

*Montage und Programmierung
eines Roboters für
ROBOCUP JUNIOR RESCUE
mit Arduino Nano
Teil 2.1: Blink LED*

Warum benutzen wir nicht `delay`(Zeit in ms)?
Stattdessen benutzen wir `_delay_ms`(Zeit in ms).

```
45 void loop() {
46   /**Testschleife*****
47   while(0) {
48     Serial.println(123);
49   }
50   /**Ende Testschleife*****
51
52   while (1) {
53     //At Pin 5 PORTB the LED of the Arduino board is mounted (Pin 13)
54     digitalWrite(LED, HIGH);
55     _delay_ms(200);
56     digitalWrite(LED, LOW);
57     _delay_ms(200);
58     /*Alternative Programmierung:*****
59     // PORTB |= (1 << PORTB5); //Push PORTB pin 5 high (LED, pin 13 "Uno")
60     // _delay_ms(200);
61     // PORTB &= ~(1 << PORTB5); //~ = "NOT" Operator, 1<<5 shifts the "1" five times to the left
62     // _delay_ms(200);
63     //A not A
64     //0 1
65     //1 0
66     //PINB |= (1<<PINB5); //A "1" to "PIN" toggles the Pin of the PORT
67   }
68
69 }
```

Die, von der Arduino IDE, bereit gestellte Funktion `delay()`, ist abhängig von einem **internen Timer**. Später werden wir ihn anders einstellen, so dass bei `delay(1000)` **NICHT** eine Pause von einer Sekunde entsteht, sondern nur von ca. 16,8 ms. (ms = Millisekunden)
Daher sollten wir jetzt schon `delay_ms()` benutzen.

Das Programm kennen wir schon aus Hello_World (01_Start).

Wir fügen zwei Pausen, je 200ms, ein und schalten die LED wieder aus. Das führt zu einer „Schwingungsdauer“ von 400 Millisekunden.

Alternativ zur Arduino IDE können wir statt **`digitalWrite(LED, HIGH);`**, auch direkte Port Anweisungen machen:

`PORTB |= (1 << PORTB5);`

`PORTB |= ...` ist die Kurzform von `PORTB = PORTB | (1 << PORTB5);`

wobei zu erkennen ist, dass „`=`“ eine Zuweisung und KEINE Gleichsetzung ist.

`PORTB5` hat den Wert 5,

`<<` ist ein Schiebe-Operator,

mit `(1 << PORTB5)` wird die ZAHL 1 um 5 Positionen

nach links geschoben, so dass der binäre

Wert `0b00100000` entsteht.

Oder wir nun `PORTB` mit diesem Wert,

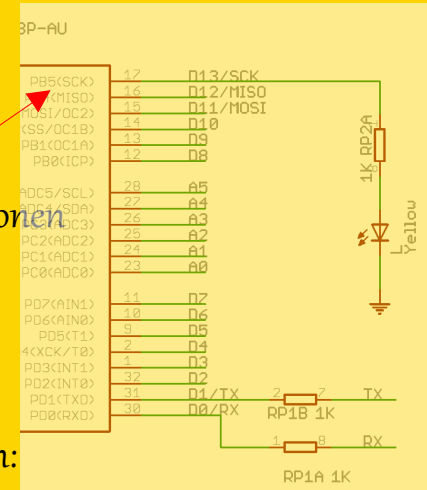
das 5. Bit von `PORTB` auf eins (=5V) gesetzt.

An `PORTB` Pin 5 ist aber die LED angeschlossen:

Die LED geht an.

Wir manipulieren also direkt den entsprechenden Pin am

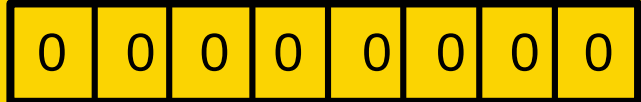
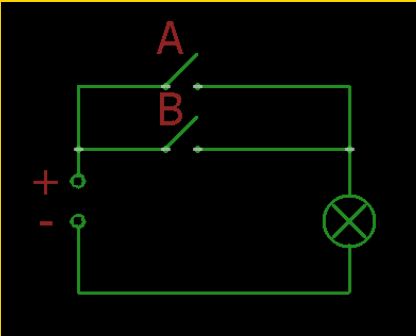
Mikrocontroller. Das gleiche macht die `digitalWrite` Anweisung.



Wahrheitstabelle:

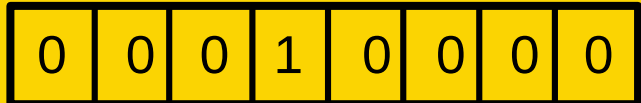
Oder (Or |)

| A | B | A B |
|---|---|-------|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |



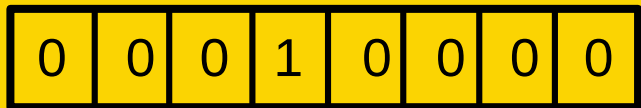
= 0

oder



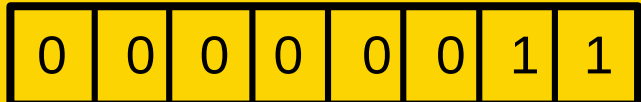
= 16

ergibt



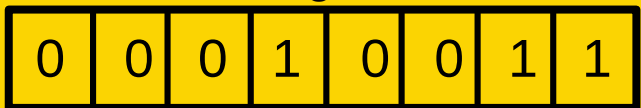
= 16

oder



= 3

ergibt

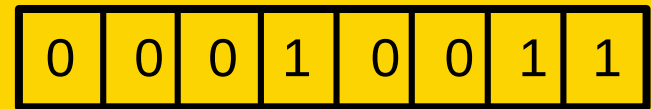
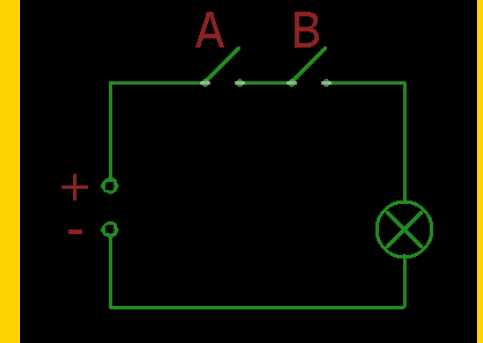


= 19

Wahrheitstabelle:

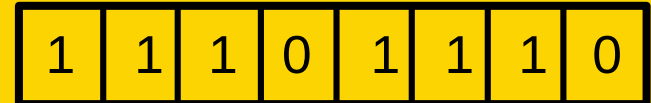
Und (And &)

| A | B | A & B |
|---|---|-------|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |



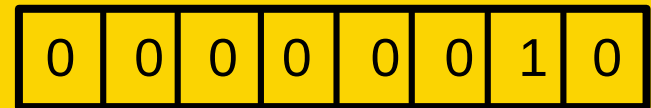
= 19

und



= 238

gleich



= 2

~ ist der NOT-Operator, der macht aus einer 0 eine 1 und umgekehrt.

Glossar

`_delay_ms(ms)`

`digitalWrite(Pin, Mode);`

`PORTX = PORTX | (1 << PORTXy);`

`0bxxxxxxxx;`

OR, Oder, |

And, Und, &

Pause in Millisekunden (ms), **NICHT** `delay()`

Einen Pin auf GND (0 Volt) oder +5V setzen, Parameter HIGH oder LOW

Einen Port-Pin auf GND oder +5V setzen, X = {B, C, D}, y = {0, ..., 7}, Parameter = Wert

Binäre Zahl, 8 bit = 1 Byte, Dezimale Zahl `0dxxxxx`

Mit OR, Oder, | kann an einer Stelle im Byte ein bestimmtes Bit **gesetzt** werden.

Mit AND, UND, & kann an einer Stelle im Byte ein bestimmtes Bit **gelöscht** werden.

*Montage und Programmierung
eines Roboters für
ROBOCUP JUNIOR RESCUE
mit Arduino Nano
Teil 2.2: Serieller Monitor*