IMTC 2003 – Instrumentation and Measurement
Technology Conference
Vail, CO, USA, 20-22 May 2003

SNMP-based Approach to Scalable Smart Transducer Networks

Balázs Scherer, Csaba Tóth, Tamás Kovácsházy, Balázs Vargha
Embedded Information Technology Research Group
Hungarian Academy of Sciences - Budapest University of Technology and Economics
Budapest, Hungary, H-1521, pf. 91.
{scherer, toth, khazy, vargha}@mit.bme.hu

***Abstract* – Nowadays on the embedded systems market there is a remarkable shift from the industry field buses to the Ethernet and internet (TCP/IP) communication technologies. The work presented in this paper aims to introduce an approach to develop TCP/IP based smart sensor and actuator networks, which are scalable, both in physical size and in the number of sensors and actuators. This approach uses the SNMP (Simple Network Management Protocol) internet technology and the IEEE 1451 Standard for Smart Transducer Interface for Sensors and Actuators.**

I. INTRODUCTION

During the last five years a remarkable spreading of the high level communication technologies, principally the Ethernet/internet, was noticeable in the embedded system market [1]. As the result, most of the leading embedded system manufacturers have started offering solutions and application notes to connect their devices into TCP/IP protocol based computer networks [2], so nowadays it is generally not too complicated to connect even an 8 bit microcontroller to a high speed Ethernet network using the TCP/IP protocol stack.

Although the hardware and software conditions of this high level communication of embedded systems are fully given, and it is relatively easy to implement such devices, it is still far from trivial to develop a complex system using these techniques. The main problem of complex system design is the not standardized way of accessing these individual devices. Currently, most of the devices are communicating with standard TCP/IP protocols, but are using unique services.

The goal of this paper is to briefly review the applicable internet protocols and other system architectures, to describe a solution for developing network capable smart sensors and actuators, with good system integration ability.

II. IMPLEMETATION ALTERNATIVES

Our goal was to create a TCP/IP based smart transducer system; therefore, as a first step we have done a search for corresponding applications have already made. This chapter shortly reviews the alternatives found as the result of our survey. Below we described the TCP/IP application-layer protocols used in embedded systems, and a reference model for networked smart transducers the IEEE 1451 Standard for Smart Transducer Interface for Sensors and Actuators [3] is also introduced.

A. Web based implementation alternative

The web based solution is currently the most widespread among the TCP/IP networked embedded devices because its user-friendly HTML pages. This solution can be used in many simple applications, where direct human control is required, but has a great disadvantage, where data logging and automatic control is needed.

It is relatively easy to implement a Web based solution either in 8 bit devices due its small resource requirements and the many off-the-shelf components [4], [5], [6], [7], [8].

B. FTP based implementation alternative

The File Transfer Protocol is generally advantageous for data logging applications. In this case the embedded system operates as an FTP server collecting the measurement data into remotely accessible files. The advantage of this solution is the simple way of accessing measurement data, and the disadvantage is its lack of configuration possibilities. It is extremely hard to change measurement parameters in an FTP based system as for example sampling frequency.

Implementing an FTP based solution is relatively easy, because there are off-the-shelf components available [4], [5], [6], [7], [8] , and cheep, because it could be hosted onto an 8 bit system.

C. SNMP based implementation alternative

The SNMP (Simple Network Management Protocol) [9] is an UDP (User Datagram Protocol) based protocol for observing and modifying the configuration and statistic parameters of remote nodes (switches, routers, PCs or embedded systems). The architecture of an SNMP managed system is distributed into two parts: SNMP mangers, and SNMP agents.

The SNMP agents are programs running on the systems intended for remote observation and configuration. Each agent is supervising a MIB (Management Information Base). The MIB-s are variable structures containing the configuration, statistic parameters and other information of the node. The SNMP managers have the opportunity to

monitor or modify the remote system's operation and attributes by observing or altering the values of the variables stored in the Management Information Bases. This is a centralized architecture, because the agent part is mainly passive. An agent is initiating communication by sending trap or notifications to the pre specified manager nodes only in the case of alarm situations.

When developing an SNMP based smart sensor or actuator, the designer has to implement an SNMP agent onto the target embedded device. After the SNMP agent is implemented the designer has to define MIB variables for representing the smart sensor or actuator functionality (for example one read only variable for the measured temperature data and another read-write one for a relay contact).

Its configuration and monitoring oriented architecture, resource frugal implementation possibilities and many off-the-shelf components [4], [6], [7], [8] makes the SNMP well useable for embedded systems.

D. IEEE 1451 Standard for Smart Transducer Interface for Sensors and Actuators

The IEEE 1451 standard aims to define a universal smart transducer model, which neither depends on the communication infrastructure nor on the type of the sensor or actuator. The IEEE Std 1451 divides the smart transducers into three segments: *1)* sensors and actuators providing contact to the physical world, complemented by an interface to the second segment; *2)* application core, which handles and manages the first block, processes and/or stores its data by sending commands to it; *3)* network interface for the whole device.

The first block is called STIM (Smart Transducer Interface Module) and is defined by [10]. The last two segments are merged into a single block named NCAP (Network Capable Application Processor), which is described in [11] (See Figure 1).

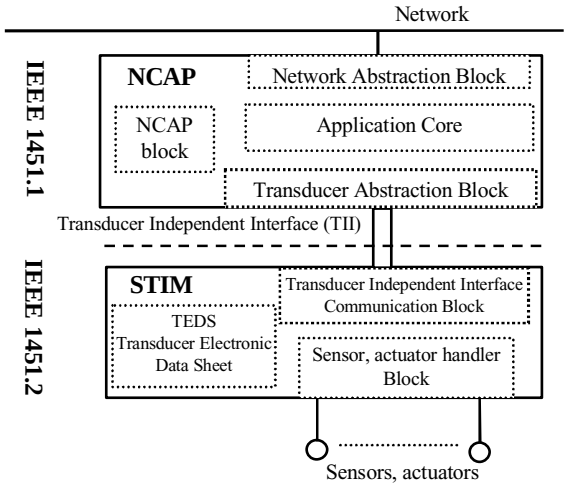

Figure 1: Architecture of the IEEE1451 standard

The NCAP, which realizes the network communication and control functions, communicates with the STIM using a standard digital interface, called Transducer Independent Interface (TII). The STIM is a simple device with limited complexity and functionality. Its primary goal is to integrate the sensors and actuators in one block, offering a common interface for the NCAP to handle them. The most important part of the STIM is the Transducer Electronic Data Sheet (TEDS). This self-describing data structure makes it possible to identify the STIM and the transducers, with their very detailed properties. Among other things, the TEDS contains identification, timing, data structure, calibration, and human readable information in a common, standardized form. These abilities enable the usage of the same STIM device across many networks and applications (assumed there is an NCAP for each network).

There are off-the-shelf components of IEEE 1451.2 based STIM-s on the market [12], [13], and an individual implementation of such a device is also relatively easy because there are useful application notes available from several manufacturers [14], [15]. Unfortunately there is currently no off-the-shelf NCAP on the market (only Hewlett Packard had an NCAP series called BFOOT, but its manufacturing has been discontinued). Experiences show that the implementation of the IEEE 1451.1 based NCAP module as a locally embedded system is not a practical solution because: *1)* It requires a large amount of human efforts. *2)* The full implementation has relatively large run-time resource requirements, which can be fulfilled only by high-end embedded processors (16 or 32 bit, e.g., ARM or PowerPC) and memory in the megabyte range, which increases the price of the unit. *3)* The object model of the NCAP is relatively hard to implement using the current embedded development systems, because these latter ones rather support procedural programming, whereas the NCAP model is object oriented.

III. PROPOSED APPROACH

Feasibility study of the above alternatives showed the SNMP and the IEEE 1451 standard based approach to be the most advantageous.

The SNMP based approach can be implemented easily, and the protocol is well known among the network designers and also supported by many companies, but a standardized MIB to describe smart transducers should be needed. Such a MIB is not available currently, and creating individual MIB-s for each transducer would not support system integration and could lead to a chaotic state. The IEEE 1451 standard has unquestionable good ideas (for example the separation of functions, and the implementation of TEDS) and should be used as an important guide during the development of smart transducers, but unfortunately, due to its complexity there is currently hardly any working implementation of it.

Because none of the two solutions were unambiguously the best we have tried to combine their benefits. We found it

possible to create an SNMP MIB called Smart Transducer MIB, which contains the information that was originally stored in the TEDS. Attaching the above MIB to an SNMP agent could result a network reachable transducer independent interface. Taking the opportunity of this network reachable transducer independent interface the control of different devices could be formalized. The formalized control of different networked devices could provide computer aided system designing technologies resulting fast and simple development of smart transducer networks. The above described solution is generally not else than an SNMP based STIM.

This SNMP STIM could be implemented in two ways:

1) The SNMP STIM is implemented without the IEEE 1451 standard proposed separation of network dependent and independent functions. This solution is very resource efficient and therefore could be implemented in cheep 8 bit devices.

2) The SNMP STIM (the SNMP agent with its MIB; the network capable part) is separated form the network independent functions as suggested by the IEEE 1451 standard providing a TII interface, for IEEE 1451.2 based STIM-s. In this case the SNMP STIM could be a pseudo NCAP. This solution keeps the hierarchic architecture of the IEEE 1451 standard with the main difference in the application core. In the IEEE 1451.1 standard the application core is enclosed in the NCAP, taking the control near to the process, thus resulting distributed system architecture. Due to this process near control the NCAP becomes very complex with high resource requirements. In the SNMP STIM case the application core is not enclosed in the pseudo NCAP, it is hosted into a centralized management station, which leads to centralized system, but the embedded devices are become simple and cheep (See Figure 2).

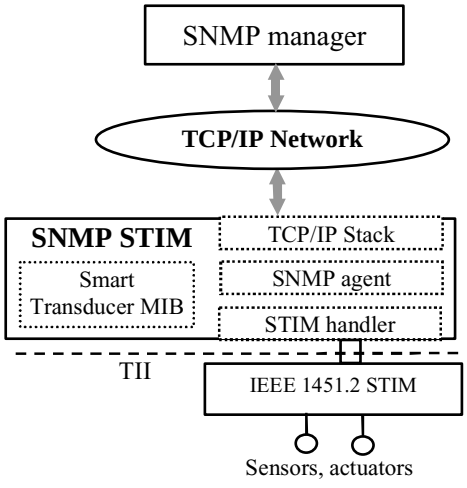

Figure 2: SNMP STIM with TII interface

Both implementation ways represents a system with centralized process control, but the most of the problems are solvable with this architecture. This architecture has also the possibilities of system level plug-and-play, because when connecting a new SNMP STIM into the network the centralized manager could be able to detect its abilities, and with minimal human interaction start to supervise it. Both systems, because their remote accessible interface provides functions very similar to the TII, also have migration ability to the IEEE 1451.1 based NCAP-s. The only thing has to be done to integrate the SNMP STIM to an IEEE 1451.1 NCAP is to implement an SNMP STIM handler transducer block similar for the IEEE 1451.2, IEEE 1451.3, IEEE 1451.4 transducer blocks.

IV. MIB CONTAINING TEDS INFORMATIONS

To create the SNMP STIM described above there is a need for a MIB containing TEDS and control information. This MIB is called the Smart Transducer MIB. The Smart Transducer MIB was created using the SNMP standard SMI (Structure of Management Information) syntax. Because SMI is platform independent, the result is useable in any SNMP agent software, assumed that the agents have appropriate tools to compile these standard MIB-s. The skeleton of the created MIB tree is showed in Figure 3 (the tree nodes usually variable groups not single variables).

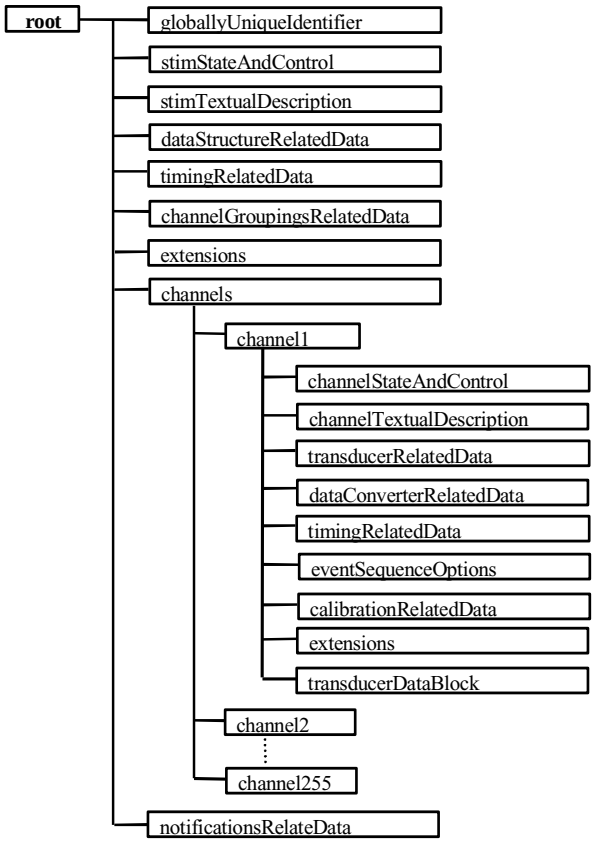

Figure 3: The skeleton of the Smart Transducer MIB

The Smart Transducer MIB contains the Meta, Meta Identification, Channel, Channel Identification, Calibration and Calibration Identification TEDS information, and extension abilities for manufacturers. The MIB furthermore contains the control and status information of the standard IEEE 1451.2 based STIM in the stimStateAndControl, and channelStateAndControl sub trees. The transducer data values are also included in the transducerDataBlock sub tree of the Smart Transducer MIB.

During the creation of this MIB our elementary conception was to transform the TEDS information directly to a MIB tree. This transformation was successful, but it was predictable that some TEDS fields, due to their general purpose design, might cause problems later in the application phase. For an example such problem is the handling of the Identification TEDS textual description fields. In this TEDS the textual descriptions character coding is not fixed it could be varied on a large scale. This variation is not supported by the public SNMP managers, because the SNMP has its own simple text representation type. The above features make the off-the-shelf SNMP managers useless or hard to use. So there is a need for a custom-made SNMP manager software.

The tree structure of the SNMP is static, which also cause some problems. In some applications the not implemented fields, instead of simply ignoring them, should be filled with constant parameters representing that the field is not used. For example, ignoring the description table of the stimTextualDescription sub tree (there is no Meta Identification TEDS information available) is not permitted.

Although the problems above are present, the Smart Transducer MIB is well useable, and has some benefits: Each individual variables of the Smart Transducer MIB is accessible. This was not available in the IEEE 1451.2 TEDS, where only the whole data sheet was accessible. In the Smart Transducer MIB it is possible to define each TEDS variable as read-only or read-write. This is also not available in the IEEE 1451.2 TEDS, where only the whole TEDS could be modified. These advantages of the SNMP are making the configuration and monitoring easier.

V. SNMP STIM

To test the applicability of our MIB we implemented an SNMP STIM that provides a TII interface for IEEE 1451.2 based STIM-s. This SNMP STIM reads the connected IEEE 1451.2 STIM's TEDS information and makes that reachable from the network. It is also possible to read or modify the status information, transducer data values, interrupt masks of the IEEE 1451.2 STIM, or to send control command to it across our pseudo NCAP.

The first phase of the implementation was to choose a development system to be the host of our SNMP STIM. The main guidelines of the development system selection were to ensure that we won't have resource problems during the test implementation and we will be able to use as many off-the-shelf components as possible.

Therefore we chose a complex development system containing an ARM7TDMI based 32bit microcontroller, the Cirrus Logic EP7312 [16], 16Mbyte of SRAM, 4Mbyte Flash, CS8900 10BaseT Ethernet controller, and enough general purpose IO pins to implement the TII interface.

An embedded operation system the RedHat's eCos was ported to this board to aid the concurrent programming development and network communication. The eCos operating system is configurable to include the TCP/IP stack, and an SNMP agent [17], which is the heart of the SNMP STIM is also portable to it. The SNMP agent was used is the eCos port of the well known open source UCD-SNMP 4.1.2 version [18] (latest versions are known as NET-SNMP).

After preparing the development system the remaining tasks were to compile the SMI syntatic textual MIB to C files and to attach these files to the SNMP agent. An API was also implemented to handle the TII communication.

The SNMP STIM was made by the above way fits our requirement. It is able to detect the connection of STIM-s. After the connection is detected the SNMP STIM reads all the TEDS information from the IEEE 1451.2 based STIM and stores it. Using the TEDS information specified values the SNMP STIM sets the TII communication frequency and the abort times. The stored information and other variables to control the STIM are mapped to the smart transducer MIB and therefore become remotely accessible. The stored information is refreshed by the effect of reseting remotely or locally the SNMP STIM or the IEEE 1451.2 STIM. This stored information is also refreshed, if the IEEE 1451.2 STIM is removed or new STIM is connected. So this SNMP STIM is absolutely transducer independent and, when somebody would like to connect a new IEEE 1451.2 STIM to a network then nothing else need to be done, just connecting this new device to an SNMP STIM. After the new device is connected to the network only the implemention of the apropriate application functions is needed in the centralised manager to make the device active.

Measuring the test system showed that the response times of the SNMP STIM are in 2-5ms range, when TII communication is not required. This seems to be satisfactory.

VI. STIM MODULES

Two IEEE 1451.2 based STIM-s were implemented to test our SNMP STIM. Both STIM were realized on a low cost 8 bit microcontroller the PIC18F452 manufactured by Microchip Corporation. The STIM programs were written in C using the application note of the Microchip [14]. The TEDS were added to these STIM-s as our TEDS editor generated C header files.

The first STIM is a test panel made for directly this purpose. This STIM has 7 channels. It contains 4 LED-s, representing one actuator channel, and 3 relays as another three actuator. There are also 3 sensor channels with light, humidity, temperature sensors.

The second STIM is a standard Microchip manufactured demonstration board the PICDEM 2 PLUS board [19]. This development system was complemented to support the TII communication. This STIM contains two sensor channels: one for temperature sensing, and another one for Voltage measurement. There are also several actuator channels on it controlling an LCD, a buzzer, and 4 LED-s.

As the description above showed, these STIM-s have very different sensing and actuating functions. Therefore they are well demonstrating that our approach is useable on wide scale of measurement and control problems.

Our experiences showed that using the IEEE 1451.2 standard defined STIM is an efficient way of developing network independent smart transducers. We also found that, the core of the STIM is function independent. Therefore only the replacement of the function dependent part of the old STIM program is required when developing a new one, which makes the development fast. The implementation of the second STIM, which was a demonstration board, and therefore no hardware design was needed, was taken less then 8 hours with TEDS editing, and testing.

VII. TEDS EDITOR

Our simply-to-use TEDS Editor is a Windows hosted dialog based program implemented in Visual C to reduce the TEDS creating procedure. The editor is able to create and modify every TEDS specified in the IEEE1451.2 standard, and the result is available in multiple formats: a binary one for EPROM downloading, and a C header file form for including it into a STIM program.

VIII. MANAGING THE SNMP STIM

For verifying our SNMP STIM and the connected IEEE 1451.2 STIM-s managers was constructed without plug-and-play functions. Simplifying the testing phase individual manager was construed for each STIM. These managers are simple MATLAB programs using DOS shell SNMP commands [18]. The managers are displaying the last 100 data values measured by the sensors and providing a GUI for controlling the actuators.

The development of these managers showed that the SNMP STIM is well controllable, but a future revision of the Smart Transducer MIB is needed to make this control easier. This revision we think should be more SNMP manager friendly and therefore less bounded to the IEEE 1451.2 TEDS.

IX. CONCLUSION

Our SNMP-based pseudo NCAP provides a transducer independent network accessible interface, which is useable to formalize the control of devices with different functions. Therefore our approach seems to be an economical and useful solution for building scalable sensor/actuator networks.

As a further development a simple version of the SNMP STIM could be implemented without hardware separation of the network dependent and independent functions (software separation is hardly recommended to ensure a fast and structured development of different devices). This low-end variant could be hosted to an 8 bit microcontroller, and therefore would be very cheep. The cost of the system core, without sensors and actuators, could be less than $40.

Other suggested future development task is the revision of the created smart transducer MIB to make it friendlier for users.

Development of an easy to use manager environment to control SNMP STIM-s is also required. The base of this environment could be for example LabView or probably Matlab Simulink mappings of the SNMP STIM, but other solutions are also conceivable.

REFERENCES

[1] Joint European Network on Embedded Internet Technologies. http://www.eurojenet.com

[2] W. Richard Stevens, *TCP/IP Illustrated*. Addision-Wesley Publishing Company, 1994

[3] Kang Lee, *Sensor Networking and Interface Standardization* IEEE Instrumentation and Measurement Technology Conference Budapest, Hungary, May 21-23 2001

[4] CMX Systems Inc*: CMX-TCPIP™ Package*: http://www.cmx.com

[5] Microchip: *PICDEM.net Internet/Ethernet Demonstration Board*: http://www.microchip.com

[6] KADAK Products Ltd: *KwikNet TCP/IP Stack for AMX RTOS*: http://www.kadak.com/html/kdkp1030.htm

[7] Lantronix: *USNET Embedded TCP/IP Protocol Suite*: http://www.lantronix.com/products/esw/usnet/index.html

[8] QNX Software Systems Ltd: TCP/IP for Qnx: http://www.qnx.com/products/networking/tcpip.html

[9] William Stallings: *SNMP, SNMPv2, SNMPv3, and RMON1 and 2 (3rd Edition)*

[10] IEEE Std 1451.2-1997. *IEEE Standard for a Smart Transducer Interface for Sensors and Actuators. Transducer to Microprocessor Communication Protocols and Transducer Electronic Data Sheet (TEDS) Formats.* Sponsor TC-9, Committee on Sensor Technology of the IEEE Instrumentation and Measurement Society. Approved 16 September 1997. IEEE Standards Board

[11] IEEE Std 1451.1-1999. *IEEE Standard for a Smart Transducer Interface for Sensors and Actuators. Network Capable Application Processor (NCAP) Information Model.* Sponsor TC-9 Committee on Sensor Technology of the IEEE Instrumentation and Measurement Society. Approved 26 June 1999 IEEE-SA Standards Board

[12] Dipper Techno Chemistry Institute: http://www.fullsense.com

[13] Telemonitor Inc: http://www.telemonitor.com

[14] Microchip AN214, *The PICmicro MCU as an IEEE 1451.2 Compatible Smart Transducer:* http://www.microchip.com/1010/suppdoc/appnote/all/an214/index.htm

[15] MicroConverter Technical Note - uC003: *The ADuC812 as an IEEE 1451.2 STIM* http://www.analog.com/

[16] *EP7xxx User's Manual* DS508UM3 Cirrus Logic 2001. http://www.cirrus.com/

[17] Redhat eCos and TCP/IP stack for eCos: http://sources.redhat.com/ecos/

[18] Wes Hardaker, University of California at Davis: *UCD-SNMP*: http://www.net-snmp.com

[19] Microchip *PICDEM 2 Plus Demonstration Board* http://www.microchip.com/1010/pline/tools/picmicro/icds/icd2/cupola/pcdmplus/index.htm