Hausbus for everyone

Ein Gemeinschaftsprojekt

Vorwort

Im Zeitalter des Netzes, wo jeder mit jedem und alles mit allem kommuniziert sollte auch das eigene Heim nicht ausgeschlossen werden. Enerie wird teurer und der Mensch immer fauler und komfortabler. Darum brauchen wir noch mehr Elektrogeräte im Haus. Ein Wiederspruch? Nein, diese Geräte sollen uns helfen Energie zu sparen und dabei den alltäglichen Komfort erhöhen.

Wer schaltet den Fernseher, die Stereoanlage, die Set-Top-Box, den Computer, und all die Energiefresser aus, wenn er außer Haus geht? Richtig, niemand. Oft wird auf brennende Lichter vergessen. Die Heizung läuft 24 Stunden, obwohl keiner zuhause ist. Es wird dauernd unnötige Energie verschwendet! Dies belastet nicht nur unsere Geldttasche, sondern auch die Umwelt.

„**Machen wir die Welt ein bisschen smater**“, heißt es in einem Werbespot eines Computer- und Elektronikgiganten. Dieses Projekt soll genau das bewirken. Mit einem Hausbus soll Energie gespart aber dabei trotzdem der tägliche komfort erhöht werden.

Table of Contents

Vorwort **1**

Anforderungen **2**
- **Billig** **2**
- **Energieeffizient** **2**
- **Einfache Installation** **2**
- **Benutzerfreundlichkeit** **2**

Schematischer Aufbau und Funktionsweise **2**
- **BUS** **2**
- **BUS Protokoll** **3**
- **Clients** **3**
  - Stromsparen an den Clients 3
- **Stromversorgung** **3**
- **RS485 <-> Ethernet** **3**
- **Anwender** **4**

Details **4**
- **RS485 Clients** **4**
  - Software 4
  - Hardware 6
  - Hardware BUS Sniffer 6
- **NET-IO Board = Schnittstelle und RS485 Master** **6**
  - Software 6
  - Hardware 7
- **Linux Server** **7**

Anforderungen

an den Hausbus.

Billig

Die einzelnen Komponenten sollen für jeden leistbar sein. 10€ für das Material einer Komponente soll nicht überschritten werden.

Energieeffizient

Das Bussystem im Haus soll Energie sparen. Das heißt, die Elektronikkomponenten dürfen nicht mehr Energie verbrauchen, als dadurch eingespart wird.

Einfache Installation

Auch ein interessierter Leihe sollte das System installieren können. Sowohl Hardware als auch Software.

Benutzerfreundlichkeit

Jeder Hausbewohner, ob jung oder alt, muss das intelligente Eigenheim bedienen können. (Smartphone, PC, Userpanel)

Erweiterbarkeit

Ein einfaches und leistungsfähiges Protokoll sollen jeden Hobbybastler die Gelegenheit bieten ganz einfach verschiedenste Komponenten selbst zu bauen.

Der Hardwarespezialist kann vorhandene Komponenten weiterentwickeln und neue Sensoren und Aktoren hinzufügen.

Die Software kann ganz einfach angepasst, erweitert und individualisiert werden. Dies wird durch definierte Softwareschnittstellen ermöglicht.

Gemeinsame Dokumentation

Mit einer gemeinsamen Dokumentation und Bedienungsanleitung können viele Leute an dem Projekt Mitarbeiten.

Schematischer Aufbau und Funktionsweise

BUS

RS-485 eignet sich meiner Meinung am besten für die Vernetzung der einzelnen Clients. Multimasterfähig + 32 Clients + via UART vom uC ansprechbar + billige Komponenten + 2 Draht + lange Leitungslängen.

Sollten mehr als 32 Clients benötigt werden kann man die Übertragungsgeschwindigkeit herabsetzen und bis zu 128 Clients dranhängen, oder man nimmt einen uC mit zwei UART's und baut eine Bridge.

BUS Protokoll

Ein einfaches selbstprogrammiertes Kommunikationsprotokoll würde voll und ganz ausreichen.

Protokoll Paket:
1Byte Typ + 1Byte Dest. Adr. + 1Byte Src. Adr. + 2Byte Length + Data + 2Byte CRC16

Clients

Jeder Client besteht aus einem MAX485/ADM483 Bustreiber und einem ATMega mit Hardware UART.

- schaltbare Steckdose
- Lichtschalter (Relay + beliebig viele Taster)
- Dimmer
- Rollosteuerung
- Heizungssteuerung
- etc.

Stromsparen an den Clients

Die Clients schlafen grundsätzlich. Das heißt, der ATMega befindet sich im sleep Zustand. Er wird nur geweckt, wenn er gebraucht wird. Mit einem Interrupt an der UART oder mit einem Interrupt von einem Sensor (Taster, ...).

Stromversorgung

Die Stromversorgung erfolgt Zentral. Zusätzlich zu den 2 Twisted Pair Leitungen vom RS485 Bus kommen noch +13,8 und GND.

Die Busleitung besteht also insgesamt aus 4 Adern.

RS485 <-> Ethernet

Eine Verbindung vom RS485 Hausbus zum IP Netz ist nötige, damit man einfach und von Überall sein Haus unter Kontrolle hat.

Am einfachsten geht das am Anfang mit einem NET-IO Board von Pollin. Dies sollte aber noch nicht die Schnittstelle zum Anwender (Smartphone, PC) sein. Mit dem NET-IO Board sollten die Clients im RS485 Bussystem angesprochen und ausgelesen werden. Ein Linux Server kommuniziert mit dem NET-IO Board. Es würde sich ein Linksys WLAN Router eigenen, da diese ganz einfach mit Linux geflashed werden können.

Warum brauchen wir einen Zusätzlichen Webserver?
Der Linux Server ist die Schnittstelle zum Anwender.

- Firewall
- VPN
- Php für Benutzerverwaltung

- Möglicherweise gibt es nicht nur ein NET-IO Board im Haus. Es können z.B. auch Webcam's etc. angesprochen werden.
- ...

Anwender

Der Anwender steuert sein Haus über ein Webinterface. Dieses ist angepasst für Smartphones, PC's und alle anderen Webclients.

Details

Es folgt eine detaillierte Schnittstellendefinition.

RS485 Clients

Software

Die Software muss auf allen ATMegas funktionieren.

Alle Clients sind Multimaster. Sie können auch ohne Aufforderung senden. z.B. HALLO, wenn sie eingeschaltet werden.

Auf jede RS485 Paketanfage gibt es eine Antwot (ACK, NACK, PING PONG)

Eine Kollisionserkennung findet zur Zeit nicht statt.

UART

uart.h, uart.c

Über die Hardware UART wird der MAX485 Bustreiber gesteuert.

```
//Öffentliche Methoden:
unsigned int uart_init  ();
unsigned int uart_put_c (unsigned char c);
unsigned int uart_get_c (); //High byte error code
unsigned int uart_put_s (unsigned char * s); //0x00 terminated

//Rückgabewerte:
#define
```

RS485 Bus Protokoll

rs485.h, rs485.c

1. 2 Byte Type
2. 1 Byte Destination Address
3. 1 Byte Source Adress
4. 2 Byte Length
5. x Byte Daten
6. 2 Byte CRC16

```
// Paket Struktur:
struct Packet
{
      unsigned int     errorcode;
```

```
      unsigned char    type;
      unsigned char    dest;
      unsigned char    src;
      unsigned int     length;
      unsigned char*   data;
      unsigned int     crc;
};

//Öffentliche Methoden:
unsigned int rs485_init ();
unsigned int rs485_send_packet(unsigned int type, unsigned char dest,
unsined char* data)
unsigned int rs485_receiv_packet (struct Packet * packet);
```

RS485 Packet Controller

rs485controller.h, rs485controller.c

Der Packet Controller ließt ein Packet ein, kontrolliert es und leitet es je nach Type weiter. Beispiel: es trifft ein Packet vom Type ping ein. Der Controller leitet es nach Überprufung an eine ping Routine weiter.

```
// Öffentliche Methoden:
unsigned int rs485controller_run ();
```

RS485 Bootloader

Neue Firmware muss über RS485 aufgespielt werden können

Sensoren und Aktoren Libraries

Je nach dem Type Feld im RS485 Packet wird die REMOTE methode des Sensors / Aktors aufgerufen.

Taster und Relay über INT0

client_switch_int0.h, client_switch_int0.c

```
//Öffentliche Methoden:
unsigned int client_switch_int0_init();
unsigned int client_switch_int0_status();
unsigned int client_switch_int0_switch(unsigned char type);
unsigned int client_switch_int0_remote(struct Packet packet);
```

Taster und Relay über INT0

client_switch_int1.h, client_switch_int1.c

Taster und Relay ohne externen Interrupt (Problem: uc kann nicht in den Sleep Modus gesetzt werden)

Phasenschnittdimmer

PWM Mortorsteuerung

Schrittmotorsteuerung
...

Main Prog
Stromsparen mit Sleepmodus.

Hardware
Die Hardware muss in eine Standard-Unterputzdose passen (Druchmesser ca. 55mm)

BUS
Für die serielle Übertragung am RS485 Bus werden 2 Leitungen benötigt (Receiv, Transmit). Twisted Pair Leidtungen sind nicht vorgeschrieben, aber sie senken die Störanfälligkeit, erhöhen die max. Übertragungsgeschwindigkeit und erweitern die max. Buslänge.

Zusätzlich benötigt man noch zwei Leitungen zur Stromversorgung. Ich würd 18 Volt bzw. 13,8 Volt empfehlen.

Jeder knoten hat zwei Stecker und somit Läuft der Bus direkt über die Platine des Kotens.

Stromversorgung Knoten
MAX485 und uC Laufen mit +5 Volt. Spannungswandler

Hardware BUS Sniffer
Um den Bus debuggen und überprüfen zu können wird ein Hardware Sniffer benötige. Dieser wird wie ein Knoten am Bus installiert.

MAX485->MAX232->Serielle Schnittstelle des PCs.

Mit einem Teminalprogramm (Hyperterm) können die Daten am Bus visualisiert werden.

NET-IO Board = Schnittstelle und RS485 Master

Software

UART / RS485 Bus Protokoll
Exakt das selbe wie bei den Clients.

Webserver
Der Webserver bietet HTML-Seiten zur Sensor- und Aktorsteuerung der Clients an. Er Speichert zustände der Clients und bietet diese ebenfalls an. Die Zustände werden auf Bestellung von den Clients abgerufen.

Auf die Sicherheit muss nicht geachtet werden. Wir befinden uns im Intranet. Firewall, Verschlüsselung und Benutzerverwaltung übernimmt der Linux Server.

Die Kommunikation mit dem Linux server könnte natürlich auch über ein eigenes UDP oder TCP Protokoll erfolgen. Aus Gründen der Einfachheit wird aber HTTP bevorzugt.

Hardware

Für den Anfang reicht das NET-IO Board von Pollin.

Linux Server

Sicherheit, Benutzerverwaltung, Firewall, VPN.

Die Logiksteuerung des Hauses ist auch hier Zuhause: Temperaturüberwachung, Energiesparmodis, etc....

Es können Natürlich auch mehrere NET-IO Boards und andere Ethernet Clients (Webcam) verwaltet werden.

Der Endanwender greift auch hier auf die Weboberfläche zu.