Electric Mayhem

xBR Interface Control Document

(ICD)

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Rev | Date | Author | Description |
| 0.1 |  | GJS | First draft |
| 0.2 | 7/1/11 | GJS | GET events  Added more explanation to GET events  Added packet sequence number to read event information  Added radio frequency to read event information  Added a "max" parameter to GET events  PUSH update\_stream – limited to one URL  Time set via Network Time Protocol (NTP)  RID (reader ID) changed to reader-name  XID changed to XLRID (xBand Long Range ID)  Reader name defaults to MAC address |
| 0.3 | 7/8/11 | GJS | Added optional ‘max’ and ‘since’ parameters to POST update\_stream  Removed URL parameter from DELETE update\_stream  Added POST hello to xBRC  Added more text in introductory sections  Added ‘event number’ field. ‘since’ parameter now specifies event number instead of time. |
| 0.4 | 7/15/11 | GJS | Shortened tag names on events |
| 0.5 | 7/16/11 | GJS | "since" parameter changed to "after" and it acquires events after the given event number.  Changed time format to ISO 8601 standard (hh:mm:ss.SSS for milliseconds). |
| 0.6 | 7/29/11 | GJS | Merged GET reader/name.json and GET reader/version.json into a single service GET reader/info.json. The payload in POST hello matches the payload in GET reader/info.json. |
| 0.7 | 7/29/11 | GJS | Added xBR, xFP and Park Entry xFP sections for events and commands specific each type of reader. |
| 1 | 8/12/11 | GJS | Added "port" and "next eno" to info.json and hello payload.  Changed reader controller "hello" address to 192.168.0.2:8080 |
| 2 | 8/16/11 | GJS | Changed most POSTs to PUTs  Added a "type" field to all events  Added xBio version to park-entry xFP info.json  Added park-entry commands for setting and playing sounds  Added command to set park-entry mode. |
| 3 | 9/6/11 | GJS | Changed PUT biometric/match data from multipart to JSON.  Changed "park-entry" reader type to "xFP+xBio" |
| 4 | 9/23/11 | GJS | Added "tail" parameter to GET events.json  Removed "PUT event" and "DELETE event"  Added "num events", "oldest event", and "push URL" to GET info.json  Replaced PUT application/software with PUT upgrade command.  Added PUT biometric/firmware and a firmware upgrade completed event for the park entry tap reader. |
| 5 | 9/27/11 | MJW | Change fonts & paragraph layout.  Separated out "hello" address.  Added MAC format.  Removed GET events/last.json  Define xband/commands.  Define xBR specific data for reader-info. |
| 6 | 10/5/11 | GJS | Added PUT biometric/cancel for xFP |
| 7 |  | GJS | Made the templates in the biometric/match command optional. |
| 8 | 10/19/11 | GJS | Updated reader/info message for park entry xFP |
| 9 | 10/26/11 | GJS | Added biometric/image URL to retrieve xBio image data |
| 10 | 10/31/11 | GJS | Added xbio-scan-error event type for park entry DAP. |
| 11 | 11/17/11 | MJW | Added Reader/Time Page for setting the time |
| 12 | 12/19/11 | GJS | Removed xFP items making this an xBR specific document. |
| 13 | 01/10/12 | MJW | Moved common message contents to end. Heavily revised "Commands". Added additional fields section to /reader/info. Fixed formatting issues. |
| 14 | 1/24/12 | MJW | Reader/info.json: radio info is more verbose; commands details refined. xBand data – changed to raw, non-translated hexadecimal. |
| 15 | 2/2/2012 | MJW | PUT update: Added method to update software from .ipk repositories |
| 16 | 2/6/2012 | MJW | Split PUT update into PUT install (application package) and PUT upgrade (update all applications based on repository dependencies) |
| 17 | 2/14/2012 | MJW | Detail out the band command format and correct incorrect example in section 5.3 |
| 18 | 2/28/2012 | MJW | **SOFTWARE RELEASE xBR-0.0.9**  Correct GET /reader/info.json example. Note areas that are not implemented. |
| 19 | 5/2/2012 | GJS | **SOFTWARE RELEASE xBR-0.2.0**  Added following commands:  PUT /xband/beacon  DELETE /xband/beacon  PUT /xband/filter  And modified PUT/xband/commands  Added ‘dbg’ and ‘wkup’ values to ping events |
| 20 | 5/16/2012 | GJS | **SOFTWARE RELEASE xBR-1.0.0**  Added "min-ss" to POST update\_stream, and to GET events.json.  Added DELETE filter service.  Added receiver sequence number to ping events.  Added PUT /events and DELETE /events  Added GET /filter |
| 21 | 6/4/2012 | GJS | **SOFTWARE RELEASE xBR-1.0.1**  Updated "hello" message and "GET reader/info" service contents. |
| 22 | 6/11/2012 | GJS | SOFTWARE RELEASE xBR-1.0.3  Changed the values under "events" in the GET /reader/info service and ‘hello’ payload. |
| 23 | 7/9/2012 | GJS | Clarified effect of XLRID parameter of 0 in GET events.  Added DLETE /xband/output.  Added GET /xband/commands  Added GET /xband/beacon |
| 24 | 7/31/2012 | CJW | Updated "PUT upgrade" command details. |
| 25 | 8/8/2012 | GJS | Added diagnostics events. |
| 26 | 8/13/2012 | CJW | Updated "GET /xband/commands" and "GET /xband/beacon" commands. Added "GET/PUT /xband/options" commands. |
| 27 | 9/17/2012 | CJW | Added "DELETE /xbrc" command. Updated section 2.6 "Controller (xBRC) Address". |
| 28 | 9/24/2012 | GJS | Added clarification on when diagnostics event is generated. |
| 29 | 9/26/2012 | GJS | Added GET /diagnostics. Added clarification on use of PUT /application/reset. Changed "min\_ss" to "min-ss" in PUT /filter (to match code) |
| 30 | 10/1/12 | CJW | Updated the Table of Contents. Changed "url" parameter to "path" in "PUT /update" command. |
| 31 | 10/2/12 | GJS | Added "update stream" and "time" to reader/info and hello messages. |
| 32 | 10/4/12 | JBM | Fixed wording on Time section and GET /xband/commands. Fixed numbering in body of text. Replaced url with path in PUT /upgrade section. Added document number for xBRC ICD. Fixed /hello message format description. Corrected formatting inconsistencies. |
| 33 | 10/11/12 | JBM | Updated PUT /upgrade to include "hwids". Various spelling and formatting updates. Updated ToC. |

Table of Contents

Table of Contents

[1 1 Introduction 8](#_Toc337735348)

[1.1 Purpose 8](#_Toc337735349)

[1.2 Scope 8](#_Toc337735350)

[1.3 Background 8](#_Toc337735351)

[1.4 Definitions 8](#_Toc337735352)

[1.5 Reference 8](#_Toc337735353)

[2 2 General Formatting Notes 9](#_Toc337735354)

[2.1 Time 9](#_Toc337735355)

[2.2 MAC 9](#_Toc337735356)

[2.3 Payload Formats 9](#_Toc337735357)

[2.4 POST vs. PUT 9](#_Toc337735358)

[2.5 HTTP addresses 9](#_Toc337735359)

[2.6 Controller (xBRC) Address 9](#_Toc337735360)

[2.7 Git Related Versions 10](#_Toc337735361)

[3 3 xBR to xBRC Messages 11](#_Toc337735362)

[3.1 /hello 11](#_Toc337735363)

[4 4 xBR Events 11](#_Toc337735364)

[4.1 Events Payload 11](#_Toc337735365)

[4.2 Event Types 12](#_Toc337735366)

[4.2.1 Ping Event 12](#_Toc337735367)

[4.2.2 Diagnostics Event 13](#_Toc337735368)

[5 5 xBR Messages 15](#_Toc337735369)

[5.1 GET /events 15](#_Toc337735370)

[5.2 PUT /events 15](#_Toc337735371)

[5.3 DELETE /events 17](#_Toc337735372)

[5.4 PUT /reader/name 17](#_Toc337735373)

[5.5 GET /reader/info.json 17](#_Toc337735374)

[5.6 GET /diagnostics 19](#_Toc337735375)

[5.7 POST /update\_stream 20](#_Toc337735376)

[5.8 DELETE /update\_stream 20](#_Toc337735377)

[5.9 PUT /application/reset 20](#_Toc337735378)

[5.10 PUT /system/reset 20](#_Toc337735379)

[5.11 PUT /install 21](#_Toc337735380)

[5.12 PUT /upgrade 21](#_Toc337735381)

[5.13 PUT /time 22](#_Toc337735382)

[5.14 GET /time.json 22](#_Toc337735383)

[5.15 PUT /filter 23](#_Toc337735384)

[5.16 GET /filter.json 23](#_Toc337735385)

[5.17 DELETE /filter 23](#_Toc337735386)

[5.18 PUT /xband/beacon 23](#_Toc337735387)

[5.19 GET /xband/beacon 24](#_Toc337735388)

[5.20 DELETE /xband/beacon 24](#_Toc337735389)

[5.21 PUT /xband/commands 24](#_Toc337735390)

[5.22 GET /xband/commands 26](#_Toc337735391)

[5.23 DELETE /xband/commands 26](#_Toc337735392)

[5.24 PUT /xband/options 26](#_Toc337735393)

[5.25 GET /xband/options 27](#_Toc337735394)

[5.26 DELETE /xband/output 27](#_Toc337735395)

[5.27 PUT /xbrc 27](#_Toc337735396)

[5.28 DELETE /xbrc 27](#_Toc337735397)

# 1 Introduction

## Purpose

Define interface between the xBR and xBRC.

## Scope

This document describes the network interface to long range reader (xBR).

## Background

The xBand Reader or xBR is an embedded Linux platform equipped with radios for communicating with xBands. The primary job of most xBRs is to receive xBand radio packets, but the xBR can also be used to send packets to the xBands to change some xBand operating parameters.

xBRs are arranged logically in groups with each group connected to a common xBand Reader Controller or xBRC. xBand radio packets received by the xBRs are queued up along with signal strength information and forwarded to the xBRC. The xBRC uses this information to detect the presence and location of xBands.

The interface between the xBR and xBRC is a RESTful HTTP interface. Payloads are typically formatted as JSON. The received radio messages or "events" are normally pushed to the controller, but may also be pulled via the HTTP interface.

The xBR acquires its IP address and the address of the xBRC assigned to it via DHCP. Under normal operation upon start up, the xBR sends an initial "hello" message to the xBRC and the xBRC responds by sending the appropriate messages to configure the xBR.

This document describes the networking interface exposed by the xBR. Sections 2 and 3 define the format of messages between the xBR and the xBRC controller, followed by the section 4, which defines parts of those messages which are in common in those messages.

## Definitions

## Reference

|  |  |
| --- | --- |
| Document Name & Version | Relationship |
| 900-0058 – xBRC Interface Control Document (ICD) | Documents interface between xBRC and computers on system bus. |
| 900-0003 – xBand Interface Control Document (ICD) | Details xBand commands and returned data. |

# 2 General Formatting Notes

## Time

Date and Time is represented using the ISO 8601 standard with the exception of an optional milliseconds field appended to the end.

#### Example:

2011-06-20T13:41:00.891

## MAC

The MAC (Media Access Control address or Ethernet Hardware Address (EHA)) is formatted according to IEEE MAC-48, using ‘:’ as a separator character, in transmission order. See <http://en.wikipedia.org/wiki/MAC_address>, Notational Conventions.

#### Example:

01:23:45:67:89:ab

## Payload Formats

Unless otherwise specified, all messages are formatted as JSON objects.

## POST vs. PUT

We always use PUT to the xBRC, because the web server software used on the xBRC expects POST to include form data and doesn’t work so well in our case.

The readers are actually agnostic as to whether the verb is PUT or POST, they are treated identically.

## HTTP addresses

HTTP addresses are specified in normal URL manner, in 4-byte dot separated format. Port must be specified, unless it is 80.

#### Example:

http://123.45.067.8:8024

## Controller (xBRC) Address

The reader can retrieve its xBRC address through three different means. These methods in order of priority are:

1. Manually by setting the "xbrc url" setting in the configuration file located at /var/mayhem/grover.conf.
2. Through DHCP: The reader requests Vendor-Specific Information (option 43), with Vendor Class Identifier (option 60) set to "SYNAPSE". If the DHCP server responds with corresponding options, the http address specified in Vendor-Specific Information will be written to /var/lib/dhcp/xbrc-url. The reader application reads this file each time it attempts to contact the xBRC.
3. Through the xBRMS: The reader will query the local DNS server for a SRV record under the service name "\_rest.\_tcp.xbrms.<domain name>" (the domain name is determined at run time) which should give the reader a list of xBRMS servers that can be contacted. The reader will then send a "hello" message (see section **Error! Reference source not found.**3.1**.**) to identify itself to the xBRMS. If the xBRMS has an xBRC in which to direct the reader, it will send a "PUT /xbrc" command to the reader to configure its xBRC address.

When the reader receives a "PUT /xbrc" command, it will save the URL in persistent storage. If the above three methods are not successful in setting the address then the reader will default to this address until instructed otherwise.

## Git Related Versions

Git version names are created from executing "git describe" for each source code repository. The general format is:

<name>-<ver>[-<offset>[-g<hash>[-dirty]]]

The tag (<name>-<ver>) is manually entered when the application is released. A version that has trailing information –(<offset>, etc.) is an in-house engineering version. "–dirty" versions should not exist in the field. The <ver> consists of three dot separated numbers, first changes with major functional change, usually hardware changes. The second denotes a production release, and the third is a non-production change. A change to either of first 2 numbers causes lessor number to restart at 0. All production versions have a zero for the third number.

See Example, below.

|  |  |
| --- | --- |
| Name | Description |
| Name | Name of the application, part of the closest tagged ancestor in the repo |
| Ver | Application version, part of the closest tagged ancestor in the repo |
| Offset | Number of generations newer than the tagged version in the repo |
| Hash | When offset is given (non-tagged version), -g<hash> is also provided, which specifies exactly what version in the repo this release is. |
| Dirty | "-dirty" specifies that the release does not directly correspond to the git repo, but that there are local changes that possibly affect the application. |

#### Example

xBR-0.0.8-231-g1eb662c-dirty

\\_\_\_\_\_\_/ \/ \\_\_\_\_\_/ \---- There were local changes when application was built

| | | and so it does not match the git repository exactly

| | |

| | \----------- Repository hash of application version

| |

| \----------------- Number of versions since the last release (last tag)

|

\------------------------- Tag. The latest tag, from the last release.

# 3 xBR to xBRC Messages

## /hello

Message sent periodically to notify the xBRC that the xBR is online and available. The JSON payload matches the reply contents of the GET reader/info service except it does not include "xbrc url" or the radio information (radio versions and calibration values).

#### Message Format

|  |  |
| --- | --- |
| Header | Value |
| Method | PUT |
| URI | /hello |
| HTTP VERSION | 1.1 |
| HOST | <xBRC-Address>. See section 2.5 |
| Content type | Text/plain |
| <content> | The content of the hello message is identical to the content of the GET reader/info service with the exceptions noted above. See GET reader/info. |

# 4 xBR Events

The primary events from the xBR are for received xBand pings. In addition to xBand pings, the xBR also sends out a periodic "diagnostic" event. Events are queued up by the xBR and can be pulled by the xBRC via this command, or the xBRC can be instructed to push events with the "PUT update\_stream" command.

## Events Payload

Whether events are pulled via a GET, or pushed, the payload looks the same and looks like this:

{

"reader-name": <reader name>,

"events":

[

Events…

]

}

#### Fields

|  |  |
| --- | --- |
| *Name* | *Description* |
| reader name | Name assigned to reader. Mac address provided if no reader name assigned. |
| events | An array of events sorted by time of arrival. The format of each event depends on the type of event. |

## Event Types

### Ping Event

The most common events are for radio packets received from xBands. Each event has the following fields:

#### Fields

|  |  |
| --- | --- |
| Name | Description |
| eno | Event number. Increments for each event sent to controller. |
| time | Time packet was received. Millisecond resolution. (See section 2.1) |
| XLRID | xBand long range ID |
| pno | The packet number from the xBand radio packet. 8 bit number that rolls over (0 – 255) |
| dbg | Value from the ‘debug’ byte of the ping packet |
| wkup | Value from the ‘wakeups’ byte of the ping packet |
| freq | Frequency packet was received on: 2401, 2424, 2450,or 2476 |
| chan | Channel packet was received on (0 or 1). Each reader has two antennas connected to separate sets of radios, creating two channels. |
| ss | Signal strength of received signal in dB (theoretically -110 to -27, practical limits -90 to -35) |
| rxno | Receiver packet number. 8 bit number that is incremented by the receiving radio for each packet received. Rolls over at 255 (0-255). |

#### Example:

{

"reader-name": "entry-1",

"events":

[

{

"eno": 54832,

"time": "2011-07-08T13:41:03.750",

"XLRID": "12A36E03ABD32103",

"pno": 5,

"dbg" : 3,

"wkup" : 103,

"freq": 2404,

"chan": 1,

"ss": -60,

"rxno": 132

},

{

"eno": 54833,

"time": "2011-07-08T13:41:04.124",

"XLRID": "54E03ABD32103323",

"pno": 255,

"dbg" : 3,

"wkup" : 43,

"freq": 2405,

"chan": 0,

"ss": -79,

"rxno": 42

}

]

}

### Diagnostics Event

This event occurs every 60 seconds and relays diagnostic information about the reader. The event is not generated if the link to the xBRC is down, that is it is only generated if events are being successfully pushed to a URL set via PUT /update\_stream. The diagnostics event contents can also be acquired with GET /diagnostics.

Each diagnostic event has the following fields:

#### Fields

|  |  |
| --- | --- |
| *Name* | *Description* |
| eno | Event number |
| type | "xfp-diagnostics" |
| time | Date and time |
| status | Status of dap reader. Either "Green", "Yellow" or "Red". |
| status msg | Reason for problem if status is other than "Green", otherwise empty string. |
| RadioN status | Status of radio N ("Green", "Yellow" or "Red"). N ranges from 0 to 8 with radio 8 being the transmit radio. |
| RadioN status msg | Reason for any problem for radio N if the status is other than "Green". N ranges from 0 to 8. |

#### Example

{

"eno": 54832,

"type": "xfp-diagnostics",

"time": "2011-07-08T13:41:03.750",

"status" : "Red",

"status msg" : "Radio interface error",

"Radio0 status" : "Green",

"Radio0 status msg" : "",

"Radio1 status" : "Green",

"Radio1 status msg" : "",

"Radio2 status" : "Green",

"Radio2 status msg" : "",

"Radio3 status" : "Green",

"Radio3 status msg" : "",

"Radio4 status" : "Green",

"Radio4 status msg" : "",

"Radio5 status" : "Red",

"Radio5 status msg" : "Radio interface error",

"Radio6 status" : "Green",

"Radio6 status msg" : "",

"Radio7 status" : "Green",

"Radio7 status msg" : "",

"Radio8 status" : "Green",

"Radio8 status msg" : ""

}

# 5 xBR Messages

## GET /events

Retrieve events. If this command is issued without specifying an "after" parameter, then the command will return events starting after the last event previously retrieved. When a reader is connected to a single host, the host could retrieve all events by periodically issuing GET events commands with no parameters. The "after" parameter provides a way for multiple hosts to retrieve events. Both "after" and ‘last event’ are independent of values associated with /stream (sections 5.7,5.8)

#### Parameters

|  |  |  |
| --- | --- | --- |
| Name | Optional | Description |
| after | Yes | Retrieves events after the given event number. |
| max | Yes | Maximum number of events to return. If the number of available events exceeds ‘max’, then the earliest events are returned. |
| tail | Yes | 0 = pull oldest events, 1 = pull newest events. |
| min-ss | Yes | Minimum signal strength. Only return ping events with a signal strength equal or greater than this value if provided. |
| XLRID | Yes | Only retrieve events for the given xBand Long Range ID. If this parameter is not given, or if it is set to 0, then events are returned for all xBands. |

#### Examples (xBRC requests)

GET http://4.3.2.1:8080/events.json

GET http://192.168.1.1:8024/events.json?XLRID=12A36E03ABD3

GET http://192.168.10.9:8077/events.json?max=10&after=2000

#### Response

The response is an events object. See section the previous Events section.

## PUT /events

This command exists for testing purposes, allowing events to be inserted into the xBR event queue and sent to the xBRC. The events are provided in a JSON payload.

#### Payload

The payload must consist of one or more events. The format of the payload closely resembles the events payload sent from the xBRC with the following differences:

1. "eno" and "time" fields are ignored because the event number and time stamp are provided by the xBR.
2. A new optional field called "delay" can be provided to insert a delay between the queuing of events. The events are added to the event queue in the order they appear in the payload, with delays inserted if provided.
3. Only the "XLRID" field is required. All other fields have default values if not provided.

#### Fields (event)

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Required | Default | Description |
| delay | no | 0 | Delay in milliseconds between previous event and this event. |
| XLRID | yes | N/A | xBand long range ID |
| pno | no | 0 | The packet number from the xBand radio packet. 8 bit number that rolls over (0 – 255) |
| *dbg* | no | 0 | Value from the ‘debug’ byte of the ping packet |
| *wkup* | no | 0 | Value from the ‘wakeups’ byte of the ping packet |
| freq | no | 2405 | Frequency packet was received on: 2401, 2424, 2450,or 2476 |
| chan | no | 0 | Channel packet was received on (0 or 1). Each reader has two antennas connected to separate sets of radios, creating two channels. |
| ss | no | ? | Signal strength of received signal in dB (theoretically -110 to -27, practical limits -90 to -35) |
| *rxno* | no | 0 | Receiver packet number. 8 bit number that is incremented by the receiving radio for each packet received. Rolls over at 255 (0-255). |

#### Example:

{

"events":

[

{

"XLRID": "54890E4FED380B32"

},

{

"delay": 1000,

"XLRID": "12A36E03ABD32103",

"pno": 5,

"dbg" : 3,

"wkup" : 103,

"freq": 2404,

"chan": 1,

"ss": -60,

"rxno": 132

},

{

"XLRID": "54E03ABD32103323",

"pno": 255,

"dbg" : 3,

"wkup" : 43,

"freq": 2405,

"chan": 0,

"ss": -79,

"rxno": 42

}

]

}

## DELETE /events

Delete any events staged to be added to the event queue by the PUT /events command, but not yet added to the event queue. Note that this will not remove events already in the event queue, but only removes those events still waiting to be added to the event queue.

(The event queue can be thought of as a queue of past events, while the PUT /events command can be used to stage future phony events. In that sense, this command removes staged future events, but not past events).

## PUT /reader/name

Set the name of the reader. The un-set reader name is "".

#### Parameters

|  |  |
| --- | --- |
| Name | Description |
| Name | Reader name |

#### Examples (xBRC requests)

PUT http://4.3.2.1:8080/reader/name?name=Charlie

PUT http://4.3.2.1:8080/reader/name?name=534

## GET /reader/info.json

Get information about the reader.

#### Parameters

None.

#### Response

The response holds a variety of information about the reader.

#### Fields

|  |  |
| --- | --- |
| Name | Description |
| HW type | The hardware type/version. |
| hello\_url | URL to which unit is sending hello messages (See section 2.5) |
| RadioX calibration | List of calibration values for each of the 8 receiver radios.  "radio <radio> calibration" :"<calibration value>" where radio = 0-7 and calibration value is -127 to +128.  If a calibration value is 0, the radio is un-calibrated. The normal range of calibrated values is around -110 to -90. |
| RadioX version | Version of firmware in this radio |
| TX radio version | Version of firmware in the transmit radio |
| linux version | Linux kernel version number |
| mac | Mac address of xBR |
| next eno | Event number to be assigned to the next event that occurs. |
| port | Listening port for HTTP services |
| radio driver version | Version of the radio driver software |
| reader name | Name assigned to the reader |
| reader type | Always "Long Range" for xBR |
| reader version | Software version of the xBR application code |
| time | Current date/time |
| update stream | URL to which the unit is sending stream messages (See section 2.5). Empty if there isn’t an active stream. |
| xbrc url | URL of the xBRC for this xBR. The xBR will periodically send "hello" messages to this URL, i.e. it will post to this URL + "/hello". |
| events | JSON object contains statistics of the internal data (circular buffer). Contains:  **capable** – Maximum number of events that can be stored in memory.  **stored** – total number of events stored in memory.  **queued (push)** – number of events queued up to be pushed to the update stream URL.  **queued (get)** – number of events queued up to be returned by GET /events service. |

#### Example

{

"HW type" : "xBR3",

"Radio0 calibration" : "-100",

"Radio0 version" : "001v",

"Radio1 calibration" : "-98",

"Radio1 version" : "001v",

"Radio2 calibration" : "-98",

"Radio2 version" : "001v",

"Radio3 calibration" : "-98",

"Radio3 version" : "001v",

"Radio4 calibration" : "-101",

"Radio4 version" : "001v",

"Radio5 calibration" : "-87",

"Radio5 version" : "001v",

"Radio6 calibration" : "-100",

"Radio6 version" : "001v",

"Radio7 calibration" : "-100",

"Radio7 version" : "001v",

"TX radio version" : "001v",

"linux version": "2.6.39",

"mac": "01:23:45:67:89:ab",

"next eno" : 2110,

"port": 8080,

"radio driver version" : "1.2.2-1901 2012May31-13:26:13 (ba9a43afcb)",

"reader name": "entry-1",

"reader type" : "Long Range",

"reader version" : "1.0.0-1915 2012 Jun 04-17:01:51 – master (316bef)",

"update stream" : "192.168.3.2:8080/stream"

"xbrc url" : "http://192.168.3.2:8080/hello",

"time" : "2011-06-23T18:30:23",

"events" : {

"capable" : 3000,

"queued (get)" : 1934,

"queued (push)" : 23,

"stored" : 3000,

},

}

## GET /diagnostics

Output the latest diagnostics. The format is in JSON and matches the format described previously for the Diagnostics Event, but does not include the "eno" and "time" fields.

## POST /update\_stream

This will cause the reader to begin posting events to a given URL at a specified frequency (See section 5.1**.**). Note that the URL to the xBRC supplied in this message does not include a protocol header (i.e. no "http://" prefix). See example.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Name | Optional | Description |
| url | No | URL to post updates to (SEE TEXT) |
| interval | Yes | Posting interval in milliseconds |
| max | Yes | Maximum number of events per POST |
| after | Yes | Retrieves events after the specified event number |
| min-ss | Yes | Minimum signal strength. Any ping events with signal strength less than this number are not sent to the update stream. |

#### Example:

PUT http://4.3.2.1/update\_stream?url=168.192.0.2/services/events&interval=2000

## DELETE /update\_stream

Command to stop sending events to a URL.

#### Parameters

None.

#### Example

DELETE http://4.3.2.1/update\_stream

## PUT /application/reset

Reset the reader application software.

NOTE: This command relies on the reader monitor script that is run as part of the normal reader start up script (i.e. "/etc/init.d/grover start"). Because of this, this command will not work if you are running the reader program from the command line.

#### Example

PUT http://4.3.2.1/application/reset

## PUT /system/reset

Resets the reader (restarts Linux)

#### Parameters

None.

#### Example

PUT http://4.3.2.1/system/reset

## PUT /install

Set URL to install a specific software package. After installing, the unit will be rebooted. Update will fail on unfulfilled dependencies.

#### Example

PUT http://4.3.2.1/install?url=http://168.192.0.2/release/product/app-1.1.1.1.ipk

## PUT /upgrade

Upgrade the device using "opkg" and a set of given repositories. Device will return the output of upgrade and reboot upon completion.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Name | Optional | Description |
| repos | *No* | Repositories information, an array of parameter objects – { "name": "<value>", "url": "<value>", "weight": "<value>"}  Details of the parameter fields for each repository follow. |
| name | *No* | *Repository Field:* Name of the source repository. |
| weight | *No* | *Repository Field:* Relative value of files in this repository compared to the other repositories listed. Higher values are selected first, in the case where a file can be found in multiple repositories. |
| path | *No* | *Repository Field:* Supplies the http address of the repository. |
| downgrade | Yes | Optional: Indicates that a downgrade is intended. Default is false. |
| hwids | *No* | Optional: Indicates all the hardware IDs that are supported by the firmware package. |

#### Example Content:

{

"repos":[

{

"name":"overo",

"path":"<http://192.168.73.56/reader/repos/overo>",

"weight":"51"

},

{

"name":"armv7a",

"path":"[http://192.168.73.56/reader/repos/ armv7a](http://192.168.73.56/reader/repos/%20armv7a%20) ",

"weight":"46"

},

{

"name":"all",

"path":"<http://192.168.73.56/reader/repos/all>",

"weight":"1"

},

{

"name":"xBR",

"path":"<http://192.168.73.56/reader/repos/xBR>",

"weight":"90"

}

],

"downgrade": false,

"hwids":[

"xBR",

"xBRv4"

]

}

## PUT /time

Set the time on the reader.

#### Parameters

|  |  |
| --- | --- |
| Name | Description |
| **Time** | Reader time (see section 2.1) |

#### Example

PUT http://4.3.2.1/reader/time?time=2011-06-20T13:41:00.891

## GET /time.json

Get the current date/time. (See section 2.1)

#### Example

GET http://4.3.2.1/reader/time

#### Response

{

"time": "2011-06-23T18:30:23"

}

## PUT /filter

Set a filter for accepting or ignoring pings.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Name | Optional | Description |
| XLRID | Yes | Set an xBand ID to filter on. If given, only pings from the given xBand will be selected. An xLRID of "all" restores collection of all pings. |
| min-ss | Yes | Minimum signal strength. Only pings with a signal strength above this value will be accepted. |

#### Example:

PUT http://4.3.2.1/filter?XLRID=2749D22AE6&min-ss=-60

## GET /filter.json

Returns the current filter settings.

#### Parameters

None.

#### Response

The response is composed of an info object (section 5.5) as well as the following additional fields:

#### Fields

|  |  |
| --- | --- |
| Name | Description |
| min-ss | Minimum signal strength filter. 0 = no filter. |
| XLRID | xBand long range read ID filter. 0 = no filter. |

#### Example

{

"XLRID": "2749D22AE6",

"min-ss": -80

}

## DELETE /filter

Removes any filters set via PUT /filter service.

## PUT /xband/beacon

Broadcast a command repeatedly. Use DELETE /xband/beacon to stop the command.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Name | Optional | Description |
| XLRID | Yes | Five byte xBand ID to transmit command to. If not specified, then the command is sent by default to the broadcast ID that all xBands listen to. Use this parameter to transmit to a single band instead. |
| cmd | No | The command to send. Needs to be six bytes formatted as a hex string. See example below. Consult the xBand RF ICD for the contents of these bytes. |

#### Examples

PUT http://4.3.2.1:8080/xband/beacon?cmd=12AB34CD009F

PUT http://4.3.2.1:8080/xband/beacon?XLRID=4900AB124E&cmd=101300FF90CD

The first example sends a command to all xBands and the second example sends to xBand with ID 4900AB124E.

## GET /xband/beacon

Respond with information about the current beacon command if any. The format of the response is the current xBand beacon information in JSON format.

#### Example (beacon enabled)

{

"enabled" : true,

"XLRID" : "4900AB124E",

"cmd" : "101300FF90CD"

}

#### Example (beacon disabled)

{

"enabled" : false

}

## DELETE /xband/beacon

Stop broadcasting command previously set up with PUT /xband/beacon

## PUT /xband/commands

Send one or more commands to one or more xBands. Unlike a broadcast command set up with PUT /xband/beacon, these commands are sent to individual xBands in response to xBand pings. The commands and parameters can be specified either via URL encoding or in a JSON payload.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Name | Optional | Description |
| XLRID | Yes | Give byte xBand ID to transmit command to. If not specified, the command is sent to any and all xBands that ping the xBR with a signal strength above the signal strength threshold. |
| ss | Yes | Signal strength threshold. The command is only sent to xBands that ping with a signal strength equal or greater than this value. If not specified, then any ping is sufficient to initiate the command. |
| cmd | No | The command to send. Needs to be six bytes formatted as a hex string. See example below. Consult the xBand RF ICD for the contents of these bytes. |

#### Example

PUT http://4.3.2.1:8080/xband/commands?cmd=12AB34CD009F

PUT http://4.3.2.1:8080/xband/commands?XLRID=4900AB124E&ss=-65&cmd=101300FF90CD

In the first example, the command is sent to any band that the xBR receives a ping from In the second example, the command is sent only to the band with ID 4900AB124E and only if it receives pings with a signal strength of -65dB or greater from that band.

#### Commands in JSON Payload

As an alternative to the URL encoded parameters, commands can be specified in a JSON payload. In this case, lists of XLRIDs can be provided and multiple commands can be specified.

The very simplest case would send a single command to any xBand that pings.

{

"cmd" : "101300FF90CD"

}

In this example, the command is sent to a specific xBand and only if it pings with a sufficient signal strength.

{

"XLRID" : "4900AB124E",

"ss" : -65,

"cmd" : "101300FF90CD"

}

To send a command to more than one xBand, you can provide a list of xBand IDs:

{

"XLRID" : [ "4900AB124E", "120032AB99", "120032AB9A" ],

"ss" : -65,

"cmd" : "101300FF90CD"

}

And you can also specify multiple commands to different xBands. In this case you must specify an array named "commands" like so:

{ "commands" :

[

{

"XLRID" : [ "4900AB124E", "120032AB99", "120032AB9A" ],

"ss" : -65,

"cmd" : "101300FF90CD"

},

{

"XLRID" : "5930AB124E",

"cmd" : "101300809001"

}

]

}

## GET /xband/commands

Respond with information about the current xBand commands if any. The format of the response is in JSON format.

#### Example (no commands)

{

"enabled" : true,

"reply\_all" : false,

"reply\_count" : 0

}

#### Example (with commands)

{

"enabled" : true,

"reply\_all" : false,

"reply\_count": 20

}

## DELETE /xband/commands

Clear all commands being sent to xBands.

## PUT /xband/options

Set options related to xBand command and control.

#### Parameters

|  |  |  |
| --- | --- | --- |
| *Name* | *Optional* | *Description* |
| reply\_timeout | yes | Sets the timeout value in seconds for bands to remain in the command list since set or the last received ping. |

#### Example

PUT http://4.3.2.1:8080/xband/options?reply\_timeout=1800

## GET /xband/options

Retrieve options related to xBand command and control. Output is in JSON format.

#### Example

{

"reply\_timeout" : 3600

}

## DELETE /xband/output

Stops all and any output to xBands. This is equivalent to sending both DELETE /xband/beacon and DELETE /xband/commands.

#### Example:

DELETE http://4.3.2.1/xband/commands

## PUT /xbrc

Set the xBRC URL. The URL value will persist on restart. This command will have no effect if the xBRC URL is configured either manually or through DHCP.

|  |  |  |
| --- | --- | --- |
| *Name* | *Optional* | *Description* |
| url | no | Full URL to the xBRC. |

#### Example:

PUT http://4.3.2.1/xbrc?url=http://8.7.6.5:8080

## DELETE /xbrc

Clear the current xBRC URL. This command will have no effect if the xBRC URL is configured either manually or through DHCP.

#### Example:

DELETE http://4.3.2.1/xbrc