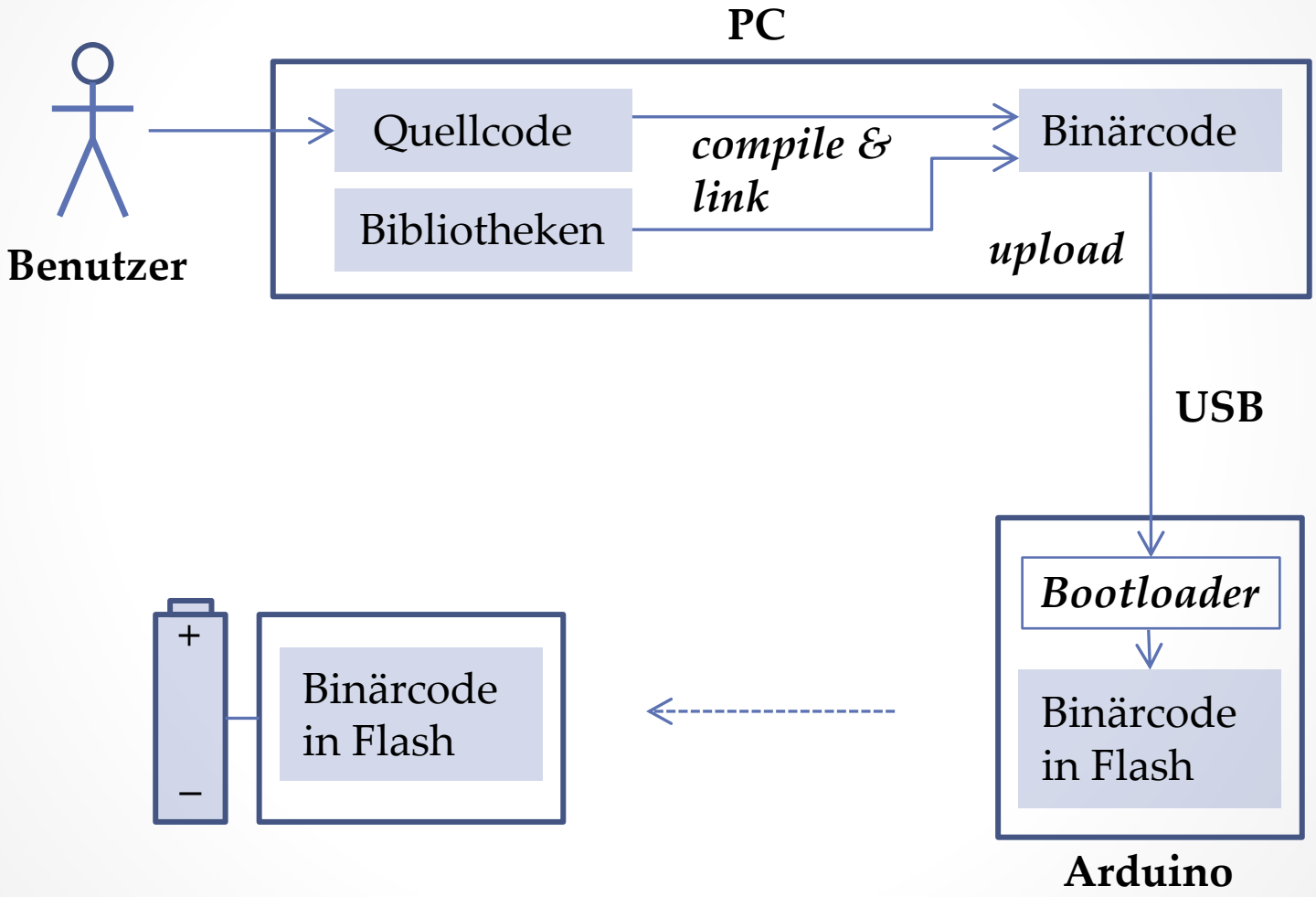
Arduino

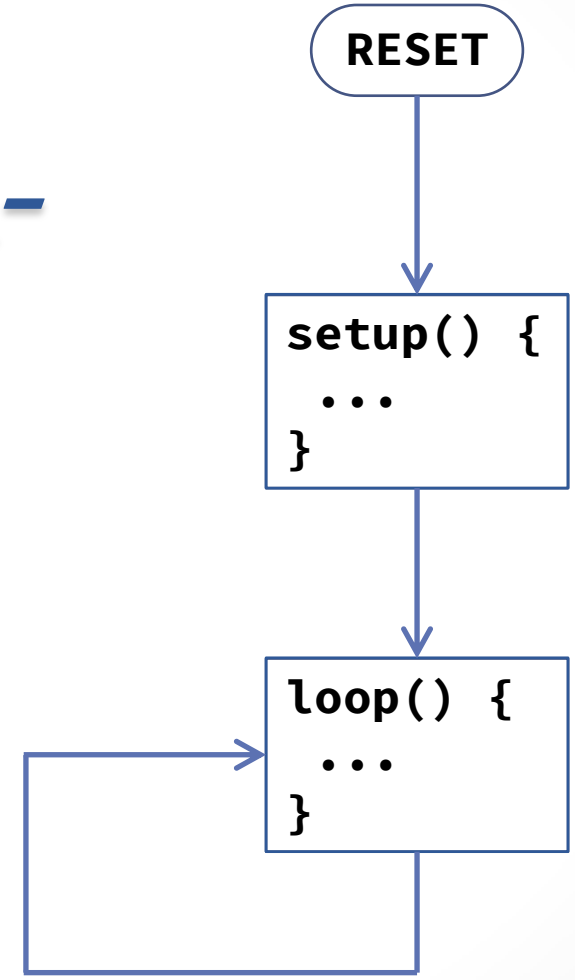
Systemsicht und Programmierung

System-Übersicht

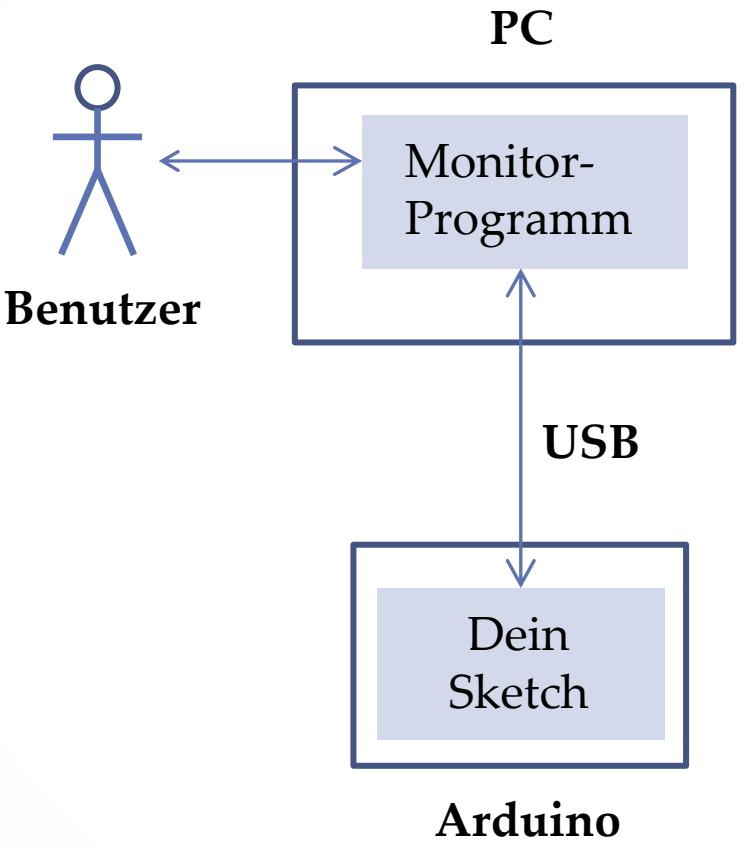


Programm-ablauf

(vereinfacht)

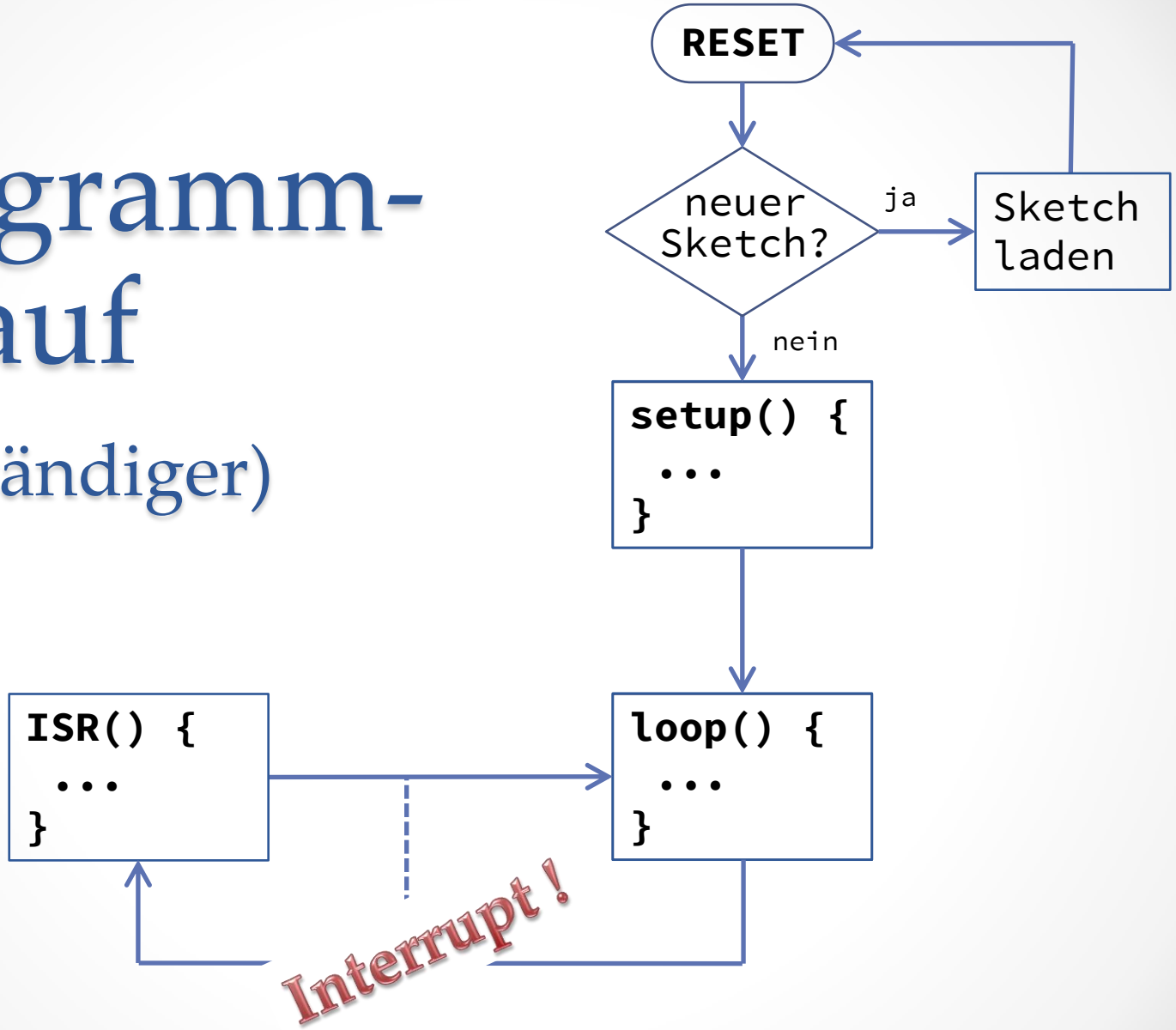


Monitor-Betrieb

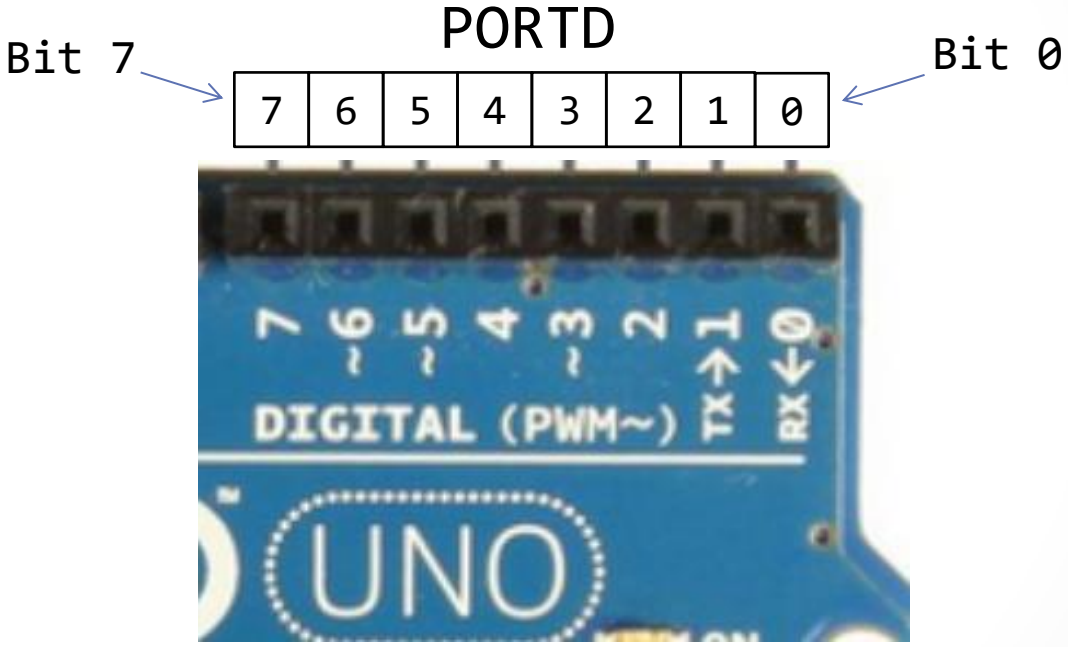


Programm-ablauf

(vollständiger)



Ports




```
byte x;
x = 7;
x = 0b00000111;
PORTD(X);  Pin 0, 1, 2 = HIGH
```

Bibliotheken

Bibliotheken z.B. Arduino API:

- Sind Sammlungen von Funktionen (*pulse()*, *tone()*)
- werden im Quellcode aufgerufen
- enthalten Maschinenbefehle
- werden mit dem Benutzerprogramm verbunden
- entlasten den Arduino-Programmierer vor lästiger Kleinarbeit

Bibliotheken existieren auch für bestimmte Peripherie: Servomotor, serielle Schnittstelle, Schieberegister

Timer (1)

Timer (=Zeitgeber) dienen folgenden Zwecken:

- **Warten**: Für eine gegebene Zeit die Ausführung des Programms anhalten → `delay()`
- **Abfrage**: Wieviel Zeit ist vergangen? → `millis()`
- **Zeitgerechte Ausführung**: Interrupt, siehe Timer1 →
  - `Timer1.initialize(<Zeit in µs>);`
  - `Timer1.attachInterrupt(<meine Funktion>);`

Timer (2)

Der Micro-Controller des Arduino UNO d.h. der ATmega328 verfügt über 3 Timer, die wie folgt verwendet werden:

- **Timer0** - 8 bit timer, benutzt von den Arduino-Funktionen `delay()`, `millis()` und `micros()`.
- **Timer1** - 16 bit timer, benutzt von der Servo-Bibliothek (library)
- **Timer2** - 8 bit timer, benutzt von der Tone-Bibliothek

➔ Darum ergeben sich für die Nutzung als Interruptquellen möglicherweise Konflikte!

Hardware-Interrupts

Programme des ATmega328 können über

- Pin 2 (= Interrupt 0) und
- Pin 3 (= Interrupt 1) unterbrochen werden.

Programmierung:

- `pinMode(<Pin #>, INPUT_PULLUP);`
- `attachInterrupt(<Interrupt #>, <meine Funktion>, FALLING);`

Nützlich: `digitalPinToInterrupt(<Pin #>)` bestimmt den Interrupt aus der Angabe des Pins.

Statt `FALLING` kann auch `RISING` oder `CHANGE` angegeben werden.

Arduino Shields

- Anschluss von Shields und die zugehörigen Bibs