Open Source BDM-JM60 Users Guide

Revised: 16 November

How to Contact Us

Corporate Headquarters	Freescale Semiconductor, Inc. 6501 William Cannon Drive West Austin, Texas 78735 U.S.A.
World Wide Web	`http://www.freescale.com/codewarrior`
Technical Support	`http://www.freescale.com/support`

Table of Contents

1

Introduction to Open Source BDM

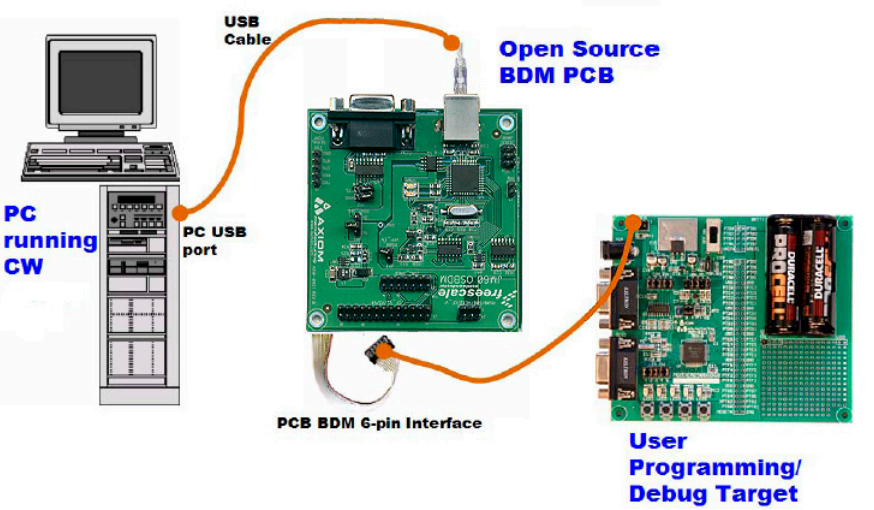
This document describes an Open Source programming and debugging development tool designed to work with Freescale HCS08, RS08, Coldfire V1,V2, V3 and V4, and DSC56800E microcontrollers.

This chapter contains the following sections:

About the Open Source BDM Interface

There is no support for Open Source BDM from Freescale, the Open Source BDM is provided with all required source code for both hardware and software components. Because it is open source, the source code can be used and/or modified from its original design free of charge. Figure 1.1 provides a pictorial overview of the typical connections required for programming and debugging using the Open Source BDM interface. A PC connects to the Open Source BDM PCB, in turn the PCB is connected to a Programming/ Debug target. In this example, the GB60 demonstration board is being programmed and debugged.

Figure 1.1 Debugging with OSBDM Interface



Open Source BDM (OSBDM), with its hardware and software components, provides a transparent connection between a computer running CodeWarrior Development Studio for MCU version to a Freescale HCS08, RS08, or Coldfire V1, V2, V3 and V4 microcontroller using BKGD pin. With a connection to the BKGD pin, the Open Source BDM enables debugger and other software tools to communicate with the microcontroller including downloading your code into the microcontroller's on-chip flash. Programming and debugger functionality is made possible by the microcontroller's Background Debug Controller (BDC) and its on-chip In-Circuit Emulation (ICE) system, if applicable.

History

- TBDML - Turbo BDM Light — It is a low cost, open source debugging interface compatible with the CodeWarrior environment for the HCS12 of MCU family. It is based on the MC908JB08 MCU and updated to the MC908JB16 device.
- TBLCF - Turbo BDM Light Coldfire — It provides interface for the open source BDM project and also to the Coldfire V2, V3, and V4 microprocessors and microcontrollers and is based on the MC908JB16 MCU.
- OSBDM - Open Source BDM — It provides interface to 9S08, 9RS08, and Coldfire V1 type of target devices and also to the MC908JB16 MCU.
- OSBDM-JM60 - 9S08JM60 Based OSBDM — It includes interfaces and firmware applied to all the targets supported by the previous open source BDMs. It includes new features and JTAG operations. It provides interface to 9S08, 9RS08, Coldfire

V1, V2, V3 and V4 target devices. It also includes JTAG connections for DSP56800E.

Open Source BDM Overview

The OSBDM-JM60 provides a USB 2.0 compliant host interface and three BDM to target interface types. With the supporting modular software projects, BDM may be generated for any of the supported target interfaces. Additional review of the software for the interface type applied is necessary if any changes are made to the defined hardware interfaces. The software drivers associated with each interface will be identified.

JM60 Controller Application

The Open Source BDM (OSBDM) is based on the MC9S08JM60 microcontroller with embedded USB port. All communication and control for BDM operation is performed by the JM60. The USB port provides the host communication and power source for the JM60 and BDM internal circuits. Peripheral devices and JM60 I/O ports provide the interface to target device, control and communication. JM60 internal flash memory contains all operating firmware for the BDM application. The JM60 firmware requires support by host PC USB drivers and software.

Open Source BDM Block Diagram

To better understand the implementation of the Open Source BDM, Figure 1.2 delineates the Open Source BDM solution into its most basic components. The diagram illustrates part of the Open Source BDM IP resides on the PC host for the debugging software and some resides on the Open Source BDM PCB.

At the time of release, the Open Source BDM is supported by the release of CodeWarrior Development Studio for Microcontrollers. The CodeWarrior provides both a software development IDE and a debugger.

Figure 1.2 Block Diagram

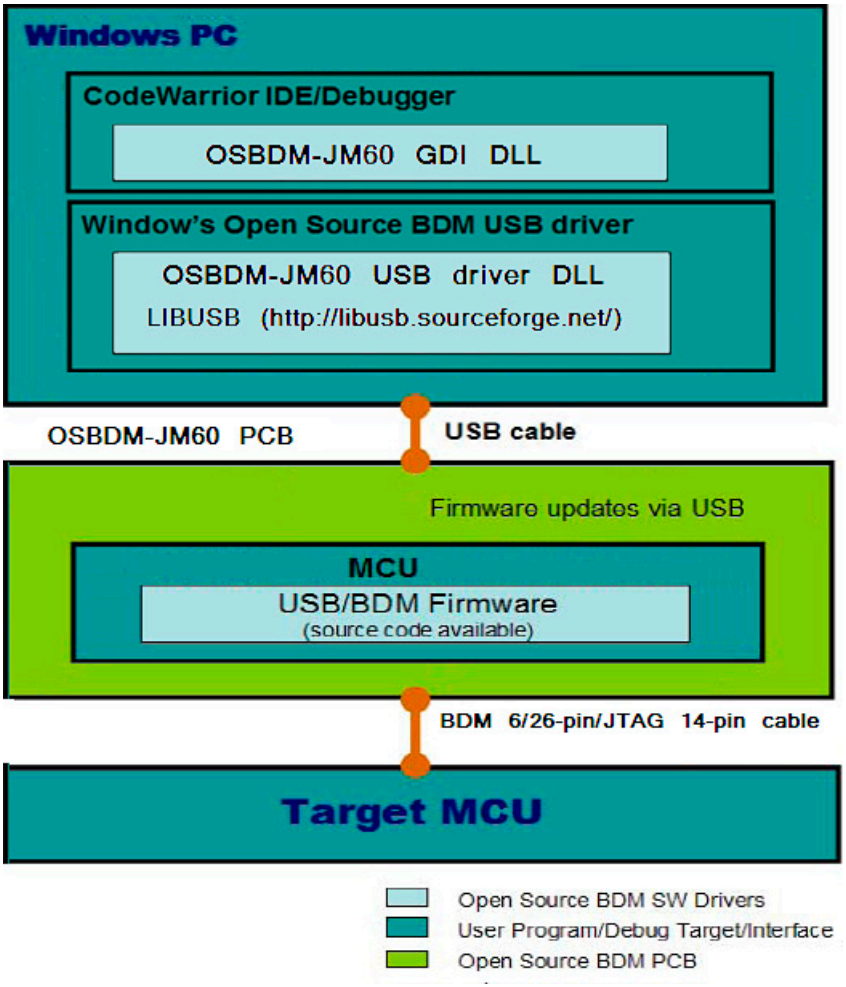


Figure 1.2 illustrates five primary components, four components are software (shown in light blue color) and one component is hardware. All the software components are intended to be used as binaries. The source code of the JM60 firmware and the Open Source BDM interface DLL is explained in Table 1.1.

OSBDM-JM60 Features

The open source BDM is implemented with the following features:

- USB 2.0 compliant communication port with standard B or mini B type connector
- The target power is determined by the MCU when OSBDM is included on a target board. Target power option of 5V or 3.3V output. Power limited to USB requirements. Adjustable voltage output optional. Embedded OSBDM does not have the target power option
- Coldfire V2, V3, and V4 interface with port options
- BGND interface with options to support 9S08, 9RS08, and Coldfire V1 targets
- JTAG interface to support DSC targets and JTAG option operation on Coldfire V2, V3, V4 targets
- +12Vpp generation provided for 9RS08 target
- Status and Target Power Indicators
- Boot load option to update firmware using USB connection
- Debug port for the JM60 BDM controller
- Low cost design, easy to embed on target boards requiring only one interface type
- Optional USB serial port application to pass target serial data to host Program Counter (PC) with USB connection

Open Source BDM Package

Table 1.1 Open Source BDM IP Components

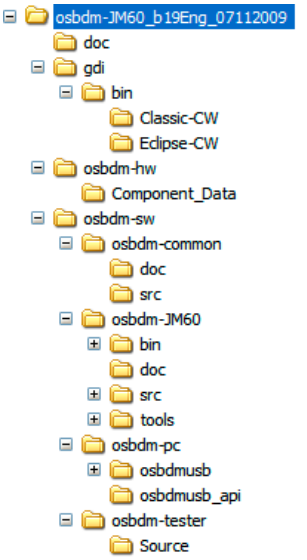
Component	Description/ Interfaces	Available/Comments
Open Source BDM GDI driver	File - OpenSourceBDM_gdi.dll Software interfacing with the CW MCU debugger GDI interface, providing function call to the Open Source BDM USB driver	This software is available only as object code in the form of a DLL
Open Source BDM USB driver	File - OpenSourceBDM.dll Software interfacing with the Open Source BDM GDI driver, providing function calls to the Windows USB drivers. This file is also required for the PC when it detects the Open Source BDM PCB as a new USB device on the PC USB port	This software is available as both object (DLL) and source code. The DLL is provided in an Open Source BDM USB install package. Point the PC Hardware Wizard to the USB install package when the PC detected the Open Source BDM PCB
Windows USB drivers	File - libusb.lib Software used by the Open Source BDM USB driver it interfaces to PC USB ports	This USB library is compiled with the Open Source BDM USB driver and is also part of the Open Source BDM USB install package. The source code for libusb can be found at *http:// libusb.sourceforge.net/*

Table 1.1 Open Source BDM IP Components

Component	Description/ Interfaces	Available/Comments
JM60 USB/BDM firmware	File - OpenSourceBDM.s19 Software running on the JM60 that receives commands via USB from the PC and converts them into commands as defined by the BDC. These commands are serial outputted - “bit -banged” - by the JM60 using port pins to drive the BKGD pin on the user’s target	This file is provided as a S-recorded and need to be programmed JM60 on-chip flash
Open Source BDM Printed Circuit Board (PCB) hardware	PCB: OpenSourceBDM Pod ver 1.0, this hardware contains the JM60 and circuitry, clock, and power to provide the interface to the program or debug target	All schematics and gerber files for the Open Source BDM PCB is provided along with a Bill Of Materials (BOM). Also included is a complete HW description which enables you to build the interface

This section describes the content of the Open Source BDM package. The package is distributed in zip file and includes development folders. The unzipped directory structure of the Open Source BDM package is shown in Figure 1.3.

Figure 1.3 Unzipped Open Source BDM Package



Support and Licensing

Open Source BDM is not supported by Freescale; it is open source. Any bugs, enhancements,or support questions should be addressed through the Open Source BDM forum. Open Source BDM has been thoroughly tested, but there is no guarantee about error-free operation. All the source files, except GDI DLL, is available to anyone under the GNU LESSER GENERAL PUBLIC LICENSE. For more information on license, refer *COPYING_LGPL.txt* file located at the root directory of the OSBDM-JM60 package. The Open Source BDM is a derivative project of the TBMDL project.

2

OSBDM-JM60 Hardware Reference Design

This chapter explains about OSBDM-JM60 hardware and its connections. It explains only standalone OSBDM board and not the circuits that are embedded onto an MCU board.

NOTE The testing criteria should be met while porting the Open Source BDM Reference Design (JM60) to an embedded OSBDM-JM60. For information on testing criteria, refer OSBDM-JM60 Firmware Testing Criteria appendix.

This chapter contains the following sections:

- OSBDM Options and Connections
- Hardware Schematic Design
- Signal Chart
- OSBDM Hardware Application

OSBDM Options and Connections

The OSBDM reference design provides jumper options for feature and target settings. Jumper shunts are applied to enable a feature or option when installed on 2 pins. Option is disabled when the jumper shunt is placed on one pin or it is removed. In some cases, shunt is placed between two pins of a 3 pin header i.e. between pins 1 and 2 or between pins 2 and 3. This allows selection between two different functions. Ribbon cables are placed on the respective interface type pin headers to connect to the various supported target development ports. Only one target type is connected at a time.

Figure 2.1 Option and Connector Diagram

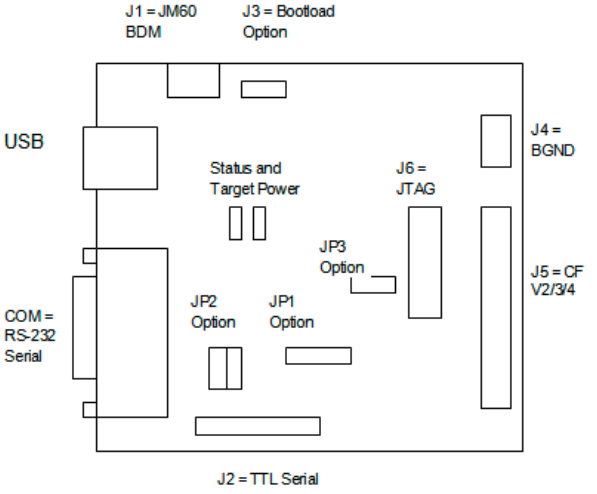
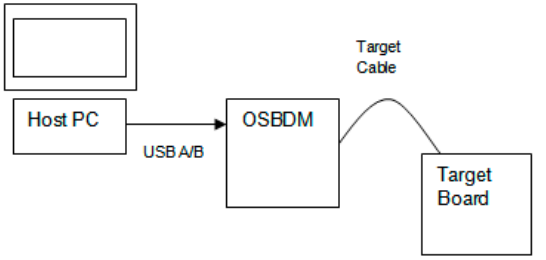


Figure 2.2 Connection Diagram

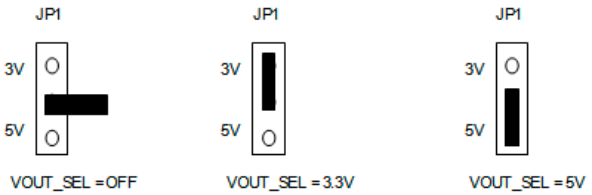


Options

JP1 — VOUT Select

JP1 allows the OSBDM to provide target power and apply the voltage level that is selected. The default option is shown in the idle position or no target power is applied. You must verify the target board's voltage level requirement, current requirement, and regulator type or other options prior to applying this option. You should avoid powering a target board voltage regulator's output or hardware failure could occur to the OSBDM, target board, and/or host PC USB port.

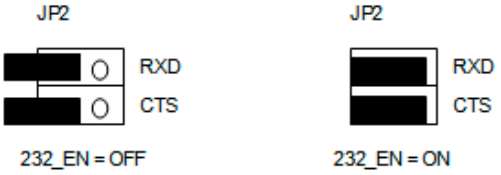
Figure 2.3 JP1 Option



JP2 — 232 Enable

JP2 provides options to connect the OSBDM COM RS232 level serial port to the JM60 SCI0 serial port signals. J2 connection should not be applied, if this option is enabled. JM60 application supports the serial port operation. For information on input and output ports, refer to the Signal Chart section.

Figure 2.4 JP2 Option

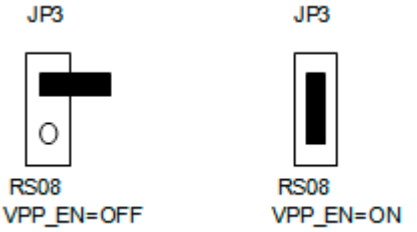


NOTE The CTS signal operation is optional for flow control. The serial port of the target board may be connected to the OSBDM COM port and the COM port is connected to host PC with the USB cable.

JP3 — RS08 VPP Enable

JP3 provides Flash programming voltage (VPP) support for RS08 type targets. This option should not be enabled with any other target type. Otherwise, the OSBDM board and/or the target board may be damaged. When enabled, the OSBDM will provide the VPP voltage on the J4 pin 4 RESET* signal under the application control. For information on power supply, refer to the VPP Generator and Control section.

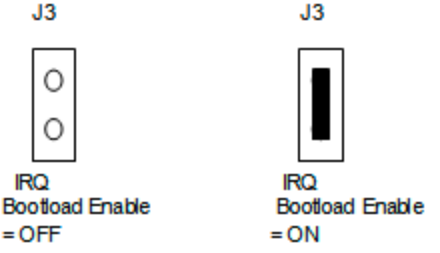
Figure 2.5 JP3 Option



J3 — IRQ/Bootload Mode Enable

J3 provides an option to enable the JM60 USB bootloader application. This application should reside in the JM60 flash firmware. Bootloader application is included in the OSBDM projects by default. J3 must be opted prior to connecting the OSBDM USB connector to the host PC. USB drivers must be installed on the host PC for proper operation. Refer *Application Note AN3561* for more information.

Figure 2.6 J3 Option



NOTE An option shunt is not provided for use on J3. When using the Boot load function, use the shunt from JP1, JP2, or JP3 and then return the shunt to the original position after boot load is complete.

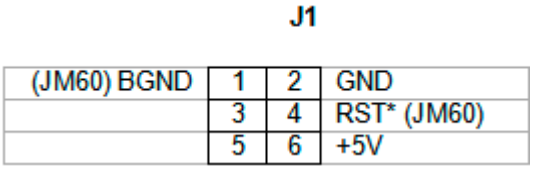
Connectors

J1 — Debug JM60

J1 provides the development connection for the OSBDM JM60 controller. This connection is applied with an external BDM to support OSBDM development or JM60 flash programming. The USB connector or external BDM must power the OSBDM board

during development or flashing operations. It is possible to apply another external JM60 OSBDM to flash or debug the JM60 as a target on the OSBDM.

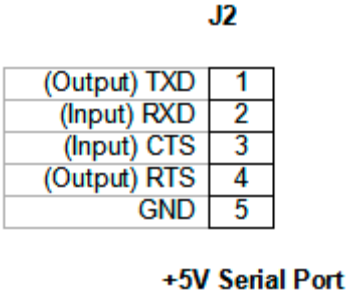
Figure 2.7 J1 Connector



J2 — Serial Port (TTL)

J2 provides access to the JM60 SCI0 serial port signals and flow control I/O ports. The connection provides direct I/O port access to the JM60 and operates at 5V levels under application control. JP2 options should be OFF to apply this port connector. For information on input and output ports, refer to the Signal Chart section.

Figure 2.8 J2 Connector



J4 — Target BGND Connector

J4 provides a 6 position ribbon cable connection to a BGND type target devices S08, RS08, and CF_V1.

Before applying power, you must verify pin 1 alignment between the OSBDM and target BDM connector. Review JP1 and JP3 options. OSBDM JM60 application firmware must support the connected target type. Length of the ribbon cable should not exceed 8 inches.

Figure 2.9 J4 Connector

J4

TBGND	1	2	GND
	3	4	TRG_RST*
	5	6	+V_TRG

RS08/S08/HC12/CF_V1

J5 — Target Coldfire V2/3/4 Connector

J5 provides the standard 26 pin ribbon cable connection for the Coldfire V2/3/4 type targets. Before applying power, you must verify pin 1 alignment between the OSBDM and target development connector. Review JP1 and JP3 options. OSBDM JM60 application firmware must support the connected target type. Length of the ribbon cable should not exceed 6 inches. This connection also supports the JTAG operation supported by Coldfire. Review target board options for BDM or JTAG operation support prior to application.

Figure 2.10 J5 Connector

J5

NC	1	2	BKPT* / TMS
GND	3	4	DSCLK / TRST
GND	5	6	TCLK
TRG_RST*	7	8	DSI / TDI
+V_TRG	9	10	DSO / TDO
GND	11	12	PST7 / DE*
PST6	13	14	PST5
PST4	15	16	NC
NC	17	18	NC
NC	19	20	GND
NC	21	22	NC
GND	23	24	NC
NC	25	26	TA*

CF_V2/V3/V4

J6 — Target DSC JTAG Connector

J6 provides the standard 14 pin ribbon cable connection for the DSC JTAG or OnCE type targets. Before applying power, you must verify pin 1 alignment between the OSBDM and target development connector. Review JP1 and JP3 options. JM60 OSBDM application firmware must support the connected target type. Length of the ribbon cable should not exceed 8 inches.

Figure 2.11 J6 Connector

J6

DSI / TDI	1	2	GND
DSO / TDO	3	4	GND
TCLK	5	6	GND
NC	7	8	KEY
TRG_RST*	9	10	BKPT* / TMS
VDD	11	12	NC
PST7 / DE*	13	14	DSCLK / TRST

JTAG / OnCE

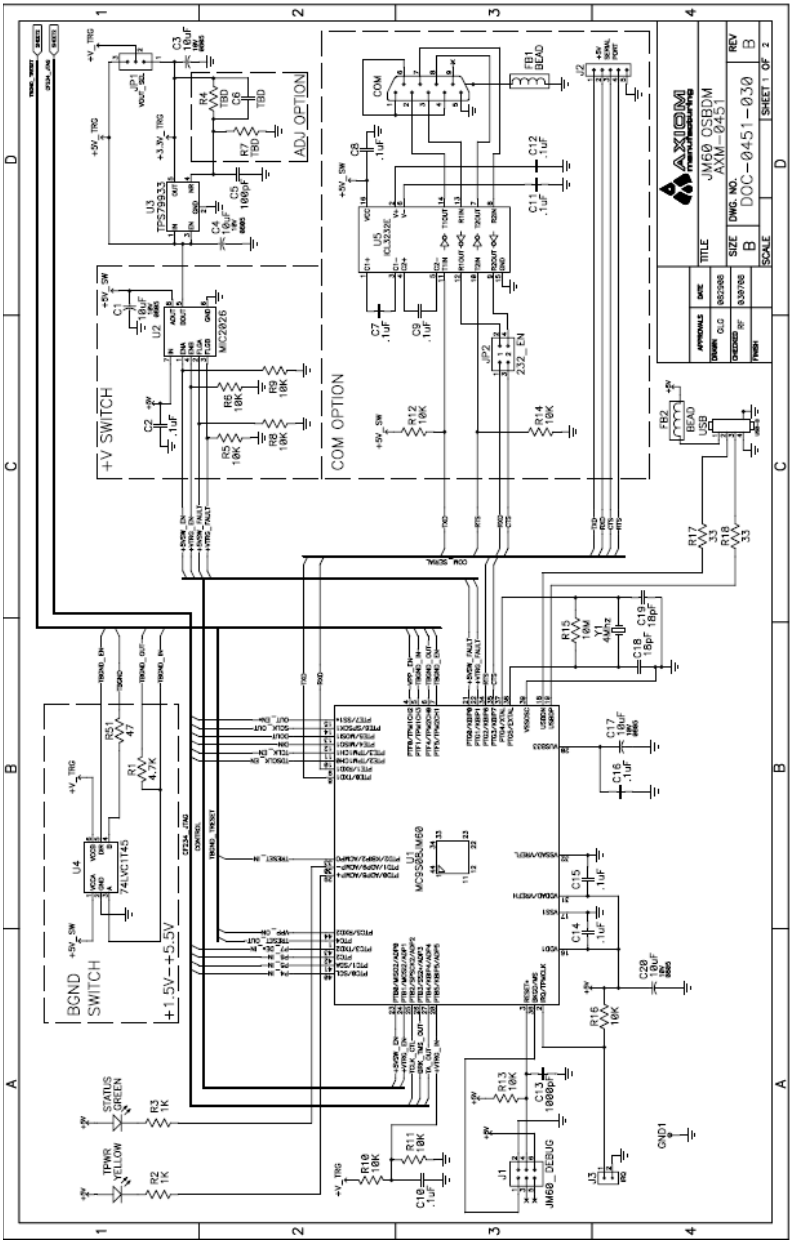
Power and Status Indicators

A green Status and yellow Target Power (TPWR) indicators are provided. Both indicators operate under the application control and are not associated with any other hardware operation. Status indicator indicates OSBDM communication or operation with the host PC. TPWR indicator indicates target voltage is detected (target is powered) and the target should be operational.

Hardware Schematic Design

Figure 2.12 shows the schematic diagram of OSBDM-JM60.

Figure 2.12 Schematic Design



Signal Chart

Figure 2.13 shows the signal chart of the OSBDM-JM60.

Figure 2.13 Signal Chart

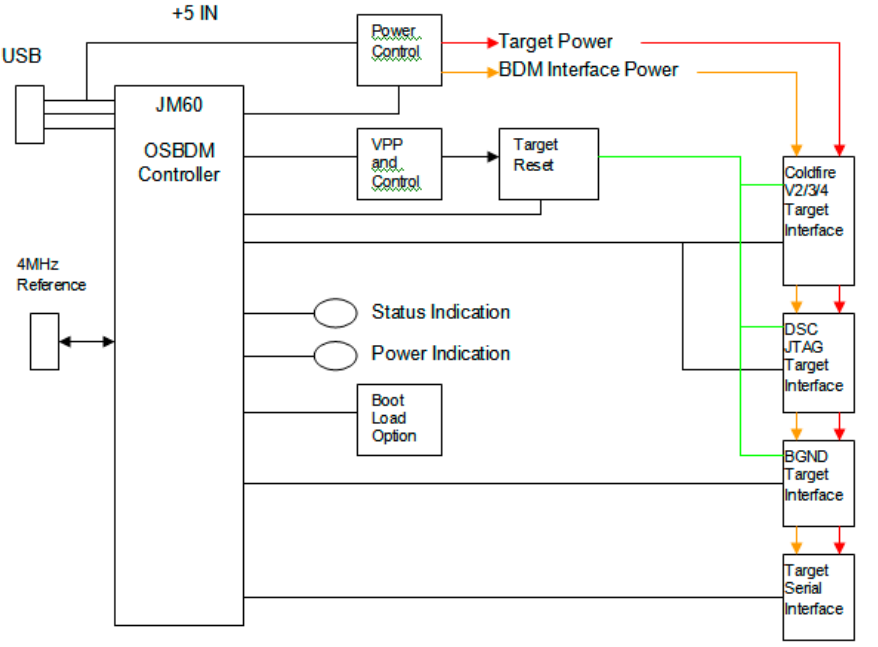
Pin	MCU Signal	PCB Signal	Pin Function	Dir	Active LVL	Notes	Comments
36	BKGD/MS						Dedicated JM60 debug signal
2	IRQ/TPMCLK	Bootload Enable	GPIO	I	low	1	Update firmware Mode if low at Reset / power on
23	PTB0/MISO2/ADP0	+5VSW_EN	GPIO	O	high	2	Enable BDM interface power
24	PTB1/MISI2/ADP1	+VTRG_EN	GPIO	O	high	2	Enable Target power out
25	PTB2/SPSCK2/ADP2	TCLK_CTL	GPIO	O	low	3	Enable CF / JTAG TCLK output
26	PTB3/SS2*\ADP3	BRK_TMS_OUT	GPIO	O	>>	3	CF BRKPT* active low, JTAG TMS active high
27	PTB4/KBIP4/ADP4	TA_OUT	GPIO	O	high	3	CF TA* output active high
28	PTB5/KBIP5/ADP5	+VTRG_IN	ADC	I	.8-2.5v		Target power detection input, ADC measure
40	PTC0/SCL	P4_IN	GPIO	I	low		CF PST4 input, halted detection
41	PTC1/SDA	P5_IN	GPIO	I	low		CF PST5 input, halted detection
42	PTC2	P6_IN	GPIO	I	low		CF PST 6 input, halted detection
43	PTC3/TXD2	P7_DE*_IN	GPIO	I	>>		CF PST7 input, low halt detect, JTAG DE active high
1	PTC4	TRESET_OUT	GPIO	O	high		All target Reset* signal
44	PTC5/RXD2	VPP_ON	GPIO	O	high		"+12V VPP supply enable
29	PTD0/ADP8/ACMP+	PWR_LED	GPIO	O	low		Target power Status indicator
30	PTD1/ADP9/ACMP-	STATUS_LED	GPIO	O	low		BDM status indicator
33	PTD2/KBIP2/ACMPO	TRESET_IN	GPIO	I	low		S08/S12/V1 Target reset detect, add to CF port pin 1
8	PTE0/TXD1	TXD	COM	O	>>		COM port option TXD
9	PTE1/RXD1	RXD	COM	I	>>		COM port option RXD
10	PTE2/TPM1CH0	TDSCLK_EN	GPIO	I/O	>>	4	CF DSCLK enable or JTAG TRST* signal
11	PTE3/TPM1CH1	TCLK_EN	GPIO	I/O	low		JTAG TCLK, tri state to clock, active to stop clock
12	PTE4/MISO1	DIN	SPI	I	>>		CF / JTAG data in
13	PTE5/MOSI1	DOUT	SPI	O	>>		CF / JTAG dtata out
14	PTE6/SPSCK1	SCLK_OUT	SPI	O	>>		CF / JTAG serial clock out
15	PTE7/SS1*	OUT_EN*	SPI	O	low		Enable Target interface outputs
4	PTF0/TPM1CH2	VPP_EN	GPIO	O	high		VPP enable to RS08 target RESET signal
5	PTF1/TPM1CH3	TBGND_IN	GPIO	I	>>		S08/S12/V1 Target BGND input
6	PTF2/TPM2CH0	TBGND_OUT	TPM	O	>>		S08/S12/V1 Target BGND output
7	PTF5/TPM2CH1	TBGND_EN	TPM	O	>>		S08/S12/V1 Target BGND direction
21	PTG0/KBIP0	+5VSW_FAULT	GPIO	I	high		BDM I/O power fault, disable on fault
22	PTG1/KBIP1	+VTRG_FAULT	GPIO	I	high		Target Power fault, disable on fault
34	PTG2/KBIP6	RTS	GPIO	O	low		COM port, transmit enable or RTS signal
35	PTG3/KBIP7	CTS	GPIO	I	low		COM port, CTS input
37	PTG4/XTAL	XTAL	XTAL				4Mhz
38	PTG5/EXTAL	XTAL	XTAL				4Mhz
3	RESET*	RESET*		I			JM60 input signal
18	USBDN	USBDN		IO			USB
19	USBNP	USBDP		IO			USB
16	VDD	+5V	PWR				
31	VDDAD/VREFH	+5V	PWR				
17	VSS	GND	PWR				
32	VSSAD/VREFL	GND	PWR				
39	VSSOSC	GND	PWR				
20	VUSB33	Internal regulator BP	PWR				Decouple

Notes:
1. Bootloader to be applied to update JM60 firmware via USB connection
2. Enables BDM Target output circuit power or Target power from USB source. Enable after USB power setting negotiated. Disable on USB Suspend detection.
3. This ouput to the target is enabled by the OUT_EN* signal.
4. TDSCLKEN provides the CF bit 17 clock signal and tristates or the JTAG TRST* output active low.

OSBDM Hardware Application

The OSBDM provide interfaces and controls supervised by the JM60 to operate different target devices and USB interface. Attempt has been made to provide a featured BDM connection with minimal cost. All development features of target device types are not supported to maintain a lower cost. The OSBDM-JM60 reference design provides multiple target device interfaces that may be applied separately in your designs or applications. The following sections describe the circuits and type of device supported. OSBDM-JM60 schematic diagram, materials list, and PCB layout files should be referred for application support. Figure 2.14 shows the block diagram of OSBDM-JM60.

Figure 2.14 OSBDM-JM60 Hardware Block Diagram



OSBDM and JM60 Internal Support Circuits

The JM60 requires hardware support to operate the application. This section describes the common circuits for OSBDM operations.

Reference Clock

The JM60 applies a 4.00MHz crystal reference (Y1) for USB communication timing on I/O ports PTG4/XTAL and PTG5/EXTAL. The crystal applied is a standard parallel type with 50ppm maximum frequency tolerance. The value of load capacitors should be applied for the crystal selected in other designs.

JM60 initialization must configure the Multipurpose Clock Generator (MCG) to enable and apply the external crystal reference. MCG configuration sets the JM60 core and USB clocks to 48Mhz and the bus clock to 24Mhz for BDM operation. Time base for JM60 operation is 24Mhz or 41.67ns minimum time period.

Source code modules:

`Main.c : MCG_Init function for initialization details and register settings.`

USB Connection

The JM60 USB connection is the primary host communication interface for application operation. The connection also powers the JM60 and OSBDM by default. The standard Freescale recommended hardware connection is applied with a USB type B connector. JM60 has dedicated I/O ports for the USB signal connections: USBDN and USBDP.

USB system operation is complex. For complete details of USB bus application and operation, refer to the *USB.org* specifications and documents. For basic details of JM60 USB application with a host PC, refer to the *JM60 USB GUI Application Note AN3561* The JM60_OSBDM USB firmware application is a software port of the drivers.

Software Modules:

`USB_DRV Folder content`

`USB_User_API.C`

Power Control

The OSBDM provides a dual power switch (U2/MIC2026) with current limit to power the OSBDM internally and the target, if enabled. Each switch provides an over current fault signal indication to the JM60 for the use of an application. At initial USB connection, the OSBDM is limited in power consumption by USB specifications and the power switches are OFF. After the USB connection with the host PC, additional power is applied and the JM60 will power the OSBDM internal circuits with switch A. During OSBDM application operation, the JM60 will power the target with power switch B. For information on the I/O ports applied for the switch and fault signals, refer to the Signal Chart section.

Power switch A provides +5V_SW signal to power U4 (BGND transceiver), U5 (RS232 COM serial port transceiver), U8 (Coldfire V2/3/4 and JTAG input interface), and the VPP voltage generator. The VPP generator is OFF but power is applied by default.

Power switch B provides +V_TRG output voltage to power the target using JP1 option. A 3.3V regulator (U3) is provided on the switch B output for target application. The regulator is a TPS79933 for a standard 3.3V output. Option is possible for other voltage outputs or an adjustable version of the regulator. Adjustable version is supported with resistor locations R4 and R7 to set the voltage level. +V_TRG range of 1.8V to 4.5V when JP1 is set to the 3.3V position is possible with component options.

NOTE The OSBDM current should not exceed 500ma from the USB connection. This includes OSBDM power and target power if applied by JP1. Damage to the host PC USB port may occur if the current exceeds 500ma. Use caution while powering target boards from the OSBDM (JP1 ON). Verify that the target board voltage regulation output will not be powered again or conflict with the supplied OSBDM power. A power conflict could cause current limitations to be easily exceeded for a short duration of time.

OSBDM Target Support Circuits

See the Power Control section for details on power control of the OSBDM and target power.

+V Target Input

The JM60 ADC input channel 5 is applied to monitor the +V_TRG voltage level. Input level to the ADC is set to 50% of the +V_TRG level by resistors R10 and R11. Application should verify that the target is powered before attempting communication.

VPP Generator and Control

VPP generation of +12V for RS08 flash programming support is provided by U6 and associated circuits. The JM60 enables the VPP generator with the VPP_ON signal. Application of the VPP voltage to the target is enabled by the VPP_EN signal. VPP application to the target is tightly integrated into the flashing commands for the RS08 target type. VPP voltage is applied to the TRG_RST* signal by JP3 option.

Target Reset I/O

Target reset input and output is applied with discrete transistors Q1, Q3, Q5, and associated components. This circuit is used to provide a compliant interface to all of the supported target types. TRG_RST* signal output and input level is set by the +V_TRG voltage level. The +12V VPP voltage appears on this signal, if applied.

TRG_RST* output is enabled by the TRESET_OUT signal from the JM60. Target reset detection is provided on the TRESET_IN signal to the JM60. The input provides the same logic level as the TRG_RST* signal level.

NOTE All the target types do not accept or provide a reset signal from this interface. In this case, an OSBDM command to the target is applied to produce a reset condition.

OSBDM Development Port Interfaces

This section describes the individual development ports. Note that only one port type is applied to a target at any time.

BGND Interface

The BGND interface provides the standard 6 pin connection for the single wire BGND signal type development port. Target types that apply this development port are the 9S08, RS08, and Coldfire V1. +V_TRG, TBGND, TRG_RST*, and Ground are applied by this port type.

The BGND interface applies U4 as the signal transceiver. U4 is a 74LVC1T45 logic gate with voltage level shifting features. Operation on the target side (+V_TRG) is 1.8V to 5.5V. The JM60 side is always +5V from the +5V_SW signal. JM60 signals TBGND_EN, TBGND_IN, and TBGND_OUT provide the communication and control for this interface. All these signals are associated with JM60 timer channels for precise timing capability to a 41.67ns time step period. For more information on the input and output ports, refer to the Signal Chart section.

TBGND_ EN provides directional control for transmit or receive operations. The signal is logic high for transmit output and logic low to receive input. JM60 timer 2 channel 1 provides the primary signal direction control during the communication with the target. The idle condition is low so that the interface is not driven unless the communication is intended. During the communication, the direction is fixed to output the command to the target. During the reception, the signal is timed in edge aligned PWM mode to provide the BGND start pulse prior to the target reply input.

TBGND_OUT provides the transmit signal output from the JM60 to the target. Timer 2 channel 0 controls this signal in edge aligned PWM mode. For data transmission, the timer channel will output an active low signal with a time period that represents a logic one bit value or logic 0 bit value. In receive mode, the timer channel will provide a low output for the start bit on the BGND signal and then provide timing internally for the reply signal input time window.

TBGND_IN provides receive signal input from the target to the JM60. Timer 1 channel 3 is applied to measure the input signal duration in capture mode (25Mhz BDC clock

maximum). This operation provides the timing to determine a logic 1 or 0 bit value input from the target. RS08 type targets apply a lower speed communication technique that inputs the JM60 port value (sample mode) instead of using the timer capture. This is due to the RS08 will not provide a stable input signal after the start bit generation and creates false timer capture edges. Other undefined target types may exhibit the same issue and may apply sample mode, if required (10MHz BDC clock maximum).

NOTE The TBGND_OUT and TBGND_IN signals are connected with resistor R1. R1 provides isolation between the 2 timer channels.

Coldfire V2/3/4 Interface

The Coldfire V2/3/4 interface provides the standard 26 pin interface for this target type. The interface is limited and does not support trace operations that require real time input of the development port PST and/or DAT signals. This interface applies these signals to the JM60.

Table 2.1 shows the signals and their operation notes of Coldfire interface.

Table 2.1 Signals and Operation Notes

JM60 Signal	J5 Port Signal	Operation Notes
TRESET_OUT	TRG_RST*	Reset Target output
TCLK_CTL	TCLK	JTAG mode TCLK signal output enable
SCLK_OUT	DSCLK_TRST*	Serial clock signal or TRST control
TCLK_EN	TCLK	Signal level override
OUT_EN	DSCLK_TRST, DSI_TDI, BKPT*_TMS	Interface output enable for associated signals
TDSCLK_EN	DSCLK_TRST*	Signal level override
DOUT	DSI_TDI	Serial Data output
BRK_TMS_OUT	BKPT*_TMS	BDM Break signal or JTAG TMS signal
TA_OUT	TA*	Transfer acknowledge to target to clear Bus errors
DIN	DSO_TDO	Serial data input

Table 2.1 Signals and Operation Notes

JM60 Signal	J5 Port Signal	Operation Notes
P4_IN	PST4	Status bit 4
P5_IN	PST5	Status bit 5
P6_IN	PST6	Status bit 6
P7_IN	PST7_DE*	Status bit 7 for BDM, Debug event for JTAG

Also applied is the +V_TRG power signal. This interface operates from 2.5 to 5.5V target voltage levels.

Interface operation applies the JM60 Serial Peripheral Interface (SPI) for communication. Mode change from BDM (default) to JTAG is possible under the application control of the signals applied. For more information on JTAG mode of opeartion, refer to the DSC JTAG OnCE Interface section.

The SCLK_OUT, DOUT, and DIN are the primary communication signals and are operated by the JM60 SPI. Maximum bit rate is 20% of the CPU bus speed of the Coldfire target device. The bit rate is fixed in the reference design to 125Khz for 500Khz minimum target clock. Due to the required shift between output and input bit clock edges, the first and last bit of the 17 bit transfer size is applied manually.

BRK_TMS_OUT is applied as the breakpoint or halt signal output in BDM mode. Assertion of this signal will stop target execution on the next instruction boundary.

TA_OUT will assert the Coldfire target device Transfer Acknowledge (TA*) or Transfer Error Acknowledge (TEA*) signal, if connected. This signal may terminate a target bus cycle created in error or due to corrupt execution and prevent the need for a reset sequence.

P4_IN to P7_IN provide target core status signals. The OSBDM will apply these signals to detect that the core has halted execution. The reference design interface inverts the logic level of these signals from the target to the JM60. Target halts indication would be a logic high on all signals, input to the JM60 is a logic low on all signals to indicate a halt. Reduced pin package targets may provide a single ALLPST type output to indicate a run or halt condition. In this case, all PST_IN signals should be connected to the ALLPST signal on the target development port and operation is transparent to the OSBDM. Embedded OSBDM designs may apply only the PST7/P7_IN connection to the ALLPST target signal and adjust the application appropriately.

DSC JTAG OnCE Interface

The DSC JTAG interface provides the standard 14 pin interface. Signal includes the JTAG mode of operation for the Coldlfire V2/3/4 targets. This interface applies these signals to the JM60.

Table 2.2 explain JM60 signals and operation notes of JTAG OnCE interface.

Table 2.2 Signals and Operation Notes of JTAG OnCE Interface

JM60 Signal	J6 Port Signal	Operation Notes
TRESET_OUT	TRG_RST*	Reset Target output
TCLK_CTL	TCLK	TCLK signal output enable
SCLK_OUT	DSCLK_TRST*	TRST control
TCLK_EN	TCLK	Signal level override
OUT_EN	DSCLK_TRST, DSI_TDI, BKPT*_TMS	Interface output enable for associated signals
TDSCLK_EN	DSCLK_TRST*	Signal level override
DOUT	DSI_TDI	Serial Data output
BRK_TMS_OUT	BKPT*_TMS	TMS signal
DIN	DSO_TDO	Serial data input
P7_IN	PST7_DE*	Debug event input

Also applied is the +V_TRG power signal. This interface operates from 2.5 to 5.5V target voltage levels.

Interface operation applies JM60 I/O port manipulation. Serial Peripheral Interface (SPI) for communication may be possible in the future but is difficult to apply due to the random clock cycle to TMS signal level change requirements.

The TCLK_EN, DOUT, and DIN provide the TCLK, TDI, and TDO JTAG signals. These signals are the primary communication signals for moving data to and from the target. Data bit count is variable on the transfers and may require sequence level changes of the TMS signal for correct operation of the interface with the target.

BRK_TMS_OUT is applied as the TMS signal output. This signal is sampled with clock changes to the target and must be modified during the data transfers for correct operation of the interface.

TDSCLK_EN is applied as the TRST* signal output. This signal is a test reset and does not reset the entire device.

TRESET_OUT is applied as TRG_RST* signal. All DSC targets provide a RESET* input pin that resets the DSC device, unless it is reconfigured as a GPIO.

P7_IN is applied as the DE* or Debug Event input signal. This signal is logic level inverted from the target. All the target do not support this signal output so it is treated as target specific.

3

JM60 Firmware Application

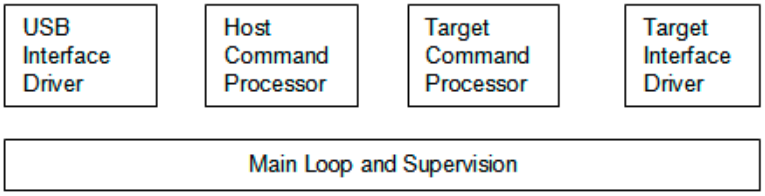
The OSBDM-JM60 firmware application supports various target types, such as, HCS08, Coldfire V1, V2, V3, V4, and DSP56800E.

This chapter contains the following sections:

Firmware Block Diagram

The JM60 BDM firmware application currently supports one target device type at a time. JM60 software for the BDM application is a CodeWarrior project that provides build target settings and software source modules corresponding to the BDM target device type. Correct target device type BDM interface hardware is required to support the firmware. Each CodeWarrior build target setting selects the associated software modules required to be included in the JM60 firmware to operate the BDM interface for the target device type.

Figure 3.1 Firmware Block Diagram



A common USB interface driver and host command processor are applied for all target types. Target command processor and target interface driver are selected to support the specific target device. Common utility source files are also included.

Firmware Source Table

Table 3.1 describes the association of various targets with the source files.

Table 3.1 Description of Source Files

Firmware Source File	Target Association	Function Summary
Start08.c	JM60	Reset Vector and start-up
main.c	All Targets	Application start, main loop and management
cmd_processing.c	All Targets	Host command processor
USB_User_API.c	All Targets	USB driver I/O, USB driver is interrupt driven
MCU.c	JM60	JM60 initialization and hardware support
bdm_9S08.c	9S08	HCS08/9S08 target specific command processor
bdm_cfv1.c	Coldfire V1	Coldfire V1/Flexis target specific command processor
bdm_bgnd_driver.c	S08	Common BGND type interface driver for higher speed targets (Edge capture type, 25Mhz maximum target BDC clock)
bdm_cf.c	Coldfire V2/3/4	Coldfire V2/3/4 Target Specific Command Processor and Interface Driver
jtag_dsc.c	DSC JTAG	DSC JTAG Target Specific Command Processor and Interface Driver
timer.c	All Targets	Timer and timing support

Table 3.1 Description of Source Files

Firmware Source File	Target Association	Function Summary
util.c	All Targets	Utility functions and support
serial_io.c	All Targets	Target Serial channel/ COM port support

NOTE `Main.c` determines if the JM60 USB bootloader application or primary application starts. Each source file listed has one or more associated header files (.h extension) to provide the necessary definitions. Select the Header tab of the CodeWarrior project to edit these files.

OSBDM-JM60 Specifications

Table 3.2 describes the OSBDM-JM60 specification.

Table 3.2 OSBDM-JM60 Specifications

Specifications	
Input Voltage	+5V from USB connection or JM60 BDM port, 40ma internal power
Output Voltages	+5V to target, 400ma maximum +3.3V to target, 250ma maximum +12V VPP programming voltage, 20ma maximum
BGND Operation	200KHz to 25MHz target BDC clock, 1.8 to 5V signaling
Coldfire V2/3/4	500KHz minimum target clock, 2.5 to 5V signaling
JTAG	2.5 to 5V signaling

NOTE BGND includes 9S08, and Coldfire V1 target type devices.

4

OSBDM PC Software Application

This chapter describes the Open Source BDM solution PC software components.

This chapter contains the following sections:

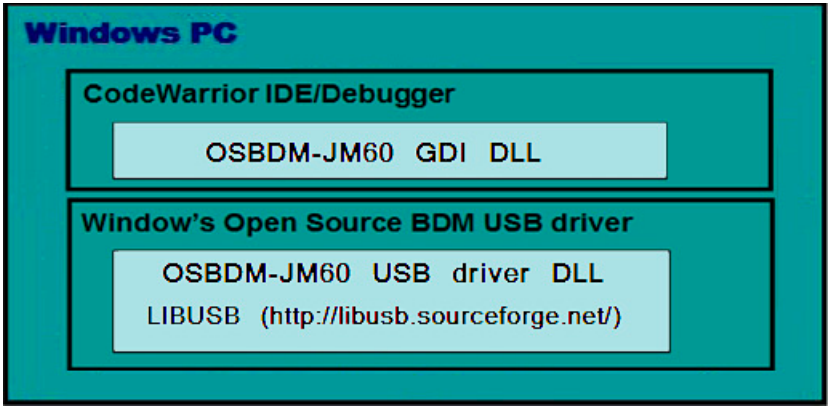
Overview

This section describes the Open Source BDM solution PC software components. These components include:

- Open Source BDM Generic Debug Instrument (GDI) DLL plug-in for the CodeWarrior debugger
- Open Source BDM interface DLL
- USB driver (libusb.lib)

Figure 4.1 illustrates these software components and their interfaces.

Figure 4.1 OSBDM Windows PC Software and Interfaces



Open Source BDM GDI DLL

The Open Source BDM GDI DLL plug-in for the CodeWarrior debugger is only available as a binary file. No source code is provided. The GDI DLL for the CodeWarrior debugger is created partially based on information not available in the public domain. The license attached to these files does not allow disclosure of the file's source code.

The Open Source BDM takes advantage of the CodeWarrior Debugger's Generic Debug Instrument (GDI) protocol interface. For more information on GDI Open Interface Specification, see *http://www.tasking.com/resources/technologies/debuggers/gdikdi*.

Open Source BDM DLL and LIBUSB

The Open Source BDM DLL provides an interface between the Open Source BDM GDI DLL and the Open Source BDM firmware. This section describes the API of the Open Source BDM DLL including a Windows Open Source USB drivers library, LIBUSB. The source code for the Open Source BDM DLL is available.

OSBDM USB Base Driver API

Software applications interface to the OSBDM-JM60 firmware using the `osbdm_dll.h`. The header files *`commands.h`* and *`osbdm_dll.h`* contain all the definition used by these APIs that should be included in your project.

The Open Source BDM DLL functions are listed and briefly described below:

unsigned char osbdm_dll_version(void)

Returns the version number of the `osbdm_dll.h` in BCD format (major in upper nibble and minor in lower nibble).

OsbdmErrT osbdm_get_version(unsigned char *version_info)

Returns the hardware (MSB) and firmware (LSB) version of the open OSBDM-JM60 device.

unsigned char osbdm_init()

Initializes a connection to the USB driver and returns the number of OSBDM-JM60 devices found. If there is a problem communicating with the USB driver or other error then -1 is returned.

This function must be called before a device can be opened.

OsbdmErrT osbdm_open(unsigned char device_no)

Opens communication with an OSBDM-JM60 specified by `device_no`. The first device is number 0.

Returns 0 on success and non-zero on failure. This function must be called before any further communication with the device can take place.

void osbdm_close()

Closes communication with currently opened device.

OsbdmErrT osbdm_target_reset(ResetT resetmode)

Resets the MCU target using the `resetmode` value. Returns 0 on success and non-zero on failure. For more information on `ResetT` type defined in `osbdm_def.h`, refer Common Headers section.

OsbdmErrT osbdm_target_go()

Starts target code execution from current PC address. Returns 0 on success and non-zero on failure.

OsbdmErrT osbdm_target_step()

Steps over a single target instruction. Returns 0 on success and non-zero on failure.

OsbdmErrT osbdm_target_halt(void)

Brings the target into active background (debug) mode with user code execution halted. Returns 0 on success and non-zero on failure.

OsbdmErrT osbdm_get_speed(unsigned long *speed)

Returns the crystal (or external source) frequency of the target in KHz.

OsbdmErrT osbdm_set_speed(unsigned long speed)

Sets the BDM communication speed which is the crystal or external source frequency in KHz. Returns `osbdm_error_ok` on success or the error is defined in `OsbdmErrT`.

OsbdmErrT osbdm_init_hardware(void)

Initialize the OSBDM-JM60 BDM hardware and establish a connection between the OSBDM-JM60 probe. Returns `osbdm_error_ok` on success or the error is defined in `OsbdmErrT`.

OsbdmErrT osbdm_status(unsigned char *data)

Retrieve the target BDM communication status information to `data`. Returns `osbdm_error_ok` on success or the error is defined in `OsbdmErrT`.

OsbdmErrT osbdm_config(unsigned char config_type, unsigned char config_param, unsigned long param_value)

Configure the OSBDM-JM60 and target. The configuration type `config_type`, defined in `ConfigT` selects the entity to be configured. The configuration parameter `config_param`, is set to the value specified in `param_value`. Returns `osbdm_error_ok` on success or the error is defined in `OsbdmErrT`.

OsbdmErrT osbdm_write_fill(unsigned char type, unsigned char width, unsigned long addr, unsigned char *data, unsigned long count)

Fill a single value to a contiguous section of target's memory with an amount of `count`. The value to fill is written to `data` and `addr` is starting address in memory, `type` specifies the memory type for fill operation, which is defined in `OsbdmMemT` and `width` is the number of bits for the memory access size. Returns `osbdm_error_ok` on success or the error is defined in `OsbdmErrT`.

unsigned char osbdm_write_block(unsigned char type, unsigned char width, unsigned long addr, unsigned char *data, unsigned long size)

Writes a block of memory to the target.

unsigned char osbdm_read_block(unsigned char type, unsigned char width, unsigned long addr, unsigned char *data, unsigned long size)

Reads a block of memory from the target. Both `osbdm_read_block` and `osbdm_write_block` share the same parameters. Some of these parameters are also used by other memory functions.

Table 4.1 describes the parameters of read and write memory functions.

Table 4.1 Description of Parameters

Parameters	Description
type	Type of memory to be read or written: MEM_RAM - Normal memory MEM_REG - Normal register MEM_CREG - Control register MEM_DREG - Debug register MEM_P - Special Program Memory MEM_P_FLASH - Special Program Flash Memory MEM_X - Special Data Memory MEM_X_FLASH - Special Data Flash Memory
width	Number of bits to be read or written at a time (8, 16 or 32)
addr	Start address
data	Pointer to the data to be written or to hold the data read
size	Number of 8-bit data bytes to be written or read

unsigned char osbdm_write32(unsigned char type, unsigned long address, unsigned long data)

Write a single 32-bit value to target memory. Returns 0 on success and 1 on error.

unsigned char osbdm_write16(unsigned char type, unsigned long address, unsigned short data)

Write a single 16-bit value to target memory. Returns 0 on success and 1 on error.

unsigned char osbdm_write8(unsigned char type, unsigned long address, unsigned char data)

Write a single 8-bit value to target memory. Returns 0 on success and 1 on error.

unsigned long osbdm_read32(unsigned char type, unsigned long address)

Read a single 32-bit value from target memory.

unsigned short osbdm_read16(unsigned char type, unsigned long address)

Read a single 16-bit value from target memory.

unsigned char osbdm_read8(unsigned char type, unsigned long address)

Read a single 8-bit value from target memory.

unsigned long osbdm_read_bd(unsigned int address)

Read from the BDM memory area.

unsigned char osbdm_write_bd(unsigned int address, unsigned long data)

Write to the BDM memory area, returns 0 on success and non-zero on failure.

OSBDM USB Driver API

OsbdmErrT osbdmAPI_connect(CoreT core_type)

Connect to the device specified by `core_type` and collect the BDM communication status information. Returns `osbdm_error_ok` on success or the error defined in `OsbdmErrT`.

OsbdmErrT osbdmAPI_get_status(ConnectStateT *status)

Get the target BDM communication status information, which contains the target reset state, the target connection state, the OSBDM-JM60 version, and the flash state. The `ConnectStateT` is defined in a common header file, `osbdm_def.h`. Returns `osbdm_error_ok` on success or the error defined in `OsbdmErrT`.

OsbdmErrT osbdmAPI_run(void)

Start the execution of a core. Returns `osbdm_error_ok` on success or the error defined in `OsbdmErrT`.

OsbdmErrT osbdmAPI_step(void)

Step over a single target instruction. Returns `osbdm_error_ok` on success or the error defined in `OsbdmErrT`.

OsbdmErrT osbdmAPI_read_mem(unsigned char mem_space, unsigned int addr, unsigned int byte_count, SizeT access_size, unsigned char *buffer)

Read `byte_count` of data from the memory. The data is read into buffer and `addr` is the starting address in memory to begin read. Memory access attribute can be specified with `access_size` whose type is defined in `SizeT`. The memory space `mem_space`, is an optional target dependent parameter and it is defined in the target specific header. Returns `osbdm_error_ok` on success or the error defined in `OsbdmErrT`.

OsbdmErrT osbdmAPI_write_mem(unsigned char mem_space, unsigned int addr, unsigned int byte_count, SizeT access_size, unsigned char *buffer)

Write `byte_count` of data to the memory. The data is written to buffer and address is the starting address in memory to begin write. Memory access attribute can be specified

with access_size whose type is defined in SizeT. The memory space mem_space, is an optional target dependent parameter and it is defined in the target specific header. Returns osbdm_error_ok on success or the error defined in OsbdmErrT.

OsbdmErrT osbdmAPI_config(ConfigT config_type, unsigned char config_param, unsigned long param_value)

Configure the OSBDM-JM60 and target. The configuration type config_type, defined in ConfigT selects the entity to be configured. The configuration parameter config_param, is set to the value specified in param_value. Returns osbdm_error_ok on success or the error defined in OsbdmErrT.

OsbdmErrT osbdmAPI_core_mode(CoreModeT *core_mode)

Poll the execution mode of the core and returns the mode defined in CoreModeT. Returns osbdm_error_ok on success or the error defined in OsbdmErrT.

OsbdmErrT osbdmAPI_secure_mode(SecureModeT *secure_mode)

Poll the target secure mode and return the mode defined in SecureModeT. Returns osbdm_error_ok on success or the error defined in OsbdmErrT.

Target Specific Header Files

osbdm_cfv1.h

The header file contains Coldfire V1 target specific definitions that are used in the OSBDM-JM60 API. Coldfire V1 register numbers are defined in this header file.

osbdm_cfv234.h

The header file contains Coldfire V2/3/4 target specific definitions that are used in the OSBDM-JM60 API. Coldfire V2/3/4 register numbers are defined in this header file.

osbdm_s08.h

The header file contains HCS08 target specific definitions that are used in the OSBDM-JM60 driver API. HCS08 register numbers are defined in this header file.

osbdm_dsc.h

The header file contains DSC target specific definitions that are used in the OSBDM-JM60 driver API. DSC register numbers are defined in this header file.

Common Headers

osbdm_configparam.h

The header file contains the definition for the OSBDM-JM60 configuration parameters.

```
typedef enum  {
    osbdmcfg        = 0x10,   // osbdm-JM60 config
```

```
    tgtcfg_generic = 0x40,   // generic target config
    tgtcfg_specific= 0x80    // target specific config
} configType;
```

osbdm_def.h

The header file contains the definition that is used for both OSBDM-JM60 driver APIs and OSBDM-JM60 base driver APIs. It includes header files `osbdm_configparam.h` and `osbdm_memtype.h`. OsbdmErrT, ResetT, FlashStateT, CoreT, SizeT, CoreModeT, SecureModeT, TargetResetDetectT, TargetConnectStateT, ConnectStateT, OsbdmInfoT are defined in this header file.

```
typedef enum  {
osbdm_error_ok,
osbdm_error_fail,
osbdm_error_invalid_parameter,
osbdm_error_invalid_target,
osbdm_error_not_supported,
osbdm_error_usb_transport,
osbdm_error_cmd_failed,
osbdm_error_bdm_not_enabled,
osbdm_error_undefined
}OsbdmErrT;

/* type of reset mode */
typedef enum  {
eSoftReset_to_DebugMode,
eSoftReset_to_NormalMode,
eHardReset_to_DebugMode,
eHardReset_to_NormalMode,
ePowerReset_to_DebugMode,
ePowerReset_to_NormalMode
} ResetT;
typedef enum  {
eCFv234,
eCFv1,
```

```
eRS08,
eS08,
eDSC,
eCoreTypeUnknown
} CoreT;

typedef enum  {
eByte,     // 1 byte
eWord,    // 2 bytes
eLong     // 4 bytes
} SizeT;

/* type of core run state */
typedef enum  {
eDebug,
eRunning,
eWait,
eStopped,
eWaitOrStopped,
eBusy,
eCoreModeUndefined
} CoreModeT;

/* type of secure mode */
typedef enum  {
eSecured,
eUnsecured,
eSecureModeUndefined,
} SecureModeT;

/* target reset detection state */
typedef enum {
```

```
RESET_INACTIVE,
RESET_DETECTED,
RESET_UNDEFINED
} TargetResetDetectT;

/* target connection state */
typedef enum {
NO_CONNECTION = 0,
SYNC = 1,
RESERVED = 2,
MANUAL_SETUP = 3,
CONNECT_UNDEFINED
}TargetConnectStateT;

/* target BDM communication state */
typedef struct {
TargetResetDetectT    reset_state;
TargetConnectStateT   connect_state;
unsigned int          osbdmJM60_version;
FlashStateT           flash_state;
} ConnectStateT;

typedef struct {
ConnectStateT  connect_state;
CoreT          coreid;
} OsbdmInfoT;
```

osbdm_memtype.h

The header file contains the memory type definitions that are used for both OSBDM-JM60 driver API and OSBDM-JM60 base driver API.

```
typedef enum  {
MEM_RAM =0,       // Memory
MEM_REG =1,       // Register
```

```
MEM_CREG =2,      // Control Register
MEM_DREG =3,      // Debug Register
MEM_P =10,        // Program Memory
MEM_P_FLASH =11,  // Program Flash Memory
MEM_X =12,        // Data Memory
MEM_X_FLASH =13   // Data Flash Memory
} OsbdmMemT;
```

osbdm_utils.h

The header file contains utility functions that are used in both OSBDM-JM60 USB driver API and OSBDM-JM60 USB base driver API. Byte ordering utility functions is defined in this header file.

Update OSBDM-JM60 Firmware with Freescale JM60 GUI Application

The Freescale JM60 GUI application includes a GUI version of the bootloader that can be used to program the OSBDM-JM60 firmware. You can download JM60 GUI installer from Freescale website using the following link: *http://www.freescale.com/webapp/sps/download/license.jsp?colCode=JM60_GUI_INSTALLER_V1.2&location=null&fsrch=1*

When OSBDM-JM60 firmware is installed with the bootloader library (by a debugger that can program the OSBDM-JM60 through the BDM connection), the firmware can be erased and reprogrammed with the JM60 GUI tool.

For the standalone (i.e. non-embedded version) OSBDM-JM60, when the jumper J3 is set to the ON position it activates the bootloader mode on connecting the USB cable to the OSBDM-JM60 board. When the JM60 GUI tool recognizes the OSBDM-JM60 in the bootloader mode, the color of the bottom right USB symbol of the JM60 GUI changes from red to green.

The following are the steps for updating OSBDM-JM60 firmware with Freescale JM60 GUI tool:

1. Install the JM60 GUI application.

NOTE You can download the JM60 GUI application from the Freescale website: www.freescale.com/webapp/sps/download/license.jsp?colCode=JM60_GUI_INSTALLER_V1.2&location=null&fsrch=1

2. Set the J3 jumper to the ON position.

3. Connect the USB cable to the OSBDM-JM60 pod and this will launch the Found New Hardware Wizard for the Freescale JM60 Bootloader device.
4. Install the drivers.
5. Open the Freescale JM60 GUI. The USB symbol in the bottom right corner of the JM60 GUI should be green if the bootloader is activated.
6. Select the Bootloader from the JM60 Application menu.
7. Load the firmware SREC file located at: *{OSDMPackageDir}\osbdm-sw\osbdm-JM60\bin* from the JM60 USB Bootloader window.
8. Click Execute on the JM60 USB Bootloader window.
9. Do not interrupt the firmware update process until it is completed.
10. Close the Freescale JM60 GUI application.
11. Disconnect the USB cable.
12. Set the J3 jumper to the OFF position.

5

Installation and Operation of the OSBDM

This chapter contains the following sections:

Configuration of the Open Source BDM PCB

This section describes that the Open Source BDM firmware is programmed on to the JM60.

Open Source BDM PCB HCS08 Configuration

Table 5.1 details the Open Source BDM PCB jumper configuration settings for Programming and debugging targets for the HCS08. Other types of target may require other settings.

Table 5.1 OSBDM PCB HCS08 Configuration

Jumper	Description	Settings
JP1	Target Power Selection	Short 1-2: Power target externally via J1 or target is self powered Short 2-3: Power target using 5V USB power
JP3	Flash Program	Off
JP4	Flash Program	Off
JP5	Flash Erase	Off

Installing Windows Open Source BDM DLL and USB Drivers

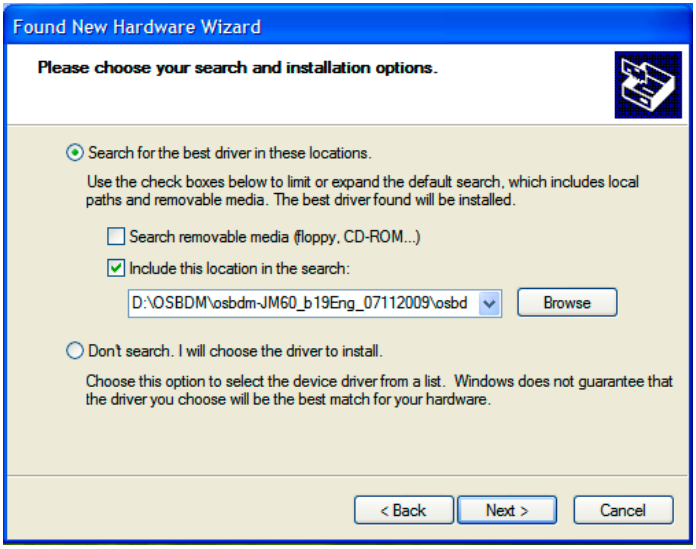
The following procedure specifies the installing of the Open Source BDM USB hardware drivers on the Windows operating system. In this procedure, it is assumed that Open Source BDM Windows USB driver package is unpacked onto the development PC, running the CodeWarrior Studio.

With the Open Source BDM PCB configured, the Open Source BDM PCB can be connected to the development PC USB port. When configured Open Source BDM PCB is connected to the PC for the first time, the Windows operating system recognizes a new USB device, the Open Source BDM PCB. Figure 5.1 illustrates that the initial connection starts the Windows driver installation and displays Windows New Hardware Wizard dialog box.

Figure 5.1 Found New Hardware Wizard

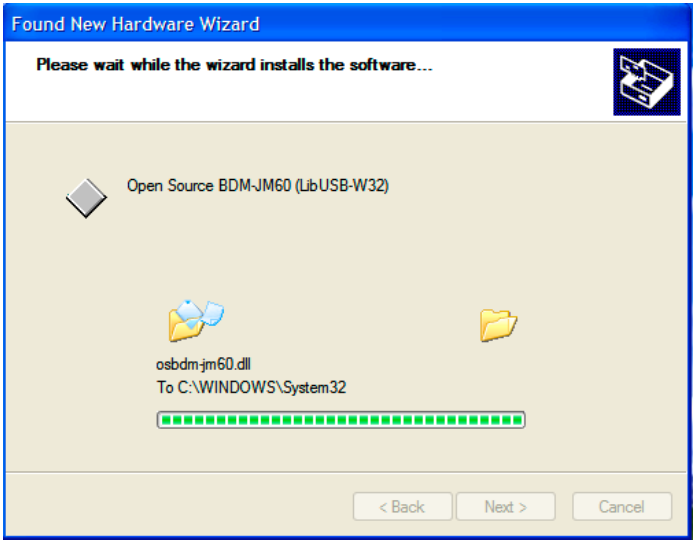
1. For this installation, select **Install from a list or specific location** option and click **Next**. The Specify Location of the Drivers dialog box is displayed as shown in Figure 5.2.

Figure 5.2 Specify Location of Drivers Dialog Box



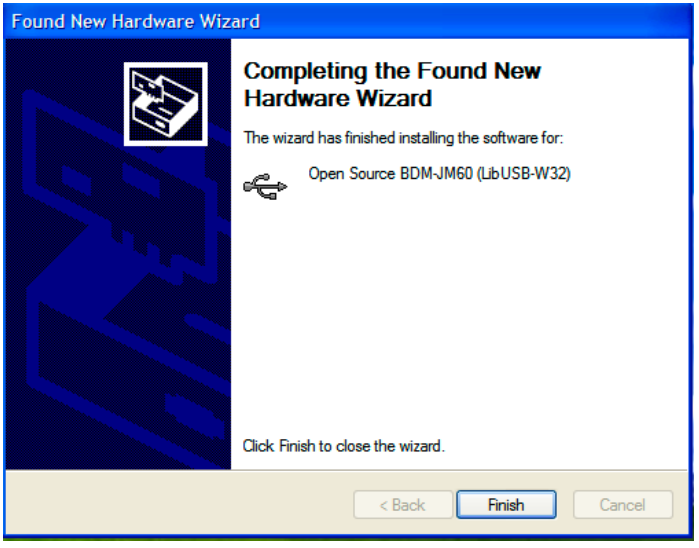
2. In the Specify Location of the Drivers dialog box, check **Include this location in the search** checkbox. Click **Browse** button to find the unzipped Open Source BDM Windows driver package. If the path specified is the correct path to the Open Source BDM windows driver package, click **Next**.
3. It will initiate the installation of the Open Source BDM USB driver and DLL file as shown in Figure 5.3.

Figure 5.3 Driver Installation



4. Once the installation procedure is completed, click **Finish** and the device will be ready to use. Because of the plug and play nature of USB, reboot of Windows is not required.

Figure 5.4 Finish Installation Open Source BDM Windows USB Driver



A

Hardware Components List

A complete Bill Of Materials is provided in this section, including all parts required for the OSBDM-JM60.

Table A.1 BDM Base W/power sw

Qty	Reference(m) BDM Base W/ power sw	Title	Detail	Vendor Part No.	Mfr	Mfr Part No.
1	U1	IC-MCU(R)	(Qfp64) MC9S08JM60	MC9S08JM60CLH-ND (Digi-Key)	Freescale	MC9S08JM60CLH
1	U2	IC-PWR SW(R)	(Soic8) Dual current limited switch	576-1059-1-ND (Digi-Key)	Micrel	MIC2026-1-YM
1	J1	Conn-USB-Thru(R)	USB-B, RA	AE9925-ND (Digi-Key)	Assman	AU-Y1007-R
1	Y1	Crystal-Smt(R)	(HC49S) 4MHz, Sht Can	XC1238CT-ND (Digi-Key)	ECS	ECS-40-20-5PX
1	U1_Debug	Conn-Pin Hdr-Thru(R)	2x3 Header	S2012E-03-ND (Digi-Key)	Sullins	PEC03DAAN

Table A.1 BDM Base W/power sw

Qty	Reference(m) BDM Base W/ power sw	Title	Detail	Vendor Part No.	Mfr	Mfr Part No.
4	C1, C2, C3, C4	Cap-Cer-Smt(R)	(0805) .1uF, X7R, 50V	311-1140-1-ND (Digi-Key)	Yaego	CC0805KRX7R9BB104
2	C5, C6	Cap-Cer-Smt(R)	(0805) 18pF,50V, npo/cog, 5%	311-1102-1-ND (Digi-Key)	Yaego	CC0805JRNP09BN180
1	C7	Cap-Cer-Smt(R)	(0805) 1000pF, X7R, 50V	311-1127-1-ND (Digi-Key)	Yaego	CC0805KRX7R9BB102
4	C8, C9, C10, C11	Cap-Cer-Smt(R)	(0805) 10uF,10V, X5R	495-3263-1-ND (Digi-Key)	Epcos	B37641C8106K062
1	Status	LED-SMT(R)	(1206) Green	516-1443-1-nd (Digi-Key)	Avago	HSMG-C150
1	T_Pwr	LED-SMT(R)	(1206) Yellow	516-1442-1-nd (Digi-Key)	Avago	HSMY-C150
1	FB1	Ind-FB-Smt(R)	(0805) 220 Ohms@ 100M, 1.5A	490-1054-1-ND (Digi-Key)	Murata	BLM21PG221SN1D
1	R1	Res-Carb-Smt(R)	(0805) 10M Ohm, 5%	RHM10MARCT-ND (Digi-Key)	Rohm	MCR10EZPJ106

Table A.1 BDM Base W/power sw

Qty	Reference(m) BDM Base W/ power sw	Title	Detail	Vendor Part No.	Mfr	Mfr Part No.
2	R2, R3	Res-Carb-Smt(R)	(0805) 33 Ohm, 1%	RHM33.0 CRCT-ND (Digi-Key)	Rohm	MCR10E ZPF33R0
2	R4, R5	Res-Carb-Smt(R)	(0805) 1K Ohm, 5%	RHM1.0K ARCT-ND (Digi-Key)	Rohm	MCR10E ZPJ102
2	R6, R7	Res-Carb-Smt(R)	(0805) 4.7K Ohm, 5%	RHM4.7K ARCT-ND (Digi-Key)	Rohm	MCR10E ZPJ472
2	R7, R8	Res-Carb-Smt(R)	(0805) 10K Ohm, 5%	RHM10K ARCT-ND (Digi-Key)	Rohm	MCR10E ZPJ103
1	PCB	Pcb(R)	JM60 BDM, 3x3", Proto pricing	AXM-0415C (ACD)	Axiom Mfg	AXM-0415C
1	USB A/B	Cable - USB A/B	USB A/B, 2.0 Compliant, 2m or 6ft	AE1493-ND (Digi-Key)	Assman	AK672/2-2-R

Table A.2 Target Reset Out/In

Qty	All Target Reset Out/In	Title	Detail	Digi-Key Part No.	Mfr	Mfr Part No.
2	Q1, Q2	Trans-Mosfet-Smt(R)	(SOT23) NPN 30V GP	568-1741-1-ND	NXP	PMBT3904
1	Q3	Trans-Mosfet-Smt(R)	(SOT23) PNP 30V GP	568-1742-1-ND	NXP	PMBT3906
1	D1	Diode - SMT, SchKy	(SOT23) 30V, 100ma	568-1610-1-ND	NXP	BAT54
2	R9, R10	Res-Carb-Smt(R)	(0805) 4.7K Ohm, 5%	RHM4.7KARCT-ND	Rohm	MCR10EZPJ472
6	R11, R12, R13, R14, R15, R16	Res-Carb-Smt(R)	(0805) 10K Ohm, 5%	RHM10KARCT-ND	Rohm	MCR10EZPJ103
1	R17	Res-Carb-Smt(R)	(0805) 47 Ohm, 5%	RHM47ARCT-ND	Rohm	MCR10EZPJ470

Table A.3 9S08/12/CF-V1 BGND I/O

Qty	9S08/12/CF-V1 BGND I/O	Title	Detail	Digi-Key Part No.	Mfr	Mfr Part No.
1	BDM-6	Conn-Pin Hdr-Thru(R)	2x3 Header	S2012E-03-ND	Sullins	PEC03DAAN
1	U3	IC-Xcvr-Smt(R)	(Sot23-6) 74LVC1T45,Sgl Bit,Dual Supply	296-16843-1-ND	TI	SN74LVC1T45DBVR
1	C12	Cap-Cer-Smt(R)	(0805) .1uF,50V	311-1140-1-ND	Yaego	CC0805KRX7R9BB104
1	R18	Res-Carb-Smt(R)	(0805) 4.7K Ohm, 5%	RHM4.7KARCT-ND	Rohm	MCR10EZPJ472
1	R19	Res-Carb-Smt(R)	(0805) 47 Ohm, 5%	RHM47ARCT-ND	Rohm	MCR10EZPJ470
1	Target cable	cable-ribbon	6 pos IDC female both end, 7in long. Note: 6 position special order, 10 postion shown)	H3AAH-1006G-ND	Assman	H3AAH-1006G

Table A.4 Target + V out Option

Qty	Target +V out Option	Title	Detail	Digi-Key Part No.	Mfr	Mfr Part No.
1	JP2 - VTRG_EN	Conn-Pin Hdr-Thru(R)	1x3 Header	S1012E-03-ND	Sullins	PEC03SAAN
1	JP-Shunt	Conn-Pin-shunt	.1 Jumper shunt	609-2462-ND	FCI	71363-102LF
1	VR2	VReg-Linear-LP-Smt(R)	(Sot23-5) 3.3V or Adj., 200ma LD	296-17783-1-ND	TI	TPS79933DDCR
1	C15 1.5 - 3.3 Vadj= Add 1x1%R, 1x1%R or RV1	Cap-Cer-Smt(R)	(0805) 10uF,10 V, X5R	495-3263-1-ND	Epcos	B37641C8106K062

Table A.5 RS08 VPP Power

Qty	RS08 VPP Power	Title	Detail	Digi-Key Part No.	Manufacturer	Manufacturer Part No.
1	JP1 - VPP_EN	Conn-Pin Hdr-Thru(R)	1x2 Header	S1012E-02-ND	Sullins	PEC02SAAN
1	JP-Shunt	Conn-Pin-shunt	.1 Jumper shunt	609-2462-ND	FCI	71363-102LF

Table A.5 RS08 VPP Power

Qty	RS08 VPP Power	Title	Detail	Digi-Key Part No.	Manufacturer	Manufacturer Part No.
1	VR1	VReg-Boost Sw-Smt(R)	(Sot23-5) Adj. Boost, 12V 50mA , W/ enable	296-12686-1-ND	TI	TPS61041DBV
1	Q4	Trans-Mosfet-Smt(R)	(SOT23) NPN 30V GP	568-1741-1-ND	NXP	PMBT3904
1	Q5	Trans-Mosfet-Smt(R)	(SOT23) PNP 30V GP	568-1742-1-ND	NXP	PMBT3906
1	D2	Diode - SMT, Schky	(SMB) 20V, 1A	SS12-E3/1GI-ND	Vishay	SS12-E3/1
1	D3	Diode - SMT, Schky	SOT23) Dual, Common cathode	568-1612-1-ND	NXP	BAT54C
1	L1	Ind-Pwr-Smt(R)	(4x4mm) 10uH, 500ma	445-3176-1-ND	TDK	VLCF4018T-100MR74-2
2	C13	Cap-Cer-Smt(R)	(0805) 1000pF, 50V	311-1127-1-ND	Yaego	CC0805KRX7R9BB102
1	C14	Cap-Cer-Smt(R)	(0805) 1uF, 16V, X5R	399-1284-1-ND	Kemet	C0805C105K4RACTU
3	R20, R21, R22	Res-Carb-Smt(R)	(0805) 4.7K Ohm, 5%	RHM4.7KARCT-ND	Rohm	MCR10EZPJ472

Table A.5 RS08 VPP Power

Qty	RS08 VPP Power	Title	Detail	Digi-Key Part No.	Manufacturer	Manufacturer Part No.
2	R23	Res-Carb-Smt(R)	(0805) 10K Ohm,5%	RHM10KARCT-ND	Rohm	MCR10EZPJ103
1	R24	Res-Carb-Smt(R)	(0805) 1K Ohm, 1%	RHM1.00KCRCT-ND	Rohm	MCR10EZPF1001
1	R25	Res-Carb-Smt(R)	(0805) 9.09K Ohm, 1%	RHM9.09KCRCT-ND	Rohm	MCR10EZPF9091
1	R26	Res-Carb-Smt(R)	(0805) 100 Ohm, 5%	RHM100ARCT-ND	Rohm	MCR10EZPJ101

Table A.6 CF-V2/V3/V4 Target

Qty	CF-V2/V3/V4	Title	Detail	Digi-Key Part No.	Mfr	Mfr Part No.
1	BDM-26	Conn-Pin Hdr-Thru(R)	2x13 Header	S2012E-13-ND	Sullins	PEC13DAAN
1	U4	IC-Logic-SMT	(SOIC14) Quad tri state buffer, 2-5V	296-21028-1-ND	TI	SN74LV125ATDR
1	U5	IC-Logic-SMT	(SOIC14) Hex Inverter, TTL in, 5V	296-1205-1-ND	TI	SN74HCT04DR

Table A.6 CF-V2/V3/V4 Target

Qty	CF-V2/ V3/V4	Title	Detail	Digi-Key Part No.	Mfr	Mfr Part No.
1	Q6	Trans-Mosfet-Smt(R)	(SOT23) NPN 30V GP	568-1741-1-ND	NXP	PMBT3904
10	R27, R28, R29, R30, R31, R32, R33, R34, R35, R36	Res-Carb-Smt(R)	(0805) 47 Ohm, 5%	RHM47ARCT-ND	Rohm	MCR10EZPJ470
3	R37, R38, R39	Res-Carb-Smt(R)	(0805) 4.7K Ohm, 5%	RHM4.7KARCT-ND	Rohm	MCR10EZPJ472
1	R40	Res-Carb-Smt(R)	(0805) 10K Ohm, 5%	RHM10KARCT-ND	Rohm	MCR10EZPJ103
1	Target cable	Cable-ribbon	26pos IDC female both end, 6 inch long	H3AAH-2606G-ND	Assman	H3AAH-2606G

Table A.7 JTAG_ONCE Target

Qty	JTAG_ OnCE	Title	Detail	Digi-Key Part No.	Mfr	Mfr Part No.
1	BDM-14	Conn-Pin Hdr-Thru(R)	2x7 Header	S2012E-07-nd	Sullins	PEC07DAAN
1	Target cable	Cable-ribbon	14 pos. IDC female both end, 6 inch long	H3AAH-1406-ND	Assman	H3AAH-1406G

Table A.8 COM Port Option

Qty	COM Port Option	Title	Detail	Digi-Key Part No.	Mfr	Mfr Part No.
1	U6	IC-RS232-Smt(R)	(Soic16) RS232,Dual Xcvr, 5V, 15K ESD	497-2058-1-ND	ST	ST232EBDR
1	COM	Connector- DB9, thru	DB9, R/A, Female	A35107-ND	Tyco	1734354-1
4	C16, C17, C18, C19	Cap-Cer-Smt(R)	(0805) .1uF, 50V	311-1140-1-ND	Yaego	CC0805KRX7R9BB104

Table A.8 COM Port Option

Qty	COM Port Option	Title	Detail	Digi-Key Part No.	Mfr	Mfr Part No.
1	FB2	Ind-FB-Smt(R)	(0805) 220 Ohms@ 100M,1.5A	490-1054-1-ND	Murata	BLM21PG221SN1D
1	Serial Cable	Cable - DB9 M/F	6ft, 2M long	AE1379-nd	Assman	AK131-2-R

Table A.9 S0812 V1 Target

Vendor	Vendor Part No.	Mfr	Mfr Part No.
Digi-Key	MC9S08JM60CFGE-ND	Freescale	MC9S08JM60CFGE
Digi-Key	576-1059-1-ND	Mircel	MIC2026-1-YM
Digi-Key	AE9925-ND	Assman	AU-Y1007-R
Digi-Key	XC1238CT-ND	ECS	ECS-40-20-5PX
Digi-Key	S2012E-03-ND	Sullins	PEC03DAAN
Digi-Key	311-1140-1-ND	Yaego	CC0805KRX7R9BB104
Digi-Key	311-1102-1-ND	Yaego	CC0805JRNP09BN180
Digi-Key	311-1127-1-ND	Yaego	CC0805KRX7R9BB102
Digi-Key	495-3263-1-ND	Epcos	B37641C8106K062

Table A.9 S0812 V1 Target

Vendor	Vendor Part No.	Mfr	Mfr Part No.
Digi-Key	516-1443-1-nd	Avago	HSMG-C150
Digi-Key	516-1442-1-nd	Avago	HSMY-C150
Digi-Key	490-1054-1-ND	Murata	BLM21PG221SN1D
Digi-Key	RHM10MARCT-ND	Rohm	MCR10EZPJ106
Digi-Key	RHM33.0CRCT-ND	Rohm	MCR10EZPF33R0
Digi-Key	RHM1.0KARCT-ND	Rohm	MCR10EZPJ102
Digi-Key	RHM4.7KARCT-ND	Rohm	MCR10EZPJ472
Digi-Key	RHM10KARCT-ND	Rohm	MCR10EZPJ103
ACD	AXM-0415C	Axiom Mfg	AXM-0415C
Digi-Key	AE1493-ND	Assman	AK672/2-2-R
Digi-Key	568-1741-1-ND	NXP	PMBT3904
Digi-Key	568-1742-1-ND	NXP	PMBT3906
Digi-Key	568-1610-1-ND	NXP	BAT54
Digi-Key	RHM4.7KARCT-ND	Rohm	MCR10EZPJ472
Digi-Key	RHM10KARCT-ND	Rohm	MCR10EZPJ103

Table A.9 S0812 V1 Target

Vendor	Vendor Part No.	Mfr	Mfr Part No.
Digi-Key	RHM47ARCT-ND	Rohm	MCR10EZPJ470
Digi-Key	S2012E-03-ND	Sullins	PEC03DAAN
Digi-Key	296-16843-1-ND	TI	SN74LVC1T45DBVR
Digi-Key	311-1140-1-ND	Yaego	CC0805KRX7R9BB104
Digi-Key	RHM4.7KARCT-ND	Rohm	MCR10EZPJ472
Digi-Key	RHM47ARCT-ND	Rohm	MCR10EZPJ470
Digi-Key	H3AAH-1006G-ND	Assman	H3AAH-1006G
Digi-Key	S1012E-02-ND	Sullins	PEC02SAAN
Digi-Key	609-2462-ND	FCI	71363-102LF
Digi-Key	296-12686-1-ND	TI	TPS61041DBV
Digi-Key	568-1741-1-ND	NXP	PMBT3904
Digi-Key	568-1742-1-ND	NXP	PMBT3906
Digi-Key	SS12-E3/1GI-ND	Vishay	SS12-E3/1
Digi-Key	568-1612-1-ND	NXP	BAT54C
Digi-Key	445-3176-1-ND	TDK	VLCF4018T-100MR74-2
Digi-Key	311-1127-1-ND	Yaego	CC0805KRX7R9BB102

Table A.9 S0812 V1 Target

Vendor	Vendor Part No.	Mfr	Mfr Part No.
Digi-Key	399-1284-1-ND	Kemet	C0805C105K4RACTU
Digi-Key	RHM4.7KARCT-ND	Rohm	MCR10EZPJ472
Digi-Key	RHM10KARCT-ND	Rohm	MCR10EZPJ103
Digi-Key	RHM1.00KCRCT-ND	Rohm	MCR10EZPF1001
Digi-Key	RHM9.09KCRCT-ND	Rohm	MCR10EZPF9091
Digi-Key	RHM100ARCT-ND	Rohm	MCR10EZPJ101
Digi-Key	S1012E-03-ND	Sullins	PEC03SAAN
Digi-Key	609-2462-ND	FCI	71363-102LF
Digi-Key	296-17783-1-ND	TI	TPS79933DDCR
Digi-Key	495-3263-1-ND	Epcos	B37641C8106K062
Digi-Key	497-2058-1-ND	ST	ST232EBDR
Digi-Key	A35107-ND	Tyco	1734354-1
Digi-Key	311-1140-1-ND	Yaego	CC0805KRX7R9BB104
Digi-Key	490-1054-1-ND	Murata	BLM21PG221SN1D
Digi-Key	AE1379-nd	Assman	AK131-2-R

Table A.10 CF V1, V2, V3 Targets

Vendor	Vendor Part No.	Mfr	Mfr Part No.
Digi-Key	MC9S08JM60CLH-ND	Freescale	MC9S08JM60CLH
Digi-Key	576-1059-1-ND	Micrel	MIC2026-1-YM
Digi-Key	AE9925-ND	Assman	AU-Y1007-R
Digi-Key	XC1238CT-ND	ECS	ECS-40-20-5PX
Digi-Key	S2012E-03-ND	Sullins	PEC03DAAN
Digi-Key	311-1140-1-ND	Yaego	CC0805KRX7R9BB104
Digi-Key	311-1102-1-ND	Yaego	CC0805JRNP09BN180
Digi-Key	311-1127-1-ND	Yaego	CC0805KRX7R9BB102
Digi-Key	495-3263-1-ND	Epcos	B37641C8106K062
Digi-Key	516-1443-1-nd	Avago	HSMG-C150
Digi-Key	516-1442-1-nd	Avago	HSMY-C150
Digi-Key	490-1054-1-ND	Murata	BLM21PG221SN1D
Digi-Key	RHM10MARCT-ND	Rohm	MCR10EZPJ106
Digi-Key	RHM33.0CRCT-ND	Rohm	MCR10EZPF33R0
Digi-Key	RHM1.0KARCT-ND	Rohm	MCR10EZPJ102

Table A.10 CF V1, V2, V3 Targets

Vendor	Vendor Part No.	Mfr	Mfr Part No.
Digi-Key	RHM4.7KARCT-ND	Rohm	MCR10EZPJ472
Digi-Key	RHM10KARCT-ND	Rohm	MCR10EZPJ103
ACD	AXM-0451	Axiom Mfg	AXM-0451
Digi-Key	AE1493-ND	Assman	AK672/2-2-R
Digi-Key	568-1741-1-ND	NXP	PMBT3904
Digi-Key	RHM10KARCT-ND	Rohm	MCR10EZPJ103
Digi-Key	RHM47ARCT-ND	Rohm	MCR10EZPJ470
Digi-Key	296-17783-1-ND	TI	TPS79933DDCR
Digi-Key	495-3263-1-ND	Epcos	B37641C8106K062
Digi-Key	S2012E-13-ND	Sullins	PEC13DAAN
Digi-Key	296-21028-1-ND	TI	SN74LV125ATDR
Digi-Key	296-1205-1-ND	TI	SN74HCT04DR
Digi-Key	568-1741-1-ND	NXP	PMBT3904
Digi-Key	RHM47ARCT-ND	Rohm	MCR10EZPJ470
Digi-Key	RHM4.7KARCT-ND	Rohm	MCR10EZPJ472
Digi-Key	RHM10KARCT-ND	Rohm	MCR10EZPJ103
Digi-Key	H3AAH-2606G-ND	Assman	H3AAH-2606G
Digi-Key	497-2058-1-ND	ST	ST232EBDR
Digi-Key	A35107-ND	Tyco	1734354-1

Table A.10 CF V1, V2, V3 Targets

Vendor	Vendor Part No.	Mfr	Mfr Part No.
Digi-Key	311-1140-1-ND	Yaego	CC0805KRX7R9BB104
Digi-Key	490-1054-1-ND	Murata	BLM21PG221SN1D
Digi-Key	AE1379-nd	Assman	AK131-2-R

Table A.11 JTAG Target

Vendor	Vendor Part No.	Mfr	Mfr Part No.
Digi-Key	MC9S08JM60CLH-ND	Freescale	MC9S08JM60CLH
Digi-Key	576-1059-1-ND	Micrel	MIC2026-1-YM
Digi-Key	AE9925-ND	Assman	AU-Y1007-R
Digi-Key	XC1238CT-ND	ECS	ECS-40-20-5PX
Digi-Key	S2012E-03-ND	Sullins	PEC03DAAN
Digi-Key	311-1140-1-ND	Yaego	CC0805KRX7R9BB104
Digi-Key	311-1102-1-ND	Yaego	CC0805JRNP09BN180
Digi-Key	311-1127-1-ND	Yaego	CC0805KRX7R9BB102
Digi-Key	495-3263-1-ND	Epcos	B37641C8106K062
Digi-Key	516-1443-1-nd	Avago	HSMG-C150

Table A.11 JTAG Target

Vendor	Vendor Part No.	Mfr	Mfr Part No.
Digi-Key	516-1442-1-nd	Avago	HSMY-C150
Digi-Key	490-1054-1-ND	Murata	BLM21PG221SN1D
Digi-Key	RHM10MARCT-ND	Rohm	MCR10EZPJ106
Digi-Key	RHM33.0CRCT-ND	Rohm	MCR10EZPF33R0
Digi-Key	RHM1.0KARCT-ND	Rohm	MCR10EZPJ102
Digi-Key	RHM4.7KARCT-ND	Rohm	MCR10EZPJ472
Digi-Key	RHM10KARCT-ND	Rohm	MCR10EZPJ103
ACD	AXM-0451	Axiom Mfg	AXM-0451
Digi-Key	AE1493-ND	Assman	AK672/2-2-R
Digi-Key	568-1741-1-ND	NXP	PMBT3904
Digi-Key	RHM10KARCT-ND	Rohm	MCR10EZPJ103
Digi-Key	RHM47ARCT-ND	Rohm	MCR10EZPJ470
Digi-Key	296-17783-1-ND	TI	TPS79933DDCR
Digi-Key	495-3263-1-ND	Epcos	B37641C8106K062
Digi-Key	S2012E-13-ND	Sullins	PEC13DAAN
Digi-Key	296-21028-1-ND	TI	SN74LV125ATDR
Digi-Key	296-1205-1-ND	TI	SN74HCT04DR

Table A.11 JTAG Target

Vendor	Vendor Part No.	Mfr	Mfr Part No.
Digi-Key	568-1741-1-ND	NXP	PMBT3904
Digi-Key	RHM47ARCT-ND	Rohm	MCR10EZPJ470
Digi-Key	RHM4.7KARCT-ND	Rohm	MCR10EZPJ472
Digi-Key	RHM10KARCT-ND	Rohm	MCR10EZPJ103
Digi-Key	H3AAH-2606G-ND	Assman	H3AAH-2606G
Digi-Key	497-2058-1-ND	ST	ST232EBDR
Digi-Key	A35107-ND	Tyco	1734354-1
Digi-Key	311-1140-1-ND	Yaego	CC0805KRX7R9BB104
Digi-Key	490-1054-1-ND	Murata	BLM21PG221SN1D
Digi-Key	AE1379-nd	Assman	AK131-2-R

Index

P

S

T

U

W