OpenOCD Quick Reference Card

OpenOCD Homepage
`http://openocd.berlios.de`
Current OpenOCD revision: 128

Server

Configuration Commands

`telnet_port <port>` Listen for telnet connections on *port*.

`gdb_port <port>` Listen for GDB connections on *port*, *port+1*, ... (target 1, target 2, ...)

User Commands

`shutdown` Shut server down.

`exit` Exit telnet. Leaves server running.

`help` Print available commands.

Interpreter

The interpreter commands may be used to define variables used within other subsystems like JTAG.
`var <name> ['del'|([size1] [sizeN])]`
Allocate, display or delete variable. Allocation has to define the size for all num_fields elements.

`field <var> <field> [value| 'flip' ]`
Display or modify variable field.

`script <file>`
Execute commands from file.

Target

Configuration Commands

`target <type> <endianness> <reset_mode>`
Target *type* is arm7tdmi, arm9tdmi, arm720t, arm920t, arm926ejs, arm966e; *endianness* is either big or little; *reset_mode* is one of reset_halt, reset_run, reset_init, run_and_halt, run_and_init. **Do not use reset_halt or reset_init on LPC2 or STR7**.

`target arm7tdmi <endianness> <reset_mode> <jtag#> [variant]` *variant* is one of arm7tdmi_r4, arm7tdmi-s_r4, lpc2000.

`target arm9tdmi <endianness> <reset_mode> <jtag#> [variant]` *variant* is one of arm920t, arm922t and arm940t.

`daemon_startup <'reset'|'attach'>` Describes what to do with target on daemon startup.

`target_script <target#> <event> <scriptfile>` *event* is either post_halt or pre_resume.

`run_and_halt_time <target#> <time>` Delay in msec between reset and debug request.

`working_area <target#> <addr> <size> [backup|nobackup]`

User Commands

`targets [num]` Display list of configured targets, or make target *num* the current target.

`reg [#|name] [value|'force']` Display or modify registers.

`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.

`wait_halt` Wait for target halt

`halt` Halt target

`resume <address>` Resume the target at the current position or at *address*.

`step <address>` Single-step at the current position or at *address*.

`reset ['run'|'halt'|'init'|'run_and_halt'|'run_and_init']` Reset the target in a few variations.

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

`md[whb] <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] <address <value>` Write *value* at the word, half-word or byte location *address*.

`bp <address> <length> [hw]` Set a breakpoint of *length* bytes at *address*.

`rbp <address>` Remove breakpoint at address.

`wp <address> <length> <r|w|a> [value] [mask]` Set a watchpoint of *length* bytes at *address*.

`rwp <address>` Remove a watchpoint at *address*.

`load_binary <file> <address>` Load binary *file* into target memory (RAM) at *address*.

`dump_binary <file> <address> <size>` Dump target memory of *size* bytes at *address* into *file*.

ARM v4/5

`armv4_5 reg` Display all banked ARM core registers.

`armv4_5 core_mode ['arm'|'thumb']` Display the current core state, or switch between Arm and Thumb state.

`armv4_5 disassemble <address> <count> ['thumb']` Disassemble instructions

ARM720T

The commands are `cp15`, `virt2phys`, `mdw_phys`, `mdh_phys`, `mdb_phys`, `mww_phys`, `mwh_phys`, `mwb_phys`. See ARM7/9 for a short description.

ARM 7/9

`arm7_9 write_xpsr <value> <spsr>`
Write the program status register. spsr selects between the current program status register (0) and the saved program status register (1) of the current mode.

`arm7_9 write_xpsr_im8 <8bit immediate> <rotate> <not cpsr|spsr>`
Same as write_xpsr, but use the immediate operand opcode.

`arm7_9 write_core_reg <num> <mode> <value>`
Write core register num of mode with value.

`arm7_9 sw_bkpts <enable|disable>`
Enable or disable the use of software breakpoints.

`arm7_9 force_hw_bkpts <enable|disable>`
Force the use of hardware breakpoints.

`arm7_9 dbgrq <enable|disable>`
Use EmbeddedICE dbgrq instead of breakpoint for target halt requests (safe for all except ARM7TDMI-S).

`arm7_9 fast_memory_access <enable|disable>`
Use fast but potentially unsafe memory accesses instead of slow accesses. Assumes a sufficiently high clock speed.

`arm7_9 dcc_downloads <enable|disable>`
use DCC downloads for larger memory writes.

ARM920T

`arm920t cp15 <num> [value]` Display/modify CP15 register

`arm920t cp15i <opcode> [value] [address]`
Display/modify cp15 (interpreted access)

`arm920t cache_info` Display information about target caches.

`arm920t virt2phys <va>` Translate virtual to physical address.

`arm920t md[whb]_phys <physical addr> [count]` display memory word, half word, byte

`arm920t mw[whb]_phys <physical addr> <value>` write memory word, half word, byte

`arm920t read_cache <filename>` display I/D cache content

`arm920t read_mmu <filename>` display I/D mmu content

ARM926EJ-S

`arm926ejs ...` XXX To Do

ARM966E

`arm966e cp15 <num> [value]` Display/modify CP15 register

JTAG

Configuration Commands

`interface <name>` – *name* is one of parport, amt_jtagaccel, ft2232, ep93xx

`jtag_device <IR length> <IR capture> <IR mask> <IDCODE instruction>` – Describes the devices that form the JTAG daisy chain, with the first device being the one closest to TDO.

`jtag_nsrst_delay <ms>` – *ms* milliseconds delay between going nSRST inactive and following JTAG operations.

`jtag_ntrst_delay <ms>` – *ms* milliseconds delay between going nTRST inactive and following JTAG operations.

`reset_config <signals> [combination] [trst-type] [srst-type]` – *signals* is one of none, trst_only, srst_only or trst_and_srst. *combination* is one of srst_pulls_trst, trst_pulls_srst, combined, separate. *trst-type* is one of trst_open_drain, trst_push_pull. *srst-type* is one of srst_push_pull, srst_open_drain.

User & Config Commands

`jtag_speed <value>` Select JTAG Speed **ft2232**: 6 MHz / (*value*+1) **parport**: maximum speed / value **amt_jtagaccel**: 8 / 2**value

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

Parport

`parport_port <port|num>` – Either I/O *port* address (e.g. 0x378) or the *number* of the /dev/parport device.

`parport_cable <name>` – *name* is one of wiggler, old_amt_wiggler, chameleon, dlc5 (Xilinx cable III), triton.

Amt_jtagaccel

`parport_port <port|number>` see previous description.

Ft2232

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

`ft2232_layout <name>` Layout *name* is one of `jtagkey`, `usbjtag`, `signalyzer`, `olimex-jtag`, `m5960`, `evb_lm3s811` .

`ft2232_vid_pid <vid> <pid>` Vendor-ID *vid* and product-ID *pid* of the FTDI device.

User Commands

`scan_chain` Print scan chain configuration.

`endstate <tap_state>` Finish JTAG operations in *tap_state*.

`jtag_reset <trst> <srst>` Toggle reset lines.

`runtest <num_cycles>` Move to Run-Test/Idle and execute *num_cycles*.

`statemove <tap_state>` Move to current endstate or *tap_state*.

`irscan <device> <instr> [devN] [instrN]` Execute IR scan.

`drscan <device> <var> [devN] [varN]` Execute DR scan.

Flash

`flash banks` Display list of configured flash banks

`flash info <num>` Display information and list of blocks of flash bank *num*.

`flash probe <num>` Probe flash bank *num* if it matches the configured bank.

`flash erase_check <num>` Check erase state of flash sectors in bank *num*.

`flash protect_check <num>` Check protect state of flash sectors in bank *num*.

`flash erase <num> <first> <last>` Erase sectors at bank *num*, starting at sector *first* up to and including *last*. Sector numbering starts at 0.

`flash write <num> <binfile> <offset>` Write *binfile* (in **binary** format!) to bank *num* at *offset* bytes from start of bank.

`flash protect <num> <first> <last> <'on'|'off'>` Enable (on) or disable (off) protection of flash sectors *first* to *last* of flash bank *num*.

`flash bank <driver> <base> <size> <chip_width> <bus_width> [driver_options...]`
Configure a flash bank at address *base* of *size* bytes with a bus of *bus_width* bits formed by chips of *chip_width* bits size using *driver* lpc2000, cfi, at91sam7, str7x, str9x.

`flash bank lpc2000 <base> <size> 0 0 <lpc_variant> <target#> <cclk> ['calc_checksum']`
The internal flash of LPC2000 devices doesn't require chip- and buswidth to be defined. The *lpc_variant* specifies the supported IAP commands of the device (**lpc2000_v1**:2104—5—6, 2114—9, 2124—9, 2194, 2212—4, 2292—4; **lpc2000_v2**: 213x, 214x, 2101—2—3). The flash bank is part of *target#* which runs at *cclk* kHz. *Calc_checksum* inserts a valid checksum into the exception vector when this area is flashed.

For LPC2138 and LPC2148 devices, the *size* argument has to reflect the user-accessible size, i.e. 0x7d000 (500kB), not 0x80000 (512kB).

`flash bank at91sam7 0 0 0 0 <target#>`

`flash bank str7x <base> <size> 0 0 <variant> <target#>` – *variant* is one of STR71x, STR73x or STR75x.

`flash bank str9x <base> <size> 0 0 0`
The str9 needs the flash controller to be configured prior to Flash programming:
`str9x flash_config b0size b1size b0start b1start`
Example: `str9x flash_config 4 2 0 0x80000`
This will setup the BBSR, NBBSR, BBADR and NBBADR registers respectively.

`flash bank cfi <base> <size> <chip_width> <bus_width> <target#>`

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

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

XSVF

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

PLD

XXX To Do

Commandline Options

```
$ openocd  --help
Open On-Chip Debugger
(c) 2005 by Dominic Rath

--help       | -h  display this help
--file       | -f  use configuration file <name>
--debug      | -d  set debug level <0-3>
--log_output | -l  redirect log output to file <name>
--interface  | -i  use jtag interface driver <name>
```

Sample Configuration

[see $(SRCDIR)/doc/configs/*.cfg]

Some GDB commands

`(gdb) target remote localhost:3333`

`(gdb) monitor arm7_9 force_hw_bkpts enable` force hardware breakpoints

More information

- Get current version with Subversion
`svn co svn://svn.berlios.de/openocd/trunk`

- OpenOCD Forum at Spark Fun Electronics
`http://www.sparkfun.com/cgi-bin/phpbb/viewforum.php?f=18`

- OpenFacts
`http://openfacts.berlios.de/index-en.phtml?title=Open_On-Chip_Debugger`

- Mailing List "openocd-development"
`http://developer.berlios.de/mail/?group_id=4148`

- Yagarto ARM toolchain (Windows)
`http://www.yagarto.de`

- Amontec JTAGkey[-Tiny], sdk4arm
`http://www.amontec.com`

- Olimex ARM-USB-TINY, ARM-USB-OCD
`http://www.olimex.com`

QuickRef written by Hubert.Hoegl@fh-augsburg.de
Date: 2007-02-03

Download this QuickRef from

`http://www.fh-augsburg.de/~hhoegl/proj/openocd/oocd-quickref.pdf`