FPGA 开发板 Low Level Debug 需要做的测试项目

很多客户都会用FPGA 开发板来做原型软件开发，一般FPGA bit/mcs 做好后，大家都迫不及待的就开始用Nuclei Studio IDE 开始下载程序来调试或者运行。但往往刚做的bit/mcs很容易不稳定，或者对应新的FPGA 板openocd.cfg 配置文件没有对应修改，而导致不能正常使用。 这时直接用IDE 来调试软件，就各种奇怪现象且不知道哪里出错导致。 本文介绍在初期做用FPGA 时，如何确认FPGA 板可以正常使用的步骤。

1. 在Windows 或者Linux 这边用命令行的方式执行OpenOCD （网站上有下载，然后下载了NucleiStudio 后，安装包里也有openocd 的可执行文件），执行OpenOCD 时需要准备openocd.cfg 文件，一般只需要参考我们demo_soc 的cfg 文件对应修改（如附件openocd_demosoc.cfg），然后执行openocd -f ****.cfg , 如果执行结果如下图，则表示FPGA 板上JTAG 接口和CPU 都是正常工作：

```
Open On-Chip Debugger 0.11.0+dev-01870-g201e7f417 (2021-08-21-10:12)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
DEPRECATED! use 'adapter speed' not 'adapter_khz'
DEPRECATED! use 'adapter driver' not 'interface'
Info : libusb_open() failed with LIBUSB_ERROR_NOT_FOUND
Info : no device found, trying D2xx driver
Info : D2xx device count: 2
Info : Connecting to "(null)" using D2xx mode...
Info : clock speed 1000 kHz
Info : JTAG tap: riscv.cpu tap/device found: 0x13000a6d (mfg: 0x536 (Nuclei System Technology Co Ltd), part: 0x3000, ver: 0x1)
Info : datacount=4 progbufsize=2
Info : Examined RISC-V core; found 1 harts
Info :  hart 0: XLEN=32, misa=0x4010912f
Info : starting gdb server for riscv.cpu on 3333
Info : Listening on port 3333 for gdb connections
Info : Valid NUSPI on device Nuclei SoC SPI Flash at address 0x20000000 with spictrl regbase at 0x10014000
Info : Nuclei SPI controller version 0x00000000
Info : Found flash device 'gd gd25q32c' (ID 0x001640c8)
cleared protection for sectors 0 through 63 on flash bank 0

Info : Listening on port 6666 for tcl connections
Info : Listening on port 4444 for telnet connections
```

另外，一些客户的FPGA 上可能没有SPI FLASH， 或者不需要openocd 来烧录Flash， 那么这个时候可以去掉cfg 里关于flash 烧写的配置：

```
$_TARGETNAME configure -work-area-phys 0x80000000 -work-area-size 10000 -work-area-backup 1

#set _FLASHNAME $_CHIPNAME.flash
#flash bank $_FLASHNAME fespi 0x20000000 0 0 0 $_TARGETNAME
# Set the ILM space also as flash, to make sure it can be add breakpoint with hardware trigger
#flash bank onboard_ilm fespi 0x80000000 0 0 0 $_TARGETNAME

# Expose Nuclei self-defined CSRS
# See https://github.com/riscv/riscv-gnu-toolchain/issues/319#issuecomment-358397306
# Then user can view the csr register value in gdb using: info reg csr775 for CSR MTVT(0x307)
riscv expose_csrs 416-496,770-800,835-850,1227-1231,1483-1486,1984-2040,2064-2070,2370-2380,2490-2500,4032-4040

init

if {[ info exists pulse_srst]} {
  ftdi_set_signal nSRST 0
  ftdi_set_signal nSRST z
}
halt
# We must turn on this because otherwise the IDE version debug cannot download the program into flash
#flash protect 0 0 last off
```

2. 如第一步完成，第二步则是用命令行开GDB，让GDB 和OpenOCD 连接， 一般在执行target remote 前，要先让GDB 知道target 端是RV32 还是RV64，用如下命令：set arch riscv:rv32 或者set arch riscv:rv64， 如果target remote 这步能正常执行，则进一步说明FPGA 上的JTAG 和FPGA 上的Nuclei Core 都是正常的（这里也需要注意，有可能因为FPGA 上的CPU 速度比较慢，GDB 首次连接时报超时，所以在target remote 前，可以用“set remotetimeout 240 ”来让GDB 多等一会）。

```
Open On-Chip Debugger 0.11.0+dev-01870-g201e7f417 (2021-08-21-10:12)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
DEPRECATED! use 'adapter speed' not 'adapter_khz'
DEPRECATED! use 'adapter driver' not 'interface'
Info : libusb_open() failed with LIBUSB_ERROR_NOT_FOUND
Info : no device found, trying D2xx driver
Info : D2xx device count: 2
Info : Connecting to "(null)" using D2xx mode...
Info : clock speed 1000 kHz
Info : JTAG tap: riscv.cpu tap/device found: 0x13000a6d (mfg: 0x536 (N
uclei System Technology Co Ltd), part: 0x3000, ver: 0x1)
Info : datacount=4 progbufsize=2
Info : Examined RISC-V core; found 1 harts
Info :  hart 0: XLEN=32, misa=0x4010912f
Info : starting gdb server for riscv.cpu on 3333
Info : Listening on port 3333 for gdb connections
Info : Valid NUSPI on device Nuclei SoC SPI Flash at address 0x2000000
0 with spictrl regbase at 0x10014000
Info : Nuclei SPI controller version 0x00000000
Info : Found flash device 'gd gd25q32c' (ID 0x001640c8)
cleared protection for sectors 0 through 63 on flash bank 0

Info : Listening on port 6666 for tcl connections
Info : Listening on port 4444 for telnet connections
Info : accepting 'gdb' connection on tcp/3333
```

```
   text    data     bss     dec     hex filename
  11162     116    4348   15626    3d0a helloworld.elf
"Run gdb to connect openocd server and debug"
riscv-nuclei-elf-gdb helloworld.elf  -ex "set remotetimeout 240" -ex "
target extended-remote localhost:3333"
C:\software\IDE\NucleiStudio\toolchain\gcc\bin\riscv-nuclei-elf-gdb.ex
e: warning: Couldn't determine a path for the index cache directory.
GNU gdb (GDB) 10.1
Copyright (C) 2020 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gp
l.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "--host=i686-w64-mingw32 --target=riscv-nuc
lei-elf".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from helloworld.elf...
Remote debugging using localhost:3333
0x20002c42 in ?? ()
(gdb)
```

3. 在第二步能正常执行的前提下，这时要用GDB 的命令来检查环境：

A， GDB 可以读写 CPU 的GPR，比如读写pc ；

B， GDB 可以读写CPU 的CSR， 比如打印misa，mstatus 等CSRs；

C， GDB 可以正常读写memory， 这一步很重要， 因为FPGA 上的SRAM 可能因为timing 问题，不能正常工作。 这一步，可以用GDB 的restore/dump 来做，可以做一个测试bin文件，先restore 到FPGA 上的SRAM 上，再从SRAM 上dump 出来，做比对， 一定要完全一样，才能说明FPGA 上的SRAM 是正常的。

D， 用GDB 做8bit/16bit 读写测试，因为RISC-V 的指令有16 bit 指令，RISC-V 的数据类似支持8 bit/16 bit 的读写。

E， 用GDB 命令读写SOC 外设的寄存器。

```
(gdb) restore ../ff-4K.bin binary 0x80000000
Restoring binary file ../ff-4K.bin into memory (0x80000000 to 0x80001000)
(gdb) dump binary memory test.bin 0x80000000 0x80000FFF
(gdb) x/50xw 0x80000000
0x80000000:     0xffffffff      0xffffffff      0xffffffff      0xffffffff
0x80000010:     0xffffffff      0xffffffff      0xffffffff      0xffffffff
0x80000020:     0xffffffff      0xffffffff      0xffffffff      0xffffffff
0x80000030:     0xffffffff      0xffffffff      0xffffffff      0xffffffff
0x80000040:     0xffffffff      0xffffffff      0xffffffff      0xffffffff
0x80000050:     0xffffffff      0xffffffff      0xffffffff      0xffffffff
0x80000060:     0xffffffff      0xffffffff      0xffffffff      0xffffffff
0x80000070:     0xffffffff      0xffffffff      0xffffffff      0xffffffff
0x80000080:     0xffffffff      0xffffffff      0xffffffff      0xffffffff
0x80000090:     0xffffffff      0xffffffff      0xffffffff      0xffffffff
0x800000a0:     0xffffffff      0xffffffff      0xffffffff      0xffffffff
0x800000b0:     0xffffffff      0xffffffff      0xffffffff      0xffffffff
0x800000c0:     0xffffffff      0xffffffff
(gdb) set *(short *)0x80000000=0x1122
(gdb) x/8b 0x80000000
0x80000000:     0x22    0x11    0xff    0xff    0xff    0xff    0xff    0xff
(gdb) set *(char *)0x80000000=0x55
(gdb) x/8b 0x80000000
0x80000000:     0x55    0x11    0xff    0xff    0xff    0xff    0xff    0xff
(gdb) p/x $a0
$4 = 0x0
```

如果step1， step2， step 3 的A，B，C，D，E 都测试过了，那么low level 的debug 项目都完成，也说明FPGA bit/msc 是稳定的， 这时就可以用IDE 来测试您的软件工程了。