Freescale Semiconductor, Inc.

GDI Target Interface

Revised 10/14/2003

metrowerks
Software Starts Here

For More Information: www.freescale.com

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

1

Overview

About this guide

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

- The Introduction section introduces the *GDI* Target Interface concept.
- The Getting Started with CodeWarrior and the HCS08 Serial Monitor, and more... section gives answers for common questions and describes how to use advanced features of the *HCS08 Serial Monitor* via *GDI* Target Interface.
- The Getting Started with CodeWarrior and the SofTec InDART-HCS08, and more... section gives answers for common questions and describes how to use advanced features of the *SofTec inDART-HCS08* via *GDI* Target Interface.
- The Getting Started with CodeWarrior and the SofTec InDART-HC08, and more... section gives answers for common questions and describes how to use advanced features of the *SofTec inDART-HC08* via *GDI* Target Interface.
- The Interfacing Your System with the Target section contains information about the connection between the 3rd party debugging hardware and the debugger.
- The GDI Target Interface Menu Entries section gives a description of the *GDI* Target Interface specific menu entries.
- The GDI Target Interface Dialogs section gives a description of the *GDI* Target Interface specific dialog boxes.
- The Status Bar Information for the GDI Target Interface section describes the status bar messages for the *GDI* Target Interface.
- The GDI Target Interface Default Environment section lists all the variables used by this Target Interface to store the configuration.
- The GDI Target Interface Command Line commands section lists all the commands specific to this Target Interface.
- The Supported GDI DLLs section introduces current GDI DLLs, i.e. the HCS08 Serial Monitor, the SofTec inDART-HCS08 and the SofTec inDART-HC08.

- The Target Interface Commands Files section gives a description of the debugger command private files.
- The Index contains all keywords of the *GDI* Target Interface.

Highlights

- The *GDI* Target Interface allows you to debug with a 3rd party GDI DLL driver interface: The Metrowerks 8/16 bits debugger (and then the Metrowerks CodeWarrior IDE) might be connected to HC08 and HCS08 hardware using the Generic Debug Instrument Interface (Revision 1.2.6). A GDI DLL can be loaded via the GDI.TGT Target Interface. The GDI DLL must be compliant to the "*Metrowerks 8/16 bits debugger Connection to Debug Instrument Using GDI interface protocol* " specification. For more information on GDI DLL implementation and connection to the Metrowerks 8/16 bits debugger, please contact Metrowerks.
-

Requirements

In order to use the *GDI* Target Interface, please make sure that the 3rd Party GDI DLL driver has been installed. The *GDI* Target Interface can then be setup to open the 3rd Party GDI DLL. Note that this DLL can have a different name than GDI.DLL. For more information, please see section Interfacing Your System with the Target.

2

Getting Started with CodeWarrior and the HCS08 Serial Monitor, and more...

Thanks for choosing *CodeWarrior*. This section guides you through installation, licensing/registration and first steps with CodeWarrior and the *HCS08 Serial Monitor* via *GDI* Target Interface. It does not replace all the documentation provided, but gives you a good starting point.

Technical Considerations about GDI Target Interface

The Metrowerks 8/16 bits debugger (and then the Metrowerks CodeWarrior IDE) might be connected to HCS08 hardware using the Generic Debug Instrument Interface (Revision 1.2.6). A GDI DLL can be loaded via the GDI.TGT Target Interface. The GDI DLL must be compliant to the "*Metrowerks 8/16 bits debugger Connection to Debug Instrument Using GDI interface protocol* " specification. For more information on GDI DLL implementation and connection to the Metrowerks 8/16 bits debugger, please contact Metrowerks.

The HCS08 Serial Monitor ***hcs08serialmon.dll*** is a GDI DLL compliant and loadable via the GDI Target Interface in CodeWarrior. This GDI DLL implements communication specifications described in the ***Serial Monitor for MC9S08GB/GT Application Note AN2140/D*** from Motorola, Inc.

When the debugger runs the ***hcs08serialmon.dll*** within the GDI Target Interface, it can communicate and debug hardware running the HCS08 Serial Monitor in full

compliancy to the Motorola ***Serial Monitor for MC9S08GB/GT Application Note AN2140/D*** specifications .

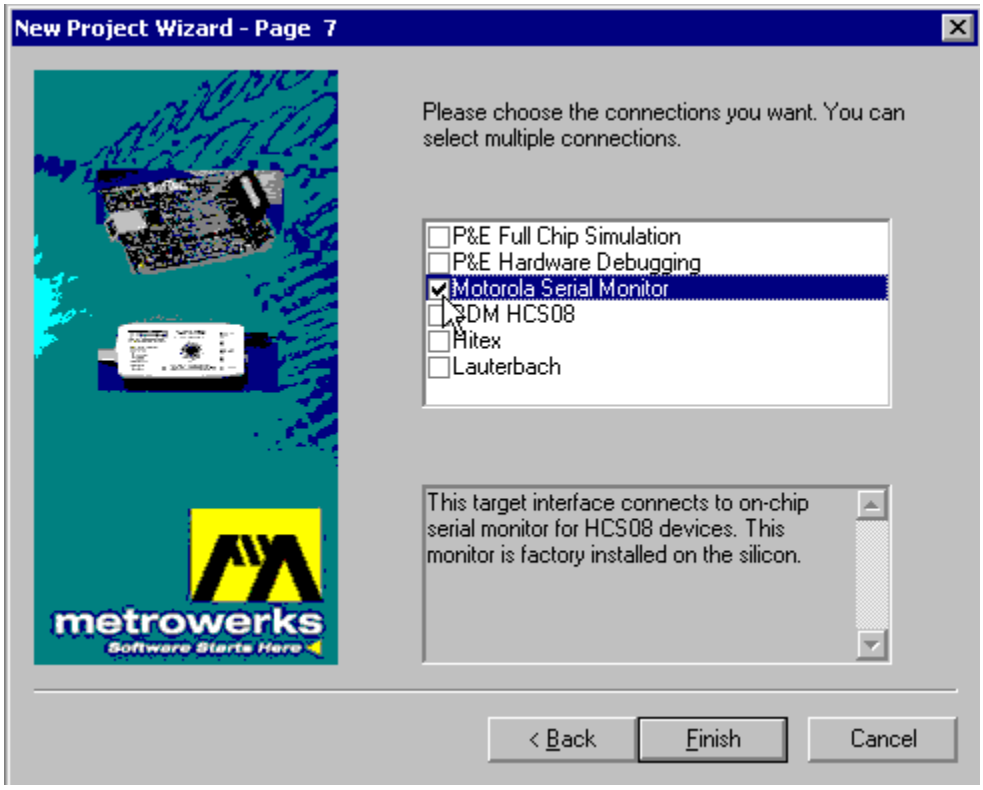
Please refer to this Application Note for communication hardware requirements.

For more detail about the HCS08 Serial Monitor GDI dll, please see HCS08 Serial Monitor section.

First Steps with CodeWarrior and the HCS08 Serial Monitor via GDI Target Interface using the Stationery Wizard

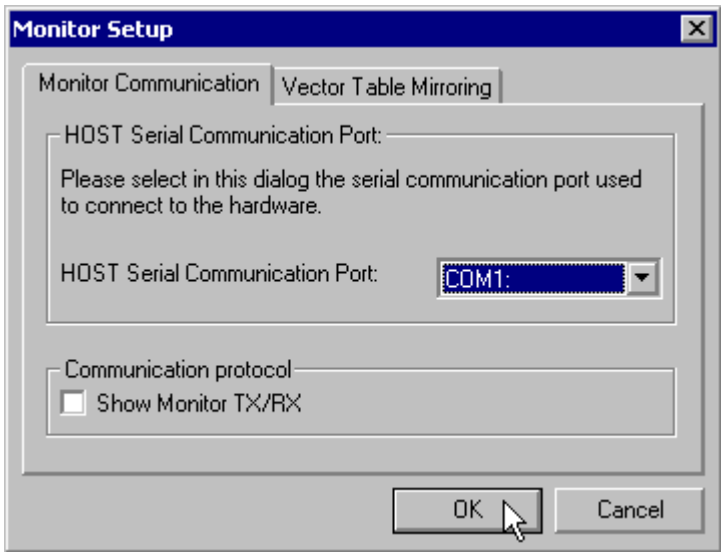
1. Run the *CodeWarrior IDE* with the shortcut created in the program group.
2. Choose the menu **File > New** to create a new project from a stationery.
3. Select *HC(S)08 New Project Wizard*, type in a project name and specify the project location. Press *OK*.
4. Please follow all wizard steps and make sure to select the **Motorola Serial Monitor** as connection type.

Figure 2.1 Wizzard Connection Selection



5. Choose the menu **Project > Debug** to start the debugger.
6. Now in the Monitor Setup dialog, choose the correct Host serial communication port if necessary.

Figure 2.2 Monitor Setup Dialog

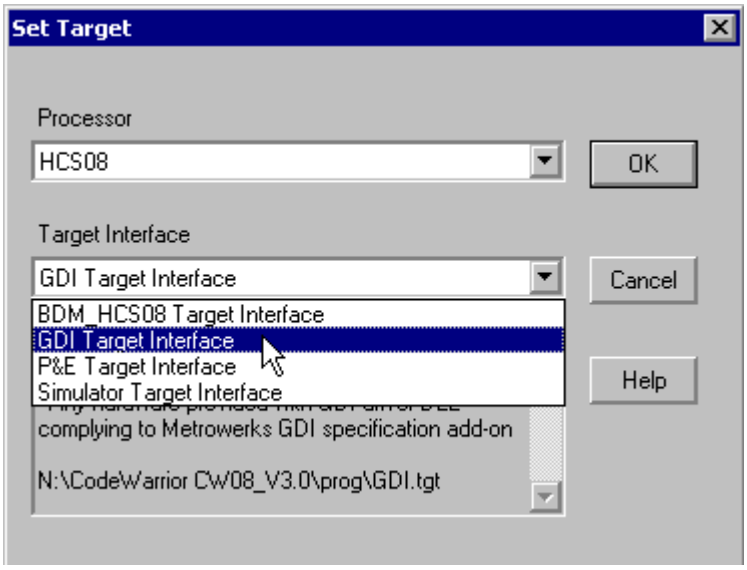


7. Press the OK button to start debugging.

First Steps with CodeWarrior and setting HCS08 Serial Monitor via GDI Target Interface within an existing project

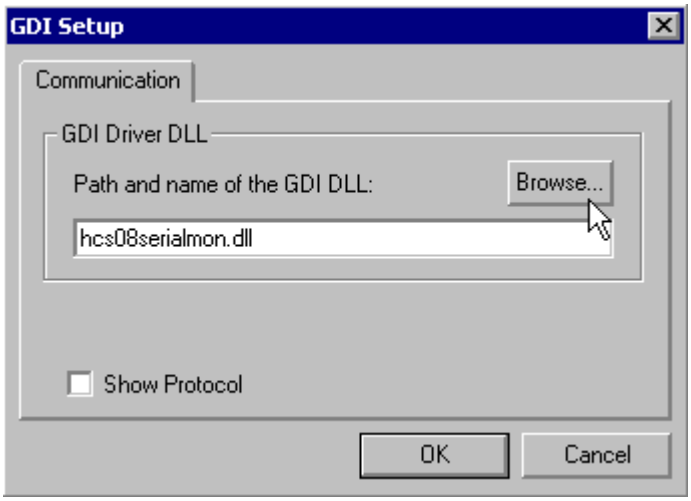
1. Run the *CodeWarrior IDE* with the shortcut created in the program group.
2. Open the project.
3. Choose the menu **Project > Debug** to start the debugger.
4. Choose in the debugger menu **Component > Set Target..** to select another target interface.
5. Select ***HCS08*** as Processor then ***GDI Target Interface*** as Target Interface.

Figure 2.3 GDI Target Interface Selection



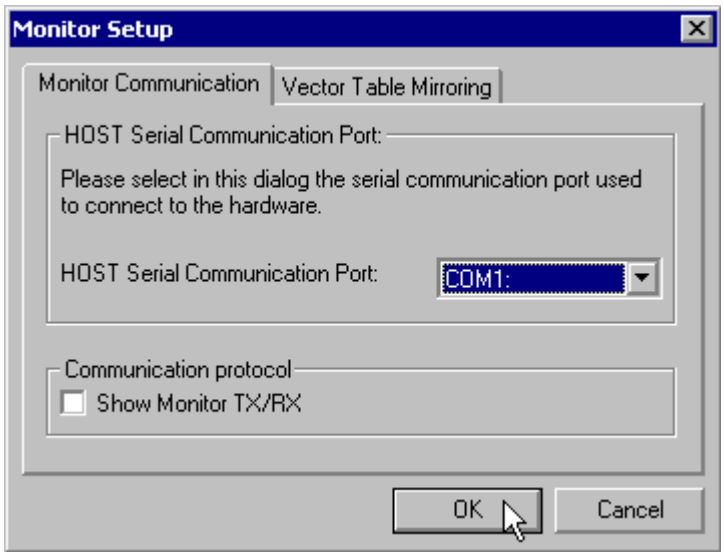
6. Now in the GDI Setup dialog, press the ***Browse*** button and browse for the ***hcs08serialmon*** dll file.

Figure 2.4 Browse for HCS08SerialMon GDI DLL



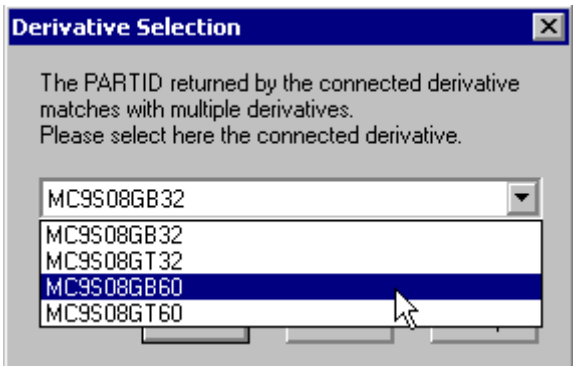
7. Now in the Monitor Setup dialog, choose the correct Host serial communication port if necessary.

Figure 2.5 Monitor Setup Dialog



8. Press the OK button. The HCS08 Serial Monitor DLL reads the device silicon ID. This ID can match several derivatives. Please set the correct derivative matching with your hardware.

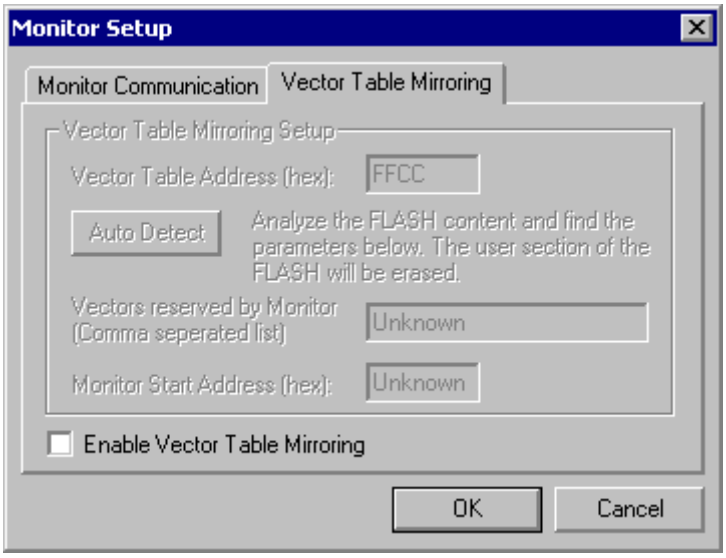
Figure 2.6 Derivative Selection Dialog



9. Press the OK button. The **Monitor Setup** dialog is opened again, to propose to use the "mirrored vector table" feature. Please see section ***Vector Redirection*** in ***Serial Monitor for MC9S08GB/GT Application Note AN2140/D*** for all details. Please see also Monitor Setup Dialog section for further details.

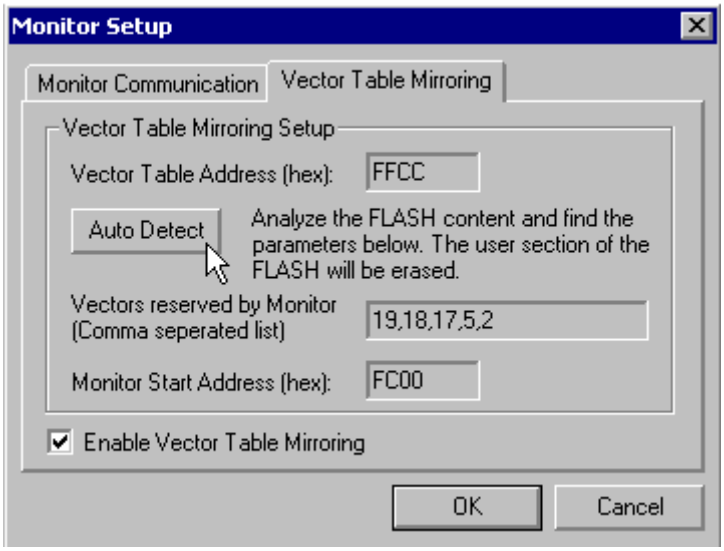
NOTE	It is recommanded to use this feature, otherwise vectors cannot be programmed, as captured and protected from erasing/writing by the HCS08 Serial Monitor.

Figure 2.7 Vector Table Mirroring Tab



10. To enable this specific feature, check the "Enable Vector Table Mirroring" checkbox. ¨

Figure 2.8 Vector Table Mirroring Tab



11. Press the “Auto Detect” button to make the debugger search for the vector table address and vectors reserved by the HCS08 Serial Monitor.
12. Once the autodetection succeeded, press the OK button to start debugging.

3

Getting Started with CodeWarrior and the SofTec InDART-HCS08, and more...

Thanks for choosing *CodeWarrior*. This section guides you through installation, licensing/registration and first steps with CodeWarrior and the *SofTec inDART-HCS08* via *GDI* Target Interface. It does not replace all the documentation provided, but gives you a good starting point.

Technical Considerations about GDI Target Interface

The Metrowerks 8/16 bits debugger (and then the Metrowerks CodeWarrior IDE) might be connected to HCS08 hardware using the Generic Debug Instrument Interface (Revision 1.2.6). A GDI DLL can be loaded via the GDI.TGT Target Interface. The GDI DLL must be compliant to the "*Metrowerks 8/16 bits debugger Connection to Debug Instrument Using GDI interface protocol* " specification. For more information on GDI DLL implementation and connection to the Metrowerks 8/16 bits debugger, please contact Metrowerks.

The ***SofTec_BDC08.dll*** is a GDI DLL compliant and loadable via the GDI Target Interface in CodeWarrior. This GDI DLL is the driver for *SofTec inDART-HCS08* in-circuit debugger/programmer unit.

When the debugger runs the ***SofTec_BDC08.dll*** within the GDI Target Interface, it can communicate and debug **HCS08** core based hardware connected throught the SofTec in-circuit debugger/programmer unit .

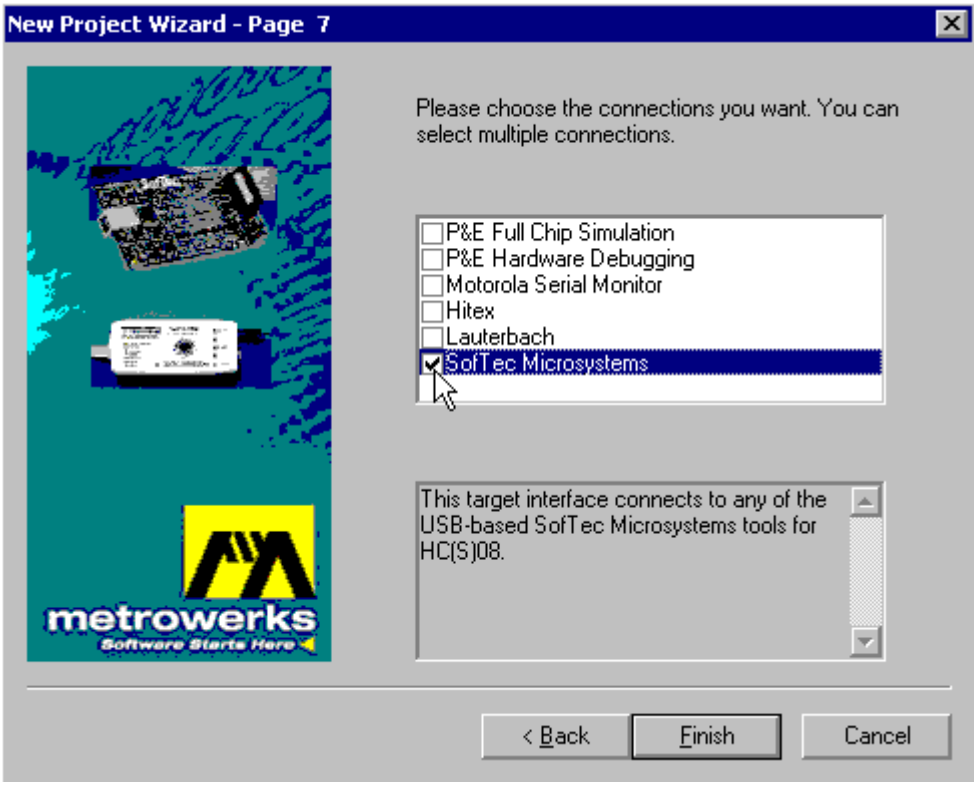
Please refer to ***"inDART®-HCS08 In-Circuit Debugger/Programmer for Motorola HCS08 Family FLASH Devices User's Manual"*** from SofTec for communication hardware requirements and SofTec product installation.

For more detail about the HCS08 Serial Monitor GDI dll, please see SofTec inDART-HCS08 section.

First Steps with CodeWarrior and the SofTec inDART-HCS08 via GDI Target Interface using the Stationery Wizard

1. Run the *CodeWarrior IDE* with the shortcut created in the program group.
2. Choose the menu **File > New** to create a new project from a stationery.
3. Select ***HC(S)08 New Project Wizard***, type in a project name and specify the project location. Press *OK*.
4. Please follow all wizard steps and make sure to select the ***SofTec Microsystems*** as connection.

Figure 3.1 Wizzard Connection Selection

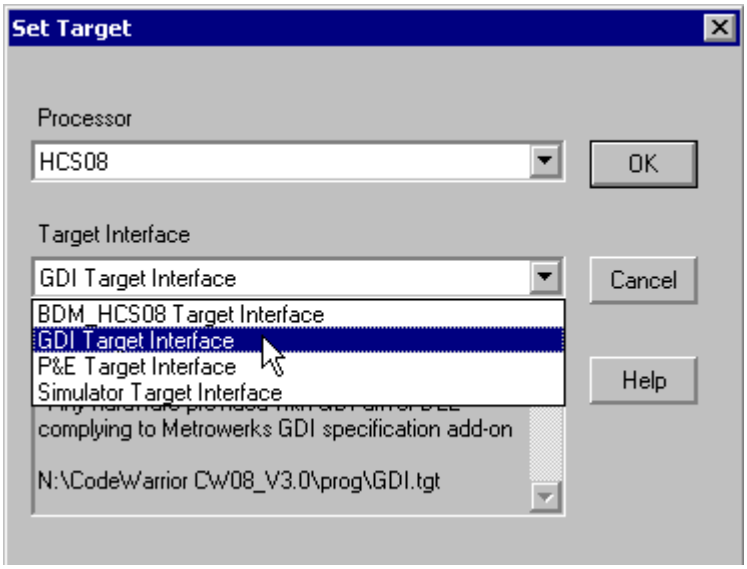


5. Choose the menu **Project > Debug** to start the debugger.

First Steps with CodeWarrior and setting SofTec inDART-HCS08 via GDI Target Interface within an existing project

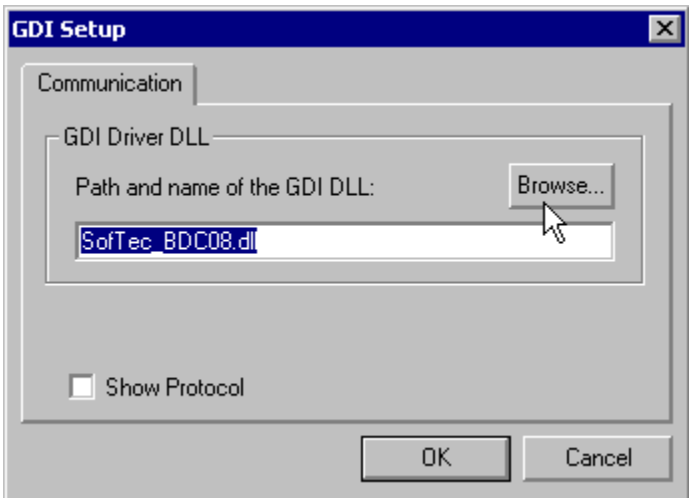
1. Run the *CodeWarrior IDE* with the shortcut created in the program group.
2. Open the project.
3. Choose the menu **Project > Debug** to start the debugger.
4. Choose in the debugger menu **Component > Set Target..** to select another target interface.
5. Select ***HCS08*** as Processor then ***GDI Target Interface*** as Target Interface.

Figure 3.2 GDI Target Interface Selection



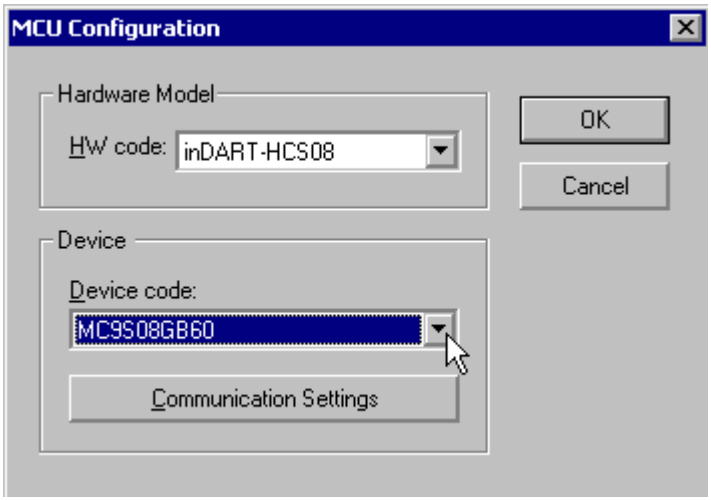
6. Now in the GDI Setup dialog, press the browse button and browse for the ***SofTec_BDC08*** dll file. Search for the ***SofTec_BDC08*** dll where SofTec applications and drivers are installed.

Figure 3.3 Browse for SofTec_BDC08.dll GDI DLL



7. When the ***SofTec_BDC08*** dll is found, press OK. Please choose in the MCU Configuration dialog the correct target processor.

Figure 3.4 MCU Configuration



8. Press the OK button to start debugging.

4

Getting Started with CodeWarrior and the SofTec InDART-HC08, and more...

Thanks for choosing *CodeWarrior*. This section guides you through installation, licensing/registration and first steps with CodeWarrior and the *SofTec inDART-HC08* via *GDI* Target Interface. It does not replace all the documentation provided, but gives you a good starting point.

Technical Considerations about GDI Target Interface

The Metrowerks 8/16 bits debugger (and then the Metrowerks CodeWarrior IDE) might be connected to HC08 hardware using the Generic Debug Instrument Interface (Revision 1.2.6). A GDI DLL can be loaded via the GDI.TGT Target Interface. The GDI DLL must be compliant to the "*Metrowerks 8/16 bits debugger Connection to Debug Instrument Using GDI interface protocol* " specification. For more information on GDI DLL implementation and connection to the Metrowerks 8/16 bits debugger, please contact Metrowerks.

The ***inDART_HC08.dll*** is a GDI DLL compliant and loadable via the GDI Target Interface in CodeWarrior. This GDI DLL is the driver for *SoftTec inDART_HC08* in-circuit debugger/programmer unit.

When the debugger runs the ***inDART_HC08.dll*** within the GDI Target Interface, it can communicate and debug **CPU08 (HC08)** core based hardware connected throught the SofTec in-circuit debugger/programmer unit.

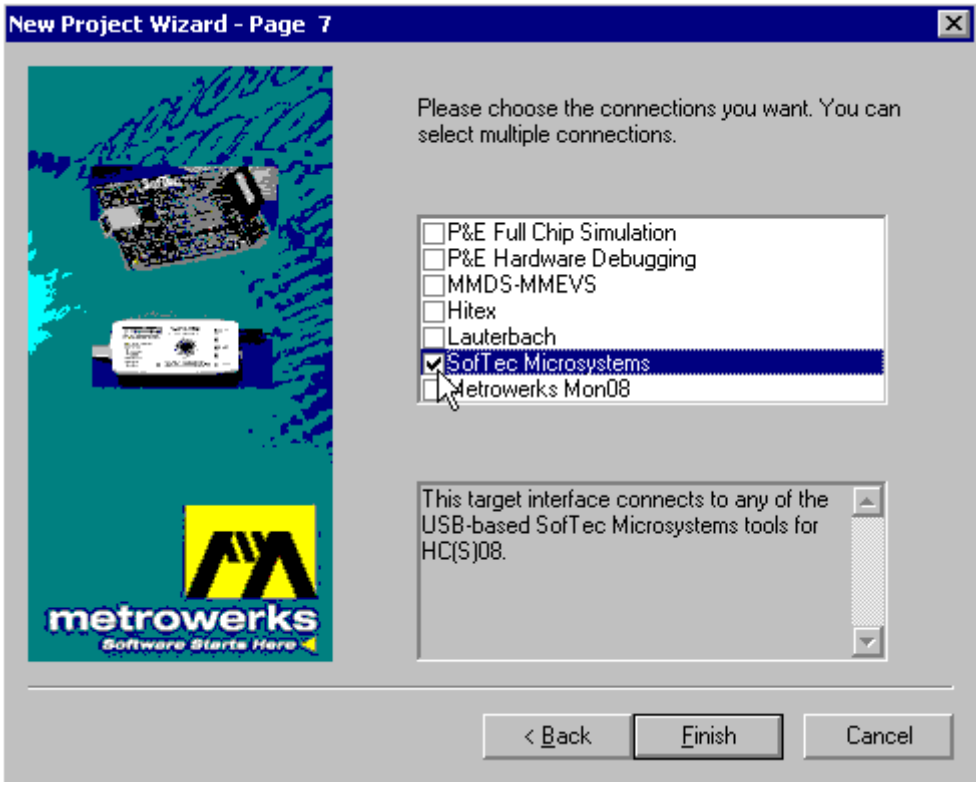
Please refer to "***inDART®-HC08 In-Circuit Debugger/Programmer for Motorola HC08 Family FLASH Devices User's Manual***" from SofTec for communication hardware requirements and SofTec product installation.

For more detail about the inDART-HC08 GDI dll, please see SofTec inDART-HCS08 section.

First Steps with CodeWarrior and the SofTec inDART-HC08 via GDI Target Interface using the Stationery Wizard

1. Run the *CodeWarrior IDE* with the shortcut created in the program group.
2. Choose the menu **File > New** to create a new project from a stationery.
3. Select ***HC(S)08 New Project Wizard***, type in a project name and specify the project location. Press *OK*.
4. Please follow all wizard steps and make sure to select ***SofTec Microsystems*** as connection.

Figure 4.1 Wizzard Connection Selection

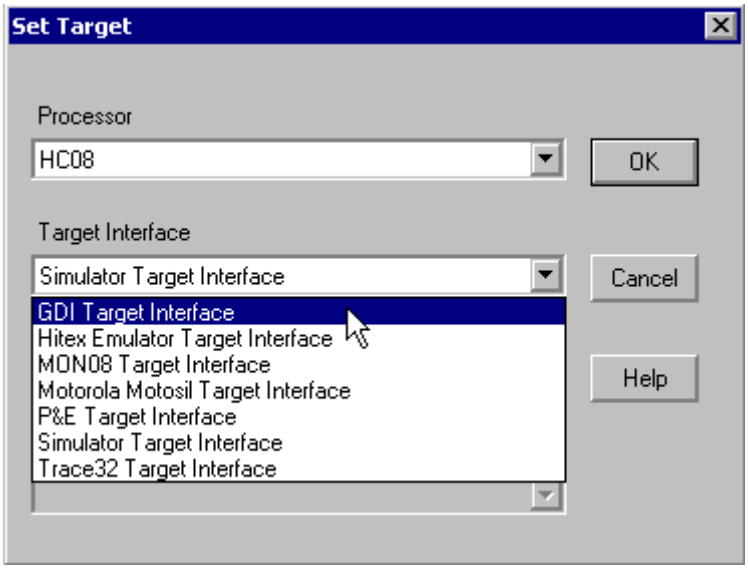


5. Choose the menu **Project > Debug** to start the debugger.

First Steps with CodeWarrior and setting SofTec inDART-HC08 via GDI Target Interface within an existing project

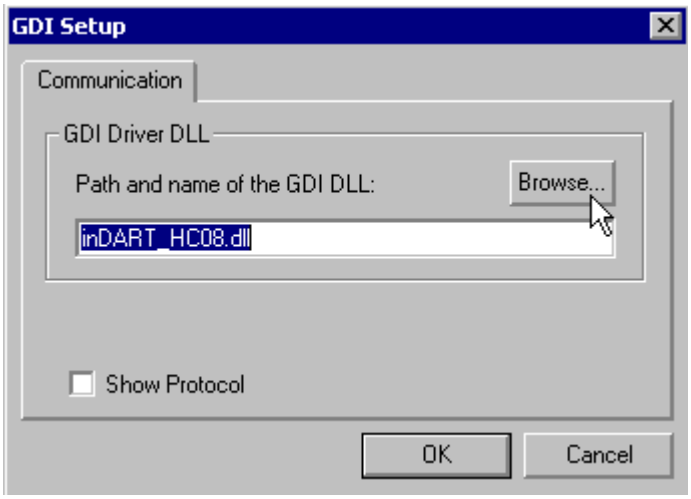
1. Run the *CodeWarrior IDE* with the shortcut created in the program group.
2. Open the project.
3. Choose the menu **Project > Debug** to start the debugger.
4. Choose in the debugger menu **Component > Set Target..** to select another target interface.
5. Select ***HC08*** as Processor then ***GDI Target Interface*** as Target Interface.

Figure 4.2 GDI Target Interface Selection



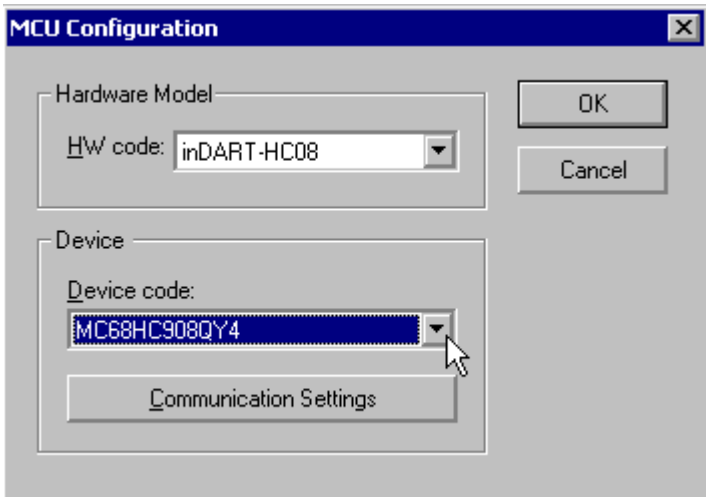
6. Now in the GDI Setup dialog, press the browse button and browse for the ***inDART_HC08*** dll file. Search for the ***inDART_HC08*** dll where SofTec applications and drivers are installed.

Figure 4.3 Browse for inDART_HC08.dll GDI DLL



7. When the ***inDART_HC08*** dll is found, press OK. Please choose in the MCU Configuration dialog the correct target processor.

Figure 4.4 MCU Configuration



8. Press the OK button to start debugging.

5

GDI Target Interface Manual

Introduction

An advanced feature of *Metrowerks debugger* 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 *GDI* Target Interface are described.

The Metrowerks 8/16 bits debugger (and then the Metrowerks CodeWarrior IDE) might be connected to HC08 and HCS08 hardware using the Generic Debug Instrument Interface (Revision 1.2.6). A GDI DLL can be loaded via the GDI.TGT Target Interface. The GDI DLL must be compliant to the "*Metrowerks 8/16 bits debugger Connection to Debug Instrument Using GDI interface protocol* " specification. For more information on GDI DLL implementation and connection to the Metrowerks 8/16 bits debugger, please contact Metrowerks.

With this interface, you can download an executable program from the *Metrowerks CodeWarrior HC08 Studio*, to an external target system based on a *Motorola HC08* or *HCS08*, which will execute the program. You will also have the feedback of the real target system behavior to *Metrowerks debugger*.

The debugger will fully supervise and monitor the MCU of the target system i.e. control the CPU execution. You can read and write in internal/external memory when the MCU is in Background Mode. You have full control over the CPU state with the possibility to stop execution, to proceed in single step mode and to set breakpoints in the code.

Interfacing Your System with the Target

Hardware Connection

The GDI Target Interface is by definition a Generic Debug Instrument interface. There is no specific hardware cable or else for this interface. Indeed, the debugging hardware is entirely bound to the GDI DLL loaded by the GDI target. Please refer to the GDI DLL designer/provider/3rd party to get details about hardware connectings and setup.

Loading the GDI Target Interface

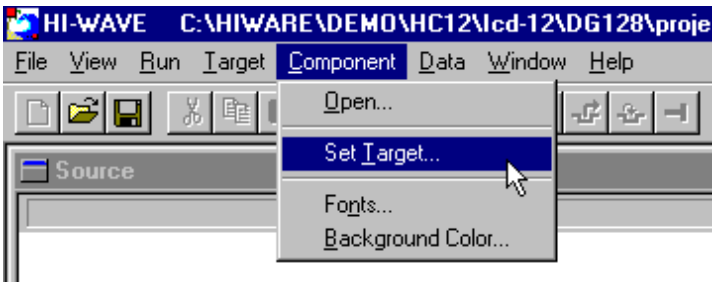
Usually the target is set in the **`[HI-WAVE]`** section of the `PROJECT` file, through the statement **`Target=GDI`**.

In this way, the *GDI* Target Interface detects automatically that the target is connected to your system.

If no target is set in the `PROJECT` file or if a different target is set, you can load the *GDI* Target Interface.

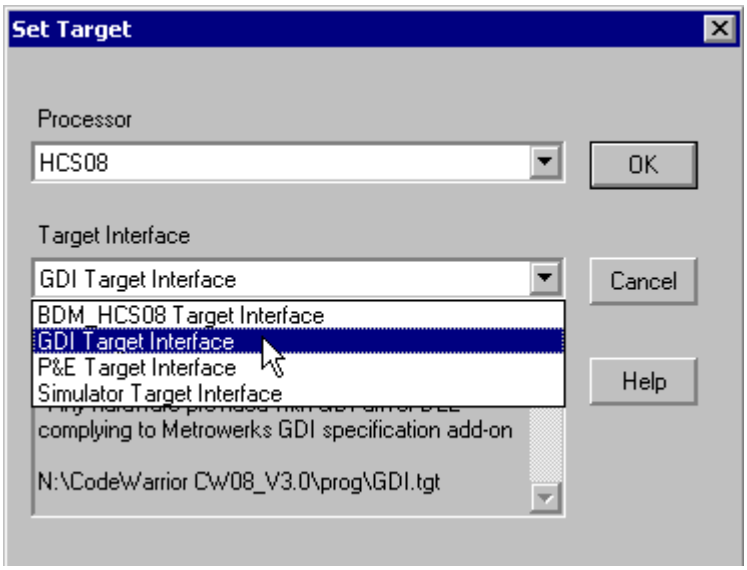
Select in the main menu **Component > Set Target...**, as shown in *Figure 3.3*.

Figure 5.1 Set Target dialog



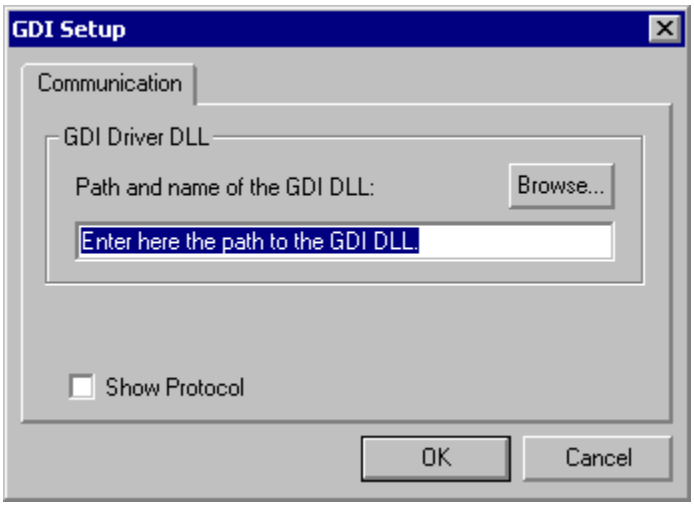
The *Set Target* dialog is displayed. Choose *GDI Target Interface* in the list of proposed targets and click *OK*.

Figure 5.2 List of available targets interfaces



Once the GDI Target interface is loaded, the GDI Setup dialog is opened. A GDI DLL must be specified in the “Path and name of the GDI DLL” edit box. Pressing the Browse button opens a typical Windows file browser.

Figure 5.3 GDI Setup dialog

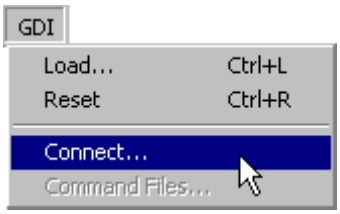


Once the GDI DLL is specificed, press OK then follow specific setup requirements of the loaded GDI DLL.

GDI Target Interface Menu Entries

After loading the *GDI* Target Interface, the *Target* menu item is replaced by *GDI*. The different entries of the *GDI* menu are described below:

Figure 5.4 GDI Menu entries.



Load...

Choose **GDI > Load...** to load the application to debug, i.e. e.g. a **.abs** file. Note that this operation is only possible when a GDI DLL has been loaded. Also, in this case, the “GDI” menu entry gets the name of the GDI DLL loaded.

Reset

The menu entry **GDI > Reset** executes the Reset Command File and resets the target system processor.

Connect...

Select entry **GDI > Connect...** to display the GDI Setup dialog.

Command Files

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

GDI Target Interface Dialogs

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

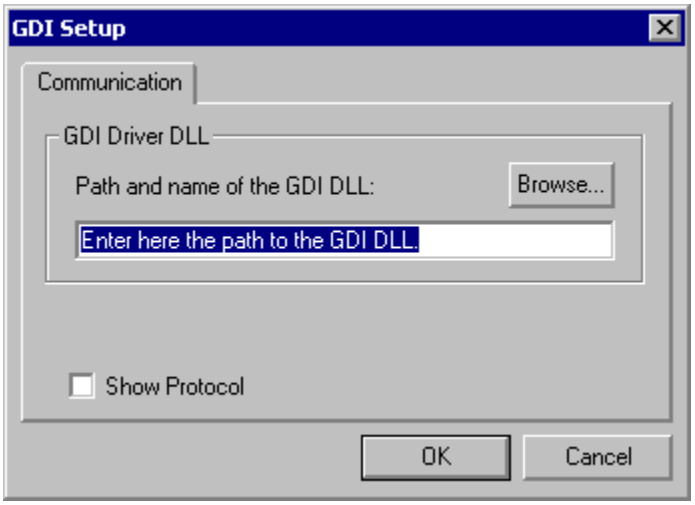
Those dialogs are:

- the GDI Setup dialog,
- the *GDI* Target Interface Command Files dialog,

GDI Setup

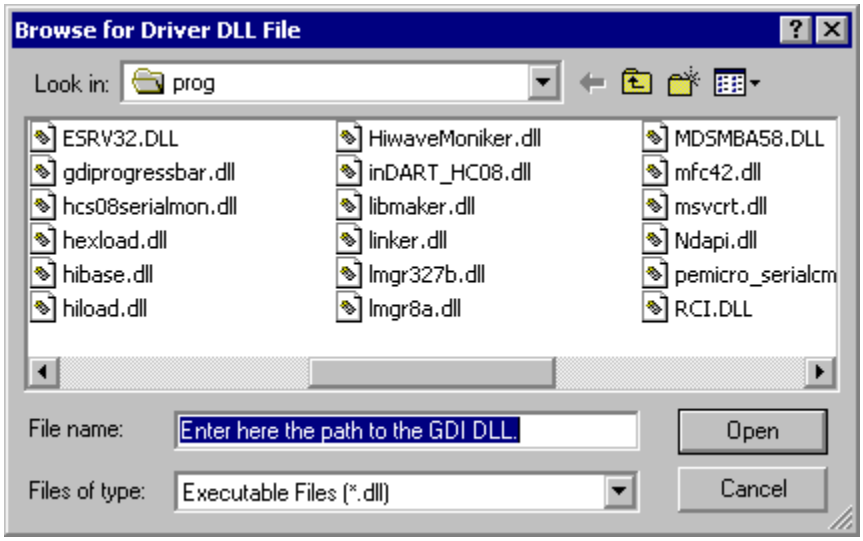
Select entry **GDI > Connect...** to display the *GDI Setup* dialog.

Figure 5.5 GDI Setup dialog.



A GDI DLL must be specified in the “Path and name of the GDI DLL” edit box. Pressing the Browse button opens a typical Windows file browser.

Figure 5.6 GDI DLL browser.



Status Bar Information for the GDI Target Interface

When a GDI DLL has been loaded, specific information belonging to the GDI dll are displayed in the debugger's status bar. If no GDI dll is loaded, "No link to Target" default message is displayed.

Figure 5.7 Status bar GDI default message

Status Messages

GDI ready

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

No Link To Target

No GDI dll is currently loaded or currently loaded GDI dll connection to the target system has failed.

RUNNING

The application is currently executing on the target.

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 *Metrowerks debugger* has been reset on user request. The menu entry **GDI > 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.

GDI Target Interface Default Environment

Default Target Setup

As any other target, the *GDI* Target Interface can be loaded from the *Target* menu or can be set as a default target in the `PROJECT` 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, yo can load the *GDI* Target Interface interactively. Please refer to section *Loading the GDI Target Interface* of this manual.

Listing 5.1 Example of `[HI-WAVE]` section from `PROJECT` file:

```
[HI-WAVE]
Window0=Source      0    0   60   30
Window1=Assembly   60    0   40   30
Window2=Procedur    0   30   60   25
Window3=Register   60   30   40   30
Window4=Memory     60   60   40   40
Window5=Data        0   55   60   23
Window6=Data        0   78   60   22
Target=GDI
```

NOTE Please see the Manual Engine HC08.pdf for further information about the PROJECT file.

GDI Target Interface Environment Variables

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

The *GDI* Target Interface specific environment variables are:

- COMSETTINGS

Some other common GDI dlls features are using variables. Those variables are:

- CMDFILEn

These variables are stored in the **[GDI]** section from the PROJECT file.

Listing 5.2 Example of [GDI] section from PROJECT file:

```
[GDI]
COMSETTINGS=SETCOMM DRIVER NOPROTOCOL NOPERIODICAL "hcs08serialmon.dll"
```

COMSETTINGS

Short Description

Communication device settings

Syntax

```
COMSETTINGS==SETCOMM DRIVER <PROTOCOL|NOPROTOCOL>
<PERIODICAL|NOPERIODICAL> "<GDI DLL file (and path)>"
```

Alias

None

File

PROJECT file

Section

[GDI]

Components

GDI Target Interface.

Description

The communication port to be used on the host computer can be specified using the variable **COMSETTINGS**.

If **PROTOCOL** is specified, all the commands and responses sent and received are reported in the *Command Line* component of the debugger. If **NOPROTOCOL** is specified, no protocol is reported.

If **PERIODICAL** and **NOPERIODICAL** are not used in the GDI target interface.

All parameters are set according to the setup in the GDI Setup dialog.

For example,

```
COMSETTINGS=SETCOMM DRIVER NOPROTOCOL NOPERIODICAL
"hcs08serialmon.dll"
```

GDI Target Interface Command Line commands

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

The *GDI* Target Interface specific commands are:

- PROTOCOL

Some other common Target Interface features are using commands. Those commands are:

- CMDFILE

Those commands can be entered in the file or in the *Command Line* component of the debugger.

PROTOCOL

Short Description

switch on/off the *Show Protocol* functionality

Syntax

```
PROTOCOL ON|OFF
```

Alias

None

Components

GDI Target Interface.

Description

If this command is used, all the messages sent to and received from the *GDI* interface are reported in the *Command Line* window of the debugger.

For Example

```
PROTOCOL ON
```

TIP

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

6

Supported GDI DLLs

HCS08 Serial Monitor

The HCS08 Serial Monitor ***hcs08serialmon.dll*** is a GDI DLL compliant and loadable via the GDI Target Interface in CodeWarrior. This GDI DLL implements communication specifications described in the *HCS08 Serial Monitor* Application Note from Motorola, Inc.

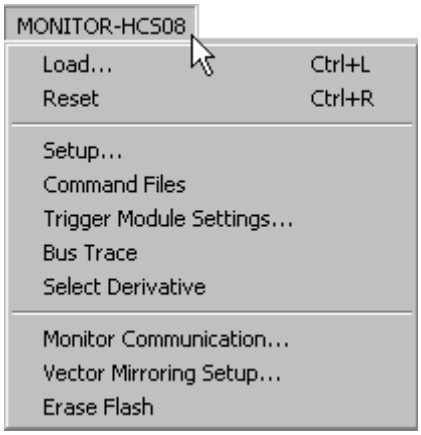
When the debugger runs the ***hcs08serialmon.dll*** within the GDI Target Interface, it can communicate and debug hardware running the HCS08 Serial Monitor in full compliancy to the Motorola *HCS08 Serial Monitor* Application Note specifications .

Please refer to this Application Note for communication hardware requirements.

Menu Entries

Once the ***hcs08serialmon.dll*** is loaded in the GDI target interface, the "*GDI*" menu entry is replaced by "*MONITOR-HCS08*" .

Figure 6.1 Menu Entries



Only specific HCS08 Serial Monitor entries are commented here. Regular menu entries are introduce in GDI Target Interface Menu Entries.

Monitor Communication...

Select entry **MONITOR-HCS08> Monitor Communications...** to display the Monitor Setup Dialog on *Monitor Communication* tab.

Vector Mirroring Setup...

Select entry **MONITOR-HCS08> Vector Mirroring Setup...** to display the Monitor Setup Dialog on *Vector Table Mirroring* tab.

Erase Flash

Select entry **MONITOR-HCS08> Erase Flash** to force immediate mass erase of the target processor flash.

Trigger Module Settings...

Select entry **MONITOR-HCS08> Trigger Module Settings...** to open the ***Trigger Module Settings*** dialog. Please refer to the "***Debugger HCS08 Onchip DBG Module User Interface***" manual to get find all related information.

Bus Trace

Select entry **MONITOR-HCS08> Bus Trace** to open the Trace window component. Please refer to the "***Debugger HCS08 Onchip DBG Module User Interface***" manual to get find all related information.

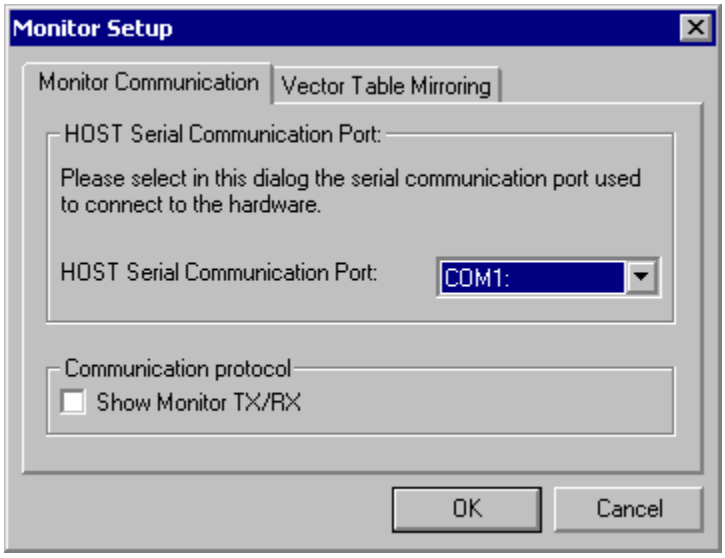
Select Derivative

Select entry **MONITOR-HCS08> Select Derivative** to open the Derivative Selection Dialog.

Monitor Setup Dialog

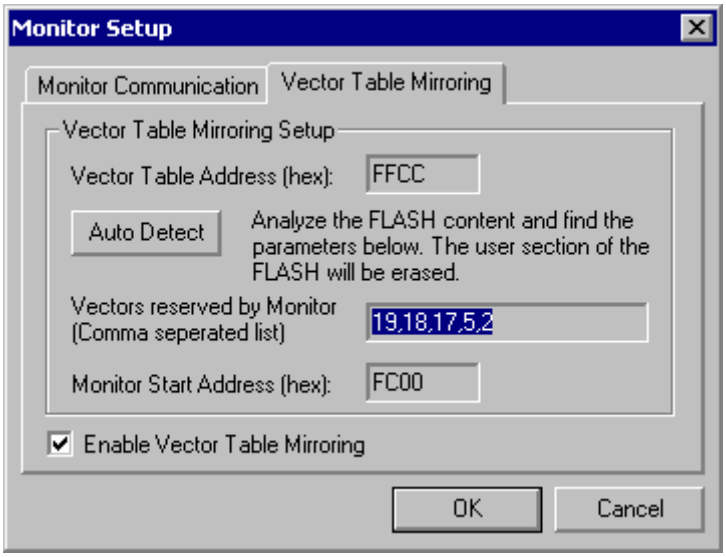
The *Monitor Setup* dialog has two tabs, has shown below.

Figure 6.2 Monitor Communication tab



Within this tab, it is possible to set or modify the current serial communication port when opening the "*HOST Serial Communication Port*" drop down list. Checking "*Show Monitor TX/RX*" reports in the debugger Command Line window all low level communication frames between the host computer and the HCS08 Serial Monitor.

Figure 6.3 Vector Table Mirroring tab



Within this tab, it is possible to set the “Vector Table Mirroring” feature. Please see section ***Vector Redirection*** in ***Serial Monitor for MC9S08GB/GT Application Note AN2140/D*** for all details.

Vector mirroring allows accessing chip vectors transparently. Indeed, the HCS08 Serial Monitor also use some vectors, and the vector area is protected from erasing and writing. It is therefore recommanded to use this feature, otherwise user application vectors cannot be programmed, as captured and protected from erasing/writing by the HCS08 Serial Monitor.

To enable this specific feature, check the “Enable Vector Table Mirroring” checkbox, then press the “Auto Detect” button to make the debugger search for the vector table address and vectors reserved by the HCS08 Serial Monitor. Once the autodetection succeeded, press the OK button to save and quit this dialog.

The HCS08 Monitor start address is given in the ***Monitor Start Address*** edit box.

The real vector table address is given in the ***Vector Table Address*** edit box.

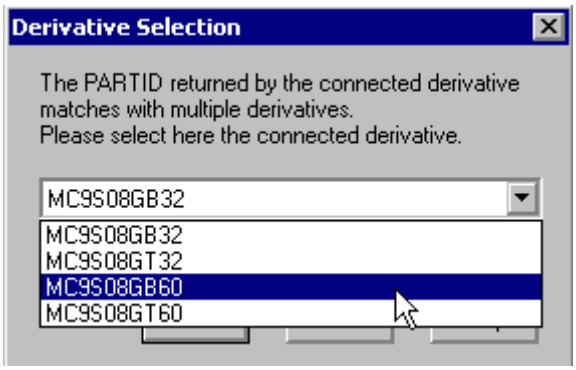
The list of vectors reserved by the HCS08 Serial Monitor is given in the ***Vectors reserved by Monitor*** edit box.

NOTE	The number “1” matches the **RESET** vector. In the list in Figure here above, “2” is the **SWI** vector, “5” is the **ICG** vector, etc.

Derivative Selection Dialog

Within this dialog, it is possible to select a specific derivative according to the *System Device Identification Register* (SDIDH, SDIDL) (also sometimes called PARTID) returned by the silicon device. As several silicon devices might return the same value, a selection list is available to select exactly (according to text reference written on the top of the silicon) the debugged derivative.

Figure 6.4 Derivative Selection Dialog



SofTec inDART-HCS08

The ***SofTec_BDC08.dll*** is a GDI DLL compliant and loadable via the GDI Target Interface in CodeWarrior. This GDI DLL is the driver for *SofTec inDART-HCS08* in-circuit debugger/programmer unit.

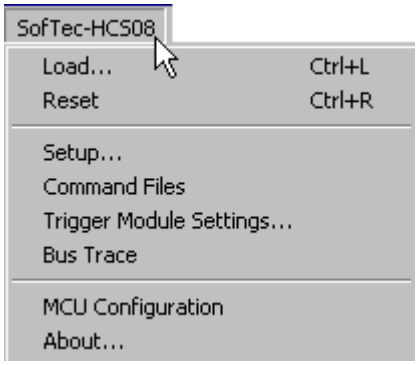
When the debugger runs the ***SofTec_BDC08.dll*** within the GDI Target Interface, it can communicate and debug **HCS08** core based hardware connected throught the SofTec in-circuit debugger/programmer unit .

Please refer to ***"inDART®-HCS08 In-Circuit Debugger/Programmer for Motorola HCS08 Family FLASH Devices User's Manual"*** from SofTec for communication hardware requirements and SofTec product installation.

Menu Entries

Once the ***SofTec_BDC08.dll*** is loaded in the GDI target interface, the "*GDI*" menu entry is replaced by "***SofTec-HCS08***" .

Figure 6.5 Menu Entries



Only specific SofTec inDART-HCS08 entries are commented here. Regular menu entries are introduce in GDI Target Interface Menu Entries.

MCU Configuration

Select entry **SofTec-HCS08 > MCU Configuration** to display the MCU Configuration dialog.

About

Select entry **SofTec-HCS08 > About...** to display the About dialog .

Trigger Module Settings

Select entry **SofTec-HCS08 > Trigger Module Settings...** to force immediate mass erase of the target processor flash. Please refer to the "***Debugger HCS08 Onchip DBG Module User Interface***" manual to get find all related information.

Bus Trace

Select entry **SofTec-HCS08 > Bus Trace** to open the Trace window component. Please refer to the "***Debugger HCS08 Onchip DBG Module User Interface***" manual to get find all related information.

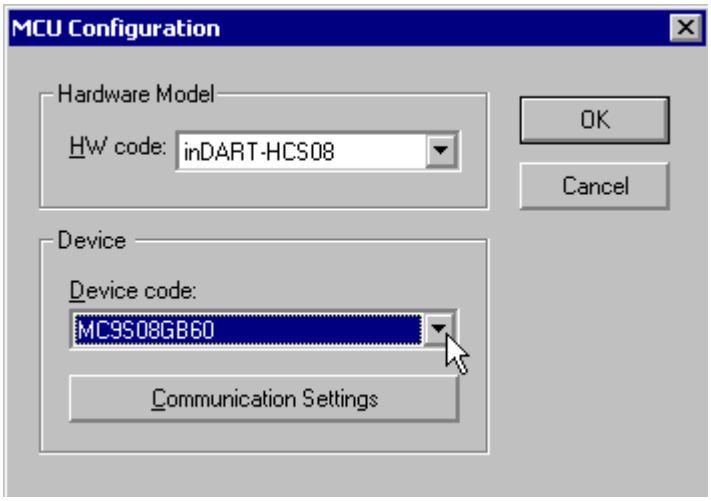
MCU Configuration dialog

The Hardware Model drop down list can be expanded to select another type of BDC debug interface than the SofTec inDART-HCS08. Note that at this document release time, only the SofTec inDART-HCS08 is available.

The *HW Code* drop down list can be expanded to select another HCS08 derivative.

Pressing the *Communication Settings* button opens the Communication Settings dialog.

Figure 6.6 MCU Configuration dialog



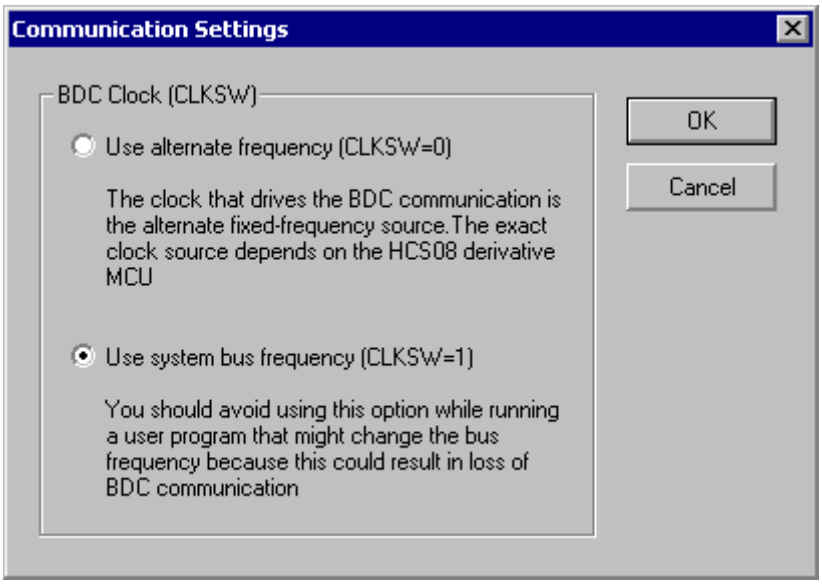
Communication Settings dialog

The *BDC Clock (CLKSW)* group is intended to setup the best BDC synchronization between the *SofTec inDART-HCS08* interface and the target processor.

When "***Use system bus frequency***" is selected, the BDC communication clock source is the microcontroller's bus frequency; when "***Use alternate frequency***" is selected, the BDC communication clock source is a constant clock source, which can vary depending on the specific HCS08 derivative. In the case of the MC9S08GB60, for example, this constant clock source is an 8 MHz internal clock. Other derivatives may use the external crystal frequency.

Please refer to ***"inDART®-HCS08 In-Circuit Debugger/Programmer for Motorola HCS08 Family FLASH Devices User's Manual"*** from SofTec for further details.

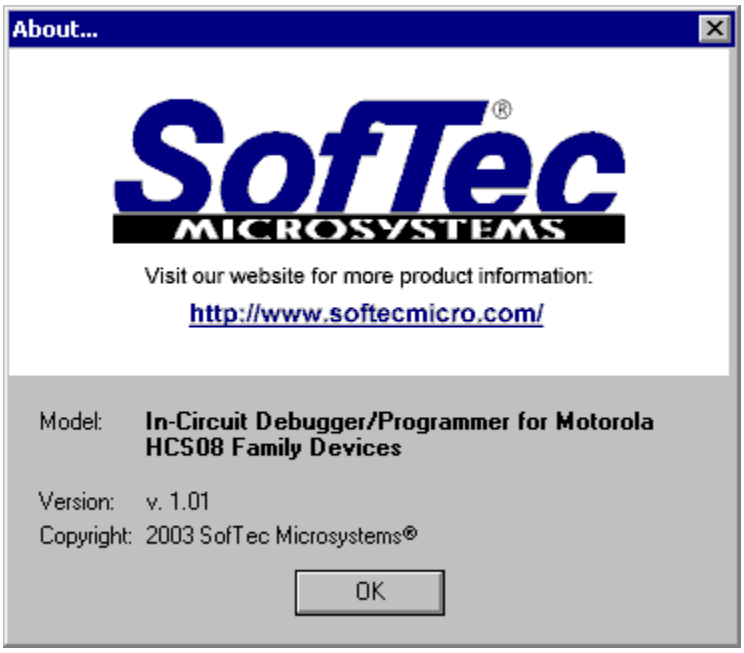
Figure 6.7 Communication Settings dialog



About dialog

This dialog belongs to the SofTec GDI dll and provides information about the ***SofTec_BDC08.dll*** release and version.

Figure 6.8 About dialog



SofTec inDART-HC08

The ***inDART_HC08.dll*** is a GDI DLL compliant and loadable via the GDI Target Interface in CodeWarrior. This GDI DLL is the driver for *SofTec inDART-HC08* in-circuit debugger/programmer unit.

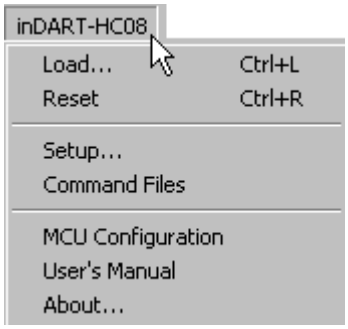
When the debugger runs the ***inDART_HC08.dll*** within the GDI Target Interface, it can communicate and debug **CPU08 (HC08)** core based hardware connected throught the SofTec in-circuit debugger/programmer unit .

Please refer to ***"inDART®-HC08 In-Circuit Debugger/Programmer for Motorola HC08 Family FLASH Devices User's Manual"*** from SofTec for communication hardware requirements and SofTec product installation.

Menu Entries

Once the ***inDART_HC08.dll*** is loaded in the GDI target interface, the "*GDI*" menu entry is replaced by "***inDART-HC08***" .

Figure 6.9 Menu Entries



Only specific SofTec inDART-HC08 entries are commented here. Regular menu entries are introduce in GDI Target Interface Menu Entries.

MCU Configuration

Select entry **inDART-HC08 > MCU Configuration** to display the MCU Configuration dialog .

User's Manual

Select entry **inDART-HC08 > User's Manual** to open the ***"inDART®-HC08 In-Circuit Debugger/Programmer for Motorola HC08 Family FLASH Devices User's Manual"*** from SofTec.

About

Select entry **inDART-HC08 > About...** to display the About dialog .

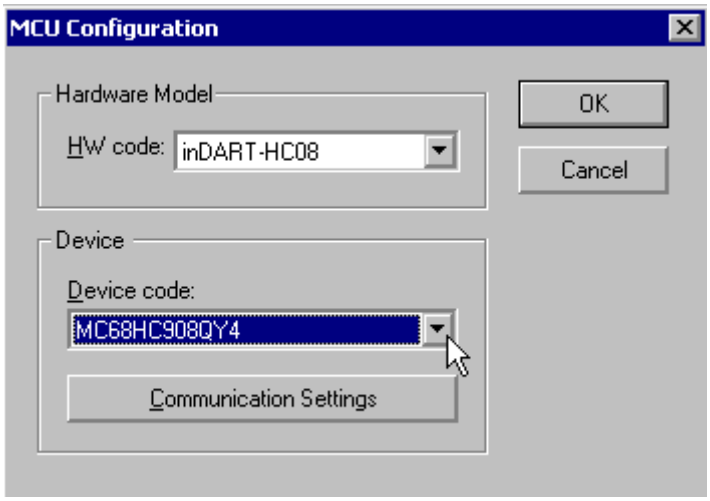
MCU Configuration dialog

The Hardware Model drop down list can be expanded to select another type of debug interface than the SofTec inDART-HC08. Note that at this document release time, only the SofTec inDART-HC08 is available.

The *HW Code* drop down list can be expanded to select another HCS08 derivative.

Pressing the *Communication Settings* button opens the Communication Settings dialog.

Figure 6.10 MCU Configuration dialog

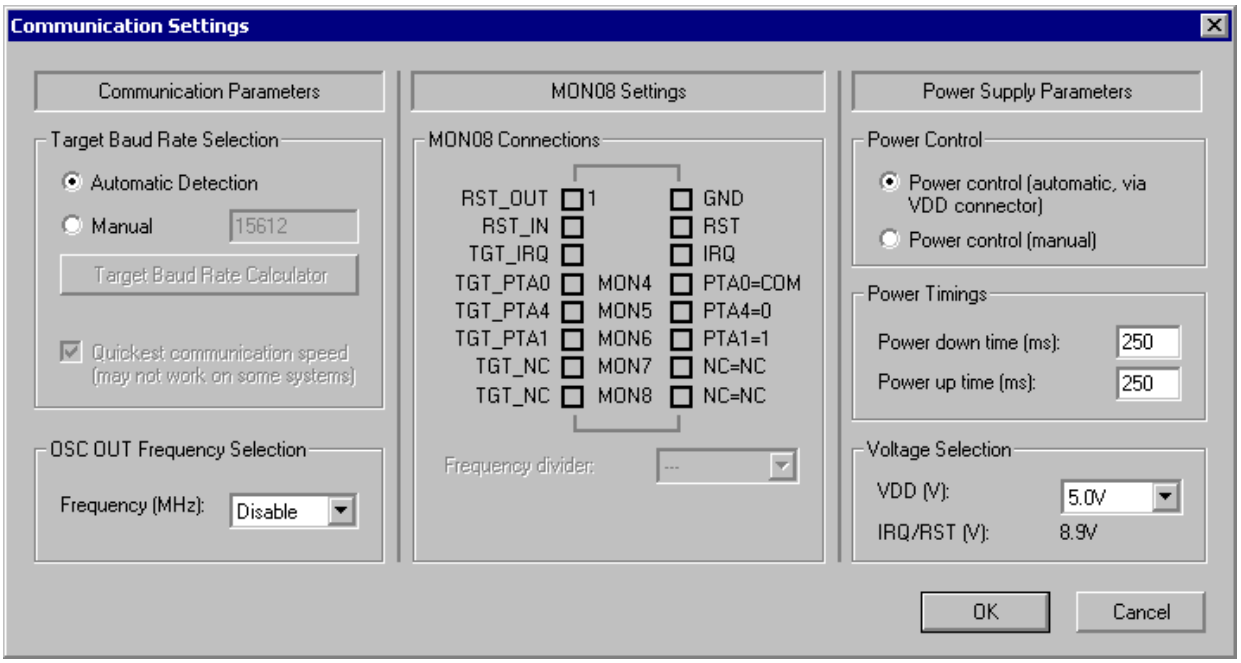


Communication Settings dialog

The "*Communication Settings*" button in the MCU Configuration dialog box allows you to fine-tune critical parameters needed for proper operation with the chosen target microcontroller. The dialog box is divided into three sections: "*Communication Parameters*", "*MON08 Settings*" and "*Power Supply Parameters*". All of the parameters must be carefully set, otherwise unsuccessful operations will result.

Please refer to ***"inDART®-HC08 In-Circuit Debugger/Programmer for Motorola HC08 Family FLASH Devices User's Manual"*** from SofTec for further details.

Figure 6.11 Communication Settings dialog

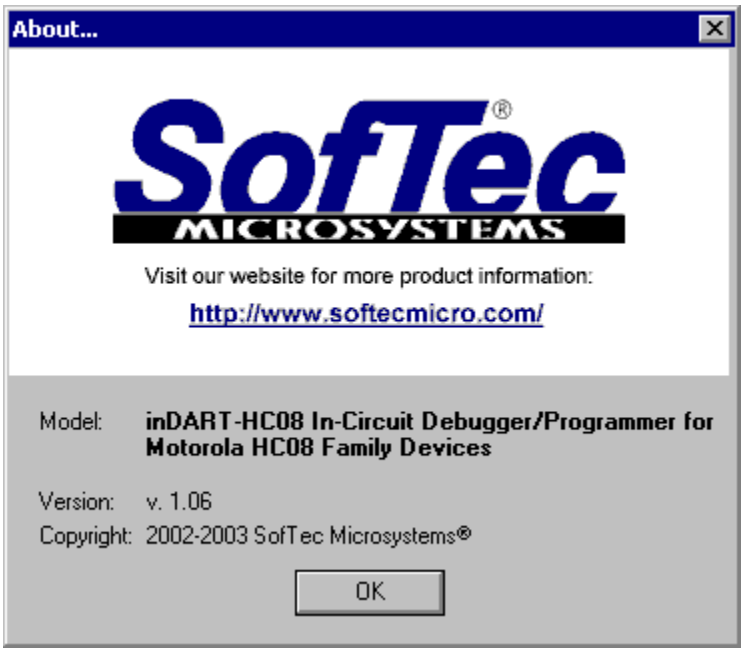


NOTE If your hardware supports stopping the application while running, an additional interrupt service routine is required for the IRQ vector. Please see ***Stop Command Handling*** section in ***"inDART®-HC08 In-Circuit Debugger/Programmer for Motorola HC08 Family FLASH Devices User's Manual"*** from SofTec for further details.

About dialog

This dialog belongs to the SofTec GDI dll and provides information about the ***inDART_HC08.dll*** release and version.

Figure 6.12 About dialog



7

Target Interface Commands Files

Target Interface Associated Command Files

The Target Interfaces offer the possibility to play a specific command file 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.
- right before a "Non Volatile Memory" is erased or right before a file is programmed in "Non Volatile Memory": Vppon Command File. This command file can be used for example to enable a programming voltage by software. This command file is not available for all Target Interfaces.
- right after a "Non Volatile Memory" has been erased or right after a file has been programmed in "Non Volatile Memory": Vppoff Command File. This command file can be used for example to disable a programming voltage by software. This command file is not available for all Target Interfaces.

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 *Metrowerks debugger* command in those files and take advantage of the wide set of commands introduced in the *Metrowerks debugger* manual to setup the target hardware on one of those events.

Listing 7.1 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.

Startup Command File

The Startup command file is executed by the *Metrowerks 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.

Reset Command File

The Reset command file is executed by the *Metrowerks 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 the Command Files dialog.

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

Preload Command File

The Preload command file is executed by the *Metrowerks 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.

Postload Command File

The Postload command file is executed by the *Metrowerks 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.

Vppon Command File

The Vppon command file is executed by the *Metrowerks debugger* right before a "Non Volatile Memory" is erased or right before a file is programmed in "Non Volatile Memory" to the target system through the Target Interface Non Volatile Memory Control dialog (Flash... menu entry) or `FLASH PROGRAM/ERASE` commands from Flash Programming utilities.

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

By default the `VPPON.CMD` file located in the current project directory is enabled as the current Vppon command file.

This command file can be used for example to enable a programming voltage by software.

WARNING! This command file is not available for all Target Interfaces.

Vppoff Command File

The Vppoff command file is executed by the *Metrowerks debugger* right after a "Non Volatile Memory" has been erased or right after a file has been programmed in "Non Volatile Memory" to the target system through the Target Interface Non Volatile Memory Control dialog (Flash... menu entry) or `FLASH PROGRAM/ERASE` commands from Flash Programming utilities.

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

By default the `VPPOFF.CMD` file located in the current project directory is enabled as the current Vppoff command file.

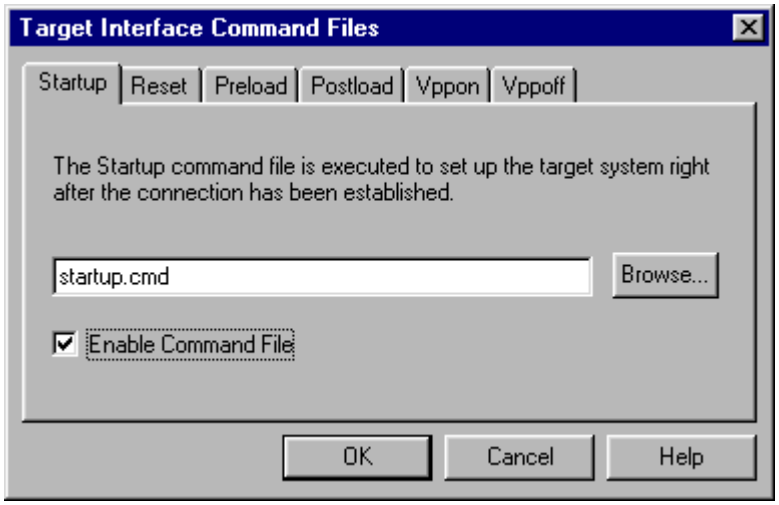
This command file can be used for example to disable a programming voltage by software.

WARNING! This command file is not available for all Target Interfaces.

Command Files dialog

The Target Interface Command Files dialog can be opened selecting menu entry **"TargetName" > Command Files**. (In this section, **TargetName** is the name of the target, like *SDI*, *Hitex*, *BDIK*, *ICD-12*, *Noral-BDM*, etc.)

Figure 7.1 Target Interface Command Files dialog



Each index of this dialog corresponds to an event on which Target Interface Commands Files can be automatically run from the *Metrowerks debugger*: Startup Command File, Reset Command File, Preload Command File, Vppon Command File, (not available for all targets), Vppoff Command File (not available for all targets), or any other Target Interface specific 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.
- the default Vppon command file is VPPON.CMD.
- the default Vppoff command file is VPPOFF.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 command which can be used when the Target Interface is set.

The Target Interface specific commands are:

CMDFILE

Those commands can be entered in the command files or in the Command Line component of the *Metrowerks debugger*.

This section describes each command available for the Target Interface. The commands are listed in alphabetical order.

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|VPPON|VPPOFF
```

Description

The CMDFILE command is to be used set up a command file 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 ["targetName"] section of the PROJECT file using variable CMDFILEn.

Example

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

```
in>CMDFILE
Hitex 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
Hitex 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 `["targetName"]` section from the project file.

Listing 7.2 Example of the `[NORAL FLEX BDM]` target section from the project file:

```
[NORAL FLEX BDM]
CMDFILE0=CMDFILE STARTUP ON "startup.cmd"
CMDFILE1=CMDFILE RESET ON "reset.cmd"
CMDFILE2=CMDFILE PRELOAD ON "preload.cmd"
CMDFILE3=CMDFILE POSTLOAD ON "postload.cmd"
CMDFILE4=CMDFILE VPPON ON "vppon.cmd"
CMDFILE5=CMDFILE VPPOFF ON "vppoff.cmd"
```

The following section describes each variable available for the Target Interface. The variables are listed in alphabetical order.

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.

The default Vppon command file is VPPON.CMD.

The default Vppoff command file is VPPOFF.CMD.

Description

The CMDFILEn variable specifies a command file definition using CMDFILE Command Line command.If there are four Target Interface Commands Files for the Target Interface, four of those 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.

Example

```
CMDFILE0=CMDFILE STARTUP ON "startup.cmd"
CMDFILE1=CMDFILE RESET ON "reset.cmd"
CMDFILE2=CMDFILE PRELOAD ON "preload.cmd"
CMDFILE3=CMDFILE POSTLOAD ON "postload.cmd"
CMDFILE4=CMDFILE VPPON OFF "vppon.cmd"
CMDFILE5=CMDFILE VPPOFF OFF "vppoff.cmd"
```

Index

S

T

U

V

W