Zunächst möchte ich mich erstmal wirklich bei den Erschaffern von HAP bedanken. Im Netz findet man selten ein Projekt was so durchdacht ausgearbeitet wurde und ich hoffe mit der Quickstart-Anleitung auch Einsteigern die ersten Schritte etwas zu vereinfachen. Man findet alle Infos auch in der vollständigen Dokumentation aber es ist weit verstreut und daher wollte ich das gröbste einmal zusammenfassen.

Diese Zusammenfassung basiert auf Informationen von http://home-automation-project.netmb.net/ und dem Mikrocontroller-Forum.

Über Ergänzungen oder Verbesserungen würde ich mich freuen!

Grundlage für diese Quickstart-Anleitung ist:

- Windows PC mit AVR-Studio & WinAVR
- Ubuntu 9.04
- 1-2 aufgebaute Controller-Units (CU)
  ->INFO: wenn nur 1 CU dann CAN **nicht** aktivieren!
- Hterm (Terminalsoftware) für Linux
- USB oder RS232 Interface

1. Installation von Ubuntu

- Installation von Ubuntu Version 9.04
- In der Konsole mit **sudo**:

```
wget http://packages.netmb.net/PublicKey
apt-key add PublicKey
echo "deb http://packages.netmb.net/ubuntu ./" >> /etc/apt/sources.list
apt-get update
apt-get install hap
```

Nach der erfolgreichen Installation sollte man vor dem Start noch definieren, wie die Kommunikation zwischen HAP-Server und den CU's erfolgen soll. Dazu muss man noch eine Datei editieren:

```
cd /opt/hap/etc/
sudo gedit hap.yml
```

Sofern noch keine CU's vorhanden sind einfach die eigene Rechner IP verwenden, damit der Server ohne Fehler starten kann.

```
ServerCUConnection:
Type: 'Network'
Host: 192.xxx.xxx.xxx
Port: 4567
#ServerCUConnection:
#  Type: 'Serial'
#  Ports: [ '/dev/ttyUSB1' ]
```

Sofern CU's schon vorhanden sind kann man die Schnittstelle eintragen

```
#ServerCUConnection:
#  Type: 'Network'
#  Host: 192.xxx.xxx.xxx
#  Port: 4567
ServerCUConnection:
Type: 'Serial'
Ports: [ '/dev/ttyUSB1' ]    # hier die benutzte Schnittstelle eintragen!
```

Danach sollte der Start möglich sein (HAP neustarten vorher):

```
HAPConfig :   http://192.xxx.xxx.xxx :8090
HAP-GUI    :  http://192.xxx.xxx.xxx :8090/GUI
```

2. Erstellung der Firmware (CU)

Vorab sollte man im HAPConfig über Tools->DownloadBootlader den Bootloader exportieren. Dabei wird automatisch eine UID im Bootloader eingesetzt. Den Bootloader muss man für jede CU erstellen.

Die UID selber steht im Bootloader bzw im Namen:
HAPBootLoader-**0F2C07**.hex

Das direkte importieren des Projekts klappte bei mir leider nicht, daher:

1. Erstellen eines neuen Projekts in AVR (MEGA32)
2. import *.h & *.c & Makefile
3. Anpassung der mv.h

Die Mindestanforderungen könnt ihr im Wiki finden. Hier meine Einstellungen für alle Controller-Units:

```
#define COHAES              // EEPROM-Support              (Bit  0 -  0)
// #define COHAER           // Externer Reset             (Bit  1 -  1)
#define COHABZ              // Buzzer                     (Bit  2 -  2)
#define COHAFM              // Funkmodul                  (Bit  3 -  3)
#define COHACB              // CAN-Bus                    (Bit  4 -  4)
// #define COHAIR           // Infrarotschnittstelle      (Bit  5 -  5)
// #define COHALCD 2        // siehe oben                 (Bit  6 -  7)
 #define COHALI             // Logischer Eingang          (Bit  8 -  8)
 #define COHAAI             // Analoger Eingang           (Bit  9 -  9)
 #define COHADIDS1820    // Dallas Digitales Thermometer  (Bit 10 - 10)
 #define COHASW             // Geschalteter Ausgang       (Bit 11 - 11)
 #define COHADM          // Gedimmter Ausgang             (Bit 12 - 12)
// #define COHARS           // Rollladensteuerung         (Bit 13 - 13)
// #define COHADG 2         // siehe oben                 (Bit 14 - 15)
// #define COHAGUI          // Bedienoberfl‰che           (Bit 16 - 16)
 #define COHAAS          // Autonome Steuerung            (Bit 17 - 17)
```

CAN muss deaktiviert werden, wenn ihr nur 1 x CU habt! (COHACB)!

4. Kompilierung der Firmware in AVR-Studio
5. Programmierung per ISP (JP7)
6. Fuses setzen siehe WIKI
   (SPIEN, CKOPT, BOOTSZ=1024, BODLEVEL=4V, BODEN, Ext HF 1k+ 4ms)
7. CHIP ERASE & Programmierung des Bootloaders aus der HAPConfig
8. Danach Programmierung der Firmware **ohne Chip-Erase**
9. Öffnen von Hterm mit 19200baud & Zeichen Dezimal
10. Nach dem Anschalten sollte das Modul einen Zeitrequest senden

Hterm: 0 0 255 123 0 0 0 0 (0 = 000 in HTerm)

3. Moduleinrichtung im Terminal

Jede CU-Platine hat eine Moduladresse bzw. eine UID. Die UID wird beim exportieren des Bootloaders aus dem HAPconfig erstellt und muss für jede Platine einzeln generiert werden. Die Moduladresse ist jedoch frei wählbar von 0-239 und jede CU sollte eine eigene pro VLAN haben.
Die UID selber muss man dann neben der Programmierung durch den Bootloader auch noch einmal für jedes Modul (CU-Platine) in die HAPConfig per Hand eintragen werden. Ebenso muss man die selbst zugewiesene Moduladresse eintragen.

Um HTERM unter Ubuntu zu nutzen bitte HAP in der Konsole erst stoppen damit die Schnittstelle frei ist:

cd /etc/init.d/
sudo ./hap-mp stop
sudo ./hap-configserver stop

Starten kann man HAP im Anschluss nach der Nutzung von Hterm wieder mit:

cd /etc/init.d/
sudo ./hap-mp start
sudo ./hap-configserver start

Hier ein Bespiel für eine Basiseinrichtung einer CU nach dem flashen der kompilierten Firmware (neue Platine, Moduladresse =0):

Dieses kann man direkt in Hterm mit Type (DEC) eingeben und die CU sollte jeweils ein Feedback geben.

Das Protokoll ist :

VLAN SOURCE DESTINATION MTYPE DEVICE V0 V1 V2

xxx = neue Werte

Moduladresse setzen (ab hier Modul)	0	0	0	76	5	xxx	0	0
CCU-Adresse einrichten	0	0	Modul	76	6	xxx	0	0
Bridge-Mode (1=on , 0=off)	0	0	Modul	76	10	xxx	0	0
Startmodus	0	0	Modul	76	4	217	0	0
EE_Konfig speichern	0	0	Modul	76	8	0	0	0
Reset (full)	0	0	Modul	76	2	0	0	0

Der Startmodus, um die neue Konfiguration zu laden muss 217 sein.
Brigdemode müssen die Platinen haben, die Daten durchleiten sollen (z.B. CU-EG)
Die CCU-Moduladresse ist für alle CU in meinem Fall 99 siehe Kapitel 4.

4. Moduleinrichtung im HAPConfig

Im folgenden soll nur das Prinzip gezeigt werden. VLAN ist dabei immer 0.

CCU : Server-HAP : UID = 000000 , Moduladresse=99
CU-EG : CU-Erdgeschoss : UID = 0F 22 E5 , Moduladresse=100
OG : CU-1.Obergeschoss : UID = 0F 2C 07 , Moduladresse=101

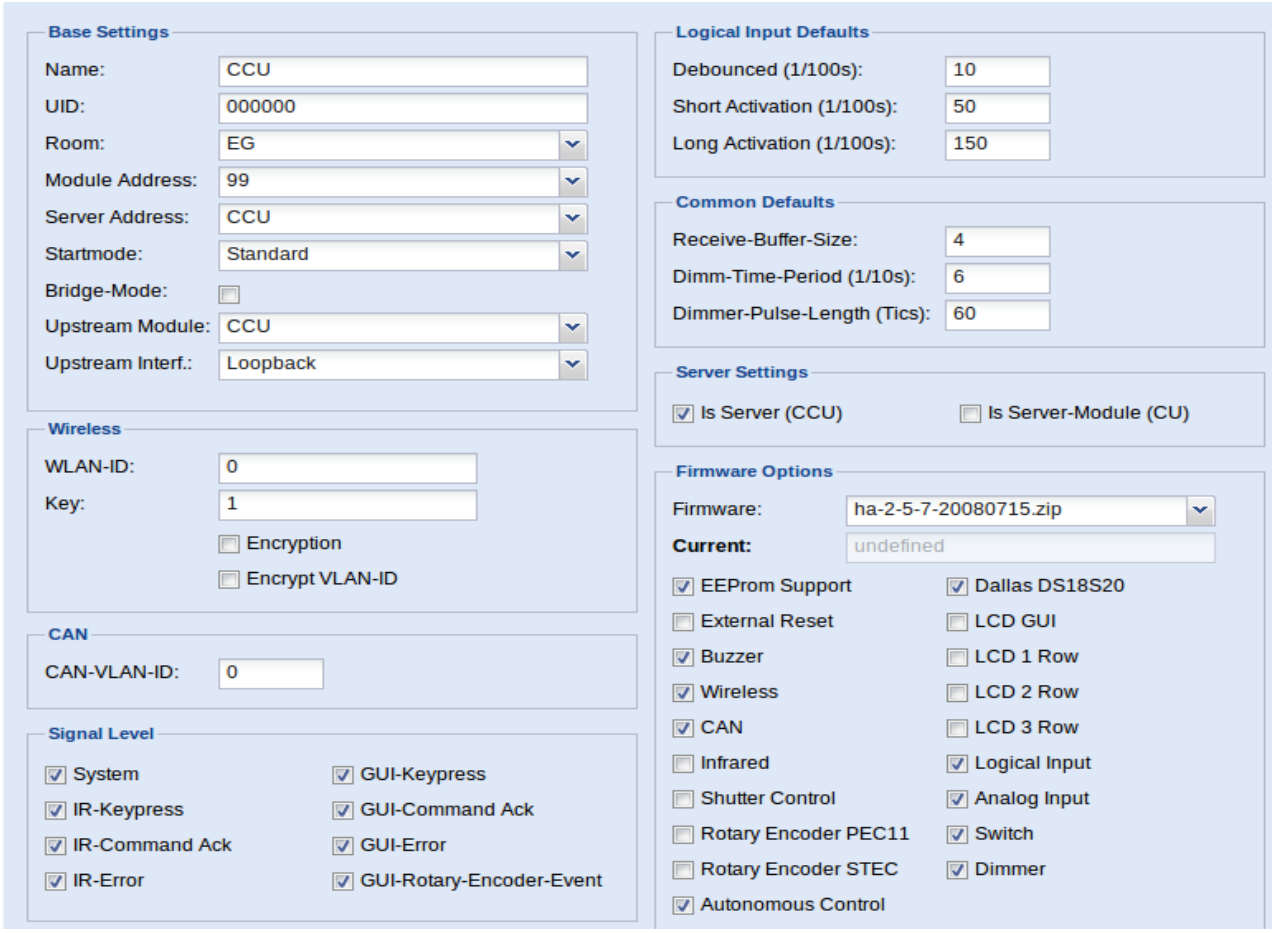
CCU *→SERIELL→* **CU-EG** *→CAN→* **OG**

Wie bereits beschrieben ist die UID=00 00 00 für die CCU. Die CCU ist virtuell und damit nur die Verbindung GUI<-> CU. Die Schnittstelle von PC zu den restlichen Units bildet **eine** CU (CU-EG). Alle weiteren CU's im CAN haben keinen Häkchen bei CCU oder CU unter Server-Settings.

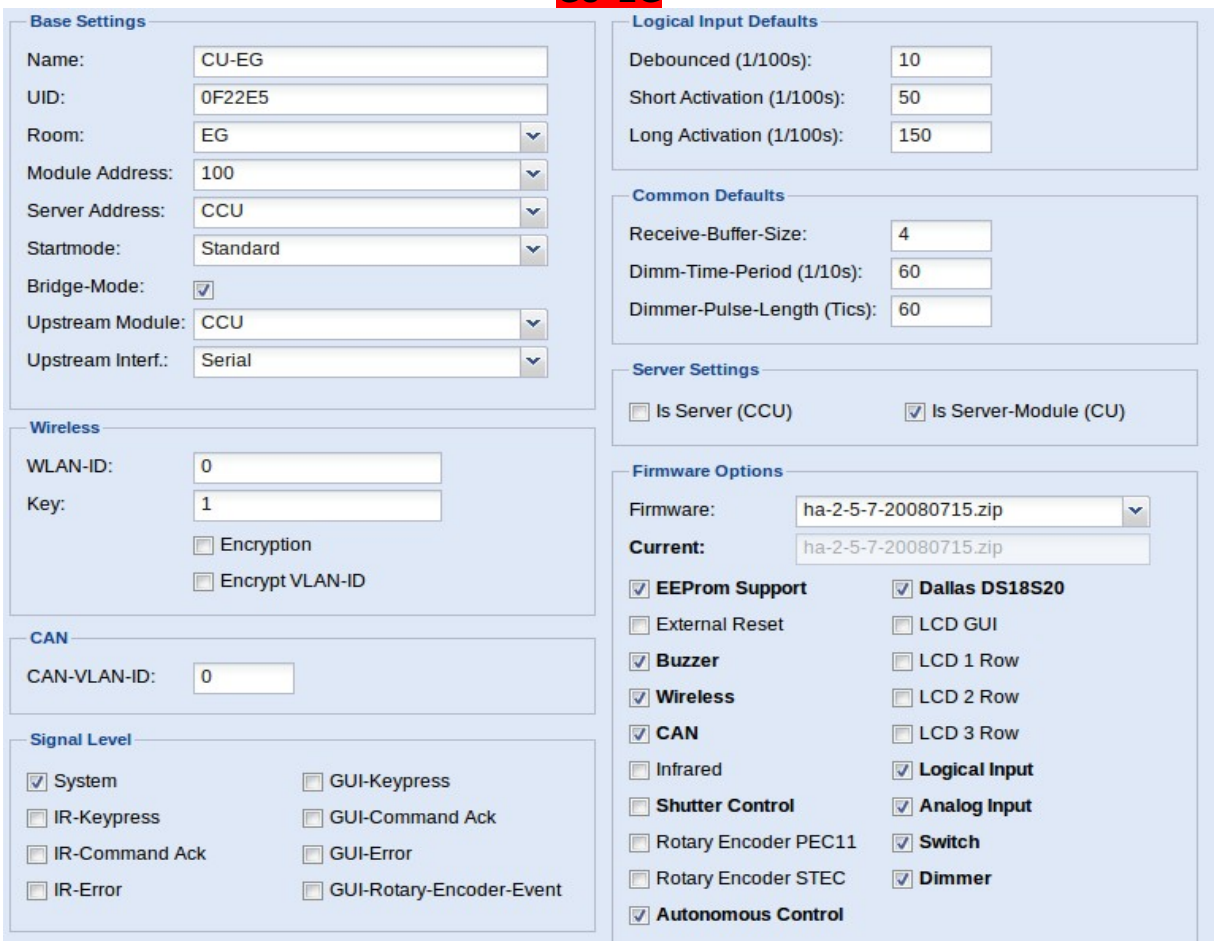
HAPConfig benötigt zum programmieren ebenfalls noch die unveränderte Firmware von der Homepage als zip. Diese muss man zuvor noch mit Tools – File Upload einbinden.

Hier Bilder als Beispiel wie es funktioniert:

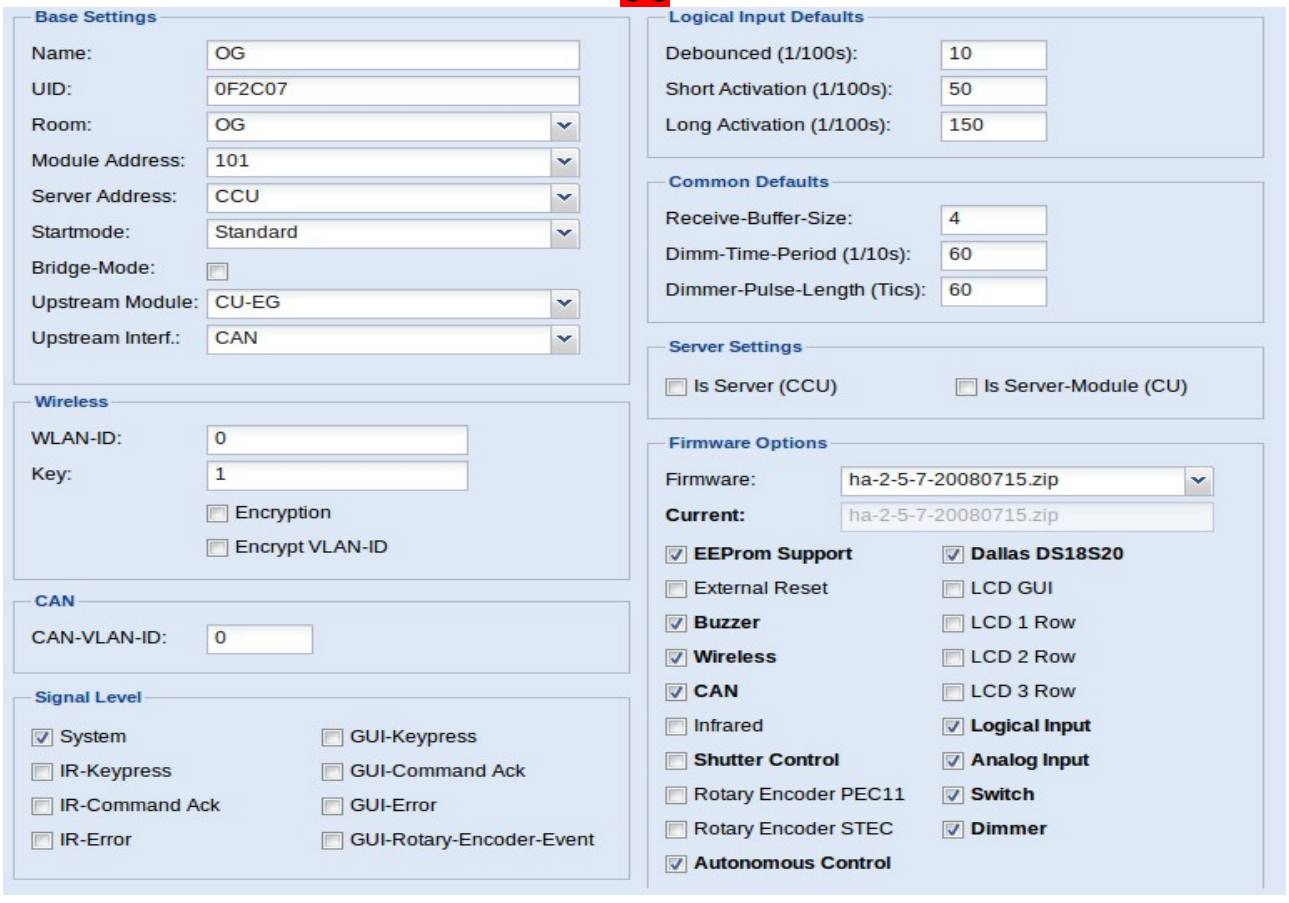
CCU



CU-EG


Base Settings
Name: CU-EG
UID: 0F22E5
Room: EG
Module Address: 100
Server Address: CCU
Startmode: Standard
Bridge-Mode:
Upstream Module: CCU
Upstream Interf.: Serial
Wireless
WLAN-ID: 0
Key: 1
Encryption
Encrypt VLAN-ID
CAN
CAN-VLAN-ID: 0
Signal Level
System
IR-Keypress
IR-Command Ack
IR-Error
GUI-Keypress
GUI-Command Ack
GUI-Error
GUI-Rotary-Encoder-Event
Logical Input Defaults
Debounced (1/100s): 10
Short Activation (1/100s): 50
Long Activation (1/100s): 150
Common Defaults
Receive-Buffer-Size: 4
Dimm-Time-Period (1/10s): 60
Dimmer-Pulse-Length (Tics): 60
Server Settings
Is Server (CCU)
Is Server-Module (CU)
Firmware Options
Firmware: ha-2-5-7-20080715.zip
Current: ha-2-5-7-20080715.zip
EEProm Support
External Reset
Buzzer
Wireless
CAN
Infrared
Shutter Control
Rotary Encoder PEC11
Rotary Encoder STEC
Autonomous Control
Dallas DS18S20
LCD GUI
LCD 1 Row
LCD 2 Row
LCD 3 Row
Logical Input
Analog Input
Switch
Dimmer

OG


Base Settings
Name: OG
UID: 0F2C07
Room: OG
Module Address: 101
Server Address: CCU
Startmode: Standard
Bridge-Mode:
Upstream Module: CU-EG
Upstream Interf.: CAN
Wireless
WLAN-ID: 0
Key: 1
Encryption
Encrypt VLAN-ID
CAN
CAN-VLAN-ID: 0
Signal Level
System
IR-Keypress
IR-Command Ack
IR-Error
GUI-Keypress
GUI-Command Ack
GUI-Error
GUI-Rotary-Encoder-Event
Logical Input Defaults
Debounced (1/100s): 10
Short Activation (1/100s): 50
Long Activation (1/100s): 150
Common Defaults
Receive-Buffer-Size: 4
Dimm-Time-Period (1/10s): 60
Dimmer-Pulse-Length (Tics): 60
Server Settings
Is Server (CCU)
Is Server-Module (CU)
Firmware Options
Firmware: ha-2-5-7-20080715.zip
Current: ha-2-5-7-20080715.zip
EEProm Support
External Reset
Buzzer
Wireless
CAN
Infrared
Shutter Control
Rotary Encoder PEC11
Rotary Encoder STEC
Autonomous Control
Dallas DS18S20
LCD GUI
LCD 1 Row
LCD 2 Row
LCD 3 Row
Logical Input
Analog Input
Switch
Dimmer