Miguel Manual

edition 0 for release 0.0
updated 06 Oct 2016

Ineiev ineiev@gnu.org

This manual is for miguel 0.0 (edition 0, updated 06 Oct 2016), which is a keyboard controller meant to be TEMPEST-resistant.

Table of Contents

1 Introduction

This project started as a result of deliberations about generating computer passwords. I was establishing a procedure for myself similar to the one suggested by Diceware (http://world.std.com/~reinhold/diceware.html). Diceware documentation recommends using dice as the random number generator, because electronic random number generators may be far from ideal. However, this method has one more advantage: it is hard to snoop on. It emits practically nothing informative in the radio range, and the optical radiations can be relatively easily controlled (e.g. with screens and curtains).

So far, so good. As long as I don't output the password on the monitor (which is not likely to be TEMPEST-resistant), the password mustn't leak—of course, I assume that the attacker intercepts the radiowaves from the computer—and that the keyboard doesn't emit signals that may identify the keys pressed.

A quick search revealed *Compromising Electromagnetic Emanations of Wired and Wireless Keyboards*, an article by Martin Vuagnoux, Sylvain Pasini. It suggested that my PS/2 keyboard does emit quite a bit of identifying signals:

- The PS/2 interface proved highly vulnerable.
- The keyboard matrix radiates patterns that can be easily intercepted (due to their high-frequency spectrum) and decoded (because firmware wasn't written with security in mind).
- The signals are especially strong when the power rails are connected (through the computer) to the mains power line.

The next step is to figure out if it's feasible to resolve these issues. Monitor electronics are rather complicated and need high-speed processing, but keyboard controller is no rocket science at all: a low cost microcontroller with some simple circuits can easily do the job. And after you decide to replace the controller, all aforementioned vulnerabilities are quite fixable (and probably some other as well).

- Low-pass filters for the signals scanning the keyboard matrix suppress most of the emissions from the matrix.
- Random and continually changing sequence of scanning signals prevents usefully interpreting even those emissions.
- Enclosing the entire controller in a conducting screen reduces the emissions from the controller itself.
- Migrating to USB makes the connection less vulnerable, and a high-quality cable further blocks the emanations from the USB interface.
- Powering up the device from a battery means that there is no leak from that side.

These measures don't require any changes in the computer, not even in its software. The device would act exactly like the standard USB keyboard.

Using an encrypted channel would provide even more security, but this would require modified bootloader (definitely) and keyboard driver (most probably); also, it is not clear if the currently used microcontroller has a sufficient amount of program memory.

1.1 Controller Specifications

Maximum column number
: 27

Maximum row number
: 8

Sampling period
: 7 ms

Voltage supply
: 3.4 V to 6 V

Power consumption
: active: 20 mA (no LED on), 30 mA (all LEDs on); sleeping: 20 uA

2 Schematics

Schematically, the keyboard controller features a 8-bit microcontroller, a noise generator used to get random numbers, a demultiplexer to add more output lines, lowpass filters connected to the keyboard matrix, switches to control keyboard LEDs, linear regulators to supply power, USB-specific circuits, and connectors.

The complete schematic diagram is provided in `hw/miguel.sch` in gschem format and exported to `hw/schematics/miguel.pdf` and `hw/schematics/miguel.ps` in the distribution tarball. The figures below are excerpts of that complete diagram.

2.1 MCU

The central part is occupied by a 32-pin AVR MCU in the TQFP-32 package (U1) Figure 2.1. ATmega48, ATmega88, ATmega168, ATmega328 will fit. ATmega8 would do (I use no new features), but it wasn't qualified for 12MHz @ 3.3V.

The amount of program memory in ATmega48 is barely sufficient to implement RNG, keyboard scanning and LEDs control, and standard USB keyboard protocol. This leaves no room for encryption, so it may make sense to use ATmega88 or a processor with higher amount of memory.

The ADC is used to poll the voltage level of power supply; another input is connected to the voltage multiplier that produces the voltage level necessary for the noise-generating transistor ([vmult], page 3), however, this input isn't used yet.

The processor clock is generated with a 12MHz crystal ZQ1.

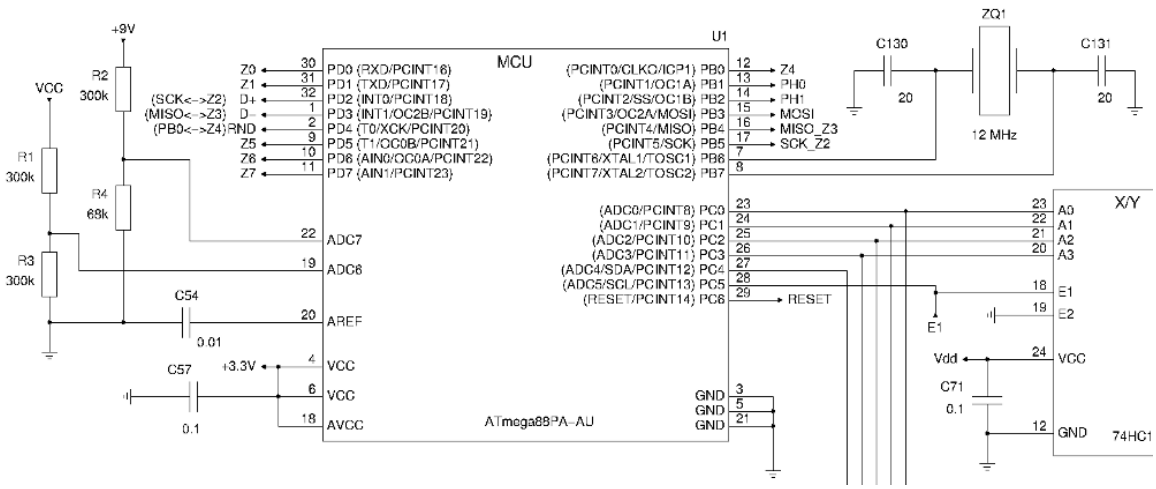

Figure 2.1: MCU connections.

2.2 Noise Generator

D20-D22 make voltage multiplier, it's fed with PWM produced by Timer1; the output voltage level (or rather the output current) can be changed within some extents modifying

the frequency. The duty cycle of the same signal regulates the bias for the comparator U6 through Q8 and Q9 (Figure 2.2).

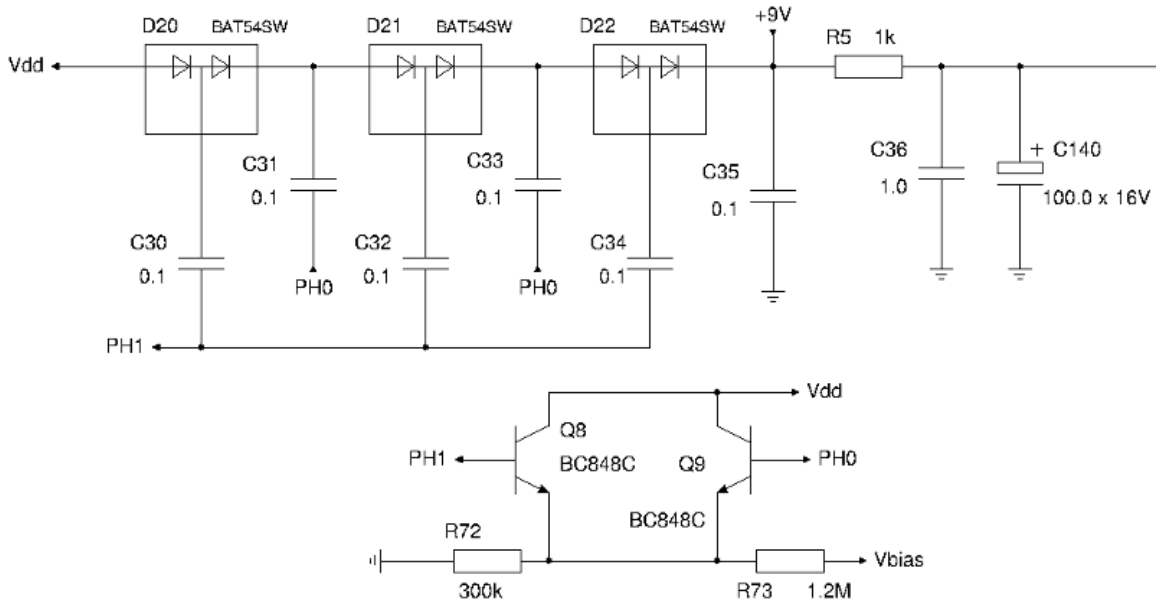

Figure 2.2: Charge pump and bias generator.

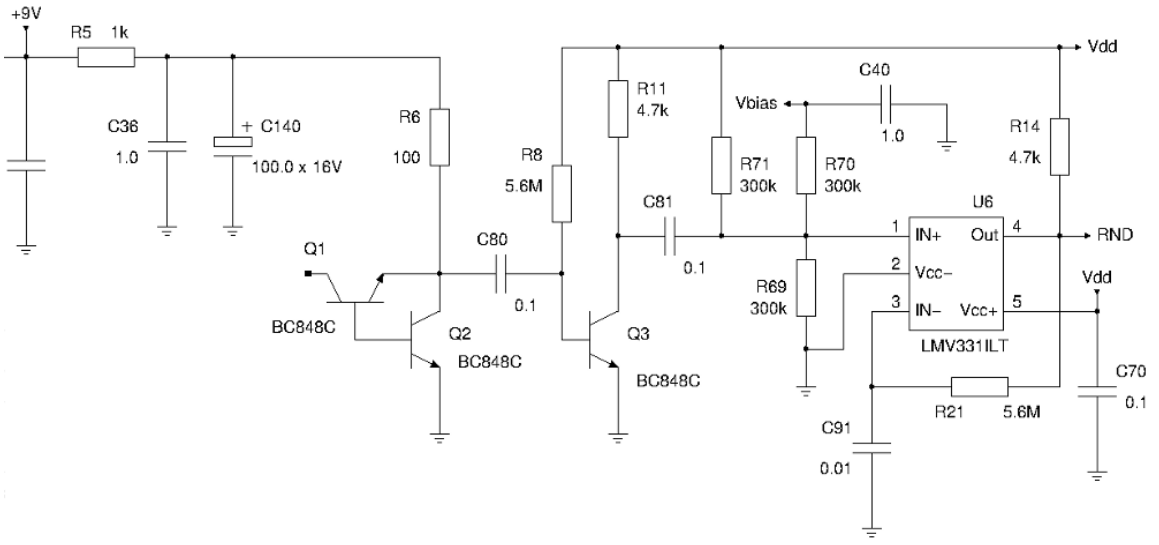

Figure 2.3: Noise generator.

The base-emitter junction of Q1 is the primary source of the noise, Q2, Q3, and U6 amplify it (Figure 2.3). The output of U6 comes to the T0 pin of U1, Timer0 counts 1-to-0 transitions. The pull-up resistor R14 (4.7k) gives a time constant of about 200ns, so 12MHz

sampling rate must be adequate. The least significant bit of Timer0 counter produces decent random numbers with rates up to 100 kbit/s.

When active, the generator draws about 10 mA from the Vdd source.

2.3 Demultiplexers

In order to workaround the insufficient number of input-output lines in the MCU, two demultiplexers are used (Figure 2.4).

U2 and U3 control the LEDs and scan the keyboard matrix. Note that the address signals of U3 are transposed for PCB layout considerations, e.g. the A0 signal of U2 is shared with the A3 signal of U3. The Yn and Xn signals are named from MCU's point of view: when it enables U3 and outputs 13 to the least significant half of PORTC, Y(13 + 16) = Y29 is drawn low.

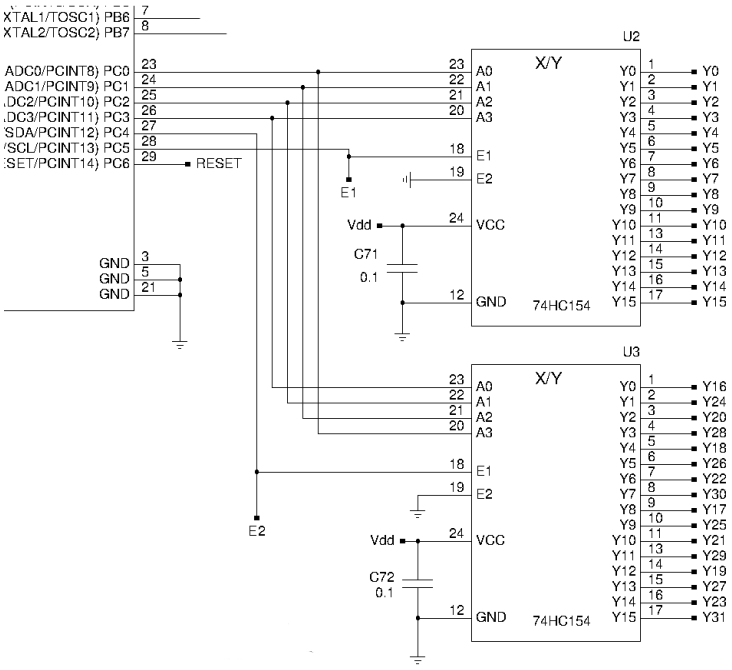

Figure 2.4: Demultiplexers.

2.4 Scanning Circuits

The signals for scanning the keyboard matrix go through low-pass filters (Figure 2.5); then the BAS16VV diodes are used to switch off the inactive lines (i.e. when there is a low level in any column, it passes to the row, the high levels in other columns are ignored). By the way, this is better than switching the columns between high impedance state and low level (and then adding low-pass filters): if more than one key is pressed in the row, the filter is discharged through the keyboard matrix; with our circuit, these currents flow within the controller PCB (which is smaller and enclosed in a screen).

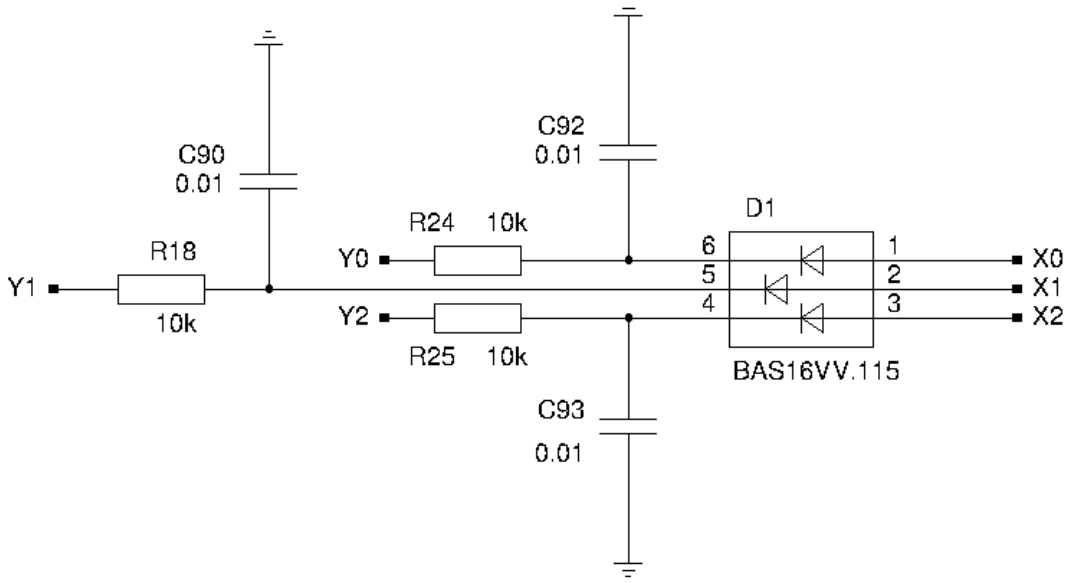

Figure 2.5: Column lines.

The signals from the matrix come to MCU pins (mostly PIND), which also have low-pass filters and are pulled up to MCU power supply(Figure 2.6).

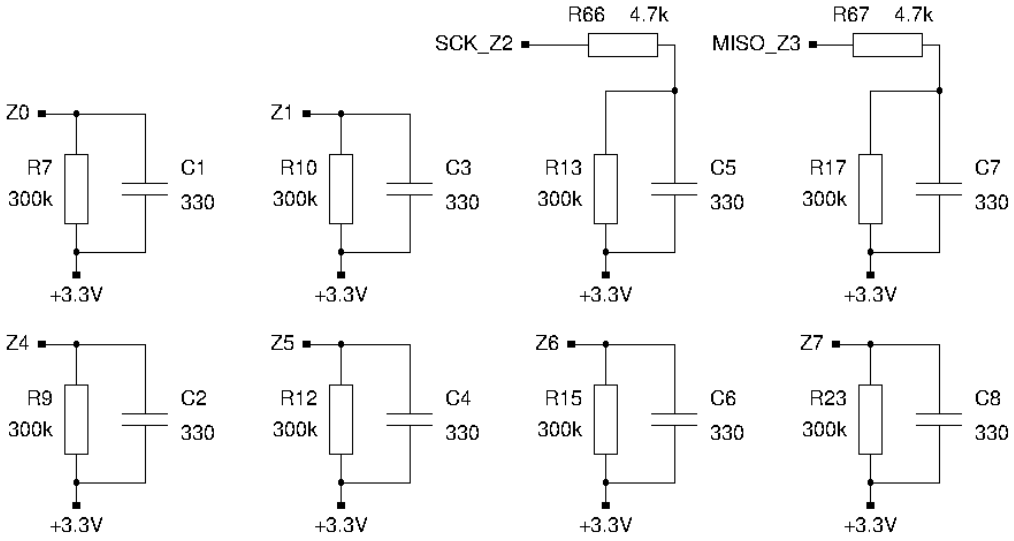

Figure 2.6: Row lines.

2.5 LED Switches

There are 4 signals for LEDs: NUM, CAPS, SCROLL and an extra LED to signal RNG failure, battery state and so on. The switches to control them are built on Q4-Q7, D3 and a part of D9 (Figure 2.7). When no pulses come to the inputs of those circuits, the LEDs glow; to switch them off, one needs to sample them continually. The time constant for switching on is about 0.1 s, so it mustn't be possible to leak too much information with them.

The outputs are to be connected to the cathodes, 3.3V for the anodes is provided on pins 12 and 13 of X1.

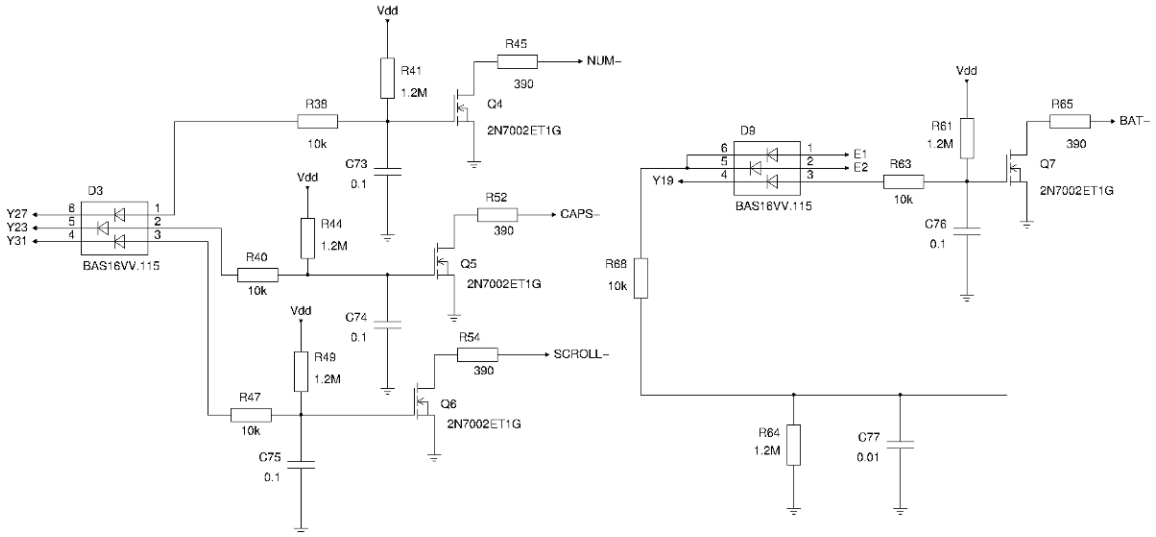

Figure 2.7: LED controls.

2.6 Power Distribution

The device is to be powered with 3 AA-size batteries. As an alternative, it might take power from USB, but this may significantly decrease TEMPEST resistance since the information may leak through power wires. The measured consumption is about 20 mA with all LEDs off; in the sleeping mode, it draws about 0.02 mA.

There are two linear regulators: U5 powers the MCU, U4 powers all the rest. U4 is switched off when both chip select signals for U2 and U3 are low (the wired AND is built on D9): that is, I only want to draw one of U2 and U3 outputs low at once.

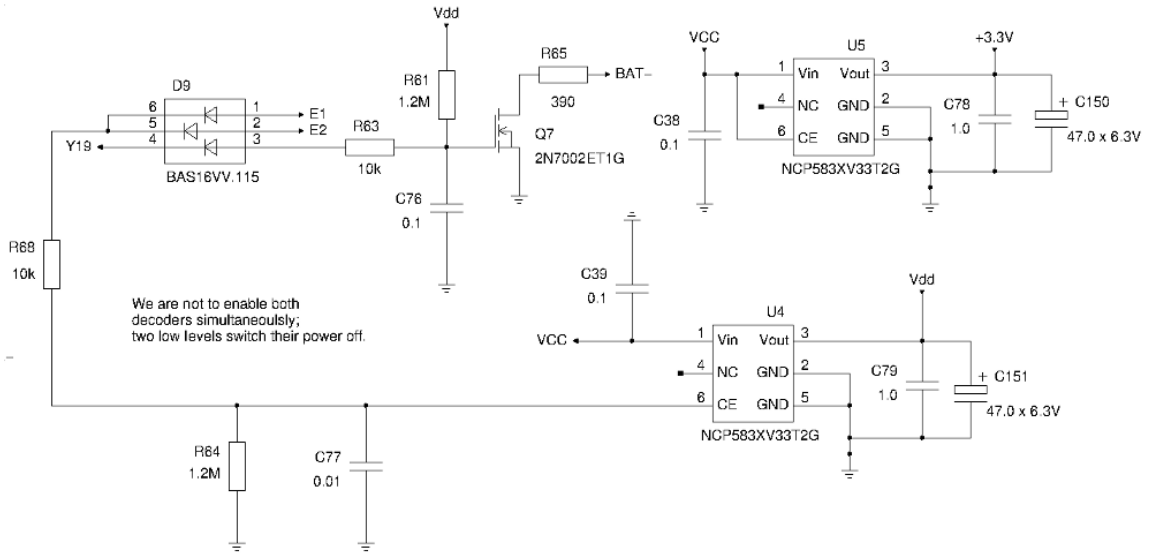

Figure 2.8: Power distribution.

2.7 USB Circuits

The USB protocol is implemented with V-USB; in addition to the usual resistor network used for self-powered devices, I use the USB power line to detect USB connection.

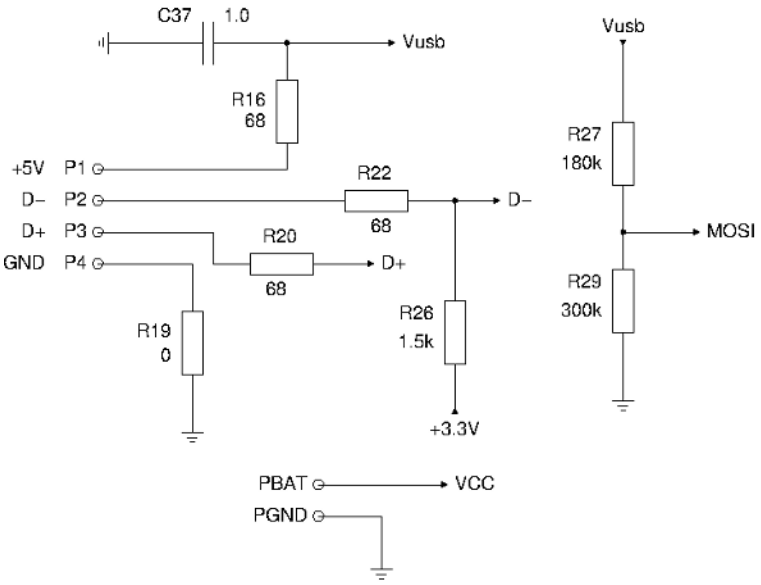

Figure 2.9: USB circuits.

2.8 Connectors

The schematic diagram shows two connectors, 30-pin two-row X1 and 15-pin single-row X2. The first 6 pins of X1 form the conventional programming header for the MCU, so actually mounting these pins makes the things considerably more convenient; however, filling the rest pins on the board provides no crucial advantage over soldering the wires directly to the holes.

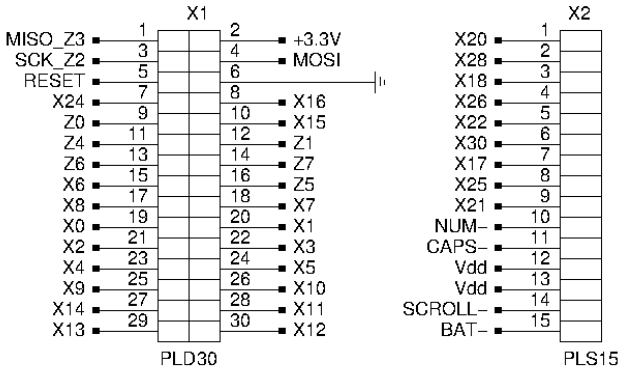

Figure 2.10: Connectors.

3 Assembly

The controller is built on a two-layer PCB. The layout in GNU PCB format comes as `hw/miguel.pcb`. The `hw/miguel` directory of the tarball contains Gerber files. You'll need `miguel.top`, `miguel.bottom`, `miguel.plated-drill.cnc` and `miguel.fab`, the silkscreen and mask layers are not used.

3.1 Assembling the Board

This section mostly applies to the manual procedure.

- The components with lower height are mounted first, otherwise the board doesn't lie firmly on the desk.
- For the crystal, cushions of solder are grown on the board first, then the part is attached and soldered from the sides.
- Some of the triple diodes will almost certainly end up looking in the wrong direction.
- Since many parts are small, shorts are common; you should check them visually, and test the resistance from the power contact to the ground before applying the power for the first time.

  This is especially true for the triple diodes. I recommend testing the conductors connected to them: the respective lines should indicate the diode, the lines of the neighboring devices should be disconnected (unless joint with conductors like in D9).

When all the components are mounted, attach the USB cable and connect the power pin to the USB power line until the board is finally enclosed in screen and powered with batteries.

Figure 3.1: Assembled controller board.

Fix the USB cable on a polygon with a piece of wire or two; the wires of the cable tend to break soon unless the end is fixed Figure 3.2.

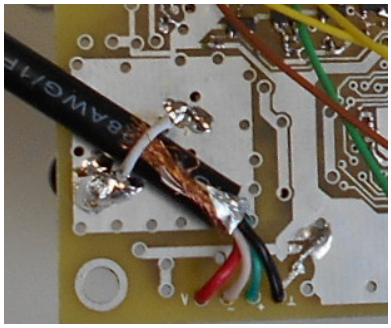

Figure 3.2: Initial cable tie.

3.2 Connecting to Keyboard

Keyboards are usually screwed from the bottom side, and each key has a small flexible dome between it and the keyboard matrix. These small parts are easy to lose (unless they are connected in a single field), so you should prepare a box for them beforehand. When assembling the keyboard back, put something under the keyboard so that the keys are not pressed Figure 3.3; if they are, these rubber parts may move from their places.

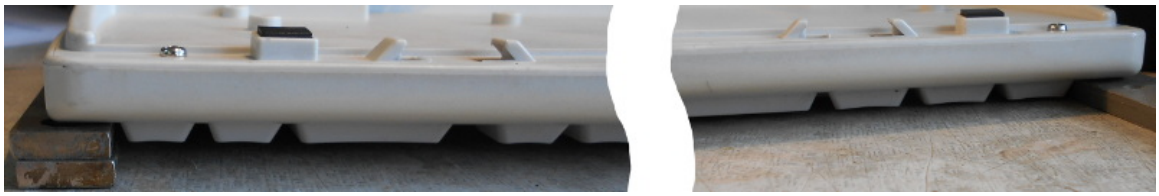

Figure 3.3: Supporting keyboard.

Columns are easily distinguished from the rows: they are laid out on the other film, and the number of the rows is much smaller (our controller supports no more than 8).

Now extract the controller board of the keyboard. Cut the conductors to disconnect the processor from the rest of the board, remove unnecessary parts like capacitors, drill holes and mount the additional LED (usually it fits between the Num Lock and the Caps Lock). Drill holes and solder wires to column and row lines (the PS/2 cable of the keyboard provides just sufficient amount of wires to connect the new controller with the old board).

If the pads of the matrix are longer than needed, and the board is protected with a mask, cut off the excessive parts of the pads on the film, they may short the wires.

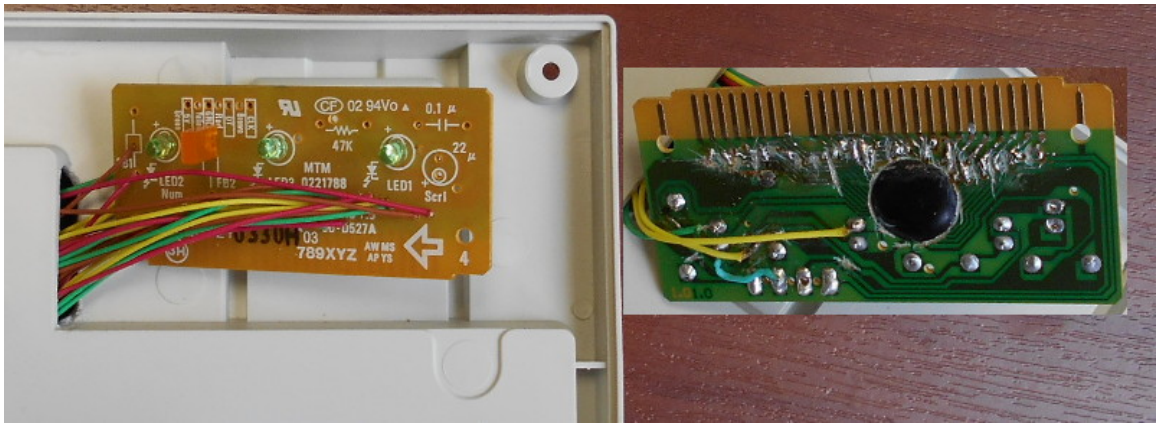

Figure 3.4: Connecting to old controller board.

The new controller isn't likely to fit in the case of the keyboard, so cut a hole to output the wires; also, cut a whole for the additional LED.

Figure 3.5: LED positions.

Connect the new controller to the old controller board: LED cathodes to the respective signals, LED anodes to the '`Vdd`' line, the rows to '`Zi`', and the columns to '`Xi`'; test the connections, check for shorts.

Assemble the keyboard. An extra caution should be taken to make sure that the contacts of the matrix are pressed tightly against the old board. If they aren't, the new controller usually doesn't respond to any keypresses.

3.3 Programming MCU

The leading software for uploading firmware into AVR seems to be `avrdude` (https://savannah.nongnu.org/projects/avrdude). Sometimes I still use `uisp`, but it is not maintained anymore and needs patching to support newer chips.

I program MCUs with a self-made ByteBlaster, but `avrdude` supports many other programmers as well. Please see `avrdude` documentation for details on how to upload firmware.

A fresh MCU will use its internal clock generator. In order to make it work with the crystal, you'll need to change its fuse bytes, in case of ATmega88PA and similar devices you set the low fuse to '`EF`', leaving the rest as per factory default.

3.4 Writing Keytable

Configure firmware with the '`--enable-raw-scancodes`' option, build and upload it. Connect the keyboard to your PC, build and run the `test-usb` utility coming with the miguel package (this may require the root access). Take doc/keys.txt and press each key on the keyboard[1], writing the received raw scancodes at the start of the respective line instead of the decimal code, like

```
66 04 Keyboard a and A4 31
73 05 Keyboard b and B 50
5C 06 Keyboard c and C4 48
5E 07 Keyboard d and D 33
60 08 Keyboard e and E 19
...
```

Then remove the ends of lines after the key codes:

```
66 04
73 05
5C 06
5E 07
60 08
...
```

When you are done, run the `tab`, another utility coming with miguel:

```
./tab < keys.txt > keytab.h
```

The utility generates a scancode table, one line per column; the empty cells are filled with '`0x03`', the code for an error. The lower part of the table will be empty because some columns are not used, for example, the columns used to control LEDs; also, I've never encountered any keyboards with more than 18 columns. The lower part (containing at most 2 keys) is hardcoded to save flash space; the array is made shorter, the keys out of the array are hardcoded in '`scan_to_code()`', and unused columns are enumerated in the '`unused_columns`' array (its size is defined with the '`UNUSED_COLUMNS`' macro); see `firmware/keytab-gen.h` for a reference implementation.

In many cases you'll want to resold the wires to make the used part of the table compact; also, some keyboards have identical matrices, but you'll have to rearrange the rows and the columns to match the same scancode table.

Now you can configure firmware for the working program, that is, without the '`--enable-raw-scancodes`' option, build and flash it.

Use `test-usb` to check that the keyboard outputs correct key codes.

[1] Note that codes for some additional keys like Sleep are absent; they would require a separate USB report descriptor, and they are currently not supported. You can write down their scancodes and replace '`0x03`' with '`0x00`' in those positions of the resulting table.

3.5 Enclosing in Screen

All elements of the device except the connectors are enclosed in a screen made of thin foiled FR–4; an additional internal screen is wrapped around the noise generator (Section 2.2 [Noise Generator], page 3) to protect the source of the noise from fast switching signals. These screens don't touch each other. The screens are mounted to the '`GND`' conductors of the board.

First, cut thin bands for the walls, tin them, apply to the board, bend and cut apertures for the conductors. Solder them to the board, cut the cover and tin its edges. Then solder the cover to the walls. Don't leave slits except for the conductors on the board and to separate the internal screen.

After soldering each part of the screen make sure that you haven't added any shorts: the power pins should show non-zero resistance, the extra LED should blink with a frequency of about 1 Hz (continuous light means failure of the noise generator), the `test-usb` utility should show exactly one scancode when pressing every key[2].

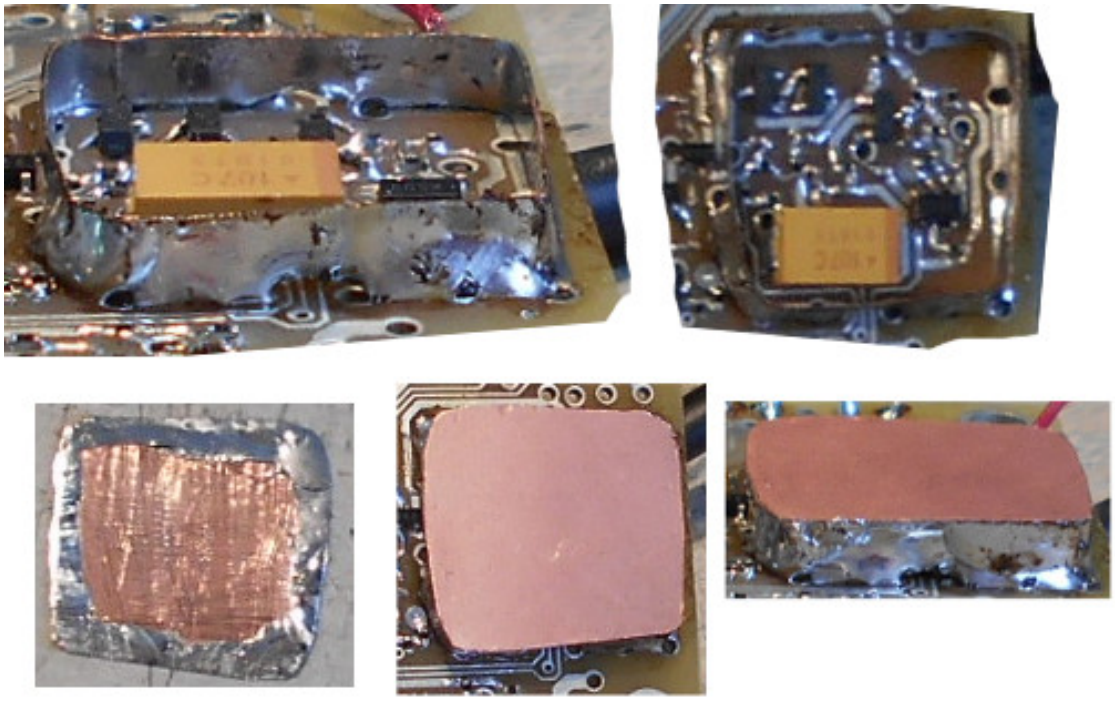

Figure 3.6: Screen of noise generator.

[2] For the modifier keys, that is, '`Shift`', '`Ctrl`', '`Alt`', '`GUI`', it should be a single bit in the first byte of the report.

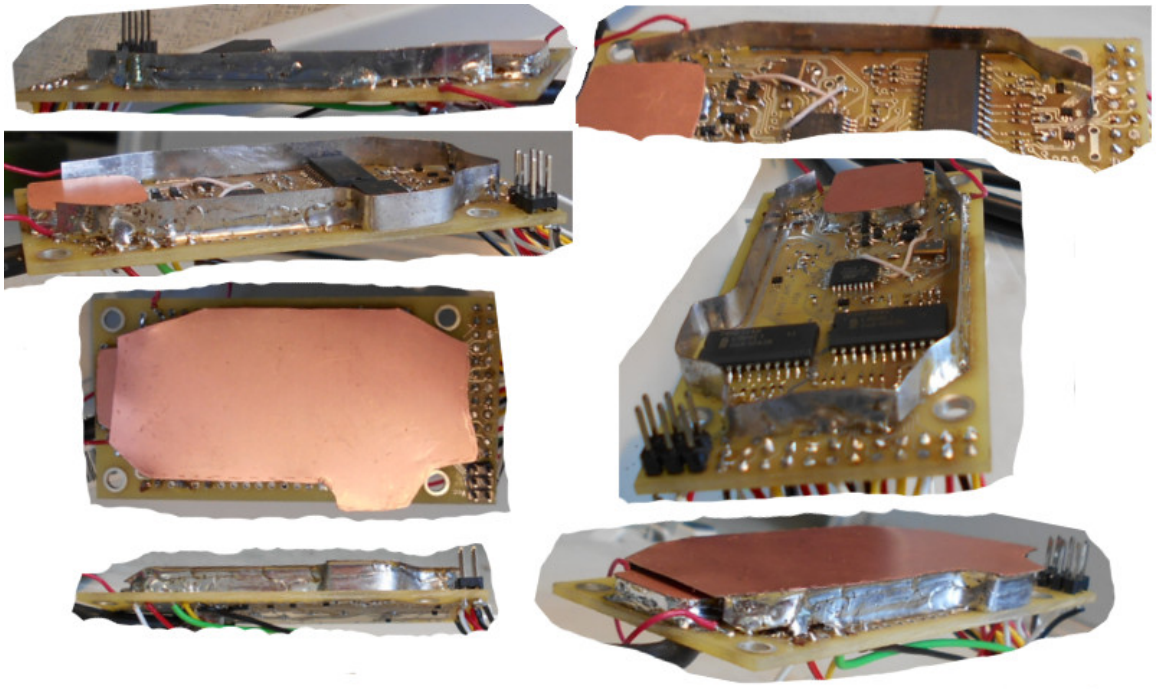

Figure 3.7: Top screen.

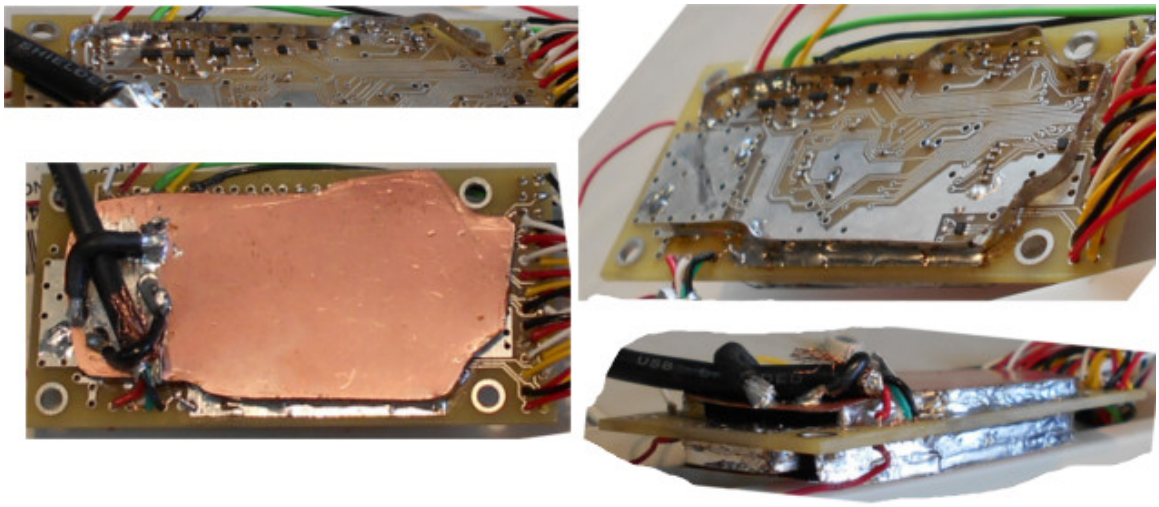

Figure 3.8: Bottom screen.

Fix USB cable on the screen with a piece of wire or two.

If needed, trim the wires. Don't make them too short: they should be long enough to let you disassemble the keyboard without unsoldering most of them.

3.6 Mounting on Keyboard

Now it's time to fix the whole construction on the case of the keyboard. Attach the battery holders to the keyboard, feed the controller from the batteries instead of the USB power line, mount the controller on the keyboard. Take care to provide a reliable contact with the batteries, they shouldn't drop out of their holders.

Figure 3.9: Device fully assembled.

4 Software

The package comes with three programs: `tab`, `test-usb` and `clav`. The two former are technological utilities used to setup the connection of the controller to the matrix, they run on a PC. The latter is firmware to run in the controller.

Software is configured and built using the standard GNU procedure:

```
tar xzf miguel-x.tar.gz && mkdir build && cd build
../miguel-x/configure && make
```

The '`install`' target, while supported, is not very useful: the utilities are launched from the build directory, and `clav.hex` is used with `avrdude`.

The `tab` utility has few dependencies. In order to build `test-usb`, you'll need libusb-1.0 installed. Building `clav` requires avr-libc and its dependencies.

Firmware is configured with a separate script invoked from the main `confiure` script; however, you can run `firmware/configure` directly.

These are the most important options of firmware `configure` script:

`--enable-raw-scancodes`
: The controller reports undecoded scancodes, the column number in the higher 5 bits, and the row number in the lower 3 bits of each byte. This is necessary when writing the keytable for the keyboard or adjusting the wires to an existing keytable.

`--with-matrix`
: This option selects the keytable to use.

`--disable-jamming`
: The program suppresses phantom keys by default: when it sees more than two non-modifier keys pressed at once, it skips the report. This algorithm takes little program memory and seems to practically solve the issue, but, on the one hand, it sometimes suppresses certainly valid keypresses, on the other hand, it still allows some phantom key combinations. This option switches it off.

`--with-max-idle-rate`
: The standard USB keyboard protocol allows the PC set the period of reports dropped when the state of keys doesn't change, and drivers typically set it to infinity. If the attacker can see the bursts of USB packets, the timings of keypresses may reveal some information about the pressed keys. This option limits the maximum idle period.

`--with-repeats`
: Some reports don't come from the controller to the PC, I have no idea why. To workaround this, I added the possibility to send multiple reports on the same event, and it helped in many cases (such behavior may confuse very old drivers, though). This option sets the number of repeated reports. It is not going to be needed when the maximum idle period is set to some small value.

5 Usage Notes

Even very old software supports USB keyboards, but in order to use it with the bootloader you may need to enable USB keyboards in BIOS (the kernel doesn't need it and detects the keyboard independently).

The keyboard should behave very similarly to the standard one except it has an additional LED. It indicates the battery level and signals about failures of random number generator.

When the keyboard is switched on, its microcontroller switches the extra LED on and runs the start-up random number generator test. It longs a second or two. If the hardware passes the test, the LED starts blink twice a second or so. The longer it is switched on, the higher the voltage of the battery.

The microcontroller continues running random number test, and if it encounters a failure, it switches the LED on and runs the start-up test again. In other words, long spans of the extra LED light indicate random number generator failure (this may mean that it's time to recharge the battery). Note that it still scans the keyboard and reports the keypresses.

Another possibility is cycling the USB power line: when the controller encounters low VUSB level, it switches off until VUSB is high again. In this mode, no LEDs are active, no keys are scanned, and the microcontroller puts itself in a sleeping mode.

6 Benchmarks

I've run two tests to estimate the relative emissions from the original and the modified keyboard. Both were done with a general oscilloscope, with no special radio equipment involved.

6.1 Wire Loop

This is the most simple test. A loop of wire is laid on the keyboard and connected to the oscilloscope.

Figure 6.1: Setup.

With the original keyboard, oscilloscope shows peaks up to 200 mV with a period of the oscillations of about 30 ns. When the controller is replaced, the oscilloscope doesn't detect such peaks. It means that the gain is at least 20 dB. The next test suggests that this estimation is very conservative.

Figure 6.2: Signal detected with wire loop.

6.2 Matrix Current

In the next test, I measured the current running through the matrix. In the original keyboard, I added a 10 Ohm resistor in series with a column, attached two probes, and used the oscilloscope to derive the difference of the voltages. Then I repeated the test with the modified keyboard, but this time I used a 10 kOhm resistor. In both cases, I pressed a key on the respective column.

The original keyboard shows pulses with a peak of about 400 mV and a width of about 30 ns. The voltage of 400 mV corresponds to a current of about 40 mA.

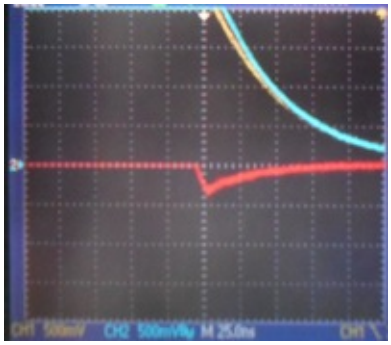

Figure 6.3: Original keyboard, 10 Ohm.

The signal from the modified keyboard is noisy, with a maximum of about 100 mV, which translates to currents of 10 uA. It's hard to tell if the oscillations are an artifact of the procedure, but at any rate their typical period is about 1 us.

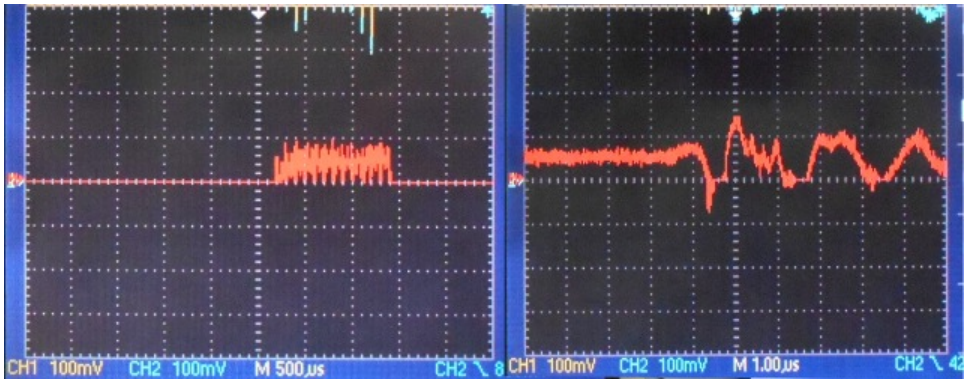

Figure 6.4: Modified keyboard, 10 kOhm.

The keyboard matrix may be thought of as a small magnetic dipole (the smallest time at hand, 30 ns, corresponds to 60 m wavelength, it's much more than the size of the keyboard). The power emitted by such dipoles is directly proportional to the squares of the amplitude and the frequency of the current. This means that the decrease in current is equivalent to 70 dB, and the lower frequency may result in additional 30 dB gain.

70 dB means that the attacker with the same equipment has to be 3000 times closer[1]. My threat model assumes that they have no access to the room where the keyboard works (if they had, they possibly could install a kind of keylogger), so in order to intercept the emissions from the modified keyboard, they must be able to intercept the emissions from the original keyboard at a range of a few kilometers. I conclude that the matrix emission hardly can be an issue for the modified keyboard, it must be easier to exploit some other side channel.

[1] Greater distances generally allow more advanced equipment, for example, large directed antennas, but I don't take this into account.

7 Licensing Terms

All texts originally written for this project are distributed under the GNU GPL version 3 or (at your option) any later version. However, the external USB library for AVR is distributed under the GPLv3-only, so the firmware part as a whole is under the GPLv3-only.

Appendix A GNU General Public License

Version 3, 29 June 2007

Preamble

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

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

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

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

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

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

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

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

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

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

TERMS AND CONDITIONS

0. Definitions.

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

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

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

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

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

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

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

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

1. Source Code.

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

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

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

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

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

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

2. Basic Permissions.

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

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

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

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

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

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

4. Conveying Verbatim Copies.

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

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

5. Conveying Modified Source Versions.

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

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

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

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

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

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

6. Conveying Non-Source Forms.

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

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

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

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

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

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

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

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

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source.

The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

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

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

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

7. Additional Terms.

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

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

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

   a. Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
   b. Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
   c. Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

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

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

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

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

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

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

8. Termination.

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

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

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

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

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance.

However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

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

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

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

11. Patents.

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

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

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

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

    If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so

available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

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

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

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

12. No Surrender of Others' Freedom.

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

13. Use with the GNU Affero General Public License.

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

14. Revised Versions of this License.

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

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

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

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

15. Disclaimer of Warranty.

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

16. Limitation of Liability.

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

17. Interpretation of Sections 15 and 16.

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

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

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

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

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

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

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

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

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

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

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

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

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

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