Deterministic and High-Performance Communication System for the Distributed Control of Mechatronic Systems Using the IEEE1394a

Mauro Zanella, Thomas Lehmann, Thorsten Hestermeyer, Andreas Pottharst
University of Paderborn, Germany
zane@mlap.de, torkin@upb.de, herster@mlap.de, pottharst@lea.upb.de

Abstract: The complexity of mechatronic elements has grown along with the customer requirements in automotive and public transportation. The mechatronic systems can not be designed as a stand-alone unit, but as a part of an interacting network. To attend safety and synchronisation between processing elements we present in this paper the use of IEEE1394a for a deterministic communication system. As a case study we use the Novel Railway System Paderborn, which will be implemented using the hardware platform RABBIT.

Key words: time-triggered communication, mechatronics, distributed real-time control, IEEE 1394a

1.   INTRODUCTION

As almost no other technology, mechatronics is an essential element of many innovations in automotive engineering. New functions and improvements of already existing functions, as well as the compliance with traffic regulations and customer requirements, have only become possible by the increasing use of mechatronic systems, especially in the fields of driving, safety, reliability, and functionality. The car manufacturer has to meet demands regarding fuel consumption, emission, ride comfort, power, drive dynamics, and ergonomics. Along with the functionalities that increase in number and have to cooperate, the complexity of the entire system will increase.

Synergy effects resulting from distributed application functionalities via several micro-controllers and the use of sensor information through the

network bring about more complex system architectures with many different sub-networks operating with different velocities and different protocol implementations. Networking may even reach a state of such complexity that a careful systematic design becomes necessary.

To manage the increasing complexity of these systems a deterministic behaviour of the communication network must be provided for, in particular when dealing with a distributed functionality or redundant realization of certain nodes. This can be achieved by applying the design philosophy of time-triggered operation [2] to the communication network (time-triggered protocol activities [3]) and on the application level (time-triggered task activation [4]). The optimum is reached when a globally synchronized time base (global time) is available to all nodes of the network at a measure of precision fulfilling the real-time requirements of the application.

Actually the TTTech company, Bosch and the joint venture of BMW and DaimlerChrysler offer solutions for networks regarding time-triggered operation, namely TTP/A [3], TTCan [1], and FlexRay [5].

Our contribution to this enterprise is the application of IEEE1394a for a deterministic, high- performance and fault-tolerant communication system in the field of distributed control systems. The IEEE1394a bus (also known as FireWire or iLink) is a high-speed serial bus for communication of up to 400 Mbit/s.

RABBIT, a modular rapid-prototyping platform for distributed mechatronic systems [6], will be used for implementation and evaluation purposes. The main features of RABBIT are its flexibility and extensibility, brought about by an open system interface and high modularity. An overview of the main features of the IEEE1394a bus, as well as the RABBIT system, will be given in Chapter 2.

This paper will also present the structuring and the development of the drive module of the Neue Bahntechnik Paderborn (Novel Railway System Paderborn) [7] as a case study. More details of the NBP and its mechatronic function-oriented structure are to be found in Chapter 3.

2. DETERMINISTIC COMMUNICATION SYSTEM USING RABBIT AND IEEE1394A

2.1 RABBIT – a Modular Rapid-Prototyping Platform for Distributed Mechatronic

We employ the RABBIT platform [6] to demonstrate the possibility of its being implemented in the case-study due to its modularity and flexibility.

RABBIT offers the chance of checking up on novel approaches to the research on rapid-prototyping systems.

The hardware modules comprise three main components: microcontroller, FPGA (Field Programming Gate Array), IEEE 1394a (FireWire). Each module is a board of its own and can operate in stand-alone or combined mode. At the moment, two new modules have been developed on the basis of these main components, the first one for the control of stepper motors, and the second one for the communication between RABBIT (System bus) and dSPACE (PHS-bus).

One node of the RABBIT system consists of a rack which can contain the different modules, as shown in Figure 1, connected via a local system bus.

The CPU module of the node is provided by a PowerPC module. It consists of an MPC555 [8] with its on-chip peripheral devices and an extra bus interface to transmit the memory bus signals to the local system bus:
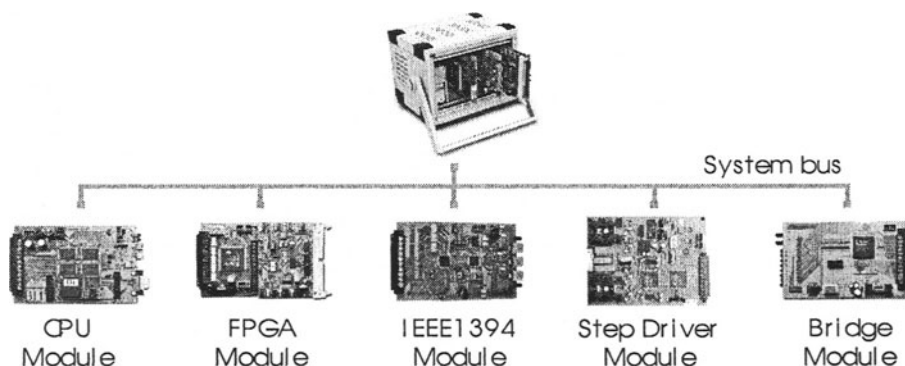

Figure 1: Implemented RABBIT modules

The main component of the DSP module is a Virtex-E FPGA [9]. In addition to the system-bus interface the Vertex-E also has an internal bus interface. Via this bus it is possible to connect I/O devices, e.g., ADCs, DACs, and encoders. These components are mounted on a piggyback board.

The IEEE1394 module provides three ports for communication. It has an MPC555 bus interface to communicate the link-layer chip to the local bus (asynchronous), and an exclusive bus interface to the DSP module for isochronous communications.

2.2   IEEE1394a: a High-Speed Serial Bus

The 1394 high-speed serial bus was conceived in 1986 by Apple Computers, who chose the trademark 'FireWire', and adopted as IEEE 1394 standard in 1995. At about the same time Sony introduced its trademarked

iLink as a significant variation of the standard. A large number of IEEE 1394 products are now available including digital camcorders, integrated and stand-alone digital video editing systems, digital VCRs, digital still and video cameras, digital audio players, harddisks, and plenty of other infrastructure products, such as connectors, cables, test equipment, software toolkits, and emulation models.

Some features of the IEEE1394a are the following [10]:

- "Hot-pluggable": devices can be added and removed while the bus is active
- Scalability: the standard defines 100, 200, and 400 Mbps devices; the connection can thus support the different speeds on a single bus
- Flexibility: the standard supports free-form daisy chaining and branching for peer-to-peer implementations
- Support of two types of data transfer: asynchronous and isochronous
- Global timebase (125 us): for synchronizing events and data
- Multi-master capability

For conventional memory-mapped data access, asynchronous transfer is appropriate and adequate. The isochronous data transfer provides guaranteed data transport at a pre-determined rate, a feature that is especially important in control applications with time-critical data transfer.

The high data rates of FireWire, the ability to mix isochronous and asynchronous data on a single line, and the global timebase encouraged us to employ this bus for real-time communication in distributed mechatronic applications.

2.3   Time-Triggered Communication Using the IEEE1394a

Purely time-triggered operation of the communication system means that any activity is determined by the progression of a (globally synchronized) time. Sending, receiving or other activities depend on a predefined time schedule and the current state of the clock [1].

In the case of the IEEE1394a, a global clock rate of 8 kHz is implicitly included in the system, i.e., communication on the bus is basically made up of cycles (time-slot) of 125 us each. Figure 2 displays two communication cycles on the bus. After start of the cycle, including the timestamp, the time slot is divided into two parts: the first one is for isochrounous (channels) and the second for asynchronous communication (packet) [10].

In the case of distributed control applications, the cyclic and deterministic data transfer will be performed in the shape of isochrounous messages, and with administrative data (e.g., tracing, parameter changes) the asynchronous section will be used:
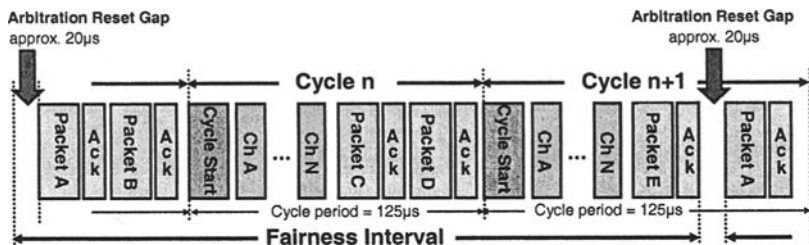
Figure 2: A series of transactions

A static scheduler with regard to the entire system has to be defined in order to determine which process and which processor will send and receive data in a pre-defined time slot, as defined in the time-triggered operation. Due the complexity of the entire system, it is nearly impossible to develop the control algorithms and to distribute them to several Electronic Control Units (ECUs) taking into account, for instance, system stability, partitioning, code generation, and deadlocks.
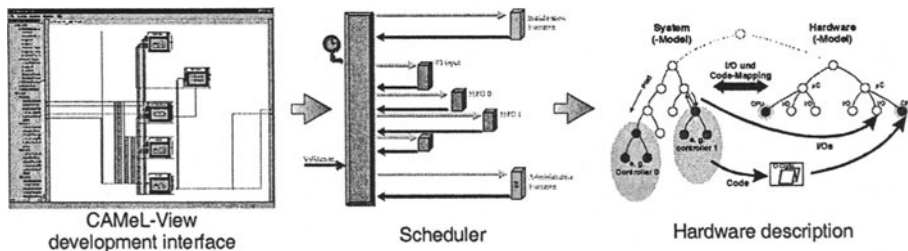

Figure 3: Code generation and mapping in CAMeL-View

Systematic development methodology, as presented in [11], is applied to the scope from modelling to analysis to synthesis and up to C-code generation for the real-time applications. High-level software tools, such as CAMeL-View[12] and Alaska [13], help the engineer during these stages.

In our case we use the software tool CAMeL-View which allows a modular and hierarchical model description of the system. The entire control algorithm is designed in dependence of the physical structure of the system, as described in Chapter 3 (CMS, AMS, MFM). From this structured model the C-code is generated according to the interdependence of variables and control strategies [14]; it runs on the ECUs.

The design of a static scheduler for one processor which takes into account the hard real-time (control algorithms and communications) and soft real-time (administrative tasks) conditions is described in [15].
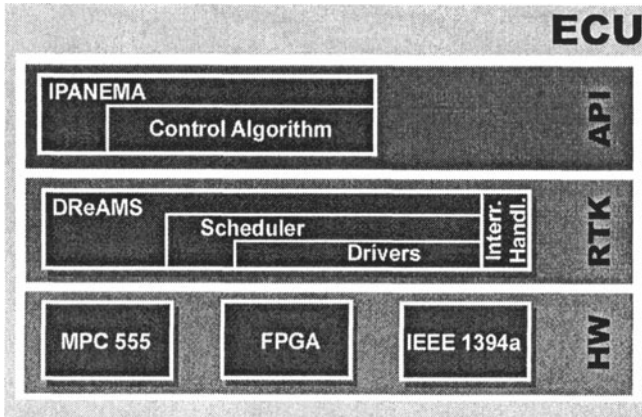

Figure 4: Software implementation of an ECU (RABBIT)

The next step is to map the abstract inputs and outputs of the model to the actual I/Os of the ECUs (microcontrollers). A further hardware description is comprised in the CAMeL-View modelling tool in parallel to the system model. In this description the inputs/outputs of the system model are linked to the respective hardware I/Os via a mapping description [16]. Communication and administration between processes will be implemented by means of the software platform IPANEMA (Integration Platform for Networked Mechatronic Applications) [17]. The objects defined in IPANEMA offer services on the application level. The re-configurable real-time operating system DReAMS (Distributed Extensible Application Management System) [18] can be used for the lower-level services. Complex micro-controllers, in our case the Motorola PowerPC MPC555, provide the engineer with many peripheral devices as a System-on-a-Chip. To configure the interrupt system of such complex microcontrollers a configuration software is in the works. It will include a synthesis of a interrupt request dispatcher [19].

3. CASE-STUDY: *NEUE BAHNTECHNIK PADERBORN* (NOVEL RAILWAY SYSTEM PADERBORN)

Two major drawbacks of conventional railway systems as compared to cars are the rigid timetables, which the customer has to submit to, and the necessity to change trains in order to reach one's destination. With its new logistical concept, the Neue Bahntechnik Paderborn (NBP - Novel Railway System Paderborn) presents a system that solves these problems. It features fully automated transport by small shuttles which make up convoys on their way if the occasion arises. These shuttles combine the common wheel/rail technology with the linear motors of the maglev trains. In order to prove the functionality of the shuttles and the basics of the logistical concept, a test track will be built up in Paderborn at a scale of 1:2.5.

Due to the complexity of the system the shuttle prototypes are structured in a modular/ hierarchical way. According to mechatronic design principles each physical aggregate disposes of its own information processing, thus forming a Mechatronic Function Module (MFM) [20] with sensors, actuators and information processing supporting the mechanical part of the module. The three major modules of an NBP shuttle are the guidance module, the linear drive module and the suspension/tilt/steering module. All these modules, linked by an additional mechanical structure, make up the Autonomous Mechatronic System (AMS), a shuttle which has its own sensors and information processing but modifies the system behaviour only by means of the MFMs. The information processing of the AMS makes up the highest hierarchical level in the control structure of the mechanically linked system. In order to control shuttles in a convoy an even higher control level is needed. These shuttles, cross-linked only by this higher-level control, form a Cross-linked Mechatronic System (CMS). A CMS can rely on its own sensors and information processing but cannot be allocated to a mechanical structure.

As regards the linear motors for railway vehicles, they fall into either one of two categories: long-stator or short-stator motors. A short-stator linear motor is designed either as an asynchronous motor with an electrified primary on the vehicle and a reaction plate between the rails (e.g., Linear-Metro Tokio, Kobe). When the primary is mounted between the rails with the secondary on board of the shuttle, we obtain a long-stator motor. In most cases the secondary is electrified. In synchronous operation DC is applied to the secondary. Thus, the vehicle moves at the speed of the migrating wave (e.g., the Transrapid). Supplying AC to the secondary leads to asynchronous operation (NBP). Hence relative motion between vehicles becomes possible

on a primary that is not split into sections. Additionally it is possible to transfer energy from the track to the shuttle in order to spare catenary wires.

In practice the primary is divided into sections so that the primary need not be current-carrying along the whole of the track. In order to ensure a smooth and accurate motion of a shuttle it is extremely important that the migrating wave be continued in phase and at the correct amplitude and frequency from one primary section to the next (see Figure 5). Therefore data connections and information processing must be provided to control primary and secondary currents:
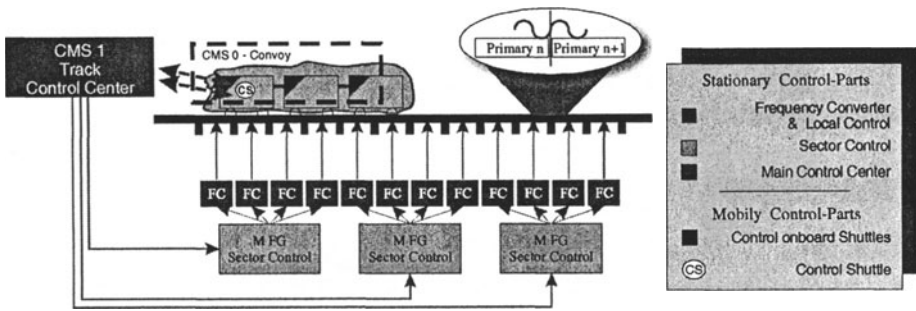

Figure5: Information structure of the NBP system with the focus on the linear motor

The information processing of the CMS Convoy is located on one "Control Shuttle" which is determined dynamically. This control shuttle communicates with the stationary track control center in two ways: for real-time data there is a serial radio transmission. As far as the stator control goes, the track-control center transfers the rotor information and information on the required wave phase and speed to the appropriate sector controls that control four frequency converters each. The frequency converters again have local control units of their own to compute the necessary currents. The track-control center and the sector controls will be implemented using the RABBIT platform. The basis of the frequency converters (also a micro-controlled unit) will be a commercial product with a peer-to-peer synchronized serial bus communication to the sector-control unit as an add-on. This means that 126 ECUs (26 RABBITs and 100 frequency converters) will be interlinked along the test track. Our focus in this work lies on the communication between the track-control center and the sector controls (26 ECUs).

4.   STATE OF AFFAIRS

The aim is a consistent design of all elements, from the system model to the automatically generated source code for all subsystems. With RABBIT it is possible to preserve the hierarchical structure of the control model and its subsystems. The aim is to show that our concept can be successfully applied to the driving module of the NBP.

Presently we are testing the IEEE1394a in the lab on other applications. A compact RABBIT hardware is also in the design stage and going to be implemented in the NBP application. The reason for developing it is the high-frequency electromagnetic noise generated by the servos and the small space available for an installation on the test track. Moreover, the missing code-generation modules for the CAMeL-View tool are being developed.

5.   REFERENCES

[1]   Müller, B.: Zeitgesteuerte Kommunikation mit CAN. Auto&Elektronik, Germany, February 2001.

[2]   Kopetz, H.: The Time-Triggered Approach to Real-Time System Design. Technical University of Vienna, 1997.

[3]   Kopetz, H.; Grünsteidl, G.: TTP – A Protocol for Fault-Tolerant Real-Time Systems. IEEE Computer, January 1994.

[4]   Krüger, A.; Domaratsky, Y.; Holzmann, B.; Schedl, A.; Ebner, Ch.; Belschner, R.; Hedenetz, B.; Fuchs, E.; Zahir, A.; Boutin, S.; Dilger, E.; Führer, T.; Nossal, R.; Pfaffeneder, B.; Poledna, S.; Glück, M.: OSEKtime: A Dependable Real-Time Fault-Tolerant Operating System and Communication Layer as an Enabling Technology for By-Wire Applications. SAE 2000 Congress, Detroit, MI, March 2000.

[5]   FlexRay für verteilte Anwendungen im Fahrzeug. Elektronik Automotive, Germany, May 2001.

[6]   Zanella, M.; Robrecht, M.; Lehmann, T.; Gielow, R.; de Freitas Francisco, A.; Horst, A.: RABBIT: A Modular Rapid Prototyping Platform for Distributed Mechatronic System. SBCCI 2001 - XIV Symposium on Integrated Circuits and Systems Design; Brasília, Brazil, 2001.

[7]   Lückel, J.; Grotstollen, H.; Jäker, K.; Henke, M.; Liu, X.:. Mechatronic Design of a Modular Railway Carriage. IEEE/ASME International Conference on Advanced Intelligence Mechatronics (AIM '99), Atlanta, GA, 1999.

[8]   Motorola, Inc.: MPC555 Evaluation Board Quick Reference, 1999.

[9]   Xilinx – The Programmable Logic. Data Book 2000. San José, CA, 2000.

[10]   IEEE Computer Society: IEEE Standard for a High Performance Serial Bus, New York, NY, March 2000.

[11]   Lückel, J.; Koch, T.; Schmitz, J.: Mechatronik als integrative Basis für innovative Produkte. Mechatronik - Mechanisch/Elektrische Antriebstechnik, VDI-Verlag, Düsseldorf, 2000.

[12]   Hahn, M.: OMD - Ein Objektmodell für den Mechatronikentwurf. VDI-Fortschrittberichte, Reihe 20, Nr. 299, VDI-Verlag, Düsseldorf, 1999.

[13]   Maißer, P.; Jungnickel, U.: Ljapunov-stable Position Control of Constrained Multibody Systems. Proc. 3rd International Heinz Nixdorf Symposium: Mechatronics and Advanced Motion Control, Paderborn, Germany, 1999.

[14]   Homburg, C.: Mechatronic Processing Objects. ASIM '98, Zurich, Switzerland, 1998.

[15]   Zanella, M.; Stolpe, R.: Distributed HIL Simulation of Mechatronic Systems Applied to an Agricultural Machine. DIPES '98, Eringerfeld, Germany, 1998.

[16]   Zanella, M.; Koch, T.; Meier-Noe, U.; Scharfeld, F.; Warkentin, A.: Structuring and Distribution of Controller Software in Dependence of the System Structure. CBA2000, Florianópolis, Brazil, 2000.

[17]   Honekamp, U.: IPANEMA - Verteilte Echtzeit-Informationsverarbeitung in mechatronischen Systemen, VDI-Fortschrittberichte, Reihe 20, Nr. 267, VDI-Verlag, Düsseldorf, Germany, 1998.

[18]   Ditze, C.: Towards Operating System Synthesis. Diss., Universität Paderborn; HNI 2000 (HNI-Verlagsschriftenreihe, Bd. 76), Paderborn, Germany, 2000.

[19]   Lehmann, T.; Zanella, M.: Modelling and software synthesis of interrupt systems. GI/ITG/GMM Workshop: Methoden und Beschreibungssprachen zur Modellierung und Verifikation von Schaltungen und Systemen,Tübingen, Germany, 2002.

[20]   Honekamp, U.; Stolpe, R.; Naumann, R.; Lückel, J.: Structuring Approach for Complex Mechatronic Systems. 30th International Symposium on Automotive Technology & Automation – Mechatronics/Automative Electronics, Florence, Italy, 1997.