IMP Series
Standalone Mode User Guide

Version: V.1.30

Date: 2014.10

http://www.epcio.com.tw

Table of Contents

1. IMP Series Standalone Mode Introduction

The Intelligent Motion control Platform (IMP) is equipped with built-in hard real-time operation system (VxWorks) and CPU (PowerPC 440), which use the Motion Control Command Library (MCCL). The IMP can be implemented and used under either PC or standalone modes.

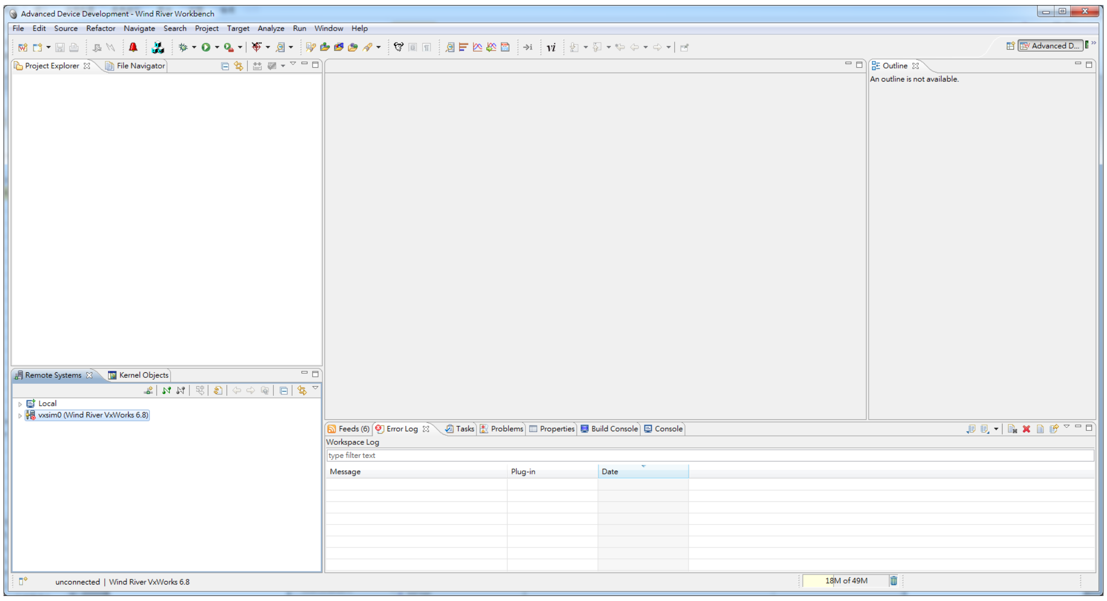
For the application under the standalone mode, users must first install the standalone mode development environment, Wind River Workbench 3.2 (Note 1), and download the IMP series standalone mode examples from the Download Section of the EPCIO website (www.epcio.com.tw) for reference. The instructions for the use of the IMP Device Driver Library (IDDL) and the Motion Control Command Library (MCCL) can be found in the related references and operation as well as user manuals.

Note 1：A one-month trial version can be provided to users for free. If needed, please contact us directly.

2. WindRiver Workbench

WindRiver Workbench allows users to write and compile work programs in standalone mode.

1. After the installation of the WindRiver CD is completed, launch the Wind River Workbench on the desktop. Once it is launched, the screen as shown below displayed：



An outline is not available.

Fig.2-1 Launch Wind River Workbench

2. Select the following from the menu in the Wind River Workbench development environment【File ➔ New ➔ VxWorks Real-Time Process Project】

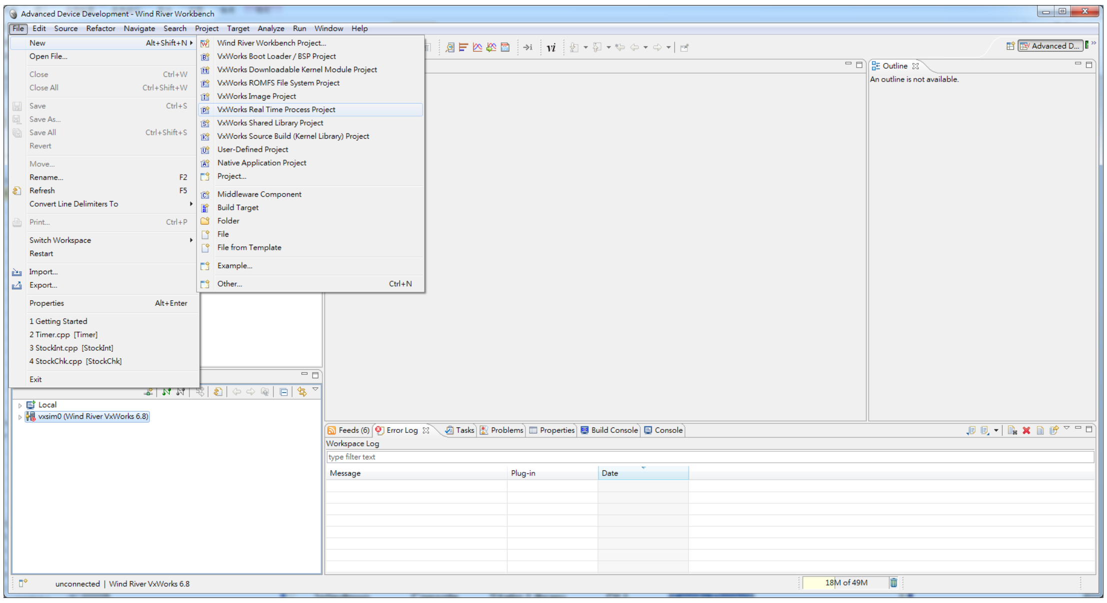
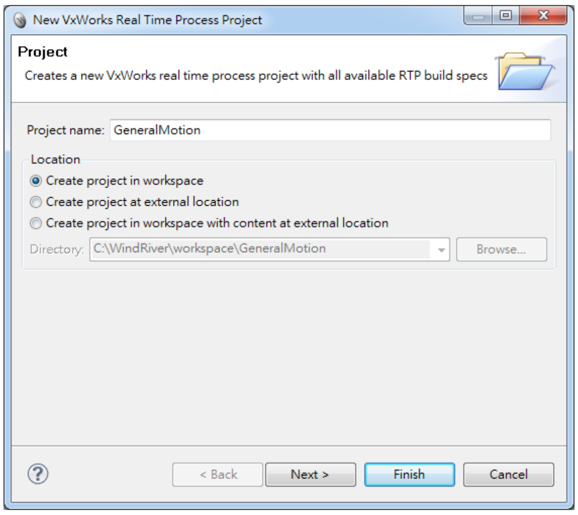

Advanced Device Development - Wind River Workbench
File Edit Source Refactor Navigate Search Project Target Analyze Run Window Help
New Alt+Shift+N
Open File...
Close Ctrl+W
Close All Ctrl+Shift+W
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Revert
Move...
Rename... F2
Refresh F5
Convert Line Delimiters To
Print... Ctrl+P
Switch Workspace
Restart
Import...
Export...
Properties Alt+Enter
1 Getting Started
2 Timer.cpp [Timer]
3 StockInt.cpp [StockInt]
4 StockChk.cpp [StockChk]
Exit
Wind River Workbench Project...
VxWorks Boot Loader / BSP Project
VxWorks Downloadable Kernel Module Project
VxWorks ROMFS File System Project
VxWorks Image Project
VxWorks Real Time Process Project
VxWorks Shared Library Project
VxWorks Source Build (Kernel Library) Project
User-Defined Project
Native Application Project
Project...
Middleware Component
Build Target
Folder
File
File from Template
Example...
Other... Ctrl+N
Outline
An outline is not available.
Advanced D...
Local
vxsim0 (Wind River VxWorks 6.8)
Feeds (6)
Error Log
Tasks
Problems
Properties
Build Console
Console
Workspace Log
type filter text
Message
Plug-in
Date
unconnected | Wind River VxWorks 6.8
18M of 49M

Fig.2-2 Create new project

After selecting "VxWorks Real-Time Process Project", the dialog box shown as below displayed：


New VxWorks Real Time Process Project

Project

Creates a new VxWorks real time process project with all available RTP build specs

Project name: GeneralMotion

Location

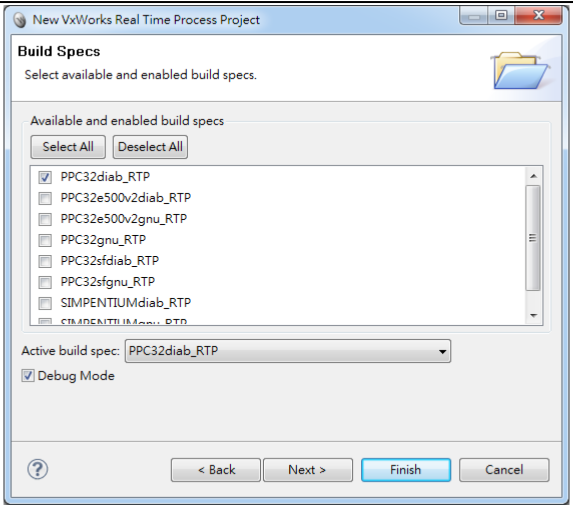
- [✓ Create project in workspace]
- Create project at external location
- Create project in workspace with content at external location

Directory: C:\WindRiver\workspace\GeneralMotion Browse...

< Back Next > Finish Cancel

Fig.2-3 Enter the name for the project

Next, enter an appropriate name for the project in the edit box after "Project name": -- for example, General Motion, or other names you like. Select "Create a project in workspace" in the Location option, and press Next until Build Specs dialog box appears as shown below：


New VxWorks Real Time Process Project

Build Specs

Select available and enabled build specs.

Available and enabled build specs

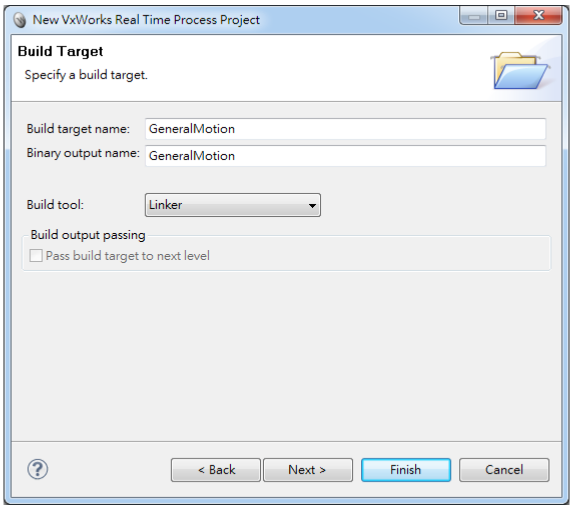
- [x] PPC32diab_RTP
- [ ] PPC32e500v2diab_RTP
- [ ] PPC32e500v2gnu_RTP
- [ ] PPC32gnu_RTP
- [ ] PPC32sfdiab_RTP
- [ ] PPC32sfgnu_RTP
- [ ] SIMPENTIUMdiab_RTP

Active build spec: PPC32diab_RTP

- [x] Debug Mode

Fig.2-4 Building specifications

In the Build Specs dialog box, check only "PPC32diab_RTP" for the Available and Enabled Build Specs, and select "PPC32diab_RTP" for the Active Build Spec. Check the Debug Mode option, and press Next to display the dialog box as shown below：


New VxWorks Real Time Process Project

Build Target

Specify a build target.

Build target name: GeneralMotion

Binary output name: GeneralMotion

Build tool: Linker

Build output passing

- [ ] Pass build target to next level

Fig.2-5 Building target

In the Build Target dialog box, select "Linker" for the Build tool, and then press the Finish button to complete the new project. Subsequently, the screen as shown below displayed：

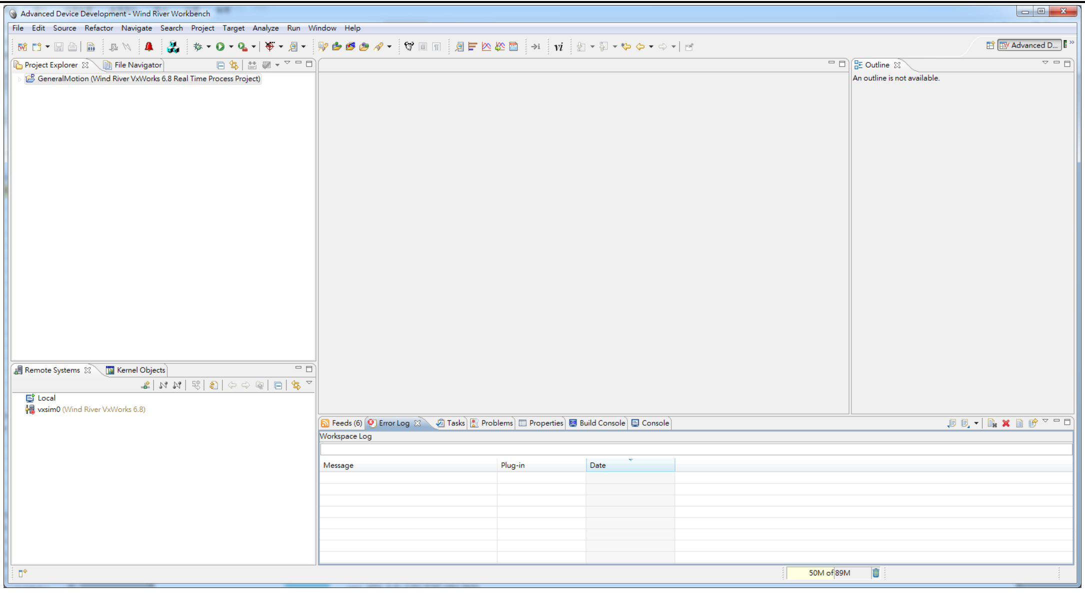

Advanced Device Development - Wind River Workbench
File Edit Source Refactor Navigate Search Project Target Analyze Run Window Help
Project Explorer
File Navigator
GeneralMotion (Wind River VxWorks 6.8 Real Time Process Project)
Outline
An outline is not available.
Remote Systems
Kernel Objects
Local
vxsim0 (Wind River VxWorks 6.8)
Feeds (6)
Error Log
Tasks
Problems
Properties
Build Console
Console
Workspace Log
Message
Plug-in
Date
50M of 89M

Fig.2-6 A new project is created

3. From the Project Explorer, select the project name and right click, then select 【New ➔ File】 from the pop-up menu to add the project file, as shown below：

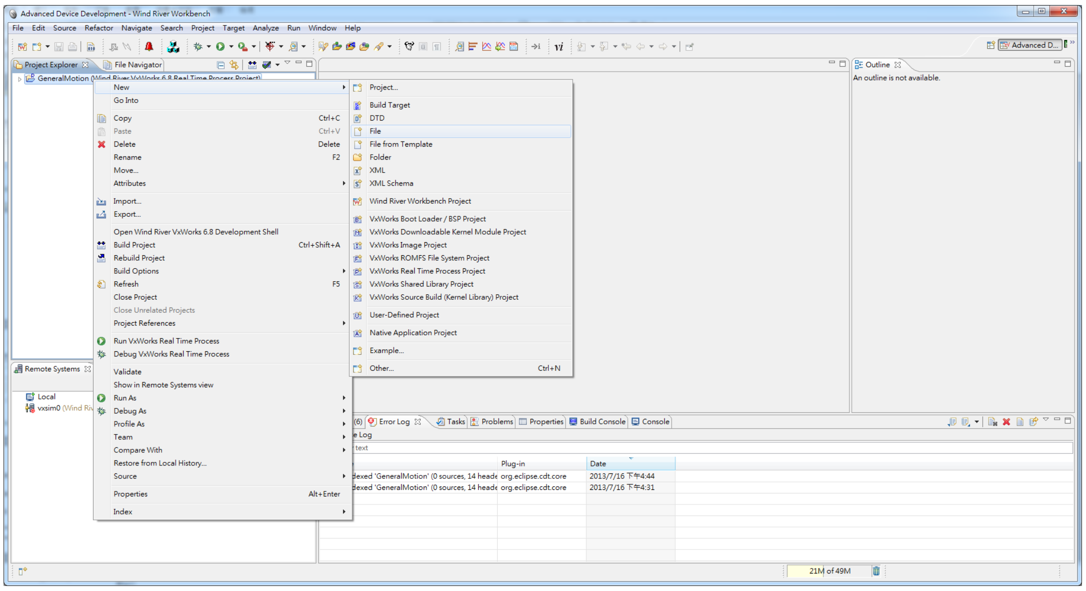
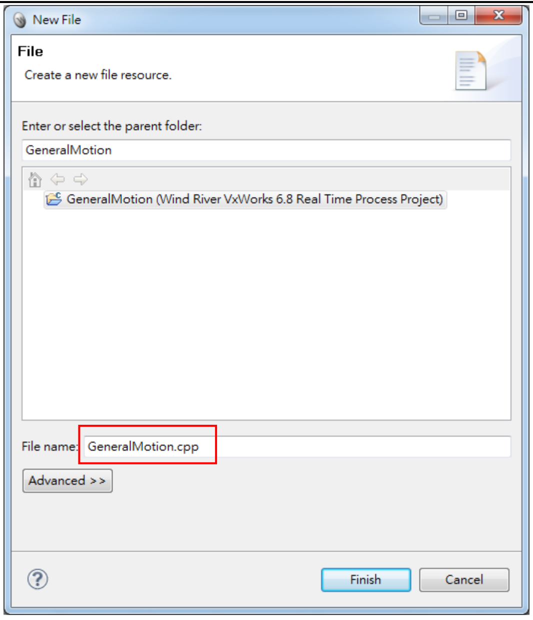

Advanced Device Development - Wind River Workbench
File Edit Source Refactor Navigate Search Project Target Analyze Run Window Help
Project Explorer
File Navigator
GeneralMotion (Wind River VxWorks 6.8 Real Time Process Project)
New
Go Into
Copy Ctrl+C
Paste Ctrl+V
Delete Delete
Rename F2
Move...
Attributes
Import...
Export...
Open Wind River VxWorks 6.8 Development Shell
Build Project Ctrl+Shift+A
Rebuild Project
Build Options
Refresh F5
Close Project
Close Unrelated Projects
Project References
Run VxWorks Real Time Process
Debug VxWorks Real Time Process
Validate
Show in Remote Systems view
Run As
Debug As
Profile As
Team
Compare With
Restore from Local History...
Source
Properties Alt+Enter
Index
Project...
Build Target
DTD
File
File from Template
Folder
XML
XML Schema
Wind River Workbench Project
VxWorks Boot Loader / BSP Project
VxWorks Downloadable Kernel Module Project
VxWorks Image Project
VxWorks ROMFS File System Project
VxWorks Real Time Process Project
VxWorks Shared Library Project
VxWorks Source Build (Kernel Library) Project
User-Defined Project
Native Application Project
Example...
Other... Ctrl+N
Outline
An outline is not available.
Remote Systems
Local
vxsim0
Error Log
Tasks
Problems
Properties
Build Console
Console
Plug-in
Date
dexed 'GeneralMotion' (0 sources, 14 heade org.eclipse.cdt.core 2013/7/16 下午4:44
dexed 'GeneralMotion' (0 sources, 14 heade org.eclipse.cdt.core 2013/7/16 下午4:31
21M of 49M

Fig.2-7 Adding file

After selecting File, the dialog box shown as below will be displayed：



File

Create a new file resource.

Enter or select the parent folder:

GeneralMotion

GeneralMotion (Wind River VxWorks 6.8 Real Time Process Project)

File name: GeneralMotion.cpp

Fig.2-8 Enter the name of the file

Next, enter an appropriate name for the file in the edit box for “File Name”:-- for example, GeneralMotion.cpp, or other names you like, and then press Finish to complete adding new file. Subsequently, the screen as shown below will be displayed：

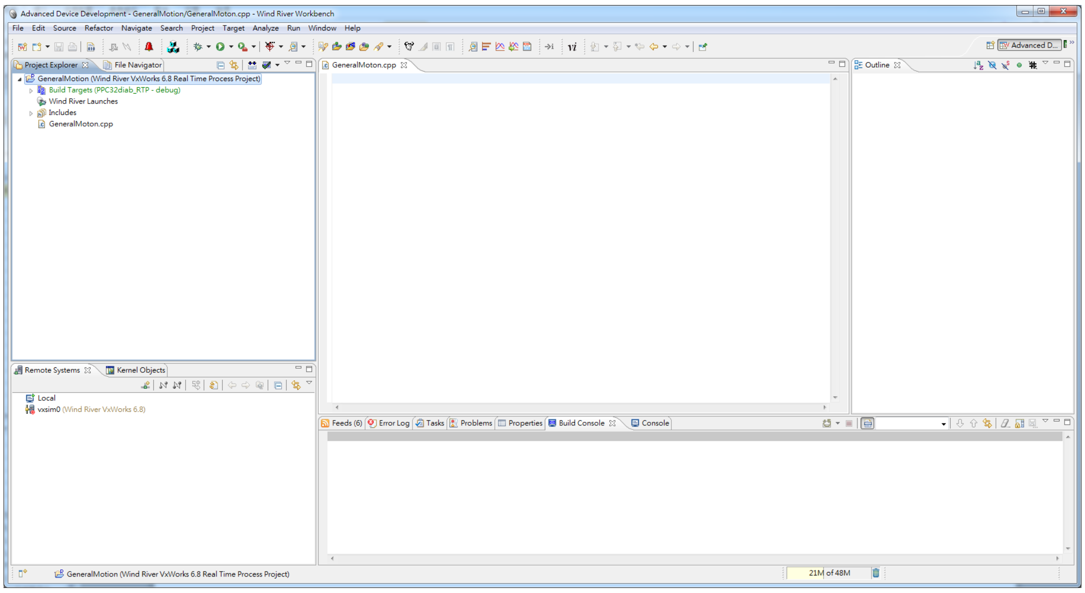

Advanced Device Development - GeneralMotion/GeneralMoton.cpp - Wind River Workbench
File Edit Source Refactor Navigate Search Project Target Analyze Run Window Help
Project Explorer
File Navigator
GeneralMotion (Wind River VxWorks 6.8 Real Time Process Project)
Build Targets (PPC32diab_RTP - debug)
Wind River Launches
Includes
GeneralMoton.cpp
GeneralMoton.cpp
Outline
Remote Systems
Kernel Objects
Local
vxsim0 (Wind River VxWorks 6.8)
Feeds (6)
Error Log
Tasks
Problems
Properties
Build Console
Console
21M of 48M

Fig.2-9 Adding file is completed

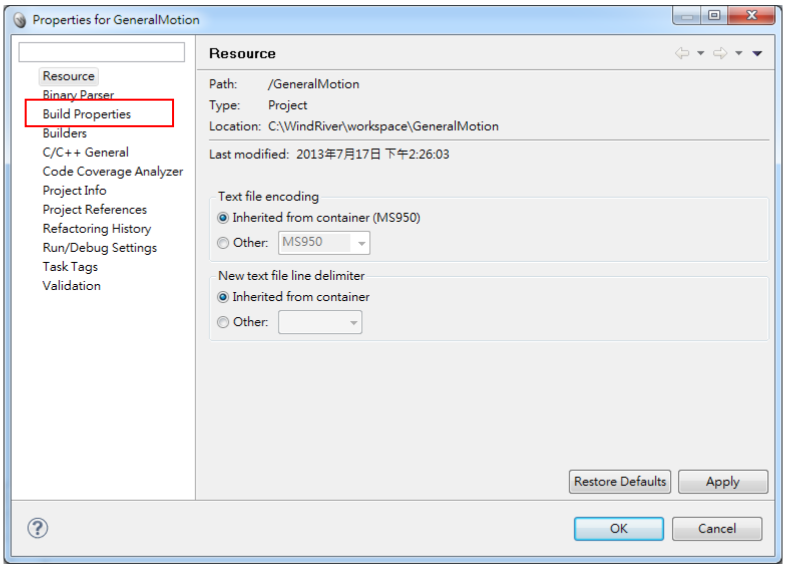
This editor is ready to accept input of source code.

4. Before using the Motion Control Command Library (MCCL), users must first set

up the Library (libIMPDriver_RTP.a and LibRtpMcc.a) and add the linking header file (MCCL_Fun.h).

Set up the Library (libIMPDriver_RTP. and LibRtpMcc.a)

From the Project Explorer, select the project name and right click, and then select 【Properties】 from the pop-up menu to generate the dialog box as shown below：



Resource

Path: /GeneralMotion
Type: Project
Location: C:\WindRiver\workspace\GeneralMotion

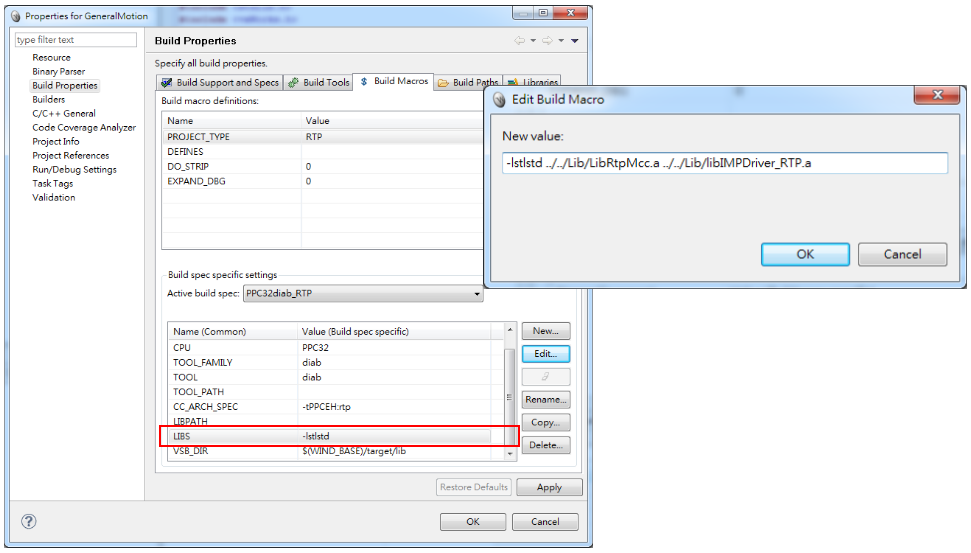
Last modified: 2013年7月17日 下午2:26:03

Text file encoding
Inherited from container (MS950) [✓]
Other: MS950

New text file line delimiter
Inherited from container [✓]
Other:

Fig.2-10 Setting up Library

In the【Properties】dialog box, select Build Properties from the options in the left, and then select the tab “Build Macros”. Select LIBS from the list in the “Build spec specific settings” and press “Edit”. Subsequently, the dialog box as shown below will be created：



Build Properties

Specify all build properties.

Build macro definitions:

Name	Value
PROJECT_TYPE	RTP
DEFINES	
DO_STRIP	0
EXPAND_DBG	0

Build spec specific settings

Active build spec: PPC32diab_RTP

Name (Common)	Value (Build spec specific)
CPU	PPC32
TOOL_FAMILY	diab
TOOL	diab
TOOL_PATH	
CC_ARCH_SPEC	-tPPCEH:rtp
LIBPATH	
LIBS	-lstlstd
VSB_DIR	$(WIND_BASE)/target/lib

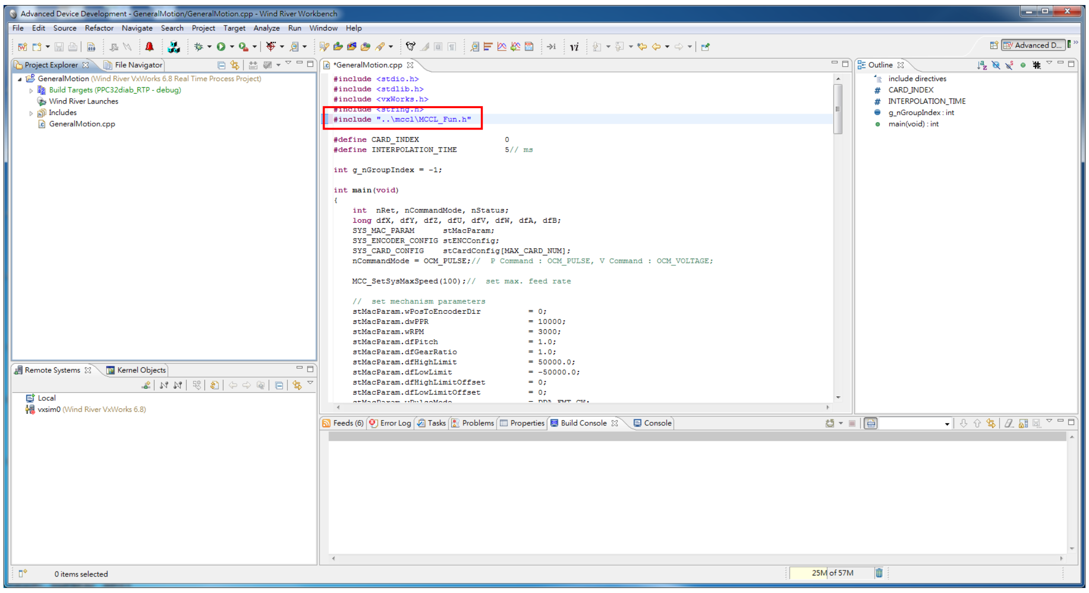
New value:

-lstlstd ../../Lib/LibRtpMcc.a ../../Lib/libIMPDriver_RTP.a

Fig.2-11 Library directory settings

In this example, the two files, namely libIMPDriver_RTP.a and LibRtpMcc.a, are stored in the following directory path: C:\WindRiver\workspace\Lib. In the edit box for New Value, enter the directory path of Library (libIMPDriver_RTP.a and LibRtpMcc.a) and press OK.

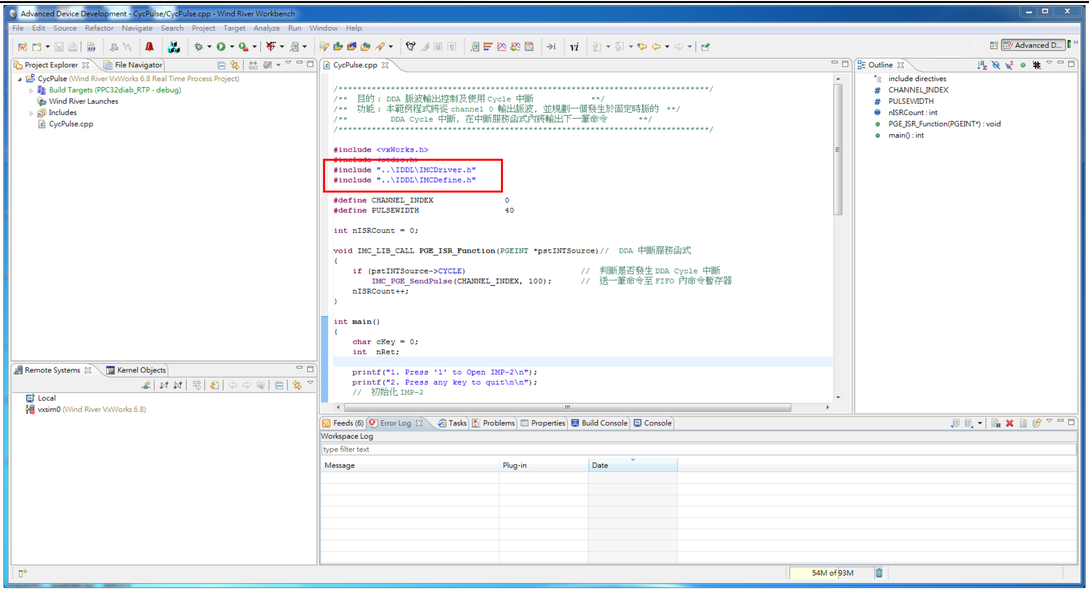
5. To use the MCCL, the MCCL linking header file must be included in the code. In this example, the two files, MCCL_Fun.h and MCCL.h, are stored in the following directory path: C:\WindRiver\workspace\MCCL. Add #include "..\mccl\MCCL_Fun.h" at the beginning of the code, as shown below：



```
#include <stdio.h>
#include <stdlib.h>
#include <vxWorks.h>
#include <string.h>
#include "..\mccl\MCCL_Fun.h"

#define CARD_INDEX                    0
#define INTERPOLATION_TIME            5// ms

int g_nGroupIndex = -1;

int main(void)
{
    int  nRet, nCommandMode, nStatus;
    long dfX, dfY, dfZ, dfU, dfV, dfW, dfA, dfB;
    SYS_MAC_PARAM      stMacParam;
    SYS_ENCODER_CONFIG stENCConfig;
    SYS_CARD_CONFIG    stCardConfig[MAX_CARD_NUM];
    nCommandMode = OCM_PULSE;//  P Command : OCM_PULSE, V Command : OCM_VOLTAGE;

    MCC_SetSysMaxSpeed(100);//  set max. feed rate

    //  set mechanism parameters
    stMacParam.wPosToEncoderDir          = 0;
    stMacParam.dwPPR                     = 10000;
    stMacParam.wRPM                      = 3000;
    stMacParam.dfPitch                   = 1.0;
    stMacParam.dfGearRatio               = 1.0;
    stMacParam.dfHighLimit               = 50000.0;
    stMacParam.dfLowLimit                = -50000.0;
    stMacParam.dfHighLimitOffset         = 0;
    stMacParam.dfLowLimitOffset          = 0;
```

Fig.2-12 Adding MCCL linking header file

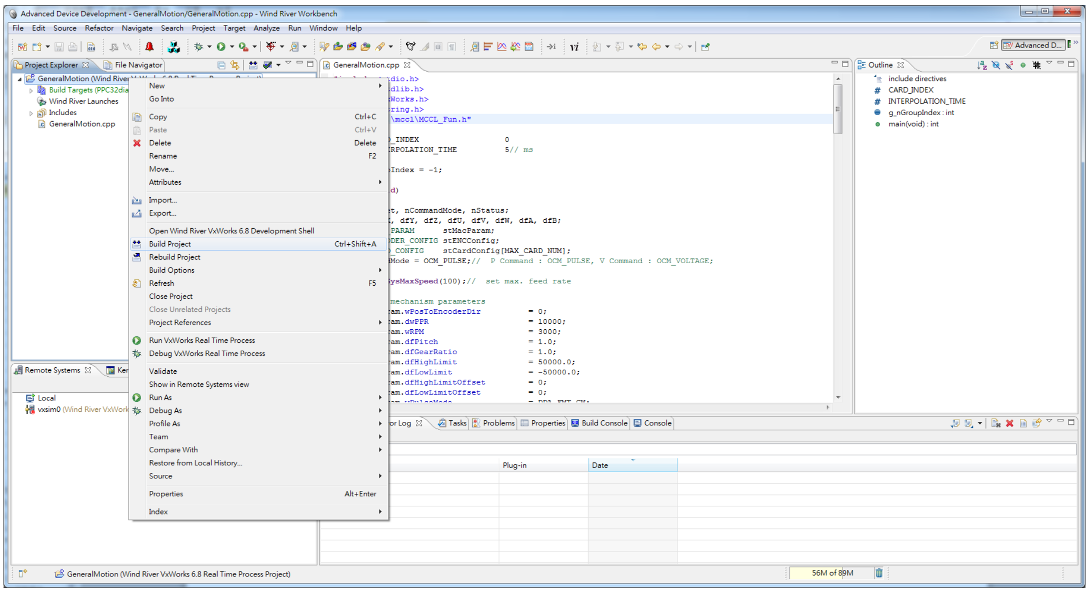
To use the IDDL, the IDDL linking header file must also be included in the code. In this example, the two files, IMCDriver.h and IMCDefine.h, are stored in the following directory path: C:\WindRiver\workspace\IDDL. Add #include "..\IDDL\IMCDriver.h" and #include "..\IDDL\IMCDefine.h" at the beginning of the code, as shown below：



```
/****************************************************************************/
/**   目的：DDA 脈波輸出控制及使用 Cycle 中斷                   **/
/**   功能：本範例程式將從 channel 0 輸出脈波，並規劃一個發生於固定時脈的 **/
/**         DDA Cycle 中斷，在中斷服務函式內將輸出下一筆命令         **/
/****************************************************************************/

#include <vxWorks.h>
#include <stdio.h>
#include "..\IDDL\IMCDriver.h"
#include "..\IDDL\IMCDefine.h"

#define CHANNEL_INDEX              0
#define PULSEWIDTH                 40

int nISRCount = 0;

void IMC_LIB_CALL PGE_ISR_Function(PGEINT *pstINTSource)//  DDA 中斷服務函式
{
    if (pstINTSource->CYCLE)                         //  判斷是否發生 DDA Cycle 中斷
        IMC_PGE_SendPulse(CHANNEL_INDEX, 100);       //  送一筆命令至 FIFO 內命令暫存器
    nISRCount++;
}

int main()
{
    char cKey = 0;
    int  nRet;

    printf("1. Press '1' to Open IMP-2\n");
    printf("2. Press any key to quit\n\n");
    //  初始化 IMP-2
```

Fig.2-13 Adding IDDL linking header file

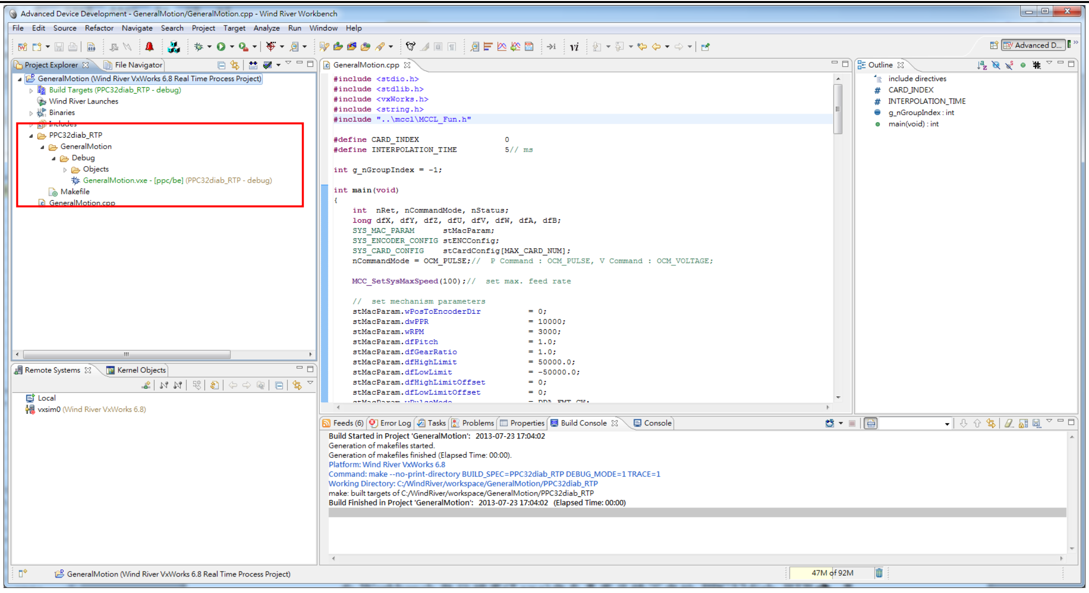
6. After completing the code, select the project name and right-click, then select 【Build Project】 from the pop-up menu to create the project, as shown below：



```
dio.h>
dlib.h>
Works.h>
ring.h>
\mccl\MCCL_Fun.h"

_INDEX                   0
RPOLATION_TIME           5// ms

Index = -1;

d)

t, nCommandMode, nStatus;
, dfY, dfZ, dfU, dfV, dfW, dfA, dfB;
PARAM       stMacParam;
DER_CONFIG stENCConfig;
_CONFIG     stCardConfig[MAX_CARD_NUM];
Mode = OCM_PULSE;//  P Command : OCM_PULSE, V Command : OCM_VOLTAGE;

ysMaxSpeed(100);//  set max. feed rate

mechanism parameters
am.wPosToEncoderDir          = 0;
am.dwPPR                     = 10000;
am.wRPM                      = 3000;
am.dfPitch                   = 1.0;
am.dfGearRatio               = 1.0;
am.dfHighLimit               = 50000.0;
am.dfLowLimit                = -50000.0;
am.dfHighLimitOffset         = 0;
am.dfLowLimitOffset          = 0;
```

Fig.2-14 Build Project

When the file is created while building the project without error, a new folder called PPC32diab_RTP will appear in the project, which contains the files generated after building the project, as shown in the following figure：

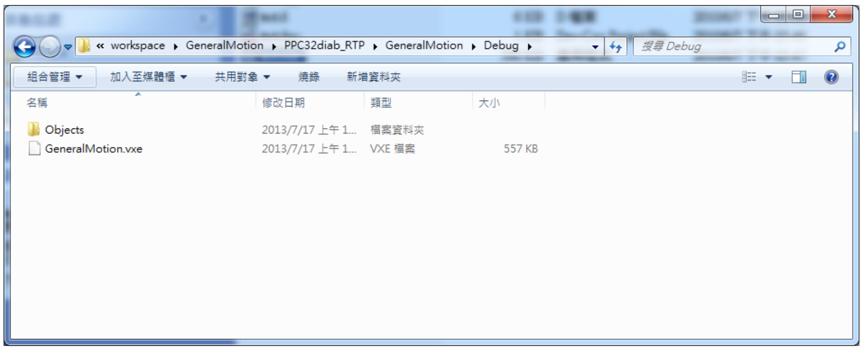

Advanced Device Development - GeneralMotion/GeneralMotion.cpp - Wind River Workbench
File Edit Source Refactor Navigate Search Project Target Analyze Run Window Help
Project Explorer
File Navigator
GeneralMotion (Wind River VxWorks 6.8 Real Time Process Project)
Build Targets (PPC32diab_RTP - debug)
Wind River Launches
Binaries
Includes
PPC32diab_RTP
GeneralMotion
Debug
Objects
GeneralMotion.vxe - [ppc/be] (PPC32diab_RTP - debug)
Makefile
GeneralMotion.cpp
Remote Systems
Kernel Objects
Local
vxsim0 (Wind River VxWorks 6.8)
GeneralMotion.cpp
Outline
include directives
CARD_INDEX
INTERPOLATION_TIME
g_nGroupIndex : int
main(void) : int
Feeds (6) Error Log Tasks Problems Properties Build Console Console
GeneralMotion (Wind River VxWorks 6.8 Real Time Process Project)
47M of 92M

```
#include <stdio.h>
#include <stdlib.h>
#include <vxWorks.h>
#include <string.h>
#include "..\mccl\MCCL_Fun.h"

#define CARD_INDEX                  0
#define INTERPOLATION_TIME          5// ms

int g_nGroupIndex = -1;

int main(void)
{
    int  nRet, nCommandMode, nStatus;
    long dfX, dfY, dfZ, dfU, dfV, dfW, dfA, dfB;
    SYS_MAC_PARAM      stMacParam;
    SYS_ENCODER_CONFIG stENCConfig;
    SYS_CARD_CONFIG    stCardConfig[MAX_CARD_NUM];
    nCommandMode = OCM_PULSE;//  P Command : OCM_PULSE, V Command : OCM_VOLTAGE;

    MCC_SetSysMaxSpeed(100);//  set max. feed rate

    //  set mechanism parameters
    stMacParam.wPosToEncoderDir          = 0;
    stMacParam.dwPPR                     = 10000;
    stMacParam.wRPM                      = 3000;
    stMacParam.dfPitch                   = 1.0;
    stMacParam.dfGearRatio               = 1.0;
    stMacParam.dfHighLimit               = 50000.0;
    stMacParam.dfLowLimit                = -50000.0;
    stMacParam.dfHighLimitOffset         = 0;
    stMacParam.dfLowLimitOffset          = 0;
```

```
Build Started in Project 'GeneralMotion':  2013-07-23 17:04:02
Generation of makefiles started.
Generation of makefiles finished (Elapsed Time: 00:00).
Platform: Wind River VxWorks 6.8
Command: make --no-print-directory BUILD_SPEC=PPC32diab_RTP DEBUG_MODE=1 TRACE=1
Working Directory: C:/WindRiver/workspace/GeneralMotion/PPC32diab_RTP
make: built targets of C:/WindRiver/workspace/GeneralMotion/PPC32diab_RTP
Build Finished in Project 'GeneralMotion':  2013-07-23 17:04:02  (Elapsed Time: 00:00)
```

Fig.2-15 Build Finish

The Workbench execution file (*.vxe) will be generated under the project directory of PPC32diab_RTP➔ project name ➔ Debug.


« workspace ▸ GeneralMotion ▸ PPC32diab_RTP ▸ GeneralMotion ▸ Debug ▸
搜尋 Debug
組合管理 ▾ 加入至媒體櫃 ▾ 共用對象 ▾ 燒錄 新增資料夾

名稱	修改日期	類型	大小
Objects	2013/7/17 上午 1...	檔案資料夾	
GeneralMotion.vxe	2013/7/17 上午 1...	VXE 檔案	557 KB

Fig.2-16 Generating Workbench execution file (*.vxe)

7. Sample code as a reference：

   GoHome：Returning to Home

   GeneralMotion：General motion

   MutiGroup：Multiple groups

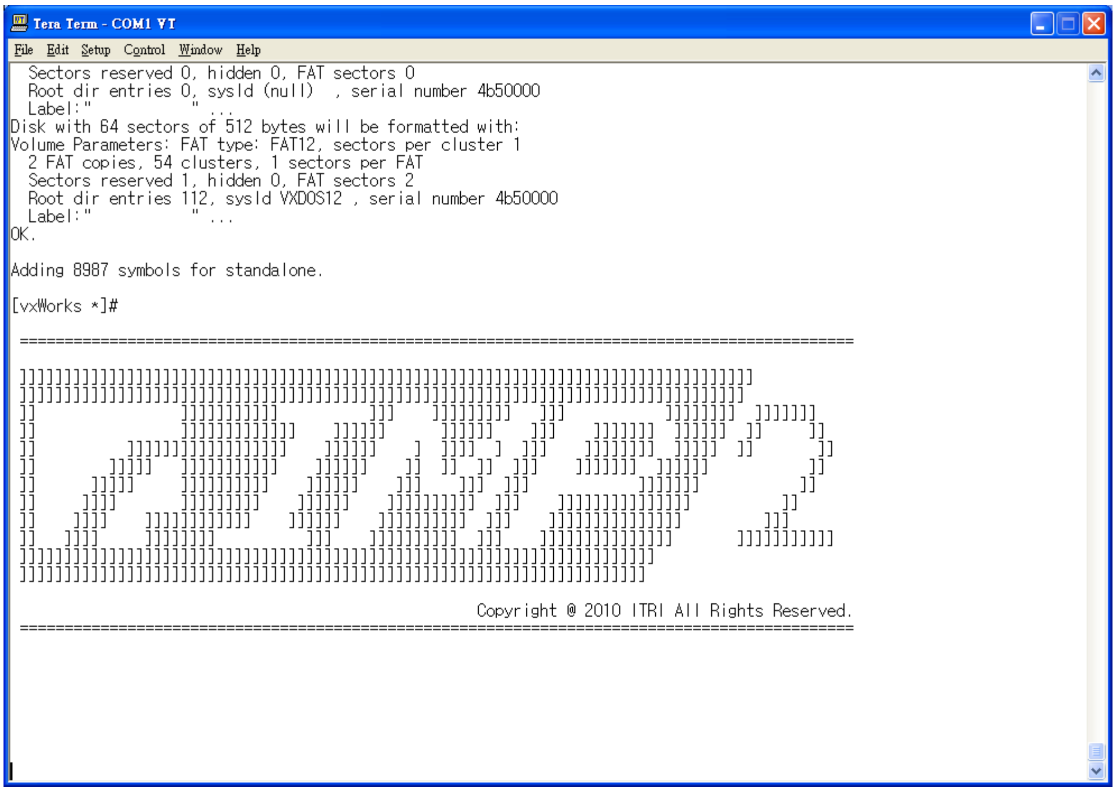
   ENCCompare：Encoder counting value triggers interrupt service function

The file (.vxe) generated during the build is transmitted to the IMP-2 via internet and executed. The following steps are required：

- Step 1： Please make sure that the IMP-2 connects to the 5V, +12V and -12V power supplies, and the RS-232 cable, internet cable and SCSI-II 100PIN and 68PIN cable are all connected.

  Attention： A null modem adapter is required between the IMP RS232 connector and the computer RS232 connector.

- Step 2：Launch a terminal software (ex: Tera Term) with the default serial setting(Buadrate=15200, no parity, 8bit data,1 stop bit), and then turn on the power supply of IMP-2. At this moment, the terminal connection window displays the IMP-2 start-up screen, indicating that the IMP-2 has been turned on successfully. In the meantime, the eight LEDs of the IMP-2 will also be turned on as shown in Fig. 2-18.



```
  Sectors reserved 0, hidden 0, FAT sectors 0
  Root dir entries 0, sysId (null)  , serial number 4b50000
  Label:"            " ...
Disk with 64 sectors of 512 bytes will be formatted with:
Volume Parameters: FAT type: FAT12, sectors per cluster 1
  2 FAT copies, 54 clusters, 1 sectors per FAT
  Sectors reserved 1, hidden 0, FAT sectors 2
  Root dir entries 112, sysId VXDOS12 , serial number 4b50000
  Label:"            " ...
OK.

Adding 8987 symbols for standalone.

[vxWorks *]#

 ==============================================================================================

 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
 ]]                ]]]]]]]]]]]          ]]]    ]]]]]]]]]    ]]]           ]]]]]]]]  ]]]]]]]
 ]]                ]]]]]]]]]]]]]    ]]]]]]      ]]]]]]     ]]]   ]]]]]]]   ]]]]]] ]]      ]]
 ]]          ]]]]]]]]]]]]]]]]]]    ]]]]]]     ] ]]]]   ] ]]]    ]]]]]]]]  ]]]]]   ]]       ]]
 ]]        ]]]]]   ]]]]]]]]]]]    ]]]]]]    ]]  ]]  ]]  ]]]    ]]]]]]]  ]]]]]]            ]]
 ]]      ]]]]]     ]]]]]]]]]]    ]]]]]]    ]]]     ]]]  ]]]            ]]]]]]]           ]]
 ]]     ]]]]       ]]]]]]]]]    ]]]]]]    ]]]]]]]]]]   ]]]    ]]]]]]]]]]]]]]]          ]]
 ]]    ]]]]    ]]]]]]]]]]]]    ]]]]]]    ]]]]]]]]]]   ]]]    ]]]]]]]]]]]]]]]         ]]]
 ]]   ]]]]     ]]]]]]]]          ]]]    ]]]]]]]]]]   ]]]    ]]]]]]]]]]]]]]]       ]]]]]]]]]]]
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]

                                                    Copyright @ 2010 ITRI All Rights Reserved.
 ==============================================================================================
```

Fig.2-17 IMP-2 start-up screen

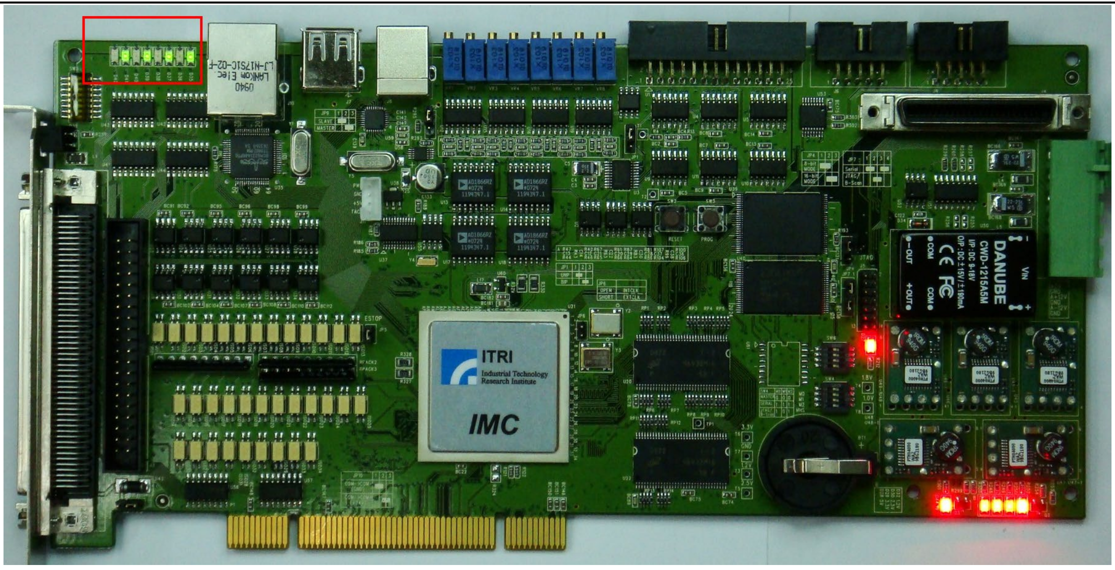
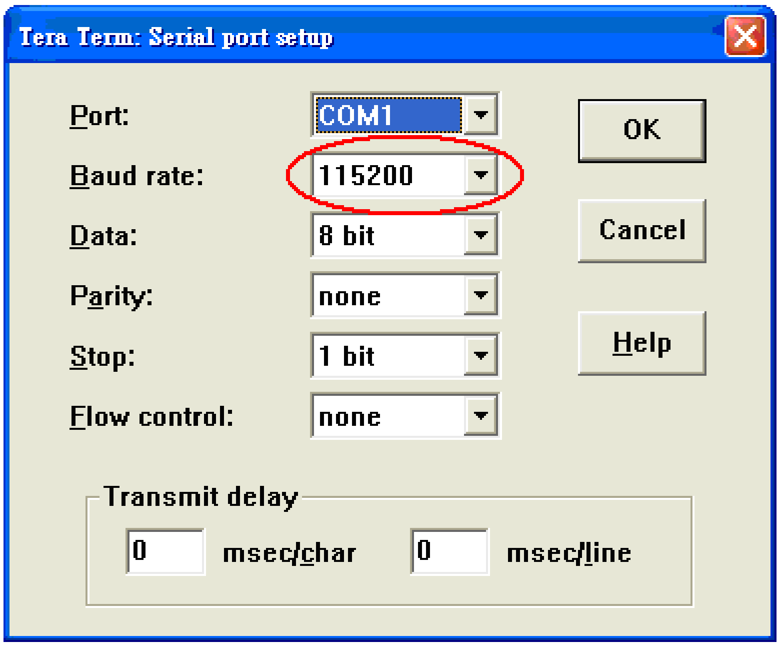


Fig.2-18 LED status when IMP-2 is turned on



Tera Term: Serial port setup

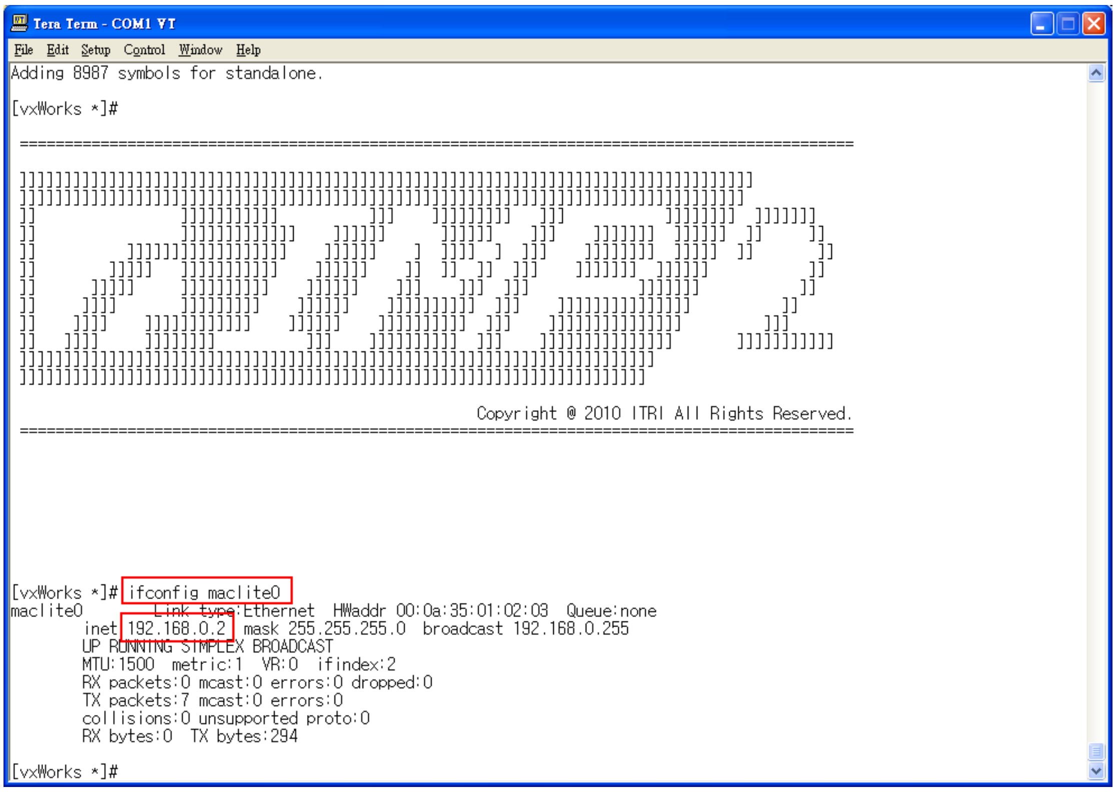
Port: COM1
Baud rate: 115200
Data: 8 bit
Parity: none
Stop: 1 bit
Flow control: none

Transmit delay
0 msec/char 0 msec/line

Fig.2-19 RS232 parameter setting for Tera Term

➢ Step 3： For network connection settings, in the terminal software and under the Command Interpreter mode (Note 2), type “ifconfig maclite0” (maclite0 is the network interface card name) to obtain the current IP address of IMP-2. Default IP address: 192.168.0.2.

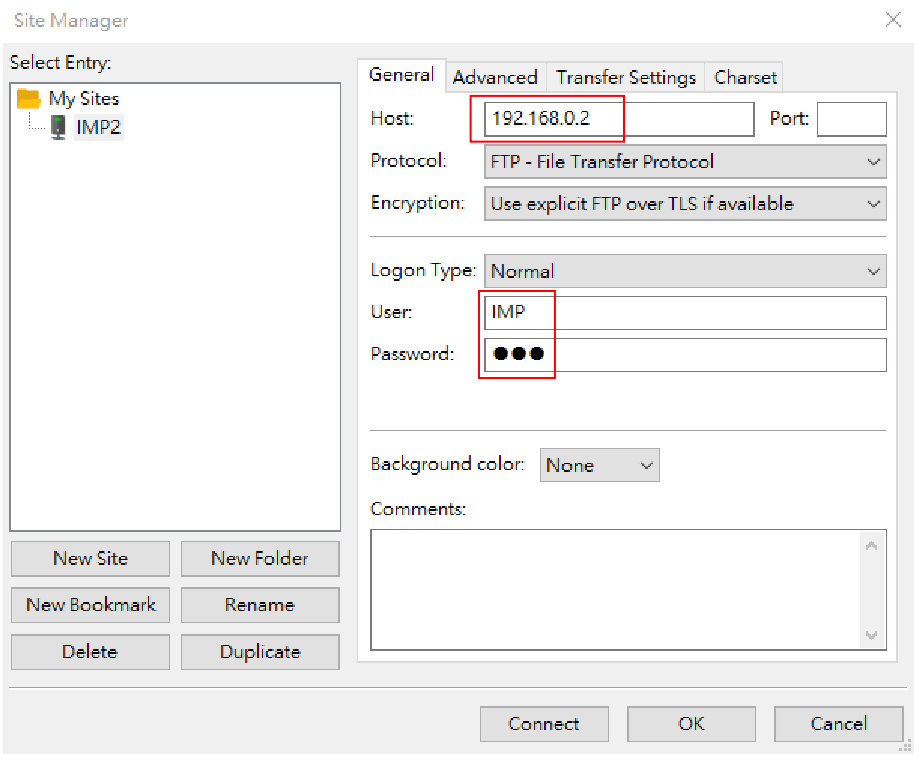
Note 2: For more detail about Command Interpreter mode, please refer to section 4.2.



```
Adding 8987 symbols for standalone.

[vxWorks *]#

 ========================================================================================

 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
 ]]               ]]]]]]]]]]          ]]]   ]]]]]]]]]   ]]]          ]]]]]]]] ]]]]]]]
 ]]               ]]]]]]]]]]]]    ]]]]]]     ]]]]]]    ]]]   ]]]]]]]  ]]]]]] ]]     ]]
 ]]         ]]]]]]]]]]]]]]]]]    ]]]]]]    ] ]]]]  ]  ]]]   ]]]]]]]]  ]]]]] ]]       ]]
 ]]       ]]]]]   ]]]]]]]]]]    ]]]]]]    ]] ]]  ]]  ]]]   ]]]]]]]  ]]]]]]          ]]
 ]]      ]]]]]    ]]]]]]]]]    ]]]]]]    ]]]   ]]]  ]]]           ]]]]]]]          ]]
 ]]     ]]]]      ]]]]]]]]]   ]]]]]]    ]]]]]]]]]  ]]]    ]]]]]]]]]]]]]]         ]]
 ]]    ]]]]   ]]]]]]]]]]]    ]]]]]]    ]]]]]]]]]  ]]]    ]]]]]]]]]]]]]]        ]]]
 ]]   ]]]]    ]]]]]]]          ]]]    ]]]]]]]]]  ]]]    ]]]]]]]]]]]]]]      ]]]]]]]]]]]
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]

                                                 Copyright @ 2010 ITRI All Rights Reserved.
 ========================================================================================



[vxWorks *]# ifconfig maclite0
maclite0       Link type:Ethernet  HWaddr 00:0a:35:01:02:03  Queue:none
        inet 192.168.0.2  mask 255.255.255.0  broadcast 192.168.0.255
        UP RUNNING SIMPLEX BROADCAST
        MTU:1500  metric:1  VR:0  ifindex:2
        RX packets:0 mcast:0 errors:0 dropped:0
        TX packets:7 mcast:0 errors:0
        collisions:0 unsupported proto:0
        RX bytes:0  TX bytes:294

[vxWorks *]#
```

Fig.2-20 Obtaining the IP address of IMP-2

➢ Step 4：Launch “FileZilla” on the desktop and click the station administrator icon at the top left of the screen. Make sure the IP address of host is correct and select the connection to IMP to upload the execution file of the work program to IMP-2. Default user account/password: IMP/IMP.



Site Manager

Select Entry:

My Sites
IMP2

General | Advanced | Transfer Settings | Charset

Host: 192.168.0.2 Port:

Protocol: FTP - File Transfer Protocol

Encryption: Use explicit FTP over TLS if available

Logon Type: Normal

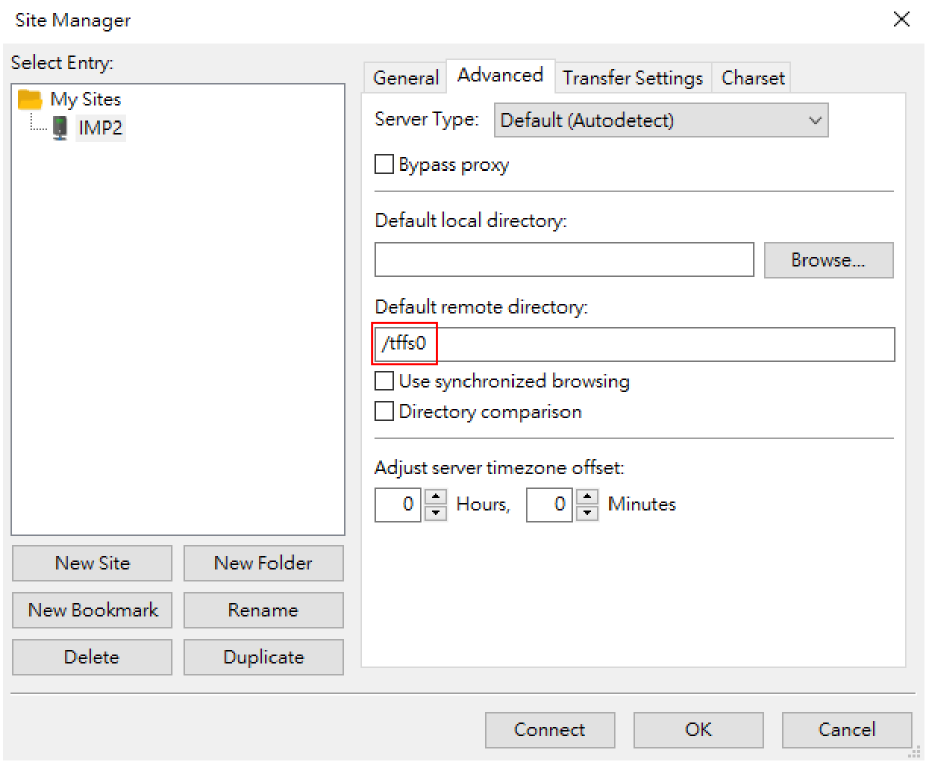
User: IMP

Password: ●●●

Background color: None

Comments:

Fig.2-21 Setting up the IP address of IMP-2



Site Manager

Select Entry:

My Sites
IMP2

General | Advanced | Transfer Settings | Charset

Server Type: Default (Autodetect)

☐ Bypass proxy

Default local directory:

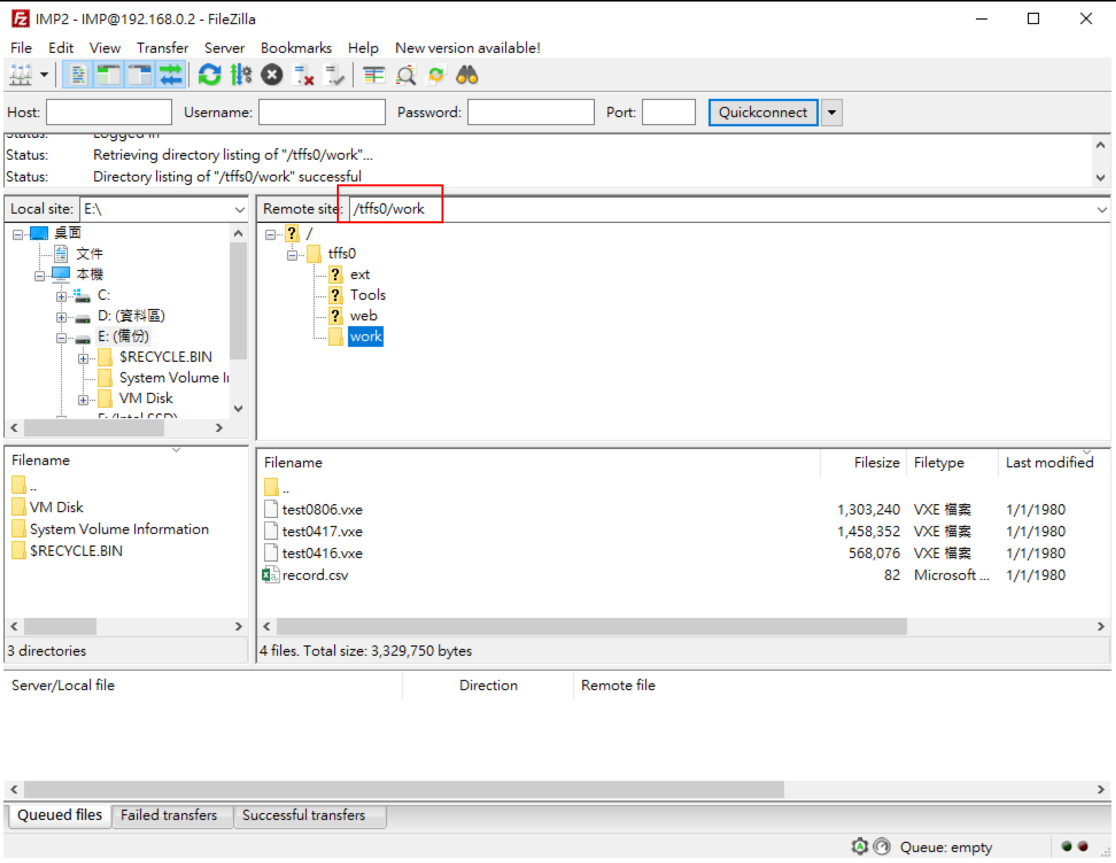
Default remote directory: /tffs0

☐ Use synchronized browsing

☐ Directory comparison

Adjust server timezone offset: 0 Hours, 0 Minutes

Fig.2-22 Setting up the Flash file folder for the connection to IMP-2



```
Status:    Retrieving directory listing of "/tffs0/work"...
Status:    Directory listing of "/tffs0/work" successful
```

Filename
..
VM Disk
System Volume Information
$RECYCLE.BIN

3 directories

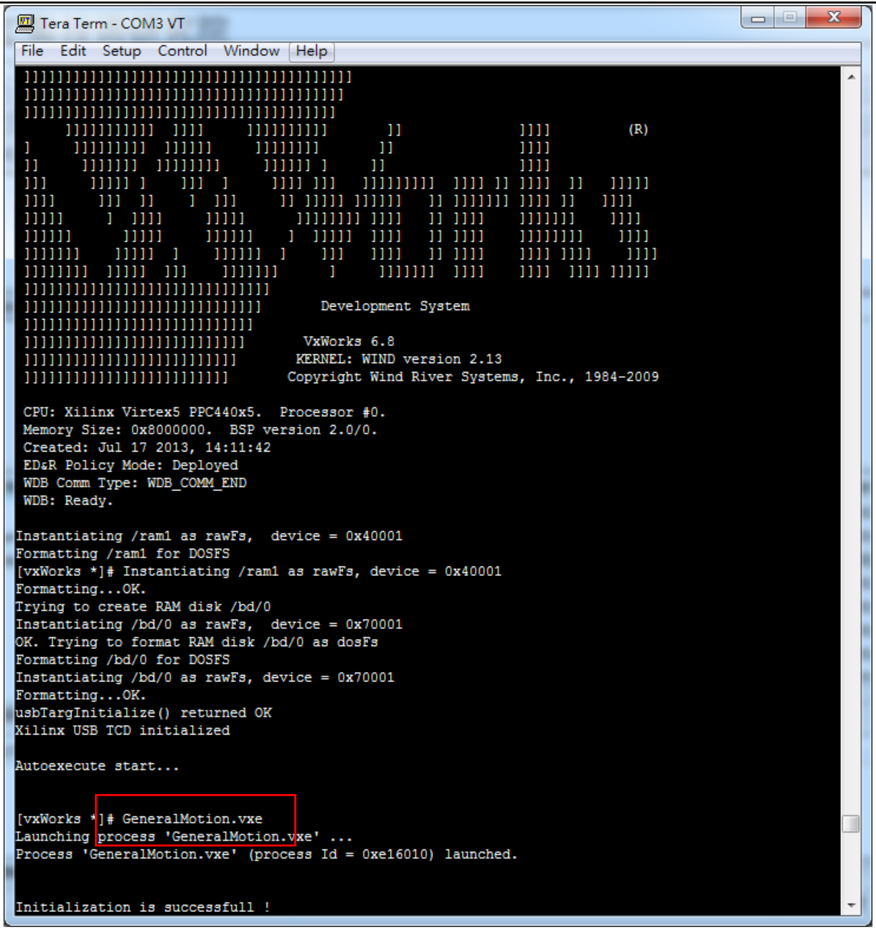
Filename	Filesize	Filetype	Last modified
..			
test0806.vxe	1,303,240	VXE 檔案	1/1/1980
test0417.vxe	1,458,352	VXE 檔案	1/1/1980
test0416.vxe	568,076	VXE 檔案	1/1/1980
record.csv	82	Microsoft ...	1/1/1980

4 files. Total size: 3,329,750 bytes

Fig.2-23 Uploading the execution file (*.vxe) of the work program to IMP-2

➢ Step 5：Go back to the terminal software and under the Command Interpreter mode, type [name of the execution file.vxe]* to execute the work program.

* If the work program is not in the following directory: /tffs0/work, please add the pathname, or change the working directory to the directory where the file is stored, and then type the file name to execute the program.



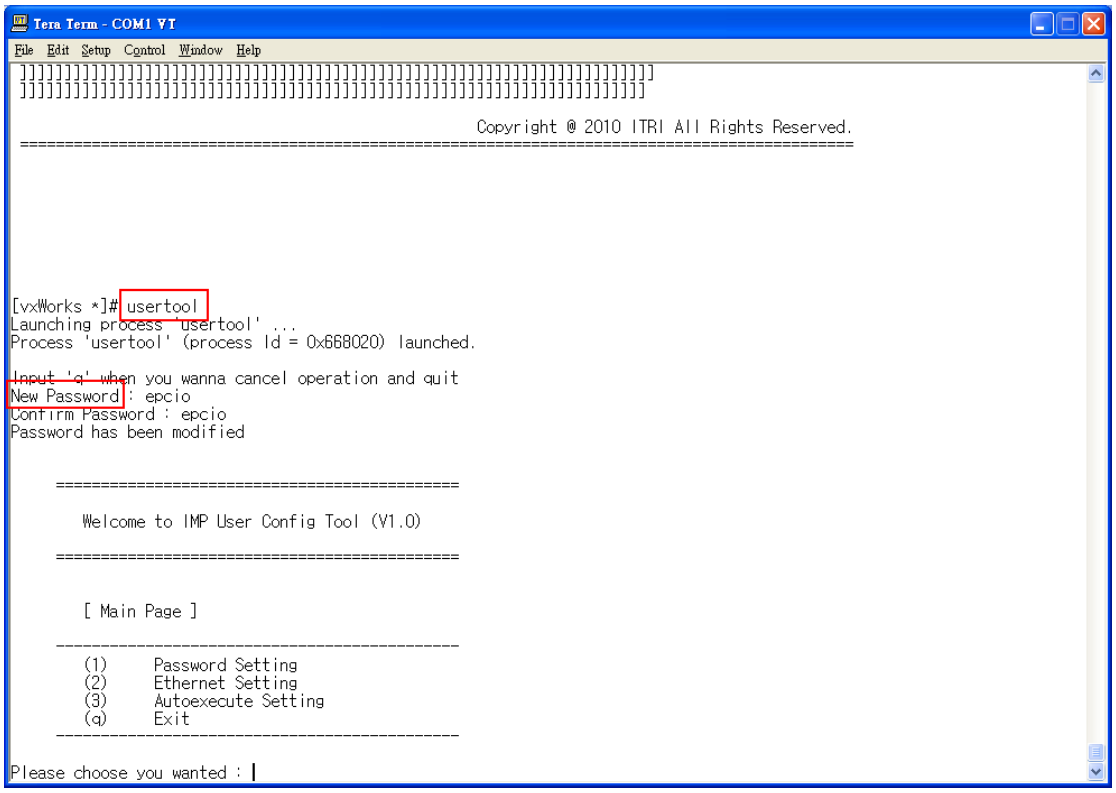
```
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
      ]]]]]]]]]]]  ]]]]     ]]]]]]]]]]       ]]              ]]]]         (R)
 ]     ]]]]]]]]]  ]]]]]]     ]]]]]]]]       ]]               ]]]]
 ]]     ]]]]]]]  ]]]]]]]]     ]]]]]] ]     ]]                ]]]]
 ]]]     ]]]]] ]    ]]]  ]     ]]]] ]]]   ]]]]]]]]]  ]]]] ]] ]]]]  ]]   ]]]]]
 ]]]]     ]]]  ]]    ]  ]]]     ]] ]]]]] ]]]]]]   ]] ]]]]]]] ]]]] ]]   ]]]]
 ]]]]]     ]  ]]]]     ]]]]]      ]]]]]]]] ]]]]   ]] ]]]]    ]]]]]]]    ]]]]
 ]]]]]]      ]]]]]     ]]]]]]    ]  ]]]]]  ]]]]   ]] ]]]]    ]]]]]]]]    ]]]]
 ]]]]]]]    ]]]]]  ]    ]]]]]]  ]    ]]]   ]]]]   ]] ]]]]    ]]]] ]]]]    ]]]]
 ]]]]]]]]  ]]]]]  ]]]    ]]]]]]]      ]     ]]]]]]]  ]]]]    ]]]]  ]]]] ]]]]]
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]       Development System
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]       VxWorks 6.8
 ]]]]]]]]]]]]]]]]]]]]]]]]]]       KERNEL: WIND version 2.13
 ]]]]]]]]]]]]]]]]]]]]]]]]]       Copyright Wind River Systems, Inc., 1984-2009

 CPU: Xilinx Virtex5 PPC440x5.  Processor #0.
 Memory Size: 0x8000000.  BSP version 2.0/0.
 Created: Jul 17 2013, 14:11:42
 ED&R Policy Mode: Deployed
 WDB Comm Type: WDB_COMM_END
 WDB: Ready.

Instantiating /ram1 as rawFs,  device = 0x40001
Formatting /ram1 for DOSFS
[vxWorks *]# Instantiating /ram1 as rawFs, device = 0x40001
Formatting...OK.
Trying to create RAM disk /bd/0
Instantiating /bd/0 as rawFs,  device = 0x70001
OK. Trying to format RAM disk /bd/0 as dosFs
Formatting /bd/0 for DOSFS
Instantiating /bd/0 as rawFs, device = 0x70001
Formatting...OK.
usbTargInitialize() returned OK
Xilinx USB TCD initialized

Autoexecute start...


[vxWorks *]# GeneralMotion.vxe
Launching process 'GeneralMotion.vxe' ...
Process 'GeneralMotion.vxe' (process Id = 0xe16010) launched.


Initialization is successfull !
```

Fig.2-24 Executing work program

3. IMP User Configuration Tool

The IMP User Configuration Tool provides users with the ability to set up network connections (Ethernet) and edit auto-execution in standalone mode. The steps for using this tool are as follows：

1. After the auto-execution is completed,
   - ➢ Type [usertool] to execute the IMP User Config Tool



```
]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]

                                                     Copyright @ 2010 ITRI All Rights Reserved.
=================================================================================================





[vxWorks *]# usertool
Launching process 'usertool' ...
Process 'usertool' (process Id = 0x668020) launched.

Input 'q' when you wanna cancel operation and quit
New Password : epcio
Confirm Password : epcio
Password has been modified


     ==============================================

        Welcome to IMP User Config Tool (V1.0)

     ==============================================


        [ Main Page ]

     ----------------------------------------------
        (1)     Password Setting
        (2)     Ethernet Setting
        (3)     Autoexecute Setting
        (q)     Exit
     ----------------------------------------------

Please choose you wanted : |
```

Fig.3-1 Main menu screen

2. When the IMP User Configuration Tool is used for the first time, users need to set the login password. After the setting is completed, the main menu screen will be displayed. The options that can be configured by users changing passwords, Ethernet settings, and auto-execute settings.
3. Password Setting

   The password can be changed, and its length must be longer than 4 characters.
4. Ethernet Setting
   - ■ Menu for Ethernet setting：(*) Type the character in parentheses to edit

     (1) Set IP Address

     (2) Set Mac Address

(3) Set IP Mask

(4) Get IP Address

(5) Get Mac Address

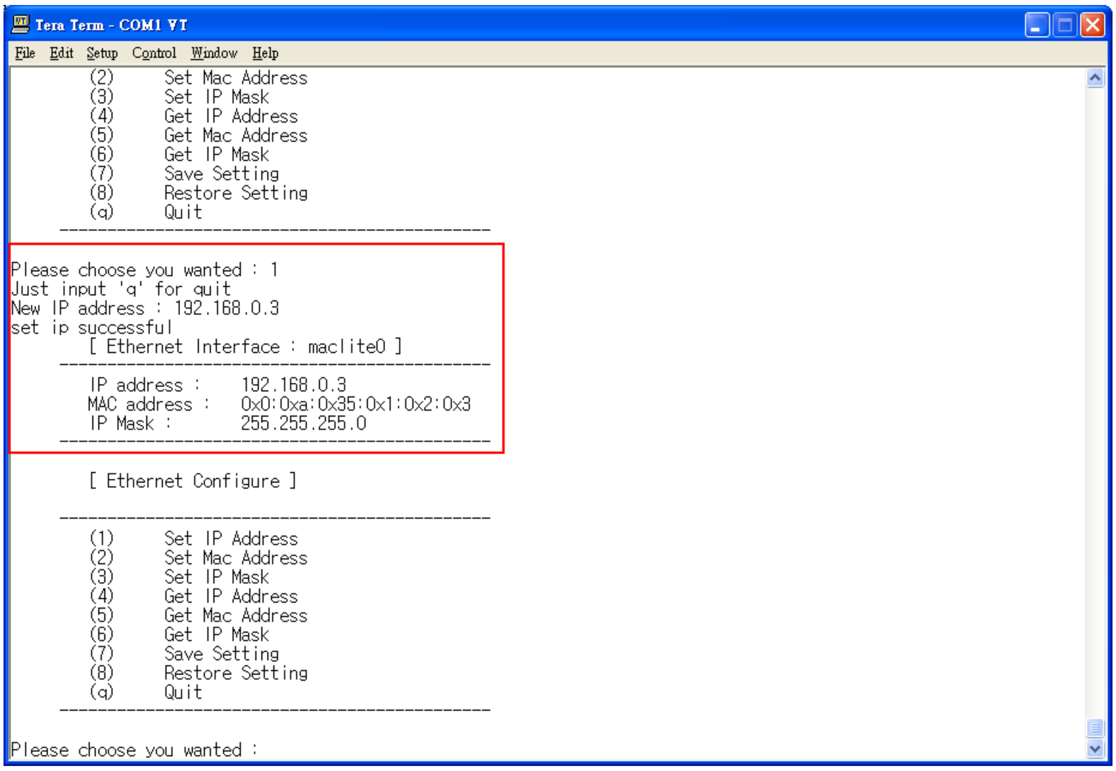
(6) Get IP Mask

(7) Save Setting

(8) Restore Setting

(q) Quit

If more than two IMP-2s are connected at the same time, users must set the IP address and the Mac address for each IMP-2 separately as different addresses.



```
         (2)      Set Mac Address
         (3)      Set  IP Mask
         (4)      Get  IP Address
         (5)      Get Mac Address
         (6)      Get  IP Mask
         (7)      Save Setting
         (8)      Restore Setting
         (q)      Quit
      ---------------------------------------------

Please choose you wanted : 1
Just input 'q' for quit
New IP address : 192.168.0.3
set ip successful
         [ Ethernet Interface : maclite0 ]
      ---------------------------------------------
         IP address :      192.168.0.3
         MAC address :     0x0:0xa:0x35:0x1:0x2:0x3
         IP Mask :         255.255.255.0
      ---------------------------------------------

         [ Ethernet Configure ]

      ---------------------------------------------
         (1)      Set  IP Address
         (2)      Set Mac Address
         (3)      Set  IP Mask
         (4)      Get  IP Address
         (5)      Get Mac Address
         (6)      Get  IP Mask
         (7)      Save Setting
         (8)      Restore Setting
         (q)      Quit
      ---------------------------------------------

Please choose you wanted :
```

Fig.3-2 Setting IP address

5. Auto-execute Setting

■ Menu for auto-execute setting：(*) Type the character in parentheses to edit

(l) list table:

Display auto-execution table

(s) save table to the config file:

Save auto-execution table to the configuration file

(a) access config file to fill table:

Access configuration file to fill the auto-execution table

(u) move the arrow up:

Move up the selection arrow of the auto-execution table

(d) move the arrow down:

Move down the selection arrow of the auto-execution table

(U) move selected program up:

Move the selected program up

(D) move selected program down:

Move the selected program down

(p) plus program to table:

Add work program name to the auto-execution table

(m) minus program to table:

Remove the selected program from the auto-execution table

(r) replace program:

Replace the selected program

(i) insert program:

Insert selected program to the auto-execution table

(h) help:

Display function descriptions

(q) quit:

Quit auto-execute setting

(e) execute:

Execute the selected program

(x) execute by order:

Execute all programs in the auto-execution table

(c) clear table:

Clear the auto-execution table

■ The use of control function：

When editing the auto-execution table, the execution order of each work program in the auto-execution table is controlled by inserting the control function (insert program).

➢ @Sleep(specified time (ms))—When this function is executed, the execution of subsequent work programs will be suspended. When the specified time has ended, the execution of subsequent work programs will continue.

- @Wait()—When this function is executed, the execution of subsequent work program execution will be suspended. The following work program will be executed after the previous work program is completed.

Example1:

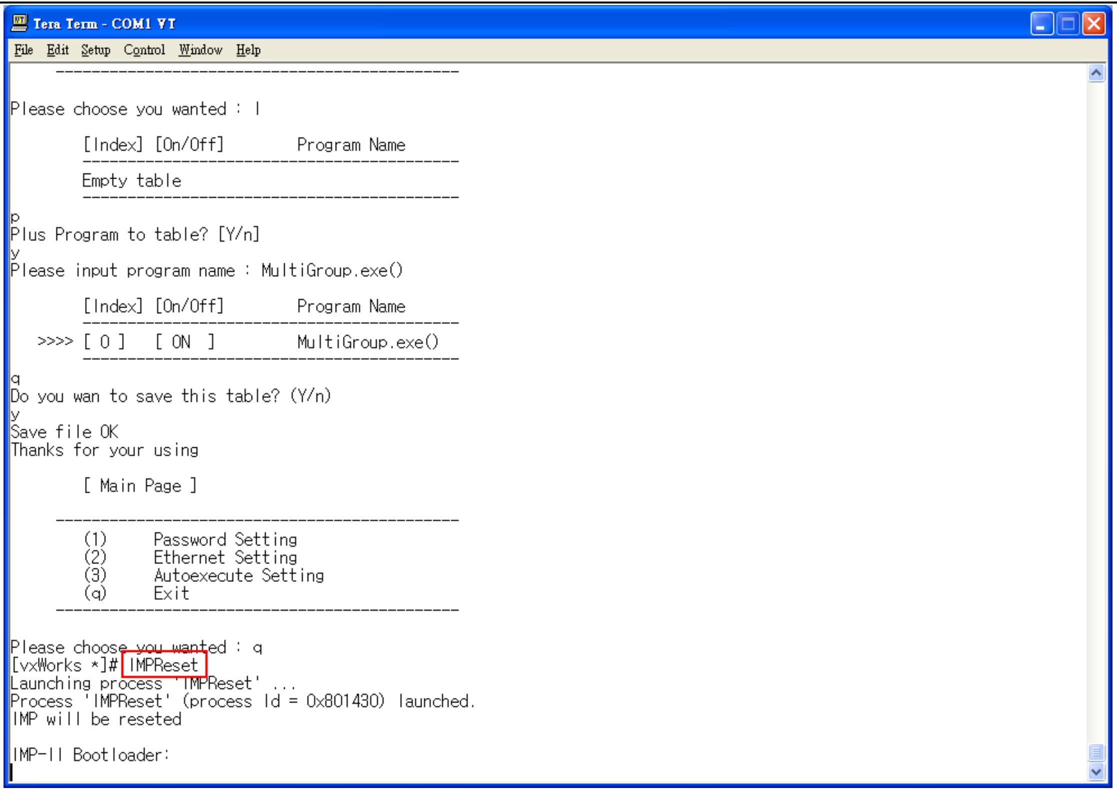
The execution procedure of Example 1 is to start Demo1 after booting, and then count for 2 seconds before starting Demo2.

```
          [Index] [On/Off]          Program Name
          ------------------------------------------
    >>>> [ 0 ]    [ ON   ]           Demo1.vxe()
          [ 1 ]    [ ON   ]            @Sleep(2000)
          [ 2 ]    [ ON   ]            Demo2.vxe()
          ------------------------------------------
```

Example2:

The execution procedure of Example 2 is to start Demo1 and Demo2 simultaneously after booting, and then wait for Demo2 to complete before starting Demo3.

```
          [Index] [On/Off]          Program Name
          ------------------------------------------
    >>>> [ 0 ]    [ ON   ]           Demo1.vxe()
          [ 1 ]    [ ON   ]            Demo2.vxe()
          [ 2 ]    [ ON   ]            @Wait()
          [ 3 ]    [ ON   ]            Demo3.vxe()
          ------------------------------------------
```



Tera Term - COM1 VT

File Edit Setup Control Window Help

```
     ---------------------------------------------

Please choose you wanted : l

        [Index]  [On/Off]        Program Name
        ------------------------------------------
        Empty table
        ------------------------------------------
p
Plus Program to table? [Y/n]
y
Please input program name : MultiGroup.exe()

        [Index]  [On/Off]        Program Name
        ------------------------------------------
   >>>> [ 0 ]    [ ON  ]         MultiGroup.exe()
        ------------------------------------------
q
Do you wan to save this table? (Y/n)
y
Save file OK
Thanks for your using

        [ Main Page ]

     ---------------------------------------------
        (1)     Password Setting
        (2)     Ethernet Setting
        (3)     Autoexecute Setting
        (q)     Exit
     ---------------------------------------------

Please choose you wanted : q
[vxWorks *]# IMPReset
Launching process 'IMPReset' ...
Process 'IMPReset' (process Id = 0x801430) launched.
IMP will be reseted

IMP-II Bootloader:
```

Fig.3-3 Reset after editing the auto-execution table

After editing the auto-execution table, type [IMPReset] to reset the motion control platform. After the booting is completed, the auto-execution table set by the user will be executed.

4. Embedded Operating Environment

This Section mainly introduces the kernel shell, the kernel object module loader, the debug tool, and the system symbol table.

4.1 Kernel Shell

VxWorks offers multi-user and multi-task operating environment to execute kernel shell simultaneously via terminal, Telnet or rlogin.

VxWorks offers users with two modes of operation:

1. C Language Interpreter Mode

   C language expression and debugging can be performed under this mode.

   C language interpreter mode command prompt symbol：－＞

2. Command Interpreter Mode

An interpreter mode is similar to Unix mode, which can be used to debug and monitor system resources.

Command interpreter mode command prompt symbol：[vxWorks *]#

4.1.1 C Language Interpreter and Command Interpreter

The kernel shell mainly includes two modes of operation, namely C language interpreter mode and command interpreter mode, and their significant differences are:

- The command interpreter mode is mainly used to perform starting, monitoring, and debugging operations on the RTP (Real-Time Process) application. It can also serve as the kernel object module loader to link or remove the kernel object module. Its operating environment is similar to that of Unix.
- The C language interpreter mode is mainly used to monitor and debug the kernel program. It can also serve as the kernel object module loader to execute, link, or remove the kernel object module. Besides, it provides APIs to start and monitor RTP applications. The command is used in a similar way to the C language function.

Switch between modes of operation：

Allow users to switch between the two modes of operation quickly. In C language interpreter mode, use **cmd** command to switch the operation mode to command interpreter mode. Conversely, **C** command can be used to switch from command interpreter mode to C language interpreter mode.

－＞ **cmd**

[vxWorks *]# **C**

－＞

Different modes can be called within each other to assist in executing commands.

[vxWorks]# **C test = malloc(100); test[0] = 10; test[1] = test[0] + 2**

4.1.2 Kernel shell control characters

Control characters	Description
Ctrl+C	Aborts and restarts the shell. However, if a process is launched with the command interpreter

	(using **rtp exec**), the function of **CTRL+C** changes. It is used to interrupt the process.
Ctrl+D	Logs out when the terminal cursor is at the beginning of a line.
Ctrl+H	Deletes a character (backspace).
Ctrl+Q	Resumes output.
Ctrl+S	Temporarily suspends output.
Ctrl+U	Deletes an entire line.
Ctrl+W	If a process is launched with the command interpreter (using **rtp exec**), this key sequence suspends the process running in the foreground.
Ctrl+X	Reboots (trap to the ROM monitor).
ESC	Toggles between input mode and edit mode (**vi** mode only).

4.2 Command Interpreter Mode

This mode is command-oriented and does not understand the syntax of C language. The command is a combination of one or more strings, which include commands, command options and command parameters, etc.

The command syntax is as follows：

Command [subcommand [subcommand…]] [option] [argument] [;]

Command and subcommand are strings of letter combination and cannot contain spaces. An argument can be any string.

For example：

```
[vxWorks]# ls -l /folk/user
[vxWorks]# task delete t1
[vxWorks]# bp -t t1 0x12345678
```

Most commands are compliant with the UNIX standard; therefore, there must be at least one space between each option and argument.

An option is a dash (-) plus a character. Users can also combine several options into a single string (for example, -oats is equivalent to -o -a -t -s).

An option can take extra parameters (for example, -f filename).

Spaces separate the argument. Therefore, if space is included, use a backslash (\) before the space to escape, otherwise use double quotes (") to surround the parameters.

For example：

[vxWorks]# **ls -l "/folk/user with space characters"**

[vxWorks]# **ls -l /folk/user\ with\ space\ characters**

4.2.1 Basic commands of command interpreter mode

Commands	**Description**
alias	Add an alias or list current alias
unalias	Remove alias
bp	Show, insert or remove breakpoints
cat	Connect or display files
cd	Switch to file folder
help	Show command description
ls	Show file folder contents
more	Browse or page text file
print errno	Display the symbol value of the error code
pwd	Show current work folder
version	Display VxWorks version information

4.2.2 Memory operation

Commands	**Description**
help memory	Display and memory-related command description
mem dump	Display memory
mem modify	Modify memory content values
mem info	Display memory information
mem list	Disassemble and display the position content of the specified command

4.2.3 Display object information

Commands	**Description**

help object	Display object-related command description
object info	Display object information
object class	Display object class information

4.2.4 Symbol operation

Commands	**Description**
help symbols	Display symbol related command description
echo	Display symbol content or value in one line of text
printf	Print out format
set / set symbol	Set symbol content
lookup	Symbol search

Access symbol content and location

The kernel shell interpreter mode is a string-oriented interface. However, users still need to separate symbol names, general strings, and numerical values.

When the symbol name is delivered to the command parameter, users may want to deliver the content value of the symbol (for example, display the symbol content value) or the position value of the symbol (for example, set the hardware breakpoint position).

Users can use & to acquire the symbol position value and use $ to acquire symbol content value. When accessing the symbol, the command must specify the access format.

For example:

[vxWorks]# **task spawn &printf %c $toto.r**

In this example, the command interpreter mode gets the position value of printf as a parameter and deliver it to the command task spawn, and use .r to specify the value of the toto symbol in the character format.

Symbol content value access format：

Use the following syntax to access content symbol values in a specific numerical format.

$*symName*[.*type*]

Type represents the format

r = Character

h= Short index

i = Integer (default)

l = Long integer

ll = Double integer

f = Single-precision floating-point

d = Double-precision floating-point

For example：When the symbol content value is 0x10

```
[vxWorks]# echo value
value
[vxWorks]# echo $value
0x10
```

The default is to read the symbol content value with a 32-bit integer. When there is a decimal point "." or an exponent symbol "E" or "e" in the string, the command interpreter mode treats the value as a double-precision floating-point.

Symbol position value access：

Use & acquire the symbol position value

For example：When the symbol position value is 0x123456789

```
[vxWorks]# echo &value
0x12345678
```

The specified symbol value has no meaning. However, the position of a symbol often represents the position value of the function.

For example：

```
[vxWorks]# bp &printf
```

4.2.5 Display, control and single-step execution of task

Commands	Description
help tasks	Display task-related command description
task	Display the TCB summary of the task
task info	Display the TCB description of detailed task
task spawn	Use default value to generate task
task stack	Show summary of each task stack
task delete	Delete task

task regs	Set the task register value
show task regs	Display the task register value
task suspend	Suspend task
task resume	Resume task
task hook	Display the function linked to the task
task step	Single-step execution of the task
task stepover	Single-step execution of the task, but not entering the sub-program
task continue	The task continues from the point of interruption
task stop	Stop task

4.2.6 Set command line content information

Commands	**Description**
help set	Display and set command line content related command description
set / set symbol	Set the symbol content value or create a new symbol. If the current work content is at the kernel, register the symbol to the symbol table.
set config	Set or display the kernel shell configuration variables
unset config	Remove the kernel shell configuration variable during the login period
set history	Set the number of command lines recorded. Display the command record if without parameters.
set prompt	Set command prompt character **%/** : Current working path **%n** : Current user **%m** : Target server name **%%** : Show % character **%c** : Current RTP name

4.2.7 Display system status

Commands	**Description**
show bootline	Show current kernel boot-line

show devices	Show current device
show drivers	Show system driver information
show fds	Display the file descriptor that is currently opened in the system.
show history	Display historical events under the current mode
show lasterror	Show the most recent error code

4.2.8 Alias

Commands	Description
alias	Set command alias for example：alias ll “ls -l”
attach	rtp attach
b	bp
bd	bp –u
bdall	bp –u #*
bootChange	set boot line
c	task continue
checkStack	task stack
cret	task continue -r
d	mem dump
detach	rtp detach
devs	show devices
emacs	set configuration LINE_EDIT_MODE=”emacs”
h	show history
i	task
jobs	rtp attach
kill	rtp detach
l	mem list
lkAddr	lookup –a
lkup	lookup
m	mem modify
memShow	mem info
ps	rtp
rtpc	rtp continue

rtpd	rtp delete
rtpi	rtp task
rtps	rtp stop
run	rtp exec
s	task step
so	task stepover
td	task delete
ti	task info
tr	task resume
ts	task suspend
tsp	task spawn
tt	task trace
vi	set configuration LINE_EDIT_MODE=”vi”

4.2.9 Initiate RTP

In the command interpreter mode, the file name path of the RTP application is a regular command, and the string following the file name path is the parameter delivered to the RTP.

For example：

```
[vxWorks]# /folk/user/TMP/helloworld.vxe
Launching process '/folk/user/TMP/helloworld.vxe' ...
Process '/folk/user/TMP/helloworld.vxe' (process Id = 0x471630) launched.
[vxWorks]# rtp
NAME ID STATUS ENTRY ADDR SIZE TASK CNT
------------ ---------- ----------- ---------- ---------- --------
```

Execute RTP in the foreground,

```
[vxWorks]# rtp exec myRTP.exe
```

Use the -i option to execute RTP in the background. However, its output is redirected to the kernel shell.

```
[vxWorks]# rtp exec -i myRTP.exe
```

Use Ctrl+W to move the RTP to the background and stop.

Use rtp background to restore the RTP stopped in the background.

Use rtp foreground to move the background RTP to the foreground.

Use Ctrl-C to remove RTP.

4.2.10 View and debug RTP

Commands	Description
rtp	Show process list
rtp stop	Stop process
rtp continue	Continue process
rtp delete	Delete process
rtp info	Show process information
rtp exec	Execute process
rtp attach	Attach shell session to process
rtp detach	Detach shell session from process
set cwc	Set current working content of the shell session
rtp task	Display the tasks being executed in the specified RTP
rtp foreground	Move the background process to the foreground
rtp background	Move process to the background

Set break point

The command bp can set the breakpoint in the task, or the RTP. Each breakpoint has a break number, and the user can remove the breakpoint according to the number.

Example：

List the file folder content

[vxWorks]# **ls -l /folk/usr**

Create command alias

[vxWorks]# **alias ls "ls -l"**

View task information

[vxWorks]# **task**

Suspend task, resume a task

[vxWorks]# **task suspend t1**

[vxWorks]# **task resume t1**

Set breakpoint at a specific location in the task

[vxWorks]# **bp -t t1 0x12345678**

Set the breakpoint in the function

[vxWorks]# **bp &printf**

Search the location of someInt

[vxWorks]# **echo &someInt**

Perform a single-step task from the breakpoint

[vxWorks]# **task stepover t1**

Continue task

[vxWorks]# **task continue t1**

Delete task

[vxWorks]# **task delete t1**

Execute RTP application

[vxWorks]# **/folk/user/TMP/helloworld.vxe**

Execute and deliver parameters to the RTP application

[vxWorks]# **cal.vxe -j 2002**

Deliver the executable option to the RTP loader and execute the RTP application (this sets the RTP application to use the 8K stack size).

[vxWorks]# **rtp exec -u 8192 /folk/user/TMP/foo.vxe -q**

List RTPs or display the basic information of the specified RTP

[vxWorks]# **rtp** [*rtpID*]

Show the specified RTP details

[vxWorks]# **rtp info** [*rtpID*]

Stop the RTP application and continue to execute the RTP application

[vxWorks]# **rtp stop 0x43210**

[vxWorks]# **rtp continue 0x43210**

4.3 C Language Interpreter Mode

In this mode, the operation of command is similar to C language expression. Several commands are provided to manage tasks or programs; however, it cannot access the program. Therefore, the debugging of the process must be carried out in the command interpreter mode.

4.3.1 Task management

Commands	**Description**
sp( )	Generate tasks with default values
sps( )	Generate tasks in stop mode

tr( )	Resume stopped task
ts( )	Stop task
td( )	Delete task
period( )	Generate a task of using a cycler to call function
repeat( )	The task of calling the function repeatedly
taskIdDefault( )	Set or report the default task ID
i( )	Display system information. Display the information about tasks in the system, such as status, CP, SP, and TCB location, etc. In order to save memory, this command is a continuous repeat value and may be slightly different.
iStrict( )	Same as above, but is a one-time value; therefore, the value is more accurate.
ti( )	Display task information
w( )	Show task waiting for summary information
tw( )	Show task waiting for object information
checkStack( )	Display task stacking information
taskIdFigure( )	Enter the task name to obtain the TID value

i( ) is a feature that is often used to view the activity information currently in the system. When you feel that there is nothing in operation at the moment, you can use it to see if the task is being executed.

```
-> i
   NAME        ENTRY       TID     PRI   STATUS    PC      SP      ERRNO   DELAY
.......................................................................................
  tExcTask    _excTask    3ad290    0    PEND     4df40   3ad0c0     0       0
  tLogTask    _logTask    3aa918    0    PEND     4df10   3aa748     0       0
  tWdbTask    0x41288     3870f0    3    READY    23ff4   386d78     3d      0004
  tNetTask    _netTask    3a59c0    50   READY    24200   3a5730     0       0
  tFtpTask    +ftpTask    3a2c18    55   PEND     23b28   3a2938     0       0
Value=0=0x0
```

4.3.2 Show system information

Commands	Description
devs( )	Display the known devices in the system
lkup( )	Look up symbol information

lkAddr( )	Look up symbol information from address
d( )	Display system memory information, users can set the initial display position, memory unit, unit number.
l( )	Disassemble and display the command location
printErrno( )	Display error code description
version( )	Display system version summary
cd( )	Switch work folder to specified location
ls( )	Show current work folder content
pwd( )	Show current work folder location
help( )	Display description
h( )	Display command record
shellHistory( )	Set or display command record
shellPromptSet( )	Replace the command prompt character
printLogo( )	Display operating system pattern

The lkup command is often used to search for symbol information. The simplest way to use is to enter the fragment string of the symbol to display the symbol information that matches.

```
-> lkup "dsm"
_dsmData 0x00049d08 text (vxWorks)
_dsmNbytes 0x00049d76 text (vxWorks)
_dsmInst 0x00049d28 text (vxWorks)
mydsm 0x003c6510 bss (vxWorks)
```

This command is case sensitive. If you do not want the command to be case sensitive, the syntax is as follows

```
-> lkup "[dD]sm"
```

4.3.3 Debug

Commands	Description
ld( )	Load object module into memory and dynamically link during execution
unld( )	Remove object module links and free up memory space
m( )	Modify the memory content value. Parameters: data width (byte, short, long...), starting position addr

	Use the m( ) command to display a contiguous memory space on the terminal. Users can type the hexadecimal value to modify the memory content value. If modification is not required, just press enter directly or type "." to leave.
mRegs( )	Modify the specified task register value
b( )	Set or display break point
bh( )	Set hardware break point
s( )	Execute program single-stepwise to the next command
so( )	Execute program single-stepwise without entering the subroutine (subprogram)
c( )	Continue execution from the break point
cret( )	Continue to execute until the current subroutine returns
bdall( )	Delete all break points
bd( )	Delete break point
bootChange( )	Modify startup parameter values

Kernel object module link/remove

The kernel object module can be dynamically loaded into the executing VxWorks kernel through the kernel object module loader.

The following is the standard kernel object module loading command in C language interpreter mode：

```
－＞ld＜./helloworld.o
```

The Ld( ) function can load the object module from the file or standard input to the kernel. Once an application module is loaded into the kernel memory, the function in the module can be called directly through the kernel shell to generate task or interrupt link, etc.

Modules can be reloaded via reld( ). When a newer version of the module is available, users can use this command to reload the module with the same name. Users can use the unld( ) command to remove a module.

Sometimes in your code, you will use a function with the same name as the kernel shell command, and when you want to call this function directly instead of a command, you need to add @ before the function name.

For example：

Execute C language description

-> test = malloc(100); test[0] = 10; test[1] = test[0] + 2

-> printf("Hello!")

Object module link

-> ld < /usr/apps/someProject/file1.o

Produce new symbol

-> MyInt = 100; MyName = "Bob"

Display system information (task related information)

-> i

Display specific task information

-> ti(s1u0)

Stop task, resume task

-> ts(s1u0)

-> tr(s1u0)

Set break point

-> b(0x12345678)

Execute program single-stepwise to the next function

-> s

Call the operating system built-in function; create a new symbol (**my_fd**).

-> my_fd = open ("file", 0, 0)

6 Debugging Applications with the Host Shell

Call the function in the user program

-> someFunction (1,2,3)

Call the function in the user program when the function name is the same as the command-line command

➔ @i()

4.3.4 Other instructions

File system：

－＞devs

```
drv   name
0     /null
1     /tyCo/0
```

3 /

4 /tffs0

RAM：

"/" This directory is mainly used to store certain temporary data by RAM DISK

FLASH：

"/tffs0" This directory is mainly used to store certain data that needs to be retained by FLASH

Telnet：

After setting the network card according to the instruction file in the disc, users can use the Telnet software to connect to the IMP and directly enter the kernel shell - C language interpreter mode, which can be used to operate the system remotely.

RS-232：

Based on the instructions of the serial communication software (tera term) in the disc, users can connect to the IMP through RS232 and enter the kernel shell - C language interpreter mode to operate the system.

FTP：

Based on the software instruction and settings in the disc, users can connect to the IMP via FTP and transfer the file to the IMP file system.