OpenOCD Quick Reference Card

OpenOCD Homepage
`http://openocd.berlios.de`
Current revision: 62

Server

Configuration Commands

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

`gdb_port <port>` Listen for GDB connections on port, port+1, ...

User Commands

`shutdown` Shut server down.

`exit` Exit telnet. Leaves server running.

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 or arm920t. Endianness is either big or little. Startup 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]`

`daemon_startup [reset|attach]` Describes what to do 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 num the current target.

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

`poll [on|off]` Print information about the current target state.

`halt` Request target halt.

`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 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_state [arm|thumb]` Display the current core state, or switch between arm and thumb state.

ARM v7/9

`armv7_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 download <filename> <address> <working_area>`
Download file to target ram using DCC with memory at working_area

JTAG

Configuration Commands

`interface <name>` One of parport, amt_jtagaccel, ftdi2232, ftd2xx.

`jtag_device <IR length> <IR capture> <IR mask> <IDCODE instruction>`

`reset_config <signals> [combination] [trst-type] [srst-type]` Signals is one of 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 stst_push_pull, stst_open_drain.

User & Config Commands

`jtag_speed <value>` Select JTAG Speed **ftdxxx**: 0=6MHz, 1=3MHz, ... **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 /dev/parport number

`parport_cable <name>` One of wiggler, old_amt_wiggler, chameleon, dlc5 (Xilinx cable III), triton

Amt_jtagaccel

`parport_port <port>`

Ftdi2232 (libftdi)

`ftdi2232_vid_pid <vid> <pid>` Vendor-ID, Product-ID of the FTDI device (Linux-only).

Ftd2xx (FTDIChip Library)

`ftd2xx_device_desc <description>` Find out *description* with `usbview` or similar tool.

`ftd2xx_layout <name>` Layout is one of `jtagkey`, `usbjtag`.

`ftd2xx_vid_pid <vid> <pid>` Vendor-ID, Product-ID of the FTDI device (Linux-only).

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

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

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

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

`flash erase <bank> <first> <last>` Erase blocks first to last of flash bank.

`flash write <bank> <file> <offset>` Write file to flash bank at offset.

`flash bank <driver> <base> <size> <chip_width> <bus_width>`
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.

`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. 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

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

`flash bank cfi <base> <size> <chip_width> <bus_width> <target#>`
Specifying a working_area address and size allows a lot faster execution of flash operations.

XSVF

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

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]

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

Date: 2006-05-26