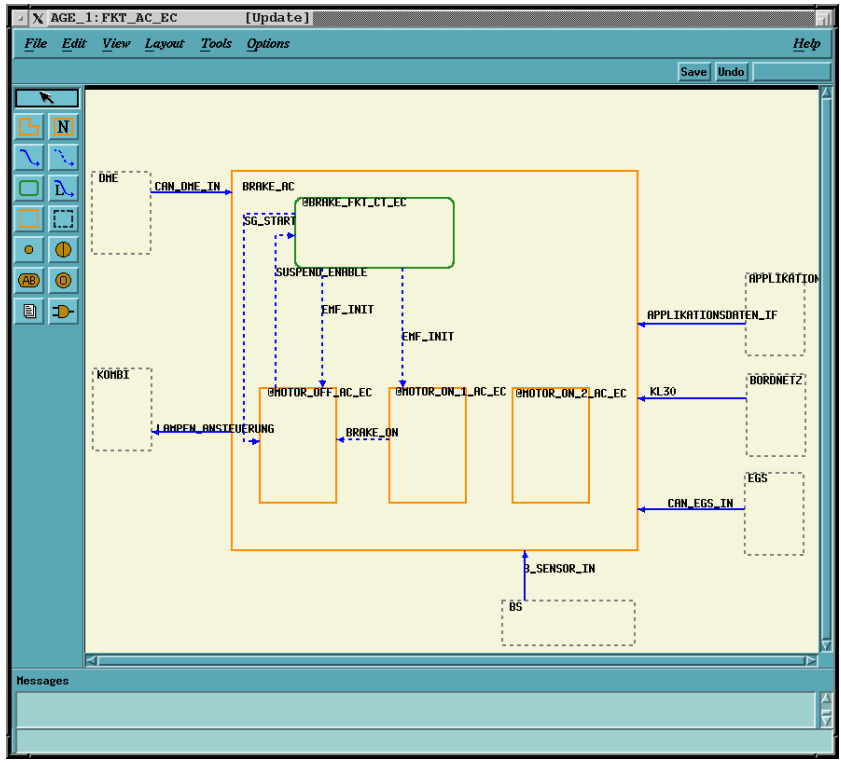

Fig. 5. Top-level activity of the system

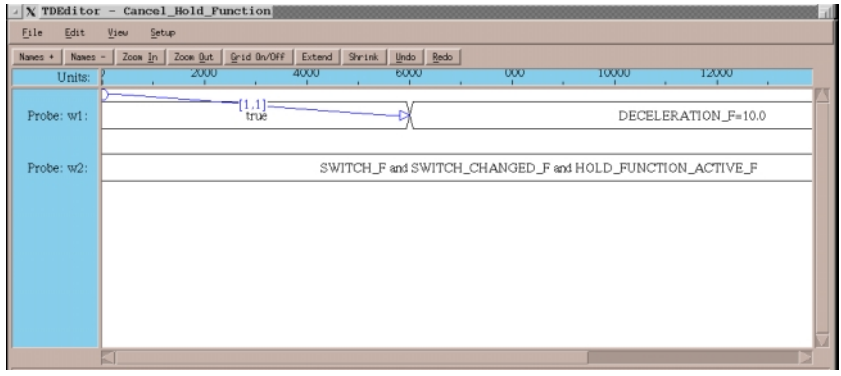

Fig. 6. The property for the brake system as a symbolic timing diagram.

4 Further Information

More information on CASE-Tools and Model-Checking can be found on the following web sites:

- Information on STATEMATE
  http://www.ilogix.com
- Information on the model-checking tool-set
  http://ca.informatik.uni-oldenburg.de/publications/publications.html
- Information on BMW
  http://www.bmw.com