

# AuroraWatchNet magnetometer manual

Steve R. Marple,  
Lancaster University.

2014-07-26 15:15:35 +0100  
Commit: 567050b354bf586e960f1f95976f5e993343f9bf



# Licence

This document is made available under the Creative Commons Attribution-ShareAlike 4.0 Unported Licence.



Please attribute this work as “AuroraWatchNet magnetometer manual. Steve R. Marple, Lancaster University. 2014.”

# Contents

|                                       |      |
|---------------------------------------|------|
| <b>Licence</b>                        | i    |
| <b>Contents</b>                       | ii   |
| <b>List of figures</b>                | viii |
| <b>Abbreviations</b>                  | ix   |
| <br>                                  |      |
| <b>I Introduction</b>                 | 1    |
| <b>1 Overview of the hardware</b>     | 2    |
| 1.1 Introduction . . . . .            | 2    |
| 1.2 Sensor unit . . . . .             | 2    |
| 1.3 Base unit . . . . .               | 6    |
| <br>                                  |      |
| <b>II Construction</b>                | 7    |
| <b>2 Beginning construction</b>       | 8    |
| 2.1 Anti-static precautions . . . . . | 8    |
| 2.2 Tools required . . . . .          | 8    |
| 2.3 Order of assembly . . . . .       | 8    |
| <br>                                  |      |
| <b>3 FLC100 shield assembly</b>       | 10   |
| 3.1 Introduction . . . . .            | 10   |

|   |            |
|---|------------|
| <b>CONTENTS</b>                                       | <b>iii</b> |
| 3.2 FLC100 shield version 1.0 . . . . .               | 10         |
| 3.2.1 Order of assembly . . . . .                     | 10         |
| <b>4 Calunium assembly</b>                            | <b>14</b>  |
| 4.1 Introduction . . . . .                            | 14         |
| 4.2 Calunium version 2.0 and version 2.1 . . . . .    | 14         |
| 4.2.1 Order of assembly . . . . .                     | 14         |
| 4.3 Testing the board . . . . .                       | 20         |
| 4.4 Programming the firmware . . . . .                | 20         |
| 4.4.1 Programming the bootloader . . . . .            | 20         |
| 4.4.2 Programming the magnetometer firmware . . . . . | 20         |
| 4.4.3 Generating the EEPROM settings . . . . .        | 21         |
| 4.4.4 Uploading the EEPROM settings . . . . .         | 22         |
| <b>5 Sensor PCB assembly</b>                          | <b>23</b>  |
| 5.1 Sensor PCB version 1.2 . . . . .                  | 23         |
| 5.1.1 Order of assembly . . . . .                     | 23         |
| <b>III Installation</b>                               | <b>29</b>  |
| <b>6 Site requirements</b>                            | <b>30</b>  |
| 6.1 Sensor unit requirements . . . . .                | 30         |
| 6.2 Base unit requirements . . . . .                  | 30         |
| 6.2.1 Network requirements . . . . .                  | 30         |
| <b>7 Raspberry Pi setup</b>                           | <b>32</b>  |
| 7.1 SD card creation . . . . .                        | 32         |
| 7.2 Configuring Raspbian . . . . .                    | 32         |
| 7.2.1 /dev/ttyAMA0 serial port setup . . . . .        | 33         |
| 7.2.2 Raspbian configuration . . . . .                | 33         |

|          |  |           |
|----------|--|-----------|
| 7.2.2.1  | Change user password . . . . .                             | 33        |
| 7.2.2.2  | Internationalisation options . . . . .                     | 33        |
| 7.2.2.3  | Advanced options . . . . .                                 | 33        |
| 7.2.2.4  | Expand Filesystem . . . . .                                | 33        |
| 7.2.3    | Configure proxy server . . . . .                           | 34        |
| 7.2.4    | Install missing software packages . . . . .                | 35        |
| 7.2.5    | Remove swap file . . . . .                                 | 35        |
| 7.2.6    | Configure file system mount options . . . . .              | 35        |
| 7.3      | Installing the AuroraWatchNet server software . . . . .    | 36        |
| 7.3.1    | Install the Git repository . . . . .                       | 36        |
| 7.3.2    | Install python numpy library . . . . .                     | 36        |
| 7.3.3    | Configure python . . . . .                                 | 36        |
| 7.3.4    | Configure cron . . . . .                                   | 37        |
| 7.3.5    | Configure ifplugd . . . . .                                | 37        |
| 7.3.6    | Create configuration file . . . . .                        | 38        |
| 7.3.7    | Create init file for server daemon . . . . .               | 38        |
| 7.3.8    | Configure ntp . . . . .                                    | 38        |
| <b>8</b> | <b>Installation procedure</b>                              | <b>40</b> |
| 8.0.9    | Tools required . . . . .                                   | 40        |
| 8.1      | Base unit installation . . . . .                           | 40        |
| 8.2      | Determining the maximum range for the radio link . . . . . | 40        |
| 8.2.1    | Factors which alter the maximum range . . . . .            | 41        |
| 8.3      | Installing the sensor unit . . . . .                       | 42        |
| <b>9</b> | <b>Contributing data to AuroraWatch UK</b>                 | <b>43</b> |
| 9.1      | Use of data by AuroraWatch UK . . . . .                    | 43        |
| 9.2      | Methods to upload data . . . . .                           | 43        |
| 9.2.1    | Rsync uploads . . . . .                                    | 44        |

|  |           |
|--|-----------|
| 9.2.2 HTTP uploads . . . . .                                   | 45        |
| <b>IV Operation</b>  | <b>46</b> |
| <b>10 Raspberry Pi operation</b>                               | <b>47</b> |
| 10.1 Introduction . . . . .                                    | 47        |
| 10.2 Shutting down the Raspberry Pi . . . . .                  | 47        |
| 10.3 Starting and stopping the data recording daemon . . . . . | 47        |
| 10.4 Monitoring the data recording process . . . . .           | 48        |
| <b>11 Software and firmware updates</b>                        | <b>49</b> |
| 11.1 Raspbian updates . . . . .                                | 49        |
| 11.2 AuroraWatchNet software updates . . . . .                 | 49        |
| 11.3 Sensor unit firmware updates . . . . .                    | 49        |
| <b>A Configuration file options</b>                            | <b>50</b> |
| A.1 Introduction . . . . .                                     | 50        |
| A.2 [DEFAULT] . . . . .  | 50        |
| A.2.1 site . . . . .   | 50        |
| A.3 [awnettextdata] . . . . .                                  | 51        |
| A.3.1 filename . . . . .                                       | 51        |
| A.4 [awpacket] . . . . .                                       | 51        |
| A.4.1 filename . . . . .                                       | 51        |
| A.4.2 key . . . . .  | 52        |
| A.5 [logfile] . . . . .  | 52        |
| A.5.1 filename . . . . .                                       | 52        |
| A.6 [daemon] . . . . .   | 52        |
| A.6.1 connection . . . . .                                     | 52        |
| A.6.2 close_after_write . . . . .                              | 52        |

|        |                                |    |
|--------|--------------------------------|----|
| A.6.3  | acknowledge . . . . .          | 53 |
| A.6.4  | read_only . . . . .            | 53 |
| A.7    | [serial] . . . . .             | 53 |
| A.7.1  | port . . . . .                 | 53 |
| A.7.2  | baudrate . . . . .             | 53 |
| A.7.3  | setup . . . . .                | 54 |
| A.8    | [ethernet] . . . . .           | 54 |
| A.8.1  | local_address . . . . .        | 54 |
| A.8.2  | local_port . . . . .           | 54 |
| A.8.3  | remote_address . . . . .       | 54 |
| A.8.4  | remote_port . . . . .          | 54 |
| A.9    | [controlsocket] . . . . .      | 54 |
| A.9.1  | filename . . . . .             | 55 |
| A.9.2  | port . . . . .                 | 55 |
| A.10   | [magnetometer] . . . . .       | 55 |
| A.10.1 | siteid . . . . .               | 55 |
| A.10.2 | key . . . . .                  | 55 |
| A.11   | [upload] . . . . .             | 56 |
| A.11.1 | method . . . . .               | 56 |
| A.11.2 | rsync_host . . . . .           | 56 |
| A.11.3 | url . . . . .                  | 56 |
| A.11.4 | password . . . . .             | 56 |
| A.11.5 | realm . . . . .                | 57 |
| A.12   | [realtime_transfer] . . . . .  | 57 |
| A.12.1 | remote_host . . . . .          | 57 |
| A.12.2 | remote_port . . . . .          | 57 |
| A.12.3 | remote_key . . . . .           | 57 |
| A.13   | [dataqualitymonitor] . . . . . | 57 |

|                            |    |
|----------------------------|----|
| A.13.1 port . . . . .      | 58 |
| A.13.2 filename . . . . .  | 58 |
| A.13.3 extension . . . . . | 58 |
| A.13.4 pidfile . . . . .   | 58 |
| A.13.5 logfile . . . . .   | 58 |

# List of figures

|     |  |    |
|-----|--|----|
| 1.1 | System overview  | 3  |
| 1.2 | Sensor unit  | 3  |
| 1.3 | Calunium microcontroller PCB                                 | 4  |
| 1.4 | FLC100 shield  | 5  |
| 1.5 | Calunium microcontroller board and FCL100 shield             | 5  |
| 1.6 | Base unit  | 6  |
| 3.1 | Completed FLC100 shield                                      | 11 |
| 3.2 | Modification to monitor sleep status of the XRF radio module | 12 |
| 3.3 | FLC100 shield v. 1.0 circuit diagram.                        | 13 |
| 4.1 | Completed Calunium v2.1                                      | 15 |
| 4.2 | LED orientation  | 17 |
| 4.3 | Real-time clock load capacitors (Calunium v 2.0)             | 17 |
| 4.4 | Calunium v. 2.0 circuit diagram.                             | 18 |
| 4.5 | Calunium v. 2.1 circuit diagram.                             | 19 |
| 5.1 | Completed sensor PCB (single-axis)                           | 24 |
| 5.2 | Completed sensor PCB (three-axis)                            | 25 |
| 5.3 | Fit the male turned-pin headers.                             | 25 |
| 5.4 | Fit the female turned-pin sockets.                           | 26 |
| 5.5 | Place the sensor PCB onto the female turned-pin sockets.     | 26 |
| 5.6 | Sensor PCB version 1.2 circuit diagram.                      | 28 |

# Abbreviations

|        |  |
|--------|--|
| ADC    | analogue to digital converter                                |
| API    | application programming interface                            |
| DHCP   | dynamic host configuration protocol                          |
| DIP    | dual-inline package  |
| DNS    | domain name system   |
| EEPROM | electrically erasable programmable read-only memory          |
| ESD    | electro-static discharge                                     |
| FAT    | file allocation table  |
| FET    | field-effect transistor                                      |
| GPU    | graphics processing unit                                     |
| HMAC   | hash-based message authentication code                       |
| HTTP   | hypertext transfer protocol                                  |
| HTTPS  | hypertext transfer protocol secure                           |
| I2C    | inter-integrated circuit (bus)                               |
| IC     | integrated circuit   |
| IP     | internet protocol  |
| ISM    | Industrial, scientific and medical (radio band)              |
| ISP    | in-circuit serial programmer (sometimes abbreviated as ICSP) |
| JTAG   | joint test action group                                      |
| LED    | light emitting diode   |
| MD5    | message digest 5   |
| NFS    | network file system  |
| NTP    | network time protocol  |
| PCB    | printed circuit board  |
| PoE    | power over ethernet  |
| RFI    | radio-frequency interference                                 |
| RTC    | real-time clock  |
| SD     | secure digital   |
| SOIC   | small outline integrated circuit                             |
| SSH    | secure shell   |
| TCP    | transmission control protocol                                |
| TTL    | transistor-transistor logic                                  |
| UDP    | user datagram protocol                                       |
| URL    | uniform resource locator                                     |
| USB    | universal serial bus   |
| UTC    | coordinated universal time                                   |



# **Part I**

## **Introduction**

# **Chapter 1**

## **Overview of the hardware**

### **1.1 Introduction**

The magnetometer is designed for low-power operation, simple installation and ease of construction. The entire design is open source, allowing anyone with reasonable soldering ability to construct one.

The magnetometer has two major parts, the base unit and the sensor unit (Figure 1.1). The sensor unit is located outdoors, away from buildings, cars and other sources of human disturbance. It is battery powered and communicates with the base unit by a radio link (433 MHz or 868 MHz), enabling the sensor to be installed without any wiring to the base unit. The base unit is placed indoors and should be positioned such that there are the minimum number of walls between it and the sensor unit.

### **1.2 Sensor unit**

The sensor unit (figure 1.2) is contained inside a waterproof enclosure approximately 1.1 m high which is partially buried to reduce temperature variations and to provide a stable foundation. The sensor itself is placed at the bottom of the enclosure, approximately 0.85 m below ground. The microcontroller, radio module and battery are positioned in the top part of the enclosure, above ground level. Insulating material (e.g. rockwool) is used to fill the space in-between.

The Calunium microcontroller board is based on the popular Arduino platform but uses the more powerful Atmel ATmega1284P microcontroller.

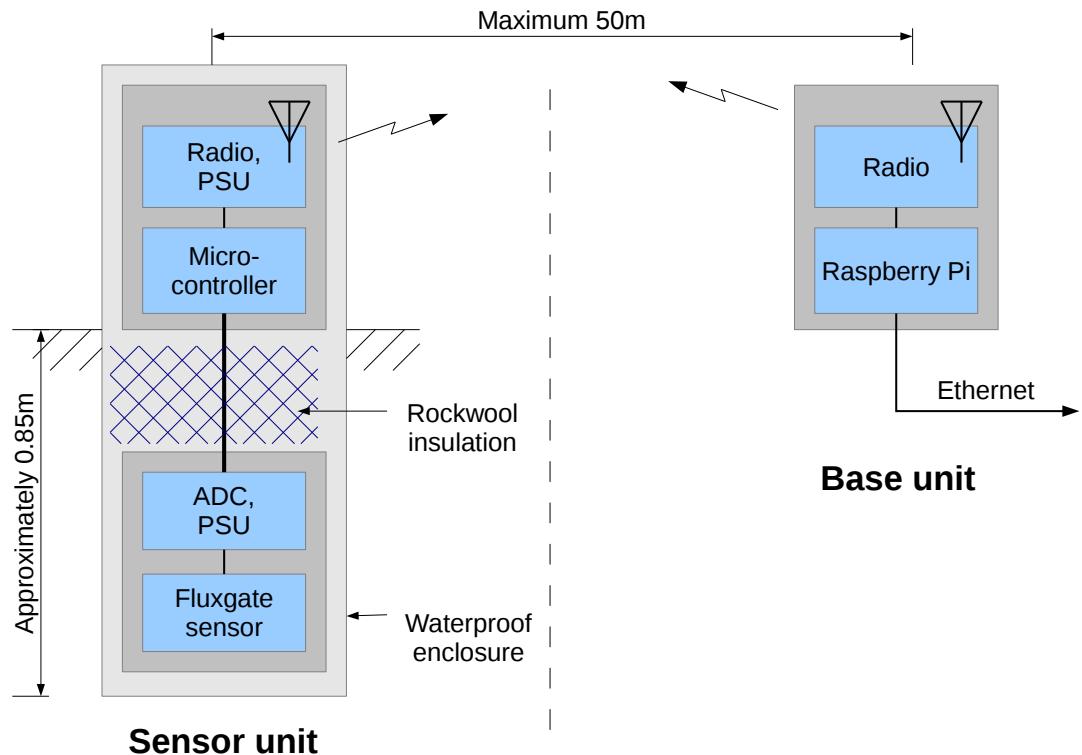


Figure 1.1: System overview.



Figure 1.2: Sensor unit.

© Steve Marple. CC BY-SA 2.0. <http://www.flickr.com/photos/stevemarple/8499204572/>

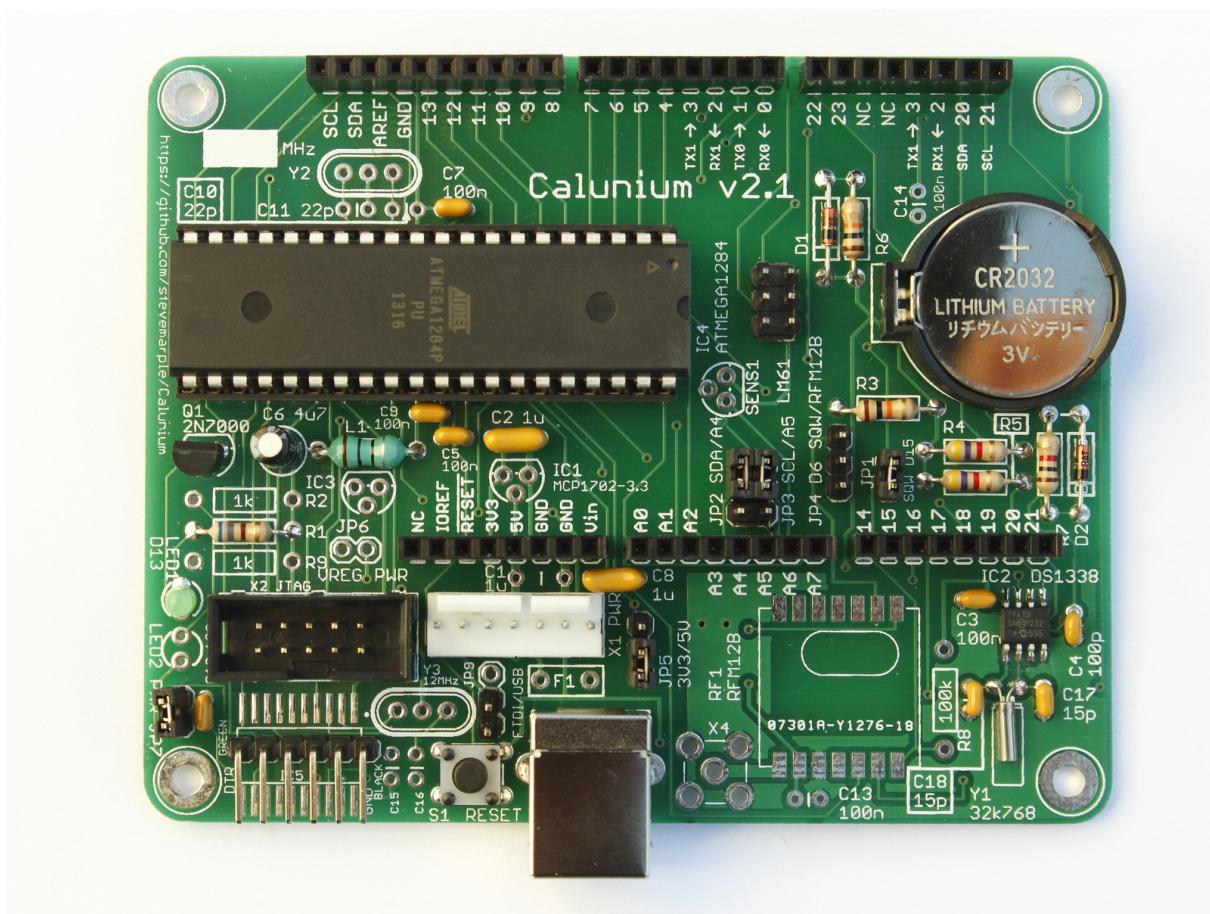


Figure 1.3: Calunium microcontroller PCB.

© Steve Marple. CC BY-SA 2.0. <http://www.flickr.com/photos/stevemarple/10786865096/>

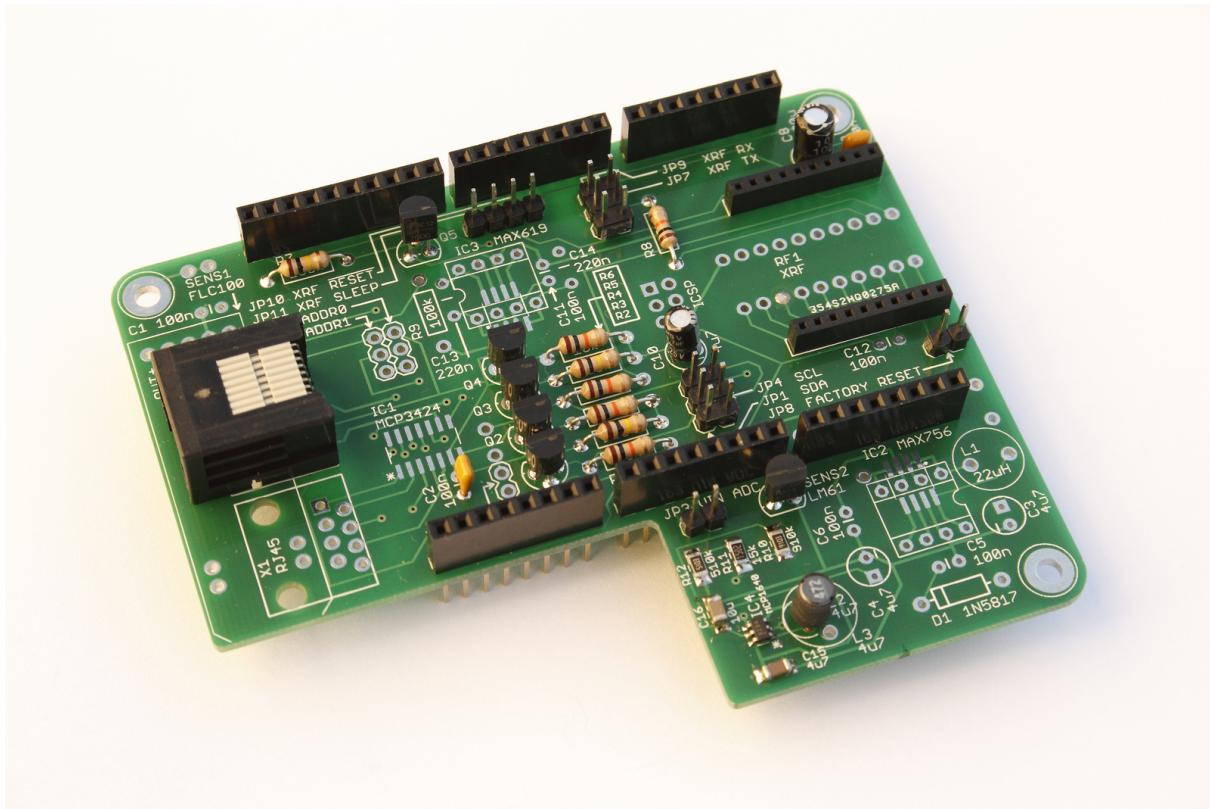


Figure 1.4: The FLC100 shield fits onto the Calunium microcontroller PCB.  
© Steve Marple. CC BY-SA 2.0. <http://www.flickr.com/photos/stevemarple/10787109594/>

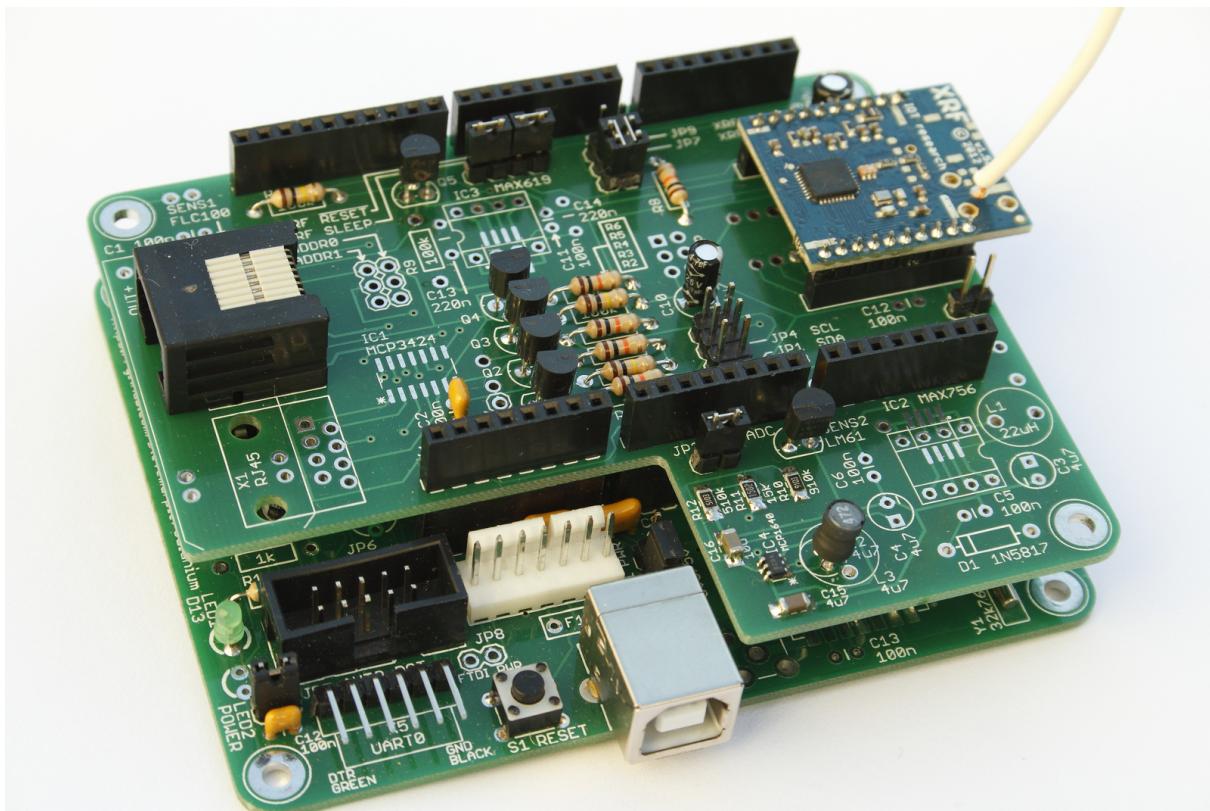


Figure 1.5: Calunium microcontroller board and FCL100 shield.  
© Steve Marple. CC BY-SA 2.0. <http://www.flickr.com/photos/stevemarple/10913562526/>

### 1.3 Base unit

The base unit is a Raspberry Pi single-board computer with a radio transceiver unit. The Ethernet interface of the Raspberry Pi is used to send the magnetic field measurements to AuroraWatch UK. When the Raspberry Pi is accessed over the network with Secure Shell (SSH) a display and keyboard are not needed. The Raspberry Pi runs the Raspbian linux distribution. The receiving software is written in Python.



Figure 1.6: Base unit.

© Steve Marple. CC BY-SA 2.0. <http://www.flickr.com/photos/stevemarple/10787215844/>

## **Part II**

# **Construction**

# Chapter 2

## Beginning construction

### 2.1 Anti-static precautions

### 2.2 Tools required

- Soldering iron.
- Side cutters.
- Small pliers.
- In-circuit serial programmer (ISP) for Atmel AVR microcontrollers. Instructions are given for the Atmel AVR Dragon but other programmers can be used.
- USB to TTL serial converter for 3.3 V operation, e.g., FTDI TTL-232R-3V3.
- Digital multimeter.
- Solderless breadboard (optional).

### 2.3 Order of assembly

For ease of access components should normally be fitted in order of increasing size, particularly increasing height. If this order is not observed it can be very difficult to access the pads of surface mount devices. It is also preferable that *passive* components (resistors, capacitors, inductors and crystals) are fitted before semiconductors (field-effect transistors, integrated circuits). This is because the semiconductors are easily damaged by electro-static discharge (sometimes this damage isn't immediately obvious). It is therefore more convenient to fit as many components as possible before fitting the semiconductors, at which point ESD precautions should be followed. As field-effect transistors are particularly vulnerable to damage by ESD it is recommended they are fitted as late as possible. From these guidelines the following order is recommended.

- Surface-mount passive components.
- Surface-mount semiconductors.
- Through-hole passive components.
- Through-hole semiconductors (FETs last).
- Switches.
- Connectors, battery holders.

The first PCB to be assembled is the FLC100 shield, this board provides the power to the system and will enable you to test each part correctly.

# **Chapter 3**

## **FLC100 shield assembly**

### **3.1 Introduction**

The FLC100 shield is an Arduino “shield” which operates at 3.3 V. Do not attempt to use it with standard Arduino boards which are operated at 5 V. The shield houses the XRF radio module, the boost power supply (which creates the 3.3 V supply for the microcontroller and radio) and the 3.3 V – 5 V level shifters.

There are both through-hole and surface mount versions of the boost power supply. It is suspected that the through-hole version causes RFI since the radio module often fails to receive messages. This problem was not apparent on the prototype board. The surface-mount version has no such problems and is the option which should be used. It is also cheaper and more efficient.

There is an option to fit a FLC100 sensor directly to the circuit board. This option is not used because the FLC100 is slightly temperature sensitive and better performance is obtained by positioning the sensor below ground. Whilst the board provides an option to fit an MCP3424 ADC and MAX619 charge-pump power supply they are also fitted remotely, below ground, for reasons of temperature stability.

### **3.2 FLC100 shield version 1.0**

#### **3.2.1 Order of assembly**

1. IC4 (MCP1640).
2. R12 (510 k $\Omega$ ).
3. R11 (15 k $\Omega$ ).
4. R10 (910 k $\Omega$ ).
5. 2 mm 10 way connectors for RF1. Ensure they are fitted flush to the PCB.
6. C15 (4.7  $\mu$ F).
7. C16 (10  $\mu$ F).

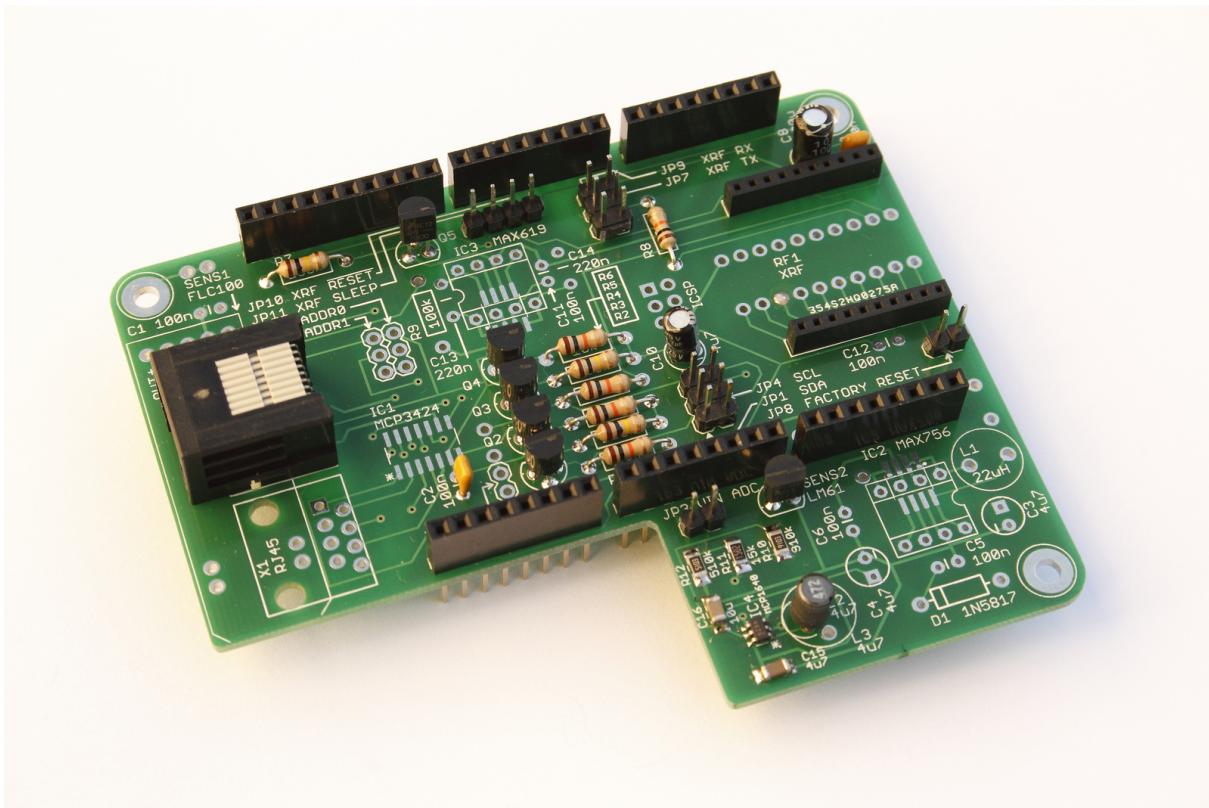


Figure 3.1: Completed FCL100 shield, except for fitting shunts onto the jumpers.

© Steve Marple. CC BY-SA 2.0. <http://www.flickr.com/photos/stevemarple/10787109594/>

8. R1, R3, R4, R6, R8 ( $10\text{ k}\Omega$ ).
9. R2, R5, R7 ( $100\text{ k}\Omega$ ).
10. C2, C7, C9 ( $100\text{ nF}$ ).
11. C10 ( $4.7\text{ }\mu\text{F}$ ).
12. C8 ( $100\text{ }\mu\text{F}$ ).
13. L2 ( $4.7\text{ }\mu\text{H}$ ). The shorter lead should be connected to pin 1, which is the hole nearest the edge of the PCB. Although the orientation of inductors is normally ignored communication with the manufacturer revealed that the shorter lead indicates the start of the winding. This arrangement is preferred to help minimise RFI.
14. Stacking connectors, five 8 way and one 10 way. Ensure they are fitted flush to the PCB; solder one end first, then the other end. Only when you are happy they are flush should you solder the remaining pins.
15. JP1, JP4 ( $2 \times 3$  jumper).
16. JP7, JP9 ( $2 \times 3$  jumper).
17. JP10, JP11 ( $1 \times 4$  jumper).
18. JP3 ( $1 \times 2$  jumper).
19. JP8 ( $1 \times 2$  jumper).
20. X2, RJ45 connector.
21. Modify the PCB by adding a link wire from the XRF ONSLEEP status pin to D23. See figure 3.2.
22. Q1, Q2, Q3, Q4, Q5 (2N7000).

Fit shunts to JP10, JP11, JP3. Fit shunts to JP7 and JP9, to the two connectors furthest from the

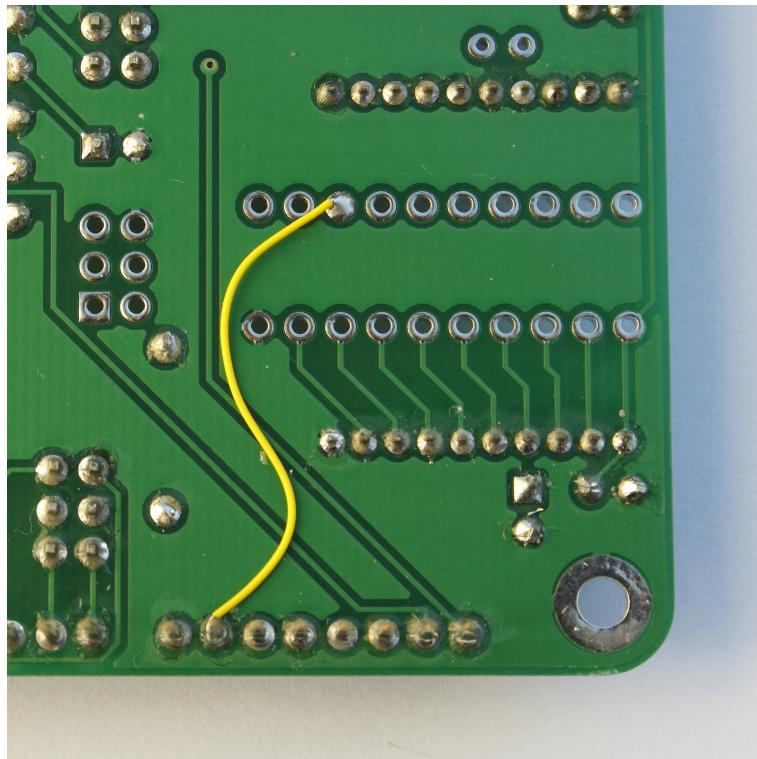


Figure 3.2: Modification to monitor sleep status of the XRF radio module.

© Steve Marple. CC BY-SA 2.0. <http://www.flickr.com/photos/stevemarple/10786910376/>

edge of the PCB. **Do not fit the shunts so that they bridge between JP7 and JP9.**

If the microcontroller board has dedicated I2C connections (e.g. Calunium v2.0 or later), then no shunts should be fitted to JP1 and JP4. If the microcontroller has I2C connections in the standard Arduino Mega positions (e.g., Calunium v1.0, Arduino Mega, Arduino Mega 2560) then fit shunts to JP1 and JP4 to the two connectors closest to C10, otherwise fit shunts to JP1 and JP4 to the two connectors closest to the stacking connectors. **In no circumstances should the shunts bridge between JP1 and JP4.**

Leave JP8 open circuit. It is fitted only when the XRF1 radio module must be forced to use the default factory configuration.

### 3.2. FLC100 SHIELD VERSION 1.0

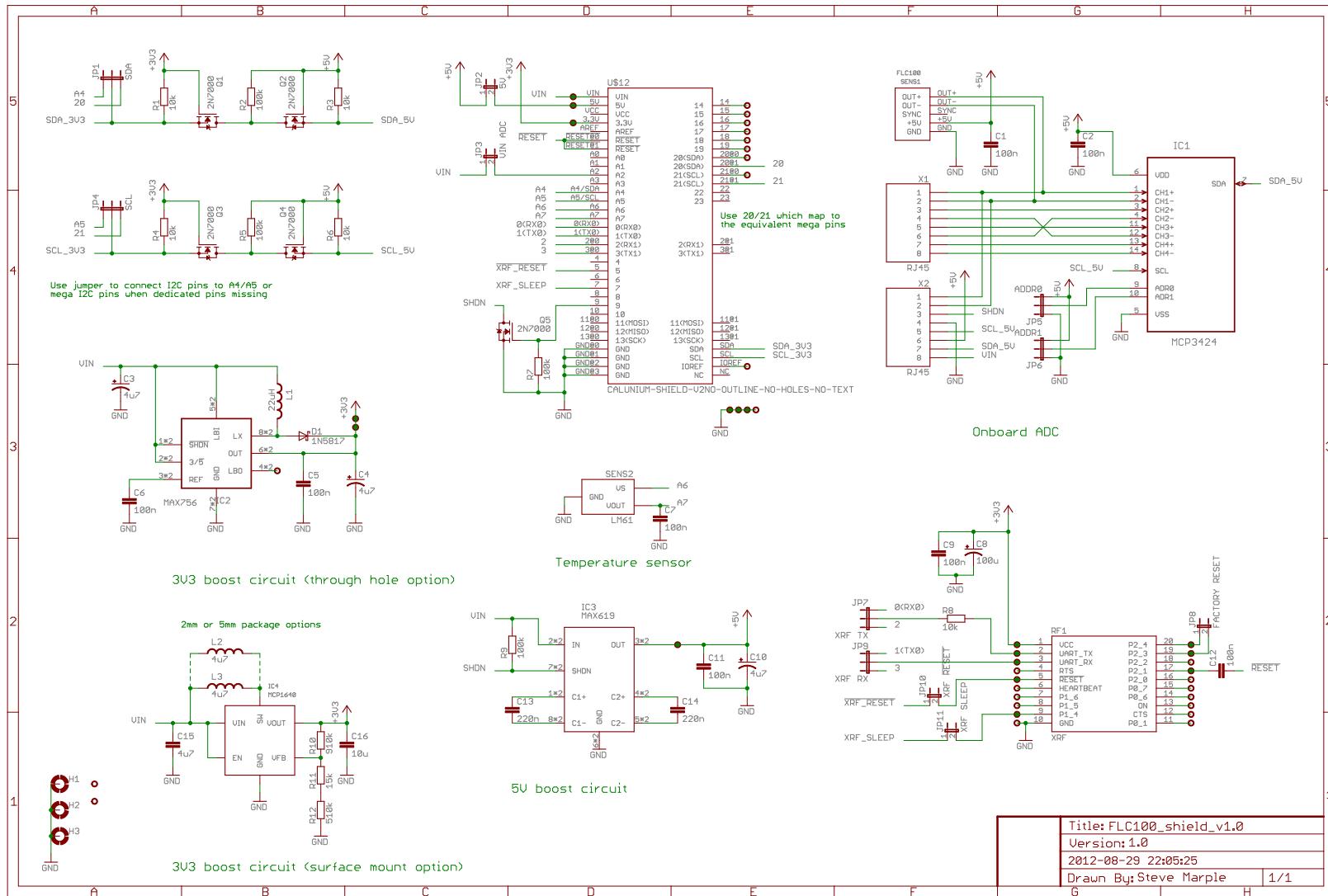


Figure 3.3: FLC100 shield v. 1.0 circuit diagram.

# Chapter 4

## Calunium assembly

### 4.1 Introduction

The Calunium microcontroller development board is intended to be a flexible system for both development and embedded use. As such it has various hardware options and careful attention must be paid to assembling it for optimum performance. Parts which are not needed are omitted to lower power consumption (e.g., power LED, USB controller).

### 4.2 Calunium version 2.0 and version 2.1

#### 4.2.1 Order of assembly

Fit components in order:

1. IC2. The standard real-time clock is the MicroChip MCP90410 but MicroChip MCP79411 or MCP79412 can be used without any other changes. It is also possible to fit the Maxim DS1338-33 real-time clock, but see below for changes.
2. Y1 (32.768 kHz).
3. R1, fit a  $680\ \Omega$  resistor. Ignore the  $1\ k\Omega$  marking; a lower value resistor is used to enable the green LED to be seen more clearly in daylight.
4. R7 ( $1\ k\Omega$ ). Do not fit if using the DS1338-33 real-time clock. Instead link between R7 and D2 at the end nearest the RTC battery, as indicated by the white line on the silkscreen. The wire will bypass both R7 and D2 which are not required for the DS1338-33.
5. R4, R5 ( $4.7\ k\Omega$ ).
6. R3, R6 ( $10\ k\Omega$ ).
7. L1 ( $10\ \mu\text{H}$ ).
8. 40 pin socket for IC4.
9. C4 ( $100\ \text{pF}$ ).
10. C3, C5, C7, C9, C12 ( $100\ \text{nF}$ ).
11. C2, C8 ( $1\ \mu\text{F}$ ).

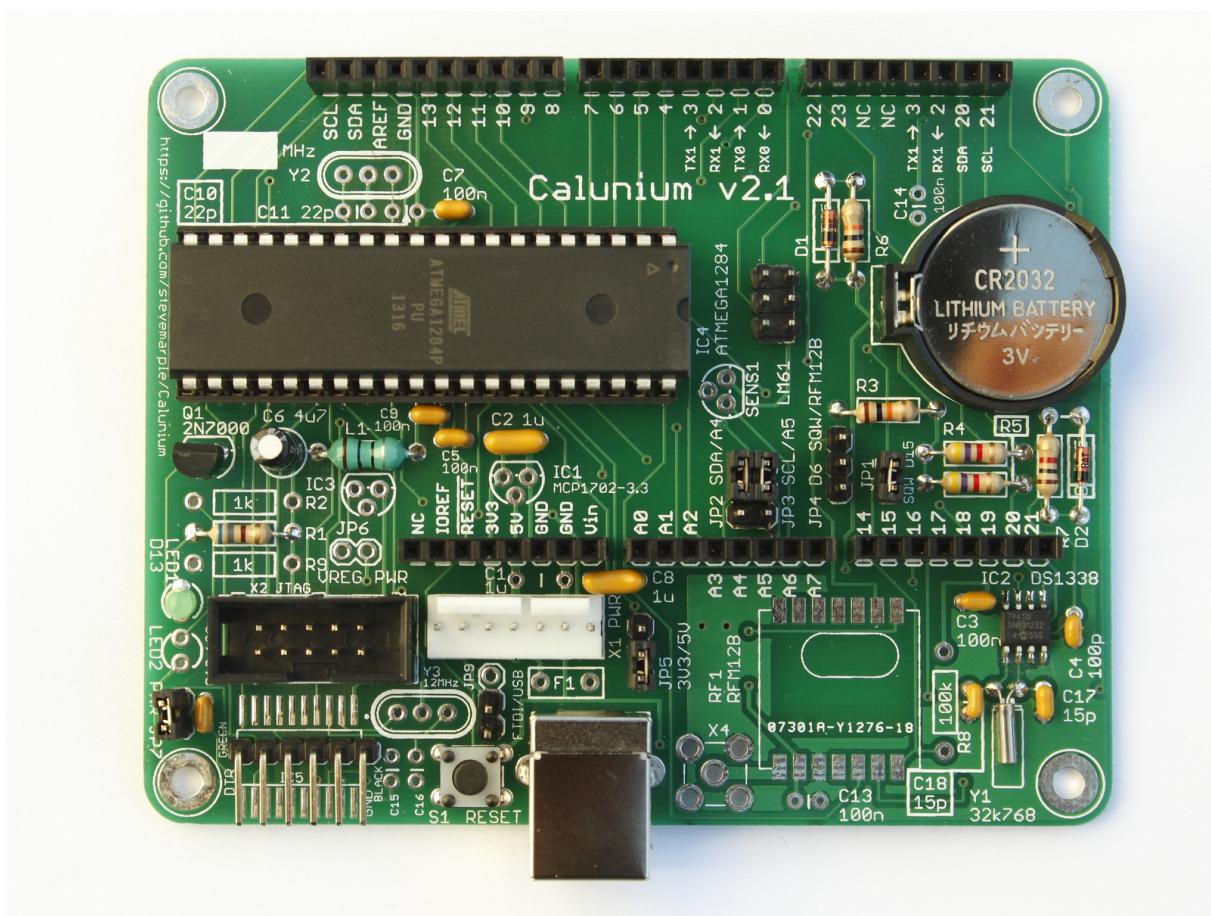


Figure 4.1: Completed Calunium v2.1.

© Steve Marple. CC BY-SA 2.0. <http://www.flickr.com/photos/stevemarple/10786865096/>

12. D1, D2 (BAT85). Do not fit D2 if using the DS1338-33 real-time clock.
13. LED1 (green LED). The cathode is nearest LED2, see figure 4.2.
14. C6 (4.7  $\mu$ F).
15. ICSP header ( $2 \times 3$  jumper block). See Figure ??.
16. JP2 and JP3. Fit as combined  $2 \times 3$  jumper block.
17. JP1, JP7 ( $1 \times 2$  jumper).
18. JP4, JP5 ( $1 \times 5$  jumper).
19. X5 ( $1 \times 6$  right-angle or vertical header for UART0).
20. S1 (reset switch).
21. Arduino headers. **TO DO: Add description**
22. C17, C18 (15 pF). Do not fit if using DS1338-33 RTC. For Calunium version 2.0 the capacitors must be fitted on the reverse side of the board (see Figure 4.3 as no specific mounting holes exist (error caused by using an earlier, incorrect datasheet which did not show the load capacitors)).
23. X1 (Molex power header). Ensure correct orientation, with the backplate closest to the Arduino headers.
24. **TO DO: Add component name** (RTC battery holder).
25. Q1 (2N7000). This item is very sensitive to damage by electrostatic discharge!
26. Battery (CR2032). Check that the battery backup pin (3) on the RTC measures 3.0 V.
27. **TO DO: Fit shunts to jumpers ...**

Do not fit the ATmega1284P microcontroller until after testing the board power supplies.

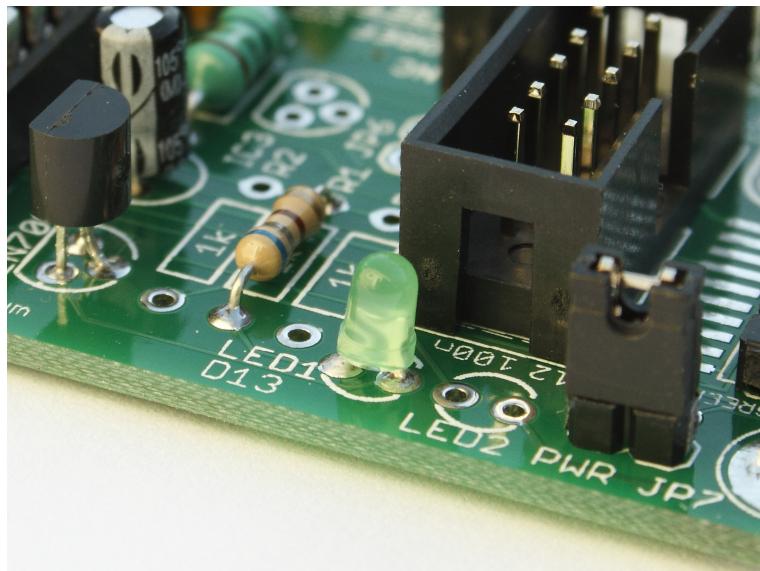


Figure 4.2: LED orientation.

© Steve Marple. CC BY-SA 2.0. <http://www.flickr.com/photos/stevemarple/10786846715/>

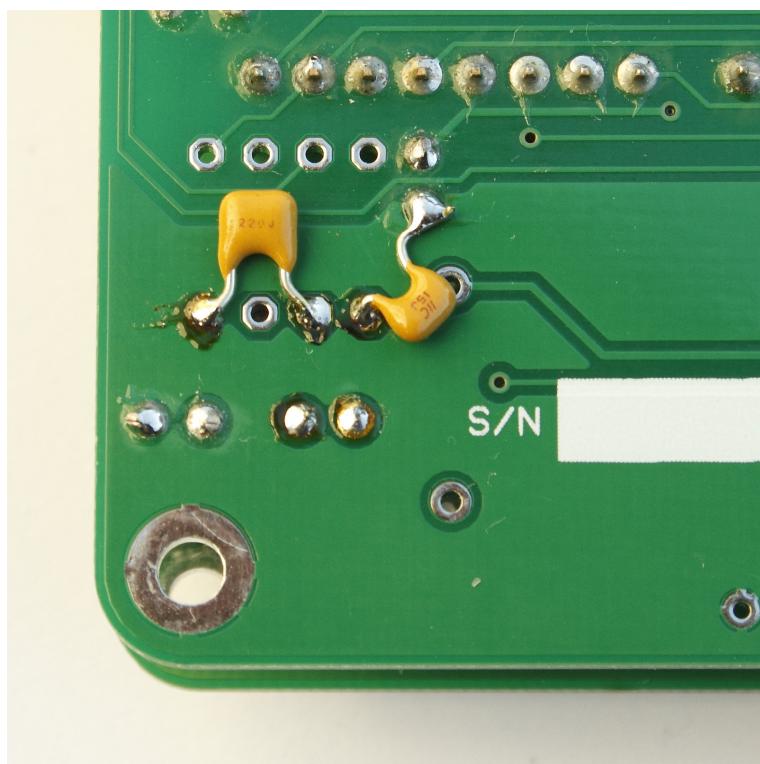


Figure 4.3: Real-time clock load capacitors (Calunium v 2.0).

© Steve Marple. CC BY-SA 2.0. <http://www.flickr.com/photos/stevemarple/10787041263/>

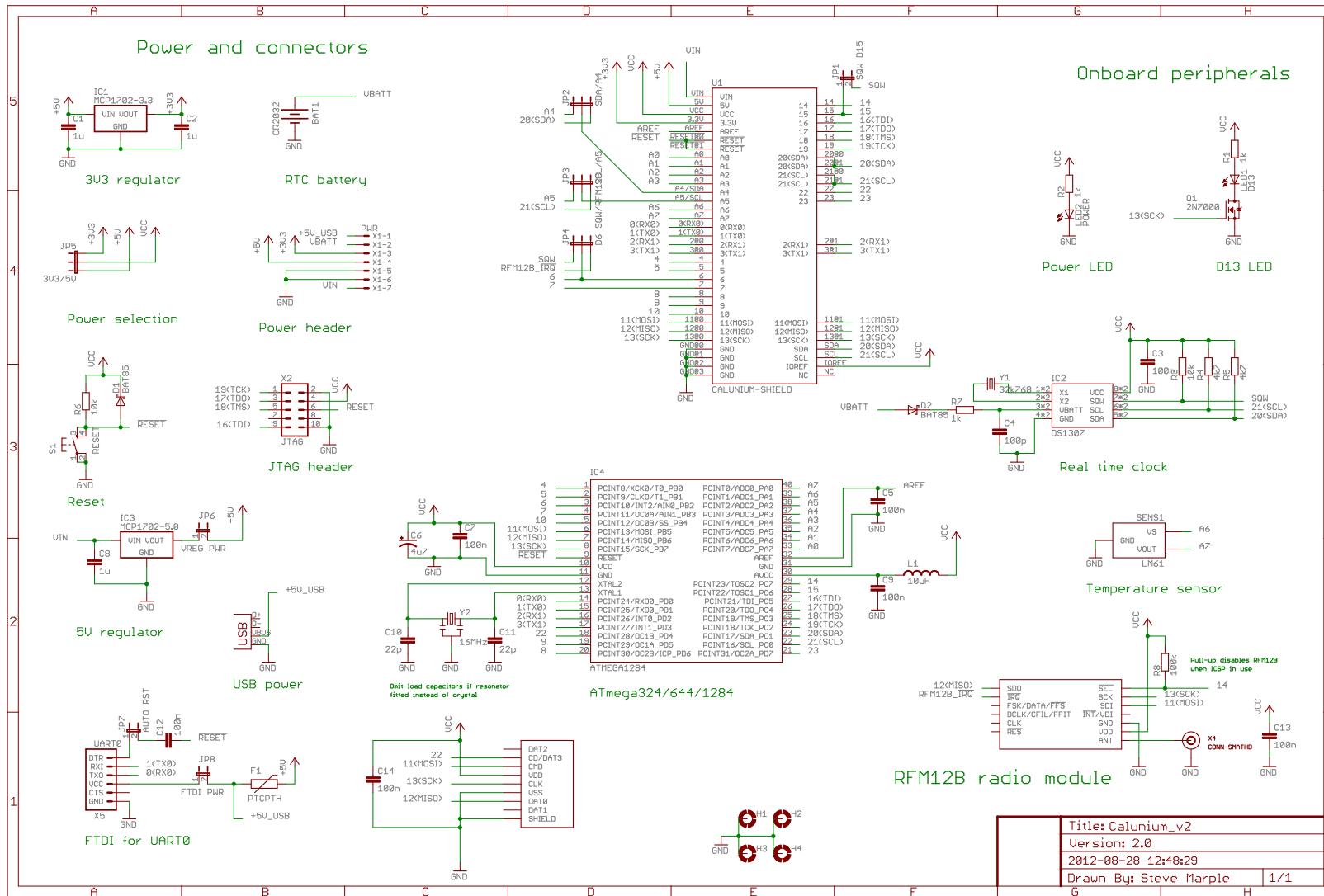


Figure 4.4: Calunium v. 2.0 circuit diagram.

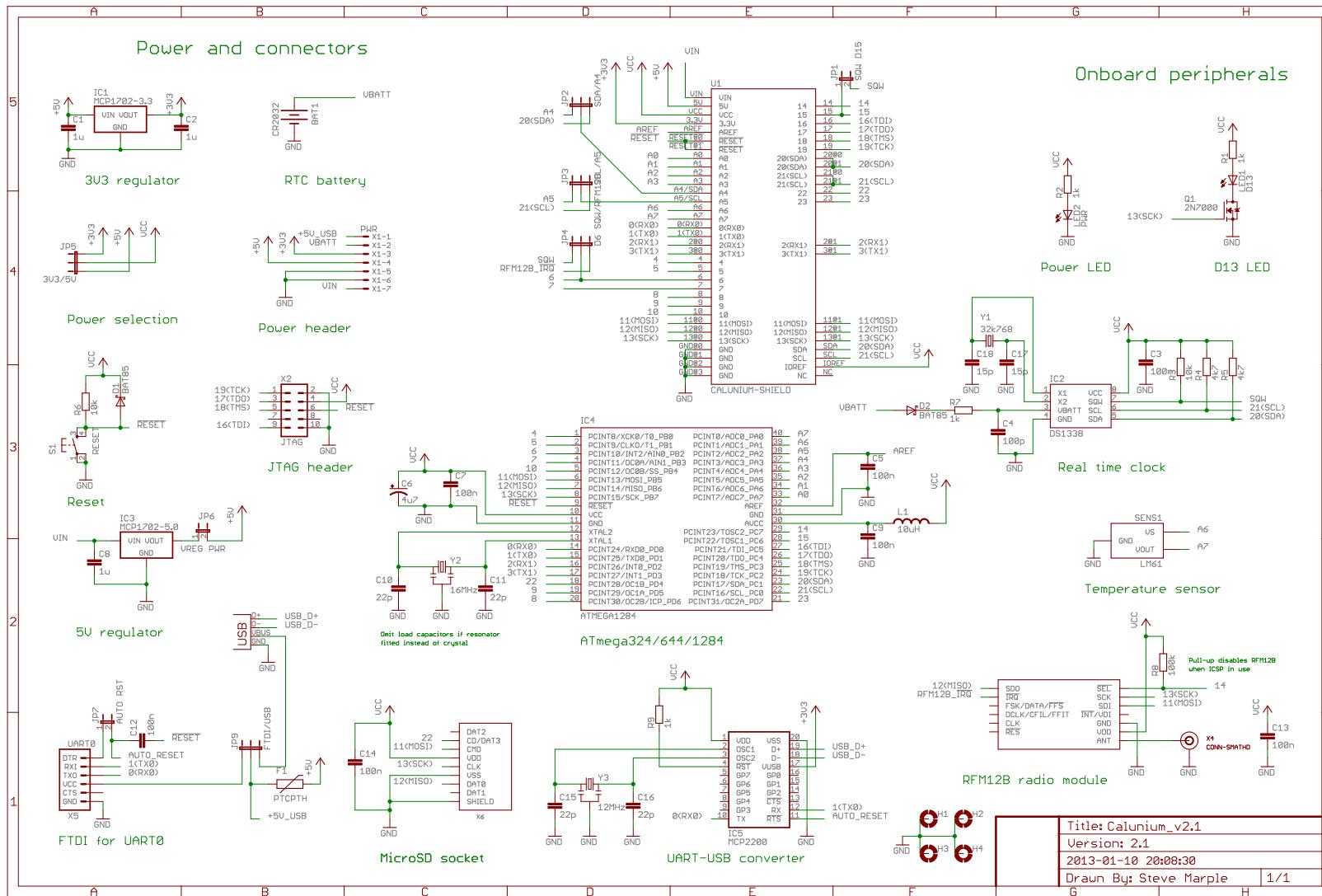


Figure 4.5: Calunium v. 2.1 circuit diagram.

## 4.3 Testing the board

## 4.4 Programming the firmware

These instructions assume you are using the Atmel AVR Dragon in ISP mode but adapting them to suit your programmer should be straightforward; see the `avrdude` manual page for further information.

### 4.4.1 Programming the bootloader

Power up the Calunium board and connect the programmer. Ensure the cable is correctly orientated at both ends. The bootloader can be compiled and programmed simply, as user pi:

**TO DO:** Check directory

```
cd /home/pi/xboot
make clean
make SHELL=bash calunium_8MHz_RC_ISP.conf.mk program
```

**TO DO:** ignore lock bit verification errors.

If the bootloader is correctly programmed the green LED connected to D13 on the Calunium PCB should flash at about 1 Hz.

### 4.4.2 Programming the magnetometer firmware

Ensure that the shunt marked “AUTO RST” is fitted, and that the shunt marked “FTDI PWR” is omitted. Connect the FTDI cable and identify the USB device file; as user pi:

```
dmesg | tail
```

Look for a line containing text similar to

```
FTDI USB Serial Device converter now attached to ttyUSB0
```

For the case above the device file `/dev/ttyUSB0`. Now program the microcontroller using the `xboot` bootloader. Replace `/dev/ttyUSB0` with the device file one your system. As user pi:

```
cd /home/pi/AuroraWatchNet/firmware/magnetometer
avrduude -p atmega1284p -b 38400 -c avr109 -P /dev/ttyUSB0 \
-U flash:w:xrf_rf12-0.10a.bin:r
```

- Whilst it is possible to program the firmware using the AVR Dragon alone this approach ensures that the xboot bootloader is present and functions correctly, allowing the microcontroller firmware to be updated over the radio link.

#### 4.4.3 Generating the EEPROM settings

Correct operation requires that the settings and the HMAC-MD5 key used for signing messages are uploaded to the EEPROM in the ATmega1284P microcontroller. To generate the EEPROM image use the `generate_eeprom_image` program, as user `pi`:

```
/home/pi/AuroraWatchNet/software/server/awndt/generate_eeprom_image.py \
--file /home/pi/eeprom --site-id XX --sampling-interval-16ths 480 \
--max-message-no-ack 60 --max-messages-led 30 --num-samples 15 \
--aggregate 1 --all-samples 0 --radio-type 0 --radio-xrf-channel 25 \
--mcp7941x-cal 0 --use-sd 0
```

Replace `XX` with the site ID for your site. AuroraWatch UK maintains a list of unique site IDs; zero can be used for testing.

These settings will cause the magnetometer to make measurements every 30 s.

- Some settings are defined in terms of samples, rather than time. If the sample interval is altered the effective interval for these settings will also change. If the number of messages without an acknowledgement exceeds that defined by `--max-message-no-ack` (60 in the above example) then the microcontroller is rebooted in the hope of recovering communication. Similarly, the number of messages before the LED is switched off (see section 8.2) is defined by `--max-messages-led` (30 in the example above).

These settings are defined by number of messages to avoid problems should the system time be automatically corrected.

- The generate\_eeprom\_image.py program generates a new random HMAC-MD5 key each time it is run. Use the --hmac-key option to preserve a previous HMAC-MD5 key setting.

#### 4.4.4 Uploading the EEPROM settings

The EEPROM settings defined in the previous section can be uploaded directly, using ISP or JTAG programmers, or indirectly via the xboot bootloader. The upload can be performed as user `pi`, but if problems accessing the programmer device file is experienced it can be uploaded as user `root`.

To upload using the AVR Dragon in ISP mode:

```
avrdude -P usb -p atmega1284p -c dragon_isp \
-U eeprom:w:/home/pi/eeprom.bin:r
```

To upload using the AVR Dragon in JTAG mode:

```
avrdude -P usb -p atmega1284p -c dragon_jtag \
-U eeprom:w:/home/pi/eeprom.bin:r
```

To upload via the xboot bootloader:

```
avrdude -p atmega1284p -P /dev/ttyUSB0 -c avr109 -b 38400 \
-U eeprom:w:/home/pi/eeprom.bin:r
```

If the FTDI adaptor is connected to a different device file substitute `/dev/ttyUSB0` with the correct device file.

# Chapter 5

## Sensor PCB assembly

### 5.1 Sensor PCB version 1.2

#### 5.1.1 Order of assembly

Fit components in order:

1. Turned pin sockets for the FLC100 sensor. Accurate alignment is important so use a piece of solderless breadboard to hold the male turned-pin headers (Figure 5.3). Insert the headers so that the conical part is pointing downwards. Fit the upside down turned-pin sockets onto the headers (Figure 5.4). Place the PCB onto the upside-down sockets (Figure 5.5) and solder all 7 connections. Remove from the breadboard, leaving the male headers in place. Carefully position the FLC100 sensor onto the male turned-pin headers. The top-side of the FLC100 has two yellow capacitors and the letters BS the circuit board. Solder the FLC100 sensor to the header. Gently remove the FLC100 sensor and place in an anti-static bag.
2. IC1 (MCP3424).
3. IC2 (MAX619) if using the SOIC option.
4. R1, R2, R3 ( $10\text{ k}\Omega$ ).
5. R4 ( $100\text{ k}\Omega$ ).
6. R5 ( $4.7\text{ k}\Omega$ ).
7. C1, C2, C5, C8 ( $100\text{ nF}$ ).
8. C9 ( $10\text{ nF}$ ).
9. C6, C7 ( $220\text{ nF}$ ).
10. C3, C4 ( $4.7\text{ }\mu\text{F}$ ).
11. D1 (BAT85).
12. IC socket for IC2 if using the DIP option.
13. SENS2 (LM61).
14. JP5 and JP7 (fit as  $2 \times 3$  male header).
15. X1 (RJ45 vertical jack).
16. Fit IC2 (MAX619) into its IC socket if using the DIP option.
17. Q1 (2N7000). This item is very sensitive to damage by electrostatic discharge! This item

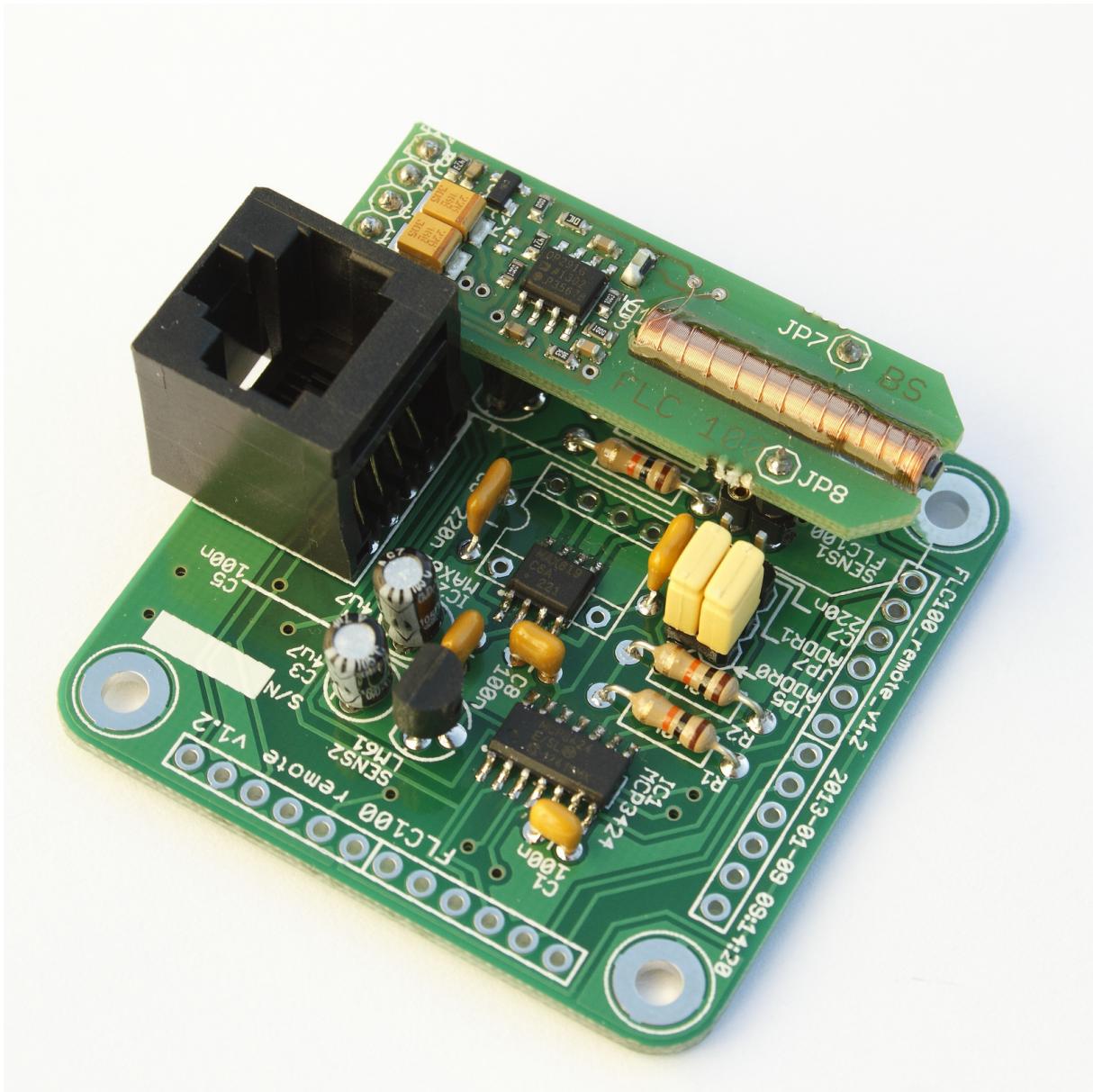


Figure 5.1: Completed sensor PCB (single-axis).

© Steve Marple. CC BY-SA 2.0. <http://www.flickr.com/photos/stevemarple/10787290503/>

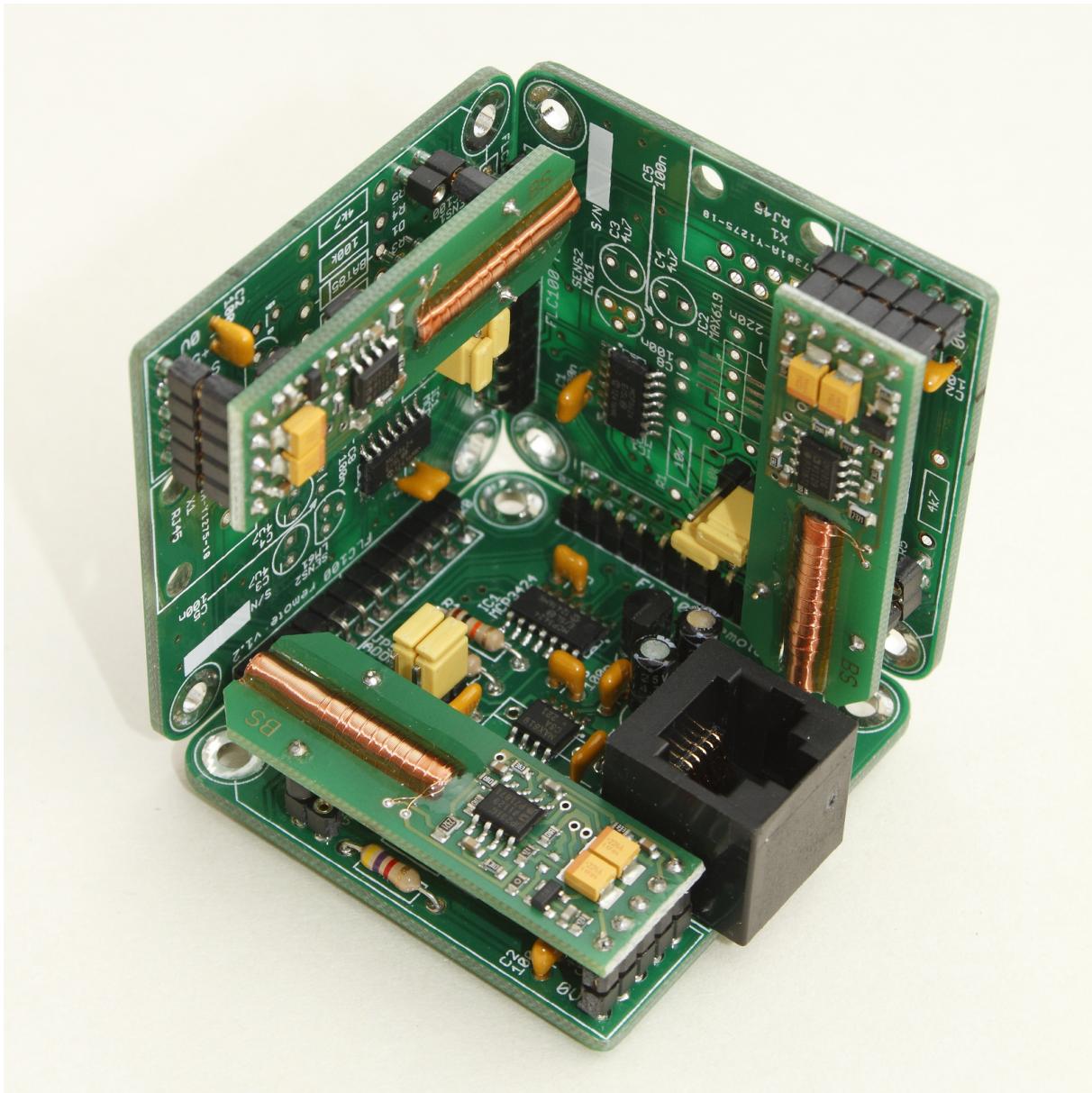


Figure 5.2: Completed sensor PCB (three-axis).

© Steve Marple. CC BY-SA 2.0. <http://www.flickr.com/photos/stevemarple/8521787269/>

TO DO: Take photo and insert

Figure 5.3: Fit the male turned-pin headers.

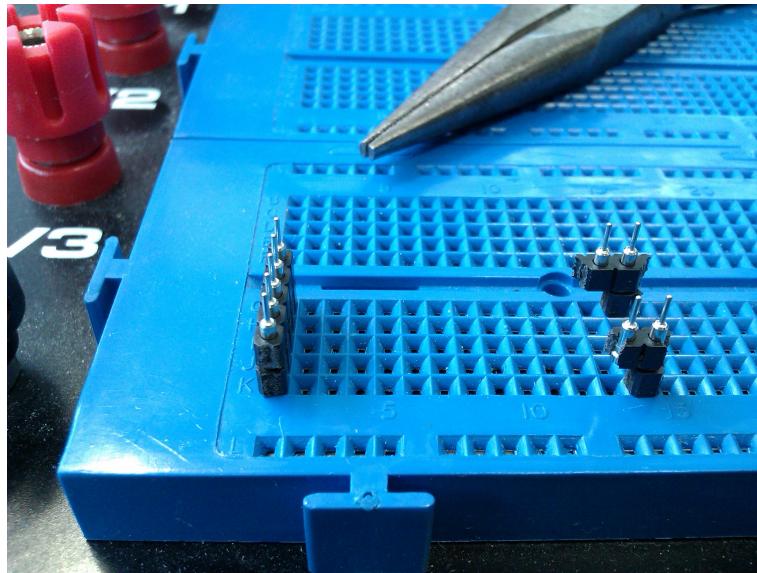


Figure 5.4: Fit the female turned-pin sockets.

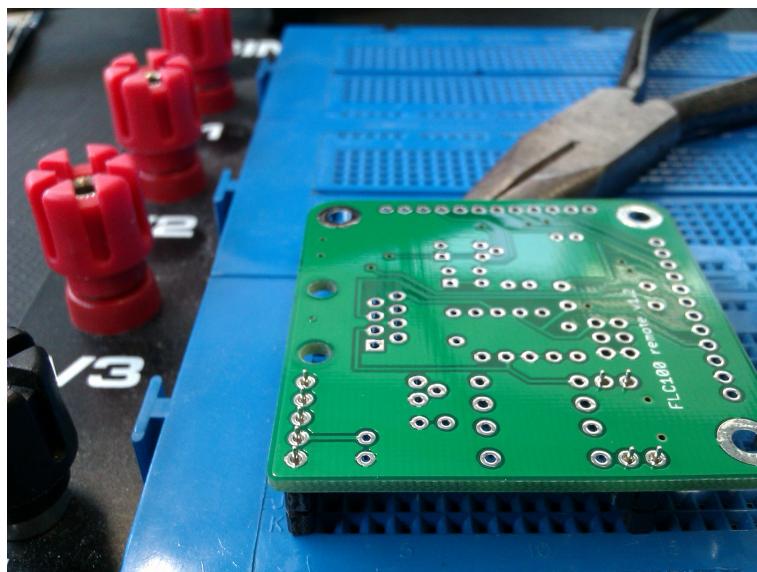


Figure 5.5: Place the sensor PCB onto the female turned-pin sockets. The pliers are used to support the other side of the PCB.

must be fitted close to the PCB to avoid fouling the FLC100 which is fitted over it.

18. SENS1. Gently fit the FLC100 sensor into the turned-pin sockets. Apply pressure only on the circuit boards, not the components or coil.

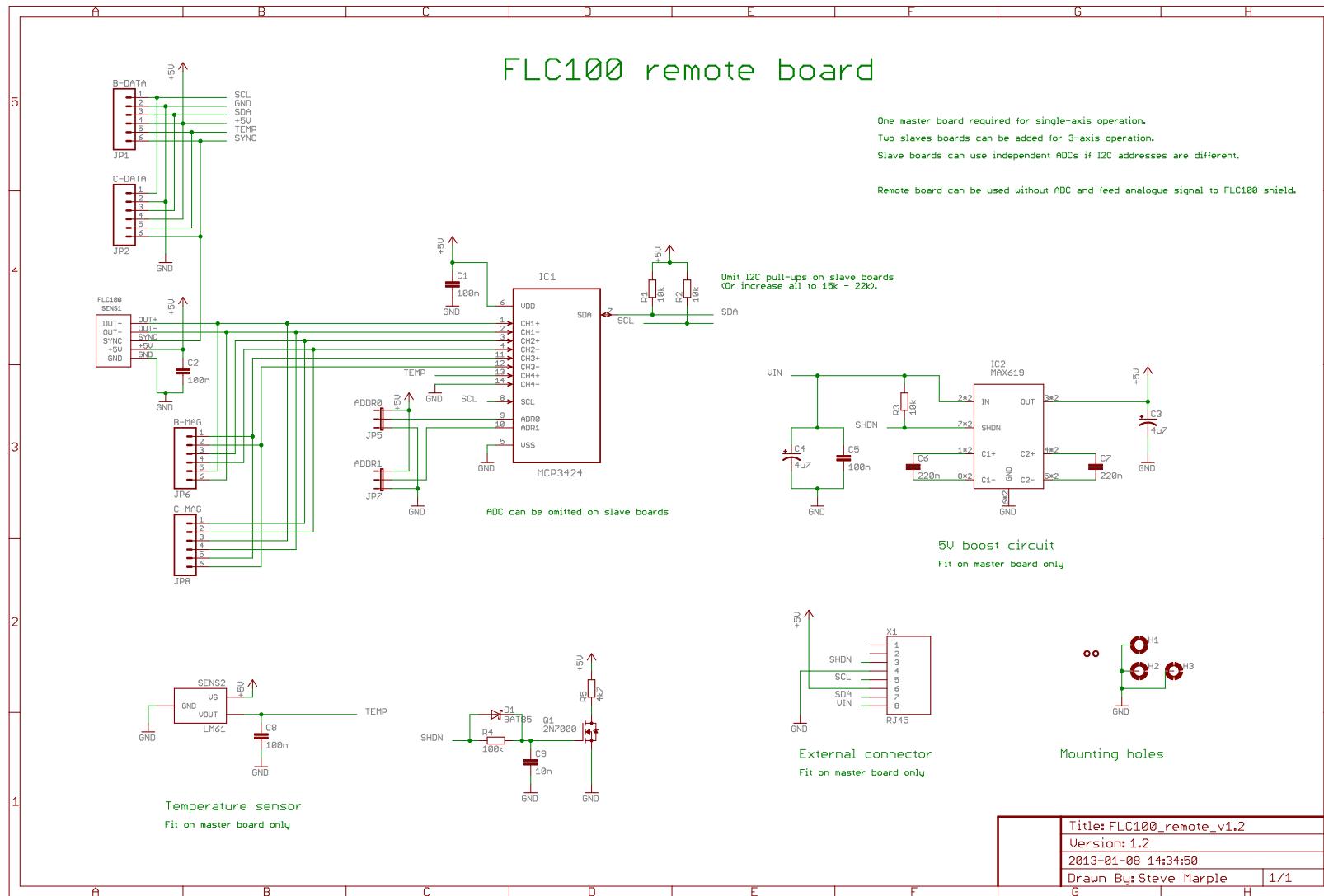


Figure 5.6: Sensor PCB version 1.2 circuit diagram.

# **Part III**

## **Installation**

# Chapter 6

## Site requirements

### 6.1 Sensor unit requirements

The sensor unit should be located outside away from human disturbances. The site for the sensor unit should be chosen with regard to the following requirements, with the highest priority given first.

- Within range of the base unit.
- Away from moving metal objects, for example, trains (more than 50 m), cars (more than 20 m) and garage doors.
- Away from static metal objects, in particular those containing the *ferro-magnetic* materials iron, nickel and cobalt.

The FLC100 fluxgate magnetometer sensor is slightly sensitive to temperature variations. To ensure correct behaviour a stabilised temperature environment is required. This is obtained by burying the sensor.

### 6.2 Base unit requirements

The base unit (Raspberry Pi and radio module) requires mains power and a wired network connection. Wireless networking is also possible but has not been tested; follow the instructions on the Raspberry Pi website. The base unit should be located indoors and as close to the sensor unit as possible.

#### 6.2.1 Network requirements

The following network requirements are needed for the system to operate fully:

- DNS resolution. This is normally provided as standard on most networks.
- Outgoing access on port 80 (HTTP) and 443 (HTTPS). Required for software updates.
- Outgoing access on port 123 (NTP), or access to a local NTP server.
- Outgoing SSH access (port 22).

# Chapter 7

## Raspberry Pi setup

### 7.1 SD card creation

If your SD card already contains Raspbian you can skip to section 7.2.

Download the latest Raspbian image and copy to the SD card following the instructions on the Raspberry Pi web site. **Copying the compressed image to a FAT partition on the SD card will not work.**

### 7.2 Configuring Raspbian

 In the command window you can press the `Tab` key to have Linux complete the command or filename.

 If you are not familiar with the nano text editor read The Beginner's Guide to Nano, the Linux Command-Line Text Editor

Raspbian is most easily configured by booting the new image. If you are able to discover the IP address (for instance, by checking the DHCP tables of your home router) you can do this over the network using SSH. Otherwise you must use attach a keyboard and monitor to the Raspberry Pi. If you are familiar with Linux it is also possible to edit the files by mounting the SD card on another Linux system.

### 7.2.1 /dev/ttyAMA0 serial port setup

Disable the console from running on /dev/ttyAMA0. Edit /boot/cmdline.txt to remove the parts which relate to ttyAMA0. Remove

```
console=ttyAMA0,115200 kgdboc=ttyAMA0,115200
```

Disable the getty process from running on /dev/ttyAMA0. Edit /etc/inittab. Find the line relating to ttyAMA0. Either delete the line entirely or comment it out by inserting a hash character (#) at the start of the line.

### 7.2.2 Raspbian configuration

Log in as pi and run

```
sudo raspi-config
```

#### 7.2.2.1 Change user password

If the default password has not been changed then do so now to keep your system secure.

#### 7.2.2.2 Internationalisation options

cron uses local time and the shift to and from daylight saving time complicates the cron tables. Set the Raspberry Pi's timezone to UTC to avoid daylight saving.

Select Internationalisation Options and then Change Timezone. For geographic area select None of the above, then select UTC. Select OK.

#### 7.2.2.3 Advanced options

Select Advanced Options and then Memory Split. Set the GPU memory to 16 (MB).

#### 7.2.2.4 Expand Filesystem

Finally select Expand Filesystem. Although the first option do this last. Choose Finish and then reboot.

### 7.2.3 Configure proxy server

Not all networks require a proxy server (or web cache) to be used, your network administrator should be able to advise. If it is necessary the setting should be configured in two places.

As user root

```
nano /etc/environment
```

At the end of the file add a line similar to

```
http_proxy='http://proxyhost:port/'  
https_proxy='http://proxyhost:port/'
```

You must replace proxyhost and port with the correct settings for your network. If the proxy server requires a username and password the lines should be similar to

```
http_proxy='http://username:password@proxyhost:port/'  
https_proxy='http://username:password@proxyhost:port/'
```

Replace username and password with the values your network administrator has provided.

Repeat for the procedure, as root

```
nano /etc/apt/apt.conf.d/10proxy
```

Add a line similar to

```
Acquire::http::Proxy "http://proxyhost:port";
```

Or, if a password is required, similar to

```
Acquire::http::Proxy "http://username:password@proxyhost:port";
```

A separate line for HTTPS is not required in /etc/apt/apt.conf.d/10proxy.

Proxy settings will not take effect until you log out and log back in. Type

```
logout
```

and then log back in.

#### 7.2.4 Install missing software packages

As user root

```
apt-get install screen lsof python-pip ipython python-matplotlib \
    python-scipy python-serial python-daemon python-lockfile \
    avahi-daemon dnsutils
```

#### 7.2.5 Remove swap file

To prolong the life of the SD card a swap file is not used. As user root

```
apt-get remove dphys-swapfile
```

#### 7.2.6 Configure file system mount options

As user root

```
nano /etc/fstab
```

Find the line where the root file system is mounted, it will look similar to

```
/dev/mmcblk0p2  /          ext4      defaults,noatime  0      1
```

Change the mount options (`defaults, noatime` in the example above) so that the mount options are now `noatime, nodiratime`. The line should look similar to the one below.

```
/dev/mmcblk0p2  /          ext4      noatime,nodiratime  0      1
```

## 7.3 Installing the AuroraWatchNet server software

### 7.3.1 Install the Git repository

As user pi

```
git clone --recursive https://github.com/stevemarple/AuroraWatchNet.git
git clone --recursive https://github.com/stevemarple/auroraplot.git
git clone --recursive https://github.com/stevemarple/Calunium.git
git clone --recursive https://github.com/stevemarple/xboot.git
mkdir ~/bin
cd ~/bin
ln -s ../AuroraWatchNet/software/server/awnetd/awnetd.py
ln -s ../AuroraWatchNet/software/server/awnetd/send_cmd.py
ln -s ../AuroraWatchNet/software/server/bin/log_ip
ln -s ../AuroraWatchNet/software/server/bin/upload_data.py
cd ~
```

### 7.3.2 Install python numpy library

As user root

```
dpkg -i /home/pi/python-numpy_1.7.1-3_armhf.deb
```

### 7.3.3 Configure python

As user pi

- The first line uses backticks, which on most keyboards can be found on the key above TAB.

```
user_site=`python -m site --user-site`  
mkdir -p $user_site  
cp ~/auroraplot/examples/example_auroraplot_custom.py \  
    $user_site/auroraplot_custom.py  
ln -t $user_site -s ~/auroraplot
```

### 7.3.4 Configure cron

As user pi

```
crontab -e
```

In the nano editor add the following two lines to periodically report that the Raspberry Pi is operating, make sure you terminate the last line by pressing the  key. This helps the AuroraWatch administrators monitor which stations are active. This reporting step can be omitted if you prefer.

```
@reboot /home/pi/bin/log_ip reboot > /dev/null 2>&1  
@hourly /home/pi/bin/log_ip > /dev/null 2>&1
```

Save the file,  -  ,  , .

### 7.3.5 Configure ifplugd

Configure ifplugd to report when the network interface has been assigned an IP address, which helps the AuroraWatch administrators monitor which stations are active. This step can be omitted if you prefer. As user root

```
cd /etc/ifplugd/action.d  
ln -s /home/pi/AuroraWatchNet/software/server/bin/log_ip
```

### 7.3.6 Create configuration file

As user root

```
mkdir /data  
chown pi.pi /data  
nano /etc/awnet.ini
```

TO DO: Create file contents, perhaps using a template copied from the repository

### 7.3.7 Create init file for server daemon

As user root

```
cd /etc/init.d  
ln -s /home/pi/AuroraWatchNet/software/server/awnetd/awnetd.sh awnetd  
update-rc.d awnetd defaults
```

### 7.3.8 Configure ntp

Check that the current time is correct by typing

```
date --utc
```

This will output the current date and time in UTC. If you aren't certain what the current time in UTC is then you can check at this web page, <https://www.google.co.uk/#q=utc+time>.

If the time is incorrect you will need to configure the NTP server. As user root edit /etc/ntp.conf:

```
nano /etc/ntp.conf
```

Find the line(s) which start server and a insert # at the beginning of the line(s) to comment it(them) out. Insert a similar line which begins with the word server but is followed by your own NTP server hostname. Your network administrator should be able to provide this information. Finally restart NTP; as user root

```
/etc/init/ntp restart
```

# **Chapter 8**

## **Installation procedure**

### **8.0.9 Tools required**

- Spade.
- Fork.
- Small bucket or other container to remove soil from the bottom of the hole.
- Compass.

The following items are optional but if they are available they may be useful for digging the hole.

- Soil auger.
- Post hole digger.

### **8.1 Base unit installation**

Connect the Raspberry Pi to wired ethernet connection. Connect the keyboard, mouse and monitor (if using) before powering up the Raspberry Pi. Connect the Raspberry Pi to a 5 V power supply with an output current rating of at least 700 mA. If you are not using a monitor connect to the Raspberry Pi using SSH, the default hostname is `raspberry.local`.

### **8.2 Determining the maximum range for the radio link**

Before installing the sensor unit first check the maximum range for radio communication. Note that this will vary according to many conditions and be sure to position the sensor well within the maximum range.

The maximum range can be determined with the following procedure:

1. Ensure that the Raspberry Pi is turned on and restart the recording daemon (p. 48). Monitor the recording process (p. 48).
2. Disconnect the remote sensor PCB from the FLC100 shield. Place the remaining circuitry about 5 m away from the Raspberry Pi. Keep the two units at least 2 m apart at all times to prevent overloading the radio receivers.
3. Connect the battery. The green LED on the Calunium PCB will flash 3 times with a frequency of about 1 Hz to indicate that the bootloader is waiting for data. The LED will then turn off.
4. About a second later the LED should light again, indicating the start of data transfer to the Raspberry Pi. If an acknowledgement is received the LED will turn off. This process should happen quickly, so that the LED flashes on momentarily. Just after the LED turns on the Raspberry Pi should output the received data message, along with its response to the sensor unit. If the LED remains on it indicates the microcontroller has not received an acknowledgement; see section ??.

This step repeats itself following every data sampling operation, normally every 30 s. The time interval can be reduced by altering the sampling period (section ??). A functioning radio communication link is required to alter the sampling period.

5. Move the sensor unit to its intended location. Try to position it at the same height as it will be when buried, approximately 10 cm above ground. Keep the radio antenna vertical. If the LED continues to flash briefly every 30 s the radio link is operating correctly. If not then find another location.
6. If the radio link appears to work correctly at the desired location then aim to find the maximum range by steadily increasing the distance between the sensor unit and base unit. Remember that the link is only tested after each sampling period. If the LED remains on move the sensor unit closer to the base unit and wait for the link to re-establish, indicated by the LED switching off.

**i** To save power the LED indicates the link status only for approximately 15 minutes after the power has been connected or the reset button pressed. If the LED has stopped indicating the link status press the reset button on the Calunium PCB.

### 8.2.1 Factors which alter the maximum range

The following factors can influence the range of the radio link:

- Obstructions. Try to find locations for the sensor unit and base unit which have direct line of sight. It is acceptable for walls to be in the path but note they will attenuate the signal. Avoid raised ground whenever possible. Placing the base unit on the first or second floor is likely to give improved range.

- Sources of radio noise. Some electrical equipment may cause RFI, which will reduce the range of the radio link. Try to keep both the sensor unit and base unit away from other electrical equipment, particularly those containing radio transmitters. “Smart” electric meters can contain radio transmitters which operate on the same 868 MHz ISM radio band.

### 8.3 Installing the sensor unit

The sensor unit must be should be buried to a depth of 0.85 m. If it is buried deeper then the radio antenna may be too close to the ground and compromise the wireless communication range, if it is buried too shallow the unit will be more susceptible to temperature variations.

After digging a suitable hole install the enclosure as vertical as possible and backfill the hole. Insert the wooden frame and rockwool into the enclosure. Using a compass align the north arrow on the wooden frame to point towards magnetic north. Connect the RJ45 cable to the FLC100 shield. Connect the power lead. Fit two D cell batteries into the battery holder and place on the wooden frame. Press the reset button and observe the green LED. It should flash 3 times to indicate the bootloader operation and then intermittently to indicate successful data transmission; see section 8.2.

# Chapter 9

## Contributing data to AuroraWatch UK

### 9.1 Use of data by AuroraWatch UK

Data contributed to AuroraWatch UK will be combined with other magnetometer data for the purpose of generating AuroraWatch UK or other auroral-related alerts. In future the alerts data are likely to be made available via a public API under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 license. Ideally you will also license the magnetometer data under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 license, remembering to define your own attribution requirements.

- i** You may choose a more permissive license, such as without the attribution, and/or non-commercial clauses, but if it includes the share-alike clause you must also dual-license it to AuroraWatch UK under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 license. This is because the share-alike clause restricts others from imposing additional restrictions. The alternative is to grant AuroraWatch UK permission to share data derived from your data under the CC BY-NC-SA 4.0 license.

The magnetic field data collected by AuroraWatch UK magnetometers will be made publically available under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 license, with a short embargo period (24 to 48 hours). If your magnetometer data is also licensed as CC BY-NC-SA 4.0 then AuroraWatch UK will share your data in the same way.

### 9.2 Methods to upload data

Two methods to upload data to AuroraWatch UK are supported, using rsync through an SSH tunnel, or by HTTP. Rsync contains an algorithm to efficiently transfer only the differences between the local and remote files and thus is ideal for transferring the real-time data files. SSH is used to provide a secure connection method. However, SSH access may not be possible on some networks (e.g., school networks). For cases when rsync cannot be used a HTTP upload method

is available which emulates some of the behaviour of rsync; whenever possible only the latest additions to a file are uploaded. The upload method is normally defined in `/etc/awnet.ini` configuration file, in the `[upload]` section.

### 9.2.1 Rsync uploads

As user root edit `/etc/awnet.ini`. At the end of the file add the `[upload]` section if it is missing, it should appear as

```
[upload]
method = rsync
```

As user pi create the keys for public key authentication:

```
ssh-keygen -t dsa
```

When prompted for the filename to save the key press `Enter` to accept the default. When prompted for the passphrase press `Enter` for an empty passphrase. Keep the private key (`/home/pi/.ssh/id_dsa`) secret, send the public key (`/home/pi/.ssh/id_dsa.pub`) to AuroraWatch UK.

Create the SSH config file to define hostname and user used for data transfer. As user pi edit the file `/home/pi/.ssh/config`, it should look similar to

```
Host awn-data
Hostname uploadhost
User uploaduser
```

You will need to obtain the upload hostname and username from AuroraWatch UK.

Insert an instruction into the crontab file to upload the data at regular intervals. As user pi:

```
crontab -e
```

Add the following lines:

```
## rsync upload
*/3 * * * * nice /home/pi/bin/upload_data.py > /dev/null 2>&1
```

## 9.2.2 HTTP uploads

As user root edit /etc/awnet.ini. At the end of the file add the [upload] section if it is missing, it should appear as

```
[upload]
method = http
url = upload_URL
realm = upload_realm
password = upload_password
```

You will need to obtain the upload URL, realm and password from AuroraWatch UK.

Insert an instruction the the crontab file to upload the data as regular intervals. As user pi:

```
crontab -e
```

Add the following lines:

```
## HTTP upload
# Upload text data for today at regular intervals
*/5 * * * * nice /home/pi/bin/upload_data.py -s today --file-types awnettextdata
# Make several attempts to upload all files from yesterday
5 */6 * * * nice /home/pi/bin/upload_data.py -s yesterday > /dev/null 2>&1
```

# **Part IV**

# **Operation**

# Chapter 10

## Raspberry Pi operation

### 10.1 Introduction

For generic operation of the Raspberry Pi (setting the hostname, assigning a fixed IP address etc.) please see the Raspbian documentation, <http://www.raspbian.org/RaspbianDocumentation>.

### 10.2 Shutting down the Raspberry Pi

The Raspberry Pi must be shutdown cleanly before power is removed:

```
sudo shutdown -h now
```

Before removing the power wait until only the red power LED is lit; wait a further two seconds to ensure futher access to the SD card is not needed. If the power is removed whilst data is being written to the SD card it will corrupt the file system.

To reboot the Raspberry Pi use

```
sudo shutdown -r now
```

### 10.3 Starting and stopping the data recording daemon

Data is recorded on the Raspberry Pi using a *daemon* process, which is started and stopped by the Debian init scripts. The scripts must be started and stopped as user root, the actual data

recording process runs as user pi.

To start data recording

```
sudo /etc/init.d/awnetd start
```

To stop data recording

```
sudo /etc/init.d/awnetd stop
```

It is also possible to check the status of the data recording process

```
sudo /etc/init.d/awnetd status
```

The restart option forcibly stops recording (if running) and then starts it again:

```
sudo /etc/init.d/awnetd restart
```

## 10.4 Monitoring the data recording process

The data recording process directs its standard output and error streams to a virtual terminal using screen. It is possible to attach to this virtual terminal to monitor the output.

As user pi

```
screen -r awnet
```

To exit from screen type **[CTRL]** - **[a]** , **[d]** .

 Pressing **[CTRL]** - **[c]** will terminate the recording process.

# Chapter 11

## Software and firmware updates

### 11.1 Raspbian updates

TO DO: See Raspberry Pi web pages

### 11.2 AuroraWatchNet software updates

The software can be updated easily simply by *pulling* a new version from the Github repository.  
As user pi

```
cd ~/AuroraWatchNet  
git pull
```

### 11.3 Sensor unit firmware updates

Only apply firmware updates if they are necessary. The firmware version selected must match the hardware, if it does not the sensor unit will be left inoperable and recovery will require an in-circuit serial programmer. First ensure that the AuroraWatchNet software has been updated (section 11.2) and the recording process restarted (section 10.3). To update the firmware

```
send_cmd.py --upgradeFirmware=name-version
```

Replace *name-version* with the firmware name/version string.

# Appendix A

## Configuration file options

### A.1 Introduction

Many of the AuroraWatchNet programs read a common configuration file, typically located at `/etc/awnet.ini`. The configuration file is broken into sections, each of which starts with a section header in square brackets (`([like_this])`). Other lines contain key names and values, written like `key = value`. Leading and trailing whitespace around both the key and value is ignored. Section headers and key names do not contain whitespace, words may be separated with an underscore). The configuration file can also contain comments, which are entered with a hash (#) or semi-colon (;) as the first character.

The configuration file is parsed using Python’s `SafeConfigParser` module. This allows values defined in the same section, or in the `[DEFAULT]` section to be inserted into other key value definitions.

### A.2 [DEFAULT]

The `[DEFAULT]` section is special as it defines values which can be used elsewhere in the configuration file.

#### A.2.1 site

Define the site code, typically a three letter abbreviation. This is used elsewhere within the configuration file, e.g., data filenames.

Default: none.

Example:

```
site = lan1
```

## A.3 [awnettextdata]

Options associated with the standard text-format output data file.

### A.3.1 filename

Define the filename used for text-format data files. This string is expanded as a strftime format string and accepts the normal strftime format specifiers; for a list of the acceptable format specifiers see <https://docs.python.org/2/library/time.html#time.strftime>. However, since Python expands the string first any percent characters used as part of a strftime format specifier must be repeated.

Default: none.

Typically set to filename = /data/aurorawatchnet/%(site)s/%%Y/%%m/%(site)s\_%%Y%%m%%d.txt

**Example:**

```
filename = /data/aurorawatchnet/%(site)s/%%Y/%%m/%(site)s_%%Y%%m%%d.txt
```

%(site) is replaced with the site abbreviation which was defined previously in the [DEFAULT] section. Notice how the strftime format specifiers require two % characters.

For June 20th 2014 with the site abbreviation cwx this would expand to

```
filename = /data/aurorawatchnet/cwx/2014/06/cwx_20140620.txt
```

## A.4 [awpacket]

Options associated with the standard binary output format. This format is inconvenient to read but preserves the received data messages from the magnetometer, and the responses sent back from the recording daemon. It is possible to play back these files to the recording daemon and regenerate other data formats.

### A.4.1 filename

See section A.3.1 for a description.

Default: none

Typically set to filename = /data/aurorawatchnet/%(site)s/%%Y/%%m/%(site)s\_%%Y%%m%%d.awp

### A.4.2 key

By default the binary data packets are written out with their original signing key. If you plan to make the binary data format available you should then for security reasons you should probably set a different signing key. You will then be able to share this key without compromising the communication channel with the magnetometer. The key is a 32 character hexadecial string, without any 0x prefix.

Default: none Typically set to a simple code, key = 00000000000000000000000000000000

## A.5 [logfile]

Options associated with the recorded log files.

### A.5.1 filename

See section A.3.1 for a description.

Default: none

Typically set to filename = /data/aurorawatchnet/%(site)s/%%Y/%%m/%(site)s\_--%%Y%%m%%d.log

## A.6 [daemon]

Options relating to the data-recording daemon.

### A.6.1 connection

Defines the connection between the daemon and the magnetometer. Radio communication emulates a serial port connection. If the magnetometer is PoE model the communication should be set to ethernet.

Default: serial

### A.6.2 close\_after\_write

Have the daemon close the data and log files after each write. It is recommended that this option only be used on an NFS file system, when used on a file system store on flash memory (such as the Raspberry Pi) this may cause excessive writes and early failure of the flash memory.

This option may be required for real-time data collection daemons where the data files are also received by rsync or HTTP uploads.

### A.6.3 **acknowledge**

If set to false then acknowledgements are not sent in response to magnetometer data messages.

Default: true.

### A.6.4 **read\_only**

If set to true then for serial connections the device is opened read-only and no set up commands are sent.

Read-only mode implies that acknowledgements are not sent, regardless of the setting of the acknowledge option.

Default: false.

## A.7 [serial]

Options relating to serial data communication between the magnetometer and data recording daemon. This section is not used when communication uses ethernet.

### A.7.1 **port**

The filename of the serial port. For the Ciseco *Slice of Radio* module this should be set to /dev/ttyAMA0, for the Ciseco *URF* module this will probably be /dev/ttyACM0.

Default: /dev/ttyACM0

### A.7.2 **baudrate**

Baud rate used with the serial port. For the Ciseco *Slice of Radio* module this should be set to 9600, for the Ciseco *URF* module a higher rate of 57600 can be used.

Default: 9600

### A.7.3 **setup**

The set-up string which should be sent to the serial device, for defining the channel number used for communication etc..

Default: none.

Typically set to ATRE;ATCN 25;ATLI R;ATAC. Ensure that the channel number defined here matches the setting programmed into the magnetometer's EEPROM.

## A.8 [ethernet]

These configuration options are used by the data recording daemon only when an ethernet connection with the magnetometer is in use.

### A.8.1 **local\_address**

The local IP address used for the data recording daemon. By default the daemon listens on all available interfaces.

Default: empty string.

### A.8.2 **local\_port**

The port number of which the recording daemon should listen for magnetometer data packets.

Default: 6588.

### A.8.3 **remote\_address**

The IP address of the magnetometer.

### A.8.4 **remote\_port**

The port on which the magnetometer expects to receive acknowledgements.

## A.9 [controlsocket]

It is possible for the `send_cmd.py` program to send commands to the magnetometer via the data recording daemon. Communication is via a UDP socket or a unix domain socket.

### A.9.1 **filename**

Use a unix domain socket for communication with the data recording daemon, with the given filename. If set to none then a control socket will not be created. If the `filename` option is present it takes priority over any `port` option.

Default: none.

### A.9.2 **port**

Use a UDP socket for communication with the data recording daemon, with the given port number. If set to none then a control socket will not be created. If the `filename` option is present it takes priority over the `port` option.

Default: 6587

Example:

```
port = 6587
```

## A.10 [magnetometer]

Settings associated with the magneometer.

### A.10.1 **siteid**

The numeric site identifier for the magnetometer. The recording daemon will ignore data packets where the site ID does not match the value set in the configuration file. The site ID should be set as an integer number in the range 0 to 255 inclusive.

Default: none.

### A.10.2 **key**

The HMAC-MD5 key used to sign communication messages sent between the magnetometer and data recording daemon. This key should be kept secret. The key is a 32 character hexadecial string, without any `0x` prefix.

Default: none.

Typically set to a simple code, `key = 00000000000000000000000000000000`

Example:

```
key = 65f024a22214ea2511ca60f251d7fb74
```

## A.11 [upload]

Configuration details for the upload\_data.py program.

### A.11.1 method

Define the upload method in use. Valid options are: rsync, rrsync, http, and https. rsync uses the rsync program to efficiently transfer only the portions of data files which have changed since the last upload; rrsync is similar. Both are tunnelled through SSH and thus require the network to permit outward TCP connections to port 22.

The http and https upload methods make use of the standard HTTP(s) protocol, and therefore require outgoing TCP connections to port 80 and 443 respectively.

### A.11.2 rsync\_host

Name of the rsync host. No facility to set the user is provided so use the SSH config file to set the appropriate details.

Example SSH config file:

```
Host awn-data
Hostname host.domain.com
User monty
```

### A.11.3 url

The URL to which data is sent.

### A.11.4 password

The upload password. Digest authentication is used.

### A.11.5 **realm**

The *realm* used for digest authentication.

## A.12 [realtime\_transfer]

The data recording daemon is capable of forwarding the incoming binary packets to a remote host. When this is enabled data is transferred in real-time. The remote host can use the same data recording daemon but it should be configured to be read-only so that it does not send acknowledgements.

When data packets are forwarded in this way the daemon does not check that they have been delivered successfully. It is recommended that this method is used in conjunction with the `upload_data.py` program to ensure any data packets which were not delivered are transferred by some other means.

### A.12.1 **remote\_host**

The hostname or IP address to which data packets are sent.

### A.12.2 **remote\_port**

The UDP port to which data packets are sent.

### A.12.3 **remote\_key**

The HMAC-MD5 key used to sign the data packets. This should be different to the key used to communicate with the magnetometer.

## A.13 [dataqualitymonitor]

The data recording daemon can monitor for the existence of a file indicating that data quality may be compromised (e.g., due to local site activities such as cutting grass). If the file is present an extension is appended to the data files to clearly separate the poor quality data. Any real time data transfers in operation are suspended. When the file is removed the normal filenames are used again and real time data transfer resume as normal.

### A.13.1 **port**

Defines the serial port used by the awnetd\_monitor daemon. If using a USB device it is recommended to modify the udev rules so that a fixed device name is generated based on the serial number.

Example udev rule to create a symbolic link for an FTDI USB serial adaptor, save to /etc/udev/rules.d/90-usb\_serial.rules or similar:

```
# Have symlinks based on serial number for FTDI devices. Used for
# awnetd_monitor
SUBSYSTEMS=="usb", KERNEL=="ttyUSB[0-9]*", \
ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6001", SYMLINK+="tty_ftdi_%s{serial}"
```

Default: none.

### A.13.2 **filename**

The name of the file which when present indicates reduced data quality. This file can be created by the awnetd\_monitor daemon or an external process.

Default: /var/aurorawatchnet/data\_quality\_warning

### A.13.3 **extension**

The extension which is appended to files to indicate reduced data quality.

Default: .bad

### A.13.4 **pidfile**

A file containing the process ID of the awnetd\_monitor daemon.

Default: none.

### A.13.5 **logfile**

The file where the log output of the awnetd\_monitor daemon is written.

Default: none.