Setting up TSC - alpha with a Raspberry Pi 3 Model B

TSC (TwoStepperControl) is a software project by Wolfgang Birkfellner and Steffen Elste to realize an open integrated control system for astronomical telescopes. The basic idea is to use a Raspberry Pi to control a telescope and to provide basic autoguiding functionality without additional external computers. As Raspbian Jessie is not an RTOS, it cannot control stepper drivers directly like a microcontroller; therefore, extra hardware is necessary which receives basic parameters (number of steps, velocity, acceleration) from the software. Currently, I am using a Phidgets 1067 bipolar driver board although other alternatives are under consideration. The steppers are monitored by the Pi in separate Qt-Concurrent threads so that software functionality is maintained during stepper operation. So far, only fork mounts are supported.

Software repository:

https://github.com/selste/TwoStepperControl

Details on the software setup are covered in the Wiki on github – this Readme documents Raspian system setup measures and hardware required.

Installing TSC:

Get yourself a good quality 16 GB SD-card and copy an image of the Raspian running TSC. In Ubuntu, this is done by typing

```
dd if=/dev/tscimage.img of=/dev/mmcblk0 bs=8M
```

Basic components:

- Operating System: Raspbian Jessie - https://www.raspberrypi.org/downloads/noobs/
- Currently, the locale should be english.
- Raspberry Pi 3 Model B, with at least a 16 GB MicroSD card and a 2A micro-USB power adapter. Cost ~ 70 €. I would recommend connecting the Pi to the motor power supply while using a step-down converter.
- Be aware that one needs a lot of power to drive the Phidget boards. Get yourself a sufficient power supply for everything else but the Pi.
- Buy cooling bodies for the Pi, especially the ARM CPU (~5 €), and plan to get a fan for the Pi and the Phidget boards.
- Stepper control boards: Currently, 1067_0 - PhidgetStepper Bipolar HC, available from www.phidgets.com or www.nodna.de are used. Two boards are necessary. These operate one bipolar stepper each with up to 4 A coil current. 1/16 microsteps are available, the board is controlled via USB 2.0. Cost for each board is currently ~90 €. A development with alternate driver boards is under way but not yet available. As Raspian is not a RTOS, it cannot control the drivers directly.
- A Raspberry Pi compatible USB 2.0 hub with separate power. Not all hubs work, so be careful.
- An Adafruit DS3231 buffered clock module, so that the Pi also knows the proper time when being

operated without internet connection: https://learn.adafruit.com/adding-a-real-time-clock-to-raspberry-pi/set-rtc-time?view=all

- -- Optional components ------------------------------------
- Optionally, a HDMI cable and an HDMI monitor. Development right now aims at using the Adafruit 5” HDMI 800*480 Backpack Touchscreen (~80 €). Basically, it is also possible to connect via W-LAN or WI-FI direct and a VNC viewer to the Raspberry using a mobile or a tablet – check the github Wiki for details. TSC is also configured to act as an independent hotspot, therefore this solution also works when no WLAN is available.
- Optionally, wireless mouse and keyboard. Cost ~20 €.
- Optionally, a RS232 adapter to connect to planetarium programs via wire. For this purpose, it is necessary to get a circuit like this one: https://www.conrad.at/de/rs232-erweiterungs-platine-fuer-den-raspberry-pi-1337093.html. Denote that this one needs a nullmodem-cable and a RS232 to USB converter if your computer does not have one. Alternatively, a direct CP2102 converter like this one might be of interest: https://electronics.semaf.at/navi.php?a=962&lang=eng. Denote that with certain programs like Sky Safari Plus and Cartes du Ciel, a TCP/IP connection is also possible. In this case, no converter is needed.
- For the optional bluetooth handbox, one also needs an Arduino Uno (24 € for the original one (strongly recommended) or a few coins for chinese copies - other models like the Nano might work as well) and a HC-05 bluetooth board (3-6 € + postage on ebay). Also, a lever switch and four touch switches, wires, a 9V power supply, a housing and 5 10k resistors are necessary. Putting the handbox on a prototype shield is feasible. An Arduino sketch can be found in the repository called “sketch_handbox_TSC.ino” for operating the handbox

- **Software requirements:**
  - Development environment is Qt and QtCreator using C++. Some info can be found – for instance on missing libxcb libraries- -here: https://wiki.qt.io/Native_Build_of_Qt_5.4.1_on_a_Raspberry_Pi
  - Install the Phidgets – library and set the right access privileges for the USB port if you don't want to run the program in root mode. Check http://www.instructables.com/id/Getting-Started-with-Phidgets-on-the-Raspberry-Pi/ … Basically, you need the libusb 1.0 and libphidgets 2.1. Changing the udev rules is also necessary (this is taken from http://tordwessman.blogspot.co.at/2012/01/running-phidgets-under-linuxubuntu.html):
    - $ sudo nano /etc/udev/rules.d/80_phidget.rules
    - Add the following content: SUBSYSTEMS=="usb", ACTION=="add", ATTRS{idVendor}=="06c2", ATTRS{idProduct}=="00[3-a][0-f]", MODE="666"

        - Set `USB_DEVFS_PATH` to `/dev/bus/usb` by adding to `~/.bashrc`:
          ```
          export USB_DEVFS_PATH=/dev/bus/usb
          ```
        - Restart udev:
          ```
          $ services udev restart
          ```
        - *I*f it doesn't work. Try the good old:
          ```
          $ sudo reboot
          ```
    - Image acquisition of guiding cameras is realized via INDI. A tutorial for installing it on the Pi is found under http://indilib.org/support/tutorials/139-indi-library-on-raspberry-pi.html
    - In order to utilize the serial port, one has to disable it's current purpose – serving as a console to the pi.
        - Edit `/boot/config.txt` by typing `enable_uart=1.`
        - Disable console output from Jessie by typing:
          ```
          $ sudo systemctl stop serial-getty@ttyS0.service
          $ sudo systemctl disable serial-getty@ttyS0.service
          ```
        - Type sudo nano /boot/cmdline.txt and remove the statement `console=serial0,115200` from that text. After reboot, `/dev/ttyS0` should be available. More can be found under

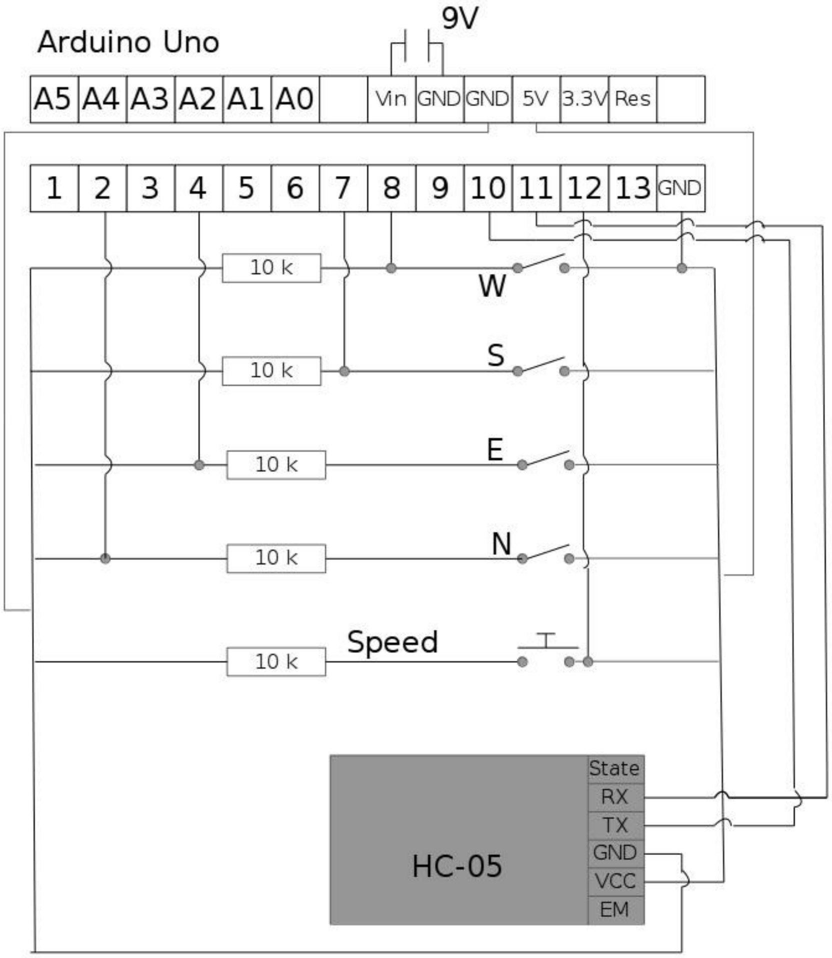
          http://spellfoundry.com/2016/05/29/configuring-gpio-serial-port-raspbian-jessie-including-pi-3/
        - Make the port readable for all: `sudo chmod a+rw /dev/ttyS0`
    - Enable the Autohotspot following the instructions from: http://www.raspberryconnect.com/network/item/315-rpi3-auto-wifi-hotspot-if-no-internet
    - Enable the hardware clock following the instructions from https://learn.adafruit.com/adding-a-real-time-clock-to-raspberry-pi/set-rtc-time?view=all

The Bluetooth handbox:
The handbox features five switches, one on/off switch for speed selection and four switches for motion. It features an Arduino Uno and a HC-05 Bluetooth interface. A simplified schematic is shown.

Setting up the HC-05:

- I follow the instructions from https://myraspberryandme.wordpress.com/2013/11/20/bluetooth-serial-communication-with-hc-05/
- For setup, the HC-05 “STATE” has to be connected to 5V. The sketch *“sketch_HC05_setup.ino”* is used to configure the HC-05. CR/LF have to be enabled for communicating via the serial interface.

- Send the following commands:
  - `AT+NAME:TSC`
  - `AT+ADDR?` gives the MAC address of the device – write it down!
  - `AT+UART:9600,1,0` sets the communication parameters
  - `AT+ROLE: 0` makes the device a slave
- Disconnect the wire from 5V to the STATE pin.



- Make sure that Bluez is downloaded - https://learn.adafruit.com/install-bluez-on-the-raspberry-pi/installation
- Write a file `rfcomm.conf` in `/etc/bluetooth`; it should read:

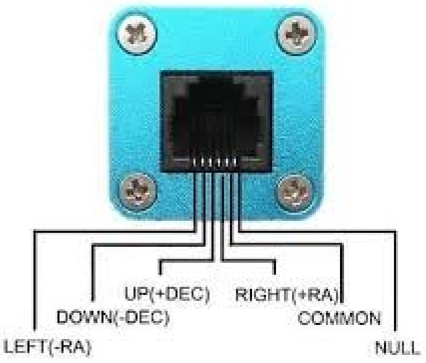
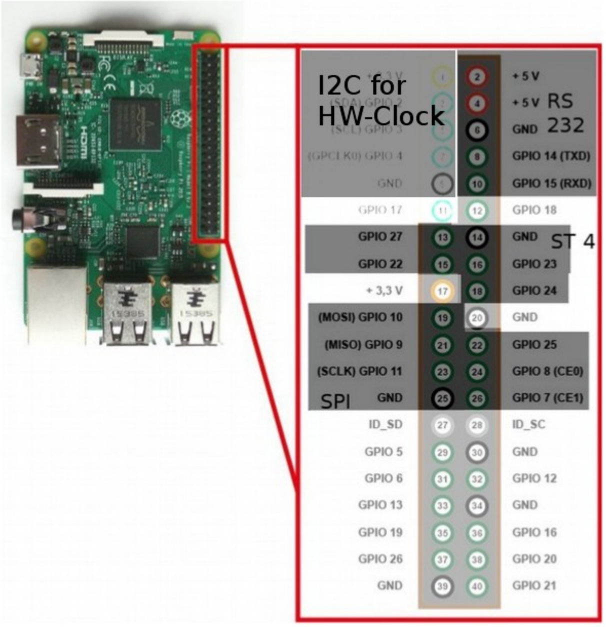
```
rfcomm0 {
# Automatically bind the device at startup
bind yes;
# Bluetooth address of the device
device 98:D3:31:FB:2A:8C
#  your MAC address for the HC05 should go here!
# RFCOMM channel for the connection
channel 1;
# Description of the connection
comment "TSC";
}
```

- Write a file “`pincodes`” in “`/var/lib/bluetooth/xx:xx:xx:xx:xx:xx/`” where the PIN code of the HC05 is stored; this is usually 1234, it can be edited in setup, but this is default.
- Try to pair the HC05 by using the BT-tool … it should work now.

The ST 4 interface:

ST 4 can be used. For this purpose, a standard RJ 11 jack has to be wired to the GPIO port of the Raspberry Pi 3 as follows:

Direction	Direction	Wiring Pi Port	BCM Pin	Pi Pin	ST 4 Pin
Declination +	North	2	27	13	5
Declination -	South	4	23	16	4
Right Ascension +	West	5	24	18	3
Right Ascension -	East	3	22	15	6
GND				14	2


 UP(+DEC)
RIGHT(+RA)
DOWN(-DEC)
COMMON
LEFT(-RA)
NULL
	
	 I2C for
	HW-Clock
	+3,3 V
	(SDA) GPIO 2
	(SCL) GPIO 3
	(GPCLK0) GPIO 4
	GND
	+5 V
	+5 V
	RS
	232
	GND
	GPIO 14 (TXD)
	GPIO 15 (RXD)
	GPIO 17
	GPIO 18
	GPIO 27
	GND
	ST 4
	GPIO 22
	GPIO 23
	+3,3 V
	GPIO 24
	(MOSI) GPIO 10
	GND
	(MISO) GPIO 9
	GPIO 25
	(SCLK) GPIO 11
	GPIO 8 (CE0)
	SPI
	GND
	GPIO 7 (CE1)
	ID_SD
	ID_SC
	GPIO 5
	GND
	GPIO 6
	GPIO 12
	GPIO 13
	GND
	GPIO 19
	GPIO 16
	GPIO 26
	GPIO 20
	GND
	GPIO 21

The ST 4 Pins, as seen from the jack. Numbering goes from right to left.	The GPIO pins of the Raspberry Pi3; the program needs the pins in the gray areas.