CIP Mini Summit: Introduction of CIP Core

Kazuhiro Hayashi, Toshiba Corporation
Jan Kiszka, Siemens AG
Embedded Linux Conference Europe 2019, Oct. 31th 2019
Introduction

CIP Core packages (tens)
CIP kernel (10+ years maintenance, based on LTS kernels)
additional packages (hundreds)

company-specific middleware and applications

Establishes an “Open Source Base Layer (OSBL)”
CIP Core

- One of the CIP projects focusing on user land software and tools

- Goals
  - Define a list of “CIP Core packages” maintained for long-term
  - Provide a reference implementation including “CIP Core packages”
  - Test the implementation on the “CIP reference hardware”

<table>
<thead>
<tr>
<th></th>
<th>SLTS kernel</th>
<th>Real-time</th>
<th>Testing</th>
<th>CIP Core</th>
<th>Security WG(*)</th>
<th>Software update WG</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>2</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>3</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>4</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>5</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>6</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>

(*): Workgroup

- Industrial grade ✔ ✔ ✔ ✔ ✔
- Sustainability ✔ ✔ ✔ ✔ ✔
- Security ✔ ✔ ✔ ✔ ✔

CIP Projects and its scopes

Embedded Linux Conference Europe 2019
CIP Core: Position in CIP Projects

- CIP Core Reference Implementation
  - CIP Core Packages
  - Build Tool
  - SLTS kernel
  - Real-Time

- CIP Reference Hardware
  - Tested on targets

- Software update WG
  - Request Packages & Configurations
  - Discussion & Decide
  - CIP members

- Security WG
  - Build & Integrate

Testing (CI)
CIP Core: Position in CIP Projects

CIP Core: Position in CIP Projects

- CIP Core
  - CIP Core Reference Implementation
  - Build Tool
- SLTS kernel
- Real-Time
- CIP Reference Hardware

Testing (CI)

Deploy

Discussion & Decide

CIP members

Request Packages & Configurations

Build & Integrate

Software update WG

Security WG

CIP members

Tested on targets

CIP Core Packages

CIP members

Embedded Linux Conference Europe 2019
CIP Core Package List

• Package Decision Process
  • [https://gitlab.com/cip-project/cip-core/cip-pkglist](https://gitlab.com/cip-project/cip-core/cip-pkglist)

• Components
  • Document for proposal, review, package list update
  • Helper scripts for proposal and updating the package list
CIP Core Package List

- Workflow

- CIP member company A
  - generate-proposal.py
  - proposal_A.yml

- CIP member company B
  - generate-proposal.py
  - proposal_B.yml

- CIP member company C
  - generate-proposal.py
  - proposal_C.yml

CIP Core NG companies (Reviewers)

- proposal_A.yml
  - Accepted

- proposal_B.yml
  - Accepted

- pdp-helper.py

Register proposed packages (add-proposal)

- pkglist.yml

CIP maintained package list
CIP Core: Position in CIP Projects

- CIP members
- Discussion & Decide
- Request Packages & Configurations
- Software update WG
- CIP Core
- CIP Core Reference Implementation
- Build Tool
- Build & Integrate
- SLTS kernel
- Real-Time
- CIP Reference Hardware
- Tested on targets
- Security WG
- CIP Core Packages
- Deploy
CIP Core: Implementation

• Debian-based implementation
  • Mature, high-quality, mainstream distro.
  • Many new & old architecture supports
  • Suitable for small and big installations
  • Security updates

• Profiles

<table>
<thead>
<tr>
<th>Approach</th>
<th>Generic profile</th>
<th>Tiny profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tool</td>
<td>Binary packages</td>
<td>Source packages</td>
</tr>
<tr>
<td></td>
<td>Isar</td>
<td>Deby</td>
</tr>
</tbody>
</table>

Build tools
- Isar
- meta-debian

Metadata
- Add packages, send patches, etc.
- Funding to Debian LTS

Build tools
- Metadata

CIP Core Project
- Implement

Binary packages
- Source packages

Use
- Contribution

Funding to Debian LTS

Generic profile
- Tiny profile

CIP Core

Isar

Source packages
- Binary packages

Binary packages
- Source packages

Use

Embedded Linux Conference Europe 2019
CIP Core: Implementation

- Debian-based implementation
  - Mature, high-quality, mainstream distro.
  - Many new & old architecture supports
  - Suitable for small and big installations
  - Security updates

- Profiles

<table>
<thead>
<tr>
<th>Approach</th>
<th>Generic profile</th>
<th>Tiny profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tool</td>
<td>Binary packages</td>
<td>Source packages</td>
</tr>
<tr>
<td>Tool</td>
<td>Isar</td>
<td>Deby</td>
</tr>
</tbody>
</table>

- Debian
- Maturity, high-quality, mainstream distro.
- Many new & old architecture supports
- Suitable for small and big installations
- Security updates

- Profiles

<table>
<thead>
<tr>
<th>Generic profile</th>
<th>Tiny profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary packages</td>
<td>Source packages</td>
</tr>
<tr>
<td>Isar</td>
<td>Deby</td>
</tr>
</tbody>
</table>
### Target Versions of CIP Core

- **CIP kernel & Debian (&Yocto in Tiny profile)**

<table>
<thead>
<tr>
<th>Version</th>
<th>Debian 9 stretch</th>
<th>Debian 10 buster</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIP kernel 4.4</td>
<td>Generic</td>
<td>Unsupported</td>
</tr>
<tr>
<td>CIP kernel 4.19</td>
<td>Generic</td>
<td>Generic, Tiny</td>
</tr>
</tbody>
</table>
CIP Core: Implementation

- Debian-based implementation
  - Mature, high-quality, mainstream distro.
  - Many new & old architecture supports
  - Suitable for small and big installations
  - Security updates

- Profiles

<table>
<thead>
<tr>
<th>Approach</th>
<th>Generic profile</th>
<th>Tiny profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tool</td>
<td>Binary packages</td>
<td>Source packages</td>
</tr>
<tr>
<td></td>
<td>Isar</td>
<td>Deby</td>
</tr>
</tbody>
</table>

- Metadata
  - Build tools
    - Isar
    - meta-debian
  - Contribution
    - Use

- Build tools
  - Add packages, send patches, etc.
  - Funding to Debian LTS

- Implement
  - Generic profile
  - Tiny profile

- CIP Core Project

- Debian

- Binary packages
  - Source packages

- Use

Embedded Linux Conference Europe 2019
Deby (meta-debian)

• Yocto Project extension for using Debian source packages
  • Source code: Debian
  • Build system: Yocto Project
  • https://github.com/meta-debian/meta-debian

• Goals
  • Achieve stability and long-term support with the Yocto advantages

• Features: Yocto based flexibility & extensibility
  • High customizability by own recipes
  • Small footprint (Around 2MB)
  • Various target CPUs and tunings
  • Adaptation to BSP layers provided by board vendors
Deby: How it works

Debian sources

Extra sources

pkgB.bbappend

app.bb

pkgB.bb
pkgC.bb
pkgD.bb

debian-package.bbclass

meta-

debian (Deby)

meta-custom

meta-

debian

meta-bspX

meta-bspY

poky

pkgA.bb

.metconf

.bbclass

bitbake

kernel

rootfs

SDK

QEMU

Board X

Board Y

U-Boot

SDK

U-Boot

SDK

Upstream sources

Debian sources

poky

pkgA.bb

upstream sources

Debian sources

pkgB.bb

pkgC.bb

pkgD.bb

debian-package.bbclass

meta-custom

pkgB.bbappend

app.bb

Debian sources

pkgB.bb

pkgC.bb

pkgD.bb

debian-package.bbclass

meta-

debian (Deby)

meta-bspX

meta-bspY

poky

pkgA.bb

.metconf

.bbclass

bitbake

kernel

rootfs

SDK

QEMU

Board X

Board Y

U-Boot

SDK

U-Boot

SDK

Debian sources

pkgB.bb

pkgC.bb

pkgD.bb

debian-package.bbclass

meta-

debian (Deby)

meta-bspX

meta-bspY

poky

pkgA.bb

.metconf

.bbclass

bitbake

kernel

rootfs

SDK

QEMU

Board X

Board Y

U-Boot

SDK

U-Boot

SDK

Debian sources

pkgB.bb

pkgC.bb

pkgD.bb

debian-package.bbclass

meta-

debian (Deby)

meta-bspX

meta-bspY

poky

pkgA.bb

.metconf

.bbclass

bitbake

kernel

rootfs

SDK

QEMU

Board X

Board Y

U-Boot

SDK

U-Boot

SDK

Debian sources

pkgB.bb

pkgC.bb

pkgD.bb

debian-package.bbclass

meta-

debian (Deby)

meta-bspX

meta-bspY

poky

pkgA.bb

.metconf

.bbclass

bitbake

kernel

rootfs

SDK

QEMU

Board X

Board Y

U-Boot

SDK

U-Boot

SDK

Debian sources

pkgB.bb

pkgC.bb

pkgD.bb

debian-package.bbclass

meta-

debian (Deby)

meta-bspX

meta-bspY

poky

pkgA.bb

.metconf

.bbclass

bitbake

kernel

rootfs

SDK

QEMU

Board X

Board Y

U-Boot

SDK

U-Boot

SDK

Debian sources

pkgB.bb

pkgC.bb

pkgD.bb

debian-package.bbclass

meta-

debian (Deby)

meta-bspX

meta-bspY

poky

pkgA.bb

.metconf

.bbclass

bitbake

kernel

rootfs

SDK

QEMU

Board X

Board Y

U-Boot

SDK

U-Boot

SDK

Debian sources

pkgB.bb

pkgC.bb

pkgD.bb

debian-package.bbclass

meta-

debian (Deby)

meta-bspX

meta-bspY

poky

pkgA.bb

.metconf

.bbclass

bitbake

kernel

rootfs

SDK

QEMU

Board X

Board Y

U-Boot

SDK

U-Boot

SDK

Debian sources

pkgB.bb

pkgC.bb

pkgD.bb

debian-package.bbclass

meta-

debian (Deby)

meta-bspX

meta-bspY

poky

pkgA.bb

.metconf

.bbclass

bitbake

kernel

rootfs

SDK

QEMU

Board X

Board Y

U-Boot

SDK

U-Boot

SDK

Debian sources

pkgB.bb

pkgC.bb

pkgD.bb

debian-package.bbclass

meta-

debian (Deby)

meta-bspX

meta-bspY

poky

pkgA.bb

.metconf

.bbclass

bitbake

kernel

rootfs

SDK

QEMU

Board X

Board Y

U-Boot

SDK

U-Boot

SDK

Debian sources

pkgB.bb

pkgC.bb

pkgD.bb

debian-package.bbclass

meta-

debian (Deby)

meta-bspX

meta-bspY

poky

pkgA.bb

.metconf

.bbclass

bitbake

kernel

rootfs

SDK

QEMU

Board X

Board Y

U-Boot

SDK

U-Boot

SDK

Debian sources

pkgB.bb

pkgC.bb

pkgD.bb

debian-package.bbclass

meta-

debian (Deby)

meta-bspX

meta-bspY

poky

pkgA.bb

.metconf

.bbclass

bitbake

kernel

rootfs

SDK

QEMU

Board X

Board Y

U-Boot

SDK

U-Boot

SDK

Debian sources

pkgB.bb

pkgC.bb

pkgD.bb

debian-package.bbclass

meta-

debian (Deby)

meta-bspX

meta-bspY

poky

pkgA.bb

.metconf

.bbclass

bitbake

kernel

rootfs

SDK

QEMU

Board X

Board Y

U-Boot

SDK

U-Boo
• Integration System for Automated Root filesystem generation
  https://github.com/ilbers/isar

• Goals
  • Build systems in a Debian way
  • Developer-centric workflow: One-command building
  • Make customizations easy and repeatable
  • Efficient building

• The best of both worlds
  • Debian: Tested binary packages, tools, security updates
  • OpenEmbedded / Yocto: bitbake, recipes, layers

• Reuse Yocto knowledge of developers
1. debootstrap Debian for target, also for host if cross-building
2. Create buildchroots (target and host)
3. Build custom Debian packages
   • pre-debianized packages
   • ad-hoc debianized packages (customizations, u-boot, kernel, ...)
4. Assemble rootfs
   • debootstrap output
   • external packages
   • self-built packages
5. Run images (typically wic)
   • Filesystem image generation
   • Partitioning
   • Bootloader installation and configuration
CIP Core: Implementation

- Debian-based implementation
  - Mature, high-quality, mainstream distro.
  - Many new & old architecture supports
  - Suitable for small and big installations
  - Security updates

- Profiles

<table>
<thead>
<tr>
<th>Approach</th>
<th>Generic profile</th>
<th>Tiny profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tool</td>
<td>Binary packages</td>
<td>Source packages</td>
</tr>
<tr>
<td></td>
<td>Isar</td>
<td>Deby</td>
</tr>
</tbody>
</table>

- Debian
  - Build tools
    - Isar
    - meta-debian
  - Add packages, send patches, etc.
  - Funding to Debian LTS

- Build tools
  - Isar
  - meta-debian

- Metadata
  - Build tools
  - Isar
  - meta-debian

- Use
  - Binary packages
  - Source packages

- Use
  - Contribution
  - Generic profile
  - Tiny profile

- Implement
  - CIP Core Project
  - Generic profile
  - Tiny profile

- CIP Core Project
CIP Core Generic Profile

- Repository
  - https://gitlab.com/cip-project/cip-core/isar-cip-core

- Recipes (A layer for Isar)
  - Distro settings to specify kernel & Debian version
  - Machine settings for CIP reference H/W
  - Other common settings for rootfs

- Kas configurations
  - Settings for each board
    - board-bbb.yml, board-iwg20m.yml, etc.
  - Optional feature changes
    - opt-rt.yml, etc.

```conf
iwg20m.conf  483 Bytes

1 #
2 # CIP Core, generic profile
3 #
4 # Copyright (c) Toshiba corp. 2019
5 #
6 # SPDX-License-Identifier: MIT
7 #
8 DISTRO_ARCH = "armhf"
9 #
10 # see wic/iwg20m.wks
11 IMAGE_TYPE = "wic-img"
12 #
13 # sets serial login getty
14 MACHINE_SERIAL = "ttyS0"
15 BAUDRATE_TTY = "115200"
16 #
17 # kernel version
18 PREFERRED_VERSION_linux-cip ?= "4.4.6"
19 PREFERRED_VERSION_linux-cip-rt ?= "4.4.6"
20 #
21 # Boot partition files
22 DTB_FILE = "r8a7743-iwg20d-q7-dbcm-ca.dtb"
23 KERNEL_IMAGE="zImage"
24 IMAGE_BOOT_FILES = "${KERNEL_IMAGE} ${DTB_FILE}"
```

conf/machine/iwg20m.conf
CIP Core Generic Profile

isar-cip-core (kas configs)
- opt-rt.yml
- opt-4.4.yml
- board-bbb.yml
- board-iwg20m.yml
- kas.yml

isar-cip-core (Recipes)
- recipes-kernel
- recipes-core
- bbb.conf
- iwg20m.conf

isar (meta)

kas.yml

```yaml
header:
  version: 8

distro: cip-core-buster

repos:
  cip-core:
    isar:
      url: https://github.com/llbers/isar
      refspec: bdf8d29eacfde391e4e17a9b953828723cd9bea0
      layers:
        meta:
        bblayers_conf_header:
          standard: |
            LCONF_VERSION = "$"
            BBPATH = "$(TOPDIR)"
            BBFILES ?= ""
        local_conf_header:
          standard: |
            CONF_VERSION = "$"
            cross: |
            ISAR_CROSS_COMPILE = "$"
            root_password: |
            USERS = "$"
```
CIP Core Tiny Profile

• Repository
  • [https://gitlab.com/cip-project/cip-core/deby](https://gitlab.com/cip-project/cip-core/deby)

• Recipes (A layer for poky + meta-debian)
  • Target poky branch: **warrior** (Yocto Project 2.7)
  • Machine settings for CIP reference H/W

• Kas configurations
  • Settings for each board
    • kas-bbb.yml, kas-iwg20m.yml, etc.
CIP Core Tiny Profile

**deby (kas configs)**
- kas-bbb.yml
- board-iwg20m.yml

**deby (Recipes)**
- recipes-kernel
- bbb.conf
- iwg20m.conf

**meta-debian**

**poky (meta)**
- kas.yml
Example: Building Images for BeagleBone Black

• Generic profile (Isar)

```bash
$ git clone https://gitlab.com/cip-project/cip-core/isar-cip-core && cd isar-cip-core
$ wget https://raw.githubusercontent.com/siemens/kas/master/kas-docker
$ chmod a+x kas-docker
$ ./kas-docker --isar build kas.yml:board-bbb.yml
$ dd if=/path/to/cip-core-image-cip-core-buster-bbb.wic.img of=/dev/mmcblk0 ...
```

• Tiny profile (Deby)

```bash
$ git clone https://gitlab.com/cip-project/cip-core/deby && cd deby
$ ./scripts/setup-kas-docker.sh
$ ./kas-docker build kas-bbb.yml
```
## Preferred Use Cases

<table>
<thead>
<tr>
<th></th>
<th>Isar (Generic Profile)</th>
<th>Deby (Tiny Profile)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Available Packages</td>
<td>= Debian</td>
<td>App. 50 (+ Yocto Extension)</td>
</tr>
<tr>
<td>Footprint</td>
<td>&gt; 100MB</td>
<td>2MB - 100MB</td>
</tr>
<tr>
<td>Compatibility</td>
<td>Debian (Binary packages)</td>
<td>Yocto Project (Recipes)</td>
</tr>
<tr>
<td>Required skill set</td>
<td>Debian (Packaging) bitbake</td>
<td>Yocto Project</td>
</tr>
<tr>
<td>Build time (minimal image)</td>
<td>Around 10min</td>
<td>Around 1h</td>
</tr>
<tr>
<td>Customization needs</td>
<td>Selected packages</td>
<td>Up to toolchain settings</td>
</tr>
<tr>
<td>Fitted systems (Examples)</td>
<td>IoT gateways, edge devices, industrial controllers...</td>
<td>Small IoT devices ...</td>
</tr>
</tbody>
</table>
CIP Core: Position in CIP Projects

- CIP Core
  - Reference Implementation
  - Build & Integrate
  - Build Tool

- Software update WG
- Request Packages & Configurations

- Testing (CI)
  - SLTS kernel
  - Real-Time

- CIP Reference Hardware

- Discussion & Decide
  - CIP members

- Deploy

- Tested on targets

Security WG
CIP Core: Testing Architecture

Build Request

Build Server (GitLab Runners on AWS)

Deploy

Artifact Storage (AWS S3)

Kernel, Boot loader, rootfs

Pull updates

isar

meta-debian

Test Request

Job scheduling & Summarizing results

Built Artifacts

LAVA Master

LAVA Worker

LAVA Worker

LAVA Worker

CIP Reference Hardware

isar-cip-core

GitLab (cip-project/cip-core)

deby

Pull updates

Build Request

Artifact Storage (AWS S3)

Kernel, Boot loader, rootfs

Pull updates

isar-cip-core

GitLab (cip-project/cip-core)
Future Plans for CIP Core Implementation

• Enable direct use in product development
  • Regular releases of tested layer with dependencies
  • Mirroring of source & binary dependencies

• Provide image corresponding to CIP package list

• Integrate and test results of other CIP workgroups
  • Robust system update (Software update WG)
  • Functions to meet cybersecurity standard requirements (Security WG)
Summary

• CIP provides long-term maintained Open Source base layer, consisting of kernel and essential packages
• CIP Core defines package set and ensures integration
• Two implementation flavors available
  • Deby for smaller, Yocto/OE-compatible projects
  • isar-cip-core for medium to larger, Debian-compatible projects
• More product-ready features to come, from software update to security hardening
Questions