AZ-Delivery

Welcome!

Thank you very much for purchasing Raspberry Pi from our AZ-Delivery shop. On the following pages, we will introduce you to how to use and setup this handy device.

Have fun!

raspberrypi.org

Contents

1.	Introduction	3
1.1.	Install and start the operating system	4
1.2.1.	Basic settings of the Pi	11
1.2.2.	Remote access to the Pi	16
1.2.3.	Change the resolution of the screen	24
1.3.	Preinstalled apps on the Raspbian	26
2.	GPIO	27
2.1.	GPIO as output	30
2.2.	LED blinking light	33
2.3.	Traffic light	38
2.4.	GPIO as input	40
2.5.	Temperature sensor	46

1. Introduction

Raspberry Pi (from now on, Pi) is credit-card sized single computer. It has main processor, RAM, graphics chip, peripherals like 4x USB 2.0 ports and one microUSB for power supply, LAN port, HDMI port, female 3.5mm audio port, camera port, DSI display port, microSD card port and 40 GPIO pins. You can read about specifications in our datasheet on our site.

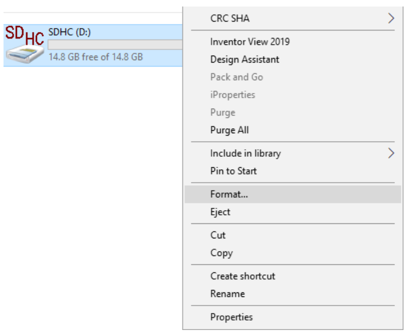
So Pi is an computer with no hard drive. To operate with it we need a hard drive (which in this case is SD card), power supply, a screen, a keyboard, a mouse and an internet connection.

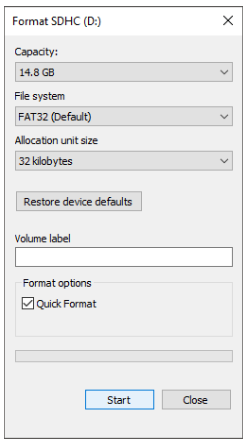
There is option called "*headless*", where we can use our Pi with only microSD, power supply and internet connection. In this case we don't use mouse, keyboard and screen. For this option, we first have to setup our Pi for headless operation using screen, keyboard and mouse (later in text) and then we can use it in headless mode.

1.1. Installing and starting the operating system

Every computer needs an operating system, including the Pi. You can run Windows operating system on the Raspberry Pi, but it is recommendable to run a system that is designed for Pi, and it is a Linux distro, more specifically, Raspbian. The Raspbian is operating system based on other known Linux distro called Debian. Some examples of Linux distros are Fedora, Ubuntu, Knoppix, etc. (you can read about them online). The name Raspbian is too composed of these two words Raspberry & Debian.

The Pi comes without hard drive, and as normal computer, it too needs a hard drive where the operating system need to be installed. The difference here is that the SD memory card serves as a hard disk. For memory card we need microSD card with at least 8GB, that is minimum requirement (in this manual we will use 16GB microSD card). Firstly we must format SD card, but be careful with formatting SD card, all data on it will be lost, so if you have any data on SD card, worth saving, back it up.




Format SDHC (D:)

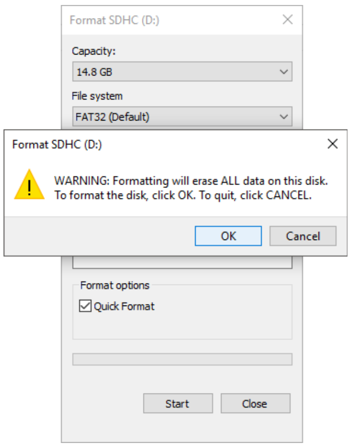
Capacity: 14.8 GB

File system: FAT32 (Default)

Allocation unit size: 32 kilobytes

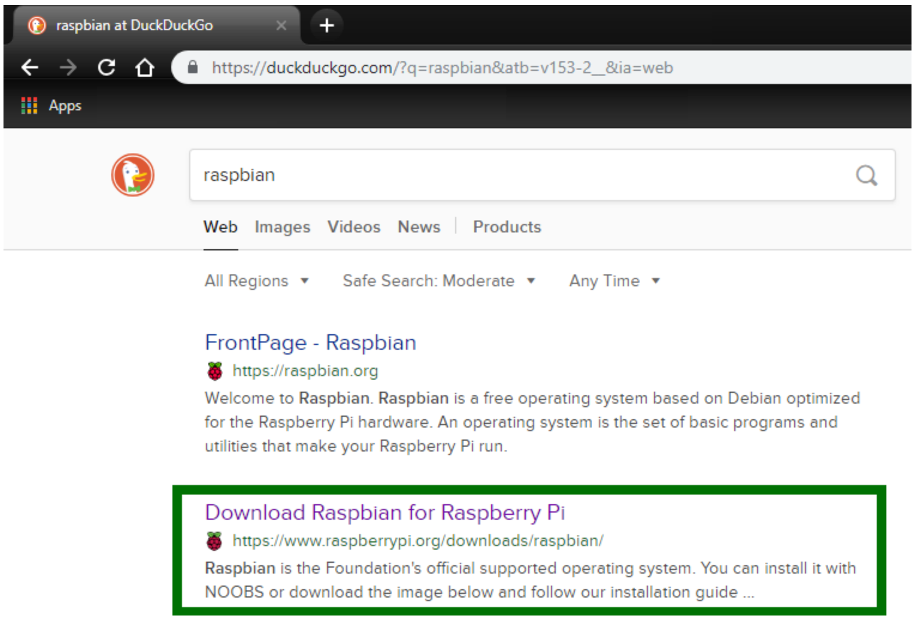
Volume label:

Format options: ☑ Quick Format


Format SDHC (D:)

WARNING: Formatting will erase ALL data on this disk.
To format the disk, click OK. To quit, click CANCEL.

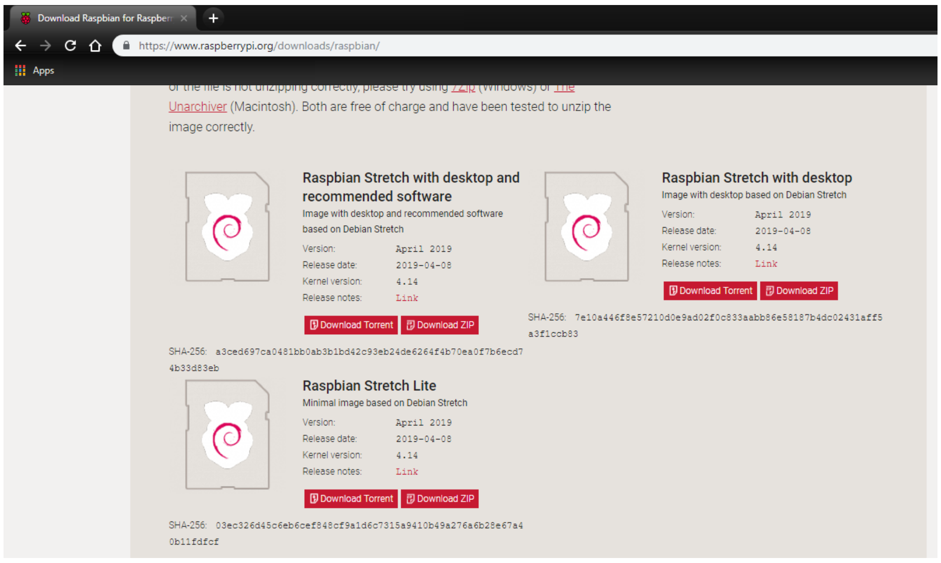
Then we download Raspbian from
https://www.raspberrypi.org/downloads/raspbian/
Just duckduckgo (google it) "*raspbian*" like on image below:



FrontPage - Raspbian
https://raspbian.org
Welcome to Raspbian. Raspbian is a free operating system based on Debian optimized for the Raspberry Pi hardware. An operating system is the set of basic programs and utilities that make your Raspberry Pi run.

Download Raspbian for Raspberry Pi
https://www.raspberrypi.org/downloads/raspbian/
Raspbian is the Foundation's official supported operating system. You can install it with NOOBS or download the image below and follow our installation guide ...

On their site find "*Raspbian Stretch with desktop and recommended software*" and download .zip file.


Download Raspbian for Raspberr
https://www.raspberrypi.org/downloads/raspbian/
Apps
Download Torrent
Download ZIP

Unarchiver (Macintosh). Both are free of charge and have been tested to unzip the image correctly.

Raspbian Stretch with desktop and recommended software
Image with desktop and recommended software based on Debian Stretch

Version:	April 2019
Release date:	2019-04-08
Kernel version:	4.14
Release notes:	Link

SHA-256: a3ced697ca0481bb0ab3b1bd42c93eb24de6264f4b70ea0f7b6ecd74b33d83eb

Raspbian Stretch with desktop
Image with desktop based on Debian Stretch

Version:	April 2019
Release date:	2019-04-08
Kernel version:	4.14
Release notes:	Link

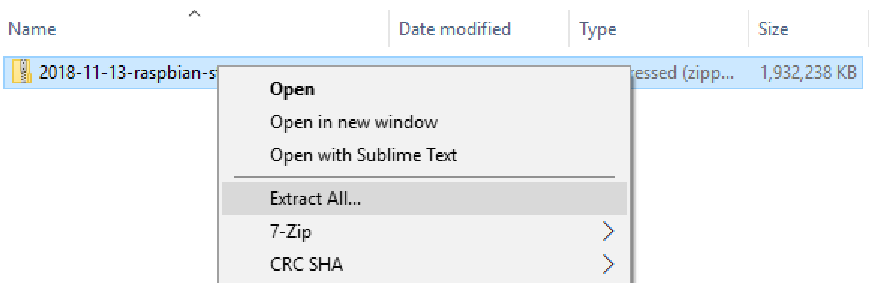
SHA-256: 7e10a446f8e57210d0e9ad02f0c833aabb86e58187b4dc02431aff5a3f1ccb83

Raspbian Stretch Lite
Minimal image based on Debian Stretch

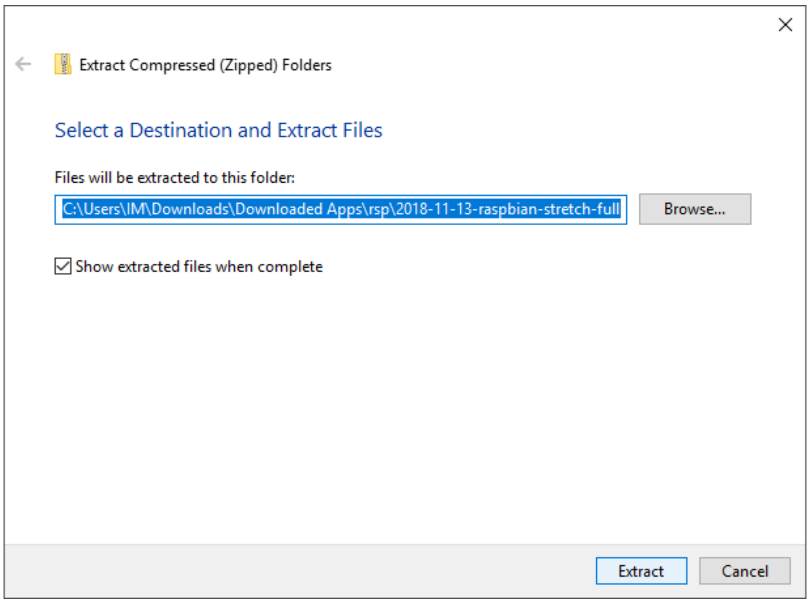
Version:	April 2019
Release date:	2019-04-08
Kernel version:	4.14
Release notes:	Link

SHA-256: 03ec326d45c6eb6cef848cf9a1d6c7315a9410b49a276a6b28e67a40b11fdfcf

When you download .zip file extract it.



Name	Date modified	Type	Size
2018-11-13-raspbian-s		essed (zipp...	1,932,238 KB



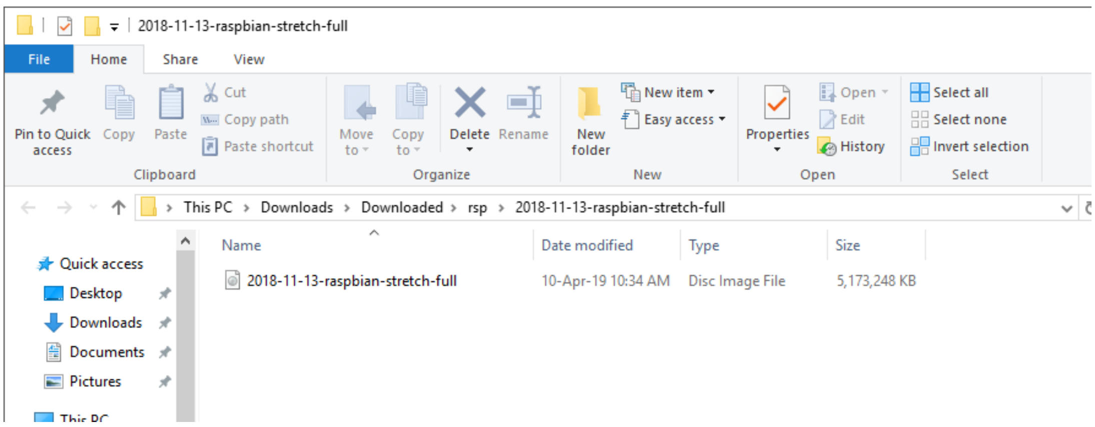
Select a Destination and Extract Files

Files will be extracted to this folder:

C:\Users\IM\Downloads\Downloaded Apps\rsp\2018-11-13-raspbian-stretch-full

[✓] Show extracted files when complete

You'll get .img file:



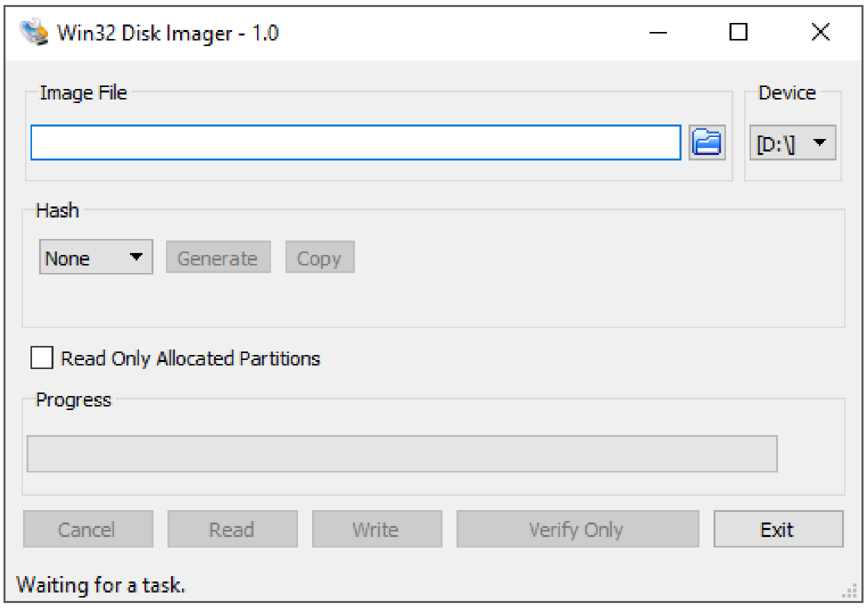
This PC › Downloads › Downloaded › rsp › 2018-11-13-raspbian-stretch-full

Name	Date modified	Type	Size
2018-11-13-raspbian-stretch-full	10-Apr-19 10:34 AM	Disc Image File	5,173,248 KB

Then download an app “*Win32DiskImager*” from
https://raspberry-projects.com/pi/pi-operating-systems/win32diskimager

When you install it (installation process is not something special, one “*I agree*” few nexts and finish xD) this icon will be on the desktop

Double click on it and new app window will be opened



Click on blue folder like icon, in "*Image File*" field and find and add you extracted .img file like on image.

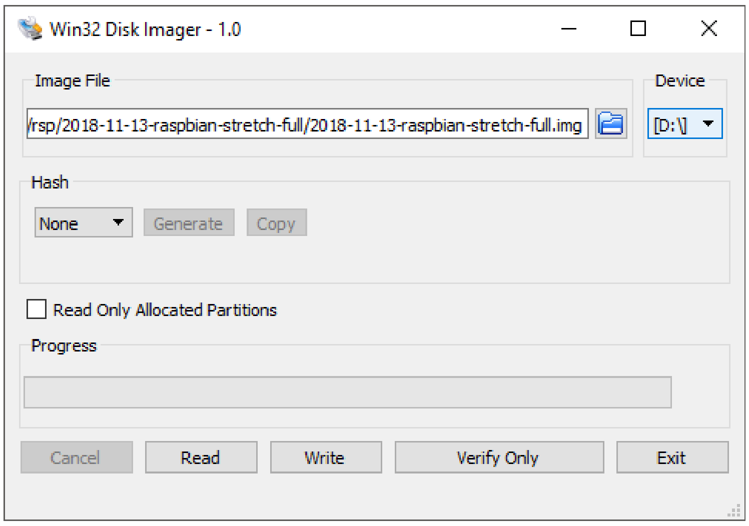


Image File: /rsp/2018-11-13-raspbian-stretch-full/2018-11-13-raspbian-stretch-full.img

Device: [D:\]

Hash: None

☐ Read Only Allocated Partitions

In "*Device*" field find your SD card name and select it. But check which **disk identification** SD card has, in our case it is D: (yours can be E:, F:, G: etc. **just double check it before continuing on next step!!!**)

Click on "*Write*" button, and this will pop up, confirm it.

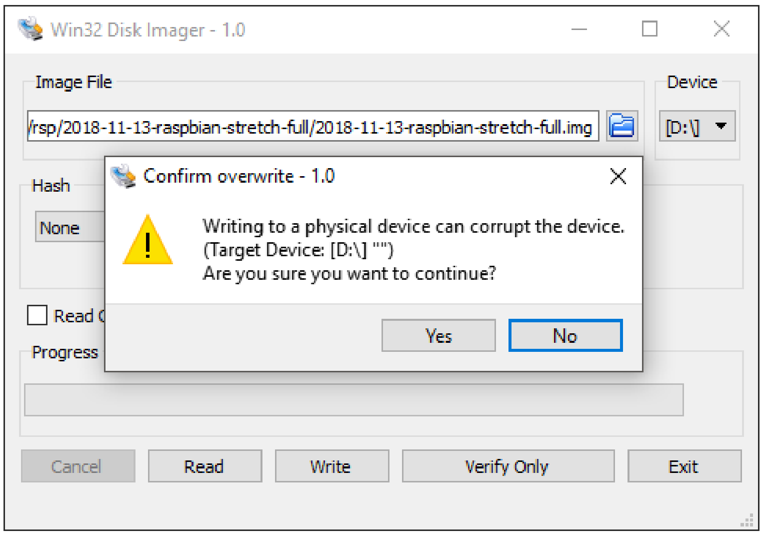


Image File: /rsp/2018-11-13-raspbian-stretch-full/2018-11-13-raspbian-stretch-full.img

Device: [D:\]

Hash: None

Confirm overwrite - 1.0

Writing to a physical device can corrupt the device.
(Target Device: [D:\] "")
Are you sure you want to continue?

Wait few minutes for process to complete,

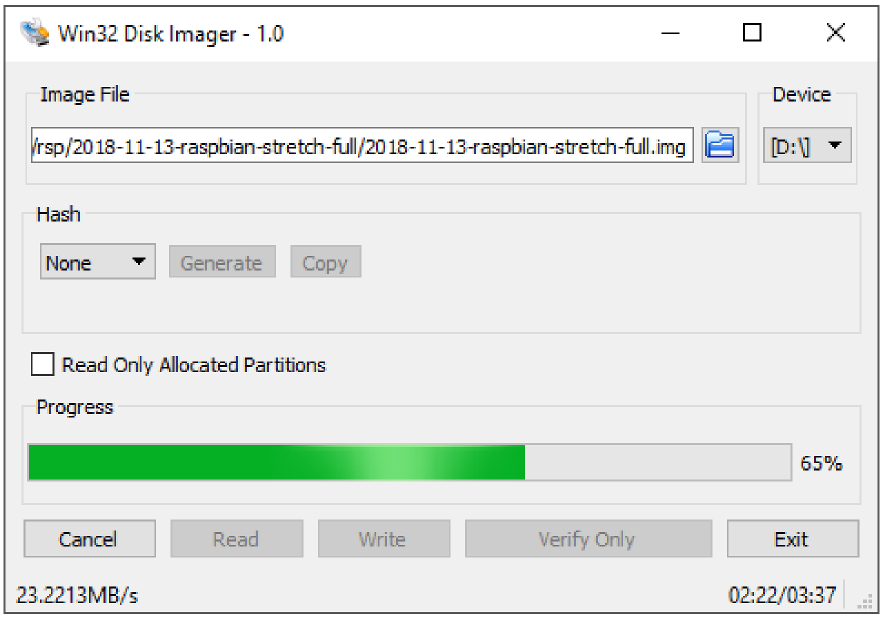

Win32 Disk Imager - 1.0

Image File: /rsp/2018-11-13-raspbian-stretch-full/2018-11-13-raspbian-stretch-full.img

Device: [D:\]

Hash: None

Read Only Allocated Partitions

Progress: 65%

23.2213MB/s 02:22/03:37

and when it is done, this will pop up.

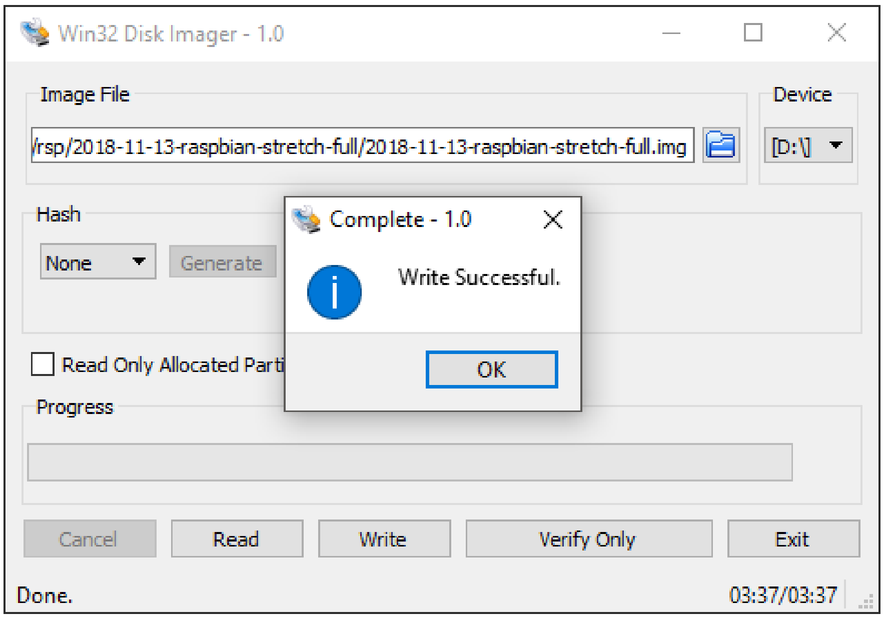

Win32 Disk Imager - 1.0

Image File: /rsp/2018-11-13-raspbian-stretch-full/2018-11-13-raspbian-stretch-full.img

Device: [D:\]

Hash: None

Complete - 1.0

Write Successful.

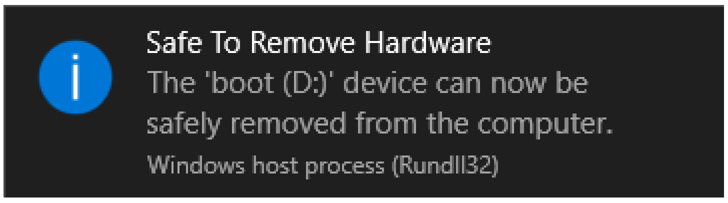
Read Only Allocated Parti

Progress

Done. 03:37/03:37

And that is it. You can plug in your microSD card into your Pi.

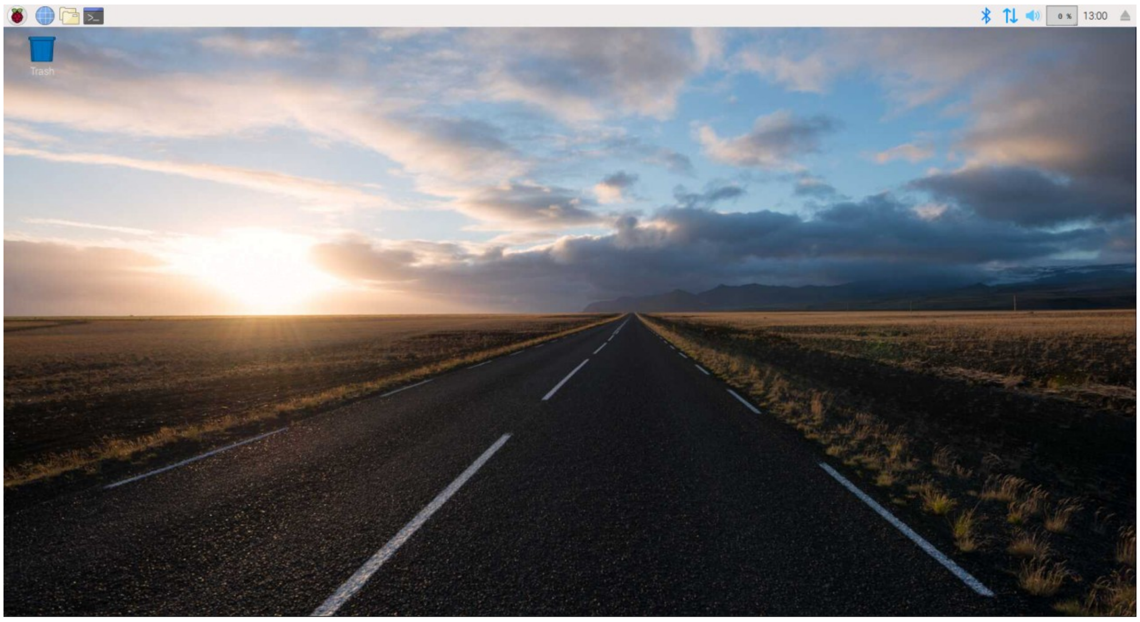
But make sure to safely remove SD from your PC.



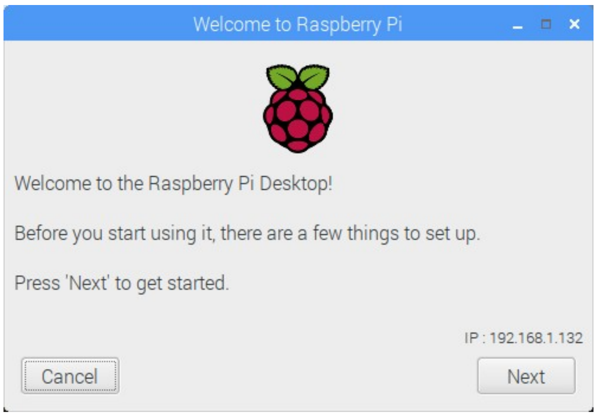
Safe To Remove Hardware
The 'boot (D:)' device can now be safely removed from the computer.
Windows host process (Rundll32)

1.2.1. Basic settings of the Pi

Connect screen via HDMI cable, mouse and keyboard via USB cables, and plug in freshly prepared microSD card into your Pi. When you plug in your microSD card into your Pi for the first time after setting Raspbian on it and power up your Pi, Raspbian operating system will boot up with no installation process. A new desktop environment GUI will be displayed on your screen.



We used LAN cable to connect Pi to the internet, but you can connect it wirelessly too (later in the text). When Raspbian is boot up for the first time, this will pop up. This is guide to quickly setup most important settings. Click next to continue.


Welcome to Raspberry Pi

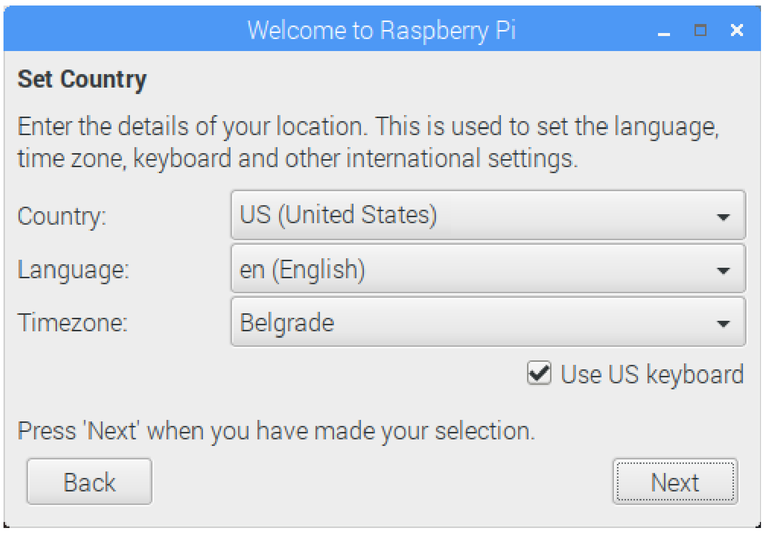
Welcome to the Raspberry Pi Desktop!

Before you start using it, there are a few things to set up.

Press 'Next' to get started.

IP : 192.168.1.132

The next step is to choose your country, desired language and time zone for Raspbian.


Welcome to Raspberry Pi

Set Country

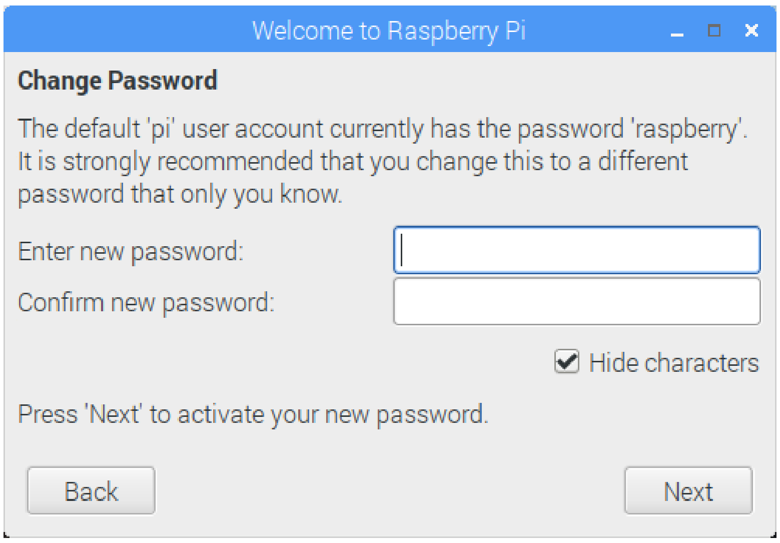
Enter the details of your location. This is used to set the language, time zone, keyboard and other international settings.

Country: US (United States)
Language: en (English)
Timezone: Belgrade

✓ Use US keyboard

Press 'Next' when you have made your selection.

Click next to continue. The next step is to choose new password for your Raspbian.


Welcome to Raspberry Pi

Change Password

The default 'pi' user account currently has the password 'raspberry'. It is strongly recommended that you change this to a different password that only you know.

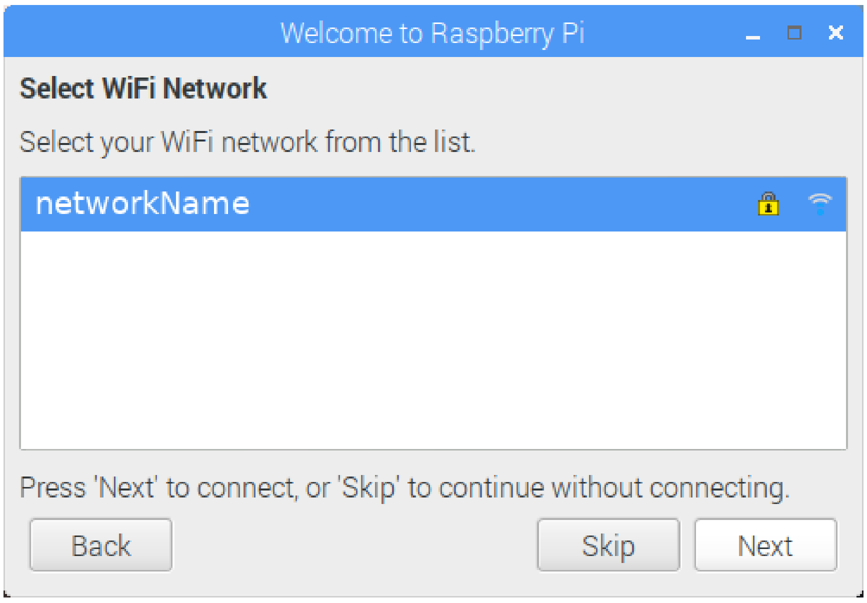
Enter new password:
Confirm new password:

✓ Hide characters

Press 'Next' to activate your new password.

When you are finished click next to continue.

The next step is to connect Raspbian to wireless network, you can skip it if you want to use LAN cable for internet connection. Just for example, we will show you how to connect to it. Select desired wifi network and click next.


Welcome to Raspberry Pi

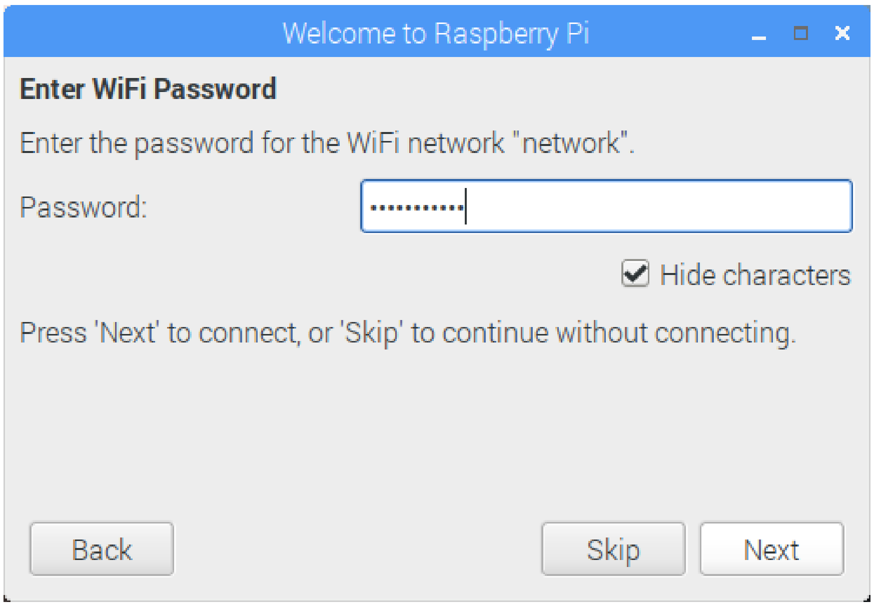
Select WiFi Network

Select your WiFi network from the list.

networkName

Press 'Next' to connect, or 'Skip' to continue without connecting.

In next step you have to enter password for selected wifi network. To continue click next.


Welcome to Raspberry Pi

Enter WiFi Password

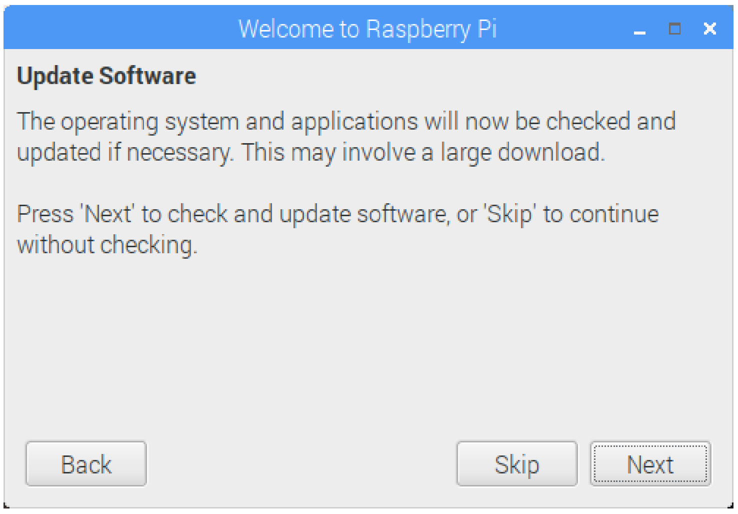
Enter the password for the WiFi network "network".

Password: ••••••••••••

[✓] Hide characters

Press 'Next' to connect, or 'Skip' to continue without connecting.

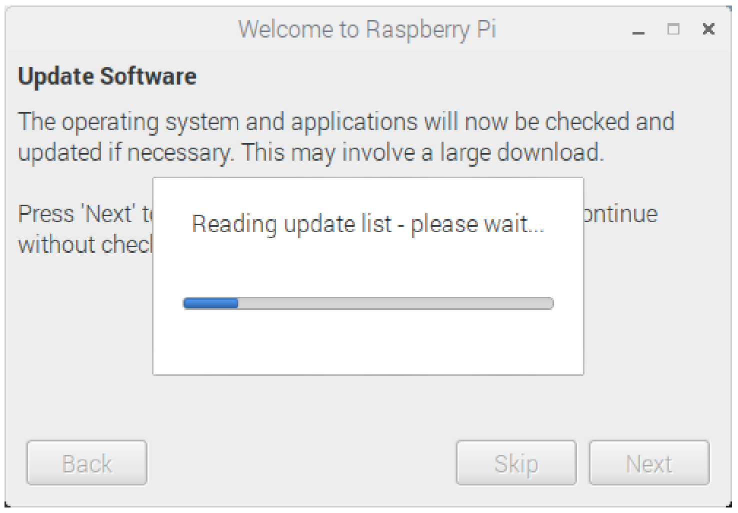
The next step is to update Raspbian. Click next to continue.


Welcome to Raspberry Pi

Update Software

The operating system and applications will now be checked and updated if necessary. This may involve a large download.

Press 'Next' to check and update software, or 'Skip' to continue without checking.

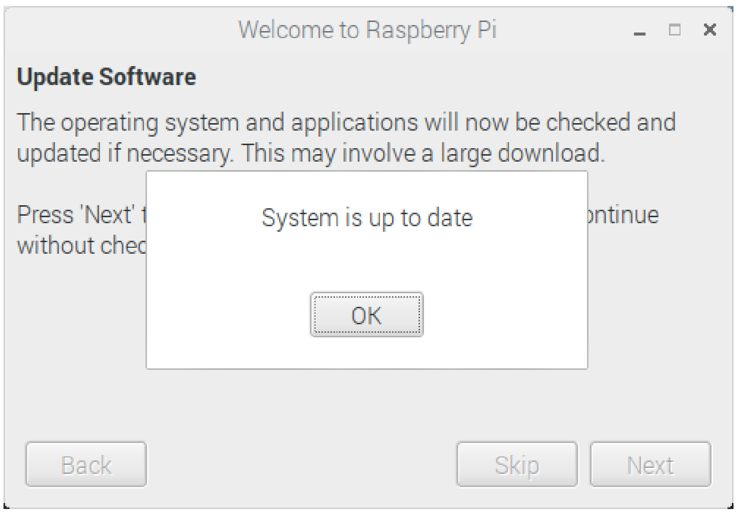

Welcome to Raspberry Pi

Update Software

The operating system and applications will now be checked and updated if necessary. This may involve a large download.

Press 'Next' t ... ontinue without checl

Reading update list - please wait...


Welcome to Raspberry Pi

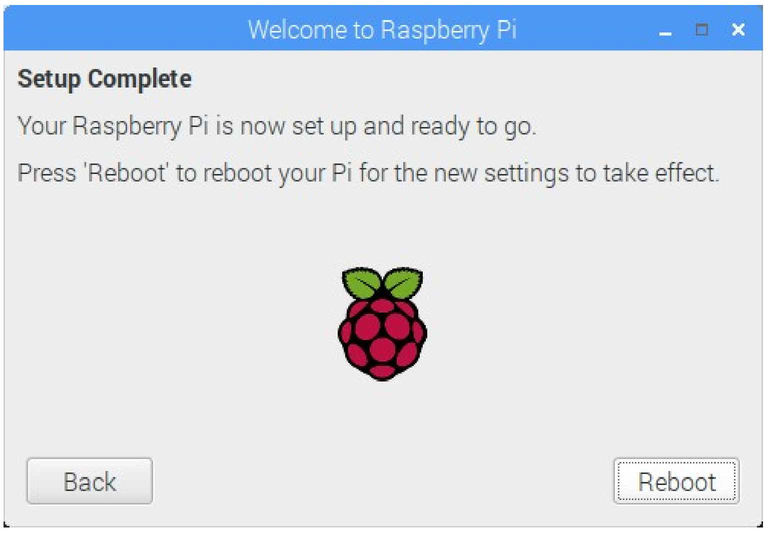
Update Software

The operating system and applications will now be checked and updated if necessary. This may involve a large download.

Press 'Next' t ... ontinue without chec

System is up to date

Wait for few minutes and when it is done. The next step is to reboot Raspbian.


Welcome to Raspberry Pi

Setup Complete

Your Raspberry Pi is now set up and ready to go.

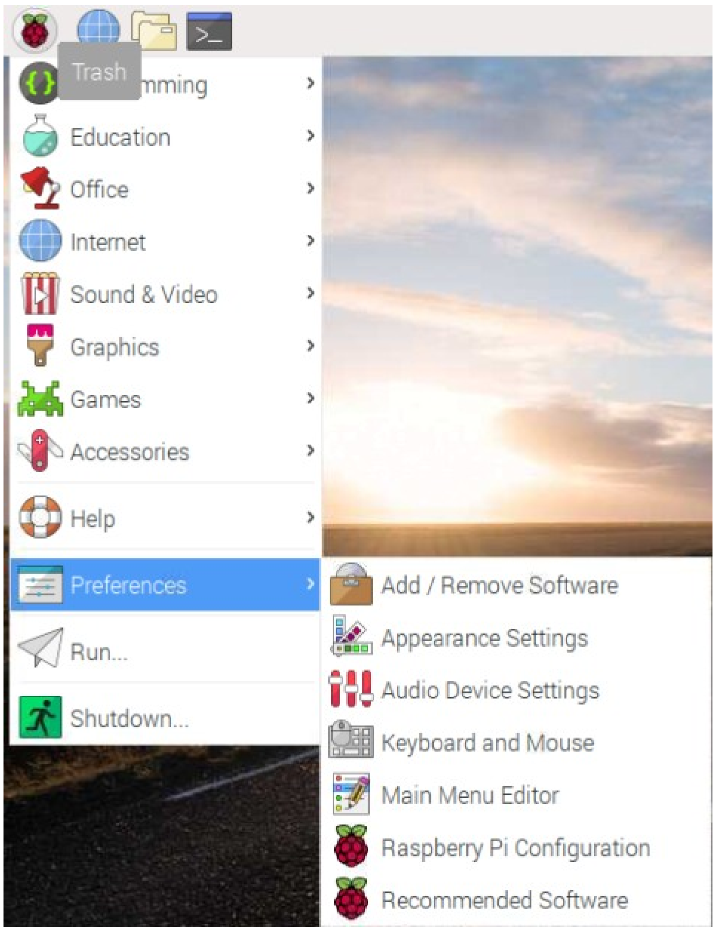
Press 'Reboot' to reboot your Pi for the new settings to take effect.

Reboot it, and you are ready for next step.

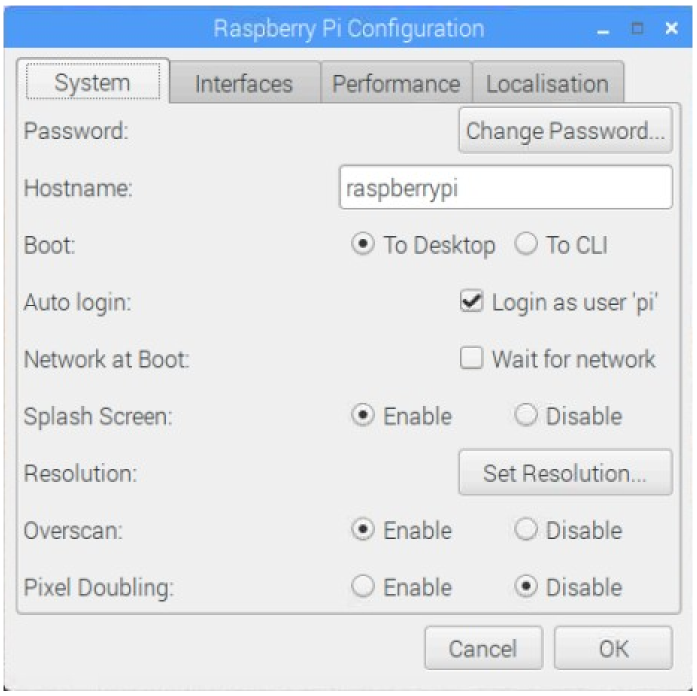
1.2.2. Remote access to the Pi

Most of us do not have spare screen for new Pi and when you hear that it can be operated without one, for sure you would want to use it remotely. To do this we need to setup VNC server and viewer.

Go on: *Start > Preferences > Raspberry Pi Configuration*



New window will open.



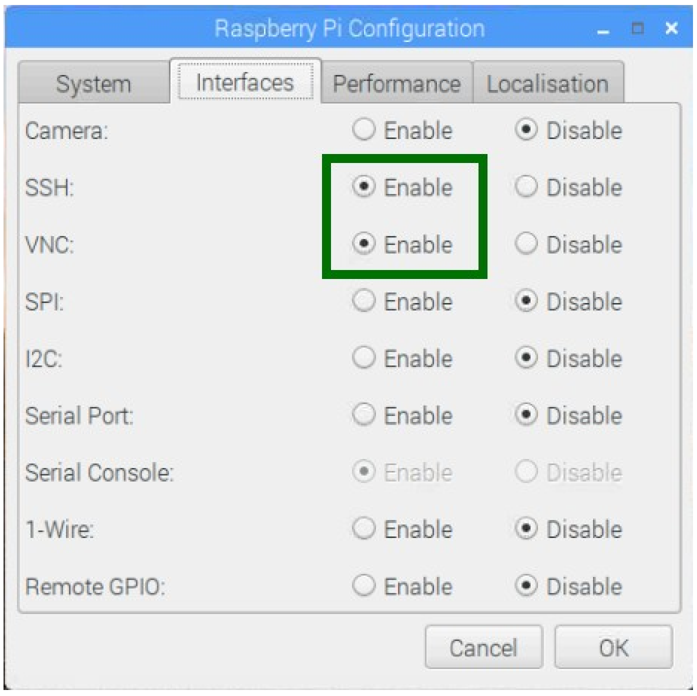
Raspberry Pi Configuration

System | Interfaces | Performance | Localisation

Password:	Change Password...	
Hostname:	raspberrypi	
Boot:	[✓ To Desktop]	To CLI
Auto login:	[✓ Login as user 'pi']	
Network at Boot:	Wait for network	
Splash Screen:	[✓ Enable]	Disable
Resolution:	Set Resolution...	
Overscan:	[✓ Enable]	Disable
Pixel Doubling:	Enable	[✓ Disable]

Cancel OK

Click on tab "*Interfaces*" and select enable for SSH and VNC like on image below.



Raspberry Pi Configuration

System | Interfaces | Performance | Localisation

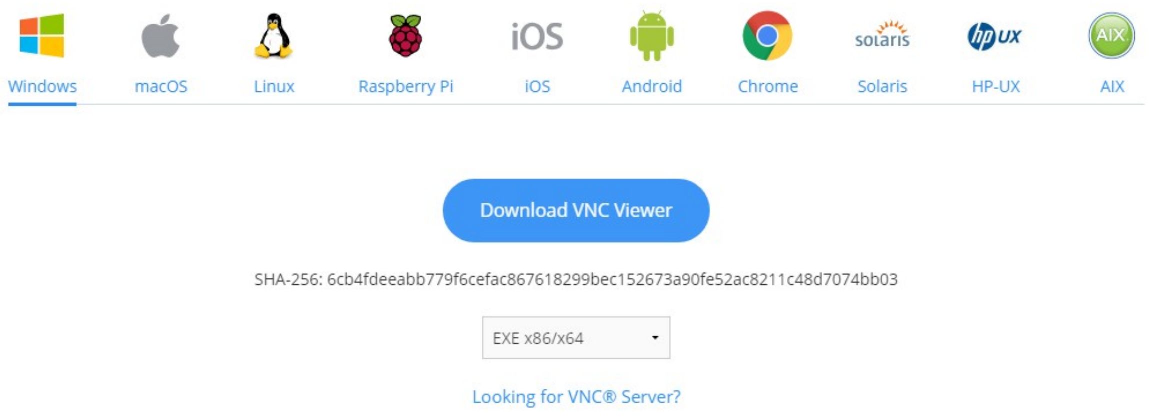
Camera:	Enable	[✓ Disable]
SSH:	[✓ Enable]	Disable
VNC:	[✓ Enable]	Disable
SPI:	Enable	[✓ Disable]
I2C:	Enable	[✓ Disable]
Serial Port:	Enable	[✓ Disable]
Serial Console:	[✓ Enable]	Disable
1-Wire:	Enable	[✓ Disable]
Remote GPIO:	Enable	[✓ Disable]

Cancel OK

Click ok and you are done. Setting up of your Pi is done.

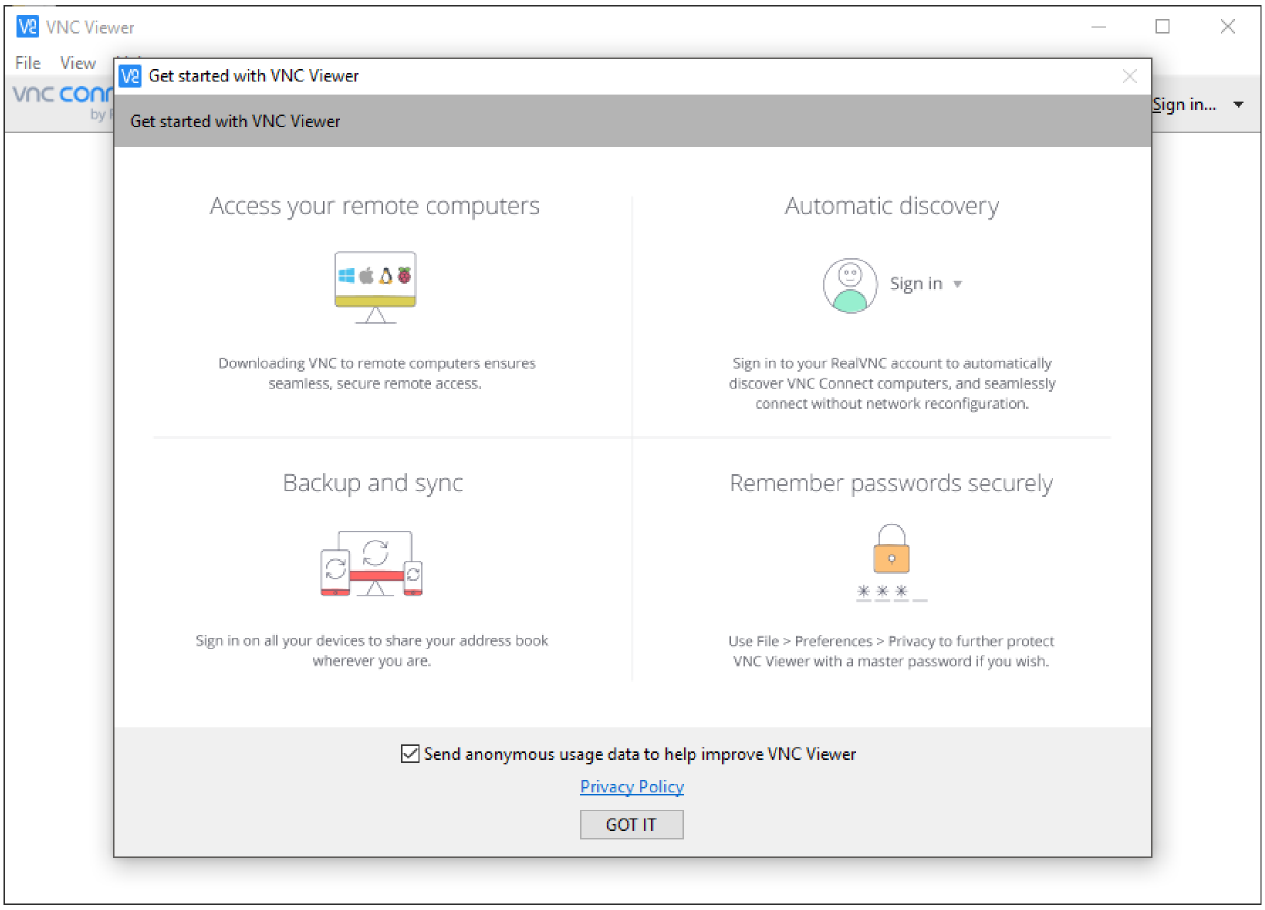
Now we need to install VNC Viewer on some other computer from which you want to access your Pi. To do this you have to download RealVNC Viewer app. Go on this link and download an app for your operating system:

https://www.realvnc.com/en/connect/download/viewer/



SHA-256: 6cb4fdeeabb779f6cefac867618299bec152673a90fe52ac8211c48d7074bb03

When you download app, install it (installation process is not something special, one "*I agree*" few nexts and finish). After installation start an app, and new window will open.


VNC Viewer

Get started with VNC Viewer

Access your remote computers

Downloading VNC to remote computers ensures seamless, secure remote access.

Automatic discovery

Sign in to your RealVNC account to automatically discover VNC Connect computers, and seamlessly connect without network reconfiguration.

Backup and sync

Sign in on all your devices to share your address book wherever you are.

Remember passwords securely

Use File > Preferences > Privacy to further protect VNC Viewer with a master password if you wish.

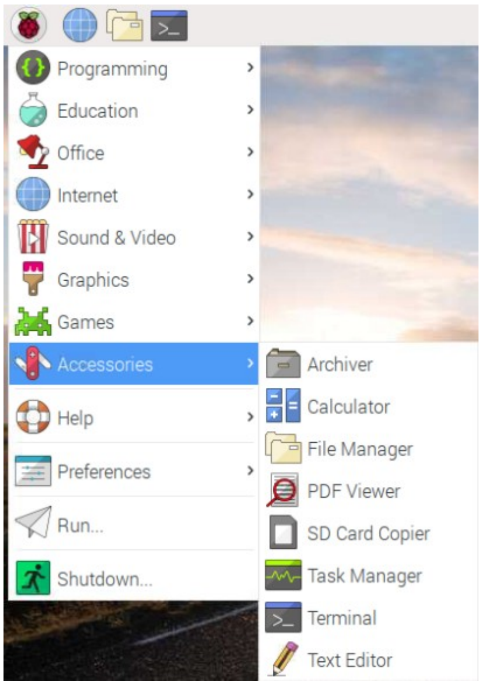
☑ Send anonymous usage data to help improve VNC Viewer

Privacy Policy

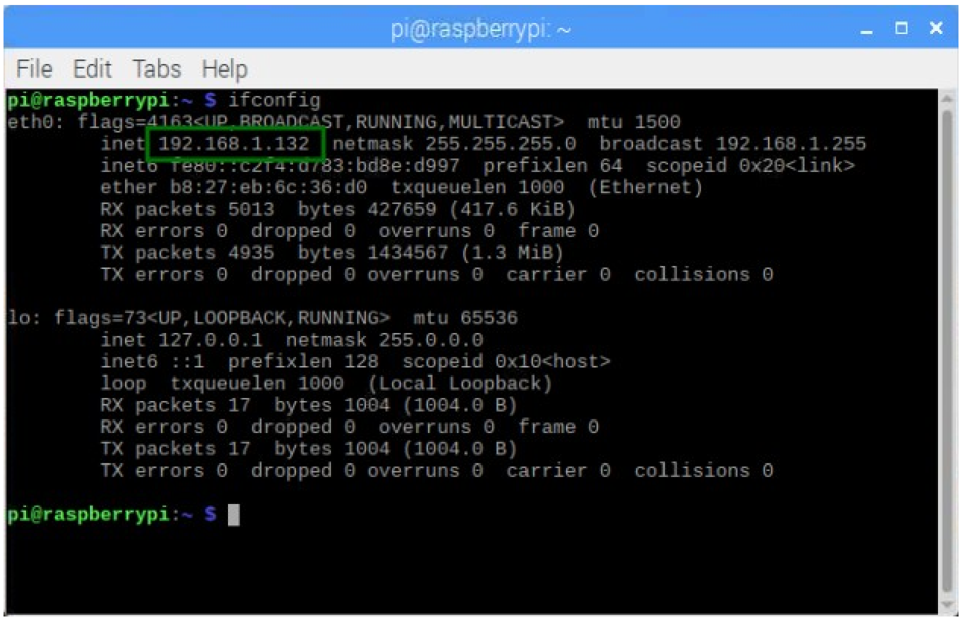
GOT IT

Click on "*GOT IT*" and default window of VNC will open.

For next step you will need an IP address of your Raspberry Pi. To find out IP address in Raspbian, go to *Start* > *Accessories* > *Terminal*



A new Terminal window will open. Type "ifconfig" and you will see IP address like on image below.



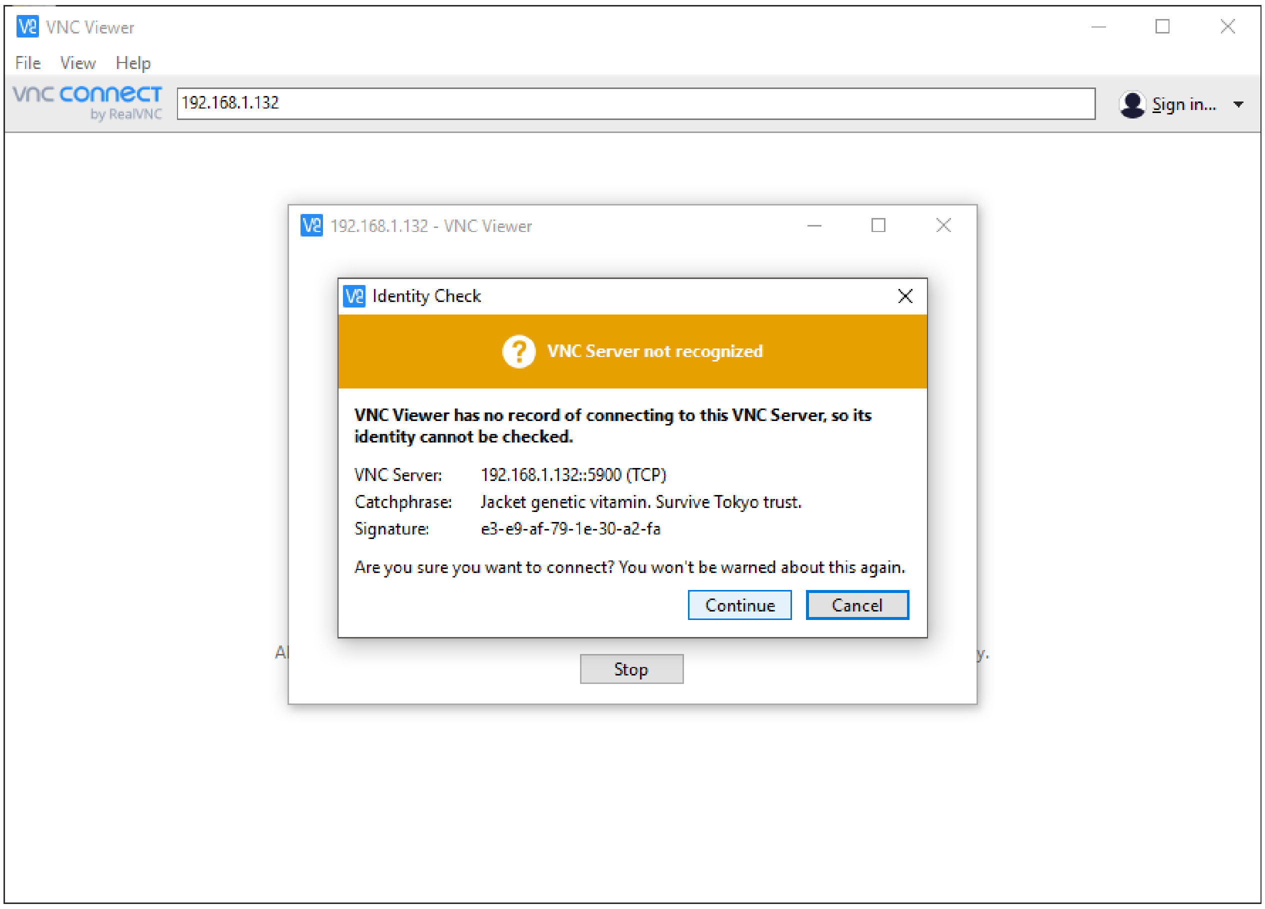
```
pi@raspberrypi:~ $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.132  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::c2f4:d783:bd8e:d997  prefixlen 64  scopeid 0x20<link>
        ether b8:27:eb:6c:36:d0  txqueuelen 1000  (Ethernet)
        RX packets 5013  bytes 427659 (417.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4935  bytes 1434567 (1.3 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 17  bytes 1004 (1004.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 17  bytes 1004 (1004.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

pi@raspberrypi:~ $ 
```

In our case IP address of our Pi is "*192.168.1.132*".

In default window of VNC Viewer, in text field "*Enter a VNC Server address or serach*" enter IP address of your Pi and hit enter. New window will pop up. Click on "Continue" like on image below.

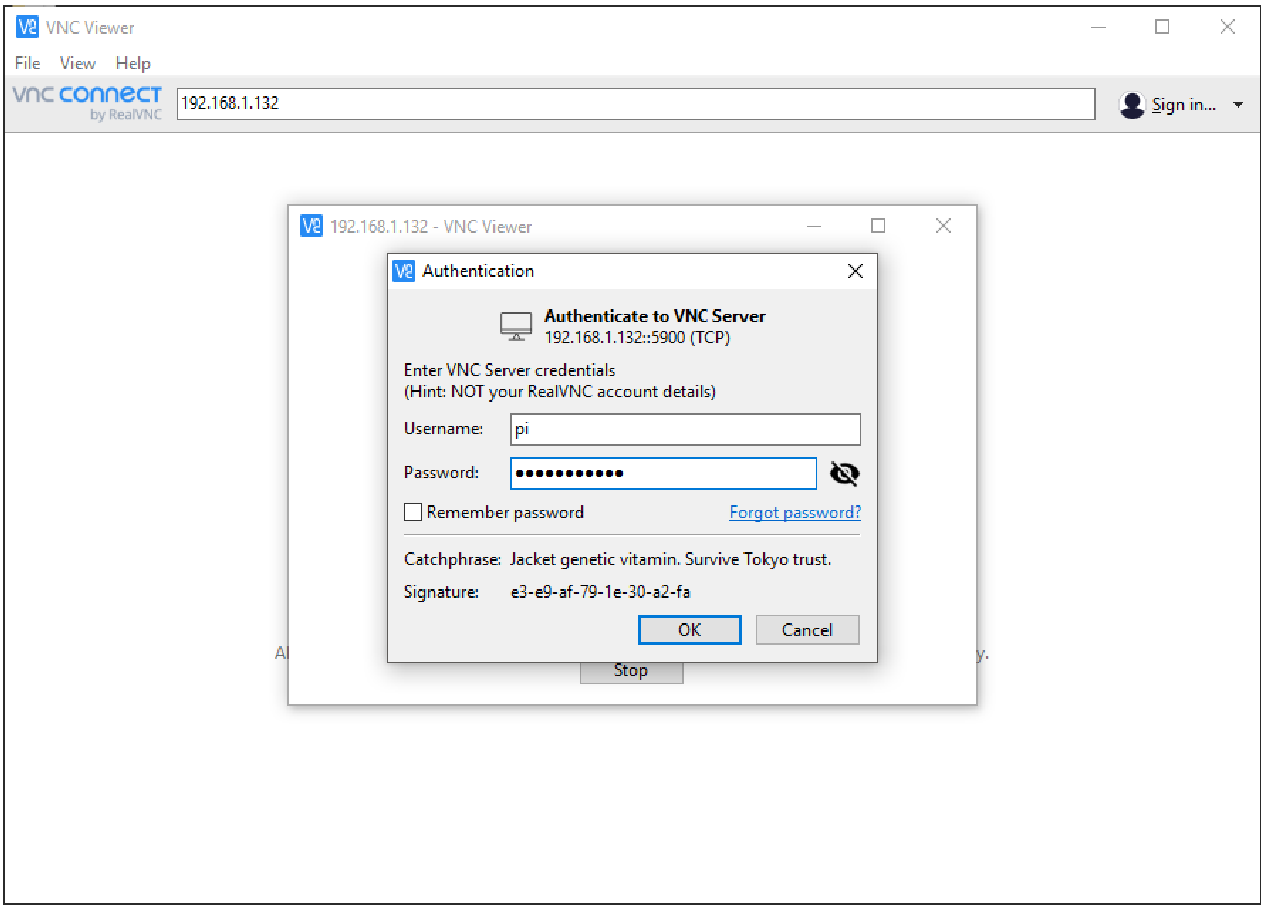


VNC Viewer has no record of connecting to this VNC Server, so its identity cannot be checked.

VNC Server: 192.168.1.132::5900 (TCP)
Catchphrase: Jacket genetic vitamin. Survive Tokyo trust.
Signature: e3-e9-af-79-1e-30-a2-fa

Are you sure you want to connect? You won't be warned about this again.

After that you have to enter Username and Password for your Raspbian. Default username is "pi" and password is the one you entered in one of previous setup steps. When you are done hit "*OK*".



Authenticate to VNC Server
192.168.1.132::5900 (TCP)

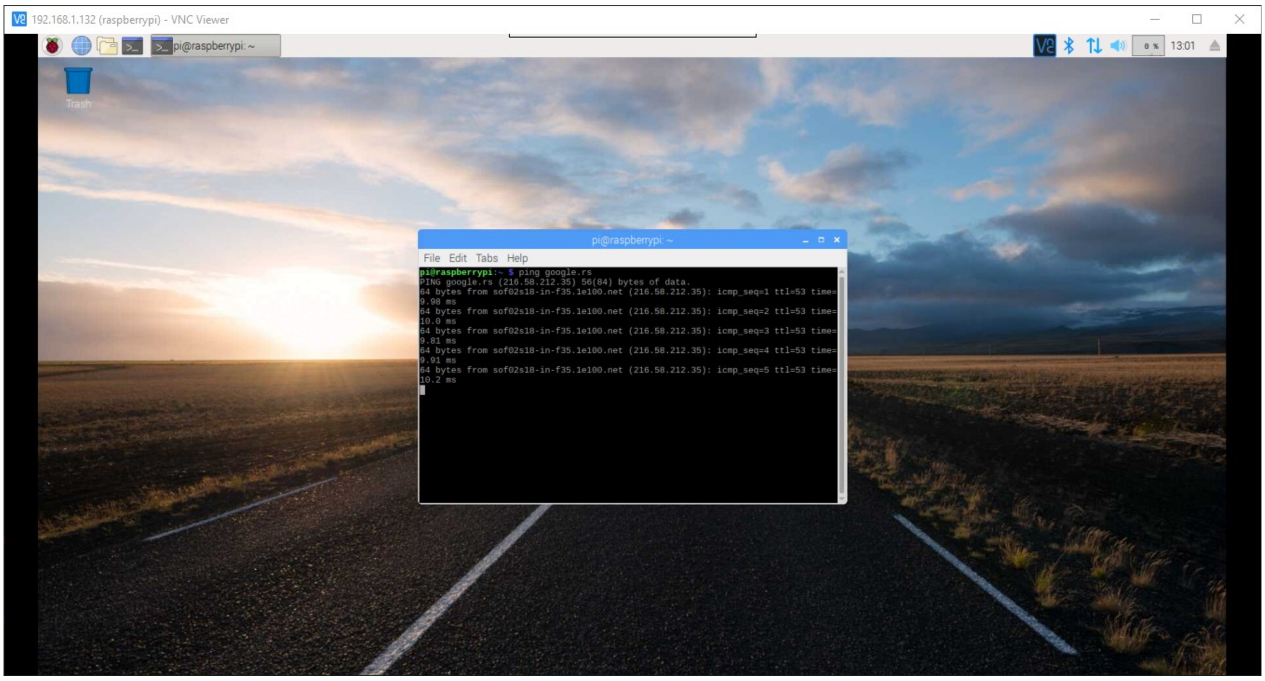
Enter VNC Server credentials
(Hint: NOT your RealVNC account details)

Username: pi
Password: ●●●●●●●●●●●●

Remember password   Forgot password?

Catchphrase: Jacket genetic vitamin. Survive Tokyo trust.
Signature: e3-e9-af-79-1e-30-a2-fa

And that is it. New window will open, and you are remotely logged in the Raspian operating system. Just for example, to show you that we can open apps in Raspbian, we run Terminal and write "ping google.com"

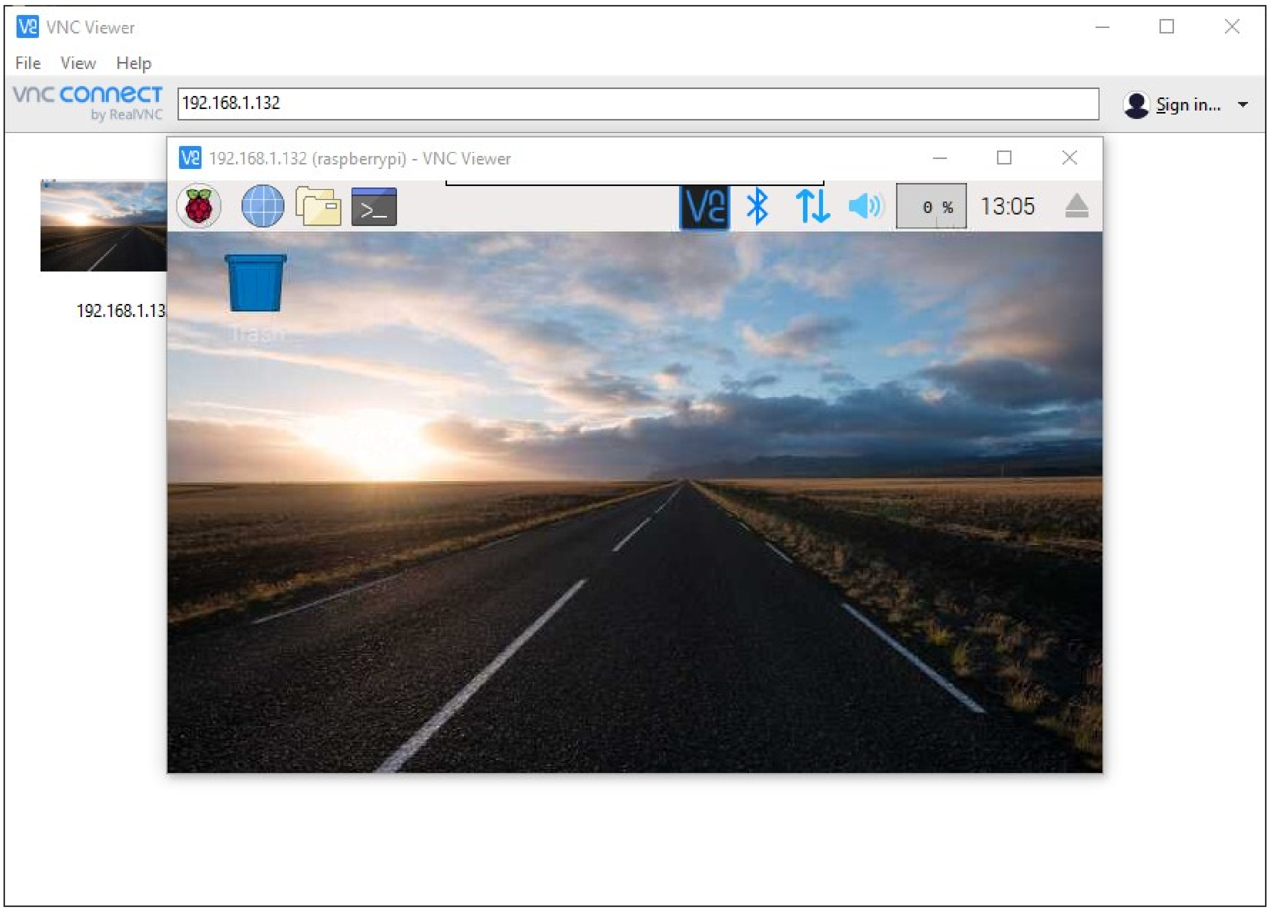


```
pi@raspberrypi:~ $ ping google.rs
PING google.rs (216.58.212.35) 56(84) bytes of data.
64 bytes from sof02s18-in-f35.1e100.net (216.58.212.35): icmp_seq=1 ttl=53 time=
9.98 ms
64 bytes from sof02s18-in-f35.1e100.net (216.58.212.35): icmp_seq=2 ttl=53 time=
10.0 ms
64 bytes from sof02s18-in-f35.1e100.net (216.58.212.35): icmp_seq=3 ttl=53 time=
9.81 ms
64 bytes from sof02s18-in-f35.1e100.net (216.58.212.35): icmp_seq=4 ttl=53 time=
9.91 ms
64 bytes from sof02s18-in-f35.1e100.net (216.58.212.35): icmp_seq=5 ttl=53 time=
10.2 ms
```

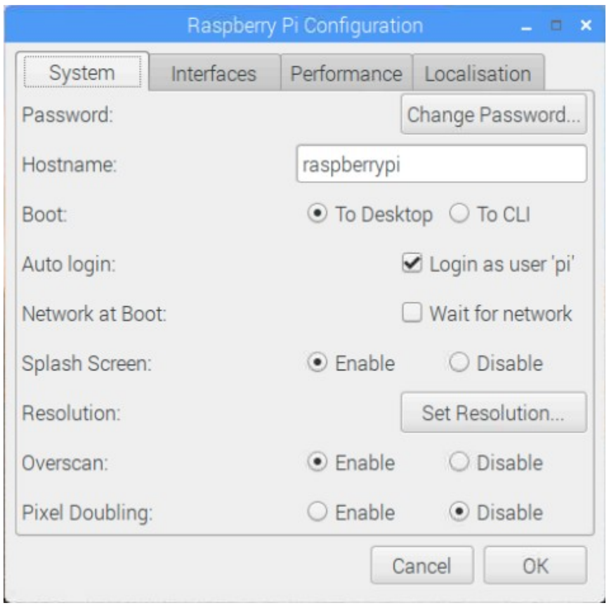
Now, we don't need screen, mouse and keyboard. We can connect to our Pi remotely.

1.2.3. Change the resolution of the screen

When you disconnect screen mouse and keyboard, and remotely connect to your Pi, resolution of displayed screen will be so small that you wouldn't be able to use Raspbian.

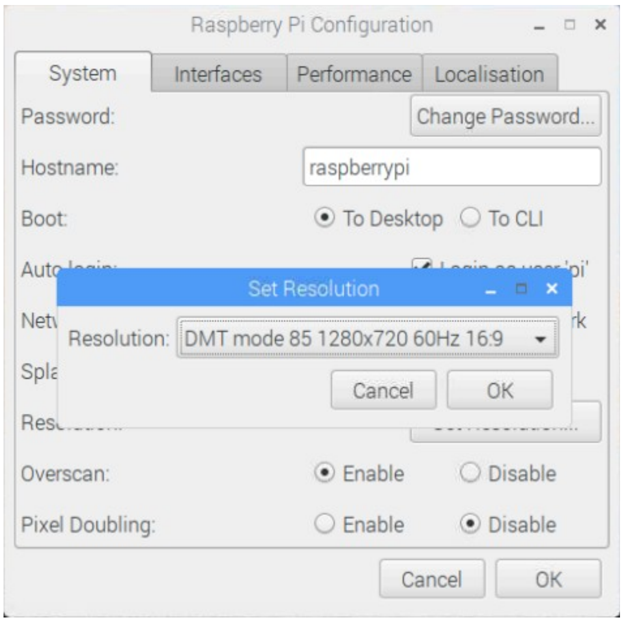


To change resolution go to *Start* > *Preferences* > *Raspberry Pi Configuration* and when new window opens,



Password:
Hostname: raspberrypi
Boot: [✓ To Desktop] To CLI
Auto login: [✓ Login as user 'pi']
Network at Boot: Wait for network
Splash Screen: [✓ Enable] Disable
Resolution:
Overscan: [✓ Enable] Disable
Pixel Doubling: Enable [✓ Disable]

click on button "*Set Resolution*". When new pop up window opens,



Resolution: DMT mode 85 1280x720 60Hz 16:9

In falling menus "*Resolution*" find one resolution you like and hit "*OK*". And that is it. You are ready to use your Pi for next steps.

1.3. Preinstalled apps on the Raspbian

Many apps are already included on the Raspbian. There is *LibreOffice* installed. This software is similar to the Microsoft Office. So we could use the Pi as a normal work computer for the office or school. *Word*, *Excel* and *PowerPoint*, *LibreOffice* calls these programs *Writer*, *Calc* and *Impress*. Furthermore, LibreOffice has programs for math (*Math*), drawing (*Draw*) and for databases (*Base*).

For surfing the internet the *Chromium* web browser is preinstalled, this is the Linux version of *Chrome* browser.

Also, some simple games are included. These games are programmed based on *Python*.

PDF viewer, calculator, text editor, image viewing software and compression program are included now on every PC and are also pre-installed on Raspbian.

Last but not least, there are programs for program development, which I would like to mention here: *Java*, *Python*, *SonicPi* and *Scratch*.

2. GPIO

The real peculiarity of the Raspberry Pi is neither its tiny size nor its price, but the huge fascination of the Raspberry Pi is based on the 40 GPIO pins that can be used to control electronic devices. With the Raspberry Pi, both electronic hobbyists and embedded Linux professionals have a toy or tool in their hands that makes the development of computer-controlled devices easier than ever before.

The board of the Raspberry Pi contains a connector strip with 2x20 contacts in one corner. The grid spacing is 2.54 mm. This connector strip is the basis for further projects, almost all hardware tinkering starts here. In addition to some general purpose input/output (GPIO) pins, the connector strip also provides two supply voltages (3.3V and 5V) and ground (0V).

Unfortunately, there are different designations of the pins, which cause a lot of confusion in practice. But we will come across this later.

The pins may be loaded together with a maximum of 50mA. The pins are protected by one self-resetting fuse (poly fuse). If too much current flows here, the Raspberry Pi switches itself off for a while. With a little luck there will be no permanent damage.

If we use GPIO pins for control (configuration as output), the voltage at the respective GPIO pin is 3.3 V. The control current per pin should not exceed 16mA or 50mA for all GPIOs. So we have to use suitable resistors!

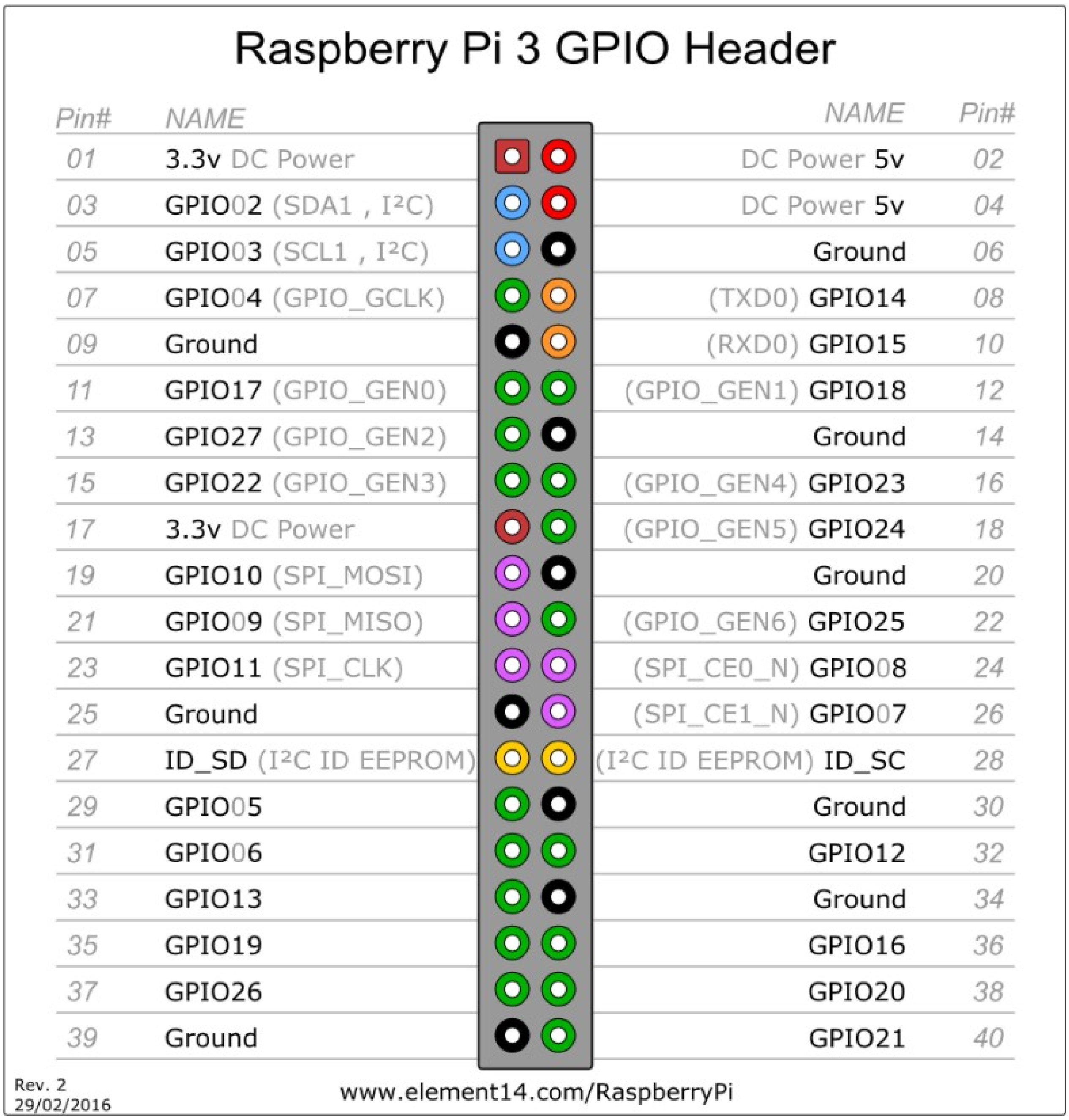
Clear information on the maximum permitted GPIO power is not given. Experiments of the Raspberry-Pi users have shown, that the Raspberry Pi is not damaged even at a slightly higher current and the output voltage drops accordingly, in order to limit the power. But we should keep the 50mA total current to avoid damage.

Many pins fulfill alternative functions depending on the programming. Before each project, we have to ask ourselves: which of the many GPIO pins do we use? As long as it's just a matter of doing little experiments and turning on and off a couple of LEDs, we can freely select the GPIO pins. Various special functions are available on selected pins and should we want to use them later, we should not use them yet until we run out of GPIO pins. As an example, pins 3 and 5 can also be used for the I2C bus.

WARNING

SAFETY INSTRUCTIONS
WHEN DEALING WITH THE GPIOs:
- Short circuits and incorrect wiring of pins can damage the Pi!
- Switch off the Pi if the circuit on the board is changed
- The max voltage on the GPIO pins is 3.3V.
- **NEVER CONNECT 5V TO GPIO!!!**

Here is an overview of all 40 GPIOs:



Raspberry Pi 3 GPIO Header

Pin#	NAME	NAME	Pin#
01	3.3v DC Power	DC Power 5v	02
03	GPIO02 (SDA1 , I²C)	DC Power 5v	04
05	GPIO03 (SCL1 , I²C)	Ground	06
07	GPIO04 (GPIO_GCLK)	(TXD0) GPIO14	08
09	Ground	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	Ground	14
15	GPIO22 (GPIO_GEN3)	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Ground	20
21	GPIO09 (SPI_MISO)	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	(SPI_CE0_N) GPIO08	24
25	Ground	(SPI_CE1_N) GPIO07	26
27	ID_SD (I²C ID EEPROM)	(I²C ID EEPROM) ID_SC	28
29	GPIO05	Ground	30
31	GPIO06	GPIO12	32
33	GPIO13	Ground	34
35	GPIO19	GPIO16	36
37	GPIO26	GPIO20	38
39	Ground	GPIO21	40

Rev. 2
29/02/2016

www.element14.com/RaspberryPi

2.1. GPIO as output

Let's start by connecting an LED to our Pi and programming Pi as well. The maximum current that the Pi can deliver is 50mA on all GPIO pins together!

Therefore, we use an LED that consumes little power. In order to calculate the required series resistance for our LED, we first have to look up the specifications of the LED in the datasheet. We use the red LED, Kingbright L-53LID. Next, let's search the datasheet on the internet for this LED. We need the voltage and the current this LED needs.

From the data sheet we can see:

I_F = 2mA, V_F = 2.0 ÷ 2.5V

With these values and Ohm's law, we can calculate our series resistance.

U = R * I   R = U / I   I = U / R

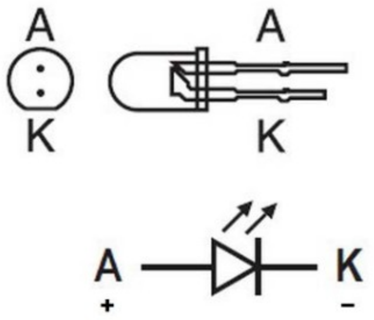
Our Pi delivers 3.3V, but our LED only needs 2V. Let's pull it off:

3,3V - 2V = 1,3V

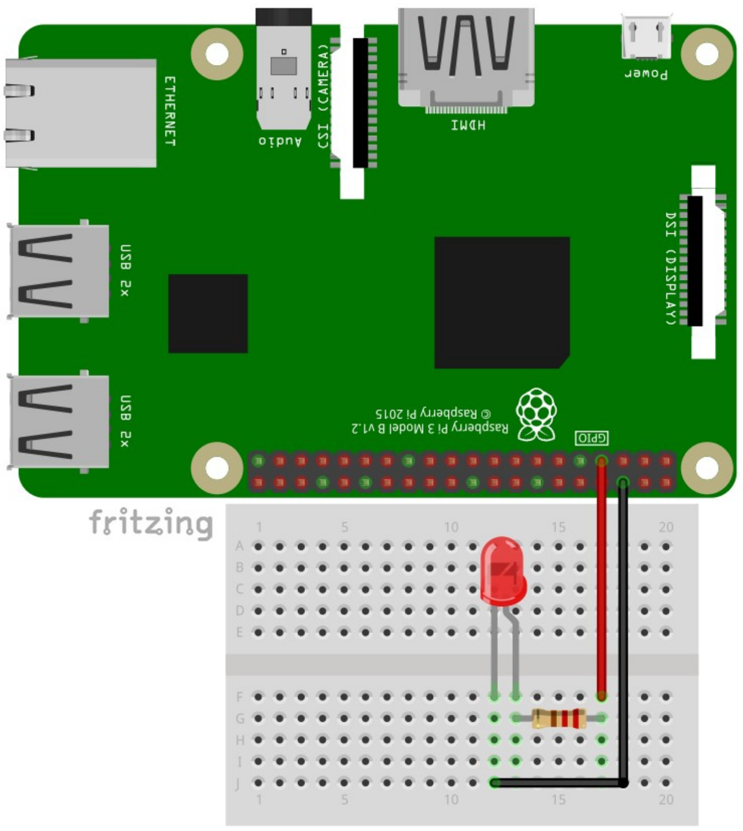
So our series resistor must consume 1.3V. The current is 2mA, let's put it in the formula:

R = 1.3V / 2mA = 1.3V / 0.002A = 650Ω

Thus we calculate resistace of the series resistor to be 650Ω. But this resistance will not exist and we choose the next higher value, that is 680Ω. The polarity of the LED has to be taken into account, here a brief explanation:



With 2 jumper wires we connect resistro, LED and breadboard with the Pi, as you can see on the picture below:



After the hardware has been wired up we start our Pi and logged in via *VNCViewer* again, and start the *Terminal*. Now we try the first easiest way to access the GPIO pin. We connected our LED to GPIO4. Then we enter the following commands into the console:

```
sudo echo "4" > /sys/class/gpio/export

sudo echo "out" > /sys/class/gpio/gpio4/direction

sudo chmod 666 /sys/class/gpio/gpio4/value

sudo chmod 666 /sys/class/gpio/gpio4/direction

echo "1" > /sys/class/gpio/gpio4/value
```

echo “4“ > xx	write a 4 in the file xx (initialize the GPIO 4)
echo “out“ > xx	write out in the file xx (define GPIO 4 as output)
chmod	change permissions
echo “1“ >	write a 1 in the file xx (turn GPIO 4 on)

With the last command (`echo "1" > /sys/class/gpio/gpio4/value`) we write in the file the output value of our GPIO4. If we write a 0, we turn off the pin.

```
echo "0" > /sys/class/gpio/gpio4/value
```

Note: After a restart of the Pi, the initialization must be carried out again!

In the next chapter, we will use another way of driving and bliking LED.

2.2. LED blinking light

So far we can only switch the LED on or off manually. But the LED does not blink yet. For the new method of accessing the GPIO, we will use *wiringPi*.

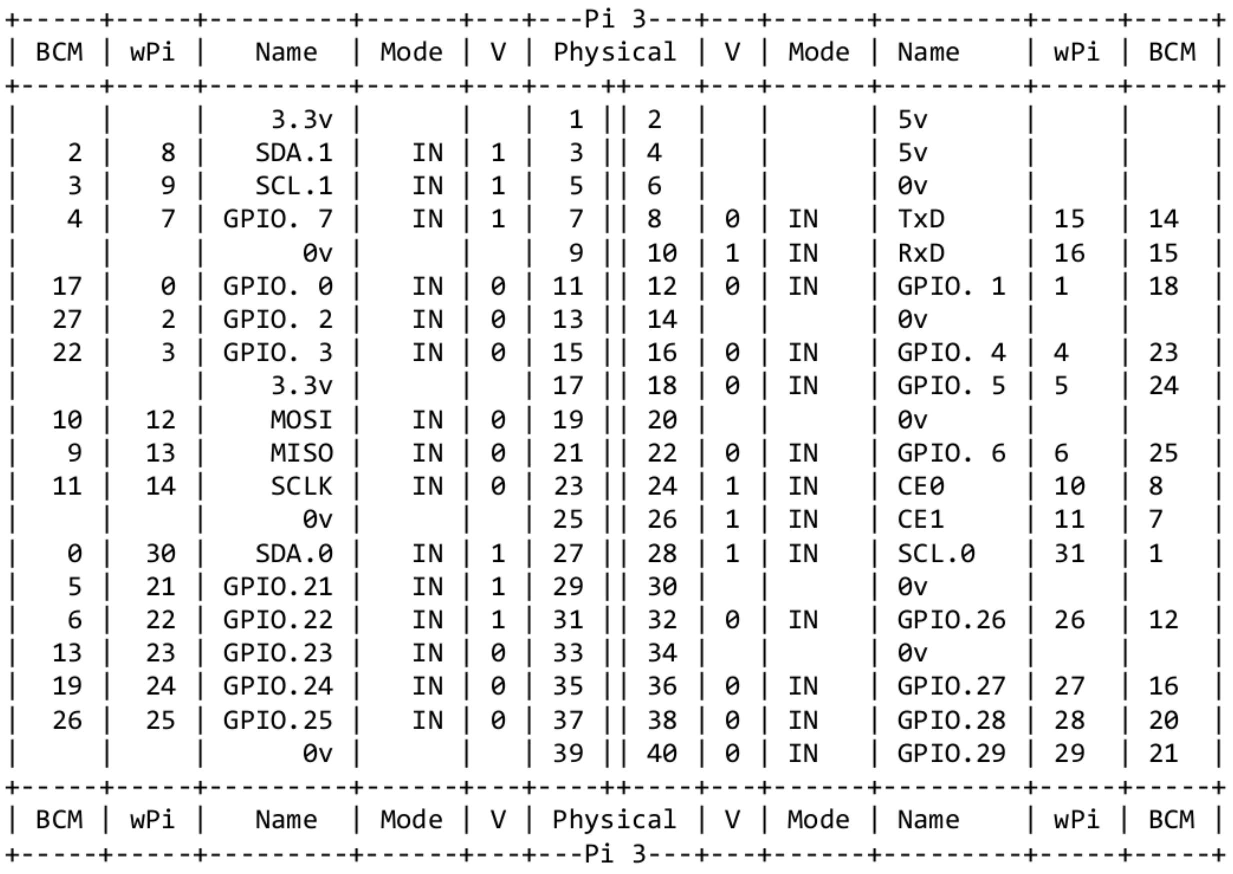
The *wiringPi* is a software that "simplifies" our control. To install of *wiringPi* we use command `gpio -v` and it does not output any information:

`gpio -v`	check version
`sudo apt-get purge wiringPi`	uinstalling old version
`sudo apt-get install git-core`	installing git
`git clone git://git.drogon.net/wiringPi`	download wiringPi
`cd ~/wiringPi`	change to the directory
`git pull origin`	check for current version
`cd ~/wiringPi`	change to the directory
`./build wiringPi`	compiling
`sudo ./build wiringPi`	install
`sudo reboot`	Pi restart

Now we give the command:

```
gpio readall
```

and we get the following output:



```
+-----+-----+---------+------+---+---Pi 3---+---+------+---------+-----+-----+
| BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
+-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
|     |     |    3.3v |      |   |  1 || 2  |   |      | 5v      |     |     |
|   2 |   8 |   SDA.1 |   IN | 1 |  3 || 4  |   |      | 5v      |     |     |
|   3 |   9 |   SCL.1 |   IN | 1 |  5 || 6  |   |      | 0v      |     |     |
|   4 |   7 | GPIO. 7 |   IN | 1 |  7 || 8  | 0 | IN   | TxD     | 15  | 14  |
|     |     |      0v |      |   |  9 || 10 | 1 | IN   | RxD     | 16  | 15  |
|  17 |   0 | GPIO. 0 |   IN | 0 | 11 || 12 | 0 | IN   | GPIO. 1 | 1   | 18  |
|  27 |   2 | GPIO. 2 |   IN | 0 | 13 || 14 |   |      | 0v      |     |     |
|  22 |   3 | GPIO. 3 |   IN | 0 | 15 || 16 | 0 | IN   | GPIO. 4 | 4   | 23  |
|     |     |    3.3v |      |   | 17 || 18 | 0 | IN   | GPIO. 5 | 5   | 24  |
|  10 |  12 |    MOSI |   IN | 0 | 19 || 20 |   |      | 0v      |     |     |
|   9 |  13 |    MISO |   IN | 0 | 21 || 22 | 0 | IN   | GPIO. 6 | 6   | 25  |
|  11 |  14 |    SCLK |   IN | 0 | 23 || 24 | 1 | IN   | CE0     | 10  | 8   |
|     |     |      0v |      |   | 25 || 26 | 1 | IN   | CE1     | 11  | 7   |
|   0 |  30 |   SDA.0 |   IN | 1 | 27 || 28 | 1 | IN   | SCL.0   | 31  | 1   |
|   5 |  21 | GPIO.21 |   IN | 1 | 29 || 30 |   |      | 0v      |     |     |
|   6 |  22 | GPIO.22 |   IN | 1 | 31 || 32 | 0 | IN   | GPIO.26 | 26  | 12  |
|  13 |  23 | GPIO.23 |   IN | 0 | 33 || 34 |   |      | 0v      |     |     |
|  19 |  24 | GPIO.24 |   IN | 0 | 35 || 36 | 0 | IN   | GPIO.27 | 27  | 16  |
|  26 |  25 | GPIO.25 |   IN | 0 | 37 || 38 | 0 | IN   | GPIO.28 | 28  | 20  |
|     |     |      0v |      |   | 39 || 40 | 0 | IN   | GPIO.29 | 29  | 21  |
+-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
| BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
+-----+-----+---------+------+---+---Pi 3---+---+------+---------+-----+-----+
```

All GPIO including the port numbers in *wiringPi*, differ to the GPIO names. Our GPIO4 will now be the port 7 in *wiringPi*.

It's time to write the first program.

```
touch blink.py             create a new file "blink.py"
nano blink.py              open file in editor
```

Then we insert the following code into the editor:

```
#!/usr/bin/python
from time import sleep
import RPi.GPIO as GPIO                  #  Integrate modules
GPIO.setmode(GPIO.BCM)                   #  Pin Description on BCM
GPIO.setup(4, GPIO.OUT)                  #  Set GPIO4 as output
while True:                              #  Generate Loop
   GPIO.output(4, GPIO.HIGH)             #  LED ON
   sleep(0.1)                            #  wait 0.1s
   GPIO.output(4, GPIO.LOW)              #  LED OFF
   sleep(0.1)                            #  wait 100ms
```

Execute the program with the command:

```
python blink.py
```

and that is it, LED is now blinking. The LED will blink until we kill the program with CTRL+C.

Let's try to connect a 2nd LED and make two LEDs to blink alternately.

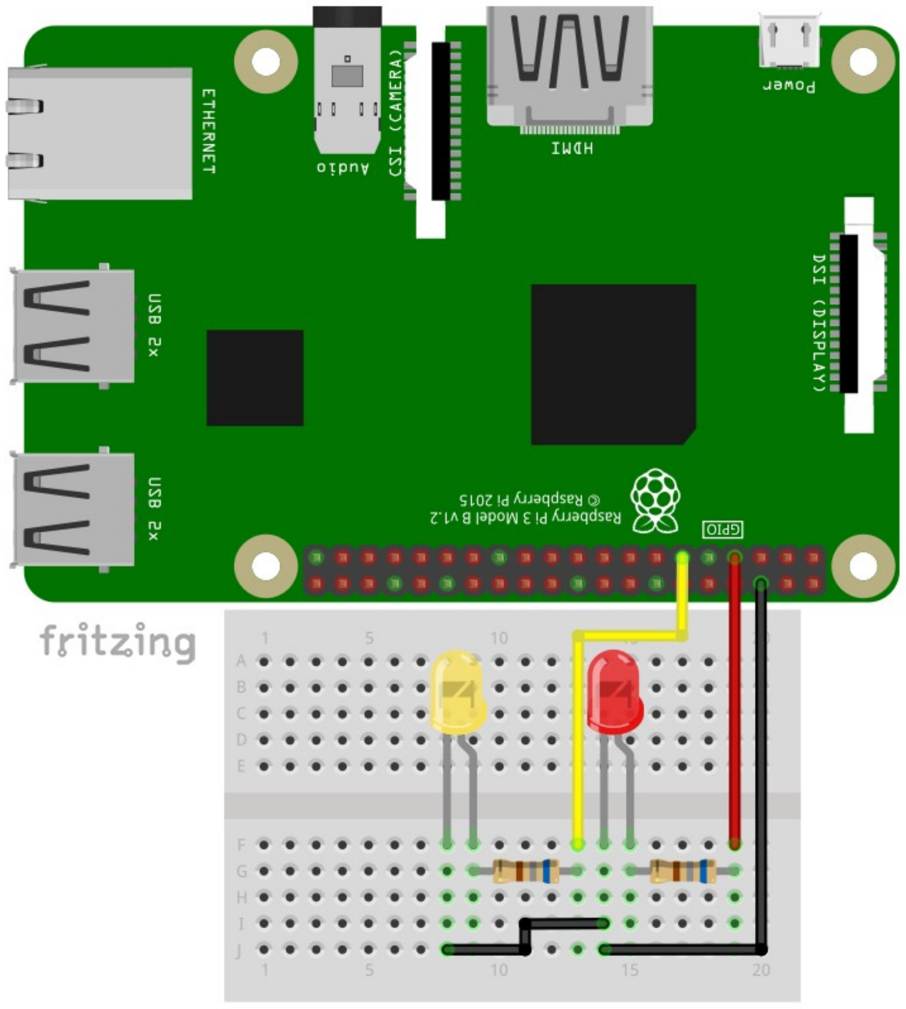
Let's use the yellow LED now. According to the datasheet of the Kingbright L-53LYD the LED has these values:

I_F = 2mA, V_F = 2.0 ÷ 2.5V

R = (3.3V – 2.1V) / 2mA = 1.2V / 0.002A = 600Ω

Also here we use a 680Ω resistor.

Let's connect our circuit now.



We also create a new file:

```
touch blink1.py
nano blink2.py
```

With the following content:

```
#!/usr/bin/python
from time import sleep
import RPi.GPIO as GPIO                #  Integrate modules
GPIO.setmode(GPIO.BCM)                 #  Pin Description on BCM
GPIO.setup(4, GPIO.OUT)                #  Set GPIO4 as output
GPIO.setup(17, GPIO.OUT)               #  Set GPIO17 as output
while True:                            #  Generate Loop
   GPIO.output(4, GPIO.HIGH)           #  LED ON
   GPIO.output(17, GPIO.LOW)           #  LED OFF
   sleep(0.1)                          #  wait 0.1s
   GPIO.output(4, GPIO.LOW)            #  LED OFF
   GPIO.output(17, GPIO.LOW)           #  LED ON
   sleep(0.1)                          #  wait 100ms
```

2.3. Traffic light

With three LEDs with the colors red, yellow and green we simulate a traffic light. We have the LEDs red and yellow already connected and calculated. The green LED is still missing. It has the following values:

I_F = 2mA, V_F = 2.0 ÷ 2.5V

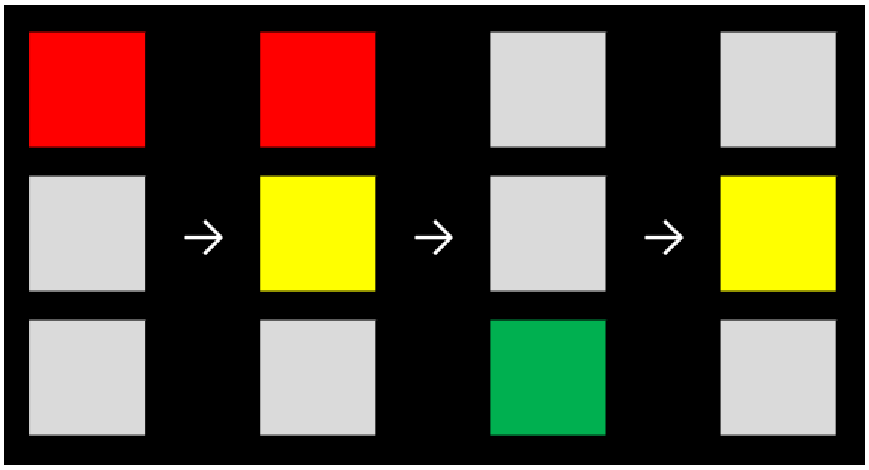
R = (3.3V – 2.2V) / 2mA = 1.1V / 0.002A = 550Ω

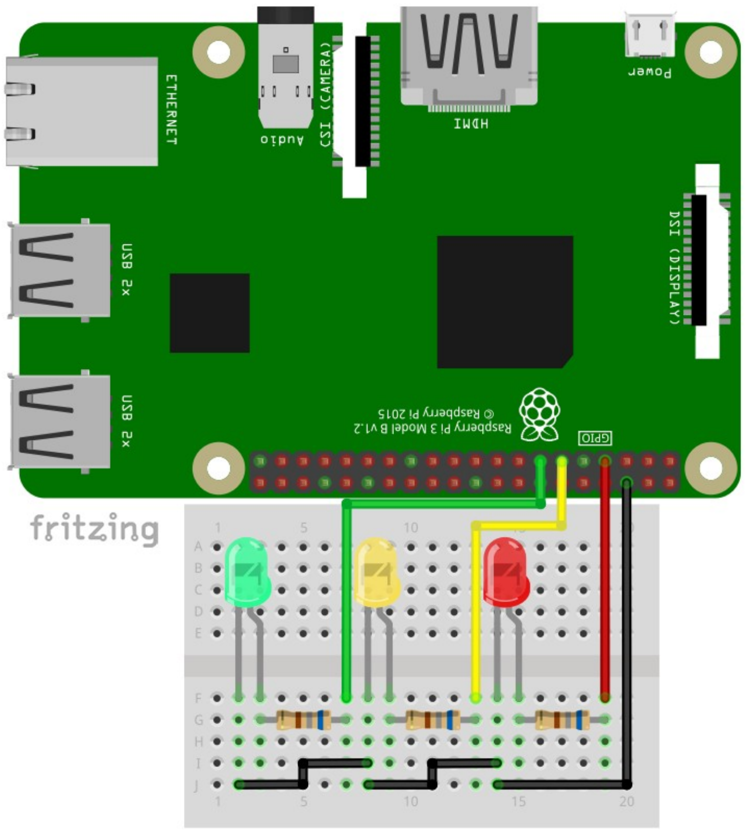
So with the green LED we need another resistor, namely 550Ω. We create a new program where we simulate a traffic light sequence:

```
touch traficLight.py
nano traficLight.py
```

and enter new script code:

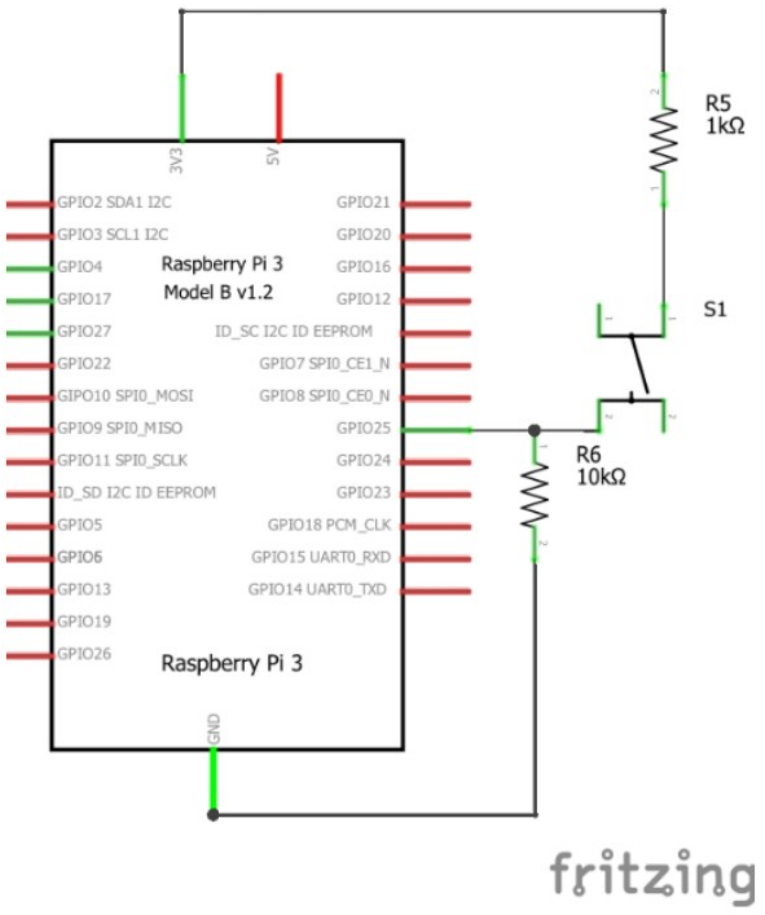
```
#!/usr/bin/python
from time import sleep
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.OUT)
GPIO.setup(17, GPIO.OUT)
GPIO.setup(27, GPIO.OUT)
while True:
    GPIO.output(4, GPIO.HIGH)
    GPIO.output(17, GPIO.LOW)
    GPIO.output(27, GPIO.LOW)
    sleep(10)
    GPIO.output(17, GPIO.HIGH)
    sleep(2)
    GPIO.output(4, GPIO.LOW)
    GPIO.output(17, GPIO.LOW)
    GPIO.output(27, GPIO.HIGH)
    sleep(10)
    GPIO.output(17, GPIO.HIGH)
    GPIO.output(27, GPIO.LOW)
    sleep(2)
```




ETHERNET
Audio
CSI (CAMERA)
HDMI
Power
DSI (DISPLAY)
USB 2x
USB 2x
Raspberry Pi 3 Model B v1.2
© Raspberry Pi 2015
GPIO
fritzing

2.4 . GPIO as input

The traffic light is located on a side of a street and does not always have to switch lights. In the street there is a sensor which detects cars, and only if a car wants to drive by the traffic light, the traffic light should change. So we will simulate the sensor as a button. For inputs you have to make sure that a GPIO pin does not float, which means that it remains unconnected. A floating connector can receive signals from the air (any radio signal) through a wire (which acts like antena) and the Pi may evaluate these signals as a push button operation. This leads to unwanted program behavior. Therefore we use the following circuit for our button:



The resistances before and after the button have a special meaning. The resistor with 10kΩ is a so-called pull-down resistor. If the button is not pressed, the ground is connected to the GPIO and pulls the voltage to 0V (ground). In contrast, there is also a pull-up resistor, this would be between GPIO and 3.3V supply voltage and this raises the voltage at the GPIO to the 3.3V.

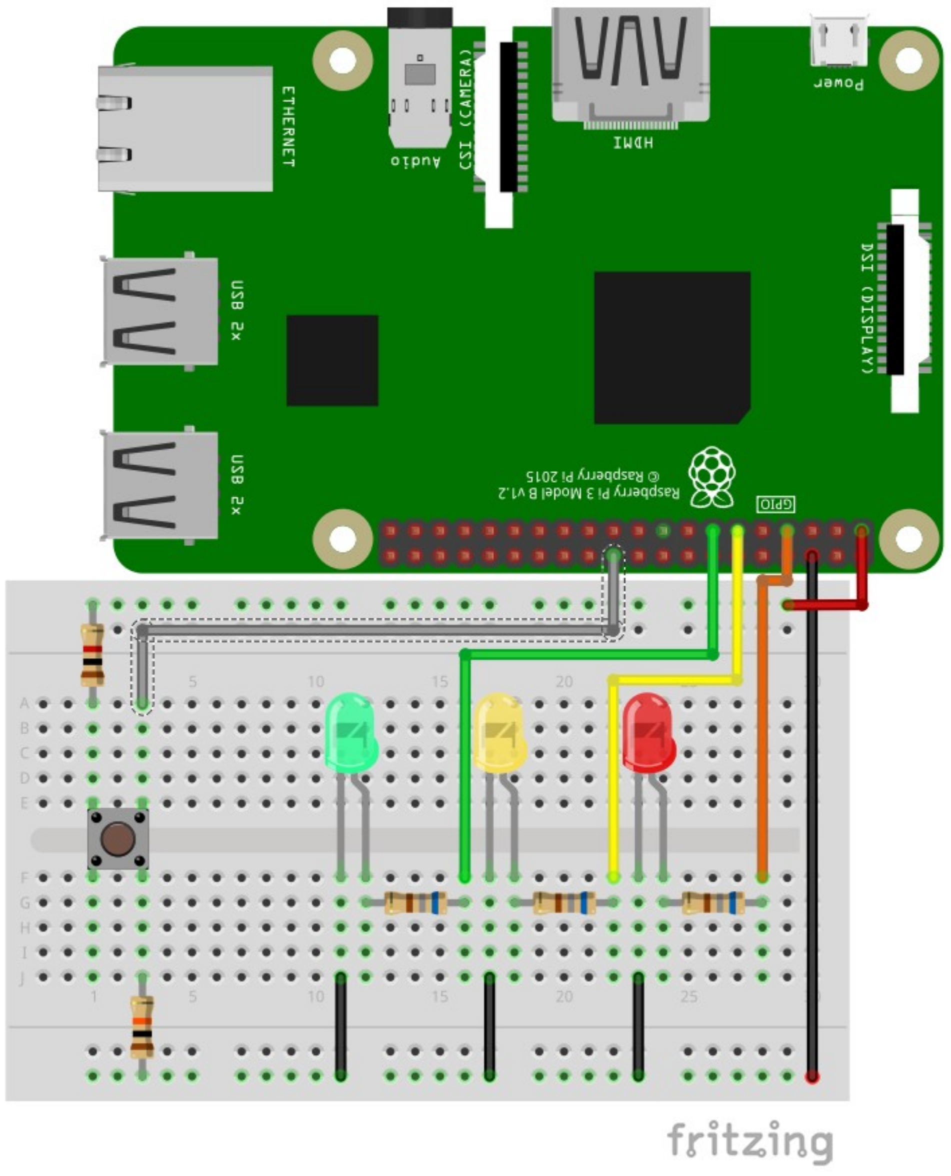
The second resistor is just a protective resistor for our Pi. As described above, the Pi delivers only 50mA and when more current flows the Poly-Fuse fuse trips. But in the worst case the Pi will be destroyed by a short circuit. This 1kΩ resistor is in series with the 10kΩ resistor when the button is closed. This results in a total resistance of 11kΩ between 3.3V and ground.

I = 3.3V / 11kΩ = 3.3V / 11000Ω = 0.0003A = 0.3mA = 300µA

So only a short circuit current flows from 300µA which is so low that nothing happens and our inputs still get a clean logical signal.

Input singnals on GPIO pins, the Pi evaluates logically. This means that from voltage of approx. 2V the input is recognized as HIGH (Logic 1). Voltages below 2V are evaluated as LOW (logic 0). This is important for just explained floating pin.

Let's build our circuit on the breadboard:



```
touch traficLight2.py
nano traficLight2.py
```

In this project we do not use the direct port assignment this time, but work with symbol variables. This means that we do not write GPIO 1 but assign the GPIO number to a variable. This makes it easier to change the pin assignment later for larger programs. With symbol variables we only have to change the pin once, e.g. ROT occurs in the following circuit a total of 5 times. These places would all have to be found and changed with pin change.

```
#!/usr/bin/python
from time import sleep
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)      # Disable warnings
RED = 4                      # GPIO Pin Assignment ROT = GPIO4
YELLOW = 17                  # GPIO Pin Assignment GELB = GPIO17
GREEN = 27                   # GPIO Pin Assignment GRUEN = GPIO27
TASTER = 25                  # GPIO Pin Assignment TASTER = GPIO25
GPIO.setup(RED, GPIO.OUT)
GPIO.setup(YELLOW, GPIO.OUT)
GPIO.setup(GREEN, GPIO.OUT)
GPIO.setup(TASTER, GPIO.IN)

def traficLightFunct():      # Traffic light function
    GPIO.output(RED, GPIO.HIGH)
    GPIO.output(YELLOW, GPIO.HIGH)
    GPIO.output(GREEN, GPIO.LOW)
    sleep(2)
    GPIO.output(RED, GPIO.LOW)
    GPIO.output(YELLOW, GPIO.LOW)
    GPIO.output(GREEN, GPIO.HIGH)
    sleep(10)
    GPIO.output(YELLOW, GPIO.HIGH)
    GPIO.output(GREEN,GPIO.LOW)
    sleep(3)
    GPIO.output(RED, GPIO.HIGH)
    GPIO.output(YELLOW, GPIO.LOW)

while True:                  # Loop
    GPIO.output(RED, GPIO.HIGH)
    GPIO.output(YELLOW, GPIO.LOW)
    GPIO.output(GREEN, GPIO.LOW)
    if GPIO.input(TASTER) == 1:
        traficLightFunct()
```

Grinding:

There are several ways to map recurring events in programming. Loops are used particularly frequently. Up to now we have always used the *while* loop in our program. With the *while* condition *True*, we create a infinite loop, because the *while* condition will never become *False*, because we cannot influence *True*.

Functions:

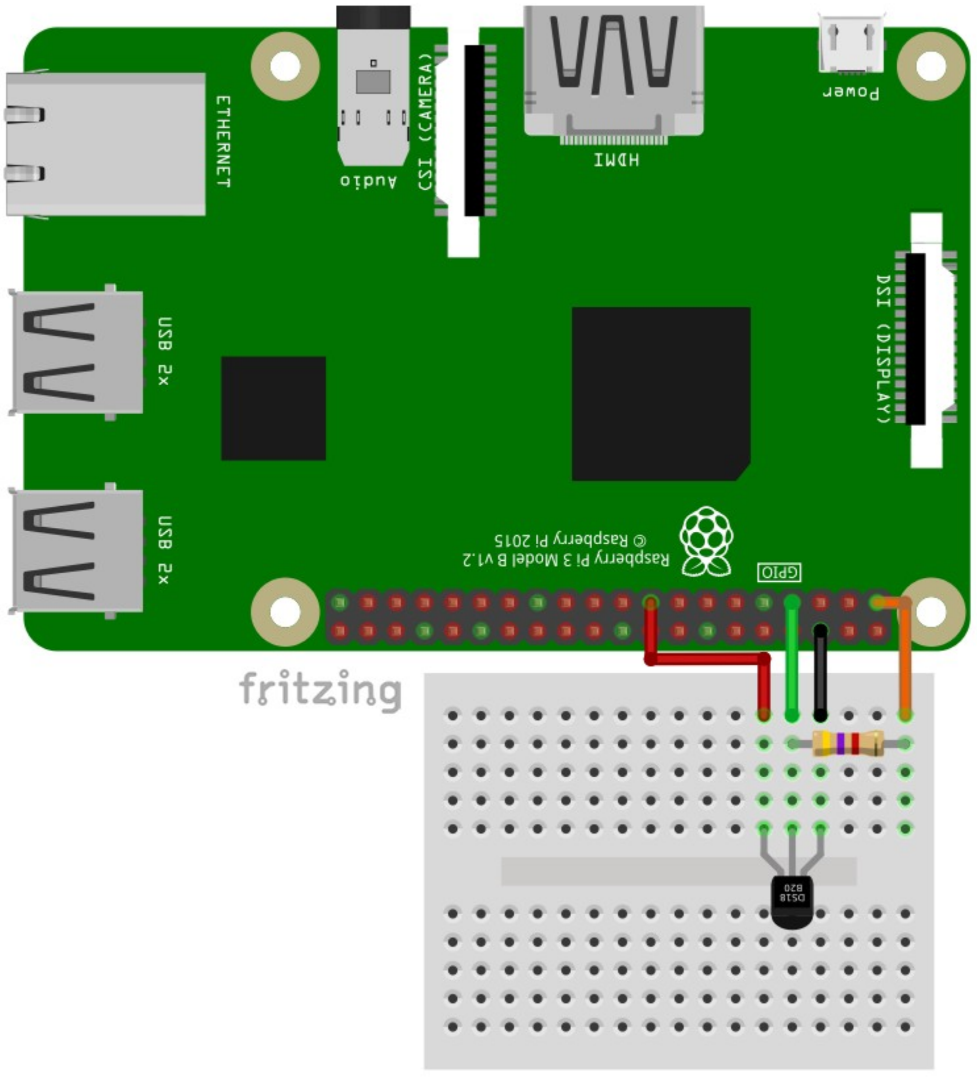
With functions you can make your program code clearer and store single recurring program parts to use less memory space. Functions can be called in a program as often as desired. Functions start with a reserved word "*def*" and a function name. The two brackets after the name are empty in our example, but here you could pass values "*arguments*" to the function that can be processed in the function.

Conditions, "*if*":

With the so-called *if-then-else* condition, we can program branches in a program code. We do not want the traffic light to start running until the button has been pressed. "If" the button is pressed, "*then*" the traffic light will start, do nothing else.

2.5. Temperature sensor

Now that we had lit a few LEDs, let's read the temperature. For this we use the temperature sensor DALLAS DS18B20. It is controlled by one wire. The one wire protocol is supported on the Pi only on the GPIO4. Therefore, the one wire connection of the temperature sensor (middle connection) must be connected here. Let's make the circuit:



When everything is properly wired, you can we activate the one wire protocol with it, write in Terminal next commands:

```
sudo modprobe w1-gpio
sudo modprobe w1-therm
```

If it worked, we can find out, by entering the following:

```
lsmod
```

The modules should now be listed, if it is not listed, wrong GPIO pin is used or an error occurred during activation.

So these modules are not loaded at each start, we need enter commands for them into the file */etc/modules*, write this command in Terminal:

```
sudo nano /etc/modules
```

and insert the following two lines at the end:

```
w1_gpio
w1_therm
```

Furthermore you have to reserve the GPIO4 for one wire in the boot.txt:

```
sudo nano /boot/config.txt
```

and add the following line at the end:

```
dtoverlay=w1-gpio,gpiopin=4
```

After a restart, we change to the *W1-Bus* directory and have all sensors displayed. For that write this lines in Terminal:

```
cd /sys/bus/w1/devices/
ls
```

The output should contain something like this: **28-02162eb1fbee**

This number should be remembered, this is the ID of the temperature sensor. Next we read the sensor, we write this command to the Terminal:

```
cat /sys/bus/w1/devices/28-02162eb1fbee/w1_slave
```

In response, we get this:

```
b8 01 4b 46 7f ff 0c 10 b1 : crc=b1 YES
b8 01 4b 46 7f ff 0c 10 b1 t=27500
```

in the second line t=27500 is our temperature value in milli degrees. This value only has to be divided by 1000, 27000 / 1000 = 27.5. The measured temperature is 27.5 °C.

Since this displaying and converting is cumbersome, we will write a small bash script and name it "*temperature*", and later in Terminal we justneed to write:

```
temperature
```

to display temperature measured by the sensor.

```
sudo nano /usr/bin/temperature
```

and enter this:

```
#! /bin/bash
# Read temperature
tempread=`cat /sys/bus/w1/devices/28-02162eb1fbee/w1_slave`
# Value Format
temp=`echo "scale=2; "\`echo ${tempread##*=}\`" / 1000" | bc`
# Output
echo "The measured temperature is" $temp "°C"
```

Now assign the appropriate rights:

```
sudo chmod +x /usr/bin/temperature
```

And if you now enter in the Terminal:

```
temperature
```

the currently measured temperature appears.

If you get an error message, you have to install *bc*, and to install it write this command in Terminal:

```
sudo apt-get install bc
```

If everything worked fine the following result should be shown:

```
The measured temperature is 27.5 °C.
```

You've done it, you can now use your module for your projects.

Now it is time to learn and make the Projects on your own. You can do that with the help of many example scripts and other tutorials, which you can find on the internet.

If you are looking for the high quality products for Arduino and Raspberry Pi, AZ-Delivery Vertriebs GmbH is the right company to get them from. You will be provided with numerous application examples, full installation guides, eBooks, libraries and assistance from our technical experts.