PI5008 Toolchain Installation Manual

PIXELPLUS

Hardware

- PC (Windows 7 or above)
- AICE Debugger : JTAG Debugger from Andes
- PI5008 EVB or RDK Board

S/W Deliverables from Pixelplus

- BSPv421_Windows : Andes BSP package
- PI5008-IDEv1.0-setup.exe : Toolchain Installation program
- PI5008 SDK
- Patch : refer to "readme.txt"

Table of Contents

BSP Installation (1)

Andes BSP package includes 3 items.

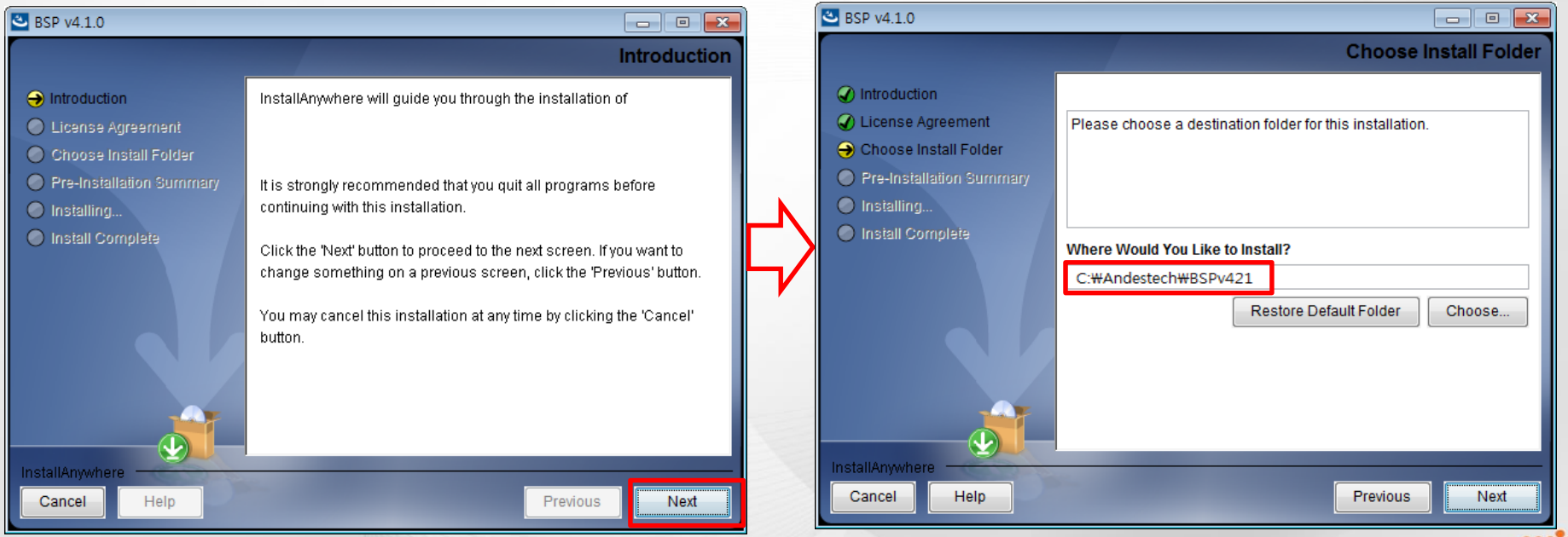
- Cygwin Environment.
- GNU cross compiler for Andes core.
- AICE debugger SW tool.

Installing Andes BSP

1. Unzip “BSPv421_Windows.zip” to a temporary directory.
2. Locate “Setup.exe” from sub-dirs.
   (Ex : “BSPv421_Windows\Windows”)
3. Run “Setup.exe” in admin privileged mode.
4. Respond to “Dialog boxes” during installation.
   (Refer to next 2 pages.)

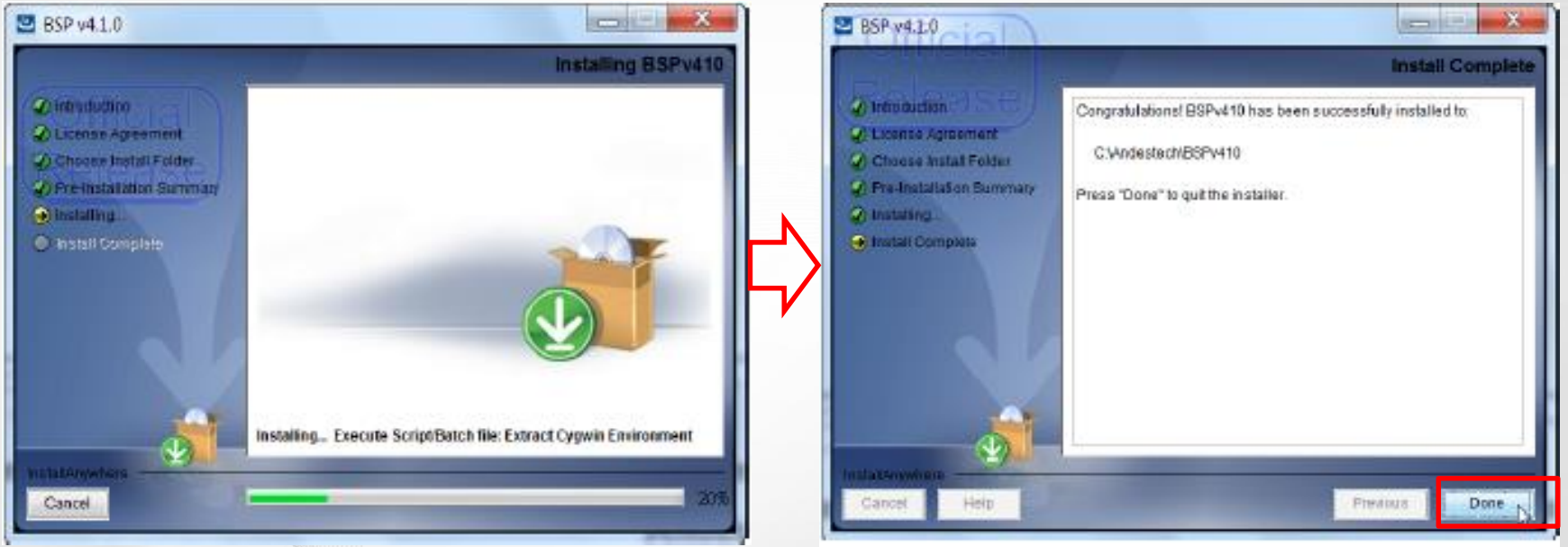
BSP Installation (2)

Andes BSP should be installed on C:\Andestech\BSPv421



BSP Installation (3)

Wait until Installation is completed.



IDE Installation (1)

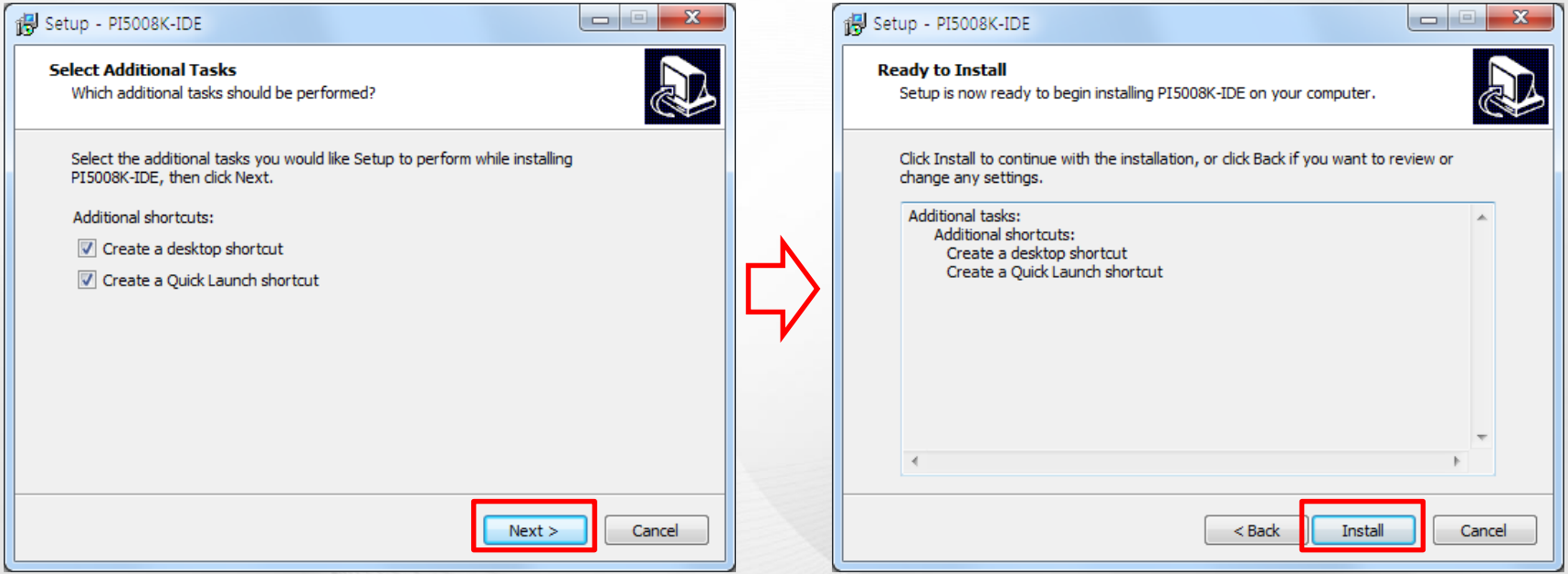
Eclipse is used to provide IDE.
(Refer to http://help.eclipse.org/oxygen/index.jsp?nav=%2F0)

Installing IDE

1. Unzip “PI5008-IDEv1.0-setup.zip” to temporary directory.
2. Locate “PI5008-IDEv1.0-setup.exe”
3. Run “PI5008-IDEv1.0-setup.exe” in admin privileged mode.
4. Respond to “Dialog boxes” during installation.
   (Refer to next 2 pages)

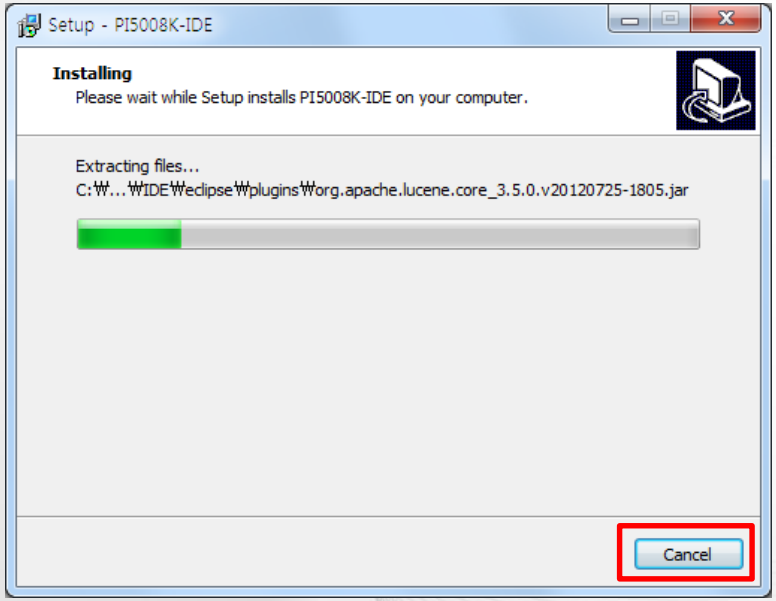
IDE Installation (2)

IDE will be installed at C:\PI5008K\IDE\eclipse.



IDE Installation (3)

Wait until the installation completes.



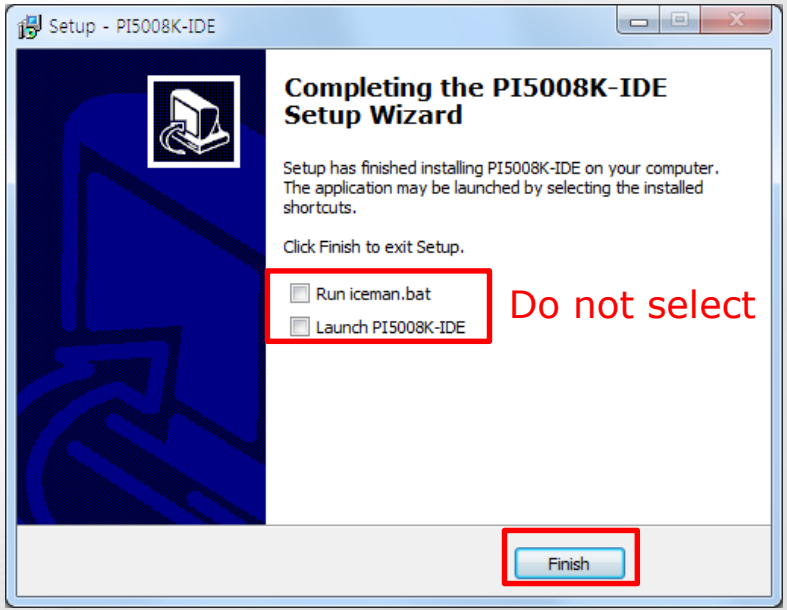
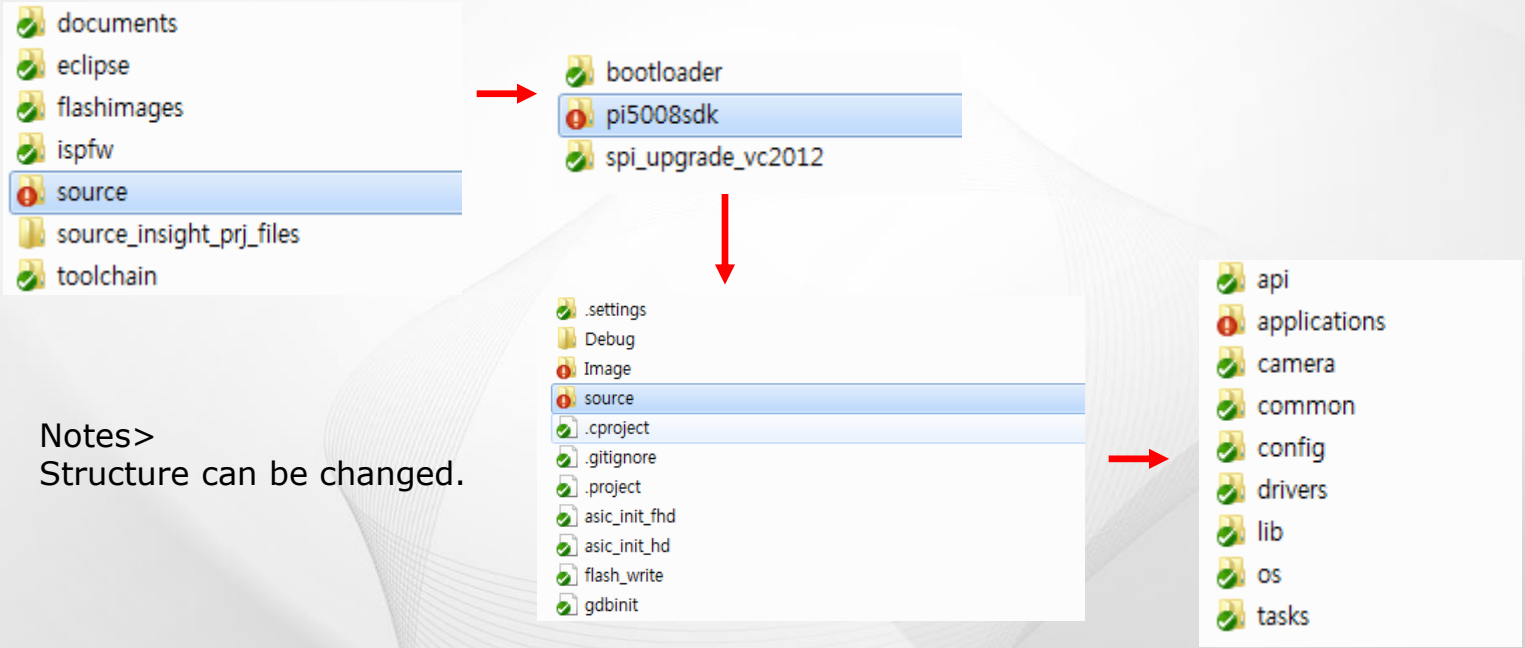


Table of Contents

Build (1) : Decompressing SDK

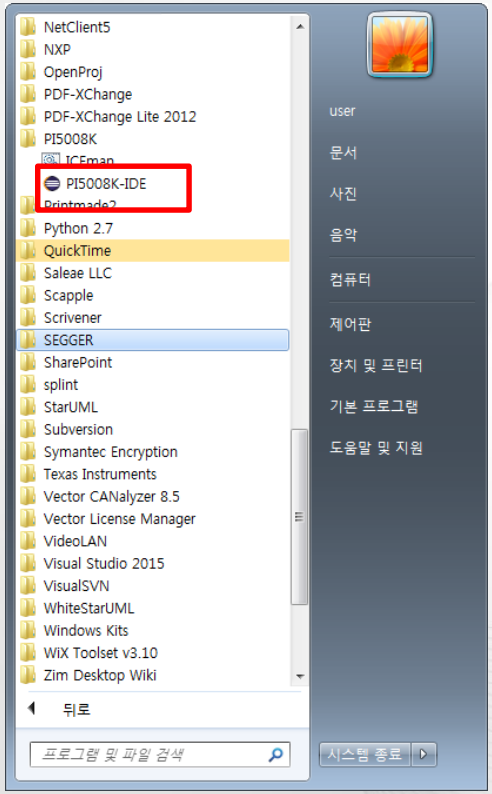
1. Unzip “PI5008_SDK.zip”.
   Folder structure is as below.*



Notes>
Structure can be changed.

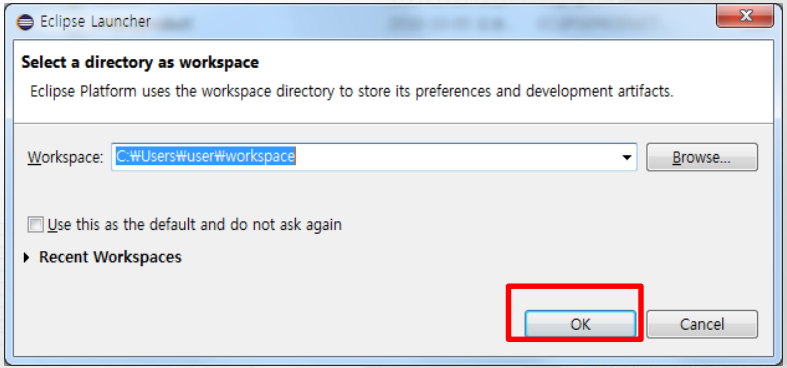
Build (2) : Run IDE

1. Run IDE



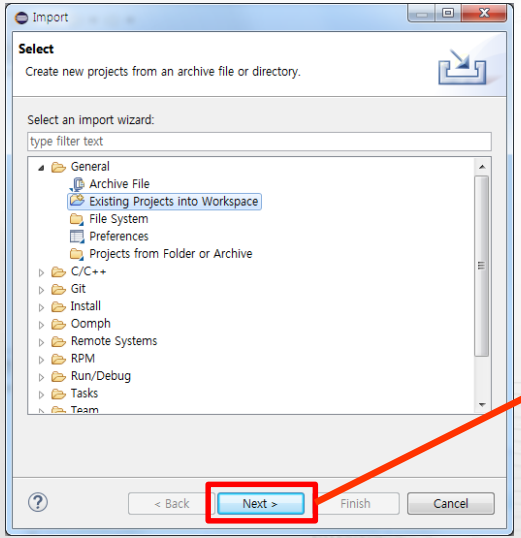
2. Splash Screen will be displayed.

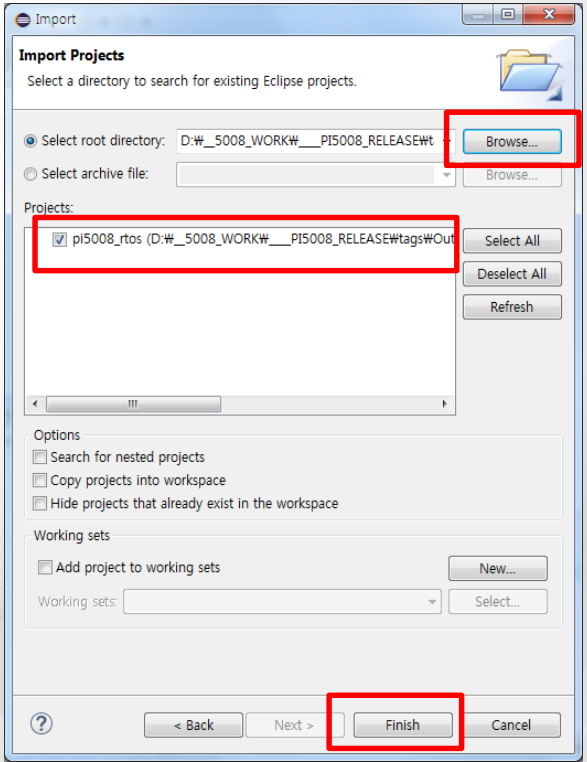
3. Select workspace folder (Any parent folder of SDK will be



Build (3) : Import Project Into Your Workspace

1. Select “File”=>”Import”.
2. Select “Existing Projects into Workspace” at the Dialog Box.
3. Select a folder which has SDK source.
   (Ex : “PI5008_SDK\source”)
4. “Finish”

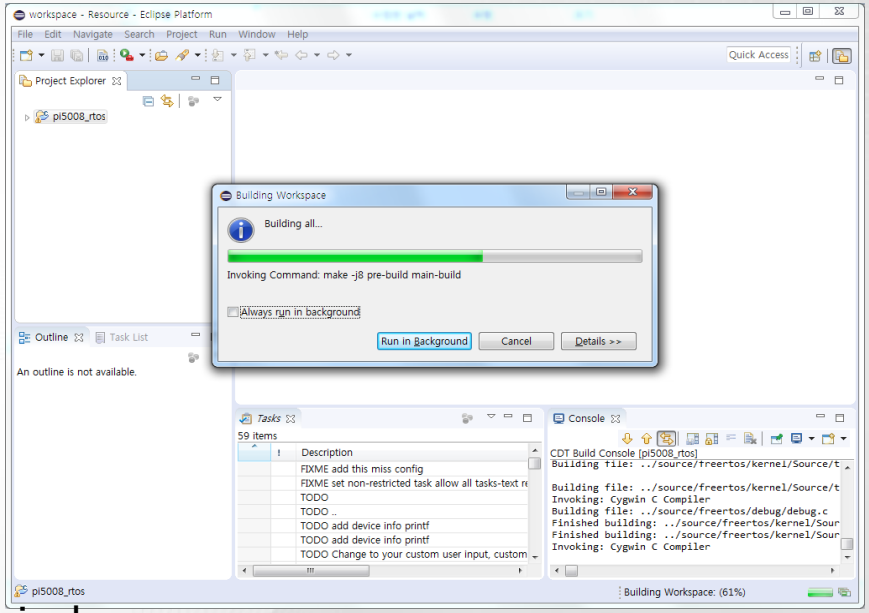




PIXELPLUS

Build (4) : Build Project

1. Select “Project”=>”Build All”



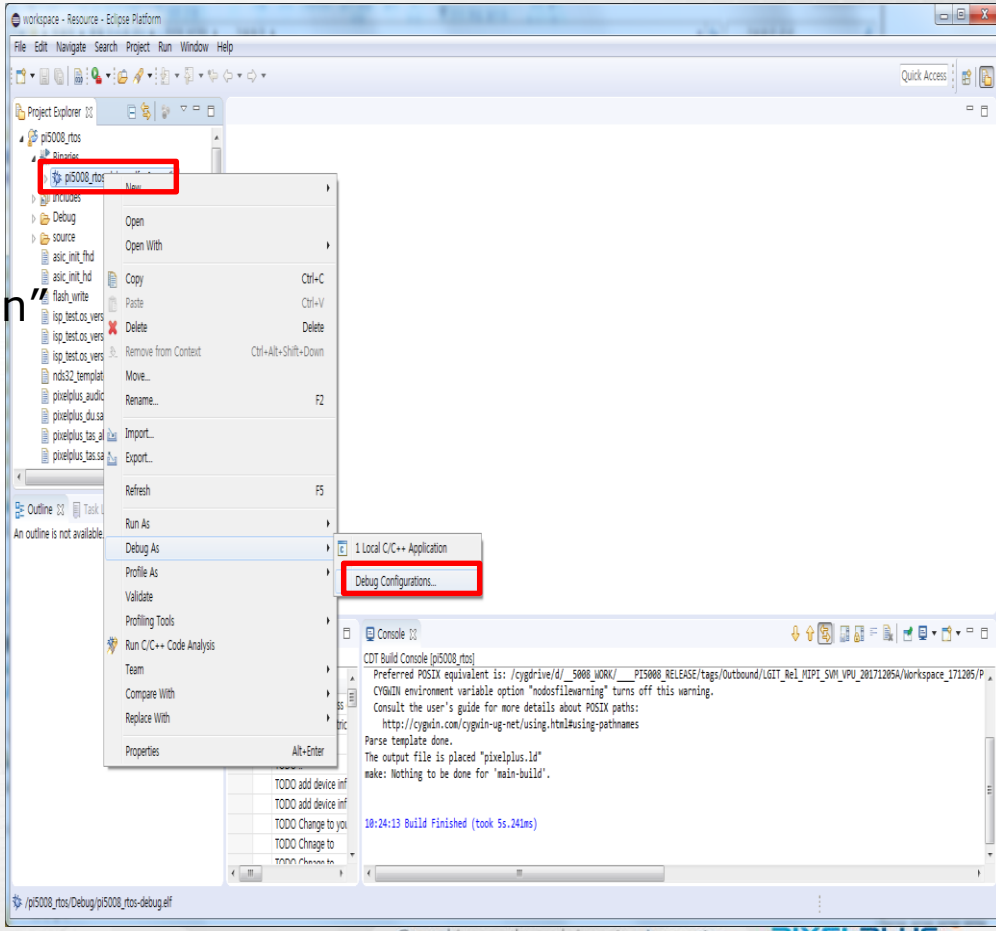
2. The result will be shown in “Console” window.

```
Building target: pi5008_rtos-debug.elf
Invoking: Cygwin C++ Linker
nds32le-elf-gcc ../source/drivers/system/pi5008_config.h -O0 -g3 -mcmodel=large -nostartfiles -fno-builtin -static -Wl,--gc-sections -mext-dsp -T"pixelplus
Finished building target: pi5008_rtos-debug.elf

/usr/bin/make --no-print-directory post-build
nds32le-elf-objdump -x -d -C pi5008_rtos-debug.elf > pi5008_rtos-debug.dasm && nds32le-elf-objcopy -S -O binary pi5008_rtos-debug.elf pi5008_rtos-debug.bi


10:59:03 Build Finished (took 2m:1s.38ms)
```

Table of Contents

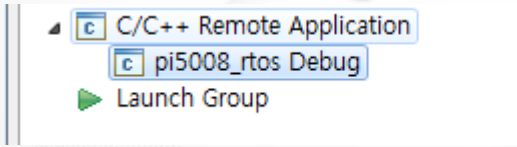
Debugging (1) : Configure Debug Session

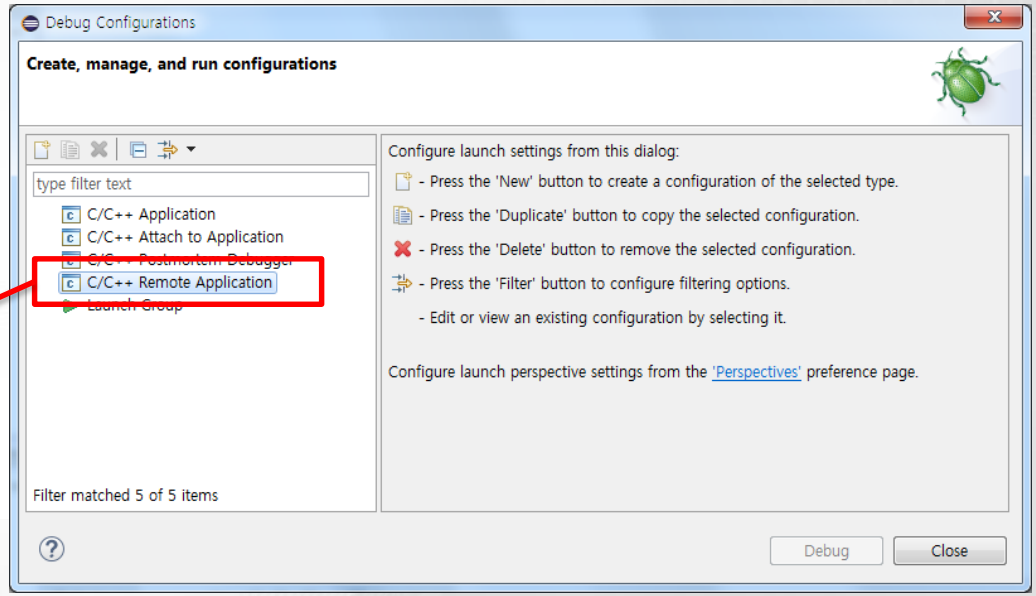
1. Choose elf file
   “Project” => “Binaries” => “.Elf”
2. Click right button on .elf
3. Select
   “Debug As” => “Debug Configuration”



Debugging (2) : Create a Remote Application

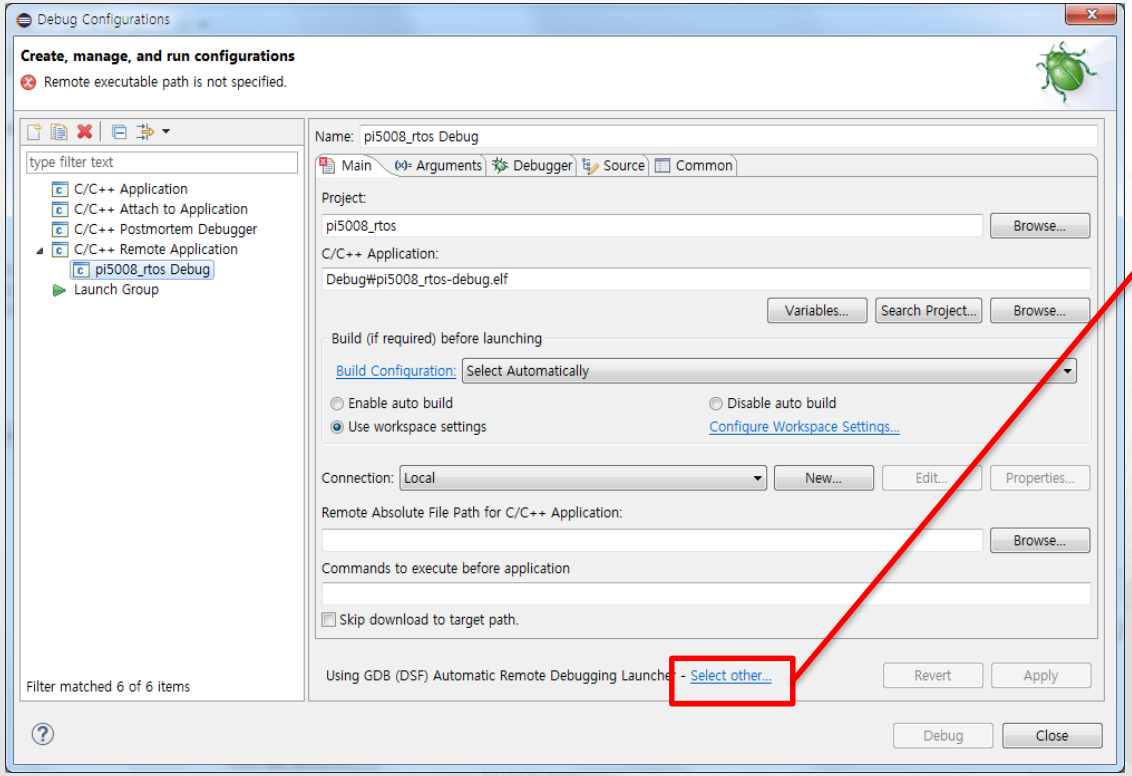
1. Double Click on
   "C/C++ Remote Application"

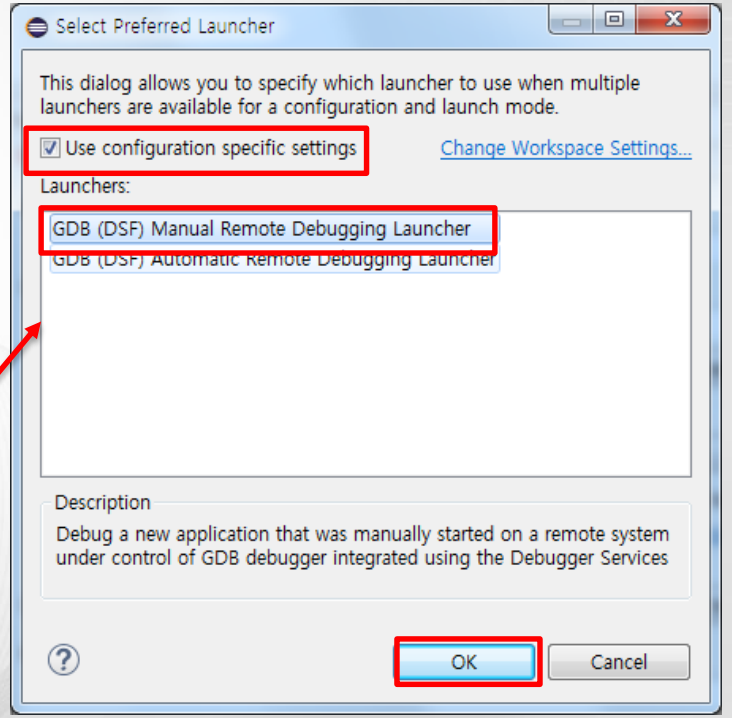




Debugging (3) : Configure “Main” Tab (1)

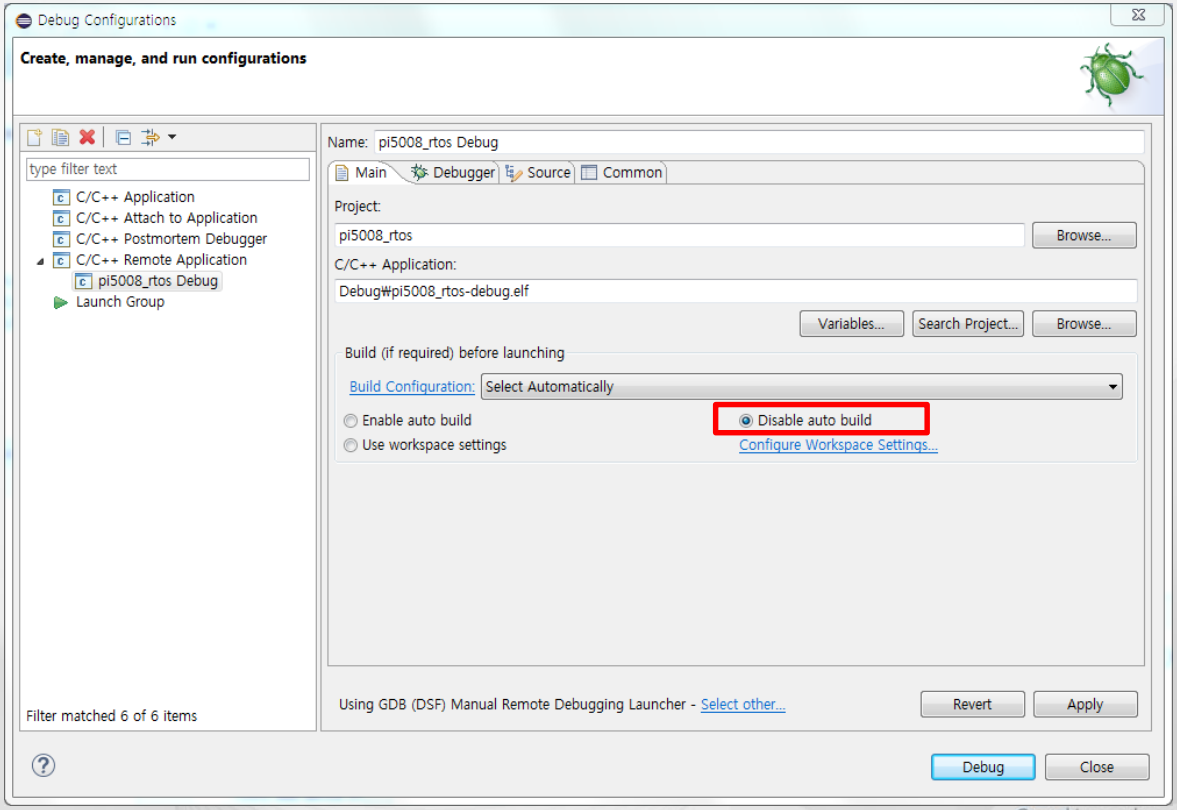
1. Click on “Select Other…”
2. Select Manual Remote Debugging Launcher.





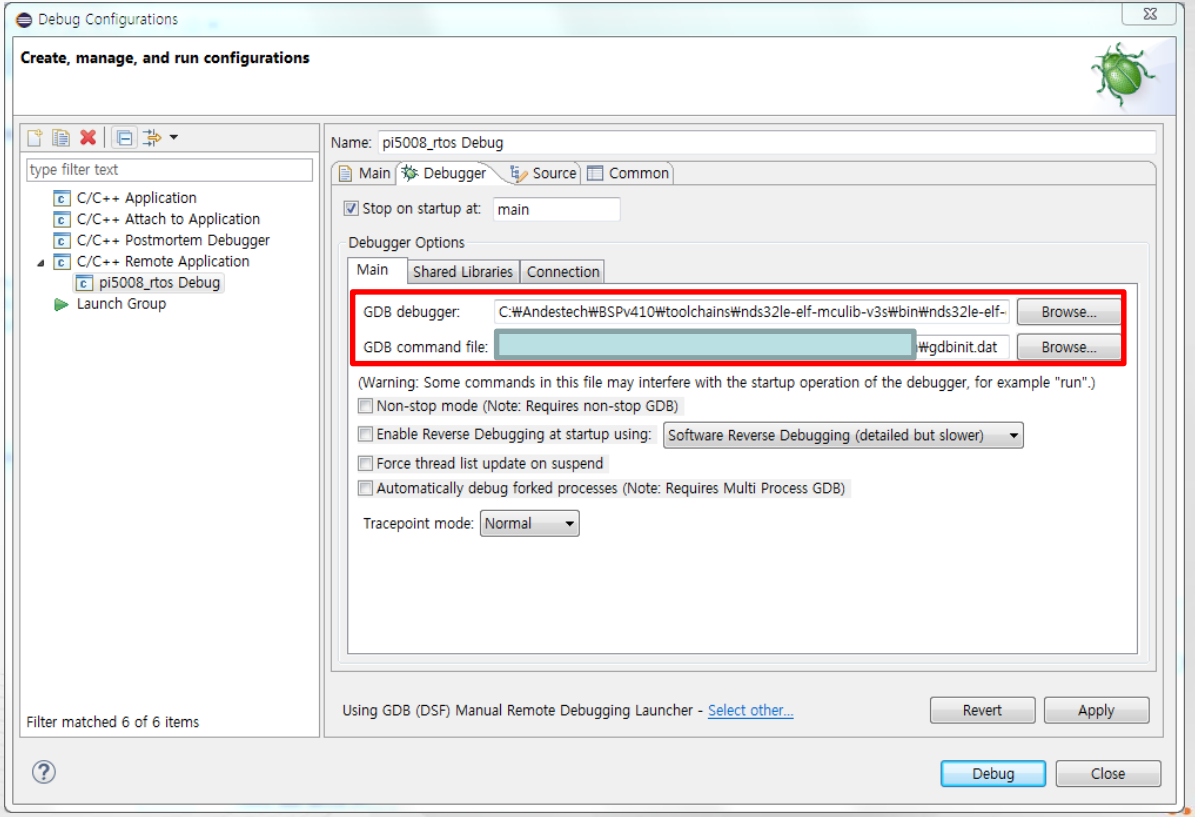
Debugging (4) : Configure “Main” Tab (2)

1. Select “Disable auto build ”



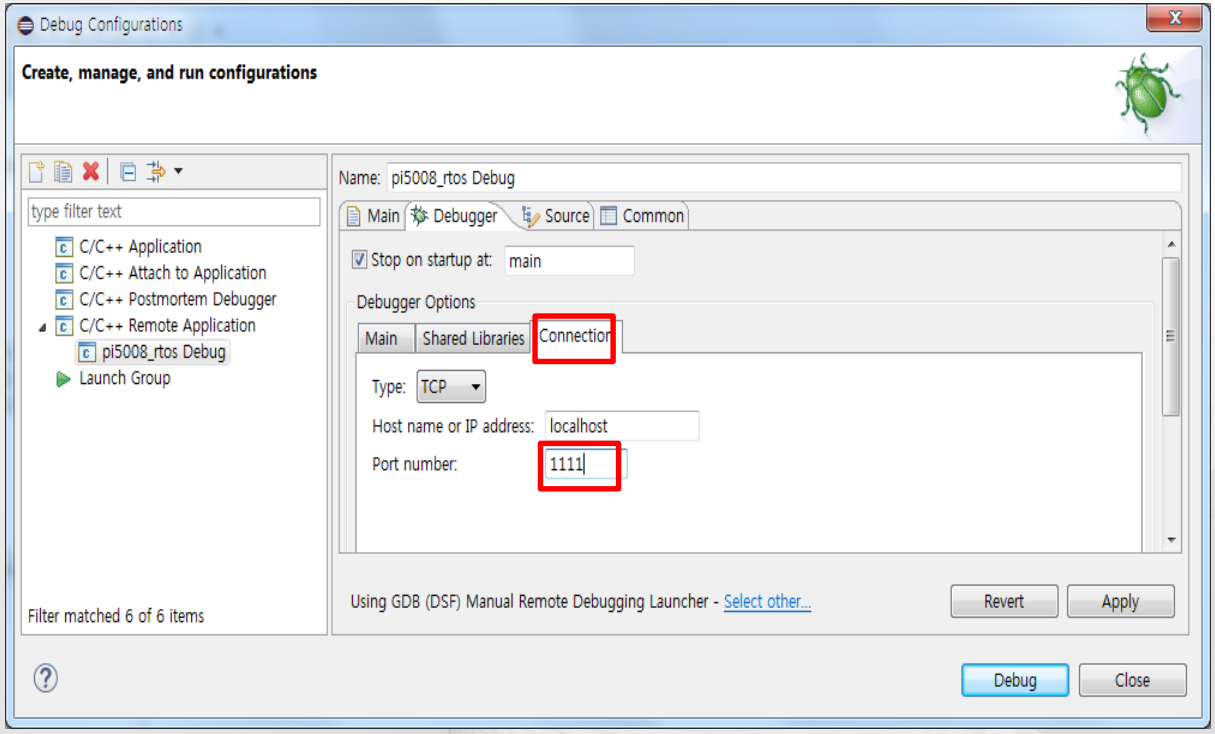
Debugging (5) : Configure “Debugger” => “Main” Tab

1. Select Debugger “nds32le-elf-gdb.exe”
2. Select GDB command file “gdbinit.dat”
   Locate and choose GDB command file at your project folder.



Debugging (6) : Configure “Debugger” => “Connection” Tab

Set port number which AICE tool will be connected to.



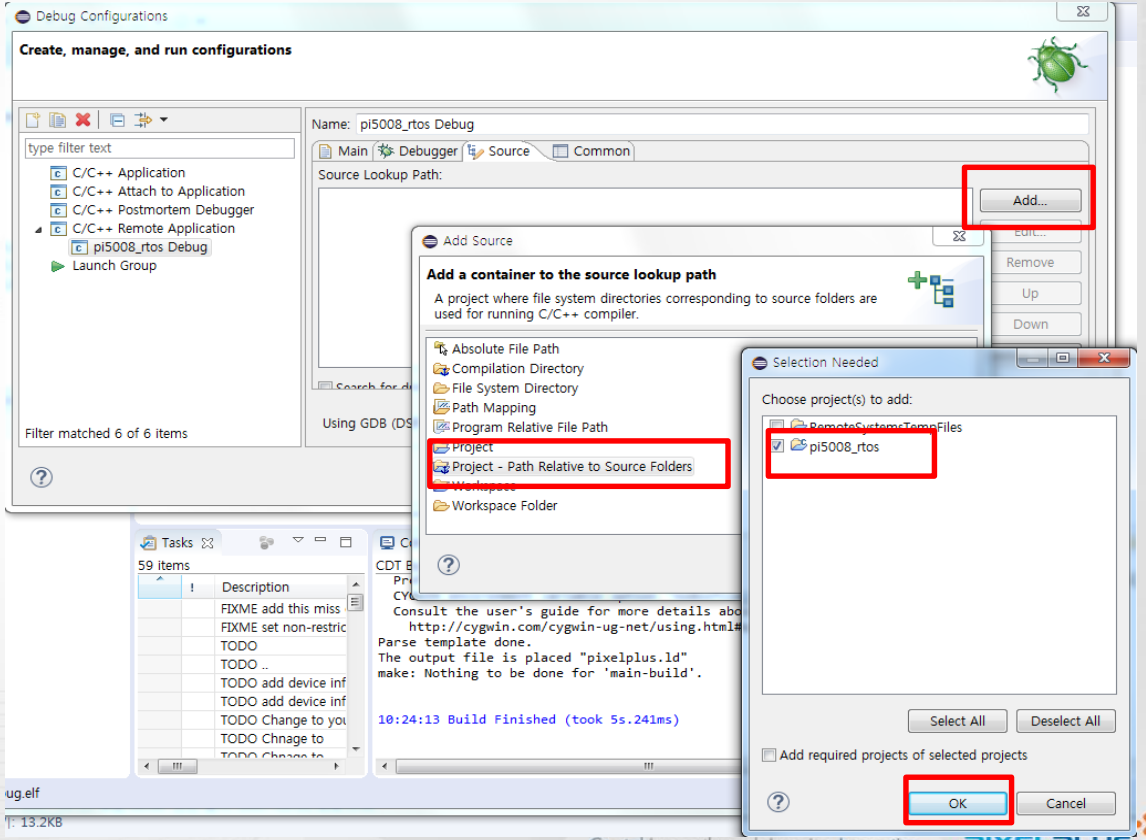

```
$ ./iceman -N EDM_rest.txt
Andes ICEman v3.2.13 (OpenOCD) BUILD_ID: 2017060911
Burner listens on 2354
Telnet port: 4444
TCL port: 6666
Open On-Chip Debugger 0.8.0-dev-00134-gce99bad (2017-06-09-11:28)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.sourceforge.net/doc/doxygen/bugs.html
Andes AICE-MCU v1.8.8
There are 2 cores in target
JTAG frequency 24 MHz
The core #0 listens on 1111.
The core #1 listens on 1112
ICEman is ready to use.
```

AICE tool console

* Refer to the next pages

Debugging (7) : Configure “Source” Tab (1)

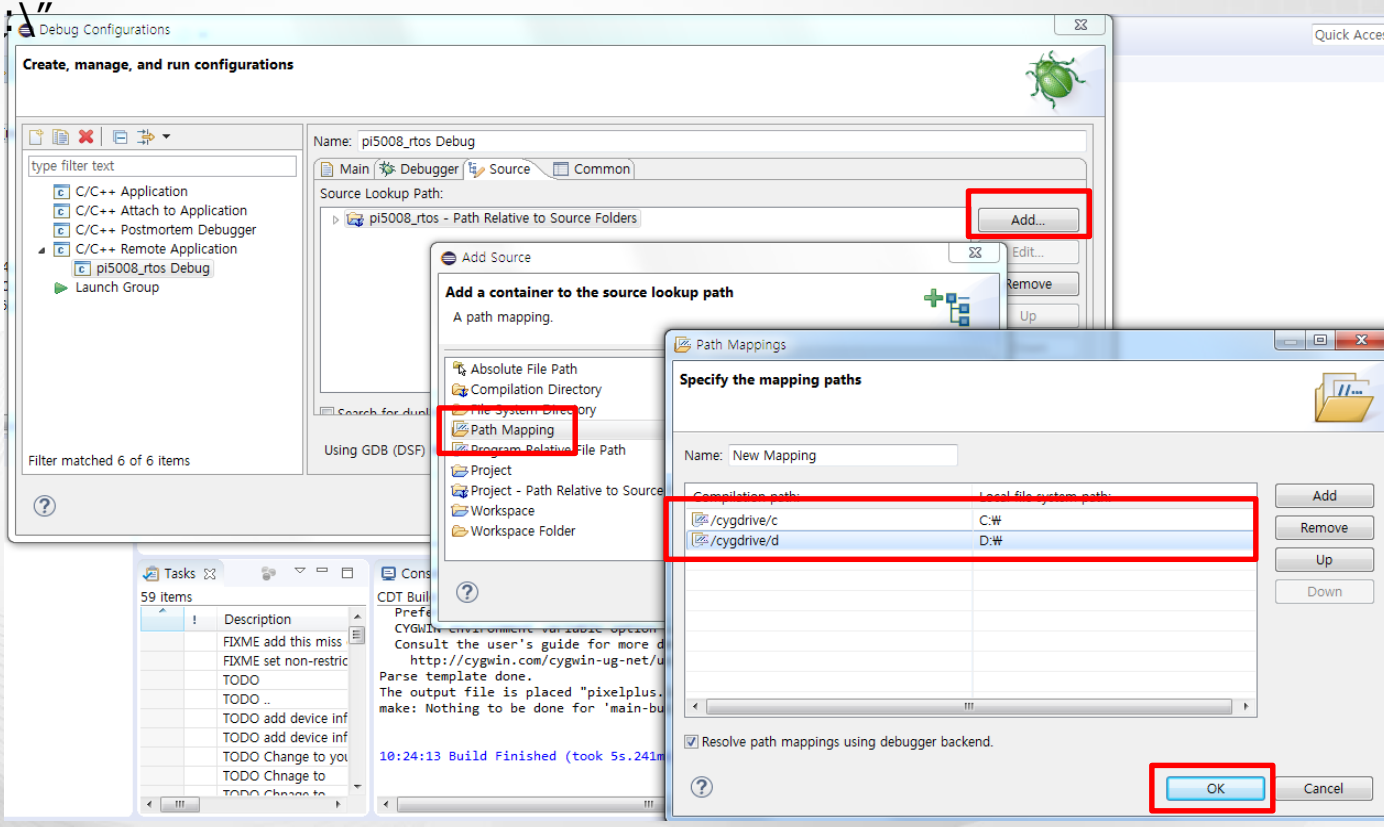
1. Remove “Default”
2. Add “Project – Path Relative to Source Folders.
3. Select “pi5008_rtos”



Debugging (8) : Configure “Source” Tab (2)

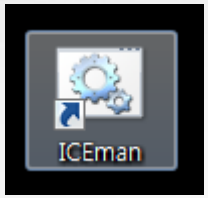
1. Add “Path Mapping”.
2. Specify “Cygwin” Mapping Path.

Ex : “/cygdrive/c“ => “C:\”
“/cygdrive/d” => “D:\”



Debugging (9) : Run Iceman

1. Click on “ICEman” to open a Cygwin console.



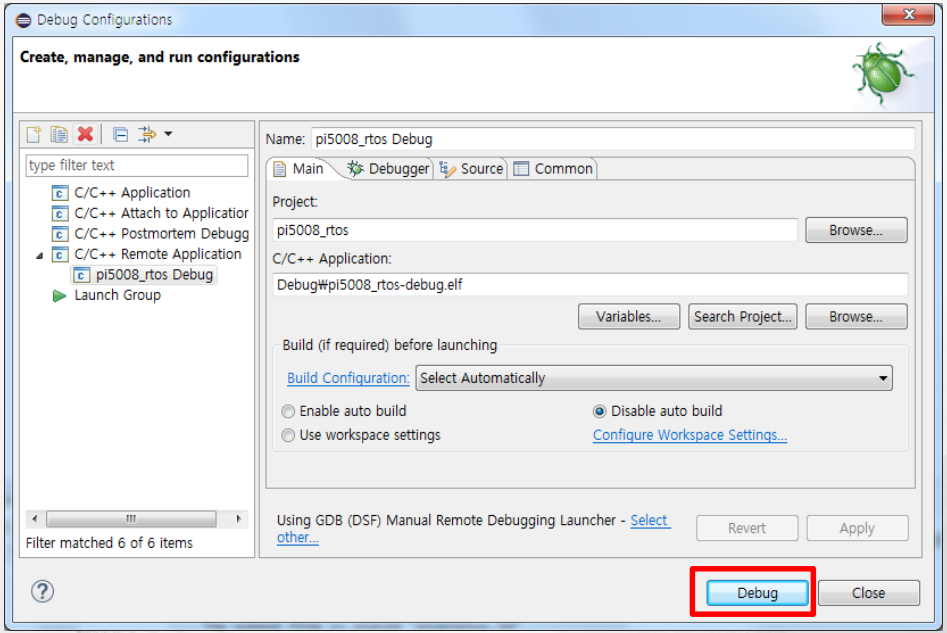
2. Type “./iceman –N EDM_rest.txt” to start AICE debugging session.

```
$ ICEman.exe -N edm_rest.txt
Andes ICEman v3.2.13 (OpenOCD) BUILD_ID: 2017060911
Burner listens on 2354
Telnet port: 4444
TCL port: 6666
Open On-Chip Debugger 0.8.0-dev-00134-gce99bad (2017-06-09-11:28)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.sourceforge.net/doc/doxygen/bugs.html
Andes AICE-MCU v1.8.8
There are 2 cores in target
JTAG frequency 24 MHz
The core #0 listens on 1111.
The core #1 listens on 1112.
ICEman is ready to use.
```

Check if port number is matching with what you specified in “Debugging Configuration”.

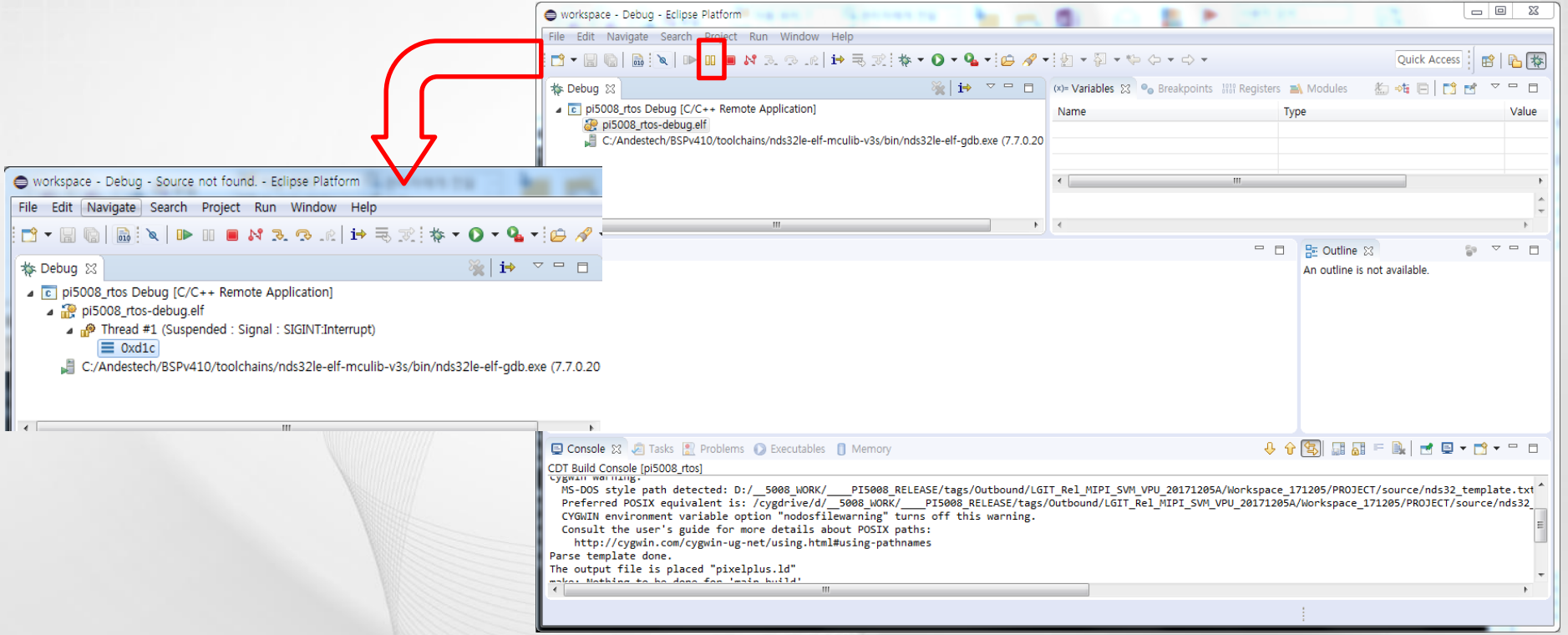
Debugging (10) : Start Debugging

1. Click “Debug” Button in “Debug Configurations” Dialog box.



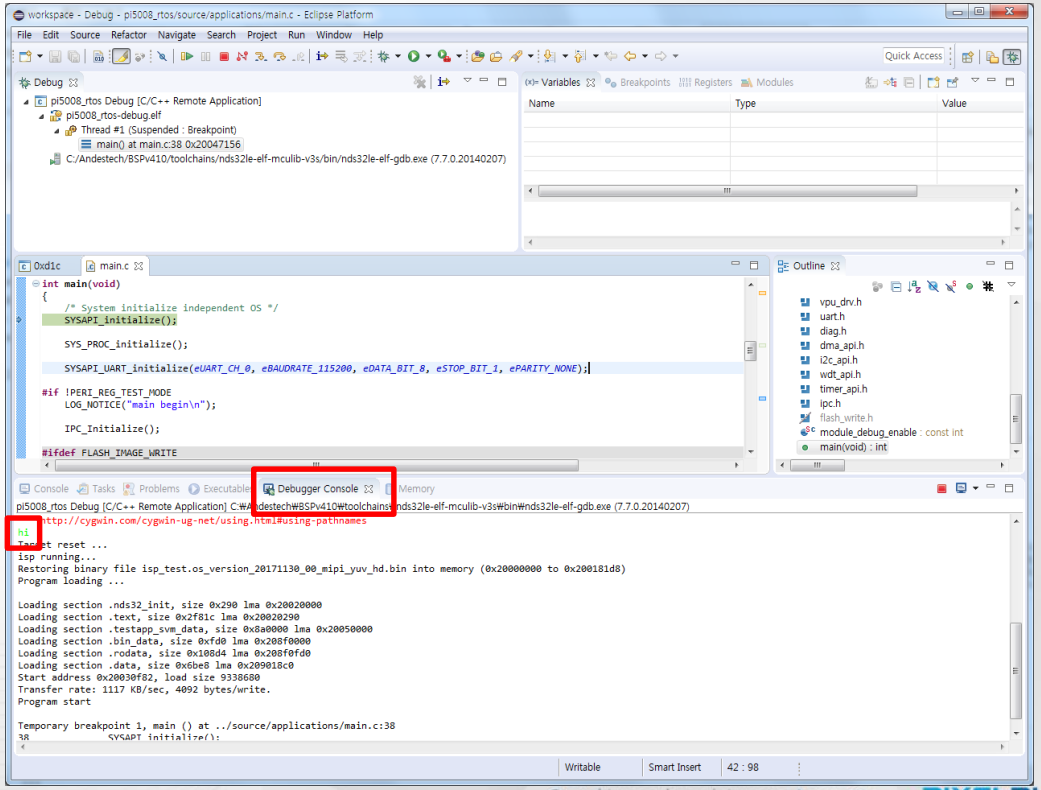
Debugging (11) : Run Debug Script (1)

1. Pause execution by pressing “Pause” button in tool bar of Debug window.



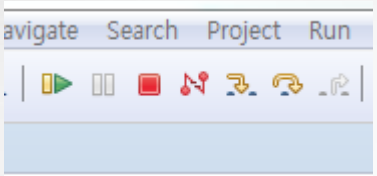
Debugging (12) : Run Debug Script (2)

1. Select “Window” => “Show View” => “Debugger Console” Menu.
2. Type “hi” at “Debugger Console”.
3. After loading execution binaries on DRAM, Debugger Stops at entry of “main” function.



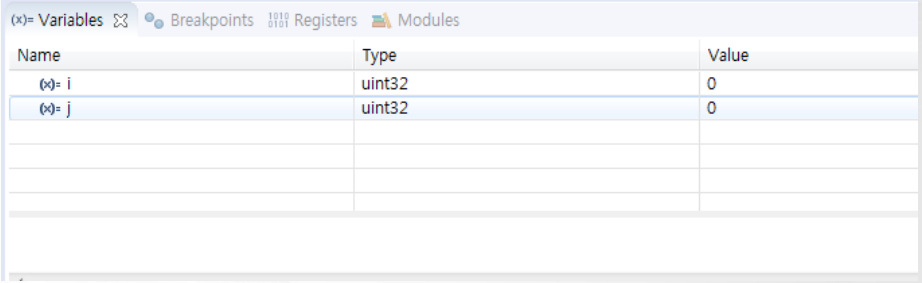
Debugging (13) : Execution Control

1. Execution Control Using Toolbar



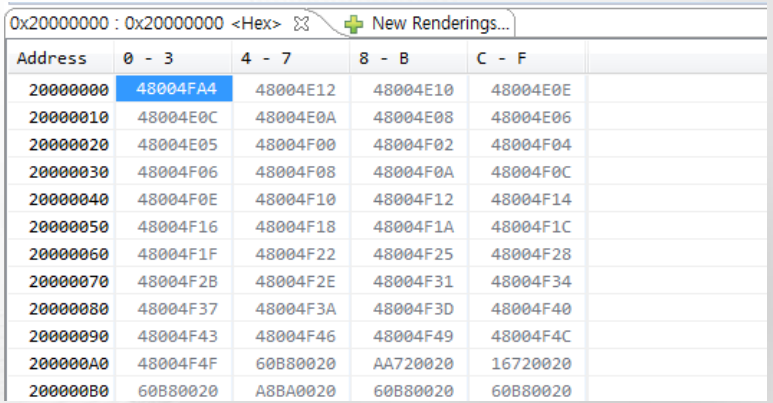
- Run/Resume
- Pause
- Stop
- Step In
- Step Over
- Step Out

2. Variable Monitoring Using "Variables" View



Name	Type	Value
(x)= i	uint32	0
(x)= j	uint32	0

3. Memory Dump Using "Memory" View

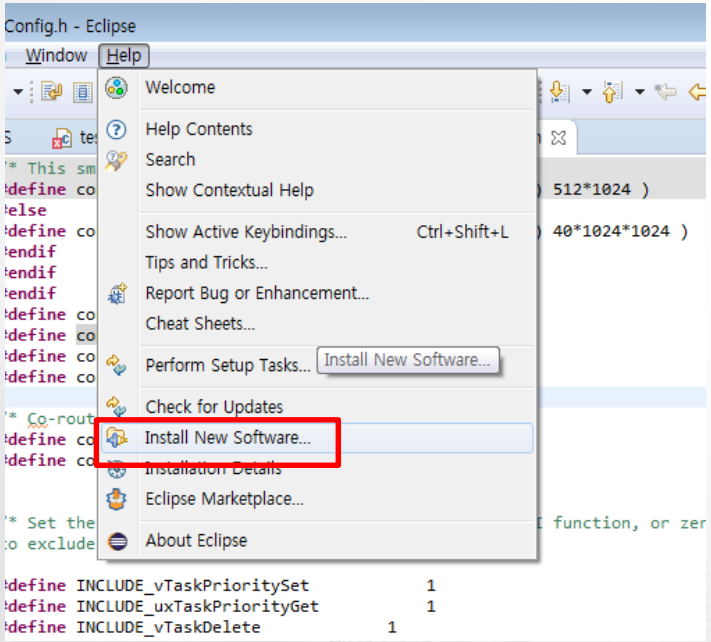


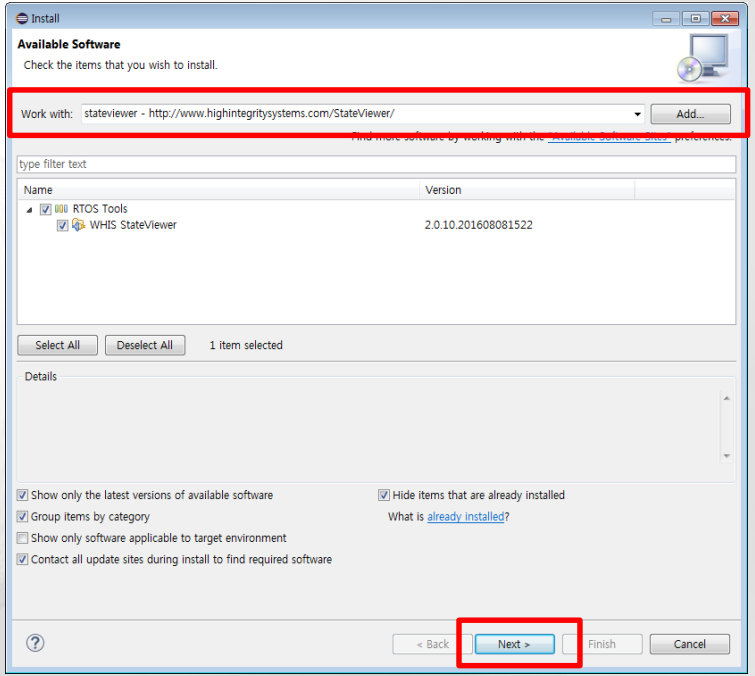
Address	0 - 3	4 - 7	8 - B	C - F
20000000	48004FA4	48004E12	48004E10	48004E0E
20000010	48004E0C	48004E0A	48004E08	48004E06
20000020	48004E05	48004F00	48004F02	48004F04
20000030	48004F06	48004F08	48004F0A	48004F0C
20000040	48004F0E	48004F10	48004F12	48004F14
20000050	48004F16	48004F18	48004F1A	48004F1C
20000060	48004F1F	48004F22	48004F25	48004F28
20000070	48004F2B	48004F2E	48004F31	48004F34
20000080	48004F37	48004F3A	48004F3D	48004F40
20000090	48004F43	48004F46	48004F49	48004F4C
200000A0	48004F4F	60B80020	AA720020	16720020
200000B0	60B80020	A8BA0020	60B80020	60B80020

Table of Contents

Appendix : Install Stateviewer plug-in to aware RTOS

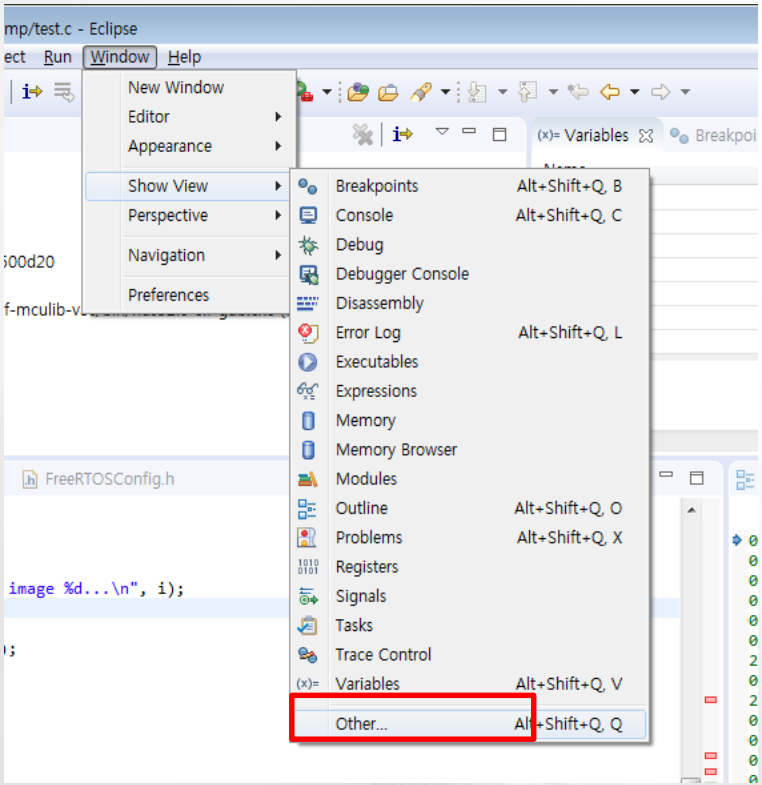
1. Select “Help” => “Install New Software…”
2. Add “http://www.highintegritysystems.com/StateViewer/”

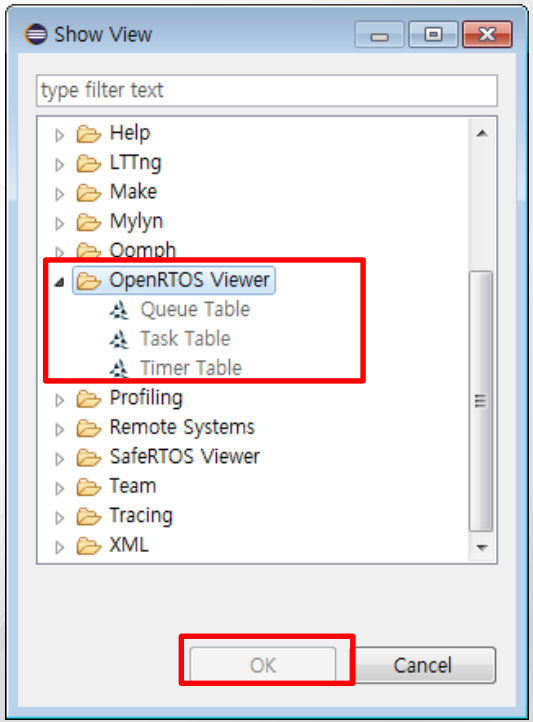




Appendix : Using Eclipse OS Awareness (1)

1. Select “Window” => “Show View” => “Others” Menu
2. Select “Queue Table” or “Task Table” or “Timer Table”





Appendix : Using Eclipse OS Awareness (2)

Ex : Task Table

Queue Table | Task Table | Timer Table

Task Name	Base/Actual Priority	Start of Stack	Top of Stack	State	Event Object	Min Free Stack	Total Runtime	Delta Runtime
AppTask_100ms	13/13	0x21f84be0	0x21f86a60	BLOCKED	None	Off	Unknown	--
AppTask_GetEven	12/12	0x21f78820	0x21f7a690	BLOCKED	None	Off	Unknown	--
AppTask_StateMa	12/12	0x21f76780	0x21f785d8	BLOCKED	None	Off	Unknown	--
IDLE	0/0	0x21f86c80	0x21f8ab38	RUNNING	None	Off	Unknown	--
TaskPI_ExtInput	12/12	0x21f7a8c0	0x21f7c738	BLOCKED	None	Off	Unknown	--
TaskPI_SYSTEM	12/12	0x21f746e0	0x21f76438	BLOCKED	None	Off	Unknown	--
TaskPI_UI	12/12	0x21f7c960	0x21f7e7d8	BLOCKED	None	Off	Unknown	--
Tmr Svc	19/19	0x21f8ae20	0x21f8b0a8	BLOCKED	Not known	Off	Unknown	--
VPU_DetectionTa	12/12	0x21f7ea00	0x21f80880	BLOCKED	None	Off	Unknown	--
VPU_DrawingTask	12/12	0x21f82b40	0x21f84978	BLOCKED	None	Off	Unknown	--
VPU_MatchingTas	12/12	0x21f80aa0	0x21f82920	BLOCKED	None	Off	Unknown	--

Crystal Image through Imaging Innovation PIXELPLUS

Q & A