ModusToolbox™ software

OpenOCD CLI user guide

About this document

Scope and purpose

This user guide provides technical information for the ModusToolbox™ version of the OpenOCD command line tool, including how to use it in stand-alone mode. OpenOCD is an Open Source programmer/debugger software that is installed as part of either the CYPRESS™ Programmer or ModusToolbox™ software.

Intended audience

This document is intended for anyone who wants to use the OpenOCD CLI as a stand-alone tool.

Document conventions

Convention	Explanation
Bold	Emphasizes heading levels, column headings, menus and sub-menus
Italics	Denotes file names and paths.
`Courier New`	Denotes APIs, functions, interrupt handlers, events, data types, error handlers, file/folder names, directories, command line inputs, code snippets
File > New	Indicates that a cascading sub-menu opens when you select a menu item

Abbreviations and definitions

The following define the abbreviations and terms used in this document:

- 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 Infineon. For example, 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™ – A family of microcontroller integrated circuits by Infineon. These chips include a CPU core and mixed-signal arrays of configurable integrated analog and digital peripherals.
- MCU – Microcontroller unit.
- 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.

Reference documents

Refer to the following documents for more information as needed:

- OpenOCD v0.12.0 user guide: http://openocd.org/doc-release/pdf/openocd.pdf
- ModusToolbox™ software installation guide
- CYPRESS™ Programmer GUI user guide

Table of contents

1 Introduction

1.1 Overview

The ModusToolbox™ OpenOCD command-line interface (CLI) is based on the Open On-Chip Debugger (OpenOCD) product. 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.

This document covers the ModusToolbox™-specific CLI extensions of OpenOCD. See the official documentation at:

http://openocd.org/documentation/

The latest released version of ModusToolbox™ OpenOCD is available from the GitHub repository:

https://github.com/Infineon/openocd/releases

1.2 Supported OS

- Windows 7 SP1 (x64), Windows 10 (x64), Windows 11 (x64)
- Linux Ubuntu 18.04 "Bionic Beaver", Ubuntu 20.04 "Focal Fossa", Ubuntu 22.04 "Jammy Jellyfish"
- macOS 11 "Big Sur", macOS 12 "Monterey", macOS 13 "Ventura"

1.3 Supported devices

- AIROC™ CYW20809 Bluetooth® LE system on chip
- PSoC™ 6 and PSoC™ 64
- PSoC™ 4
- PMG1, CCGx
- CYW4390x [1]
- XMC7000
- TRAVEO™ T2G Body High

1.4 Supported hardware (probes)

- SEGGER J-Link
- Infineon KitProg3 on-board programmer
- Infineon MiniProg4 standalone programmer
- FTDI-based adapter on CYW954907AEVAL1F / CYW943907AEVAL1F kits

1 Currently, OpenOCD does not provide a "built-in" Flash driver for the CYW4390x chip. All Flash-related operations are fully implemented in the TCL scripts. The behavior of the TCL-based driver is slightly different from the built-in one. Refer to the CYW4390x commands section for details.

1.5 Installation

The ModusToolbox™ OpenOCD CLI software is installed as part of either CYPRESS™ Programmer or ModusToolbox™ software. You can also download the latest version from the GitHub repository:

https://github.com/Infineon/openocd/releases

Refer the ModusToolbox™ software installation guide for details.

Note: CYPRESS™ Programmer is not part of ModusToolbox™ software and must be installed separately. See the CYPRESS™ Programmer GUI user guide.

1.6 Error codes

The OpenOCD tool returns '0' as the response code on successful completion; on a failure, it returns '1'.

2 Getting started

2.1 Connect the device

Connect the host computer to a probe or kit device; e.g. KitProg3 kit with the PSoC™ 6 MCU target, used in the following examples. Make sure that the target MCU is attached to your probe.

2.2 List the connected targets

This example displays the target names available for the PSoC™ 6 MCU connected to the KitProg3 programmer. The programmer communicates with the PSoC™ 6 MCU over the SWD interface.

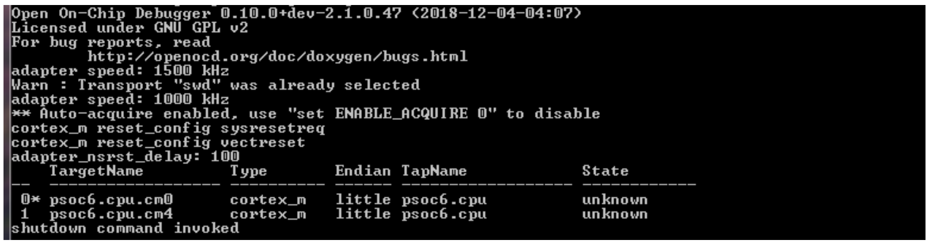
2.2.1 Windows:

1. On a command-line window, enter the following command to change the directory to the CYPRESS™ Programmer or ModusToolbox™ software installation folder:

   ```
   cd %installation folder%\openocd\bin
   ```

2. Run the following command:

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

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

2.2.2 Linux:

1. On the terminal window, go to the directory where CYPRESS™ Programmer or ModusToolbox™ software is installed (for example, `~/openocd/bin`).

2. Run the following 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.

2.2.3 macOS:

1. One the terminal window, go to the directory where CYPRESS™ Programmer or ModusToolbox™ software is installed (for example, `~/openocd/bin`).

2. Run the following command:

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

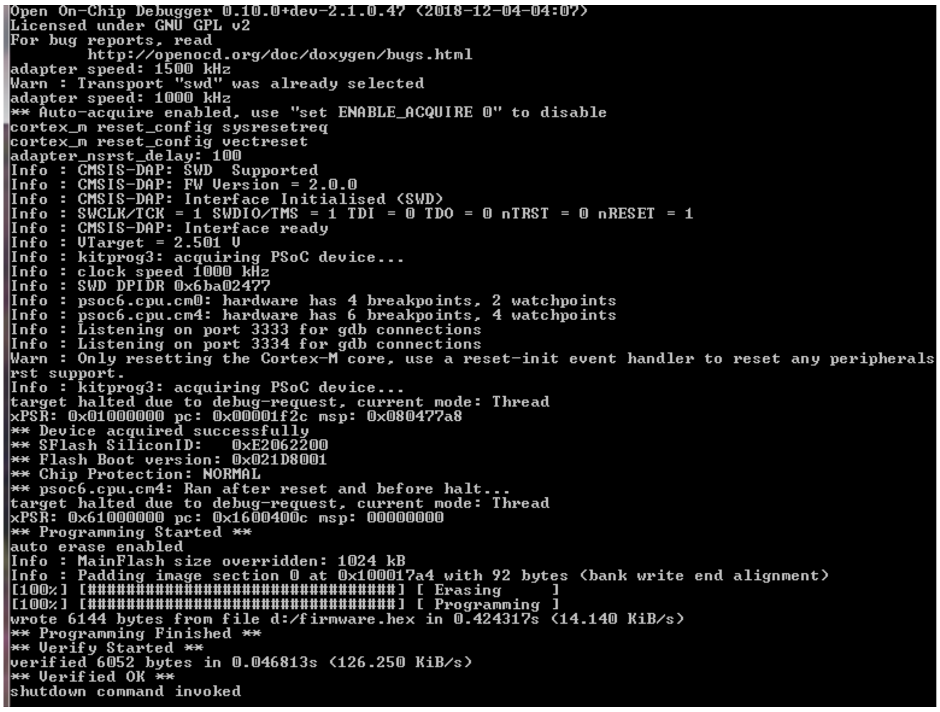
The command output displays the list of target names (JTAG TAPs) attached to the programming device.

2.3 Program the PSoC™ 6 MCU target

This example initializes the KitProg3 probe with the PSoC™ 6 MCU, programs the flash with the *firmware.hex* file, verifies the programmed data, and finally shuts down the OpenOCD programmer.

Run the following command:

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

2.4 Program the PSoC™ 64 "Secure Boot" MCU target

This example initializes the KitProg3 probe with the PSoC™ 64 MCU, programs the flash with the *firmware.hex* file, verifies the programmed data, and finally shuts down the OpenOCD programmer.

Run the following 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 the internal flash is performed via the SYS_AP access port. OpenOCD will not affect CM0_AP and CM4_AP by default, so both cores will not be visible to OpenOCD. Choose the access port using the TARGET_AP variable.*

Programming of the external memory is done by the flash loader, so the CM4 access port must be used for QSPI memory programming. After choosing the CM4 access port, the QSPI memory bank will be exposed automatically.

Note: *See Supported target configurations for the list of available target configurations.*

2.5 Program the PSoC™ 4 MCU target

This example initializes the KitProg3 probe with the PSoC™ 4 MCU, programs the flash with the *firmware.hex* file, verifies programmed data, and finally shuts down the OpenOCD programmer.

Execute the following command:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc4.cfg -c "program
d:/firmware.hex verify exit"
```

2.6 Program the device using the 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 MCU connected using the KitProg3 debug probe. This file initializes the target device, programs the 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 following command:

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

2.7 Program the device using the 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 following command:

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

2.8 Remote debugging of PSoC™ 6 MCU target

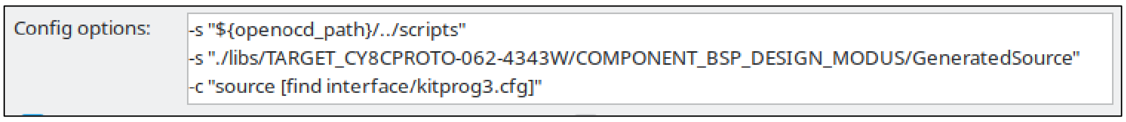
OpenOCD is a server application which implements the remote *gdbserver* protocol and, as such, supports remote debugging out of the box. All communication between the debugger (GDB) and OpenOCD is done via TCP connections, even when debugging a locally connected target. This example shows how to configure ModusToolbox™ software for remote debugging and launch a remote debug session of the PSoC™ 6 MCU target.

This example uses two PCs: a local PC where ModusToolbox™ software will be running and a remote PC with the CY8CPROTO-062-4343W kit connected. The remote PC runs OpenOCD and acts as a server; both machines will communicate via TCP protocol.

On a local machine:

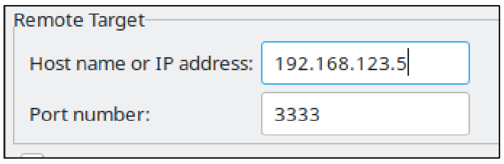
1. Start the ModusToolbox™ software. Create and build a project for CY8CPROTO-062-4343W.

2. Open the **Debug Configurations** window and select the **Debug (KitProg3_MiniProg4)** configuration.
3. Do the following on the **Debugger** tab:
    a. Copy the contents of the **Config options** text box and paste them to a text editor.
    b. Replace all new lines with spaces because all configuration options should be in a single line separated by spaces.
    c. Copy the contents to the clipboard. You will need to specify these options in the OpenOCD command line on a remote machine later.



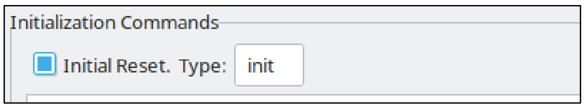
Config options:
```
-s "${openocd_path}/../scripts"
-s "./libs/TARGET_CY8CPROTO-062-4343W/COMPONENT_BSP_DESIGN_MODUS/GeneratedSource"
-c "source [find interface/kitprog3.cfg]"
```

4. Disable the **Start OpenOCD locally** checkbox.
5. Specify the **Host name or IP address** of the remote machine in the **Remote Target** group box.



Remote Target
Host name or IP address: 192.168.123.5
Port number: 3333

6. Navigate to the **Startup** tab and make sure that the **Initial Reset** checkbox is checked and the **Reset Type** is set to **init**.



Initialization Commands
☑ Initial Reset. Type: init

7. Click **Apply** to save changes to the launch configuration.

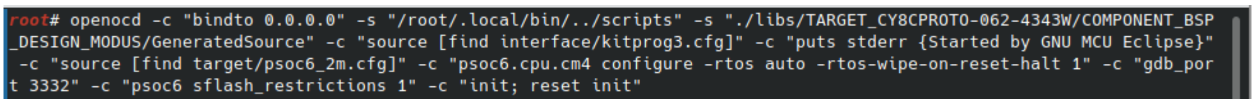
On a remote machine:

1. Type the following command in the command prompt, and then paste the configuration options from the clipboard to complete the command line (see step 3 above).

    ```
    openocd -c "bindto 0.0.0.0"
    ```

2. Replace the `${openocd_path}` with the full path to the OpenOCD executable on the remote machine.

    The completed command line should look similar to the following:



```
root# openocd -c "bindto 0.0.0.0" -s "/root/.local/bin/../scripts" -s "./libs/TARGET_CY8CPROTO-062-4343W/COMPONENT_BSP
_DESIGN_MODUS/GeneratedSource" -c "source [find interface/kitprog3.cfg]" -c "puts stderr {Started by GNU MCU Eclipse}"
 -c "source [find target/psoc6_2m.cfg]" -c "psoc6.cpu.cm4 configure -rtos auto -rtos-wipe-on-reset-halt 1" -c "gdb_por
t 3332" -c "psoc6 sflash_restrictions 1" -c "init; reset init"
```

3. Press [**Enter**] to start the OpenOCD session.
4. Switch back to a local machine and start the modified Debug (KitProg3_MiniProg4) launch configuration.

3 Supported target configurations

Target configuration files are in the *target/* directory of the OpenOCD tree. To connect ModusToolbox™ OpenOCD CLI to a device, pass one of the following configuration files as the argument for the `--file` command-line option; for example, `-f target/psoc6.cfg`.

#	Target config	Description
1	*psoc6.cfg*	CY8C6xx7, CY8C6xx6 target configuration
2	*psoc6_2m.cfg*	CY8C6xxA, CY8C6xx8 target configuration
3	*psoc6_512k.cfg*	CY8C6xx5 target configuration
4	*psoc6_256k.cfg*	CY8C6xx4 target configuration
5	*psoc6_secure.cfg*	CYB06447, CYB06447-BL target configuration
6	*psoc6_2m_secure.cfg*	CYS0644A, CYB0644A target configuration
7	*psoc6_512k_secure.cfg*	CYB06445 target configuration
8	*psoc4.cfg*	Configuration for all PSoC™ 4 MCU targets except PSoC™ 4500H MCU
9	*psoc4500.cfg*	Configuration file for PSoC™ 4500H MCU
10	*pag2s.cfg*	Configuration file for PAG2S MCU
11	*cyw208xx.cfg*	Configuration file for the AIROC™ CYW208xx Wi-Fi & Bluetooth® combo chips
12	*bcm4390x.cfg* [2]	Configuration file for CYW4390x family of devices
13	*cat1c.cfg*	Configuration file for XMC7xxx and TRAVEO™ T2G Body High MCU devices

[2] The CYW9WCD1EVAL1 kit is equipped with an onboard FTDI-based JTAG adapter. OpenOCD provides board-level configuration file for this kit which will configure the JTAG adapter and the CYW4390x chip automatically. Use single *board/cyw9wcd1eval1.cfg* configuration file for the CYW9WCD1EVAL1 board (instead of separate files for the probe and chip).

4 Command-line options

OpenOCD command-line 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 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

4.1 --file (-f)

Specifies the configuration file to use.

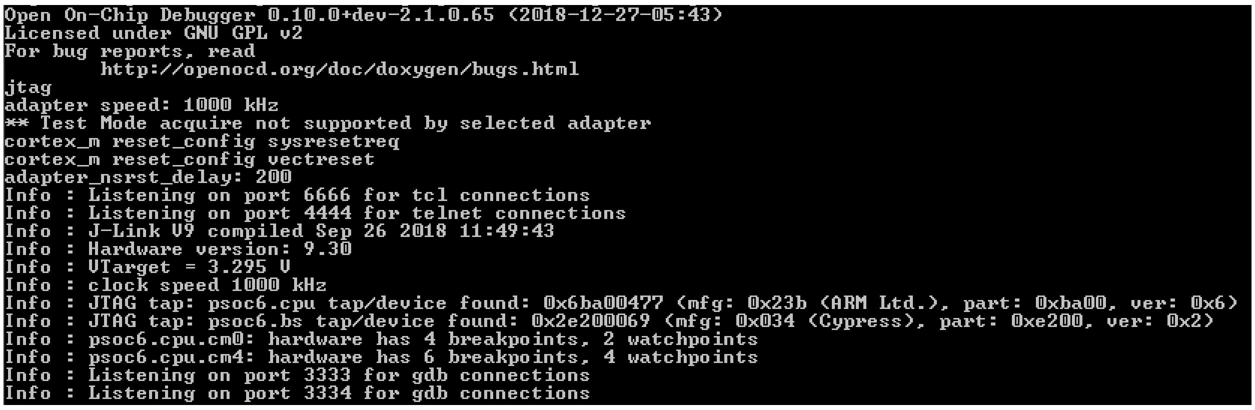
Multiple configuration files can be specified from a command line. They are interpreted in the order they are specified in the command line.

```
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
```

The output should appear similar to the following:



```
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 appear with no warnings, which means the JTAG communication is working.

4.2 --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 file first found 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.

4.3 --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"
```

4.4 --debug (-d)

Specifies the debug level. The debug level is 2 by default.

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

This affects the kind of messages sent to the server log:

- Level 0: Error messages only
- Level 1: Level 0 messages + warnings
- Level 2: Level 1 messages + informational messages
- Level 3: Level 2 messages + debugging messages

Example:

```
openocd -d1
```

4.5 --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"
```

4.6 --help (-h)

Displays the help message.

```
openocd -h
```

4.7 --version (-v)

Displays the OpenOCD version.

```
openocd -v
```

5 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 the `-c command` option.

Command	**Description**
`version`	Displays a string identifying the OpenOCD version
`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 speed`	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 a supported transport 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 protect`	Enables (on) or disables (*off*) protection of flash blocks
`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 the flash memory with the specified word (32 bits), half-word (16 bits), or byte
`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 the memory regions to be compared during the *verify* operation

Command	Description
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 a 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 KitProg3/MiniProg4 internal power supply parameters and optionally enables power
kitprog3 power_control	Turns on or off the KitProg3/MiniProg4 internal power supply
kitprog3 led_control	Controls KitProg3/MiniProg4 LEDs
kitprog3 get_power	Reports the target voltage in millivolts
psoc6 sflash_restrictions	Enables or disables writes to dedicated SFlash regions
psoc6 allow_efuse_program	Allows or disallows writes to the EFuse region
psoc6 reset_halt	Simulates a broken vector catch on PSoC™ 6 MCUs
cat1c sflash_restrictions	Enables or disables writes to dedicated SFlash regions
cat1c allow_efuse_program	Allows or disallows writes to the EFuse region
cat1c reset_halt	Simulates a broken vector catch on XMC7xxx and TRAVEO™ T2G MCUs
cat1c ecc_error_reporting	Enables or disables the ECC error reporting
cat1c wflash_blank_map	Displays per-word validity map of the given sectors of Fork Flash
cat1c wflash_write_image	Programs individual 32-bit words from given file to the Work Flash
cat1c wflash_write_words	Modifies individual 32-bit words in Work Flash
psoc4 mass_erase	Performs a mass erase operation on the given flash bank
psoc4 chip_protect	Changes chip protection mode to PROTECTED
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

6 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 or PSoC™ 64 MCU targets. See Supported Target Configurations for the detailed list of available target devices and corresponding OpenOCD configuration files.

6.1 General OpenOCD commands

6.1.1 version

Displays a string identifying the OpenOCD version.

Example:

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

6.1.2 help

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

```
help [string]
```

Example:

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

6.1.3 shutdown

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

```
shutdown [error]
```

Example:

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

6.1.4 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"
```

6.1.5 debug_level

Displays the debug level. If *n* (from 0..3) is provided, set it to that level.

This affects the kind of messages sent to the server log:

- Level 0: Error messages only
- Level 1: Level 0 messages + warnings
- Level 2: Level 1 messages + informational messages
- Level 3: Level 2 messages + 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"
```

6.1.6 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`).

The `signals` option 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` (default)

`trst_only`

`srst_only`

`trst_and_srst`

See the OpenOCD documentation for details.

Example:

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

6.1.7 adapter speed

Sets a non-zero speed in kHz for the debug adapter. Therefore, to specify 3 MHz, provide `3000`.

```
adapter speed <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 lower than that peak rate.

For example, most Arm® cores accept up to 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
speed 2000; shutdown"
```

6.1.8 transport list

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

Example:

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

6.1.9 transport select

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

```
transport select <transport_name>
```

When invoked with the `transport_name` option, OpenOCD 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"
```

6.1.10 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 relevant only to boards with more than one target.

```
targets [name]
```

Examples:

Displays all available targets of the connected PSoC™ 6 MCU:

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

Selects the CM4 core of the PSoC™ 6 MCU as the current target:

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

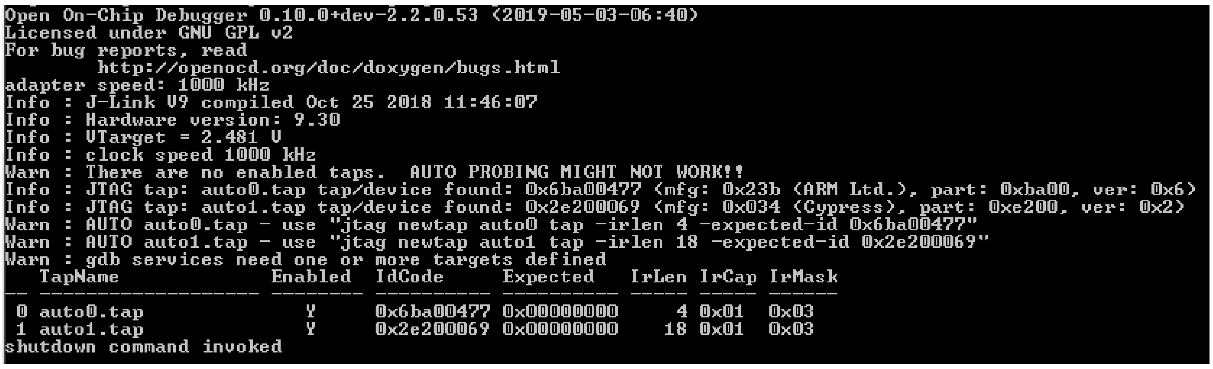
6.1.11 scan_chain

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 MCU.

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

6.1.12 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`).

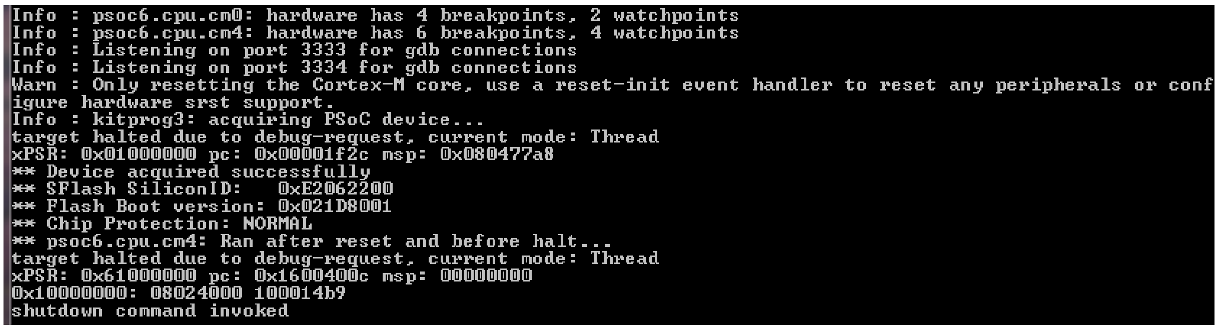
```
mdw [phys] <addr> [count]
mdh [phys] <addr> [count]
mdb [phys] <addr> [count]
```

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 PSoC™ 6 MCU.

```
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
```

6.1.13 mw(w)(h)(b)

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

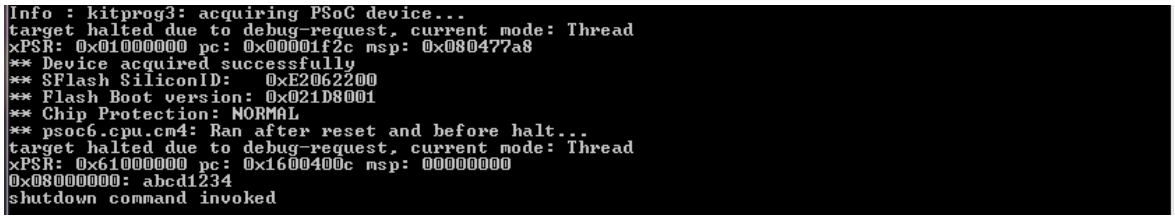
```
mww [phys] <addr> <word>
mwh [phys] <addr> <halfword>
mwb [phys] <addr> <byte>
```

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 PSoC™ 6 MCU.

```
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
```

6.1.14 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 and programming flash. 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 MCU target):

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

6.1.15 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 initialize the KitProg3 + PSoC™ 6 MCU target:

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

6.1.16 program

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

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

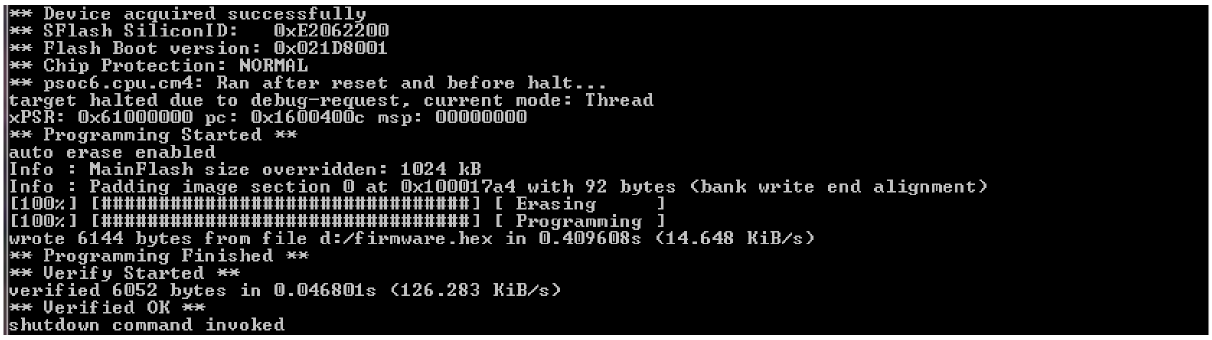
The only required parameter is `filename`; others are optional.

- `preverify` – Performs the verification step before flash programming. Programming will be skipped if the flash contents match the data file.
- `verify` – Compares the contents of the data file `filename` with the contents of the flash after flash programming
- `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` – If relocation offset is specified, it is added to the base address for each section in the image.

Example:

The next example connects ModusToolbox™ OpenOCD CLI to the KitProg3 probe with the PSoC™ 6 MCU target, 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"
```



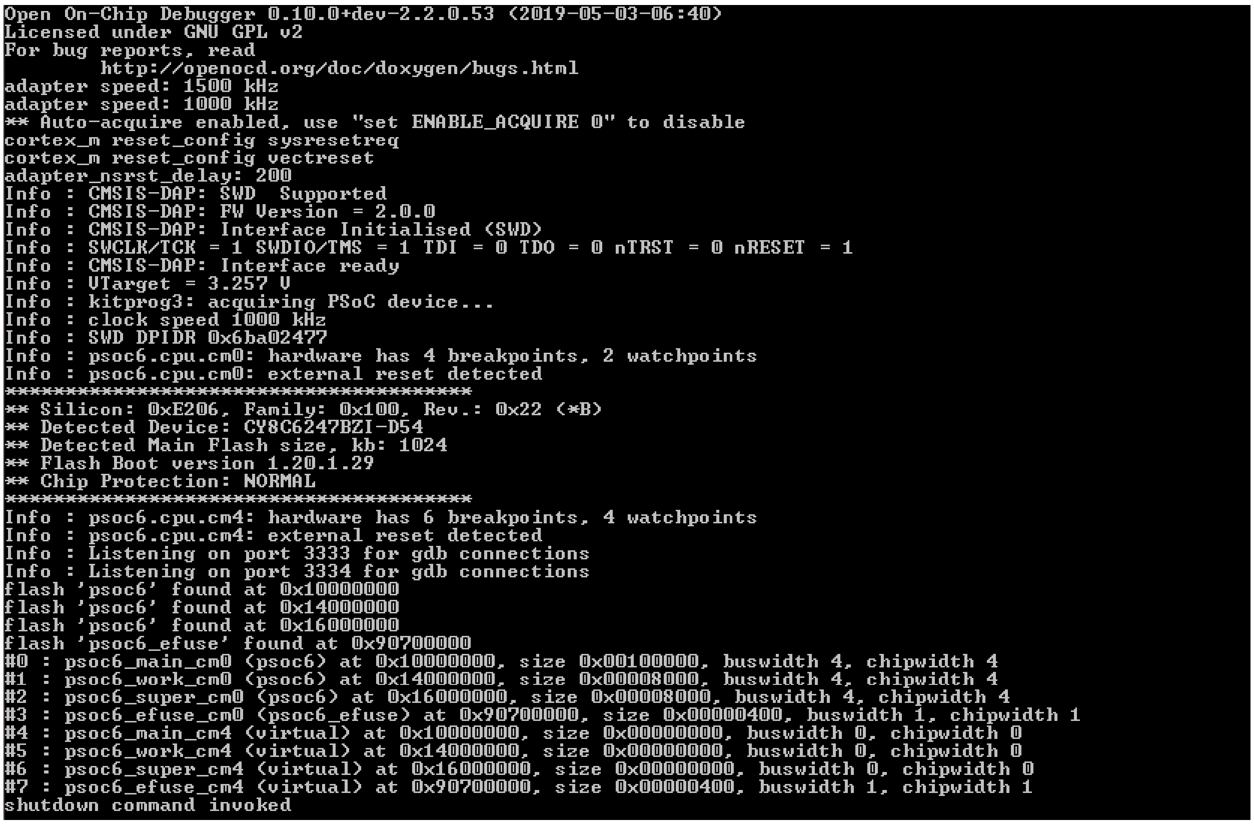
```
** Device acquired successfully
** SFlash SiliconID:   0xE2062200
** Flash Boot version: 0x021D8001
** Chip Protection: NORMAL
** psoc6.cpu.cm4: Ran after reset and before halt...
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x1600400c msp: 00000000
** Programming 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
```

6.1.17 flash banks

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

Example (KitProg3 + PSoC™ 6 MCU):

```
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"
```



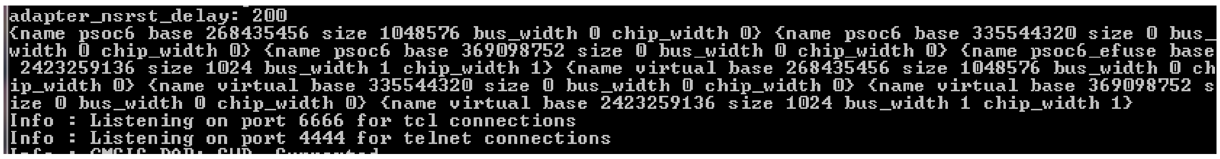
```
Open On-Chip Debugger 0.10.0+dev-2.2.0.53 (2019-05-03-06:40)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
adapter speed: 1000 kHz
** Auto-acquire enabled, use "set ENABLE_ACQUIRE 0" to disable
cortex_m reset_config sysresetreq
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
```

6.1.18 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 MCU):

```
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
```

6.1.19 flash info

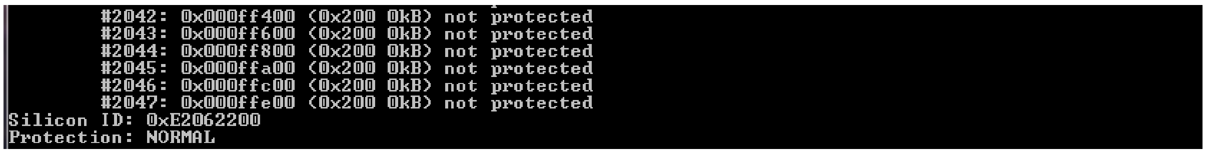
```
flash info <num> [sectors]
```

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 the information about flash bank 0 of the KitProg3 + PSoC™ 6 MCU:

```
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
```

6.1.20 flash protect

Enables (*on*) or disables (*off*) protection of flash blocks in flash bank `num`, starting at protection block `first` and continuing up to and including `last`.

Note: *This command is applicable for PSoC™ 4 MCU only.*

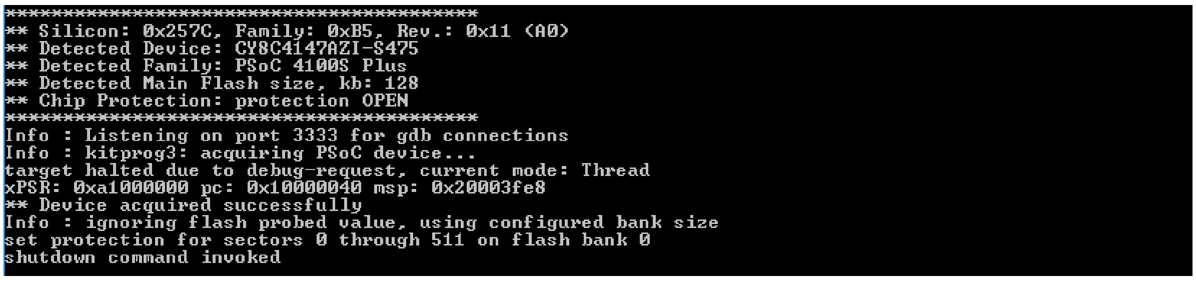
```
flash protect num first last (on|off)
```

Providing a `last` block of last specifies "to the end of the flash bank". The `num` parameter is a value shown by flash banks. The protection block is usually identical to a flash sector. Some devices may utilize a protection block distinct from the flash sector. See `flash info` for a list of protection blocks.

Example:

Protects all sectors from being written in flash bank 0 of the KitProg3 + PSoC™ 4 MCU:

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc4.cfg -c "init;
reset init; flash protect 0 0 last on; shutdown"
```



```
*****************************************
** Silicon: 0x257C, Family: 0xB5, Rev.: 0x11 (A0)
** Detected Device: CY8C4147AZI-S475
** Detected Family: PSoC 4100S Plus
** Detected Main Flash size, kb: 128
** Chip Protection: protection OPEN
*****************************************
Info : Listening on port 3333 for gdb connections
Info : kitprog3: acquiring PSoC device...
target halted due to debug-request, current mode: Thread
xPSR: 0xa1000000 pc: 0x10000040 msp: 0x20003fe8
** Device acquired successfully
Info : ignoring flash probed value, using configured bank size
set protection for sectors 0 through 511 on flash bank 0
shutdown command invoked
```

6.1.21 flash erase_sector

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

```
flash erase_sector <num> <first> <last>
```

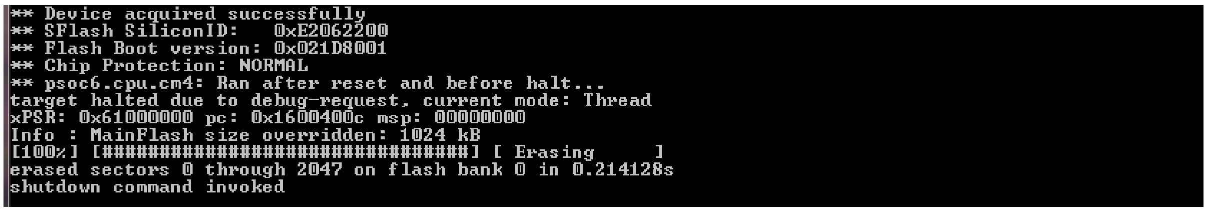
Sector numbering starts at 0. Providing the `last` sector of `last` specifies "to the end of the flash bank". The `num` parameter is a value shown by flash banks.

Note: On PSoC™ 4 MCU devices, per-sector erase operation is not supported, only mass-erase is available. This command is ignored on PSoC™ 4 unless a full erase of the flash is requested (`flash erase_sector 0 0 last`).

Example:

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

```
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
```

6.1.22 flash erase_address

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

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

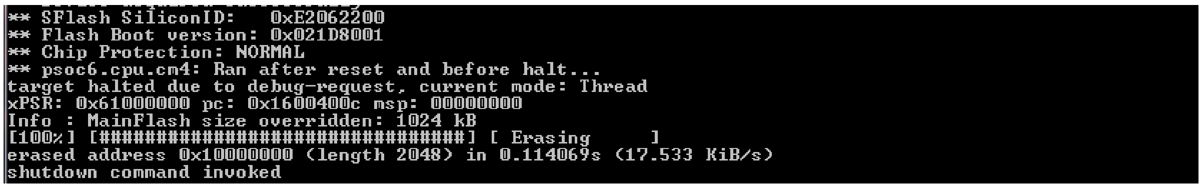
Unless `pad` is specified, `address` must begin a flash sector, and `address` + `length` - 1 must end a sector. Specifying `pad` erases the extra data at the beginning and/or end of the specified region, as needed to erase only full sectors. The flash bank to use is inferred from the `address`, and the 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, the flash is unprotected before `erase` starts.

Note: On PSoC™ 4 MCU devices, per-sector erase operation is not supported, only mass-erase is available. This command is ignored on PSoC™ 4 MCU unless a full erase of the flash is requested (e.g. `flash erase_address 0 65536` for 64 KB parts).

Example:

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

```
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
```

6.1.23 flash write_bank

Writes the binary `filename` to flash bank `num`, starting at `offset` bytes from the beginning of the bank.

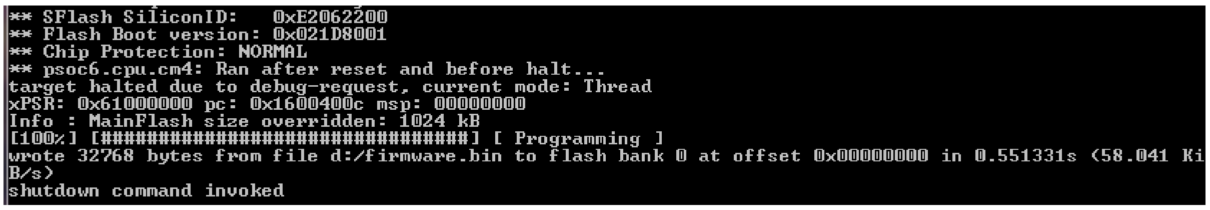
```
flash write_bank <num> <filename> <offset>
```

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

Example:

Writes the binary file *firmware.bin* to flash bank 0 of KitProg3 + PSoC™ 6 MCU 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
```

6.1.24 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. If a relocation `offset` is specified, 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), or `s19` (Motorola s19). The relevant flash sectors will be erased prior to programming if the `erase` parameter is given. If `unlock` is provided, the flash banks are unlocked before `erase` and `program`. The flash bank to use is inferred from the address of each image section.

Attention: ***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.***

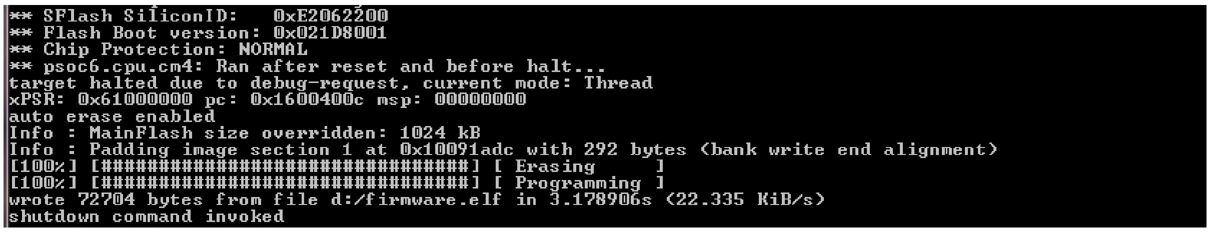
- When a section of the image being written does not fill out all the sectors it uses, the unwritten parts of those sectors are necessarily also erased, because sectors cannot be partially erased.
- Data stored in sector "holes" between image sections are also affected. For example, `flash write_image 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 reapply it after it has been removed by the `unlock` flag.

Example:

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

```
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
```

6.1.25 flash fill(w)(h)(b)

Fills the 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).

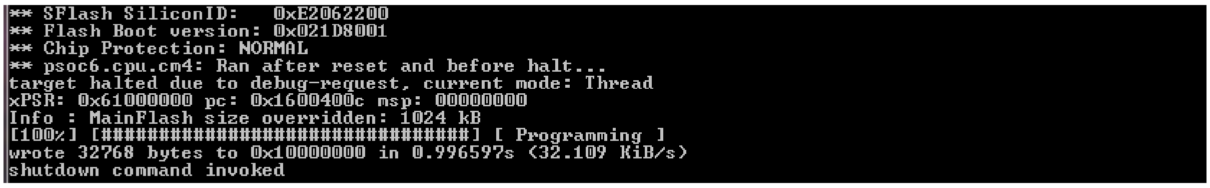
```
flash fillw <address> <word> <length>
flash fillh <address> <halfword> <length>
flash fillb <address> <byte> <length>
```

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 MCU 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
```

6.1.26 flash read_bank

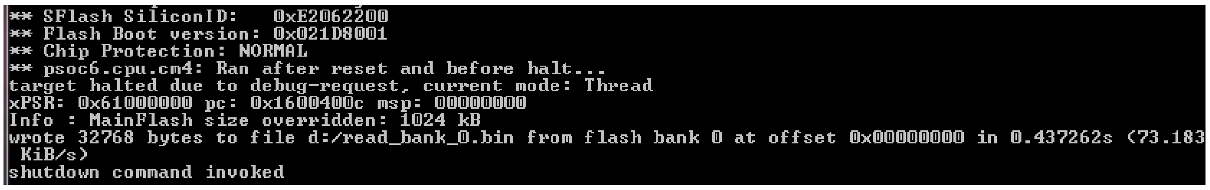
Reads the `length` bytes from the flash bank `num` starting at `offset` and writes the contents to the binary `filename`. The `num` parameter is a value shown by flash banks.

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

Example:

Reads the 32-KB block of bank #0 from the PSoC™ 6 MCU 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
```

6.1.27 flash verify_bank

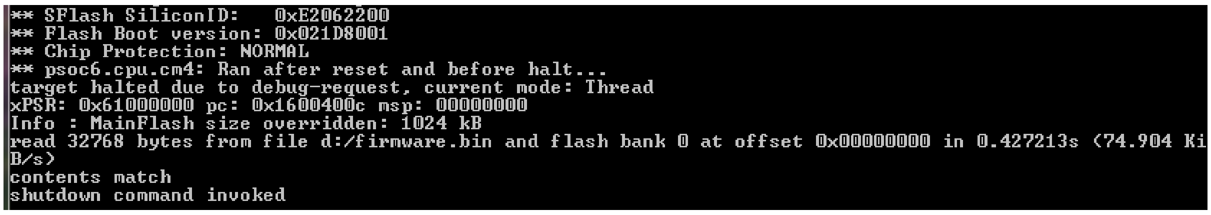
Compares the contents of the binary file `filename` with the contents of the flash `num` starting at `offset`. Fails if the contents do not match. The `num` parameter is a value shown by flash banks.

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

Example:

Verifies the content of bank #0 of the PSoC™ 6 MCU:

```
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
```

6.1.28 flash padded_value

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

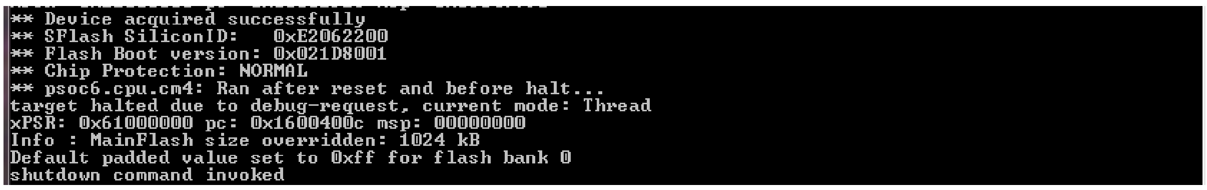
```
flash padded_value <num> <value>
```

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

Example:

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

```
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
```

6.1.29 flash rmw

The command is intended to modify flash individual bytes.

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

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

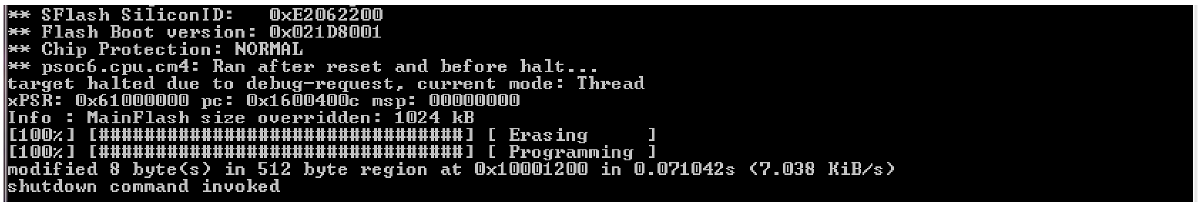
- `address` – The start address for the programming.
- `data` – The hexadecimal string with data to be programmed. The format of the string is shown in the following example:

Note: *`flash rmw` is a custom command implemented in ModusToolbox™ OpenOCD CLI to extend its functionality.*

Example:

Modifies 8 bytes of the PSoC™ 6 MCU 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: 00000000
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
```

6.1.30 add_verify_range

The command allows specifying memory regions to be compared during the `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 the 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.

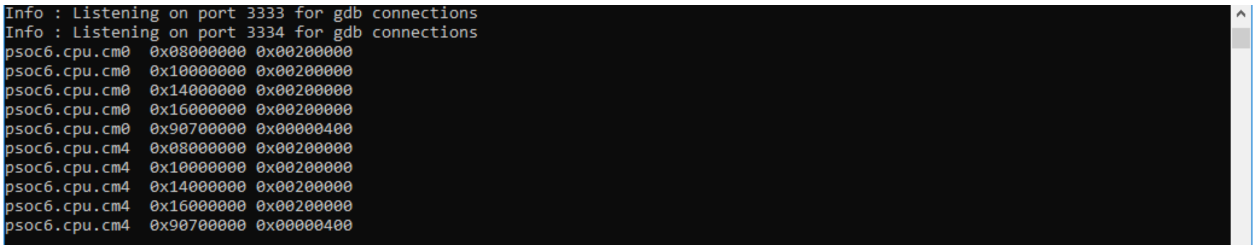
Note: The `add_verify_range` command is a custom command implemented in ModusToolbox™ OpenOCD CLI to extend its functionality.

6.1.31 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
```

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

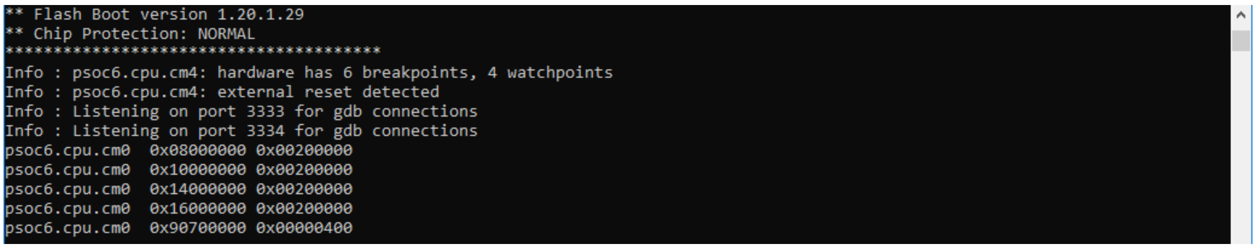
6.1.32 clear_verify_ranges

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"
```



```
** 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
```

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

6.1.33 verify_image

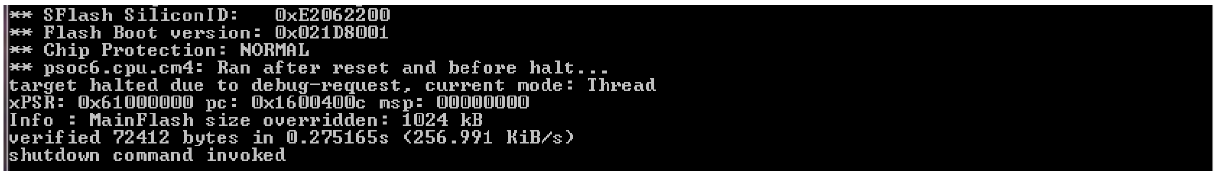
Verifies `filename` against the target memory starting at `address`. The file format may optionally be specified (bin, ihex, or elf). This will first attempt a comparison using a CRC checksum; if that fails, it will try a binary compare.

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

Examples:

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

```
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
```

6.1.34 verify_image_checksum

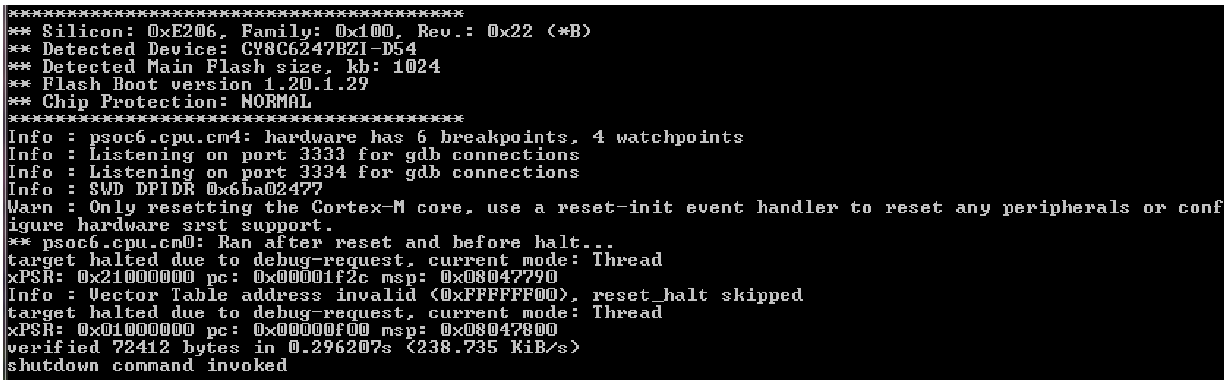
Verifies *filename* against the target memory starting at *address*. The file format may optionally be specified (bin, ihex, or elf). This perform a comparison using a CRC checksum only.

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

Example:

Verifies a *firmware.elf* image against the target memory of the PSoC™ 6 MCU using the CRC checksum 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
```

6.1.35 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:

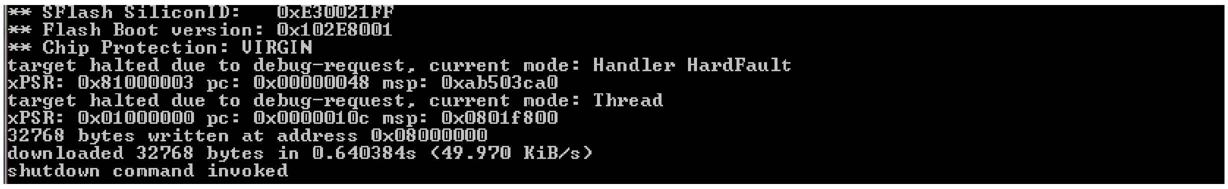
- `min_addr` - Ignore the data below min_addr (this is w.r.t. to the target's load address + address)
- `max_length` - Maximum number of bytes to load

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

Examples:

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

```
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
```

6.1.36 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 8 KB of the PSoC™ 6 MCU memory to the 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"
```

6.2 KitProg3/MiniProg4 driver commands

The KitProg3/MiniProg4 probe implements the CMSIS-DAP protocol defined by Arm® with some extensions. Consequently, the KitProg3/MiniProg4 driver in OpenOCD is a wrapper around the native CMSIS-DAP driver that extends its functionality with the KitProg3-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 KitProg3 driver exposes several custom Tcl configuration commands. All commands in this section must be prefixed with the name of the driver – "kitprog3".

6.2.1 kitprog3 acquire_config

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

```
acquire_config <status> [target_type] [mode] [attempts] [timeout] [ap]
```

- `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 the acquisition mode. This parameter is mandatory only if `status=on`. The possible values: 0 – Reset, 1 – Power Cycle. The mode affects only the first step(how to reset the part at the start of the acquisition flow).
  - Reset mode: To start programming, the host toggles the XRES line and then sends SWD/JTAG commands
  - Power Cycle mode: To start programming, the KitProg3-based probe powers on the MCU and then starts sending the SWD/JTAG commands. The XRES line is not used. Power Cycle mode support is optional and should be used only if the XRES pin is not available on the part's package.

Note: *Before using Power Cycle acquisition, make sure that the target is not powered externally!*

- `attempts` – The number of attempts to acquire the target device. This parameter is mandatory only if `status=on`.
- `timeout` – (Optional) Timeout value in seconds. The maximum value for the timeout is 30 seconds.
- `ap` – Access port to use for the acquisition. The value of this parameter should be in range 0…255. This parameter is mandatory if the `timeout` parameter is specified.

Example:

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

6.2.2 kitprog3 acquire_psoc

Performs device acquisition. Called only after the initialization phase. The acquisition procedure must be configured using `acquire_config` before 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"
```

6.2.3 kitprog3 power_config

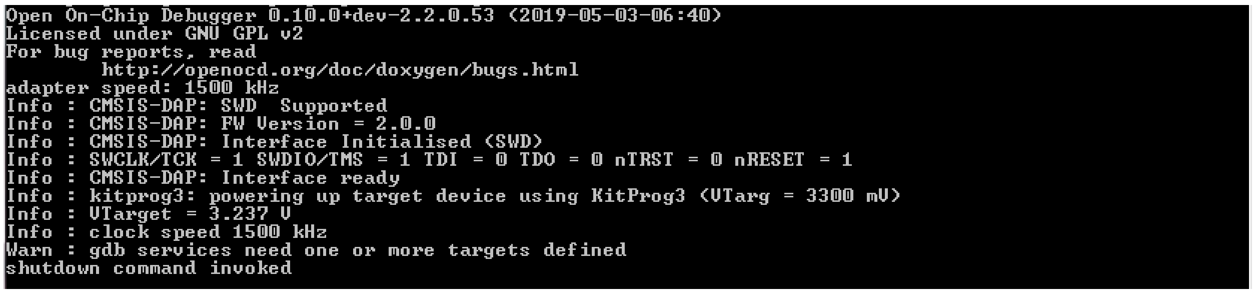
Controls the KitProg3-internal power supply parameters and optionally enables power during the early initialization phase. Can be called at any time.

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

- `status` –Mandatory; enables or disables power supply during the initialization phase. Possible values: `on|off`.
- `voltage` – The power supply voltage in millivolts. This parameter is optional. Either default (2.5 volts) or kit-specific voltage will be applied if this parameter is not specified.

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
```

6.2.4 kitprog3 power_control

This command is deprecated, please use kitprog3 power_config instead.

6.2.5 kitprog3 led_control

Controls the KitProg3 LEDs. Can be called only after the initialization phase.

```
kitprog3 led_control <type>
```

- `type` – Mandatory; specifies the type of the LED indication. The possible values:
  - 0 – Ready
  - 1 – Programming
  - 2 – Success
  - 3 – Error

Example:

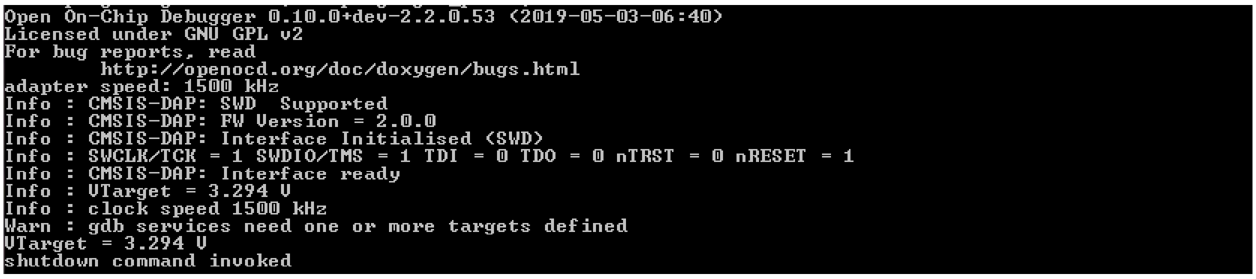
```
openocd -s ../scripts -f interface/kitprog3.cfg -c "init; kitprog3 led_control 2"
```

6.2.6 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"
```



```
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
```

6.2.7 cmsis_dap_serial

Specifies the serial number of the KitProg3/MiniProg4 device to use. If not specified, serial numbers are not considered. Command can be used to specify which device to use if multiple devices are connected to the host PC.

6.3 Flash driver commands

This section contains a list of custom commands exposed by the target MCU's Flash driver.

6.3.1 psoc6/cat1c sflash_restrictions

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

```
psoc6 sflash_restrictions <mode>
cat1c sflash_restrictions <mode>
```

The command can be called at any time.

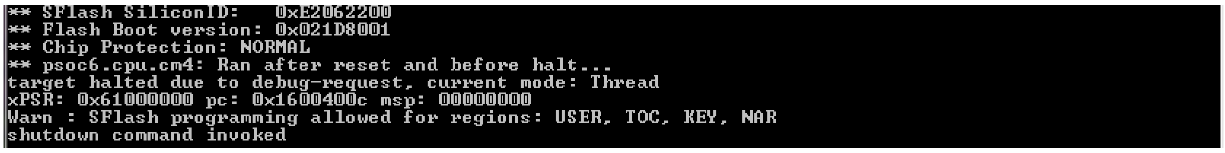
- `mode` – Mandatory; specifies the behavior of SFlash programming. The possible values:
  - 0 – Erase/Program of SFlash is prohibited.
  - 1 – Erase and Program of USER/TOC/KEY is allowed.
  - 2 – Erase of USER/TOC/KEY and program of USER/TOC/KEY/NAR is allowed.

    Be aware that the NAR sub-region cannot be overwritten or erased if the new data is less restrictive than the existing data. **Unintentional writing to this region may corrupt your device!**
  - 3 – Erase of USER/TOC/KEY and program of the whole SFlash region is allowed.

Writes to SFlash regions other than USER/TOC/KEY/NAR is possible only on the VIRGIN silicon, so the mode=3 option is mostly intended for internal use. It is useful for flash boot developers and validation teams. Note that erase (programming with 0x00 for PSoC™ 6, or 0xFF for XMC7xxx and TRAVEO™ T2G MCUs) is performed only for the USER, TOC2, and KEY regions; it is skipped for other SFlash regions regardless of this command.

Example (KitProg3 + PSoC™ 6 MCU):

```
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
```

6.3.2 psoc6/cat1c allow_efuse_program

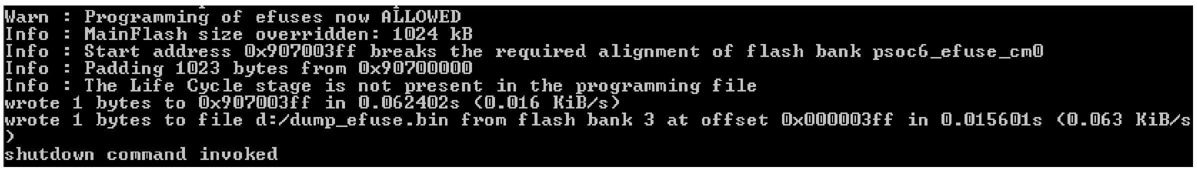
Allows or disallows writes to the EFuse region. Can be called any time. Writes to the EFuse region are skipped by default. Be aware that EFuses are one-time programmable. Once an EFuse is blown, there is no way to revert its state. EFuse programming must be allowed for lifecycle transitions to work.

```
psoc6 allow_efuse_program <on|off>
cat1c allow_efuse_program <on|off>
```

Example:

Writes 1 bit to the EFuse region at address 0x907003FF of the PSoC™ 6 MCU:

```
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
```

6.3.3 psoc6/cat1c reset_halt

The command simulates a broken vector catch on PSoC™ 6, XMC7xxx and TRAVEO™ T2G MCUs.

```
psoc6 reset_halt <mode>
cat1c reset_halt <mode>
```

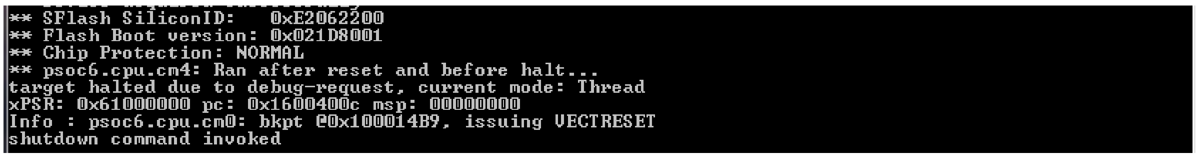
The command retrieves the address of the vector table from the VECTOR_TABLE_BASE registers, detects the location of the application entry points, sets a hardware breakpoint at that location, and performs a reset of the target. The type of the reset can be specified by the optional `mode` parameter.

Parameters:

- `mode` – (Optional) The type of reset to be performed. Possible values are `sysresetreq` and `vectreset`. If not specified, `SYSRESETREQ` is used for the CM0 core and `VECTRESET` is used for other cores in the system.

Example (KitProg3 + PSoC™ 6 MCU):

```
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
```

6.3.4 cat1c ecc_error_reporting

Enables or disables the ECC error reporting during OpenOCD operations.

```
cat1c ecc_error_reporting <on|off>
```

OpenOCD supports the detection and reporting of ECC errors during the flash *read* operation. In the current implementation, OpenOCD reads word-by-word a requested amount of data and checks for the ECC status after each Read. This ensures all ECC errors for all memory locations are properly detected. If an ECC error occurs, OpenOCD retrieves the address of the faulty location from the hardware. All ECC errors along with their locations are reported to the user by means of warning messages. This process will be performed until all requested data has been read.

Example (XMC7xxx device):

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/cat1c.cfg -c "init;
reset init; cat1c ecc_error_reporting on; shutdown"
```

6.3.5 cat1c wflash blank_map [first_sector [last_sector | 'last']]

Displays per-word validity map of the given sectors of work flash. It accepts two optional parameters with the following rules:

- If no parameters are given, command displays validity map for all sectors
- If one parameter is given, parameter means sector number and the command displays validity map for given sector
- If two parameters are given, command displays validity map for range of sectors (param1 ... param2). The word 'last' can be used as a parameter #2 (same as erase_sector)

Example output on WFlash sector 0:

```
> cat1c wflash blank_map 0

WorkFlash word validity map:
0x14000000 (#000): -+++-+-+--------------------------------------------------
0x14000100 (#000): ----------------------------------------------------------
0x14000200 (#000): ----------------------------------------------------------
0x14000300 (#000): ----------------------------------------------------------
0x14000400 (#000): ----------------------------------------------------------
0x14000500 (#000): ----------------------------------------------------------
0x14000600 (#000): ----------------------------------------------------------
0x14000700 (#000): ----------------------------------------------------------
```

6.3.6 cat1c wflash write_image <filename> [offset]

Programs individual 32-bit words from given file to the work flash. All data in the file that does not belong to WFlash region is skipped. All unaligned data is trimmed to make the starting address and length of the data aligned on 32-bit boundaries. Appropriate warnings are displayed in this case. Command works with elf, hex, srec and bin files. Optional offset can be specified (same as flash write_image).

Example output (file contains unaligned data and also data which does not belong to WFlash region):

```
> cat1c wflash write_image foo.hex

Warn : Section [0x13fff950, 0x13fff991) will be skipped
Warn : Section [0x13fffdac, 0x13fffe19) will be skipped
Warn : Section [0x13fffff0, 0x1400005d) will be truncated to [0x14000000,
0x1400005c)
Warn : Section [0x14000290, 0x14000299) will be truncated to [0x14000290,
0x14000298)
Warn : Section [0x140003b0, 0x140003e5) will be truncated to [0x140003b0,
0x140003e4)
[100%] [################################] [ Programming ]
```

6.3.7 cat1c wflash write_words <address> <word_1> [word_2] ... [word_N]

Command is similar to 'flash rmw' except:

- Starting address must be aligned on 32-bit boundary
- Command works with 32-bit words which must be separated with a space

Example output:

```
> cat1c wflash write_words 0x14000004 0xDEADBEEF 0xBAADF00D 0xBAADC0DE
> cat1c wflash write_words 0x14000014 0x01234567
> cat1c wflash write_words 0x1400001C 0x89ABCDEF
> cat1c wflash blank_map 0

[100%] [################################] [ Programming ]
[100%] [################################] [ Programming ]
[100%] [################################] [ Programming ]
WorkFlash word validity map:
0x14000000 (#000): -+++-+-+--------------------------------------------------------
0x14000100 (#000): ----------------------------------------------------------------
0x14000200 (#000): ----------------------------------------------------------------
0x14000300 (#000): ----------------------------------------------------------------
0x14000400 (#000): ----------------------------------------------------------------
0x14000500 (#000): ----------------------------------------------------------------
0x14000600 (#000): ----------------------------------------------------------------
0x14000700 (#000): ----------------------------------------------------------------
```

6.3.8 psoc6 secure_acquire

Performs acquisition of PSoC™ 64 "Secure Boot" MCUs.

```
psoc6 secure_acquire <magic_num_addr> <mode> <handshake> <timeout>
```

Parameters:

- `magic_num_addr` – Address in RAM to poll for the magic number. This address is different across different PSoC™ 6 MCU devices:
  - CYB06447, CYB06447-BL - 0x08044804
  - CYS0644A, CYB0644A - 0x080FE004
  - CYB06445 - 0x0803E004
- `mode` –Mode of acquisition. Possible values: `run`, `halt`.
  - In `run` mode, the command will perform reset and will wait for the "secure" application to open the corresponding access port.
  - In `halt` mode, a "secure" handshake will be performed right after reset to prepare the device for flash programming.
- `handshake` – Specifies whether full or short acquisition procedure should be executed. The short acquisition procedure simply waits until "secure" FW opens the given access port. This is intended for multi-core configuration when full acquisition has already been done with the other CPU core.
  - Possible values: `handshake` – full acquisition, `no_handshake` – short acquisition
- `timeout` –Timeout in milliseconds

6.3.9 psoc4 reset_halt

Performs the alternate acquire sequence as described in the PSoC™ 4 MCU programming specification.

```
psoc4 reset_halt
```

The command detects the location of the application entry points, sets a hardware breakpoint at that location, and issues a `SYSRESETREQ` reset.

Example (KitProg3 + PSoC™ 4 MCU):

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc4.cfg -c "init;
reset init; psoc4 reset_halt; shutdown"
```

6.3.10 psoc4 mass_erase

Performs mass erase operation on the given flash bank. The list of all flash banks can be obtained using `flash banks` command. This command is a shortcut and performs the same operation as the `flash erase_sector <bank_id> 0 last` command. The peculiarity of this command is that erasing of the `mflash` bank also erases the `flashp` bank. If the chip is in PROTECTED state, this command moves the protection state of the device from PROTECTED to OPEN and erases the entire flash device.

```
psoc4 mass_erase <bank_id>
```

Example (KitProg3 + PSoC™ 4 MCU):

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc4.cfg -c "init;
reset init; psoc4 mass_erase 0; shutdown"
```

6.3.11 psoc4 chip_protect

Changes the chip protection mode to PROTECTED. This mode disables all debug access to the user code or memory. Access to most registers is still available; debug access to registers to reprogram flash is not available. Protection mode can be changed back to OPEN by performing the mass erase operation described above.

```
psoc4 chip_protect
```

Example (KitProg3 + PSoC™ 4 MCU):

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/psoc4.cfg -c "init;
reset init; psoc4 chip_protect; shutdown"
```

6.4 cmsis_flash flash driver commands

The "cmsis_flash" is a generic driver which uses the standard CMSIS flash loaders to program the flash. On Infineon devices, this driver is typically used for external flash programming but it also can be used for other purposes.

6.4.1 cmsis_flash init

Some types of flash banks are not mapped to the CPU address space right after reset. For example, the external flash connected to an SMIF peripheral requires special configuration of the MCU's hardware blocks in order to be mapped to the CPU address space. Usually, the flash loader's `Init()` function is responsible for enabling such mapping.

```
cmsis_flash init [bank_num]
```

This command loads the flash loader to the RAM and executes the `Init()` function. Beware that this function is intrusive. It requires that the MCU is acquired in "good-state" and all CPUs are halted (e.g., `reset init` is performed) before it can be called.

This command takes one optional argument – the number of the flash bank to be initialized. This command will initialize all `cmsis_flash` banks if no argument is specified.

6.4.2 cmsis_flash prefer_sector_erase

Controls driver strategy used during mass-erase of the flash bank. There are two possible strategies:

- Use the `EraseChip` API (if available)
- Use per-sector erase using the `EraseSector` API

The `EraseChip` method is used by default. This method is usually faster but it does not display the progress of the erase operation. The `EraseChip` API is optional; the driver will fall back to per-sector erase if the `EraseChip` API is not implemented in the flash loader. The other downside of this method is that depending on the flash loader implementation, it may erase all external memory banks, not only the bank specified in the `erase_sector` command.

Per-sector erase is usually slower but it displays the progress information and always erases the single flash bank specified in the `erase_sector` command.

```
cmsis_flash prefer_sector_erase [bank_num] <0/1|false/true>
```

Command takes two arguments:

- `bank_num` – (Optional) Flash bank number to enable/disable the per-sector erase strategy. This option will be applied to all cmsis_flash banks if this argument is omitted.
- `parameter_value` – Mandatory boolean value specifying whether per-sector strategy should be enabled or disabled.

6.5 Other commands

6.5.1 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 MCU):

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

6.5.2 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"
```

6.5.3 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"
```

6.5.4 sleep

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

```
sleep msec
```

Example:

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

6.5.5 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"
```

6.6 CYW4390x commands

The CYW4390x chip supports only limited subset of flash-related commands. Currently, the following flash programming commands are supported:

6.6.1 program

Programs a given binary programming into the flash of the target device.

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

The only required parameter is `filename`; the others are optional.

- `reset` – Calls `reset run` (see reset for details)
- `exit` – Shuts down OpenOCD

Limitations compared to standard `program` command:

- Only binary files are supported. Files in any format other than binary will be programmed as a binary data. For example, if a HEX file is programmed, the flash will contain textual representation of the file.
- The `preverify`, `verify`, and `offset` parameters are not supported. Verification is done automatically during programming.

Example:

The next example connects ModusToolbox™ OpenOCD CLI to the CYW9WCD1EVAL1 board with the CYW4390x MCU target, programs the flash with the *firmware.bin* file, resets the target, and finally shuts down the OpenOCD programmer.

```
openocd -s ../scripts -f board/cyw9wcd1eval1.cfg -c "program d:/firmware.bin reset
exit"
```

6.6.2 erase_all

Erases the external flash memory chip.

Example:

```
openocd -s ../scripts -f board/cyw9wcd1eval1.cfg -c "erase_all; exit"
```

6.7 AIROC™ CYW20829 Wi-Fi & Bluetooth® combo chip commands

6.7.1 provision_no_secure

Performs a transition from NORMAL to NORMAL_NO_SECURE lifecycle. CYW20829 devices come from a factory in the NORMAL lifecycle stage. In this stage, the boot code will not launch the programmed application after reset. The lifecycle must be changed to either SECURE or NORMAL_NO_SECURE stage to use the device. Normally, this task is performed using *cysecuretools*; this command is just a shortcut which simplifies the process. Transition to the SECURE lifecycle is not supported by this command and must be performed using *cysecuretools*.

Note: *This command must be executed before the* `init` *command.*

```
provision_no_secure <service_app> <app_params> [service_app_addr] [params_addr]
```

Required parameters:

- `service_app` – File name of the binary service application image, with path
- `app_params` – File name of the service application's parameters image, with path

Optional parameters:

`service_app_addr` – Address in the RAM where the service application will be loaded; 0x20004000 by default.

`params_addr` - Address in the RAM where the service application's parameters will be loaded; 0x2000D000 by default.

Example:

The next example connects ModusToolbox™ OpenOCD CLI to the CYW9WCD1EVAL1 board with the CYW4390x MCU target, programs the flash with the *firmware.bin* file, resets the target, and finally shuts down the OpenOCD programmer.

```
openocd -s ../scripts -f interface/kitprog3.cfg -f target/cyw20829.cfg -c
"provision_no_secure D:/service_app.bin D:/app_params.bin; exit"
```

7 Global variables

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

7.1 PSoC™ 6 MCU global variables

7.1.1 ENABLE_ACQUIRE

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

Possible values:

- 1 – Reset acquisition enabled (default with KitProg3/MiniProg4)
- 2 – Power Cycle acquisition enabled. The voltage level can be controlled by using ENABLE_POWER_SUPPLY.
- 0 – Acquisition disabled (default for other debug adapters)

7.1.2 ENABLE_POWER_SUPPLY

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

Possible values:

- 0 – Power supply disabled
- Any other value defines target voltage in millivolts.
- default – Sets the last used voltage before KitProg3/MiniProg4 was powered off.

7.1.3 ENABLE_CM0, ENABLE_CM4

Allows specifying the CPU cores to be visible to OpenOCD. OpenOCD never affects disabled cores.

Possible values:

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

7.1.4 TARGET_AP

Applicable for "secure" (PSoC™ 64) MCUs only. Enables the choice of DAP access port that will be used for programming.

Possible values:

- sys_ap – SYS_AP (AP #0, default)
- cm0_ap – CM0_AP (AP #1)
- cm4_ap – CM4_AP (AP #2). Choosing this access port will enable external SMIF memory banks.

7.1.5 FLASH_RESTRICTION_SIZE

Applicable for "secure" (PSoC™ 64) MCUs only. Use this variable to limit the size of accessible flash so OpenOCD will not affect flash locations where the "secure" CyBootloader is located. The default value of this variable varies across different PSoC™ 64 MCUs.

7.1.6 ENABLE_WFLASH, ENABLE_SFLASH, ENABLE_EFUSE

Applicable for "secure" (PSoC 64) MCUs only. Enables the corresponding flash bank when set to a non-zero value. The WorkFlash is enabled by default on PSoC™ 64 CYS0644A, CYB0644A, CYB06447 and CYB06447-BL MCU devices. SFlash and eFuse banks are disabled by default on all PSoC™ 64 MCU targets.

7.1.7 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` – XIP mapping address
- `BankSzN` – Total size of this flash bank, in bytes
- `ProgramSzN` – Minimal programming granularity (program block size), in bytes
- `EraseSzN` – Minimal erase granularity (erase block size), in bytes

7.2 PSoC™ 4 MCU global variables

7.2.1 PSOC4_USE_ACQUIRE

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

Possible values:

- 1 – Reset acquisition enabled (default with KitProg3/MiniProg4)
- 2 – Power Cycle acquisition enabled. The voltage level can be controlled by using ENABLE_POWER_SUPPLY.
- 0 – Acquisition disabled (default for other debug adapters)

7.3 AIROC™ CYW20829 Wi-Fi & Bluetooth® combo chip global variables

7.3.1 DEBUG_CERTIFICATE

Allows to specify the location of the debug certificate binary file. This variable is used to configure a secure debug session. OpenOCD checks the status of the CM33 access port during the initialization phase and after each reset. It will attempt to reopen the CM33 AP by sending a debug certificate to a target and issuing WFA request #2.

This variable should contain the full path to the debug certificate binary file.

7.3.2 DEBUG_CERTIFICATE_ADDR

(Optional) Allows to specify the location in the RAM where the debug certificate will be loaded. The default address 0x2000FC00 will be used if this variable is not set by the user.

7.3.3 SMIF_BANKS

See SMIF_BANKS section under PSoC 6™ MCU.

7.4 XMC7xxx and TRAVEO™ T2G global variables

7.4.1 ENABLE_ACQUIRE

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

Possible values:

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

7.4.2 ENABLE_POWER_SUPPLY

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

Possible values:

- 0 – Power supply disabled
- Any other value defines target voltage in millivolts.
- default – Sets the last used voltage before KitProg3/MiniProg4 was powered off.

7.4.3 ENABLE_CM71

Allows specifying the CPU cores to be visible to OpenOCD. Useful for single core XMC7xxx devices. OpenOCD never affects disabled cores.

Possible values:

- 1 – CM71 core is enabled.
- 0 – Core is disabled.

7.4.4 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` – XIP mapping address
- `BankSzN` – Total size of this flash bank, in bytes
- `ProgramSzN` – Minimal programming granularity (program block size), in bytes
- `EraseSzN` – Minimal erase granularity (erase block size), in bytes

8 Usage examples

All the examples in this chapter assume that you have a PSoC™ 6 MCU target 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\openocd\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 the default configuration suitable for the majority of PSoC™ 6 MCU kits:

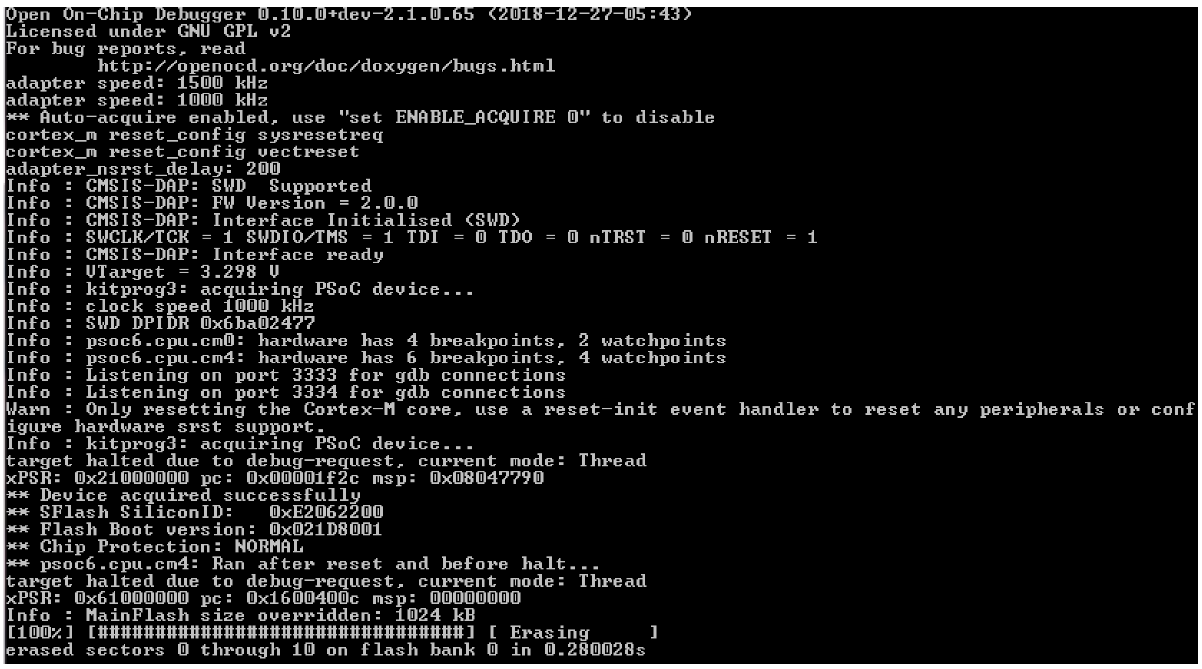
```
source [ find interface/kitprog3.cfg ]
source [ find target/psoc6.cfg ]
init
reset init
```

See Supported target configurations for a detailed list of available target devices and corresponding OpenOCD configuration files.

8.1 Erase main flash rows 0…10 of PSoC™ 6 MCU

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

A possible output of OpenOCD:

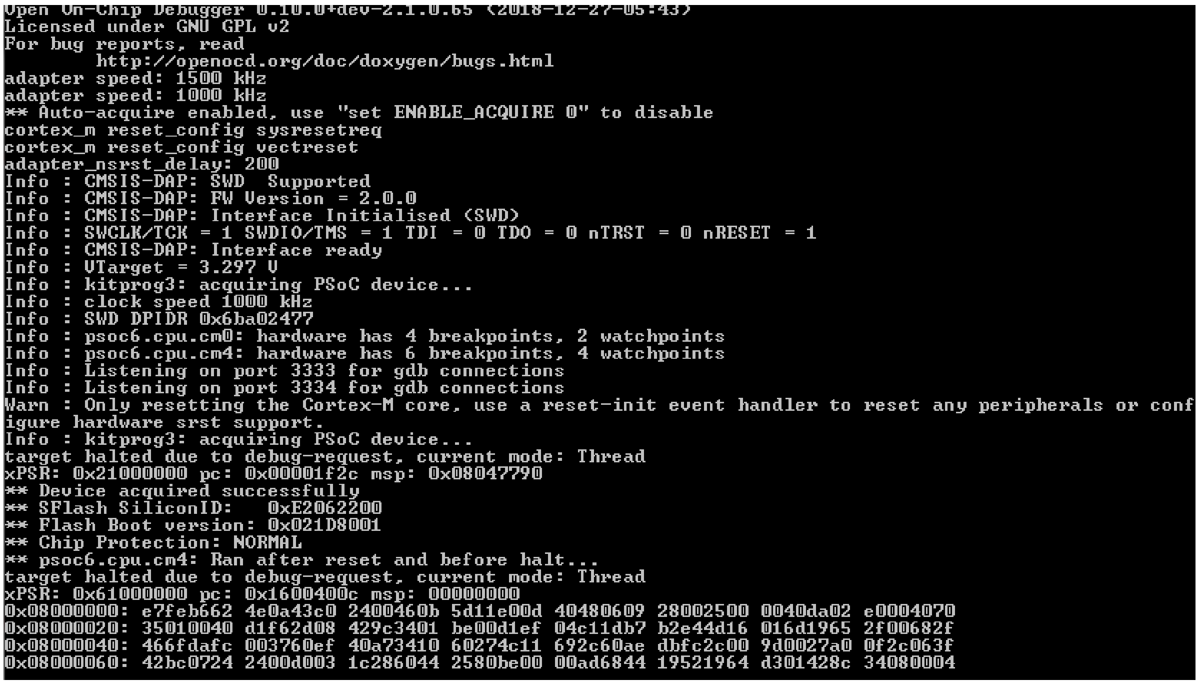


```
Open On-Chip Debugger 0.10.0+dev-2.1.0.65 (2018-12-27-05:43)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
adapter speed: 1000 kHz
** Auto-acquire enabled, use "set ENABLE_ACQUIRE 0" to disable
cortex_m reset_config sysresetreq
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
```

8.2 Display memory contents (32 words at address 0x08000000) of PSoC™ 6 MCU

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

A possible output of OpenOCD:



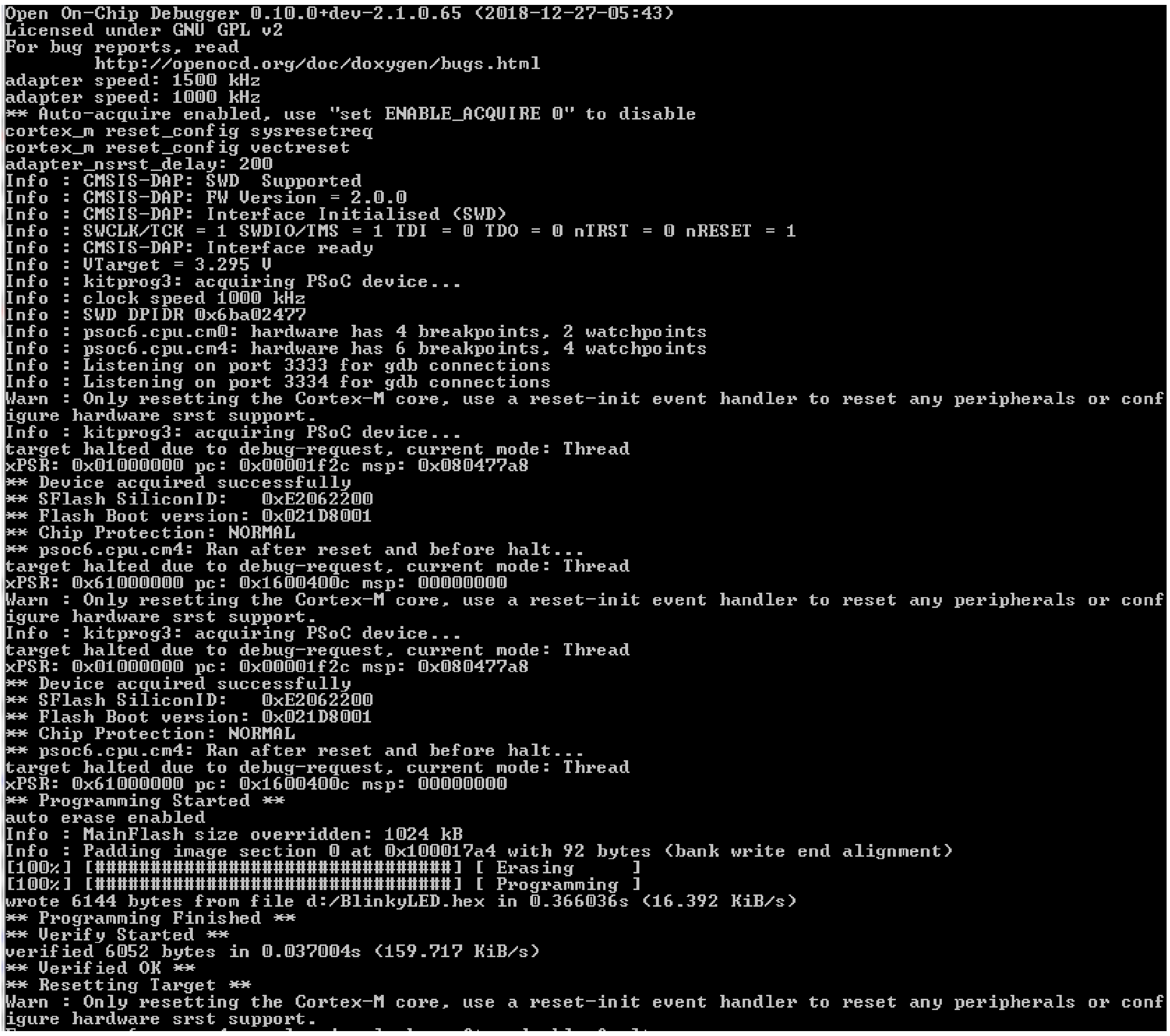
```
Open On-Chip Debugger 0.10.0+dev-2.1.0.65 (2018-12-27-05:43)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
adapter speed: 1000 kHz
** Auto-acquire enabled, use "set ENABLE_ACQUIRE 0" to disable
cortex_m reset_config sysresetreq
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
```

8.3 Program the PSoC™ 6 MCU 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:



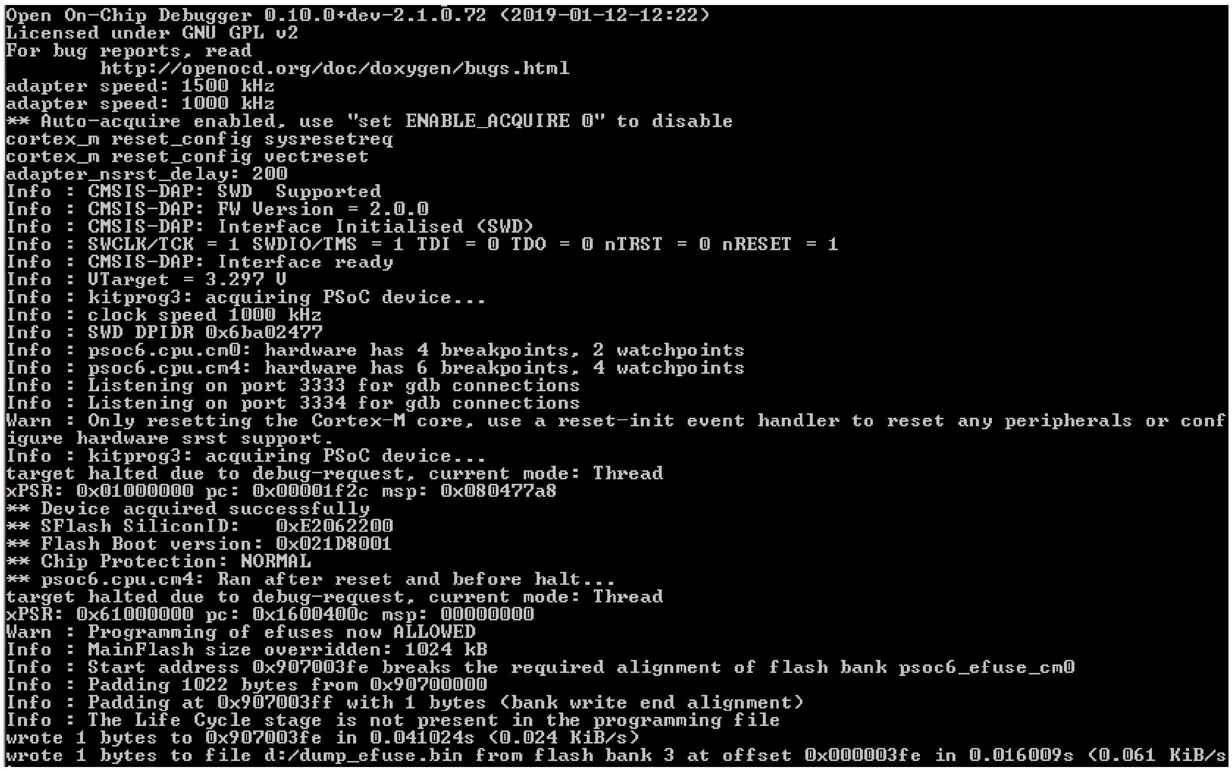
```
Open On-Chip Debugger 0.10.0+dev-2.1.0.65 (2018-12-27-05:43)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
adapter speed: 1000 kHz
** Auto-acquire enabled, use "set ENABLE_ACQUIRE 0" to disable
cortex_m reset_config sysresetreq
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.
```

8.4 Program the EFuse region of PSoC™ 6 MCU

This example writes a single bit of data to the EFuse region of the PSoC™ 6 MCU 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:

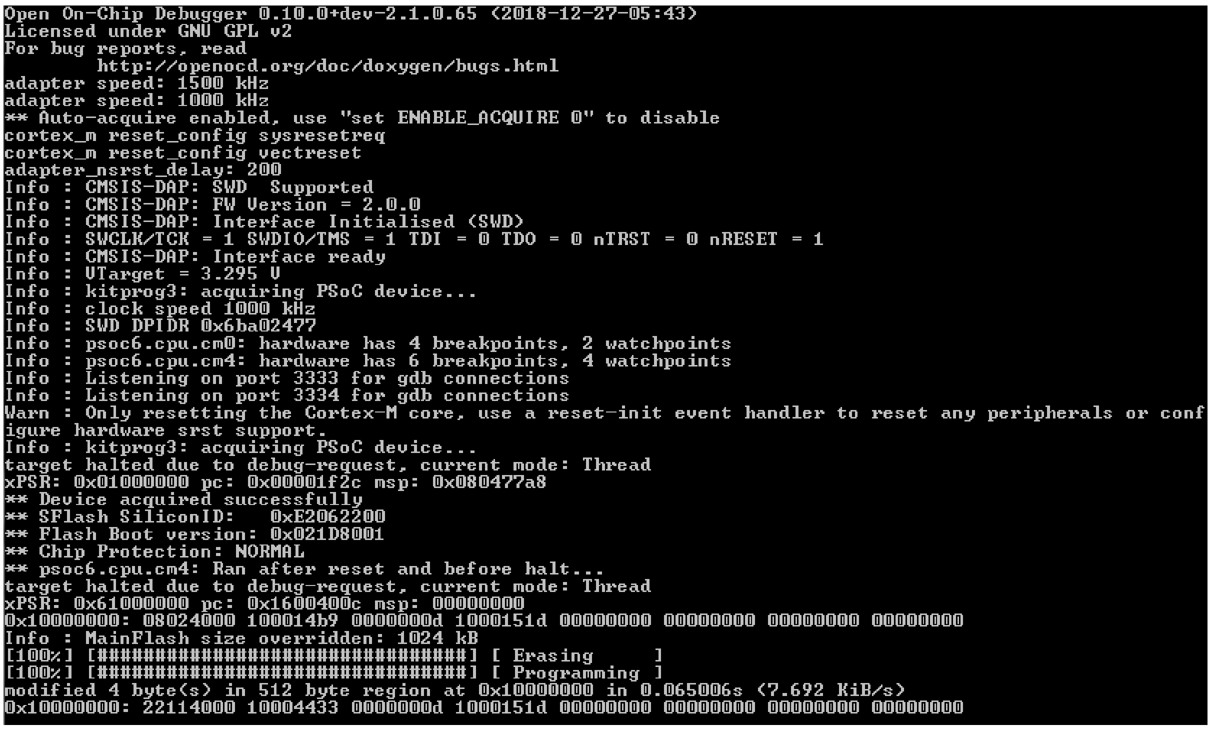


```
Open On-Chip Debugger 0.10.0+dev-2.1.0.72 (2019-01-12-12:22)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
adapter speed: 1000 kHz
** Auto-acquire enabled, use "set ENABLE_ACQUIRE 0" to disable
cortex_m reset_config sysresetreq
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
```

8.5 Modify individual bytes of PSoC™ 6 MCU 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:



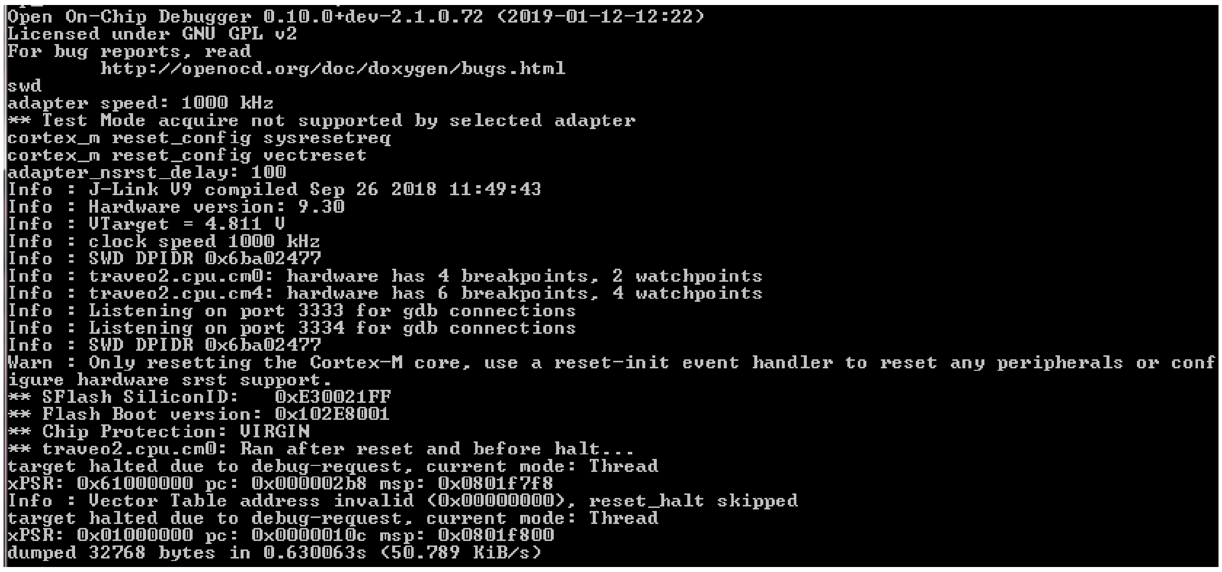
```
Open On-Chip Debugger 0.10.0+dev-2.1.0.65 (2018-12-27-05:43)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 1500 kHz
adapter speed: 1000 kHz
** Auto-acquire enabled, use "set ENABLE_ACQUIRE 0" to disable
cortex_m reset_config sysresetreq
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
```

8.6 Read the memory of PSoC™ 6 MCU to binary file

The example reads 32 KB of the PSoC™ 6 MCU memory to a file named *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:

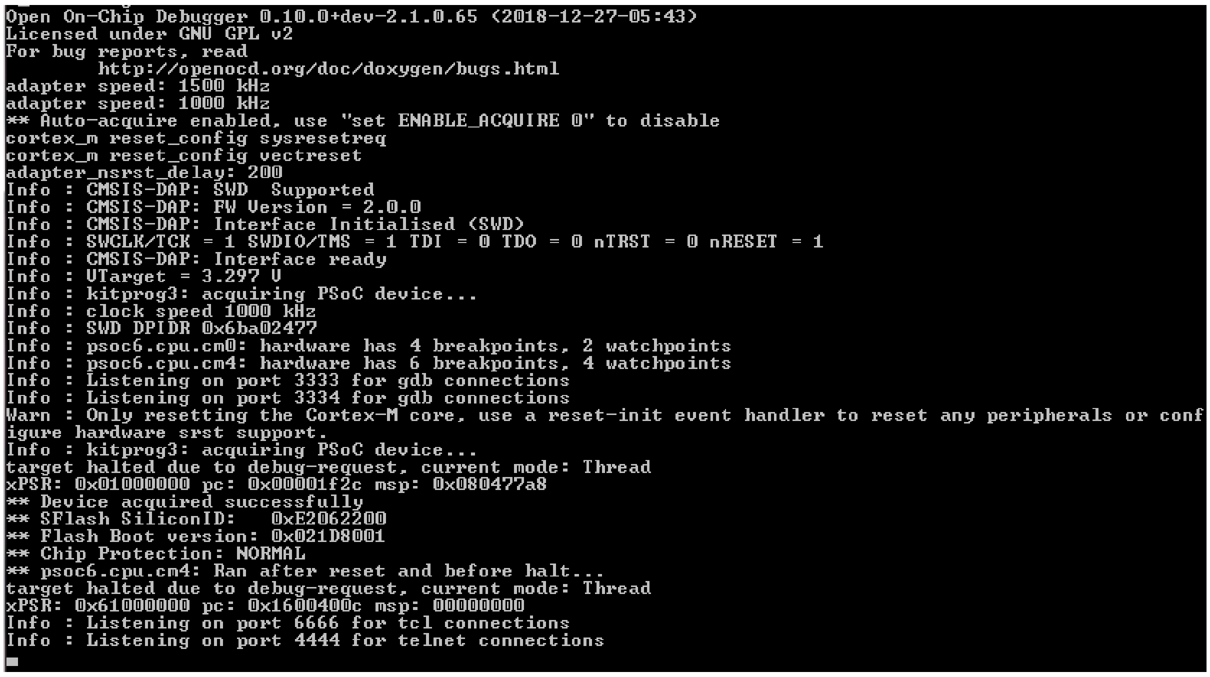


```
Open On-Chip Debugger 0.10.0+dev-2.1.0.72 (2019-01-12-12:22)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
swd
adapter speed: 1000 kHz
** Test Mode acquire not supported by selected adapter
cortex_m reset_config sysresetreq
cortex_m reset_config 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)
```

8.7 Start the 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
```

Revision history

Revision	Date	Description
**	2019-01-17	New document.
*A	2019-01-24	Updated installation procedures for Windows and Linux sections.
*B	2019-04-16	Updated for version 2.2. Added CY8C6xx5 configuration. Added descriptions for *show_verify_ranges*, *clear_verify_ranges*, and *psoc6 secure_app* commands.
*C	2019-06-26	Removed all Traveo II (automotive) related information
*D	2019-12-13	Changed document name from CYPRESS™ Programmer 2.2 OpenOCD CLI User Guide to Cypress OpenOCD CLI User Guide. Clean-up in whole document. Deleted section 2 – Cypress Programmer Installation. Updated Supported Target Configurations table. Added description for "TARGET_AP" and "FLASH_RESTRICTION_SIZE" variables. Added "psoc6 secure_acquire" and deleted "psoc6 secure_app" commands. Added "cmsis_dap_serial" command description.
*E	2020-03-16	Added PSoC 4-related descriptions.
*F	2020-03-19	Added "flash protect" command description. Added description of ENABLE_WFLASH, ENABLE_SFLASH, ENABLE_EFUSE variables.
*G	2020-06-11	Added mention of the latest released version of Cypress OpenOCD is located on GitHub: https://github.com/cypresssemiconductorco/openocd/releases Updated section "Supported MCU Devices" Updated section "Supported Target Configurations" Updated description of "kitprog3 acquire_config" and "psoc6 secure_acquire" commands
*H	2020-07-29	Added configuration file for CY8C6xx4 device
*I	2020-02-08	Added description for "psoc4 chip_protect" command Added configuration file for PSoC 4500H Updated "flash erase_sector" and "flash erase_address" sections – document limitations of PSoC4 chips Updated "PSOC4_USE_ACQUIRE", "ENABLE_ACQUIRE", "kitprog3 acquire_config" sections – add support for power-cycle acquisition mode
*J	2021-09-10	Updated document title to ModusToolbox™ software OpenOCD CLI user guide Added description for the cmsis_flash driver and related commands Added description for global variables used with CYW20829 target Added CYW20829-specific commands Added CYW4390x-specific commands and limitations Added example of remote PSoC6 debugging configuration
*K	2022-08-30	The “kitprog3 power_control” command is now deprecated Support for XMC7xxx and TRAVEO™ T2G Body High devices Removed PSOC4_USE_MEM_AP variable as it is no longer required Added section 7.4 XMC7xxx and TRAVEO™ T2G global variables Updated section 6.3 with new CAT1C-related commands
*L	2023-05-05	Updated list of supported operating systems

Edition 2023-05-05

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Email: erratum@infineon.com

Document reference

002-26234 Rev. *L