Debug Uboot and Kernel on ColdFire V4/V4e Platform

Version: 1.1

Revision	Date	Author	Reason for Changes
1.0	2010/07/16	Lanttor.Guo	Draft
1.1	2010/08/02	Lanttor.Guo	Update u-boot debug

This document described how to debug u-boot and Linux kernel with CodeWarrior on Coldfire V4/V4e core based boards. This document takes the mcf54451evb board as the example.

Preparation for DEBUG

1. Install CodeWarrior Linux version on PC host. The latest version is recommended. The name of its installation package name is "CW_MCU_v10.0_Linux_SELH.tar". Before installing it, please read its README.txt and Release_Notes.txt.

2. get proper CW MCU v10.0 license file and copy it to *codewarrior install directory/MCU* sub-directory.

3. Install BSP for ColdFire platform you need on PC host. Before installing it, please read its user manual or getting start guide. Use Linux Target Image Builder tool (LTIB) to build BSP.

Debugging u-boot

1. Get the u-boot source from BSP with following commands:

cd bsp_install_path

./ltib –m prep –p u-boot

The u-boot source code will be stored at bsp_install_path/rpm/BUILD directory.

2. Modify and build u-boot source

go to u-boot source code directory

*modify lib_m68k/board.c: change the line **debug** to **printf** on the line "Now running in RAM ..."*

go to bsp installation directory

rebuild u-boot:

./ltib -m scbuild -p u-boot

In the u-boot source directory you will have a u-boot raw binary file (u-boot.bin) and u-boot ELF format executable binary file (u-boot).

3. Create CodeWarrior Project for u-boot

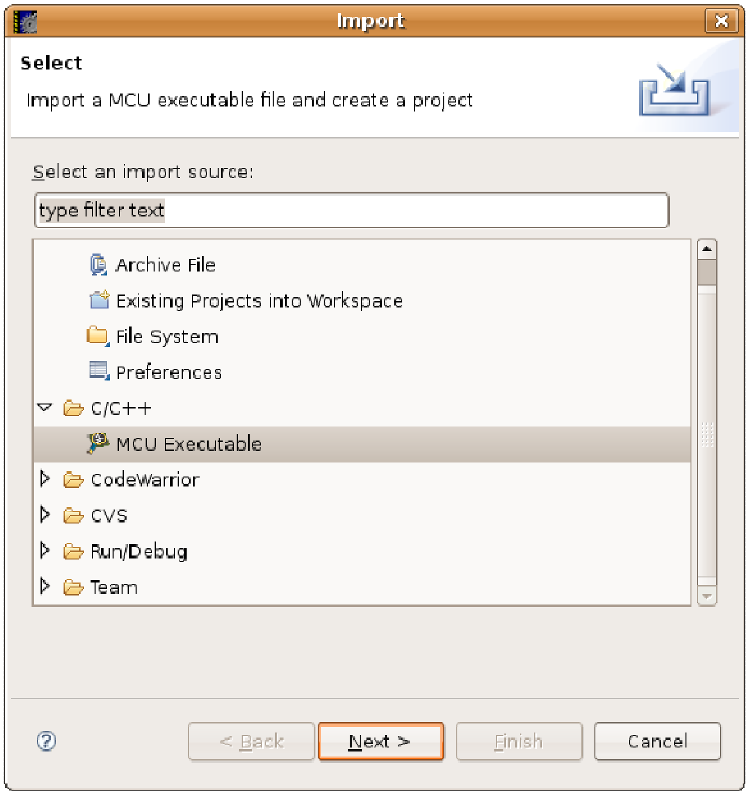
1) start the CodeWarrior IDE

cd /usr/local/Freescale/CodeWarrior_MCU_10.0/eclipse/

./cwide &

*2) select **File->Import***

The import wizard appears. Please refer to Figure-1.



Select

Import a MCU executable file and create a project

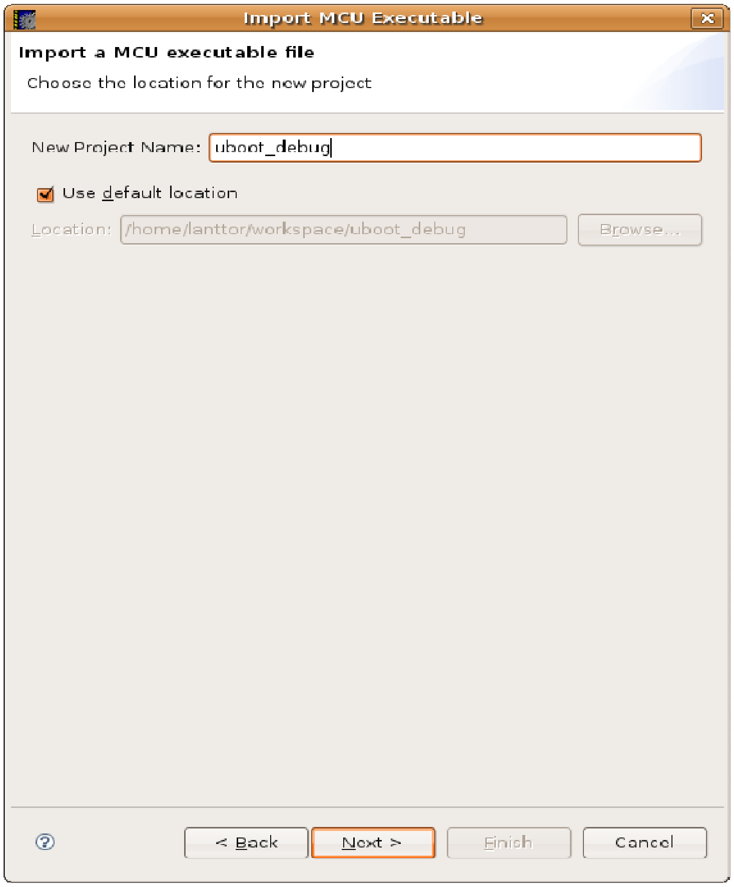
Select an import source:

type filter text

Figure-1 Import Wizard

*3) Expand the **C/C++ group** and select **MCU Executable**. Click Next.*

4) Specify a name in the New Project Name text box. Click Next. Please refer to Figure-2.



Import a MCU executable file

Choose the location for the new project

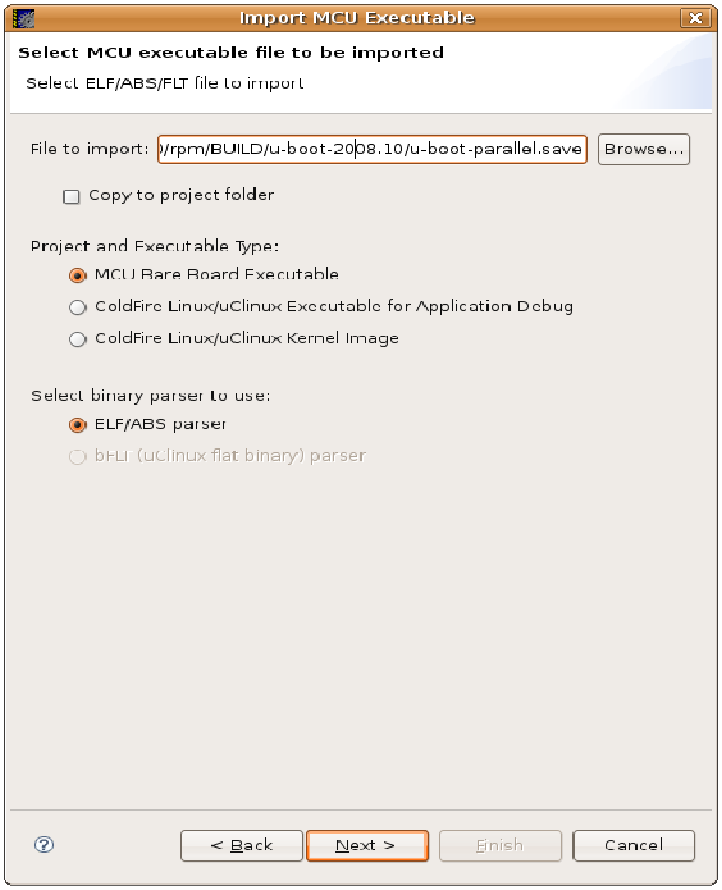
New Project Name: uboot_debug

☑ Use default location

Location: /home/lanttor/workspace/uboot_debug

Figure-2 Import MCU Executable

5) In the File to Import text box, specify the path of u-boot ELF file from u-boot source directory. Please refer to Figure-3. Click Next.



Import MCU Executable

Select MCU executable file to be imported

Select ELF/ABS/FLT file to import

File to import: /rpm/BUILD/u-boot-2008.10/u-boot-parallel.save

□ Copy to project folder

Project and Executable Type:

● MCU Bare Board Executable
○ ColdFire Linux/uClinux Executable for Application Debug
○ ColdFire Linux/uClinux Kernel Image

Select binary parser to use:

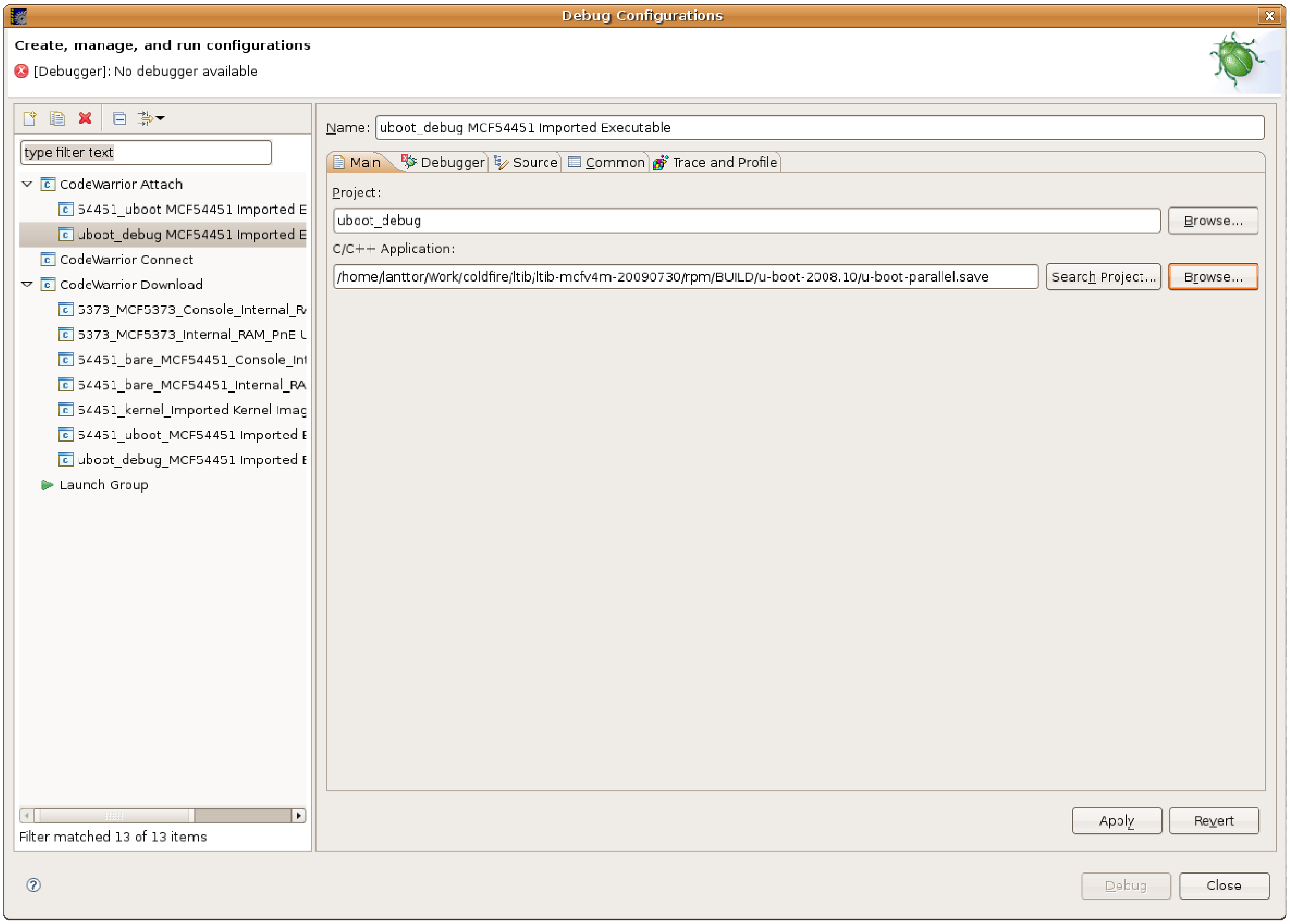
● ELF/ABS parser
○ bFLT (uClinux flat binary) parser

Figure-3 Select MCU executable file

*6) Select **Device**. Click Next.*

*7) Select Connection debug tool: choose **P&E USB BDM Multlink** for ColdFire platform. Click Finish.*

*8) Right click on the project and select **Debug As->Debug Configuration...**, The debug configuration dialog appears. Double **Click CodeWarrior Attach** to create new attach configuration for project. Please refer to Figure-4.*


Debug Configurations

Create, manage, and run configurations

[Debugger]: No debugger available

Name: uboot_debug MCF54451 Imported Executable

Project: uboot_debug

C/C++ Application: /home/lanttor/Work/coldfire/ltib/ltib-mcfv4m-20090730/rpm/BUILD/u-boot-2008.10/u-boot-parallel.save

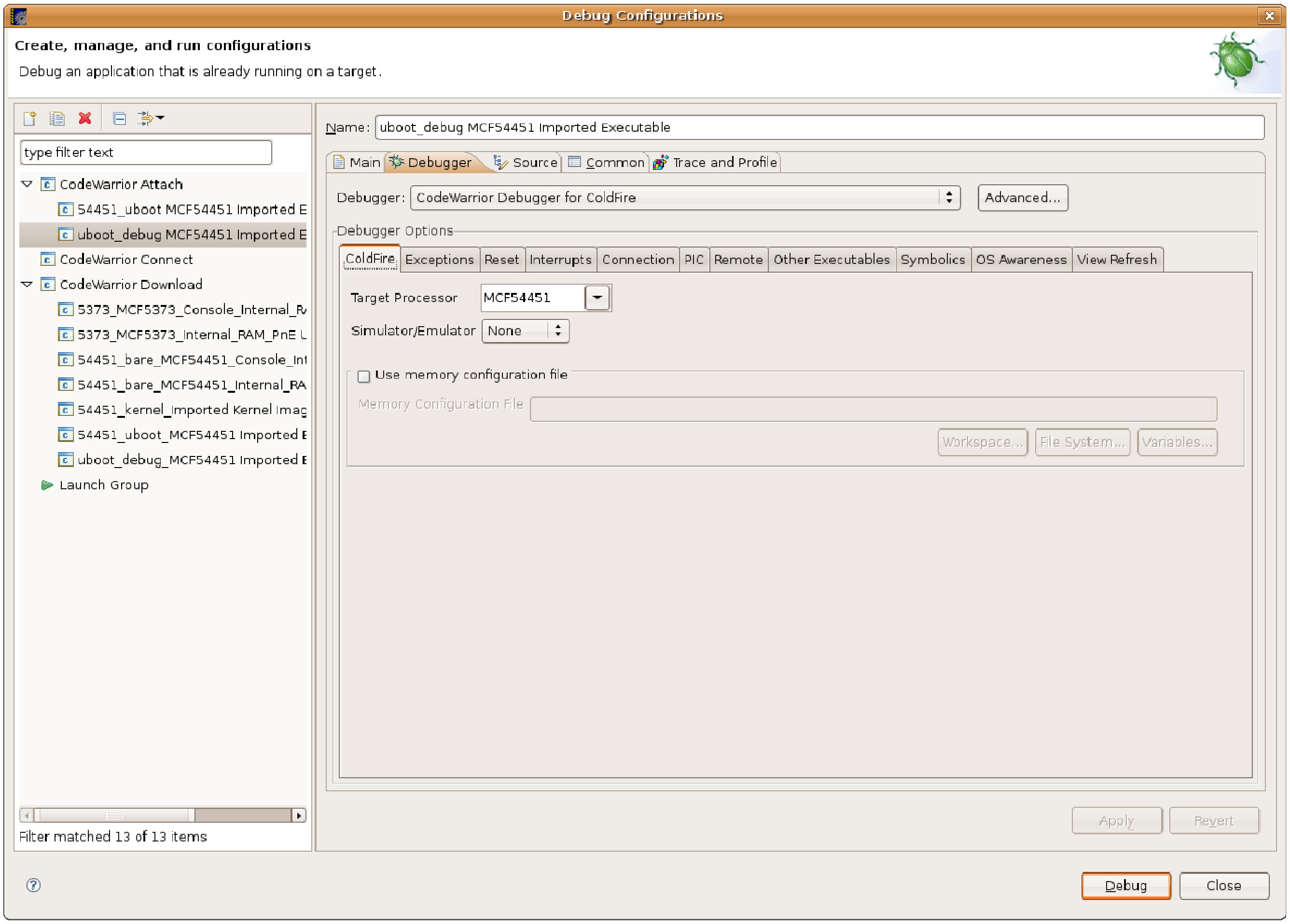
Filter matched 13 of 13 items

type filter text
CodeWarrior Attach
54451_uboot MCF54451 Imported E
uboot_debug MCF54451 Imported E
CodeWarrior Connect
CodeWarrior Download
5373_MCF5373_Console_Internal_R
5373_MCF5373_Internal_RAM_PnE L
54451_bare_MCF54451_Console_Int
54451_bare_MCF54451_Internal_RA
54451_kernel_Imported Kernel Imag
54451_uboot_MCF54451 Imported E
uboot_debug_MCF54451 Imported E
Launch Group
Main
Debugger
Source
Common
Trace and Profile
Browse...
Search Project...
Browse...
Apply
Revert
Debug
Close

Figure-4 Debug configureation

*9) At **C/C++ Application** text box, click Browse button and choose u-boot ELF format file you need debug.*

*10) Click **Debugger** table. Choose Debugger "CodeWarrior Debugger for ColdFire". Choose proper target processor. Please refer to Figure-5.*



Debug Configurations

Create, manage, and run configurations

Debug an application that is already running on a target.

type filter text

- CodeWarrior Attach
  - 54451_uboot MCF54451 Imported E
  - uboot_debug MCF54451 Imported E
- CodeWarrior Connect
- CodeWarrior Download
  - 5373_MCF5373_Console_Internal_R/
  - 5373_MCF5373_Internal_RAM_PnE L
  - 54451_bare_MCF54451_Console_Int
  - 54451_bare_MCF54451_Internal_RA
  - 54451_kernel_Imported Kernel Imag
  - 54451_uboot_MCF54451 Imported E
  - uboot_debug_MCF54451 Imported E
- Launch Group

Filter matched 13 of 13 items

Name: uboot_debug MCF54451 Imported Executable

Debugger: CodeWarrior Debugger for ColdFire

Debugger Options

Target Processor: MCF54451

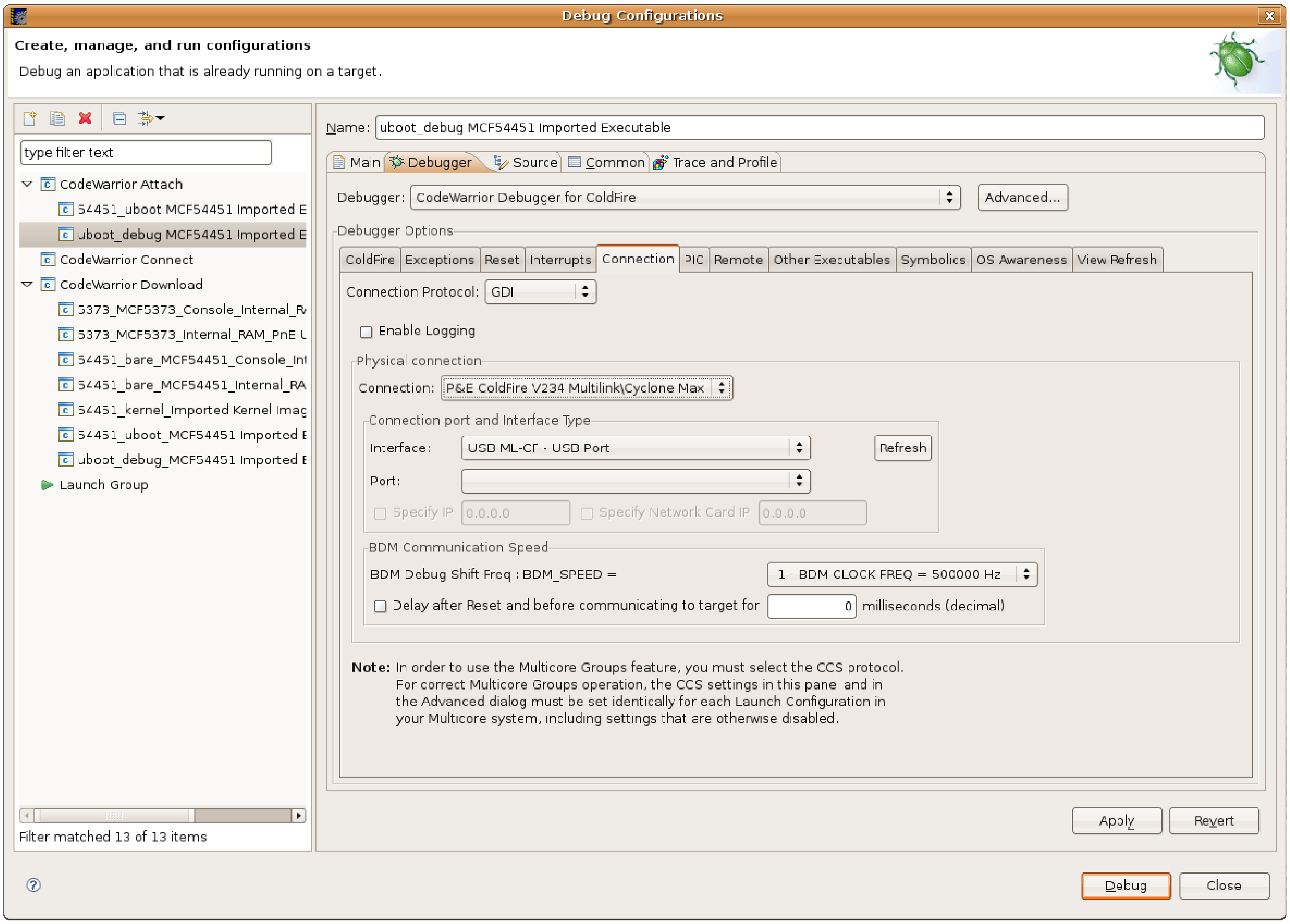
Simulator/Emulator: None

□ Use memory configuration file

Memory Configuration File

Figure-5 Debug configuration cont.

12) Click Debugger page ***Connection*** *table, choose connection protocol: GDI, choose physical connection:P&E ColdFire V234 Multilink. Please refer to Figure-6.*



Debug Configurations

Create, manage, and run configurations

Debug an application that is already running on a target.

type filter text

- CodeWarrior Attach
  - 54451_uboot MCF54451 Imported E
  - uboot_debug MCF54451 Imported E
- CodeWarrior Connect
- CodeWarrior Download
  - 5373_MCF5373_Console_Internal_R
  - 5373_MCF5373_Internal_RAM_PnE L
  - 54451_bare_MCF54451_Console_Int
  - 54451_bare_MCF54451_Internal_RA
  - 54451_kernel_Imported Kernel Imag
  - 54451_uboot_MCF54451 Imported E
  - uboot_debug_MCF54451 Imported E
- Launch Group

Filter matched 13 of 13 items

Name: uboot_debug MCF54451 Imported Executable

Main | Debugger | Source | Common | Trace and Profile

Debugger: CodeWarrior Debugger for ColdFire   Advanced...

Debugger Options

ColdFire | Exceptions | Reset | Interrupts | Connection | PIC | Remote | Other Executables | Symbolics | OS Awareness | View Refresh

Connection Protocol: GDI

□ Enable Logging

Physical connection

Connection: P&E ColdFire V234 Multilink\Cyclone Max

Connection port and Interface Type

Interface: USB ML-CF - USB Port   Refresh

Port:

□ Specify IP 0.0.0.0   □ Specify Network Card IP 0.0.0.0

BDM Communication Speed

BDM Debug Shift Freq : BDM_SPEED = 1 - BDM CLOCK FREQ = 500000 Hz

□ Delay after Reset and before communicating to target for 0 milliseconds (decimal)

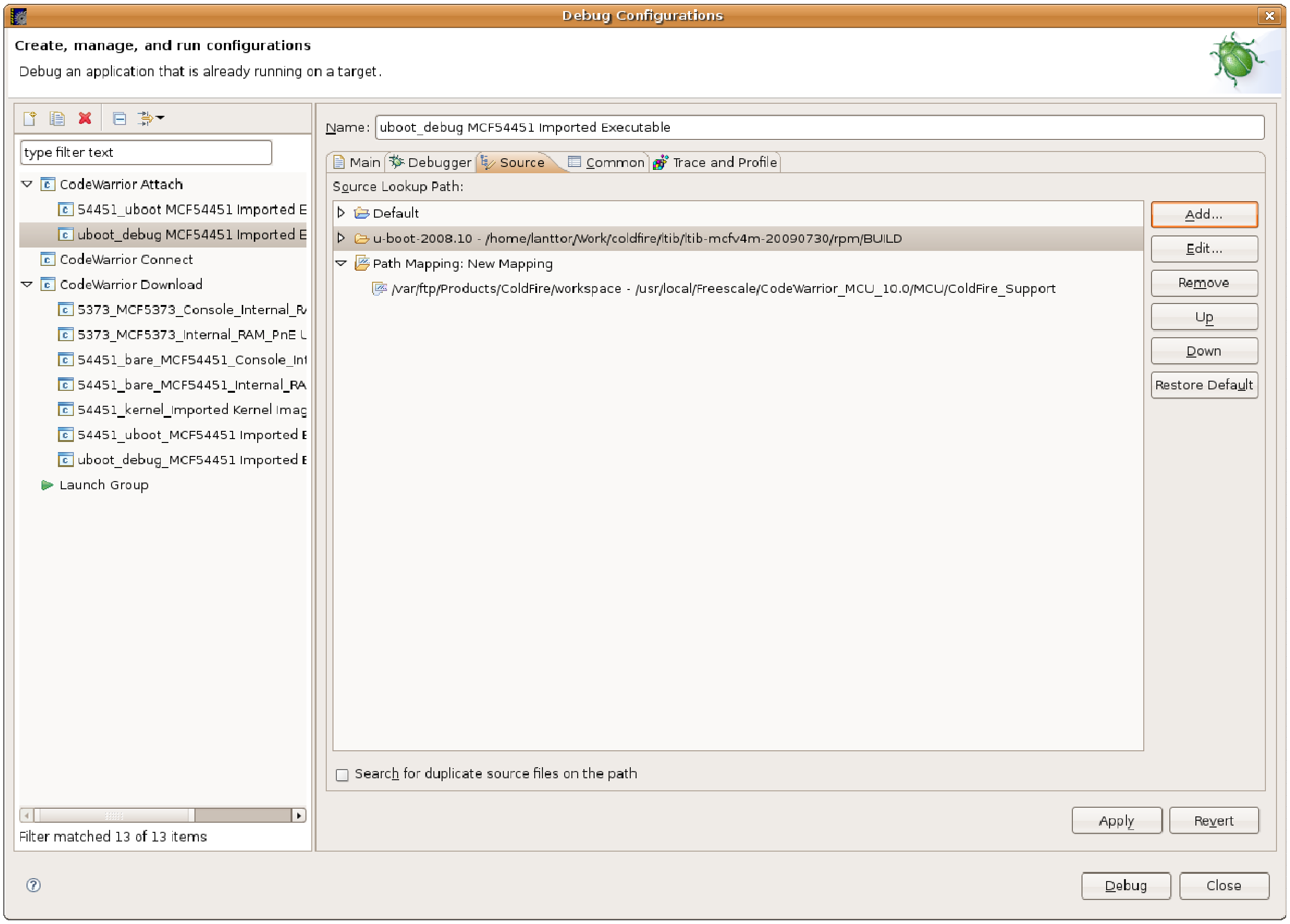
Note: In order to use the Multicore Groups feature, you must select the CCS protocol. For correct Multicore Groups operation, the CCS settings in this panel and in the Advanced dialog must be set identically for each Launch Configuration in your Multicore system, including settings that are otherwise disabled.

Apply   Revert

Debug   Close

Figure-6 Debug configureation cont.

*14) Click **Source** table, Add source path. Please refer to Figure-7.*



Debug Configurations

Create, manage, and run configurations

Debug an application that is already running on a target.

Name: uboot_debug MCF54451 Imported Executable

Source Lookup Path:

- Default
- u-boot-2008.10 - /home/lanttor/Work/coldfire/ltib/ltib-mcfv4m-20090730/rpm/BUILD
- Path Mapping: New Mapping
  - /var/ftp/Products/ColdFire/workspace - /usr/local/Freescale/CodeWarrior_MCU_10.0/MCU/ColdFire_Support

□ Search for duplicate source files on the path

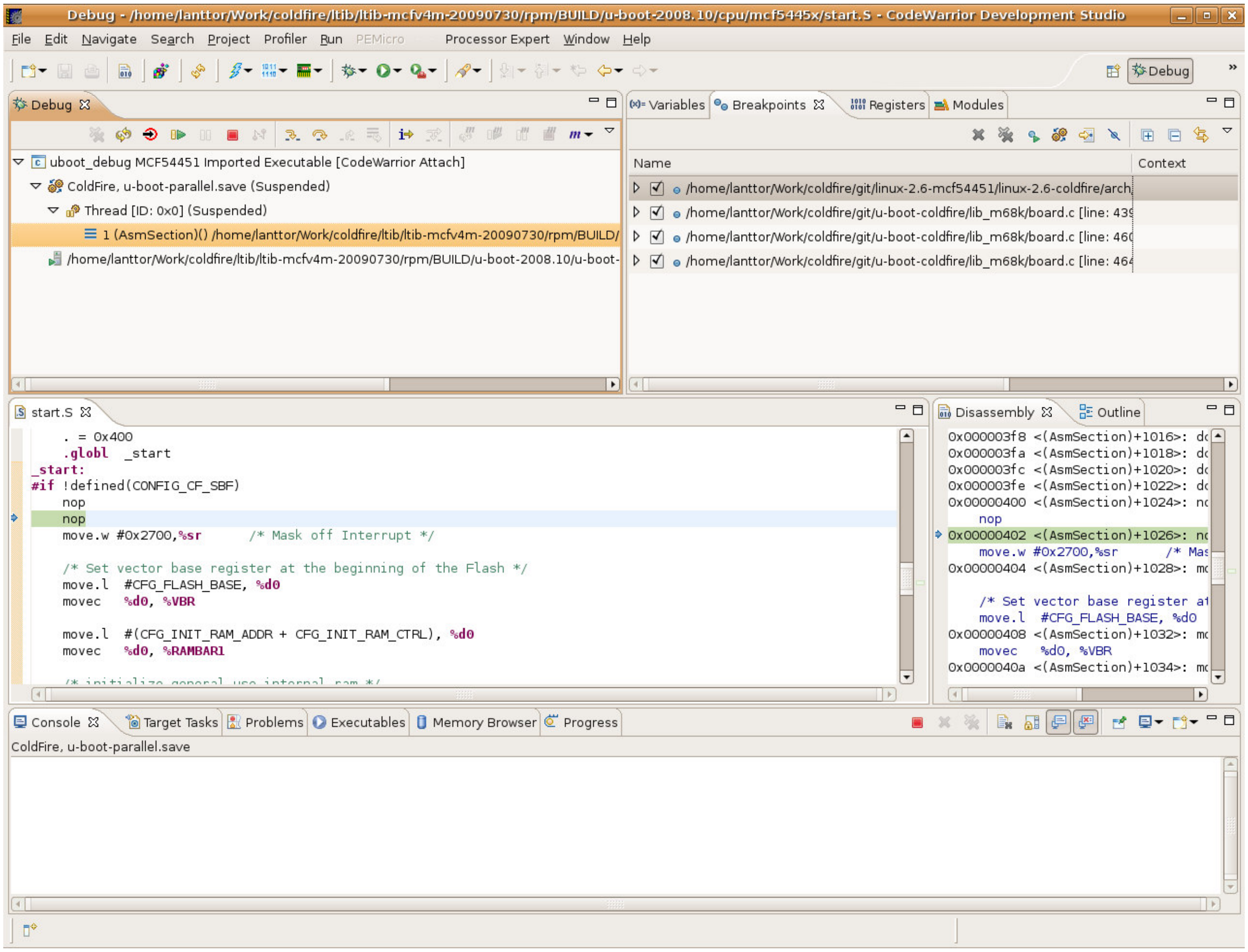
- CodeWarrior Attach
  - 54451_uboot MCF54451 Imported E
  - uboot_debug MCF54451 Imported E
- CodeWarrior Connect
- CodeWarrior Download
  - 5373_MCF5373_Console_Internal_R/
  - 5373_MCF5373_Internal_RAM_PnE L
  - 54451_bare_MCF54451_Console_Int
  - 54451_bare_MCF54451_Internal_RA
  - 54451_kernel_Imported Kernel Imag
  - 54451_uboot_MCF54451 Imported E
  - uboot_debug_MCF54451 Imported E
- Launch Group

Filter matched 13 of 13 items

Figure-7 Debug configuration cont.

15) All the settings have been prepared now. Check board connection between P&E debug interface and BDM interface is correct. Power on board and click debug button.

16) This debugs the u-boot. Please refer to Figure-8.



Debug - /home/lanttor/Work/coldfire/ltib/ltib-mcfv4m-20090730/rpm/BUILD/u-boot-2008.10/cpu/mcf5445x/start.S - CodeWarrior Development Studio

```
uboot_debug MCF54451 Imported Executable [CodeWarrior Attach]
  ColdFire, u-boot-parallel.save (Suspended)
    Thread [ID: 0x0] (Suspended)
      1 (AsmSection)() /home/lanttor/Work/coldfire/ltib/ltib-mcfv4m-20090730/rpm/BUILD/
  /home/lanttor/Work/coldfire/ltib/ltib-mcfv4m-20090730/rpm/BUILD/u-boot-2008.10/u-boot-
```

Name	Context
☑ /home/lanttor/Work/coldfire/git/linux-2.6-mcf54451/linux-2.6-coldfire/arch	
☑ /home/lanttor/Work/coldfire/git/u-boot-coldfire/lib_m68k/board.c [line: 439	
☑ /home/lanttor/Work/coldfire/git/u-boot-coldfire/lib_m68k/board.c [line: 460	
☑ /home/lanttor/Work/coldfire/git/u-boot-coldfire/lib_m68k/board.c [line: 464	

```
	. = 0x400
	.globl	_start
_start:
#if !defined(CONFIG_CF_SBF)
	nop
	nop
	move.w #0x2700,%sr	/* Mask off Interrupt */

	/* Set vector base register at the beginning of the Flash */
	move.l	#CFG_FLASH_BASE, %d0
	movec	%d0, %VBR

	move.l	#(CFG_INIT_RAM_ADDR + CFG_INIT_RAM_CTRL), %d0
	movec	%d0, %RAMBAR1

	/* initialize general use internal ram */
```

```
0x000003f8 <(AsmSection)+1016>: dc
0x000003fa <(AsmSection)+1018>: dc
0x000003fc <(AsmSection)+1020>: dc
0x000003fe <(AsmSection)+1022>: dc
0x00000400 <(AsmSection)+1024>: nc
    nop
0x00000402 <(AsmSection)+1026>: nc
    move.w #0x2700,%sr      /* Mas
0x00000404 <(AsmSection)+1028>: mc

    /* Set vector base register at
    move.l  #CFG_FLASH_BASE, %d0
0x00000408 <(AsmSection)+1032>: mc
    movec   %d0, %VBR
0x0000040a <(AsmSection)+1034>: mc
```

ColdFire, u-boot-parallel.save

Figure-8 Debug View

During a typical u-boot startup sequence, the target processor starts executing u-boot in Flash Memory. U-boot then enables the Memory Management Unit (MMU) and relocates itself to RAM.

CodeWarrior build target settings required to debug u-boot in flash memory differ from the settings required to debug u-boot in RAM. Therefore, you must use individual CodeWarrior debugs session to debug the Flash Memory and RAM sections:

17) Debugging u-boot in Flash

*Enable **Debugger PIC settings-> Alternate Load Address**:<flash base address>.*

Start u-boot on the board.

Create an attach configuration with settings in Figure-6.

Start debug.

Stop u-boot.

*In the debug view, click **Reset** to reset the board.*

You are now at the __start section.

18) Debugging u-boot in RAM

*Enable Debugger **PIC settings-> Alternate Load Address**:<flash base address>. Note: The second stage starts when you reach relocate_code() in lib_m68k/board.c. For this stage, you switch to RAM and do not execute code in Flash. In order to have source correspondence, to see not only assembly, PIC settings-> Alternate Address must be enabled with the value u-*

boot prints: “Now running in RAM – U-Boot at <some address>” from which you have to subtract 0x400.

Start debug.

Stop u-boot.

*In the debug view, click **Reset** to reset the Debugger.*

Set a breakpoint at board_int_r().

*Select **Run**, the breakpoint is encountered.*

Click debug.

Tricks

If you follow above steps, you may meet this issue--- the breakpoint at RAM can't stop. If it's true, please use the trick:At Debugger menu, open Window->Show View->Debugger Shell, and run command “set picloaddr <Alternate Address>” in it. Then click Reset to reset the Debugger. It can work normally.

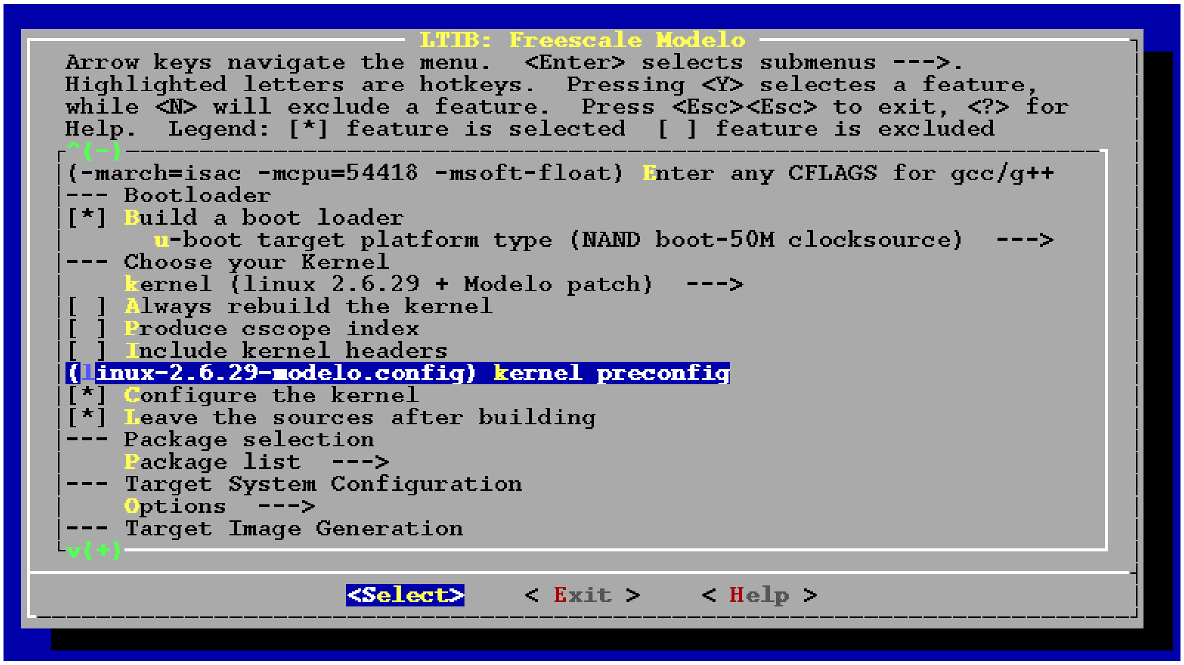
Debugging Linux Kernel

1. Get Linux kernel from following steps

cd bsp_install_path

./ltib -c

*choose **Configure the kernel** and **Leave the sources after building** options, save and exit ltib configure menu. Please refer to Figure-9.*



```
                     LTIB: Freescale Modelo
 Arrow keys navigate the menu.  <Enter> selects submenus --->.
 Highlighted letters are hotkeys.  Pressing <Y> selectes a feature,
 while <N> will exclude a feature.  Press <Esc><Esc> to exit, <?> for
 Help.  Legend: [*] feature is selected  [ ] feature is excluded
^(-)
 (-march=isac -mcpu=54418 -msoft-float) Enter any CFLAGS for gcc/g++
 --- Bootloader
 [*] Build a boot loader
       u-boot target platform type (NAND boot-50M clocksource)  --->
 --- Choose your Kernel
     kernel (linux 2.6.29 + Modelo patch)  --->
 [ ] Always rebuild the kernel
 [ ] Produce cscope index
 [ ] Include kernel headers
 (linux-2.6.29-modelo.config) kernel preconfig
 [*] Configure the kernel
 [*] Leave the sources after building
 --- Package selection
     Package list  --->
 --- Target System Configuration
     Options  --->
 --- Target Image Generation
v(+)

              <Select>    < Exit >    < Help >
```

Figure-9 ltib configuration for kernel

The kernel souce code will be stored in bsp_install_path/rpm/BUILD/linux directory.

2. Build kernel source code

go to kernel source code directory.

*When building the kernel, make sure select **kernel hacking-> Compile the kernel with debug info** option and **general setup-> configure standard kernel features->Load all symbols** for debugging/ksymoops option.*

go to bsp_install_path directory and run build commands:

./ltib -m scbuild -p kernel

./ltib -m scdeploy -p kernel

You will have kernel ELF format binary file – vmlinux and raw data binary file – uImage in the kernel source code directory.

3. Create CodeWarrior project for Linux Kernel

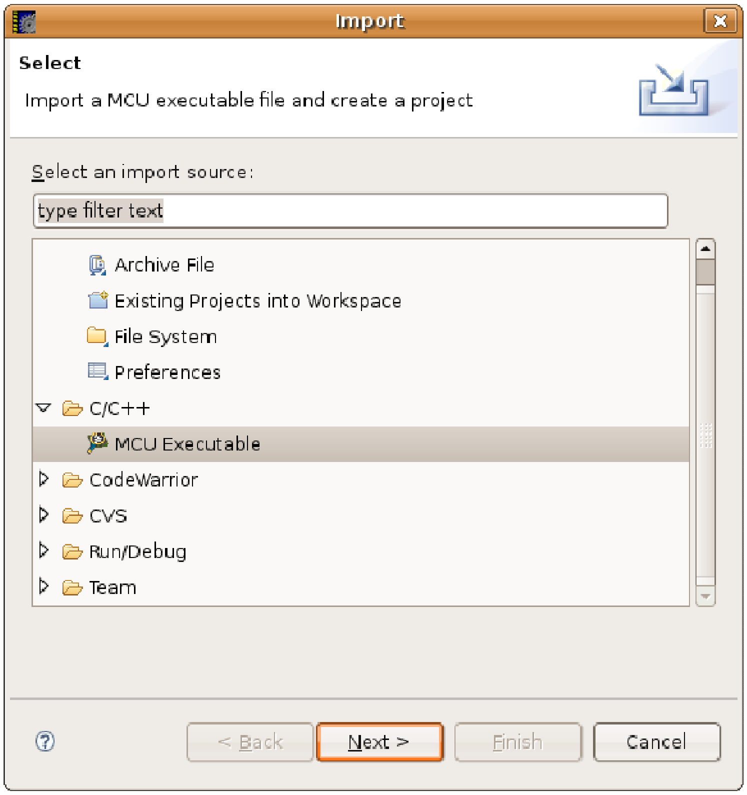
1) start the CodeWarrior IDE

cd /usr/local/Freescale/CodeWarrior_MCU_10.0/eclipse/

./cwide &

*2) select **File->Import***

The import wizard appears. Please refer to Figure-10.



Select

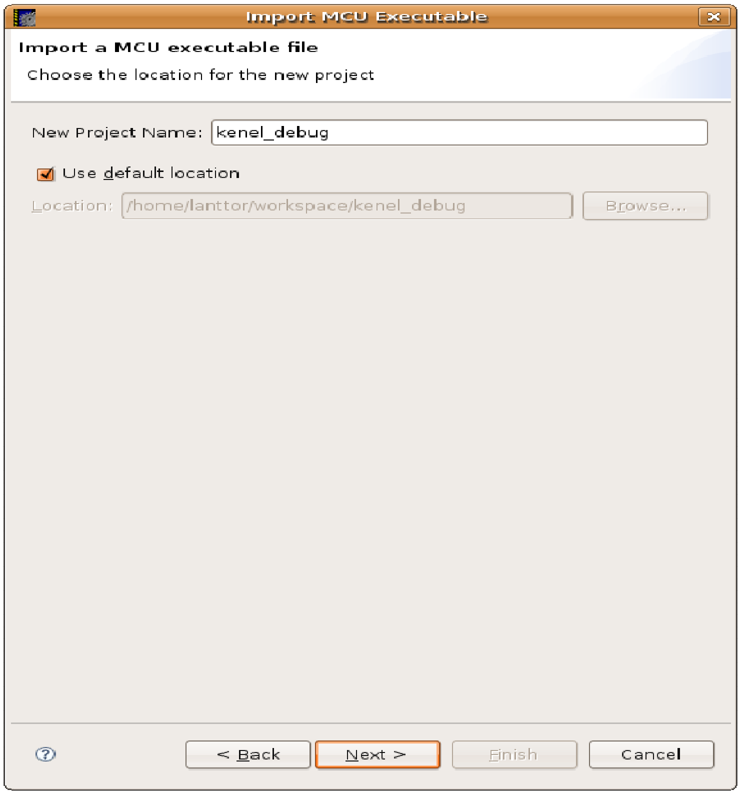
Import a MCU executable file and create a project

Select an import source:

Figure-10 Import Wizard

*3) Expand the **C/C++ group** and select **MCU Executable**. Click Next.*

4) Specify a name in the New Project Name text box. Click Next. Please refer to Figure-11.



Import a MCU executable file

Choose the location for the new project

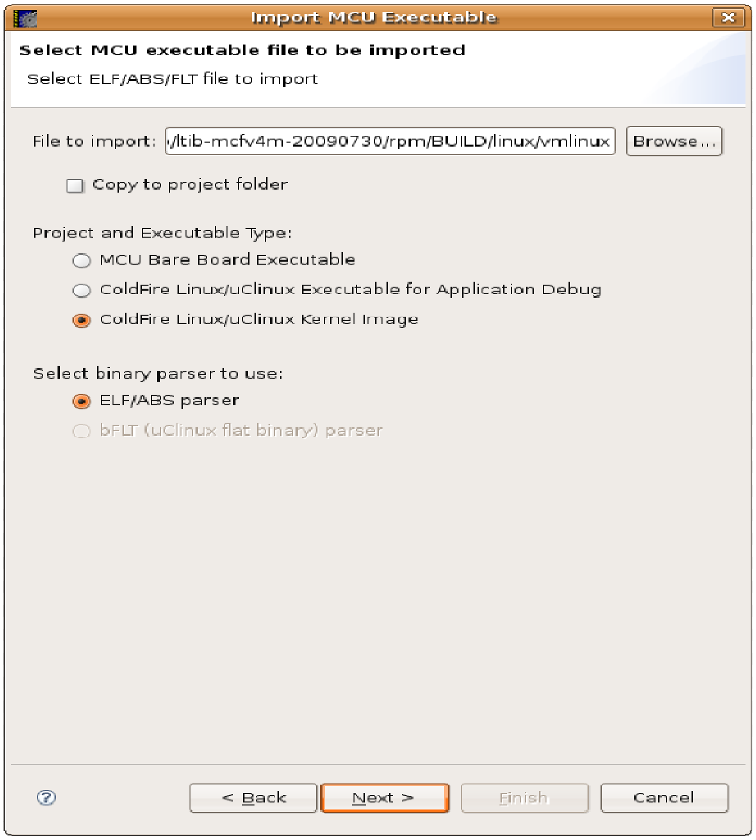
New Project Name: kenel_debug

☑ Use default location

Location: /home/lanttor/workspace/kenel_debug

Figure-11 Import MCU Executable

5) In the File to Import text box, specify the path of kernel ELF file from kernelsource directory. Please refer to Figure-12. Click Next.



Select MCU executable file to be imported

Select ELF/ABS/FLT file to import

File to import: ı/ltib-mcfv4m-20090730/rpm/BUILD/linux/vmlinux

□ Copy to project folder

Project and Executable Type:

○ MCU Bare Board Executable

○ ColdFire Linux/uClinux Executable for Application Debug

● ColdFire Linux/uClinux Kernel Image

Select binary parser to use:

● ELF/ABS parser

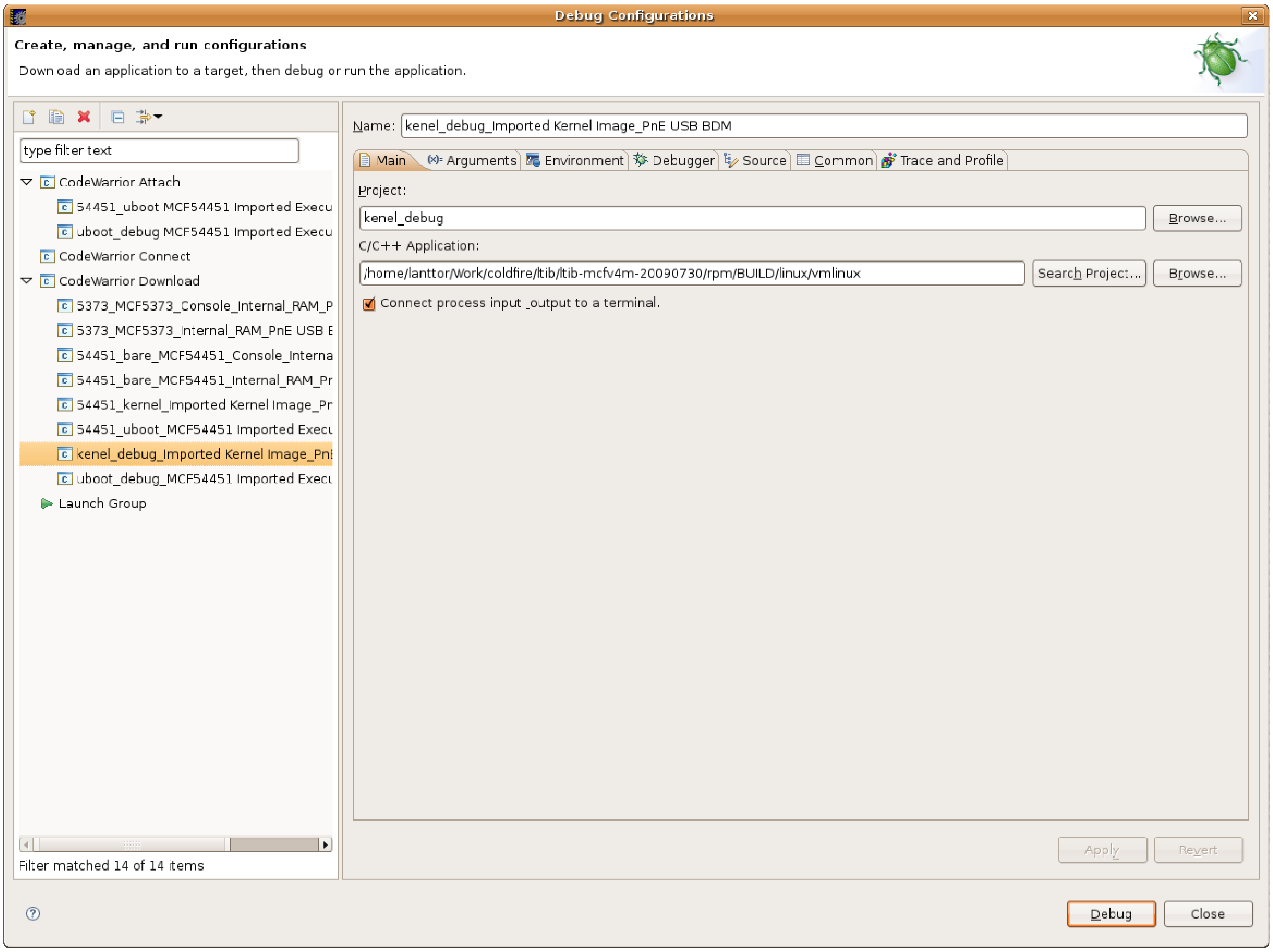
○ bFLT (uClinux flat binary) parser

Figure-12 Select MCU executable file

6) Select proper ***Device and Board****. Click Next.*

7) Select Connection debug tool: choose ***P&E USB BDM Multlink*** *for ColdFire platform. Click Finish.*

8) Right click on the project and select ***Debug As->Debug Configuration...****, The debug configuration dialog appears. Choose* ***kernel_debug_Imported Kernel Image_PnE USB BDM*** *from* ***CodeWarrior Download*** *menu. Please refer to Figure-13.*



Debug Configurations
Create, manage, and run configurations
Download an application to a target, then debug or run the application.
type filter text
CodeWarrior Attach
54451_uboot MCF54451 Imported Execu
uboot_debug MCF54451 Imported Execu
CodeWarrior Connect
CodeWarrior Download
5373_MCF5373_Console_Internal_RAM_P
5373_MCF5373_Internal_RAM_PnE USB E
54451_bare_MCF54451_Console_Interna
54451_bare_MCF54451_Internal_RAM_Pr
54451_kernel_Imported Kernel Image_Pr
54451_uboot_MCF54451 Imported Execu
kenel_debug_Imported Kernel Image_PnE
uboot_debug_MCF54451 Imported Execu
Launch Group
Filter matched 14 of 14 items
Main
Arguments
Environment
Debugger
Source
Common
Trace and Profile
Browse...
Search Project...
Browse...
Apply
Revert
Debug
Close

Name: kenel_debug_Imported Kernel Image_PnE USB BDM

Project: kenel_debug

C/C++ Application: /home/lanttor/Work/coldfire/ltib/ltib-mcfv4m-20090730/rpm/BUILD/linux/vmlinux

☑ Connect process input _output to a terminal.

Figure-13 Debug configurations

9) Click Debugger table.

Choose ***Program entry point****.*

Click ***Debugger Options-> ColdFire*** *table, choose proper target processor.*

Click ***Debugger Options-> connection*** *table, set* ***connect protocol****: GDI; set* ***physical connection****: P&E ColdFire V234 multlink.*

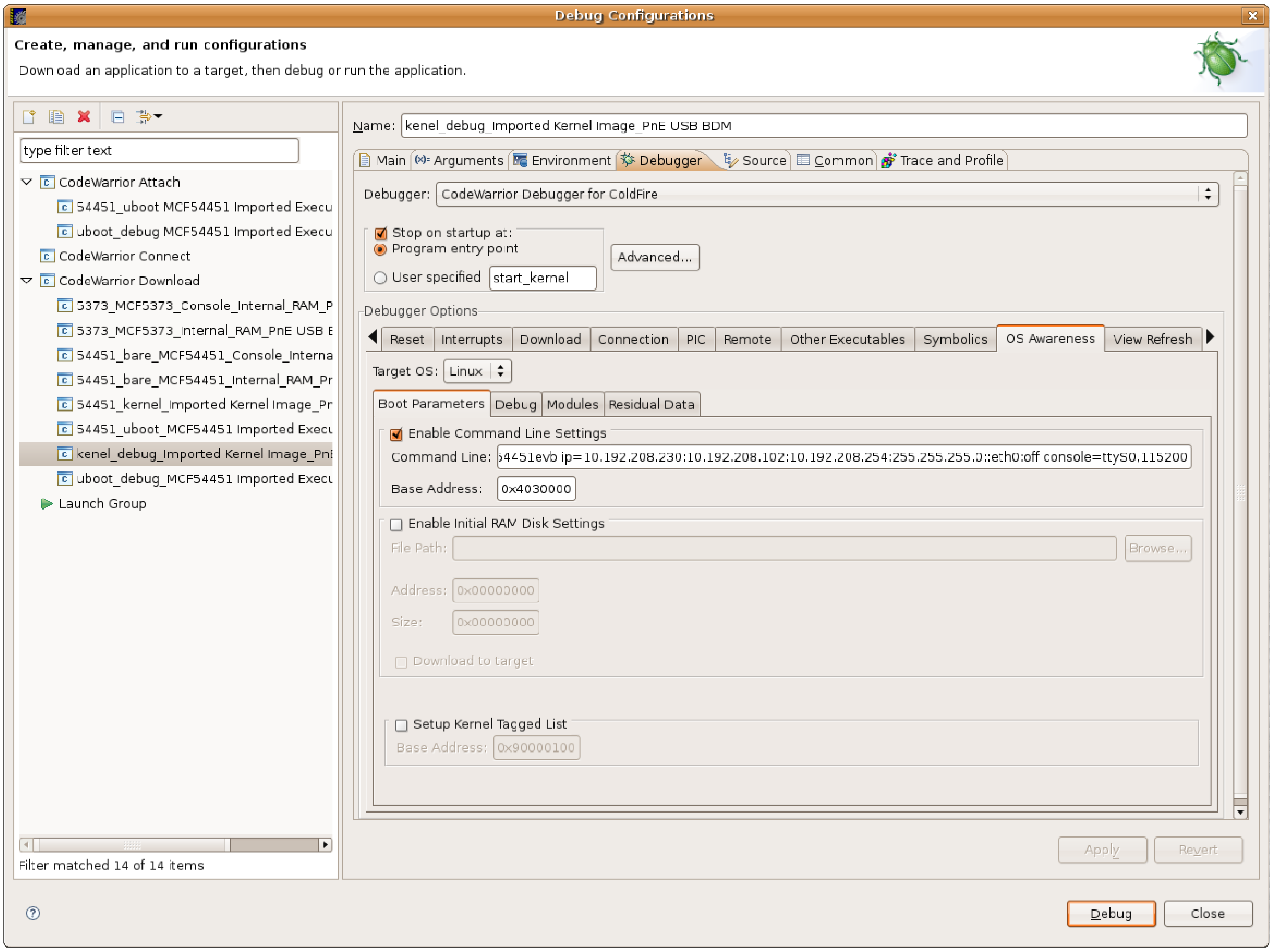
Click ***Debugger Options-> OS Awareness*** *table, set target OS: Linux.*

On ***OS Awareness*** *page, click* ***boot parameters*** *table: enable command line settings.*

command line: root=/dev/nfs rw nfsroot=10.192.208.112:/tftpboot/rootfs_mcf54451evb ip=10.192.208.230:10.192.208.102:10.192.208.254:255.255.255.0::eth0:off console=ttyS0,115200

base address:0x40300000 (The choice of base address needs a valid memory address which should be around the address of symbol '_end' and needs page align. For example, you could check the System.map under your Linux source directory to look up the address of the symbol '_end': 0x40327760, then you can specify the address 0x40400000 as the Base Address).

Please refer to Figure-14.



Debug Configurations

Create, manage, and run configurations

Download an application to a target, then debug or run the application.

type filter text
CodeWarrior Attach
54451_uboot MCF54451 Imported Execu
uboot_debug MCF54451 Imported Execu
CodeWarrior Connect
CodeWarrior Download
5373_MCF5373_Console_Internal_RAM_P
5373_MCF5373_Internal_RAM_PnE USB E
54451_bare_MCF54451_Console_Interna
54451_bare_MCF54451_Internal_RAM_Pr
54451_kernel_Imported Kernel Image_Pr
54451_uboot_MCF54451 Imported Execu
kenel_debug_Imported Kernel Image_PnE
uboot_debug_MCF54451 Imported Execu
Launch Group
Main
Arguments
Environment
Debugger
Source
Common
Trace and Profile
Advanced...
Reset
Interrupts
Download
Connection
PIC
Remote
Other Executables
Symbolics
OS Awareness
View Refresh
Boot Parameters
Debug
Modules
Residual Data
Browse...
Apply
Revert
Debug
Close

Filter matched 14 of 14 items

Name: kenel_debug_Imported Kernel Image_PnE USB BDM

Debugger: CodeWarrior Debugger for ColdFire

☑ Stop on startup at:

● Program entry point

○ User specified start_kernel

Debugger Options

Target OS: Linux

☑ Enable Command Line Settings

Command Line: 54451evb ip=10.192.208.230:10.192.208.102:10.192.208.254:255.255.255.0::eth0:off console=ttyS0,115200

Base Address: 0x4030000

☐ Enable Initial RAM Disk Settings

File Path:

Address: 0x00000000

Size: 0x00000000

☐ Download to target

☐ Setup Kernel Tagged List

Base Address: 0x90000100

Figure-14 Debug configurations cont.

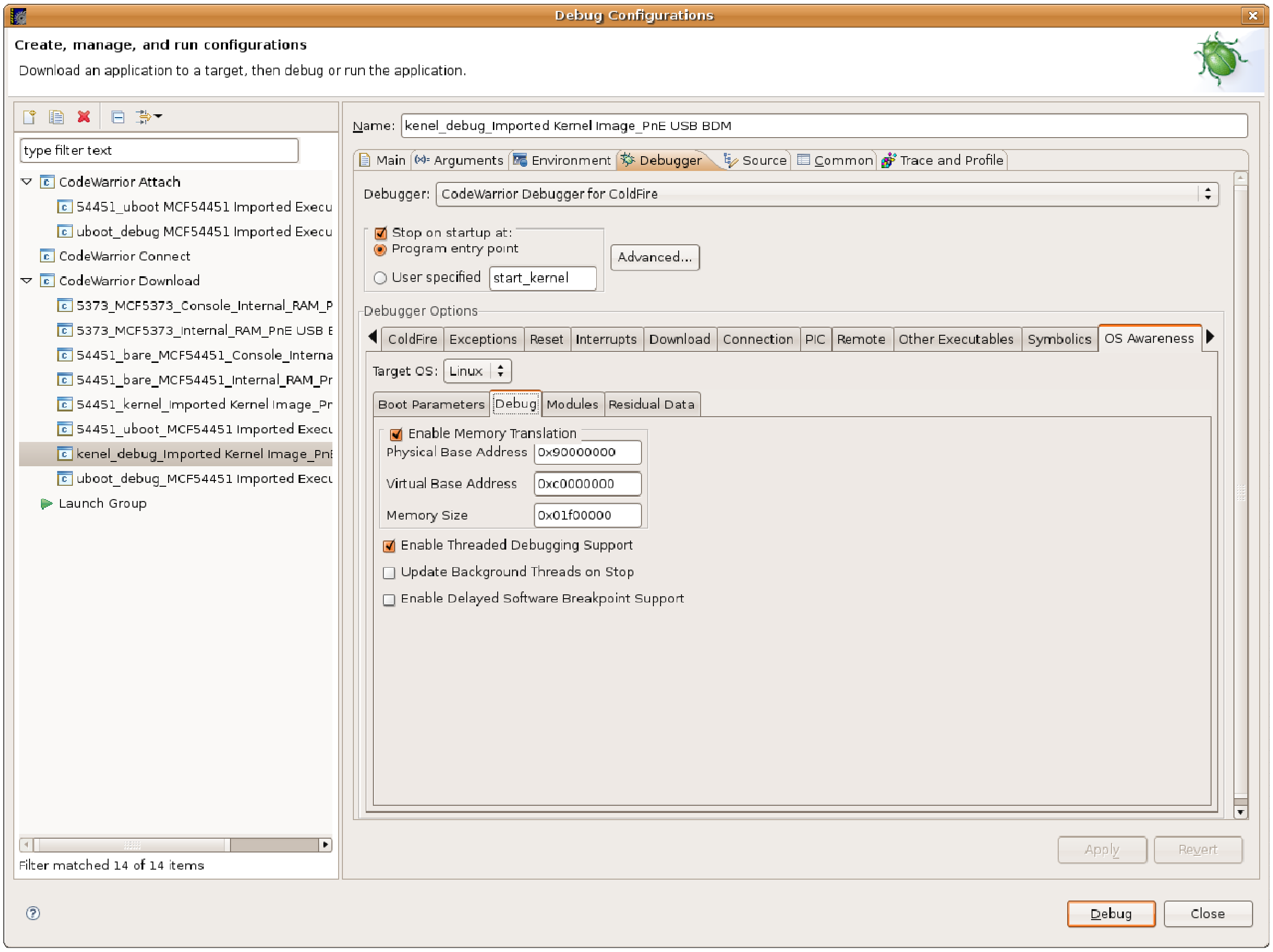
*On **OS Awareness** page, click **Debug** table, **Enable Memory Translation**.*

Physical base address: 0x40000000

Virtual base address: 0x40000000

Memory size: 0x08000000

Please refer to Figure-15.



Create, manage, and run configurations

Download an application to a target, then debug or run the application.

type filter text

- CodeWarrior Attach
  - 54451_uboot MCF54451 Imported Execu
  - uboot_debug MCF54451 Imported Execu
- CodeWarrior Connect
- CodeWarrior Download
  - 5373_MCF5373_Console_Internal_RAM_P
  - 5373_MCF5373_Internal_RAM_PnE USB E
  - 54451_bare_MCF54451_Console_Interna
  - 54451_bare_MCF54451_Internal_RAM_Pr
  - 54451_kernel_Imported Kernel Image_Pr
  - 54451_uboot_MCF54451 Imported Execu
  - kenel_debug_Imported Kernel Image_PnE
  - uboot_debug_MCF54451 Imported Execu
- Launch Group

Filter matched 14 of 14 items

Name: kenel_debug_Imported Kernel Image_PnE USB BDM

Debugger: CodeWarrior Debugger for ColdFire

☑ Stop on startup at:
● Program entry point
○ User specified start_kernel

Debugger Options

Target OS: Linux

☑ Enable Memory Translation

Physical Base Address 0x90000000

Virtual Base Address 0xc0000000

Memory Size 0x01f00000

☑ Enable Threaded Debugging Support

☐ Update Background Threads on Stop

☐ Enable Delayed Software Breakpoint Support

Figure-15 Debug configurations cont.

*On **OS Awareness** page, click **Residual Data** table, **Enable Residual Data settings**. Set proper residual data according to your board.*

***Base Type name**: bd_info*

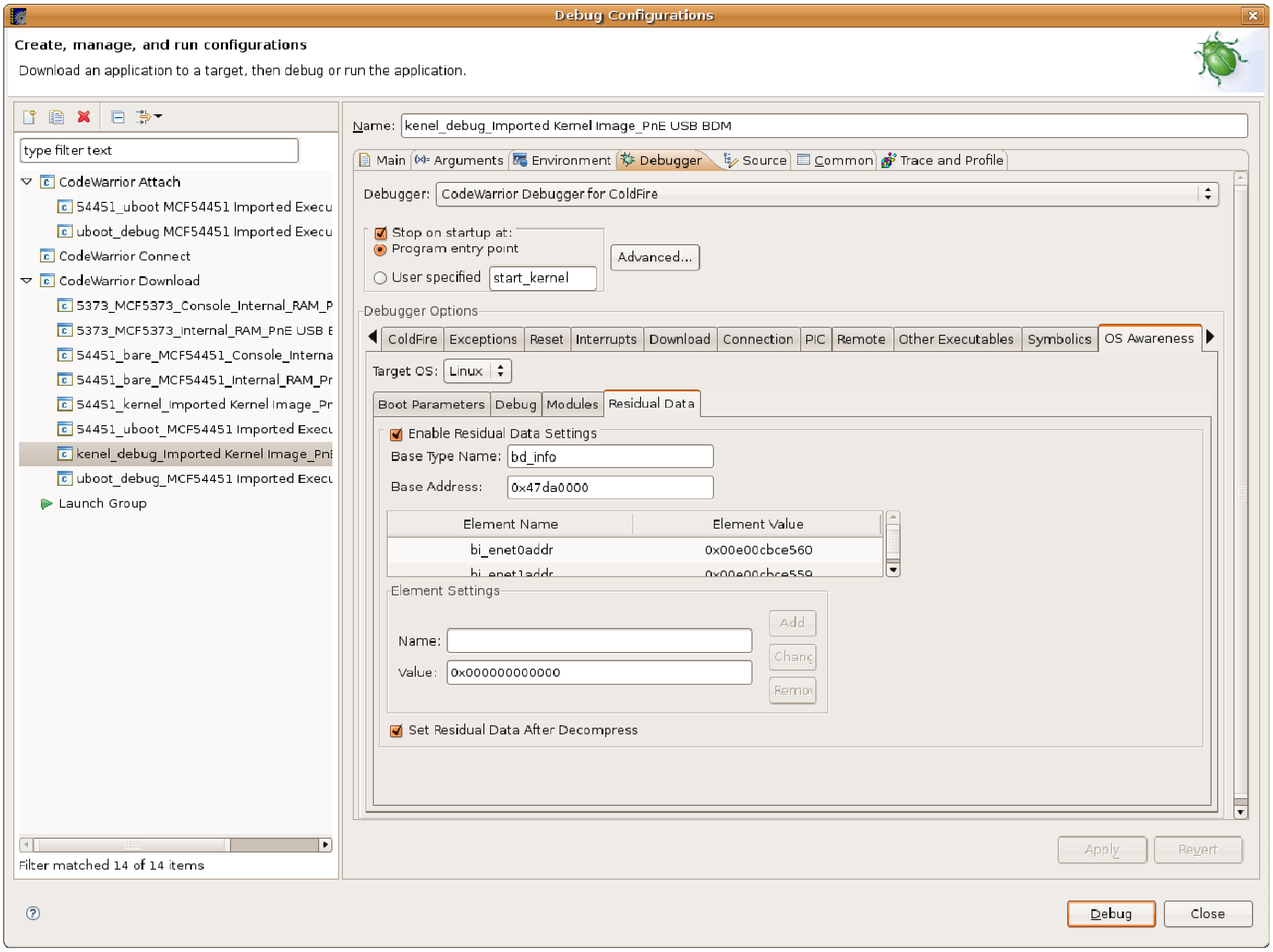
***Base Address**: 0x47da0000 (Note, the choice of “Base Address” needs a valid memory address around the end of the memory and needs page align. For example, here the address “0x47da0000” is to be chose as the Base Address.)*

Add two elements in the Element Settings:

bi_enet0addr: 0x00e00cbce560

bi_enet1addr:0x00e00cbce559

Please refer to Figure-16.



Create, manage, and run configurations

Download an application to a target, then debug or run the application.

type filter text

- CodeWarrior Attach
  - 54451_uboot MCF54451 Imported Execu
  - uboot_debug MCF54451 Imported Execu
- CodeWarrior Connect
- CodeWarrior Download
  - 5373_MCF5373_Console_Internal_RAM_P
  - 5373_MCF5373_Internal_RAM_PnE USB E
  - 54451_bare_MCF54451_Console_Interna
  - 54451_bare_MCF54451_Internal_RAM_Pr
  - 54451_kernel_Imported Kernel Image_Pr
  - 54451_uboot_MCF54451 Imported Execu
  - kenel_debug_Imported Kernel Image_PnE
  - uboot_debug_MCF54451 Imported Execu
- Launch Group

Filter matched 14 of 14 items

Name: kenel_debug_Imported Kernel Image_PnE USB BDM

Debugger: CodeWarrior Debugger for ColdFire

☑ Stop on startup at:
● Program entry point
○ User specified start_kernel

Debugger Options

Target OS: Linux

☑ Enable Residual Data Settings

Base Type Name: bd_info

Base Address: 0x47da0000

Element Name	Element Value
bi_enet0addr	0x00e00cbce560
bi_enet1addr	0x00e00cbce559

Element Settings

Name:

Value: 0x000000000000

☑ Set Residual Data After Decompress

Figure-16

10) Click Debug button.

This debugs the kernel.

You will “bras 1f” line at head.S and can debug kernel step by step.