Freescale Semiconductor, Inc.

ICD_PPC Target Interface

Because of last-minute software changes, some information in this manual may be inaccurate. Please read the Release Notes, on the CodeWarrior CD, for the latest information.

Revised: 2/12/04

How to Contact Metrowerks

Corporate Headquarters	Metrowerks Corporation 7700 West Parmer Lane Austin, TX 78729 U.S.A.
World Wide Web	`http://www.metrowerks.com`
Sales	Voice: 800-377-5416 Fax: 512-996-4910 Email: sales@metrowerks.com
Technical Support	Voice: 800-377-5416 Email: support@metrowerks.com

Table of Contents

Table of Contents 1

1 Introduction 3

- Read the Release Notes . . . 3
- Technical Support . . . 4
- Contacting the Documentation Team . . . 4

2 ICD_PPC Target Interface Demo 5

- Debugging with the ICD_PPC Target Interface . . . 5
- Preparing the target system to communicate with the Debugger . . . 5
- Starting with a stationery . . . 6
- Starting with an example . . . 6
- Debugging an application . . . 7

3 ICD_PPC Target Interface Control 11

- Introduction. . . . 11
- Interfacing Your System and the Target . . . 12
  - P&E Driver DLL . . . 12
  - Hardware Connection . . . 12
  - Loading the ICD_PPC Target Interface . . . 12
- ICD_PPC Target Interface Menu Entries . . . 15
  - Load... . . . 16
  - Reset . . . 16
  - Setup... (or Connect...) . . . 16
  - Set MCU Type... . . . 16
  - Command Files . . . 16
  - Help . . . 16
- ICD_PPC Target Interface Dialogs. . . . 16
  - ICD_PPC Setup dialog . . . 17
  - ICD_PPC Target Interface MCU Selection dialog . . . 20
  - ICD_PPC Target Interface Command Files dialog . . . 21
- Status Bar Information for the ICD_PPC Target Interface . . . 22
  - Status Messages . . . 22
  - Stepping and Breakpoint Messages . . . 23

4 ICD_PPC Target Interface Environment 25

Default Target Setup 25
ICD_PPC Target Interface Environment Variables 25
ICD_PPC Target Interface Commands 35

5 Non volatile memory user interface 39

Flash commands 39
Commands usage example 48
Hardware Considerations 48
  Available drivers 48
  External flash 50

6 Target Interface Command Files 53

Target Interface Command Files Description 54
  Startup Command File 54
  Reset Command File. 54
  Preload Command File 55
  Postload Command File 56
Command Files dialog 56
Associated Commands 58
Associated Environment Variables 60

Index 63

1 Introduction

This document includes information to become familiar with the *ICD_PPC Target Interface* and to help you understand how to use this target interface. This document is divided into following sections:

- The ICD_PPC Target Interface Demo chapter give answers for common questions and describes how to use advanced features of the *ICD_PPC* Target Interface.
- The ICD_PPC Target Interface Control chapter gives a description of the *ICD_PPC* Target Interface specific dialog boxes and menu.
- The ICD_PPC Target Interface Environment chapter contains information about the environment variables which are used by the *ICD_PPC* Target Interface.
- The Target Interface Command Files chapter section gives information about the *ICD_PPC* Target Interface specific command file functionalities.
- The Non volatile memory user interface chapter section decribes how to load and write your application into the non volatile memory of your target.
- The Index contains all keywords of the *ICD_PPC Target Interface*.

Click any of the following links to jump to the corresponding section of this chapter:

- Read the Release Notes
- Technical Support
- Contacting the Documentation Team

Read the Release Notes

Before you use the ICD_PPC Target Interface, read the release notes. They contain important last-minute information about new features and technical issues or incompatibilities that may not be included in the documentation.

Technical Support

If you are having problems installing or using any Metrowerks product, contact Technical Support as shown in Table 1.1.

Table 1.1 Ways to Contact Technical Support

E-mail	`support@metrowerks.com` (USA) `support_europe@metrowerks.com` (Europe)

Telephone	`USA : +1 512.997.4700`
	`EMEA: +41 61.690.7505`

World Wide Web	`http://www.metrowerks.com/`
Compuserve	`Go: Metrowerks`
`CW_Release_Notes\email_Report_Forms\email_Tech_Question_Form.txt`	

Contacting the Documentation Team

Please report errors or omissions to the Metrowerks Documentation Department: `wordwarrior@metrowerks.com`.

2

ICD_PPC Target Interface Demo

Debugging with the ICD_PPC Target Interface

This section gives an overview of debugging with the *ICD_PPC* Target Interface. This target interface allows to debug a PowerPC derivative, in background mode, using the P&E PowerPC Interface Cable.

With this interface, you can download an executable program from the Debugger environment to an external evaluation board. You will also have the feedback of the real target system behaviour to the Debugger.

The Debugger will fully supervise and monitor the target system i.e. control the CPU execution. You can read and write to memory, single-step/ run/stop the CPU, set breakpoints and watchpoints.

NOTE **Unconcerned Components**: As the code is executed by an external processor, memory statistics are not available with the ICD_PPC Target Interface. Therefore, Profiling, Coverage analyzing and I/O simulation are not available with the ICD_PPC Target Interface.

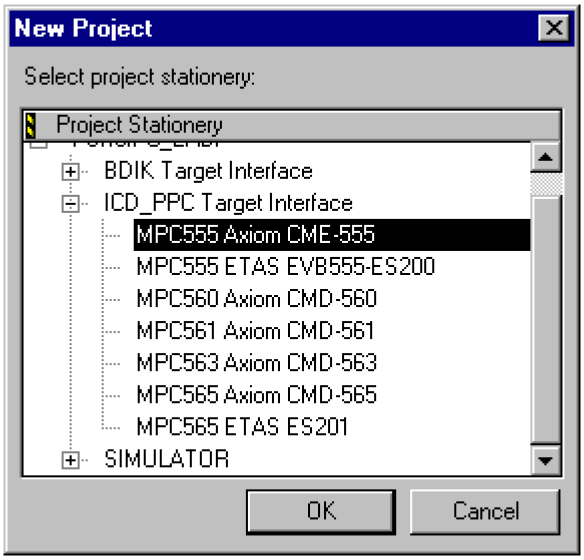
Preparing the target system to communicate with the Debugger

1. Connect the P&E PowerPC Interface Cable to the parallel Port of the host.
2. Connect the 10 pins connector of the cable to the BDM-Port of the board.

Starting with a stationery

1. Run the CodeWarrior IDE with the shortcut created in the program group or from your installation.
2. Choose the menu **File> New** to create a new project from a stationery. Select **ICD_PPC Target Interface stationery**, type in a project name and specify the project location. Press **OK**.

Figure 2.1 New Project Dialog

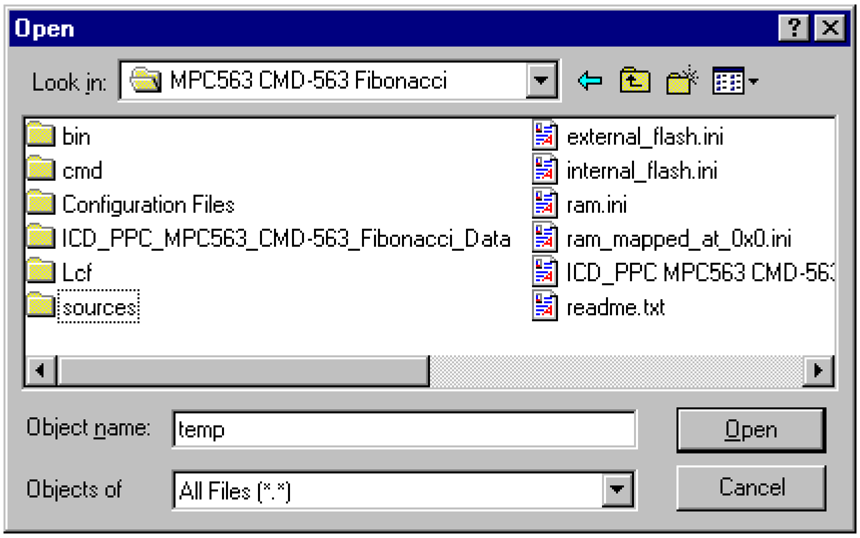


3. Select the project stationery corresponding to your board, press **OK**.
4. Choose one of the CodeWarrior targets of the project (for instance **Ram Application**). Now you have a project with all files set up, and an empty `main.c` file that you may fill in.

Starting with an example

1. Run the CodeWarrior IDE with the shortcut created in the program group or from your installation.
2. Open a *ICD_PPC's* project`(.mcp)`example from your installation like shown in Figure 2.2. The CodeWarrior examples are located in the `Examples` subdirectory of your installation.

Figure 2.2 Open Project Dialog

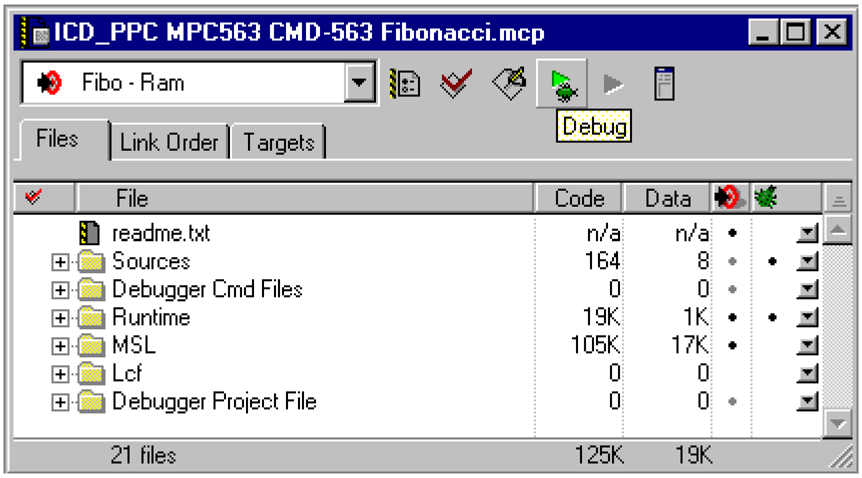


3. Click **Open** to load the example project.

Debugging an application

Once the example project or the stationary project has been opened, choose the menu **Project> Debug** or click on the **Debug** button like shown in Figure 2.3. The project is build and the Debugger is launched.

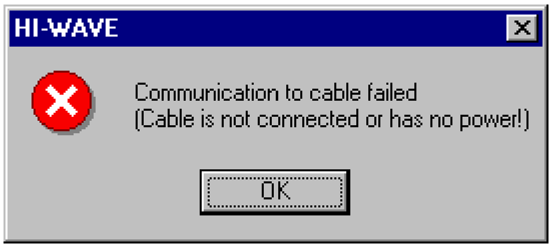
Figure 2.3 Start the Debugger



File	Code	Data
readme.txt	n/a	n/a
Sources	164	8
Debugger Cmd Files	0	0
Runtime	19K	1K
MSL	105K	17K
Lcf	0	0
Debugger Project File	0	0
21 files	125K	19K

1. If there is a communication problem, an error box will be displayed like shown in Figure 2.4.

Figure 2.4 Error message box

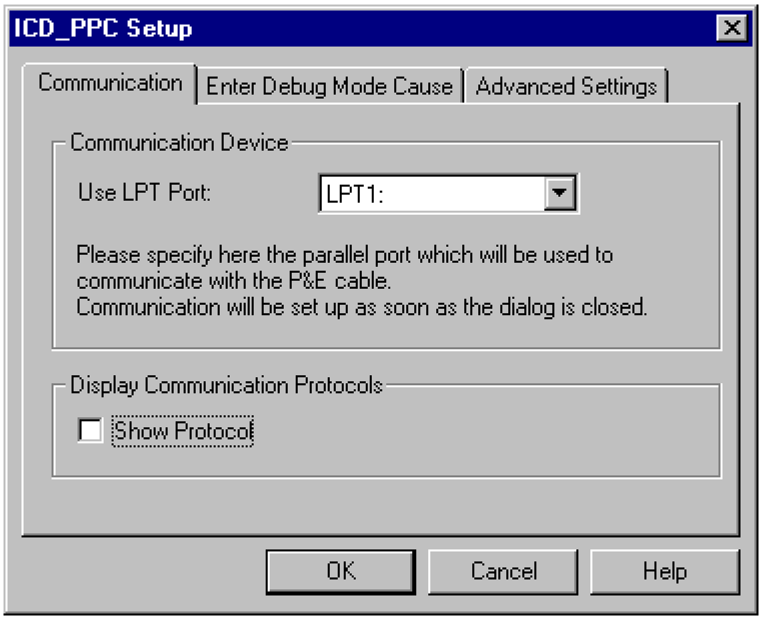


HI-WAVE

Communication to cable failed
(Cable is not connected or has no power!)

2. Close this message box clicking **OK**. The ICD_PPC Setup dialog is displayed. This dialog is shown in Figure 2.5.

Figure 2.5 ICD_PPC Setup dialog - Communication



ICD_PPC Setup

Communication Device

Use LPT Port: LPT1:

Please specify here the parallel port which will be used to communicate with the P&E cable.
Communication will be set up as soon as the dialog is closed.

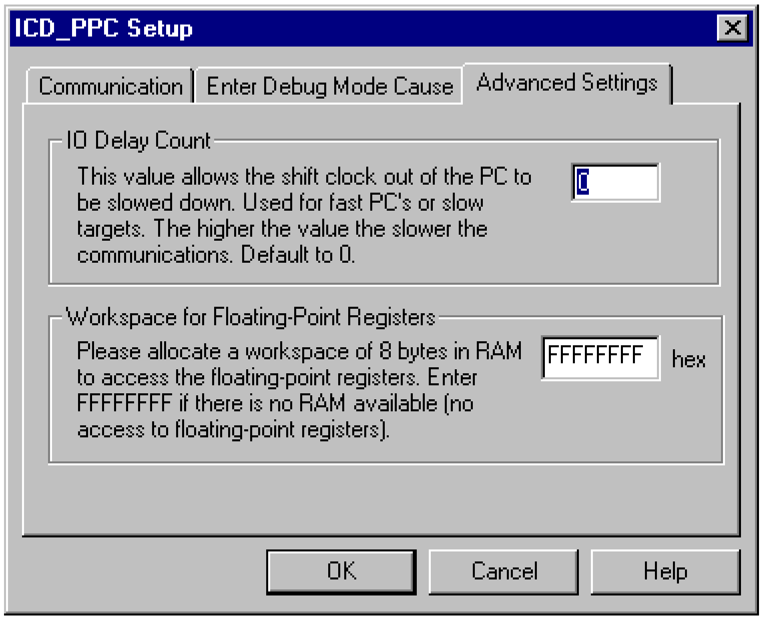
Display Communication Protocols

☐ Show Protocol

3. Correct the Communication settings if necessary.

4. Check the advanced settings panel from the ICD_PPC Setup dialog shown in Figure 2.6: the I/O delay count variable allows to slow down or speed up the communication. If you want to use the floating point registers display, type the address of a free workspace of 8 bytes in RAM .

Figure 2.6 ICD_PPC Setup dialog - advanced settings.



IO Delay Count

This value allows the shift clock out of the PC to be slowed down. Used for fast PC's or slow targets. The higher the value the slower the communications. Default to 0.

0

Workspace for Floating-Point Registers

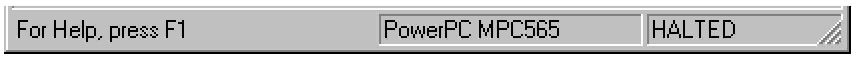
Please allocate a workspace of 8 bytes in RAM to access the floating-point registers. Enter FFFFFFFF if there is no RAM available (no access to floating-point registers).

FFFFFFFF hex

5. Click **Ok**.
6. If the connection is correct, the application is automatically loaded, you can debug it: choose **Run>Start/Continue** or click ➜ in the debugger.
7. The application is started.
8. Choose **Run>Halt** or click ⊣.The execution of the program is stopped.

The Debugger status bar shown in Figure 2.7 gives the cpu name and the target status.

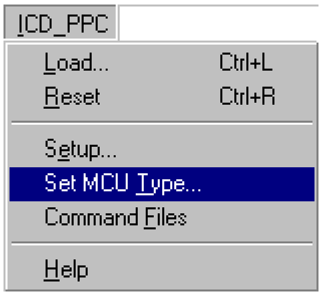
Figure 2.7 Debugger status bar



For Help, press F1 | PowerPC MPC565 | HALTED

The Debugger main menu contains a *ICD_PPC* entry shown in Figure 2.8 from where your can reset the application when selecting *Reset*, set the communication parameters when selecting **Setup...** or load a `.ABS` application to debug when selecting **Load...**

Figure 2.8 ICD_PPC menu entry



The **Setup..., Set MCU Type** menu entries open dialogs to set up your target system. The **Command Files** menu entry of the *ICD_PPC* menu allows to open the Target Interface Command Files dialog to set up the command files for the target interface.

3 ICD_PPC Target Interface Control

Introduction

Another advanced feature of the Debugger tools for the embedded systems development world is the ability to load different target interfaces, which implements the interface with target systems. In this document, the specific features of the *ICD_PPC Target Interface* are described.

This target interface allows to debug a PowerPC derivative, in background mode, using the P&E PowerPC Interface Cable.

With this interface, an executable program can be downloaded from the *CodeWarrior* environment to an external target system based on a PowerPC derivative.

The Debugger will fully supervise and monitor the CPU of the target system and control the CPU execution such as read and write in memory, single-step/run/stop processes in the CPU, setting of breakpoints and watchpoints.

The target system will execute the program and give the feedback of the real target system behaviour to the Debugger.

NOTE **Unconcerned Components**: as the code is executed by an external processor, memory statistics are not available with the ICD_PPC Target Interface. Therefore, Profiling, Coverage analysing and I/O simulation will not work with the ICD_PPC Target Interface.

Interfacing Your System and the Target

P&E Driver DLL

The Debugger needs a DLL `UNITPPCZ.DLL` from *P&E* to communicate through the cable to the PowerPC.

This DLL is installed during the software installation in your Windows directory.

Hardware Connection

The P&E PowerPC Interface Cable should be connected to your PC via a bidirectional parallel port, and linked with a standard 10-pin connector to the background mode port on your PowerPC target system.

For more information on the hardware connection, please refer to the P&E documentation.

Loading the ICD_PPC Target Interface

Usually the target is set in the **`[HI-WAVE]`** section of the **`PROJECT`** file, through the statement **`Target=ICD_PPC`**.

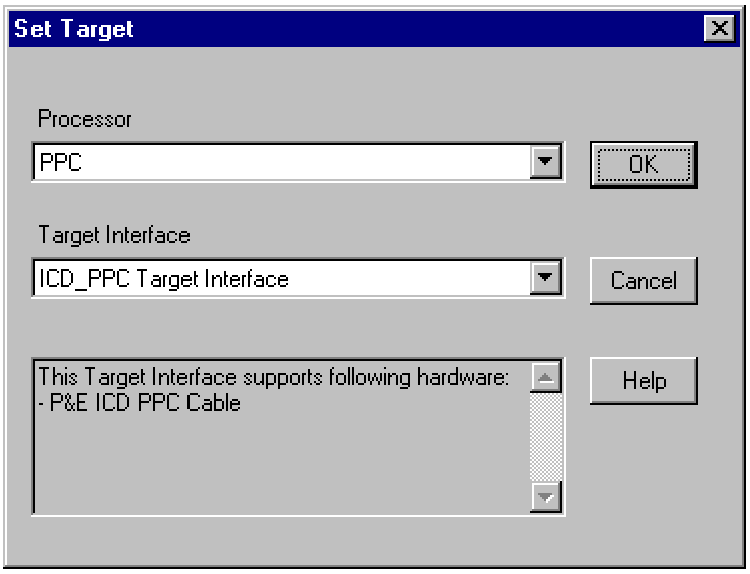
In this way, the debugger will try to connect to the external hardware using the *ICD_PPC* Target Interface, and will detect automatically that the target is connected to your system. In case the communication fails, the ICD_PPC Setup dialog pops up: the target is not connected or is connected to a different port.

If no target is set in the `PROJECT` file or if a different target is set, you can nevertheless load the *ICD_PPC* Target Interface: select in the main menu **Component>Set Target...**

The **Set Target** dialog shown in Figure 3.1 is displayed.

Select PPC in the “Processor” list, then choose *ICD_PPC* in the list of proposed targets and click **OK**.

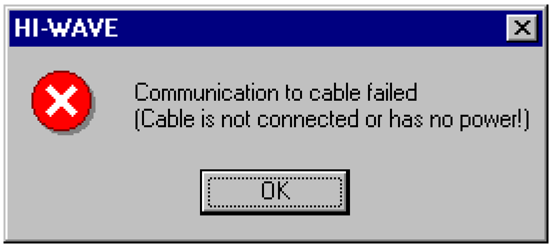
Figure 3.1 *Set Target* dialog



This Target Interface supports following hardware:
- P&E ICD PPC Cable

The debugger tries then to connect to the target. If there is a communication problem, an error box will be displayed like shown in Figure 3.2.

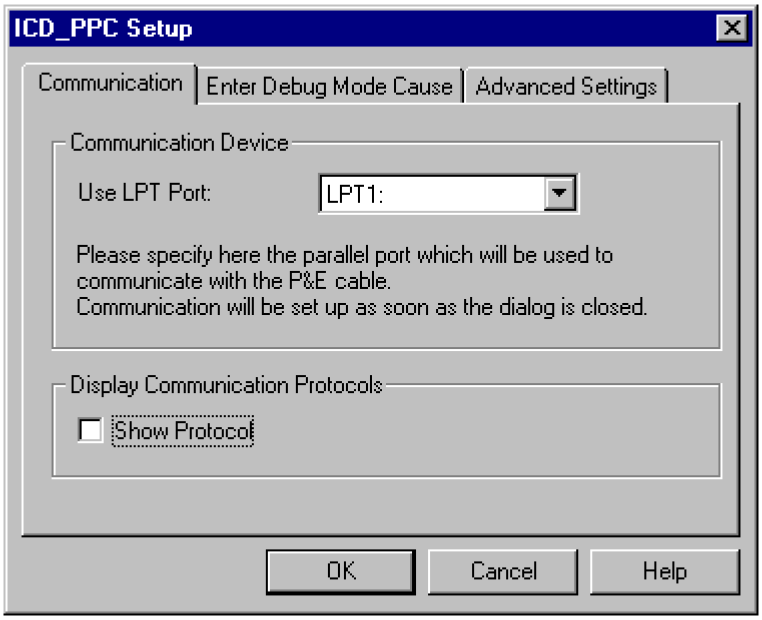
Figure 3.2 Target loading error box



Communication to cable failed
(Cable is not connected or has no power!)

Click **Ok**, the ICD_PPC Setup dialog shown in Figure 3.3 dialog pops up.

Figure 3.3 ICD_PPC Setup dialog



Use LPT Port: LPT1:

Please specify here the parallel port which will be used to communicate with the P&E cable.
Communication will be set up as soon as the dialog is closed.

[ ] Show Protocol

In this dialog it is possible to modify the *Communication Parameters*.

Make sure that the parameters on your host computer are correctly set up.

NOTE This dialog can also be opened selecting ICD_PPC>Connect... when previous connection attempts with the P&E Cable have failed or if the communication was lost.

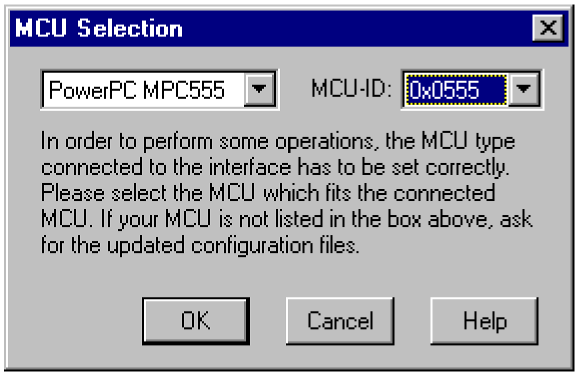
For details about *ICD_PPC Setup dialog* please see section ICD_PPC Setup dialog.

Once the *Communications Parameters* have been specified, click **OK**.

If no connection could be established, choose **ICD_PPC>Connect...** This opens the ICD_PPC Setup dialog where you can check if the used parameters are correct. Try again to connect to the target system by clicking **OK** .

The Debugger has to know which derivative is connected in order to load the matching Register File. The MCU selection dialog pops up if no MCUID has been defined in your project file, like shown Figure 3.4.

Figure 3.4 MCU Selection


MCU Selection

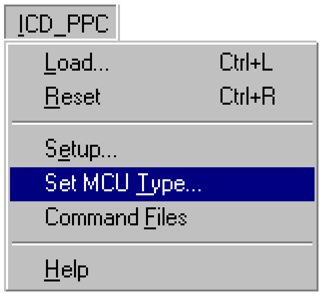
PowerPC MPC555 MCU-ID: 0x0555

In order to perform some operations, the MCU type connected to the interface has to be set correctly. Please select the MCU which fits the connected MCU. If your MCU is not listed in the box above, ask for the updated configuration files.

ICD_PPC Target Interface Menu Entries

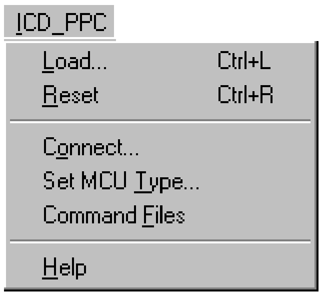
After loading the *ICD_PPC* Target Interface, the *Target* menu item is replaced by *ICD_PPC* shown in Figure 3.5.

Figure 3.5 ICD_PPC menu



However, if the connection to the target has failed, the entry **Setup...** of menu *ICD_PPC* is replaced with **Connect...**, like shown in Figure 3.6:

Figure 3.6 ICD_PPC menu when connection failed



The different entries of the *ICD_PPC* menu are described below:

Load...

Choose **ICD_PPC>Load...** to load the application to debug, for example an **.**ABS file.

Reset

The menu entry **ICD_PPC>Reset** resets the target system processor and executes the Reset Command File if this one is enabled.

Setup... (or Connect...)

Select entry **ICD_PPC>Connect...** or **ICD_PPC>Setup** to display the ICD_PPC Setup dialog. If the connection to the target has succeeded, the entry **Connect...** in menu *ICD_PPC* is replaced with **Setup...**.

Set MCU Type...

Select entry **ICD_PPC>Set MCU Type...** to display the ICD_PPC Target Interface MCU Selection dialog.

Command Files

Select entry **ICD_PPC>Command Files** to display the ICD_PPC Target Interface Command Files dialog.

Help

Select entry **ICD_PPC>Help** to access the Help file.

ICD_PPC Target Interface Dialogs

This section describes the dialogs which are specific to the *ICD_PPC* Target Interface.

Those dialogs are:

- The ICD_PPC Setup dialog.
- The ICD_PPC Target Interface MCU Selection dialog.
- The ICD_PPC Target Interface Command Files dialog.

ICD_PPC Setup dialog

The *ICD_PPC Setup dialog* pops up automatically if the communication with your board can not be established. However, this dialog can be opened by selecting menu entry **ICD_PPC>Connect...** or **ICD_PPC>Setup...**

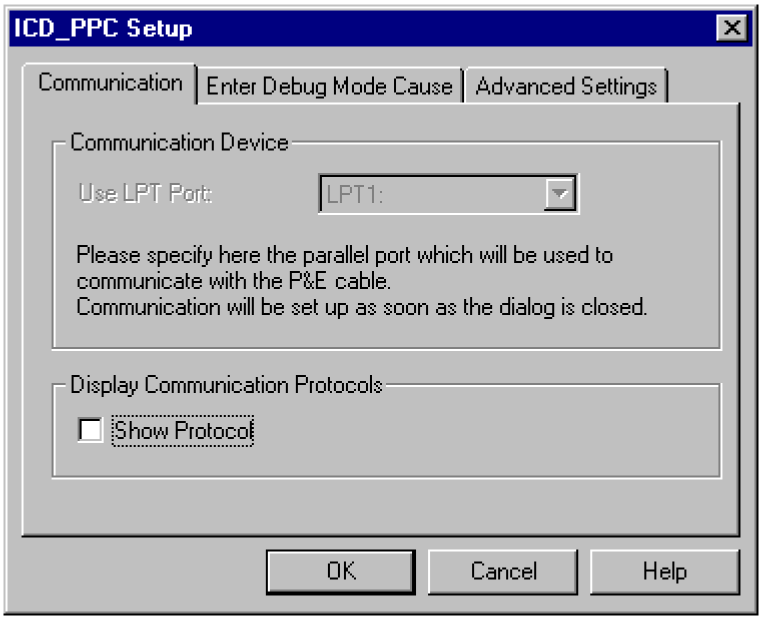
This dialogs consists of three property pages:

- The Communication property page.
- The Enter Debug Mode Cause property page
- The Advanced Settings property page.

Communication property page

If the connection to the target has been successfully achieved (dialog opened using menu entry **ICD_PPC>Setup...**), it is not possible to modify the parallel port to be used for communication. Only the *Show Protocol* check box can be modified like shown in Figure 3.7.

Figure 3.7 Communication property page

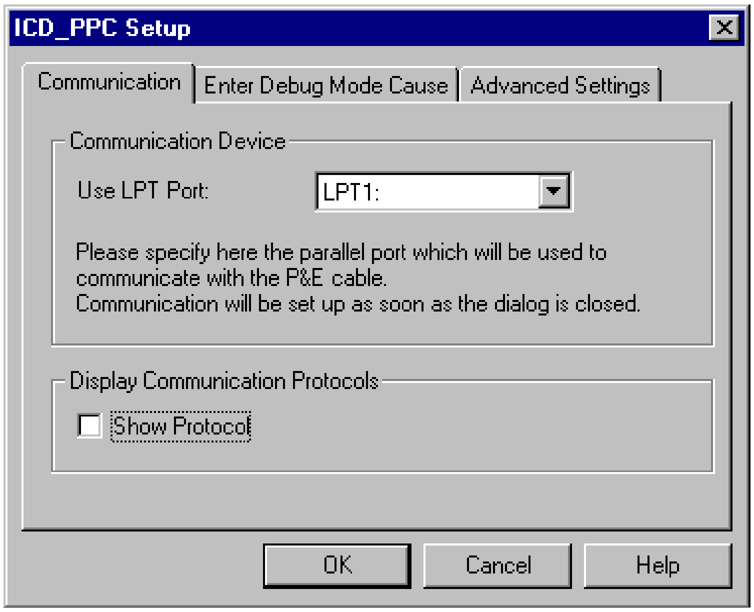


Please specify here the parallel port which will be used to communicate with the P&E cable.
Communication will be set up as soon as the dialog is closed.

☐ Show Protocol

If the connection to the target has failed, the ICD_PPC Setup dialog is automatically opened (otherwise use menu entry **ICD_PPC>Connect...**). It is then possible to modify all the *Communication Parameters* used for communication like shown in Figure 3.8.

Figure 3.8 ***ICD_PPC Communication Parameters***



Communication Device

Use LPT Port: LPT1:

Please specify here the parallel port which will be used to communicate with the P&E cable.
Communication will be set up as soon as the dialog is closed.

Display Communication Protocols

☐ Show Protocol

Use LPT port: The Use LPT port combo box contains the parallel port that will be used to communicate with the target system.

Show Protocol: The Show Protocol check box allows to switch on/off the display of the messages sent between the target system and the debugger. If the Show Protocol check box is checked, all the commands and responses sent and received are reported in the Command Line window.

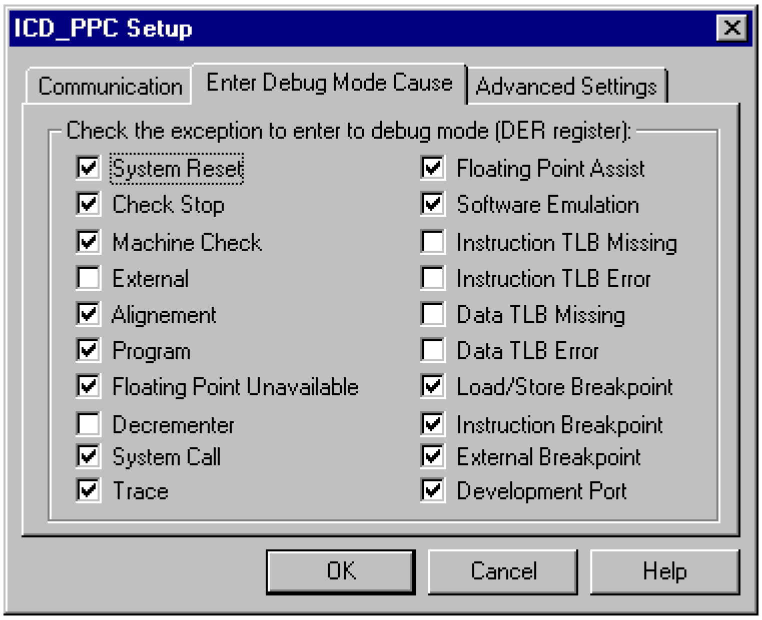
NOTE The Show Protocol is a useful debugging feature if there is a communication problem.

NOTE The settings performed in this dialog are stored for a later debugging session in the **[ICD_PPC]** section of the PROJECT file using variables **COMDEV** and **SHOWPROT**.

Enter Debug Mode Cause property page

The Enter Debug Mode Cause property page is shown in Figure 3.9 .

Figure 3.9 Enter Debug Mode Cause



Check the exception to enter to debug mode (DER register):

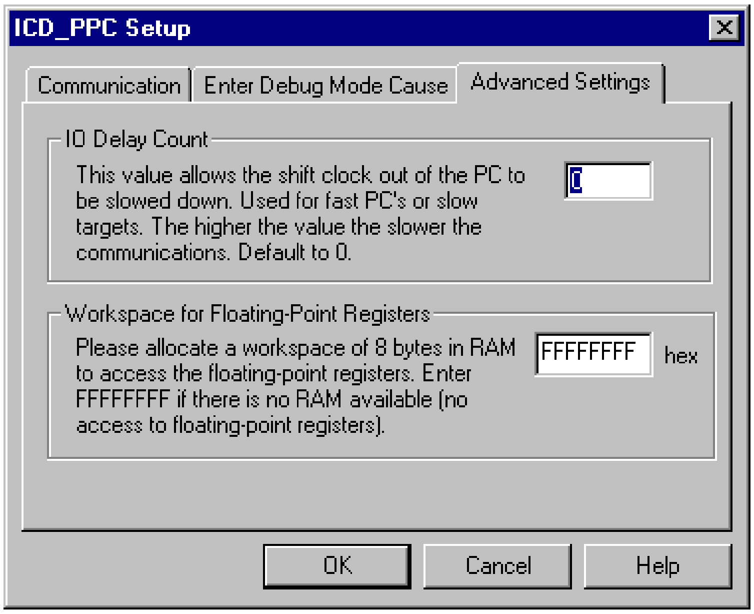
☑ System Reset	☑ Floating Point Assist
☑ Check Stop	☑ Software Emulation
☑ Machine Check	☐ Instruction TLB Missing
☐ External	☐ Instruction TLB Error
☑ Alignement	☐ Data TLB Missing
☑ Program	☐ Data TLB Error
☑ Floating Point Unavailable	☑ Load/Store Breakpoint
☐ Decrementer	☑ Instruction Breakpoint
☑ System Call	☑ External Breakpoint
☑ Trace	☑ Development Port

This dialog allows the user to set up the Debug Enable Register (DER). This register enables the events that may cause the processor to enter into debug mode. Please refer to your MPC Reference manual for further information.

Advanced Settings property page

The Advanced Settings property page is shown in Figure 3.10

Figure 3.10 Advanced Settings property page



ICD_PPC Setup

IO Delay Count

This value allows the shift clock out of the PC to be slowed down. Used for fast PC's or slow targets. The higher the value the slower the communications. Default to 0.

0

Workspace for Floating-Point Registers

Please allocate a workspace of 8 bytes in RAM to access the floating-point registers. Enter FFFFFFFF if there is no RAM available (no access to floating-point registers).

FFFFFFFF hex

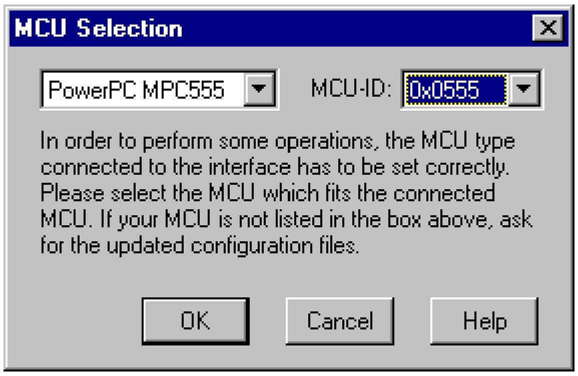
IO Delay Count: The IO Delay Count edit box contains a value that controls the communication speed. The higher is the value, the slower the communications.

Workspace for Floating Point registers: To access to a floating-point register, the target interface requires an empty buffer to temporarily store its value. This edit box should contain the address of a free area of eight bytes in the RAM memory of your target. If there is no RAM available, please enter FFFFFFFF: in this case, be aware that there will be no floating-point support.

ICD_PPC Target Interface MCU Selection dialog

The ICD_PPC Target Interface MCU Selection dialog is shown in Figure 3.11.

Figure 3.11 ICD_PPC Target Interface MCU Selection dialog



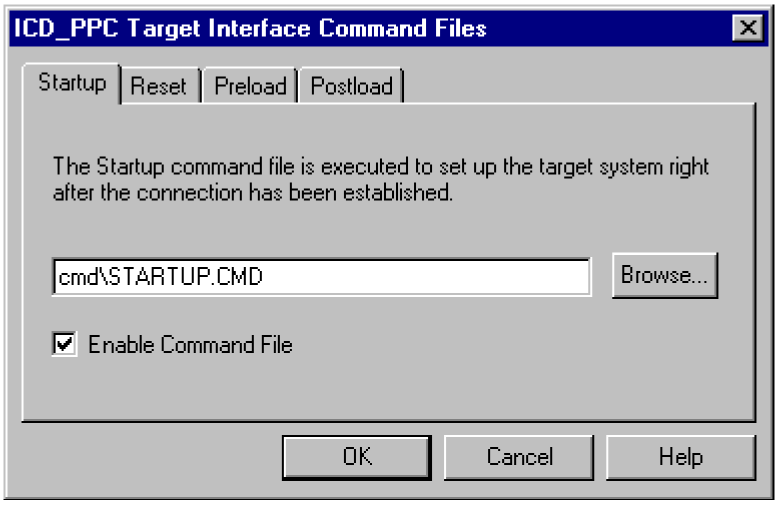
In order to perform some operations, the MCU type connected to the interface has to be set correctly. Please select the MCU which fits the connected MCU. If your MCU is not listed in the box above, ask for the updated configuration files.

The **Set MCU Type...** entry let you choose the current MCU of your application. This detection is not automatic and must be done by the user. The selected CPU derivative is displayed in the status bar. This selection is important, as it is used to load the CPU derivative matching I/O register file, which can be used to access the memory mapped registers and SPRs (Special Purpose Registers).

ICD_PPC Target Interface Command Files dialog

The **ICD_PPC Target Interface** Command Files dialog shown in Figure 3.12 can be opened selecting menu entry **ICD_PPC>Command Files**.

Figure 3.12 ICD_PPC Target Interface Command Files dialog



The Startup command file is executed to set up the target system right after the connection has been established.

cmd\STARTUP.CMD

☑ Enable Command File

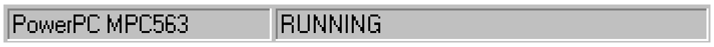
For more information, please refer to the Target Interface Command Files chapter.

NOTE The settings performed in this dialog are stored for a later debugging session in the `[ICD_PPC]` section of the `PROJECT` file *using variables* `CMDFILE0, CMDFILE1,... CMDFILEn`

Status Bar Information for the ICD_PPC Target Interface

The Status Bar Information for the ICD_PPC Target Interface is shown in Figure 3.13.

Figure 3.13 Status Bar Information for the ICD_PPC Target Interface


PowerPC MPC563 | RUNNING

When the *ICD_PPC* Target Interface has been loaded, specific information are given in the Debugger status bar. From left to right, the derivative name and the Debugger status (target status) are displayed.

Status Messages

Target ready

The Debugger is ready and waits until a new target or application is loaded. This message is generated once the Debugger has been started and the connection to the target system has been established.

No Link To Target

Connection to the target system has failed.

RUNNING

The application is currently executing in the Debugger.

HALTED

Execution of the application has been stopped on user request. The menu entry **Run>Halt** or the **Halt** icon in the tool bar has been selected.

RESET

This message is generated when the Debugger has been reset on user request. The menu entry **ICD_PPC>Reset** or the **Reset** icon in the tool bar has been selected, or the command Reset has been used.

Stepping and Breakpoint Messages

STEPPED

Execution of the application has been stopped after a single step on source level. The menu entry **Run>Single Step** or the **Single Step** icon in the tool bar has been selected.

STEPPED OVER

Execution of the application has been stopped after a step over a function call. The menu entry **Run>Step Over** or the **Step Over** icon in the tool bar has been selected.

STOPPED

Execution of the application has been stopped after a step out from function call. The menu entry **Run>Step Out** or the **Step Out** icon in the tool bar has been selected.

TRACED

Execution of the application has been stopped after an single step on assembler level. The menu entry **Run>Assembly Step** or the **Assembly Step** icon in the tool bar has been selected.

BREAKPOINT

Execution of the application has been stopped because a breakpoint has been reached.

WATCHPOINT

Execution of the application has been stopped because a watchpoint has been reached.

ILLEGAL_BP

Execution of the application has been stopped (unknown cause).

4
ICD_PPC Target Interface Environment

Default Target Setup

As any other target, the *ICD_PPC* Target Interface can be loaded from the *Target* menu or can be set as a default target in the `PROJECT` file (.ini file) which should be located in the working directory.

Typically the target is set in the **`[HI-WAVE]`** section from the `PROJECT` file as shown above. However, if the target is not defined, load the *ICD_PPC* Target Interface interactively. Please refer to section Loading the ICD_PPC Target Interface of this manual.

Listing 4.1 Example of [HI-WAVE] section from PROJECT file:

```
[HI-WAVE]
Target=ICD_PPC
```

NOTE Please refer to the simulator/Debugger User's Manual for further information about the PROJECT file.

ICD_PPC Target Interface Environment Variables

This section describes the environment variables which are used by the *ICD_PPC* Target Interface.

The *ICD_PPC* Target Interface specific environment variables are:

- CMDFILEn

- COMDEV
- FPR_WORKSPACE
- IO_DELAY_COUNT
- MCUID
- SHOWPROT
- USERDER

These variables are stored in the **[ICD_PPC]** section from the **PROJECT** file. An example is shown in Listing 4.2

Listing 4.2 Example of [ICD_PPC] section from PROJECT file:

```
[ICD_PPC]
CMDFILE0=CMDFILE STARTUP ON ".\cmd\startup.cmd"
CMDFILE1=CMDFILE RESET ON ".\cmd\reset.cmd"
CMDFILE2=CMDFILE PRELOAD ON ".\cmd\preload_RAM.cmd"
CMDFILE3=CMDFILE POSTLOAD ON ".\cmd\postload_RAM.cmd"
COMDEV=LPT1:
SHOWPROT=0
FPR_WORKSPACE=0x3FC800
IO_DELAY_COUNT=0
USERDER=0x71C7400F
MCUId=0x4563
```

The remainder of this section is devoted to describing each of the variables available for the *ICD_PPC* Target Interface. The variables are listed in alphabetical order and each is divided into several topics. The Table 4.1 gives the variable description format.

Table 4.1 Variable description format

Topic	Description
Short Description	Provides a short description of the variable.
Syntax	Specifies the syntax of the variable in a EBNF format.
Default	Shows the default setting for the variable.

Topic	Description
File	Name of the file which will hold the variable.
Section	Name of the section in the file.
Components	Specify the component from which the variable depends.
Description	Provides a detailed description of the variable and how to use it.
Example	Small example on how to use the variable.

COMDEV

Short Description

Specifies the parallel port of your computer to be used to communicate with your hardware.

Syntax

COMDEV=LPTn

File

PROJECT

Section

[ICD_PPC]

Components

ICD_PPC Target Interface.

Description

The communication port to be used on the host computer can be specified using the variable **COMDEV**.This variable is set according to the *Communication Device* edit box of the ICD_PPC Setup dialog.

Default communication device is **LPT1**.

```
COMDEV=LPT1:
```

FPR_WORKSPACE

Short Description

Start address of a free area of eight bytes to access floating-point registers.

Syntax

FPR_WORSPACE=<FPR_WORSPACE>

File

PROJECT

Section

[ICD_PPC]

Components

ICD_PPC Target Interface.

Description

This variable is set in the Advanced Settings property page. Its aim is to allocate a buffer of eight bytes where the value of floating-point registers will be temporarily stored.

Default value is 0xFFFFFFFF

```
FPR_WORKSPACE=0xFFFFFFFF
```

MCUID

Short Description

MCU identification number of the currently connected device.

Syntax

MCUID=<MCUID>

File

PROJECT

Section

[ICD_PPC]

Components

ICD_PPC Target Interface.

Description

This variable is set in the Set MCU Type... dialog when the user selects its derivative.

When loading a Debugger target, the definition of the I/O registers and SPRs (Special Purpose Registers) is loaded from a file, whose name is derived from the MCU identification number (MCUID).

```
MCUId=0x4563
```

IO_DELAY_COUNT

Short Description

Controls the communication speed.

Syntax

IO_DELAY_COUNT=<IO_DELAY_COUNT>

File

PROJECT

Section

[ICD_PPC]

Components

ICD_PPC Target Interface.

Description

This variable is set in the Advanced Settings property page. It controls the communication speed. The higher is the value, the slower the communications.

Default value is 0

```
IO_DELAY_COUNT=5
```

SHOWPROT

Short Description

Set Show Protocol On/Off

Syntax

SHOWPROT=1|0

File

PROJECT

Section

[ICD_PPC]

Components

ICD_PPC Target Interface.

Description

If the Show Protocol is used, all the commands and responses sent and received are reported in the Command Line component of the Debugger

This variable is set according to the Show Protocol check box of the Communication property page dialog. If the variable is set to 1, Show Protocol is activated.

The default value is 0 (Protocol Off).

```
SHOWPROT=1
```

NOTE The Show Protocol is a useful debugging feature if there is a communication problem.

USERDER

Short Description

Contains the value of the Debug Enable Register (DER).

Syntax

USERDER=<USERDER> hexadecimal value

File

PROJECT

Section

[ICD_PPC]

Components

ICD_PPC Target Interface.

Description

This variable is set in the Enter Debug Mode Cause property page. It contains the value of the Debug Enable Register, which is written by the debugger on the target at connection.

Default value is 0x71C7400F

```
USERDER=0x70C5400C
```

ICD_PPC Target Interface Commands

This section describes the *ICD_PPC* Target Interface specific commands which can be used when the *ICD_PPC* Target Interface is loaded.

The *ICD_PPC* Target Interface specific commands are:

- CMDFILE
- PROTOCOL
- FLASH

Those commands can be entered in the Target Interface Command Files or in the **Command Line** component of the Debugger.

The remainder of this section is devoted to describing each of the commands available for the *ICD_PPC* Target Interface. The commands are listed in alphabetical order and each is divided into several topics. The Table 4.2 gives the command description format.

Table 4.2 Command description format

Topic	Description
Short Description	Provides a short description of the command.
Syntax	Specifies the syntax of the command in a EBNF format.
Alias	Specify synonym for the command if any.
Components	Specify the component from which the command depends.
Description	Provides a detailed description of the command and how to use it.
Example	Small example of how to use the command.

FLASH

Short Description

Lists the current settings of the flash driver. Can be extended to other functions when used with extensions. Please refer to Non volatile memory user interface for a more complete description.

Syntax

FLASH

FLASH SETUP TYPE=<driverName> [SIZE=<device size in bytes>] [BUS=<bus width in bits>][CLOCK=<CPU frequency in MHz>] [OFFSET=<location offset>]

FLASH LOAD

FLASH IDLE

FLASH BLANKCHECK <first address> <last address>

FLASH ERASE <first address> <last address>

FLASH MASSERASE

Alias

None

Components

ICD_PPC Target Interface.

Description

This command lists the current settings of the flash driver. If no driver has been selected with the FLASH SETUP command, a message will inform you that no information is available, as the driver is not setup.

Extended commands:

FLASH SETUP: sets the desired driver to handle the non volatile memory

FLASH LOAD: opens a “non volatile memory loading session”

FLASH IDLE: closes a “non volatile memory loading session”

FLASH BLANKCHECK: check if the specified memory range is blank.

FLASH ERASE: erases partially the device.

FLASH MASSERASE: erases the entire non volatile memory.

```
in>flash
Driver:
-------
Name:             mpc555
Revision number: 1
Entry point:      0x003fb424
Buffer address:   0x003fc000
Buffer size:      0x00003000
Input address:    0x003fb400
Output address:   0x003fb400
Alignement mask: 0x40

Device:
-------
Offset:           0x00000000
Size (1 unit):    0x00070000 Bytes / 448 kBytes
Bus width:        32-bit
Bus clock:        40 MHz
```

NOTE The commands for non volatile memory can be typed directly in the Command Line component, but this is not really user friendly. It is more useful to insert these commands, especially for setup, erase and program operation in system command files (See Target Interface Command Files). These Command Files will be automatically and transparently executed when an application is loaded for example, from the "**File**" menu entry, "**Load application...**" or with the **LOAD** command.

PROTOCOL

Short Description

Switch on/off the *Show Protocol* functionality for your Evaluation Board.

Syntax

PROTOCOL ON|OFF

Alias

None

Components

ICD_PPC Target Interface.

Description

If this command is used, all the messages sent to and received from the *ICD_PPC* target system are reported in the *Command Line* window of the Debugger.

The *Show Protocol* facility can also be switched on/off using the corresponding check box in the Communication property page.

The state of the *Show Protocol* is stored in the **`[ICD_PPC]`** section of the **`PROJECT`** file using variable **`SHOWPROT`**.

Example

```
PROTOCOL ON
```

NOTE The Show Protocol is a useful debugging feature if there is a communication problem.

5

Non volatile memory user interface

In order to write into flashes, eeproms or other non volatile memory modules found in modern MCUs, special algorithms defined by microprocessor designers have to be used. Also, the Flash devices have to be erased before they can be written and may need some initialization in order to be accessible and may be protected.

The ICD_PPC target interface provides a non volatile memory user interface via command line commands that can be executed from system command files, like the startup, preload and postload command files, to transparently erase and program internal and external flash modules (See Target Interface Command Files). All operations results are displayed in the Command Line window.

The user does not need to care about code generation alignment and block chunks.

Flash commands

The available commands are listed below:

- FLASH
- FLASH SETUP
- FLASH LOAD
- FLASH IDLE
- FLASH BLANKCHECK
- FLASH ERASE
- FLASH MASSERASE

These commands can be entered in the Target Interface Command Files or in the *Command Line* component of the Debugger .

The remainder of this section is devoted to describing each of the commands available for the *ICD_PPC* Target Interface. The commands are listed in alphabetical order and each one is divided into several topics.

FLASH

Short Description

Lists the current settings of the flash driver

Syntax

FLASH

Alias

None

Components

ICD_PPC Target Interface.

Description

This command lists the current settings of the flash driver. If no driver has been selected with the FLASH SETUP command, a message will inform you that no information is available, as the driver is not set.

If a driver has already been selected, a list of information will be displayed like shown in the example below.

The Driver information part is quite technical and is more a support debugging information that information for the user. The Device information part shows default values given by the driver file itself. These information can be useful for driver operations, more for external devices, that can be of variable size, bus width and regularly located in a different place in the memory map. The FLASH SETUP commands is used to load the desired driver and set the device parameters.

```
in>flash
Driver:
-------
Name:             mpc555
Revision number: 1
Entry point:      0x003fb424
Buffer address:   0x003fc000
Buffer size:      0x00003000
Input address:    0x003fb400
Output address:   0x003fb400
Alignement mask: 0x40

Device:
-------
Offset:           0x00000000
Size:             0x00070000 Bytes / 448 kBytes
Bus width:        32-bit
Bus clock:        40 MHz
```

FLASH SETUP

Short Description

This command sets the desired driver to handle a non volatile memory.

Syntax

FLASH SETUP TYPE=<driverName> [SIZE=<device size in bytes>] [BUS=<bus width in bits>][CLOCK=<CPU frequency in MHz>] [OFFSET=<location offset>]

driverName is the name of the driver to select. Please see section "Available Drivers" to select the correct driver to handle a non volatile memory.

SIZE is the total size of a non volatile memory block in Bytes. If several devices (chips) are in parallel, the SIZE parameter should be the sum of all device sizes. This parameter is not relevant for all algorithms. Please check driver specific section for other parameters utility.

BUS is the width of the data bus addressed by the processor when accessing the non volatile memory. It is not always relevant when setting internal flash. When setting external flash, the driver can evaluate the total device size and sectors size.

The CLOCK parameter can be used to inform the driver about a the processor bus clock, to set programming constants. This parameter is not relevant for all algorithms. Please check driver specific section for other parameters utility.

The OFFSET parameter specifies where the non volatile memory starts in the memory map.

Alias

None

Components

ICD_PPC Target Interface.

Description

This command sets the desired driver to handle a non volatile memory. The user needs at least to specify the driver name. The other parameters are optional. Please check driver specific section for other parameters utility.

The command above will set the driver for an AM29BL162CB flash device, a 2x2 Megabytes device, 32-bit access, i.e. 2 devices in parallel, the non volatile memory starting at 0xC00000.

```
in>flash setup type=am29bx16 bus=32 size=0x400000 offset=0xC00000
```

FLASH LOAD

Short Description

This command opens a "non volatile memory loading session" and arms the debugger before loading an application to non volatile memory.

Syntax

FLASH LOAD [OFFSET [-]<offset value>]

Alias

None

Components

ICD_PPC Target Interface.

Description

Instead of writing code data directly to the device memory, the Debugger buffers the code data, dumps the buffer in the processor onchip ram, then calls the flash driver which will program the dumped buffer, and all this in a cyclic way until the application has been programmed.

The driver is loaded also in the processor onchip ram before starting dumping any buffer.

When the FLASH LOAD command has been executed, an application can be loaded. For example, from the "File" menu entry, "Load application..." or with the LOAD command.

An offset value can be specified after the FLASH LOAD command. This offset, which can be signed, is a value retrieved from the memory address destination of the data code given in the application absolute or elf file. For example, in standalone, the internal flash is planned to be at the address 0xC00000. At debugging time, it is required by the driver to be at address 0x0, default location. The user can generate an application for the internal flash at 0xC00000 and use the OFFSET option to correctly program the device which is at debug/programmation time at address 0x0.

```
in>flash load
SX program written to target.
Flash driver loaded to device.
Driver is armed for flash loading.
```

The result of the commands indicates that the driver has been loaded to the device and that is armed. From this point, an application can be loaded for example, from the "File" menu entry, "Load application...".

FLASH IDLE

Short Description

This command closes a "non volatile memory loading session" and disarms the debugger after loading an application to non volatile memory.

Syntax

FLASH IDLE

Alias

None

Components

ICD_PPC Target Interface.

Description

When this command is executed, the debugger will close the programming session. The driver will not be called any longer and the debugger code data buffering/programming mechanism is closed.

Driver disarmed for flash loading:

```
in>flash idle
```

FLASH BLANKCHECK

Short Description

To check if a device memory range is blank.

Syntax

FLASH BLANKCHECK <first address> <last address>

Alias

None

Components

ICD_PPC Target Interface.

Description

This command is provided to easily check if a device memory range is blank (erased, ready to be programmed).

The first and the last memory addresses to be verified must be passed. The driver is also loaded to the onchip ram of the processor.

Note that usually for internal flash drivers, adresses must be 4-byte aligned, like 0x0 to 0x3, 0x1000 0x1FFF. A range like 0x100 to 0x110 would fail.

```
in>flash blankcheck 0x0 0xFFFF
SX program written to target.
Flash driver loaded to device.
Operation failed. Driver returned message: DEVICE_NOT_BLANK
Flash range 0x00000000-0x0000ffff is not blank possibly at
0x00000000.
```

In the example above, the device is probably programmed at address 0x0. It might be possible that the driver provides an invalid address.

FLASH ERASE

Short Description

To erase partially the device.

Syntax

FLASH ERASE <first address> <last address>

Alias

None

Components

ICD_PPC Target Interface.

Description

The first memory address to erase and the last memory address to erase must be passed. The driver is also loaded to the onchip ram of the processor. Sector erasing does not provide an exact erasing. But the driver will at least erase the passed range of memory. Therefore, if this range is part of a sector or multiple sectors, all these sectors will be entirely erased.

```
in>flash erase 0x0 0xFFFF
SX program written to target.
Flash driver loaded to device.
Flash range 0x00000000-0x0000ffff erased successfully.
```

FLASH MASSERASE

Short Description

To erase the entire non volatile memory.

Syntax

FLASH MASSERASE

Alias

None

Components

ICD_PPC Target Interface.

Description

This command erases the entire non volatile memory. It usually uses a special runtime provided by the device manufacturer. It takes for example typically 40 seconds for an AM29BL162CB device. It might be more interesting to use a regular ERASE command if only a part of the device is programmed.

```
in>flash masserase
SX program written to target.
```

```
Flash driver loaded to device.
Flash mass erased successfully.
```

Commands usage example

The commands for non volatile memory can be type directly in the Command Line component, but this is not really user friendly. It is more useful to insert these commands, especially for setup, erase and program operation in system command files select in the ICD_PPC menu entry "Command Files", "Preload" and "Postload" indexes.

These command files will be automatically and transparently executed when an application is loaded for example, from the "File" menu entry, "Load application…" or with the LOAD command.

For example, in a "PRELOAD_MPC555_ONCHIP_FLASH.CMD" file of your project:

```
flash setup type=mpc555
flash erase 0x0 0xFFFF
// flash masserase
flash load
```

For example, in a "POSTLOAD_MPC555_ONCHIP_FLASH.CMD" file of your project:

```
flash idle
```

Hardware Considerations

Available drivers

Internal flash

MPC555 driver, version 1

This driver has been designed for the MPC555 CMF A and CMF B internal flash.

Driver default settings:

- data bus width is 32-bit.
- device size is 0x70000 bytes, 448 kB.
- bus clock is 40 MHz (parameter not used).
- offset location is 0.

This driver has been implemented with the General Market CMF Driver for MPC555 v3.0.2 from Motorola, algorithm 6.1 for part revision K1, K2 (CMF rev.6), K3 (CMF rev.7) and M (CMF rev.8).

Limitation: The General Market CMF Driver requires the chip to be at 40 Mhz. Before programming or erasing, the command below should be executed when using a 4.0 Mhz reference Xtal as oscillator (see Table 8-10 of the MPC555 User Manual for PLPRCR info):

```
WL 0x002FC284 0x00910000
```

MPC555shd driver, version 1

This driver has been designed for the MPC555 CMF A and CMF B internal shadow flash.

Driver default settings:

- data bus width is 32-bit.
- device size is 0x200 bytes (0x100 at address 0 in CMF A and 0x100 at address 0x40000 in CMF B) .
- bus clock is 40 MHz (parameter not used).
- offset location is 0.

This driver has been implemented with the General Market CMF Driver for MPC555 v3.0.2 from Motorola, algorithm 6.1 for part revision K1, K2 (CMF rev.6), K3 (CMF rev.7) and M (CMF rev.8).

Limitation: The General Market CMF Driver requires the chip to be at 40 MHz. Before programming or erasing, the command below should be executed when using a 4.0 MHz reference Xtal as oscillator: (see Table 8-10 of the MPC555 User Manual for PLPRCR info)

```
WL 0x002FC284 0x00910000
```

MPC565 driver, version 1

This driver has been designed for the MPC565 UC3F A and UC3F B internal flash.

Driver default settings:

- data bus width is 32-bit.
- device size is 0x100000 bytes, 1MB.
- bus clock is 40 MHz (parameter not used).
- offset location is 0.

This driver has been implemented with the General Market C3F Driver for MPC565 v3.0.1 from Motorola.

MPC565shd driver, version 1

This driver has been designed for the MPC565 UC3F A and UC3F B internal shadow flash.

Driver default settings:

- data bus width is 32-bit.
- device size is 0x400 bytes (0x200 at address 0 in UC3F A and 0x200 at address 0x80000 in UC3F B).
- bus clock is 40 MHz (parameter not used).
- offset location is 0.

This driver has been implemented with the General Market C3F Driver for MPC565 v3.0.1 from Motorola.

External flash

AM29BX16 driver, version 1

This driver has been designed for the following AMD devices (registered device Ids):

- AM29F200AB,AM29F200AT, AM29LV200BB, AM29LV200BT,
- AM29LV400BB, AM29LV400BT,
- AM29LV800BB, AM29LV800BT,
- AM29BL162CB

However, any AMD flash having the same kind of program and erase command should also be supported.

Driver default settings:

- data bus width is 32-bit.
- device size is 0x400000 bytes, 4MB.
- bus clock is 40 MHz (parameter not used).
- offset location is 0.

Possible bus width: 16 and 32-bit. Please set this value according your Memory Controller Base Registers ("BRx" registers) setting and board settings.

AMD unregistered devices erase and masserase: Erase will be performed by a smart algorithm combining blankchecking then erasing block after block. The device size is relevant and must be properly set with the setup command to avoid unimplemented memory access errors. Mass erase is done using chip erase command.

I28BX16 driver, version 1

This driver has been designed for the following Intel devices (registered device Ids):

- 28F800F3T, 28F800F3B, 28F800B3T, 28F800B3B, 28F800C3T, 28F800C3B,
- 28F160F3T, 28F160F3B, 28F160B3T, 28F160B3B, 28F160C3T,28F160C3B

However, any Intel flash having the same kind of program and erase command should also be supported.

Driver default settings:

- data bus width is 32-bit.
- device size is 0x400000 bytes, 4MB.
- bus clock is 40 MHz (parameter not used).
- offset location is 0.

Possible bus width: 16 and 32-bit. Please set this value according your Memory Controller Base Registers ("BRx" registers) setting and board settings.

Intel unregistered devices erase and masserase: Erase and masserase will be performed by a smart algorithm combining blankchecking then erasing block after block. The device size is relevant and must be properly set with the setup command to avoid unimplemented memory access errors.

6 Target Interface Command Files

The Target Interface offers the possibility to play specific command files on different events:

- at connection: Startup Command File,
- at reset: Reset Command File,
- right before a file is loaded: Preload Command File,
- right after a file has been loaded: Postload Command File.

The command files full name and status (enable/disable) can be specified either with the **CMDFILE** Command Line command or using the Command Files dialog.

You can use any of the Debugger commands in those files and take advantage of the wide set of commands introduced in the Debugger manual to setup the target hardware on one of those events. An example of command file is shown in Listing 6.1

Listing 6.1 Example of a command file content:

```
WB 0x0035 0x00
WB 0x0012 0x11
PROTOCOL OFF
```

- The **WB 0x0035 0x00** command sets memory location **0x35 to 0**.
- The **WB 0x0012 0x11** command sets memory location **0x12 to 0x11**.
- The command **PROTOCOL OFF** switch of the Show Protocol.

Target Interface Command Files Description

This section describes the Target Interface command files.

The command files will be very useful to debug your MPC board. They will allow you to configure your board at startup to make the debugging session possible. They will play commands after a reset, to recover this initial state. They will play commands for you to open and close the flash session.

Startup Command File

The *Startup* Target Interface Command Files is executed by the Debugger straight after the Target Interface has been loaded.

The *Startup* command file full name and status (enable/disable) can be specified either with the **`CMDFILE STARTUP`** *Command Line* command or using the *Startup* index of the Command Files dialog.

By default the **`STARTUP.CMD`** file located in the current project directory is enabled as the current *Startup* command file.

To ensure a proper connection, it is recommended that the startup file contains the command **`RESET`**. In this way, the `RESET` command file will be played and will configure your target for debugging.

Reset Command File

The *Reset* Target Interface Command Files is executed by the Debugger straight after the reset button, menu entry or *Command Line* command has been selected.

The *Reset* command file full name and status (enable/disable) can be specified either with the **`CMDFILE RESET`** *Command Line* command or using the *Reset* index of the Command Files dialog

By default the **`RESET.CMD`** file located in the current project directory is enabled as the current *Reset* command file.

Use this file to configure your board properly, before re-starting the debugging cession.

- To disable the watchdog

```
WL 0x002FC004 0xFFFFFF88
```

- To disable internal flash

```
//Set FLEN bit to 0 in IMMR register (overwrites RCW register,
FLEN flag)
RS SPR638 0xFFF00000
```

- To configure the memory

  Use the chip select functionality to move the address of your Ram or Flash memory.

```
// CS1:OR1 and BR1 moves the external SRAM at address 0x00000000
(2 MBytes at 0x000000-0x1FFFFF)
WL 0x002FC108 0x00000003 //BR1: 32-bit port
WL 0x002FC10C 0xFFE00000 //OR1: 0 wait state, asynchronous
operation
```

Preload Command File

The *Preload* Target Interface Command Files is executed by the Debugger right before an application is loaded to the target system through the Target Interface.

The *Preload* command file full name and status (enable/disable) can be specified either with the **CMDFILE PRELOAD** *Command Line* command or using the *Preload* index of the Command Files dialog.

By default the **PRELOAD.CMD** file located in the current project directory is enabled as the current *Preload* command file.

The *Preload* command is extremely useful when using the Non volatile memory user interface . The Example of a Preload command file, with use of flash commands demonstrates how to choose the non volatile memory driver, erase a part of the flash and arm the debugger before loading an application to the FLASH memory. It may also contains a RESET command, to ensure a clean initial state.

Listing 6.2 Example of a *Preload* command file, with use of flash commands

```
flash setup type=mpc555
flash erase 0x0 0xFFFF
// flash masserase
flash load
```

Postload Command File

The *Postload* Target Interface Command Files is executed by the Debugger right after an application has been loaded to the target system through the Target Interface.

The *Postload* command file full name and status (enable/disable) can be specified either with the **`CMDFILE POSTLOAD`** Command Line command or using the *Postload* index of the Command Files dialog.

By default the **`POSTLOAD.CMD`** file located in the current project directory is enabled as the current *Postload* command file.

In the example below, Example of a Postload command file, with use of flash commands, the *Postload* command file automatically closes the “non volatile memory loading session”.

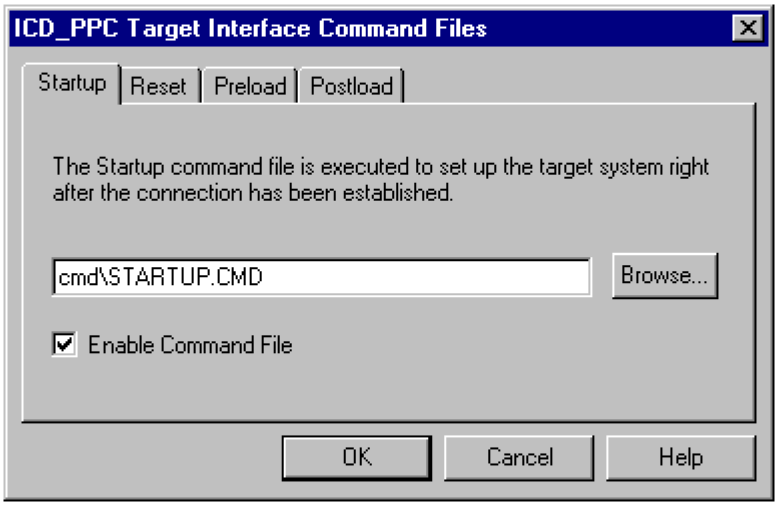
Listing 6.3 Example of a *Postload* command file, with use of flash commands

```
flash idle
```

Command Files dialog

The **Target Interface Command Files** dialog shown in Figure 6.1 can be opened selecting menu entry **"TargetName">Command Files**. (In this section, **TargetName** is the name of the target, like **ICD_PPC**.

Figure 6.1 Command Files dialog



The Startup command file is executed to set up the target system right after the connection has been established.

cmd\STARTUP.CMD

[✓] Enable Command File

Each index of this dialog corresponds to an event on which a Target Interface Command Files can be automatically run from the Debugger : Startup Command File, Reset Command File, Preload Command File, Postload Command File.

The command file in the edit box is executed when the corresponding event occurred.

Using the **Browse** button, you can set up the path and name of the command file.

The **Enable Command File** check box allows to enable/disable a command file on a event. By default, all command files are enabled:

- the default **Startup** command file is `STARTUP.CMD`,
- the default **Reset** command file is `RESET.CMD`,
- the default **Preload** command file is `PRELOAD.CMD`,
- the default **Postload** command file is `POSTLOAD.CMD`.

NOTE The settings performed in this dialog are stored for a later debugging session in the **["targetName"]** section of the PROJECT file using variables **CMDFILE0, CMDFILE1, ... CMDFILEn**

Associated Commands

This section describes the Command Files commands which can be used when the Target Interface is set:

- CMDFILE

These commands can be entered trough the Target Interface Command Files or in the *Command Line* component of the Debugger.

The commands are listed in alphabetical order. The Table 6.1 gives the description format.

Table 6.1 Command description format

Topic	Description
Short Description	Provides a short description of the command.
Syntax	Specifies the syntax of the command in a EBNF format.
Description	Provides a detailed description of the command and how to use it.
Example	Small example of how to use the command.

CMDFILE

Short Description

Defines a command file path, name and status (enable/disable).

Syntax

CMDFILE <file kind> ON|OFF ["<file name and path>"]

and

file kind = STARTUP|RESET|PRELOAD|POSTLOAD

Description

The CMDFILE command is to be used set up a Target Interface Command Files full name and status (disabled/enabled).

This command allows to perform the same settings than using the Command Files dialog through the Command Line component.

The settings of a command file are stored in the ["ICD_PPC"] section of the PROJECT file using variable CMDFILEn.

The list of available command files (and their status) can be get typing CMDFILE without any parameters in the Command Line component.

```
in>CMDFILE
ICD_PPC Target Interface Command Files:
STARTUP ON startup.cmd
RESET ON reset.cmd
PRELOAD ON preload.cmd
POSTLOAD ON postload.cmd
```

The status of the *Startup* command file can be changed:

```
in>CMDFILE STARTUP OFF "my own startup.cmd"
in>CMDFILE
ICD_PPC Target Interface Command Files:
```

```
STARTUP OFF my own startup.cmd
RESET ON reset.cmd
PRELOAD ON preload.cmd
POSTLOAD ON postload.cmd
```

Associated Environment Variables

This section describes the Command Files dialog environment variables which are used by the Target Interface.

“CMDFILEn”

These variables are stored in the **[ICD_PPC]** section from the project file.

Example of the**[ICD_PPC]** target section from the project file:

```
[ICD_PPC]
CMDFILE0=CMDFILE STARTUP ON "startup.cmd"
CMDFILE1=CMDFILE RESET ON "reset.cmd"
CMDFILE2=CMDFILE PRELOAD ON "preload.cmd"
CMDFILE3=CMDFILE POSTLOAD ON "postload.cmd"
```

The following section describes the Command Files dialog environment variables available. The variables are listed in alphabetical order. The Table 6.2 give the variable description format.

Table 6.2 Variable description format

Topic	Description
Short Description	Provides a short description of the variable.
Syntax	Specifies the syntax of the variable in a EBNF format.
Default	Shows the default setting for the variable.
Description	Provides a detailed description of the variable and how to use it.
Example	Small example of how to use the variable.

CMDFILEn

Short Description

Contains a CMDFILE Command Line command to be used to define a command file on a event.

Syntax

CMDFILEn=<command file specified using CMDFILE Command Line command>

Default

All command files are enabled by default.

The default Startup command file is STARTUP.CMD,

The default Reset command file is RESET.CMD,

The default Preload command file is PRELOAD.CMD,

The default Postload command file is POSTLOAD.CMD.

Description

The CMDFILEn variable specifies a command file definition using CMDFILE Command Line command. Four of these entries should be present.

Those variables are used to store the command files status (enable/disable) and full name specified either with the CMDFILE Command Line command or using the Command Files dialog.

```
CMDFILE0=CMDFILE STARTUP ON "startup.cmd"
CMDFILE1=CMDFILE RESET ON "reset.cmd"
CMDFILE2=CMDFILE PRELOAD ON "preload.cmd"
CMDFILE3=CMDFILE POSTLOAD ON "postload.cmd"
```

Index

Symbols
.ABS 16

B
BREAKPOINT 23
Breakpoint
  BREAKPOINT 23
  ILLEGAL_BP 24
  WATCHPOINT 23

C
CMDFILE 59
CMDFILEn 61
COMDEV 28
Command Files dialog 21
Commands 35, 58
Communication Device Specification dialog 17
Communication Parameters 14, 18

D
Default target 25
DER (Debug Enable Register) 19

E
Enter Debug Mode Cause 18
Environment variables 25

F
FLASH 40
FLASH BLANKCHECK 45
FLASH ERASE 46
FLASH IDLE 44
FLASH LOAD 43
FLASH MASSERASE 47
FLASH SETUP 41
FPR_WORKSPACE 29

H
HALTED 22
Hardware Connection 12
Help 16
How To
  Start with a stationery 6
  Start with an example 6

I
ICD_PPC
  Menu Entries 15
  Target Interface Dialogs 16
ICD_PPC Target Interface Command Files dialog 21
ICD_PPC Target Interface Dialogs 16
  Command Files dialog 21
  Communication Device Specification dialog 17
ICD_PPC>Command Files 16, 21
ICD_PPC>Connect... 14, 17
ICD_PPC>Load... 16
ICD_PPC>Reset 16
ICD_PPC>Setup... 17
IO_DELAY_COUNT 34

L
Load... 16
Loading an application 16
Loading the ICD_PPC Target Interface 12
LPT port 18

M
MCU 21
MCU Selection dialog 20
MCUID 30

N
No Link To Target 22

P
P&E PowerPC Interface Cable 11
Port 18
Postload command file 56
Preload command file 55
PROJECT File 12, 25
PROTOCOL 36

R

Release Notes 3
RESET 23
Reset 16
Reset command file 54
RUNNING 22

S

Show Protocol 18
SHOWPROT 32
Startup command file 54
Status Bar Information for the ICD_PPCTarget Interface 22
Status Message 22
- HALTED 22
- No Link To Target 22
- Reset 23
- RUNNING 22
- Target Ready 22

STEPPED 23
STEPPED OVER 23
Stepping and Breakpoints Messages 23
Stepping Message
- STEPPED 23
- STEPPED OVER 23
- STOPPED 23
- TRACED 23

STOPPED 23
support@metrowerks.com (USA) 4
support_europe@metrowerks.com (Europe) 4

T

Target commands 35, 58
Target Interface Command Files dialog 56
Target Interface Dialogs
- Target Interface Command Files dialog 56

Target Ready 22
TRACED 23

U

USERDER 34

V

Variable 25

W

Watchpoint 23
wordwarrior@metrowerks.com 4