mkc64u
Universal USB-MIDI keyboard encoder
* user's guide *

1. What is this?

mkc64u is the USB-MIDI-successor of our best-selling unit **mkc64x** and covers its entire functionality, adding better performance and wider capabilities.

mkc64u has 8x8 scan matrix input, capable to scan and encode to MIDI up to 64 key contacts or general contacts. It has 5 additional switch/button inputs as well as 8 analog inputs where potentiometers can be wired directly for allowing Continuous control by faders, pedals etc.

Hence, **mkc64u** has the capability of encoding up to 64 + 5=690 contacts and 8 potentiometers. This is enough for most typical applications involving keyboard and expression controls (faders, pedals).

The most important capability of this unit though is its ***user-programmability***! Each of contacts can be programmed to send any combination of up to 32 MIDI bytes upon engaging and up to 32 MIDI bytes upon releasing. Each of potentiometers can be programmed to send up to 32 MIDI bytes upon position change where one or more of these bytes can carry the data about potentiometer's position (number between 0 and 127).

The support of user-programmability allows using **mkc64u** for **controlling virtually any MIDI-controllable software.**

Last but not least: **mkc64u** has USB-MIDI input capable of full merging, with 256-byte buffer, allowing other software applications to be cascaded to **mkc64u** without additional virtual merging software. The USB-MIDI input is also used for programming MIDI events in **mkc64u.**

2. How it works?

In **mkc64u** internal program memory there is a table of MIDI bytes. For each of key inputs there are two entries/strings containing 32 MIDI-bytes: one string for On event and the other string for Off event. For each of potentiometers there is an 32-byte entry/string containing 32 MIDI bytes that are sent upon each potentiometer position change.

Hence, there is a table of totally 64 + 5 + 8 = 77 MIDI strings, each 32 bytes long.

Each of these strings can be reprogrammed by user for any MIDI content. The string can contain MIDI status bytes, MIDI Data bytes, System and Realtime MIDI messages, etc. It is entirely up to the user what MIDI string will be sent by triggering/changing each of **mkc64u** inputs.

The programming is done by uploading the specific MIDI string to specific table entry via **mkc64u** USB-MIDI input. An special Windows-based application was designed for this purpose, called **mkc64uprg.exe**. It is available for free download on our site (follow links on **mkc64u** product page).

The **mkc64u** can be programmed using other Windows or non-Windows applications, providing that they can send user-defined System Exclusive messages and user-defined MIDI strings. The programming sequence and messages protocol and format are described in ***Appendix A*** of this document.

3. MIDI implementation

Appendix B shows the <u>*factory-programmed*</u> MIDI implementation of **mkc64u**.

This MIDI implementation can be used as it is and can be changed by user when/if necessary. In it each of scan points triggers NoteOn/NoteOff messages on separate channel per scan matrix. MIDI Channel 1 is covered. The note range is MIDI notes 36 – 99. The additional switch/button inputs trigger Control Change messages (CC80-CC85) on MIDI channel 1. The continuous/potentiometer inputs trigger continuous controller CC7 (Volume) on MIDI channels 1-8.

4. Wiring diagrams

There are various ways of wiring **mkc64u**, depending on contact system and wiring scheme chosen by user. The **mkc64u** can be wired directly to key contacts in a scheme called *scan matrix wiring*. No other devices are necessary in this scheme, except single serial diode added to each scan matrix contact being encoded. Usually diodes are soldered directly on one terminal of contact, so there is no need of additional PCB for them. The scan matrix/keyboard contacts should be organized in groups of 8 called Rows, 8 Rows that form scanmatrix. Unused scan points should be left unconnected. Such architecture requires 16 wire ribbon cable between the scanmatrix header of **mkc64u** and the scan matricx itself. The drawback of such wiring is that it requires a lot of wiring, good soldering and wiring skills and quite a patience.

The additional switch/button and potentiometer control inputs should be connected to switches/buttons and potentiometers as shown on schematic. Each non-wired button/switch input will be read as being open contact (break). Each potentiometer input left unconnected will be read as potentiometer left in Max position. A button input connected to GND will be read as being closed contact (make). In case potentiometer input is grounded it will be read as potentiometers left in Min position.

The proper Control Change messages for each button and potentiometer will be transmitted once upon on initializing (starting up).

The other possible wiring schemes (as per bundles offered) are shown in ***Appendices C,D,E.***

5. Technical specification.

Parameter	Value	Unit	Comment
Power supply voltage	5	V	*Powered from USB host!*
Power supply current	100	mA	
Number of contact inputs (scan points)	8x8 + 5 = 69	–	Normally open or normally closed type
Scan rate for contacts	300	S^{-1}	Each key contact is being scanned 300 times per second
Number of analog/potentiometer inputs	8		10 – 100 kOhm linear potentiometers (preferably 10kOhm)
Scan rate or potentiometers	50	S^{-1}	Each potentiometer is being scanned 50 times per second
MIDI messages	User-defined	–	Up to 32-byte MIDI user-defined string per event
MIDI channels	User-defined	–	Defined per MIDI event (single MIDI string can contain MIDI messages going on various MIDI channels
USB-MIDI Merge	Yes	–	256 byte buffer
Size	8x7x2.5	cm	Approx. 3.1"x2.8"x1"
Weight	32	g	Approx. 1.1 oz

Appendix A. mkc64u – Programming

Step-by-step programming sequence

1. Download the ***mkc64uprg.zip*** file from our site (www.midiboutique.com). The actual link can be found on **mkc64u** product page.
2. Copy and unzip the downloaded file to dedicated folder.
3. Connect **mkc64u** to one of computer's USB ports using standard USB cable. The **mkc64u** will be automatically powered from the USB port.
4. Run the unzipped ***mkc64uprg.exe*** utility.
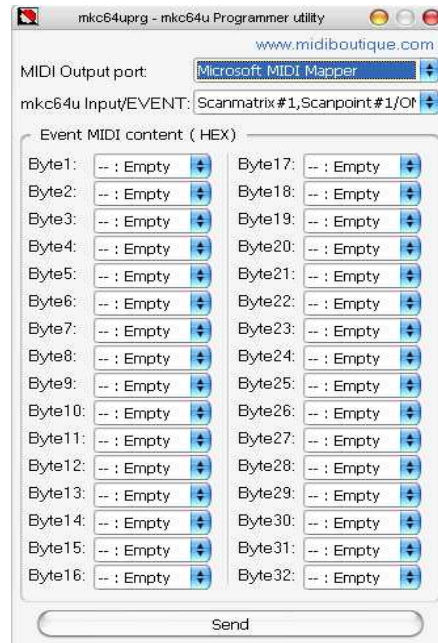

5. Select the proper MIDI output port from *'MIDI Output'* drop-down list.
   NOTES:
   Some systems may have more than one MIDI output, there could be hardware and virtual ports as well. Make sure you have selected the virtual MIDI output that is assigned to **mkc64u**. Normally in Windows it appears as *USB Audio device*.
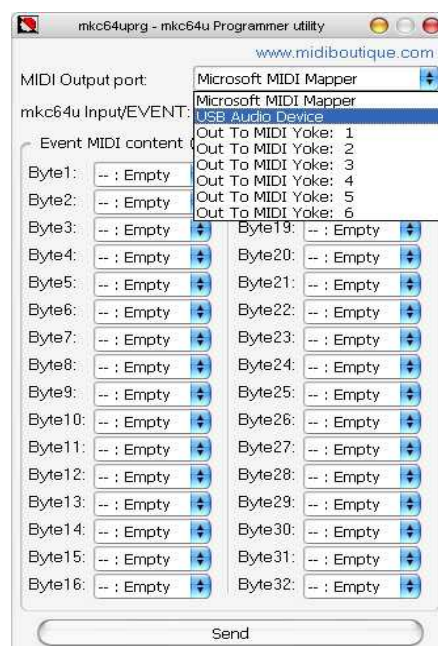
6. Select the event you want to program. All user programmable MIDI events are listed in '**mkc64u** Input/EVENT' drop-down list. There are three groups of events: ON-event for contact inputs, OFF-event for contact inputs and CHANGE events for potentiometer inputs.
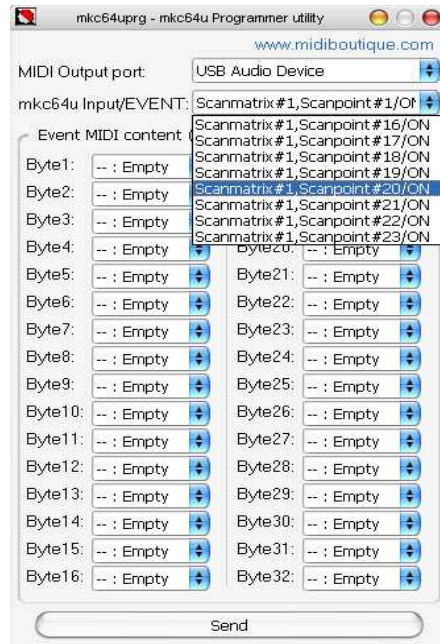
7. Choose desired event content byte-by-byte by selecting byte values.
   <u>NOTES:</u>
   Bytes that have assigned 'Empty' will be reset to value of FF.
   > *For any event, each byte that has been assigned HEX value of FF will be ignored and won't be producing MIDI output traffic.*
   > *For potentiometer events each byte that has been assigned HEX value of F6 will be replaced by potentiometer data in range 0-127 as read from potentiometer.*
   > *Hence, **System Reset** MIDI message (HEX FF) cannot be programmed in any event and **Tune Request** MIDI message (HEX F6) cannot be programmed in potentiometer events.*
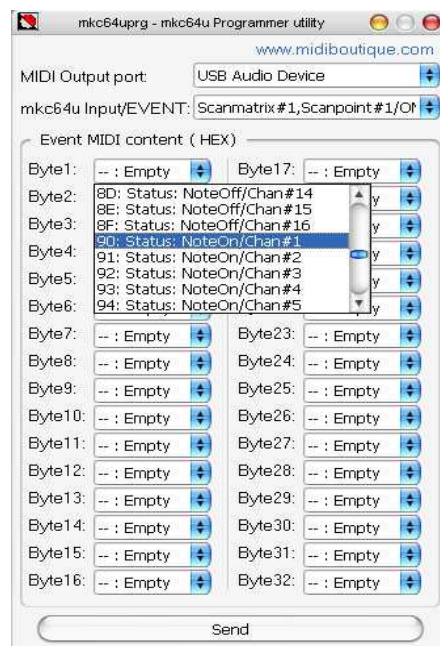
8. After the MIDI string has been configured, press once the 'Send' button at the bottom.
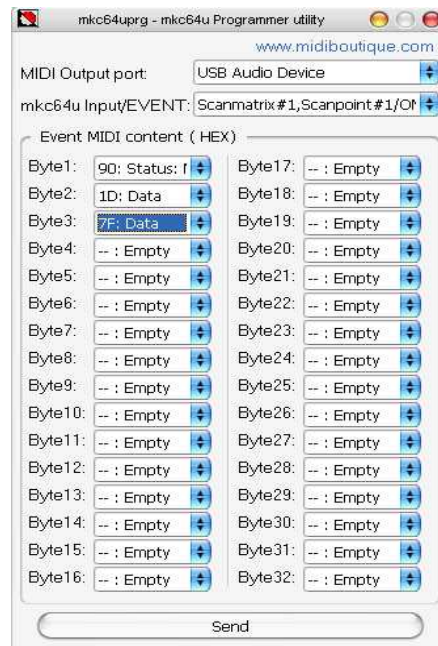
9. Repeat steps 6 .. 9 as many times as necessary for programming desired events.
10. The unit is programmed and can be used.
    NOTES:
    The programmed MIDI strings will be permanently kept in non-volatile (power-independent) memory and won't change until next programming.
    *More than one **mkc64u** unit can be chained together and will be programmed simultaneousness as each **mkc64u** would retransmit whatever MIDI traffic it gets.*

Programming protocol

The programming protocol includes three parts:

Header message.
This is *optional* 10-byte System exclusive message that only causes reset of program input queue (the 32-byte input buffer that receives the MIDI string to be programmed). During reset, all the 32 bytes of Program buffer are set to HEX value of FF. If the string to be programmed is 32 bytes long, the header message can be omitted as the input queue will be updated entirely . The format of this message is:

Header message - 10 bytes (all shown in HEX format)

```
F0 - SysEx start
00 - first ID
21 - sec. ID  (MGB)
7F - thd. ID  (MGB)
0C - Device ID (mkc64u)
00 - Device sub-ID (message ID = 00: Reset buffer)
xx - future use byte (set to 00)
xx - future use byte (set to 00)
xx - future use byte (set to 00)
F7 - SysEx end
```

The MIDI string itself.
It can be between 0 and 32 bytes long. In case of 0 bytes, the MIDI string for the programmed event is considered empty and this event won't be producing any MIDI output. This has the same effect as programming HEX FF value to all 32 bytes of this

string. This feature is suitable for programming switches that would transmit Program change messages upon contact make and nothing upon contact break.
If an MIDI string is longer than 32 bytes, only the last 32 bytes will take place. Any MIDI bytes of any order can be transmitted. The only two values that have special meaning are HEX FF (MIDI Reset) and HEX F6 (Tune Request). These are used for inserting special parameters in MIDI string. Refer to notes after the Step 8 of step-by-step programming sequence described above about these special considerations.

Footer message.
This is *obligatory* 10-byte System exclusive message that passes to **mkc64u** the table entry number to be programmed, and invokes the memory-write routines to copy the input buffer content to proper table entry.

Footer message - 10 bytes (all shown in HEX format)

```
F0 - SysEx start
00 - first ID
21 - sec. ID  (MGB)
7F - thd. ID
0C - Device ID (mkc64u)
01 - Device sub-ID (message ID = 01: Store buffer)
ll - Entry number, LSB (7-bit value 00..7F)
mm - Entry number, MSB (7-bit value 00..07)
xx - Future use byte (set to 00)
F7 - SysEx end
```

Appendix B. mkc64u – factory MIDI implementation

Scan matrix					
Input#	Control type	MIDI message on make	MIDI message on break	MIDI channel	Comment
1	momentary contact/switch	NoteOn #36	NoteOff #36	1	
2	momentary contact/switch	NoteOn #37	NoteOff #37	1	
3	momentary contact/switch	NoteOn #38	NoteOff #38	1	
4	momentary contact/switch	NoteOn #39	NoteOff #39	1	
5	momentary contact/switch	NoteOn #40	NoteOff #40	1	
6	momentary contact/switch	NoteOn #41	NoteOff #41	1	
7	momentary contact/switch	NoteOn #42	NoteOff #42	1	
8	momentary contact/switch	NoteOn #43	NoteOff #43	1	
9	momentary contact/switch	NoteOn #44	NoteOff #44	1	
10	momentary contact/switch	NoteOn #45	NoteOff #45	1	
11	momentary contact/switch	NoteOn #46	NoteOff #46	1	
12	momentary contact/switch	NoteOn #47	NoteOff #47	1	
13	momentary contact/switch	NoteOn #48	NoteOff #48	1	
14	momentary contact/switch	NoteOn #49	NoteOff #49	1	
15	momentary contact/switch	NoteOn #50	NoteOff #50	1	
16	momentary contact/switch	NoteOn #51	NoteOff #51	1	
17	momentary contact/switch	NoteOn #52	NoteOff #52	1	
18	momentary contact/switch	NoteOn #53	NoteOff #53	1	
19	momentary contact/switch	NoteOn #54	NoteOff #54	1	
20	momentary contact/switch	NoteOn #55	NoteOff #55	1	
21	momentary contact/switch	NoteOn #56	NoteOff #56	1	
22	momentary contact/switch	NoteOn #57	NoteOff #57	1	
23	momentary contact/switch	NoteOn #58	NoteOff #58	1	
24	momentary contact/switch	NoteOn #59	NoteOff #59	1	
25	momentary contact/switch	NoteOn #60	NoteOff #60	1	
26	momentary contact/switch	NoteOn #61	NoteOff #61	1	
27	momentary contact/switch	NoteOn #62	NoteOff #62	1	
28	momentary contact/switch	NoteOn #63	NoteOff #63	1	
29	momentary contact/switch	NoteOn #64	NoteOff #64	1	
30	momentary contact/switch	NoteOn #65	NoteOff #65	1	
31	momentary contact/switch	NoteOn #66	NoteOff #66	1	
32	momentary contact/switch	NoteOn #67	NoteOff #67	1	
33	momentary contact/switch	NoteOn #68	NoteOff #68	1	
34	momentary contact/switch	NoteOn #69	NoteOff #69	1	
35	momentary contact/switch	NoteOn #70	NoteOff #70	1	
36	momentary contact/switch	NoteOn #71	NoteOff #71	1	
37	momentary contact/switch	NoteOn #72	NoteOff #72	1	
38	momentary contact/switch	NoteOn #73	NoteOff #73	1	
39	momentary contact/switch	NoteOn #74	NoteOff #74	1	
40	momentary contact/switch	NoteOn #75	NoteOff #75	1	
41	momentary contact/switch	NoteOn #76	NoteOff #76	1	
42	momentary contact/switch	NoteOn #77	NoteOff #77	1	
43	momentary contact/switch	NoteOn #78	NoteOff #78	1	

44	momentary contact/switch	NoteOn #79	NoteOff #79	1	
45	momentary contact/switch	NoteOn #80	NoteOff #80	1	
46	momentary contact/switch	NoteOn #81	NoteOff #81	1	
47	momentary contact/switch	NoteOn #82	NoteOff #82	1	
48	momentary contact/switch	NoteOn #83	NoteOff #83	1	
49	momentary contact/switch	NoteOn #84	NoteOff #84	1	
50	momentary contact/switch	NoteOn #85	NoteOff #85	1	
51	momentary contact/switch	NoteOn #86	NoteOff #86	1	
52	momentary contact/switch	NoteOn #87	NoteOff #87	1	
53	momentary contact/switch	NoteOn #88	NoteOff #88	1	
54	momentary contact/switch	NoteOn #89	NoteOff #89	1	
55	momentary contact/switch	NoteOn #90	NoteOff #90	1	
56	momentary contact/switch	NoteOn #91	NoteOff #91	1	
57	momentary contact/switch	NoteOn #92	NoteOff #92	1	
58	momentary contact/switch	NoteOn #93	NoteOff #93	1	
59	momentary contact/switch	NoteOn #94	NoteOff #94	1	
60	momentary contact/switch	NoteOn #95	NoteOff #95	1	
61	momentary contact/switch	NoteOn #96	NoteOff #96	1	
62	momentary contact/switch	NoteOn #97	NoteOff #97	1	
63	momentary contact/switch	NoteOn #98	NoteOff #98	1	
64	momentary contact/switch	NoteOn #99	NoteOff #99	1	

Buttons

Pin#	Control type	MIDI message on change		MIDI channel	Comment
1	contact On/Off	CC#80 On	CC#80 Off	1	
2	contact On/Off	CC#81 On	CC#80 Off	1	
3	contact On/Off	CC#82 On	CC#80 Off	1	
4	contact On/Off	CC#83 On	CC#81 Off	1	
5	contact On/Off	CC#84 On	CC#82 Off	1	

Potentiometers

Pin#	Control type	MIDI message on change	MIDI channel	Comment
1	potentiometer or control voltage 0 .. +5V	CC7 (Volume)	1	
2	potentiometer or control voltage 0 .. +5V	CC7 (Volume)	2	
3	potentiometer or control voltage 0 .. +5V	CC7 (Volume)	3	
4	potentiometer or control voltage 0 .. +5V	CC7 (Volume)	4	
5	potentiometer or control voltage 0 .. +5V	CC7 (Volume)	5	
6	potentiometer or control voltage 0 .. +5V	CC7 (Volume)	6	
7	potentiometer or control voltage 0 .. +5V	CC7 (Volume)	7	
8	potentiometer or control voltage 0 .. +5V	CC7 (Volume)	8	

Appendix C. Wiring diagram for Scan matrix wiring
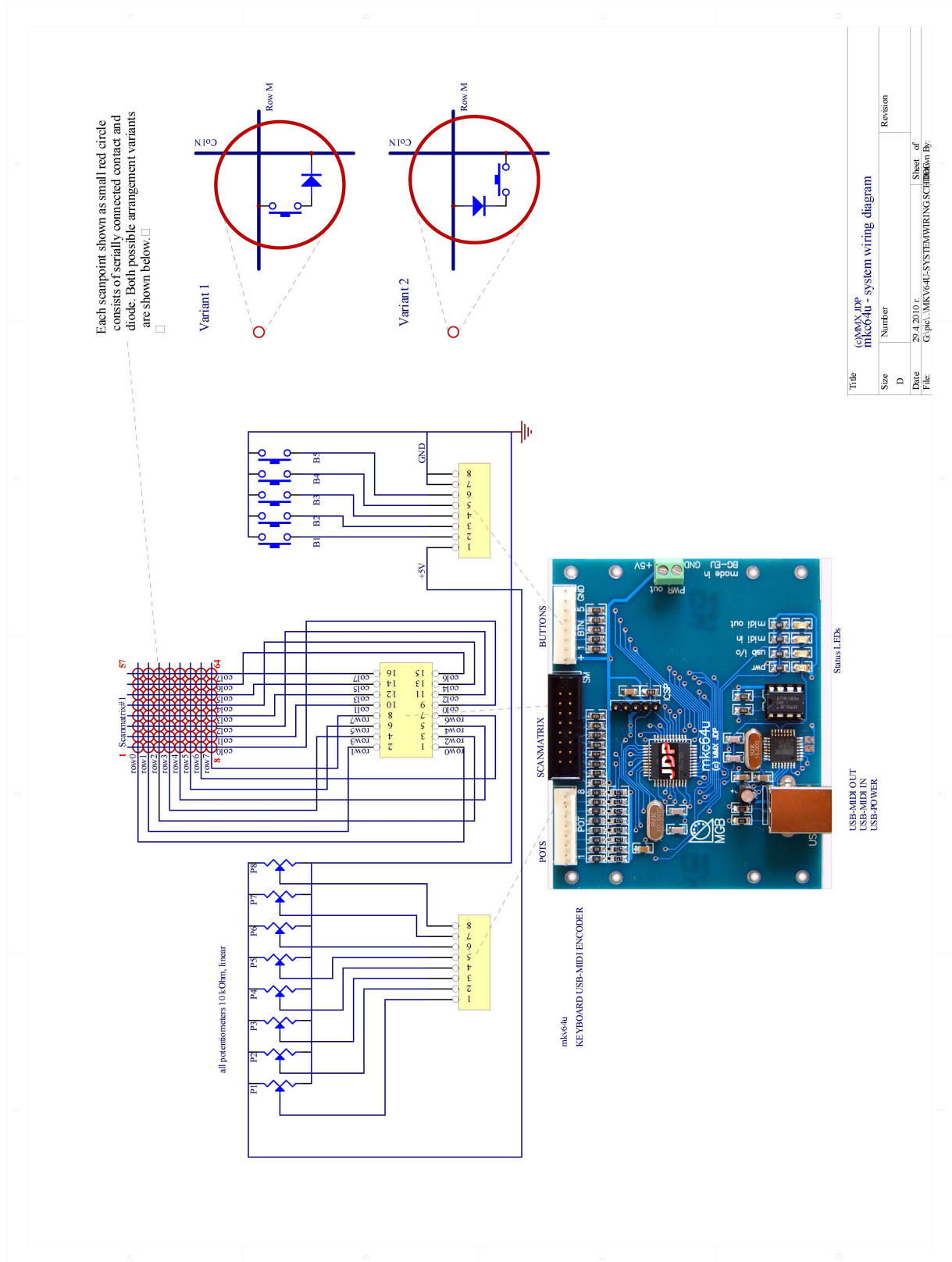
NOTE:
More detailed copy of this diagram is available on our site in **.pdf** format!

info@midiboutique.com
www.midiboutique.com

Appendix D. Wiring diagram for mkc64u-bundle#1

Appendix E. Wiring diagram for mkc64u-bundle#2

NOTE:
More detailed copy of this diagram is available on our site in **.pdf** format!

info@midiboutique.com
www.midiboutique.com