Embedded Linux Conference
North America

OpenIoTSummit
North America

OpenEmbedded/Yocto on RISC-V

Khem Raj, Comcast

THE LINUX FOUNDATION

Agenda

- Introduction
- What works
- Demo (maybe)
- Work In progress
- Future work

RISC-V

yocto PROJECT

openembedded

RISC-V

“RISC-V: The Free and Open RISC Instruction Set Architecture”

RISC-V

- RISC-V Open ISA
  - pronounced "risk-five”
  - Licensed under BSD License
  - Started with clean slate in 2010
- RISC-V Foundation
  - https://riscv.org/
- RISC-V GitHub
  - https://github.com/riscv
- Several Commercial adopters

Yocto/OpenEmbedded

“The Yocto Project is not an Embedded Linux Distribution. It creates a custom one for You!”

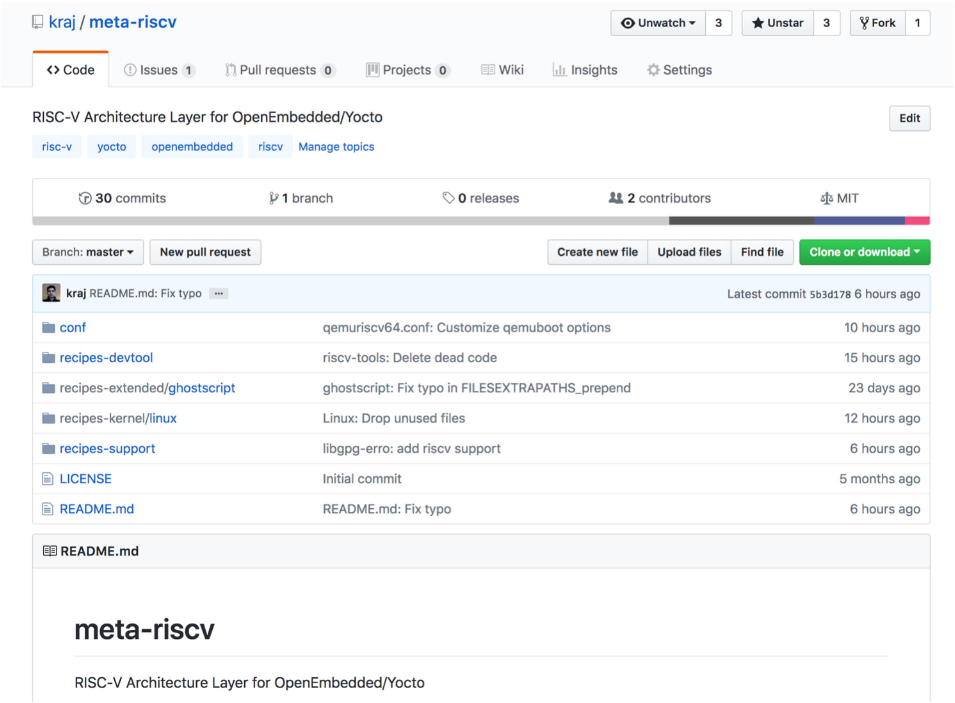
“It is not a single project – its an Ecosystem”

Yocto Project/OpenEmbedded

- Yocto Project uses OpenEmbedded Build system
- Combines Ready to run Linux Distribution with Flexibility to Customize
- System Integration tools
    - Images, SDKs, Extensible SDKs
- Layered Architecture
    - Add only what's needed
    - 1000’s of package recipes available

Porting RISC-V

- A RISC-V Layer
  - Contains RISC-V specific package recipes
  - QEMU machine configuration for RISC-V
    - -machine virt
  - bbapends for packages from other layers
    - Who need a bit of tweaks



kraj / meta-riscv

RISC-V Architecture Layer for OpenEmbedded/Yocto

risc-v yocto openembedded riscv Manage topics

30 commits · 1 branch · 0 releases · 2 contributors · MIT

kraj README.md: Fix typo ... Latest commit 5b3d178 6 hours ago

conf	qemuriscv64.conf: Customize qemuboot options	10 hours ago
recipes-devtool	riscv-tools: Delete dead code	15 hours ago
recipes-extended/ghostscript	ghostscript: Fix typo in FILESEXTRAPATHS_prepend	23 days ago
recipes-kernel/linux	Linux: Drop unused files	12 hours ago
recipes-support	libgpg-erro: add riscv support	6 hours ago
LICENSE	Initial commit	5 months ago
README.md	README.md: Fix typo	6 hours ago

README.md

meta-riscv

RISC-V Architecture Layer for OpenEmbedded/Yocto

Porting RISC-V - Setup

```
mkdir riscv-yocto && cd riscv-yocto
git clone git://git.openembedded.org/openembedded-core
(cd openembedded-core && git clone git://git.openembedded.org/bitbake)
git clone git://github.com/kraj/meta-riscv

. openembedded-core/oe-init-build-env riscv-build
bitbake-layers add-layer ../meta-riscv
```

Porting RISC-V - Build

- Select machine (in local.conf)

```
MACHINE = ”qemuriscv64”
```

- Build

```
bitbake core-image-minimal
```

- Go for lunch ☺

Porting RISC-V - Run

- Run image in QEMU – Use Yocto tooling

```
runqemu nographic
runqemu nographic slirp
```

- Forwards port 22 to 22222 on host
- SSH into RISC-V emulator

```
ssh root@localhost –p 22222
```

RISC-V Porting Status (Core packages)

- Kernel
  - linux-riscv 4.15
- Glibc
  - 2.27 – Upstream
- GCC
  - 7.3 – Upstream
- Binutils
  - 2.30 – Upstream
- GDB
  - riscv-binutils-gdb
- QEMU
  - riscv-qemu
- RISC-V tools
  - bbl, fesvr, spike

What works

- Images

  core-image-minimal
  core-image-base
  core-image-full-cmdline

- SDKs

  bitbake –cpopulate_sdk <image>

- Extensible SDK

  bitbake –cpopulate_sdk_ext <image>

What works

- Init systems
  - Sysvinit
  - Systemd
  - Busybox/mdev

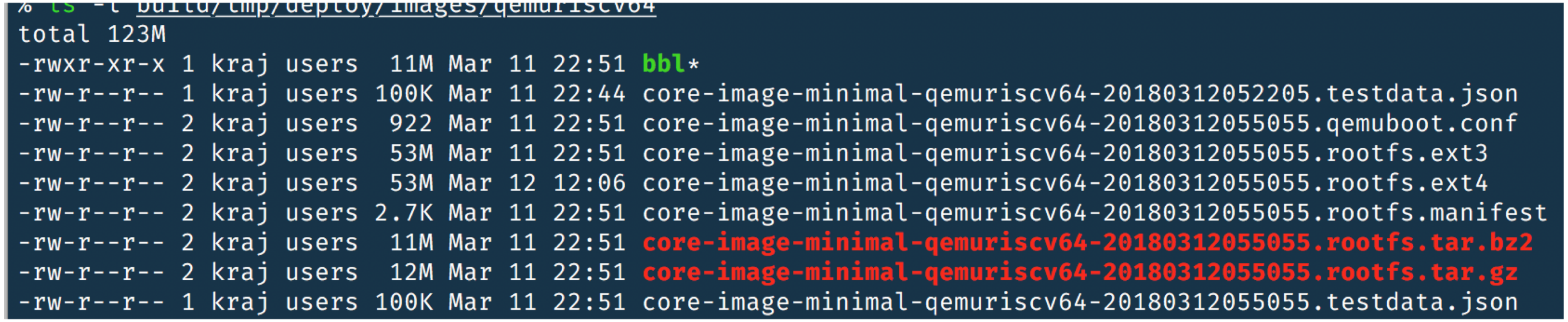
What works

- Running images QEMU
  - runqemu nographic slirp
    - ssh root@localhost –p 22222
  - runqemu nographic
    - Uses tap
    - ssh root@192.168.7.2

Booting in QEMU

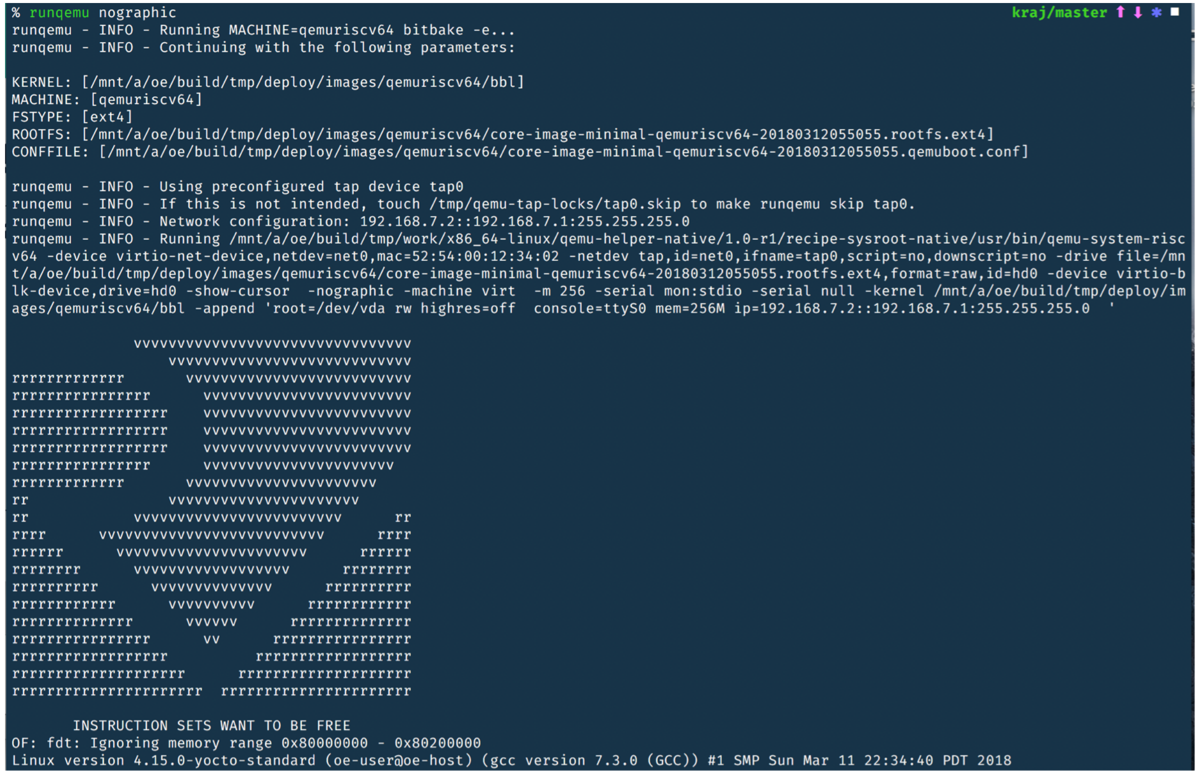
- Kernel
  - Proxy kernel - Bundled with bbl (Berkley bootloader)
  - BBL - meta-riscv/recipes-devtool/riscv-tools/riscv-pk.bb
  - Final kernel image is called ‘bbl’ in DEPLOYDIR
- Rootfs
  - Ext4 image

$ runqemu nographic

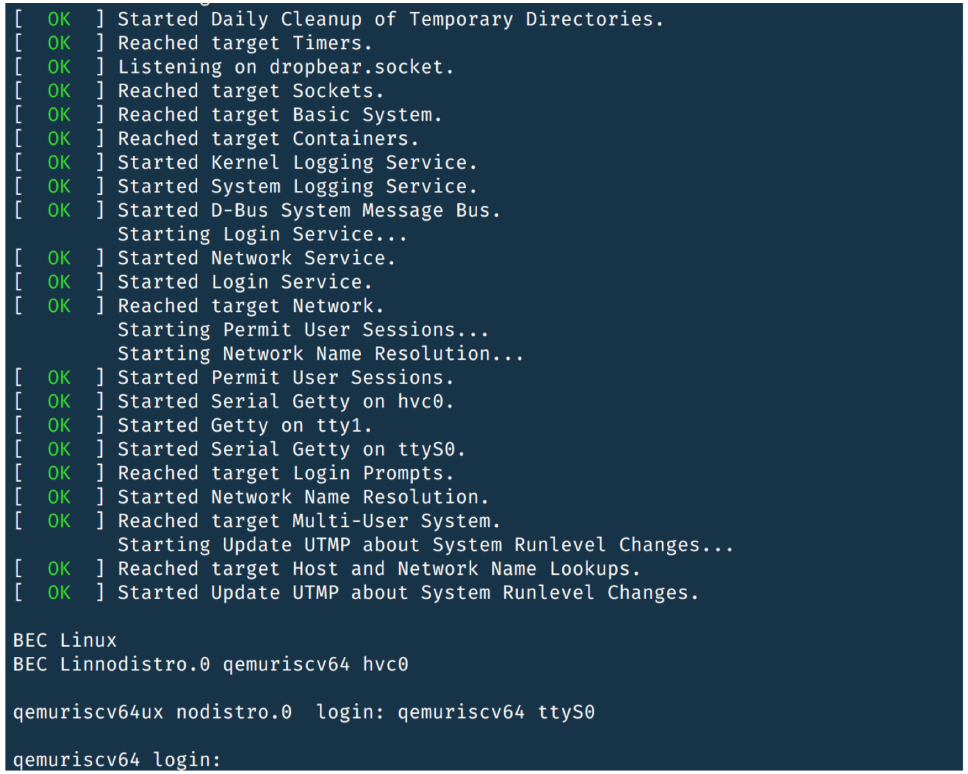


```
% ls -l build/tmp/deploy/images/qemuriscv64
total 123M
-rwxr-xr-x 1 kraj users  11M Mar 11 22:51 bbl*
-rw-r--r-- 1 kraj users 100K Mar 11 22:44 core-image-minimal-qemuriscv64-20180312052205.testdata.json
-rw-r--r-- 2 kraj users  922 Mar 11 22:51 core-image-minimal-qemuriscv64-20180312055055.qemuboot.conf
-rw-r--r-- 2 kraj users  53M Mar 11 22:51 core-image-minimal-qemuriscv64-20180312055055.rootfs.ext3
-rw-r--r-- 2 kraj users  53M Mar 12 12:06 core-image-minimal-qemuriscv64-20180312055055.rootfs.ext4
-rw-r--r-- 2 kraj users 2.7K Mar 11 22:51 core-image-minimal-qemuriscv64-20180312055055.rootfs.manifest
-rw-r--r-- 2 kraj users  11M Mar 11 22:51 core-image-minimal-qemuriscv64-20180312055055.rootfs.tar.bz2
-rw-r--r-- 2 kraj users  12M Mar 11 22:51 core-image-minimal-qemuriscv64-20180312055055.rootfs.tar.gz
-rw-r--r-- 1 kraj users 100K Mar 11 22:51 core-image-minimal-qemuriscv64-20180312055055.testdata.json
```

Booting In QEMU



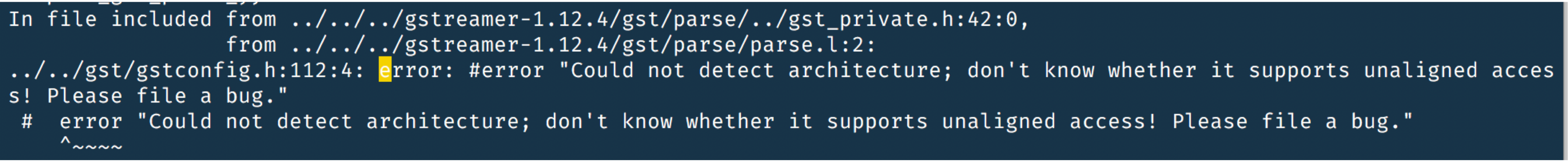
```
% runqemu nographic                                                                                          kraj/master ⬆ ⬇ ✱ ■
runqemu - INFO - Running MACHINE=qemuriscv64 bitbake -e...
runqemu - INFO - Continuing with the following parameters:

KERNEL: [/mnt/a/oe/build/tmp/deploy/images/qemuriscv64/bbl]
MACHINE: [qemuriscv64]
FSTYPE: [ext4]
ROOTFS: [/mnt/a/oe/build/tmp/deploy/images/qemuriscv64/core-image-minimal-qemuriscv64-20180312055055.rootfs.ext4]
CONFFILE: [/mnt/a/oe/build/tmp/deploy/images/qemuriscv64/core-image-minimal-qemuriscv64-20180312055055.qemuboot.conf]

runqemu - INFO - Using preconfigured tap device tap0
runqemu - INFO - If this is not intended, touch /tmp/qemu-tap-locks/tap0.skip to make runqemu skip tap0.
runqemu - INFO - Network configuration: 192.168.7.2::192.168.7.1:255.255.255.0
runqemu - INFO - Running /mnt/a/oe/build/tmp/work/x86_64-linux/qemu-helper-native/1.0-r1/recipe-sysroot-native/usr/bin/qemu-system-risc
v64 -device virtio-net-device,netdev=net0,mac=52:54:00:12:34:02 -netdev tap,id=net0,ifname=tap0,script=no,downscript=no -drive file=/mn
t/a/oe/build/tmp/deploy/images/qemuriscv64/core-image-minimal-qemuriscv64-20180312055055.rootfs.ext4,format=raw,id=hd0 -device virtio-b
lk-device,drive=hd0 -show-cursor  -nographic -machine virt  -m 256 -serial mon:stdio -serial null -kernel /mnt/a/oe/build/tmp/deploy/im
ages/qemuriscv64/bbl -append 'root=/dev/vda rw highres=off  console=ttyS0 mem=256M ip=192.168.7.2::192.168.7.1:255.255.255.0  '

              vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
                  vvvvvvvvvvvvvvvvvvvvvvvvvvvv
rrrrrrrrrrrrr       vvvvvvvvvvvvvvvvvvvvvvvvvv
rrrrrrrrrrrrrrrr      vvvvvvvvvvvvvvvvvvvvvvvv
rrrrrrrrrrrrrrrrrr    vvvvvvvvvvvvvvvvvvvvvvvv
rrrrrrrrrrrrrrrrrr    vvvvvvvvvvvvvvvvvvvvvvvv
rrrrrrrrrrrrrrrrrr    vvvvvvvvvvvvvvvvvvvvvvvv
rrrrrrrrrrrrrrrr      vvvvvvvvvvvvvvvvvvvvvv
rrrrrrrrrrrrr       vvvvvvvvvvvvvvvvvvvvvv
rr                vvvvvvvvvvvvvvvvvvvvvv
rr            vvvvvvvvvvvvvvvvvvvvvvvv      rr
rrrr      vvvvvvvvvvvvvvvvvvvvvvvvvv      rrrr
rrrrrr      vvvvvvvvvvvvvvvvvvvvvv      rrrrrr
rrrrrrrr      vvvvvvvvvvvvvvvvvv      rrrrrrrr
rrrrrrrrrr      vvvvvvvvvvvvvv      rrrrrrrrrr
rrrrrrrrrrrr      vvvvvvvvvv      rrrrrrrrrrrr
rrrrrrrrrrrrrr      vvvvvv      rrrrrrrrrrrrrr
rrrrrrrrrrrrrrrr      vv      rrrrrrrrrrrrrrrr
rrrrrrrrrrrrrrrrrr          rrrrrrrrrrrrrrrrrr
rrrrrrrrrrrrrrrrrrrr      rrrrrrrrrrrrrrrrrrrr
rrrrrrrrrrrrrrrrrrrrrr  rrrrrrrrrrrrrrrrrrrrrr

       INSTRUCTION SETS WANT TO BE FREE
OF: fdt: Ignoring memory range 0x80000000 - 0x80200000
Linux version 4.15.0-yocto-standard (oe-user@oe-host) (gcc version 7.3.0 (GCC)) #1 SMP Sun Mar 11 22:34:40 PDT 2018
```



```
[  OK  ] Started Daily Cleanup of Temporary Directories.
[  OK  ] Reached target Timers.
[  OK  ] Listening on dropbear.socket.
[  OK  ] Reached target Sockets.
[  OK  ] Reached target Basic System.
[  OK  ] Reached target Containers.
[  OK  ] Started Kernel Logging Service.
[  OK  ] Started System Logging Service.
[  OK  ] Started D-Bus System Message Bus.
         Starting Login Service...
[  OK  ] Started Network Service.
[  OK  ] Started Login Service.
[  OK  ] Reached target Network.
         Starting Permit User Sessions...
         Starting Network Name Resolution...
[  OK  ] Started Permit User Sessions.
[  OK  ] Started Serial Getty on hvc0.
[  OK  ] Started Getty on tty1.
[  OK  ] Started Serial Getty on ttyS0.
[  OK  ] Reached target Login Prompts.
[  OK  ] Started Network Name Resolution.
[  OK  ] Reached target Multi-User System.
         Starting Update UTMP about System Runlevel Changes...
[  OK  ] Reached target Host and Network Name Lookups.
[  OK  ] Started Update UTMP about System Runlevel Changes.

BEC Linux
BEC Linnodistro.0 qemuriscv64 hvc0

qemuriscv64ux nodistro.0  login: qemuriscv64 ttyS0

qemuriscv64 login:
```

Work In Progress

- Graphical images
  - core-image-sato
  - core-image-x11
  - core-image-weston



```
In file included from ../../../gstreamer-1.12.4/gst/parse/../gst_private.h:42:0,
                 from ../../../gstreamer-1.12.4/gst/parse/parse.l:2:
../../gst/gstconfig.h:112:4: error: #error "Could not detect architecture; don't know whether it supports unaligned acces
s! Please file a bug."
 #  error "Could not detect architecture; don't know whether it supports unaligned access! Please file a bug."
    ^~~~~
```

Work In Progress

- _REENTRANT missing in risc-v gcc

```
% gcc -dumpspecs | grep REENTRANT
%{posix:-D_POSIX_SOURCE} %{pthread:-D_REENTRANT}

% ./build/tmp/work/riscv64-bec-linux/liburcu/0.10.1-r0/recipe-sysroot-native/usr/bin/riscv64-bec-linux/riscv64-bec-linux-gcc  --sysroot=/mnt/a/oe/build/tmp/work/riscv64-bec-linux/liburcu/0.10.1-r0/recipe-sysroot -dumpspecs| grep _REENTERANT
??
```

Work In Progress

- Profiling packages (LTTng)

```
#error "Cannot build: unrecognized architecture detected."
```

☺

- GCC Sanitizers

Work In Progress

- QEMU auto Testing
  - do_testimage_auto
- Enabling Testing

```
INHERIT += "testimage”
DISTRO_FEATURES_append = " ptest”
EXTRA_IMAGE_FEATURES_append = " ptest-pkgs”
TEST_SUITES = "auto”
TEST_IMAGE_qemuall = "1”
TEST_TARGET_qemuall = "qemu"
```

Work In Progress

- Core layer support for RISC-V
  - Targeted for Upstream Yocto 2.5 release (April 2018)

Work In Progress

- Cross Prelink
  - Used in Few places during build
    - Detect library deps using prelink-rtld in cross environment.
  - Patches submitted
    - https://lists.yoctoproject.org/pipermail/yocto/2018-March/040254.html
- libatomic-ops
  - Support in master upstream
  - Openembedded uses releases

Work In Progress

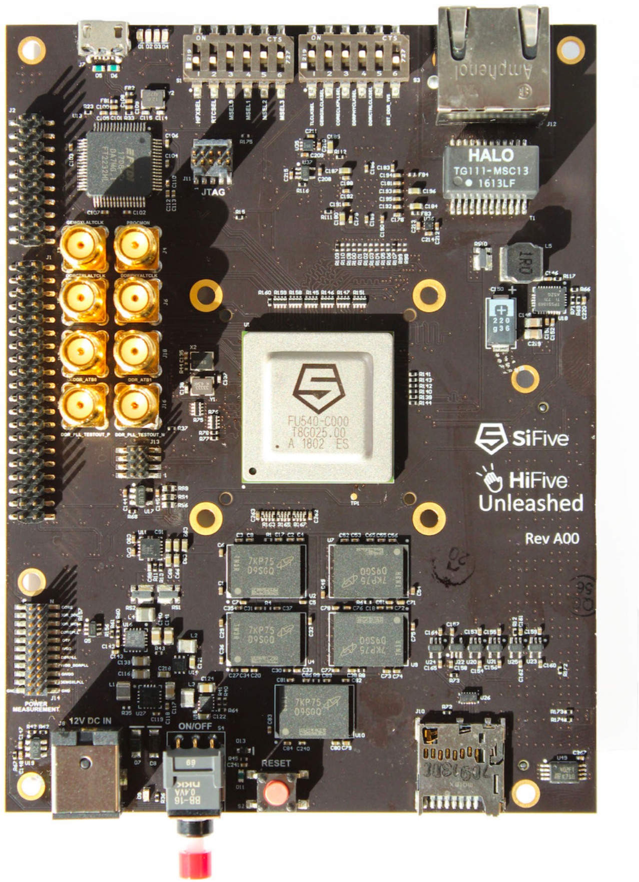
- Upstreaming patches
  - some packages e.g libffi, nspr, libgpg-error
  - Carrying local patches for RISC-V support in meta-riscv

Future

- Make RISC-V core supported architecture
    - Few steps until reaching that point
    - GDB not up-streamed yet
    - MUSL support
    - Golang support
    - A QEMU Release supporting RISC-V
        - upstream in QEMU master already

Future

- BSP layers for RISC-V based boards
  - SiFive Freedom Platform
  - SiFive Freedom U540 SoC
- RISC-V 32-bit
  - Needs 32bit glibc



RISC-V Resources

- https://riscv.org/
- https://github.com/riscv/
- https://github.com/sifive
- IRC #riscv on freenode
- Mailing lists
    - https://riscv.org/mailing-lists/
    - Software Development (sw-dev@groups.riscv.org)

Yocto Project Resources

- http://git.openembedded.org
- http://git.yoctoproject.org/
- https://www.yoctoproject.org/
- https://www.openembedded.org/wiki/Main_Page
- IRC #yocto and #oe on freenode
- Mailing lists
  - https://www.openembedded.org/wiki/Mailing_lists
  - https://lists.yoctoproject.org/listinfo

That’s All for Today

Thank you

Embedded Linux Conference
North America

OpenIoTSummit
North America

THE LINUX FOUNDATION