

MBlight

The Mind-Machine

DIY description

Kurt 2009

We_R_1@web.de

The **Mind-Machine** generates audio-visual signals that influence the frequency of a person's brain waves, thereby inducing different mental states depending on the chosen pattern. I will refer to this process as **Brain-Lighting**.



Disclaimer

- * I don't take any responsibility if your brain or body is harmed by the use of the Mind Machine.
- * Please don't brain-light if you are prone to epilepsy or a similar condition. Please don't use the Mind Machine together with drugs.
- * Write a diary of your impressions and feelings after each session!

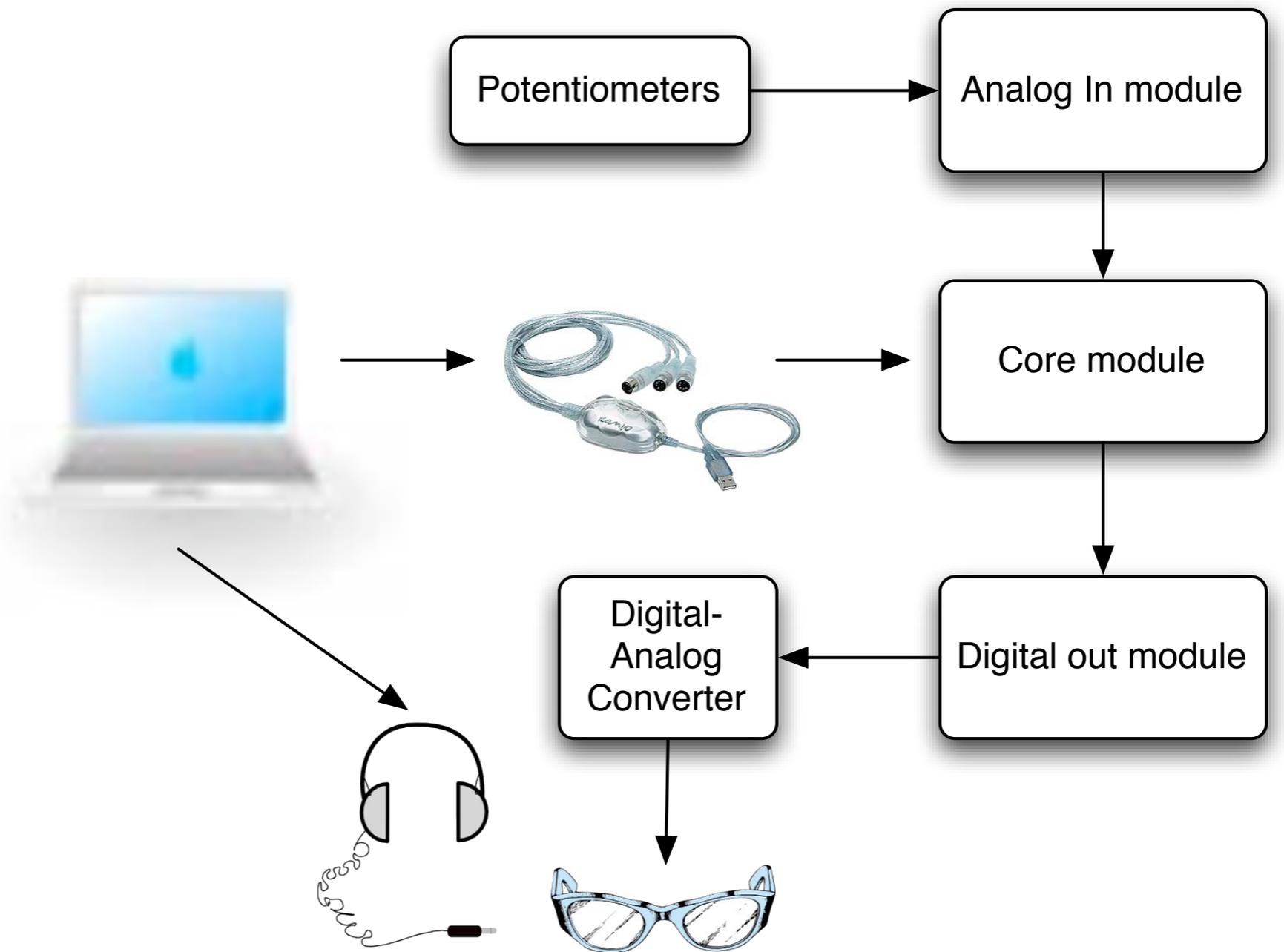
Mind-Machine Hardware

- * The hardware of the Mind-Machine is based on the MidiBox project, which is a collection of non-commercial Do-It-Yourself MIDI projects.
- * Personally, I use the Mind-Machine for non-commercial purposes only, and there is a patent ([WO/2004/022154](#)). There are also commercially available sets that do similar things. Please respect that - I take no responsibility for misuse.
- * This is a hobby project, and I did not invest the time and effort as I do as a scientist - with all the consequences.



uCApps.de

Hardware overview



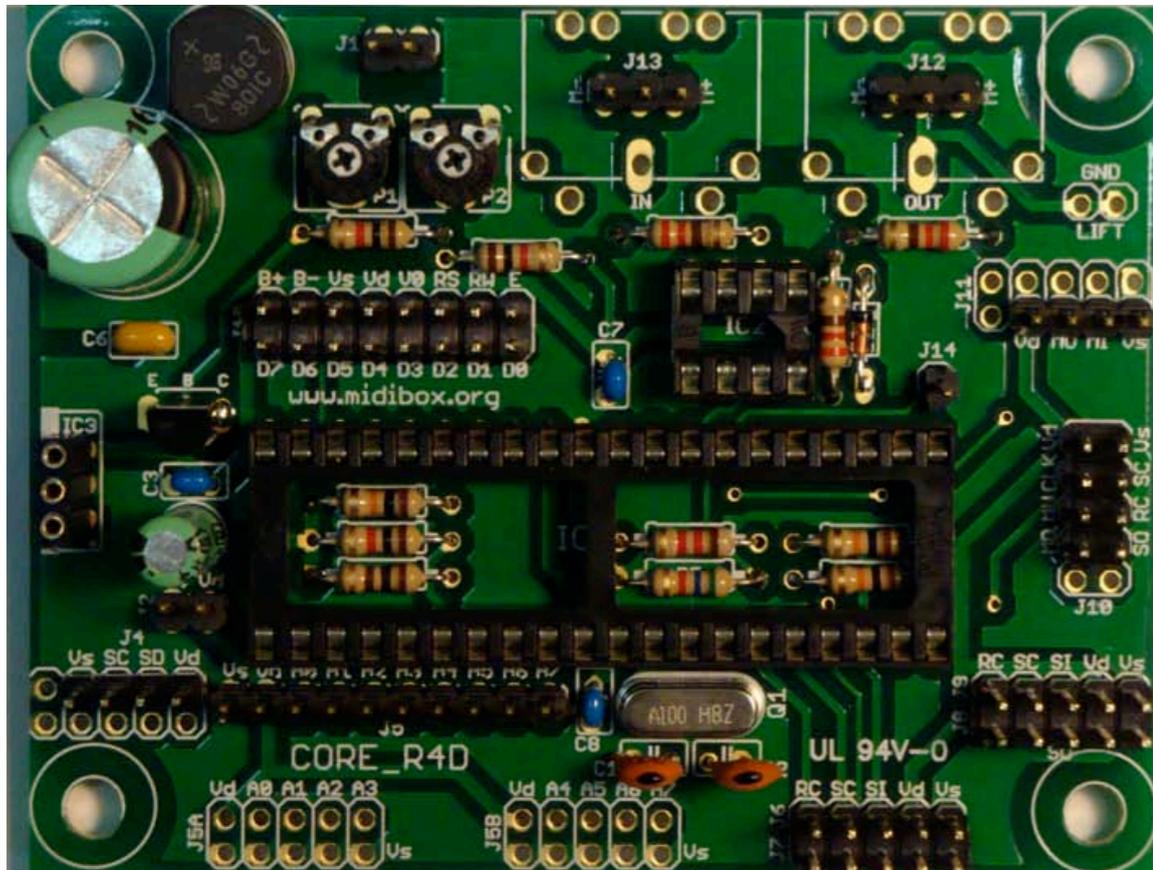
The computer generates the audio signal and also communicates via a MIDI interface with the core module, which translates MIDI signals into a digital representation which are then converted into a variable voltage that drives the LEDs built into the spectacles. Additionally, user input from potentiometers is directed to the computer.

The MidiBox part of the Mind-Machine

Core module

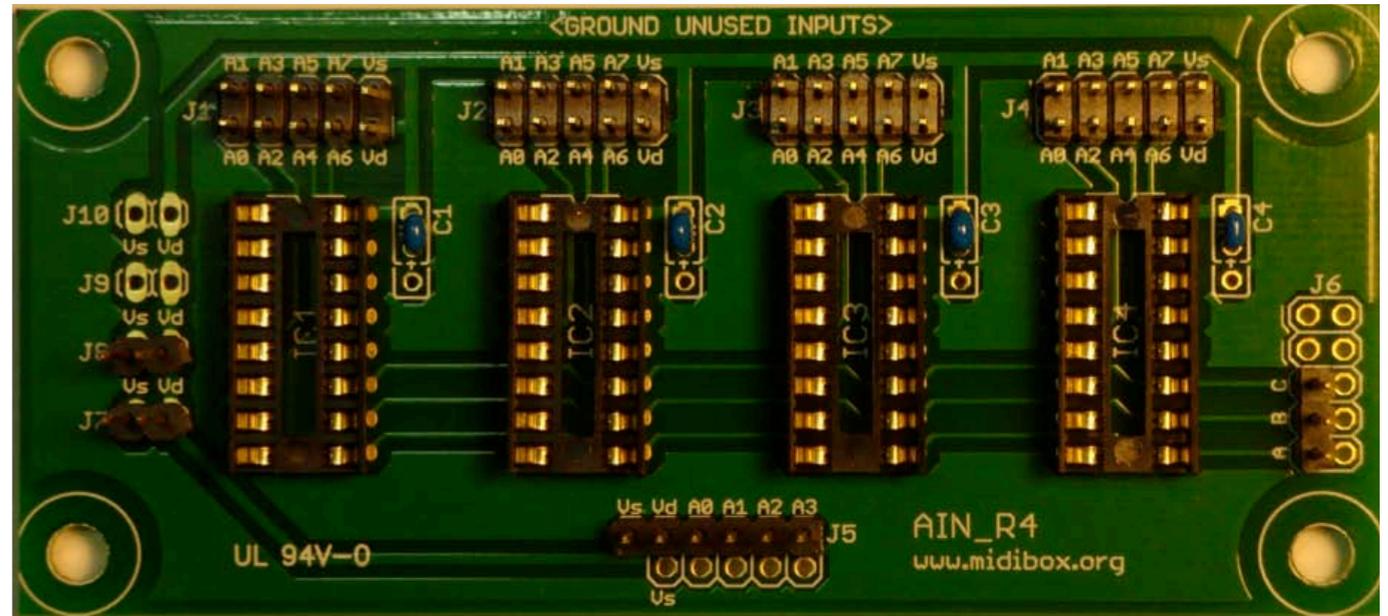
PIC 18F4620, BSL V1.2B

MIOS V1.9F

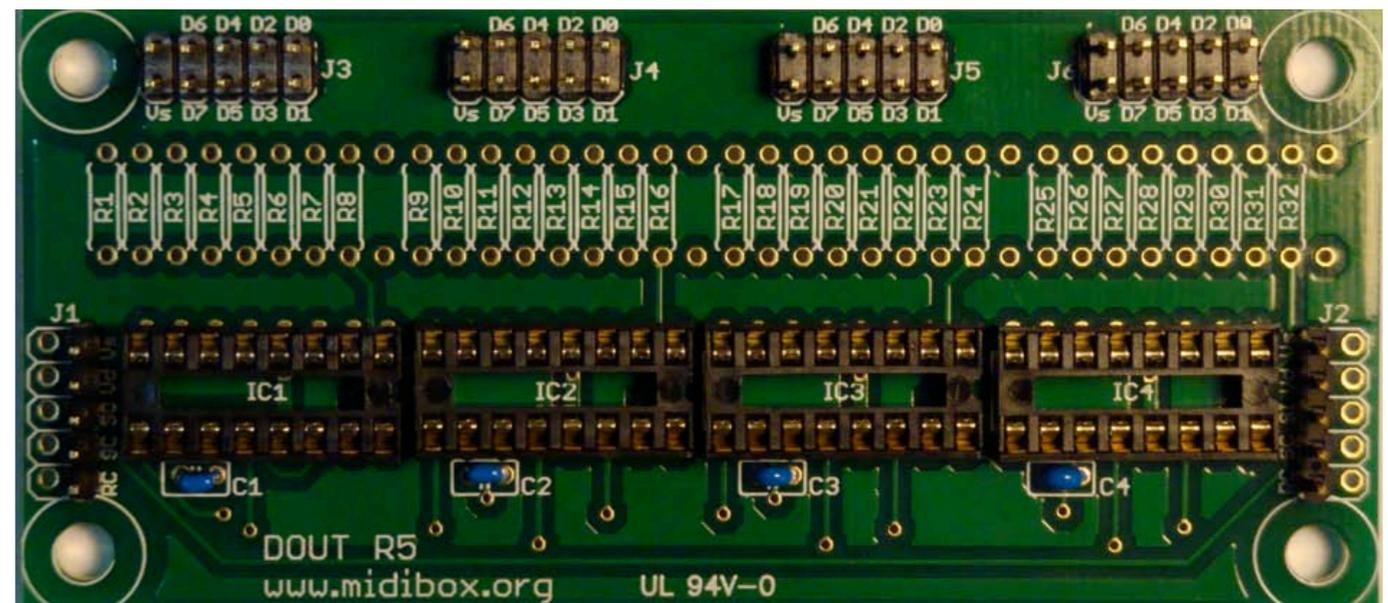


The printed circuit boards and the parts, including the preprogrammed PIC microcontroller were purchased from [AVI Showtech](http://www.avi-showtech.com) for approx. \$70.

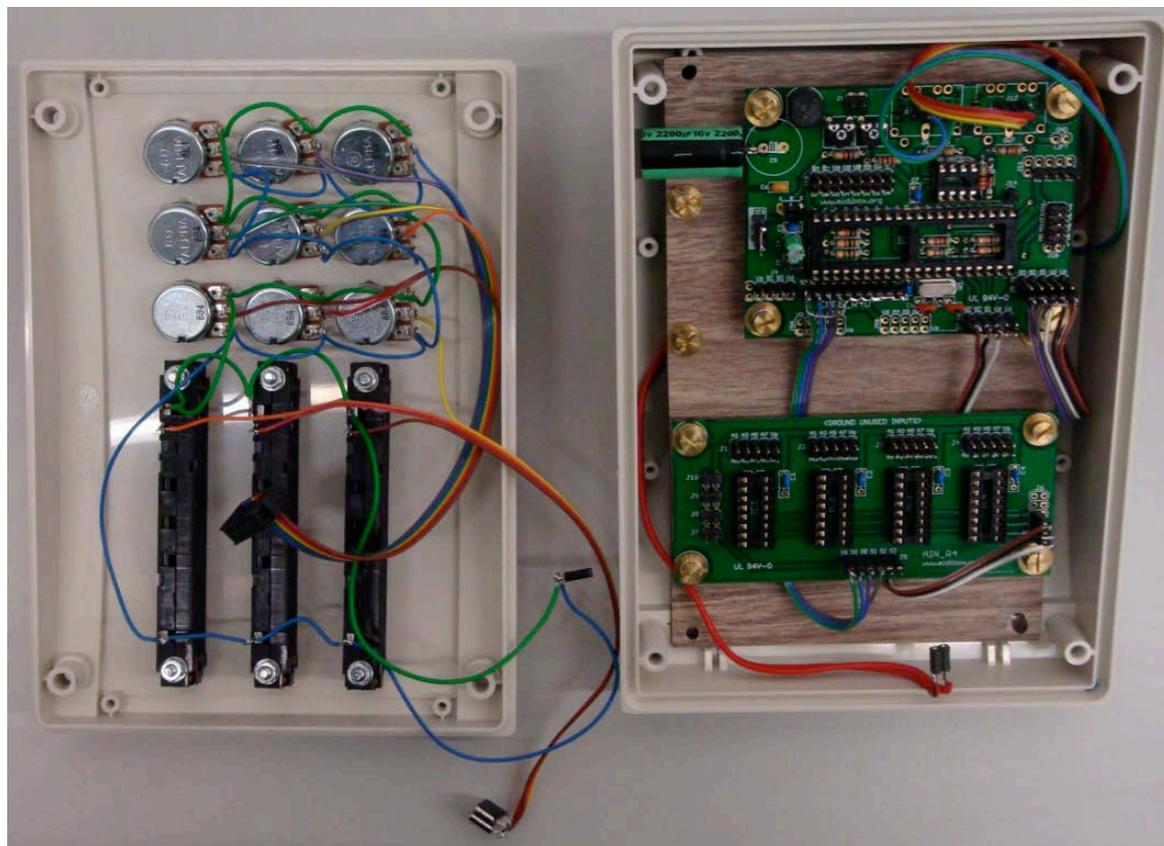
Analog In module



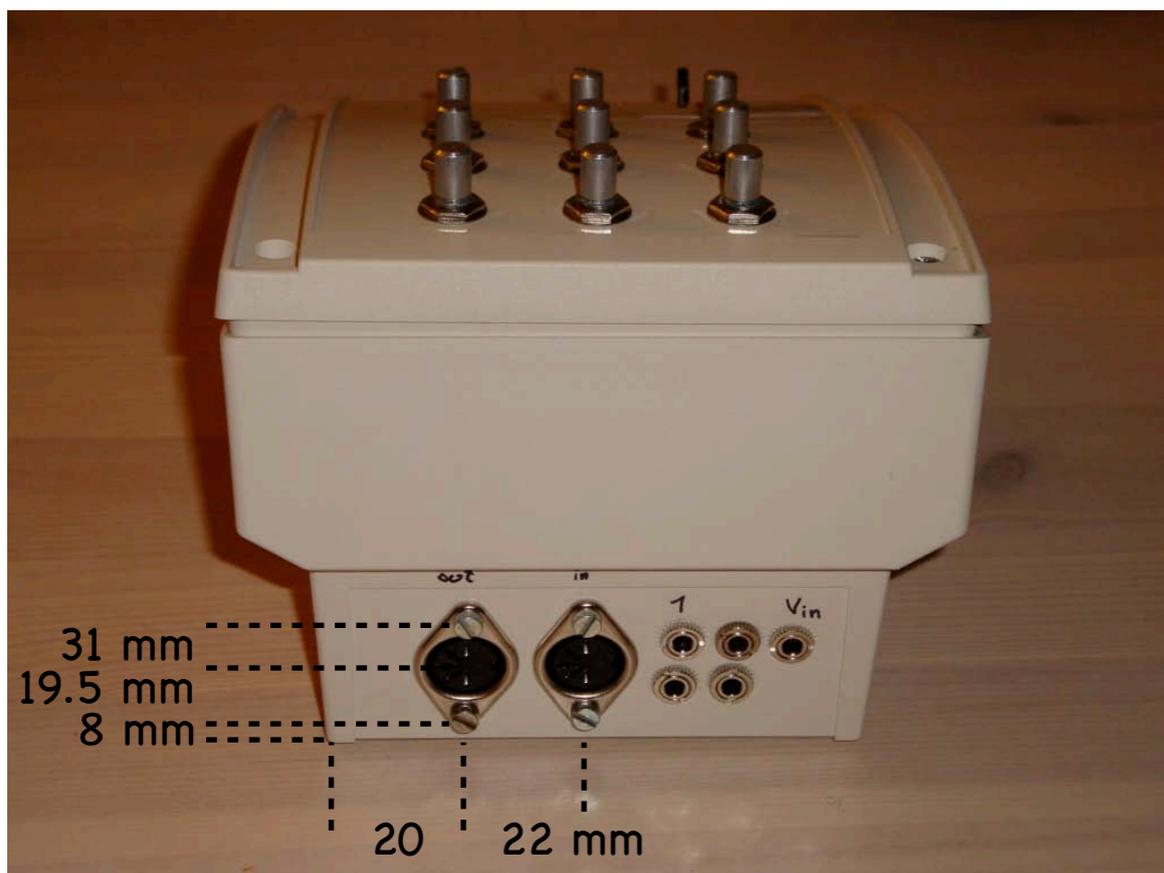
Digital Out module



The Box

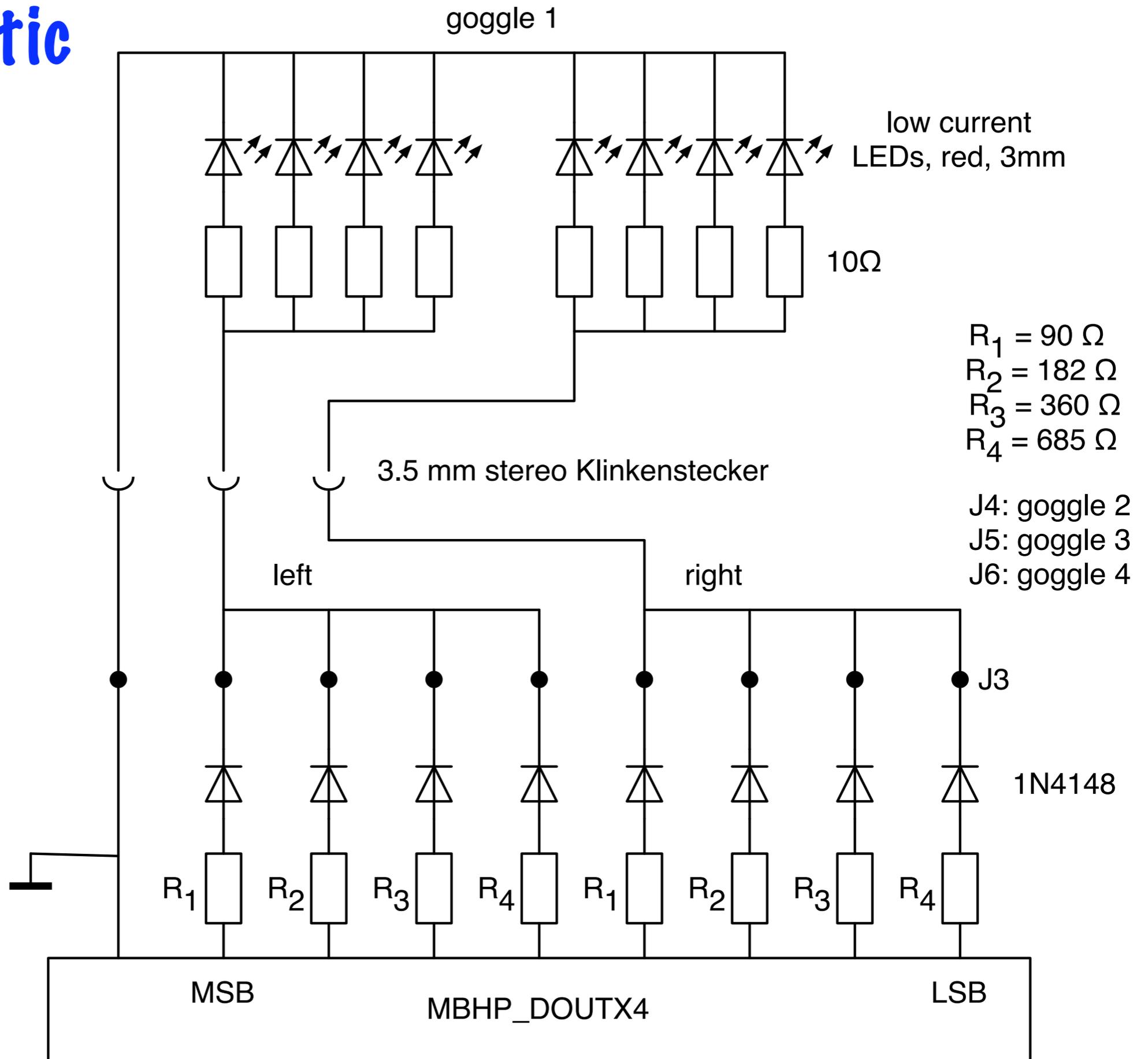


- * The box is from conrad.de
- * The DOUT module is mounted underneath the Core module
- * Up to 4 spectacles can be connected
- * Powered by USB or external power supply
- * The 3+9 potentiometers would not be necessary, but since I am DJ-ing a bit, they come in handy. Also good to have when writing Brain-Light scripts
- * Costs: approx \$100 including small parts



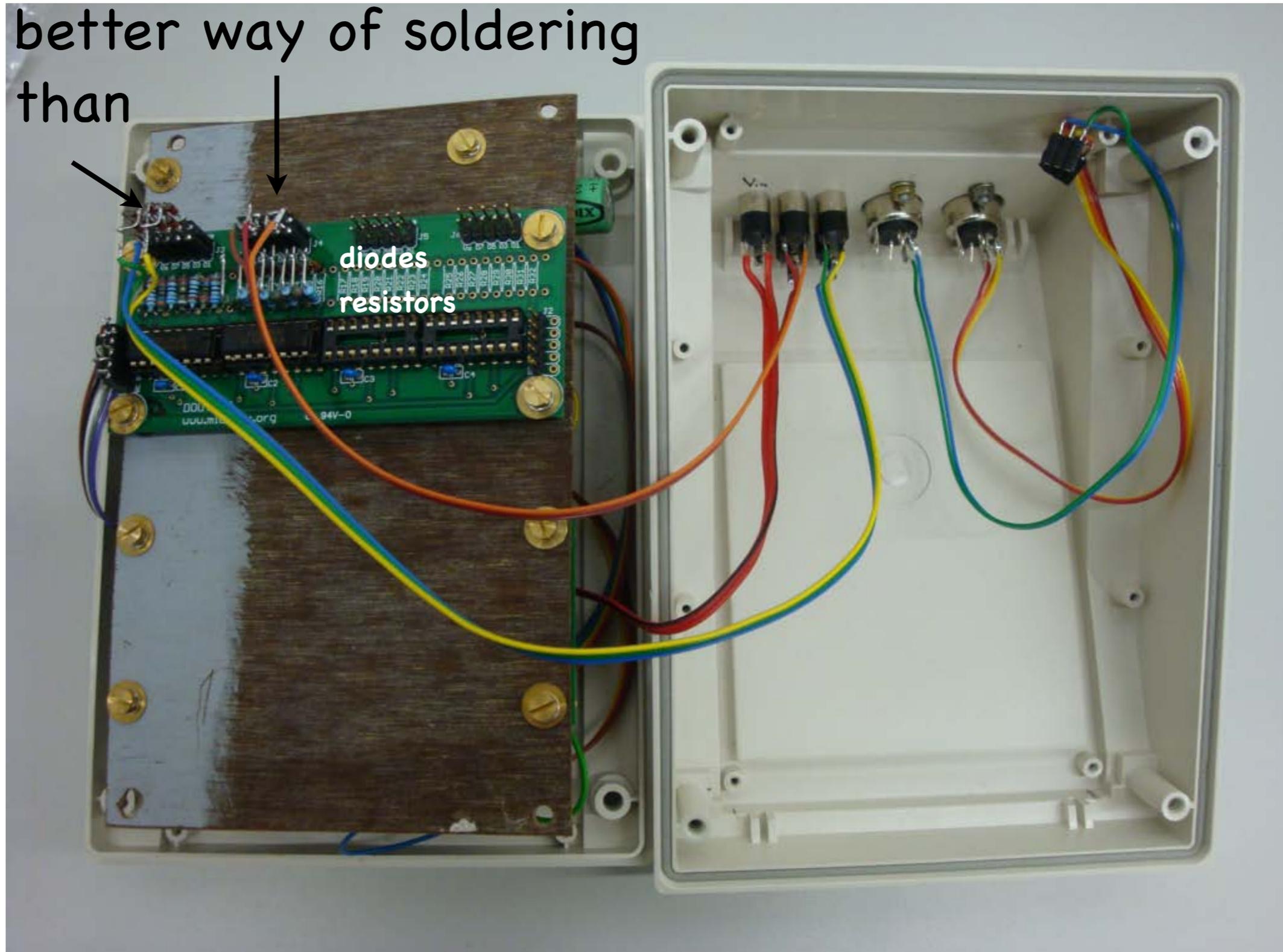
Box and sliders: Conrad [522620](http://conrad.de), LIN 10K: [441449](http://conrad.de); Pots: [AVI Showtech](http://conrad.de)

Schematic



Hardware implementation

better way of soldering
than



LEDs in the Goggles



This is how the LEDs are mounted in the \$1 sun-glasses. To find the position of the LEDs, lie down comfortably, wear the goggles (and ear-phones), close your eyes and look straight. Place the LEDs in direct sight.

Firmware

After a note „c“ or „d“ was received on midi channel 1 or 2, the note and the brightness value (midi velocity) is sent to a function SetBrightness() discussed next.

```
////////////////////////////////////  
// This function is called by MIOS when a complete MIDI event has been received  
////////////////////////////////////  
void MPROC_NotifyReceivedEvt(unsigned char evnt0, unsigned char evnt1, unsigned char  
evnt2) __wparam  
{  
    unsigned char Brightness;  
  
    // divide velocity-value by 8 to map the 0..127 velocity range  
    // to the 0..15 brightness range (4 bit D/A converter)  
    // this is now done in SetBrightness() (is it really done there?)  
    // actually I am not sure if the next command should not be uncommented  
    // Brightness = evnt2 >> 3;  
  
    // check if note on or off event at channel 1 has been received  
    if( evnt0 == 0x80 || evnt0 == 0x90 ) {  
        // if note off event: force evnt2 to 0 for easier handling of 'LED off'  
        if( evnt0 == 0x80 )  
            evnt2 = 0;  
  
        // Set brightness of goggle 1: SetBrightness(Channel, Note, Brightness)  
        // Channel = 1..4 and corresponds to the Goggle number  
        // Note c(0x3c) = right light  
        // Note 0x3d = left light  
        // Brightness = 0..15 ( 4 bit )  
        SetBrightness( 1, evnt1, Brightness );  
    }  
  
    // check if note on or off event at channel 2 has been received  
    if( evnt0 == 0x81 || evnt0 == 0x91 ) {  
        // if note off event: force evnt2 to 0 for easier handling of 'LED off'  
        if( evnt0 == 0x81 )  
            evnt2 = 0;  
  
        // Set brightness of goggle 2  
        SetBrightness( 2, evnt1, Brightness );  
    }  
}
```

Firmware

SetBrightness():

This code translates the note c/d on midi channel 1-4 into a bit-code, which is then set on the DOUT module to drive the left/right LED with the desired brightness.

```
/////////////////////////////////////////////////////////////////
// Set brightness of left/right channel of goggles 1..4, 0 should be all channels
// Channel = 1..4 and corresponds to the Goggle number
// Note c(0x3c) = right light
// Note 0x3d = left light
// Brightness = 0..15 ( 4 bit )
/////////////////////////////////////////////////////////////////
void SetBrightness(unsigned char Channel,
                  unsigned char Note,
                  unsigned char Brightness) __wparam
{
    unsigned char MSB_Pin;
    unsigned char t;

    // react on channel 1..4 ( 4 Goggles )
    if ( (Channel >= 1) && (Channel <= 4) && (Note >= 0x3c) && (Note <= 0x3d) )
    {
        MSB_Pin = (Channel - 1) * 8 + ( (Note == 0x3c) ? 0 : 4 );
        // yields 1, 5, 9...

        // set 4 pins successively
        for (t=0; t<=3; t++)
        {
            // set pin - MSB first
            MIOS_DOUT_PinSet(MSB_Pin + t, (Brightness & ( 8 >> t)) ? 0x01 : 0x00 );
        }
    }
}
}
```

Computer control: Ableton Live

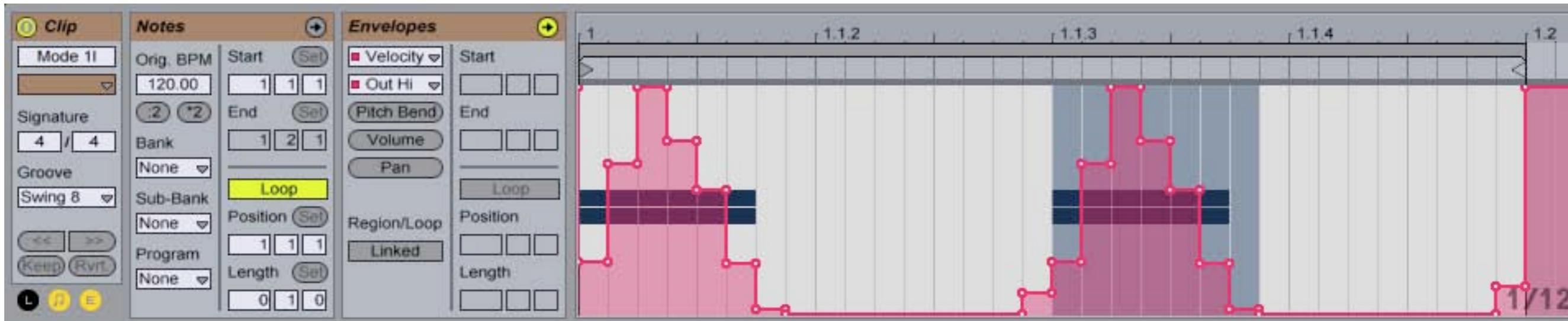
1 Audio	2 Audio	3 light	4 click	5 Goggle1	6 Goggle2
▶ 01 Crises	▶ Lasal Praise	▶ Mode 1l	▶ Mode 1c		
▶ 05 Why Does My	▶ LasalLove Diving				
	▶ Lasal Will Creat	▶ Mode 2l	▶ Mode 2c		
▶ P9 Ambient Elec		▶ Mode 3l	▶ Mode 3c		
		▶ Mode 4l	▶ Mode 4c		
▶ 05 Oracle	▶ LasalWho I Am	▶ Mode 5l	▶ Mode 5c		
▶ 02 Moonlight Sh		▶ Mode 6l criss	▶ Mode 6c cross1		
▶ 03 In High Place					
▶ 04 Foreign Affai		▶ Mode 6l left	▶ Mode 6c circle		
Audio From No Input	Audio From Ext. In	MIDI From No Input	MIDI From No Input	MIDI From 3-light	MIDI From 3-light
	1/2			Track Out	Track Out
Audio To Master	Audio To Master	MIDI To No Output	Audio To Master	MIDI To ESI-ROMIO (Ans	MIDI To ESI-ROMIO (Ans
				Ch. 1	Ch. 2
Sends A B	Sends A B		Sends A B		
1 S	2 S	3 S	4 S	5 S	6 S

Velocity control panel showing Drive, Out Hi, Comp., Out Low, Mode (Clip, Gate, Fixed), Random, and Range (127, Lowest, 1) settings.

- * Tracks 1&2 are for (relaxing) sound
- * Track 3/4 contain clips for different light/click patterns discussed below
- * Midi note c/d is left/right light, velocity (Out Hi) gives the brightness (0..127 is firmware-mapped to 0..15)
- * The MIDI Signal is routed with tracks 5/6 to goggles 1/2 (MIDI channels 1/2)

Control details

The light intensity (velocity) is gradually changed to make the flashing less harsh.



The sound clicks (Simpler Synth) are 1/64 shorter than a 1/16 note which contains 2 blinks/clicks (1 tick with adaptive grid middle). 120 bpm/4 = 30 beats per minute = 1 Hz

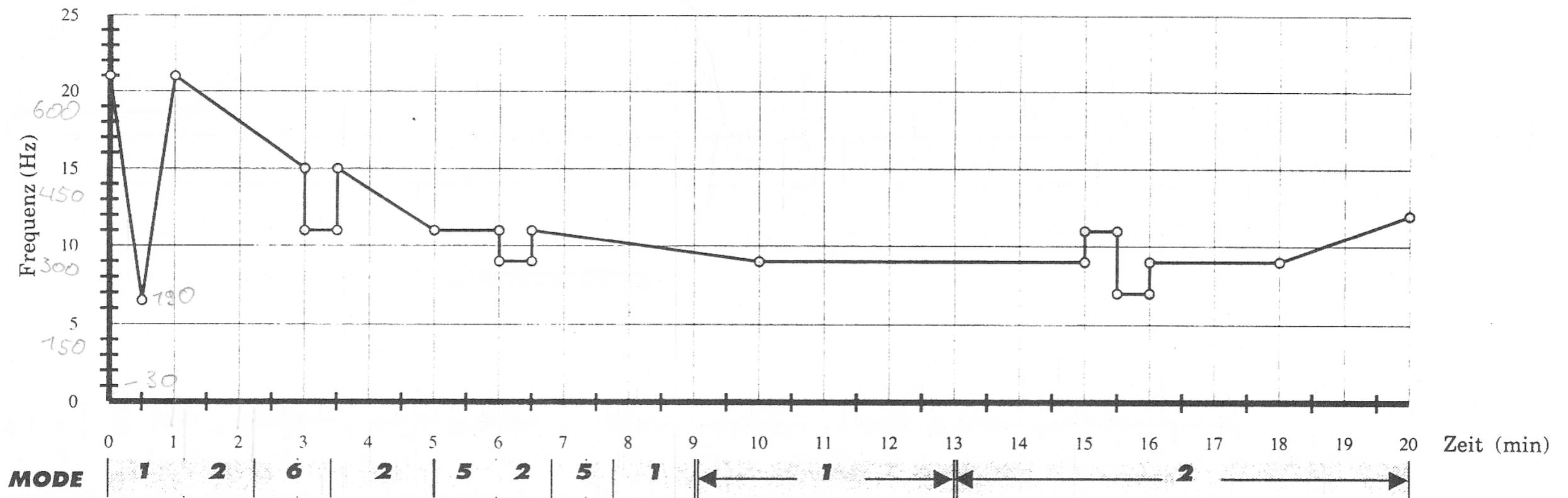


Click/Flash modes

There are a few ways of synchronizing the clicks with the flashing - principally there is always 1/4 note pause between clicking/flashing:

- * All:
ears and eyes are click/flushing at the same time
- * Left-Right:
both click/flash right, then left
- * Slanted up/down:
both eyes, ears changing or both ears and eyes changing
- * Up-down:
first both eyes than both ears
- * Diagonal:
one eye and the other ear
- * Round:
one eye/ear at a time; can be (anti-)clockwise

An example script suitable for beginners



This script helps you to relax and gain fresh energy. It's well suited for beginners, although for the very first session it's a bit long.

Modes: 1..all, 2..left/right, 6..diagonal, 5..both eyes and ears changing
 [source unknown]

Some links

- * [Mind-Machines.de](#)
- * [MindModulations.com](#)
- * [Wikipedia entry german](#)
- * [Wikipedia entry english](#)
- * [A research article on EEG pattern](#)