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 Übersicht beinhaltet die technische Beschreibung der einzelnen Sensoren, wie z.B. die Pin-Belegung oder den jeweils verwendeten Chipsatz.

Die Sensoren werden ohne Verbindungsmaterial geliefert, welches Sie aber auch über uns beziehen können. Die Verbindung können Sie entweder über ein Breadboard machen als auch löten oder direkt verbinden via Dupont Kabeln.

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

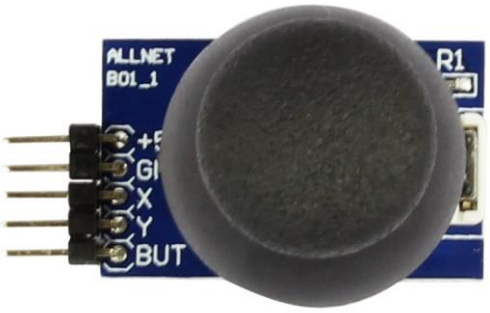
Ihr ALLNET Team

Inhaltsverzeichnis Sensorenübersicht

Sensorenübersicht

B01 Joystick Modul (XY-Achsen)

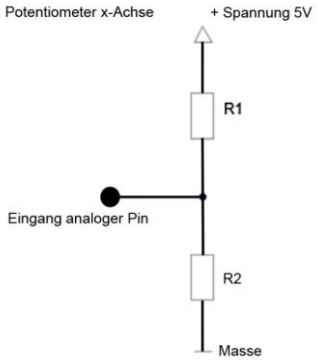
Bild



Kurzbeschreibung / Technische Daten

X und Y Position des Joysticks, werden als analoge Spannung auf den Ausgangspins ausgegeben.

In diesem Joystick wurde für die X-Achse, sowie für die Y-Achse, ein eigenes Potentiometer verbaut. Diese ergeben einen Spannungsteiler.

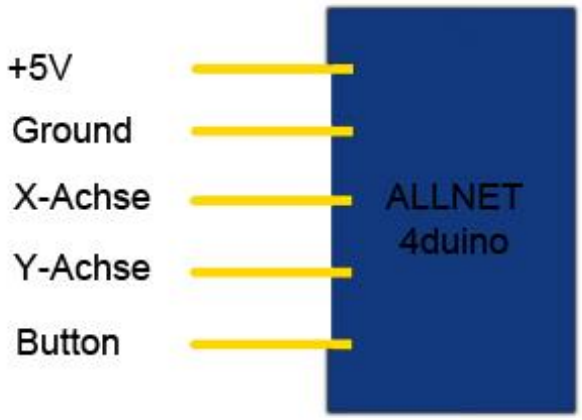


Im Ruhezustand befind sich der Joystick in der Mitte, so dass die angelegte Spannung an beide Widerstände verteilt wird.

Wird jetzt die Position v verändert, so ändern sich abhängig zur aktuellen eine Richtung so wird W Widerstand 2 größer, g so wird dann Widerstand kleiner.

Je nachdem wie sich die aufteilen, resultiert dies Spannungswert, den m Widerständen (beim Po messen und somit die P kann.

Pin-Belegung



Codebeispiel

Das Programm liest die aktuellen Werte der Eingang-Pins und gibt diese auf der seriellen Ausgabe aus.

Anschlussbelegung:

Sensor +V	= [Pin 5V]
Sensor GND	= [Pin GND]
Y-Position	= [Pin A1]
X-Position	= [Pin A0]
Knopf	= [Pin 3]

```
// 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 ()
{
```

```
  //Deklarieren von temporären Zwischenspeichern
  int x, y, b;

  //Einlesen der Joystick Daten in die
Zwischenspeicher
  x = analogRead (JoyStick_X);
  y = analogRead (JoyStick_Y);
  b = !digitalRead (JoyStick_Button);

  //Serielle Ausgabe der Werte als Dezimalzal
  Serial.print (x, DEC);
  Serial.print (",");
  Serial.print (y, DEC);
  Serial.print (",");
  Serial.println (b, DEC);

  //Pause
  delay (100);
}
```

B02 5V Relay Modul

Bild

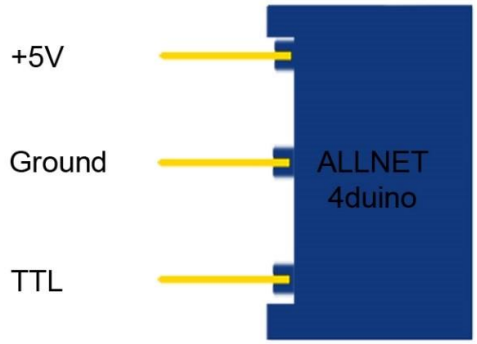
Kurzbeschreibung / Technische Daten

Spannungsbereich: 240VAC / 10A | 28VDC / 10AEin Relais zum Schalten von höherer Spannungen mittels eines 5V Ausgangs.
Die Ausgangsleiste des Relais besitzt zwei Ausgangsterminals:

- Das eine welches mit "NC" für "normally closed" gekennzeichnet ist, was bedeutet dass dieser Durchgang ohne elektrische Umschaltung am Relais standardmäßig kurzgeschlossen ist.
- Das andere welches mit "NO" für "normally open" gekennzeichnet ist, was 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

Das Programm bildet einen Blinker nach - es schaltet das Relais in vorher definierter Zeit (delayTime) zwischen den beiden Zuständen (bzw. Ausgangsterminals) um.

Anschlussbelegung:

Sensor +V = [Pin 5V]
Sensor GND = [Pin GND]
Sensor TTL = [Pin 10]

```
// 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

Bild

Kurzbeschreibung / Technische Daten

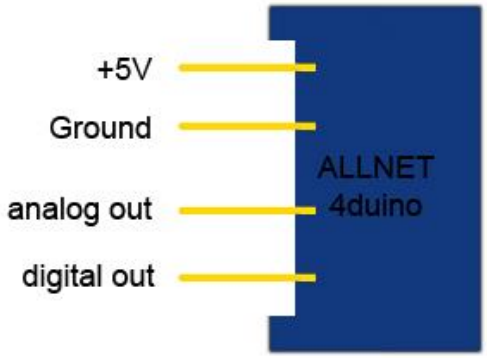
Digitaler Ausgang: Über das Potentiometer, kann ein Grenzwert für den empfangenen Schall eingestellt werden, bei dem der digitale Ausgang schalten soll.

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

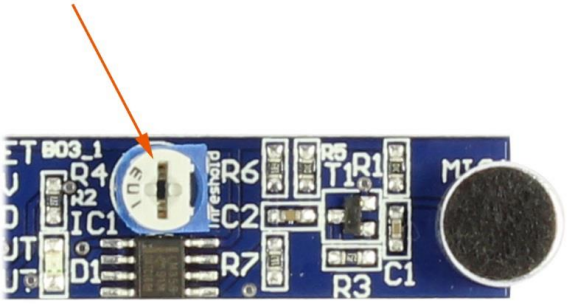


Funktionsweise des Sensors

Dieser Sensor besitzt auf seiner Platine drei funktionelle Bestandteile. Die ist die Sensoreinheit vorne am Modul, welche das aktuelle Umfeld physikalisch misst und als analoges Signal auf die zweite Einheit, dem Verstärker, ausgibt. Dieser verstärkt das Signal abhängig vom eingestellten Widerstand am Drehpotentiometer und leitet es auf den analogen Ausgang des Moduls.

Hierbei ist zu beachten: Das Signal ist invertiert; wird ein hoher Wert gemessen, so resultiert dies in einen niedrigeren Spannungswert am analogen Ausgang.

Die dritte Einheit stellt einen Komparator dar, welcher den digitalen Ausgang und die LED schaltet, wenn das Signal unter einen bestimmten Wert fällt. Mittels des Drehpotentiometers kann somit die Empfindlichkeit eingestellt werden, wie es im folgenden Bild aufgezeigt wird:



Dieser Sensor gibt somit keine absoluten Werte aus (z.B. genau gemessene Temperatur in °C oder Magnetfeldstärke in mT) , sondern es handelt sich hierbei um eine Relativ-Messung: Man definiert hierbei einen Grenzwert relativ zur gegebenen normalen Umweltsituation und es wird ein Signal ausgegeben was weiterverarbeitet werden kann, falls dieser Grenzwert überschritten bzw. ein anderer Zustand als der Normalfall eingetreten ist.

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:

Sensor +V	= [Pin 5V]
Sensor GND	= [Pin GND]
Sensor Digital	= [Pin 3]
Sensor Analog	= [Pin A0]

```
// ALLNET Mikrofon Sensor Modul B03
// 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;
  int digital;

  //Aktuelle Werte werden ausgelesen, auf den
Spannungswert konvertiert...
  analog = analogRead (Analog_Eingang) * (5.0 / 1023.0);
  digital = digitalRead (Digital_Eingang);

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

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

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

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

B04 IR Optical Detection / IRErkennung

Bild

Kurzbeschreibung / Technische Daten

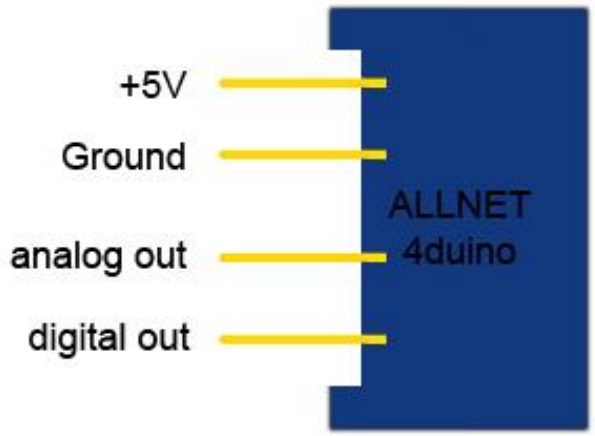
Digitaler Ausgang: Über das Potentiometer, kann ein Grenzwert eingestellt werden, bei dem der digitale Ausgang schalten soll.

Analoger Ausgang: Direktes IR-Signal als Spannungspegel

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

LED2: Zeigt an, dass ein IR-Signal detektiert wurde

Pin-Belegung

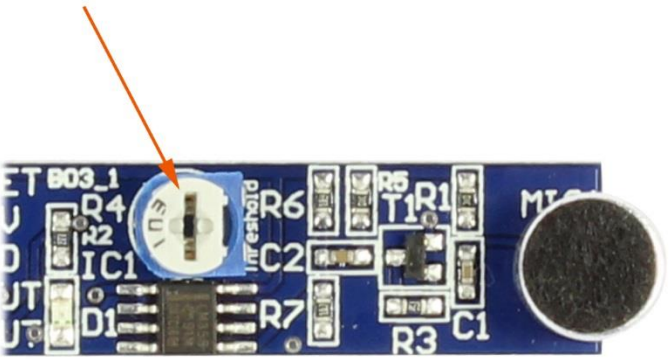


Funktionsweise des Sensors

Dieser Sensor besitzt auf seiner Platine drei funktionelle Bestandteile. Die ist die Sensoreinheit vorne am Modul, welche das aktuelle Umfeld misst und als analoges Signal auf die zweite Einheit, dem Verstärker, ausgibt. Dieser verstärkt das Signal abhängig vom eingestellten Widerstand am Drehpotentiometer und leitet es auf den analogen Ausgang des Moduls.

Hierbei ist zu beachten: Das Signal ist invertiert; wird ein hoher Wert gemessen, so resultiert dies in einen niedrigeren Spannungswert am analogen Ausgang.

Die dritte Einheit stellt einen Komparator dar, welcher den digitalen Ausgang und die LED schaltet, wenn das Signal unter einen bestimmten Wert fällt. Mittels des Drehpotentiometers kann somit die Empfindlichkeit eingestellt werden, wie es im folgenden Bild aufgezeigt wird:



Dieser Sensor gibt somit keine absoluten Werte aus (z.B. genau gemessene Temperatur in °C oder Magnetfeldstärke in mT) , sondern es handelt sich hierbei um eine Relativ-Messung: Man definiert hierbei einen Grenzwert relativ zur gegebenen normalen Umweltsituation und es wird ein Signal ausgegeben was weiterverarbeitet werden kann, falls dieser Grenzwert überschritten bzw. ein anderer Zustand als der Normalfall eingetreten ist.

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:

Sensor +V	= [Pin 5V]
Sensor GND	= [Pin GND]
Sensor Digital	= [Pin 3]
Sensor Analog	= [Pin A0]

```
// ALLNET IR Detection / IR Erkennung B04
// 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;
  int digital;

  //Aktuelle Werte werden ausgelesen, auf den Spannungswert konvertiert...
  analog = analogRead (Analog_Eingang) * (5.0 / 1023.0);
  digital = digitalRead (Digital_Eingang);

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

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

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

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

B05 Flame Sensor / Flammensensor

Bild

Kurzbeschreibung / Technische Daten

Die angebrachte Fotodiode ist empfindlich auf den Spektralbereich von Licht, welches von offenen Flamen erzeugt wird.

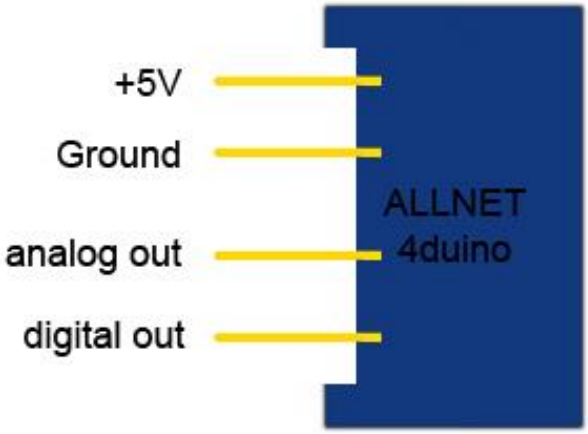
Digitaler Ausgang: Wird eine Flame erkannt, wird hier ein Signal ausgegeben

Analoger Ausgang: Direkter Messwert der Sensoreinheit

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

LED2: Zeigt an, dass eine Flamme detektiert wurde

Pin-Belegung

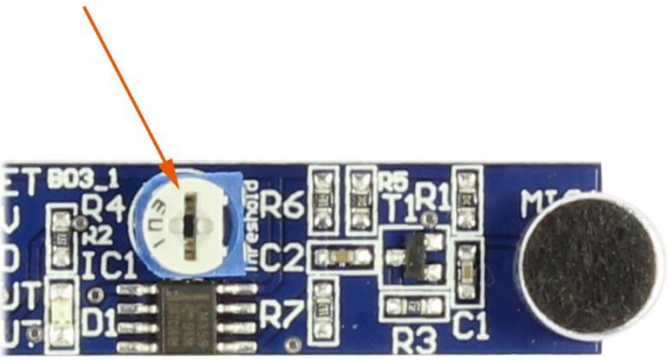


Funktionsweise des Sensors

Dieser Sensor besitzt auf seiner Platine drei funktionelle Bestandteile. Die ist die Sensoreinheit vorne am Modul, welche das aktuelle Umfeld physikalisch misst und als analoges Signal auf die zweite Einheit, dem Verstärker, ausgibt. Dieser verstärkt das Signal abhängig vom eingestellten Widerstand am Drehpotentiometer und leitet es auf den analogen Ausgang des Moduls.

Hierbei ist zu beachten: Das Signal ist invertiert; wird ein hoher Wert gemessen, so resultiert dies in einen niedrigeren Spannungswert am analogen Ausgang.

Die dritte Einheit stellt einen Komparator dar, welcher den digitalen Ausgang und die LED schaltet, wenn das Signal unter einen bestimmten Wert fällt. Mittels des Drehpotentiometers kann somit die Empfindlichkeit eingestellt werden, wie es im folgenden Bild aufgezeigt wird:



Dieser Sensor gibt somit keine absoluten Werte aus (z.B. genau gemessene Temperatur in °C oder Magnetfeldstärke in mT) , sondern es handelt sich hierbei um eine Relativ-Messung: Man definiert hierbei einen Grenzwert relativ zur gegebenen normalen Umweltsituation und es wird ein Signal ausgegeben was weiterverarbeitet werden kann, falls dieser Grenzwert überschritten bzw. ein anderer Zustand als der Normalfall eingetreten ist.

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:

Sensor +V	= [Pin 5V]
Sensor GND	= [Pin GND]
Sensor Digital	= [Pin 3]
Sensor Analog	= [Pin A0]

```
// ALLNET Flame Sensor / Flammensensor B05
// 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;
  int digital;

  //Aktuelle Werte werden ausgelesen, auf den Spannungswert konvertiert...
  analog = analogRead (Analog_Eingang) * (5.0 / 1023.0);
  digital = digitalRead (Digital_Eingang);

  //... und an dieser Stelle ausgegeben
  Serial.print ("Analoger Spannungswert:");
  Serial.print (analog);
  Serial.print (" V, ");
  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 erreicht und dies wird als Meldung ausgegeben
    Serial.println (" noch nicht erreicht");
  }
```

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

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

B06 Hall TTL Sensor

Bild

Kurzbeschreibung / Technische Daten

Der angebrachte Hall-Sensor ist empfindlich reagiert empfindlich auf das umliegende Magnetfeld.

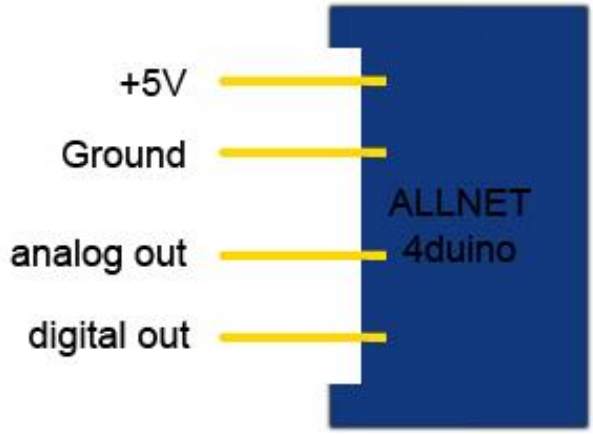
Digitaler Ausgang: Wird eine Ängerung des Magnetfeldes erkannt, wird hier ein Signal ausgegeben

Analoger Ausgang: Direkter Messwert der Sensoreinheit

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

LED2: Zeigt an, dass ein Magnetfeld detektiert wurde

Pin-Belegung

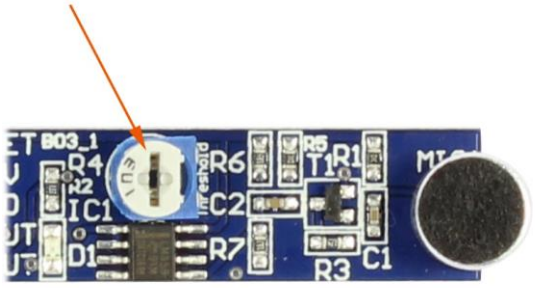


Funktionsweise des Sensors

Dieser Sensor besitzt auf seiner Platine drei funktionelle Bestandteile. Die ist die Sensoreinheit vorne am Modul, welche das aktuelle Umfeld physikalisch misst und als analoges Signal auf die zweite Einheit, dem Verstärker, ausgibt. Dieser verstärkt das Signal abhängig vom eingestellten Widerstand am Drehpotentiometer und leitet es auf den analogen Ausgang des Moduls.

Hierbei ist zu beachten: Das Signal ist invertiert; wird ein hoher Wert gemessen, so resultiert dies in einen niedrigeren Spannungswert am analogen Ausgang.

Die dritte Einheit stellt einen Komparator dar, welcher den digitalen Ausgang und die LED schaltet, wenn das Signal unter einen bestimmten Wert fällt. Mittels des Drehpotentiometers kann somit die Empfindlichkeit eingestellt werden, wie es im folgenden Bild aufgezeigt wird:



Dieser Sensor gibt somit keine absoluten Werte aus (z.B. genau gemessene Temperatur in °C oder Magnetfeldstärke in mT) , sondern es handelt sich hierbei um eine Relativ-Messung: Man definiert hierbei einen Grenzwert relativ zur gegebenen normalen Umweltsituation und es wird ein Signal ausgegeben was weiterverarbeitet werden kann, falls dieser Grenzwert überschritten bzw. ein anderer Zustand als der Normalfall eingetreten ist.

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.

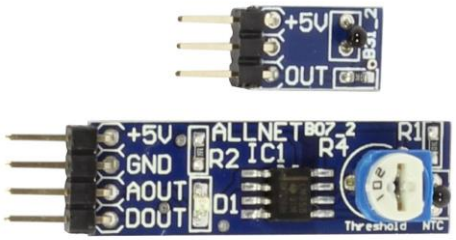
Anschlussbelegung:

Sensor +V	= [Pin 5V]
Sensor GND	= [Pin GND]
Sensor Digital	= [Pin 3]
Sensor Analog	= [Pin A0]

```
// ALLNET Hall TTL Sensor B06
// 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;
  int digital;

  //Aktuelle Werte werden ausgelesen, auf den Spannungswert konvertiert...
  analog = analogRead (Analog_Eingang) * (5.0 / 1023.0);
  digital = digitalRead (Digital_Eingang);

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

B07/B31 NTC Threshold TTL/ NTC 10k

Bild



Kurzbeschreibung / Technische Daten

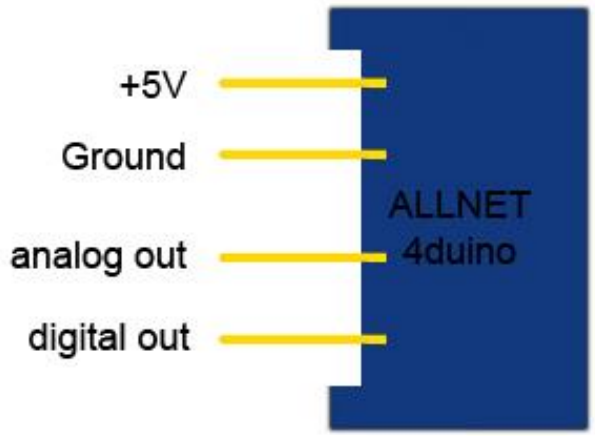
Digitaler Ausgang: Wird eine Flame erkannt, wird hier ein Signal ausgegeben

Analoger Ausgang: Direkter Messwert der Sensoreinheit

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

LED2: Zeigt an, dass die aktuelle Temperatur den Grenzwert überchritten hat

Pin-Belegung

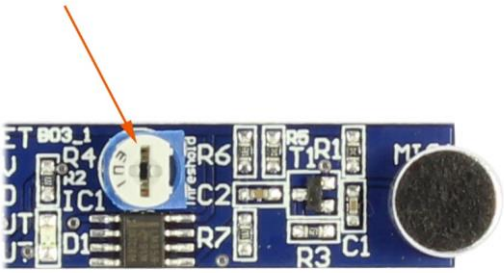


Funktionsweise des Sensors

Dieser Sensor besitzt auf seiner Platine drei funktionelle Bestandteile. Die ist die Sensoreinheit vorne am Modul, welche das aktuelle Umfeld physikalisch misst und als analoges Signal auf die zweite Einheit, dem Verstärker, ausgibt. Dieser verstärkt das Signal abhängig vom eingestellten Widerstand am Drehpotentiometer und leitet es auf den analogen Ausgang des Moduls.

Hierbei ist zu beachten: Das Signal ist invertiert; wird ein hoher Wert gemessen, so resultiert dies in einen niedrigeren Spannungswert am analogen Ausgang.

Die dritte Einheit stellt einen Komparator dar, welcher den digitalen Ausgang und die LED schaltet, wenn das Signal unter einen bestimmten Wert fällt. Mittels des Drehpotentiometers kann somit die Empfindlichkeit eingestellt werden, wie es im folgenden Bild aufgezeigt wird:



Dieser Sensor gibt somit keine absoluten Werte aus (z.B. genau gemessene Temperatur in °C oder Magnetfeldstärke in mT) , sondern es handelt sich hierbei um eine Relativ-Messung: Man definiert hierbei einen Grenzwert relativ zur gegebenen normalen Umweltsituation und es wird ein Signal ausgegeben was weiterverarbeitet werden kann, falls dieser Grenzwert überschritten bzw. ein anderer Zustand als der Normalfall eingetreten ist.

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:

Sensor +V	= [Pin 5V]
Sensor GND	= [Pin GND]
Sensor Digital	= [Pin 3]
Sensor Analog	= [Pin A0]

```
// ALLNET NTC Threshold TTL/NTC 10k B07/B31
// 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;
  int digital;

  //Aktuelle Werte werden ausgelesen, auf den Spannungswert konvertiert...
  analog = analogRead (Analog_Eingang) * (5.0 / 1023.0);
  digital = digitalRead (Digital_Eingang);

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

  //Wenn der Wert digital 1 entspricht
  if (digital == 1)
  {
    //Dann ist der Grenzwert erreicht 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);
}
```

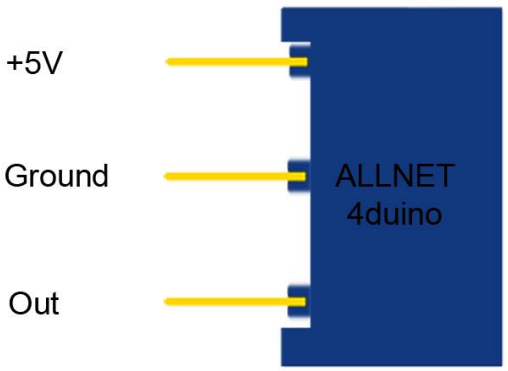
B08 Touch Sensor

Bild

Kurzbeschreibung / Technische Daten

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

Pin-Belegung



Codebeispiel

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

Anschlussbelegung:

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

```
// 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);
}
```

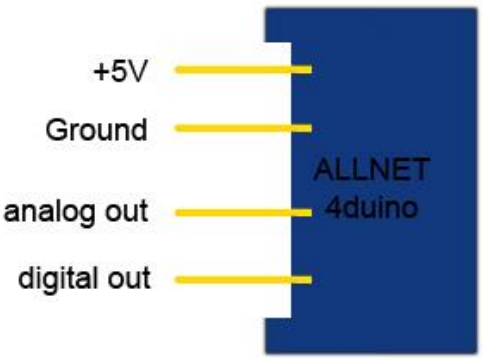
B09 RGB LED

Bild

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.

Anschlussbelegung:

LED GND	= [Pin GND]
LED Rot	= [Pin 3]
LED Grün	= [Pin 4]
LED Balu	= [Pin 5]

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


//Deklarieren der benötigten Variablen
int Led_Rot = 3;
int Led_Gruen = 4;
int Led_Blau = 5;

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

//dauerhaft wiederholte Hauptschleife
void loop ()
{
  //die rote LED wird eingeschaltet und die grüne und die blaue LED werden ausgeschaltet
  digitalWrite (Led_Rot, HIGH);
  digitalWrite (Led_Gruen, LOW);
  digitalWrite (Led_Blau, LOW);

  //Pause
  delay (3000);

  //die grüne LED wird eingeschaltet und die rote und die blaue LED werden ausgeschaltet
  digitalWrite (Led_Rot, LOW);
  digitalWrite (Led_Gruen, HIGH);
  digitalWrite (Led_Blau, LOW);

  //Pause
  delay (3000);

  //die blaue LED wird eingeschaltet und die grüne und die rote LED werden ausgeschaltet
  digitalWrite (Led_Rot, LOW);
  digitalWrite (Led_Gruen, LOW);
  digitalWrite (Led_Blau, HIGH);

  //Pause
  delay (3000);
}// ALLNET RGB LED B09
// Information http://www.allnet.de

//Deklarieren der benötigten Variablen
int Led_Rot = 3;
int Led_Gruen = 4;
int Led_Blau = 5;

//einmalig ausgeführte SetUp Befehle
void setup ()
{
  //Zuweisen der Pin Funktion
  pinMode (Led_Rot, OUTPUT);
  pinMode (Led_Gruen, OUTPUT);
  pinMode (Led_Blau, OUTPUT);
}
```

```
//dauerhaft wiederholte Hauptschleife
void loop ()
{
  //die rote LED wird eingeschaltet und die grüne und die blaue LED werden ausgeschaltet
  digitalWrite (Led_Rot, HIGH);
  digitalWrite (Led_Gruen, LOW);
  digitalWrite (Led_Blau, LOW);

  //Pause
  delay (3000);

  //die grüne LED wird eingeschaltet und die rote und die blaue LED werden ausgeschaltet
  digitalWrite (Led_Rot, LOW);
  digitalWrite (Led_Gruen, HIGH);
  digitalWrite (Led_Blau, LOW);

  //Pause
  delay (3000);

  //die blaue LED wird eingeschaltet und die grüne und die rote LED werden ausgeschaltet
  digitalWrite (Led_Rot, LOW);
  digitalWrite (Led_Gruen, LOW);
  digitalWrite (Led_Blau, HIGH);

  //Pause
  delay (3000);
}
  //die grüne LED wird eingeschaltet und die rote und die blaue LED werden ausgeschaltet
  digitalWrite (Led_Rot, LOW);
  digitalWrite (Led_Gruen, HIGH);
  digitalWrite (Led_Blau, LOW);

  //Pause
  delay (3000);

  //die blaue LED wird eingeschaltet und die grüne und die rote LED werden ausgeschaltet
  digitalWrite (Led_Rot, LOW);
  digitalWrite (Led_Gruen, LOW);
  digitalWrite (Led_Blau, HIGH);

  //Pause
  delay (3000);
}
  //die grüne LED wird eingeschaltet und die rote und die blaue LED werden ausgeschaltet
  digitalWrite (Led_Rot, LOW);
  digitalWrite (Led_Gruen, HIGH);
  digitalWrite (Led_Blau, LOW);

  //Pause
  delay (3000);

  //die blaue LED wird eingeschaltet und die grüne und die rote LED werden ausgeschaltet
  digitalWrite (Led_Rot, LOW);
  digitalWrite (Led_Gruen, LOW);
  digitalWrite (Led_Blau, HIGH);

  //Pause
  delay (3000);
}
```

Codebeispiel

Mittels Puls-Weiten-Modulation [PWM] lässt sich die Helligkeit einer LED regulieren - dabei wird die LED in bestimmten Zeitintervallen ein und ausgeschaltet, wobei das Verhältnis der Einschalt- und Ausschaltzeit einer relativen Helligkeit entspricht - aufgrund der Trägheit des menschlichen Sehvermögens, interpretieren die menschlichen Augen ein solches Ein-/Ausschaltverhalten als Helligkeitsänderung. Nähere Informationen zu diesem Thema finden Sie in diesem [Artikel von mikrokontroller.net].

In diesem Modul sind mehrere LEDs integriert - durch die Überlagerung von unterschiedlichen Helligkeitsstufen lassen sich somit verschiedene Farben kreieren. Dieses wird im folgenden Codebeispiel gezeigt.

Anschlussbelegung:

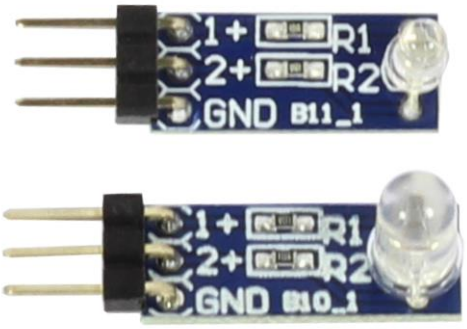
LED GND	= [Pin GND]
LED Rot	= [Pin 3]
LED Grün	= [Pin 4]
LED Balu	= [Pin 5]

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


//Deklarieren der benötigten Variablen
int Led_Rot = 3;
int Led_Gruen = 4;
int Led_Blau = 5;

int val;

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

//dauerhaft wiederholte Hauptschleife
void loop ()
{
  // Innerhalb einer For-Schleife werden den drei LEDs verschiedene PWM-Werte uebergeben
  // Dadurch entsteht ein Farbverlauf, in dem sich durch das Vermischen unterschiedlicher
  // Helligkeitstufen der beiden integrierten LEDs, unterschiedliche Farben entstehen
  for (val = 255; val > 0; val--)
  {
    analogWrite (Led_Blau, val);
    analogWrite (Led_Gruen, 255 - val);
    analogWrite (Led_Rot, 128 - val);
    delay (1);
  }
  // In der zweiten For-Schleife wird der Farbverlauf rückwärts durchgegangen
  for (val = 0; val <255; val++)
  {
    analogWrite (Led_Blau, val);
    analogWrite (Led_Gruen, 255 - val);
    analogWrite (Led_Rot, 128 - val);
    delay (1);
  }
}
```

```
  // In der zweiten For-Schleife wird der Farbverlauf rückwärts durchgegangen
  for (val = 0; val <255; val++)
  {
    analogWrite (Led_Blau, val);
    analogWrite (Led_Gruen, 255 - val);
    analogWrite (Led_Rot, 128 - val);
    delay (1);
  }
}
```

B10/B11 Zweifarbige LED 5/3mm

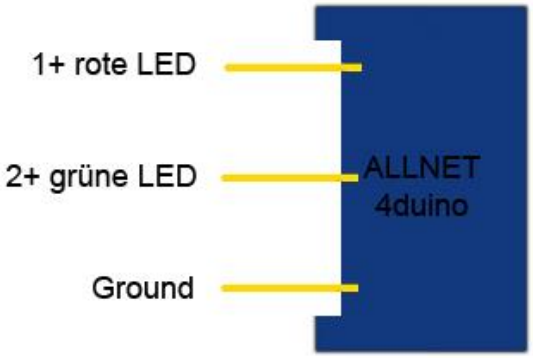
Bild



Kurzbeschreibung / Technische Daten

LED-Modul welche eine rote 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.

Anschlussbelegung:

LED GND = [Pin GND]
LED Rot = [Pin 1+]
LED Grün = [Pin 2+]

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


//Deklarieren der benötigten Variablen
int Led_Rot = 3;
int Led_Gruen = 4;

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

//dauerhaft wiederholte Hauptschleife
void loop ()
{
  //die rote LED wird eingeschaltet und die grüne LED wird ausgeschaltet
  digitalWrite (Led_Rot, HIGH);
  digitalWrite (Led_Gruen, LOW);

  //Pause
  delay (3000);

  //die grüne LED wird eingeschaltet und die rote LED wird ausgeschaltet
  digitalWrite (Led_Rot, LOW);
  digitalWrite (Led_Gruen, HIGH);

  //Pause
  delay (3000);
}
```

Mittels Puls-Weiten-Modulation [PWM] lässt sich die Helligkeit einer LED regulieren - dabei wird die LED in bestimmten Zeitintervallen ein und ausgeschaltet, wobei das Verhältnis der Einschalt- und Ausschaltzeit einer relativen Helligkeit entspricht - aufgrund der Trägheit des menschlichen Sehvermögens, interpretieren die menschlichen Augen ein solches Ein-/Ausschaltverhalten als Helligkeitsänderung. Nähere Informationen zu diesem Thema finden Sie in diesem [Artikel von mikrokontroller.net].

In diesem Modul sind mehrere LEDs integriert - durch die Überlagerung von unterschiedlichen Helligkeitsstufen lassen sich somit verschiedene Farben kreieren. Dieses wird im folgenden Codebeispiel gezeigt.

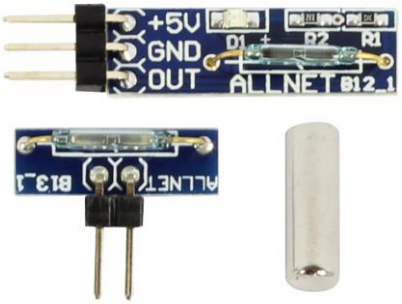
Anschlussbelegung:

LED GND = [Pin GND]
LED Rot = [Pin 1+]
LED Grün = [Pin 2+]

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


//Deklarieren der benötigten Variablen
int Led_Rot = 3;
int Led_Gruen = 4;

int val;

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

//dauerhaft wiederholte Hauptschleife
void loop ()
{
  // Innerhalb einer For-Schleife werden den drei LEDs verschiedene PWM-Werte uebergeben
  // Dadurch entsteht ein Farbverlauf, in dem sich durch das Vermischen unterschiedlicher
  // Helligkeitstufen der beiden integrierten LEDs, unterschiedliche Farben entstehen
  for (val = 255; val > 0; val--)
  {
    analogWrite (Led_Gruen, 255 - val);
    analogWrite (Led_Rot, 128 - val);
    delay (1);
  }
  // In der zweiten For-Schleife wird der Farbverlauf rückwärts durchgegangen
  for (val = 0; val <255; val++)
  {
    analogWrite (Led_Gruen, 255 - val);
    analogWrite (Led_Rot, 128 - val);
    delay (1);
  }
}
```

B12/B13 Reed Sensor/Magnet Kontakt Sensor

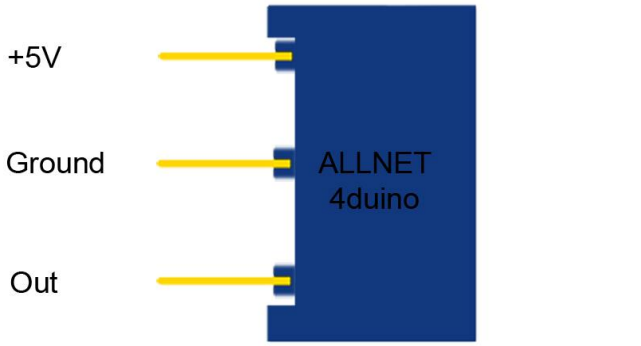
Bild



Kurzbeschreibung / Technische Daten

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

Pin-Belegung



Codebeispiel

Hier bei handelt es sich um ein Beispielprogramm, welches eine serielle Anzeige auslöst, wenn am Sensor ein Signal detektiert wurde.

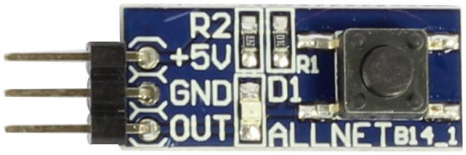
Anschlussbelegung:

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

```
// ALLNET Reed Sensor/Magnet Kontakt Sensor B12/B13
// 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 ein Magnetfeld erkannt und dies wird als Meldung ausgegeben
    Serial.println ("Berührung erkannt");
  }

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

B14 Button/Taster

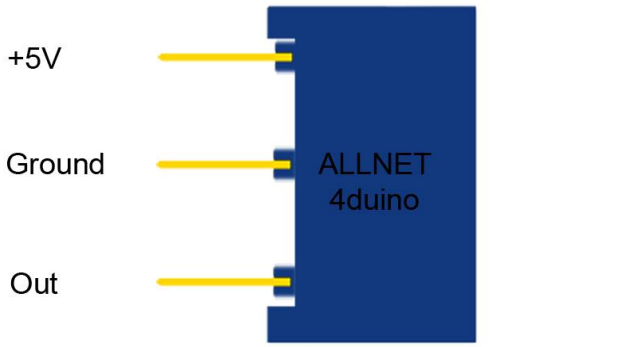
Bild



Kurzbeschreibung / Technische Daten

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

Pin-Belegung



Codebeispiel

Hier bei handelt es sich um ein Beispielprogramm, welches ein Druck des Tasters an die serielle Ausgabe weiter gibt.

Anschlussbelegung:

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

```
// 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);
}
```

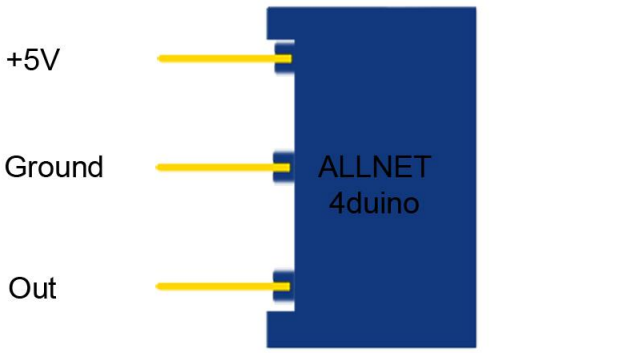
B15 Tilt Sensor/Neigungssensor

Bild

Kurzbeschreibung / Technische Daten

Je nach Neigung, schließt ein Schalter die Eingangspins kurz.

Pin-Belegung



Codebeispiel

Hier bei handelt es sich um ein Beispielprogramm, welches eine Nachricht in die serielle Ausgabe sendet, wenn am Sensor ein Signal detektiert wurde.

Anschlussbelegung:

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 Neigung und dies wird als Meldung ausgegeben
    Serial.println ("Taster gedrückt");
  }

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

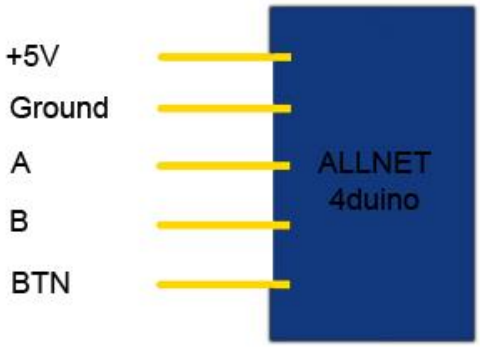
B16 Rotary Encoder / Kodierter Drehschalter

Bild

Kurzbeschreibung / Technische Daten

Die aktuelle Position des Drehschalters wird kodiert über die Ausgänge gegeben.

Pin-Belegung



Kodierung

Die Idee bei einem Drehschalter/Drehgeber ist es, dass zu jedem gedrehten "Schritt", sich der Zustand jeweils immer nur einer der beiden Ausgangs-Pins ändert. Je nachdem welhcer der beiden sich zuerst geändert hat, so kann man auf die Drehrichtung schließen, wenn man auf die folgende Kodierung achtet.

Dieser Sensor gibt somit keine absoluten Werte aus (z.B. genau gemessene Temperatur in °C oder Magnetfeldstärke in mT) , sondern es handelt sich hierbei um eine Relativ-Messung: Man definiert hierbei einen Grenzwert relativ zur gegebenen normalen Umweltsituation und es wird ein Signal ausgegeben was weiterverarbeitet werden kann, falls dieser Grenzwert überschritten bzw. ein anderer Zustand als der Normalfall eingetreten ist.

Im Uhrzeigersinn [A ändert sich zuerst] -> Pin_A

A	B
0	0
1	0
1	1
0	1
0	0

Gegen den Uhrzeigersinn [B ändert sich zuerst] -> Pin_B

A	B
0	0
0	1
1	1
1	0
0	0

Codebeispiel

Das Programm überprüft, falls eine Änderung der Pin-Zustände sich ereignet hat, welcher der beiden Pins sich zuerst geändert hatte, was auf die Drehrichtung schließen lässt. Diese Information erhält man, in dem man einen der beiden Pin-Werte aus einem vorherigen Durchlauf mit dem Wert des aktuellen Durchlaufs vergleicht.

Nachdem die Richtung festgestellt wurde, werden die Schritte von der Startposition an gezählt und ausgegeben. Ein Drücken auf den Knopf des Drehgebers resettet die aktuelle Position.

Anschlussbelegung:

Sensor +V	= [Pin 5V]
Sensor GND	= [Pin GND]
Sensor A	= [Pin 3]
Sensor B	= [Pin 4]
Sensor BNT	= [Pin 5]

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


//Deklarieren der benötigten Variablen
int Counter = 0;
boolean Richtung;
int Pin_A_Letzter;
int Pin_A_Aktuell;

int pin_A = 3;
int pin_B = 4;
int button_pin = 5;

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

  //Pull-Up Widerstände werden aktiviert
  digitalWrite(pin_A, true);
  digitalWrite(pin_B, true);
  digitalWrite(button_pin, true);

  //Initiales Auslesen des Pin_A
  Pin_A_Letzter = digitalRead(pin_A);

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

//dauerhaft wiederholte Hauptschleife
void loop()
{
  //Auslesen des aktuellen Statuses
  Pin_A_Aktuell = digitalRead(pin_A);

  //Überprüfung auf Änderung
  if (Pin_A_Aktuell != Pin_A_Letzter)
  {
    if (digitalRead(pin_B) != Pin_A_Aktuell)
    {

      //Pin_A hat sich zuerst verändert
      Richtung = true;
      Counter ++;
    }

    else
    {
      //Andernfalls hat sich Pin_B zuerst verändert
      Richtung = false;
      Counter--;
    }
```

```
//Ausgabe der Drehrichtung und der aktuelen Position
    Serial.println ("Drehung erkannt: ");
    Serial.print ("Drehrichtung: ");

    if (Richtung)
    {
      Serial.println ("Im Uhrzeigersinn");
    }

    else
    {
      Serial.println("Gegen den Uhrzeigersinn");
    }

    Serial.print("Aktuelle Position: ");
    Serial.println(Counter);
    Serial.println("------------------------------");

  }

  //Der Wert des aktuellen Durchlaufs ist beim
nächsten Druchlauf der vorherige Wert
  Pin_A_Letzter = Pin_A_Aktuell;

  //Wenn der Button gedrückt wird und die Position
nicht 0 ist
  if (!digitalRead(button_pin) && Counter != 0)
  {
    //Dann setze die Position zurrück
    Counter = 0;

    //Ausgaber Information des Resets
    Serial.println("Position resettet");
  }

}
```

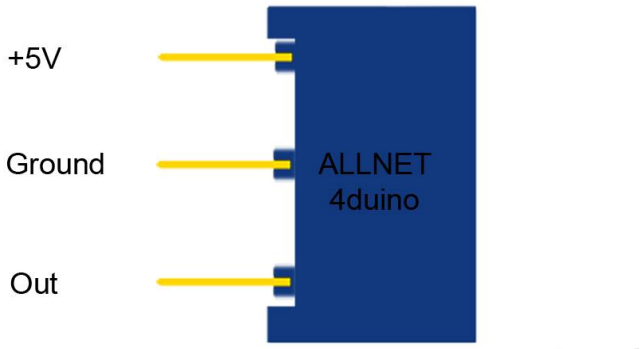
B18 Light Barrier / Lichtschranke

Bild

Kurzbeschreibung / Technische Daten

Es wird über den OUT Pin ein Signal ausgesendet, falls die Lichtschranke unterbrochen wird.

Pin-Belegung



Codebeispiel

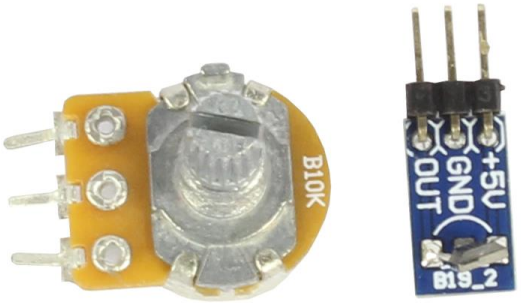
Hier bei handelt es sich um ein Beispielprogramm, wlches eine serielle Ausgabe auslöst, wenn am Sensor ein Signal detektiert wurde.
Anschlussbelegung:

Sensor +V = [Pin 5V]
Sensor GND = [Pin GND]
Sensor Digital = [Pin 10]

```
// ALLNET Tilt Light Barrier/Lichtschranke B18
// 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 Unterbrechung erkannt und dies wird als Meldung ausgegeben
    Serial.println ("Lichtschranke durchbrochen");
  }

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

B19 Potentiometer / Analog Hall

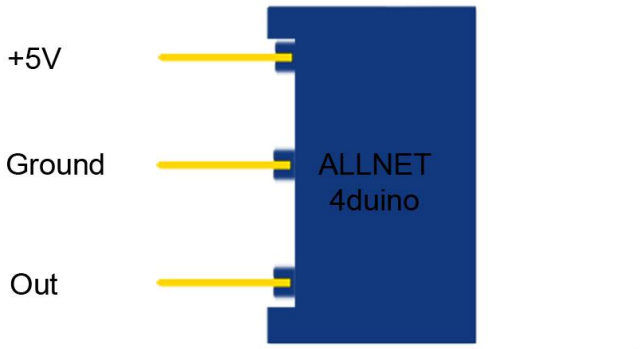
Bild



Kurzbeschreibung / Technische Daten

Der Wiederstand 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 Wiederstang jedoch nach dem magnetischen Umfeld.

Pin-Belegung



Codebeispiel

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

Anschlussbelegung:

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

```
// 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);
```

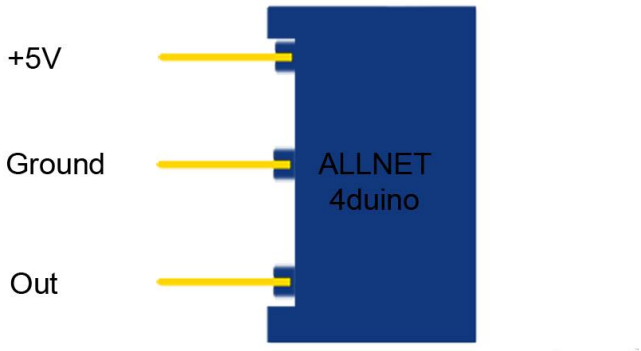
B20 Temperatur Sensor Modul

Bild

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:

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);
}
```

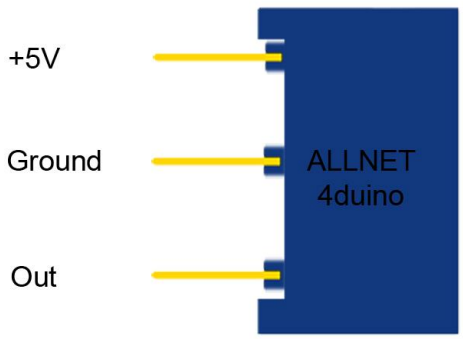
B21 IR LED / Infrarot LED

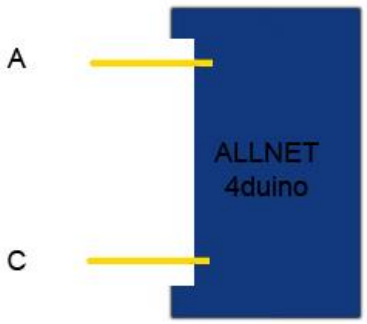
Bild

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:

- [Arduino-IRRemote]

Die Librarie kann direkt in der Arduino IDE installiert werden. Sie steht im Library Maneger zum download bereit.

Bei Infrarot-Sendesystemen, gibt es verschiedene Protokolle, in denen die Daten versendet werden können. In dem folgenden Beispiel wird für das versenden das RC5 Protokoll verwendet - die verwendete Library "Arduino-IRremote" kümmert sich eigenständig um die Konvertierung in die richtige Datenfolge. Es gibt innerhalb der Library jedoch auch andere Protokolle/Kodierungen - diese sind in der Dokumentation/Code der Library gekennzeichnet.

Code für den Empfänger (B22):

Anschlussbelegung:

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:

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

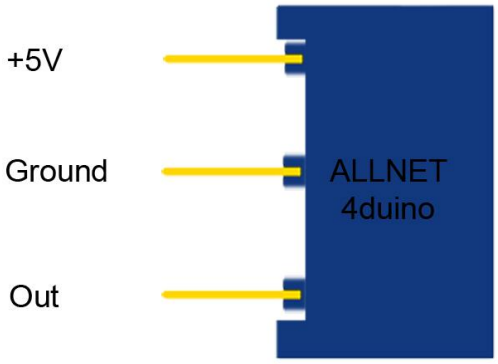
Bild

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:

- [Arduino-IRremote]

Die Librarie kann direkt in der Arduino IDE installiert werden. Sie steht im Library Maneger zum download bereit.

Bei Infrarot-Sendesystemen, gibt es verschiedene Protokolle, in denen die Daten versendet werden können. In dem folgenden Beispiel wird für das versenden das RC5 Protokoll verwendet - die verwendete Library "Arduino-IRremote" kümmert sich eigenständig um die Konvertierung in die richtige Datenfolge. Es gibt innerhalb der Library jedoch auch andere Protokolle/Kodierungen - diese sind in der Dokumentation/Code der Library gekennzeichnet.

Code für den Empfänger (B22):

Anschlussbelegung:

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:

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);
  }
}
```

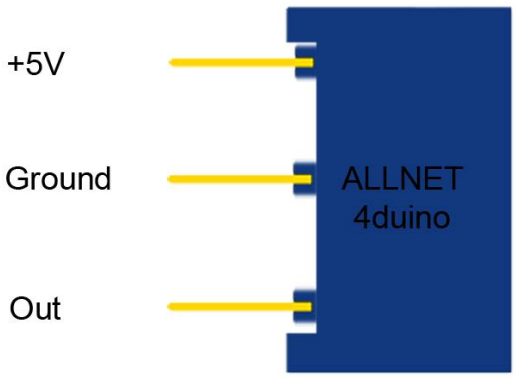
B23 Shock Sensor / Schocksensor

Bild

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 Erschürrerung erkannt wird.
Anschlussbelegung:

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

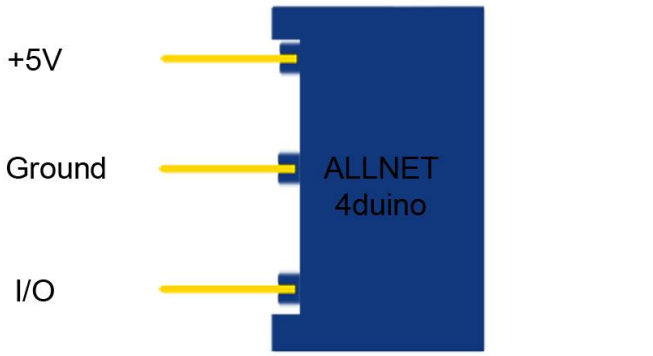
Bild

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:

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

```
// ALLNET Temperature & Humidity/Temp. & Feuchtigkeit
B24
// Information http://www.allnet.de


//Arduino-DHT Iibrary wird hinzugefuegt
#include "DHT.h"

// Arduino-DHT Library wird initialisiert
#define DHTPIN 2
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

//einmalig ausgeführte SetUp Befehle
void setup()
{
  //Starten der seriellen Übertragung
  Serial.begin(9600);
  Serial.println("DHTxx test!");

  //DHT11 Sensor wird gestartet
  dht.begin();
}

void loop()
{
  //Pause
  delay(2000);

  //Auslesen der Luftfeuchtigkeit
  float h = dht.readHumidity();
```

```
//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");
}
```

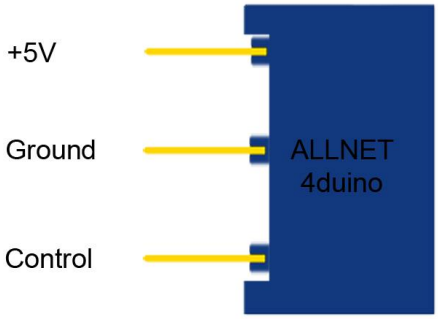
B25 1 Watt LED Module

Bild

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:

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);
}
```

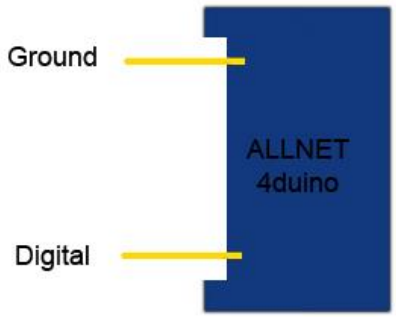
B26 Piezo Speaker

Bild

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:

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);
}
```

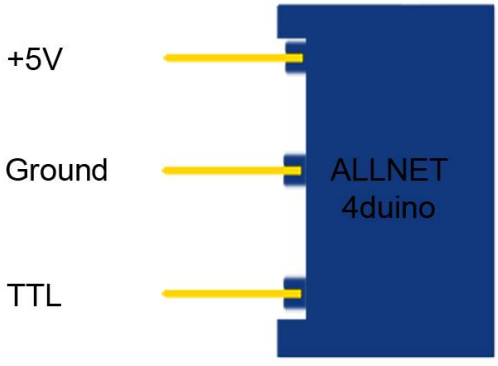
B27 Buzzer

Bild

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:

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);
}
```

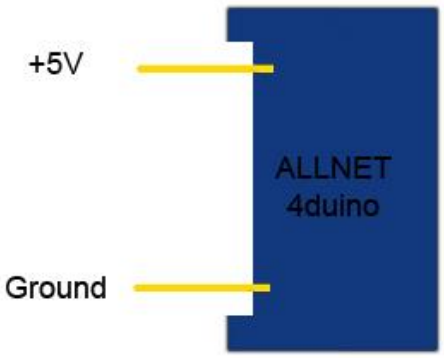
B28 Flash LED

Bild

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.

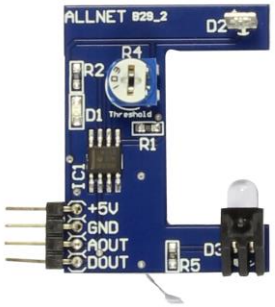
Anschlussbelegung:

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

Bild



Kurzbeschreibung / Technische Daten

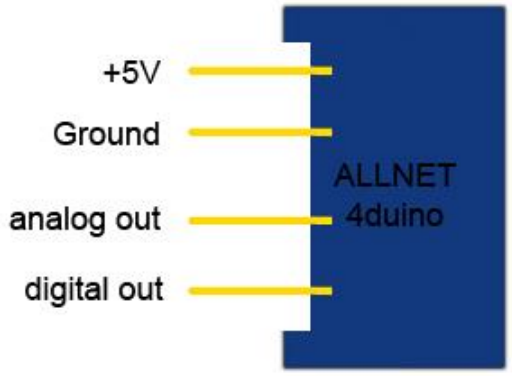
Digitaler Ausgang: Wird ein Herzschlag erkannt, wird hier ein Signal ausgegeben

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



ALLNET 4duino

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:

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;
  int digital;

//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);
}
```

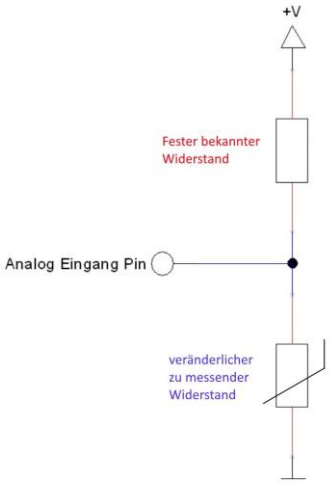
B30 Photoresistor / Lichtsensor

Bild

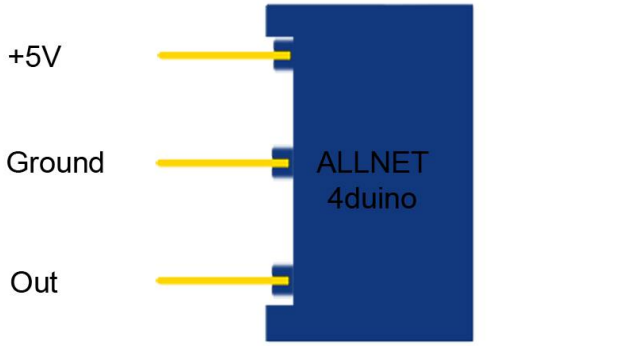
Kurzbeschreibung / Technische Daten

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

Dieser Widerstand lässt sich mit Hilfe eines Spannungsteilers bestimmen, wo sich eine bekannte Spannung über einen bekannten und einen unbekannten (veränderlichen) Widerstand aufteilt. Mittels dieser gemessenen Spannung lässt sich dann der Widerstand berechnen - die genaue Berechnung ist in den unten stehenden Codebeispielen enthalten.



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.

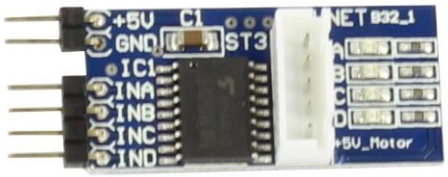
Anschlussbelegung:

Sensor V+ = [Pin +5V]
Sensor GND = [Pin GND]
Sensor Analog = [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

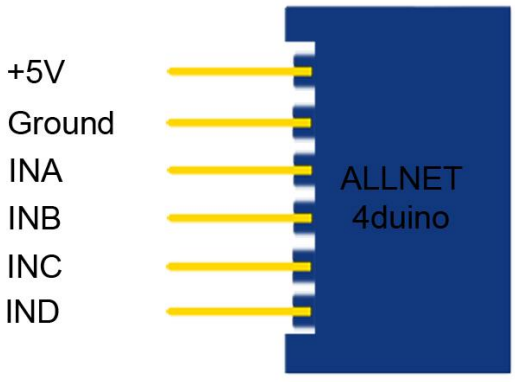
Bild



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:

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);
}
```

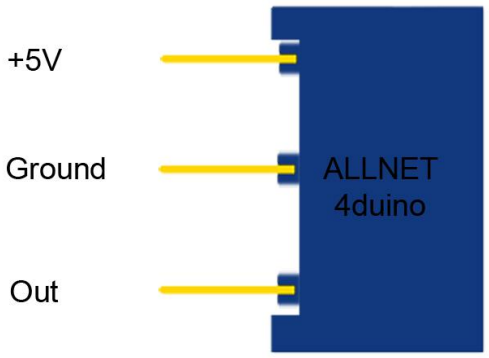
B33 Gas MQ2 Sensor

Bild

Kurzbeschreibung / Technische Daten

Der Sensor legt unterschiedliche Spannung an den 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 Sensots auf der seriellen Anzeige ausgibt.

Anschlussbelegung:

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);
}
```

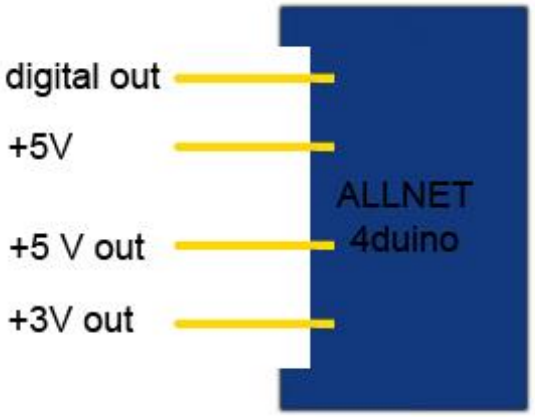
B34 Voltage Regulator linear

Bild

Kurzbeschreibung / Technische Daten

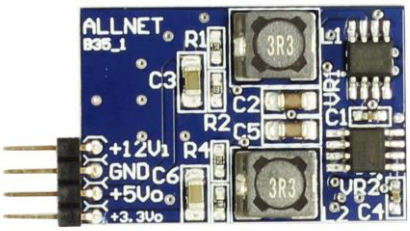
Der ALLNET B34 ist ein linearer Spannungs regulierer. Er senkt eine anliegende 12V Spannung auf 5 und 3 Volt. Vareiert die Eingangsspannung, so ändern sich die Ausgangsspannungen in gleichem Maße.

Pin-Belegung



B35 Voltage Regulator

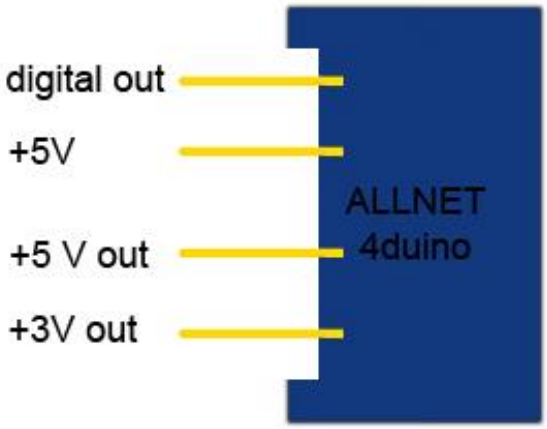
Bild



Kurzbeschreibung / Technische Daten

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

Pin-Belegung



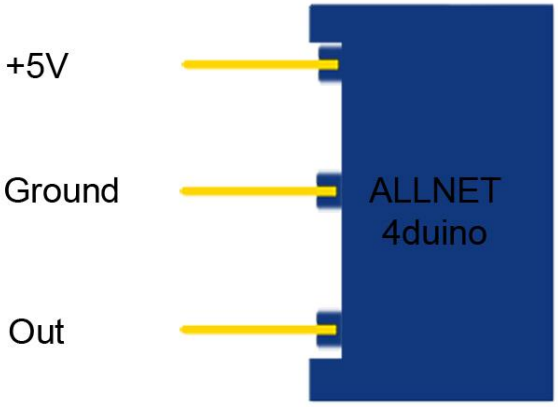
B36 Motion Detection / Bewegungsmelder

Bild

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:

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);
}
```

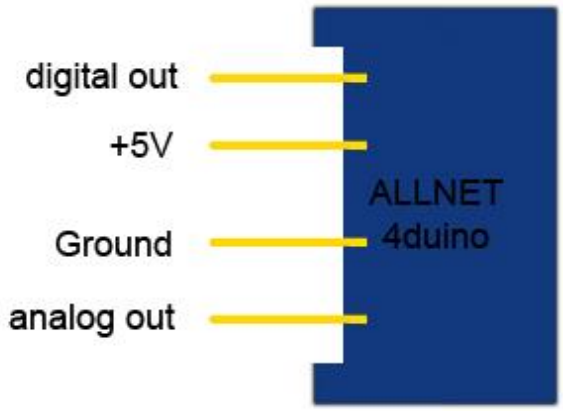
B37 8 LED PCB / 8-fach LED PCB

Bild

Kurzbeschreibung / Technische Daten

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

Pin-Belegung



ALLNET 4duino

Codebeispiel

Das Beispielprogramm simuliert eine fallende Akkuanzeige.

Anschlussbelegung:

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);
}
```

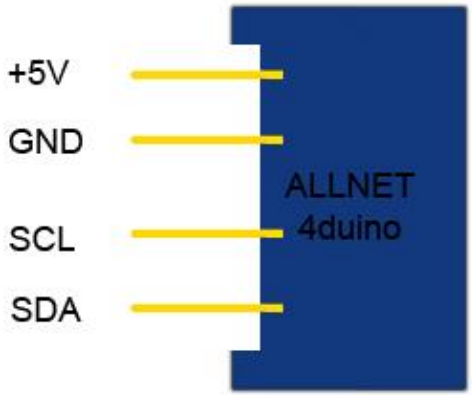
B38 Temperature I2C / Temperatur I2C Sensor

Bild

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:

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;
}
```

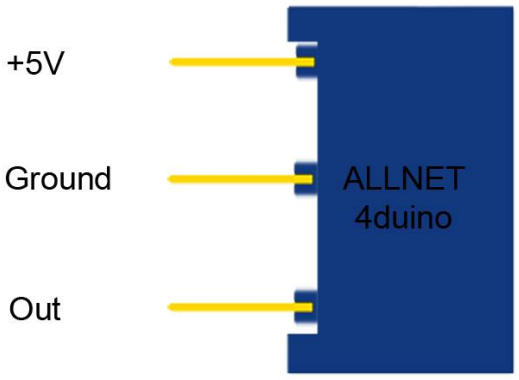
B39 Vibration Sensor / Vibrations Sensor

Bild

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:

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);
}
```