

# ***J-Link ColdFire*** ***BDM 26***

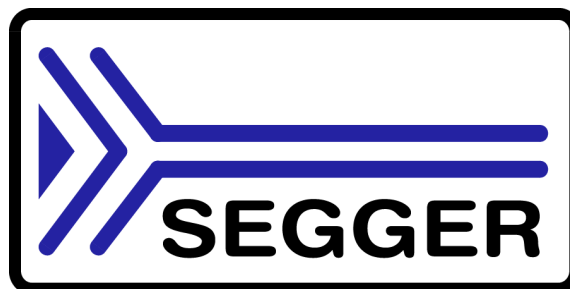
**User guide of the  
J-Link ColdFire BDM 26**



**Manual Rev. 5**

**Date: January 6, 2009**

**Document: UM08009**



**A product of SEGGER Microcontroller GmbH & Co. KG**

**[www.segger.com](http://www.segger.com)**

## Disclaimer

Specifications written in this document are believed to be accurate, but are not guaranteed to be entirely free of error. The information in this manual is subject to change for functional or performance improvements without notice. Please make sure your manual is the latest edition. While the information herein is assumed to be accurate, SEGGER MICROCONTROLLER GmbH & Co. KG (the manufacturer) assumes no responsibility for any errors or omissions. The manufacturer makes and you receive no warranties or conditions, express, implied, statutory or in any communication with you. The manufacturer specifically disclaims any implied warranty of merchantability or fitness for a particular purpose.

## Copyright notice

You may not extract portions of this manual or modify the PDF file in any way without the prior written permission of the manufacturer. The software described in this document is furnished under a license and may only be used or copied in accordance with the terms of such a license.

© 2004 - 2008 SEGGER Microcontroller GmbH & Co. KG, Hilden / Germany

## Trademarks

Names mentioned in this manual may be trademarks of their respective companies.

Brand and product names are trademarks or registered trademarks of their respective holders.

## Contact address

SEGGER Microcontroller GmbH & Co. KG

In den Weiden 11  
D-40721 Hilden

Germany

Tel. +49 2103-2878-0

Fax. +49 2103-2878-28

Email: [support@segger.com](mailto:support@segger.com)

Internet: <http://www.segger.com>

## Manual versions

This manual describes the latest software version. If any error occurs, please inform us and we will try to assist you as soon as possible.

For further information on topics or routines not yet specified, please contact us.

Revision	Date	By	Explanation
5	081219	AG	Chapter "Flash download" added.
4	080605	AG	Chapter "Working with J-Link": Section "Using J-Link with different debuggers" added.
3	070924	AG	Several spelling corrections.
2	070912	AG	Chapter "Working with J-Link": Section "Command strings" updated, "Supply-PowerDefault" command added.
1	070904	AG	Initial version.

## Software versions

Refers to Release.html for information about the changes of the software versions.

# About this document

---

This document describes J-Link ColdFire® BDM 26. It provides an overview over the major features of J-Link ColdFire® BDM 26, gives you some background information about BDM and describes J-Link ColdFire® BDM 26 related software packages available from Segger. Finally, the chapter *Support and FAQs* on page 51 helps to troubleshoot common problems.

For simplicity, we will refer to J-Link ColdFire® BDM 26 as J-Link in this manual.

## Typographic conventions

This manual uses the following typographic conventions:

Style	Used for
Body	Body text.
<b>Keyword</b>	Text that you enter at the command-prompt or that appears on the display (that is system functions, file- or pathnames).
<i>Reference</i>	Reference to chapters, tables and figures or other documents.
<b>GUIElement</b>	Buttons, dialog boxes, menu names, menu commands.

**Table 1.1: Typographic conventions**



**SEGGER Microcontroller GmbH & Co. KG** develops and distributes software development tools and ANSI C software components (middleware) for embedded systems in several industries such as telecom, medical technology, consumer electronics, automotive industry and industrial automation.

SEGGER's intention is to cut software development-time for embedded applications by offering compact flexible and easy to use middleware, allowing developers to concentrate on their application.

Our most popular products are emWin, a universal graphic software package for embedded applications, and embOS, a small yet efficient real-time kernel. emWin, written entirely in ANSI C, can easily be used on any CPU and most any display. It is complemented by the available PC tools: Bitmap Converter, Font Converter, Simulator and Viewer. embOS supports most 8/16/32-bit CPUs. Its small memory footprint makes it suitable for single-chip applications.

Apart from its main focus on software tools, SEGGER develops and produces programming tools for flash microcontrollers, as well as J-Link, a JTAG emulator to assist in development, debugging and production, which has rapidly become the industry standard for debug access to ARM cores.

**Corporate Office:**

<http://www.segger.com>

**United States Office:**

<http://www.segger-us.com>

## EMBEDDED SOFTWARE (Middleware)



**emWin**

**Graphics software and GUI**

emWin is designed to provide an efficient, processor- and display controller-independent graphical user interface (GUI) for any application that operates with a graphical display. Starterkits, eval- and trial-versions are available.



**embOS**

**Real Time Operating System**

embOS is an RTOS designed to offer the benefits of a complete multitasking system for hard real time applications with minimal resources. The profiling PC tool embOSView is included.



**emFile**

**File system**

emFile is an embedded file system with FAT12, FAT16 and FAT32 support. emFile has been optimized for minimum memory consumption in RAM and ROM while maintaining high speed. Various Device drivers, e.g. for NAND and NOR flashes, SD/MMC and CompactFlash cards, are available.



**emUSB**

**USB device stack**

A USB stack designed to work on any embedded system with a USB client controller. Bulk communication and most standard device classes are supported.

## SEGGER TOOLS

**Flasher**

**Flash programmer**

Flash Programming tool primarily for microcontrollers.

**J-Link**

**JTAG emulator for ARM cores**

USB driven JTAG interface for ARM cores.

**J-Trace**

**JTAG emulator with trace**

USB driven JTAG interface for ARM cores with Trace memory. supporting the ARM ETM (Embedded Trace Macrocell).

**J-Link Related Software**

Add-on software to be used with SEGGER's industry standard JTAG emulator, this includes flash programming software and flash breakpoints.



# Table of Contents

1	Introduction .....	7
1.1	J-Link overview .....	8
1.1.1	Features of J-Link .....	8
1.2	Specifications .....	9
1.2.1	Specifications for J-Link .....	9
1.2.2	Download speed .....	9
1.3	Requirements .....	9
2	Setup .....	11
2.1	Installing the J-Link software and documentation .....	12
2.1.1	Setup procedure .....	13
2.1.2	Verifying correct driver installation .....	14
2.2	Uninstalling the J-Link USB driver .....	16
2.3	Connecting the target system .....	17
2.3.1	Power-on sequence .....	17
2.3.2	Verifying target device connection .....	17
2.3.3	Problems .....	17
3	J-Link related software .....	19
3.1	J-Link related software .....	20
3.1.1	J-Link software and documentation .....	20
3.1.2	List of additional software packages .....	20
3.2	J-Link software and documentation in detail .....	21
3.2.1	J-Link Commander (Command line tool) .....	21
3.3	Additional software packages in detail .....	21
3.3.1	J-Link Software Developer Kit (SDK) .....	21
3.4	Using the JLinkCF.dll .....	22
3.4.1	What is the JLinkCF.dll? .....	22
3.4.2	Updating the DLL in third-party programs .....	22
3.4.3	Determining the version of JLinkCF.dll .....	22
3.4.4	Determining which DLL is used by a program .....	23
4	Working with J-Link .....	25
4.1	Supported ColdFire® Cores .....	26
4.2	Command strings .....	26
4.2.1	List of available commands .....	26
4.2.2	Using command strings .....	29
4.3	Using J-Link with different debuggers .....	31
4.3.1	Using J-Link with IAR Embedded Workbench for ColdFire® .....	31
4.3.2	Using J-Link with Freescale CodeWarrior for ColdFire® .....	32
5	Flash download .....	33
5.1	Introduction .....	34
5.2	Licensing .....	35
5.3	Supported devices .....	36
5.4	Using flash download with different debuggers .....	37
5.4.1	IAR Embedded Workbench .....	37

6	Hardware .....	39
6.1	BDM Connector .....	40
6.1.1	Pinout.....	41
6.2	How to determine the hardware version .....	42
7	Background information .....	43
7.1	BDM .....	44
7.2	The ColdFire® core .....	45
7.2.1	Processor modes .....	45
7.2.2	Registers .....	45
7.2.3	Breakpoints and watchpoints .....	47
7.3	Flash programming .....	48
7.3.1	How does flash programming via J-Link work ? .....	48
7.3.2	Data download to RAM .....	48
7.3.3	Available options for flash programming .....	48
7.4	J-Link firmware .....	49
7.4.1	Firmware update .....	49
7.4.2	Invalidating the firmware .....	49
8	Support and FAQs .....	51
8.1	Troubleshooting .....	52
8.1.1	General procedure.....	52
8.1.2	Typical problem scenarios .....	52
8.2	Contacting support .....	53
8.3	Frequently Asked Questions.....	54
9	Glossary.....	55
10	Literature and references.....	59

# Chapter 1

## Introduction

---

This chapter gives a short overview about J-Link.

## 1.1 J-Link overview

J-Link is a BDM emulator designed for ColdFire® cores. It connects via USB to a PC running Microsoft Windows 2000, Windows XP, Windows 2003, or Windows Vista. J-Link has a built-in 26-pin BDM connector, which is compatible with the standard 26-pin connector defined by Freescale.

### 1.1.1 Features of J-Link

- USB 2.0 interface
- Easy to use: Fully plug and play compatible
- Any ColdFire® V2/3/4 supported
- Download speed up to 120 Kbytes/second
- Seamless integration into IAR Embedded Workbench
- No power supply required, powered through USB
- 5V Power can be supplied to the target (on pin 1, KS-power)
- Maximum interface speed: 2 MHz (Multilink: 1 MHz)
- Automatic core recognition
- All interface signals can be monitored and target voltage can be measured
- A 26-pin standard connector
- A USB and 26-pin flat cable included
- Wide target voltage range: 1.2V - 5V
- J-Mem (live memory view/edit) included
- A J-Link server (connects J-Link via TCP/IP) included
- A Software Developer Kit (SDK) available: write your own application using J-Link, directly accessing the core
- Flash download: debugger can download into internal ColdFire-flash
- Applications can be debugged in RAM or flash



## 1.2 Specifications

### 1.2.1 Specifications for J-Link

Power Supply	USB powered <50mA
USB Interface	USB 2.0, full speed
Target Interface	BDM 26-pin
Serial Transfer Rate between J-Link and Target	up to 2 MHz
Supported Target Voltage	1.2 - 3.3 V (5V adapter available)
Target supply voltage	4.5V .. 5V (if powered with 5V on USB)
Target supply current	Max. 300mA
Operating Temperature	+5°C ... +60°C
Storage Temperature	-20°C ... +65 °C
Relative Humidity (non-condensing)	<90% rH
Size (without cables)	100mm x 53mm x 27mm
Weight (without cables)	70g
Electromagnetic Compatibility (EMC)	EN 55022, EN 55024
Supported OS	Microsoft Windows 2000 Microsoft Windows XP Microsoft Windows XP x64 Microsoft Windows 2003 Microsoft Windows 2003 x64 Microsoft Windows Vista Microsoft Windows Vista x64

**Table 1.1: J-Link specifications**

### 1.2.2 Download speed

The following table lists performance values (Kbytes/second) for writing to memory (RAM):

Hardware	Memory download
J-Link Rev. 1	120 KBytes/second

**Table 1.2: Download speed differences between hardware revisions**

**Note:** The actual speed depends on various factors, such as BDM frequency, target CPU speed, host system used etc.

## 1.3 Requirements

### Host System

To use J-Link you need a host system running Windows 2000, Windows XP, Windows 2003, or Windows Vista.

### Target System

A ColdFire® target system is required. The system should have a standardized 26-pin connector as defined by Freescale.



# Chapter 2

## Setup

---

This chapter describes the setup procedure required to work with J-Link. Primarily, this includes the installation of the J-Link software and documentation, which also includes a kernel mode J-Link USB driver in your host system.

## 2.1 Installing the J-Link software and documentation

J-Link is shipped with a command line tool, a DLL for using J-Link with Freescale CodeWarrior for ColdFire and J-Link USB driver.

Refer to chapter *J-Link related software* on page 19 for an overview about the J-Link software and its documentation.

## 2.1.1 Setup procedure

To install the J-Link software and documentation, follow this procedure:

**Note:** We recommend to check if a newer version of the J-Link software and documentation is available for download before starting the installation. Check therefore the J-Link related download section of our website:

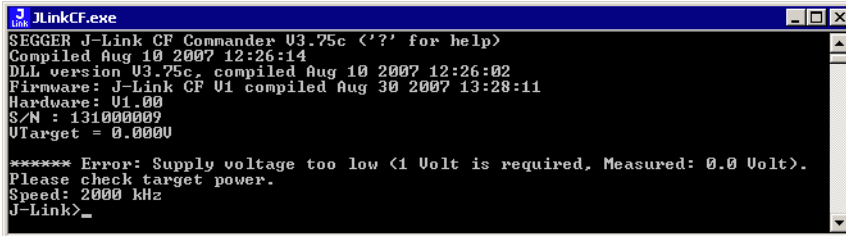
*[http://www.segger.com/download\\_jlink\\_cf.html](http://www.segger.com/download_jlink_cf.html)*

1. Before you connect your J-Link into your computer's USB port, extract J-Link software and documentation package **JLinkCF\_V<VersionNumber>.zip**. The software and documentation package includes the certified J-Link USB driver. Start installing the USB drivers by double clicking **USBdriver/InstallDrivers.exe**.
2. Connect your J-Link via USB with your PC. The J-Link will be identified and after a short period the J-Link LED stops rapidly flashing and stays on permanently.

## 2.1.2 Verifying correct driver installation

To verify the correct installation of the driver, disconnect and reconnect J-Link to the USB port. During the enumeration process which takes about 2 seconds, the LED on J-Link is flashing. After successful enumeration, the LED stays on permanently.

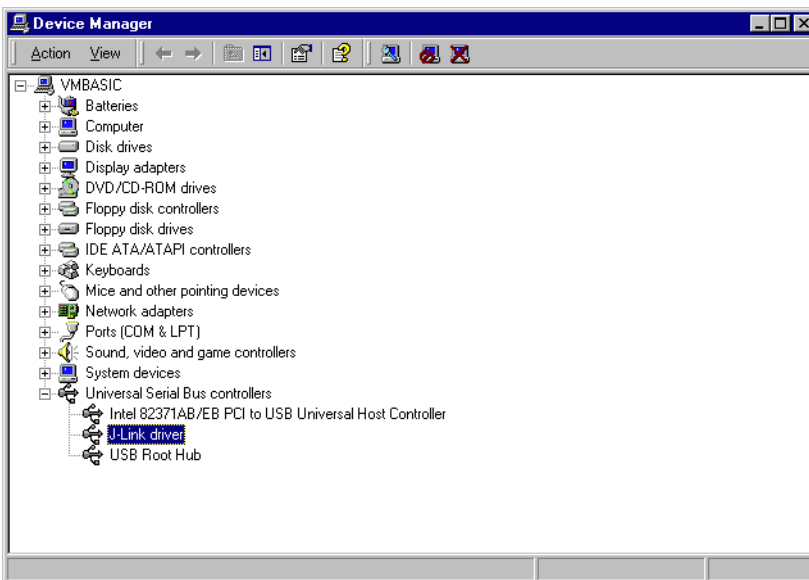
Start the provided example application **JLinkCF.exe**, which should display the compilation time of the J-Link firmware, the serial number, a target voltage of 0.000V, a complementary error message which says that the supply voltage is too low if no target is connected to J-Link, and the speed selection. The screenshot below shows an example.



```
JLinkCF.exe
SEGGER J-Link CF Commander V3.75c <'?' for help>
Compiled Aug 10 2007 12:26:14
DLL version V3.75c, compiled Aug 10 2007 12:26:02
Firmware: J-Link CF V1 compiled Aug 30 2007 13:28:11
Hardware: V1.00
S/N : 131000009
VTarget = 0.0000

***** Error: Supply voltage too low (1 Volt is required, Measured: 0.0 Volt).
Please check target power.
Speed: 2000 kHz
J-Link>
```

In addition, you can verify the driver installation by consulting the Windows device manager. If the driver is installed and your J-Link is connected to your computer, the device manager should list the J-Link USB driver as a node below "Universal Serial Bus controllers" as shown in the following screenshot:



Right-click on the driver to open a context menu which contains the command **Properties**. If you select this command, a **J-Link driver Properties** dialog box is opened and should report: **This device is working properly**.



If you experience problems, refer to the chapter *Support and FAQs* on page 51 for help. You can select the **Driver** tab for detailed information about driver provider, version, date and digital signer.



## 2.2 Uninstalling the J-Link USB driver

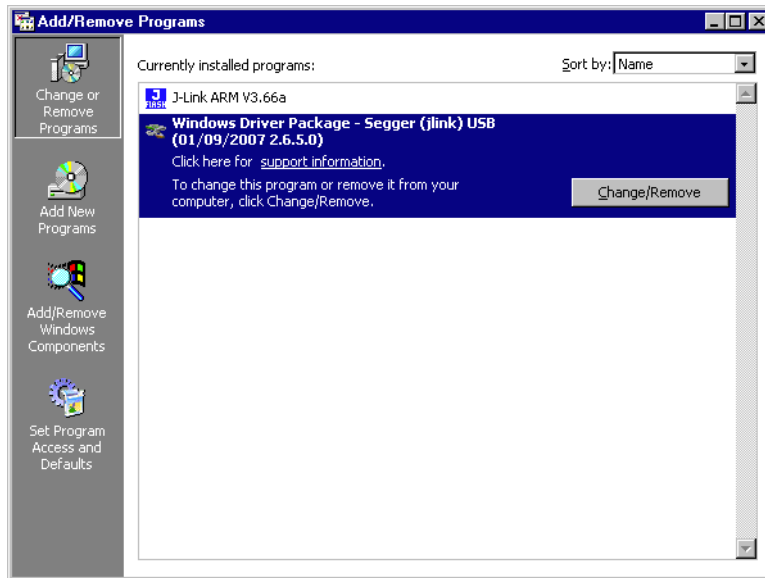
If J-Link is not properly recognized by Windows and therefore does not enumerate, it make sense to uninstall the J-Link USB driver.

This might be the case when:

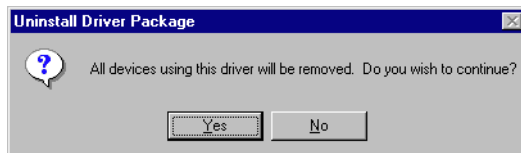
- The LED on the J-Link is rapidly flashing.
- The J-Link is recognized as **Unknown Device** by Windows.

To have a clean system and help Windows to reinstall the J-Link driver, follow this procedure:

1. Disconnect J-Link from your PC.
2. Open the **Add/Remove Programs** dialog box (**Start** > **Settings** > **Control Panel** > **Add/Remove Programs**) and select **Windows Driver Package - Segger (jlink) USB** and click the **Change/Remove** button.



3. Confirm the uninstallation process.





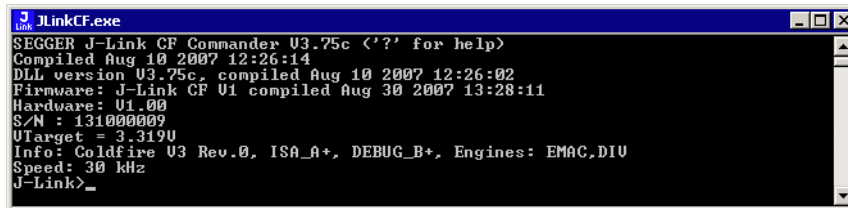
## 2.3 Connecting the target system

### 2.3.1 Power-on sequence

In general, J-Link should be powered on before you connect it with the target device. That means you should first connect J-Link with the host system via USB and then connect J-Link with the target device via BDM. Activate power supply for device after you connected J-Link to it.

### 2.3.2 Verifying target device connection

If the USB driver is working properly and your J-Link is connected with the host system, you can connect J-Link to your target hardware. Then start **JLinkCF.exe** again which should now display the same J-Link related information as above. In addition it should report that it found a BDM target. The screenshot below shows the output of **JLinkCF.exe**. As you can see, it reports a J-Link with one BDM device connected.



```

SEGGER J-Link CF Commander V3.75c ('?' for help)
Compiled Aug 10 2007 12:26:14
DLL version V3.75c, compiled Aug 10 2007 12:26:02
Firmware: J-Link CF V1 compiled Aug 30 2007 13:28:11
Hardware: V1.00
S/N : 131000009
VTarget = 3.319V
Info: Coldfire V3 Rev.0, ISA_A+, DEBUG_B+, Engines: EMAC,DIU
Speed: 30 kHz
J-Link>_
  
```

### 2.3.3 Problems

If you experience problems with any of the steps described above, read the chapter *Support and FAQs* on page 51 for troubleshooting tips. If you still do not find appropriate help there and your J-Link is an original Segger product, you may contact Segger support via e-mail. Provide the necessary information about your target processor, board etc. and we will try to solve your problem. A checklist of the required information together with the contact information can be found in chapter *Support and FAQs* on page 51 as well.



# Chapter 3

## J-Link related software

---

This chapter describes Segger's J-Link related software portfolio which covers nearly all phases of developing embedded applications.

## 3.1 J-Link related software

### 3.1.1 J-Link software and documentation

J-Link is shipped with J-Link USB driver, a command line tool and a DLL for using J-Link with Freescale CodeWarrior for ColdFire®.

Software	Description
<b>JLinkCF.dll</b>	DLL for using J-Link with third-party programs.
unit_cfz.dll	DLL for using J-Link with Freescale CodeWarrior for ColdFire®. For more information about how to use J-Link with Freescale CodeWarrior for ColdFire® please refer to <i>Using J-Link with Freescale CodeWarrior for ColdFire®</i> on page 32
<b>JLinkCF.exe</b>	Free command-line tool with basic functionality for target analysis.
USBDriver	J-Link USB driver.

**Table 3.1: J-Link related software**

### 3.1.2 List of additional software packages

The software packages listed below are available upon request from [www.segger.com](http://www.segger.com).

Software	Description
J-Link Software Developer Kit (SDK)	The J-Link Software Developer Kit is needed if you want to write your own program with J-Link.

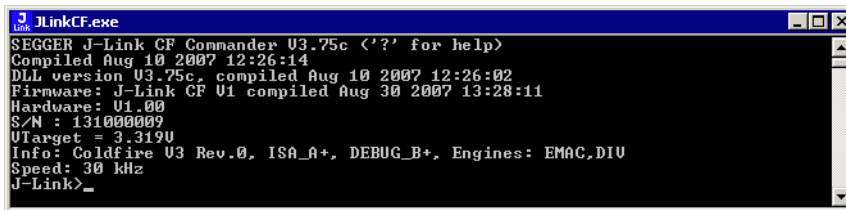
**Table 3.2: J-Link additional software packages**

## 3.2 J-Link software and documentation in detail

The J-Link software documentation can be downloaded from [www.segger.com/download\\_jlink\\_cf.html](http://www.segger.com/download_jlink_cf.html).

### 3.2.1 J-Link Commander (Command line tool)

J-Link Commander (**JLinkCF.exe**) is a tool that can be used for verifying proper installation of the USB driver and to verify the connection to the ColdFire® chip, as well as for simple analysis of the target system. It permits some simple commands, such as memory dump, halt, step and go, as well as some more in-depths analysis of the state of the ColdFire® core.



## 3.3 Additional software packages in detail

The packages described in this section are not available for download. If you wish to use one of them, contact SEGGER Microcontroller System directly.

### 3.3.1 J-Link Software Developer Kit (SDK)

The J-Link Software Developer Kit is needed if you want to write your own program with J-Link. The J-Link DLL is a standard Windows DLL typically used from C programs (Visual Basic or Delphi projects are also possible). It makes the entire functionality of J-Link available through its exported functions, such as halting/stepping the ColdFire® core, reading/writing CPU and BDM registers and reading/writing memory. Therefore it can be used in any kind of application accessing a ColdFire® core. The standard DLL does not have API functions for flash programming. However, the functionality offered can be used for programming the flash. In this case, a flash loader is required. The table below lists some of the included files and their respective purpose.

Files	Contents
<b>GLOBAL.h</b> <b>JLinkCFDLL.h</b>	Header files that must be included to use the DLL functions. These files contain the defines, typedef names, and function declarations.
<b>JLinkCFDLL.lib</b>	A Library that contains the exports of the <b>JLinkCF.dll</b> .
<b>JLinkCF.dll</b>	The DLL itself.
<b>Main.c</b>	A example application, which calls some <b>JLinkCF.dll</b> functions.
<b>JLinkCF.dsp</b> <b>JLinkCF.dsw</b>	Project files of the example application. Double-click <b>JLinkCF.dsw</b> to open the project.
<b>JLinkCFDLL.pdf</b>	Extensive documentation (API, example projects etc.).

Table 3.3: J-Link SDK

## 3.4 Using the J-LinkCF.dll

### 3.4.1 What is the JLinkCF.dll?

The J-LinkCF.dll is a standard Windows DLL typically used from C or C++, but also Visual Basic or Delphi projects. It makes the entire functionality of the J-Link available through the exported functions.

The functionality includes things such as halting/stepping the ColdFire® core, reading/writing CPU and BDM registers, and reading/writing memory. Therefore, it can be used in any kind of application accessing a ColdFire® core.

### 3.4.2 Updating the DLL in third-party programs

The **JLinkCF.dll** can be used by any debugger that is designed to work with it. Some debuggers, like the IAR C-SPY® debugger, are usually shipped with the **JLinkCF.dll** already installed. Anyhow it may make sense to replace the included DLL with the latest one available, to take advantage of improvements in the newer version.

#### 3.4.2.1 Updating the JLinkCF.dll in the IAR Embedded Workbench IDE

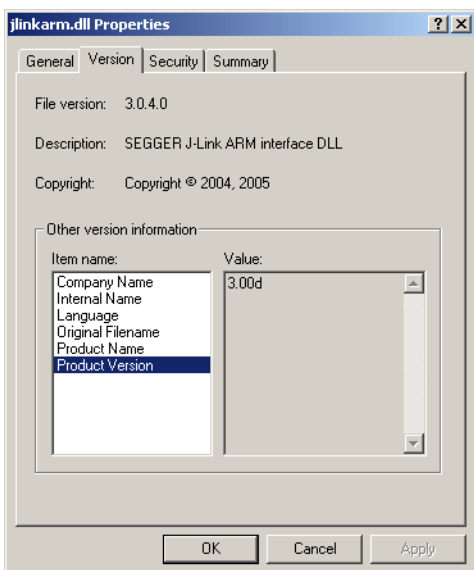
The IAR Embedded Workbench IDE is a high-performance integrated development environment with an editor, compiler, linker, and debugger. The compiler generates very efficient code and is widely used. The IAR Embedded Workbench comes with the **J-LinkCF.dll** in the **cf\bin** subdirectory of the installation directory. To update this DLL, you should backup your original DLL and then replace it with the new one.

Typically, the DLL is located in **C:\Program Files\IAR Systems\Embedded Workbench 5.0\cf\bin\**.

After updating the DLL, it is recommended to verify that the new DLL is loaded as described in *Determining which DLL is used by a program* on page 23.

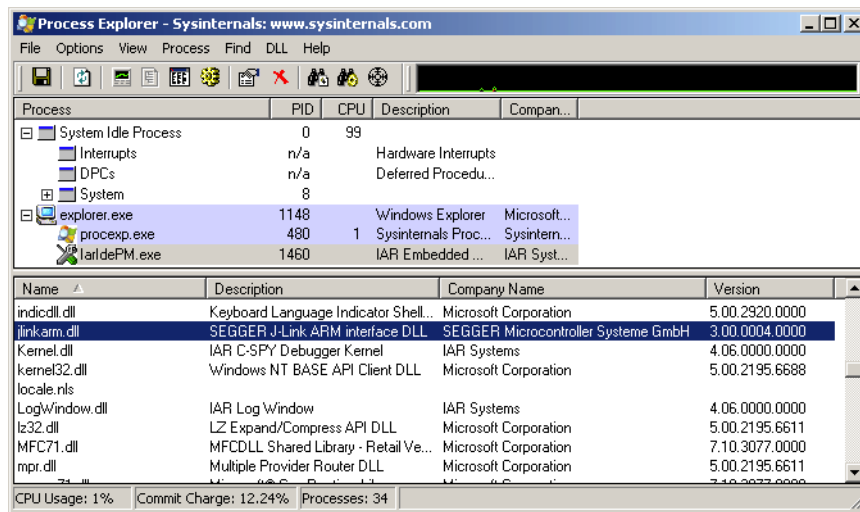
### 3.4.3 Determining the version of JLinkCF.dll

To determine which version of the **JLinkCF.dll** you are facing, the DLL version can be viewed by right-clicking the DLL in windows explorer, and choosing **Properties** from the context menu. Click the **version** tab to display information about the product version.



### 3.4.4 Determining which DLL is used by a program

To verify that the program you are working with is using the DLL you expect it to use, you can investigate which DLLs are loaded by your program with tools like Sysinternals' Process Explorer. It shows you details about the DLLs, used by your program, such as manufacturer and version.



Process Explorer is - at the time of writing - a free utility which can be downloaded from [www.sysinternals.com](http://www.sysinternals.com).





# Chapter 4

## Working with J-Link

---

This chapter describes functionality and how to use J-Link.

## 4.1 Supported ColdFire® Cores

J-Link works with any V2/3/4 ColdFire® core.

## 4.2 Command strings

The behaviour of J-Link can be customized via command strings passed to the `JLinkCF.d11` which controls J-Link. Applications such as the J-Link Commander, but also the C-SPY debugger which is part of the IAR Embedded Workbench, allow passing one or more command strings. Command line strings can be used for passing commands to J-Link (such as switching on target power supply), as well as customize the behaviour (by defining memory regions and other things) of J-Link. The use of command strings enables options which can not be set with the configuration dialog box provided by C-SPY.

### 4.2.1 List of available commands

The table below lists and describes the available command strings.

Command	Description
<code>map exclude</code>	Ignores all memory accesses to specified area.
<code>map illegal</code>	Marks a specified memory region as an illegal memory area
<code>map ram</code>	Specifies location of target RAM.
<code>map reset</code>	Restores the default mapping, which means all memory accesses are permitted.
<code>SupplyPower</code>	Activates/Deactivates power supply over pin 1 of the BDM connector.
<code>SupplyPowerDefault</code>	Activates/Deactivates power supply over pin 1 of the BDM connector permanently.

**Table 4.1: Available command line options**

#### 4.2.1.1 map exclude

This command excludes a specified memory region from all memory accesses. All subsequent memory accesses to this memory region are ignored. This command can be used for excluding more than one memory region by subsequent calls to `JLINK_ExecCommand()`.

##### Syntax

```
map exclude <SAddr>--<EAddr>
```

##### Additional information

Some devices do not allow access of the entire 4Gb memory area. Ideally, the entire memory can be accessed; if a memory access fails, the CPU reports this by switching to abort mode. The CPU memory interface allows halting the CPU via a WAIT signal. On some devices, the WAIT signal stays active when accessing certain unused memory areas. This halts the CPU indefinitely (until RESET) and will therefore end the debug session. This is exactly what happens when accessing critical memory areas. Critical memory areas should not be present in a device; they are typically a hardware design problem. Nevertheless, critical memory areas exist on some devices. To avoid stalling the debug session, a critical memory area can be excluded from access: J-Link will not try to read or write to critical memory areas and instead ignore the access silently. Some debuggers (such as IAR C-SPY) can try to access memory in such areas by dereferencing non-initialized pointers even if the debugged program (the debuggee) is working perfectly. In situations like this, defining critical memory areas is a good solution.

## Example

```
char acOut[256];
char acRead[32];
//
// Exclude memory regions
//
JLINK_ExecCommand("map exclude 0x200000-0x2FFFFFF", acOut, sizeof(acOut));
JLINK_ExecCommand("map exclude 0x400000-0x4FFFFFF", acOut, sizeof(acOut));
//
// Read memory
//
JLINK_ReadMem(0x200000, sizeof(acRead), acRead); // Memory access is ignored
JLINK_ReadMem(0x300000, sizeof(acRead), acRead); // Memory access is permitted
JLINK_ReadMem(0x400000, sizeof(acRead), acRead); // Memory access is ignored
```

### 4.2.1.2 map illegal

This command marks a specified memory region as an illegal memory area. All subsequent memory accesses to this memory region produces a warning message and the memory access is ignored. This command can be used for marking more than one memory region as an illegal area by subsequent calls to **JLINK\_ExecCommand()**

#### Syntax

```
map illegal <SAddr>-<EAddr>
```

### 4.2.1.3 map ram

This command should be used for defining an area in RAM of the target device. The area must be 256-byte aligned. The data which was located in the defined area will not be corrupted. Data which resides in the defined RAM area is saved and will be restored if necessary.

#### Syntax

```
map ram <StartAddressOfArea>-<EndAddressOfArea>
```

## Example

```
char acOut[256];
char acRead[32];
//
// Mark memory regions as illegal areas
//
JLINK_ExecCommand("map illegal 0x200000-0x2FFFFFF", acOut, sizeof(acOut));
JLINK_ExecCommand("map illegal 0x400000-0x4FFFFFF", acOut, sizeof(acOut));
//
// Read memory
//
JLINK_ReadMem(0x200000, sizeof(acRead), acRead); // Produces a warning
JLINK_ReadMem(0x300000, sizeof(acRead), acRead); // Memory access is permitted
JLINK_ReadMem(0x400000, sizeof(acRead), acRead); // Produces a warning
```

### 4.2.1.4 map reset

This command restores the default memory mapping, which means all memory accesses are permitted.

#### Syntax

```
map reset
```

## Example

```
char acOut[256];
char acRead[32];
```

```
//
// Mark memory regions as illegal areas
//
JLINK_ExecCommand("map illegal 0x200000-0x2FFFFFF", acOut, sizeof(acOut));
JLINK_ExecCommand("map illegal 0x400000-0x4FFFFFF", acOut, sizeof(acOut));
//
// Reset memory mapping
//
JLINK_ExecCommand("map reset", acOut, sizeof(acOut));
//
// Read memory
//
JLINK_ReadMem(0x200000, sizeof(acRead), acRead); // Memory access is permitted
JLINK_ReadMem(0x300000, sizeof(acRead), acRead); // Memory access is permitted
JLINK_ReadMem(0x400000, sizeof(acRead), acRead); // Memory access is permitted
```

### 4.2.1.5 SupplyPower

This command activates power supply over pin 1 of the BDM connector. The KS (Kick-start) versions of J-Link ColdFire® have the 5V supply over pin 1 activated by default. This feature is useful for some evaluation boards that can be powered over the BDM connector.

#### Syntax

Enable power supply:	<b>SupplyPower = 1</b>
Disable power supply:	<b>SupplyPower = 0</b>

#### Example

```
char acOut[256];

//
// Enable power supply
//
JLINK_ExecCommand("SupplyPower = 1", acOut, sizeof(acOut));
```

### 4.2.1.6 SupplyPowerDefault

This command activates power supply over pin 1 of the BDM connector. The KS (Kick-start) versions of J-Link ColdFire® have the 5V supply over pin 1 activated by default. This feature is useful for some eval boards that can be powered over the BDM connector.

#### Syntax

Enable power supply permanently:	<b>SupplyPowerDefault = 1</b>
Disable power supply permanently:	<b>SupplyPowerDefault = 0</b>

#### Example

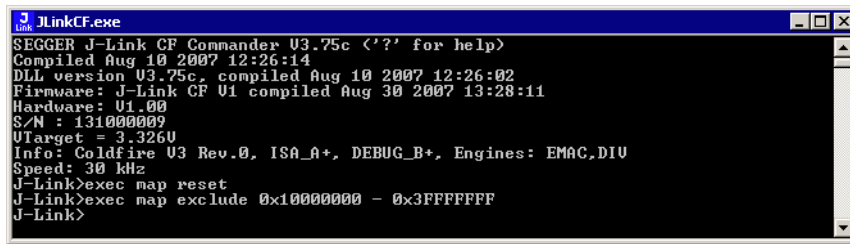
```
char acOut[256];

//
// Enable power supply
//
JLINK_ExecCommand("SupplyPowerDefault = 1", acOut, sizeof(acOut));
```

## 4.2.2 Using command strings

### 4.2.2.1 J-Link Commander

The J-Link command strings can be tested with the J-Link Commander. Use the command **exec** supplemented by one of the command strings.



```

JLinkCF.exe
SEGGER J-Link CF Commander V3.75c <'?' for help>
Compiled Aug 10 2007 12:26:14
DLL version V3.75c, compiled Aug 10 2007 12:26:02
Firmware: J-Link CF V1 compiled Aug 30 2007 13:28:11
Hardware: V1.00
S/N : 131000009
VTarget = 3.326V
Info: Coldfire V3 Rev.0, ISA_A+, DEBUG_B+, Engines: EMAC,DIU
Speed: 30 kHz
J-Link>exec map reset
J-Link>exec map exclude 0x10000000 - 0x3FFFFFFF
J-Link>
  
```

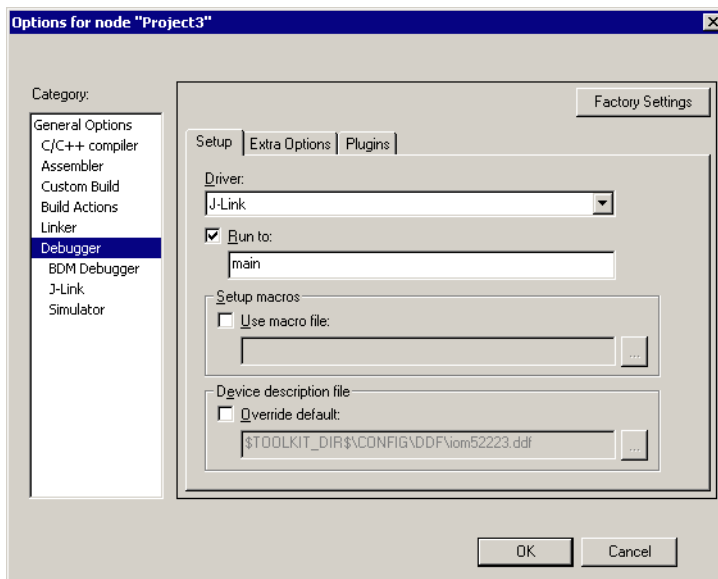
#### Example

```

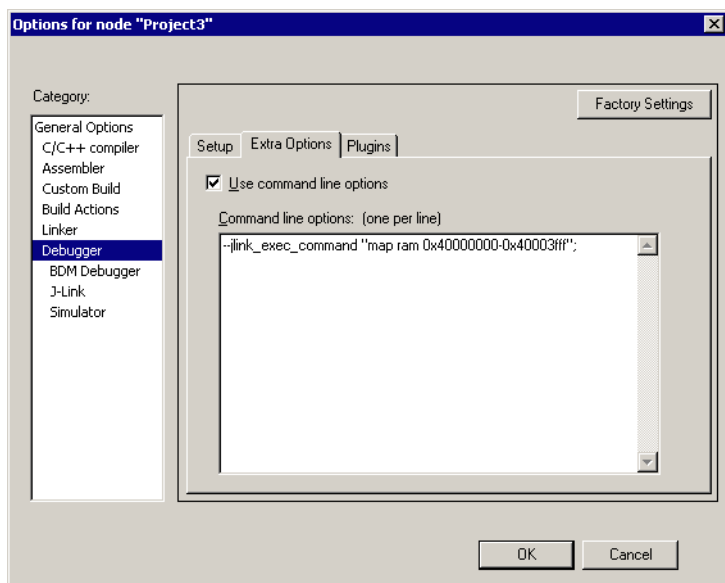
exec SupplyPower = 1
exec map reset
exec map exclude 0x10000000-0x3FFFFFFF
  
```

### 4.2.2.2 IAR Embedded Workbench

To supply J-Link command strings using the C-SPY debugger of the IAR Embedded Workbench, open the **Project options** dialog box and select **Debugger**.



On the **Extra Options** page, select **Use command line options**. Enter **--jlink\_exec\_command "<CommandLineOption>"** in the textfield, as shown in the screenshot below.



If more than one command should be used separate the commands with semicolon.

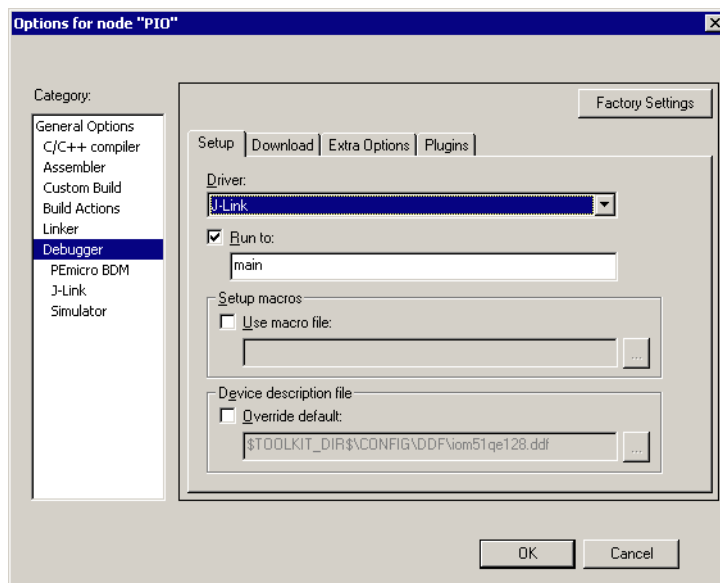
## 4.3 Using J-Link with different debuggers

Currently J-Link can be used with the following debuggers:

- IAR Embedded Workbench for ColdFire®
- Freescale CodeWarrior for ColdFire®

### 4.3.1 Using J-Link with IAR Embedded Workbench for ColdFire®

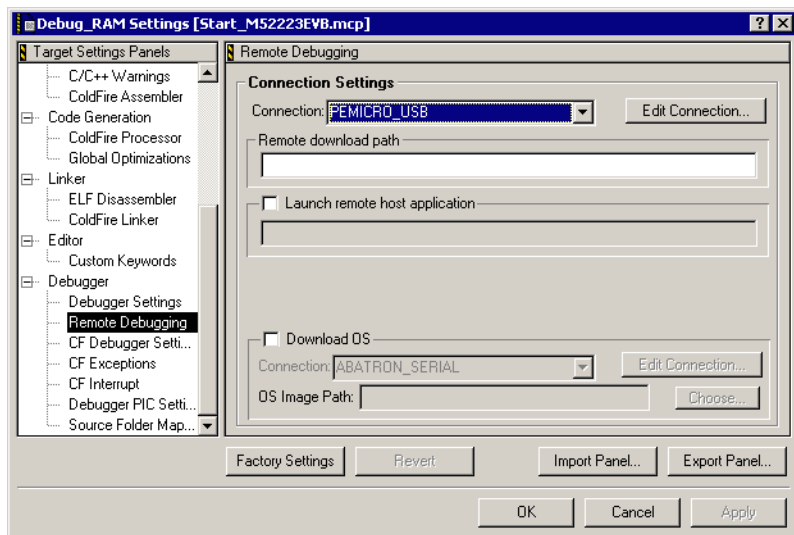
The IAR Embedded Workbench already comes with a JLinkCF.dll so it can be used "out-of-the-box" with J-Link. To use J-Link with IAR Embedded Workbench for ColdFire® simply open the **Project options** dialog box and select **Debugger**. Change the **Driver** settings to **J-Link** as shown in the screenshot below.



In order to update an old version of the **JLinkCF.dll** please refer to *Updating the DLL in third-party programs* on page 22.

### 4.3.2 Using J-Link with Freescale CodeWarrior for ColdFire®

J-Link can be used with Freescale CodeWarrior for ColdFire® via USB connection. Currently CodeWarrior does not come with an appropriate DLL for J-Link, you have to copy the `unit_cfz.dll` and the `JLinkCF.dll` from the J-Link installation directory to the `/bin/` folder of the CodeWarrior installation directory. You should backup the original `unit_cfz.dll` before overwriting it with the one which comes with J-Link. To tell CodeWarrior to use J-Link as remote debug connection just choose **Target settings** and select **Debugger | Remote debugging**. Change **Connection Setting** to **PEMICRO\_USB** as shown in the screenshot below.





# Chapter 5

## Flash download

---

This chapter describes how flash download with J-Link work. In addition to that it contains a list of supported microcontrollers for J-Link.

## 5.1 Introduction

The `JLinkCF.dll` is able to download to internal flash memory of ColdFire devices. This feature requires an additional license which can be purchased from SEGGER.

## 5.2 Licensing

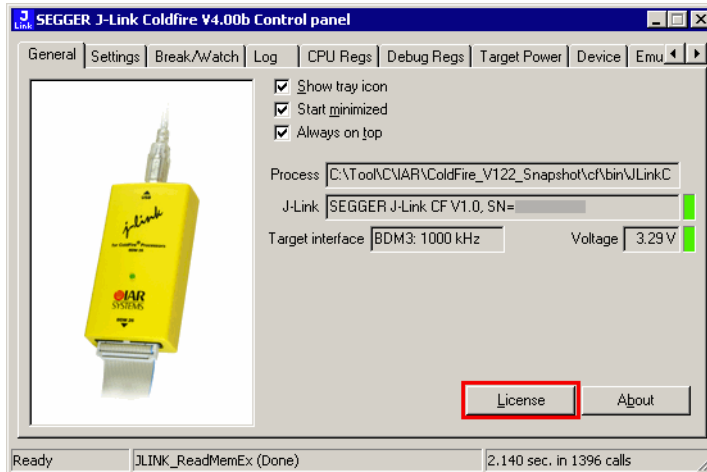
The standard J-Link does not come with a built-in license for flash download. You will need to obtain a license for every J-Link.

To purchase a key-based license, please contact [sales@segger.com](mailto:sales@segger.com).

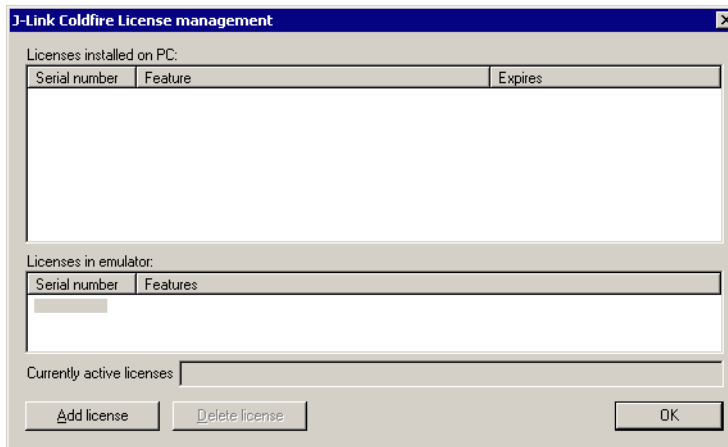
### Entering a license

The easiest way to enter a license is the following:

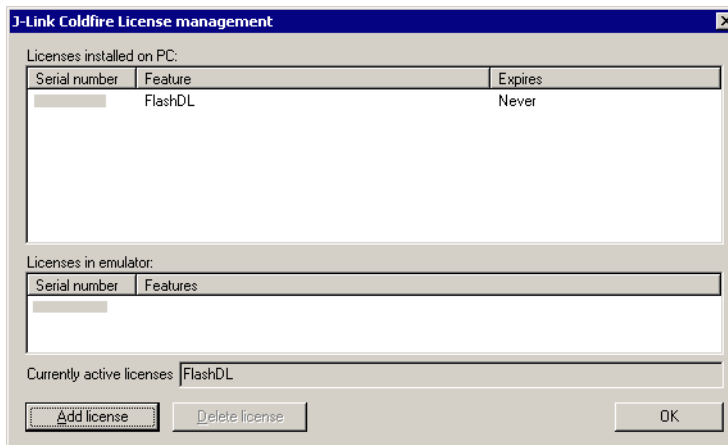
Open the J-Link control panel window, go to the **General** tab and choose **License**.



Now the J-Link license manager will open and show all licenses, both key-based and built-in licenses of J-Link.



Now choose **Add license** to add one or more new licenses. Enter your license(s) and choose **OK**. Now the licenses should have been added.



## 5.3 Supported devices

The following table lists the microcontrollers for which flash download is available.

**Note:** Only the devices listed below are currently supported with the flash download feature. Currently, flash download work on the internal flash of the devices only. It is customer's responsibility to make sure that the device he wants to use flash programming with, is supported. In case of doubt, you should contact SEGGER and ask for a trial license.

The device is selected by its device identifier.

Manufacturer	Device ID	Devices
Freescale	MCF5211	MCF5211
Freescale	MCF5212	MCF5212
Freescale	MCF5213	MCF5213
Freescale	MCF5214	MCF5214
Freescale	MCF5216	MCF5216
Freescale	MCF52100	MCF52100
Freescale	MCF52110	MCF52110
Freescale	MCF52210	MCF52210
Freescale	MCF52211	MCF52211
Freescale	MCF52212	MCF52212
Freescale	MCF52213	MCF52213
Freescale	MCF52221	MCF52221
Freescale	MCF52223	MCF52223
Freescale	MCF52230	MCF52230
Freescale	MCF52231	MCF52231
Freescale	MCF52232	MCF52232
Freescale	MCF52233	MCF52233
Freescale	MCF52234	MCF52234
Freescale	MCF52235	MCF52235
Freescale	MCF52236	MCF52236
Freescale	MCF52252	MCF52252
Freescale	MCF52254	MCF52254
Freescale	MCF52255	MCF52255
Freescale	MCF52256	MCF52256
Freescale	MCF52258	MCF52258
Freescale	MCF52259	MCF52259
Freescale	MCF5280	MCF5280
Freescale	MCF5281	MCF5281
Freescale	MCF5282	MCF5282

**Table 5.1: Supported microcontrollers**

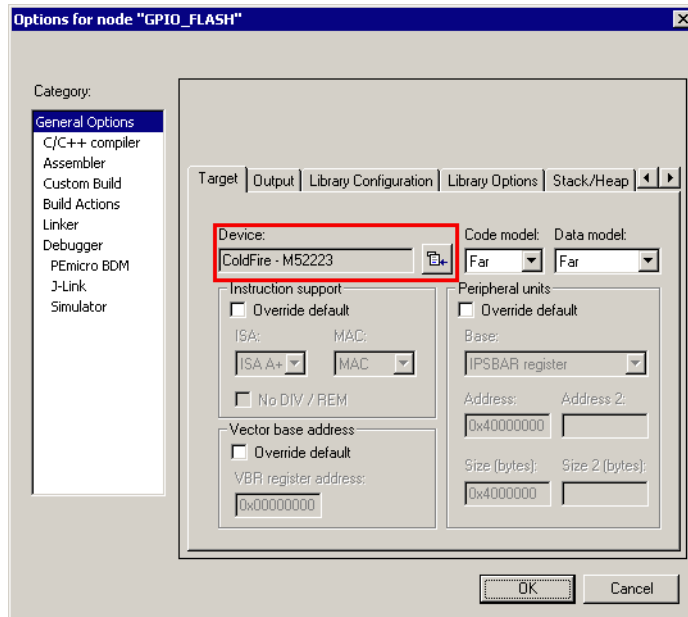
## 5.4 Using flash download with different debuggers

The flash download feature can be used by different debuggers, such as IAR Embedded Workbench. For different debuggers there are different steps required to enable flash download, which will be explained in this section.

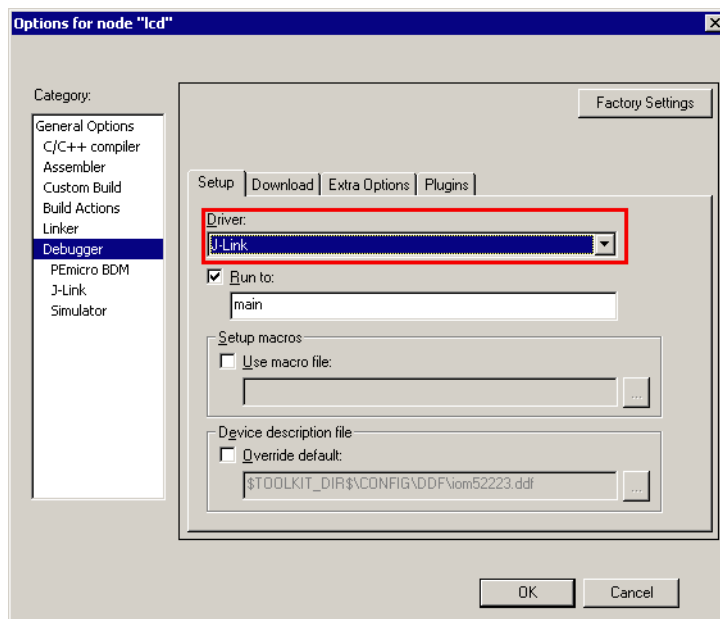
### 5.4.1 IAR Embedded Workbench

To use the J-Link flash download feature with IAR Embedded Workbench is quite simple:

First, choose the right device in the project settings if not already done. The device settings can be found at **Project->Options->General Options->Target**.

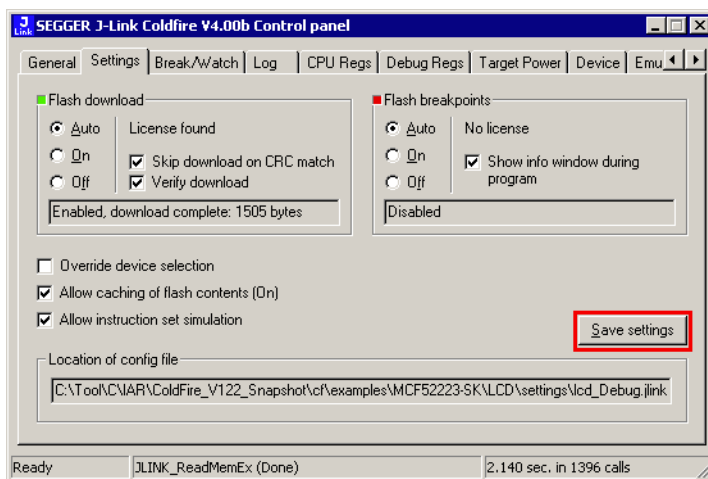


Make sure that J-Link is the selected emulator. The emulator settings can be found at **Project->Options->Debugger->Setup**.



If you use the IAR project for the first time, a settings file is created in which the configuration of the control panel is saved. This settings file is created for every project configuration (e.g. Debug\_RAM, Debug\_FLASH), so you can save different control panel configurations for different project configurations. When the debug ses-

sion starts, you should see the selected target in the **Device** tab of the J-Link status window. When the debug session is running you can modify the settings regarding flash download, in the **Settings** tab and save them to the settings file.



# Chapter 6

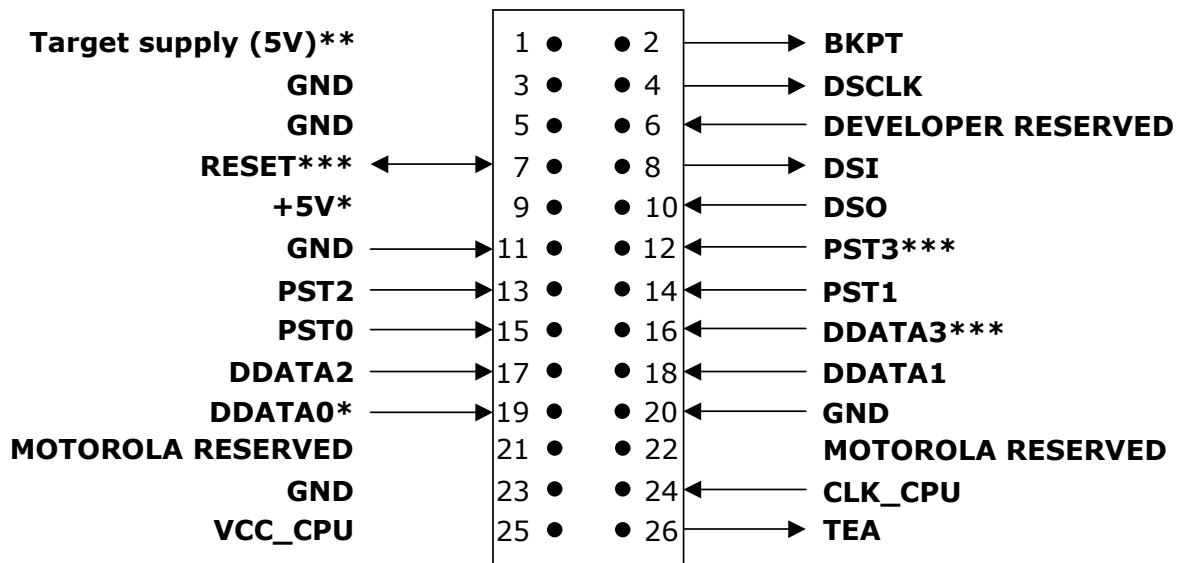
## Hardware

---

This chapter gives an overview about J-Link specific hardware details, such as the pinouts and available adapters.

## 6.1 BDM Connector

J-Link has a standardized BDM connector, defined by Freescale. The BDM connector is a 26 way Insulation Displacement Connector (IDC) keyed box header (2.54mm male) that mates with IDC sockets mounted on a ribbon cable.





## 6.1.1 Pinout

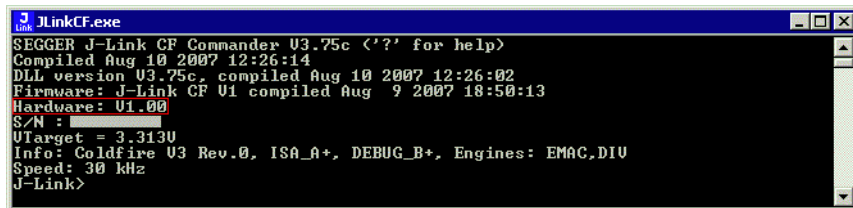
PIN	SIGNAL	TYPE	Description
1	5V	-	Target supply
2	BKPT	Output	Breakpoint
3	GND	-	Ground
4	DSCLK	Output	Development Serial Clock
5	GND	-	Ground
6	NC	-	Developer reserved
7	RESET	I/O	Reset
8	DSI	Output	Development Serial Input
9	+5V	-	Voltage supply
10	DSO	Input	Development Serial Output
11	GND	-	Ground
12	PST3	Input	Processor Status
13	PST2	Input	Processor Status
14	PST1	Input	Processor Status
15	PST0	Input	Processor Status
16	DDATA3	Input	Debug Data
17	DDATA2	Input	Debug Data
18	DDATA1	Input	Debug Data
19	DDATA0	Input	Debug Data
20	GND	-	Ground
21	NC	-	Motorola reserved
22	NC	-	Motorola reserved
23	GND	-	Ground
24	CLK_CPU	Input	
25	VCC_CPU	-	
26	TEA	Output	

**Table 6.1: Pinout of the 26-pin BDM connector**

## 6.2 How to determine the hardware version

To determine the hardware version of your J-Link, the first step should be to look at the label at the bottom side of the unit. J-Links have the hardware version printed on the back label.

If this is not the case with your J-Link, start **JLinkCF.exe**. As part of the initial message, the hardware version is displayed.

A screenshot of a Windows command window titled "JLinkCF.exe". The window has a black background with white text. The text displays the following information: "SEGGER J-Link CF Commander V3.75c '<?>' for help", "Compiled Aug 10 2007 12:26:14", "DLL version V3.75c, compiled Aug 10 2007 12:26:02", "Firmware: J-Link CF V1 compiled Aug 9 2007 18:50:13", "Hardware: V1.00" (where "V1.00" is highlighted with a red box), "S/N : ", "VTarget = 3.3130", "Info: Coldfire V3 Rev.0, ISA\_A+, DEBUG\_B+, Engines: EMAC,DIU", "Speed: 30 kHz", and "J-Link>".

```
JLinkCF.exe
SEGGER J-Link CF Commander V3.75c '<?>' for help
Compiled Aug 10 2007 12:26:14
DLL version V3.75c, compiled Aug 10 2007 12:26:02
Firmware: J-Link CF V1 compiled Aug 9 2007 18:50:13
Hardware: V1.00
S/N : 
VTarget = 3.3130
Info: Coldfire V3 Rev.0, ISA_A+, DEBUG_B+, Engines: EMAC,DIU
Speed: 30 kHz
J-Link>
```

# Chapter 7

## Background information

---

This chapter provides background information about BDM and ColdFire®.

## 7.1 BDM

The ColdFire® Family supports a modified version of the background debug mode (BDM) functionality found on Motorola's CPU32 family of parts. BDM implements a low-level system debugger in the microprocessor hardware. Communication with the development system is handled via a dedicated, high-speed serial command interface. Unless noted otherwise, the BDM functionality provided by ColdFire® is a proper subset of the CPU32 functionality.

## 7.2 The ColdFire® core

The ColdFire® core combines the architectural simplicity of conventional 32-bit RISC with a memory-saving, variable-length instruction set. In the FlexCore program, high-volume manufacturers can create their own integrated microprocessor containing a core processor (such as the ColdFire® 2/2M) and their own proprietary technology. A FlexCore integrated processor allows significant reductions in component count, power consumption, board space, and cost—resulting in higher system reliability and performance.

### 7.2.1 Processor modes

The ColdFire® architecture supports two processor modes.

Processor mode		Description
User	usr	Normal program execution mode; user programming model is selected.
Supervisor	svc	Supervisor mode; only system control software is intended to use the supervisor programming model. Supervisor programming model is selected.

**Table 7.1: ColdFire® processor modes**

### 7.2.2 Registers

#### CPU core

These registers are accessible in user- and supervisor mode.

Register	Width [bits]	Explanation
D0	32	Data register 0
D1	32	Data register 1
D2	32	Data register 2
D3	32	Data register 3
D4	32	Data register 4
D5	32	Data register 5
D6	32	Data register 6
D7	32	Data register 7
A0	32	Address register 0
A1	32	Address register 1
A2	32	Address register 2
A3	32	Address register 3
A4	32	Address register 4
A5	32	Address register 5
A6	32	Address register 6
A7	32	User stack pointer
PC	8	Program counter
CCR	8	Condition code register (the lower 8-bits of the SR)

**Table 7.2: CPU core registers**

## FPU

Register	Width [bits]	Explanation
FP0	64	Floating-point data register 0
FP1	64	Floating-point data register 1
FP2	64	Floating-point data register 2
FP3	64	Floating-point data register 3
FP4	64	Floating-point data register 4
FP5	64	Floating-point data register 5
FP6	64	Floating-point data register 6
FP7	64	Floating-point data register 7
FPCR	32	Floating-point control register
FPSR	32	Floating-point status register
FPIAR	32	Floating-point instruction address register

**Table 7.3: FPU registers**

## MAC/EMAC

Register	Width [bits]	Explanation
MACSR	32	MAC status register
ACC0	32	MAC accumulator 0
ACC1	32	MAC accumulator 1 (EMAC only)
ACC2	32	MAC accumulator 2 (EMAC only)
ACC3	32	MAC accumulator 3 (EMAC only)
ACCext01	32	ACC0 and ACC1 extensions
ACCext23	32	ACC2 and ACC3 extensions
MASK	32	MAC mask register

**Table 7.4: MAC/EMAC registers**

## Supervisor

All supervisor registers are only accessible when the CPU is in supervisor mode.

Register	Width [bits]	Explanation
SR	16	Status register (the lower 8-bits are the CCR)
OTHER_A7	32	Supervisor A7 stack pointer
VBR	32	Vector base register
CACR	32	Cache control register
ASID	32	Address space ID register
ACR0	32	Access control register 0 (data)
ACR1	32	Access control register 1 (data)
ACR2	32	Access control register 2 (instruction)
ACR3	32	Access control register 3 (instruction)
MMUBAR	32	MMU base address register

**Table 7.5: Supervisor registers**

## 7.2.3 Breakpoints and watchpoints

### Breakpoints

A breakpoint stops the core when a selected instruction is executed. It is then possible to examine the contents of both memory and variables.

### Watchpoints (called data breakpoints)

A watchpoint stops the core if a selected memory location is accessed. For a watchpoint (WP), the following properties can be specified:

- Address (including address mask)
- Type of access (R, R/W, W)
- Data (including data mask)

### Software / hardware breakpoints

Hardware breakpoints are real breakpoints. Hardware breakpoints can be set in any type of memory (RAM, ROM, flash) and also work with self-modifying code. Unfortunately, there is only a limited number of hardware breakpoints available. When debugging a program located in RAM, another option is to use software breakpoints. With software breakpoints, the instruction in memory is modified. This does not work when debugging programs located in ROM or flash, but has one huge advantage: the number of software breakpoints is not limited.

The debug module provides a number of hardware resources to support various hardware breakpoint functions. Specifically, three types of breakpoints are supported: **PC** with mask, operand address range, and data with mask. These three basic breakpoints can be configured into one- or two-level triggers with the exact trigger response also programmable. The debug module programming model is accessible from either the external development system using the serial interface or from the processor's supervisor programming model using the **WDEBUG** instruction.

## 7.3 Flash programming

J-Link comes with a DLL, which allows - amongst other functionalities - reading and writing RAM, CPU registers, starting and stopping the CPU, and setting breakpoints/watchpoints. The standard DLL does not have API functions for flash programming. However, the functionality offered can be used to program the flash. In that case, a flashloader is required.

### 7.3.1 How does flash programming via J-Link work ?

Flash programming via J-Link requires extra code. This extra code typically downloads a program into the RAM of the target system, which is able to erase and program the flash. This program is called RAM code and "knows" how to program the flash; it contains an implementation of the flash programming algorithm for the particular flash. Different flash memory devices have different programming algorithms; the programming algorithm also depends on other things such as endianness of the target system and organization of the flash memory (for example 1 \* 8 bits, 1 \* 16 bits, 2 \* 16 bits or 32 bits). The RAM code requires data to be programmed into the flash memory.

### 7.3.2 Data download to RAM

Data (or part of it) is downloaded to another part of the RAM of the target system. The program counter (PC) is then set to the start address of the RAM code, the CPU is started executes the RAM code. The RAM code, which contains the programming algorithm for the flash memory device, copies data into the flash. The CPU is stopped after this. This process might have to be repeated until the entire data is programmed into the flash.

### 7.3.3 Available options for flash programming

There are different solutions available to program internal or external flashes connected to ColdFire® cores using J-Link. The different solutions have different fields of application, but of course also some overlap.

#### 7.3.3.1 Flash loader of compiler / debugger vendor such as IAR

A lot of debuggers (some of them integrated into an IDE) come with their own flash loaders. The flash loaders can of course be used if they match your flash configuration, which is something that needs to be checked with the vendor of the debugger.



## 7.4 J-Link firmware

The heart of J-Link is a microcontroller. The firmware is the software executed by the microcontroller inside of the J-Link. The J-Link firmware sometimes needs to be updated. This firmware update is performed automatically if necessary by the **JLinkCF.dll**.

### 7.4.1 Firmware update

Every time you connect to J-Link, **JLinkCF.dll** checks if its embedded firmware is newer than the one used in J-Link. The DLL will then update the firmware automatically. This process takes less than 3 seconds and does not require a reboot.

It is recommended that you always use the latest version of **JLinkCF.dll**.

```

C:\JLinkCF_V375h\JLinkCF.exe
SEGGER J-Link CF Commander V3.75a <'?' for help>
Compiled Sep 10 2007 10:45:17
Updating firmware: J-Link CF V1 compiled Sep 06 2007 09:32:25
Replacing firmware: J-Link CF V1 compiled Sep 06 2007 09:32:25
... Firmware update successful. CRC=9F8C
Waiting for new firmware to boot
DLL version V3.75h, compiled Sep 6 2007 16:15:06
Firmware: J-Link CF V1 compiled Sep 06 2007 09:32:25
Hardware: V1.00
S/N :
UTarget = 3.326V
Info: Coldfire V2 Rev.0, ISA_A+, DEBUG_B+, Engines: DIU,MAC
Speed: 30 kHz
J-Link>

```

In the screenshot:

- The red box identifies the new firmware
- The green box identifies the old firmware which has been replaced

### 7.4.2 Invalidating the firmware

Downdating J-Link is not performed automatically through an old **JLinkCF.dll**. J-Link will continue using its current, newer firmware when using older versions of the **JLinkCF.dll**.

**Note:** Downdating J-Link is not recommended, you do it at your own risk.

**Note:** Note also that the firmware embedded in older versions of **JLinkCF.dll** might not execute properly with newer hardware versions.

To downdate J-Link, you need to invalidate the current J-Link firmware, using the command **exec InvalidateFW**.

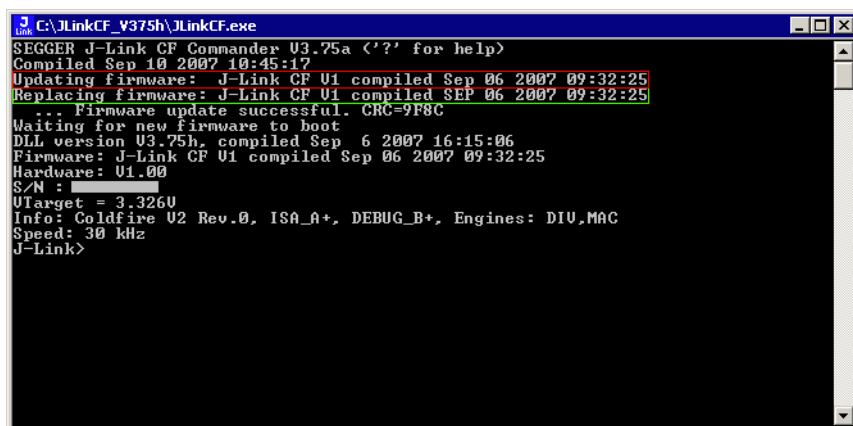
```

C:\JLinkCF_V375h\JLinkCF.exe
SEGGER J-Link CF Commander V3.75a <'?' for help>
Compiled Sep 10 2007 10:45:17
DLL version V3.75h, compiled Sep 6 2007 16:15:06
Firmware: J-Link CF V1 compiled Sep 06 2007 09:32:25
Hardware: V1.00
S/N :
UTarget = 3.326V
Info: Coldfire V2 Rev.0, ISA_A+, DEBUG_B+, Engines: DIU,MAC
Speed: 30 kHz
J-Link>

```

In the screenshot, the red box contains information about the formerly used J-Link firmware version.

Use an application (for example **JLinkCF.exe**) which uses the desired version of **JLinkCF.dll**. This automatically replaces the invalidated firmware with its embedded firmware.



```
C:\JLinkCF_V375h\JLinkCF.exe
SEGGER J-Link CF Commander V3.75a <'?' for help>
Compiled Sep 10 2007 10:45:17
Updating firmware: J-Link CF V1 compiled Sep 06 2007 09:32:25
Replacing firmware: J-Link CF V1 compiled SEP 06 2007 09:32:25
... Firmware update successful. CRC=9F8C
Waiting for new firmware to boot
DLL version V3.75h, compiled Sep 6 2007 16:15:06
Firmware: J-Link CF V1 compiled Sep 06 2007 09:32:25
Hardware: V1.00
S/N : 
VTarget = 3.326V
Info: Coldfire V2 Rev.0, ISA_A+, DEBUG_B+, Engines: DIU,MAC
Speed: 30 kHz
J-Link>
```

In the screenshot:

- The red box identifies the new firmware
- The green box identifies the old firmware which has been replaced

# Chapter 8

## Support and FAQs

---

This chapter contains troubleshooting tips together with solutions for common problems which might occur when using J-Link. There are several steps you can take before contacting support. Performing these steps can solve many problems and often eliminates the need for assistance. This chapter also contains a collection of frequently asked questions (FAQs) with answers.

## 8.1 Troubleshooting

### 8.1.1 General procedure

If you experience problems with J-Link, you should follow the steps below to solve these problems:

1. Close all running applications on your host system.
2. Disconnect the J-Link device from USB.
3. Disable power supply on the target.
4. Re-connect J-Link with the host system (attach USB cable).
5. Enable the power supply on the target.
6. Try your target application again. If the problem remains, continue the following procedure.
7. Close all running applications on your host system again.
8. Disconnect the J-Link device from USB.
9. Disable power supply on the target.
10. Re-connect J-Link with the host system (attach the USB cable).
11. Enable the power supply on the target.
12. Start `JLinkCF.exe`.
13. If `JLinkCF.exe` displays the J-Link serial number, the J-Link is working properly and cannot be the cause of the problem.
14. If the problem persists and you own an original product (not an OEM version), see section *Contacting support* on page 53.

### 8.1.2 Typical problem scenarios

#### J-Link LED is off

**Meaning:**

The USB connection does not work.

**Remedy:**

Check the USB connection. Try to re-initialize J-Link by disconnecting and reconnecting it. Make sure that the connectors are firmly attached. Check the cable connections on your J-Link and the host computer. If this does not solve the problem, check if your cable is defect. If the USB cable is Ok, try a different host computer.

#### J-Link LED is flashing at a high frequency

**Meaning:**

J-Link could not be enumerated by the USB controller.

**Most likely reasons:**

- a.) Another program is already using J-Link.
- b.) The J-Link USB driver does not work correctly.

**Remedy:**

- a.) Close all running applications and try to reinitialize J-Link by disconnecting and reconnecting it.
- b.) If the LED blinks permanently, check the correct installation of the J-Link USB driver. Deinstall and reinstall the driver as described in chapter *Setup* on page 11.

#### J-Link does not connect to the target

**Most likely reasons:**

- a.) The BDM cable is defect.
- b.) The target hardware is defect.

**Remedy:**

Follow the steps described in section 8.1.1.

## 8.2 Contacting support

Before contacting support, make sure you tried to solve your problem by following the steps outlined in section *General procedure* on page 52. You may also try your J-Link with another PC and if possible with another target system to see if it works there. If the device functions correctly, the USB setup on the original machine or your target hardware is the source of the problem, not J-Link.

If you need to contact support, send the following information to [support@segger.com](mailto:support@segger.com):

- A detailed description of the problem
- J-Link serial number
- Output of **JLinkCF.exe** if available
- Your findings of the signal analysis
- Information about your target hardware (processor, board, etc.).

J-Link is sold directly by SEGGER or as OEM-product by other vendors. We can support only official SEGGER products.

## 8.3 Frequently Asked Questions

Q: Which CPUs are supported?

A: J-Link ColdFire® can be used with any ColdFire® core.

Q: Can I access individual BDM registers via J-Link?

A: Yes, you can access all individual BDM registers via J-Link.

Q: I want to write my own application and use J-Link. Is this possible?

A: Yes. We even supply a template project and documentation.

Q: Can J-Link read back the status of the connection pins?

A: Yes, the status of all pins can be read. This includes the outputs of the J-Link as well as the supply voltage and can be useful to detect hardware problems on the target system.

Q: J-Link is quite inexpensive. What is the advantage of some more expensive ColdFire® probes?

A: Some of the more expensive ColdFire® probes offered by other manufacturers support higher download speeds or an ethernet interface. The basic functionality is the same. Some high-end probes also support Trace.

Q: Does J-Link support Trace?

A: No.

# Chapter 9

## Glossary

---

This chapter describes important terms used throughout this manual.

**Application Program Interface**

A specification of a set of procedures, functions, data structures, and constants that are used to interface two or more software components together.

**Big-endian**

Memory organization where the least significant byte of a word is at a higher address than the most significant byte. See Little-endian.

**Cache cleaning**

The process of writing dirty data in a cache to main memory.

**Coprocessor**

An additional processor that is used for certain operations, for example, for floating-point math calculations, signal processing, or memory management.

**Dirty data**

When referring to a processor data cache, data that has been written to the cache but has not been written to main memory is referred to as dirty data. Only write-back caches can have dirty data because a write-through cache writes data to the cache and to main memory simultaneously. See also cache cleaning.

**Dynamic Linked Library (DLL)**

A collection of programs, any of which can be called when needed by an executing program. A small program that helps a larger program communicate with a device such as a printer or keyboard is often packaged as a DLL.

**Halfword**

A 16-bit unit of information. Contents are taken as being an **unsigned integer** unless otherwise stated.

**Host**

A computer which provides data and other services to another computer. Especially, a computer providing debugging services to a target being debugged.

**ICache**

Instruction cache.

**ID**

Identifier.

**IEEE 1149.1**

The IEEE Standard which defines TAP. Commonly (but incorrectly) referred to as JTAG.

**Image**

An executable file that has been loaded onto a processor for execution.

**Little-endian**

Memory organization where the least significant byte of a word is at a lower address than the most significant byte. See also Big-endian.

**Memory coherency**

A memory is coherent if the value read by a data read or instruction fetch is the value that was most recently written to that location. Obtaining Memory coherency is difficult when there are multiple possible physical locations that are involved, such as a system that has main memory, a write buffer, and a cache.



## **Memory management unit (MMU)**

Hardware that controls caches and access permissions to blocks of memory, and translates virtual to physical addresses.

## **Memory Protection Unit (MPU)**

Hardware that controls access permissions to blocks of memory. Unlike an MMU, an MPU does not translate virtual addresses to physical addresses.

## **RESET**

Abbreviation of System Reset. The electronic signal which causes the target system other than the TAP controller to be reset. This signal is also known as "nSRST" "nSYSRST", "nRST", or "nRESET" in some other manuals. See also nTRST.

## **Open collector**

A signal that may be actively driven LOW by one or more drivers, and is otherwise passively pulled HIGH. Also known as a "wired AND" signal.

## **Processor Core**

The part of a microprocessor that reads instructions from memory and executes them, including the instruction fetch unit, arithmetic and logic unit, and the register bank. It excludes optional coprocessors, caches, and the memory management unit.

## **Program Status Register (PSR)**

Contains some information about the current program and some information about the current processor state. Often, therefore, also referred to as Processor Status Register.

Also referred to as Current PSR (CPSR), to emphasize the distinction to the Saved PSR (SPSR). The SPSR holds the value of PSR when the current function was called, and which will be restored when control is returned.

## **Remapping**

Changing the address of physical memory or devices after the application has started executing. This is typically done to make RAM replace ROM once the initialization has been done.

## **RTOS**

Real Time Operating System.

## **Semihosting**

A mechanism whereby the target communicates I/O requests made in the application code to the host system, rather than attempting to support the I/O itself.

## **Target**

The actual processor (real silicon or simulated) on which the application program is running.

## **Transistor-transistor logic (TTL)**

A type of logic design in which two bipolar transistors drive the logic output to one or zero. LSI and VLSI logic often used TTL with HIGH logic level approaching +5V and LOW approaching 0V.

## **Watchpoint**

A location within the image that will be monitored and that will cause execution to stop when it changes.

**Word**

A 32-bit unit of information. Contents are taken as being an unsigned integer unless otherwise stated.

# Chapter 10

## Literature and references

---

This chapter lists documents, which we think may be useful to gain deeper understanding of technical details.

Reference	Title	Comments
[BDM]	ColdFire® CF4e Core User's Manual, V4ECFUM/D Rev. 0	Describes the ColdFire® V4 core and the BDM

**Table 10.1: Literature and references**

# Index

---

<b>A</b>		Memory management unit (MMU) .....	57
Application Program Interface .....	56	Memory Protection Unit (MPU) .....	57
<b>B</b>		<b>O</b>	
BDM .....	44	Open collector .....	57
BDM Connector .....	40	<b>P</b>	
Big-endian .....	56	Pinout .....	41
<b>C</b>		Processor Core .....	57
Cache cleaning .....	56	Program Status Register (PSR) .....	57
Coprocessor .....	56	<b>R</b>	
<b>D</b>		Remapping .....	57
Dirty data .....	56	RESET .....	57
Dynamic Linked Library (DLL) .....	56	RTOS .....	57
<b>H</b>		<b>S</b>	
Halfword .....	56	Semihosting .....	57
Host .....	56	Support .....	51, 55
<b>I</b>		<b>T</b>	
ICache .....	56	Target .....	57
ID .....	56	The Coldfire Core .....	45
IEEE 1149.1 .....	56	Processor modes .....	45
Image .....	56	Registers .....	45
<b>J</b>		Transistor-transistor logic (TTL) .....	57
J-Link		<b>U</b>	
Developer Pack DLL .....	21	Using J-Link with different debuggers ....	31
Features .....	8	<b>W</b>	
Specifications .....	9	Watchpoint .....	57
Supported chips .....	26	Word .....	58
J-Link Commander .....	21		
<b>L</b>			
Little-endian .....	56		
<b>M</b>			
Memory coherency .....	56		

