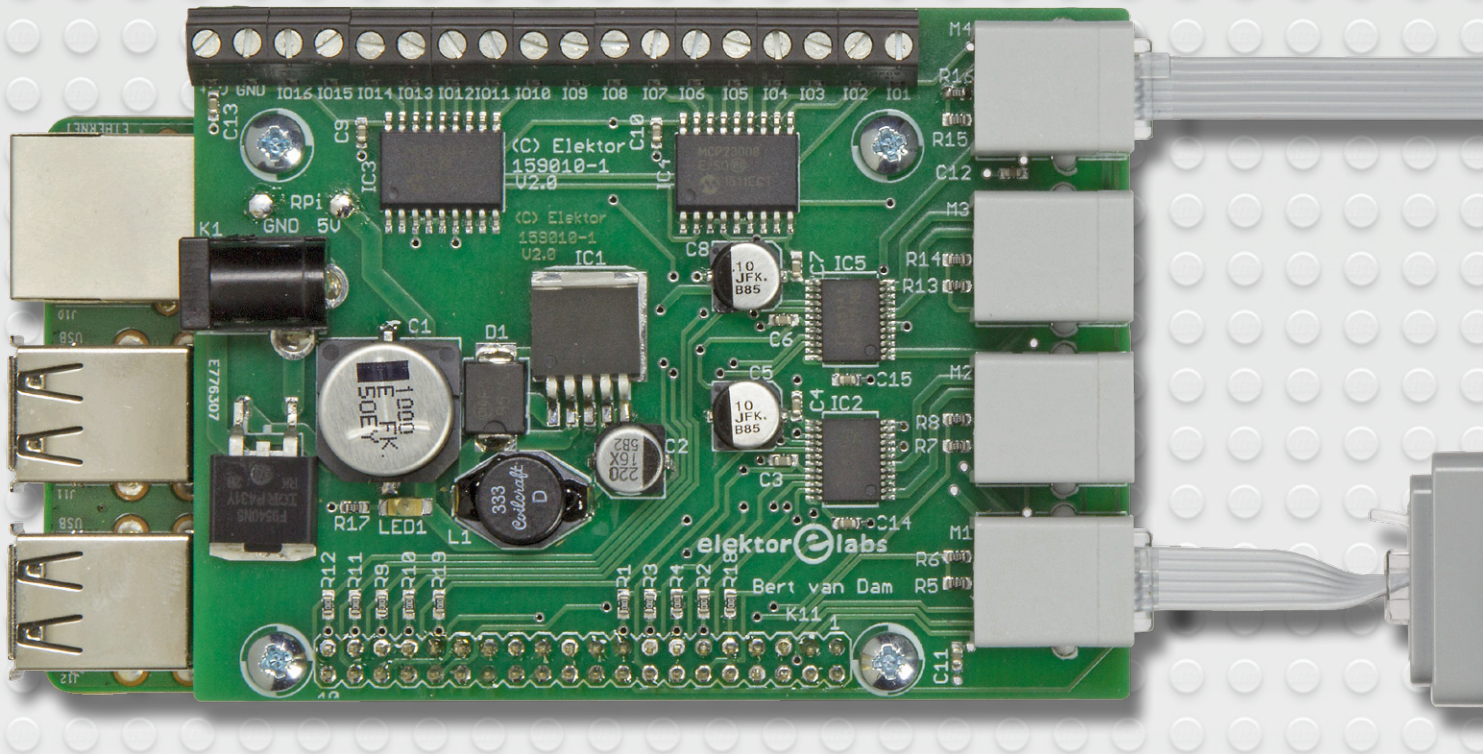
LEGO® Control Board for the Raspberry Pi



This Raspberry Pi HAT puts at your disposal 4 motor-control outputs for LEGO EV3 motors and 16 buffered I/O connections that can be used in combination with a powerful Raspberry Pi. Programming is possible in languages such as C and Python. As an application example we show how this board and a few LEGO parts can be used to build a 'useless box'.

The board described here combines the flexibility of LEGO with regards to mechanical construction and the flexibility of the Raspberry Pi with regards to software, Internet communications and a wide range of potential sensors.

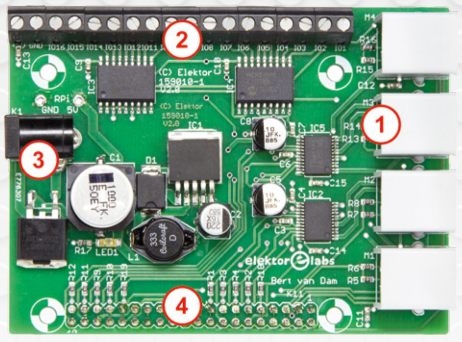


Figure 1. The connections to the LEGO control board. 1) 4 LEGO motor connections; 2) 16 I/O connections with screw terminals; 3) power supply socket, 9 V; 4) feedthrough of the Raspberry Pi I/O pins.

When you attach this board to a Raspberry Pi 2 or 3, you will have at your disposal 4 motor-control outputs for EV3 motors (or compatible types), which you can use to animate your LEGO models. There are also 16 buffered digital inputs and outputs present, to which you can connect sensors, LEDs, etc. You cannot, however, connect LEGO sensors (in that case you would be better served using a standard EV3 controller), but you can connect all sorts of other sensors such as infrared, ultrasonic, tilt and switches. You are limited only by your imagination. The board also contains a power supply for the Raspberry Pi, you therefore only need to connect a 9-V (line) power supply to this board (recommended rating 5 A; positive to center pin). From this the motors, the board itself and the Raspberry Pi are powered. Mobile applications can use batteries, because the board uses an efficient buck converter.

Since the Raspberry Pi header is continued through the LEGO control board, you can also connect SPI and I^{2}C devices and sensors. You can therefore combine all the projects and sensors from the book "Raspberry Pi – Explore the RPi in 45 Electronic Projects" with your LEGO models!
Through the Raspberry Pi itself you can communicate via the Internet, with your

Lego is a registered trademark of the LEGO Group

With 4 motor-control outputs and 16 digital I/Ochannels

By **Bert van Dam** (Netherlands)

choice of a wired or Wi-Fi connection. If you connect a USB/4G dongle to your Raspberry Pi then you can even communicate with your model when it is away from your home.
With this board you can therefore make models where you combine LEGO motors and various types of sensors with the power of the Raspberry Pi!

Table 1. Strong and weak points of the LEGO EV3 and Raspberry Pi.

LEGO — strong • The mechanical construction, which allows you to do anything you like. • The motors, although for serious applications you would need servos and hydraulics.	**LEGO — weak** • The number of connections for sensors. A reasonably serious robot would need 4 and that is only the start. • Video processing and all the other things for which LEGO have not come up with a sensor.
Raspberry Pi — strong • Openness of the system; in combination was the easy-to-learn language Python much is possible. For the enthusiast C is an even more powerful option. • Many connectors and via I/O-expanders offers the possibility of ‘unlimited’ expansion.	**Raspberry Pi — weak** • Connection pins are mechanically weak and not electrically protected. • Current consumption is on the high side.

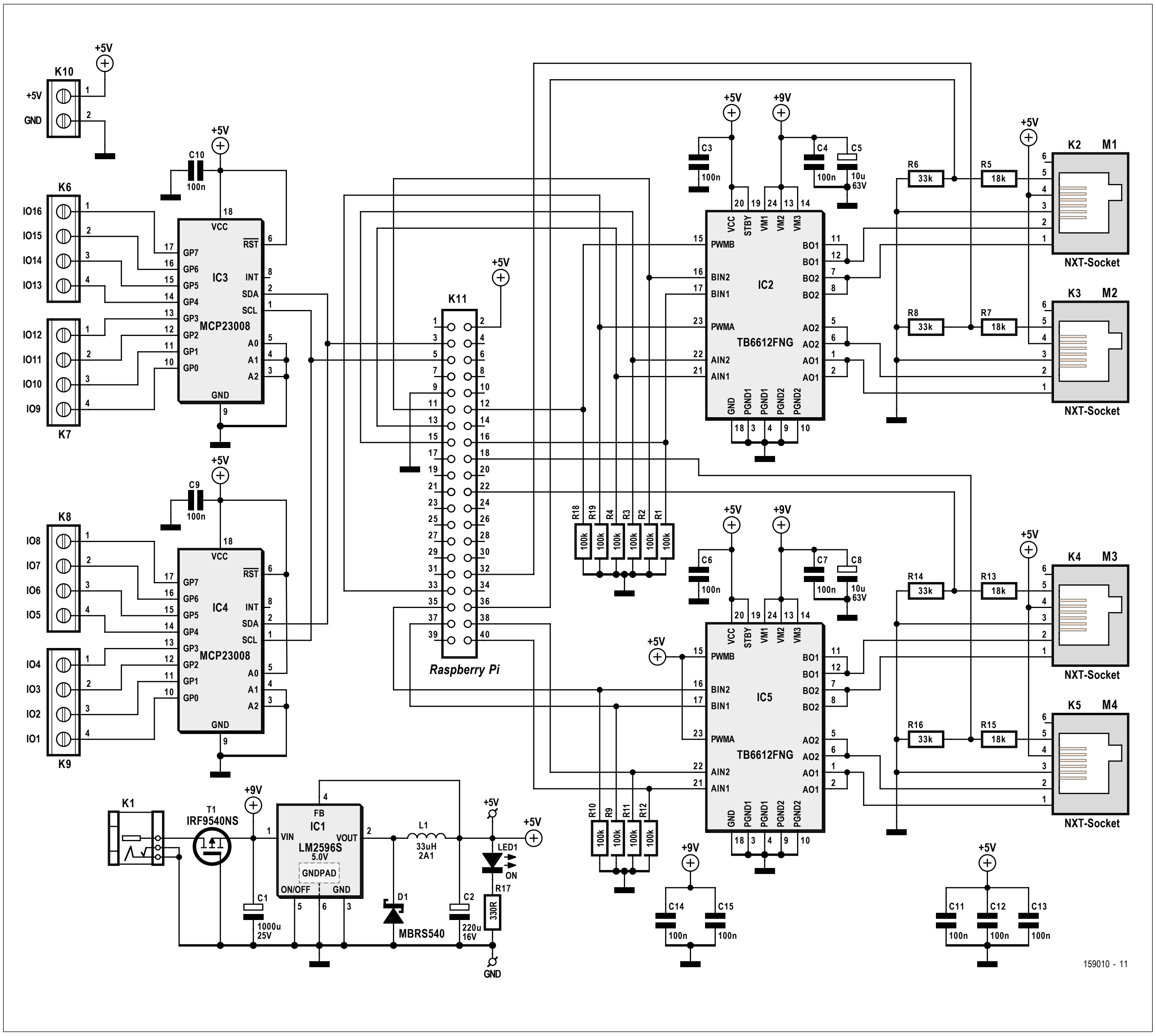


Figure 2. The circuit of the control board comprises 2 motor driver ICs, 2 I/O expanders and a step-down voltage regulator.

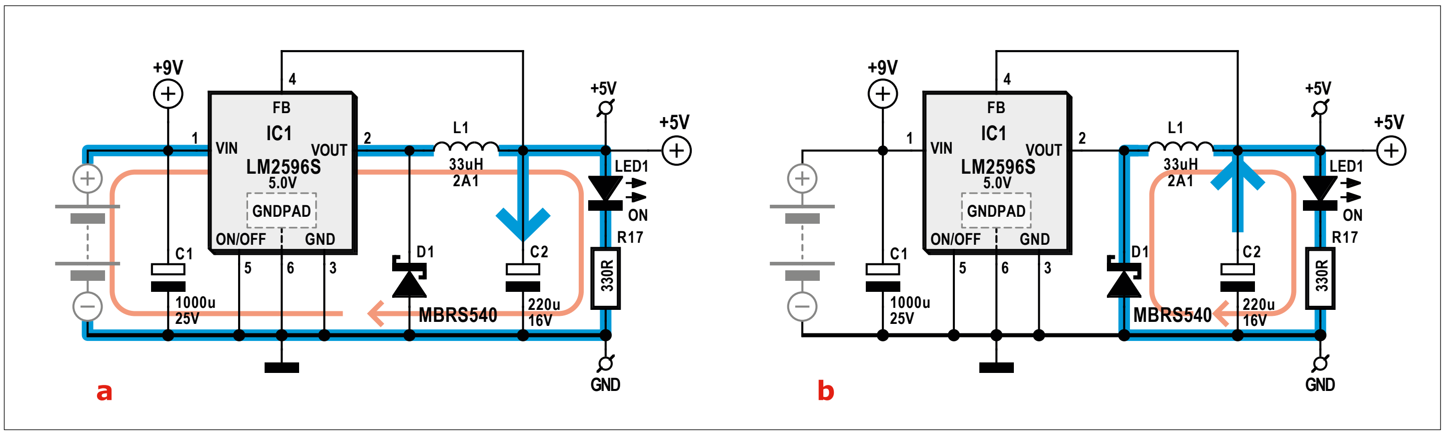


Figure 3. Buck converter: a) cranking up the flywheel b) coasting of the flywheel.

Design considerations

The LEGO control board is certainly not a copy of the EV3 controller, but it does combine the strong points of LEGO with those of the Raspberry Pi. **Table 1** lists the strong and weak points of each.
The conclusion is that this board needed to have the following characteristics (and it therefore was given them):

- 4 LEGO EV3 motor control connections (for LEGO 45502, 45503 or compatible), with all 4 of provided with tacho feedback signals and 2 provided with PWM outputs;
- 16 buffered I/O pins (expandable thanks to the I^{2}C connection);
- energy efficient power supply for the Raspberry Pi (using buck-converter);
- free Python 2 software library;
- feed-through of the Raspberry Pi header (you therefore can add further connections yourself, such as protected I/O or analog I/O).

Design result

The complete schematic for the LEGO control board is shown in **Figure 1**. The most relevant parts in this circuit are the two motor-driver ICs type TB6612FNG from Toshiba (which each can drive two motors with a maximum output current of 1.2 A) and two 8-bit I/O-expanders type MCP23008 from Microchip with a high-speed I^{2}C interface and SPI interface. We now take a look at a few interesting details of the circuit.

Power supply

In order to enable everything to be powered a single source, the Raspberry Pi is powered from the control board. The board itself is powered from a 9-V DC power source. This power source can either be batteries or a substantial line power supply (recommend rating 5 A).
To go from 9 to 5 V (the voltage required by the Raspberry Pi) an energy-efficient method needs to be used to prevent the batteries from draining too quickly. The choice therefore fell on a buck converter based on the step-down regulator LM2596S-5.0V, a so-called Simple Switcher Power Converter from Texas Instruments. This operates at a frequency of 150 kHz and can supply 3 A.
The buck converter functions as follows (see **Figure 3**). The loop formed by the diode (D1), the inductor (L1) and the capacitor (C2) act like a kind of electronic flywheel. The LM2596 begins by ‘turning up’ the voltage to ‘full’. This gets the flywheel started. This does not happen immediately, because the capacitor first needs to be charged via the inductor. The inductor itself builds up a magnetic field. The diode does not conduct at this time since the current returns via the LM2596 itself.

When the voltage at the output threatens to become too high, the LM2596 switches its output off. The magnetic field in the inductor will now collapse, which causes the voltage across it to reverse and the diode will now conduct. This creates a current that circulates via the load and the diode (the path via the LM2596 is no longer possible because this is switched ‘off’). The capacitor also discharges. When the voltage at the output threatens to become too low, the LM2596 switches on again.

The LM2596 therefore only switches on and off. The effect of this is that the efficiency is very high and the LM2596 therefore barely heats up. Because of this continuous on and off switching, the value of C1 is also important and for both C1 and C2 low-ESR types need to be used.

I/O

The MCP23008 I/O extenders IC3 and IC4 are controlled via I^{2}C and ensure that the I/O connections of the Raspberry Pi do not need to be used directly. This design has several advantages:

- The I/O connections operate at 5 V
- The I/O pins have a greater current capability than the Raspberry Pi pins
- Should something nevertheless go wrong, the damage will generally be limited to the I/O extender chip

The maximum current that the MCP23008 can supply is 25 mA, with a maximum per IC of 125 mA. Note that one IC can dissipate a maximum power of no more than 700 mW. This applies to each MCP23008, so if you need a lot of current it would be best to spread the loads across both MCP23008s and not connect everything to a single IC.
Because the Raspberry Pi headers are continued through to the top of the control board you can add additional I^{2}C components yourself, provided that they have different addresses. The LEGO control board uses addresses 0x20 and 0x21. You can, for example, add further I/O-extenders, D/A converters for generating analog signals, or A/D converters to sense analog signals.

Motors

The two TB6621FNG ICs (IC2 and IC5) each contain two H-bridges to allow the control of four motors in total. The current consumption of the LEGO motors is approximately:

	EV3 small	EV3 large
free running	85 mA	80 mA
slowly forward	500 mA	1000 mA
stalled	780 mA	1800 mA

An EV3 motor can be stalled for only a short amount of time, because the protection built into the motor will switch it off. It is nevertheless always a good idea to prevent stalling the motor, because should the protection not work for some reason the motor could burn out.

The TB6621FNG can typically handle 1.2 A (per channel, so per motor) with a peak of 3 A; this is therefore more than sufficient.

If you intend to make a software library for the motor controllers, then take into account that the PWM pins have to be high in order for the TB6621FNG to operate. When you apply a PWM signal then it is always correct, independent of the direction of rotation.

Circuit board

In **Figure 4** you can see the circuit board that was designed for this LEGO control board. The dimensions and connector positions are obviously selected such that the board will fit trouble-free on a Raspberry Pi model 2 or 3 (using a 40-way I/O connector). Since most of the components are in SMD packages, you can also purchase a completely built version of the board from Elektor. Of course, it is also possible to do everything yourself, the circuit board layout is available from [1]. Note when assembling the board that the 40-way connector K11 is fitted on the bottom side of the circuit board and is soldered from the top side. The four NXT sockets for the motors are quire rare, you will have to search on the Internet for a suitable supplier.

Component List

Resistors
R1,R2,R3,R4,R9,R10,R11,R12,R18,R19 = 100kΩ 1%, 0.1W, case 0603
R5,R7,R13,R15 = 18kΩ 1%, 0.1W, case 0603
R6,R8,R14,R16 = 33kΩ 1%, 0.1W, case 0603
R17 = 330Ω, 0.1W, case 0603

Capacitors
C1 = 1000µF, 25V, aluminum, d 12.5mm, h 13.5mm, ELPP-CP-125-135
C2 = 220µF, 16V, aluminum, d 6mm, h 7.7mm, ELPP-CP-063-066
C3,C4,C6,C7,C9,C10,C11,C12,C13,C14,C15 = 100nF 10%, 16V, X7R, case 0603
C5,C8 = 10µF, 63V, case D

Inductors
L1 = 33µH, 2.1A (e.g. Coilcraft DO3316)

Semiconductors
D1 = MBRS540, 40V 5A, ELPP-DO-214AB
LED1 = LED, red, case 1206
T1 = IRF9540NSPBF, P-channel MOSFET, D2-PAK
IC1 = LM2596S-5.0, DC/DC buck converter, TO-263-5
IC2,IC5 = TB6612, motor driver, 24-SSOP (Digikey # TB6612FNGC8ELCT-ND)
IC3,IC4 = MCP23008-E/SO, I/O expander, SOIC-18

Miscellaneous
K1 = DC adapter connector, 12V 3A, 1.95mm center pin
K2,K3,K4,K5 = NXT socket
K6,K7,K8,K9 = 4-way PCB screw terminal block, 3.81mm pitch
K10 = 2-way PCB screw terminal block, 3.81mm pitch
K11 = 40-pin (2x20) GPIO stacking header for RPi, extra tall
PCB # 159010-1

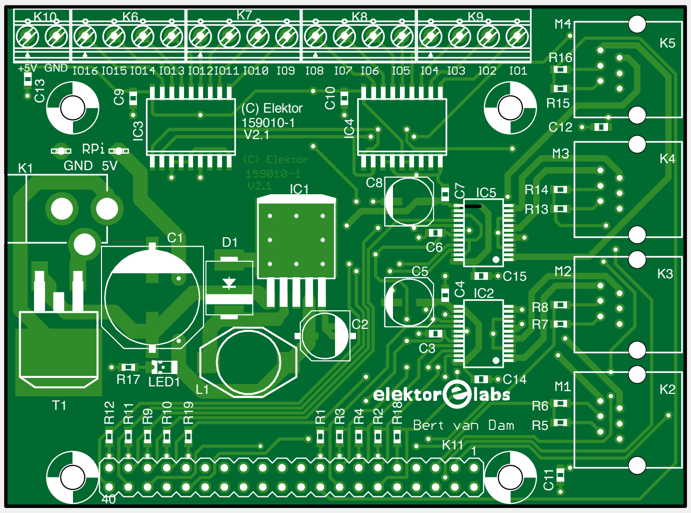


Figure 4. The layout of the circuit board. The 40-way connector K11 is mounted on the bottom of the circuit board.

workings; making the box we leave up to you. You can make this easily using cardboard or wood. With the download for this article you will find a short video of this project. Both the pictures accompanying this article as well as the video show only the mechanical part, without the box. This way you can get a clear view of its operation.

What do you need?

- The SD card from the book 'Raspberry Pi – Explore the RPi in 45 Electronic Projects';
- The free download that goes with this article (contains the library and the source code for the demo project);
- A LEGO motor (type 45502 'large' motor) with connecting cable and a few LEGO parts (you can build the demo project using the LEGO Mindstorms EV3 box).
- A Raspberry Pi model 2 or 3;
- The LEGO control board;
- Two resistors: 1 kΩ and 10 kΩ, 0.25W;
- An (easily operated) small in-line switch (see **Figure 4**).

The hardware

We use a 'large' LEGO motor type 45502 which can be found in, for example, the LEGO Mindstorms EV3 box and the parts that are shown in **Figure 5**. In the download you will find a short video where you can see how you assemble these parts into the motorized finger, which is shown on the right in the figure. You now attach the in-line switch on top of the assembly in such a way that the black beam can push the switch to its off position.

Demo project: The Useless Box

The concept of this 'Useless Box' is a box with a switch on top. When you operate the switch, a 'finger' appears from the box which turns the switch off again. For this project we only build the inner

This switch is connected with two resistors (see **Figure 6**) to I/O pin 16. This could also be accomplished with only one resis-

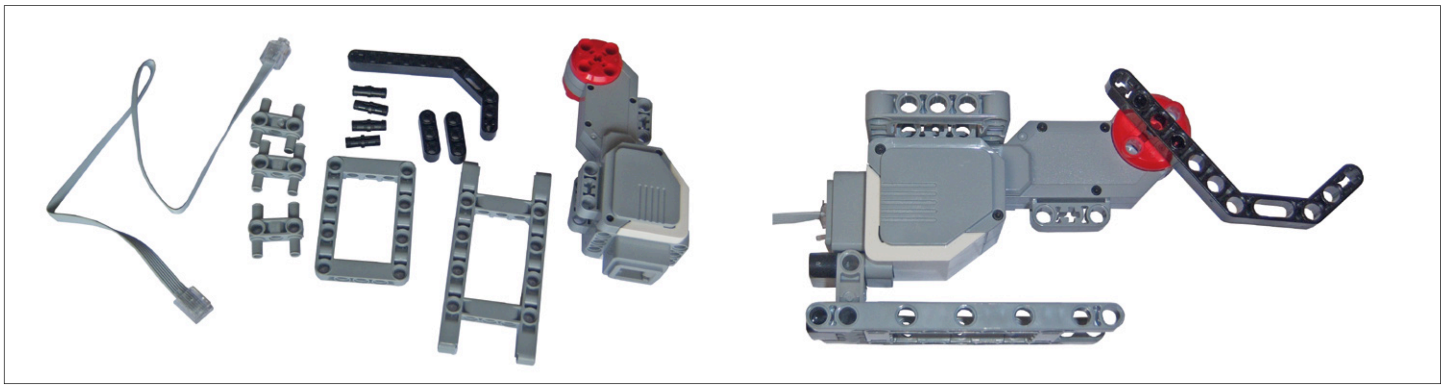


Figure 5. Parts and the assembled model of the motorized finger.

tor (the 10 kΩ one), but the additional resistor serves as protection. If you now inadvertently make the input into an output and set its level low then nothing bad will happen. Without the additional 1-kΩ resistor the I/O port could be damaged.

The software

The software is surprisingly simple. (Note: the library expects Python-2 programs!) You can find more information about the library and the instructions in the manual, which is a free download accompanying this project [1]. The very first thing we do is load the rpirobot library and execute it with the instruction 'execfile'. The motor outputs of the LEGO control board are now automatically configured correctly and the PWM threads are running. We then make pin 16 of the I/O on the control board an input, because this is where the switch is connected to.

```
execfile('rpirobot.lib')

# Switch on io 16, input
iodir(16,1)

print "Useless Box Program running"
```

We use a try/except construct (see **Listing 1a**), so that the program waits until the switch is turned on by the user or the user has quit the program using Ctrl-C. Once this happens, the program waits half a second to give the user timer to remove their finger and then activates the LEGO finger with a PWM value of 60 (60% of the maximum speed). We wait in a loop because of the threads that are running. Since nothing happens in the loop this is not actually helpful, so we introduce a short sleep instruction. This instruction takes 0.000001 seconds (1 µs), and will not noticeably slow down the program. As soon as the switch is off again, the program will turn on the brake in the motor. The next step (**Listing 1b**) is to return the LEGO finger at a slower speed (PWM 30, 30% of the maximum speed) to its rest position. For this case there is no switch at the end of the movement, so the program now uses the tacho. The program counts the tacho changes and once there are 100 (28% of a circle) the motor stops with the float instruction. The LEGO finger is now about one centimeter above the surface.

In the except-construct (**Listing 1c**) any errors or a Ctrl-C instruction from the user are caught. The program gives a brief abort instruction to the Raspberry Pi library and then terminates itself as well. Without this construct it is obviously still possible to exit the program using Ctrl-C, but then the library continues running and that is clearly not the intention.

The steps

1. Set the LEGO finger in the rest position, that means as far as possible from the switch (against the bottom surface for example).
2. Start the Raspberry Pi and wait until the booting process has completed.
3. Copy the test folder from the download to the Raspberry Pi (using WinSCP, for example).
4. Return to the test folder with the instruction 'cd test'.
5. Now issue the instruction 'python uselessbox.py'.
6. Wait until the text 'Useless Box Program running' appears on the Raspberry Pi and turn the switch on (and get your fingers out the way!). The LEGO finger will now return the switch to its off position.
7. Repeat step 6 as many times as you like and then stop the program using Ctrl-C.

This is only a simple application example. The LEGO control board can of course be used to realize much more complicated applications with its four motor control connections and 16 I/O pins.

(150597)

Web Links

[1] www.elektor.com/150597

Listing 1a.

```
try:
  while 1:
    # wait for the user
    # to push switch on
    if ioread(16):
      # push switch off
      time.sleep(0.5)
      reverse(1)
      pwm(1,60)
      while ioread(16):
        time.sleep(0.000001)
      brake(1)
```

Listing 1b.

```
    # retract lever
    forward(1)
    pwm(1,30)
    counter1=0
    lastpin1=tacho(1)
    while counter1<100:
      time.sleep(0.000001)
      if tacho(1)!=lastpin1:
        counter1+=1
        lastpin1=tacho(1)
    # float motor
    float(1)
```

Listing 1c.

```
except:
  # user aborts program with
Ctrl-C
  print "Program aborting"
  abort()
  print "Done"
```

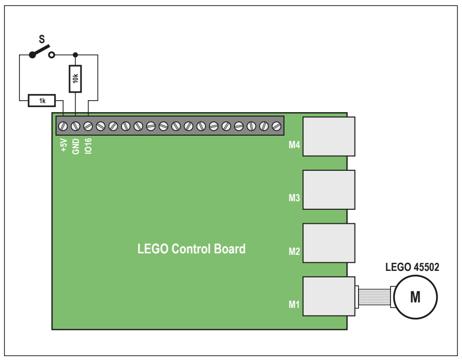


Figure 6. Connection diagram for the switch with the two resistors.

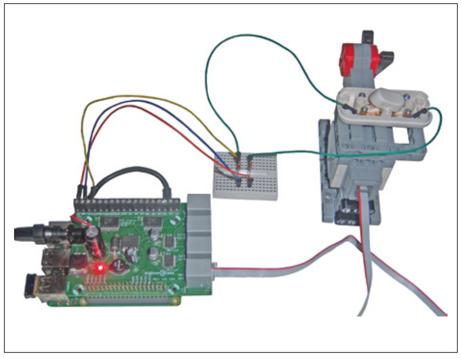


Figure 7. The complete model.