98P-178 (49)

A New Approach to Functional and Software Structure for Engine Management Systems - BOSCH ME7

J. Gerhardt, H. Hönninger, and H. Bischof
Robert Bosch GmbH

ABSTRACT

This paper describes the new Engine Management System (EMS) ME7.

Torque and A/F demands for modern EMS result from both, internal functions (i.e. engine start, idle speed control, catalyst heating) and external systems (i.e. driver's request, transmission or vehicle dynamic control). With ME7 these demands are processed to the optimized actions of the actuators by a centrally coordinated torque and A/F management. The design of the functions is physically based to provide optimum portability and minimum calibration time. Examples are given for the physical manifold pressure model and the cylinder charge control of ME7 with electronic throttle control.

The real time operating system „ERCOS" and a layer based software architecture enable the implementation of these functions in a flexible family of products for current and future systems.

Topics, such as warm-up strategies for catalysts in conventional port injection systems, gasoline direct injection systems (with their switch-over strategies between stoichiometric and stratified operation), NOx catalyst control, and the requirements of future integrated drive train management systems, all require maximum flexibility and expandability.

The introduction of the ME7 is an important step towards this future. The design represents a good basis for development sharing with customers and is also an important prerequisite for the vehicle management system CARTRONIC®.

1. INTRODUCTION

The functional structure of engine management systems has evolved over several years [Ref. 1]. Starting with a simple injection system with a separate ignition unit in the early 70's, injection and ignition were integrated into one single electronic control unit during the 80's. A modern EMS is comprised of a large number of subsystems, and not only controls basic EMS-functions such as injection and ignition timing or emission control (i.e. Lambda closed loop control or catalyst heating) but also manages additional functions, such as continuous camshaft control, resonance flap actuation or engine fan control. A modern EMS must also be equipped with a complete on-board diagnostic and monitoring system.

The introduction of electronic throttle control (ETC) as a drive-by-wire system with it's adjustable relationship between the pedal position and throttle position enables the EMS to now control all torque-influencing outputs over the entire operating range of the engine. With stand alone ETC systems, mutual functional impacts have to be considered, such as idle speed control which must be divided into the two subsystems. The fully integrated system with control of injection, ignition and cylinder charge can eliminate this drawback but then a complete redesign of the entire system is required.

The new functional architecture of the ME7 system is characterized by the following main features:

- Centrally coordinated torque management:
  The engine torque represents the central system variable. All torque requirements derived from EMS internal functions or external systems (i.e. drive train or vehicle dynamic control) result in a variation of torque or efficiency and are defined on this basis.
- Centrally coordinated A/F management:
  Similarly, all mixture demands are coordinated in one central manager. Based on the operating conditions, a set of basic functions controls the A/F ratio within the physical limits defined by the flammability of the mixture.
- Subsystems based on physical models with physically defined interfaces:
  The use of physically based functions improves the transparency of the system's architecture. Computed values can be directly compared with physically measurable values.

Using physically based functions in combination with a centrally coordinated torque and A/F management allows for an improved handling of function variants. Due to their relationship to the physical structure, single functions as well as functionally linked groups of functions (subsystems) could be easily compared with customer's requests using physically measured values. Therefore a set of basic platform functions was realized and applied over the entire EMS family.

The realization of an appropriate Software structure guarantees the system's modularity which allows different customer's requests to be met as well as

future challenges. By means of the torque based functional structure the software implementation is independent of demands generated by external systems. The real time operating system „ERCOS“ and a layer based software architecture enable a system evolution with optimum portability at a high level for future microcontrollers. Most of the functions are realized in the programming language ANSI C, to provide good modularity and integration of customer specific functions.

2. A NEW FUNCTIONAL AND SOFTWARE STRUCTURE FOR ENGINE MANAGEMENT SYSTEMS

2.1 ME7 SYSTEM OVERVIEW

2.1.1 System Configuration

Figure 1 shows an ME7 system overview with the main sensors and actuators. In addition to the components of a conventional EMS, the system is comprised of the ETC related elements, which include the accelerator pedal module to interpret the driver's request, the throttle actuator for cylinder charge control and the cruise control lever.

2.1.2 Brief Functional Overview

The ME7 contains all functions to control a modern SI-engine. In this section only a brief functional overview is given. Due to the system's modularity very different system configurations can be realized. For example systems with different sensors for cylinder charge determination (air mass or speed density), naturally aspirated or turbocharged engines, engines with or without EGR and engines with variable camshaft actuation are possible. The main system features are as follows:

- The engine torque management which controls all torque influencing actuators (see also section 'Functional Structure').
- A/F ratio control with a central A/F manager, λ-pilot control, λ-closed loop control, or alternatively with a Nernst or universal λ-sensor and trim control (details see section 'Functional Structure').
- Sequential, cylinder individual fuel injection.
- Ignition timing, including control of dwell angle and ignition angle.
- Cylinder individual knock control.
- Emission control functions for optimized emissions during cranking, start and after start which enable the realization of different catalyst warm-up strategies, using a lean mixture or a rich mixture

Figure 1: Engine Management System ME7

including exhaust gas recirculation (EGR) and secondary air injection (SAI) control if necessary [Ref. 2], [Ref. 3].
- Canister purge control based on canister charge.
- Idle speed control.
- Diagnostic and monitoring functions:
- The system is comprised of the complete OBD II functionality to meet both MY '98 and future EOBD requirements. A torque-based monitoring systems supervises the throttle control under all operating conditions and reacts with the appropriate limp-home functionality in case of a failure.
- To communicate with external systems, such as a transmission control system or a vehicle dynamic control system, torque demands can be received via a torque interface, realized via CAN. Therefore the EMS is able to process external torque demands within the torque manager. (see also section 'Functional Structure').
- Conventional or continuous camshaft control.
- Resonance flap actuation.
- Engine fan control.
- Control of air-conditioner (A/C).
- Cruise control.
- The system contains the necessary interfaces to application tools, end of line programming tools, service and SCAN-tools.
- Immobilizer.
- Additional customer defined functions as required.

2.2 FUNCTIONAL STRUCTURE

Essential functional features of the ME7 structure are the central torque and A/F management and the use of physically-based functions.

2.2.1 Torque-based system structure

2.2.1.1 Structural approach

Choosing a torque-based system architecture was initiated by the followings concerns when the previous situation was analyzed [Ref. 4], (Figure 2)

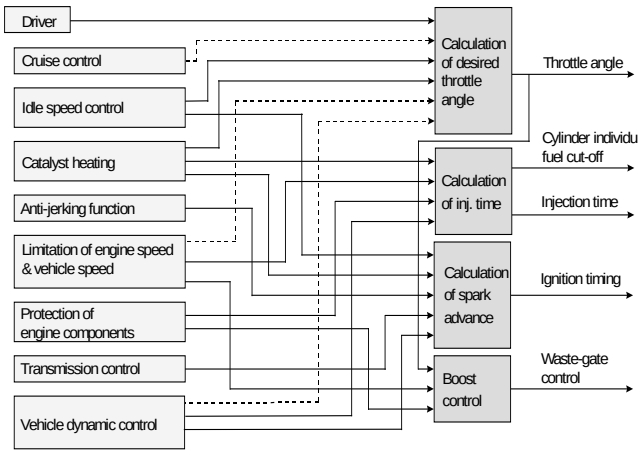


Figure 2: Influences on Engine Torque
- previous situation

In the case of several torque or efficiency demands, derived simultaneously from different subsystems, there was no central torque coordination. This meant, that subsystems inside the EMS as well as external systems directly required, for example, a reduction of the throttle or an ignition retard to obtain a certain torque reduction. The priority of each demand had to be defined independently in each subsystem. This lack of a central coordination caused interactions of different demands (due to shifts of operation points) resulting in a strong interdependence of calibration data of the different subsystems (i.e. calibration of ignition timing influenced idle speed pilot control).

On the contrary the functional structure of the ME7 system is characterized by two coordination steps (Figure 3).

- Torque demand manager:
  Input values for the torque demand manager are all internal and external requirements which can be defined as a torque or efficiency value. Internal demands are for example generated by the start function, idle speed control, engine speed limitation, as well as engine protection functions or catalyst heating. External torque demands are defined by the driver, cruise control, or vehicle dynamic control. The major task of the torque demand manager is the priority handling which is processed by a minimum/maximum selection.
- In the next step the resulting torque demands are processed in driveability functions (mostly filtering or slope limiting functions), dashpot function (to limit the minimal intake manifold pressure) and the anti-jerking function. The driveability functions, due to customer's requests, allow calibrations over a wide range of applications. The calibration can vary between a comfortable and a sportive characteristic.

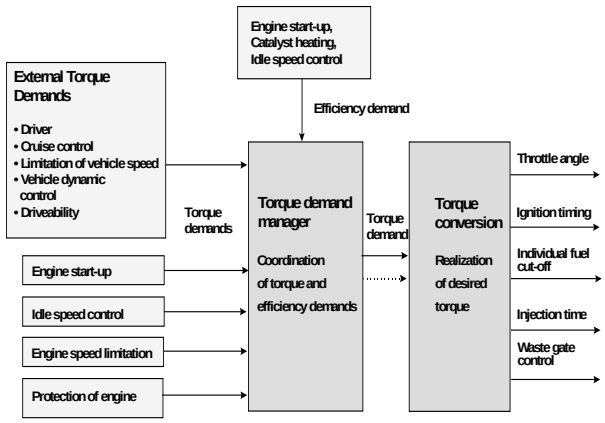


Figure 3:Influencing Engine Torque -
Situation with Torque-Based System Structure

The output value of this management block is a resulting torque demand, taking the required efficiency into consideration, which is equal to 1.0 during normal operation and which can be reduced for example during catalyst heating.
- Torque converter:

In a second step the resulting torque demand is converted into the available controller outputs which are able to adjust the engine output torque. These controller outputs are the throttle angle, ignition timing and injection timing (including a cylinder individual fuel cut-off) supplemented by a waste-gate control in the case of turbocharged engine.

This simple structure can be extended in case of highly dynamic torque demands resulting for example from vehicle dynamic control. In this case two resulting torque demands are defined:

- The long-term torque demand has to be realized via a variation of the cylinder charge. This demand results in a variation of the throttle position and the waste-gate opening, which means that it's dynamic behavior is limited for example by the regulating speed of the throttle actuator and the time constant of the intake manifold which can amount to several 100 ms at low engine speeds.
- A separate short-term demand is realized via the crank synchronous controller outputs (injection and ignition timing) which enable a modification of engine torque within the following combustion cycle. With the separate short-term component all highly dynamic torque demands, which only last some 100 ms can be realized without affecting the slow cylinder charge path. Examples for highly dynamic demands are the torque reduction generated in a transmission control for a comfortable gear shifting or a quick and short demand when the vehicle dynamic control is active.

2.2.1.2 Internal torque model

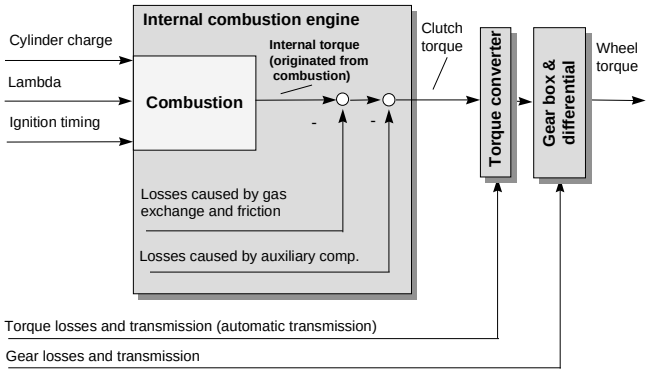


Figure 4: Torque definitions concerning the drive train

Figure 4 illustrates different torque values related to a drive train. Influenced by the major input variables

- relative cylinder charge (fresh air mass per stroke),
- Lambda (A/F ratio related to a stoichiometric ratio) and
- ignition timing

the combustion generates an internal torque „tq_i", which does not yet consider losses caused by the gas exchange. Therefore „tq_i" differs from the value of indicated torque represented by the work integral from the entire 4-stroke process. Subtraction of losses derived from gas exchange and friction leads to the engine output torque. The clutch torque results when taking into account the torque which is necessary to drive auxiliary components such as the alternator, the steering pump or the A/C.

Taking the torque losses and the gear ratio inside the torque converter, the gear box and the differential into consideration, wheel torque results, which represents the available torque for vehicle motion.

Due to the complexity of the combustion process in an SI-engine the generation of internal torque is described by an empirical approach with physical intermediate and interface values which take the limited computation resources of a mass production ECU into account.

The necessity to determine the controller outputs based on the required resulting torque means, that the inverse calculation of the torque model must be possible.

The basic structure of the model, used for calculating the internal torque value „tq_i" is shown in Figure 5.

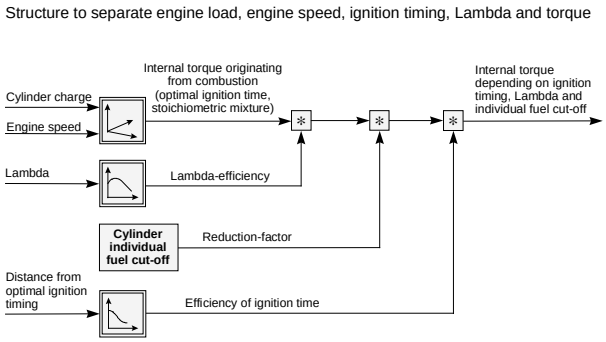


Figure 5: Simplified Torque Model

In principal, the comprehensive dependency of internal torque on cylinder charge, engine speed, Lambda, ignition timing and cylinder individual fuel cut-off could be described in a five dimension map. The decisive step to simplify this dependency is the introduction of two central reference values:

- the optimal spark advance „sa_opt" and
- the corresponding optimal internal torque „tqi_opt", which reaches it's maximum value at optimal spark advance.

In some operating points the optimal spark advance is a theoretical value, because of the engine knock limit. Both reference values refer to Lambda equal to 1.0 (→ „sa_opt_l1" and „tqi_opt_l1") and are defined by 2-dimensional look-up tables:

$$sa\_opt\_l1 = fn.(rc, n\_eng) \quad (1)$$

$$tqi\_opt\_l1 = fn.(rc, n\_eng) \quad (2)$$

Relative cylinder air charge „rc“ refers to a 100% value defined by the displacement per cylinder and the standard air density. The second influencing variable is the engine speed „n_eng“.

The actual torque value „tqi“ is the result of a multiplication with Lambda- and spark advance efficiencies

$$eff\_lam = fn.(lam) \quad (3)$$

$$eff\_sa = fn.(d\_sa) \quad (4)$$

and the reduction factor „eff_red“ caused by a cylinder individual fuel cut-off:

$$tqi = tqi\_opt\_l1 * eff\_lam * eff\_red * eff\_sa \quad (5)$$

In equations 3 through 5 „lam“ represents Lambda. For simplification of the basic equation (equation 5), spark advance efficiency is defined depending on the difference between actual spark advance „sa“ and the optimal spark advance:

$$d\_sa = sa\_opt - sa \quad (6)$$

2.2.1.3 Identification of model parameters

Equation (5) describes the decoupling of the five-dimensional relationship between the internal torque and it’s influencing variables by introduction of the optimal internal torque and the optimal spark advance.

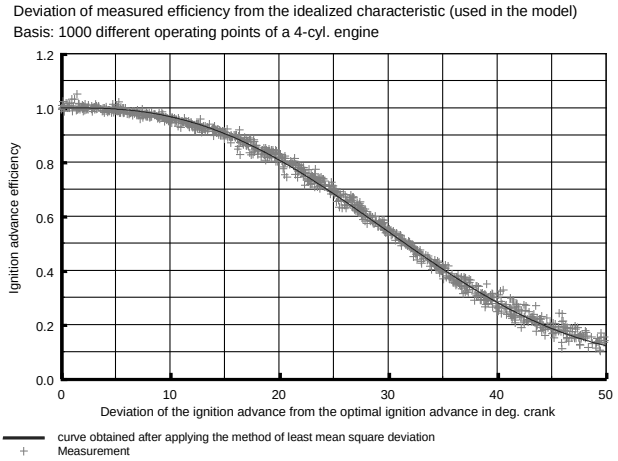


Figure 6: Resulting Error of Spark Advance Efficiency

Figure 6 shows the dependency of the spark advance efficiency „eff_sa“ on the difference „d_sa“ between optimal and actual spark advance measured at 1000 operating points of a 4-cylinder engine, representing the total operating range (in engine speed and load).

In Figure 6 a characteristic line is also shown, as a result of a least-square-optimization [Ref. 5]. The possibility to reduce the dependency of spark advance efficiency on „d_sa“ to one single characteristic line is confirmed by applications of this structure on several engines with a wide range of cylinder numbers, engine displacements and combustion chamber designs. This characteristic is related therefore to the engine design only and not to an operating point.

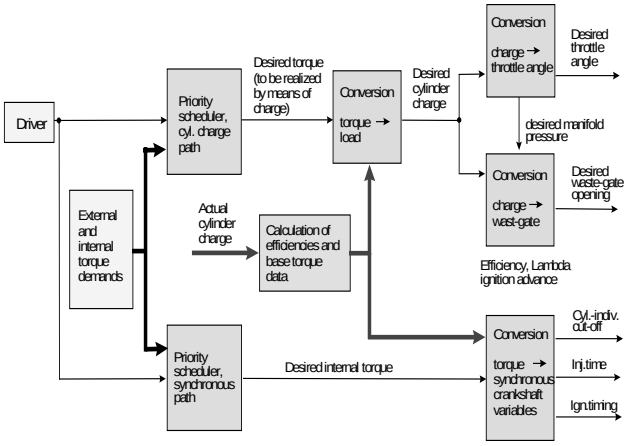
Identifying the parameters of the model for a certain engine only means measuring „tqi_opt_l1“ and the dependency of „tqi“ on spark advance and Lambda. The numeric optimization combined with a suitable control of an engine dyno allows for an automatic identification of the model parameters, which are shown in Figure 5.

2.2.1.4 Calculation of the desired values

As previously mentioned, the torque model is not only used to determine the actual value of the internal torque. The basic equation (equation 5) can also deliver the desired values of the controller outputs:

$$tqi\_tar = tqi\_opt\_l1(rc\_tar, n\_eng) * eff\_lam\_tar * eff\_red\_tar * eff\_sa\_tar \quad (6)$$

Figure 7: Determination of Torque Influencing Target Values



The target torque value „tqi_tar“ is calculated by multiplication of the optimal torque at lamba = 1.0 and optimal spark advance by the efficiencies. Solving equation (6) for „rc_tar“, „eff_lam_tar“, „eff_red_tar“ or „eff_sa_tar“ delivers the target values for the controller outputs which influence torque (Figure 7).

- Priority handling
  When determining target values, a priority handling must be considered. Under normal operating conditions, torque demands must be realized by cylinder charge control to guarantee that spark advance and Lambda do not differ from the values determined in the basic calibration.

- Influence of basic calibration data:
  To compute the desired value of the cylinder charge „rc_tar“, the target value for „tqi_opt_l1“ has to be calculated first, taking into account the pilot control values of the mixture and the spark advance chosen in the basic calibration, which may differ from Lambda 1.0 and the optimal spark advance. Then the desired cylinder charge „rc_tar“ can be computed by using an inverse torque look-up table

  rc_tar = fn. (tqi_opt_l1, n_eng) (7)

- Determination of desired throttle position
  The resulting value „rc_tar“ represents the target cylinder charge which is necessary to realize the demanded torque under consideration of the boundary values (i.e. Lambda and spark advance) of the specific calibration. In the next step „rc_tar“ is converted into a target throttle opening, using state parameters from the intake manifold model described in section ‘Functional Structure’. Finally the throttle is positioned by a closed loop controller.
- Crank synchronous controller outputs:
  The set of target values for the crank synchronous controller outputs are determined in a similar manner. For example the most important target spark advance is derived from „eff_sa_tar“:

  eff_sa_tar =

  tqi_tar / (tqi_bas*eff_lam_act*eff_red_act) (8)

- Influence of actual cylinder charge:
  In equation 8 „tqi_tar“ represents the target value for the internal torque. The value „tqi_bas(rc_act, n_eng)“ is an output value of the torque look-up table with respect to engine speed and the actual value of the cylinder charge, measured for example by an air mass flow meter or computed based on a measured intake manifold pressure.
  Taking the real cylinder charge into consideration, the two torque conversion paths (cylinder charge path and crank synchronous path in Figure 7) are linked, so that no other coordination of the two paths is necessary.

2.2.1.5 Valuation of torque-based structure

The system architecture based on the internal torque provides the following advantages:

- Improved accuracy when processing system internal or external torque demands. This improvement is reached by means of a central conversion of coordinated torque demands which avoid interactions between the control variables (cylinder charge, Lambda, ignition timing and cylinder individual fuel cut-off).
- Simplified calibration:
  The characteristic lines and look-up tables of the torque control are only dependent on engine data, so there are less interactions with other control functions. As a result of modifying the engine design only the characteristic lines and look-up tables have to be actualized.
  Due to the fact that external systems have no direct impact on the throttle angle or the ignition timing, for example torque demands of a traction control or drive train control system are independent on the current status of ME7 data calibration.
  Further simplification is a result of the consistent data-base for all controller outputs. If for example the engine efficiency should be reduced at a given driver’s demand, the desired torque value to be realized via the cylinder charge path can be increased, leading automatically to a decrease of spark advance. Therefore it is not necessary to calibrate the two paths separately.
  The system can be expanded easily for future system requirements such as a drive train management for conventional gear box or a CVT [4], gasoline direct injection, variable valve timing, and so on.

2.2.2 A/F management

In addition to the torque management the ME7 system is also provided with A/F management (Figure 8).

The A/F management consists of 3 essential components:

- The basic calibration:
  Here all mixture variations caused by systematic, reproducible system tolerances are eliminated. The target of the basic calibration is a Lambda equal to 1.0 under all operating conditions.

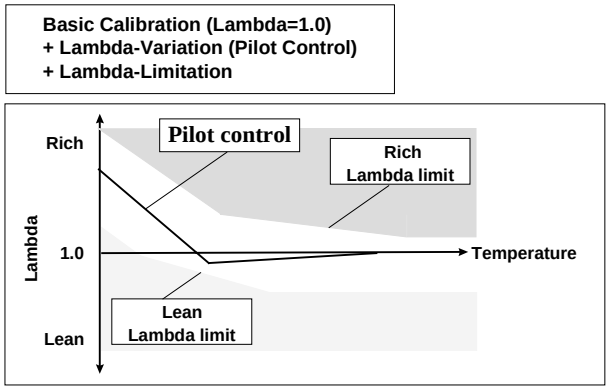


Figure 8: Characteristic Lines of Lambda Limits and Lambda Pilot Values

- Lambda pilot control:
  In addition to the basic calibration the desired Lambda value can be chosen depending on the operating condition. Therefore an enrichment during start or warm-up could be carried out for example. Furthermore an enleanment could be realized in the case of lean-burn concepts.

- Lambda limits:
  To coordinate the A/F ratio without cross-coupling of the calibration data, (i.e. for warm-up and

catalyst heating) the range of Lambda variation is limited. The limit values are defined by the mixture's flammability dependent on the engine operating point.

2.2.3 Physically based functions

Many platform functions in the ME7 are realized, based on a physical model of the controlled system. The step from heuristic functions to physically based functions requires a sufficient mathematical model of the controlled system. A major advantage of this concept is the physical interpretation of internal and interface values for improved comprehensibility and transparency.

Because of the link to physical reality it is easier to define platform functions and to reuse them in different system configurations or future EMS generations. Using a set of platform functions also allows to transfer a calibration data set from one project to another.

As an example for the physically based functions used in the ME7 the following determination and control of the cylinder charge is described on the basis of an intake manifold model.

In this model the intake manifold pressure as a central state variable is calculated to determine the cylinder charge during dynamic and steady state operation. The model considers the influences of all operating conditions which modify the manifold pressure, such as purge control, external and internal EGR and resonance flap control.

Figure 9 shows the system configuration:

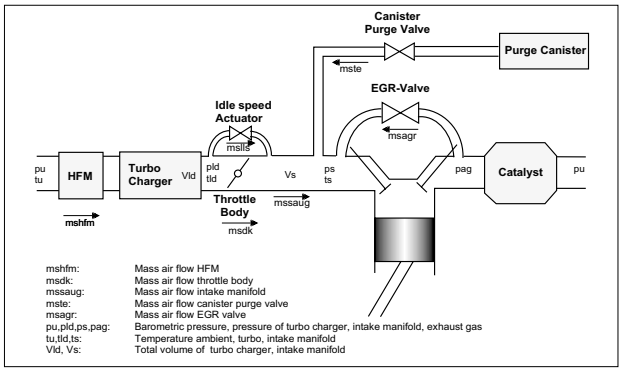


Figure 9: Intake Manifold of an SI-Engine

The model balances all mass flows into the intake manifold and all mass flows into the combustion chamber. Important input variables of the model are the air mass flow into the intake manifold metered for example via an air mass flow meter and the modeled combustion chamber temperature. As shown in Figure 10 the basic equation of the intake manifold model is generated by differentiation of the gas equation.

In the first step the intake manifold pressure is calculated based on the air mass flow measured by the air mass flow meter. With the assumption of a linear relationship to the intake manifold pressure the relative cylinder charge can be determined. [Ref.6],[Ref.7]. In a speed-density system the modeled intake manifold pressure is replaced by the measured value.

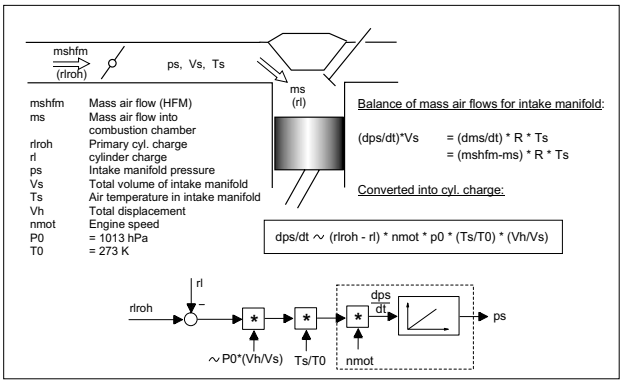


Figure 10: Basic Equation of Intake Manifold

The integration of an intake manifold model allows for more the determination of a correct cylinder charge under steady state and dynamic operating conditions. Further motivation to apply this model to the ME7 system is the necessity of a cylinder charge control. The throttle position has to be calculated according to the required engine torque (Figure 11).

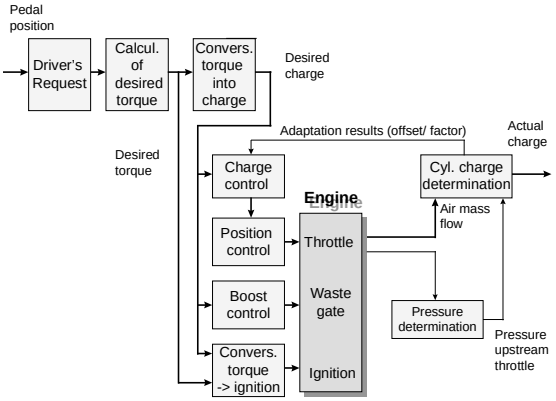


Figure 11: Cylinder Charge Control Structure in ME7

Main input values of this controller are the modeled intake manifold pressure and two adaptive values which eliminate the influences of air leakage and tolerances between the cylinder charge signal and the expected throttle position at a given air mass flow and engine speed .

2.3 SOFTWARE STRUCTURE

2.3.1 Changes Made

Requirement analysis or constructing a requirement model is not a simple sequential step, which is finished before the architecture, design, implementation and test development activities have started. There is extensive overlap between the phases. The process is

iterative, both within each phase and over multiple phases. This is especially true for a platform development, supporting a complete product family with a broad variance in system and customer specific configurations.

Systems evolve in time. Building a new system from scratch is not the typical approach, due to restrictions in development time and effort. The ME7 was built from an existing system in an incremental and evolutionary manner, thus being able to deliver a running system for test and calibration purposes right from the beginning.

The maintenance or reengineering phase therefore does not begin with the delivery and it does not end with delivery and mass production. The maintenance phase is in fact the most important phase to prove expansion capability, reuse and variants engineering.

Hard facts are the ever increasing complexity, changing functional requirements and continuously tightened nonfunctional requirements related to quality, cost and time to market.

Development systems have to handle these real world aspects to immediately improve the situation. In the ME7 development therefore a selected set of changes was planned and introduced step by step. This process is still ongoing.

There are bottom up forced changes. The use of high level programming languages, ANSI C instead of Assembler in the ME7, is one example. There is no sensible way to handle Assembler programs of such complexity. The systematic introduction of specification simulation and code inspection methods are further examples for this class of changes, which were absolutely necessary.

Changes with "top down character" in the project were the development of a new functional structure and of a software architecture to support their implementation in a product line.

2.3.2 Context

What is the system we are talking about? Traditionally we find a dedicated electronic control unit (ECU) for engine management, connected to a lot of sensors, actuators and other control units or monitoring devices in the car. The system consists of the ECU itself and all the devices it is connected to. Proceeding in an incremental and evolutionary way the context diagram is drawn by replacing the ECU with the single transformation "control & monitor engine" and the devices connected to it are the terminators. In the ME7 project this partitioning was a given fact. (The context diagram partitions the information and processes that are within the scope of the system and those that are outside the scope)

The ECU has one or two standard microcontrollers and peripheral electronics functionally comparable to former systems. The peripheral electronics are more highly integrated. The mechanical design of the ECU is also new, supporting surface mounting of all electronic devices. Furthermore variants in microhybrid technology are available, which offer new options in mechanical system integration. Yet the partitioning between hardware and software was not under consideration in the project. Being aware of the many changes in the development process, functional structure and software was a conscious decision for risk management.

There are various possible system configurations with more than a hundred input and output flows, however all have in principle the same topology and a lot of commonalties. Measuring engine speed and load are examples for commonalties on an abstract level between all system variants.

Most important for a given system variant a static environment can be assumed, therefore a deterministic system approach was chosen.

2.3.3 Architecture

In the architectural development the question how to design has to be answered. The functional requirements at one level must be rigorously allocated to a physical structure with nonfunctional requirements added.

2.3.4 Software architecture

Following strategies for real time system specification as described, for example, by Hatley and Pirbhai [Ref. 11] or Goldsmith [Ref. 12] and using the principles of abstraction and decomposition, the technology independent requirements model has to be transformed in a technology-nonspecific physical model. Therefore buffers are inserted between the essential requirements model core and the environment. The resulting architecture template embeds the requirements model core into input processing and output processing architecture blocks. Also blocks for interface-processing and self-test are added as shown in Figure 12.

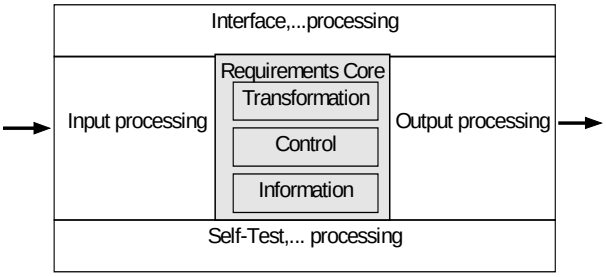


Figure 12: Architecture Template

The essential requirements model describes what the logical outputs should be, depending on the logical inputs. It assumes ideal technology, though all the input flows are in parallel and instantly transformed into the corresponding output flows. The input and output processing blocks have to convert the physical flows

into the logical flows described by the requirements core model. This model has to be enhanced by the timing requirements. For all the logical flows response times or actualization rates, the resolution, accuracy and value ranges must be specified.

The further decomposition into architecture modules, architecture flows and so on, ending in code organization charts is a sophisticated process. Important guidelines for partitioning are the general principles of cohesion and coupling, as known from structural analysis and design methods. This is also valid for object based or other methods and languages for real time system description and construction. Independent from the methods used, this is an example of the already mentioned inherent iterative and recursive character of complex systems development.

The intention here was to minimize the influence of the µC and the various configurations, using different sensors and actuators on the design and implementation of the requirements core block. The requirements model represents the knowledge about the application work and should be reused even on the implementation level for the whole product family and also for further generations of engine management systems. A set of rules for architecture, design and implementation is necessary to encapsulate the µC and system configuration dependencies.

2.3.4.1 Layer structure model

The top level of the set of rules used in this project is shown graphical in Figure 13.

One idea of a layer based architecture is, that lower layers offer all the services needed by higher layers. The supporting character of offering services is visualized here by the triangles. It is important to recognize that their semantic is not equal to flow. The user program gets it's inputs by using services and it provides the outputs for example by using services of the hardware encapsulation layer.

The run-time part of ERCOS (Embedded Real-time Control Operating System), [Ref. 13] follows the concepts of virtual machines, or simply it offers services for scheduling, inter-process communication and so on, to construct and support applications with hard real-time requirements.

The user library consists of arithmetic, filtering, integration and interpolation routines. They are application dependent and most of them are implemented in Assembler, using all the features of the µC for best efficiency.

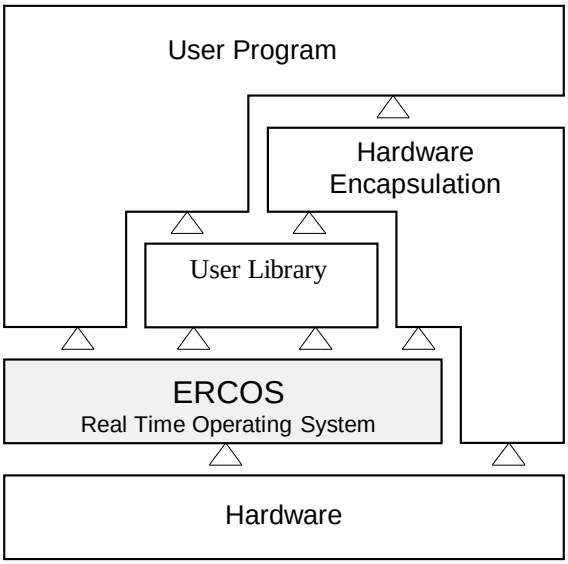


Figure 13: Layer Structure Model

The hardware encapsulation contents serves to read ADC-Channels, reading digital inputs, setting digital outputs, CAN-Handling, or generating PWM-Signals. Also most parts of the input and output processing are contentented in this layer. This is also true for the interface processing and self-test architecture blocks. The hardware encapsulation therefore has an inner structure, which has to be very carefully designed and shown as a mission critical point for success.

Together the operating system, the user library and the hardware encapsulation can be thought of as a hardware abstraction layer.

Looking at Figure 12 and remembering the concept to make the design and implementation for the requirements core model as independent from the hardware as possible, it should be reside in the user program layer.

Maybe this sounds theoretical, but in the project we experienced positive examples to show that it works. Different electronic throttle bodies are supported with no influence on the software in the user program layer and different load measurement sensors can be used by simply exchanging a few parts in the software.

2.3.4.2 Object Model

ERCOS offers a complete framework and is a object based language for real time system construction.

The basic object classes that are supported are processes, functions, messages and resources (Figure 15). Out of the basic classes only processes are active. A function is a passive object which can be called. Messages are basic objects for communication between processes and provide the methods send and receive. To model resources which can only be accessed exclusively, resource objects are provided with the methods get and release.

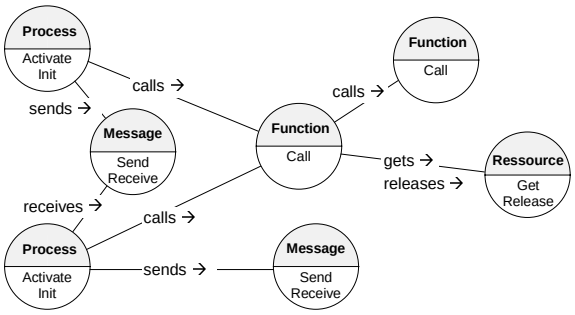


Figure 14: Basic Objects and their Relations

Complex objects, the previously mentioned „subsystems“ with well defined interfaces can be constructed out of basic objects. Each subsystem defines which objects constitute the interface and which are hidden inside. Figure 15 shows an example of a subsystem which provides a function and a message as an interface. Additionally, to implement the required functionality there are two processes, two functions and one resource which are hidden inside the subsystem.

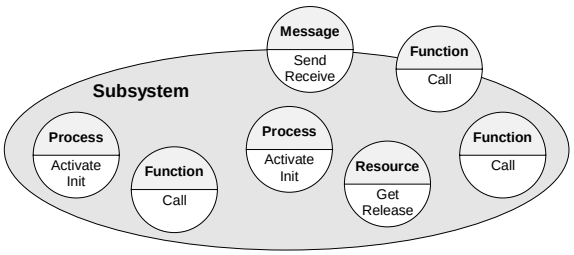


Figure 15: A Complex Object (Subsystem)

The object model is supported by a set of tools (ERCOS-Off Line Tools) which have been developed to allow easy construction of object-based software while providing interface checks for consistency and optimization techniques for the runtime management of objects. The subsystem concept of ERCOS is one major step towards software sharing and open systems.

2.3.4.3 Scheduling

A combination of static and dynamic scheduling together with a mixed preemptive/cooperative scheduling strategy is supported. Due to hard efficiency constraints in the project as a result of hardware cost, mostly static scheduling is used. Tasks are implemented as schedule-sequences that contain a sequence of processes to be executed in the specified order and at a given priority level upon occurrence of a certain activation event (Figure 16).

Only a small amount of the application has short latency requirements which requires preemptive scheduling while the large remainder can be scheduled cooperatively. This is realized by a hierarchical scheduler concept where the cooperative scheduler is subordinated under the preemptive scheduler. The cooperative scheduler is treated as a single task at the lowest priority level of the preemptive scheduler. Both scheduling strategies use fixed priority assignments.

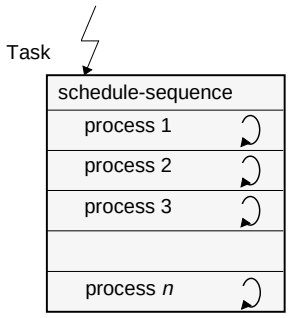


Figure 16: Tasks

It is possible to assign an interrupt source to a preemptive priority level. There is no distinction necessary between tasks that are activated by hardware (interrupts) or by software. Thus a unified concept for hardware and software activated tasks is provided.

Additionally multiple operating modes are supported, to allow completely different configurations (i.e. factory testing, flash programming and driving mode).

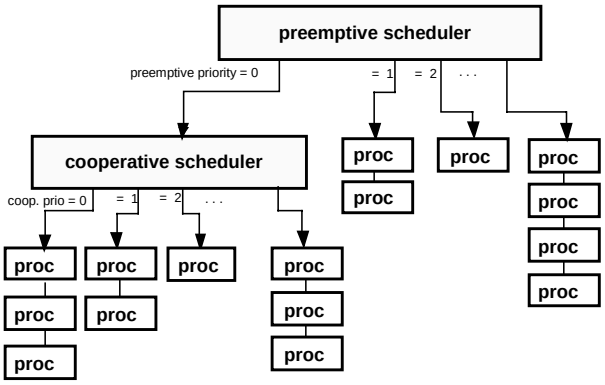


Figure 17: Scheduler

2.4 SYSTEM FAMILY

The system architecture characterized by a torque-based system structure, an A/F management and the use of physical based functions allows the generation of a system family containing

- the EMS with integrated throttle control (ME7)
- the conventional (M7) system with a bypass idle speed actuator
- the system for gasoline direct injection (MED7)
- and the system for a complete drive train management (MEG7).

The design of a consecutive system generation, based on the same system architecture but using a different CPU architecture is also possible (Figure 18).

- M7: System without ETC
  To create this conventional or basic system without

ETC the cylinder charge control has to be limited to the operating range of the idle speed actuator. Due to the fact, that there is a fixed mechanical link between the pedal and the throttle position, the throttle position represents the driver's request. With the help of these major supplements the M7 system was derived very easily and in a very short development time.

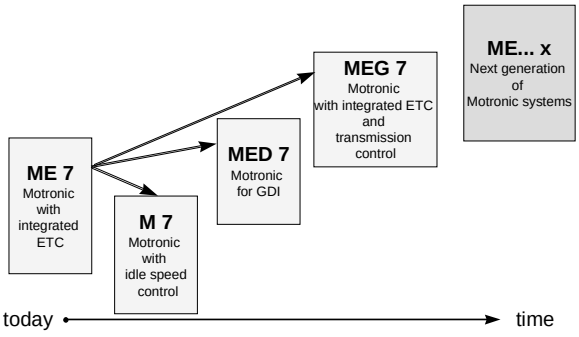


Figure 18: Roadmap of System Family

- MED7: System for gasoline direct injection
  The main functions provided in addition to the ME7 system are: [Ref.14],[Ref.15]
  - ♦ Control of torque and emissions under stratified and homogeneous operation and the transition between these two states.
  - ♦ Control of an EGR system with high flow rates.
  - ♦ Control of the entire fuel system including the high pressure control and the operation of high pressure injectors.
  - ♦ Canister purge control for homogeneous and stratified operating conditions.
  - ♦ Control of a NOx storage catalyst
  - ♦ Suitable monitoring concept
- MEG7: Drive train management
  The step from the management of engine related torques to a drive train management includes not only the EMS but also the control of the transmission and the torque converter. Therefore an integrated drive train control means coordinating all torque demands on the level of drive train output torque. An optimized control strategy of all drive train components allows a further reduction of emissions/ fuel consumption [Ref. 15], [Ref. 3].
- CARTRONIC®
  A further step is the management of all torque-related demands on the basis of wheel torque, which allows the integration of all vehicle related torque demands including not only current vehicle dynamic control systems but also an active braking system for example. [Ref. 4].

3. CONCLUSION

- There is a broad range of system configurations supported by the EMS product family based on the ME7. Standard MOTRONIC M7, MOTRONIC with electronic throttle control ME7, MOTRONIC with electronic throttle control and integrated transmission control MEG7, and MED7 for gasoline direct injection engines are all members of this family.
- ERCOS is now the standard operating system for BOSCH automotive products. As an open market product it is available from the company ETAS for different microcontroller platforms . In the EMS product line the run time part is reused bit for bit in all products.
- The encapsulation of sensors/actuators for different system configurations and the µC in a hardware abstraction layer are the foundation for reuse and flexibility. Benefits of the new functional structure are the improved physical transparency of the system, and the separation of engine and function related calibration data. The introduction of the new functional and software structure as a core of the product family enables future challenges to be met, such as an extended system, for example CARTRONIC®.

CONTACT

Jürgen Gerhardt
Robert Bosch GmbH - K3/ESY
P.O. Box 30 02 40
D 70 442 Stuttgart/ Germany
E-mail: Juergen.Gerhardt@pcm.bosch.de

Harald Hönninger
Robert Bosch GmbH - K3/EAP
P.O. Box 30 02 40
D 70 442 Stuttgart/ Germany
E-mail: Harald.Hoenninger@pcm.bosch.de

Dr. Hubert Bischof
Robert Bosch GmbH - K3/ESY
P.O. Box 30 02 40
D 70 442 Stuttgart/ Germany
E-mail: Hubert.Bischof@pcm.bosch.de

REFERENCES

1. Glöckler,O.; Benninger,N.F.:

„Beitrag der Motorsteuerung für Ottomotoren zur Senkung des Kraftstoffverbrauchs“
Stuttgarter Symposium 1995

2. Glöckler, O.; Bischof, H.:
„Motorsteuerungssysteme der Zukunft“
16. Internationales Wiener Motorensymposium, Wien 1995
3. Streib,H.M.; Bischof,H.
„ETC: A cost Effective System for Improved Emissions, Fuel Economy and Driveability“
SAE Technical Paper Series 960338
4. Gerhardt,J., Benninger,N., Heß,W.
„Drehmomentorientierte Funktionsstruktur der EMS als neue Basis für Triebstrangsysteme“
6. Aachener Kolloquium; Fahrzeug und Motorentechnik 1997
5. Gill, P.E.; Murray, W.:
„**Quasi-Newton Methods for Unconstrained Optimization.“**
Journal of the Institute of Mathematics and it’s Applications
6. Hendricks, E. , Chevalier, A. ; Jensen, M.; Sorensen, S.C.:
„Modelling of the Intake Manifold Filling Dynamics“
SAE Technical Paper Series 96 0037, 1996
7. Streib, H.M.; Wild, E.:
„Modellbasierte Füllungserfassung und -steuerung zur Verbindung von hoher dynamischer Aktualität“
Symposium Steuerungssysteme f.d. Antriebsstrang von Kfz., Berlin 1996
8. A.S. Tanenbaum.:
„Modern Operating Systems“
Prentice-Hall. 1992.
9. B. Meyer.:
„Object-Oriented Software Construction“.
Prentice Hall Book Co., Inc. 1988.
10. NN:
„OSEK (Open Systems and the Corresponding Interfaces for Automotive Electronics)“
Operating System. 1995.
11. D. J. Hatley and I. A. Pirbhai,:
„Strategies for Real-Time System Specification“
Dorset House Publishing, 1987
12. Sylvia Goldsmith:
„A Practical Guide to Real-Time Systems“
Development, Prentice Hall 93
13. Poledna, Mocken, Schiemann, Beck:
„ERCOS: An Operating System for Automotive Applications“.
SAE International Congress, Michigan, 1996
14. Stutzenberger, H; Preussner, C.; Gerhardt, J.:
„Benzin-Direkteinspritzung für Ottomotoren - Entwicklungsstand und Ausblick“
18. Internationales Wiener Motorensymposium, 1997
15. Moser,W.; Mentgen, D.; Rembold, H.:
„Benzin-Direkteinspritzung - eine neue Herausforderung für zukünftige Motorsteuerungssysteme“
Motortechnische Zeitschrift 58 (1997)/Nr.9 & Nr.10
16. Streib, H.M.; Blank, H.; Wimmer, W.:
„**Eine optimierte Antriebstrangsteuerung mit der Funktion Mastershift“**
BDI-Berichte Nr. 1099, S111-124, Düsseldorf 1993

ABBREVIATIONS

A/C	AirConditioner
ADC	Analog Digital Converter
ANSI C	American National Standard Institution - C-Programming Language
CAN	Controller Area Network
CARTRONIC®	Structure and order of all functions, controlled by elecTRONIC control modules in a CAR.
CPU	Central Processing Unit
CVT	Continuously Variable Transmission
ECU	Electronic Control Unit
EGAS	Electronic GASpedal
EGR	Exhaust Gas Recirculation
EMS	Engine Management System
EOBD	European On-Board Diagnosis
ERCOS	Embedded Real Time Operating System
ETC	Electronic Throttle Control
M7	MOTRONIC system without ETC
ME7	MOTRONIC system with integrated ETC
MED7	MOTRONIC system with ETC for gasoline Direct injection
MEG7	Powertrain management system
MOTRONIC®	BOSCH engine management system
MY	Model Year
µC	Micro Controller
OBD II	On-Board Diagnosis - stage II
OSEK	Open Systems and the Corresponding interfaces for automotive Electronics
PWM	Pulse Width Modulation
SI-engine	Spark Ignition Engine

d_sa	= sa_opt - sa
eff_lam	Lambda efficiency
eff_lam_act	Actual Lambda efficiency
eff_red	Reduction factor
eff_red_act	Actual reduction factor
eff_sa	Spark advance efficiency
eff_..._tar	Target efficiency values
fn.	Function of, depending on
lam	Lambda
lam_bas	Lambda of basic calibration
n_eng	Engine speed
rc	Relative cylinder charge
rc_act	Actual relative cylinder charge
rc_tar	Target value rc
sa	Ignition angle reffering to TDC
sa_bas	Spark advance of basic calibration
sa_opt	Optimal spark advance
sa_opt_l1	Optimal sa at lamda 1.0
tq_i	Internal torque, generated by combustion
tqi_bas	tqi at sa_bas and lam_bas
tqi_opt,	Optimal internal torque
tqi_opt_l1	Optimal tq_i at Lambda 1.0
tqi_tar	Target value t_qi
tqi_opt_l1	Optimal tq_i at Lambda 1.0