Raspberry Pi: Prepare SD Card

***Note**: The term SD card mentioned here generally covers/means the microSD card.*

This is a guide to prepare such card from scratch. If you already have the one that comes along with the board (NOOBS card, is it?), this procedure will override and delete existing content. Backup or forever hold your peace! Well, if you cannot do that and you want to have NOOBS back, head on to this page and download the latest NOOBS image.

Bare-metal Programming

***Note:** Your instructor will show you how to partition/format the SD card.*

To exclusively work on bare-metal programming (no OS):

- Format the card with at least one FAT32 partition
  - Linux platform - use *fdisk* for this (usually need root... unless your user have permission)
  - Windows platform - use *diskpart* (run as administrator)... OR, maybe *this*? (The link is for portable version.. there is also an installer if that is your cup of coffee 😎)
- Get the required firmware from here - we only need 2 files from boot folder (bootcode.bin and start.elf). For the sake of being 'politically correct', I also copied LICENSE.broadcom file. OR, get my personal copy here (which is what I usually pass to my students on a USB drive).
- Place those 3 files in the root of the previously FAT32-formatted SD card
- Along with your kernel.img code, you should be able to take control of the Pi

A nice config for multiple kernels - this is what I use to have multiple kernel (my1barepi codes) images on my SD card

config-my1barepi.txt

```
# kernel is the alternative kernel filename
# - [Pi 1, Pi Zero, and Compute Module] kernel.img
# - [Pi 2, Pi 3, and Compute Module 3] kernel7.img
# - [Pi4] kernel7l.img.
#kernel=kernel_video_temp.img
#kernel=kernel_sdcard.img
kernel=kernel_pick_one.img
```

Installing Raspbian OS

***Update20210907** The official OS is now known as Raspberry Pi OS (instead of Raspbian). Check out this official page. I will try to update this site A.S.A.P. - but, do not hold your breath* 😬

We will be using Raspbian (the official Linux distribution for Raspberry Pi). This enables us to run web servers and other network-related stuffs.

[201804011654] *Note: I just noticed there is now an option to use Windows10 IOT Core (which is prepared by Microsoft as a third party option), but I will not be using that here. Checkout this page for other options*

- Download system image
  - Official Raspbian (latest)
    - This is a ZIP file containing an image file - e.g. 2018-11-13-raspbian-stretch.zip
  - For PGT302 students, get one with PGT302-specific customization
    - Ask your instructor
- Extract (unzip) the image (file with *.img extension)
  - e.g. 2018-03-13-raspbian-stretch.img
  - currently, at least 8GB SD card is required...
- Write the image to SD card
  - Insert the SD card to your SD card reader
  - Linux Platform - use *dd* for this
    - assuming the device is at /dev/sdb
    - use `dd if=2018-03-13-raspbian-stretch.img of=/dev/sdb` and wait...
  - Windows Platform - I recommend Win32DiskImager (I got the Win32DiskImager-1.0.0-binary.zip file)
    - find (and VERIFY) drive letter for your SD card
    - select image, start write and wait...

Raspbian-ready Plus Bare-metal Programming

To have Raspbian-ready card as well as working on bare-metal programming:

- Follow normal instruction for installing Raspbian OS

Note: *Do the following procedure on your PC.*

To use bare-metal code:

- Rename kernel.img and config.txt to avoid being used
  - e.g. rename kernel.img → kernel-raspbian.img
  - e.g. rename config.txt → config-raspbian.txt
- Make sure there is no config.tx file
- Simply copy your own compiled kernel.img to the FAT32 partition

To get Raspbian back running:

- Get the saved files appropriately named:
  - e.g. copy kernel-raspbian.img → kernel.img
  - e.g. copy config-raspbian.txt → config.txt

Raspberry Pi Zero as USB Client

Pi Zero has a USB On-the-Go (OTG) hub - which, basically means that it can be both host (like USB hub on a PC) AND client (like USB hub on Android or most gadgets these days). So, to setup Pi Zero as a client (this is done on a PC - while preparing the card),

1. Follow normal instruction for installing Raspbian OS
2. Edit `config.txt` (on boot partition) and insert `dtoverlay=dwc2` line
3. Edit `cmdline.txt` and insert `modules-load=dwc2,g_ether` kernel parameter
    - also, insert `g_ether.host_addr=<mac_addr>` to get a fixed MAC address (easier to manage!)
4. Add empty file called `ssh` (same location as `config.txt`) - this will enable ssh

Connect Pi Zero to a PC (allow some time for it to finish booting) and it should appear as ethernet device. To connect to it,

- using network manager:
    - config ipv4 as link-local only
    - config ipv6 as ignore
- use ssh to connect
    - ssh pi@raspberrypi.local
        - default password: raspberry
    - use ssh-copy-id to use key-based auth

To share the internet with the Pi

- find the IP address on usb0

```
$ ifconfig usb0
```

    - we need this later on Pi
- allow IP forwarding on the host

```
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

    - to remove this later

```
echo 0 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -D POSTROUTING -o eth0 -j MASQUERADE
```

    - *note for wifi internet, change `eth0` to `wlan0`*
- ssh into pi
- (OPTIONAL) add a name server (e.g. nameserver <ip-add-of-usb0>) to /etc/resolv.conf
    - or, just use 8.8.8.8
- set the default gateway on Pi to host PC's IP

```
# route add default gw <ip-add-of-usb0>
```

Have fun!

Note:*I have tested this using Raspbian Lite on Pi Zero and it worked! (Obviously)*

Advanced Configurations

Using the above setup should get simple projects going without any problems. However, some things on the BCM2835 require special configurations on the GPU side. This can be changed by having a `config.txt` file in the same path as the files above.

Example of the configuration file:

config.txt

```
# --------------
# MEMORY OPTIONS
# --------------

# specify gpu memory allocation
# - min 16, max 192 (256), 448(512), 944 (1024)
# - default 64
#gpu_mem=64

# disables CPU access to GPU L2 cache
# - default 0 (enabled)
#disable_l2cache=1

# --------------
# KERNEL OPTIONS
# --------------

# specify kernel name
# - kernel7.img default for pi2/pi3
# - kernel8.img preferred on pi3 (for 64b mode)
# - common default is kernel.img?
#kernel=kernel.img

# specify startup address for ARM kernel
# - default 32b: 0x8000
# - default 64b: 0x80000
# - kernel_old=1 option overrides to 0?
#kernel_address=0x8000
#kernel_old=1

# ----------------
# ADVANCED OPTIONS
# ----------------

# camera needs start_x.elf firmware
# - or, start_file=start_x.elf, fixup_file=fixup_x.dat
#start_x=1
```

```
# prevent red camera led to turn on while camera is active
# - default 0 (enabled)
#disable_camera_led=1
```

To activate an option, simply remove the '#' character for the beginning of the option line (uncomment). More information on `config.txt` can be found here. We can also have conditional filters.

Other custom configuration(s):

config.txt

```
# for 5-inch lcd with touchscreen
max_usb_current=1
hdmi_group=2
hdmi_mode=87
hdmi_cvt 800 480 60 6 0 0 0
dtoverlay=ads7846,cs=1,penirq=25,penirq_pull=2,speed=50000,keep_vref_on
=0,swapxy=0,pmax=255,xohms=150,xmin=200,xmax=3900,ymin=200,ymax=3900
#display_rotate=0
```

General Issues

MicroSD Card Failed Boot

Sometimes, the card simply cannot boot. Use fdisk to check/create 255 heads, 63 sectors and calculate the required cylinders based on

```
disk_size = cylinders * head * sector * sector_size
```

Not sure why this happens... maybe BIOS issue when formatting on PC? Or, maybe my students' laptops are infected with virus?

Raspbian Update Error

- I used the 2017-09-07-raspbian-stretch.img and got an error while trying to update
  - problems seem to be with storage space (98% usage)
  - turns out the partition for root fs was only 4.9GB
- to resize the partition, use fidsk
  - assume card is /dev/sdb (i use USB card reader)
  - `fdisk /dev/sdb`
  - delete partition 2 and recreate (make sure use the same start sector!)
  - save and exit
  - clean the fs `e2fsck -f /dev/sdb2`
  - resize `resize2fs /dev/sdb2`

2026/02/09 22:23

Notes for config.txt

Dumped...

config.txt

```
# hw switch for shutdown
dtoverlay=gpio-shutdown,gpio_pin=21,active_low=1,gpio_pull=up
```

2026/02/09 22:23

Raspberry Pi: Prepare Raspberry Pi OS (Lite)

Dumped...

```
- write image (bullseye) to sd card, boot
- change password
$ sudo passwd pi
- change hostname
$ sudo raspi-config nonint do_hostname <hostname>
- change locale
$ sudo raspi-config nonint do_change_locale en_US.UTF-8
- enable ssh
$ sudo raspi-config nonint do_ssh 0
- change timezone
$ sudo timedatectl set-timezone Asia/Kuala_Lumpur
- change keyboard map
$ sed -i /etc/default/keyboard -e "s/^XKBMODEL.*/XKBMODEL=\"pc105\"/" \
    -e "s/^XKBLAYOUT.*/XKBLAYOUT=\"us\"/" \
    -e "s/^XKBVARIANT.*/XKBVARIANT=\"\"/" \
    -e "s/^XKBOPTIONS.*/XKBOPTIONS=\"\"/"

- (optional) if not using wifi (e.g. on pi 1), disable wpa_supplicant
$ sudo systemctl disable wpa_supplicant
- (optional) change wifi country
$ sudo raspi-config nonint do_wifi_country MY
- (optional) disable camera
$ sudo raspi-config nonint do_camera 1
- (optinal) running lite most probably do not need much graphics!
$ sudo raspi-config nonint get_config_var gpu_mem /boot/config.txt
$ sudo raspi-config nonint do_memory_split 32

- update pkg repo
$ sudo apt update
```

```
- install dev tools
$ sudo apt install git raspberrypi-kernel-headers screen vim

- work stuff
$ mkdir $HOME/work
$ cd $HOME/work
$ git clone https://codeberg.org/azman/my1shell
$ git clone https://codeberg.org/azman/my1linuxpi
$ git clone https://codeberg.org/azman/my1codelib
$ git clone https://codeberg.org/azman/my1apisrvc
$ mkdir $HOME/.local
$ cd $HOME/.local
$ ln -sf $HOME/work/my1shell
$ cd my1shell
$ ./bash-setup --profile --write

- install minimal gui
$ sudo apt install --no-install-recommends xserver-xorg xinit
- install pi desktop
$ sudo apt install raspberrypi-ui-mods
- customize ''/etc/xdg/user-dirs.defaults''
- reboot to desktop
```

Another try... **incomplete**

```
[]
- write image (bullseye) to sd card, boot
  = auto-resize, generate ssh, reboot
  = change to english (us) keyboard
  = setup username/password
- change hostname
$ sudo raspi-config nonint do_hostname <hostname>
  = reboot, just in case
- change locale
$ sudo raspi-config nonint do_change_locale en_US.UTF-8
- change timezone
$ sudo timedatectl set-timezone Asia/Kuala_Lumpur
- enable ssh
$ sudo raspi-config nonint do_ssh 0
  > perl: warning: Setting locale failed.
```

2026/02/09 22:23

Raspberry Pi: Prepare Raspbian

***Note**: This assumes the previous how-to on preparing the SD card has been covered and Raspbian is already 'installed' on the SD card.*

Booting Raspbian

- Simply insert the SD card and power up
  - It will take some time, but a Linux Desktop will appear on screen.
- Initial Configuration:
  - locale set to US
  - timezone changed to Asia/Kuala_Lumpur
  - keyboard layout set to US (English)
- Run update `sudo apt update` and upgrade '' sudo apt upgrade'' if necessary
  - if error occurs, most probably problem with the mirror (change to another mirror)

Getting the tools

- Get kernel headers (compiling loadable kernel module)
  - *sudo apt install* `raspberrypi-kernel-headers`
- To be able to compile codes that require sqlite
  - *sudo apt install* `sqlite3 libsqlite3-dev`
- To have web server with php
  - *sudo apt install* `apache2 php php-cgi libapache2-mod-php php-sqlite3`
  - edit `php.ini` to enable pdo support
  - enable mod_rewrite (create link in mods-enabled from mods_available)
    - look for `rewrite`
  - default path for web is `/var/www/html`
- I also want `screen` and `ntfs-3g` (ntfs with r/w)
  - *sudo apt install* `screen ntfs-3g`

Prepare for GUI Development

This is about doing GUI development while running Raspbian on Pi itself.

- install glade (will also get gtk library)
  - *sudo apt install glade*

Using Camera & Image Processing

- to access camera, load the Video4Linux kernel module
  - *sudo modprobe bcm2835-v4l2*
  - not needed if you enabled camera using Configuration Page
- install ffmpeg stuffs
  - *sudo apt install libavcodec-dev libavdevice-dev*
- if going for GTK
  - *sudo apt install libgtk2.0-dev*
- if going for SDL
  - *sudo apt install libsdl1.2-dev*

Get some codes

- Get my source codes (e.g. download everything @/home/pi/Work)

```
mkdir -pv /home/pi/Work && cd /home/pi/Work
git clone git://github.com/azman/my1codelib.git
git clone git://github.com/azman/my1webapp.git
git clone git://github.com/azman/my1apisrv.git
git clone git://github.com/azman/my1linuxpi.git
git clone git://github.com/azman/my1termu.git
git clone git://github.com/my1matrix/my1goose.git
```

- Obviously, we need `git` for this
  - *`sudo apt install`* `git`

Upgrading Installed Packages

To upgrade locally installed sofware:

- Update apt database

```
$ sudo apt update
```

- Upgrade local packages

```
$ sudo apt upgrade
```

If packages got held/kept back, simply reinstall those packages:

- Reinstall

```
$ sudo apt install --reinstall
```

Upgrading Raspbian Version

To do an upgrade (e.g. I did this to upgrade stretch to buster):

- Modify `/etc/apt/sources.list` and replace the release codenames (e.g. change stretch to buster)
  - do the same for `/etc/apt/sources.list.d/*.list`
- Update package list

```
$ sudo apt update
```

- Upgrade distribution

```
$ sudo apt dist-upgrade
```

- Do house cleaning

```
$ sudo apt autoremove
$ sudo apt clean
```

That should do it!

Making Pi Visible on Local Network

Note: *This is already in the default install.*

To make Pi hostname visible on local network, get avahi-daemon (default install)

- *sudo apt install* avahi-daemon
- this is part of Bonjour/Rendezvous/ZeroConf multicast DNS (aka mDNS) services
- change hostname as desired
  - use raspi-configuration tool
- clients need to support mDNS as well:
  - windows need this
  - linux require avahi (for slackers, slackbuilds.org got this)

2026/02/09 22:23

Raspberry Pi: Prepare PiCore

Note*: This is an alternative to using Raspbian. Although not necessary, reading how-to on preparing the SD card is recommended.*

Preparing SD Card with PiCore

work in progress...

```
- download image from http://distro.ibiblio.org/tinycorelinux/ports.html
- select latest stable image (in zip format)
- unzip and dd to sdcard
- use fdisk to resize second partition (linux)
  = e.g. fdisk /dev/sde
  = watch the start block!
- use resize2fs to resize to full capacity
  = e.g. resize2fs /dev/sde2
- reboot and do stuffs
  = [INIT] filetool.sh -b
  = [BASE] tce-load -wi TC
  = [MINE] tce-load -wi compiletc git geany
```

```
fdisk -lu /path/disk.img

-> in sector counts (usually 512-bytes sectors)

mount -o loop,offset=xxxx /path/disk.img /mnt/disk.img.partition

-> offset in bytes!

picore requires:
> tce-load -wi i2c-tools
> sudo modprobe i2c-dev

i2cdetect -y 1
```

2026/02/09 22:23

Raspberry Pi: System Setup

dumped for now...

pi3_tablet.txt

```
pi3_tablet:
- pi3 with 7in display
(https://www.raspberrypi.com/products/raspberry-pi-touch-display/) &
enclosure
INSTALL:202408181341
- using raspios-bookworm-arm64 image (with desktop)
- first boot
  = setup user => common!
  = skip update - do that later
- reboot
  = display is inverted (Menu>Preferences>Screen Configuration)
    > right click DSI-1
  = update system (if applicable)
  = customize system (Menu>Preferences>Raspberry Pi Configuration)
    > set hostname
    > enable ssh & i2c
- reboot
  = install: vim geany screen
  = remove: nano
  = touch is independent of display setting (inverted)
    > edit /boot/firmware/config.txt (insert:dtoverlay=vc4-kms-
dsi-7inch,invx,invy)
    > also need this (dtparam=i2c_vc_baudrate=50000)
- get my usual stuff from codeberg
  = my1shell, my1linuxpi, my1codelib, my1codebase
  = my1imgpro, my1imgproX
```

pi0_config.txt

```
- Raspberry Pi OS lite
  = for Pi Zero (bookworm armhf lite)
- write image to sd card, boot
  = english (us) keyboard layout
  = setup username/password
- change hostname (default is raspberrypi)
$ sudo raspi-config nonint do_hostname <hostname>
- change locale (default is en_GB.UTF-8)
$ sudo raspi-config nonint do_change_locale en_US.UTF-8
  = reboot, make sure locale properly set up
- enable ssh
$ sudo raspi-config nonint do_ssh 0
- change timezone
$ sudo timedatectl set-timezone Asia/Kuala_Lumpur

- (optional) if not using wifi (e.g. on pi 1), disable wpa_supplicant
$ sudo systemctl disable wpa_supplicant
- (optional) change wifi country
$ sudo raspi-config nonint do_wifi_country MY
- (optional) disable camera
$ sudo raspi-config nonint do_camera 1
- (optional) running lite most probably do not need much graphics!
$ sudo raspi-config nonint get_config_var gpu_mem /boot/config.txt
$ sudo raspi-config nonint do_memory_split 32
- (optional) run update
$ sudo apt update
$ sudo apt upgrade

- install/remove stuff
$ sudo apt install vim git screen
$ sudo apt remove nano

- get my stuff (git): my1shell, my1linuxpi, my1codelib, my1codebase
```

2026/02/09 22:23

Raspberry Pi: Build Options

Some notes on build/compiler stuffs.

BCM2835

The gcc compiler flags which will produce the most optimal code for the BCM2835 are:

```
-Ofast -mfpu=vfp -mfloat-abi=hard -march=armv6zk -mtune=arm1176jzf-s
```

Note that `-Ofast` may produce compile errors. In this case, `-O3` or `-O2` should be used instead.

Note that `-mcpu=arm1176jzf-s` can be used in place of `-march=armv6zk -mtune=arm1176jzf-s`.

2026/02/09 22:23

Raspberry Pi Tips / Issues

Some stuffs...

Check if Running Raspbian-Lite

To check if we are on lite version, (well one way to do it...) is to check if raspberrypi-ui-mods package is installed. The following command will return nothing if it is not installed - so, we are probably on lite then.

```
$ sudo apt list raspberrypi-ui-mods --installed
```

Setting Timezone on Pi from commandline

I run desktop-less Raspbian (Buster) and somehow the timezone got reset. To set that,

```
# sudo timedatectl set-timezone Asia/Kuala_Lumpur
```

Change hostname manually

Edit `/etc/hostname` and `/etc/hosts` - modify accordingly, and reboot immediately.

```
$ name=my1pi
$ prev=$(cat /etc/hostname)
$ sudo sed -i "s/$prev/$name/" /etc/hostname
$ sudo sed -i "s/$prev/$name/" /etc/hosts
$ sudo reboot
```

Disable Display from Sleeping

The simplest way is... install xscreensaver and disable it!

```
# sudo apt install xscreensaver
```

ePhoto Frame

Script to make Pi an e-Photo Frame.

eframe.sh

```
#!/bin/bash

SHOW_PATH="/home/pi/Pictures"
[ -d "$1" ] && SHOW_PATH=`cd $1;pwd`
SHOW_EXEC="feh"
SHOW_FULL=`which $SHOW_EXEC 2>/dev/null`
# args: delay 5s, fullscreen, recursive, randomize
SHOW_ARGS="-D5 -F -r -z"

echo "Accessing pictures in $SHOW_PATH..."
# press esc to quit?
${SHOW_FULL} ${SHOW_ARGS} ${SHOW_PATH}
```

Set Static IP

- get the following info: address, netmask, broadcast, network, gateway
  - to get address, netmask, broadcast:

```
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.170.7.195  netmask 255.255.255.0  broadcast
10.170.7.255
```

  - to get network, gateway:

```
$ netstat -nr
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window
irtt Iface
0.0.0.0         10.170.7.1      0.0.0.0         UG        0 0
0 eth0
10.170.7.0      0.0.0.0         255.255.255.0   U         0 0
0 eth0
```

- edit `/etc/network/interfaces` and change

```
iface eth0 inet dhcp
```

to

```
iface eth0 inet static
address xxx.xxx.xxx.xxx
netmask xxx.xxx.xxx.xxx
network xxx.xxx.xxx.xxx
broadcast xxx.xxx.xxx.xxx
gateway xxx.xxx.xxx.xxx
```

note: fill in the info accordingly

Display Issues

Note *Got these from R-Pi site.*

The HDMI monitor (or HDMI adapter) may only support a limited set of formats for CEA (TV) or DMT (Monitor). To find out which formats are supported, use the following method:

- Set the output format to VGA 60Hz (hdmi_group=1 and hdmi_mode=1) and boot up your Raspberry Pi
- Enter the following command to give a list of CEA-supported modes: /opt/vc/bin/tvservice -m CEA
- Enter the following command to give a list of DMT-supported modes: /opt/vc/bin/tvservice -m DMT
- Enter the following command to show your current state: /opt/vc/bin/tvservice -s

Then edit config.txt to select:

- hdmi_group={0:EDID_auto_detect,1:CEA,2:DMT}
- hdmi_mode={select from tvservice output}

Auto-run Script/Program on Startup

Note*: This assumes a Raspbian installation with desktop environment, and username is* ***pi****.*

- create path `/home/pi/.config/autostart`
- create a desktop file in there (e.g. monitor.desktop)

monitor.desktop

```
[Desktop Entry]
Type=Application
Name=Monitor
Exec=/path/to/exec
```

Script to run browser full-screen

start_fullscreen.sh

```
#!/bin/bash

chromium-browser http://server.local:1337 --start-fullscreen
```

2026/02/09 22:23

Raspberry Pi: Peripheral Interfacing

Update in progress...

SIM7600 4G Module

More info here.

This is the code i use... I may put a more elaborate one some time in the future.

WARNING: Do NOT simply copy and run this.... make sure you understand what it does.

start_7600.sh

```
#!/bin/bash

#sudo apt install libqmi-utils udhcpc

#sudo qmicli -d /dev/cdc-wdm0 --dms-set-operating-mode='online'
#sudo qmicli -d /dev/cdc-wdm0 --dms-get-operating-mode
#sudo qmicli -d /dev/cdc-wdm0 --nas-get-signal-strength
#sudo qmicli -d /dev/cdc-wdm0 --nas-get-home-network
#sudo qmicli -d /dev/cdc-wdm0 -w

sudo ip link set wwan0 down
echo 'Y' | sudo tee /sys/class/net/wwan0/qmi/raw_ip
sudo ip link set wwan0 up

sudo qmicli -p -d /dev/cdc-wdm0 --device-open-net='net-raw-ip|net-no-
qos-header' --wds-start-network="apn=celcom3g,ip-type=4" --client-no-
release-cid

sudo qmi-network /dev/cdc-wdm0 start

sudo udhcpc -i wwan0
```

```
#ip a s wwan0
#ip r s
```

2026/02/09 22:23

Building OS Image for Pi

Using this...

Enables using Devuan instead of Raspberry Pi OS!

work-in-progress

2026/02/09 22:23

From:
http://azman.my1matrix.cc/ - **Azman @MY1**

Permanent link:
http://azman.my1matrix.cc/doku.php?id=raspi:raspi

Last update: **2026/02/08 03:46**