Router HAL Configuration

Generated by Doxygen 1.9.3

Chapter 1

Router Control

1.1 Information

-
-

1.2 Features

- Faults
- Feedback

1.3 ToDo

1.3.1 General Improvements

- Document spindle
- Document steppers
- Document estop latch

1.3.2 General Nice to Have's

- Document Commander SK VFD

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Chapter 3

Module Documentation

3.1 EStopLatch

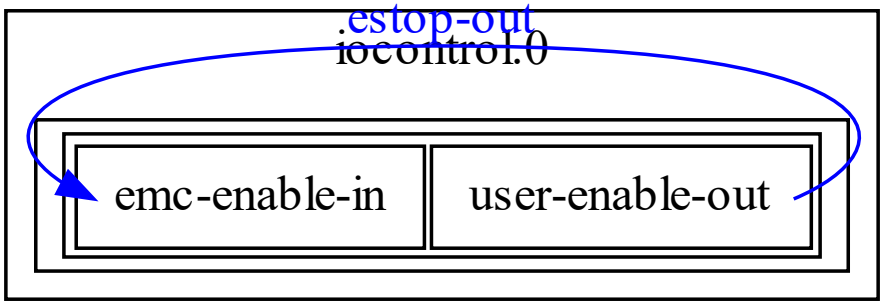
This approach is based on https://forum.linuxcnc.org/47-hal-examples/25861-external-e-stop?star

I like diagrams when I'm troubleshooting or modifying a HAL configuration. I've used the Doxygen tool and Vizgraph to illustrate how to use the LinuxCNC EStopLatch to implement EStop with multiple hardware signals.

3.1.1 Generic Estop HAL

The default configuration (from stepconf or) simply connects iocontrol pin user-enable-out to pin emc-enable-in with signal estop-out. This allows software control of estop. If you look in your machine's .hal file you will find lines similar to:

```
#  ---estop signals---
net estop-out <= iocontrol.0.user-enable-out
net estop-out => iocontrol.0.emc-enable-in
```



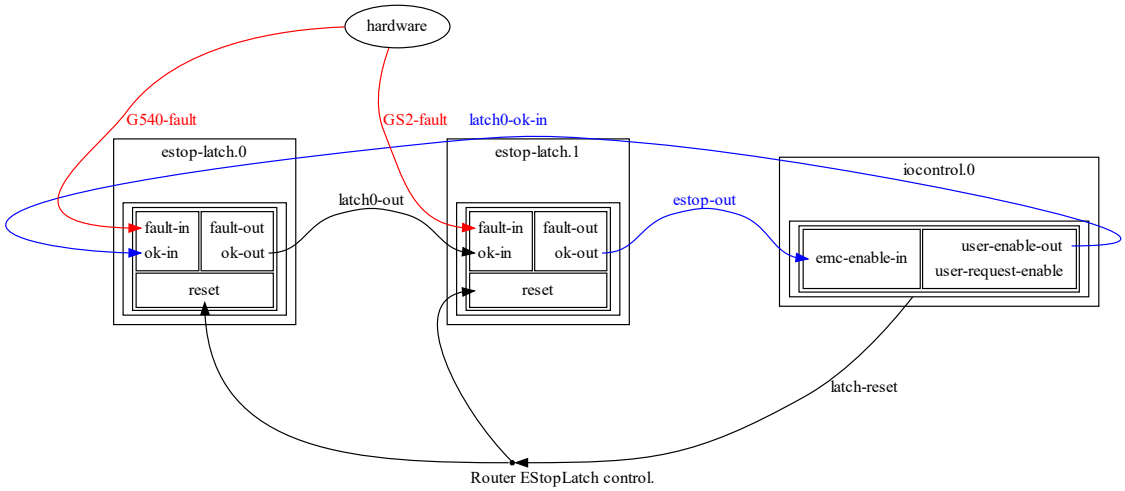
Generic EStop HAL.

3.1.2 Router EStopLatch HAL

If you have hardware signals that you wish to estop your machine, you can insert one or more estop latch components into your .hal configuration.

This diagram illustrates how I add two estop latches to my router configuration. The hardware signals that can estop are:

- G540-fault - Fault signal from Gecko G540. This signal indicates either the hardware estop switch is pressed or a driver fault.
- GS2-fault - GS2 VFD fault signal (overload).



3.1.3 Router EStopLatch Implementation

I like to keep functionality in separate files. For EStopLatch, I keep the EStop latch config in estop-chain.hal. I add the following lines to my ∗.ini. "HALFILE = estop-chain.hal" is the relevant line. The other lines are just to illustrate how I organize HAL functionality.

```
#
# I break up the .hal into multiple files to make it easier to document and share features between
      configurations.
#
[HAL]
TWOPASS = on
HALUI = halui
HALFILE = router.hal
HALFILE = parport.hal
HALFILE = spindle.hal
HALFILE = estop-chain.hal
HALFILE = probes.hal
HALFILE = postgui_load.hal # file_with_all_loads_for_postgui.hal
POSTGUI_HALFILE = postgui_call_list.hal
```

Here are the contents of my estop-chain.hal file.

```
#
# This file contains the Hardware Abstraction Level for e-stop hardware signals.
# Based on
      https://forum.linuxcnc.org/10-advanced-configuration/32789-educate-me-about-estop-chains-and-latches?start=10#93394
#
#
# The default setup has estop-out <= iocontrol.0.user-enable-out => iocontrol.0.emc-enable-in.
# This file inserts a chain of estop latches between user-enable-out and emc-enable-in.
```

```
# Set up estop latches for G540 fault, VFD fault.
loadrt estop_latch count=2
addf estop-latch.0            servo-thread
addf estop-latch.1            servo-thread
#
#   Reset signal for each latch comes from user-request-enable pin.
#
net latch-reset <= iocontrol.0.user-request-enable => estop-latch.0.reset estop-latch.1.reset
#
#  Hook the first latch ok-in to iocontrol user-enable-out.
#
net latch0-ok-in <= iocontrol.0.user-enable-out => estop-latch.0.ok-in
#
#  Connect each ok_out to the next ok_in to create a chain of latches.
#
net latch0-out <= estop-latch.0.ok-out => estop-latch.1.ok-in
#
#   Last estop in the chain tells LinuxCNC we have an estop.
#
net estop-out <= estop-latch.1.ok-out => iocontrol.0.emc-enable-in
#
#   Connect various input signals to the estop latch fault_in pins.
#   These signals are defined for my router as connected to parallel port pins.
#
net G540-fault => estop-latch.0.fault-in
net GS2-fault  => estop-latch.1.fault-in
```

Don't forget to comment out or remove the orginal estop lines from your ∗.hal.

```
#  ---estop signals---
# net estop-out <= iocontrol.0.user-enable-out
# net estop-out => iocontrol.0.emc-enable-in
```

My router uses two different parallel ports for IO and the HAL is defined in parport.hal. Depending on your hardware estop connections, it will look something like this:

```
net G540-fault      <= parport.0.pin-15-in
net GS2-fault       <= parport.0.pin-11-in
```

If you have any problems, use halmeter to examine your hardware signals and the iocontrol.emc-enable-in status.

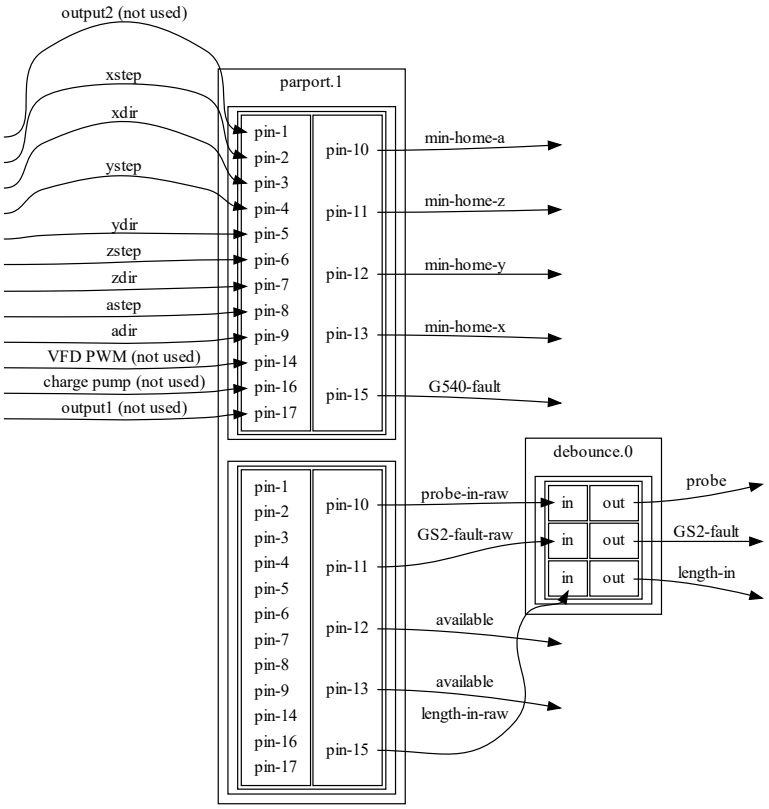
3.2 ParallelPort

This diagram illustrates the .hal components for the parallel ports of my router.

Two parallel ports in "out" direction mode.

- parport.0 is connected to a Gecko G540.
- parport.1 is just a breakout cable.

Notice, from the perspective of HAL, the "input" pins drive other HAL signal (act as ouputs).



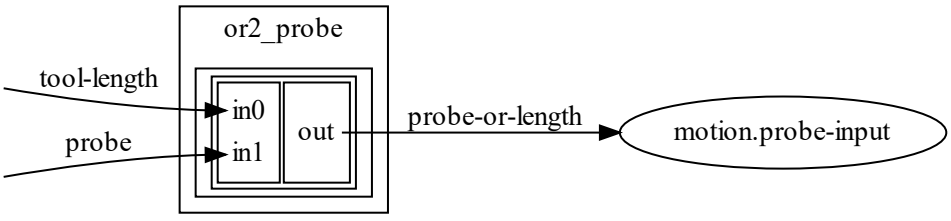
Router parallel ports.

3.3 Probes

This diagram illustrates the .hal components for the tool length and touch off probes for my router.

There are two probes:

- tool length
- touch off

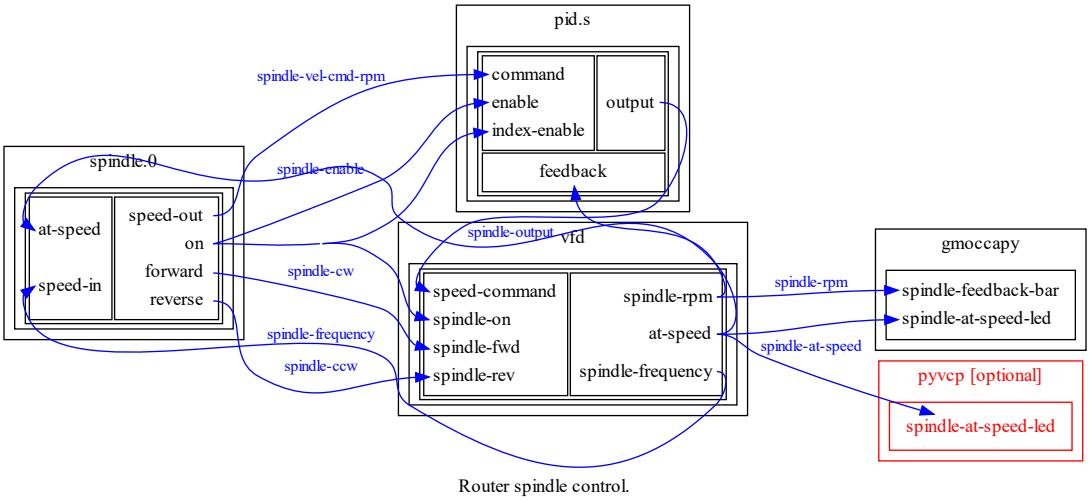


Router Probes.

3.4 SpindleControl

This diagram illustrates the .hal components I use with a GS2 VFD using the serial communications. Note, pid is used to get the correct rpm from the vfd.

The fault relay is documented in the estoplatch documentation.



3.5 StepperControl

My router uses simple stepper motors using a parallel port and Gecko G540.

Index