ANT goes KNX
An open platform gateway for ANT and KNX

Luka Samardzija and Friedrich Praus
Institute of Embedded Systems
University of Applied Sciences Technikum Wien
Hoechstaedplatz 5, A-1200 Vienna, Austria
{luka.samardzija,praus}@technikum-wien.at

Abstract

This paper investigates the interoperability of ANT and KNX. ANT is a low power wireless networking technology targeted for embedded systems. KNX is a world wide standard for home and building control. Although the KNX standard already implements a wide range of application fields and further additional implementations e.g. a tabled-pc contorled home environment, ANT is capable to enhancement the KNX expirience. This paper handles the interoperability of ANT and KNX, starting with brief hardware introduction, examining the firm- and middleware, and in a final step analysing the application level and protocol features of both technologies. The main focus of this work is not only a theoretical and informal investigation, but a pragmatic realisation of an ANT/KNX gateway. The result of this paper is a generic and open framework for the junction of ANT and KNX.

1 Motivation & Background

The demographic change towards an aged population in western industrial countries [1] brings up difficulties in health and elderly care systems. This trend has been realized in an early stage and so the research field of Ambient Assisted Living (AAL) came up. Closely linked to AAL and an integral part of it are Ambient Assistive Technologies (AAT). AAT implies a wide range of applications, hardware and technical issues with one similar goal, the intention to support elderly or handicapped people in general in their daily life. The UAS Technikum Wien is engaged in several Ambient Assistive Technologies research projects with a special focus on interfaces between home and building automation and AAT. In this special case the attention lies on KNX and ANT [2]. Although the KNX standard already implements a wide range of application fields, there are not much dedicated AAL solutions among it. In contrast to KNX, ANT needs a short introduction regarding the technology, protocol and application fields. ANT is a low power wireless networking technology targeted for embedded systems and has been integrated into a wide range of personal fitness and health monitoring devices by various hardware developers. ANT is a proprietary low power sensor network technology, providing its own software stack and hardware solutions. ANT fits best in short range and low data rate applications. In recent years ANT has become well established in the sports and fitness sector supporting a wide range of devices, some of which measure heart rate, blood pressure, weight, bicycle speed and cadence. Moreover, some of these devices can be used in the health-care sector. More and more devices in the health care sector are being enhanced with ANT. Even almost all Sony (Ericsson) smartphones out of the Xperia series are capable of ANT, which shows the importance for the mass market. ANT is a useful enhancement for KNX, especially for Personal Health and Vital Data Monitoring. This development is one major aspect which supports the idea for an ANT and KNX interoperability analysis to push KNX towards AAL. The idea of an open platform gateway for KNX and ANT arose due to the core business of the USA Technikum Wien, teaching and execution of student projects, where a flexible and open platform without restrictions is essential.

2 Interoperability Analysis

For the realisation of an ANT/KNX Gateway a deeper introduction into ANT is needed, a detailed knowledge about KNX is assumed. ANT is a radio based network technology with a protocol enhancement called ANT+. ANT+ is a extension to ANT, it is a standardised and uniformed data handling protocol on top of the general ANT network technology. It determines how data is transmitted, handled and interpreted to guarantee interoperability between different devices and various manufacturers. ANT+ defines so called device profiles which have a similar goal like KNX application descriptions.

2.1 Hardware

Interoperability of ANT and KNX without further hardware is not possible. ANT is a dedicated wireless technology, whereas KNX has various propagation media. For KNX-IP, KNX-TP and KNX-PL it is self-evident that a hardware coupler is required. However, for KNX-RF, which operates on the 868 MHz SRD band, a hardware coupler is also required because ANT operates on the 2.4 GHz ISM band. Furthermore, ANT uses a very different modulation and coding scheme. The ANT/KNX gateway has a general structure without being tied to a special media. Both, ANT and KNX, entirely use different transmission methods and thus require hardware couplers for any kind of cooperation.

2.2 Data representation

In both technologies, the network protocol has a key role. It is the junction point, where ANT and KNX are connected by the developed concept. It is essential to distinguish between ANT as basic network technology and ANT+ as protocol enhancement. KNX is specified up to the application layer, whereas ANT is only specified to layer 4 of the OSI model. But for the implementation of a gateway, a data representation of ANT from layer 5 to 7 is also required. This data representation can either be user-defined or ANT+. For example, if KNX defines a data point with a temperature value, this temperature value must also have a defined data representation in the ANT domain. But the general ANT protocol does not provide such a structure. To link ANT and KNX a meaningful data representation on top of the ANT protocol is needed. Usualy it is not reasonable to define a further non standard data representation for something which already exist, but in this case the ANT+ Music Player Control device profile is protected by copyright restrictions and therefore can not be outlined in this paper. For further details to the ANT+ protocol please visit [2]. Nevertheless this mapping is a good example because both, ANT and KNX support such a audio device.

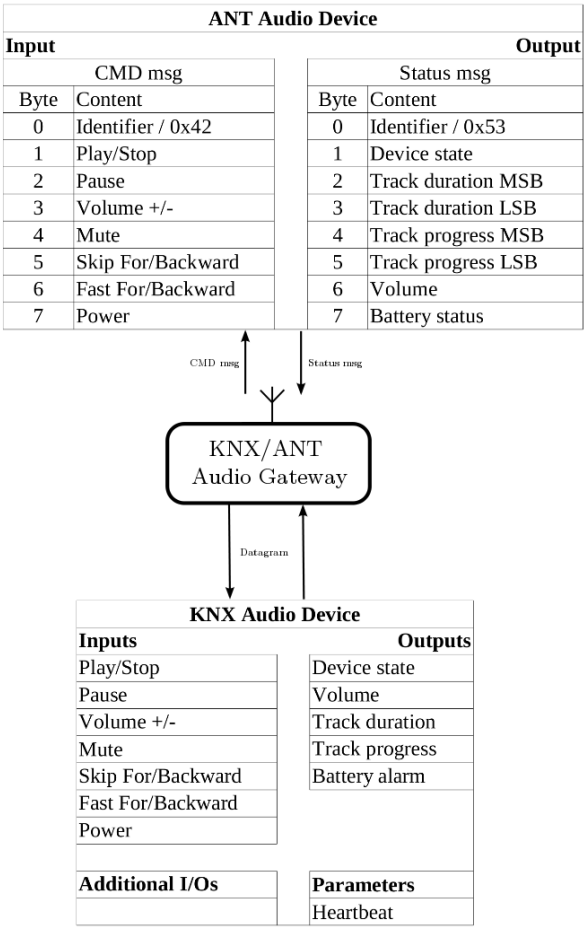

Figure 1: KNX/ANT audio gateway

For demonstration purpose a self defined ANT data representation for audio contol will be stated and mapped to the standardised KNX audio and video (AV) application specification. KNX has no dedicated application description with explicit functional blocks for an AV device. The application specification defines data points which have to be used for AV applications but do not restrict the final implementation of the devices. Figure 1 shows on top a self implemented ANT Audio Device data representation and on the bottom a prototype functional block of a KNX Audio Device. The KNX/ANT Audio Gateway receives data from the KNX network

and forward this data via ANT to the ANT Audio Device. The gateway broadcasts command messages to the ANT Audio Device which answers with status messages.
If an ANT connection between the two devices is established, the nodes constantly exchange massages with a message rate of 4Hz. Each time the ANT audio device receives a command message it answers with a status message. If, for example, the user within the KNX network sends a Play datagram via KNX to the Play/Stop data point (Table 3), the ANT/KNX Audio Gateway maps this data point state change to an ANT command message. The command message will contain the Value=1 in Byte 1 of the message according to Table 1. The ANT Audio Device will answer with a corresponding status message according to Table 2. This schemeatic data exchange was tested with the original audio device profile and the ANT+ simulator.

cmd msg		
Byte	**Content**	**Coding**
0	Identifier	0x42
1	Play/Stop	0=Idle 1=Play 2=Stop
2	Pause	0=Idle 1=Pause 2=Resume
3	Volume	0=Idle 1= + 2= -
4	Mute	0=Idle 1=Enable 2=Disable
5	Skip	0=Idle 1=Forward 2=Backward
6	Fast	0=Idle 1=Forward 2=Backward
7	Power	0=Idle 1=Power 2=Off

Table 1: ANT Audio Device command message

status msg		
Byte	**Content**	**Coding**
0	Identifier	0x53
1	Device state	0=Idle 1=Play 2=Pause 3=Stop 4=Mute 5=Error 6=Fast Forward 7=Fast Backward
2	Track duration MSB	Seconds
3	Track duration LSB	
4	Track progress MSB	Seconds
5	Track progress LSB	
6	Volume	0-100%
7	Battery status	0-100%

Table 2: ANT Audio Device status message

Data point	**Description/Remarks**	**Data point Type**	**Coding**
Play	Start playing - Stop playing	DPT_Start	1=play/start 0=stop
Pause	Temporarily halt playing	DPT_Enable	1=Pause 0=Resume
Skip	Browsing through tracks up and down on by one	DPT_Step	0=Backward 1=Forward
Fast Forward / Backward	Moving fast for/backward within the current track or	DPT_Control Dimming	See coding DPT 3.007
Stop Fast Forward / Backward	Stopp fast for/backward	DPT_Control Dimming	See coding DPT 3.007
Volume Relative Step	Increasing or decreasing the volume, bass, treble, balance (relative by steps)	DPT_Step	0=Decrease 1=Increase
Main Power (standby)	Putting main power to standby	DPT_Enable	1=enable standby 0=disable standby
Audio Mute	Entering and exiting mute mode	DPT_Enable	1 = enable mute 0 = disable mute
Text information (short)	Returns the denomination of the source, track, station, ... (for displaying purposes)	DPT_String _ASCII	See coding of DPT 16.001

Table 3: KNX/ANT Audio Gateway data representation details

2.3 Software

The ANT/KNX gateway contains several software layers. Starting at the hardware-related firmware, across some middleware and libraries, up to the application software. For a clear software design, it is essential to stick to precise interfaces between different functional and logical building blocks. Furthermore, precise interfaces are needed to ensure copy right restrictions. This applies to ordinary copy rights held by companies like Dynastream, but also for open source software licences like the general public licence (GPL).
For interoperability between ANT and KNX, both hardware layers must be studied. For ANT the serial communicatoin interface via USB is the first point of intersection which can be accessed by engineers. In addition Dynastream provides high level software libraries for Windows and Mac. Since the ANT/KNX gateway is Linux' based and no libraries for Linux are provided the interface to the ANT hardware must be realised from scratch. For KNX however, it is enough to interface any layer of the KNX network, since KNX provides a lot of gateways which route between the four KNX native medias. The ANT/KNX gateway uses Calimero for interfacing KNX. The junction point between KNX and ANT are data points which are handled by a custom application program. Since KNX is a distributed system, the ANT/KNX gateway also acts like one node in this network. It wraps ANT data to a KNX format and sends it to the KNX network. The system architecture is designed to enable both directions of data flow. To merge two differing protocols like ANT and KNX, an extensive middleware is needed, which must be capable of ANT and KNX, Calimero is one part of this middleware. Calimero offers mighty functionalities for manipulating KNX networks. The distinction between

ANT and ANT+ must again be considered at the higher OSI Layers. There is no universal ANT/KNX gateway with a one-fits-all character, a special problem requires a special solution. This project should be considered as a template with a flexible software structure which is easy to adopt to changing requirements. In respect to the ANT licensing model the middleware must be ANT+ ready without actually implementing it. This means it must have mechanisms and functions to handle easily ANT+ features but not entirely implement the protocol.

3 Design Requirements

On the basis of current knowledge the following requirements must be satisfied:

- The system should be considered as an experimental platform for active development. It is a completely new development and therefore it will run through several redesigns and improvements. The basic structure should be flexible enough to easily allow changes to hardware and software.
- The system must be open source. The project makes extensively use of open source components and therefore it should also be open source and not violate existing copy rights.
- The system should build up on existing resources. Nowadays many great projects are released under non-restirctive licenses. These projects include thousands of man-years of development and a lot of know-how. This effort is reused in the ANT/KNX gateway, e.g. it makes use of Linux and Calimero. It is a composition of existing resources to a new devices.
- The system components should be highly available, generic and low cost. Hardware and software, as far as possible, must be vendor independent.
- Development tools should be freely available and easily manageable.

These universal requirements should be valid for the whole project, special hardware and software requirements are listed below.

3.1 Hardware

Hardware must be powerful enough to handle a general purpose OS with a JAVA VM (Virtual Machine). In the sense of an usual gateway it should be a small hand-held device, therefore a powerful embedded system is crucial. The minimal hardware requirements are an USB-controller, an Ethernet-interface and a RS232-interface. The most suitable solution is a powerful SOC with all required peripherals on a single chip. The minimum requirement is a 32-bit CPU because this architecture offers a wide rage of possibilities in terms of scalability and availability. The USB-controller is needed for interfacing the ANT hardware. In respect to further development, the platform must be able to connect further devices, for this purpose USB is a quite good choice. A SPI and a I2C interface as well as a few GPIOs are optional for hardware-related developments.
For the very fist stage of development a ready-to-use evaluation board is preferred, because a whole new hardware development is not reasonable.

3.2 Software

The middleware and application software of the ANT/KNX gateway requires a lot of basic hard- and software like TCP/IP, JAVA, USB and many more. Therefore the main focus lies on reusing existing software projects. This strategy is essential, because without the availability of so much open source software it would not be possible to realise such a gateway within such a short time. Reusing open source software enables utilising well developed technologies and frameworks. The basic platform must be able to run a Linux based operating system. For interfacing the ANT-USB2-stick an USB stack is needed, since Dynastream do not provide a driver package for Linux. Any hardware and software which supports the developing process should be integrated in the development environment. This means a graphical user interface with lots of supporting programs and debugging tools. The tools should be freely available and feature a good usability.

4 System Architectue

Figure 2 shows which components have been realised and how they are interconnected. This Chapter is about the structure of the realised ANT/KNX gateway. The base hardware is not that important because it is abstracted by the OS. For the ANT interface the ANT-USB2-stick was chosen. It is a ready-to-use device and completely maintained by the vendor, which is indicated through the red shading. It is interfaced through the USB subsystem of the OS, indicated through the turquoise shading. The next

major component is the ANT-Handler written in C. This piece of SW is a kind of middleware which interfaces the ANT-USB2-stick. For this purpose an ANT library is needed which is aware of the ANT communication protocol and usage. The communication over the USB is handled through the libUSB. The ANT-Handler is connected over a local TCP/IP port to the application program. The white frames indicate that these parts have been realised completely created frome scratch. The application program handles the incoming data, processes and forwards it to the KNX subsystem. The big application program is shaded with a green gradient which means that it utilises third parties open source software. One is the Calimero framework which is integrated in the application program and serves as a KNX interface, the other third party software is JAVA. Because the application program is realised in JAVA it requires a JAVA environment. Calimero utilises the mighty TCP/IP support of JAVA to communicate with the KNX-IP network. After presenting the big picture of the framework, details of the development and realisation will be discussed.

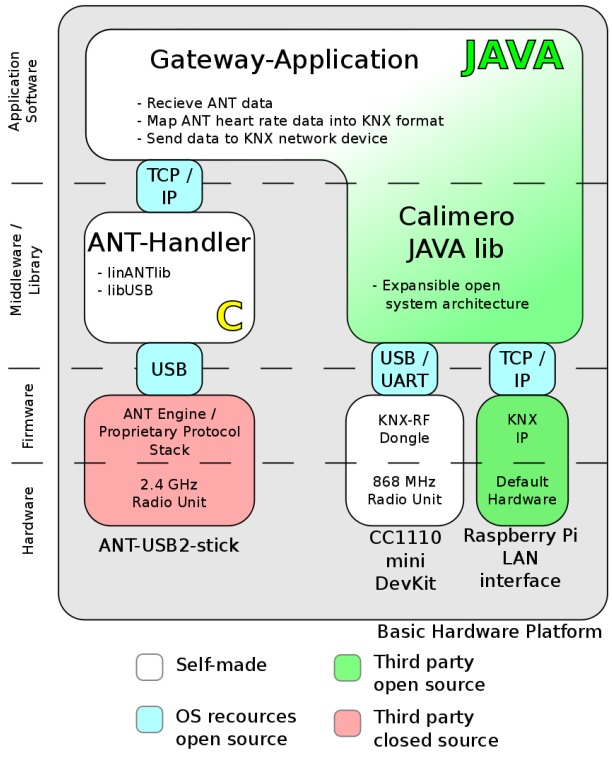

Figure 2: System architecture

5 Prototype Implementation

The first working prototype is a display application. An ANT heart rate monitor records the data and forwards it to the KNX/ANT gateway. The gateway receives the ANT data disassembles it and forwards it to a general purpose KNX display.

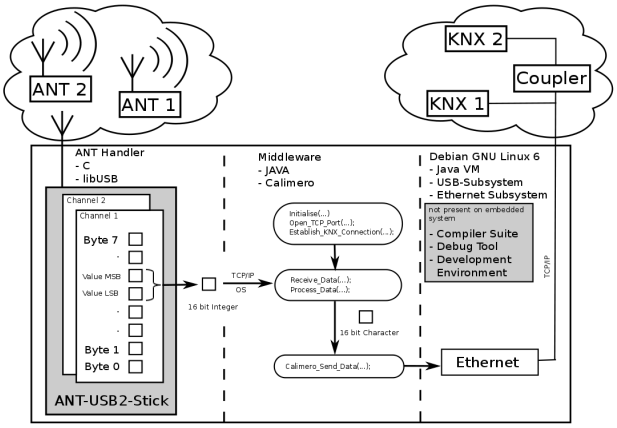

Figure 3: Data flow and mapping scheme

Figure 3 illustrates the data flow and the mapping scheme from origin to sink. The first cloud represents the ANT wireless network where the ANT heart rate measurement device is present. The link is established through the ANT-USB2-stick which is interfaced by the ANT-Handler. The ANT-Handler acts as middleware for interfacing the USB subsystem via the libUSB and it does the first stage data processing. It receives periodic data messages. Knowing the data format, the ANT-Handler picks out the bytes of interest, in this case byte 4 and 5, and processes them. The ANT-Handler assembles the 2 byte long values of interest and typecasts the value into an integer. Then it opens a TCP/IP server port and waits for the client to connect.
The connection between the two programming languages is realised through a local TCP/IP connection. The Gateway-Application is written in JAVA mainly because Calimero is a JAVA library but also because it was intended that the Gateway-Application is platform-independent. As shown in Figure 3, the Gateway-Application first connects to the local TCP/IP server of the ANT-Handler. The Gateway-Application acts as client and receives data from the ANT-Handler. The incoming data is again processed i.e. the integer value is again typecasted to a character which is passed to the Calimero send function. Special attention has to be given to the amount and kind of data which is forwarded to the KNX network. Data from the ANT-Handler is re-

ceived with 4Hz, which is pretty fast compared to the change rate of the value itself. Thus it is not necessary to forward the complete data. To minimise unneeded KNX network traffic the data must be filtered. It makes only sense to forward changed values to the desired destination, which means that the Gateway-Application constantly receives up-to-date data, but forwards only data that has changed. To display the ANT heart rate value, a KNX general purpose display was used. For this purpose a new KNX group address had to be created with the ETS4 toolsuite and then it was associated with the KNX display. The Gateway-Application simply sends the converted data via Calimero to the group address. The whole KNX stack is implemented and handled by Calimero.
The last part of Figure 3 illustrates the OS itself which plays a major role. To utilise all these functionalities it was self-evident to use an OS. All the used open source software, projects and libraries build upon the OS. The grey shaded box in the OS subsystem indicates the differences between the embedded system and the desktop development system. Figure 4 shows the embedded version of the gateway utilizing a Raspberry Pi [3] with a attached ANT-USB2-stick.

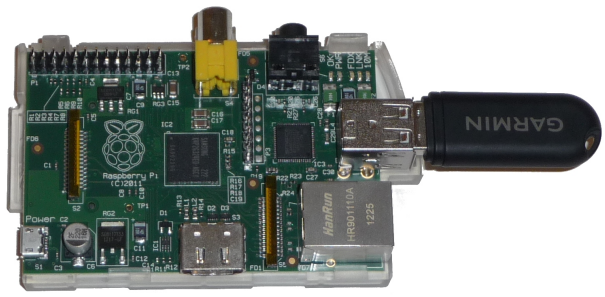

Figure 4: Raspberry Pi Single Board Computer with ANT-USB2-stick

6 Conclusions

Within this paper a basic example of an ANT/KNX gateway has been realised. It shows how a standard KNX installation can be enhanced with medical monitoring capabilities in sense of AAT in a very convenient way. The open architecture of this project is intended for improvements and further development. The interoperability analysis between KNX and ANT shows what must be done to interconnect both technologies. It concentrates on the basic ANT concepts and handles ANT+ capabilities in a limited way due to copy right restrictions. The center of this work is an ANT/KNX gateway framework. For this purpose lots of sub projects were included and linked together. The big accomplishment is not a specialisation to one issue, but a general combination of various domains and technologies. The main focus, ANT to KNX mapping required also profound knowledge and understanding of ANT and KNX. In this paper the basic principles for an ANT/KNX gateway were exposed and a flexible multifunctional open platform for further development was realised.The ANT/KNX gateway is a big, flexible, good partitioned and easy to understand framework. This project is currently under development and as soon as the sources are in a publishable state it will be uploaded to sourceforge, the first will be a Linux ANT library released under the name linANTlib.

References

[1] *European Commission, Demographic Report 2010, Social Affairs and Inclusion, http://ec.europa.eu/social/BlobServlet?docId=6824&langId=en, 2010.*

[2] *ANT homepage, http://www.thisisant.com [Accessed on 29.October 2012]*

[3] *Raspberry-Pi project homepage, http://www.raspberrypi.org/ [Accessed on 29. October 2012]*