Reference Manual for the Control of Pneumatic Valve over a Local Network

Miguel Simão (miguel.simao@uc.pt)
Collaborative Robotics Group
Department of Mechanical Engineering
University of Coimbra

Abstract

Documentation for the setup that actuates the pneumatic valve at the base of the robot.

Contents

Acronyms

GND : Ground.

GPIO : General Purpose Input/Output (Raspberry Pi).

NC : Normally Closed.

NO : Normally Open.

RPi : Raspberry Pi.

1 Introduction

1.1 Usage

2 Hardware

In this section we list the hardware used and how it is set up.

2.1 Material List

The hardware used is the following:

1. Pneumatic valve with electric control
2. 24VDC power supply
3. Relay board (breakout)
4. 5VDC regulated power supply A regular USB power supply of at least 2.5A suffices.
5. Raspberry Pi 2
6. USB Wi-fi adapter
7. 1 Red LED
8. 1 100Ω resistor
9. 2 Yellow LEDs
10. 2 1500Ω resistors

2.2 Actuation Subsystem

The actuation subsystem is composed by the commutation pneumatic valve. It interfaces with the control subsystem. It moves the valve between the following two positions:

Ouputs:

1. P2 – Pneumatic output 2
2. P4 – Pneumatic output 4

It requires as input the 24 VDC signals and a GND connection (referenced to the input signals):

Inputs:

1. S1 – 24V signal 1 (S1)
2. S2 – 24V signal 2 (S2)
3. GND24 – Ground

The input signal S1 activates P2 and S2 activates P4.

A real representation in shown in fig. 1. In this figure, P1 is the pressured air input from its source, P3 and P5 are sinks, P12 and P14 are connected with P1 and assist the actuation of the valve.

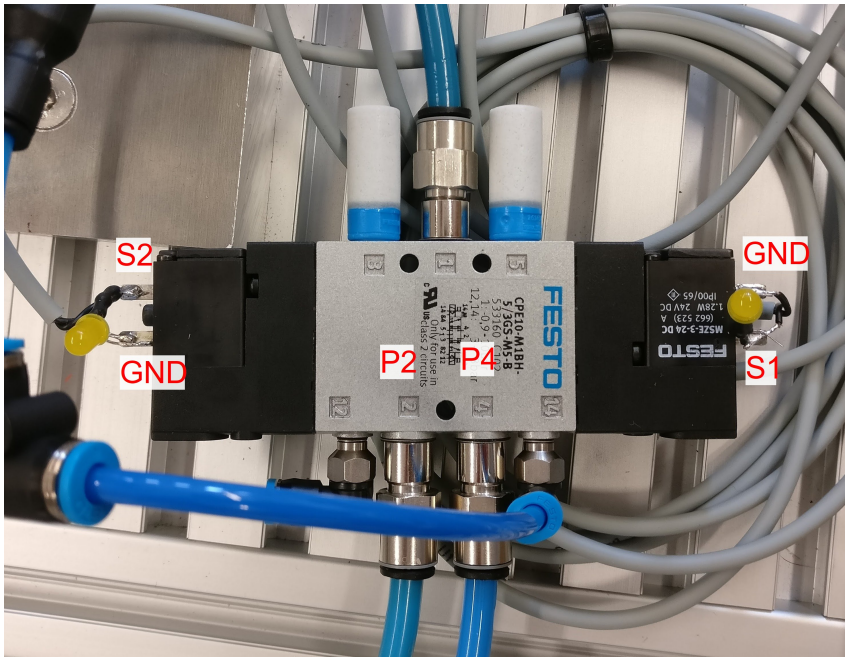

Figure 1: Actuation subsystem. Pneumatic valve with 3 positions and electrically-assisted control.

2.3 Control Subsystem

The control subsystem is composed of the relay and interfaces with the communications subsystem and the actuation subsystem.

→ See Section 2.2

The actuation valve is normally centred and requires two 24V signals to commutate between P2 and P4 (see section 2.2). To provide those, we use a single relay (fig. 2) to commutate a 24V source from a power supply, between the two signals S1 and S2. The control signal should be 5V but can be done with the 3.3V output level of the Raspberry Pi (RPi).

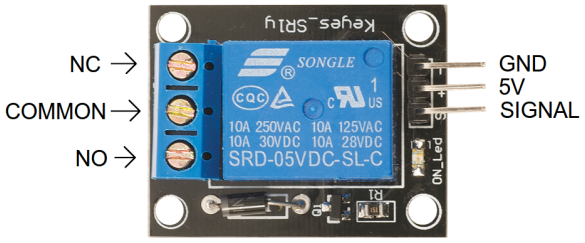

Figure 2: Board with one relay. NC stands for normally closed and NO for normally open.

When the RPi digital output is high (3.3V), the relay connects (left side of the picture) the common line to the Normally Open (NO) line. Otherwise, the Normally Closed (NC) line is active. The 5V (DC) input may come

from the RPi 5V line and Ground (GND) from the GND line on the RPi's General Purpose Input/Output (Raspberry Pi) (GPIO) pins.

Outputs:

1. S1 – 24V signal 1 (S1)
2. S2 – 24V signal 2 (S2)

Inputs:

1. S – Control signal (3.3V)
2. PWR24 – 24 VDC from source
3. PWR5 – 5 VDC from source
4. GND5 – Ground referenced to PWR5

Figure 3: Control subsystem. Single relay board with 24V rail on the left side and the control side on the right.

2.4 Communications Subsystem

The communications subsystem is a Raspberry Pi with a socket server running that allows remote connections. It is composed by the RPi (fig. 4) and an USB Wi-fi adapter OR an Ethernet connection. Both interfaces can also be connected (Ethernet + Wi-fi). For reference, the GPIO pin layout is shown in fig. 5.

→ See Figure 5 to see GPIO header pin layout.

A real representation of the subsystem in shown in fig. 4. Power to the RPi is provided by the μUSB input, but it is also possible to power it from a regulated[1] 5V power supply to pin 2. Regarding outputs, power and ground to the control subsystem are derived from the 5V output at pin 4 and 6. The control signal `S` is provided by pin 7 (GPIO4). There's also a red indication LED connected to pins 26 (anode) and 25 (cathode).

Input/output list:

Ouputs:

1. S – Control signal for the control subsystem
2. PWR5 – 5VDC (limited amperage)
3. GND5

Inputs:

1. Network (USB Wi-fi or Ethernet)
2. PWR5 – 5VDC from source or USB
3. (Optional) GND5 – Ground, if USB is NOT used

[1] Careful! There's no circuit protection on the GPIO header, so if your supply peaks above 5V, you can fry your RPi.

Figure 4: Real set-up of the communications subsystem.

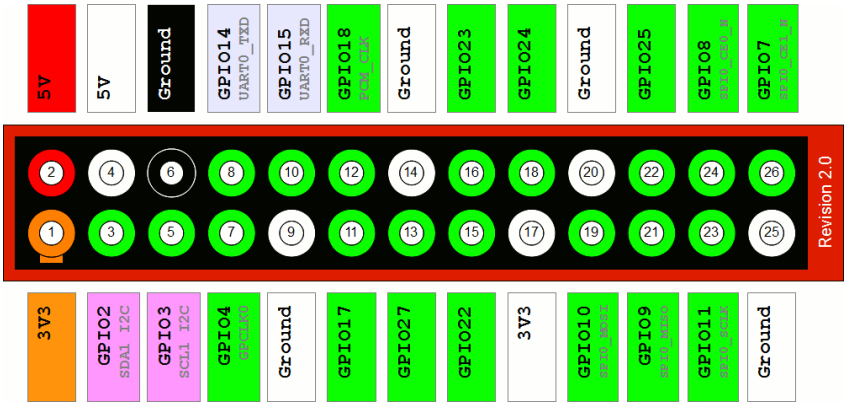

Figure 5: Raspberry Pi 2/3 GPIO layout.

2.5 Power Subsytem

There are two power rails in this system: 24 VDC and 5 VDC. There are separate power supplies for each, so the grounds are referenced individually. The 5 VDC supply is required for the Raspberry Pi and the control subsystem.

Ouputs:

1. PWR24 – 24 VDC source
2. GND24 – Ground for 24 VDC devices
3. PWR5 – 5 VDC source
4. GND5 – Ground for 5 VDC devices

Inputs:

1. Network (Wi-fi or Ethernet)
2. PWR5 – 5VDC from USB or, optionally, source
3. (Optional) GND – Ground, if USB is NOT used.

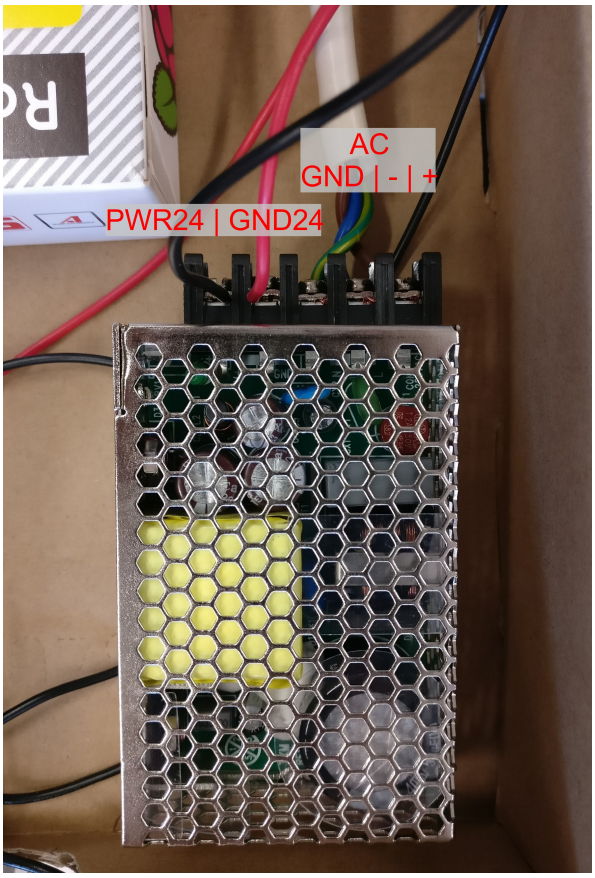

Figure 6: 24 VDC power supply.

2.6 Display subsystem

This subsystem is responsible for giving feedback to the user about the status of important signals in the different subsystems. For now, the only indicators we use are LEDs, which are:

1. Valve P2 status
2. Valve P4 status
3. RPi server status

All of the LEDs are connected in a circuit that is simplified in fig. 7. One yellow LED connects S1 to GND and another connects S2 to GND (see fig. 1). Since these poles have a voltage of 24V, the resistors were calculated to be of 1500Ω.

A red LED, is connected on the RPi's GPIO on the pin GPIO07 (3.3V, fig. 5) with a 100Ω resistor.

2.7 Full Setup

Schematically, the whole system is represented in fig. 8.

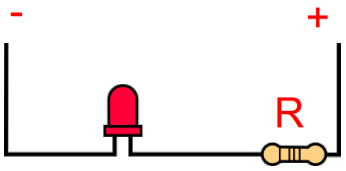

Figure 7: Simplified circuit diagram used to power an LED.

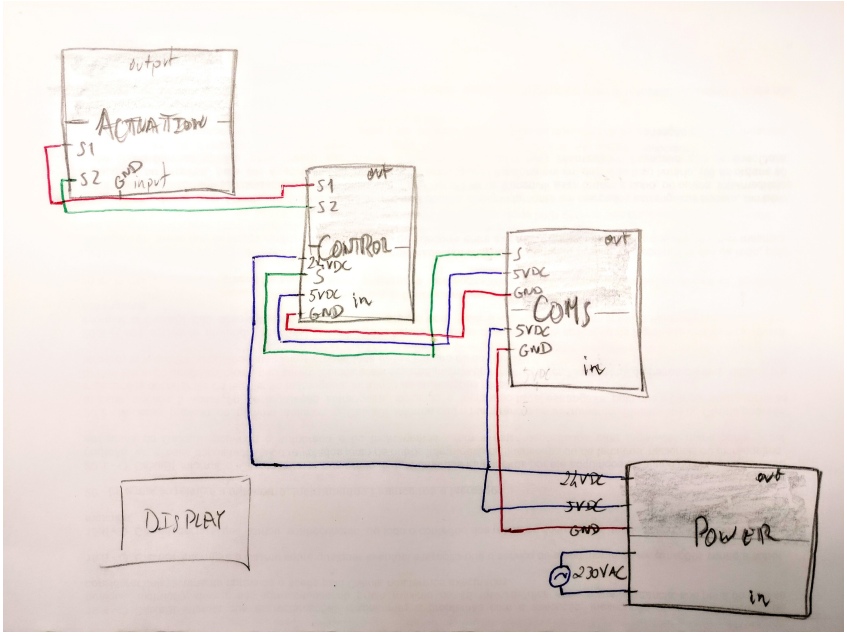

Figure 8: Subsystems' interactions.

3 Software

3.1 Operating System and Remote Access

OS Configuration

The Raspberry Pi is running a light version of Raspbian, Raspbian Jessie Lite[2]. This image does not have desktop environment, so all the configuration has to be done over a terminal. Configuration steps: [3]

1. Install OS image on the RPi's SD card
2. Connect a keyboard, Wi-fi adapter and screen to the RPi
3. Log into the OS with root user (pi)

   User name: *pi*

   Password: *raspberry*
4. Run `sudo raspi-config`

   Change the keyboard layout to *Portuguese* (all default)

   Enable SSH
5. Set network settings, both for wi-fi and Ethernet

[3] https://www.raspberrypi.org/downloads/raspbian/

6. Install `git`:

```
sudo apt-get update
sudo apt-get install git
```

At this point, the RPi is ready to run Python 2 code. All the following configuration can be done remotely.

We can access the RPi mainframe from another computer – preferably Linux or Windows with Putty – over the SSH protocol using the command:

```
ssh pi@RPI_IP_ADDRESS -p RPI_SSH_PORT
```

→ See section 3.3

where `RPI_PI_ADDRESS` is the IP address of the RPi and `RPI_SSH_PORT` is the SSH port allocated to the RPi (default: 20). There should be a SSH port for each of the devices in the local network, see section 3.3.

If the SSH connection was successful, you can manage the Pi. The first step is creating a known directory where to put the code into. Use the following code:

```
mkdir /python/socket_server
```

where you should replace `socket_server` by another name, if your following this documentation for another purpose.

3.1.1 Uploading or updating the code on the Pi

On the Pi, Python code is saved below the folder `python` on the home directory:

```
~/python
```

With individual code packages on the directories below:

```
~/python/socket_server
~/python/code_package_1
~/python/code_package_2
~/python/code_package_3
~/python/...
```

Copying files remotely

It is possible to copy the code remotely with the `scp` command. From your origin computer, run in one line:

```
scp -P RPI_SSH_PORT pi@RPI_IP_ADDRESS:~/python/code_package_x/FILE.py
/your/local/directory/FILE.py
```

! →

which copies the `FILE.py` to the specified `/python/code_package_x directory`. If there's already a file there with the same name, it will be overwritten without warning.

Alternatively, you can download code from a git repository[4]. You just need to go to the target directory, e.g.:

```
cd ~/python/socket_server
```

and close the repository:

```
cd ~/python/socket_server
```

[4] The author's git repository (master branch) can be found on: https://github.com/MiguelSimao/RPI_Control.git

3.2 Socket Server

3.3 Network Configuration