4duino SensorKit 40 in 1

Sehr geehrter Kunde,

vielen Dank, dass Sie sich für unser 4duino Sensor Kit entschieden haben.

Das 4duino Sensor Kit wurde von uns neu aufgesetzt und die Platinen wurden speziell für die gängigsten Open-Source Plattformen entwickelt. Die hohe Kompatibilität zeichnet dieses Sensor Kit aus.

Die folgende Anleitung beinhaltet die technische Beschreibung der einzelnen Sensoren, die Pin-Belegung und die Code-Beispiele.

Das Sensor Kit beinhaltet nicht die Verbindungskabel, das Arduino Board oder sonstiges Zubehör.

Das von uns zu Verfügung gestellte Sensorkit ist für Anfänger und versierte Bastler geeignet, welche sich für Elektronik interessieren und eigene Ideen und Schaltungen entwerfen wollen. Der Einsatz im industriellen Umfeld wird nicht empfohlen.

Wir wünschen Ihnen viel Spaß mit den Sensoren und Ihren Experimenten.

Eine online Anleitung bzw. Bebilderung finden sie unter
http://www.allnet.de/40in1-sensorkit/

Ihr ALLNET Team

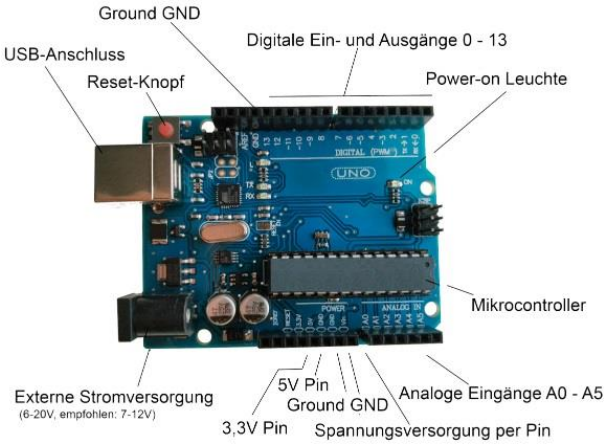
Sicherheitshinweise:

Beachten Sie unbedingt folgende Hinweise:

• Nicht die offenen Kontakte der Bauteile beruehren.
• Betreiben Sie die Bauteile niemals in der Naehe von Waermequellen.
• Stellen Sie das Set niemals auf Oberflaechen, die waermeempfindlich sind.
• Schuetzen Sie die Bauteile vor Naesse, Staub, Fluessigkeiten und Daempfen.
• Verwenden Sie das Geraet nicht in Feuchtraeumen und keinesfalls in explosionsgefaehrdeten Bereichen.
• Verwenden Sie zur Reinigung keine loesungsmittelhaltigen Putzmittel, sondern lediglich ein weiches, trockenes Antistatik Tuch.
• Eine Reparatur darf nur durch geschultes, autorisiertes Personal durchgefuehrt werden.
• Bei nicht bestimmungsgemaeßem Gebrauch ist eine Haftung durch ALLNET® ausgeschlossen.

Der bestimmungsgemaeße Gebrauch der Bauteile und der 4duino Platine sieht vor, eine Entwicklungsumgebung fuer Tests und wissenschaftliche Schaltungen zu bieten. Die Bauteile sind nicht fuer den Einbau in Geraete gedacht bzw. damit Geraete fuer den Verbraucher zu bauen. Die ALLNET uebernimmt keine Haftung fuer solche Verwendungszwecke.

Layout:



Kurzanleitung:

Software:

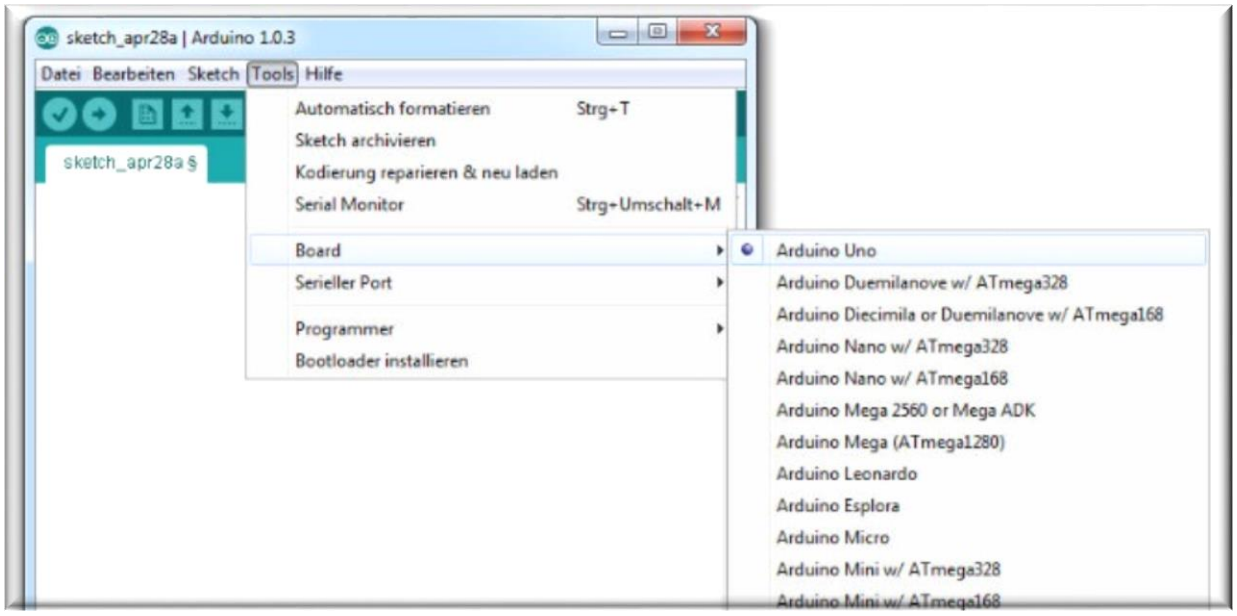
Die Software, mit welcher der UNO programmiert wird, ist open-Source-Software und kann auf www.arduino.org kostenlos heruntergeladen werden. Diese Software wird fuer die Erstellung der Programme benoetigt, die der Mikrocontroller spaeter ausfuehren soll. Diese Programme werden „Sketch'' genannt und sind teilweise als Beispiele in der Software integriert.

Installation:

Es muss nacheinander die Arduino-Software und der USB-Treiber fuer das Board installiert werden.

Installation und Einstellung der Arduino-Software:

Software von www.arduino.cc downloaden und auf dem PC installieren. Nach der Installation, wir im Softwareordner die Datei arduino.exe gestartet. Folgende Einstellungen muessen beachtet werden. Es muss das richtige Board ausgewaehlt werden, dass man anschließen moechte.
Das „4duino Uno'' Board muessen Sie als „Arduino Uno'' auswaehlen.

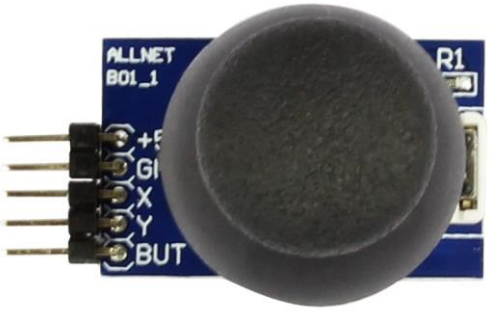


Es muss der richtige „Serial-Port" ausgewaehlt werden, damit der PC weiß, an welchem USB Anschluss das Board angeschlossen ist. Dies ist jedoch nur moeglich, wenn der Treiber richtig installiert wurde. Das kann folgendermaßen geprueft werden:
Wenn der Arduino noch nicht am PC angeschlossen ist, sehen sie in der Software unter „Serial Ports". mehrere Ports z.B. (COM1 / COM2 / COM3 / …) Wenn das Board richtig installiert und angeschlossen ist, waehlen Sie den neuen Port mit einer hoeheren Portzahl.

Inhaltsverzeichnis Sensorenübersicht

Sensorenübersicht

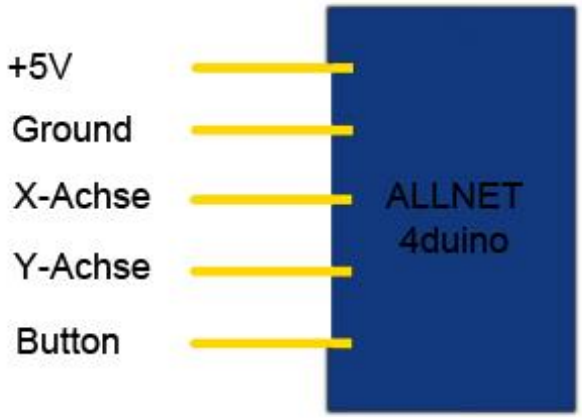
B01 Joystick Modul (XY-Achsen)



Kurzbeschreibung / Technische Daten

In diesem Programm werden die Werte des Joysticks (X und Y Achse sowie den Knopf (Z)) 10x pro Sekunde ausgelesen und übder die serielle Schnittstelle ausgegeben.

Pin-Belegung



Codebeispiel

```
// ALLNET Joystick B01
// Information http://www.allnet.de

//Deklarieren der benötigten Variablen
int JoyStick_X = A0;
int JoyStick_Y = A1;
int JoyStick_Button = 3;

//einmalig ausgeführte SetUp Befehle
void setup ()
{
  //Zuweisen der Pin Funktion
  pinMode (JoyStick_X, INPUT);
  pinMode (JoyStick_Y, INPUT);
  pinMode (JoyStick_Button, INPUT_PULLUP);

  //Starten der seriellen Übertragung
  Serial.begin (9600);
}

//dauerhaft wiederholte Hauptschleife
void loop ()
{
```

B02 5V Relay Modul

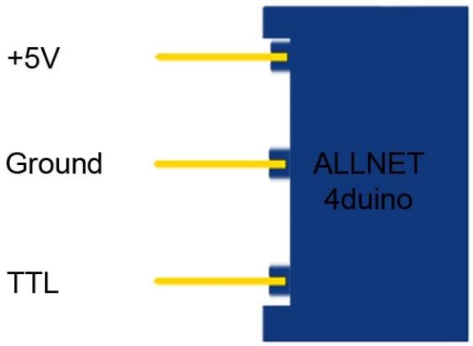
Kurzbeschreibung / Technische Daten

Spannungsbereich: 250VAC / 10A | 30VDC / 10A
Relais zum Schalten von höherer Spannungen mittels eines 5V Ausgangs.
Das Relais besitzt folgende Ausgänge

- "NC" für "normally closed", bedeutet dass dieser Durchgang ohne elektrische Umschaltung am Relais standardmäßig kurzgeschlossen ist.
- "NO" für "normally open", bedeutet dass dieser Durchgang ohne elektrische Umschaltung am Relais standardmäßig offen bzw. getrennt ist.

normally closed: Ausgangszustand geschlossen, Strom fließt
normally open: Ausgangszustand geöffnet, kein Stromfluss

Pin-Belegung



Codebeispiel

```
// ALLNET 5V Relay Modul B02
// Information http://www.allnet.de

//Deklarieren der benötigten Variablen
int releyPin = 10;

//einmalig ausgeführte SetUp Befehle
void setup ()
{
  //Zuweisen der Pin Funktion
  pinMode (releyPin, OUTPUT);
}

//dauerhaft wiederholte Hauptschleife
void loop ()
{
  //Anlegen von 5V (digital HIGH) an den releyPin
  digitalWrite (releyPin, HIGH);
  //Pause
  delay (3000);
  //Anlegen von 0V (digital LOW) an den releyPin
  digitalWrite (releyPin, LOW);
  //Pause
  delay (3000);
}
```

B03 Mikrofon Sensor Modul

Kurzbeschreibung / Technische Daten

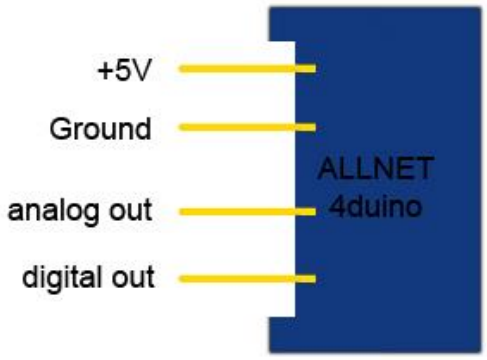
In dieser Schaltung wird das Mikrofon ausgelesen und der ausgelesene Wert über die serielle Schnittstelle ausgegeben.

Analoger Ausgang: Direktes Mikrofon-Signal als Spannungspegel

LED1: Zeigt an, dass der Sensor mit Spannung versorgt ist

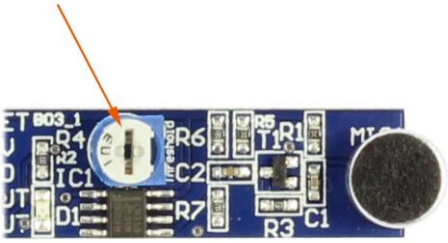
LED2: Zeigt an, dass ein Magnetfeld detektiert wurde

Pin-Belegung



Funktionsweise des Sensors

Der Sound Sensor gibt keine festen bzw. relativen Werte zurück. Er verstärkt das Signal abhängig vom eingestellten Widerstand am Drehpotentiometer und leitet es auf den analogen Ausgang des Moduls.



Codebeispiel

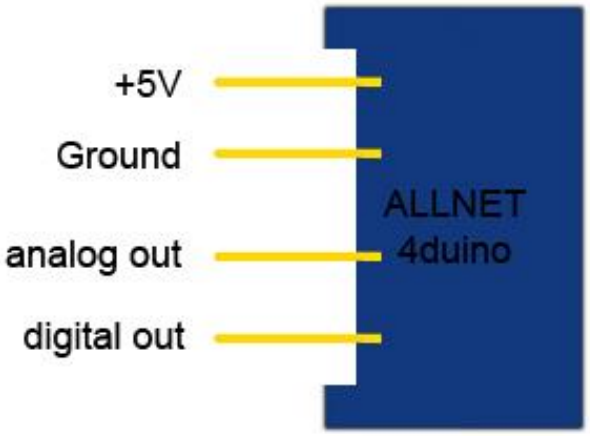
```
// ALLNET Mikrofon Sensor Modul B03
// Information http://www.allnet.de


int sensorPin = A0; // select the input pin for the potentiometer
int sensorValue = 0; // variable to store the value coming from the sensor

void setup () {
  pinMode (ledPin, OUTPUT);
  Serial.begin (9600);
}

void loop () {
  sensorValue = analogRead (sensorPin);
  Serial.println (sensorValue, DEC);
}
```

B04 IR Optical Detection / IRErkennung

Kurzbeschreibung / Technische Daten

Bei dieser Schaltung werden die Werte des IR-Sensors über die Serielle Schnittstelle ausgegeben.

Pin-Belegung



Funktionsweise des Sensors

Das Modul besteht aus Sensoreinheit bzw. Empfangseinheit Diode, dem Potentiometer zur Einstellung der Empfindlichkeit und der Ausgabe über die Pins.

Codebeispiel

Das Programm liest den aktuellen Spannungswert aus, der am analogen Ausgang gemessen werden kann, und gibt diesen auf der seriellen Schnittstelle aus.

Zudem wird ebenfalls der Zustand des digitalen Pins in der Konsole angegeben, was bedeutet ob der Grenzwert unterschritten wurde oder nicht.

```
// ALLNET IR Detection / IR Erkennung B04
// Information http://www.allnet.de

int a,b,c;
void setup() {
Serial.begin(9600);
pinMode(6,OUTPUT);
}

void loop() {
  digitalWrite(6,HIGH);
  delayMicroseconds(500);
  a=analogRead(A3);
  digitalWrite(6,LOW);
  delayMicroseconds(500);
  b=analogRead(A3);
  c=a-b;
  Serial.println(c);
}
```

B05 Flame Sensor / Flammensensor

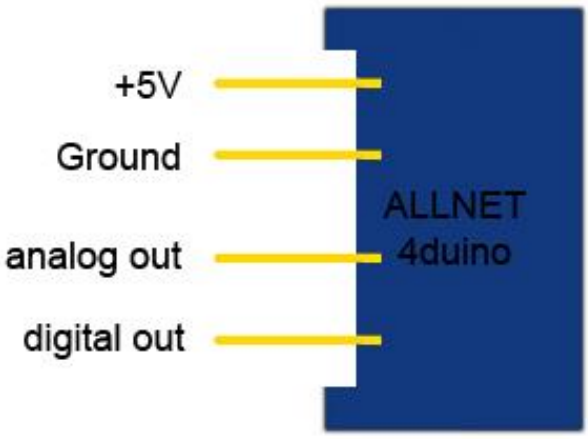
Kurzbeschreibung / Technische Daten

Diese Schaltung liest den analogen Flamesensor aus und gibt die Werte über Seriell aus. Weitere Bemerkungen stehen im Sourcecode.

LED1: Zeigt an, dass der Sensor mit Spannung versorgt ist

LED2: Zeigt an, dass eine Quelle erkannt wurde

Pin-Belegung



Funktionsweise des Sensors

Der Flammensensor ist sehr empfindlich auf IR-Wellenlänge bei 760 nm ~ 1100 nm Licht.

Analogausgang (A0): Echtzeit-Ausgangsspannungssignal am thermischen Widerstand.

Digitalausgang (D0): Wenn die Temperatur eine bestimmte Schwelle erreicht, kann der Ausgang high und low über Potentiometer eingestellt werden.

Codebeispiel

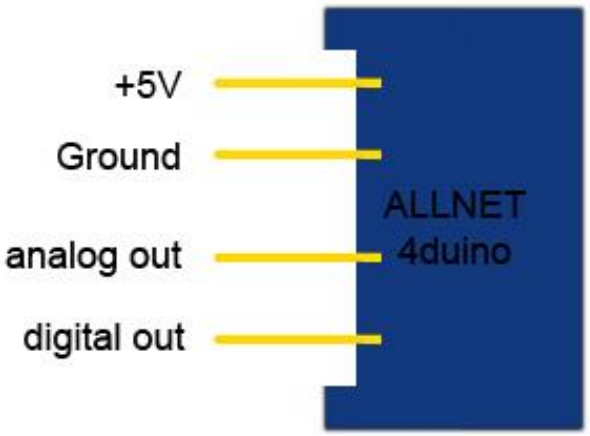
```
// ALLNET Flame Sensor / Flammensensor B05
// Information http://www.allnet.de


const int sensorMin = 0;     // sensor minimum
const int sensorMax = 1024;  // sensor maximum

void setup() {
  Serial.begin(9600);
}

void loop() {
  int sensorReading = analogRead(A0);
  int range = map(sensorReading, sensorMin, sensorMax, 0, 3);

  switch (range) {
  case 0:    // A fire closer than 1.5 feet away.
    Serial.println("** Close Fire **");
    break;
  case 1:    // A fire between 1-3 feet away.
    Serial.println("** Distant Fire **");
    break;
  case 2:    // No fire detected.
    Serial.println("No Fire");
    break;
  }
  delay(1);  // delay between reads
}------------------------------");

  //Pause
  delay (200);
}
```

B06 Hall TTL Sensor

Kurzbeschreibung / Technische Daten

Ein Hallsensor reagiert auf magnetische Felder und setzt sie in elektrische Impulse um.
Hallsensoren dienen dazu, berührungslos Messungen von Magnetfeldern vorzunehmen

Pin-Belegung



Funktionsweise des Sensors

Der Magnetische Hall Sensor reagiert auf ein Magnetfeld und je nachdem wie dieses gepolt ist (+ / -) reagiert der Sensor.

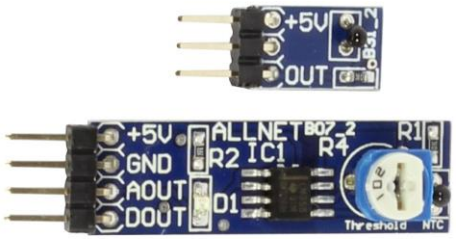
Codebeispiel

Das Programm liest den aktuellen Spannungswert aus, der am analogen Ausgang gemessen werden kann, und gibt diesen auf der seriellen Schnittstelle aus.

Zudem wird ebenfalls der Zustand des digitalen Pins in der Konsole angegeben, was bedeutet ob der Grenzwert unterschritten wurde oder nicht.

```
// ALLNET Hall TTL Sensor B06
// Information http://www.allnet.de


int Led = 2 ; // define LED Interface
int buttonpin = 3; // define the linear Hall magnetic sensor interface
int val ; // define numeric variables val
void setup () {
  pinMode (Led, OUTPUT) ; // define LED as output interface
  pinMode (buttonpin, INPUT) ; // define linear Hall magnetic sensor output interface
}
void loop ()
{
  val = digitalRead (buttonpin) ; // digital interface will be assigned a value of 3 to
read val
  if (val == HIGH) { // When the linear Hall sensor detects a magnetic signal, LED
flashes
    digitalWrite (Led, HIGH);
    Serial.println("Magnetic signal detected");
  } else {
    digitalWrite (Led, LOW);
    Serial.println("No magnetic signal detected");
  }
}
```

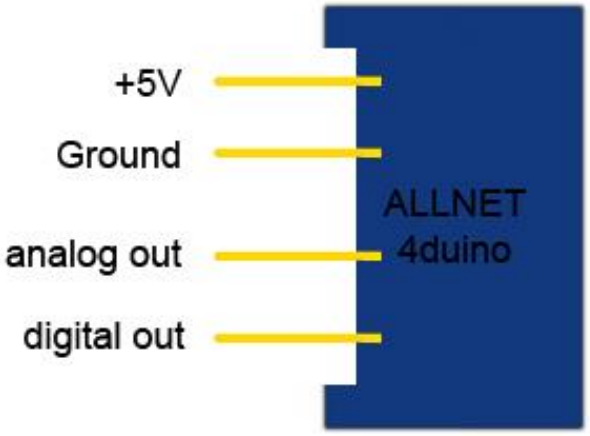
B07/B31 NTC Threshold TTL/ NTC 10k



Kurzbeschreibung / Technische Daten

Der Sensor ist ein Thermischer Resistor mit dem die Temperatur bestimmt werden kann. Die analogen Messwerte werden über die serielle Schnittstelle an den Rechner ausgegeben.

Pin-Belegung



Funktionsweise des Sensors

Ein NTC Thermistor ist ein temperaturabhängiger Widerstand, der im Gegensatz zu einem DHT11 Temperatursensor bei höheren Temperaturen eingesetzt werden kann.

Temperaturbereich:

Min/Max	-55° bis 150 °C
Toleranz	ca. ±0,4°

Codebeispiel

```
// ALLNET NTC Threshold TTL/NTC 10k B07/B31
// Information http://www.allnet.de

#define ABSZERO 273.15
#define MAXANALOGREAD 1023.0
#define ANALOGPIN A0

float temperature_NTCB(float T0, float R0, float B,
float RV, float VA_VB) {
  T0+=ABSZERO;  // umwandeln Celsius in absolute
Temperatur
  float RN=RV*VA_VB / (1-VA_VB); // aktueller
Widerstand des NTC
  return T0 * B / (B + T0 * log(RN / R0))-ABSZERO;
}

void setup() {
  Serial.begin(9600);
}

void loop() {
  float T0=25;    // Nenntemperatur des NTC-
Widerstands in °C
  float R0=10000; // Nennwiderstand des NTC-Sensors
in Ohm
  float B=3976;   // Materialkonstante B
  float RV=10000; // Vorwiderstand in Ohm
  float temp;
  int aValue=analogRead(ANALOGPIN);
  // Berechnen bei bekannter Materialkonstante B;
  temp=temperature_NTCB(T0, R0, B, RV,
aValue/MAXANALOGREAD);
  Serial.print("NTCB:
");Serial.print(temp);Serial.println(" C");
  delay(500);
}
```

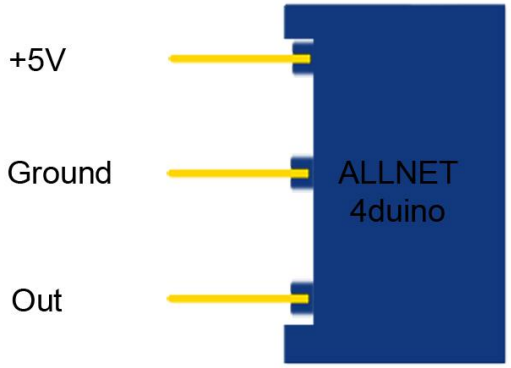
B08 Touch Sensor

Kurzbeschreibung / Technische Daten

Der Sensor gibt ein Signal am OUT Pin aus, wenn er an den beiden silbernen Kontakten berührt wird.

Das Programm gitbt eine Touch Berührung des Sensors an den Seriellen Monitor weiter.

Pin-Belegung



Codebeispiel

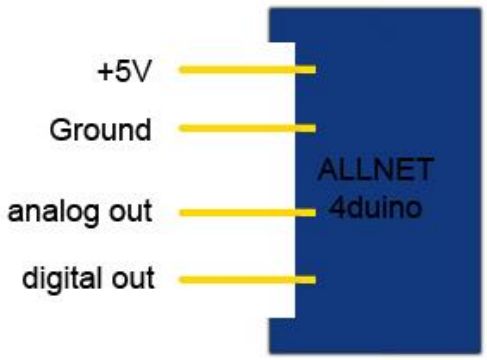
```
// ALLNET Touch Sensor B08
// Information http://www.allnet.de


//Deklarieren der benötigten Variablen
int Digital_Eingang = 3;

//einmalig ausgeführte SetUp Befehle
void setup ()
{
  //Zuweisen der Pin Funktion
  pinMode (Digital_Eingang, INPUT);

  //Starten der seriellen Übertragung
  Serial.begin (9600);
}

//dauerhaft wiederholte Hauptschleife
void loop ()
{
  //Wenn der Wert des Digital_Eingang 1 entspricht
  if (digitalRead (Digital_Eingang) == 1)
  {
    //Dann ist eine Berührung erkannt und dies wird
als Meldung ausgegeben
    Serial.println ("Berührung erkannt");
  }

  //Pause
  delay (200);
}
```

B09 RGB LED

Kurzbeschreibung / Technische Daten

LED-Modul welche eine rote, blaue und grüne LED beinhaltet. Diese sind mittels gemeinsamer Kathode miteinander verbunden.

Pin-Belegung

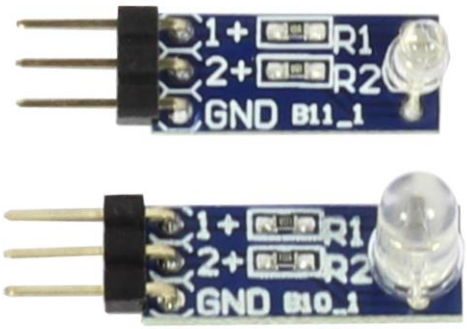


Codebeispiel

Dieses Codebeispiel zeigt auf, wie die integrierten LEDs mittels eines definierbaren Ausgangspins abwechselnd, in 3 Sekunden Takt, angeschaltet werden können.

```
// ALLNET RGB LED B09
// Information http://www.allnet.de


int led_blue = 5;
int led_green = 6;
int led_red = 7;

void setup() {
  Serial.begin(9600);
  pinMode(led_blue, OUTPUT);
  pinMode(led_green, OUTPUT);
  pinMode(led_red, OUTPUT);
}

void loop() {
  digitalWrite(led_blue, HIGH);
  delay(500);
  digitalWrite(led_blue, LOW);
  digitalWrite(led_green, HIGH);
  delay(500);
  digitalWrite(led_green, LOW);
  digitalWrite(led_red, HIGH);
  delay(500);
  digitalWrite(led_red, LOW);
  digitalWrite(led_blue, HIGH);
  digitalWrite(led_green, HIGH);
  delay(500);
  digitalWrite(led_blue, LOW);
  digitalWrite(led_red, HIGH);
  delay(500);
  digitalWrite(led_green, LOW);
  digitalWrite(led_blue, HIGH);
  delay(500);
  digitalWrite(led_red, LOW);
  digitalWrite(led_blue, LOW);
  delay(500);
}
```

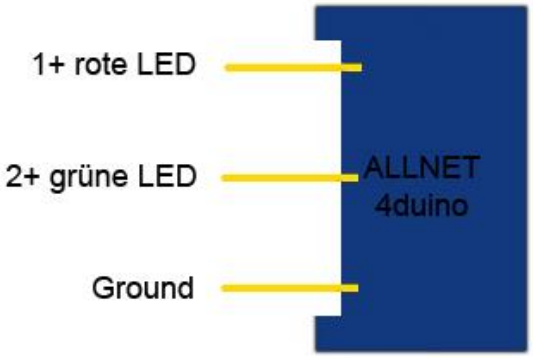
B10/B11 Zweifarbige LED 5/3mm



Kurzbeschreibung / Technische Daten

LED-Modul welche eine rote und grüne LED beinhaltet. Diese sind mittels gemeinsamer Kathode miteinander verbunden.

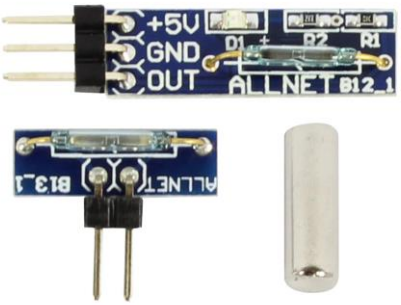
Pin-Belegung



Codebeispiel

```
// ALLNET Zweifarbige LED 3/5mm B10/B11
// Information http://www.allnet.de

int B10_1 = 3;
int B10_2 = 2;
int B11_1 = 5;
int B11_2 = 4;

void setup() {
  Serial.begin(9600);
  pinMode(B10_1, OUTPUT);
  pinMode(B10_2, OUTPUT);
  pinMode(B11_1, OUTPUT);
  pinMode(B11_2, OUTPUT);
}

void loop() {
  digitalWrite(B10_1, HIGH);
  digitalWrite(B11_1, HIGH);
  delay(500);
  digitalWrite(B10_1, LOW);
  digitalWrite(B11_1, LOW);
  digitalWrite(B10_2, HIGH);
  digitalWrite(B11_2, HIGH);
  delay(500);
  digitalWrite(B10_2, LOW);
  digitalWrite(B11_2, LOW);
  delay(500);
  digitalWrite(B10_1, HIGH);
  digitalWrite(B11_1, HIGH);
  digitalWrite(B10_2, HIGH);
  digitalWrite(B11_2, HIGH);
  delay(500);
  digitalWrite(B10_1, LOW);
  digitalWrite(B11_1, LOW);
  digitalWrite(B10_2, LOW);
  digitalWrite(B11_2, LOW);
  delay(500);
}
```

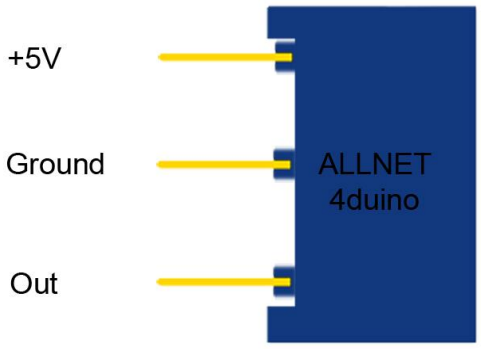
B12/B13 Reed Sensor/Magnet Kontakt Sensor



Kurzbeschreibung / Technische Daten

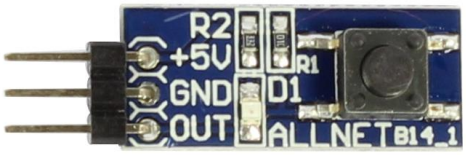
Wird ein Magnetfeld detektiert, so werden die beiden Pins kurzgeschlossen bzw. ein Signal am PUT Pin angelegt.

Pin-Belegung



Codebeispiel

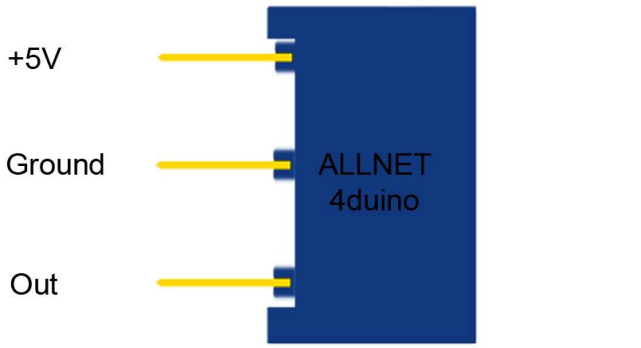
```
// ALLNET Reed Sensor/Magnet Kontakt Sensor B12/B13
// Information http://www.allnet.de


int pinSwitch = 2;
int pinLed = 3;
int StatoSwitch = 0;

void setup() {
  Serial.begin(9600);
  pinMode(pinLed, OUTPUT);
  pinMode(pinSwitch, INPUT);
}

void loop() {
  StatoSwitch = digitalRead(pinSwitch);
  if (StatoSwitch == HIGH) {
    digitalWrite(pinLed, HIGH);
    Serial.println("Switch : on");
  } else {
    digitalWrite(pinLed, LOW);
    Serial.println("Switch : off");
  }
}
```



Kurzbeschreibung / Technische Daten

Beim Drücken des Tasters, wird ein Signal am OUT Pin ausgegeben.

Pin-Belegung



Codebeispiel

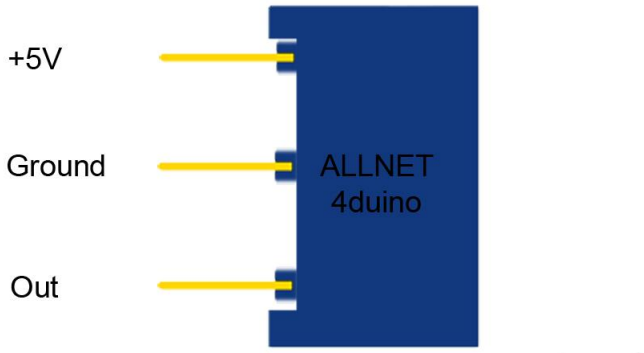
```
// ALLNET Button/Taster B14
// Information http://www.allnet.de


//Deklarieren der benötigten Variablen
int Digital_Eingang = 3;

//einmalig ausgeführte SetUp Befehle
void setup ()
{
  //Zuweisen der Pin Funktion
  pinMode (Digital_Eingang, INPUT);

  //Starten der seriellen Übertragung
  Serial.begin (9600);
}

//dauerhaft wiederholte Hauptschleife
void loop ()
{
  //Wenn der Wert des Digital_Eingang 1 entspricht
  if (digitalRead (Digital_Eingang) == 1)
  {
    //Dann ist der Taster gedrückt und dies wird als Meldung ausgegeben
    Serial.println ("Taster gedrückt");
  }

  //Pause
  delay (200);
}
```

B15 Tilt Sensor/Neigungssensor

Kurzbeschreibung / Technische Daten

Je nach Neigung, schließt ein Schalter die Eingangspins kurz.

Pin-Belegung



Codebeispiel

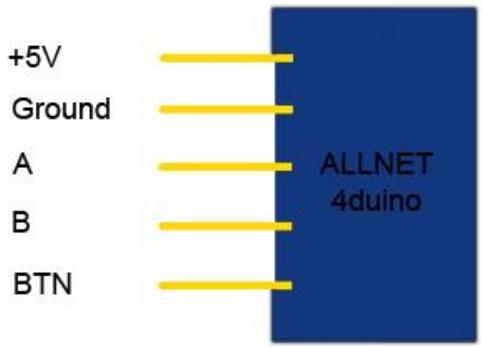
```
// ALLNET Tilt Sensor/Neugungssensor B15
// Information http://www.allnet.de


//Deklarieren der benötigten Variablen
int Digital_Eingang = 3;

//einmalig ausgeführte SetUp Befehle
void setup ()
{
  //Zuweisen der Pin Funktion
  pinMode (Digital_Eingang, INPUT);

  //Starten der seriellen Übertragung
  Serial.begin (9600);
}

//dauerhaft wiederholte Hauptschleife
void loop ()
{
  //Wenn der Wert des Digital_Eingang 1 entspricht
  if (digitalRead (Digital_Eingang) == 1)
  {
    //Dann ist eine Neigung und dies wird als Meldung ausgegeben
    Serial.println ("Taster gedrückt");
  }

  //Pause
  delay (200);
}
```

B16 Rotary Encoder / Kodierter Drehschalter

Kurzbeschreibung / Technische Daten

In dieser Schaltung wird ein Drehgeber mit Druckknopf ausgelesen. Ausgabe der Werte erfolgt über die serielle Schnittstelle.

Pin-Belegung



Codebeispiel

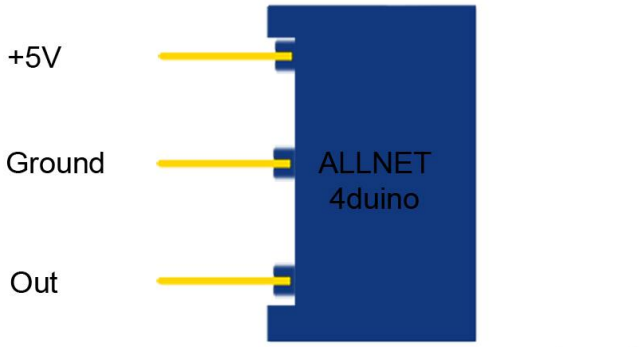
```
// ALLNET Rotary Encoder/Kodierter Drehschalter B16
// Information http://www.allnet.de


/* Read Quadrature Encoder
  * Connect Encoder to Pins encoder0PinA,
encoder0PinB, and +5V.
  */


 int val;
 int encoder0PinA = 7;
 int encoder0PinB = 6;
 int encoder0PinC = 5;
 int encoder0Pos = 0;
 int encoder0PinALast = LOW;
 int n = LOW;


 void setup() {
   pinMode (encoder0PinA,INPUT);
   pinMode (encoder0PinB,INPUT);
   pinMode (encoder0PinC,INPUT);
   Serial.begin (9600);
 }

 void loop() {
   n = digitalRead(encoder0PinA);
   if ((encoder0PinALast == LOW) && (n == HIGH)) {
     if (digitalRead(encoder0PinB) == LOW) {
       encoder0Pos--;
     } else {
       encoder0Pos++;
     }
        Serial.print ("Encoder: ");
   Serial.print (encoder0Pos);
   Serial.print (", ");
    int button = digitalRead(encoder0PinC);
   Serial.print ("Button pressed: ");
   Serial.print (button);
   Serial.println (" ");

   }


   encoder0PinALast = n;


 }
```

B18 Light Barrier / Lichtschranke

Kurzbeschreibung / Technische Daten

In dieser Schaltung wird eine Lichtschranke als Schalter benutzt um eine LED zu schalten.

Pin-Belegung



Codebeispiel

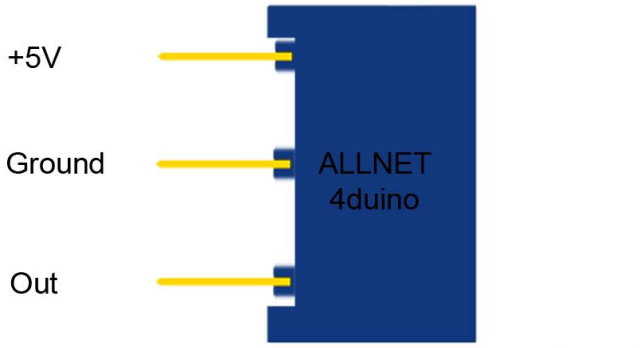
```
// ALLNET Tilt Light Barrier/Lichtschranke B18
// Information http://www.allnet.de


int sensorPin = 3;
int ledPin = 2;

void setup() {
  Serial.begin(9600);
  pinMode(sensorPin, INPUT);
  pinMode(ledPin, OUTPUT);
}
void loop() {
  int val = digitalRead(sensorPin);
  Serial.println(val);
  if ( val == HIGH) {
    digitalWrite(ledPin, HIGH);
  } else {
    digitalWrite(ledPin, LOW);
  }
}
```

B19 Potentiometer / Analog Hall

Kurzbeschreibung / Technische Daten

Der Widerstand des Potentiometers ändert sich je nach Stellung des Drehschalters. Hierdurch ändert sich die am Sensorpin anliegende Spannung. Analog hiezu funktioniert auch der Analog Hall Sensor. Hier ändert sich der Wiederstand jedoch nach dem magnetischen Umfeld.

Pin-Belegung



Codebeispiel

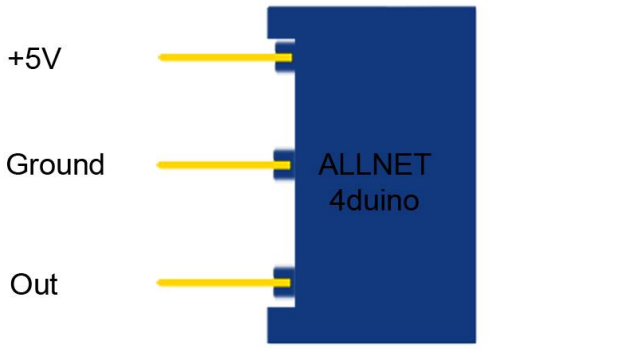
```
// ALLNET Potentiometer/Analog Hall B19
// Information http://www.allnet.de

//Deklarieren der benötigten Variablen
int Analog_Eingang = A0;

//einmalig ausgeführte SetUp Befehle
void setup ()
{
  //Zuweisen der Pin Funktion
  pinMode (Analog_Eingang, INPUT);

  //Starten der seriellen Übertragung
  Serial.begin (9600);
}

//dauerhaft wiederholte Hauptschleife
void loop ()
{
  //Deklarieren einem temporären Zwischenspeicher
  float analog;

  //Aktueller Wert wird ausgelesen, auf den Spannungswert konvertiert...
  analog = analogRead (Analog_Eingang) * (5.0 / 1023.0);

  //... und an dieser Stelle ausgegeben
  Serial.print ("Analoger Spannungswert:");
  Serial.print (analog);
  Serial.print (" V, ");
  Serial.println ("---------------------------------");

  //Pause
  delay (200);
```

B20 Temperatur Sensor Modul

Kurzbeschreibung / Technische Daten

Kommunikationsprotokoll: 1-Wire
Liefert eine 9 -12 Bit genaue Temperaturmessung über den 1-Wire Pin

Pin-Belegung



Codebeispiel

Für das folgende Codebeispiel werden zwei zusätzliche Libraries benötigt:

- [OneWire Library]
- [Dallas Temperature Control Library]

Beide Libraries können direkt in der Arduino IDE installiert werden. Sie stehen im Library Manager zum download bereit.

Anschlussbelegung an Arduino::

Sensor +V = [Pin 5V]
Sensor GND = [Pin GND]
Sensor DQ = [Pin Out]

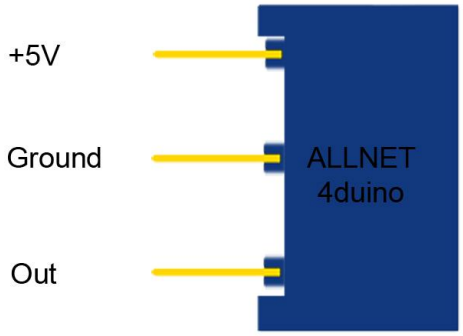
```
// ALLNET Temperatur Sensor Modul B20
// Information http://www.allnet.de


//Benötigte Libraries werden importiert
#include <DallasTemperature.h>
#include <OneWire.h>

//Hier wird der Eingangs-Pin deklariert, an dem das
Sensor-Modul angeschlossen ist
#define B20_Signal_PIN 4

//Libraries werden konfiguriert
OneWire oneWire(B20_Signal_PIN);
DallasTemperature sensors(&oneWire);

//einmalig ausgeführte SetUp Befehle
void setup() {

  //Starten der seriellen Übertragung
  Serial.begin(9600);
  Serial.println("B20 Temperaturmessung");

  //Sensor wird initialisiert
  sensors.begin();
}

//dauerhaft wiederholte Hauptschleife
void loop()
{
  //Temperaturmessung wird gestartet...
  sensors.requestTemperatures();

  //... und gemessene Temperatur ausgeben
  Serial.print("Temperatur: ");
  Serial.print(sensors.getTempCByIndex(0));
  Serial.write(176); // UniCode-Angabe eines char-Symbols für das "°-Symbol"
  Serial.println("C");

  //Pause
  delay (1000);
}
```

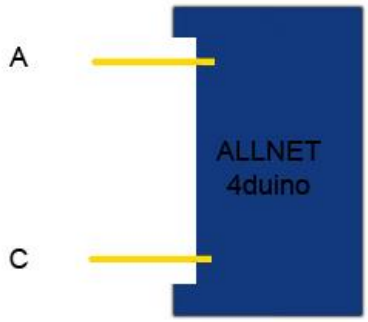
B21 IR LED / Infrarot LED

Kurzbeschreibung / Technische Daten

Eine Leuchtdiode, die im infraroten Bereich ausstrahlt.

Pin-Belegung





Codebeispiel

Mithilfe der beiden Sensormodule B21 und B22 lässt sich ein Infrarot-Fernbedienung + Infrarot Receiver System aufbauen. Hierzu werden neben den zwei Modulen auch zwei einzelne Arduinos benötigt. Der eine fungiert hierbei dann als Sender und der andere empfängt die Signale und gibt diese dann in der seriellen Konsole aus.

Für das folgende Codebeispiel wird eine zusätzliche Library benötigt:

Download Library hier

Die Library kann über den Manager der Arduino IDE installiert werden.

Code für den Empfänger (B22):

Anschlussbelegung an Arduino::

Sensor V+ = [Pin 5V]
Sensor Signal = [Pin 11]
Sensor GND = [Pin GND]

```
// ALLNET IR LED / Infrarot LED B21
// Information http://www.allnet.de

//Arduino-IRremote Iibrary wird hinzugefuegt
#include <IRremote.h>

//Deklarieren der benötigten Variablen
int RECV_PIN = 11;

// Arduino-IRremote Library wird initialisiert
IRrecv irrecv(RECV_PIN);
decode_results results;

//einmalig ausgeführte SetUp Befehle
void setup()
{
//Starten der seriellen Übertragung
  Serial.begin(9600);

  //Infrarot-Receiver wird gestartet
  irrecv.enableIRIn();
}

//dauerhaft wiederholte Hauptschleife
void loop()
{
  //Wenn am Recveiver ein Signal eingegangen ist
  if (irrecv.decode(&results))
  {
    //Dann gebe das Empfangene in der seriellen Konsole aus
    Serial.println(results.value, HEX);

    //Fortsetzen des Empfangens
    irrecv.resume();
  }
}
```

Code für den Sender (B21):

Anschlussbelegung an Arduino::

LED Signal = [Pin 3]
LED GND = [Pin GND]

```
// ALLNET IR LED / Infrarot LED B21
// Information http://www.allnet.de

//Arduino-IRremote Library wird hinzugefügt...
#include <IRremote.h>

//...und hier initialisiert
IRsend irsend;

// Die Einstellungen für den Ausgang werden von der Library übernommen
// Die entsprechenden Ausgänge unterscheiden sich je nach verwendeten Arduino
// Arduino UNO:  Ausgang = D3
// Arduino MEGA: Ausgang = D9
// Eine komplette Auflistung der entsprechenden Ausgänge finden Sie unter
// http://z3t0.github.io/Arduino-IRremote/

//einmalig ausgeführte SetUp Befehle
void setup()
{
}

//dauerhaft wiederholte Hauptschleife
void loop()
{
  // Der Sender verschickt in diesem Beispiel das Signal A90 (in hexdezimaler Form) in
der Kodierung "RC5"
  // Dieses wird dreimal hintereinander gesendet und danach eine Pause für 5 Sekunden
eingelegt
  for (int i = 0; i < 3; i++)
  {
    // [0xA90] zu versendetes Signal | [12] Bit-Länge des zu versendeten Signals (hex A90
= 1010 1001 0000)
    irsend.sendRC5(0xA90, 12);
    delay(40);
  }
}
```

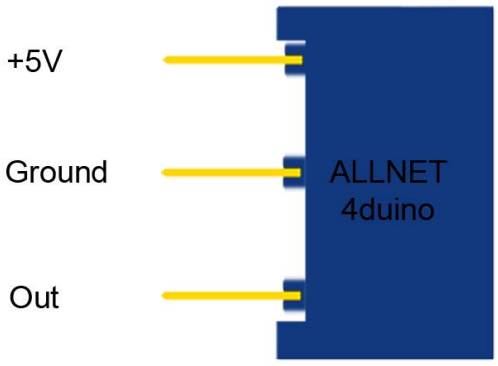
B22 IR Receiver 38KHz / IR Empfanger 38KHz

Kurzbeschreibung / Technische Daten

Trägerfrequenz: 38kHz Kann Infrarotsignale empfangen und gibt diese am Signalausgang als digitale Abfolge aus.

Zusätzlich blinkt die auf dem Modul integrierte LED kurz auf, wenn ein Infrarot-Signal detektiert wurde.

Pin-Belegung



Codebeispiel

Mithilfe der beiden Sensormodule B21 und B22 lässt sich ein Infrarot-Fernbedienung + Infrarot Receiver System aufbauen. Hierzu werden neben den zwei Modulen auch zwei einzelne Arduinos benötigt. Der eine fungiert hierbei dann als Sender und der andere empfängt die Signale und gibt diese dann in der seriellen Konsole aus.

Für das folgende Codebeispiel wird eine zusätzliche Library benötigt:

Download Library hier

Code für den Empfänger (B22):

Anschlussbelegung an Arduino::

Sensor V+ = [Pin 5V]
Sensor Ground = [Pin Ground]
Sensor Out = [Pin Digital]

```
//Starten der seriellen Übertragung
  Serial.begin(9600);

  //Infrarot-Receiver wird gestartet
  irrecv.enableIRIn();
}

//dauerhaft wiederholte Hauptschleife
void loop()
{
  //Wenn am Recveiver ein Signal eingegangen ist
  if (irrecv.decode(&results))
  {
    //Dann gebe das Empfangene in der seriellen Konsole aus
    Serial.println(results.value, HEX);

    //Fortsetzen des Empfangens
    irrecv.resume();
  }
}
```

```
// ALLNET IR Reciever 38mHz / IR Empfänger 38mHz B22
// Information http://www.allnet.de

//Arduino-IRremote Library wird hinzugefuegt
#include <IRremote.h>

//Deklarieren der benötigten Variablen
int RECV_PIN = 11;

// Arduino-IRremote Library wird initialisiert
IRrecv irrecv(RECV_PIN);
decode_results results;
//einmalig ausgeführte SetUp Befehle
void setup()
{

  //Starten der seriellen Übertragung
  Serial.begin(9600);

  //Infrarot-Receiver wird gestartet
  irrecv.enableIRIn();
}

//dauerhaft wiederholte Hauptschleife
void loop()
{
  //Wenn am Recveiver ein Signal eingegangen ist
  if (irrecv.decode(&results))
  {
    //Dann gebe das Empfangene in der seriellen Konsole aus
    Serial.println(results.value, HEX);

    //Fortsetzen des Empfangens
    irrecv.resume();
  }
}
```

Code für den Sender (B21):

Anschlussbelegung an Arduino::

LED +V = [Pin 3]
LED GND = [Pin GND]

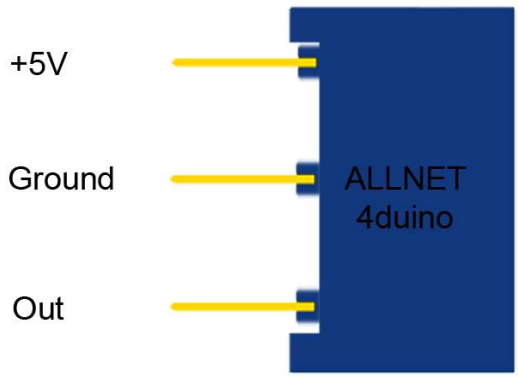
```
// ALLNET IR Reciever 38mHz / IR Empfänger 38mHz B22
// Information http://www.allnet.de

//Arduino-IRremote Library wird hinzugefügt...
#include <IRremote.h>

//...und hier initialisiert
IRsend irsend;

// Die Einstellungen für den Ausgang werden von der Library übernommen
// Die entsprechenden Ausgänge unterscheiden sich je nach verwendeten Arduino
// Arduino UNO:  Ausgang = D3
// Arduino MEGA: Ausgang = D9
// Eine komplette Auflistung der entsprechenden Ausgänge finden Sie unter
// http://z3t0.github.io/Arduino-IRremote/

//einmalig ausgeführte SetUp Befehle
void setup()
{
}

//dauerhaft wiederholte Hauptschleife
void loop()
{
  // Der Sender verschickt in diesem Beispiel das Signal A90 (in hexdezimaler Form) in
der Kodierung "RC5"
  // Dieses wird dreimal hintereinander gesendet und danach eine Pause für 5 Sekunden
eingelegt
  for (int i = 0; i < 3; i++)
  {
    // [0xA90] zu versendetes Signal | [12] Bit-Länge des zu versendeten Signals (hex A90
= 1010 1001 0000)
    irsend.sendRC5(0xA90, 12);
    delay(40);
  }
}
```

B23 Shock Sensor / Schocksensor

Kurzbeschreibung / Technische Daten

Der Sensor-Pin sendet ein Signal aus, sobald eine Erschütterung erkannt eird.

Pin-Belegung



Codebeispiel

Hier bei handelt es sich um ein Beispielprogramm, welches eine serielle Ausgabe auslöst, wenn eine Erschütterung erkannt wird.

Anschlussbelegung an Arduino::

Sensor +V = [Pin 5V]
Sensor GND = [Pin GND]
Sensor Digital = [Pin 3]

```
// ALLNET Tilt Sensor/Neugungssensor B15
// Information http://www.allnet.de


//Deklarieren der benötigten Variablen
int Digital_Eingang = 3;

//einmalig ausgeführte SetUp Befehle
void setup ()
{
  //Zuweisen der Pin Funktion
  pinMode (Digital_Eingang, INPUT);

  //Starten der seriellen Übertragung
  Serial.begin (9600);
}

//dauerhaft wiederholte Hauptschleife
void loop ()
{
  //Wenn der Wert des Digital_Eingang 1 entspricht
  if (digitalRead (Digital_Eingang) == 1)
  {
    //Dann ist eine Erschütterung erkannt und dies
wird als Meldung ausgegeben
    Serial.println ("Erschütterung erkant");
  }

  //Pause
  delay (200);
}
```

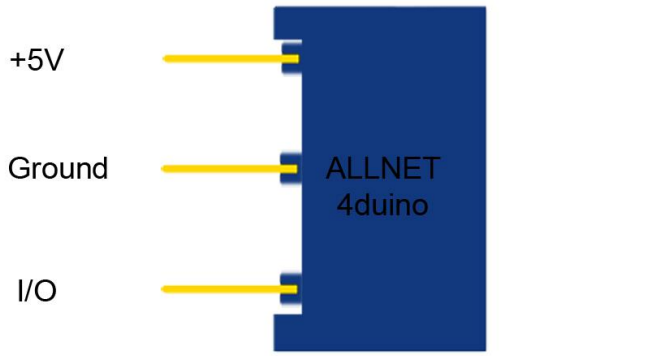
B24 Temperature & Humidity / Temp. & Feuchtigkeit

Kurzbeschreibung / Technische Daten

Chipsatz: DHT11 | Kommunikationsprotokoll: 1-Wire Messbereich Luftfeuchtigkeit: 20-90%RH Messbereich Temperatur: 0-50°C

Vorteile dieses Sensors ist die Kombination von Temperaturmessung und Luftfeuchtigkeitsmessung in einer kompakten Bauform - der Nachteil ist jedoch die geringe Abtastrate der Messung, so dass nur jede 2 Sekunden ein neues Messergebnis zur Verfügung steht - dieser Sensor ist somit sehr gut für Langzeit- Messungen geeignet.

Pin-Belegung



Codebeispiel

Hier bei handelt es sich um ein Beispielprogramm, welches den Wert des Sensors an den seriellen Monitor ausgibt.

Anschlussbelegung an Arduino::

Sensor +V	= [Pin 5V]
Sensor GND	= [Pin GND]
Sensor Digital	= [Pin 2]

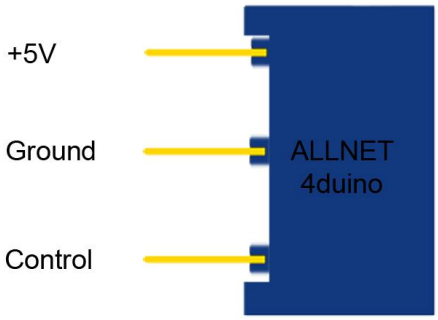
```
//Auslesen der Temperatur in Celsius
  float t = dht.readTemperature();
  //Auslesen der Temperatur in Farenheit
  float f = dht.readTemperature(true);

  //Prüft ob eine Messung fehlerhaft ist und bricht
in dem Fall vorzeitig ab
  //Eine Information darüber wird in der seriellen
Ausgabe bereitgestellt
  if (isnan(h) || isnan(t) || isnan(f))
  {
    Serial.println("Failed to read from DHT
sensor!");
    return;
  }

  //Berechnet den Heat Index in Farenheit
  float hif = dht.computeHeatIndex(f, h);
  //Berechnet den Heat Index in Celsius (isFahreheit
= false)
  float hic = dht.computeHeatIndex(t, h, false);

  //Ausgabe der Sensorwerte
  Serial.print("Luftfeuchtigkeit: ");
  Serial.print(h);
  Serial.print(" %\t");
  Serial.print("Temperatur: ");
  Serial.print(t);
  Serial.print(" *C ");
  Serial.print(f);
  Serial.print(" *F\t");
  Serial.print("Heat index: ");
  Serial.print(hic);
  Serial.print(" *C ");
  Serial.print(hif);
  Serial.println(" *F");
}
```

B25 1 Watt LED Module

Kurzbeschreibung / Technische Daten

Das Modul besitzt eine 1W LED, welche durch den Control Pin Ein bzw. Aus geschalten werden kann.

Pin-Belegung



Codebeispiel

Hier bei handelt es sich um ein Beispielprogramm, welches die 1W LED blinken lässt.

Anschlussbelegung an Arduino::

Sensor +V	= [Pin 5V]
Sensor GND	= [Pin GND]
Sensor Control	= [Pin 3]

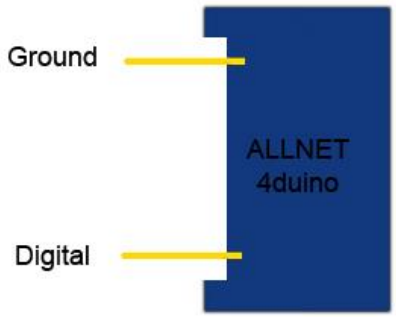
```
// ALLNET 1W LED Module B25
// Information http://www.allnet.de

//Deklarieren der benötigten Variablen
int LED_Eingang = 3;

//einmalig ausgeführte SetUp Befehle
void setup()
{
  //Zuweisen der Pin Funktion
  pinMode (LED_Eingang, INPUT);
}

//dauerhaft wiederholte Hauptschleife
void loop()
{
  //LED an
  digitalWrite (LED_Eingang, HIGH);

  //Pause
  delay(1000);

  //LED aus
  digitalWrite (LED_Eingang, LOW);

  //Pause
  delay(1000);
}
```

B26 Piezo Speaker

Kurzbeschreibung / Technische Daten

Der Piezo Speaker sendet einen Ton aus, sofern an die Pins Strom angelegt wird.

Pin-Belegung



Codebeispiel

Hier bei handelt es sich um ein Beispielprogramm, welches ein Alarmsignal sendet.

Anschlussbelegung an Arduino::

Sensor GND = [Pin GND]
Sensor Digital = [Pin 8]

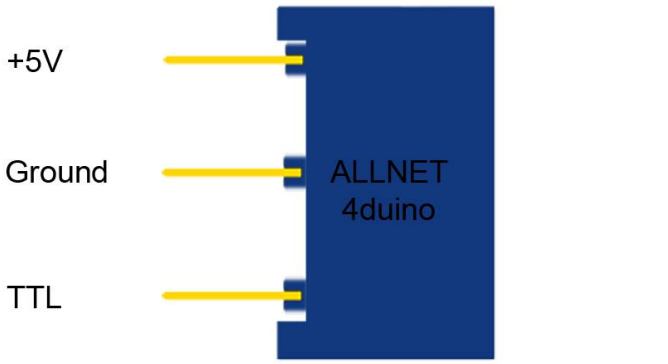
```
// ALLNET Piezo Speaker B26
// Information http://www.allnet.de


//einmalig ausgeführte SetUp Befehle
void setup()
{
}

//dauerhaft wiederholte Hauptschleife
void loop()
{
  //Piezo an Pin 8 aktivieren mit Ton 100
  tone(8,100);

  //Pause
  delay(1000);

  //Piezo an Pin 8 deaktivieren
  noTone(8);

  //Pause
  delay(1000);
}
```

B27 Buzzer

Kurzbeschreibung / Technische Daten

Der Buzzer sendet einen Ton aus, sofern ein Signal am TTL Pin anliegt.

Pin-Belegung



Codebeispiel

Hier bei handelt es sich um ein Beispielprogramm, welches ein Alarmsignal sendet.

Anschlussbelegung an Arduino::

Sensor +5V = [+5V]
Sensor GND = [GND]
Sensor Digital = [TTL]

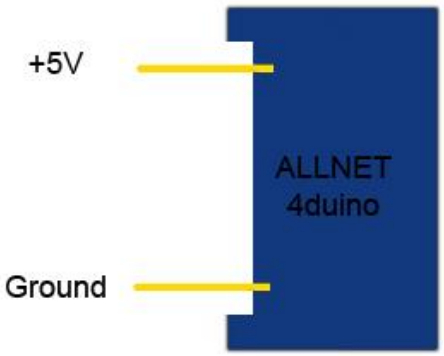
```
// ALLNET Buzzer B27
// Information http://www.allnet.de

//Deklarieren der benötigten Variablen
int Buzzer_Eingang = 3;

//einmalig ausgeführte SetUp Befehle
void setup()
{
  //Zuweisen der Pin Funktion
  pinMode (Buzzer_Eingang, INPUT);
}

//dauerhaft wiederholte Hauptschleife
void loop()
{
  //Buzzer an
  digitalWrite (Buzzer_Eingang, HIGH);

  //Pause
  delay(500);

  //Buzzer aus
  digitalWrite (Buzzer_Eingang, LOW);

  //Pause
  delay(500);
}
```

B28 Flash LED

Kurzbeschreibung / Technische Daten

Wird zwischen den beiden Pins eine Spannung angelegt, so fängt die LED an zu blinken.

Pin-Belegung



Codebeispiel

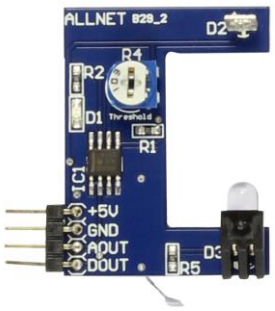
Hier bei handelt es sich um ein Beispielprogramm, welches ein Blinklicht immitiert.

Anschlussbelegung an Arduino::

Sensor +5V = [+5V]
Sensor Ground = [Ground]

```
// ALLNET Flash LED B28
// Information http://www.allnet.de

//Deklarieren der benötigten Variablen
int LED_Eingang = 3;

//einmalig ausgeführte SetUp Befehle
void setup()
{
  //Zuweisen der Pin Funktion
  pinMode (LED_Eingang, INPUT);
}

//dauerhaft wiederholte Hauptschleife
void loop()
{
  //LED blinkt
  digitalWrite (LED_Eingang, HIGH);

  //Pause
  delay(500);

  //LED aus
  digitalWrite (LED_Eingang, LOW);

  //Pause
  delay(500);
}
```

B29 Heartbeat



Kurzbeschreibung / Technische Daten

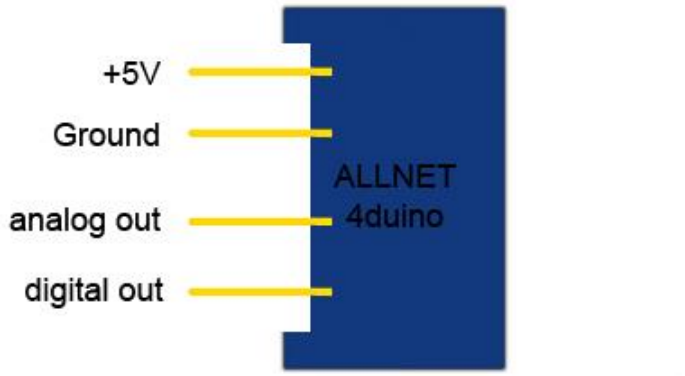
Digitaler Ausgang: Wird ein Herzschlag erkannt, wird hier ein Signal ausgegeben

Analoger Ausgang: Direkter Messwert der Sensoreinheit

LED1: Zeigt an, dass der Sensor mit Spannung versorgt ist

LED2: Zeigt an, dass der Herzschlag den Grenzwert überchritten hat

Pin-Belegung



Codebeispiel

Das Programm liest den aktuellen Spannungswert aus, der am analogen Ausgang gemessen werden kann, und gibt diesen auf der seriellen Schnittstelle aus.

Zudem wird ebenfalls der Zustand des digitalen Pins in der Konsole angegeben, was bedeutet ob der Grenzwert unterschritten wurde oder nicht.

Anschlussbelegung an Arduino::

Sensor +V = [Pin +5V]
Sensor GND = [Pin GND]
Sensor Digital = [Pin 4]
Sensor Analog = [Pin 3]

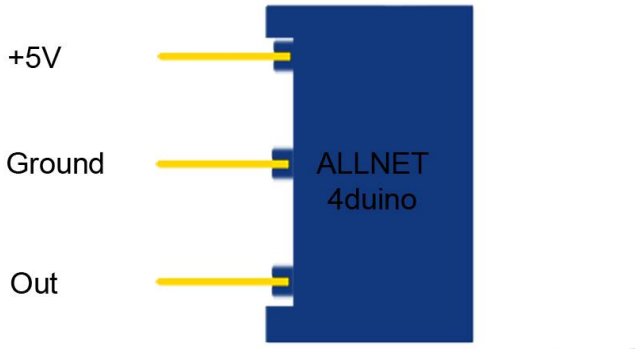
```
// ALLNET Heartbeat B29
// Information http://www.allnet.de


//Deklarieren der benötigten Variablen
int Analog_Eingang = A0;
int Digital_Eingang = 3;

//einmalig ausgeführte SetUp Befehle
void setup ()
{
  //Zuweisen der Pin Funktion
  pinMode (Analog_Eingang, INPUT);
  pinMode (Digital_Eingang, INPUT);

  //Starten der seriellen Übertragung
  Serial.begin (9600);
}

//dauerhaft wiederholte Hauptschleife
void loop ()
{
  //Deklarieren von temporären Zwischenspeichern
  float analog;
//Aktuelle Werte werden ausgelesen, auf den Spannungswert konvertiert...
  analog = analogRead (Analog_Eingang);
  digital = digitalRead (Digital_Eingang);

  //... und an dieser Stelle ausgegeben
  Serial.print ("Analoger Spannungswert:");
  Serial.print (analog);
  Serial.print (", ");
  Serial.print ("Grenzwert:");

  //Wenn der Wert digital 1 entspricht
  if (digital == 1)
  {
    //Dann ist der Grenzwert erreit und dies wird als Meldung ausgegeben
    Serial.println (" erreicht");
  }
  else
  {
    //Sonst ist der Grenzwert nicht erreit und dies wird als Meldung ausgegeben
    Serial.println (" noch nicht erreicht");
  }

  //Optische Abtrennung der Daten in der seriellen Ausgabe
  Serial.println ("--------------------------------");

  //Pause
  delay (200);
}
```

B30 Photoresistor / Lichtsensor

Kurzbeschreibung / Technische Daten

LDR-Widerstand, dessen Widerstandswert bei hellerer Umgebung kleiner wird.

Pin-Belegung



Codebeispiel

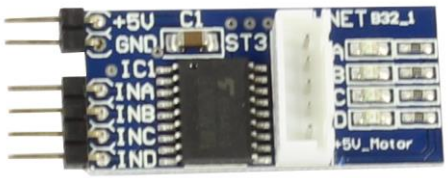
Das Programm misst den aktuellen Spannungswert am Sensor, berechnet aus diesen und dem bekannten Serienwiderstand den aktuellen Widerstandswert des Sensors und gibt die Ergebnisse auf der serielle Ausgabe aus.

Anschlussbelegung an Arduino:

Sensor GND = [Pin +5V]
Sensor +5V = [Pin Ground]
Sensor Output = [Pin A0]

```
// ALLNET Photoresistor/Lichtsensor B30
// Information http://www.allnet.de

//Deklarieren der benötigten Variablen
int sensorPin = A0;

//einmalig ausgeführte SetUp Befehle
void setup()
{
  //Zuweisen der Pin Funktion
  pinMode (sensorPin, INPUT);

  //Starten der seriellen Übertragung
  Serial.begin(9600);
}

// Das Programm misst den aktuellen Spannungswert am Sensor,
// berechnet aus diesem und dem bekannten Serienwiederstand den aktuellen
// Wiederstandswert des Sensors un gibt die Ergebnisse auf der Seriellen Ausgabe aus

void loop()
{
  // Aktueller Wert wird gemessen und die Voltzahl und der Wiederstand berechnet...
  int rawValue = analogRead(sensorPin);
  float voltage = rawValue * (5.0 / 1023) * 1000;
  float resitance = 10000 * ( voltage / ( 5000.0 - voltage) );

  // ... und hier auf die serielle Schnittstelle ausgegeben
  Serial.print("Spannungswert:");
  Serial.print(voltage);
  Serial.print("mV");
  Serial.print(", Widerstandswert:");
  Serial.print(resitance);
  Serial.println("Ohm");
  Serial.println("---------------------------------------");

  //Pause
  delay(500);
}
```

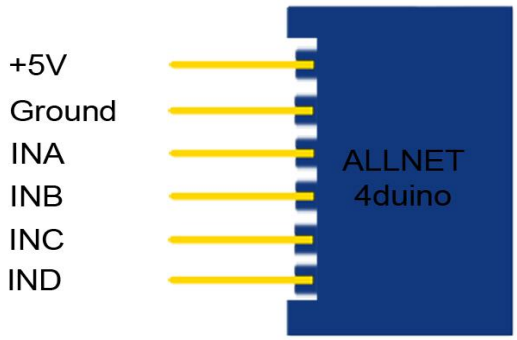
B32 5V Step-engine with driver PCB / Schrittmotor Treiber Board



Kurzbeschreibung / Technische Daten

Der Sensor ermöglicht es den Stepper Motor mit dem Arduino zu betreiben.

Pin-Belegung



Codebeispiel

Hier bei handelt es sich um ein Beispielprogramm, welches einen Steppermotor langsam immer einen Schritt weiter drehen lässt

Anschlussbelegung an Arduino:

Sensor GND	= [Pin GND]	Sensor INB	= [Pin 9]
Sensor +5V	= [Pin +5V]	Sensor INC	= [Pin 10]
Sensor INA	= [Pin 8]	Sensor IND	= [Pin 11]

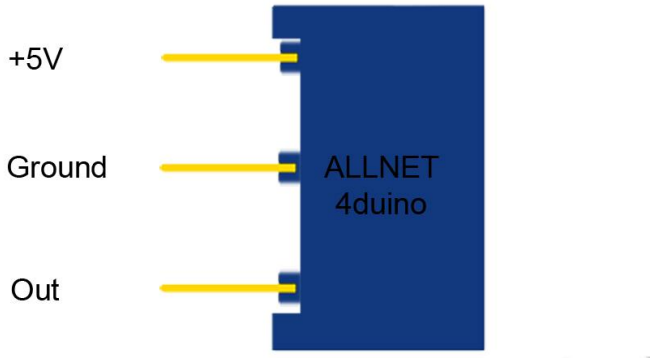
```
// ALLNET 5V Stepper-Engline with PCB Driver B32
// Information http://www.allnet.de

//Benötigte Libraries werden importiert
#include <Stepper.h>

//Deklarieren der benötigten Variablen
const int stepsPerRevolution = 32;
int stepCount = 0;

//Libraries werden konfiguriert
Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);

//einmalig ausgeführte SetUp Befehle
void setup()
{
  //Starten der seriellen Übertragung
  Serial.begin(9600);
}

//dauerhaft wiederholte Hauptschleife
void loop()
{
  //Stepper Motor 1 Schritt
  myStepper.step(1);
  stepCount++;

  //Ausgabe der gemachten Schritte
  Serial.print("steps:");
  Serial.println(stepCount);

  //Pause
  delay(1000);
}
```

B33 Gas MQ2 Sensor

Kurzbeschreibung / Technische Daten

Der Sensor legt eine unterschiedliche Spannung am OUT Pin an, je nach Gasmenge in der Luft.

Pin-Belegung



Codebeispiel

Hier bei handelt es sich um ein Beispielprogramm, welches den analogen Wert des Sensors auf der seriellen Anzeige ausgibt.

Anschlussbelegung an Arduino::

Sensor GND	= [Pin +5V]
Sensor +5V	= [Pin Ground]
Sensor Output	= [Pin A0]

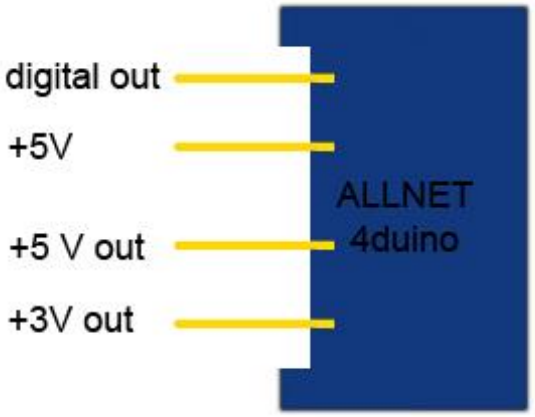
```
// ALLNET Gas MQ2 Sensor B33
// Information http://www.allnet.de

//Deklarieren der benötigten Variablen
int sensorPin = A0;

//einmalig ausgeführte SetUp Befehle
void setup()
{
  //Zuweisen der Pin Funktion
  pinMode(sensorPin, INPUT);

  //Starten der seriellen Übertragung
  Serial.begin (9600);
}

//dauerhaft wiederholte Hauptschleife
void loop()
{
    //Temporäre Variable mit dem Sensorwert
    int val = analogRead (sensorPin);

    //Ausgabe des Sensorwertes
    Serial.print ("Sensor Wert:  ");
    Serial.println (val);
    Serial.println ("---------------------------------------");

    //Pause
    delay (1000);
}
```

B34 Voltage Regulator linear

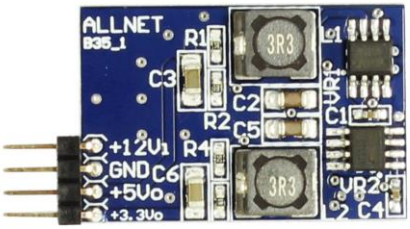
Kurzbeschreibung / Technische Daten

Der ALLNET B34 ist ein linearer Spannungs regulierer. Er senkt eine anliegende 12V Spannung auf 5 und 3 Volt. Vareiert die Eingangsspannung, so ändern sich die Ausgangsspannungen in gleichem Maße.

Pin-Belegung



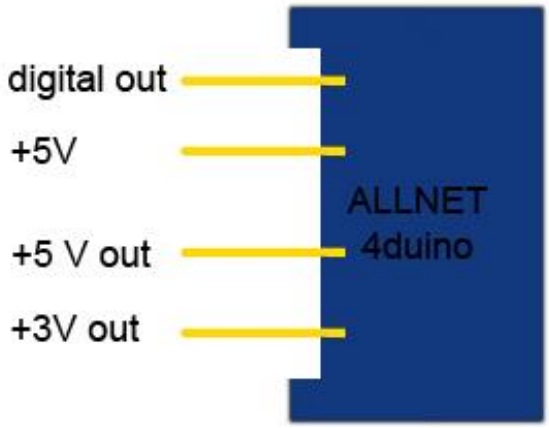
B35 Voltage Regulator



Kurzbeschreibung / Technische Daten

Der ALLNET B35 ist ein Spannungs regulierer. Er senkt eine anliegende 12V Spannung auf 5 und 3 Volt.

Pin-Belegung

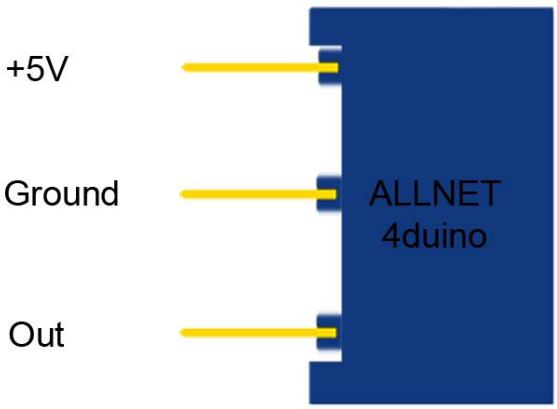


B36 Motion Detection / Bewegungsmelder

Kurzbeschreibung / Technische Daten

Der Motion Detector gibt ein Signal aus, wenn eine Bewegung erkannt wird.

Pin-Belegung



Codebeispiel

Hier bei handelt es sich um ein Beispielprogramm, welches eine Bewegung an den seriellen Monitor ausgibt.

Anschlussbelegung an Arduino::

Sensor GND	= [Pin GND]
Sensor +5V	= [Pin 5V]
Sensor Output	= [Pin 3]

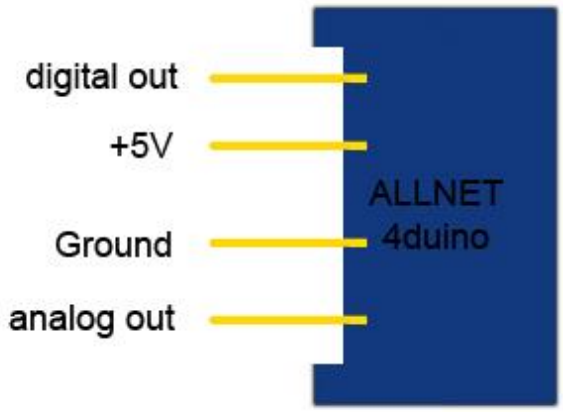
```
// ALLNET Motion Detection/Bewegungsmelder B36
// Information http://www.allnet.de

//Deklarieren der benötigten Variablen
int Motion_Eingang = 3;

//einmalig ausgeführte SetUp Befehle
void setup ()
{
  //Zuweisen der Pin Funktion
  pinMode (Motion_Eingang, INPUT);

  //Starten der seriellen Übertragung
  Serial.begin (9600);
}

//dauerhaft wiederholte Hauptschleife
void loop ()
{
  //Wenn der Wert des Digital_Eingang 1 entspricht
  if (digitalRead (Motion_Eingang) == 1)
  {
    //Dann ist eine Bewegung erkannt und dies wird als Meldung ausgegeben
    Serial.println ("Bewegung erkannt");
  }

  //Pause
  delay (200);
}
```

B37 8 LED PCB / 8-fach LED PCB

Kurzbeschreibung / Technische Daten

Durch das 8-fach LED PCB lassen sich 8 seperate LEDs mittels des I²C Busses ansteuern.

Pin-Belegung



Codebeispiel

Das Beispielprogramm simuliert eine fallende Akkuanzeige.

Anschlussbelegung an Arduino::

Sensor GND	= [Pin GND]
Sensor +5V	= [Pin 5V]
Sensor Output	= [Pin A0]

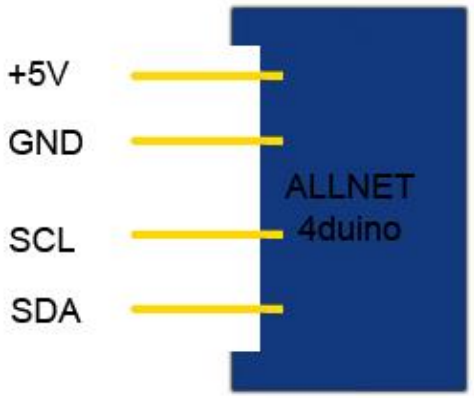
```
// ALLNET Flash LED B28
// Information http://www.allnet.de

//Benötigte Libraries werden importiert
#include<Wire.h>

//Deklarieren der benötigten Variablen
int counter = 0;
int power = 7 ;
int y = 128;

//einmalig ausgeführte SetUp Befehle
void setup()
{
  //Starten der I2C Sensor Verbindung
  Wire.begin();

  //Starten der seriellen Übertragung
  Serial.begin (9600); // 9600 bps
}

//dauerhaft wiederholte Hauptschleife
void loop()
{
  if (counter % 10000 == 0)
  {
    //Serielle Ausgabe der Variablen
    Serial.print (power, DEC);
    Serial.print (" - ");
    Serial.println (y, DEC);
    //Begin der Datenübertragung
    Wire.beginTransmission(32);
    //Senden der Daten
    Wire.write(y);
    //Ende der Datenübertragung
    Wire.endTransmission();
    y = y / 2;
    power = power - 1;

  }
  counter++;
  if (power < 0)
  {
    y = 128;
    power = 7;
  }

  //Pause
  delay(1);
}
```

B38 Temperature I2C / Temperatur I2C Sensor

Kurzbeschreibung / Technische Daten

Der ALLNET B38 ist ein digitale Temperatur Sensor, mit dem über I²C kommuniziert wird.

Pin-Belegung



Codebeispiel

Hier bei handelt es sich um ein Beispielprogramm, welches den analogen Wert des Sensots auf der seriellen Anzeige ausgibt.

Anschlussbelegung an Arduino::

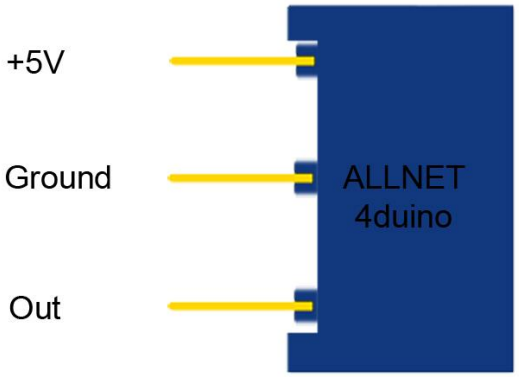
Sensor GND	= [Pin GND]
Sensor +5V	= [Pin 5V]
Sensor SCL	= [Pin A5]
Sensor SDA	= [Pin A4]

```
// ALLNET Temperature I2C/Temperatur I2C Sensor B38
// Information http://www.allnet.de


//Benötigte Libraries werden importiert
#include <Wire.h>

//Deklarieren der benötigten Variablen
#define adress 0x4F

//einmalig ausgeführte SetUp Befehle
void setup()
{
  //Starten der seriellen Übertragung
  Serial.begin(9600);

  //Starten der Sensor Verbindung
  Wire.begin();
}

//dauerhaft wiederholte Hauptschleife
void loop()
{
  //Ausführen der unten deklarierten Funktion
  int c1 = read_temp(address);

  // Ausgabe des ermittelten Sensorwertes
  Serial.print("Sensor 1: ");
  Serial.print(c1);

  //Pause
  delay(500);
}

int read_temp(int address)
{
  //Start der Übertragung mit dem Sensor
  Wire.beginTransmission(address);
  //Senden eines Bits zum erhalt der Informationen
  Wire.write(0x00);
  //Anfrage 1 Bytes des Sensors
  Wire.requestFrom(address, 2);
  //Warten auf eine Antwort
  if (Wire.available() == 0)
  {
  //Speichern und Rückgabe des Wertes
  int c = Wire.read();
  }
  //Beende Übertragung
  Wire.endTransmission();
  return c;
}
```

B39 Vibration Sensor / Vibrations Sensor

Kurzbeschreibung / Technische Daten

Der Sensor legt ein Signal am OUT Pn an, wenn eine Vibration erkannt wird.



Pin-Belegung

Codebeispiel

Hier bei handelt es sich um ein Beispielprogramm, welches den analogen Wert des Sensots auf der seriellen Anzeige ausgibt.

Anschlussbelegung an Arduino:

Sensor GND	= [Pin GND]
Sensor +5V	= [Pin 5V]
Sensor Output	= [Pin 3]

```
// ALLNET Vibration Sensor/Vibrations Sensor B39
// Information http://www.allnet.de

//Deklarieren der benötigten Variablen
int Vibration_Eingang = 3;

//einmalig ausgeführte SetUp Befehle
void setup ()
{
  //Zuweisen der Pin Funktion
  pinMode (Vibration_Eingang, INPUT);

  //Starten der seriellen Übertragung
  Serial.begin (9600);
}

//dauerhaft wiederholte Hauptschleife
void loop ()
{
  //Wenn der Wert des Digital_Eingang 1 entspricht
  if (digitalRead (Vibration_Eingang) == 1)
  {
    //Dann ist eine Vibration erkannt worden und dies wird als Meldung ausgegeben
    Serial.println ("Vibration erkannt");
  }

  //Pause
  delay (1000);
}
```