

# lome6

## (Lights-Out-Management-Ether6)

by warhog <[warhog@gmx.de](mailto:warhog@gmx.de)> 2011

### Inhaltsverzeichnis

1 GPL.....	2
2 Einleitung.....	3
3 Funktionen.....	3
4 Erster Prototyp .....	4
5 Hardware.....	4
6 Software.....	5
6.1 Ethersex.....	5
6.2 Externer Client.....	5
6.3 Webseite.....	5
7 ECMD.....	5

*Lome6 (Lights-Out-Management-Ether6)*

## **1 GPL**

*Copyright (c) 2011 by warhog <warhog@gmx.de>*

*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; either version 3 of the License, or (at your option) any later version.*

*This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.*

*You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.*

*For more information on the GPL, please go to:  
<http://www.gnu.org/copyleft/gpl.html>*

## 2 Einleitung

Da meine Server (Backup und Web+Fileserver, beide Linux) im Keller stehen und ab und an (sei es durch Spieltrieb oder Stromausfall etc...) diverse Probleme haben, habe ich beschlossen ein einfaches Lights Out Management (LOM) System auf Basis des Ethersex Projektes zu realisieren.

## 3 Funktionen

### Anforderungen an den ersten Prototyp:

- Reset und Power Taster vom Gehäuse über das LOM durchschleifen (Reset, Power kurz und Power lang drücken)
- Serielle Schnittstelle per yPort weiterleiten
- Temperatursensoren für Lufteinlass, RAM und Netzteilauslass per OneWire (DS18x20)
- LCD Display vorne am Server (angesteuert durch TTY Layer)
- Status LEDs vorne am Server
- Externe Temperatur für CPU und SB (Southbridge) vom Mainboard per kleinem Daemon (siehe Kapitel 6.3)
- WHM (Uptime) des Moduls
- Anzeige Serverstatus (An, Aus) per Optokoppler von der PowerLED abgegriffen
- Power on delay (Server mit einstellbarer Verzögerung nach Stromausfall einschalten)

### TODO:

- LOM disable Schalter um das LOM temporär zu deaktivieren
- Serverdaemon der ein Heartbeat Signal ans LOM schickt, fehlt dieses für ~10sek -> Server hart resetten
- SNMP Support
- IPv6 Support (mangels Testnetz und eigenem Wissen aktuell nicht möglich)

## **4 Erster Prototyp**

Mittlerweile läuft der erste Prototyp stabil, der Schaltplan findet sich unter `services/lome6/doc`. Eine erste Prototypenplatine findet sich dort auch.

Ein Problem im Betrieb trat mit dem yPort Protokoll auf. Regelmäßig verschluckte dieses Zeichen oder hängte sogar das Ethersex auf.

Besonders ärgerlich wenn man dem Server beim booten zugeschaut hat. Das Problem lässt sich beheben wenn man die Größe des Puffers von yPort auf 512 byte oder mehr erhöht.

## **5 Hardware**

Die Hardware ist an das Netio von Pollin angelehnt.

Als CPU habe ich mich für einen atmega1284 entschieden, dieser hat genug RAM und Flashspeicher für zukünftige Entwicklungen. Ein atmega644 reicht aber auch, wenn man HTTP und LCD deaktiviert reicht theoretisch sogar ein atmega32 (nicht getestet).

Unter `/pinning/hardware/` findet sich die `lome6.m4`. Dort sind alle Portdefinitionen eingetragen.

Die Stromversorgung ist über die Standby Versorgung der internen USB Header realisiert. Den aktuellen Power Status (Server ein/aus) findet das lome6 über einen Optokoppler von der PowerLED. Die Taster (Reset + Power) vom Gehäuse zum Mainboard werden über das LOM geschleift und dort per Optokoppler vom LOM getriggert. Hier ist auf die Polung zu achten!

Die serielle Schnittstelle wird über einen MAX232 realisiert.

## 6 Software

### 6.1 Ethersex

Eingestellt werden die Daten des lome6 Moduls im Menü unter "Applications/lome6 Service".

Dort kann der LCD und OneWire Support und die Zeiten für das Triggern der Taster eingestellt werden. Zur Sicherheit sollte noch eins der PAM Module aktiviert werden!

### 6.2 Externer Client

Das lome6 kann über den externen Client gesteuert werden (jlome6gui).

Dieser ist in services/lome6/jlome6gui zu finden.

### 6.3 Daemon

Um die Temperature vom Server zum lome6 zu bekommen gibt es einen kleinen Daemon.

Dieser liegt in services/lome6/jlome6d.

Auf dem Server wird das lm-sensors Paket benötigt um die Temperatursensoren auszulesen.

Im Verzeichnis der .jar Datei muss eine Konfigurationsdatei namens „config.xml“ liegen.

*Dateiinhalt config.xml:*

```
<?xml version="1.0" ?>
<configuration>
  <ip address="192.168.144.251" />
  <port number="2701" />
  <auth username="admin" password="kkw1822" />
  <sensors cpu="Core0 Temp:" sb="temp1:" />
</configuration>
```

ip address	IP Adresse des lome6
port	ecmd Port des lome6
auth	PAM Username und Passwort
sensors	Beginn der Zeile der Ausgabe von sensors des gewünschten Sensors

## **6.4 Webseite**

Die andere Möglichkeit zum Steuern des lome6 ist der HTTP Server. Dazu den Inhalt des Verzeichnisses `services/lome6/embed` ins `embed/` kopieren und den HTTP Server anschalten.

## **7 ECMD**

lome6 erweitert die ECMD um folgende Befehle:

### **lome6 state**

Power Status (on, off)

### **lome6 power**

Trigger Power Taster kurz

### **lome6 power long**

Trigger Power Taster lang

### **lome6 reset**

Trigger Reset Taster

### **lome6 set\_t <type> <temperatur>**

Setze Temperatur, *type* kann "cpu" oder "sb" sein, *temperatur* in Decigrad!

### **lome6 get\_t <type>**

Hole Temperatur, *type* kann "cpu", "sb", "ram", "psu" oder "air" sein

### **lome6 uptime <uptime>**

Wenn *uptime* angegeben, wird diese gesetzt (in Sekunden).

Ansonsten wird die Uptime (in Sekunden) ausgegeben .