dewi Documentation

Release 0.1.0

Louis Taylor

Mar 14, 2018

Contents

Contents:

CHAPTER 1

Wiring standard

This wiring standard is our best practice, all new cabling should comply with this at all times, and be labled as compliant. Any non-compliant existing wiring should be labled as non-compliant, with a complete list of connections listed both on the wires and in the coresponding git repository

1.1 Wire colour

1.1.1 Power

- Black = gnd/0v
- Purple = 12v
- Red = 5v
- Orange = 3.3v
- Green = Battery voltage

1.1.2 I2C

- Blue = SCL
- Yellow = SDA

1.1.3 Serial

- White & Grey = RX and TX, Rx mark on both ends with a red mark.

1.1.4 Servos

- Brown = gnd
- Red = pwr
- Orange = signal

1.2 Wire thickness

Wire thickness should be chosen accorning to the maximum current draw through the wire.

For currents up to 6 amps a guage of AWG 14 (1.6mm diameter) or greater should be used, only the battery or sailwinch wires should require this much current.

For currents up to 1.8 amps a guage of AWG 19 (.1mm diameter) or greater should be used, the electronics power wires, RoWind and servos may require this wire.

for currents up to 900mA a guage of AWG 22 (.64mm diameter) or greater should be used, this will handle all external data and sensor power wires, with the exception of those mentioned above.

1.3 Termination

1.3.1 Solder termination

TBD

CHAPTER 2

Provisioning

To run the provisioning scripts, get a local copy of this repository on your machine:

```
$ git clone --recursive https://github.com/abersailbot/dewi
```

Now set up ssh configuration to connect to the target device (the pi on the boat). If you don't already have one, create `~/.ssh/config` and add something like the following, changing the hostname accordingly.

..code-block:

```
Host dewi
  User pi
  HostName 192.168.0.10
```

Generate an ssh key if you do not already have one:

```
$ ssh-keygen -f ~/.ssh/id_rsa -t rsa
```

Copy the ssh key to the pi:

```
$ ssh-copy-id dewi
```

Then, run:

```
$ ./deploy dewi
```

This should copy all the local data to the pi and install the configuration to the appropriate locations.

CHAPTER 3

Testing

Integration testing of configuration will be done using a QEMUed jessie raspbian image.

These steps assume you have the necessary tools installed.

Fedora:

```
$ sudo dnf install qemu-system-arm
```

Debian/Ubuntu:

```
$ sudo apt install qemu-system-arm
```

3.1 Building disk image

Building a new disk image to use for testing is currently a manual process. In the future, this could be automated.

1. Grab a copy of the latest 'raspbian jessie lite' image from the raspberry pi website (https://www.raspberrypi.org/downloads/raspbian/). I'm using `2015-11-21-raspbian-jessie-lite.img` here.
2. Use *fdisk* to find the partition boundaries in this image:

```
~/g/dewi (master) $ fdisk -l 2015-11-21-raspbian-jessie-lite.img
Disk 2015-11-21-raspbian-jessie-lite.img: 1.4 GiB, 1458569216 bytes, 2848768␣
↪sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xb3c5e39a

Device                              Boot  Start     End Sectors  Size Id Type
```

(continues on next page)

(continued from previous page)

```
2015-11-21-raspbian-jessie-lite.img1         8192  131071  122880   60M  c W95␣
↪FAT32 (LBA)
2015-11-21-raspbian-jessie-lite.img2       131072 2848767 2717696  1.3G 83 Linux
```

3. Next, we need to mount that image. To do this, we need to use the information `fdisk` told us. The start of the main partition is block `131072`, and the block size is `512`, so the total offset is `131072*512` = `67108864`.

   Make a temporary directory to mount the image in:

   ```
   $ mkdir raspbian-jessie-mount-point
   ```

   and mount the image:

   ```
   $ sudo mount -o loop,offset=67108864 2015-11-21-raspbian-jessie-lite.img raspbian-
   ↪jessie-mount-point
   ```

4. Edit `raspbian-jessie-mount-point/etc/ld.so.preload` and comment out all lines.
5. Edit `raspbian-jessie-mount-point/etc/fstab` and comment out any entry related to `mmcblk`.
6. Unmount the image:

   ```
   $ sudo umount raspbian-jessie-mount-point
   ```

The resulting image can be reduced in size by converting to a sparse file:

```
$ fallocate -v --dig-holes 2015-11-21-raspbian-jessie-lite.img
2015-11-21-raspbian-jessie-lite.img: 494.7 MiB (518729728 bytes) converted to sparse␣
↪holes.
```

Or compressing:

```
$ xz 2015-11-21-raspbian-jessie-lite.img
```

3.2 Booting test machine

Now that we have an image to boot, we need a kernel to run. Sadly, a modified kernel is required, since QEMU does not support raspberry pi hardware. Luckily the work of patching and building has already been done by someone else. https://github.com/polaco1782/raspberry-qemu appears to work well.

Clone a copy of that repository and copy `kernel-qemu`:

```
$ git clone git@github.com:polaco1782/raspberry-qemu.git
$ cp raspberry-qemu/kernel-qemu .
```

Now a boot should be possible. Run

```
$ qemu-system-arm -kernel kernel-qemu \
    -cpu arm1176 \
    -m 256 \
    -M versatilepb \
    -no-reboot \
    -serial stdio \
    -display none \
    -append "root=/dev/sda2 panic=1 rootfstype=ext4 rw" \
```

(continues on next page)

(continued from previous page)

```
    -net user,hostfwd=tcp::10022-:22 \
    -net nic -hda \
    2015-11-21-raspbian-jessie-lite.img
```

CHAPTER 4

Dewi's Checklist

4.1 Boat

- Dewi's Stand
- Dewi's Hull
- Mast
- Boom
- Jib Boom
- Full Sail & Jib
- Storm Sail & Jib
- Rudder
- Rudder Pin
- Rudder-Servo Linkage
- Wind Vane & Sensor
- Travel Box

4.2 Electronics

- Remote Control
- Raspberry Pi
- Micro SD Card
- Micro SD Card Converter
- Arduino Nano

- PCB (Cpt. Morgan)
- PCB Spacers
- USB WiFi Stick
- XBee (x2)
- Mini USB Cable (x2 for XBee)
- Power Supply
- Battery Charger
- Charging Cables
- Boat Batteries x8 (NiMH)
- Cpt. Morgan Power Cable
- GPS Antenna
- Deck Antennae

4.3 Spare (Boat)

- Wind Vane
- Main Sheet
- Rope
- Glands
- Rudder Servo Horn
- Rudder Servo Screw
- Linkage Nuts

4.4 Spare (Electronics)

- Raspberry Pi
- Micro SD Card
- Arduino Nano
- PCB (Cpt. Morgan)
- XBee & Cable
- 2.4GHz Antenna
- RC Receiver
- AA Batteries (can be sourced anywhere)
- Rudder Servo
- Sail Winch
- Molex Connectors (for Cpt. Morgan)
- Deans connectors

4.5 Tools

- Soldering Iron
- Solder Sucker
- Glue Gun
- Wire Cutters
- Wire Strippers
- Extension Cable
- Power Adapter (EU,US)
- Lighter
- Screwdrivers
- Multimeter
- Oscilloscope
- Knife
- Scissors

4.6 Consumables

- Solder
- Header Pins
- Vaseline
- Silicone Grease
- Silicone Sealant
- Epoxy/Araldite
- Cable Ties
- Duct Tape
- Electrical Tape
- Desoldering Wick
- Flux
- Wire
- Heat Shrink
- Glue Gun Sticks
- Cling Film (sealing travel box)

4.7 Other

- GoPro
- GoPro MicroSD Card
- GoPro Case
- GoPro Mount (for vision)
- GoPro Float
- Ethernet Cable
- Extension Lead (Foreign & UK)

CHAPTER 5

Writing docs

Docs here are written using the ReStructuredText language (similar to markdown) and rended using sphinx.

Some useful links to get started with each of these:

- http://docutils.sourceforge.net/docs/user/rst/quickstart.html
- http://docutils.sourceforge.net/docs/user/rst/quickref.html
- http://www.sphinx-doc.org/en/1.4.9/contents.html

5.1 Editing docs

These docs live here: https://github.com/abersailbot/documentation

To edit, do what you'd do with any other git repo: clone the repo and make your changes. When possible, prefer creating a pull request to allow another set of eyes to look it over.

5.2 Creating new pages

1. Create a new rst file: e.g. `doot.rst`
2. Add a title to the page
3. link to this file in `index.rst` along with the other pages:

```
.. toctree::
   :maxdepth: 2

   wiring-standard
   hardware-spec
   provisioning
   testing
   doot
```

CHAPTER 6

Indices and tables

- genindex
- search