OpenOCD Quick Reference Card

Rev. 2011-02-21

Homepage: `http://openocd.berlios.de`

OpenOCD revision (git):
0.5.0-dev-00757-g3f4b9e3 2011-02-16-11:25

Card Content

Startup

`openocd -h` Print help

`[sudo] openocd -d <level>` Start server with default config `openocd.cfg`. Debug level 0 to 3 (0: error msgs, 1: warnings, 2: info msgs, 3: debug msgs, default is 2)

`[sudo] openocd -d <level> -f <config1> -f <config2> ...` Start server with multiple configuration files, e.g. for interface and board.

Configuration file directories for `board/`, `chip/`, `cpu/`, `interface/`, `target/` in `$INSTDIR/share/openocd/scripts/`

Example `[sudo] openocd -d <level> -f interface/arm-usb-ocd.cfg -f board/olimex_sam7_ex256.cfg`

Debugging

`log_output file_name` redirect logging to a file (default: stderr) (command valid any time)

`ms` Returns ever increasing milliseconds. Used to calculuate differences in time. (command valid any time)

`find [file]` print full path to file according to OpenOCD search rules (command valid any time)

`version` Show program version.

`reg [(register_name|register_number) [value]]` display or set a register; with no arguments, displays all registers and their values.

`targets [target]` change current default target (one parameter) or prints table of all targets (no parameters) (command valid any time)

`wait_halt [milliseconds]` wait up to the specified number of milliseconds (default 5) for a previously requested halt

`halt [milliseconds]` request target to halt, then wait up to the specified number of milliseconds (default 5) for it to complete

`resume [address]` Resume the target at the current position or at *address*.

`stacktrace` returns the stacktrace as a list of triples: proc, file, line.The stack trace is reset when a new stack trace is being built after a new failure has occurred. (command valid any time)

`step [address]` Step one instruction from current PC or *address*.

`reset [run|halt|init]` Reset all targets into the specified mode. Default reset mode is run, if not given.

`reset_nag ['enable'|'disable']` Nag after each reset about options that could have been enabled to improve performance. (command valid any time)

`soft_reset_halt` Halt the target and do a soft reset.

`md[whb] ['phys'] address [count]` Display *count* words (32 bit), half-words (16 bit) or bytes at *address*. If *count* is omitted, one element is displayed.

`mw[whb] ['phys'] address value [count]` Write *value* at the word, half-word or byte location *address*.

`bp [address length ['hw']]` list or set hardware or software breakpoint. Length is 2 (thumb) or 4 byte (arm).

`rbp address` Remove breakpoint at address.

`wp [address length [('r' | 'w' | 'a') value [mask]]]` list (no params) or create watchpoints

`rwp address` Remove watchpoint at address.

`load_image filename address ['bin' | 'ihex' | 'elf' | 's19'] [min_address] [max_length]` Load image file to address.

`fast_load` loads active fast load image to current target - mainly for profiling purposes

`fast_load_image filename address ['bin' | 'ihex' | 'elf' | 's19' ] [min_address [max_length]]` Load image into server memory for later use by `fast_load`; primarily for profiling (command valid any time).

`dump_image <filename> <address> <size>` Dump *size* bytes of target memory starting at *address* to a (binary) *file*.

Reset

`jtag_ntrst_delay [milliseconds]` delay after deasserting trst in ms (command valid any time)

`jtag_ntrst_assert_width [milliseconds]` delay after asserting trst in ms (command valid any time)

`reset_config [none | trst_only | srst_only | trst_and_srst] [srst_pulls_trst | trst_pulls_srst | combined | separate] [srst_gates_jtag | srst_nogate] [trst_push_pull | trst_open_drain] [srst_push_pull | srst_open_drain]` configure adapter reset behavior (command valid any time)

`jtag arp_init`

`jtag arp_init-reset`

NOR Flash Commands

`flash banks` Display table with information about flash banks. (command valid any time).

`flash list` Returns a list of details about the flash banks. (command valid any time).

`flash probe num` Identify a flash bank.

`flash erase_sector bank_id first last` Erase a range of sectors in a flash bank.

`flash erase_address ['pad'] ['unlock'] address length`

Erase flash sectors starting at address and continuing for length bytes. If 'pad' is specified, data outside that range may also be erased: the start address may be decreased, and length increased, so that all of the first and last sectors are erased. If 'unlock' is specified, then the flash is unprotected before erasing.

`flash fillw address word length` Fill n words with 32-bit value, starting at word address. (No autoerase.)

`flash fillh address halfword length` Fill n halfwords with 16-bit value, starting at word address. (No autoerase.)

`flash fillb address byte length` Fill n bytes with 8-bit value, starting at word address. (No autoerase.)

`flash write_bank num filename offset` Write binary data from file to flash bank, starting at specified byte offset from the beginning of the bank.

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

Write an image to flash. Optionally first unprotect and/or erase the region to be used. Allow optional offset from beginning of bank (defaults to zero)

`flash erase_check num` Check erase state of all blocks in a flash bank.

`flash info num` Print information about a flash bank.

`flash protect bank_id first_sector [last_sector | 'last'] ('on' | 'off')`

Turn protection on or off for a range of sectors in a given flash bank.

Note that each flash driver also adds specific commands, e.g. `at91sam3 gpnvm` and many others.

NOR Flash Configuration

`flash bank bank_id driver_name base_address size_bytes chip_width_bytes bus_width_bytes target [driver_options ...]`

Define a new bank with the given name, using the specified NOR flash driver. (configuration command)

Flash drivers are: cfi, stmsmi, aduc702x, at91sam3, at91sam7, avr, ecosflash, lpc2000, lpc288x, lpc2900, ocl, pic32mx, stellaris, stm32x, str7x, str9x, str9xpec, tms470.

`flash init`

Initialize flash devices. (configuration command)

AT91SAM7 specific commands

`flash erase <num> first_plane last_plane`

`at91sam7 gpnvm <num> <bit> <'set'|'clear'>` set or clear at91sam7 gpnvm bit

LPC2000 specific commands

`lpc2000 part_id <num>` print part id of lpc2000 flash bank *num*.

STM32x specific commands

`stm32x lock <num>` lock stm32 device
`stm32x unlock <num>` unlock stm32 device
`stm32x options_read <num>` read stm32 option bytes
`stm32x options_write <num> <SWWDG|HWWDG> <RSTSTNDBY|NORSTSTNDBY> <RSTSTOP|NORSTSTOP>` write stm32 option bytes
`stm32x mass_erase <num>` mass erase flash memory

STR7X specific commands

`str7x disable_jtag <bank>`
`str7x disable_jtag <bank> ok`

STR9 specific commands

str9xpec enable_turbo <num> enable turbo mode, simply this will remove the str9 from the chain and talk directly to the embedded flash controller
str9xpec disable_turbo <num> restore the str9 into jtag chain
str9xpec lock <num> lock str9 device. The str9 will only respond to an unlock command that will erase the device.
str9xpec unlock <num> unlock str9 device
str9xpec options_read <num> read str9 option bytes
str9xpec options_write <num> write str9 option bytes

STR9XPEC option byte config

str9xpec options_cmap <num> <bank0|bank1> configure str9 boot bank
str9xpec options_lvdthd <num> <2.4v|2.7v> configure str9 lvd threshold
str9xpec options_lvdsel <num> <vdd|vdd_vddq> configure str9 lvd source
str9xpec options_lvdwarn <bank> <vdd|vdd_vddq> configure str9 lvd reset warning source
str9xpec part_id <num> print part id of str9xpec flash bank num

Stellaris

stellaris gpnvm <num> <bit> <'set'|'clear'> set or clear stellaris gpnvm bit.

TMS470

tms470 flash_keyset <key0> <key1> <key2> <key3> TMS470 flash_keyset.
tms470 osc_megahertz <MHz>
tms470 plldis <0|1>

Ecosflash

ecosflash (*)

NAND Flash

nand list

nand probe num

nand dump num filename offset length [oob_option]

nand erase num [offset length]

nand write num filename offset [option...]

nand verify num filename offset [option...]

nand check_bad_blocks num [offset length]

nand info num

nand raw_access num ('enable'|'disable')

NAND Flash Configuration

nand device name driver target [configparams ...]

Current **NAND Drivers**: davinci, lpc3180, lpc32xx, orion, s3c2410, s3c2412, s3c2440, s3c2443, s3c6400, imx27, imx31, at91sam9, nuc910.

LPC3180 NAND Flash

lpc3180 select 'mlc'|'slc'

ARM

arm core_state ['arm' | 'thumb'] display/change ARM core state

arm disassemble address [count ['thumb']] disassemble instructions

arm mcr cpnum op1 CRn op2 CRm value write co-processor register

arm mrc cpnum op1 CRn op2 CRm read coprocessor register

arm reg display ARM core registers

arm semihosting ['enable' | 'disable'] activate support for semihosting operations

ARM7/9

arm7_9 dbgrq ['enable' | 'disable']
arm7_9 dcc_downloads ['enable' | 'disable']
arm7_9 fast_memory_access ['enable' | 'disable']

ARM720

arm720t cp15 opcode [value]

ARM9

arm9 vector_catch ['all' | 'none' | list]

ARM920T

arm920t cache_info
arm920t cp15 regnum [value]
arm920t cp15i opcode [value [address]]
arm920t read_cache filename
arm920t read_mmu filename

ARM926EJ-S

arm926ejs cache_info

ARM966E

arm966e cp15 regnum [value]

XScale

xscale analyze_trace
xscale cache_clean_address address
xscale cache_info
xscale cp15 regnum [value]
xscale debug_handler target address
xscale dcache ['enable' | 'disable']
xscale dump_trace filename
xscale icache ['enable' | 'disable']
xscale mmu ['enable' | 'disable']
xscale trace_buffer ['enable' | 'disable' ['fill' [n] | 'wrap']]
xscale trace_image filename [offset [type]]
xscale vector_catch [mask]

`xscale vector_table [('low' | 'high') index value]`

ARM11 (ARM v6)

`arm11 memwrite burst ['enable' | 'disable']`
`arm11 memwrite error_fatal ['enable' | 'disable']`
`arm11 step_irq_enable ['enable' | 'disable']`
`arm11 vcr [value]`

ARMv7 Debug Access Port

`dap apid [num]`
`dap apsel [num]`
`dap baseaddr [num]`
`dap info [num]`
`dap memaccess [value]`

Cortex-M3 specific commands

`cortex_m3 maskisr ('on' | 'off')`
`cortex_m3 vector_catch ['all' | 'none' | 'list']`
`cortex_m3 reset_config ('srst' | 'sysresetreq' | 'vectreset')`

JTAG Commands

`jtag names`

`scan_chain` Print scan chain configuration.

`drscan tap_name [num_bits value]* ['-endstate' state_name]` Execute Data Register (DR) scan for one TAP. Other TAPs must be in BYPASS mode.

`flush_count` Returns the number of times the JTAG queue has been flushed.

`irscan [tap_name instruction]* ['-endstate' state_name]` Execute Instruction Register (DR) scan. The specified opcodes are put into each TAP's IR, and other TAPs are put in BYPASS.

`jtag_reset trst_active srst_active` Set reset line values. Value '1' is active, value '0' is inactive.

`pathmove start_state [next_state ...]`

`runtest num_cycles` Move to Run-Test/Idle, and issue TCK for num_cycles.

`verify_ircapture ['enable' | 'disable']` Display or assign flag controlling whether to verify values captured during Capture-IR. (command valid any time)

`verify_jtag ['enable' | 'disable']` Display or assign flag controlling whether to verify values captured during IR and DR scans. (command valid any time)

`jtag_flush_queue_sleep [sleep in ms]` For debug purposes(simulate long delays of interface) to test performance or change in behavior. Default 0ms. (command valid any time)

Adapter

`interface driver_name` (cfg) Select a debug adapter interface (driver) (configuration command). The current driver names are `amt_jtagaccel`, `arm-jtag-ew`, `at91rm9200`, `dummy`, `ep93xx`, `ft2232`, `usb_blaster`, `gw16012`, `jlink`, `parport`, `presto`, `rlink`, `usbprog`, `vsllink`, `ZY1000`.

Each adapter adds specific configuration commands. Only the commands for ft2232 are shown in section *Daemon / Configuration / FT2232* below.

`interface_transports transport ...` Declare transports the interface supports. (configuration command)

`interface_list` List all built-in debug adapter interfaces (drivers) (command valid any time)

`adapter_khz [khz]` With an argument, change to the specified maximum jtag speed. For JTAG, 0 KHz signifies adaptive clocking. With or without argument, display current setting. (command valid any time)

See also: `jtag_hkz`

Note: Max. JTAG-Clock $\approx \frac{1}{6} \times$ CPU-Clock.

`jtag_rclk [fallback_speed_khz]` With an argument, change to to use adaptive clocking if possible; else to use the fallback speed. With or without argument, display current setting. (command valid any time)

`adapter_name` Returns the name of the currently selected adapter (driver) (command valid any time)

`adapter_nsrst_assert_width [milliseconds]` delay after asserting SRST in ms (command valid any time)

`adapter_nsrst_delay [milliseconds]` delay after deasserting SRST in ms (command valid any time)

`power 'on' | 'off'` Turn power switch to target on/off. No arguments: print status

`transport init` Initialize this session's transport (command valid any time)

`transport list` List all built-in transports (command valid any time)

Typical transports are JTAG, SWD and SPI.

`transport select [transport_name]` Select this session's transport (command valid any time)

Target

`target count` Returns the number of targets as an integer (DEPRECATED) (command valid any time)

`target create name type '-chain-position' name [options ...]` Creates and selects a new target (command valid any time)

`target configure configparams...` The options accepted by this command may also be specified as parameters to 'target create'. Their values can later be queried one at a time by using the '$target_name cget' command.

`$target_name configure configparams...`

`$target_name arp_examine`

`$target_name arp_halt`

`$target_name arp_poll`

`$target_name arp_reset`

`$target_name arp_waitstate`

`$target_name array2mem arrayname width address count`

`$target_name mem2array arrayname width address count`

`$target_name cget queryparm`

`target current` Returns the currently selected target (command valid any time)

`target names` Returns the names of all targets as a list of strings (command valid any time)

`target init` initialize targets (configuration command)

`target types` Returns the available target types as a list of strings (command valid any time)

Currently supported CPU types: arm11, arm720t, arm7tdmi, arm920t, arm926ejs, arm966e, arm9tdmi, avr, cortex_a8, cortex_m3, dragonite, dsp563xx, fa526, feroceon, mips_m4k, xscale.

`target_request debugmsgs ['enable' | 'charmsg' | 'disable']` display and/or modify reception of debug messages from target

`trace history ['clear' | 'count']`

`trace point ['clear' | 'identifier']`

Command

`command mode [command_name ...]` Returns the command modes allowed by a command:'any', 'config', or 'exec'.

`command type command_name [...]` Returns the type of built-in command:'native', 'simple', 'group', or 'unknown'.

Miscellaneous

`echo [-n] string` Logs a message at "user" priority. Output message to stdout. Option "-n" suppresses trailing newline (command valid any time)

`add_usage_text command_name usage_string` Add new command usage text; command can be multiple tokens. (command valid any time)

`srst_deasserted` Overridable procedure run when srst deassert is detected. Runs 'reset init' by default. (command valid any time)

`verify_image filename [offset [type]]` XXX what does it do?

`test_image filename [offset [type]]` XXX what does it do?

`add_help_text command_name helptext_string` Add new command help text; Command can be multiple tokens. (command valid any time)

`add_script_search_dir` dir to search for config files and scripts (command valid any time)

`noinit` Prevent 'init' from being called at startup. (configuration command)

gdb_sync Next stepi will return immediately allowing GDB to fetch register state without affecting target state (command valid any time)

`virt2phys virtual_address` translate a virtual address into a physical address (command valid any time)

`profile` profiling samples the CPU PC

`power_restore` Overridable procedure run when power restore is detected. Runs 'reset init' by default. (command valid any time)

Daemon

sleep milliseconds ['busy'] Sleep for specified number of milliseconds. "busy" will busy wait instead (avoid this). (command valid any time)

`shutdown` shut the server down (command valid any time)

`debug_level <n>` Sets the verbosity level of debugging output. 0 shows errors only; 1 adds warnings; 2 (default) adds other info; 3 adds debugging.

`log_output <file>`

`script <file>` filename of OpenOCD script (tcl) to run (command valid any time)

`gdb_detach [resume|reset|halt|nothing]` NEW

`exit` Exit telnet session.

`help [command_name]` Show full command help; command can be multiple tokens. (command valid any time)

`usage` Show command usage.

Daemon Configuration

`init` (cfg) Initializes configured targets and servers. Changes command mode from CONFIG to EXEC. Unless 'noinit' is called, this command is called automatically at the end of startup. (command valid any time)

`jtag_init` (proc)

`gdb_port [port_num]` (def. 3333) Normally gdb listens to a TCP/IP port. Each subsequent GDB server listens for the next port number after the base port number specified. No arguments reports GDB port. "pipe" means listen to stdin output to stdout, an integer is base port number, "disable" disables port. Any other string is are interpreted as named pipe to listen to. Output pipe is the same name as input pipe, but with 'o' appended. (command valid any time)

`tcl_port [port_num]` Specify port on which to listen for incoming Tcl syntax. Read help on `'gdb_port'`. (configuration command)

`telnet_port <port>` (def. 4444) Listen for telnet connections on *port*

`gdb_breakpoint_override ('hard' | 'soft' | 'disable')` Display or specify type of breakpoint to be used by gdb 'break' commands. (command valid any time)

`gdb_flash_program <'enable'|'disable'>` Enable or disable flash program (configuration command)

`gdb_memory_map <'enable'|'disable'>` Enable or disable memory map (configuration command).

`gdb_report_data_abort ('enable'|'disable')` Enable or disable reporting data aborts (configuration command)

`poll ['on'|'off']` Print information about the current target state. If the target is in debug mode, architecture specific information about the current state are printed. Enable or disable continuous polling with optional parameter.

FT2232

`ft2232_device_desc <description>` USB device *description*. Use `usbview` or similar tool to get this string.

`ft2232_serial serial_string` set the serial number of the FTDI FT2232 device (configuration command)

`ft2232_layout layout` Some `layout` values are `usbjtag`, `jtagkey`, `jtagkey_prototype_v1`, `oocdlink`, `olimex-jtag`, `signalyzer`, *flyswatter*, `turtelizer2`, `comstick`, `evb_lm3s811` .

`ft2232_vid_pid vid pid` the vendor ID and product ID of the FTDI FT2232 device (configuration command)

`ft2232_latency <msec>` set the FT2232 latency timer to a new value (configuration command)

XSVF

`xsvf <devnum> <file>` Program Xilinx Coolrunner CPLD

XSVF

- **Eproo OpenOCD USB Adapter** (JTAG + RS-232)
`http://www.eproo.net`
- **ARM-USB-JTAG**, **ARM-USB-TINY**
`http://www.olimex.com`
- **Signayzer**
`http://www.xverve.com`
- **Turtelizer**
`http://www.ethernut.de/en/hardware/turtelizer`
- **Versaloon**
`http://www.versaloon.com`
- **OOCDLink/Small** (Jörn Kaipf)
`http://www.joernonline.de`
- **USBprog**
`http://code.google.com/p/usbprog-jtag/`
- many others

Compilation

```
./bootstrap
./configure [options]
make
make install
```

Example (for JLink and and FT2232-based adapters, installation in `/home/user/local/`)

```
./configure --prefix=/home/user/local/
--enable-jlink --enable-ft2232_libftdi
```

Print available options with `./configure --help`.

Commandline Options

```
Open On-Chip Debugger
   0.5.0-dev-00746-g177fe9d-dirty
   (2011-02-14-15:25)
Licensed under GNU GPL v2
For bug reports, read
http://openocd.berlios.de/doc/doxygen/bugs.html
Open On-Chip Debugger
Licensed under GNU GPL v2
--help       | -h display this help
--version    | -v display OpenOCD version
--file       | -f use configuration file <name>
--search     | -s dir to search for config files and scripts
--debug      | -d set debug level <0-3>
--log_output | -l redirect log output to file <name>
--command    | -c run <command>
```

GDB

Startup (replace `arm-gdb` with your debugger name)
`arm-gdb [-x <file>]` Execute GDB commands from file.
`arm-gdbtui` Text user interface
`cgdb -d arm-gdb -x <file>` Run cgdb front end.
`ddd --debugger arm-gdb -x <file>` Run graphical ddd front end.

IDEs with gdb integration: Codeblocks, Anjuta, Eclipse.

Some GDB commands

`(gdb) target remote localhost:3333` contact target at `localhost`, portnumber 3333.

`(gdb) monitor reset halt`

`(gdb) set remote hardware-watchpoint-limit` 2 only two hardware breakpoints available

`(gdb) set remote hardware-breakpoint-limit` 2 only two hardware breakpoints available

`set mem inaccessible-by-default off`

Telnet

Start `telnet` when OpenOCD is running:

```
hhoegl@egg:$ telnet localhost 4444
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Open On-Chip Debugger
>
```

Type `help` on the telnet prompt to see a list of available commands. Type `exit` to quit.

Tcl Scripting

XXX to do

Jim Tcl Dokumenyation in OpenOCD installation: /local/doc/jim/Tcl.html
Jim shell `local/bin/jimsh`

Python Scripting

The telnet interface can easily be automated by a program. High-level language Python (`www.python.org`) demonstrates this.
Example using `telnetlib`:

```
import telnetlib
tn = telnetlib.Telnet("localhost", 4444)
TIMEOUT = 1.0

def mww(addr, data):
    tn.write("mww %s %s\n" % (addr, data))
    (result, mobj, str) = tn.expect(["(.*)\n>"],\
                                    TIMEOUT)
    return

def mdw(addr):
    tn.write("mdw %s\n" % addr)
    (result, mobj, str) = tn.expect(\
              ["(.*)\n(.*):(.*)\n>"], TIMEOUT)
        if mobj:
            return string.atoi(\
              mobj.groups()[2].strip(), 16)
```

Example using `pexpect`:

```
import pexpect

def mww(addr, data):
    child.sendline("mww 0x%x 0x%x" % \
                      (addr, data))
    child.expect("> ")

def mdw(addr):
    child.sendline("mdw 0x%x" % addr)
    child.expect("> ")
    n = child.before.find(':')
    return string.atoi(child.before[n+1:], 16)
```

Further information

- Get **current version** from git repository

```
GIT=openocd.git.sourceforge.net
git clone git://$GIT/gitroot/openocd/openocd
```

- For **OpenOCD documentation** in texinfo format see directory `<INSTDIR>/share/info`. Read online with GNU info reader `info openocd.info`.

- **OpenOCD User Forum** at Spark Fun Electronics
`http://forum.sparkfun.com/viewforum.php?f=18`

- **Mailing List "openocd-development"**
`http://lists.berlios.de/mailman/listinfo/openocd-development`

- **Yagarto ARM toolchain for Windows**
`http://www.yagarto.de`

- Martin Thomas, **Accessing ARM-Controllers with OpenOCD** `http://www.siwawi.arubi.uni-kl.de/avr_projects/arm_projects/openocd_intro/index.html`

QuickRef written by Hubert.Hoegl@hs-augsburg.de

Download this QuickRef from
http://elk.informatik.fh-augsburg.de/pub/openocd-quickref/