Zwischen Mikrocontroller und Mini-Rechner

R. Sitter 25.06.2020

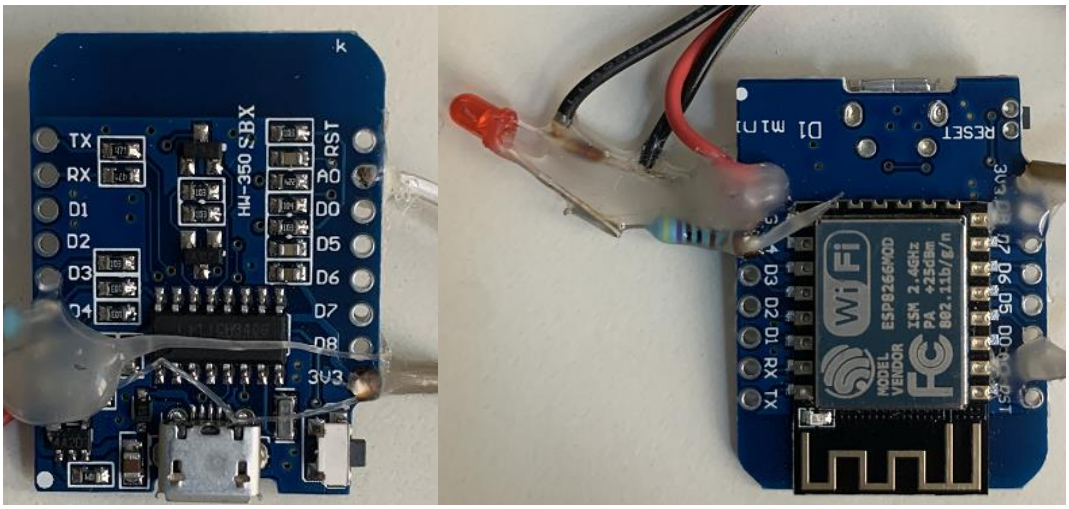
www.sitter.de

Inhaltsverzeichnis

0. Vorbemerkungen

Calliope und BBC micro:bit sind gut für Anfänger geeignet. Sie haben aber nicht die Möglichkeiten eines Computers (z.B. Tastatur und Bildschirm). Dafür kann man gut die Hardware verstehen und damit etwas Einfaches aufbauen.
Auch der ARDUINO nimmt eher eine Zwischenposition ein. Beim ARDUNIO benötigt man mehr Wissen, kann aber auch mehr Peripherie anschließen und bauen.
Das Modul mit dem geringsten Umfang an Leistung ist der ESP8266, ein WLAN – Miniprozessor, der für die Hausautomation ideal geeignet ist und sich mit der Entwicklungsumgebung des ARDUIONO in C gut programmieren lässt.
Mit allen Mini-Computern kann man Hausautomation betreiben, kleine Roboter bauen oder Maschinen betreiben, die computergesteuerte Prozesse benötigen. Vor allem kann man mit den Geräten viel lernen.
In der Regel geht es um Spannungswerte zwischen 3 und 5 Volt. Die Geräte sind diesbezüglich natürlich empfindlich. Die Last an den Pins sollte eingehalten werden. In der Regel schadet eine LED mit Vorwiderstand nicht. Verschiedene Module, die z.B. für den ARDUINO angeboten werden, sind fast bei allen Geräten einsetzbar. Die Steuerung von Servos und Motoren folgt fast immer den Regeln, wie sie bei den Mikrocontrollern gelten (sind auch dort beschrieben).

1. ESP8266



Die ESP8266 sind Chips vom chinesischen Hersteller Espressif sind 32bit Mikrocontroller mit integrierter Wi-Fi Schnittstelle. Diese sind meist in verschieden Module verbaut, die auch Flashspeicher und USB-Schnittstelle bereithalten (z.B. der D1 Mini). Mit Preisen um 2 Euro sind die ESP Module eine sehr preisgünstige Netzwerk-Anbindung zum Basteln.

Der D1 Mini hat einen ESP-8266 (mit 80MHz) inklusive knapp 36 kB frei verfügbarem RAM sowie 4MB Flash-Speicher, neun (für 3.3V ausgelegte, nicht 5V-tolerante) digitale Ein-/Ausgänge (mit Ausnahme von D0 alle mitsamt Interrupt-/PWM-/I2C/One-Wire-Unterstützung), einen analogen Eingang (für Spannungen bis 3.2V - aufgrund eines internen Spannungsteilers) und einen USB-Anschluss, über den der Baustein mit Strom versorgt und programmiert werden kann.

Genauso wie viele andere preisgünstige Controller aus China verwendet auch der WeMos D1 mini einen "USB-to-Serial"-Baustein vom Typ CH340 bzw. CH341. Bevor mit dem D1 mini (an einem Windows- oder Mac-Rechner) gearbeitet werden kann, muss deshalb erst einmal ein passender Treiber für diesen USB-Chip installiert werden.

Das ESP-01 Modul basiert auf dem ESP8266. Es wird mit 512 kByte (in blau) oder 1 MByte (in schwarz) Flash Speicher verkauft. Die rote Power-LED nimmt etwa 0,5 mA Strom auf, die blaue LED hängt an GPIO2.

Das ESP-07 Modul ist normalerweise mit einem 1 MByte Flash Chip, einer Keramikantenne und einer Steckbuchse für externe Antennen ausgestattet. Um die externe Antenne nutzen zu können, muss

man den gezeigten Kondensator entfernen. Die rote Power-LED nimmt etwa 0,5 mA Strom auf. Die blaue LED hängt an GPIO2.

Das NodeMCU Board besteht aus einem ESP-12 Modul mit Spannungsregler und USB-UART Interface. Der Flash Speicher ist 4 Megabytes groß. Für die erstem Programmierversuche ist das die ideale Ausstattung.

1.1. Programmieren des ESP8266

Am einfachsten kann man den ESP8266 mit der ARDUINO IDE programmieren. Man muss dazu das Board ESP8266 nachladen, da es nicht zum Standardumfang gehört. Alternativ kann man mit der ARDUINO IDE programmieren, die Binärdatei erzeugen und mit einem Brennprogramm auf den ESP8266 hochladen. Das Brennprogramm würde auch reichen, wenn man fertige Programme in Binärform aus dem Internet hochladen möchte.

Für den ESP8266 gibt es ein breites Angebot. Fertige Binärdateien bieten z.B. über einen AP (AccessPoint) zuerst die Einstellungen an, so dass man danach über das Hauseigene WLAN den ESP8266 mit dem Router verbinden kann.

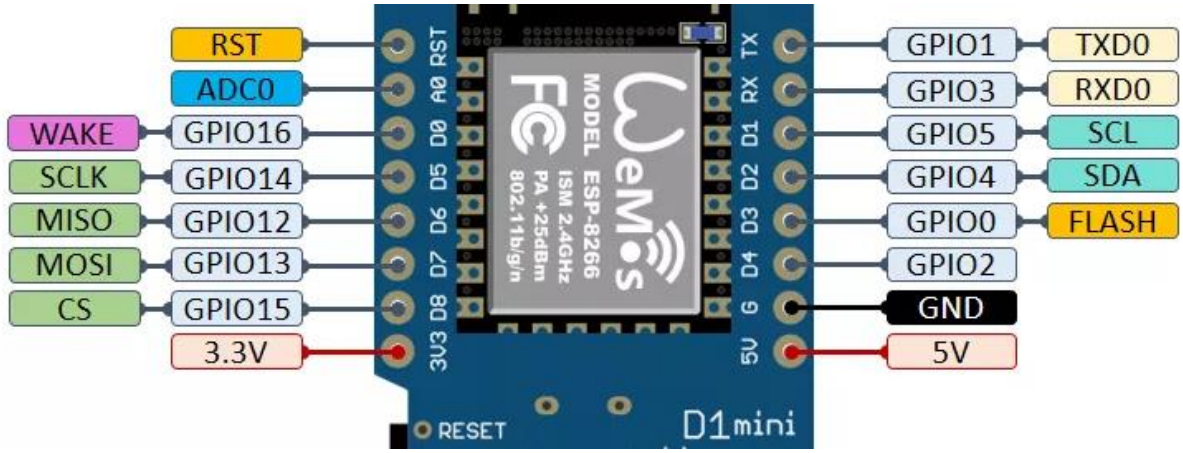
Flexibler ist man natürlich trotzdem, wenn man die Programmierung selbst vornimmt. Hier ein Beispiel für **ein einfaches Programm** in C für die ARDUINO IDE:

```
// Pins definieren
const int ledPin = 2;

void setup()
{
  pinMode(ledPin, OUTPUT);
}

void loop()
{
  digitalWrite(ledPin,HIGH);
  delay(1000);
  digitalWrite(ledPin,LOW);
  delay(1000);
}
```

Pinout des ESP8266 D1 mini:



D4 wäre in diesem Fall GPIO2.

Ein Beispielprogramm für die Verbindung mit einem hauseigenem Router und dem Senden eines Analogwertes über eine Internetadresse, die diesen Wert weiter verarbeitet:

```
#include<ESP8266WiFi.h>
// nur bei Tastendruck
```

```
const char* ssid     = "XXXXXXXXXX";
const char* password = "YYYYYYYYYY";
const char* host = "www.zzzzzz.de";
const int EPIN = 16; // D0 als Taster
const int APIN = A0;

// Analog Eingang
int Volume;
// Digital Eingang
int Eingang;

void setup() {

  pinMode(5, OUTPUT);
  pinMode(EPIN, INPUT);
  digitalWrite(5, 0);
  Serial.begin(9600);
  delay(100);
  /* Mit dem WLAN-Netzwerk verbinden und Rückmeldungen im seriellen Monitor ausgeben*/
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

  digitalWrite(5, 1);
}

// Haupt-Schleife
void loop() {

// etwas warten
  delay(5000);

  Volume = analogRead(APIN);
  Eingang = digitalRead(EPIN);

// Wenn Taste gedrückt
if (Eingang == 0) {

  Serial.print("connecting to ");
  Serial.println(host);
  // Den WLAN-Client benutzen für TCP-Verbindungen
  WiFiClient client;
  const int httpPort = 80;
  if (!client.connect(host, httpPort)) {
    Serial.println("connection failed");
    return;

  }

  // Zugang für die Anfrage festlegen
  String url = "//ESP8266/m.php?Value=";
```

```
  url += String(Volume);
  Serial.print("Requesting URL: ");
  Serial.println(url);
  // Anfrage an den Server stellen
  client.print(String("GET ") + url + " HTTP/1.1\r\n" +
               "Host: " + host + "\r\n" +
               "Connection: close\r\n\r\n");
  delay(500);
  // Rückmeldungen vom Server lesen und im seriellen Monitor ausgeben
  while(client.available()){
    String line = client.readStringUntil('\r');
    Serial.print(line);
  }

  Serial.println();
  Serial.println("closing connection");
 }
}
```

Pin 5 ist so programmiert, dass eine LED an diesem Pin das erfolgreiche Anmelden am Router signalisieren kann. XXX.., YYY.. sind die Zugangsdaten zum Router und zzz… ist eine Domäne, die ein php-Programm zur Weiterverarbeitung des Messwertes zur Verfügung stellt. Den seriellen Monitor kann man später auch ausschalten, wenn das Programm stabil läuft.

Ein PHP-Programm „m.php“, welches einen solchen Analogwert verarbeitet, könnte z.B. so aussehen:

```
<?php
$servername = "db.xxxx.de";
$username = "yyyyy";
$password = "zzzzzz";
$dbname = "NNNNN";

// HTML

$varip = getenv('REMOTE_ADDR'); // GET Argumente Einlesen
$Nachricht=htmlspecialchars($_GET["Value"]);

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
//23
$temp=str($Nachricht);

$sql = "INSERT INTO temperaturen (temp) VALUES ('$temp')";

if ($conn->query($sql) === TRUE) {
    echo "Record updated successfully";
} else {
    echo "Error updating record: " . $conn->error;
}

$conn->close();
?>
```

Im Beispiel wird der Analogwert in eine mySQL-Datenbank geschrieben und kann mit einem weiteren php-Programm in der Historie aufgelistet werden:

```
<?php
$servername = "db.xxxx.de";
```

```
$username = "yyyyy";
$password = "zzzzzz";
$dbname = "NNNNN";echo "<h1>Temperaturen Zuhause</h1>";
echo "<br>";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "SELECT * FROM temperaturen order by id desc";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {
$dtime=new Datetime($row["time"]);
        echo "id: " . $row["id"]. " - Temperatur: <b>" . $row["temp"]. "</b> Grad am " . $dtime-
>format('d.m.Y - H:i'). "<br>";
    }
} else {
    echo "0 results";
}
$conn->close();
echo "<br><br><a href='templist_del.php'>Liste l&#246;schen</a><br><br>";
?>
```

Ein Beispiel-Programm für eine Hauslichtsteuerung mit 8 Ausgängen:

```
#include <ESP8266WebServer.h>
const char* ssid = "XXXXXXX";          //Zugangsdaten des Routers
const char* password = "YYYYYY";

const int PIN = 16; // D0 als Taster

ESP8266WebServer server(80);

void handleRoot() {                    //Html Startseite
  String content = "<!DOCTYPE html><html lang='de'><head><meta charset='UTF-8'><meta name=
viewport content='width=device-width, initial-scale=1.0,' user-scalable=yes>";
  content += "<title>Sitter Hauslicht</title>\r\n<style type='text/css'><!-- .button1 {height:40px;
width:160px; background-color: #268FFF; font-size:16px} .button2 {height:40px; width:160px;
background-color: #FE0320; font-size:16px} --></style></head>\r\n<body><h2>Sitter
Hauslicht</h2><p>";
  if (ledState1 == HIGH) {
    content += "<form action='/' method='POST'><input class='button1' name='ledon1' type='submit'
value='K&uuml;che T&uuml;r an'><br><small>K&uuml;che T&uuml;r ist aus</small><br>";
  }
  else {
    content += "<form action='/' method='POST'><input class='button2' name='ledoff1' type='submit'
value='K&uuml;che T&uuml;r aus'><br><small>K&uuml;che T&uuml;r ist an</small><br>";
  }
  if (ledState2 == HIGH) {
    content += "<form action='/' method='POST'><input class='button1' name='ledon2' type='submit'
value='Schlafzimmer an'><br><small>Schlafzimmer ist aus</small><br>";
  }
  else {
    content += "<form action='/' method='POST'><input class='button2' name='ledoff2' type='submit'
value='Schlafzimmer aus'><br><small>Schlafzimmer ist an</small><br>";
  }
  if (ledState3 == HIGH) {
```

```
    content += "<form action='/' method='POST'><input class='button1' name='ledon3' type='submit'
value='Flur an'><br><small>Flur ist aus</small><br>";
  }
  else {
    content += "<form action='/' method='POST'><input class='button2' name='ledoff3' type='submit'
value='Flur aus'><br><small>Flur ist an</small><br>";
  }
  if (ledState4 == HIGH) {
    content += "<form action='/' method='POST'><input class='button1' name='ledon4' type='submit'
value='Bad an'><br><small>Bad ist aus</small><br>";
  }
  else {
    content += "<form action='/' method='POST'><input class='button2' name='ledoff4' type='submit'
value='Bad aus'><br><small>Bad ist an</small><br>";
  }
  if (ledState5 == HIGH) {
    content += "<form action='/' method='POST'><input class='button1' name='ledon5' type='submit'
value='Bibo T&uuml;r an'><br><small>Bibo T&uuml;r ist aus</small><br>";
  }
  else {
    content += "<form action='/' method='POST'><input class='button2' name='ledoff5' type='submit'
value='Bibo T&uuml;r aus'><br><small>Bibo T&uuml;r ist an</small><br>";
  }
  if (ledState6 == HIGH) {
    content += "<form action='/' method='POST'><input class='button1' name='ledon6' type='submit'
value='Rotlicht an'><br><small>Rotlicht ist aus</small><br>";
  }
  else {
    content += "<form action='/' method='POST'><input class='button2' name='ledoff6' type='submit'
value='Rotlicht aus'><br><small>Rotlicht ist an</small><br>";
  }
  if (ledState7 == HIGH) {
    content += "<form action='/' method='POST'><input class='button1' name='ledon7' type='submit'
value='L1 an'><br><small>L1 ist aus</small><br>";
  }
  else {
    content += "<form action='/' method='POST'><input class='button2' name='ledoff7' type='submit'
value='L1 aus'><br><small>L1 ist an</small><br>";
  }
  if (ledState8 == HIGH) {
    content += "<form action='/' method='POST'><input class='button1' name='ledon8' type='submit'
value='L2 an'><br><small>L2 ist aus</small><br>";
  }
  else {
    content += "<form action='/' method='POST'><input class='button2' name='ledoff8' type='submit'
value='L2 aus'><br><small>L2 ist an</small><br>";
  }
  content += "</form></html>";
  server.send(200, "text/html", content);
}

void setup(void) {
  pinMode(LEDPIN1, OUTPUT);
  digitalWrite(LEDPIN1, HIGH);
  pinMode(LEDPIN2, OUTPUT);
  digitalWrite(LEDPIN2, HIGH);
  pinMode(LEDPIN3, OUTPUT);
  digitalWrite(LEDPIN3, HIGH);
  pinMode(LEDPIN4, OUTPUT);
  digitalWrite(LEDPIN4, HIGH);
  pinMode(LEDPIN5, OUTPUT);
  digitalWrite(LEDPIN5, HIGH);
```

```
  pinMode(LEDPIN6, OUTPUT);
  digitalWrite(LEDPIN6, HIGH);
  pinMode(LEDPIN7, OUTPUT);
  digitalWrite(LEDPIN7, HIGH);
  pinMode(LEDPIN8, OUTPUT);
  digitalWrite(LEDPIN8, HIGH);
  Serial.begin(9600);
  WiFi.mode(WIFI_STA);        //Station-Modus --> Esp8266 im Heimnetzwerk einloggen
  WiFi.begin(ssid, password);
  Serial.println("");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("SSID: ");
  Serial.println(ssid);
  Serial.print("Passwort: ");
  Serial.println(password);
  Serial.print("IP-Adresse: ");
  Serial.println(WiFi.localIP());
  server.on("/", handleRoot);
  server.onNotFound([]() {
    server.sendContent("HTTP/1.1 303 OK\r\nLocation: /\r\nCache-Control: no-cache\r\n\r\n");
  });
  server.begin();
  Serial.println("HTTP server started");
}

void loop(void) {
  server.handleClient();
}
```

Der ESP8266 ist also vielfältig einsetzbar, entweder als AP oder im hausinternen Netzwerk und über eine Portfreigabe sogar über das Internet frei zugänglich.

Mit dem geringen Preis ist der ESP8266 ideal zur Verwirklichung von IoT – Projekten (InternetofThings).

2. ARDUINO

Der ARDUIONO kommt auch in verschiedenen Versionen daher. Der Kern des ARDUINO ist ein ATMEL Prozessor (siehe Mikrocontroller). Es ist also relativ einfach, den ARDUINO selbst zu bauen. Für den ATMEGA 8, 16… und auch für den ATtiny2313 gibt es Schaltungen mit USB.

Der ARDUINO UNO ist sehr verbreitet. Modelle gibt es dann auch mit WIFI. Der ARDUINO Nano ist der kleinste und passt gut auf kleine Roboter.

2.1. ARDUINO Programmierung

Das einfachste Programm für den ARDUINO ist natürlich das „Blinken“:

```
void setup() {
  // digital pin LED_BUILTIN ist output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

Wenn der erste Test erfolgreich war, kann an sich an andere Dinge wagen. In der ARDUINO IDE muss das Bord eingestellt sein, also z.B. der ARDUINO UNO oder ARDUINO Nano und das Hochladen des kompilierten Programms muss natürlich geklappt haben.

MotorShield

Ein Programm für einen Roboter mit zwei Motoren in zwei Richtungen (z.B. ein Motorshield mit den erforderlichen Treibern benutzen), einem Entfernungsmesser und einem Servo, der den Entfernungsmesser bewegt, könnte z.B. so aussehen:

```
//
// Sitter Robotics
// 19.10.2018
//
// Definition Servo und Abstandssensor
//
#include <Servo.h>
Servo myservo;  // Servo-Objekt
int pos = 0;    // variable to store the servo position
// Abstandssensor
int trigger = 7;
int echo = 8;
long dauer = 0; // Zeitdauer der Refelexion
long entfernung = 0; // Entfernung in cm
//
// Definintion der Motorsteuerung
//
int links_hin = 9;    // linke Kette Rückwärts
int links_vor = 10;   // linke Kette Vorwärts
int rechts_vor = 11;  // rechte Kette Vorwärts
int rechts_hin = 12;  // rechte Kette Rückwärts

int live = 0;

void setup() {
// Servo festlegen
```

```
  myservo.attach(6);  // Servo an Pin 6
// Abstandsensor Pinrichtungen festlegen
  pinMode(trigger, OUTPUT);
  pinMode(echo, INPUT);
// Steuerkanäle auf Ausgang
  pinMode(links_vor, OUTPUT);
  pinMode(links_hin, OUTPUT);
  pinMode(rechts_vor, OUTPUT);
  pinMode(rechts_hin, OUTPUT);
  motor_stop();
  myservo.write(90);
  delay(500);
}

void loop() {
 if (live < 20) {
// Enfernung nach vorn messen
  radar();
  if (entfernung < 20) {
    motor_stop();
    delay(500);
    motor_back(200);
    myservo.write(180); // Servo nach rechts
    radar();
    delay(500);
    if (entfernung > 20) {
        motor_links(1000);
    } else {
        myservo.write(0); // Servo nach links
        radar();
        delay(500);
        if (entfernung > 20) {
            motor_rechts(1000);
        }
    }
    myservo.write(90);
    delay(500);
  } else {
    // geradeaus fahren
    motor_vor(500);
  }
 live = live + 1;
 } // end live if
}

// Motor-Funktionen
void motor_stop() {
  // stop bzw. Startposition
  digitalWrite(links_vor, LOW);
  digitalWrite(links_hin, LOW);
  digitalWrite(rechts_vor, LOW);
  digitalWrite(rechts_hin, LOW);
}
void motor_vor(int dauer) {
// vorwärts fahren
  digitalWrite(links_vor, HIGH);
  digitalWrite(rechts_vor, HIGH);
  delay(dauer);                    // bleiben
  digitalWrite(links_vor, LOW);    // Ende
  digitalWrite(rechts_vor, LOW);
}
void motor_rechts(int dauer) {
```

```
// rechts drehen
  digitalWrite(links_vor, HIGH);
  digitalWrite(rechts_hin, HIGH);
  delay(dauer);                  // bleiben
  digitalWrite(links_vor, LOW);    // Ende
  digitalWrite(rechts_hin, LOW);
}
void motor_links(int dauer) {
// links drehen
  digitalWrite(links_hin, HIGH);
  digitalWrite(rechts_vor, HIGH);
  delay(dauer);                  // bleiben
  digitalWrite(links_hin, LOW);    // Ende
  digitalWrite(rechts_vor, LOW);
}
void motor_back(int dauer) {
// rückwärts fahren
  digitalWrite(links_hin, HIGH);
  digitalWrite(rechts_hin, HIGH);
  delay(dauer);                  // bleiben
  digitalWrite(links_hin, LOW);    // Ende
  digitalWrite(rechts_hin, LOW);
}
// Entferung messen
int radar() {
  digitalWrite(trigger, LOW);
  delay(5);
  digitalWrite(trigger, HIGH);
  delay(10);
  digitalWrite(trigger, LOW);
  dauer = pulseIn(echo,HIGH);
  entfernung = (dauer/2) * 0.03432;
}
```

Dem Einsatz von Sensoren sind am ARDUNINO eigentlich kaum Grenzen gesetzt. Anschlüsse kann man über I2C erweitern oder über andere Porterweiterungen. Man sollte immer die Laufzeit der Protokolle im Blick haben, wenn es um Reaktionszeiten geht, die Berücksichtigung finden müssen. Die andere Grenze des ARDUINO ist natürlich die Gesamtleistung, der verfügbare Strom und der Speicherplatz. Ein ARDUIONO mit ATmega328P hat z.B. 32 KByte Flash-Speicher und 1 kByte EEPROM, 2 KByte internen SRAM. Auch eine SD-Card für externe Speicherung von Daten usw. kann natürlich mit dem ARDUNIO über ein SD-Card-Modul verbunden werden.

Zu Schrittmotoren etc. siehe Mikrocontroller.

Die ARDUINO funktionieren mit einem Bootloader, d.h. wer einen selbst baut, muss auch den Bootloader brennen, damit es ein ARDUINO wird. Anfang 2018 hat Arduino den Bootloader für die Nano-Boards ausgetauscht. Benutzer eines Original-Boards müssen daher die Software für das Hochladen der Programme aktualisieren. Je nach Alter bzw. Kaufdatum muss man dann später den entsprechenden Bootloader auswählen. Damit steigt auch die Datenübertragungsrate von 57.600 Baud auf 115.200 Baud.

3. B O B 3

Auch der B O B 3 basiert auf einem ATmega88. Er hat Tastsensoren, Photosensoren und „Augen“ in Form von RGB-LEDs. Eine IR-LED kann für einen Fernbedienung benutzt werden oder mit der IR-LED als Annäherungssensor.

Programmiert wird der B O B 3 über die USB-Schnittstelle. Es gibt Programmierumgebungen für die verschiedenen Betriebssysteme oder aber auch in HTML5 (Browserversion) über www.progbob.org.

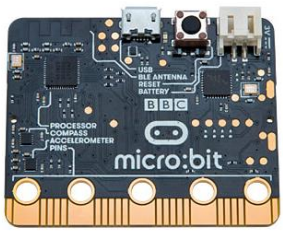
4. Calliope

Den Calliope gibt es auch in verschiedenen Versionen. Er hat Taster, Tastsensoren, 25 rote sowie eine RGB-LED, einen kombinierten Lagesensor mit Bewegungssensor und Kompass sowie ein Funk-Modul, Mikrofon und Piezo Lautsprecher. Er basiert auf einem 32-bit ARM Cortex M0 Prozessor (16MHz).

Programmiert wird der Calliope auch über die USB-Schnittstelle. Es gibt kostenlose Editoren zum Programmieren, z. B. Open Roberta Lab von der Fraunhofer Gesellschaft. Oder sogar Apps für Smartphones.

Der Calliope erfreut sich großer Beliebtheit und ist sehr einfach zu programmieren. Er wird im IT-Unterricht ab Klasse 3 (ab einem Alter von 8 Jahren) eingesetzt.

5. BBC micro:bit



Der micro:bit ist dem Calliope sehr ähnlich. Er hat auch 25 LEDs, 2 Buttons, 3 Pins (für externe Beschaltung), Kompass, Bluetooth, einen USB-Anschluss und weitere Sensoren.

Man kann den micro:bit z.B. mit Python programmieren. Es gibt eine große Gemeinschaft mit Beispielen auch in anderen Sprachen.

6. Quellenangaben:

https://www.mikrocontroller.net

https://www.arduino.cc/

https://www.instructables.com

https://www.letscontrolit.com/wiki/index.php/ESP_Easy_web_interface

http://www.schatenseite.de/2016/05/17/lichtsteuerung-mit-esp8266-und-mqtt/

http://esp8266-server.de/I2C-Server.html

https://www.reichelt.de

http://www.netzmafia.de/skripten/hardware/Arduino/index.html

https://robokits.co.in/

https://www.bob3.org/de/

https://calliope.cc/

https://microbit.org/