Building RT image with Yocto

Pierre Ficheux (pierre.ficheux@smile.fr)

02/2018
• Poor English speaker !
• But “good” French speaker and writer :-)

“Loin du français je meurs”
(Louis-Ferdinand Céline)
$ whoami

- Embedded Linux developer, writer and teacher
- CTO @ Smile ECS (Embedded & Connected Systems)
- Last book about embedded Linux (in french !)
Building RT image with Yocto

Yocto quick history

- Open Embedded started in 2003 (OpenZaurus)
- Buildroot was not smart enough (static, no packages)
- Recipes = “how to cross-compile X for target Y”
- BitBake program to run OE recipes
  - Inspired by “Portage” (Gentoo)
  - written in Python
- Yocto started in 2010 (by Intel & friends) including
  - OE (core)
  - BitBake
  - Poky (reference distro)
  - E-Glibc (Glibc since 2014)
  - Smart documentation !!
Building RT image with Yocto

Yocto layers

- OpenEmbedded core metadata (oe-core)
  - Yocto specific layer (meta-yocto)
    - Hardware specific BSP layer (meta-fsl-arm, meta-digi)
      - UI specific layer (meta-efl, meta-gnome, meta-gpe, meta-xfce)
        - commercial layer (open source vendors)
          - developer-specific layer (user software)

My layer
Building RT image with Yocto

Yocto / OE principles

- Recipes (.bb)
- Extended recipes (.bbappend)
- Based on classes (.bbclass) → inherit
- Include files (.inc) → include / require
- Configuration files (.conf)
- Everything external should be in meta-<whatevever>
- Very few number of mainline targets (13)
  - QEMU-* (x86, ARM, MIPS, PPC)
  - BeagleBone Black
  - PPC
  - Generic x86 + x86_64
  - EdgeRouter
- Most real platforms need a meta-<hw-type> layer
Building RT image with Yocto

Yocto + Pi 3 « in a nutshell »

- Installing Poky (Yocto reference distro)
  
  $ git clone -b <branch> git://git.yoctoproject.org/poky

- Installing Raspberry Pi layer (meta-raspberrypi)
  
  $ cd poky
  $ git clone -b <branch> git://git.yoctoproject.org/meta-raspberrypi

- Creating work directory
  
  $ source oe-init-build-env rpi3-build

- Adding Pi layer to bblayers.conf
  
  $ bitbake-layers add-layer ../meta-raspberrypi

- Adding machine type to local.conf
  
  echo "MACHINE = "raspberrypi3"" >> conf/local.conf

- Building minimal image
  
  $ bitbake core-image-minimal

- Writing Micro-SD
  
  $ sudo dd if=core-image-minimal-raspberrypi3.rpi-sdimg of=/dev/mmcblk0
• Several talks about it (FOSDEM, etc.)
• 2 ways
  – PREEMPT_RT (single kernel patch)
  – RTAI / Xenomai (co-kernel approach - Cobalt, kernel patch + libs)
    – Xenomai can work on top PREEMPT_RT (Mercury)
• PREEMPT_RT is “the official Linux RT patch” since 2015 (Thomas Gleixner, Steven Rostedt)
• RTAI is a fork for RTLinux (Paolo Mantegazza)
• Xenomai is a fork for RTAI (Philippe Gerum)
• Co-kernel is more complex to use but more efficient (2x+)
  – Hardware support
  – Specific (RT) kernel interface (RTDM)
  – Application design (migration problem)
• Yocto and Buildroot are “build systems”
• Easy to build RT image in Buildroot
  – PREEMPT_RT is a kernel patch
  – Xenomai / RTAI support available
• PREEMPT_RT support available in Yocto
  – Dedicated kernel recipe = linux-yocto-rt
  – Image recipe = core-image-rt (depending on linux-yocto-rt)
  – Only for test (?)

$ grep COMPATIBLE linux/linux-yocto-rt_4.12.bb COMPATIBLE_MACHINE = "(qemux86|qemux86-64|qemuarm|qemuppc|qemumips)"

• Support for Xenomai in meta-eldk (very old)
Extending recipe

- Updating recipe (.bb) with a .bbappend
- Default logo in meta/recipes-core/psplash

```
meta/recipes-core/psplash/
├── files
│   ├── splash-init
│   └── splash-poky-img.h
└── splash_git.bb
```

- Yocto logo in meta-poky/recipes-core/psplash

```
meta-poky/recipes-core/psplash
├── files
│   └── splash-poky-img.h
└── splash_git.bbappend
```

- Enabling I²C for Pi in rpi_config_git.bbappend

```
do_deploy_append() {
    # Enable i2c by default
    echo "dtparam=i2c_arm=on" >> ${DEPLOYDIR}/bcm2835-bootfiles/config.txt
}
```
Building RT image with Yocto

Using real hardware

- Just “extend” kernel for PREEMPT_RT
  - Create a new layer meta-<rt-test-name>
  - Add a recipes-kernel/linux-<board>-rt directory

```
meta-article-bis/
├── conf
 │   └── layer.conf
├── recipes-kernel
 │   └── linux-rpi3
      ├── files
      │   ├── defconfig
      │   └── patch-4.4.50-rt63.patch
      └── linux-rpi3_4.4.bbappend
```

- PREEMPT_RT kernel branch available for some boards (BBB)
- Specific layer is needed for Xenomai
• More difficult as:
  - Need to run `prepare-kernel.sh` script to apply kernel patch (I-pipe)
  - Need to install user-space files
• New layer `meta-xenomai`
  - Kernel recipe `linux-xenomai-<board>`
  - User-space recipe (Autotools based)
• Kernel recipe adds `do_prepare_kernel()` function for patch (Cobalt support only)
• Executed before `do_configure()` (use `addtask`)
• Add *meta-xenomai* layer path
  
  $ cd <path>/poky
  $ git clone https://github.com/pficheux/meta-xenomai.git
  $ cd rpi3-build
  $ bitbake-layers add-layer ../meta-xenomai

• Add Xenomai support to *local.conf* or dedicated image recipe
  
  PREFERRED_PROVIDER_virtual/kernel = "linux-xenomai-<board>"
  IMAGE_INSTALL_append = " xenomai rt-tests"
• Test on Yocto 2.3 + 2.4
• New boards (mostly done by end-users)
References

- Yocto layer meta-xenomai https://github.com/pficheux/meta-xenomai
- Xenomai project http://xenomai.org/