Preparation of MIMXRT1050-EVK Target and Host Development Environment

CSE 522, Spring 2022

Zephyr Source Code and SDK

Please follow the Getting Started Guild in https://docs.zephyrproject.org/2.6.0/getting_started/index.html to set up Zephyr development environment in a host machine (Linux, Windows, or macOS).

1. Use "west init" to clone Zephyr 2.6.0 for assignment development. The command is

   west init --mr v2.6.0 zephyrproject

2. Make sure you run "west update" after "west init". It initializes a local Git repository for the project in the workspace. The NXP's HAL that includes the peripheral drivers and device support directly from the MCUXpresso SDK is then cloned in local workspace. The HAL library is re-used when building Zephyr for NXP processors.

3. To build blinky sample for MIMXRT1050-EVK you can try the following steps in Linux:

   a. mkdir project_RTES && cd project_RTES

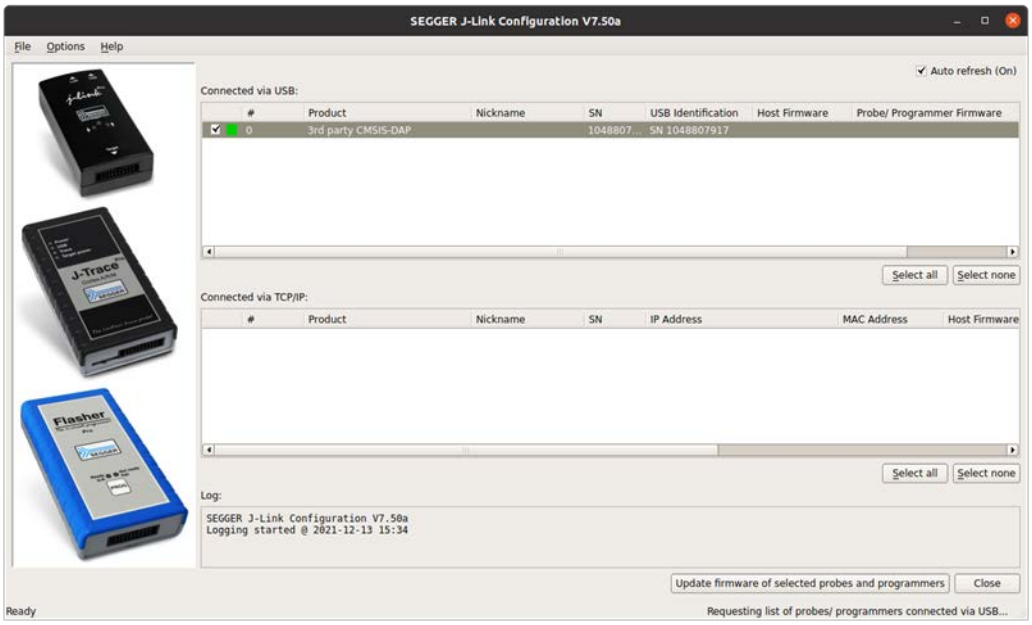
   b. cp -r ~/zephyrproject/zephyr/samples/basic/blinky .

   c. west build -b mimxrt1050_evk

MIMXRT1050-EVK Target

1. Getting Started with the MIMXRT1050-EVK Evaluation Board: https://www.nxp.com/document/guide/getting-started-with-the-mimxrt1050-evk-evaluation-board:GS-MIMXRT1050-EVK

2. Install J-Link Software and Documentation pack in your host computer from https://www.segger.com/downloads/jlink/. The software contains useful tools such as J-Link Commander, J-Link GDB Server, and J-Flash. You can run JlinkConfigExe to show the connected devices via USB:



3. Modifying Debug Firmware to Segger J-link OpenSDA (for SWD debug protocol and virtual COM port).

    a. Download JLink firmware for MIMXRT1050-EVK-Hyperflash from https://www.segger.com/downloads/jlink#JLinkOpenSDABoardSpecificFirmwares. The name of the firmware file is 49_OpenSDA_MIMXRT1050-EVK-Hyperflash.bin

    b. Steps to load Jlink OpenSDA firmware on the MIMXRT1050-EVKB board:

        i. Unplug the EVK board
        ii. Hold the reset button SW4. Continue holding the button while connecting the USB cable between host computer and USB connector J28. This forces OpenSDA to boot in bootloader mode. It will enumerate in the host computer as a mass-storage drive, with volume label MAINTENANCE.
        iii. Drag-and-Drop the file 49_OpenSDA_MIMXRT1050-EVK-Hyperflash.bin to the OpenSDA MAINTENANCE drive. This will update the OpenSDA firmware with the Jlink file.
        iv. After the file has copied, unplug the EVK board, and plug it back in.
        v. OpenSDA should now enumerate as a Jlink debug probe. It will no longer enumerate as a mass-storage drive, and instead will enumerate as a COM port labeled “Jlink DCD UART Port”.

    Now the Jlink debug probe can be used with Zephyr and west commands. If you run JlinkConfigExe again, you will see a different product connected via USB. You can also test JLink connection by running JLinkExe:

```
yhlee@yhlee-UP-APL01:~/z2.6/project/led_pwm$ JLinkExe
SEGGER J-Link Commander V7.50a (Compiled Jul  8 2021 18:21:10)
DLL version V7.50a, compiled Jul  8 2021 18:20:53

Connecting to J-Link via USB...O.K.
Firmware: J-Link OpenSDA 2 compiled May 27 2019 11:01:03
Hardware version: V1.00
S/N: 621000000
VTref=3.300V


Type "connect" to establish a target connection, '?' for help
J-Link>connect
Please specify device / core. <Default>: MIMXRT1052XXXXA
Type '?' for selection dialog
Device>
Please specify target interface:
  J) JTAG (Default)
  S) SWD
  T) cJTAG
TIF>s
Specify target interface speed [kHz]. <Default>: 4000 kHz
Speed>
Device "MIMXRT1052XXXXA" selected.


Connecting to target via SWD
Found SW-DP with ID 0x0BD11477
DPv0 detected
Scanning AP map to find all available APs
AP[1]: Stopped AP scan as end of AP map has been reached
AP[0]: AHB-AP (IDR: 0x04770041)
Iterating through AP map to find AHB-AP to use
AP[0]: Core found
AP[0]: AHB-AP ROM base: 0xE00FD000
CPUID register: 0x411FC271. Implementer code: 0x41 (ARM)
Found Cortex-M7 r1p1, Little endian.
FPUnit: 8 code (BP) slots and 0 literal slots
CoreSight components:
ROMTbl[0] @ E00FD000
ROMTbl[0][0]: E00FE000, CID: B105100D, PID: 000BB4C8 ROM Table
ROMTbl[1] @ E00FE000
ROMTbl[1][0]: E00FF000, CID: B105100D, PID: 000BB4C7 ROM Table
ROMTbl[2] @ E00FF000
```

ROMTbl[2][0]: E000E000, CID: B105E00D, PID: 000BB00C SCS-M7
ROMTbl[2][1]: E0001000, CID: B105E00D, PID: 000BB002 DWT
ROMTbl[2][2]: E0002000, CID: B105E00D, PID: 000BB00E FPB-M7
ROMTbl[2][3]: E0000000, CID: B105E00D, PID: 000BB001 ITM
ROMTbl[1][1]: E0041000, CID: B105900D, PID: 001BB975 ETM-M7
ROMTbl[1][2]: E0042000, CID: B105900D, PID: 004BB906 CTI
ROMTbl[0][1]: E0040000, CID: B105900D, PID: 000BB9A9 TPIU-M7
ROMTbl[0][2]: E0043000, CID: B105F00D, PID: 001BB101 TSG
Cache: Separate I- and D-cache.
I-Cache L1: 32 KiB, 512 Sets, 32 Bytes/Line, 2-Way
D-Cache L1: 32 KiB, 256 Sets, 32 Bytes/Line, 4-Way
Cortex-M7 identified.
J-Link>F
Firmware: J-Link OpenSDA 2 compiled May 27 2019 11:01:03
Hardware: V1.00

4. Recommended debugger: Segger's Ozone from https://www.segger.com/products/development-tools/ozone-j-link-debugger/

   Ozone is RTOS aware and it is able to deal with Zephyr threads. To do Zephyr debug with Jlink and Ozone:

   - For thread awareness support, Zephyr projects should be compiled with

     CONFIG_DEBUG_THREAD_INFO=y
     CONFIG_THREAD_NAME=y

   - In Ozone new project wizard, select
     - Target device: MIMXRT1052xxxxA
     - Register set: Cortex_M7 (with FPU)
     - Target interface: SWD
     - the elf file of the Zephyr build for debug.
     - In the optional setting, initial PC and initial stack pointer: Do not set
   - In the bottom left console, type Project.SetOSPlugin("ZephyrPlugin"). That will load the Zephyr RTOS awareness plugin.
   - You should now be able to access the Zephyr debug window from "View" menu.

5. Install Segger's SystemView https://www.segger.com/products/development-tools/systemview/. SystemView can visualize the runtime events at the target and will be needed in assignment 1.