

Energy Switch API for Approved Third Parties

Offering residential microgrid management services and Energy Switch product add-ons

Version 1.0
Pecan Street Inc.
Last Updated: 10/20/2016

Contents

Version History	2
Introduction	3
SARM Serial Communication.....	3
GetLoad	3
GetInfo.....	4
SetMode	5
SetSell	6
SARM API Communication	7
Utilizing the API	7
Cyber-Security	7
Fetching Data - GetLoad	7
Fetching Data - GetInfo	9
Setting Data - SetMode	10
Setting Data - SetSell	10

Version History

v1.0 – (2016-07-16) – Created document

v1.1 – (2016-10-20) – Included cybersecurity description

Introduction

This document is meant to serve as an overview document detailing how to interface with the SARM (define here) module. The SARM module included in the RAVI prototype units allows for control over an energy management system which can incorporate many different types of input power and load control.

Access to the SARM module can be accomplished via a reverse proxy system (external network access) or via local network access. The methods used and documented below are the same.

Included in the documentation below is also the low-level serial communication between the local gateway (BluCube) and the SARM, as well as the Application Programming Interface (API) which is exposed as a RESTful state service from the BluCube.

SARM Serial Communication

The SARM physical interface consists of a two-wire serial communication bus at 115200 baud. Stops are 8, N, 1. Communication between the BluCube and SARM module will need to be configured appropriately for adequate throughput.

Commands passed to the SARM module should be passed as text, followed by an ASCII carriage return.

There are four different commands which can be passed: GetLoad, GetInfo, SetMode, SetSell. These modes are defined in greater detail below.

GetLoad

About

The GetLoad serial command is passed to the SARM module when load information is wanted. The serial request sent is "GetLoad" followed by a newline break. By rule, this function should not be called more than once per minute by any serial-attached device, unless specifically tested and verified to not interfere with the operations of the SARM.

The GetLoad data package is returned in a comma-separated strings followed by an end of line character. Multiple lines can be returned from a GetLoad command. The contents of the comma-separated string are as follows:

Field 1 – Request Identification

Description: The request identification used in the gathering and logging of data. For the GetLoad request, a constant "GetLoad" will be returned.

Type: Char(7)

Field 2 – Circuit Identification

Description: The circuit identification of the circuit being monitored. This will return a string no longer than 255 characters (including null-terminator)

Type: Char(255)

Field 3 – Wattage

Description: The wattage of the circuit being monitored, measured to two significant digits.

Type: Double

Field 4 – Volts RMS

Description: The volts in RMS for the specific circuit being returned.

Type: Double

Field 5 – Current RMS

Description: The current measured using RMS for the specific circuit being returned.

Type: Double

Field 6 – Volt-Amperes

Description: The volt-amperes measured for the specific circuit being returned.

Type: Double

Field 7 – Total Power Factor

Description: The total power factor (real over apparent power) for the specific circuit being returned.

Type: Double

Sample Request and Return

Request: GetLoad

Return result: [note: </n> denotes a carriage return]

```
GetLoad,Load_1,50.11,120.12,2.01,2.29,0.88</n>
GetLoad,Load_12,100.11,121.12,66.01,70.29,0.98</n>
GetLoad,Load_13,20.11,122.12,5.01,2.29,0.87</n>
GetLoad,Load_14,40.11,120.12,3.01,2.29,0.85</n>
```

Request: GetLoad

Return result: [note: </n> denotes a carriage return]

```
GetLoad,Load_1,493.93,120.56,4.10,494.65,1.00
GetLoad,Load_2,492.68,121.19,4.07,493.58,1.00
GetLoad,Load_3,0.26,121.67,0.01,0.61,0.42
GetLoad,Load_4,0.34,118.87,0.02,2.46,0.14
GetLoad,Load_5,0.32,121.65,0.01,0.62,0.51
GetLoad,Solar,-0.34,119.41,0.01,0.79,-0.43
GetLoad,Grid,971.90,239.94,4.07,977.54,0.99
GetLoad,Use,980.17,241.51,4.06,981.82,1.00
GetLoad,InvGrid,13.50,238.93,0.11,26.54,0.51
GetLoad,InvLoad,-0.09,238.08,0.01,1.87,-0.05
```

GetInfo

About

The GetInfo serial command is passed to the SARM module when general device information is wanted. The serial request sent is “GetInfo” followed by a newline break. By rule, this function should not be called more than once per second by any serial-attached device, unless specifically tested and verified to not interfere with the operations of the SARM.

The GetInfo data package returns a single comma-separated string followed by an end of line character. The contents of the comma-separated string are as follows:

Field 1 – Request Identification

Description: The request identification used in the gathering and logging of data. For the GetInfo request, a constant “GetInfo” will be returned.

Type: Char(7)

Field 2 – Mode

Description: The current mode of the SARM module

Type: Unsigned int

Field 3 – Grid

Description: Total load or production at the grid connection of the residential structure.

Type: Signed int

Field 4 – Solar

Description: Legacy grid tie solar PV AC circuit production.

Type: Signed int

Field 5 – Load

Description: The total power used by the structure, in the absence of any solar, or battery generation this would be identical to Grid. However, with solar or battery generation grid may be negative (selling back to the utility) while the load is positive.

Type: Signed int

Field 6 – Battery Power

Description: Power flow into and out of the battery. Negative number is power flowing out of the battery being sold to the grid, positive is power flowing into the battery.

Type: Signed int

Field 7 – Battery State-of-Charge

Description: %of battery capacity storarage remaining for use. Ex: 75% means that 75% of the 9.6kWh of storage capacity remains in the battery.

Type: Signed int

Sample Request and Return

Request: GetInfo

Return result: GetInfo,5,965,0,998,0,95

Request: GetInfo

Return Result: GetInfo,2,12,0,994,-1113,95

SetMode

About

The SetMode serial command is passed to the SARM module when a change in mode is requested. The serial request sent is “SetMode” followed by an unsigned int. By rule, this function should not be called more than once per second by any serial-attached device, unless specifically tested and verified to not interfere with the operations of the SARM.

The SetMode data package has an acknowledgement returned if the mode change was successful. Successful modification to the SARM mode returns Boolean True. An unsuccessful mode change returns Boolean False. The return package consists of a comma-separated field with a static string of “SetMode” (char(7)), followed with the Boolean state of success.

Sample Request and Return

Request: SetMode 5 </n>

Return result: SetMode,True

Request: SetMode 9 </n>

Return result: SetMode,False

SetSell

About

The SetSell serial command is passed to the SARM module when a need to sell power to the grid is identified. The serial request sent is "SetSell" followed by an unsigned int. By rule, this function should not be called more than once per second by any serial-attached device, unless specifically tested and verified to not interfere with the operations of the SARM.

The SetSell data package has an acknowledgement returned if the sell request was successfully started. Successfully starting to sell power returns Boolean True. An unsuccessful attempt returns Boolean False. The return package consists of a comma-separated field with a static string of "SetSell" (char(7)), followed with the Boolean state of success.

Sample Request and Return

Request: SetSell 1000 </n>

Return result: SetSell,True

Request: SetSell 37000 </n>

Return result: SetSell,False

SARM API Communication

In addition to the low-level serial communication on-board the SARM, an API is also provided for ease of use and controllability remotely. The API functions are handled by the Pecan Street BluCube – a device programmed to interact on a serial-level with the SARM. Access to the BluCube API is provided via a RESTful state API. All queries should be accompanied by a key provided to you.

Per RFC 2616, acceptable responses include:

200 OK

400 Bad Request

401 Unauthorized

416 Requested Range Not Satisfiable

500 Internal Server Error

The URL used for local device access should be either the local IP, or the BluCube name followed by “.cube.blue” (e.g. – 10.0.0.1, or BC00099.cube.blue). Routing via the cube.blue proxy requires a working internet connection. The API URL can be found at [http://\[IP or Proxy\]/api/](http://[IP or Proxy]/api/). All queries to the API are processed as REST queries. If applicable, all response bodies are returned in JSON format.

Utilizing the API

In working with the BluCube API, two distinct methods are used – getting information and setting modes. Regardless of method, there is one mandatory element that must be passed in every API call: the provided key.

Cyber-Security

API commands are passed to the Energy Switch through an encrypted tunnel. This provides the highest level of security normally available for these types of applications. Un-authorized commands are therefore only possible if a hacker is able to obtain authorized credentials from the user. Since the potential for disruption is substantial if a successful hack is achieved, the Energy Switch has an additional line of defense to minimize the impact of a security breach. The Energy Switch will not respond to any command other than those specifically listed in the API library. None of the API commands are able to put the Energy Switch into a dangerous condition because internal logic overrides any request that would create unsafe voltages, currents or charging levels. When the API and control gateway are fully developed, overly frequent or erratic API calls will cause the Energy Switch to ignore API calls for a period of time. If API traffic is deemed to be erratic, the Energy Switch will default to cycling between on-grid and off-grid operation that will minimize wear and tear on both the energy switch and the distribution grid. This cycling will be benign and random so that a hacker cannot exploit a known response to achieve a grid overload condition. The system will send out alerts when hacking is detected and there is no logical path to prevent these alerts. With this architecture a successful hack is highly unlikely and, given the limited damage that can be done, there is little motivation to gain control of Energy Switch systems in the first place.

Fetching Data - GetLoad

When fetching load data from the BluCube API, please utilize the API URL/getload/ call. The getload call has three required fields, and one optional field that is available. The fields should be passed as key-value pairs directly within the URL.

When fetching data from a BluCube, there is a limit of 2000 records returned per call. As such, a narrow-band start_time and end_time should be used while accessing data.

REST Object	Required	Description
key	YES	The key must be provided with each API call to authenticate the user. This field can be made of up alpha-numeric characters.

start_time	YES	The start time, in Unix epoch UTC, that you would like to begin returning data. This is limited to numeric characters only.
end_time	YES	The end time, in Unix epoch UTC, that you would like to not receive data afterwards. Please note: a start_time is required if an end_time is provided. This is limited to numeric characters only.
circuit_type	NO	The specific circuit identification to be included in the result set. If no circuit type is passed, all circuits will be returned. If a passed circuit type is not found, no results will be returned.

Returned Data

The data returned from the API call is a JSON array containing individual meter reads. Each read has:

- the time it was read (recorded_time),
- the circuit identifier (circuit),
- the wattage of the circuit (watts),
- the volts RMS of the circuit (volts),
- the current RMS of the circuit (current),
- the volt-amperes of the circuit (va),
- the total power factor of the circuit (pf)

A sample JSON return is shown below.

```
[
  {
    "recorded_time": "2016-01-11 20:00:23.107351-00",
    "circuit": "Load_1",
    "watts": "6522.25",
    "volts": "244.02",
    "current": "19.52",
    "va": "20.01",
    "pf": "0.98"
  },
  {
    "recorded_time": "2016-01-11 20:00:23.107351-00",
    "circuit": "Load_2",
    "watts": "123.25",
    "volts": "120.02",
    "current": "10.52",
    "va": "12.01",
    "pf": "0.92"
  }
]
```

Examples

To return all data between Jan 1, 2015 and Jan 2, 2015:

http://bc00099.cube.blue/api/getload/?key=abcd1234&start_time=1420092000&end_time=1420178400

To return all data for a specific circuit type on a specific date:

http://bc00099.cube.blue/api/getload/?key=abcd1234&start_time=1420092000&end_time=1420178400&circuit_type=Load_1

Fetching Data - GetInfo

When fetching load data from the BluCube API, please utilize the API URL/`getinfo/` call. The `getinfo` call has three required fields, and one optional field that is available. The fields should be passed as key-value pairs directly within the URL.

When fetching data from a BluCube, there is a limit of 2000 records returned per call. As such, a narrow-band `start_time` and `end_time` should be used while accessing data.

REST Object	Required	Description
key	YES	The key must be provided with each API call to authenticate the user. This field can be made of up alpha-numeric characters.
start_time	YES	The start time, in Unix epoch UTC, that you would like to begin returning data. This is limited to numeric characters only.
end_time	YES	The end time, in Unix epoch UTC, that you would like to not receive data afterwards. Please note: a <code>start_time</code> is required if an <code>end_time</code> is provided. This is limited to numeric characters only.
field_type	NO	The specific field identification to be included in the result set. If no field type is passed, all fields will be returned. If a passed field type is not found, no results will be returned.

Returned Data

The data returned from the API call is a JSON array containing individual meter reads. Each read has:

- the time it was read (`recorded_time`),
- the mode (`mode`),
- the grid status (`grid`),
- the solar status (`solar`),
- the load results (`load`),
- the battery power values (`battery`),
- the battery state of charge (`soc`)

A sample JSON return is shown below.

```
[
  {
    "recorded_time": "2016-01-11 20:00:23.107351-00",
    "mode": "5",
    "grid": "1235",
    "solar": "4262",
    "load": "1952",
    "battery": "2001",
    "soc": "80"
  },
  {
    "recorded_time": "2016-01-11 20:00:24.107351-00",
    "mode": "5",
    "grid": "1235",
    "solar": "4262",
    "load": "1952",
```

```
        "battery": "2001",
        "soc": "80"
    }, ...
]
```

Examples

To return all data between Jan 1, 2015 and Jan 2, 2015:

http://bc00099.cube.blue/api/getinfo/?key=abcd1234&start_time=1420092000&end_time=1420178400

To return all data for a specific circuit type on a specific date:

http://bc00099.cube.blue/api/getinfo/?key=abcd1234&start_time=1420092000&end_time=1420178400&circuit_type=Load_1

Setting Data - SetMode

When setting the mode status using the BluCube API, please utilize the API URL/setmode/ call. The setmode call has two required fields. The fields should be passed as key-value pairs directly within the URL.

REST Object	Required	Description
key	YES	The key must be provided with each API call to authenticate the user. This field can be made of up alpha-numeric characters.
mode	YES	The mode you would like to change to.

Returned Data

The data returned from the API call is a JSON array containing the results of the attempt to change the mode.

A sample JSON return is shown below.

```
[{"success": "True"}]
```

Examples

To set the mode to mode 5:

<http://bc00099.cube.blue/api/setmode/?key=abcd1234&mode=5>

Setting Data - SetSell

When setting the sell status using the BluCube API, please utilize the API URL/setsell/ call. The setsell call has two required fields. The fields should be passed as key-value pairs directly within the URL.

REST Object	Required	Description
key	YES	The key must be provided with each API call to authenticate the user. This field can be made of up alpha-numeric characters.
sell	YES	The amount of power (in watts) to sell.

Returned Data

The data returned from the API call is a JSON array containing the results of the attempt to switch into sell state.

A sample JSON return is shown below.

```
[{"success": "True"}]
```

Examples

To set the mode to mode 5:

<http://bc00099.cube.blue/api/setsell/?key=abcd1234&sell=5>