Helvar testing rig quick start guide

Ville Hiltunen

02.09.2019

1. Plug both the 12V and 5V power adapters into the machine. Plugging the connectors into the wrong plugs won't cause electrical problems, but the machine won't turn on.
2. Turn on the power switch. The xy-plane stepper motors should now blink their LEDs. If not, check power adapters.
3. Connect the Arduino to the device.
4. Open a serial connection with Arduino. Baud rate should be 38400 unless modified. The Arduino serial monitor has been tested to work, but in theory any serial terminal should work.
5. The serial should display formatted replies from the motors. If not, you've ran into the mysterious bug that blocks communication on the RS-485 line. Try again later and/or after cycling the device power.
6. If replies are displayed on the screen, insert the command "homesen" (without the quotes) to home the sensor header. All commands should have a newline ('\n') as the end character.
7. Next, insert command "homexy" to home the XY-plane.
8. The device should now be ready to use! The command format for commands with arguments is cmd(arg1,arg2,...,argx).
9. Remove a cover from a slot, then place a model to the slot. The magnets should pull the model and the connector together. Use srva(slotnumber) to add the model. If everything went right, the servo should be waving back and forth and heating up. If not, either there is a problem with the connection or the model is misplaced. Use srvl to list servo locations. Note that the first slot starts from the corner where the Arduino usb plug is connected.
10. Try more commands from the list below.

Command list and descriptions:

Main functionalities:

Command word	Arguments	Description
homesen	0	Homes the sensor header to the middle position, which is set to be angle 0.
homexy	0	Homes the xy-plane. Note: This command will flush the command buffer, and while the device is homing, no new commands will be read from the buffer. At the end of homing, the sensor head will be placed above the corner of the simulated room, and this position will be considered (0,0) for millimeter coordinates.
xypos	2	Run the sensor head to a xy-position in millimeters. Argument 1: x-axis from 0 to 1000. Argumen 2: y-axis from 0 to 500. Example input: xypos(100,250)
sen	1	Turns the sensor head to the given angle. Input: Sensor angle relative to the 0 position from -125 to 125. If the sensor is unhomed, the sensor head might overshoot and hit a limit switch. If this happens, the device should be rehomed. Example input: sen(50)
help	0	Lists the commands, arguments and constraints to the serial line.
heatc	1	Sets the heating power as a percentage between 0 and 100. Example input: heatc(15)
heatl	0	Lists the current heating power.
heati	0	Activates full heating power for 60 seconds.
srva	1	Adds a model to the given slot. Accepts inputs from 1-32.

		Example input: srva(1)
srvr	1	Removes a model from the given slot. Input: 1 to 32 or -1. Inputting -1 purges all servos. Example input: srvr(1) Bug: Current code allows 0 as a valid input, which results in an access outside of array bounds.
srvl	0	Lists the positions of models and speeds and hold delays of all slots.
srvs	2	Changes model swaying speed. Argument 1: The slot number from 1 to 32 or -1 for all models. Argument 2: The speed percentage from 1 to 100. The command displays the actual microsecond delay after input. Example input: srvs(-1,50) Bug: Current code allows 0 as a valid input, which results in an access outside of array bounds.
srvm	2	Sets the model to the given movement state. Argument 1: The slot number from 1 to 32 or -1 for all models. Argument 2: The new state, either 0 or 1. 0 for constant sway, 1 for delay at the max and min angles. Example input: srvm(1,1) Bug: Current code allows 0 as a valid input, which results in an access outside of array bounds.
srvh	2	Changes model holding delay. Default setting holds the model in place for 3 seconds at minimum and maximum angles. Argument 1: The slot number from 1 to 32 or -1 for all models. Argument 2: The hold in milliseconds from 5 to 30000. The command displays the actual microsecond delay after input. Example input: srvh(-1,2000)

		Bug: Current code allows 0 as a valid input, which results in an access outside of array bounds.

Trinamic stepper motor commands:

These commands are very similar to the Trinamic PD42 stepper commands and speak directly to the rig stepper motors. Care should be taken with these functions, as they might alter the behavior of the program.

Command word	Arguments	Description
mst	1	Stops the target motor. Argument is the motor address, either 1 or 2.
gap	2	Asks for a certain axis parameter from the driver. Argument 1 is the motor address, 1 or 2. Argument 2 is the command type, aka which parameter we are interested in , from 0 to 255. See trinamic documentation in references for details.
sap	3	Sets a certain axis parameter on the driver. Argument 1 is the motor address, 1 or 2. Argument 2 is the command type, aka which parameter we are interested in, from 0 to 255. Argument 3 is the value we want to set the parameter to. See trinamic documentation in references for details.
ggp	3	Asks for a certain global parameter from the driver. Argument 1 is the motor address, 1 or 2. Argument 2 is the command type, aka which parameter we are interested in, from 0 to 255. Argument 3 is the memory bank from 0 to 3. See trinamic documentation in references for details.
sgp	4	Sets a certain global parameter in the driver. Argument 1 is the motor address, 1 or 2. Argument 2 is the command type, aka which parameter we are interested in, from 0 to 255. Argument 3 is the memory bank from 0 to 3. Argument 4 is the new value for the parameter.

		See trinamic documentation in references for details.
rfs	2	Controls the homing of the target motor. Argument 1 is the target motor, 1 or 2. Argument 2 is the command. 0 for starting, 1 for stopping and 2 for asking for status. See trinamic documentation in references for details.
exp	5	Not a true trinamic command, rather this command emulates a totally free form 8 byte package for sending to the motor drivers. Consult the firmware documentation and know exactly what you are doing before using this. Argument 1 is the target motor address. Argument 2 is the instruction number. Argument 3 is the type number. Argument 4 is the motor or bank number. Argument 5 is the value. Checksum is calculated automatically for the user.

References

Trinamic PD42 stepper page. See the firmware manual for more info on stepper commands: https://www.trinamic.com/products/drives/stepper-pandrives-details/pd42-x-1141/