

inDART[®] -HCS12
In-Circuit
Debugger/Programmer
for Motorola HCS12 Family
FLASH Devices

User's Manual



Copyright © 2003 SofTec Microsystems®

DC00624

We want your feedback!

SofTec Microsystems is always on the look-out for new ways to improve its Products and Services. For this reason feedback, comments, suggestions or criticisms, however small, are always welcome.

SofTec Microsystems

E-mail (general information): info@softecmicro.com

E-mail (marketing department): marketing@softecmicro.com

E-mail (technical support): support@softecmicro.com

Web: <http://www.softecmicro.com>

Important

SofTec Microsystems reserves the right to make improvements to the inDART® Series In-Circuit Debuggers, their documentation and software routines, without notice. Information in this manual is intended to be accurate and reliable. However, SofTec Microsystems assumes no responsibility for its use; nor for any infringements of rights of third parties which may result from its use.

SOFTEC MICROSYSTEMS WILL NOT BE LIABLE FOR DAMAGES RESULTING FROM LOSS OF DATA, PROFITS, USE OF PRODUCTS, OR INCIDENTAL OR CONSEQUENTIAL DAMAGES, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.

Trademarks

inDART is a trademark of SofTec Microsystems.

Motorola and DigitalDNA are trademarks or registered trademarks of Motorola, Inc.

Metrowerks and CodeWarrior are trademarks or registered trademarks of Metrowerks Corp.

Microsoft and Windows are trademarks or registered trademarks of Microsoft Corporation.

PC is a registered trademark of International Business Machines Corporation.

Other products and company names listed are trademarks or trade names of their respective companies.

Written by Paolo Xausa

Contents

1. Overview	5
What is inDART-HCS12?	5
Background Debug Module (BDM)	6
CodeWarrior Integrated Development Environment	7
SofTec Microsystems DataBlaze Programming Utility	8
Demo Boards	8
Recommended Reading	8
Software Upgrades	9
2. Getting Started	11
inDART-HCS12 Components	11
Host System Requirements	12
Installing the Software	13
<i>Installing Metrowerks CodeWarrior IDE</i>	13
<i>Installing SofTec Microsystems Additional Components</i>	13
Installing the Hardware	14
Application Tutorial	16
3. inDART-HCS12 Operations	21
inDART-HCS12 Working Principles	21
Configuring a Debugging Session	21
<i>Creating Your Own Application</i>	21
<i>MCU Configuration</i>	22
<i>MCU Configuration > Communication Settings</i>	22
Breakpoints and Trace	24
Using Existing Projects with inDART-HCS12	25
Notes and Tips	26
<i>Entering Debug Session with CodeWarrior</i>	26
<i>Reading Peripheral Status</i>	26
<i>Breakpoints and BGND Instruction</i>	26
<i>Real-Time Memory Update</i>	27
DataBlaze Programming Utility	27
<i>DataBlaze Notes</i>	28

4. Troubleshooting	31
Common Problems and Solutions	31
<i>Communication Can't Be Established with inDART-HCS12</i>	31
<i>Stepping Execution is Slow</i>	32
Diagnostic Test	32
Getting Technical Support	32
Appendix A. Electrical and Physical Specifications	35

1. Overview

What is inDART-HCS12?

inDART-HCS12 is a powerful entry-level tool for Motorola HCS12-based systems. inDART-HCS12 takes advantage of Metrowerks CodeWarrior HC(S)12 Integrated Development Environment and the BDM (Background Debug Module) feature to debug the user program. Together with CodeWarrior HC(S)12, inDART-HCS12 provides you with everything you need to download (program), in-circuit emulate and debug user code. Full-speed program execution allows you to perform hardware and software testing in real time. inDART-HCS12 is connected to the host PC through a USB port, while the 6-pin BDM connector of the product fits into the target's standard BDM connector. Design Kit packages also include a full-featured experiment board for a specific HCS12 microcontroller.

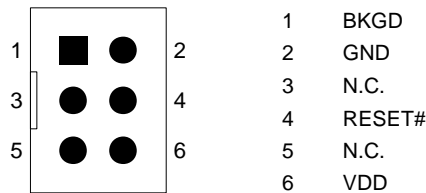
inDART-HCS12 offers you the following benefits:

- Real-time code execution without probes—works with all packages (BDM-compatible connector);
- In-circuit debugging;
- Built-in FLASH programmer (DataBlaze programming utility);
- 1.8 to 5.5 V devices supported;
- Standard chip used—no bondouts, 100% electrical characteristics guaranteed;
- Working frequency up to the microcontroller's maximum;
- Jumperless hardware mode setting;
- Automatic target frequency detection;
- Hardware self diagnostic test;
- USB connection to the PC;
- Metrowerks CodeWarrior IDE (the same user interface of all Motorola tools), with editor, assembler, C compiler and debugger.

Background Debug Module (BDM)

All MCUs in the HCS12 family contain a single-wire background debug interface which supports in-circuit programming of on-chip non-volatile memory and sophisticated non-intrusive debug capabilities. This system does not interfere with normal application resources. It does not use any user memory or locations in the memory map and does not share any on-chip peripherals. The background debug module (BDM) uses a single-wire communication interface to allow non-intrusive access to target system memory and registers.

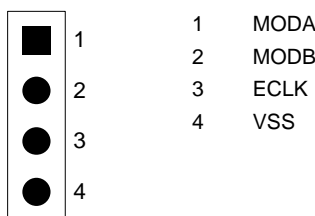
inDART-HCS12 uses the standard, 6-pin BDM connector defined by Motorola to program and debug the target device. You must therefore provide such connector (see the diagram below) on your target board.



BDM Connector

- **BKGD:** Single-wire background interface pin. This signal is required.
- **GND:** System ground. This signal is required.
- **RESET#:** Reset signal to target system. This signal is required.
- **VDD:** Power supply voltage from target. This signal is required by inDART-HCS12 signal conditioning.

inDART-HCS12 also has an additional connector (the “Enhanced BDM” connector) which features additional signals (MODA, MODB, ECLK, VSS) to extend the standard debugging capabilities.



Enhanced BDM Connector

- **MODA, MODB:** Used to automatically configure the target device mode after reset, in case the target device doesn't already power-up in the operating mode required for BDM debugging.
- **ECLK:** Used to synchronize the BDM communication through the BKGD line. If the target's clock frequency changes, the BDM communication speed must change accordingly. The ECLK clock signal provides a safe way to always keep the BDM communication working under any circumstances. This signal is particularly useful with those devices which have a BDM peripheral that doesn't support the ACK and SYNC commands.
- **VSS:** System ground.

CodeWarrior Integrated Development Environment

inDART-HCS12 comes with a free version of CodeWarrior Development Studio for HC(S)12 Microcontrollers, Special Edition.

CodeWarrior Development Studio for HC(S)12 is a powerful and easy-to-use tool suite designed to increase your software development productivity. Its Integrated Development Environment (IDE) provides unrivaled features such as Processor Expert application design tool, full chip simulation, Data Visualization and project manager with templates to help you concentrate on the added value of your application.

The comprehensive, highly visual CodeWarrior Development Studio for Motorola HC(S)12 Microcontrollers enables you to build and deploy HC(S)12 systems quickly and easily. This tool suite provides the capabilities required by every engineer in the development cycle, from board bring-up to firmware development to final application development.

To use the Special Edition, you must have a valid license key. Without the license key the product will run in a 1 KB code-size limited demonstration mode.

To request the license key, please refer to Metrowerks website.

This documentation covers the basic setup and operation of the CodeWarrior IDE, but does not cover all of its functions. For further information, please refer to the CodeWarrior on-line help and online documentation provided.

SofTec Microsystems DataBlaze Programming Utility

The SofTec Microsystems “**System Software**” CD-ROM contains the DataBlaze programming utility targeted to the inDART-HCS12 board.

DataBlaze is a full-featured programming utility which offers a complete set of programming features like memory editing, blank check/erase/verify operations and read/write operations.

Demo Boards

Design Kit packages include a full-featured, microcontroller-specific experiment board. The demo board can be used for evaluation/experiments in the absence of a target application board.

Recommended Reading

This documentation describes how to use inDART-HCS12 together with Metrowerks CodeWarrior HC(S)12 IDE and DataBlaze programming environment. Additional information can be found in the following documents:

- **Evaluation Board-Specific Information**—If you bought a Design Kit, additional documentation which explains the evaluation board’s details is available under **Start > Programs > SofTec Microsystems > inDART-HCS12 > User’s Manuals**.
- **Metrowerks’ Additional Documentation**— Available from the CodeWarrior IDE.
- **Motorola HCS12 Datasheets**—Include detailed information on the devices’ background debug module.

Software Upgrades

The latest version of the inDART-HCS12 system software is always available free of charge from our website: <http://www.softecmicro.com>. Metrowerks CodeWarrior upgrades can be found at <http://www.metrowerks.com>.

2. Getting Started

2

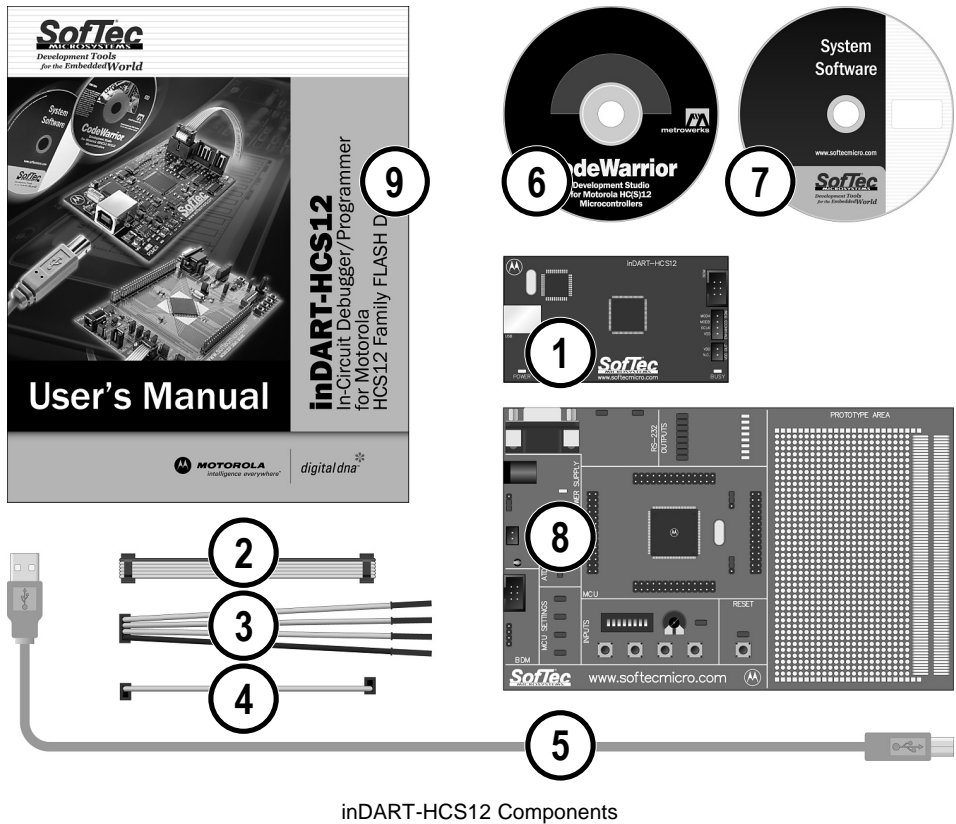
inDART-HCS12 Components

The inDART-HCS12 package includes the following items:

1. The inDART-HCS12 in-circuit debugger/programmer unit;
2. A 20-cm, 6-conductor BDM cable;
3. A 30-cm, 4-conductor Enhanced BDM cable;
4. A 30-cm, 1-conductor “VDD OUT” cable;
5. A USB cable;
6. The Metrowerks CodeWarrior HC(S)12 CD-ROM;
7. The SofTec Microsystems inDART-HCS12 “System Software” CD-ROM;
8. A full-featured, microcontroller-specific experiment board (Design Kit packages only);
9. This user’s manual.

2. Getting Started

2



Host System Requirements

The inDART-HCS12 in-circuit debugger is controlled by an Integrated Development Environment running under Windows (CodeWarrior HC(S)12). The following hardware and software are required to run the CodeWarrior HC(S)12 user interface together with inDART-HCS12:

1. A 133-MHz (or higher) PC compatible system running Windows 98, Windows 2000 or Windows XP;
2. 128 MB of available system RAM plus 500 MB of available hard disk space;

3. A USB port;
4. CD-ROM drive for installation.

Installing the Software

inDART-HCS12 requires that both Metrowerks CodeWarrior IDE and SofTec Microsystems inDART-HCS12 additional components be installed in the host PC.

Note: *Metrowerks CodeWarrior HC(S)12 IDE must be installed first. Please note that inDART-HCS12 only works with CodeWarrior for HC(S)12 version 3.0 or above.*

Installing Metrowerks CodeWarrior IDE

To install the CodeWarrior IDE insert the CodeWarrior CD-ROM into your computer's CD-ROM drive. A startup window will automatically appear. Follow the on-screen instructions.

Installing SofTec Microsystems Additional Components

The SofTec Microsystems additional components install all of the other required components to your hard drive. These components include:

- The inDART-HCS12 USB driver;
- inDART-HCS12 software plug-in for CodeWarrior HC(S)12;
- Sample source code for SofTec Microsystems evaluation boards;
- SofTec Microsystems DataBlaze programming utility;
- inDART -HCS12 hardware diagnostic test utility;
- Documentation in PDF format.

To install the SofTec Microsystems additional components insert the SofTec Microsystems “**System Software**” CD-ROM into your computer's CD-ROM drive. A startup window will automatically appear. Choose “**Install**

Instrument Software” from the main menu. A list of available software will appear. Click on the **“inDART-HCS12 Additional Components”** option. Follow the on-screen instructions.

***Note:** if you are installing the inDART-HCS12 additional components on Windows 2000 or Windows XP you must have logged in as Administrator.*

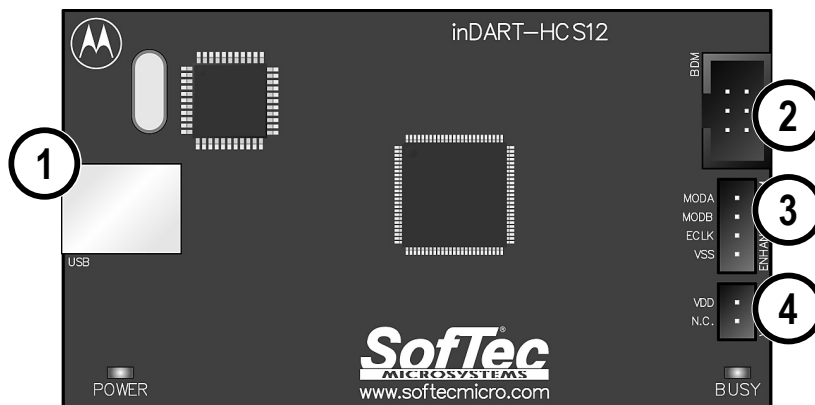
Installing the Hardware

The inDART-HCS12 in-circuit debugger is connected through a USB port to a host PC. Connection steps are listed below in the recommended flow order:

1. Install all the required system software as described in the previous section.
2. Insert one end of the USB cable into a free USB port.
3. Insert the other end of the USB cable into the “USB” connector on the inDART-HCS12 board. The green “POWER” LED on the instrument should turn on. Windows will automatically recognize the instrument and will load the appropriate USB driver.
4. Insert one end of the BDM cable into the 6-way BDM connector on the inDART-HCS12 board.
5. Insert the other end of the BDM cable into the BDM connector of the demo board or target application.
6. Power the demo board or target application. When using a SofTec Microsystems demo board, there are two power options:
 - a. **Power the board via the “UNREG VDD” connector:** The “UNREG VDD” connector accepts 9-12 V DC, 500 mA wall plug-in power supply with a 2.1 mm pin and sleeve plug with positive in the center and sleeve as ground. When powering the board through this connector, make sure the “VDD SOURCE” jumper selects the “UNREG” position. The “UNREG VDD” voltage is internally regulated to the microcontroller’s operating voltage (5.0 V DC in the case of the MC9S12DP256).

- b. **Power the board via the “REG VDD” connector:** The “REG VDD” connector accepts a voltage up to the microcontroller’s operating voltage (5.0 V DC in the case of the MC9S12DP256). When powering the board through this connector, make sure the “VDD SOURCE” jumper selects the “REG” position. The “REG VDD” voltage directly powers the microcontroller and the rest of the board. The “REG VDD” connector has been designed to be used together with inDART-HCS12’s “VDD OUT” connector: the “VDD OUT” cable can be conveniently inserted into the “VDD OUT” connector on the inDART-HCS12 board and into the “REG VDD” connector on the demo board. This way inDART-HCS12 powers the demo board.

Please refer to the following figure to get an overview of the inDART-HCS12 unit.



1. USB Connector
2. BDM Connector
3. Enhanced BDM Connector
4. "VDD OUT" Connector

The inDART-HCS12 Unit

Note: *both Windows 2000 and Windows XP may issue a warning the first time inDART-HCS12 is connected to the PC. This warning is related to the fact that the USB driver used by inDART-HCS12 is not digitally signed by Microsoft, and Windows considers it to be potentially malfunctioning or dangerous for the system. However, you can safely ignore the warning, since every kind of compatibility/security test has been carried out by SofTec Microsystems. Additionally, under Windows XP, the “Found New Hardware Wizard” procedure may occur twice.*

Application Tutorial

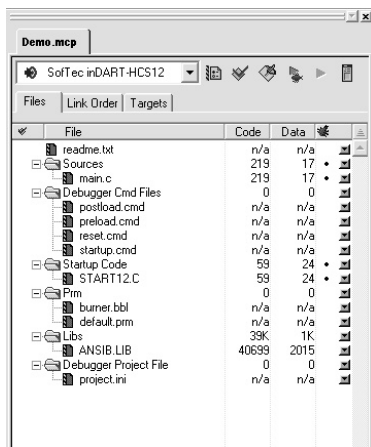
This section will provide a step-by-step guide on how to launch your first inDART-HCS12 project and get started with the CodeWarrior HC(S)12 user interface.

Note: *the example provided assumes that inDART-HCS12 is used with the IDB-HCS12DP demo board. If you have a different demo board, some procedures may differ slightly. This tutorial is based on a C example. For evaluation board-specific features, please refer to the evaluation boards’ PDF manuals installed by SofTec Microsystems additional components.*

The sample application does the following:

- By rotating the potentiometer, you affect the results of the A/D conversion, and the value of each conversion is displayed on the LEDs;
- By pressing the PPO push button, the DIP-switches status is displayed on the LEDs;
- By pressing the PP1 push button, the string “PP1 Push-Button Pressed” is sent over the RS-232 (9600, N, 8, 1);
- By pressing the PP2 push button, the string “PP2 Push-Button Pressed” is sent over the RS-232 (9600, N, 8, 1);

- By pressing the PP3 push button, the string “PP3 Push-Button Pressed” is sent over the RS-232 (9600, N, 8, 1).
1. Ensure that the inDART-HCS12 board is connected to the PC (via the USB cable) and that the demo board is powered on via the “REG. VDD” connector (the “VDD OUT” cable must be connected and the “VDD SOURCE” jumper must select the “REG.” option).
 2. Ensure that the inDART-HCS12 board is connected to the demo board via the BDM connector.
 3. Make sure that all of the evaluation board’s jumpers are set to their factory position.
 4. Start the CodeWarrior HC(S)12 IDE by selecting **Start > Programs > Metrowerks CodeWarrior > CW12 > CodeWarrior IDE**. The CodeWarrior HC(S)12 IDE will open.
 5. From the main menu, choose **File > Open**. Select the “**Demo.mcp**” workspace file that is located under the “**\Program Files\Metrowerks\CodeWarrior CW12\CodeWarrior_Examples\MC9S12\SofTec Microsystems\inDART-HCS12\IDB-HCS12DP\C\Demo**” directory. Click “**Open**”. The following window will appear.

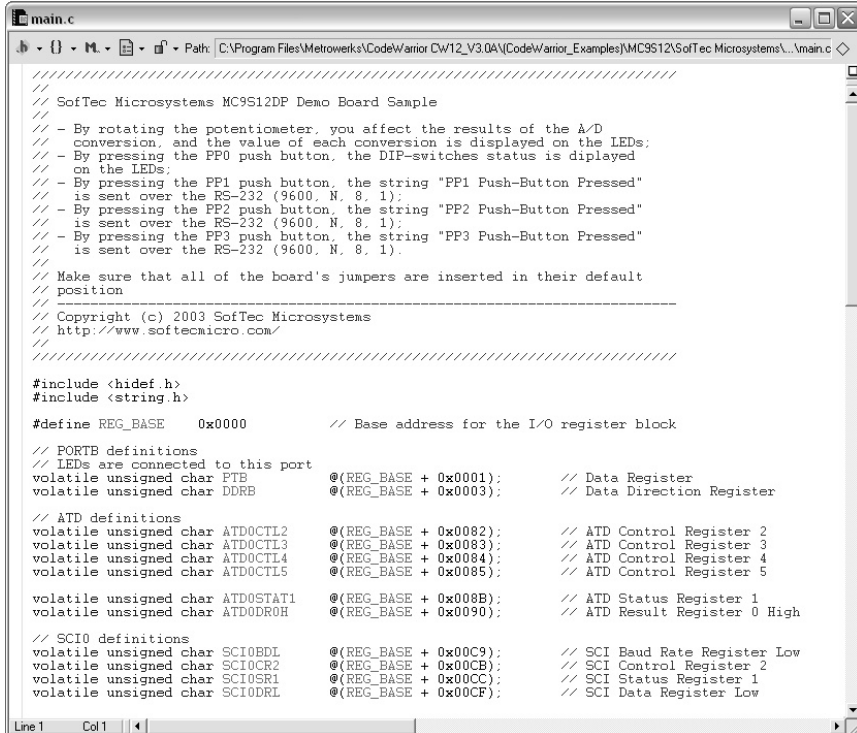


The Project Window

2. Getting Started

6. The C code of this example is contained in the “**main.c**” file. Double click on it to open it. The following window will appear.

2



```
main.c
C:\Program Files\Metrowerks\CodeWarrior CW12_V3.0A\CodeWarrior_Examples\MC9S12\SofTec Microsystems\...\main.c

////////////////////////////////////
// SofTec Microsystems MC9S12DP Demo Board Sample
//
// - By rotating the potentiometer, you affect the results of the A/D
// conversion, and the value of each conversion is displayed on the LEDs;
// - By pressing the PP0 push button, the DIP-switches status is displayed
// on the LEDs;
// - By pressing the PP1 push button, the string "PP1 Push-Button Pressed"
// is sent over the RS-232 (9600, N, 8, 1);
// - By pressing the PP2 push button, the string "PP2 Push-Button Pressed"
// is sent over the RS-232 (9600, N, 8, 1);
// - By pressing the PP3 push button, the string "PP3 Push-Button Pressed"
// is sent over the RS-232 (9600, N, 8, 1).
//
// Make sure that all of the board's jumpers are inserted in their default
// position
//-----
// Copyright (c) 2003 SofTec Microsystems
// http://www.softecmicro.com/
//-----
////////////////////////////////////

#include <hidef.h>
#include <string.h>

#define REG_BASE    0x0000        // Base address for the I/O register block

// PORTB definitions
// LEDs are connected to this port
volatile unsigned char PTB        @(REG_BASE + 0x0001);    // Data Register
volatile unsigned char DDRB       @(REG_BASE + 0x0003);    // Data Direction Register

// ATD definitions
volatile unsigned char ATD0CTL2   @(REG_BASE + 0x0082);    // ATD Control Register 2
volatile unsigned char ATD0CTL3   @(REG_BASE + 0x0083);    // ATD Control Register 3
volatile unsigned char ATD0CTL4   @(REG_BASE + 0x0084);    // ATD Control Register 4
volatile unsigned char ATD0CTL5   @(REG_BASE + 0x0085);    // ATD Control Register 5

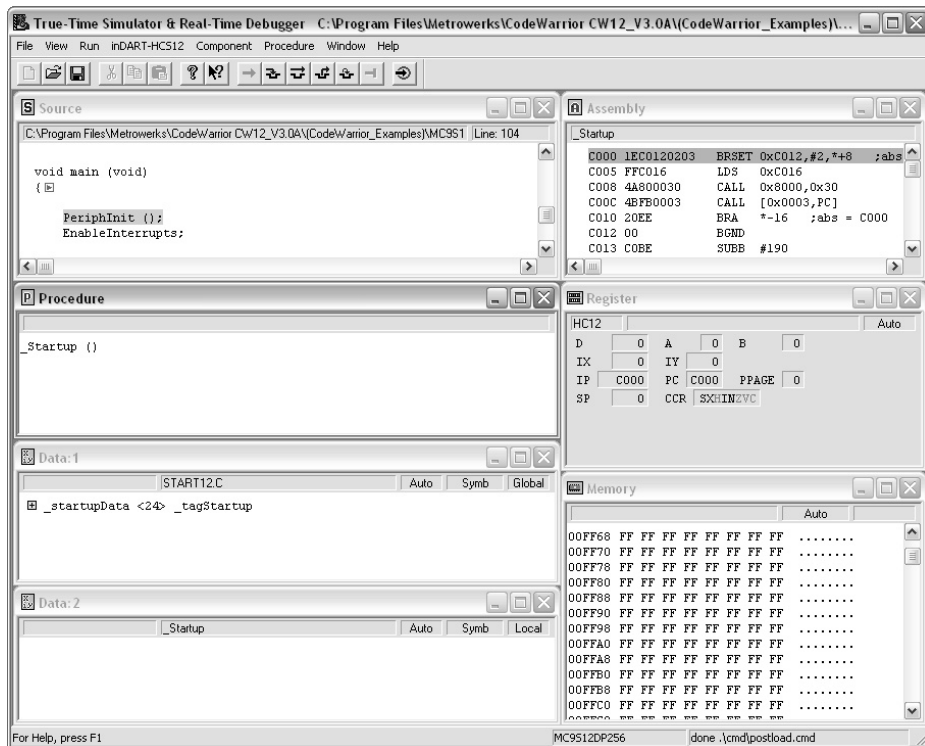
volatile unsigned char ATD0STAT1  @(REG_BASE + 0x008B);    // ATD Status Register 1
volatile unsigned char ATD0DR0H   @(REG_BASE + 0x0090);    // ATD Result Register 0 High

// SCI0 definitions
volatile unsigned char SCI0BDL    @(REG_BASE + 0x00C9);    // SCI Baud Rate Register Low
volatile unsigned char SCI0CR2    @(REG_BASE + 0x00CB);    // SCI Control Register 2
volatile unsigned char SCI0SR1    @(REG_BASE + 0x00CC);    // SCI Status Register 1
volatile unsigned char SCI0DRL    @(REG_BASE + 0x00CF);    // SCI Data Register Low

Line1 Col1
```

The Example's Source Code

7. From the main menu, choose **Project > Debug**. This will compile the source code, will generate an executable file and will download it to the demo board.
8. A new debugger environment will open.



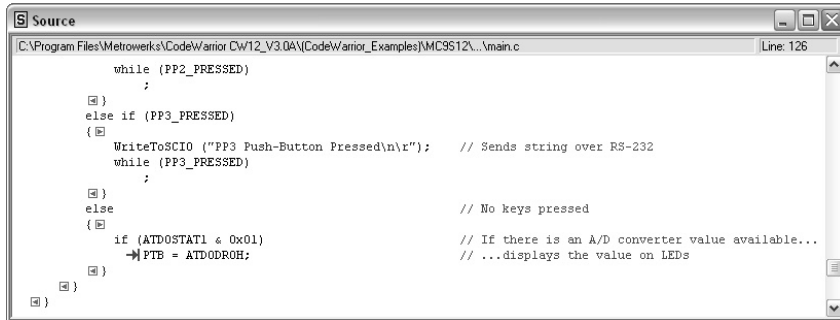
Debugging Session Started

9. From the main menu, choose **Run > Start/Continue**. The program will be executed in real-time. Please note that the “BUSY” LED on the inDART-HCS12 unit turns on. By rotating the potentiometer on the demo board, you affect the results of the A/D conversion, and the value of each conversion is displayed on the LEDs.
10. From the main menu, choose **Run > Halt**. The program execution will stop. The next instruction to be executed is highlighted in the *Source* window.
11. From the main menu, choose **Run > Single Step**. The instruction highlighted in the *Source* window will be executed, and the program execution will be stopped immediately after.

2. Getting Started

2

12. In the *Source* window, insert a breakpoint at the “**PTB = ATDoDRoH;**” instruction in the **main** function. To insert the breakpoint, right-click on the “**PTB = ATDoDRoH**” line and, from the pop-up menu, select “**Set Breakpoint**”.



Breakpoint Set

13. Rotate the potentiometer slightly. Then, from the main menu, choose **Run > Start/Continue**. The application will restart from where it was previously stopped. The application will stop at the breakpoint location as soon as the next A/D conversion is done.
14. Issue a Single Step command (**Run > Single Step**). The value of the A/D conversion will be displayed on the LEDs.

Congratulations! You have successfully completed this tutorial! You can continue to experiment with the CodeWarrior user interface and discover by yourself its potentialities. For an in-depth guide of all of the user interface features, select **Help > CodeWarrior Help** or **Help > Online Manuals** from the CodeWarrior HC(S)12 IDE's main menu.

3. inDART-HCS12 Operations

inDART-HCS12 Working Principles

inDART-HCS12 is an in-circuit debugger as well as a programming tool. It programs files into the HCS12 microcontrollers and offers debugging features like real-time code execution, stepping, and breakpoint. Its debugging features are achieved thanks to the microcontroller's integrated Background Debug Module (BDM).

The BDM peripheral communicates with the inDART-HCS12 board through a single-wire dedicated line of the microcontroller. The same line is also used during device programming.

Contrariwise to traditional in-circuit emulation (where the target application is executed and emulated inside the emulator), inDART-HCS12 uses the very same target microcontroller to carry on in-circuit execution. This means that all microcontroller's peripherals (timers, A/D converters, I/O pins, etc.) are not reconstructed or simulated by an external device, but are the very same target microcontroller's peripherals. Moreover, the inDART-HCS12 debugging approach ensures that the target microcontroller's electrical characteristics (pull-ups, low-voltage operations, I/O thresholds, etc.) are 100% guaranteed.

3

Configuring a Debugging Session

Creating Your Own Application

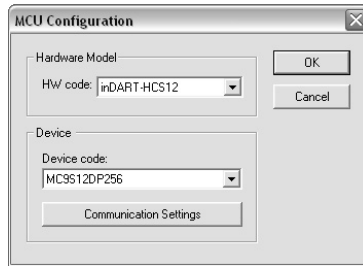
CodeWarrior HC(S)12 helps you get started with your own application by including a project wizard specific for the inDART-HCS12 board. To create a new inDART-HCS12 project:

1. From the main menu, select **File > New**.
2. A dialog box will appear. Select “**HC(S)12 New Project Wizard**”.
3. Follow the Project Wizard steps, making sure you select the correct microcontroller derivative you are working with and the inDART-HCS12 board as emulator.

MCU Configuration

The first time you start the debugging of your application (with the command **Project > Debug**), the *MCU Configuration* dialog box will appear, allowing you to select the hardware model and the target microcontroller you are working with.

3

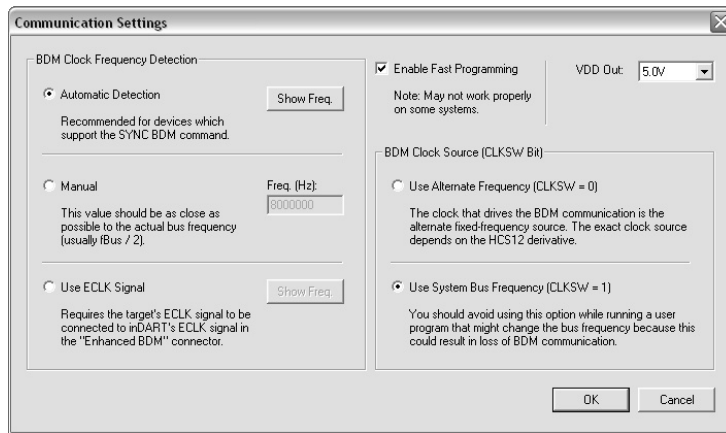


The *MCU Configuration* Dialog Box

First, ensure that the “**Hardware Model**” parameter is set to “**inDART-HCS12**”. Then, set the “**Device Code**” parameter to the specific target microcontroller you are working with. Additional communication parameters can be specified by clicking the “**Communication Settings**” button.

MCU Configuration > Communication Settings

The “**Communication Settings**” button in the *MCU Configuration* dialog box allows you to specify critical parameters needed for proper operation with the chosen target microcontroller.



The *Communication Settings* Dialog Box

The first parameter, “**BDM Clock Frequency Detection**”, specifies how to detect the target microcontroller’s BDM clock frequency. There are three options:

- “**Automatic Detection**”: inDART-HCS12 automatically detects the target microcontroller’s BDM frequency. This setting is highly recommended for devices which support the SYNC BDM command, since in such devices the frequency detection is totally accurate. For devices which do not support the SYNC command, the automatic frequency detection may not be accurate.
- “**Use ECLK signal**”: inDART-HCS12 uses the microcontroller’s ECLK signal to automatically determine the microcontroller’s BDM frequency. The ECLK clock signal provides a safe way to always keep the BDM communication working. However, the “ECLK” pin of the inDART-HCS12’s enhanced BDM connector must be connected to the target microcontroller’s ECLK pin (using the Enhanced BDM cable). Of course, the ECLK pin of the microcontroller must not be used as an I/O pin.
- “**Manual**”: if the two options above are not suitable for you, you can manually specify the microcontroller’s BDM frequency. Usually, this value is half that of the external oscillator frequency. Since this frequency is also used by the programming algorithms, it is important to carefully specify this parameter. Programming the device with an incorrect BDM frequency may

result in stress to the FLASH/EEPROM memory cells, thus reducing the lifetime of the memory.

Another parameter, “**BDM Clock Source**”, specifies the link between the microcontroller’s bus frequency and the BDM clock frequency during debugging. The BDM clock source can be selected by the CLKSW bit in the BDM Status register.

3

- If the CLKSW bit is set to 1, the BDM communication clock source is the microcontroller’s bus frequency.
- If the CLKSW bit is set to 0, the BDM communication clock source is a constant clock source, which can vary depending on the specific HCS12 derivative. In the case of the MC9S12DP256, for example, this constant clock source is half the frequency of the external oscillator.

Which CLKSW setting to use depends on the target system and on the user application. The idea is to set the CLKSW bit so that the highest (and less subject to changes) clock frequency is used for the BDM communication. A low clock frequency will result in slow BDM communication (and therefore slow debugging), while a clock frequency which changes frequently (as in the case of the user application modifying the PLL peripheral) may result in loss of BDM communication.

The “**Enable Fast Programming**” parameter, if selected, speeds up programming by driving the microcontroller’s internal PLL circuitry to the maximum settings. In order for this feature to work, an appropriate loop filter circuitry must be provided to the device’s XFC pin.

Finally, the “**VDD Out**” parameter specifies the voltage provided by inDART-HCS12 on the “VDD OUT” connector, which can be used to power up SofTec Microsystems’ demo boards.

Breakpoints and Trace

CodeWarrior offers a variety of tools for analyzing the program flow: breakpoints (both simple and complex), watchpoints and a trace buffer. All these features are implemented by taking advantage of the target microcontroller’s debug peripheral.

However, depending on the details of the debug peripheral embedded in the specific microcontroller you are working with, some debug features may or may not be available.

Note: *the number of available hardware breakpoints depends on the details of the debug peripheral embedded in the specific microcontroller you are working with, and on its settings.*

Note: *when setting an instruction breakpoint on a RAM location, a software breakpoint is set (the opcode present at that location is automatically replaced by the **BGND** Assembly instruction). Therefore, no hardware breakpoints are wasted.*

Note: *the Single Step command (in a C source code) and the Step Over and Step Out commands (both in a C and Assembly source code) use one hardware breakpoint.*

3

Using Existing Projects with inDART-HCS12

If your project has been targeted to an emulator/simulator other than inDART-HCS12 and you wish to use inDART-HCS12 as the debugger for your project, please do the following:

1. CodeWarrior is interfaced to the inDART-HCS12 engine through a so-called “GDI interface”. From the CodeWarrior HC(S)12 debugger interface, select **Component > Set Target** and choose “**HC12**” as processor and “**GDI Target Interface**” as target interface.
2. A dialog box will appear asking you to locate the GDI dll file needed to interface with inDART-HCS12. Select the **SofTec_BDM12.dll** file located

into the `\Program Files\Metrowerks\CodeWarrior CW12\prog\` directory.

3. CodeWarrior will then recognize inDART-HCS12 as the target interface for your project. The *MCU Configuration* dialog box will appear allowing you to select the derivative you are working with and the appropriate communication settings.
4. On the CodeWarrior HC(S)12 debugger interface a new menu (**inDART-HCS12**) will be created. From this menu, select **Load** and locate the object file your project is based on.

3

Notes and Tips

Entering Debug Session with CodeWarrior

When entering a debug session, the target microcontroller's FLASH memory is automatically erased, unsecured and programmed with the user application.

Note: *When programming the microcontroller with the user application (after having unsecured the device), CodeWarrior ignores (doesn't program) the security bits. As a result, when entering a debug session, the device is always unsecured, regardless of other user settings.*

Reading Peripheral Status

Care must be taken when reading some peripheral's status/data registers, since a reading operation may cause the clearing of flags. This may happen when the *Memory* window or the *Data* window is open, since these windows read microcontroller's resources during refresh operations.

Breakpoints and BGND Instruction

The BGND Assembly instruction forces the target microcontroller to enter the Active Background Debug mode, stopping program execution. CodeWarrior recognizes this event as a breakpoint and updates the contents of registers, memory, etc. Successive commands (Start/Continue, Single Step, etc.) will continue the execution of the program from the next instruction.

Real-Time Memory Update

During program execution, it is possible to view/edit the contents of the *Memory* window and *Data* window in real time (edit operations are only available for RAM locations). For example, it is possible to set the periodical refresh of the *Memory* window contents by choosing **Mode > Periodical** from the pop-up menu which appears by right-clicking on the *Memory* window.

3

DataBlaze Programming Utility

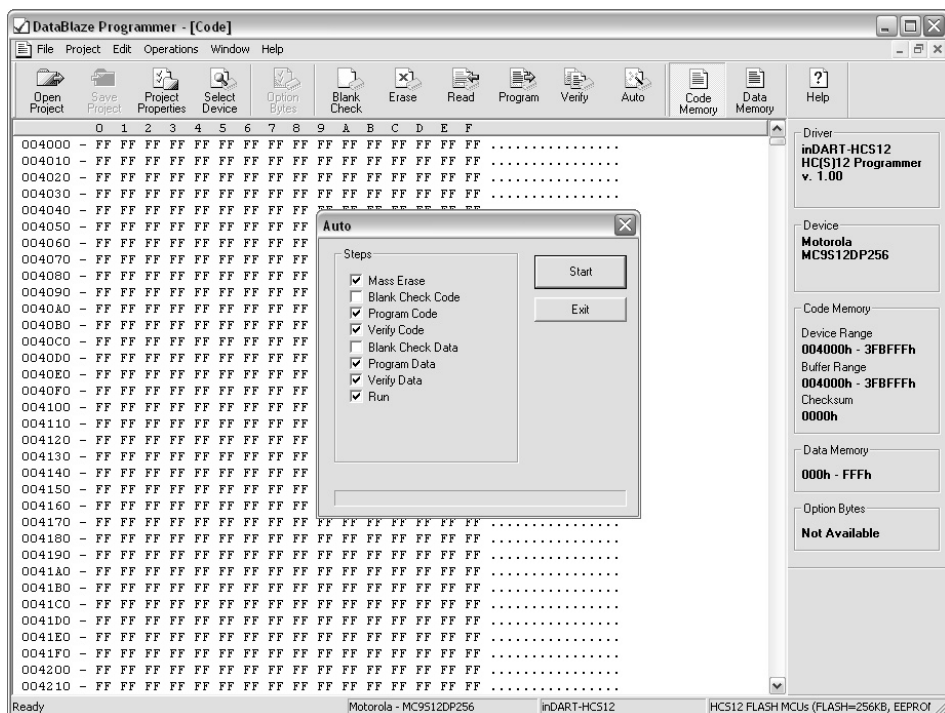
A full-featured programming utility (DataBlaze) is also provided with inDART-HCS12. To start the DataBlaze utility select **Start > Programs > SofTec Microsystems > inDART-HCS12 > DataBlaze Programmer**.

DataBlaze offers the following advanced features:

- Code memory editing;
- Data memory (EEPROM) editing;
- Blank check/erase/program/verify operations on Code memory and Data memory;
- Read operations from Code memory and Data memory;
- Project handling;
- One-button, multiple-operations programming (“Auto” feature);
- Serial numbering.

3. inDART-HCS12 Operations

3



The DataBlaze User Interface

DataBlaze Notes

- The “Mass Erase” operation always blanks the device (even if the device is protected or secured) and “unsecures” the device (the FLASH Options/Security Byte location is programmed with 0xFF).
- The “Blank Check” operation doesn’t blank check the FLASH Options/Security Byte location.
- The “Program” operation automatically verifies the programmed data, by reading back the programmed data and checking it against the buffer sent to the target device. The “Verify” operation is much more secure (but slower), since it reads back the programmed data and checks it against the data buffer present in the host PC.
- In case of verifying error, please verify that the FLASH Options/Security Byte location is programmed with a value different from 0xFF.

- The “Read”, “Program” and “Verify” operations are performed (when possible) by setting the target microcontroller’s PLL peripheral so that the maximum BDM communication speed is achieved.
- In the “Auto” operation, a “Run” option is available which, if enabled, resets the microcontroller and runs the user application at the end of programming.

4. Troubleshooting

Common Problems and Solutions

This section reports some common problems that may arise during general use. However, working with a specific target device may cause device-specific issues.

Communication Can't Be Established with inDART-HCS12

1. Make sure the inDART-HCS12 in-circuit debugger is connected to the PC and powered on. inDART-HCS12 is powered by the USB connection.
2. Make sure you are working with the correct inDART hardware model. To view/change the inDART hardware model in use, choose **inDART-HCS12 > MCU Configuration** the CodeWarrior HC(S)12 debugger's main menu.
3. If the **inDART-HCS12** menu is not present in the CodeWarrior HC(S)12 debugger's main menu, this is because the target has not been recognized by CodeWarrior ("No link to Target" appears in the status bar). In this case, you must do the following:
 - From the **GDI** menu, choose **MCU Configuration** and verify that the hardware code and device code parameters are set correctly.
 - Press the "**Communication Settings**" button and verify that all of the communication parameters are set correctly.
4. Make sure the demo board/target application board is powered on and the target microcontroller is working. Programming and debugging rely on a BDM communication between the inDART-HCS12 board and the demo board/target application. This means that, in order to work correctly, the target microcontroller must be running. In particular, make sure that:
 - The BDM cable is connected to the demo board/target application's BDM connector.
 - All of the required BDM connector signals are correctly tied to the target microcontroller.

Stepping Execution is Slow

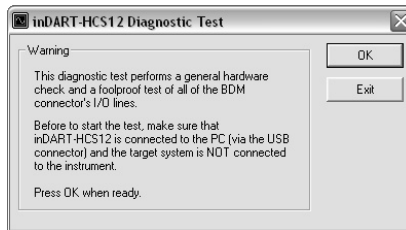
When the *Memory* window is open, step commands may execute slower, since the *Memory* window contents need to be refreshed after every step.

Diagnostic Test

inDART-HCS12 has built-in self-test capabilities. This means that you can verify by yourself, at any time, the correct operation of the instrument's hardware. The diagnostic test is accessible through a small, separate test utility. To perform the diagnostic test:

4

1. Start the inDART-HCS12 diagnostic test utility by selecting **Start > Programs > SofTec Microsystems > inDART-HCS12 > inDART-HCS12 Diagnostic Test**. The following dialog box will appear.



The *inDART-HCS12 Diagnostic Test Utility*

2. Make sure that inDART-HCS12 is connected to the host PC;
3. Make sure that no target system is connected to inDART-HCS12;
4. Click **“OK”**. The test will be performed. In case of problems, please contact our technical support.

Getting Technical Support

Technical assistance is provided free to all customers. For technical assistance, documentation and information about products and services, please refer to your local SofTec Microsystems partner.

SofTec Microsystems offers its customers a free technical support service at support@softecmicro.com. Before getting in contact with us, we advise you firstly to visit our online FAQ section and to be sure you are working with the latest version of the inDART-HCS12 user interface (upgrades are available free of charge at <http://www.softecmicro.com/download.html>).

Appendix A. Electrical and Physical Specifications

Operating Voltage	4.75 to 5.0 V DC (provided by the USB connection)
Power Consumption	300 mA (max)
V_{DD} (Voltage Supplied by the V_{DD} Line on the “VDD OUT” Connector)	3.0 V or 5.0 V
I_{DD} (Current Drawn by the Target via the V_{DD} Line on the “VDD OUT” Connector)	100 mA (max)
Dimensions	95 x 55 x 15 mm
Weight	25 g
Operating Temperature	0 °C to 50 °C
Storage Temperature	-20 °C to 70 °C
Humidity	90% (without condensation)

Electrical and Physical Specifications

A

