Fix Me!
→ Please add to this page any fixes to problems you have had.
→ Add links to all the troubleshooting procedures on ucapps.de and eventuaqlly convert them all to wiki format

Reminder:
→ **Have you read the FAQ?**

Golden Rules of Troubleshooting

Hi, I'm stryd_one. I'm a professional IT engineer who fixes bugs for a living, and I've been hanging around here long time, and seen a lot of bugs come and go.... Here are some tips which would solve 99% of the problems I've seen. I hope that you'll pay attention to these rules, and that they'll help you too!

1 Patience

Take your time. If you need it finished by a certain deadline, prepare to fail, and move on accordingly. Rushing will only make things worse. Assume nothing. Test everything.
Don't do a half-a job, pay attention to the details and take pride in your work. Do it properly :)

2 One thing at a time

Don't go changing a whole bunch of stuff inbetween tests. If you are lucky enough to fix it doing that, you won't know what the cause was. “So what? It's fixed!” I hear you say...well...maybe it is, maybe you just moved the problem somewhere else, maybe the problem will return... you don't really know! Similarly, don't try to fix two completely separate matters as though they are one. If your LCD and buttons do not work, then first fix the LCD, then the buttons. If you try to fix both you will just confuse matters.

3 RTFM. Again. Search. Again

Seriously. You should read the entire ucapps.de site, and search both the forum and wiki, several times, with different search parameters if required. When you're done, do it again, in case you missed something. It's VERY likely that you will find the answer you are searching for - IF YOU TRY!
Maybe not, but if you find something useful - which you will - put it in your post when you ask for help. It might give a hint to people trying to help you, and after all, if you save them time searching, they have more time to help you :)
When you go searching, you will almost certainly find test applications and troubleshooting guides. Use them, and your goal will become a lot closer.

4 Everyone wants to help when you are respectful

This forum is mega friendly, probably the friendliest place I've seen in all my 15+ years online. If you're friendly too, then you'll see that for yourself. If you're rude/disrespectful/selfish/lazy/don't follow rules and advice, etc, you'll find that I can be very unpleasant when I want to be ;) and will probably find that noone will want to help you. You probably won't be punished for your disrespect, but your busted midibox will do that for you ;)

5 Give full, clear explanation, and copy/paste error logs

I don't think you could give too much information. No, really. The more info, the better. Information overload, is better than information black-hole. I'll just say it again, to get the message across: COPY and PASTE errors and logs and such. COPY and PASTE!!! Just typing what you see, is not enough. It's very important to provide a clear description of the entire situation. Why? Because...

6 Defining the problem = finding the fix

Fixing a problem is easy! The hard part, is figuring out what the problem is. “My LED won't turn on” is not a problem, it's a symptom. Maybe the problem is that you forgot to switch it on? maybe it's in backwards? The fixes to these problems are obvious (switch it on, turn the LED around), because the problem is well defined. This is the aim of the game - define the problem. Don't worry about how to fix it just yet, just find the exact root cause of your worries. If the fix is not obvious at that stage, the forum will help you.

7 Workaround != fix

Don't get these two confused. Workarounds usually come back to bite you in the @$& when you least expect it. Requiring a workaround, is evidence that you didn't define the problem. Because of this, you may be able to fix a symptom, but not the problem itself.

8 Share the answer!

Please share the answer when you find it! It's a good way to give back to the community, and might one day save someone a lot of trouble!

Thanks, and GOOD LUCK!

stryd_one

Control Surface Troubleshooting

Please see the page control_surface_troubleshooting

LCD Troubleshooting

Please see the page Troubleshooting LCD Displays

MIDI Troubleshooting

Please see the page midi_troubleshooting

After power-on, my MIDIbox sends a lot of random MIDI events, how can I avoid this?

Basically there are two reasons which can cause random MIDI events:

* the pull-up resistor R9 is not mounted (see schematic of Core module) and no DIN module is connected to port J9, so that the serial input pin is floating and therefore propagates random ones and zeroes (→ random button events).

Solution: don't forget to mount this resistor. If a DINX4 module is connected, you should also ensure that all pull-ups (10k resistors) of this module are soldered properly.

* you've uploaded an application which uses the analog inputs (like [MIDIbox64]) and some analog pins are open (not connected to pots, faders, AINX4 module or other voltage sources). In this case MIOS will permanently report random values to the application which will mostly cause an immense amount of MIDI events (since MIOS is so fast )

Solution: connect all open analog pins to ground. Not only the unused pins at J5 of the core module, but also all unused analog inputs of the AINX4 module! If this still doesn't help, then continue at how_can_i_avoid_pot_flickering_jitter

AIN/Pots Troubleshooting

How can I avoid pot flickering/jitter?

Most of the MIDIbox users don't experience pot flickering, whatever type of board they have created (PCB or prototyping boards) - following hints are intended for people who notice an continuous MIDI stream or sporadic, unwanted MIDI events coming out of their box.

Before I explain, how to avoid it, I should descripe how the MIDIbox transforms the pot positions into MIDI values. The PIC includes a so called ADC (*A*nalog *D*igital *C*onverter) which allows the software to measure the voltage levels that are delivered by the potentiometers. Since the valency of a MIDI event is 2^7 (0 to 127) and the pot voltage goes from 0V to 5V, the voltage difference between each MIDI value step is 0.039V. That means, that MIDI value “0” is sent on a change from any voltage level to level 0.00V-0.38V, value “1” from 0.039V-0.077V, ... value “127” from 4.96V-5.00V **in theory!** But in practice the inaccuracy of the ADC and the influence of ambient noise which comes from your electrical equipment (Computer, Monitor, Radio, Mobile Phone, ...) could interfere with the sensible pot levels. As a result, the MIDI values will toggle between two numbers sometimes. Another source of interference are temperatural changes.

Note that jittering can be even more an issue if you are using an application which sends MIDI events with a higher resolution, MIOS supports up to 10-bit (1024 steps, 0.0049V difference between every step!)

Your job is to minimize these influences as much as possible.

Use a metall chassis and connect it with ground (“Faraday cage”). If your pots or faders have a ground pin, connect it with the chassis. Avoid long cables to the pots. Cables to Vss (ground) should be wired starlike. To avoid temperature influences, try to minimize the voltage of your power adapter before the 7805 in order to avoid that the 7805 gets too hot. Best results with 6V to 7V *before* the 7805.(NOTE: Most 7805 voltage regulators require AT LEAST 7 to 7.5 volts to operate properly, check the device datasheet to be sure)

The cables between the pots and the 4051 multiplexers should be as short as possible (see also http://www.ucapps.de/mbhp/mbhp_ain_9.jpg), the cables between the multiplexers and the PIC can be longer, but shouldn't exceed 50 cm.

The voltage of high-frequent noise cannot be measured with a common multimeter, but with a scope. You can temporary upload the Jitter measuring application which is available at the MIOS download page. It displays a number which corresponds to the noise. Values between 00 and 08 are ok.

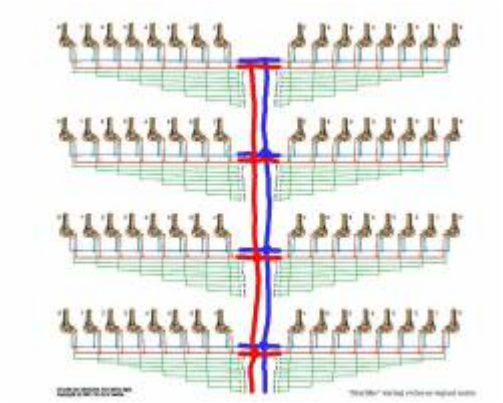
Update: most of these hardware countermeasures are not necessary anymore, nearly all built MIDIboxes are working from the first startup without any jitter problems, since I improved the software on a way that it can handle with jitter.

Starlike-Wiring

Starlike Wiring helps to reduce jitter!

Read the description here:
http://www.midibox.org/users/tor_arne/midibox64_walkthrough/potsbuttons.html

– Click to Enlarge –



Current Drain

If a core module is not working, the current drain could give you a hint if the PIC is running or not. Current has to be measured with an amperemeter (or multimeter in ampere mode) at J1, one supply cable must be removed and the multimeter must be in between the break.

You should measure:

- without PIC: ca. 7.5 mA
- with PIC, bootloader running: 25 mA, a short peak of ca. 30 mA after two seconds (MIOS init phase)
- with PIC, MCLR# reset permanently active (connected with ground): more than 50 mA !
- with PIC, but without crystal: 9 mA
- with “virgin” PIC (not flashed) - the same as without crystal: 9 mA, because the oscillator mode is not configured (the bootloader is not correctly programmed)

From:
https://www.midibox.org/dokuwiki/ - **MIDIbox**

Permanent link:
https://www.midibox.org/dokuwiki/doku.php?id=troubleshooting&rev=1308657731

Last update: **2011/06/21 12:02**