An economical approach for small sized automation tasks

Rolf Biesenbach[1], Albrecht Weinert[2]
[1,2] Faculty of Electrical Engineering and Computer Science
Bochum University of Applied Sciences, Bochum, Germany

ABSTRACT

Introducing automation distributed in small (sub-) systems will be an enabler to more functions, comfort, energy savings, growth of quality and other advantages in many fields so far not at all or only sparsely or automated. The latter is the case in small buildings and private homes. Here it is essential that the extra automation equipment brought in is sparing with energy and other resources. This is essential for acceptance per se. And it would make no sense if the costs of equipment exceed the savings achieved by automation.
Using common lean micro-controllers or tiny computer/controller boards usual in trade with open software would accomplish the energy and cost savings on the one hand. Insisting on established automation standards on the other hand forbids their naive (wide spread) use. To handle this conflict the specifications for a suitable approach is presented.
This specification was realized in an evaluation board as a proof of concept. The hardware complexity of interfaces goes far beyond the majority of small computer/micro-controller boards and is comparable to industrial PLC (programmable logic controller, small industrial automation system) process I/O (input/output) modules. The complexity of the software stems from bringing PLC tasks, services and protocols to a small micro-controller as well as from some sophisticated algorithms to make that work on quite small computing power.

1. INTRODUCTION

1.1. Motivation

There are many fields of application suitable for small or distributed automation (modules) in which one virtually finds none. Just one exemplary field are buildings and especially (private) home automation.
In this exemplary field proprietors and lodgers see growing costs of energy and all other utilities. In the meantime these are often higher than the building costs (depreciation, amortization, tax). As further growing of energy and public utilities prices is probable, reducing their consumption seems the only escape. But in modern homes thermal insulation of walls, doors and windows is up-to-date, as are the heating equipment, the dish washer, the computers, TVs and else. And if 90 % of the lighting is by LEDs (Light Emitting Diode) the only source of further savings is either a painful reduction of living comfort and quality or introducing a smart/ intelligent management of lighting, heating, ventilation, doors, shutters etc. The latter we can as well call small scale automation.
Obviously, the line of arguments, the specification presented as well as a realization according to the proof of concept can be used in other fields of application including logistics, transport, vehicles and many more. For sake of brevity and universal awareness we'll stay with the home automation example.

1.2. Commercial approaches to home automation

There are commercial home automation solutions, some of which are based on specialized electrical supply wiring and replacements for most wall switches and outlets. The features of those solutions are often appealing but come at high extra prices. The approaches are kind of “all or nothing” suitable for new buildings or complete refurbishing but not for retrofitting. And being mostly proprietary they neither do interface well with other equipment nor build they up any pressure to standardize those interfaces and protocols.
To experience the last point, ask your heating/cooling equipment manufacturer for the communication/control interfaces. Usually the answer is in the range “none we'll tell you about” to “none”. And sometimes the question is not understood. Externally controlling a gas or oil fueled central heater, e. g., by a home automation system, usually involves a cheating proxy to its sensors. An example, equally bad, are solar panel inverters, coming with a secret ZigBee protocol (Enecsys e. g.), only exchanging data with the vendor's server after the owner bought an extra router (for about 170 €).

1.3. Existing DIY approaches

Some commercial approaches (powerNet, iBus) are scarce in private homes. The main reasons are lacking awareness with the proprietors, the craftsmen/builders and their price. It is sensed as too expensive either at all, compared to their features or at building construction or renovation time.
The desire to fit some smartness to their homes later on, is often felt by some enthusiasts with some engineering or IT (information technology, computer science) background. This quite seldom leads to the deployment of industrial automation systems or PLCs (programmable logic controller, small industrial automation system). The reasons against are a) the price felt to be too high – often disregarding or not understanding the purchased quality and standard conformity [20] and b) the programming tools and languages quite behind state of IT art.
This situation opens a gateway for micro-electronic boards into home automation. Some of them got a kind of open standard status by support of some distributors and electronics magazines. Notwithstanding the often good quality as product their DIY (do it yourself) deployment leads to all problems named above. A wireless system to replace standard electrical installation, for example, used in noteworthy numbers by the German scene, due to lacking standard conformance, failed with the advent of LTE (Long Term Evolution) cellular phone infrastructure in the neighborhood.

1.4. Bringing it together

The predominant absence of desirable standardized automation in a field or market tends to induce a confusing plenty of diverse approaches offered by various (non-automation) vendors, usually labeled smart or intelligent. Most of them, of course, are compatible to nought and often non-compliant to even the most modest electrical, interference, communication, handling and, regrettably, electrical safety or EMC (Electromagnetic compatibility) [24] standards.
It seems desirable to combine small modular inexpensive controllers and open software with industrial automation standards to bring the latter to new application fields and markets not ex-

cluding a DIY scene. To this end we specify a small automation module and show a proof of concept realization.

2. SPECIFICATION

We specify an appliance for digital control and automation as well as a basic operating system or run-time software.

2.1. Hardware oriented requirements

- industry standard voltages (12 V, 24 V) for supply and process I/O (input/ output) [17..19]
- wide supply voltage range for the process I/O, also known as “load voltage” or LV for short
- 9..30 V (LV), this range covering both 12V (home, facilities, small vehicles) and 24V (industrial automation, lorries, trucks) applications including all required tolerances (19 .. 27V e. g.).
- optional second redundant 9...30 V feed for the electronics, being suitable for uninterrupted battery supply or even AC (12..18 V~, alternating current)
- buffering of 20 ms supply outages without reset/restart
- separation of protecting (PE, protecting earth) and signal/supply ground (Gnd)
- fit for "green automation" thanks to low power consumption and by an efficient run-time software
- 8 digital output (DO) channels (LV, 100 mA, protected)
- 8 protected input channels for nominal +/-70 V input voltage (surviving 250 V~eff as absolute max. rating

Every input channel can be configured either as

- digital input in three threshold/hysteresis modes for different (12 V/24 V) sensors or as
- analogue input in three different single ended voltage ranges (3..50V)

Standard communication links connect to other devices, a central controlling computer, to a human (terminal) or else:

- Ethernet [2] and
- V.24/RS232, offering all usual modes, baud-rates up to 500 KBaud and optional flow control
- extensibility by SPI (Serial Peripheral Interface Bus), I²C (twoWire, [21]) etc.
- widely-used micro-controller/architecture [22] with a broad base of tried and tested software and tools
- on board/in system programming (ISP)
- persistent storage by embedded EEPROM (Electrically Erasable Programmable Read-Only Memory) and a slot for small memory cards (SMCs)

Being compact and low power (green automation) we nevertheless have to have some other typical PLC features:

- protected process I/O with error detection (overload, over-temperature, wire breakage, short-circuit)
- supervision of its supply (LV)
- status LEDs for all (16) DI/AI/DO
- status respectively error LEDs for supplies and drivers

2.2. Software/run-time oriented requirements

Besides the possibility to program the hardware's processor from scratch to implement the tasks in question, there has to be a powerful suitable run-time to support the user in his application/automation tasks. As these are often “endless” the run-time or operation system (OS) should offer the approved approach of the

- usual automation "cycles" (1 ms, 10 ms, 100 ms, 1 s) known from most PLCs, multi-treading [17, 18]

Besides that all hardware features specified above – under 2.1 – have to be supported by:

- handling of process inputs – analogue, digital, counter etc. – i. e. a good support for all available process I/O variants as well as some optional filtering
- handling and supervising of digital process outputs including supervision for overload over-temperature
- controlling the supervision of the load supply (LV)
- watchdog supervision of application/user software
- logging of (last) re-start cause
- handling of the optionally inserted small memory card providing the basic functions of the file system (FAT32, file allocation table 32 bit cluster address) prevalent there
- handling of the Ethernet communication [2], providing a set of protocol implementations: DHCP (Dynamic Host Configuration Protocol) [9-11], Telnet (server, [4]), Modbus (server [12-14]), NTP (net time protocol) [5-8]
- handing of communication and command line interfaces (CLI) be it via serial communication (V.24, RS232) or by Ethernet/Telnet
- handling of date, time, zones and daylight saving, preferably by information got as DHCP or NTP client in good accuracy and 1 ms resolution
- providing both absolute (date time) and relative (duration) timers [17]

3. REALIZATION

3.1. Hardware

The hardware concept shown as block structure in figure 1 is based on an AVR Atmel 8 bit micro-controller 1284P [22]. This is a RISC (reduced instruction set computer) in Harvard architecture [23]. The latter means a structural separation of program and data memory marked “Flash” respectively “RAM” (random access memory) in figure 1. For the automation applications the program (flash) memory being non volatile and practically read only (RO) is an advantage for long term automation applications.

Most input, supervising and timing is done by the 1284's integrated peripheral features. All blocks in the upper line and in both outer columns in figure 1 are integrated in the controller chip. These blocks are supplemented by an Ethernet chip (28J60) and an industrial 8 channel supervised and protected digital output (DO) driver, both controlled via SPI (serial peripheral interface).

SPI is also used to interface the small memory card (SMC, [15, 16]). SMC is better known as micro-SD (Secure Digital) memory card or “µSD”. (But it may not be called so and only used in SPI mode to avoid licensing troubles – hence we use SMC). An optionally inserted SMC and more so the EEPROM are used for non volatile configuration and logging data.

The 8 channel digital/analogue input (DI/AI) as specified is done by suitable controller ports furnished with elaborate (and spacy) protection circuits.

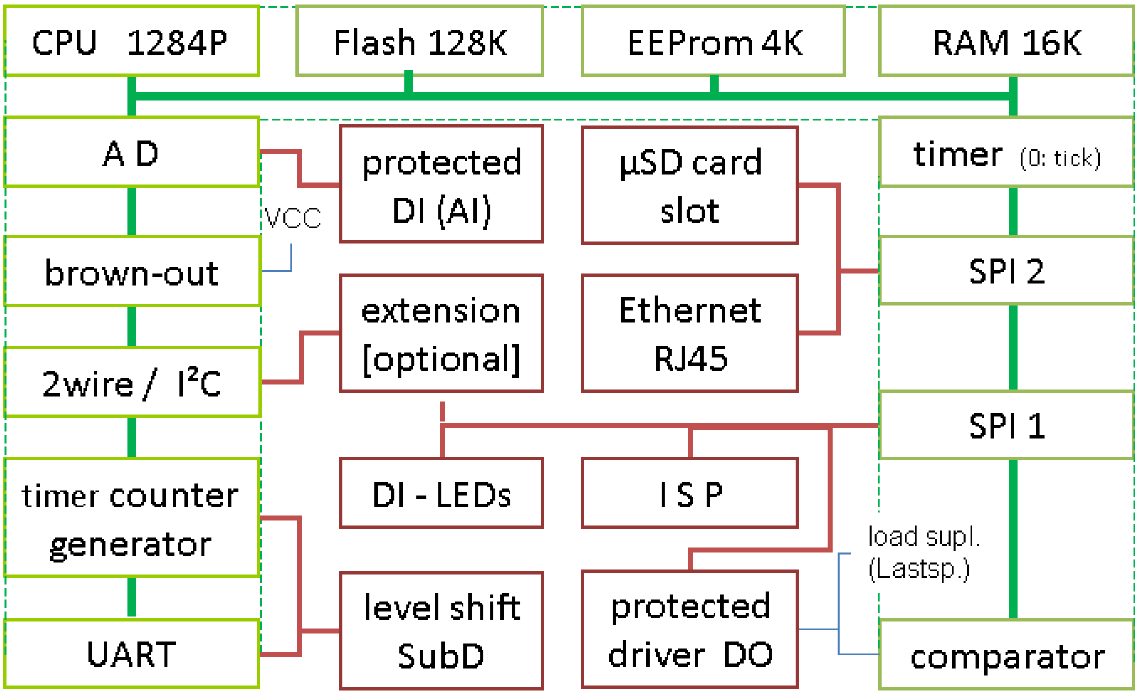

Figure 1. *Block structure of the concept realized*.

Figure 2 shows the proof of concept respectively evaluation board (called "weAut_01" [26]). This appliance for digital control and automation is a (single board) micro-controller based module fulfilling all requirements of chapter 2.1. The compact board fits in in a wide range of compact cases, including some DIN rail (TS 35)/top-hat rail boxes. If the interface to the outside is restricted to the connectors at the upper and lower edge (figure 2) – preferably by suitable housing – no controller or other device's pin susceptible to EMI damage will be exposed [24, 25].

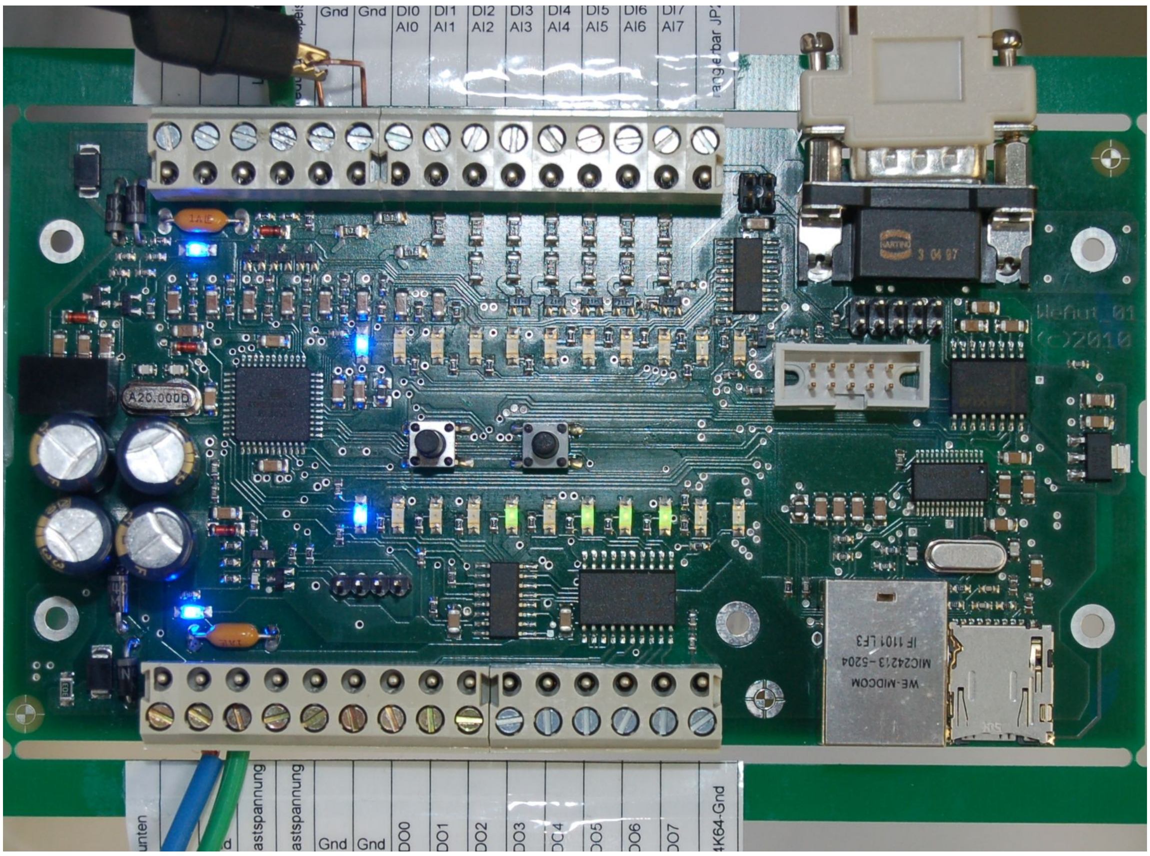

Figure 2. *Proof of concept – evaluation board*.

3.2. Software/run-time

A run-time (called "weAutSys") was created, the main but not only target of which is the board presented. It fulfills all specifications given above. Written completely in C [28, 29] it adapts three pieces of professional open source software with a non-infecting license – i. e. of a type allowing commercial use.
Multitasking is needed for the specified system and user "cycles" – 1 s, 100 ms, 10 ms and 1 ms. ("Cycles" is automation speak, IT people call such "periodic tasks".) Considering the intentionally small processor all usual approaches of task/context switching and in consequence locking and synchronizing would be more engaging than the productive/application code. Hence a non-preemptive run-time based on Adam Dunkels' ingenious Protothreads [1] was created reducing that ballast to virtually zero. The price is a required programming discipline: one has to yield every thread as often as one third of fastest required cycle. It is quite easily done, just keep it in mind.
Considering again the small and slow processor some effort was put in the run-time's library to give the programmer optimized algorithms for formatting, parsing, and much more. This includes frequent arithmetic operations [3]. Keep in mind that the RISC processor can neither divide nor has multi-bit shift operations [23]. The AVR/GNU C library [29] respectively GNU-compiler [28] workarounds are perfectly correct but often sub-optimal in processor load (GNU "GNU's Not Unix!", an open source operating system).
The second open source package incorporated was the TCP/IP (Transmission Control Protocol / Internet Protocol) stack named "uIP", again by Adam Dunkels [2]. The main burden here was creating a suitable driver for the Ethernet-Chip (ENC28J60) used. ChaN's open source FAT32 implementation named "fatFS" served as starting point for the file system implementation for SMCs interfaced by ISP.

4. DISCUSSION

We presented a specification of an automation module combining industry automation proceedings, interfaces and standards with sparing and "green" micro-controller technique as well as a proof of concept realization in form of an evaluation board and run-time. Compared to automation systems respectively PLCs we get a lower price for a board as well as the usability of free standard IT software tools [28, 29]. Especially when optionally using the run-time [27] provided, the application programmer has PLC cycles, timers, all mentioned protocols, communication and CLI at hand and writes his applications as just a few small C functions.

TABLE 1: COMPARING PRIZE, SIZE AND SOME OTHER BASIC DATA

Property	weAut_01	Raspberry Pi
Price (€)	170	30
units made & sold	100	millions
size (mm * mm)	82 * 151	86 * 54
supply voltage (V)	9..30	5
supply power (W)	1.5	3.5
memory (Byte)	144 K	512 M
CPU clock (MHz)	20	700

Comparing the automation board with popular micro-controller boards, like the most popular Raspberry Pi single board computer, is to compare apples and oranges. We nevertheless do it for two reasons. In some fields as in the home automation example used throughout here we see those single board computers and not PLCs penetrating that market. And the second reason is this comparison being just done – naively and at first sight – by most people: Why should one pay a sixfold price and having thrice the board area for comparably wee computing power?
Of course, in our use cases, like digging a board in a rolling shutters compartment, to remote control/automate four of them e. g., the 8 bit AVR's computing "power" is more than enough. Any more is taking a sledgehammer to crack a nut. Nevertheless it has to be admitted that with heavy file operations on the memory card e. g. one gets to its limits. The consumption of less than half the power is achieved without opening the wide bag of tricks, of putting unused parts of the micro-controller or communication electronics to sleep. Utilizing those possibilities one can bridge long power outages on small batteries.
The price in money and board area is, of course, paid for process signal treatment, protection circuits and flexible buffered and redundant power supplies as well as for cage or screw clamps for external wiring. The components therefore are usually more expensive than the micro-controller circuit. For popular mini computer boards you can find some process signal extensions, seldom offering conformance to settled habits of industrial automation. Getting not the number of items, and having to use expensive components, they often more than triple the mainboard's price.
Adding those considerations to the comparison would bring the specification from chapter 2.1 to Table I. That seems unfair in the other way round as, of course, the exemplary Raspberry Pi and most other tiny computer/controller boards were never designed with such requirements in mind. So in the light of automation, process signal handling, electrical robustness, supply buffering etc. the approach presented here must not eschew the comparison.
Of course, there were discussions on these quite specific requirements, too: more or fewer I/O channels, fewer or more interfaces, faster processor or Ethernet. The points most controversial were the omission of the V.24 interface (saving 8 € and some space) and the request for (24V) analogue output (AO) channels not realized here.
The point for V.24 was the still habitual use while commissioning and the existence of some peripheral devices controlled via that interface. The point against (quite expensive) 24V AO was the lacking need in our use cases respectively the often sufficient replaceability by pulse wide modulated (PWM) DO.

5. CONCLUSION

Perhaps biased by our industrial and standardization past we felt some of the qualities that come without saying with industrial automation systems and PLCs should be found too, when using other products or approaches for small process control applications in the widest sense. Home automation is just one field where one can fiend those intelligent products being connected to real world sensors and actuators – and "real" electrical power.
To make this quality transfer seizable we compiled it in a specification for specific small automation modules. We presented a proof of concept realization, thus demonstrating the feasibility of bringing the both worlds – tiny controller boards and open source with automation standards – together. In retrospect the software effort mainly to have some algorithmic problems successfully solved was underestimated.
Implemented use cases are a tiny automation of four rolling shutters now enabling remote control and programming of the times to go up and down by home PC. A control of two hot-

water heat pumps in a home's district heating transfer unit is the typical proxy installation. The automation/control unit sold with this central heating has no specified interfaces and could by no other means be persuaded to do what the customer wanted: just re-heating the tank water without hours of (cold) delay when six people had their showers and baths.

Another one is a – truly simple – process I/O distribution for a server based alarm system (burglar protection). It just collects data from sensors (movement, doors, windows) and controls actuators (lamps, flash lights, horns/sirens). All peripheral equipment is approved by the society of property insurance companies. This equipment approved therefore uses 12 V supply as LV throughout, and not the 24 V industry standard (for reasons unknown). The communication to the server is by Modbus protocol. It was a direct replacement of a previous PLC solution that suffered from regular communication outages about twice a year, requiring power off to recover. Additionally the LV flexibility saved all the 24 V to 12 V adaption.

The modules used are now in uninterrupted service over more than a year. The basic idea and approach seems quite feasible.

REFERENCES

The references are ordered by topics: [1] is the base idea for the efficient implementation of a the non-preemtive run-time.

[2 - 14] describe the implementation of TCP/IP and of the protocols realized thereupon. [15 & 16] provide the public specification of the SMC interface.

[17 - 20] describe the industrial automation/PLC standards adopted as requirements. [21 - 26] describe the hardware realization and its base as do [27 - 29] for the software.

[1] Adam Dunkels, O. Schmidt, Th. Voigt, Muneeb Ali, “Protothreads: Simplifying Event-Driven Programming of Memory-Constrained Embedded Systems”, Proceedings of the 4th international conference on Embedded networked sensor systems, 2006

[2] Adam Dunkels, “uIP - A Free Small TCP/IP Stack” November 25, 2001

[3] R. Braden, D. Borman, C. Partridge, “Computing the internet checksum” RFC 1071, Internet Engineering Task Force, September 1988J. Clerk, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68-73.

[4] J. Postel, J. Reynolds, “Telnet Line mode option,” RFC 1116, Internet Engineering Task Force, Network Working Group, August 1983

[5] D. L. Mills, “Internet Time Synchronization: the Network Time Protocol,” RFC 1129, Internet Engineering Task Force, Network Working Group, August 1983

[6] Judah Levine, “Time synchronization over the Internet using an Adaptive Frequency-Locked Loop,” IEEE Trans. Ultrason., Ferroelectr., Freq. Contr., vol. 46, no. 4, pp.888 -896 1999

[7] Katsuhisa Sato, Kazuyoshi Asari, “Characteristics of Time Synchronization Response of NTP Clients on MS Windows OS and Linux OS,” 38th Annual Precise Time and Time Interval (PTTI) Meeting

[8] Julien Ridoux, Darryl Veitch, “Principles of Robust Timing over the Internet,” acmQueue 1542-7730, 2004

[9] J. Reynolds, “BOOTP Vendor Information Extensions,” RFC 1497, Internet Engineering Task Force, Network Working Group, October 1989

[10] S. Alexander, “DHCP Options and BOOTP Vendor Information Extensions,” RFC 2132, Internet Engineering Task Force, Network Working Group, March 1997

[11] IEEE Standards Organisation, "Guidelines for use of the 24-bit Organizationally Unique Identifiers (OUI)," IEEE-SA. 2011

[12] Modbus-IDA, “Modbus Application Protocol Specification,” V1.1b December 2006

[13] Modbus-IDA, “Modbus Messaging on TCP/IP Implementation Guide,” V1.0b October 2006

[14] Wago, “Modbuskommunikation zwischen WAGO Ethernet Kopplern und Controllern,” A300003, German, Version 1.2.1

[15] SD Group, “SD Specifications, Part 1, Physical Layer, Simplified Specification,” Version 3.01, May 2010

[16] SD Group, “SD Specifications, Part A2, SD Host Controller, Simplified Specification,” Version 2.00, February 2008

[17] Siemens, "SIMATIC, Automation System S7-400 CPU Specifications," Reference Manual 2002 (6ES7398-8AA03-8BA0

[18] Siemens, "SIMATIC, Automation System S7-300 Baugruppendaten," Gerätehandbuch 2011 (6ES7398-8FA10-8AA0)

[19] Siemens, "SIMATIC, S7-300 Automation System Module Data," Manual 2011 (6ES7398-8FA10-8BA0)

[20] IEEE Power Engineering Society, "IEEE Standard for SCADA and Automation Systems," IEEE Std C37.1™-2007, May 2008

[21] NXP Semiconductors, “I2C-bus specification and user manual,” UM10204, Rev. 3 July 2007

[22] Atmel Corporation, “8-bit Microcontroller with 128K Bytes In-System Programmable Flash – Atmega1284P,” Summary 2009

[23] [23] Atmel Corporation, “8-bit AVR ® Instruction Set,” Overview 2010

[24] Atmel Corporation, “8-bit AVR ® 8-bit AVR Microcontrollers – EMC Design Considerations,” Application Note AVR040, 2006

[25] Atmel Corporation, “8-bit AVR ® 8-bit AVR Microcontrollers – Hardware Design Considerations,” Application Note AVR042, 2010

[26] Albrecht Weinert, “weAut_01 automation controller,” User manual (German), weinert-automation.de, 2012

[27] Albrecht Weinert, “weAut_01 / weAutSys,” manual, software documentation, weinert-automation.de, 2013

[28] William von Hagen, "The definite guide to GCC," second edition, apress 2006

[29] Sandra Loosemore, Richard M. Stallman, Roland McGrath, Andrew Oram, Ulrich Drepper, "The GNU C Library Reference Manual," Edition 0.12, 2007

Rolf Biesenbach received the Dipl.-Ing. degree from Wuppertal University, Wuppertal, Germany, in 1989, and the Dr.-Ing from RWTH Aachen University in 1995. From 1991 to 1994 he worked at the Forschungszentrum Jülich, Germany, and the Argonne National Laboratory, Lemont, IL, United States of America. In 1998 he joined the University of Applied Sciences Amberg-Weiden. Since 2003 he is Professor for Control Theory and Robotics at the Bochum University of Applied Sciences, Bochum, Germany. At present, Biesenbach is Director of the Mechatronics Centre North-Rhine Westphalia, and Dean of the Faculty of Electrical Engineering and Computer Science. He is member of the German engineering association VDI and President of the Mechatronics Association Germany, Deutsche Gesellschaft für Mechatronik.

rolf.biesenbach@hs-bochum.de, albrecht@a-weinert.de

Albrecht Weinert received the Dipl.-Phys. degree from the Technische Universität Clausthal, Germany, in 1977 and the Dr.-Ing. in 1983. From 1984 to 1997 he worked with System-technische Entwicklung of Siemens AG in Karlsruhe. His work areas were redundancy, fault tolerance and safety in process control systems and architectural concepts and project responsibility for high demand systems within TELEPERM M and SIMATIC S7. Since 1997 Weinert is Professor for Computer science at the Bochum University of Applied Sciences, Bochum, Germany, teaching Java, object orientation and computer architecture. In 2010 he put his extra-university development, services and consulting work under the label weinert – automation. Besides being active in several working groups he is member of the German engineering association VDI, PLCopen and of the Mechatronics Association Germany, Deutsche Gesellschaft für Mechatronik, providing some technical consulting and services to the latter.