Intelligent IO-Terminals for complex Communication Scenarios in Control Applications

Volker Schuermann
Department of Electrical Engineering and Computer Science, Bochum University of Applied Sciences
Bochum, 44801, Germany

Rafael Leidinger
Department of Electrical Engineering and Computer Science, Bochum University of Applied Sciences
Bochum, 44801, Germany

and

Joerg F. Wollert
Department of Electrical Engineering and Computer Science, Bochum University of Applied Sciences
Bochum, 44801, Germany

ABSTRACT

In the automation technology one observes at present an increasing spreading of subbuses for the communication with sensors and actors. The manufacturers use here frequently proprietary application layer protocols, which makes the reusability of PLC (Programmable Logic Controller) software more difficult, because this contains frequently application layer protocol elements. This Paper shows a way to release the applications from all protocol-specific elements, so that they can be reused more simply. The protocol information is then shifted by configuration into an intelligent subbus master, which makes an application appropriate mapping in the process image of the PLC.

Keywords: CAN, Automation Technology, Building Automation, Protocol Integration, Subbus

1. Introduction

We are at present in the middle of a trend to a more strongly and important networking of electronic devices. That is not only valid for PCs, which are connected by the internet, but also for embedded devices. For example in the building automation or in the vehicle. Usually TCP/IP is not suitable as communication basis here due to its high requirements to the clients. Rather one needs a slim and efficient communication technology. The CAN bus (Controller Area network) [1] [2],which is originally a communication technology from the motor vehicle, is such a technology, which has an increasing popularity over the last years. Also in other fields of application, like for example in the building automation.

However results the following problem, with the increasing spreading of CAN bus implementations the number of assigned protocols on application level is rising to, because the CAN Specification makes no defaults and also CANopen [3] [4] is not yet established within all fields of application. That has for the application developer as a consequence that he must partly handle several protocol implementations for the same application. Especially for PLC systems, those are rather designed for handling simple digital and analog IOs, this means a large expenditure.

CAN bus is outstanding suitable for the demonstration of the advantages of intelligent IO terminals, which have the goal to release the application developer from the additional expenditure for handling different application layer protocols within one program.

Thus this contribution points a way out for the integration of subbuses in PLC systems, without the frequently expenditure exerted by the implementation of multiple application protocols. Which particularly the reusability of PLC applications increases.

2. Integration of Subbuses in PLC Systems

The tendency towards highly flexible distributed control systems within the industrial automation is clearly reflected in the modular form factors of recent I/O assemblies. The typical data communication in industrial control systems takes place in a synchronous manner. The process data of each subscriber and superordinate control systems is periodically exchanged.

The integration of the process data of sensors and actors may be realized via the Fieldbus interface. However the distributed structure is primarily realized by the use of Bus Terminal Units. These consist of a Fieldbus Coupler/Controller, the latter with PLC functionality for preprocessing the data, and a modular expandable number of IO-Terminals. Via the Fieldbus interface of the Fieldbus Coupler/Controller the bundled process data of sensors and actors may be distributed in the control system.

By the use of an internal bus (K-bus) the communication between the Fieldbuscoupler/controller and the IO-Terminals takes place. The bus system is designed as a serial ring shift register. In order to update the process images (PI), the IO-data is exchanged with each ring cycle. The typical update time of 1000 digital IOs is below 20 ms. The amount of IO-data of each IO-Terminal varies with its complexity. Besides simple digital IO-Terminals, which only transfer a few bits of data, there are complex Terminals, which act as gateways to subordinate systems. Bus Terminal Units for all common Fieldbus standards are available.

The integration of subbuses in PLC systems is already for a long time an active research topic at the Bochum University of Applied Sciences [5] [6] [7]. In the past mainly wireless technologies as for example Bluetooth [8] or IEEE 802.15.4/ZigBee [9] were successfully integrated. A goal was the creation of a uniform and for the automation technician admitted environment for the integration of wireless components in control networks. Aspects such as real time ability and security should be considered thereby in the sufficient measure.

In Figure 1 a programmable controller with a wireless network as subbus system is illustrated. Both systems run in their own cycles, in which they update their data completely asynchronously to each other. In the subbus system an own process image is held by the subbus master and each data exchange with the programmable logic controller is constantly served immediately from this process image. With this data exchange a synchronously full duplex exchange takes place between both systems. Subsequently, the PLC takes over the data of the subbus system and integrates them into their own process image.

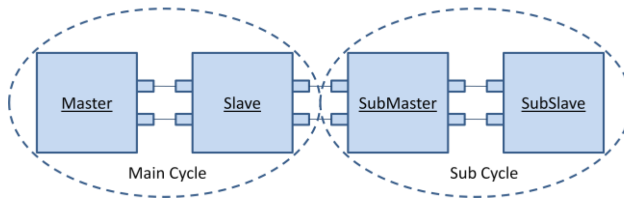

Figure 1: Integration of Subbuses in PLC systems

The binding of the subbus is been made by a particularly developed IO Terminal for the PLC system. This IO Terminal has three substantial tasks with the classical integration, which has the implementation of a transparent communication channel as the main goal:

1. The IO Terminal is the master of the attached subbus and is so responsible for the configuration and operation of the subbus.
2. Like all other IO Terminals, it is also a slave at the internal bus.
3. It is also responsible for the mapping of the data from the subbus to the process image of the PLC and vice versa.

These tasks reflect themselves also in the physical structure of the IO Terminal. It consists of two, to a large extent independent, components with in each case own microcontroller and own board.

The internal bus coupler establishes the physical connection to the internal bus of the PLC. In addition it embeds the data received from the subbus into the process image of the PLC and/or reads data from the process image and passes these on to the subbus. This part of the IO Terminal is independent of the communication technology used as subbus and can be usually reused from there without changes.

The subbus is now tied up physically to the programmable logic controller and from the view of the PLC a uniformly abstracted interface over the process image for the different communications network works is available.

3. Intelligent subbus master

So far with the development of the subbus masters at the implementation of a transparent channel was aimed with priority. Higher protocols of the application layer were not more considered but only tunneled through. This had the consequence that the handling of the different protocols had to take place at the PLC. That is possible and the usual IEC 61131-3 [10] programming languages as for example structured text permits it also, however that is not the fundamental work behavior of a PLC. Here the focus lies frequently on interconnecting digital and analog inputs and outputs, which are represented in the simplest case only over one bit in the process image.

A multiplicity of the messages received or dispatched over the subbus can be represented finally also on a such simple action. This representation had to be realized however so far by the application developer in an IEC 61131-3 programming language.

The here introduced "intelligent" way shifts this expenditure from application programm into the IO Terminal. So that these do not only handle one protocol statically, it is necessary to influence the mapping of messages from the subbus to the process image by configuration. The concrete case goes one more step ahead and permits even the linkage of individual bits of the subbus messages with individual bits in the process image. This was realized exemplarily for a CAN subbus master.

The mapping of the individual bits takes place with the help of a XML configuration file. In this file for example baudrate or operating mode can be defined, a transparent communication mode will be supported further. The second part includes configuration definitions, which are met apart from the usual configuration attitudes. It is defined how each bit from each CAN message has to be mapped on an appropriate bit in the process image of the PLC.

```
<?xml version="1.0" encoding="UTF-
8" ?>
<can>
<!-- Miscellaneous configuration
options.-->
  <misc>
    <mode>1</mode>
    <baudrate>500000</baudrate>
  </misc>
   <!-- Process Image mapping.-->
  <modus_1_config>
    <processimage byte="0" bit="1">
      <can_id>2</can_id>
      <byte_nr>1</byte_nr>
     <bit_nr>1</bit_nr>
    </processimage>
    <processimage byte="2" bit="7">
     <can_id>5</can_id>
     <byte_nr>0</byte_nr>
     <bit_nr>7</bit_nr>
    </processimage>
  </modus_1_config>
</can>
```

A possible structure of the XML file is represented in the above code example. The structure is always in principle alike. Beginning with the element <can> in the following element <misc> the general attitudes are fixed, which are always the same, like baudrate and operation mode. The second part of the XML file always differs, depending on which mode was selected. In this example the mode 1 was selected, which corresponds to the bit configuration mode. In the elements <processimage byte="0" bit="1"> following on it is to be recognized that the individual bits of the process image are configured here. As example here the bit 1 of the byte 0 is configured and is assigned to the values CAN_ID = 2, byte = 1 and bit = 1. This means for practice that if a CAN message with the ID 2 will be received and within this message the first bit of the first byte is set, exactly then the bit 1 of the byte 0 in the process image is also set.

4. CAN Configuration Manager

In order to not have to program again the microcontroller for each new attitude of the baudrate or the operating mode, the CAN Configuration Manager manages these changes and transfers them to the IO Terminal, see Figure 2. Beside the selection of the operating mode and the baudrate, the bit configuration offers also the possibility, to map the individual bits of the process image of the CAN IO Terminal on a certain bit of a certain CAN message, whereby in the current version for the input and output process image the same configuration is used.

Apart from the primary tasks of CAN Configuration Manager still another few secondary tasks have to be considered. As the first a XML file is to be produced from the entire configuration. This serves for the fact that one can again view and edit the configuration settings on the host computer to a later time, in order to e.g. look for errors. Further the use of XML offers the possibility to validate the configuration of on the basis a XML schema and thus to guarantee that the file keeps the fixed form and contains no errors. The processing of the configuration on sides of the micro controller is to be kept as simple as possible. Due to that fact in a last step from the validated XML file a proprietary text file with the assistance of XSLT is provided. XSLT stands for XSL transformation and is a language for the transformation of XML documents. Since the text file was produced by transformation of a validated XML file, it is guaranteed that it keeps the desired form and thus no errors on the microcontroller appear.

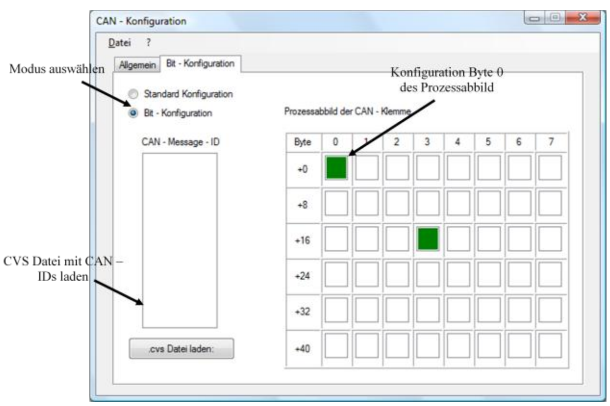

Figure 2: CAN Configuration Manager

5. Use Cases

For a so configurable CAN IO Terminal a set of use cases for example in the latch technology or for Rapid-Prototyping for vehicles result.

Latch Technology:
While in the motorring industry an integrated and radio-controlled latch system is not to be excluded any longer, then is key and lock still the most used aid in the building protection field. Apart from this latch technology the use of key cards and transponders within many ranges like for example in hotels constantly increases and completely interconnected latch systems are also no more rarity. With these devices it concerns frequently with proprietary solutions, which are not standardized. However lately clear trends are to be observed toward CAN bus with some manufacturers. The superordinate protocols remain however proprietary, which means for the system integrator in the building automation that for each attached latch system the PLC software must be modified. With the help of an intelligent IO Terminal he reuses his application "latch system integration" and adapts only the configuration settings with the protocol relevant information.

Rapid Prototyping for Automotive Applications:
Rapid-Prototyping of control units within the automotive field of application is frequently done via the employment of simple microcontroller boards. These are proprietary and mostly bound to a concrete vehicle configuration. Besides the connection of IO and bus systems turns out as difficult. Alternatively one can use also expensive universal control units. A more economical alternative are from the automatic control engineering admitted Programmable Logic Controller (PLC). With the help of a CAN IO Terminal a problem-free integration in the vehicle is possible and by the configurability regarding higher protocols vehicle and/or manufacturer-specific characteristics can be neglected during the application development. This is particularly interesting for suppliers, which work for several vehicle manufacturers.

6. Conclusion

The CAN subbus master described in this paper serves as example for the advantages, which arise as a result of the employment of intelligent IO Terminals in the automation technology. Particularly the development of application software on the PLC is simplified substantial, since protocol-specific characteristics are treated to the IO Terminal and are no more part of the application.

For this the CAN IO Terminal makes the bit configuration mode available, which permits a development of the PLC application independent of the used application protocol. Subsequently, with the help of CAN Configuration Manager each individual bit of the process image is configured in such a way, as it is needed for the respective application without having to change something in the source code of the application.

Such a possibility for the application development did not yet exist up to this time and thus represents the CAN IO Terminal in connection with the CAN Configuration Manager an extremely flexible and applicable CAN solution. With this solution it is possible to reduce development costs and develop new prototypes within shortest time. It is not any longer necessary to develop new hardware prototypes and the range of the software development costs are reduced likewise to a minimum. That saves time and money.

References

[1] ISO 11898, Beuth Verlag, 2009

[2] Robert Bosch GmbH, CAN-Spezifikation Version 2.0, 1991

[3] CAN in Automation, www.can-cia.org, 2010

[4] Lawrenz, W., CAN Controller Area Network, Hüthig Verlag, 1999.

[5] Wollert, J. F., Research Project: DANA-Dynamic Ad-hoc Netzworks in Automation (in german), 2008

[6] Buda A., Schuermann V., Wollert J.F.: A Middleware Approach for IndustrialWireless Automation Networks,IASTED – Wireless and Optical Communication 2008, Quebec City, Canada,26-28.05.2008

[7] Schuermann V., Buda A., Wollert J.F.: XML-based Middleware Approach for Industrial Wireless Communication Systems, IEEE – Emerging Technologies and Factory Automation 2008, Hamburg, Germany, 15-18.09.2008

[8] IEEE 802.15.1 Specification, URL: www.ieee802.org/15/pub/TG1.html

[9] IEEE 802.15.4 Specfication, URL: http://www.ieee802.org/15/pub/TG4.html

[10] IEC 61131-3, VDE Verlag, 2003