MIDItriggs

MIDI to/from trigger interface

What is it

MIDItriggs is a simple MIDI to/from trigger interface developed from the sudden desire to add MIDI capabilities to an old Hohner Automatic Rhythm Player. Intended to just trigger the analog sound generators via MIDI in the first place, the project has grown to also provide MIDI output to synchronize MIDI sequencers to the internal sequencer of the machine using a MIDItriggs module. Meanwhile, the other way also works, as the CR-78 adaption makes quite a lot of use from this.
The project is designed as a very versatile module not just for the first unit it is used for, and so the number of usable features may vary with different host machines using this module. More about this can be found later in the technical section.

How to use it

MIDItriggs is designed for simplicity in use and especially in the mechanical work you have to do on your maybe very precious vintage equipment. So no fancy displays or rotary encoders, everything is reduced to just drill some holes for the MIDI connectors and a button.

General description

The module is installed in the host machine and uses a small amount of power from it. The main feature, which usually is used in every installation is the control of the host instrument trigger signals. Depending on the installation and the way the host machine works, it is also possible to read those trigger signals to generate MIDI events from them and if all features are used, the machine clock and start/stop signals are also read to convert them to MIDI signals, while incoming MIDI sync data can be used to control and synchronize the internal sequencer of the host machine.
There is a very simple configuration interface built into the firmware to set things like the MIDI channel, instrument mapping, clock handling and MIDI output features, which is completely controlled by one single button and incoming MIDI note-on events.

MIDI implementation

The MIDI capabilities are very rudimentary, as nothing freaky is needed to control some trigger signals. So is just knows about MIDI note on events coming in, which are mapped to the desired trigger signals, and the MIDI clock/start/stop/continue for some synchronization. The same is supported on the MIDI out, it sends note on/off events to send the trigger signals via MIDI and it sends start/stop/clock messages for synchronization of external equipment.

Trigger synchronization

One MIDI note on event needs at least 0.6 ms to be transmitted, so there is no chance to get rid of latencies when just handling the note on events. So a sort of trigger latch timeout has been implemented which delays firing of the triggers by 2 ms after the reception of the last MIDI note on event. So if in your sequencer, 5 instruments are on the same spot, they still need about 3 ms to be transferred, which leads to sequential triggering of your instruments during this period in a random order. This can sound really evil, so there is a delay, which resets to 2 ms after the first MIDI note on event has been received, and if for those 2 ms no further events are received, the triggers are all switched on at once. The older sync onto the MIDI clock has gone, as it gave too long delays. The consequence is that the new trigger synchronization is now always enabled.

Incoming MIDI start/stop/clock

If you want to synchronize your drum machines internal sequencer, you can easily do it. The hardware driver for your host machine needs to implement the details on how to tell your hardware to start, eventually catch lost clocks to feed them later, and even generate clock signals for a fade-out, while the MIDI clock is already gone. There are sync masters behaving strange (like the Yamaha QY-10, which sends a load of fast clocks after stop) and there are sync slaves needing special handling (like the Roland CR-78). A lot of this is already implemented due to several days of research on how to implement it the best way.
The result is simple and works as desired:
A MIDI start message starts the internal sequencer, but does not yet clock it. The MIDI specs say that after a MIDI start, the first clock is the real start of the playback. It is implemented this way.
A MIDI continue message usually doesn‘t start your pattern at the beginning, but at a specific spot. As this is not possible with the host machines that are currently implemented in MIDItriggs, there is no special handling of this case, continue is handled as start.
MIDI clocks then advance the sequencer, while the machine clock might use a different speed, like the CR-78 using 12 ppqn, while MIDI uses 24 ppqn. So there also is an adaption of this.
After a MIDI stop, no further received clocks are used, as they can be totally wrong. If the machine needs to continue running after a MIDI stop for a while (like the fade-out on the CR-78), the now missing clocks are generated by MIDItriggs. For this, an internal timer is synchronized to the incoming MIDI clocks, providing some sort of PLL to provide the missing clocks. There currently is no rounding, but if there is a jitter of more than 10% in the clock, the timer is not updated and the clock system set to „unlocked“ until 6 clean clocks are received. After start, this lock counter is set to 24 clocks. If a stop is received within those 24 clocks after start or during a jitter period, the sync timer is forced to a locked state, as we need to get clocks from somewhere.

Outgoing MIDI start/stop/clock

If you want to use the sequencer of your host to be the synchronization master of other machines like a sequencer, MIDItriggs can also provide the necessary messages for this if you enable this feature.
As the machine might work with a lower clock rate like 6 ppqn, while MIDI demands 24 ppqn, there is some mechanism to reconstruct the clocks between the machine clocks of the host. This mechanism is using the same timer that is used for MIDI in synchronization, it also uses the locking mechanism to not provide incorrect MIDI clocks due to the sync timer running wild.

This currently is a bit tricky if the host machine is far off the 24 ppqn. For 12 ppqn, both clock signal transitions can be used, while for 6 ppqn, there are 3 of 4 clocks that need to be generated by MIDItriggs.
Besides this, the same messages like MIDI start, stop and clock are transmitted in sync with the machine clock.

Module blocking

It might be necessary to block several states against each other. So on the Hohner Automatic Rhythm Player there is no way to control the internal sequencer. So when the internal sequencer is running, MIDI in is switched off and the configuration menu is aborted, if running. On the CR-78, there appears no sound if the sequencer is not in „running“ state, so the configuration starts the sequencer on it‘s own but doesn‘t provide clocks to it, so no instruments are triggered by the machine.
The in/out locking is done via the state engine, but it‘s perfectly okay to have both running in parallel, due to no sound comes out if the internal sequencer is not in running state.

Configuration menu

The configuration menu uses the single button and MIDI note events as the input devices and gives feedback by triggering instruments. For sure, this is not perfectly intuitive, but as configuration is not the main purpose of MIDItriggs, it should be possible to live with this - on the other side you don‘t need to drill lots of holes into your vintage equipment. As the project is open source, it is no problem to add a more sophisticated configuration system using displays and buttons or encoders if this is desired. In fact, this already exists, but is not (yet) part of the open source version.

To enter the configuration menu, hold the button for at least 1 second. You then hear the kick drum 3 times, a small break, and either the kick or the snare another time. As soon as you hear the first kick drum, you can release the button. This pattern can be simply written as a pattern of 2 measures with quarter notes, running with 120 bpm. Writing the break as „-“, the kick drum as „0“ and the snare drum as „1“, you get the following pattern after entering the configuration:

`000-0---` or `000-1---`

The first 3 triggers are the menu item number in a binary notation. If you are not familiar with binary numbers, it doesn‘t matter, all the patterns for the menu items are described later. The single trigger in the 2nd measure if the value of the parameter currently being edited.

To change the value or trigger an action for the current item, you again push the button for at least 1 second and release it. To switch to the next configuration item, push the button for a short time (less than 1 second).
As the immediate audible feedback can be a bit confusing, you can wait 2 seconds for the pattern to repeat to exactly know where you are.

The following configuration items are available, while the sequence in brackets is the first measure of the trigger pattern, 0 is kick, 1 is snare, and the second measure might give some information described in the description of the configuration item under „value“.

MIDI channel (000)
Value: 0 = omni mode enabled, 1 = MIDI channel configured, omni off
Change: pushing the button for at least 1 second switches to omni mode, sending any note event via MIDI uses the channel of this note event to set the channel of the unit and disable omni mode.

Incoming instrument assignment (001)
Value: 0 on entering the item, last assigned instrument during assignment
Change: sending a note event changes the assignment for this note event by cycling the possible instruments and the empty assignment.
Each time, you send the same note, you will hear the next instrument, if you cycled through all instruments, you will hear nothing, which is there to clear the assignment for this note, and after this, the cycle of instruments re-starts at the first instrument.
If you are happy with the instrument played on the last time you hit this note, you can proceed with the next note, as every note on message changes the assignment and gives immediate feedback.
The value in the feedback sequence (second measure) always is the last assigned instrument if you want to re-check your assignment, but you need to remember the note you sent last to know where this assignment takes place. You can assign the same instrument to multiple notes.

MIDI sync in (010)
Value: 0 if incoming MIDI sync is disabled, 1 if enabled
Change: toggle the value by pushing the button for more than 1 second

MIDI out (011)
Value: 0 if MIDI out is completely disabled, 1 if enabled
Change: toggle the value by pushing the button for more than 1 second

Outgoing instrument assignment (100)
Value: 0 on entering the item, last assigned instrument during assignment
Change: sending a note event changes the outgoing assignment if an instrument is assigned to this note in the incoming assignment.
You can only assign outgoing notes to instruments that already have an incoming note assigned. To assign the outgoing note for an instrument, just hit the incoming note of this instrument you want to be sent for this trigger.
So if you have the kick drum assigned to incoming notes C3 and C4 and the snare drum to D3 and hit C3 here, the kick drum sends a C3 now. If you hit D3, the snare drum now sends D3, the kick drum is unchanged. If you hit C4, the kick drum sends C4 now, the snare drum still sends D3.
So if you want a different assignment for the outgoing instruments, you first need to set the incoming assignments as you want the outgoing assignments to be, then hit all those notes once in the outgoing assignment.

On the other hand, this means that you may need to run the outgoing assignment again after changing the incoming assignment.

MIDI sync out (101)
Value: 0 if MIDI sync out is disabled, 1 if enable
Change: toggle the value by pushing the button for more than 1 second

Exit configuration without saving (110)
Value: 0 as confirmation that the configuration system ended without saving
Activate: push the button for more than 1 second
This doesn‘t save the configuration to non-volatile memory, but keeps it active while the machine is running. If you power down the machine, you will have the previous, stored configuration back on power-up.

Exit configuration with saving (111)
Value: 1 as confirmation that the configuration system ended and the configuration was saved
Activate: push the button for more than 1 second
This stores the current configuration to non-volatile memory, so you keep it between power cycled.

Technical documentation

The following describes the inner workings of MIDItriggs, while all information is valid for the full-featured installation in a Hohner Automatic Rhythm Player and the firmware delivered in the original package. Other installations may vary in the details.

MCU resource usage

Ports
Several port pins are used to input/output the signals from/to the host machine. The use of the port pins depends on installation and is best seen in the platform driver header file.

Timer 0
Running at 1 ms interface, this timer controls all the internal timing like trigger lengths, button time and configuration output sequencer.

Timer 1
This is a 16 bit timer to give the most possible accuracy for the generation of missing clocks for the MIDI clock output feature or machine clocks for the MIDI sync in feature.

Timer 2
This 8 bit timer runs the full range and updates another 16 bit variable, giving a 24 bit CPU clock cycle counter. This counter is used to determine the machine clock speed to calculate the timer value for Timer 1

External interrupts
Installation-dependent, used for handling of machine signals like clock, stop and start to update the timers for the MIDI clock and handle machine trigger scanning.

USART
Configured to 31250 bps to handle MIDI reception and transmission.

Most important module functions

main.c functions
mt_midi.c functions
mt_serial.c functions
mt_config.c functions
mt_config_button.c functions

Host platform driver module functions.

-TBD-

Adding new host platform drivers

Host platform drivers should have separate implementation files, as implemented in the 2 files mt_harp.h and mt_harp.c. Those files can be used as a basis for new host implementations.

-TBD-

Switching to a different micro controller

Hardware specific parts in mt_serial.c and host platform driver, clock out timer stuff in main.c

-TBD-

Schematics, board layouts, installation instructions

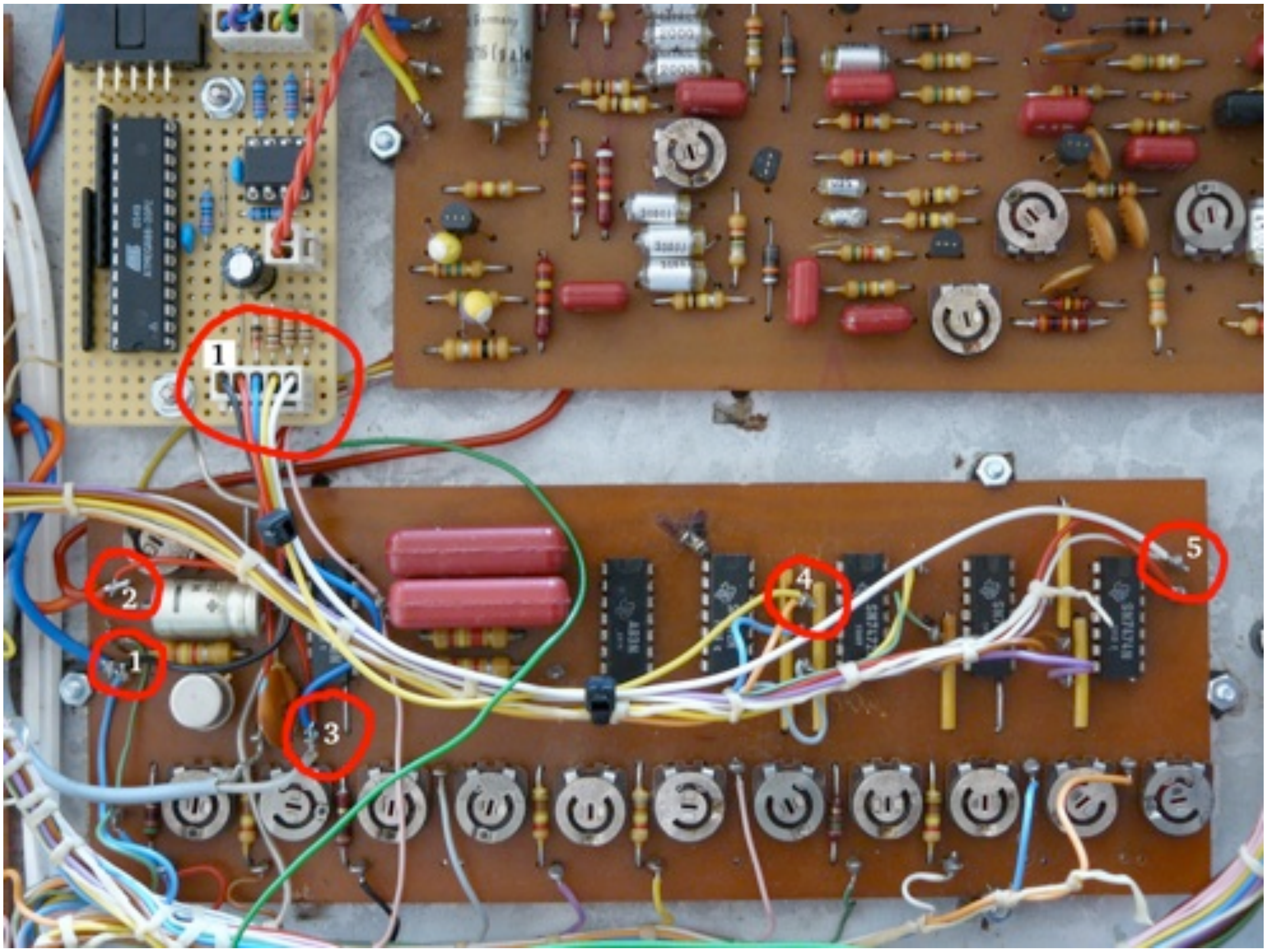
As the MIDItriggs project is very generic, there is no single hardware implementation. This document is growing for the different hardware variants that are implemented and for all the variants, schematics and PCB layout go here. To make your own PCBs, you should use the postscript files included in the hardware subdirectory of the project, as scaling might create some problems otherwise.

Hohner Automatic Rhythm Player veroboard prototype

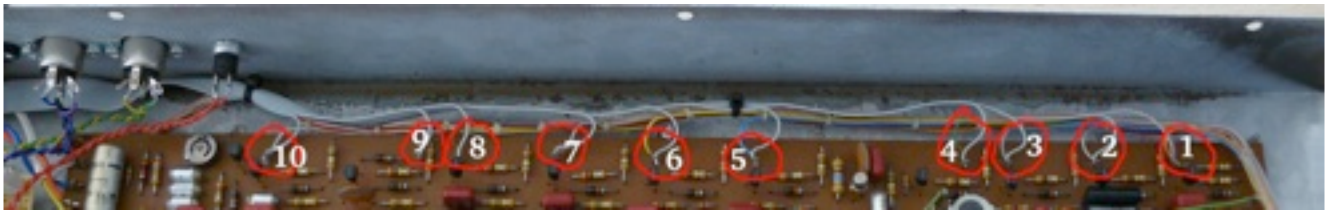
This is the first implementation that existed. It is not recommended to use the layout for a real PCB, though it would be possible. The layout is done for the veroboard version of this implementation, while some of the traces (those that are not on the 100 mil raster) are coated copper wires.

Installation

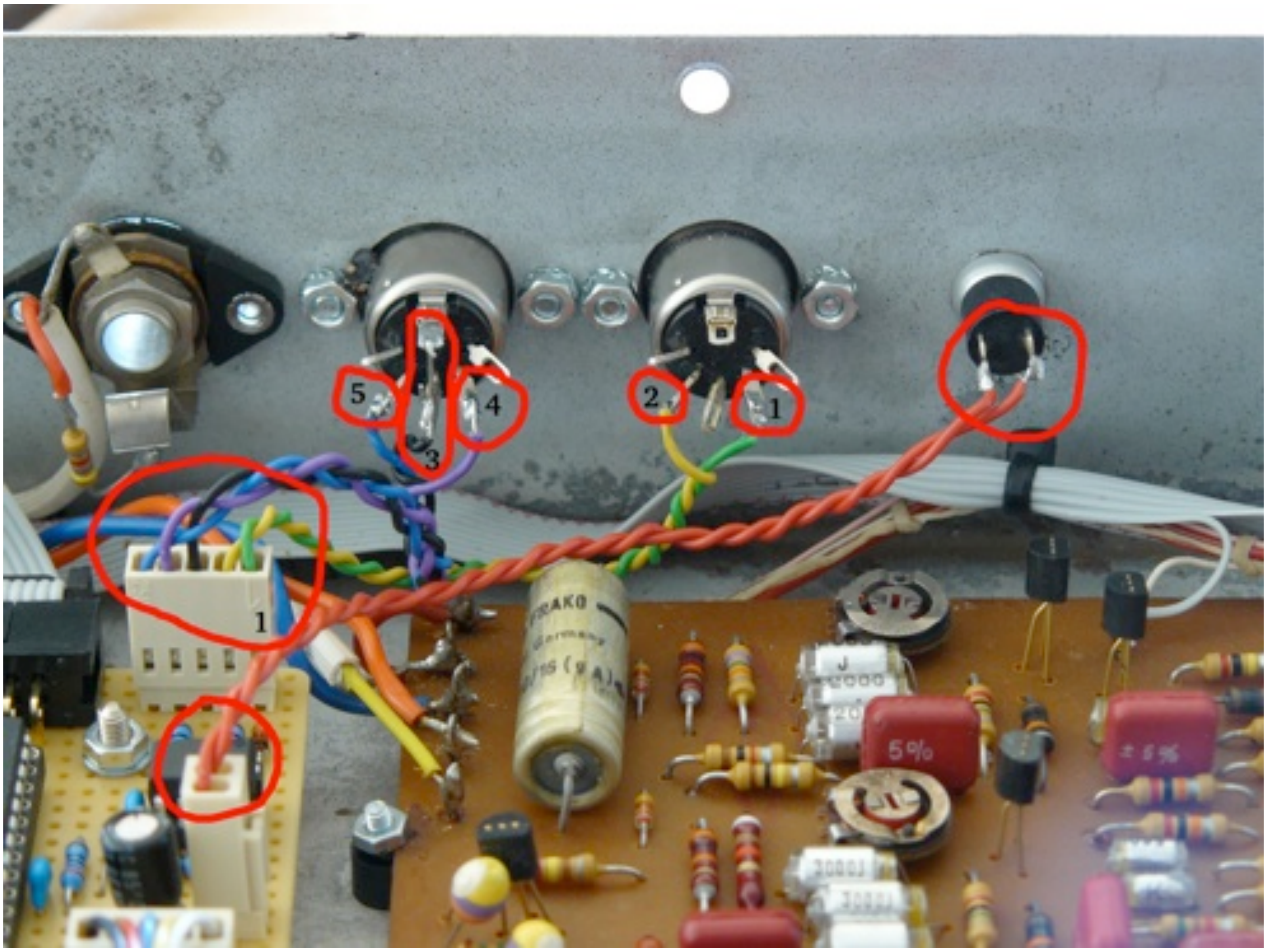
First, solder the cable carrying the power and sync signals in place. See the photo on the next page for pin assignment, the spots of interest are marked with red circles, the pin assignments labeled with white numbers.

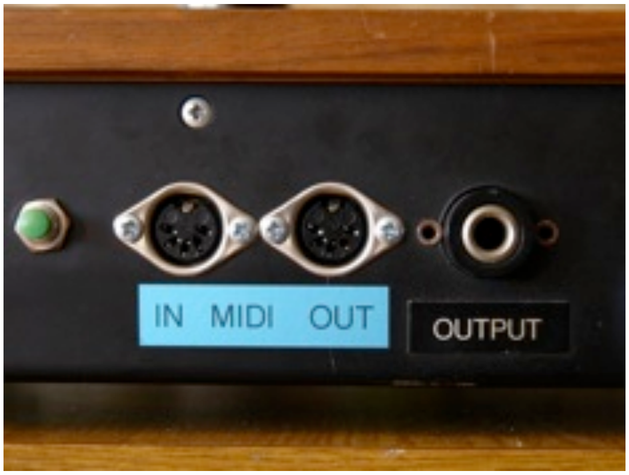


Next, solder the trigger lines for the instruments in place. The flat cable starts with the marked wire on the right side of the voice board and continues in order of wires and instrument signals to the right. In the following photo, the wire assignment is labeled with white numbers

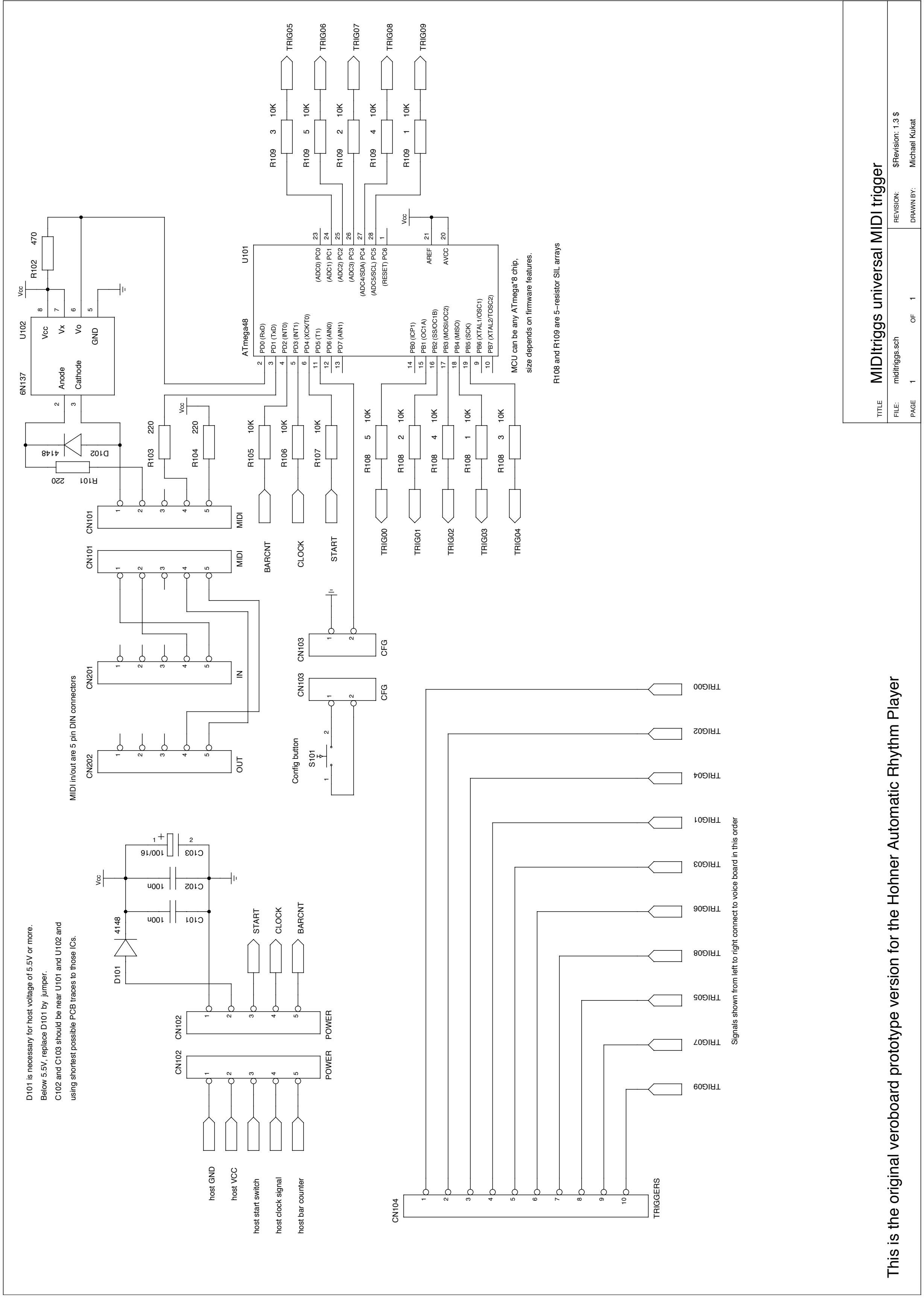


Now it‘s time to connect the configuration button and the MIDI connectors. The following photo illustrates the pin assignment of the board connector to the MIDI connectors. Take care to connect the middle pin of the MIDI OUT connector to the ground connector of the jack. The wiring of the configuration button doesn‘t matter.





The following pages show the schematic diagram and the board layout of this variant.



MCU can be any ATmega*8 chip, size depends on firmware features.

R108 and R109 are 5−resistor SIL arrays

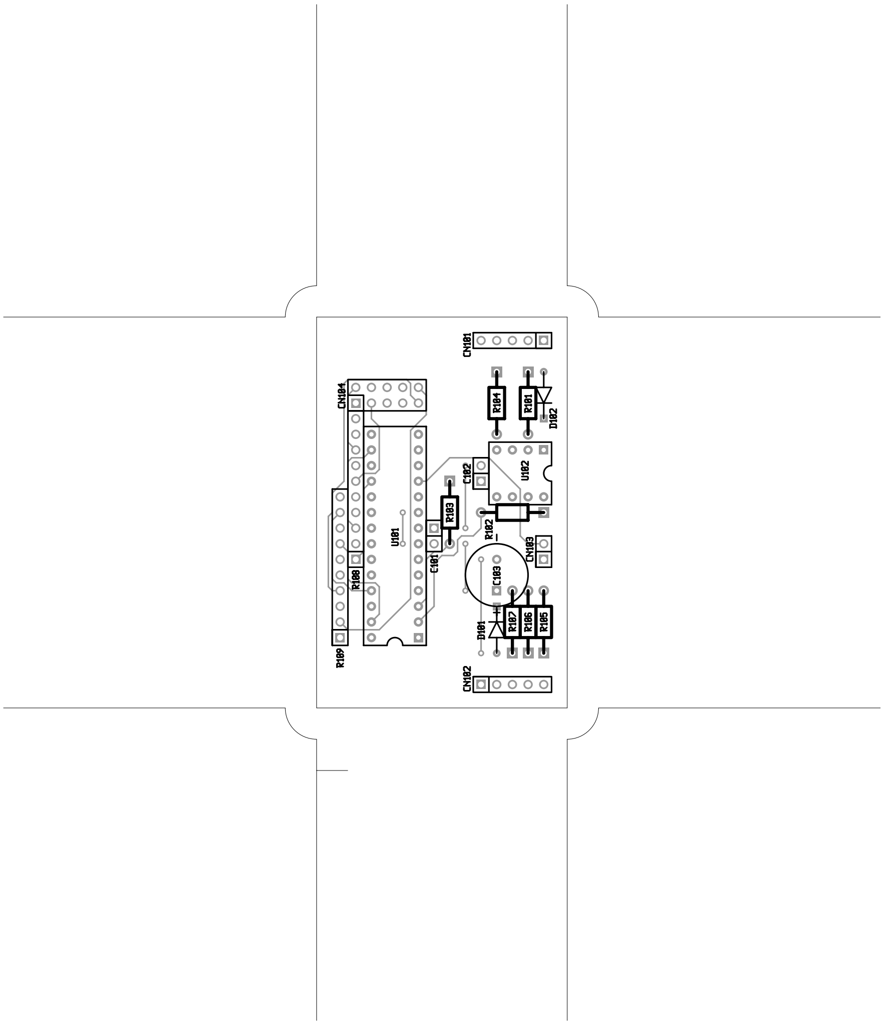
MIDI in/out are 5 pin DIN connectors

D101 is necessary for host voltage of 5.5V or more. Below 5.5V, replace D101 by jumper. C102 and C103 should be near U101 and U102 and using shortest possible PCB traces to those ICs.

Signals shown from left to right connect to voice board in this order

TITLE MIDItriggs universal MIDI trigger
FILE: miditriggs.sch
REVISION: $Revision: 1.3 $
PAGE 1 OF 1
DRAWN BY: Michael Kukat

This is the original veroboard prototype version for the Hohner Automatic Rhythm Player



frontassembly, scale = 1:1.000
/Users/michael/workspace/miditriggs/hardware/miditriggs.pcb

MIDItriggs universal board

After the public interest in this project, i created a real PCB layout for this project, which is the MIDItriggs universal board. In fact, it‘s just a simple micro controller board with the MIDI glue electronics, providing all the spare port pins via protection resistors on 2 header connectors. The purpose of this board is completely defined in the firmware, so you can trigger your drums with it, use it as a trigger to MIDI converter for your DIY e-drum kit, control your lights or power outlets via MIDI or just use it as a MIDI filtering tool. The board is designed in SMT to keep it very small, so you can mount it behind the MIDI connectors in your vintage beatbox or DIY project. The protection resistors are installed in arrays of 4 resistors each, while the Hohner version (called HARP2 in the sources) uses 10 KΩ resistors for the trigger and control signals, the CR-78 version (called CR78 in the sources) uses 1 KΩ resistors. For sure, you can mix the arrays, so if you attach a display/encoder on the 10 pin header connector, you use 1 KΩ resistor arrays or even 0 KΩ.

To upload the boot loader to this board, you can use the SPI signals, which also works with the 10 KΩ resistors in the way. I build a „programming adapter“ connecting the MIDItriggs to my AVRISP programmer. The RESET pin is not available on the header connectors, but there is a larger pad on the board that can be touched with a piece of wire. This is okay if you just want to upload the boot loader onto the board, while it is not very handy if you want to frequently update the firmware. Using the boot loader (the MIDIboot project), you can update the firmware via MIDI, which simplifies development a lot.

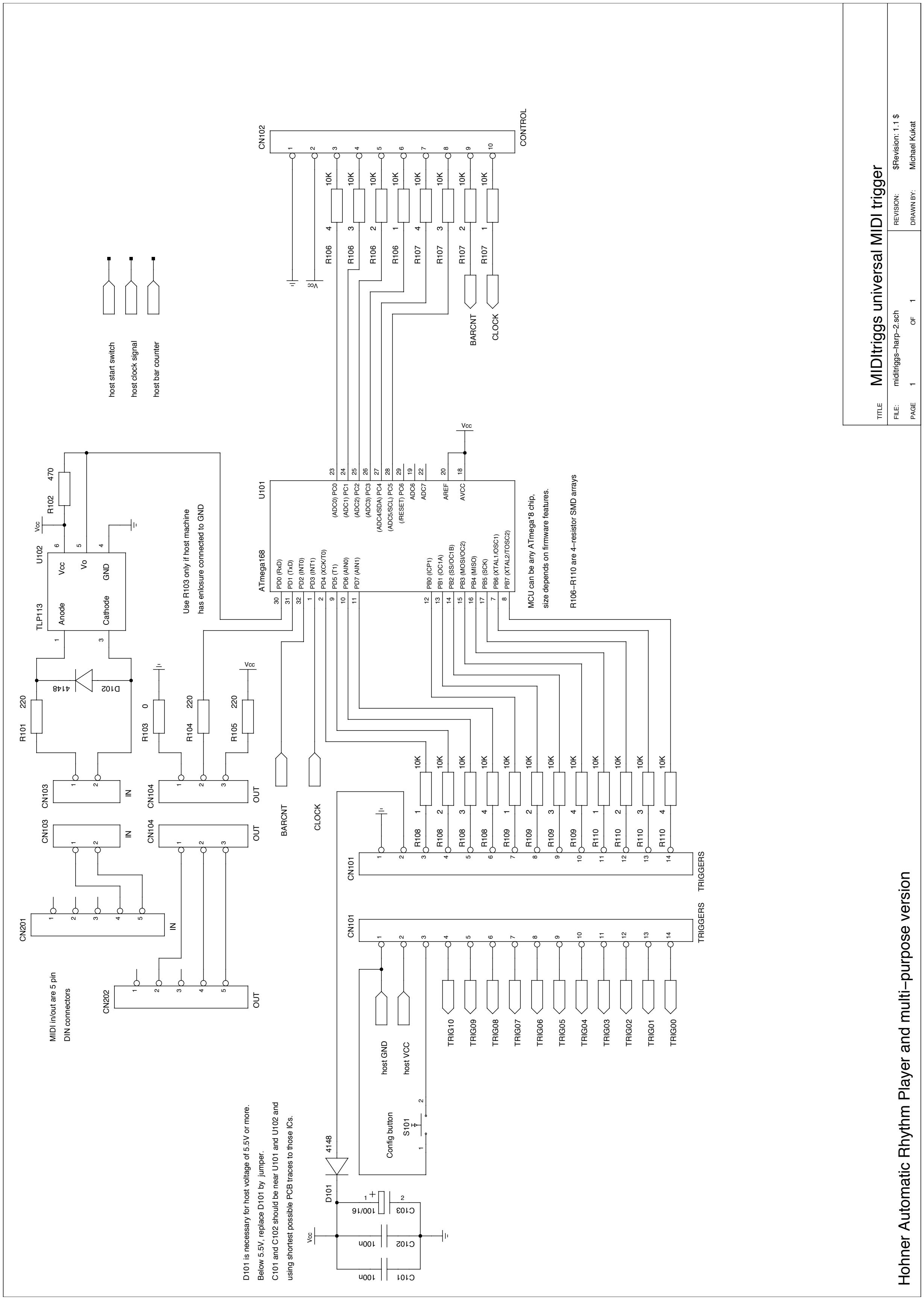
The schematics shown on the following page are for the Hohner Automatic Rhythm Player and are untested in this configuration, because the only host machine using this board version uses an encoder/display configuration (not included in the source package) and so doesn‘t have the single configuration button. Besides this, the trigger signals are mapped differently.

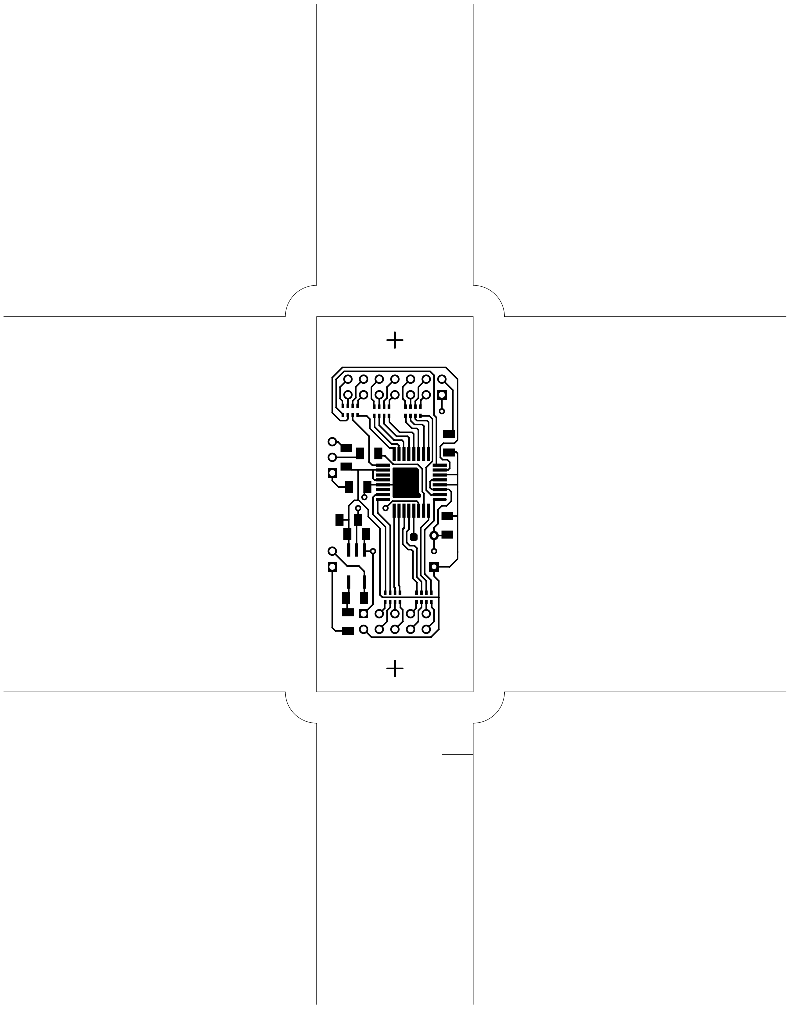
There are some important things about this circuit:

If your enclosure isn‘t connected to the GND of the MIDItriggs, you need to leave away R103 and connect pin 2 of the MIDI out connector to its shield. If you are sure that you don‘t have GND on a different voltage potential than the enclosure ground, install R103 (a 0 Ω resistor) to correctly ground pin 2 of the MIDI out connector.

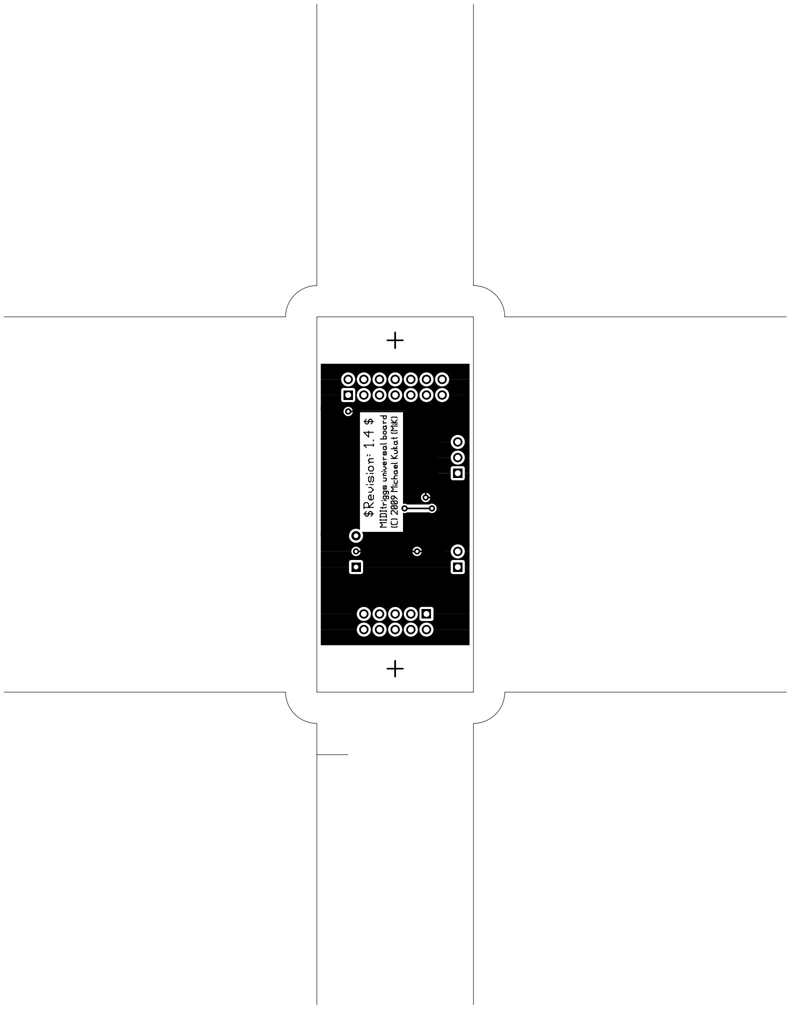
If your power supply is not clean enough and provides more than 5V, install D101 and feed power to the board through CN101. If you have lots more than 5V, you can use a Zener diode as D101. For exact 5V in the host machine, install a 0 Ω resistor in place of D101.

Carefully plan your configuration and the port assignment for simple wiring and efficient use of resources in the host machine. Especially take care of the 10 pin header, as you have 2 interrupt signals and the TWI on the pins there, which you might want to use with those features. The 10 pin header also connects the analog inputs to the MCU, giving even more possibilities.

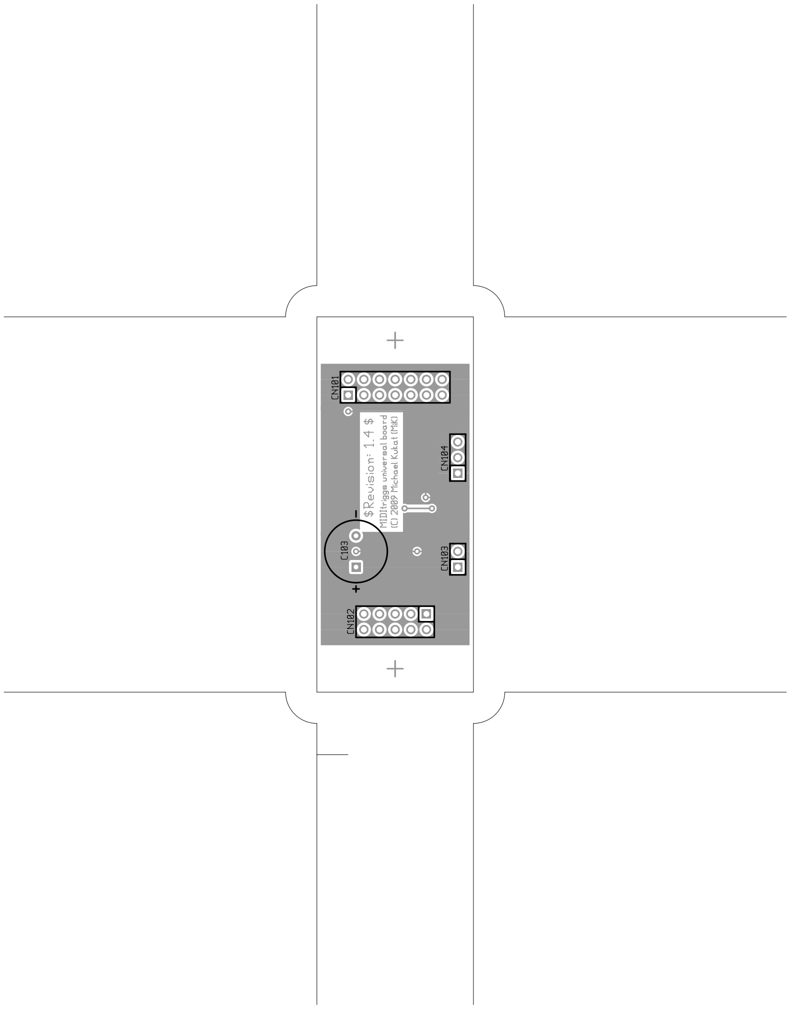

MIDItriggs universal MIDI trigger
TITLE
FILE: miditriggs–harp–2.sch
PAGE 1 OF 1
REVISION: $Revision: 1.1 $
DRAWN BY: Michael Kukat
Hohner Automatic Rhythm Player and multi–purpose version
CN102
CONTROL
R106
R107
10K
BARCNT
CLOCK
host start switch
host clock signal
host bar counter
U101
ATmega168
PD0 (RxD)
PD1 (TxD)
PD2 (INT0)
PD3 (INT1)
PD4 (XCK/T0)
PD5 (T1)
PD6 (AIN0)
PD7 (AIN1)
PB0 (ICP1)
PB1 (OC1A)
PB2 (SS/OC1B)
PB3 (MOSI/OC2)
PB4 (MISO)
PB5 (SCK)
PB6 (XTAL1/OSC1)
PB7 (XTAL2/TOSC2)
(ADC0) PC0
(ADC1) PC1
(ADC2) PC2
(ADC3) PC3
(ADC4/SDA) PC4
(ADC5/SCL) PC5
(/RESET) PC6
ADC6
ADC7
AREF
AVCC
Vcc
MCU can be any ATmega*8 chip,
size depends on firmware features.
R106–R110 are 4–resistor SMD arrays
U102
TLP113
Anode
Cathode
Vcc
Vo
GND
R102
470
R101
220
D102
4148
Use R103 only if host machine
has enlosure connected to GND
R103
0
R104
220
R105
220
CN103
IN
CN104
OUT
CN201
CN202
MIDI in/out are 5 pin
DIN connectors
CN101
TRIGGERS
R108
R109
R110
TRIG10
TRIG09
TRIG08
TRIG07
TRIG06
TRIG05
TRIG04
TRIG03
TRIG02
TRIG01
TRIG00
host GND
host VCC
Config button
S101
D101
4148
C101
100n
C102
100n
C103
100/16
D101 is necessary for host voltage of 5.5V or more.
Below 5.5V, replace D101 by jumper.
C101 and C102 should be near U101 and U102 and
using shortest possible PCB traces to those ICs.



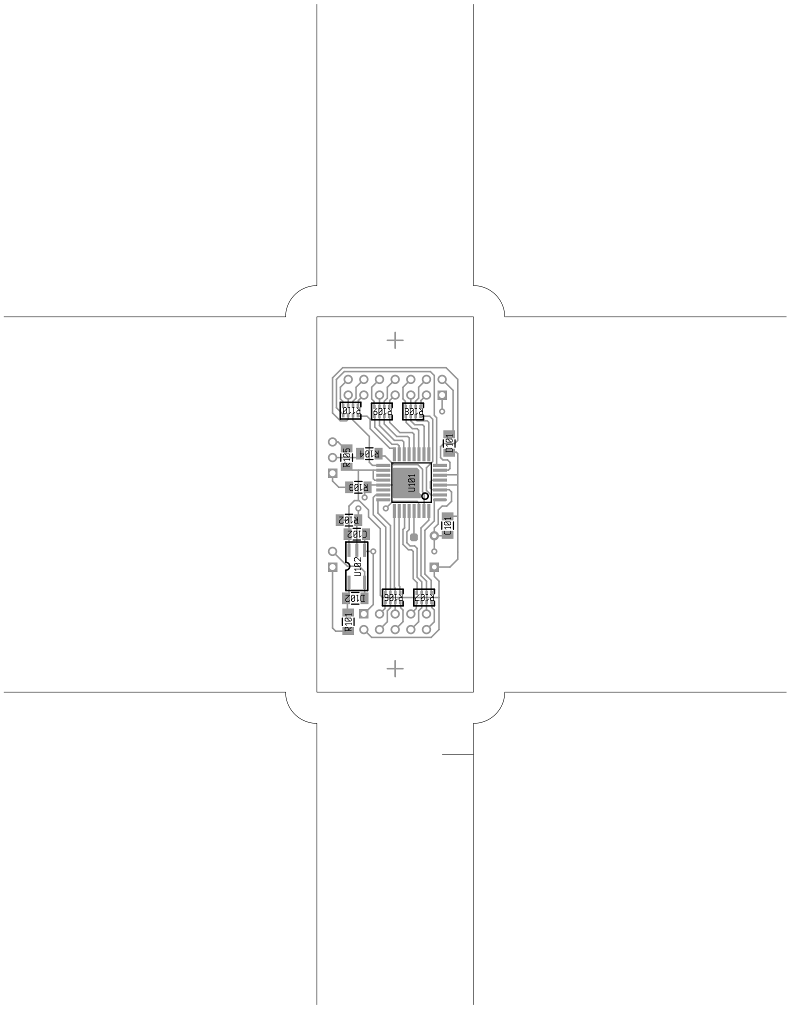
back (mirrored), scale = 1:1.000
miditriggs-harp-2.pcb



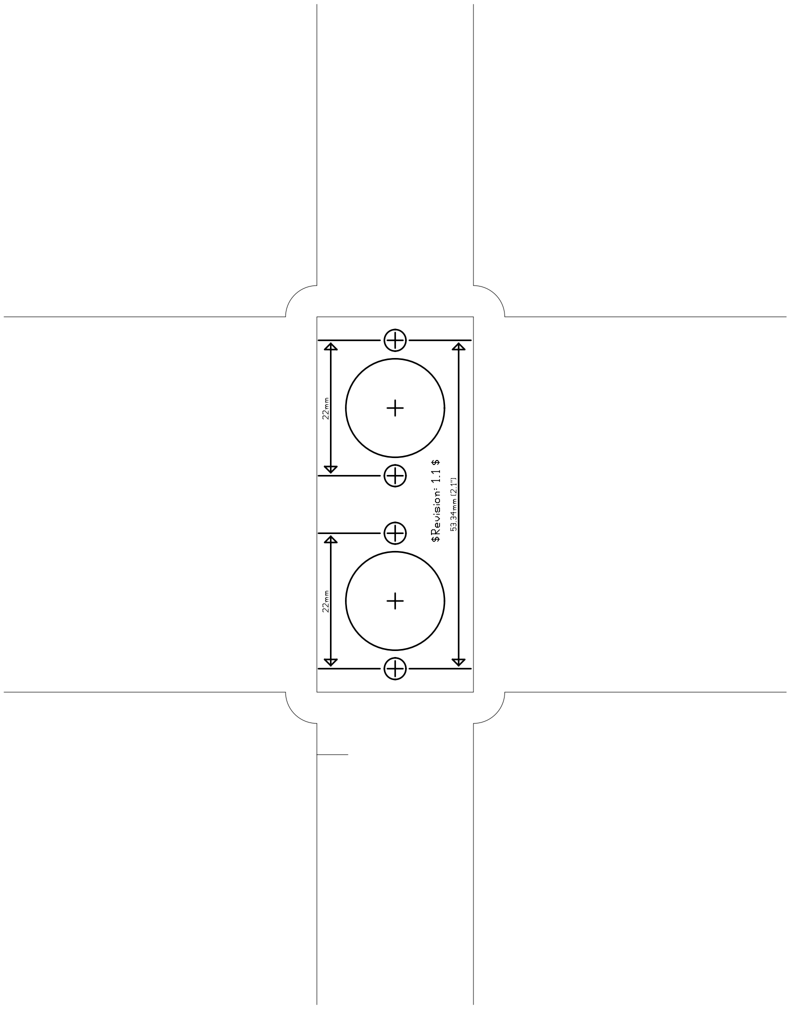
front, scale = 1:1.000
miditriggs-harp-2.pcb



frontassembly, scale = 1:1.000
miditriggs-harp-2.pcb



backassembly (mirrored), scale = 1:1.000
miditriggs-harp-2.pcb



frontsilk, scale = 1:1.000
drillmask.pcb

MIDItriggs universal board in a Roland CR-78

The installation of MIDItriggs into a CR-78 is a bit more complicated as you need some modifications to your CR-78 to enable the bidirectional triggers and MIDI clock control without disrupting the normal standalone operation of the machine. For the CR-78, D101 is a 0 Ω resistor, R103 is a 0 Ω resistor and all the resistor arrays are 1 KΩ. The following pinout for the headers is used:

CN101, Trigger signals and configuration button:
Pins 1 and 2 go to some power lines near the trigger transistors
Pins 3-13 go to the trigger resistors after modifying them. It starts with the Accent trigger on pin 3, BD on pin 4, up to RS on pin 13
Pin 14 is connected to the configuration button, the other end of this button is connected to a GND spot nearby (the connector board is suitable for this)

CN102, control signals and TWI interface:
Pins 1 and 2 carry GND and VCC and are used for TWI peripherals (like the PCF8574 on the memory expansion board)
Pin 3 - Connected between the diodes feeding the start/stop button signal to the flip flop
Pin 4 - Start/stop flip flop output, best grabbed at the correct end of R120
Pin 5 - The machine clock to the machine, connected to the through-hole where the trace has been cut
Pin 6 - Write switch signal, connected to IC120-13
Pins 7 and 8 carry the TWI signals for peripherals
Pin 9 - Start/stop state the CPU sees, connected to IC122-9
Pin 10 - Machine clock from machine clock generator, connected to IC120-2

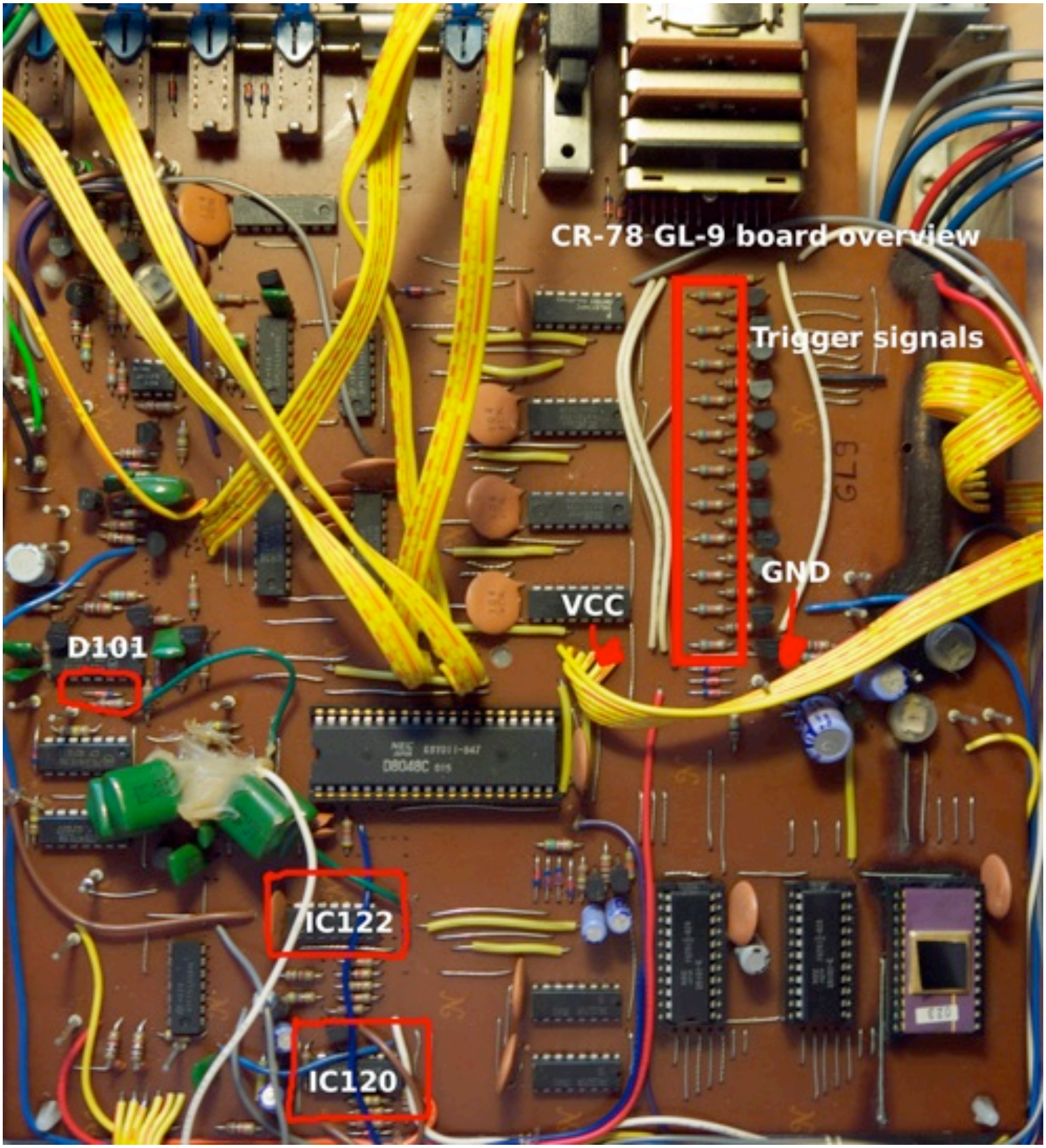
Installation

WARNING! There are 3 different board versions, GL-9, GL-9A and GL-9B. This installation procedure was initially designed for the GL-9B board.
The GL-9 and GL-9A should be nearly identical (single sided board), but the GL-9A has not been seen yet. On the GL-9, some components (like R201, which has to be removed) are on the lower side of the board. The following instructions are valid for the GL-9B board, differences on the GL-9 board are explicitly mentioned.

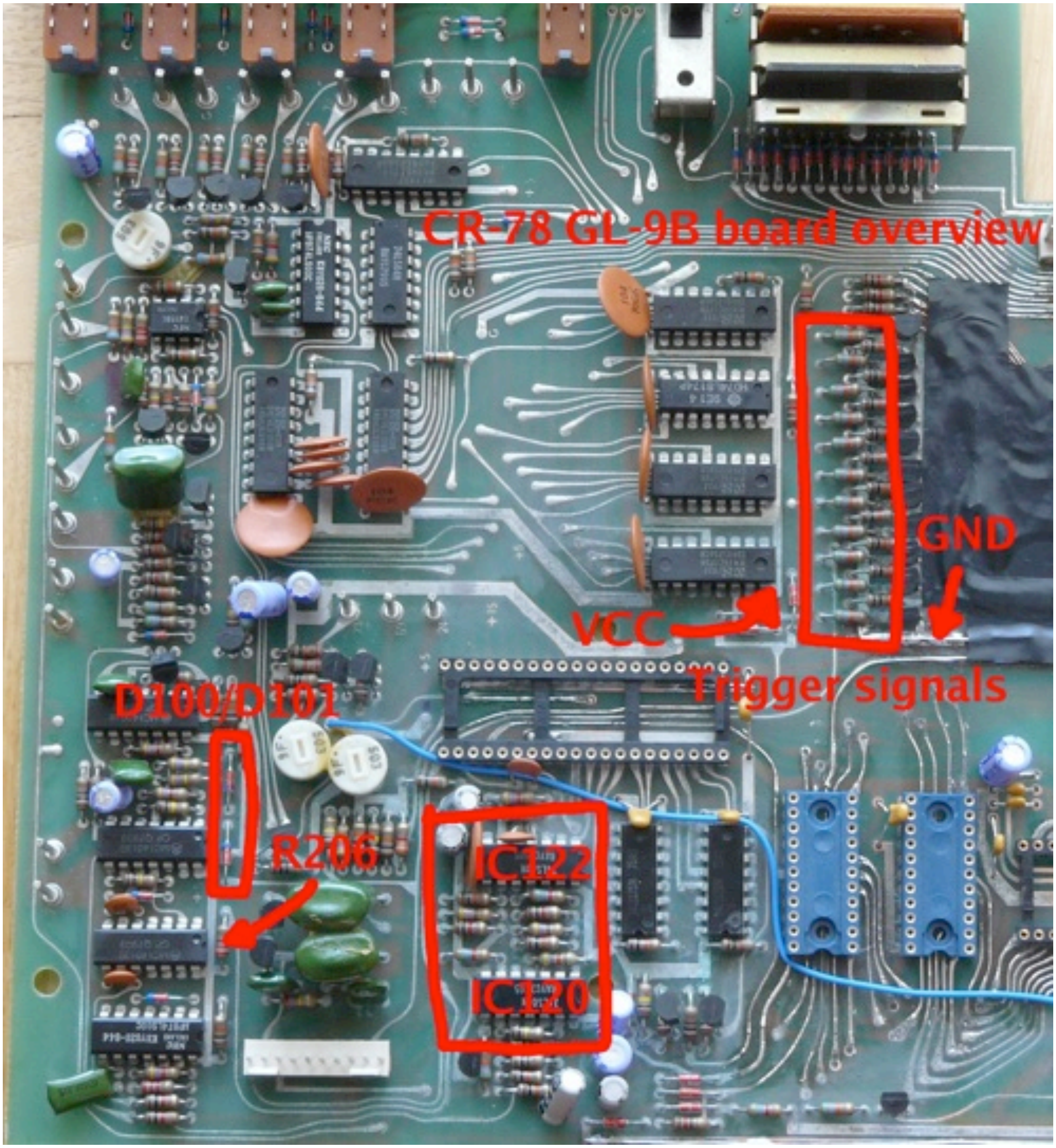
Check the following overview photos against your board to see if you have the right board version.

GL-9B: On the upper side, make sure, D100 and D101 on the left side of the photo are arranged in the same way.
GL-9: D100 is next to IC122, D101 is between IC109 and IC111, make sure you find them. IC120 and IC122 have the same horizontal position on GL-9B, they are a bit shifted on the GL-9 (and maybe GL-9A).
Besides this, the photos give you an overview in what areas you have to work, so you can find those areas for the detail photos coming later.

GL-9 board, upper side

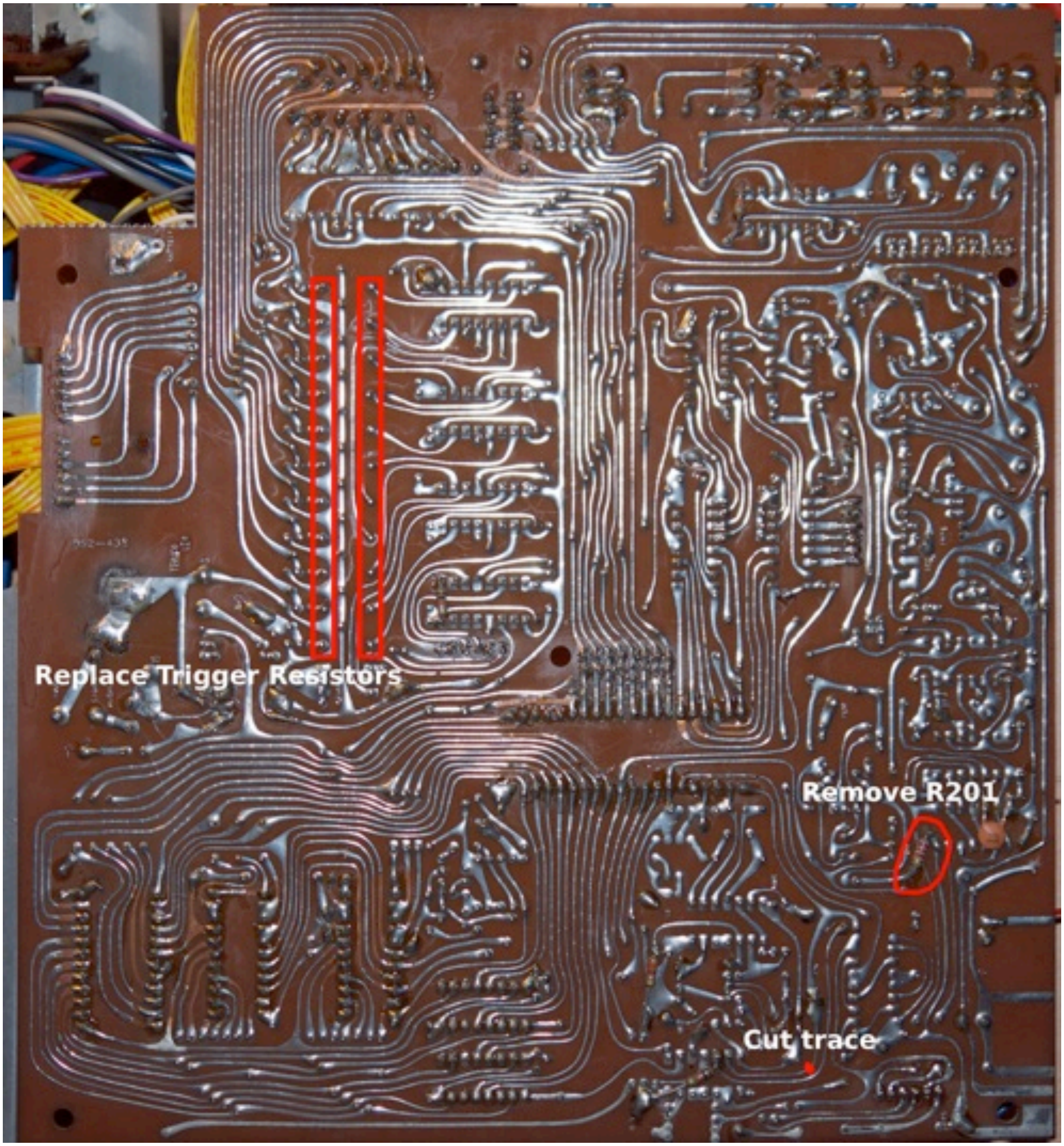


GL-9B board, upper side

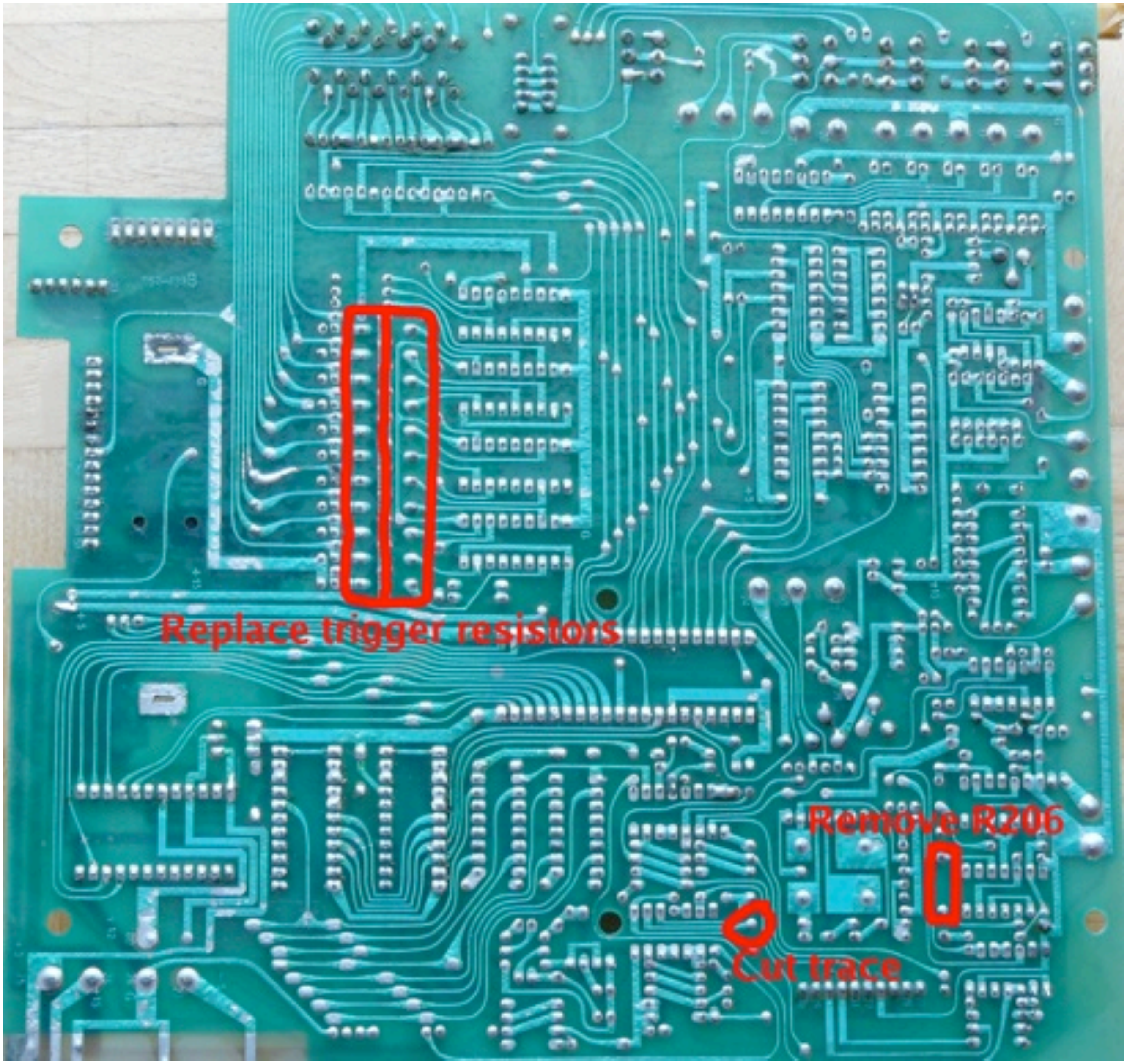


CN101 connects to the spots marked with GND and VCC and all the trigger signals after you replaced the resistors. R206 (on GL-9B board) needs to be removed
2 (GL-9) or 4 (GL-9B) wires of CN102 go to IC120 and IC122, a trace on the other side needs to be cut, 2 wires of CN102 go near D101 and a resistor left of the IC next to it. On GL-9, 2 wires go to jumpers next to the CPU and left of IC122.

GL-9 board, lower side

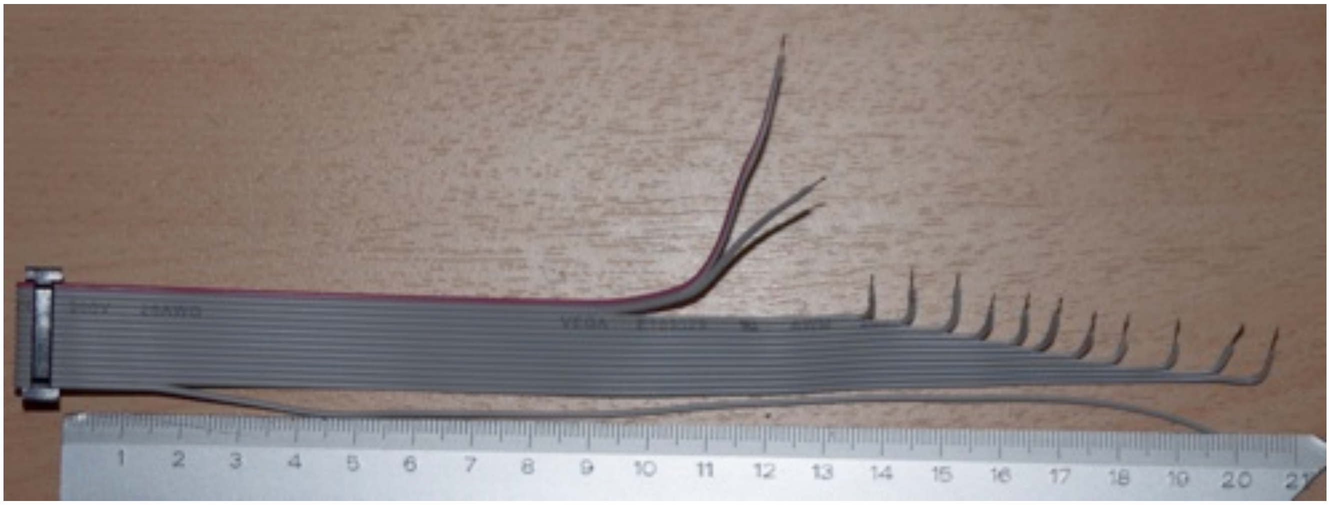


GL-9B board, lower side

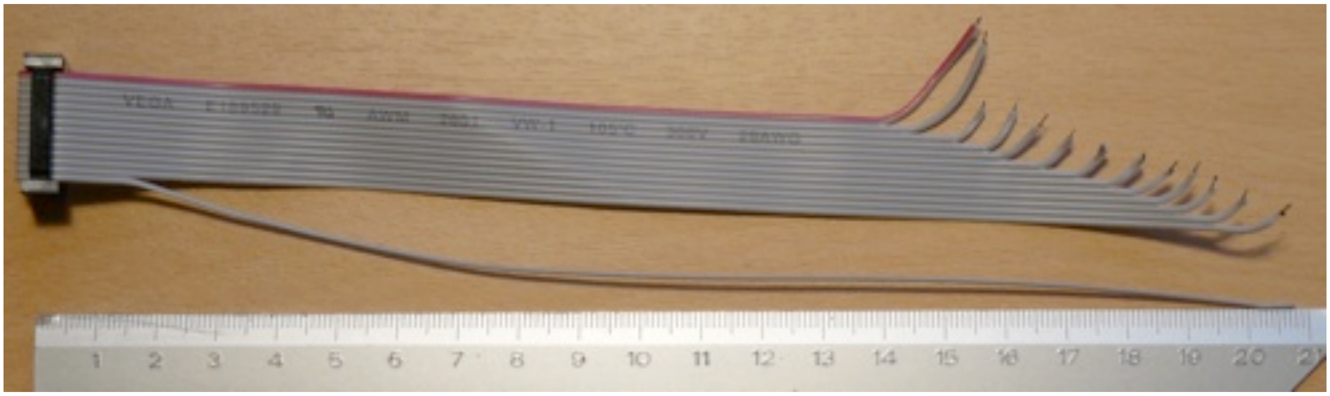


When unsoldering the trigger resistors, only solder at those 22 pins within the red markings. This also shows the places of the trace to cut and the R201 (GL-9) or R206 (GL-9B) to remove. On the GL-9A, R201 should be on the other side of the board, you should be able to locate it with the CR-78 service notes document available on the web.

Now have a look at the cables
Trigger cable for GL-9 (any maybe GL-9A)

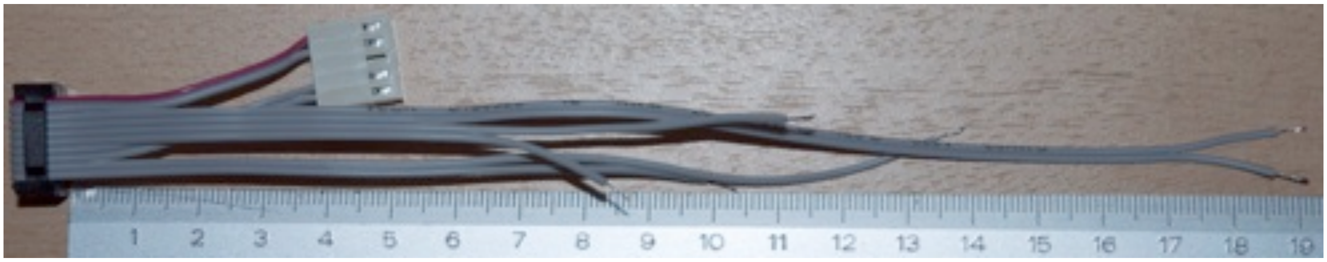


Trigger cable for GL-9B

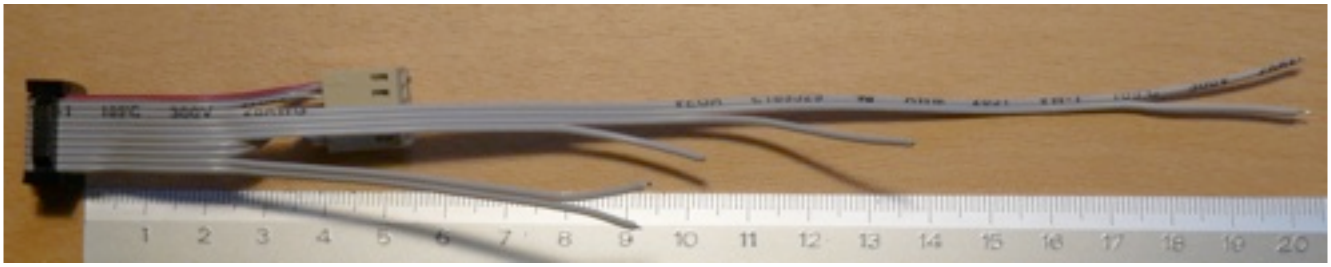


This is the cable for CN101, which is connected to GND with pin 1 (the wire with the red stripe), VCC with pin 2, the 11 trigger signals and the last wire is for the configuration button.

Control cable for GL-9



Control cable for GL-9B



This cable goes to CN102 and carries peripheral connection wires on pins 1, 2, 7 and 8 to the other connector (pins 1, 2, 4 and 5). Pins 3-6 and 9-10 go to the CR-78 board.

With those photos, you should be able to create your own cables with the wires in the tested lengths for this installation. The CN101 cable has a long wire for the configuration button, which usually is cut as needed, while the part of the wire being cut off can be used as the other wire for the configuration button, being connected to GND.

Now it‘s time to take the unit apart and put the MIDItriggs into it. This needs lots of concentration and electronic skills as you might need to remove the wire wrap connections to the board and later being able to restore them. Besides this, it needs good soldering skills. If you make any mistakes here, you can trash MIDItriggs, your CR-78 or both. Or maybe even your life, if you don‘t take care! So please read through the following steps carefully before removing the first screw and then decide if you really want to dive into this adventure.
For the wire wrap connections - if you carefully unwrap them from the end and keep the post with the sharp edges intact, you should be able to wrap the wire back later with having the same good contact as before. If you make mistakes here, you still can apply a bit solder to make sure the cables don‘t fall off, but it definitely is possible to restore the wire wrap connections if you have some experience with this technique, even if the normal wire wrap you might have learned works with different wires and tools. The physics are the same.

First, take the unit apart the following way:
- After removing the wooden enclosure, remove the connector board by unscrewing all the connectors at the rear - take care about the order of all the washers and keep them in a safe place
- Remove the rear panel by unscrewing the 4 screws and pulling it off. Pull the power cable through the hole to be able to put it a bit away from the machine
- Now remove the front panel by pulling all the knobs and unscrewing the 2 screws on the upper side and the 2 screws on the lower side. Take care about washers between the front panel and the main chassis.
- Remove the 2 screws left and right of the cancel switches and remove the nut of the instrument select switch
- Pull all connectors of the logic board and put them a bit away (only possible on the GL-9B, you may need to open the wire wrap connectors on the GL-9)
- Squeeze the board retainers and carefully pull the board upwards until you can get it a bit out of the unit, turn it around carefully without ripping off any cables, you need to work on the lower side later

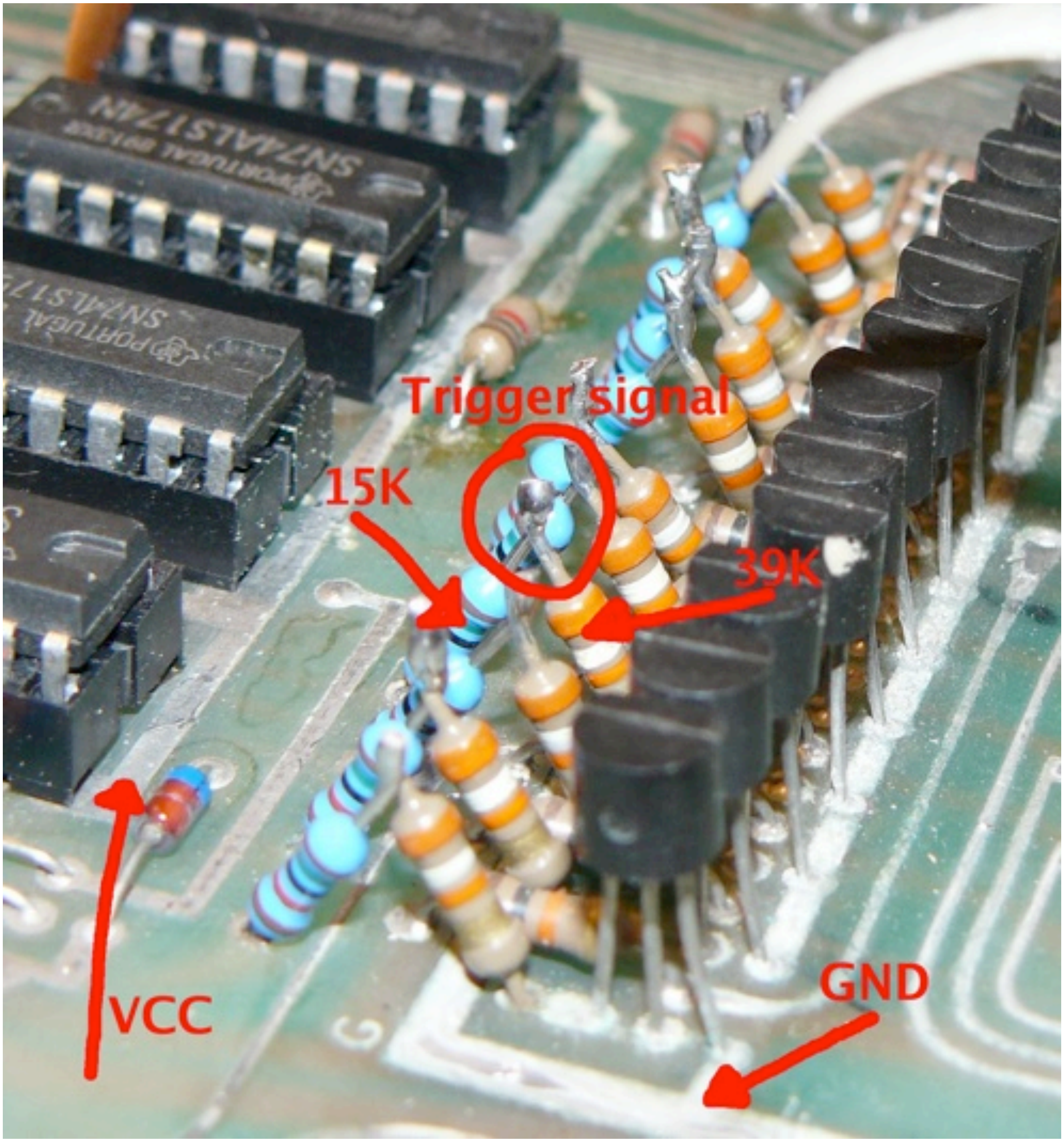
Now, it‘s time to prepare the rear panel for the MIDItriggs board to be mounted

Drill the holes for the MIDI connectors and configuration button:
- Use the drill mask (check the scaling after printing!) to find a suitable spot. Leaving 20 mm between the center of the connector row of the CR-78 and the upper edge of the MIDItriggs board is a good vertical position, center the board around the Start/Stop switch connector and put the configuration button on the same vertical position, aligned to the center of the 8/16/COMBO switch. Make the drill mask stay in position with some adhesive tape.
- Mark all the necessary holes if you don‘t dare to drill through the drill mask (which could be ripped away by the drill) and remove the drill mask afterwards.
- Drill all 7 holes with 3.5 mm
- Drill the 2 MIDI connector holes and the configuration button hole with 7.5 mm
- Drill the 2 MIDI connector holes with 10 mm
- Use a step drill to drill the 2 MIDI connector holes to 16 mm
- Use a deburring tool or some sandpaper to remove sharp edges on all the holes
- Looking at the rear of the unit, insert the MIDI out connector (the one with 3 wires), into the left hole, forming an U with the holes, so the cable goes downwards, put a screw into the right hole of this connector and on the other side, put a lock washer and a nut onto the screw. Tighten the screw a little bit, not too hard currently.
- Do the same with the MIDI in connector, but use the screw on the left hole of the connector
- Use a spacer bolt, add 1 normal washer, put it through the MIDItriggs board, the bolt on the side with the SMD components, add another washer on the other side and a nut, tighten the bold. Do the same with the other spacer bolt.
- Insert a screw in each remaining MIDI connector board, add the lock washer and use the MIDItriggs board (MIDI connectors facing downwards) with the bolts to screw this together. Tighten the screws, holding the bolts so that they don‘t turn.
- Now tighten the remaining screws of the MIDI connectors.
- Connect the MIDI connectors to the MIDItriggs board
- Install the configuration button, having the lock washer inside the unit and the nut outside

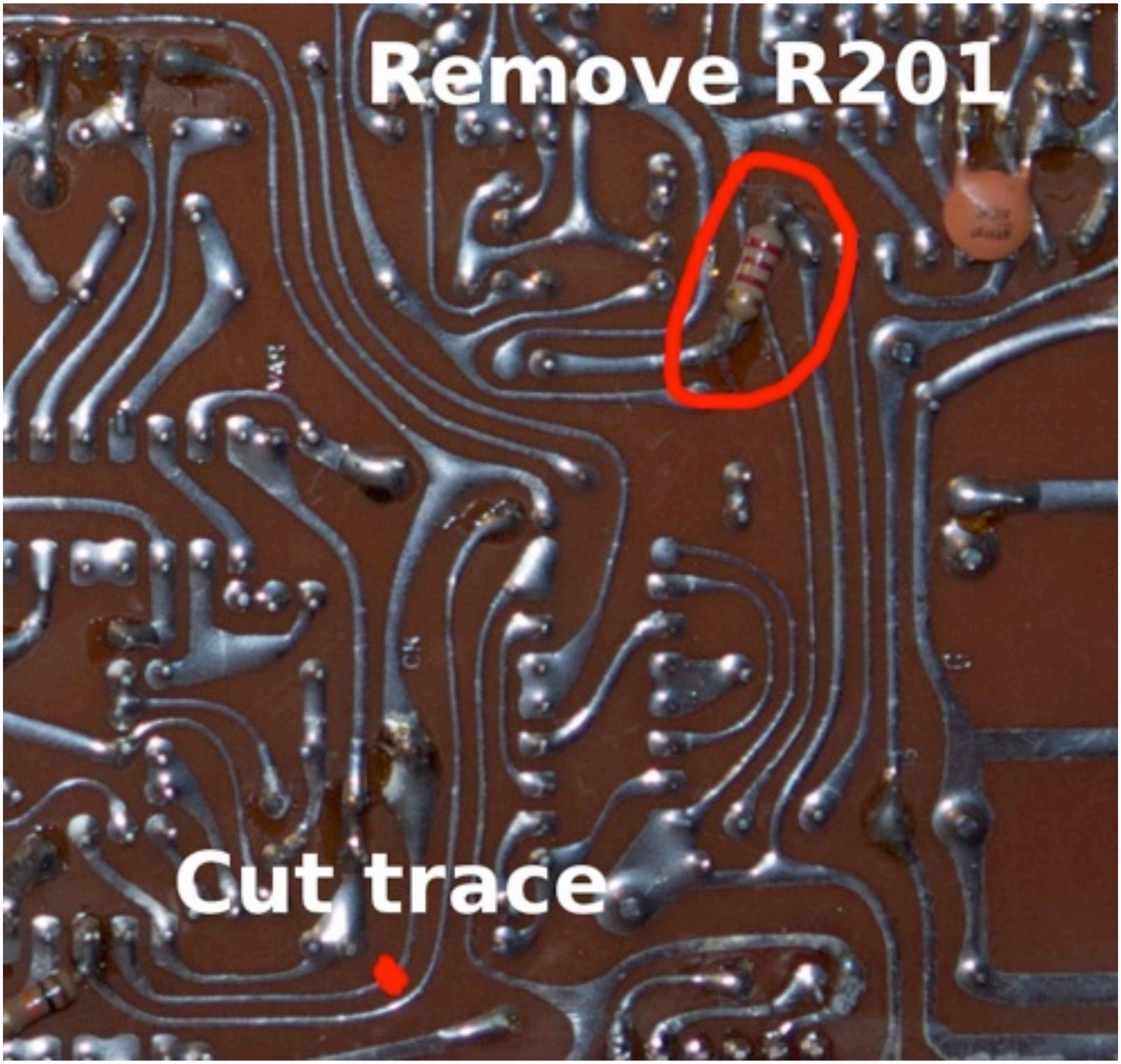
Your rear panel is now nearly completed, it‘s time to work on the board.

Working on the lower side of the board:
- See the board photos above to locate the trigger resistors and unsolder all of them
- Solder one wire of each 39 KΩ resistor to the transistor end of where the original trigger resistors have been (see photo on the next page for details)
- Put something under the board, away from the trigger resistors, to work on the upper side while putting wires through the board.
- Put the 15 KΩ resistors into the holes, twist the other end with the free end of the 39 KΩ resistors and apply a bit of solder to keep them together, cut off the remaining wires.
- Turn around the board and solder the 15 KΩ resistor wires in place, cut off all remaining wires.
- See photos above and detail photos on the next pages to locate the trace to cut. Carefully cut this trace with a sharp knife, make sure there is no connectivity left between the 2 ends
- See photos above to locate R206 (R201 on GL-9 and GL-9A boards, see below) and remove it

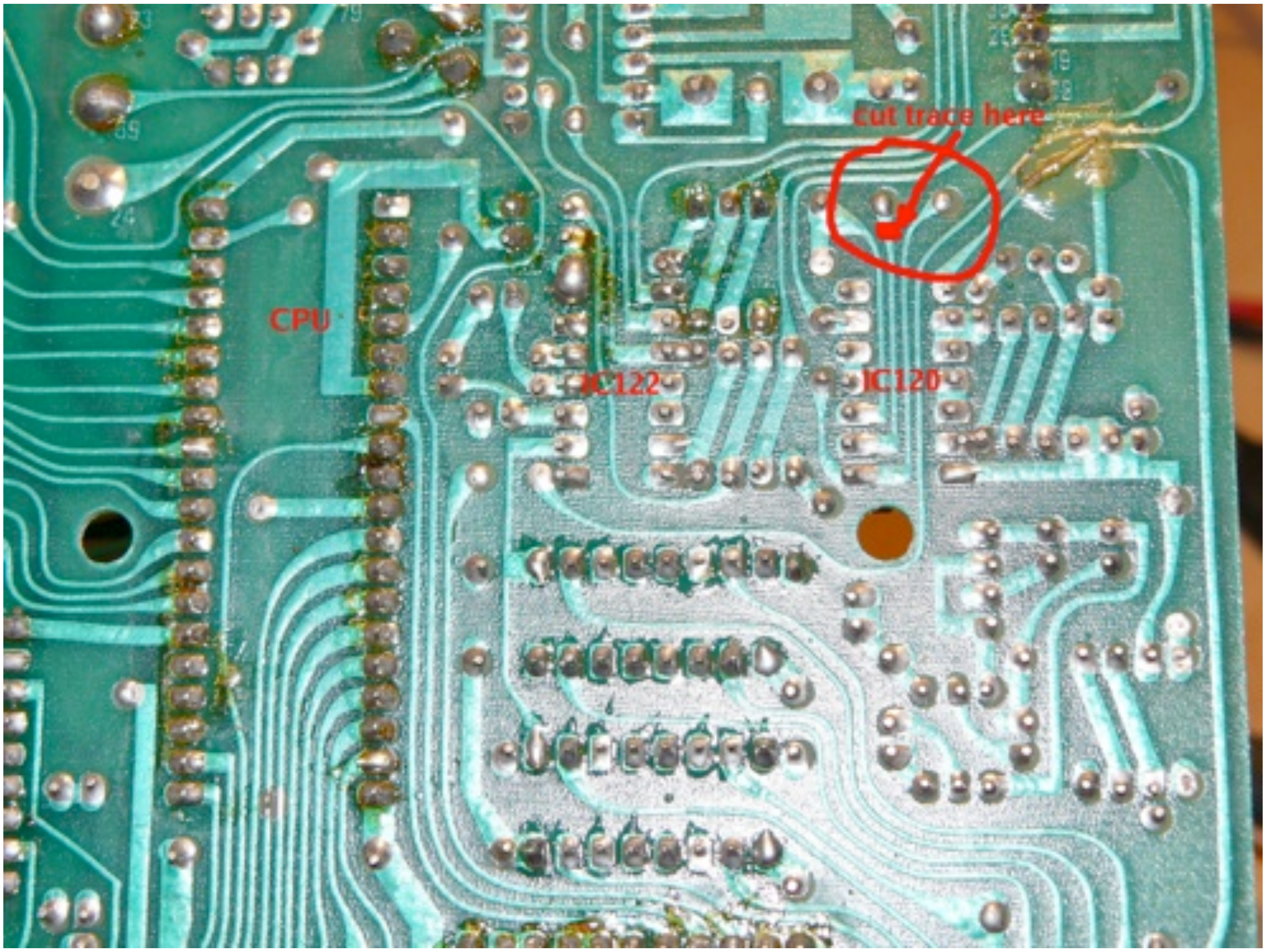
Trigger resistors (shown on GL-9B board)



GL-9 board lower side detail



GL-9B board lower side detail



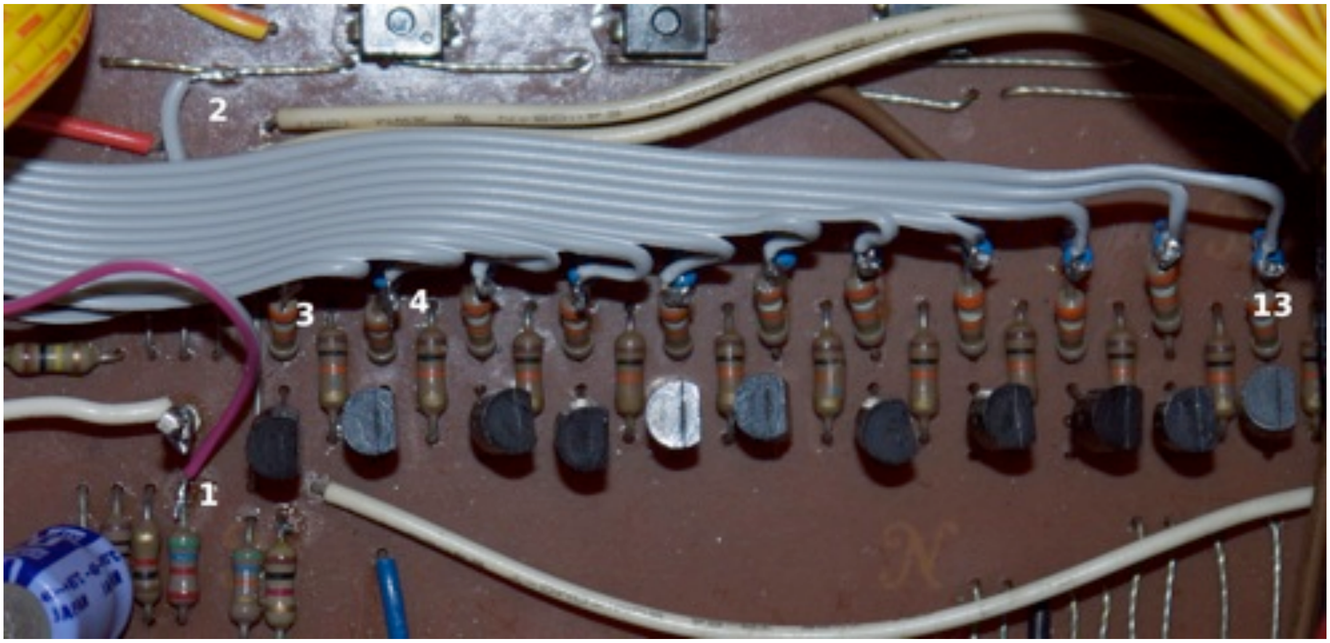
Putting the board and front panel back in place:

- Carefully put the board back into the unit until it safely sits on the retainers, make sure you didn‘t get any cables between board and chassis.
- Put the screws for the cancel switches and the nut for the instrument select switch back in place, take care of the instrument select switch washer.
- Put the front panel back in place, take care about the washers you found while removing it

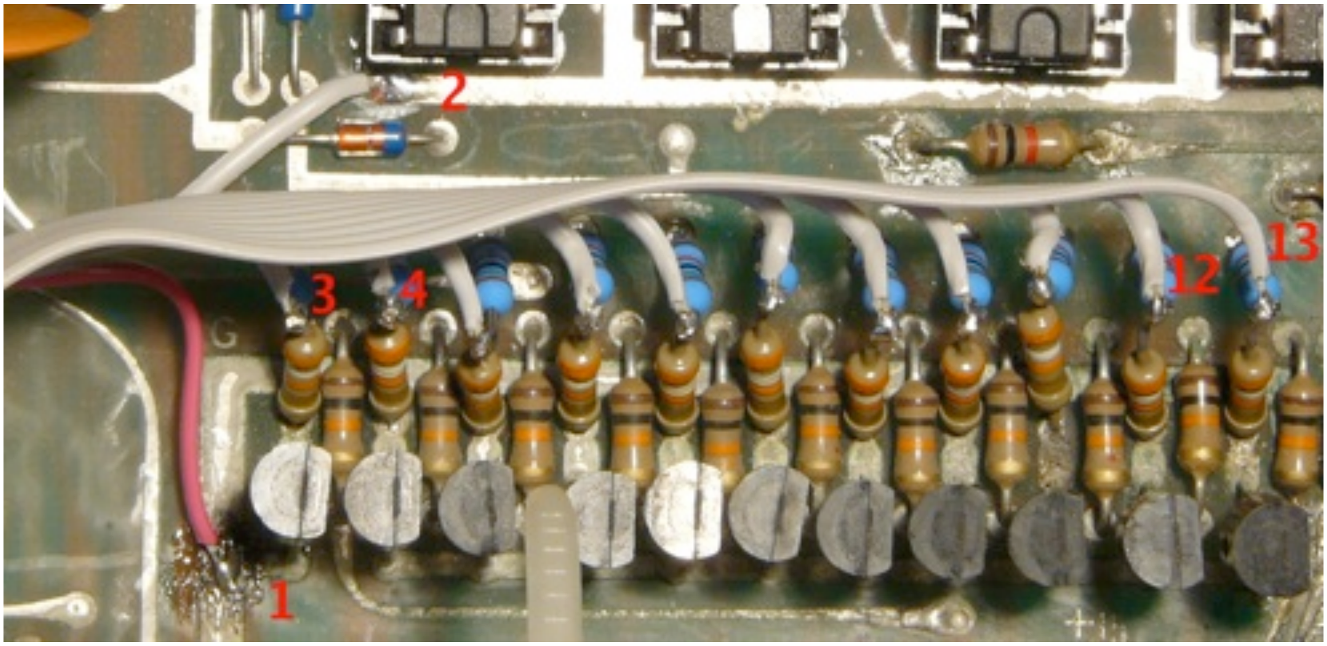
Installing the trigger cable, CN101:

- See the photos below for the details about the trigger cable and solder it into place. Pin 1 (the wire with the red stripe) goes to GND, which is one end of a resistor (GL-9) or a large trace passing the transistors (GL-9B), pin 2 goes to +5V, which is a jumper next to the trigger latch ICs (GL-9) or a large trace passing the left edges of the 4 ICs on the other side of the trigger resistors. Pins 3-13 then go to the spots where you soldered the 39 KΩ and 15 KΩ resistors together, while pin 3 starts with the resistor nearest to the rear of the unit

GL-9 board trigger cable installation



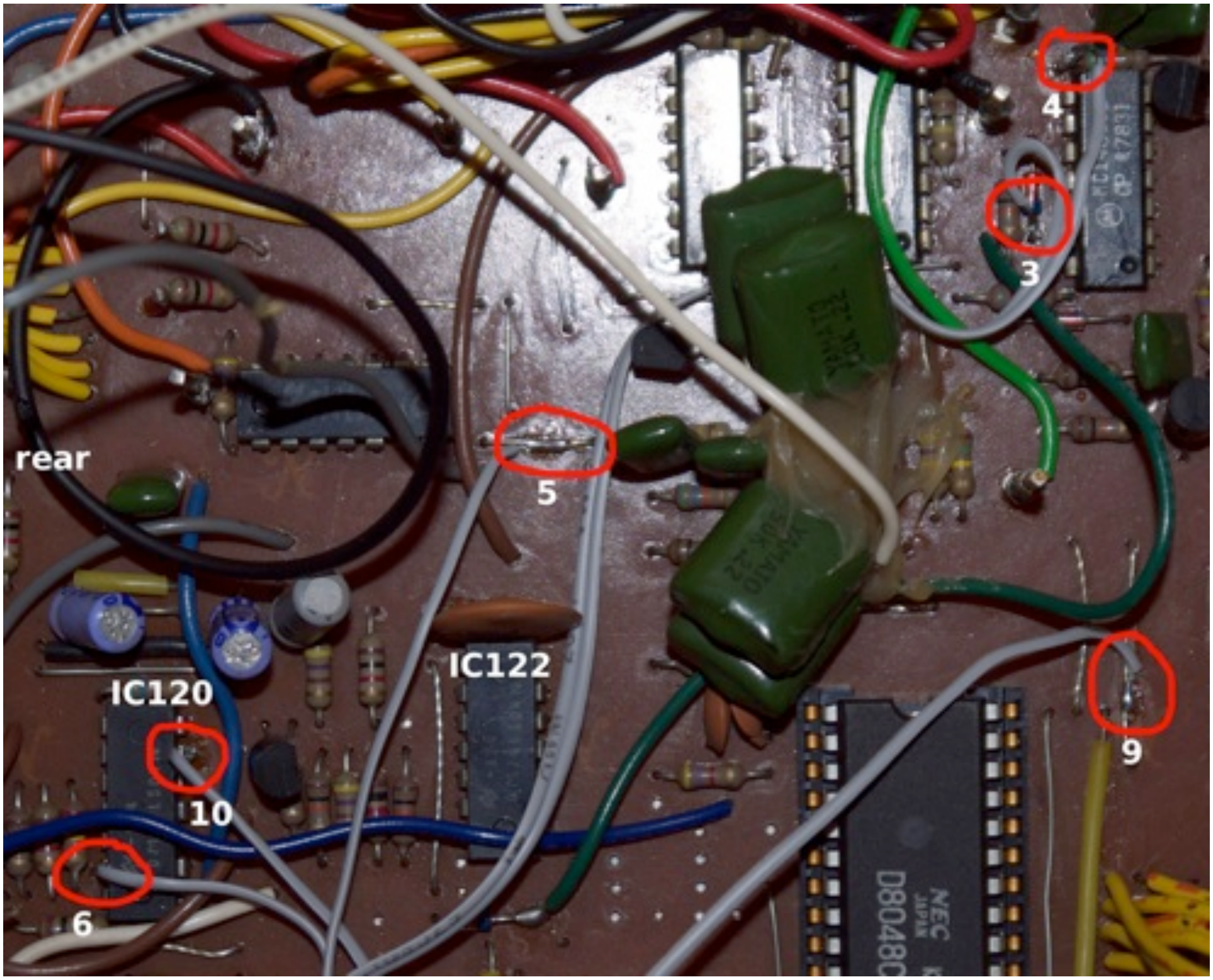
GL-9B board trigger cable installation



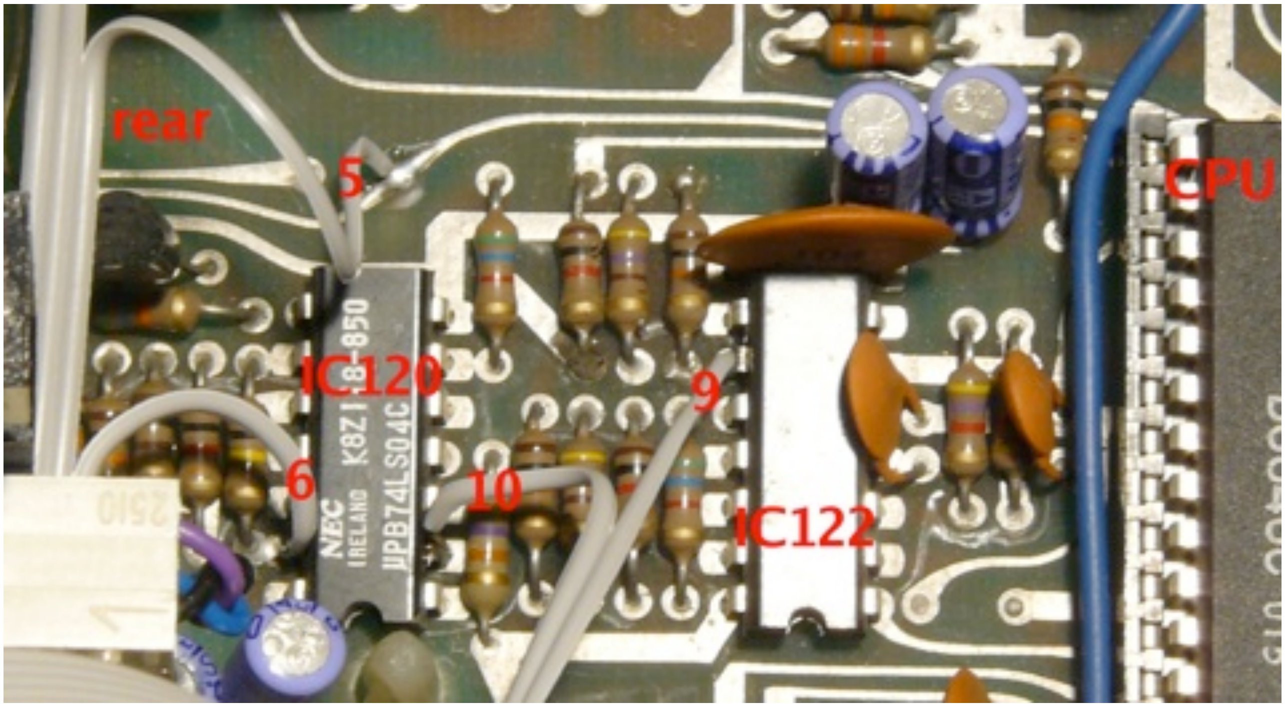
Installing the control cable, CN102:

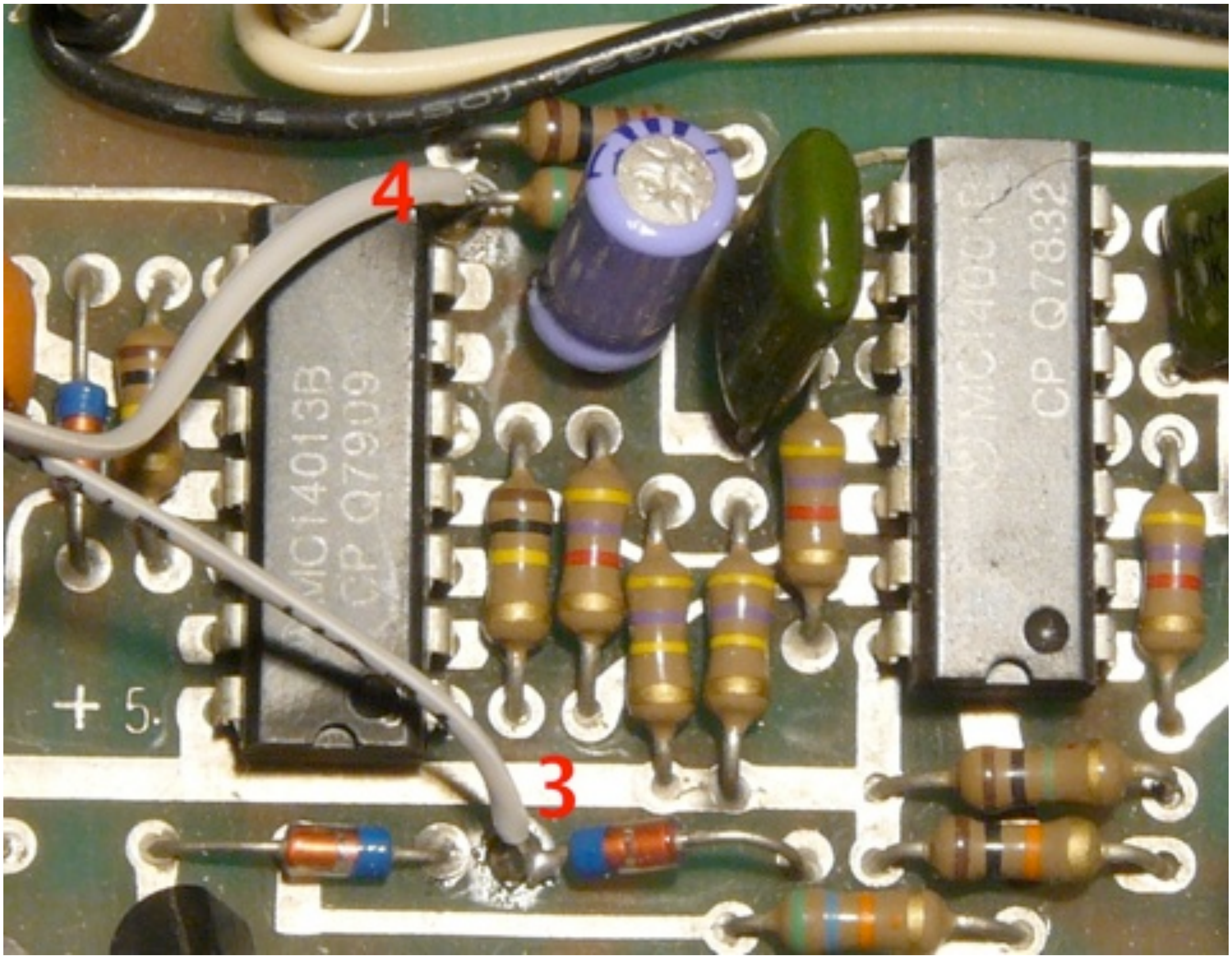
- Carefully check the 10 pin cable and make familiar with the wire numbering. Pin 1 again is the wire with the red stripe. 1, 2, 7 and 8 go to the peripheral connector, 3-6, 9 and 10 are wires of different lengths to go to board specific places.
- Solder wire 6 to IC120, pin 13 (GL-9 and GL-9B)
- Solder wire 10 to IC120, pin 6 (GL-9) or pin 2 (GL-9B)
- Solder wire 9 to the jumper next to the CPU as shown in the photo (GL-9) or IC122, pin 9 (GL-9B)
- Solder wire 5 to the jumper next to the removed R201 (GL-9) or to the trace (GL-9B) as shown in the photo
- Solder wire 3 to the cathode of D101 (GL-9) or one of the 2 diodes where they point to each other (GL-9B)
- Solder wire 4 to R120, the resistor right next to the IC, as shown in the photo below (GL-9 and GL-9B)

Control cable installation on GL-9 board



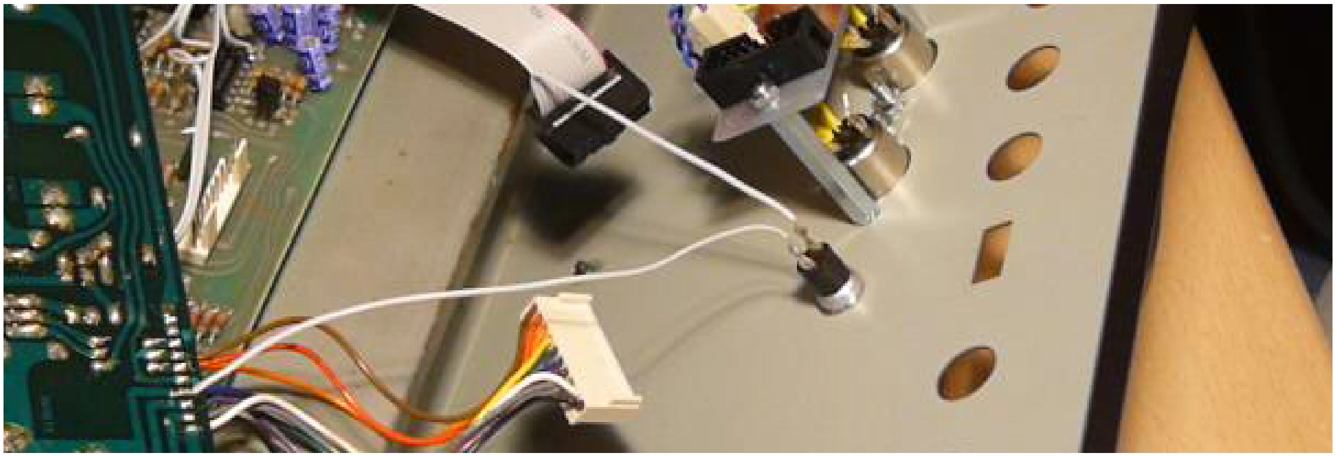
Control cable installation on GL-9B board





Connecting the configuration button:

- Solder the last wire of the 14 pin cable, pin 14, to the configuration button. Depending on where you installed this button, you may cut the wire for a more suitable length.
- Solder the other end of the configuration button to a GND spot, the connector board, pin 4, is suitable for this (see photo, you can locate it by following the large GND traces of the connectors. On GL-9, you can use the GND wire of the EXT CLOCK connector, next to the transistor. If you cut the wire for the configuration button, you can use the part you cut off for this.



Putting the unit back together:

- Re-attach the rear panel, tighten the 4 screws
- Make sure all connectors on the main board are in place
- Connect the CN101 (14 wire) cable to the MIDItriggs board, the red wire shows upwards, the cable shows away from the board
- Connect the CN102 (10 wire) cable to the MIDItriggs board, the red wire shows downwards, the cable shows away from the board.
- Make sure both connectors are perfectly in place and not off by one pin or so.
- Mount the connector board back in place, using all the washers you took care of during disassembly. The thin metal washer on the connector, followed by a plastic washer, with the edge pointing to the rear panel, then comes the rear panel, another plastic washer with the edges pointing to the rear panel again, a thick metal washer and the nut.
- Again make sure all connectors and cables are in place before powering up the unit

First tests:

- Power up the unit, dial Tempo to about 12 o‘clock, select any rhythm, turn off all cancel switches, turn up the volume, turn off the fade-in and fade-out and push start. The unit should behave as usual, react on start/stop like before.
- Connect the unit‘s MIDI out to any MIDI gear able to produce sound when receiving notes in the range C1-A#1 and push start again. The Rhythm should appear on the MIDI out.
- Push the 2 rhythm cancel buttons, connect a MIDI keyboard to the MIDI in and push start. You should be able to play 10 instruments with C1-A1, while A#1 is the accent trigger, which is a bit difficult to hit the right time to hear the accent.
- Push stop on the unit, push and hold the configuration button for longer than 1 second. You should hear a Pattern like this: BD.BD.BD.-.BD.-.-.-
- Power off the unit, put the wooden case back in place, you‘re done with this installation.

License

This project is free according to the GNU General Public License Version 3. The project consists of the firmware source code, the schematics, the PCB layouts and this documentation, while all those files are covered by the following license:

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
    <one line to give the program's name and a brief idea of what it does.>
    Copyright (C) <year>  <name of author>

    This program is free software: you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program.  If not, see <http://www.gnu.org/licenses/>.
```

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
    <program>  Copyright (C) <year>  <name of author>
    This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
    This is free software, and you are welcome to redistribute it
    under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.