User's Guide

SLAU688B–May 2016–Revised March 2017

GCC ARM® Embedded Toolchain for SimpleLink™ MSP432™ Microcontrollers

This manual describes the setup and basic operation of the SimpleLink™ MSP432™ microcontroller (MCU) programming and debugging using GCC ARM compiler and the GDB debugger.

Contents

List of Figures

List of Tables

Trademarks

SimpleLink, MSP432, XDS, E2E are trademarks of Texas Instruments.
OS X is a registered trademark of Apple Inc.
Linux is a registered trademark of Linus Torvalds.
Windows is a registered trademark of Microsoft Corp.
All other trademarks are the property of their respective owners.

Preface: Read This First

How to Use This User's Guide

This manual describes only the setup and basic operation of the SimpleLink MSP432 MCU programming and debugging using GCC ARM compiler and the GDB debugger but does not fully describe the GCC ARM compiler or MSP432 microcontrollers or the complete development software and hardware systems. For details on these items, see the appropriate documents listed in Related Documentation.

This manual applies to the use of SimpleLink MSP432 software GCC Support package as a stand-alone package with the TI XDS110-ET, XDS100, or XDS2xx debuggers.

These tools contain the most up-to-date materials available at the time of packaging. For the latest materials (including data sheets, user's guides, software, and application information), visit the TI SimpleLink MSP432 MCU website or contact your local TI sales office.

Information About Cautions and Warnings

This document may contain cautions and warnings. The information in a caution or a warning is provided for your protection. Read each caution and warning carefully.

CAUTION

This is an example of a caution statement.

A caution statement describes a situation that could potentially damage your software or equipment.

WARNING

This is an example of a warning statement.

A warning statement describes a situation that could potentially cause harm to you.

Related Documentation

The primary sources of information about MSP432 MCUs are the device-specific data sheets and user's guides. The MSP432 MCU website contains the most recent version of these documents. The GCC documentation can be found on the GNU website. All related information for the GCC ARM compiler is available on the GCC ARM Embedded website.

MSP432 GCC ARM documentation

Using the GNU Compiler Collection, Richard M. Stallman (http://gcc.gnu.org/onlinedocs/gcc.pdf). See the *ARM Options* section.

GDB: The GNU Project Debugger, Free Software Foundation, Inc. (https://sourceware.org/gdb/current/onlinedocs/)

GCC ARM Embedded

MSP432 development tools documentation

MSP432P401R SimpleLink™ Microcontroller LaunchPad™ Development Kit (MSP-EXP432P401R) User's Guide

XDS™ Emulation Software Package

MSP432 device data sheets

MSP432P401R, MSP432P401M SimpleLink™ Mixed-Signal Microcontrollers

MSP432 device family technical reference manual

MSP432P4xx SimpleLink™ Microcontrollers Technical Reference Manual

If You Need Assistance

Support for the MSP432 MCUs and the hardware development tools is provided by the TI Product Information Center (PIC). Contact information for the PIC can be found on the TI website. The TI E2E™ Community support forums for the MSP432 MCUs provide open interaction with peer engineers, TI engineers, and other experts. Additional device-specific information can be found on the MSP432 MCU website.

1 Introduction

The MSP432 software GCC support package brings to you a new and fully supported open-source package.

This free GCC based package supports all MSP432 devices. This package can be used as a stand-alone package and has no code size limit.

This manual describes the use of the GCC ARM compiler and GDB with the MSP432 microcontrollers. The package supports Windows®, Linux®, and OS X® operating systems. This manual describes only the Windows operating systems package. The versions for Linux and OS X operating systems are similar and, therefore, are not described separately.

2 Installing MSP432 Software GCC Support Package

The MSP432 software GCC support package supports the following operating systems:

- Windows XP 32 bit or 64 bit
- Windows 7 32 bit or 64 bit
- Windows 8 32 bit or 64 bit
- Windows 10 32 bit or 64 bit
- Linux 32 bit or 64 bit
- OS X 64 bit

The MSP432 software GCC support package can be downloaded and installed as stand-alone package from the TI website for all supported operating systems. The MSP432 GCC stand-alone support package contains:

- Device support files
- XDS emulation pack including GDB Agent for XDS110-ET, XDS100, or XDS2xx
- USB drivers needed for Windows (ICDI and XDS) and Linux (XDS)

The GCC ARM Embedded contains the compiler and can be downloaded separately from the GCC ARM Embedded website.

NOTE: See the MSP432 software GCC support package release notes to find the GCC ARM version used for testing.

To install the MSP432 software GCC support package:

1. Download the corresponding package installer and run it (see Figure 1).

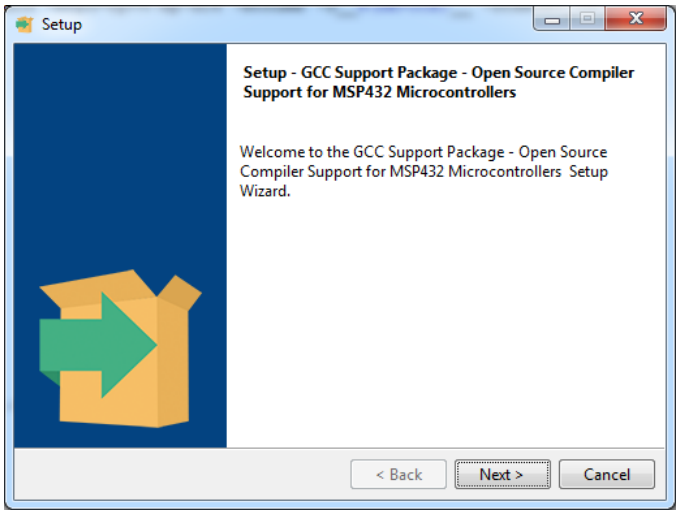

Figure 1. MSP432 Software GCC Support Stand-Alone Package Installer

2. Select the install directory and click Next (see Figure 2).

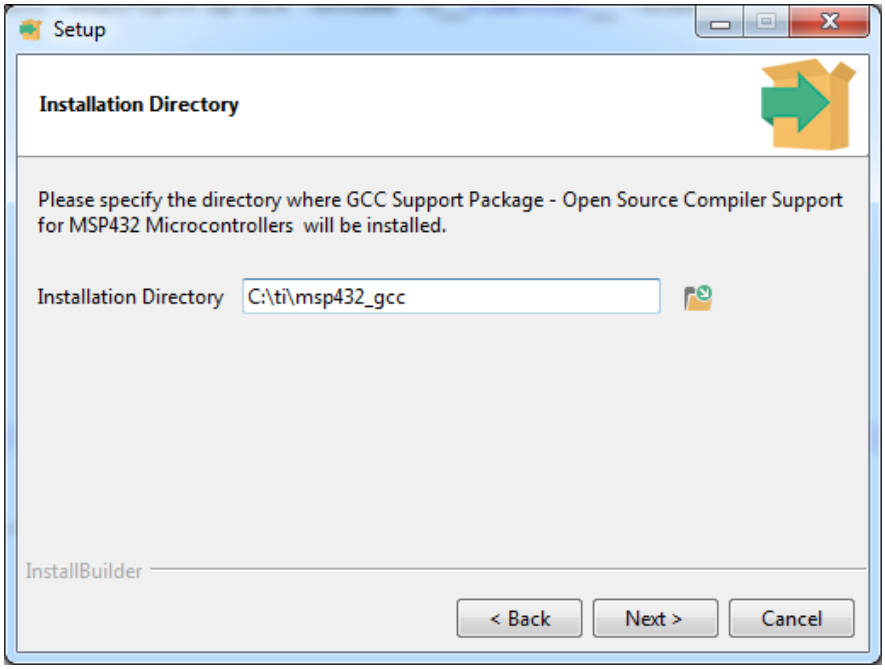

Figure 2. MSP432 Software GCC Support Stand-Alone Package Installation Directory

NOTE: For the Linux installer, apply sudo chmod +x <installer> before executing the package.

3. Download the GCC ARM Embedded package and extract the archive in the MSP432 software GCC Support package (see Figure 3). The default folder is c:\ti\msp432_gcc\arm_compiler.

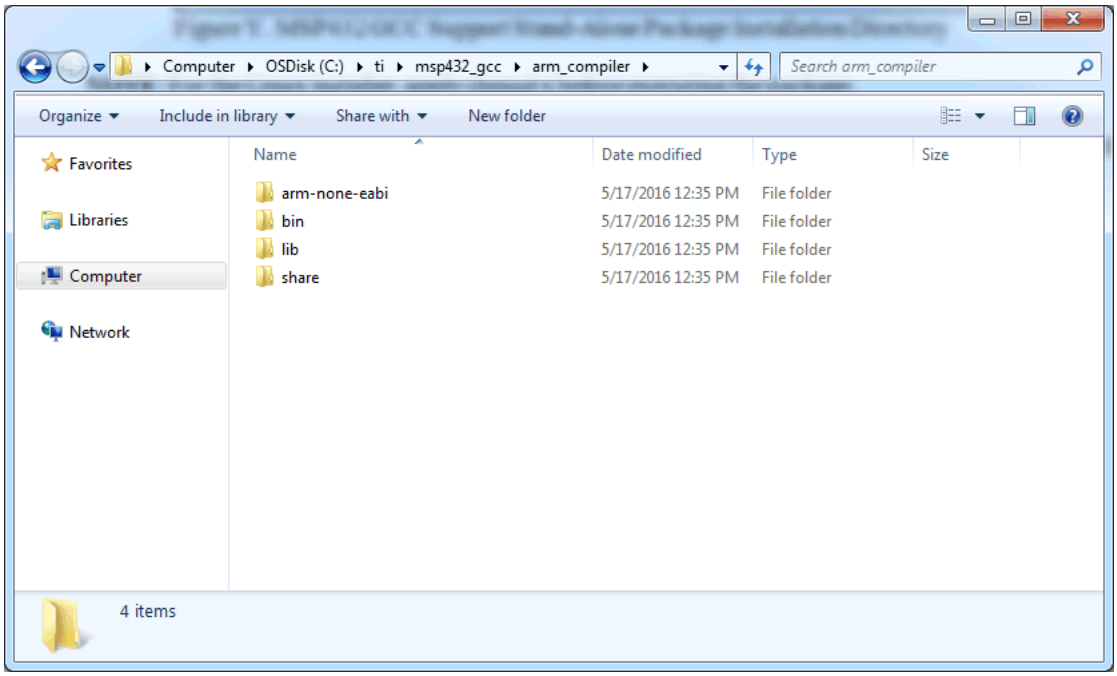

Figure 3. GCC ARM Embedded Installation Directory

3 MSP432 Software GCC Support Stand-Alone Package

3.1 MSP432 Software GCC Support Stand-Alone Packages

MSP432 software GCC Support stand-alone package is provided for users who prefer to use GCC compiler with GDB solutions for compiling and debugging. This stand-alone package supports different operating systems and is provided in different formats:

- MSP432 header and linker files
- GBD agent configuration for Windows, Linux, and Mac
- Examples

Table 1 lists all the available MSP432 software GCC Support stand-alone packages.

Table 1. MSP432 Software GCC Support Stand-Alone Package

Software	Description
msp432-gcc-x.x.x.x-linux-support-package-installer.run	MSP432 software GCC support package Linux installer includes support files and XDS emulation pack and USB drivers. Apply sudo chmod +x <installer> before executing the package.
msp432-gcc-x.x.x.x-linux-x64-support-package-installer.run	MSP432 software GCC support package Linux 64-bit installer includes support files and XDS emulation pack and USB drivers. Apply sudo chmod +x <installer> before executing the package.
msp432-gcc-x.x.x.x-windows-support-package-installer.exe	MSP432 software GCC support package Windows installer incl. support files and XDS emulation pack and USB drivers
msp432-gcc-x.x.x.x-osx-support-package-installer.app.zip	MSP432 software GCC support package Mac installer incl. support files and debug stack
msp432-gcc-support-files.zip	MSP432 device support files (CMSIS, headers, linkers, startup and system files.)
md5sum.txt	MD5 checksums

3.1.1 MSP432 Software GCC Support Stand-Alone Package Folder Structure

The placeholder INSTALL_DIR refers to the directory where you installed the MSP432 software GCC Support package.

- INSTALL_DIR
  - arm_compiler (**folder not provided with the package**)
    - arm-none-eabi
    - bin
    - lib
    - share
  - docs
  - emulation
    - common
      - bin
      - uscif
        - xds2xx
        - xds110
        - GDB Agent
        - xds2xxu_msp432_jtag.dat
        - xds2xxu_msp432_swd.dat
        - xds110_msp432_jtag.dat
        - xds110_msp432_swd.dat
        - FlashMSP432.dll

    - emulation (Windows XDS USB Drivers)
      - drivers
      - windows
  - examples
    - MSP432P401
  - arm (MSP432 Device Support files (CMSIS, headers, linkers, startup and system files)
    - include
      - CMSIS
    - src
  - MSP432-GCC_Standalone_Package_1.0.0_manifest.html
  - Release_Notes.txt
  - Software_Agreement.pdf

3.2 Quick Start

This document assumes that a version of the GNU Make utility is installed on the system and that it is available on the system path. The placeholder INSTALL_DIR refers to the directory where the MSP432 software GCC Support package is installed.

If the GCC ARM compiler is located at a different place than the default (Windows) C:\ti\msp432_gcc\arm_compiler, then the makefile needs to be adjusted with the GCC ARM path.

3.2.1 Building With a Makefile

1. In the command terminal, go to the INSTALL_DIR\examples directory.
2. There are examples that can run on Windows, Linux, and OS X.
   Choose one of the examples suitable for the MSP432 target device.
3. Change to the directory and type **make DEVICE=MSP432P401R** or **make DEVICE=MSP432P401M**.
4. The binary can now be downloaded and debugged on the target hardware.

3.2.2 Debugging

3.2.2.1 Starting GDB Agent

The GDB Agent is available only as command line version on all platforms (Windows, Linux, and OS X).

Open a command terminal, change to INSTALL_DIR/emulation/common/uscif/ and type:

For JTAG connection:

gdb_agent_console -f MSP432 xds110_msp432_jtag.dat

For SWD connection:

gdb_agent_console -f MSP432 xds110_msp432_swd.dat

NOTE: If using XDS2xx, then use the following *.dat files instead of the files listed above:
- xds2xxu_msp432_jtag.dat
- xds2xxu_msp432_swd.dat

3.2.2.2 Debugging With GDB

3.2.2.2.1 Running a Program in the Debugger

1. In the command terminal, go to the INSTALL_DIR\examples\[Selected example], and type the command **make debug DEVICE=MSP432P401R** or **make debug DEVICE=MSP432P401M**.
2. This command starts the GDB and waits for commands. This is indicated by the prompt <gdb>.
3. To connect GDB to the GDB Agent, type the command **target remote :55000** and press enter.
4. To load a program binary to the MSP432 target device, type **load**.
5. To reset the device, type **monitor reset**.
6. Type the command **continue** (short version: **c**) to tell GDB to run the loaded program.

3.2.2.2.2 GDB Breakpoints

Software breakpoints are not supported for flash code on MSP432. To use breakpoints for debugging using the GNU toolchain, use the ***hbreak*** command instead of the standard ***break*** instruction.

As a consequence of the fact that every single step sets up a software breakpoint by default, users must run the GDB command ***set breakpoint auto-hw on***.

3.2.2.2.3 Setting a Breakpoint

1. Connect the GDB to the GDB Agent as previously described and load a program to the device.
2. To set a breakpoint on a function, type **hbreak function name**.
3. To set a breakpoint on a source line, type **hbreak filename:line**.
4. When you run the program, the program execution stops at the entry to the specified function or stops at the specified line.

3.2.2.2.4 Single Stepping

1. Connect the GDB to the GDB Agent as previously described and load a program to the device.
2. After the debugger has stopped the program at a breakpoint, you can step through the code:
   (a) To execute the source line, type **next**. **next** does not step into functions, it executes the complete function and stops on the line following the function call.
   (b) To execute the next source line and step into functions, type **step**.
   (c) To execute the next instruction, type **nexti**.
   (d) To execute the next instruction and step into functions, type **stepi**.

3.2.2.2.5 Stopping or Interrupting a Running Program

1. Connect the GDB to the GDB Agent as previously described and load a program to the device.
2. To stop a running program and get back to the GDB command prompt, type **Ctrl+C**.

3.2.3 Creating a New Project

1. Create a directory for your project.
2. Copy one of the example project makefiles into the project directory.
3. Include all of the project source files (that is, the *.c files) as a dependency for the first target of the makefile.
4. Go to the project directory in a terminal and type **make DEVICE=MSP432P401R** or **make DEVICE=MSP432P401M** to build the project or **make debug DEVICE=MSP432P401R** or **make debug DEVICE=MSP432P401M** to start debugging the project.

The following is an example makefile.

```
#================================================================
#Makefile for building MSP Code Examples in command line
#environment using the GCC Open Source Compiler for MSP432
#================================================================

# Require DEVICE to be specified
ifndef DEVICE
$(info Please specify a device, e.g. DEVICE=MSP432P401R)
$(error unspecified device)
endif

OBJ_DIR            := output

###################### Windows OS ######################
ifeq ($(OS),Windows_NT)
    ################## GCC Root Variable ##################
    INSTALL_DIR         := ../../..
    GCC_MSP_INC_DIR     ?= $(INSTALL_DIR)/arm/include
    GCC_CMSIS_INC_DIR     ?= $(GCC_MSP_INC_DIR)/CMSIS
    LDDIR                := $(GCC_MSP_INC_DIR)/$(DEVICE)
    ifneq (,$(findstring cygwin, $(PATH)))
        RM                     := rm -rf
        MKDIR                = mkdir -p -- $@
    else
        RM                     := rd /s /q
        MKDIR                = mkdir
    endif
################## Linux or Mac OS ##################
else
    OS                     := $(shell uname)
    ################## GCC Root Variable ##################
    INSTALL_DIR            := ../../..
    GCC_MSP_INC_DIR     ?= $(INSTALL_DIR)/arm/include
    GCC_CMSIS_INC_DIR     ?= $(GCC_MSP_INC_DIR)/CMSIS
    LDDIR                := $(GCC_MSP_INC_DIR)/$(shell echo $(DEVICE) | tr A-Z a-z)
    RM                    := rm -rf
    MKDIR                = mkdir -p -- $@
endif

#######################################
GCC_BIN_DIR        ?= $(INSTALL_DIR)/arm_compiler/bin/
GCC_INC_DIR        ?= $(INSTALL_DIR)/arm_compiler/arm-none-eabi/include
#######################################
CC              := $(GCC_BIN_DIR)arm-none-eabi-gcc
GDB                := $(GCC_BIN_DIR)arm-none-eabi-gdb
#######################################
INCLUDES        := -I $(GCC_CMSIS_INC_DIR) -I $(GCC_MSP_INC_DIR) -I $(GCC_INC_DIR)
CFLAGS            := -mcpu=cortex-m4 -march=armv7e-m -mfloat-abi=hard -mfpu=fpv4-sp-
d16 -mthumb -D__$(DEVICE)__ -DTARGET_IS_MSP432P4XX -Dgcc -g -gstrict-dwarf -Wall -
ffunction-sections -fdata-sections -MD -std=c99
LDFLAGS            := -mcpu=cortex-m4 -march=armv7e-m -mfloat-abi=hard -mfpu=fpv4-sp-
d16 -mthumb -D__$(DEVICE)__ -DTARGET_IS_MSP432P4XX -Dgcc -g -gstrict-dwarf -Wall -
T$(LDDIR).lds -l'c' -l'gcc' -l'nosys'

#######################################

ifeq ($(DEVICE), MSP432P401R)
```

```
    STARTUP            := startup_msp432p401r_gcc
    SYSTEM            := system_msp432p401r
else
    STARTUP            := startup_msp432p401m_gcc
    SYSTEM            := system_msp432p401m
endif
SRC_FILE        := msp432p401_blinkingLED

OBJECTS            := $(OBJ_DIR)/$(SRC_FILE).o $(OBJ_DIR)/$(STARTUP).o
$(OBJ_DIR)/$(SYSTEM).o
########################################
all: $(OBJ_DIR)/$(SRC_FILE).out

$(OBJECTS): | $(OBJ_DIR)

$(OBJ_DIR):
    @$(MKDIR) $(OBJ_DIR)

$(OBJ_DIR)/%.o: %.c
    @echo ============================================
    @echo Generating $@
    $(CC) $(CFLAGS) $(INCLUDES) -c $< -o $@

$(OBJ_DIR)/$(SRC_FILE).out: . $(OBJECTS)
    @echo ============================================
    @echo Linking objects and generating output binary
    $(CC) $(LDFLAGS) $(word 2,$^) $(OBJ_DIR)/$(STARTUP).o $(OBJ_DIR)/$(SYSTEM).o -
o $@ $(INCLUDES)

debug: all
    $(GDB) $(OBJ_DIR)/$(SRC_FILE).out

clean:
    @$(RM) $(OBJ_DIR)
```

3.3 GDB Settings

The GDB Agent is a tool to connect the GDB with the target hardware to debug software. The GDB Agent uses the XDS emulation stack to connect to the hardware and provides an interface to the GDB (see Figure 4). Only the console application is supported on Windows, Linux, and OS X.

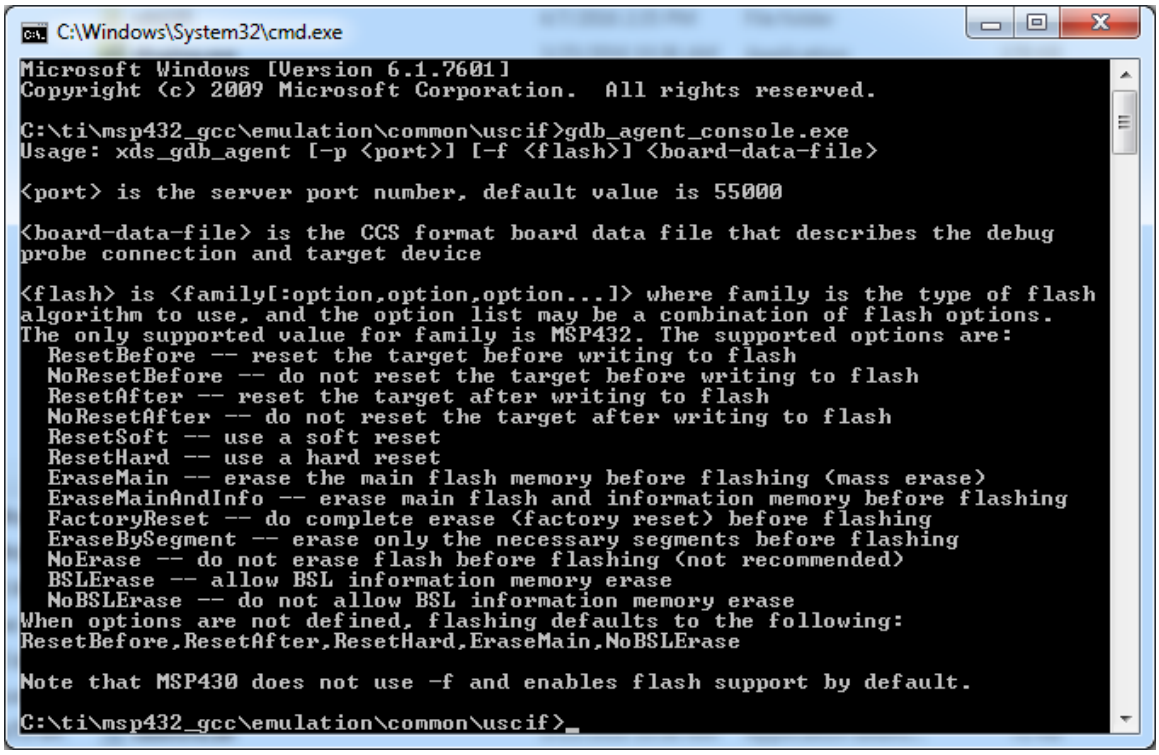

Figure 4. XDS GDB Agent

3.3.1 Console Application

Using the console application, invoke it from a command terminal using following syntax:

For XDS110 JTAG connection:

gdb_agent_console -f MSP432 xds110_msp432_jtag.dat

For XDS110 SWD connection:

gdb_agent_console -f MSP432 xds110_msp432_swd.dat

For XDS2xx JTAG connection:

gdb_agent_console -f MSP432 xds2xxu_msp432_jtag.dat

For XDS2xx SWD connection:

gdb_agent_console -f MSP432 xds2xxu_msp432_swd.dat

The console application opens a TCP/IP port on the local machine. It displays the port number in the console. By default, this port number is 55000.

3.3.2 Attaching the Debugger

After starting the debugger and to attach to the GDB server, use the target remote **[<host ip address>]:<port>** command, where **<port>** is the TCP/IP port from above. If the GDB Agent runs locally, omit the host IP address.

3.3.3 XDS Debug Probes Firmware Update

The GDB Agent does not support automatic XDS110 or XDS2xx firmware update. See the following sections for instructions on how to update the firmware of the XDS debug probes.

NOTE: The GDB Agent may be unable to connect to an XDS110 or XDS2xx that has out-of-date firmware. If GBD Agent has a problem connecting, proceed with updating the XDS110 or XDS2xx firmware.

3.3.3.1 XDS110 Firmware Update Using Firmware Maintenance Utility – xdsdfu

xdsdfu is a command line utility that provides several features for examining and maintaining the firmware of the XDS110 debug probe. xdsdfu also allows the user to view and set the XDS110 probe serial number.

xdsdfu provides the following features:

- Report the XDS110 firmware version and serial number
- Place the XDS110 into flash programming mode (DFU mode)
- Download the bootloader into the XDS110
- Download the firmware into the XDS110
- Set a new serial number into the XDS110
- Reset the XDS110 to restart the firmware

Installation path: INSTALL_DIR/emulation/common/uscif/xds110

Usage: xdsdfu <command> <...>

Supported commands:

- -e

  Enumerate connected devices, show information, then exit.
- -m

  Switch XDS110 into programming mode (DFU mode).
- -b <FILE>

  Download the given bootloader file into the device.
- -f <FILE>

  Download the given firmware file into the device.
- -s <TEXT>

  Set the XDS110 serial number to given text, any eight characters string (no spaces). This option replaces the entire serial number.
- -r

  Reset the XDS110 on completion of another command.
- -? or -h

  Show help for these and additional commands.

Examples

1. **How to examine the firmware in all connected XDS110 probes:** xdsdfu examines all connected XDS110 probes and devices in DFU mode, and reports the details of each device.

   xdsdfu -e

2. **How to program new firmware into the XDS110 probe:** The -m command must be executed separately. When -m is executed, the XDS110 reconfigures its USB interface to enable the DFU mode. It then reconnects as a different USB device, and the OS needs a moment to recognize it. The -r command tells the XDS110 to reboot after programming the firmware.

   xdsdfu -m

   xdsdfu -f firmware.bin –r

3. **How to program a new bootloader and firmware into the XDS110 probe:** The -m commands must be executed separately, but the second -m may not be necessary if the XDS110 probe flash was blank.

   xdsdfu -m

   xdsdfu -b bootloader.bin -r

   xdsdfu -m

   xdsdfu -f firmware.bin -r

4. **How to change the serial number of the XDS110 probe**

   xdsdfu –m

   xdsdfu -s 00000000 -r

3.3.3.2 XDS2xx Firmware Update

For instructions to update the XDS2xx firmware, see Updating the XDS200 firmware.

3.3.4 Resetting the Target

To reset the target, use the **monitor reset** command.

3.3.5 Halting the Target

To halt the target, use the **monitor halt** command.

3.3.6 MSP432 Flash Support Using GDB Agent

The GDB Agent supports MSP432 Flash programming options. The supported options are:

- **ResetBefore**: reset the target before writing to flash
- **NoResetBefore**: do not reset the target before writing to flash
- **ResetAfter**: reset the target after writing to flash
- **NoResetAfter**: do not reset the target after writing to flash
- **ResetSoft**: use a soft reset
- **ResetHard**: use a hard reset
- **EraseMain**: erase the main flash memory before flashing (mass erase)
- **EraseMainAndInfo**: erase main flash and information memory before flashing
- **FactoryReset**: do complete erase (factory reset) before flashing
- **EraseBySegment**: erase only the necessary segments before flashing
- **NoErase**: do not erase flash before flashing (not recommended)
- **BSLErase**: allow BSL information memory erase
- **NoBSLErase**: do not allow BSL information memory erase

When options are not defined, flashing defaults to the following: ResetBefore, ResetAfter, ResetHard, EraseMain, NoBSLErase.

NOTE: MSP430 MCUs do not use -f and enable flash support by default.

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from October 22, 2016 to March 6, 2017	Page
• Added "SimpleLink" branding and updated titles of referenced documents as necessary	1

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ('TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/sampterms.htm).

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265