K6Bt

Bluetooth Serial Adapter

Extended Protocol Documentation

Version 1.1 final

For newest version email to Clemens Gerlach <gerlach@k6-team.de>

Nov 1, 2011

1 Introduction

The K6Bt Bluetooth Serial Adapter is designed to be operated with glider avionics. For communication between the instruments or for supplying data to an attached PDA these instruments normally utilize proprietary NMEA-0183 sentences send over a serial RS-232 interface. Newer PDAs (e.g iPAQ 210) do not provide any serial interface on their extension port. To connect the avionics to such PDAs the K6Bt operates as transparent Bluetooth Serial Adapter.

During normal operation the communication happens at a constant serial baud rate. Since a bluetooth connection always runs at constant speed the baud rate for the serial interface can be set at the K6Bt using either DIP switches or jumpers (see chapter 2). Problems arise when the attached device (logger/flight computer) needs to communicate at non-constant baud rates. These situations normally happen when the instruments are in direct communication with the PDA for setup, waypoint uploading or flight downloading. An example is the Volkslogger by Garrecht Avionik GmbH which can change it's baud rate during flight declaration and downloading.

1.1 Bluetooth SPP

The K6Bt uses the Bluetooth Serial Port Profile (SPP). The connection at the PDA shows up as a virtual serial port. This virtual port can be used by any application using the OS's API just as any normal physically available serial interface . The virtual COM port also allows setting of it's baud rate. It's important to know that this baud rate does **not** affect the data throughput for the bluetooth connection. The baud rate setting is just for the virtual (emulated) serial port and has no impact on the transmission speed between the PDA and the K6Bt.

The Bluetooth SPP profile protocol requires that the virtual serial port parameters (including the baud rate) are send as informational messages to the attached device. Thus (in theory) the K6Bt should be able to decode these messages and set it's physical serial port parameters accordingly. Sadly, during development of the K6Bt it revealed that the information about changed serial port configuration is not always send by the bluetooth stack of the PDA's or PC's operating system. Even worse every stack seems to behave different, some of them even send wrong information.

If you like to confirm or "play" with this behavior, we can supply a debugging firmware for the K6Bt which decodes this informational bluetooth messages as human readable text.

1.2 Baud rate changing: The K6-Team way

For a reliable product we decided not to rely on the bluetooth informational messages for changing the baud rate. Thus the K6Bt can be operated in two modes:

1. For simple communication a fixed baud rate can be set on the DIP switches/jumpers at the K6Bt. This baud rate never changes on the serial line regardless what the application on the PDA sets on the virtual com port.
2. The K6Bt can be switched into a mode where the application on the PDA is able to change the serial baud rate by using a proprietary protocol. This protocol is designed to be easily implemented in the serial driver of any application, thus requiring only little changes to existing implementations.

2 K6Bt Settings

2.1 Firmware < 2.0

The most important settings of the K6Bt Bluetooth Serial Adapter can be changed by using the DIP switches numbered 1 to 6. The following tables outline their assignment.

Switch 1	
ON	TX line connected to IGC connector Use if the K6Bt should be able to send data to the connected avionics
OFF	TX line disconnected from IGC connector Use if other device sends data to to the avionics and the K6Bt only forwards serial data **to** the bluetooth link

Switch 2	**Switch 3**	**Switch 4**	**Baudrate**
OFF	OFF	OFF	2400
ON	OFF	OFF	4800
OFF	ON	OFF	9600
ON	ON	OFF	19200
OFF	OFF	ON	38400
ON	OFF	ON	57600
OFF	ON	ON	115200
ON	ON	ON	Auto[1]

Switch 5	
ON	Enable protocol extensions. All commands and protocol descriptions in this document only apply if switch 5 is set ton ON
OFF	Disable protocol extensions. The K6Bt transparently forwards all data between the bluetooth link and the serial line using the baud rate set by switches 3,4 and 5. The protocol description throughout this document do no apply.

Switch 6	
	Not used by current firmware

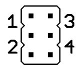
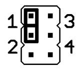
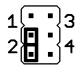
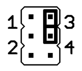
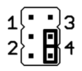
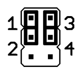
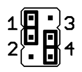
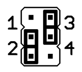
Setting of the Bluetooth ID and the Bluetooth PIN can be done by connecting the K6Bt using a crossed (RX and TX lines) cable to the PC and using a terminal software (e.g. Hyperterm) at 19200 Baud. To enter the configuration menu send three times 'k' to the K6Bt within 3 seconds after power-up (while both LEDs are on).

1 Baud rate setting “Auto” tries to automatically detect the serial baud rate of the connected device. This only works if the device sends NMEA-0183 conforming sentences (at least one sentence per second). During the search for the correct baud rate the red LED flashes twice. As soon as the baud rate is detected the red LED stays of.

2.2 Firmware >= 2.0

In newer versions of the K6 Bt most of the settings are done in the setup menu accessed by a terminal software and serial cable. Please consult the K6 Bt manual on how to change the settings.

Within the setup menu the proprietary K6 Bt protocol can be enabled or disabled. The baud rate setting on the serial port is done using jumpers on the circuit board:

Jumper	Baud rate
	Auto
	4800
	19200
	38400
	115200
	2400
	9600
	57600

3 Protocol

The K6Bt transparently forwards data from the bluetooth connection to the serial port except when receiving the escape character 0xa5. Upon reception of 0xa5 the K6Bt enters command mode. In command mode the next received byte (command) determines the action to be performed. After receiving the command byte the K6Bt continues with transparent forwarding until the next escape character is received.

After executing the command no confirmation is send back via the bluetooth connection (exceptions are the "Get Identification, binary" and "Get Identification, NMEA" commands. See below.)

Reception of an illegal command byte after the escape character is silently ignored and the K6Bt returns to transparent forward.

Data from the serial port is always forwarded to the bluetooth connection and even the byte 0xa5 triggers no specific action.

By choosing the escape character as 0xa5, protocols using only ASCII characters (< 0x80) do not need any modification. When binary data is transfered from the bluetooth connection to the serial port every occurrence of 0xa5 in the payload data just needs to be doubled (see command 0xa5).

The following commands are currently implemented (Version 1.0):

No Operation

Command Byte:	0x00
Description:	Does not do anything. By sending a 0x00 one can be sure the K6Bt is in transparent forward mode.

Send 0xa5

Command Byte:	0xa5
Description:	Sends a 0xa5 to the serial line. Since a 0xa5 is the escape character to get into command mode this command enables the transmission of 0xa5 to the serial line. A serial driver that sends it's data through the K6Bt just needs to double each occurrence of 0xa5 in the payload.

Get Identification, binary

Command Byte:	0x10
Description:	The following byte sequence is immediately send back to the bluetooth connection: 0xa5, 0x00, 0x4b, 0x36, 0x42, 0x74, <major>, <major> XOR 0xff, <minor>, <minor> XOR 0xff, 0xa5, 0x00 Where <major> is the major version and <minor> the the minor version of the protocol, both as binary numbers. Currently major is 0x01 and minor is 0x00. This command can also be used to detect the K6Bt and its correct setup. The K6Bt will not answer to this command if not switched into the proprietary protocol mode (using the on board DIP switches)

Get Identification, NMEA

Command Byte:	0x11
Description:	The K6Bt waits up to 1.5 seconds to receive a newline (<LF>) on the serial interface. Upon reception of a newline or after timeout the following ASCII sequence is injected into the data stream: $PK6BT,ID,<major>,<minor>*<cksum><CR><LF> <major> and <minor> are the protocol version as ASCII decimal numbers. <cksum> the normal NMEA checksum. This command can also be used to detect the K6Bt and its correct setup. The K6Bt will not answer to this command if not switched into the proprietary protocol mode (using the board DIP switches)

Change baud rate, wait

Command Byte:	0x3r
Description:	Changes the baud rate on the serial line. If there are still bytes in the tx buffer of the K6Bt these are send first. After the tx buffer is empty the new baud rate is applied. The lower 4 bits ('r') determine the new serial baud rate: r: 0x0 - 2400 0x1 - 4800 0x2 - 9600 0x3 - 19200 0x4 - 38400 0x5 - 57600 0x6 - 115200 0xf - baud rate selected by DIP switches Setting to other values is silently ignored. After power-up and after disconnection of the bluetooth link the baud rate is set to the one selected by the DIP switches/jumpers. If “auto” is selected a new baudrate autodetect is started on the serial port.

Flush buffers

Command Byte:	0x4r
Description:	Flushes the K6Bts serial buffers. The lower 4 bits ('r') determine which buffer to flush: r: 0x01 – flush serial RX buffer 0x02 – flush serial TX buffer 0x03 – flush RX and TX buffer

4 Debugging

On request we provide a firmware for the K6Bt which prints out the bluetooth port configuration requests as well as the decoded proprietary protocol messages on the serial port. To see these messages the K6Bt should be connected to a serial terminal (using a crossed cable) listening at 115200 Baud (8N1).

5 Document Changelog

Apr 28, 2008
- Development firmware version 01a_dev; Initial release

Jul 19, 2008
- Added K6Bt settings section
- Changed protocol section to reflect development firmware 01b_dev

Oct 12, 2008
- Updated Firmware Changelog (v01c_dev, v01d)
- Changed document version to v1.0 final

Nov 01.2011
- Note in “Change baud rate, wait” command, that DIP switch/jumper selected baudrate is selected after bluetooth link disconnect
- Added setup/jumper settings for Hardware >= 8.1.0, Firmware >= 2.1 (new version)
- Changed document version to v1.1 final

6 Firmware Changelog

v01a_dev (Apr 2008)
- Initial development version release

v01b_dev (Jul 2008)
- added “Flush buffers” command (0xa5 0x4r)
- removed “Change baud rate, flush”, use “Flush buffers” followed by “Change baud rate, wait” instead
- Baud rate can be changed to DIP switch setting (0xa5 0x3f)
- Changed behavior after startup in extended protocol mode: Starts up using the baud rate selected by the DIP switches (old behavior was to disable the serial port until the baud rate was set the first time).
- Changed behavior: After closing the bluetooth connection the baud rate is set back to the one selected by the DIP switches. If the DIP switches select “Auto” the baud rate gets detected again (like at power-up).

v01c_dev (Oct 2008)
- fixed a bug causing loss of data at high data rates (rs232 → bt)
- setup menu always shows baud rate chosen by DIP switches even if the proprietary protocol is enabled
- paired devices get removed after pin change
- id is set to “K6Bt” (without braces) if user settable part is empty

v01d (Oct 2008)
- no changes. First public “non-development” release

v2.1 (Nov 2011)

- new version system (different versions for HW and FW)
- ported to new K6 Base hardware/processor platform
- better buffer management allows datastream of continuous 115200 baud RS-232 $\rightarrow$ Bluetooth without losing data
- fixed bug that prevented the K6 Bt from returning it's ID (Commands “Get idetification, binary” and “Get identification, NMEA”) when the serial port was busy with autobauding.