USB Stick Disguised as a Keyboard

An emulated keyboard on USB, with an AVR controller

By **Markus Hirsch**
(Germany)

To the average e-enthusiast, the RS232 interface always appears friendlier than the USB port. However, with a small 8-bit microcontroller, nowadays it's possible to implement the USB bus in device mode. The advantages are obvious. First, you save the cost of a USB/RS232 converter. Then, USB devices such as keyboards can be easily emulated without needing a separate driver, meaning you are able to design a programmable USB keyboard at minimum hardware cost and software effort. And as you will discover here, our versatile USB Keyboard Stick not only helps to build useful designs—it's also fun to use by itself!

Hardware—what this really is

As already mentioned, the hardware is plain sailing and inexpensive (**Figure 1**). The cornerstone of the USB Keyboard Stick is an Atmel AVR Attiny85 microcontroller. By means of two 1N4148 diodes, D1 and D2, the 5 V supply voltage is reduced to about 3.6 V, with capacitor C1 doing the smoothing. The 1.5-kΩ resistor (R3) sets the speed identification of the USB—in this case we have a low-speed device. A 16-MHz quartz crystal generates an accurate clock signal. It's also possible to omit this component and rely on the internal oscillator, if configured to operate at 16.5 MHz. An LED with its series resistor R4 is used as a minimal status indicator.
The circuit can be built quickly on a piece of prototyping board with a USB-A connector, keeping it compact (see **Figure 2**). An SMD version and a 'real' etched board will be even more compact and can be easily housed in a junked USB drive enclosure.

Owing to space constraints, an SPI port header was omitted here. The ATtiny85 micro can be programmed using just three wires, including the Debug Wire (dW) and the power supply. To do so, feel free to connect these lines to the controller with improvised cables and test clips.

Software: what this claims to be

The software was written in C using the well-known (and free!) Atmel Studio 6 suite [1]. The basis of the software is the USB Device Stack. It's also available as a self-contained, free library. V-USB can be downloaded directly from their developer's website [2], where it is graced by a dedicated microsite [3]. There you may also find some useful examples, which are great as starting points for creating USB projects without having to delve deep into horrid protocol stuff.

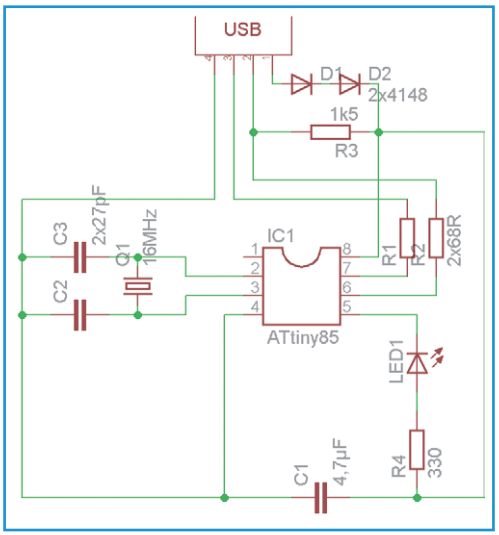

Figure 1.
USB Keyboard Stick schematic.

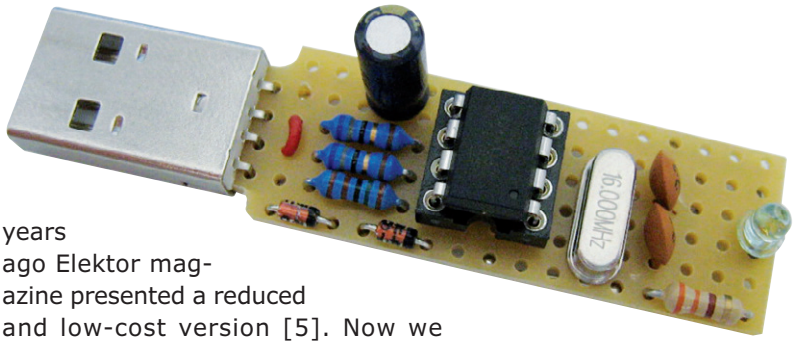

Figure 2.
The USB Keyboard Stick can be easily built on a prototyping board.

Once the USB stack is properly configured, the AVR microprocessor can be detected by most operating systems, without an extra driver. As a consequence, the indication of the specific LEDs used for the Caps Lock, Num Lock and Scroll Lock keys gets sent to the stick. In order to emulate keystrokes, the key codes are provided to the PC, as shown in **Tables 1, 2** and **3**. These codes may include a modifier—for instance, to differentiate between an "A" (capital) and an "a" (lower case). Beware that these codes do not correspond to the equivalents of the same characters in ASCII.

Below, two applications of the USB Keyboard Stick are described. The AVR Studio code and hex files for both applications are available at the Elektor.LABS website [6].

Function #1: USB Annoy-a-Tron: an essential evil genius tool

Annoy-a-Tron is a popular hide-and-seek game designed by ThinkGeek [4]. Three years ago Elektor magazine presented a reduced and low-cost version [5]. Now we introduce the USB Annoy-a-Tron, based on a vastly different concept and featuring *improved annoying capabilities* that will drive your workmates and relatives completely crazy. A typical and well known joke is to swap the letters "v" and "b" in a standard keyboard (physically), as they are close together. The USB Annoy-a-Tron goes way further, let's see what it has to offer...

The USB Annoy-a-Tron is basically a permanent Caps Lock. Once the stick is plugged into the PC, it will be recognized as a keyboard. Afterwards, the PC will provide the status of the three LEDs (Caps, Num and Scroll Lock). The stick will now evaluate the bit that represents the Caps Lock key. Once you press it again to deactivate it, the USB Annoy-a-Tron will simulate a virtual keystroke. As a result, Caps Lock will

Table 1. Coding of the USB keyboard characters (standard keys)

A	B	C	D	E	F	G	H	I	J	K	L	M	N
4	5	6	7	8	9	10	11	12	13	14	15	16	17

O	P	Q	R	S	T	U	V	W	X	Y	Z	1	2
18	19	20	21	22	23	24	25	26	27	28	29	30	31

3	4	5	6	7	8	9	0	Enter	Space	Esc	Caps Lock	Num Lock	Scroll Lock
32	33	34	35	36	37	38	39	40	44	41	57	83	71

Table 2.
USB keyboard characters coding (special keys)

Ctrl	Shift	Alt	GUI	R. Ctrl	R. Shift	R. Alt	R. GUI
0x01	0x02	0x04	0x08	0x10	0x20	0x40	0x80

Table 3.
USB keyboard LED bit codes

Num Lock	Caps Lock	Scroll Lock
0x01	0x02	0x04

be always on (**Figure 3**), and the user will not be able to do much else than suffer the consequences.

Now you only have to insert the stick into a USB port on the PC that's not in sight, like at the back side. And if you are evil enough, it's even more exciting to install it inside the computer case by means of a small cable adaptor.

Function #2:
Masterkey: stay safe like a pro

The USB Annoy-a-Tron is an amusing prank, but admittedly it's not terribly useful considering the possibilities of such a device. Thus, the second function is a password storage system. Passwords are great and much needed, but also a source of problems. Most of you will have several passwords to ensure good encryption levels in various places on the web, but they may not be so easy to remember. For this purpose, there are password managers out there like KeePass. But again, a 'strong' (meaning lengthy and idiosyncratic) master password is required. Here is where the USB Masterkey comes in. Although it's not a novel concept, our version introduces an essential feature not present in other USB password storage devices. To 'release' the password it's not enough to just plug the USB into the PC, as that would be very unsafe in case you lose the stick. Instead, to retrieve the password you'll have to provide a combination of the three keyboard LEDs, in a required order. For instance, in our example the user has to follow the steps according to the file "code.h", as part of the firmware. The steps noted here are: 1. Num Lock, 2. Num Lock + Caps Lock, 3. Num Lock, afterwards the password will be released. The first step should already match the combination before starting the whole process. This means that Num Lock (step 1) has to be on even before connecting the USB. Then you have to press Caps Lock, so you get Num Lock + Caps Lock together (step 2). Then, you press Caps Lock again so the result is that only Num Lock is on (step 3). Now the password will be released. Such a short sequence can be easily remembered. Should you provide an incorrect combination, you're forced to remove the stick from the PC, and connect it back again. Needless to say, this combination of steps can be manually modified in the file called 'code.h' according to our preferences.

Figure 3.
The Caps Lock on the keyboard will flash again. What's funny to you isn't to the victim.

Over to you

All files needed to replicate the project are available at the web page allocated to the project on the Elektor**.**LABS website [6], including the schematics and the board as EAGLE files.

Here we've shown that implementing an emulated USB keyboard is not difficult at all, and the possibilities of using a 'disguised' microcontroller in your own designs are almost endless.

As a suggestion for a third application, consider storing different passwords in one USB stick, and set different combinations of keystrokes to retrieve them. Well, just in case you want to take the useful way. If not, any other funny device would be great. Gags are always welcome at the Elektor**.**POST desk—unless we are victimized, of course.

(120583)

Internet Links

[1] www.atmel.com/microsite/atmel_studio6
[2] www.obdev.at
[3] www.obdev.at/products/vusb/index.html
[4] www.thinkgeek.com/product/8c52/
[5] www.elektor.com/090084
[6] www.elektor-labs.com/120583

COMPONENT LIST

Resistors
R1,R2 = 68Ω, 0.25W, 1%
R3 = 1.5kΩ, 0.25W, 1%
R4 = 330Ω, 0.25W, 1%

Capacitors
C1 = 4.7µF 100V electrolytic
C2,C3 = 27pF 100V ceramic

Semiconductors
D1,D2 = 1N4148
LED1 = LED, 5mm, yellow

Miscellaneous
IC1 = ATtiny85, programmed
X1 = 16MHz quartz crystal
K1 = USB-A plug