Cypress Programmer 2.2 OpenOCD

CLI User Guide

Document Number: 002-26234 Rev. *C

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
http://www.cypress.com

Copyrights

Contents

1 Introduction 6

Overview 6

Acronyms and Abbreviations 6

Supported OS 7

Supported MCU Devices 7

Supported Hardware (Probes) 7

Installation 7

Document Convention 8

Revision History 8

2 Cypress Programmer Installation 9

Windows Package Contents 9

System Prerequisites 9

Installation Procedure on Windows 7,8,10 (x86 / x64) 9

J-Link configuration 11

Installation Procedure on Ubuntu Linux (x64) 11

Installation Procedure on Mac OS 12

3 Getting Started 13

Connect Device 13

List Connected Targets 13

Under Windows OS: 13

Under Linux OS: 13

Under macOS X: 14

Program PSoC 6 Target Device 14

Program Secure PSoC 6 Target Device (PSoC 64) 15

Program Device Using Configuration File Only 16

Program Device Using Configuration File and Command Line 16

4 Supported Target Configurations 17

5 Command-Line Options 18

--file (-f) 18

--search (-s) 19

--command (-c) 19

--debug (-d) 20

--log_output (-l) 20

--help (-h) 20
--version (-v) 20

6 OpenOCD Commands Overview 21

7 OpenOCD Commands Description 23

General OpenOCD Commands 23
version 23
help 23
shutdown 23
log_output 23
debug_level 24
reset_config 24
adapter_khz 24
transport list 25
transport select 25
targets 25
scan_chain 26
md(w)(h)(b) 26
mw(w)(h)(b) 27
init 27
reset [run] [halt] [init] 28
program 28
flash banks 29
flash list 29
flash info 30
flash erase_sector 30
flash erase_address 31
flash write_bank 31
flash write_image 32
flash fill(w)(h)(b) 32
flash read_bank 33
flash verify_bank 33
flash padded_value 34
flash rmw 34
add_verify_range 35
show_verify_ranges 35
clear_verify_ranges 36
verify_image 36
verify_image_checksum 37
load_image 37
dump_image 38

KitProg3/MiniProg4 Driver Commands 38
kitprog3 acquire_config 38
kitprog3 *acquire_psoc* 39
kitprog3 power_config 39
kitprog3 power_control 39
kitprog3 led_control 40
kitprog3 get_power 40

Flash Driver Commands 41
psoc6 sflash_restrictions 41
psoc6 allow_efuse_program 41
psoc6 reset_halt 42
psoc6 secure_app 42

Other Commands 42
source 42
find 43
set 43
sleep 43
add_script_search_dir 43

8 Global Variables 44
ENABLE_ACQUIRE 44
ENABLE_POWER_SUPPLY 44
ENABLE_CM0, ENABLE_CM4 44
SMIF_BANKS 44

9 Error Codes 46

10 Usage Examples 47
Erase Main Flash Rows 0…10 of PSoC 6 device 48
Display Memory Contents (32 words at address 0x08000000) of PSoC 6 device 48
Program PSoC 6 Device with Verification (Intel HEX file) 50
Program EFuse region of PSoC 6 Device 51
Modify Individual Bytes of PSoC 6 in Main Flash and Display Results 52
Read Memory of PSoC 6 Device to Binary File 53
Start GDB Server and Leave It Running 54

11 Documentation and Links 55

1 Introduction

Overview

Cypress Programmer (CYP) 2.2 is a flexible, cross-platform, integrated application to allow programming Cypress devices. It can perform Program, Erase, Verify, and Read operations on the flash of the target device. It can target an entire device, a specific region, a sector, and even a byte of a device.

The CYP 2.2 command-line interface (CLI) is based on the Open On-Chip Debugger (OpenOCD) project. OpenOCD is a powerful tool whose interface interacts with the target device via the JTAG/SWD debug ports. OpenOCD allows programming internal and external flash memories of a wide range of target devices, CFI-compatible flashes, and some CPLD/FPGA devices.

OpenOCD was originally developed by Dominic Rath at the University of Applied Sciences in Augsburg. The OpenOCD source code is now available through the GNU General Public License (GPL).

This document covers the Cypress-specific CLI extensions of OpenOCD. For more details about OpenOCD, refer to the official documentation available at http://openocd.org/documentation/.

Acronyms and Abbreviations

- CYP – Cypress Programmer.
- OpenOCD - Open On-Chip Debugger. An open-source tool that allows programming internal and external flash memories of a wide range of target devices.
- CLI – Command-line interface.
- Tcl - Tool Command Language. A high-level, general-purpose, interpreted, dynamic programming language.
- MPN – Marketing Part Number. This number is associated with each specific device and used to order a device or find information about a device from Cypress. Exp. CY8C616FMI-BL603, CY8C616FMI-BL673.
- SWD – Serial Wire Debug interface.
- JTAG – Joint Test Action Group. Specifies the use of a dedicated debug port implementing a serial communication interface for low-overhead access without requiring direct external access to the system address and data buses.
- TAP – JTAG Test Access Port.
- PSoC –Programmable System-on-Chip. A family of microcontroller integrated circuits by Cypress. These chips include a CPU core and mixed-signal arrays of configurable integrated analog and digital peripherals.
- MCU – Microcontroller Unit.
- UDD – Universal Device Database.
- FLD – Flash Loader Database.

- AP – Access Port register of ARM Cortex CPU. Used for programming and debugging, along with the corresponding SWD address bit selections.
- DP – Debug Port register of ARM Cortex CPU. Used for programming and debugging, along with the corresponding SWD address bit selections.
- Region – A logical area within the target device the programmer operates on.
- KP3 – KitProg3 device.
- MP4 – MiniProg4 device.

Supported OS

- Windows 7 SP1 (x86 / x64)
- Windows 8.1 (x86 / x64)
- Windows 10 (x86 / x64)
- Linux
- MacOS X

Supported MCU Devices

- PSoC 6
- STM32xxx

Supported Hardware (Probes)

- SEGGER J-Link
- Cypress KitProg3
- Cypress MiniProg4

Installation

The CYP 2.2 OpenOCD CLI software is installed as part of the ModusToolbox software installation. Refer to the *ModusToolbox Installation Guide* for more details.

For Auto BU follow installation instructions in Section 2.

Note Cypress Programmer 2.1 GUI will be installed separately from ModusToolbox. Refer to the *Cypress Programmer 2.1 GUI User Guide* for more details.

Document Convention

This guide uses the following conventions:

Convention	Usage
`Courier New`	Displays file locations and source code: `C:\ …cd\icc\`, user entered text
Italics	Displays file names and reference documentation: *sourcefile.hex*
[**bracketed, bold**]	Displays keyboard commands in procedures: [**Enter**] or [**Ctrl**] [**C**]
File > New Project	Represents menu paths: **File > New Project > Clone**
Bold	Displays commands, menu paths and selections, and icon names in procedures: Click the **Debugger** icon, and then click **Next**.
Text in gray boxes	Displays cautions or functionality unique to the device software.

Revision History

Document Title: Cypress Programmer 2.2 OpenOCD CLI User Guide
Document Number: 002-26234

Revision	Date	Description of Change
**	1/17/19	New document.
*A	01/24/19	Updated installation procedures for Windows and Linux sections.
*B	04/16/19	Updated for version 2.2. Added PSoC6A-512K configuration. Added descriptions for *show_verify_ranges*, *clear_verify_ranges*, and *psoc6 secure_app* commands.
*C	06/26/19	Removed all Traveo II (automotive) related information

2 Cypress Programmer Installation

Windows Package Contents

The CYP 2.2 package for Windows contains:

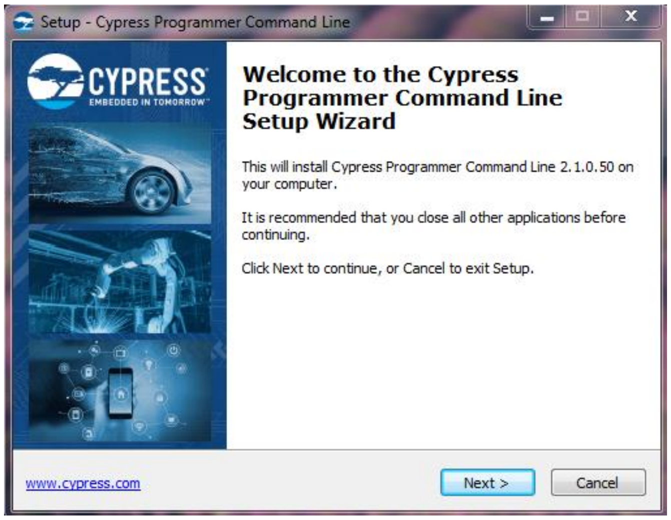
- *openocd.exe* – The CLI application to program PSoC devices based on the Open On-Chip Debugger.
- The drivers for the Cypress KitProg3/MiniProg4 hardware programmer.

System Prerequisites

- Windows 7 x86/x64 or later.

Installation Procedure on Windows 7,8,10 (x86 / x64)

1. Run the Windows installer for CYP 2.2.
2. Follow the instructions of the installation wizard:



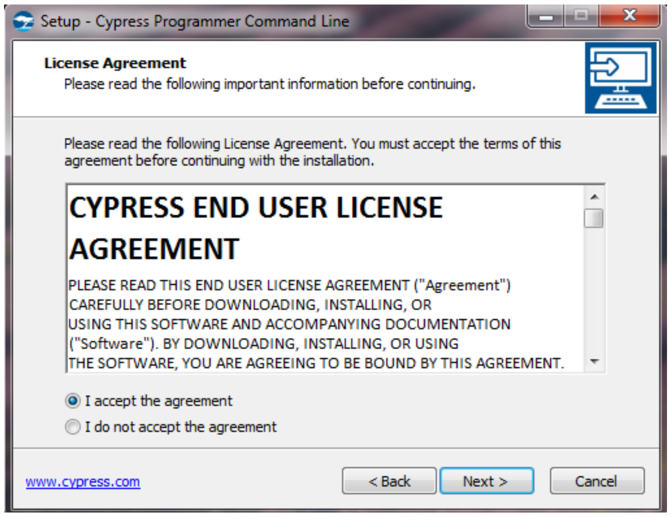
Setup - Cypress Programmer Command Line

Welcome to the Cypress Programmer Command Line Setup Wizard

This will install Cypress Programmer Command Line 2.1.0.50 on your computer.

It is recommended that you close all other applications before continuing.

Click Next to continue, or Cancel to exit Setup.



Setup - Cypress Programmer Command Line

License Agreement

Please read the following important information before continuing.

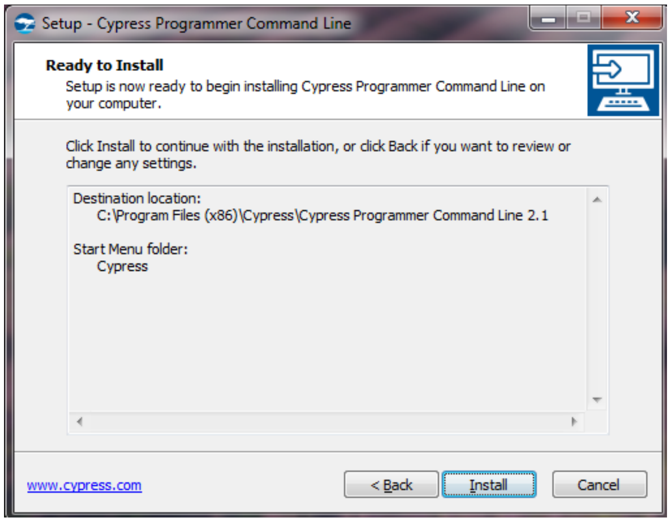
Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

CYPRESS END USER LICENSE AGREEMENT

PLEASE READ THIS END USER LICENSE AGREEMENT ("Agreement") CAREFULLY BEFORE DOWNLOADING, INSTALLING, OR USING THIS SOFTWARE AND ACCOMPANYING DOCUMENTATION ("Software"). BY DOWNLOADING, INSTALLING, OR USING THE SOFTWARE, YOU ARE AGREEING TO BE BOUND BY THIS AGREEMENT.

[✓ I accept the agreement]

I do not accept the agreement



Setup - Cypress Programmer Command Line

Ready to Install

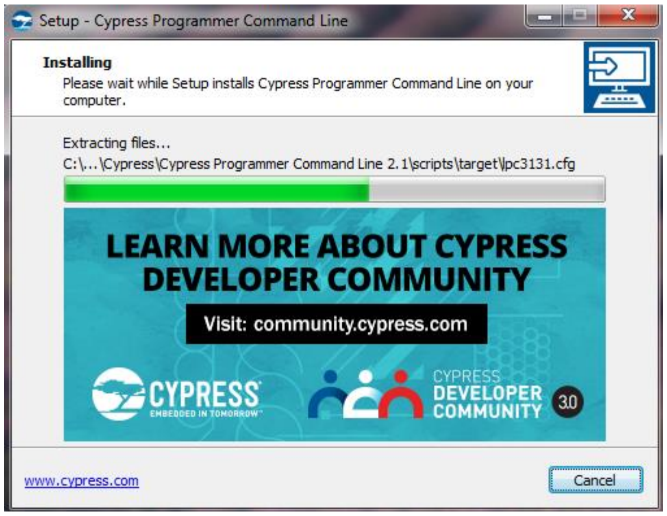
Setup is now ready to begin installing Cypress Programmer Command Line on your computer.

Click Install to continue with the installation, or click Back if you want to review or change any settings.

Destination location:
C:\Program Files (x86)\Cypress\Cypress Programmer Command Line 2.1

Start Menu folder:
Cypress

www.cypress.com
< Back
Install
Cancel



Setup - Cypress Programmer Command Line

Installing

Please wait while Setup installs Cypress Programmer Command Line on your computer.

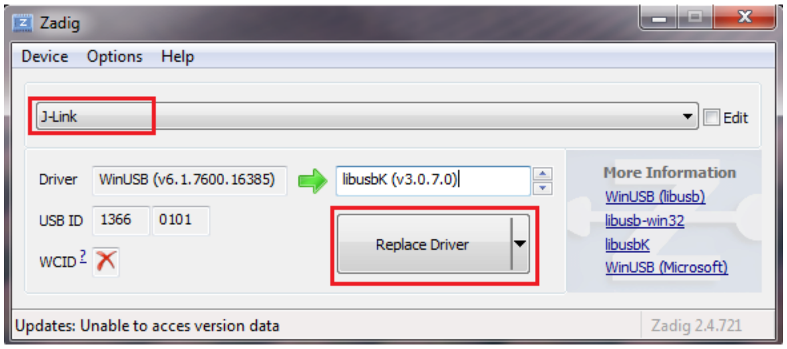
Extracting files...
C:\...\Cypress\Cypress Programmer Command Line 2.1\scripts\target\lpc3131.cfg

LEARN MORE ABOUT CYPRESS DEVELOPER COMMUNITY
Visit: community.cypress.com
CYPRESS DEVELOPER COMMUNITY 3.0
www.cypress.com
Cancel

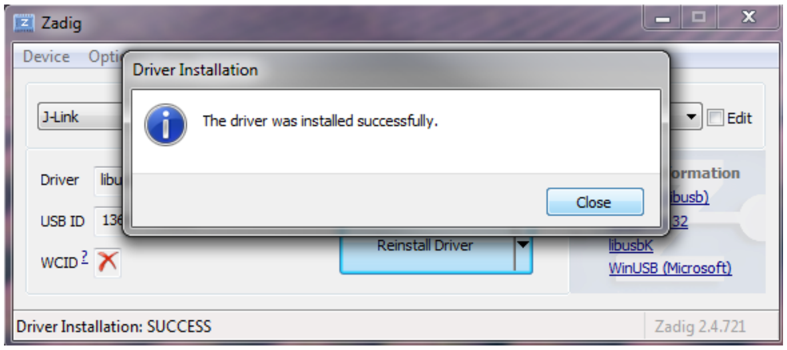
J-Link configuration

In order to use J-Link with Cypress Programmer 2.2, you need to replace the J-Link driver with the libusbK driver:

1. Download Zadig tool from https://zadig.akeo.ie/
2. Run zadig-2.4 executable.
3. Select "Options > List all devices".
4. Select J-Link in the drop-down menu.
5. Select "libusbK (v3.0.7.0)" driver.
6. Click Replace Driver.



Updates: Unable to acces version data



The driver was installed successfully.

Driver Installation: SUCCESS

Installation Procedure on Ubuntu Linux (x64)

1. Open a Terminal window.
2. Unpack the Cypress Programmer 2.2 installation archive. Run the following command:

```
tar -xvzf openocd_2.2.0.xxx.zip
```

3. The script *install_rules.sh* copies *60-openocd.rules* and *66-wiced-JTAG.rules* files to */etc/udev/rules.d*. It allows a non-superuser to have access to the connected devices.

   Install the rules for the connected programming hardware MiniProg4/KitProg3 from the following directory:

   openocd-2.2/udev_rules/

4. Run the following command:

```
./install_rules.sh
```

5. Under Ubuntu 18.xx OS, install additional "libusb-0.1-4" package in order for Cypress Programmer 2.2 to work correctly. Run the following command:

   ```
   sudo apt-get install libusb-0.1-4
   ```

6. Run the following command to change directory

   ```
   cd openocd-2.2/bin
   ```

7. Launch Cypress Programmer 2.2. Run the following command:

   ```
   ./openocd --help
   ```

Installation Procedure on Mac OS

1. Download the Cypress Programmer 2.2 installation archive package and extract it to a writable location (e.g. your home folder).
2. Run the Terminal window.
3. Change the directory:

   ```
   cd openocd-2.2/bin
   ```

4. Launch Cypress Programmer 2.2. Run:

   ```
   ./openocd –help
   ```

3 Getting Started

Connect Device

Connect the host computer to a Probe or Kit device – e.g. KitProg3 Kit with the PSoC 6 target device, used in the examples below.

Make sure the target MCU device is attached to your probe.

List Connected Targets

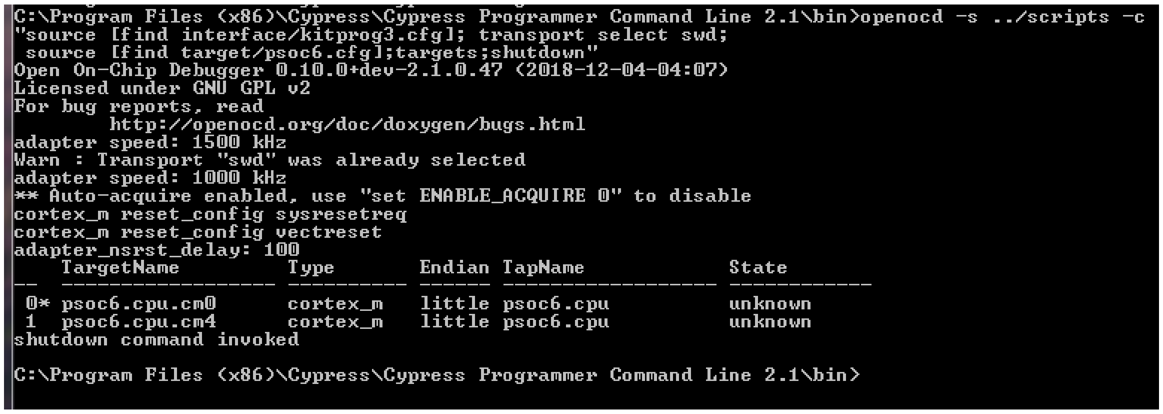
The example below displays target names available in the PSoC 6 device connected to the KitProg3 hardware programmer. The programmer communicates with the PSoC 6 device over the SWD hardware interface.

Under Windows OS:

Open the command-line window. Invoke "`cd C:\Program Files (x86)\Cypress\Cypress Programmer Command Line 2.2\bin`" to change the directory to the CYP 2.2 installation folder.

Execute the command:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "targets;
shutdown"
```



```
C:\Program Files (x86)\Cypress\Cypress Programmer Command Line 2.1\bin>openocd -s ../scripts -c
"source [find interface/kitprog3.cfg]; transport select swd;
 source [find target/psoc6.cfg];targets;shutdown"
Open On-Chip Debugger 0.10.0+dev-2.1.0.47 (2018-12-04-04:07)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
Warn : Transport "swd" was already selected
adapter speed: 1000 kHz
** Auto-acquire enabled, use "set ENABLE_ACQUIRE 0" to disable
cortex_m reset_config sysresetreq
cortex_m reset_config vectreset
adapter_nsrst_delay: 100
    TargetName         Type       Endian TapName            State
--  ------------------ ---------- ------ ------------------ ------------
 0* psoc6.cpu.cm0      cortex_m   little psoc6.cpu          unknown
 1  psoc6.cpu.cm4      cortex_m   little psoc6.cpu          unknown
shutdown command invoked

C:\Program Files (x86)\Cypress\Cypress Programmer Command Line 2.1\bin>
```

The command output displays the list of target names (JTAG TAPs) attached to the programming device.

Under Linux OS:

Open the terminal window.

Go to the directory where CYP 2.2 is installed (e.g. `~/cyprogrammer_2.2/bin`).

Execute the command:

```
./openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "targets;
shutdown"
```

The command output displays the list of target names (JTAG TAPs) attached to the programming device.

Under macOS X:

Open the terminal window.

Go to the directory where CYP 2.2 is installed (e.g. `~/openocd-2.2/bin`).

Execute the command:

```
./openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "targets;
shutdown"
```

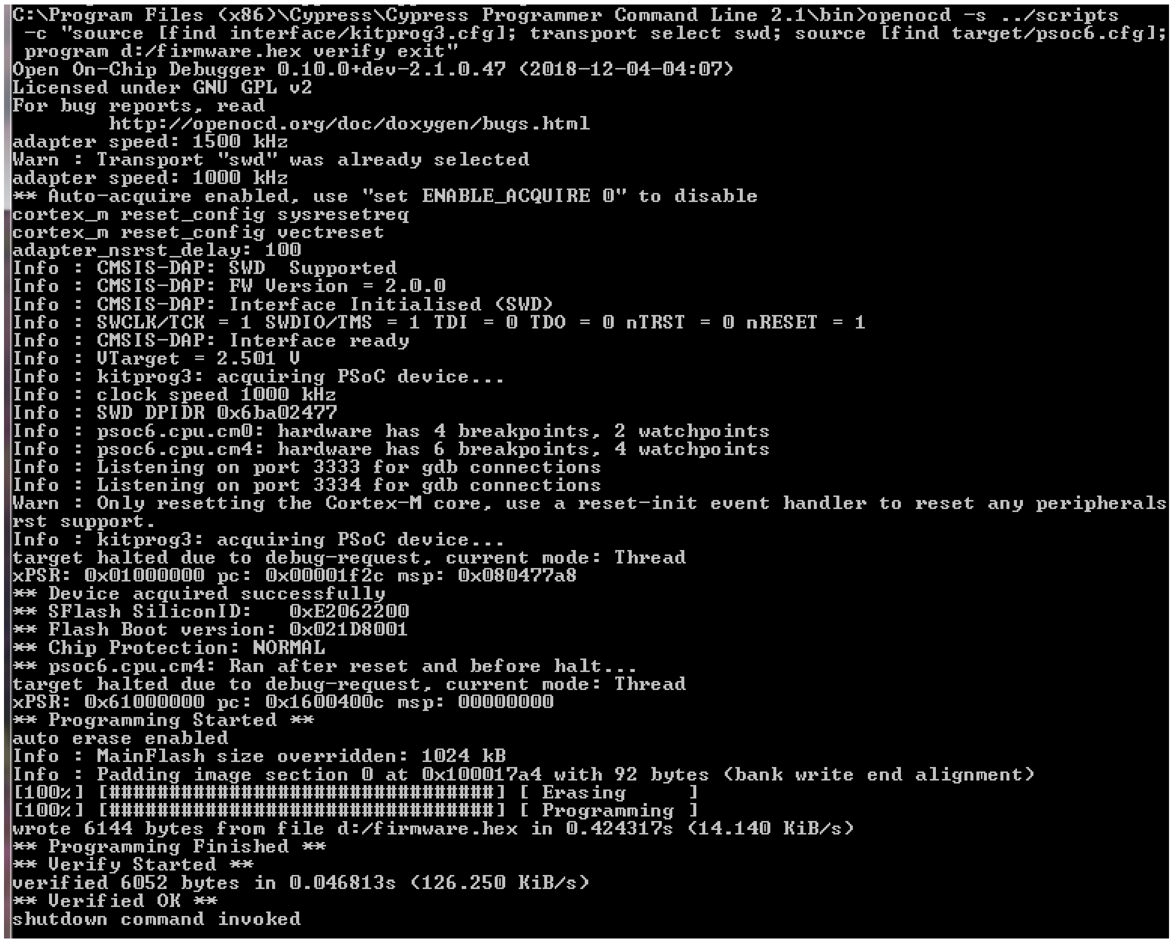
The command output displays the list of target names (JTAG TAPs) attached to the programming device.

Program PSoC 6 Target Device

The next example initializes the KitProg3 probe with the PSoC 6 target device, programs flash with the *firmware.hex* file, verifies programmed data, and finally shuts down the OpenOCD programmer.

Execute the command:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "program
d:/firmware.hex verify exit"
```



```
C:\Program Files (x86)\Cypress\Cypress Programmer Command Line 2.1\bin>openocd -s ../scripts
 -c "source [find interface/kitprog3.cfg]; transport select swd; source [find target/psoc6.cfg];
 program d:/firmware.hex verify exit"
Open On-Chip Debugger 0.10.0+dev-2.1.0.47 (2018-12-04-04:07)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
Warn : Transport "swd" was already selected
adapter speed: 1000 kHz
** Auto-acquire enabled, use "set ENABLE_ACQUIRE 0" to disable
cortex_m reset_config sysresetreq
cortex_m reset_config vectreset
adapter_nsrst_delay: 100
Info : CMSIS-DAP: SWD  Supported
Info : CMSIS-DAP: FW Version = 2.0.0
Info : CMSIS-DAP: Interface Initialised (SWD)
Info : SWCLK/TCK = 1 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 1
Info : CMSIS-DAP: Interface ready
Info : VTarget = 2.501 V
Info : kitprog3: acquiring PSoC device...
Info : clock speed 1000 kHz
Info : SWD DPIDR 0x6ba02477
Info : psoc6.cpu.cm0: hardware has 4 breakpoints, 2 watchpoints
Info : psoc6.cpu.cm4: hardware has 6 breakpoints, 4 watchpoints
Info : Listening on port 3333 for gdb connections
Info : Listening on port 3334 for gdb connections
Warn : Only resetting the Cortex-M core, use a reset-init event handler to reset any peripherals
rst support.
Info : kitprog3: acquiring PSoC device...
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x00001f2c msp: 0x080477a8
** Device acquired successfully
** SFlash SiliconID:    0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
** Programming Started **
auto erase enabled
Info : MainFlash size overridden: 1024 kB
Info : Padding image section 0 at 0x100017a4 with 92 bytes (bank write end alignment)
[100%] [##################################] [ Erasing     ]
[100%] [##################################] [ Programming ]
wrote 6144 bytes from file d:/firmware.hex in 0.424317s (14.140 KiB/s)
** Programming Finished **
** Verify Started **
verified 6052 bytes in 0.046813s (126.250 KiB/s)
** Verified OK **
shutdown command invoked
```

Program Secure PSoC 6 Target Device (PSoC 64)

The next example initializes the KitProg3 probe with the PSoC 64 target device, programs flash with the *firmware.hex* file, verifies programmed data, and finally shuts down the OpenOCD programmer.

Execute the command:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6_secure.cfg -c
"program d:/firmware.hex verify exit"
```

Note The psoc6_secure.cfg configuration file programming of internal flash is performed via SYS_AP Access Port. OpenOCD will not touch CM0_AP and CM4_AP by default, so both cores will not be visible to OpenOCD. Any core can be individually enabled using the ‘ENABME_CMx’ group of global variables. Refer to the ENABLE_CM0, ENABLE_CM4 section for more details.

Programming of external memory is done by the flash loader, so the CM4 core must be enabled for QSPI memory programming. After enabling the CM4 core, the QSPI memory bank will be exposed automatically.

Note See Supported Target Configurations for the list of available target configurations.

Program Device Using Configuration File Only

The whole configuration is stored in a single *sample.cfg* configuration file. For example, the following configuration file describes the PSoC 6 device connected using the KitProg3 debug probe. This file initializes the target device, programs flash with the *firmware.hex* file, verifies programmed data, and finally shuts down the OpenOCD programmer.

```
source [find interface/kitprog3.cfg]
transport select swd
source [find target/psoc6.cfg]
program d:/firmware.hex verify exit
```

Execute the command:

```
openocd -s ../scripts -f path/to/sample.cfg
```

Program Device Using Configuration File and Command Line

A significant part of the configuration file specifies the debug adapter, transport type, target chip, SWD frequency, reset type, etc. This part of the file reflects the hardware configuration and thus stays unchanged between sessions. In some cases, a combined method of passing the Tcl commands is more convenient:

The example *sample.cfg* file contents:

```
source [find interface/kitprog3.cfg]
transport select swd
source [find target/psoc6.cfg]
```

Execute the command:

```
openocd -s ../scripts -f path/to/sample.cfg -c "program d:/firmware.hex verify
exit"
```

4 Supported Target Configurations

To connect Cypress Programmer 2.2 to a target device, pass one of the following configuration files as the argument for the `--file` command-line option; for example, `-f target/psoc6.cfg`. The following configuration files are located in the `target/` directory of the OpenOCD tree.

#	Target Config	Description
1	psoc6.cfg	PSoC 6 1M target configuration (including PSoC6A-BLE2 devices with 512 Kbytes of the application flash)
2	psoc6_2m.cfg	PSoC 6 2M target configuration.
3	psoc6_512K.cfg	PSoC 6 512K target configuration
4	psoc6_secure.cfg	PSoC 6 1M secure target configuration.

5 Command-Line Options

OpenOCD is a command-line tool but it has only several command-line options. Several options can be combined in a single command-line.

The most important options and commands:

Option	Description
--file (-f)	Specifies the configuration file to use.
--search (-s)	Specifies the directory to search for configuration files.
--command (-c)	Executes an OpenOCD command. See Section OpenOCD Commands Overview for details.
--debug (-d)	Specifies the debug level.
--log_output (-l)	Redirects the log output to the file.
--help (-h)	Displays the help message.
--version (-v)	Displays the OpenOCD version.

--file (-f)

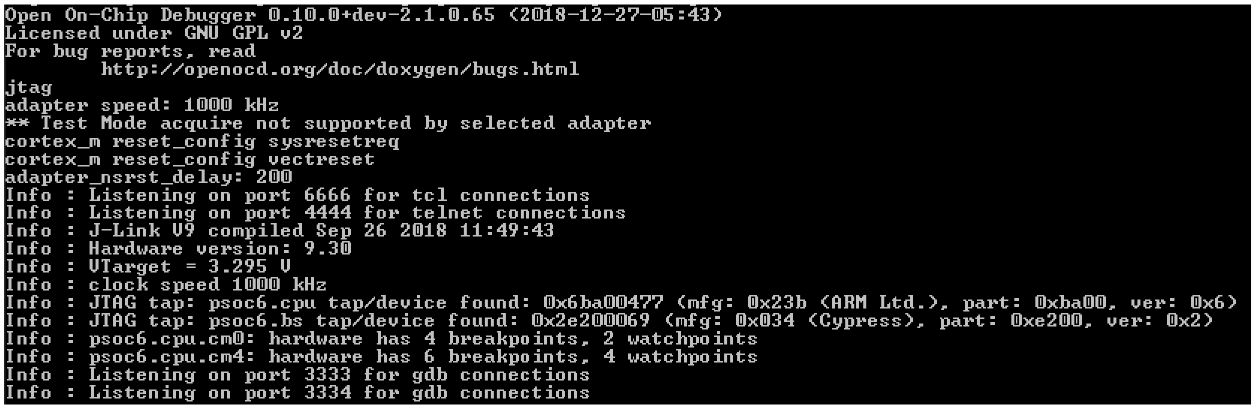
Specifies the configuration file to use. Multiple configuration files can be specified from a command line. They are interpreted in the order they are specified in the command line.

```
openocd -f <filename.cfg>
openocd -f interface/ADAPTER.cfg -f target/TARGET.cfg
```

Example:

```
openocd -s ../scripts -f interface/jlink.cfg -c "transport select jtag" -f
target/psoc6.cfg
```

Output similar to the following should display:



```
Open On-Chip Debugger 0.10.0+dev-2.1.0.65 (2018-12-27-05:43)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
jtag
adapter speed: 1000 kHz
** Test Mode acquire not supported by selected adapter
cortex_m reset_config sysresetreq
cortex_m reset_config vectreset
adapter_nsrst_delay: 200
Info : Listening on port 6666 for tcl connections
Info : Listening on port 4444 for telnet connections
Info : J-Link V9 compiled Sep 26 2018 11:49:43
Info : Hardware version: 9.30
Info : VTarget = 3.295 V
Info : clock speed 1000 kHz
Info : JTAG tap: psoc6.cpu tap/device found: 0x6ba00477 (mfg: 0x23b (ARM Ltd.), part: 0xba00, ver: 0x6)
Info : JTAG tap: psoc6.bs tap/device found: 0x2e200069 (mfg: 0x034 (Cypress), part: 0xe200, ver: 0x2)
Info : psoc6.cpu.cm0: hardware has 4 breakpoints, 2 watchpoints
Info : psoc6.cpu.cm4: hardware has 6 breakpoints, 4 watchpoints
Info : Listening on port 3333 for gdb connections
Info : Listening on port 3334 for gdb connections
```

The "tap/service found" message should display with no warnings. That means the JTAG communication is working.

--search (-s)

Specifies the directory to search for configuration files. Multiple -s options can be specified. Configuration files and scripts are searched for in the following paths:

- the current directory
- any search directory specified on the command line using the -s option
- any search directory specified using the **add_script_search_dir** command
- $HOME/.openocd (not on Windows)
- a directory in the OPENOCD_SCRIPTS environment variable (if set)
- the site wide-script library $pkgdatadir/site
- the OpenOCD-supplied script library $pkgdatadir/scripts.

The first found file with a matching file name is used.

```
openocd -s <directory>
```

Example:

```
openocd -s ../scripts -f interface/jlink.cfg -f target/psoc6.cfg
```

In this example, the -s option specifies the relative path to the directory where the interface and target configurations are located.

--command (-c)

Executes the Tcl command(s). Multiple commands can be executed by either specifying the multiple -c options or passing several commands to the single -c options. In the latter case, separate the commands with a semicolon.

```
openocd -c <command>
openocd -c <"command1; command2; …">
```

Example:

```
openocd -s ../scripts -f interface/jlink.cfg -f target/psoc6.cfg -c "targets;
shutdown"
```

--debug (-d)

Specifies the debug level. This affects the kind of messages sent to the server log. Level 0 is error messages only; Level 1 adds warnings; Level 2 adds informational messages; and Level 3 adds debugging messages. The debug level is 2 by default.

```
openocd -d<n>
```

Example:

```
openocd -d1
```

--log_output (-l)

Redirects the log output to the file <logfile.txt>.

```
openocd -l <logfile.txt>
```

Example:

```
openocd -s ../scripts -f interface/jlink.cfg -f target/psoc6.cfg -l d:/log.txt -c
"targets; shutdown"
```

--help (-h)

Displays the help message.

```
openocd -h
```

--version (-v)

Displays the OpenOCD version.

```
openocd -v
```

6 OpenOCD Commands Overview

The available OpenOCD Tcl commands are listed in the following table. You can combine several commands in a single command-line or pass them via the configuration file.

The command can be invoked with "`-c command`" command line option.

Command	Description
version	Displays a string identifying the version of the OpenOCD.
help	With no parameters, prints the help text for all commands.
shutdown	Closes the OpenOCD server, disconnecting all clients.
log_output	Redirects logging to the filename; the initial log output channel is *stderr*.
debug_level	Displays the debug level.
reset_config	Displays or modifies the reset configuration of your combination of the board and target.
adapter_khz	Sets the non-zero speed in KHZ for the debug adapter.
transport list	Displays the names of the transports supported by this version of OpenOCD.
transport select	Selects which of the supported transports to use in this OpenOCD session.
targets	Displays a table of all known targets or sets the current target to a given target with a given name.
scan_chain	Displays the TAPs in the scan chain configuration, and their status.
md(w)(h)(b)	Displays the contents of the address as 32-bit words (mdw), 16-bit half-words (mdh), or 8-bit bytes (mdb).
mw(w)(h)(b)	Writes the specified word (32 bits), half-word (16 bits), or byte (8-bit) value, at the specified address.
init	Terminates the configuration stage and enters the run stage.
reset [run] [halt] [init]	Performs as hard a reset as possible, using SRST if possible.
program	Programs a given programming file in the HEX, SREC, BIN or ELF formats into flash.
flash banks	Prints a one-line summary of each flash bank of the target device.
flash list	Retrieves a list of associative arrays for each device that was declared using a flash bank numbered from zero.
flash info	Prints info about the flash bank, a list of protection blocks and their status.
flash erase_sector	Erases sectors in a given bank.
flash erase_address	Erases sectors starting at a given address.
flash write_bank	Writes the binary file to a given flash bank.
flash write_image	Writes the image file to the current target's flash bank(s).
flash fill(w)(h)(b)	Fills flash memory with the specified word (32 bits), half-word (16 bits), or byte.

Command	Description
flash read_bank	Reads bytes from the flash bank and writes the contents to the binary file.
flash verify_bank	Compares the contents of the binary file with the contents of the flash.
flash padded_value	Sets the default value used for padding-any-image sections.
flash rmw	Can be used to modify flash individual bytes.
add_verify_range	Allows specifying memory regions to be compared during *verify* operation.
show_verify_ranges	Displays all active verify ranges for all targets that were added using the add_verify_range command. This command does not take any arguments.
clear_verify_ranges	Deletes all verify ranges for the specified target that were added using the add_verify_range command.
verify_image	Verifies a file against the target memory starting at a given address.
verify_image_checksum	Verifies a file against the target memory starting at a given address.
load_image	Loads an image from a file to the target memory offset from its load address.
dump_image	Dumps bytes of the target memory to the binary file.
kitprog3 acquire_config	Controls device acquisition parameters and optionally enables acquisition during the early initialization phase.
kitprog3 acquire_psoc	Performs device acquisition.
kitprog3 power_config	Controls the KP3/MP4 internal power supply parameters and optionally enables power.
kitprog3 power_control	Turns on or off the KP3/MP4 internal power supply.
kitprog3 led_control	Controls the KP3/MP4 LEDs.
kitprog3 get_power	Reports the target voltage in millivolts.
psoc6 sflash_restrictions	Enables or disables writes to Sflash regions other than USER, NAR, TOC2, and KEY.
psoc6 allow_efuse_program	Allows or disallows writes to the EFuse region.
psoc6 reset_halt	Simulates a broken Vector Catch on PSoC6 devices.
psoc6 secure_app	Enables or disables workarounds for the secure family of PSoC 6 devices.
source	Reads a file and executes it as a Tcl script.
find	Finds and returns the full path to a file with the Tcl script.
set	Creates a Tcl variable.
add_script_search_dir	Adds a directory to the file/script search path.
sleep	Waits for a given number of milliseconds before resuming.

7 OpenOCD Commands Description

This section includes all relevant OpenOCD commands along with their descriptions and usage examples.

All examples described in this section can be executed against different PSoC 6 target devices (E.g. 1M, 2M, 512K or secure device). See Supported Target Configurations for the detailed list of available target devices and corresponding OpenOCD configuration files.

General OpenOCD Commands

version

Displays a string identifying the version of the OpenOCD.

Example:

```
openocd -c "version; shutdown"
```

help

With no parameters, prints help text for all commands. Otherwise, prints each help-text-containing string. Not each command provides help text.

```
help [string]
```

Example:

```
openocd -c "help; shutdown"
```

shutdown

Closes the OpenOCD server, disconnecting all clients (GDB, telnet, other). If option `error` is used, OpenOCD will return non-zero exit code to the parent process.

```
shutdown [error]
```

Example:

```
openocd -c "shutdown error"
```

log_output

Redirects logging to the filename; the initial log output channel is stderr.

```
log_output [filename]
```

Example:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "log_output
d:/log.txt; targets; shutdown"
```

debug_level

Displays the debug level. If *n* (from 0..3) is provided, then set it to that level. This affects the kind of messages sent to the server log. Level 0 is error messages only; Level 1 adds warnings; Level 2 adds informational messages; and Level 3 adds debugging messages. The default is Level 2, but that can be overridden on the command line along with the location of that log file (which is normally the server's standard output).

```
debug_level [n]
```

Example:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "debug_level
1; targets; shutdown"
```

reset_config

Displays or modifies the reset configuration of your combination of the board and target.

```
reset_config <mode_flag> ...
```

The *mode_flag* options can be specified in any order, but only one of each type – `signals`, `combination`, `gates`, `trst_type`, `srst_type and connect_type` – may be specified at a time. If you don't provide a new value for a given type, its previous value (perhaps the default) remains unchanged. For example, do not say anything about TRST just to declare that if the JTAG adapter should want to drive SRST, it must explicitly be driven high (`srst_push_pull`).

`signals` specifies which of the reset signals is/are connected. For example, If the board doesn't connect SRST provided by the JTAG interface properly, OpenOCD cannot use it. The possible values are:

- `none` (the default)
- `trst_only`
- `srst_only`
- `trst_and_srst`

For more details, refer to the OpenOCD documentation.

Example:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c
"reset_config trst_and_srst; targets; shutdown"
```

adapter_khz

Sets a non-zero speed in KHZ for the debug adapter. Hence: 3000 is 3 MHz.

```
adapter_khz <max_speed_kHz>
```

JTAG interfaces usually support a limited number of speeds. The speed actually used will not be faster than the speed specified. Chip datasheets generally include a top JTAG clock rate. The actual rate is often a function of a CPU core clock, and is normally smaller than that peak rate. For example, most ARM cores accept at most one sixth of the CPU clock. Speed 0 (khz) selects the RTCK method. If your system uses RTCK, you will not need to change the JTAG clocking after a setup.

Example:

```
openocd -s ../scripts -f interface/jlink.cfg -c "transport select jtag; adapter_khz
2000; shutdown"
```

transport list

Displays the names of the transports supported by this version of OpenOCD.

Example:

```
openocd -c "transport list; shutdown"
```

transport select

Selects which of the supported transports to use in this OpenOCD session.

```
transport select <transport_name>
```

When invoked with *transport_name*, attempts to select the named transport. The transport must be supported by the debug adapter hardware and by the version of OpenOCD you are using (including the adapter's driver). If no transport has been selected and no *transport_name* is provided, `transport select` auto-selects the first transport supported by the debug adapter. `transport select` always returns the name of the session's selected transport, if any.

Example:

```
openocd -s ../scripts -f interface/jlink.cfg -c "transport select jtag"
```

targets

With no parameter, this command displays a table of all known targets in a user-friendly form. With a parameter, this command sets the current target to a given target with a given *name*; this is only relevant to boards with more than one target.

```
targets [name]
```

Examples:

Displays all available targets of the connected PSoC 6 device:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "targets;
shutdown"
```

Selects the CM4 core of the PSoC 6 device as the current target:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "targets
psoc6.cpu.cm4; target current"
```

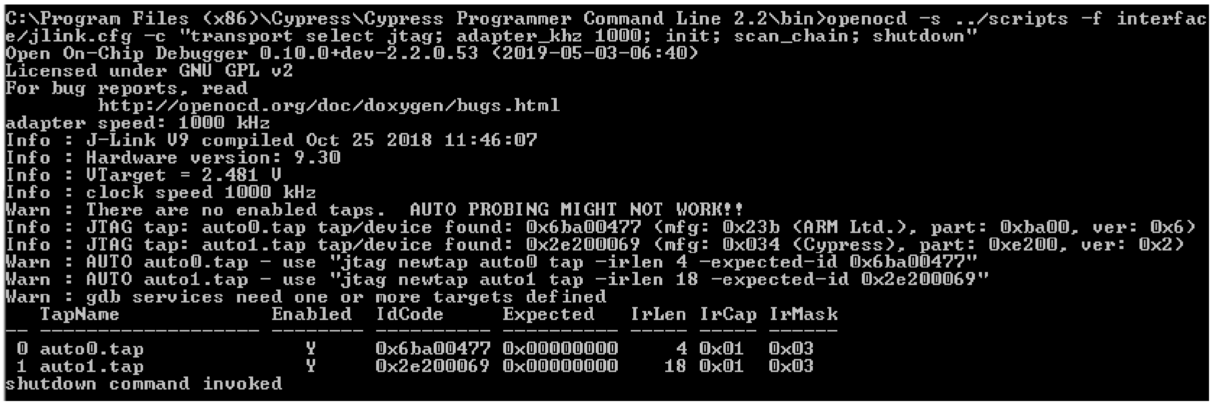
scan_chain

Displays the TAPs in the scan chain configuration, and their status. (Do not confuse this with the list displayed by the `targets` command. That only displays TAPs for CPUs configured as debugging targets.)

Example:

Displays TAPs of the PSoC 6 1M device.

```
openocd -s ../scripts -f interface/jlink.cfg -c "transport select jtag; adapter_khz
1000; init; scan_chain; shutdown"
```



```
C:\Program Files (x86)\Cypress\Cypress Programmer Command Line 2.2\bin>openocd -s ../scripts -f interfac
e/jlink.cfg -c "transport select jtag; adapter_khz 1000; init; scan_chain; shutdown"
Open On-Chip Debugger 0.10.0+dev-2.2.0.53 (2019-05-03-06:40)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1000 kHz
Info : J-Link V9 compiled Oct 25 2018 11:46:07
Info : Hardware version: 9.30
Info : VTarget = 2.481 V
Info : clock speed 1000 kHz
Warn : There are no enabled taps.  AUTO PROBING MIGHT NOT WORK!!
Info : JTAG tap: auto0.tap tap/device found: 0x6ba00477 (mfg: 0x23b (ARM Ltd.), part: 0xba00, ver: 0x6)
Info : JTAG tap: auto1.tap tap/device found: 0x2e200069 (mfg: 0x034 (Cypress), part: 0xe200, ver: 0x2)
Warn : AUTO auto0.tap - use "jtag newtap auto0 tap -irlen 4 -expected-id 0x6ba00477"
Warn : AUTO auto1.tap - use "jtag newtap auto1 tap -irlen 18 -expected-id 0x2e200069"
Warn : gdb services need one or more targets defined
   TapName             Enabled  IdCode     Expected   IrLen IrCap IrMask
-- ------------------- -------- ---------- ---------- ----- ----- ------
 0 auto0.tap              Y     0x6ba00477 0x00000000     4 0x01  0x03
 1 auto1.tap              Y     0x2e200069 0x00000000    18 0x01  0x03
shutdown command invoked
```

md(w)(h)(b)

Displays the contents of address *addr*, as 32-bit words (*mdw*), 16-bit half-words (*mdh*), or 8-bit bytes (*mdb*).

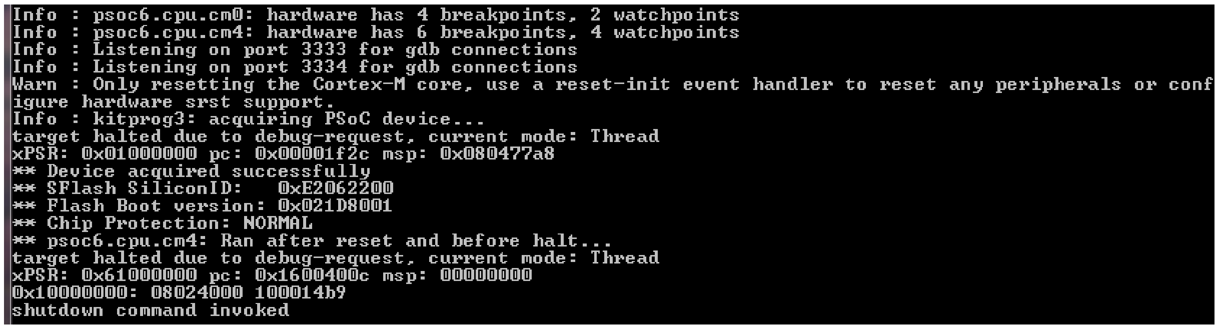
```
mdw [phys] <addr> [count]
mdh [phys] <addr> [count]
mdb [phys] <addr> [count]
```

When the current target has a present and active MMU, *addr* is interpreted as a virtual address. Otherwise, or if the optional *phys* flag is specified, *addr* is interpreted as a physical address. If *count* is specified, displays that many units.

Example:

Displays two 32-bit words of memory of the PSoC6 device.

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; mdw 0x10000000 2; shutdown"
```



```
Info : psoc6.cpu.cm0: hardware has 4 breakpoints, 2 watchpoints
Info : psoc6.cpu.cm4: hardware has 6 breakpoints, 4 watchpoints
Info : Listening on port 3333 for gdb connections
Info : Listening on port 3334 for gdb connections
Warn : Only resetting the Cortex-M core, use a reset-init event handler to reset any peripherals or conf
igure hardware srst support.
Info : kitprog3: acquiring PSoC device...
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x00001f2c msp: 0x080477a8
** Device acquired successfully
** SFlash SiliconID:    0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
0x10000000: 08024000 100014b9
shutdown command invoked
```

mw(w)(h)(b)

Writes the specified *word* (32 bits), *halfword* (16 bits), or *byte* (8-bit) value, at the specified address *addr*.

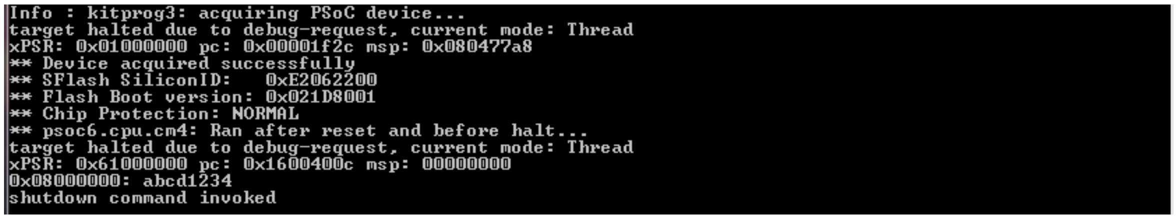
```
mww [phys] <addr> <word>
mwh [phys] <addr> <halfword>
mwb [phys] <addr> <byte>
```

When the current target has a present and active MMU, *addr* is interpreted as a virtual address. Otherwise, or if the optional *phys* flag is specified, *addr* is interpreted as a physical address.

Example:

Write a 32-bit word to the memory of the PSoC6 device.

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; mww 0x8000000 0xABCD1234; mdw 0x8000000; shutdown"
```



```
Info : kitprog3: acquiring PSoC device...
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x00001f2c msp: 0x080477a8
** Device acquired successfully
** SFlash SiliconID:    0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
0x08000000: abcd1234
shutdown command invoked
```

init

This command terminates the configuration stage and enters the run stage. This helps to have the startup scripts manage tasks such as resetting the target, programming flash, etc. To reset the CPU upon a startup, add "init" and "reset" at the end of the config script or at the end of the OpenOCD command line using the -c command line switch.

If this command does not appear in any startup/configuration file, OpenOCD executes the command for you after processing all configuration files and/or command-line options.

Note This command normally occurs at or near the end of your config file to force OpenOCD to initialize and make the targets ready. For example: If your config file needs to read/write memory on your target, initialization must occur before the memory read/write commands.

Example (KitProg3 + PSoC 6 target):

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init;
shutdown"
```

reset [run] [halt] [init]

Performs as hard a reset as possible, using SRST if possible. All defined targets will be reset, and target events will fire during the reset sequence.

The optional parameter specifies what should happen after a reset. If there is no parameter, a reset run is executed. The other options will not work on all systems. See reset_config.

- `run` - Let the target run
- `halt` - Immediately halt the target
- `init` - Immediately halt the target, and execute the reset-init script.

Example:

Reset and Init the KitProg3 + PSoC 6 target:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; shutdown"\
```

program

Programs a given programming file in the HEX, SREC, ELF or BIN formats into the flash of the target device.

```
program <filename> [verify] [reset] [exit] [offset]
```

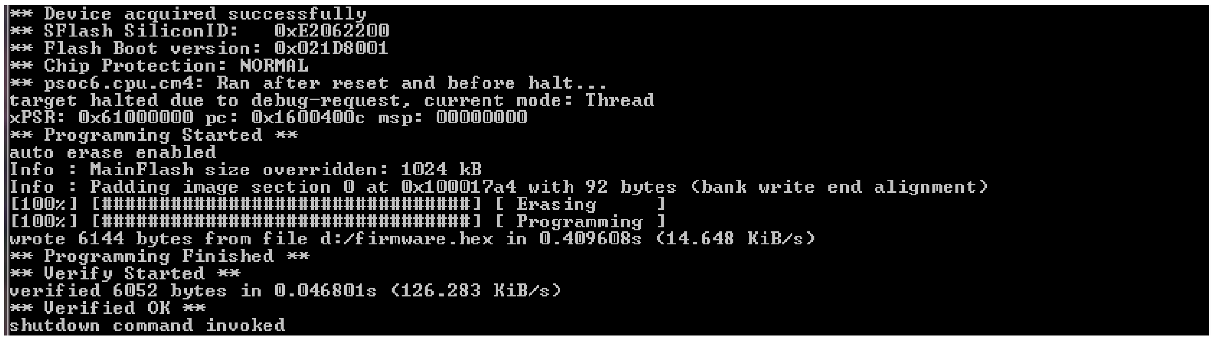
The only required parameter is *filename*, the others are optional.

- `verify` – Compares the contents of the binary file *`filename`* with the contents of the flash.
- `reset` – "reset run" is called if this parameter is given (see reset for details).
- `exit` – OpenOCD is shut down if this parameter is given.
- `offset` – A relocation offset may be specified, then it is added to the base address for each section in the image

Example:

The next example connects Cypress Programmer 2.2. to the KitProg3 probe with the PSoC 6 target device, programs flash with the *firmware.hex* file, verifies programmed data, and finally shuts down the OpenOCD programmer.

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "program
d:/firmware.hex verify exit"
```



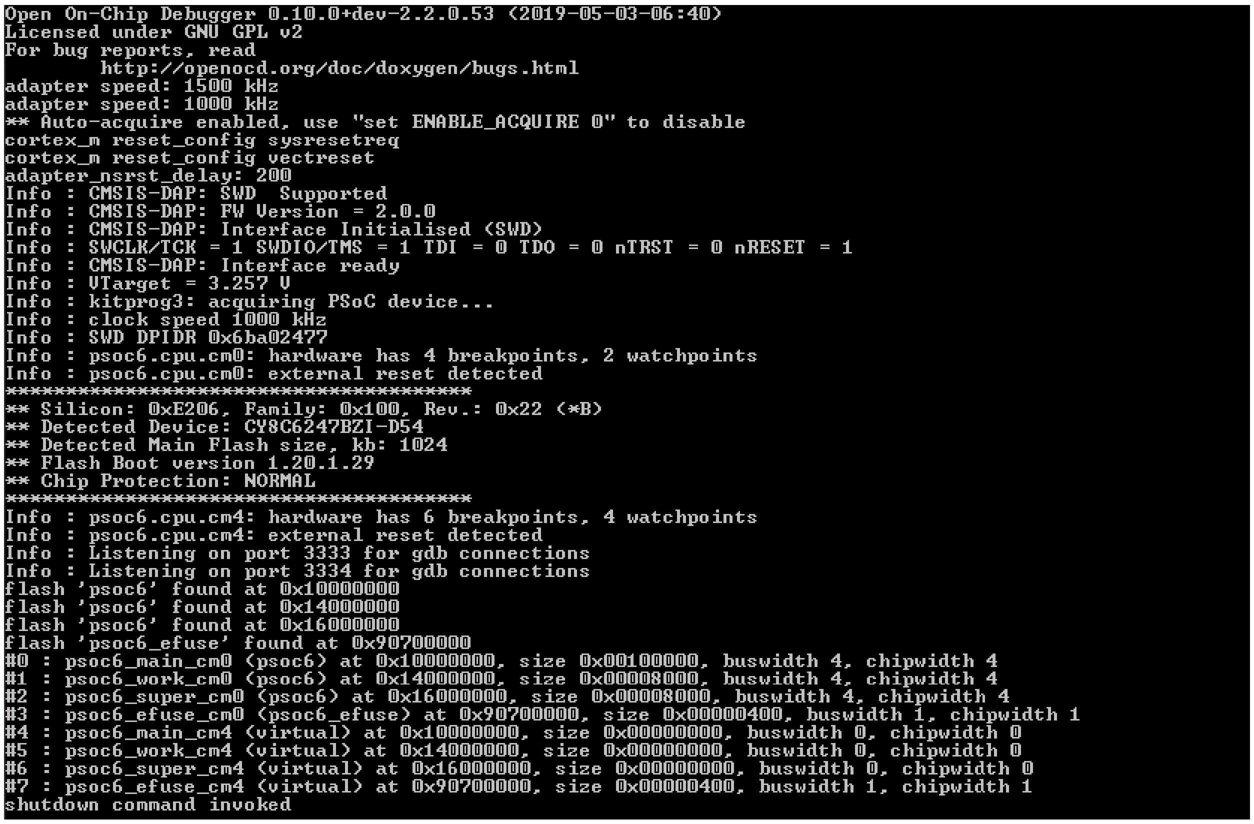
```
** Device acquired successfully
** SFlash SiliconID:    0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
** Programming Started **
auto erase enabled
Info : MainFlash size overridden: 1024 kB
Info : Padding image section 0 at 0x100017a4 with 92 bytes (bank write end alignment)
[100%] [################################] [ Erasing     ]
[100%] [################################] [ Programming ]
wrote 6144 bytes from file d:/firmware.hex in 0.409608s (14.648 KiB/s)
** Programming Finished **
** Verify Started **
verified 6052 bytes in 0.046801s (126.283 KiB/s)
** Verified OK **
shutdown command invoked
```

flash banks

Prints a one-line summary of each flash bank of the target device.

Example (KitProg3 + PSoC 6 device):

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; flash
probe 0; flash probe 1; flash probe 2; flash probe 3; flash banks; shutdown"
```



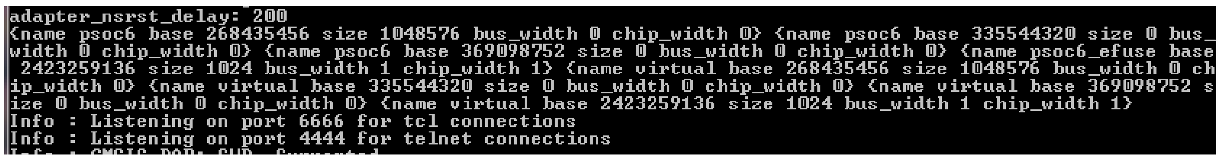
```
Open On-Chip Debugger 0.10.0+dev-2.2.0.53 (2019-05-03-06:40)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
adapter speed: 1000 kHz
** Auto-acquire enabled, use "set ENABLE_ACQUIRE 0" to disable
cortex_m reset_config sysresetreq
cortex_m reset_config vectreset
adapter_nsrst_delay: 200
Info : CMSIS-DAP: SWD  Supported
Info : CMSIS-DAP: FW Version = 2.0.0
Info : CMSIS-DAP: Interface Initialised (SWD)
Info : SWCLK/TCK = 1 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 1
Info : CMSIS-DAP: Interface ready
Info : VTarget = 3.257 V
Info : kitprog3: acquiring PSoC device...
Info : clock speed 1000 kHz
Info : SWD DPIDR 0x6ba02477
Info : psoc6.cpu.cm0: hardware has 4 breakpoints, 2 watchpoints
Info : psoc6.cpu.cm0: external reset detected
***************************************
** Silicon: 0xE206, Family: 0x100, Rev.: 0x22 (*B)
** Detected Device: CY8C6247BZI-D54
** Detected Main Flash size, kb: 1024
** Flash Boot version 1.20.1.29
** Chip Protection: NORMAL
***************************************
Info : psoc6.cpu.cm4: hardware has 6 breakpoints, 4 watchpoints
Info : psoc6.cpu.cm4: external reset detected
Info : Listening on port 3333 for gdb connections
Info : Listening on port 3334 for gdb connections
flash 'psoc6' found at 0x10000000
flash 'psoc6' found at 0x14000000
flash 'psoc6' found at 0x16000000
flash 'psoc6_efuse' found at 0x90700000
#0 : psoc6_main_cm0 (psoc6) at 0x10000000, size 0x00100000, buswidth 4, chipwidth 4
#1 : psoc6_work_cm0 (psoc6) at 0x14000000, size 0x00008000, buswidth 4, chipwidth 4
#2 : psoc6_super_cm0 (psoc6) at 0x16000000, size 0x00008000, buswidth 4, chipwidth 4
#3 : psoc6_efuse_cm0 (psoc6_efuse) at 0x90700000, size 0x00000400, buswidth 1, chipwidth 1
#4 : psoc6_main_cm4 (virtual) at 0x10000000, size 0x00000000, buswidth 0, chipwidth 0
#5 : psoc6_work_cm4 (virtual) at 0x14000000, size 0x00000000, buswidth 0, chipwidth 0
#6 : psoc6_super_cm4 (virtual) at 0x16000000, size 0x00000000, buswidth 0, chipwidth 0
#7 : psoc6_efuse_cm4 (virtual) at 0x90700000, size 0x00000400, buswidth 1, chipwidth 1
shutdown command invoked
```

flash list

Retrieves a list of associative arrays for each device that was declared using a flash bank numbered from zero.

Example (KitProg3 + PSoC 6 device):

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; flash list"
```



```
adapter_nsrst_delay: 200
{name psoc6 base 268435456 size 1048576 bus_width 0 chip_width 0} {name psoc6 base 335544320 size 0 bus_
width 0 chip_width 0} {name psoc6 base 369098752 size 0 bus_width 0 chip_width 0} {name psoc6_efuse base
 2423259136 size 1024 bus_width 1 chip_width 1} {name virtual base 268435456 size 1048576 bus_width 0 ch
ip_width 0} {name virtual base 335544320 size 0 bus_width 0 chip_width 0} {name virtual base 369098752 s
ize 0 bus_width 0 chip_width 0} {name virtual base 2423259136 size 1024 bus_width 1 chip_width 1}
Info : Listening on port 6666 for tcl connections
Info : Listening on port 4444 for telnet connections
```

flash info

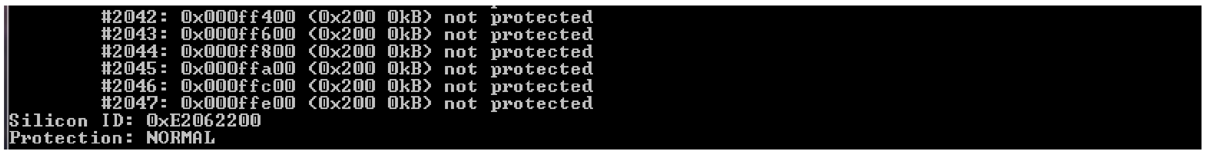
```
flash info <num> [sectors]
```

Prints info about the flash bank *num*, a list of protection blocks and their status. Uses sectors to show a list of sectors instead. The *num* parameter is a value shown by flash banks. This command will first query the hardware, it does not print cached and possibly stale information.

Example:

Prints information about flash bank 0 of the KitProg3 + PSoC 6 device:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; flash info 0; shutdown"
```



```
        #2042: 0x000ff400 (0x200 0kB) not protected
        #2043: 0x000ff600 (0x200 0kB) not protected
        #2044: 0x000ff800 (0x200 0kB) not protected
        #2045: 0x000ffa00 (0x200 0kB) not protected
        #2046: 0x000ffc00 (0x200 0kB) not protected
        #2047: 0x000ffe00 (0x200 0kB) not protected
Silicon ID: 0xE2062200
Protection: NORMAL
```

flash erase_sector

Erase sectors in the bank *num*, starting at Sector *first* up to and including Sector *last*.

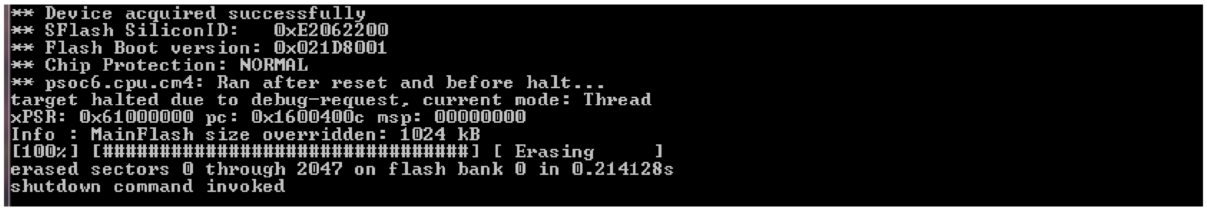
flash erase_sector <*num*> <*first*> <*last*>

The sector numbering starts at 0. Providing the *last* sector of `last` specifies "to the end of the flash bank". The *num* parameter is a value shown by flash banks.

Example:

Erases all sectors in flash bank 0 of the KitProg3 + PSoC 6 device:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; flash erase_sector 0 0 last; shutdown"
```



```
** Device acquired successfully
** SFlash SiliconID:    0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
Info : MainFlash size overridden: 1024 kB
[100%] [##################################] [ Erasing      ]
erased sectors 0 through 2047 on flash bank 0 in 0.214128s
shutdown command invoked
```

flash erase_address

Erases sectors starting at *address* for the *length* bytes.

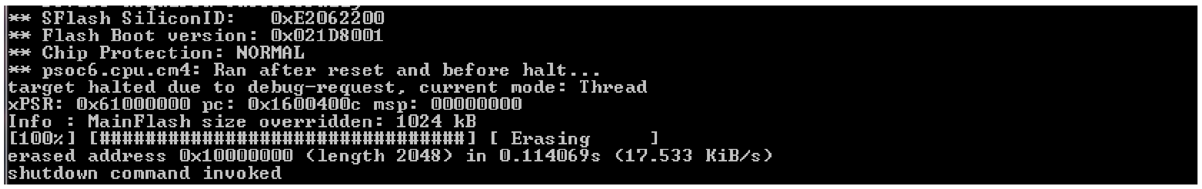
```
flash erase_address [pad] [unlock] <address> <length>
```

Unless *pad* is specified, *address* must begin a flash sector, and *address* + *length* - 1 must end a sector. Specifying *pad* erases extra data at the beginning and/or end of the specified region, as needed to erase only full sectors. The flash bank to use is inferred from the *address*, and the specified *length* must stay within that bank. As a special case, when *length* is zero and *address* is the start of the bank, the whole flash is erased. If *unlock* is specified, then the flash is unprotected before *erase* starts.

Examples:

Erases the 2-KB block starting at address 0x10000000 of the KitProg3 + PSoC 6 device:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; flash erase_address 0x10000000 2048; shutdown"
```



```
** SFlash SiliconID:    0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
Info : MainFlash size overridden: 1024 kB
[100%] [##################################] [ Erasing      ]
erased address 0x10000000 (length 2048) in 0.114069s (17.533 KiB/s)
shutdown command invoked
```

flash write_bank

Writes the binary *filename* to flash bank *num*, starting at *offset* bytes from the beginning

of the bank.

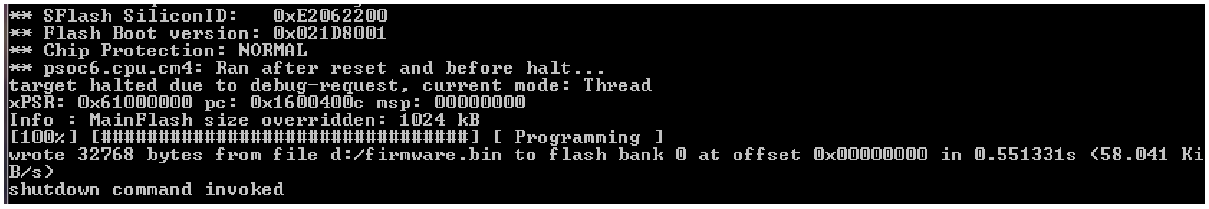
```
flash write_bank <num> <filename> <offset>
```

The *num* parameter is a value shown by flash banks.

Example:

Writes the binary file *firmware.bin* to flash bank 0 of the KitProg3 + PSoC 6 device starting at offset 0:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; flash write_bank 0 d:/firmware.bin 0x0; shutdown"
```



```
** SFlash SiliconID:   0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
Info : MainFlash size overridden: 1024 kB
[100%] [##################################] [ Programming ]
wrote 32768 bytes from file d:/firmware.bin to flash bank 0 at offset 0x00000000 in 0.551331s (58.041 Ki
B/s)
shutdown command invoked
```

flash write_image

Writes the image *filename* to the current target's flash bank(s).

```
flash write_image [erase] [unlock] <filename> [offset] [type]
```

Only loadable sections from the image are written. A relocation *offset* may be specified, then it is added to the base address for each section in the image. The file [*type*] can be specified explicitly as `bin` (binary), `ihex` (Intel hex), `elf` (ELF file), `s19` (Motorola s19). The relevant flash sectors will be erased prior to programming if the *erase* parameter is given. If *unlock* is provided, then the flash banks are unlocked before *erase* and *program*. The flash bank to use is inferred from the address of each image section.

Warning Be careful using the *erase* flag when the flash is holding data you want to preserve. Portions of the flash outside those described in the image's sections might be erased with no notice.

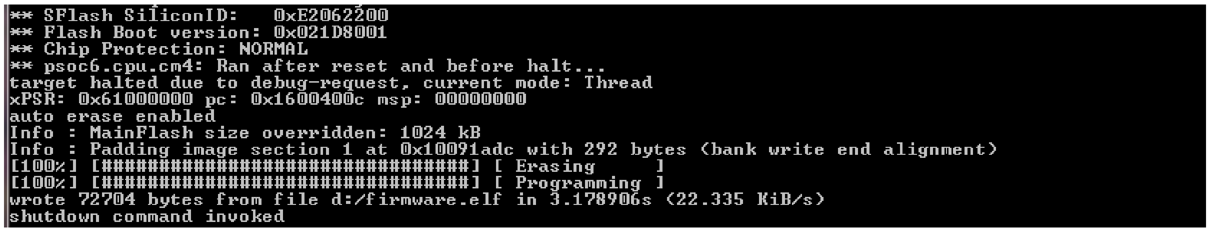
- When a section of the image being written does not fill out all the sectors it uses, the unwritten parts of those sectors are necessarily also erased, because sectors cannot be partially erased.
- Data stored in sector "holes" between image sections are also affected. For example, "`flash write_image erase ...`" of an image with one byte at the beginning of a flash bank and one byte at the end erases the entire bank – not just the two sectors being written.

Also, when flash protection is important, you must re-apply it after it has been removed by the *unlock* flag.

Examples:

Writes the ELF image *firmware.elf* to the KitProg3 + PSoC 6 device:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; flash write_image erase d:/firmware.elf; shutdown"
```



```
** SFlash SiliconID:   0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
auto erase enabled
Info : MainFlash size overridden: 1024 kB
Info : Padding image section 1 at 0x10091adc with 292 bytes (bank write end alignment)
[100%] [##################################] [ Erasing     ]
[100%] [##################################] [ Programming ]
wrote 72704 bytes from file d:/firmware.elf in 3.178906s (22.335 KiB/s)
shutdown command invoked
```

flash fill(w)(h)(b)

Fills flash memory with the specified word (32 bits), half-word (16 bits), or byte (8-bit) pattern, starting at *address* and continuing for *length* units (word/half-word/byte).

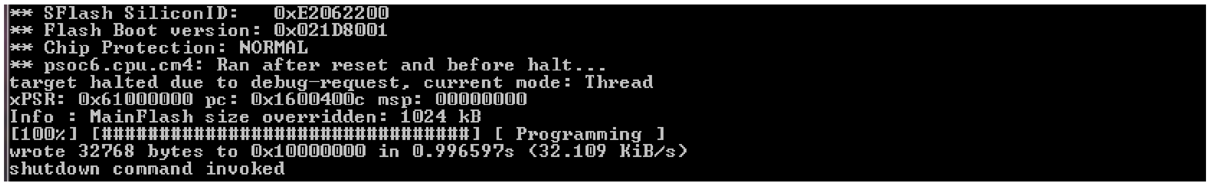
```
flash fillw <address> <word> <length>
flash fillh <address> <halfword> <length>
flash fillb <address> <byte> <length>
```

No *erase* is done before writing; when needed, that must be done before issuing this command. Writes are done in blocks of up to 1024 bytes, and each *write* is verified by reading back the data and comparing it to what was written. The flash bank to use is inferred from the address of each block, and the specified length must stay within that bank.

Example:

Fills the 32-KB block of the PSoC 6 device memory starting at address 0x10000000 with the pattern 0x5A:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; flash fillw 0x10000000 0x5A5A5A5A 0x2000; shutdown"
```



```
** SFlash SiliconID:   0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
Info : MainFlash size overridden: 1024 kB
[100%] [################################] [ Programming ]
wrote 32768 bytes to 0x10000000 in 0.996597s (32.109 KiB/s)
shutdown command invoked
```

flash read_bank

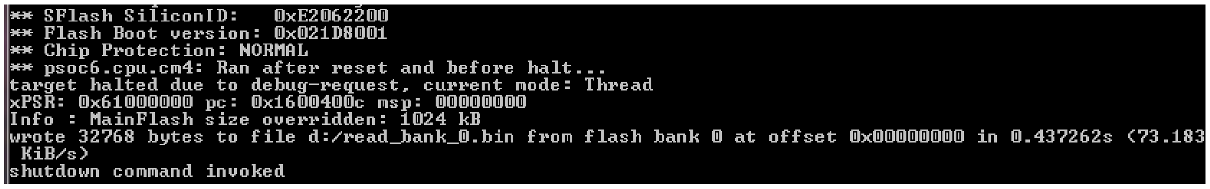
Reads the *length* bytes from the flash bank *num* starting at *offset* and writes the contents to the binary *filename*. The *num* parameter is a value shown by flash banks.

```
flash read_bank <num> <filename> <offset> <length>
```

Examples:

Reads the 32-KB block of bank #0 from the PSoC 6 device memory and writes it to the binary file:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; flash read_bank 0 d:/read_bank_0.bin 0x0 0x8000; shutdown"
```



```
** SFlash SiliconID:   0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
Info : MainFlash size overridden: 1024 kB
wrote 32768 bytes to file d:/read_bank_0.bin from flash bank 0 at offset 0x00000000 in 0.437262s (73.183
 KiB/s)
shutdown command invoked
```

flash verify_bank

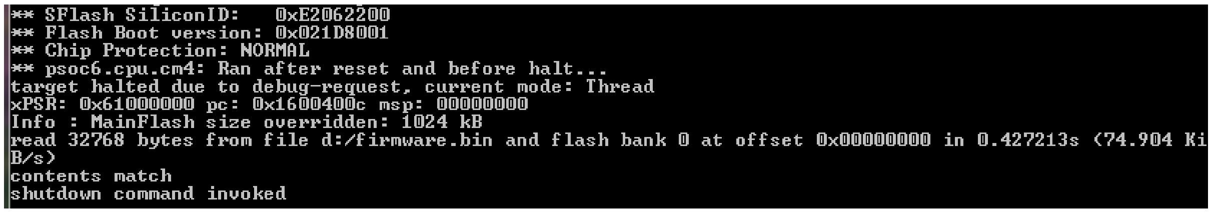
Compares the contents of the binary file *filename* with the contents of the flash *num* starting at *offset*. Fails if the contents do not match. The *num* parameter is a value shown by flash banks.

```
flash verify_bank <num> <filename> <offset>
```

Example:

Verifies the content of bank #0 of the PSoC 6 device:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; flash verify_bank 0 d:/firmware.bin 0x0; shutdown"
```



```
** SFlash SiliconID:   0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
Info : MainFlash size overridden: 1024 kB
read 32768 bytes from file d:/firmware.bin and flash bank 0 at offset 0x00000000 in 0.427213s (74.904 Ki
B/s)
contents match
shutdown command invoked
```

flash padded_value

Sets the default value used for padding-any-image sections.

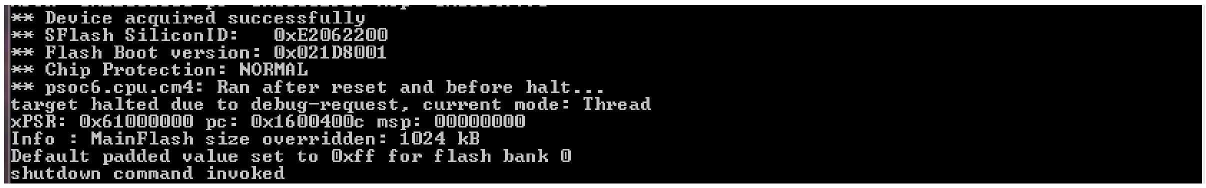
```
flash padded_value <num> <value>
```

This should normally match the flash bank erased value. If not specified by this command or the flash driver, then it defaults to 0xff.

Example:

Sets a padded value to 0xFF for bank #0 of the PSoC 6 device.

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; flash padded_value 0 0xFF; shutdown"
```



```
** Device acquired successfully
** SFlash SiliconID:   0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
Info : MainFlash size overridden: 1024 kB
Default padded value set to 0xff for flash bank 0
shutdown command invoked
```

flash rmw

The command is intended to modify flash individual bytes.

```
flash rmw <address> <data>
```

The command can be used to program data to an arbitrary flash address preserving all data that belongs to the same flash sector.

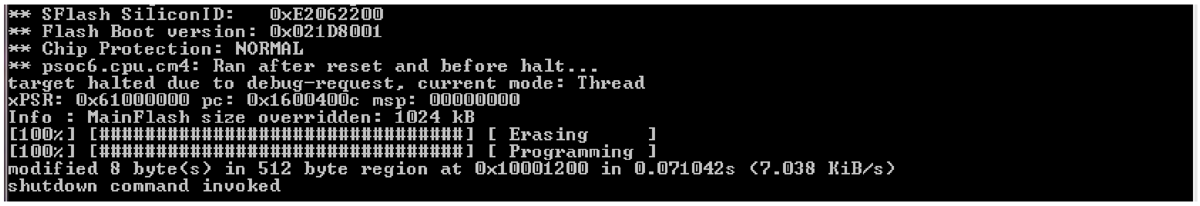
- address – The start address for the programming.
- data – The hexadecimal string with data to be programmed. The format of the string is shown in the following example:

Note `flash rmw` is a custom command implemented in OpenOCD to extend its functionality.

Examples:

Modifies 8 bytes of the PSoC 6 device flash at address 0x10001234.

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; flash rmw 0x10001234 DEADBEEFBAADC0DE; shutdown"
```



```
** SFlash SiliconID:    0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 000000000
Info : MainFlash size overridden: 1024 kB
[100%] [################################] [ Erasing     ]
[100%] [################################] [ Programming ]
modified 8 byte(s) in 512 byte region at 0x10001200 in 0.071042s (7.038 KiB/s)
shutdown command invoked
```

add_verify_range

The command allows specifying memory regions to be compared during *verify* operation.

```
add_verify_range <target> <address> <size>
```

By default, when no regions are defined, all the regions present in the firmware image file are compared with corresponding target memory. This breaks the verification process for some non-memory-mapped regions such as EFuses. When the target has at least one *verify* region specified, only data that belongs to that *verify* region is verified.

- target – The target device to assign *verify* regions.
- address – The start address of the region.
- size – The size of the region, in bytes.

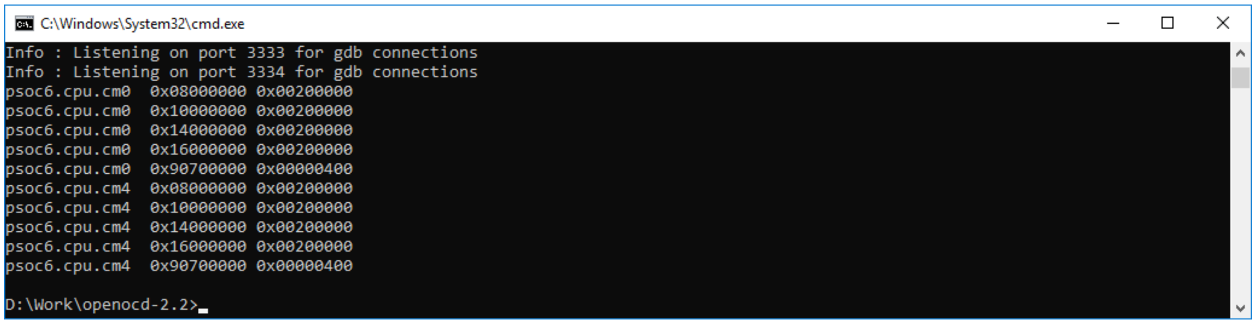
Note The `add_verify_range` command is a custom command implemented in OpenOCD to extend its functionality.

show_verify_ranges

This command displays all active verify ranges for all targets that were added using the *add_verify_range* command. This command does not take any arguments.

Example output:

```
bin\openocd.exe -s scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init;
show_verify_ranges; exit"
```



```
Info : Listening on port 3333 for gdb connections
Info : Listening on port 3334 for gdb connections
psoc6.cpu.cm0  0x08000000 0x00200000
psoc6.cpu.cm0  0x10000000 0x00200000
psoc6.cpu.cm0  0x14000000 0x00200000
psoc6.cpu.cm0  0x16000000 0x00200000
psoc6.cpu.cm0  0x90700000 0x00000400
psoc6.cpu.cm4  0x08000000 0x00200000
psoc6.cpu.cm4  0x10000000 0x00200000
psoc6.cpu.cm4  0x14000000 0x00200000
psoc6.cpu.cm4  0x16000000 0x00200000
psoc6.cpu.cm4  0x90700000 0x00000400

D:\Work\openocd-2.2>
```

Note The `show_verify_ranges` command is a custom command implemented in OpenOCD to extend its functionality.

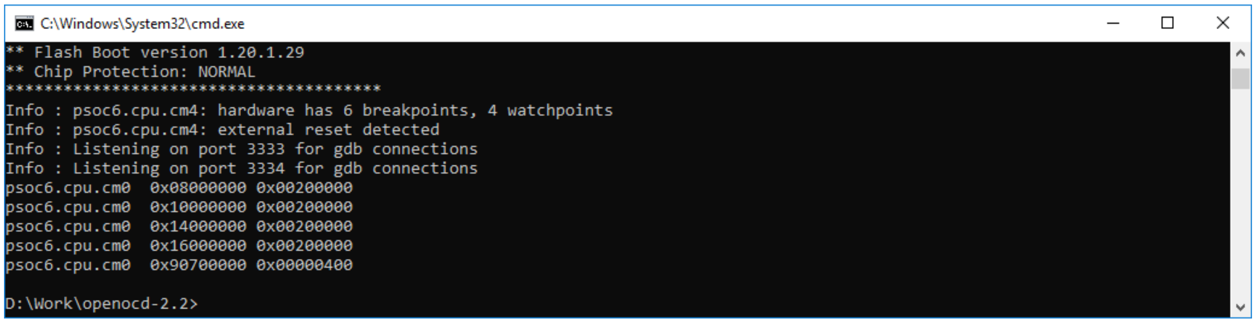
clear_verify_ranges

This command deletes all verify ranges for the specified target that were added using the *add_verify_range* command.

```
clear_verify_ranges <target>
```

Example output:

```
bin\openocd.exe -s scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init;
clear_verify_ranges psoc6.cpu.cm4; show_verify_ranges; exit"
```



C:\Windows\System32\cmd.exe

```
** Flash Boot version 1.20.1.29
** Chip Protection: NORMAL
****************************************
Info : psoc6.cpu.cm4: hardware has 6 breakpoints, 4 watchpoints
Info : psoc6.cpu.cm4: external reset detected
Info : Listening on port 3333 for gdb connections
Info : Listening on port 3334 for gdb connections
psoc6.cpu.cm0  0x08000000 0x00200000
psoc6.cpu.cm0  0x10000000 0x00200000
psoc6.cpu.cm0  0x14000000 0x00200000
psoc6.cpu.cm0  0x16000000 0x00200000
psoc6.cpu.cm0  0x90700000 0x00000400

D:\Work\openocd-2.2>
```

Note The `clear_verify_ranges` command is a custom command implemented in OpenOCD to extend its functionality.

verify_image

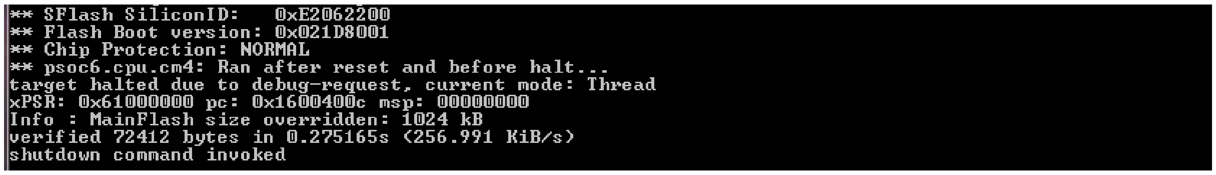
Verifies *filename* against target memory starting at *address*. The file format may optionally be specified (bin, ihex, or elf). This will first attempt a comparison using a CRC checksum, if that fails, it will try a binary compare.

```
verify_image <filename> <address> [bin|ihex|elf]
```

Examples:

Verifies a *firmware.elf* image against the target memory the PSoC 6 device.

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; verify_image d:/firmware.elf 0x0; shutdown"
```



```
** SFlash SiliconID:   0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
Info : MainFlash size overridden: 1024 kB
verified 72412 bytes in 0.275165s (256.991 KiB/s)
shutdown command invoked
```

verify_image_checksum

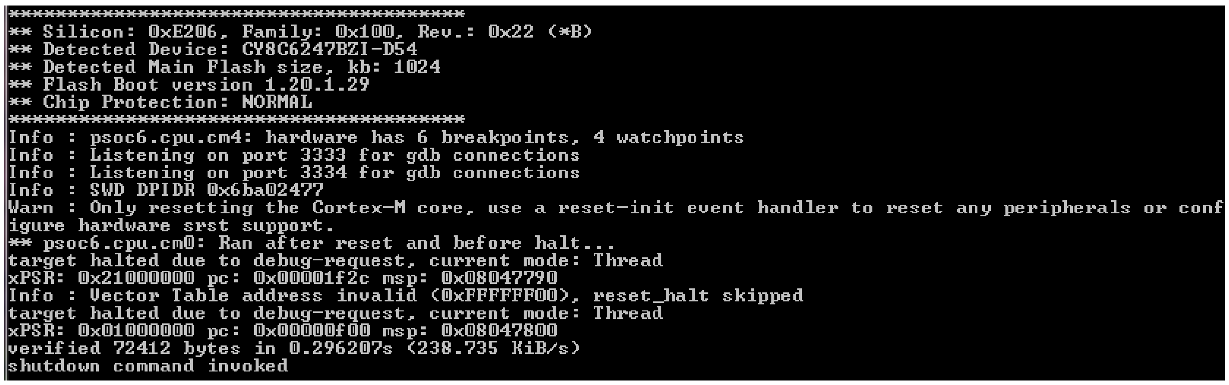
Verifies *filename* against the target memory starting at *address*. The file format may optionally be specified (bin, ihex, or elf). This perform a comparison using a CRC checksum only.

```
verify_image_checksum <filename> <address> [bin|ihex|elf]
```

Example:

Verifies a *firmware.elf* image against the target memory of the PSoC 6 1M device using the CRC check only.

```
openocd -s ../scripts -f interface/jlink.cfg -c "transport select swd" -f
target/psoc6.cfg -c "init; reset init; verify_image_checksum d:/firmware.elf 0x0;
shutdown"
```



```
***************************************
** Silicon: 0xE206, Family: 0x100, Rev.: 0x22 (*B)
** Detected Device: CY8C6247BZI-D54
** Detected Main Flash size, kb: 1024
** Flash Boot version 1.20.1.29
** Chip Protection: NORMAL
***************************************
Info : psoc6.cpu.cm4: hardware has 6 breakpoints, 4 watchpoints
Info : Listening on port 3333 for gdb connections
Info : Listening on port 3334 for gdb connections
Info : SWD DPIDR 0x6ba02477
Warn : Only resetting the Cortex-M core, use a reset-init event handler to reset any peripherals or conf
igure hardware srst support.
** psoc6.cpu.cm0: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x21000000 pc: 0x00001f2c msp: 0x08047790
Info : Vector Table address invalid (0xFFFFFF00), reset_halt skipped
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x00000f00 msp: 0x08047800
verified 72412 bytes in 0.296207s (238.735 KiB/s)
shutdown command invoked
```

load_image

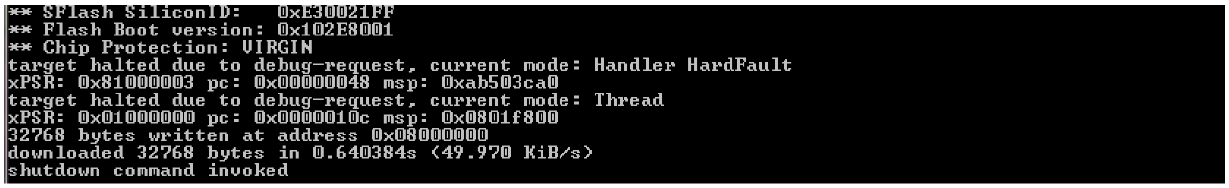
Loads an image from file *filename* to the target memory offset by *address* from its load address. The file format may optionally be specified (bin, ihex, elf, or s19). Also, the following arguments may be specified: *min_addr* - ignore data below *min_addr* (this is w.r.t. to the target's load address + address) *max_length* - maximum number of bytes to load.

```
load_image filename address [[bin|ihex|elf|s19] min_addr max_length]
```

Examples:

Loads binary file *firmware.bin* to the RAM of the PSoC 6 device.

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; load_image d:/firmware.bin 0x8000000; shutdown"
```



```
** SFlash SiliconID:    0xE30021FF
** Flash Boot version: 0x102E8001
** Chip Protection: VIRGIN
target halted due to debug-request, current mode: Handler HardFault
xPSR: 0x81000003 pc: 0x00000048 msp: 0xab503ca0
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x0000010c msp: 0x0801f800
32768 bytes written at address 0x08000000
downloaded 32768 bytes in 0.640384s (49.970 KiB/s)
shutdown command invoked
```

dump_image

Dumps *size* bytes of the target memory starting at *address* to the binary file named *filename*.

```
dump_image <filename> <address> size
```

Example:

Dumps 8KB of the PSoC 6 device memory to file *dump_mem.bin*.

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; dump_image d:/dump_mem.bin 0x10001234 0x2000; shutdown"
```

KitProg3/MiniProg4 Driver Commands

The KitProg3/MiniProg4 (KP3/MP4) probe implements the CMSIS-DAP protocol defined by Arm with some extensions. Consequently, the KP3/MP4 driver in OpenOCD is a wrapper around the native CMSIS-DAP driver that extends its functionality with the KP3-specific extensions.

A full list of the CMSIS-DAP-specific configuration commands can be found in the OpenOCD official documentation.

Besides the standard CMSIS-DAP options, the KP3 driver exposes several custom Tcl configuration commands. All commands in this section must be prefixed with the name of the driver – “kitprog3”.

kitprog3 acquire_config

The command controls device acquisition parameters and optionally enables acquisition during the early initialization phase. Can be called at any time.

```
kitprog3 acquire_config <status> [target_type] [mode] [attempts]
```

- `status` – A mandatory parameter, enables or disables the acquisition procedure during the initialization phase. The possible values: On, Off.
- `target_type` – Specifies the target device type. This parameter is mandatory only if status=on. The possible values:
  - 0 – PSoC4
  - 1 – PSoC5
  - 2 – PSoC6
- `mode` – Specifies Acquisition mode. This parameter is mandatory only if status=on. The possible values: 0 – Reset, 1 – Power Cycle.
- `attempts` – The number of attempts to acquire the target device. This parameter is mandatory only if status=on.

Example:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "kitprog3
acquire_config on 2 0; init; reset init; shutdown"
```

kitprog3 *acquire_psoc*

Performs device acquisition and is called only after the initialization phase. The acquisition procedure must be configured using acquire_config prior to calling this command.

Example:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "kitprog3
acquire_config on 2 0; init; kitprog3 acquire_psoc; reset init; shutdown"
```

kitprog3 power_config

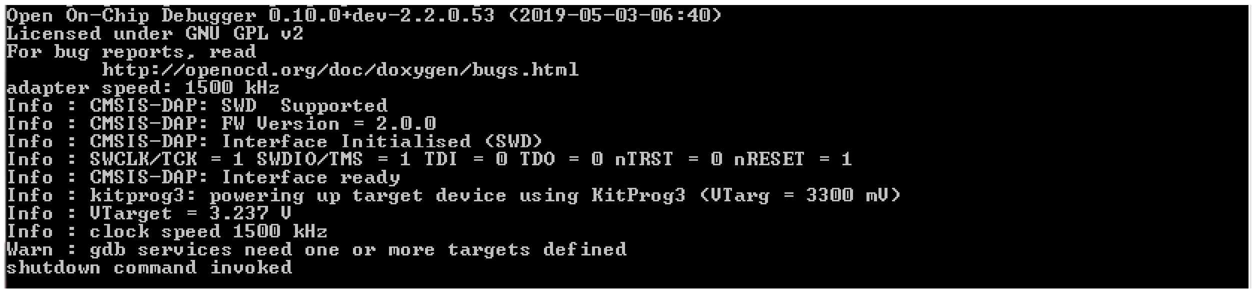
Controls the KP3 internal power supply parameters and optionally enables power during the early initialization phase. Can be called at any time.

```
kitprog3 power_config <status> [voltage]
```

- `status` – A mandatory parameter, enables or disables power supply during the initialization phase. The possible values: `on|off`.
- `voltage` – The power supply voltage in millivolts. This parameter is mandatory only if status=on.

Example:

```
openocd -s ../scripts -f interface/kitprog3.cfg -c "kitprog3 power_config on 3300;
init; shutdown"
```



```
Open On-Chip Debugger 0.10.0+dev-2.2.0.53 (2019-05-03-06:40)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
Info : CMSIS-DAP: SWD  Supported
Info : CMSIS-DAP: FW Version = 2.0.0
Info : CMSIS-DAP: Interface Initialised (SWD)
Info : SWCLK/TCK = 1 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 1
Info : CMSIS-DAP: Interface ready
Info : kitprog3: powering up target device using KitProg3 (VTarg = 3300 mV)
Info : VTarget = 3.237 V
Info : clock speed 1500 kHz
Warn : gdb services need one or more targets defined
shutdown command invoked
```

kitprog3 power_control

The command turns on or off the KP3 internal power supply. Can be called only after the initialization phase.

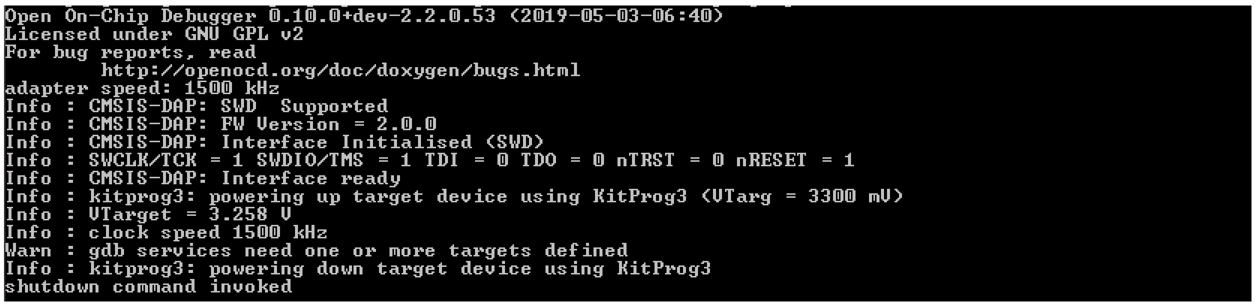
```
kitprog3 power_control <status>
```

The voltage must be configured using `power_config` prior to calling this command.

- `status` – A mandatory parameter, enables or disables power supply: `on|off`.

Example:

```
openocd -s ../scripts -f interface/kitprog3.cfg -c "kitprog3 power_config on 3300;
init; kitprog3 power_control off; shutdown"
```


```
Open On-Chip Debugger 0.10.0+dev-2.2.0.53 (2019-05-03-06:40)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
Info : CMSIS-DAP: SWD  Supported
Info : CMSIS-DAP: FW Version = 2.0.0
Info : CMSIS-DAP: Interface Initialised (SWD)
Info : SWCLK/TCK = 1 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 1
Info : CMSIS-DAP: Interface ready
Info : kitprog3: powering up target device using KitProg3 (VTarg = 3300 mV)
Info : VTarget = 3.258 V
Info : clock speed 1500 kHz
Warn : gdb services need one or more targets defined
Info : kitprog3: powering down target device using KitProg3
shutdown command invoked
```

kitprog3 led_control

Controls the KP3 LEDs. Can be called only after the initialization phase.

```
kitprog3 led_control <type>
```

- `type` – A mandatory parameter, specifies the type of the LED indication. The possible values:
  - 0 – READY
  - 1 – PROGRAMMING
  - 2 – SUCCESS
  - 3 – ERROR

Example:

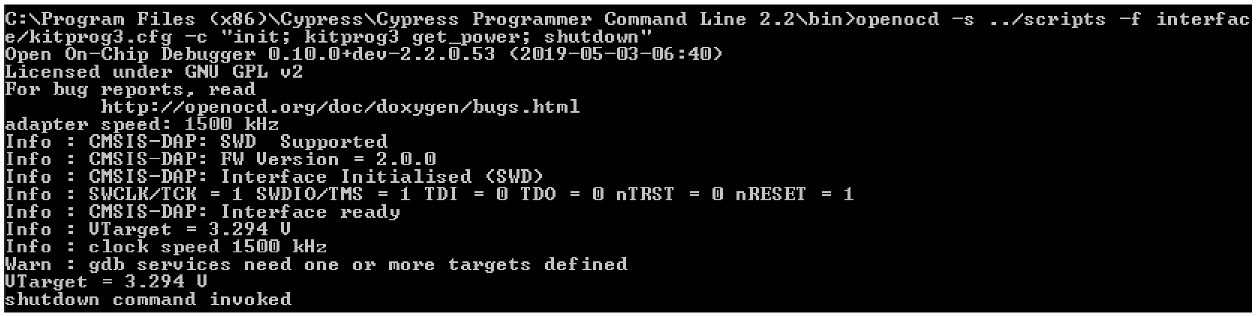
```
openocd -s ../scripts -f interface/kitprog3.cfg -c "init; kitprog3 led_control 2"
```

kitprog3 get_power

Reports the target voltage in millivolts. Can be called only after the initialization phase.

Example:

```
openocd -s ../scripts -f interface/kitprog3.cfg -c "init; kitprog3 get_power;
shutdown"
```


```
C:\Program Files (x86)\Cypress\Cypress Programmer Command Line 2.2\bin>openocd -s ../scripts -f interfac
e/kitprog3.cfg -c "init; kitprog3 get_power; shutdown"
Open On-Chip Debugger 0.10.0+dev-2.2.0.53 (2019-05-03-06:40)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
Info : CMSIS-DAP: SWD  Supported
Info : CMSIS-DAP: FW Version = 2.0.0
Info : CMSIS-DAP: Interface Initialised (SWD)
Info : SWCLK/TCK = 1 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 1
Info : CMSIS-DAP: Interface ready
Info : VTarget = 3.294 V
Info : clock speed 1500 kHz
Warn : gdb services need one or more targets defined
VTarget = 3.294 V
shutdown command invoked
```

Flash Driver Commands

This section contains flash driver commands for PSoC 6 devices.

psoc6 sflash_restrictions

The command enables or disables writes to Sflash regions other than USER, NAR, TOC2, and KEY.

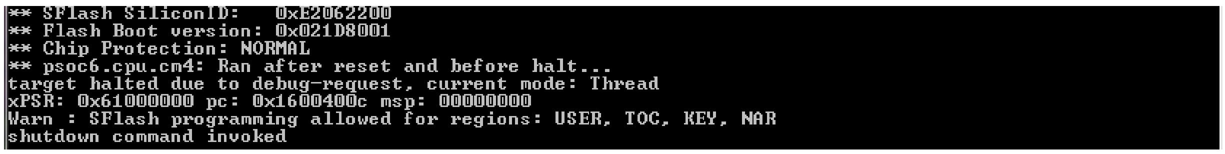
```
psoc6 sflash_restrictions <mode>
```

The command can be called at any time. Writes to these regions are possible only on the VIRGIN silicon, so the command is mostly intended for internal use. It is useful for flash boot developers and validation teams. Note that *erase* (performed by programming with zeros for PSoC 6) is performed only for the USER, NAR, TOC2, and KEY regions; it is skipped for other Sflash regions regardless of this command.

- `mode` – A mandatory parameter, specifies the behavior of Sflash programming. The possible values:
  - 0 – Erase/Program of Sflash is prohibited
  - 1 – Erase and Program of USER/TOC/KEY is allowed
  - 2 – Erase of USER/TOC/KEY and Program of USER/TOC/KEY/NAR is allowed
  - 3 – Erase of USER/TOC/KEY and Program of the whole Sflash region is allowed.

Example (KitProg3 + PSoC6 device):

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; psoc6 sflash_restrictions 2; shutdown"
```



```
** SFlash SiliconID:    0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
Warn : SFlash programming allowed for regions: USER, TOC, KEY, NAR
shutdown command invoked
```

psoc6 allow_efuse_program

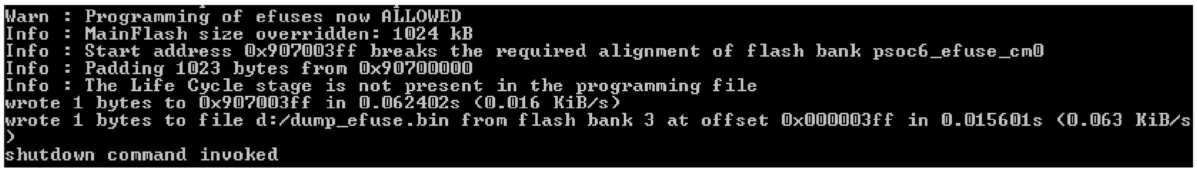
Allows or disallows writes to the EFuse region. Can be called any time. Writes to the EFuse region are skipped by default. EFuse programming must be allowed for Life Cycle transitions to work.

```
psoc6 allow_efuse_program <on|off>
```

Example:

Writes 1 bit to the EFuse region at address 0x907003FF of the PSoC6 device:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; psoc6 allow_efuse_program on; flash fillb 0x907003FF 1 1; flash read_bank 3
d:/dump_efuse.bin 0x3FF 0x1; shutdown"
```



```
Warn : Programming of efuses now ALLOWED
Info : MainFlash size overridden: 1024 kB
Info : Start address 0x907003ff breaks the required alignment of flash bank psoc6_efuse_cm0
Info : Padding 1023 bytes from 0x90700000
Info : The Life Cycle stage is not present in the programming file
wrote 1 bytes to 0x907003ff in 0.062402s (0.016 KiB/s)
wrote 1 bytes to file d:/dump_efuse.bin from flash bank 3 at offset 0x000003ff in 0.015601s (0.063 KiB/s
)
shutdown command invoked
```

psoc6 reset_halt

The command simulates a broken Vector Catch on PSoC6 devices.

```
psoc6 reset_halt <mode>
```

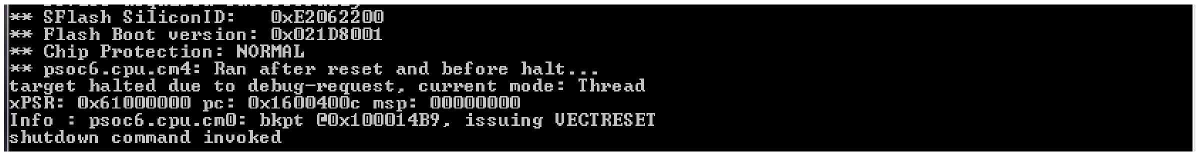
The command retrieves the address of the Vector Table from the VECTOR_TABLE_BASE registers, detects the location of the application entry points, sets a hardware breakpoint at that location and performs a reset of the target. The type of the reset can be specified by the optional <mode> parameter.

Parameters:

- `mode` – The type of a reset to be performed. The possible values are `sysresetreq` and `vectreset`. This parameter is optional. If it is not specified, SYSRESETREQ is used for the CM0 core and VECTRESET is used for other cores in the system.

Example (KitProg3 + PSoC 6 device):

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; psoc6 reset_halt vectreset; shutdown"
```



```
** SFlash SiliconID:   0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
Info : psoc6.cpu.cm0: bkpt @0x100014B9, issuing VECTRESET
shutdown command invoked
```

psoc6 secure_app

Enables or Disables workarounds for the secure family of PSoC 6 devices.

```
psoc6 secure_app <on|off>
```

Currently this command, if enabled, prevents OpenOCD from accessing IPC_INTR_STRUCT register which is protected by secure firmware.

Other Commands

source

Reads a file and executes it as a script. It is usually used with the result of the `find` command.

```
source [find FILENAME]
```

Example (Kitprog3 + PSoC 6 target):

```
openocd -s ../scripts -c "source [find interface/kitprog3.cfg]; source [find
target/psoc6.cfg]; targets; shutdown"
```

find

Finds and returns a full path to a file with a given name. It is usually used as an argument of the `source` command. This command uses an internal search path. (Do not try to use a filename which includes the "#" character. That character begins Tcl comments.)

```
source [find FILENAME]
```

Example:

```
openocd -s ../scripts -c "source [find interface/kitprog3.cfg]; source [find
target/psoc6.cfg]; targets; shutdown"
```

set

Stores a value to a named variable, first creating the variable if it does not already exist.

```
set VARNAME value
```

Example:

```
openocd -s ../scripts -c "set ENABLE_CM0 0; source [find interface/kitprog3.cfg];
source [find target/psoc6.cfg]; targets; shutdown"
```

sleep

Waits for at least `msec` milliseconds before resuming. Useful in a combination with script files.

```
sleep msec
```

Example:

```
openocd -c "sleep 1000; shutdown"
```

add_script_search_dir

Adds a directory to a file/script search path. Equivalent to the `--search` command-line option.

```
add_script_search_dir [directory]
```

Example:

```
openocd -c "add_script_search_dir ../scripts; source [find interface/kitprog3.cfg];
source [find target/psoc6.cfg]; targets; shutdown"
```

8 Global Variables

The global variables listed below control the behavior of a target configuration file (e.g. psoc6.cfg). They are set in the command line prior to any configuration file, such as kitprog3.cfg or psoc6.cfg. See the command set for details.

ENABLE_ACQUIRE

Enables or disables acquisition of the target device in Test mode.

The possible values:

- 1 – Acquisition enabled (default with KitProg3/MiniProg4).
- 0 – Acquisition disabled (default for other debug adapters).

ENABLE_POWER_SUPPLY

Controls internal power supply of KitProg3/MiniProg4 adapters. If this command is specified, the KitProg3 driver, enables power supply thus powering on the target during initialization.

The possible values:

- 0 – Power supply disabled.
- Any other value defines target voltage in millivolts.

ENABLE_CM0, ENABLE_CM4

Allows specifying CPU cores to be visible to OpenOCD. OpenOCD never touches disabled cores.

The possible values:

- 1 – Corresponding core is enabled.
- 0 – Core is disabled.

SMIF_BANKS

Defines QSPI Memory banks. This variable is a two-dimensional associative Tcl array of the following format:

```
set SMIF_BANKS {
  1 {addr <XIPaddr1> size <BankSz1> psize <ProgramSz1> esize <EraseSz1>}
  2 {addr <XIPaddr2> size <BankSz2> psize <ProgramSz2> esize <EraseSz2>}
  ...
  N {addr <XIPaddrN> size <BankSzN> psize <ProgramSzN> esize <EraseSzN>}
}
```

Where:

- XIPaddrN – The XIP mapping address.
- BankSzN – The total size of this flash bank, in bytes.
- ProgramSzN – The minimal programming granularity (program block size), in bytes.
- EraseSzN – The minimal erase granularity (erase block size), in bytes.

9 Error Codes

The OpenOCD tool returns 0 as an error code on successful completion, on a failure it returns 1.

10 Usage Examples

All the examples in this chapter assume you have a PSoC 6 target device connected to the PC via the KitProg3/MiniProg4 or J-Link debug probe. The current working directory is the default install directory (for example, `c:\Program Files (x86)\Cypress\Cypress Programmer Command Line 2.2\bin` on Windows).

For convenience, the *psoc6_kp3_board.cfg* config file has been created in the same directory as the OpenOCD executable. The file contains default configuration suitable for the majority of the PSoC 6 kits:

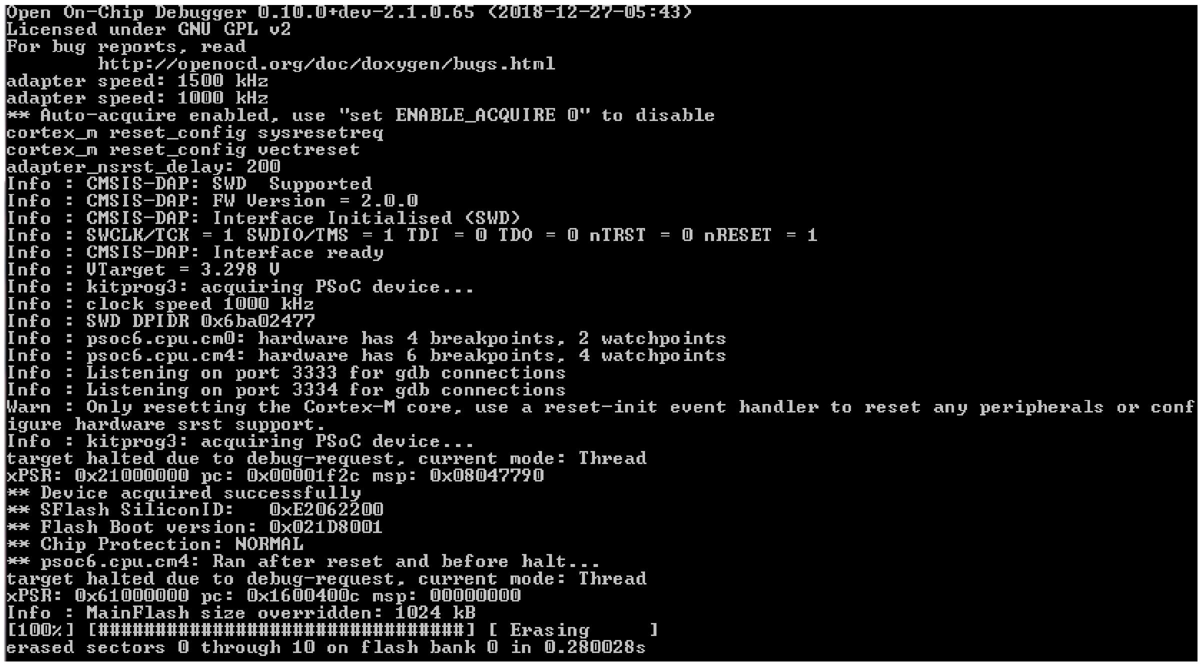
```
source [ find interface/kitprog3.cfg ]
source [ find target/psoc6.cfg ]
init
reset init
```

See Supported Target Configurations for the detailed list of available target devices and corresponding OpenOCD configuration files.

Erase Main Flash Rows 0…10 of PSoC 6 device

```
openocd -s ../scripts -f psoc6_kp3_board.cfg -c "flash erase_sector 0 0 10; exit"
```

A possible output of OpenOCD:

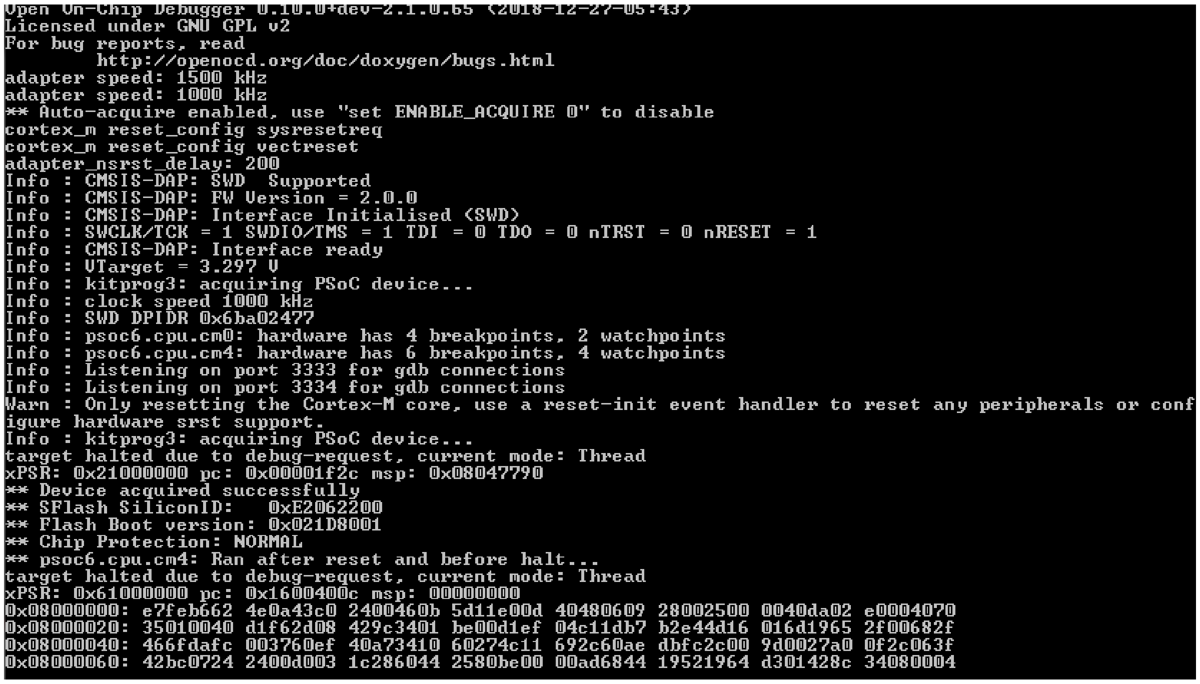


```
Open On-Chip Debugger 0.10.0+dev-2.1.0.65 (2018-12-27-05:43)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
adapter speed: 1000 kHz
** Auto-acquire enabled, use "set ENABLE_ACQUIRE 0" to disable
cortex_m reset_config sysresetreq
cortex_m reset_config vectreset
adapter_nsrst_delay: 200
Info : CMSIS-DAP: SWD  Supported
Info : CMSIS-DAP: FW Version = 2.0.0
Info : CMSIS-DAP: Interface Initialised (SWD)
Info : SWCLK/TCK = 1 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 1
Info : CMSIS-DAP: Interface ready
Info : VTarget = 3.298 V
Info : kitprog3: acquiring PSoC device...
Info : clock speed 1000 kHz
Info : SWD DPIDR 0x6ba02477
Info : psoc6.cpu.cm0: hardware has 4 breakpoints, 2 watchpoints
Info : psoc6.cpu.cm4: hardware has 6 breakpoints, 4 watchpoints
Info : Listening on port 3333 for gdb connections
Info : Listening on port 3334 for gdb connections
Warn : Only resetting the Cortex-M core, use a reset-init event handler to reset any peripherals or conf
igure hardware srst support.
Info : kitprog3: acquiring PSoC device...
target halted due to debug-request, current mode: Thread
xPSR: 0x21000000 pc: 0x00001f2c msp: 0x08047790
** Device acquired successfully
** SFlash SiliconID:    0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
Info : MainFlash size overridden: 1024 kB
[100%] [################################] [ Erasing     ]
erased sectors 0 through 10 on flash bank 0 in 0.280028s
```

Display Memory Contents (32 words at address 0x08000000) of PSoC 6 device

```
openocd -s ../scripts -f psoc6_kp3_board.cfg -c "mdw 0x08000000 32; exit"
```

A possible output of OpenOCD:



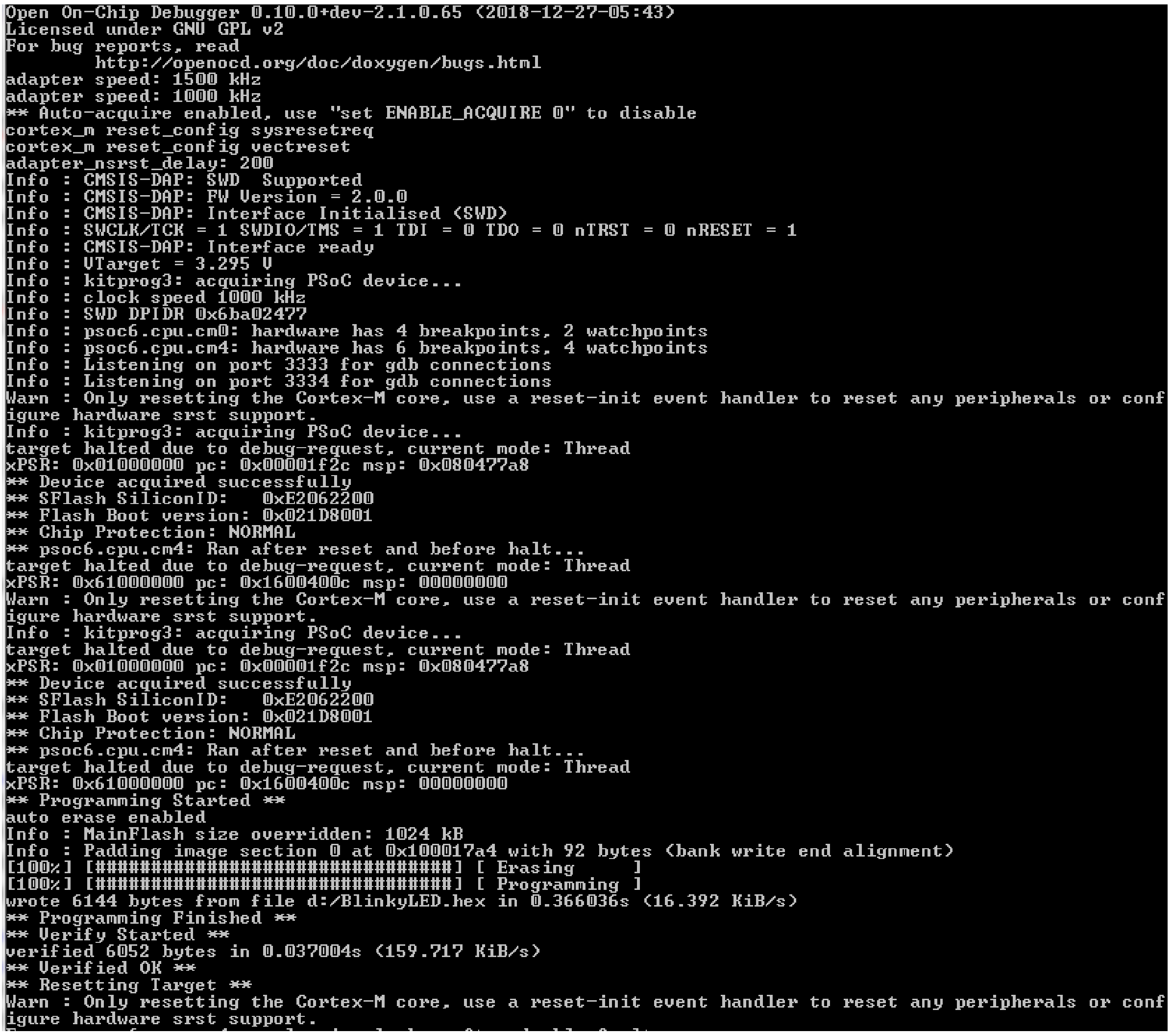
```
Open On-Chip Debugger 0.10.0+dev-2.1.0.65 (2018-12-27-05:43)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
adapter speed: 1000 kHz
** Auto-acquire enabled, use "set ENABLE_ACQUIRE 0" to disable
cortex_m reset_config sysresetreq
cortex_m reset_config vectreset
adapter_nsrst_delay: 200
Info : CMSIS-DAP: SWD  Supported
Info : CMSIS-DAP: FW Version = 2.0.0
Info : CMSIS-DAP: Interface Initialised (SWD)
Info : SWCLK/TCK = 1 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 1
Info : CMSIS-DAP: Interface ready
Info : VTarget = 3.297 V
Info : kitprog3: acquiring PSoC device...
Info : clock speed 1000 kHz
Info : SWD DPIDR 0x6ba02477
Info : psoc6.cpu.cm0: hardware has 4 breakpoints, 2 watchpoints
Info : psoc6.cpu.cm4: hardware has 6 breakpoints, 4 watchpoints
Info : Listening on port 3333 for gdb connections
Info : Listening on port 3334 for gdb connections
Warn : Only resetting the Cortex-M core, use a reset-init event handler to reset any peripherals or conf
igure hardware srst support.
Info : kitprog3: acquiring PSoC device...
target halted due to debug-request, current mode: Thread
xPSR: 0x21000000 pc: 0x00001f2c msp: 0x08047790
** Device acquired successfully
** SFlash SiliconID:   0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
0x08000000: e7feb662 4e0a43c0 2400460b 5d11e00d 40480609 28002500 0040da02 e0004070
0x08000020: 35010040 d1f62d08 429c3401 be00d1ef 04c11db7 b2e44d16 016d1965 2f00682f
0x08000040: 466fdafc 003760ef 40a73410 60274c11 692c60ae dbfc2c00 9d0027a0 0f2c063f
0x08000060: 42bc0724 2400d003 1c286044 2580be00 00ad6844 19521964 d301428c 34080004
```

Program PSoC 6 Device with Verification (Intel HEX file)

OpenOCD supports programming of the elf, Intel HEX, Motorola SREC, and binary file formats. For binary files, the relocation offset must be specified as an argument to the *program* command.

```
openocd -s ../scripts -f psoc6_kp3_board.cfg -c "program d:/BlinkyLED.hex verify
reset; exit"
```

A possible output of OpenOCD:



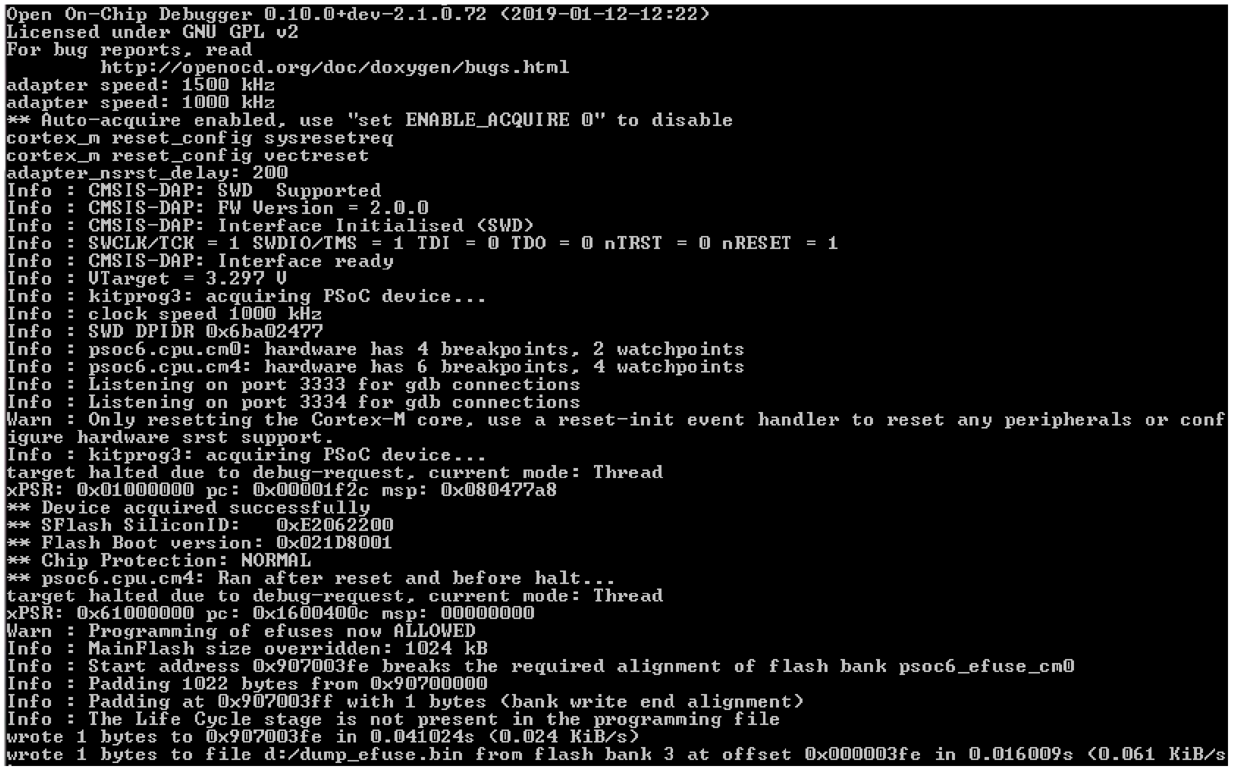
```
Open On-Chip Debugger 0.10.0+dev-2.1.0.65 (2018-12-27-05:43)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
adapter speed: 1000 kHz
** Auto-acquire enabled, use "set ENABLE_ACQUIRE 0" to disable
cortex_m reset_config sysresetreq
cortex_m reset_config vectreset
adapter_nsrst_delay: 200
Info : CMSIS-DAP: SWD  Supported
Info : CMSIS-DAP: FW Version = 2.0.0
Info : CMSIS-DAP: Interface Initialised (SWD)
Info : SWCLK/TCK = 1 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 1
Info : CMSIS-DAP: Interface ready
Info : VTarget = 3.295 V
Info : kitprog3: acquiring PSoC device...
Info : clock speed 1000 kHz
Info : SWD DPIDR 0x6ba02477
Info : psoc6.cpu.cm0: hardware has 4 breakpoints, 2 watchpoints
Info : psoc6.cpu.cm4: hardware has 6 breakpoints, 4 watchpoints
Info : Listening on port 3333 for gdb connections
Info : Listening on port 3334 for gdb connections
Warn : Only resetting the Cortex-M core, use a reset-init event handler to reset any peripherals or conf
igure hardware srst support.
Info : kitprog3: acquiring PSoC device...
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x00001f2c msp: 0x080477a8
** Device acquired successfully
** SFlash SiliconID:   0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
Warn : Only resetting the Cortex-M core, use a reset-init event handler to reset any peripherals or conf
igure hardware srst support.
Info : kitprog3: acquiring PSoC device...
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x00001f2c msp: 0x080477a8
** Device acquired successfully
** SFlash SiliconID:   0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
** Programming Started **
auto erase enabled
Info : MainFlash size overridden: 1024 kB
Info : Padding image section 0 at 0x100017a4 with 92 bytes (bank write end alignment)
[100%] [#################################] [ Erasing     ]
[100%] [#################################] [ Programming ]
wrote 6144 bytes from file d:/BlinkyLED.hex in 0.366036s (16.392 KiB/s)
** Programming Finished **
** Verify Started **
verified 6052 bytes in 0.037004s (159.717 KiB/s)
** Verified OK **
** Resetting Target **
Warn : Only resetting the Cortex-M core, use a reset-init event handler to reset any peripherals or conf
igure hardware srst support.
```

Program EFuse region of PSoC 6 Device

This example writes a single bit of data to the EFuse region of the PSoC6 device at address 0x907003FE:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; psoc6 allow_efuse_program on; flash fillb 0x907003FE 1 1; flash read_bank 3
d:/dump_efuse.bin 0x3FE 0x1; exit"
```

A possible output of OpenOCD:

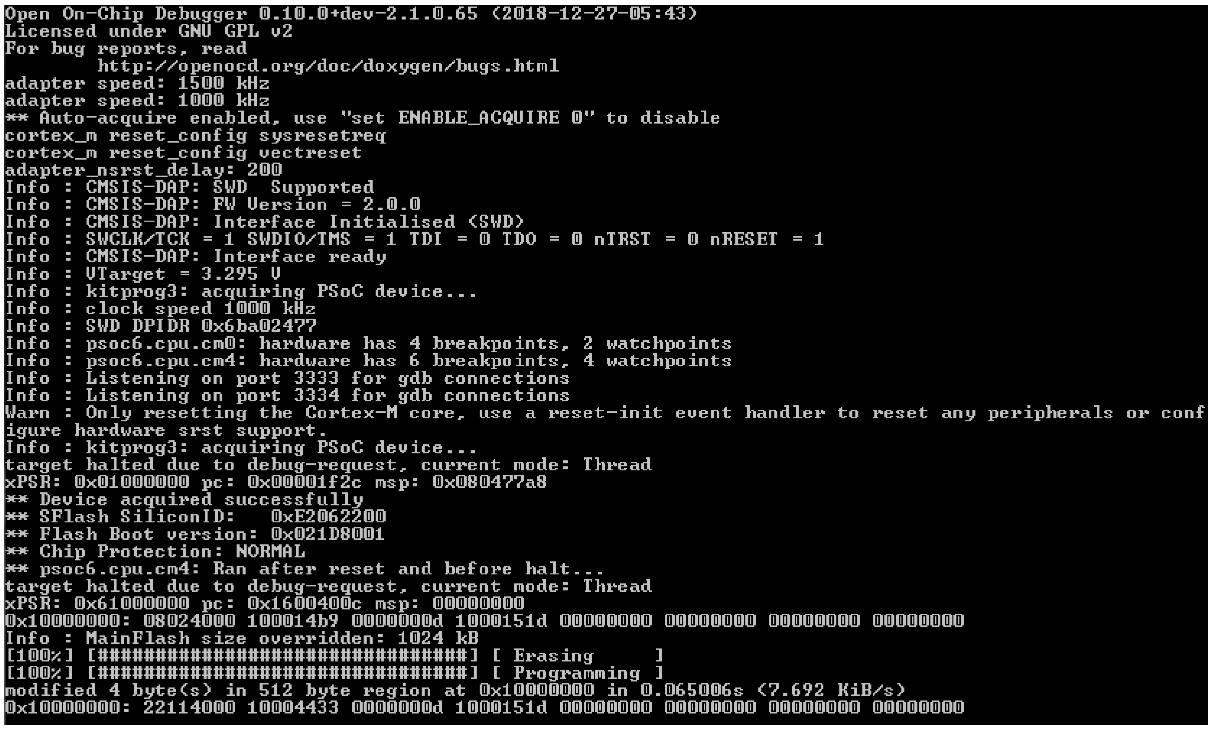


```
Open On-Chip Debugger 0.10.0+dev-2.1.0.72 (2019-01-12-12:22)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
adapter speed: 1000 kHz
** Auto-acquire enabled, use "set ENABLE_ACQUIRE 0" to disable
cortex_m reset_config sysresetreq
cortex_m reset_config vectreset
adapter_nsrst_delay: 200
Info : CMSIS-DAP: SWD  Supported
Info : CMSIS-DAP: FW Version = 2.0.0
Info : CMSIS-DAP: Interface Initialised (SWD)
Info : SWCLK/TCK = 1 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 1
Info : CMSIS-DAP: Interface ready
Info : VTarget = 3.297 V
Info : kitprog3: acquiring PSoC device...
Info : clock speed 1000 kHz
Info : SWD DPIDR 0x6ba02477
Info : psoc6.cpu.cm0: hardware has 4 breakpoints, 2 watchpoints
Info : psoc6.cpu.cm4: hardware has 6 breakpoints, 4 watchpoints
Info : Listening on port 3333 for gdb connections
Info : Listening on port 3334 for gdb connections
Warn : Only resetting the Cortex-M core, use a reset-init event handler to reset any peripherals or conf
igure hardware srst support.
Info : kitprog3: acquiring PSoC device...
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x00001f2c msp: 0x080477a8
** Device acquired successfully
** SFlash SiliconID:    0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
Warn : Programming of efuses now ALLOWED
Info : MainFlash size overridden: 1024 kB
Info : Start address 0x907003fe breaks the required alignment of flash bank psoc6_efuse_cm0
Info : Padding 1022 bytes from 0x90700000
Info : Padding at 0x907003ff with 1 bytes (bank write end alignment)
Info : The Life Cycle stage is not present in the programming file
wrote 1 bytes to 0x907003fe in 0.041024s (0.024 KiB/s)
wrote 1 bytes to file d:/dump_efuse.bin from flash bank 3 at offset 0x000003fe in 0.016009s (0.061 KiB/s
```

Modify Individual Bytes of PSoC 6 in Main Flash and Display Results

```
openocd -s ../scripts -f psoc6_kp3_board.cfg -c "mdw 0x10000000 8; flash rmw
0x10000002 11223344; mdw 0x10000000 8; exit"
```

A possible output of OpenOCD:



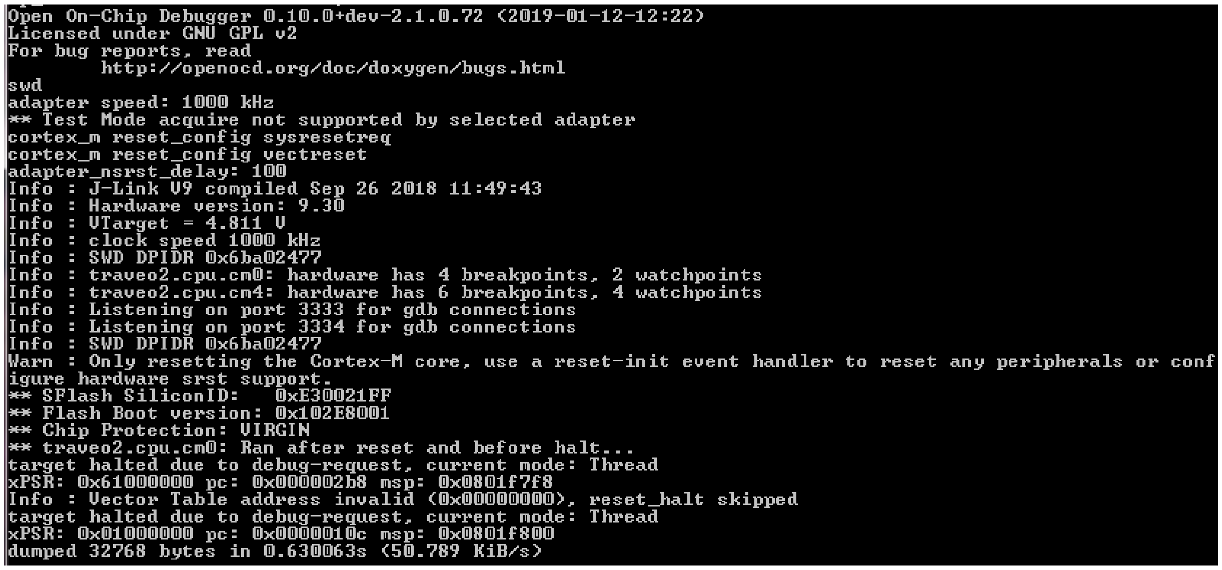
```
Open On-Chip Debugger 0.10.0+dev-2.1.0.65 (2018-12-27-05:43)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
adapter speed: 1000 kHz
** Auto-acquire enabled, use "set ENABLE_ACQUIRE 0" to disable
cortex_m reset_config sysresetreq
cortex_m reset_config vectreset
adapter_nsrst_delay: 200
Info : CMSIS-DAP: SWD  Supported
Info : CMSIS-DAP: FW Version = 2.0.0
Info : CMSIS-DAP: Interface Initialised (SWD)
Info : SWCLK/TCK = 1 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 1
Info : CMSIS-DAP: Interface ready
Info : VTarget = 3.295 V
Info : kitprog3: acquiring PSoC device...
Info : clock speed 1000 kHz
Info : SWD DPIDR 0x6ba02477
Info : psoc6.cpu.cm0: hardware has 4 breakpoints, 2 watchpoints
Info : psoc6.cpu.cm4: hardware has 6 breakpoints, 4 watchpoints
Info : Listening on port 3333 for gdb connections
Info : Listening on port 3334 for gdb connections
Warn : Only resetting the Cortex-M core, use a reset-init event handler to reset any peripherals or conf
igure hardware srst support.
Info : kitprog3: acquiring PSoC device...
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x00001f2c msp: 0x080477a8
** Device acquired successfully
** SFlash SiliconID:   0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
0x10000000: 08024000 100014b9 0000000d 1000151d 00000000 00000000 00000000 00000000
Info : MainFlash size overridden: 1024 kB
[100%] [##################################] [ Erasing     ]
[100%] [##################################] [ Programming ]
modified 4 byte(s) in 512 byte region at 0x10000000 in 0.065006s (7.692 KiB/s)
0x10000000: 22114000 10004433 0000000d 1000151d 00000000 00000000 00000000 00000000
```

Read Memory of PSoC 6 Device to Binary File

The example reads a 32KB of the PSoC 6 device memory to a file *dump_mem.bin*.

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc6.cfg -c "init; reset
init; dump_image d:/dump_mem.bin 0x10000000 0x8000; exit"
```

A possible output of OpenOCD:

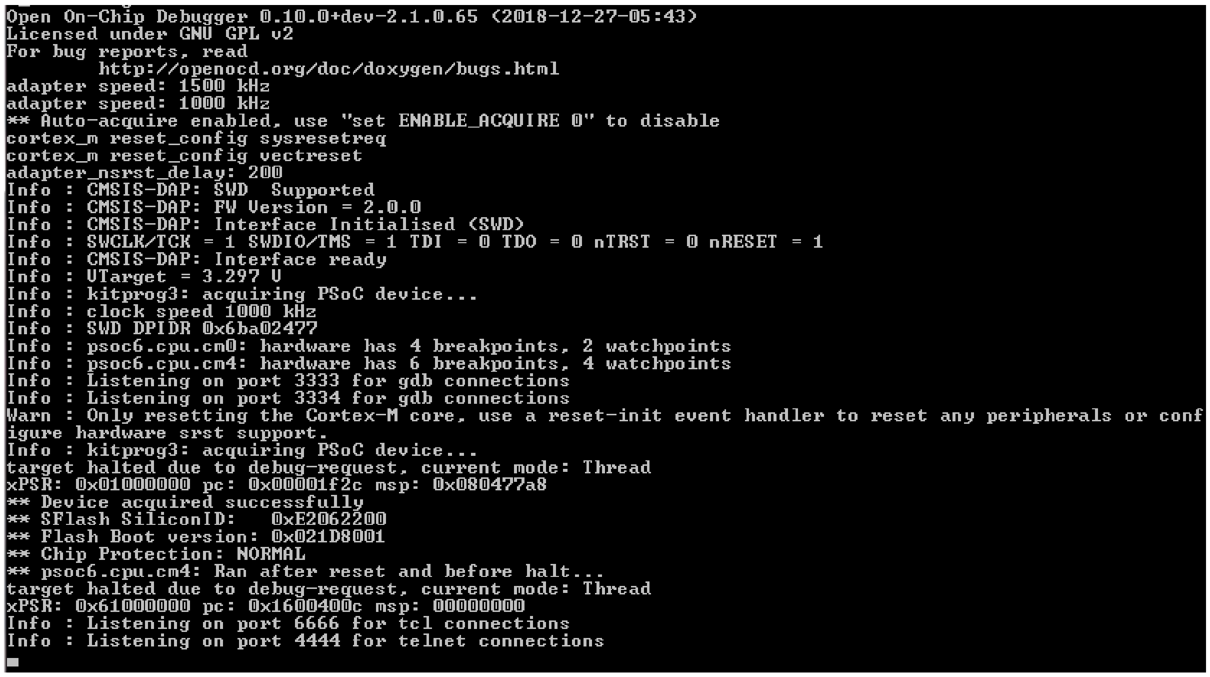


```
Open On-Chip Debugger 0.10.0+dev-2.1.0.72 (2019-01-12-12:22)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
swd
adapter speed: 1000 kHz
** Test Mode acquire not supported by selected adapter
cortex_m reset_config sysresetreq
cortex_m reset_config vectreset
adapter_nsrst_delay: 100
Info : J-Link V9 compiled Sep 26 2018 11:49:43
Info : Hardware version: 9.30
Info : VTarget = 4.811 V
Info : clock speed 1000 kHz
Info : SWD DPIDR 0x6ba02477
Info : traveo2.cpu.cm0: hardware has 4 breakpoints, 2 watchpoints
Info : traveo2.cpu.cm4: hardware has 6 breakpoints, 4 watchpoints
Info : Listening on port 3333 for gdb connections
Info : Listening on port 3334 for gdb connections
Info : SWD DPIDR 0x6ba02477
Warn : Only resetting the Cortex-M core, use a reset-init event handler to reset any peripherals or conf
igure hardware srst support.
** SFlash SiliconID:    0xE30021FF
** Flash Boot version: 0x102E8001
** Chip Protection: VIRGIN
** traveo2.cpu.cm0: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x000002b8 msp: 0x0801f7f8
Info : Vector Table address invalid (0x00000000), reset_halt skipped
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x0000010c msp: 0x0801f800
dumped 32768 bytes in 0.630063s (50.789 KiB/s)
```

Start GDB Server and Leave It Running

```
openocd -s ../scripts -f psoc6_kp3_board.cfg
```

A possible output of OpenOCD:



```
Open On-Chip Debugger 0.10.0+dev-2.1.0.65 (2018-12-27-05:43)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
adapter speed: 1000 kHz
** Auto-acquire enabled, use "set ENABLE_ACQUIRE 0" to disable
cortex_m reset_config sysresetreq
cortex_m reset_config vectreset
adapter_nsrst_delay: 200
Info : CMSIS-DAP: SWD  Supported
Info : CMSIS-DAP: FW Version = 2.0.0
Info : CMSIS-DAP: Interface Initialised (SWD)
Info : SWCLK/TCK = 1 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 1
Info : CMSIS-DAP: Interface ready
Info : VTarget = 3.297 V
Info : kitprog3: acquiring PSoC device...
Info : clock speed 1000 kHz
Info : SWD DPIDR 0x6ba02477
Info : psoc6.cpu.cm0: hardware has 4 breakpoints, 2 watchpoints
Info : psoc6.cpu.cm4: hardware has 6 breakpoints, 4 watchpoints
Info : Listening on port 3333 for gdb connections
Info : Listening on port 3334 for gdb connections
Warn : Only resetting the Cortex-M core, use a reset-init event handler to reset any peripherals or conf
igure hardware srst support.
Info : kitprog3: acquiring PSoC device...
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x00001f2c msp: 0x080477a8
** Device acquired successfully
** SFlash SiliconID:    0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
Info : Listening on port 6666 for tcl connections
Info : Listening on port 4444 for telnet connections
```

11 Documentation and Links

OpenOCD v0.10.0 User Guide:

http://openocd.org/doc-release/pdf/openocd.pdf