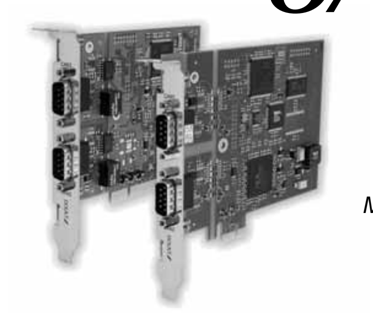

CANopen in X-ray systems

Martin Merkel and Christian Schlegel (Ixxat Automation)

Already in the early days of CAN, Philips Medical Systems noticed the advantages of CAN and decided to use this network protocol as communication network for interconnecting various components such as collimators, generators, and patient tables in their X-ray systems. To achieve a modular and open approach, a group within Philips Medical Systems, managed by Tom Suters, developed the first higher layer protocol for CAN, the CAN Message Specification (CMS), which was presented to the public in 1992. CMS provided the framework for the first official CAN based communication protocol specified by CAN in Automation, the CAN Application Layer (CAL). One of the reasons that may have lead to the limited market penetration was that CAL did not provide the functionality necessary for a straightforward description of generic device or application profiles.

As a consequence the CANopen application layer and communication profile was defined and published as official CAN in Automation standard in 1996. CANopen did not completely reinvent the wheel. Instead, it reused many elements and protocols of CAL and added missing but necessary features such as the object dictionary, which represents a simple way of describing and referencing application data. In fact, CANopen is now perceived as the enabling technology for the use of CAN in many different application areas.

Advantages of using CANopen in medical applications

Today, the necessity for using data communication systems in medical applications becomes more and more evident. Reasons for this are increasing requirements regarding:

- Interoperability, which is primarily required for the exchange of data between medical devices but also allows for the implementation of modular architectures of medical devices. At the same time interoperability enables central control of various medical devices, which is an important issue when looking for improved and more gentle respectively tolerable examination procedures.
- Improvement of autonomous operation of medical devices allowing for unsupervised examination procedures and improvement of patient safety.
- Cost reduction due to higher level of modularity in medical devices and faster execution of examination procedures.

Deploying Ethernet TCP/IP (transmission control protocol/internet protocol) or one of the emerging industrial respectively real-time Ethernet protocols as data communication system for control purposes like transmission of control data, commands and parameters is not adequate. Compared to CAN, the implementation of Ethernet requires more powerful and therefore more expensive CPUs (central processing unit) and thus more expensive hardware (this applies in particular to high-quality PHYs, transformers and connectors). A secure transmission of data without any loss of data is only possible with suitable protocols implementing acknowledgement mechanisms or with cyclic transmission of data. Also, for free or arbitrary topologies, wiring is typically less expensive for CAN installations compared to Ethernet networks where additional topology components such as switches or hubs may need to be deployed.

Integrating Ethernet technologies within medical systems is nonetheless suitable for the transmission of bulk data such as patient information data or diagnosis results, which may not be time-critical and is typically based on a physical peer-to-peer connection between sender and consumer.

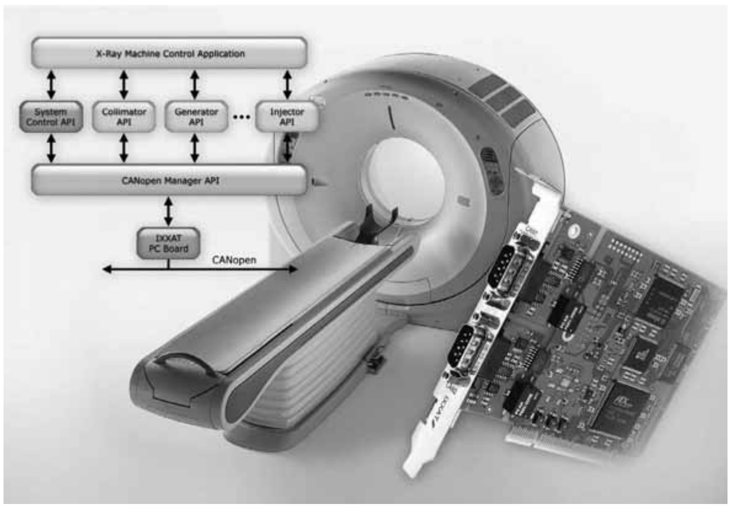

Fig. 2: CANopen manager software architecture on an X-ray control system

Medical devices impose very specific requirements to communication systems in particular with respect to the control of the devices. These requirements are: Very high reliability and robustness for data transmission, short latency times for the transmission of important and high priority data, short error recovery times, data transmission as broadcast (producer/consumer principle) but also client/server data transmission, low device and implementation costs and simple manageability. ▶

Analyzing the requirements of X-ray systems shows that CANopen is in fact perfectly well suited both as internal communication network used to interconnect all modules and functions inside the X-ray system as well as to connect external add-on devices to the X-ray system. Due to the nature of CAN, CANopen offers very high error reliability, short latency and error recovery times, a robust data transmission, various possibilities for the modularization of systems and networks, plug-and-play support and standardized system services. It allows building flexible network topologies and can be implemented at low costs. Furthermore, the technology of CAN and CANopen is already accepted by both the TÜV (Technical Monitoring Association) in Germany and the FDA (Food and Drug Administration) in the USA for the use in medical systems, as there are already a number of approved applications using this technology.

Within CAN in Automation, a special interest group (SIG) is working on the specification of device and application profiles for medical devices and applications (SIG medical devices) with focus on X-ray systems. An additional task force is working on a dedicated specification for the use of injectors as X-ray add-on devices.

Implementation of a CANopen NMT master interface

When implementing a CANopen NMT (network management) master interface in medical devices like X-ray systems, it is recommended to consider very early in the design phase an approach that later on will allow to easily set-up and maintain a modular CANopen network. The X-ray system itself already consists of a number of functional modules such as generator, collimator, stand, patient table, dose measurement system and optional add-on devices, which may be integrated within the network to enhance the overall system functionality. Additional external medical devices may be connected to the X-ray system and shall be controlled by the X-ray system depending on the requirements of an examination. Typical examples for external devices are contrast media injectors or electrocardiograms.

The CANopen NMT master interface solution proposed by Ixxat is based on a clear separation between application and communication software. The advantage of such an approach is that an existing application can be more easily upgraded with a CANopen communication interface, without necessitating major changes to the existing software. Possible side effects, which may be caused by the execution of the additional CANopen network software as part of the device application, are thus minimized.

Typically the main controller of an X-ray system is based on a PC or an embedded PC hardware running Windows, Linux or QNX, providing PCI (peripheral component interconnect) or PCIe (PCI express) interfaces. This allows the use of active PCI or PCIe to CAN interfaces for implementing the CANopen NMT master functionality. The CPU on the active CAN interface then processes the entire data exchange and all CANopen services independent of the X-ray control application and exchanges the data to be transmitted or received with the control application over a dual ported memory interface. The CAN interface itself has to conform to the electrical requirements of IEC 60601-1.

CANopen profiles for medical devices

The following CANopen profile specifications for medical devices are available and released as draft standard (DS) or draft standard proposal (DSP) or are in preparation respectively planned (P):

- CiA 412-1 (DS): General definitions for medical devices
- CiA 412-2 (DS): Automatic X-ray collimator
- CiA 412-3 (P): X-ray generators
- CiA 412-4 (P): Patient tables
- CiA 412-5 (P): X-ray stands
- CiA 412-6 (DS): Dose measurement system
- CiA 425-1 (DSP): General definitions for medical diagnostic add-on devices
- CiA 425-2 (DSP): Injector
- CiA 425-3 (P): Electrocardiogram

The CANopen NMT master is based on the Ixxat's CANopen manager protocol software. This software package provides a comprehensive implementation of a CANopen NMT master including the standardized boot-up procedure according to CiA 302-2, a configuration manager according to CiA 302-3 and the handling of PDO (process data object) data by means of network variables according to CiA 302-4. The standardized boot-up procedure includes mechanisms that verify the network consistency meaning that all expected NMT slave nodes and the correct types and versions of NMT slave devices are present in the network and guarantees, that all devices operate with exactly the configuration corresponding to the deployed network and application.

Complementing the standard mechanisms, the CANopen manager implementation provides a unique automatic boot-up functionality, which starts up a network without any given CANopen NMT slave device list and searches and identifies all CANopen devices in the network automatically during boot-up. The control application essentially needs only to verify that all expected (mandatory) devices are available in the CANopen network and operating before starting the X-ray system and the examination. The advantage of this solution is that the number and types of CANopen NMT slave devices in the CANopen network can change without the need of generating and downloading new network configuration data to the CANopen manager each time the network configuration changes. The X-ray control application however usually knows the various network configurations and is able to verify the correctness of the available devices in the CANopen network. Typical scenarios are the different types of injectors, which have to be used and connected to the X-ray system for different types of examinations.

The auto boot-up mechanism also identifies, which data is communicated by the devices. This information may be used to implement dedicated software modules for each possible type of device in the network providing the functionality for exchanging device-specific data with the application. The CANopen manager API (application programming interface) provides the interface to the CANopen manager software executing on the active CAN interface. This API ▶

POSITAL
FRABA

CANOPEN SAFETY
PROVED FOR NEW MACHINE STANDARD

CANopen Absolute Rotary Encoder For Safety Applications

Compact and redundant architecture
Supports standard CANopen and CANopen Safety protocols
SIL 3 according to DIN EN 62061 / 61508
Safety category 4 according to 98/37
PLe according to DIN EN ISO 13849

www.posital.eu

POSITAL GmbH
Carlswerkstrasse 13c, D-51063 Köln,
Telefon +49 (0) 221-9 62 13-0, Telefax +49 (0) 221-9 62 13-20,
www.posital.eu, info@posital.de

consists of a command interface for controlling the CANopen manager (covering services such as init, reset, start boot-up, start/stop network), a diagnostic and status interface and a process data interface. On top of the CANopen manager API, dedicated software modules are implemented for each device type exporting the API required to control the particular devices. A module for a collimator may provide API functions such as CollimatorShutterPosition(), ReportShutterStatus(), SetCommand(), ReportRuler(), ReportTemperature() and so on. A module for an injector may then implement API functions such as SetInjectorState(), ReadInjectorState(), SetVolume() or GetCurrentFlowRate(). Using these API functions, the X-ray control application does not need to know CANopen specific aspects like node-IDs assigned to devices or index and sub-index for addressing parameters or PDO mappings. The device type specific software modules know all necessary information inherently and retrieve all information, which is run-time or configuration-specific from the CANopen manager. A software module with the system control API finally is responsible for controlling system start respectively stop and system consistency.

The separation between X-ray control application on the (embedded) PC and CANopen protocol processing on the active CAN interface board also has the advantage, that if the X-ray control application suffers a malfunction, the CANopen manager still is in able to perform a controlled network shut-down. To support this functionality, a monitoring and watchdog mechanism between CANopen manager and X-ray control application is implemented.

Implementation of a CANopen NMT slave interface

For the implementation of a CANopen NMT slave Interface in a medical device two alternative solutions are possible: the implementation of the CANopen NMT slave interface on the CPU running the device application or using a separate active CAN interface running the CANopen NMT slave functionality.

With the first alternative a CANopen NMT slave protocol stack is implemented together with the application and the application interacts directly with the data and parameters defined in the object dictionary entries. Depending on the functionality and complexity of the medical device an operating system may be used if the application for example consists of several different tasks interacting with the CANopen stack. Particularly having in mind these architectures, Ixxat provides CANopenRT, a CANopen protocol stack, which is specifically designed for the use in real-time or multitasking operating systems. Compared to normal CANopen stacks, which are multitasking aware but block other tasks as long as a task executes API functions of the CANopen stack (even if the other tasks are of higher priority), CANopenRT provides an API, which allows several tasks to use the API without blocking.

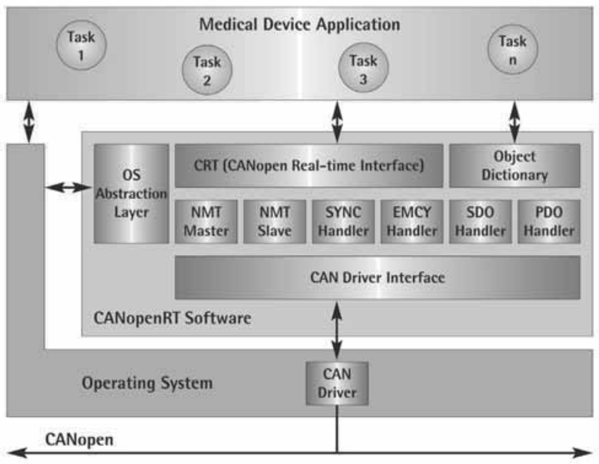

Fig. 3: Software architecture of a medical device based on the Ixxat CANopenRT protocol stack

The second solution, using an active CAN interface, is an interesting alternative for applications, where the medical device already exists in form of hardware and software and needs to be enhanced by a CANopen NMT slave interface without introducing too many changes to the existing hardware and software. In such a solution, the active CAN interface can be connected to the control hardware of the medical device via a local bus interface, via PCI respectively PCIe, Ethernet or USB (universal serial bus). Examples for this solution are injectors, which need to be enhanced by a CANopen interface according to CiA 425-1/2. Typically an injector has a user terminal, which is based on an embedded system with a display running Windows, Linux or QNX. The terminal controls the overall function of the injector and provides the user interface for the operator, whereas the real safety relevant control for performing the injection is performed by a dedicated second embedded system. This embedded system receives commands from and return status information to the terminal. The CANopen interface for controlling the injector from the X-ray device (scanner) needs to be implemented in the terminal. In order to avoid that major changes to the terminal hardware and software need to be applied, an active CAN interface can be developed, which is connected to the terminal by an interface already available on the terminal (e.g. USB or Ethernet) and fitting in form and housing to the terminal (perhaps it can also be integrated into the housing of the terminal). The complete CANopen NMT slave functionality including the object dictionary for the injector is then implemented on the active CAN interface. The CANopen NMT slave handles all PDO or SDO (service data object) read and write accesses to the object dictionary entries depending on the current injector mode and the status of the injector state machine, but does not process the data nor does it execute transitions within the injector state machine. All data and status information received by the active CAN interface is communicated to the injector application on the terminal via the terminal interface. The injector application handles the data and decides when transitions of the injector state machine shall be executed and processes the data to be sent to the CANopen NMT master (X-ray system) over to the active CAN interface. A lean API is provided for the integration of the exchange of data, status and commands between injector application on the terminal and the CANopen NMT slave on the active CAN interface. As all CANopen related functions, mechanisms, timings are handled and processed by the active CAN interface, the injector application only has to handle and process the application process data and the transitions of the injector state machine. This can usually be implemented in the injector application without changes to the overall software architecture and without any influences on timing behavior.

info@ixxat.de