Programming the BL233B Internal EEProm

28th June 2006

Contents

1 Introduction

The BL233B datasheet has full details of EEProm configuration, macros and programming commands. The datasheet is the primary command reference. Always get the latest version of the datasheet from the website.
Some users have experienced problems with the timing of eeprom writes, and this document provides more specific details about programming.

This does not explain how to use BL233 to read and write I2C and SPI serial EEProms!

2 Programming

Any byte in the EEProm can be written. Writes to each byte take 10ms. During this period, the BL233B cannot process characters and will simply lose any that are sent to it. This can easily corrupt the programming sequence. Each byte is written immediately the second hex data character is received.
Most EEProm settings such as baud rate, only have an effect at power on or reset
The EEWr_Protect bit in fSerial will prevent any further eeprom writes, unless "special pins mode" is used.

Timing Problem Symptoms

If you can program a single byte interactively from the keyboard, but multi-byte sequences are unreliable, this is likely a timing problem.

2.1 Using Character Delay and a Programming File

The easiest (and recommended) approach is just to pad every character sent with a 10ms delay. This guarantees no problems. Handshaking does not need to be working, and you can write the whole eeprom array at once if you want, without worrying about the buffer size.

- Put all the eeprom programming commands into a text file
- Send from Realterms “Send” tab.
- You can set the delays from the command line, and even send the file directly from the commandline.

```
Realterm chardelay=10 flow=2 sendfile=eedata.txt
```

2.2 Using Pause “:” command

The pause “:” command makes the BL233B wait until it receives a linefeed before actioning the characters in the buffer. Precede the programming sequence with colon, “:” . Programming does not begin until a linefeed is received.

```
:V 60D0 454647484950<LF> <60ms delay before next char>
```

- Programming sequence must fit in the buffer (Buffer is 48 chars, so 16 bytes max per sequence)
- After the linefeed, you must not send any characters for 10ms per byte programmed

This is ideal for programming interactively from the keyboard or “send string” in realterm. For porgramming in production the chardelay method is preferred.

3 Dumping and Verifying

The U command dumps the *whole* EEProm as hex. Dumps start at F7 (fserial), not at address 0.

3.1 Programming and Verifying with Realterm

If the programming file has a U command at the end then realterm can capture the dumped data. Realterm can do both automatically.

```
realterm chardelay=10 flow=2 sendfile=eedata.txt capsecs= 2 capquit=eedump.txt
```

Now eedump.txt has the whole eeprom data. By comparing this file with one captured from a known good unit you can verify. If you try to use the dos command FC (file compare) you will discover that it is useless in batch files as it does not return a result code. Get cmp.exe in the “unxutils” package instead.

```
cmp.exe -c eedump.txt eedump_ok.txt
```

4 Special Pins Mode

Special Pins Mode may be able to rescue you from:

- Baud Rate set to something you can't talk to
- Start-up macro has and endless loop or crash preventing BL233 from ever reaching command mode.
- EEWr_Protect bit of fSerial has been set, preventing any further writes of eeprom.

In Special Pins Mode the baud rate and fSerial are in their factory state (57,600bd), and the start-up macro does not run. You will not get the hello message, as this is the startup macro.

4.1 Entering Special Pins Mode

To enter SPM, force all of pins 1,2,17,18, and 6 to 0V, during and 500ms after reset / power on. These pins should be pulled up if unused, to prevent accidental entry to SPM. For surface mount devices you may like to ensure that there is some way to access these pins if SPM is needed, eg pcb test points.
These pins are SDA1, SCL1, SDA2, SCL2, IRQ on the I2C2PC adaptor.

For BL233A,B it is only necessary to pull either Pin 6 to 0V or All of Pins1,2,17,18 to 0. Future versions will behave as described above.

5 Restoring the EEProm to Factory

The file"bl233b_factory_eeprom_settings.dat" contains the data and commands to rewrite the whole eeprom to the factory state, then dump the eeprom. As described above, you may need to be in "special pins mode". The following command will send the reprogram the bl233b, and save a dump of the eeprom afterwards. (all one one line)

```
"c:\program files\realterm\realterm" chardelay=10 flow=2 sendfile=bl233b_factory_eeprom_settings.dat
capsecs=2 capture=bl233b_eeprom_dump.dat
```

"bl233b_factory_eeprom_settings.dat" contains:

```
V F708 210F660E08000D726E5434383439323034393332343332303736333133323331304 13C5432313C4E
4A3030303053353431323334504A3030303153353431323334505432313C5335343132333454343 8
3439575432313C54FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF54
36393C5437373C3C
U
```

6 BL233 Versions

At the time of writing this applies to all available versions. Future versions with welcome string greater than V118 eg"HI I2C v121", receive characters during eeprom byte programming. However the procedure described here will always work correctly with all parts, regardless of handshaking and buffer size, so it is to be recommended.