ELSEVIER

Sensors and Actuators A 68 (1998) 460–465

SENSORS AND ACTUATORS A PHYSICAL

A local bus for multi-chip-module-based microinstrumentation systems

J.H. Correia, E. Cretu *, M. Bartek, R.F. Wolffenbuttel

Delft University of Technology, Department of Electrical Engineering, Laboratory for Electronic Instrumentation/DIMES, Mekelweg 4, NL-2628 CD, Delft, Netherlands

Abstract

A local smart sensor bus is described which is designed for use in a multi-chip-composed microinstrumentation system. The bus is capable of handling information in digital code, bitstream, analog voltage, and frequency or duty cycle modulation and also provides calibration facilities, service-request and interrupt-request modes. The resulting sensor bus interface has been implemented in a 1.6 μm CMOS process and successfully tested in a local sensor network.

Keywords: Instrumentation microsystems; Internal bus interfaces; Integrated smart sensor bus protocol

1. Introduction

A microinstrumentation system, as shown schematically in Fig. 1, can be regarded as a merger between microelectronics and micromechanics. Optimizing the balance between high performance and a reduced time to market favours the design of a flexible architecture, reusable for different applications. The concept of an ASIC (application-specific integrated circuit) based on standard blocks has provided an attractive solution in microelectronics. A similar approach was undertaken for designing a general MCM (multi-chip module) architecture, based on an internal bus protocol adjusted for the specific requirements of a measurement system. The resulted MCM device should be considered as a genuine (micro)instrument, which supplies high-level data over an external instrumentation bus. Moreover, the microsystem behaves as an autonomous unit, which supports internally all vital functions, such as power management, self-calibration and in-system data transfer. Drivers are designed for a generic sensor module; the application will decide the type of sensors to be used. In such a way, the driver modules become reusable components, independent of the type of measured quantities [1]. The versatility of the instrumentation system is due to the internal bus protocol used. Its capability to support both analog and digital data transfers allows an optimum balance between functions that are performed at the analog level and those managed at the digital level. The border between analog and digital information processing can thus be adjusted for a minimum power consumption, a major requirement for the integration in a single package of the entire microsystem [2].

Based on the Manchester encoding scheme, this improved integrated smart sensor (I^2S^2) bus protocol is an upgrade of the basic integrated smart sensor (IS^2) bus [3], while remaining downward compatible. The basic IS^2 serial bus includes two communication wires and allows digital data transfer using the Manchester encoding scheme and semi-digital data transfer. The I^2S^2 bus is based on a single controller to coordinate the activity on the bus and includes a maskable interrupt mechanism, which makes it very suitable for implementation of event-triggered processes. It refines the protocol-level description for a set of generic commands, in order to allow a sensor-independent design of the bus drivers.

2. Design and implementation

A microinstrumentation system is characterized by a potentially high diversity of its subsystems. This imposes special requirements for exchanging data over a common internal bus. There can be modules which deliver only non-preprocessed analog signals, coexisting on the same platform with modules which manipulate data in digital format. The protocol used has to cope with this specificity. The normal standard interfaces, like the I^2C bus, universal serial bus (USB) [4] or the controller area network (CAN) bus [5], are designed to interconnect digital subsystems only. Their level of abstraction is too high for flexible smart silicon sensor implementations. Instead, in the present architecture, repre-

* Corresponding author. Tel.: +31-15-278-16-02; Fax: +31-15-278-57-55; E-mail: cretu@ei.et.tudelft.nl

0924-4247/98/$19.00
PII S0924-4247(98)00062-4

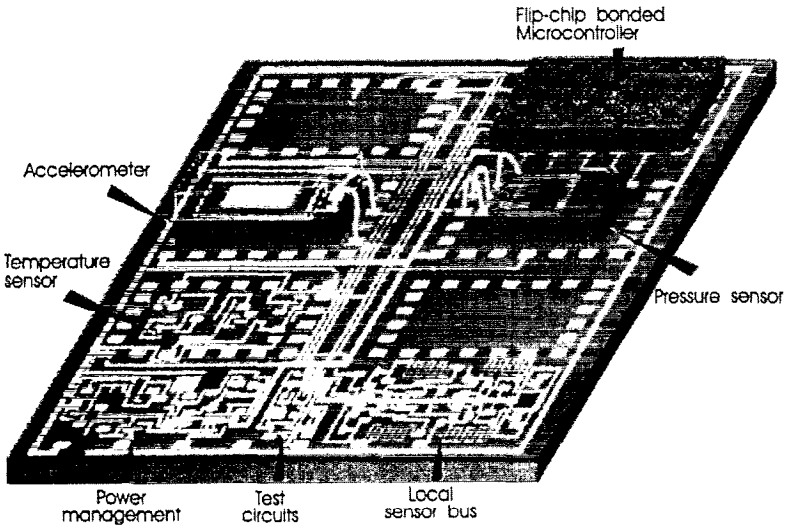

Fig. 1. Microinstrumentation platform.

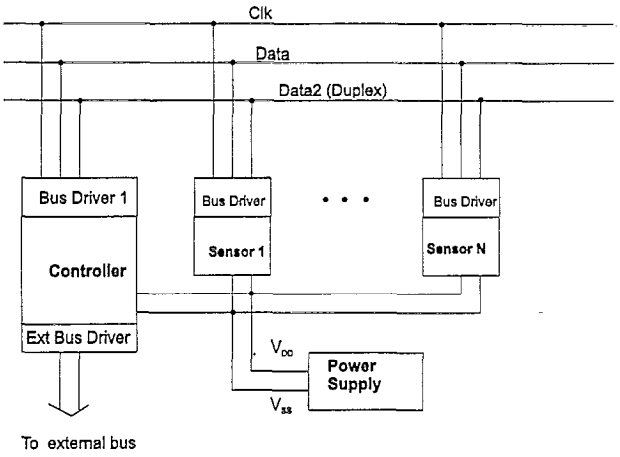

Fig. 2. General architecture of the microinstrumentation system.

sented in Fig. 2, a key issue was to concentrate most of the intelligence at the microcontroller level, and to optimize the sensors with a minimum of added on-chip electronics. The requirements imposed were:

- simplicity, for minimizing die-area consumption (for instance, to be a single master on the bus)
- a minimum number of wires
- to send/receive both analog and digital data
- on-line sensor calibration
- to allow both a polling mode, in which the master asks a specified device for data, and an event-triggered mode, in which a device signalizes the availability of data
- to let the master configure an addressed device or to read its state
- a simple and uniform way for handling bus conflicts.

It was convenient to treat the bus as having four hierarchical layers, at increasing levels of abstraction: mechanical and physical layer (number of wires, maximum distance between two consecutive units, etc.); electrical layer (input and output impedances, signal attributes, etc.); logical layer (binary representation); and protocol layer (associates a meaning with the transmitted signals).

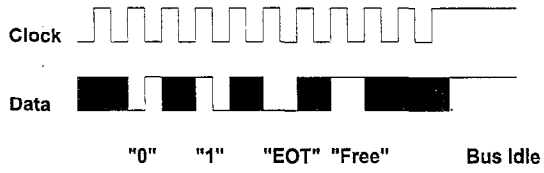

Fig. 3. Manchester code.

The starting point was the basic IS^2 bus [3]. It consists of two wires, a data line and a clock line, both open-drain driven. It offers a high flexibility, but it does not allow the simultaneous bi-directional data transfer that is needed, e.g., for on-line sensor calibration. To make this possible, a second data line was added. For backward compatibility, the second data line (calibration line) is used only when a feedback loop between a sensor and the local microcontroller is to be formed.

At the logical level, the Manchester encoding scheme for transmission of synchronous data allows a compact protocol. In such mapping, there are four available symbols (Fig. 3), so two of them were used for mapping of the logic symbols '0' and '1' and the remaining two were used as metasymbols (named 'Free' and 'EOT' (end of transmission)) for separating the messages and distinct fields inside a message frame.

The open-drain connection solves the bus arbitration problem at a low level of abstraction, simplifying the protocol. A low-level voltage will always be the dominant state, while the high level is a recessive state. This makes EOT the dominant symbol of the alphabet.

The basic idea is to use a variable-length frame for sending each message. At the beginning of the frame, the master puts a start bit (coded as '0'), followed by a synchronously transmitted fixed-length field. This has four bits for address specification and another four for the command coding. After sending the eight-bit-wide field, the master waits for the acknowledgment symbol from the receiver. The remainder of

Table 1
The codification of the command set

Tx	c_2	c_1	c_0	Command
0	0	0	0	get async. data
0	0	0	1	get sync. data
0	0	1	x	get status
0	1	0	x	unused
0	1	1	0	start async. duplex
0	1	1	1	start sync. duplex
1	IRqE	SRqE	0	set IRqE, SRqE and default config.
1	IRqE	SRqE	1	set IRqE, SRqE and a new config.

the frame is of variable length, until the master sends an EOT symbol. The set of commands must be general enough to let the design of the bus driver be independent of application-specific parts.

The correspondence between the command set and the four-bit-wide command field $Txc_2c_1c_0$ is presented in Table 1.

Outside of any frame, the bus is maintained in a low-power mode. This state (the 'Idle' state) corresponds to both clock and data lines being kept high.

Normally, the master initiates a frame, polling the sensors to acquire information or to configure them. There are two general cases in which such an approach interferes with proper system operation. First, in the case of rare events, there is the risk that either the microcontroller loses the information, or the power consumption necessary for monitoring it is too high. Secondly, there are critical events, like signalizing an on-chip high temperature, which have a high priority and must be immediately dealt with. These two cases justify the necessity of also having an event-triggered mode. In the simplest case, this can be used by a sensor module to announce to the controller when it has data available [6], or more generally, when some particular event has happened. In the developed interface this flexibility is obtained by adding interrupt-request (IRq) and service-request (SRq) protocols. An interrupt request has the meaning of an urgent message, so the sensor may signalize even in the middle of another frame. A service request has a lower priority; it may be sent only when the bus is in the idle state, that is, outside of any message frame. The interrupt- and service-request modes allow a flexible and low-power data handling for rare or very important events [7]. The handling of both interrupt and service requests by the controller is combined in a single mechanism, since both represent the same abstraction: a request message from a slave module. Except for the request messages, all the other messages are initiated by the controller and are frame oriented.

An example of a message frame is presented in Fig. 4. The local controller initiates the frame and asks the device $a_3a_2a_1a_0$ to send data on its asynchronous output. Then the device first puts on the data line an 'Acknowledge' bit (coded as the symbol '1'), and after it sends its data, until the controller forces an EOT symbol on the data line.

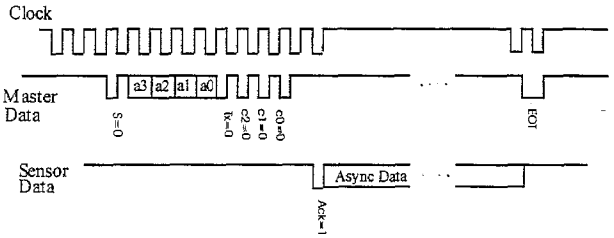

Fig. 4. A message frame for getting asynchronous data.

3. Device realization

The hardware structure of the bus driver is shown in Fig. 5. It has a modular structure, with a central 'Configuration Control' block as its core. The 'Sensor' block is application dependent, and not included in the bus driver. The only care taken was to ensure an interface compatibility between the bus driver and a diversity of sensors. The bus driver is composed from a set of synchronously interacting blocks, which reflect a function-oriented design style. Care was taken to reduce power wastage due to clocking of blocks which do not perform an active function at some specific moment. To manage the power consumption two modes are implemented:

- An idle mode for reducing the power dissipation in the standby mode. Waking-up is initiated only at the start of frame to enable verification of the address.
- Selective shut-down of different blocks based on the level of activity required to run a particular application. Different blocks of the chip may be idle for a certain period of time when running different applications (this happens with the service- and interrupt-request blocks).

The circuit was implemented in a 1.6 μm CMOS process. A photograph of the chip is presented in Fig. 6. The required area is 1 mm^2, and the measured power consumptions (for 5 V supply voltage) were less than 500 μW at 100 kHz and less than 2 mW at 4 MHz. A network with three bus interfaces was implemented and succesfully tested.

4. Experimental results

Fig. 7 shows a sample of oscilloscope traces during a message frame. The upper trace is the clock line, while the bottom trace represents the data-line voltage level. After two clock periods, in which the data line remained in the 'Free' state, the master starts a new message frame. First, it puts a start bit (0), followed by the address of the sensor (0000). In response to the master's request, the sensor '0000' confirms by putting '1' on the data bus. The master still controls data transfer by steering the clock. After that, it simply outputs a data bitstream (in the image, a string of zeros). The transmission is ended when the microcontroller forces an 'EOT' symbol on the data line. The hardware arbitration resulting from the open-drain logic ensures that the 'EOT' symbol is not overwritten by the signals sent by the sensor.

The response to a service request is recorded in Fig. 8. The controller starts a special 'interrogation frame' to find out the

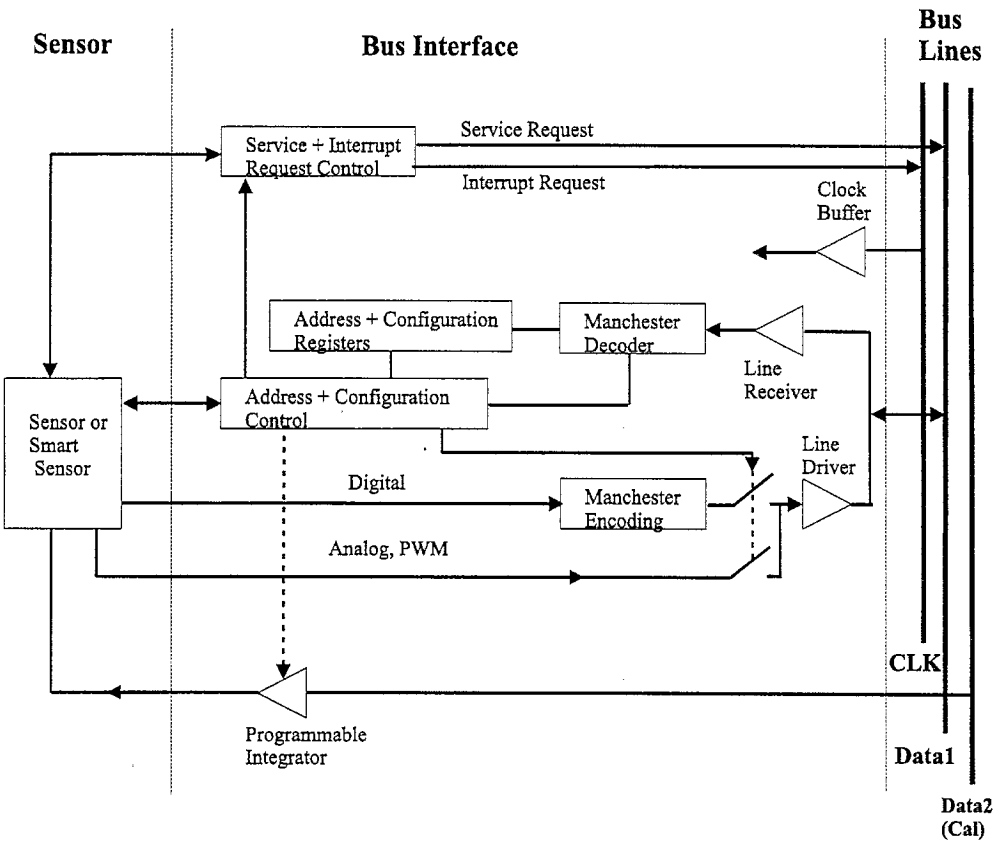

Fig. 5. Block diagram of the bus driver.

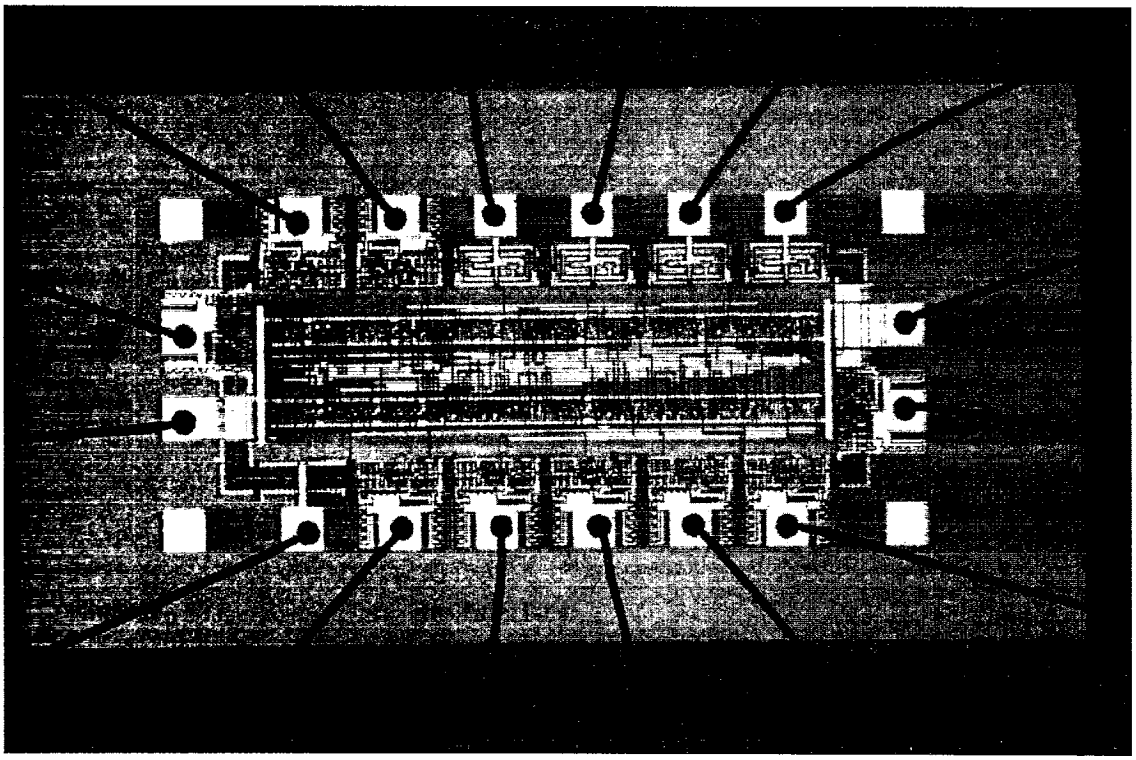

Fig. 6. A microphotograph of the bus interface.

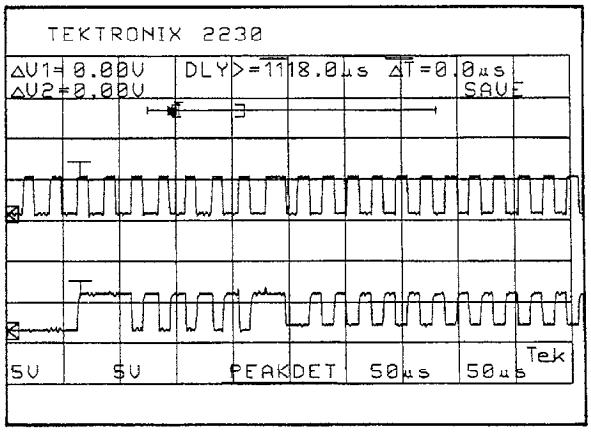

Fig. 7. Oscilloscope traces: reading synchronous data.

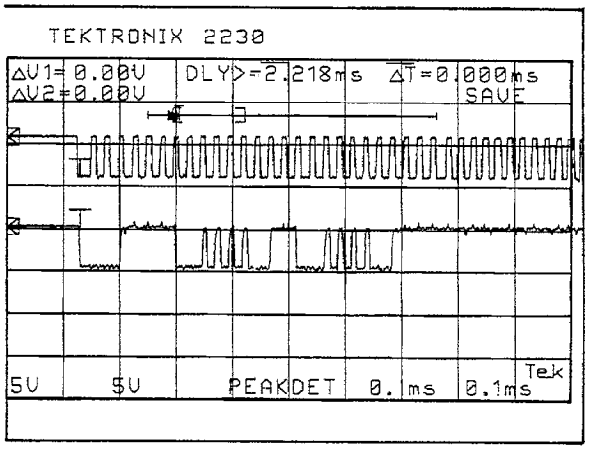

Fig. 8. Oscilloscope record of a service request mechanism.

address of the requester module. If multiple sensors signalized a request before the interrogation frame, then the zero-winning hardware arbitration mechanism establishes an address priority. In the recorded case, the address received by the controller was '1111'. Finally, the controller ends the frame in the normal way, with an EOT symbol.

5. Conclusions

The improved integrated smart sensor (I^2S^2) bus presented here was designed for an optimum flexibility/simplicity compromise. The goal is to use it in general MCM-based microinstrumentation systems. Both the protocol and the hardware implementation were designed for both separate realization and co-integration with a wide range of sensors, without causing fabrication compatibility infringements. The main features of the bus protocol are:

- simplicity of structure: only two communication wires are used in the minimum configuration
- reliable data transfer by using the Manchester encoding scheme
- flexibility of signal type, as synchronous and asynchronous transmission of digital data is possible in combination with semi-digital signals, such as bitstreams, or even analog signals
- flexibility of signal handling based on a maskable interrupt mechanism
- sensor self-test capability over the bus using separate directional data lines
- the bus drivers can be used either as separate dies in microsystems or on-chip integrated with sensors.

Such an interface is suitable for designing application-specific integrated microinstrumentation systems, containing application-independent parts, which can be made as standard modules. The user will decide the type of sensors to be used, which will determine the specificity of the system. At present the focus is toward the design of two microinstrumentation systems using I^2S^2 as an internal bus: a miniature spectrometer and a condition monitoring system. Both involve a mixture between analog and digital processing. The benefits of transferring both analog and digital data over the bus can thus be exploited. The low power consumption required for the bus driver makes this solution very attractive for integrated microsystems.

Acknowledgements

This work is supported in part by STW (project DEL 55.3733), TU Delft and JNICT – Portugal (Program Praxis XXI-BD/5181/95).

References

[1] E. Cretu, J.H. Correia, S.H. Kong, M. Bartek, R.F. Wolffenbuttel, Flexible architecture for microinstrumentation systems in silicon, Proc. PRORISC, 8th Workshop CSSP 97, 27–28 Nov., 1997, Mierlo, The Netherlands.

[2] J. H. Correia, E. Cretu, M. Bartek, R.F. Wolffenbuttel, A low-power low-voltage digital bus interface for MCM-based microsystems, Proc. 23rd Eur. Solid-State Circuits Conf. (ESSCIRC '97), Southampton, UK, 16–18 Sept., 1997, pp. 116–119.

[3] J.H. Huijsing, F. Riedijk, G. vd. Horn, Developments in integrated smart sensors, Proc. 7th Int. Conf. Solid-State Sensors and Actuators (Transducers '93), Yokohama, Japan, 7–10 June, 1993, pp. 320–326.

[4] M. Talbot, R. Nayer, CAN and USB – the serial future, Electronic Eng. 69 (1997) 63–64.

[5] S. Josifovska, Understanding CAN, Electronics World 103 (1997) 874–876.

[6] J. Bryzek, Introduction to IEEE-P1451, the emerging hardware independent communication standard for smart transducers, Sensors and Actuators A 62 (1997) 711–723.

[7] A. Mason, N. Yazdi, K. Najafi, K. D. Wise, A low-power wireless microinstrumentation system for environmental monitoring, Tech. Digest, 8th Int. Conf. Solid-State Sensors and Actuators (Transducers '95/Eurosensors IX), Stockholm, Sweden, 25–29 June, 1995, pp. 107–110.

Biographies

José Higino Correia graduated in physical engineering from the University of Coimbra, Portugal, in 1990. He has been a lecturer in the Department of Industrial Electronics at the University of Minho, Portugal, since 1991. At this university he obtained the PAPCC in electronics and instrumentation (equivalent to an M.Sc. degree) in 1994. Since 1995 he has been at the Laboratory for Electronic Instrumentation, Delft University of Technology, working toward a Ph.D. degree in the field of mycrosystems for optical spectrum analysis.

Edmond Cretu was born in Bucharest, Romania, in 1965. He received the M.Sc. degree in electronic engineering from the 'Politehnica' University, Bucharest, Romania, in 1989. He worked as a researcher in the Romanian Academy of Science and as an associate assistant at the Faculty of Electrical Engineering, 'Politehnica' University. He is currently completing his Ph.D. thesis at the Laboratory for Electronic Instrumentation, Delft University of Technology, Netherlands. His research interests include the design of a microinstrumentation system for condition monitoring and modern methods of vibration analysis.

Marian Bartek was born in Trnava, Slovakia, on 18 March, 1965. He received the M.Sc. degree (cum laude) in electrical engineering in 1988 from the Electrical Engineering Department of the Slovak Technical University, Bratislava, Slovakia. His master's thesis dealt with liquid-phase epitaxy of InGaAsP quaternary layers for optoelectronic applications. After completion of his military service in October 1989, he continued this research at the Electrical Engineering Department of the Slovak Technical University as a research assistant. In August 1991 he joined the Electronic Instrumentation Laboratory, Department of Electrical Engineering at the Delft University of Technology, Netherlands, where in 1995 he obtained the Ph.D. degree for research on selective epitaxial growth for smart silicon sensor applications. Currently, he is a research fellow at the same laboratory and his work deals with technological aspects of integrated silicon sensor systems.

Reinoud F. Wolffenbuttel received his M.Sc. degree in 1984 and his Ph.D. degree in 1988, both from the Delft University of Technology. His thesis work dealt with the application of silicon to colour sensing. Between 1986 and 1993 he was an assistant professor and since 1993 he has been an associate professor at the Laboratory of Electronic Instrumentation of Delft University of Technology, where he is involved in instrumentation and measurement in general and on-chip functional integration of microelectronic circuits and silicon sensors, fabrication compatibility issues and micromachining in silicon and microsystems in particular. In 1992/93 he was a visiting scientist at the University of Michigan, Ann Arbor, MI, USA, and was involved in research on low-temperature wafer-to-wafer bonding.