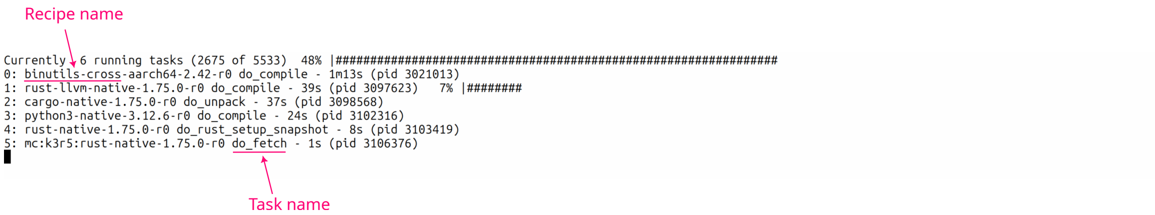
Yocto: learning from BitBake output

Lab observations: recipes and tasks



Recipe name

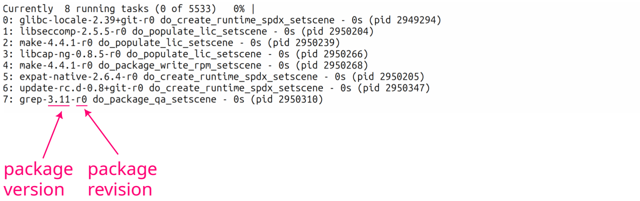
```
Currently  6 running tasks (2675 of 5533)  48% |########################################################################
0: binutils-cross-aarch64-2.42-r0 do_compile - 1m13s (pid 3021013)
1: rust-llvm-native-1.75.0-r0 do_compile - 39s (pid 3097623)   7% |########
2: cargo-native-1.75.0-r0 do_unpack - 37s (pid 3098568)
3: python3-native-3.12.6-r0 do_compile - 24s (pid 3102316)
4: rust-native-1.75.0-r0 do_rust_setup_snapshot - 8s (pid 3103419)
5: mc:k3r5:rust-native-1.75.0-r0 do_fetch - 1s (pid 3106376)
```

Task name

What to understand:

- Each component to build (program, library, kernel, image) is represented by a **recipe**.
- Each recipe describes a set of **tasks**: fetch, unpack, compile, install...
- BitBake **manages build dependencies at the task level**, not at the recipe level.

Lab observations: versions and revisions



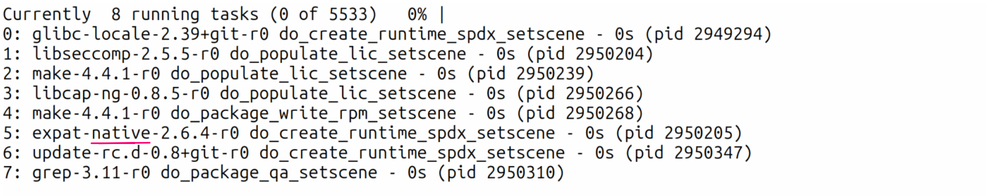
```
Currently  8 running tasks (0 of 5533)   0% |
0: glibc-locale-2.39+git-r0 do_create_runtime_spdx_setscene - 0s (pid 2949294)
1: libseccomp-2.5.5-r0 do_populate_lic_setscene - 0s (pid 2950204)
2: make-4.4.1-r0 do_populate_lic_setscene - 0s (pid 2950239)
3: libcap-ng-0.8.5-r0 do_populate_lic_setscene - 0s (pid 2950266)
4: make-4.4.1-r0 do_package_write_rpm_setscene - 0s (pid 2950268)
5: expat-native-2.6.4-r0 do_create_runtime_spdx_setscene - 0s (pid 2950205)
6: update-rc.d-0.8+git-r0 do_create_runtime_spdx_setscene - 0s (pid 2950347)
7: grep-3.11-r0 do_package_qa_setscene - 0s (pid 2950310)
```

Recipes are deployed through **packages** (see `tmp/deploy/[rpm|ipk|deb]`)

- Packages allow to split components into several parts: binaries, binaries with debug info, headers, data, documentation... We can just install what we need.
- Packages can be removed too, useful to have the ability to remove no-longer needed stuff without regenerating everything (hello Buildroot 😜)
- Each package has a **version**, which coincides with the recipe version.
- Each package has a **revision**, typically useful to generate different binaries from the same sources (different configuration, patches applied)

Lab observations: native recipes

"native" recipe

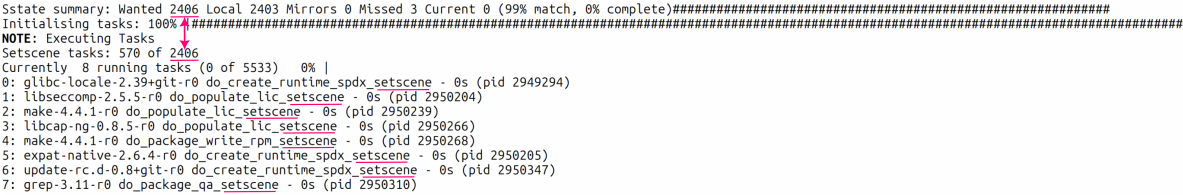


```
Currently  8 running tasks (0 of 5533)   0% |
0: glibc-locale-2.39+git-r0 do_create_runtime_spdx_setscene - 0s (pid 2949294)
1: libseccomp-2.5.5-r0 do_populate_lic_setscene - 0s (pid 2950204)
2: make-4.4.1-r0 do_populate_lic_setscene - 0s (pid 2950239)
3: libcap-ng-0.8.5-r0 do_populate_lic_setscene - 0s (pid 2950266)
4: make-4.4.1-r0 do_package_write_rpm_setscene - 0s (pid 2950268)
5: expat-native-2.6.4-r0 do_create_runtime_spdx_setscene - 0s (pid 2950205)
6: update-rc.d-0.8+git-r0 do_create_runtime_spdx_setscene - 0s (pid 2950347)
7: grep-3.11-r0 do_package_qa_setscene - 0s (pid 2950310)
```

Some recipes have the `-native` suffix:

- **native** recipes generate code for the **host** machine (your PC running Yocto)
- Yocto prefers to build the tools it needs by itself rather than rely on distribution provided ones. This brings reproducibility!
- Examples: compilers, CMake, QEMU...
- Exceptions: Python (needed to run BitBake), git, gcc (needed to build the native gcc)...

Lab observations: setscene tasks

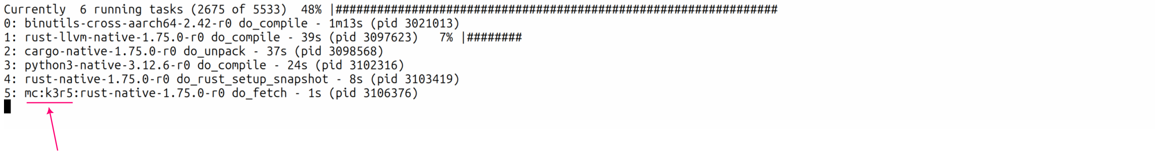


```
Sstate summary: Wanted 2406 Local 2403 Mirrors 0 Missed 3 Current 0 (99% match, 0% complete)############################################################
Initialising tasks: 100% |########################################################################################################################################
NOTE: Executing Tasks
Setscene tasks: 570 of 2406
Currently  8 running tasks (0 of 5533)   0% |
0: glibc-locale-2.39+git-r0 do_create_runtime_spdx_setscene - 0s (pid 2949294)
1: libseccomp-2.5.5-r0 do_populate_lic_setscene - 0s (pid 2950204)
2: make-4.4.1-r0 do_populate_lic_setscene - 0s (pid 2950239)
3: libcap-ng-0.8.5-r0 do_populate_lic_setscene - 0s (pid 2950266)
4: make-4.4.1-r0 do_package_write_rpm_setscene - 0s (pid 2950268)
5: expat-native-2.6.4-r0 do_create_runtime_spdx_setscene - 0s (pid 2950205)
6: update-rc.d-0.8+git-r0 do_create_runtime_spdx_setscene - 0s (pid 2950347)
7: grep-3.11-r0 do_package_qa_setscene - 0s (pid 2950310)
```

Some tasks have the `_setscene` suffix:

- They correspond to tasks which were already built before in the same conditions (same input).
- To save time, their output is retrieved from the **Shared State Cache** (”Sstate”).

Lab observations: did you notice?



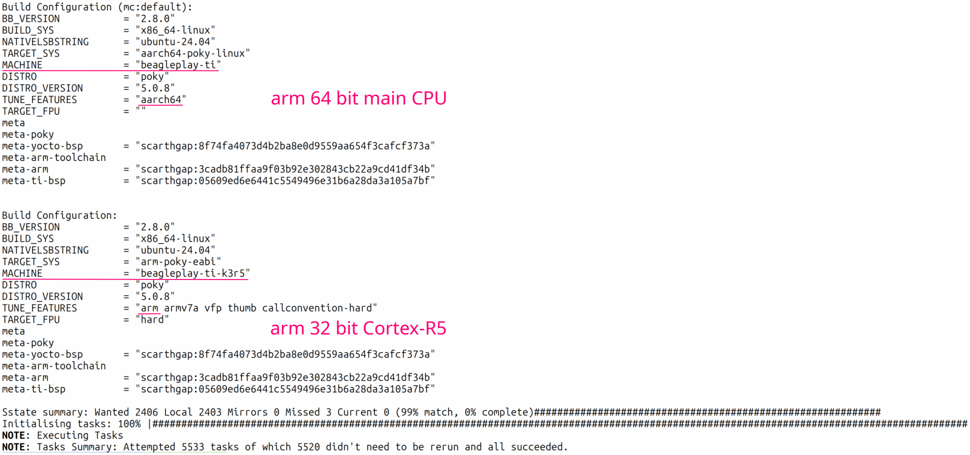
```
Currently  6 running tasks (2675 of 5533)  48% |################################################################
0: binutils-cross-aarch64-2.42-r0 do_compile - 1m13s (pid 3021013)
1: rust-llvm-native-1.75.0-r0 do_compile - 39s (pid 3097623)   7% |########
2: cargo-native-1.75.0-r0 do_unpack - 37s (pid 3098568)
3: python3-native-3.12.6-r0 do_compile - 24s (pid 3102316)
4: rust-native-1.75.0-r0 do_rust_setup_snapshot - 8s (pid 3103419)
5: mc:k3r5:rust-native-1.75.0-r0 do_fetch - 1s (pid 3106376)
```

Additional configuration

Here, we have a special task prefixed by `mc:k3r5`

- `mc` means **multiconfig**
- In the case of the BeaglePlay, multiconfig means generating two images for the board
  - The default configuration to generate the image for the Cortex-A53 cores in the AM625 SoC.
  - The `k3r5` configuration to generate another image for the Cortex-R5 processor in AM625.

Lab observations: build output



```
Build Configuration (mc:default):
BB_VERSION           = "2.8.0"
BUILD_SYS            = "x86_64-linux"
NATIVELSBSTRING      = "ubuntu-24.04"
TARGET_SYS           = "aarch64-poky-linux"
MACHINE              = "beagleplay-ti"
DISTRO               = "poky"
DISTRO_VERSION       = "5.0.8"
TUNE_FEATURES        = "aarch64"
TARGET_FPU           = ""
meta
meta-poky
meta-yocto-bsp       = "scarthgap:8f74fa4073d4b2ba8e0d9559aa654f3cafcf373a"
meta-arm-toolchain
meta-arm             = "scarthgap:3cadb81ffaa9f03b92e302843cb22a9cd41df34b"
meta-ti-bsp          = "scarthgap:05609ed6e6441c5549496e31b6a28da3a105a7bf"


Build Configuration:
BB_VERSION           = "2.8.0"
BUILD_SYS            = "x86_64-linux"
NATIVELSBSTRING      = "ubuntu-24.04"
TARGET_SYS           = "arm-poky-eabi"
MACHINE              = "beagleplay-ti-k3r5"
DISTRO               = "poky"
DISTRO_VERSION       = "5.0.8"
TUNE_FEATURES        = "arm armv7a vfp thumb callconvention-hard"
TARGET_FPU           = "hard"
meta
meta-poky
meta-yocto-bsp       = "scarthgap:8f74fa4073d4b2ba8e0d9559aa654f3cafcf373a"
meta-arm-toolchain
meta-arm             = "scarthgap:3cadb81ffaa9f03b92e302843cb22a9cd41df34b"
meta-ti-bsp          = "scarthgap:05609ed6e6441c5549496e31b6a28da3a105a7bf"

Sstate summary: Wanted 2406 Local 2403 Mirrors 0 Missed 3 Current 0 (99% match, 0% complete)############################################################
Initialising tasks: 100% |#############################################################################################################################################
NOTE: Executing Tasks
NOTE: Tasks Summary: Attempted 5533 tasks of which 5520 didn't need to be rerun and all succeeded.
```

arm 64 bit main CPU

arm 32 bit Cortex-R5

Configurations are also shown in the build output.

Lab observations: key takeaways 💡

- BitBake processes **recipes** to build components
- Recipes describe multiple **tasks**: fetching sources, configuring, compiling, installing
- BitBake schedules tasks, managing dependencies at task level.
- BitBake also builds **native recipes**: tools for your build machine.
- **Setscene** tasks: tasks which output can be retrieved from the **Shared State Cache**.
- Components are deployed through **packages**.
- With BitBake, you can generate multiple images at once (**multiconfig** capability).

More Yocto tricks and tips

Check out our training course:
`https://rootcommit.com/training/yocto/`