

AtHomePowerlineServer

Copyright © 2014, 2015 Dave Hocker (AtHomeX10@gmail.com)

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, version 3 of the License. See the LICENSE file for more details.

Contents

1 Overview.....	2
2 License.....	3
3 Source Code.....	3
4 Installation.....	3
4.1 Basic Steps.....	3
4.2 Files and Their Location.....	3
4.2.1 Unix Type Systems.....	4
4.2.2 Windows.....	4
4.3 Configuration File.....	4
4.4 Running the Server.....	4
4.4.1 Raspbian Wheezy.....	4
4.4.1.1 As an Application.....	4
4.4.1.2 As a Daemon.....	4
4.4.2 Windows.....	5
5 TCP/IP Protocol.....	5
5.1 Standard Response Content.....	5
5.2 Requests.....	6
5.2.1 StatusRequest.....	6
5.2.1.1 Request.....	7
5.2.1.2 Response.....	7
5.2.2 LoadTimers.....	7
5.2.2.1 Request.....	7
5.2.2.2 Response.....	9
5.2.3 LoadActions.....	9
5.2.3.1 Request.....	9
5.2.3.2 Response.....	10
5.2.4 On.....	10
5.2.4.1 Request.....	10
5.2.4.2 Response.....	10
5.2.5 Off.....	10
5.2.5.1 Request.....	11
5.2.5.2 Response.....	11
5.2.6 Dim.....	11
5.2.6.1 Request.....	11
5.2.6.2 Response.....	11
5.2.7 Bright.....	12
5.2.7.1 Request.....	12
5.2.7.2 Response.....	12
5.2.8 All Units Off.....	12
5.2.8.1 Request.....	12
5.2.8.2 Response.....	12
5.2.9 All Lights Off.....	12
5.2.9.1 Request.....	13
5.2.9.2 Response.....	13
5.2.10 All Lights On.....	13
5.2.10.1 Request.....	13

5.2.10.2 Response.....	13
5.2.11 Get Time.....	13
5.2.11.1 Request.....	13
5.2.11.2 Response.....	13
5.2.12 Get Time.....	13
5.2.12.1 Request.....	14
5.2.12.2 Response.....	14
5.2.13 Set Time.....	14
5.2.13.1 Request.....	14
5.2.13.2 Response.....	14
5.2.14 Get Sun Data.....	14
5.2.14.1 Request.....	14
5.2.14.2 Response.....	14
5.3 Configuration File.....	15
6 Client Examples.....	15
6.1 AtHomePowerlineServer SDK.....	15
6.2 Test Client.....	16
6.3 AtHomeX10.....	16
6.4 AtHomeWeb.....	16
7 References.....	16
8 Appendix.....	16
8.1 Raspberry Pi.....	16

1 Overview

In the world of X10, most of the power line controllers like the CM11/CM11A have been around for years. The majority of the sophisticated X10 management applications run on a PC or Mac and more or less use the power line controller as a relatively “dumb” controller. There are a number of problems with the current configurations that the AtHomePowerlineServer aims to address.

First, the AtHomePowerline server is designed to run on a light weight system (e.g. a Raspberry Pi) that can be run head-less and fan-less. This is as opposed to a PC or Mac based solution. The small size of such a system like the Raspberry Pi allows it to be positioned more freely (ideally, close to the breaker panel). And, a Raspberry Pi system can be assembled for considerably less than a PC. While the server was designed to run on a lightweight system, it will run on any system that supports Python 2.7 (including Windows).

Second, the communication mechanism to the AtHomePowerlineServer is TCP/IP (over Ethernet). A light weight system like the Raspberry Pi supports many WiFi USB interfaces. This further improves the freedom of location for the X10 controller. In the past, the distance between a PC and the X10 controller was gated by the need for physical wiring and RS-232 limitations. With the AtHomePowerlineServer architecture, this limitation is effectively eliminated. See Illustration 1.

Finally, the AtHomePowerlineServer application is open source. Anyone can fork it and build upon it.

The remainder of this document describes:

- Installation
- Configuration
- Programming interface

While the functionality of the AtHomePowerlineServer is designed with knowledge of the CM11/CM11A architecture, it does not exactly match that architecture. The AtHomePowerlineServer is an abstraction of the capabilities of the CM11/CM11A. For example, the AtHomePowerlineServer features timers and actions that are conceptually similar to the CM11/CM11A timer initiators and macros, but they are not exactly alike.

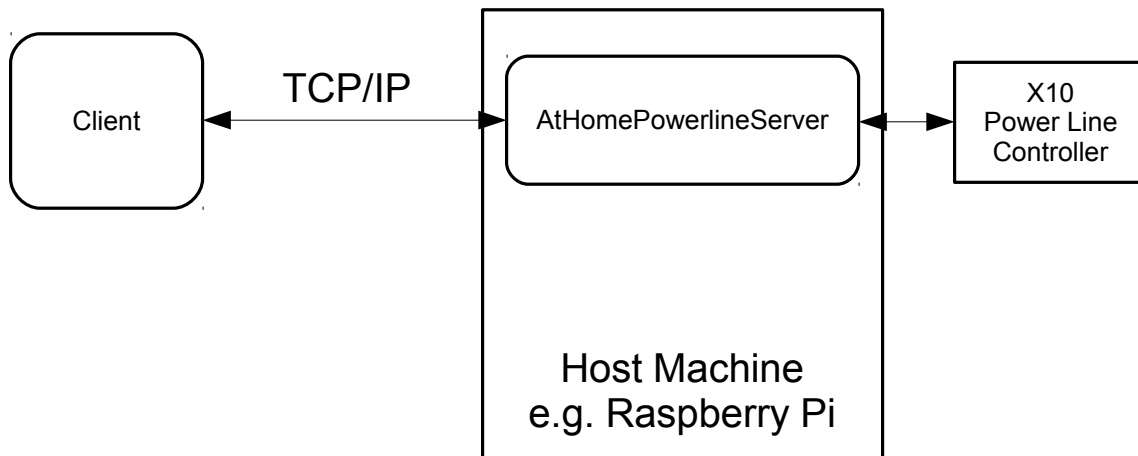


Illustration 1: AtHomePowerlineServer Architecture

2 License

The server is licensed under the GNU General Public License v3 as published by the Free Software Foundation, Inc.. See the LICENSE file for the full text of the license.

3 Source Code

The server is written completely in Python 2.7 (latest maintenance release). The source code can be found on GitHub at: TBD

If you plan on modifying the source consider using virtualenv and virtualenvwrapper. There is a requirements.txt file that can be used with pip to create the required virtual environment.

4 Installation

4.1 Basic Steps

1. Open a terminal window (Linux) or a command prompt (Windows)
2. Clone the repository from GitHub. Under Raspbian Wheezy the recommended location for cloning is `/home/pi/rpi/elfstone_athomeserver`.
3. Change into the main repo directory.
4. Copy the file `sample AtHomePowerlineServer.example.conf` to `AtHomePowerlineServer.conf`.
5. Edit the `AtHomePowerlineServer.conf` configuration file as needed. Generally you will need to edit the `ComPort` parameter.
6. Move the `AtHomePowerlineServer.conf` file to the location identified by section 4.2. Use `sudo` to move/copy the file so that it has root ownership.
7. Move or copy the `AtHomePowerlineServer.sqlite3` to the location identified by section 4.2. Be sure to use `sudo` so that the copied/moved file will belong to root. Otherwise, you will have to “chown” the copied file to root ownership.

4.2 Files and Their Location

There are two key files:

- AtHomePowerlineServer.conf – the configuration file
- AtHomePowerlineServer.sqlite3 – the database file

These files are kept in OS dependent locations. AtHomePowerlineServer automatically creates the database file (if it does not exist), so there is little to worry about there.

4.2.1 Linux Based Systems

On Linux based systems like Raspbian Wheezy, the key files are kept in traditional locations.

File	File Path
Configuration	<home>/AtHomePowerlineServer.conf or /etc/AtHomePowerlineServer.conf
Database	<DatabasePath>/AtHomePowerlineServer.sqlite3
Logfile	<LogFile> entry in AtHomePowerlineServer.conf

<home> is the path to AtHomePowerlineServer.py file.

<DatabasePath> is the path specified by the AtHomePowerlineServer.conf DatabasePath entry.

<LogFile> is the path and file name specified by the AtHomePowerlineServer.conf LogFile entry.

Note that a configuration file in <home> takes precedent over one in /etc.

4.2.2 Windows

On Windows based systems the configuration file and the database file are kept in the following locations.

File	File Path
Configuration	Home directory of the AtHomePowerlineServer application. This will be the directory where AtHomePowerlineSever.py resides.
Database	c:\users\username\AppData\Local\AtHomePowerlineServer\AtHomePowerlineServer.sqlite3

4.3 Configuration File

TBD

Parameter	Description
X10ControllerDevice	
ComPort	
Port	
LogFile	

LogConsole	
LogLevel	
DatabasePath	
City	The city where the installation is located. City, latitude and longitude are used for accurately calculating the sunset and sunrise (using the astral package).
Latitude	The latitude of installation.
Longitude	The longitude of the installation.

4.4 Running the Server

4.4.1 Raspbian Wheezy

4.4.1.1 As an Application

Start the server: `sudo python AtHomePowerlineServer.py`

4.4.1.2 As a Daemon

This technique will set up AtHomePowerlineServer so that it automatically starts when Raspbian boots up.

1. Change the AtHomePowerlineServerD.sh script according to where you have cloned the source code (approximately line 14). If you cloned the source code into the recommended location no changes should be required.
2. Copy the edited AtHomePowerlineServerD.sh file to /etc/init.d
3. Run: `sudo update-rc.d AtHomePowerlineServerD.sh defaults`
4. Reboot or run: `sudo /etc/init.d/AtHomePowerlineServerD.sh start`

4.4.2 Windows

AtHomePowerlineServer can be run as an ordinary Python application. Change into the home directory and run: `python AtHomePowerlineServer.py`

5 DatabasePath

TBD

6 TCP/IP Protocol

A client communicates with the server by sending a JSON formatted request and the server returning a JSON formatted response. For each request, the client opens a TCP/IP socket to the server, sends the request, receives the response and closes the socket.

Client	Server
Open socket to server. The default port is 9999. This can be changed through the configuration file.	

	Accept call from client. The server will accept a call on any host network interface.
Send JSON formatted request to server	
	Receive request from client
	Execute request
	Return JSON formatted response to client
Receive response from server	
Close socket to server	Close socket to client

6.1 Standard Response Content

Each request generates a return response. The content of a response is partially standard and partially request dependent. A request can add any number of key/value pairs to the standard content. The standard content is shown here.

JSON Format with standard content:

```
{
  "X10Response":
  {
    "request": "LoadTimers",
    "server": "AtHomePowerlineServer",
    "server-version": "1.0.0.0",
    "result-code": 0,
    "date-time": "2014-01-29 11:04:06.093000",
    "error": "Error description",
    "message": "Message text",
    "call-sequence": 1
  }
}
```

Key	Value(s)	Description
request	LoadTimers	Identifies the request for which the response is given.
server	AtHomePowerlineServer	The name of the server yielding the response.
server-version	1.0.0.0	The version of the responding server.
result-code	0 1 2 3 4 5 6	The command was successfully executed. Time out waiting for checksum from controller. Time out waiting for interface ready from controller. Ack was not received from controller. COM port is not available. An exception occurred. Checksum error.
date-time	Local time, ISO formatted	The server time when the request was executed.
error	Text	Human readable text describing error condition. This

		parameter will only be present if the resultcode > 0.
message	Text	Human readable text providing extra details on the request. Consider this to be non-error text.
call-sequence	A sequential number	An ever increasing number that identifies the order of the request. The sequence number is reset on every server start. Useful for client side logging.

Programmer Note: Typically, you are looking for a resultcode == 0. That means your request was successful. Any non-zero resultcode means your request failed for some reason. You will likely need to look at the server console or the server log to determine why your request failed.

6.2 Requests

- StatusRequest
- LoadTimers
- LoadActions
- On
- Off
- Dim
- Bright
- AllUnitsOff
- AllLightsOff
- GetTime
- SetTime

6.2.1 StatusRequest

The StatusRequest is useful as a server handshake.

6.2.1.1 Request

```
{
  "request": "StatusRequest",
  "args": {}
}
```

6.2.1.2 Response

The StatusRequest request returns a standard response as defined in section 6.1.

6.2.2 LoadTimers

6.2.2.1 Request

The LoadTimers request sends a complete set of timer programs to the server (replacing the entire current set of timer programs). A timer program is reasonably analogous to a CM11 timer initiator. Each timer program specifies a start and stop event along with a day mask. These parameters identify when a start or stop event occurs. The timer program includes a house/device code that designates the target of the start/stop events. Finally, the timer program names a start and stop action that is to be executed when the corresponding event occurs.

AtHomePowerlineServer offers different methods for triggering a start or stop event. The interesting case is the sunset/sunrise based case. You can specify an event to occur relative to sunrise or sunset. For example, you can create an event where the start time occurs 10 minutes before sunset. This would be useful for turning on a light at dusk. Or, you might set a stop event to occur a few minutes after sunrise. This might be useful for turning off outside lights in the morning. The obvious value of sunset/sunrise based events is that you do not have to change them to accommodate the time of year.

As an aside, there are 10 years worth of sunset/sunrise times in the `sun_table`. These times are for Atlanta, GA. The times can be replaced with data for another location. A good place to get the data is:

http://aa.usno.navy.mil/data/docs/RS_OneYear.php. If you want to change the sun data, look for the file `database/sunrise_sunset.sql`. This file is automatically run during the create database process.

It is possible to obtain the sunrise/sunset times on demand by always going to the Navy site (and the source code repo contains code that can do this). However, this technique requires a constant Internet connection. For this reason, the static data in the `sun-table` was used.

A timer program may also include a randomization factor for start and stop triggers. This feature can be used to adjust trigger events for a more “human” behavior.

Timer programs are persisted in a Sqlite3 database in the Timers table. They are loaded at server start up.

JSON Format

```
{
  "request": "LoadTimers",
  "args": {
    "programs": [
      {
        "name": "program-name",
        "house-device-code": "a1",
        "start-trigger-method": "clock-time",
        "stop-trigger-method": "clock-time",
        "start-time": "15:30",
        "start-time-offset": "10",
        "start-randomize": "1",
        "start-randomize-amount": "10",
        "stop-time": "23:30",
        "stop-time-offset": "-10",
        "stop-randomize": "1",
        "stop-randomize-amount": "10",
        "day-mask": "mtwtfss",
        "start-action": "action-name",
        "stop-action": "action-name"
      },
      {
        ...
      }
    ]
  }
}
```



```

    }
  ]
}

```

args Key	Value(s)	Description
programs	[]	The array of timer programs.
name	LightOn	Human readable name for the program. The name must be unique. Currently, not used by the server.
house-device-code	a1...a16 through p1...p16 or a through p	For X10 modules, the house code concatenated with the device code. Case insensitive. For house “modules”, the house code. Case insensitive.
start-trigger-method	none clock-time sunrise sunset	Event is not used. Event triggered on local clock time. Event triggered relative to sunrise. Event triggered relative to sunset.
stop-trigger-method	none clock-time sunrise sunset	Event is not used. Event triggered on local clock time. Event triggered relative to sunrise. Event triggered relative to sunset.
start-time	HH:MM	Using a 24 hour clock, the time when the start event triggers. For start trigger method clock-time, only.
start-time-offset	+/-n (minutes)	Offset from sunrise/sunset for the start event.
start-randomize	0 or 1	0=false, 1=true. If true, randomize the start trigger time using the start-randomize-amount value.
start-randomize-amount	n (minutes)	If start-randomize is true, the effective start trigger time will be adjusted by a randomized value r (in minutes) where -amount <= r <= amount.
stop-time	HH:MM	Using a 24 hour clock, the time when the stop event triggers. For stop trigger method clock-time, only.
stop-time-offset	+/- n (minutes)	Offset from sunrise/sunset for the stop event.
stop-randomize	0 or 1	0=false, 1=true. If true, randomize the stop trigger time using the stop-randomize-amount value.
stop-randomize-amount	n (minutes)	If stop-randomize is true, the effective stop trigger time will be adjusted by a randomized value r (in minutes) where -amount <= r <= amount.
day-mask	mtwtfss (whole week) mtwtf-- (week days) -----ss (weekend days)	The day(s) of the week when the program is effective. Starts with Monday and ends with Sunday. Days with letters are effective. Days with a dash (-) or a period (.) are not effective.
start-action	on “”	The name of the action to be taken when the start event occurs. A value of “” causes no action to be taken. If the named action is not found in the Actions table, no action is taken.
stop-action	off “”	The name of the action to be taken when the stop event occurs. A value of “” causes no action to be taken. If the

		named action is not found in the Actions table, no action is taken.
--	--	---

6.2.2.2 Response

The LoadTimers request returns a standard response as defined in section 6.1.

6.2.3 LoadActions

The LoadActions request establishes the set of timer program actions that are available to timer programs.

6.2.3.1 Request

Actions are persisted in a Sqlite3 database in the Actions table. They are reloaded at server start up.

JSON Format

```
{
  "request": "LoadActions",
  "args":
    {
      "actions":
        [
          {
            "name": "action-name",
            "command": "x10-command",
            "dim-amount": "nn"
          },
          {
            ...
          }
        ]
    }
}
```

args Key	Value(s)	Description
actions	[]	The array of actions to be loaded.
name	Unique action name	The action name must be unique among all other action names. A timer program specifies an action name that is to be executed when the timer event occurs.
command	on off dim brighten or bright allunitsoff alllightsoff alllightson	The X10 command to be executed as the action. Valid for X10 module types (e.g. appliance or lamp) Valid for house "module" type. These commands use a house code as opposed to a house/device code.
dim-amount	nn	For commands that utilize a dim amount. Expressed as a percent. Must be a value in the range 0-100. The power line controller driver will convert this value into a something more meaningful to the actual power line controller.

6.2.3.2 Response

The LoadActions request returns a standard response as defined in section 6.1.

6.2.4 On

The On request is an immediate request that specifies a device to be turned on.

6.2.4.1 Request

```
{
  "request": "On",
  "args": {
    "house-device-code": "A1",
    "dim-amount": "nn"
  }
}
```

args Key	Value(s)	Description
house-device-code	a1...p16	The house code concatenated with the device code. Case insensitive.
dim-amount	nn	Dim amount expressed as a percent. Must be a value in the range 0-100. The power line controller driver will convert this value into a something more meaningful to the actual power line controller.

6.2.4.2 Response

The On request returns a standard response as defined in section 6.1.

6.2.5 Off

The Off request is an immediate request that specifies a device to be turned off.

6.2.5.1 Request

```
{
  "request": "Off",
  "args": {
    "house-device-code": "A1",
    "dim-amount": "nn"
  }
}
```

args Key	Value(s)	Description
house-device-code	a1...p16	The house code concatenated with the device code. Case insensitive.
dim-amount	nn	Dim amount expressed as a percent. Must be a value in the range 0-100. The power line controller driver will convert

		this value into a something more meaningful to the actual power line controller.
--	--	--

6.2.5.2 Response

The Off request returns a standard response as defined in section 6.1.

6.2.6 Dim

The Dim request is an immediate request that specifies a device to be dimmed by a given percent.

6.2.6.1 Request

```
{
  "request": "Dim",
  "args":
    {
      "house-device-code": "A1",
      "dim-amount": "nn"
    }
}
```

args Key	Value(s)	Description
house-device-code	a1...p16	The house code concatenated with the device code. Case insensitive.
dim-amount	nn	Dim amount expressed as a percent. Must be a value in the range 0-100. The power line controller driver will convert this value into a something more meaningful to the actual power line controller.

6.2.6.2 Response

The Dim request returns a standard response as defined in section 6.1.

6.2.7 Bright

The Dim request is an immediate request that specifies a device to be brightened by a given percent.

6.2.7.1 Request

```
{
  "request": "Bright",
  "args":
    {
      "house-device-code": "A1",
      "bright-amount": "nn"
    }
}
```

args Key	Value(s)	Description
house-device-code	a1...p16	The house code concatenated with the device code. Case

		insensitive.
bright-amount	nn	Bright amount expressed as a percent. Must be a value in the range 0-100. The power line controller driver will convert this value into a something more meaningful to the actual power line controller.

6.2.7.2 Response

The Bright request returns a standard response as defined in section 6.1.

6.2.8 All Units Off

The all units off request is an immediate request that turns off all devices in a given house code.

6.2.8.1 Request

```
{
  "request": "AllUnitsOff",
  "args": {
    "house-code": "A",
  }
}
```

args Key	Value(s)	Description
house-code	a...p	The house code whose units are to be turned off. Case insensitive.

6.2.8.2 Response

The All Units Off request returns a standard response as defined in section 6.1.

6.2.9 All Lights Off

The all lights off request is an immediate request that turns off all lights for a given house code.

6.2.9.1 Request

```
{
  "request": "AllLightsOff",
  "args": {
    "house-code": "A",
  }
}
```

6.2.9.2 Response

The All Lights Off request returns a standard response as defined in section 6.1.

6.2.10 All Lights On

The all lights on request is an immediate request that turns on all lights for a given house code.

6.2.10.1 Request

```
{
  "request": "AllLightsOn",
  "args": {
    "house-code": "A",
  }
}
```

6.2.10.2 Response

The All Lights Off request returns a standard response as defined in section 6.1.

6.2.11 Get Time

The GetTime request returns the current time on the machine where AtHomePowerlineServer is running.

6.2.11.1 Request

Not implemented.

6.2.11.2 Response

The GetTime request returns a standard response as defined in section 6.1. The date-time value is the current time on the AtHomePowerlineServer host.

6.2.12 Get Time

The GetTime request returns the current time on the machine where AtHomePowerlineServer is running.

6.2.12.1 Request

Not implemented.

6.2.12.2 Response

The GetTime request returns a standard response as defined in section 6.1. The date-time value is the current time on the AtHomePowerlineServer host.

6.2.13 Set Time

Not implemented.

6.2.13.1 Request

6.2.13.2 Response

The SetTime request returns a standard response as defined in section 6.1.

6.2.14 Get Sun Data

The Get Sun Data request returns the sunset and sunrise times for a given date.

6.2.14.1 Request

```
{
  "request": "GetSunData",
  "args": {
    "date": "2014-10-02"
  }
}
```

args Key	Value(s)	Description
date	ISO formatted date	The date for which sunset/sunrise times are desired.

6.2.14.2 Response

The GetSunData request returns a standard response augmented with the sunset/sunrise times.

```
{
  "X10Response": {
    "request": "LoadTimers",
    "server": "AtHomePowerlineServer",
    "server-version": "1.0.0.0",
    "result-code": 0,
    "date-time": "2014-01-29 11:04:06.093000",
    "error": "Error description",
    "message": "Message text",
    "call-sequence": 1,
    "data": {
      "sunset": "yyyy-mm-ddTHH:mm:ss",
      "sunrise": "yyyy-mm-ddTHH:mm:ss"
    }
  }
}
```

6.3 Configuration File

The server is controlled by a small text configuration file: **AtHomePowerlineServer.conf**. The contents of the configuration file are JSON formatted.

JSON Format

```
{
```

```

"Configuration":
{
  "X10ControllerDevice": "XTB232",
  "ComPort": "/dev/ttyAMA0",
  "Port": 9999,
  "LogFile": ""
}
}

```

Parameter	Value(s)	Description
X10ControllerDevice	XTB232 XTB-232 CM11 CM11A	Identifies the X10 controller to be supported.
ComPort	COM1 /dev/ttyAMA0 /dev/ttyUSB0	A COM port on Windows. This can be a regular serial port or it can be a USB-to-Serial adapter. Typical UART port on Raspberry Pi. A USB-to-Serial adapter on Raspberry Pi.
Port	9999	TCP/IP port number where server will listen for incoming client calls
Logfile	Full path and file name to be used for logging	If there is no log file value, all logging will go to the console. On a Raspberry Pi, you might want to log to a console or you might want to log to a file that is in RAM (to avoid constant writing to the SDCard).

Table 1: Configuration File Parameters

7 Client Examples

7.1 AtHomePowerlineServer SDK

The SDK facilitates writing Python scripts to control X10 devices. The SDK can be found on GitHub at TBD (elfstone_ahps_sdk).

The SDK includes a sample test program that illustrates how to use most of the SDK function. Look for elfstone_ahps_sdk/ahps_test.py in the repo.

7.2 Test Client

The test client application in elfstone_athomeserver/testclient/ahps_client.py provides several examples of sending commands to the server and receiving command responses.

7.3 AtHomeX10

AtHomeX10 is an open source Windows application that works with AtHomePowerlineServer. It provides a very basic UI and supports the most basic home management functions. It can be found on GitHub at TBD.

7.4 AtHomeWeb

AtHomeWeb will be an open source browser based application that works with AtHomePowerlineServer. Currently, it is a work-in-progress.

8 References

1. CM11A Protocol - http://jvde.us/info/CM11A_protocol.txt
2. XTB-232 - http://jvde.us/xtb/XTB-232_description.htm
3. Raspberry Pi - <http://www.raspberrypi.org/>

9 Appendix

9.1 Raspberry Pi

Home: www.raspberrypi.org

If you are unfamiliar with the Raspberry Pi, here is a list of parts you will need to get started. Good sources of these parts are:

Newark/Element14: www.newark.com

adafruit: www.adafruit.com

Sparkfun: www.sparkfun.com

Here is a reasonable bill of materials.

Raspberry Pi Model B

1 amp USB power supply with micro USB cable

HDMI to VGA converter, if you do not have a monitor with HDMI input

8 GB Class 10 SDCard

USB WiFi interface (check here for a list of WiFi interfaces that have been tested with the RPi: [USB WiFi Adapters](#))

RS-232 level converter ([RS232 Shifter SMD](#))

Powered USB hub. You might need this depending on your choice of WiFi interface and your need to attached a keyboard and mouse. The RPi USB ports (2) can source a limited amount of power to attached USB devices.

Case (optional)

The best way to get started is to go to the [RPi home site](#), download the Raspbian Wheezy operating system and follow the installation instructions.