
Pidas Documentation

Release 1.0

Christophe Bastin

Sep 21, 2018

Contents:

1	Raspberry Pi configuration	1
1.1	Power with USB to TTL serial cable connection	1
1.2	Change keyboard layout	1
1.3	Create user	1
1.4	Static ip	2
1.5	Change computer name	2
1.6	Add 1-Wire support	2
1.7	Script to add Arduino vendor info	2
2	Connect 1-Wire sensors	5
2.1	Connection scheme to Pi GPIO	5
2.2	DS18B20 pinout	6
3	Getting started	7
3.1	DATABASE	7
3.2	CSV_HEADER	7
3.3	PIDAS_DIR	7
3.4	DATA_FILE	8
3.5	LOG_DIR	8
3.6	NB_SENSOR	8
3.7	SIMULATION_MODE	8
4	Automatization	9
5	How to use minicom	11
5.1	Installation	11
5.2	Configure	11
5.3	Serial port configuration	11
6	Indices and tables	13

1.1 Power with USB to TTL serial cable connection

- Pin 2: 5V(red)
- Pin 6: Gnd (Black)
- Pin 8 : Tx (White)
- Pin 10: Rx (Green)

1.2 Change keyboard layout

Install required packages:

```
sudo apt-get install console-data keyboard-configuration
```

Reconfigure and choose from the list:

```
dpkg-reconfigure console-data  
dpkg-reconfigure keyboard-configuration  
service keyboard-setup restart
```

1.3 Create user

```
useradd username -m -p password  
usermod -aG dialout,sudo username
```

1.4 Static ip

Edit `/etc/network/interfaces`:

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
    address X.X.X.X
    netmask 255.255.255.0
    gateway X.X.X.1
```

1.5 Change computer name

Edit `/etc/hosts`:

```
127.0.0.1    localhost
127.0.1.1    COMPUTER_NAME
```

Edit `/etc/hostname`:

```
COMPUTER_NAME
```

1.6 Add 1-Wire support

Start by adding the following line to `/boot/config.txt`

You can edit that file with nano by running `sudo nano /boot/config.txt` and then scrolling to the bottom and typing it there

```
dtoverlay=w1-gpio
```

Add the required modules at the bottom of `/etc/modules`

```
w1-gpio
w1-therm
```

reboot with **sudo reboot**

Test

```
cd /sys/bus/w1/devices
ls
cd 28-xxxx (change this to match what serial number pops up)
cat w1_slave
```

1.7 Script to add Arduino vendor info

This is useful to upload Arduino code from the Pi to the Arduino board

Create a file `get_arduino_id.sh`, and copy this content:

```
echo "Getting usb ids for Arduino..."
# Search for the keyword Arduino and print the sixth column of that line
vendorInfo=$(lsusb | awk '/Arduino/ {print $6}')
idVendor=${vendorInfo:0:4}
idProduct=${vendorInfo:5:4}
symlink="USBT001"
echo "Writting vendor info..."
echo "SUBSYSTEM M=='tty', ATTRS{idVendor}==$idVendor, ATTRS{idProduct}==$idProduct, \
↳SYMLINK=$symlink" > /etc/udev/rules.d/99-usb_serial.rules
```

Make it executable:

```
chmod +x get_arduino_id.sh
```

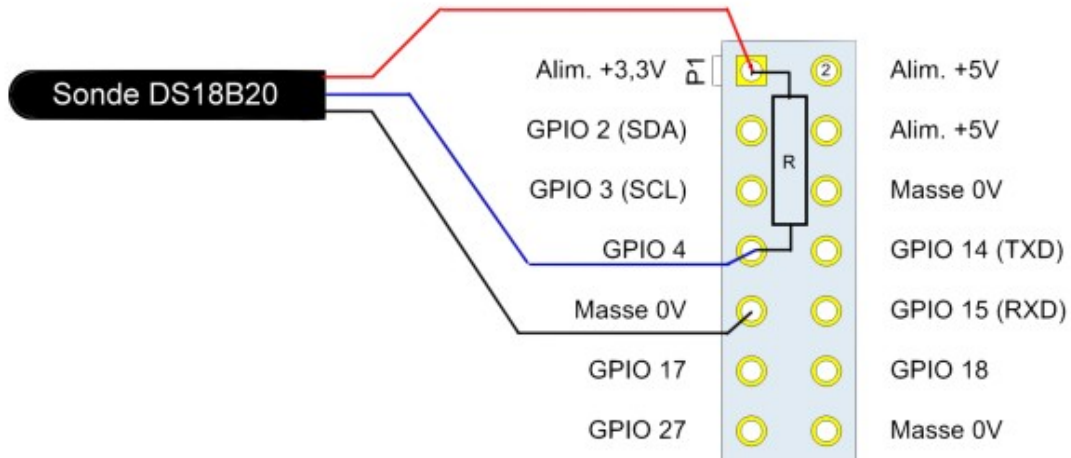
Connect your arduino board and launch the script:

```
sh get_arduino_id.sh
```

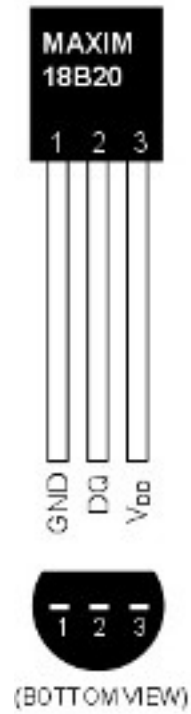

Connect 1-Wire sensors

2.1 Connection scheme to Pi GPIO

Don't forget the pull-up resistor (4,7K is fine)



2.2 DS18B20 pinout



CHAPTER 3

Getting started

Get the code:

```
git clone https://github.com/UMONS-GFA/pidas.git
```

Create a **settings.py** file in the pidas/pidas directory. You can now configure your custom settings.

3.1 DATABASE

InfluxDB is used. This can be configured using the following:

```
DATABASE = {
    'HOST': '127.0.0.1',
    'PORT': 8086,
    'USER': 'mydatabaseuser',
    'PASSWORD': 'mypassword',
    'NAME': 'mydatabase'
}
```

3.2 CSV_HEADER

Your CSV file header:

```
CSV_HEADER = ["sensorID", "sensorName", "value", "timestamp"]
```

3.3 PIDAS_DIR

The absolute path to the project:

```
PIDAS_DIR = '/home/USERNAME/pidas'
```

3.4 DATA_FILE

File where your data will be saved:

```
DATA_FILE = 'data.csv'
```

3.5 LOG_DIR

The relative path directory to your logs:

```
LOG_DIR = 'logs/'
```

3.6 NB_SENSOR

Number of sensors you want to generate:

```
NB_SENSOR = 8
```

3.7 SIMULATION_MODE

If simulation mode is set to 1, sensors will be created:

```
SIMULATION_MODE = 0
```

CHAPTER 4

Automatization

Edit your cron file:

```
crontab -e
```

Add your PYTHONPATH at the beginning of the file

```
PYTHONPATH=/home/USERNAME/pidas
```

And add a command to launch the script at boot:

```
@reboot /usr/bin/python3 /home/USERNAME/pidas/pidas/save_sensor_data.py >> /home/  
↳ USERNAME/pidas/pidas/cronlog 2>&1
```

How to use minicom

5.1 Installation

```
apt-get install minicom
```

5.2 Configure

```
sudo minicom -s
```

5.3 Serial port configuration

Press A to edit Serial port to : /dev/ttyUSB0 then press Enter

Press E to edit baud speed to 9600 then press Enter

Press F to disable hardware flow control

To get out, press Enter Save the config as dfl

Press Esc to use minicom

Utilisation

Press Ctrl+A then Z to enter the menu Press E to activate echo

Press Esc to return to minicom

To quit minicom, press Ctrl+A then Q to quit

CHAPTER 6

Indices and tables

- `genindex`
- `modindex`
- `search`