

**NAME**

linuxcnc – LinuxCNC (The Enhanced Machine Controller)

**SYNOPSIS**

**linuxcnc** [-h] [-v] [-d] [-r] [-l] [-k] [-t <tpmodulename> [parameters]] [-m <homemodulename> [parameters]] [-H <dirname>] [INI file]

**DESCRIPTION**

**linuxcnc** is used to start LinuxCNC (The Enhanced Machine Controller). It starts the realtime system and then initializes a number of LinuxCNC components (IO, Motion, GUI, HAL, etc). The most important parameter is *INI file*, which specifies the configuration name you would like to run. If *INI file* is not specified, the **linuxcnc** script presents a graphical wizard to let you choose one.

**OPTIONS**

- h** Shows the help
- v** Be a little bit verbose. This causes the script to print information as it works.
- d** Print lots of debug information. All executed commands are echoed to the screen. This mode is useful when something is not working as it should.
- r** Disable redirection of stdout and stderr to ~/linuxcnc\_print.txt and ~/linuxcnc\_debug.txt when stdin is not a tty. Used when running linuxcnc tests non-interactively.
- l** Use the last-used INI file without prompting. This is often a good choice for a shortcut command or startup item.
- k** Continue in the presence of errors in HAL files
- t <tpmodulename> [parameters]**  
Specify custom trajectory\_planning\_module overrides optional INI setting [TRAJ]TPMOD
- m <homemodulename> [parameters]**  
Specify custom homing\_module overrides optional INI setting [EMCMOT]HOMEMOD
- H <dirname>**  
Search dirname for HAL files before searching INI directory and system library: \$SHALLIB\_DIR

**<INIFILE>**

The INI file is the main piece of a LinuxCNC configuration. It is not the entire configuration; there are various other files that go with it (NML files, HAL files, TBL files, VAR files). It is, however, the most important one, because it is the file that holds the configuration together. It can adjust a lot of parameters itself, but it also tells **linuxcnc** which other files to load and use.

There are several ways to specify which config to use:

Specify the absolute path to an INI, e.g.,

**linuxcnc** /usr/local/linuxcnc/configs/sim/sim.ini

Specify a relative path from the current directory, e.g.

**linuxcnc** configs/sim/sim.ini

Otherwise, in the case where the **INIFILE** is not specified, the behavior will depend on whether you configured LinuxCNC with **--enable-run-in-place**. If so, the LinuxCNC config chooser will search only the configs directory in your source tree. If not (or if you are using a packaged version of LinuxCNC), it may search several directories. The config chooser is currently set to search the path:

~/linuxcnc/configs:/var/lib/buildbot/workers/docs-testing/20-docs/build/configs

**EXAMPLES**

**linuxcnc**

**linuxcnc** *configs/sim/sim.ini*

**linuxcnc** */etc/linuxcnc/sample-configs/stepper/stepper\_mm.ini*

**SEE ALSO**

**halcmd(1)**

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/linuxcnc/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Alex Joni, as part of the LinuxCNC Enhanced Machine Controller project.

**REPORTING BUGS**

Report bugs to alex\_joni AT users DOT sourceforge DOT net

**COPYRIGHT**

Copyright © 2006 Alex Joni.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

5axisgui – Vismach Virtual Machine GUI

**DESCRIPTION**

**5axisgui** is one of the sample **Vismach** GUIs for LinuxCNC.

**SEE ALSO**

linuxcnc(1)

See the main LinuxCNC documentation for more details at <https://linuxcnc.org/docs/html/gui/vismach.html>.

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

axis-remote – AXIS Remote Interface

**SYNOPSIS**

**axis-remote** *OPTIONS*[*FILENAME*]

**DESCRIPTION**

**axis-remote** is a small script that triggers commands in a running AXIS GUI.

Use **axis-remote --help** for further information.

**OPTIONS**

**--ping, -p**

Check whether AXIS is running.

**--reload, -r**

Make AXIS reload the currently loaded file.

**--clear, -c**

Make AXIS clear the backplot.

**--quit, -q**

Make AXIS quit.

**--help, -h, -?**

Display a list of valid parameters for **axis-remote**.

**--mdi** *COMMAND*, **-m** *COMMAND*

Run the MDI command *COMMAND*.

*FILENAME*

Load the G-code file *FILENAME*.

**SEE ALSO**

axis(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at [/usr/share/doc/linuxcnc/](http://usr/share/doc/linuxcnc/).

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Alex Joni, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2007 Alex Joni.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

axis – AXIS LinuxCNC Graphical User Interface

**SYNOPSIS**

**axis** *-ini INIFILE*

**DESCRIPTION**

**axis** is one of the Graphical User Interfaces (GUI) for LinuxCNC. It gets run by the runscrip usually.

**OPTIONS**

*INIFILE*

The INI file is the main piece of an LinuxCNC configuration. It is not the entire configuration; there are various other files that go with it (NML files, HAL files, TBL files, VAR files). It is, however, the most important one, because it is the file that holds the configuration together. It can adjust a lot of parameters itself, but it also tells **LinuxCNC** which other files to load and use.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Alex Joni, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2007 Alex Joni.

This is free software; see the source for copying conditions. There is **NO** warranty; not even for **MERCHANTABILITY** or **FITNESS FOR A PARTICULAR PURPOSE**.

**NAME**

debuglevel – sets the debug level for the non–realtime part of LinuxCNC

**SYNOPSIS**

**debuglevel** **-ini** *INIFILE*

**DESCRIPTION**

**debuglevel** displays a checkbox to select the current debug level of some parts of LinuxCNC.

**SEE ALSO**

halcmd(1) – debug subcommand linuxcnc(1)\*

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at </usr/share/doc/LinuxCNC/>.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

elbpcom – Communicate with Mesa ethernet cards

**SYNOPSIS**

Common options

```
*elbpcom* [***--ip=*_IP_] [***--port=*_PORT_] [***--timeout=*_TIMEOUT_]
[***--space=*_MEMSPACE_] [***--size=*_TRANSFER_SIZE_]
```

Reading data

```
*elbpcom* [_common options_] [***--info*] ***--address=*_ADDRESS_
***--read=*_LENGTH_
```

Writing data

```
*elbpcom* [_common options_] ***--address=*_ADDRESS_ ***--write=*_HEXDATA_
```

Read and decode memory space info area

```
*elbpcom* [***--space=*_MEMSPACE_] ***--read-info*
```

Sending arbitrary packets

```
*elbpcom* [_common options_] _HEXDATA_
```

**DESCRIPTION**

Read or write data from a Mesa ethernet card that uses the LBP16 protocol, such as the 7I80. This can be useful for performing certain low-level tasks.

For more information about the meaning of each address space, see the card documentation. Incorrect use of this utility can have negative effects such as changing the board's IP address or even corrupting the FPGA bitfile in the eeprom. For some tasks, such as updating FPGA bitfiles and setting IP addresses, **mesaflash(1)** is a more appropriate tool.

If not specified, the default values are

```
***--ip=*_192.168.1.121_ ***--port=*_27181_ ***--timeout=*_2_ ***--space=*_0_
***--size=*_0_
```

If the **--size** argument *TRANSFER\_SIZE* is 0, elbpcom will look up the preferred transfer size of the space in the space's info area.

This example demonstrates reading the HOSTMOT2 identifying string from the IDROM in space 0:

```
$ elbpcom --address 0x104 --read 8
> 82420401
< 484f53544d4f5432
  HOSTMOT2
```

First the request is shown in hex. Then the response (if any) is shown in hex. Finally, the response is shown in ASCII, with "." replacing any non-ASCII characters. This is similar to the following invocations of **mesaflash**:

```
$ ./mesaflash --device 7i80 --rpo 0x104
54534F48
$ ./mesaflash --device 7i80 --rpo 0x108
32544F4D
```

but notice its different treatment of byte order.

**SEE ALSO**

mesaflash(1), hostmot2(9), hm2\_eth(9), Mesa's documentation for the Anything I/O boards.

**NAME**

emccalib – Adjust ini tuning variables on the fly with save option

**SYNOPSIS**

**emccalib.tcl** [*options*]

**DESCRIPTION**

The Calibration assistant. This tool Reads the HAL file and for every *setp* that uses a variable from the INI file that is in an [AXIS\_L], [JOINT\_N], [SPINDLE\_S], or [TUNE] section it creates an entry that can be edited and tested.

**USAGE**

The calibration/tuning tool supports the following stanzas:

[JOINT\_N], [AXIS\_L], [SPINDLE\_S], [TUNE]

where *N* is a joint number (0 .. ([KINS]JOINTS-1) ), *L* is an axis coordinate letter (X,Y,Z,A,B,C,U,V,W), and *S* is a spindle number (0 .. 9).

**Note**

The number of allowed spindles is 8 but legacy configurations may include a stanza [SPINDLE\_9] unrelated to an actual spindle number.

**Note**

The [TUNE] stanza may be used for specifying tunable items not relevant to the other supported stanzas.

**EXIT STATUS**

Return exit code 1 if the ini file is incompatible with the current version of emccalib.

**SEE ALSO**

linuxcnc(1)

**AUTHOR**

Original version by Petter Reinholdtsen as part of the LinuxCNC project. Improvements by several other members of the LinuxCNC development team.

**COPYRIGHT**

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.



**NAME**

gladevcp – Virtual Control Panel for LinuxCNC based on Glade, Gtk and HAL widgets

**SYNOPSIS**

**gladevcp** [-g *WxH+X+Y*] [-c *component-name*] [-u *handler*] [-U *useroption*] [-H *HAL-file*] [-d]  
*myfile.ui*

**OPTIONS**

**-g** *WxH+X+Y*

This sets the initial geometry of the root window. Use *WxH* for just size, *+X+Y* for just position, or *WxH+X+Y* for both. Size / position use pixel units. Position is referenced from top left.

**-c** *component-name*

Use *component-name* as the HAL component name. If the component name is not specified, the basename of the UI file is used.

**-u** *handler*

Instructs GladeVCP to inspect the Python script *handler* for event handlers, and connect them to signals in the UI file.

**-U** *useroption*

GladeVCP collects all *useroption* strings and passes them to the handler *init()* method as a list of strings without further inspection.

**-x** *XID*

Reparent GladeVCP into an existing window *XID* instead of creating a new top level window.

**-H** *halfile*

GladeVCP runs *HAL file* – a list of HAL commands – by executing *halcmd -c filename* after the HAL component is finalized.

**-d**

enable debug output.

**-R** *gtkrcfile*

explicitly load a gtkrc file.

**-t** *THEME*

set gtk theme. Default is the *system* theme. Different panels can have different themes.

**-m** *MAXIMUM*

force panel window to maximize. Together with the *-g geometry* option one can move the panel to a second monitor and force it to use all of the screen

**-R**

explicitly deactivate workaround for a GTK bug which makes matches of widget and widget\_class matches in GTK theme and gtkrc files fail. Normally not needed.

**SEE ALSO**

*GladeVCP* in the LinuxCNC documentation for a description of GladeVCP's capabilities and the associated HAL widget set, along with examples.

**NAME**

gladevcp\_demo – used by sample configs to deonstrate Glade Virtual\_demo

**SYNOPSIS**

**gladevcp\_demo** Control Panels

**DESCRIPTION**

**gladevcp\_demo** is a sample GladeVCP handler

**SEE ALSO**

linuxcnc(1), <https://linuxcnc.org/docs/html/gui/gladevcp.html>

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

gmoccapy – TOUCHY LinuxCNC Graphical User Interface

**SYNOPSIS**

**gmoccapy** *-ini* <INI file>

**DESCRIPTION**

**GMOCCAPY** is one of the Graphical User Interfaces (GUI) for LinuxCNC. It gets run by the runscrip usually.

**OPTIONS****INI file**

The *INI file* is the main piece of an LinuxCNC configuration. It is not the entire configuration; there are various other files that go with it (NML files, HAL files, TBL files, VAR files). It is, however, the most important one, because it is the file that holds the configuration together. It can adjust a lot of parameters itself, but it also tells **LinuxCNC** which other files to load and use.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at /usr/share/doc/LinuxCNC/.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

gremlin\_view – G-code graphical preview

**SYNOPSIS**

**gremlin\_view**

**DESCRIPTION**

**gremlin\_view** is a Python wrapper for the gremlin G-code graphical preview.

PGremlinView for gremlin with buttons for simpler embedding Standalone functionality if linuxcnc running A default ui file (gremlin\_view.ui) is provided for a default button arrangement but a user may provide their own by supplying the glade\_file argument. The following objects are mandatory:  
 gremlin\_view\_window:: toplevel window gremlin\_view\_hal\_gremlin:: hal\_gremlin gremlin\_view\_box::  
 HBox or VBox containing hal\_gremlin

Optional radiobutton group names

*select\_p\_view select\_x\_view select\_y\_view select\_z\_view select\_z2\_view*

Optional checkbuttons names

*enable\_dro show\_machine\_speed 'show\_distance\_to\_go show\_limits show\_extents show\_tool  
 show\_metric*

Callbacks are provided for the following button actions

*on\_clear\_live\_plotter\_clicked on\_enable\_dro\_clicked on\_zoomin\_pressed on\_zoomout\_pressed  
 on\_pan\_x\_minus\_pressed on\_pan\_x\_plus\_pressed on\_pan\_y\_minus\_pressed on\_pan\_y\_plus\_pressed  
 on\_show\_tool\_clicked on\_show\_metric\_clicked on\_show\_extents\_clicked on\_select\_p\_view\_clicked  
 on\_select\_x\_view\_clicked on\_select\_y\_view\_clicked on\_select\_z\_view\_clicked  
 on\_select\_z2\_view\_clicked on\_show\_distance\_to\_go\_clicked on\_show\_machine\_speed\_clicked  
 on\_show\_limits\_clicked*

**SEE ALSO**

linuxcnc(1), <https://wiki.linuxcnc.org/cgi-bin/wiki.pl?Gremlin>

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at /usr/share/doc/LinuxCNC/.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

gs2\_vfd – HAL non–realtime component for Automation Direct GS2 VFDs

**SYNOPSIS**

**gs2\_vfd** [OPTIONS]

**DESCRIPTION**

This manual page explains the **gs2\_vfd** component. This component reads and writes to the GS2 via a modbus connection.

**gs2\_vfd** is for use with LinuxCNC.

**OPTIONS**

- b, --bits *n*  
Set number of data bits to *n*, where *n* must be from 5 to 8 inclusive (default 8).
- d, --device *path*  
Set the path to the file representing the serial device to use (default /dev/ttyS0).
- v, --verbose  
Turn on verbose mode.
- g, --debug  
Turn on debug messages. Note that if there are serial errors, this may become annoying. Debug mode will cause all modbus messages to be printed in hex on the terminal.
- n, --name *string*  
Set the name of the HAL module. The HAL comp name will be set to *string*, and all pin and parameter names will begin with *string* (default gs2\_vfd).
- p, --parity [even,odd,none]  
Set serial parity to even, odd, or none (default odd).
- r, --rate *n*  
Set baud rate to *n*. It is an error if the rate is not one of the following: 110, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200 (default 38400).
- s, --stopbits [1,2]  
Set serial stop bits to 1 or 2 (default 1).
- t, --target <*n*>  
Set MODBUS target (slave) number. This must match the device number you set on the GS2 (default 1).
- A, --accel–seconds <*n*>  
Seconds to accelerate the spindle from 0 to Max RPM (default 10.0).
- D, --decel–seconds <*n*>  
Seconds to decelerate the spindle from Max RPM to 0. If set to 0.0 the spindle will be allowed to coast to a stop without controlled deceleration (default 0.0).
- R, --braking–resistor  
This argument should be used when a braking resistor is installed on the GS2 VFD (see Appendix A of the GS2 manual). It disables deceleration over–voltage stall prevention (see GS2 modbus Parameter 6.05), allowing the VFD to keep braking even in situations where the motor is regenerating high voltage. The regenerated voltage gets safely dumped into the braking resistor.

**PINS**

- <name>.DC–bus–volts (float, out)  
from the VFD
- <name>.at–speed (bit, out)  
when drive is at commanded speed
- <name>.err–reset (bit, in)

reset errors sent to VFD

<name>.firmware-revision (s32, out)  
from the VFD

<name>.frequency-command (float, out)  
from the VFD

<name>.frequency-out (float, out)  
from the VFD

<name>.is-stopped (bit, out)  
when the VFD reports 0 Hz output

<name>.load-percentage (float, out)  
from the VFD

<name>.motor-RPM (float, out)  
from the VFD

<name>.output-current (float, out)  
from the VFD

<name>.output-voltage (float, out)  
from the VFD

<name>.power-factor (float, out)  
from the VFD

<name>.scale-frequency (float, out)  
from the VFD

<name>.speed-command (float, in)  
speed sent to VFD in RPM. It is an error to send a speed faster than the Motor Max RPM as set in the VFD.

<name>.spindle-fwd (bit, in)  
1 for FWD and 0 for REV sent to VFD

<name>.spindle-on (bit, in)  
1 for ON and 0 for OFF sent to VFD, only on when running

<name>.spindle-rev (bit, in)  
1 for ON and 0 for OFF, only on when running

<name>.status-1 (s32, out)  
Drive Status of the VFD (see the GS2 manual)

<name>.status-2 (s32, out)  
Drive Status of the VFD (see the GS2 manual) Note that the value is a sum of all the bits that are on. So a 163 which means the drive is in the run mode is the sum of 3 (run) + 32 (freq set by serial) + 128 (operation set by serial).

## PARAMETERS

<name>.error-count (s32, RW), <name>.loop-time (float, RW)  
how often the modbus is polled (default 0.1)

<name>.nameplate-HZ (float, RW)  
Nameplate Hz of motor (default 60)

<name>.nameplate-RPM (float, RW)  
Nameplate RPM of motor (default 1730)

<name>.retval (s32, RW)  
the return value of an error in HAL

<name>.tolerance (float, RW)

speed tolerance (default 0.01)

<name>.ack-delay (s32, RW)

number of read/write cycles before checking at-speed (default 2)

**SEE ALSO**

*GS2 Driver* in the LinuxCNC documentation for a full description of the **GS2** syntax

*GS2 Examples* in the LinuxCNC documentation for examples using the **GS2** component

**AUTHOR**

John Thornton

**LICENSE**

GPL

**NAME**

gscreen – TOUCHY LinuxCNC Graphical User Interface

**SYNOPSIS**

**gscreen** **-ini** *INIFILE*

**DESCRIPTION**

**gscreen** is one of the Graphical User Interfaces (GUI) for LinuxCNC. It gets run by the runscrip usually.

**OPTIONS**

*INIFILE*

The INI file is the main piece of an LinuxCNC configuration. It is not the entire configuration; there are various other files that go with it (NML files, HAL files, TBL files, VAR files). It is, however, the most important one, because it is the file that holds the configuration together. It can adjust a lot of parameters itself, but it also tells **LinuxCNC** which other files to load and use.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.



**NAME**

hal-histogram – plots the value of a HAL pin as a histogram

**SYNOPSIS**

**hal-histogram**

**DESCRIPTION**

**hal-histogram** represents the values of a HAL pin graphically.

Details: [https://linuxcnc.org/docs/html/hal/tools.html#\\_hal\\_histogram](https://linuxcnc.org/docs/html/hal/tools.html#_hal_histogram)

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at [/usr/share/doc/LinuxCNC/](#).

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

hal\_input – control HAL pins with any Linux input device, including USB HID devices

**SYNOPSIS**

```
loadusr*_ *hal_input [-KRAL] inputspec ...
```

**DESCRIPTION**

**hal\_input** is an interface between HAL and any Linux input device, including USB HID devices. For each device named, **hal\_input** creates pins corresponding to its keys, absolute axes, and LEDs. At a fixed rate of approximately 10 ms, it synchronizes the device and the HAL pins.

**INPUT SPECIFICATION**

The *inputspec* may be in one of several forms:

A string *S*

A substring or shell-style pattern match will be tested against the "name" of the device, the "phys" (which gives information about how it is connected), and the "id", which is a string of the form "Bus=... Vendor=... Product=... Version=...". You can view the name, phys, and id of attached devices by executing `less /proc/bus/input/devices`.

Examples:

```
SpaceBall
```

```
"Vendor=001f Product=0001"
```

```
serio*/input0
```

A number *N*

This opens `*/dev/input/event*N`. Except for devices that are always attached to the system, this number may change over reboots or when the device is removed. For this reason, using an integer is not recommended.

When several devices are identified by the same string, add `:*N` where *N* is the index of the desired device. For example, if `*Mouse` matches `input3` and `input10`, then `Mouse` and `Mouse:0` select `input3`. Specifying `mouse:1` selects `input10`.

For devices that appear as multiple entries in `/dev/input`, these indices are likely to stay the same every time. For multiple identical devices, these indices are likely to depend on the insertion order, but stay the same across reboots as long as the devices are not moved to different ports or unplugged while the machine is booted.

If the first character of the *inputspec* is a "+", then **hal\_input** requests exclusive access to the device. The first device matching an *inputspec* is used. Any number of *inputspecs* may be used.

A *subset option* may precede each *inputspec*. The subset option begins with a dash. Each letter in the subset option specifies a device feature to **include**. Features that are not specified are excluded. For instance, to export keyboard LEDs to HAL without exporting keys, use

```
hal_input -L keyboard ...
```

**DEVICE FEATURES SUPPORTED**

- EV\_KEY (buttons and keys). Subset -K
- EV\_ABS (absolute analog inputs). Subset -A

- EV\_REL (relative analog inputs). Subset -R
- EV\_LED (LED outputs). Subset -L

## HAL PINS AND PARAMETERS

### For buttons

**input.N.btn-name** bit out

**input.N.btn-name-not** bit out

Created for each button on the device.

### For keys

**input.N.key-name**, **input.N.key-name-not**

Created for each key on the device.

### For absolute axes

**input.N.abs-name-counts** s32 out, **input.N.abs-name-position** float out, **input.N.abs-name-scale** parameter float rw, **input.N.abs-name-offset** parameter float rw, **input.N.abs-name-fuzz** parameter s32 rw, **input.N.abs-name-flat** parameter s32 rw, **input.N.abs-name-min** parameter s32 r, **input.N.abs-name-max** parameter s32 r

Created for each absolute axis on the device. Device positions closer than **flat** to **offset** are reported as **offset** in **counts**, and **counts** does not change until the device position changes by at least **fuzz**. The position is computed as **position** = (**counts** - **offset**) / **scale**. The default value of **scale** and **offset** map the range of the axis reported by the operating system to [-1,1]. The default values of **fuzz** and **flat** are those reported by the operating system. The values of **min** and **max** are those reported by the operating system.

### For relative axes

**input.N.rel-name-counts** s32 out, **input.N.rel-name-position** float out, **input.N.rel-name-reset** bit in, **input.N.rel-name-scale** parameter float rw, **input.N.rel-name-absolute** parameter s32 rw, **input.N.rel-name-precision** parameter s32 rw, **input.N.rel-name-last** parameter s32 rw

Created for each relative axis on the device. As long as **reset** is true, **counts** is reset to zero regardless of any past or current axis movement. Otherwise, **counts** increases or decreases according to the motion of the axis. **counts** is divided by **position-scale** to give **position**. The default value of **position** is 1. There are some devices, notably scroll wheels, which return signed values with less resolution than 32 bits. The default value of **precision** is 32. **precision** can be set to 8 for a device that returns signed 8 bit values, or any other value from 1 to 32. **absolute**, when set true, ignores duplicate events with the same value. This allows for devices that repeat events without any user action to work correctly. **last** shows the most recent count value returned by the device, and is used in the implementation of **absolute**.

### For LEDs

**input.N.led-name** bit out, **input.N.led-name-invert** parameter bit rw

Created for each LED on the device.

## PERMISSIONS AND UDEV

By default, the input devices may not be accessible to regular users — **hal\_input** requires read-write access, even if the device has no outputs.

Different versions of udev have slightly different, incompatible syntaxes. For this reason, it is not possible for this manual page to give an accurate example. The **udev(7)** manual page documents the syntax used on your Linux distribution. To view it in a terminal, the command is `man 7 udev`.

## BUGS

The initial state of keys, buttons, and absolute axes are erroneously reported as FALSE or 0 until an event is received for that key, button, or axis.

## SEE ALSO

`udev(8)`, `udev(7)`

**NAME**

hal\_manualtoolchange – HAL non–realtime component to enable manual tool changes.

**SYNOPSIS**

```
loadusr hal_manualtoolchange
```

**DESCRIPTION**

hal\_manualtoolchange is a LinuxCNC non–realtime component that allows users with machines lacking automatic tool changers to make manual tool changes. In use when a M6 tool change is encountered, the motion component will stop the spindle and pause the program. The hal\_manualtoolchange component will then receive a signal from the motion component causing it to display a tool change window prompting the user which tool number to load based on the last T– number programmed. The dialog will stay active until the "continue" button is pressed. When the "continue" button is pressed, hal\_manualtoolchange will then signal the motion component that the tool change is complete thus allowing motion to turn the spindle back on and resume program execution.

Additionally, the hal\_manualtoolchange component includes a hal pin for a button that can be connected to a physical button to complete the tool change and remove the window prompt (hal\_manualtoolchange.change\_button).

hal\_manualtoolchange can be used even when AXIS is not used as the GUI. This component is most useful if you have presettable tools and you use the tool table.

**PINS**

**hal\_manualtoolchange.number** s32 in  
Receives last programmed T– number.

**hal\_manualtoolchange.change** bit in  
Receives signal to do tool change.

**hal\_manualtoolchange.changed** bit out  
Signifies that the tool change is complete.

**hal\_manualtoolchange.change\_button** bit in  
Pin to allow an external switch to signify that the tool change is complete.

**USAGE**

Normal usage is to load the component in your HAL file and net the appropriate pins from the *motion* and *io* components. The following lines are typical in a HAL file when using the hal\_manualtoolchange non–realtime component.

```
loadusr -W hal_manualtoolchange
```

This will load the hal\_manualtoolchange non–realtime component waiting for the component to be ready before continuing.

```
net tool-change ioccontrol.0.tool-change â hal_manualtoolchange.change
```

When an M6 code is run, motion sets ioccontrol.0.tool-change to high indicating a tool change. This pin should be netted to hal\_manualtoolchange.change. This causes the Tool change dialog to be displayed on screen and wait for the user to either click the continue button on the dialog or press an externally connected button.

```
net tool-changed ioccontrol.0.tool-changed â hal_manualtoolchange.changed
```

When the Tool change dialog's continue button is pressed, it will set the hal\_manualtoolchange.changed pin to high, this should be netted to the ioccontrol.0.tool-changed pin, indicating to the motion controller that the tool change has been completed and can continue with the execution of the G–code program.

```
net tool-number ioccontrol.0.tool-prep-number â hal_manualtoolchange.number
```

When a T– command is executed in a G–code program, the tool number will held in the ioccontrol.0.tool-prep-number. This pin should be netted to hal\_manualtoolchange.number. The value of this pin, the tool number is displayed in the tool change dialog to let the user know which tool

should be loaded.

```
net tool-prepare-loopback iocontrol.0.tool-prepare & iocontrol.0.tool-prepared
```

The `iocontrol.0.tool-prepare` pin will go true when a `Tn` tool prepare is requested. Since there is not automated tool changer this pin should be netted to `iocontrol.0.tool-prepared` to indicate that the tool has been prepared.

If you wish to use an external button to signal the `hal_manualtoolchange` component that the tool change is complete simply bring the button into HAL (via a parport input pin or a hostmot2 gpio input or similar), and wire it directly to the `hal_manualtoolchange.change_button` pin. For Example:

```
net tool-changed-btn hal_manualtoolchange.change_button <= parport.0.pin-15-in
```

#### **SEE ALSO**

`motion(1)`, `iocontrol(1)`, `halcmd(1)`.

**NAME**

halcmd – manipulate the LinuxCNC HAL from the command line

**SYNOPSIS**

**halcmd** [*OPTIONS*] [*COMMAND* [*ARG*]]

**DESCRIPTION**

The tool **halcmd** is used to manipulate the HAL (Hardware Abstraction Layer) from the command line.

**halcmd** can optionally read commands from a file, allowing complex HAL configurations to be set up with a single command.

If the **readline** library is available when LinuxCNC is compiled, then **halcmd** offers commandline editing and completion when running interactively. Use the up arrow to recall previous commands, and press tab to complete the names of items such as pins and signals.

**OPTIONS**

**-I**

Before tearing down the realtime environment, run an interactive halcmd. **halrun** only. If **-I** is used, it must precede all other commandline arguments.

**-f** [*<file>*]

Ignore commands on command line, take input from *file* instead. If *file* is not specified, take input from *stdin*.

**-i** *<INI file>*

Use variables from the specified *INI file* for substitutions. See **SUBSTITUTION** below.

**-k**

Keep going after failed command(s). The default is to stop and return failure if any command fails.

**-q**

display errors only (default)

**-Q**

display nothing, execute commands silently

**-s**

Script-friendly mode. In this mode, *show* will not output titles for the items shown. Also, module names will be printed instead of ID codes in pin, param, and funct listings. Threads are printed on a single line, with the thread period, FP usage and name first, followed by all of the functions in the thread, in execution order. Signals are printed on a single line, with the type, value, and signal name first, followed by a list of pins connected to the signal, showing both the direction and the pin name.

**-R**

Release the HAL mutex. This is useful for recovering when a HAL component has crashed while holding the HAL mutex.

**-v**

display results of each command

**-V**

display lots of debugging junk

**-h** [*command*]

display a help screen and exit, displays extended help on *command* if specified

**COMMANDS**

Commands tell **halcmd** what to do. Normally **halcmd** reads a single command from the command line and executes it. If the **-f** option is used to read commands from a file, **halcmd** reads each line of the file as a new command. Anything following **#** on a line is a comment.

**loadrt** *modname*

(short for *load realtime module*) Loads a realtime HAL module called *modname*. **halcmd** looks for

the module in a directory specified at compile time.

In systems with kernel-based realtime support (e.g. RTAI), **halcmd** calls the **linuxcnc\_module\_helper** to load realtime modules. **linuxcnc\_module\_helper** is a setuid program and is compiled with a whitelist of modules it is allowed to load. This is currently just a list of **LinuxCNC**-related modules. The **linuxcnc\_module\_helper** execs `insmod`, so return codes and error messages are those from `insmod`. Administrators who wish to restrict which users can load these **LinuxCNC**-related kernel modules can do this by setting the permissions and group on **linuxcnc\_module\_helper** appropriately.

In systems with userspace-based realtime support (e.g. Preempt-RT) and in systems without realtime support **halcmd** calls the **rtapi\_app** which creates the realtime environment (simulated realtime, on systems with no userspace realtime support) if it did not yet exist, and then loads the requested component with a call to **dlopen(3)**.

**unloadrt** *modname*

(*unload* realtime module) Unloads a realtime HAL module called *modname*. If *modname* is "all", it will unload all currently loaded realtime HAL modules. **unloadrt** also works by execing **linuxcnc\_module\_helper** or **rtapi\_app**, just like **loadrt**.

**loadusr** [*flags*] *UNIX-command*

(*load* Userspace component) Executes the given *UNIX-command*, usually to load a non-realtime component. [*flags*] may be one or more of:

- **-W** to wait for the component to become ready. The component is assumed to have the same name as the first argument of the command.
- **-Wn name** to wait for the component, which will have the given name.
- **-w** to wait for the program to exit
- **-i** to ignore the program return value (with **-w**)

**waitusr** *name*

(*wait* for Userspace component) Waits for non-realtime component *name* to disconnect from HAL (usually on exit). The component must already be loaded. Useful near the end of a HAL file to wait until the user closes some user interface component before cleaning up and exiting.

**unloadusr** *compname*

(*unload* Userspace component) Unloads a non-realtime component called *compname*. If *compname* is "all", it will unload all non-realtime components. **unloadusr** works by sending SIGTERM to all non-realtime components.

**unload** *compname*

Unloads a non-realtime component or realtime module. If *compname* is "all", it will unload all non-realtime components and realtime modules.

**newsig** *signame type*

(OBSOLETE – use **net** instead) (*new* signal) Creates a new HAL signal called *signame* that may later be used to connect two or more HAL component pins. *type* is the data type of the new signal, and must be one of "bit", "s32", "u32", or "float". Fails if a signal of the same name already exists.

**delsig** *signame*

(*delete* signal) Deletes HAL signal *signame*. Any pins currently linked to the signal will be unlinked. Fails if *signame* does not exist.

**sets** *signame value*

(*set* signal) Sets the value of signal *signame* to *value*. Fails if *signame* does not exist, if it already has a writer, or if *value* is not a legal value. Legal values depend on the signals's type.

**stype** *name*

(*signal* type) Gets the type of signal *name*. Fails if *name* does not exist as a signal.

**gets** *signame*

(*get signal*) Gets the value of signal *signame*. Fails if *signame* does not exist.

**linkps** *pinname* [*arrow*] *signame*

(OBSOLETE – use **net** instead) (*link pin to signal*) Establishes a link between a HAL component pin *pinname* and a HAL signal *signame*. Any previous link to *pinname* will be broken. *arrow* can be "**â**", "**â**", "<**â**", or omitted. **halcmd** ignores arrows, but they can be useful in command files to document the direction of data flow. Arrows should not be used on the command line since the shell might try to interpret them. Fails if either *pinname* or *signame* does not exist, or if they are not the same type type.

**linksp** *signame* [*arrow*] *pinname*

(OBSOLETE – use **net** instead) (*link signal to pin*) Works like **linkps** but reverses the order of the arguments. **halcmd** treats both link commands exactly the same. Use whichever you prefer.

**linkpp** *pinname1* [*arrow*] *pinname2*

(OBSOLETE – use **net** instead) (*link pin to pin*) Shortcut for **linkps** that creates the signal (named like the first pin), then links them both to that signal. **halcmd** treats this just as if it were: **halcmd newsig pinname1 halcmd linksp pinname1 pinname1 halcmd linksp pinname1 pinname2**

**net** *signame pinname ...*

Create *signame* to match the type of *pinname* if it does not yet exist. Then, link *signame* to each *pinname* in turn. Arrows may be used as in **linkps**. When linking a pin to a signal for the first time, the signal value will inherit the pin's default value.

**unlinkp** *pinname*

(*unlink pin*) Breaks any previous link to *pinname*. Fails if *pinname* does not exist. An unlinked pin will retain the last value of the signal it was linked to.

**setp** *name value*

(*set parameter or pin*) Sets the value of parameter or pin *name* to *value*. Fails if *name* does not exist as a pin or parameter, if it is a parameter that is not writable, if it is a pin that is an output, if it is a pin that is already attached to a signal, or if *value* is not a legal value. Legal values depend on the type of the pin or parameter. If a pin and a parameter both exist with the given name, the parameter is acted on.

*paramname = value, pinname = value*

Identical to **setp**. This alternate form of the command may be more convenient and readable when used in a file.

**pptype** *name*

(*parameter or pin type*) Gets the type of parameter or pin *name*. Fails if *name* does not exist as a pin or parameter. If a pin and a parameter both exist with the given name, the parameter is acted on.

**getp** *name*

(*get parameter or pin*) Gets the value of parameter or pin *name*. Fails if *name* does not exist as a pin or parameter. If a pin and a parameter both exist with the given name, the parameter is acted on.

**addf** *funcname threadname*

(*add function*) Adds function *funcname* to realtime thread *threadname*. *funcname* will run after any functions that were previously added to the thread. Fails if either *funcname* or *threadname* does not exist, or if they are incompatible.

**delf** *funcname threadname*

(*delete function*) Removes function *funcname* from realtime thread *threadname*. Fails if either *funcname* or *threadname* does not exist, or if *funcname* is not currently part of *threadname*.

**start**

Starts execution of realtime threads. Each thread periodically calls all of the functions that were added to it with the **addf** command, in the order in which they were added.

**stop**

Stops execution of realtime threads. The threads will no longer call their functions.



**show** [*item*]

Prints HAL items to *stdout* in human readable format. *item* can be one of "**comp**" (components), "**pin**", "**sig**" (signals), "**param**" (parameters), "**funct**" (functions), "**thread**", or "**alias**". The type "**all**" can be used to show matching items of all the preceding types. If *item* is omitted, **show** will print everything.

**save** [*item*]

Prints HAL items to *stdout* in the form of HAL commands. These commands can be redirected to a file and later executed using **halcmd -f** to restore the saved configuration. *item* can be one of the following:

"**comp**" generates a **loadrt** command for realtime component.

"**alias**" generates an **alias** command for each pin or parameter alias pairing

"**sig**" (or "**signal**") generates a **newsig** command for each signal, and "**sigu**" generates a **newsig** command for each unlinked signal (for use with **netl** and **netla**).

"**link**" and "**linka**" both generate **linkps** commands for each link. (**linka** includes arrows, while **link** does not.)

"**net**" and "**neta**" both generate one **newsig** command for each signal, followed by **linksp** commands for each pin linked to that signal. (**neta** includes arrows.)

"**netl**" generates one **net** command for each linked signal, and "**netla**" (or "**netal**") generates a similar command using arrows.

"**param**" (or "**parameter**") generates one **setp** command for each parameter.

"**thread**" generates one **addf** command for each function in each realtime thread.

"**unconnectedinpins**" generates a **setp** command for each unconnected HAL input pin.

If *item* is **allu**), **save** does the equivalent of **comp**, **alias**, **sigu**, **netla**, **param**, **thread**, and **unconnectedinpins**.

If *item* is omitted (or **all**), **save** does the equivalent of **comp**, **alias**, **sigu**, **netla**, **param**, and **thread**.

**source** *filename.hal*

Executes the commands from *filename.hal*.

**alias** *type name alias*

Assigns "**alias**" as a second name for the pin or parameter "name". For most operations, an alias provides a second name that can be used to refer to a pin or parameter, both the original name and the alias will work. "type" must be **pin** or **param**. "name" must be an existing name or **alias** of the specified type. Note that the "show" command will only show the aliased name, but the original name is still valid to use in HAL. The original names can still be seen with "show all" or "show alias". Existing nets will be preserved when a pin name is aliased.

**unalias** *type alias*

Removes any alias from the pin or parameter alias. "type" must be **pin** or **param** "alias" must be an existing name or **alias** of the specified type.

**list** *type [pattern]*

Prints the names of HAL items of the specified type. *type* is **comp**, **pin**, **sig**, **param**, **funct**, or **thread**. If *pattern* is specified it prints only those names that match the pattern, which may be a *shell glob*. For **sig**, **pin** and **param**, the first pattern may be **-datatype** where datatype is the data type (e.g., *float*) in

this case, the listed pins, signals, or parameters are restricted to the given data type Names are printed on a single line, space separated.

**print** [*message*]

Prints the filename, linenummer and an optional message. wrap the message in quotes if it has spaces.

**lock** [*all*|*tune*|*none*]

Locks HAL to some degree. none – no locking done. tune – some tuning is possible (**setp** & such). all – HAL completely locked.

**unlock** [*all*|*tune*]

Unlocks HAL to some degree. tune – some tuning is possible (**setp** & such). all – HAL completely unlocked.

**status** [*type*]

Prints status info about HAL. *type* is **lock**, **mem**, or **all**. If *type* is omitted, it assumes **all**.

**debug** [*level*]

Sets the rtapi messaging level (see man3 rtapi\_set\_msg\_level).

**help** [*command*]

Give help information for command. If *command* is omitted, list command and brief description.

**SUBSTITUTION**

After a command is read but before it is executed, several types of variable substitution take place.

**Environment Variables**

Environment variables have the following formats:

**\$ENVVAR** followed by end-of-line or whitespace

**\$(ENVVAR)**

**INI file variables**

INI file variables are available only when an INI file was specified with the halcmd **-i** flag. They have the following formats:

**[SECTION]VAR** followed by end-of-line or whitespace

**[SECTION](VAR)**

**LINE CONTINUATION**

The backslash character (\) may be used to indicate the line is extended to the next line. The backslash character must be the last character before the newline.

**BUGS**

None known at this time.

**AUTHOR**

Original version by John Kasunich, as part of the LinuxCNC project. Now includes major contributions by several members of the project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2003 John Kasunich.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**SEE ALSO**

halrun(1) — a convenience script to start a realtime environment, process a HAL or a .tcl file, and optionally start an interactive command session using **halcmd** (described here) or haltcl(1).

**NAME**

halcmd\_twopass – short description

**SYNOPSIS**

{name}

**DESCRIPTION**

**halcmd\_twopass** is a utility script used when parsing HAL files. It is of little relevance to normal users and is used internally by the system.

**SEE ALSO**

linuxcnc(1), <https://linuxcnc.org/docs/html/hal/twopass.html>

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

halcompile – Build, compile and install LinuxCNC HAL components

**SYNOPSIS**

**halcompile** [**--compile**|**--preprocess**|**--document**|**--view-doc**] compfile...

*sudo* **halcompile** [**--install**|**--install-doc**] compfile...

**halcompile** **--compile** **--userspace** cfile...

*sudo* **halcompile** **--install** **--userspace** cfile...

*sudo* **halcompile** **--install** **--userspace** pyfile...

When personalities are used in a comp file, HAL instances are exported sequentially (typically by the mutually exclusive `count=` or `names=` parameters). If the number of exports exceeds the maximum number of personalities, subsequent personalities are assigned modulo the maximum number of personalities allowed.

By default, the maximum number of personalities is 64. To alter this limit, use the **--personalities=** option with **halcompile**. For example, to set the maximum of personality items to 4: `[sudo] halcompile --personalities=4 --install ...`

Do not use `[sudo]` for RIP installation.

**DESCRIPTION**

**halcompile** performs many different functions:

- Compile **.comp** and **.c** files into **.so** or **.ko** HAL realtime components (the **--compile** flag)
- Compile **.comp** and **.c** files into HAL non-realtime components (the **--compile --userspace** flag)
- Preprocess **.comp** files into **.c** files (the **--preprocess** flag)
- Extract documentation from **.comp** files into **.9** manpage files (the **--document** flag)
- Display documentation from **.comp** files onscreen (the **--view-doc** flag)
- Compile and install **.comp** and **.c** files into the proper directory for HAL realtime components (the **--install** flag), which may require *sudo* to write to system directories.
- Install **.c** and **.py** files into the proper directory for HAL non-realtime components (the **--install --userspace** flag), which may require *sudo* to write to system directories.
- Extract documentation from **.comp** files into **.9** manpage files in the proper system directory (the **--install** flag), which may require *sudo* to write to system directories.
- Preprocess **.comp** files into **.c** files (the **--preprocess** flag)

**SEE ALSO**

*Halcompile HAL Component Generator* in the LinuxCNC documentation for a full description of the **.comp** syntax, along with examples

**pydoc**, **HAL** and *Creating Non-realtime Python Components* in the LinuxCNC documentation for documentation on the Python interface to HAL components

**NAME**

halmeter – observe HAL pins, signals, and parameters

**SYNOPSIS**

**halmeter** [-s] [**pin**|**sig**|**param** *name*] [-g *X-position Y-position [Width]*]

**DESCRIPTION**

**halmeter** is used to observe HAL (Hardware Abstraction Layer) pins, signals, or parameters. It serves the same purpose as a multimeter does when working on physical systems.

**OPTIONS****pin** *name*

Display the HAL pin *name*.

**sig** *name*

Display the HAL signal *name*.

**param** *name*

Display the HAL parameter *name*.

If neither **pin**, **sig**, or **param** are specified, the window starts out blank and the user must select an item to observe:

**-s**

Small window. Non-interactive, must be used with **pin**, **sig**, or **param** to select the item to display. The item name is displayed in the title bar instead of the window, and there are no "Select" or "Exit" buttons. Handy when you want a lot of meters in a small space.

**-g**

Geometry position. Allows one to specify the initial starting position and optionally the width of the meter. Referenced from top left of screen in pixel units. Handy when you want to load a lot of meters in a script with out them displaying on top of each other.

**USAGE**

Unless **-s** is specified, there are two buttons, "Select" and "Exit". "Select" opens a dialog box to select the item (pin, signal, or parameter) to be observed. "Exit" does what you expect.

The selection dialog has "OK", "Apply", and "Cancel" buttons. OK displays the selected item and closes the dialog. "Apply" displays the selected item but keeps the selection dialog open. "Cancel" closes the dialog without changing the displayed item.

**EXAMPLE****halmeter**

Opens a meter window, with nothing initially displayed. Use the "Select" button to choose an item to observe. Does not return until the window is closed.

**halmeter &**

Open a meter window, with nothing initially displayed. Use the "Select" button to choose an item. Runs in the background leaving the shell free for other commands.

**halmeter pin** *parport.0.pin-03-out* **&**

Open a meter window, initially displaying HAL pin *parport.0.pin-03-out*. The "Select" button can be used to display other items. Runs in background.

**halmeter -s pin** *parport.0.pin-03-out* **&**

Open a small meter window, displaying HAL pin *parport.0.pin-03-out*. The displayed item cannot be changed. Runs in background.

**halmeter -s pin** *parport.0.pin-03-out* **-g** 100 500 **&**

Open a small meter window, displaying HAL pin *parport.0.pin-03-out*. places it 100 pixels to the left and 500 pixels down from top of screen. The displayed item cannot be changed. Runs in background.

**halmeter -s pin** *parport.0.pin-03-out* **-g** 100 500 400 **&**

Open a small meter window, displaying HAL pin *parport.0.pin-03-out*. places it 100 pixels to the left and 500 pixels down from top of screen. The width will be 400 pixels (270 is default) The displayed item cannot be changed. Runs in background.

**AUTHOR**

Original version by John Kasunich, as part of the LinuxCNC project. Improvements by several other members of the LinuxCNC development team.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2003 John Kasunich.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

halreport – creates a report on the status of the HAL

**SYNOPSIS**

**halreport** [*outfilename*]

**DESCRIPTION****halreport**

1. supports components made by halcompile and numerous legacy components
2. Known unhandled components:

at\_pid

naming conflicts with pid, seldom used

boss\_plc

no manpage or docs (any users?)

watchdog

seldom used (no users in-tree)

3. deprecated/obsolete components counter supply

Identificaion of functions used according to pin name. Default handling works for components that: . use names=*count*= (.comp components created with halcompile) . have a **single** function

Full docs: [https://linuxcnc.org/docs/html/hal/tools.html#\\_halreport](https://linuxcnc.org/docs/html/hal/tools.html#_halreport)

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at </usr/share/doc/LinuxCNC/>.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.



**NAME**

halrmt – remote–control interface for LinuxCNC

**SYNOPSIS**

**halrmt** [**--port** <port number>] [**--name** <server name>] [**--connectpw** <password>] [**--enablepw** <password>] [**--sessions** <max sessions>] [**-ini** <INI file>]

**OPTIONS**

**--port** *port*

Waits for socket connections (telnet) on specified socket, without port specification it uses default port 5006. (note: linuxcncrsh uses 5007 as default)

**--name** *server\_name*

Sets the server name to specified name for Hello.

**--connectpw** *password*

Sets the connection password to *password*. Default: EMC.

**--enablepw** *password*

Sets the enable password to *password*. Default EMCTOO.

**--sessions** <max sessions>

Sets the maximum number of simultaneous connexctions to max sessions. Default is no limit (-1).

**-ini** <INI file>

Uses the specified *INI file* instead of the default emc.ini.

**DESCRIPTION**

**halrmt** supports six commands of which the commands *set* and *get* contain HAL specific sub–commands based on the commands supported by *halcmd*. Commands and most parameters are not case sensitive. The exceptions are passwords, file paths and text strings. The supported commands are as follows: **HELLO** Hello <password> <client> <version>

If a valid password was entered the server will respond with HELLO ACK *Server Name* > <*Server Version*> where server name and server version are looked up from the implementation. If an invalid password or any other syntax error occurs then the server responds with: HELLO NAK

**Get**

The get command includes one of the HAL sub–commands, described below and zero or more additional parameters.

**Set**

The set command includes one of the HAL sub–commands, described below and one or more additional parameters.

**Quit**

The quit command disconnects the associated socket connection.

**Shutdown**

The shutdown command tells LinuxCNC to shut down before quitting the connection. This command may only be issued if the Hello has been successfully negotiated and the connection has control of the CNC (see enable sub–command below). This command has no parameters.

**Help**

The help command will return help information in text format over the telnet connection. If no parameters are specified, it will itemize the available commands. If a command is specified, it will provide usage information for the specified command. Help will respond regardless of whether a "Hello" has been successfully negotiated.

**HAL sub–commands:**

echo on | off

With get will return the current echo state, with set, sets the echo state. When echo is on, all

commands will be echoed upon receipt. This state is local to each connection.

verbose on | off

With get will return the current verbose state, with set, sets the verbose state. When in verbose mode is on, all set commands return positive acknowledgement in the form SET <COMMAND> ACK. In addition, text error messages will be issued when in verbose mode. This state is local to each connection.

enable <pwd> | off

With get will return On or Off to indicate whether the current connection is enabled to perform control functions. With set and a valid password, the current connection is enabled for control functions. "OFF" may not be used as a password and disables control functions for this connection.

config [TBD] comm\_mode ascii | binary

With get, will return the current communications mode. With set, will set the communications mode to the specified mode. The binary protocol is TBD.

comm\_prot <version no>

With get, returns the current protocol version used by the server, with set, sets the server to use the specified protocol version, provided it is lower than or equal to the highest version number supported by the server implementation.

Comps [<substring>]

Get only, returns all components beginning with the specified substring. If no substring is specified then it returns all components.

Pins [<substring>]

Get only, returns all information about all pins beginning with the specified substring. If no substring is specified then it returns all pins.

PinVals [<substring>]

Get only, returns only value information about all pins beginning with the specified substring. If no substring is specified then it returns all pins.

Signals [<substring>]

Get only, returns all information about all signals beginning with the specified substring. If no substring is specified then it returns all signals.

SigVals [<substring>]

Get only, returns only value information about all signals beginning with the specified substring. If no substring is specified then it returns all pins.

Params [<substring>]

Get only, returns all information about all parameters beginning with the specified substring. If no substring is specified then it returns all parameters.

ParamVals [<substring>]

Get only, returns only value information about all parameters beginning with the specified substring. If no substring is specified then it returns all pins parameters.

Functs [<substring>]

Get only, returns all information about all functions beginning with the specified substring. If no substring is specified then it returns all functions.

Threads

Get only, returns all information about all functions.

Comp <name>

Get only, returns the component matching the specified name.

Pin <name>

Get only, returns all information about the pin matching the specified name.

PinVal <name>

Get only, returns the value of the pin matching the specified name.

Sig <name>

Get only, returns all information about the pin matching the specified name.

SigVal <name>

Get only, returns just the value of the signal matching the specified name.

Param <name>

Get only, returns all information about the parameter matching the specified name.

ParamVal <name>

Get only, returns just the value of the parameter matching the specified name.

Funct <name>

Get only, returns all information about the parameter matching the specified name.

Thread <name>

Get only, returns all information about the thread matching the specified name.

LoadRt <name>

Set only, loads the real time executable specified by name.

Unload <name>

Set only, unloads the executable specified by name.

LoadUsr <name>

Set only, loads the user executable specified by name.

Linkps <pin name> <signal name>

Set only, links the specified pin to the specified signal.

Linksp <signal name> <pin name>

Set only, links the specified signal to the specified pin.

Linkpp <pin name 1> <pin name 2>

Set only, links the pin specified by pin 1 with the pin specified by pin 2.

Net <net list>

Set only, nets the specified net list.

Unlinkp <pin name 1> <pin name 2>

Set only, unlinks the specified pins.

Lock, Unlock, NewSig <name> <type>

Set only, creates the signal specified by name and of type specified by type.

DelSig <name>

Set only, deletes the signal specified by name.

SetP <name> <value>

Set only, sets the parameter specified by name to the value specified by value.

SetS <name> <value>

Set only, sets the signal specified by name to the value specified by value.

AddF <name> <thread> [<parameters>]

Set only, adds the function specified by name, to the thread specified by thread, with the optional parameters specified by parameters.

DelF <name>

Set only, deletes the function specified by name.

Save, Start, Stop

== SEE ALSO

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

It is not known if this interface currently works.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

halrun – manipulate the LinuxCNC HAL from the command line

**SYNOPSIS**

**halrun -h**

**halrun** [-I\*] [*halcmd\_opts*] [*filename[.hal|.tcl]*]

**halrun -T** [*halcmd\_opts*] [*filename[.hal|.tcl]*]

**halrun -U**

**DESCRIPTION**

The convenience script **halrun** can manipulate the HAL (Hardware Abstraction Layer) from the command line. When invoked, **halrun**:

- Sets up the realtime environment.
- Executes a command interpreter (**halcmd** or **haltcl**).
- (Optionally) runs an interactive session.
- Tears down the realtime environment.

If no filename is specified, an interactive session is started. The session will use **halcmd**(1) unless **-T** is specified in which case **haltcl**(1) will be used.

If a filename is specified and neither the **-I** nor the **-T** option is included, the filename will be processed by the command interpreter corresponding to the filename extension (**halcmd** or **haltcl**). After processing, the realtime environment will be torn down.

If a filename is specified and the **-I** or **-T** option is included, the file is processed by the appropriate command interpreter and then an interactive session is started for **halcmd** or **haltcl** according to the **-I** or **-T** option.

**OPTIONS****halcmd\_opts**

When a .hal file is specified, the **halcmd\_opts** are passed to **halcmd**. See the man page for **halcmd**(1). When a .tcl file is specified, the only valid options are: **-i** <INI file> **-f** <filename[.tcl|.hal]> (alternate means of specifying a file).

**-I**

Run an interactive **halcmd** session

**-T**

Run an interactive **haltcl** session.

**-U**

Forcibly cause the realtime environment to exit. It releases the HAL mutex, requests that all HAL components unload, and stops the realtime system. **-U** must be the only commandline argument.

**-h**

Display a brief help screen and exit.

**BUGS**

None known at this time.

**AUTHOR**

Original version by John Kasunich, as part of the LinuxCNC Enhanced Machine Controller project. Now includes major contributions by several members of the project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2003 John Kasunich.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**SEE ALSO**

halcmd(1), haltcl(1)

**NAME**

halsampler – sample data from HAL in realtime

**SYNOPSIS**

**halsampler** [*options*]

**DESCRIPTION**

**sampler**(9) and **halsampler** are used together to sample HAL data in real time and store it in a file.

**sampler** is a realtime HAL component that exports HAL pins and creates a FIFO in shared memory. It then begins sampling data from the HAL and storing it to the FIFO. **halsampler** is a non-realtime program that copies data from the FIFO to stdout, where it can be redirected to a file or piped to some other program.

**OPTIONS**

**-c** *CHAN*

instructs **halsampler** to read from FIFO *CHAN*. FIFOs are numbered from zero, and the default value is zero, so this option is not needed unless multiple FIFOs have been created.

**-n** *COUNT*

instructs **halsampler** to read *COUNT* samples from the FIFO, then exit. If **-n** is not specified, **halsampler** will read continuously until it is killed.

**-t**

instructs **halsampler** to tag each line by printing the sample number in the first column.

*FILENAME*

instructs **halsampler** to write to *FILENAME* instead of to stdout.

**USAGE**

A FIFO must first be created by loading **sampler**(9) with **halcmd loadrt** or a **loadrt** command in a HAL file. Then **halsampler** can be invoked to begin printing data from the FIFO to stdout.

Data is printed one line per sample. If **-t** was specified, the sample number is printed first. The data follows, in the order that the pins were defined in the config string. For example, if the **sampler** config string was "ffbs" then a typical line of output (without **-t**) would look like:

```
123.55 33.4 0 -12
```

**halsampler** prints data as fast as possible until the FIFO is empty, then it retries at regular intervals, until it is either killed or has printed *COUNT* samples as requested by **-n**. Usually, but not always, data printed by **halsampler** will be redirected to a file or piped to some other program.

The FIFO size should be chosen to absorb samples captured during any momentary disruptions in the flow of data, such as disk seeks, terminal scrolling, or the processing limitations of subsequent program in a pipeline. If the FIFO gets full and **sampler** is forced to overwrite old data, **halsampler** will print *overrun* on a line by itself to mark each gap in the sampled data. If **-t** was specified, gaps in the sequential sample numbers in the first column can be used to determine exactly how many samples were lost.

The data format for **halsampler** output is the same as for **halstreamer**(1) input, so *waveforms* captured with **halsampler** can be replayed using **halstreamer**. The **-t** option should not be used in this case.

**EXIT STATUS**

If a problem is encountered during initialization, **halsampler** prints a message to stderr and returns failure.

Upon printing *COUNT* samples (if **-n** was specified) it will shut down and return success. If it is terminated before printing the specified number of samples, it returns failure. This means that when **-n** is not specified, it will always return failure when terminated.

**SEE ALSO**

sampler(9), streamer(9), halstreamer(1)

**AUTHOR**

Original version by John Kasunich, as part of the LinuxCNC project. Improvements by several other members of the LinuxCNC development team.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2006 John Kasunich.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.



**NAME**

halscope – short description

**SYNOPSIS**

**halscope**

**DESCRIPTION**

**halscope** Software oscilloscope for LinuxCNC/HAL

Digital oscilloscope for viewing real time waveforms of HAL pins and signals

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

halshow – Show HAL parameters, pins and signals

**SYNOPSIS**

**halshow** [*options*] [*watchfile*]

**OPTIONS**

--help

(help)

--fformat *format\_string\_for\_float*, --iformat *format\_string\_for\_int*, --noprefs  
don't use preference file to save settings

For format see <https://www.tcl.tk/man/tcl8.6.11/TclCmd/format.html>

Example: "%0.5f" displays a float with 5 digits right of the decimal point

**DESCRIPTION**

**halshow** creates a GUI interface to view and interact with a running HAL session. It is documented in the PDF and HTML docs much more completely than is possible in a manpage:  
<https://linuxcnc.org/docs/html/hal/halshow.html>

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at </usr/share/doc/LinuxCNC/>.

**HISTORY**

LinuxCNC 2.9

- Added buttons for pin/parameter/signal manipulation.
- Added menu entries for setting update interval, adding manually.
- Added right-click menu for copy, set, unlink pin and remove from view.

**BUGS**

None known at this time.

**AUTHOR**

Raymond E. Henry

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

halstreamer – stream file data into HAL in real-time

**SYNOPSIS**

**halstreamer** [*options*]

**DESCRIPTION**

**streamer**(9) and **halstreamer** are used together to stream data from a file into the HAL in real-time. **streamer** is a real-time HAL component that exports HAL pins and creates a FIFO in shared memory. **hal\_streamer** is a non-realtime program that copies data from stdin into the FIFO, so that **streamer** can write it to the HAL pins.

**OPTIONS**

**-c** *CHAN*

Instructs **halstreamer** to write to FIFO *CHAN*. FIFOs are numbered from zero, and the default value is zero, so this option is not needed unless multiple FIFOs have been created.

*FILENAME*

Instructs **halsampler** to read from *FILENAME* instead of from stdin.

**USAGE**

A FIFO must first be created by loading **streamer**(9) with **halcmd loadrt** or a **loadrt** command in a HAL file. Then **halstreamer** can be invoked to begin writing data into the FIFO.

Data is read from stdin, and is almost always either redirected from a file or piped from some other program, since keyboard input would be unable to keep up with even slow streaming rates.

Each line of input must match the pins that are attached to the FIFO, for example, if the **streamer** config string was "ffbs" then each line of input must consist of two floats, a bit, and a signed integer, in that order and separated by whitespace. Floats must be formatted as required by **strtod**(3), signed and unsigned integers must be formatted as required by **strtol**(3) and **strtoul**(3), and bits must be either *0* or *1*.

Input lines that begin with *#* will be treated as comments and silently skipped.

**halstreamer** transfers data to the FIFO as fast as possible until the FIFO is full, then it retries at regular intervals, until it is either killed or reads EOF from stdin. Data can be redirected from a file or piped from some other program.

The FIFO size should be chosen to ride through any momentary disruptions in the flow of data, such as disk seeks. If the FIFO is big enough, **halstreamer** can be restarted with the same or a new file before the FIFO empties, resulting in a continuous stream of data.

The data format for **halstreamer** input is the same as for **halsampler**(1) output, so *waveforms* captured with **halsampler** can be replayed using **halstreamer**.

**EXIT STATUS**

If a problem is encountered during initialization, **halstreamer** prints a message to stderr and returns failure.

If a badly formatted line is encountered while writing to the FIFO, it prints a message to stderr, skips the line, and continues (this behavior may be revised in the future).

Upon reading EOF from the input, it returns success. If it is terminated before the input ends, it returns failure.

**SEE ALSO**

streamer(9), sampler(9), halsampler(1)

**AUTHOR**

Original version by John Kasunich, as part of the LinuxCNC project. Improvements by several other members of the LinuxCNC development team.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2006 John Kasunich.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

haltcl – manipulate the LinuxCNC HAL from the command line using a Tcl interpreter.

**SYNOPSIS**

**haltcl** [-i <INI file>] [filename]

**DESCRIPTION**

Tcl is a scripting language from the 90s that is very easy to extend. **haltcl** extends the regular Tcl interpreter with a set of commands to interact with HAL, i.e. it allows to manipulate the HAL (Hardware Abstraction Layer) from the command line using a Tcl interpreter. **haltcl** can optionally read commands from a file (filename), allowing complex HAL configurations to be set up with a single command.

**OPTIONS**

-i <INI file>

If specified, the INI file is read and used to create Tcl global variable arrays. An array is created for each SECTION of the INI file with elements for each ITEM in the section.

For example, if the INI file contains:

```
[SECTION_A]
ITEM_1 = 1
[SECTION_A]
ITEM_2 = 2
[SECTION_B]
ITEM_1 = 10
```

The corresponding Tcl variables are:

```
SECTION_A(ITEM_1) = 1
SECTION_A(ITEM_2) = 2
SECTION_B(ITEM_1) = 10
```

-ini <INI file> — declining usage, use -i <INI file>, **filename**

If specified, the Tcl commands of **filename** are executed. If no filename is specified, haltcl opens an interactive session.

**COMMANDS**

The executable **haltcl** includes the commands of a Tcl interpreter augmented with commands for the hal language as described for **halcmd**(1). The augmented commands can be listed with the command:

```
haltcl: hal --commands
```

```
addf alias delf delsig getp gets ptype stype help linkpp linkps linksp
list loadrt loadusr lock net newsig save setexact_for_test_suite_only
setp sets show source start status stop unalias unlinkp unload unloadrt
unloadusr unlock waitusr
```

Two of the augmented commands, *list* and *gets*, require special treatment to avoid conflict with Tcl built-in commands having the same names. To use these commands, precede them with the keyword *hal*:

```
hal list
hal gets
```

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**SEE ALSO**

halcmd(1), halrun(1)

**NAME**

halui – observe HAL pins and command LinuxCNC through NML

**SYNOPSIS**

**halui** [**-ini** <*path-to-INI*>]

**DESCRIPTION**

**halui** is used to build a User Interface using hardware knobs and switches. It exports a big number of pins, and acts accordingly when these change.

**OPTIONS**

**-ini** *filename*

Use the *filename* as the configuration file. Note: halui must find the nml file specified in the INI, usually that file is in the same folder as the INI, so it makes sense to run halui from that folder.

**USAGE**

When run, **halui** will export a large number of pins. A user can connect those to his physical knobs & switches & leds, and when a change is noticed halui triggers an appropriate event.

**halui** expects the signals to be debounced, so if needed (bad knob contact) connect the physical button to a HAL debounce filter first.

**PINS****Abort**

\*halui.abort bit in  
pin for clearing most errors

**Tool**

\*halui.tool.length-offset.a float out  
current applied tool length offset for the A axis

\*halui.tool.length-offset.b float out  
current applied tool length offset for the B axis

\*halui.tool.length-offset.c float out  
current applied tool length offset for the C axis

\*halui.tool.length-offset.u float out  
current applied tool length offset for the U axis

\*halui.tool.length-offset.v float out  
current applied tool length offset for the V axis

\*halui.tool.length-offset.w float out  
current applied tool length offset for the W axis

\*halui.tool.length-offset.x float out  
current applied tool length offset for the X axis

\*halui.tool.length-offset.y float out  
current applied tool length offset for the Y axis

\*halui.tool.length-offset.z float out  
current applied tool length offset for the Z axis

\*halui.tool.diameter float out  
Current tool diameter, or 0 if no tool is loaded.

\*halui.tool.number u32 out  
current selected tool

**Spindle**

**halui.spindle.N.brake-is-on\*** bit out  
status pin that tells us if brake is on

- halui.spindle.N.brake-off\*** bit in  
pin for deactivating the spindle brake
- halui.spindle.N.brake-on\*** bit in  
pin for activating the spindle brake
- halui.spindle.N.decrease\*** bit in  
a rising edge on this pin decreases the current spindle speed by 100
- halui.spindle.N.forward\*** bit in  
a rising edge on this pin makes the spindle go forward
- halui.spindle.N.increase\*** bit in  
a rising edge on this pin increases the current spindle speed by 100
- halui.spindle.N.is-on\*** bit out  
status pin telling if the spindle is on
- halui.spindle.N.reverse\*** bit in  
a rising edge on this pin makes the spindle go reverse
- halui.spindle.N.runs-backward\*** bit out  
status pin telling if the spindle is running backward
- halui.spindle.N.runs-forward\*** bit out  
status pin telling if the spindle is running forward
- halui.spindle.N.start\*** bit in  
a rising edge on this pin starts the spindle
- halui.spindle.N.stop\*** bit in  
a rising edge on this pin stops the spindle

### Spindle Override

(SO = spindle override. FO = feed override), **halui.spindle.N.override.count-enable** bit in (default: TRUE)

When TRUE, modify spindle override when counts changes.

- halui.spindle.N.override.counts** s32 in  
counts X scale = spindle override percentage
- halui.spindle.N.override.decrease** bit in  
pin for decreasing the SO (-=scale)
- halui.spindle.N.override.direct-value** bit in  
pin to enable direct spindle override value input
- halui.spindle.N.override.increase** bit in  
pin for increasing the SO (+=scale)
- halui.spindle.N.override.reset** bit in  
pin for resetting the scale SO value (scale=1.0)
- halui.spindle.N.override.scale** float in  
pin for setting the scale of counts for SO
- halui.spindle.N.override.value** float out  
current FO value

### Program

- halui.program.block-delete.is-on** bit out  
status pin telling that block delete is on
- halui.program.block-delete.off** bit in  
pin for requesting that block delete is off
- halui.program.block-delete.on** bit in



pin for requesting that block delete is on

**halui.program.is-idle** bit out

status pin telling that no program is running

**halui.program.is-paused** bit out

status pin telling that a program is paused

\***halui.program.is-running** bit out

status pin telling that a program is running

**halui.program.optional-stop.is-on** bit out

status pin telling that the optional stop is on

**halui.program.optional-stop.off** bit in

pin requesting that the optional stop is off

**halui.program.optional-stop.on** bit in

pin requesting that the optional stop is on

**halui.program.pause** bit in

pin for pausing a program

**halui.program.resume** bit in

pin for resuming a program

**halui.program.run** bit in

pin for running a program

**halui.program.step** bit in

pin for stepping in a program

**halui.program.stop** bit in

pin for stopping a program (note: this pin does the same thing as halui.abort)

## Mode

**halui.mode.auto** bit in

pin for requesting auto mode

**halui.mode.is-auto** bit out

pin for auto mode is on

**halui.mode.is-joint** bit out

pin showing joint by joint jog mode is on

**halui.mode.is-manual** bit out

pin for manual mode is on

**halui.mode.is-mdi** bit out

pin for MDI mode is on

**halui.mode.is-teleop** bit out

pin showing coordinated jog mode is on

**halui.mode.joint** bit in

pin for requesting joint by joint jog mode

**halui.mode.manual** bit in

pin for requesting manual mode

**halui.mode.mdi** bit in

pin for requesting MDI mode

**halui.mode.teleop** bit in

pin for requesting coordinated jog mode

**MDI (optional)****halui.mdi-command-XX** bit in

**halui** looks for INI variables named [HALUI]MDI\_COMMAND, and exports a pin for each command it finds. When the pin is driven TRUE, **halui** runs the specified MDI command. *XX* is a two digit number starting at

1. If no [HALUI]MDI\_COMMAND variables are set in the INI file, no halui.mdi-command-XX pins will be exported by halui.

**Mist coolant****halui.mist.is-on** bit out

pin for mist is on

**halui.mist.off** bit in

pin for stopping mist

**halui.mist.on** bit in

pin for starting mist

**Max-velocity****halui.max-velocity.count-enable** bit in (default: **TRUE**)\*

When True, modify max velocity when halui.max-velocity.counts changes.

**halui.max-velocity.counts** s32 in

When .count-enable is True, halui changes the max velocity in response to changes to this pin. It's usually connected to an MPG encoder on an operator's panel or jog pendant. When .count-enable is False, halui ignores this pin.

**halui.max-velocity.direct-value** bit in

When this pin is True, halui commands the max velocity directly to (.counts \* .scale). When this pin is False, halui commands the max velocity in a relative way: change max velocity by an amount equal to (change in .counts \* .scale).

**halui.max-velocity.increase** bit in

A positive edge (a False to True transition) on this pin increases the max velocity by the value of the .scale pin. (Note that halui always responds to this pin, independent of the .count-enable pin.)

**halui.max-velocity.decrease** bit in

A positive edge (a False to True transition) on this pin decreases the max velocity by the value of the .scale pin. (Note that halui always responds to this pin, independent of the .count-enable pin.)

**halui.max-velocity.scale** float in

This pin controls the scale of changes to the max velocity. Each unit change in .counts, and each positive edge on .increase and .decrease, changes the max velocity by .scale. The units of the .scale pin are machine-units per second.

**halui.max-velocity.value** float out

Current value for maximum velocity, in machine-units per second.

**Machine****halui.machine.units-per-mm** float out

pin for machine units-per-mm (inch:1/25.4, mm:1) according to INI file setting: [TRAJ]LINEAR\_UNITS

**halui.machine.is-on** bit out

pin for machine is On/Off

**halui.machine.off** bit in

pin for setting machine Off

**halui.machine.on** bit in

pin for setting machine On

**Joint**

$N$  = joint number (0 ... num\_joints-1)

**halui.joint.N.select** bit in  
pin for selecting joint  $N$

**halui.joint.N.is-selected** bit out  
status pin that joint  $N$  is selected

**halui.joint.N.has-fault** bit out  
status pin telling that joint  $N$  has a fault

**halui.joint.N.home** bit in  
pin for homing joint  $N$

**halui.joint.N.is-homed** bit out  
status pin telling that joint  $N$  is homed

**halui.joint.N.on-hard-max-limit** bit out  
status pin telling that joint  $N$  is on the positive hardware limit

**halui.joint.N.on-hard-min-limit** bit out  
status pin telling that joint  $N$  is on the negative hardware limit

**halui.joint.N.on-soft-max-limit** bit out  
status pin telling that joint  $N$  is on the positive software limit

**halui.joint.N.on-soft-min-limit** bit out  
status pin telling that joint  $N$  is on the negative software limit

**halui.joint.N.override-limits** bit out  
status pin telling that joint  $N$ 's limits are temporarily overridden

**halui.joint.N.unhome** bit in  
pin for unhoming joint  $N$

**halui.joint.selected** u32 out  
selected joint number (0 ... num\_joints-1)

**halui.joint.selected.has-fault** bit out  
status pin selected joint is faulted

**halui.joint.selected.home** bit in  
pin for homing the selected joint

**halui.joint.selected.is-homed** bit out  
status pin telling that the selected joint is homed

**halui.joint.selected.on-hard-max-limit** bit out  
status pin telling that the selected joint is on the positive hardware limit

**halui.joint.selected.on-hard-min-limit** bit out  
status pin telling that the selected joint is on the negative hardware limit

**halui.joint.selected.on-soft-max-limit** bit out  
status pin telling that the selected joint is on the positive software limit

**halui.joint.selected.on-soft-min-limit** bit out  
status pin telling that the selected joint is on the negative software limit

**halui.joint.selected.override-limits** bit out  
status pin telling that the selected joint's limits are temporarily overridden

**halui.joint.selected.unhome** bit in  
pin for unhoming the selected joint

**Joint jogging (N = joint number (0 ... num\_joints-1))****halui.joint.jog-deadband** float in

pin for setting jog analog deadband (jog analog inputs smaller/slower than this (in absolute value) are ignored).

**halui.joint.jog-speed** float in

pin for setting jog speed for plus/minus jogging.

**halui.joint.N.analog** float inpin for jogging the joint *N* using an float value (e.g. joystick). The value, typically set between 0.0 and  $\pm 1.0$ , is used as a jog-speed multiplier.**halui.joint.N.increment** float inpin for setting the jog increment for joint *N* when using increment-plus/minus**halui.joint.N\*\*.increment-minus** bit ina rising edge will will make joint *N* jog in the negative direction by the increment amount**halui.joint.N.increment-plus** bit ina rising edge will will make joint *N* jog in the positive direction by the increment amount**halui.joint.N.minus** bit inpin for jogging joint *N* in negative direction at the halui.joint.jog-speed velocity**halui.joint.N.plus** bit inpin for jogging joint *N* in positive direction at the halui.joint.jog-speed velocity**halui.joint.selected.increment** float in

pin for setting the jog increment for the selected joint when using increment-plus/minus

**halui.joint.selected.increment-minus** bit in

a rising edge will will make the selected joint jog in the negative direction by the increment amount

**halui.joint.selected.increment-plus** bit in

a rising edge will will make the selected joint jog in the positive direction by the increment amount

**halui.joint.selected.minus** bit in

pin for jogging the selected joint in negative direction at the halui.joint.jog-speed velocity

**halui.joint.selected.plus**

pin for jogging the selected joint bit in in positive direction at the halui.joint.jog-speed velocity

**Axis***L* = axis letter (xyzabcuvw)**halui.axis.L.select** bit in

pin for selecting axis by letter

**halui.axis.L.is-selected** bit outstatus pin that axis *L* is selected**halui.axis.L.pos-commanded** float out float out

Commanded axis position in machine coordinates

**halui.axis.L.pos-feedback** float out float out

Feedback axis position in machine coordinates

**halui.axis.L.pos-relative** float out float out

Commanded axis position in relative coordinates

**Axis Jogging***L* = axis letter (xyzabcuvw)**halui.axis.jog-deadband** float in

pin for setting jog analog deadband (jog analog inputs smaller/slower than this (in absolute value) are ignored)

- halui.axis.jog-speed** float in  
pin for setting jog speed for plus/minus jogging.
- halui.axis.L.analog** float in  
pin for jogging the axis L using an float value (e.g. joystick). The value, typically set between 0.0 and  $\pm 1.0$ , is used as a jog-speed multiplier.
- halui.axis.L.increment** float in  
pin for setting the jog increment for axis L when using increment-plus/minus
- halui.axis.L\*\*.increment-minus** bit in  
a rising edge will will make axis L jog in the negative direction by the increment amount
- halui.axis.L.increment-plus\*** bit in  
a rising edge will will make axis L jog in the positive direction by the increment amount
- halui.axis.L.minus** bit in  
pin for jogging axis L in negative direction at the halui.axis.jog-speed velocity
- halui.axis.L.plus** bit in  
pin for jogging axis L in positive direction at the halui.axis.jog-speed velocity
- halui.axis.selected** u32 out  
selected axis (by index: 0:x 1:y 2:z 3:a 4:b 5:cr 6:u 7:v 8:w)
- halui.axis.selected.increment** float in  
pin for setting the jog increment for the selected axis when using increment-plus/minus
- halui.axis.selected.increment-minus** bit in  
a rising edge will will make the selected axis jog in the negative direction by the increment amount
- halui.axis.selected.increment-plus** bit in  
a rising edge will will make the selected axis jog in the positive direction by the increment amount
- halui.axis.selected.minus** bit in  
pin for jogging the selected axis in negative direction at the halui.axis.jog-speed velocity
- halui.axis.selected.plus** FIXME MISSING  
pin for jogging the selected axis bit in in positive direction at the halui.axis.jog-speed velocity

#### Flood coolant

- halui.flood.is-on** bit out  
pin for flood is on
- halui.flood.off** bit in  
pin for stopping flood
- halui.flood.on** bit in  
pin for starting flood

#### Feed Override

- halui.feed-override.count-enable** bit in (default: **TRUE**)\*  
When TRUE, modify feed override when counts changes.
- halui.feed-override.counts** s32 in  
counts X scale = feed override percentage
- halui.feed-override.decrease** bit in  
pin for decreasing the FO (-=scale)
- halui.feed-override.direct-value** bit in  
pin to enable direct value feed override input
- halui.feed-override.increase** bit in  
pin for increasing the FO (+=scale)
- halui.feed-override.reset** bit in

pin for resetting the FO (scale=1.0)

**halui.feed-override.scale** float in  
pin for setting the scale on changing the FO

**halui.feed-override.value** float out  
current feed override value

### Rapid Override

**halui.rapid-override.count-enable** bit in (default: **TRUE**)\*  
When TRUE, modify rapid override when counts changes.

**halui.rapid-override.counts** s32 in  
counts X scale = rapid override percentage

**halui.rapid-override.decrease** bit in  
pin for decreasing the rapid override (-=scale)

**halui.rapid-override.direct-value** bit in  
pin to enable direct value rapid override input

**halui.rapid-override.increase** bit in  
pin for increasing the rapid override (+=scale)

**halui.rapid-override.reset** bit in  
pin for resetting the rapid override (scale=1.0)

**halui.rapid-override.scale** float in  
pin for setting the scale on changing the rapid override

**halui.rapid-override.value** float out  
current rapid override value

### E-stop

**halui.estop.activate bit** in  
pin for setting E-stop (LinuxCNC internal) On

**halui.estop.is-activated bit** out  
pin for displaying E-stop state (LinuxCNC internal) On/Off

**halui.estop.reset** bit in  
pin for resetting E-stop (LinuxCNC internal) Off

### Homing

**halui.home-all** bit in  
pin for requesting home-all (only available when a valid homing sequence is specified)

### SEE ALSO

axis(1), iocontrol(1)

### BUGS

None known at this time.

### AUTHOR

Written by Alex Joni, as part of the LinuxCNC project. Updated by John Thornton

### REPORTING BUGS

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>

### COPYRIGHT

Copyright © 2006 Alex Joni.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

hbmgui – Vismach Virtual Machine GUI

**DESCRIPTION**

**hbmgui** is one of the sample **Vismach** GUIs for LinuxCNC, simulating a Horizontal Boring Machine.

See the main LinuxCNC documentation for more details.

<https://linuxcnc.org/docs/html/gui/vismach.html>

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

hexagui – Vismach Virtual Machine GUI

**DESCRIPTION**

**hexagui** is one of the sample **Vismach** GUIs for LinuxCNC, simulating a Horizontal Boring Machine.

See the main LinuxCNC documentation for more details.

<https://linuxcnc.org/docs/html/gui/vismach.html>

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>

**COPYRIGHT**

Copyright © 2020 Andy Pugh. This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.



**NAME**

hy\_gt\_vfd – HAL non–realtime component for Huanyang GT–series VFDs

**SYNOPSIS**

hy\_gt\_vfd [*OPTIONS*]

**DESCRIPTION**

The hy\_gt\_vfd component interfaces a Huanyang GT–series VFD to the LinuxCNC HAL. The VFD is connected via RS–485 serial to the LinuxCNC computer.

**HARDWARE SETUP**

At least some Huanyang GT VFDs must be physically modified to enable Modbus communication.

The circuit board location marked "SW1" is identified in the manual as "Switch of terminal resistor for RS485 communication". On the only VFD I have experience with, the circuit board contained no switch at that location, instead holding a pair of crossed jumper wires (top–left pad connected to bottom–right pad, top–right to bottom–left). In this configuration, no Modbus communication is possible. We had to desolder the two crossed jumper wires and re–solder them parallel to each other (top–left to bottom–left, top–right to bottom–right).

**FIRMWARE SETUP**

The Huanyang GT VFD must be configure via the faceplate to talk Modbus with LinuxCNC. Consult the Operation section of the Huanyang GT–series Inverter Manual for details. Set the following parameters:

P0.01 = 2

Set Run Command Source to Modbus serial port.

P0.03

Set Maximum Frequency to the maximum frequency you want the VFD to output, in Hz.

P0.04

Set Upper Frequency Limit to the maximum frequency you want the VFD to output, in Hz. This should be the same as the value in P0.03.

P0.05

Set Lower Frequency Limit to the minimum frequency you want the VFD to output, in Hz.

P0.07 = 7

Set Frequency A Command Source to Modbus serial port.

P2.01 = ???

Set Motor Rated Power to the motor's power rating in kW.

P2.02 = ???

Set Motor Rated Frequency to the motor's max frequency in Hz.

P2.03 = ???

Set Motor Rated Speed to the motor's speed in RPM at its rated maximum frequency.

P2.04 = ???

Set Motor Rated Voltage to the motor's maximum voltage, in Volts.

P2.05 = ???

Set Motor Rated Current to the motor's maximum current, in Amps.

PC.00 = 1

Set Local Address to 1. This matches the default in the hy\_gt\_vfd driver, change this if your setup has special needs.

PC.01 = 5

Set baud rate selection to 5 (38400 bps). This matches the default in the hy\_gt\_vfd driver, change this if your setup has special needs.

0 = 1200 1 = 2400 2 = 4800 3 = 9600 4 = 19200 5 = 38400

PC.02 = 0

Set Data Format (8n1 RTU). This matches the default in the `hy_gt_vfd` driver, change this if your setup has special needs.

PC.03 = 1

Set Communication Delay Time to 1 ms. This is expected by the `hy_gt_vfd` driver.

## OPTIONS

**-b, --bits** *N*

(default 8) For Modbus communication. Set number of data bits to *N*. *N* must be between 5 and 8 inclusive.

**-p, --parity** [Even,Odd,None]

(default None) For Modbus communication. Set serial parity to Even, Odd, or None.

**-r, --rate** *N*

(default 38400) For Modbus communication. Set baud rate to *N*. It is an error if the rate is not one of the following: 1200, 2400, 4800, 9600, 19200, 38400

**-s, --stopbits** [1,2]

(default 1) For Modbus communication. Set serial stop bits to 1 or 2.

**-t, --target** *N*

(default 1) For Modbus communication. Set Modbus target (slave) number. This must match the device number you set on the Huanyang GT VFD.

**-d, --device** *PATH*

(default `/dev/ttyS0`) For Modbus communication. Set the name of the serial device node to use.

**-v, --verbose**

Turn on verbose mode.

**-S, --motor-max-speed** *RPM*

The motor's max speed in RPM. This must match the motor speed value configured in VFD register P2.03.

**-F, --max-frequency** *HZ*

This is the maximum output frequency of the VFD in Hz. It should correspond to the motor's rated max frequency, and to the maximum and upper limit output frequency configured in VFD register P0.03 and P0.04.

**-f, --min-frequency** *HZ*

This is the minimum output frequency of the VFD in Hz. It should correspond to the minimum output frequency configured in VFD register P0.05.

## PINS

**hy\_gt\_vfd.period** (float, in)

The period for the driver's update cycle, in seconds. This is how frequently the driver will wake up, check its HAL pins, and communicate with the VFD. Must be between 0.001 and 2.000 seconds. Default: 0.1 seconds.

**hy\_gt\_vfd.speed-cmd** (float, in)

The requested motor speed, in RPM.

**hy\_gt\_vfd.speed-fb** (float, out)

The motor's current speed, in RPM, reported by the VFD.

**hy\_gt\_vfd.at-speed** (bit, out)

True when the drive is on and at the commanded speed (within 2%), False otherwise.

**hy\_gt\_vfd.freq-cmd** (float, out)

The requested output frequency, in Hz. This is set from the `.speed-cmd` value, and is just shown for debugging purposes.

**hy\_gt\_vfd.freq-fb** (float, out)

The current output frequency of the VFD, in Hz. This is reported from the VFD to the driver.

**hy\_gt\_vfd.spindle-on** (bit, in)

Set this pin True to command the spindle on, at the speed requested on the .speed-cmd pin. Set this pin False to command the spindle off.

**hy\_gt\_vfd.output-voltage** (float, out)

The voltage that the VFD is currently providing to the motor, in Volts.

**hy\_gt\_vfd.output-current** (float, out)

The current that the motor is currently drawing from the VFD, in Amperes.

**hy\_gt\_vfd.output-power** (float, out)

The power that the motor is currently drawing from the VFD, in Watts.

**hy\_gt\_vfd.dc-bus-volts** (float, out)

The current voltage of the VFD's internal DC power supply, in Volts.

**hy\_gt\_vfd.modbus-errors** (u32, out)

A count of the number of modbus communication errors between the driver and the VFD. The driver is resilient against communication errors, but a large or growing number here indicates a problem that should be investigated.

**hy\_gt\_vfd.input-terminal** (float, out)

The VFD's input terminal register.

**hy\_gt\_vfd.output-terminal** (float, out)

The VFD's output terminal register.

**hy\_gt\_vfd.AI1** (float, out)

The VFD's AI1 register.

**hy\_gt\_vfd.AI2** (float, out)

The VFD's AI2 register.

**hy\_gt\_vfd.HDI-frequency** (float, out)

The VFD's HDI-frequency register.

**hy\_gt\_vfd.external-counter** (float, out)

The VFD's external counter register.

**hy\_gt\_vfd.fault-info** (float, out)

The VFD's fault info register in floating point representation. This is kept for backwards compatibility with existing setups and will be removed in the future.

**hy\_gt\_vfd.fault-info-code** (u32, out)

The VFD's fault code register value. Introduced in LinuxCNC version 2.10. 0x00 if no fault is detected, see GT Series Inverter Manual page 87 for list of fault codes.

## ISSUES

The VFD produces the output frequency that it sends to the motor by adding a manually specified offset to the frequency command it gets over modbus.

The manual offset is controlled by pressing the Up/Down arrows on the faceplate while the VFD is turning the motor.

If you command a speed on the .speed-cmd pin and get a different speed reported on the .speed-fb pin, first verify that the VFD registers listed in the FIRMWARE SETUP section above and the driver's command-line arguments all agree with the info on the motor's name plate. If you still aren't getting the speed you expect, zero the VFD's frequency offset by starting the motor running, then pressing the Up/Down buttons to zero the offset.

**AUTHOR**

Sebastian Kuzminsky

**LICENSE**

GPL-2.0+

**NAME**

hy\_vfd – HAL non–realtime component for Huanyang VFDs

**SYNOPSIS**

hy\_vfd [OPTIONS]

**DESCRIPTION**

This component connects the Huanyang VFD to the LinuxCNC HAL via a serial (RS–485) connection.

The Huanyang VFD must be configured via the face plate user interface to accept serial communications:

PD001 = 2

Set register PD001 (source of run commands) to 2 (communication port).

PD002 = 2

Set register PD002 (source of operating frequency) to 2 (communication port).

PD004

Set register PD004 (Base Frequency) according to motor specs. This is the rated frequency of the motor from the motor's name plate, in Hz.

PD005

Set register PD005 (max frequency) according to motor specs. This is the maximum frequency of the motor's power supply, in Hz.

PD011

Set register PD011 (min frequency) according to motor specs. This is the minimum frequency of the motor's power supply, in Hz.

PD141

Set register PD141 (rated motor voltage) according to motor name plate. This is the motor's maximum voltage, in Volts.

PD142

Set register PD142 (rated motor current) according to motor name plate. This is the motor's maximum current, in Ampere.

PD143

Set register PD143 (Number of Motor Poles) according to motor name plate.

PD144

Set register PD144 (rated motor revolutions) according to motor name plate. This is the motor's speed in RPM at 50 Hz. Note: This is not the motor's max speed (unless the max motor frequency happens to be 50 Hz)!

PD163 = 1

Set register PD163 (communication address) to 1. This matches the default in the hy\_vfd driver, change this if your setup has special needs.

PD164 = 2

Set register PD164 (baud rate) to 2 (19200 bps). This matches the default in the hy\_vfd driver, change this if your setup has special needs.

PD165 = 3

Set register PD165 (communication data method) to 3 (8n1 RTU). This matches the default in the hy\_vfd driver, change this if your setup has special needs. Note that the hy\_vfd driver only supports RTU communication, not ASCII.

Consult the Huanyang instruction manual for details on using the face plate to program the VFDs registers, and alternative values for the above registers.

Access to devices such as /dev/ttyUSB0 is restricted to members of the "dialout" group. If you see error messages such as **open: Permission denied ERROR Can't open the device /dev/ttyUSB0 (errno 13)**

Check your groups membership with the command **groups** Then add your user to the dialout group with **sudo addgroup your-username dialout** You will need to log out and back in again for this to take effect.

## OPTIONS

- d, --device *<path>*  
Set the name of the serial device node to use (defaults to /dev/ttyS0).
- g, --debug  
Turn on debug messages. Note that if there are serial errors, this may become annoying. Debug mode will cause all serial communication messages to be printed in hex on the terminal.
- y, --regdump  
Print the current value of all registers as soon as the VFD is enabled.
- n, --name *string*  
Sets the name of the HAL module. The HAL comp name will be set to *string*, and all pin and parameter names will begin with *string* (defaults to hy\_vfd).
- b, --bits *n*  
Set number of data bits to *n*, where *n* must be from 5 to 8 inclusive. This must match the setting in register PD165 of the Huanyang VFD (defaults to 8).
- p, --parity [even,odd,none]  
Set serial parity to even, odd, or none. This must match the setting in register PD165 of the Huanyang VFD (default odd).
- r, --rate *n*  
Set baud rate to *n*. It is an error if the rate is not one of the following: 110, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200. This must match the setting in register PD164 of the Huanyang VFD (defaults to 38400).
- s, --stopbits [1,2]  
Set serial stop bits to 1 or 2. This must match the setting in register PD165 of the HuanyangVFD (defaults to 1).
- t, --target *n*  
Set HYCOMM target (slave) number. This must match the device number you set on the Hyanyang VFD in register PD163 (defaults to 1).
- F, --max-frequency *n*  
If specified, program register PD005 of the VFD with the specified max frequency of *n* Hz (and use the same max frequency in the hy\_vfd driver). If not specified, read the max frequency to use from register PD005 of the VFD (default: read from VFD).
- f, --min-frequency *n*  
If specified, program register PD011 of the VFD with the specified minimum frequency of *n* Hz (and use the same minimum frequency in the hy\_vfd driver). If not specified, read the minimum frequency to use from register PD011 of the VFD (default: read from VFD).
- V, --motor-voltage *n*  
If specified, program register PD141 of the VFD with the specified max motor voltage of *n* Volts. If not specified, read the max motor voltage from register PD141 of the VFD (default: read from VFD).
- I, --motor-current *n*  
If specified, program register PD142 of the VFD with the specified max motor current of *n* Amps. If not specified, read the max motor current from register PD142 of the VFD (default: read from VFD).
- S, --motor-speed *n*  
(default: compute from value read from VFD P144) This command-line argument is the motor's max speed. If specified, compute the motor's speed at 50 Hz from this argument and from the motor's max frequency (from the --max-frequency argument or from P011 if --max-frequency is not specified) and program register PD144 of the VFD. If not specified, read the motor's speed at 50 Hz from register P144 of the VFD, and use that and the max frequency to compute the motor's max speed.

- P, --motor-poles *n*  
 (default: read value from VFD P143) This command-line argument is the number of poles in the motor. If specified, this value is sent to the VFD's register PD143. If not specified, the value is read from PD143 and reported on the corresponding HAL pin.
- x, --register PD*nnn*=*mmm* *n*  
 Set a specific register to a new value. Can be used to set up to 10 registers. Parameters will "stick" (but only after `hy_vfd.enable` has been set true) so to set more than ten parameters it is possible to repeatedly load the driver with a set of registers to set then enable (`setp hy_vfd.enable 1`) and unload (`unload hy_vfd`) the driver at a `halrun(1)` prompt. For example:

```
loadusr -W hy_vfd -d /ttyUSB0 --register PD014=30 --register PD015=30
```

Will set both ramp1 times to 3 seconds. The values should be scaled according to the manual data. The example above uses values with a resolution of 0.1 seconds, so the numbers are 10x larger than the required value.

## PINS

<name>.enable (bit, in)

Enable communication from the `hy_vfd` driver to the VFD.

<name>.SetF (float, out), <name>.OutF (float, out), <name>.OutA (float, out), <name>.Rott (float, out), <name>.DCV (float, out), <name>.ACV (float, out), <name>.Cont (float, out), <name>.Tmp (float, out), <name>.spindle-forward (bit, in), <name>.spindle-reverse (bin, in), <name>.spindle-on (bin, in), <name>.CNTR (float, out), <name>.CNST (float, out), <name>.CNST-run (bit, out), <name>.CNST-jog (bit, out), <name>.CNST-command-rf (bit, out), <name>.CNST-running (bit, out), <name>.CNST-jogging (bit, out), <name>.CNST-running-rf (bit, out), <name>.CNST-bracking (bit, out), <name>.CNST-track-start (bit, out), <name>.speed-command (float, in), <name>.spindle-speed-fb (float, out)

Current spindle speed as reported by Huanyang VFD (rpm).

<name>.spindle-speed-fb-rps (float, out)

Current spindle speed as reported by Huanyang VFD (rps).

<name>.spindle-at-speed-tolerance (float, in)

Spindle speed error tolerance. If the actual spindle speed is within `.spindle-at-speed-tolerance` of the commanded speed, then the `.spindle-at-speed` pin will go True. The default `.spindle-at-speed-tolerance` is 0.02, which means the actual speed must be within 2% of the commanded spindle speed.

<name>.spindle-at-speed (bit, out)

True when the current spindle speed is within `.spindle-at-speed-tolerance` of the commanded speed.

<name>.frequency-command (float, out), <name>.max-freq (float, out), <name>.base-freq (float, out), <name>.freq-lower-limit (float, out), <name>.rated-motor-voltage (float, out), <name>.rated-motor-current (float, out), <name>.rated-motor-rev (float, out), <name>.motor-poles (u32, out), <name>.hycomm-ok (bit, out), <name>.error-count (s32, RO), <name>.retval (u32, R0)

== AUTHOR

Sebastian Kuzminsky

## LICENSE

GPL

**NAME**

image-to-gcode – converts bitmap images to G-code

**SYNOPSIS**

**image-to-gcode**

**DESCRIPTION**

**image-to-gcode** converts a bitmap image to G-code interpreting the brightness of each pixel as a Z-height.

**SEE ALSO**

linuxcnc(1)

Detailed docs: <https://linuxcnc.org/docs/devel/html/gui/image-to-gcode.html>

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.



**NAME**

inivar – Query an INI file

**SYNOPSIS**

**inivar** **-var** *variable* [**-sec** *section*] [**-num** *occurrence\_number*] [**-tildeexpand**] [**-ini** *FILE*]

**DESCRIPTION**

Prints to stdout the INI file result of a variable-in-section search, useful for scripts that want to pick things out of INI files.

Uses *emc.ini* as default filename. *variable* needs to be supplied. If *section* is omitted, first instance of *variable* will be looked for in any section. Otherwise, only a match of the variable in *section* will be returned.

**OPTIONS**

**-var** *variable*

The variable to search for, if multiple matches exists and **-num** is not specified, the first match is returned.

**-sec** *section*

The section to search in, if omitted, all sections are searched.

**-num** *occurrence\_number*

The occurrence number specifies which instance of the variable within the *FILE*, and *section* if provided, should be returned. If omitted, the first matching occurrence is returned.

**-tildeexpand**

Replace the tilde (~) with the home directory path (equivalent to **\$(HOME)**) in the value obtained from *variable* in *FILE*.

**-ini** *FILE*

The INI file to search in, defaults to *emc.ini*.

**EXIT STATUS**

**0**

Success.

**1**

*variable* was not found.

**-1**

Failure.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright (c) 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

io, iocontrol – interacts with HAL or G-code in non-realtime

**SYNOPSIS**

[EMCIO] EMCIO = io

**DESCRIPTION**

I/O control handles I/O tasks like coolant, toolchange and E-stop. The signals are turned on and off in non-realtime with G-code or in the case of E-stop in HAL.

The following pins are created by the non-realtime IO controller, usually found in `$LINUXCNC_HOME/bin/io`.

iocontrol is a non-realtime process – if you have strict timing requirements or simply need more I/O, consider using the realtime synchronized I/O provided by **motion(9)** instead.

The INI file is searched for in the directory from which halcmd was run, unless an absolute path is specified.

**PINS**

**iocontrol.0.coolant-flood** (Bit, Out)

TRUE when flood coolant is requested.

**iocontrol.0.coolant-mist** (Bit, Out)

TRUE when mist coolant is requested.

**iocontrol.0.emc-enable-in** (Bit, In)

Should be driven FALSE when an external E-stop condition exists.

**iocontrol.0.tool-change** (Bit, Out)

TRUE when a tool change is requested.

**iocontrol.0.tool-changed** (Bit, In)

Should be driven TRUE when a tool change is completed.

**iocontrol.0.tool-number** (s32, Out)

Current tool number.

**iocontrol.0.tool-prep-number** (s32, Out)

The number of the next tool, from the RS274NGC T-word.

**iocontrol.0.tool-prep-pocket** (s32, Out)

This is the pocket number (location in the tool storage mechanism) of the tool requested by the most recent T-word.

**iocontrol.0.tool-prepare** (Bit, Out)

TRUE when a  $T_n$  tool prepare is requested.

**iocontrol.0.tool-prepared** (Bit, In)

Should be driven TRUE when a tool prepare is completed.

**iocontrol.0.user-enable-out** (Bit, Out)

FALSE when an internal E-stop condition exists.

**iocontrol.0.user-request-enable** (Bit, Out)

TRUE when the user has requested that E-stop be cleared.

**iocontrol.0.tool-prep-index** (s32, Out)

IO's internal array index of the prepped tool requested by the most recent T-word. 0 if no tool is prepped. On Random toolchanger machines this is the tool's pocket number (i.e., the same as the tool-prep-pocket pin), on Non-random toolchanger machines this is a small integer corresponding to the tool's location in the internal representation of the tool table. This parameter returns to 0 after a successful tool change (M6).

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**AUTHOR**

Derived from a work by Fred Proctor & Will Shackleford.

**COPYRIGHT**

Copyright © 2004 the LinuxCNC project.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

latency-histogram – plot a histogram of machine latency

**SYNOPSIS**

**latency-histogram** [-?|--help] [--base *ns*] [--servo *ns*] [--bbinsize *ns*] [--sbinsize *ns*] [--bbins *ns*] [--sbins *ns*] [--logscale 0|1] [--text *note*] [--show] [--nobase] [--verbose] [--nox]

**DESCRIPTION**

The latency test is important when configuring a LinuxCNC system. An adjunct to the standard latency-test latency-histogram plots the distribution of latency. This can be useful to get a feel for how frequent the high latency excursions are.

LinuxCNC and HAL should not be running, stop with halrun -U. Large number of bins and/or small binsizes will slow updates. For single thread, specify **---nobase** (and options for servo thread). Measured latencies outside of the +/- bin range are reported with special end bars. Use **---show** to show count for the off-chart [pos|neg] pin.

More details: <https://linuxcnc.org/docs/html/install/latency-test.html>

**OPTIONS**

**-?, --help**

Show options and exit.

**\*---base\*\_ ns\_**

base thread interval, default: 25000, min: 5000

**\*---servo\*\_ ns\_**

servo thread interval, default: 1000000, min: 25000

**\*---bbinsize\*\_ ns\_**

base bin size, default: 100

**\*---sbinsize\*\_ ns\_**

servo bin size, default: 100

**\*---bbins\*\_ ns\_**

base bins, default: 200

**\*---sbins\*\_ ns\_**

servo bins, default: 200

**\*---logscale\*\_ 0|1\_**

y axis log scale, default: 1

**\*---text\*\_ note\_**

additional note, default: ""

**---show**

show count of undisplayed bins

**---nobase**

servo thread only

**---verbose**

progress and debug

**---nox**

no GUI, display elapsed, min, max, sdev for each thread

**SEE ALSO**

latency-plot(1), latency-test(1), linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at */usr/share/doc/linuxcnc/*.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

latency-plot – another way to view latency numbers

**SYNOPSIS**

```
latency-plot [ -? | --help ] [ -H | --hal ] [ -b | --base ns ] [ -s | --servo ns ] [ -t | --time ms ] [ --relative ] [ --actual ]
```

**DESCRIPTION**

**latency-plot** makes a strip chart recording for a base and a servo thread. It may be useful to see spikes in latency when other applications are started or used. Mainly superseded by latency-histogram.

LinuxCNC and HAL should not be running, stop with *halrun -U*.

More details: <https://linuxcnc.org/docs/html/install/latency-test.html>

**OPTIONS**

**-?, --help**  
Show options and exit.

**-H, --hal, -b, --base *ns***  
base thread interval in nanoseconds, default: 25000

**-s, --servo *ns***  
servo thread interval in nanoseconds, default: 1000000

**-t, --time *ms***  
report interval in milliseconds, default: 1000, min: 100, max: 10000

**--relative**  
relative clock time (default)

**--actual**  
actual clock time

**SEE ALSO**

**latency-histogram(1)**, **latency-test(1)**, **linuxcnc(1)**

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at */usr/share/doc/linuxcnc/*.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

latency-test – test the realtime system latency

**SYNOPSIS**

**latency-test** [**--help**] [*base-period* [*servo-period*]]

**DESCRIPTION**

**latency-test** runs a simple latency test.

Use **latency-test --help** for a description of available options.

**SEE ALSO**

linuxcnc(1)

<https://linuxcnc.org/docs/html/install/latency-test.html>

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

lineardelta – Vismach Virtual Machine GUI

**DESCRIPTION**

**lineardelta** is one of the sample **Vismach** GUIs for LinuxCNC, simulating a delta robot with linear actuators

**SEE ALSO**

linuxcnc(1)

See the main LinuxCNC documentation for more details. <https://linuxcnc.org/docs/html/gui/vismach.html>

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.



**NAME**

linuxcnc\_info – collects information about the LinuxCNC version and the host

**SYNOPSIS**

**linuxcnc\_info**

**DESCRIPTION**

**linuxcnc\_info** supplies information about the LinuxCNC version and system info. It creates a text file and opens it in the default text editor.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at [/usr/share/doc/LinuxCNC/](#).

**BUGS**

It appears to hang until the text editor is closed.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

linuxcnc\_module\_helper – controls root access for system hardware

**SYNOPSIS**

**linuxcnc\_module\_helper**

**DESCRIPTION**

This module exists to give root access to system hardware for LinuxCNC. It is not directly useful to users.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at </usr/share/doc/LinuxCNC/>.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

linuxcnc\_var – retrieves LinuxCNC variables

**SYNOPSIS**

**linuxcnc\_var** [ *varname* | all ]

**DESCRIPTION**

FIXME: missing

**OPTIONS**

Option *all* returns *varname=value* for all supported varnames

Varnames supported: LINUXCNCVERSION LINUXCNC\_AUX\_GLADEVCP  
LINUXCNC\_AUX\_EXAMPLES REALTIME RTS HALLIB\_DIR

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at </usr/share/doc/LinuxCNC/>.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

linuxcnclcd – LinuxCNC Graphical User Interface for LCD character display

**SYNOPSIS**

**linuxcnclcd** **-ini** *<INI file>*

**DESCRIPTION**

**linuxcnclcd** is one of the Graphical User Interfaces (GUI) for LinuxCNC. It gets typically run by the runscript. Linuxcnclcd is designed to run on a 4 x 20 LCD character display. It is not clear if it has ever worked.

**OPTIONS**

*INI file*

The INI file is the main piece of an LinuxCNC configuration. It is not the entire configuration; there are various other files that go with it (NML files, HAL files, TBL files, VAR files). It is, however, the most important one, because it is the file that holds the configuration together. It can adjust a lot of parameters itself, but it also tells **LinuxCNC** which other files to load and use.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

linuxcncmkdesktop – create a desktop icon for LinuxCNC

**SYNOPSIS**

**linuxcncmkdesktop**

**DESCRIPTION**

**linuxcncmkdesktop** Script used by pickconfig to create a desktop icon for LinuxCNC.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at </usr/share/doc/LinuxCNC/>.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

linuxcncrsh – text-mode interface for commanding LinuxCNC over the network

**SYNOPSIS**

**linuxcncrsh** [OPTIONS] [-- LINUXCNC\_OPTIONS]

**DESCRIPTION**

The command **linuxcncrsh** is a fully-functional text-based user interface for LinuxCNC. Instead of popping up a GUI window like **AXIS(1)** and **Touchy(1)** do, it processes text-mode commands that it receives via the network. A human (or a program) can interface with **linuxcncrsh** using **telnet(1)**, **nc(1)** or similar programs.

It cannot be stressed enough that all features of LinuxCNC are available via the **linuxcncrsh** interface. It can substitute a graphical UI or be started in addition to it. If your environment needs to control/monitor more than one machine from a single location, this text interface may raise your interest. Also, a speech to text interface may be easier to adapt to this text-based interface than to a graphical one.

**OPTIONS**

**-p, --port** *PORT\_NUMBER*

Specify the port for linuxcncrsh to listen on. Defaults to 5007 if omitted.

**-n, --name** *SERVER\_NAME*

Sets the server name that linuxcncrsh will use to identify itself during handshaking with a new client. Defaults to EMCNETSVR if omitted.

**-w, --connectpw** *PASSWORD*

Specify the connection password to use during handshaking with a new client. Note that the password is sent in the clear, so it can be read by anyone who can read packets on the network between the server and the client. Defaults to EMC if omitted.

**-e, --enablepw** *PASSWORD*

Specify the password required to enable LinuxCNC via linuxcncrsh. Note that the password is sent in the clear, so it can be read by anyone who can read packets on the network between the server and the client. Defaults to EMCTOO if omitted.

**-s, --sessions** *MAX\_SESSIONS*

Specify the maximum number of simultaneous connections. Defaults to -1 (no limit) if not specified.

In addition to the options listed above, linuxcncrsh accepts an optional special LINUXCNC\_OPTION at the end:

**-ini** *LINUXCNC\_INI\_FILE*

LinuxCNC INI file to use. The **-ini** option **must** be preceded by two dashes: "--". Defaults to *emc.ini* if omitted.

**STARTING LINUXCNCRSH**

To use **linuxcncrsh** instead of a normal LinuxCNC GUI like **AXIS** or **Touchy**, specify it in your INI file like this:

```
[DISPLAY]
```

```
DISPLAY=linuxcncrsh
```

To use linuxcncrsh in addition to a normal GUI, you can either 1. start it at the end of your HAL file, or 2. run it by hand in a terminal window.

To start it from HAL, add a line like this to the end of your HAL file:

```
loadusr linuxcncrsh [OPTIONS] [-- LINUXCNC_OPTIONS]
```

To start it from the terminal, run linuxcncrsh manually like this:

```
linuxcncrsh [OPTIONS] [-- LINUXCNC_OPTIONS]
```

## CONNECTING

Once LinuxCNC is up and **linuxcncrsh** is running, you can connect to it using **telnet** or **nc** or similar:

**telnet** *HOST PORT*

*HOST* is the hostname or IP address of the computer running linuxcncrsh, and *PORT* is the port it's listening on (5007 if you did not give linuxcncrsh the `--port` option).

## NETWORK PROTOCOL

linuxcncrsh accepts TCP connections on the port specified by the `--port` option, or 5007 if not specified.

The client sends requests, and the linuxcncrsh server returns replies. Requests consist of a command word followed by optional command-specific parameters. Requests and most request parameters are case insensitive. The exceptions are passwords, file paths and text strings.

Requests to linuxcncrsh are terminated with line endings, any combination of one or more `\r` and `\n` characters. Replies from linuxcncrsh are terminated with the sequence `\r\n`.

The supported commands are as follows:

**hello** *<password>* *<client>* *<version>*

*<password>* must match linuxcncrsh's connect password, or "EMC" if no `--connectpw` was supplied. The three arguments may not contain whitespace. If a valid password was entered the server will respond with:

```
HELLO ACK <ServerName> <ServerVersion>
```

If an invalid password or any other syntax error occurs then the server responds with:

```
HELLO NAK
```

**get** *<subcommand>* [*<parameters>*]

The get command takes one of the LinuxCNC sub-commands (described in the section **LinuxCNC Subcommands**, below) and zero or more additional subcommand-specific parameters.

**set** *<subcommand>* *<parameters>*

The set command takes one of the LinuxCNC sub-commands (described in the section **LinuxCNC Subcommands**, below) and one or more additional parameters.

**quit**

The quit command disconnects the associated socket connection.

**shutdown**

The shutdown command tells LinuxCNC to shutdown and disconnect the session. This command may only be issued if the Hello has been successfully negotiated and the connection has control of the CNC (see **enable** subcommand in the **LinuxCNC Subcommands** section, below).

**help**

The help command will return help information in text format over the connection. If no parameters are specified, it will itemize the available commands. If a command is specified, it will provide usage information for the specified command. Help will respond regardless of whether a "Hello" has been successfully negotiated.

## LINUXCNC SUBCOMMANDS

Subcommands for **get** and **set** are:

**echo** {on|off}

With get, any on/off parameter is ignored and the current echo state is returned. With set, sets the echo

state as specified. Echo defaults to on when the connection is first established. When echo is on, all commands will be echoed upon receipt. This state is local to each connection.

**verbose** { on|off }

With get, any on/off parameter is ignored and the current verbose state is returned. With set, sets the verbose state as specified. When verbose mode is on all set commands return positive acknowledgement in the form

SET <COMMAND> ACK

and text error messages will be issued (FIXME: I don't know what this means). The verbose state is local to each connection, and starts out OFF on new connections.

**enable** { <passwd> | off }

The session's enable state indicates whether the current connection is enabled to perform control functions. With get, any parameter is ignored, and the current enable state is returned. With set and a valid password matching linuxcncrsh's `--enablepw` (EMCTOO if not specified), the current connection is enabled for control functions. "OFF" may not be used as a password and disables control functions for this connection.

**config** [TBD]

Unused, ignore for now.

**comm\_mode** { ascii | binary }

With get, any parameter is ignored and the current communications mode is returned. With set, will set the communications mode to the specified mode. The ASCII mode is the text request/reply mode, the binary protocol is not currently designed or implemented.

**comm\_prot** <version>

With get, any parameter is ignored and the current protocol version used by the server is returned.

With set, sets the server to use the specified protocol version, provided it is lower than or equal to the highest version number supported by the server implementation.

**inifile**

Not currently implemented! With get, returns the string *emc.ini*. Should return the full path and file name of the current configuration INI file. Setting this does nothing.

**plat**

With get, returns the string *Linux*.

**ini** <var> <section>

Not currently implemented, do not use! Should return the string value of <var> in section <section> of the INI file.

**debug** <value>

With get, any parameter is ignored and the current integer value of EMC\_DEBUG is returned. Note that the value of EMC\_DEBUG returned is the from the UI's INI file, which may be different than emc's INI file. With set, sends a command to the EMC to set the new debug level, and sets the EMC\_DEBUG global here to the same value. This will make the two values the same, since they really ought to be the same.

**wait\_mode** { received | done }

The wait\_mode setting controls the wait after receiving a command. It can be "received" (after the command was sent and received) or "done" (after the command was done). With get, any parameter is ignored and the current wait\_mode setting is returned. With set, set the wait\_mode setting to the specified value.

**wait** { received | done }

With set, force a wait for the previous command to be received, or done.

**set\_timeout** <timeout>

With set, set the timeout for commands to return to <timeout> seconds. Timeout is a real number. If



it's  $\infty$ , it means wait forever. Default is 0.0, wait forever.

**update** { none | auto }

The update mode controls whether to return fresh or stale values for "get" requests. When the update mode is "none" it returns stale values, when it's "auto" it returns fresh values. Defaults to "auto" for new connections. Set this to "none" if you like to be confused.

**error**

With get, returns the current error string, or "ok" if no error.

**operator\_display**

With get, returns the current operator display string, or "ok" if none.

**operator\_text**

With get, returns the current operator text string, or "ok" if none.

**time**

With get, returns the time, in seconds, from the start of the epoch. This starting time depends on the platform.

**estop** { on | off }

With get, ignores any parameters and returns the current estop setting as "on" or "off". With set, sets the estop as specified. E-stop "on" means the machine is in the estop state and won't run.

**machine** { on | off }

With get, ignores any parameters and returns the current machine power setting as "on" or "off". With set, sets the machine on or off as specified.

**mode** { manual | auto | mdi }

With get, ignores any parameters and returns the current machine mode. With set, sets the machine mode as specified.

**mist** { on | off }

With get, ignores any parameters and returns the current mist coolant setting. With set, sets the mist setting as specified.

**flood** { on | off }

With get, ignores any parameters and returns the current flood coolant setting. With set, sets the flood setting as specified.

**spindle** { forward | reverse | increase | decrease | constant | off } { <spindle> }

With get, any parameter is ignored and the current spindle state is returned as "forward", "reverse", "increase", "decrease", or "off". With set, sets the spindle as specified. Note that "increase" and "decrease" will cause a speed change in the corresponding direction until a "constant" command is sent. If "spindle" is omitted, spindle 0 is selected. If -1, all spindles are selected.

**brake** { on | off } { <spindle> }

With get, any parameter is ignored and the current brake setting is returned. With set, the brake is set as specified. If "spindle" is omitted, spindle 0 is selected. If -1, all spindles are selected.

**tool**

With get, returns the id of the currently loaded tool.

**tool\_offset**

With get, returns the currently applied tool length offset.

**load\_tool\_table** <file>

With set, loads the tool table specified by <file>.

**home** { 0|1|2|... } | -1

With set, homes the indicated joint or, if -1, homes all joints.

**jog\_stop** *joint\_number*|*axis\_letter* With set, stop any in-progress jog on the specified joint or axis. If TELEOP\_ENABLE is OFF, use *joint\_number*. If TELEOP\_ENABLE is ON, use *axis\_letter*.

**jog** *joint\_number* | *axis\_letter* <*speed*>

With set, jog the specified joint or axis at <*speed*>; sign of speed is direction. If TELEOP\_ENABLE is OFF, use *joint\_number*; If TELEOP\_ENABLE is ON, use *axis\_letter*.

**jog\_incr** *jog\_number* | *axis\_letter* <*speed*> <*incr*>

With set, jog the indicated joint or axis by increment <*incr*> at the <*speed*>; sign of speed is direction. If TELEOP\_ENABLE is OFF, use *joint\_number*. If TELEOP\_ENABLE is ON, use *axis\_letter*.

**feed\_override** <*percent*>

With get, any parameter is ignored and the current feed override is returned (as a percentage of commanded feed). With set, sets the feed override as specified.

**spindle\_override** <*percent*> {<*spindle*>}

With get, any parameter is ignored and the current spindle override is returned (as a percentage of commanded speed). With set, sets the spindle override as specified. If "spindle" is omitted, spindle 0 is selected. If -1, all spindles are selected.

**abs\_cmd\_pos** [{0|1|...}]

With get, returns the specified axis' commanded position in absolute coordinates. If no axis is specified, returns all axes' commanded absolute position.

**abs\_act\_pos** [{0|1|...}]

With get, returns the specified axis' actual position in absolute coordinates. If no axis is specified, returns all axes' actual absolute position.

**rel\_cmd\_pos** [{0|1|...}]

With get, returns the specified axis' commanded position in relative coordinates, including tool length offset. If no axis is specified, returns all axes' commanded relative position.

**rel\_act\_pos** [{0|1|...}]

With get, returns the specified axis' actual position in relative coordinates, including tool length offset. If no axis is specified, returns all axes' actual relative position.

**joint\_pos** [{0|1|...}]

With get, returns the specified joint's actual position in absolute coordinates, excluding tool length offset. If no joint is specified, returns all joints' actual absolute position.

**pos\_offset** [{X|Y|Z|R|P|W}]

With get, returns the position offset associated with the world coordinate provided.

**joint\_limit** [{0|1|...}]

With get, returns limit status of the specified joint as "ok", "minsoft", "minhard", "maxsoft", or "maxhard". If no joint number is specified, returns the limit status of all joints.

**joint\_fault** [{0|1|...}]

With get, returns the fault status of the specified joint as "ok" or "fault". If no joint number is specified, returns the fault status of all joints.

**joint\_homed** [{0|1|...}]

With get, returns the homed status of the specified joint as "homed" or "not". If no joint number is specified, returns the homed status of all joints.

**mdi** <*string*>

With set, sends <*string*> as an MDI command.

**task\_plan\_init**

With set, initializes the program interpreter.

**open** <*filename*>

With set, opens the named file. The <*filename*> is opened by linuxcnc, so it should either be an absolute path or a relative path starting in the LinuxCNC working directory (the directory of the active INI file).

**run** [<*StartLine*>]

With set, runs the opened program. If no StartLine is specified, runs from the beginning. If a StartLine is specified, start line, runs from that line. A start line of -1 runs in verify mode.

**pause**

With set, pause program execution.

**resume**

With set, resume program execution.

**abort**

With set, abort program or MDI execution.

**step**

With set, step the program one line.

**program**

With get, returns the name of the currently opened program, or "none".

**program\_line**

With get, returns the currently executing line of the program.

**program\_status**

With get, returns "idle", "running", or "paused".

**program\_codes**

With get, returns the string for the currently active program codes.

**joint\_type** [*<joint>*]

With get, returns "linear", "angular", or "custom" for the type of the specified joint (or for all joints if none is specified).

**joint\_units** [*<joint>*]

With get, returns "inch", "mm", "cm", or "deg", "rad", "grad", or "custom", for the corresponding native units of the specified joint (or for all joints if none is specified). The type of the axis (linear or angular) is used to resolve which type of units are returned. The units are obtained heuristically, based on the EMC\_AXIS\_STAT::units numerical value of user units per mm or deg. For linear joints, something close to 0.03937 is deemed "inch", 1.000 is "mm", 0.1 is "cm", otherwise it's "custom". For angular joints, something close to 1.000 is deemed "deg", PI/180 is "rad", 100/90 is "grad", otherwise it's "custom".

**program\_units**

Synonym for program\_linear\_units.

**program\_linear\_units**

With get, returns "inch", "mm", "cm", or "none", for the corresponding linear units that are active in the program interpreter.

**program\_angular\_units**

With get, returns "deg", "rad", "grad", or "none" for the corresponding angular units that are active in the program interpreter.

**user\_linear\_units**

With get, returns "inch", "mm", "cm", or "custom", for the corresponding native user linear units of the LinuxCNC trajectory level. This is obtained heuristically, based on the EMC\_TRAJ\_STAT::linearUnits numerical value of user units per mm. Something close to 0.03937 is deemed "inch", 1.000 is "mm", 0.1 is "cm", otherwise it's "custom".

**user\_angular\_units**

Returns "deg", "rad", "grad", or "custom" for the corresponding native user angular units of the LinuxCNC trajectory level. Like with linear units, this is obtained heuristically.

**display\_linear\_units**

With get, returns "inch", "mm", "cm", or "custom", for the linear units that are active in the display. This is effectively the value of linearUnitConversion.

**display\_angular\_units**

With get, returns "deg", "rad", "grad", or "custom", for the angular units that are active in the display. This is effectively the value of angularUnitConversion.

**linear\_unit\_conversion** { inch | mm | cm | auto }

With get, any parameter is ignored and the active unit conversion is returned. With set, sets the unit to be displayed. If it's "auto", the units to be displayed match the program units.

**angular\_unit\_conversion** { deg | rad | grad | auto }

With get, any parameter is ignored and the active unit conversion is returned. With set, sets the units to be displayed. If it's "auto", the units to be displayed match the program units.

**probe\_clear**

With set, clear the probe tripped flag.

**probe\_tripped**

With get, return the probe state – has the probe tripped since the last clear?

**probe\_value**

With get, return the current value of the probe signal.

**probe** <x> <y> <z>

With set, move toward a certain location. If the probe is tripped on the way stop motion, record the position and raise the probe tripped flag.

**teleop\_enable** [ on | off ]

With get, any parameter is ignored and the current teleop mode is returned. With set, sets the teleop mode as specified.

**kinematics\_type**

With get, returns the type of kinematics functions used (identity=1, serial=2, parallel=3, custom=4).

**override\_limits** { on | off }

With get, any parameter is ignored and the override\_limits setting is returned. With set, the override\_limits parameter is set as specified. If override\_limits is on, disables end of travel hardware limits to allow jogging off of a limit. If parameters is off, then hardware limits are enabled.

**optional\_stop** {0|1}

With get, any parameter is ignored and the current "optional stop on M1" setting is returned. With set, the setting is set as specified.

**EXAMPLE SESSION**

This section shows an example session to the local machine (**localhost**). Bold items are typed by you, non-bold is machine output. Default values are shown for **--port** *PORT\_NUMBER* (**5007**), **--connectpw** *PASSWORD* (**EMC**), and **--enablepw** *PASSWORD* (**EMCTOO**).

The user connects to linuxcncrsh, handshakes with the server (hello), enables machine commanding from this session (set enable), brings the machine out of E-stop (set estop off) and turns it on (set machine on), homes all the axes, switches the machine to mdi mode, sends an MDI G-code command, then disconnects and shuts down LinuxCNC.

```
> *telnet localhost 5007* +
Trying 127.0.0.1... +
Connected to 127.0.0.1 +
Escape character is '^]'. +
*hello EMC user-typing-at-telnet 1.0* +
HELLO ACK EMCNETSVR 1.1 +
*set enable EMCTOO* +
set enable EMCTOO +
*set mode manual* +
set mode manual +
```

```
*set estop off* +
set estop off +
*set machine on* +
set machine on +
*set home 0* +
set home 0 +
*set home 1* +
set home 1 +
*set home 2* +
set home 2 +
*set mode mdi* +
set mode mdi +
*set mdi g0x1* +
set mdi g0x1 +
*help* +
help +
Available commands: Hello <password> <client name> <protocol version>
Get <emc command> Set <emc command> Shutdown Help <command> +
*help get* +
help get +
Usage: Get <emc command> Get commands require that a hello has been
successfully negotiated. Emc command may be one of: Abs_act_pos
Abs_cmd_pos +
* ... * +
*shutdown* +
shutdown +
Connection closed by foreign host.
```

**NAME**

linuxcncsvr – Allows network access to LinuxCNC internals via NML

**SYNOPSIS**

**linuxcncsvr**

**DESCRIPTION**

FIXME: Missing

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

linuxcnc`top` – live LinuxCNC status description

**SYNOPSIS**

**linuxcnc`top` -ini** *INIFILE*

**DESCRIPTION**

**linuxcnc`top`** displays much of the LinuxCNC state in a live format similar to the Linux "top" command.

It is more fully documented in the AXIS GUI documentation but can be run standalone or with other GUIs.

**SEE ALSO**

linuxcnc(1)

<https://linuxcnc.org/docs/html/gui/axis.html>

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

maho600gui – Vismach Virtual Machine GUI

**DESCRIPTION**

**maho600gui** is one of the sample **Vismach** GUIs for LinuxCNC, simulating a Maho 600 CNC Milling Machine.

See the main LinuxCNC documentation for more details.

<https://linuxcnc.org/docs/html/gui/vismach.html>

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.



**NAME**

max5gui – Vismach Virtual Machine GUI

**DESCRIPTION**

**max5gui** is one of the sample **Vismach** GUIs for LinuxCNC, simulating a 5 axis CNC Milling Machine.

See the main LinuxCNC documentation for more details.

<https://linuxcnc.org/docs/html/gui/vismach.html>

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

mb2hal – HAL non–realtime component for Modbus

**SYNOPSIS**

Default component name

```
loadusr -W mb2hal config=config_file.ini
```

Custom component name

```
loadusr -Wn mymodule mb2hal config=config_file.ini
```

**DESCRIPTION**

MB2HAL is a generic non–realtime HAL component to communicate with one or more Modbus devices. It supports Modbus RTU and Modbus TCP.

See <https://linuxcnc.org/docs/html/drivers/mb2hal.html> for more information.

**PINS****fnc\_01\_read\_coils:**

**mb2hal.m.n.bit** bit out **mb2hal.m.n.bit–inv** bit out

**fnc\_02\_read\_discrete\_inputs:**

**mb2hal.m.n.bit** bit out **mb2hal.m.n.bit–inv** bit out

**fnc\_03\_read\_holding\_registers:**

**mb2hal.m.n.float** float out **mb2hal.m.n.int** s32 out

**fnc\_04\_read\_input\_registers:**

**mb2hal.m.n.float** float out **mb2hal.m.n.int** s32 out

**fnc\_05\_write\_single\_coil:**

**mb2hal.m.n.bit** bit in

NELEMENTS needs to be 1 or PIN\_NAMES must contain just one name.

**fnc\_06\_write\_single\_register:**

**mb2hal.m.n.float** float in

**mb2hal.m.n.int** s32 in

NELEMENTS needs to be 1 or PIN\_NAMES must contain just one name.

Both pin values are added and limited to 65535 (UINT16\_MAX). Use one and let the other open (read as 0).

**fnc\_15\_write\_multiple\_coils:**

**mb2hal.m.n.bit** bit in

**fnc\_16\_write\_multiple\_registers:**

**mb2hal.m.n.float** float in

**mb2hal.m.n.int** s32 in

Both pin values are added and limited to 65535 (UINT16\_MAX). Use one and let the other open (read as 0).

**Each transaction**

**mb2hal.m.num\_errors** u32 in

Error counter

m = HAL\_TX\_NAME or transaction number if not set n = element number (NELEMENTS)

Example:

mb2hal.00.01.int (TRANSACTION\_00, second register)

mb2hal.readStatus.01.bit (HAL\_TX\_NAME=readStatus, first bit)

**AUTHOR**

Victor Rocco

**LICENSE**

GPL

**NAME**

mdi – Send G–code commands from the terminal to the running LinuxCNC instance

**SYNOPSIS**

**mdi**

**DESCRIPTION**

**mdi** sends G–code commands to LinuxCNC. The command starts an environment in which G–code commands are sent to the interpreter and machine feedback is displayed.

**USAGE**

send a single command and exit:

```
mdi m2 s1400
```

interactive session

```
$mdi
MDI> m3 s1000
MDI> G0 X100
MDI> ^Z
$stopped
```

**SEE ALSO**

**linuxcnc(1)**

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

**mdro** – manual only Digital Read Out (DRO)

**SYNOPSIS**

**mdro** [-v] [-p *point\_size*] [-m] [-I *file.var*] [*axes*]

**DESCRIPTION**

**mdro** is a manual only DRO providing functionality similar to a traditional manual DRO. It is most useful for manual machines converted to CNC. It allows the user to manually control the machine while continuing to use the DRO scales on the axes. The GUI can be sized to match the user's screen. It is mouse-only and touchscreen friendly.

**OPTIONS**

These command line options are normally used when **mdro** is started in a HAL file. See below for the corresponding .ini file options.

**v**

Turn on verbose debug prints. **-vv** is even more verbose.

**p** *point\_size*

Set the point size for the text in the application. This option controls the overall size of the window on the screen. Default is 20. Typical values range from 20 to 30.

**m**

Set this if the DRO scales provide data scaled in millimeters.

**I** *file.var*

Load G54 through G57 coordinates from *file.var*.

*axes*

This option is used to specify the names of the axes handled by the program. The default is "XYZ". A four axis mill would use "XYZA", and a lathe with a two axis DRO might use "XZ".

**SCREEN CONFIGURATION**

The top of the screen includes a row for each axis specified in *axes*. Data in these rows are derived from signals on the *mdro.axis.n* pins that are instantiated when **mdro** is started. Each row includes buttons that allow the value to be zeroed, to be halved or a new value to be entered. There is also a button that enables the index zero process for that axis.

The screen includes buttons that allow the selection of one of four different coordinate systems. The machine coordinate system can also be selected though it cannot be changed.

The screen includes a keypad that can be used with a mouse or a touch screen to enter coordinate data.

Finally, buttons on the screen allow the selection of inch or mm data display.

**USAGE**

**mdro** is normally started from the *[DISPLAY]* entry in a dedicated **mdro.ini** file. The INI file and the associated HAL files should include the pins and signals that support the DRO scales. The HAL connections to **mdro** must be done in the *POSTGUI\_HALFILE* referenced in the INI file.

Other *[DISPLAY]* section options

*GEOMETRY* = *axes*

Names the coordinate axes used in the program. For example, "XYZ" for a 3 axis mill or "XZ" for a lathe, Default is "XYZ".

*MDRO\_VAR\_FILE* = *file.var*

Preload a VAR file. This is typically the VAR file used by the operational code.

*POINT\_SIZE* = *n*

This option sets the size of the font used which sets the overall size of the window. The default point size is 20, Typical sizes are 20 to 30.

*MM = 1*

Set this if the DRO scales provide data scaled in millimeters.

### EXAMPLE

Using an example of "XYZA" for an *axes* argument, these pins will be created when **mdro** starts:

```
mdro.axis.0 mdro.axis.1 mdro.axis.2 mdro.axis.3 mdro.index-enable.0 mdro.index-enable.1
mdro.index-enable.2 mdro.index-enable.3
```

In this example, the first row will be labeled "X" and will show the data associated with pin `mdro.axis.0`. In many configurations, `mdro.axis.0` can be connected directly to `x-pos-fb` in the `POSTGUI-HAL` file. The index pins should be connected to the corresponding `index-enable` pins from the DRO.

**mdro** can also be started via a "loadusr" command in a HAL file for a trial. Here's an example of a sim setup:

```
loadusr -W mdro -l sim.var XYZ net x-pos-fb â mdro.axis.0 net y-pos-fb â mdro.axis.1 net z-pos-fb â
mdro.axis.2
```

### AUTHOR

Robert Bond

### COPYRIGHT

Copyright © 2022 Robert Bond

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

melfagui – Vismach Virtual Machine GUI

**DESCRIPTION**

**melfagui** is one of the sample **Vismach** GUIs for LinuxCNC, simulating a Mitsubishi serial manipulator

See the main LinuxCNC documentation for more details.

<https://linuxcnc.org/docs/html/gui/vismach.html>

**SEE ALSO**

**linuxcnc(1)**

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>

**COPYRIGHT**

Copyright © 2023 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

milltask – Non–realtime task controller for LinuxCNC

**DESCRIPTION**

**milltask** is an internal process of LinuxCNC. It is generally not invoked directly but by an INI file setting: **[TASK]TASK=milltask**. The **milltask** process creates the ini.\\* HAL pins listed below and owned by the **inihal** component. These pins may be modified while LinuxCNC is running to alter values that are typically specified statically in an INI file.

The **inihal** pins are sampled in every task cycle, however, commands affected by their values typically use the value present at the time when the command is processed. Such commands include all codes handled by the interpreter (**G–code** programs and **MDI** commands) and NML **jogging** commands issued by a GUI (including **halui**). **Wheel jogging** is implemented in the realtime motion module so **inihal** pin changes (e.g., ini.\*.max\_velocity, ini.\*.max\_acceleration) may be honored as soon as altered values are propagated to the motion module.

**PINS****Per–joint pins (N == joint number)****ini.N.backlash**

Allows adjustment of [JOINT\_N]BACKLASH

**ini.N.ferror**

Allows adjustment of [JOINT\_N]FERROR

**ini.N.min\_ferror**

Allows adjustment of [JOINT\_N]MIN\_FERROR

**ini.N.min\_limit**

Allows adjustment of [JOINT\_N]MIN\_LIMIT

**ini.N.max\_limit**

Allows adjustment of [JOINT\_N]MAX\_LIMIT

**ini.N.max\_velocity**

Allows adjustment of [JOINT\_N]MAX\_VELOCITY

**ini.N.max\_acceleration**

Allows adjustment of [JOINT\_N]MAX\_ACCELERATION

**ini.N.home**

Allows adjustment of [JOINT\_N]HOME

**ini.N.home\_offset**

Allows adjustment of [JOINT\_N]HOME\_OFFSET

**ini.N.home\_offset**

Allows adjustment of [JOINT\_N]HOME\_SEQUENCE

**Per–axis pins (L == axis letter)****ini.L.min\_limit**

Allows adjustment of [AXIS\_L]MIN\_LIMIT

**ini.L.max\_limit**

Allows adjustment of [AXIS\_L]MAX\_LIMIT

**ini.L.max\_velocity**

Allows adjustment of [AXIS\_L]MAX\_VELOCITY

**ini.L.max\_acceleration**

Allows adjustment of [AXIS\_L]MAX\_ACCELERATION

**Global pins****ini.traj\_default\_acceleration**

Allows adjustment of [TRAJ]DEFAULT\_ACCELERATION



**ini.traj\_default\_velocity**

Allows adjustment of [TRAJ]DEFAULT\_VELOCITY

**ini.traj\_max\_acceleration**

Allows adjustment of [TRAJ]MAX\_ACCELERATION

**ini.traj\_max\_velocity**

Allows adjustment of [TRAJ]MAX\_VELOCITY

**Global pins (arc\_blend trajectory planner)****ini.traj\_arc\_blend\_enable**

Allows adjustment of [TRAJ]ARC\_BLEND\_ENABLE

**ini.traj\_arc\_blend\_fallback\_enable**

Allows adjustment of [TRAJ]ARC\_BLEND\_FALLBACK\_ENABLE

**ini.traj\_arc\_blend\_gap\_cycles**

Allows adjustment of [TRAJ]ARC\_OPTIMIZATION\_DEPTH

**ini.traj\_arc\_blend\_optimization\_depth**

Allows adjustment of [TRAJ]ARC\_BLEND\_GAP\_CYCLES

**ini.traj\_arc\_blend\_ramp\_freq**

Allows adjustment of [TRAJ]ARC\_BLEND\_RAMP\_FREQ

**NOTES**

The **inihal** pins cannot be linked or set in a HAL file that is specified by an INI file [HAL]HALFILE item because they are not created until **milltask** is started. The **inihal** pin values can be altered by independent halcmd programs specified by [APPLICATION]APP items or by GUIs that support a [HAL]POSTGUI\_HALFILE.

The INI file is not automatically updated with values altered by **inihal** pin settings but can be updated using the calibration program (emccalib.tcl) when using a [HAL]POSTGUI\_HALFILE.

**NAME**

millturn, millturngui – Vismach Virtual Machine GUI

**DESCRIPTION**

**millturngui** is one of the sample **Vismach** GUIs for LinuxCNC, simulating a Mill–Turn machine.

See the main LinuxCNC documentation for more details.

<https://linuxcnc.org/docs/html/gui/vismach.html>

**SEE ALSO**

**linuxcnc(1)**

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>

**COPYRIGHT**

Copyright © 2023 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

mitsub\_vfd – HAL non–realtime component for Mitsubishi A500 F500 E500 A500 D700 E700 F700–series VFDs (others may work) This uses the COMPUTER LINK protocol `_not_` MODBUS. The connection is made through the PU connector.

**SYNOPSIS**

**loadrt mitsub\_vfd** [**--baud** *baudrate*] [**--port** *devicename*] **name1=number1**[,**name2=number2**...]

**name1**

is user selectable (usually a description of the controlled device).

*number1*

is the slave number that was set on the VFD. Must be two digits (Parameter 117).

*nameN=numberN*

can be repeated for multiple VFD's connected together.

**--baud** *baudrate*

is optional as it defaults to 9600, all networked vfd's must be set to the same baudrate.

**--port** *devicename*

is optional as it defaults to `ttyS0`, a common alternative is `/dev/ttyUSB0`.

**DESCRIPTION**

The mitsub\_vfd component interfaces a Mitsubishi VFD to LinuxCNC. The VFD is connected via RS–485 serial to the computer's USB or serial port using a RS–232/RS–485 converter.

**HARDWARE SETUP**

reference manual *communication option reference manual* and A500 technical manual for 500 series. Fr–A700 F700 E700 D700 technical manual for the 700 series. The inverter must be set manually for communication (you may have to set PR 77 to 1 to unlock PR modification). You must power cycle the inverter for some of these, e.g. 79.

**VFD INTERNAL PARAMETERS:****PARAMETER 79**

1 or 0

**PARAMETER 117**

Station number – 1

(can be optionally set 0 – 31) if component is also set

**PARAMETER 118**

Communication speed 96

(can be optionally set 48, 96, 192 if component is also set)

**PARAMETER 119**

Stop bit/data length – 1

(8 bits, two stop – don't change)

**PARAMETER 120**

Parity – 0

(no parity – don't change)

**PARAMETER 121**

COM tries – 10

(if maximum 10 COM errors then inverter faults– can change.)

**PARAMETER 122**

COM check time interval 9999

(never check – if communication is lost inverter will not know (can change))

**PARAMETER 123**

Wait time – 9999

No wait time is added to the serial data frame (don't change).

**PARAMETER 124**

CR selection – 0

Don't change.

**PARAMETER 549**

Communication protocol – 0

Computer link protocol – don't change – (not all VFDs have this)

**NOTES**

This driver assumes certain other VFD settings:

- That the motor frequency status is set to show Hertz.
- That the status bit 3 is up to speed.
- That the status bit 7 is alarm.

Some models, e.g. the E500, cannot monitor status. You must set set the monitor pin to false. In this case pins such as up-to-speed, amps, alarm and status bits are not useful.

**PINS**

*VFD\_NAME.fwd* (bit, in)

Forward/reverse pin

*VFD\_NAME.run* (bit, in)

Run/stop pin

*VFD\_NAME.debug* (bit, in)

Set debug mode pin. This will print many messages to the terminal.

*VFD\_NAME.monitor* (bit, in)

Set monitor mode pin. If false, request-status command will not be sent to VFD. Status, amps, power, motor-feedback, and alarm would then not be useful.

*VFD\_NAME.estop* (bit, in)

Set E-stop mode pin. This will stop the VFD. Restarting requires the run pin to cycle.

*VFD\_NAME.fwd* (bit, out)

Up-to-speed status pin. Motor is at requested speed within VFD's settings tolerance.

*VFD\_NAME.alarm* (bit, out)

Alarm status pin

*VFD\_NAME.motor-cmd* (float, in)

The requested motor speed, in Hertz (Hz)

*VFD\_NAME.motor-fb* (float, out)

The motor feedback speed (from VFD) in Hertz (Hz)

*VFD\_NAME.motor-amps* (float, out)

The motor current, in amperes (A)

*VFD\_NAME.motor-power* (float, out)

The motor power

**VFD\_NAME.scale-cmd** (float, in)

Motor command's scale setting defaults to 1

**VFD\_NAME.scale-cmd** (float, in)

Motor command's scale setting defaults to 1

**VFD\_NAME.scale-cmd** (float, in)

Motor command's scale setting defaults to 1

**VFD\_NAME.stat-bit-0** (bit, out)

Raw status bit

**VFD\_NAME.stat-bit-1** (bit, out)

Raw status bit

**VFD\_NAME.stat-bit-2** (bit, out)

Raw status bit

**VFD\_NAME.stat-bit-3** (bit, out)

Raw status bit. Configure the VFD so that the function *Up to frequency* or *motor-at-speed* is assigned to status bit 3 (parameter 191 for 700 series).

**VFD\_NAME.stat-bit-4** (bit, out)

Raw status bit

**VFD\_NAME.stat-bit-5** (bit, out)

Raw status bit

**VFD\_NAME.stat-bit-6** (bit, out)

Raw status bit

**VFD\_NAME.stat-bit-7** (bit, out)

Raw status bit. Configure the VFD so that the function *alarm* is assigned to status bit 7 (parameter 195 for 700 series)

### SAMPLE HAL

```
loadusr -Wn coolant mitsub_vfd --port /dev/ttyUSB0 spindle=02 coolant=01
# ***** Spindle VFD setup slave 2 *****
net spindle-vel-cmd spindle.motor-cmd
net spindle-cw spindle.fwd
net spindle-on spindle.run
net spindle-at-speed spindle.up-to-speed
net estop-out spindle.estop
# cmd scaled to RPM (belt/gearbox driven)
setp spindle.scale-cmd .135
# feedback is in rpm (recipicale of command)
setp spindle.scale-fb 7.411
# turn on monitoring so feedback works
setp spindle.monitor 1
net spindle-speed-indicator spindle.motor-fb
# ***** Coolant VFD setup slave 1 *****
net coolant-flood coolant.run
net coolant-is-on coolant.up-to-speed
# cmd and feedback scaled to hertz
setp coolant.scale-cmd 1
setp coolant.scale-fb 1
# command full speed
setp coolant.motor-cmd 60
# allows us to see status
setp coolant.monitor 1
```

net estop-out coolant.estop

## ISSUES

Some models, e.g. E500, cannot monitor status, so set the monitor pin to false. In this case, pins such as up-to-speed, amps, alarm and status bits are not useful.

**NAME**

modcompile – Utility for compiling Modbus drivers

**DESCRIPTION**

**modcompile** is used to compile modbus drivers that use the Mesa card UARTs.

See the main LinuxCNC documentation for more details.

[https://linuxcnc.org/docs/stable/html/drivers/mesa\\_modbus.html](https://linuxcnc.org/docs/stable/html/drivers/mesa_modbus.html)

**SEE ALSO**

linuxcnc(1)\*

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>

**COPYRIGHT**

Copyright © 2023 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

monitor-xhc-hb04 – monitors the XHC-HB04 pendant and warns of disconnection

**SYNOPSIS**

**monitor-xhc-hb04**

**DESCRIPTION**

**monitor-xhc-hb04** is included to monitor disconnects and reconnects of the pendant. This script runs in the background and will pop up a message when the pendant is disconnected or reconnected.

Usage is optional; if used it is specified with INI file entry: ““

APP = monitor-xhc-hb04 ““

**SEE ALSO**

xhc-hb04(1), linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at [/usr/share/doc/LinuxCNC/](#).

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.



**NAME**

motion-logger – log motion commands sent from LinuxCNC's Task module

**SYNOPSIS**

**motion-logger**

**DESCRIPTION**

**motion-logger** is a test program to log motion commands sent from LinuxCNC's Task module to the Motion module.

It is largely used by the regression tests and is poorly documented.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

moveoff\_gui – a GUI for the moveoff component

**SYNOPSIS**

**moveoff\_gui** [**--help** | **--h** | \*-?]

**moveoff\_gui** [*options*]

**DESCRIPTION**

Moveoff\_gui is a sample graphical user interface (GUI) for controlling a HAL moveoff component to implement HAL-only offsets. See the manpage (man moveoff) for **IMPORTANT** limitations and warnings.

Supported configurations must use a known kinematics module with KINEMATICS\_TYPE = KINEMATICS\_IDENTITY. The modules currently supported are: **trivkins**

**OPTIONS**

**--help** | **-?** | **--h**

Show options and exit.

**-mode onpause** | always

onpause: pop up GUI to control offsets when program paused

always: show GUI to control offsets always

Default: **onpause**

**-axes** *axis-names*

Letters from set of {x y z a b c u v w}.

Examples: **-axes** x, **-axes** xyz, **-axes** xz (no spaces)

Default: **xyz**

**-inc** *incrementvalue*

Specify one increment value per **-inc** (up to 4).

Defaults: \* 0.001 0.01 0.10 1.0\*

**-size** *integer*

Overall gui size is based on font size, typically 8 – 20.

Default: **14**

**-loc** center | +x+y

Initial location on screen.

Examples: **-loc** center, **-loc** +20+100

Default: **center**

**-autoresume**

Resume program when move-enable deasserted.

Default: notused

**-delay** *delay secs*

Delay for autoresume (allow time to restore spindle speed etc).

Default: **5**

## OTHER OPTIONS

These options are available for special cases:

### **-noentry**

Disables creation of entry widgets.

Default: notused

### **-no\_resume\_inhibit**

Disable use of resume-inhibit to controlling gui.

Default: notused

### **-no\_pause\_requirement**

Disable check for halui.program.is-paused.

Default: notused

### **-no\_cancel\_autoresume**

Useful for retracting offsets with simple external controls.

Default: notused

### **-no\_display**

Use when both external controls and external displays are in use.

Default: notused

## NOTES

LinuxCNC must be running.

Halui must be loaded, typical INI file setting:

```
[HAL]
HALUI = halui
```

The moveoff component must be loaded with the name *mv* as: **loadrt moveoff names=*mv* personality=*\_number\_of\_axes***

If the pin *mv.motion-enable* is **not** connected when *moveoff\_gui* is started, **controls will be provided** to enable offsets and set offset values. If the pin **is** connected, **only a display** of offsets is shown and control must be made by **external** HAL connections.

If a pin named *\*.resume-inhibit* exists and is not connected, it will be set while offsets are applied. This pin may be provided by the controlling LinuxCNC GUI in use. Use of the pin may be disabled with the option **-no\_resume\_inhibit**.

The **-autoresume** option uses *halui.program.resume* to automatically resume program execution when the *move-enable* pin is deactivated and all offsets are removed. The resume pin is not activated until an additional interval (**-delay delay\_secs**) elapses. This delay interval may be useful for restarting related equipment (a spindle motor for example). While timing the delay, a popup is offered to cancel the automatic program resumption.

## USAGE

The INI file in the configuration directory must provide HALFILES to loadrt the moveoff component, connect its pins, and addf its read and write functions in the proper order. These steps can be done at runtime using an existing configuration INI file and specifying a system library HALFILE **hookup\_moveoff.tcl** as illustrated below:

```
[HAL]
HALUI = halui
HALFILE = user_halfile_1
etc ...
HALFILE = user_halfile_n
HALFILE = LIB:hookup_moveoff.tcl
```

The `hookup_moveoff.tcl` HAL file will use INI file settings for the moveoff component control pins:

```
[OFFSET]
EPSILON =
WAYPOINT_SAMPLE_SECS =
WAYPOINT_THRESHOLD =
BACKTRACK_ENABLE =
```

The `hookup_moveoff.tcl` will use INI file settings for the moveoff per-axis limits:

```
[AXIS_m]
OFFSET_MAX_VELOCITY =
OFFSET_MAX_ACCELERATION =
OFFSET_MAX_LIMIT =
OFFSET_MIN_LIMIT =
```

The `moveoff_gui` program should be specified in the APPLICATIONS stanza of the INI file, for example:

```
[APPLICATIONS]
DELAY = delay_in_secs_to_allow_hal_connections
APP = moveoff_gui -option1 -option2 ...
```

## SEE ALSO

Simulation configurations that demonstrate the `moveoff_gui` and the moveoff component are located in:

`configs/sim/axis/moveoff` (`axis-ui`) `configs/sim/touchy/ngcgui` (`touchy-ui`)

See also `moveoff(9)` for details on the component.

**NAME**

mqtt-publisher – send HAL pin data to MQTT broker periodically

**SYNOPSIS**

**loadusr -W mqtt-publisher** [*options*] **keys=pin1[,pin2...]**

**DESCRIPTION**

**mqtt-publisher** is a non-realtime program that reads HAL values periodically and passes the values to a MQTT broker.

When specifying the MQTT settings in the INI file, this is the recommended setup:

HAL file:

```
loadusr -W mqtt-publisher [MQTT]DRYRUN --mqtt-broker=[MQTT]BROKER \
--mqtt-user=[MQTT]USERNAME --mqtt-password=[MQTT]PASSWORD keys=halui.estop.is-activated
```

INI file:

```
[MQTT]
DRYRUN = --dryrun
BROKER = broker.local
USERNAME = username
PASSWORD = password
```

This component need the Paho python library installed to function. On debian this is available from the python3-paho-mqtt package.

**OPTIONS**

**keys=pin1[,pin2,...]**

The name of HAL pins, signals and other values to publish using MQTT. The names are also used as the JSON keys in the MQTT message published with the broker. If multiple "keys=" options are specified, the lists are merged.

**--dryrun**

Do not set up MQTT connection, only print message to stdout. Useful for debugging and testing.

**--mqtt-broker=FQDN**

The fully qualified DNS name of the MQTT broker. The default broker name is "localhost".

**--mqtt-port=PORTNUMBER**

The port to use of the MQTT broker. The default port is 1883.

**--mqtt-user=USERNAME**

The user name to use when connecting to the MQTT broker.

**--mqtt-password=PASSWORD**

The password to use when connecting to the MQTT broker.

**--mqtt-prefix=PREFIX**

The MQTT prefix/topic to use when publishing to the MQTT broker. The default prefix is "devices/linuxcnc/machine".

**FUNCTIONS**

**mqtt-publisher**

The loop reading HAL values and publishing MQTT messages.

**PINS**

**mqtt-publisher.enable** bit input

When TRUE, publish messages to MQTT broker. When FALSE, do not publish messages. Default is TRUE.

**mqtt-publisher.period** u32 input

The number of seconds to sleep between publishing MQTT messages to the broker. Default is 10 seconds.

**mqtt-publisher.lastpublish** u32 output

When the last MQTT publication was published in number of seconds since EPOCH. If no publication has taken place, the value is zero.

**EXAMPLE**

Any set of HAL pins and signals can be published. This setup might be a useful starting point:

```
loadusr -W mqtt-publisher \  
[MQTT]DRYRUN \  
--mqtt-broker=[MQTT]BROKER \  
--mqtt-user=[MQTT]USERNAME \  
--mqtt-password=[MQTT]PASSWORD \  
keys=halui.axis.a.pos-feedback,halui.axis.b.pos-feedback,\  
halui.axis.c.pos-feedback,halui.axis.u.pos-feedback,\  
halui.axis.v.pos-feedback,halui.axis.w.pos-feedback,\  
halui.axis.x.pos-feedback,halui.axis.y.pos-feedback,\  
halui.axis.z.pos-feedback,halui.estop.is-activated,\  
halui.joint.0.is-homed,halui.joint.1.is-homed,halui.joint.2.is-homed,\  
halui.joint.3.is-homed,halui.joint.4.is-homed,halui.joint.5.is-homed,\  
halui.joint.6.is-homed,halui.joint.7.is-homed,halui.joint.8.is-homed,\  
halui.machine.is-on,halui.max-velocity.value,halui.mode.is-auto,\  
halui.mode.is-manual,halui.mode.is-mdi,halui.mode.is-teleop,\  
halui.program.is-running
```

Note: It is recommended to use the line continuation character "\" as shown here to improve readability. But note that spaces must be left in to delimit options, and must not be included (including at the beginning of a line) inside a single option.

**SEE ALSO**

hal(3hal)

**AUTHOR**

Component and documentation created by Petter Reinholdtsen, as part of the LinuxCNC project.

**COPYRIGHT**

Copyright © 2023 Petter Reinholdtsen.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

ngcgui – a framework for conversational G-code generation on the controller

**SYNOPSIS**

**ngcgui**

**DESCRIPTION**

**ngcgui** details: <https://linuxcnc.org/docs/html/gui/ngcgui.html>

**SEE ALSO**

linuxcnc(1)\*

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at </usr/share/doc/LinuxCNC/>.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

panelui – interface buttons to LinuxCNC or HAL

**SYNOPSIS**

**panelui**

**DESCRIPTION**

**panelui** is a non–realtime component to interface buttons to LinuxCNC or HAL. It decodes MESA 7I73 style key–scan codes and calls the appropriate routine. It gets input from a realtime component – sampler. Sampler gets it’s input from either the MESA 7i73 or sim\_matrix\_kb component. Panelui is configurable using an INI style text file to define button types, HAL pin types, and/or commands.

Full documentation can be found in the HTML or PDF docs:  
<https://linuxcnc.org/docs/html/gui/panelui.html>

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at [usr/share/doc/LinuxCNC/](http://usr/share/doc/LinuxCNC/).

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.



**NAME**

pi500\_vfd – Powtran PI500 Modbus driver

**SYNOPSIS**

**pi500\_vfd**

**PINS**

**pi500-vfd.N.commanded-frequency** float in  
Frequency of vfd

**pi500-vfd.N.run** bit in  
run the vfd

**pi500-vfd.N.enable** bit in  
1 to enable the vfd. 0 will remote trip the vfd, thereby disabling it.

**pi500-vfd.N.is-running** bit out  
1 when running

**pi500-vfd.N.is-at-speed** bit out  
1 when running at assigned frequency

**pi500-vfd.N.is-ready** bit out  
1 when vfd is ready to run

**pi500-vfd.N.is-alarm** bit out  
1 when vfd alarm is set

**pi500-vfd.N.motor-current** float out  
Output current in amps

**pi500-vfd.N.heatsink-temp** float out  
Temperature of drive heatsink

**pi500-vfd.N.watchdog-out** bit out  
Alternates between 1 and 0 after every update cycle. Feed into a watchdog component to ensure vfd driver is communicating with the vfd properly.

**PARAMETERS**

**pi500-vfd.N.mbslaveaddr** u32 rw  
Modbus slave address

**LICENSE**

GPLv2 or greater

**NAME**

pmx485-test – Modbus communications testing with a Powermax Plasma Cutter

**SYNOPSIS**

**pmx485-test**

**DESCRIPTION**

pmx485 is a python script for testing communications with a Hypertherm Powermax plasma cutter using the Modbus ASCII protocol over RS485.

**USAGE**

Select the correct port from the list of available ports.

Check RS485 to establish communications.

Changing MODE, CURRENT, or PRESSURE will change the corresponding value on the Powermax and the new value will be reported back to the test panel.

Check PANEL to end communication.

**AUTHOR**

Phillip Carter & Gregory D Carl

**LICENSE**

GPL

**NAME**

pmx485 – Modbus communications with a Powermax Plasma Cutter.

**SYNOPSIS**

```
loadusr -Wn pmx485 pmx485 /dev/ttyUSB0
```

**DESCRIPTION**

pmx485 is a non–realtime HAL component to communicate with a Hypertherm Powermax plasma cutter using the Modbus ASCII protocol over RS485.

**SEE ALSO**

See the Drivers section of the LinuxCNC Documentation for more information on pmx485.

**AUTHOR**

Phillip Carter & Gregory D Carl

**LICENSE**

GPL

**NAME**

pncconf – configuration wizard for Mesa cards

**SYNOPSIS**

**pncconf**

**DESCRIPTION**

**pncconf** is used to configure systems that use Mesa cards.

Details: <https://linuxcnc.org/docs/html/config/pncconf.html>

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at [/usr/share/doc/LinuxCNC/](#).

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

puma560gui – Vismach Virtual Machine GUI

**DESCRIPTION**

**puma560gui** is one of the sample **Vismach** GUIs for LinuxCNC, simulating a Puma 560 robot arm.

See the main LinuxCNC documentation for more details.

<https://linuxcnc.org/docs/html/gui/vismach.html>

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

pumagui – Vismach Virtual Machine GUI

**DESCRIPTION**

**pumagui** is one of the sample **Vismach** GUIs for LinuxCNC, simulating a generic Puma style robot arm.

See the main LinuxCNC documentation for more details.

<https://linuxcnc.org/docs/html/gui/vismach.html>

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

pyngcgui – Python implementation of NGCGUI

**SYNOPSIS**

**pyngcgui**

**DESCRIPTION**

**pyngcgui** is an alternative implementation of NGCGUI: <https://linuxcnc.org/docs/html/gui/ngcgui.html>

**SEE ALSO**

ngcgui(1), linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

pyui – utility for panelui

**SYNOPSIS**

**loadusr pyui**

**DESCRIPTION**

**pyui** validates panelui.ini files.

This will read, try to correct, then save the panelui.ini file. It will print errors to the terminal if found.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at [/usr/share/doc/LinuxCNC/](#).

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.



**NAME**

pyvcp – Virtual Control Panel for LinuxCNC

**SYNOPSIS**

**pyvcp** [-g *WxH+X+Y*] [-c *component-name*] *myfile.xml*

**OPTIONS**

**-g** *WxH+X+Y*

This sets the initial geometry of the root window. Use *WxH* for just size, *+X+Y* for just position, or *WxH+X+Y* for both. Size / position use pixel units. Position is referenced from top left.

**-c** *component-name*

Use *component-name* as the HAL component name. If the component name is not specified, the basename of the XML file is used.

**SEE ALSO**

*Python Virtual Control Panel* in the LinuxCNC documentation for a description of the xml syntax, along with examples.

**NAME**

pyvcp\_demo – Python Virtual Control Panel demonstration component

**SYNOPSIS**

**pyvcp\_demo**

**DESCRIPTION**

**pyvcp\_demo** is mainly used by sample configurations.

**USAGE**

pyvcp\_demo filename1.xml filename2.hal [compname]

If not provided, use compname == pyvcp

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at /usr/share/doc/LinuxCNC/.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

qtplasmac-materials – Create a plasma materials file.

**SYNOPSIS**

**qtplasmac-materials**

**DESCRIPTION**

qtplasmac-materials is a Python script for creating a materials file.

The file can be created by text entry or by importing text from a CAM tool file.

**SEE ALSO**

See the QtPlasmaC section of the LinuxCNC Documentation for more information.

[https://linuxcnc.org/docs/devel/html/plasma/qtplasmac.html#\\_material\\_converter](https://linuxcnc.org/docs/devel/html/plasma/qtplasmac.html#_material_converter)

**AUTHOR**

Phillip Carter & Gregory D Carl

**LICENSE**

GPL

**NAME**

qtplasmac\_gcode – Python script shipping with Plasmac, a Plasma cutting system.

**DESCRIPTION**

**qtplasmac\_gcode** is used by qtplasmac and is not intended to be used standalone.

See the QtPlasmaC section of the LinuxCNC Documentation for more information.  
<https://linuxcnc.org/docs/devel/html/plasma/qtplasmac.html>

**SEE ALSO**

linuxcnc(1)\*

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at [/usr/share/doc/LinuxCNC/](#).

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>

**COPYRIGHT**

Copyright © 2023 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

qtvcp – Qt-based virtual control panels

**SYNOPSIS**

**qtvcp** [*OPTIONS*] myfile.ui

**DESCRIPTION**

**QtVCP** is a system for creating user interfaces for LinuxCNC.

Full documentation at <https://linuxcnc.org/docs/html/gui/qtvcp.html>

**OPTIONS**

**-h, --help**

Show this help message and exit.

**-c** [*<NAME>*]

Set component name to NAME. Default is basename of UI file.

**-a**

Set the window to always be on top.

**-d**

Enable debug output.

**-v**

Enable verbose debug output.

**-q**

Enable only error debug output.

**-g** [*<GEOMETRY>*]

Set geometry WIDTHxHEIGHT+XOFFSET+YOFFSET. Values are in pixel units, XOFFSET/YOFFSET is referenced from top left of screen. Use **-g** WIDTHxHEIGHT for just setting size or **-g** +XOFFSET+YOFFSET for just position.

Example: **-g** 200x400+0+100

**-H** [*<FILE>*]

Execute HAL statements from *FILE* with halcmd after the component is set up and ready.

**-i**

Enable info output.

**-m**

Force panel window to maximize.

**-f**

Force panel window to fullscreen.

**-t** [*<THEME>*]

Set Qt style. Default is system theme.

**-x** [*<XID>*]

Reparent QtVCP into an existing window XID instead of creating a new top level window.

**--push\_xid**

Reparent window into a plug add push the plug xid number to standardout.

**-u** [*<USERMOD>*]

File path of user defined handler file.

**-o** [*<USEROPTS>*]

Pass *USEROPTS* strings to handler under self.w.USEROPTIONS\_ list variable.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

rotarydelta – Vismach Virtual Machine GUI

**DESCRIPTION**

**rotarydelta** is one of the sample **Vismach** GUIs for LinuxCNC, simulating a delta robot with rotary actuators

See the main LinuxCNC documentation for more details. <https://linuxcnc.org/docs/html/gui/vismach.html>

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

rs274 – standalone G-code interpreter

**SYNOPSIS**

**rs274** [-p interp.so] [-t tool.tbl] [-v var-file.var] [-n 0|1|2] [-b] [-s] [-g] [ *input\_file* [ *output\_file* ] ]

**DESCRIPTION**

**rs274** Standalone G-code interpreter interface

Usage: rs274 [-p interp.so] [-t tool.tbl] [-v var-file.var] [-n 0|1|2] [-b] [-s] [-g] [input file [output file]]

**OPTIONS**

-p  
Specify the pluggable interpreter to use

-t  
Specify the .tbl (tool table) file to use

-v  
Specify the .var (parameter) file to use

-n  
Specify the continue mode:

0: continue

1: enter MDI mode

2: stop (default)

-b  
Toggle the *block delete* flag (default: OFF)

-s  
Toggle the *print stack* flag (default: OFF)

-g  
Toggle the *go (batch mode)* flag (default: OFF)

-i  
specify the .ini file (default: no ini file)

-T  
call task\_init()

-l  
specify the log\_level (default: -1)

**EXAMPLE**

To see the output of a loop for example we can run rs274 on the following file and see that the loop never ends. To break out of the loop use Ctrl Z. The following two files are needed to run the example.

FIXME: Some good soul please fix the whitespace for the examples below

test.ngc

```
#<test> = 123.352 o101 while [[#<test> MOD 60 ] NE 0]
(debug,#<test>) #<test> = [#<test> + 1] 101 endwhile M2
```

test.tbl

```
T1 P1 Z0.511 D0.125 ;1/8 end mill
```



T2 P2 Z0.1 D0.0625 ;1/16 end mill  
T3 P3 Z1.273 D0.201 ;#7 tap drill

command

```
rs274 -g test.ngc -t test.tbl
```

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

rtapi\_app – creates a simulated real time environment

**SYNOPSIS**

**rtapi\_app**

**DESCRIPTION**

**rtapi\_app** Creates a simulated real time environment.

Used for loading real time modules on systems without real time (for simulation).

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at </usr/share/doc/LinuxCNC/>.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

scaragui – Vismach Virtual Machine GUI

**DESCRIPTION**

**scaragui** is one of the sample **Vismach** GUIs for LinuxCNC, simulating a SCARA style robot arm.

See the main LinuxCNC documentation for more details.

<https://linuxcnc.org/docs/html/gui/vismach.html>

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

schedrmt – telnet based scheduler for LinuxCNC

**SYNOPSIS**

```
schedrmt {-- --port <port number> --name <server name> --connectpw <password> --enablepw
<password> --sessions <max sessions> --path <path> -ini <INI file>}
```

**DESCRIPTION**

With `--port`

Waits for socket connections (Telnet) on specified socket, without port uses default port 5007.

With `--name <server name>`

Sets the server name to specified name for Hello.

With `--connectpw <password>`

Sets the connection password to *password*. Default *EMC*.

With `--enablepw <password>`

Sets the enable password to *password*. Default *EMCTOO*.

With `--sessions <max sessions>`

Sets the maximum number of simultaneous connexctions to max sessions. Default is no limit (-1).

With `--path`

Sets the base path to program (G-Code) files, default is `"../nc_files/"`. Make sure to include the final slash (/).

With `-ini <INI file>`

Uses specified *INI file* instead of default *emc.ini*.

There are six commands supported, for which the commands set and get contain LinuxCNC-specific sub-commands based on the commands supported by emcsh, but where the "emc\_" prefix is omitted. Commands and most parameters are not case-sensitive. The exceptions are passwords, file paths and text strings.

The supported commands are as follows: `= HELLO =:: Hello <password> <client> <version>` If a valid password was entered the server will respond with `HELLO ACK <Server Name> <Server Version>`, here server name and server version are looked up from the implementation. If an invalid password or any other syntax error occurs then the server responds with: `HELLO NAK`. `= Get =::` The get command includes one of the emc sub-commands, described below and zero or more additional parameters. `= Set =::` The set command includes one of the emc sub-commands, described below and one or more additional parameters. `= Quit =::` The quit command disconnects the associated socket connection. `= Shutdown =::` The shutdown command tells EMC to shutdown before quitting the connection. This command may only be issued if the Hello has been successfully negotiated and the connection has control of the CNC (see enable sub-command below). This command has no parameters. `= Help =` The help command will return help information in text format over the telnet connection. If no parameters are specified, it will itemize the available commands. If a command is specified, it will provide usage information for the specified command. Help will respond regardless of whether a "Hello" has been successfully negotiated.

**EMC sub-commands:**

echo on | off

With get will return the current echo state, with set, sets the echo state. When echo is on, all commands will be echoed upon receipt. This state is local to each connection.

verbose on | off

With get will return the current verbose state, with set, sets the verbose state. When in verbose mode is on, all set commands return positive acknowledgement in the form `SET <COMMAND> ACK`. In addition, text error messages will be issued when in verbose mode. This state is local to each connection.

enable <pwd> | off

With get will return On or Off to indicate whether the current connection is enabled to perform control functions. With set and a valid password, the current connection is enabled for control functions. OFF may not be used as a password and disables control functions for this connection.

config

TBD

comm\_mode ascii | binary

With get, will return the current communications mode. With set, will set the communications mode to the specified mode. The binary protocol is TBD.

comm\_prot <version no>

With get, returns the current protocol version used by the server. With set, sets the server to use the specified protocol version, provided it is lower than or equal to the highest version number supported by the server implementation.

INIFILE

Returns the path and file name of the current configuration INI file.

plat

Returns the platform for which this was compiled, e.g., linux\_2\_0\_36

ini <var> <section>

Returns the string value of <var> in section <section>, in EMC\_INIFILE.

debug { <new value> }

With get, returns the integer value of EMC\_DEBUG, in the EMC. Note that it may not be true that the local EMC\_DEBUG variable here (in emcsh and the GUIs that use it) is the same as the EMC\_DEBUG value in the EMC. This can happen if the EMC is started from one INI file, and the GUI is started with another that has a different value for DEBUG. With set, sends a command to the EMC to set the new debug level, and sets the EMC\_DEBUG global here to the same value. This will make the two values the same, since they really ought to be the same.

QMode <mode> stop | run | pause | resume (Set only) | error (Get only)

With no arg, returns the program queue status as "stop", "run", "pause" or "error". Otherwise, sends a command to set the current mode to "stop", "run" or "pause".

QStatus <Queue Size> <First Tag Id> <Last Tag Id> <Queue CRC> (Get only)

Returns then number of programs in queue (Queue Size), the Tag id of the first program in the queue, the Tag id of the last program in the queue, and the CRC of all of the Tag Ids in the queue. The actual calculation of the CRC is not important, the purpose is to be able to compare the current CRC with the previous CRC. If they differ, then there has been a change to the size or order of the programs in queue.

AutoTagId <Start Id>

With get, returns the next autoincremented unique tag id to associate with a queue record. With set, resets auto tag generation to begin at the specified value.

PgmAdd <priority> <tag id> <x> <y> <z> <zone> <file name> <feed override> <spindle override> <tool>

With set, adds a program to the queue with priority of the program, a unique tag identifying the program, the x, y and z offsets or zone for the origin of the program, the path + file name, the feed and spindle overrides to apply, and the default tool to use. If tag id is zero, the tag id will be generated automatically. If zone is zero, then the x, y, and z offsets will be used, otherwise zones 1 to 9 correspond to G54 to G59.3 respectively.

PgmById <tag id> [priority] [tag id] [x] [y] [z] [zone] [file name] [feed override] [spindle override] [tool]

With get, returns the queue entry matching the specified tag id, including the priority, tag id, x, y, and z coordinates, the zone, file name, feed and spindle overrides and the default tool.

PgmByIndex <\_index\_> [priority] [tag id] [x] [y] [z] [zone] [file name] [feed override] [spindle override]

[*tool*]

With `get`, returns the queue entry matching the specified index into the queue, including the priority, tag id, x, y, and z coordinates, the zone, file name, feed and spindle overrides and the default tool.

`PgmAll`

With `get`, performs effectively a `PgmByIndex` for every entry in the queue. Each result will be returned in the form: "PGMBYINDEX ..." with `cr lf` at the end of each record.

`PriorityById <_tag id_> <_priority_>`

With `get`, returns the priority of the queue entry matching the specified tag. With `set`, changes the priority of the queue entry to the specified priority.

`PriorityByIndex <_tag id_> <_priority_>`

With `get`, returns the priority of the queue entry matching the specified index into the queue. With `set`, changes the priority of the queue entry to the specified priority.

`DeleteById <_tag id_>`

With `set`, deletes the queue entry matching the specified tag id.

`DeleteByIndex <_index_>`

With `set`, deletes the queue entry matching the specified index into the queue.

`PollRate <_rate_>`

With `set`, sets the rate at which the scheduler polls for information. The default is 1.0 or one second.

With `get`, returns the current poll rate.

## SEE ALSO

`linuxcnc(1)`

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

## BUGS

None known at this time.

## AUTHOR

This man page written by Andy Pugh, as part of the LinuxCNC project.

## REPORTING BUGS

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

## COPYRIGHT

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

scorbot-er-3 – to link the Intellitek Scorbot educational robot to LinuxCNC

**DESCRIPTION**

**scorbot-er-3** is a non-realtime component that interfaces the control box of a Scorbot ER-3 robot arm to the LinuxCNC HAL.

Joint 0

rotation around the base

Joint 1

shoulder

Joint 2

elbow

Joint 3

wrist (+ is wrist up & rotate hand)

Joint 4

wrist (+ is wrist down & rotate hand)

Joint 5

unused

Joint 6

unused

Joint 7

hand open/close (+ is close)

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

sendkeys – send input events based on pins or scancodes from HAL

**SYNOPSIS**

```
loadusr sendkeys config=s8t5, 16, t12
```

**DESCRIPTION**

This component is intended as a partner component to matrix\_kb or the hostmot2 7i73 driver. It accepts the key-up and key-down event codes from either of these and converts them to keystrokes sent from a virtual keyboard.

It also allows for keystrokes to be generated by individual HAL pins.

The **config** parameter to the **loadusr** HAL command defines how many scancodes will be supported and how many individual pins are created. **config=s16** would support the 16 scancodes of a 4x4 matrix. **config=t10** would create 10 individual HAL pin triggers. **config=s16t10** would create one instance with both the above.

Multiple configs separated by commas will create multiple instances of the component. The accepted codes can be seen in the extract from the linux headers here: <https://wiki.linuxcnc.org/cgi-bin/wiki.pl?Scancodes>

The component requires the user to have write permissions to /dev/uinput which is not available by default. To give access perform the following:

1. Create the uinput group and add the LinuxCNC user to it:

```
sudo groupadd -f uinput
sudo gpasswd -a username uinput
```

2. Create a new entry in .B/etc/udev/rules.d/99-input.rules

```
sudo echo KERNEL=="uinput", GROUP="uinput", MODE=="0660" | sudo tee /etc/udev/rules.d/88-input.rules
```

3. Reboot the machine. You can test that it has worked:

```
ls -l /dev/uinput
crw-rw---- 1 root uinput 10, 223 Nov 11 15:35 /dev/uinput
```

It is possible to link the 7i73 codes to both the matrix\_kb comp and this comp, so that some codes operate HAL pins and some send keystrokes. Where the option exists it is *MUCH* better to use HAL pins for things like jogging and machine control. This component should really be used only for text entry and GUI operations.

Each key on the matrix is allocated a scan code. The simplest way to configure the component is to load the component and open a halmeter showing sendkeys.0.current-event. Note the code for each physical key. (If keys do not give consistent results then you probably need to toggle the value of the matrix\_kb.0.negative-logic pin and/or invert io pins).

Then edit the HAL file to assign a key event to each scancode. For example:

```
setp sendkeys.0.scan-event-21 34
```

To set a button to type the letter "G". The key events related to each physical key need to be set up prior to the component activating, but after the component is loaded.

To achieve this there is a pin **sendkeys.N.init** which should be set to "true" once the events to be sent for each scancode and pin have been set up.



To generate keystrokes from other sources note that a keydown is simply 0xC0 & keycode and keyup is 0x80 & keycode.

## PINS

**sendkeys.N.keycode** u32 in

Connect to scancode generator.

**sendkeys.N.current-event** s32 out

Shows the current scancode without keyup / keydown markers.

**sendkeys.N.init** bit in

Set this pin TRUE once all the event parameters have been set.

## PARAMETERS

**sendkeys.N.scan-event-MM** u32 in

Assign the uinput event codes associated with each scancode.

**sendkeys.N.pin-event-MM** u32 in

Assign the uinput codes associated with each HAL bit pin.

## EXAMPLE

```
loadusr -W sendkeys config=16t2
net scancodes hm2_7i73.0.0.keycode => sendkeys.0.keycode
```

```
setp sendkeys.0.scan-event-00 34 # Key G
setp sendkeys.0.scan-event-01 2 # Key 1
setp sendkeys.0.scan-event-02 3 # Key 2
setp sendkeys.0.scan-event-03 4 # Key 3
setp sendkeys.0.scan-event-04 50 # Key M
setp sendkeys.0.scan-event-05 05 # Key 4
setp sendkeys.0.scan-event-06 06 # Key 5
setp sendkeys.0.scan-event-07 07 # Key 6
setp sendkeys.0.scan-event-08 31 # Key S
setp sendkeys.0.scan-event-09 8 # Key 7
setp sendkeys.0.scan-event-10 9 # Key 8
setp sendkeys.0.scan-event-11 10 # Key 9
setp sendkeys.0.scan-event-12 20 # Key T
setp sendkeys.0.scan-event-13 11 # Key 0
setp sendkeys.0.scan-event-14 52 # Key Dot
setp sendkeys.0.scan-event-15 14 # Backspace
setp sendkeys.0.pin-event-00 29 # Left Ctrl
setp sendkeys.0.pin-event-01 57 # Space
setp sendkeys.0.init 1

#Send Ctl + Space from one trigger
net clear-errors parport.0.pin.00.in sendkeys.0.trigger-00 sendkeys.0.trigger-01
```

## AUTHOR

Andy Pugh

## LICENSE

GPL-2.0+

**NAME**

setup\_designer – A script to configure the system for use of QTdesigner

**DESCRIPTION**

**setup\_designer** Run this script to prepare your system for custom GUI creatioun using the QT toolset.

See the QTVCP documentation for more details.

**SEE ALSO**

linuxcnc(1)\*

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>

**COPYRIGHT**

Copyright © 2023 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

shuttle – control HAL pins with the ShuttleXpress, ShuttlePRO, and ShuttlePRO2 device made by Contour Design

**SYNOPSIS**

**loadusr shuttle** [*DEVICE* ...]

**DESCRIPTION**

Shuttle is a non–realtime HAL component that interfaces Contour Design’s ShuttleXpress, ShuttlePRO, and ShuttlePRO2 devices with LinuxCNC’s HAL.

If the driver is started without command–line arguments, it will probe all /dev/hidraw\* device files for Shuttle devices, and use all devices found. If it is started with command–line arguments, it will only probe the devices specified.

The ShuttleXpress has five momentary buttons, a 10 counts/revolution jog wheel with detents, and a 15–position spring–loaded outer wheel that returns to center when released.

The ShuttlePRO has 13 momentary buttons, a 10 counts/revolution jog wheel with detents, and a 15–position spring–loaded outer wheel that returns to center when released.

The ShuttlePRO2 has 15 momentary buttons, a 10 counts/revolution jog wheel with detents, and a 15–position spring–loaded outer wheel that returns to center when released.

**UDEV**

The shuttle driver needs read permission to the Shuttle devices' /dev/hidraw\* device files. This can be accomplished by adding a file **/etc/udev/rules.d/99–shuttle.rules**, with the following contents:

```
SUBSYSTEM=="hidraw", ATTRS{idVendor}=="0b33",
ATTRS{idProduct}=="0020", MODE="0444"
```

```
SUBSYSTEM=="hidraw", ATTRS{idVendor}=="05f3",
ATTRS{idProduct}=="0240", MODE="0444"
```

```
SUBSYSTEM=="hidraw", ATTRS{idVendor}=="0b33",
ATTRS{idProduct}=="0030", MODE="0444"
```

The LinuxCNC Debian package installs an appropriate udev file automatically, but if you are building LinuxCNC from source and are not using the Debian packaging, you’ll need to install this file by hand. If you install the file by hand you’ll need to tell udev to reload its rules files by running `udevadm control --reload–rules`.

**A WARNING ABOUT THE JOG WHEEL**

The Shuttle devices have an internal 8–bit counter for the current jog–wheel position. The shuttle driver can not know this value until the Shuttle device sends its first event. When the first event comes into the driver, the driver uses the device’s reported jog–wheel position to initialize counts to 0. This means that if the first event is generated by a jog–wheel move, that first move will be lost.

Any user interaction with the Shuttle device will generate an event, informing the driver of the jog–wheel position. So if you (for example) push one of the buttons at startup, the jog–wheel will work fine and notice the first click.

**PINS**

All HAL pin names are prefixed with shuttle followed by the index of the device (the order in which the driver found them), for example shuttle.0 or shuttle.2.

(*prefix*).*button–(number)* (bit out), (*prefix*).*button–(number)–not* (bit out)

The momentary buttons. "*number*" identifies which button corresponds to the HAL pin. The

"button-(number)" pins are True when the button is pushed, the "button-(number)-not" pins are True when the button is not pushed.

*(prefix).counts* (s32 out)

Accumulated counts from the jog wheel (the inner wheel).

*(prefix).spring-wheel-s32* (s32 out)

The current deflection of the spring-wheel (the outer wheel). It's 0 at rest, and ranges from -7 at the counter-clockwise extreme to +7 at the clockwise extreme.

*(prefix).spring-wheel-f* (float out)

The current deflection of the spring-wheel (the outer wheel). It's 0.0 at rest, -1.0 at the counter-clockwise extreme, and +1.0 at the clockwise extreme. (The Shuttle devices report the spring-wheel position as an integer from -7 to +7, so this pin reports only 15 discrete values in its range.)

**NAME**

sim-torch – A simulated plasma torch

**SYNOPSIS**

**loadusr Wn sim-torch sim-torch**

**DESCRIPTION**

A simulated plasma torch for arc-ok testing.

*VERSION:* 0.1

**PINS**

**sim-torch-rt.cut-noise-in** float in (default: 0.75)

the maximum amount of noise during cutting (volts)

**sim-torch-rt.cycles-in** s32 in\*

the number of cycles that the arc voltage overshoots the cut voltage (cycles) (default: 200)

**sim-torch-rt.on-delay-in** s32 in\*

the time from turn on until overshoot begins (cycles) (default: 10)

**sim-torch-rt.offset-in** float in

the cut voltage offset(volts)

**sim-torch-rt.overshoot-in** s32 in

the percentage of the cut voltage that the arc voltage overshoots (percent) (default: 50)

**sim-torch-rt.ramp-noise-in** float in (default: 5)\*

the maximum amount of noise during overshoot (volts)

**sim-torch-rt.ramp-up-in** s32 in (default: 80)

percent of *cycles\_in* that the arc voltage ramps up (percent)

**sim-torch-rt.start** bit in

start the arc

**sim-torch-rt.voltage-in** float in

the cut voltage (volts) (default: 100)

**sim-torch-rt.voltage-out** float out

output voltage (volts)

**AUTHOR**

Phillip A Carter & Gregory D Carl

**LICENSE**

GPLv2 or greater

**NAME**

`sim_pin` – GUI for displaying and setting one or more HAL inputs

**SYNOPSIS**

**\*sim\_pin** [*Options*] *name1* [*name2* [*name3* ...]]\*

*Options*: **--help** (shows help text) **--title** *title\_string*

For bit items, the name may include a `/mode=` specifier: *namei*\*/mode=[\***pulse** | **toggle** | **hold**] (default is **toggle**)

**DESCRIPTION**

HAL boolean items (bit) and numerical items (u32, s32, float) are supported.

If the named input is a numerical type, the GUI displays:

**Entry** Entry widget for value or a valid Tcl expression. **Set** Pushbutton to set new value from Entry (or use <RETURN>) **Reset** Pushbutton to reset to the value present on initiation If the input is a **bit** type, the GUI shows a single pushbutton that is controlled by radio-button selectors:

mode=**pulse** Pulse input to 1 for each pushbutton press mode=**toggle** Toggle input for each pushbutton press mode=**hold** Set input to 1 while pushbutton pressed

If the bit item mode begins with an uppercase letter, the radio buttons for selecting other modes are not shown

**NOTES**

LinuxCNC or a standalone HAL application must be running

A named item can specify a **pin**, **param**, or **signal**. The named item must be writable:

**pin** IN or I/O (and not connected to a signal with a writer) **param** RW **signal** connected to a writable **pin**

**USAGE**

**sim\_pin** can be used interactively from a shell command line or started automatically from a configuration INI file.

**EXAMPLE**

Example for INI file usage:

```
[APPLICATIONS] DELAY = 5 APP = sim_pin \ halui.machine.off/mode=pulse \
ini.traj_arc_blend_enable \ motion-command-handler-tmax
```

**NAME**

simulate\_probe – simulate a probe input

**SYNOPSIS**

**simulate\_probe**

**DESCRIPTION**

**simulate\_probe** Creates an on–screen GUI button to simulate touch probe input. Typically used in sim configs or debugging.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at </usr/share/doc/LinuxCNC/>.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

stepconf – A configuration wizard for parallel–port based machines.

**SYNOPSIS**

**stepconf**

**DESCRIPTION**

**stepconf** aids in the configuration of machines using the parallel port interface.

Detailed docs: <https://linuxcnc.org/docs/html/config/stepconf.html>

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.



**NAME**

svd-ps\_vfd – HAL non-realtime component for SVD-P(S) VFDs

**SYNOPSIS**

**svd-ps\_vfd** [*OPTIONS*]

**DESCRIPTION**

The svd-ps\_vfd component interfaces a Soyas Power SVD-P(S) VFD to the LinuxCNC HAL. The VFD is connected via RS-485 to the LinuxCNC computer.

The SVD-P(S) VFDs are also sold under the LAPOND brand.

**HARDWARE SETUP**

The SVD-P(S) VFDs do not come with a Modbus daughterboard by default, it needs to be purchased separately.

**FIRMWARE SETUP**

The sad-ps\_vfd component uses standard Modbus protocol communication, which requires that one parameter be changed from the default settings:

PD-05 = 1 (Standard Modbus protocol)

The following settings have been tested successfully and are the default per Soyas documentation:

PD-00 = 6005 (9600 baud)

PD-01 = 0 (8N2)

PD-02 = 1 (Slave address)

PD-03 = 2 (Response delay)

PD-04 = 0 (Communication timeout)

PD-06 = 0 (Current resolution)

**OPTIONS**

**-b, --bits** *N*

For Modbus communication, set number of data bits to *N*. *N* must be between 5 and 8 inclusive.

(default 8) **-p, --parity** [Even,Odd,None] For Modbus communication, set serial parity to Even, Odd, or None. (default None)

**-r, --rate** *N*

For Modbus communication, set baud rate to *N*. It is an error if the rate is not one of the following: 1200, 2400, 4800, 9600, 19200, 38400 (default 9600)

**-s, --stopbits** [1,2]

For Modbus communication set serial stop bits to 1 or 2. (default 2)

**-t, --target** *N*

For Modbus communication, set Modbus target (slave) number. This must match the device number you set on the Huanyang GT VFD. (default 1)

**-d, --device** *PATH*

For Modbus communication, set the name of the serial device node to use. (default /dev/ttyS0)

**-v, --verbose**

Turn on verbose mode.

**-S, --motor-max-speed** *RPM*

The motor's max speed in RPM.

**-F, --max-frequency** *HZ*

This is the maximum output frequency of the VFD in Hz.

**-f, --min-frequency** *HZ*

This is the minimum output frequency of the VFD in Hz.

## PINS

**svd-ps\_vfd.period** (float, in)

The period for the driver's update cycle, in seconds. This is how frequently the driver will wake up, check its HAL pins, and communicate with the VFD. Must be between 0.001 and 2.000 seconds.

Default: 0.1 seconds.

**svd-ps\_vfd.speed-cmd** (float, in)

The requested motor speed, in RPM.

**svd-ps\_vfd.speed-fb** (float, out)

The motor's current speed, in RPM, reported by the VFD.

**svd-ps\_vfd.at-speed** (bit, out)

True when the drive is on and at the commanded speed (within 2%), False otherwise.

**svd-ps\_vfd.freq-cmd** (float, out)

The requested output frequency, in Hz. This is set from the .speed-cmd value, and is just shown for debugging purposes.

**svd-ps\_vfd.freq-fb** (float, out)

The current output frequency of the VFD, in Hz. This is reported from the VFD to the driver.

**svd-ps\_vfd.spindle-on** (bit, in)

Set this pin True to command the spindle on, at the speed requested on the .speed-cmd pin. Set this pin False to command the spindle off.

**svd-ps\_vfd.output-voltage** (float, out)

The voltage that the VFD is currently providing to the motor, in Volts.

**svd-ps\_vfd.output-current** (float, out)

The current that the motor is currently drawing from the VFD, in Amperes.

**hsvd-ps\_vfd.output-power** (float, out)

The power that the motor is currently drawing from the VFD, in Watts.

**svd-ps\_vfd.dc-bus-voltage** (float, out)

The current voltage of the VFD's internal DC power supply, in Volts.

**svd-ps\_vfd.modbus-errors** (u32, out)

A count of the number of modbus communication errors between the driver and the VFD. The driver is resilient against communication errors, but a large or growing number here indicates a problem that should be investigated.

**svd-ps\_vfd.input-terminal** (float, out)

The VFD's input terminal register.

**svd-ps\_vfd.AI1** (float, out)

The VFD's AI1 register.

**svd-ps\_vfd.AI2** (float, out)

The VFD's AI2 register.

## AUTHOR

Tinic Uro

## LICENSE

GPL-2.0+

**NAME**

teach-in – jog the machine to a position, and record the state

**SYNOPSIS**

**teach-in** [*> outfile*]

**DESCRIPTION**

**teach-in** is a script to learn set positions for later use by a script.

A dialog box is shown with options to choose the coordinate system. Each press of the "Learn" button outputs a line of text to stdout or the file chosen at load time.

Line format: line-no X Y Z flood mist spindle

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at [usr/share/doc/LinuxCNC/](http://usr/share/doc/LinuxCNC/).

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

thermistor – compute temperature indicated by a thermistor

**SYNOPSIS**

**thermistor**

**DESCRIPTION**

This component computes the temperature indicated by a thermistor in a voltage-divider ladder. It uses the Beta-parameter variant of the Steinhart-Hart equation, described here:

<http://en.wikipedia.org/wiki/Thermistor>

**PINS**

**thermistor.N.t0-c** float in

Reference temperature of the thermistor, in degrees Celsius (typically 25 C). This must be set before the component can compute the thermistor temperature. The reference temperature information is supplied by the thermistor manufacturer.

**thermistor.N.r0** float in

Resistance of the thermistor at the reference temperature. This must be set before the component can compute the thermistor temperature. The reference resistance information is supplied by the thermistor manufacturer.

**thermistor.N.beta** float in

Beta parameter of the thermistor (sometimes just called B). This must be set before the component can compute the thermistor temperature. The Beta parameter is supplied by the thermistor manufacturer.

**thermistor.N.r-other** float in

Resistance of the other resistor in the voltage-divider ladder. This must be set before the component can compute the thermistor temperature.

**thermistor.N.v-total** float in

Supply voltage of the voltage-divider ladder.

**thermistor.N.v-thermistor** float in

Voltage drop across the thermistor.

**thermistor.N.temperature-c** float out

Temperature sensed by the thermistor, in degrees Celsius.

**thermistor.N.temperature-k** float out

Temperature sensed by the thermistor, in Kelvins.

**thermistor.N.temperature-f** float out

Temperature sensed by the thermistor, in degrees Fahrenheit.

**thermistor.N.resistance** float out

Computed resistance of the thermistor.

**LICENSE**

GPL

**NAME**

tool\_mmap\_read – A component of the tool database system (an alternative to the classic tooltable)

**DESCRIPTION**

**tool\_mmap\_read** is not intended to be invoked by the user.

**SEE ALSO**

linuxcnc(1)\*

See the Tool Database Interface section of the LinuxCNC Documentation for more information at <https://linuxcnc.org/docs/stable/html/tooldatabase/tooldatabase.html>.

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at [/usr/share/doc/LinuxCNC/](#).

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2023 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

tool\_watch – A component of the tool database system (an alternative to the classic tooltable)

**DESCRIPTION**

**tool\_watch** is not intended to be invoked by the user.

See the Tool Database Interface section of the LinuxCNC Documentation for more information.  
<https://linuxcnc.org/docs/stable/html/tooldatabase/tooldatabase.html>

**SEE ALSO**

linuxcnc(1)\*

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at [/usr/share/doc/LinuxCNC/](#).

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>

**COPYRIGHT**

Copyright © 2023 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

tooledit – tool table editor

**SYNOPSIS**

**tooledit**

**DESCRIPTION**

**tooledit** a graphical tool table editor

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

touchy – TOUCHY LinuxCNC Graphical User Interface

**SYNOPSIS**

**touchy** **-ini** *<INI file>*

**DESCRIPTION**

**touchy** is one of the Graphical User Interfaces (GUI) for LinuxCNC. It gets run by the runscrip usually.

**OPTIONS****INI file**

The INI file is the main piece of an LinuxCNC configuration. It is not the entire configuration; there are various other files that go with it (NML files, HAL files, TBL files, VAR files). It is, however, the most important one, because it is the file that holds the configuration together. It can adjust a lot of parameters itself, but it also tells **LinuxCNC** which other files to load and use.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is **NO** warranty; not even for **MERCHANTABILITY** or **FITNESS FOR A PARTICULAR PURPOSE**.



**NAME**

update\_ini – converts 2.7 format INI files to 2.8 format

**SYNOPSIS**

**update\_ini** [-f] [-d] <INI file>\_

**DESCRIPTION**

**update\_ini** is run automatically by the "linuxcnc" script when an INI file in the pre-joints-axes format is opened.

-d causes a dialog box to be shown asking if the script should be run.

-f is designed for auto-conversion and will not create the backup files.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at [usr/share/doc/LinuxCNC/](http://usr/share/doc/LinuxCNC/).

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

vfdb\_vfd – HAL non–realtime component for Delta VFD–B Variable Frequency Drives

**SYNOPSIS**

**vfdb\_vfd** [OPTIONS]

**DESCRIPTION**

This manual page explains the **vfdb\_vfd** component. This component reads and writes to the VFD–B device via a Modbus connection.

**vfdb\_vfd** is for use with LinuxCNC.

**QUICK START**

The VFD–B ships in a configuration that can not talk to this driver. The VFD–B must be reconfigured via the face plate by the integrator before it will work. This section gives a brief description of what changes need to be made, consult your Delta VFD–B manual for more details.

Switch the VFD–B to Modbus RTU frame format

Switch parameter 09–04 from the factory default of 0 (Ascii framing) to 3, 4, or 5 (RTU framing). The setting you choose will determine several serial parameters in addition to the Modbus framing protocol.

Set the frequency control source to be Modbus, not the keypad

Switch parameter 02–00 from factory default of 00 (keypad control) to 5 (control from RS–485).

Set the run/stop control source to be Modbus, not the keypad

Switch parameter 02–01 from the factory default of 0 (control from keypad) to 3 (control from Modbus, with Stop enabled on the keypad).

**OPTIONS**

**-n** **--name** <halname>

set the HAL component name

**-d** **--debug**

Turn on debugging messages. Also toggled by sending a USR1 signal to the vfdb\_vfd process.

**-m** **--modbus–debug**

Turn on Modbus debugging messages. This will cause all Modbus messages to be printed in hex on the terminal. Also toggled by sending a USR2 signal to the vfdb\_vfd process.

**-I** **--ini** <INI file>

take configuration from this *INI file*. Defaults to environment variable INI\_FILE\_NAME. Most vfdb\_vfd configuration comes from the INI file, not from command–line arguments.

**-S** **--section** <section name>

take configuration from this section in the INI file. Defaults to *VFD–B*.

**-r** **--report–device**

report device propertiers on console at startup

**INI CONFIG VARIABLES****DEBUG**

Set to a non–zero value to enable general debug output from the VFD–B driver. Optional.

**MODBUS\_DEBUG**

Set to a non–zero value to enable modbus debug output from the VFD–B driver. Optional.

**DEVICE**

Serial port device file to use for Modbus communication with the VFD–B. Defaults to */dev/ttyS0*.

**BAUD**

Modbus baud rate. Defaults to 19200.

**BITS**

Modbus data bits. Defaults to 8.

**PARITY**

Modbus parity. Defaults to Even. Accepts *Even*, *Odd*, or *None*.

**STOPBITS**

Modbus stop bits. Defaults to 1.

**TARGET**

Modbus target number of the VFD-B to speak to. Defaults to 1.

**POLLCYCLES**

Only read the less important variables from the VFD-B once in this many poll cycles. Defaults to 10.

**RECONNECT\_DELAY**

If the connection to the VFD-B is broken, wait this many seconds before reconnecting. Defaults to 1.

**MOTOR\_HZ, MOTOR\_RPM**

The frequency of the motor (in Hz) and the corresponding speed of the motor (in RPM). This information is provided by the motor manufacturer, and is generally printed on the motor's name plate.

**PINS**

<name>.at-speed (bit, out)

True when drive is at commanded speed (see *speed-tolerance* below)

<name>.enable (bit, in)

Enable the VFD. If False, all operating parameters are still read but control is released and panel control is enabled (subject to VFD setup).

<name>.frequency-command (float, out)

Current target frequency in HZ as set through speed-command (which is in RPM), from the VFD.

<name>.frequency-out (float, out)

Current output frequency of the VFD.

<name>.inverter-load-percentage (float, out)

Current load report from VFD.

<name>.is-e-stopped (bit, out)

The VFD is in emergency stop status (blinking "E" on panel).

<name>.is-stopped (bit, out)

True when the VFD reports 0 Hz output.

<name>.jog-mode (bit, in)

1 for ON and 0 for OFF, enables the VFD-B *jog mode*. Speed control is disabled. This might be useful for spindle orientation.

<name>.max-rpm (float, out)

Actual RPM limit based on maximum frequency the VFD may generate, and the motors nameplate values. For instance, if *nameplate-HZ* is 50, and *nameplate-RPM* is 1410, but the VFD may generate up to 80Hz, then *max-rpm* would read as 2256 (80\*1410/50). The frequency limit is read from the VFD at startup. To increase the upper frequency limit, the UL and FH parameters must be changed on the panel. See the VFD-B manual for instructions how to set the maximum frequency.

<name>.modbus-ok (bit, out)

True when the Modbus session is successfully established and the last 10 transactions returned without error.

<name>.motor-RPM (float, out)

Estimated current RPM value, from the VFD.

<name>.motor-RPS (float, out)

Estimated current RPS value, from the VFD.

<name>.output-voltage (float, out)

From the VFD.

- <name>.output-current (float, out)  
From the VFD.
- <name>.speed-command (float, in)  
Speed sent to VFD in RPM. It is an error to send a speed faster than the Motor Max RPM as set in the VFD.
- <name>.spindle-on (bit, in)  
1 for ON and 0 for OFF sent to VFD, only on when running.
- <name>.max-speed (bit, in)  
Ignore the loop-time parameter and run Modbus at maximum speed, at the expense of higher CPU usage. Suggested use during spindle positioning.
- <name>.status (s32, out)  
Drive Status of the VFD (see the VFD manual). A bitmap.
- <name>.error-count (s32, out)  
Total number of transactions returning a Modbus error.
- <name>.error-code (s32, out)  
Most recent Error Code from VFD.
- <name>.frequency-limit (float, out)  
Upper limit read from VFD setup.

## PARAMETERS

- <name>.loop-time (float, RW)  
How often the Modbus is polled (default interval 0.1 seconds).
- <name>.nameplate-HZ (float, RW)  
Nameplate Hz of motor (default 50). Used to calculate target frequency (together with *nameplate-RPM*) for a target RPM value as given by speed-command.
- <name>.nameplate-RPM (float, RW)  
Nameplate RPM of motor (default 1410)
- <name>.rpm-limit (float, RW)  
Do-not-exceed soft limit for motor RPM (defaults to *nameplate-RPM*).
- <name>.tolerance (float, RW)  
Speed tolerance (default 0.01) for determining whether spindle is at speed (0.01 meaning: output frequency is within 1% of target frequency).

## USAGE

The `vfdb_vfd` driver takes precedence over panel control while it is enabled (see `.enable` pin), effectively disabling the panel. Clearing the `.enable` pin re-enables the panel. Pins and parameters can still be set, but will not be written to the VFD until the `.enable` pin is set. Operating parameters are still read while bus control is disabled.

Exiting the `vfdb_vfd` driver in a controlled way will release the VFD from the bus and restore panel control.

See the LinuxCNC Integrators Manual for more information. For a detailed register description of the Delta VFD-B, see the VFD manual.

## AUTHOR

Yishin Li; based on `vfd11_vfd` by Michael Haberler.

## LICENSE

GPL

**NAME**

vfs11\_vfd – HAL non–realtime component for Toshiba–Schneider VF–S11 Variable Frequency Drives

**SYNOPSIS**

**vfs11\_vfd** [OPTIONS]

**DESCRIPTION**

This manual page explains the **vfs11\_vfd** component. This component reads and writes to the vfs11 via a Modbus connection.

**vfs11\_vfd** is for use with LinuxCNC.

**OPTIONS**

- n --name** <halname>  
set the HAL component name
- d --debug**  
Turn on debugging messages. Also toggled by sending a USR1 signal to the vfs11\_vfd process.
- m --modbus–debug**  
Turn on Modbus debugging messages. This will cause all Modbus messages to be printed in hex on the terminal. Also toggled by sending a USR2 signal to the vfs11\_vfd process.
- I --ini** <INI file>  
takes configuration from this INI file. Defaults to environment variable INI\_FILE\_NAME.
- S --section** <section name>  
take configuration from this section in the INI file. Defaults to *VFS11*.
- r --report–device**  
Reports device properties on console at startup.

**PINS**

- <name>.acceleration–pattern (bit, in)  
when true, set acceleration and deceleration times as defined in registers F500 and F501 respectively. Used in PID loops to choose shorter ramp times to avoid oscillation.
- <name>.alarm–code (s32, out)  
non–zero if drive is in alarmed state. Bitmap describing alarm information (see register FC91 description). Use *err–reset* (see below) to clear the alarm.
- <name>.at–speed (bit, out)  
when drive is at commanded speed (see *speed–tolerance* below)
- <name>.current–load–percentage (float, out)  
Reported from the VFD.
- <name>.dc–brake (bit, in)  
Engage the DC brake. Also turns off spindle–on.
- <name>.enable (bit, in)  
Enable the VFD. If false, all operating parameters are still read but control is released and panel control is enabled (subject to VFD setup).
- <name>.err–reset (bit, in)  
Reset errors (alarms a.k.a Trip and e–stop status). Resetting the VFD may cause a 2–second delay until it’s rebooted and Modbus is up again.
- <name>.estop (bit, in)  
Put the VFD into emergency–stopped status. No operation possible until cleared with *err–reset* or powercycling.
- <name>.frequency–command (float, out)  
Current target frequency in Hz as set through speed–command (which is in RPM), from the VFD.

- <name>.frequency-out (float, out)  
Current output frequency of the VFD.
- <name>.inverter-load-percentage (float, out)  
Current load report from VFD.
- <name>.is-e-stopped (bit, out)  
The VFD is in emergency stop status (blinking "E" on panel). Use *err-reset* to reboot the VFD and clear the e-stop status.
- <name>.is-stopped (bit, out)  
True when the VFD reports 0 Hz output
- <name>.jog-mode (bit, in)  
1 for ON and 0 for OFF, enables the VF-S11 *jog mode*. Speed control is disabled, and the output frequency is determined by register F262 (preset to 5 Hz). This might be useful for spindle orientation.
- <name>.max-rpm (float, R)  
Actual RPM limit based on maximum frequency the VFD may generate, and the motors nameplate values. For instance, if *nameplate-HZ* is 50, and *nameplate-RPM\_* is 1410, but the VFD may generate up to 80Hz, then *max-rpm* would read as 2256 (80\*1410/50). The frequency limit is read from the VFD at startup. To increase the upper frequency limit, the UL and FH parameters must be changed on the panel. See the VF-S11 manual for instructions how to set the maximum frequency.
- <name>.modbus-ok (bit, out)  
True when the Modbus session is successfully established and the last 10 transactions returned without error.
- <name>.motor-RPM (float, out)  
Estimated current RPM value, from the VFD.
- <name>.output-current-percentage (float, out)  
from the VFD
- <name>.output-voltage-percentage (float, out)  
from the VFD
- <name>.output-voltage (float, out)  
from the VFD
- <name>.speed-command (float, in)  
Speed sent to VFD in RPM. It is an error to send a speed faster than the Motor Max RPM as set in the VFD.
- <name>.spindle-fwd (bit, in)  
1 for FWD and 0 for REV, sent to VFD.
- <name>.spindle-on (bit, in)  
1 for ON and 0 for OFF sent to VFD, only on when running.
- <name>.spindle-rev (bit, in)  
1 for ON and 0 for OFF, only on when running.
- <name>.max-speed (bit, in)  
Ignore the loop-time parameter and run Modbus at maximum speed, at the expense of higher CPU usage. Suggested use during spindle positioning.
- <name>.status (s32, out)  
Drive Status of the VFD (see the TOSVERT VF-S11 Communications Function Instruction Manual, register FD01). A bitmap.
- <name>.trip-code (s32, out)  
Trip code if VF-S11 is in tripped state.
- <name>.error-count (s32, RW)

Total number of transactions returning a Modbus error.

## PARAMETERS

<name>.frequency-limit (float, RO)

Upper limit read from VFD setup.

<name>.loop-time (float, RW)

How often the Modbus is polled (default interval 0.1 seconds)

<name>.nameplate-HZ (float, RW)

Nameplate Hz of motor (default 50). Used to calculate target frequency (together with *nameplate-RPM*) for a target RPM value as given by speed-command.

<name>.nameplate-RPM (float, RW)

Nameplate RPM of motor (default 1410)

<name>.rpm-limit (float, RW)

Do-not-exceed soft limit for motor RPM (defaults to *nameplate-RPM*).

<name>.tolerance (float, RW)

Speed tolerance (default 0.01) for determining whether spindle is at speed (0.01 meaning: output frequency is within 1% of target frequency)

## USAGE

The vfs11\_vfd driver takes precedence over panel control while it is enabled (see `_.enable` pin), effectively disabling the panel. Clearing the `_.enable` pin re-enables the panel. Pins and parameters can still be set, but will not be written to the VFD until the `_.enable` pin is set. Operating parameters are still read while bus control is disabled.

Exiting the vfs11\_vfd driver in a controlled will release the VFD from the bus and restore panel control.

See the LinuxCNC Integrators Manual for more information. For a detailed register description of the Toshiba VFD's, see the "TOSVERT VF-S11 Communications Function Instruction Manual" (Toshiba document number E6581222) and the "TOSVERT VF-S11 Instruction manual" (Toshiba document number E6581158).

## AUTHOR

Michael Haberler; based on gs2\_vfd by Steve Padnos and John Thornton.

## LICENSE

GPL

**NAME**

wj200\_vfd – Hitachi wj200 modbus driver

**SYNOPSIS**

**wj200\_vfd**

**PINS**

**wj200-vfd.N.commanded-frequency** float in  
Frequency of vfd (scaled to RPM)

**wj200-vfd.N.motor-frequency** float out  
Motor's actual frequency (scaled to RPM)

**wj200-vfd.N.motor-reverse** bit out  
1 when actually turning in reverse

**wj200-vfd.N.reverse** bit in  
1 when reverse 0 when forward

**wj200-vfd.N.run** bit in  
run the vfd

**wj200-vfd.N.enable** bit in  
1 to enable the vfd. 0 will remote trip the vfd, thereby disabling it.

**wj200-vfd.N.is-running** bit out  
1 when running

**wj200-vfd.N.is-at-speed** bit out  
1 when running at assigned frequency

**wj200-vfd.N.is-stopped** bit out  
1 when stopped

**wj200-vfd.N.is-ready** bit out  
1 when vfd is ready to run

**wj200-vfd.N.is-alarm** bit out  
1 when vfd alarm is set

**wj200-vfd.N.motor-current** float out  
Output current in amps

**wj200-vfd.N.heatsink-temp** float out  
Temperature of drive heatsink

**wj200-vfd.N.watchdog-out** bit out  
Alternates between 1 and 0 after every update cycle. Feed into a watchdog component to ensure vfd driver is communicating with the vfd properly.

**PARAMETERS**

**wj200-vfd.N.mbslaveaddr** u32 rw  
Modbus slave address

**wj200-vfd.N.frequency-scale** float rw (default: 1)  
RPM / frequency, default 1

**LICENSE**

GPLv2 or greater



**NAME**

xhc-hb04-accel - Obsolete script for jogging wheel

**SYNOPSIS**

**xhc-hb04-accel**

**DESCRIPTION**

**xhc-hb04-accel** Obsolete script, xhc-hb04.tcl now controls reduced wheel jogging accels.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

xhc-hb04 – HAL non-realtime component for the xhc-hb04 pendant.

**DESCRIPTION**

The xhc-hb04 component supports a common USB pendant that provides a number of pushbuttons, a manual pulse generator (mpg or jog wheel), and a selector switch for the wheel.

There are at least two hardware versions — one with 16 buttons and a more common one with 18 buttons. The information herein is based on the 18 button device with a USB Vendor:Product code of 10CE:EB70.

In addition to buttons, the pendant provides an LCD display for the current stepsize multiplier (from a set of available integer values), position (absolute and relative, labeled MC and WC respectively), feedrate (override percent and value in units per minute), and spindle speed (override percent and value in revolutions per minute (RPM)). The display is managed by a rotary switch that selects one of four axes for wheel positioning, feed override, spindle override, or OFF.

The pendant display, its rotary selector switch, and the component pin names use designators x,y,z,a. While this arrangement presumes a machine configured as XYZA, the pins can be assigned independently as required in a HAL configuration.

**UDEV**

The xhc-hb04 executable needs permission for reading the pendant's USB device. Debian package installs (debs) handle this automatically but Run-In-Place (RIP) builds may need a udev rules file. This file should be created (using sudo and a text editor) as:

/etc/udev/rules.d/99-xhc-hb04.rules with the single line:

```
ATTR{idProduct}=="eb70", ATTR{idVendor}=="10ce", MODE="0666", OWNER="root", GROUP="plugdev"
```

**STANDALONE USAGE**

The xhc-hb04 program can be run from the command line without LinuxCNC to test a pendant in a simulation mode. This standalone mode is used to identify the button codes produced for each button press and to verify proper counting of the jog wheel. The identified button codes can be used to create a **button-cfg-file**. When a **button-cfg-file** exists, pendant operation can be verified using the **-I** option to specify the file.

Usage:

```
$ xhc-hb04 [options]
```

**OPTIONS**

**-h**

list command line options and exit

**-I button-cfg-file**

see below for file format

**-H**

run in real-time HAL mode (simulation mode is default)

**-x**

wait for pendant detection before creating HAL pins.

**-s n**

n is one of the following stepsize sequences

1: 1,10,100,1000 (default) 2: 1,5,10,20 3: 1,10,100 4: 1,5,10,20,50,100 5: 1,10,50,100,1000

The stepsize selected is always multiplied by 0.001.

## BUTTON-CFG-FILE FORMAT

Standard configuration files are provided in the distribution for known button configurations:

```
/usr/share/linuxcnc/hallib/xhc-hb04-layout1.cfg
/usr/share/linuxcnc/hallib/xhc-hb04-layout2.cfg
```

or for a RIP build:

```
rip_base_dir/lib/hallib/xhc-hb04-layout1.cfg
rip_base_dir/lib/hallib/xhc-hb04-layout2.cfg
```

layout1 describes the 16 button pendant, layout2 describes the more common 18 button pendant.

The button configuration file follows the same format as INI files but should use a file suffix of .cfg.

File format:

```
[XHC-HB04]
BUTTON=X1:button-thename1
BUTTON=X2:button-thename2
BUTTON=X3:button-thename3
etc.
```

XN is the code reported for a button press and button-thenameN is the name to be assigned to the pin created for the button.

## HAL USAGE

Use the `-H` option to specify HAL mode and other options as required:

```
loadusr -W xhc-hb04 -H [options]
```

Example: `loadusr -W xhc-hb04-H -I path_to_cfg_file -s 2`

## INPUT PINS (CONTROL)

`_xhc-hb04.stepsize-up` (bit in)

A 1 pulse on this pin changes the stepsize to the next higher stepsize in the stepsize sequence specified in the `xhc-hb04` (`loadusr`) command.

`xhc-hb04.stepsize-down` (bit in)

A 1 pulse on this pin changes the stepsize to the next lower stepsize in the stepsize sequence specified in the `xhc-hb04` (`loadusr`) command.

## INPUT PINS (TO THE PENDANT LCD DISPLAY)

`xhc-hb04.[xyza].pos-absolute` (float in)

Absolute position display. The LCD display for `pos-absolute` is fixed format with a sign, 4 number digits and 3 fraction digits (+XXXX.XXX), require:  $-9999.999 \hat{=} \text{value} \hat{=} 9999.999$  (typically connect to: `halui.axis.N.pos-feedback`).

`xhc-hb04.[xyza].pos-relative` (float in)

Relative position display (typically connect to: `halui.axis.N.pos-relative`). The LCD display for `pos-relative` is fixed format with a sign, 4 number digits and 3 fraction digits (+XXXX.XXX), require:  $-9999.999 \hat{=} \text{value} \hat{=} 9999.999$ .

`xhc-hb04.feed-override` (float in)

Feed-override value. The float value is converted to a 16 bit integer and multiplied by 100 in order to display as percent, require:  $0 \hat{=} \text{pinvalue} \hat{=} 655$  (typically connect to: `halui.feed-override.value`).

`xhc-hb04.feed-value` (float in)

Current Feed-value (units/sec). The float value is converted to a 16 bit integer and multiplied by 60 in order to display as units-per-minute, require:  $0 \hat{=} \text{pinvalue} \hat{=} 1092$  (65520 units-per-minute) (typically connect to: `motion.current-vel`).

xhc-hb04.spindle-override (float in)

Spindle-override value. The float value is converted to a 16 bit integer and multiplied by 100 in order to display as percent, require: 0 ≤ pinvalue ≤ 655 (typically connect to: halui.spindle-override.value).

xhc-hb04.spindle-rps (float in)

Spindle speed in RPS (revolutions per second). The float value is converted to a 16 bit integer and multiplied by 60 in order to display as RPMs, require: 0 ≤ pinvalue ≤ 1092 (65520 RPM) (typically connect to: spindle.N.speed-out-rps-abs).

xhc-hb04.inch-icon (bit in)

Use inch icon (default is mm):

## OUTPUT PINS (STATUS)

xhc-hb04.sleeping (bit out)

True when the driver receives a pendant inactive (sleeping) message.

xhc-hb04.jog.enable-off (bit out)

True when the pendant rotary selector switch is in the OFF position or when the pendant is sleeping.

xhc-hb04.enable-[xyza] (bit out)

True when the pendant rotary selector switch is in the [xyza] position and not sleeping.

xhc-hb04.enable-spindle-override (bit out)

True when the pendant rotary selector switch is in the Spindle position and not sleeping (typically connect to: halui.spindle-override-count-enable).

xhc-hb04.enable-feed-override (bit out)

True when the pendant rotary selector switch is in the feed position and not sleeping (typically connect to: halui.feed-override-count-enable).

xhc-hb04.connected (bit out)

True when connection to the pendant is established over the USB interface.

xhc-hb04.require\_pendant (bit out)

True if driver started with the -x option.

xhc-hb04.stepsize (s32 out)

Current stepsize in the stepsize sequence as controlled by the stepsize-up and/or stepsize-down pins.

## OUTPUT PINS (FOR JOGGING USING AXIS.N.JOG-COUNTS)

xhc-hb04.jog.counts (s32 out)

Number of counts of the wheel since start-up (50 counts per wheel revolution) (typically connect to axis.N.jog-counts (lowpass filtering may be helpful)).

xhc-hb04.jog.counts-neg (s32 out)

The value of the xhc-hb04.jog.counts multiplied by -1.

xhc-hb04.jog.scale (float out)

Value is the current stepsize multiplied by 0.001 (typically connect to axis.N.jog-scale).

## EXPERIMENTAL: PINS FOR HALUI PLUS/MINUS JOGGING.

These pins provide some support for non-trivkins, world mode jogging.

xhc-hb04.jog.max-velocity (float in)

Connect to halui.max-velocity.value.

xhc-hb04.jog.velocity (float out)

Connect to halui.jog-speed.

xhc-hb04.jog.plus-[xyza] (bit out)

Connect to halui.jog.N.plus.

xhc-hb04.jog.minus-[xyza] (bit out)

Connect to halui.jog.N.minus.

xhc-hb04.jog.increment (float out)

Debug pin — `abs(delta_pos)`.

## BUTTON OUTPUT PINS (FOR THE 18 BUTTON, LAYOUT2 PENDANT)

The output bit type pins are TRUE when the button is pressed.

### ROW 1

(bit out) `xhc-hb04.button-reset`  
 (bit out) `xhc-hb04.button-stop`

### ROW 2

(bit out) `xhc-hb04.button-goto-zero`  
 (bit out) `xhc-hb04.button-rewind`  
 (bit out) `xhc-hb04.button-start-pause`  
 (bit out) `xhc-hb04.button-probe-z`

### ROW 3

(bit out) `xhc-hb04.button-spindle`  
 (bit out) `xhc-hb04.button-half`  
 (bit out) `xhc-hb04.button-zero`  
 (bit out) `xhc-hb04.button-safe-z`

### ROW 4

(bit out) `xhc-hb04.button-home`  
 (bit out) `xhc-hb04.button-macro-1`  
 (bit out) `xhc-hb04.button-macro-2`  
 (bit out) `xhc-hb04.button-macro-3`

### ROW 5

(bit out) `xhc-hb04.button-step`  
 (bit out) `xhc-hb04.button-mode`  
 (bit out) `xhc-hb04.button-macro-6`  
 (bit out) `xhc-hb04.button-macro-7`

## SYNTHESIZED BUTTON PINS

Additional buttons are synthesized for buttons named **zero**, **goto-zero**, and **half**. These synthesized buttons are active when the button is pressed AND the selector-switch is set to the corresponding axis [xyza].

(bit out) `xhc-hb04.button-zero-[xyza]`  
 (bit out) `xhc-hb04.button-goto-zero-[xyza]`  
 (bit out) `xhc-hb04.button-half-[xyza]`

## DEBUGGING

For debugging USB activity, use environmental variable `LIBUSB_DEBUG`:

```
export LIBUSB_DEBUG=[2 | 3 | 4]; xhc-hb04 [options]
    2:warning, 3:info, 4:debug
```

## SIM CONFIGS

The distribution includes several simulation configurations in the directory:

`/usr/share/doc/linuxcnc/examples/sample-configs/sim/axis/xhc-hb04/`  
 or for a RIP build:  
`rip_base_dir/configs/sim/axis/xhc-hb04/`

These configurations use a distribution-provided script (`xhc-hb04.tcl`) to configure the pendant and make necessary HAL connections according to a number of INI file settings. The script uses an additional HAL component (`xhc_hb04_util`) to provide common functionality and includes support for a standard method

for the start-pause button.

The settings available include: 1) specify button-cfg-file for standard layout1 or layout2 2) select axes (up to 4 axes from set of x y z a b c u v w) 3) implement per-axis filtering coefficients 4) implement per-axis acceleration for mpg jogging 5) implement per-axis scale settings 6) select normal or velocity based jog modes 7) select stepsize sequence 8) option to initialize pin for inch or mm display icon 9) option to require pendant on startup

The sim configs illustrate button connections that: 1) connect pendant stepsize-up button to the step input pin. 2) connect buttons to halui.\* pins 3) connect buttons to motion.\* pins

Another script is included to monitor the pendant and report loss of USB connectivity. See the README and .txt files in the above directory for usage.

The sim configs use the AXIS GUI but the scripts are available with any HAL configuration or GUI. The same scripts can be used to adapt the xhc-hb04 to existing configurations provided that the halui, motion, and axis.N pins needed are not otherwise claimed. Instructions are included in README file in the directory named above.

Use halcmd to display the pins and signals used by the xhc-hb04.tcl script:

```
halcmd show pin xhc-hb04    (show all xhc-hb04 pins)
halcmd show pin pendant_util (show all pendant_util pins)
halcmd show sig pendant:   (show all pendant signals)
```

## **AUTHOR**

Frederick Rible (fribler@teaser.fr)

**NAME**

xhc-whb04b-6 – Non-realtime jog dial HAL component for the wireless XHC WHB04B-6 USB device.

**SYNOPSIS**

```
xhc-whb04b-6 [-h] | [-H] [OPTIONS]
```

**DESCRIPTION**

The xhc-whb04b-6 HAL component supports the XHC WHB04B-6, a 6-axis wireless USB pendant. It provides a number of push-buttons, a jogwheel, two rotary buttons for axis and speed / step selection and an ordinary LCD display.

The LCD display, having a very simple firmware interface, indicates the following listed information only. No other information, such as custom data, can be printed.

- Activated axis (X, Y, Z, A, B or C)
- Current axis position of X, Y, Z and separately of A, B, C.
- Whether machine (X, Y, Z, A, B or C) or relative (X1, Y1, Z1, A1, B1 or C1) coordinates are displayed.
- Step size or velocity depending on the operating mode (MPG or Step or Continuous).
- Feedrate override
- Spindle Feedrate override
- Machine state such as reset.
- Battery level
- Wireless signal strength

The pendant display, its rotary selector switch, and the component pin names use designators x, y, z, a, b and c. While this arrangement presumes a machine configured as X, Y, Z, A, B and C, the pins can be assigned independently as required in a HAL configuration.

**OPTIONS****-h, --help**

Prints the synopsis and the most commonly used commands.

**-H**

Run xhc-whb04b-6 in HAL-mode instead of interactive mode. When in HAL mode commands from device will be exposed to HAL's shared memory. Interactive mode is useful for testing device connectivity and debugging.

**-s**

Lead + jogwheel changes the spindle override speed. Each tick will increase/decrease the spindle override.

**-f**

MPG + jogwheel changes the feed override. Each tick will increment/decrement the feed override.

**-B**

Add 5 mm and 10 mm to Step feedrate output

**-t**

Wait with timeout for USB device then proceed, exit otherwise. Without -t the timeout is implicitly infinite.

**-u, -U**

Show received data from device. With -U received and transmitted data will be printed. Output is prefixed with "usb".

**-p**

Show HAL pins and HAL related messages. Output is prefixed with "hal".

- e** Show captured events such as button pressed/released, jog dial, axis rotary button, and feed rotary button event. Output is prefixed with "event".
- a** Enable all logging facilities without explicitly specifying each.
- c** Enable checksum output which is necessary for debugging the checksum generator function. Do not rely on this feature since it will be removed once the generator is implemented.
- n** Force being silent and not printing any output except of errors. This will also inhibit messages prefixed with "init".

## UDEV

The xhc-whb04b-6 executable needs permission for reading the pendant's USB device. There may be the need for additional udev rules. If so, this file /etc/udev/rules.d/99-xhc-whb04b-6.rules should be created with the single line `ATTR{idProduct}=="eb93", ATTR{idVendor}=="10ce", MODE="0666", OWNER="root", GROUP="plugdev".`

## STANDALONE USAGE

The xhc-whb04b-6 program can be run from the command line without LinuxCNC to test a pendant. This standalone mode is used to identify the button codes produced for each button press and debug transmitted USB data.

### EXAMPLES

xhc-whb04b-6 -ue

Start in simulation mode and prints incoming USB data transfer and generated key pressed/released events.

xhc-whb04b-6 -p

Start in simulation mode and prints HAL pin names and events distributed to HAL memory.

xhc-whb04b-6 -H

Start in HAL mode (Normal mode for real machine use).

xhc-whb04b-6 -HsfB

Start in HAL mode + Spindle Override + Feedrate Override + Big step (5/10 mm).

## HAL USAGE

Use the -H option to specify HAL mode and other options as required: `loadusr -W xhc-whb04b-6 -HsfB`

### Input/Output Signals

Note: For each button an output pin is provided even if no functionality is realized with that signal. For example, to stop a running program the Stop button pin may be directly connected to `halui.program.stop`. However, to start/pause/resume a program, the corresponding button toggles besides `whb.button.start-pause` also the `whb.halui.program.{"run,pause,resume"}` signals accordingly.

Note: The Spindle+/Spindle- buttons do manipulate the spindle override. The spindle speed is set with the respective combos `Fn + Spindle-` and `FN + Spindle+`.

The following tables list all in-/output pins and state which signals they are meant to be connected to.

### Axis and Steppen

Signals utilized for moving axis.

<N> ... denotes the axis number, which is of {x, y, z, a, b, c}.

`whb.halui.home-all` (bit,out)

connect to `halui.home-all`, driven by M-Home. Pin for requesting all axis to home. See also



- whb.button.m-home.
- whb.halui.axis.\_<N>\_.select (bit,out)  
connect to halui.axis.\_<N>\_.select. Pin to select axis.
- whb.axis.\_<N>\_.jog-counts (s32,out)  
connect to axis.\_<N>\_.jog-counts. The count pin of the jogwheel.
- whb.axis.\_<N>\_.jog-enable (bit,out)  
connect to axis.\_<N>\_.jog-enable. If true (and in manual mode), any change to "jog-counts" will result in motion. If false, "jog-counts" is ignored.
- whb.axis.\_<N>\_.jog-scale (float,out)  
connect to axis.\_<N>\_.jog-scale'. The distance to move for each count on "jog-counts", in machine units.
- whb.axis.\_<N>\_.jog-vel-mode (bit,out)  
connect to axis.\_<N>\_.jog-jog-vel-mode'. If false the jogwheel operates in position mode. The axis will move exactly jog-scale units for each count, regardless of how long that might take. If true, the jogwheel operates in velocity mode – motion stops when the wheel stops, even if that means the commanded motion is not completed.
- whb.halui.max-velocity.value (float,in)  
connect to halui.max-velocity.value. The maximum allowable velocity, in units per second (<N> is two digit 0-padded).
- whb.halui.feed-override.scale (float,in)  
connect to halui.feed-override.scale. The scaling for feed override value.
- whb.halui.axis.\_<N>\_.pos-feedback' (float,in)  
connect to halui.axis.\_<N>\_.pos-feedback'. Feedback axis position in machine coordinates to be displayed.
- whb.halui.axis.\_<N>\_.pos-relative (float,in)  
connect to halui.axis.\_<N>\_.pos-relative'. Commanded axis position in relative coordinates to be displayed.

## Machine

Signals utilized for toggling machine status.

- whb.halui.machine.on (bit,out)  
Connect to halui.machine.on. Pin for requesting machine on.
- whb.halui.machine.is-on (bit,in)  
Connect to halui.machine.is-on. Pin that indicates machine is on.
- whb.halui.machine.off (bit,out)  
Connect to halui.machine.off. Pin for requesting machine off.

## Spindle

**Signals utilized for operating a spindle.**

- whb.halui.spindle.start (bit,out)  
Connect to halui.spindle.0.start. Pin to start the spindle.
- whb.halui.spindle.is-on (bit,in)  
Connect to halui.spindle.0.on. Pin to indicate spindle is on (either direction).
- whb.halui.spindle.stop (bit,out)  
Connect to halui.spindle.0.stop. Pin to stop the spindle.
- whb.halui.spindle.forward (bit,out)  
Connect to halui.spindle.0.forward. Pin to make the spindle go forward.

whb.halui.spindle.reverse (bit,out)

Connect to halui.spindle.0.reverse. Pin to make the spindle go reverse.

whb.halui.spindle.decrease (bit,out)

Connect to halui.spindle.0.decrease. Pin to decrease the spindle speed.

whb.halui.spindle.increase (bit,out)

Connect to halui.spindle.0.increase. Pin to increase the spindle speed.

whb.halui.spindle-override.increase (bit,out)

Connect to halui.spindle.0.override.increase. Pin for increasing the spindle override by the amount of scale.

whb.halui.spindle-override.decrease (bit,out)

Connect to halui.spindle.0.override.decrease. Pin for decreasing the spindle override by the amount of scale.

whb.halui.spindle-override.value (float,in)

Connect to halui.spindle.0.override.value. The current spindle override value.

whb.halui.spindle-override.scale (float,in)

Connect to halui.spindle.0.override.scale. The current spindle scaling override value.

## Feed

Signals utilized for operating spindle and feed override. The feed rotary button can serve in

- Continuous move  $x\%$  from max velocity
- Step move  $x$  mm
- MPG override feed/spindle
- The special position Lead.

**Continuous:** In this mode jogging is performed at the selected feed rate. As long the jogwheel turns, the selected axis moves.

**Step:** In this mode the machine moves  $\text{steps} * \text{wheel\_counts}$  at the currently selected step size and the current set feed rate in machine units. If the commanded position is not reached the machine keeps moving even the jogwheel is not turning.

**Lead:** Manipulates the spindle override.

**MPG:** Manipulates the feedrate override.

Note: As a consequence of 3 modes from manufacturer, switching the feed rotary button back from Lead revert to MPG mode, MPG mode is default mode at startup. Depending on the mode before turning the rotary button, the feed override results in different values. In MPG/CON the feed rate will change to 100%, 60%, ... and so forth. In Step mode the feed rate is specified in mm.

whb.halui.feed-override.value (float,in)

Connect to halui.feed-override.value. The current feed override value.

whb.halui.feed-override.decrease (bit,out)

Connect to halui.feed-override.decrease. Pin for decreasing the feed override by amount of scale.

whb.halui.feed-override.increase (bit,out)

Connect to halui.feed-override.increase. Pin for increasing the feed override by amount of scale.

whb.halui.feed-override.scale (float,out)

Connect to halui.feed-override.scale. Pin for setting the scale on changing the feed override.

whb.halui.max-velocity.value (float,out)  
Connect to halui.max-velocity.value.

## Program

Signals for operating program and MDI mode.

whb.halui.program.run (bit,out)  
Connect to halui.program.run in for running a program.

whb.halui.program.is-running (bit,in)  
Connect to halui.program.is-running in indicating a program is running.

whb.halui.program.pause (bit,out)  
Connect to halui.program.pause. Pin for pausing a program.

whb.halui.program.is-paused (bit,in)  
Connect to halui.program.is-paused. Pin indicating a program is pausing.

whb.halui.program.resume (bit,out)  
Connect to halui.program.resume. Pin for resuming a program.

whb.halui.program.stop (bit,out)  
Connect to program.stop. Pin for stopping a program.

whb.halui.program.is-idle (bit,in)  
Connect to halui.program.is-idle. Pin indicating no program is running.

whb.halui.mode.auto (bit,out)  
Connect to halui.mode.auto. Pin for requesting auto mode.

whb.halui.mode.is-auto (bit,in)  
Connect to halui.mode.is-auto. Pin for indicating auto mode is on.

whb.halui.mode.joint (bit,out)  
Connect to halui.mode.joint Pin for requesting joint by joint mode.

whb.halui.mode.is-joint (bit,in)  
Connect to halui.mode.is-joint. Pin indicating joint by joint mode is on.

whb.halui.mode.manual (bit,out)  
Connect to halui.mode.manual. Pin for requesting manual mode.

whb.halui.mode.is-manual (bit,in)  
Connect to halui.mode.is-manual. Pin indicating manual mode is on.

whb.halui.mode.mdi (bit,out)  
Connect to halui.mode.mdi. Pin for requesting MDI mode.

whb.halui.mode.is-mdi (bit,in)  
Connect to halui.mode.is-mdi. Pin indicating MDI mode is on.

whb.halui.mode.teleop (bit,out)  
Connect to halui.mode.teleop. Pin for requesting axis by axis mode.

whb.halui.mode.is-teleop (bit,in)  
Connect to halui.mode.is-teleop. Pin indicating axis by axis mode is on.

## Buttons

For flexibility reasons each button provides an output pin even if no functionality is realized directly with that signal. The Fn button can be combined with each other push-button. This includes also RESET, Stop, Start/Pause, Macro-10, and Step|Continuous. By default the more frequent used orange buttons are executed, whereas blue ones (whb.button.macro- $\langle M \rangle$ ) by combining them with Fn (press Fn first then button).

Button macro needs to be added to your INI and needs to be edited for your own use:

```
[HALUI]
MDI_COMMAND=(debug,macro0) # this one is for numbering but not used by pendant (need 1 to 16)
MDI_COMMAND=(debug,macro1)
MDI_COMMAND=(debug,macro2)
MDI_COMMAND=(debug,macro3)
MDI_COMMAND=(debug,macro4)
MDI_COMMAND=(debug,macro5)
MDI_COMMAND=(debug,macro6)
MDI_COMMAND=(debug,macro7)
MDI_COMMAND=(debug,macro8)
MDI_COMMAND=(debug,macro9)
MDI_COMMAND=(debug,macro10)
MDI_COMMAND=(debug,macro11)
MDI_COMMAND=(debug,macro12)
MDI_COMMAND=(debug,macro13)
MDI_COMMAND=(debug,macro14)
MDI_COMMAND=(debug,macro15)
MDI_COMMAND=(debug,macro16)
```

<M> ... denotes an arbitrary macro number which is of {1, 2, ..., 16}

whb.button.reset (bit,out)

See whb.halui.estop.{activate, reset} True one Reset button down, false otherwise. For toggling E-stop use whb.halui.estop .active and .reset.

whb.button.stop (bit,out)

See whb.halui.program.stop. True on Stop button down, false otherwise. For stopping a program use whb.halui.program.stop.

whb.button.start-pause (bit,out)

See whb.halui.program.{run, pause, resume}. True on Start-Pause button down, false otherwise. For toggling start-pause use 'whb.halui.program.run, .pause, and .resume.

whb.button.feed-plus (bit,out)

True on Feed+ button down, false otherwise.

whb.button.feed-minus (bit,out)

True on Feed- button down, false otherwise.

whb.button.spindle-plus (bit,out)

See halui.spindle.0.override.increase. True on Spindle+ button down, false otherwise. This button is meant to manipulate the spindle override. For increasing the spindle override use halui.spindle.0.override.increase.

whb.button.spindle-minus (bit,out)

See halui.spindle.0.override.decrease. True on Spindle- button down, false otherwise. This button is meant to manipulate the spindle override. For decreasing the spindle override use halui.spindle.0.override.decrease.

whb.button.m-home (bit,out)

Connect to halui.home-all. True on M-Home button down, false otherwise. Requests MDI mode before button pin is set. See also whb.halui.mode.mdi.

whb.button.safe-z (bit,out)

Connect to halui.mdi-command-'\_\_<M>\_\_ True on Safe-Z button down, false otherwise. Requests MDI mode before button pin is set. See also 'whb.halui.mode.mdi.

whb.button.w-home (bit,out)

Connect to `halui.mdi-command-‘__<M>__` True on W-HOME button down, false otherwise. Requests MDI mode before button pin is set. See also `‘whb.halui.mode.mdi`.

`whb.button.s-on-off` (bit,out)

See `‘whb.halui.spindle.‘‘{‘‘start‘‘, ‘‘stop‘‘}` True on S-ON/OFF button down, false otherwise. For toggling spindle on-off use `halui.spindle.0.start`. For toggling spindle on-off use `halui.spindle.0.stop`.

`whb.button.fn` (bit,out)

True on Fn button down, false otherwise.

`whb.button.probe-z` (bit,out)

Connect to `halui.mdi-command-‘__<M>__` True on Probe-Z button down, false otherwise. Requests MDI mode before button pin is set. See also `‘whb.halui.mode.mdi`.

`whb.button.macro-1` (bit,out)

Connect to `halui.mdi-command-<M>` True on Macro-1 button (Fn + Feed+) down, false otherwise.

`whb.button.macro-2` (bit,out)

Connect to `halui.mdi-command-<M>` True on Macro-2 button (Fn + Feed-) down, false otherwise.

`whb.button.macro-3` (bit,out)

See `whb.halui.spindle.increase` True on Macro-3 button (Fn + Spindle+) down, false otherwise. This button is meant to manipulate the spindle speed. For decreasing the spindle speed use `whb.halui.spindle.increase`.

`whb.button.macro-4` (bit,out)

See `whb.halui.spindle.decrease` True on Macro-4 button down (Fn + Spindle-), false otherwise. This button is meant to manipulate the spindle speed. For decreasing the spindle speed use `whb.halui.spindle.decrease`.

`whb.button.macro-5` (bit,out)

Connect to `halui.mdi-command-<M>` True on Macro-5 button down (Fn + M-HOME), false otherwise.

`whb.button.macro-6` (bit,out)

Connect to `halui.mdi-command-<M>` True on Macro-6 button down (Fn + Safe-Z), false otherwise.

`whb.button.macro-7` (bit,out)

Connect to `halui.mdi-command-<M>` True on Macro-7 button down (Fn + W-HOME), false otherwise.

`whb.button.macro-8` (bit,out)

Reserved for Spindle Direction True on Macro-8 button down (Fn + S-ON/OFF), false otherwise.

`whb.button.macro-9` (bit,out)

Connect to `halui.mdi-command-<M>` True on Macro-9 button down (Fn + Probe-Z), false otherwise.

`whb.button.macro-10` (bit,out)

Reserved for toggle DRO Abs/rel. True on Macro-10 button down, false otherwise. Switches the display coordinates to relative coordinates. On display the axis are denoted then as X1, Y1, Z1, A1, B1 and C1. See also `whb.halui.axis.‘__<N>__.pos-relative‘`.

`whb.button.macro-11` (bit,out)

Connect to `halui.mdi-command-<M>` True on Macro-11 button down (Fn + RESET), false otherwise.

`whb.button.macro-12` (bit,out)

Connect to halui.mdi-command- $\langle M \rangle$  True on Macro-12 button (Fn + Stop) down, false otherwise.

whb.button.macro-13 (bit,out)

Connect to halui.mdi-command- $\langle M \rangle$  True on Macro-13 button (Fn + Start/Pause) down, false otherwise.

whb.button.macro-14 (bit,out)

Connect to halui.mdi-command- $\langle M \rangle$  True on Macro-14 button (Fn + Macro-10) down, false otherwise.

whb.button.macro-15 (bit,out)

Connect to halui.mdi-command- $\langle M \rangle$  True on Macro-15 button down (Fn + MPG), false otherwise.

whb.button.macro-16 (bit,out)

Connect to halui.mdi-command- $\langle M \rangle$  True on Macro-16 button (Fn + Step) down, false otherwise.

whb.button.mode-continuous (bit,out)

True on Continuous mode button down, false otherwise.

whb.button.mode-step (bit,out)

True on Step mode button down, false otherwise.

## Pendant

whb.pendant.is-sleeping (bit,out)

True as long pendant is in sleep mode (usually a few seconds after turned off), false otherwise.

whb.pendant.is-connected (bit,out)

True as long pendant is not in sleep mode (turned on), false otherwise.

## HAL CONFIGURATION EXAMPLE

Exercise caution if using copy and paste of this example code from the online web docs. Certain characters are incompatibly encoded by the web site (minus becomes em-dash). It is safer to copy and paste from [https://raw.githubusercontent.com/LinuxCNC/linuxcnc/devel/src/hal/user\\_comps/xhc-whb04b-6/example-configuration](https://raw.githubusercontent.com/LinuxCNC/linuxcnc/devel/src/hal/user_comps/xhc-whb04b-6/example-configuration).

```
#
### Hal File xhc_whb04b_6.hal Example
#
# #####
# load pendant components
# #####

loadusr -W xhc-whb04b-6 -HsfB

# #####
# pendant signal configuration
# #####

# On/Off signals
net machine.is-on          halui.machine.is-on          whb.halui.machine.is-on
net pdnt.machine.on       whb.halui.machine.on          halui.machine.on
net pdnt.machine.off      whb.halui.machine.off          halui.machine.off

# program related signals
net pdnt.program.is-idle  whb.halui.program.is-idle     halui.program.is-idle
net pdnt.program.is-paused whb.halui.program.is-paused  halui.program.is-paused
net pdnt.program.is-running whb.halui.program.is-running  halui.program.is-running
```

net pdnt.program.resume	whb.halui.program.resume	halui.program.resume	
net pdnt.program.pause	whb.halui.program.pause	halui.program.pause	
net pdnt.program.run	whb.halui.program.run	halui.program.run	
net pdnt.program.stop	whb.halui.program.stop	halui.program.stop	
 # machine mode related signals			
net pdnt.mode.auto	whb.halui.mode.auto	halui.mode.auto	
net pdnt.mode.manual	whb.halui.mode.manual	halui.mode.manual	
net pdnt.mode.mdi	whb.halui.mode.mdi	halui.mode.mdi	
net pdnt.mode.joint	whb.halui.mode.joint	halui.mode.joint	
net pdnt.mode.teleop	whb.halui.mode.teleop	halui.mode.teleop	
net pdnt.mode.is-auto	halui.mode.is-auto	whb.halui.mode.is-auto	
net pdnt.mode.is-manual	halui.mode.is-manual	whb.halui.mode.is-manual	
net pdnt.mode.is-mdi	halui.mode.is-mdi	whb.halui.mode.is-mdi	
net pdnt.mode.is-joint	halui.mode.is-joint	whb.halui.mode.is-joint	
net pdnt.mode.is-teleop	halui.mode.is-teleop	whb.halui.mode.is-teleop	
 # "is-homed" axis signal for allowing pendant when machine is not homed			
net pdnt.axis.X.is-homed	halui.joint.0.is-homed	whb.halui.joint.x.is-homed	
net pdnt.axis.Y.is-homed	halui.joint.1.is-homed	whb.halui.joint.y.is-homed	
net pdnt.axis.Z.is-homed	halui.joint.2.is-homed	whb.halui.joint.z.is-homed	
 # "selected axis" signals			
net pdnt.axis.X.select	whb.halui.axis.x.select	halui.axis.x.select	
net pdnt.axis.y.select	whb.halui.axis.y.select	halui.axis.y.select	
net pdnt.axis.Z.select	whb.halui.axis.z.select	halui.axis.z.select	
 # jog scale			
net pdnt.axis.x.jog-scale	whb.axis.x.jog-scale	axis.x.jog-scale	
net pdnt.axis.y.jog-scale	whb.axis.y.jog-scale	axis.y.jog-scale	
net pdnt.axis.z.jog-scale	whb.axis.z.jog-scale	axis.z.jog-scale	
 # jog counts			
net pdnt.axis.x.jog-counts	whb.axis.x.jog-counts	axis.x.jog-counts	
net pdnt.axis.y.jog-counts	whb.axis.y.jog-counts	axis.y.jog-counts	
net pdnt.axis.z.jog-counts	whb.axis.z.jog-counts	axis.z.jog-counts	
 # jog enable			
net pdnt.axis.x.jog-enable	whb.axis.x.jog-enable	axis.x.jog-enable	
net pdnt.axis.y.jog-enable	whb.axis.y.jog-enable	axis.y.jog-enable	
net pdnt.axis.z.jog-enable	whb.axis.z.jog-enable	axis.z.jog-enable	
 # jog vel-mode			
net pdnt.axis.x.jog-vel-mode	whb.axis.x.jog-vel-mode	axis.x.jog-vel-mode	
net pdnt.axis.y.jog-vel-mode	whb.axis.y.jog-vel-mode	axis.y.jog-vel-mode	
net pdnt.axis.z.jog-vel-mode	whb.axis.z.jog-vel-mode	axis.z.jog-vel-mode	
 # macro buttons to MDI commands			
net pdnt.macro-1	whb.button.macro-1	halui.mdi-command-01	# use MDI command
net pdnt.macro-2	whb.button.macro-2	halui.mdi-command-02	# use MDI command
net pdnt.reserved.for.spindle+	whb.button.macro-3		# Hardcoded for spindle+ whb
net pdnt.reserved.for.spindle-	whb.button.macro-4		# Hardcoded for spindle- whb
net pdnt.macro-5	whb.button.macro-5	halui.mdi-command-05	# use MDI command
net pdnt.macro-6	whb.button.macro-6	halui.mdi-command-06	# use MDI command
net pdnt.macro-7	whb.button.macro-7	halui.mdi-command-07	# use MDI command
net pdnt.reserved.for.spindle.dir	whb.button.macro-8		# Hardcoded for spindle direct whb
net pdnt.macro-9	whb.button.macro-9	halui.mdi-command-09	# use MDI command

```

net pdnt.reserved.for.ABS-REL      whb.button.macro-10      # Hardcoded for swap Dro
net pdnt.macro-14                 whb.button.macro-14      halui.mdi-command-14     # use MDI command
net pdnt.reserved.for.flood       whb.button.macro-15      # Hardcoded for halui.flood on/
net pdnt.reserved.for.mist        whb.button.macro-16      # Hardcoded for halui.mist on/c

net pdnt.macro.11                 whb.button.macro-11      halui.mdi-command-11     # use MDI command
net pdnt.macro.12                 whb.button.macro-12      halui.mdi-command-12     # use MDI command
net pdnt.macro.13                 whb.button.macro-13      halui.mdi-command-13     # use MDI command

# flood and mist toggle signals
net pdnt.flood.is-on             whb.halui.flood.is-on    halui.flood.is-on        #return signal is on or off
net pdnt.flood.off               whb.halui.flood.off      halui.flood.off          #reserved whb.button.macro-15
net pdnt.flood.on                whb.halui.flood.on       halui.flood.on           #reserved whb.button.macro-15

net pdnt.mist.is-on              whb.halui.mist.is-on     halui.mist.is-on         #return signal is on or off
net pdnt.mist.off                whb.halui.mist.off       halui.mist.off           #reserved whb.button.macro-16
net pdnt.mist.on                 whb.halui.mist.on        halui.mist.on            #reserved whb.button.macro-16

# default function button signals
net pdnt.button.m-home           whb.button.m-home        halui.home-all           # Homeing use built-in
net pdnt.button.safe-z           whb.button.safe-z        halui.mdi-command-03     # Safe-z use MDI cor
net pdnt.button.w-home           whb.button.w-home        halui.mdi-command-04     # Unpark use MDI
net pdnt.button.probe-z          whb.button.probe-z       halui.mdi-command-08     # Probe-Z use MDI

# unused, just exposes pendant internal status or as basic button
#net pdnt.mode-lead              whb.halui.feed.selected-lead
#net pdnt.mode-mpg-feed          whb.halui.feed.selected-mpg-feed
#net pdnt.mode-continuous        whb.halui.feed.selected-continuous
#net pdnt.mode-step              whb.halui.feed.selected-step

#net pdnt.button.mode-mpg        whb.button.mode-continuous
#net pdnt.button.mode-step       whb.button.mode-step
#net pdnt.button.fn              whb.button.fn
#net pdnt.button.reset           whb.button.reset
#net pdnt.button.stop            whb.button.stop
#net pdnt.button.start-pause     whb.button.start-pause
#net pdnt.button.s-on-off        whb.button.s-on-off
#net pdnt.button.spindle-plus    whb.button.spindle-plus
#net pdnt.button.spindle-minus   whb.button.spindle-minus
#net pdnt.button.feed-plus       whb.button.feed-plus
#net pdnt.button.feed-minus      whb.button.feed-minus

# spindle related signals
net pdnt.spindle.is-on           whb.halui.spindle.is-on  spindle.0.on
net pdnt.spindle.start           whb.halui.spindle.start  halui.spindle.0.start
net pdnt.spindle.stop            whb.halui.spindle.stop   halui.spindle.0.stop
net pdnt.spindle.forward         whb.halui.spindle.forward halui.spindle.0.forward
net pdnt.spindle.reverse         whb.halui.spindle.reverse halui.spindle.0.reverse
net pdnt.spindle.increase        whb.halui.spindle.increase halui.spindle.0.increase # reserved whb.button.
net pdnt.spindle.decrease        whb.halui.spindle.decrease halui.spindle.0.decrease # reserved whb.button
net pdnt.spindle-speed-abs       whb.halui.spindle-speed-cmd spindle.0.speed-out-abs # speed cmd fro

```



```

# spindle speed override signals
net pdnt.spindle-override.scale      whb.halui.spindle-override.scale  halui.spindle.0.override.scale # needed for bo
net pdnt.spindle.override.value      halui.spindle.0.override.value     whb.halui.spindle-override.value # GUI feed rate
net pdnt.spindle.override.increase   whb.halui.spindle-override.increase halui.spindle.0.override.increase
net pdnt.spindle.override.decrease   whb.halui.spindle-override.decrease halui.spindle.0.override.decrease

# GUI feed rate related signals can be used when program is running moving GUI slider
net pdnt.feed-override.scale         whb.halui.feed-override.scale      halui.feed-override.scale # needed for both I
net pdnt.max-velocity.value          whb.halui.max-velocity.value       halui.max-velocity.value # needed for Mp

# take feed override min/max values from/to the GUI
net pdnt.feed-override.value         halui.feed-override.value          whb.halui.feed-override.value # GUI feed rate r
net pdnt.feed-override.increase      whb.halui.feed-override.increase   halui.feed-override.increase
net pdnt.feed-override.decrease      whb.halui.feed-override.decrease   halui.feed-override.decrease

# axis position related signals feedback
net pdnt.axis.x.pos-feedback         halui.axis.x.pos-feedback          whb.halui.axis.x.pos-feedback
net pdnt.axis.y.pos-feedback         halui.axis.y.pos-feedback          whb.halui.axis.y.pos-feedback
net pdnt.axis.z.pos-feedback         halui.axis.z.pos-feedback          whb.halui.axis.z.pos-feedback

# axis position related signals relative
net pdnt.axis.x.pos-relative         halui.axis.x.pos-relative          whb.halui.axis.x.pos-relative
net pdnt.axis.y.pos-relative         halui.axis.y.pos-relative          whb.halui.axis.y.pos-relative
net pdnt.axis.z.pos-relative         halui.axis.z.pos-relative          whb.halui.axis.z.pos-relative

```

**SEE ALSO**

xhc-whb04b-6 developer documentation on GitHub

**NOTES**

The CRC code function is not disclosed by the manufacturer. Thus the CRC value transmitted with each package is not checked yet. Feel free to help us enhance the component.

**AUTHOR**

This component was started by Raoul Rubien based on predecessor device component xhc-hb04.cc. <https://github.com/machinekit/machinekit/graphs/contributors> gives you a more complete list of contributors.

**HISTORY**

The component was developed accidentally as leisure project. The development started with the xhc-whb04 (4-axis wireless pendant) implementation as reference. 73 & many thanks to the developers who delivered provided an excellent preparatory work!

**COPYRIGHT**

Copyright © 2018 Raoul Rubien ([github.com/rubienr](https://github.com/rubienr)) Updated for Linuxcnc 2020 by alkabal\_free.fr.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

xyzab-tdr-gui – Vismach Virtual Machine GUI

**DESCRIPTION**

**xyzab-tdr-gui** is one of the sample **Vismach** GUIs for LinuxCNC, simulating a dual-rotary machine with switchable kinematics.

**SEE ALSO**

linuxcnc(1)\*

See the main LinuxCNC documentation for more details: <https://linuxcnc.org/docs/html/gui/vismach.html>

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>

**COPYRIGHT**

Copyright © 2023 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

xyzac-trt-gui – Vismach Virtual Machine GUI

**DESCRIPTION**

**xyzac-trt-gui** is one of the sample **Vismach** GUIs for LinuxCNC, simulating a 5-axis milling machine with tool-point kinematics.

See the main LinuxCNC documentation for more details.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**NAME**

xyzbc-trt-gui – Vismach Virtual Machine GUI

**DESCRIPTION**

**xyzbc-trt-gui** is one of the sample **Vismach** GUIs for LinuxCNC, simulating a 5-axis milling machine with tool-point kinematics

See the main LinuxCNC documentation for more details.

**SEE ALSO**

linuxcnc(1)

Much more information about LinuxCNC and HAL is available in the LinuxCNC and HAL User Manuals, found at `/usr/share/doc/LinuxCNC/`.

**BUGS**

None known at this time.

**AUTHOR**

This man page written by Andy Pugh, as part of the LinuxCNC project.

**REPORTING BUGS**

Report bugs at <https://github.com/LinuxCNC/linuxcnc/issues>.

**COPYRIGHT**

Copyright © 2020 Andy Pugh.

This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.