ZEPHYR + MICROPYTHON

QUICK GUIDE

PUBLIC

NXP

SECURE CONNECTIONS
FOR A SMARTER WORLD

Zephyr project

Scalable real-time operating system (RTOS) that supports:

- Multiple hardware architectures
- Optimized for resource constrained devices
- Built with safety and security in mind

https://www.zephyrproject.org

Objectives

- Initialize Zephyr repository and configure environment
- Checkout branch v.1.14 with command “git checkout v1.14-branch”
- Clone the MicroPython git repo
- Configure environment and Makefile
- Build and Flash Zephyr demo application
- Build and Flash MicroPython
- Build and Flash Zephyr project using Eclipse

STEP BY STEP USING TERMINAL

Install required applications (already in VM image)

- sudo apt-get install --no-install-recommends git cmake ninja-build gperf \

  ccache dfu-util device-tree-compiler wget \

  python3-pip python3-setuptools python3-tk python3-wheel xz-utils file \

  make gcc gcc-multilib

 # Update CMAKE. CMake 3.13.1 or higher is required.

- pip3 install --user cmake

 # Update Device Tree Compiler. dtc_1.4.6 or higher is required.

- curl -L http://mirrors.kernel.org/ubuntu/pool/main/d/device-tree-compiler/device-tree-compiler_1.4.7-1_amd64.deb > device-tree-compiler_1.4.7-1_amd64.deb
- sudo apt install ./device-tree-compiler_1.4.7-1_amd64.deb

 # Configure PATH environment variable

- export PATH=$PATH:$HOME/.local/bin

 # Install west tool

- pip3 install --user -U west

 # Initialize west at $HOME/zephyrproject location and update it

- west init zephyrproject
- cd zephyrproject
- west update

Setup zephyr environment

 # Most of these commands require you to be here

- cd $HOME/zephyrproject/zephyr

 # Switch branch

- git checkout v1.14-branch

 # Install additional applications required

- pip3 install --user -r scripts/requirements.txt

 # Set environment variables to use cross-compile toolchain

- export ZEPHYR_TOOLCHAIN_VARIANT=cross-compile
- export CROSS_COMPILE=$HOME/toolchain/riscv32-unknown-elf-gcc/bin/riscv32-unknown-elf-

 # Run script to setup Zephyr environment

- source zephyr-env.sh

Test Zephyr using riscv32-unknown-elf-gcc toolchain

 # Go to Zephyr's project folder

- cd $ZEPHYR_BASE

 # Build an example project, in this case the samples/basic/blinky demo application

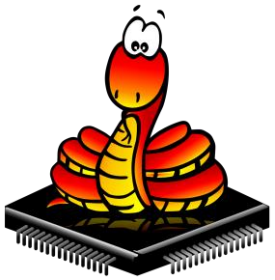
- cmake -B blinkyBuild -GNinja -DBOARD=rv32m1_vega_ri5cy -DCMAKE_REQUIRED_FLAGS=-Wl,-dT=/dev/null samples/basic/blinky

 # Download the application to the board (Make sure the J-link is connected)

- cd blinkyBuild
- west flash --openocd=$HOME/toolchain/openocd

 # Disconnect and connect the board, you should see the Green LED blinking

Micropython



 # Get Micropython (already in VM image)

- cd $HOME/zephyrproject
- git clone https://github.com/micropython/micropython.git
- cd $HOME/zephyrproject/micropython/ports/zephyr

 # Modify Makefile (line 107), insert: -DCMAKE_REQUIRED_FLAGS=-Wl,-dT=/dev/null **in mkdir command, after –Boutdir/$(BOARD), it should look like this:**

- vi Makefile

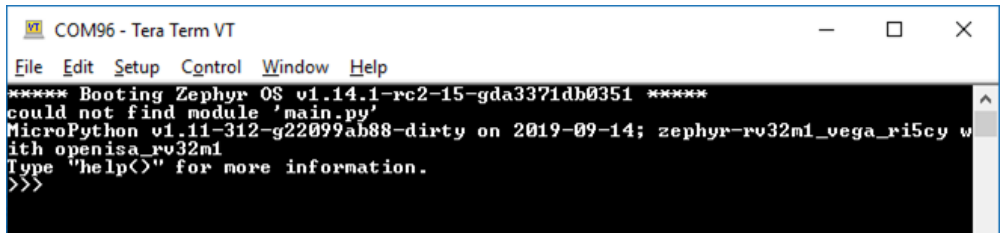
mkdir -p outdir/$(BOARD) && cmake -DBOARD=$(BOARD) -DCONF_FILE=$(CONF_FILE) -Boutdir/$(BOARD) **-DCMAKE_REQUIRED_FLAGS=-Wl,-dT=/dev/null** -H.

 # Build the application

- make BOARD=rv32m1_vega_ri5cy

 # Program application into flash

- cd $HOME/zephyrproject/micropython/ports/zephyr/outdir/rv32m1_vega_ri5cy/
- west flash --openocd=$HOME/toolchain/openocd
- **# Disconnect and connect the board, and press the reset button.**



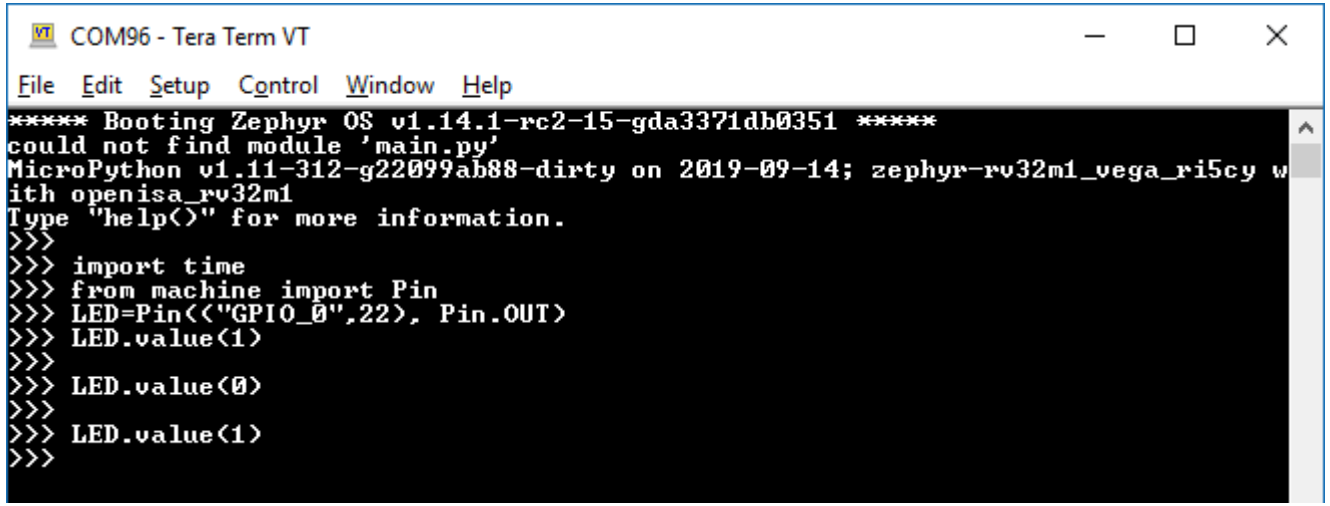
Test MicroPython

- Use following python scripts for test
  - Turn on the blue LED (PTA22)

```
import time
from machine import Pin
LED=Pin(("GPIO_0",22), Pin.OUT)
LED.value(1)
```

  - Flash the blue LED

```
import time
from machine import Pin
LED=Pin(("GPIO_0",22), Pin.OUT)
while True:
    LED.value(1)
    time.sleep(0.5)
    LED.value(0)
    time.sleep(0.5)
```



Zephyr in eclipse

- cd $ZEPHYR_BASE

 # Create a folder for project

- mkdir myProject && cd myProject

 # Compile project and generate eclipse files

- cmake -G"Eclipse CDT4 - Ninja" -DBOARD=rv32m1_vega_ri5cy -DCMAKE_REQUIRED_FLAGS=-Wl,-dT=/dev/null $ZEPHYR_BASE/samples/hello_world/

 # Open eclipse and import project from $ZEPHYR_BASE/myProject

- cd $HOME/eclipse
- ./eclipse

 # Build the project using the Hammer icon 

 # Configure the debugger using the "Debug configurations" section. See Configuring a zephyr eclipse project for VEGAboard **for instructions.**

- Summary:
  - Main tab: C/C++ Application: zephyr/zephyr.elf
  - Debugger tab:
    - OpenOCD: /home/user**/toolchain/openocd**
    - Config options: /home/user**/vega/rv32m1_sdk_riscv/boards/rv32m1_vega/rv32m1_ri5cy.cfg**
    - GDB Client: /home/user**/ /toolchain/riscv32-unknown-elf-gcc/bin/riscv32-unknown-elf-gdb**

Reference

- Zephyr
  - Getting Started Guide
  - Programming and Debugging

- MicroPython
  - MicroPython port to Zephyr RTOS Readme

- Open-ISA.org
  - Configuring a zephyr eclipse project for VEGAboard

SECURE CONNECTIONS
FOR A SMARTER WORLD