BTM44X – Firmware Upgrade

Application Note *v1.0*

Introduction

The BTM44x Enhanced Data module has a feature set, in firmware, which is continually evolving.

At any time the module has a specific production release of the firmware. At the same time on application, newer engineering builds of the firmware can be made available. These are released to fix issues that have been discovered since the production release or have new profiles and features.

The module is available so that one of two interfacing protocols can be activated over the UART port. One is the multipoint packet protocol (MP) mode and the other is the AT protocol.

In AT mode, the firmware version information is sent in response to the ATi3 command.

In MP mode, the firmware version information is sent in response to the CMD_INFORMATION[0] command.

Upgrading firmware over the UART consists of temporarily changing the protocol to one which is used by the boot loader, is packet based and is specific to the upgrade process, called BCSP mode. It is not documented in the user manual given that it is only used for upgrades and that process is managed via PC-based utilities, so it is normally transparent to a host. The only time this protocol is exposed to the host is when an upgrade is interrupted and the module has incomplete firmware. In that case, the boot loader detects corrupt firmware and remains in boot mode, exposing BCSP mode to allow further attempts at firmware downloads.

It is also possible to upgrade the firmware over a SPI bus interface which is exposed via four dedicated pins on the module labelled as SPI_CSB, SPI_MISO, SPI_MOSI and SP_CLK. See the user manual for full pinout.

Upgrading firmware over the air is not planned as it is not inherently supported by the chipset vendor.

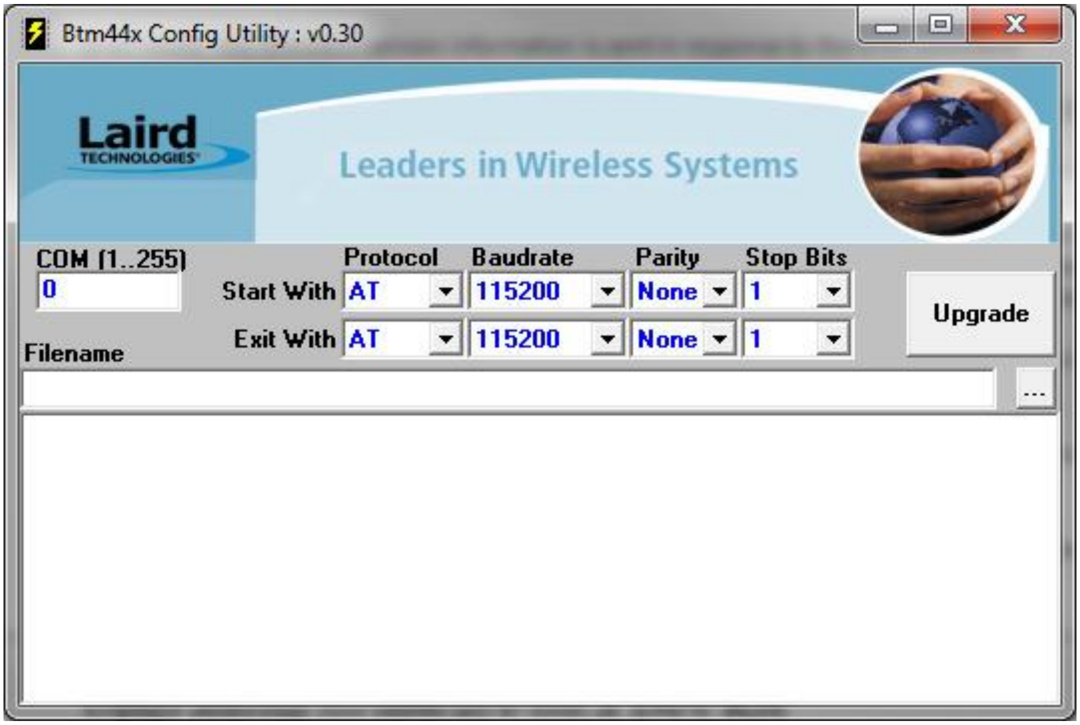
New firmware is deployed in files with extension .dfu, which are self-contained with all the necessary information to expedite a firmware upgrade.

This document describes measures a designer can take to make the process of upgrading the firmware easier based on understanding the various options.

Upgrading firmware on a Development Kit

The BTM44x module on a dev kit has its UART exposed directly to a Windows PC via a USB virtual COM port. In this case the process of upgrading the firmware is as simple as launching a utility called Btm44xConfig.exe in interactive mode and pointing it to the new firmware file (with a .dfu extension).

The main window for the firmware upgrade utility is as per the snapshot below.



COM (1..255): 0

	Protocol	Baudrate	Parity	Stop Bits
Start With	AT	115200	None	1
Exit With	AT	115200	None	1

Filename

Change the COM port as per the setting in the PC for the dev kit, specify the protocol and COM port parameters for ‘Start With’ and ‘Exit With’, select the filename in the appropriate text box (which may be automatically selected if there is only one .dfu file in the same folder as the utility) and then click **Upgrade**.

The whole process takes up to 5 minutes. On successful upgrade the module will be left in the state specified by the parameters labelled as ‘Exit With’

If the upgrade process was previously aborted, i.e. via a power interruption, the module starts in BCSP boot loader mode. The utility is designed to detect for that state if the ‘Start With’ parameters do not yield a response from the module. It will check for BCSP mode and if that results in a positive detection it will download the new firmware. It may try to exit with the parameters specified in ‘Exit With’. That final phase may fail. If so, manually try to detect the module by trying MP and AT modes at 115200 and 9600 baud.

The firmware upgrade utility can operate in automatic mode by supplying relevant parameters as command line arguments. For example, if the kit is on COM34, the new firmware file is called NewFirmware.dfu and the module is in AT mode but must exit in MP mode, then the command line arguments will be

```
COM=34 "DFU=NewFirmware.dfu" PROTs=AT BAUDs=115200 PARs=NONE STOPs=1 PROTe=MP
BAUDe=115200 PARe=NONE STOPe=1 UPGRADE
```

If the utility is launched with these arguments, it starts and exits on completion without further intervention.

If the upgrade process fails, a log file (which is always) offers full trace information to diagnose the failure.

Upgrading firmware on Host Hardware

In real world applications the UART interface of the BTM44x module will be most likely directly connected to the UART port of a host CPU which is managing the module and the UART signals of the module are not exposed to allow a Windows PC to use the Btm44xConfig utility to upgrade the firmware. In this scenario it is not possible to upgrade the firmware.

However, it is possible to augment the host/module hardware design to expedite a firmware upgrade.

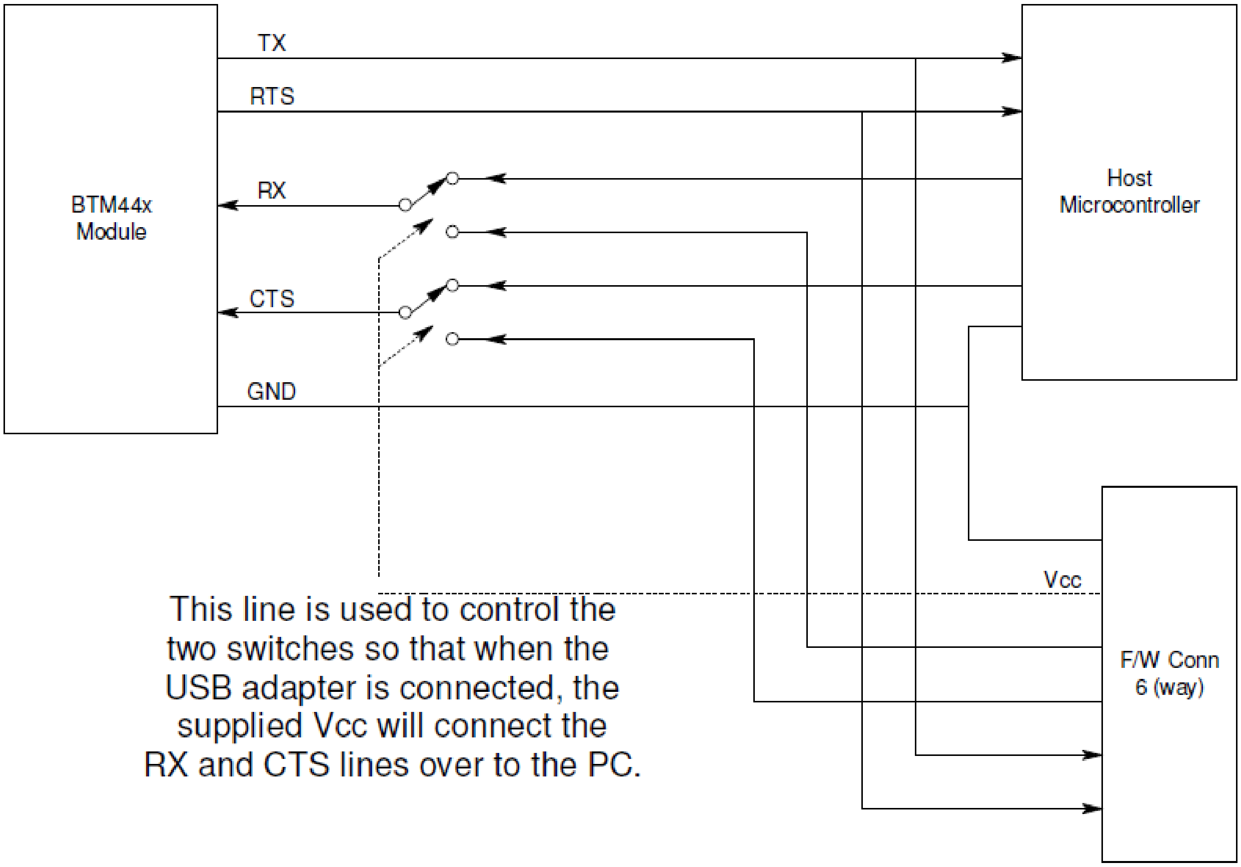
There are several options available and they are described in subsequent sections of this document.

Option 1: Multiplex the UART lines

This upgrade option requires direct connection to RX, TX, CTS and RTS lines of the module, via appropriate RS232 level conversion, to a serial port on a Windows PC.

In that case, 2 of the 4 UART lines of the module cannot be directly exposed because the RX and CTS input lines will be driven by the host microcontroller and hence cannot also be driven by a Windows PC.

One solution would be to incorporate the hardware logic illustrated below and use a USB to Serial adapter as described at the website http://www.ftdichip.com/Products/Cables/USBTTLSerial.htm This has the further advantage of not requiring RS232 level conversion.


TX
RTS
RX
CTS
GND
BTM44x Module
Host Microcontroller
Vcc
F/W Conn 6 (way)
This line is used to control the two switches so that when the USB adapter is connected, the supplied Vcc will connect the RX and CTS lines over to the PC.

In this arrangement the extra cost to the bill of materials consists of components that implement the multiplexing of the RX and CTS lines. This multiplexing can be via a header/jumper arrangement or it can be logic gates driven by the Vcc line of the 6 way USB F/W Conn that allows the USB to Serial adapter to gain control of the BT module.

With this arrangement, while the external PC is interacting with the BT module, any outgoing comms traffic will also reach the host microcontroller. This does not detrimentally affect the operation of the controller. If this could be an issue, we advise you also multiplex the TX line from the modules. The host controller may also sense the Vcc line from the F/W Conn to provide logic to ignore UART traffic when it is asserted.

Option 2: Bridge UART traffic

This upgrade option requires that the host controller has permanent connections to the UART port of the BT module AND has some other means of communicating with the outside world (a TCP/IP connection or a second UART port exposed so that a Windows PC has direct connection).

If a TCP/IP connection is the other means of connection, you will need a windows PC with TCP/IP to virtual COM port bridge software (an example: http://www.tacticalsoftware.com/products/serialip/index.htm). This TCP/IP to Serial port bridge is required to use Btm44xConfig.exe.

The host microcontroller which controls the bridge for the UART traffic must have firmware functionality to prepare the BT module for a firmware upgrade. When triggered, the firmware must revert the module to BCSP boot loader mode and then bridge all bidirectional traffic between the BT module and the Windows PC.

When the host controller is triggered, it shall perform the following actions, where <cr> implies the carriage return character 0x0D and <del> is the backspace character 0x08.

1. If the BT module is in MP mode, go to step 9.
2. Send command ATS520? and store the returned value for later use.
3. Send command ATS520=115200<cr>
4. Send command ATS522? and store the returned value for later use.
5. Send command ATS522=1<cr>
6. Send command ATS523? and store the returned value for later use.
7. Send command ATS523=0<cr>
8. Advance to step 16.
9. Read content of S register 240 and store for later use using command CMD_READ_SREG.
10. Write 115200 to S register 240 (0xF0) using the MP command CMD_WRITE_SREG.
11. Read content of S register 242 and store for later use using command CMD_READ_SREG.
12. Write 1 to S register 242 (0xF2) using the MP command CMD_WRITE_SREG.
13. Read content of S register 243 and store for later use using command CMD_READ_SREG.
14. Write 0 to S register 243 (0xF3) using the MP command CMD_WRITE_SREG.
15. Write 2 to S register 255 (0xFF) using the MP command CMD_WRITE_SREG.
16. Send command CMD_RESET. The module wakes in AT mode at 115200,N,8,1
17. Configure the host microcontroller UART for 115200,N,8,1 operation and AT protocol.
18. Send command ATS32767=? <cr> and store the returned decimal value (in the range 0 to 255).
19. Send command ATS32767=n<cr> where n is the decimal value returned in step 12.
20. Send command AT+BT!<del>!<cr> where <del> is the 0x08 ASCII character.
21. The module is in BCSP mode and sends BCSP sync packets at 115200,N,8,1. The host can verify this by looking for at least one 5 byte sequence C0 01 0B 09 90.
22. Activate the bridge. From here on, all traffic is bridged across bidirectionally.
23. Launch the firmware upgrade utility on the Windows PC.
24. While the bridge is active, restart a 40 second timer as long as there is data traffic from the PC towards the BT module
25. If the timer expires, this signals that the firmware upgrade is complete. Disable the bridge.
26. Reset the BT module by either toggling the hardware RESET line of the module, or by sending a BREAK for about 500 milliseconds.
27. The BT module wakes in AT Mode at 115200,N,8,1.
28. Write to S Reg 520 the value returned in either step 2 or 9.
29. Write to S Reg 522 the value returned in either step 4 or 11.
30. Write to S Reg 523 the value returned in either step 6 or 13.
31. If the module was in MP mode at step 1, send command ATS9255=1.
32. Reset the BT module by either toggling the hardware RESET line of the module, or by sending a BREAK for about 500 milliseconds.
33. The module is now in the same mode and state as at step 1, but with new firmware installed.

Option 3: Upgrade via SPI bus

This option requires that the five pins on the BT module be exposed via appropriate means so that a Windows PC connected by USB to SPI bus adapter may run a utility and upgrade the firmware. This is NOT Btm44xConfig.exe, used previously.

Instead, the firmware upgrade file consists of a pair of files, one with a .xdv extension and the other with .xpv. These files are not made generally available, but will be available to customers, on application, who are willing to sign an NDA with Laird so that a third party utility can be made available.

The USB to SPI bus adapter can be purchased online (e.g. http://parts.digikey.com/1/parts/1406287-converter-usb-spi-devsys-1808-1a.html). The five pins to make available for external connection to the USB to SPI adapter are:

GND	Pin 7
SPI_CSB	Pin 8
SPI_MISO	Pin 9
SPI_MOSI	Pin 10
SPI_CLK	Pin 11

The USB to SPI bus adapter documentation describes the RJ45 socket pinout, enabling an appropriate cable to be made up from an existing network patch cable.

Revision History

Revision	Date	Description	Approved By
1.0	24 Nov 2011	Initial Release	Jonathan Kaye