



HARDWARE – SENSOREN

MKR1010

OKTOBER 2020

VORTRAGENDE: ARMIN FISCHER
MARTIN SCHUBERT

E-MAIL: FIA@HTLWRN.AC.AT
SUM@HTLWRN.AC.AT

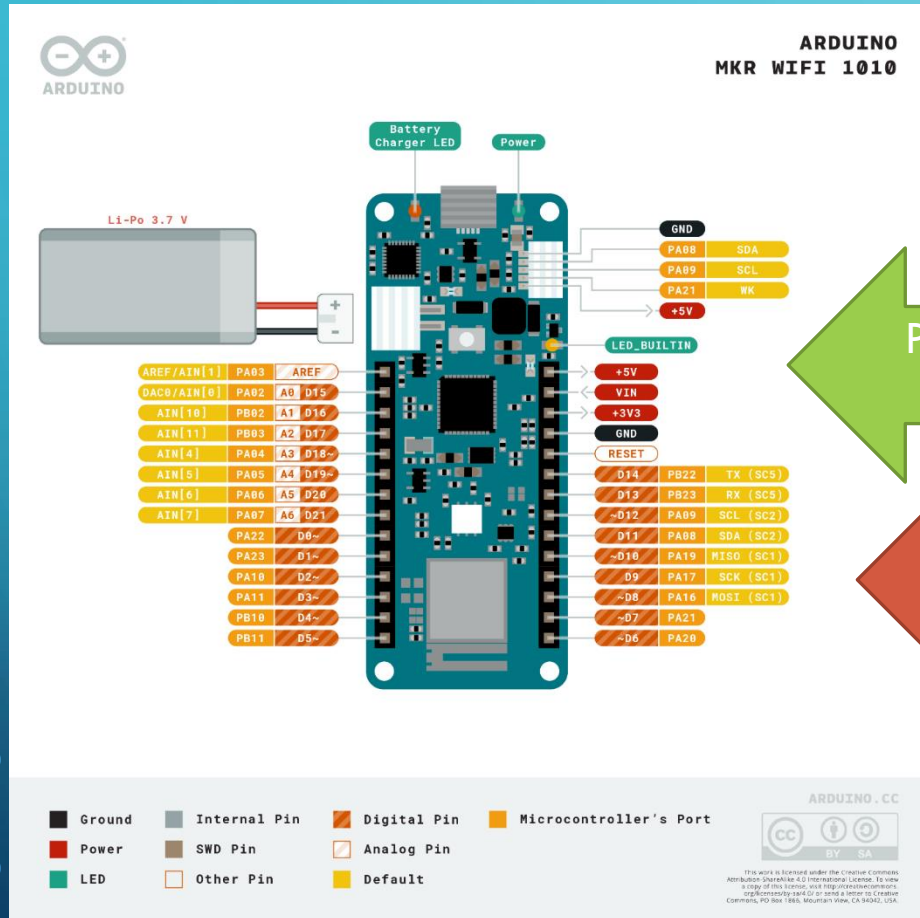
An abstract graphic on the left side of the slide, consisting of a network of white lines and small circles on a blue gradient background. The lines are vertical and horizontal, with some diagonal branches, resembling a circuit board or a neural network. The circles are small and white, some of which are connected to the lines.

VERBAUTE SENSOREN

HARDWAREKOMPONENTEN?

- μ -Controller MKR1010
- Pegelwandler + Spannungsteiler (Am Digi-Pro Board)
- Stromsensor ACS712 (30A)
- Drehzahlsensor-Gabellichtschranke (HC-020K)
- Temperatursensor (DHT22)
- Distanzsensor (HC-SR04)
- Beschleunigungssensor (GY-521 / MPU-6050)

MKR1010



Platine beinhaltet Spannungsregler
von 5v auf 3,3V

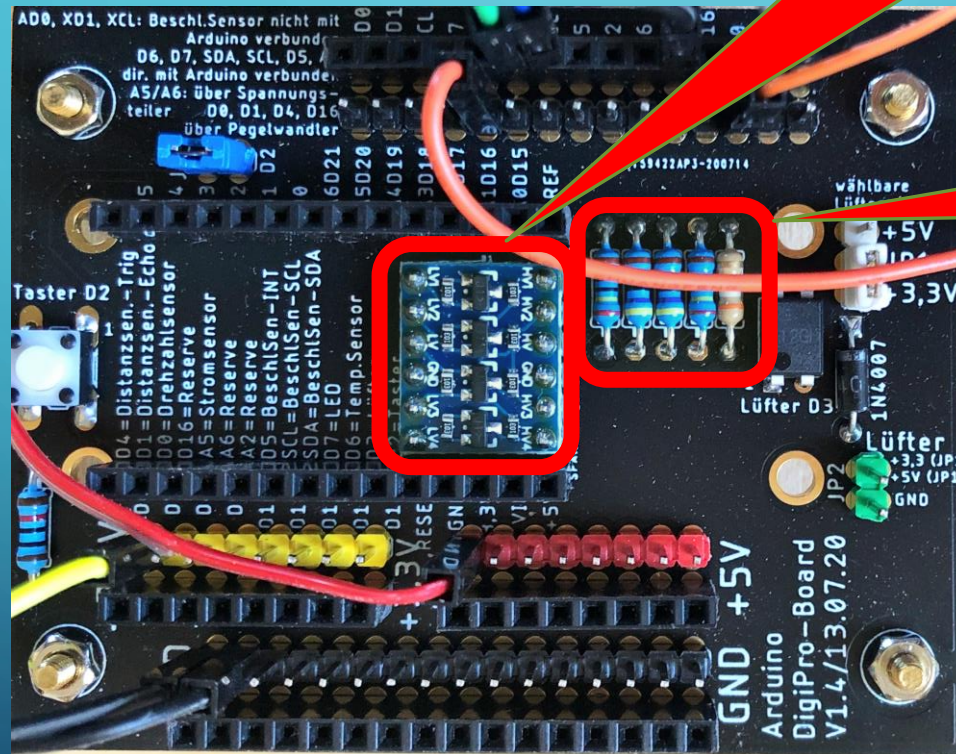
Alle Eingänge an maximal 3,3V
anschießen!

Sensoren sind teilweise aber
nur an 5V zu betreiben !

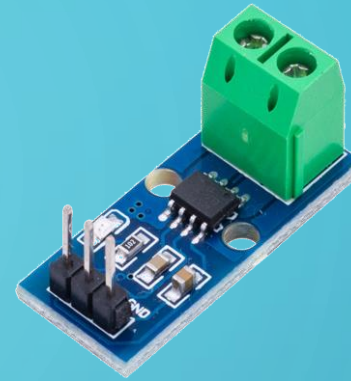
ARDUINO DIGI-PRO-BOARD

Pegelwandler

Spannungsteiler



STROMSENSOR ACS712

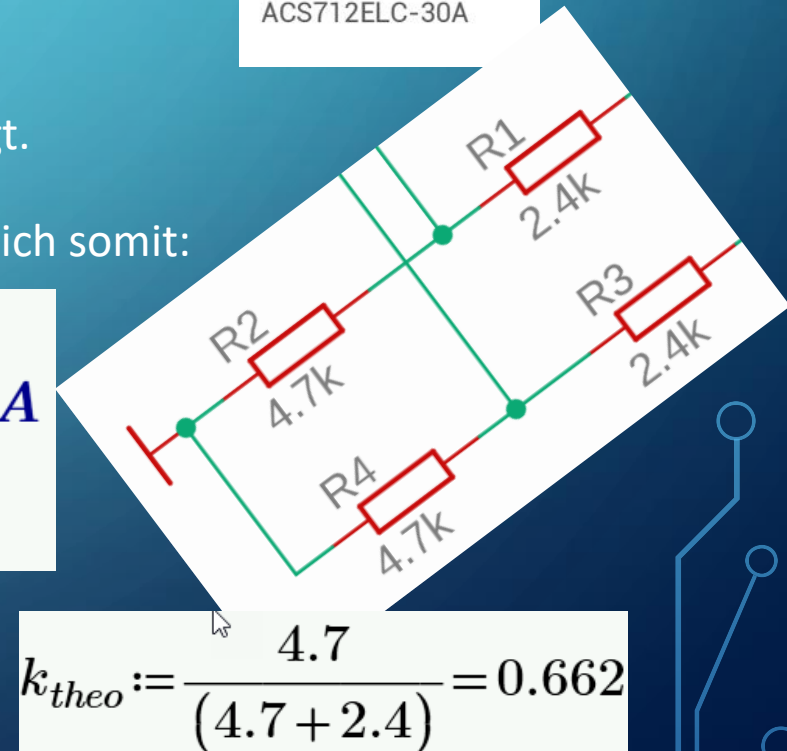


30A Module
5Vdc Nominal
-30 to +30 Amps
VCC/2 (nominally 2.5VDC)
66 mV per Amp
ACS712ELC-30A

- Der MKR1010 hat gegenüber dem „normalen“ Arduino einen 12-bit A-D Wandler und somit bei 3.3V eine Auflösung von $3.3/2^{12} = 3.3/4096 = 0,806 \text{ mV}$
- Der Stromsensor ACS712 liefert ein je nach Stromrichtung einen Wert + bzw. – **66 mV/A** um die halbe Eingangsspannung, die aber 5 Volt beträgt.
- Durch den Spannungsteiler mit dem Verhältnis 0.663 (gemessen) ergibt sich somit:

$$k := \frac{U_{dig}}{k_{sens} \cdot k_{div}} \xrightarrow{\text{explicit, ALL}} \frac{0.000806 \cdot V}{66 \frac{mV}{A} \cdot 0.663} = 0.0184 \text{ A}$$

... nimmt der Wert von AnalogRead um1 zu ,dann ist die Stromstärke um 18,4mA gestiegen



$$k_{theo} := \frac{4.7}{(4.7 + 2.4)} = 0.662$$

PROGRAMMCODE

```
/**** CURRENT SENSOR ****  
float sens_analog_value=0;  
float sens_analog_k = 0.0184;  
float sens_analog_d = 2122;  
// currentsensor - analog input  
analogReadResolution(12);
```

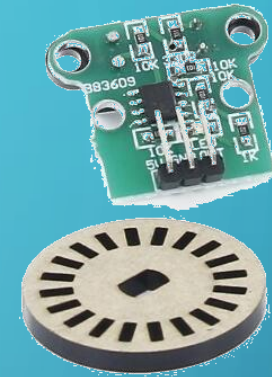
AnalogRead Wert bei 0A

```
int n;  
float v_value = 0;  
for (n=1;n<=10;n++){  
    v_value=v_value+analogRead(PIN_Current);  
}  
v_value=v_value/10;
```

Mittelwertbildung

```
sens_analog_value = roundf(((v_value - sens_analog_d)*sens_analog_k)*100)/100
```

DREHZAHL – LICHTSCHRANKE HC-020K



- Betriebsspannung 5VDC → angeschlossen über Pegelwandler
- Schlitzscheibe hat 20 Löcher
- Zählung der Impulse mittels Interrupt

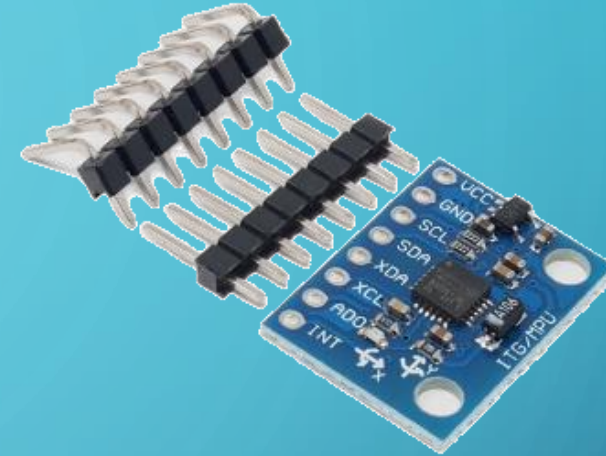
```
// interrupt for rotary encoder  
pinMode(PIN_RPM, INPUT);  
attachInterrupt ( digitalPinToInterrupt (PIN_RPM), sens_rpm_isr, FALLING);
```

```
sens_rpm_value = (unsigned)(long)(60000 * sens_rpm_count /  
    (sens_rpm_t2 - sens_rpm_t1))/RPM_HOLES;  
  
sens_rpm_t1 = sens_rpm_t2;  
sens_rpm_count = 0;
```

Interrupthandler

```
void sens_rpm_isr(){  
    sens_rpm_count++;  
    sens_rpm_t2 = millis();  
}
```


BESCHLEUNIGUNGSSENSOR



- 3-Achsen Gyroskop
- 3-Achsen Beschleunigungsmesser
- Sensordaten werden über den I²C Bus ausgelesen

```
// initialize MPU6050  
Wire.begin();  
Wire.beginTransmission(0x68);  
Wire.write(0x6B);  
Wire.write(0);  
Wire.endTransmission(true);
```

Der I2C-Bus ist als Zweidrahtverbindung zwischen einem Master (Controller) und an ihn angeschlossene Sensoren oder IC-Bausteine (Slaves) für kurze Distanzen konzipiert worden.

Adresse des Sensors

Startet die Kommunikation

ANPASSEN DER AUFLÖSUNG

```
// Configure Accelerometer Sensitivity - Full Scale Range (default +/- 2g)
```

```
Wire.beginTransmission(MPU);
```

```
Wire.write(0x1C); //Talk to the ACCEL_CONFIG register (1C hex)
```

```
Wire.endTransmission(true);
```

```
Wire.requestFrom(MPU, 1);
```

```
byte x = Wire.read(); //the value of Register-28 is in x
```

```
x = x | 0b00011000; //appending values of Bit4 and Bit3 --> AFS_SEL=3 (+/- 16g)
```

```
Wire.beginTransmission(MPU);
```

```
Wire.write(0x1C);
```

```
Wire.write(x); //( +/- 16g );
```

Register für Auflösung des
Sensors

Nur Bit 3 & 4 setzen

The register map for the MPU-60X0 is listed below.

Addr (Hex)	Addr (Dec.)	Register Name	Serial I/F	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0D	13	SELF_TEST_X	R/W	XA_TEST[4-2]			XG_TEST[4-0]				
0E	14	SELF_TEST_Y	R/W	YA_TEST[4-2]			YG_TEST[4-0]				
0F	15	SELF_TEST_Z	R/W	ZA_TEST[4-2]			ZG_TEST[4-0]				
10	16	SELF_TEST_A	R/W	RESERVED		XA_TEST[1-0]		YA_TEST[1-0]		ZA_TEST[1-0]	
19	25	SMPLRT_DIV	R/W	SMPLRT_DIV[7:0]							
1A	26	CONFIG	R/W	-	-	-	EXT_SYNC_SET[2:0]		-	DLPF_CFG[2:0]	
1B	27	GYRO_CONFIG	R/W	-	-	-	FS_SEL [1:0]		-	-	-
1C	28	ACCEL_CONFIG	R/W	XA_ST	YA_ST	ZA_ST	AFS_SEL[1:0]				
				TEMP	YG	YG	YG	YG	YG	YG	YG

WERTE LESEN

```
Wire.beginTransaction(MPU);  
Wire.write(0x3B); // starting with register 0x3B (ACCEL_XOUT_H)  
Wire.endTransmission(false);  
Wire.requestFrom(MPU,14,true); // request a total of 14 registers (6 AC; 2 Temp; 6 GY)
```

```
//read accel data and compute g-value (AFS_SEL ist set to 3 in setup!! +/- 16 g )  
AcX=(float)(int16_t)(Wire.read()<<8 | Wire.read())/2048; //Typcasting to float!  
AcY=(float)(int16_t)(Wire.read()<<8 | Wire.read())/2048;  
AcZ=(float)(int16_t)(Wire.read()<<8 | Wire.read())/2048;
```

erstes Byte lesen

zweites Byte lesen