

# A Local Bus for MCM-Based Microinstrumentation Systems

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

Delft University of Technology, Dept. of Electrical Engineering,

Lab. for Electronic Instrumentation/DIMES,

Mekelweg 4, 2628 CD, Delft, The Netherlands

## Abstract

A local smart sensor bus is described which is designed for use in a multi-chip-composed microinstrumentation system. The bus is able to transmit a digital code, bitstream, analog voltage, frequency, duty-cycle and also provides calibration facilities, service request and interrupt request modes. The resulting sensor bus interface was implemented in a 1.6  $\mu\text{m}$  CMOS process and successfully tested in a local sensor network.

---

\*Corresponding author. Phone: +31-152781602. Fax:+31-152785755. Email:  
cretu@ei.et.tudelft.nl

## List of Figures

1	Microinstrumentation platform . . . . .	17
2	General architecture of the microinstrumentation system . . . . .	18
3	Manchester code . . . . .	19
4	A message frame for getting asynchronous data . . . . .	20
5	Block diagram of the bus driver . . . . .	21
6	A microphotograph of the bus interface . . . . .	22
7	Oscilloscope traces - reading synchronous data . . . . .	23
8	Oscilloscope record of a service request mechanism . . . . .	24

## List of Tables

1	The codification of the command set . . . . .	25
---	---	----

*Keywords* : instrumentation microsystems; internal bus interface; 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 mar-

ket favours the design of a flexible architecture, reusable for different applications. The concept of ASIC (Application Specific Integrated Circuits) 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 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 the 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 be thus 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 ISS serial bus includes two communication wires and allows digital data transfer using the Manchester encoding scheme and semi-digital data transfer. The improved version of the ISS 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 I<sup>2</sup>C 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 Si sensor implementations. Instead, in the present

architecture, represented 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 signals 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 the protocol layer (associates a meaning with the transmitted signals).

The starting point was the basic ISS 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 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 put 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 8 bit-wide field, the master waits for the acknowledgment symbol from the receiver. The remaining of the frame is of variable length, until the master sends an **EOT** (End of Transmission) symbol. The set of commands must be general enough to let the design of the bus driver independent of application-specific parts.

The correspondence between the command set and the four bit wide command field  $Tx\ c_2\ c_1\ c_0$  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. Firstly, in the case of rare events, there is either the risk that the microcontroller loses the information, or the power consumption necessary for monitoring it is too high. Secondly, there are critical events, like signaling an on-chip high temperature, which have a high priority and must be immediately dealt with. These two cases justifies the necessity of having also an event-triggered mode. In the simplest case, this can be used by a sensor module to announce the controller when

it has data available [6], or more generally, when some particular event happened. In the developed interface this flexibility is obtained by adding an interrupt request (IRq) and a service request (SRq) protocol. 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 idle state, that is, outside of any message frame. The interrupt and service request modes allows 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 are 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  $a_3 a_2 a_1 a_0$  to send data on its asynchronous output. Then the device firstly 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.



### 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 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 waste 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 saving of 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  $\mu\text{m}$  CMOS process. A photograph of the chip is presented in Fig. 6. The required area is 1 mm<sup>2</sup>, and the mea-

sured power consumptions (for 5 V supply voltage) were less than 500  $\mu$ W at 100 kHz and less than 2 mW at 4 MHz. A network with three bus interfaces was implemented and successfully 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 2 clock periods in which the data line remained in "Free" state, the master starts a new message frame. Firstly, it puts a start bit ('0'), followed by the address of the sensor ('0000'). In response to 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 bit stream (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 resulted 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 address of the requester module. If multiple sensors signalled 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<sup>2</sup>S<sup>2</sup>) bus presented aimed at an optimum compromise flexibility/simplicity. 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 designing of two microinstrumentation systems using I<sup>2</sup>S<sup>2</sup> as internal bus: a miniature spectrometer and a condition monitoring system. Both involves a mixture between analog and digital processing. The benefits of transferring both analog and digital data over the bus can be thus exploited. The low power consumption required for the bus driver makes this solution very attractive for integrated microsystems.

## 6 Acknowledgements

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

## 7 Biographies

*José Higinio Correia* graduated in Physical Engineering from the University of Coimbra, Portugal, in 1990. He is a Lecturer in Dept. of Industrial Electronics at University of Minho, Portugal, since 1991. In this university he obtained the PAPCC in Electronics and Instrumentation (equivalent to 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 mycosystems for optical spectrum analysis.

*Edmond Cretu* was born in Bucharest, Romania, in 1965. He received the M.Sc. degree in electronic engineering from '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 In the Laboratory for Electronic Instrumentation, Delft University of Technology, the 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 March 18, 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, the Netherlands, where he, in 1995, 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 has been an assistant professor and since 1993 an associate professor at the Laboratory of Electronic Instrumentation of the Delft University of Technology and is involved in instrumentation and measurement in general and on-chip functional integration of microelectronic circuits and silicon sensor, 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 Michigan, USA, and was involved in the research on low-temperature wafer-to-wafer bonding.

## References

- [1] E. Cretu, J.H. Correia, S.H. Kong, M. Bartek and R.F. Wolffenbuttel, "Flexible Architecture for Microinstrumentation Systems in Silicon", in Proc. PRORISC, The 8th workshop 'CSSP 97', November 27-28, 1997, Mierlo, The Netherlands
- [2] J. H. Correia, E. Cretu, M. Bartek and R.F. Wolffenbuttel, "A low-power low-voltage digital bus interface for MCM-based microsystems", in Proc. 23rd European Solid-State Circuits Conf. (ESSCIRC'97), Southampton, UK, September 16-18, pp. 116-119
- [3] J.H. Huijsing, F. Riedijk and G.vd. Horn, "Developments in integrated smart sensors", in Proc. Transducers'93, pp. 320-326
- [4] M. Talbot, R. Nayer, "CAN and USB - the serial future", in Electronic Engineering, Vol. 69, No. 849 (September 1997) , pp 63-64
- [5] S. Josifovska, "Understanding CAN", in Electronics World, Vol. 103, No. 1738 (October 1997), pp. 874-876
- [6] J. Bryzek, Introduction to IEEE-P1451 the emerging hardware independent communication standard for smart transducers, Proc. Eurosensors X - 1996 Belgium, pp.853-876.

- [7] A. Mason, N. Yazdi, K. Najafi, K. D. Wise, A low-power wireless microinstrumentation system for environmental monitoring, Proc. Transducers 95, pp. 107-110.



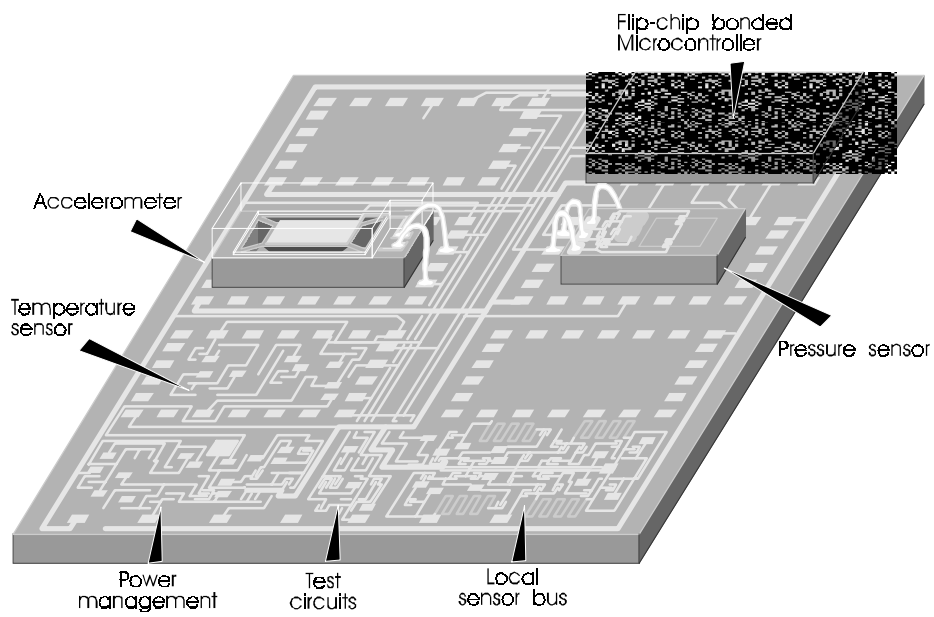


Figure 1: Microinstrumentation platform

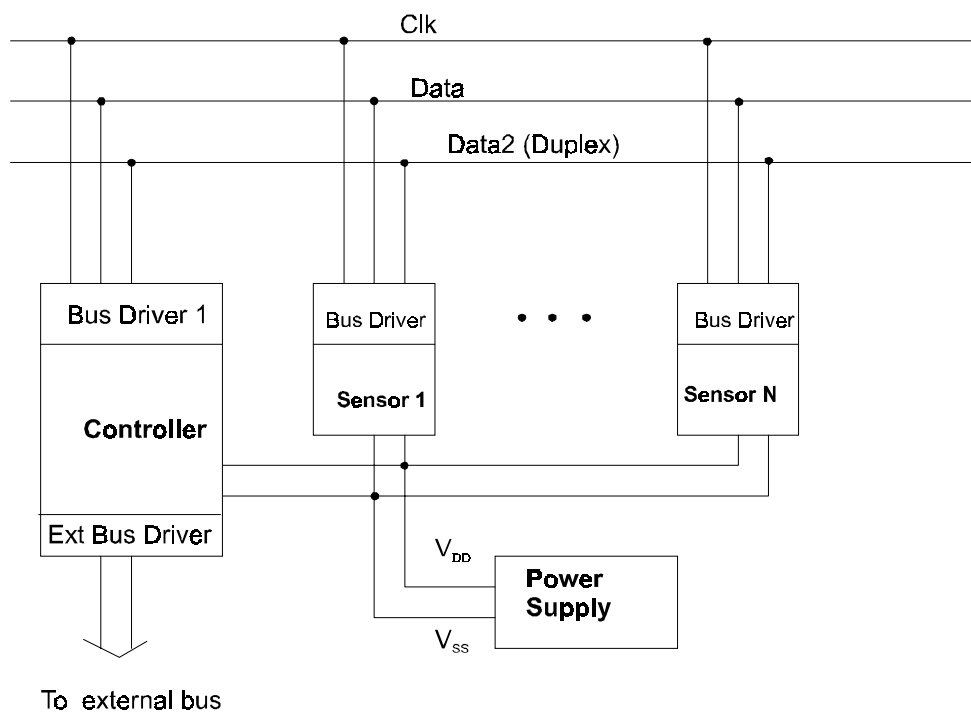


Figure 2: General architecture of the microinstrumentation system

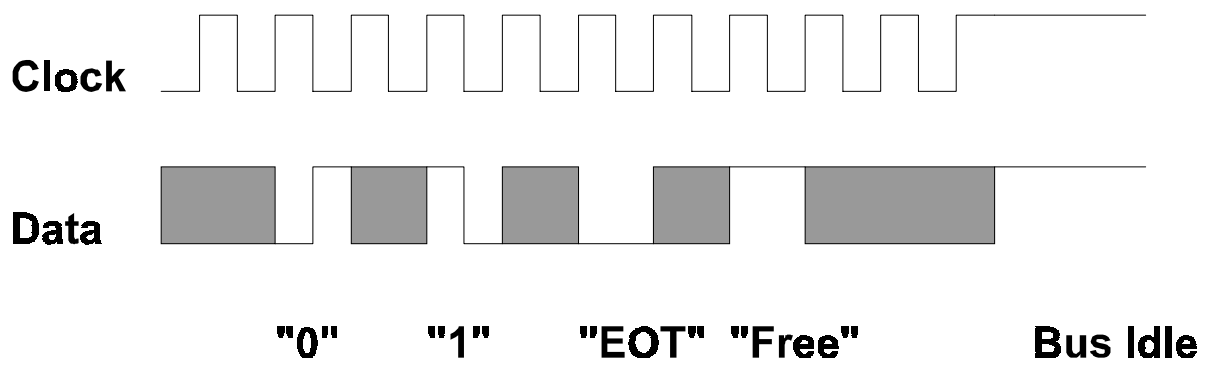


Figure 3: Manchester code

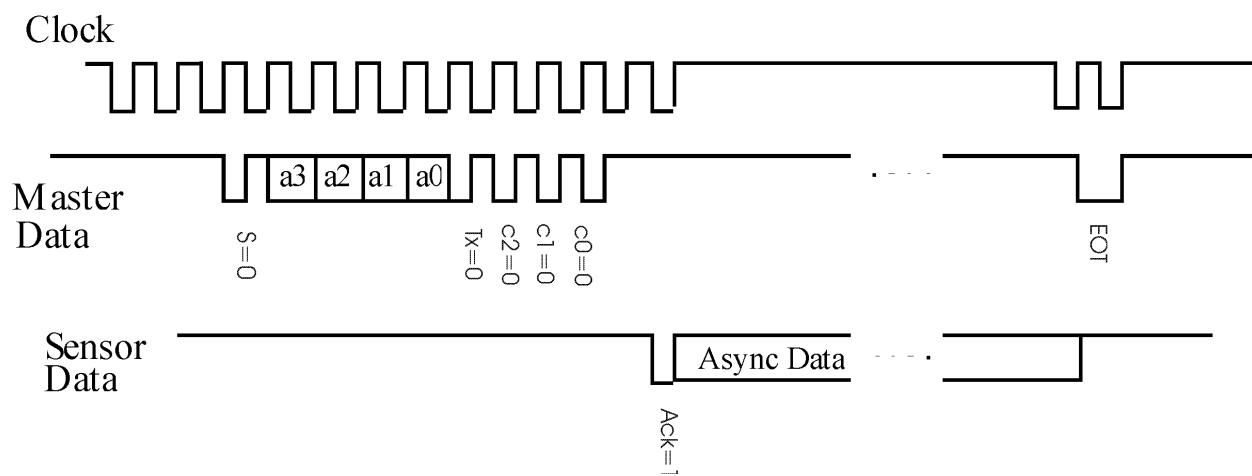


Figure 4: A message frame for getting asynchronous data

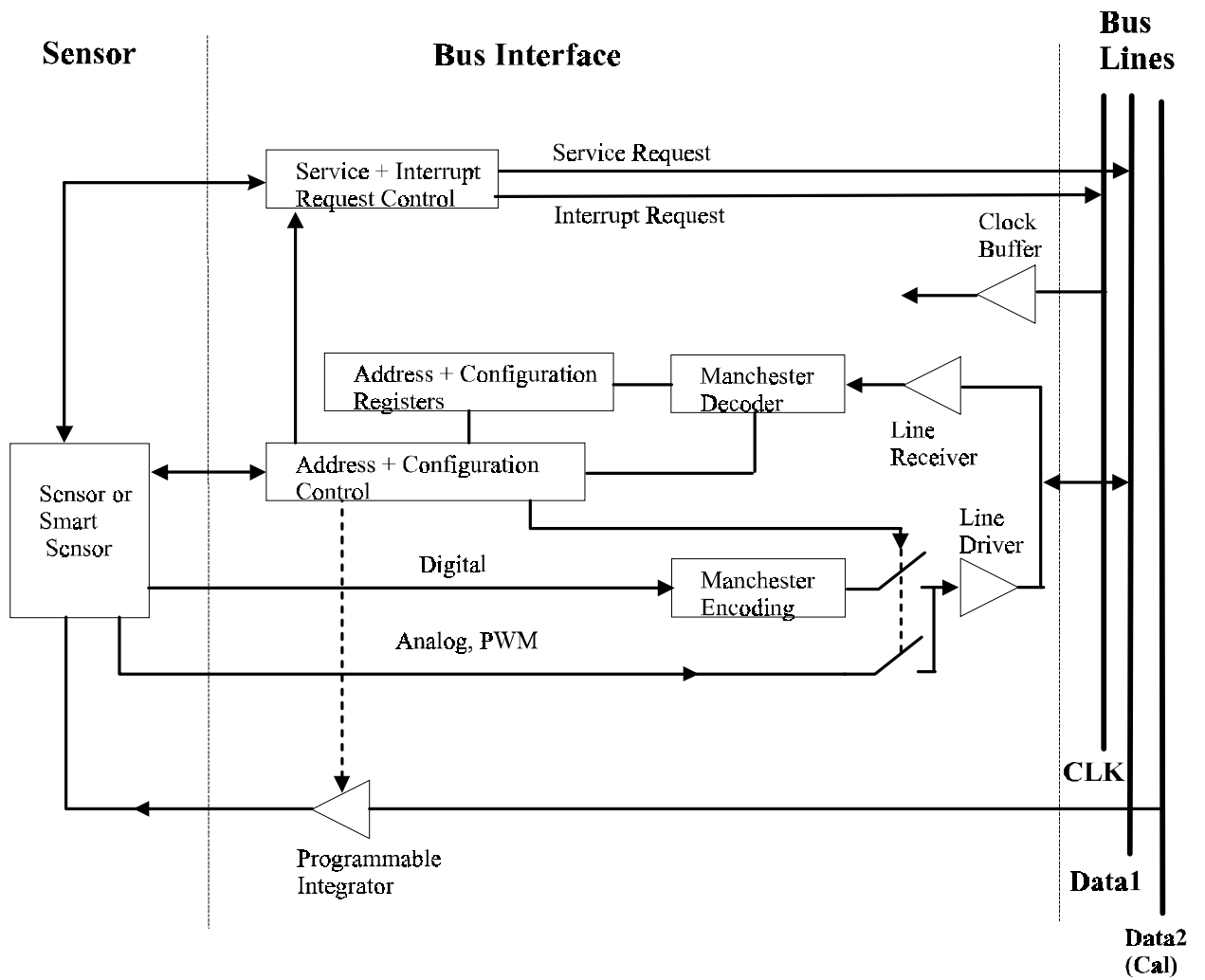


Figure 5: Block diagram of the bus driver

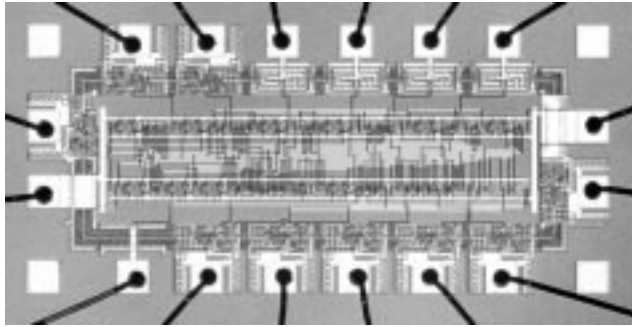


Figure 6: A microphotograph of the bus interface

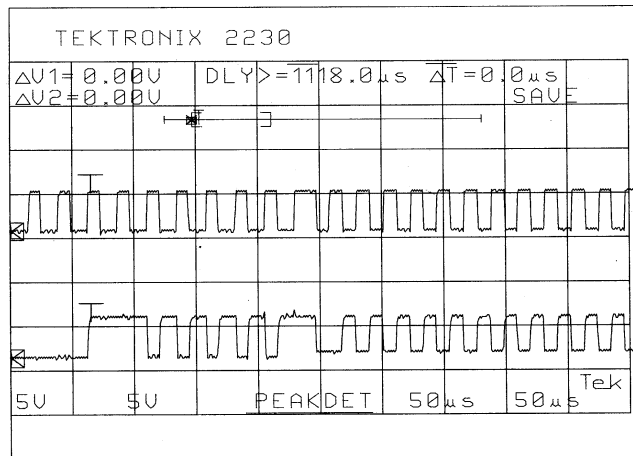


Figure 7: Oscilloscope traces - reading synchronous data

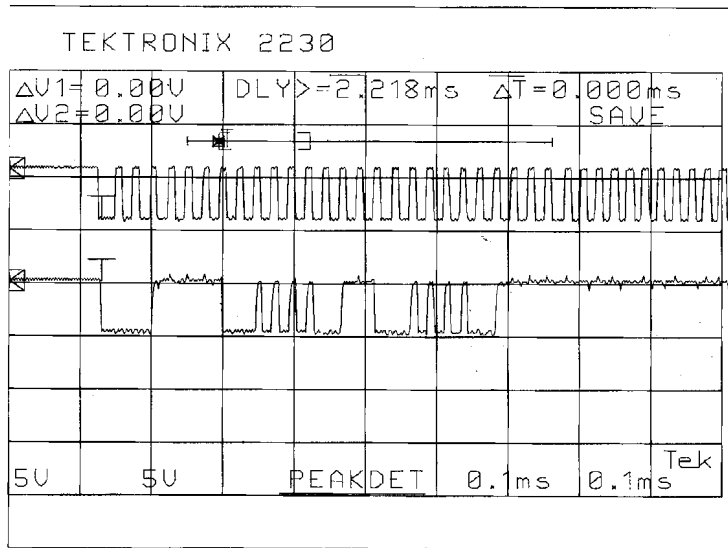


Figure 8: Oscilloscope record of a service request mechanism



Table 1: The codification of the command set

Tx	c <sub>2</sub>	c <sub>1</sub>	c <sub>0</sub>	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.