State of OpenEmbedded Internal Toolchain and SDKs

Khem Raj

Embedded Linux Conference
11-13 April 2011, San Francisco
Agenda

➢ Introduction
➢ Internal Toolchain
  ▶ Current state
  ▶ Features
  ▶ Pains
➢ SDKs
  ▶ Build your own SDK
  ▶ Using SDK
➢ Q & A
What is OpenEmbedded?

- Framework to build Embedded Linux distributions
- Meta-data describing how to build software
- Includes tools to help build various RFS types.
- Learn more at http://www.openembedded.org
- http://wiki.openembedded.net/index.php/Mailing_lists
- IRC channel #oe on irc.freenode.net
Internal Toolchain

➢ GNU Toolchain based
➢ Supports C, C++, Fortran, objective-C
➢ Choice of system C libraries
  ▪ Glibc, eglibc, uClibc
Internal Toolchain

- Multiple versions of Toolchain components available
  - GCC 4.5, 4.4, 4.3, 4.2
  - Glibc 2.10.1, 2.9
  - Eglibc 2.12, 2.11, 2.10, 2.9
  - uClibc 0.9.30, 0.9.31, git
    - Option to select threading model (NPTL/LT)
- Distributions select sane combinations
Internal Toolchain

- Build required Native packages
- Bootstrap Cross Toolchain
- Rest of build process
- External or prebuilt toolchain
Internal Toolchain

- Supports Multiple architectures
  - ARM, MIPS/MIPS64, powerpc, x86, SuperH …
- Detects host include poisoning
- ARM
  - Can be configured for hardfp or softfp floating ABI
  - Chosen by setting $\text{ARM\_FP\_ABI} = \{\text{hardfp}|\text{softfp}\}$
    - Only supported with gcc 4.5
  - Linaro gcc 4.5 improvements
- MIPS
  - Configured with --with-mips-plt
Internal Toolchain

Cross binutils

GCC cross bootstrap stage 1

C Library headers/runtime

GCC cross bootstrap stage 2

C Library(uClibc/eglibc/glibc)

GCC cross Final

Native sysroot
Internal Toolchain - Niggles

- Rebuilding not so straightforward
- Target support libraries and headers intertwined with cross compiler
- No support for multilibs
OpenEmbedded SDK

- Some SDKs pre-exist in OE metadata
  - meta-toolchain, meta-toolchain-qte, meta-toolchain-shr etc.
- But that's not all
OpenEmbedded SDK

➢ OE's SDK generation is easily customizable
➢ SDK has
  ➢ Host packages e.g. cmake, qemu
  ➢ Target packages e.g. libc headers/libraries
OpenEmbedded SDK

- **Host packages** e.g. cmake
- **Target Packages** e.g. libc-dev
- **Extra host packaged**
- **Additional Target packages**
Creating your own SDK

- Host side
  - Create sdk recipes that need to be part of SDK
    
    ```
    BBCLASSEXTEND = "sdk"
    ```
  - Create new task in recipes/tasks task-<YOURS>-toolchain-host.bb
  - Append your packages-<sdk> to RDEPENDS_${PN}
    - ```
      RDEPENDS_${PN} += "your-recipe-sdk"
    ```
Creating Your own SDK

➢ Target side

➢ -dev packages provide headers/libraries and other useful files for development

➢ Create a new task task-<YOUR>-toolchain-target.bb

➢ Append -dev packages of your recipe to RDEPENDS_${PN} along with task-sdk-bare

RDEPENDS_${PN} += "your-lib-dev"
Creating your own SDK

- Create `meta-toolchain-<YOUR>.bb` in `recipes/meta`
- Set `TOOLCHAIN_TARGET_TASK = "task-<YOUR>-toolchain-target"`
- Set `TOOLCHAIN_HOST_TASK = "task-<YOUR>-toolchain-host"`
- Add `require meta-toolchain.bb`
- Set `SDK_SUFFIX` to something
Creating your own SDK

bitbake meta-toolchain-<YOUR>
Using SDK

- Untar SDK tarball
  ```bash
  tar -C / -xjf angstrom-armv5te-linux-gnueabii-
toolchain-qte.tar.bz2
  ```

- Source the setup script
  ```bash
  source /usr/local/angstrom/arm/environment-setup
  ```

- SDK is configured !!
Future

➢ Use OpenEmbedded Core Features
➢ Multilib support
➢ Upgrade to gcc 4.6
➢ Enable gold along with GNU ld
➢ Automatic regression testing
Thank you