Intel Cup Embedded System Design Contest

Customizing with Yocto

Dexuan Cui

Intel Corporation

Agenda

Intel Cup Embedded System Design Contest

- Yocto Project: Build System, Workflow and Components
- Key concepts in Poky: Configurations files, recipe, class, layer
- Poky directory tree overview
- Add a new package and build our own image
- HOB: a GUI image customization tool

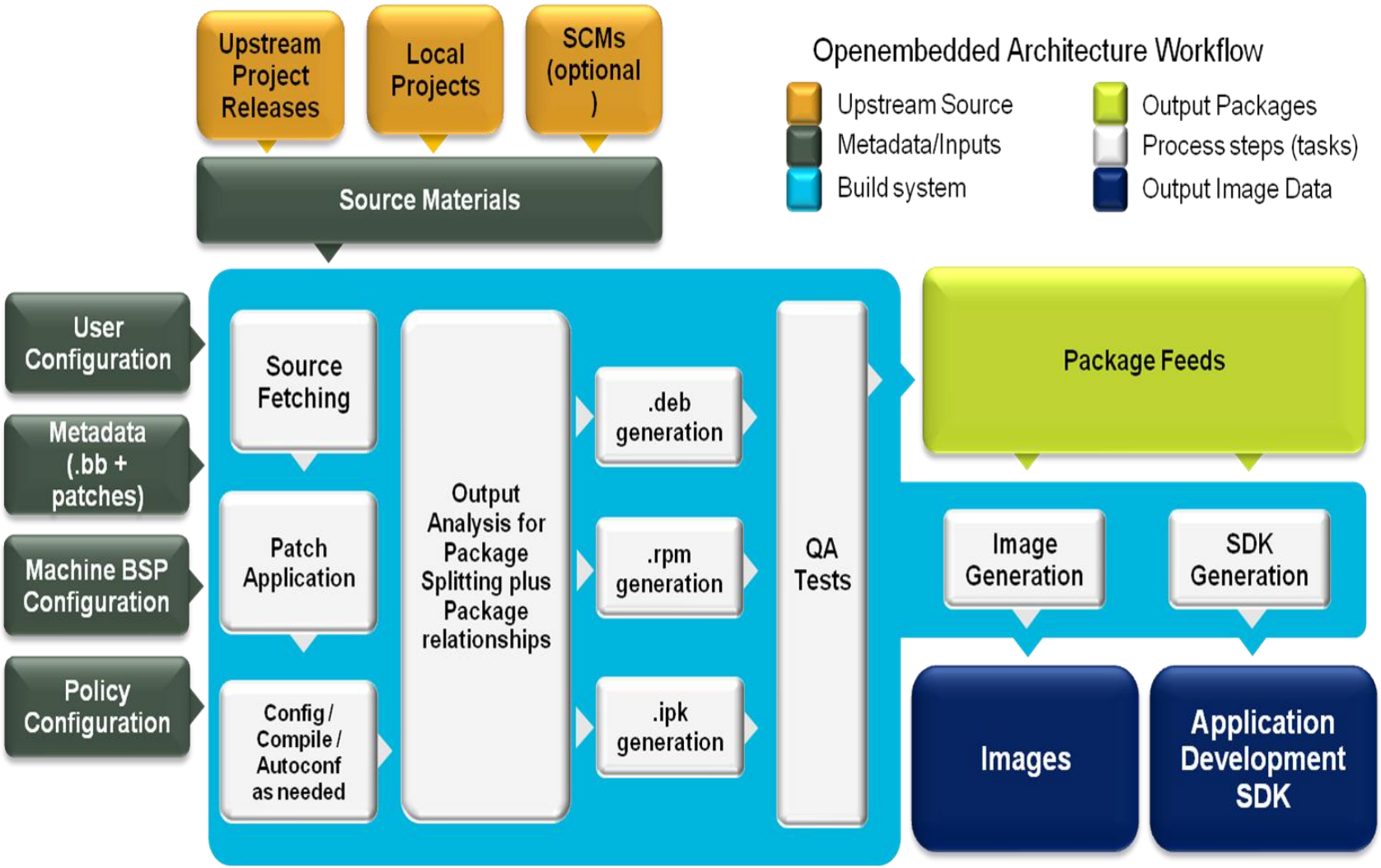
Yocto Project Build System

Intel Cup Embedded System Design Contest

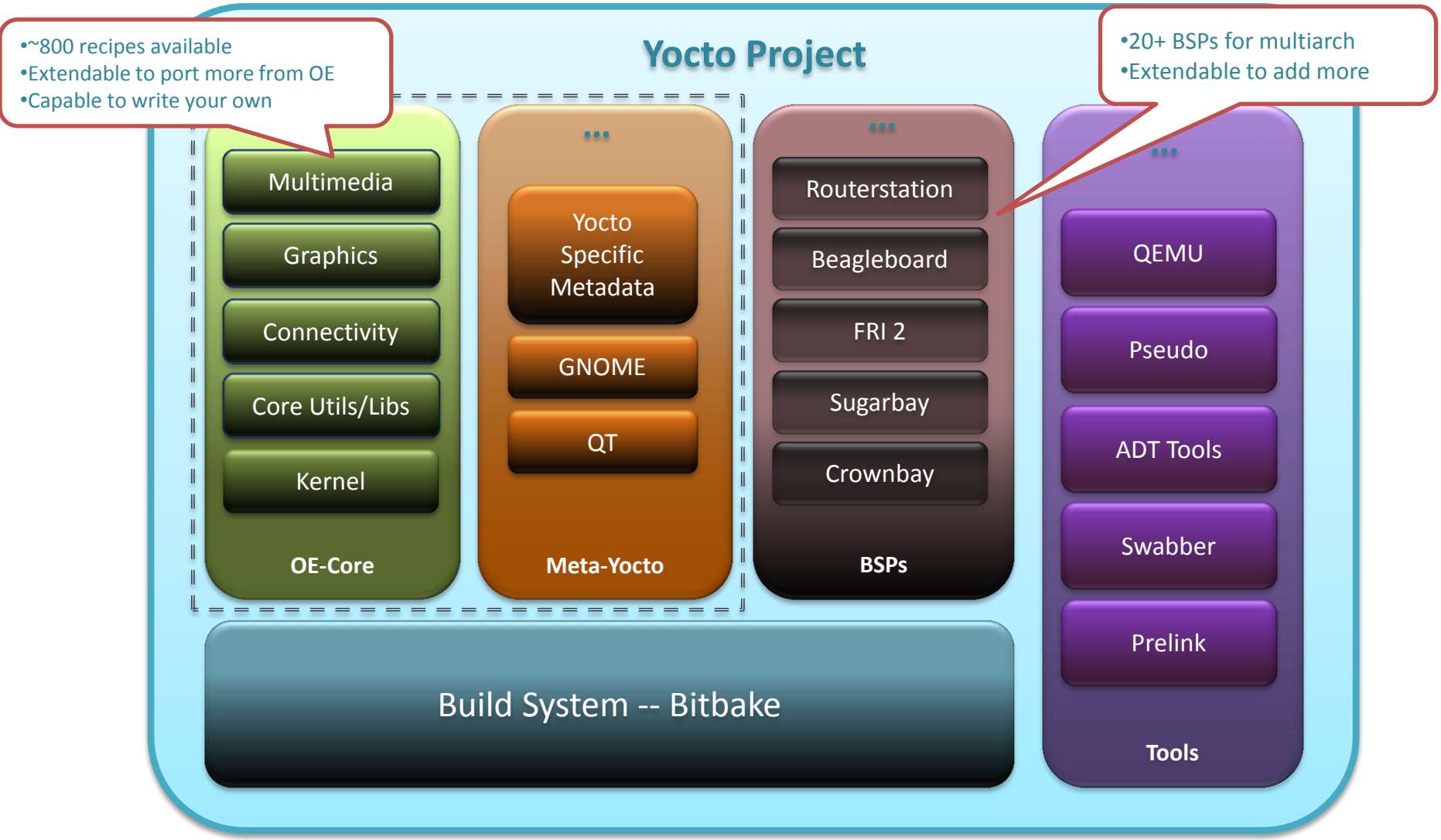
Poky = BitBake + metadata

- **Poky – build system used by the Yocto Project**
- **Bitbake – a task executor and scheduler**
- **Metadata – task definitions**
  - Configuration (.conf) – global definitions of variables
  - Recipes (.bb) – the logical units of software/images to build
  - Classes (.bbclass) – encapsulation and inheritance of build logic, packaging, etc.

Yocto Project Workflow



Yocto Components



Configuration files in poky

- Configuration (*.conf files) – definition of variables:
  - meta/conf/bitbake.conf (global bitbake variables)
  - distro/poky.conf (Yocto policy config variables)
  - machine/ atom-pc.conf(machine-specific variables)
  - build/conf/local.conf (local user-defined variables)

Example: **local.conf**

- BB_NUMBER_THREADS = "8"
- PARALLEL_MAKE = "-j 4"
- MACHINE ?= "qemux86"
- DL_DIR ?= "/distro/poky/sources/"
- DISTRO ?= "poky"
- PACKAGE_CLASSES ?= "package_rpm”
- **EXTRA_IMAGE_FEATURES = "debug-tweaks"**

Recipe

- A **recipe**(.bb files) is a set of instructions for building packages, including:
  - Where to obtain the upstream sources and which patches to apply
  - Dependencies (on libraries or other recipes)
  - Configuration/compilation options
  - Define what files go into what output packages
  - Etc.

Example: a recipe

meta/recipes-extended/msmtp/msmtp_1.4.24.bb

```
SUMMARY = "msmtp is an SMTP client."
DESCRIPTION = "A sendmail replacement for use in MTAs like mutt"
HOMEPAGE = "http://msmtp.sourceforge.net/"
SECTION = "console/network”
LICENSE = "GPLv3"
DEPENDS = "zlib gnutls"
PR = "r0”
LIC_FILES_CHKSUM =
                “file://COPYING;md5=d32239bcb673463ab874e80d47fae504”
SRC_URI =
“http://sourceforge.net/projects/msmtp/files/msmtp/${PV}/${PN}-${PV}.tar.bz2”
SRC_URI[md5sum] = "5fb7ae88186624cdb125d3efad3fdc16"
SRC_URI[sha256sum]
="269cd30eeb867167c6a599e23399f4fc24196fcdef3bac5b120d806b3b421810”
inherit gettext autotools
…
```

Specify build dependencies

Where to fetch source

Inherit internal classes:
gettext/autotools

Then Poky takes care of all the rest of the build tasks in the background:
do_fetch
do_unpack
do_patch
do_configure
do_compile
do_install
do_populate_sysroot
do_package
do_rootfs
…

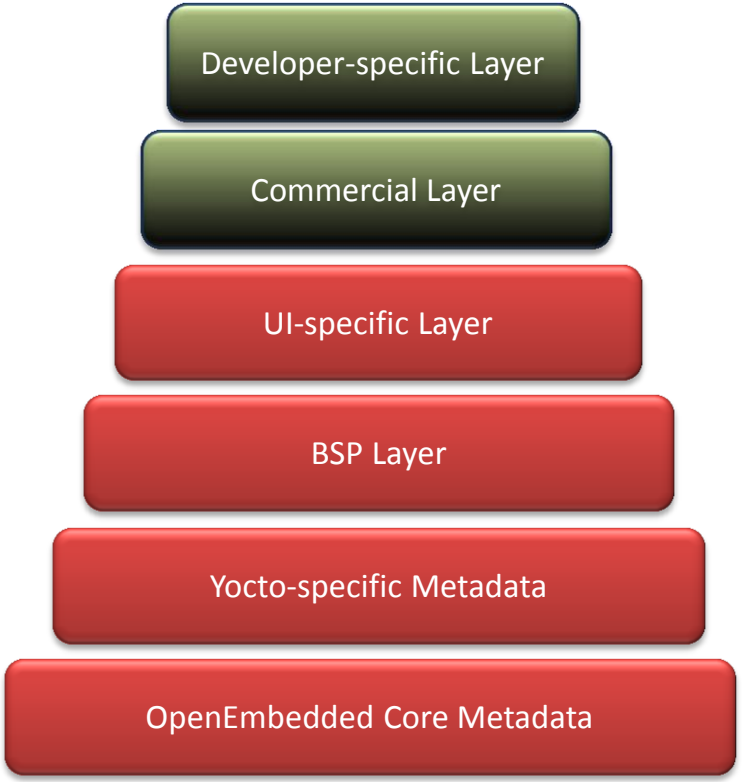
Class (.bbclass files)

- Provide common functionalities.
  - Make createing new recipes become easy
  - Avoid code duplication
  - All the class files live in meta/class/*.bbclass
- Examples 1: autotools.bbclass
  - Configure the code and generate Makefile(s)
  - Compile the source code
  - Install the built output
- Examples 2: package_rpm.bbclass
  - Package the build output into RPM format
  - Handle the dependencies among packages

Layer

- Stacked layers allow deep customization with low maintenance cost.
  - Add new recipes
  - Tune arch specific flags
  - Override package options

- **A sample stack in the right side**
  - Commercial layers from embedded OSVs or ISVs
  - BSP layers from Intel or other silicon vendors



Example: Add a New Layer

bblayers.conf:

```
BBLAYERS ?= " \
        /home/build/poky/meta \
        /home/build/poky/meta-yocto \
        /home/build/bsp/meta-intel \
        /home/build/bsp/meta-intel/meta-esdc2012\
        "
```

Specify to add 2 layers

Poky Directory Tree

Intel Cup Embedded System Design Contest

- **bitbake/:** the BitBake utility itself
- **documentation/:** documentation stuff
- **scripts/:** various support scripts (e.g, runqemu)
- **meta/conf/:** important configuration files, **bitbake.conf,** reference distro config, machine configs for qemu architectures
- **meta/classes/:** bitbake classes
- **meta/recipes-*/:** various recipes
- **build/:** anything generated with a build
- **build/conf/:** including **bblayer.conf** and **local.conf**

Add a new package: helloworld

Intel Cup Embedded System Design Contest

cd poky/meta/recipes-support/ && tree helloworld/
helloworld/
|-- files
| `-- helloworld.c
`-- helloworld_1.0.bb

$ bitbake helloworld

We will get the package helloworld_1.0-r0_i586.rpm

```
DESCRIPTION = "Simple helloworld application"
SECTION = "examples"
LICENSE = "MIT"
LIC_FILES_CHKSUM = "file://${COMMON_LICENSE_DIR}/MIT;
                    md5=0835ade698e0bcf8506ecda2f7b4f302"

PR = "r0"
SRC_URI = "file://helloworld.c"
S = "${WORKDIR}"

do_compile() {
   ${CC} helloworld.c -o helloworld
}

do_install() {
   install -d ${D}${bindir}
   install -m 0755 helloworld ${D}${bindir}
}
```

Build our own image

Intel Cup Embedded System Design Contest

- Poky has many existing image definitions
  - core-image-minimal, core-image-sato,…
    - Defined by recipes: core-image-minimal.bb, core-image-sato.bb
    - We can create our own image type, too.
- Install the new package into the target image
  - add a line into local.conf: IMAGE_INSTALL_append = " helloworld”
    - $ bitbake core-image-sato
    - $ runqemu qemux86 and in the target we see /usr/bin/helloworld
- More complex examples, see http://www.yoctoproject.org/docs/current/poky-ref-manual/poky-ref-manual.html

HOB: the GUI image customization tool
Human Oriented Builder

Intel Cup Embedded System Design Contest

$ cd poky

$ source oe-init-build-env

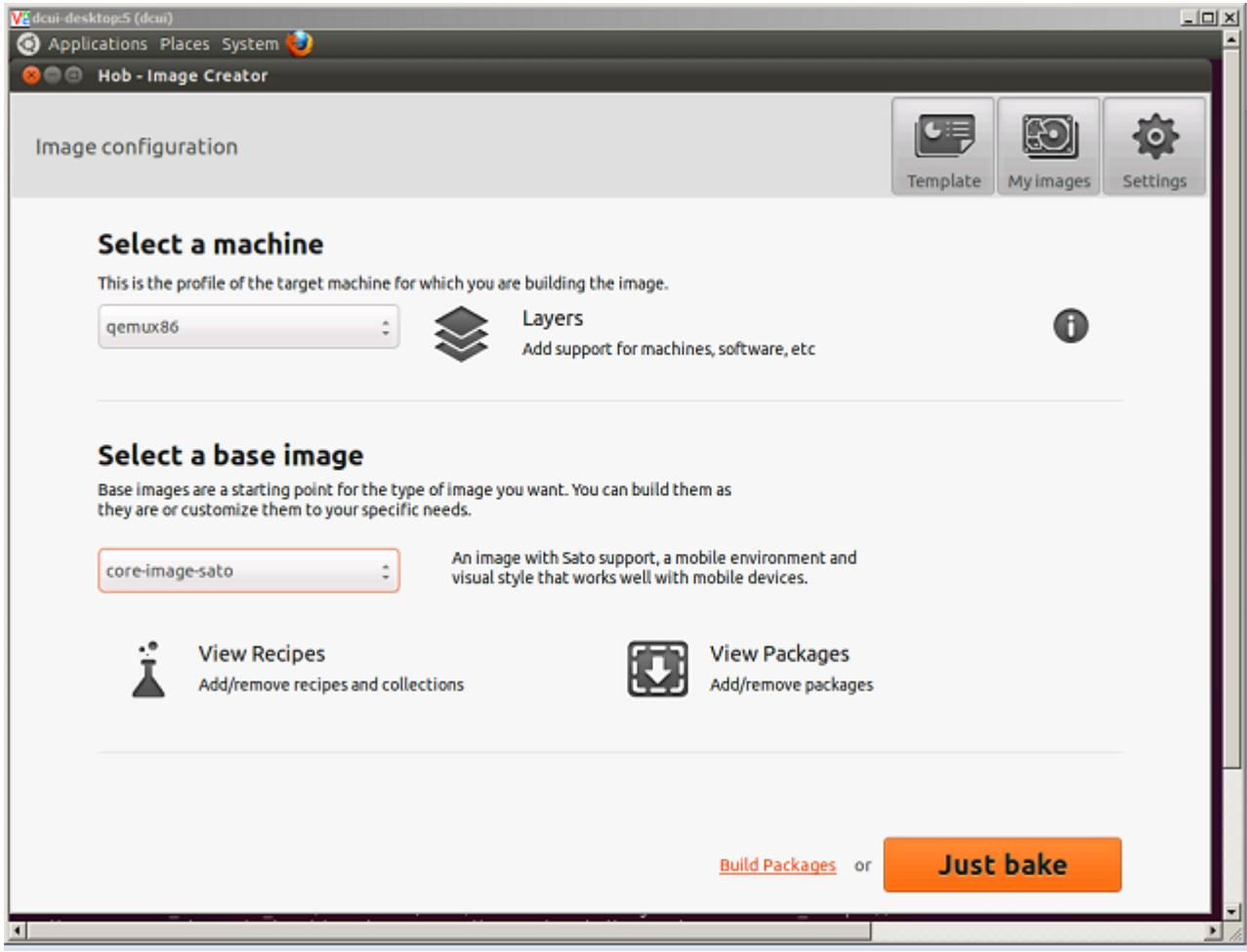
$ hob

- Provides full functionality with parameters

GTK-base Hob

- Wizard-like from configuration selection to image deployment
- Customize your image as you want

Web-based Hob is coming soon.

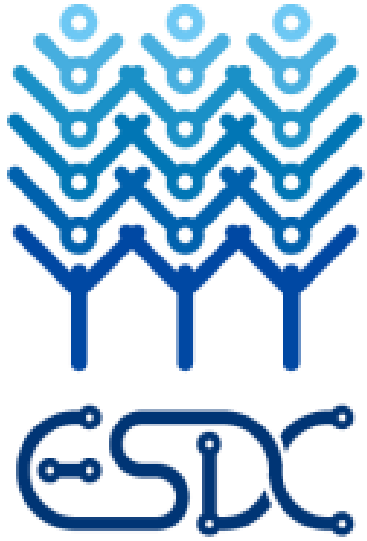


Yocto Resources

- **The Yocto Project is an open source project, and aims to deliver an open standard for the embedded Linux community and industry**
- **Development is done in the open through public mailing lists:**
  - openembedded-core@lists.openembedded.org,
  - poky@yoctoproject.org, yocto@yoctoproject.org
- **And public code repositories:**
  - http://git.yoctoproject.org, http://git.openembedded.net
- **Documentations:**
  - http://www.yoctoproject.org/documentation
- **Bug reports and feature requests:**
  - http://bugzilla.yoctoproject.org

Join us! Join the community!

Legal Information



Intel Cup Embedded System Design Contest