BuildAX Documentation

The BuildAX family of wireless environmental monitoring devices comprises of sensors, actuators and a logger/router made available under an open license. The devices were developed within a joint research project undertaken by Newcastle University's Digital Interaction Group and Sir Joseph Swan Centre for Energy Research and the University of Strathclyde's Energy Systems Research Unit.

The devices can be rapidly deployed to enable low cost built environment monitoring, and are fully compatible with the Open Source EnTrak™ system as a means to deploy 'eServices' for systems conditions monitoring, occupant feedback, performance certification and the like.

Downloads

BuildAX resources and downloads are available in the OpenMovement subversion repository (click here), including firmware and utilities.

APIs to be documented

- UDP connection
- Actuators / Power Switches

For our other other projects, please visit the Open Movement repository.

People

- Karim Ladha - Hardware, lead Firmware developer
- Sam Mitchell Finnigan - Software, LRS Firmware, Documentation

Thanks also to Jon Hand at ESRU (University of Strathclyde) for feedback, usability testing, and bug reports.

Developed at Newcastle University, UK. Available under CC 3.0 BY Attribution License
Docs built with MkDocs • Theme based on Read the Docs

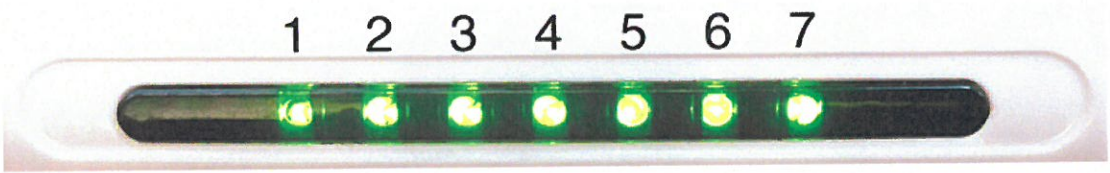
Docs » Hardware

BuildAX LRS

The BuildAX LRS (Logger Router Server) is primarily a data-logger , but has a lot of extra functionality which you may find useful in a wide range of deployment scenarios.



LED codes



The BuildAX LRS includes 7 LEDs which indicate the status of the device:

LED	Function
1	Power
2	USB CDC (Serial connection available)

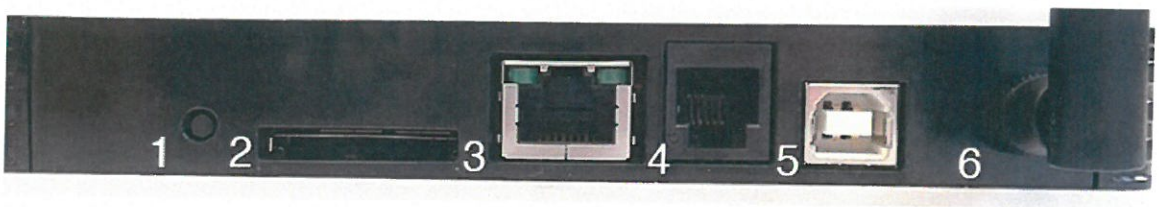
LED	Function
3	Network Connectivity
4	SD Card Mounted
5	Log File Activity
6	433MHz Radio Activity
7	Remote connection (telnet) active

LRS LED Flash Codes

There are a number of situations where a flashing pattern will be displayed on the LRS LEDs:

Pattern	Meaning	Action
Odd/Even LEDs flash alternately	Pairing Mode	Hold the button on the rear of the LRS for **1 second**. The LRS will enter pairing mode. Proceed to pair up BuildAX sensors.
2 groups of 3 LEDs flash alternately	Mount USB device	Hold the button on the rear of the device for **5 seconds**. This will mount (or unmount) the USB disk, and re-initialise the filesystem. The Si44 Radio will also be re-initialised.
All LEDs flash quickly, twice	Load Factory Defaults	Hold the button on the rear of the LRS for **10 seconds**. This will reset the device to factory defaults but will not wipe files from the SD card (sensors will remain paired).
LEDs flash counting up from 1-7 continually	Bootloader Mode	The device is in bootloader mode. This is triggered by holding the button on the rear of the device while the power is being connected. See the Firmware page for more info.
All LEDs flash slowly, continuously	General Exception	A fatal LRS firmware error occurred. This usually signifies that there is a bug which will need to be fixed. Please send us your `ERRORS.TXT` from the SD card to help diagnose the problem.

Connections



No.	Function	Description
1	Pairing Button	Used to make the LRS enter pairing mode, in which BuildAX sensors may be paired to the device.
2	SD Card Slot	Accepts any SD card formatted with the FAT filesystem
3	Ethernet Jack	Used for connecting the BuildAX LRS to a TCP/IP network
4	Debugger Jack	Debugging interface for firmware development
5	USB B Jack	Provides power to the LRS. Also exposes a USB CDC Serial connection and Mass Storage Device to access the SD card

No.	Function	Description
6	433MHz Antenna	433MHz Radio receives data packets from BuildAX sensors

Sensors

BuildAX ENV

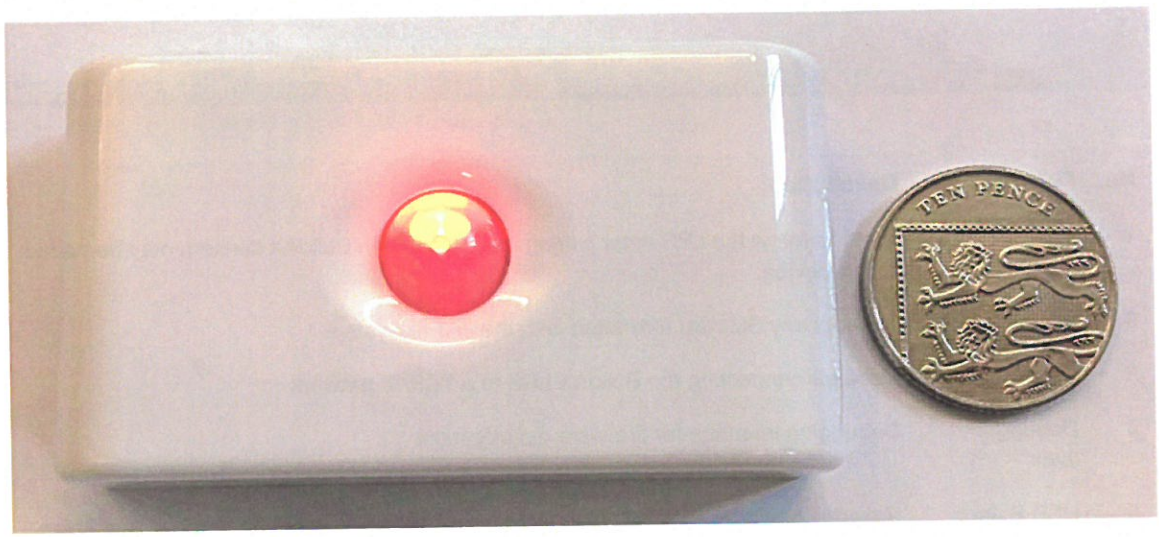
The BuildAX LRS is designed to receive data from a variety of sensor nodes. The currently available device is the BuildAX ENV (environmental sensor node).



This is a small, indoor environment sensor which transmits the following values:

- Temperature
- Humidity
- Light
- PIR (Passive Infrared movement sensor)
- Magnetic Reed Switch

Front

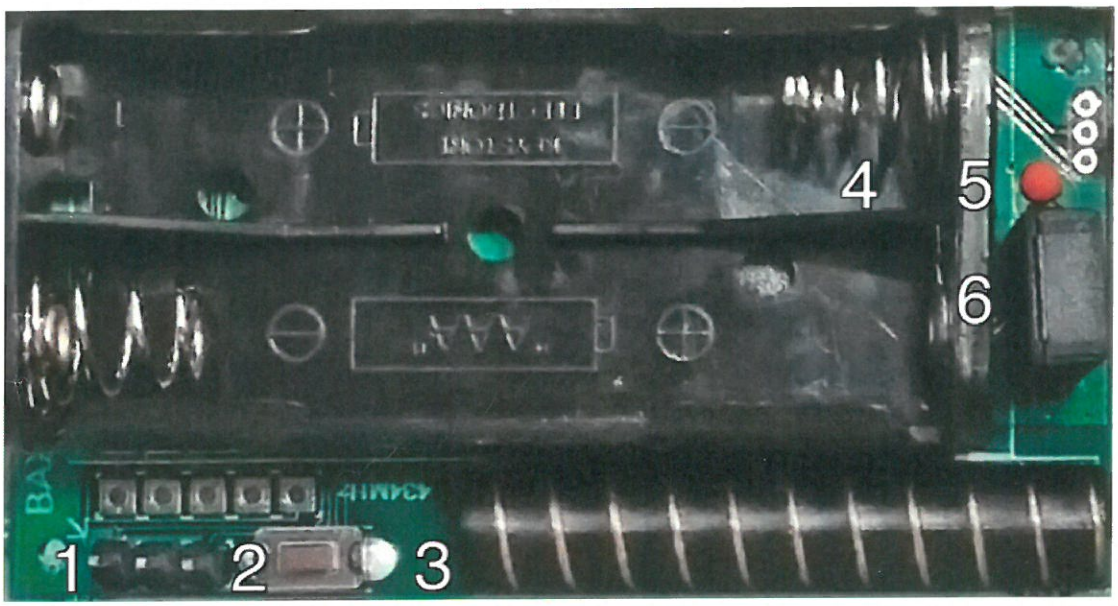


The PIR sensor by default shows a red LED when triggered, similar to those found in security/alarm systems. This indicates that the sensor node is working correctly.

A green LED is displayed when the magnetic switch is triggered, or when the button inside the device is pressed to send an encryption packet.

Rear

The back panel of the device is removed to reveal the battery compartment and other components.



No.	Function	Description
1	FTDI Header	A connector used to configure the device, and program the BuildAX Sensor firmware
2	Pairing Button	Press this to send the sensor's encryption packet. See Pairing Sensors for full instructions on pairing sensors to the LRS.
3	Antenna	433MHz Radio antenna for sending BuildAX sensor data to the LRS wirelessly.
4	Battery Holder	Two AAA (triple A) batteries power the BuildAX sensor, expected battery life is over a year with the default configuration
5	Temperature Sensor	Temperature sensor (1% accuracy point-matched thermistor)
6	Humidity Sensor	Humidity sensor (5% accuracy between 20-90% RH)

The inclusion of a magnetic switch allows the sensor to be mounted on a door or window frame, and transmit data on whether the aperture is open or closed.

A magnet similar to those used on cupboard doors should trigger the switch, and should be aligned on the opposite side of the device to the antenna.

ENV LED Flash Codes

Pattern	Meaning
RED flash	PIR event (event detected)
GREEN flash	Magnetic switch event (either direction of magnet movement)
RED-GREEN-RED-GREEN-RED-GREEN flash	Device reset(e.g. reset command or new batteries)
GREEN-OFF-GREEN-OFF-GREEN-OFF	Device settings invalid or encryption+id changed (e.g. button hold + new batteries)
RED dim flicker at 1 Hz	Device battery depleted (remove depleted cells to avoid cell leakage)

Device control

Pattern	Meaning
Press button (<10 seconds)	▪ LED Flashes at 1Hz Green ▪ Encryption data is transmitted at low power (short range) ▪ Sensor data is transmitted at normal power ▪ Full event credits are granted
Hold button for 10 seconds	▪ LED turns red ▪ Factory default settings are latched ▪ Device will reset on button release
Connect cable and press button	Device enters settings mode immediately
Insert batteries while holding button	Device factory resets AND generates new id and encryption

Sensor Specification

Full specification of the BuildAX ENV sensor hardware:

Battery power

- 2 standard AAA cells
- Battery life depends on usage and location e.g. for a 900 mAh typical alkaline battery:
  - Sensor sample every minute + 1 event every minute = **~0.5 Years**
  - Sensor sample every 5 minutes + 50 events per day = **~3.0 Years**

Temperature

- -10 to 50°C ±2°C
- -7 to 47°C ±1°C
- 0.1°C resolution

Humidity

- 20 to 90% Saturation
- 5 to 60°C
- 5% accuracy
- 0.1% resolution

Light

- Linear sensor with ±30% gain error max
- Typical gain error ±10%
- Fluorecent lamp flicker filter
- 2 point calibration possible for more accuracy
- 1000 Lux range
- 1 Lux resolution

PIR

- PIR energy integrator indicates total movement
- PIR events count movement detections
- Events can trigger transmit of data

Magnetic switch

- Reed switch magnetic sensor
- Switch events are generated when the magnetic field changes accross threshold
- Events can trigger transmit of data
- Events accumulate with LSB set to switch state.
- 0 = magnet present, 1 = magnet not present

Radio

- Si4032 433 MHz chipset
- Centre frequency is 434 MHz, 2GFSK 62.5KHz deviation
- -96 dBm sensitivity, ±210 kHz MOD BW, +10dBm Tx power (0 - 20dBm configurable)
- AES encryption

Developed at Newcastle University, UK. Available under CC 3.0 BY Attribution License

Docs built with MkDocs • Theme based on Read the Docs

Quick Start Guide

Depending on the number of sensors needing to be paired, and the size of the building a full BuildAX deployment may be done from scratch in as little as 20 minutes. There are three main parts in this guide:

1. Set up the BuildAX LRS
2. Pair sensors to LRS
3. Deploy sensors in building

Part List

The following components are required for a basic deployment:

- BuildAX LRS
- SD Card (1GB+ recommended)
- USB A-B cable
- USB Mains power adapter
- Ethernet cable

If you do not require network connectivity for your deployment, feel free to disregard steps related to this and use the alternate serial commands.

LRS Setup

First-time Setup
For first-time setup, you may prefer to connect the LRS to a laptop (rather than the USB mains power adapter) in order to troubleshoot network connectivity using the serial interface.

Pre-deployment steps:

- Download the Web Interface zip file found in the Downloads section of the OpenMovement repository to your computer. Note that this only needs to be done once, when you first set up the LRS.
- Extract the files (keeping them inside the "WWW" directory) to the root of your SD card, using a card reader

Setting up:

1. With the LRS powered OFF, eject card from computer & insert into LRS
2. Connect the ethernet cable between the BuildAX LRS and a network access point (or home Broadband router)
3. Connect USB power cable to the BuildAX LRS and wall adapter
4. Connect to the LRS by typing its address into your browser (click [here] for tips on finding address). (Additional steps are required if your network doesn't support NETBIOS or your LRS is missing the sticker with this information)
5. Log into the LRS with the default credentials (`admin/password`)

6. Click "Settings" and turn on logging (and set other options if required)
7. Change default credentials

Logging options can be set via the web interface (see here) or via the console (see here).

Video

Pair/Deploy ENV Sensors

For a more in-depth walk through pairing, please see the the deployment section.

Quick pairing steps:

1. Press the button on the back of the LRS until the LEDs flash. You have 5 minutes to pair your sensors
2. Open the BuildAX sensor. Press the button next to the battery compartment, holding the sensor close to the LRS.
3. The LED on the front of the sensor should illuminate green to indicate that a pairing packet has been sent.
4. Determine if the sensor is paired using the live data interface (or console)

Quick deployment steps:

1. Choose a location out of direct sunlight, with good airflow.
2. Using a sticky pad or hook, attach the back plate of the sensor to the wall
3. Clip the sensor into the backplate
4. Verify that packets are still being received from the device

Video

You're done!

Developed at Newcastle University, UK. Available under CC 3.0 BY Attribution License

Docs built with MkDocs • Theme based on Read the Docs

Deployment

This section covers topics that might affect your sensor deployment as well as some basic best practices.

Pairing Sensors

To pair sensors to the BuildAX LRS it must first be put into Pairing Mode. To do this, hold the button on the rear of the device until the lights flash alternately. The LRS will stay in pairing mode for a period of 5 minutes.

What is "Pairing"?

The BuildAX sensors broadcast data packets containing information about their environment over the air. This data could potentially be used by eavesdroppers to infer whether a building is occupied, something existing systems have typically not addressed. As a security measure, all BuildAX sensors use 128-bit AES encryption to prevent packets being read while in transmission.

To decrypt data from the sensors, the LRS must know the key which was used to encrypt it. An encryption key is transmitted from the sensor to the LRS when the button on the rear of the sensor (see #2 on the hardware diagram) is pressed. The radio broadcast power is set to the minimum on the sensor, to limit the range at which the packet can be received.

The LRS then stores the unique key for the sensor in the `BAX_INFO.BIN` file on the SD card. To instruct the LRS to listen for keys, it must be put into "Pairing Mode", by pressing the button on the rear of the device.

> **Note:**
> The LRS will log packets from BuildAX sensors regardless of whether it has received an encryption packet, as these may be decrypted at a later date and would otherwise be lost.

ENV Sensor Deployment

ENV Sensor Range

The achievable range of the sensors will depend on a number of factors, including the materials used in the construction of the building, the thickness of the walls, and the "noise-floor" of your deployment environment.

If there are a lot of devices transmitting on the same 433MHz band as the sensors (for example, in a city environment) the radio frequency will be noisier and the sensors will not be "heard" by the LRS at further distances.

Tips on deploying sensors in a building:

- Position the LRS at a central location, so as to be in range of all sensors.
- Air temperature and humidity in a room are stratified (change based on height).
- The PIR lens is directional. Mount sensors on the wall **vertically** if this is important.
- If a sensor at the limit of its range, try upping the transmission interval.
- Walls appear "thicker" when the radio signal has to travel through them diagonally.
- Try to avoid direct sunlight when attaching sensors to surfaces. This will skew the data they generate.
- Also remember the position of the Sun will change throughout the day!

Note also that the battery life of the ENV sensor will be lower on average in high-traffic areas, as a data packet will be transmitted every time the PIR sensor is triggered.

Handling Sensors:
Note that after handling the ENV sensors, the humidity/temperature reading will be biased by the temperature of your hands. This will settle again after 5 minutes or so.

RSSI

A sensor's signal strength is indicated by its RSSI value (Received Signal Strength Indication). This can be thought of as the "loudness" of the radio signal and is measured in dBm.

RSSI values can be found in the web interface (see the Live Data page), or in the 4th column of the CSV-format data available as a file, or over the various streaming interfaces (telnet, serial, websocket).

Signal strength will be affected by the battery level of the sensor, the distance from the LRS, and any obstacles which the signal has to travel through in-between.

Noise floor

All environments will have a certain amount of potential noise present in the radio spectrum used by the BuildAX sensors. This level of noise is referred to as the noise "floor", and sensors transmitting must be "louder" than this background noise level in order to be heard.

Developed at Newcastle University, UK. Available under CC 3.0 BY Attribution License
Docs built with MkDocs • Theme based on Read the Docs

Connecting to the BuildAX LRS

Multiple methods of connecting to the LRS are supported- these fall into two categories: a text-mode command interface, and a web configuration interface.

Serial (USB)

> **Driver:**
> On first-time use, the USB CDC serial driver will need to be installed. (Download the CDC driver).

The serial connection is available when the LRS is directly connected to a PC via the USB cable. To access a serial connection, some additional (free) software is required. We have tested using the following apps on these platforms:

- **Cross-platform**: CoolTerm
- **Windows**: Hyperterminal,PuTTy
- **Linux / Mac OSX**: GNU screen

Microsoft Windows

The device will appear as a COM port on Windows (can be found in the Device Manager, Hyperterminal and CoolTerm will also enumerate serial ports). The COM port number will usually be COM2 or above (COM1 is now a virtual interface on modern Windows).

Linux, OSX (*nix platforms)

Devices on *nix platforms are listed under the `/dev/` directory.

USB Mass Storage

Finally, data files logged by the LRS can be retrieved directly by using it as a USB Mass Storage device. For this option to be available, data logging on the LRS must be disabled (this is because it would be unsafe to continue writing data to files which could be read at any time).

To override this, use the `mount` command.

> **Warning:**
> Manually deleting (or otherwise removing) data files from the mass storage device may cause the data fetch service to become unstable.

Network Access

When the BuildAX LRS is connected to your network using an Ethernet cable, it will request an IP address using DHCP.

On home networks, the IP address assigned will normally be of the form 192.168.x.x. On corporate or academic networks you may need to request an address from your network administrator, quoting the device's MAC address. They may also be able to assign a DNS address for convenient access.

Finding the IP address

The MAC and IP address assigned to the LRS may be discovered by connecting via serial and typing `status`. You may also find it useful to assign a static IP address to the device for the purposes of port forwarding.

Telnet

Telnet can be used to access the LRS command terminal over the network without needing to plug the USB cable into a PC. All of the commands listed in Commands (LRS) are supported.

The `telnet` command is used to connect to this console:

```
telnet 192.168.0.15
```

... or use PuTTy on Windows. The LRS's telnet service runs on the default TCP port 23.

Web Interface (HTTP)

The BuildAX LRS configuration interface is accessed using a web browser, providing a convenient way to change settings and view LRS status without needing to use the text-mode terminal.

On Windows, type the NETBIOS name of the device may be typed directly into the address bar. This will be in the form BAX-xxxxxx (e.g. `bax-434c32`), and can be found on the sticker on the rear of the device.

For information on using the web interface, please see the Web Interface User Guide.

Connecting to a BuildAX ENV Sensor

In order to configure settings on the BuildAX ENV sensors, you may connect directly to the sensor using an FTDI cable. You will require a VCP (Virtual COM port) USB driver for FTDI, available here.

This guide demonstrates how to connect to an ENV sensor using HyperTerminal. On other operating systems, the settings will be the same, though the procedure may differ.

For information on flashing firmware to the device, please refer to the Firmware section.

On creating a new serial connection with HyperTerminal, you should select the new COM port created by the FTDI driver (this will usually be a number greater than COM1).



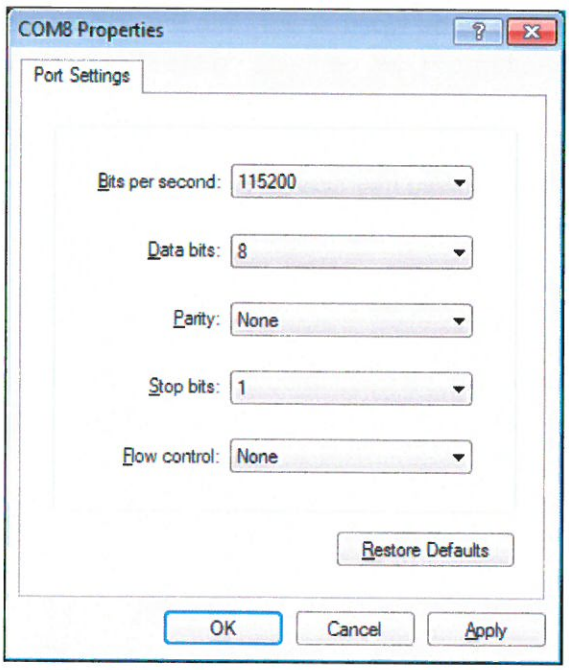
Hyperterminal will enumerate the COM serial ports available on the system, as in this example. To connect using PuTTy, you will need to find the COM port number using the Device Manager in Windows, or use a fork of the project with COM enumeration.

To connect on a Unix platform, the `screen` command may be used along with the device path (under `/dev/`). Details of using this command for serial connections can be found here.

All connection methods will require configuration with the correct serial parameters.

Connection Settings

- **Baud Rate**: 115200
- **Data bits**: 8
- **Parity**: None
- **Stop bits**: 1
- **Flow control**: None



Select "OK", and the FTDI Serial connection will be open, ready to connect a sensor.

Connecting to ENV sensors

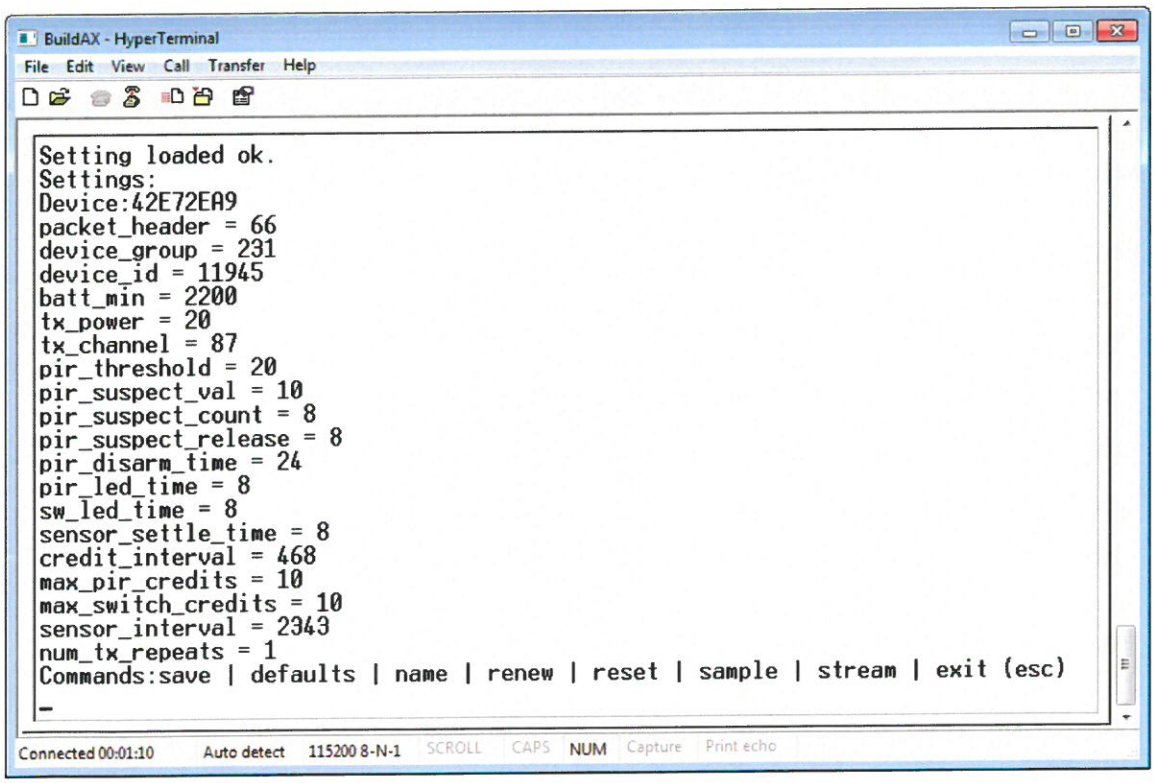
Starting from the corner of the sensor and working inwards, the following FTDI wires should be connected to the 3-pin connector:

1. Orange (pin 1)
2. Black (pin 2)
3. Yellow (pin 3)

Your FTDI cable may also have Red, Brown and Green wires, these are not used by the ENV sensor and should be left disconnected.

To connect to an ENV sensor:

1. Insert batteries into the sensor (if they are not already inserted)
2. Push the 3 pins (listed above) of the FTDI connector onto the header (see item **1** here), with the orange wire on pin 1.
3. Press the sensor button (see item **2** here) to enter into command mode.
4. Watch your command terminal. The sensor should start up and print its status over the connected UART:



Note that while the FTDI cable will provide a voltage and power the sensor, it is recommended that batteries are left in the device while the the serial connection is accessed.

Developed at Newcastle University, UK. Available under CC 3.0 BY Attribution License
Docs built with MkDocs • Theme based on Read the Docs

Commands

This document summarizes the serial command set of the BuildAX-LRS. The serial command set is available through the remote `telnet` interface and through the USB CDC (Communication Device Class) serial port. The command set and responses are subject to change as the open source BuildAX project expands.

The command strings are followed by `[CR][LF]` (carriage return, line feed) in all cases; meaning that you should press the `Enter` key to send a command after typing it.

(On Unix platforms, only an `[LF]` is sent- this is detected and handled accordingly.)

For information on how to access the text-mode command terminal, please see the Connection Guide.

Settings

Settings commands are expressed in the form `setting.type.x=y` and are used by the LRS to determine where to route messages. A 'message' in this context is a data packet sent by the environment sensors and received by the device, which must be routed to various outputs.

Each `setting.type` describes a data output, or "stream":

Type	Description
usb	USB CDC (Communications Device Class) Serial connection
file	Logging of data packets to a file on the SD card
udp	Network packet-forwarding over UDP (User Datagram Protocol)
telnet	Telnet protocol (virtual terminal connection) provided as a networked alternative to the serial interface
websocket	Websocket packet-forwarding (used in the Web Interface to provide real-time data)

Each stream has two configurable options: `stream` and `mode`.

- **Stream**: Enable/disable the data output. Off: `stream=0`, On: `stream=1`
- **Mode**: Output as plaintext (`mode=0`) or slip-encoded binary (`mode=1`)

The LRS will print a summary to confirm the command after it is typed.

It is possible to configure the LRS with options which are potentially useless- streaming data over Telnet in binary, for instance, would be undesirable in most situations.

BuildAX LRS Command List

Basic commands

These commands are for the normal user and provide access to the devices usual functions.

Commands below are shown in the form: *command*, **summary**, `example` (and response).

`help`

Summary: Requests LRS to print a summary of common commands

Example: `help`

```
HELP: help|status|time|reset|mount|exit|save|setting
```

`status`

Summary: Request a printout of current status and settings

Example: `status`

```
BAX, HW:1.1, FW:1.1
NAME: BAX-434C32
Start time: 2014/09/03,14:57:01
Time:       2014/09/03,15:26:43
SysVolts: 4850mV
VddVolts: 3321mV
SysTemp: 39.5?C
Settings: From eeprom

Username:admin
Password:password
Si44 Radio:  On
Subnet:4200
Mask:FF00

ETHERNET: Connected
MAC:02:04:A3:43:4C:32
IP: 10.67.34.50
MSK:255:255:240:0

USB:
Mode: 0
Setting: Turned on
Status:  Streaming

File:
Mode: 1
Setting: Turned off
```

```
Status:  Card present

Telnet:
Mode:0
Setting: Turned on
Status:  Streaming

Udp:
Mode: 0
Setting: Turned on
Status:  Enabled

Websocket:
Mode: 0
Setting: Turned on
Status:  Enabled
```

`time`

Summary: Request a printout of or set the time

Example: `time` or `time=2014/08/14,10:48:54`

```
Time: 2014/08/14,10:48:54
```

`save`

Summary: Request all open filed to be saved. This ensures all the cluster chains are written to disk. Files in use will be re-opened.

Example: `save`

```
Saved files
```

`reset`

Summary: Request a reset. Unsaved settings are lost. Open files are closed in orderly fashion

Example: `reset`

```
Resetting
```

`exit`

Summary: Request to close the terminal. Not possible for USB connections

Example: `exit`

```
Telnet session closed
```

or

```
Cannot exit this Terminal
```

`setting.save`

Summary: Saves the current settings to non-volatile memory. Without this command, the settings are lost on reset or power loss.

Example: `setting.save`

```
Saved settings
```

`setting.username`

Summary: Change the device username

Example: `setting.username=newuser`

```
Username:newuser
Password:password
```

`setting.password`

Summary: Change the device password

Example: `setting.username=newpassword`

```
Username:admin
Password:newpassword
```

`setting.name`

Summary: Change the device name. This is also the NETBIOS name. NETBIOS names must be 15 or fewer characters.

Example: `setting.name=BAX_ROUTER6`

```
Name: BAX_ROUTER6
```

setting.session

Summary: Changes the text session number. The current file will be used until a save command is issued or the device autosaves. Permitted range is 0 to 65535.

Example: `setting.session=12345`

```
Text session: 12345
```

setting.usb.stream

Summary: Enable of disable the stream of the USB serial port. If turned off, ALL COMMUNICATIONS ARE SUSPENDED including the response to this setting

Example: `setting.usb.stream=1`

```
USB:
Mode: 0
Setting: Turned on
Status: Streaming
```

setting.usb.mode

Summary: Change the stream output mode to text (0) or binary slip encoded (1)

Example: `setting.usb.mode=0`

```
USB:
Mode: 0
Setting: Turned on
Status: Streaming
```

setting.telnet.stream

Summary: Enable or disable the stream of the telnet server (0 is off)

Example: `setting.telnet.stream=0`

```
Telnet
Mode:0
Setting: Turned on
Status: Enabled
```

setting.telnet.mode

Summary: Change the stream mode of the telnet connection to text (0) or binary slip (1)

Example: `setting.telnet.mode=0`

```
Telnet
Mode:0
Setting: Turned on
Status: Enabled
```

setting.file.stream

Summary: Enable or disable logging to file

Example: `setting.file.stream=0`

```
File:
Mode: 0
Setting: Turned off
Status: Card present
```

setting.file.mode

Summary: Change the stream mode of the log file to text (0) or binary raw (1)

Example: `setting.file.mode=1`

```
File:
Mode: 1
Setting: Turned off
Status: Card present
```

setting.udp.stream

Summary: Enable or disable the udp output (0=off)

Example: `setting.udp.stream=0`

```
Udp:
Mode: 0
Setting: Turned on
Status: Enabled
```

setting.udp.mode

Summary: Change the stream mode of the telnet connection to text (0) or binary slip (1)

Example: `setting.udp.mode=0`

```
Udp:
Mode: 0
Setting: Turned on
Status: Enabled
```

`setting.websocket.stream`

Summary: Enable or disable the websocket output (0=off)

Example: `setting.websocket.stream=0`

```
Websocket:
Mode: 0
Setting: Turned on
Status: Enabled
```

Complex commands

These commands are complex in nature and should be used only by users who fully understand the devices functionality, or who are familiar with the source code.

`mount`

Summary: Request to mount the memory card so it is available to the PC connection. If data logging is running, this is not safe. When logging the disk will appear write protected to the operating system.

Example: `mount=1` or `mount=0`

The first number represents the mount state, the second number is the write protect state.

```
Mounted: 1, 0
```

or

```
Mounted: 0, 0
```

`format`

Summary: Request an SD card format. ALL DATA WILL BE LOST. The number on failure is the

internal error code.

Example: `format`

```
Formatted
```

or

```
Format failed: 00
```

`hash`

Summary: Request the binary hash for opening a UDP session. Debug purposes only

Example: `hash`

```
3C24B1A1C87BAA6290049ACCB387A556AA58C1299B38C099E57F5C68ECECC0EB6207B8A6309D99272E565BDFE80842E
```

`setting.bax.subnet`

Summary: Change the subnet of the radio, accepts hex input.

Example: `setting.bax.subnet=4201`

```
Si44 Radio: On
Subnet:4201
Mask:FF00
```

`setting.bax.mask`

Summary: Change the OR mask of the radio subnet, accepts hex input. The mask is used in the same way as an ethernet subnet mask upon the subnet value.

Example: `setting.bax.mask=FF01`

```
Si44 Radio: On
Subnet:4201
Mask:FF01
```

`mac!`

Summary: This is used to change the device mac address. Value is not latched unless saved and device is reset. IEEE format used.

Example: `mac!=12:34:56:78:90:ab`

```
ETHERNET: Disconnected
MAC:12:34:56:78:90:AB
IP: 169.254.1.1
MSK:255:255:0:0
```

`bax_test`

Summary: Sets the bax radio to go into a pre-defined test mode. Reception is halted (0=test off, 1=CW carrier, 2=PN9 modulation pattern).

Example: `bax_test=1`

```
Bax->cw
```

Developed at Newcastle University, UK. Available under CC 3.0 BY Attribution License
Docs built with MkDocs • Theme based on Read the Docs

Commands

This section describes the commands which can be issued to a BuildAX ENV sensor. For information on connecting to the ENV sensors, please refer to the Connection Guide.

Command strings are followed by `[CR][LF]` (carriage return, line feed) in all cases; meaning that you should press the `Enter` key to send a command after typing it.

After most commands complete, the sensor will print its new operating status, and the list of basic commands. This will look like:

```
Settings:
Device:42609215
Name:dig0001
packet_header = 66
device_group = 96
device_id = 37397
batt_min = 2200
tx_power = 20
tx_channel = 87
pir_threshold = 20
pir_suspect_val = 10
pir_suspect_count = 8
pir_suspect_release = 8
pir_disarm_time = 24
pir_led_time = 8
sw_led_time = 8
sensor_settle_time = 8
credit_interval = 468
max_pir_credits = 10
max_switch_credits = 10
sensor_interval = 2343
num_tx_repeats = 1
Commands:save | defaults | name | renew | reset | sample | stream | exit (esc)
```

BuildAX ENV Command List

Basic commands

The following commands may be sent over the UART connection to perform an action on the device:

`name`

Summary: Set the device name to transmit on pairing

Example: `name=dig-bax0001`

```
Done.
```

`save`

Summary: Save current settings to non-volatile memory

Example: `save`

```
Saved.
```

`defaults`

Summary: Reset all settings to defaults and save them

Example: `defaults`

```
Done.
```

`renew`

Summary: Reset all settings, id and encryption data

Example: `renew`

```
Done.
```

`reset`

Summary: Reset the device, unsaved changes discarded

Example: `reset`

```
Startup.
Initial battery: 1663mV
Setting loaded ok.
```

`sample`

Summary: Sample the sensors and print their values

Example: `sample`

```
Sample:
Temp:24.7
Humidity:38.3, 302
Light:145
Batt:1663
PIR:165
SW:1
```

`stream`

Summary: Stream the sensor values in the same format as above

Example: `stream`

```
<Button/key exit>
Sample:
Temp:24.7
Humidity:38.7, 316
Light:145
Batt:1663
PIR:159
SW:1
```

`exit`

Summary: Exit command mode, use current (possibly unsaved) settings now

Example: `exit`

```
Ok.

Turn off
```

Complex commands

These commands are complex in nature and should be used only by users who fully understand the devices functionality, or are familiar with the source code.

Command	Description
`packet_header=66`	ASCII 'B' (0x42, 66) code to identify BAX packets
`device_group=162`	Integer form of bits 16 to 24 of identifier, used for subnet mask
`device_id=62199`	Integer form of bottom 16 bits of address
`batt_min=2100`	Minimum battery state before sensor disables itself (millivolts mV)

Command	Description
`tx_power=10`	Transmit power in dBm
`pir_threshold=20`	High threshold for known movement
`pir_suspect_val=6`	Low threshold for suspected movement
`pir_suspect_count=4`	Number of suspected movements to reach threshold and make event
`pir_suspect_release=8`	Time in each suspect count persists for †
`pir_disarm_time=8`	Time during which the PIR sensor will not create a new event (if triggered) †
`pir_led_time=8`	Time for which the PIR led illuminates for †
`sw_led_time=8`	Time for which the magnetic switch led illuminated for †
`sensor_settle_time=8`	Time after led illumination before continuing to monitor sensor events †
`credit_interval=234`	Time between granting event transmit credits †
`max_pir_credits=5`	Maximum number of PIR transmit credits
`max_switch_credits=5`	Maximum number of magnetic switch TX (transmit) credits
`sensor_interval=234`	Time between sampling and transmitting environment sensors †
`num_tx_repeats=1`	Number of environment sensor packets to send (repeats)

† Times
Times are in Ticks which are ~128 ms. To work out the ticks required from a time *in seconds*, multiply by 1000 then divide by 128 (rounding any decimal places to the nearest integer).

Example Timings

The following values are multiples of 128ms ticks, rounded to the nearest integer. Exact timing is not possible with extremely low-power hardware such as the ENV, so there will be some drift over time. This also helps to prevent radio data packet collisions in deployments with many sensors, as no two sensors will drift by the same amount.

Time	Value (ticks)
1 second	8
5 seconds	39
30 seconds	234
1 minute	469 938 2 minutes
5 minutes	2344
30 minutes	14063

Developed at Newcastle University, UK. Available under CC 3.0 BY Attribution License
Docs built with MkDocs • Theme based on Read the Docs

BuildAX Data File Format

CSV

CSV Format data is logged to files named `LOGxxxxx.TXT` on the SD card when the logging format is set to plaintext mode. It can also be streamed over the serial console and telnet.

A CSV formatted packet from a BuildAX Environment Sensor looks like this:

```
2014/08/29,09:26:05,Bedroom,-48,1,171,0,2911,50.25,220,241,13876,36099,25
```

Field List

Field Name	Data Type
Date	YYYY/MM/DD formatted ISO 8601 date of the packet
Time	HH:MM:SS formatted time the packet was received at
Name	The sensor's name, or its address if none has been assigned
RSSI	Received Signal Strength Indication, in dBm
Type	Received packet type. See Packet Types for full definition
Sequence No.	Packet identifier (sensors send packets incrementally)
Transmit Power	Sensor-configured transmission power in dBm
Battery	Battery level, in millivolts, of the BuildAX sensor
Humidity	Relative Humidity (percentage of the saturation vapour pressure)
Temp	Temperature in degrees Celsius (°C), multiplied by 10.
Light	Luminous flux measurement in Lux
PIR Count	Activation counts of the Passive Infra-red sensor
PIR Energy	PIR energy last captured
Switch	Magnetic switch triggers on the device

Packet Types

The type of a BuildAX packet indicates its transmission source. Listed here are the currently defined packet types:

ID	Type
0	Encryption packet type (not seen in CSV output)
1	Normal Packet RX'd at sensor transmit interval

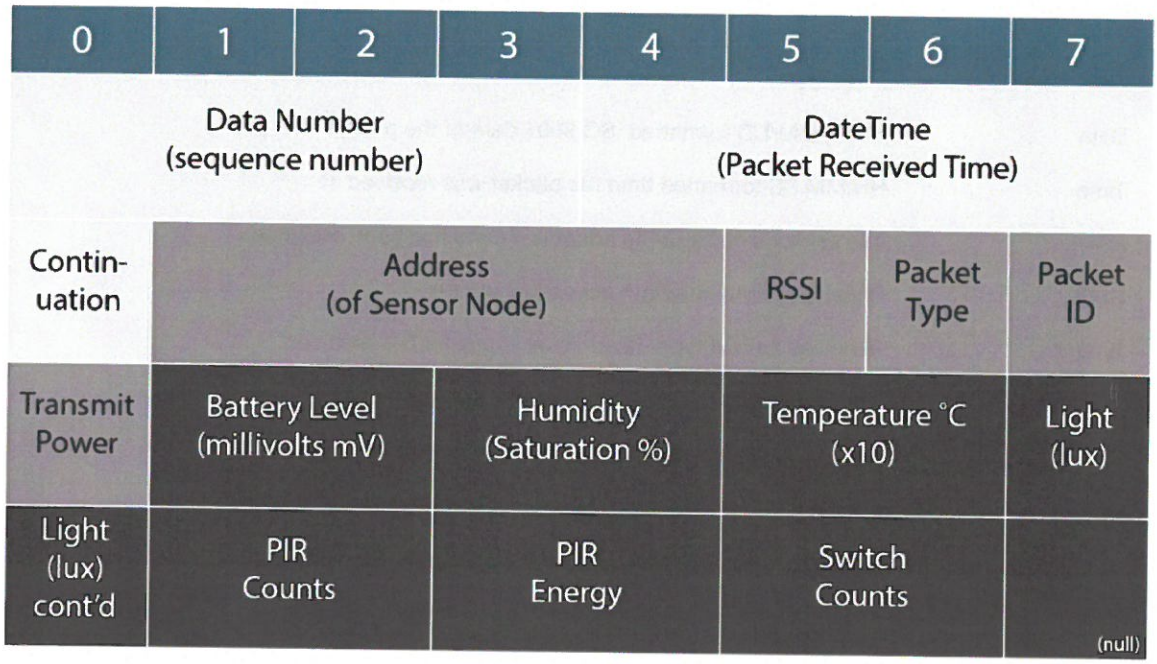
ID	Type
2	Packet sent when PIR sensor triggered
3	Packet sent when Magnetic Switch triggered

Further packet types will be defined as more types of BuildAX sensor are developed.

Binary

The binary format can be converted into CSV using the `bax2csv` utility found in the OpenMovement repository. The utility is designed to compile on most machines with a Make and GCC environment.

Binary BuildAX packets are 32 bytes long. The layout of the binary packet is similar to how it is rendered in CSV. The data type widths for an ENV sensor are described below:



The data portion of the packet (darkest part in the diagram) is the part which will vary depending on the packet type. This diagram shows a packet from a BuildAX environment sensor (types 1,2,3), which would vary with (for example) a CO2 sensor.

Other Files

There are a number of different files on the SD card which are used by the LRS:

- `BAX_INFO.BIN`
  - This binary file contains 16-byte long entries for each sensor paired with the LRS. Contains the encryption key (5 bytes), and the name of each sensor (11 bytes).
- `ERRORS.TXT`
  - Error log for the router. Although this file is named "Errors", it is also used for logging a timestamp and reset code each time the router is powered on (e.g. reset due to power loss).
- `DATxxxxx.BIN`
  - Data files containing logged *binary* sensor data

- `LOGxxxxx.TXT`
  - Text files containing logged *plaintext* sensor data
- `BAX_SURV.BMP`
  - (Specialist option). If the LRS is flashed with a firmware supporting radio-survey mode, it will generate this file containing a survey of the radio spectrum noise at your location. Note that the router is not currently distributed with this firmware as it is a specialist option.

Developed at Newcastle University, UK. Available under CC 3.0 BY Attribution License

Docs built with MkDocs • Theme based on Read the Docs

BuildAX LRS Admin Panel

A web configuration interface is provided to make setting up a BuildAX installation easier. This document gives an overview of the admin panel functionality.

Browser Support:
All modern, standards-compliant web browsers are supported. Older browsers may experience reduced functionality, for example Microsoft Internet Explorer version 8 or lower. A warning message will be displayed in this case.

Setting up

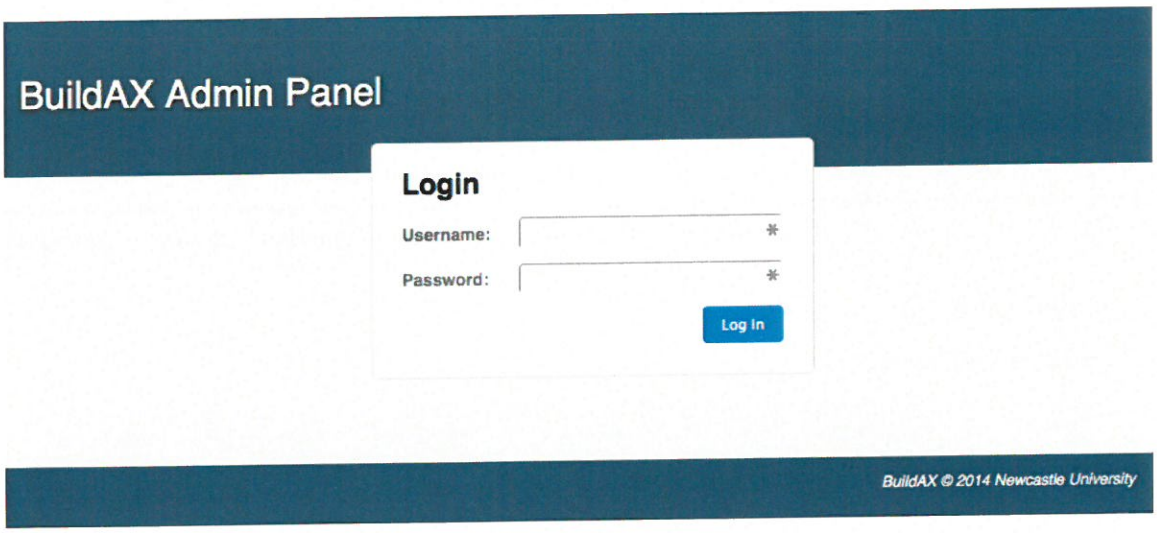
As described in the Quick-Start guide, the data files for the web-administration interface must be copied onto the SD card before use. A 404 error will be reported if the files are not found in the correct location on-disk, under the WWW folder. The data files for the most recent version can be downloaded here.

Upgrading the Web Interface:
The web data files may be periodically updated as new features become available. The new files will be uploaded to the Open Movement repository, and can be upgraded on the LRS by deleting the previous version from the SD card, and copying over the new files.

Note that you may also need to upgrade the firmware on the LRS if you experience error messages after upgrading the web UI files.

Login

The LRS configuration interface is protected with a username and password. These should be updated from the Admin panel when you first log into the site.



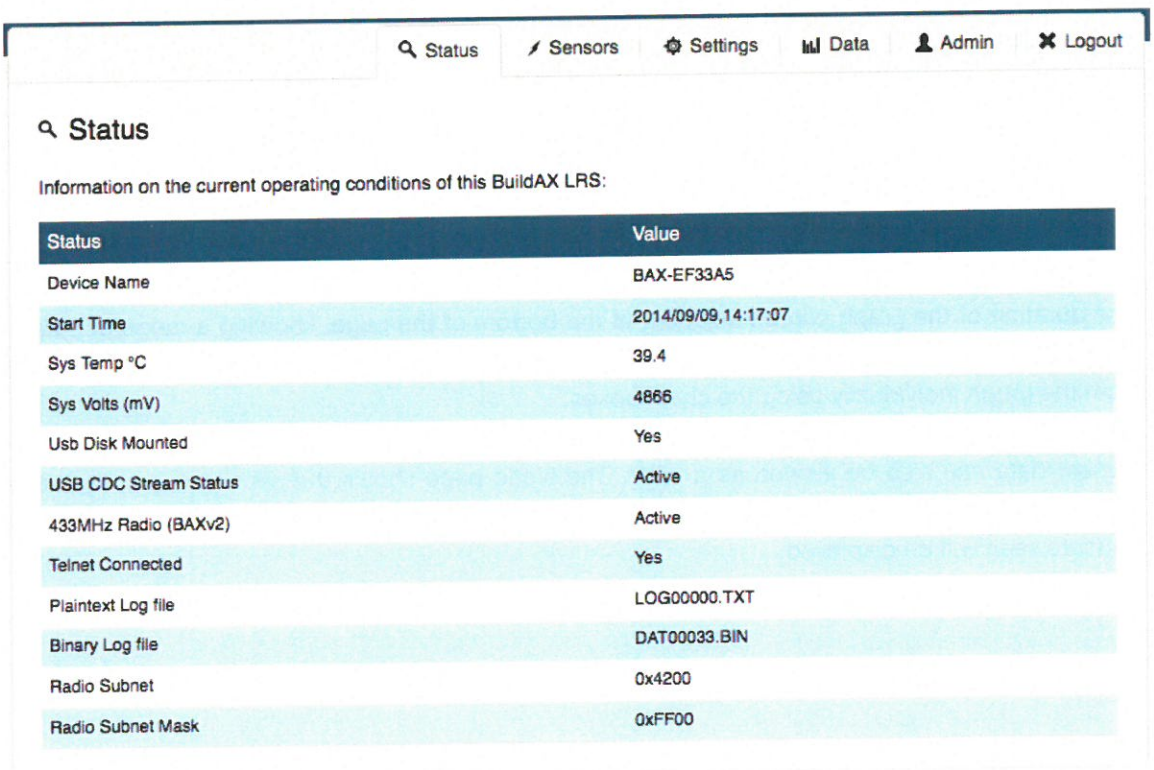
The default login credentials for the BuildAX LRS are:

- Username: admin
- Password: password

For information on how to change these credentials, see here. Note also that the login form on this page is not yet protected by SSL encryption (the https:// scheme), so using a unique password is encouraged. A future firmware update will be deployed to enable this functionality for self-signed SSL certificates.

Status

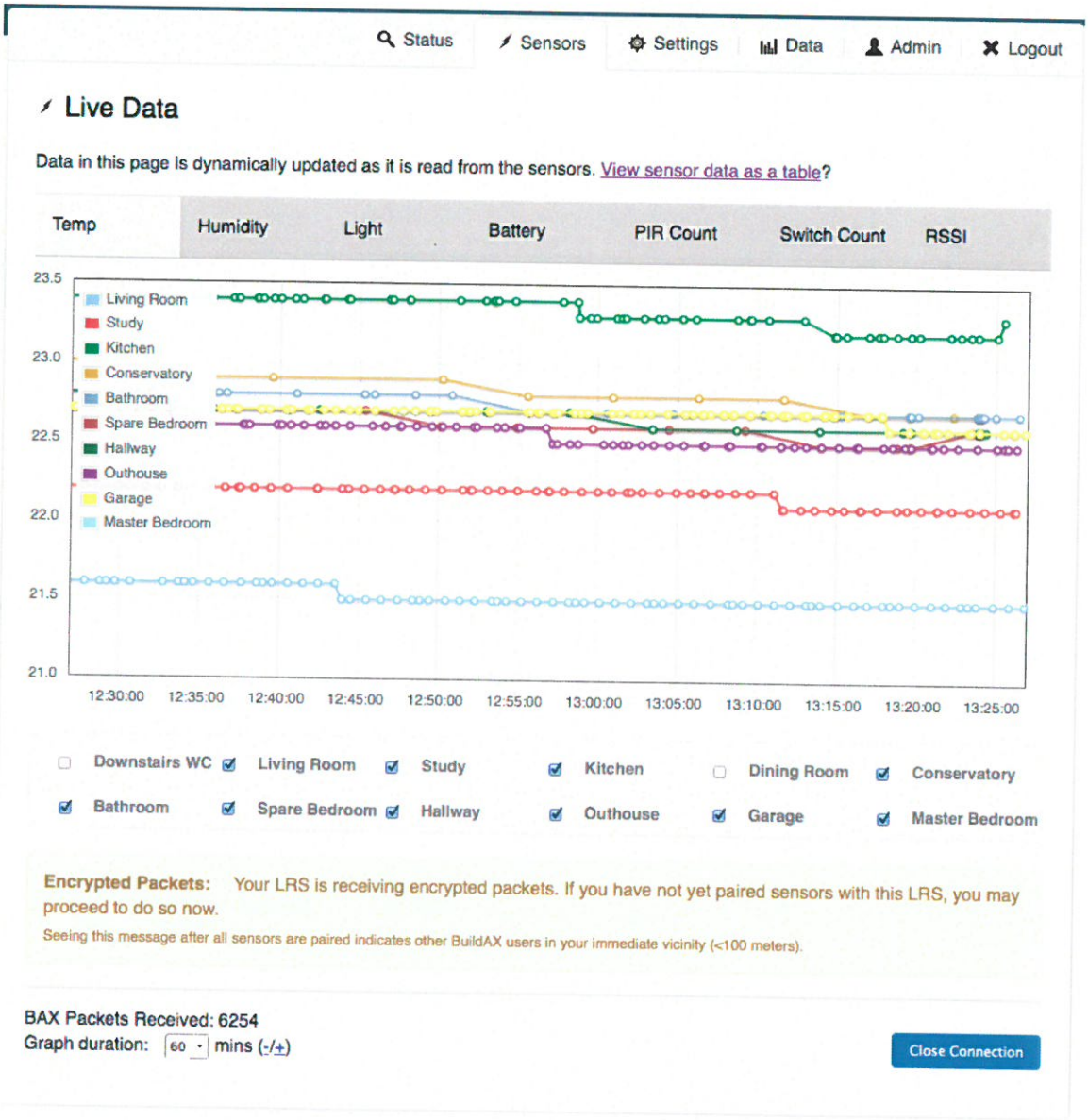
The Status tab is the landing page for the system, and allows users to verify the running state of the device, including the log file (binary or plaintext) and the status of LRS peripherals.



Status	Value
Device Name	BAX-EF33A5
Start Time	2014/09/09,14:17:07
Sys Temp °C	39.4
Sys Volts (mV)	4866
Usb Disk Mounted	Yes
USB CDC Stream Status	Active
433MHz Radio (BAXv2)	Active
Telnet Connected	Yes
Plaintext Log file	LOG00000.TXT
Binary Log file	DAT00033.BIN
Radio Subnet	0x4200
Radio Subnet Mask	0xFF00

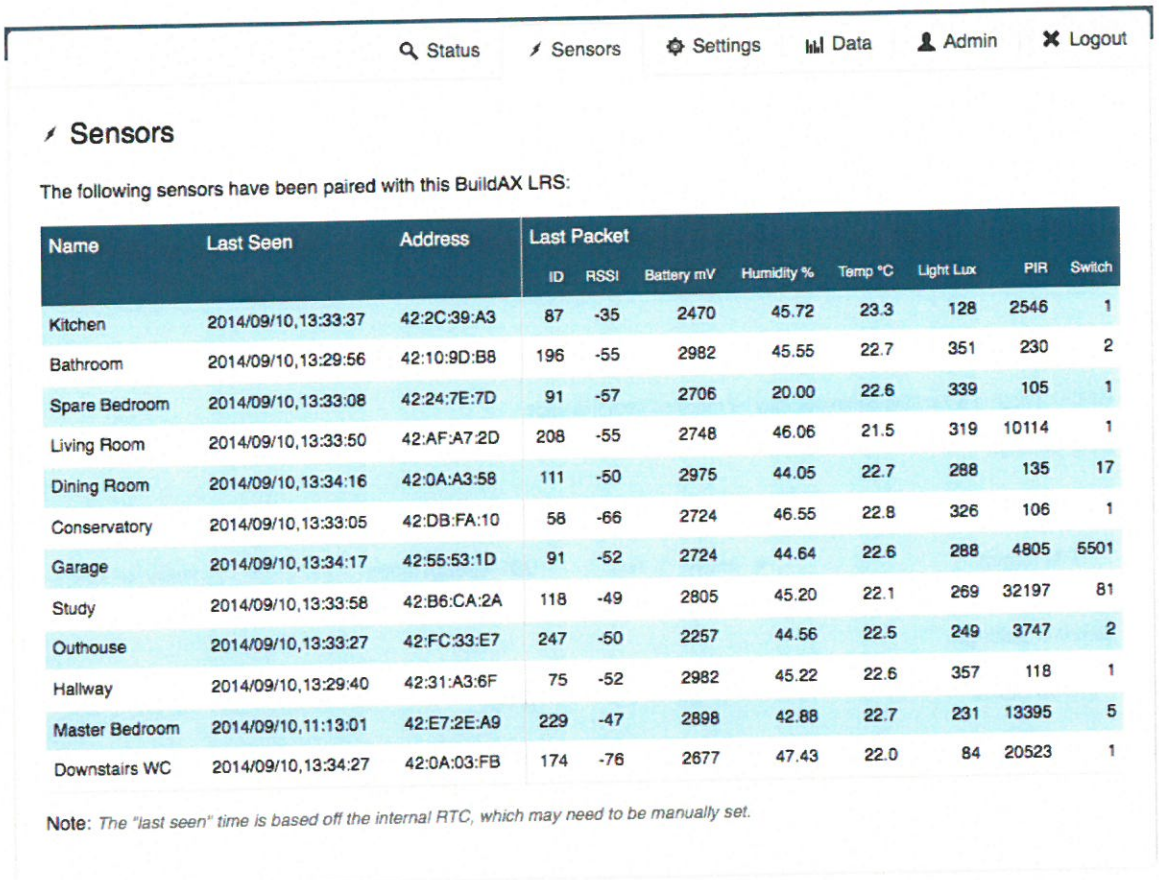
Sensors

The sensor display page shows a live data stream from the sensors. This can be useful while paring sensors, as they will appear immediately on the graph when connected to the LRS.



The duration of the graph can be adjusted at the bottom of the page, showing a range of data up to 60 minutes. The number of sensors displayed is not limited, and a sensors can be excluded from the graph individually using the checkboxes.

Sensor data can also be viewed as a table. The static page shows the sensors currently loaded into memory on the LRS, along with their details and the last values received. Only the last 20 sensors seen will be displayed.



Status Sensors Settings Data Admin Logout

Sensors

The following sensors have been paired with this BuildAX LRS:

Name	Last Seen	Address	Last Packet							
			ID	RSSI	Battery mV	Humidity %	Temp °C	Light Lux	PIR	Switch
Kitchen	2014/09/10,13:33:37	42:2C:39:A3	87	-35	2470	45.72	23.3	128	2546	1
Bathroom	2014/09/10,13:29:56	42:10:9D:B8	196	-55	2982	45.55	22.7	351	230	2
Spare Bedroom	2014/09/10,13:33:08	42:24:7E:7D	91	-57	2706	20.00	22.6	339	105	1
Living Room	2014/09/10,13:33:50	42:AF:A7:2D	208	-55	2748	46.06	21.5	319	10114	1
Dining Room	2014/09/10,13:34:16	42:0A:A3:58	111	-50	2975	44.05	22.7	288	135	17
Conservatory	2014/09/10,13:33:05	42:DB:FA:10	58	-66	2724	46.55	22.8	326	106	1
Garage	2014/09/10,13:34:17	42:55:53:1D	91	-52	2724	44.64	22.6	288	4805	5501
Study	2014/09/10,13:33:58	42:B6:CA:2A	118	-49	2805	45.20	22.1	269	32197	81
Outhouse	2014/09/10,13:33:27	42:FC:33:E7	247	-50	2257	44.56	22.5	249	3747	2
Hallway	2014/09/10,13:29:40	42:31:A3:6F	75	-52	2982	45.22	22.6	357	118	1
Master Bedroom	2014/09/10,11:13:01	42:E7:2E:A9	229	-47	2898	42.88	22.7	231	13395	5
Downstairs WC	2014/09/10,13:34:27	42:0A:03:FB	174	-76	2677	47.43	22.0	84	20523	1

Note: *The "last seen" time is based off the internal RTC, which may need to be manually set.*

Name

The name of the BuildAX sensor device can be set when it is programmed using an FTDI cable, and will be sent as part of the encryption key transmission at pairing time. This is stored along with the key on the root of the SD card (external storage), and as such can be subsequently modified. The maximum length of a sensor node name is 32 characters.

Last Seen

The last time a packet was received from a sensor, as defined by the internal RTC on the BuildAX LRS.

Address

The unique hardware identifier of the BuildAX sensor. This is generated at random by the sensor, but the pool of values (48-bit) is large enough to make collisions very unlikely.

Last Packet

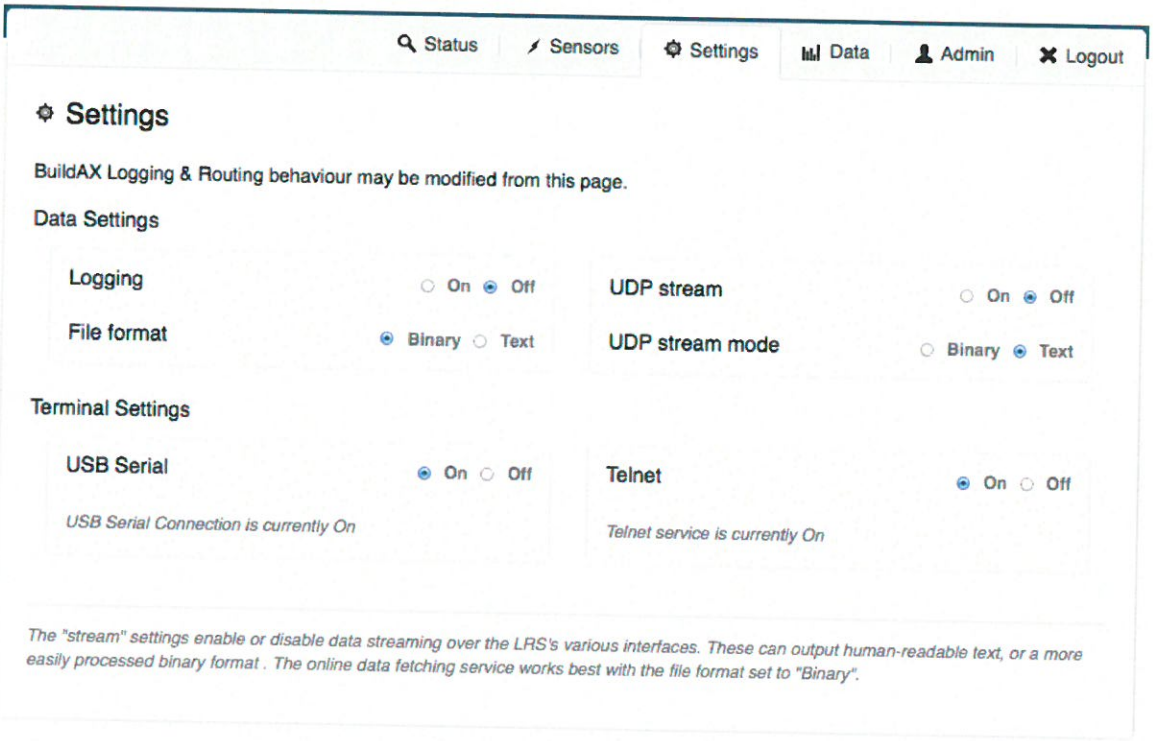
The last packet field is broken down into the individual values broadcast by the sensor. For a breakdown of the sensor values, see the Field List section.

Limitations

Sensors appear in this table while their encryption keys are loaded into RAM on the LRS. As the amount of RAM on the device is limited, only the most recent 20 sensors will be displayed in this table. This may become possible with future hardware revisions.

Settings

The BuildAX LRS has the ability to route streams of data to various different data sources. The Settings page is used to turn on/off different streams, and switch between streaming formats (binary or plaintext). The form is arranged by stream type.



When a setting has been changed, it needs to be applied before it becomes active. The Update button will appear when a setting needs to be applied.

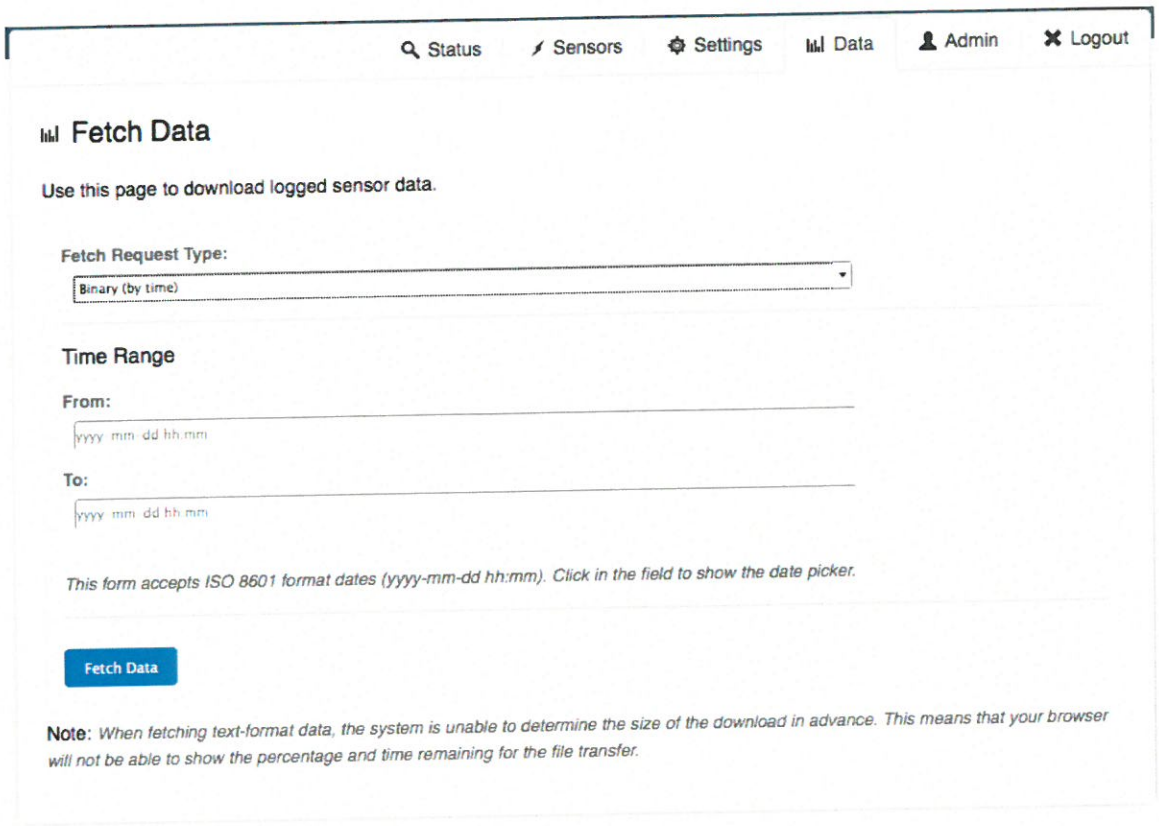


If successfully applied, you will see the confirmation dialog appear below the settings form:



Fetch

Data can be retrieved from the LRS remotely via the fetch interface. The form on the Data tab is used to format requests to the LRS.



Fetch types relate directly to the logging types- if your LRS is logging in plaintext mode you should use the plaintext fetch- the same goes for if the LRS is logging in binary.

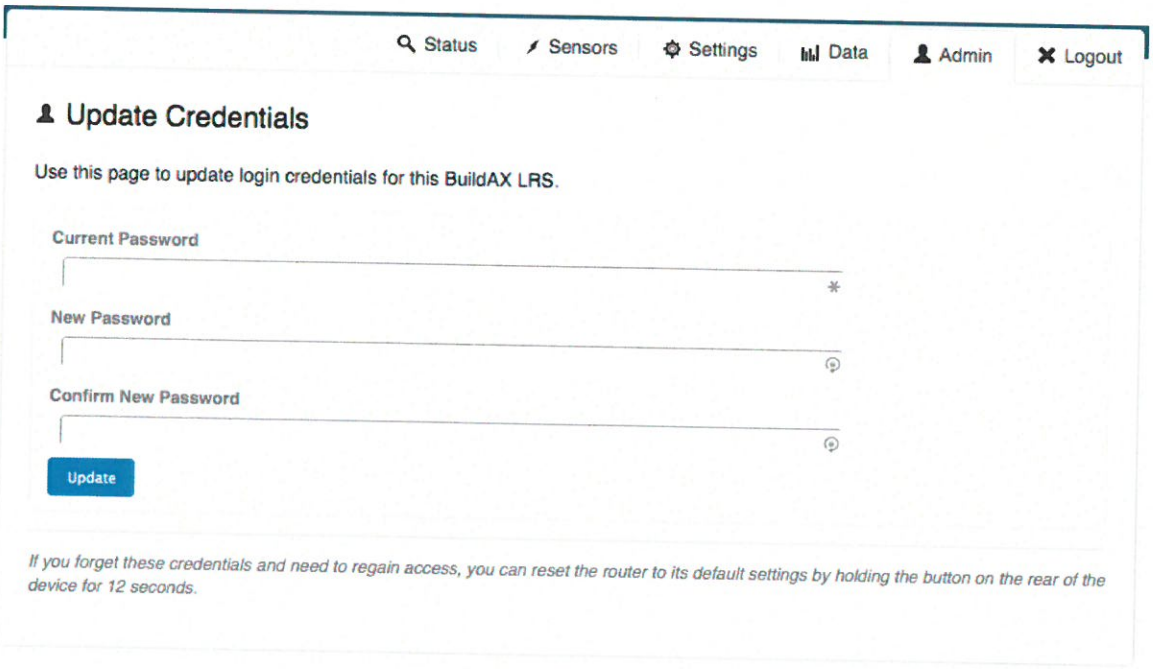
The Fetch form exposes four different ways to request data:

- Date/Time
- File
- Sample Number (as a range)
- Last N Samples (from most recent backwards)

In addition, if the LRS determines that the parameters entered do not translate to an acceptable range, an error 400 (Bad Request) will be returned.

Admin

To update the login credentials for the LRS, simply fill in the form on the Admin tab. There are no hard-coded complexity requirements for the password, but normal convention should be followed. The maximum password length is 32 characters.



Reset Password:
If the password is forgotten, it must be reset by physically accessing the LRS and loading the default settings. This reset procedure can be performed by holding down the button on the rear of the LRS for 12 seconds.

Developed at Newcastle University, UK. Available under CC 3.0 BY Attribution License
Docs built with MkDocs • Theme based on Read the Docs

BuildAX LRS Developer API

The BuildAX LRS can be programatically configured via a remote connection. API calls to the BuildAX LRS are made as RESTful HTTP requests. For you, this means that setting state on the LRS is performed via `POST` requests, and `GET` requests are used to retrieve data.

The examples in this document use the `wget` command to communicate with the web server on the BuildAX LRS. On Linux `wget` is usually installed by default, but can be downloaded for other OSes (including MacOSX and Windows) from the GNU website.

An equivalent software with much the same functionality as wget is `curl`, which may be used in preference to `wget`.

Logging In

The Web API uses a unique session key (a *cookie*) to authenticate users. All API sessions start when this session key is requested by making a POST request to the BuildAX LRS.

The user agent must first authenticate with the LRS and obtain a session cookie before further REST API requests can be made:

Method	Resource	Parameters	Response
`POST`	/WWW/login.htm	user, pass	303 redirect with SID= session cookie.

Example `wget` command (a backslash is used to separate the command onto multiple lines):

```
wget --save-cookies cookies.txt \
    --delete-after \
    --no-http-keep-alive \
    --post-data 'user=admin&pass=password' \
    --keep-session-cookies \
    http://your-bax/WWW/login.htm
```

Equivalent `curl` command:

```
curl --silent --output /dev/null \
    -c cookies.txt \
    --data 'user=admin&pass=password' \
    http://your-bax/WWW/login.htm
```

Changing Settings

Settings are passed to the server in the same format as the text command mode. Please refer to the LRS commands for a full description of the settings commands which may be sent to the LRS.

Settings can be concatenated with an ampersand (&) in the POST body to change multiple settings simultaneously.

Method	Resource	Parameters	Response
POST	/WWW/a_settng.htm	Setting arguments in the format specified above	200 OK Set-Cookie: update=ok

```
wget --load-cookies cookies.txt \
    --post-data 'setting.usb.stream=0' \
    http://your-bax/WWW/a\_settng.htm
```

Valid settings types are: `file`, `usb`, `telnet`, and `udp`.

Fetching Data

Data fetching is performed via GET request, so parameters are passed in the URL.

```
http://your-bax/fetch?type=BT&start=449622000&end=485434800
```

There are three parameters in a fetch request: a type, and the start/end range specifiers.

Method	Resource	Valid Parameters	Response
GET	/fetch	type, start, end	200 OK (and download)

Fetch Types

The **type** parameter may be specified as one of the following:

Type	Fetch mode
BT	Binary / Time range fetch
TT	Text / Time
BF	Binary / File
TF	Text / File
BS	Binary / Sample range *(note: text/sample range is not implemented)*
LS	Last N Samples *(note: specify an empty 'end' parameter with this request)*

Fetch Parameters

The parameters **start** and **end** are always present, but their value changes based on the type of request.

Fetch Type	Range specifiers
Time Range	Start and end are specified in *seconds after the epoch 2000-01-01* matching the RTC timestamp

Fetch Type	Range specifiers
File Range	Start and end are data file numbers as stored on the external flash card storage
Sample Range	Start and end are a range of packets numbers received from the sensors
Last N Samples	Start is the number of most recent samples. Specify end as an empty string.

Time Format

To calculate the time range, take the Unix timestamp for the time required (a standard time representation) and subtract the Unix timestamp for the date `2000-01-01`. Eg:

```
Current Unix Time – (2000-01-01 00:00) = BuildAX RTC Timestamp
       1407940334 –      946684800     = 461255534
```

The reason for this difference is that the hardware real-time-clock (RTC) on the BuildAX LRS works with an epoch of the year 2000 and a 32-bit timestamp, meaning dates up to 2068 can be represented internally.

Example `wget` commands

> **Login:**
> Remember that you must first run the Login command to get a session cookie!

Get data file 0 from the SD card (no more than 5 files (10MB) may be requested in one download):

```
wget --load-cookies cookies.txt \
    --no-http-keep-alive \
    --content-disposition \
    http://your-bax/fetch?type=BF\&start=0\&end=0
```

Get last 500 samples:

```
wget --load-cookies cookies.txt \
    --no-http-keep-alive \
    --content-disposition \
    http://your-bax/fetch?type=LS&start=500&end=
```

Get data within a time range:

```
wget --load-cookies cookies.txt \
    --no-http-keep-alive \
    --content-disposition \
    http://your-bax/fetch?type=BT\&start=449622000\&end=485434800
```

One possible 'gotcha' is that the ampersand (&) symbol has a special meaning in most command terminals, and must be escaped with a backslash (as in the above examples) so that the command

is not forked.

Generic Commands

In addition to the commands listed above, some commands listed in the LRS Commands section can be sent using this interface.

Method	Resource	Valid Parameters	Response
POST	/cmd	LRS command	200 OK (even if the command failed)

A specific limitation of this HTTP interface is that it is not currently possible to include feedback from this command in the HTTP response. The server will return the response code, along with an empty response body. This means that commands (e.g. `status`) which are only useful for their response will not be useful to call through this interface. Commands which perform some action (e.g. `save`) are the use case for this feature.

Example:

```
http://your-bax/cmd?save
```

Additionally, HTTP special characters will not be parsed correctly until firmware v1.4. This will make it possible to set the RTC time via the REST interface.

Developed at Newcastle University, UK. Available under CC 3.0 BY Attribution License
Docs built with MkDocs · Theme based on Read the Docs

Device Firmware

Firmware updates for BuildAX devices will be provided as a .hex file. This document describes the process of flashing the devices with updated firmware.

Warning:
Updating the firmware is a potentially risky process. If you flash a device with an incorrect firmware file (or use a third-party bootloader application which has a bug) you could render it permanently unusable!

Downloads

Tested bootloader software for the BAX devices is provided for Windows OSes in the OpenMovement repository, along with the firmware for each device.

Download BuildAX Firmware here. Ensure you download the correct firmware for your device.

BuildAX LRS

Web Interface Upgrades:
The web interface data files do not also need to be upgraded when a firmware upgrade is applied, but this is recommended in order to access new and improved functionality. See here for more information.

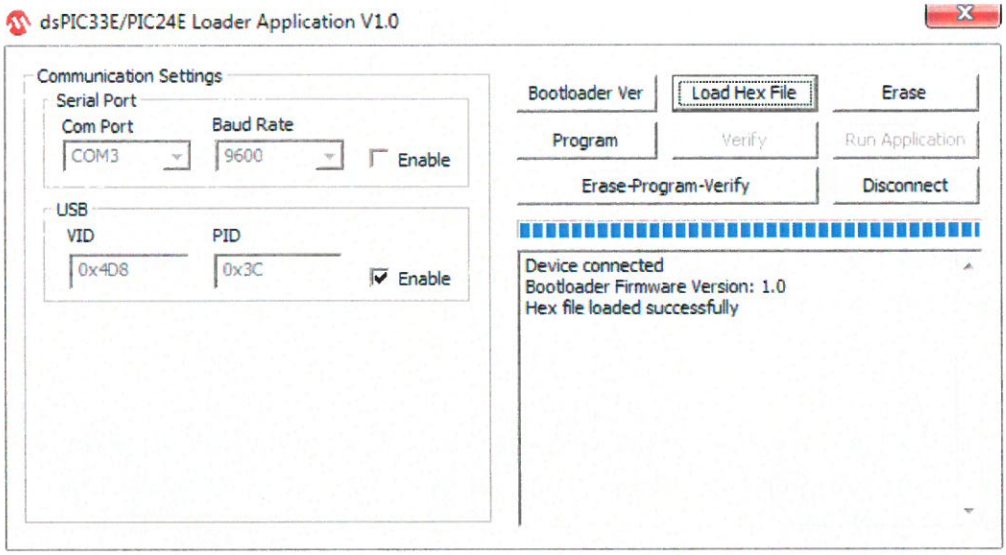
Put the LRS into bootloader mode

Remove the USB power connector from the LRS. Hold down the button on the rear of the device and connect the USB cable between the PC and the LRS. The LRS should enter bootloader mode, indicated by the LEDs counting up from 1-7 continuously.

Connecting to the PC

Open the firmware flashing tool provided in the OpenMovement repository. Tools for other platforms may be available online.

Enable the USB checkbox. The Vendor ID should be `0x4D8`, and the Product ID `0x03C`. Now hit the "Connect" button. The tool should show that it has connected to the device.



Next, click "Load Hex File" and select the new firmware. **Double check** that you have selected the correct firmware file for the LRS. Then, click "Erase-Program-Verify", and the new firmware will be written to the device.

When this is completed, "Run Application" will cause the LRS to exit bootloader mode. If using a USB wall power adaptor, it is now safe to remove the device from the PC and re-connect it to its own adapter.

Reset ALL Settings (re-generate MAC address)

To permanently reset all settings, including assigning a new random MAC address, hold the button on the rear of the LRS and click "Run Application" at this stage. Note that this may result in the LRS not being able to access the network if you have used the device's old MAC address to assign it an IP.

Video

BuildAX ENV Sensor

Writing a firmware `.hex` file onto a BuildAX ENV sensor requires a PIC18f bootloader application. A Windows executable will be provided in the OpenMovement repository for this purpose (AN1310ui.exe).

> **FTDI cable:**
> To connect to a BuildAX sensor, a suitable FTDI cable is required. Please refer to the Connection Guide for details of the cable and driver installation.

Put the ENV into bootloader mode

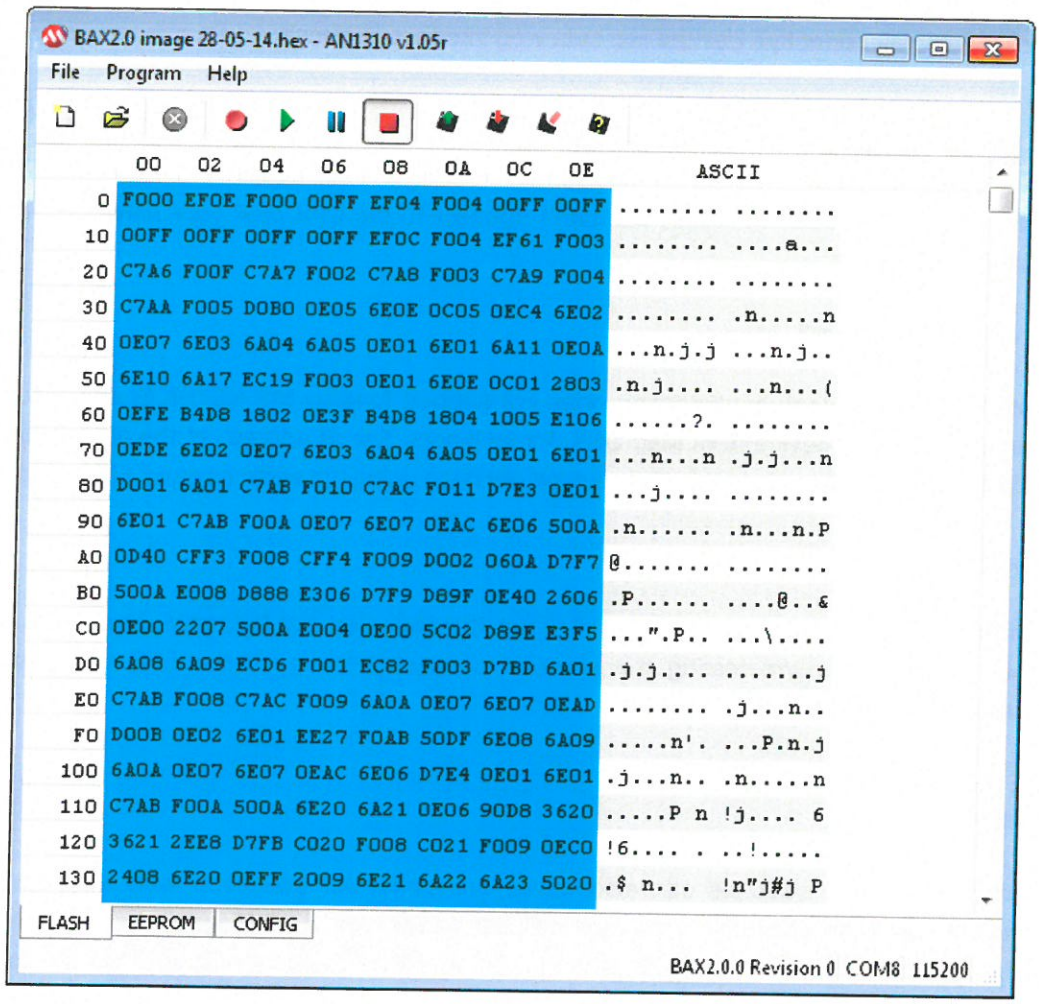
1. Remove one of the AAA batteries from the ENV.
2. Push the FTDI cable onto the 3-pin header on the device, with the orange wire facing the corner of the device (the USB end should be connected to your PC).

3. Re-insert the battery
4. Push the button next to the programming header while the green LED on the sensor is lit.

The LED will flash red/green briefly before lighting solid red. This indicates that the sensor is now in bootloader mode.

Writing new firmware

While the red LED is lit on the device, you should connect the PC bootloader application to the sensor. In the AN1310 PIC18f bootloader, this is the red "stop" button in the toolbar. Then, load the new firmware into the application using the "open" menu. The application should now look similar to this:



Hit the "Write Device" button on the toolbar (rectangle with red arrow) to program the device with the new firmware. When completed, the status bar should show "Write complete (Flash 11.897s)" or similar time. Finally, hit the "Run Application Firmware" button (the green "play" arrow) to exit bootloader mode.

Video

Developed at Newcastle University, UK. Available under CC 3.0 BY Attribution License
Docs built with MkDocs · Theme based on Read the Docs

Frequently Asked Questions

My sensor won't pair up!

On the sensor, press the button next to the battery compartment. The green LED on the sensor should light on the front to indicate a pairing packet has been sent. Things to check:

1. Is the LRS in Pairing Mode?
2. Ensure the batteries are inserted correctly.
3. Do other sensors pair correctly? Use the live data web interface to check.

Networking issues

Please read the Network Access section. Still can't connect to the LRS over your network? Try these troubleshooting steps:

Using the NETBIOS Address?

It is possible that your network configuration does not support loading MS NETBIOS addresses (you are connecting to the device from Mac OSX, for instance). The IP address of the device can be used as a fallback in this case, and can always be obtained using the serial `status` command.

Networking LED

Is the networking LED (LED 3) on the LRS lit? If it is not, this indicates that the networking stack is un-initialised. Try re-inserting/re-seating the ethernet connector to see if the LRS connects. Possible reasons for this LED not being lit:

- Network not assigning IP address
- Bad Ethernet cable

Some gigabit unmanaged switches can also cause this behaviour (observed with a Netgear GS108). The ethernet chip (enc28j60) used by the BuildAX LRS runs at 10 Megabits/second, and some unmanaged switches do not negotiate the speed and wire protocol of a 10M connection properly. Trying a different cable (for example, CAT5 instead of 5e or 6) may mitigate this problem (untested).

Corporate/Academic Networks

Are you on a corporate/academic network? Your network administrator may require that the device MAC address is registered before it can be assigned an address. When programming firmware it is possible to reset the MAC address by holding down the button and clicking "Run Program" in the bootloader tool. If this is the case you will have to obtain the new MAC via the serial connection and re-register the device.

Have you just flashed new firmware to the device?

It is possible that when the firmware is written to the device and the device reset, the MAC address which the LRS uses to negotiate with the network will be re-generated. To do this intentionally, hold down the button on the LRS when clicking "Run Application" from the bootloader app.

To change the MAC address, use the `mac!` command followed by `setting.save` and `reset`.

How do I retrieve data from the device?

Please see Fetching Data in the developer guide for more information on how to fetch data over the network. To directly connect to the LRS as a Mass Storage device, see USB Mass Storage.

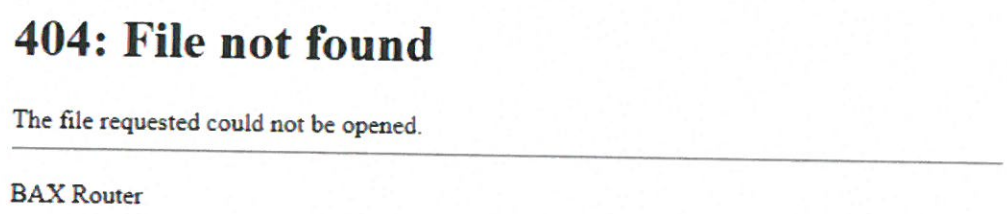
How do I read the binary data format?

The `bax2csv` utility can be used to convert the binary data format into a usable plaintext csv format. Please read the Data Files documentation for more info.

Web interface issues

A number of HTTP error codes may be encountered with the web interface. The following is not an exhaustive list, but these errors are the most common.

404 File Not Found



404: File not found

The file requested could not be opened.

BAX Router

This error indicates that the files for the web interface on the SD card are missing or in the wrong place. These should be copied to the SD card manually before running, and should be placed under the directory "WWW".

400 Bad Request

This is an error code indicating that the LRS was unable to process a data fetch request. This will occur when a request is made for data which does not exist on the LRS, for example:

- Data files are missing due to manual deletion
- Data has been requested from a time range when no data was logged
- The fetch has been specified for the wrong logging mode (e.g. requesting text when the LRS is logging in binary mode)

This error may also be seen when automating fetch requests with `wget` or `curl`, possible reasons include:

1. The parameters provided in the URL were not correctly formatted and cannot be recognised
2. The ampersands (&) in the URL are not correctly escaped (`\&`) and the command has forked
3. A request has been made which is impossible for the LRS to fulfill (missing data file?)

I cannot connect to a device using Serial!

There are a number of things you can check here:

- Am I connecting to the correct serial device (COM port in Windows)?
- Is the USB cable a known good (working) cable?

LRS Specific checks:

- Have I installed a USB Serial driver?
- Is LED 2 (the USB Serial status LED) lit?

ENV / other BAX sensors specific checks:

- Have I installed the driver for my FTDI cable?
- Are the 3 pins of my FTDI connector connected in the correct order?

I have tried the above and still cannot solve my problem

The first thing you should do is reset the router, and check if this solves the problem. Locally, simply unplug the device and plug it in again. This can also be done remotely by issuing the `reset` command over telnet or serial.

If this does not solve your problem, you should check if there are any assertions in the ERRORS.txt file on the SD card with timestamps around the time you saw this problem. "Reset" lines in this file are usually fine (they are added when the LRS starts up to indicate the reset type), unless you have encountered a General Exception. The 4-digit number indicates the type of reset and is taken from the value of the reset register.

If you encounter a line reading "ASSERT" and a file/line number, this might indicate a firmware issue. Please send us this file and a description of what you were trying to do, and steps to reproduce your problem (if possible).

Will BuildAX interfere with my other equipment/BMS /WiFi/etc

As BuildAX runs on the 433MHz ISM radio band, it will not interfere with (or be disrupted by) networks like 802.11n WiFi which run on the 2.4GHz band.

Additionally, the design of the radio protocol means that our packets are very short- this reduces

the probability that they will interfere with other 433MHz protocols (or each other).

Finally, the radio chipsets used have been EMC-tested (and type-approved) in order for the hardware to receive the CE marking. This means that BuildAX conforms to the EEA directives for electromagnetic noise emission.

Developed at Newcastle University, UK. Available under CC 3.0 BY Attribution License
Docs built with MkDocs • Theme based on Read the Docs