Using linux-yocto as a Yocto BSP kernel

Managing your BSP kernel in a different way

Dmitry Baryshkov
mailto:dmitry.baryshkov@linaro.org
About me

- OpenEmbedded contributor since 2007
  - ... and even remembers OpenZaurus
- Linux kernel contributor since 2007
  - Around 2200 commits
- worked with Linaro in 2017-2019, joined back in 2020
  - A part of Qualcomm Ecosystem Team
- meta-qcom leading developer since 2020
  - Maintainer since 2023
Typical OE BSP
From the Linux Kernel point of view
linux-awesome-bsp

- Custom bb recipe in vendor’s BSP layer
linux-awesome-bsp

- Custom bb recipe in vendor’s BSP layer
- SRC_URI pointing to vendor’s Git tree
  - Which might track development history
  - ... or it might not
  - “Revert fix for the fix for the commit”
  - Was this patch ever shown to upstream developers?
  - LTS version if you are lucky
  - Security updates if you are extremely lucky
Custom bb recipe in vendor’s BSP layer

SRC_URI pointing to vendor’s Git tree
  - Which might track development history
  - ... or it might not
  - “Revert fix for the fix for the commit”
  - Was this patch ever shown to upstream developers?
  - LTS version if you are lucky
  - Security updates if you are extremely lucky

kernel config
  - usually a file under the same git tree
  - ... or a config file in the layer
  - Anyway, good luck modifying it in a simple but future-proof way
Custom bb recipe in vendor’s BSP layer
- SRC_URI pointing to vendor’s Git tree
  - Which might track development history
  - … or it might not
  - “Revert fix for the fix for the commit”
  - Was this patch ever shown to upstream developers?
  - LTS version if you are lucky
  - Security updates if you are extremely lucky
- kernel config
  - usually a file under the same git tree
  - … or a config file in the layer
  - Anyway, good luck modifying it in a simple but future-proof way

But everybody does it this way?!
linux-yocto kernel
 linux-yocto kernel recipe

- The kernel recipe used by OE-Core for the QEMU targets
- Also BSP for several standard platforms
- Follows linux-stable releases
- Tracks the latest released kernel and LTS kernels
- Has very powerful kernel configuration framework (scc)
- Endorsed by YoctoProject Compatible Layer
linux-yocto kernel recipe

- The kernel recipe used by OE-Core for the QEMU targets
- Also BSP for several standard platforms
- Follows linux-stable releases
- Tracks the latest released kernel and LTS kernels
- Has very powerful kernel configuration framework (scc)
- Endorsed by YoctoProject Compatible Layer

Sounds perfect, doesn’t it?
But nobody uses it. Almost.
**linux-yocto kernel recipe**

- The kernel recipe used by OE-Core for the QEMU targets
- Also BSP for several standard platforms
- Follows linux-stable releases
- Tracks the latest released kernel and LTS kernels
- Has very powerful kernel configuration framework (scc)
- Endorsed by YoctoProject Compatible Layer

Sounds perfect, doesn’t it?
But nobody uses it. Almost.

We do. Now!
Entry points

- `linux-yocto_%bbappend`

```
# do not override KBRANCH and SRCREV_machine, use defaults
COMPATIBLE_MACHINE:qcom = "qcom-armv8a"
FILESEXTRAPATHS:prepend:qcom := "${THISDIR}/${PN}:

# include all Qualcomm-specific files
SRC_URI:append:qcom = " \n  file://qcom.scc \n  "
```

Configuration gets assembled from `scc` and `cfg` files
And you add more in your layers using this BSP!

Defer -stable tracking to linux-yocto maintainers (Thank you, Bruce!)
Entry points

- `linux-yocto_6.6.bbappend`

```bash
SRC_URI:append:qcom = " \
    file://0001-arm64-dts-qcom-disable.patch \ 
    file://qca6390-driver/0001-dt-bindings-mfd-qcom.patch \ 
    file://qca6390-driver/0002-mfd-qca639x-add-support.patch \ 
    file://qca6390-driver/0003-mfd-qcom-qca639x-switch.patch \ 
...
"
```

Now each patch MUST have ‘Upstream-Status’ trailer!
History is no longer lost when somebody switches Git branches
Config fragments

- recipes-kernel/linux/linux-yocto/qcom.scc
  - empty file, triggers inclusion of other files
- recipes-kernel/linux/linux-yocto/bsp/qcom-armv8a/qcom-armv8a.scc

```
kconf hardware qcom.cfg

include qcom-sdm845.scc
include qcom-sm8250.scc

# include standard features and config fragments
include features/i2c/i2c.scc
include features/power/arm.scc

include cfg/timer/rtc.scc
include cfg/dmaengine.scc
```
Config fragments

- recipes-kernel/linux/linux-yocto/bsp/qcom-armv8a/qcom.cfg

```bash
CONFIG_ARCH_QCOM=y
CONFIG_ARM_PSCI_CPUIDLE=y

# CONFIG_MOUSE_PS2 is not set
# CONFIG_KEYBOARD_ATKBD is not set
CONFIG_KEYBOARD_GPIO=y
```

- recipes-kernel/linux/linux-yocto/bsp/qcom-armv8a/qcom-sdm845.scc

```bash
kconf hardware qcom-rpmh.cfg
kconf hardware qcom-sdm845.cfg
```

- etc.
Downsides

● No control over the exact kernel version
● Sometimes linux-yocto kernels get delayed a bit
● Additional patches on top of the Linux release tag

● Development becomes more complicated
● Responsibilities shift onto OE layer maintainers

● What if we have several hundred of BSP patches?
Links

- [https://git.yoctoproject.org/linux-yocto/](https://git.yoctoproject.org/linux-yocto/)
- [https://git.yoctoproject.org/yocto-kernel-cache/](https://git.yoctoproject.org/yocto-kernel-cache/)
- [https://github.com/Linaro/meta-qcom/](https://github.com/Linaro/meta-qcom/)
- [https://www.linaro.org/services/](https://www.linaro.org/services/)
- [https://mastodon.social/@LinaroLtd](https://mastodon.social/@LinaroLtd)
  - We are hiring!
Thank you