meta-raspberrypi Documentation

Release master

meta-raspberrypi contributors

Feb 27, 2021

Contents

Contents:

CHAPTER 1

meta-raspberrypi

Yocto BSP layer for the Raspberry Pi boards - http://www.raspberrypi.org/.

Build Status Build Status Build Status Build Status Documentation Status Matrix

| | | |:-: | :-: | | Build server sponsored by | 

balena.io. |

1.1 Quick links

- Git repository web frontend: https://github.com/agherzan/meta-raspberrypi
- Mailing list (yocto mailing list): yocto@yoctoproject.org
- Issues management (Github Issues): https://github.com/agherzan/meta-raspberrypi/issues
- Documentation: http://meta-raspberrypi.readthedocs.io/en/latest/

1.2 Description

This is the general hardware specific BSP overlay for the RaspberryPi device.

More information can be found at: http://www.raspberrypi.org/ (Official Site)

The core BSP part of meta-raspberrypi should work with different OpenEmbedded/Yocto distributions and layer stacks, such as:

- Distro-less (only with OE-Core).
- Yoe Disto (Video and Camera Products).

- Yocto/Poky (main focus of testing).

1.3 Dependencies

This layer depends on:

- URI: git://git.yoctoproject.org/poky
  - branch: master
  - revision: HEAD
- URI: git://git.openembedded.org/meta-openembedded
  - layers: meta-oe, meta-multimedia, meta-networking, meta-python
  - branch: master
  - revision: HEAD

1.4 Quick Start

1. source poky/oe-init-build-env rpi-build
2. Add this layer to bblayers.conf and the dependencies above
3. Set MACHINE in local.conf to one of the supported boards
4. bitbake core-image-base
5. Use bmaptool to copy the generated .wic.bz2 file to the SD card
6. Boot your RPI

1.5 Quick Start with kas

1. Install kas build tool from PyPi (sudo pip3 install kas)
2. kas build meta-raspberrypi/kas-poky-rpi.yml
3. Use bmaptool to copy the generated .wic.bz2 file to the SD card
4. Boot your RPI

To adjust the build configuration with specific options (I2C, SPI, . . .), simply add a section as follows:

```
local_conf_header:
  rpi-specific: |
    ENABLE_I2C = "1"
    RPI_EXTRA_CONFIG = "dtoverlay=pi3-disable-bt"
```

To configure the machine, you have to update the `machine` variable. And the same for the `distro`.

For further information, you can read more at https://kas.readthedocs.io/en/1.0/index.html

1.6 Maintainers

- Andrei Gherzan <andrei at gherzan.ro>

CHAPTER 2

Layer Contents

2.1 Supported Machines

- raspberrypi
- raspberrypi0
- raspberrypi0-wifi
- raspberrypi2
- raspberrypi3
- raspberrypi3-64 (64 bit kernel & userspace)
- raspberrypi4
- raspberrypi4-64 (64 bit kernel & userspace)
- raspberrypi-cm (dummy alias for raspberrypi)
- raspberrypi-cm3

Note: The raspberrypi3 machines include support for Raspberry Pi 3B+.

2.2 Images

- rpi-test-image
  - Image based on core-image-base which includes most of the packages in this layer and some media samples.

For other uses it's recommended to base images on `core-image-minimal` or `core-image-base` as appropriate. The old image names (`rpi-hwup-image` and `rpi-basic-image`) are deprecated.

2.3 WiFi and Bluetooth Firmware

Be aware that the WiFi and Bluetooth firmware for the supported boards is not available in the base version of `linux-firmware` from OE-Core (poky). The files are added from Raspbian repositories in this layer's bbappends to that recipe. All machines define `MACHINE_EXTRA_RRECOMMENDS` to include the required wireless firmware; raspberrypi3 supports 3, 3B, and 3B+ and so include multiple firmware packages.

CHAPTER 3

Optional build configuration

There are a set of ways in which a user can influence different parameters of the build. We list here the ones that are closely related to this BSP or specific to it. For the rest please check: http://www.yoctoproject.org/docs/latest/ref-manual/ref-manual.html

3.1 Compressed deployed files

1. Overwrite IMAGE_FSTYPES in local.conf
   - `IMAGE_FSTYPES = "tar.bz2 ext3.xz"`
2. Overwrite SDIMG_ROOTFS_TYPE in local.conf
   - `SDIMG_ROOTFS_TYPE = "ext3.xz"`

Accommodate the values above to your own needs (ex: ext3 / ext4).

3.2 GPU memory

- `GPU_MEM`: GPU memory in megabyte. Sets the memory split between the ARM and GPU. ARM gets the remaining memory. Min 16. Default 64.
- `GPU_MEM_256`: GPU memory in megabyte for the 256MB Raspberry Pi. Ignored by the 512MB RP. Overrides gpu_mem. Max 192. Default not set.
- `GPU_MEM_512`: GPU memory in megabyte for the 512MB Raspberry Pi. Ignored by the 256MB RP. Overrides gpu_mem. Max 448. Default not set.
- `GPU_MEM_1024`: GPU memory in megabyte for the 1024MB Raspberry Pi. Ignored by the 256MB/512MB RP. Overrides gpu_mem. Max 944. Default not set.

See: https://www.raspberrypi.org/documentation/configuration/config-txt/memory.md

3.3 VC4

By default, each machine uses `vc4` for graphics. This will in turn sets mesa as provider for `gl` libraries. `DISABLE_VC4GRAPHICS` can be set to `1` to disable this behaviour falling back to using `userland`. Be aware that `userland` has not support for 64-bit arch. If you disable `vc4` on a 64-bit Raspberry Pi machine, expect build breakage.

3.4 Add purchased license codecs

To add your own licenses use variables `KEY_DECODE_MPG2` and `KEY_DECODE_WVC1` in local.conf. Example:

```
KEY_DECODE_MPG2 = "12345678"
KEY_DECODE_WVC1 = "12345678"
```

You can supply more licenses separated by comma. Example:

```
KEY_DECODE_WVC1 = "0x12345678,0xabcdabcd,0x87654321"
```

See: https://www.raspberrypi.org/documentation/configuration/config-txt/codeclicence.md

3.5 Disable overscan

By default the GPU adds a black border around the video output to compensate for TVs which cut off part of the image. To disable this set this variable in local.conf:

```
DISABLE_OVERSCAN = "1"
```

3.6 Disable splash screen

By default a rainbow splash screen is shown after the GPU firmware is loaded. To disable this set this variable in local.conf:

```
DISABLE_SPLASH = "1"
```

3.7 Boot delay

The Raspberry Pi waits a number of seconds after loading the GPU firmware and before loading the kernel. By default it is one second. This is useful if your SD card needs a while to get ready before Linux is able to boot from it. To remove (or adjust) this delay set these variables in local.conf:

```
BOOT_DELAY = "0"
BOOT_DELAY_MS = "0"
```

3.8 Set overclocking options

The Raspberry Pi can be overclocked. As of now overclocking up to the "Turbo Mode" is officially supported by the Raspberry Pi and does not void warranty. Check the config.txt for a detailed description of options and modes. The following variables are supported in local.conf: `ARM_FREQ`, `GPU_FREQ`, `CORE_FREQ`, `SDRAM_FREQ` and `OVER_VOLTAGE`.

Example official settings for Turbo Mode in Raspberry Pi 2:

```
ARM_FREQ = "1000"
CORE_FREQ = "500"
SDRAM_FREQ = "500"
OVER_VOLTAGE = "6"
```

See: https://www.raspberrypi.org/documentation/configuration/config-txt/overclocking.md

3.9 HDMI and composite video options

The Raspberry Pi can output video over HDMI or SDTV composite (the RCA connector). By default the video mode for these is autodetected on boot: the HDMI mode is selected according to the connected monitor's EDID information and the composite mode is defaulted to NTSC using a 4:3 aspect ratio. Check the config.txt for a detailed description of options and modes. The following variables are supported in local.conf: `HDMI_FORCE_HOTPLUG`, `HDMI_DRIVE`, `HDMI_GROUP`, `HDMI_MODE`, `CONFIG_HDMI_BOOST`, `SDTV_MODE`, `SDTV_ASPECT` and `DISPLAY_ROTATE`.

Example to force HDMI output to 720p in CEA mode:

```
HDMI_GROUP = "1"
HDMI_MODE = "4"
```

See: https://www.raspberrypi.org/documentation/configuration/config-txt/video.md

3.10 Video camera support with V4L2 drivers

Set this variable to enable support for the video camera (Linux 3.12.4+ required):

```
VIDEO_CAMERA = "1"
```

3.11 Enable offline compositing support

Set this variable to enable support for dispmanx offline compositing:

```
DISPMANX_OFFLINE = "1"
```

This will enable the firmware to fall back to off-line compositing of Dispmanx elements. Normally the compositing is done on-line, during scanout, but cannot handle too many elements. With off-line enabled, an off-screen buffer is allocated for compositing. When scene complexity (number and sizes of elements) is high, compositing will happen off-line into the buffer.

Heavily recommended for Wayland/Weston.

See: http://wayland.freedesktop.org/raspberrypi.html

3.12 Enable kgdb over console support

To add the kdbg over console (kgdboc) parameter to the kernel command line, set this variable in local.conf:

```
ENABLE_KGDB = "1"
```

3.13 Disable rpi boot logo

To disable rpi boot logo, set this variable in local.conf:

```
DISABLE_RPI_BOOT_LOGO = "1"
```

3.14 Boot to U-Boot

To have u-boot load kernel image, set in your local.conf:

```
RPI_USE_U_BOOT = "1"
```

This will select the appropriate image format for use with u-boot automatically. For further customisation the KERNEL_IMAGETYPE and KERNEL_BOOTCMD variables can be overridden to select the exact kernel image type (eg. zImage) and u-boot command (eg. bootz) to be used.

3.15 Image with Initramfs

To build an initramfs image:

- Set this 3 kernel variables (in kernel's do_configure_prepend in linux-raspberrypi.inc after the line kernel_configure_variable LOCALVERSION “””””)
  - kernel_configure_variable BLK_DEV_INITRD y
  - kernel_configure_variable INITRAMFS_SOURCE “”
  - kernel_configure_variable RD_GZIP y
- Set the yocto variables (e.g. in local.conf)
  - `INITRAMFS_IMAGE = "<name for your initramfs image>"`
  - `INITRAMFS_IMAGE_BUNDLE = "1"`
  - `BOOT_SPACE = "1073741"`
  - `INITRAMFS_MAXSIZE = "315400"`
  - `IMAGE_FSTYPES_pn-${INITRAMFS_IMAGE} = "${INITRAMFS_FSTYPES}"`

3.16 Including additional files in the SD card image boot partition

The SD card image class supports adding extra files into the boot partition, where the files are copied from either the image root partition or from the build image deploy directory.

To copy files that are present in the root partition into boot, FATPAYLOAD is a simple space-separated list of files to be copied:

```
FATPAYLOAD = "/boot/example1 /boot/example2"
```

To copy files from the image deploy directory, the files should be listed in the DEPLOYPAYLOAD as a space-separated list of entries. Each entry lists a file to be copied, and an optional destination filename can be specified by supplying it after a colon separator.

```
DEPLOYPAYLOAD = "example1-${MACHINE}:example1 example2"
```

Files that are to be included from the deploy directory will be produced by tasks that image building task must depend upon, to ensure that the files are available when they are needed, so these component deploy tasks must be added to: RPI_SDIMG_EXTRA_DEPENDS.

```
RPI_SDIMG_EXTRA_DEPENDS_append = " example:do_deploy"
```

3.17 Enable SPI bus

When using device tree kernels, set this variable to enable the SPI bus:

```
ENABLE_SPI_BUS = "1"
```

3.18 Enable I2C

When using device tree kernels, set this variable to enable I2C:

```
ENABLE_I2C = "1"
```

Furthermore, to auto-load I2C kernel modules set:

```
KERNEL_MODULE_AUTOLOAD_rpi += "i2c-dev i2c-bcm2708"
```

3.19 Enable PiTFT support

Basic support for using PiTFT screens can be enabled by adding below in local.conf:

- `MACHINE_FEATURES += "pitft"`
  - This will enable SPI bus and i2c device-trees, it will also setup framebuffer for console and x server on PiTFT.

NOTE: To get this working the overlay for the PiTFT model must be build, added and specified as well (dtoverlay= in config.txt).

Below is a list of currently supported PiTFT models in meta-raspberrypi, the modelname should be added as a MACHINE_FEATURES in local.conf like below:

```
MACHINE_FEATURES += "pitft <modelname>"
```

List of currently supported models:

- pitft22
- pitft28r
- pitft28c
- pitft35r

3.20 Misc. display

If you would like to use the Waveshare "C" 1024×600, 7 inch Capacitive Touch Screen LCD, HDMI interface (http://www.waveshare.com/7inch-HDMI-LCD-C.htm) Rev 2.1, please set the following in your local.conf:

```
WAVESHARE_1024X600_C_2_1 = "1"
```

3.21 Enable UART

RaspberryPi 0, 1, 2 and CM will have UART console enabled by default.

RaspberryPi 0 WiFi and 3 does not have the UART enabled by default because this needs a fixed core frequency and enable_uart will set it to the minimum. Certain operations - 60fps h264 decode, high quality deinterlace - which aren't performed on the ARM may be affected, and we wouldn't want to do that to users who don't want to use the serial port. Users who want serial console support on RaspberryPi 0 Wifi or 3 will have to explicitly set in local.conf:

```
ENABLE_UART = "1"
```

Ref.:

- https://github.com/raspberrypi/firmware/issues/553
- https://github.com/RPi-Distro/repo/issues/22

3.22 Enable USB Peripheral (Gadget) support

The standard USB driver only supports host mode operations. Users who want to use gadget modules like g_ether should set the following in local.conf:

```
ENABLE_DWC2_PERIPHERAL = "1"
```

3.23 Enable USB host support

By default in case of the Compute Module 4 IO Board the standard USB driver that usually supports host mode operations is disabled for power saving reasons. Users who want to use the 2 USB built-in ports or the other ports provided via the header extension should set the following in local.conf:

```
ENABLE_DWC2_HOST = "1"
```

3.24 Enable Openlabs 802.15.4 radio module

When using device tree kernels, set this variable to enable the 802.15.4 hat:

```
ENABLE_AT86RF = "1"
```

See: https://openlabs.co/OSHW/Raspberry-Pi-802.15.4-radio

3.25 Enable CAN

In order to use CAN with an MCP2515-based module, set the following variables:

```
ENABLE_SPI_BUS = "1"
ENABLE_CAN = "1"
```

In case of dual CAN module (e.g. PiCAN2 Duo), set following variables instead:

```
ENABLE_SPI_BUS = "1"
ENABLE_DUAL_CAN = "1"
```

Some modules may require setting the frequency of the crystal oscillator used on the particular board. The frequency is usually marked on the package of the crystal. By default, it is set to 16 MHz. To change that to 8 MHz, the following variable also has to be set:

```
CAN_OSCILLATOR="8000000"
```

Tested modules:

- PiCAN2 (16 MHz crystal): http://skpang.co.uk/catalog/pican2-canbus-board-for-raspberry-pi-23-p-1475.html
- WaveShare RS485 CAN HAT (8 MHz or 12 MHz crystal): https://www.waveshare.com/rs485-can-hat.htm
- PiCAN2 Duo (16 MHz crystal): http://skpang.co.uk/catalog/pican2-duo-canbus-board-for-raspberry-pi-23-p-1480.html

3.26 Enable infrared

Users who want to enable infrared support, for example for using LIRC (Linux Infrared Remote Control), have to explicitly set in local.conf:

```
ENABLE_IR = "1"
```

This will add device tree overlays gpio-ir and gpio-ir-tx to config.txt. Appropriate kernel modules will be also included in the image. By default the GPIO pin for gpio-ir is set to 18 and the pin for gpio-ir-tx is 17. Both pins can be easily changed by modifying variables `GPIO_IR` and `GPIO_IR_TX`.

3.27 Manual additions to config.txt

The `RPI_EXTRA_CONFIG` variable can be used to manually add additional lines to the `config.txt` file if there is not a specific option above for the configuration you need. To add multiple lines you must include `\n` separators. If double-quotes are needed in the lines you are adding you can use single quotes around the whole string.

For example, to add a comment containing a double-quote and a configuration option:

```
RPI_EXTRA_CONFIG = ' \n \
    # Raspberry Pi 7\" display/touch screen \n \
    lcd_rotate=2 \n \
    '
```

3.28 Enable Raspberrypi Camera V2

RaspberryPi does not have the unicam device (RaspberryPi Camera) enabled by default. Because this unicam device (bcm2835-unicam) as of now is used by libcamera opensource. So we have to explicitly set in local.conf.

```
RASPBERRYPI_CAMERA_V2 = "1"
```

This will add the device tree overlays imx219 (RaspberryPi Camera sensor V2 driver) to config.txt. Also, this will enable adding Contiguous Memory Allocation value in the cmdline.txt.

Ref.:

- https://github.com/raspberrypi/documentation/blob/master/linux/software/libcamera/README.md
- https://www.raspberrypi.org/blog/an-open-source-camera-stack-for-raspberry-pi-using-libcamera/

CHAPTER 4

Extra apps

4.1 omxplayer

omxplayer depends on libav which has a commercial license. So in order to be able to compile omxplayer you will need to whiteflag the commercial license in your local.conf:

```
LICENSE_FLAGS_WHITELIST = "commercial"
```

CHAPTER 5

Contributing

5.1 Mailing list

The main communication tool in use is the Yocto Project mailing list:

- yocto@yoctoproject.org
- https://lists.yoctoproject.org/listinfo/yocto

Feel free to ask any kind of questions but please always prepend your email subject with `[meta-raspberrypi]` as this is the global *Yocto* mailing list and not a dedicated *meta-raspberrypi* mailing list.

5.2 Formatting patches

First and foremost, all of the contributions to the layer must be compliant with the standard openembedded patch guidelines:

- http://www.openembedded.org/wiki/Commit_Patch_Message_Guidelines

In summary, your commit log messages should be formatted as follows:

```
<layer-component>: <short log/statement of what needed to be changed>

(Optional pointers to external resources, such as defect tracking)

The intent of your change.

(Optional: if it's not clear from above, how your change resolves
the issues in the first part)

Signed-off-by: Your Name <yourname@youremail.com>
```

The `<layer-component>` is the layer component name that your changes affect. It is important that you choose it correctly. A simple guide for selecting a a good component name is the following:

- For changes that affect *layer recipes*, please just use the **base names** of the affected recipes, separated by commas (`,`), as the component name. For example: use `omxplayer` instead of `omxplayer_git.bb`. If you are adding new recipe(s), just use the new recipe(s) base name(s). An example for changes to multiple recipes would be `userland,vc-graphics,wayland`.
- For changes that affect the *layer documentation*, please just use `docs` as the component name.
- For changes that affect *other files*, i.e. under the `conf` directory, please use the full path as the component name, e.g. `conf/layer.conf`.
- For changes that affect the *layer itself* and do not fall into any of the above cases, please use `meta-raspberrypi` as the component name.

A full example of a suitable commit log message is below:

```
foobar: Adjusted the foo setting in bar

When using foobar on systems with less than a gigabyte of RAM common
usage patterns often result in an Out-of-memory condition causing
slowdowns and unexpected application termination.

Low-memory systems should continue to function without running into
memory-starvation conditions with minimal cost to systems with more
available memory.  High-memory systems will be less able to use the
full extent of the system, a dynamically tunable option may be best,
long-term.

The foo setting in bar was decreased from X to X-50% in order to
ensure we don't exhaust all system memory with foobar threads.

Signed-off-by: Joe Developer <joe.developer@example.com>
```

A common issue during patch reviewing is commit log formatting, please review the above formatting guidelines carefully before sending your patches.

5.3 Sending patches

The preferred method to contribute to this project is to send pull requests to the GitHub mirror of the layer:

- https://github.com/agherzan/meta-raspberrypi

In addition, you may send patches for review to the above specified mailing list. In this case, when creating patches using `git` please make sure to use the following formatting for the message subject:

```
git format-patch -s --subject-prefix='meta-raspberrypi][PATCH' origin
```

Then, for sending patches to the mailing list, you may use this command:

```
git send-email --to yocto@yoctoproject.org <generated patch>
```

5.4 GitHub issues

In order to manage and track the layer issues more efficiently, the GitHub issues facility is used by this project:

- https://github.com/agherzan/meta-raspberrypi/issues

If you submit patches that have a GitHub issue associated, please make sure to use standard GitHub keywords, e.g. `closes`, `resolves` or `fixes`, before the "Signed-off-by" tag to close the relevant issues automatically:

```
foobar: Adjusted the foo setting in bar

Fixes: #324

Signed-off-by: Joe Developer <joe.developer@example.com>
```

More information on the available GitHub close keywords can be found here:

- https://help.github.com/articles/closing-issues-using-keywords

CHAPTER 6

Indices and tables

- genindex
- modindex
- search