RPi GPIO Interface Circuits

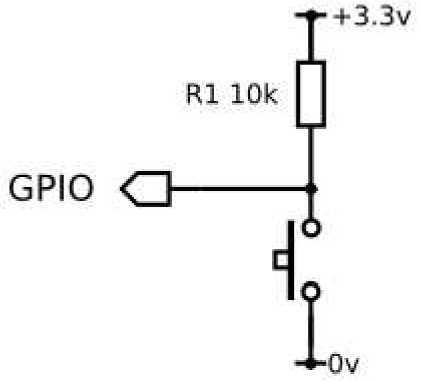
From eLinux.org

This page shows a number of common circuits used for interfacing the Raspberry Pi's GPIO pins to various other electronic circuits.

Contents

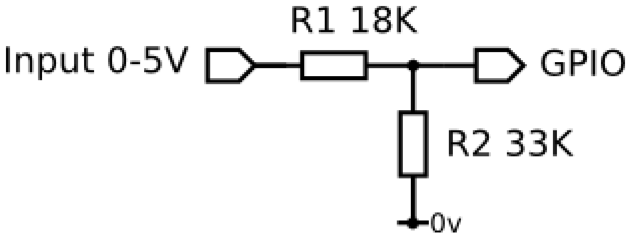
Input circuits

Buttons and switches



A simple pushbutton or switch is very easy to connect to a GPIO, as shown above. When the button is pressed (or the switch is closed) the GPIO input will be '0'. When it is released (or the switch is opened) the GPIO will be '1'. R1 is a *pull-up* resistor, which "pulls" the GPIO input to 3.3V when the switch is open.

Voltage divider



This circuit may be used to connect a digital signal from a 5V device to a GPIO pin on the Pi, which requires a 3.3V input level. The **Ov** point in the schematic should be connected to one of the Ground lines on the Pi's GPIO connector, and also to the 0v (or Ground) signal on the input device.

The important thing here is the *ratio* of R2 to (R1+R2); the ratio should be such that 5V is divided down to 3.3V. In this example the ratio is 0.647 (=33/(18+33)), so 5V is divided to 3.24V (=0.647 * 5). The values shown here should be suitable for most uses but please check the drawbacks.

Drawbacks

This circuit adds a resistance to the output equal to (R1+R2)/R1*R2 so you have less current to drive a difficult load. In most cases that's not a problem but you may need something better if you want to drive a relatively high load or to get fast switching of a capacitive load. An other detail is that when the output is high you get a very light current leaking to the ground (5V/(R1+R2)~=0.1mA).

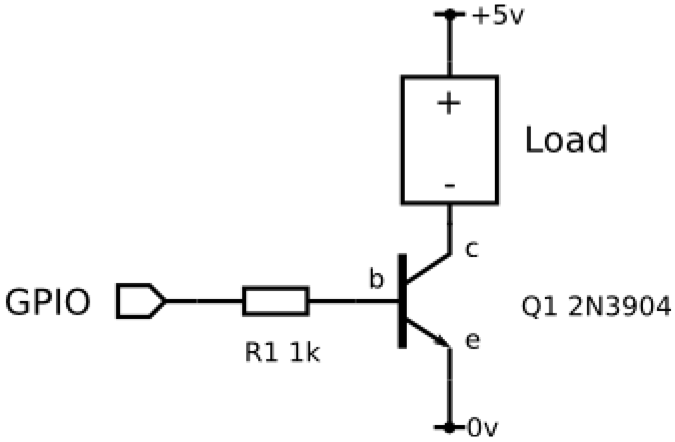
(For more information, see the Wikipedia article (http://en.wikipedia.org/wiki/Voltage_divider) on voltage dividers).

Output circuits

The GPIO pins are connected *directly* to the BCM2835 chip at the heart of the Raspberry Pi. These provide only a 3.3V output level, and are not capable of supplying much power. More importantly, if they are damaged through misuse the Pi itself will need to be replaced.

So, if you are connecting anything more than a small LED to the GPIO output, you should use an additional circuit to boost the voltage and/or current.

Using an NPN transistor

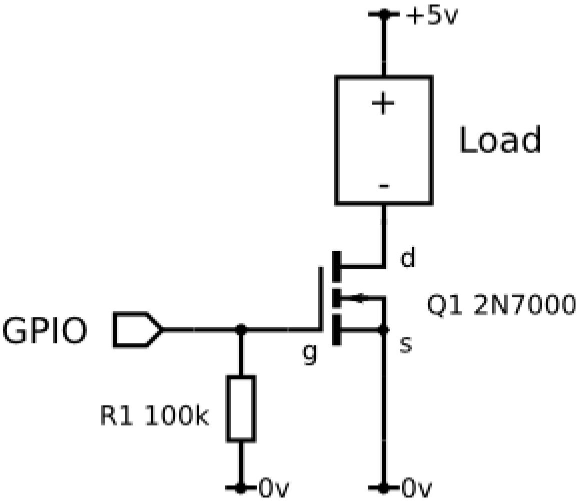


This shows a cheap NPN transistor being used to switch a load on and off. When the GPIO is 'high' (logic 1) the load will be turned on; when it is 'low' the load will be off. The circuit shows a 5V power rail, but with the 2N3904 transistor shown, a supply of up to 40V may be used (this is the V_{CEO} value given in the transistor's datasheet). It should be suitable for a load current of up to 100mA.

For bigger loads, you will need a bigger transistor to switch higher voltages (the V_{CEO} value) or currents (the I_C value), but note that you will need a transistor with a higher *gain* (the h_{FE} value) as well. The gain is the ratio of the current in R1 (in this case, about 2.5mA), to the current in the load: a 1A load will need a gain of 400. So-called *Darlington* transistors, such as the TIP112 (1A, 100V) provide high gain at high currents.

Note! Transistors designed for high power operation generally need a heat sink. The current ratings quoted in the data sheet assume ideal cooling is provided: you will not get anywhere close to these without adequate cooling.

Using a FET



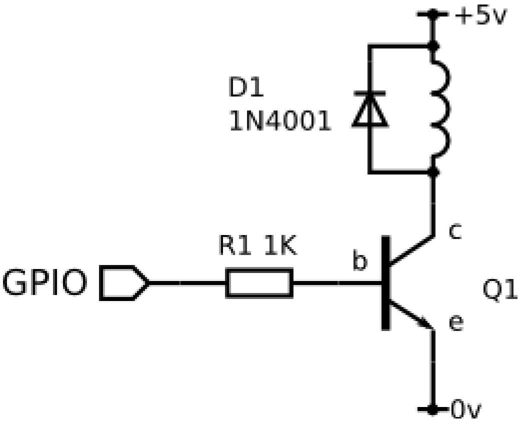
A *field-effect transistor* is another alternative to the ('bipolar') transistor shown above. Again, when the GPIO is 'high' (or 1), the load will be switched on, and 'low' or 0 switches the load off. Resistor R1 is provided for safety; it ensures that the load is switched off if the GPIO is set to be an input.

As with the bipolar transistor circuit, supply voltages higher than 5V may be used for the load. The 2N7000 device has a maximum (V_{DS}) voltage of 60V; it is suitable for load currents of about 100mA.

For bigger loads, you can choose a bigger FET: the current rating (I_D) and voltage rating (V_{DS}) are given in the data sheet. Unlike bipolar transistors, FETs do not have a 'gain' to worry about. Instead you should look for a 'threshold' voltage (the input voltage at which the FET begins to turn on) of 3V or less, so that it is properly 'on' when driven from the GPIO's 3.3 volts. Some datasheets don't give a threshold voltage - instead they give a value R_{DS}(on) measured at 2.5 or 2.7 volts. If the FET isn't suitable for 3.3V logic input, this value won't be given. Fairchild's FQP30N06L has a low threshold voltage (2.5V max) and can switch 32 amps at 60 volts.

As with bipolar transistors, a power FET will need a good heatsink if you are using it at anywhere near its rated current.

Driving a relay



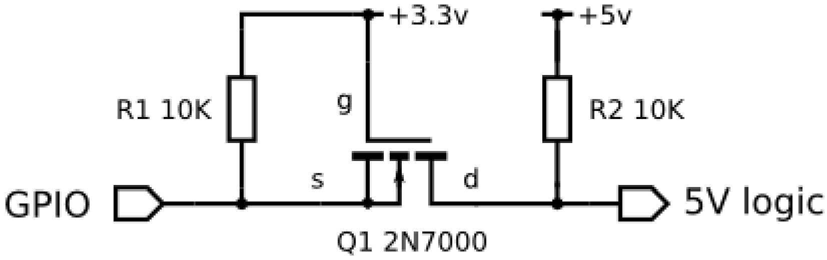
If you are connecting an *inductive* load (essentially this means anything with windings, such as a relay, a motor, or a solenoid) it is advisable to fit a diode D1 as shown above (note that the *cathode* is connected to the positive supply rail). When an inductive load is switched off, a voltage spike is produced which can exceed the maximum rating of transistor Q1; the diode 'clamps' this voltage to prevent damage.

Level Shifters

Many digital logic circuits (particularly older ones) use a 5 volt power supply. These cannot be connected to the GPIO signals on the Pi, which uses a 3.3V power supply. There are two cases: a) a Pi output driving a peripherals' input b) a peripherals' output driving a Pi input. In case a) the Pi GPIO will output +3.3V for a logic 'high', which is less than the 3.5V (= 0.7 * V_{DD}) required by most 5V CMOS logic [1]. In case b) a 5V logic output will send an excessively high voltage into a GPIO input of the Pi, which may damage either or both of them. This problem however can easily be solved with a resistor divider (see above).

The solution is a *level shifter* circuit. Level shifters are available in IC form or as pre-built modules from many suppliers (e.g. Adafruit, Sparkfun, Cool Components), but it's quite easy to make your own.

Classic MOSFET level shifter

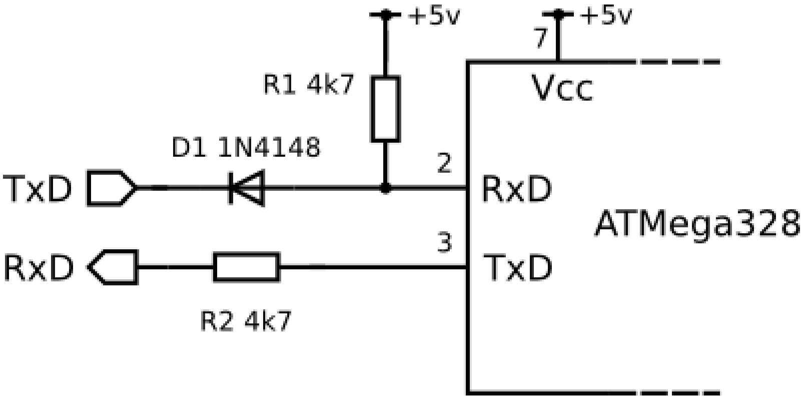


This uses a single MOSFET; when the GPIO is 'low', the MOSFET is turned on (because the gate-source voltage is 3.3V), so the 5V side is connected to the GPIO pin - it is therefore also low. When the GPIO is 'high' the MOSFET is off (the gate-source voltage is near zero), and R2 pulls the 5V side high.

The circuit also works in the *other* direction, and lets you safely drive a 3.3V GPIO input from a 5V source. Q1 has a (hidden) diode between the source and drain terminals - if the 5V logic side is driven low, this will pull the GPIO low through the diode. When the 5V logic side is high, everything is off and the GPIO will be pulled to 3.3V via R1.

For this reason, the above circuit is often used to convert levels on **bidirectional buses such as I^2C**. Note that for an I^2C bus, you will need two copies of the circuit, one for the SDA line, and one for SCL.

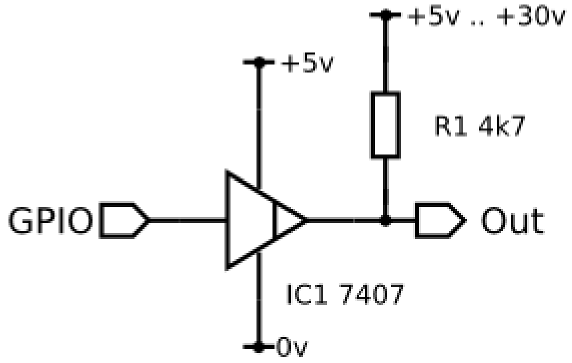
Simple diode circuit



This quick-and-dirty circuit has been used successfully to connect the serial port of a 5V ATMega328 processor (as used on many Arduinos) to the Pi's serial port. The 0 and 3.3V logic levels on the Pi's TxD pin are shifted (via D1 and R1) to approximately 0.7V and 4.0V. This is within specification for the ATMega328's RxD pin, so the logic levels are recognised correctly. Note that it is *only just* within specification; this means electrical noise (interference) can cause errors more easily than when using a 'proper' level shifter.

There is no level conversion between the ATMega328's TxD and the Pi's RxD. Instead, resistor R2 limits the current flowing from one device to the other to a safe value (under 0.5mA).

SN7407 IC



The SN7407 open-collector buffer IC is useful for a variety of level shifting and output-driver tasks. When the GPIO output is '0' (low), the 7407 output will be low. When the GPIO is '1', the 7407 output is 'not driven', and R1 will pull the circuit output high. The output-high voltage can be as high as 30V.

The 7407 output can sink up to 30mA when the output is low, so low-current loads (e.g. LEDs) can be connected directly in place of R1. The IC has six independent circuits in a 14-pin DIP package, and is quite cheap (roughly £0.50).

---

1. ↑ See e.g. http://www.allaboutcircuits.com/vol_4/chpt_3/10.html

Note: If you wish to produce new or modified schematics matching the ones above, you can use Inkscape (http://www.inkscape.org) using symbols from Circuit_Symbols.svg (http://www.elinux.org/File:Circuit_Symbols.svg).

Retrieved from "https://elinux.org/index.php?title=RPi_GPIO_Interface_Circuits&oldid=410011"