



### Der Arduino und unabhängige Timer

Viele Arduino-Funktionen verwenden Timer, z. B. `delay()`, `millis()`. Ein Timer oder Zähler ist eine unabhängige Schaltung auf dem Arduino. Sie können einfach Ereignisse zählen oder eine vorgegebene Zeiteinheit abstoppen. Man kann die Timer dazu nutzen, nach Ablauf der Zeit den Ablauf der `loop()` zu unterbrechen und vorgesehene Anweisungen auszuführen, bevor die Ausführung der `loop()` wieder aufgenommen werden. Der große Vorteil liegt darin, dass man z.B. Messungen in regelmäßigen Intervallen durchführen kann. Denn wenn in der Hauptschleife umfangreiche Berechnungen durchgeführt werden, kann es zu ungenauen Abständen von Messungen kommen, wenn man die Abstandsabfragen einfach in den Code einbaut.

Der Hintergrund dieser Timer soll zu einem späteren Zeitpunkt Gegenstand der Theorie sein, hier nutzen wir einfach eine Bibliothek, durch die Details verborgen bleiben.

Mit der Funktion

**`Timer1.initialize(<Zeit in µsec>)`**

initialisieren wir den Timer. Der Timer ist wie eine Eieruhr, die sich nach der angegebenen Zahl Mikro-Sekunden sofort wieder gestartet wird. Anstatt aber zu klingeln, können wir eine Funktion bestimmen, die aufgerufen werden soll, wenn die Zeit abgelaufen ist.

Diese Funktion wird als Interrupt – also als Unterbrechung der normalen Programm-Ausführung – aufgerufen. Nachdem diese Interrupt-Funktion abgearbeitet ist, kehrt die Programmausführung wieder dahin zurück, wo sie vor dem Interrupt war und macht dort weiter. Nehmen wir an, die Funktion soll `timer_isr()` (isr steht für „Interrupt Service Routine“) heißen, so binden wir diese Funktion an den Timer mit

**`Timer1.attachInterrupt(timer_isr); // ohne Klammern`**

In der Funktion `timer_isr()` können wir nun die neuen Messwerte bestimmen und dann in der `loop()` benutzen.

```
void loop() {
  forward();
  // Calculating the distance
  if (distance <= 20) {
    speed = speed / 2;
    if (distance < 5) {
      speed = 0;
    }
  }
  setSpeed(speed);
}

// Diese Funktion wird aufgerufen nach 0,1 s
void timer_isr(void) {
  distance = getDistanceFromSonic();
}
```