ucapps.de Introduction Wiki

The aim of this article, designed for fast reading is to the number of typical questions throughout the forum which mainly can be answered with one or two hyperlinks. It will mainly aim to help absolute Newbies and check through ucapps.de at last. At least I hope so…

I´ll keep everything as brief as possible, but anyhow, suggestions/edits are always welcome! So, I´ll start there, where all people finding ucapps.de start:

WHAT THE?!?

If you haven't figured it out already, ucapps.de is a DIY page dedicated to the controlling of MIDI (interfaces and devices) and some audio applications (mainly Synths) by certain PIC microcontrollers (the PIC18F452 is mainly used). Throughout the site, you´ll find different devices for certain applications. As I assume you´re a Newbie (;D) I can tell you that the MIOS software (I´ll talk about that later) is totally modular and so you can bring up nearly ANY application you want as long as the used PIC-Microcontroller can bring up the power. BUT that´s NOT AT ALL easy!!! At first you really should stick to the already finished applications and good starters, which are listed below together with their “hardness to build” * = easy, * * * = heavy (e.g. SMD desoldering & soldering) so you know what you can do with all that stuff later on. But at first I want to give an overview of the basic things all MIOS applications have in common:

- Updates of the application can be done via MIDI (also totally new application)
- all applications are able to send MIDI messages (depending on the application)
- Patches/Snapshots and other stuff can be saved using hot-pluggable and cheap so-called “bankstick” (EEPROM)
- A wide range of LCDs can be used
- very modular design! Meaning: You´re able to run many applications on the same hardware, without having to change anything
- cewl community! :D

Now a brief introduction to the list of already-totally-built apps: I´ll only tell the MAX of things which can be connected, you can nearly always go down or leave something out like you want (e.g. it does not matter at all if you use 64 faders + 64 LED´s + 64 buttons or just one potentiometer - the application is nearly the same, just minor changes are needed). I´ll use common MBHP abbreviations, which are also used in the forum:

- pots = potentiometers (faders, rotary potentionmeters)
- encs = rotary encoders, or “endless pots”
- buttons = tact switches or normal switches, no matter
- MB = Midibox
- MBHP = Midibox Hardware Platform
- MIOS = Midibox operating system
- TK = Thorsten Klose, the godfather of ucapps.de, the saviour of all PICs, the light-fast avatar of Assembler land, the…
- Overview of all MBHP Acronyms

Midibox64 *

max 64 pots, 64 LEDs and 64 buttons

This is a powerful tool, but simple enough for beginners. Connect up to 64 pots and configure them to send a variety of MIDI messages! You can control most software and hardware synths with this little thing. ⇒Midibox64

Midibox 64E *

max 64 Encoders, 64 pots *OR* 8 motorfaders, 64 LEDs and 64 buttons

Even a little more modular than the Midibox64. The encoders make it possible to send messages without knowing the original value of the parameter you want to change. That means it´s possible to change it without “jumping” or workarounds like used in MB64 (snap function or similar). But please be aware that this baby is mainly used with Encoders! The pots CAN be used, but the implementation and the flexibility of the messages sent by them is much more comfortable within the Midibox64. ⇒Midibox 64E

MIDIO 128 *

max 128 switch inputs and 128 on/off outputs

This is the choice for people doing projects where everything is on/off, like organ projects. The project was originally done to control a band organ. It is used by people who want to “midify” an organ console, a thing that has a lot of switches–the organ keys, stops, pistons, etc.–that need to be made to generate MIDI output. For this you need a Core, plus one DIN board for every 32 inputs. To control the output of something like organ pipes where you need a lot of on/off signals that are switched by MIDI input, you use a Core plus one DOUT board for every 32 outputs. You can use both DIN and DOUT boards to a maximum of 128 inputs and 128 outputs, a total of 9 standard boards. Beyond that, you can link multiple Cores together, which is often needed for midifying an organ console which can easily have more than 128 input switches. ⇒MIDIO 128

Midibox Seq * *

Defined number of Encoders and buttons (changeable in some ways)

This very cool analog-style sequencer is perfect for programming 4/4 tact patterns and controlling stuff like TB303 clones and drum machines. Patterns can be arranged into songs, and much more stuff is possible (e.g. you can also use it as a “normal” MIDI control device). But do not think that you could arrange complete songs with this! It´s mainly for peaking out 4/4 tact patterns and bending them in some cool sounding ways!

- you can MORPH (!) between different patterns without hard breaks

- it is a very functional arpeggiator

⇒Midibox Seq

Midibox SID * *

Defined number of Encs and buttons (changeable in some ways)

This was a hard decision, giving * * or * * *... There are many forum threads about the MBSID where BIG problems came up. The problem is that you have to know what you´re doing. If you never build some electronical stuff, I really do NOT recommend to start right here! You will have to know about grounding, different voltages, measure where to bring the problems to a point. In 98% of all cases the forum can help, but anyhow it´s a HARD starter´s project. The good point here is also again the modularity: You can start with a pretty easy one-channel SID synthi without to much stuff and end up within a 4 channel SID sound desaster with complete control surface!! And you know what? IT ROCKS!! ⇒Midibox SID

Midibox LC * * *

This is a complete Logic Control clone (!!!) ... (!!!!!!!). Did I yet say: !!! In the meantime you´re able to emulate Logic Control, Mackie Control and the Steinberg Houston Controller. But I won´t go any further into the details as this is TOTALLY NOT a beginners project. If you know UCapps.de well (and you will after your first project) you´ll be able to build it and then you can also inform yourself. ;D ⇒Midibox LC

Midibox FM * * *

This synth uses the famous YMF262 (also known as OPL3) for FM synthesizing sounds. It goes beyond typical FM synths by offering wavetables and control of analog filters. Those FM screaming things, plus a bunch of sounds you've probably never heard, are within the reach of this powerful instrument. But unfortunately it is not a beginners project, as you´ll have to find one of several PC sound cards, then desolder and solder a YMF262 chip which only comes in a SMD (surface mount) package. As this is kinda hard for a beginner and it's easy to ruin the chip (and the motivation) I´ll not go any further. Just listen to the demo sounds anyhow! Sweeet stuff! ⇒Midibox FM

other (smaller) projects *

Beside all those monsters of MIDI there are also some very small projects which can be built and are all pretty easy:

- Midibox Mon: A simple MIDI monitor only using a fairly simple hardware setup. See what you´re doing directly.
- Midibox CV: 8 CV outputs and 8 gate outputs can be used here for controlling older synthies which are working with so called Control voltages (=CV). A kinda special application, but very

handy if you´ve got such a baby. The building itself is not sooooo hard, BUT: Keep in mind that the used IC´s are VERY expensive! Appr. 30 Euros for one chip and two are needed! UPDATE! Sorry, but forget the stated prices. In the meantime a LC=low cost solution for the AOUT is available, see later on in the description of the modules. This means, a MBCV can be build expensive & exact or cheap & a little less exact in the meantime.
- Midimerger: Very very easy application for combining two MIDI Ins to one MIDI Out. Please notice: Chances are high that this will be outdated in some time as TK´s over a very complex MIDI architecture baby including interface/merger/splitter and more capabilities!
- Midiprocessor/filter: Also two very easy applications which are used to filter out and/or process (meaning e.g. transposing, echoing ...) MIDI messages. Normally also needed in special applications, but sometimes also for some fun stuff (remember the typical MIDI echo devices?)

older projects

These projects are discontinued and if you have problems with them (and you´re very likely) nearly only TK will be able to help you out. ALL apps have their (better) subsidues in the newer applications, so if possible, please don´t use them anymore.

So far for that one, let´s go to the next step

CEWL! What do I need?!?

The most upcoming question in the forum, I think. And a pretty easy answer, too: See in the application docu. ;D Anyhow: Like already said (three times?) MB´s are very modular so there are some things you always need and some things you´ll eventually need. See this little “MB Hardware Platform” on the left of ucapps.de? Start there! You´ll (nearly) always need:

- **Core** The heart and the brain
- **LCD** And even if you aren't planning to use it in the end, it´s very practical also in the building stage! You really should have at least one LCD at home for using it as testing device (e.g. a cheap 2×16 LCD is commonly used) More:LCDs

And now for the different applications: There is **always** an exact number of things which can *maximally* (less are always possible!) be connected to a certain hardware piece. This WILL not differ! So if you want e.g. more pots, get more AINs. This list should clear up a little bit:

- **AIN** Pots and faders are connected here, one AIN can handle up to 32 inputs. So normally the max are 2 AINs.
- **DIN** Buttons,switches and Encs are connected here. One DIN can handle up to 32 inputs, too. BUT: One button needs one input, one Enc need **two** inputs! So on a single DIN you can bring in 32 buttons OR 16 Encs OR 8 Encs and 16 buttons
- **DOUT** LEDs (or other on/off outputs) are connected here in various forms. Also here 32 outputs are possible. So 32 LEDs can *directly* be connected. By multiplexing (that means building up a matrix of LEDs and so lower the pins needed in total) even more LEDs can be connected, but this is application-dependend and is stated in the application documentation. *Normally* you´ll

connect the LEDs directly. You can hookup anything that can be controlled by an on/off signal. The newest DOUT boards from SmashTV allow a Darlington driver chip to be added to the outputs to drive higher current devices such as relays, to drive really high currents and voltages.

- **Bankstick** This is sooooo cool! TK thought up the cheapest hot-plugable patch saver ever! This is not really a ucapps.de module, it´s only a single IC connected to the core. This way the interconnection is very easy and patches/prgramms or whatever can be saved and exchanged in a very comfortable way. Normally EVERY application has at least one, although ALL applications also run without it. But I really recommend at least one Bankstick for every application

- **LTC** This "only" expands the MIDI Interface already installed on the core. So you actually don´t need it, but it´s optional for *all* apps. It will give you a MIDI In Status LED, a MIDI Out Status LED, one MIDI THRU port (that means, the MIDI In of the core is copied and given out here directly) and a second MIDI Out (yupp, copied directly from the Core MIDI Out). Besides that you *can* use it for directly connecting the system not over MIDI, but over a RS-232 port (better known as serial port on PC´s). BUT: **Either** you use MIDI **or** the RS-232 option, you can **not** use both at the same time! You´ll bring in some shorts if you try to! The RS-232 option is very seldom used as it´s not as easy as MIDI (plug&play) to get implemented in your system.

- **USB modules (both)** This is under heavy reconstruction by TK, and NOT a beginners project in any means! You will have to solder SMD´s, programm 24LC EEPROMS and more, so think about getting a commercial 2In/2Out USB-MIDI interface instead! Those don´t cost much more than the components of these modules and so the USB modules are more something for DIY purists, I think... although I have two of them ... *eeeeeerrr*....:P ;D

- **MF** I´ll not explain much here, as this module is *nearly* only used in the MBLC. You *can* use it also in other apps, but you can get the most use out of it in a MBLC. As the MBLC is, like already said, NO beginners project AT ALL, there´s no need for explanation. Anyhow: It´s used for controlling (in the term of actually moving) motorfaders.

- **JDM** This was the old version of a PIC microcontroller programmer. Unfortunately the stability and the compability of this module was not very high, so it will be substituted by the following module. My recommendation: Just forget about this one here. There´ll be no support or anything for this anymore, you know. ;D

- **PIC Burner** You can programm the PIC microcontrollers with that... **SURPRISE**... ;D. Please think of the following: When using the PIC´s with MIOS (this is the software used throughout MBHP and ucapps.de) you will have to actually "burn" your PIC´s *exactly* ONCE in their lifetime. The rest *will* be done via the MIDI line! So it´s only worth building this here if you plan to make let´s say five or more projects. Otherwise you can get already preprogrammed (preburned) PICs at e.g. SmashTV´s shop or also @mike´s shop, which is worth it up to *approximately* five projects (taking the problems and the time with burning into account).

- **SID** This is the speaking soul of the MBSID. Or more... like screaming. Just listen to the demos! Then you know, what I mean. Here the messages coming from the core module will be translated into sounds and given out. Sounds easy, sounds loud.

- **OPL3** *Exactly* the same as for the SID: The messages from the core module are translated into real sounds here (through the OPL3 IC). That´s all it does, again much to easy explanation for such a nice little baby. So: Just DO listen to the demos again. :)

- **AOUT** This is the first version of the available AOUT = analogue out modules. It offers up to 8

CV (=control voltage) outputs and 2 gate outputs. If you do not know what a CV is, you really won´t need it, believe me, as this is used for analogue synths and/or older stuff. This module right here is the very exact version with a resolution of 12 bit, BUT it´s kinda expensive, as the DAC IC´s are kinda good ones. So the price for one module will reach approximately 70 Euros (!). But also to mention: This here is compatible to all applications offering AOUTs, the low cost version (following) is NOT. For details just look up both on ucapps.de.

- **AOUT_LC** This is the low-cost version of the module above. But in some threads and topics you can read about specific problems with this one. I would not recommend it, as it´s not as exact as needed for good applications and it is not supported by all MIOS applications which offer analogue outs. For more details pls look it up on ucapps.de. Just invest a little more and get the right (upper) stuff if you plan doing something with CV´s and gates!

Ok, I have everything, what now?

This will be more about ALL the stuff coming up after your first encounters with ucapps.de. I will **not** go into details, I´ll just give an overview. For *every* section here there are very detailed Howto´s, FAQ´s and Troubleshooting guides. I´ll just summarize here everything

Powering up

Depending what you are building, a normal wall adapter (let´s say 9 Volts and 500 mA max) will probably suffice, but there are some great pointers here:

Parts FAQ: What Power Supply Should I Use?

Programming the application

Now for the programming routine in the order like you´ll have to do it:

- Burning the PIC: Here, the bootloader gets programmed into a PIC microcontroller using a chip programmer, or “burner” module. Perhaps you´ll want to get a PIC which has already been programmed with the appropiate stuff. If you **really** want to burn yourself, look up here.
- MIOS! Yeah! This is something like your Windows/Linux/whatever on your PC. It´s the backbone of all applications here. After you've assembled your CORE, it can be sent to the PIC via MIDI. This process is explained on these pages
- Editing and bringing your application to the core: You can (optionally) tailor the MIDIbox application to your needs by editing the main.asm (or other files), then again “compiling” it (that means translating it to the “PIC language”). After that you can download it to the PIC via the MIDI line, and it will work with MIOS (hopefully appropiate ;D). Look up these pages for further stuff: MPLAB, MIOS Studio, hex2syx and perhaps also here: mbsyx

So much for programming! So far your project should RUN NOW! CONGRATZ! ;D ;D ;D … … if not:

I'm STUCK!!!

This really really really really really really really really really happened to EVERY ONE here at least once! The good point is that we have a pretty big support section in the meantime. If you don´t know any further I suggest following stuff:

- Surf around in the WIKI. There are many FAQs and Troubleshooting guides which can help you out!
- Look up on ucapps.de again and click through the left bar once. Perhaps there´s something which you haven´t seen up to now
- Look at the various Walkthroughs/blogs and so on. As ALL the projects here are totally modular to a very high level, the same problems come up again and again. Perhaps some other or similar project can bring you outta there.
- If you still haven´t found the right stuff, try **searching** the forum. I can not say that loud enough: Pls pls pls **search** the forum with the famous so-called **search-function**. Really! Reason: In at least 70% of all cases the question you want to ask is already answered 2 - 10 times.
- Yupp, now you´re in the forum, if you STILL got problems, WELCOME to the forum! ;D

Finalize

A little ending words here. I hope I could help a little with the experience I have gained in my last 4 years of ucappsing around. I can tell you that it´s worth the steep learning curve at the beginning, the applications ALL still search for similar solutions in commercial stuff! And the stuff you learn here can be adapted to many other electrical problems, too. So, really, just DO go ahead!

Just a few words more: DIY does not mean “SAVE MUCH MONEY”! It means “Do it yourself”. So do not think you will save a lot of money, as the time-intensive DIY process eats up all money you are saving here for sure. If you do it, do it for yourself as a perfect adaption to your needs and for all those guys telling you “YOU (!) did that?!?”. Do NOT look on the money to much as cheap and cranky stuff, like plastic faders, can bring down the fun with your device dramatically. On the other hands, high-quality stuff can bring in the great “wanna touch” feeling!

I really want to say BIG BIG *THANKS* again to all the great guys here (not that I already did hundred times). This means in the first place TK and again TK TK TK. THNX for this great community, work, apps and more! Moreover I wanna thank all those guys around here helping me and/or ucapps.de out of different situations, especially SmashTV, TwinX, NorthernLightX, moebius, d2k, pilo, StevenC, ScreamingRabbit, strydone, Jidis, illogik, Captain Hastings, raphael, JimHenry and ALL those guys I just forgot in this long list. Just PN me and tell me, or edit the article by yourself. ;D ;D ;D

For the ending: This article should be living. So please if you find any errors/updates please feel free to edit! In other cases just contact me.

So, like always:

Greetz!

From:
https://wiki.midibox.org/ - **MIDIbox**

Permanent link:
https://wiki.midibox.org/doku.php?id=introduction_to_ucapps.de&rev=1205478587

Last update: **2008/03/14 07:19**