Wind River On-chip Debugging

快速入门指南

通常情况下，用户拿到的 Wind River OCD 产品包含：配套软件 Workbench OCD（含 License）；硬件仿真器（Wind River Probe；Wind River ICE；Wind River ICE 2），仿真器会随硬件带有至少一种 JTAG 连接头。

其中 Probe 和 ICE 2 使用相同的 JTAG 连接头 52-pin 转 n-pin，典型情况下，n 对于 PowerPC 来说是 16，对于 MIPS 来说是 14，对于 ARM 来说是 20，对于 ColdFire 是 26，对于 MPC8xx 是 BDM 10-pin（请注意，ICE 2 目前不支持 BDM 和 ColdFire，Xscale）。

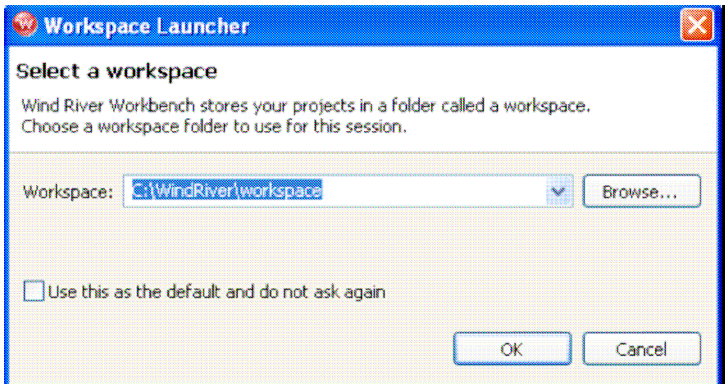
而 ICE 使用一种叫做 Personality Module 的连接模块，这种模块会依照不同处理器进行区分，相对复杂些，共分为：PowerPC 16-pin（MPC82xx，MPC83xx，MPC85xx，MPC86xx，MPC52xx，MPC74xx）；BDM 10-pin；PowerPC44x 16-pin；PowerPC40x 16-pin；ColdFire 26-pin；ARM 20-pin；Xscale 20-pin；MIPS 14-pin。除此以外还用一种用于 Xilinx V5 PPC440（参考板 ML507）的 14-pin 的 2mm 小连接头，Wind River 提供用于 ICE 和 ICE 2 的 16-pin 转 14-pin 的转接头。

注：以上均为典型应用中的连接头类型，Wind River 还提供不同管脚的连接头，详细请参考 PUF 文档。

一. 启动 Workbench

首先需要安装 Workbench OCD 软件，在软件 DVD 根目录下执行 setup.exe（Windows），setup.linux（Linux），给定安装目录，Lincense 文件。安装完毕后，启动 Workbench，Windows 下选择开始>程序>Wind River>Wind River Workbench；Linux 下执./startWorkbench.sh。

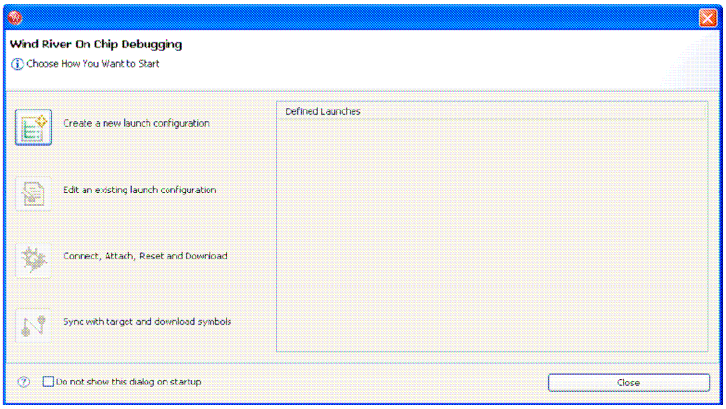
会出现 Workspace Launcher，



输入将来要做为工作目录的文件夹路径。

Workbench 完全启动后，会出现 Welcome 画面

点击 Workbench 图标，出现 Quick Target Launch dialog 画面。



在这个画面中，可以建立新的连接，或管理已有的连接（仿真器和目标板）。如果以前建立过多种连接，会在画面右侧的 Defined Launches 里列出来。这个连接管理工具也可以通过点击 Workbench 工具栏上的 OCD Quick Launch 按钮重新进入。

二. 仿真器的连接

1. 连接 Wind River ICE

1.1 配置 ICE 的 IP 地址

a. 将 Wind River ICE SX 连接到网络。ICE 在背面有两个以太网口：一个是 10BaseT 端口，另一个是 10/100BaseT 端口自适应。将以太网线一端连接 ICE，另一端连接网络。

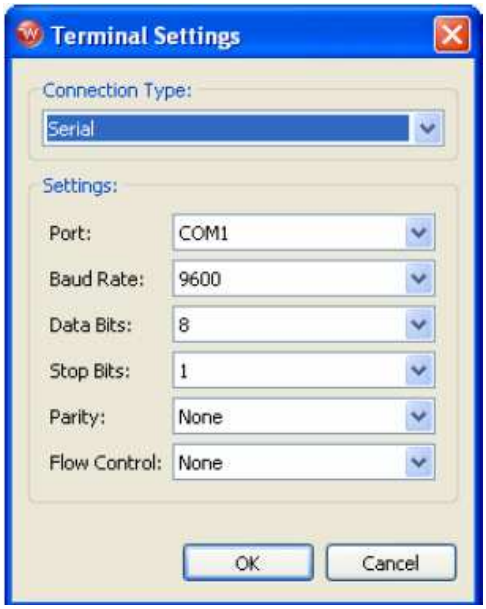
b. 将 ICE SX 的串口连接到主机串口。将随 ICE 自带的串口线一端 RJ-45 适配头连接到 ICE 背部的 RS-232 端口，另一端 DB-9 连接头连接到主机的串行 COM 口。

注意：RS-232 在 ICE 后面板上，不要将 RS232 端口和 ICE 前面板上的 TGTCONS 混淆。

c. 在 Workbench 工具栏，选择 Window>Open Perspective>On Chip Debug。

d. 选择 Window>Show View>Terminal。

e. 在 Terminal 串口，选择 Setting 图标，进行串口参数设置。



f. Connection Type 选择 Serial；Port 选择 COM1，Baud Rate 选择 9600，Data Bits 为 8，Stop Bits 为 1，Parity 为 None，Flow Control 为 None，点击 OK。

g. 此时 Terminal 进入>NET>提示符，输入 ethsetup，进入以太网设置菜单：

```
>NET>ethsetup
                                        Ethernet Setup Mode

Select from the operations below
     1. Display Basic IP parameters   2. Modify Basic IP parameters
     3. Display Routing parameters    4. Modify Routing parameters
     5. Display Server parameters     6. Modify Server parameters
     7. View ethernet address         8. Save parameters
     9. Exit setup mode              10. Port A/B select
    11. Advanced Options

Make a selection:
```

如果网络中有 DHCP 服务器，并用此方式，选择 1，Display Basic IP parameters，可以看到 ICE 通过 DHCP 获得的 IP 地址。

如果给 ICE 设置静态 IP，选择 2，Modify Basic IP parameters，会进入第二个菜单；选择 1，Enter the IP manually，设置地址，掩码和网关。

h. 配置完 IP 参数后，会回到 Ethesetup 菜单，保存选择 8，按回车。

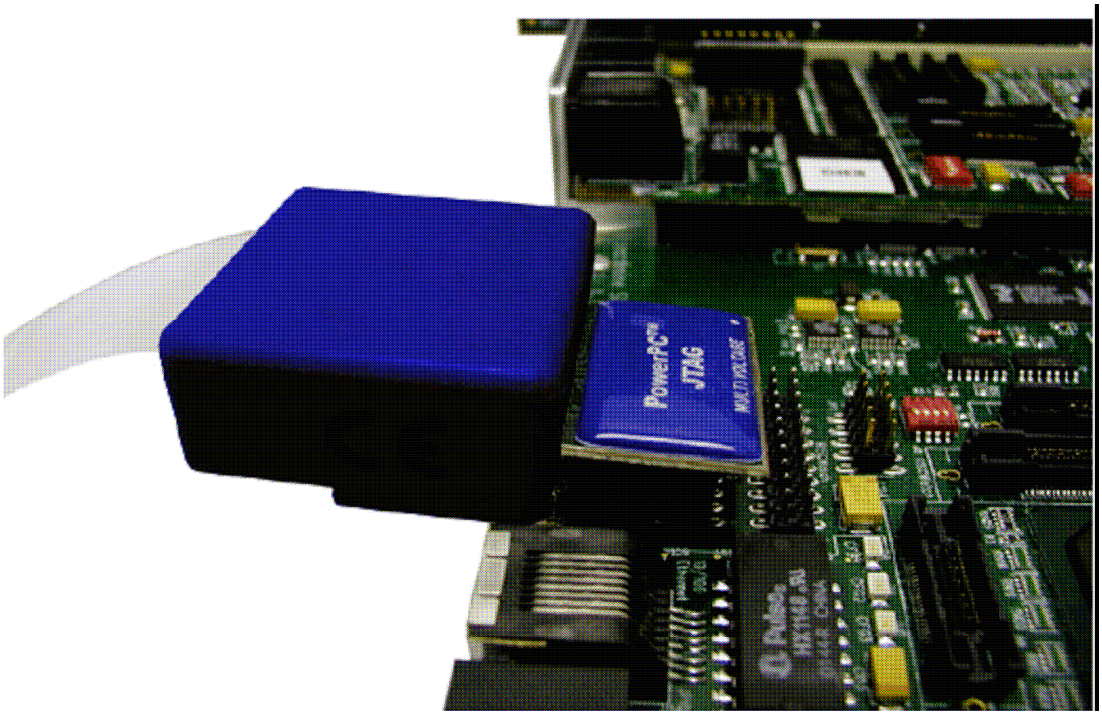
i. 选择 9 退出 Ethsetup 菜单。

j. 要使设置生效，需要重启 Wind River ICE SX，把开关置到 RESET 并回到 ON。（开关有三档：RESET, ON, OFF）

k. 此时配置网络完毕，可断掉串口连接。

1.2 连接 ICE

a. 将 ICE 连接到目标板。ICE 通过一条 51-pin 的电缆和 personality module 连接到目标板上的 JTAG 或 BDM 接口。Personality module 的管脚 1 标记为白点。确认其和板上 JTAG 管脚 1 对应。如果空间不足，可以用延长线连接到目标板。



b. 将 5-pin DIN 的电源线圆头连接到 ICE 背部的电源接口，电源要求 110-240 VAC 50/60 Hz。

c. 上电，并将开关置于 ON。

d. 建立目标板连接，请参看后面章节。

2. 连接 Wind River ICE 2

a. 将以太网线连接到 Wind River ICE 2 背部的千兆网口。

b. 另一端连接到主机网络设备。Wind River ICE 2 缺省采用 DHCP，你的 DHCP 服务器会自动分配 IP 地址。IP 地址会显示在 Wind River ICE 2 前面的 LCD 屏上。

c. 连接到主机后，通过随 ICE 2 自带的适配头连接到目标板上的 JTAG 或 BDM 接口。注意 ICE 2 和目标板 JTAG 管脚 1 的对应。

d. 将 52-pin 的电缆连接到 ICE 2 前面的 RC 1 run control 端口。

e. 将电缆另一端连接到自带适配头。

f. 为 Wind River ICE 2 上电。ICE 2 电源支持 110-240 VAC 50/60 Hz 到 12V DC@5.5A 的转换。包含 4 芯 DIN 连接口。按下 ICE 2 前端的 Power 开关，LED 屏将点亮。

注： Wind River ICE 2 支持对目标板 JTAG 接口的热插拔。

3. 建立目标连接

现在，主机、仿真器和目标板都已经连接好，需要在 Workbench 中建立目标连接，以使仿真器能够控制处理器的工作。

a. 在 Workbench 的左下角可以看到 Remote Systems 的窗口。

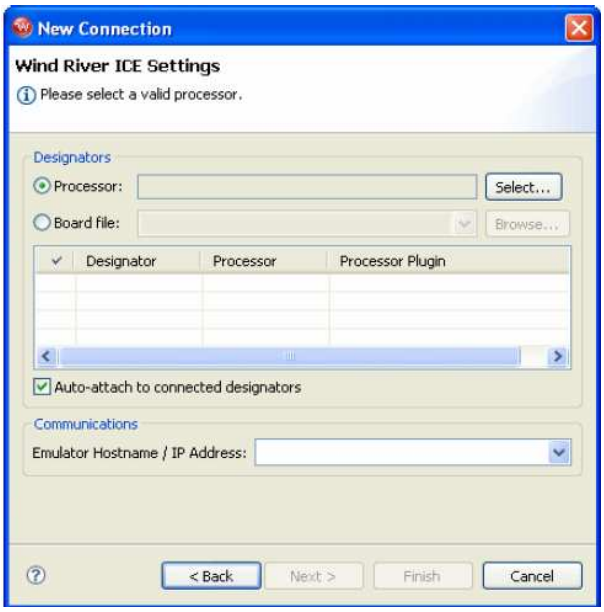
b. 在 Remote Systems 窗口中点击 Define a connection to remote system 来启动 New Connection Wizard。

c. 在连接列表上，选择 Wind River OCD Probe Connection，Wind River OCD ICE Connection，或 Wind River OCD ICE 2 Connection。

如果用的是 ICE SX 或 ICE 2，会进入 ICE 通讯设置。选择 Specify all communication setting manually，并点击 Next。

如果是 Probe，直接点击 Next。

d. 接下来会要求指定目标处理器类型或板卡文件(board file)，板卡文件将在后面介绍。



e. 点击 Select。在随后的列表中选择处理器类型，点击 OK。

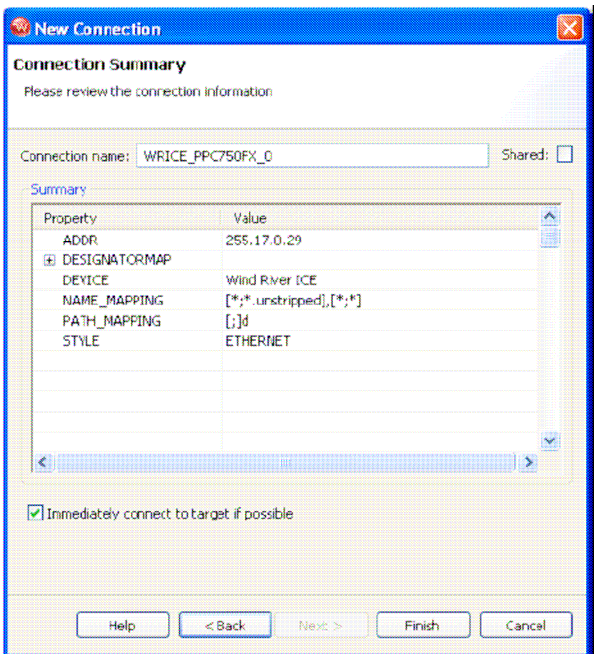
注意：如果目标处理器包含一个扫描链上的多个逻辑设备（例如多核，或包含其它 FPGA，CPLD 设备），必须指定板卡文件(board file)。

f. 指定仿真器连接。

如果是 Probe，你的 Probe 序列号会自动出现在 USB Device Names 区域；如果是 ICE SX，在 Emulator Hostname/IP Address 中输入 IP 地址；如果是 ICE 2，将前面 LED 屏上显示的 IP 地址输入到 Emulator Hostname/IP Address 中。

g. 点击 Next。

h. 后续的几步均采用缺省设置，点击 Next 直到进入 Connection Summary 画面。



i. 点击 Finish 完成连接设置。

Workbench 将建立连接，在 Remote Systems 窗口会显示连接项。

4. 向目标板下载代码

开始调试程序之前，你需要初始化目标板处理器，设置寄存器。分为两步来实现：首先在仿真器内存中设置寄存器，然后利用 IN 初始化命令将仿真器中的寄存器设置复制到目标板上。

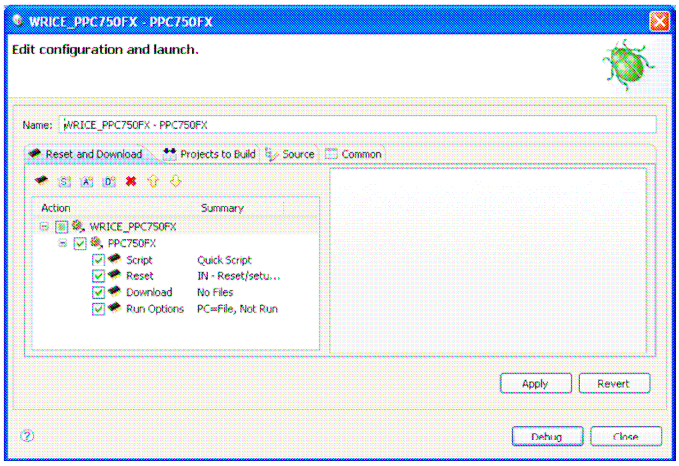
Wind River 仿真器采用底层命令 SCGA 来设置寄存器的值。这些底层命令的集合被存储在被称为**寄存器文件**的脚本里，一种扩展名为*.reg 的文本文件。Wind River 硬件参考设计板的寄存器文件在 *installDir*/workbench-3.x/ocd/*build*/RegisterFiles 目录中。

Wind River 还提供大量用于通用板卡的寄存器文件。查询支持的板卡和寄存器列表，请参考 **https://portal.windriver.com/windsurf/appnotes/hwTools/index.html** 上的 *Application Note313：Register Files/Board Files Cross-Reference*。

如果没有需要的寄存器文件，或板卡没有程序来初始化处理器的寄存器，那么需要建立一个寄存器文件。更多信息，请参考*Wind River Workbench for On-Chip Debugging User Tutorials: Configuring Target Registers*。你可以在不设置寄存器情况下调试，只是此时不可以访问存储器和外设。

要配置寄存器，初始化目标板，下载代码和符号信息到目标板中，执行以下步骤：

5.1 在Remote Systems窗口，选中目标连接的名字，点击Reset and download setting for this core。随后Reset and Download窗口出现。



5.2 指定Script选项。

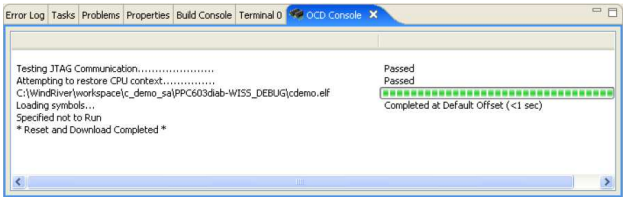
5.3 选择Play script file，点击Browse，选择寄存器文件。

5.4 指定Download选项。

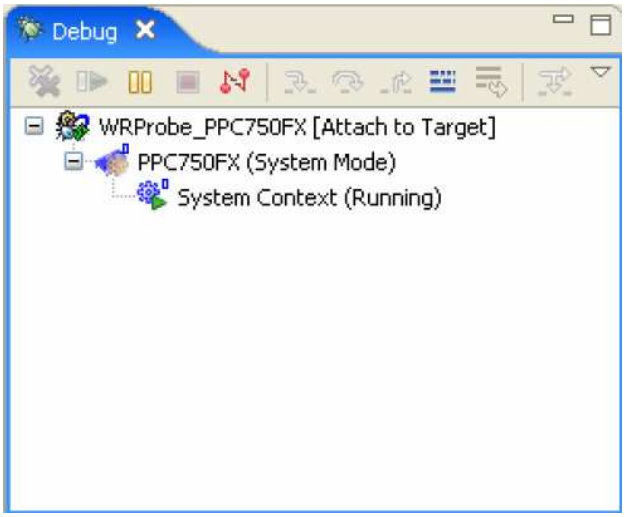
5.5 点击Add Files，找到*installDir*/standalone-1.0/samples/c_demo_sa/*target*/cdemo/Debug，选择cdemo.elf，点击Open。

5.6 确认右侧的Download和Load Symbols被选中，Verify设为None。

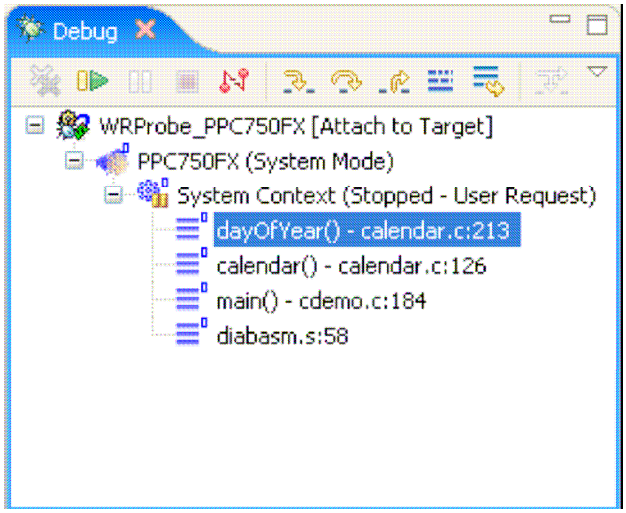
5.7 其余采用缺省设置，点击Debug，OCD Console窗口出现。OCD Console显示Workbench下载例程代码的过程。在Debug窗口，Workbench创建了System Context，Debug，窗口包含了调试所用的各个工具，如run control按钮，可以start，stop处理器，或在应用程序代码中单步执行。



5.8 要运行代码，点击Debug窗口中的Resume按钮。Workbench将Debug窗口中的System Context状态更改为Running。



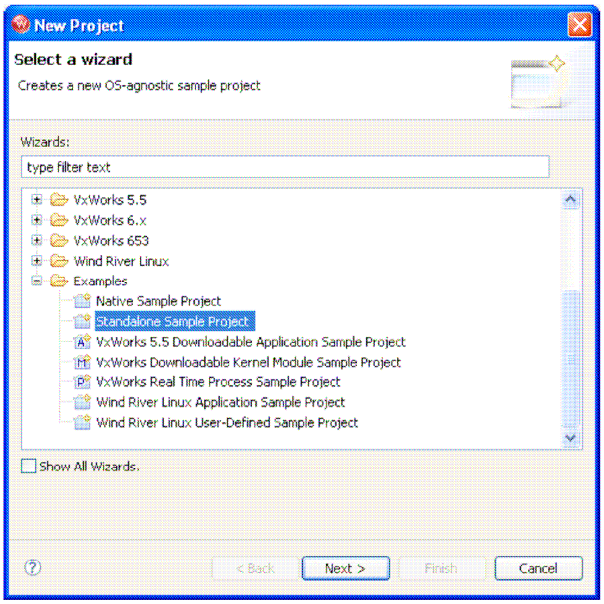
5.9 要停止代码的运行，点击Debug窗口的Suspend按钮。Workbench将Debug窗口的System Context状态更改为Stopped，并更新代码编辑框的代码状态。



5. 利用 Wind River Compiler 创建工程

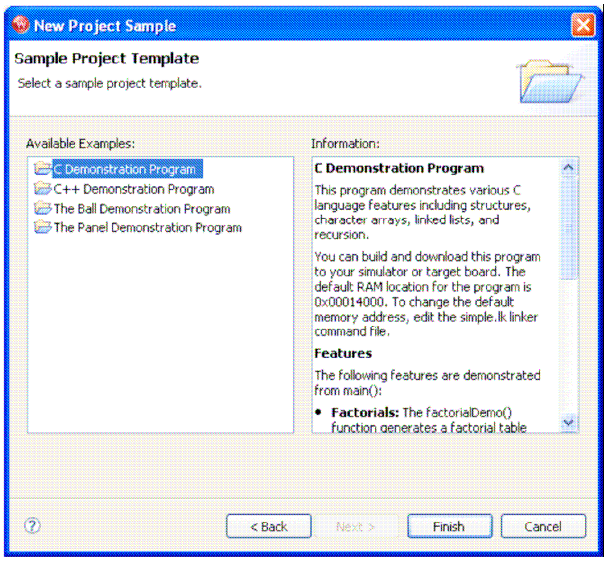
在安装 Wind River Compiler 后，会带有 C Demonstration 程序，通过以下步骤进行创建：

6.1 在 Workbench 工具栏中，选择 File>New>Project。新建工程向导出现。

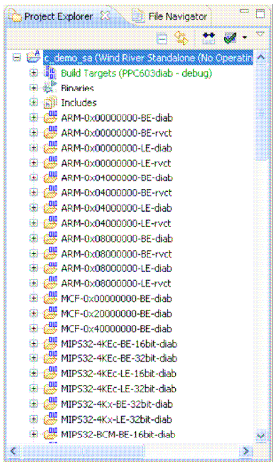


6.2 扩展 Examples，选择 Standalone Sample Project。

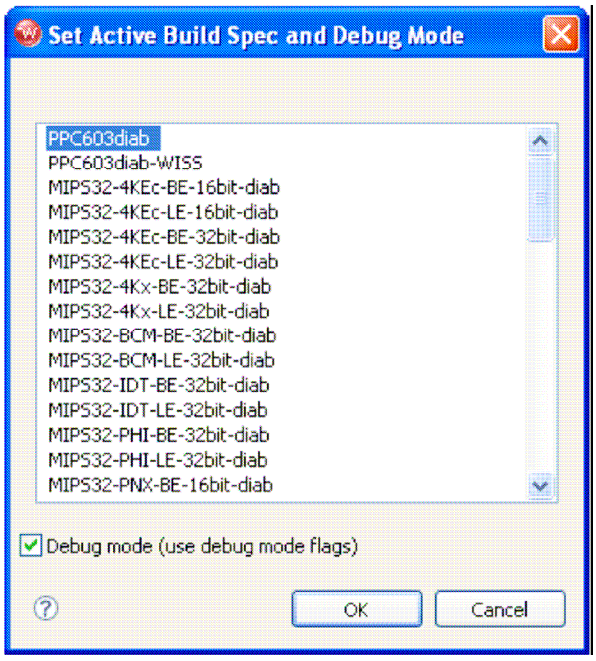
6.3 点击 Next。出现工程样本。



6.4 选择 C Demonstration Program，点击 Finish。Workbench 将在缺省 workspace 目录中创建工程，并在 Project Explorer 窗口中显示 c_demo_sa 的工程。



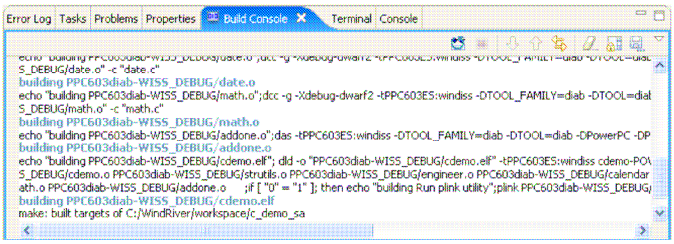
6.5 要编译工程，右击 c_demo_sa，选择 Build Options>Set Active Build Spec。Set Active Build Spec and Debug Mode 对话框出现。



6.6 选择适用于你所用目标板处理器的编译选项，例如 PPC603diab。

6.7 确认 Debug mode (use debug mode flags)被选中（这样 Workbench 会生成符号调试信息），点击 OK。

6.8 右击工程名，选择 Build Project。Workbench 将编译工程，编译结果会显示在 Build Console 窗口。



附录一 常用的 OCD 命令

OCD 常用命令指的是在 Workbench 中的 OCD Command Shell 里执行的底层命令，熟练使用这些命令能够快速，准确的获得目标板信息，配合 Workbench GUI，能够更加快速的调试目标程序，定位问题。下面介绍的是我们在调试中经常会用到的命令。

1. 初始化命令 inn 和 in

inn：复位 CPU，但不设置核寄存器的值，此时核寄存器的值均为缺省值。inn 命令不向目标板处理器下载寄存器设置或片选设置(这些设置保存着仿真器中)，inn 仅仅将目标板置于 background 模式。通常情况下，在调试 bootloader（VxWorks bootrom；Linux u-boot）时，需要 inn 命令，执行此命令后 PC 会在处理器的复位向量处，所有的寄存器均为缺省值，然后 bootcode 会设置初始化这些寄存器。如果要用 OCD 重新启动目标板，在 Command Shell 下执行：inn;go。

```
>BKM>inn
*******************************************************************************
Wind River ICE Initialization Sequence.
Copyright (c) Wind River Systems, Inc., 1999-2004. All rights reserved.
*******************************************************************************
Support Expires....... 5/17/07
Target Processor...... MPC8260
Wind River ICE            Group ID#= 0
Wind River ICE             Serial#= W0000000       Firmware= vn2.3a
Type CF For a Menu of Configuration Options
Initializing Background Debug Mode.............Successful
>BKM>
```

in：类似于 inn，将目标板置于 background 模式。除了复位 CPU 外，在建立正确的通讯后，仿真器会向目标板传送片选表(可用 CS 命令修改)和存储在仿真器中的寄存器设置(可用 SC 命令修改)。如果之前下载过寄存器文件，那么寄存器文件里的寄存器设置会被存储在仿真器里的文件系统中，每次 in 命令会将这些寄存器设置写到目标板中。

```
>BKM>in
*******************************************************************************
Wind River ICE Initialization Sequence.
Copyright (c) Wind River Systems, Inc., 1999-2004. All rights reserved.
*******************************************************************************
Support Expires....... 5/17/07
Target Processor...... MPC8260
Wind River ICE             Group ID#= 0
Wind River ICE             Serial#= W0000000       Firmware= vn2.3a
Type CF For a Menu of Configuration Options
Initializing Background Debug Mode.............Successful
>BKM>
```

通常情况下，如果用户要调试 RAM 中的代码，有两种方式：用 bootloader 初始化存储器和外设；用寄存器文件初始化存储器和外设。如果用寄存器文件的方式，就需要在下载寄存器文件后执行 in 命令。

2. cs 片选命令

cs 命令能够显示当前目标板的片式设置。

```
>BKM>cf tar 8260
>BKM>cs
Name BA        AM        AT ATM PS PARE WP MS V CSNT/SAM ACS  BI SCY SETA TRLX
CS0  FFC00000  FFC00000  0  0   16  No   RW GP Y Normal   1/2  Y  6ws Int. Norm
CS1  00000000  00000000  0  0   32  No   RW GP Y Normal   0    N  0ws Int. Norm
CS2  00000000  FFC00000  0  0   32  No   RW UB Y I.M.AMB  0    N  0ws Int. Norm
CS3  00000000  00000000  0  0   32  No   RW GP N Normal   0    N  0ws Int. Norm
CS4  00000000  00000000  0  0   32  No   RW GP N Normal   0    N  0ws Int. Norm
CS5  00000000  00000000  0  0   32  No   RW GP N Normal   0    N  0ws Int. Norm
CS6  00000000  00000000  0  0   32  No   RW GP N Normal   0    N  0ws Int. Norm
CS7  00000000  00000000  0  0   32  No   RW GP N Normal   0    N  0ws Int. Norm
```

cs *片选名称* 可以修改某个片选的设置，修改后用 in 命令传送至目标板

```
>BKM>cs cs0
00000000 -> FFFF8000                |      Base Register = FFC00000     >
00000000 -> FFFF8000                |      Address Mask = FFC00000      >
0 -> 7                              |      Address Type = 0             >
0 -> 7                              |      Address Type Mask = 0        >
(0-2)=32, 8, 16 bits / 3=Rsvd       |      Port Size = 16 Bits          >
0 = Disabled, 1 = Enabled           |      Parity Enable = Disabled     >
0 = Read/Write, 1 = Read Only       |      Write Protect = Read/Write   >
(0-3) = GPCM, Rsvd, UPMA, UPMB      |      Machine Select = GPCM        >
0 = Not Valid, 1 = Valid            |      Valid state = Valid          >
(0-3)=Norm, Rsvd, 1/4 clk, 1/2 clk  |      Address/CS Setup = 1/2 clk   >
```

3. cf 命令

cf 命令的设置选项很多，而且会因处理器的不同而不同，详细请参考 *installDir*\docs\extensions\eclipse\plugins\com.windriver.ide.doc.wr_workbench_ocd\wr_workbench_ocd_cf_options_ref_3.0\wr_workbench_ocd_cf_options_ref_3.0.pdf 参考手册。在 Workbench 的 Window>Show View>CF Options 也能够显示 CF 选项。有几个比较重要的 cf 设置：

a. **Set Breakpoint** SB[SB,IHBC] = SB，此选项通常情况下不修改，但如果调试 Flash 中的 bootcode 时要使用 step over 来调试代码，需要修改为”cf sb ihbc”，否则 step over 会遇到不能设置软件断点的错误。

b. **MMU 和 BL**，此选项缺省为 DISABLE，但如果要调试 Linux 操作系统，需要修改为 ENABLE，”cf mmu enable”, “cf bl enable”。其中 MMU 使能让仿真器可以建立 mmu 转换列表，以便可以在 Linux 初始化 MMU 前在虚地址设置断点，而 BL 使能可以向 Linux 传递启动参数。详细请参考 *installDir*\docs\extensions\eclipse\plugins\com.windriver.ide.doc.wr_workbench_ocd\wr_workbench_ocd_debug_tutorials_3.0\wr_workbench_ocd_debug_tutorials_3.0.pdf 中第 17 章 On-Chip Debugging for Linux。

其中 MMU 还要配合 mmua 命令建立映射表，例如”mmua c0000000 00000000 f0000000 p 00000000”，具体含义参考上面提到的手册。

c. **Trpexp**，此选项通常为 YES，此时仿真器会截获目标板代码运行过程中产生的异常事件，如果选择 NO，那么用户的代码中要有异常处理程序。注意在调试 Linux 中，应设置为”cf

```
>BKM>cf
Set BreakPoint                                SB[SB,IHBC] = SB
Vector Table Location                         VECTOR[HIGH,LOW,IGNORE] = LOW
Monitor Target reset                          RST[YES,NO,HALT,RUN] = YES
Target CPU                                    TAR[AUTO,603E,EC603E,603P,603R,740
                                               745,750,750CX,750CXE,750FX,750GX,
                                               755,7400,7410] = 750FX
Target CPU( SLAVE )                           SLAVE[NONE,8260] = NONE
Slave IMMR reset value                        SLIMMRVAL[AUTO,VALUE] = AUTO
JTAG clock rate                              CLK[0.025,0.3,0.5,1,3,6,12,16] = 16
Application IMMR Exclusion Range              AIMMRER[OFF,START and END] = OFF
Application IMMR Value                        AIMMRVAL[VALUE] = 0e000000
Real time Preservation                        RTP[YES,NO] = NO
Little Endian Mode                            LENDIAN[YES,NO] = NO
Processor Mode                                MODE[32,64] = 64
Download Mode                                 DLD[NORMAL,8] = NORMAL
Emulator HRESET Control                       HRESET[ENABLE,DISABLE] = ENABLE
Emulator HRESET Command Control               CMDRST[IN,RST,BOTH] = BOTH
Data Parity Checking                          PAR[YES,NO] = NO
Set Work Space                             WSPACE[BASE and SIZE] = 00000000 fff
Set Stack Range                               STACK[OFF / LOWER and UPPER] = OFF
Target Console Redirection                    TGTCONS[TGTCONS,BDM] = TGTCONS
Drive TReset line                             TRESET[OPENC,ACTIVE] = ACTIVE
External Trigger In            TRGIN[OFF,LEVELHI,LEVELLO,EDGEHI,EDGELO] = OFF
Trigger In Filter Mode                        TRGINFILTER[OFF,ON] = OFF
Trigger Out Mode              TRGOUTMODE[OFF,ONALLSTOPS,ONBREAKPOINT] = OFF
External Trigger Out        TRGOUT[LEVELHI,LEVELLO,PULSEHI,PULSELO] = LEVELHI
Invalidate Instruction Cache on GO            INVCI[YES,NO] = YES
Reset Pulse Length N*1ms                      RPL[1..2000] = 1
Sense Power via HRESET                        SPOWER[YES,NO] = YES
Power On Reset Length N*1ms                   PONR[0..500] = 0
CPU Reset Type       RESET[HRESET,SRESET,HRESET_UNFILTER,SRESET_UNFILTER] =
                                                                    HRESET
Trap exception                          TRPEXP[YES,NO,SOI,BREAKPOINTONLY] = YES
Issue an IN on coldstart                      INCOLD[YES,NO] = NO
Display L2 Data Cache Warning                 L2WARNING[YES,NO] = NO
Logic Analyzer Trace                    LATRACE[NONE,AGILENT,TEKTRONIX] = NONE
Memory Management Unit Mode                   MMU[ENABLE,DISABLE] = DISABLE
Load Boot Table On IN                         BL[ENABLE,DISABLE] = DISABLE
Trigger In Report Mode                        BRKREP[REPONLY,BRKREP] = BRKREP
TMD Mode                                      TMD[ENABLE,DISABLE] = DISABLE
Run Counter Length                            RCL[1000..FFFF] = 1000
Delay after Reset Nms                         DRST[0..10000] = 25
>BKM>
```

trpexp breakpointonly”，此时仿真器仅仅处理断点，而不处理异常事件，否则 Linux 启动过程中会不断出现异常并被仿真器截获，进入 BKM 模式。Linux 有自己的异常处理程序，不需要进入 BKM 模式进行调试。

d. **TMD 模式**，缺省为 DISABLE。此选项仅仅用于调试 VxWorks 操作系统。TMD 是 Transparent Mode Debug 缩写。如果用户想用 WDB 方式调试，目标板的以太网不可用，不稳定，或根本没有以太网资源，此时可以用 Wind River ICE 或 ICE 2 通过 JTAG 进行 WDB 调试，称为 TMD 模式。在 VxWorks 组件中要包含 TMD 组件，此组件与网络组件互斥。此时 CF 选项中的 TMD 就要设为”cf tmd enable”。**Wind River Probe 不支持这一功能**。

e. **INCOLD**，缺省为 NO。如果设置为 YES，”cf incold yes”，那么下次进行目标板与仿真器连接时，仿真器会向目标板发送 in 命令，连接完毕后进入 BKM 模式，PC 在复位向量地址。但通常情况用户在连接目标板时不想改变目标板目前的运行状态，那就需要设置为 NO，连接完毕后进入 RUN>状态，并不干扰目标板目前代码的运行。

f. 有一个特殊的 CF 命令”cf tflash reverse”，在手册里没有提到，当用户的 flash 连接是字节反转的，此时要执行此 cf 命令。我们的仿真器针对不同的 flash 器件都有两套 flash 烧写算法，一种是 normal 的，一种是 reverse 的，也即支持字节反转连接。

其余 CF 设置因处理器不同而设置不同，请参考 CF Options 手册。

4. 显示/修改内存命令 DM/SM

DM 命令(display memory)有四种 DMB(以字节形式显示)；DMW(以字的形式显示)；DML(以长字的形式显示)；DMS(以字符串的形式显示)。直接用 DM 时，缺省是以字的形式显示。注：还有一种 DMD，用于显示双字 64-bit。

```
>BKM>DM 200010 20
00200010: FFFF DFFF FFFF FFFF FFFF FFFF FFFF FFFF     ................
00200020: FFFF FFFE FFFF FFFD FFFF FFFF FFFF FFFF     ................
00200030: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF     ................
00200040: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF     ................
>BKM>
```

```
>BKM>dmb 300300 5
00300300: FF FF FF FF FF      .....
>BKM>
```

```
>BKM>dmd 200010 20
00200010: FFFFDFFFFFFFFFFF FFFFFFFFFFFFFFFF     ................
00200020: FFFFFFFEFFFFFFFD FFFFFFFFFFFFFFFF     ................
00200030: FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF     ................
00200040: FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF     ................
00200050: FFFFFFFFFFFFFFFF FDFFFFFDFFFFFFBF     ................
00200060: FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF     ................
00200070: FFFFFFFFFFFFFFFF FFFFFFFFFFFFFF7F     ................
00200080: FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFE     ................
00200090: FFFFFFFFFFFFFFF7 FFFFFFFFFFFFFFFF     ................
002000A0: FFFFFFFFFFFFFFFF F7FFFFFFFFFFFFFD     ................
002000B0: FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF     ................
002000C0: FFFFFFFFDFFFFFFF FFFFFFFFFFFFFFFF     ................
002000D0: FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF     ................
002000E0: FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF     ................
002000F0: FFFFFFFBFFFFFFFF FFFFFFFFFFFFFFFF     ................
00200100: FFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF     ................
>BKM>
```

SM 命令(Set Memory)类似，用于修改内存里的值，分为三种(没有 S)：SMB(以字节形式修改)；SMW(以字的形式修改)；SML(以长字的形式修改)，SM 缺省采用字的形式修改。注：还有一种 SMD，以双字 64-bit 形式修改。

```
BKM>DMB 200010 3
200010: FF 0F 00
BKM>SMB 200010 0B 7F 34
BKM>DMB 200010 3
200010: 0B 7F 34
BKM>
```

```
>BKM>dmd 200010 3
00200010: FFFFDFFFFFFFFFFF FFFFFFFFFFFFFFFF    ................
00200020: FFFFFFFFFFFFFFFD    ................
>BKM>smd 200010 0B 7F 34
>BKM>dmd 200010 3
00200010: 000000000000000B 000000000000007F    ................
00200020: 0000000000000034    .......4........
>BKM>
```

SM 与 MM 命令的区别：MM(Modify Memory)命令只修改内存中的数值，SM 命令除了修改内存值，还要读回来进行校验以确认写入的值是正确的。

5. 显示/修改寄存器命令 DR/SR

DR(Display Register)命令用于显示寄存器的值，可以单独使用 DR 命令，也可以用：dr *寄存器名* 来显示单个寄存器的值。

```
>BKM>dr
R00     = 00E17F2C R01     = 00EFFF30 R02     = 00E07FF0 R03     = 00000001
R04     = 00000000 R05     = 00EFFF38 R06     = 00000001 R07     = 00EFFF3C
R08     = 000D1788 R09     = 00000008 R10     = 00000000 R11     = 00000000
R12     = 00000000 R13     = 00E22C00 R14     = 00E2847C R15     = 00000000
R16     = 00000000 R17     = 00E294F8 R18     = E396FFBE R19     = ADAEF77B
R20     = 762EFEDD R21     = F813F3A2 R22     = F3E71E7F R23     = A836CDEE
R24     = 7B7F9EFE R25     = 593A65BF R26     = 2513F4FB R27     = E2FB608F
R28     = EC6DCFDF R29     = A9F275EB R30     = 00E23144 R31     = 00E24310
CR      = 20000000 MSR     = 00009042 LR      = 00E19870 SRR0    = 00E1986C
SRR1    = 00009002 SPRG0   = 00000000 SPRG1   = 00000000 SPRG2   = 00000000
SPRG3   = 00000000 XER     = 00000300 CTR     = 00000000 PC      = 00E198BC
>BKM>
>BKM>SR R00 1000000 R03 120000 R08 1234 R16 55
>BKM>DR R08 R16 R00 R03
R08     = 00001234 R16     = 00000055 R00     = 01000000 R03     = 00120000
>BKM>
```

SR(Set Register)命令用于设置修改寄存器的值，用法为：sr *寄存器名 寄存器值*。

```
>BKM>dr r00 r01
R00     = 00000000 R01     = 00001234
>BKM>sr r00 12345678 r01 88888888
>BKM>dr r00 r01
R00     = 12345678 R01     = 88888888
>BKM>
```

6. 显示/删除断点命令 DB/RB

DB(Display Breakpoint)命令显示目前断点状态，包括有几个断点，什么类型的断点，断点地址是什么。

```
>BKM>db
    Software Code Breakpoints
1.  00040418  count = 0001 actual = 0000 enabled
2.  0004042C  count = 0001 actual = 0000 enabled
3.  0004044C  count = 0001 actual = 0000 enabled
4.  00040494  count = 0001 actual = 0000 enabled
!INFO! - [msg82001] No internal hardware breakpoints installed
>BKM>
```

RB(Remove Breakpoint)命令删除一个或全部的断点设置。RB *断点地址* 将删除此地址的断点设置，如果不给参数，只执行 RB 命令，会删除所有的断点设置。

```
>BKM>db
Software Code Breakpoints
1.  00010000  count = 0001 actual = 0000 enabled
2.  00010010  count = 0001 actual = 0000 enabled
!INFO! - [msg82001] No internal hardware breakpoints installed
>BKM>RB 10000
>BKM>db
Software Code Breakpoints
1.  00010010  count = 0001 actual = 0000 enabled
!INFO! - [msg82001] No internal hardware breakpoints installed
>BKM>
```

SB 是用于设置软件断点，IHBC 用于设置硬件断点。软件断点没有个数限制，但不能设置到 Flash；硬件断点个数取决于 CPU，可以设置到任何可访问的地址。

7. 运行/停止命令 GO/HA

GO 是目标板程序从当前 PC 地址开始执行，HA(同 HALT 或 Ctrl+c)停止程序运行，进入 BKM 调试模式。这里衍生出一个用于多核调试的 GOS 和 HALTS 的命令，其中 S 指 Synchronize，如果 SCTRL 表里设置的为并行调试模式，此时 GOS 和 HALTS 会并行让所有的核运行和停止，**Wind River Probe 不支持 GOS 和 HALTS 命令**。

8. SY 命令

SY 命令后面可以跟不同的参数，执行 SY 命令可以查看所支持的命令，对于不同的处理器，SY 支持的命令有所不同。此命令在诊断硬件上的问题时经常用到，主要是以下一些命令：

a. **sy jtag stat** 此命令是查看 JTAG 链上有几个设备（CPU/Core），对于 PowerPC 单核，通常会看到以下输出：

```
>BKM>sy jtag stat
Total number of devices on the scan chain   = 1
Total IR Register Length                    = 8
```

对于 MIPS 单核，会有以下输出：

```
>BKM>sy jtag stat
Total number of devices on the scan chain   = 1
Total IR Register Length                    = 5

Total EJTAG_ADDR Register Length            = 32
Total EJTAG_DATA Register Length            = 32
Total EJTAG_CTRL Register Length            = 32
```

如果遇到 Device 数目和 IR 长度均为 1 的情况，说明在硬件上 JTAG 连接有问题，可能是 ICE 连接头或者 PCB 目标板上的 JTAG 连接头出现故障。

b. **sy rev** 此命令通常作用于 PowerPC 处理器，可以读回 PVR 寄存器存储的芯片信息

```
>BKM>sy rev
Part PVR  Value = 0x80811014, Voyager/Kaluah 82xx Revision 0.14
```

c. sy prog1 *address* 此命令用于测试 RAM 是否可用，通常在 Flash 烧写过程中如果出现“无法装载算法到某某地址”时，要用到此命令测试放置算法的 RAM 地址是否可用。完整测试命令如下：

```
>BKM>sy prog1 5000              -将存储在仿真器中的测试程序下载到0x5000地址处
>BKM>sr pc 5000                 -将PC设置到测试程序起始地址
>BKM>di                         -查看0x5000处的指令，确认是否存在
 $00005000   : 0x60000000 :ppc nop
 $00005004   : 0x60000000 :ppc nop
 $00005008   : 0x60000000 :ppc nop
 $0000500C   : 0x60000000 :ppc nop
 $00005010   : 0x7C0004AC :ppc sync
 $00005014   : 0x4BFFFFF0 :ppc b               0x5004
 $00005018   : 0x00000000 :ppc dc.l            0x0
 $0000501C   : 0x00000000 :ppc dc.l            0x0
 $00005020   : 0x00000000 :ppc dc.l            0x0
 $00005024   : 0x00000000 :ppc dc.l            0x0
 $00005028   : 0x00000000 :ppc dc.l            0x0
 $0000502C   : 0x00000000 :ppc dc.l            0x0
 $00005030   : 0x00000000 :ppc dc.l            0x0
 $00005034   : 0x00000000 :ppc dc.l            0x0
 $00005038   : 0x00000000 :ppc dc.l            0x0
 $0000503C   : 0x00000000 :ppc dc.l            0x0
 $00005040   : 0x00000000 :ppc dc.l            0x0
 $00005044   : 0x00000000 :ppc dc.l            0x0
 $00005048   : 0x00000000 :ppc dc.l            0x0
 $0000504C   : 0x00000000 :ppc dc.l            0x0
>BKM>go                                        -运行程序
>RUN>dr pc                                     -查看当前PC
 PC       = 00005010
>RUN>sb 5008                                   -在0x5008处设置断点

>RUN>

!BREAK! - [msg12000] Software breakpoint; PC = 0x00005008 [EVENT
Taken]
>BKM>rb                                        -删除所有断点
>BKM>
```

如果测试结果如上所示，证明 RAM 可用。

9. 其它一些命令

a. **DI**：反汇编。后面可以跟两个参数：第一个参数是地址，即从某个地址进行反汇编；第二个参数是长度，即列出多少条指令。如果不跟任何参数，DI 从当前地址反汇编

```
>BKM>di 5000 20
 $00005000   : 0x60000000 :ppc nop
 $00005004   : 0x60000000 :ppc nop
 $00005008   : 0x60000000 :ppc nop
 $0000500C   : 0x60000000 :ppc nop
 $00005010   : 0x7C0004AC :ppc sync
 $00005014   : 0x4BFFFFF0 :ppc b               0x5004
 $00005018   : 0x00000000 :ppc dc.l            0x0
 $0000501C   : 0x00000000 :ppc dc.l            0x0
 $00005020   : 0x00000000 :ppc dc.l            0x0
 $00005024   : 0x00000000 :ppc dc.l            0x0
 $00005028   : 0x00000000 :ppc dc.l            0x0
 $0000502C   : 0x00000000 :ppc dc.l            0x0
 $00005030   : 0x00000000 :ppc dc.l            0x0
 $00005034   : 0x00000000 :ppc dc.l            0x0
 $00005038   : 0x00000000 :ppc dc.l            0x0
 $0000503C   : 0x00000000 :ppc dc.l            0x0
 $00005040   : 0x00000000 :ppc dc.l            0x0
 $00005044   : 0x00000000 :ppc dc.l            0x0
```

```
$00005048   : 0x00000000 :ppc dc.l           0x0
$0000504C   : 0x00000000 :ppc dc.l           0x0
$00005050   : 0x00000000 :ppc dc.l           0x0
$00005054   : 0x00000000 :ppc dc.l           0x0
$00005058   : 0x00000000 :ppc dc.l           0x0
$0000505C   : 0x00000000 :ppc dc.l           0x0
$00005060   : 0x00000000 :ppc dc.l           0x0
$00005064   : 0x00000000 :ppc dc.l           0x0
$00005068   : 0x00000000 :ppc dc.l           0x0
$0000506C   : 0x00000000 :ppc dc.l           0x0
$00005070   : 0x00000000 :ppc dc.l           0x0
$00005074   : 0x00000000 :ppc dc.l           0x0
$00005078   : 0x00000000 :ppc dc.l           0x0
$0000507C   : 0x00000000 :ppc dc.l           0x0
```

b. **SYNC** 此命令通常用于 PowerPC 和 MIPS 处理器，可以同步仿真器和处理器的连接。当目标板非正常重启或其它意外操作导致仿真器和处理器断掉连接时，用 SYNC 可以快速重建连接。该命令不会复位目标板。

```
>BKM>

!FAULT! - [msg20003] Unexpected reset while target is stopped [EVENT
Taken]
>ERR>
>ERR>sync

Synchronization Complete. Current PC = 0x00101c90
>BKM>
```

c**. DC VER** 命令，显示当前处理器名称，仿真器 firmware 版本等等信息

```
>BKM>dc ver
Target = MPC8260 :PPC82XX :Firm Rev = pr3.9a :NOEVENT :Wind River
Probe :TF Library = D5.9m :Slave = NONE :FLEXLM
```

10. SC Default：将仿真器中所存储的对寄存器的设置恢复为初始值

11. SCT：列出当前目标设备中所有寄存器的设置，

```
>BKM>sct
****************** LA                                  *****************
CCSRBAR                          E0000000  000E0000       ALTCBAR
E0000008  00000000
ALTCSR                           E0000010  00000000       BPTR
E0000020  00000000
LAWBAR0                          E0000C08  00000000       LAWAR0
E0000C10  00000000
LAWBAR1                          E0000C28  00000000       LAWAR1
E0000C30  80F0001B
LAWBAR2                          E0000C48  00080000       LAWAR2
E0000C50  8000001C
LAWBAR3                          E0000C68  000E2000   *   LAWAR3
E0000C70  80000017   *
LAWBAR4                          E0000C88  000F0000   *   LAWAR4
E0000C90  8040001B   *
```

```
LAWBAR5                          E0000CA8  00000000  *   LAWAR5
E0000CB0  00000000  *
LAWBAR6                          E0000CC8  00000000  *   LAWAR6
E0000CD0  00000000  *
LAWBAR7                          E0000CE8  00000000      LAWAR7
E0000CF0  00000000
```

12. SCT COPY：将目标设备中寄存器的值拷贝到仿真器中，通常用于 bootcode 初始化目标板，然后将寄存器值存入仿真器。然后从仿真器中导出寄存器文件，可以用于其它没有 bootcode 的同型号目标板。

13. PJ UPLOAD：列出当前仿真器中对所有寄存器的设置，这些设置会输出到 OCD Command Shell 下，可以将其复制到文本文件中，存成.reg 文件。

```
>BKM>pj upload
REM ****************************************
REM CF CONFIGURATION
REM ****************************************

CF TAR                                  8548E                                   ;
OPERATION
CF SB                                   SB                                      ;
OPERATION
CF SBE                                  NORMAL                                  ;
OPERATION
CF MMU                                  DISABLE                                 ;
OPERATION
CF BL                                   DISABLE                                 ;
OPERATION
CF RST                                  YES                                     ;
OPERATION
CF CLK                                  16                                      ;
OPERATION
CF LENDIAN                              NO                                      ;
OPERATION
CF DLD                                  NORMAL                                  ;
OPERATION
CF HRESET                               ENABLE                                  ;
OPERATION
CF TRESET                               ACTIVE                                  ;
OPERATION
CF RESET                                HRESET                                  ;
OPERATION
CF CHECKSTOP                            NO                                      ;
OPERATION
CF SPOWER                               YES                                     ;
OPERATION
CF PONR                                 0                                       ;
OPERATION
CF TRPEXP                               YES                                     ;
OPERATION
CF INCOLD                               NO                                      ;
OPERATION
CF WSPACE                               00005000 e95c                           ;
OPERATION
```

```
CF TGTCONS                      BDM                             ;
OPERATION
CF RTP                          NO                              ;
OPERATION
CF RPL                          1                               ;
OPERATION
CF ETM                          c2010                           ;
OPERATION
CF PORABIST                     YES                             ;
OPERATION
CF TMD                          DISABLE                         ;
OPERATION
CF BOOTMODE                     NORMAL                          ;
OPERATION
CF DDRDLL                       YES                             ;
OPERATION
CF LBCLK                        NO                              ;
OPERATION
CF DRST                         1                               ;
OPERATION
CF L2TLB                        00000000                        ;
OPERATION
CF FLUSH_DCACHE                 NO                              ;
OPERATION
CF TRCAQU                       OFF                             ;
OPERATION
CF TRCCLR                       YES                             ;
OPERATION
REM ****************************************
REM SC CONFIGURATION
REM ****************************************
SC GRP ERASE
SCGA LA                           CCSRBAR
00000000  000E0000 LA                          /va_dr /ua:0 /sa:0
/ue:0
SCGA LA                           ALTCBAR
00000008  00000000 LA                          /ua:0
SCGA LA                           ALTCSR
00000010  00000000 LA                          /ua:0
SCGA LA                           BPTR
00000020  00000000 LA                          /ua:0
SCGA LA                           LAWBAR0
00000C08  00000000 LA                          /ua:0
SCGA LA                           LAWAR0
00000C10  00000000 LA                          /ua:0
SCGA LA                           LAWBAR1
00000C28  00000000 LA                          /ua:0
SCGA LA                           LAWAR1
00000C30  80F0001B LA                          /ua:0
SCGA LA                           LAWBAR2
00000C48  00080000 LA                          /ua:0
SCGA LA                           LAWAR2
00000C50  8000001C LA                          /ua:0
SCGA LA                           LAWBAR3
00000C68  000A0000 LA                          /ua:0
SCGA LA                           LAWAR3
00000C70  8010001C LA                          /ua:0
SCGA LA                           LAWBAR4
00000C88  000E2000 LA                          /ua:0
SCGA LA                           LAWAR4
00000C90  80000017 LA                          /ua:0
```

```
SCGA LA                                 LAWBAR5
00000CA8  000E3000 LA                                   /ua:0
SCGA LA                                 LAWAR5
00000CB0  80100017 LA                                   /ua:0
SCGA LA                                 LAWBAR6
00000CC8  000F0000 LA                                   /ua:0
SCGA LA                                 LAWAR6
00000CD0  8040001B LA                                   /ua:0
SCGA LA                                 LAWBAR7
00000CE8  00000000 LA                                   /ua:0
SCGA LA                                 LAWAR7
00000CF0  00000000 LA
```

14. ESTKEY; ESTKEY DISPLAY： 查看当前仿真器和 Workbench 软件的 License 状况，包括 FlexLM 是否 enable，仿真器的序列号等等。

```
>BKM>estkey
262HCZRF2GLN3LJJXB95ZCPBOVQ31 (*) Main Key
>BKM>estkey display

                      Key In Use: Main Key
                FLEXLM licensing: Enabled
                   Serial Number: PR062078
                        Group Id: 0
```

15. SET VERBOSE ON: 通常用于连接仿真器目标板失败时，可以通过此命令获取更多的信息，以判断失败原因。

```
>BKM>in
************************************************************************
********
Wind River Probe Initialization Sequence.
Copyright (c) Wind River Systems, Inc. 1999-2009.  All rights
reserved.
************************************************************************
********
    Support Expires....... FlexLM key in use.
    Target Processor...... MPC8548E:U1
Wind River Probe    Group ID#= 0
Wind River Probe      Serial#= PR062078       Firmware= pr2.7a
Type CF For a Menu of Configuration Options
Initializing Background Debug Mode.............Successful
>BKM>set verbose on
>BKM>in
************************************************************************
********
Wind River Probe Initialization Sequence.
Copyright (c) Wind River Systems, Inc. 1999-2009.  All rights
reserved.
************************************************************************
********
    Support Expires....... FlexLM key in use.
    Target Processor...... MPC8548E:U1
Wind River Probe    Group ID#= 0
Wind River Probe      Serial#= PR062078       Firmware= pr2.7a
Type CF For a Menu of Configuration Options
```

```
Testing Communications to Hardware
Interface....Passed.........................
Driving HRESET to be High.....................Passed
Driving HRESET to be Low......................Passed
Waiting HRESET Low Acknowledge................Passed
Attempting JTAG communication.................Passed
Waiting for HReset to be released.............Passed
Testing for target STOP State.................Passed
Comparing target CPU with CF setting..........Passed
Waiting for HRESET High Acknowledge...........Passed
Testing JTAG Communication....................Passed
Loading Internal Registers....................Passed
Testing JTAG Communication....................Passed
Getting value of cf mmu option ...............Passed
Attempting to restore CPU context.............Passed
>BKM>
```

16. TA：此命令用于显示和设置 e500 core 中的 TLB，包括指令和数据，如 MPC85xx，QorIQ2xxx/4xxx

```
>BKM>ta
Entry  EPN          RPN      TID  TMASK   WIMGE  TSIZ U0:3  X0:1   PID
TS  PROT SHEN   UR   UW   UX   SR   SW   SX  TIDZ VAL

IT0  FFFFF000     FFFFF100 00     0FC     06     0     0     0     0
0    U    P    D    D    D    D    D    D    D    I
IT1  FFFFF000     FFFFF100 00     0FC     06     0     0     0     0
0    U    P    D    D    D    D    D    D    D    I
IT2  FFFFF000     FF000100 00     000     06     0     0     2     0
0    U    S    E    E    E    D    D    D    D    I
IT3  FFD8C000     10000000 00     000     06     0     0     0     0
0    U    S    E    E    E    D    D    D    D    I
DT0  FFB3C000     10000000 00     000     06     0     0     0     0
0    U    S    E    E    E    D    D    D    D    I
DT1  8004F000     F8000100 00     0C0     06     0     0     2     0
0    U    S    E    E    E    D    D    D    D    I
DT2  0000E000     E0000100 00     000     06     0     0     2     0
0    U    S    E    E    E    D    D    D    D    I
DT3  0000C000     00000000 00     000     06     0     0     0     0
0    U    S    E    E    E    D    D    D    D    I
LT0  FF000000     0FF00000 00     0FF     00     8     0     0     0
0    P    P    D    D    D    E    E    E    D    V
LT1  80000000     08000000 00     0FF     0A     9     0     0     0
0    P    P    D    D    D    E    E    E    D    V
LT2  90000000     09000000 00     0FF     0A     9     0     0     0
0    P    P    D    D    D    E    E    E    D    V
LT3  A0000000     0A000000 00     0FF     0A     9     0     0     0
0    P    P    D    D    D    E    E    E    D    V
LT4  B0000000     0B000000 00     0FF     0A     9     0     0     0
0    P    P    D    D    D    E    E    E    D    V
LT5  E0000000     0E000000 00     0FF     0A     8     0     0     0
0    P    P    D    D    D    E    E    E    D    V
LT6  F0000000     0F000000 00     0FF     00     8     0     0     0
0    P    P    D    D    D    E    E    E    D    V
LT7  F8000000     0F800000 00     0FF     0A     9     0     0     0
0    P    P    D    D    D    E    E    E    D    V
LT8  00000000     00000000 00     0FF     00     9     0     0     0
0    P    P    D    D    D    E    E    E    D    V
LT9  10000000     01000000 00     0FF     00     9     0     0     0
0    P    P    D    D    D    E    E    E    D    V
```

```
LT10 00000000     00000000 00     000     00     0     0     0     0
0    U    P    D    D    D    D    D    D    D    I
LT11 00000000     00000000 00     000     00     0     0     0     0
0    U    P    D    D    D    D    D    D    D    I
LT12 00000000     00000000 00     000     00     0     0     0     0
0    U    P    D    D    D    D    D    D    D    I
LT13 00000000     00000000 00     000     00     0     0     0     0
0    U    P    D    D    D    D    D    D    D    I
LT14 00000000     00000000 00     000     00     0     0     0     0
0    U    P    D    D    D    D    D    D    D    I
LT15 00000000     00000000 00     000     00     0     0     0     0
0    U    P    D    D    D    D    D    D    D    I
```

TA 命令后面可以跟参数，来单独设置某个 TLB Entry，具体设置可参考 Application Note:

AN0257_PPC85xx_TLB_Command.pdf

17. TF: 目标板 Flash 显示/配置命令

```
>BKM>tf
 - BDM TFlash programming Interface Settings -
Current device selected       :    INTEL V28F640Jx ( 8192 x  8 )  1
Device
Start of work space in target :    00005000..0001394C
Start address of the flash    :    FF800000..FFFFFFFF

 * Use 'tf conf' to change parameters
 * Use 'tf device' to change the device
```

TF CONF：用于配置 flash 参数，包括 flash 型号，flash 物理起始地址，flash 算法所在的 RAM 工作区域等

```
>BKM>tf conf
 - BDM TFlash programming Interface Settings -
Current device selected       :    INTEL V28F640Jx ( 8192 x  8 )  1
Device
Start of work space in target :    00005000 > 0
Start address of the flash    :    FF800000 > ff800000
>BKM>tf
 - BDM TFlash programming Interface Settings -
Current device selected       :    INTEL V28F640Jx ( 8192 x  8 )  1
Device
Start of work space in target :    00000000..0000E94C
Start address of the flash    :    FF800000..FFFFFFFF

 * Use 'tf conf' to change parameters
 * Use 'tf device' to change the device
```

采用 TF CONF 命令也可以在后面采用一连串参数的方式一次设置完毕，格式如下：

TF CONF *device_number RAM_workspace_address workspace_size base_address*

```
>BKM>tf conf 275 100000 2000 ff800000
275 00100000 1868 FF800000
>BKM>tf
 - BDM TFlash programming Interface Settings -
Current device selected       :    INTEL V28F640Jx ( 8192 x  8 )  1
Device
Start of work space in target :    00100000..0010074C
Start address of the flash    :    FF800000..FFFFFFFF

 * Use 'tf conf' to change parameters
 * Use 'tf device' to change the device
```

其中 flash 的设备号可以通过 TF DEVICE 命令查看：

```
>BKM>tf device
00:  SAMSUNG   KFG1216x2A-xxB5 (32768 x 16 ) 1 Device
01:  AMD    29LV004T (  512 x  8 )  2 Devices
02:  AMD    29LV004B (  512 x  8 )  1 Device
03:  AMD    29LV004B (  512 x  8 )  2 Devices
04:  AMD    29LV004B (  512 x  8 )  4 Devices
05:  AMD    29LV008BT( 1024 x  8 )  1 Device
06:  AMD    29LV008BB( 1024 x  8 )  4 Devices
07:  AMD    29F010   (  128 x  8 )  1 Device
08:  AMD    29F010   (  128 x  8 )  2 Devices
09:  AMD    29F010   (  128 x  8 )  4 Devices
10:  AMD    29F010   (  128 x  8 )  8 Devices
11:  AMD    29F040   (  512 x  8 )  1 Device
12:  AMD    29F040   (  512 x  8 )  2 Devices
13:  AMD    29F040   (  512 x  8 )  4 Devices
14:  AMD    29F040   (  512 x  8 )  8 Devices
15:  AMD    29F080/81( 1024 x  8 )  1 Device
16:  AMD    29F080/81( 1024 x  8 )  2 Devices
17:  AMD    29F080/81( 1024 x  8 )  4 Devices
18:  AMD    29F080/81( 1024 x  8 )  8 Devices
19:  AMD    29F016/17( 2048 x  8 )  1 Device
20:  AMD    29F016/17( 2048 x  8 )  2 Devices
21:  AMD    29F016/17( 2048 x  8 )  4 Devices
```

最左端的数字就是 flash 在仿真器中的设备号

TF ERASE：flash 擦除命令，后面跟要擦除空间的起始和结束地址

```
>BKM>tf conf 275 100000 2000 ff800000
275 00100000 1868 FF800000

>BKM>tf erase ff800000 ff900000

!ERROR! - [msg100020] WorkSpace to small for the algorithm size
>BKM>tf conf 275 100000 20000 ff800000
275 00100000 19788 FF800000
>BKM>tf erase ff800000 ff900000
INTEL V28F640Jx ( 8192 x  8 )  1 Device
Erasing Flash(s) ... Done
>BKM>dml ff800000 20
 FF800000: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF     ................
 FF800010: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF     ................
 FF800020: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF     ................
 FF800030: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF     ................
 FF800040: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF     ................
 FF800050: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF     ................
 FF800060: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF     ................
 FF800070: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF     ................
>BKM>
```

18.

附录二 寄存器文件的编写或修改

一． 简介

在下载任何应用程序代码并运行前，你都需要设置板上的内部寄存器值。通常有两种方式：bootcode 或寄存器文件。

配置寄存器值实际上分为两步：首先，寄存器文件中对寄存器的配置会保存在仿真器的存储器中，然后执行 IN 命令将仿真器的寄存器设置复制到目标板的内部寄存器中。

Wind River 仿真器里带有存储区域来保存对目标板寄存器的设置。Wind River Probe 和 Wind River ICE SX 采用 NVRAM；Wind River ICE 2 则采用板上 Flash。如果要选择某些寄存器的值写入目标板，可以在寄存器文件里 enable 或 disable 特定的寄存器组。

Wind River 仿真器采用底层命令设置寄存器的值。这些底层命令的集合被存储在称为**寄存器文件**的脚本里，一种扩展名为*.reg 的文本文件。

如果目标板的寄存器值已经设置过了，你可以将这些已经存着的设置上传给仿真器，或直接上传到你的主机。

操作目标板上的内部寄存器既可以通过 Wind River Workbench GUI 中的 Register 窗口，也可以通过在 Workbench 的 OCD Command Shell 中使用底层命令。

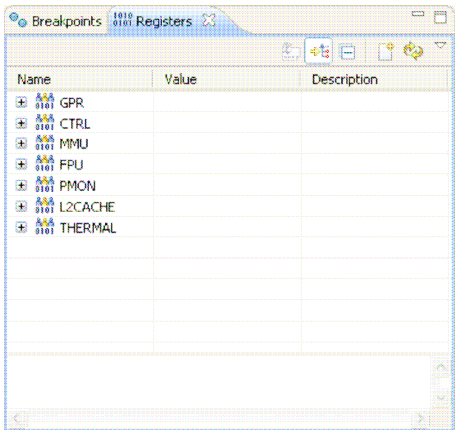
注：在下载寄存器文件后，一定注意要执行 IN 命令，此命令才真正将寄存器值写到目标板内部寄存器中。

二．上传目标板中的寄存器设置

如果你的目标板的寄存器已经被正确的配置或初始化过，你可以将这些配置上传并保存在主机的文件里。这个功能在你有一块寄存器已被设置好的板卡是非常有用，因为你可能要用这些寄存器设置去初始化另外一块板卡，上传出的寄存器文件可以被下载到一块新的板卡中。

从目标板上传寄存器设置，有以下步骤：

1. 选择 Window>Show View>Registers。出现寄存器窗口



2. 点击 Register 窗口的 Menu 按钮，选择 OCD Operations>Save Target Register Values to File。你可以用编辑器打开它，或存储在文件里。

3. 点击 Save。指定文件名和存储路径，扩展名为*.reg。

还有一种在 OCD Command Shell 通过底层命令上传寄存器设置至仿真器存储区域，即 SCT COPY(注意，只有那些 enabled 的寄存器组才可以被上传)，下一步将仿真器里的寄存器设置复制到寄存器文件中，这个采用 PJ UPLOAD 命令。

三．手工创建或修改寄存器文件

如果 Wind River 没有提供你自己板卡的寄存器文件，你们需要手工修改一个已有的寄存器文件。注意，寄存器文件只设置仿真器里存储的寄存器值，只有执行 IN 命令后，才会把仿真器里的设置写入目标板。如果开始就没有寄存器文件，仿真器里存储的寄存器值是用于 Wind River 参考板的，大多数情况下并不适合你自己的板卡，此时的 IN 命令不会正确的设置你的目标板的寄存器。如果你要下载程序代码并运行，寄存器文件至少要初始化 memroy 以便能够下载程序。初始化 memory 需要查看 CPU 和 memory 器件的硬件手册。

通常情况下，一个寄存器文件有以下结构：

```
REM ******************************************
REM CF CONFIGURATION
REM ******************************************
CF TAR                  8548E
CF SB                   SB
CF MMU                    DISABLE
CF BL                   DISABLE
CF RST                   YES
CF CLK                  16
CF LENDIAN                 NO
CF DLD                  NORMAL
CF HRESET                  ENABLE
CF TRESET                  ACTIVE
CF RESET                HRESET
CF CHECKSTOP               NO
CF SPOWER                  YES
CF PONR                 0
CF TRPEXP                  YES
CF INCOLD                  NO
……..
REM ******************************************
REM SC CONFIGURATION
REM ******************************************
SC GRP ERASE
SCGA LA         CCSRBAR             00000000  000E0000 LA          /va_dr /ua:0
/sa:0 /ue:0
SCGA LA         ALTCBAR             00000008  00000000 LA          /ua:0
```

```
SCGA LA          ALTCSR              00000010  00000000 LA          /ua:0
SCGA LA          BPTR                00000020  00000000 LA          /ua:0
SCGA LA          LAWBAR0             00000C08  00000000 LA          /ua:0
SCGA LA          LAWAR0              00000C10  00000000 LA          /ua:0
REM DDR2 Access Window
SCGA LA          LAWBAR1             00000C28  00000000 LA          /ua:0
SCGA LA          LAWAR1              00000C30  80F0001B LA          /ua:0
REM PCI Access Window
SCGA LA          LAWBAR2             00000C48  00080000 LA          /ua:0
SCGA LA          LAWAR2              00000C50  8000001C LA          /ua:0
……..
REM ****************************************
REM CF GROUP CONFIGURATION
REM ****************************************
CF GRP           LA ENABLED
CF GRP           ECM DISABLED
CF GRP           DDRMC ENABLED
CF GRP           I2C DISABLED
CF GRP           DUART DISABLED
CF GRP           LBC ENABLED
CF GRP           TA ENABLED
CF GRP           PCI DISABLED
CF GRP           PEX DISABLED
CF GRP           L2SRAM ENABLED
CF GRP           DMA DISABLED
CF GRP           ETSEC1 DISABLED
CF GRP           ETSEC2 DISABLED
CF GRP           PIC DISABLED
CF GRP           SRIO DISABLED
CF GRP           GLOBU ENABLED
CF GRP           PM DISABLED
CF GRP           WP DISABLED
CF GRP           WRLSDRAM ENABLED
CF GRP           CUSTOM ENABLED
REM ****************************************
REM TF CONFIGURATION
REM v28F640Jx (8192x8) 1 device
REM ****************************************
TF CONF 227 00005000 59812 FF800000
REM ****************************************
```

```
REM MMU CONFIGURATION
REM ****************************************
MMUD ALL
REM ****************************************
REM BL CONFIGURATION
REM ****************************************
BL DELETE
REM ****************************************
REM MMUOS CONFIGURATION
REM ****************************************
MMUOS DELETE
```

其中红色部分是主体，蓝色部分可选。开始部分的 CF 选项是针对仿真器的配置；接着是寄存器的设置，也是寄存器文件的主要部分；再下来是寄存器组使能与否的设定，只有 Enabled 的寄存器组的寄存器值才会被写入到目标板中。蓝色部分是对板上 Flash 型号，地址，烧写算法的地址和空间进行设定，以及如果要调试 Linux，需要对 MMU 配置，并用 BL 进行参数传递，但这些在调试非 Linux 系统时都是可选的。

设置寄存器值使用 SCGA 命令，注意这实际是一个创建寄存器组和设置寄存器值的命令，而 SC 命令是对一个已有的寄存器进行设置。

SCGA *寄存器组名 寄存器名 地址 数据 选项*

例如 SCGA SIM_MMU SIM_IBATOL 4014 00000004 /cpur

其中 SIM_MMU 为寄存器组名，可以在 CPU 手册中查到；SIM_IBATOL 为这个寄存器组中的 IBATOL 寄存器；4014 为 IBATOL 寄存器的偏移地址；00000004 为写入这个寄存器的值；/cpur 是 SCGA 选项，共有多种选项，可多选，详细请查看 OCD Command 手册。注意：在文件中有一条”SC GRP ERASE”，这是为了删除仿真器里的寄存器设置，以便建立新的数据表。

大多数情况下，用户并不愿意从头创建一个寄存器文件，那么可以参考一个同 CPU 型号的寄存器文件进行修改。Wind River 提供了大量的寄存器文件作为参考，位于 *installDir*/workbench-3.x/ocd/*build*/RegisterFiles 目录中，按照处理器体系架构、参考板生产厂家进行分类。编写或修改好目标板的寄存器文件后，不仅能够下载应用程序代码，还可以进行设置用于启动操作系统 VxWorks 或 Linux，此时就不需要 ROM 中的 Bootloader (VxWorks Bootrom；Linux U-boot)来初始化了。