

[Slackware ARM project web site](#) | [Forum](#) | [Slackware ARM development documentation](#) | [Slackware ARM installation guides](#)

# Installing Slackware on the Raspberry Pi

Platform	AArch64/ARM64
Hardware Model	Raspberry Pi 4/400
Document Version	1.06, Mar 2024
Author	Stuart Winter <mozes@slackware>
Contributors	Brenton Earl <el0226@slackware> (R&D for the RPi4 Hardware Model)

## Supported Raspberry Pi versions



At the moment, only the Raspberry Pi 4 is officially supported.

The Raspberry Pi 400 has [been validated as working](#) by the user community.

## Video Tutorial

This tutorial is also available in [video form](#).



The video tutorial demonstrates the original installation approach where the Slackware installation media was separate. A single Slackware Installer image is provided that contains all of the media, so it's easier than shown in the video.

## Help / Support

Please post questions to the [Slackware ARM forum](#).

## Caveats

### Fragility due to upstream support

The support within the proper [Linux Kernel](#) is fragile due to the development model of the Raspberry Pi company. You may need to switch to the RPi Kernel fork for the best experience. Instructions on how to do this are included in the post installation section of this document.

## Installation Lifecycle

The Installation consists of a number of distinct stages:

- 1. Acquiring all required hardware
- 2. Setup of the Raspberry Pi hardware
- 3. Downloading and Verifying the Slackware assets
- 4. Writing the Slackware Installer to the Micro SD card
- 5. Booting the Slackware Installer
- 6. Installing Slackware
- 7. Completing the installation
- 8. Booting the Slackware OS
- 9. Post installation configuration and tweaks

## Requirements

### Hardware



This hardware list is what has been recommended by the Slackware community as a good starting point for setting up a Raspberry Pi. There are a plethora of peripherals available for the Raspberry Pi which you can use with Slackware.



SD cards: Some SD cards are known to have issues on the Raspberry Pi4. Please use one of the brands provided in the hardware table below.



Please purchase the hardware from any vendor: Slackware ARM has no affiliation with The PiHut, but it is a useful one-stop-shop for RPi's and accessories, and this author can attest to good service within the UK.

Item	Specification/Type	Required?	Notes
Raspberry Pi 4 Model B	4GB and 8GB RAM models have been tested	Yes	This is the bare board
'Ice Tower' CPU heat sink & cooling fan		A heat sink or CPU is required	This has been recommended as a suitable fan for high-CPU workloads.
Power supply	Official Raspberry Pi PSU	Yes	It's best to use the official PSU as others can cause instability over time.
Inline power switch	-	Optional but recommended	The Raspberry Pi 4 does not have reset nor power buttons on the board. Reports from the Slackware community suggest that the power socket on the Rpi will become damaged over time from wear. This inline power switch prevents damage to the RPi.

Item	Specification/Type	Required?	Notes
<a href="#">Micro to full-size HDMI adapter</a>	-	Optional	Depends on your local setup and whether you wish to connect an HDMI monitor to the RPi.
<a href="#">Real Time Clock (RTC)</a>	Adafruit PiRTC - Precise DS3231	Optional	This RTC been recommended by the Slackware community.
<a href="#">Battery for RTC</a>	Lithium 3volt CR1220	Optional	Depends on whether you use the RTC module. This enables the RTC to maintain the time whilst the RPi is powered down.
<a href="#">Dupont wires</a>	Female to male	Optional	This is to wire in the RTC module, as the large sized CPU heat sink prevents the RTC module from connecting directly to the GPIO. These wires are also useful for connecting other peripherals to the RPi's GPIO.
<a href="#">SATA storage / SSD</a>	Minimum useful size: 30GB	Required	Will contain the Operating System. You can install to other storage, but this documentation covers this particular configuration only. The Kingston-SA400S37-240G has thoroughly proved itself in the Slackware ARM build infrastructure - most build machines use these, but any SSD or spinning hard disk should work
<a href="#">USB to SATA adapter</a>	-	Required	Many models will work, but this one has been tested on many of this author's ARM hardware.
<a href="#">USB to Serial adapter</a>	PL2303 chip. Other models may work, but this one has been tested. If your model has the option to set voltages, ensure <b>3volts</b> is set!	Optional	A USB to Serial/UART adapter is recommended if you want to access the console remotely, but is unnecessary if you only plan on using an HDMI monitor. This document covers installing using an HDMI monitor - you can find information about connecting the Serial/UART adapter at the foot of this document
<a href="#">Micro SD Card</a>	16GB <b>minimum capacity</b> , Class 10 (fast speed)	Required	Initially used to boot the Slackware Installer, and subsequently transformed into Slackware's /boot partition.
USB keyboard and mouse	Standard peripherals	Required	A USB keyboard and mouse are required for this Slackware installation guide but are not required for the running of the OS. Slackware can also be installed headless (without a keyboard or monitor) over the Serial/UART or SSH (but this isn't documented).

### Recommended/ tested SD cards and other storage peripherals

In addition to the examples of tested hardware, here follows other storage devices and peripherals that have been tested by the Slackware community.

Type	Details
Micro SD card	<a href="#">Gigastone 32GB</a>
Micro SD card	<a href="#">Kingston 64GB microSDHC Canvas Select Plus</a>
Micro SD card	<a href="#">Kingston SDCG3</a>

Type	Details
Micro SD card	<a href="#">SanDisk 16GB</a>
USB to SATA adapter (alternative to the one in the table above)	<a href="#">SKL Tech</a>

Alternatives to SSD

If you prefer to use NVME storage, these have been tested:

NVME Storage Module	<a href="#">Kingston A2000 SSD 250G &amp; 500GB models</a>
USB to NVME M.2 adapter & enclosure	<a href="#">LEMORELE M.2 NVMe SSD Enclosure USB C External PCIe B+M/M-Key SSD Caddy 10Gbp UASP USB 3.1 Gen 2 M2</a>

Notes on storage setup

The setup documented here (2.5" SSD connected to a USB-to-Serial adapter for power and data) has proven stable for this author.


It is possible to create a 4th partition on the Micro SD card ('MMC') which can house the Slackware OS. However, this is not recommended due to the longevity of this type of storage - particularly if it has high I/O. If you want to install the OS to the MMC card, it's preferable to use the ["bare" Slackware Installer image](#) rather than the standard 'AiO' (All in One) Installer (which is what this Installation guide covers). Using the bare Installer means you will need to install over the network or from some locally attached storage.

Computing / Network Environment

Item	Specification	Notes
Host Computer: an Internet-connected computer running an existing Linux distribution	The Host Computer needs approximately <b>5GB free storage</b> to download the required software assets. <b>You must be able to obtain root access to this Host computer.</b>	This will be used to download the Slackware distribution from the Internet and to write the Slackware Installation media to the Micro SD card.

Hardware Setup

In this section we'll prepare the physical aspects of the Raspberry Pi4 to receive Slackware Linux.



Note that this is just one of a vast array of hardware configurations for the Raspberry Pi4. If you have a different configuration that's already working, you can skip to the 'Software and Network Environment Setup ' section

Attach the base board

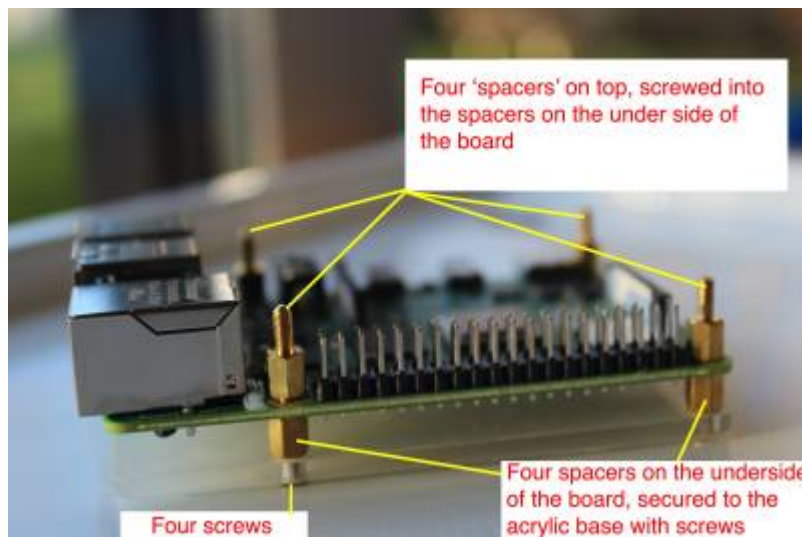
Attach the base board to the RPi4 using the brass spacers. Ensure that each brass spacer is screwed

into the base board, as shown.



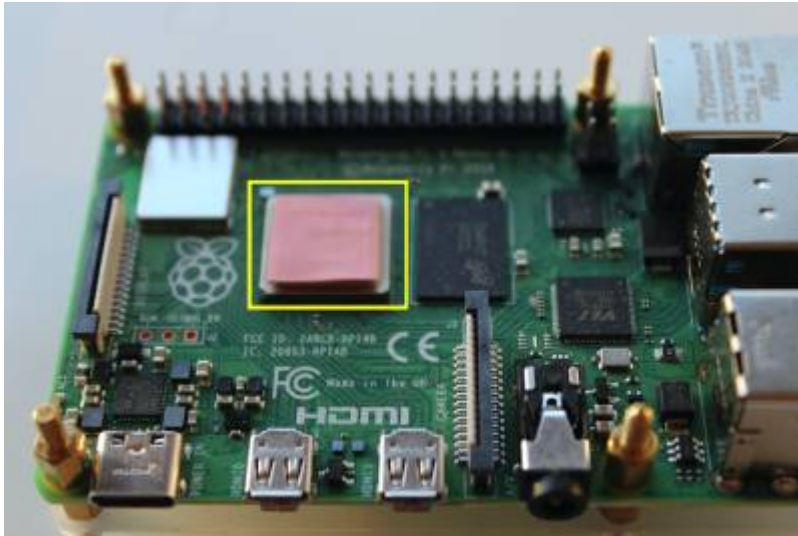
The base board comes with the 'Ice Tower' CPU cooling pack (as mentioned within the Hardware Requirements table). Other RPi “open” case designs would connect the RPi in a similar manner (e.g., the bottom element of Micro Connectors' Acrylic Stackable Raspberry Pi case)

#### **Screw four spacers to the top to secure the board to the platform**

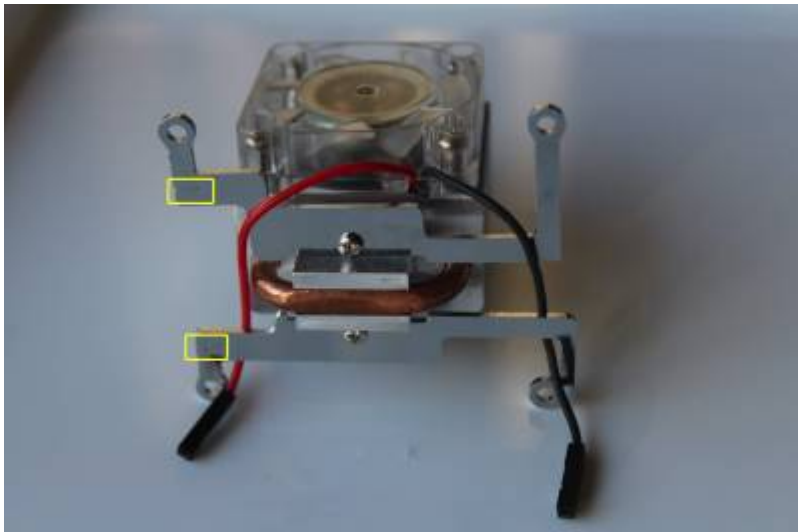


#### **Attach the thermal tape to the CPU**

Lay one of the pieces of thermal tape atop the CPU as shown:



### Attach the heat sink brackets



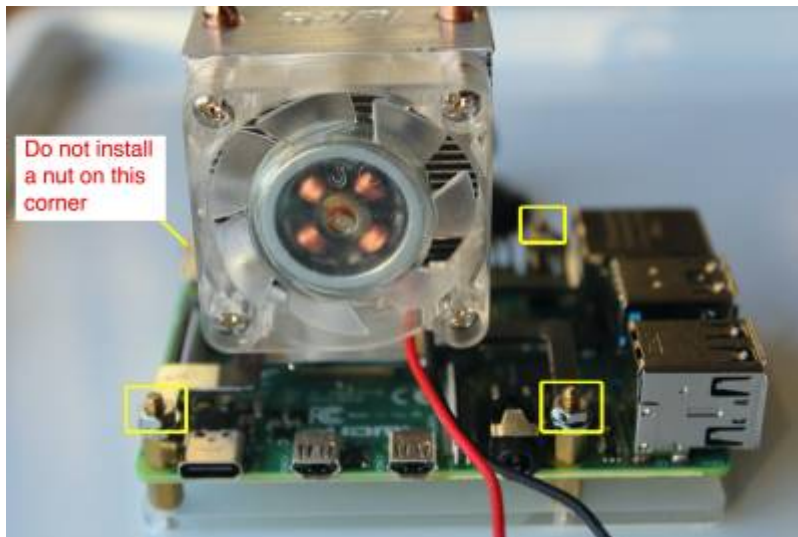
Pay close attention to the positioning of the brackets, otherwise it won't sit on the board properly.



Use the symbols on the brackets for orientation

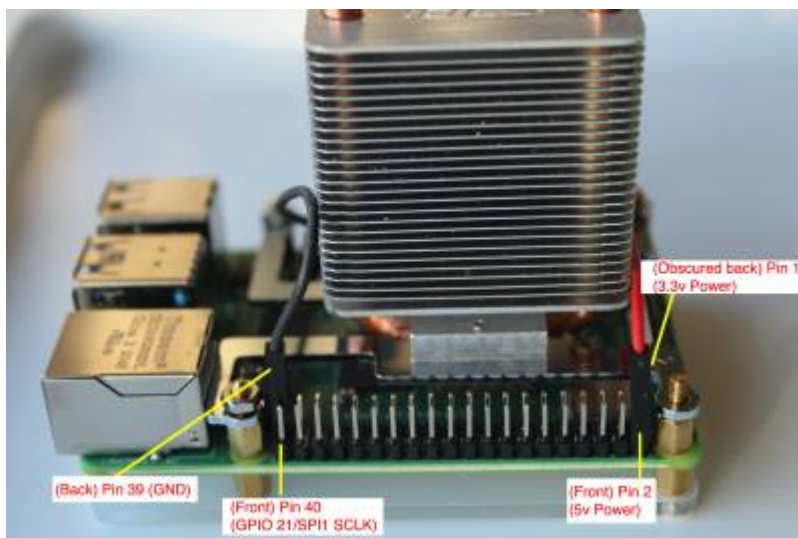
### Secure the board to the platform



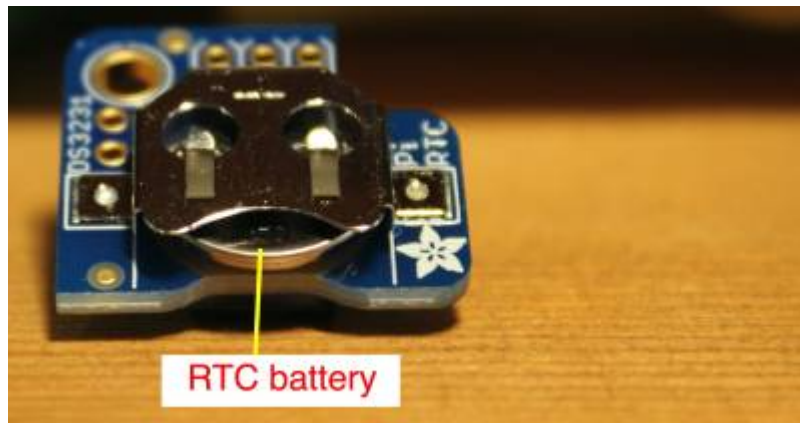


Only apply 3 nuts as shown. The fourth brass-coloured 'spacer' will be used to seat the RTC (Real Time Clock). If you do not have an RTC, secure this with a fourth nut.

### Attach CPU fan power wires

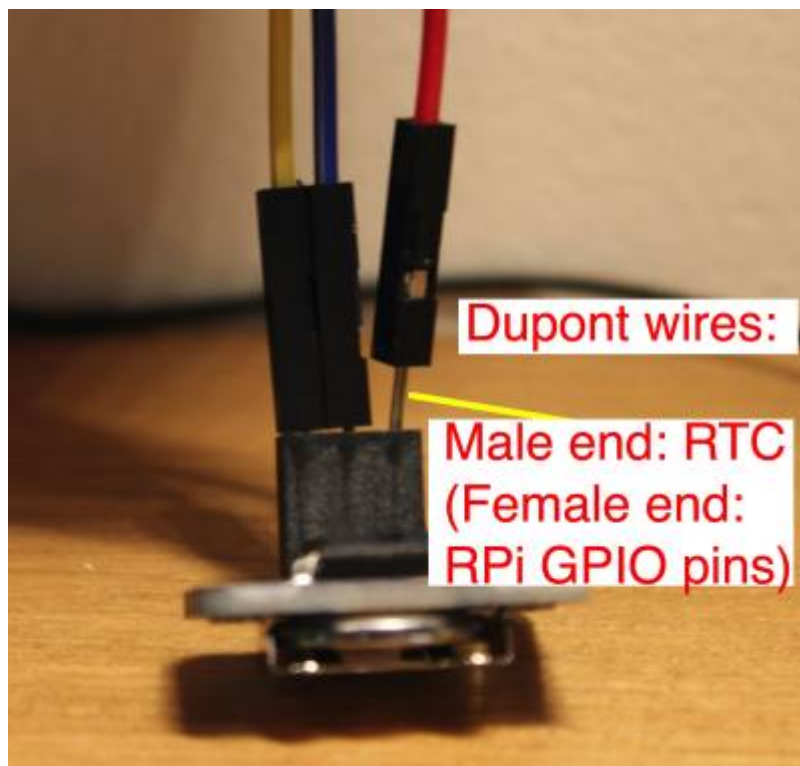


### RTC (Real Time Clock): Insert battery



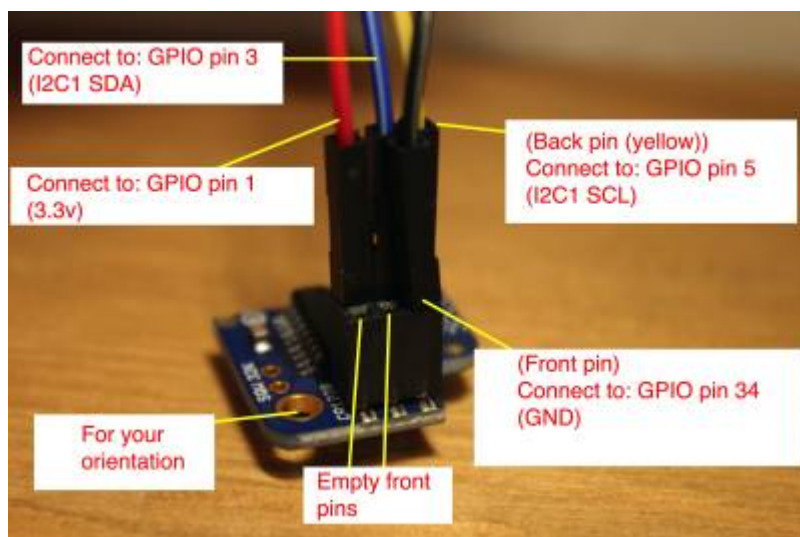
RTC battery

## RTC (Real Time Clock): Wiring



Dupont wires:

Male end: RTC  
(Female end:  
RPi GPIO pins)



Connect to: GPIO pin 3  
(I2C1 SDA)

Connect to: GPIO pin 1  
(3.3v)

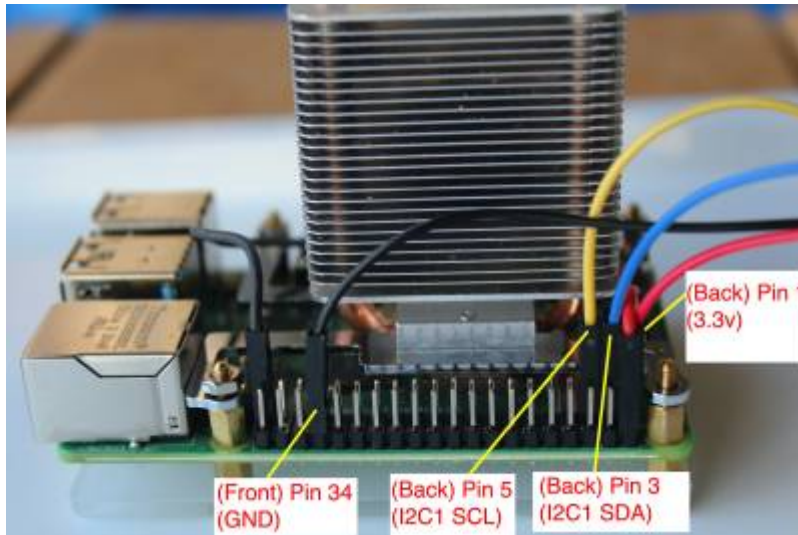
(Back pin (yellow))  
Connect to: GPIO pin 5  
(I2C1 SCL)

(Front pin)  
Connect to: GPIO pin 34  
(GND)

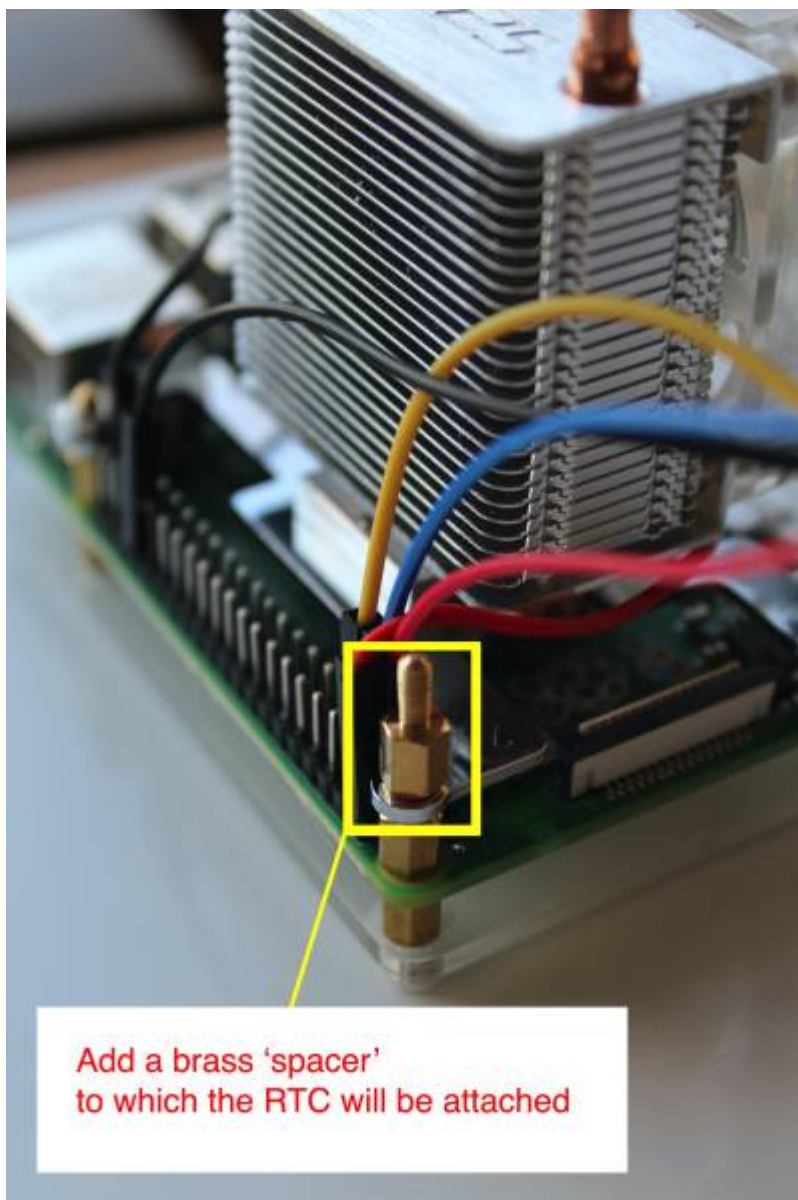
For your  
orientation

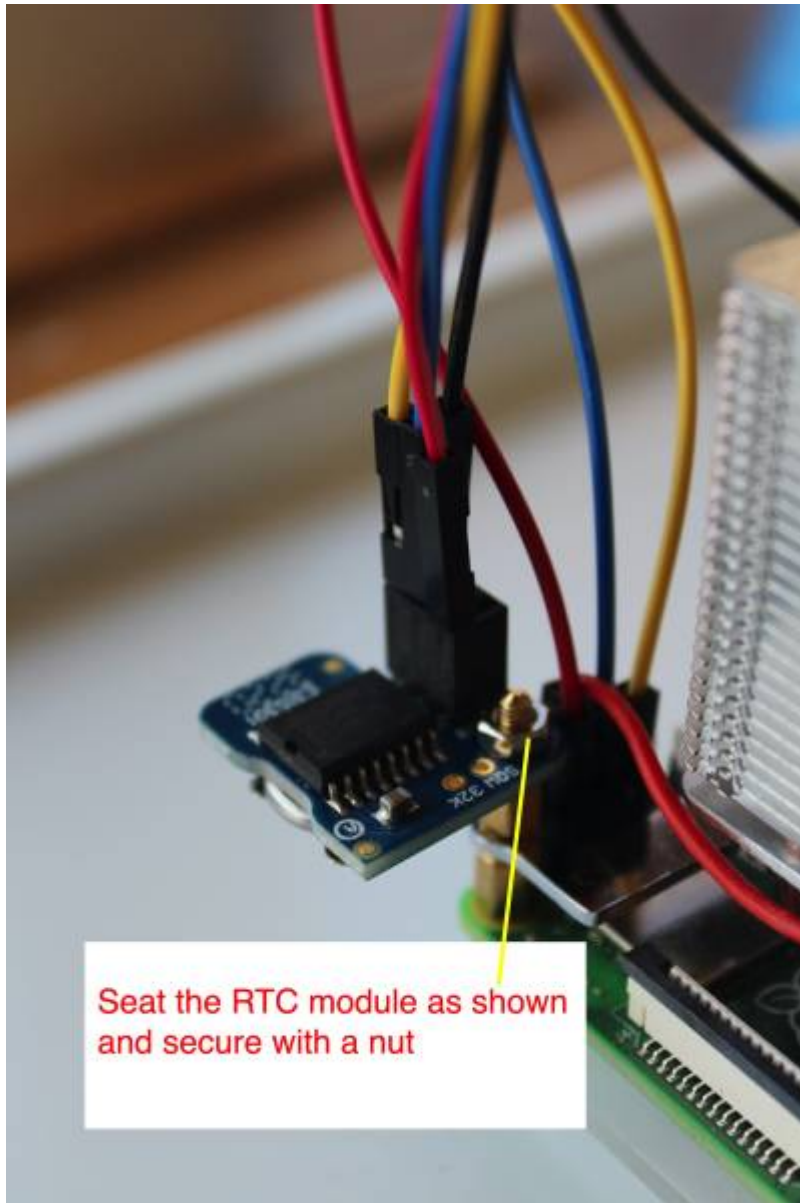
Empty front  
pins





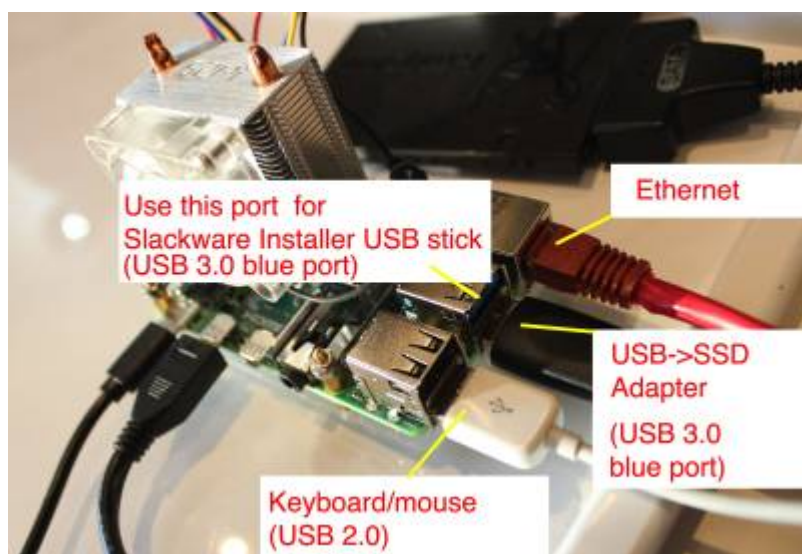
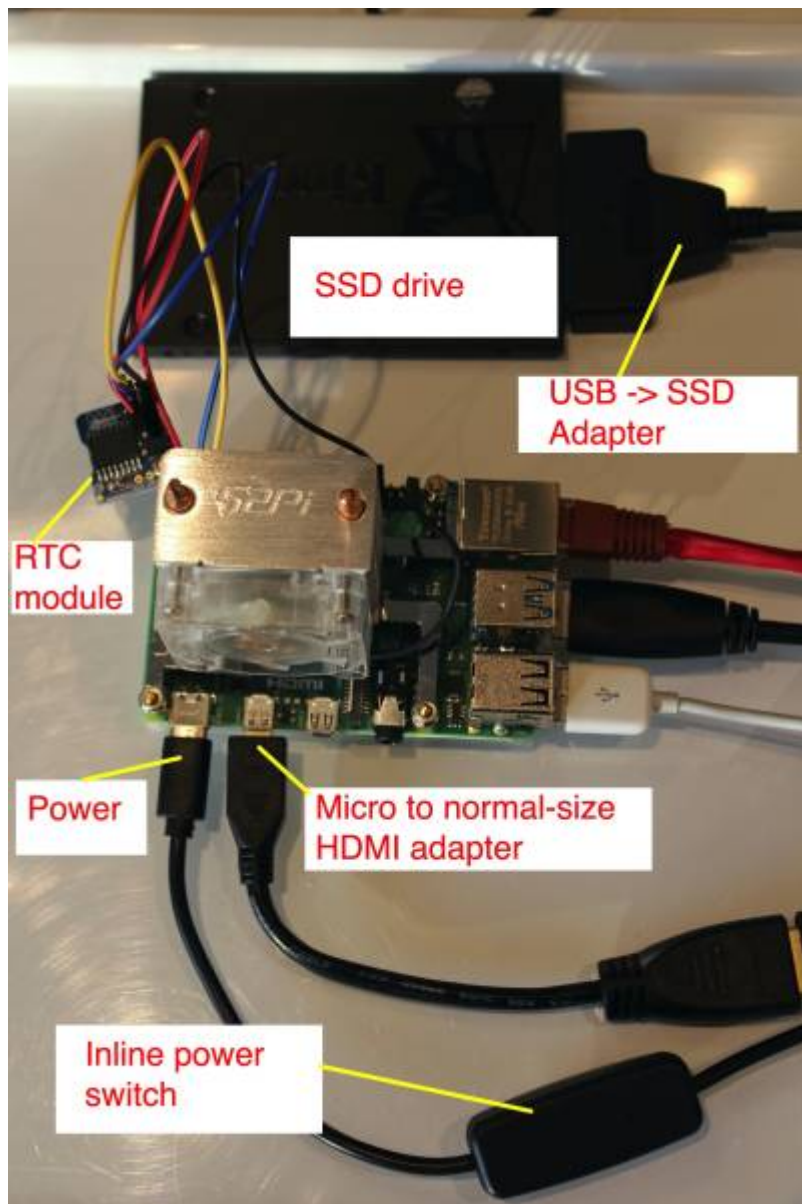
### RTC (Real Time Clock): Attach to Raspberry Pi





#### Add all other peripherals required to install Slackware

- Connect the storage - ensure that it's connected to a **blue USB 3.0** port
- Connect the Ethernet cable (optional - required if you want to set the date via NTP before commencing installation)
- Connect the HDMI cable
- Connect the USB keyboard and mouse



The basic hardware setup is complete.

# Downloading the Slackware assets

In this section, we'll prepare the Linux Host Computer to receive and download the Slackware assets required for the installation.

## 1. Downloading the Slackware Linux AArch64 Distribution and Installation Assets



The '\$' prefixes in the commands indicates the shell prompt - it's not to be typed/copied

Open a shell on the Linux Host Computer.

### Prepare a directory to hold and serve the Slackware Distribution

We'll download the Slackware Linux distribution into a directory named 'slackware'.

```
$ cd ## this returns to the root of your home directory
$ mkdir slackware
$ cd slackware
```

### Determine where you are within the Host Computer's Filesystem

```
$ pwd
/home/mozes/slackware
```



Note the directory location returned - you'll need this later

### Installing the Slackware ARM GPG key

The Slackware ARM GPG key will be used to verify your downloads.

```
$ curl -sSL https://www.slackware.com/infra/keys/arm/GPG-KEY | gpg --import -
```

### Set the model of Raspberry Pi



The Raspberry Pi Slackware Installer images are identical, so if you have both the RPi3



& 4 you should only download one image. If this changes, this message will be removed.

For the Raspberry Pi 3:

```
$ HWM=bcm2837_rpi3
```

For the Raspberry Pi 4:

```
$ HWM=bcm2711_rpi4
```

### Set the version of Slackware AArch64 to download

At the time of writing, the only version available is 'current'.

```
$ SLKVER=current
```

### Set the Internet media distribution server

If you are using a mirror server rather than the master Slackware ARM server, set it here. The format is: <hostname>::<rsync module name>

```
$ SLKSRV=ftp.arm.slackware.com::slackwarearm
```

### Download the Slackware Linux Installer

```
$ rsync -PavL $SLKSRV/platform/aarch64/bootware/installer-  
aio/slackwareaarch64-${SLKVER}/${HWM}.img.xz.asc slkaio.img.xz.asc  
$ rsync -PavL $SLKSRV/platform/aarch64/bootware/installer-  
aio/slackwareaarch64-${SLKVER}/${HWM}.img.xz slkaio.img.xz
```

*The Slackware Installer images are approximately 5 GBytes in size.*

### Verify the Slackware Installer image

Verify the digital signature of the Slackware Installer:

```
$ gpg --verify slkaio.img.xz.asc
```



As the images are large, verification may take a minute or two.

The output will be similar to this. You are looking for 'Good signature from Slackware ARM...'



```
gpg: assuming signed data in `slkaio.img.xz'
gpg: Signature made Wed 24 Nov 2021 06:07:44 PM BST
gpg:                using RSA key F7ABB8691623FC33
gpg: Good signature from "Slackware ARM (Slackware ARM Linux Project)
<mozes@slackware.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:                There is no indication that the signature belongs to the
owner.
Primary key fingerprint: 36D3 7609 2F12 9B6B 3D59  A517 F7AB B869 1623 FC33
```



If you see 'BAD signature' you should re-download as it may have become corrupt. If this doesn't help, drop a note to the [Slackware ARM forum](#)

## Writing the Slackware assets to the Micro SD

### Elevate yourself to root

On your Host Computer, obtain root:



The # prefix indicates that you're using the **root** user - it's not to be typed in!

```
$ su -  ## Note the hyphen - it's required
```

### Write the Slackware Installer to the Micro SD card

#### Check what block devices are present

Prior to inserting the Micro SD Card into the USB adapter, we need to see what's already present within the OS so that we can easily locate our Micro SD card:

```
# lsblk -d
NAME MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda   8:0    0 465.8G  0 disk
```

As you can see, this Host Computer there is a single storage device - *sda*.

**Insert the Micro SD card into your USB Card Reader and connect the adapter to a free USB port on the Host Computer**

Run lsblk again:



```
# lsblk -d
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda          8:0    0 465.8G  0 disk
sdc          8:32    1    58G  0 disk
sdd          8:48    1     0B  0 disk
```

As you can see, *sdc* is 58 GBytes in size. This is the Micro SD card (in this example, it's labeled as '64GB' on the exterior of Micro SD card).

If your Micro SD card has existing partitions, you will not see them surfaced in this list - use *lsblk -b* to view them.



You'll also observe the presence of *sdd* - often the USB adapter itself obtains a block device. You can ignore this as it's 0Bytes.

## Write the Slackware Installer to the Micro SD card

Enter the directory into which the Slackware assets were downloaded previously:

```
# cd ~mozes/slackware ## Substitute with the path you noted earlier
# xzcat slkaio.img.xz | dd status=progress bs=4M iflag=fullblock of=/dev/XXX
## Substitute/dev/XXX with the correct block device
# sync
```

## Remove the MicroSD card from the Host Computer

You may now disconnect the USB adapter/remove the Micro SD card from the Host Computer.

## Logout from the root user

We no longer need to use the Host Computer, so you can logout of the root shell.

```
# logout
```

# Installing Slackware

To proceed, you must have:

- Connected the storage to the Raspberry Pi 4
- Connected the HDMI monitor
- Connected the keyboard (and optionally, mouse)
- Optionally connected the Ethernet cable to set the date via NTP from the Internet
- Inserted the Micro SD card containing the Slackware Installer

## Encrypted storage

If you'd like to encrypt your storage, check the [Disk Encryption Guide](#).

## Begin installation



Disconnect any USB storage devices that aren't required for the OS installation



Due to bugs in the upstream Linux Kernel, the screen may blank every few seconds. This is normal and can be resolved by switching to the Raspberry PI Kernel fork (detailed later in this guide).

## Power on the Raspberry Pi 4

Apply power to the Raspberry Pi and after a few seconds, you will see the following on screen:

```
Found /extlinux/extlinux.conf
Retrieving file: /extlinux/extlinux.conf
1:      Boot Installer: Slackware Linux
Retrieving file: /initrd-armv8.img
```

It takes several seconds to load and boot the installer, and it may take several seconds more for any further output to appear on the HDMI monitor. The process of obtaining an IP address via DHCP can also delay the ability to interact with the Installer.

Once an IP address has been obtained, you will be presented with a prompt. Press **ENTER**

```
Please wait, initializing network...
Please wait, initializing SSH server...

The Slackware Installer is ready

Press Enter to activate this console.
```

## Set the keymap

```
<OPTION TO LOAD SUPPORT FOR NON-US KEYBOARD>

If you are not using a US keyboard, you may now load a different
keyboard map. To select a different keyboard map, please enter 1
now. To continue using the US map, just hit enter.

Enter 1 to select a keyboard map: 1
```



```
OK, the new map is now installed. You may now test it by typing
anything you want. To quit testing the keyboard, enter 1 on a
line by itself to accept the map and go on, or 2 on a line by
itself to reject the current keyboard map and select a new one.

1
```

## Font size

If you're using a smaller monitor, such as one with a screen size less than 20 inches, it may be necessary to adjust the console font size to ensure that menus and other interface elements fit correctly on the screen. If so, type this into the shell prompt:

```
setfont ter-v18n
```

## Set the date/time

Even if you have a battery pack for the RTC (Real Time Clock), the date on your system may be incorrect. We will sync the date from a highly-available NTP server:

```
ntpdate clock.akamai.com
```

```
hwclock -w
```

```
root@slackware:~# ntpdate clock.akamai.com
22 Nov 14:36:57 ntpdate[956]: step time server 2.18.25.79 offset +279177954.929463 sec
root@slackware:~# █
```

Setup disk partitions

For this installation a basic partitioning scheme will be created.

Partition

Partition number	Device name	Size	Purpose
1	/dev/sda1	4GB	Swap
2	/dev/sda2	Rest of storage	OS root ('/') partition



/boot will reside on the Micro SD card and is automatically configured by the Slackware Installer

Open fdisk against the /dev/sda block device. In this guide, /dev/sda will be your primary storage, and in this guide is the SSD connected to the USB adapter.

```
fdisk /dev/sda
```

```
root@slackware:~# fdisk /dev/sda

Welcome to fdisk (util-linux 2.37.2).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0xd88c478f.
```

Clear an existing partition table: Press 'o' to clear the partition table

```
Command (m for help): o

Created a new DOS disklabel with disk identifier 0xc32fb51d.
```

Create the Swap partition:

Type 'n' for new partition:

```
Command (m for help): n
Partition type
  p   primary (0 primary, 0 extended, 4 free)
  e   extended (container for logical partitions)
Select (default p):

Using default response p.
Partition number (1-4, default 1):
First sector (2048-468862127, default 2048):
Last sector, +/-sectors or +/-size[K,M,G,T,P] (2048-468862127, default 468862127): +4G

Created a new partition 1 of type 'Linux' and of size 4 GiB.
```

Type 'p' for primary partition type:

Press ENTER for the 'First sector'

Type '+4G' for the 'Last Sector'/size:

Change the partition type to 'Swap'. Type 't' then hex code '82':

```
Command (m for help): t
Selected partition 1
Hex code or alias (type L to list all): 82
Changed type of partition 'Linux' to 'Linux swap'.
```

Create the partition for the root filesystem ('/'):

Type 'n' for new partition. Press ENTER to accept the defaults - this will create partition 2 as the maximum size available.

```
Command (m for help): n
Partition type
  p   primary (1 primary, 0 extended, 3 free)
  e   extended (container for logical partitions)
Select (default p):

Using default response p.
Partition number (2-4, default 2):
First sector (8390656-468862127, default 8390656):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (8390656-468862127, default 468862127):

Created a new partition 2 of type 'Linux' and of size 219.6 GiB.
```

Type 'a' to mark the root partition (number 2) as bootable Type '2' to select partition 2.

```
Command (m for help): a
Partition number (1,2, default 2): 2

The bootable flag on partition 2 is enabled now.
```

Type 'p' to print to view the partition table.

```
Command (m for help): p
Disk /dev/sda: 223.57 GiB, 240057409536 bytes, 468862128 sectors
Disk model: SA400S37240G
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 33553920 bytes
Disklabel type: dos
Disk identifier: 0x9feb6c33

Device    Boot  Start      End  Sectors  Size Id Type
/dev/sda1             2048  8390655   8388608    4G 82 Linux swap
/dev/sda2 *    8390656 468862127 460471472 219.6G 83 Linux
```

Type 'w' to write the partition table:

```
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

root@slackware:~#
```

fdisk will now exit.

## Load the Setup menu

```
root@slackware:~# setup
```

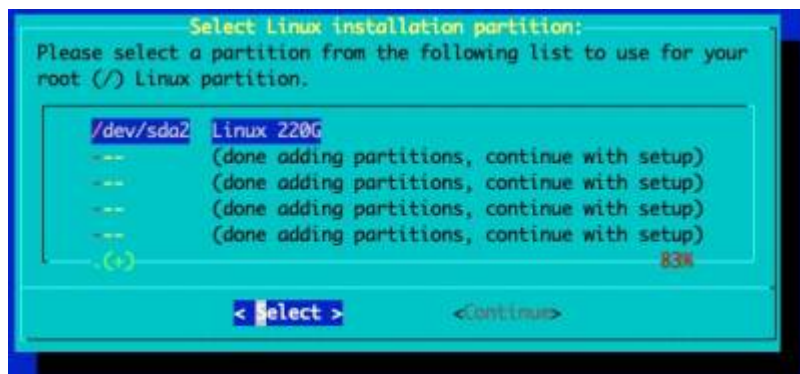
## Setup Swap partition

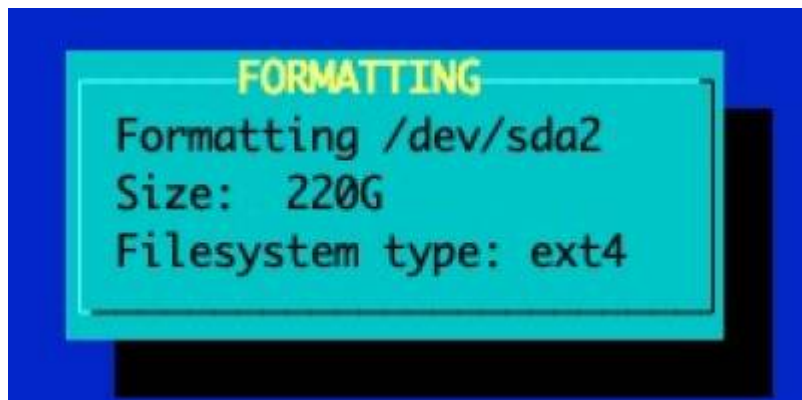






Select and format the partition for the OS' root file system

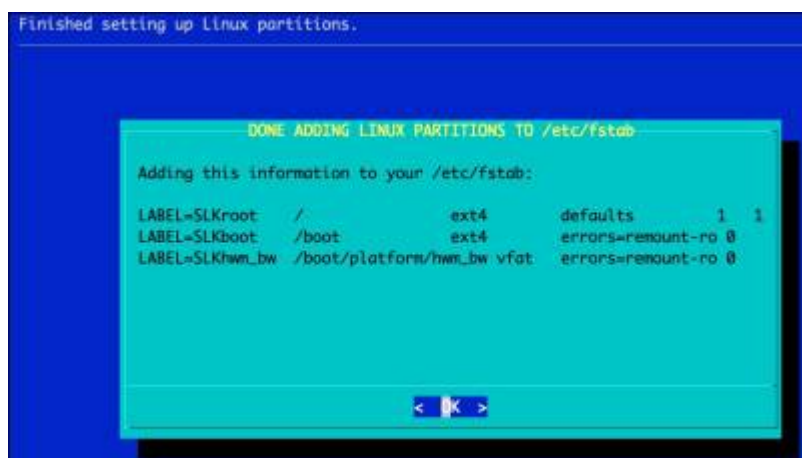




## Boot Loader Configuration

The Installer will configure the Boot Loader and the OS' /etc/fstab automatically:

```
Detected Slackware bootable SD card: /dev/mmcblk0p2
Attempting to mount ...
mount: /dev/mmcblk0p2 mounted on /mnt/boot.
Configuring U-Boot (/mnt/boot/extlinux/extlinux.conf)
...enabling awaitrootdev OS InitRD feature
...setting root filesystem
...setting the default boot option to Slackware OS
Detected Hardware Model Bootware partition: /dev/mmcblk0p1
Attempting to mount ...
mount: /dev/mmcblk0p1 mounted on /mnt/boot/platform/hwm_bw.
```



On the ARM platform, the Swap and root file systems are addressed by labels (see above: 'LABEL=') where as on x86/64 it's addressed by a direct reference to the block device (e.g. /dev/sda). The Slackware Installer will label swap partitions as 'SLKswap<x>', and the root file system 'SLKroot'. Other labels in use are 'SLKboot' for the OS boot partition (/boot), and 'SLKhwm\_bw' for the Hardware Model Bootware partition (/boot/platform/hwm\_bw). These labels are pre-labeled on the SD card image that you deployed and won't be presented during the installation process.

The rationale behind this divergence is that on x86 the root file system is typically on a storage bus (SCSI, SATA, ATA), where the physical configuration (which port the storage is connected to) of the storage rarely changes. This can be the case on ARM, but it's generally to a lesser extent and the root file system may be connected to a

hot-plug bus such as USB. This lends itself to the risk of device re-ordering across boot cycles (e.g. /dev/sda becomes /dev/sdb), causing boot failure.



Please be aware that the Slackware Installer *only* labels the swap and root file system. Therefore you are advised to manually label the file systems and modify the OS /etc/fstab accordingly. If you have only a single storage device and don't plan on adding more, you can use the settings that the Slackware Installer configures.

## Select Source Media



Press ENTER to say 'Yes'.

If you would like to install from an alternate media source, pick 'No' and you will be presented with options to install over NFS, USB and HTTP amongst others.

Ordinarily you should always say 'Yes' unless you've been directed to do otherwise.

## Package Series Selection

You can now choose the package sets to install. The recommendation is to install everything. A full Slackware installation will occupy approximately 15GB.



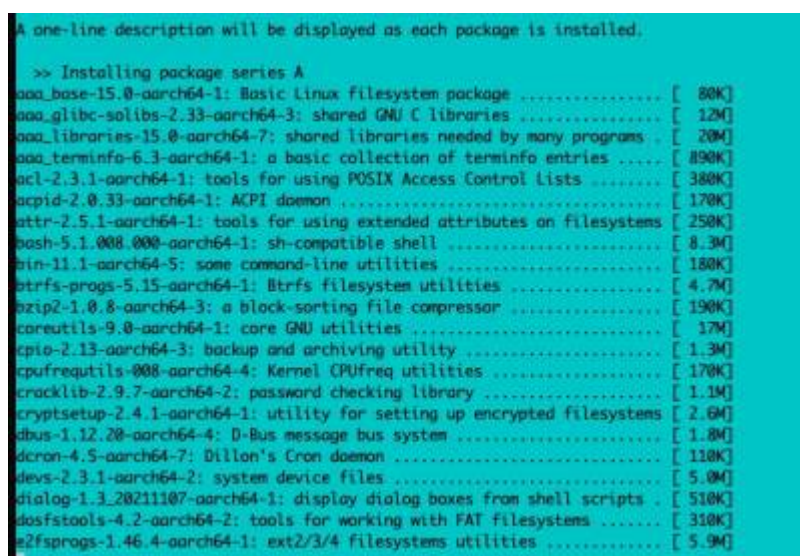
If you do not plan to use the graphical window manager such as KDE, you should de-select it.



Pick the 'terse' option:



The packages will begin installing:



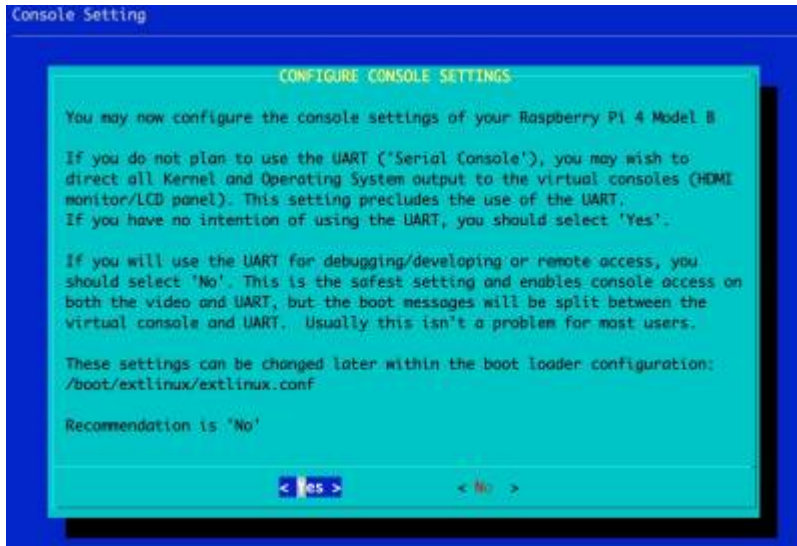
## Configure the Console Settings



If you plan on using the UART/'Serial' console, you should select 'No' here. If you plan on exclusively using an HDMI monitor, you should pick 'Yes'.



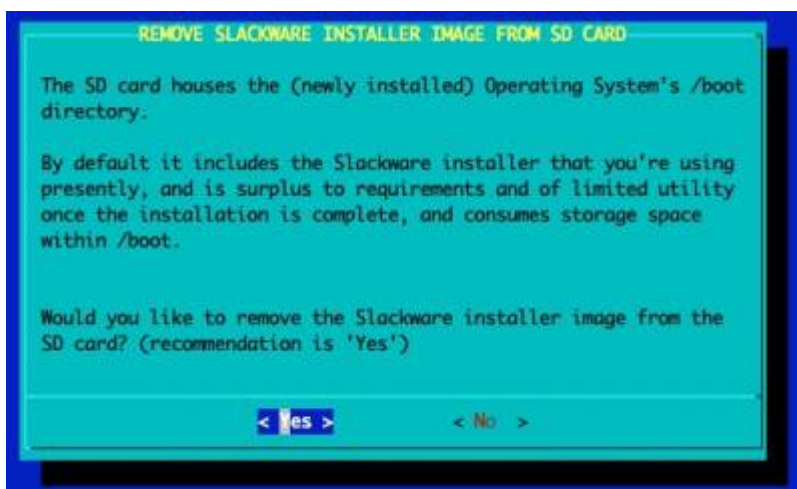
This setting can be reset to the default by editing `/boot/extlinux/extlinux.conf` and removing the `'console='` setting once the OS has booted.



### Remove the Slackware Installer from the SD card

The Micro SD card is transformed from being the Slackware Installer into the Slackware OS's `/boot` partition. At this stage, if the installation has worked for you (at certain points in the Slackware installer you are past the point of no return) you can delete the Installer. However, if something has gone wrong you can reset the Raspberry Pi and reboot the installer without having to re-deploy the Slackware Installer image from your Linux Host Computer.

Generally you should say 'Yes' here.



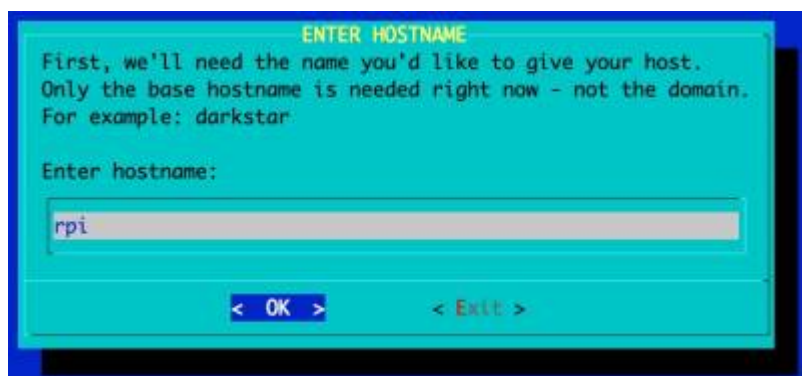
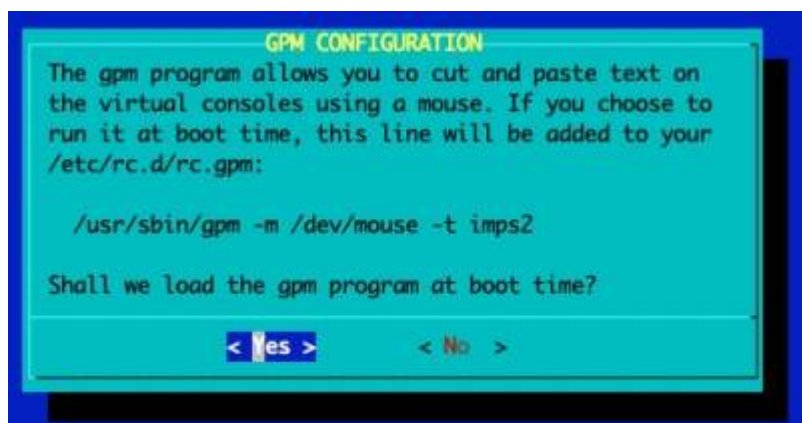
You may be tempted to retain the Slackware Installer, but note that the Installer contains Linux Kernel modules for the Kernel that the Installer was originally shipped with. This means that as soon as you upgrade the Slackware Kernel package, the



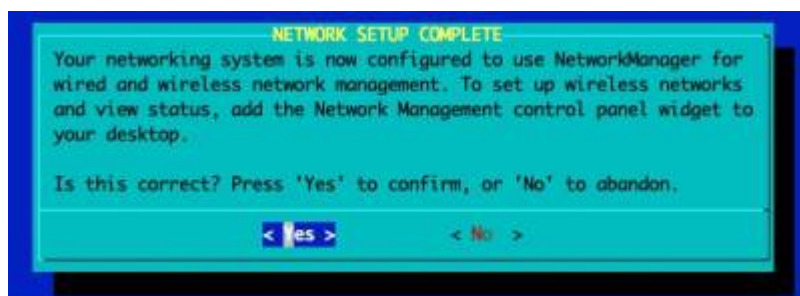
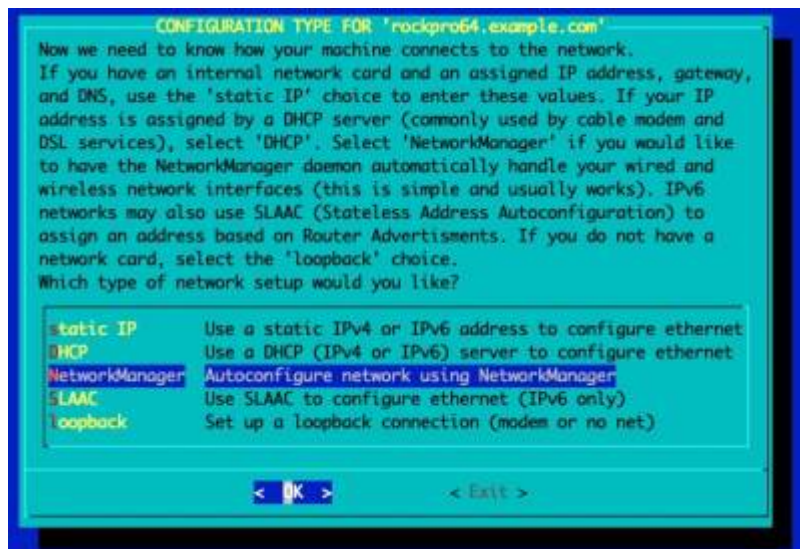
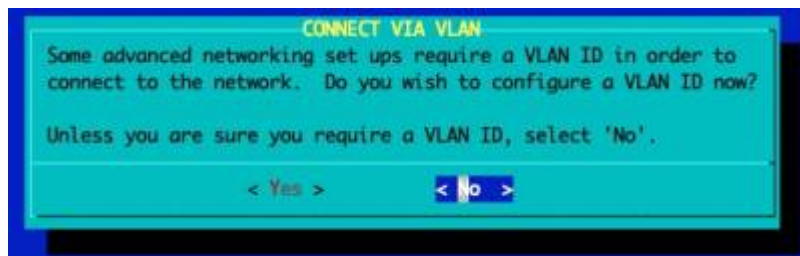
Installer will fail to boot. The option to retain the Installer is present purely because on a number of occasions, this author only realised that the installation was incorrectly performed upon completion, and needed to reinstall. Retaining the Installer avoids the requirement to re-deploy the image to the SD card.

## Post Installation Configuration

The Slackware Installer will walk you through the standard Slackware setup. The on-screen instructions will suffice.





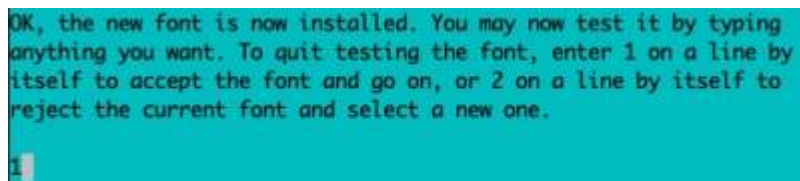


## Select a Console Font

It's recommended for the Raspberry Pi that a larger console font is configured for the virtual console.

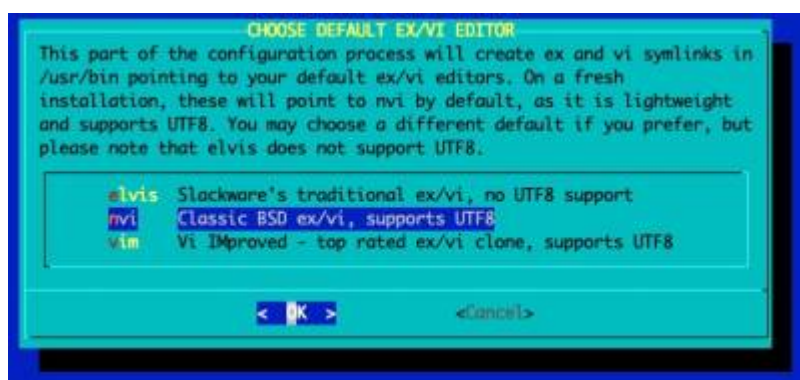


The recommended font is 'ter-732b.psf'. This is the font used within the Installer.



## Continue Post Installation Configuration



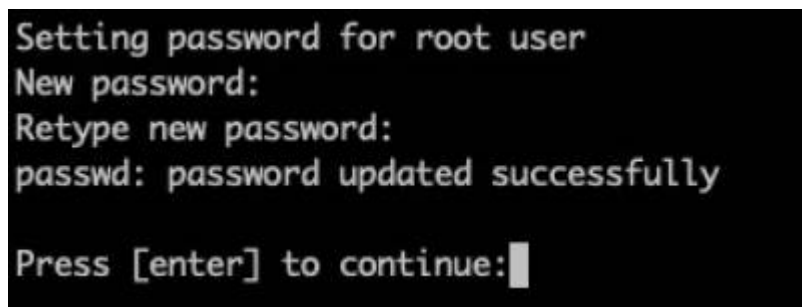


## Configure GUI Window Manager

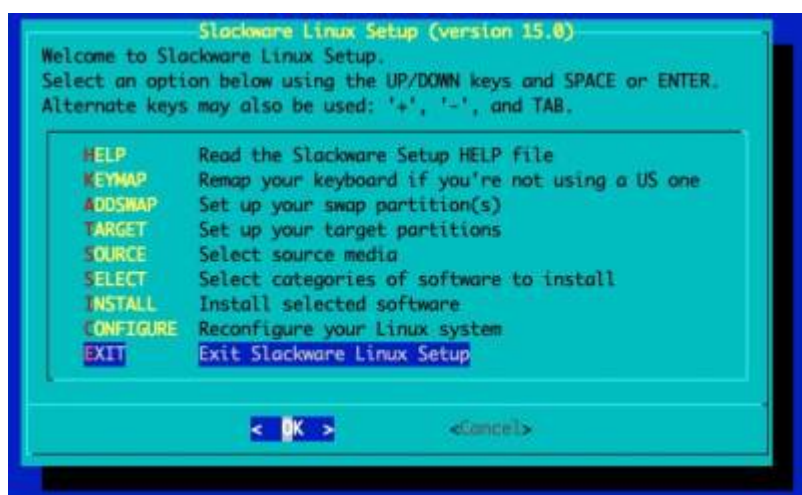
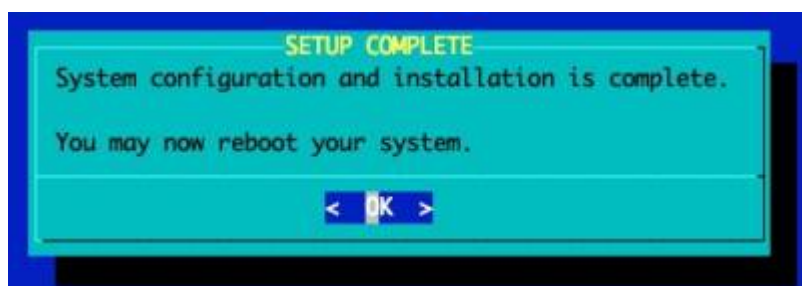
This author recommends using XFCE as it's light weight versus KDE.



## Continue Post Installation Configuration

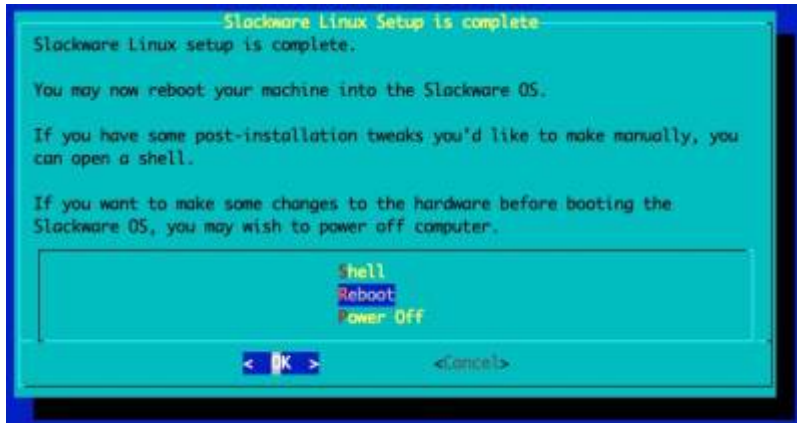


## Slackware Setup Complete



## Reboot into the Slackware OS





Generally you'll want to reboot into the OS.

However, if you are planning on setting up RAID or need to customise the Operating System Initial RAM Disk, you should select 'Shell'.

The Slackware OS will be found within '/mnt'. You can use the 'os-initrd-mgr' tool ([Video tutorial](#)).

### Booting the Slackware OS

```
U-Boot 2021.10 (Dec 10 2021 - 11:11:45 +0000) Slackware

DRAM: 7.9 GiB
RPI 4 Model B (0xd03114)
MMC: mmc@7e300000: 0, mmc@7e340000: 1
Loading Environment from FAT... Card did not respond to voltage select! : -110
In: serial
Out: serial
Err: serial
Net: eth0: ethernet@7d580000
PCIe BRCM: link up, 5.0 Gbps x1 (SSC)
starting USB...
Bus usb@7e980000: USB DWC2
Bus xhci_pci: Register 5000420 NbrPorts 5
Starting the controller
USB XHCI 1.00
scanning bus usb@7e980000 for devices... 1 USB Device(s) found
scanning bus xhci_pci for devices... 5 USB Device(s) found
    scanning usb for storage devices... 0 Storage Device(s) found
Hit any key to stop autoboot: 0
Card did not respond to voltage select! : -110
switch to partitions #0, OK
mmc1 is current device
Scanning mmc 1:2...
Found /extlinux/extlinux.conf
Retrieving file: /extlinux/extlinux.conf
6351 bytes read in 18 ms (343.8 KiB/s)
1: Boot OS: Slackware Linux
Retrieving file: /initrd-armv8
```

### Login to the Slackware OS

```
Welcome to Linux 5.15.9-armv8 aarch64 (tty1)

rpi login:
```

You may now login as 'root', using the password you set within the installer.

# Post Installation Configuration

There are a few post-installation configuration tasks to complete.

## Initial Time/Date Sync

If Internet access is available to the Raspberry Pi, prior to proceeding with any further setup, you may wish to set the time now as a one-off event:

Elevate yourself to **root** and use **ntpdate**:

```
$ su -  
# ntpdate rolex.ripe.net # or pick your favourite NTP server  
# hwclock -w # writes the date to the RTC  
# logout
```

## NTP (Network Time Protocol) setup

Even if your Raspberry Pi has an RTC (as documented in this guide), you may wish to configure it to [set time from an Internet NTP Server](#). The Raspberry Pi requires continuous Internet access for this to function.

## Add a plebeian user

You should add a plebeian (non-root) user using the 'adduser' tool.

This is documented [here](#).

## KDE fixups

If you are using KDE, you need to first adjust a setting.

As your plebian user, **prior** to loading KDE, run this command.



This will disable the *Compositing* feature. This is required when using the mainline Linux Kernel (as Slackware does).

```
kwriteconfig5 --file kwinrc --group Compositing --key Enabled false
```

You may now start KDE.



## Use a graphical login manager

If you prefer to use a graphical login manager, you can configure the default runlevel as 4:

```
su -  
sed -i 's?id:3:?id:4:?g' /etc/inittab  
reboot
```

# Managing Slackware on the Raspberry Pi

## Keeping the Slackware OS up to date

One of the preferred tools to keep your system up to date is [slackpkg](#).



**Upgrading the Kernel:** in Slackware x86/64 manual steps are required after upgrading the Kernel packages. In Slackware ARM, you simply upgrade the Kernel packages and reboot. If you make customisations to the OS InitRD (see below) these will automatically be re-incorporated during the upgrade process.

## Loading Additional Linux Kernel Modules within the OS Proper

Often Kernel modules for discovered hardware will be automatically loaded, but occasionally you will need to manually configure the loading of some modules.

```
/etc/rc.d/rc.modules.local
```

This file is a shell script that is run as one of the last steps before the OS has fully booted. You can enter `modprobe` commands here to load the specific modules you require.

Configuration files within the directory `/lib/modprobe.d/` can be used to configure the parameters of the modules. Existing files within that directory serve as reference examples should you need them.

## Loading Additional Linux Kernel Modules early in the boot sequence

There are a number of peripherals that may require Kernel modules loading early on in the boot sequence. An example of this would be RTCs (Real Time Clocks) or storage controllers that are required to access the file systems on which the OS lives.



Usually you won't need to load modules early in the boot sequence. See the previous section about loading modules from within the OS Proper.

To load Kernel modules during the early boot sequence, read:

```
/boot/local/README.txt
```

As root, the easiest way to begin is by renaming the example script:

```
mv /boot/local/load_kernel_modules.post.sample  
/boot/local/load_kernel_modules.post
```

Then add the appropriate module loading commands to:

`/boot/local/load_kernel_modules.post` You can also add shell code here to initialise a peripheral - writing something to the peripheral's Kernel interface, for example.

## Slackware repository partition

The Slackware Installer image contains a type ext4 partition labeled `SLKins_aio-pkgs` from which the packages are installed.

```
root@slackware:~# mount LABEL=SLKins_aio-pkgs /mnt/zip
```

```
root@slackware:~# cat /mnt/zip/README.txt
```

This file system contains the Slackware repository that is used during the installation of Slackware.

Once you've booted into your OS you can delete or change this partition if you wish, or perhaps you might like to retain it for future reference.

```
root@slackware:~#
```



Most users simply leave the partition alone, as it causes no issues.

## Customising the Slackware Linux Kernel

If you'd like to customise the Linux Kernel, the easiest way is to follow the [HOWTO](#) guide and use the Slackware ARM Kernel build script to create new packages.

This document also covers using the Raspberry Pi Kernel fork (although this is not recommended).

## Reducing Boot Time

Slackware ARM ships with a generic OS InitRD (Operating System Initial RAM Disk - the environment that prepares the machine to boot the Operating System Proper), so as to support a wide range of Hardware Models.

However, this isn't the optimal setup once the Slackware OS has been installed because the generic OS InitRD typically exceeds 250MB, which in some cases can add several seconds to the boot time whilst it's loaded from the SD card.

The `os-initrd-mgr` (Operating System Initial RAM Disk Manager) tool has an option to synchronize the OS InitRD's Kernel modules with *only* those presently loaded within the Operating System.

To do this:

```
$ su -c 'os-initrd-mgr --sync-loaded-kmods' - # note the final -
```



To have this setting persist across Kernel upgrades, you must upgrade the `a/kernel-modules` package before `a/kernel`. If not, it'll revert to the generic OS InitRD until you next reboot. If you are using `slackpkg` to manage upgrades, this is handled for you.

This option isn't the default, but you can make it so by following the instructions within `/etc/os-initrd-mgr.conf.sample`

This way when you upgrade the Kernel packages in the order described above, it'll automatically synchronize the modules.



`os-initrd-mgr` has a safety check to only proceed when the running Kernel and incoming Kernel are at the same major version and patch level.

For example, when running Linux 5.17.1, upgrading to 5.17.2 will work; but an upgrade of Linux 5.17.1 → 5.18.1 will require a reboot then to run `os-initrd-mgr` again to re-sync.



If at any point you want to revert to the generic OS InitRD, simply reinstall the `a/kernel` package (and unset the setting if you configured it in `/etc/os-initrd-mgr.conf`).

## Managing the Raspberry Pi Firmware



Do not use the `rpi-update` script (found in other distributions) - this is unnecessary



and is incompatible with Slackware.

The Raspberry Pi firmware is managed by the Slackware packages.

## Bootware

The Raspberry Pi boots from a FAT partition on the Micro SD card. This partition contains the closed-source proprietary boot loader, firmware and various other assets it requires. It is mounted within the Slackware OS under `/boot/platform/hwm_bw` (on other distributions these files reside within `/boot`).

These assets are provided and managed by the Slackware package `a/hwm-bw-raspberrypi`.



Simply upgrading to the latest available Slackware packages will update this firmware.

## Firmware for the EEPROM

The Slackware package `a/hwm-bw-raspberrypi` contains the available firmware that can be programmed to the Raspberry Pi's EEPROM.

You also need to have the `a/rpi-userland` package installed.



This firmware update requires manual action - see below

To update the firmware on your RPi's EEPROM; as root:

```
$ rpi-eeprom-update -d
```

This will report if there's newer firmware available.

To update the firmware:

```
$ rpi-eeprom-update -d -a
```

If the firmware was successfully updated, reboot:

```
$ reboot
```



The EEPROM tool configuration file is: `/etc/rpi-eeprom-update`. By default we're using firmware from the "critical" (most stable and well-tested) release channel, but you can change it within that configuration file.

## Raspberry Pi SoC/IP Core Firmware

The firmware for the peripherals on the Raspberry Pi Hardware Models is contained within the Slackware `a/kernel-firmware` package.



Simply upgrading to the latest available Slackware packages will update this firmware.

## Using Device Tree Overlays

[Device Tree Overlays](#) can be configured within the Raspberry Pi's Native Boot Loader configuration file: `/boot/platform/hwm_bw/config.txt`



Note: The Raspberry Pi's Native Boot Loader configuration is outside of the management of the Slackware OS so you need not be concerned with any automatic modifications.

## Raspberry Pi 3: Enabling Blue Tooth



This does **not apply** to Raspberry Pi 4.

The Raspberry Pi 3 requires that the serial connection be disabled in order to use Blue Tooth.

To do this you need to edit one setting:

### Elevate yourself to 'root'

```
$ su -
```

### Configure the UART setting

```
$ nano /boot/config.txt
```

Find the line: `enable_uart=1`

Change it to: `enable_uart=0`

Save the file and reboot.



## Switching to the Raspberry Pi Kernel fork

The Raspberry Pi's are only intended to run the vendor's own Linux distribution, or to run as an 'Appliance' using the Raspberry Pi Linux Kernel fork.

As a consequence, the upstream mainline Linux Kernel in which support for the Raspberry Pi is fragile and prone to breaking without notice. Slackware uses official upstream repositories/releases and has a no-patch policy. As such, the recommendation of using the vendor's own Kernel fork is far outside the scope of Slackware.

Over time the fixes and new features for the Raspberry Pi will continue to appear in the main stream Kernel. However, if you are facing instability or need some specific feature that isn't yet available in the mainline official Linux Kernel, Slackware [provides replacement Kernel packages](#) that are built from the Raspberry Pi's Linux Kernel fork.



Switching to the Raspberry Pi Kernel fork packages is **not** a recommendation, just an option for users who face instability.

## Installing extra Software

Slackware comes with a good base of software applications, but there are plenty more available in the Open Source Ecosystem.

The best way to add new software is to [use the build scripts from SlackBuilds.org](#).

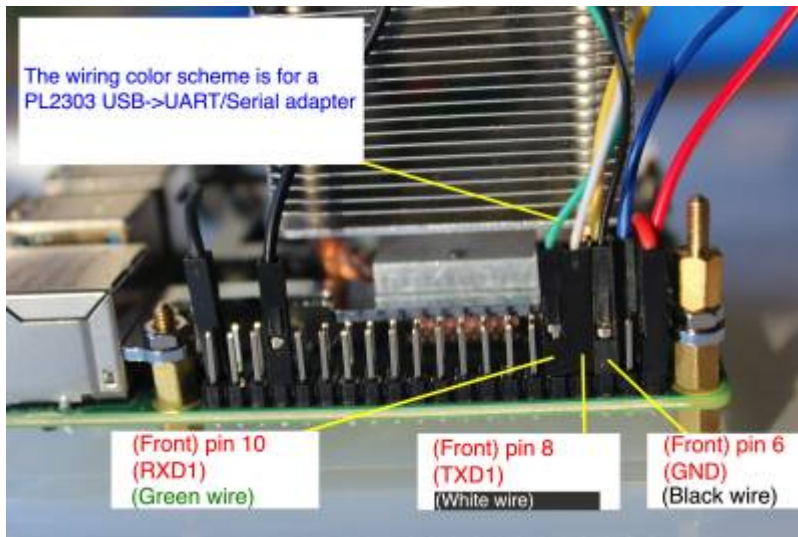
## Using the Serial/UART adapter

This documentation discusses using the Raspberry Pi without the UART/Serial console.

If you'd like to use one, there are two that have been tested.

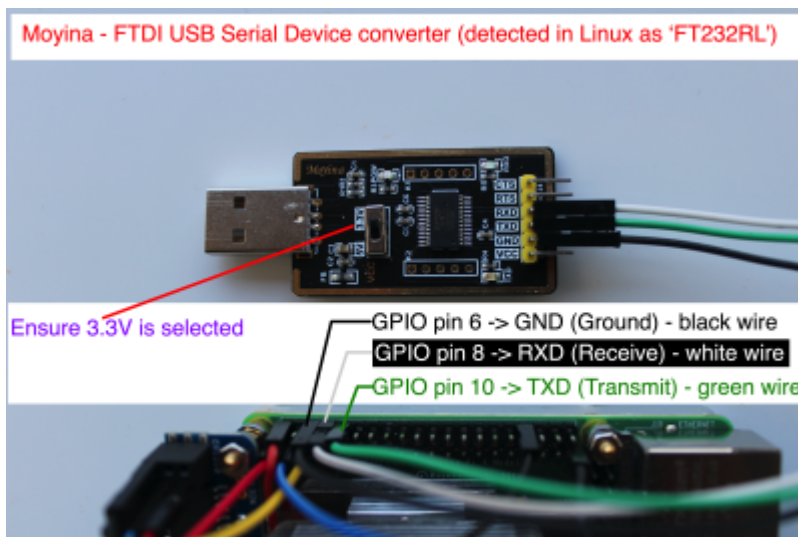
### USB Serial Device converter: Prolific Technology Inc / PL2303

This image below shows the PL2302 (the Serial adapter listed in the Hardware table at the head of this document) connected to the Raspberry Pi:



## USB Serial Device converter: FTDI / FT232RL

This is the model shown here.



## Using the USB Serial Device converter on the Linux Host Computer

Once wired up, connect the USB end of the adapter into your Linux Host Computer, and use the following command.



This assumes that there are no other similar adapters occupying `/dev/ttyUSB0`. If so, you will need to adjust the device name accordingly (e.g. perhaps `/dev/ttyUSB1`).

```
screen -T screen-256color /dev/ttyUSB0 115200
```

## Known Limitations / Bugs

Issue	Work around	Notes
General fragility/instability	<a href="#">Transition to the Raspberry Pi Kernel fork packages</a>	

## Contributing to the Slackware ARM project

There are a plethora of ARM devices on the market which requires initial R&D and continuous testing. If you'd like to help Slackware support more ARM boards, please check out [the documentation](#) explaining how to get involved.

## Supporting / Sponsoring the Slackware ARM project

Maintenance of the Slackware ARM port takes not only a lot of time, but also has financial costs such as the on-going use of electricity, Internet hosting and purchasing and maintenance of ARM hardware.

Once you find yourself enjoying using the ARM port of Slackware, please take a few moments to [show your appreciation through sponsorship](#).

## References

- [RPI GPIO exposition](#)
- [RPi4 GPIO pin layout](#) (credit: Les Pounder)
- [Adafruit RTC wiring documentation](#)

From:  
<https://docs.slackware.com/> - **SlackDocs**

Permanent link:  
[https://docs.slackware.com/slackwarearm:inst\\_sa64\\_bcm2711\\_rpi4](https://docs.slackware.com/slackwarearm:inst_sa64_bcm2711_rpi4)

Last update: **2024/04/30 14:59 (UTC)**

