



# mbed

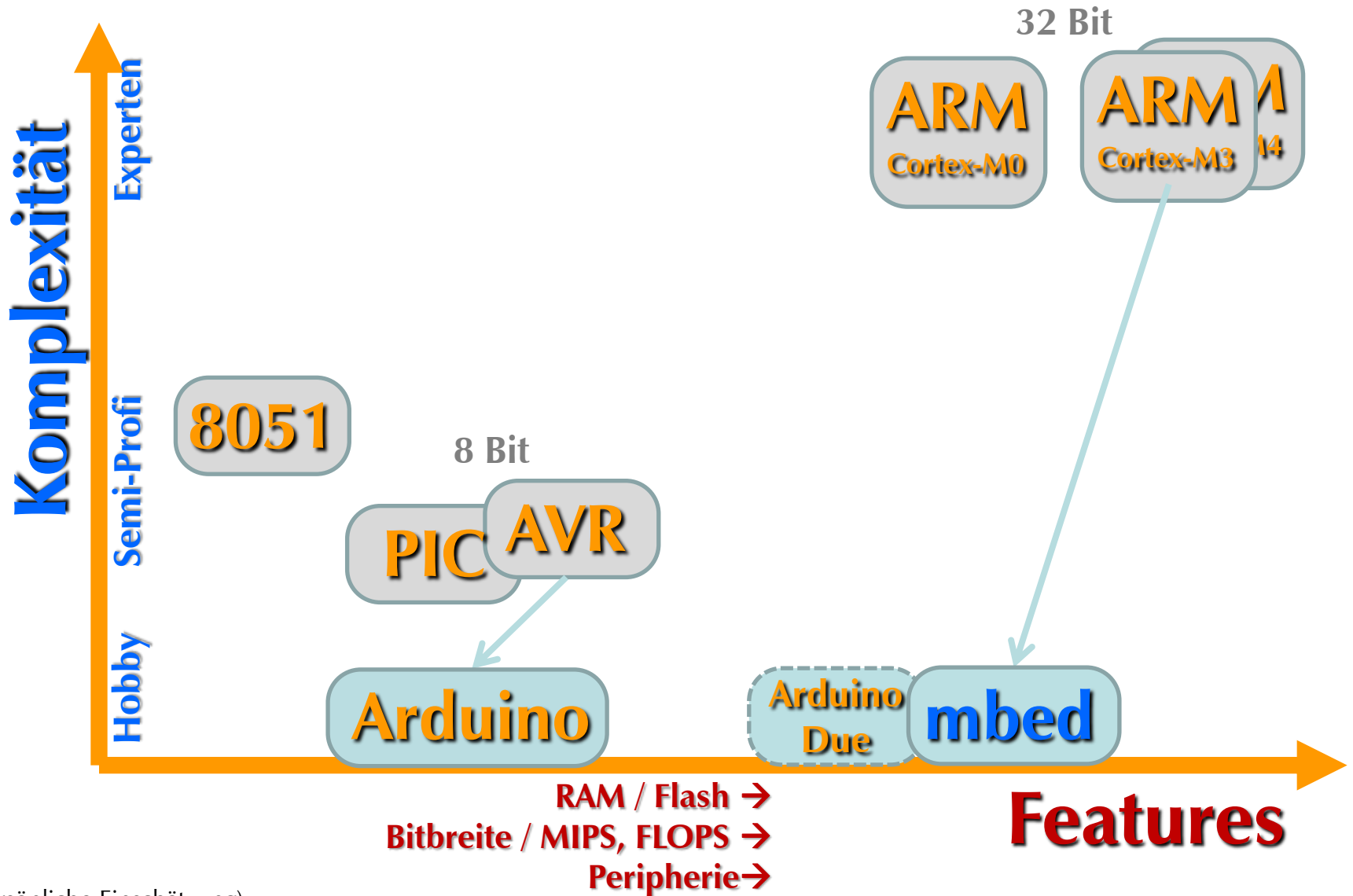
**mbed - der schnellste Einstieg  
in Web-basierte Steuerung und Regelung**

KNF-Kongress 2012 - Jochen Krapf

# Agenda

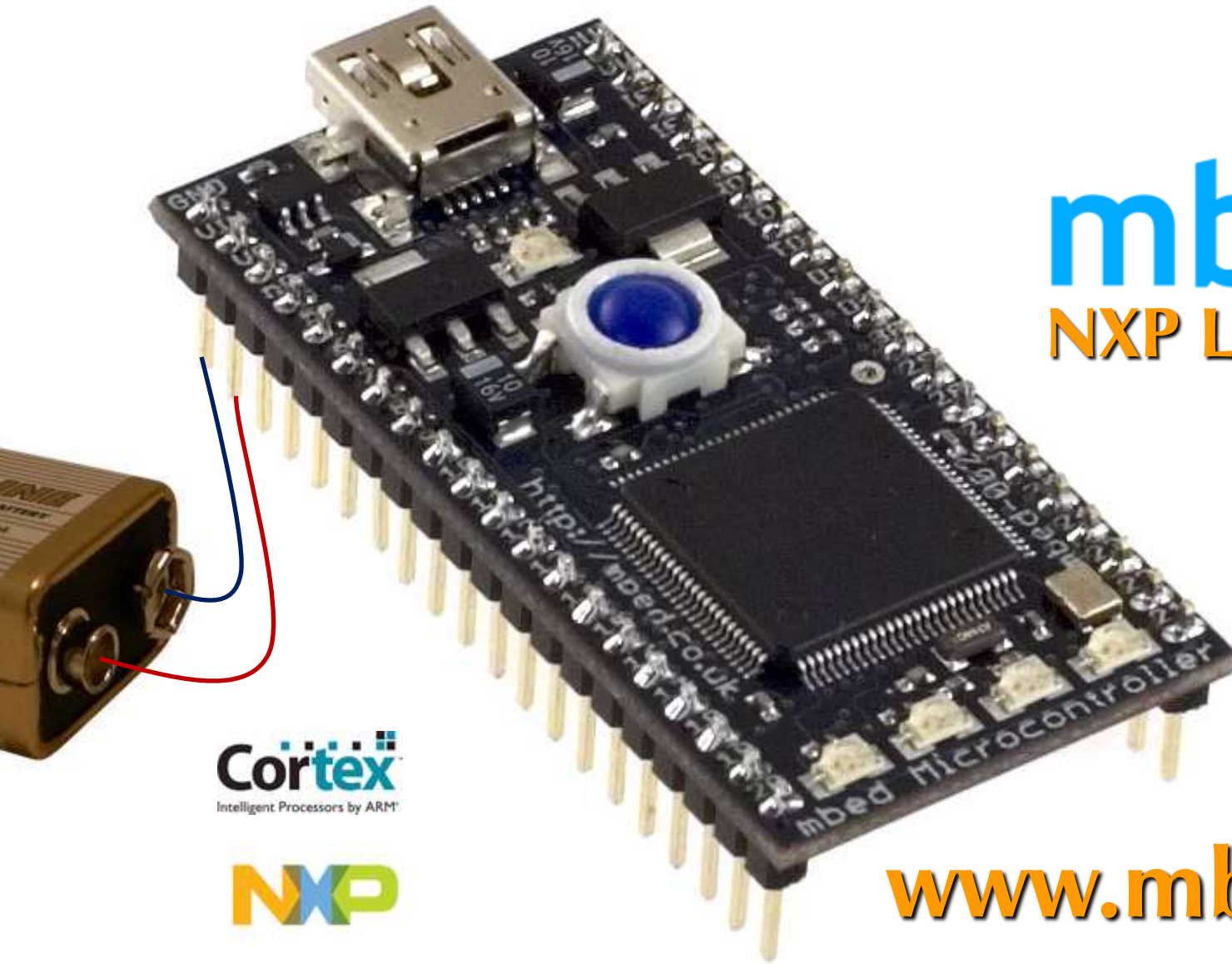
- **mbed ?!**
- **Typische Anwendungen**
- **Entwicklungsumgebung**
- **Libraries**
- **Mit mbed ins Netz**
- **Referenzen**
  
- **Internet of Things (2. Teil)**

# mbed → System-Auswahl



(persönliche Einschätzung)

# mbed → Modul



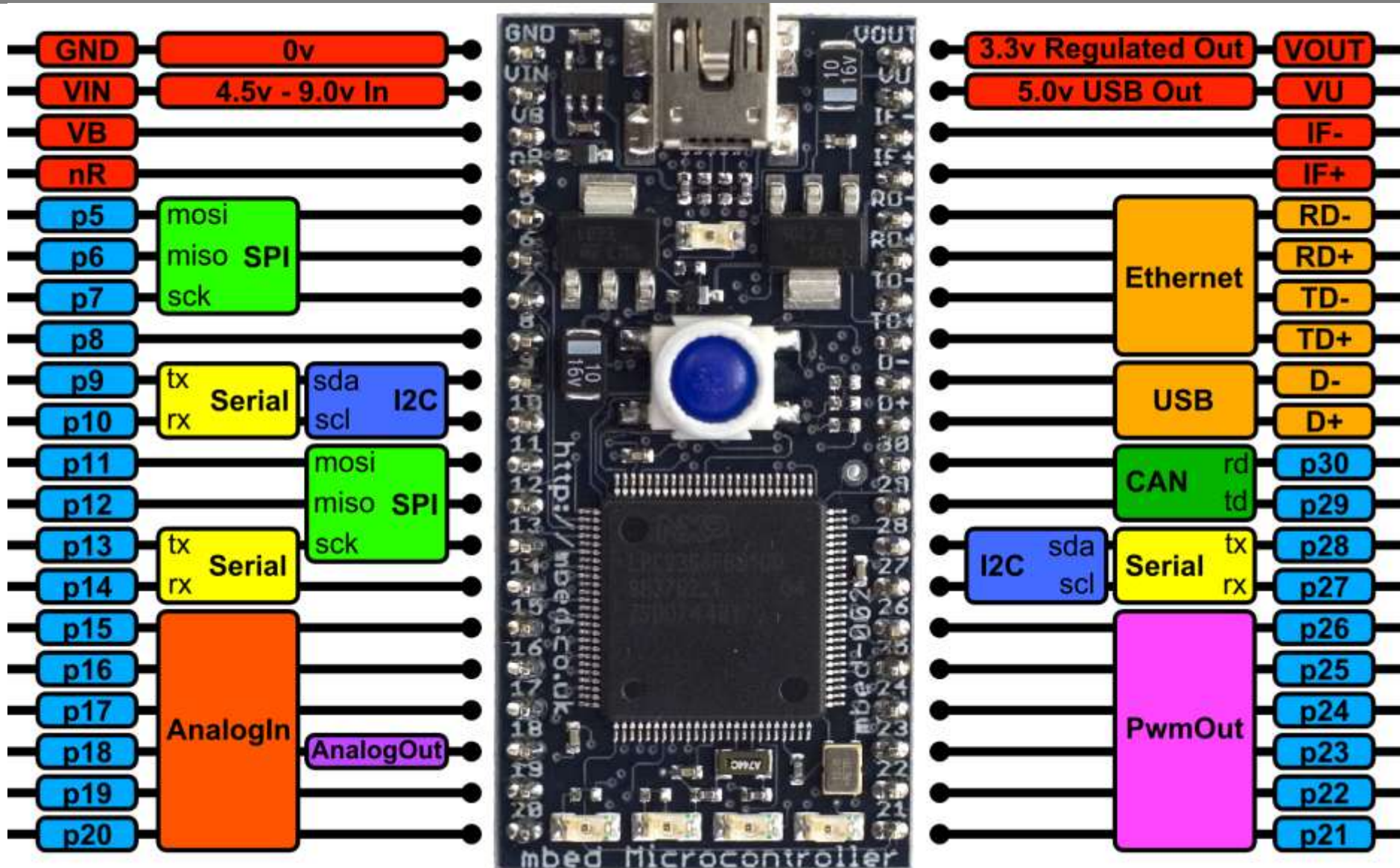
**mbed**  
**NXP LPC1768**

**Cortex**  
Intelligent Processors by ARM<sup>®</sup>

**NXP**

**[www.mbed.org](http://www.mbed.org)**

# Mbed → Pinout



# mbed → Features HW

<b>Feature</b>	<b>mbed</b>	<b>Arduino Uno</b>
<b>CPU</b>	<b>ARM Cortex-M3</b>	<b>ATmega328</b>
<b>Takt</b>	<b>96MHz</b>	<b>16MHz</b>
<b>RAM</b>	<b>32kB</b>	<b>2kB</b>
<b>FLASH</b>	<b>512kB</b>	<b>32kB</b>
<b>I/O</b>	<b>3,3V (5V tol.) mit 40mA</b>	
<b>Analog-In</b>	<b>0...3,3V</b>	<b>=</b>
<b>On-Board</b>	<b>Eth.Phy, RTC, CAN, DC/DC...</b>	
<b>Pinout</b>	<b>Raster 2,54mm</b>	<b>Proprietär</b>

# mbed → Features SW

<b>Feature</b>	<b>mbed</b>	<b>Arduino Uno</b>
<b>Code</b>	<b>C++</b>	<b>ähnlich C</b>
<b>Compiler</b>	<b>Online</b>	<b>Java-Prog.</b>
<b>Ext. Compiler</b>	<b>ja (Keil, IAR, GCC)</b>	<b>-</b>
<b>SdtLib...</b>	<b>ja</b>	<b>-</b>
<b>RTOS</b>	<b>ja</b>	
<b>...</b>		
<b>...</b>		
<b>Community</b>	<b>ja</b>	<b>ja</b>

# mbed → Konzept CMSIS

(CMSIS = Cortex Microcontroller Software Interface Standard)

```
void setPWM(int width, int period){
    LPC_SC->PCONP |= (1 << 6); // PWM1 power on
    LPC_SC->PCLKSEL0 |= (1 << 12); // Divide CCLK by 1 for PWM1
    LPC_PINCON->PINSEL4 |= 1; // DIP26 = pin P2.0, set to PWM1.1
    LPC_PINCON->PINMODE4 |= 2; // enable neither pull up nor pull down.
    LPC_PWM1->TCR = 0; // reset PWM
    LPC_PWM1->MCR = 0; // no interrupts (set for no interrupts by default)
    LPC_PWM1->MR0 = period; // set pulse period in clock cycles
    LPC_PWM1->MCR |= 1 << 1; // Reset timer on Match0
    LPC_PWM1->MR1 = width; // set pulse width in clock cycles
    LPC_PWM1->LER = 3; // make match register writes effective for MR0+MR1
    LPC_PWM1->PCR |= (1 << 9); // enable PWM1 output, single edge mode
    LPC_PWM1->TC = 0; // clear timer counter
    LPC_PWM1->PC = 0; // clear prescale counter
    LPC_PWM1->PR = 0; // clear prescale register
    LPC_PWM1->TCR = 8+1; // enable PWM mode and PWM timer counter
}
```

Nur für Experten!



# mbed → Konzept C++

Für ALLES gibt es eine Klasse...

C++ Klasse

#include "mbed.h"

DigitalOut

int

Objekt

myrelais

i

Parameter  
(Konstruktor)

(p26);

= 0;

# mbed → Konzept C++

```
#include "mbed.h"
int main() {

// <Class>    <Object>(<Parameter>);
  DigitalOut myrelais(p26);

// <Object>.<Method>(<Parameter>);
  myrelais.write(1);

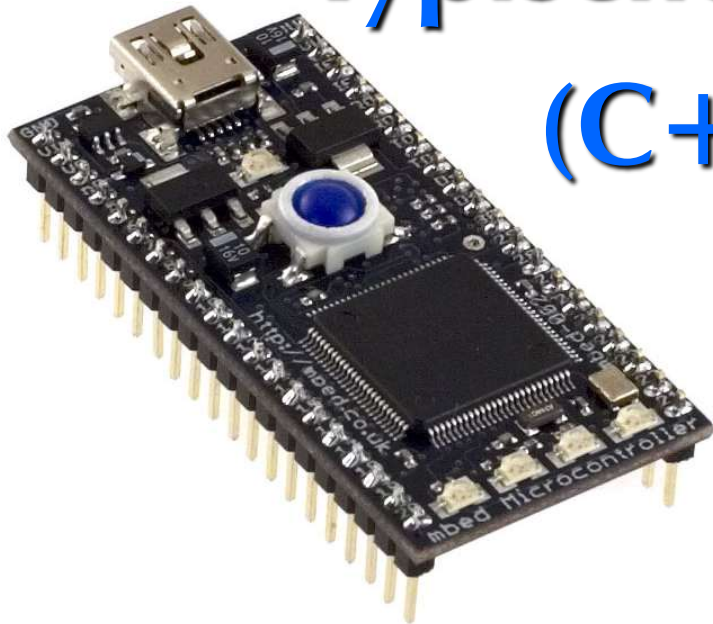
// <Object> = <Value>;
  myrelais = 1;           // operator=() ---> .write(1)

// <Variable> = <Object>.<Method>(<Parameter>);
  i = myrelais.read();

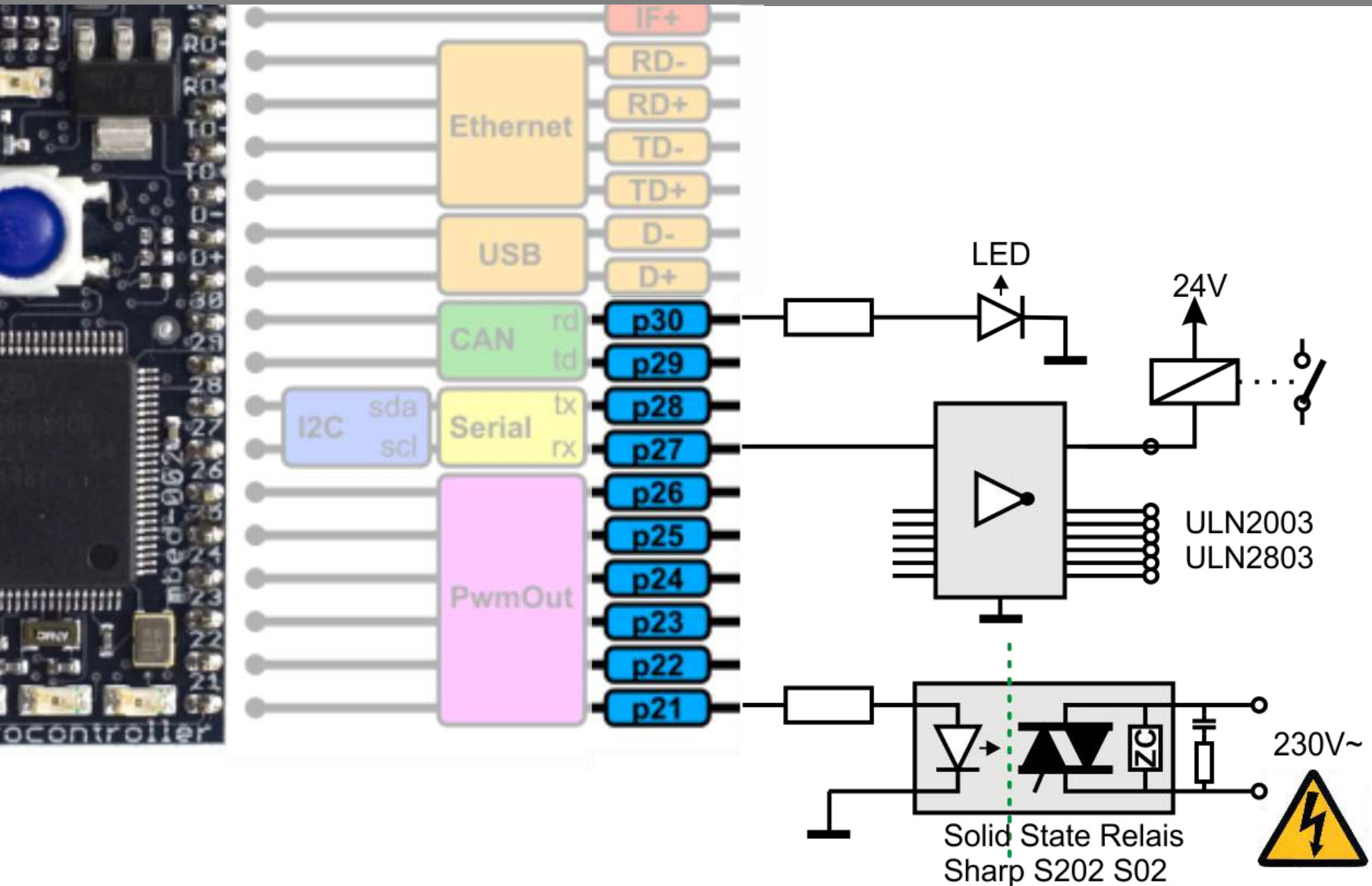
// <Variable> = <Object>;
  i = myrelais;         // int operator() ---> .read()
}
```

# Beispiele

## Typische Anwendungen (C++ Klassen)



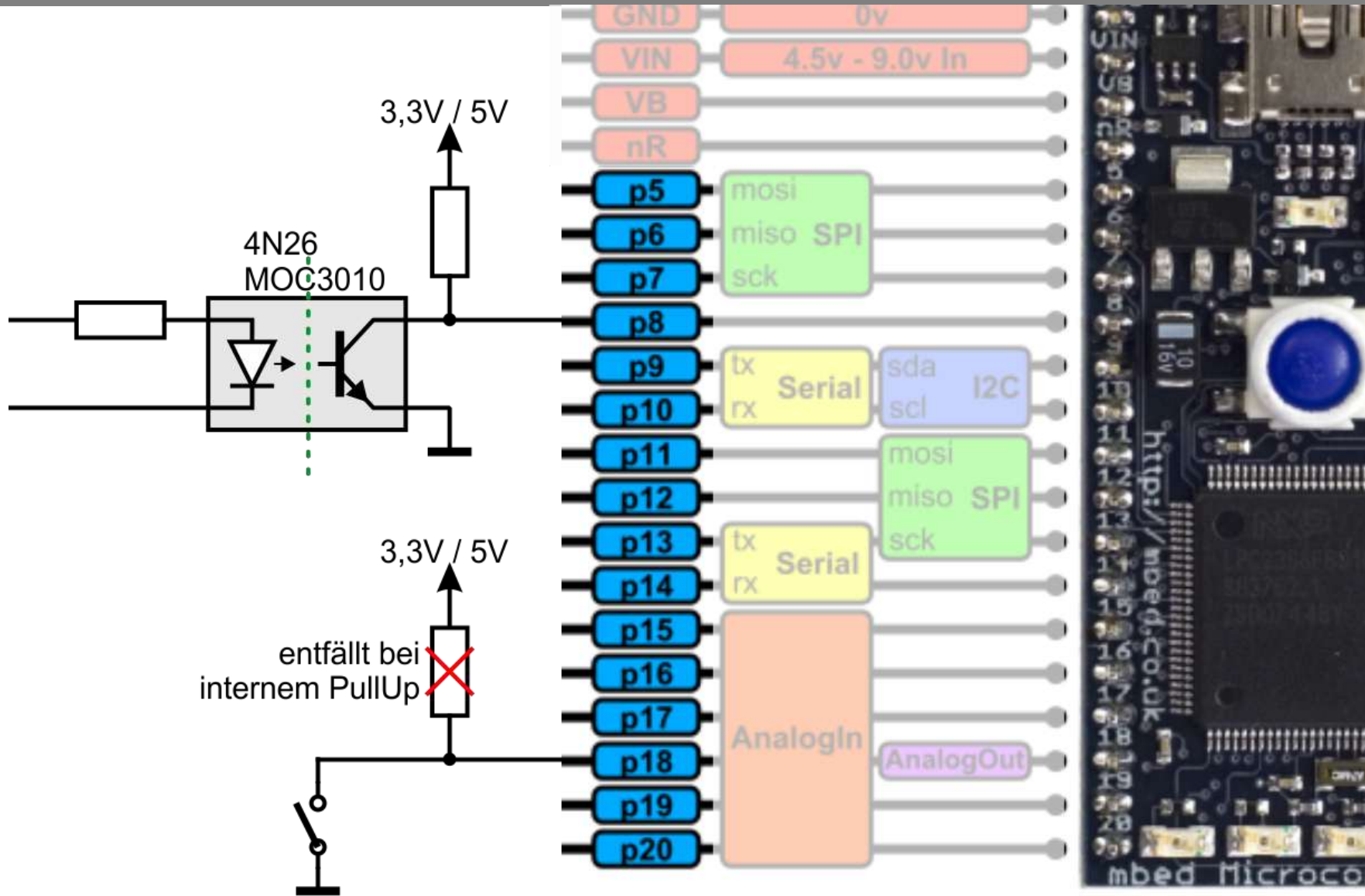
# Beispiele → Digital-Out



# Beispiele → Digital Out C++

```
1  #include "mbed.h"
2  int main() {
3      // Relais und Solid-State-Relais
4      DigitalOut relais(p27);
5      DigitalOut SSR(p21);
6
7      relais.write(1);           // oder auch   relais = 1;
8      SSR = 0;                  // entspricht SSR.write(0);
9
10     // Blink-LED
11     DigitalOut led(p30);
12
13     while(1) {
14         led = !led;           // entspricht led.write( !led.read() );
15         wait(0.5);
16     }
17 }
18
19
```

# Beispiele → Digital-In



# Beispiele → Digital In C++

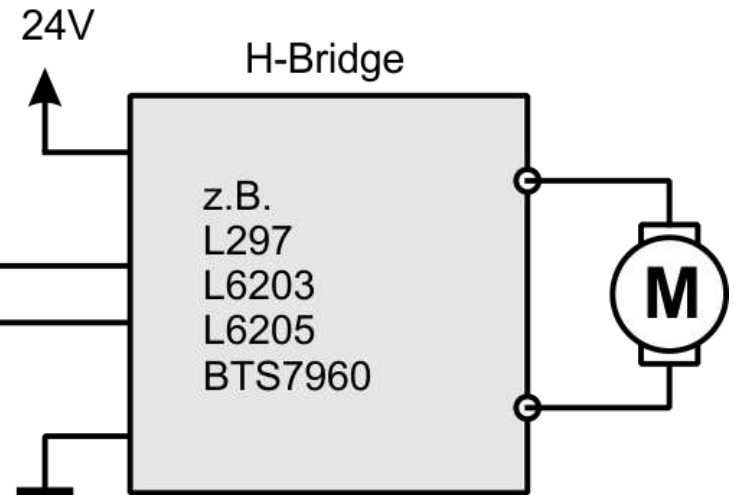
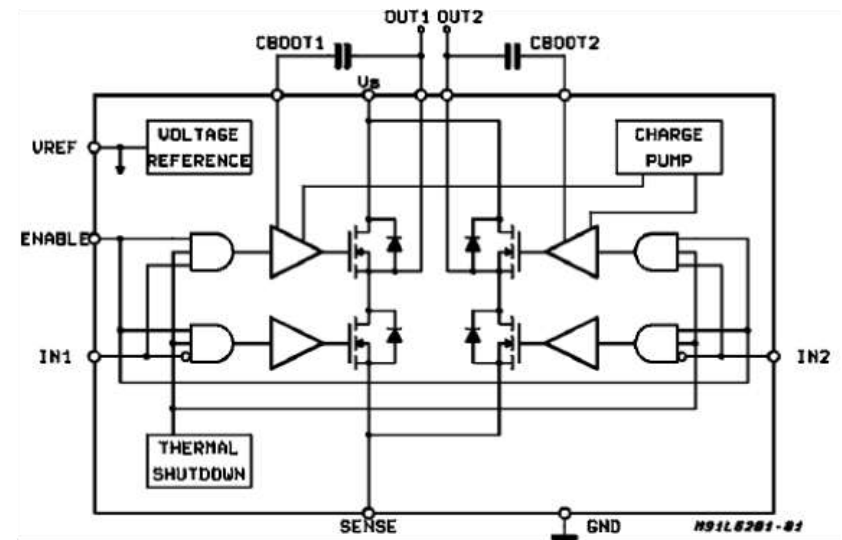
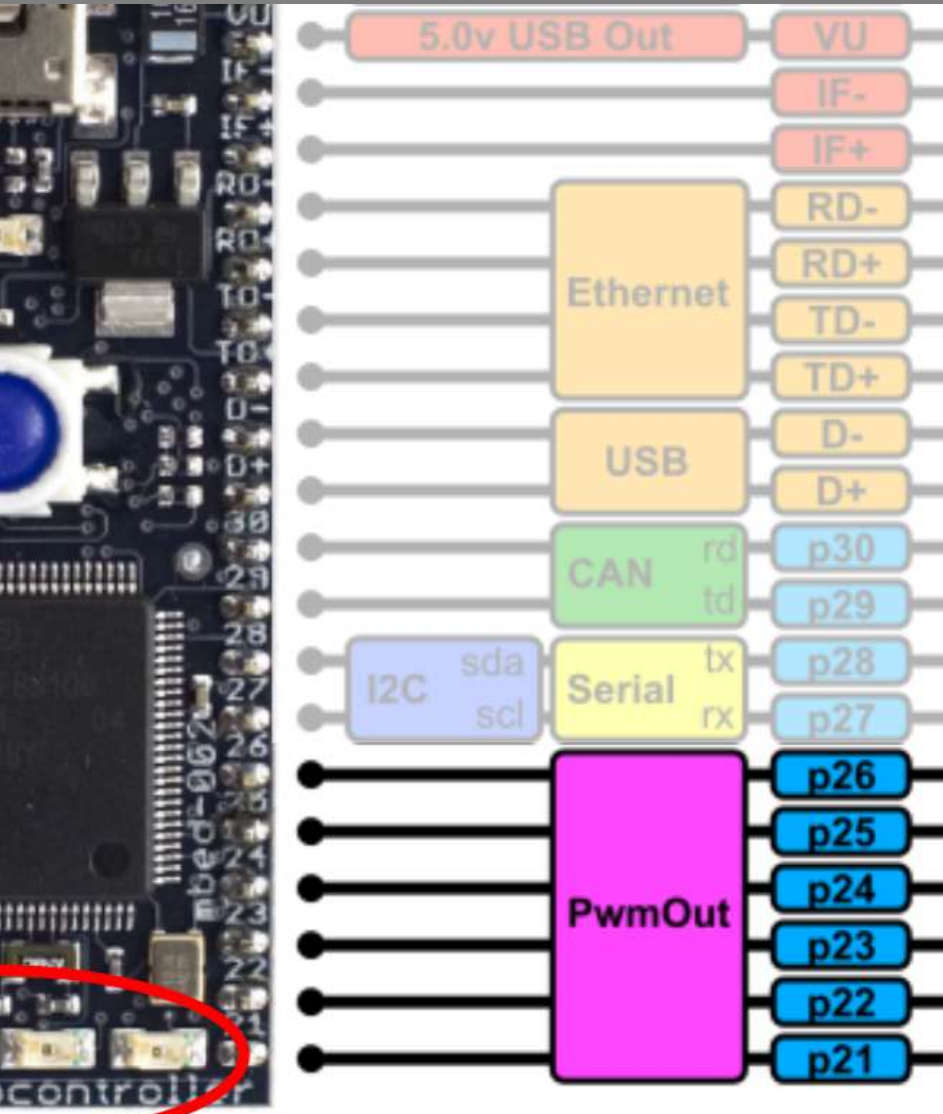
```
1  #include "mbed.h"
2  int main() {
3
4  // Taster
5      DigitalIn taster(p18);
6
7      taster.mode ( PullUp );
8
9      int i = taster;           // entspricht taster.read();
10
11     if ( taster == 0 )
12     {
13         // ...
14     }
15
16     funktion_abc ( taster );
17 }
18
19
```

# Beispiele → Digital In C++

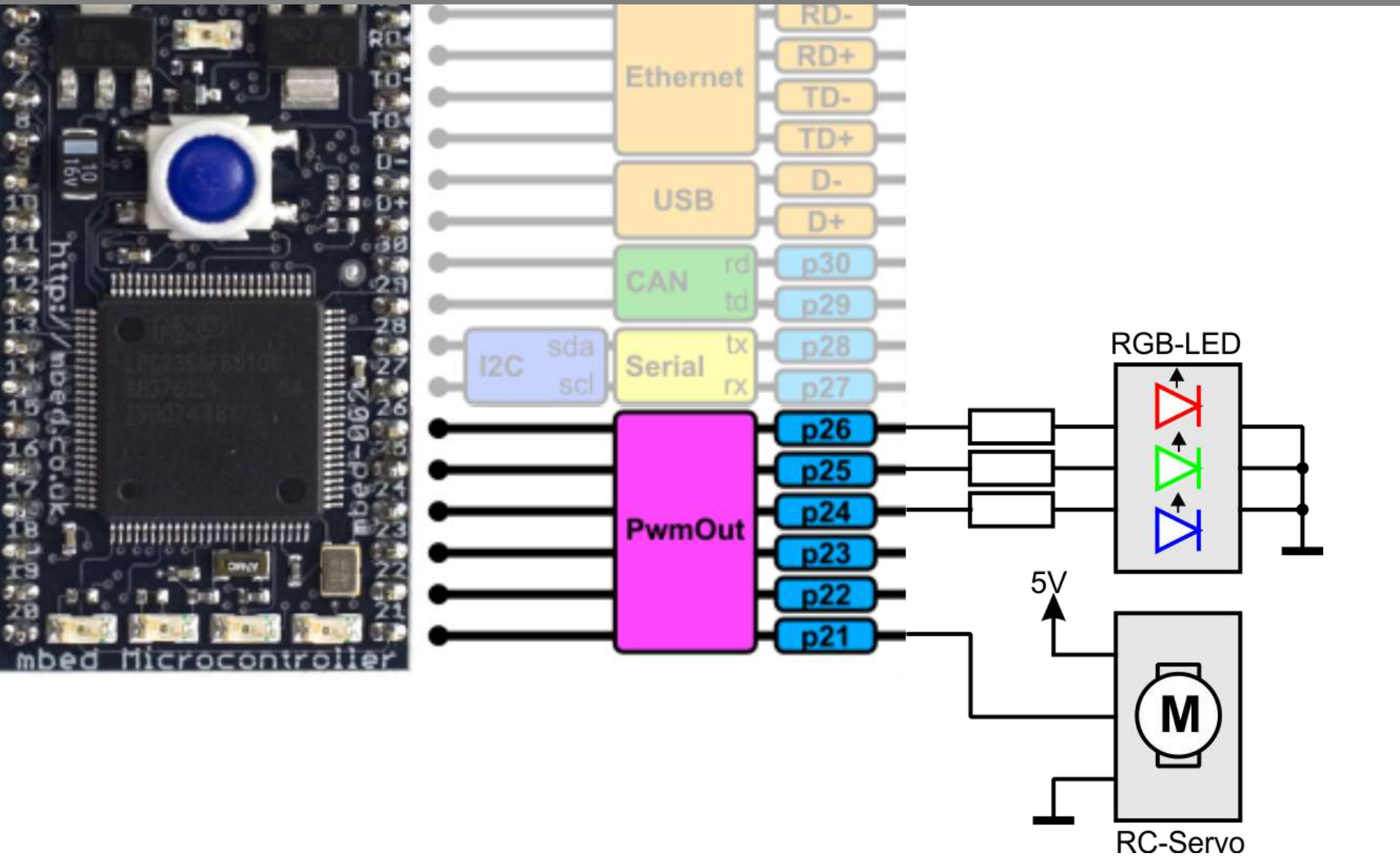
```
1  #include "mbed.h"
2
3  // Taster
4  InterruptIn taster(p26);
5
6  // ISR (Interrupt Service Routine)
7  void callback() {
8      // ...
9  }
10
11 int main() {
12
13     taster.mode ( PullUp );
14     taster.fall ( &callback );           // attach ISR
15
16     while(1) {} //Endlosschleife
17 }
18
19
```



# Beispiele → PWM (Pulsweiten-Modulation)



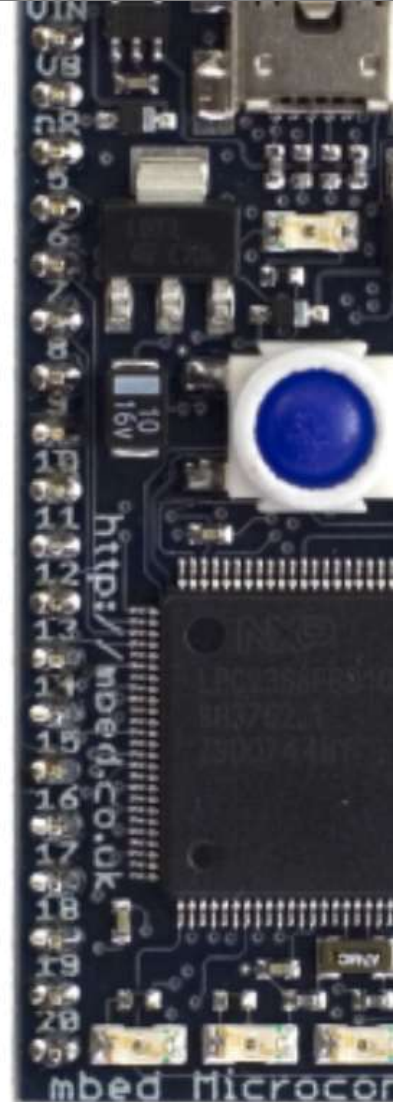
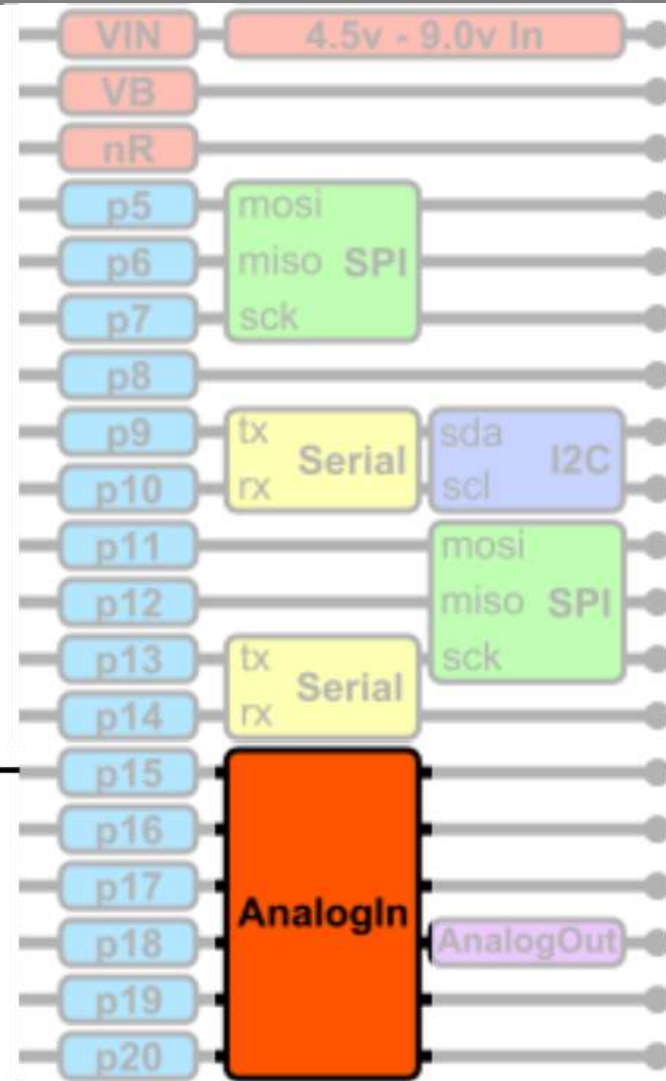
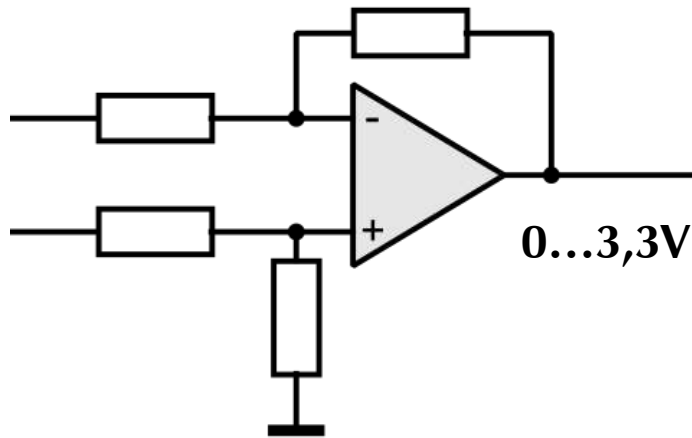
# Beispiele → PWM (Pulsweiten-Modulation)



# Beispiele → PWM C++

```
1  #include "mbed.h"
2  int main() {
3  // RGB-LED auf orange
4      PwmOut ledR(p26), ledG(p25), ledB(p24);
5
6      ledR = 1.0; // 100%           // == ledR.write(1.0);
7      ledG = 0.7; // 70%
8      ledB = 0.0; // 0%
9
10 // RC-Servo
11     PwmOut servoRC(p21);
12
13     servoRC.period_ms(20);       // == servoRC.period(0.020);
14
15     // RC-Servo in Mittelstellung
16     float value = 0.5;           // 0...1
17     servoRC.pulsewidth(0.001 + 0.001*value);
18 }
19
```

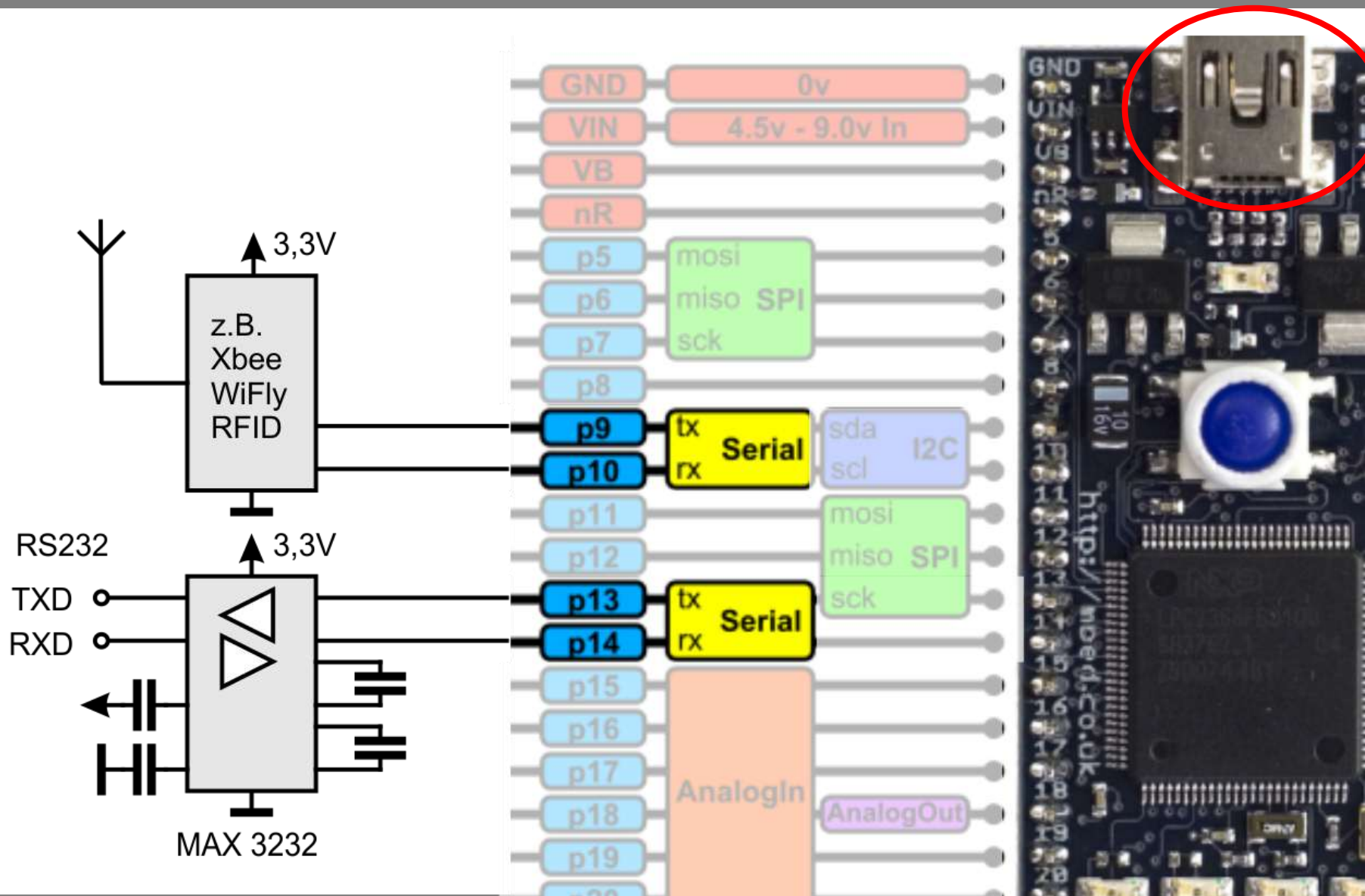
# Beispiele → Analog-In



# Beispiele → Analog In C++

```
1  #include "mbed.h"
2  int main() {
3  // Analog In
4      AnalogIn ain1(p18), ain2(p19);
5
6      float value = ain1.read();
7
8      PwmOut pwm(p21);
9      while (1) {
10         pwm = pow(ain1, 1.2f) + ain2;
11     }
12
13     unsigned short samples[1024];
14     for(int i=0; i<1024; i++) {
15         samples[i] = ain1.read_u16();
16         wait_ms(0.1);
17     }
18 }
19
```

# Beispiele → Serielle (RS232, V24)



# Beispiele → Serielle C++

```
1  #include "mbed.h"
2  int main() {
3  // Virtuelle Serielle
4      Serial pc(USBTX, USBRX); // tx, rx
5
6      pc.printf("Hello World!");
7
8  // Serielle zu Device
9      Serial sensor(p9, p10); // tx, rx
10
11     sensor.baud(19200);
12     sensor.format(8, Serial::Odd, 1);
13
14     sensor.putc('X');
15     while ( sensor.readable() ) {
16         int i = sensor.getc();
17     }
18 }
19
```

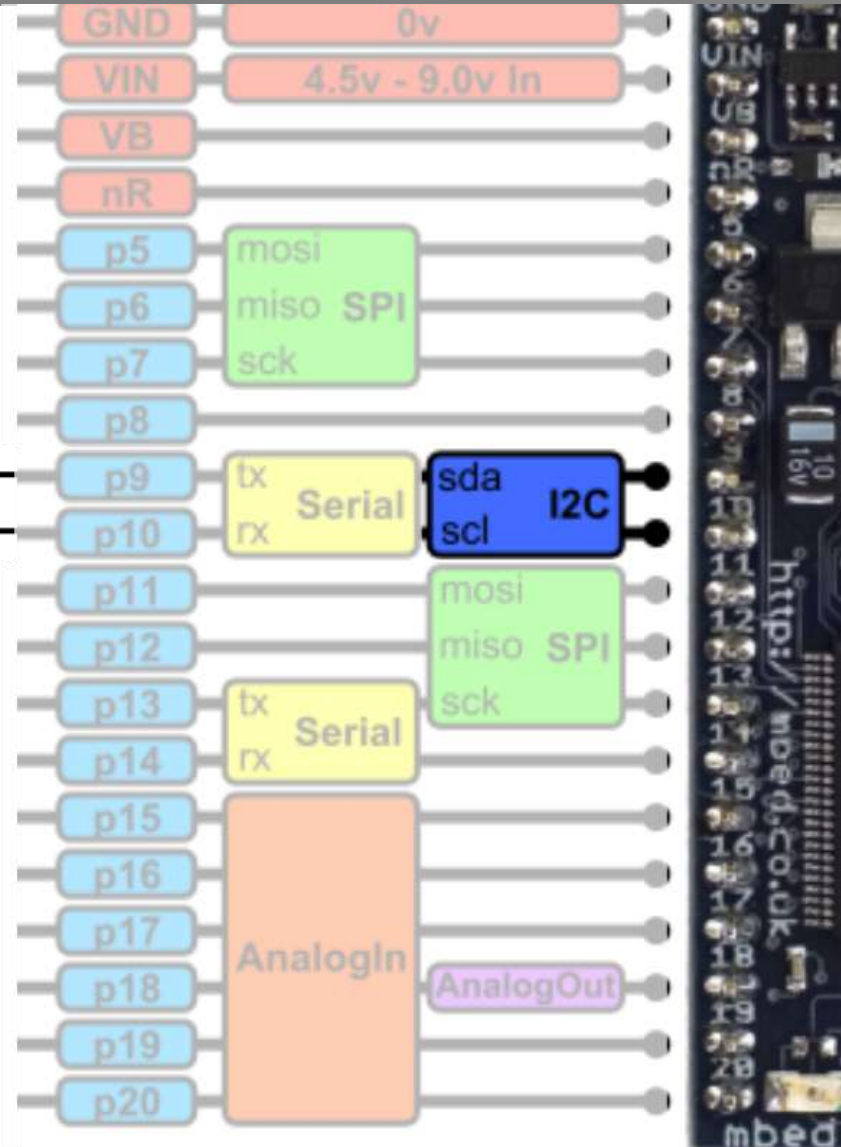
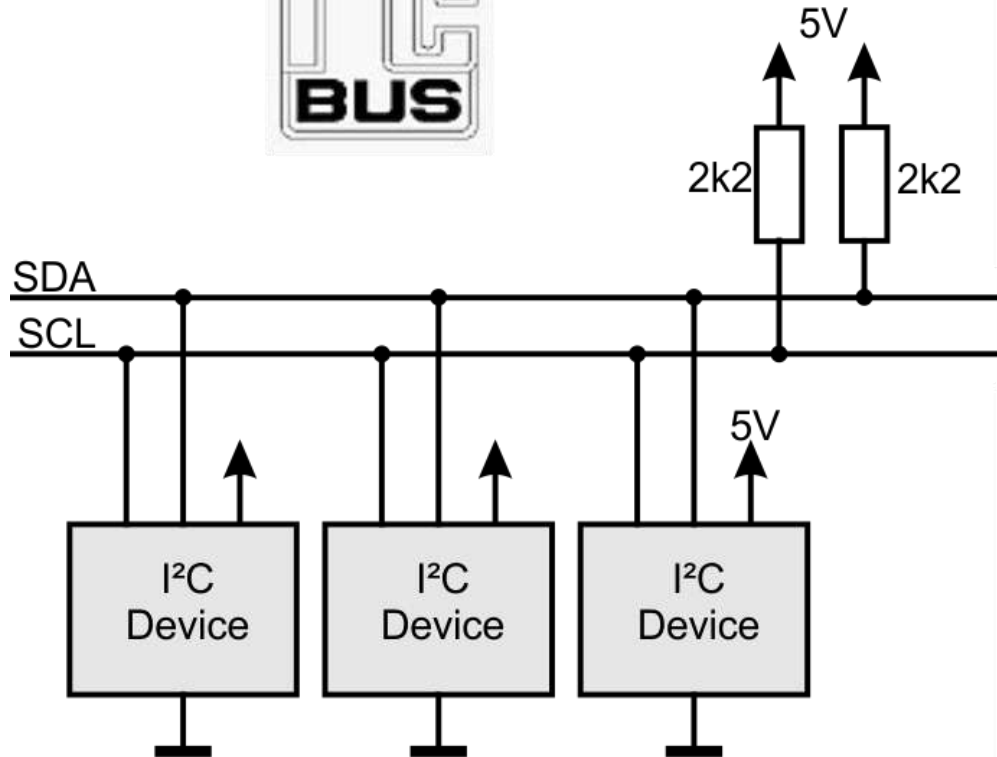
# Beispiele → Serielle C++

```
1  #include "mbed.h"
2
3  // Serielle zu Device
4  Serial sensor(p9, p10); // tx, rx
5
6  void callback() {
7      int i = sensor.getc();
8      // ...
9  }
10
11 int main() {
12
13     sensor.baud(19200);
14     sensor.attach(&callback);
15
16     sensor.putc('X');
17
18     while(1) {} //Endlosschleife
19 }
```



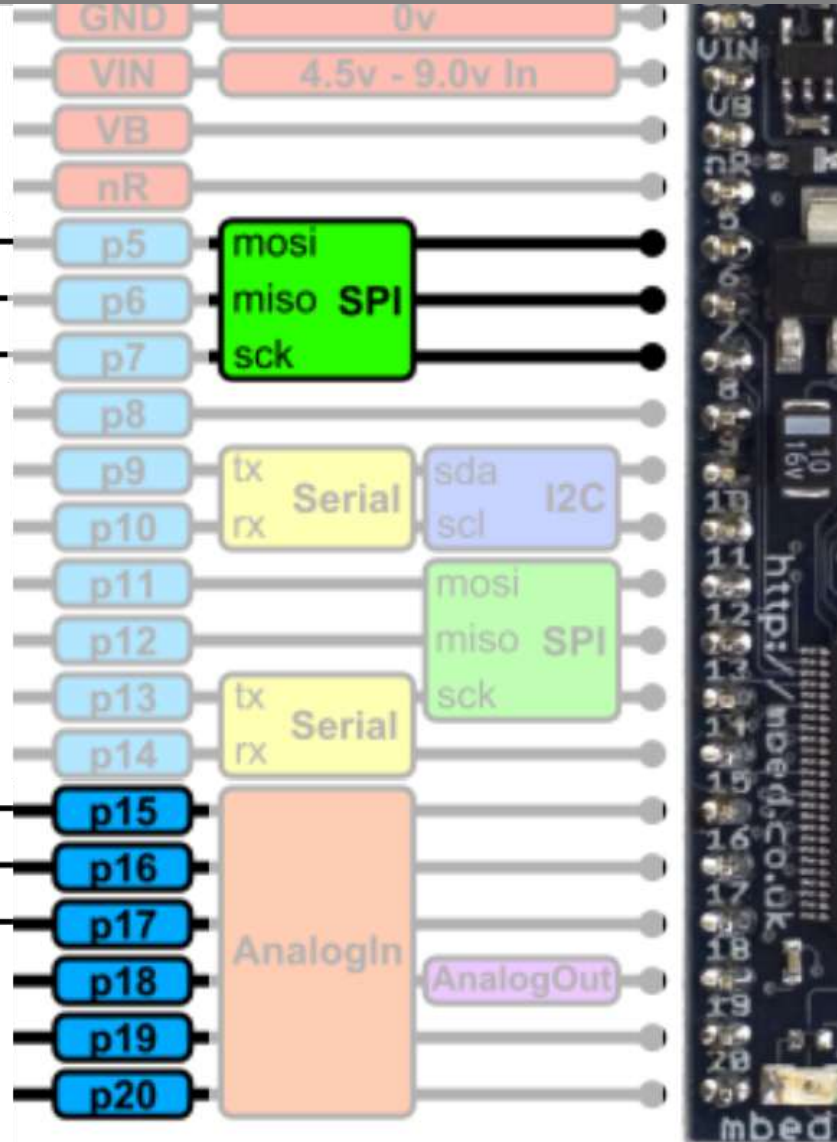
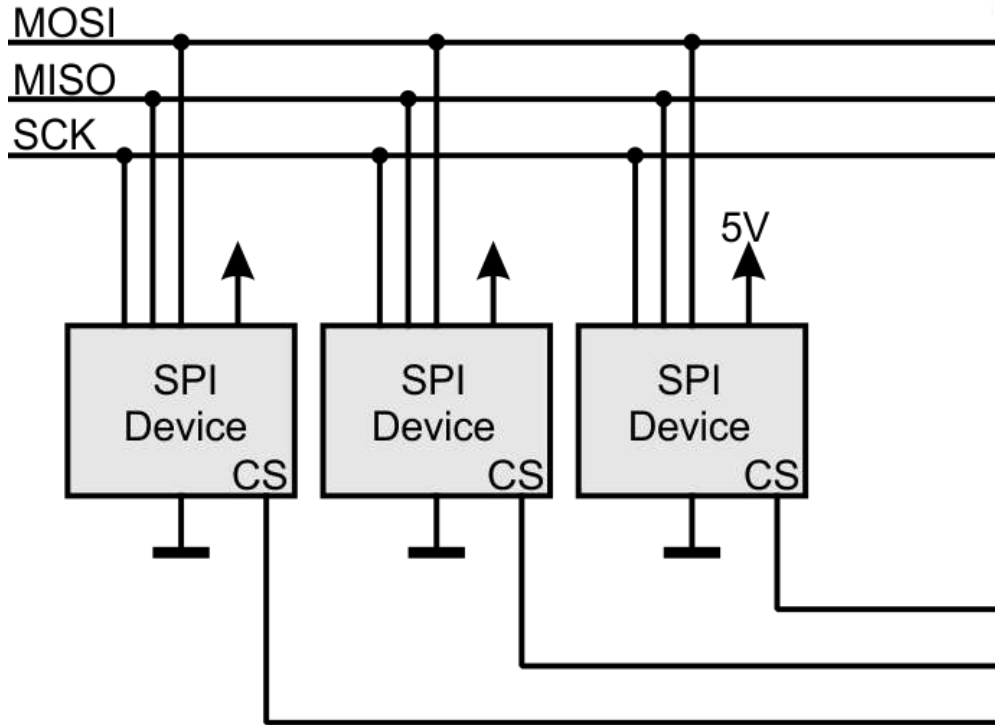
# Beispiele → I<sup>2</sup>C (IIC, I2C, Two-Wire)

(IIC = Inter-Integrated Circuit)



# Beispiele → SPI

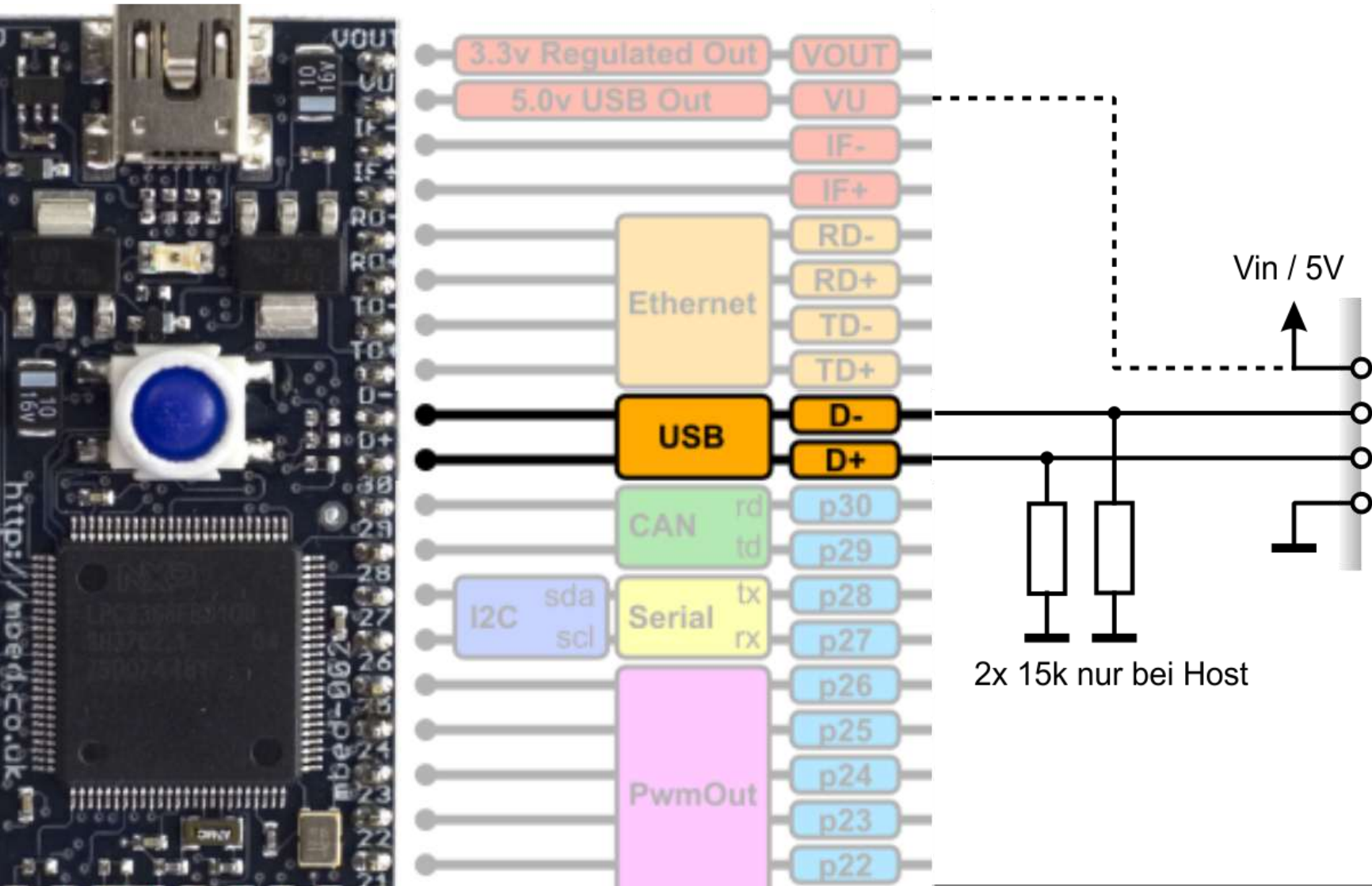
(SPI = Serial Peripheral Interface)



# Beispiele → I<sup>2</sup>C, SPI C++

```
1  #include "mbed.h"
2  int main() {
3  // SPI
4      SPI spi(p5, p6, p7);           // mosi, miso, sclk
5
6      spi.format(8,3);               // 8 Bits (4-16), Mode 3
7      spi.frequency(1000000);       // default 1MHz
8
9      int ret = spi.write(0x12);
10
11 // I2C (Two-Wire)
12     I2C i2c(p9, p10);              // sda, scl
13
14     const int addr = 0x70;         // define the I2C Address
15     char cmd[2] = { 0x00, 0x51 };
16
17     i2c.write(addr, cmd, 2);
18     i2c.read(addr, cmd, 2);
19 }
```

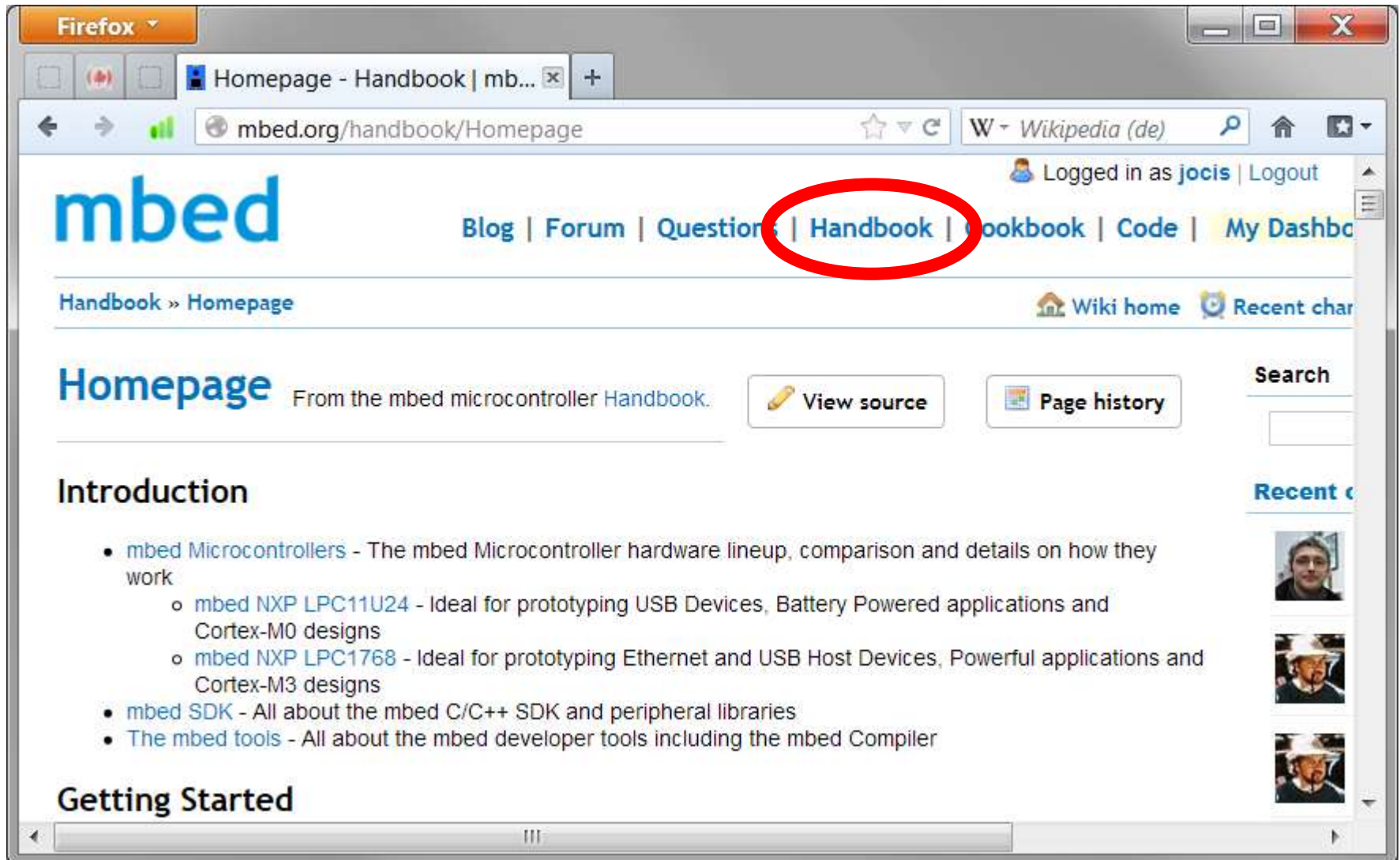
# Beispiele → USB Device/Host



# Beispiele → USB-Mouse C++

```
1  #include "mbed.h"
2  #include "USBMouse.h"
3
4  USBMouse mouse(ABS_MOUSE);
5
6  int main() {
7      int16_t x, y;
8      float radius = 5000;
9      float angle = 0;
10
11     while (1) {
12         x = 10000 + cos(angle*3.14f/180.0f)*radius;
13         y = 10000 + sin(angle*3.14f/180.0f)*radius;
14         mouse.move(x, y);
15         angle += 3.0f;
16         wait(0.01);
17     }
18 }
19
```

# Beispiele → Handbook

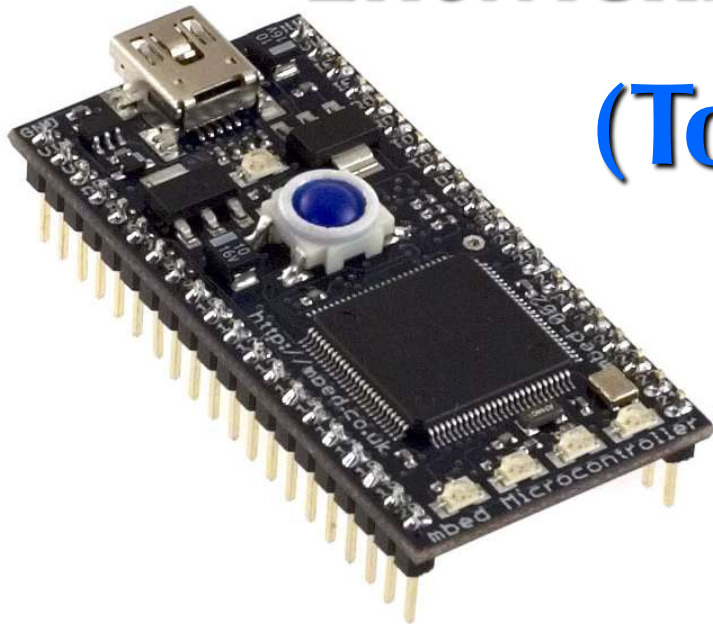


The screenshot shows a Firefox browser window displaying the mbed.org Handbook homepage. The address bar shows the URL `mbed.org/handbook/Homepage`. The page features a navigation menu with links for [Blog](#), [Forum](#), [Questions](#), [Handbook](#) (circled in red), [Cookbook](#), [Code](#), and [My Dashboard](#). The user is logged in as `jocis`. The main content area includes a search bar, a 'Recent changes' section with user avatars, and a list of articles under the 'Introduction' section:

- [mbed Microcontrollers](#) - The mbed Microcontroller hardware lineup, comparison and details on how they work
  - [mbed NXP LPC11U24](#) - Ideal for prototyping USB Devices, Battery Powered applications and Cortex-M0 designs
  - [mbed NXP LPC1768](#) - Ideal for prototyping Ethernet and USB Host Devices, Powerful applications and Cortex-M3 designs
- [mbed SDK](#) - All about the mbed C/C++ SDK and peripheral libraries
- [The mbed tools](#) - All about the mbed developer tools including the mbed Compiler

The 'Getting Started' section is partially visible at the bottom of the page.

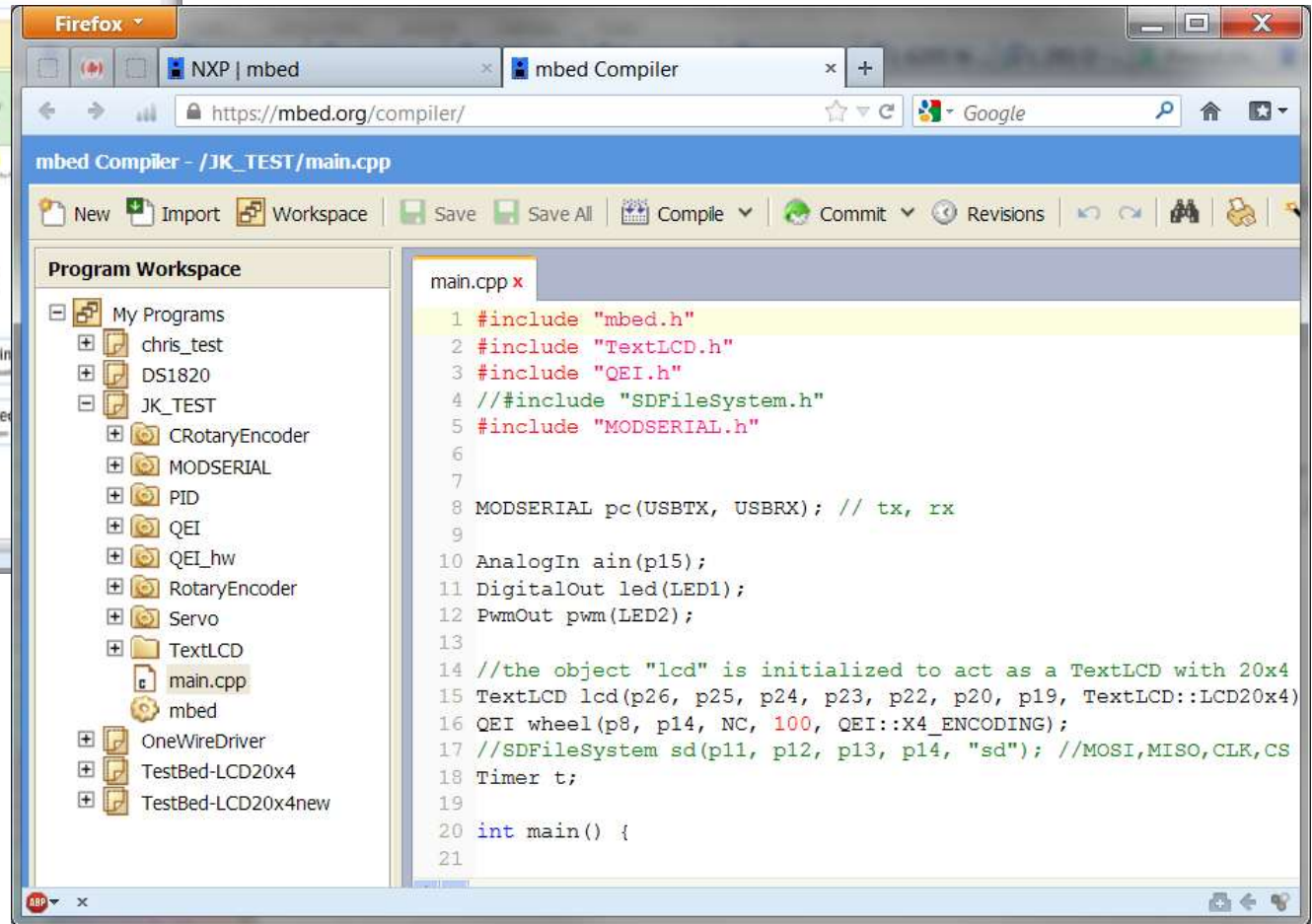
## Entwicklungsumgebung (Tool-Chain)



# IDE → Online

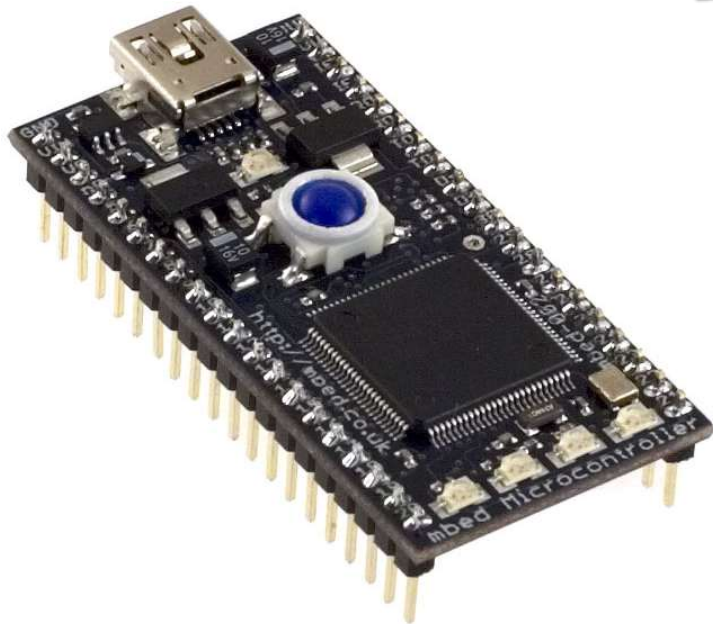
(IDE = Integrated Development Environment)

→ LIVE





## Libraries



# Libs → Selber machen...

## main.cpp (vorher)

```
1  #include "mbed.h"
2  int main() {
3
4  // RC-Servo
5      PwmOut servoRC(p21);
6
7      // RC-Servo Init
8      servoRC.period_ms(20);
9
10     // RC-Servo in Mittelstellung
11     float value = 0.5;
12     servoRC.pulsewidth(0.001 + 0.001*value);
13
14     // RC-Servo nach links
15     value = 0.1;
16     servoRC.pulsewidth(0.001 + 0.001*value);
17 }
18
19
```

# Libs → Selber machen...

## ServoRC.h

```
1  #include "mbed.h"
2
3  /** Interface to control a RC-servo ... */
4  class ServoRC {
5  protected:
6      PwmOut _pwm;           // Class Member
7
8  public:
9      /** Create an instance of ... */
10     ServoRC(PinName pwmpin); // Constructor
11     ~ServoRC()              // Destructor
12         { _pwm=0; }
13
14     void write ( float value ); // Method
15
16     float operator= ( float value ) // Assign-Operator
17         { write(value); return value; }
18 }
19
```

# Libs → Selber machen...

## ServoRC.cpp

```
1  #include <ServoRC.h>
2
3  // Constructor
4  ServoRC::ServoRC(PinName pwmpin) : _pwm(pwmpin)
5  {
6      _pwm.period_ms(20);
7  }
8
9
10 // Method write()
11 void ServoRC::write(float value)
12 {
13     if ( value<0.0 ) value = 0.0;
14     if ( value>1.0 ) value = 1.0;
15     _pwm.pulsewidth(0.001 + 0.001*value);
16 }
17
18
19
```

# Libs → Selber machen...

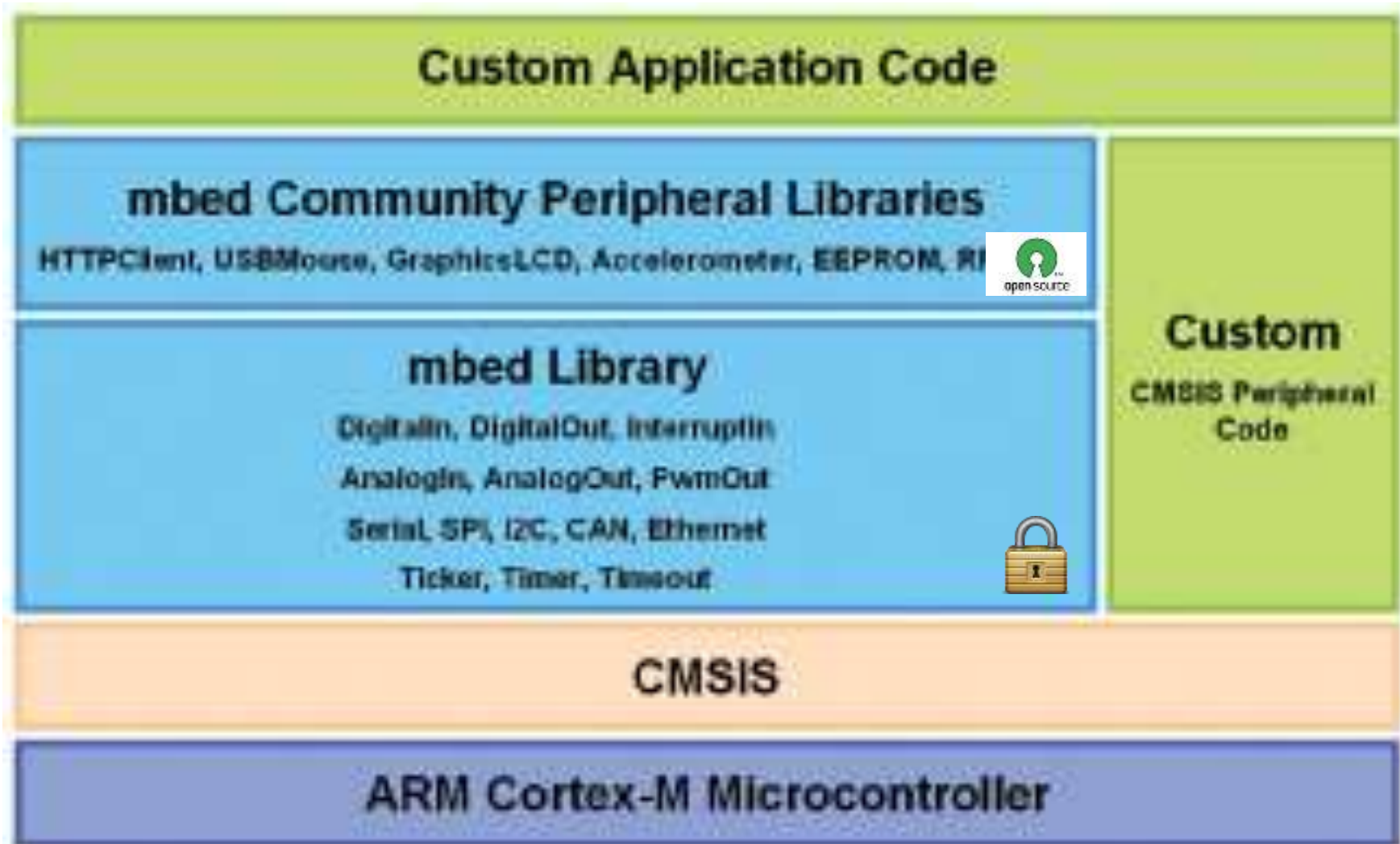
## main.cpp (nachher)

```
1  #include "mbed.h"
2
3  #include <ServoRC.h>
4
5  ServoRC myServo(p21);
6
7  int main() {
8
9      // RC-Servo in Mittelstellung
10     myServo = 0.5;
11
12     // RC-Servo nach links
13     myServo = 0.1;
14 }
15
16
17
18
19
```

# Libs → Selber machen...

The image shows a Firefox browser window displaying the mbed.org website. The main window shows the 'Servo' class reference page for user Simon Ford. The page includes a navigation menu, a user profile picture, and a description of the class: 'A class to control a model R/C servo, using a PwmOut'. Below the description are links for 'Home', 'History', 'Graph', 'API Documentation', and 'Wiki'. A smaller, semi-transparent window in the foreground shows the 'Constructor & Destructor Documentation' for the Servo class. It includes the constructor signature: `Servo ( PinName pin )` and a description: 'Create a servo object connected to the specified PwmOut'. It also lists the parameter: 'pin PwmOut pin to connect to' and the definition location: 'Definition at line 37 of file Servo.cpp'. Below this, the 'Member Function Documentation' section shows the signature for the `calibrate` function: `void calibrate ( float range = 0.0005, float degrees = 45.0 )`.

# Libs → Architektur



# Libs → Community

The image displays two screenshots of the mbed.org website in a Firefox browser window. The top screenshot shows the homepage navigation menu, where the 'Cookbook' link is circled in red. The bottom screenshot shows a list of libraries and sections, including 'Motors and Actuators'.

**Firefox** | Homepage - Cookbook | mb... x +

mbed.org/cookbook/Homepage

Logged in as **jocis** | Logout

**mbed** | Blog | Forum | Questions | Handbook | **Cookbook** | Code | My Dashbo

Cookbook » Homepage

## Homepage

From the mbed microcontroller Cookbook.

**mbed Cookbook Community Wiki**

Welcome to the mbed Cookbook, a wiki for publishing user-contributed community "Handbook" for all the useful component and library building blocks for your prototypes.

Please feel free contribute component libraries, building blocks and see ways to improve existing resources with extra examples, explanations.

## Introduction and Help

- [About the Cookbook](#) - What it is for, how to use it, and how you can contribute.

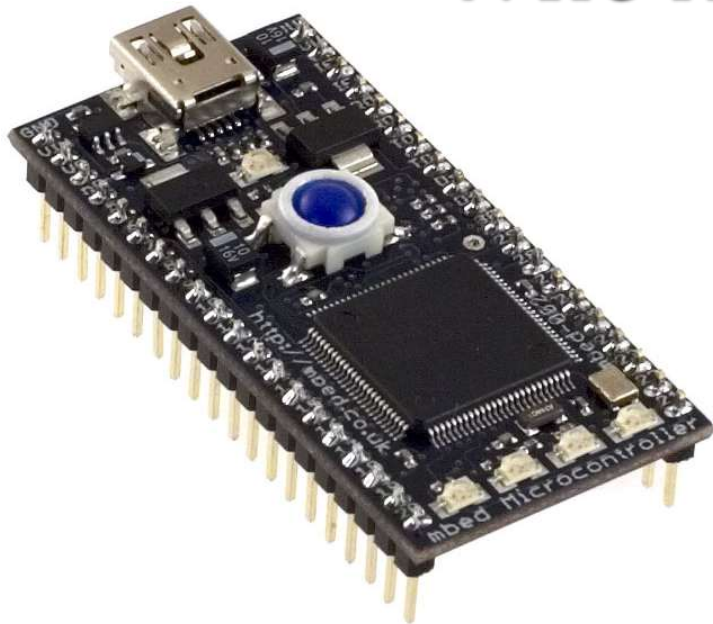
**Motors and Actuators**

- [XBee](#) - Simple zigbee modules
- [RN-42-Bluetooth](#) - Simple serial bluetooth module
- [USBBluetoothHost](#) - Using a USB dongle to connect via bluetooth
- [WiflyInterface](#) - Roving Networks wifi module
- [Xbee-Pro](#) - Library and example for using the Xbee Pro
- [GainSpan Wi-Fi GS1011](#) - ultra low power 802.11b wireless module
- [XBee-mbed](#) - XBee API mode, Series 1 (802.15.4) and Series 2

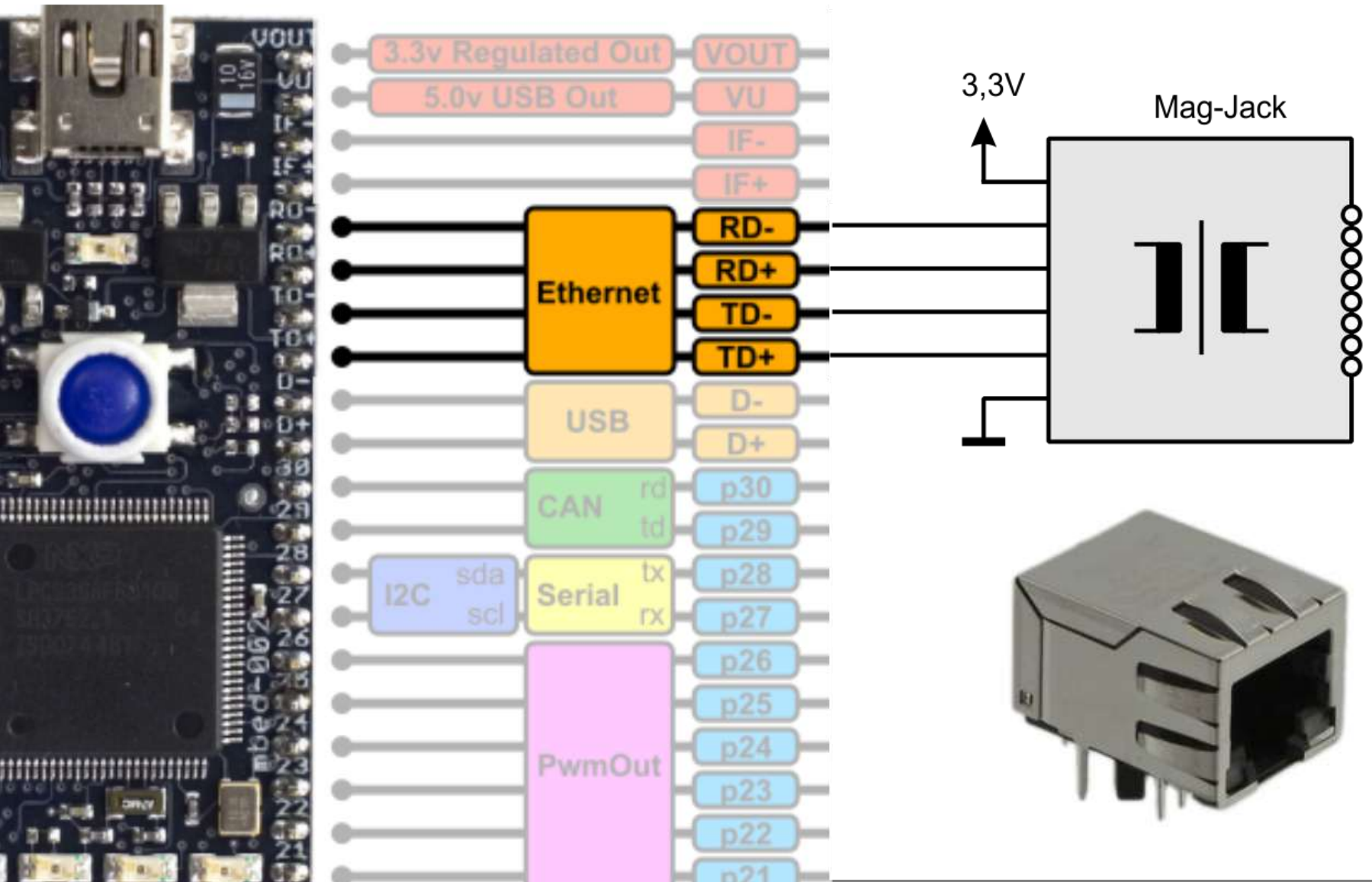
**Sensors**



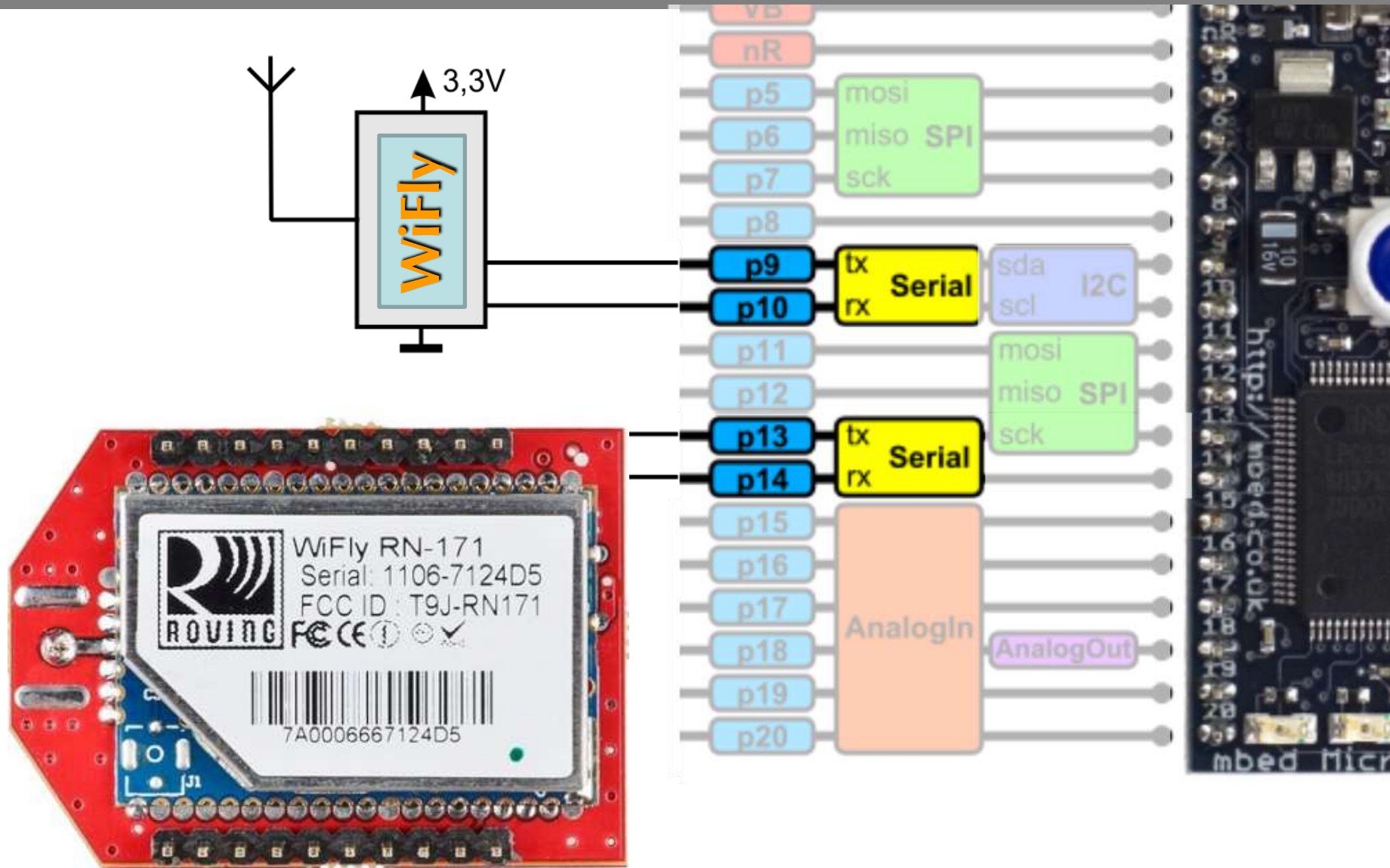
## Mit mbed in Netz



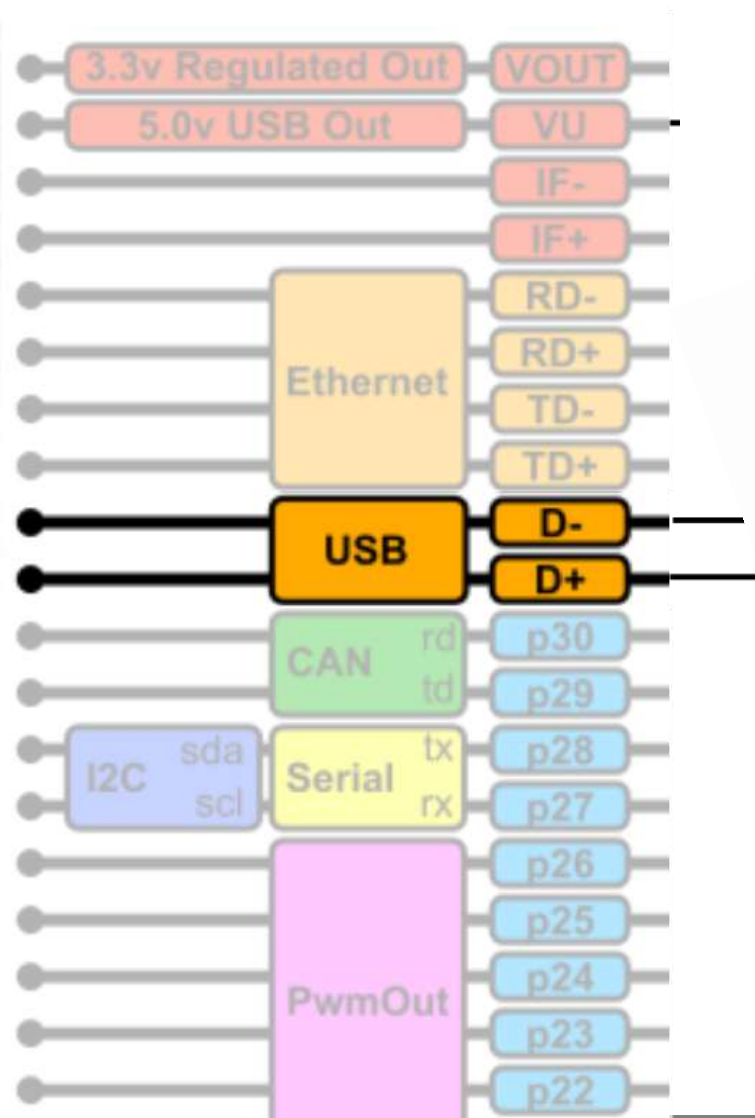
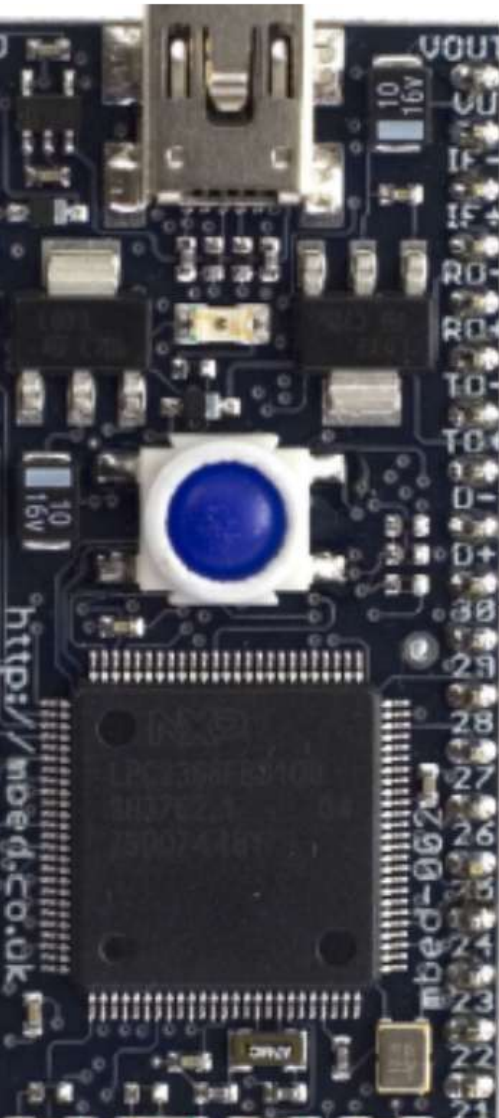
# Web → Ethernet (10/100Mbit)



# Web → WLAN 802.11b/g (WiFly)



# Web → UMTS (SurfStick)



# Web → Ethernet C++

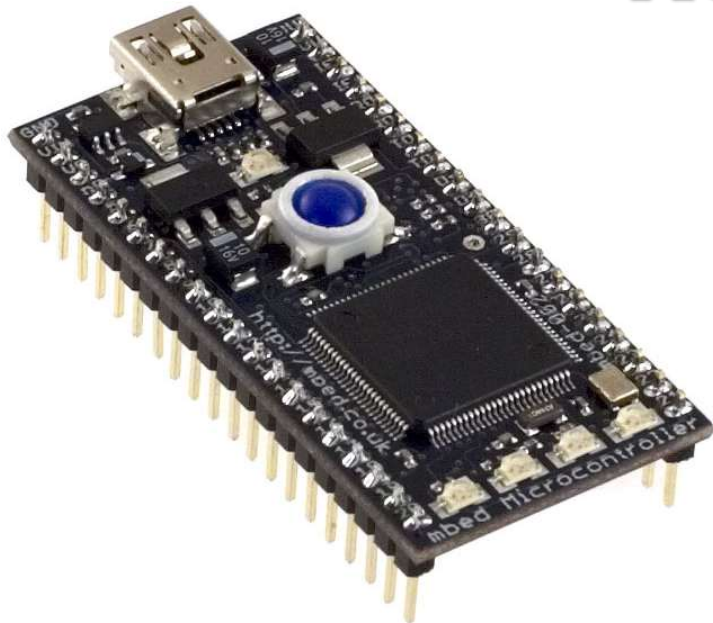
```
1  #include "mbed.h",
2  #include "EthernetInterface.h"
3  int main() {
4      EthernetInterface eth;
5      eth.init();                //Use DHCP
6      eth.connect();
7
8      TCPSocketConnection sock;
9      sock.connect(„www.franken.de“, 80);
10     char http_cmd[] = "GET /index.htm HTTP/1.0\n\n";
11     sock.send_all(http_cmd, sizeof(http_cmd)-1);
12
13     char buffer[300];    int ret;
14     do {
15         ret = sock.receive(buffer, sizeof(buffer));
16         // ... buffer[]
17     } while (ret>0);
18     sock.close();    eth.disconnect();
19 }
```

# Web → Ethernet NTP C++

```
1  #include "mbed.h",
2  #include "EthernetInterface.h"
3  #include „NTPClient.h“
4  int main() {
5      EthernetInterface eth;
6      eth.init();                //Use DHCP
7      eth.connect();
8
9      NTPClient ntp;
10
11     if ( ntp.setTime("0.pool.ntp.org") != 0 ) {
12         // Fehler ...
13     }
14
15     eth.disconnect();
16 }
17
18
19
```



# Referenzen



# mbed ☹️

## Vachteil

**Preis** 56...60...70 €

**IDE** nur Online

**Code** im Netz (mbed.org)

**Anleitungen** *only in english*

...

**Weniger Features als direkt in CMSIS**

...

**Bezugsquellen** nur wenige...



# Bezugsquellen mbed

## Für Privatkunden



[www.watterott.com](http://www.watterott.com)



[www.amazon.de](http://www.amazon.de)

## Nur gewerbliche Kunden

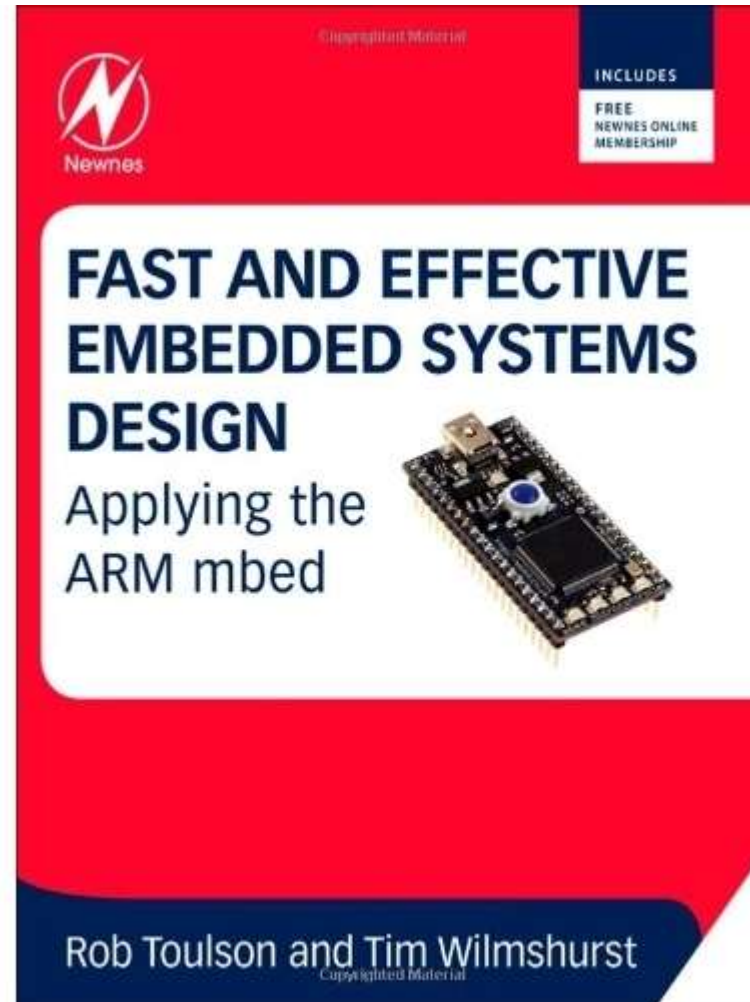


[www.elmicro.com](http://www.elmicro.com)

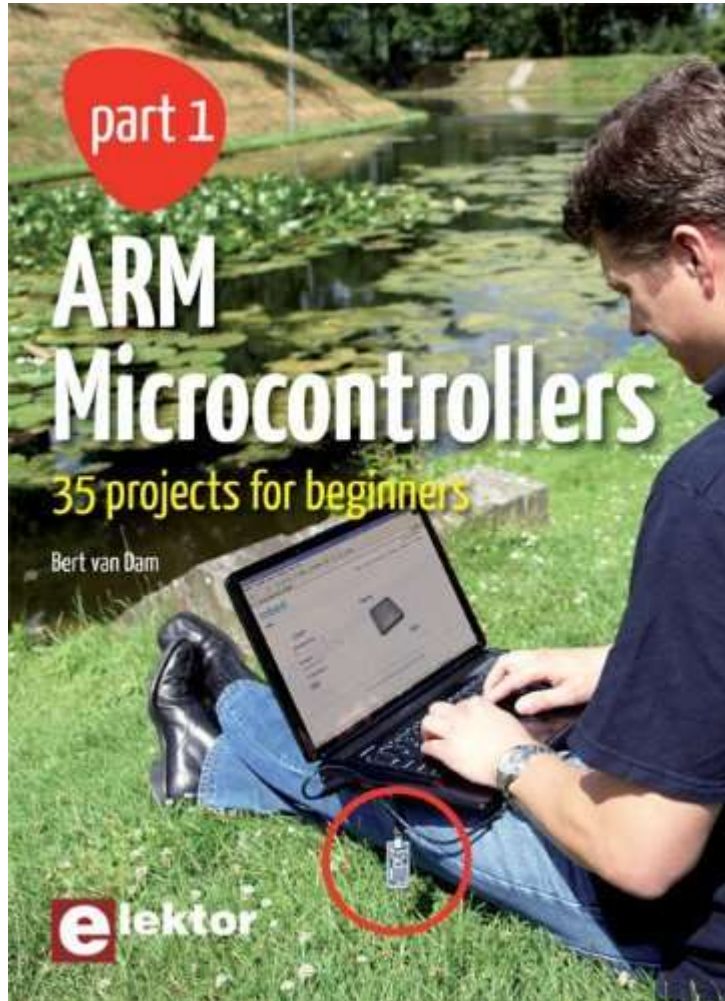


[www.farnell.de](http://www.farnell.de)

# Bücher



# Bücher

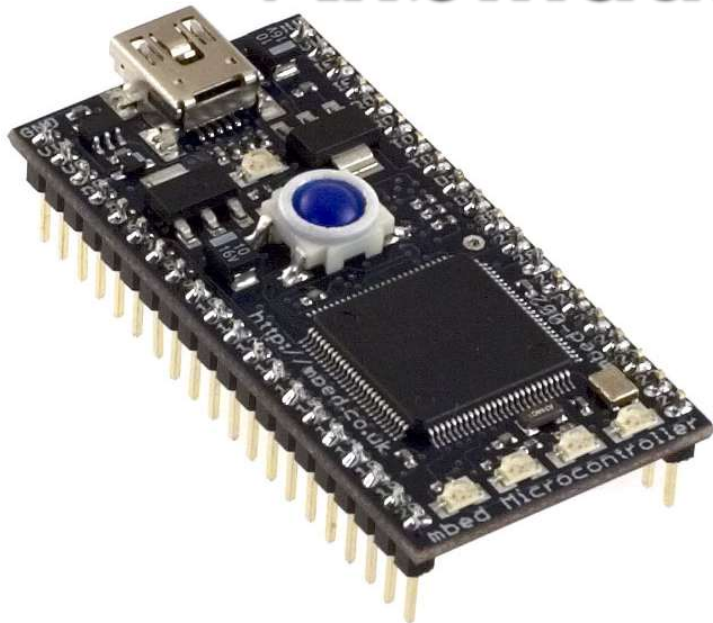




**Alle Logos, Marken und Namen sind Eigentum der jeweiligen Firmen**

# Sensoren

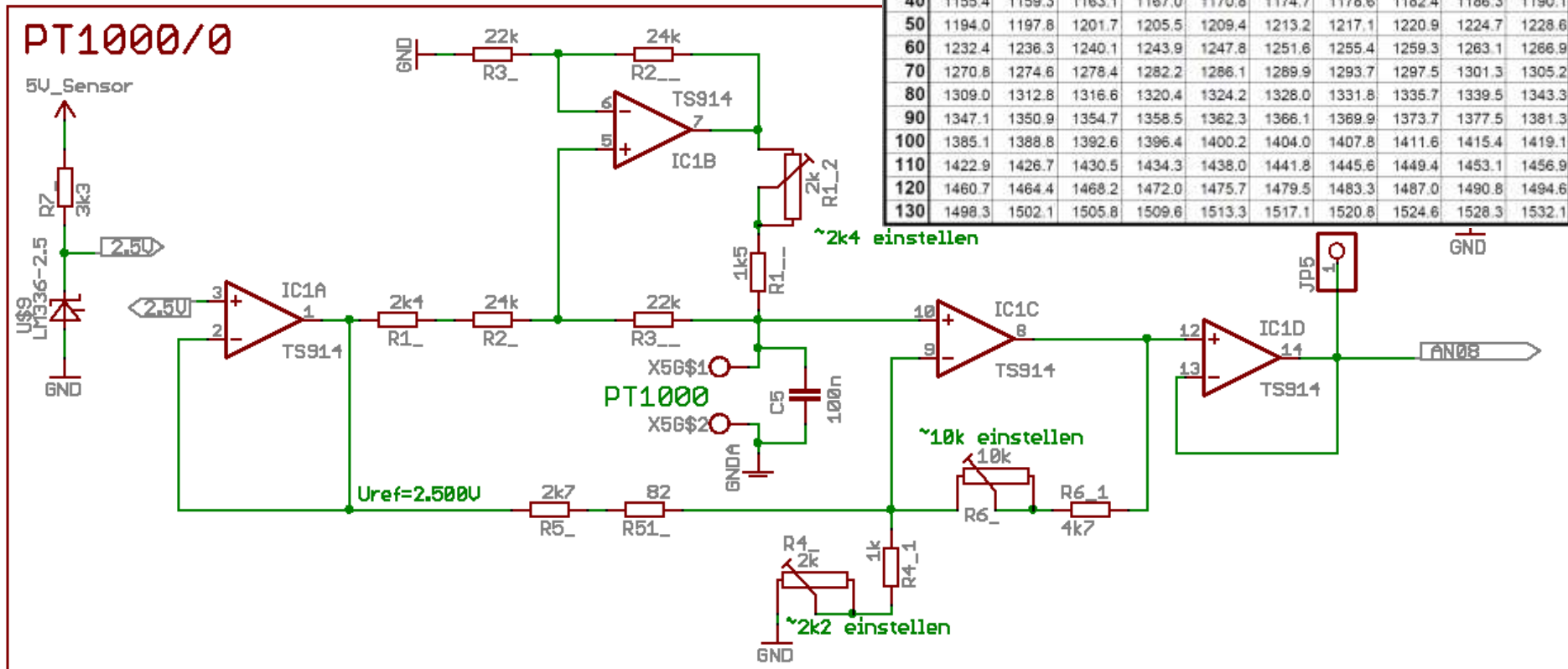
## Anbindung von Sensoren



# Sensoren → Temperatur analog

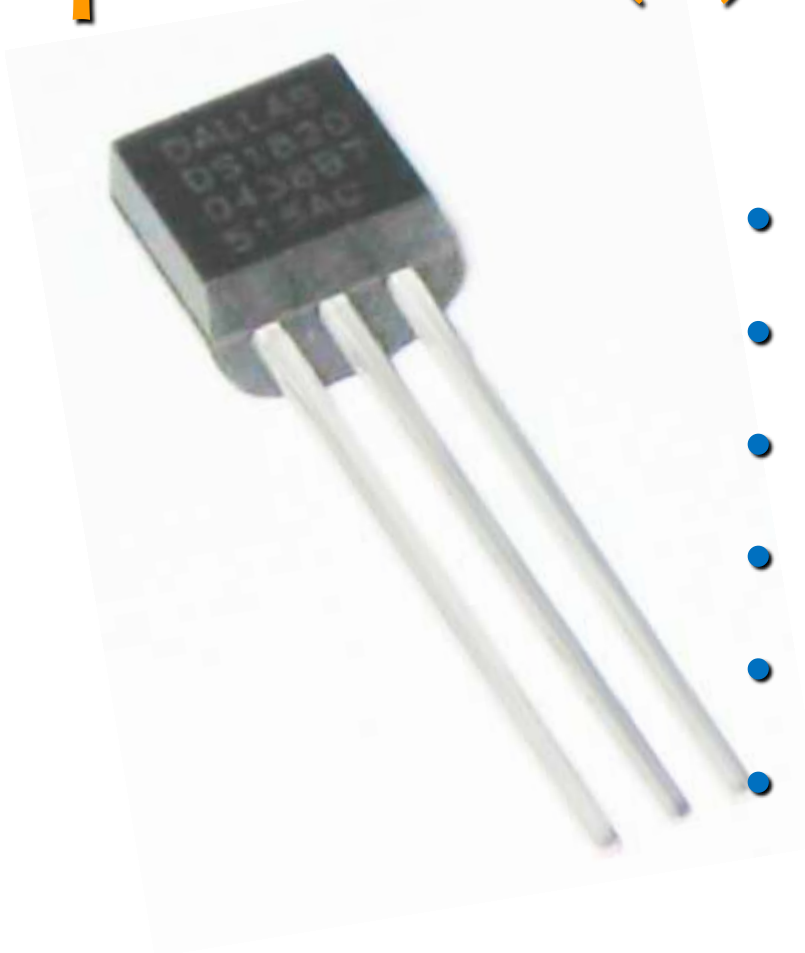
## Beispiel: PT1000

°C	0.0	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0
-50	803.1	807.0	811.0	815.0	818.9	822.9	826.9	830.8	834.8	838.7
-40	842.7	846.7	850.6	854.6	858.5	862.5	866.4	870.4	874.3	878.3
-30	882.2	886.2	890.1	894.0	898.0	901.9	905.9	909.8	913.7	917.7
-20	921.6	925.5	929.5	933.4	937.3	941.2	945.2	949.1	953.0	956.9
-10	960.9	964.8	968.7	972.6	976.5	980.4	984.4	988.3	992.2	996.1
0	1000.0	1003.9	1007.8	1011.7	1015.6	1019.5	1023.4	1027.3	1031.2	1035.1
10	1039.0	1042.9	1046.8	1050.7	1054.6	1058.5	1062.4	1066.3	1070.2	1074.0
20	1077.9	1081.8	1085.7	1089.6	1093.5	1097.3	1101.2	1105.1	1109.0	1112.9
30	1116.7	1120.6	1124.5	1128.3	1132.2	1136.1	1140.0	1143.8	1147.7	1151.5
40	1155.4	1159.3	1163.1	1167.0	1170.8	1174.7	1178.6	1182.4	1186.3	1190.1
50	1194.0	1197.8	1201.7	1205.5	1209.4	1213.2	1217.1	1220.9	1224.7	1228.6
60	1232.4	1236.3	1240.1	1243.9	1247.8	1251.6	1255.4	1259.3	1263.1	1266.9
70	1270.8	1274.6	1278.4	1282.2	1286.1	1289.9	1293.7	1297.5	1301.3	1305.2
80	1309.0	1312.8	1316.6	1320.4	1324.2	1328.0	1331.8	1335.7	1339.5	1343.3
90	1347.1	1350.9	1354.7	1358.5	1362.3	1366.1	1369.9	1373.7	1377.5	1381.3
100	1385.1	1388.8	1392.6	1396.4	1400.2	1404.0	1407.8	1411.6	1415.4	1419.1
110	1422.9	1426.7	1430.5	1434.3	1438.0	1441.8	1445.6	1449.4	1453.1	1456.9
120	1460.7	1464.4	1468.2	1472.0	1475.7	1479.5	1483.3	1487.0	1490.8	1494.6
130	1498.3	1502.1	1505.8	1509.6	1513.3	1517.1	1520.8	1524.6	1528.3	1532.1



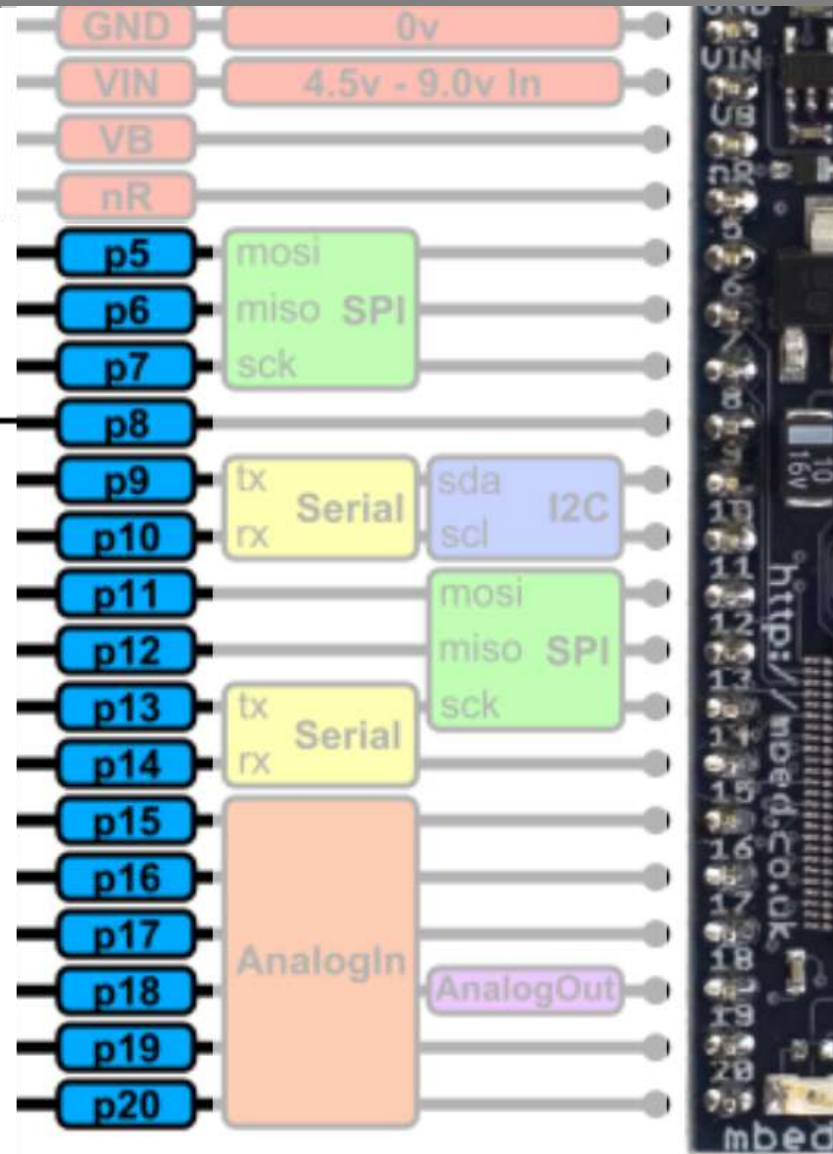
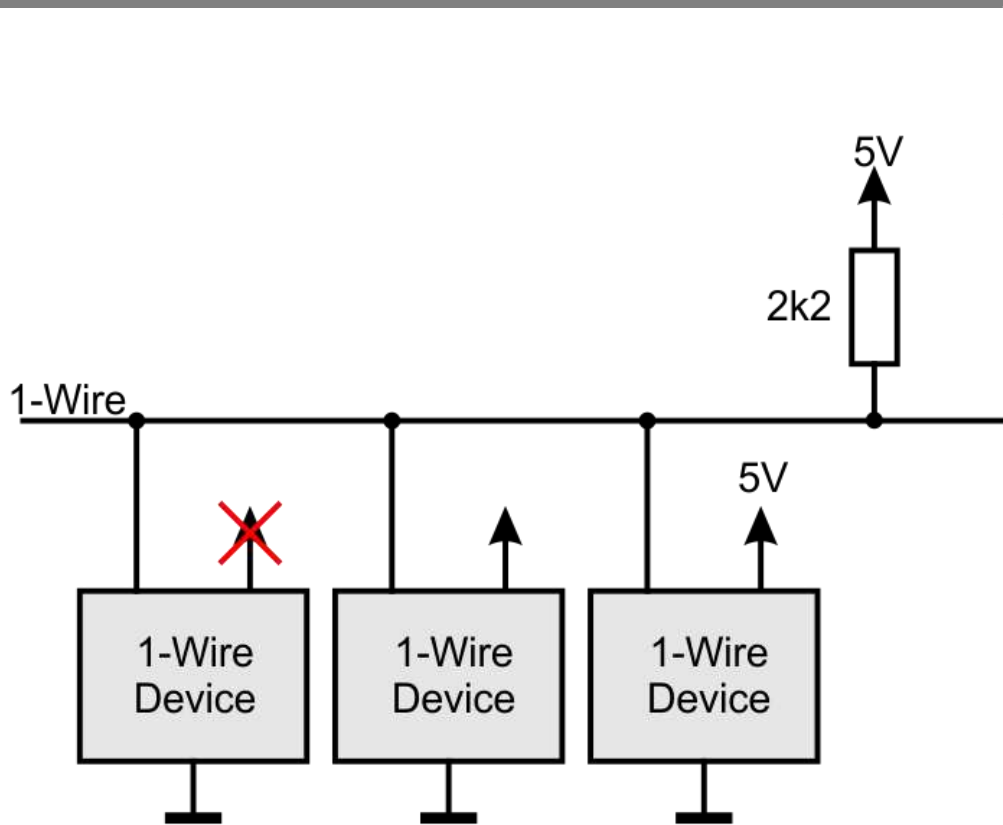
# Sensoren → Temperatur digital

## Beispiel: DS18(B)20



- **1-Wire-Bus**
- **-55...125°C**
- **Genauigkeit  $\pm 0,5^\circ\text{C}$**
- **3...5,5V oder pars.**
- **Eindeutige ID**
- **2,15 €** (Reichelt.de)

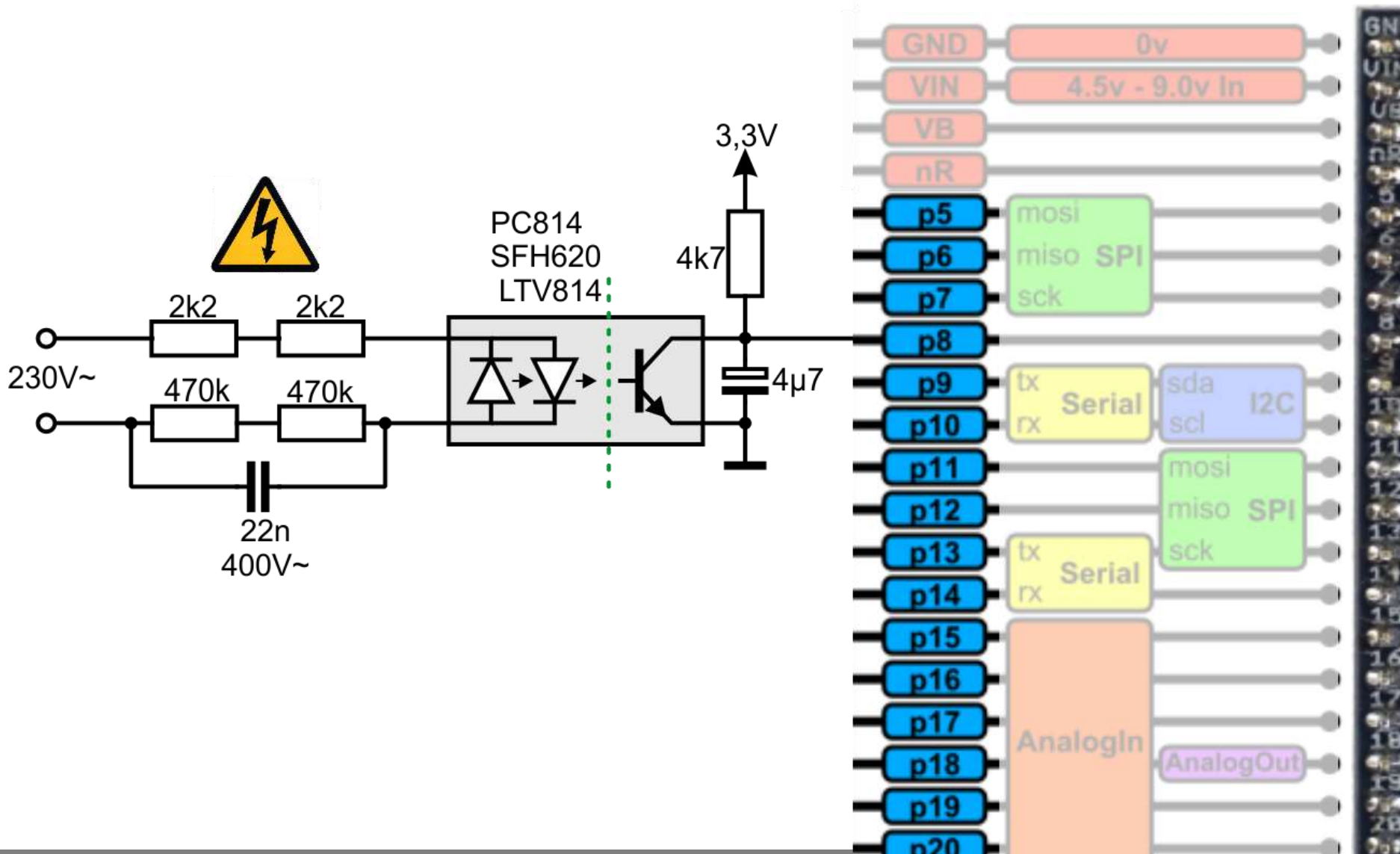
# Sensoren → 1-Wire



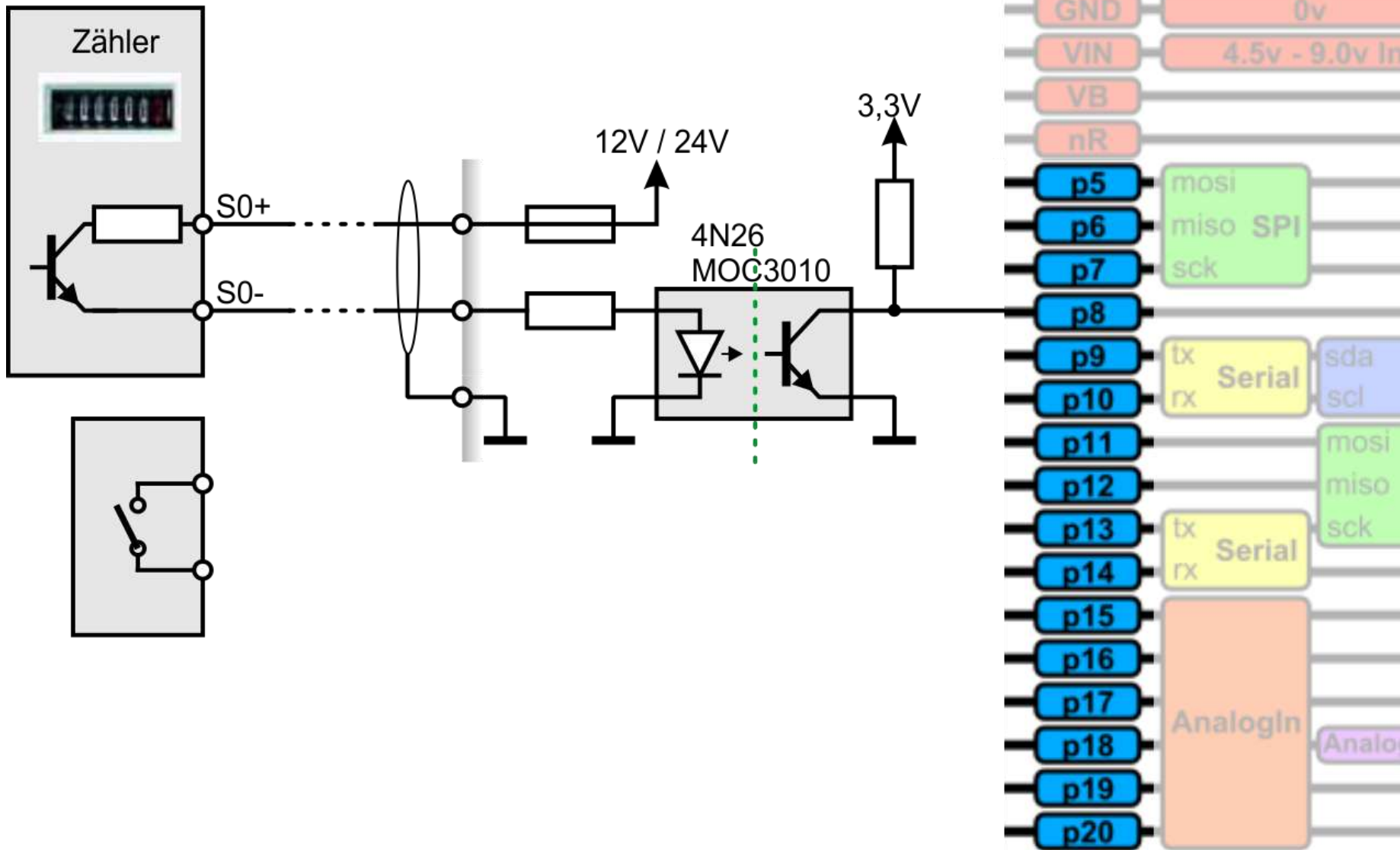
MSB		64-bit 'Registration' ROM number				LSB	
8-bit CRC		48-bit Serial Number		8-bit Family Code			
MSB	LSB	MSB	LSB	MSB	LSB		



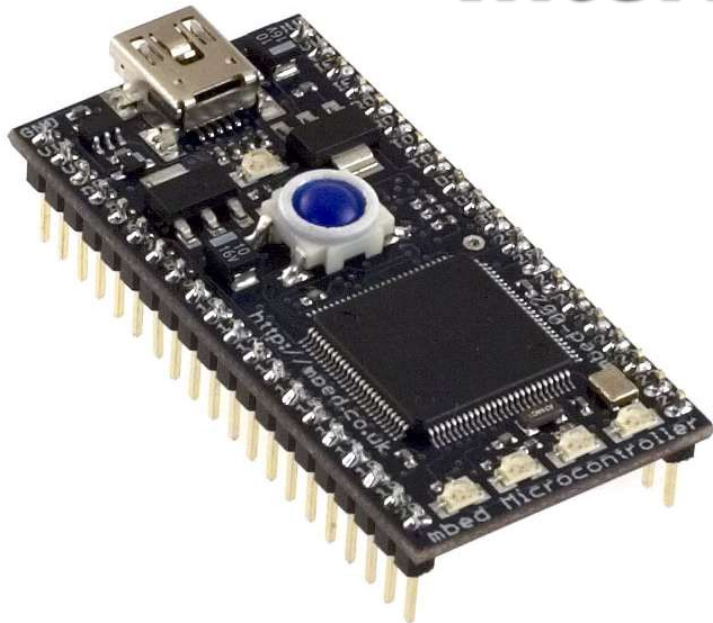
# Sensoren → 230V-Erkennung



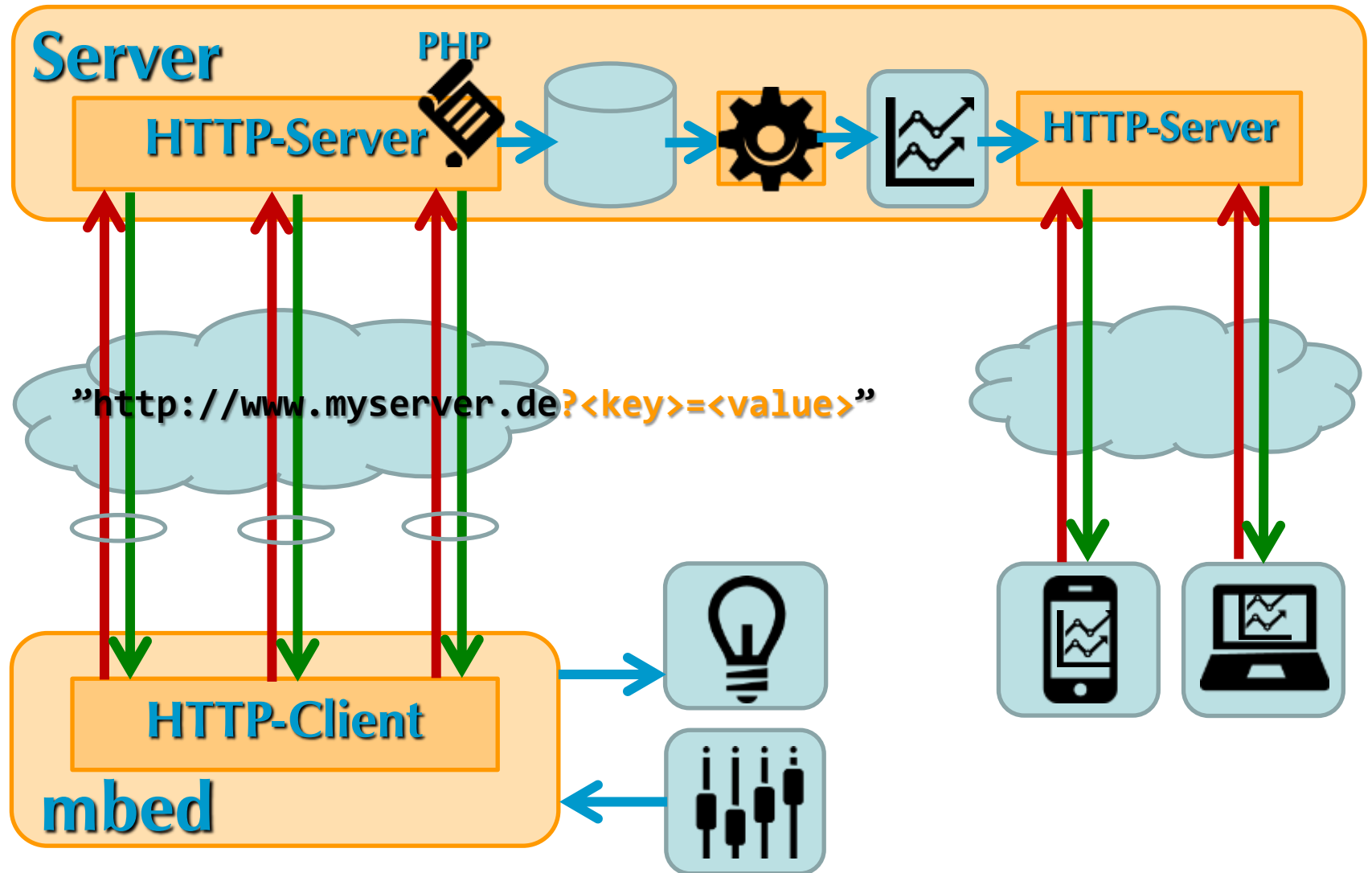
# Sensoren → S0-Bus



## Internet of Things



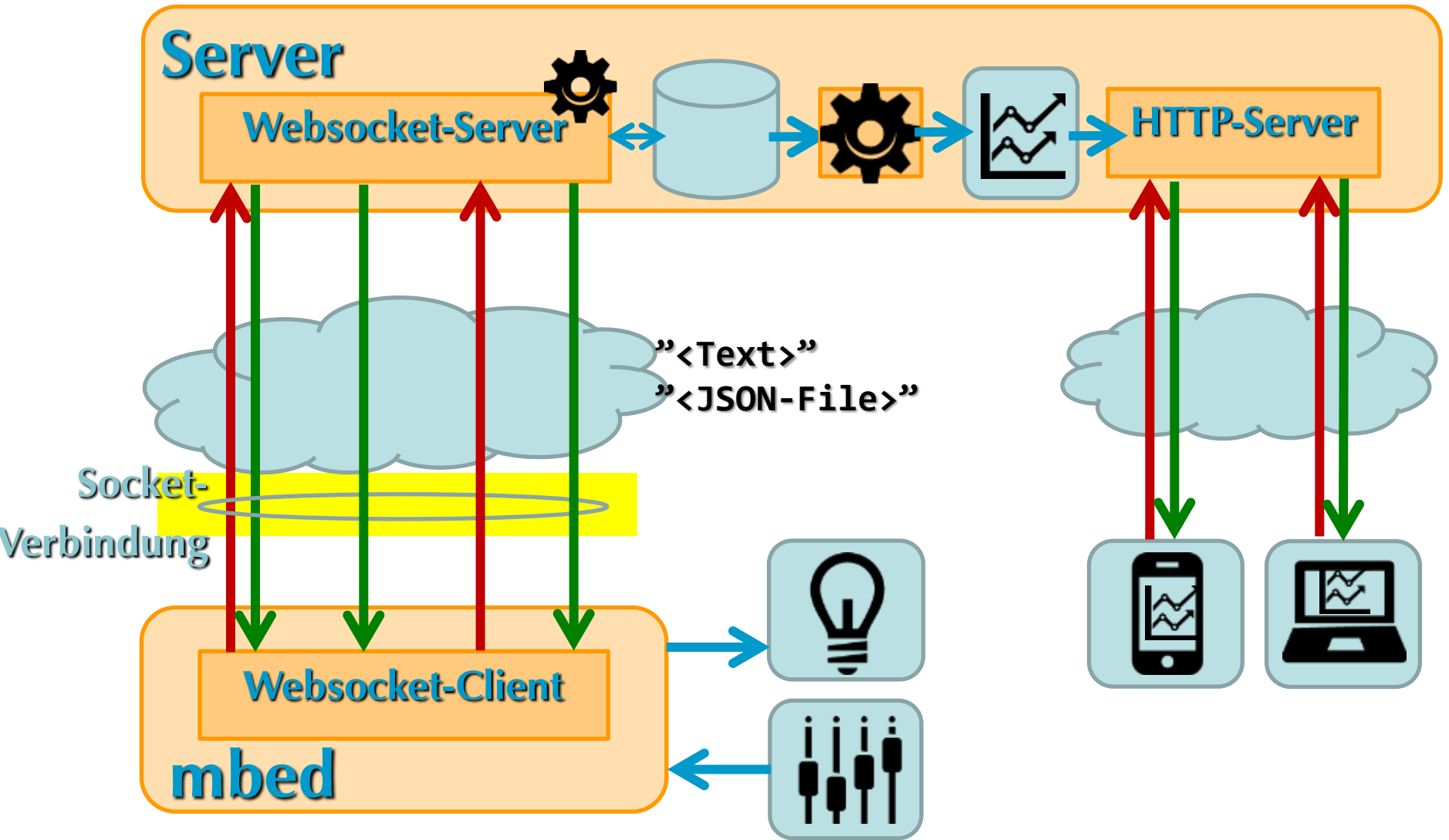
# IOT → mbed als HTTP-Client



# IOT → mbed als HTTP-Client

```
1  #include "mbed.h,,
2  #include "EthernetInterface.h,,
3  #include "HTTClient.h"
4  int main() {
5      EthernetInterface eth;
6      eth.init();                //Use DHCP
7      eth.connect();
8
9      HTTPClient http;
10     char str[512];
11     int ret = http.get("http://www.myserver.de?<key>=<value>", str, 512);
12
13     ret = http.post(...
14
15     eth.disconnect();
16 }
17
18
19
```

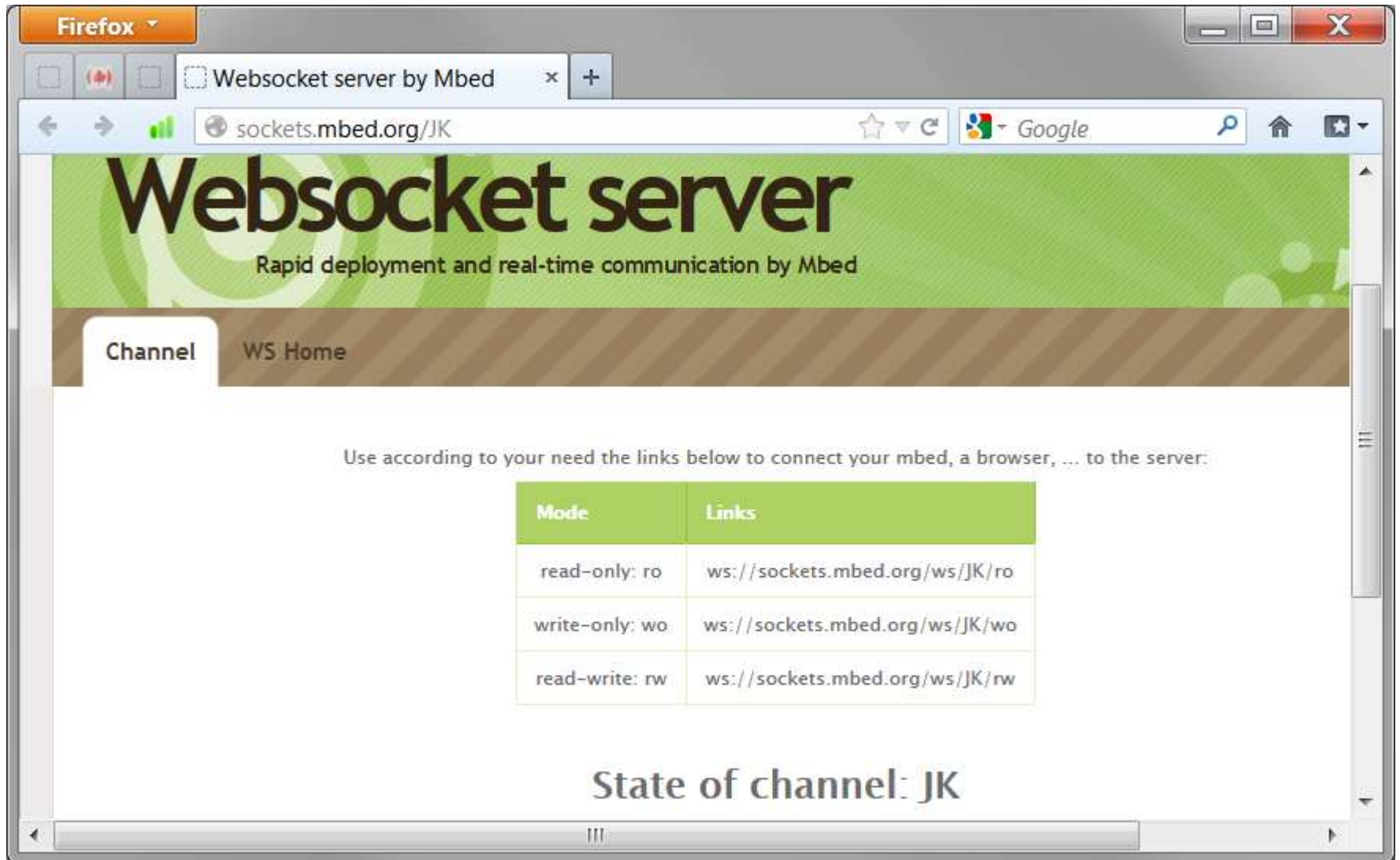
# IOT → mbed als WebSocket-Client



# IOT → mbed als WebSocket-Client

```
1  #include "mbed.h"
2  #include "EthernetInterface.h"
3  #include "WebSocket.h"
4  BusIn taster(p8, p14);   AnalogIn ain(p15);
5  char szData[128];
6
7  int main() {
8      EthernetInterface eth;   eth.init();   eth.connect();
9
10     WebSocket ws("ws://sockets.mbed.org:443/ws/JK/rw");
11     ws.connect();
12
13     while (1) {
14         sprintf(szData, "mbed: Poti=%d Taster=%d", (int)(ain*1000),taster);
15         ws.send(szData);
16
17         if (ws.read(szData))   printf("rcv: %s\r\n", szData);
18         wait(1.0);
19     }
```

# IOT → Websocket-Beispiel: socket.mbed.org



Firefox

Websocket server by Mbed

sockets.mbed.org/JK

## Websocket server

Rapid deployment and real-time communication by Mbed

Channel WS Home

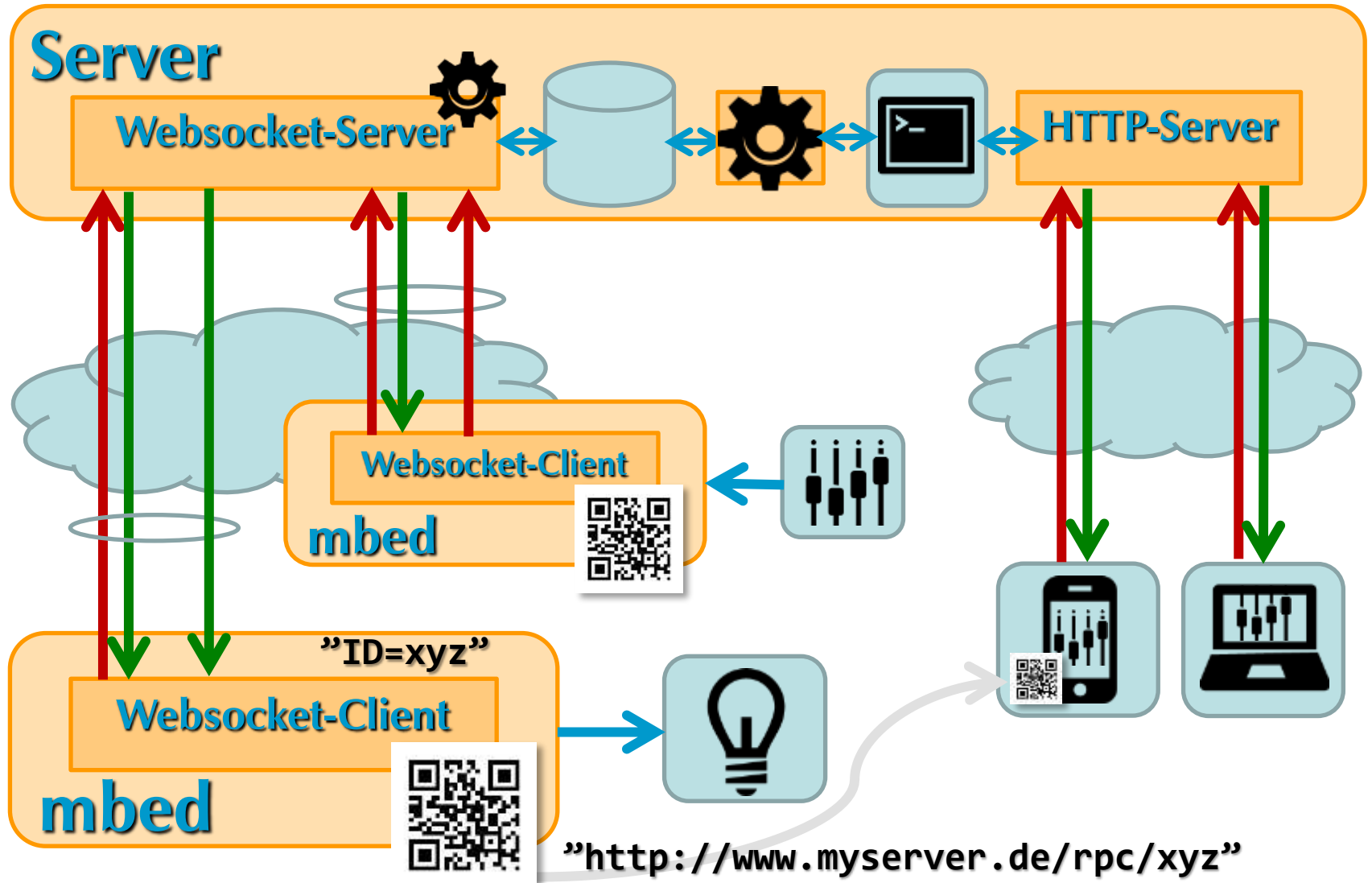
Use according to your need the links below to connect your mbed, a browser, ... to the server:

Mode	Links
read-only: ro	<a href="ws://sockets.mbed.org/ws/JK/ro">ws://sockets.mbed.org/ws/JK/ro</a>
write-only: wo	<a href="ws://sockets.mbed.org/ws/JK/wo">ws://sockets.mbed.org/ws/JK/wo</a>
read-write: rw	<a href="ws://sockets.mbed.org/ws/JK/rw">ws://sockets.mbed.org/ws/JK/rw</a>

State of channel: JK



# IOT → mbed als WebSocket-Client



# IOT → WebSocket-Beispiel: tools.mbed/iot

Firefox

Websocket server by Mbed

Internet of Things

tools.mbed.org/iot/

# Internet of Things

Rapid deployment and real-time sensors demo

Real-time Sensors

About

Clear All

## HTML5

## mbed

## Cortex-M0

Intelligent Processors by ARM\*

WIC

Debug

NVIC

ARM CoreLink

Debugger interface

Bus Matrix

AMBA AHB-to-APB interface

### Account

This account is:

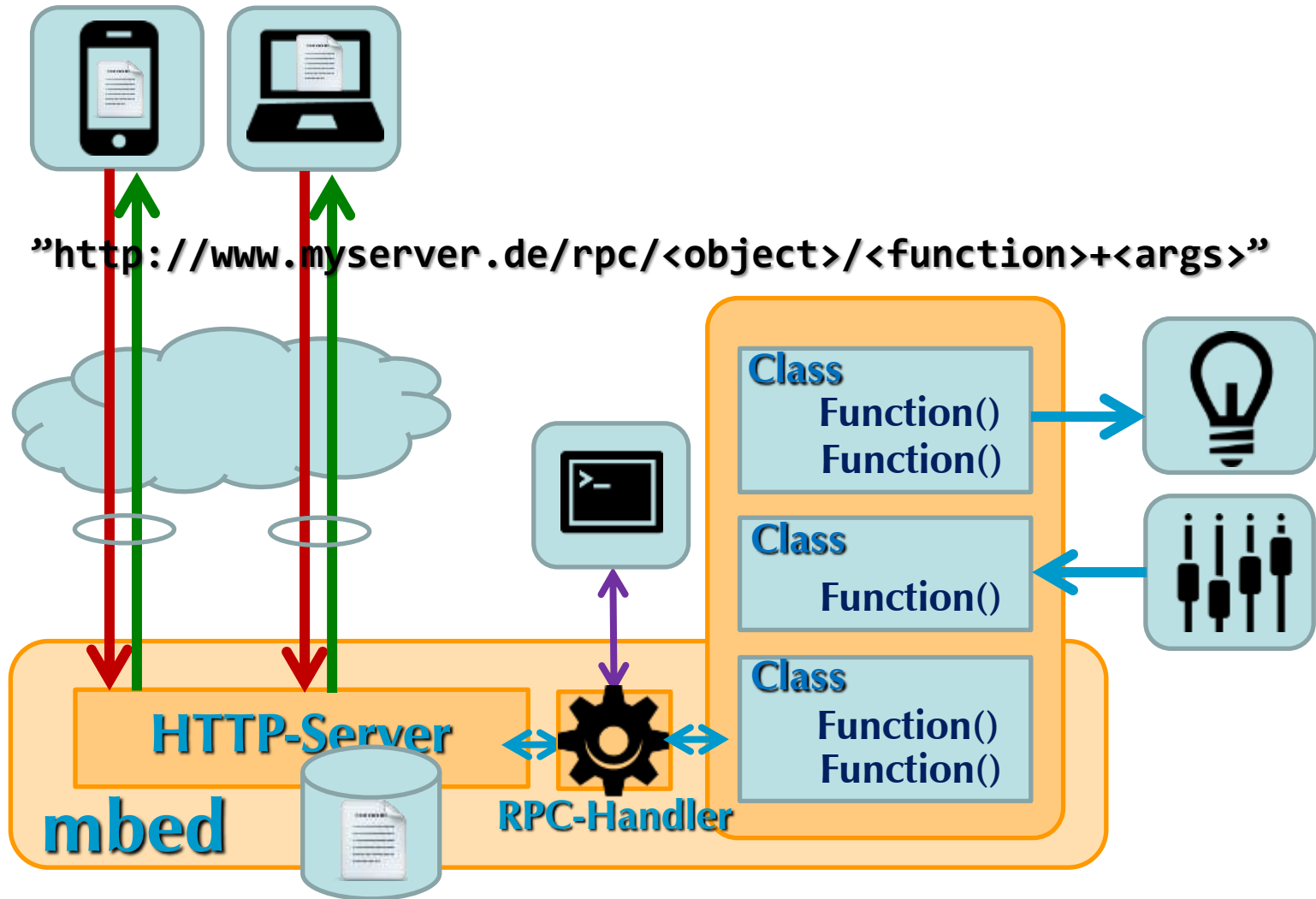
You :)

### Instructions

To interact with sensors:

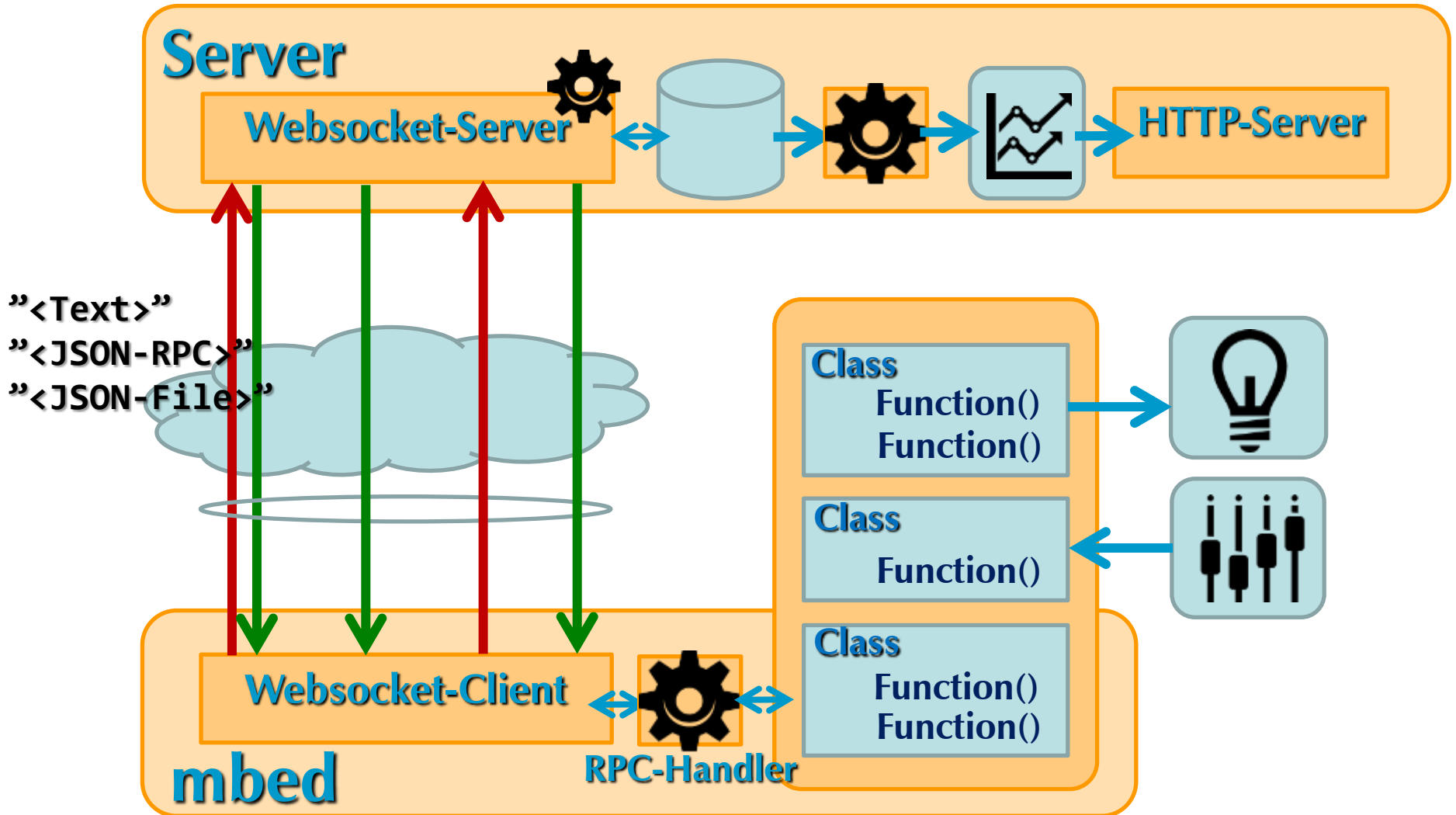
# IOT → mbed als HTTP-Server+RPC

(RPC = Remote Procedure Call)



# IOT → mbed als Websocket-Client+RPC

(RPC = Remote Procedure Call)



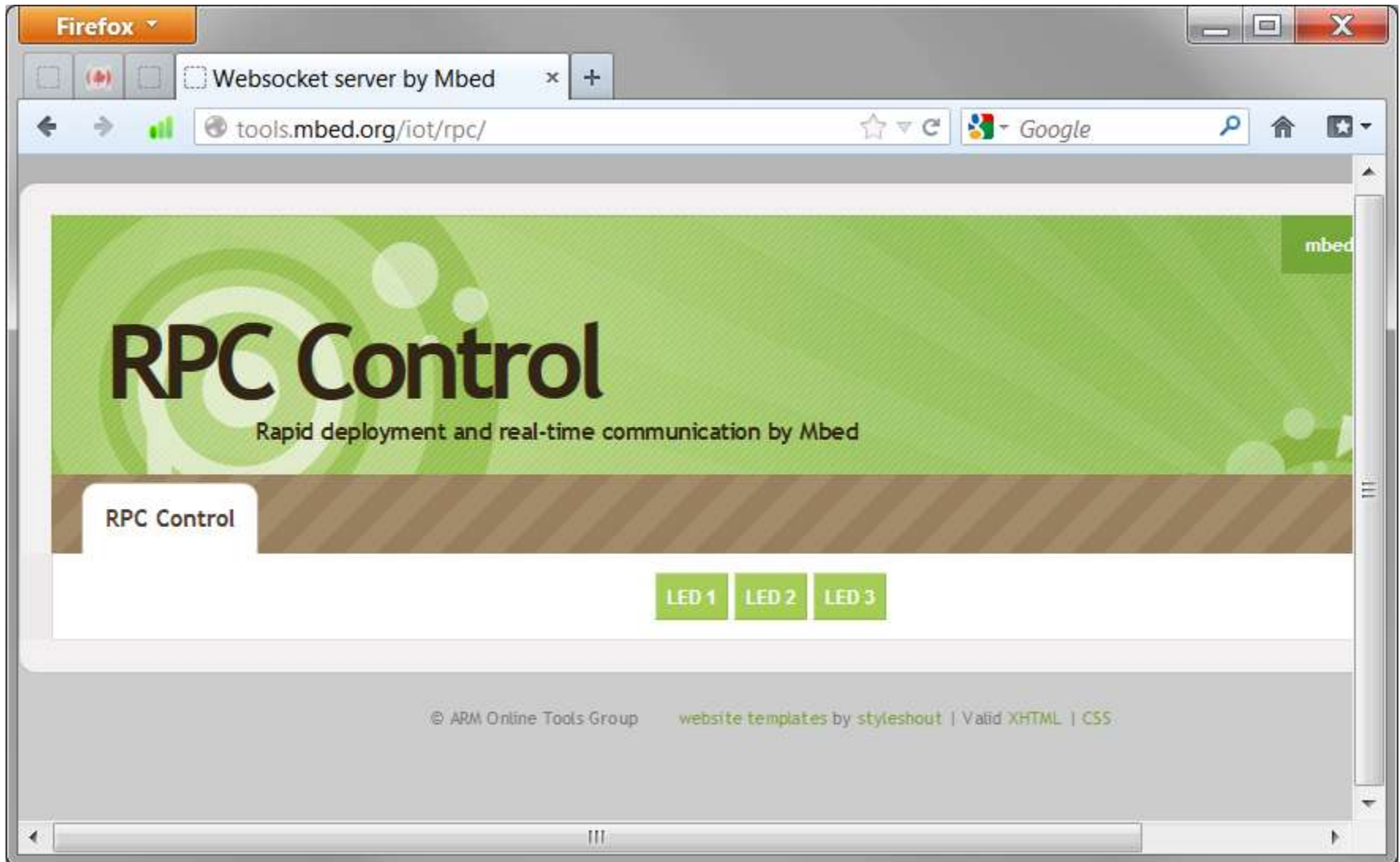
# IOT → mbed als WebSocket-Client+RPC

```
1  #include "mbed.h"
2  #include "MbedJSONRpc.h"
3  #include "MbedJSONValue.h"
4  #include "EthernetInterface.h"
5  #include "WebSocket.h"
6  BusOut ledbus(LED1, LED2, LED3, LED4);
7  //websocket: configuration with sub-network = demo and mbed_id = mbed_led
8  WebSocket ws("ws://sockets.mbed.org/rpc/KNF/mbed_jk");
9  //RPC object attached to the websocket server
10 MbedJSONRpc rpc(&ws);
11
12 //Test class
13 class MyRPCclass {
14     //...
15     void led(MbedJSONValue& in, MbedJSONValue& out) {
16         ledbus = in[0].get<int>();
17     }
18 };
19
20 MyRPCclass myRPCobject;
21
22 // to be continued...
```

# IOT → mbed als WebSocket-Client+RPC

```
1 // ...continued
2
3 int main() {
4     EthernetInterface eth;    eth.init();    eth.connect();
5
6     ws.connect();
7
8     Base::add_rpc_class<AnalogIn>();
9     Base::add_rpc_class<DigitalIn>();
10    Base::add_rpc_class<DigitalOut>();
11    Base::add_rpc_class<BusOut>();
12    Base::add_rpc_class<BusIn>();
13
14    // register „led“
15    rpc.registerMethod("led", &myRPCobject, &MyRPCclass::led));
16
17    // wait for incoming CALL requests
18    rpc.work();
19 }
```

# IOT → RPC-Beispiel: tools.mbed.org/iot/rpc



# IOT → RPC-Beispiel: cosm.com

The screenshot shows a Firefox browser window with the address bar displaying `https://cosm.com/users/jocis`. The page title is "Cosm - My Console". A notification banner at the top states "Pachube is now Cosm" with a link to "Find out more". The main header features the "cosm BETA" logo, a "Menu..." dropdown, a search bar, and a user profile for "jocis".

The main content area is titled "My Console" and includes a "+ Device/Feed" button. The first device card is for "mbed", which is locked. It displays a line graph with two data series: "Poti" and "Taster". The "Poti" series shows a decreasing trend, while the "Taster" series shows an increasing trend. To the right of the graph are two red percentage indicators. The "mbed" card has a "5 minu..." refresh interval and a settings gear icon.

The second device card is titled "Sound Level feed" and has a "6 hours" refresh interval and a settings gear icon.





**Alle Logos, Marken und Namen sind Eigentum der jeweiligen Firmen**