FIRMWARE ESSENTIALS E4357 HOMEWORK 4

Dwayne Dilbeck

HOMEWORK REQUIREMENTS FROM SUPPLIED LINK

MBED DEBUGGING

1. Get debugging working on MBed using either Keil or pyOCD
2. Write a page report

REQUIREMENTS MENTIONED IN CLASS

1. Upgrade MBED firmware
2. Get debugger online
3. Single step through HelloWorld

STATUS

TIME SHEET

Task	**Time**
Update Firmware	0 minutes
Install pyOCD	25 minuites
Verify pyOCD and openOCD	8 hours
Total	**8 hours 25 minutes**

DEBUGGERS

PYOCD

PYOCD INSTALL

1) Yum install libusb-devel
2) Yum install zlib.i686
3) Tar -jxvf gcc-arm-none-eabi-misc.tar.bz2

VERIFY PYOCD DEBUG FLOW :TIME 10MINUTES

1) pyOCD_linux
    a. Mbed board found

```
[root@localhost test]# ./pyocd_linux
Welcome to the PyOCD GDB Server Beta Version
INFO:root:new board id detected: 1010bebf73074a115edd606be71ae59d0820
id => usbinfo | boardname
0 =>   (0xd28, 0x204) [lpc1768]
INFO:root:DAP SWD MODE initialised
INFO:root:IDCODE: 0x2BA01477
INFO:root:6 hardware breakpoints, 4 literal comparators
INFO:root:CPU core is Cortex-M3
INFO:root:GDB server started at port:3333
```

2) Arm-none-eabi-gdb lab1.elf
    a. Target remote local host:3333

```
INFO:root:GDB server started at port:3333
INFO:root:One client connected!
```

    b. Load

```
Welcome to
INFO:root:
id => usbi
0 =>   (0x
INFO:root:
INFO:root:
INFO:root:
(gdb) load
Loading section .text, size 0x6d38 lma 0x0
Loading section .ARM.exidx, size 0x8 lma 0x6d38
Loading section .data, size 0xb4 lma 0x6d40
Start address 0x62c, load size 28148
Transfer rate: 3 KB/sec, 1655 bytes/write.
(gdb)
INFO:root:CPU core is Cortex-M3
INFO:root:GDB server started at port:3333
INFO:root:One client connected!
[====================] 100%
```

    c. B main

```
(gdb) b main
Breakpoint 1 at 0x1cc: file main.cpp, line 8.
```

    d. C
        i. Result: incorrect program execution. LEDs do not blink.

Initially this was believed to be a problem with the downloaded tool. Later trials found several issues.

VERIFY PYOCD #2

1) Load MBED with Blinky.bin
2) pyOCD_linux
    a. Mbed board found

```
[root@localhost test]# ./pyocd_linux
Welcome to the PyOCD GDB Server Beta Version
INFO:root:new board id detected: 1010bebf73074a115edd606be71ae59d0820
id => usbinfo | boardname
0 =>   (0xd28, 0x204) [lpc1768]
INFO:root:DAP SWD MODE initialised
INFO:root:IDCODE: 0x2BA01477
INFO:root:6 hardware breakpoints, 4 literal comparators
INFO:root:CPU core is Cortex-M3
INFO:root:GDB server started at port:3333
```

    b.
3) Arm-none-eabi-gdb lab1.elf

a. Target remote local host:3333

```
INFO:root:GDB server started at port:3333
INFO:root:One client connected!
```

b. Mon reset halt
c. Load
d. Mon reset
    i. Correctly functioning application. LEDs flash in the Lab1 sequence and not in the blinky sequence.
e. Mon reset halt
f. B 10
g. C
    i. Result SWD Fault

```
[====================] 100%
Exception in thread Thread-1:
Traceback (most recent call last):
  File "/home/build/tonwan01/pyOCD/test/build/gdb_server/out00-PYZ.pyz/threading
", line 532, in __bootstrap_inner
  File "/home/build/tonwan01/pyOCD/test/build/gdb_server/out00-PYZ.pyz/pyOCD.gdb
server.gdbserver", line 167, in run
  File "/home/build/tonwan01/pyOCD/test/build/gdb_server/out00-PYZ.pyz/pyOCD.gdb
server.gdbserver", line 233, in handleMsg
  File "/home/build/tonwan01/pyOCD/test/build/gdb_server/out00-PYZ.pyz/pyOCD.gdb
server.gdbserver", line 289, in resume
  File "/home/build/tonwan01/pyOCD/test/build/gdb_server/out00-PYZ.pyz/pyOCD.tar
get.cortex_m", line 605, in resume
  File "/home/build/tonwan01/pyOCD/test/build/gdb_server/out00-PYZ.pyz/pyOCD.tar
get.cortex_m", line 362, in writeMemory
  File "/home/build/tonwan01/pyOCD/test/build/gdb_server/out00-PYZ.pyz/pyOCD.tra
nsport.cmsis_dap", line 133, in writeMem
  File "/home/build/tonwan01/pyOCD/test/build/gdb_server/out00-PYZ.pyz/pyOCD.tra
nsport.cmsis_dap_core", line 215, in dapTransfer
ValueError: SWD Fault

ERROR:root:exception during uninit
```

It should be noted that if the 'mon reset halt' command is issued and the pyOCD_linux is restarted, Debugging will function. But the work around is clumsy.

VERIFY PYOCD #3

1) Repeat pyOCD #2 steps
2) Reconnect pyOCD_linux and gdb
    a. Target remote localhost:3333
    b. B main
    c. B 10
    d. C
        i. Result: LEDs blink in the correct order, and the specified breakpoints are triggered.

INSTALL PYOCD #2

Build attempted for pyOCD instead of using premade pyOCD_linux from GCC Arm embedded.

1. Result: pyOCD was less stable than pre-packaged.

VERIFY OPENOCD.

While investigating ways to get pyOCD to be stable after a 'mon reset halt' I came across postings comparing openOCD flash time to pyOCD.

3) Openocd –f /usr/share/openocd/scripts/board/mbed-lpc1768.cfg

```
Open On-Chip Debugger 0.8.0 (2014-04-29-12:22)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.sourceforge.net/doc/doxygen/bugs.html
Error: The specified debug interface was not found (cmsis-dap)
The following debug interfaces are available:
1: parport
2: dummy
3: ftdi
4: usb_blaster
5: jtag_vpi
6: amt_jtagaccel
7: gw16012
8: usbprog
9: jlink
10: vsllink
11: rlink
12: ulink
13: arm-jtag-ew
14: buspirate
15: remote_bitbang
16: hla
17: osbdm
18: opendous
19: sysfsgpio
20: aice
```

The default package for openOCD does not have the cmsis-dap support enabled.

COMPILE OPENOCD 0.8.0 AND VERIFY

The openOCD 0.8.0 installed by 'yum install openOCD' does not have the CMSIS-DAP interface enabled. This requires a recompile from source code

1) Install HIDAPI library
2) In openocd ./configure –enable-cmsis-dap
3) Make || make install
   i. Openocd –f /usr/share/openocd/scripts/board/mbed-lpc1768.cfg

```
[root@localhost pyOCD-master]# /usr/openocd/bin/openocd -f /usr/share/openocd/sc
ripts/board/mbed-lpc1768.cfg
Open On-Chip Debugger 0.8.0 (2015-04-09-04:04)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.sourceforge.net/doc/doxygen/bugs.html
Info : only one transport option; autoselect 'cmsis-dap'
Info : CMSIS-DAP: SWD  Supported
Info : CMSIS-DAP: Interface Initialised (SWD)
adapter speed: 10 kHz
adapter_nsrst_delay: 200
cortex_m reset_config sysresetreq
Info : CMSIS-DAP: FW Version = 1.0
Info : SWCLK/TCK = 0 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 1
Info : DAP_SWJ Sequence (reset: 50+ '1' followed by 0)
Info : CMSIS-DAP: Interface ready
Info : clock speed 10 kHz
Info : IDCODE 0x2ba01477
Info : lpc1768.cpu: hardware has 6 breakpoints, 4 watchpoints
```

   ii. Test same gdb steps as pyOCD#2 without loading new elf.
      1. Result: All break points are triggered. No SWD Faults occur.
   iii. Test elf load.
      1. Result: ELF load hangs

ISSUES

DEMO GDB INSTRUCIONS RESULTED IN AN INCORRECTPROGRAM EXECUTION

If was seen that any time a program was loaded without issuing 'mon reset halt' before load and 'mon reset' after load, the program execution would be incorrect. But this also led to another issue

PYOCD CRASH

When performing the actions needed to get a proper program execution after load of an image and settings break point, a crash of pyOCD would occur. Simply restarting pyOCD would allow debug to function.

OPENOCD FAILED TO CONNECT TO CMSIS-DAP

The distributed package for openOCD does not have CMSIS-DAP enabled. A recompile of openOCD is required.

OPENOCD FAILS TO LOADIMAGES TO MBED

Using the load function in openOCD, would cause a hang to occur. But the “Mon resethalt” flow that causes a failure in pyOCD works correctly in openOCD. Loading the mbed program via the MBED USB and debugging using mon reset commands gave full access to all the openOCD commands. openOCD has more commands available than pyOCD.