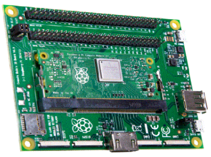
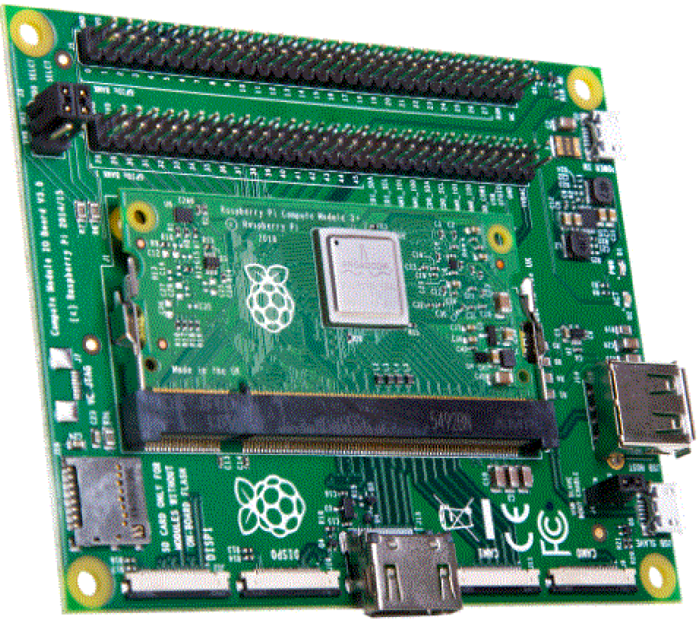


Provisioning the Raspberry Pi Compute Module User Guide

Contents

Provisioning the Raspberry Pi Compute Module



Provisioning the Raspberry Pi Compute Module (Versions 3 and 4)
Raspberry Pi Ltd
2022-07-19: githash: 94a2802-clean

Colophon

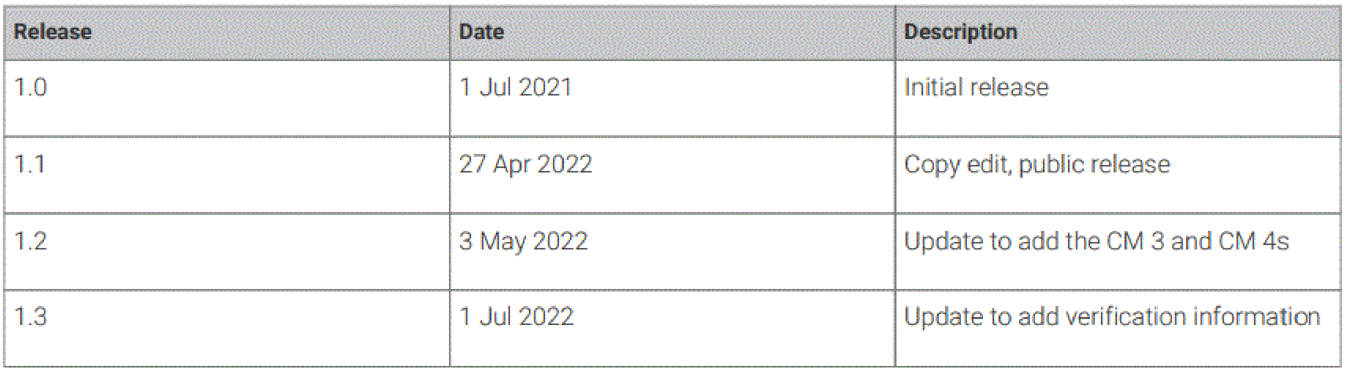
© 2020-2022 Raspberry Pi Ltd (formerly Raspberry Pi (Trading) Ltd.)
This documentation is licensed under a Creative Commons Attribution-NoDerivatives 4.0 International (CC BY-ND). build-date: 2022-07-19 build-version: githash: 94a2802-clean

Legal disclaimer notice

TECHNICAL AND RELIABILITY DATA FOR RASPBERRY PI PRODUCTS (INCLUDING DATASHEETS) AS MODIFIED FROM TIME TO TIME (“RESOURCES”) ARE PROVIDED BY RASPBERRY PI LTD (“RPL”) “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW IN NO EVENT SHALL RPL BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THE RESOURCES, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

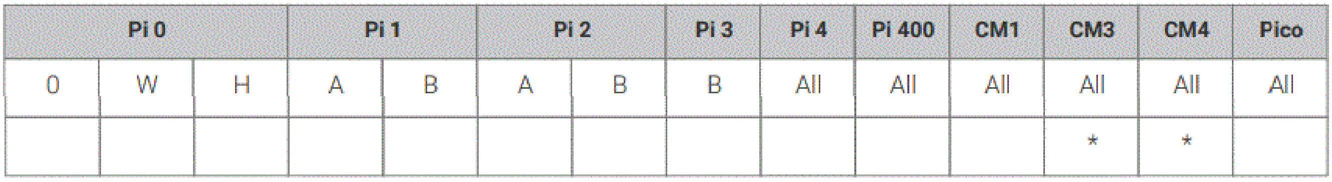
RPL reserves the right to make any enhancements, improvements, corrections or any other modifications to the RESOURCES or any products described in them at any time and without further notice. The RESOURCES are intended for skilled users with suitable levels of design knowledge. Users are solely responsible for their selection and use of the RESOURCES and any application of the products described in them. User agrees to indemnify and hold RPL harmless against all liabilities, costs, damages or other losses arising out of their use of the RESOURCES. RPL grants users permission to use the RESOURCES solely in conjunction with the Raspberry Pi products. All other use of the RESOURCES is prohibited. No licence is granted to any other RPL or other third party intellectual property right. HIGH RISK ACTIVITIES. Raspberry Pi products are not designed, manufactured or intended for use in hazardous environments requiring fail safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, weapons systems or safety-critical applications (including life support systems and other medical devices), in which the failure of the products could lead directly to death, personal injury or severe physical or environmental damage (“High Risk Activities”). RPL specifically disclaims any express or implied warranty of fitness for High Risk Activities and accepts no liability for use or inclusions of Raspberry Pi products in High Risk Activities. Raspberry Pi products are provided subject to RPL’s Standard Terms. RPL’s provision of the RESOURCES does not expand or otherwise modify RPL’s Standard Terms including but not limited to the disclaimers and warranties expressed in them.

Document version history



Release	Date	Description
1.0	1 Jul 2021	Initial release
1.1	27 Apr 2022	Copy edit, public release
1.2	3 May 2022	Update to add the CM 3 and CM 4s
1.3	1 Jul 2022	Update to add verification information

Scope of document
This document applies to the following Raspberry Pi products:



Pi 0			Pi 1		Pi 2		Pi 3	Pi 4	Pi 400	CM1	CM3	CM4	Pico
0	W	H	A	B	A	B	B	All	All	All	All	All	All
											*	*	

Introduction

The CM Provisioner is a web application designed to make programming a large number of Raspberry Pi Compute Module (CM) devices much easier and quicker. It is simple to install and simple to use. It provides an interface to a database of kernel images that can be uploaded, along with the ability to use scripts to customise various parts of the installation during the flashing process. Label printing and firmware updating is also supported. This whitepaper assumes that the Provisioner server, software version 1.5 or newer, is running on a Raspberry Pi.

How it all works

CM4

The Provisioner system needs to be installed on its own wired network; the Raspberry Pi running the server is plugged in to a switch, along with as many CM4 devices as the switch can support. Any CM4 plugged into this network will be detected by the provisioning system and automatically flashed with the user’s required firmware. The reason for having its own wired network becomes clear when you consider that any CM4 plugged into the network will be provisioned, so keeping the network separate from any live network is essential to prevent the unintentional reprogramming of devices.

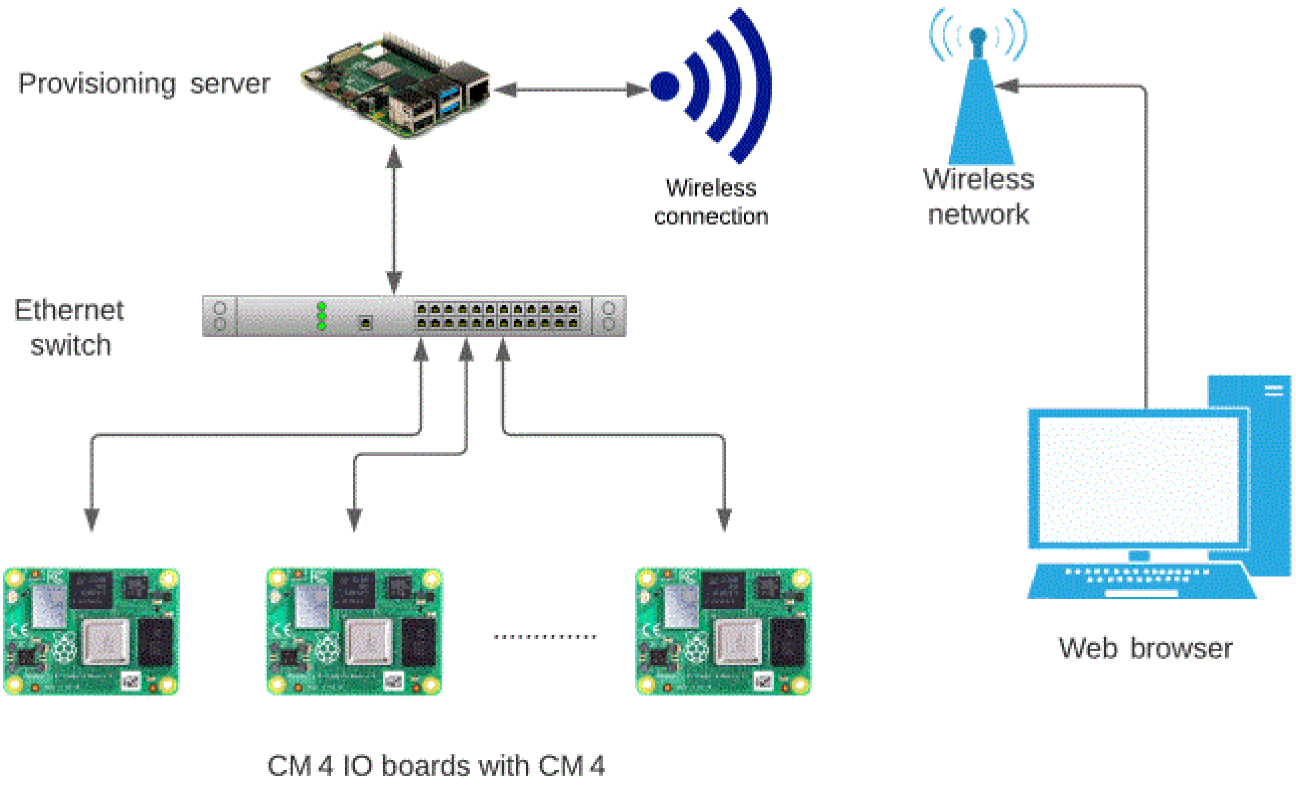


IMAGE CHANGES CM 4 IO boards with CM 4 -> CM4 IO Boards with CM4

By using a Raspberry Pi as the server, it is possible to use wired networking for the Provisioner but still allow access to external networks using wireless connectivity. This allows easy downloading of images to the server, ready for the provisioning process, and allows the Raspberry Pi to serve up the Provisioner web interface. Multiple images can be downloaded; the Provisioner keeps a database of images and makes it easy to select the appropriate image for setting up different devices.

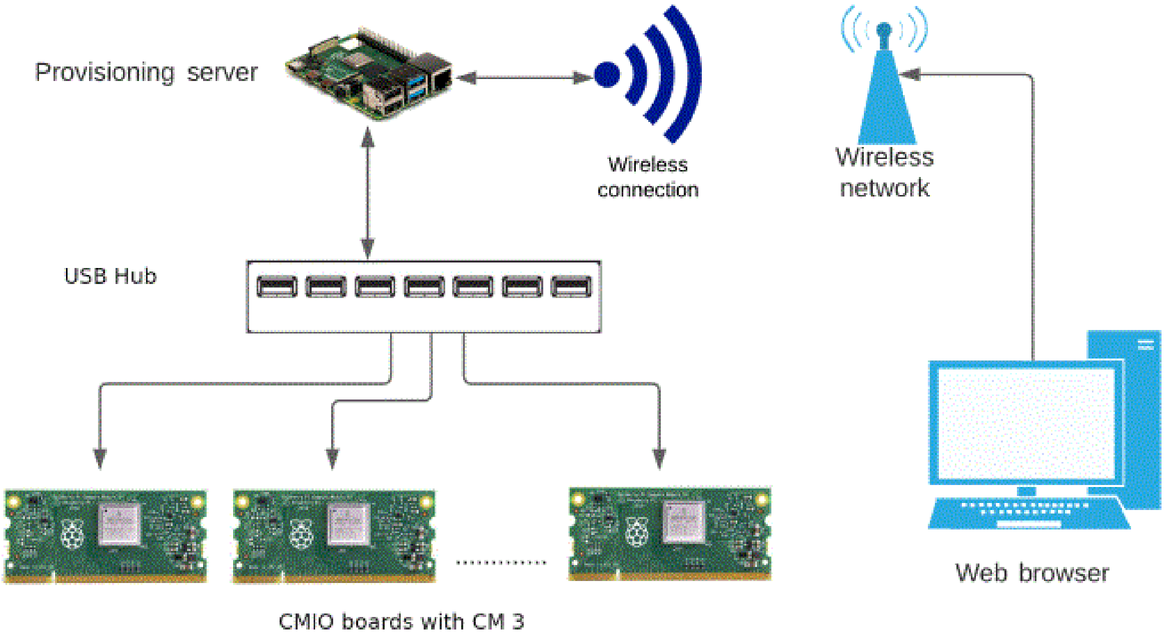
When a CM4 is attached to the network and is powered up it will try to boot, and once other options have been tried, network booting is attempted. At this point the Provisioner Dynamic Host Configuration Protocol (DHCP) system responds to the booting CM4 and provides it with a minimal bootable image that is downloaded to the CM4 then run as root. This image can program the embedded Multi-Media Card (eMMC) and run any required scripts, as instructed by the Provisioner.

More details

CM4 modules ship with a boot configuration that will try to boot from eMMC first; if that fails because the eMMC is empty, it will perform a preboot execution environment (PXE) network boot. So, with CM4 modules that have not yet been provisioned, and have an empty eMMC, a network boot will be performed by default. During a network boot on a provisioning network, a lightweight utility operating system (OS) image (actually a Linux kernel and a scriptexecute initramfs) will be served by the provisioning server to the CM4 module over the network, and this image handles the provisioning.

CM 3 and CM 4s

The CM devices based on the SODIMM connector cannot network boot, so programming is achieved over USB. Each device will need to be connected to the Provisioner. If you need to connect more than 4 devices (the number of USB ports on the Raspberry Pi), a USB hub can be used. Use good quality USB-A to Micro-USB cables, connecting from the Raspberry Pi or hub to the USB slave port of each CMIO board. All the CMIO boards will also need a power supply, and the J4 USB slave boot enable jumper should be set to enable



IMPORTANT
Do NOT connect the Ethernet port of the Pi 4. The wireless connection is used to access the management web interface.

Installation

The following instructions were correct at the time of issue. The very latest installation instructions can be found on the Provisioner GitHub page.

Installing the Provisioner web application on a Raspberry Pi

WARNING
Make sure eth0 connects to an Ethernet switch that only has the CM4 IO Boards connected. Do not connect eth0 to your office/public network, or it may 'provision' other Raspberry Pi devices in your network as well. Use the Raspberry Pi wireless connection to connect to your local network.

The Lite version of the Raspberry Pi OS is recommended as the base OS on which to install the Provisioner. For simplicity use rpi-imager, and activate the advanced settings menu (Ctrl-Shift-X) to set up the password, hostname, and wireless settings. Once the OS is installed on the Raspberry Pi, you will need to set up the Ethernet system:

1. Configure eth0 to have a static Internet Protocol (IP) address of 172.20.0.1 inside a /16 subnet (netmask 255.255.0.0) by editing the DHCP configuration:
    - sudo nano /etc/dhcpcd.conf
    - Add to the bottom of the file:

      interface eth0

      static ip_address=172.20.0.1/16
    - Reboot to allow the changes to take effect.
2. Ensure the OS installation is up to date:

   sudo apt update

   sudo apt full-upgrade
3. The Provisioner is supplied as a ready-made .deb file on the Provisioner GitHub page. Download the latest version from that page or using wget, and install it using the following command:

   sudo apt install ./cmprovision4_*_all.deb

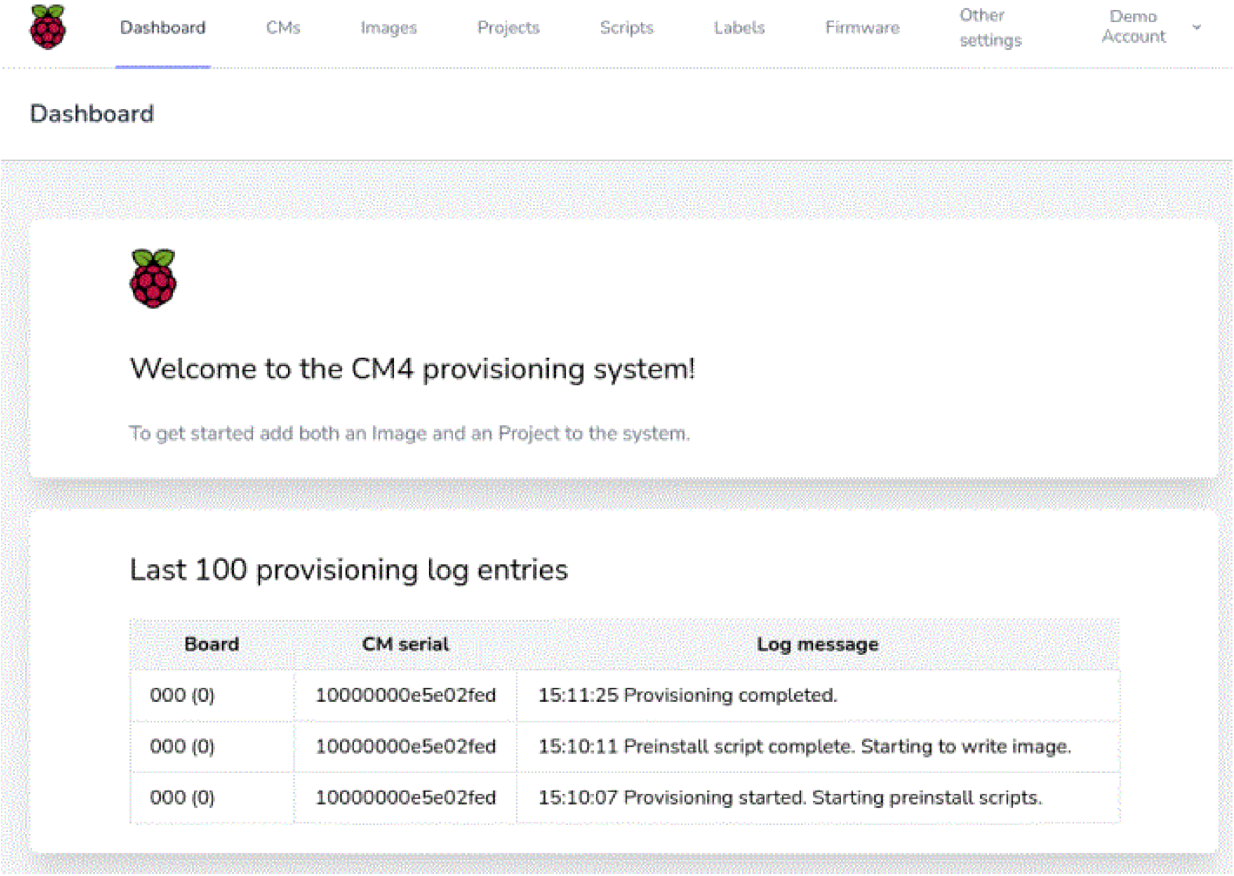
4. Set the web application username and password:

   sudo /var/lib/cmprovision/artisan auth:create-user

You can now access the web interface of the Provisioner with a web browser using the Raspberry Pi wireless IP address and the username and password entered in the previous section. Just enter the IP address in the address bar of your browser and press Enter.

Usage

When you first connect to the Provisioner web application with your web browser you will see the Dashboard screen, which will look something like this:



Dashboard

Welcome to the CM4 provisioning system!

To get started add both an Image and an Project to the system.

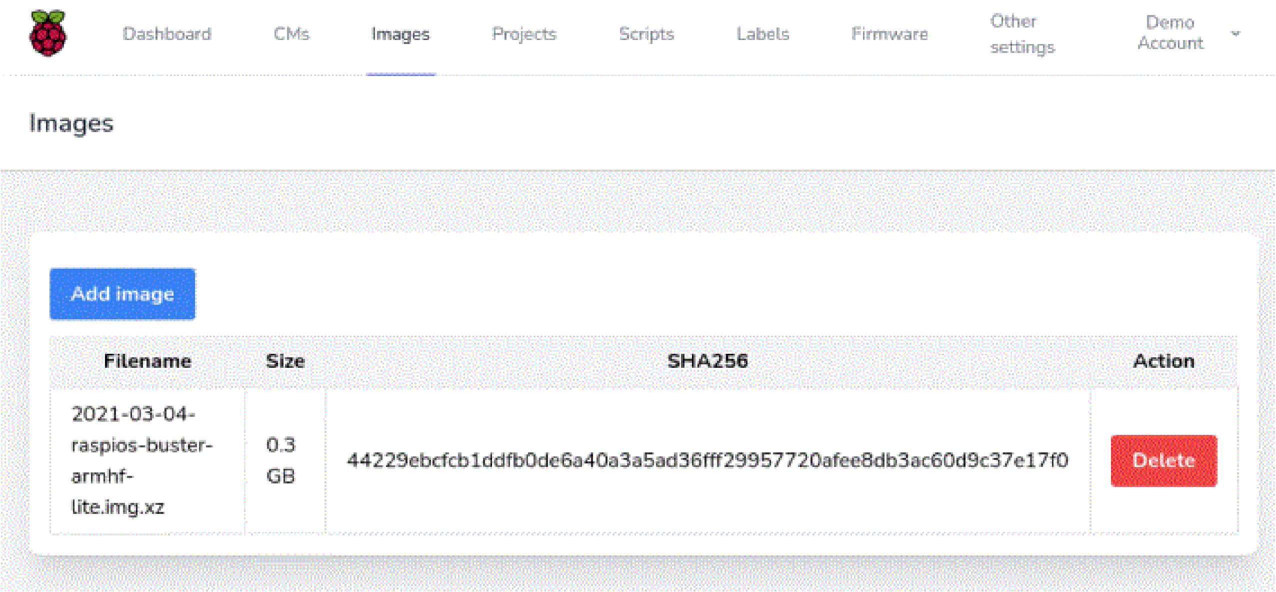
Last 100 provisioning log entries

Board	CM serial	Log message
000 (0)	10000000e5e02fed	15:11:25 Provisioning completed.
000 (0)	10000000e5e02fed	15:10:11 Preinstall script complete. Starting to write image.
000 (0)	10000000e5e02fed	15:10:07 Provisioning started. Starting preinstall scripts.

This landing page simply gives some information on the latest action performed by the Provisioner (in the example above, a single CM4 has been provisioned).

Uploading images

The first operation required when setting up is to load your image to the server, from where it can be used to provision your CM4 boards. Click the ‘Images’ menu item at the top of the web page and you should get a screen similar to the one shown below, showing a list of currently uploaded images (which will initially be empty).



Images

Filename	Size	SHA256	Action
2021-03-04-raspios-buster-armhf-lite.img.xz	0.3 GB	44229ebcfcb1ddfb0de6a40a3a5ad36fff29957720afee8db3ac60d9c37e17f0	Delete

Select the Add Image button to upload an image; you will see this screen:

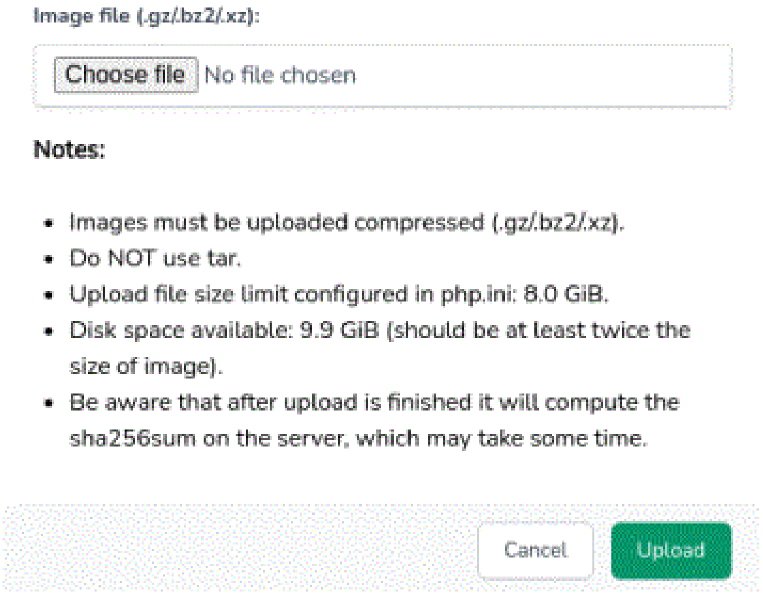


Image file (.gz/.bz2/.xz): No file chosen

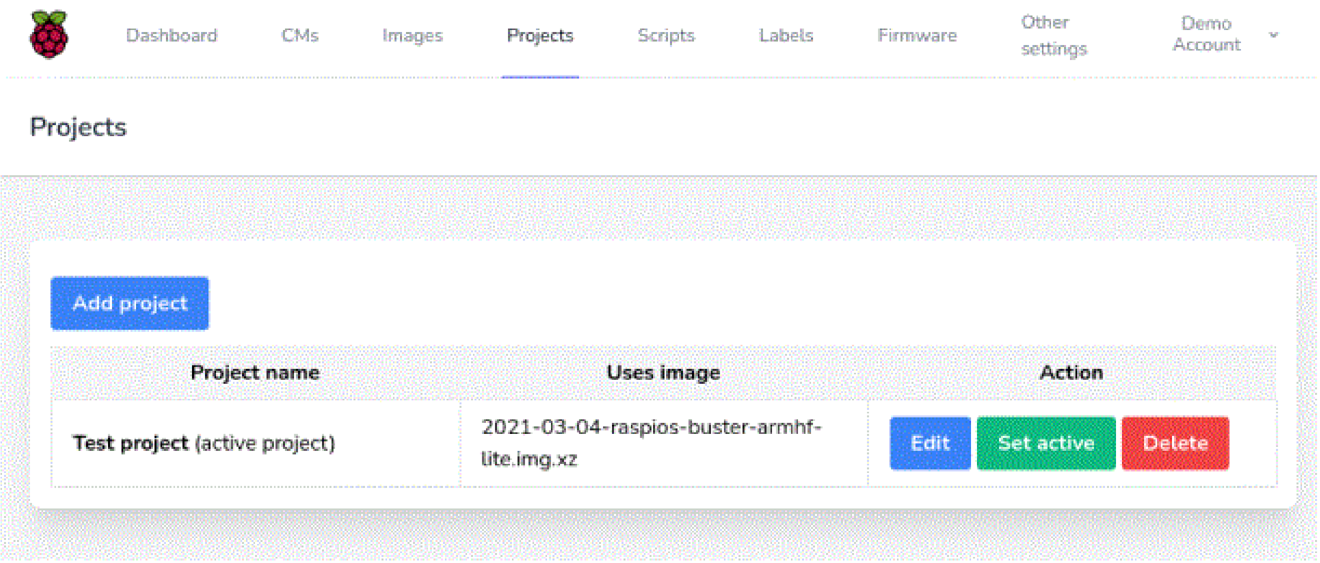
Notes:

- Images must be uploaded compressed (.gz/.bz2/.xz).
- Do NOT use tar.
- Upload file size limit configured in php.ini: 8.0 GiB.
- Disk space available: 9.9 GiB (should be at least twice the size of image).
- Be aware that after upload is finished it will compute the sha256sum on the server, which may take some time.

The image needs to be accessible on the device where the web browser is running, and in one of the image formats specified. Select the image from your machine using the standard file dialog, and click ‘Upload’. This will now copy the image from your machine to the Provisioner server running on the Raspberry Pi. This can take some time. Once the image is uploaded, you will see it on the Images page.

Adding a project

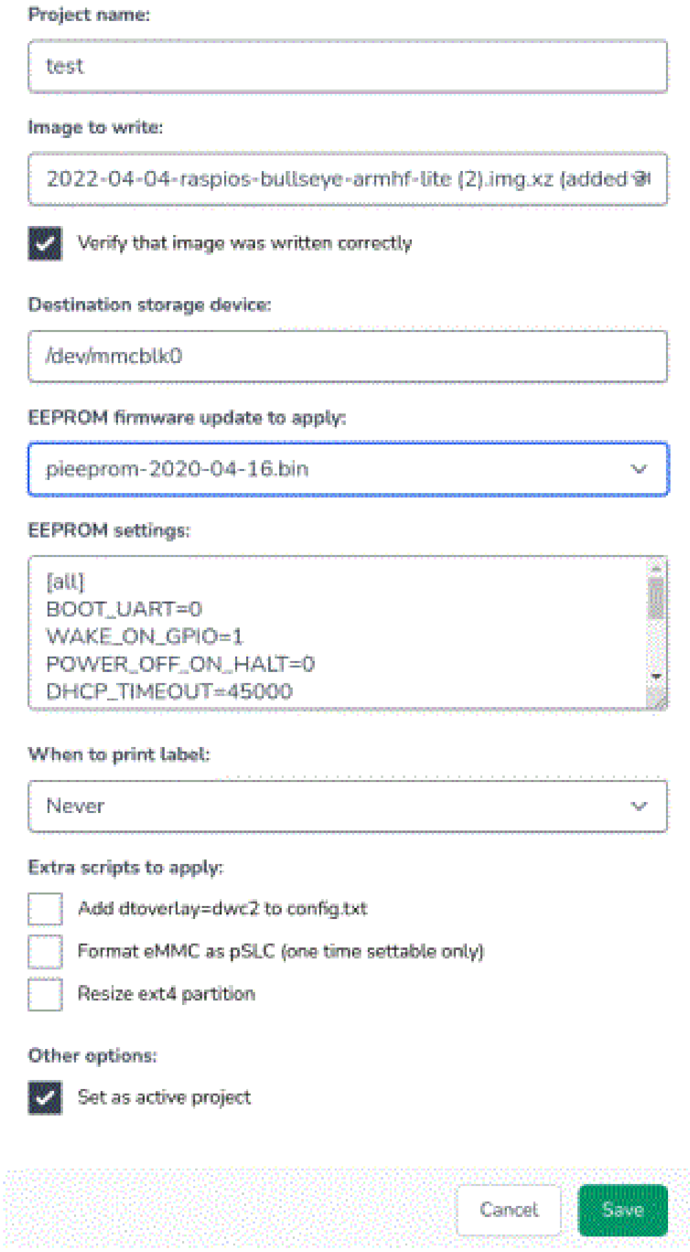
Now you need to create a project. You can specify any number of projects, and each can have a different image, set of scripts, or label. The active project is the one that is currently used for provisioning.
Click on the ‘Projects’ menu item to bring up the Projects page. The following example already has one project, called ‘Test project’, set up.



Projects

Project name	Uses image	Action
Test project (active project)	2021-03-04-raspios-buster-armhf-lite.img.xz	Edit Set active Delete

Now click on ‘Add project’ to set up a new project



Project name: test

Image to write: 2022-04-04-raspios-bullseye-armhf-lite (2).img.xz (added 3

☑ Verify that image was written correctly

Destination storage device: /dev/mmcblk0

EEPROM firmware update to apply: pieeprom-2020-04-16.bin

EEPROM settings:

```
[all]
BOOT_UART=0
WAKE_ON_GPIO=1
POWER_OFF_ON_HALT=0
DHCP_TIMEOUT=45000
```

When to print label: Never

Extra scripts to apply:

☐ Add dtoverlay=dwc2 to config.txt

☐ Format eMMC as pSLC (one time settable only)

☐ Resize ext4 partition

Other options:

☑ Set as active project

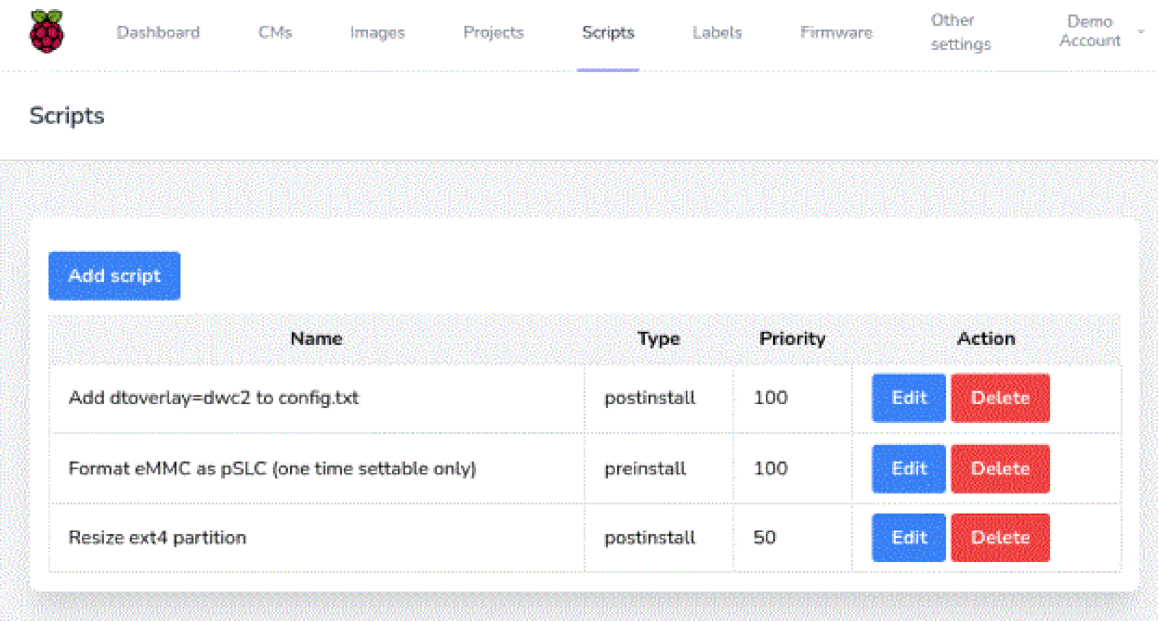
- Give the project an appropriate name, then select which image you wish this project to use from the drop-down list. You can also set a number of other parameters at this stage, but often only the image will suffice.
- If you are using v1.5 or newer of the Provisioner, then you have the option of verifying that the flashing has

completed correctly. Selecting this will read back the data from the CM device after flashing, and confirm that it matches the original image. This will add extra time to the provisioning of each device, the amount of time added will depend on the size of the image.

- If you select the firmware to install (this is optional), you also have the ability to customise that firmware with some specific configuration entries that will be merged in to the bootloader binary. The available options can be found on the Raspberry Pi website.
- Click 'Save' when you have fully defined your new project; you will return to the Projects page, and the new project will be listed. Note that only one project can be active at any one time, and you can select it from this list.

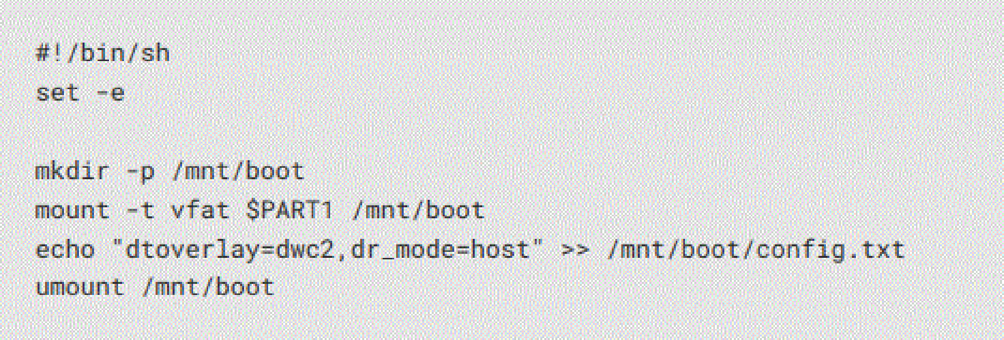
Scripts

A really useful feature of Provisioner is the ability to run scripts on the image, before or after installation. Three scripts are installed by default in the Provisioner, and can be selected when creating a new project. They are listed on the Scriptspage



Name	Type	Priority	Action
Add dtoverlay=dwc2 to config.txt	postinstall	100	Edit Delete
Format eMMC as pSLC (one time settable only)	preinstall	100	Edit Delete
Resize ext4 partition	postinstall	50	Edit Delete

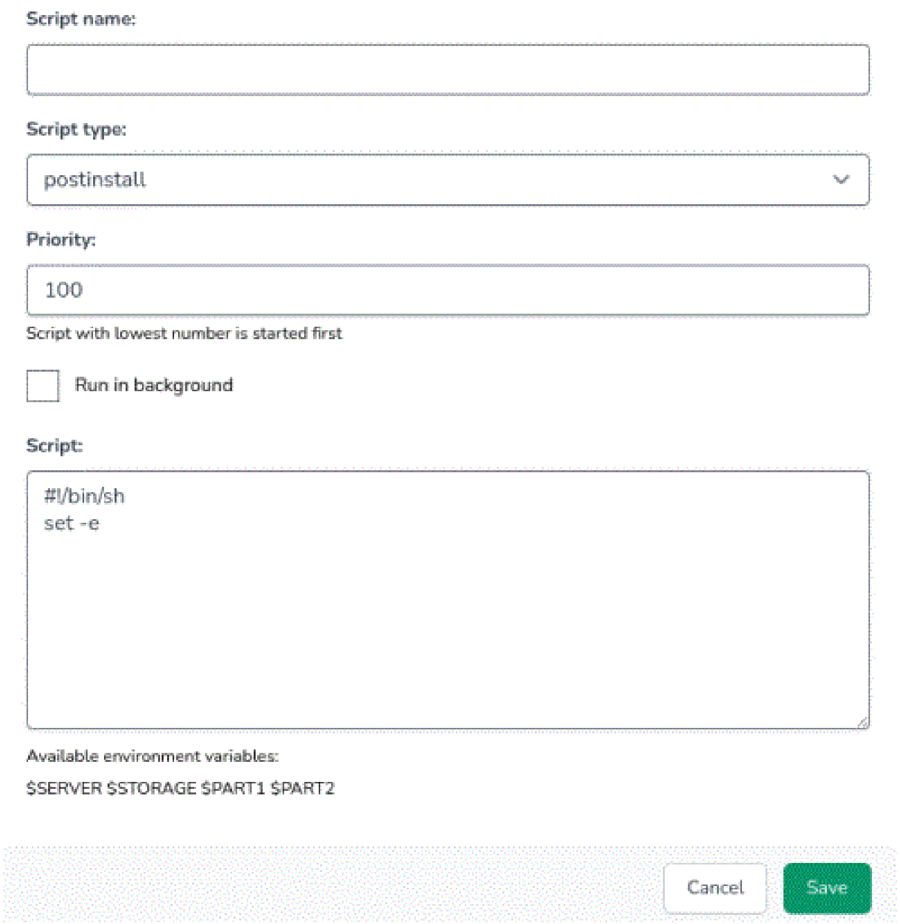
An example use of scripts might be to add custom entries to config.txt. The standard script Add dtoverlay=dwc2 to config.txt does this, using the following shell code:



```
#!/bin/sh
set -e

mkdir -p /mnt/boot
mount -t vfat $PART1 /mnt/boot
echo "dtoverlay=dwc2,dr_mode=host" >> /mnt/boot/config.txt
umount /mnt/boot
```

Click on 'Add script' to add your own customisations:



Script name:

Script type: postinstall

Priority: 100

Script with lowest number is started first

□ Run in background

Script:

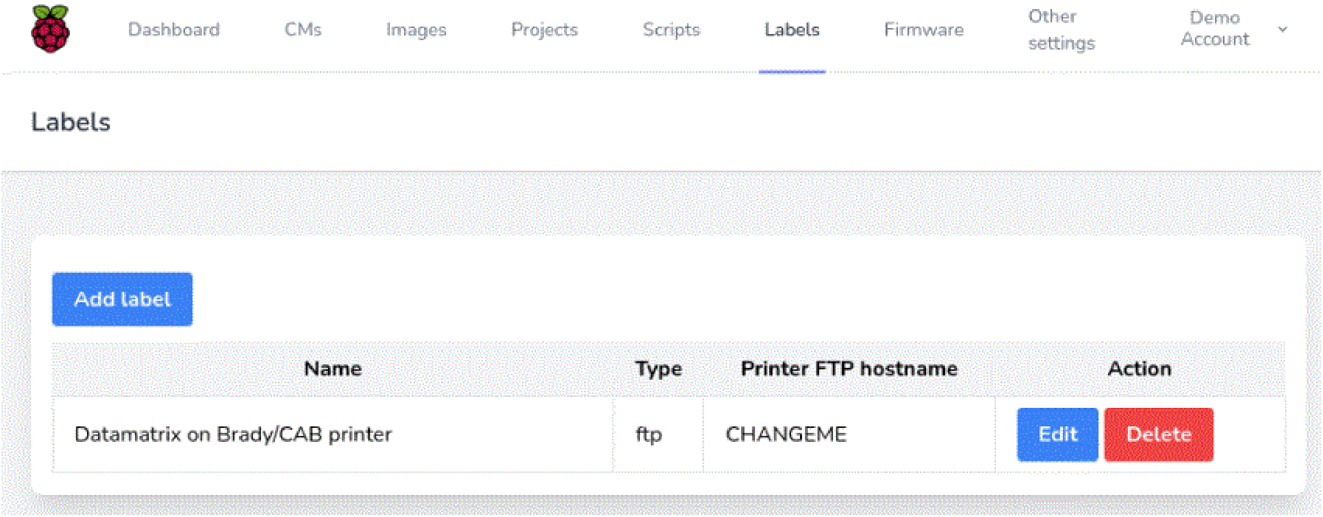
```
#!/bin/sh
set -e
```

Available environment variables:

$SERVER $STORAGE $PART1 $PART2

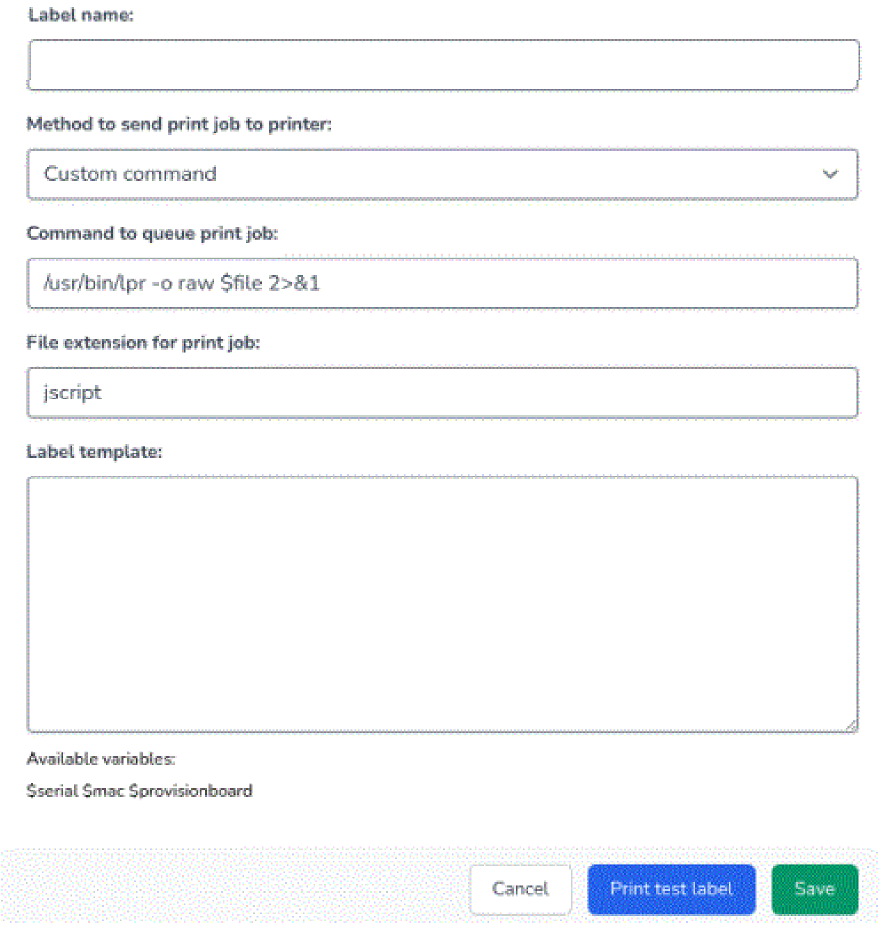
Labels

The Provisioner has the facility to print out labels for the device being provisioned. The Labels page shows all the predefined labels that can be selected during the project editing process. For example, you may wish to print out DataMatrix or quick response (QR) codes for each board provisioned, and this feature makes this very easy.



Name	Type	Printer FTP hostname	Action
Datamatrix on Brady/CAB printer	ftp	CHANGEME	Edit Delete

Click on ‘Add label’ to specify your own:



Label name:

Method to send print job to printer: Custom command

Command to queue print job: /usr/bin/lpr -o raw $file 2>&1

File extension for print job: jscript

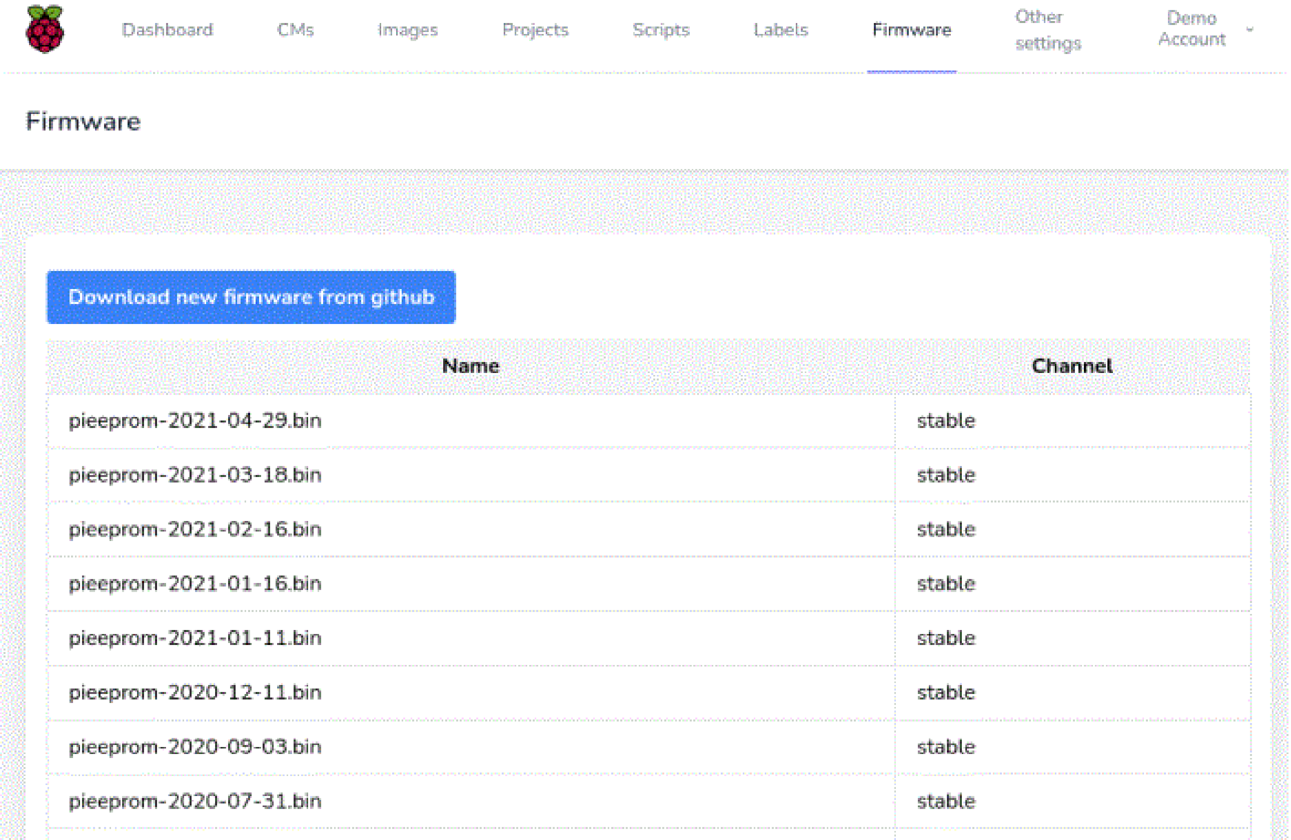
Label template:

Available variables:

$serial $mac $provisionboard

Firmware

The Provisioner provides the ability to specify which version of the bootloader firmware you wish to install on the CM4. On the Firmware page there is a list of all the possible options, but the most recent one is usually the best.



Firmware

Name	Channel
pieeprom-2021-04-29.bin	stable
pieeprom-2021-03-18.bin	stable
pieeprom-2021-02-16.bin	stable
pieeprom-2021-01-16.bin	stable
pieeprom-2021-01-11.bin	stable
pieeprom-2020-12-11.bin	stable
pieeprom-2020-09-03.bin	stable
pieeprom-2020-07-31.bin	stable

To update the list with the latest versions of the bootloader, click on the ‘Download new firmware from github’ button.

Possible problems

Out-of-date bootloader firmware

If your CM4 is not detected by the Provisioner system when it is plugged in, it is possible that the bootloader firmware is out of date. Note that all CM4 devices manufactured since February 2021 have the correct bootloader installed at the factory, so this will only happen with devices that were manufactured before that date.

Already programmed eMMC

If the CM4 module already has boot files in the eMMC from a previous provisioning attempt then it will boot from the eMMC and the network boot required for provisioning will not occur.
If you do wish to reprovision a CM4 module, you will need to:

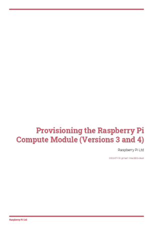
- Attach a USB cable between the provisioning server and the micro USB port of the CM4 IO Board (labelled ‘USB slave’).
- Put a jumper on the CM4 IO Board (J2, ‘Fit jumper to disable eMMC boot’).

This will cause the CM4 module to perform a USB boot, in which case the provisioning server will transfer the files of the utility OS over USB.
After the utility OS has booted, it will contact the provisioning server over Ethernet to receive further instructions, and download additional files (e.g. the OS image to be written to eMMC) as usual. So, an Ethernet connection in addition to the USB cable is still necessary.

Spanning Tree Protocol (STP) on managed Ethernet switches

PXE booting will not work correctly if STP is enabled on a managed Ethernet switch. This can be the default on some switches (e.g. Cisco), and if that is the case it will need to be disabled for the provisioning process to work correctly.
Raspberry Pi is a trademark of the Raspberry Pi Foundation
Raspberry Pi Ltd

Documents / Resources

 Provisioning the Raspberry Pi Compute Module (Versions 3 and 4) Raspberry Pi Ltd Raspberry Pi Ltd	Raspberry Pi Provisioning the Raspberry Pi Compute Module [pdf] User Guide Provisioning the Raspberry Pi Compute Module, Provisioning, the Raspberry Pi Compute Module, Compute Module

References

- Creative Commons — Attribution-NoDerivatives 4.0 International — CC BY-ND 4.0
- GitHub - raspberrypi/cmprovision: Provisioning system for CM4 products
- Releases · raspberrypi/cmprovision · GitHub
- GitHub - raspberrypi/scriptexecutor: Simple buildroot based system for executing a remote script for manufacture programming
- Terms and Conditions of Sale - Raspberry Pi
- Raspberry Pi Documentation - Raspberry Pi hardware