

Plena USB and – USB to RS-232 convertor matters



1. Introduction

The Bosch Plena Voice Alarm System firmware upgrade requires a RS232 connection. Also special open interface connections are communication data via this interface. This times less PC's are equipped with RS232 ports and there is a good chance that if they have one, the port is derived from the internal USB hub.

The market offers a lot of possibilities to cover the RS232 problem with use of USB-RS232 convertors. This is because USB is a quite common interface those days. But what should solve the problem seems to turn into a nightmare for some installers.

This application note explains what happens, why it happens and how the hardware incompatibility can be solved.

The application note is divided into 3 chapters:

- 1) Inexpectable hardware and software problems using USB to RS232 convertors
- 2) Selected USB-RS232 convertor.
- 3) USB device connection problems

2. Inexpectable hardware and software problems using USB-RS232 convertors

Noticed problems:

Although people think they have good USB to RS232 convertors, in practice there are problems upgrading the firmware and controlling the system using open interface enabled applications. It's not always clear where the fault occurs, because in a lot of cases the USB to RS232 convertor works correct. And of course, at a certain moment it is not possible to get the firmware correct into the system and / or network supervision faults occur using applications with open interface connections. If those inexpectable problems arise, please take a serious note of this application note. After long research and experience we have gathered solutions that can solve or reduce those problems.

USB to RS232 conversion

RS232 and USB are both serial communication standards to connect peripherals to computers, but they are totally different in design. A simple rewired cable is not able to connect RS232 devices to a computer with only USB ports. A USB to RS232 convertor cable contains electronics, and the success rate depends on the capabilities of this electronics and the device driver software shipped with the converter. Sounds very simple to get the job done.

Unfortunately this simple idea is implemented to a lot of cheap USB to RS232 convertors. and of course in a lot of situations it will work. But before buying a USB to RS232 converter, make yourself familiar with the aspects written in this application note.

Differences from the application driver point of view

RS232 is specified for serial communication on a 1:1 base. Regarding this specification, the RS232 defines the interface layer from the OSI model and not the application layer.

To use RS232 communication, the application software must be written on both ends of the connected RS232 cable. The data communication protocol is not bounded to any definition and is in that way complete free for own implementation. RS232 ports also can be accessed directly by an application, via a device driver in the operating system.

What about USB? USB is a bus system which allows more than one peripheral to be connected via one host USB port. USB bubs can be used in the USB chain to extend even more devices connected to the same USB port. The USB standard not only describes the physical properties of the interface, but also the protocols to be used.

Because of the complex USB protocol requirements, communication with USB ports on a computer is always performed via a device driver.

And here you already can see where the problems arise. RS232 developers have lots of freedom where it comes to RS232 protocols and the ports are often (almost) directly accessible from the software program. Baudrate, databits, hardware and / or software flow control can be changed within the application.

The USB interface does not give this flexibility.

However, if an RS232 port is used via an USB to RS232 converter, this flexibility should be available somehow. That's exact why a second device driver is necessary. This driver emulates a RS232, but communicates via USB standards.

Many applications expect certain timings when RS232 communication is used.

With hardware ports directly fitted in a computer it most of the time does not turn in to problems. The application communicates directly, UART, and everything happens within a well defined time frame. The USB bus however is shared by several devices. The timeframe in which specific RS232 actions are performed might not be so well defined as in the direct hardware port approach. Also, the RS232 emulation driver working on top of the complex USB driver might add extra overhead to the communications, resulting in delays.

Hardware specific problems

RS232 ports which are physically mounted in a computer are often powered by three power sources: +5 Volt for the UART logic, and -12 Volt and +12 Volt for the output drivers.

USB however only provides a +5 Volt power source.

A good USB to RS232 converters uses an integrated DC/DC converter to create the appropriate voltage levels regarding the RS232 standards, but a (very) cheap implementation only uses the +5 Volt voltage which is directly derived from the USB output.

If we take a close look to the RS232 standard, we see in practice that the RS232 ports recognize a voltage above 2 Volt as a *space* signal ("1") and a voltage of 0 Volt or less is recognized as a *mark* signal ("0"). Be aware that these values are not according the original specified standard values. The original RS232 standard specifies that all voltages between

-3 Volt and +3 Volt result in an undefined signal state.

Nevertheless, the well known maxim MAX232 series of RS232 driver chips accepts those non-standard voltage behaviors. Although the outputs of this maxim driver swing between

-10 Volt and +10 Volt and the specifications says that the inputs recognize all signals swinging below 0 Volt and above 2 Volt as valid signals space and mark signals. Be aware of the word "below"!

This non-standard behavior of RS232 inputs makes it difficult to select the right USB to RS232 converter. If you connect and test this cheap USB to RS232 converter might it work in a lot of cases. So nothing wrong with your convertor you think.

But this convertor can particularly become a problem with industrial applications like our Plena Voice Alarm System.

Also low-cost computers are often equipped with cheap RS232 drivers or with cheap RS232 convertors derived from the USB ports. During you test, everything look fine again.

The chances that RS232 ports from low-cost computers accept signals in the 0..5 Volt range are simply higher than with industrial equipment. Those equipment is most of the time well designed to be immune for noise on de data lines.

Other hardware specific problem arises with handshaking to prevent buffer overflows.

Not all USB to RS232 converters provide these hardware handshaking flow control lines.

Plena Voice Alarms does not use the hardware handshaking and again you have a chance that cheap USB to RS232 converters will work without problems a lot of times.

Now the background of the USB and the RS232 way of working is clear, the correct convertor must be chosen.

USB to RS232 converter selection criteria

When choosing the right USB to RS232 converter, look at the following potential problems:

- Take a convertor that meets the requirements and specifications of the RS232 standards.
- Keep in mind that some Bosch PA equipment (e.g. the IP audio interface) require handshaking. If hardware flow control is required, make sure that these inputs and outputs are present on the converter.

3. Selected USB-RS232 convertor

After reading all problems which you can expect using incorrect USB-RS232 convertors, we realize that searching a good RS232 convertor can take some time. Often we are asked for a link or some product reference of a good convertor.

This chapter only refers to a selected and tested USB-RS232 convertor, but feel free to find a good RS232 convertor, meeting all requirements yourself. There are a lot of good USB-RS232 convertors available on the market, but unfortunately also a lot of incorrect (cheap) RS232 convertors.

One good convertor is e.g. the FTDI - US232R-100

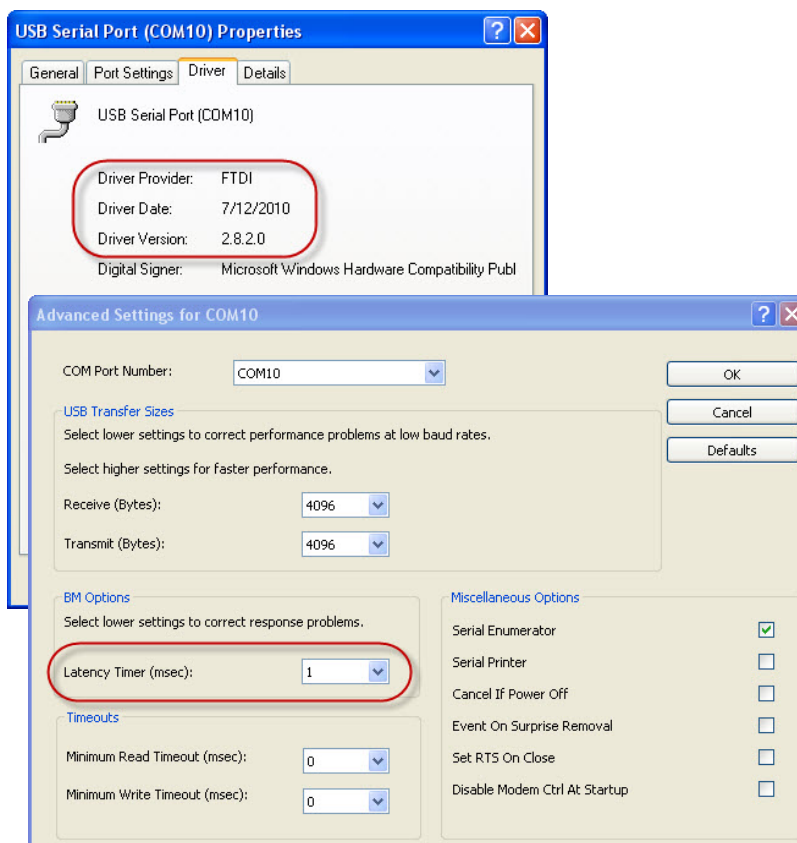
This convertor can be ordered via e.g. Farnell.

The direct link to this product is;

<http://nl.farnell.com/ftdi/us232r-100/cable-usb-to-rs232-serial-converter/dp/1329310>

The convertor has a build in DC/DC level convertor which sends RS232 space an mark signals as -5Volt and + 5Volt signals. The driver also enables some more communication setting like latency etc. Those setting are important, cause using the standard setting, you still get communication errors. The standard latency setting of 5-10ms is often used by USB convertors. If you are lucky they have set a low latency. But you need to test a lot of convertors then. The advised FTDI chip convertor meets the RS232 standard.

The pictures below show the driver version and the tested USB convertor settings



USB-RS232 convertor connection



- 1) Install the USB – RS232 driver software, (tested driver version 2.08.02 (from fdtichip.com))
- 2) Connect the USB-RS232 convertor
- 3) Set the driver latency to 2ms or less
- 4) Connect a full wired RS-232 cable to the convertor and to the Plena Voice Alarm Controller
- 5) Proceed with the firmware update and/or
- 6) Start the RS-232 open interface application.

The firmware and the open interface application should run without any problem.

4. USB device connection problems

People often have problems with the Plena device detection on their PC using the USB connection. The most common problems are gathered in this chapter. Following this application note problems that can occur should be solved or reduced seriously.

Software configuration installation:

Always first install the configuration software before any USB connection with the Plena Voice alarm controller is made. Once Windows has connected the wrong driver to the wrong device ID, it will not be easy to correct this. In some cases, reinstalling the software solves the problem, but there are also applications found where the connection to the Plena will not become successfully anymore.

Using the correct USB cable:

Ensure yourself that you are using a correct USB cable.

In time, there came a several USB speed standards.

It ranges from v1.0 (1,5Mbps), v1.1(12Mbps), v2.0 (480Mbps) to v3.0. (5Gbps)

Those different speed standards also require different USB cables.

You cannot use simply use USB 1.0 cable for USB 1.1 or USB 2.0 connections.

The worse thing is that you cannot see on the outside which USB cable you have.

Most of the time a label is on the USB cable when you buy it.

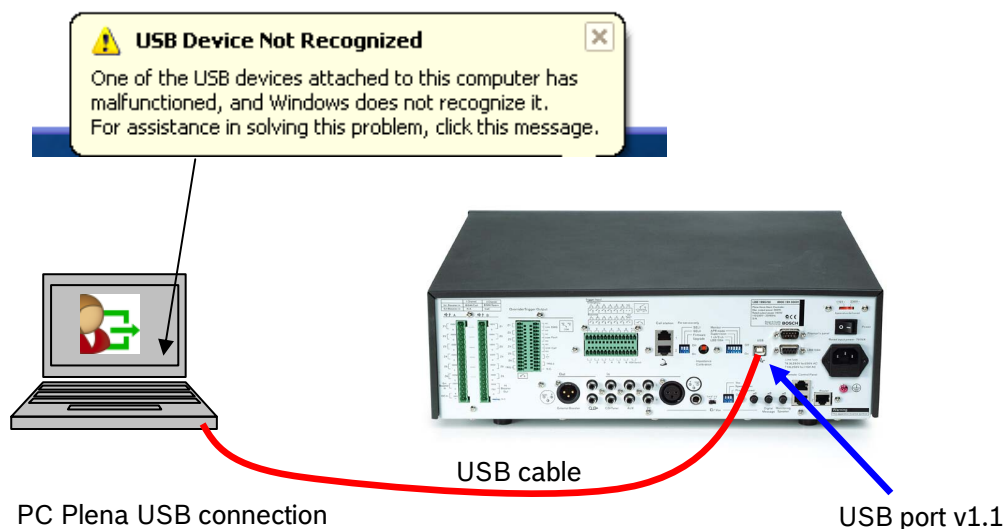
The Plena Voice Alarm controller is compatible with USB v1.1, and this requires at least a 1.1 compatible USB cable. It seems to be that some cheap USB v1.1 cables does not meet the USB-IF 1.1. speed requirements and this can result to e.g. connection problems.

If you experience this problem, we advice you to use a USB v2.0 cable. Those are more expensive, but will not drop the recognition, the connection and the communication with your Plena devices.

Typical errors seen using incorrect cables are: Communication errors and reports like Unknown devices.

This result into wave files that are not uploaded (upload stops), applications that stop responding and you run into situations where you are not able to get any correct recognitions of you device.

All done could be caused by wrong or cheap USB cables.





5. Definitions, acronyms and abbreviations

AN	Application note
VAS	Voice Alarm System
MIC	Microphone level
Pcs	Pieces
EMG	Emergency
RS232	Serial communication interface
USB	Universal Serial Bus
Open Interface	Communication protocol to control the VAS
BGM	Background music
VOX	Voice activate function
Mic.	Microphone
FPA	Fire Panel
Mbps	Mega bit per second
Gbps	Giga bit per second



6. About

Bosch Security Systems B.V.

Sales Support Organization PA
(Product) Application Specialist EMEA
Wilbert Maximus, B.Sc.
P.O. Box 90106
4800 RA Breda, Netherlands
www.boschsecurity.com