Montage und Programmierung eines Roboters für den Hessen SolarCup

Disziplin: SolaRobot

Teil 2.2: US/Interrupt

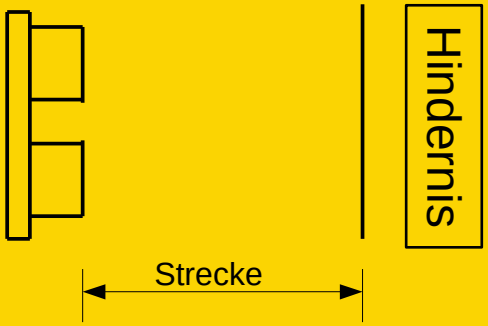
INTERRUPT

„Paralleler“ Programmablauf

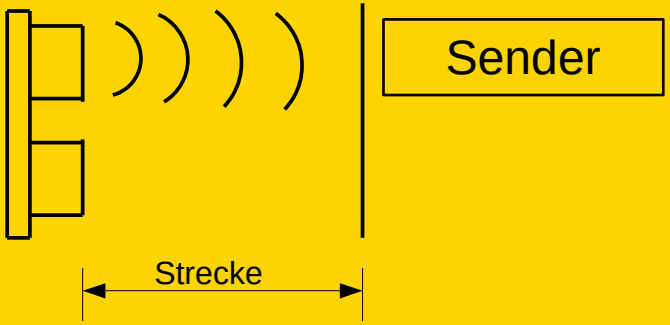
Ultraschall-Sensor

Ultraschallsensor
von oben gesehen

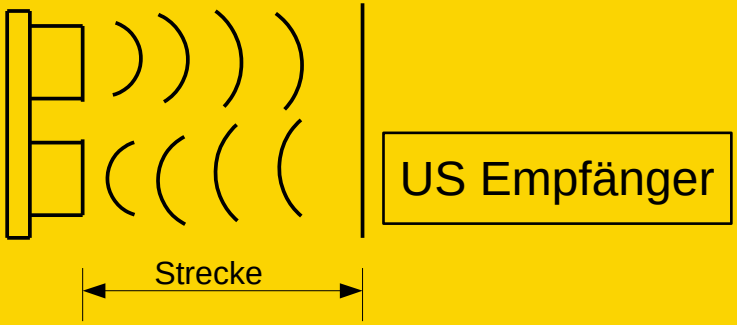
Ultraschall-Sensor



Ultraschall-Sensor

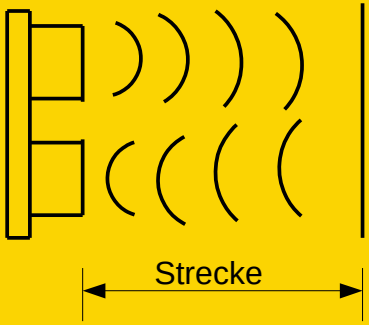


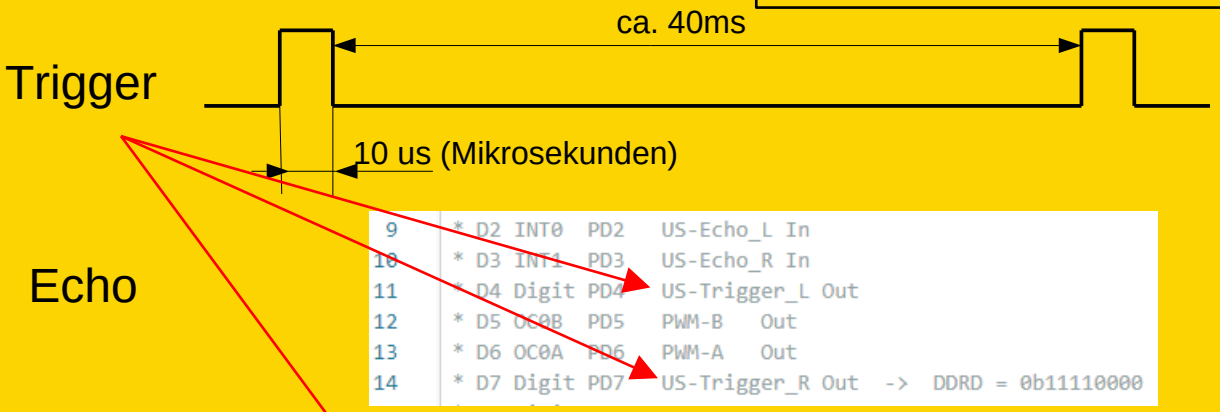
Ultraschall-Sensor



Ultraschall-Sensor

Alle 40ms (Millisekunden) kann ein neuer Wert generiert werden.

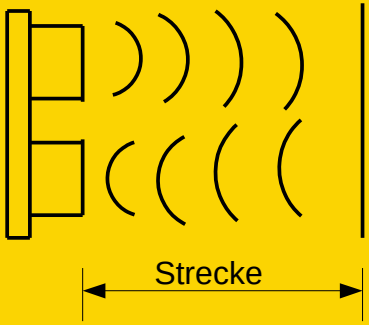


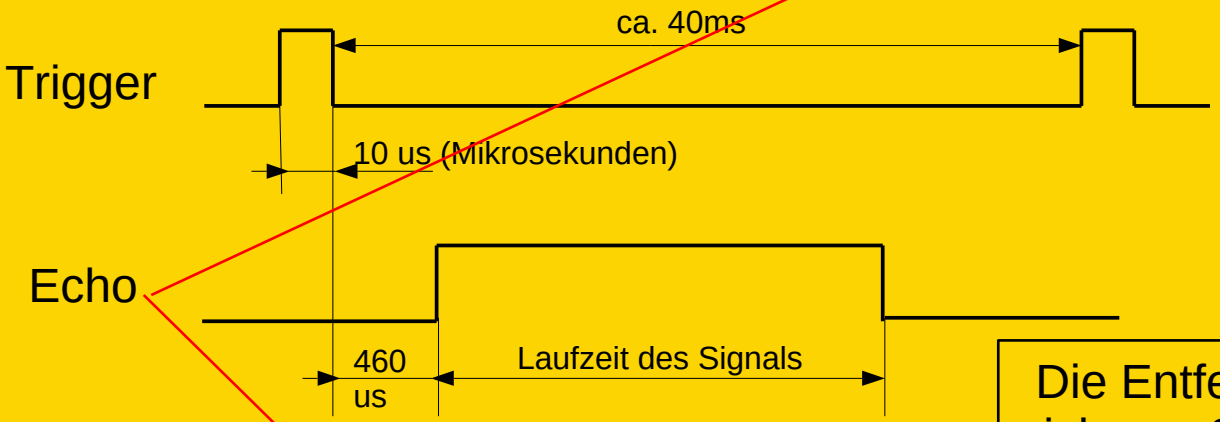



```
 9    * D2 INT0  PD2   US-Echo_L In
10    * D3 INT1  PD3   US-Echo_R In
11    * D4 Digit PD4   US-Trigger_L Out
12    * D5 OC0B  PD5   PWM-B    Out
13    * D6 OC0A  PD6   PWM-A    Out
14    * D7 Digit PD7   US-Trigger_R Out  ->  DDRD = 0b11110000
```

Ultraschall-Sensor

```
 9   * D2 INT0  PD2   US-Echo_L In
10   * D3 INT1  PD3   US-Echo_R In
11   * D4 Digit PD4   US-Trigger_L Out
12   * D5 OC0B  PD5   PWM-B    Out
13   * D6 OC0A  PD6   PWM-A    Out
14   * D7 Digit PD7   US-Trigger_R Out  ->  DDRD = 0b11110000
```

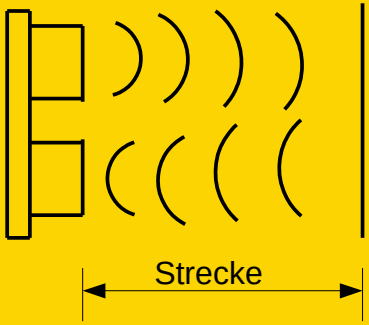


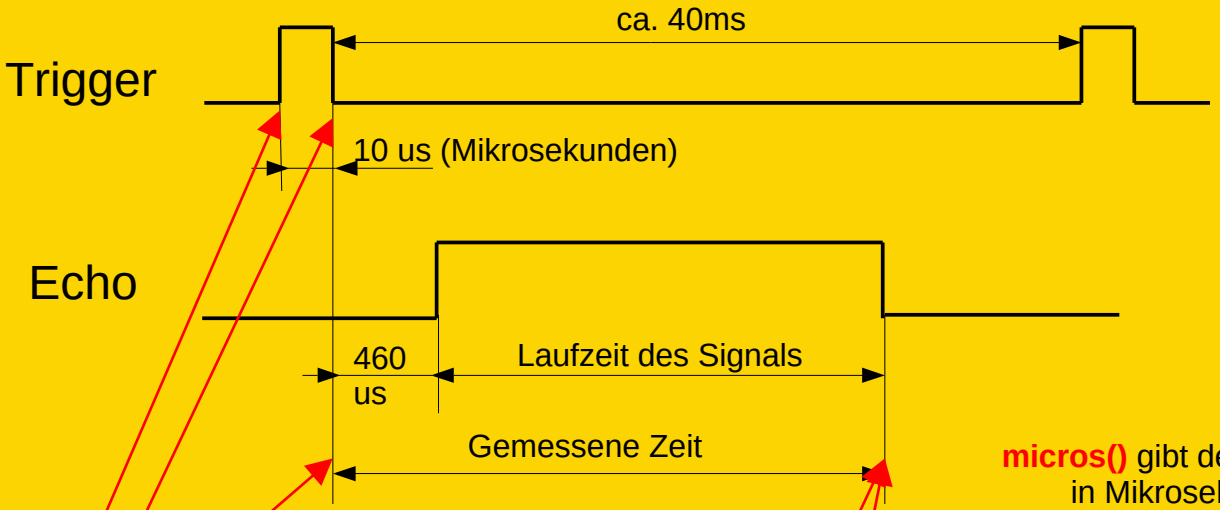


Die Entfernung berechnet sich aus Sendezeit (460us) und Laufzeit des Signals.

Bei einigen US-Sensoren beträgt die Sendezeit 2250us!

Ultraschall-Sensor





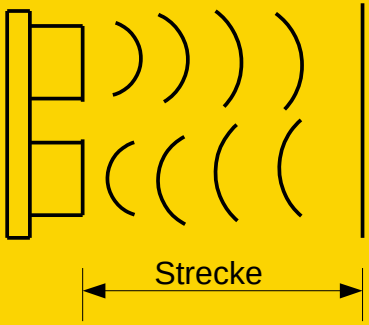
micros() gibt den Wert der inneren Uhr in Mikrosekunden (us) zurück.

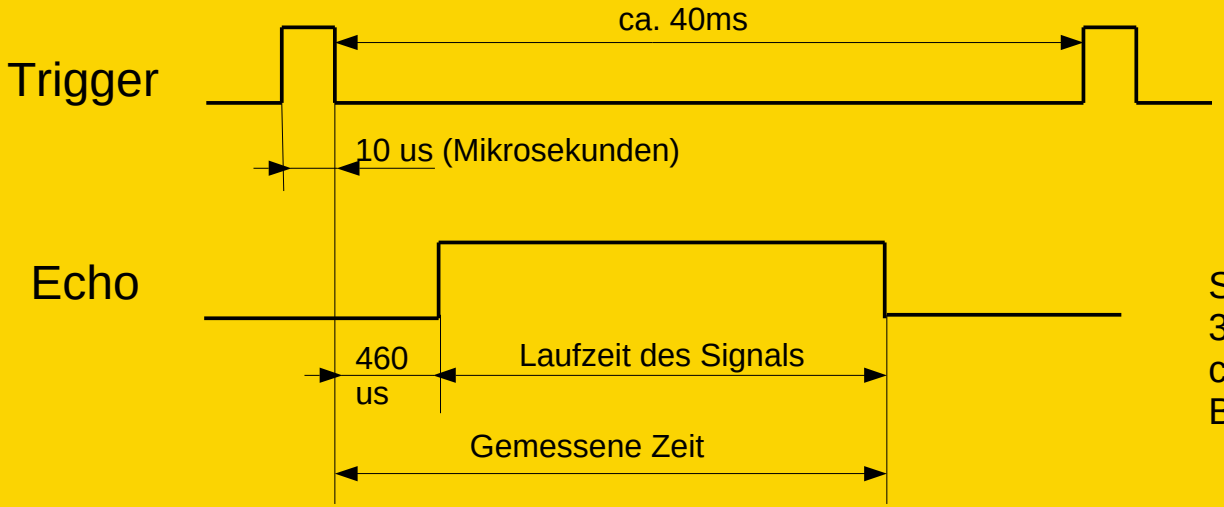
```
172  void US_Trigger (void) {
173    digitalWrite(US_Trig_L_Pin, HIGH); //US-Triggerpin auf +5V ziehen
174      digitalWrite(US_Trig_R_Pin, HIGH); //US-Triggerpin auf +5V ziehen
175      delayMicroseconds(10); //10Mikrosekunden warten
176      digitalWrite(US_Trig_L_Pin, LOW); //US-Triggerpin wieder auf GND ziehen
177      digitalWrite(US_Trig_R_Pin, LOW); //US-Triggerpin wieder auf GND ziehen
178      US_Time_Start_L = micros(); //Startzeit abspeichern
179      US_Time_Start_R = micros(); //Startzeit abspeichern
180  }
```

```
void US_Links (void) {
  US_Wert_L = ((micros() - US_Time_Start_L) - 2250) / 59; //Abstand links berechnen
}
```

```
void US_Rechts (void) {
  US_Wert_R = ((micros() - US_Time_Start_R) - 2250) / 59; //Abstand rechts berechnen
}
```

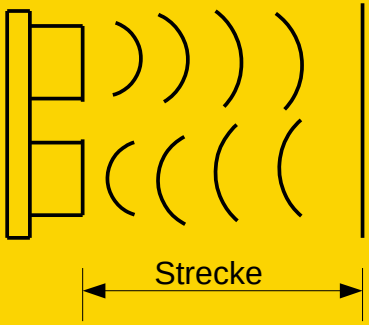
Ultraschall-Sensor

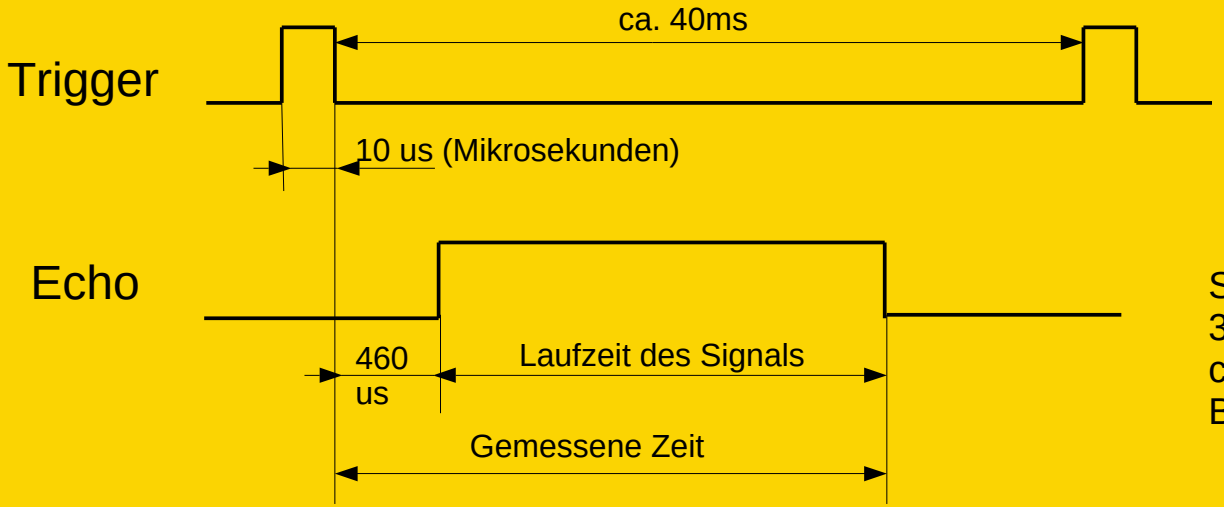




Schallgeschwindigkeit
343m/s = 1236km/h
ca. 3s für 1km
Blitz - Donner

Ultraschall-Sensor

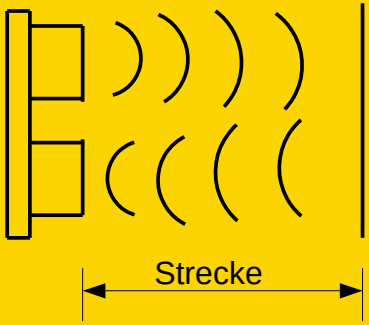


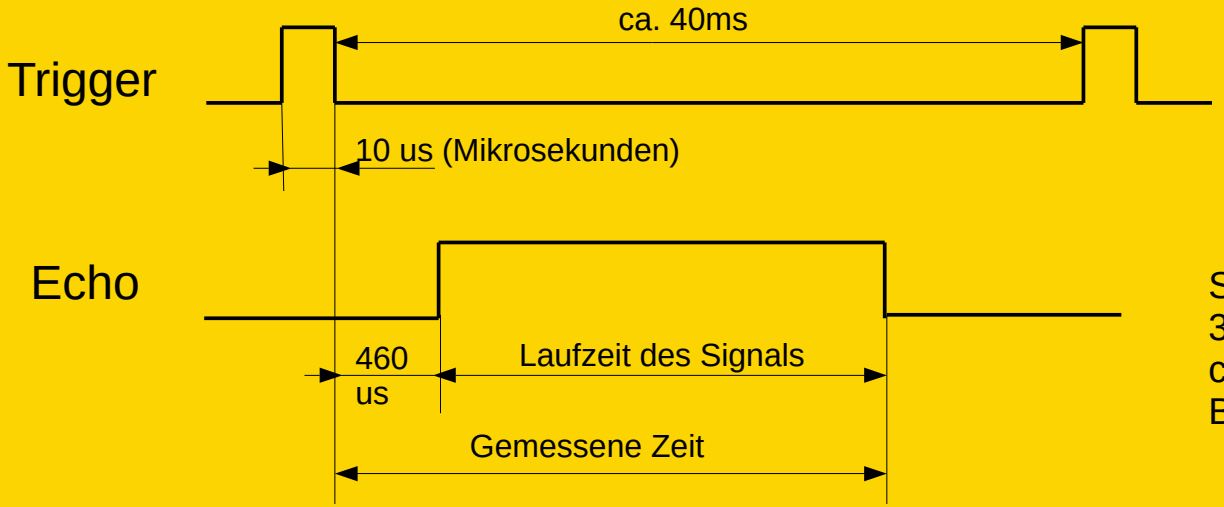


Schallgeschwindigkeit
343m/s = 1236km/h
ca. 3s für 1km
Blitz - Donner

Rechenbeispiel:
Strecke = 10cm → 0,1m

Ultraschall-Sensor

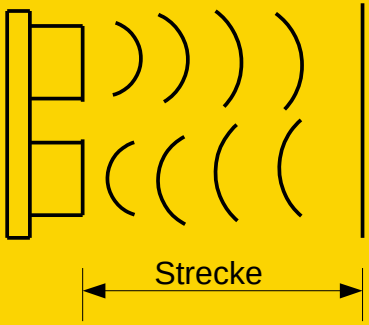


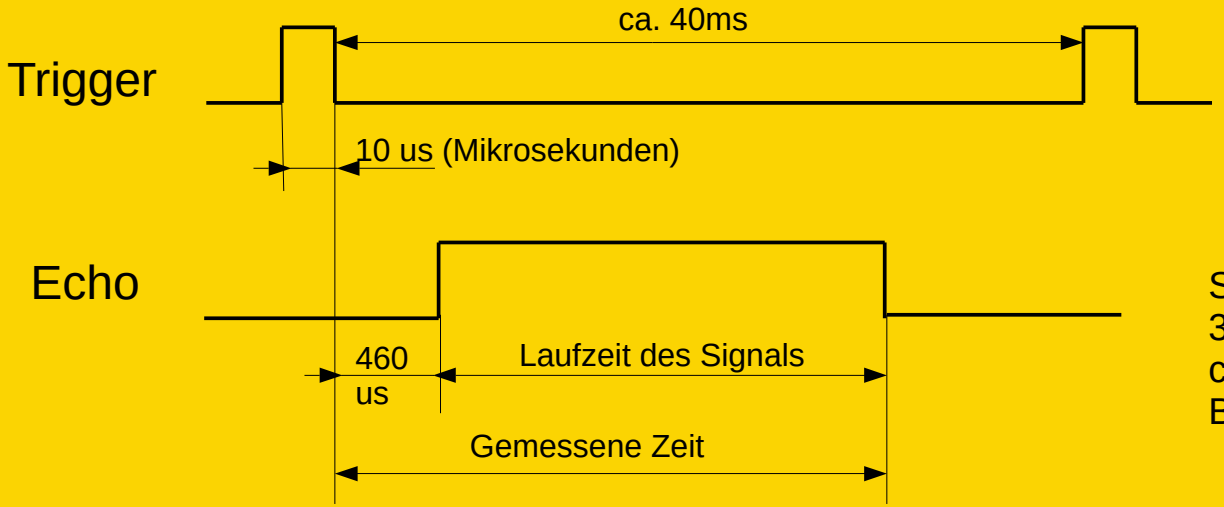


Schallgeschwindigkeit
343m/s = 1236km/h
ca. 3s für 1km
Blitz - Donner

Rechenbeispiel:
Strecke = 10cm → 0,1m
0,1m / 343m/s = 290us → 29us/cm

Ultraschall-Sensor

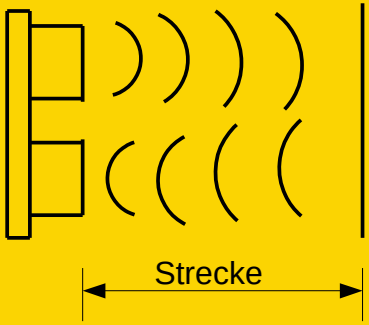


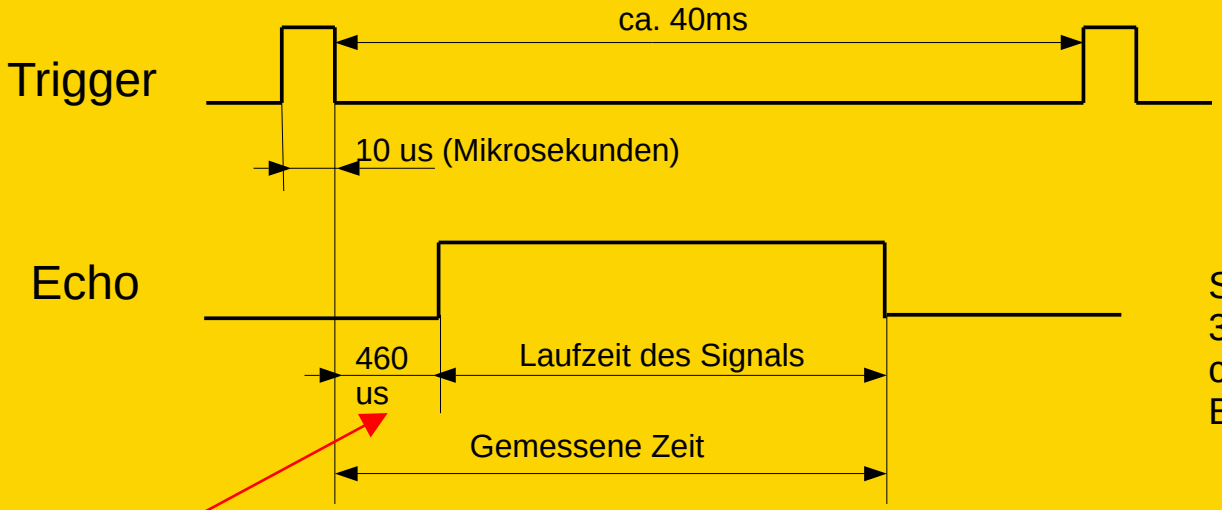


Schallgeschwindigkeit
343m/s = 1236km/h
ca. 3s für 1km
Blitz - Donner

Rechenbeispiel:
Strecke = 10cm → 0,1m
0,1m / 343m/s = 290us → 29us/cm
290us * 2 = 580us (läuft 2mal die Strecke)

Ultraschall-Sensor

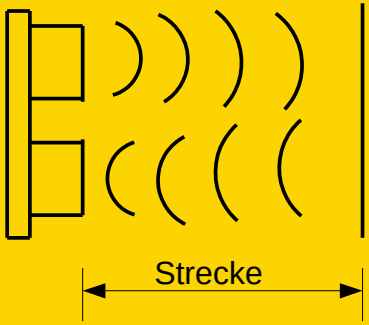


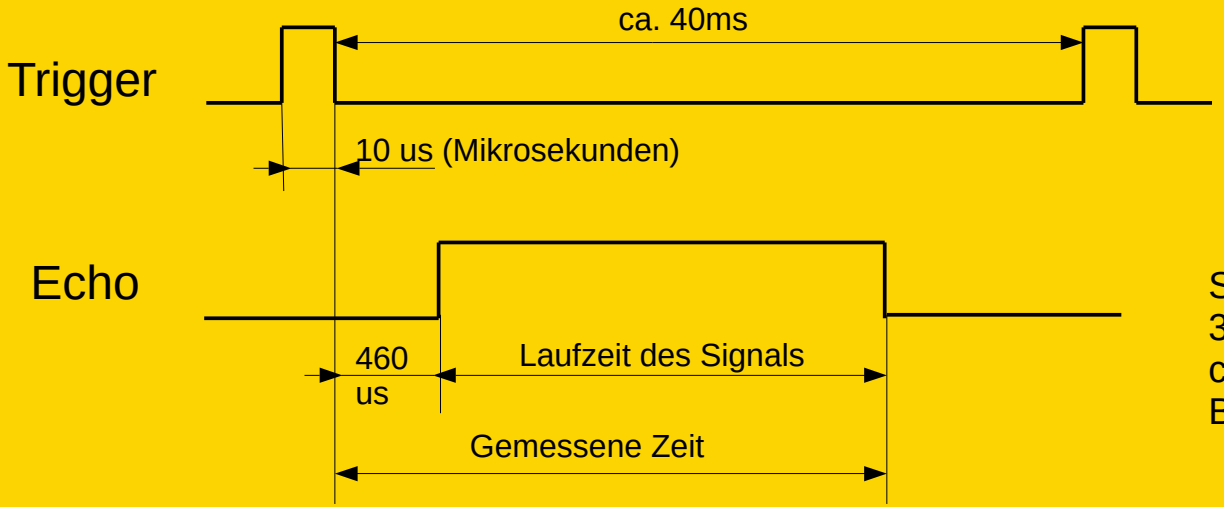


Schallgeschwindigkeit
343m/s = 1236km/h
ca. 3s für 1km
Blitz - Donner

Rechenbeispiel:
Strecke = 10cm → 0,1m
0,1m / 343m/s = 290us → 29us/cm
290us * 2 = 580us (läuft 2mal die Strecke)
580us + 460us = 1040us

Ultraschall-Sensor



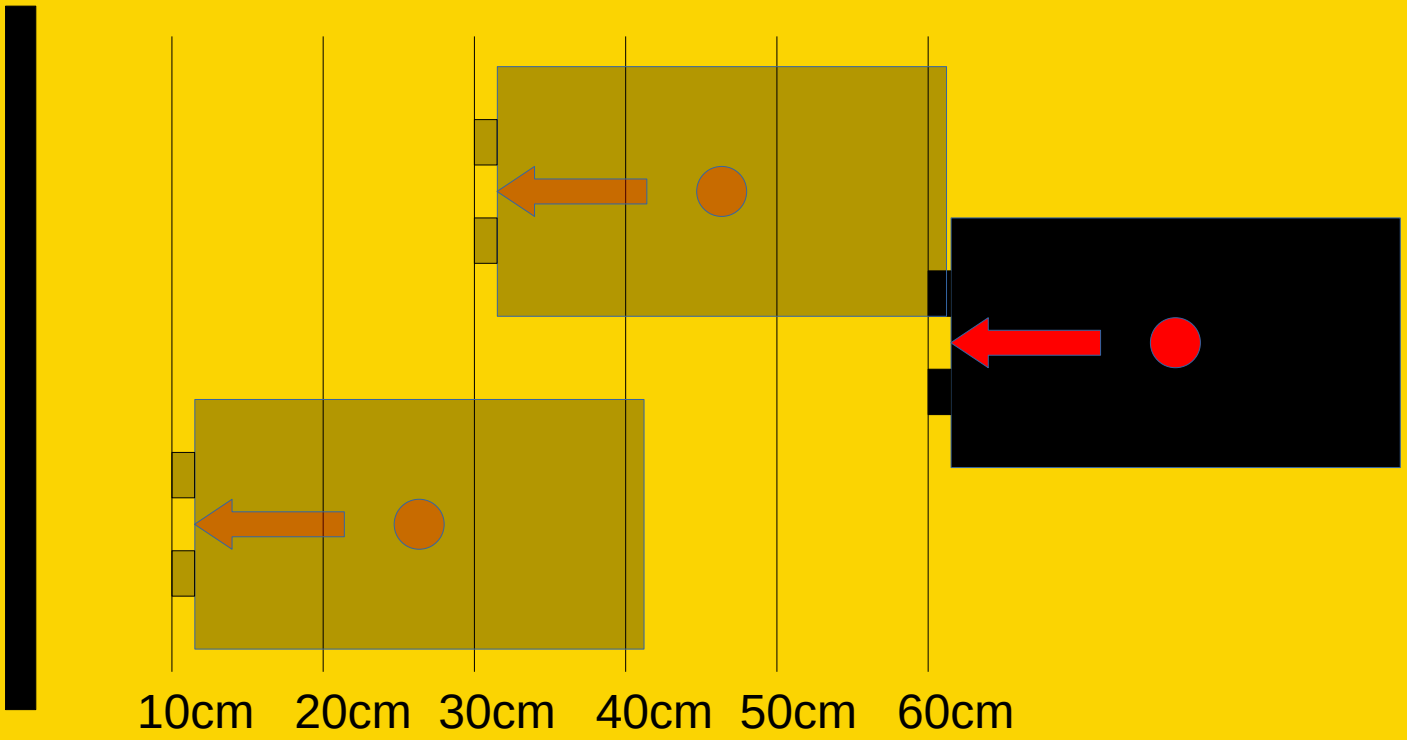


Schallgeschwindigkeit
343m/s = 1236km/h
ca. 3s für 1km
Blitz - Donner

Rechenbeispiel:
Strecke = 10cm → 0,1m
0,1m / 343m/s = 290us → 29us/cm
290us * 2 = 580us (läuft 2mal die Strecke)
580us + 460us = 1040us

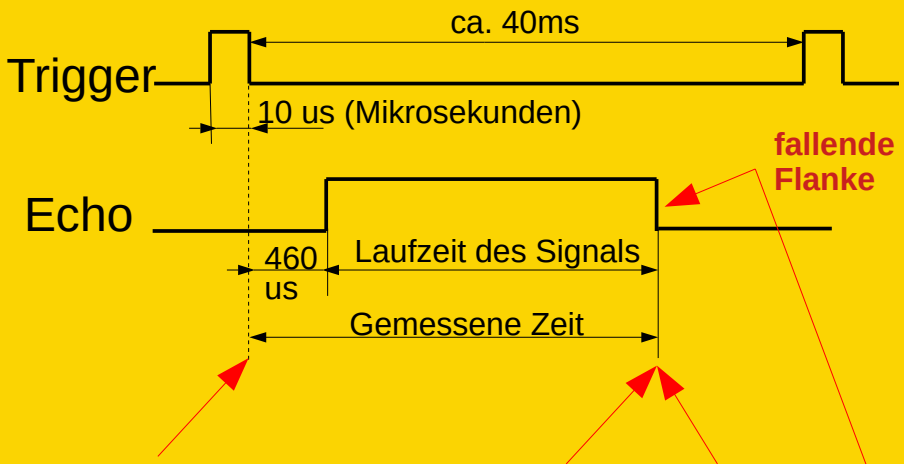
US_Wert_L = ((micros() - US_Time_Start_L) - 460) / 59
micros() = „1040“, US_Time_Start_L = „0“
(1040 – 460) / 59 = 9,83 -> ungefähr 10
micros() ist natürlich nicht 1040, sondern 1040 größer als US_Time_Start_L.

Ultraschall-Sensor



Abstand[cm]	US Wert
10	
20	
30	
40	
50	
60	

Aufgabe: Ermittelt die angezeigten Werte
für unterschiedliche Entfernungen zum Hindernis.



US_Time_Start_L = micros();
US_Time_Start_R = micros();
Anfangszeit wird gespeichert

Interrupt wird ausgelöst

Timer 1 wird ausgelesen

```
77    //Externe Interrupts
78    attachInterrupt(digitalPinToInterrupt(US_Echo_L_Pin), US_Links, FALLING);
79    attachInterrupt(digitalPinToInterrupt(US_Echo_R_Pin), US_Rechts, FALLING);
```

```
void US_Links (void) {
  US_Wert_L = ((micros() - US_Time_Start_L) - 2250) / 59; //Abstand links berechnen
}
```

```
void US_Rechts (void) {
  US_Wert_R = ((micros() - US_Time_Start_R) - 2250) / 59; //Abstand rechts berechnen
}
```

03_Interrupt_US | Arduino IDE 2.3.8

Datei Bearbeiten Sketch Werkzeuge Hilfe

Arduino Nano

03_Interrupt_US.ino Drive_Functions.h Init.h

```
127  //Unterschiedliche Uhren
128  void Clock (void) {
129    if (millis() >= (Time_absolute_old + 1)) { //Jede Millisekunde
130      Time_absolute_old = millis();
131      Time_BNO055++; //Uhrzeit um 1 erhöhen
132      Time_Mag++; //Uhrzeit um 1 erhöhen
133      Time_Serial_Print++; //Uhrzeit um 1 erhöhen
134      Time_US++; //Uhrzeit um 1 erhöhen
135      Time_LED++;
136    }
137    //1000 bis 2000us:
138    if (Time_BNO055 == 5) { //Alle 10 Millisekunden, versetzt zum Magnetsensor um 5ms
139      Gyro_Values(); //Gyroskop Werte ermitteln, X-Achse
140      Time_BNO055 = -5; //Bei -5 anfangen zu zählen (bis 5 sind dann 10 Stufen)
141    }
142    //1000us:
143    if (Time_Mag == 10) { //Alle 10 Millisekunden, versetzt zum Gyroskop um 5ms
144      Mag_Values(); //Magnet-Werte ermitteln
145      Time_Mag = 0; //Zähler wieder auf Null
146    }
147    //400us:
148    if (Time_Ser
149      Time_Seria
150      Data_Visua
151      Serial.pri
152      Serial.pri
153      Index_Data
154      if (Index_
155        Index_Da
156        Serial.p
157      }
158    }
159    //12us
160    if (Time_US == 40) { //Alle 40ms einen neuen US-Wert initieren
161      Time_US = 0; // Zähler wieder auf Null
162      US_Trigger();
163    }
164    if (Toggle_On_Off) {
165      if (Time_LED == 250) { //alle 256ms togglet die LED
166        Time_LED = 0;
167        LED_TOGGLE;
168      }
169    }
170  }
```

```
172  void US_Trigger (void) {
173    digitalWrite(US_Trig_L_Pin, HIGH); //US-Triggerpin auf +5V ziehen
174    digitalWrite(US_Trig_R_Pin, HIGH); //US-Triggerpin auf +5V ziehen
175    delayMicroseconds(10); //10Mikrosekunden warten
176    digitalWrite(US_Trig_L_Pin, LOW); //US-Triggerpin wieder auf GND ziehen
177    digitalWrite(US_Trig_R_Pin, LOW); //US-Triggerpin wieder auf GND ziehen
178    US_Time_Start_L = micros(); //Startzeit abspeichern
179    US_Time_Start_R = micros(); //Startzeit abspeichern
180  }
```

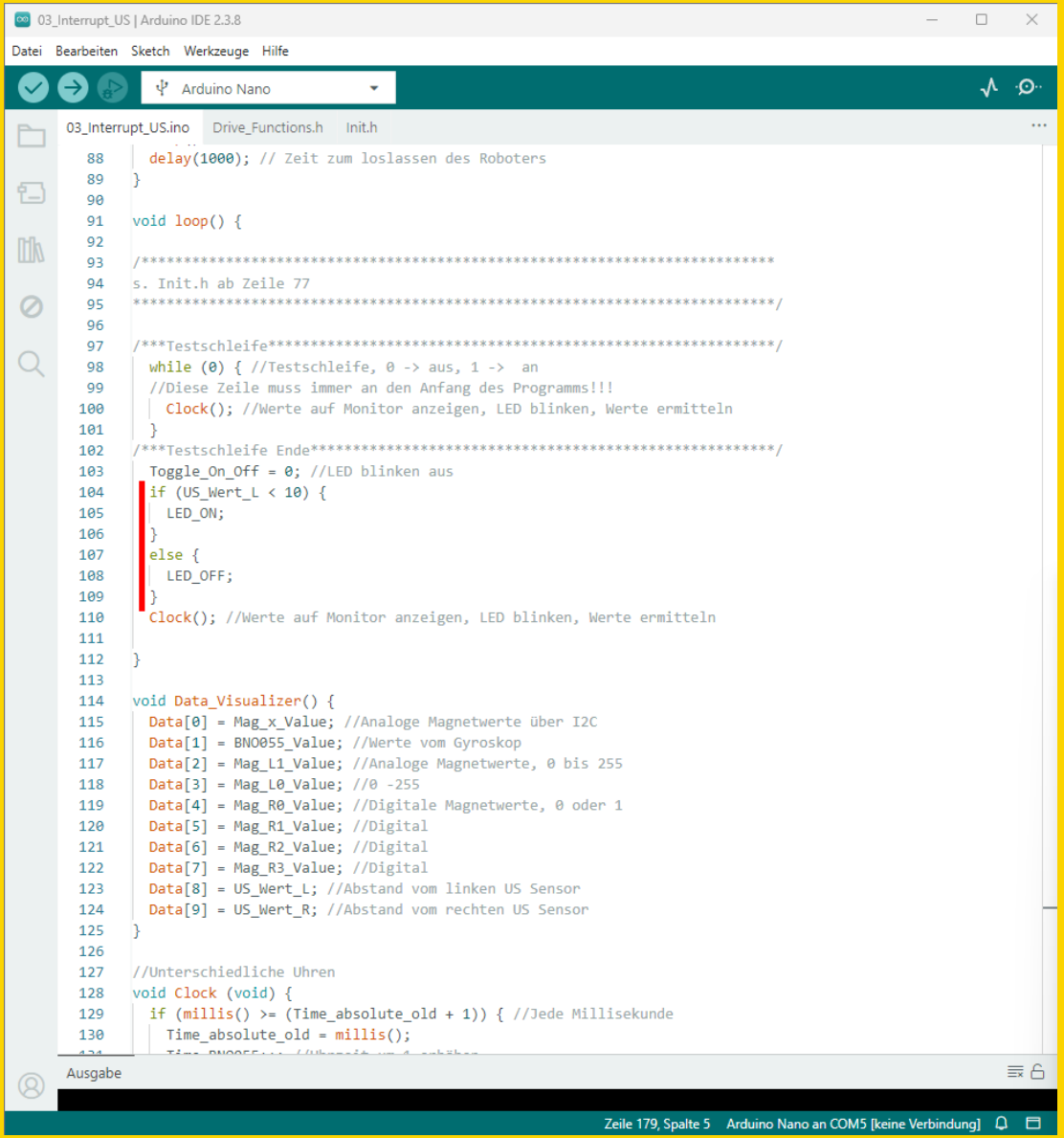
Ausgabe

Zeile 126, Spalte 1 Arduino Nano an COM5 [keine Verbindung]

if:
Falls Bedingung erfüllt ist, mache etwas
else:
Sonst mache etwas anderes

Syntax:
if(Bedingung) {
Anweisungen
}
else {
Andere Anweisungen
}

Bedingung ist ein Wert.
Ist die Aussage falsch ist er Null.
Ist die Aussage richtig ist er Eins.
Die Bedingung ist erfüllt, wenn sie **NICHT Null** ist!



Montage und Programmierung eines Roboters für den Hessen SolarCup Disziplin: SolaRobot Teil 2.3: Magnetsensor/Motor