A brief guide to migrating from Bullseye to Bookworm

Raspberry Pi Ltd

2024-08-15: githash: 656a2b0-clean

Colophon

build-date: 2024-08-15
build-version: githash: 656a2b0-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

NOTE

This is a draft document, and is not yet finalised. It is intended to be technically complete, but it has not yet been edited. If you notice any issues, please let us know by email to applications@raspberrypi.com so that we can improve it for you and other customers.

Release	Date	Description
1.0	23 Oct 2023	• Initial release

Scope of document

This document applies to the following Raspberry Pi products:

Pi Zero			Pi 1				Pi 2		Pi 3			Pi 4	Pi 5	Pi 400	CM1	CM3	CM4	Pico
Zero	W	H	A	B	A+	B+	A	B	B	A+	B+	All	All	All	All	All	All	All
*	*	*	*	*	*	*	*	*	*	*	*	*		*	*	*	*	

Introduction

The Raspberry Pi OS is provided by Raspberry Pi Ltd to run on the Raspberry Pi Ltd devices indicated above. The Raspberry Pi OS is a Linux OS based on the Debian distribution. This distribution is updated approximately every two years, and the Raspberry Pi OS follows these updates, usually a few months after the main Debian release. Debian releases are given codenames: the two most recent at the time of writing have been called Bullseye and Bookworm. While Raspberry Pi Ltd continually provides incremental updates between major OS releases, major changes can be introduced at the point of these named release changes. In addition, Raspberry Pi Ltd introduces further changes to the system during these named updates, over and above those provided by Debian.

This document describes some of the major changes made between the Raspberry Pi OS Bullseye and Bookworm releases, providing examples where procedures may need to be changed. It does not cover all possible differences between the OS versions.

Many of the features now used by Bookworm were available in the Buster release. There is a whitepaper describing the move to Bullseye from Buster which should be read in conjunction with this document, as much of the advice presented there is equally applicable to Bookworm.

Terminology

Legacy graphics stack: A graphics stack wholly implemented in the VideoCore firmware blob, exposed by a Linux framebuffer driver. This is what has been used on the majority of Raspberry Pi Ltd Pi devices since launch, but has been completely replaced by KMS/DRM in Bookworm.

KMS (vc4-kms-v3d): The full Kernel Mode Setting driver. Controls the entire display process, including talking to the hardware directly with no firmware interaction. Used by default on Bullseye and Bookworm.

DRM: Direct rendering manager, a subsystem of the Linux kernel used to communicate with graphics processing units (GPUs). Used in partnership with KMS.

Wayland: A communication protocol that specifies the communication between a display server and its clients.

wlroots: A compositor framework providing backends that abstract the underlying display and input hardware, including KMS/DRM, libinput, Wayland, X11, and headless backends.

Wayfire: Wayfire is a Wayland compositor based on wlroots.

Main differences between Bullseye and Bookworm

As with all Debian releases, a large number of third-party applications will have been updated to more recent versions. It should be noted that although these applications are more recent, they will not necessarily be the very latest versions. Debian is conservative when it comes to software releases, prioritising stability over having the very latest features. Therefore, Debian tends to lag behind the very latest releases in order to ensure a reliable system. This document will not cover how to get more recent versions of software than those installed by `apt` in Bookworm.

Some major architectural changes have been made to the Raspberry Pi OS with the Bookworm release. The main ones are listed here, and are described in more detail in the appropriate subsections:

- Removal of legacy and FKMS graphics stack, to enable move to KMS.
- The open-source libcamera system is now the only camera option.
- Deprecation of Multi-Media Abstraction Layer (MMAL) and OpenMAX, and move to Video4Linux (V4L2) for codec and video pipeline support.
- Support for the DispmanX API has been dropped.

Many of these changes are intended to provide a more open system, moving away from code based on closed-source firmware to open-source application programming interfaces (APIs). This offers users better access to underlying hardware that was previously only usable via proprietary APIs. Where previously many parts of the hardware were controlled from the VideoCore processor, which is not open source, these hardware blocks are now controllable from the ARM cores via standard Linux APIs. Another very important side effect of these changes is that all the new libraries are available when running a 64-bit OS, whereas many of the older ones will not work in that environment.

In addition to the architectural changes, several mechanisms for providing standard features are now implemented using different software stacks. Alongside smaller optimisations, some major changes have been introduced:

- Desktop graphics backend has moved from X11 to Wayland.
- Networking configuration has moved from DHCP to NetworkManager.
- Audio support has moved from PulseAudio to PipeWire.

In summary, the changes can be described as follows:

Previous	New	Comments
Bullseye	Bookworm	Many upstream packages from Debian have been updated.
Legacy/FKMS graphics stack/KMS	KMS only	
Legacy firmware camera stack	libcamera	Python picamera support has been superceded by picamera2.
MMAL/OpenMAX	V4L2	
DispmanX	DRM	
X11	Wayland	
PulseAudio	PipeWire	Also provides access to JACK, ALSA and GStreamer.
DHCPCD	NetworkManager	Also provides VPN, hotspot and hidden network support.

A summary of the new features in Bookworm is available from Debian. == KMS

Kernel Mode Setting is a standard Linux API for controlling graphics output. Combined with the DRM library it controls all output to display devices, such as HDMI and composite monitors, LCD panels, etc. In Bullseye, the KMS graphics driver was used, but could be replaced by the earlier legacy or FKMS graphics stacks. In Bookworm only KMS is available. The KMS system controls the HVS hardware directly, and there is no firmware involvement.

The KMS system provides a standard way of setting display orientation and a number of other features:

- The ability to write your own DSI (LCD panel interface) drivers
- Support for HDMI hotplugging.
- No binary blobs of closed-source graphics code.
- Multiple displays are fully integrated, including mixing HDMI, composite, DSI, and Display Parallel Interface (DPI), with up to six displays possible when using the Raspberry Pi 5, and three when using the Raspberry Pi 4.
- Standard API to control display modes.

The sister document A brief guide to migrating from Buster to Bullseye describes many of the KMS features and how they have replaced previous mechanisms for display control.

libcamera

While the new libcamera API has been available in earlier versions of the Raspberry Pi OS, on Bookworm it is now the default and only available camera system (the legacy camera stack does not work in Bookworm). Libcamera is a new camera API for Linux and replaces the custom and closed-source camera stack from earlier Raspberry Pi Ltd systems. This means much better access to the camera hardware than was previously available, and adds some very useful new features, but does remove some less-used options.

TIP

Why libcamera? Libcamera is a new API developed for Linux that is entirely open source. Raspberry Pi Ltd has collaborated with the libcamera developers to make sure it provides almost everything that the previous legacy camera stack does, but with the advantage of being entirely open source, easy to tune, and easy to use. Applications written to the libcamera specification will work on any device that supports it and are not limited to Raspberry Pi Ltd devices.

A significant difference between the legacy stack and libcamera is the removal of the `raspistill`, `raspivid`, `raspistillyuv`, and `raspiyuv` applications, and their replacement with libcamera-based alternatives. Raspberry Pi Ltd has gone to great lengths to replicate the command line features of these original applications, and for many people this will mean simply changing the name of the application used as per the following table. The most noticeable difference in the applications is the way preview windows are displayed. On {bookworm} these now appear in desktop windows (if you are using a desktop), rather than being superimposed over the top of the desktop. If you are not using a desktop, the preview uses the whole display.

WARNING

Not all of the short-form versions of the command line options are available in libcamera apps. Use `--help` with the required app to get a list of all the available libcamera commands for that application.

Please read the sister whitepaper A brief guide to migrating from Buster to Bullseye for details on using the libcamera system when migrating from the legacy camera stack.

Video pipeline

Prior to Bullseye, all video codecs were handled in firmware via the proprietary MMAL API. This was used in conjunction with the more open OpenMAX API, with MMAL providing an easier way to access the underlying hardware features. However, the MMAL API is only used on Raspberry Pi Ltd devices, so code written on other devices is not immediately compatible. MMAL and OpenMAX are not 64-bit-friendly, so do not work very well, if at all, when used on a 64-bit system.

In Bookworm, these APIs have been removed in favour of the standard V4L2 Linux API. This means better code compatibility and immediate compatibility with 64-bit systems.

Code that uses MMAL or OpenMAX will no longer work on Bookworm, so will need to be rewritten to use V4L2. Examples of this can be found in the `libcamera-apps` GitHub repository, where it is used to access the H264 encoder hardware. For example, the MMAL/IL `video_render` component will no longer work, nor will any calls made directly to `dispmanx_` functions.

OMXPlayer is no longer supported, and for video playback you should use the VLC application. There is no command-line compatibility between these applications: see the VLC documentation for details on usage.

When using VLC on the desktop, the output will be displayed in a window unless full-screen is selected, unlike OMXPlayer, which does not use desktop windows and simply superimposes the video over the desktop.

DispmanX and DRM

DispmanX is a legacy API used to drive the compositing and video hardware on Raspberry Pi devices. Usually, end users would not use this API directly, but it has been used in some applications to provide high-speed access to the display subsystems. On Bookworm the DispmanX API has been removed, and will no longer work. The API has been replaced by the standard Linux API Direct Rendering Manager (DRM). DRM provides many of the features of the DispmanX API. Any software using DispmanX directly will need to move to the DRM API to work on Bookworm.

WARNING

DRM has a concept of only allowing one master to control the display for security and concurrency reasons. When running a desktop, the Wayland compositor will be the DRM master. Programs that try to open the DRM API as master will therefore not work if the desktop is already running. You either boot to a console or bring up a console using CTRL-ALT Fn (the actual Fn key will depend on setup: for example, CTRL-ALT F1 may bring up a console; and CTRL-ALT F7 may return to the desktop).

For applications that require writing to the whole screen, you will need to run from a console and make your application the DRM master; alternatively you can run full-screen through the windowing system. This is not as efficient as being the DRM master, as this will go through the Wayland compositing system rather than direct to the display, as would happen if the application was the DRM master.

More reading

DRM is a complex API, and there are several useful documents and example applications available online.

- https://en.wikipedia.org/wiki/Direct_Rendering_Manager
- https://landley.net/kdocs/htmldocs/drm.html
- https://events.static.linuxfound.org/sites/events/files/slides/brezillon-drm-kms.pdf
- https://github.com/ascent12/drm_doc
- https://github.com/girish2k44/drmmodeset

Wayland

Wayland is a communication protocol that specifies the communication between a display server and its clients, as well as a C library implementation of that protocol. A display server using the Wayland protocol is called a Wayland compositor because it also performs the task of a compositing window manager. The compositor used in Raspberry Pi OS Bookworm is called `Wayfire`, which is itself based on `wlroots`, a graphics backend that abstracts the underlying display and input hardware.

This suite of software, which shall be referred to as Wayland, replaces the X11 system which has been used on all previous Raspberry Pi OS releases.

NOTE

Why move from X11? Wayland is much better suited to modern GPU architectures than X11, which started development in 1984, and is now rather dated. Many Linux distributions have now started moving to Wayland-based systems.

The consequences of moving to Wayland are usually not noticed. The Raspberry Pi OS desktop should appear unchanged, although there are many changes beneath the surface. Applications that use standard graphics toolkits, like GTK and Qt, are now Wayland-compliant. They can detect when an application using them is running in a Wayland environment, then route all graphics calls over the Wayland protocol, running as native Wayland applications. Most applications pre-installed as part of the Raspberry Pi Desktop use one of these toolkits, and so now run as Wayland applications.

For those applications that are not Wayland compliant, Bookworm includes `XWayland`, which is an X-server running on top of Wayland. Most X11 applications should work fine when running over `XWayland`, although there may be some slight inconsistencies in window decorations.

NOTE

Wayland is only the default on Raspberry Pi 4 and 5. The performance of Wayfire on earlier platforms is still being optimised, so for now they will continue to run the old X11 display server and the Openbox window manager. At some point, these platforms will also be switched to Wayfire.

Can I still use X11

Yes, you can switch back to X11 using `raspi-config`. However, Wayland provides a faster, smoother, and tear-free experience in comparison.

What about VNC?

VNC (Virtual Network Computing) is a graphical desktop-sharing system that allows you to remote into another computer and use the desktop as if it were local. With the move to Wayland, a new VNC backend is required to do the required display and input device capturing. For Bookworm we currently recommend WayVNC on the server side.

At the time of writing, RealVNC, which has been provided with previous versions of Raspberry Pi OS does not work with Wayland and WayVNC. Our current recommendation for a VNC client is TigerVNC, which works well with WayVNC.

Audio changes

The software that provides audio services has changed from `Pulseaudio` to `Pipewire`. This provides better support for audio accompanying video, by reducing latency. It also improves the management of Bluetooth audio devices, remembering which ones were in use at power-down, and automatically reconnecting them at boot. Finally, it is designed to operate better in the more secure Wayland environment.

`PipeWire` takes over the role of `PulseAudio` and `JACK` from previous Raspberry Pi OS releases. It provides seamless support for `PulseAudio`, `JACK`, `ALSA`, and `GStreamer` applications.

Basic information and documentation are available from PipeWire for those who require extra support.

Under normal usage, there should be no noticeable differences when comparing the audio features of Bullseye and Bookworm.

Networking changes

In Bullseye and earlier, the Dynamic Host Configuration Protocol Client Daemon (dhcpcd) implemented the network management protocol used when dynamically assigning an IP address to the device on the network, so the device can communicate using IP. In Bookworm it has been replaced by NetworkManager, which is now used as standard in most Linux distributions.

NetworkManager does everything dhcpcd does, but adds extra functionality. This includes the ability to connect to hidden wireless networks, to connect to virtual private networks (VPNs), and to use a Raspberry Pi as a wireless hotspot.

The networking plugin on the taskbar looks almost identical to that which controlled dhcpcd in Bullseye, but now has an `Advanced Options` item at the bottom which allows access to the extra functionality offered by NetworkManager.

NetworkManager configuration files are stored in `/etc/NetworkManager`. NetworkManager uses a plugin approach, and on Bookworm one of those plugins is `ifupdown`, which reads configuration information for the network from `/etc/networks`. However, it regards `/etc/networks` as read-only, and any configuration changes are written via a plugin called `keyfile` to `/etc/NetworkManager/system-connections`.

`nmcli` is a command-line utility for controlling all NetworkManager features. There is a very comprehensive manual page which can be accessed using `man nmcli` from the Bookworm command line. For example, if you make changes to the /etc/network settings, you will need to run `nmcli con reload` to force NetworkManager to scan for the changes.

More information on the plugin system and NetworkManager is available from the project site.

General application issues

This section describes any changes for commonly used applications that may be needed to work successfully on Bookworm.

Python package management

In Bookworm, Python packaging now uses the PEP 668 standard. This moves the responsibility for Python package management from `pip` to the Raspberry Pi OS package manager `apt`.

NOTE

The Python Foundation has made information on the change from `pip` to `apt` available, including the reasoning behind it.

This means that for Debian-supported packages you should now use `apt` to install those packages.

For example, if you want to install the `mutagen` package, previously you would use `pip` as follows:

```
sudo pip3 install mutagen
```

On Bookworm you would now use `apt`:

```
sudo apt install python3-mutagen
```

However, this will only work for packages that are supported in the Debian repo. For other packages, the best approach is to use a https://docs.python.org/3/library/venv.html.[virtual environment (venv]).

The following commands create a folder for the `venv`; create the `venv`; activate it; then install a package into it.

```
mkdir myvenv
python3 -m venv myvenv/
cd myvenc/bin
. activate
pip3 install <my package>
```

Much more detail on Python on Raspberry Pi is available in the Raspberry Pi documentation.

LibreOffice

The default installation of LibreOffice uses XWayland, which has some rendering issues on Bookworm. Installing the GTK3 plugin for LibreOffice forces LibreOffice to use a GTK3 backend, which uses a dedicated Wayland backend and improves performance considerably.

NOTE

libreoffice-gtk3 is now installed by default on Raspberry Pi OS.

```
sudo install libreoffice
sudo install libreoffice-gtk3
```

Wireless setup on a headless system

On Bullseye, a wpa_suplicant.conf file containing wireless credentials could be placed in the boot folder on a newly imaged SD card. This would set up the wireless connection automatically on first boot, without user interaction.

In Bookworm this no longer works, and has been replaced by an updated configuration screen in Raspberry Pi Imager.

Run Raspberry Pi Imager and select the OS you wish to install. Click the cogwheel Configuration icon. Enable wireless settings in the new window, and fill in the required details. Now create your SD card as usual.

If you are creating many SD cards using this process, to save typing the wireless details each time you can set the Image Customisation option at the top of the window to "to always use".

Screen configuration

Command-line screen configuration on Bullseye was handled by an application called `xrandr`. On Bookworm this has moved to a Wayland-based application called `wlr-randr`. Functionality is very similar between the two.

`xrandr` will still work, using the XWayland system to convert the X11 protocol to Wayland. However, its functionality will be limited in comparison to `wlr-rand`.

On-screen keyboards

On-screen keyboards based on X will no longer work with the Wayland backend. There is a Wayland-based on-screen keyboard called "wvkbd" which works well.

```
sudo apt install wvkbd
```

The instructions for use can be found on the developer's GitHub repository.

Headless setup options

On Buster it was possible to add files to the boot folder on a newly imaged SD card, to enable SSH and wireless networking before the SD card was used for the first time. This is extremely useful when setting up a lot of SD cards on a production line, where booting a device to set these options manually takes much too long.

In Bookworm the `ssh` option is maintained, but the move to NetworkManager means the `wp_supplicant` option has changed.

The first option is to use Raspberry Pi Imager, which now has options to set up wireless networking during the imaging process. However, this may not be convenient when programming devices on a production line.

tvservice

The `tvservice` application is deprecated on KMS-based systems. This application is often used for getting information about the display system, for example modes and EDID. These and other functions are now handled as follows:

tvservice option	Action	Replacement command
-a, --audio	Get supported audio information	edid-decode /sys/class/drm/card1-HDMI-A-1/edid[1]
-d, --dumpedid <filename>	Dump EDID information to file	cp /sys/class/drm/card1-HDMI-A-1/edid <filename> [1]
-l, --list	List all attached devices	kmsprint
-m, --modes=GROUP	Get supported modes for GROUP (CEA, DMT)	kmsprint -l
-s, --status	Get HDMI status	kmsprint
-o, --off	Turn display off	wlr-randr --output HDMI-A-1 --off [1][2]
-p, --preferred	Power on display	wlr-randr --output HDMI-A-1 --on [1][2]

[1] The card number or HDMI reference may vary [2] Works when used under Wayland/desktop, not from full-screen consoles.

Miscellaneous hints and tips

The boot folder

The `boot` folder in previous versions of Raspberry Pi OS contained all the required firmware files to boot the system. In Bookworm many of the firmware files have moved to a subfolder of the `boot` folder, called `firmware`. The `boot` folder now contains `initramfs` images for the various Raspberry Pi models.

Desktop keyboard shortcuts

The Wayfire system provides a number of keyboard shortcuts for window management.

The default settings are as follows:

Keystroke	Action
Alt+TAB	Select window
Alt+ESC	Fast-switch between windows
Ctrl+Alt+Left/Right	Half screen app
Ctrl+Alt+Up	Maximise
Ctrl+Alt+Down	Un-maximise
Super+mouse scroll	Zoom
Super+i	Invert display colours
Super+Num7 or Home	Move window to top-left quadrant
Super+Num9 or PgUp	Move window to top-right quadrant
Super+Num1 or End	Move window to bottom-left quadrant
Super+Num3 or PgDn	Move window to bottom-right quadrant
Super+Num8 or Up	Move window to top half of desktop
Super+Num2 or Down	Move window to bottom half of desktop
Alt+Ctrl+mouse press and move	Rotate between desktops
Alt+Ctrl+W	Network menu
Alt+Ctrl+B	Bluetooth menu
Alt+Ctrl+M	Screen magnifier
Alt+Ctrl+Backspace	Logout

All key bindings can be changed by editing settings in `~/.config/wayfire.ini`. Note that system defaults will be overwritten by entries in `wayfire.ini`. To retain any required default settings, copy the appropriate section from `~/etc/wayfire/default.ini`. A list of settings can be found on the Wayfire GitHub repository.

Desktop window animations

By default under Wayfire, when windows are opened and closed they zoom in and out of existence. There are several

options available for the animation of these desktop windows, and these can be changed by updating the Wayfire configuration file as described in the previous section.

To turn off animation, edit the file and add a section to the file as follows:

```
[animate]
close_animation=none
open_animation=none
```

The options you can use are: `none`, `fade`, `zoom` (default on Raspberry Pi OS), and `fire`.

Virtual desktops

Three virtual desktops are provided by default. The specific number can be changed by editing the `wayfire.ini` files, and adding the following section (in this example the number is set to five).

```
[core]
vwidth=5
```

Autohide the menu bar

To enable the menu bar autohide feature, add the following to the `~/.config/wf-panel-pi.ini` file:

```
autohide=1
```

Screensavers

Although software such as `XScreensaver` works reasonably well under Wayland, there are options in Wayfire to provide similar features. Wayfire uses the `Idle` plugin to provide screensaver functionality, and the parameters can again be set by editing the `wayfire.ini` file.

In the following example, the timeouts have been set very short to demonstrate the operation.

```
[idle]
cube_max_zoom = 1.500000
cube_rotate_speed = 1.000000
cube_zoom_speed = 1000
disable_initially = false
disable_on_fullscreen = false
dpms_timeout = 120
screensaver_timeout = 60
```

The screensaver_timeout defines the length of inactivity (in seconds) before the Cube screensaver starts, and the dpms_timeout determines how long to wait before the monitor is turned off.

Screen capture

By default, in Bookworm the `PrtScn/SysRq` key will capture an image of the entire screen and save it into the `Pictures` folder.

The underlying command-line application that does this is called `grim`, which is an application that captures the output of a Wayland compositor. If you want to select an area of the display, you can use another application call ed `slurp`, but this will need to be installed separately.

```
sudo apt install slurp
```

From the command line you can now use the following to select an area of the screen and capture it:

```
slurp | grim -g -
```

You can assign a key binding to this command. For example, the following addition to `wayfire.ini` will assign it to SHIFT PrtScn.

```
binding_screenshot_interactive = <shift> KEY_SYSRQ
command_screenshot_interactive = slurp | grim -g -
```

Use `man slurp` and `man grim` to get documentation on these commands.

HDMI hotplug detection

Before KMS, the command `force_hdmi_hotplug=1` forced the HDMI hotplug detect to be ON, even if not detected.

In KMS-based systems like Bookworm, this command no longer works, and you should use the Linux kernel equivalent. Use the kernel command line to set the kernel driver to the HDMI settings you want.

For example, the following information, added to the end of the kernel command line (`\boot\cmdline.txt`), will force the display on, and set it to 1080p60.

```
video=HDMI-A-1:1920x1080@60D
```

Raspberry Pi