Building Zephyr with Yocto

Naveen Saini

Anuj Mittal

intel®

Speakers

Naveen Saini

- Operating Systems Developer, Intel
- Maintainer of Zephyr and ACRN layers.
- Contribute to Intel Yocto BSP layers i.e meta-intel, meta-dpdk, meta-qat
- naveen.kumar.Saini@intel.com

Anuj Mittal

- Operating Systems Developer, Intel
- Maintainer of Intel Yocto BSP layers ie meta-intel, meta-dpdk, meta-qat
- Contributor to Yocto Project development
- anuj.mittal@intel.com

Outline

- ✓ Introduction
- ✓ meta-zephyr layer
- ✓ Get Started with qemu
- ✓ Running Zephyr with ACRN Hypervisor and Yocto

Introduction

- ✓ Yocto Project delivers a set of tools that help create Linux based images for embedded systems. Terminologies:
  - ✓ Layers: Repositories that contain related sets of instructions that tell the build system what to do.
  - ✓ Recipes (files with .bb extension): Files that provide details about a particular piece of software, needed by the build system.
  - ✓ Bitbake: Task executor and scheduler used by Yocto.
- ✓ meta-zephyr provides an alternative way to build Zephyr application using Yocto build system.
- ✓ meta-zephyr consists of two layers:
  - ✓ meta-zephyr-core: distribution specific changes, zephyr kernel, application recipes and validation support.
  - ✓ meta-zephyr-bsp: hardware board configurations

```
meta-zephyr/
├── ci
├── meta-zephyr-bsp
├── meta-zephyr-core
└── README.txt
```

meta-zephyr layers in detail

https://git.yoctoproject.org/meta-zephyr

- ✓ Machine configurations for ARM and x86 architectures boards
- ✓ Interactive Kconfig interface support using menuconfig
- ✓ Zephyr SDK toolchain support
- ✓ QEMU support for development
- ✓ Flashing support using dfu, pyocd and bossac

```
meta-zephyr/meta-zephyr-bsp/conf/machine/
├── 96b-avenger96.conf
├── 96b-nitrogen.conf
├── arduino-nano-33-ble.conf
├── frdm-kw41z.conf
├── include
│   ├── nrf52.inc
│   ├── stm32mp1-cortex-m4.inc
│   ├── tune-arc.inc
│   ├── tune-corei7-common.inc
│   ├── tune-iamcu.inc
│   └── tune-nios2.inc
├── intel-x86-64.conf
├── mps2-an385.conf
├── mps2-an521.conf
├── mps3-an547.conf
├── nrf52840dk-nrf52840.conf
├── nrf52840-mdk-usb-dongle.conf
├── qemu-cortex-a53.conf
├── qemu-cortex-a9.conf
├── qemu-cortex-m0.conf
├── qemu-cortex-m3.conf
├── qemu-cortex-r5.conf
├── qemu-nios2.conf
├── qemu-x86.conf
├── stm32mp157c-dk2.conf
├── v2m-beetle.conf
├── v2m-musca-b1.conf
└── v2m-musca-s1.conf
```

Get started with qemu

- ✓ Layer setup
- ✓ Build configuration
- ✓ Zephyr kernel configuration using menuconfig (optional)
- ✓ Build
- ✓ Run

Layer setup

✓ Clone the repositories.

git clone https://git.yoctoproject.org/git/poky

git clone https://git.yoctoproject.org/meta-zephyr

git clone https://git.openembedded.org/meta-openembedded

✓ Set up the OpenEmbedded build environment.

source poky/oe-init-build-env

✓ Add bitbake layers.

bitbake-layers add-layer ../meta-openembedded/meta-oe

bitbake-layers add-layer ../meta-openembedded/meta-python

bitbake-layers add-layer ../meta-zephyr/meta-zephyr-core

bitbake-layer add-layer ../meta-zephyr/meta-zephyr-bsp

```
build/conf/
├── bblayers.conf
└── local.conf
```

Configuration

- ✓ Setup extra configuration in conf/local.conf

  MACHINE = "qemu-x86"

  DISTRO = "zephyr"

- ✓ To build with Zephyr SDK toolchain. By default, it is set to Yocto.

  ZEPHYR_TOOLCHAIN_VARIANT = "zephyr"

- ✓ To setup for a board, both 'MACHINE' and 'ZEPHYR_BOARD' need to be set. For example, to build application for 'x86 Elkahrt Lake' board, set

  MACHINE = "intel-x86-64"

  ZEPHYR_BOARD = "intel_ehl_crb"

```
meta-zephyr-bsp/conf/machine/
├── 96b-avenger96.conf
├── 96b-nitrogen.conf
├── arduino-nano-33-ble.conf
├── frdm-kw41z.conf
├── include
│   ├── nrf52.inc
│   ├── stm32mp1-cortex-m4.inc
│   ├── tune-arc.inc
│   ├── tune-corei7-common.inc
│   ├── tune-iamcu.inc
│   └── tune-nios2.inc
├── intel-x86-64.conf
├── mps2-an385.conf
├── mps2-an521.conf
├── mps3-an547.conf
├── nrf52840dk-nrf52840.conf
├── nrf52840-mdk-usb-dongle.conf
├── qemu-cortex-a53.conf
├── qemu-cortex-a9.conf
├── qemu-cortex-m0.conf
├── qemu-cortex-m3.conf
├── qemu-cortex-r5.conf
├── qemu-nios2.conf
├── qemu-x86.conf
├── stm32mp157c-dk2.conf
├── v2m-beetle.conf
├── v2m-musca-b1.conf
└── v2m-musca-s1.conf
```

Build & Run

- ✓ To build helloworld application

  $ bitbake zephyr-helloworld

- ✓ On successful build, binary can be found in deploy directory:

  build/tmp-newlib/deploy/images/qemu-x86/zephyr-helloworld.elf

- ✓ Run

  $ runqemu qemu-x86 nographic

```
SeaBIOS (version rel-1.16.3-0-ga6ed6b701f0a-prebuilt.qemu.org)
Booting from ROM..
*** Booting Zephyr OS build v3.6.0 ***
Hello World! qemu_x86
```

menuconfig

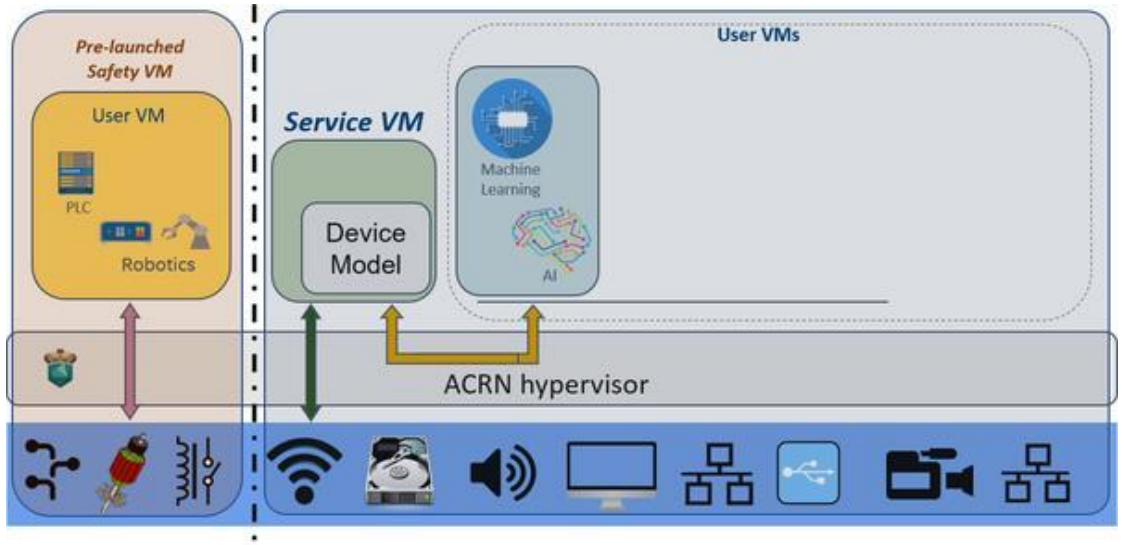
```
(Top)
                                                    Zephyr Kernel Configuration
    Devicetree Info  ----
    Modules  --->
    Board Selection (QEMU x86)  --->
    Board Options  --->
    SoC/CPU/Configuration Selection (Generic IA32 SoC)  --->
    Hardware Configuration  --->
    X86 Architecture Options  --->
    General Architecture Options  --->
[*] Assign appropriate permissions to kernel areas in SRAM
    DSP Options  ----
    Floating Point Options  --->
    Cache Options  --->
[ ] Custom arch_cpu_idle implementation
    General Kernel Options  --->
    Device Options  --->
    Virtual Memory Support  --->
    Device Drivers  --->
[ ] Require complete C library
[ ] Requires floating point support in printf
    C Library  --->
    C++ Language Support  --->
[ ] Cyclic redundancy check (CRC) Support
    Additional libraries  --->
    Subsystems and OS Services  --->
    Build and Link Features  --->
    Boot Options  --->

[Space/Enter] Toggle/enter  [ESC] Leave menu           [S] Save
[O] Load                    [?] Symbol info            [/] Jump to symbol
[F] Toggle show-help mode   [C] Toggle show-name mode  [A] Toggle show-all mode
[Q] Quit (prompts for save) [D] Save minimal config (advanced)
[devshell-0:devshell-1562032*                                          "andromeda02" 06:06 07-Mar-24
```

$ bitbake zephyr-helloworld –c menuconfig

Use case: Running Zephyr with ACRN and Yocto

Consider a scenario where Type-1 ACRN Hypervisor runs Zephyr as Prelaunch safety VM and Yocto based VM as service VM.



https://docs.zephyrproject.org/3.6.0/boards/x86/acrn/doc/index.html#building-and-running-zephyr-with-acrn

Running Zephyr with ACRN and Yocto

- ✓ It could be achieved using multiconfiguration, meta-acrn & meta-intel layers
- ✓ meta-acrn README: https://github.com/intel/meta-acrn/blob/master/docs/getting-started.md

- ✓ conf/multiconf/zephyr.conf.

  MACHINE = "intel-x86-64"

  ZEPHYR_BOARD = "acrn"

  DISTRO = "zephyr"

  TMPDIR="${TOPDIR}/master-zephyr-app"

- ✓ Set up the OpenEmbedded build environment. Append in conf/local.conf

  BBMULTICONFIG = "sos zephyr"

```
build/conf/
├── bblayers.conf
├── local.conf
└── multiconfig
    ├── sos.conf
    └── zephyr.conf
```

Running Zephyr with ACRN and Yocto

- ✓ Service VM configuration Conf/multiconf/sos.conf.

```
MACHINE = "intel-corei7-64“ # Machine configuration from meta-intel
TMPDIR = "${TOPDIR}/master-acrn-sos“

#ACRN HV specific Configuraiton
DISTRO = "acrn-demo-service-vm“ # Distro configuration from meta-acrn
ACRN_BOARD = "nuc11tnbi5“ # Tigerlake
ACRN_SCENARIO = "hybrid“ # Hybrid Scenario

# Required while packaging
CONTAINER_PACKAGE_DEPLOY_DIR = "${TOPDIR}/master-zephyr-app-newlib/deploy/images/intel-x86-64"

#Ensure zephy application built before wic image creation
do_image_wic[mcdepends] += "mc:sos:zephyr:zephyr-helloworld:do_deploy“

# Install in boot dir
IMAGE_EFI_BOOT_FILES:append = " ${CONTAINER_PACKAGE_DEPLOY_DIR}/zephyr-helloworld.elf;zephyr.elf ACPI_VM0.bin”
```

Running Zephyr with ACRN and Yocto

- ✓ To build ACRN HV image with Service VM and Zephyr as pre-launched VM:

  $ bitbake mc:sos:acrn-image-base

- ✓ Flash image on board and boot. Select 'ACRN (Yocto)' option from GRUB menu.

- ✓ At HV Console it should list both VMs

  $ vm_list


```
ACRN:\>vm_list

VM_ID VM_NAME                          VM_STATE
===== ================================ ========
  0   SAFETY_VM0                       Running
  1   ACRN_Service_VM                  Running
ACRN:\>
```

```
ACRN:\>
ACRN:\>vm_console 0

----- Entering VM 0 Shell -----
*** Booting Zephyr OS build v3.6.0 ***
Hello World! acrn


 ---Entering ACRN SHELL---
ACRN:\>
```

```
ACRN:\>
ACRN:\>vm_console 1

----- Entering VM 1 Shell -----

root@intel-corei7-64:~# uname -a
Linux intel-corei7-64 5.15.137-linux-intel-acrn-service-vm #1 SMP PREEMPT Fri Oct 27 17:42:22 UTC 2023 x86_64 GNU/Linux
root@intel-corei7-64:~#
root@intel-corei7-64:~#
root@intel-corei7-64:~#
root@intel-corei7-64:~#

 ---Entering ACRN SHELL---
ACRN:\>
```

To contribute..

- ✓ --subject-prefix “meta-zephyr][PATCH”
- ✓ Send patches to : yocto-patches@lists.yoctoproject.org
- ✓ Report bugs for meta-zephyr layer at https://bugzilla.yoctoproject.org/

Thank You !

Any questions ?