The infamous EUSART bug had been found on a PIC18F4550 during the development of the MBHP_USB_PIC module. Unfortunately it turned out, that many PICs (such as the PIC18F4620) were affected by this bug as well.

For detailed infos, how the bug has been found, please read the detailed report at the MBHP_USB_PIC page.

Since there was no adequate software workaround available for this silicon bug, it made the old chip revisions nearly useless for MIDI applications: zero bytes were sporadically inserted into the MIDI Out stream, so that the MIDI protocol was violated. As a hardware workaround for this issue, a MBHP_IIC_MIDI module was used with minimal configuration (MIDI Out only).

Meanwhile the bug has been fixed by Microchip. All newer chip revisions (produced 2007 and later) are not affected anymore! 🙂

There is a small application available at the MIOS Download page which allows you to determine the revision ID (search for “revision_id”) This ID will usually also be displayed by PIC programmers (e.g. P18)

At least these chip revisions are affected by the bug:

- 0C 03 (PIC18F4620 A3)
- 0C 04 (PIC18F4620 A4)
- 12 02 (PIC18F4550 A3)
- 12 02 (PIC18F4550 A3)

Please note, that this list is not complete, it only contains devices which have been tested by TK.

Chip Revisions not affected by the bug anymore:

- 0C 06 (PIC18F4620 B4) and later
- 12 05 (PIC18F4550 B5) and later

Some obsolete Informations for PIC18F4620 A3 and A4

Do I really need to build a MBHP_IIC_MIDI module as workaround for the EUSART bug?

If you are using a PIC18F4620 B4 or later: no!

PIC18F4620 A3 and A4: it depends on the application. The EUSART based MIDI Out port of the core module will mostly work, you are even able to upload code without failures. The error (inserted 0x00 bytes which violate the MIDI protocol) only happens very rarely under special circumstances. It can only happen, when MIDI data is also received at the MIDI In port at the same time at the wrong moment (in simple words, details are documented in the errata sheet). As a result, you could notice a hanging note, or a wrong controller once or two times a hour when you are using the MIDIbox.

Accordingly, so long the MIDIbox application only receives MIDI data, or so long it only sends MIDI data while nothing is received, the bug will never appear, and therefore a MBHP_IIC_MIDI module is not required in this case.

MIDIbox SEQ: when used as standalone sequencer, no external MIDI keyboard or MIDI clock source connected to the MIDI In, a MBHP_IIC_MIDI module is not required. In all other configurations one or up to 4 MBHP_IIC_MIDI modules are recommented. Note that MIDIbox SEQ allows to output MIDI events on different MBHP_IIC_MIDI ports, this is nice for reducing the MIDI latency

Do I need to built a complete MBHP_IIC_MIDI module with MIDI In and Out port?

If you are using a PIC18F4620 B4 or later: no!

PIC18F4620 A3 and A4: no, you only need the reduced version with MIDI Out port only. The MIDI In port is *not* natively supported by most applications, currently it is only used by the MIDI router project. Adding support for multiple MIDI Ins is not so trivial, since it affects the performance of the application - therefore it cannot be expected that it will be supported by many applications in future, only by special ones.

Info for Newbies: if you are unsure, how a “MIDI Out only” version looks like when you are using the MBHP_IIC_MIDI PCB, just stuff all components, and left out the optocoupler and the MIDI In LED.

How do I upload MIOS and application code via the MBHP_IIC_MIDI interface?

It is assumed, that you bought a PIC18F4620 2006 or earlier (if you bought it from SmashTV 2007 or later, your device is not affected)

Proposal for Beginners/Electronic Newbies:

- build the core module and test the MIDI In/Out of the board
- upload the PIC18F4620 version of MIOS via the MIDI Out port of the Core module
- build a MBHP_IIC_MIDI module and test it like explained at the MBHP_IIC_MIDI page
- select the ID 10 (both jumpers stuffed)
- upload the iic_midi_10.hex binary of the change_id package, this will change the PIC ID to 0000000000100000
- now the MIDI out stream is redirected to the MIDI Out port of the MBHP_IIC_MIDI module, you don't need to use the MIDI Out port of the core module anymore (even for code uploads it's not required anymore)

Note that it is possible to change the PIC ID, or to upload MIOS without the MIDI Out Port, when the “no feedback from core” option is selected in MIOS Studio. This method is not recommented, as erroneous uploads won't be notified, but it helps in “emergency cases”, e.g. when you wrongly uploaded the main.hex file of the change_id application and you are not able to open the case of your MIDIbox quickly in order to get access to the core based MIDI Out port.

Can I use one or more BankSticks together with the MBHP_IIC_MIDI module(s)?

Yes, on an IIC bus all SCL/SDA (clock/data) lines are connected together, each device must have an unique address, so that the IIC Master (the core module) can perform read/write transactions on a specific IIC Slave. On the MBHP_IIC_MIDI module the address 10/12/14/16 (hexadecimal) is selected via jumpers, on IIC EEPROMs (BankSticks) the address 50/52/54/56/58/5A/5C/5E (hexadecimal) is selected with the Chip Select inputs Pin 1, 2 and 3. As you can see, there is no address conflict between MBHP_IIC_MIDI modules and BankSticks