AtHomePowerlineServer Interface

Contents

1 Overview

This document describes the programming interface to the AtHomePowerline Server. The server was designed to run on a lightweight system like a Raspberry Pi; but, it will run on any system that supports Python 2.7 (including Windows). Essentially, it provides network access to a CM11 or XTB-232 X10 power line controller. It also gives the XTB-232 extra capability (namely timer programs).

TBD: Need a diagram here.

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. The source code can be found on GitHub at: TBD

4 Installation

1. Open a terminal window (Linux) or a command prompt (Windows)
2. Clone the repository from GitHub.
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.
6. Start the server: python AtHomePowerlineServer.py

5 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	
	Accept socket from client
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

5.1 Response

JSON Format

```
{
   "X10Response":
      {
         "command": "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
command	LoadTimers	Identifies the request/command 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	The command was successfully executed. Time out waiting for checksum from controller. Time out waiting for interface ready from controller.

	3 4 5 6	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 successfully. 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.

5.2 Requests

5.2.1 LoadTimers

5.2.1.1 Request

The LoadTimers command (aka request) sends a complete set of timer programs to the server (replacing the entire current set of time programs). A timer program is reasonably analogous to a CM11 timer initiator. Each timer program specifies a start and end time along with a day mask. These parameters identify when an 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.

In a typical case, a timer program might identify when to turn a light on and when to turn it off.

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

JSON Format

```
{
   “command”: “LoadTimers”,
   “args”:
         {
            “programs”:
               [
                  {
                     “name”: “program-name”,
                     “house-device-code”: “a1”,
                     “start-time”: “15:30”,
                     “stop-time”: “23:30”,
                     “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. Not actually used by the server.
house-device-code	a1...a16 through p1...p16	The house code concatenated with the device code. Case insensitive.
start-time	HH:MM	Using a 24 hour clock, the time when the event starts.
stop-time	HH:MM	Using a 24 hour clock, the time when the event stops.
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	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	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.

5.2.1.2 Response

5.2.2 LoadActions

5.2.2.1 Request

5.2.2.2 Response

6 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
  }
}
```

Parameter	Value(s)	Description
X10ControllerDevice	XTB232 XTB-232 CM11 CM11A	Identifies the X10 controller to be supported.
ComPort	COM1 /dev/ttyAMA0	A COM port on Windows. Typical UART port on Raspberry Pi
Port	9999	TCP/IP port number where server will listen for incoming client calls

Table 1: Configuration File Parameters

7 Client Examples