Link zur Dokumentation: API

Primäre Ziele der API:

- Einfach für Hardware Neueinsteiger
- Keine komplizierten C / C++ schreibweisen
- z.T. Angelehnt an Arduino Funktionen

Intern GPIO Ausgang

```
#define BOARD GNUBLIN
//#define BOARD RASPBERRY_PI

#include "gnublin.h"

int main()
{
   gnublin_gpio gpio;

   gpio.pinMode(3,OUTPUT);

   while(1){
     gpio.digitalWrite(3,HIGH);
     sleep(2);
     gpio.digitalWrite(3,LOW);
     sleep(2);
   }
}
```

Intern GPIO Eingang

```
#define BOARD GNUBLIN
//#define BOARD RASPBERRY_PI

#include "gnublin.h"


int main()
{
   gnublin_gpio gpio;

   gpio.pinMode(3,INPUT);

   while(1){
     if(gpio.digitalRead(3))
```

```
     {
        printf("GPIO is set \n");
     }
     sleep(2);
   }
}
```

Intern Analog Eingang

```
#define BOARD GNUBLIN
//#define BOARD RASPBERRY_PI

#include "gnublin.h"


int main()
{
   gnublin_adc ad;


   while(1){
        printf("AD value %i \n",ad.getValue(1));
   }
}
```

Module-Relay (Relaiskarte)

```
#define BOARD GNUBLIN
//#define BOARD RASPBERRY_PI

#include "gnublin.h"


int main()
{
   gnublin_module_relay relay;

   relay.setAddress(0x72);

   relay.switchPin(1,ON);

}
```

Module-Display (2x16 Zeichen)

```
#define BOARD GNUBLIN
//#define BOARD RASPBERRY_PI

#include "gnublin.h"


int main()
{
   gnublin_module_dogm dogm;

   dogm.init();
   dogm.setRsPin(14);
   dogm.setCS(11);

   dogm.print("Hallo Welt");

   sleep(2);

   dogm.clear();
   dogm.print("Zeile 1", 1);
   dogm.print("Zeile 2", 2);
   dogm.shift(5);

   sleep(2);

   dogm.returnHome();
   dogm.clear();
   dogm.print("Zeile 1, Offset 2", 1, 2);

   sleep(2);

   dogm.controlDisplay(0,1,0);
}
```

Module-Temperature (Temperatursensor)

```
#define BOARD GNUBLIN
//#define BOARD RASPBERRY_PI

#include "gnublin.h"


int main()
{
   gnublin_module_lm75 lm75;


   lm75.setAddress(0x4f);
```

```
   printf("Temperature %i \n", lm75.getTemp());
   printf("Raw Value %i \n",lm75.getValue());
   printf("Temperature Float Value %f \n",lm75.getTempFloat());

}
```

Module-Step (Schrittmotor)

```
#define BOARD_GNUBLIN
//#define BOARD RASPBERRY_PI

#include "gnublin.h"


int main()
{
        gnublin_module_step motor;


        motor.setAddress(0x76);
        motor.getFullStatus1();
        motor.runInit();
        motor.resetPosition();
        motor.setPosition(1000);
        return 0;

}
```

Module-RTC (Uhrzeit)

```
#define BOARD GNUBLIN
//#define BOARD RASPBERRY_PI

#include "gnublin.h"


int main()
{
   gnublin_module_rtc rtc;


   rtc.setAddress(0x72);

   while(1){
        printf("Timestamp %i \n",rtc.getTimestamp());
   }
}
```

I2C

```
#define BOARD GNUBLIN
//#define BOARD RASPBERRY_PI

#include "gnublin.h"


int main()
{
   gnublin_i2c i2c;

   i2c.setAddress(0x42);

   char buffer[8];
   char RxBuf[8];

   buffer[0]=0x22;

   i2c.send(buffer,5);
   i2c.send(0x12, buffer, 2);   //sende 2 byte aus buffer an Register-Adresse
0x12

   i2c.receive(RxBuf, 3);       // lese 3 bytes und speichere sie in RxBuf
   i2c.receive(0x23, RxBuf, 3);  // lese von Register-Adresse 0x23 3 bytes
und speichere sie in RxBuf

}
```

SPI

```
#define BOARD GNUBLIN
//#define BOARD RASPBERRY_PI

#include "gnublin.h"


int main()
{
   gnublin_spi spi;

   spi.setSpeed(10000);
   spi.setCS(14);
   unsigned char send_buffer[9] = "getValue";
   char buffer[9];
   while(1){
     spi.send(sendbuffer,8);
     sleep(2);
```

```
     spi.receive(buffer,8);
   }
}
```

Wrapper for Webcam

```
#define BOARD GNUBLIN
//#define BOARD RASPBERRY_PI

#include "gnublin.h"


int main()
{
   gnublin_cam cam;


   cam.doSnapshot("/tmp/test.jpg");
}
```

Allgemeine Funktionen

- Dateien lesen / schreiben
- CSV Parser
- MySQL / sqlite
- Threads

Ordner Struktur

- examples *Beispielprogramme*
- modules *Klassen für Module*
- boards *Defintion für Boards*
- common *Allgemeine Funktionen wir println, delay, ...*
- drivers *I2C, UART, SPI, ... Funktionen dienen als Basis für Module & Co.*