

**USER MANUAL**  
for  
**POT32 – MIDI controller**  
firmware version 3.x

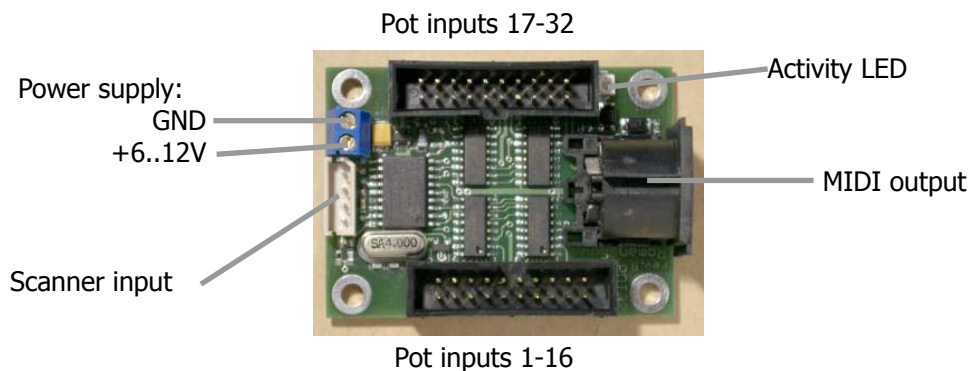
# 1 Overview

This little board is standalone controller for 32 potentiometers or analog inputs, that converts knob movement or voltage change into MIDI. In addition it has one expander input, this would be typically used to connect numeric keypad for setting up all the features, but you can connect also one of the keyboard scanners, and apart from knob assignment it can be used as normal keyboard. To have 64 or 96 potentiometers, add one or two POT32X in series.

# 2 Features

- 32 on-board inputs for potentiometers or switches or CV (control voltage)
- 1 scanner input for up to 128 contacts and 64 additional pots
- user defined MIDI channel and event, separately for each pot
- available MIDI events: Control Change, Pitch Bend, Channel After Touch, Program Change, notes
- user defined independent channel and transposition for each keyboard
- on-board DIN-5 MIDI-OUT socket
- requires 5-12V DC power supply, about 1-15mA current consumption depending on number and value of pots used.
- dimensions: 40x55mm (1.6"x2.2")
- all settings stored in non volatile memory, meaning they remain after disconnecting power

# 3 Layout

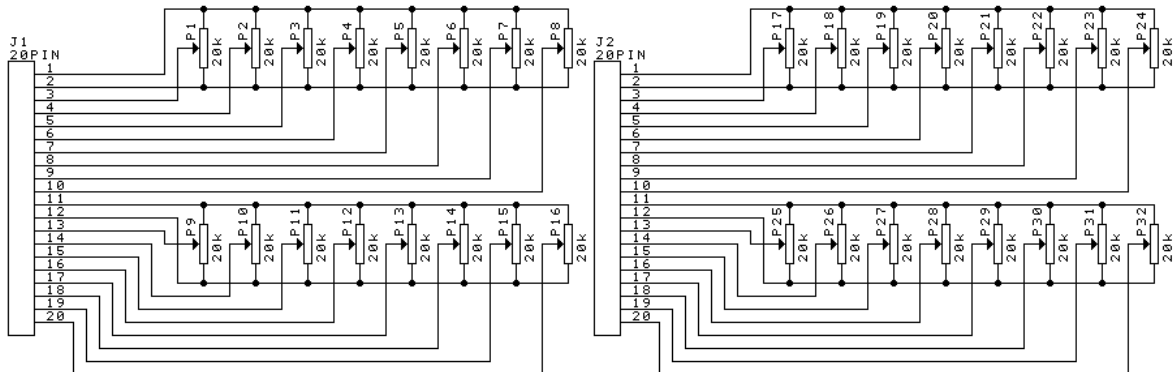


## 4 Power supply

Recommended power supply range is between 5.5 and 12V DC. It is possible to run this board even from as low voltage as 3V battery, but its operation is not guaranteed then. Current consumption is about 3mA with 2 scanners connected, and up to 50mA with all potentiometers installed. If e.g. 10k pots are used, current consumption will rise by 0.5mA per pot. The board comes with screw terminals for applying power. Positive terminal (+) is closer to 4-pin scanner input, while negative terminal (-) is closer to corner mounting hole. Connecting power in reverse makes no damage to the board, but it will work only when proper polarity is provided.

## 5 Connecting potentiometers

32 inputs can be used as continuous controllers for things like volume, modulation etc. Usually those inputs would be connected to potentiometers, but it's possible to use them as analog inputs with range of 0..+5V. Applying voltage of 0V causes POT32 to generate CC with lowest value, while +5V generates highest possible value of assigned MIDI parameter. There are 32 available inputs, all user configurable in terms of controller type and MIDI channel. Potentiometers are connected to 2 bigger black connectors (IDC20) in POT32 board. Following is schematic of potentiometers connections.



The cable from each IDC20 connector can be split in half and routed to a group of 8 potentiometers.

if you don't use all potentiometers you may want to connect unused inputs to VCC (pins 1 and 11). Keeping any, or all inputs open is quite safe, but in some cases may result in unexpected MIDI messages. Potentiometers must be linear taper (not audio) in range 10-50k, preferably 20k.

## 6 Connecting keyboards

Keyboard switches are connected with keyboard scanner, or in another words - expander board. There are several scanners available: for 16, 32, 48, 64, 128 keys, with switches organized in 8x8 matrix and single-rod bus-bar. All are described in chapter 10. Type of the scanner needed is determined by keyboard size and the way how switches are organized. Connection between keyboard scanner and POT32 main board is always the same, regardless of the type of scanner, whether it is 32 or 128 keys, or potentiometers. MIDI settings of those keyboards can be changed by the user after all connections are in place. All available scanners are described at the end of this manual.

8x8 scanner driver can be used if the keyboard has "scanning diode matrix", that's special kind of very simple circuit, made of diodes forming electric XY array of 8 rows and 8 columns. Usually all modern keyboards are equipped with it. In fact it is integral part of the contacts board found beneath the keys. Those kind of keyboards can work directly with MIDI128X scanner. Its advantage is that it can cover 2 such keyboards. Older keyboards, and especially those used in old analog organs, usually don't have such a thing, so in order to use 8x8 scanner, the DMTX64 board is needed in between the keyboard and MIDI128X. This is actually a diode matrix itself and it can be treated as a split board bridging MIDI128X and individual keys of the keyboard. There are also scanners especially designed for keyboards without diodes, where all keys share only one common bus, this is typical keyboard arrangement in all old organ consoles.

Below is a short table showing which scanner can be used with different keyboards:

Keyboard type	MIDI128X	BBS	PEDSCAN
Independent switches, no connections	◇	●	◇
Switches organized in 8x8 matrix, with diodes	●	‡	●
Switches organized in 5x12 and other		‡	
One common rail for all switches		●	

● - can be used directly

◇ - with additional diodes

‡ - some rewiring of original contacts circuit may be required

## 7 MIDI settings and special functions

All settings are accessible from a keyboard connected to POT32 with any type of compatible scanner. To access any settings, you have to enter into EDIT mode by temporary pushing 64th key of any keyboard scanner. Detailed procedures for all settings are described later in this chapter. To make the settings more ergonomic and easier, there's optional numeric keypad available, similar to phone keypad. It is connected the same way as any keyboard scanner with 4-wire cable, and works like actual keyboard, meaning it is possible to play notes with it. But the advantage is the „#” key, which enters the EDIT mode - it works like 64th key of keyboard scanner.

To change any settings in EDIT mode, you have to enter new value of given parameter. To do so, use lowest 10 keys of the keyboard as numeric entry. Lowest key is digit „0“, while 10<sup>th</sup> key is digit „9“. This is obvious when using mentioned numeric keypad. As a general rule, any change on a controller (keyboard, or potentiometer) requires selecting this controller first before making change. For example, if you want to change MIDI channel of certain potentiometer, move it a bit, and go into MIDI channel settings mode. Or to change the split point – first play any note on the keyboard to be split, and enter split-point change mode. In this chapter, describing how to set all parameters, whenever „#“ sign is mentioned, it means either 64th input of any scanner, or the „#“ key on numeric keypad if one is present in the system. Numeric entries are provided with the assumption that numeric keypad is used, but the same can be achieved with lowest 10 keys of any keyboard connected to POT32. It helps to add a sticker over lowest 10 keys with numbers from 0 to 9 if only musical keyboard is used.

### ***7.1 Transposition of keyboards***

Transposition of keyboards connected to POT32 is unlimited, that means any key can generate any MIDI note from range of over 12 octaves. There are two ways of using it. Typical one is by selecting “new middle C” position. First you have to select the keyboard you want to edit by playing any note on it. Enter „#“ then „1“ on the keypad. Now, whatever key you press, it will be the new position of the middle C MIDI note afterwards. You can select new position of middle C note anywhere between 3rd and top key of the keyboard.

Another option is to use lowest 2 keys of the keyboard, or numbers "0" and "1" of the keypad. It doesn't matter if keyboard starts with key C or F or whatever, those are always two lowest keys. The 1st one shifts the keyboard one semitone down with each sequence (#10), the 2nd shifts the keyboard one semitone up (#11). This is useful when you want to shift the keyboard in range not available by the first method, for example very low bass, or high treble.

Both methods require first selecting the keyboard to be changed by playing a note, then entering "# 1" on the keypad, and selecting transposition.

### ***7.2 MIDI event assignment for keyboards and analog inputs***

MIDI event assigned to given potentiometer or keyboard split can be easily changed. To perform this, turn a bit the knob, or play a key on the split you want to assign, and then select the controller type by entering keys # then 2 and then appropriate number from table in Appendix A. You need to enter 2 or 3 digits for each input controller depending on entered number. To assign another one, again you must turn the pot it a bit, or play the key on another split, and then start from "#2" sequence followed by event type number.

Possible MIDI event codes are from number 000 to 149.

**Standard setting for a keyboard is "# 2 131" - single notes, and for analog input "# 2 007" - Channel Volume. This is factory default.**

Possible settings are:

### 7.2.1 Control Change - #2 CC

Any MIDI Control Change number in range from "000" up to "127". Numbers above 127 are used to generate MIDI events other than Control Change, or turn them into other functions, what is described next.

### 7.2.2 Pitch Bend - #2128

The pot will work then as pitch bender. If assigned to a keyboard, each key will set pitch bender in 1/128 steps across the keyboard. Range can be adjusted with transposition settings.

### 7.2.3 Program Change - #2129

Although this is rather unusual usage, this pot will then generate MIDI Program Change messages with its every move. Program Change can be also generated from the keyboard, by using sequence "# 4 <number>" – this is described later. If assigned to a keyboard, pressing each key will generate MIDI Program Change message with different patch number. Starting number can be adjusted with transposition setting. This is useful for organ emulators, where bank of Program Change buttons can be used to work as pistons (sets of registers)

### 7.2.4 Channel After Touch - #2130

Turning such pot will cause Channel After Touch messages to be sent out. If assigned to a keyboard, each key will set After Touch in 1/128 steps across the keyboard. Range can be adjusted with transposition settings.

### 7.2.5 Standard keyboard action – single notes - #2131

Whenever MIDI event 131 is assigned to a keyboard, it works as typical MIDI keyboard, playing MIDI notes. It is also possible to generate notes played in glissando, when this event is assigned to a pot. Select the pot to be edited, enter "# 2 131". This knob becomes then a note generator resembling quantized Theremin. Move the knob and a series of notes will be played. There's only one note played at a time (with velocity set like described later) and it is released just before new note is about to play. Whole knob slow rotation plays 128 notes from entire MIDI range.

### 7.2.6 Note on only - #2132

This mode is somehow similar to the action described above, but only "note-on" messages are generated, that means whenever you move this pot, new notes will be played, and they will stay on forever unless proper note-off message will be issued by another means. If assigned to a keyboard, only note-on messages will be sent. It will work like with constantly depressed sustain pedal.

### 7.2.7 Note off only - #2133

This is like "note-on" mode described before, but instead it sends out only note-off messages. It can be used to mute part of notes already played, or as some kind of panic button – slow full rotation mutes all notes in assigned channel. If assigned to a keyboard, it will send only note-offs, so it may be used to quiet some notes played earlier.

### 7.2.8 One-touch Patch Recall - #2134

Keyboard in this mode serves as an array of favorite patches buttons. Each key recalls Program Change (or in another words - selects a patch/preset) that was earlier programmed. There's 64 memory locations, so you can use one full 64 key scanner. For example you can program key 1 to send Program Change 37, key 2 as PC#76, key 3 as PC#20 etc. Assigning Program Change numbers to a specific key is described later, in paragraph "Favorite Patches".

### 7.2.9 CC keyboard - #2135

This feature has no effect on a pot, i.e. the pot will generate no MIDI event if it has this feature assigned. In this mode you can use keyboard as toggle switches selecting min/max values of range of CCs. All keys have increasing MIDI Continuous Controller assigned. Pressed key sends CC with max value (127), while key release generates the same CC but with minimum value (0). CCs are ordered just like there would be MIDI notes, i.e. typically they start from CC#36 at the lowest key, next key is CC#37 etc. Use transposition settings to set different starting CC.

### 7.2.10 MIDI channel shift for all controls - #2136

This is usable only with keyboard scanners, because using it with pot needs high precision, as full change takes about 1/8 of a turn. When assigned to a keyboard, first 16 keys work like MIDI channel selector for all controllers. After one of the keys is hit, notes played on another keyboard are played on changed channel. Individual channel settings for every keyboard and potentiometer described in chapter 7.3 work together with this setting. For example if one keyboard was set to channel 3, and you change the channel using this feature to +4 (by hitting 5th key), resulting channel is 7 (3+4). If all controllers are set to channel 1, then all 16 keys assigned to this feature are direct channel selectors from 1 to 16.

### 7.2.11 Small Transposer - #2137

Select the pot to be edited, enter "# 2 137". Turning such pot will shift all notes played on all connected keyboard scanners by number of semitones determined by pot position. In the middle it gives no shift, and full rotation has range from -4 to +4 semitones. It's most useful when pot is replaced by 9-position switch with 8 resistors of equal value connected between switch leads. Assigning this to a keyboard is also possible, but you cannot reach full range of transposition with 5-octave keyboard.

### 7.2.12 Big Transposer - #2138

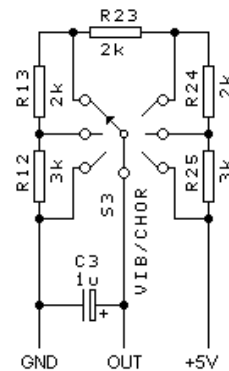
Turning such pot will shift all notes played on all connected keyboard scanners by number of semitones determined by pot position. In the middle it gives no shift, and full rotation has range from -8 to +8 semitones. Assigning this to a keyboard is also possible, but you cannot reach full range of transposition with 5-octave keyboard.

### 7.2.13 Velocity - #2139 and #2149

Position of this pot will then determine velocity parameter of all MIDI notes generated by this board, and all connected expanders/scanners. If assigned to a keyboard, each key will set velocity of all notes in 1/128 steps across the keyboard. Range can be adjusted with transposition settings. There can be only one such potentiometer for the whole system, and its settings affect all notes played on all keyboards in all channels. It's not intended for initialisation setup during installation, but rather as a way of performance expression during play. To set the velocity once, and always use chosen value, even after power cycle, use parameter #2149. It works exactly like the one described here, but additionally it remembers last position in nonvolatile memory, so it will use the same value after next power-up.

### 7.2.14 Native Instruments B4 chorus/vibrato - #2140

Turning this pot will be then reflected in B4 as "chorus/vibrato" switch position. It has only 6 positions, and appropriate command will be sent from POT32 to B4 every time the potentiometer crosses each threshold representing another vibrato/chorus mode. That's why it is recommended to use small circuit with rotary switch on the right for easier B4 usage.



### 7.2.15 MidiTzer stops control - #2141

When assigned to a keyboard, each key becomes specific stop controller. When a key is pressed, MIDI controller 81 (51 hex) is sent, and when it is released, MIDI controller 80 (50 hex). Value of the controller is determined by the button pressed. This is default way of controlling stops in MidiTzer organ software.

### 7.2.16 Ahlborn Archive module stops control - #2142

When assigned to a keyboard, each key becomes specific stop controller in Ahlborn Archive organ sound-module. When a key is pressed, MIDI controller 73 (49 hex) is sent, and when it is released, MIDI controller 74 (4A hex). Value of the controller is determined by the button pressed. This is default way of controlling stops in Ahlborn Archive module.

### 7.2.17 Ahlborn Organs stops control - #2143

When assigned to a keyboard, each key becomes specific stop controller in Ahlborn Organs. When a key is pressed, MIDI controller 70 (46 hex) is sent with bit 6 of the value set, and when it is released, the same MIDI controller but with bit 6 of the value cleared. Other bits of the value are determined by the button pressed. In another words, pressing the button sends CC 70 with value range 0-63, and releasing a button - CC 70 with value range 64-127. This is default way of controlling stops in Ahlborn Organs.



## 7.4 **Program Change**

POT32 allows you to send Program Change MIDI messages, or in another words – change patches. Three ways are available. Two were mentioned in Knob Assignment chapter, where you could program the potentiometer to act like 128-position patch rotary switch, or use keyboard assigned to Program Change to act like array of single touch patch select buttons. To change the patch on selected keyboard directly to specific number, play a note on this keyboard and enter the sequence: "# 4 <program number>". The Program Change MIDI message is sent directly after last digit of entered patch number. This may happen after 2nd or 3rd digit. You only need to enter 2 digit, when the patch number is in range 13-99. Programs lower than 13 require 3 digits, with 0s in front, for example 012, or 003. Obviously, programs with numbers higher than 99 also need 3 digits. The range of Program Change is from 000 to 127.

## 7.5 **Keyboard split**

It is possible to split each keyboard into 2 independent parts. The split point can be anywhere on the keyboard, and both parts can work with independently adjusted MIDI channel, type of event and starting note (transposition) or range of other controllers if something else than notes is assigned to a keyboard. Assuming that all 4 inputs of POT32 are equipped with MIDI128X dual keyboard scanner, it is possible to make a system with 8 individual splits making of 16 independent parts. To set up the split point, you have to select the keyboard to be split by playing a note in it, and then enter sequence „# 5” followed by stroke of the key that you want to be the last one of the lower part. Since then lower part remains on the same channel that was used for whole keyboard, while upper part takes settings of upper part, which by default is 8 MIDI channels higher. To change MIDI channel, type of event, transposition, or send a Program Change for split part, follow directions described above, regarding non-split keyboard, but now changes are made to this split part, which was selected by playing a note prior entering the edit mode (pressing „#”).

## 7.6 **Favorite patches**

Whenever a program/patch/instrument selected from POT32 is often used, or interesting, it is worth to memorize for fast recall in the future. There can be 64 such favorite patches, selected for last played keyboard or split by only 1 keystroke. To change a patch for last played keyboard, simply select one of the buttons assigned as one-touch program recall described in chapter 7.2.8. POT32 will send the patch number in channel of last played keyboard, or last turned pot. To memorize any patch for use in this way, first select this patch, either by entering „# 4 <patch number>”, or by turning „Program Change” knob if one is assigned, or by using one of the keys in keyboard assigned to Program Change event. Then press "# 6", and then the key where the patch should be stored. Next time whenever you this key/button, the MIDI Program Change message will be issued, setting the patch that was programmed into that button.

## 8 Advanced settings

Apart from MIDI settings described before, there are few settings associated with boards themselves, and not related to MIDI data. As always, settings can be changed in EDIT mode invoked by the las input of any scanner, or "#" key in numeric keypad. All such parameters start with "#9" (digit 9 on the keypad, or 10th key on any keyboard scanner).

### 8.1 *Analog inputs update rate*

All analog inputs, on-board and in optional potentiometer scanners (POT32X, POT12, POT3, or PEDSCAN-X) translate input voltages, or potentiometer position into MIDI. Actual pot position is updated via MIDI every time it changes. This update is however not immediate - this is common to any MIDI knob box. The fastest response for potentiometer movement on a single input is about 5ms. It means that when you constantly move the pot, POT32 will issue a MIDI update every 5ms. This is more than enough for most of uses. In some instruments, either hardware, or virtual, some problem may occur when there is a lot of MIDI traffic. It is also sometimes desirable to limit MIDI traffic e.g. to minimize the size of MIDI file recorded on a sequencer. It is possible to change this setting using command "#98n" from the keypad, where "n" determines update rate according to the table below. Default factory setting is 18ms.

keypad sequence	#980	#981	#982	#983	<b>#984</b>	#985	#986	#987	#988	#989
pot latency	5ms	7ms	9ms	13ms	<b>18ms</b>	25ms	35ms	50ms	70ms	0.1s
update rate	200Hz	145Hz	115Hz	80Hz	<b>55Hz</b>	40Hz	30Hz	20Hz	15Hz	10Hz

This setting is available for POT32 and all POT32X, POT12, POT3 and PEDSCAN-X scanners connected to POT32 board. Each board can have different update rates. For example you can set it to 50ms on POT32 board, and 13ms on additional scanner. As usual, to change any settings for particular board, select it first by moving a bit any potentiometer connected to board in question. Then using your keypad enter the code from above table. New settings will be activated and remembered. Unlike MIDI settings from chapter 7, all advanced settings described here are associated with actual scanner board and memorized there. If you replace the board, or move it to another system, it will still remember last settings entered.

### 8.2 *Scanner behavior - LITSW only*

At this moment only one scanner - LITSW can be programmed this way, others to follow. It is possible to change the way how LITSW operates. Normally it is used to generate note-on and note-off messages with momentary buttons and LEDs, most useful as register control in organ emulator. But it can be converted into a few variations, finding its way toward other non-typical uses. Possible controls are:

### 8.2.1 Split point - #905

This is different kind of split than the one described in paragraph 7.5 and is independent of that one. So you can use both kinds of split at one time, and they can be at different points. The kind described here controls only button's behavior, and not actual MIDI data transmitted by POT32 to MIDI OUT socket. Each split can work in different mode, with independent or dependent buttons, generating CCs or notes. To have different MIDI events or channels in the splits you have to also split it logically, using procedure described in 7.5.

### 8.2.2 Independent mode - #908

In independent mode, all buttons work without interactions to each other. If a button is pressed, associated LED lights, and note-on is sent by POT32 (only if it is configured to send notes on this input - sequence #2131). Another touch of this button and LED turns off and note-off is sent. To set this mode on a split, you have to select it first by pressing one of the buttons in that split.

### 8.2.3 Dependent mode - #909

In dependent mode, there can be only one LED active at a time. If you press another button, it will light up and any one that was previously lit, will now turn off. In this mode only note-on messages are generated, there is no note-off. This is most useful for selecting presets on MIDI instrument. To set this mode on a split, you have to select it first by pressing one of the buttons in that split.

### 8.2.4 Bank/preset select - #910

It is possible to split the LITSW board into 2 parts by keypad sequence #5 described in MIDI settings paragraph. If you type #910 on the keypad, LITSW board will then work as patch selector with separate row of "banks" buttons and "presets" buttons. If the split is made after 10 keys, lowest 10 buttons will act like preset selector within a bank, and all buttons above 10 - as bank selectors. Although it is possible to use this mode with independent buttons, described in 8.2.1, most obvious use is together with dependent buttons mode described in 8.2.2. To use it as Program Change selector, you should program appropriate MIDI event on this input, which for Program Change is - #2129. The split point doesn't have to be after 10 buttons. It can be e.g. after 6 buttons. Then first bank will select patches from 1 to 6, second bank from 7 to 12 etc.

### 8.2.5 Disable bank/select mode - #911

To disable bank/select mode described above, you must type #911 sequence on the keypad. It then returns to normal mode, where each button has the same weight, there's no split for banks or presets.

### 8.2.6 Note keyboard scanner mode - #912

This is normal way of operation for LITSW. When a button is pressed, POT32 can then generate MIDI notes, or any other MIDI event that can be programmed to a keyboard scanner. This is the opposite to CC mode described next. To set this mode on a split, you have to select it first by pressing one of the buttons in that split.

### 8.2.7 CC scanner mode - #913

In this mode LITSW behaves like potentiometer scanner. When button is pressed, it simulates turning a potentiometer to a maximum, and when the button is deactivated (LED is turned off) it simulates a potentiometer turned into minimum. So it works like 24 potentiometers with only 2 valid positions: min and max. This is useful for switching parameters in virtual instruments, e.g. Hammond emulators. For each button you can then setup individual MIDI event (like CC or anything else) and MIDI channel. To set this mode on a split, you have to select it first by pressing one of the buttons in that split.

### 8.2.8 Left split blink - #916

When you type this sequence on a keypad, LITSW will blink all LEDs in lower split of LITSW. This can be used to determine where the split point is, and where are the LEDs connected.

### 8.2.9 Right split blink - #917

When you type this sequence on a keypad, LITSW will blink all LEDs in upper split of LITSW. This can be used to determine where the split point is, and where are the LEDs connected.

### 8.2.10 All LEDs blink - #918

When you type this sequence on a keypad, whole LITSW will blink all LEDs. This can be used to see if every LED is connected properly and where they are located.

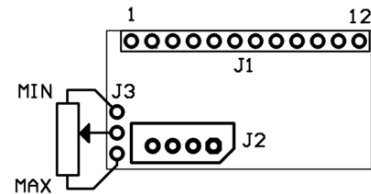
## 9 Scanners

POT32 board is the merging point for all keyboard and potentiometer scanners. Depending on their sizes, the whole system can cover up to 10 keyboards or 640 keys over single MIDI socket together with 64 potentiometers at the same time. When MIDI merging input is used to chain multiple POT32 systems, total size is only limited by MIDI standard itself.

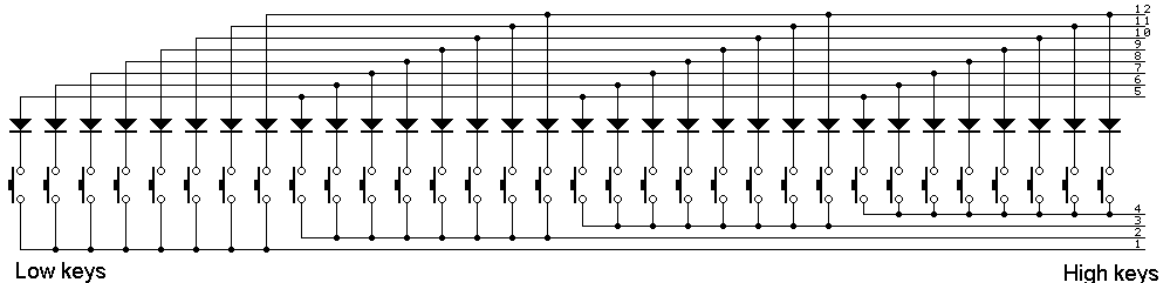
Currently available sizes are: 32, 64 and 128 keys, the last one has the 128 keys split into 2 keyboards.

### 9.1 *pedal scanner - PEDSCAN-X*

Pedal board controller takes care of 32 keys and one analog input, usually all what's needed for pedals with a swell shoe. It's a small board that can be fitted inside pedal board, and it connects to the main board via supplied 4-wire cable from J2 connector. 3 pads labeled in this picture as J3, are for potentiometer. The picture shows how to connect the pot, and also in which pot position you get the maximum, or minimum value of given MIDI parameter controlled by the pot.



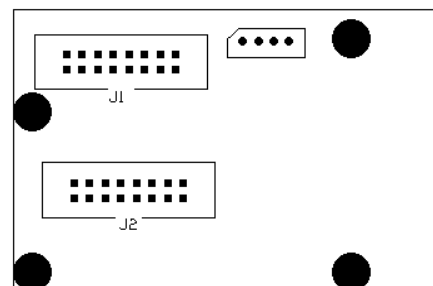
Keys must be connected in diode matrix exactly like in the schematics below:



**The groups of 8 switches with 1 common lead must be separate. In case of keyboard with 1 common bus bar going through entire keyboard, you have to cut the bar every 8th key.**

### 9.2 *dual keyboard scanner - MIDI128X*

There can be another optional board connected, adding 128 inputs. This can be described as "diode matrix driver" for 2 keyboards. It is useful when you want to connect 2 modern keyboards, that usually have 8x8 diode matrix built in together with switches. The layout is shown below. There are two 16-way connectors, that usually are used to connect diode-matrix keyboards. Smaller connector marked as J5 is



used to connect it with main controller board - POT32.  
 J2 – connector of 1st group of keys (1 to 64)  
 J1 – connector of 2nd group of keys (65 to 128)  
 J3 – connector linking this board with the main board

Each 16-way connector covers one keyboard. The keyboards must have "8x8 scanning diode matrix", that's special kind of very simple circuit, made of diodes forming electric XY matrix. Usually all modern keyboards are equipped with it although sometimes the matrix is organized differently, say 5x12 or 6x11. In fact it is integral part of the contacts board usually found beneath the keys.

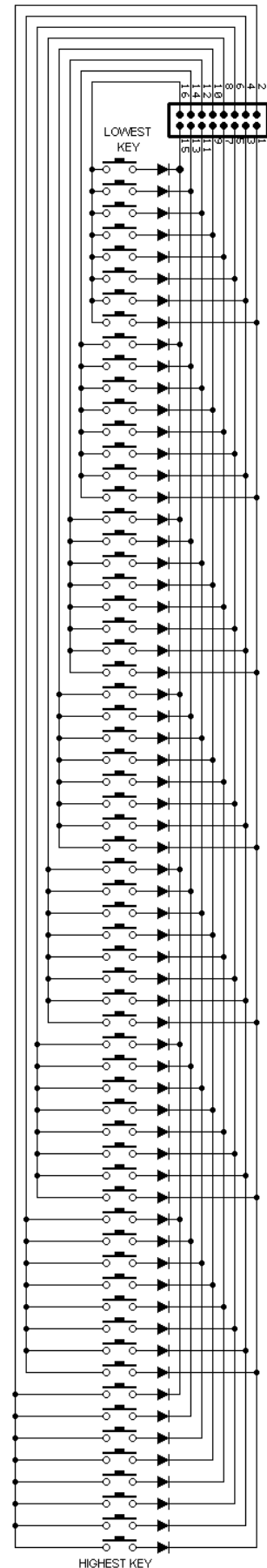
Schematic on the right shows example of diode matrix compatible with MIDI128X board.

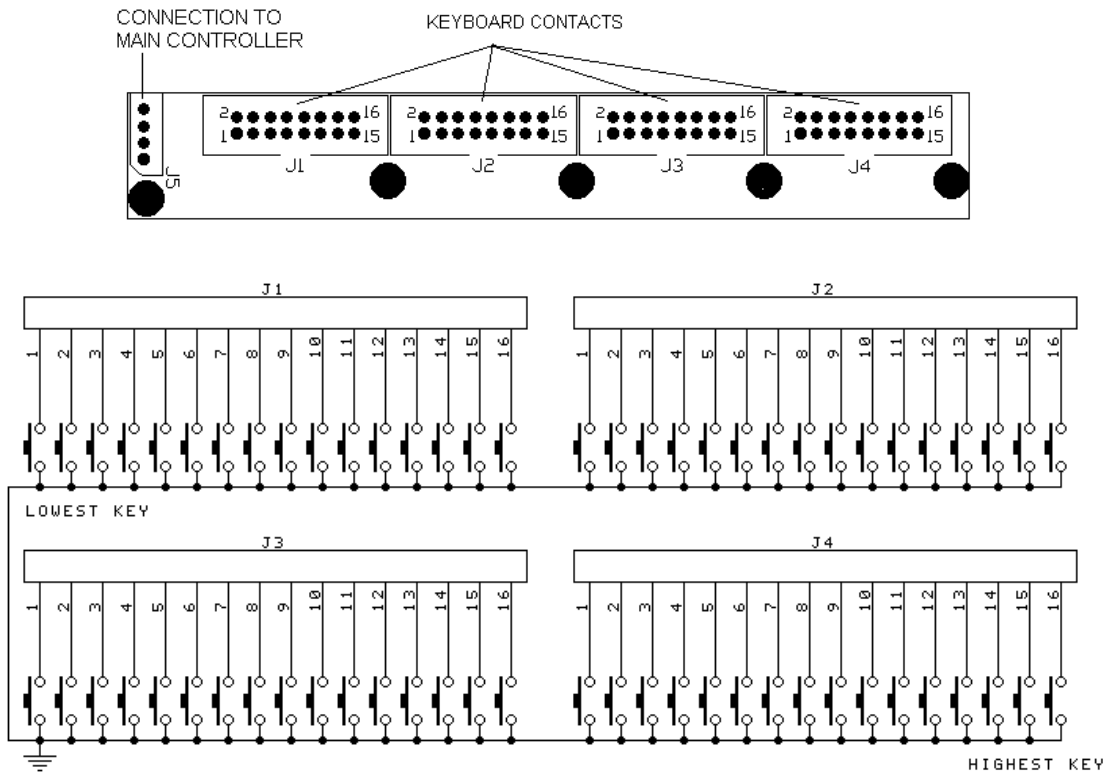
### 9.3 *common contact scanner - BBS64X, BBS32X*

There's another family of keyboard scanners, especially suitable for keyboards with single rod used as common bus for all switches in entire keyboard. This one does not use diode matrix, and can be used with almost any type of switch arrangement, it can also be TTL controlled. The BBSX boards come in 2 variants, or lengths. They can work with either positive or negative keying, which means that key pressed is represented by 0V, or +5V. The board can have less 16-pin connectors available, covering only 32 keys. It's then called BBS32X.

In case of the big, 64-keys version, keyboard is connected to four 16-pin connectors - J1, J2, J3 and J4 shown at right. Each of them covers 16 keys. The key contacts can have one common buss bar (with GND), or it can be driven from logic IC outputs. 0V at an input means "key pressed", +5V at input or left open means "key released". The BBS board can be also ordered with reversed logic, i.e. positive voltage at input means "key pressed", 0V - "key released".

Keyboard should be connected to four 16-pin headers according to the diagram on the next page. It's best to use 4 IDC connectors and 16-wire flat computer grade cable. Each cable connects to 16 consecutive keys. Connection of all keys to pins in BBS64X is also shown in diagram on the next page.

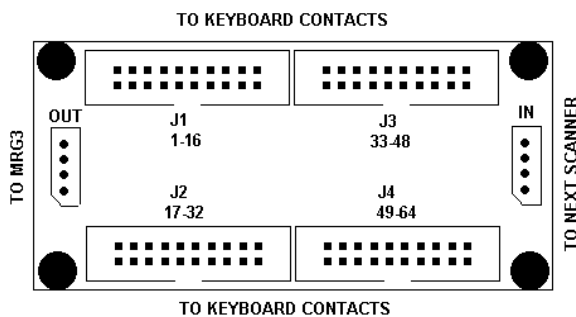




## 9.4 *common contact chained scanner - BBS-1K*

This is new member of midi-hardware.com keyboard scanners. It has the same functionality as BBSX boards described in chapter 10.3, but has some interesting features. For basic functionality and connections description please refer to chapter 10.3, here are only outlined the differences between those 2 boards.

BBS-1K has daisy-chain input, which allows to pack more scanners into the system. For example with POT32 you can use 10 BBS-1K boards, but only 5 BBSX board. It also has a bit different connector layout, but all 4 connectors for ribbon cable are the same. In contrary to BBSX, this one comes always in 64-inputs version, there's no smaller one. Each black connector covers 16 keys, and little sticker on the board shows what range of inputs are assigned to each connector. This is also explained in the drawing below.



BBS-1K must be connected towards the POT32 with 4-way connector indicated above as "OUT". The connector marked as "IN" is for the purpose of adding next optional scanner. There can be only 2 keyboard scanners (BBS-1K or other) connected in one chain. If third keyboard scanner is used in one chain, it will behave like paralleled inputs of 2nd BBS-1K in the system. Hence the limit of 10 keyboards for POT32 (2 x 5 inputs).

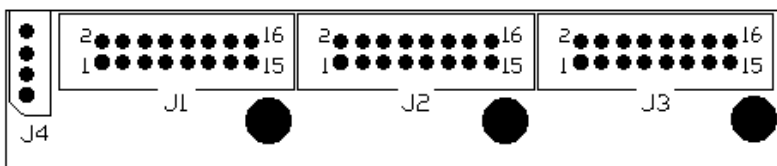
BBS-1K must be connected towards the POT32 with 4-way connector indicated above as "OUT". The connector marked as "IN" is for the purpose of adding next optional scanner. There can be only 2 keyboard scanners (BBS-1K or other) connected in one chain. If third keyboard scanner is used in one chain, it will behave like

## 9.5 LITSW - button scanner with LED drivers

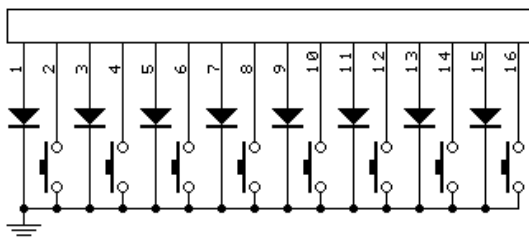
LITSW is the controller for lighted switches. It has 24 inputs for contacts, and 24 outputs for LEDs associated with them. Currently available modes of operation for this board:

1. independent registers, where each button pressing lights up or darkens the LED, and proper MIDI message is generated according to POT32 setting for that scanner. If this is MIDI note, a note-on is generated at LED turn-on and note-off when LED turns off. If it is Program Change, only one event is generated, when the LED turns on
2. dependent choice, where only one LED-button can be active (lights up). Pressing any other button causes previously lighted to turn off, and the one pressed lights up now. If assigned to notes, only MIDI note-on is generated, this is more suitable for use with Program Change
3. keyboard mode, where button action can be treated as typical keyboard scanner. So all kinds of MIDI events applicable to a keyboard are also possible here
4. potentiometer mode, where LITSW simulates potentiometer scanner. Buttons don't act like in keyboard scanner, but simulate min/max position of a potentiometer. This allows to assign MIDI events and channels separately to each of the buttons.
5. bank/select, useful for preset selectors. In this mode LITSW is split in 2 parts. One is serving as bank select, the other one - as program select. If the split is made on 10th key, lower split is representing units of given Program number, and the upper one - tens of this number. Of course it makes more sense when MIDI event 129 is programmed in POT32 (that's Program Change)

It is possible to use more than one at once, although not always it makes sense. For example modes 1-2, as well as 3-4 are mutually exclusive, but you can set for example modes 1, 3 and 5 together.



- J1 - LED-buttons 1-8
- J2 - LED buttons 9-16
- J3 - LED buttons 17-24
- J4 - connector to POT32



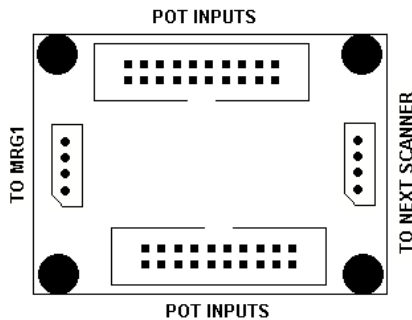
The pins of each LED-button connector are interlaced for easier installation. Odd pins are LED outputs, and even pins are button inputs in the following manner: pin 1 - LED 1, pin 2 - button 1, pin 3 - LED 2, pin 4 - button 2, etc. This is shown in the following on the left. Each 16-pin connector is the same.

## 9.6 POT32X - potentiometer scanner

POT32X adds 32 potentiometer to MIDI system built with POT32 as main board. It has two 20-pin black connectors for potentiometers, and 2 small 4-pin sockets, typical for all midi-hardware.com MIDI board. One of those is used to connect the potentiometer scanner in POT32. The other one can be used to connect another scanner, be it keyboard or another potentiometer. You can chain only up to 2 POT32X scanners, giving you total of 64 potentiometer inputs for the whole system. If there are 2 POT32X in the system, they must be chained, i.e. first POT32X is connected to POT32, and second POT32X is connected to first POT32X. If you use 2 inputs of POT32 to connect 2 POT32X boards, they both will use the same settings. That means it would be like having 2 potentiometers for the same MIDI parameter on the same channel.

The chain of potentiometer scanners can be connected to any one of the 5 POT32 inputs, or any daisy-chain input of other connected scanners, but only one. Only keyboard scanners can be connected to all 5 inputs.

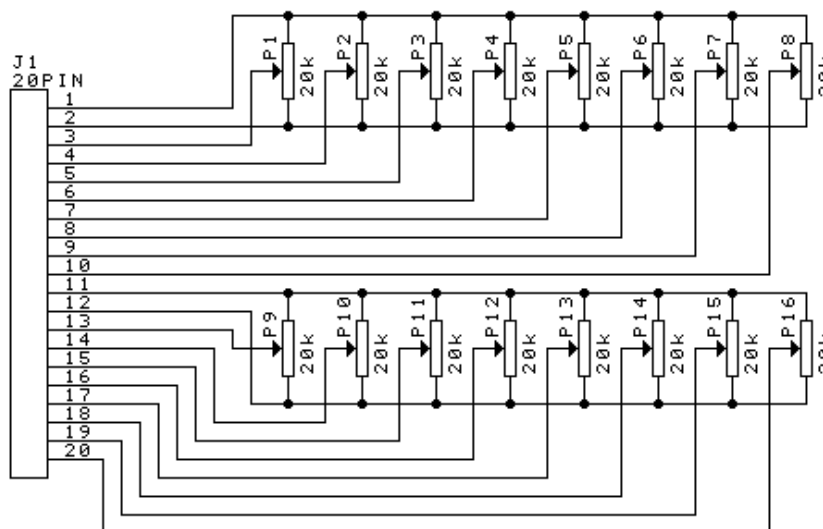
Here's layout of the POT32X potentiometer scanner:



Notice the orientation of angled corners of the small connectors in the drawing on the left to properly connect it. 4-pin connector on the left side must be connected to POT32, either directly, or via another POT32X. Connector on the right side can be used to add another scanner, e.g. next POT32X, or keypad. If you reverse connections between those 2 sockets, the board

will not work, and potentiometer movement will not result in any MIDI activity. However this does not cause any damage to the POT32X board, or POT32.

Two bigger black connectors are the potentiometer inputs. Their connection is shown below (only one connector shown, 2nd one is identical):

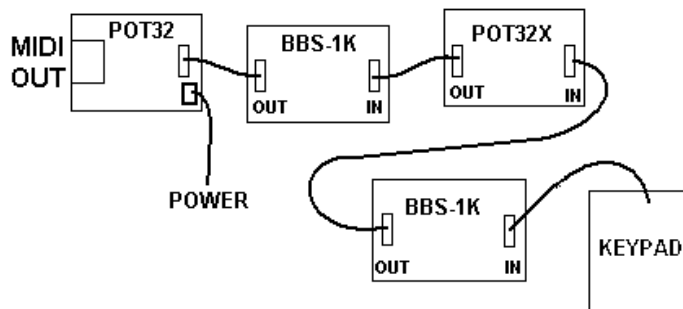


Each sockets contains 2 pairs of contacts for potentiometer common rails, namely GND and +5V. This is only for convenience of installation, and one pair can be omitted if needed. The cable from each IDC20 connector can be split in half and routed to a group of 8 potentiometers.

If you don't use all potentiometers you may want to connect unused inputs to VCC (pins 1 and 11). Keeping any, or all inputs open is quite safe, but in some cases may result in unexpected MIDI messages. Potentiometers must be linear taper (not audio) in range 10-50k, preferably 20k.

## 10 Example of MIDI set with POT32 & scanners

Below is a simple diagram showing one of many possible sets of MIDI boards. The system below covers 2 keyboards with single common bus (BBS-1K), 64 potentiometer inputs and numeric keypad for easier setting parameters and entering Program Change numbers.



Position of each board determines its settings. If for example you set first keyboard scanner to channel 1 and 2nd to channel 2, then reconnect them so their position in the chain is changed, MIDI notes played on those scanners will play on swapped MIDI channels. Likewise, if there are 2 POT32X in the system, changing their relative position will swap all MIDI parameters associated with them.

On the contrary, relative position in the chain of different scanner types, like keyboard scanners and potentiometer scanners doesn't matter. POT32X can be before BBS-1K or the other way around, and all settings will stay unchanged. It is recommended however, to place keyboard scanners as the first one in the chain. This may slightly reduce keyboard latency.

**Appendix A** – numbers for assigning MIDI event type to potentiometers. For up to date assignment of CC numbers to controllers go to [www.midi.org](http://www.midi.org). Note that features described in *italic* with numbers above 127 don't represent a CC, but other MIDI events or parameters.

digits	Controller name	digits	Controller name
000	Bank Select	68	Legato Footswitch
001	Modulation Wheel or Lever	69	Hold 2
002	Breath Controller	70	Sound Controller 1 (Sound Variation)
004	Foot Controller	71	Sound Controller 2 (Timbre/Harmonic Intens.)
005	Portamento Time	72	Sound Controller 3 (Release Time)
006	Data Entry MSB	73	Sound Controller 4 (Attack Time)
007	Channel Volume	74	Sound Controller 5 (Brightness)
008	Balance	75	Sound Controller 6 (Decay Time)
010	Pan	76	Sound Controller 7 (Vibrato Rate)
011	Expression Controller	77	Sound Controller 8 (Vibrato Depth)
012	Effect Control 1	78	Sound Controller 9 (Vibrato Delay)
013	Effect Control 2	79	Sound Controller 10 (default undefined)
16	General Purpose Controller 1	80	General Purpose Controller 5
17	General Purpose Controller 2	81	General Purpose Controller 6
18	General Purpose Controller 3	82	General Purpose Controller 7
19	General Purpose Controller 4	83	General Purpose Controller 8
32	LSB for Control 0 (Bank Select)	84	Portamento Control
33	LSB for Control 1 (Modulation Wheel or Lever)	91	Effects 1 Depth (Reverb Send Level)
34	LSB for Control 2 (Breath Controller)	92	Effects 2 Depth
36	LSB for Control 4 (Foot Controller)	93	Effects 3 Depth (Chorus Send Level)
37	LSB for Control 5 (Portamento Time)	94	Effects 4 Depth
38	LSB for Control 6 (Data Entry)	95	Effects 5 Depth
39	LSB for Control 7 (Channel Volume)	96	Data Increment (Data Entry +1)
40	LSB for Control 8 (Balance)	97	Data Decrement (Data Entry -1)
42	LSB for Control 10 (Pan)	98	Non-Registered Parameter Number (NRPN) - LSB
43	LSB for Control 11 (Expression Controller)	99	Non-Registered Parameter Number (NRPN) - MSB
44	LSB for Control 12 (Effect control 1)	100	Registered Parameter Number (RPN) - LSB
45	LSB for Control 13 (Effect control 2)	101	Registered Parameter Number (RPN) - MSB
48	LSB for Control 16 (General Purpose Controller 1)	120	[Channel Mode Message] All Sound Off
49	LSB for Control 17 (General Purpose Controller 2)	121	[Channel Mode Message] Reset All Controllers
50	LSB for Control 18 (General Purpose Controller 3)	122	[Channel Mode Message] Local Control On/Off
51	LSB for Control 19 (General Purpose Controller 4)	123	[Channel Mode Message] All Notes Off
64	Damper Pedal on/off (Sustain)	124	[Channel Mode Message] Omni Mode Off
65	Portamento On/Off	125	[Channel Mode Message] Omni Mode On
66	Sostenuto On/Off	126	[Channel Mode Message] Poly Mode On/Off
67	Soft Pedal On/Off	127	[Channel Mode Message] Poly Mode On
digits	feature name	digits	feature name
128	<i>Pitch Bend</i>	138	<i>Transpose by +/-8</i>
129	<i>Program Change</i>	139	<i>Velocity of all notes played</i>
130	<i>Channel After Touch</i>	140	<i>B4 chorus-vibrato</i>
131	<i>Standard keyboard action, glissando for pot</i>	141	<i>MidiTzer stops</i>
132	<i>Note-on</i>	142	<i>Ahlborn Archive stops</i>
133	<i>Note-off</i>	143	<i>Ahlborn Organs stops</i>
134	<i>Favorite Patches</i>	144	<i>Program Change select within a bank of 12</i>
135	<i>CC keyboard</i>	145	<i>Program Change Bank select with rotary switches</i>
136	<i>MIDI channel change for all</i>	149	<i>setup of default velocity for all notes</i>
137	<i>Transpose by +/-4</i>		