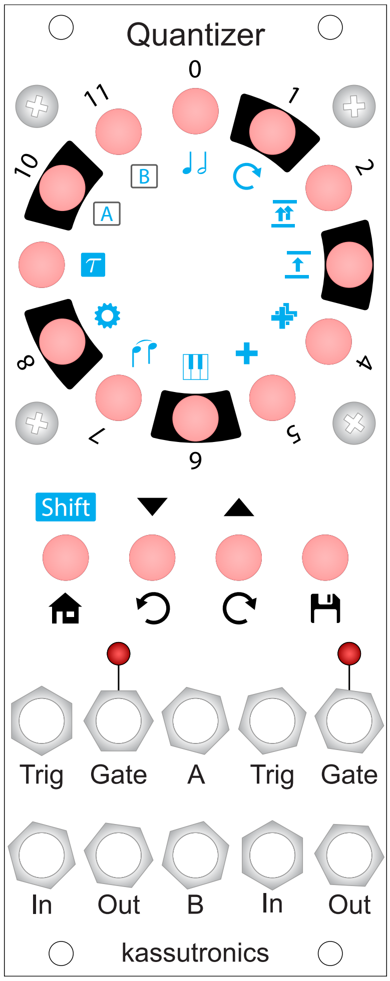
Description

A quantizer is a nearly essential utility in a modular synth. By quantizing any incoming control voltage to the 1V/Oct standard, the Quantizer allows your VCOs to play in musical tune. With two channels and flexible custom musical scales, the Quantizer can create two voices or simple chords. Operating either free-running or triggered from an external rhythm, the Quantizer can generate all kinds of random and pseudorandom melodies, or simply quantize the voltage from a variable source such as a ribbon controller. The Gate outputs indicate whenever a new note has been quantized.

While the Quantizer is a digital module, the user interface is designed to retain much of the immediacy that makes analog synthesizers so much fun. The musical scale is set using a rather unique circle of illuminated buttons representing the semitones in an octave, which can be enabled or disabled by pressing the buttons. Breaking free from traditional piano keyboard layouts, the circle layout avoids silly complexities in standard musical theory and encourages experimentation. The rotate buttons and allow you to change the root note of the scale, and further highlight the circular nature of musical harmony. The module automatically retains its state when powered down.

More advanced options are supported by a single menu layer, accessed with the Shift button and explained by the blue icons. These options allow different kinds of transpositions, gate length changes, and above all flexible CV control of almost any parameter using the A and B CV inputs. Under CV control the Quantizer turns into a dynamic composition element.

For a more detailed description of the modules operation, see the separate User manual. This document contains all information needed to build a Quantizer for yourself. Please read the Build instructions and Bill of materials carefully before starting your build!



Front panel

Features

- Two channel quantizer
- External trigger inputs and free-running mode
- Easy selection of scales
- Flexible CV control
- Save/recall of scales and autosave of current state
- 10hp Eurorack format
- Arduino-compatible open-source code
- Mostly through-hole construction

Power supply recommendations

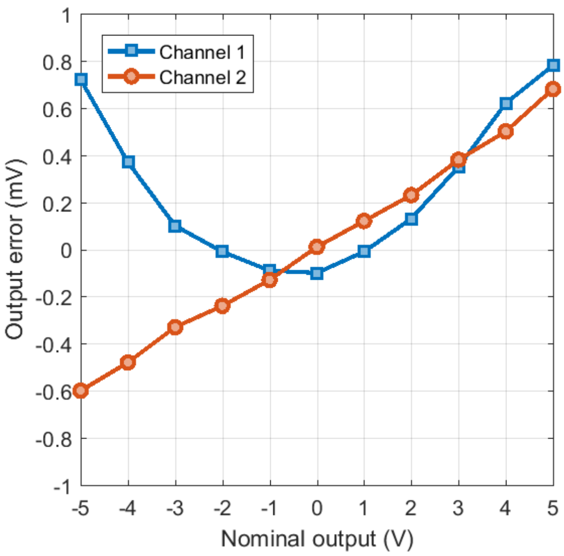
Symbol	Parameter	Voltage	Current (typ)
V_{CC}	Positive supply	+12 V	70 mA
V_{EE}	Negative supply	−12 V	30 mA

Typical performance characteristics

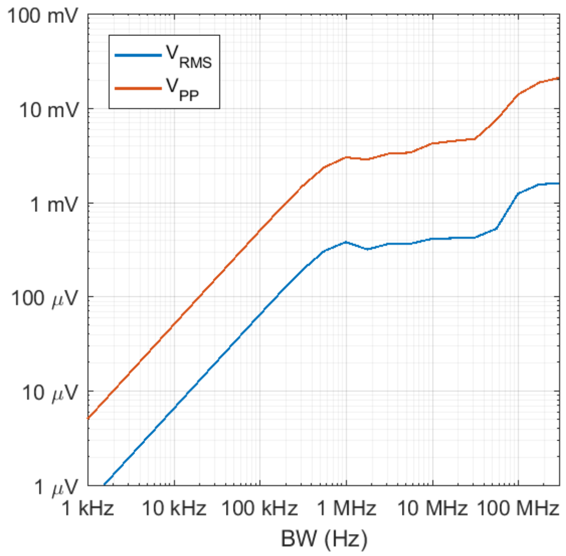
Parameter	Typical value	Unit
Output characteristics		
Output voltage range	±5.33	V
Output voltage step size	83.3	mV
Output voltage accuracy and linearity[1]	±1	mV
Ripple and noise (20 MHz bandwidth)	4	mVpp
Input characteristics (In, A and B jacks)		
Input voltage range	±5.33	V
Hysteresis	10	mV
Timing characteristics		
Sampling rate, per channel	4.8	kHz
Output rise time, t_r	250	μs
Delay, trigger to sample[2], t_{ts}	150 – 350	μs
Delay, sample to gate high[3], t_{sg}	450	μs
Minimum trigger pulse length	< 10	μs

[1] Using well-matched resistors, see build instructions.
[2] The Trigger delay function can be used to add an extra delay up to 11 ms.
[3] The output changes voltage during this delay, and is settled completely before the gate goes high.

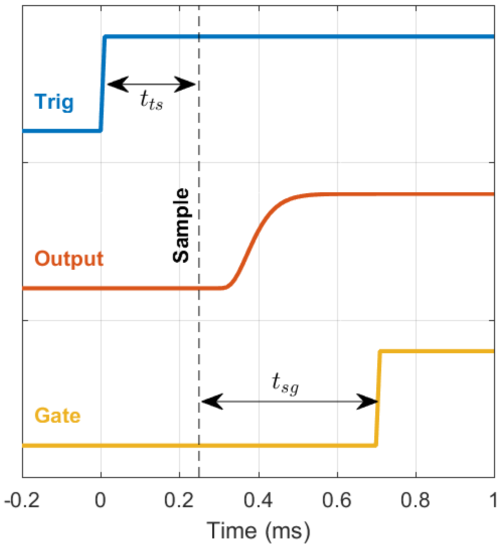
Output error and linearity



Output ripple and noise vs bandwidth



Timing diagram, triggered



Build instructions

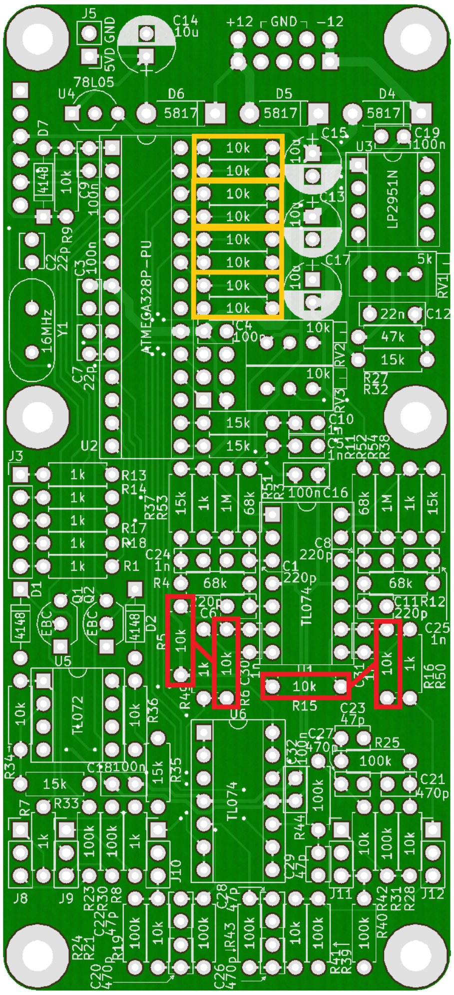
Main PCB

The main board contains the power circuitry, micro controller, analog input and output circuitry and trigger/gate circuitry, all in through-hole parts.

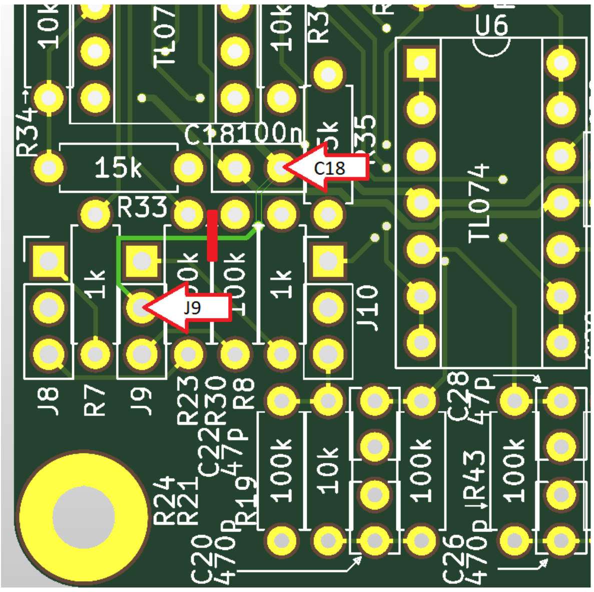
1. For accurate 1V/Oct output on both output channels, resistors R5, R6, R15 and R16 should be matched to 0.1% or better. You can either buy 0.1% tolerance resistors, or measure a bunch of normal 1% resistors and choose two closely matched pairs R5 = R6 and R15 = R16. Their value should agree within 10Ω.

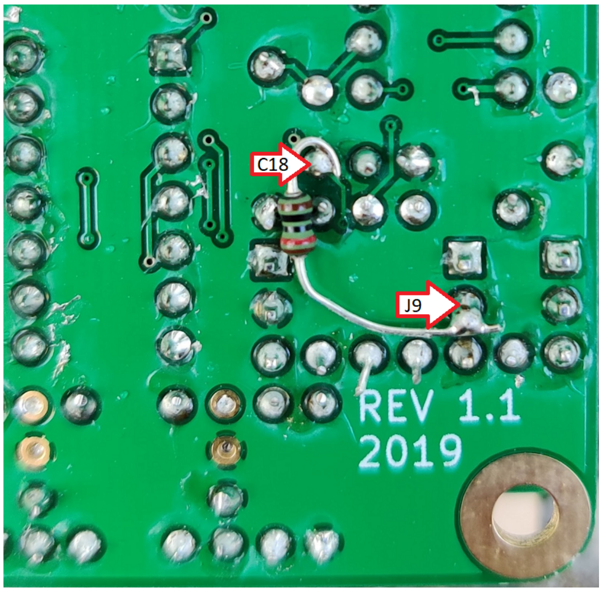
2. Optionally the input resistor dividers could also be matched, which will improve the tracking at the extreme ends of the scale (around +5V and -5V). If you choose to do this step, they should be matched in pairs R22=R20, R29=R26, R45=R47, R46=R48. Note, however, that matching on these resistors is much less critical since the inputs will be quantized anyway, and even standard 1% tolerance resistors are fine for most uses.

   The following image shows the locations of matched resistor pairs. Red boxes indicate pairs that **must** be matched for output performance, and yellow boxes indicate pairs that can be matched for best input performance.



3. J1 and J2 are for programming and debugging, and should be soldered on the bottom side of the PCB. In most cases J2 will never be used.
4. The eurorack power header should also be soldered on the bottom of the PCB. Either a shrouded or unshrouded header can be used.
5. I recommend using sockets for all ICs, especially for the microcontroller. Before inserting the microcontroller, follow the power up test described in the next section.
6. **Recommended modification on PCB revision 1.1:** Some users have reported the switch contact of the Trig A and Trig B jack sockets to momentarily short the +12V supply to ground when inserting or removing a cable. To avoid this from causing any issues, it is recommended to do the following modification. First, cut the trace between C18 and J9 as marked by the red line on the left figure below. Then, solder a 10k resistor between these points. This is easiest done ont he back side, as shown in the photo on the right. This issue is only present in board revision 1.1 and has been adressed in later revisions.





Front panel PCB

The front panel PCB contains all buttons and jacks, as well as I/O expansion ICs and components needed to read the buttons and drive the LEDs. Some of the components are surface mount devices. The SMD pads were specifically designed to allow easy hand soldering with a normal soldering iron.

Pay special attention to the orientation of these components:

1. For SMD ICs pin 1 is marked on the silkscreen with a circle inside the component, and an extended line next to the solder pad. On the chip itself markings may vary, but pin 1 is always in the bottom left corner when the text is in normal reading orientation.
2. The PB6149L buttons have 6 legs, but there are 7 holes in the PCB footprint. The buttons must be rotated such that a small pastic alignment stub fits into the extra hole in the PCB. Make sure to completely insert the buttons into the PCB before soldering.
3. The flat side and shorter leg of the 3mm LEDs correspond to the square pad on the PCB.

4. All pin sockets should be soldered on the bottom side of the PCB such that they connect with the corresponding pin headers on the main PCB.

5. J213-J217 are very close to the Thonkiconn jacks on the other side, getting all these components in place is a bit of a puzzle. I recommend to first solder J214-J217 (but not yet J213), then the Thonciconns, and finally J213.

6. The leads of S204-S206 should be trimmed fairly short to make space for the trimpots RV1-RV3 when the boards are connected together.

Powerup test

Before inserting the microcontroller and assembling the two boards together, follow these steps.

1. Inspect the boards for shorts or bad joints.

2. Measure the resistance between the +12V and GND and between the GND and -12V power connections. The values will change while measuring due to the large capacitors, but should increase to at least the kΩ range after several seconds.

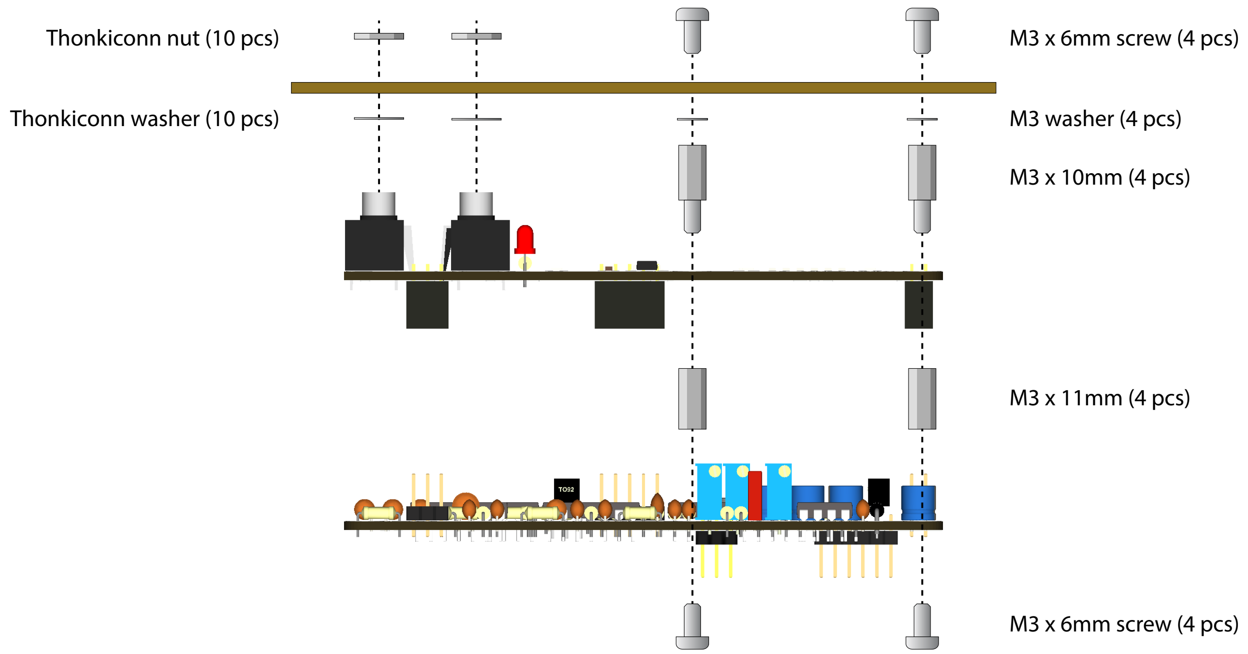
3. With U3 (LP2951N) inserted, but not yet the microcontroller, apply power to the main board. Measure the 5V3 power rail on pin 4 of J2 (backside of the PCB). Adjust trimmer RV1 until the voltage is close to 5.34V.

4. With the main board still powered up, measure the 5VD power rail on J5 (frontside of the PCB). It should read between 4.8V and 5.2V and is not adjustable.

5. Disconnect power and insert the ATMega328P microcontroller.

Mechanical assembly

The front panel must be connected to the front panel PCB with M3 standoffs to ensure proper operation of the push buttons. To obtain the correct spacing of 10.3 to 10.5 mm, use 10mm standoffs + one M3 washer. Similarly, washers must be used between the Thonkiconn jacks and the front panel. The recommended mechanical assembly is shown in the following diagram.



Programming the microcontroller

If you have a pre-programmed microcontroller, skip this section.

The board is designed to be compatible with the Arduino Uno, and can be easily programmed from the Arduino software. The chip can be programmed through the ICSP header J1.[1]

Getting started

First, make sure you have a recent version of the Arduino IDE. I used version 1.8.5 while developing the module. Download the latest Quantizer firmware source code from github and open it up in Arduino. Press Verify to compile the code, and check that it compiles successfully.

Programming with the ICSP header

Here, I give the steps needed for programming briefly. This is exactly the same as programming an Arduino via it's ICSP header, so plenty of instructions can be found online in case you are having trouble.

For programming, you need one of the programmers that is supported by Arduino. I used the USBasp, which is cheaply available from Ebay, but there are several others that work pretty much in the same way. See the USBasp website for drivers and instructions how to install them.[2]

The quantizer board has a 6-pin ICSP header, with the standard pinout also written on the PCB. If your programmer only comes with a 10-pin header, you need to buy or make an adapter cable to the 6-pin version. Now come the important steps:

[1] When the Arduino bootloader is installed on the chip, the board can also be programmed through the serial header J2. This option is mostly useful for debugging and will not be discussed here.

[2] To get the USBasp working on Windows 10 I had to choose the libusbK driver; others report using the libusb-win32 driver successfully. These are all included in the Zadig software linked from the USBasp site.

1. **Remove the *Power device* jumper from the programmer**.
2. Power up the Quantizer board from a eurorack power supply.
3. Connect the 6-pin ICSP header to J1. Pin 1 is the bottom-right pin when looking at the back of the module, and is labeled MISO.
4. In the Arduino IDE, choose Tools → Programmer and select the programmer you are using.
5. Choose Tools → Burn Bootloader.[3]
6. Choose Sketch → Upload Using Programmer.

If all goes well, the scale lights now light up with a Major scale and the quantizer is working!

Note: Programming the microcontroller via the ISCP header erases the EEPROM, resetting the Quantizer to factory defaults.

Alternative: programming via the serial port

Once the Arduino Bootloader has been uploaded (using the Burn Bootloader command described above or by starting with a preprogrammed Arduino chip), the Quantizer can also be programmed via the serial header J2. This way the EEPROM doesn't get reset, which is especially useful for firmware development.

The serial header can be connected to any 5V-compatible USB-to-serial adapter. Connect RX (adapter) to TX (quantizer), and vice versa TX (adapter) to RX (quantizer). Connect also GND and DTR, but **do not connect 5V3**. Power the board through a eurorack power source, and upload the sketch with the normal Upload button.

Calibration

To calibrate the output to 1V/octave, a multimeter with reasonably good precision is needed. I recommend a 4-digit or better multimeter; however, if you only have a 3-digit it will still work. You can always do a more precise calibration later if needed!

- Set the multimeter to DC Volt and connect it to the left Out jack.
- Press Shift + to enter keyboard mode. Button 0 should light up, and the output should be around 0V.
- Adjust RV2 (the middle trimmer when looking from the side) such that the output voltage is exactly 0.000V, within a few mV.
- Press ▲ five times, it will flash quickly. The output voltage should be close to 5V. Adjust RV1 (the trimmer near the board edge) to make it exactly 5.000V, within a few mV.
- Go back down five times with ▼ to check the 0V again, and adjust if needed. If you made an adjustment, also check the 5V again. Finally, check the -5V output.
- Now, connect the multimeter to the right Out jack, and go back to octave zero (both ▲ and ▼ are not lit up). Adjust RV3 to set this voltage to exactly 0.000V (within a few mV).

[3] If you get an error "target doesn't answer" you may need to connect the *Slow* jumper on the programmer during this step.

- Finally, check the high and low octaves of the right output. They should be close to correct and normally no further adjustments are needed.

Note: If the resistor matching was done accurately, all octaves on both outputs can be matched to within about 2 mV. However, don't fuss too much about the accuracy here. An error of up to 5mV will be comparable to the tracking accuracy of a typical analog VCO such as my VCO 3340, and is totally acceptable. For reference, 1mV error corresponds to 1.2 cents detuning.

Bill of materials

Main PCB

Qty	Designator	Value	Note
2	C2, C7	22p	C0G ceramic, pitch 2.5mm
4	C22, C23, C28, C29	47p	C0G ceramic, pitch 2.5mm
4	C1, C6, C8, C11	220p	C0G ceramic, pitch 2.5mm
4	C20, C21, C26, C27	470p	C0G ceramic, pitch 2.5mm
6	C5, C10, C24, C25, C30, C31	1n	C0G/X7R cer., pitch 2.5mm
1	C12	22n	Film, pitch 5.0mm
7	C3, C4, C9, C16, C18, C19, C32	100n	X7R ceramic, pitch 2.5mm
4	C13, C14, C15, C17	10u	Electrolytic, min. 25V, diameter 6.3mm, pitch 2.5mm, **max. height 9mm**[1]
3	D1, D2, D7	1N4148	THT
3	D4, D5, D6	1N5817	THT
1	J5	1x2	Pin header, pitch 2.54mm
5	J8, J9, J10, J11, J12	1x3	Pin header, pitch 2.54mm
1	J3	1x5	Pin header, pitch 2.54mm
1	J2	1x6	Pin header, pitch 2.54mm
1	J1	2x3	Pin header, pitch 2.54mm
1	J4	2x5	Pin header, pitch 2.54mm
2	Q1, Q2	2N3904	TO-92, square pad = emitter
11	R1, R7, R8, R13, R14, R17, R18, R49, R50, R53, R54	1k	1/4W or 1/8W, 1% metal film
8	R9, R21, R28, R34, R36, R41, R42, + mod (**see build instructions**)	10k	1/4W or 1/8W, 1% metal film
4	R5, R6, R15, R16	10k	**0.1% matched pairs**
8	R20, R22, R26, R29, R45, R47 R46, R48	10k	1% or 0.1%, see instructions
7	R2, R10, R32, R33, R35, R37, R38	15k	1/4W or 1/8W, 1% metal film
1	R27	47k	1/4W or 1/8W, 1% metal film
4	R3, R4, R11, R12	68k	1/4W or 1/8W, 1% metal film
10	R19, R23, R24, R25, R30, R31, R39, R40, R43, R44	100k	1/4W or 1/8W, 1% metal film
2	R51, R52	1M	1/4W or 1/8W, 1% metal film
2	RV2, RV3	10k	Trimmer 3296X or T910-X
1	RV1	5k	Trimmer 3296X or T910-X
1	U4	78L05	TO-92
1	U2	ATMega328P	ATMega328P-PU, DIP-28
1	U3	LP2951ACN	DIP-8, ON Semi
1	U5	TL072	DIP-8
2	U1, U6	TL074	DIP-14
1	Y1	16MHz	HC-49/S, HC-49/US or HC-49-4H crystal

[1] Nichicon UST1H100MDD or equivalent, pay close attention to dimensions.

Front panel PCB

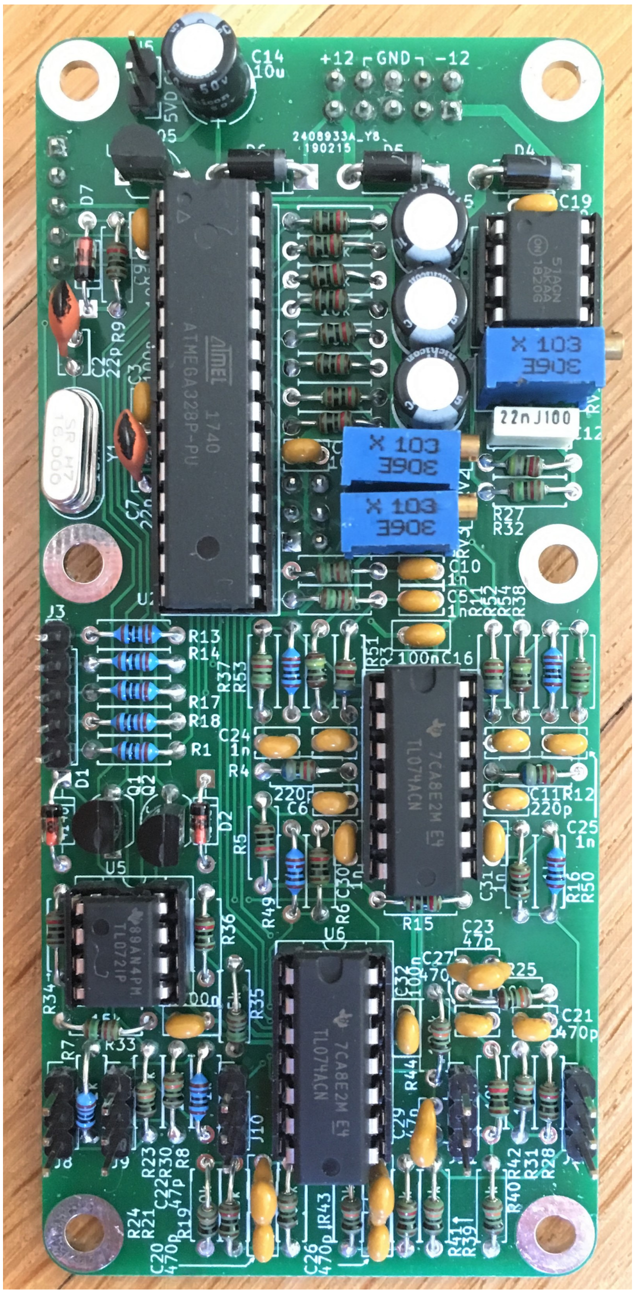
Qty	Designator	Value	Note
4	C201, C202, C203, C204	100n	0805 SMD, X7R ceramic
2	D201, D202	LED	3mm, square pad = negative
1	J212	1x2	Socket strip, pitch 2.54mm
5	J213, J214, J215, J216, J217	1x3	Socket strip, pitch 2.54mm
1	J201	1x5	Socket strip, pitch 2.54mm
10	J6, J7, J204, J205, J206, J207, J208, J209, J210, J211	Thonkiconn	Thonkiconn
16	R202, R204, R206, R208, R210, R212, R214, R216, R218, R220, R222, R224, R226, R228, R230, R232	2.2k	0805 SMD
2	R233, R234	10k	1/4W or 1/8W THT
16	R201, R203, R205, R207, R209, R211, R213, R215, R217, R219, R221, R223, R225, R227, R229, R231	100k	0805 SMD
16	S201, S202, S203, S204, S205, S206, S207, S208, S209, S210, S211, S212, 213, S214, S215, S216		Highly PB6149L-1[1]
2	U201, U203	74HC165	SOIC-16
2	U202, U204	74HC595	SOIC-16

[1] Highly PB6149L-x, where x indicates the LED color, is available from TME.

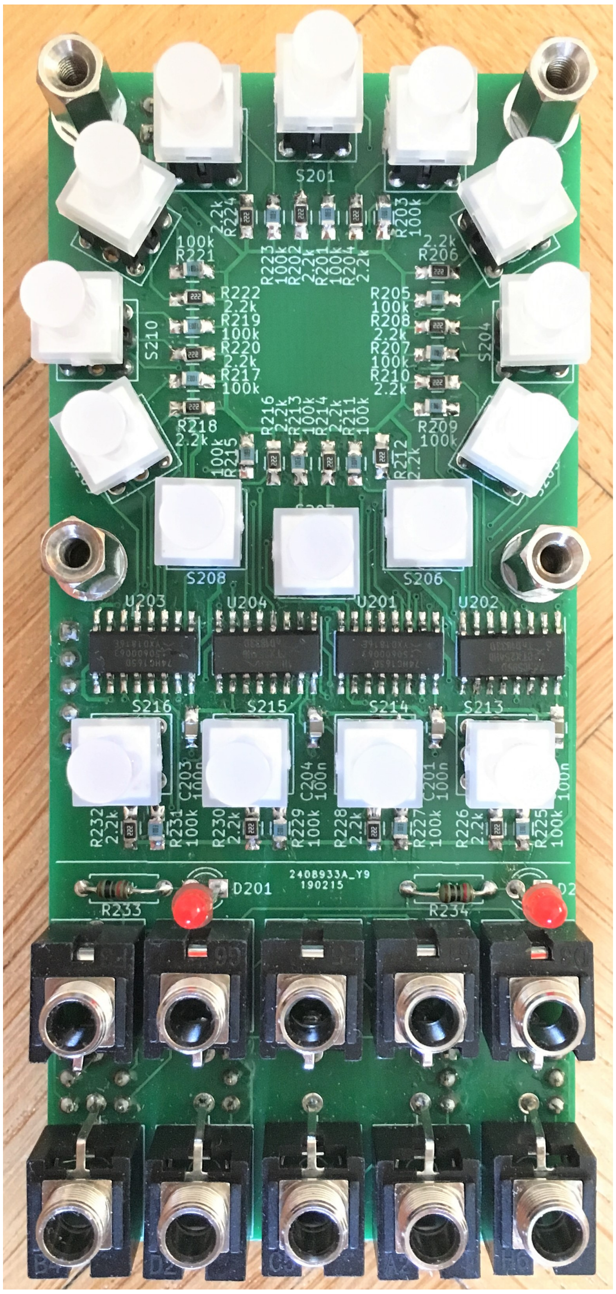
Mechanical parts

Qty	Item
4	M3 hex stand-offs, 10mm long, one male and one female thread
4	M3 hex stand-offs, 11mm long, two female threads
4	M3 washers
8	M3 screws, cheese head
10	Washers for Thonkiconn jacks

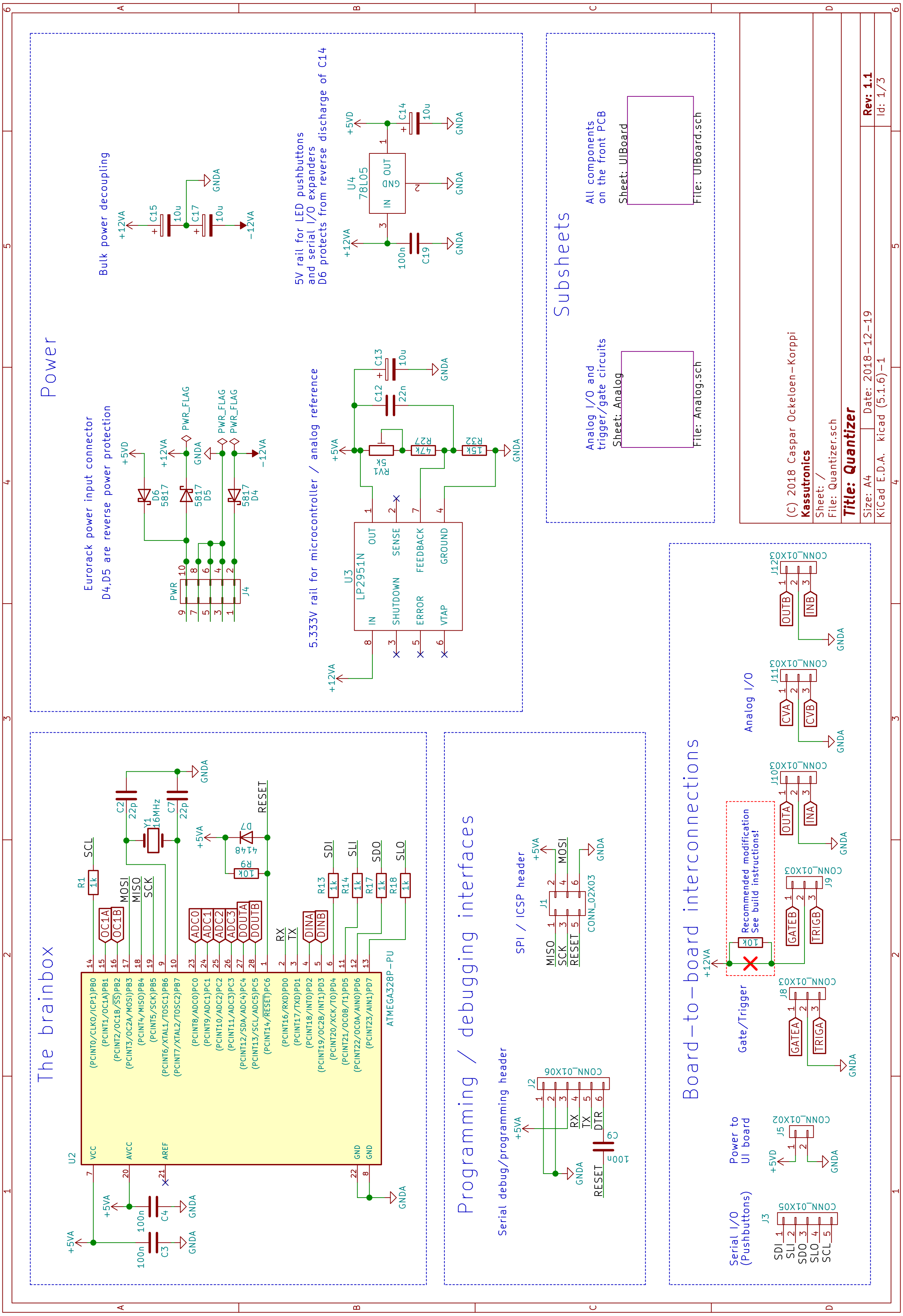
Board view

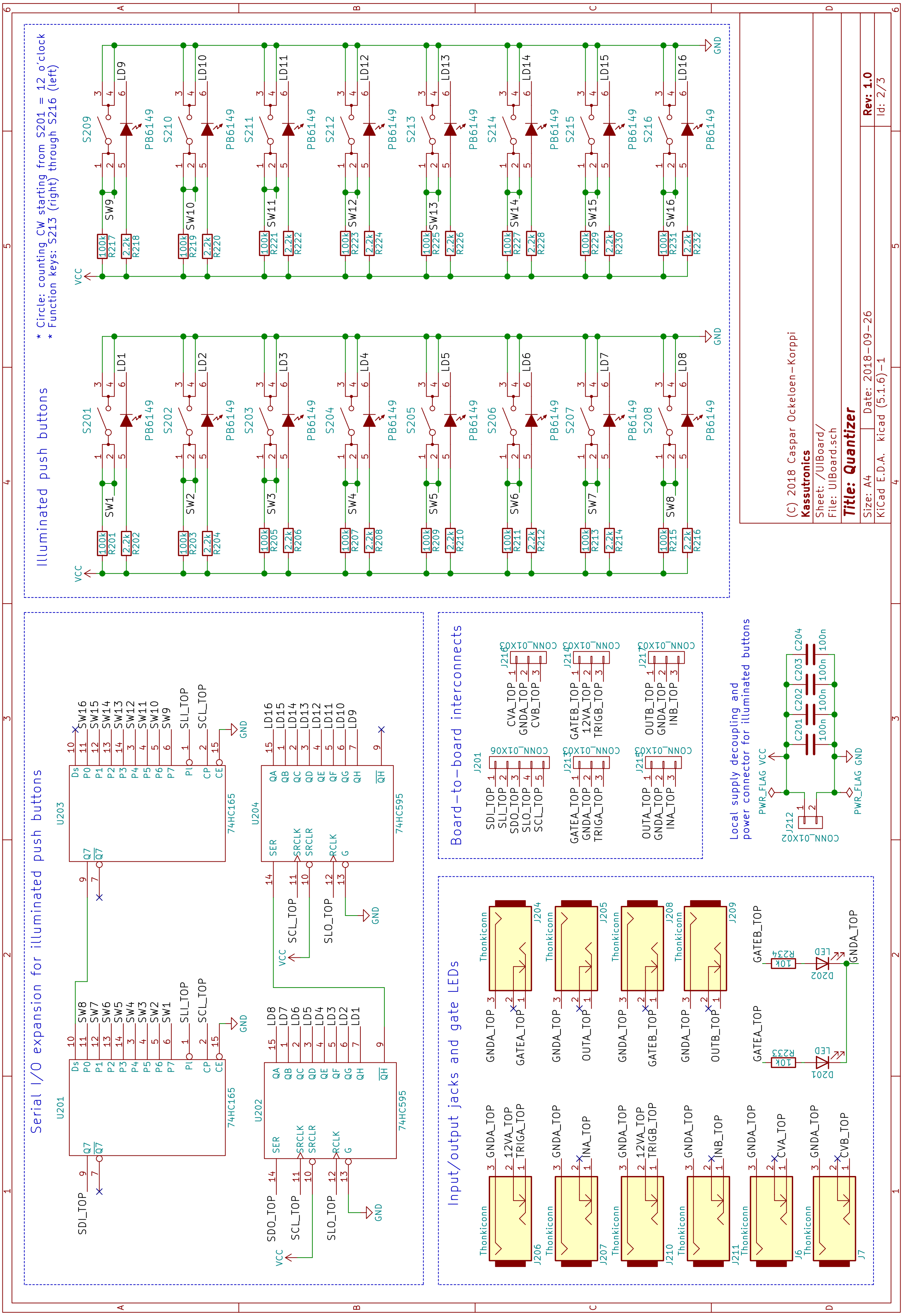


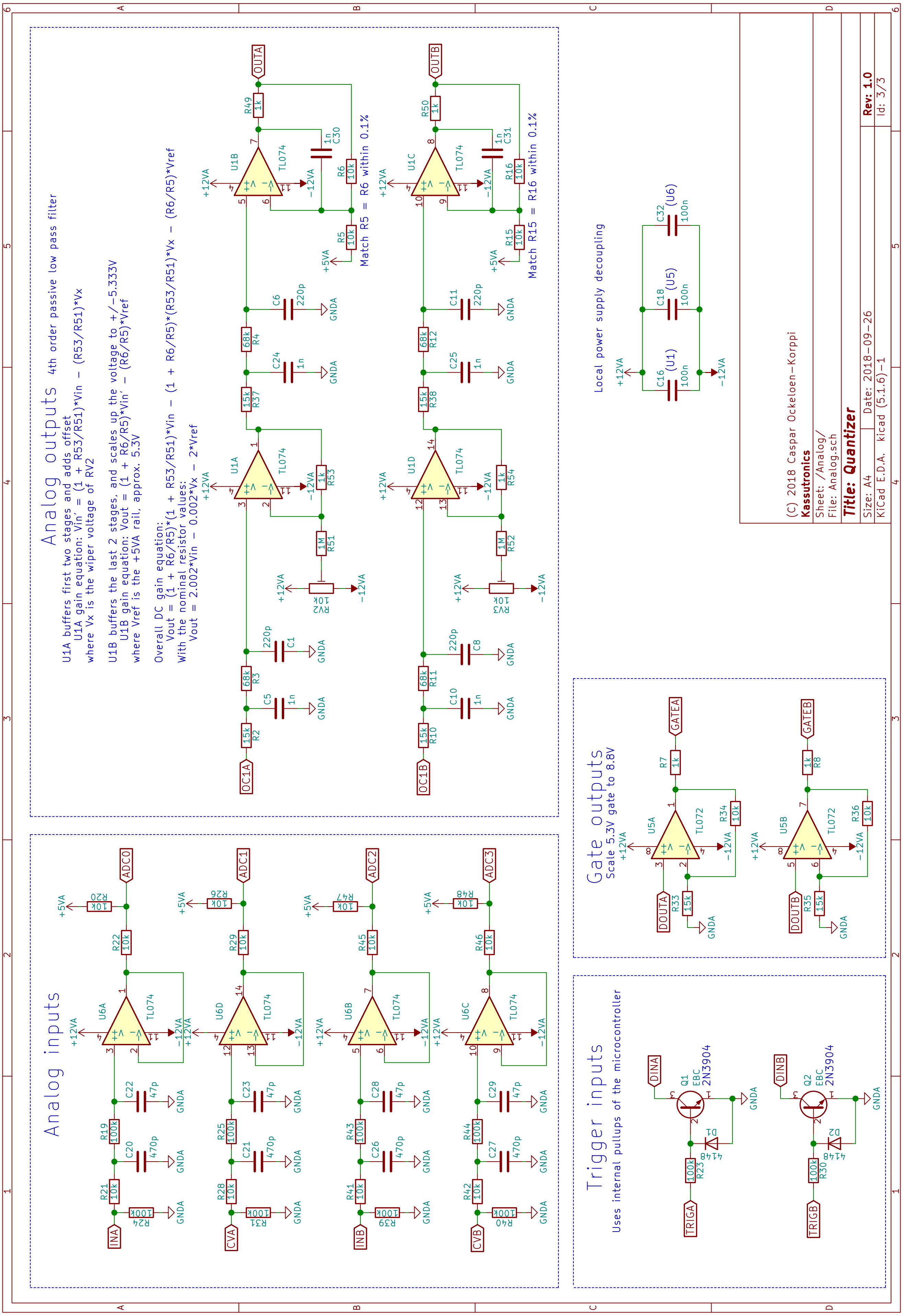
Main PCB



Front panel PCB


Power
Bulk power decoupling
Eurorack power input connector
D4,D5 are reverse power protection
5V rail for LED pushbuttons
and serial I/O expanders
D6 protects from reverse discharge of C14
5.333V rail for microcontroller / analog reference
Subsheets
Analog I/O and trigger/gate circuits
Sheet: Analog
File: Analog.sch
All components on the front PCB
Sheet: UIBoard
File: UIBoard.sch
The brainbox
ATMEGA328P-PU
Programming / debugging interfaces
Serial debug/programming header
SPI / ICSP header
Board-to-board interconnections
Serial I/O (Pushbuttons)
Power to UI board
Gate/Trigger
Recommended modification
See build instructions!
Analog I/O
(C) 2018 Caspar Ockeloen-Korppi
Kassutronics
Sheet: /
File: Quantizer.sch
Title: Quantizer
Size: A4
Date: 2018-12-19
Rev: 1.1
KiCad E.D.A. kicad (5.1.6)-1
Id: 1/3


Illuminated push buttons
* Circle: counting CW starting from S201 = 12 o'clock
* Function keys: S213 (right) through S216 (left)
Serial I/O expansion for illuminated push buttons
Board-to-board interconnects
Input/output jacks and gate LEDs
Local supply decoupling and power connector for illuminated buttons
(C) 2018 Caspar Ockeloen-Korppi
Kassutronics
Sheet: /UIBoard/
File: UIBoard.sch
Title: Quantizer
Size: A4
Date: 2018-09-26
Rev: 1.0
KiCad E.D.A. kicad (5.1.6)-1
Id: 2/3


Analog outputs 4th order passive low pass filter
U1A buffers first two stages and adds offset
U1A gain equation: Vin' = (1 + R53/R51)*Vin − (R53/R51)*Vx
where Vx is the wiper voltage of RV2
U1B buffers the last 2 stages, and scales up the voltage to +/−5.333V
U1B gain equation: Vout = (1 + R6/R5)*Vin' − (R6/R5)*Vref
where Vref is the +5VA rail, approx. 5.3V
Overall DC gain equation:
Vout = (1 + R6/R5)*(1 + R53/R51)*Vin − (1 + R6/R5)*(R53/R51)*Vx − (R6/R5)*Vref
With the nominal resistor values:
Vout = 2.002*Vin − 0.002*Vx − 2*Vref
Match R5 = R6 within 0.1%
Match R15 = R16 within 0.1%
Local power supply decoupling
Analog inputs
Gate outputs
Scale 5.3V gate to 8.8V
Trigger inputs
Uses internal pullups of the microcontroller
(C) 2018 Caspar Ockeloen-Korppi
Kassutronics
Sheet: /Analog/
File: Analog.sch
Title: Quantizer
Size: A4 Date: 2018-09-26 Rev: 1.0
KiCad E.D.A. kicad (5.1.6)-1 Id: 3/3

Circuit description

The Quantizer contains mostly fairly standard digital circuitry. However, a few unusual design choices were made, as discussed below.

Analog outputs

For the pitch CV outputs, the Quantizer uses two pulse width modulated (PWM) signals that are filtered and scaled to form analog signals. While PWM is often mostly considered as a cheap alternative to other DAC architectures, it has a specific advantage. The output voltage is directly derived from the microcontrollers (MCU) clock crystal, which provides very accurate timing (low jitter), which converts to excellent linearity in the voltage domain. As shown in the performance characteristics, the nonlinearity is below 0.01% of full scale. This vastly outperforms popular low-cost DACs such as the MCP4822, which has a typical nonlinearity of 0.1%, and is comparable to high-performance DACs such as the DAC8565. A PWM output is also extremely easy to use in software, since updating the value only requires writing to a register of the MCU.

The challenges with PWM outputs are accuracy of the reference voltage, properly filtering the PWM signal, and a trade-off between resolution and PWM frequency. The latter was no big issue in the Quantizer, since we only need steps of whole semitones, or 12 steps per octave. Using a 7-bit DAC (128 steps), a range of more than 10 octaves is reached. Using the 16 MHz clock to drive the PWM we end up with a 125 kHz PWM frequency, well above audio range and allowing for quick voltage changes.

The reference voltage of the PWM DAC is essentially the supply voltage of the MCU, so a well-regulated, low-drift and accurately tunable supply is needed. This is implemented with U3, a LP2951N adjustable regulator. The supply voltage is tuned to calibrate the DAC output scale. By setting the supply voltage to a special value of nominally 5.333V (higher than the usual 5V, but within spec for the MCU), the PWM output gets scaled to exactly 24 steps per Volt. The analog output circuit described next amplifies this by a factor 2, reaching the final required 12 steps per Volt. Note that in practice the supply voltage must be tuned slightly higher around 5.34V to compensate for the on-resistance of the MCU output transistors.

Finally, carefully filtering the PWM signal to a DC output is essential. In the Quantizer I chose a 4-pole passive RC filter, consisting of two 2-pole sections with a buffer in between and at the end. It should be noted that filtering the 125 kHz PWM carrier frequency is fairly straight forward, choosing a trade off between cutoff frequency, number of poles and residual ripple. However, filtering the high-frequency harmonics is more difficult. Even the for modern standards slow ATmega MCU has very fast signal edges, with frequency content up to about 100 MHz. For filtering the high frequency content a passive filter performs somewhat better than for example a Sallen-Key topology, because in the latter the opamp has to actively keep up with the high frequency content. A passive filter also has naturally exactly unity gain, simplifying the output scaling setup in this design. Even so, stray electromagnetic coupling allows some high-frequency content to pass through, as shown in the figure *Output ripple and noise vs bandwidth.*

The first buffer opamp in each channel (U1A and U1D) has an offset trimmer connected with a gain of 1/1000, allowing to trim the total offset to zero. The offset, which when unadjusted is typically below 10 mV, originate partly from the opamps, and partly from asymmetric on-resistance of the MCU output transistors. Trimming the offset to zero means you can connect the Quantizer to a VCO without affecting the tuning.

Finaly output scaling is done by the last opamps, U1B and U1C, using matched resistors to set a gain of exactly 2. These opamps also have in-the-loop frequency compensation to allow significant capacitive loads without affecting DC accuracy.

Analog inputs

The four analog inputs, two channel inputs and two CV inputs, all use identical circuitry. They use the internal 10-bit ADC of the MCU, which was found to be good enough in terms of noise and linearity. The ADC is freely running at the maximum recommended rate of 52μs per sample, resulting in a 4.8 kHz sample rate per channel when divided over the four inputs.

Each input is filtered by a two-pole passive filter. The cutoff frequency of each pole is set at 33 kHz, above audio rate. Hence, these filters in most cases do nothing! However, the actual sampling time of the DAC is only a few μs, and the DAC is able to pick up high-frequency noise if presented with an noisy CV source. In prototypes I found I had plenty of noisy CV sources around, in particular poorly filtered PWM outputs from other hardware. The input filters mitigate this noise.

You may have noticed that the filter frequency is also well above the Nyquist frequency of half the sampling frequency. This is a deliberate choice. The Quantizer does not aim to faithfully reconstruct the input signal. Rather, it acts more like a sample-and-hold circuit, sampling the signal at an instant. This analogy works in particular when using the Trigger input, and for example sampling a noise source to generate a random melody.

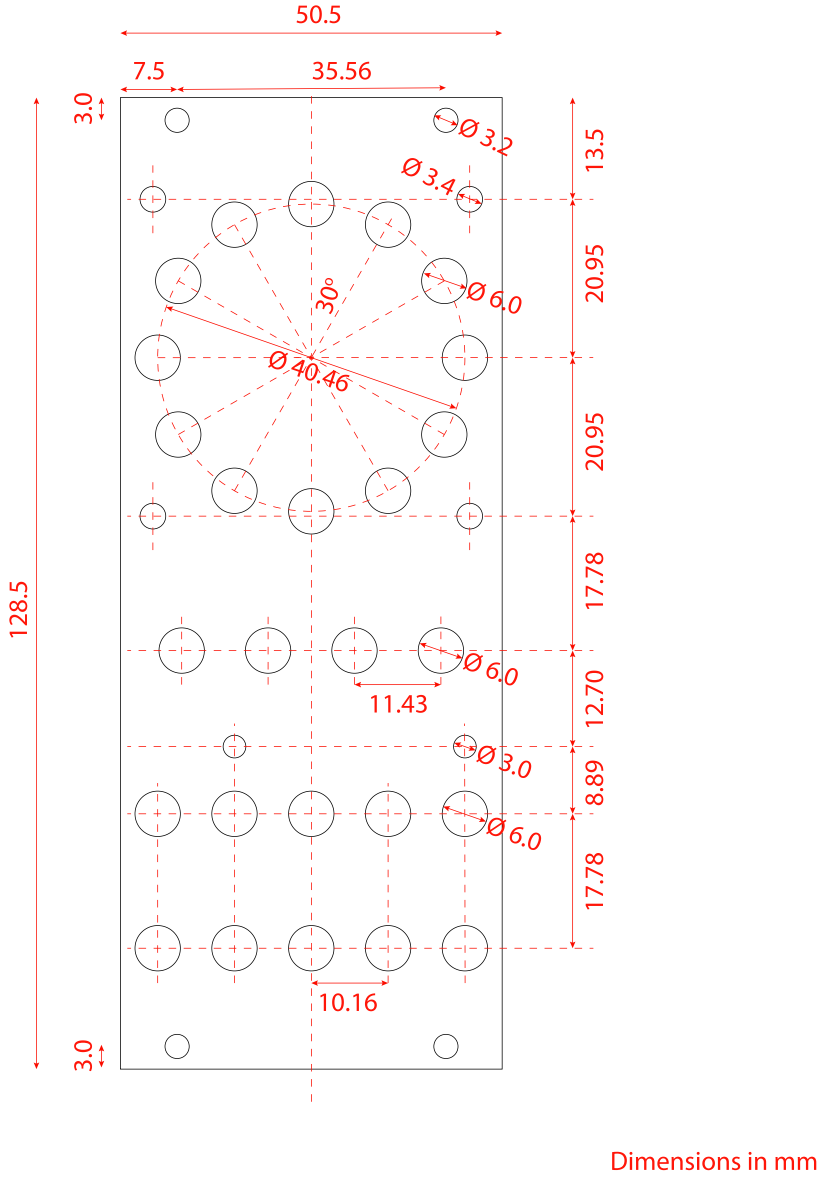
Trigger inputs and gate outputs

The trigger inputs are handled by discrete transistor inverters Q1 and Q2. These make for threshold voltages below 1V, and tolerance to external voltages well above and below the eurorack supply voltage range. The gate outputs are scaled by opamp U5 to around 9V, which should be enough to drive almost any other module or synth.

Front panel controls

The front panel buttons and LEDs require more I/O lines than available on the MCU. To drive the LEDs and read the buttons, standard shift register ICs U201 – U204 are used. By mounting these on the front panel PCB also the number of PCB interconnects is reduced. The buttons and LEDs are not multiplexed, meaning they do not flicker and also not induce ripple into the power lines. The LEDs, buttons and related circuitry are powered from a separate regulator U4, to minimize cross-talk between the LED current draw and the DAC output voltage.

Front panel dimensions



Possible modifications

Alternate front panels or formats

While the quantizer is specifically designed for Eurorack, the PCB can be adapted for other formats if desired. Especially bigger formats like 5U are suitable, since this adaptation requires panel wiring which will take up some extra space.

The the jacks section of the front panel board can be separated from the buttons section by cutting along the white line printed on the PCB. No traces run through this line. The buttons section should be used as is, and mounted to the front panel with 10.5mm (or longer up to 14mm) standoffs and M3 screws as described in the Mechanical assembly section. The main PCB can be connected to the buttons PCB either as described there, or by replacing J3/J201 and J5/J212 with with longer pin headers or wire links if more space is needed.

The front panel jacks and LEDs can be simply panel mounted and wired to J8 through J10. The following table lists the connections of these headers. Pin 1 is the square pad.

	J8	J9	J10	J11	J12
Pin 1:	Gate A	Gate B	Out A	CV A	Out B
Pin 2:	GND	+12V	GND	GND	GND
Pin 3:	Trig A	Trig B	In A	CV B	In B

The LEDs should be wired from the Gate A/B outputs through a 10k resistor and the other leg to GND. The +12V point should connect to the switch connection of the Trig A and Trig B jacks. The latter is needed for auto-detection of triggered mode vs free-running mode. Alternatively, a manual switch could be used for this function if desired. See the schematic diagram for more details on the wiring of this section.

Revision history

Board revisions

1.0 Initial prototype

1.1 Second prototype. Add CV inputs, add input filtering, improve output filtering.

Documentation revisions

A Initial documentation for board revision 1.1.

B Add recommended modification to prevent shorts of the +12V rail to ground.

Contact

Check for updated documentation and other information on my blog at kassu2000.blogspot.com. I am always happy to answer questions and receive feedback at kassutronics@gmail.com.