CONTROLLING OF THE MOBILE ROBOT USING A RASPBERRY PI MICROCONTROLLER AND PROGRAMMING IN PYTHON

Gîlcă Gheorghe, *"Constantin Brâncuși" University from Tîrgu-Jiu, ROMANIA*

ABSTRACT: Raspberry Pi is very popular for IoT projects because of its seamless ability of wireless communication over the internet. In this paper I will show you how to connect some motors to your Raspberry Pi. Doing so will allow your Raspberry Pi to interact in the real world, making it possible to build a robot, turn on a fan on a hot day or even drop a treat for your cat or dog while your away.

KEY WORDS: robot control, programming, direction, motor, drivers.

1. INTRODUCTION

The Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote the teaching of basic computer science in schools and in developing countries [1]-[3].

Raspberry Pi is a single board, credit-card size computer that can run Linux. Raspberry Pi hardware has low-level interfaces intended to connect directly with external devices such as A/D converters, sensors, motor drivers, etc. We can take advantage of these low-level interfaces to develop meaningful real-world applications.

In this work is the safely connect two motors to the Raspberry Pi with as few components as possible. Once we have the electronics put together on the breadboard, I will show you how to control them easily using Python to first make the motor spin, and then add some control to change the motor direction so we can go backwards.

The main processor can only supply enough power to light a LED, roughly 20mA. A motor will want at least 400mA of current to start turning.

Most of the low-level interfaces of Raspberry Pi hardware are not plug-and-play. To use these low-level interfaces, you must have a sound understanding of basic electrical concepts. If you mis-wire a GPIO pin, for example, you risk losing a GPIO pin, and, in some cases, your Raspberry Pi hardware.

2. THE ARCHITECTURE OF SYSTEM

Figure 1 shows the block diagram of the control system using the raspberry pi microcontroller.

2.1. Required Hardware

- A Raspberry Pi with SD card preinstalled with Raspbian;
- A Breadboard to connect everything on;
- An L293 or SN755410 motor driver chip (I will refer both as L293D in this work);
- Jumper cables to connect everything up (Male to male and female to male);
- One or two DC motors rated for 6v;
- 4x AA batteries and holder.

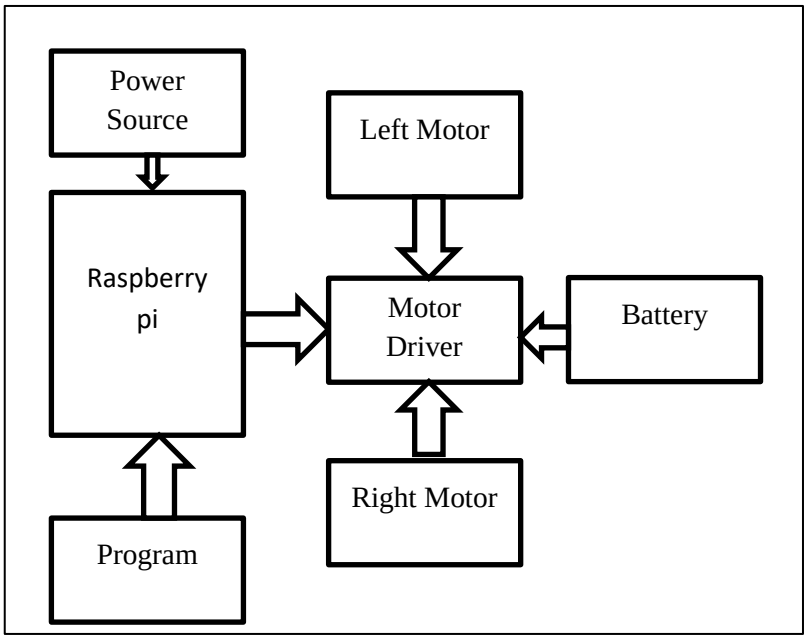

Figure 1. Block diagram of the control system using the raspberry pi

2.2. Microcontroller

GPIO pins

There are 26 pins grouped in two rows of 13, shown in the figure 2, and these collectively are called the *General Purpose Input Output header* or *GPIO* for short. These are a mix of four power pins, five ground pins and 17 data pins. Some of these data pins have extra functions such as an *i2c bus*, *SPI bus* and *UART* serial connectors, all of which can connect to other hardware to allow the Raspberry Pi to talk to items such as an *Arduino*, an *Analogue to Digital Convertor (ADC)* or add-on boards such as a *PiGlow* or *PiFace*.

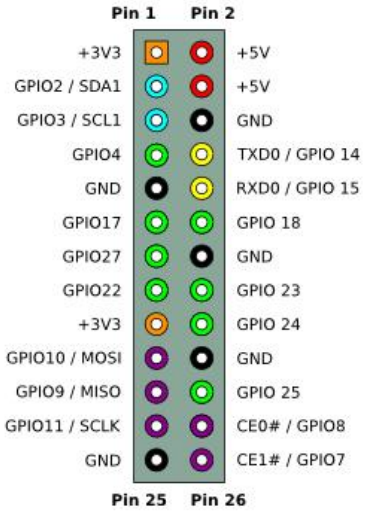

Figure 2. The Layout of the GPIOs with the pin numbering. Pin 1 is the top left labelled 3V3 [4]

2.3. Motor drivers L293D

The L293 and L293D are quadruple high-current half-H drivers. The L293 is designed to provide bidirectional drive currents of up to 1 A at voltages from 4.5 V to 36 V. The L293D is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5 V to 36 V. Both devices are designed to drive inductive loads such as relays, solenoids, dc and bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications. All inputs are TTL compatible. Each output is a complete totem-pole drive circuit, with a Darlington transistor sink and a pseudoDarlington source. Drivers are enabled in pairs, with drivers 1 and 2 enabled by 1,2EN and drivers 3 and 4 enabled by 3,4EN. When an enable input is high, the associated drivers are

enabled, and their outputs are active and enable input is low, those drivers are disabled, and their outputs are off and in the high-impedance state. With the proper data inputs, each pair of drivers forms a in phase with their inputs. When the full-H (or bridge) reversible drive suitable for solenoid or motor applications.

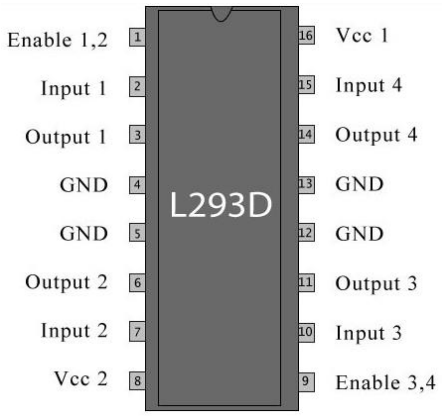

Figure 3. Pin diagram of L293 circuit [5]

3. ASSEMBLING THE CIRCUIT

3.1. Adding Power and Ground

It is important to do this while the power to the Raspberry Pi is off, or disconnected, as you want to avoid shorting any connectors by mistake.
The first thing you need to do is connect up the power and ground wires. As with most electronics projects, everything that connects together will require a common ground. This is shown with the black wires.
The ground on the Raspberry Pi is physical pin 6 (black color). Referring to figure 4 one this is worked out by starting at the top left with pin 3V3, counting left to right so 5V is pin 2 (red color), GPIO 2 (labelled 2) is pin 3.

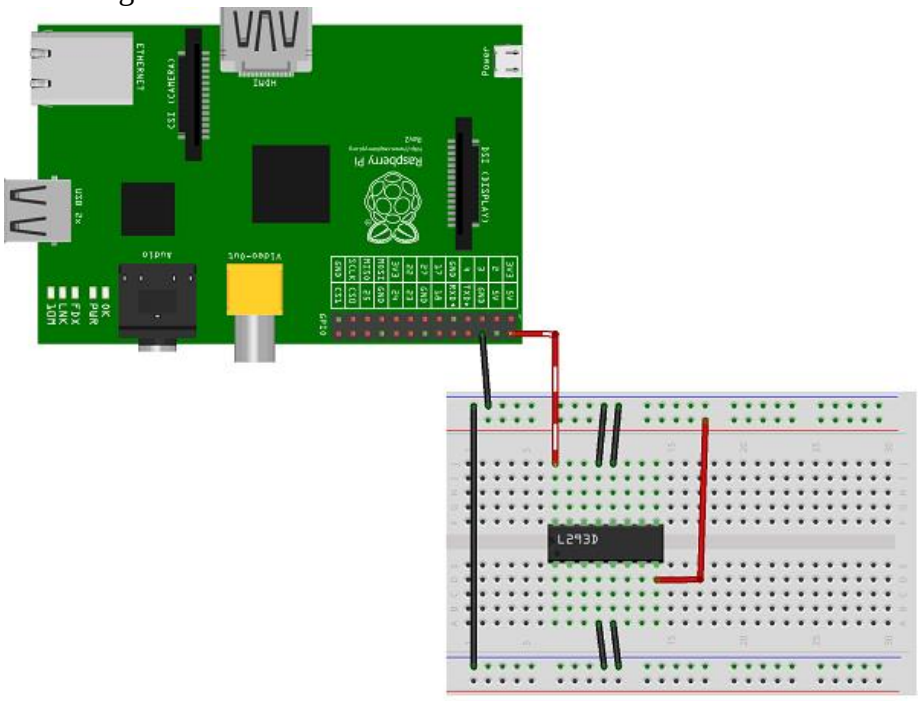

Figure 4. Connect the power and ground wires

3.2. Adding the Data Wires

Now add three wires from the GPIO pins to the L293D.

- GPIO 25–Pin 22(blue color) > L293D–Pin 1
- GPIO 24–Pin 18 (yellow color) > L293D–Pin 2
- GPIO 23–Pin 16 (yellow color) > L293D–Pin 7

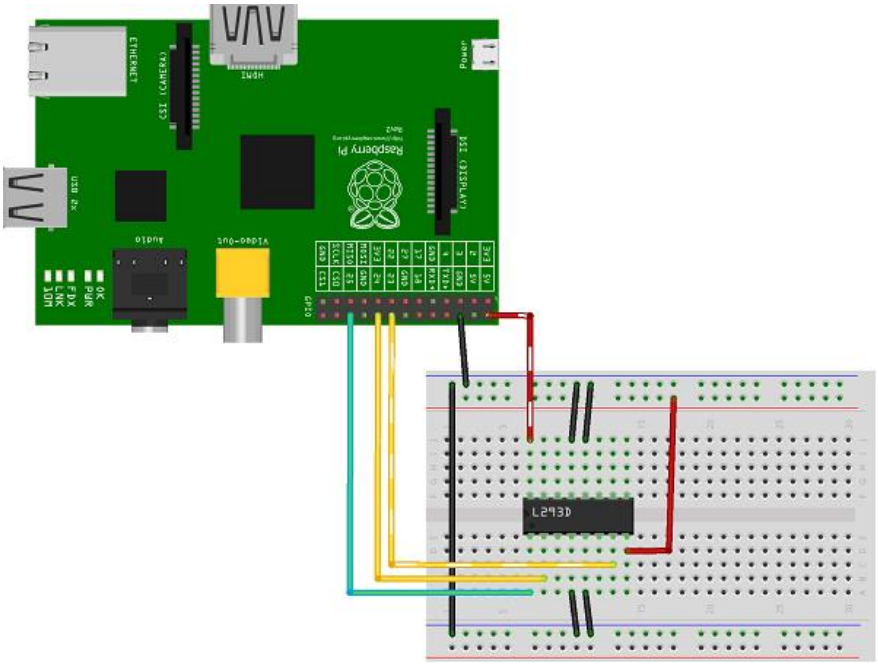

Figure 5. Add the three GPIO wires to control the motor

3.3. Adding the motors and baterry

Then connect the two motors and the batteries:

- Motor–wire 1(red color) > L293D–pin 3
- Motor–wire 2 (brown color) > L293D–pin 6

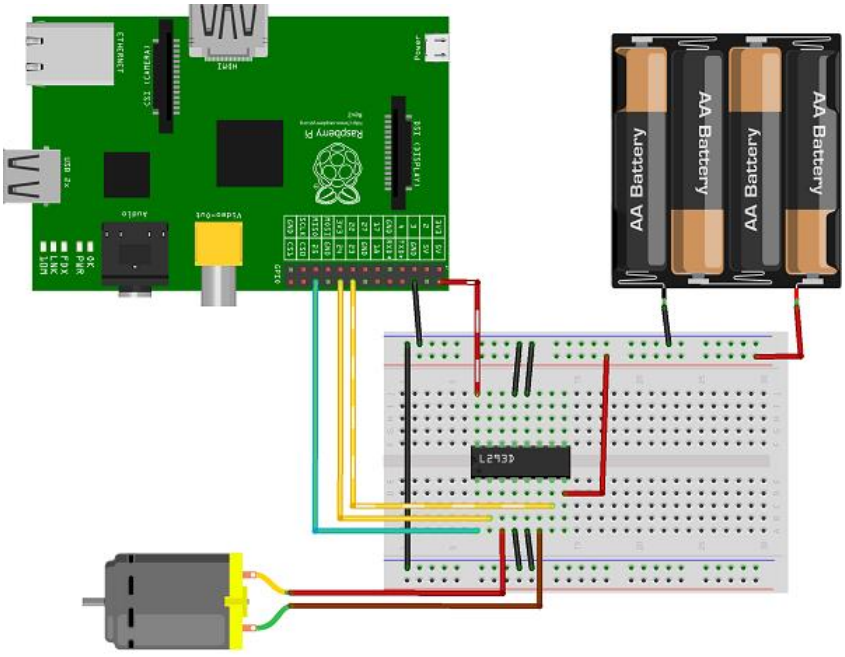

Figure 6. Attached the battery and the motor

Add a Second Motor

One of the great features of the L293D is it that it can handle two motors independently and each motor can run at different speeds or directions. Using this one IC makes it possible to create a two-wheeled robot capable of turning, going forwards and going backwards easily.

Adding a second motor involves just three additional wires and another motor.

- GPIO 11–Pin 23 > L293D–Pin 9
- GPIO 9–Pin 21 > L293D–Pin 10
- GPIO 10–Pin 19 > L293D–Pin 15

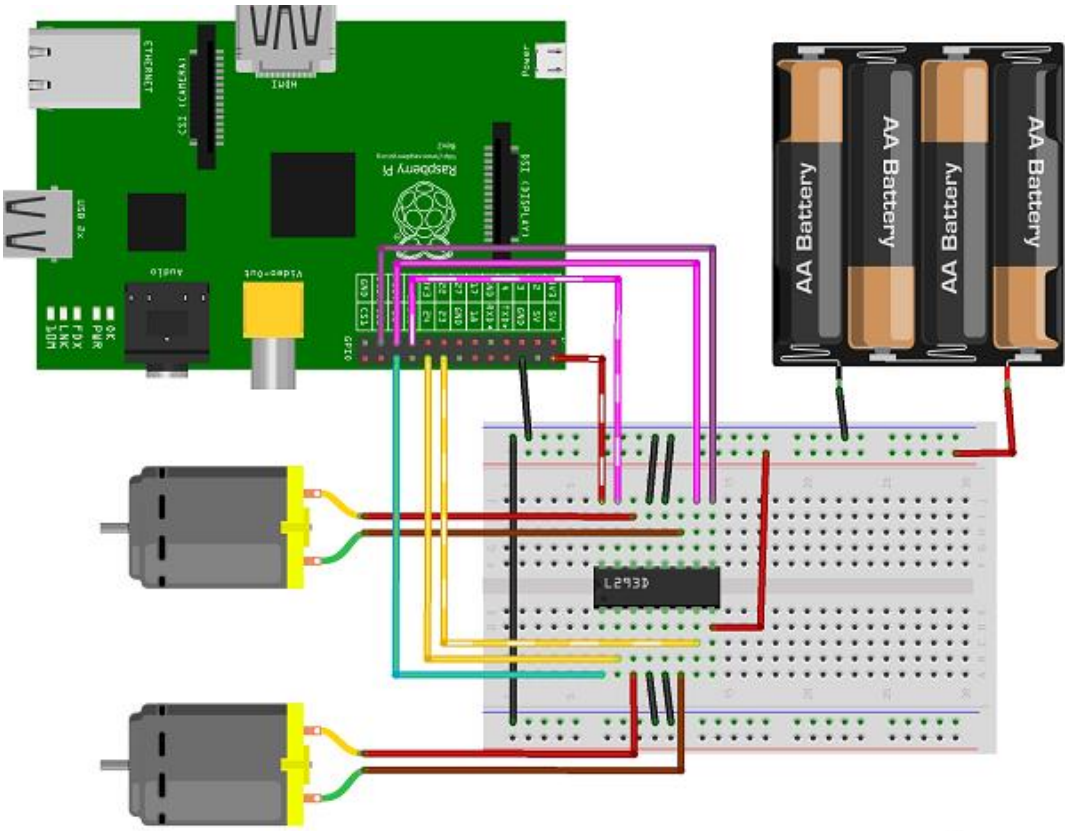

Figure 7. The complete set up ready for robotics

4. TESTING

In order to get the motors to work double-click **LXTerminal** on the desktop to bring up a terminal window. This is where you will be writing Python code using a program called *Nano*.

Nano is a text editor, similar to Notepad or TextEdit but for the command prompt, I will teach your some commands as we go along if you are new to it. To turn the motor on for two seconds use the following code:

```
import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode(GPIO.BOARD)
Motor1A = 16
Motor1B = 18
Motor1E = 22
GPIO.setup(Motor1A,GPIO.OUT)
GPIO.setup(Motor1B,GPIO.OUT)
GPIO.setup(Motor1E,GPIO.OUT)
print "Turning motor on"
GPIO.output(Motor1A,GPIO.HIGH)
GPIO.output(Motor1B,GPIO.LOW)
GPIO.output(Motor1E,GPIO.HIGH)
sleep(2)
print "Stopping motor"
GPIO.output(Motor1E,GPIO.LOW)
GPIO.cleanup()
```

The first two lines tell Python what is needed in the program. The first line will want to access a module called RPi.GPIO. This module handles all the hard work involved around turning the GPIO pins on and off on the Raspberry Pi.

The second line brings in sleep from the module time to make it possible to pause the script giving it time to perform a certain action, in this case leaving a motor on for a few seconds. The function setmode tells RPi. GPIO to use the board numbering on the Raspberry Pi. The numbers 16, 18 and 22 we will use to tell Python they are the pins associated with the motors.

When using the L293D you can give it a direction, by turning one side on to turn in one direction, called pin A and vice versa is pin B. To turn the motor on use a pin called Enable, labelled E in the test script–this is pin 22.

Finally, tell the Raspberry Pi these are all outputs which is done with GPIO.OUT. Save and exit by pressing CTRL-X, along the bottom a message asks you to confirm the changes. Press Y and Enter to confirm. Now you are back at the command prompt to run the script and see the motor spin to life.

For the motion backwards the script is very similar to the previous one, but if you notice for backwards we made Motor1A low and Motor1B high. High and low are programming names for on and off. To stop the motor you'll turn off, low, Motor1E.

Enable is the switch to turn the motor on and off, regardless of what A and B are doing.

In the table 1 is show the truth table of the motors functioning.

Table 1. Truth table

Enable	A	B	Result
Low	High	Low	Not spinning-Enable is off
Low	Low	High	Not spinning-Enable is off
High	Low	Low	Not spinning-both inputs are off
High	High	Low	Turning clockwise
High	Low	High	Turning counter-clockwise
High	High	High	Not spinning-both inputs are on

For two motors I created the following script. All that is different is a couple more lines to set up the second motor.

```
import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode(GPIO.BOARD)
Motor1A = 16
Motor1B = 18
Motor1E = 22
Motor2A = 19
Motor2B = 21
Motor2E = 23
GPIO.setup(Motor1A,GPIO.OUT)
GPIO.setup(Motor1B,GPIO.OUT)
GPIO.setup(Motor1E,GPIO.OUT)
GPIO.setup(Motor2A,GPIO.OUT)
GPIO.setup(Motor2B,GPIO.OUT)
GPIO.setup(Motor2E,GPIO.OUT)
print "Going forwards"
GPIO.output(Motor1A,GPIO.HIGH)
GPIO.output(Motor1B,GPIO.LOW)
GPIO.output(Motor1E,GPIO.HIGH)
GPIO.output(Motor2A,GPIO.HIGH)
GPIO.output(Motor2B,GPIO.LOW)
GPIO.output(Motor2E,GPIO.HIGH)
sleep(2)
print "Going backwards"
GPIO.output(Motor1A,GPIO.LOW)
GPIO.output(Motor1B,GPIO.HIGH)
GPIO.output(Motor1E,GPIO.HIGH)
GPIO.output(Motor2A,GPIO.LOW)
GPIO.output(Motor2B,GPIO.HIGH)
GPIO.output(Motor2E,GPIO.HIGH)
sleep(2)
print "Now stop"
GPIO.output(Motor1E,GPIO.LOW)
GPIO.output(Motor2E,GPIO.LOW)
GPIO.cleanup()
```

5. CONCLUSIONS

In this work I have shown you the basics of connecting motors to your Raspberry Pi. It may take a deep breath and can-do it attitude if you are new to connecting anything to your brand new Pi, but you will soon find that once you start playing with the GPIO pins that it is hard to stop.
This work opens the doors to making anything like robots with blinking LED lights and ultrasonic sensors in order to sense its environment.
Find a chassis to mount everything on use a USB mobile phone charger battery to make your Raspberry Pi fully mobile.

REFERENCES

[1]Cellan-Jones Rory, *A 15 pound computer to inspire young programmers*, 5 May 2011, http://www.bbc.co.uk/blogs/thereporters/rorycellanjones/2011/05/a_15_computer_to_inspire_young.html.

[2]Price Peter, *Can a 15 computer solve the programming gap?*, 3 June 2011, http://news.bbc.co.uk/2/hi/programmes/click_online/9504208.stm.

[3]Bush Steve, *Dongle computer lets kids discover programming on a TV*, Electronics Weekly, 25 May 2011.

[4]https://github.com/stephaneAG/RPi/blob/master/README.md

[5]https://www.amazon.com/Raspberry-Pi-RASP-PI-3-Model-Motherboard/dp/B01CD5VC92/?tag=howchoo-20