
Fachprojekt for Embedded System: Design and Implement Your Own Embedded Systems (2)

Junjie Shi and Niklas Ueter

LS 12, TU Dortmund

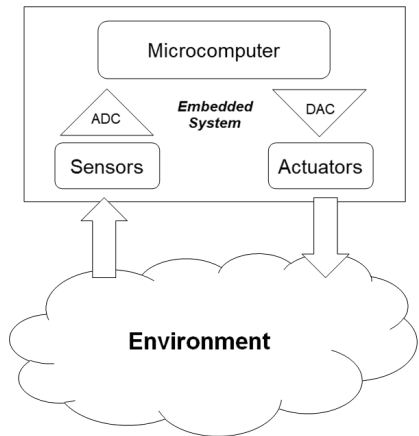
10, April, 2019

- 1 Was ist ein Eingebettetes System?
- 2 Warum Arduino?
- 3 Wie entwickeln?
- 4 Einfache Beispiele
- 5 Referenzen

Was ist ein Eingebettetes System?

- "Dortmund" Definition: [Peter Marwedel]
Embedded systems are information processing systems embedded into a larger product.
- Berkeley: [Edward A. Lee]
Embedded software is software integrated with physical processes. The technical problem is managing time and concurrency in computational systems.
- Cyber-Physical Systems (CPS) are integrations of computation with physical processes [Edward Lee, 2006].
- Cyber-physical system (CPS) = Embedded System (ES) + physical environment

In diesem Fachprojekt:



- Informationen aus der Umgebung beziehen und darauf reagieren.
- Regelsysteme: Closed loop feed back control system.
- Arduino als Microcomputer.
- *'Hardware'* und Software müssen entworfen werden.

Warum Arduino?

- Eine leichte Open-Source Software, Hardware & Elektronik Plattform für die man leicht entwickeln kann.
- Preiswert
- Cross-platform (Windows, Mac OS, Linux)
- Einfache Programmierumgebung
- Viele Bibliotheken und Tutorials verfügbar
- Open-Source Software- und Hardware-Erweiterungen

<ref: <https://www.arduino.cc/en/Guide/Introduction>>

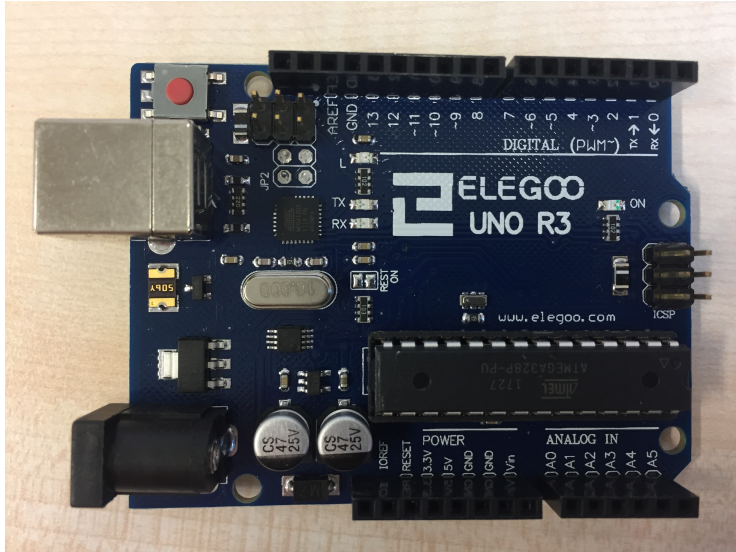
Arduino Mega 2560



Arduino Mega 2560

- Maximal 500 mA Versorgungsstrom (USB Port).
- Stromversorgung via USB oder externem Netzteil.
- 54 Digital I/O Pins (davon 15 für **PulseWidthModulation**).
- PWM ist eine einfache Möglichkeit der D/A in Microcontroller-Systemen.
- 16 Analog Input Pins
- 16 MHz Clock Speed
- Weitere Details:
<https://store.arduino.cc/arduino-mega-2560-rev3>

Arduino UNO



Arduino UNO

- Gleiches Netzteil wie der Arduino Mega 2560.
- Weniger Pins and Flash Speicher,
14 Digital I/O pins (davon 6 für PWM),
6 Analog Input Pins.
- Real-Time Anwendungen sind möglich.
- Weitere Details:
<https://store.arduino.cc/arduino-uno-rev3>

Wie entwickeln?

- Arduino IDE downloaden:
<https://www.arduino.cc/en/Guide/HomePage>
- Installation des IDE auf ihrem Computer oder Verwendung des Web Editors (**Nicht empfohlen!**)
- C-ähnliche Sprache:
<https://www.arduino.cc/reference/en/>

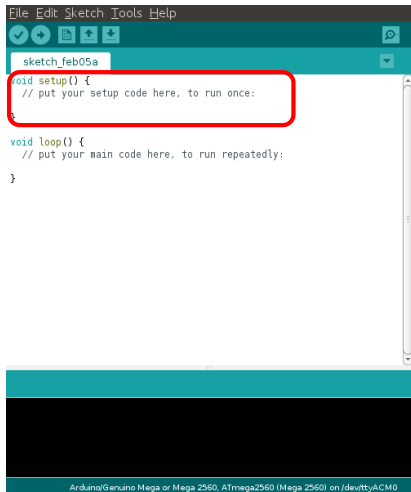
Wie entwickeln?

```
File Edit Sketch Tools Help
sketch_feb05a
void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}

Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on /dev/ttyACM0
```

Wie entwickeln?



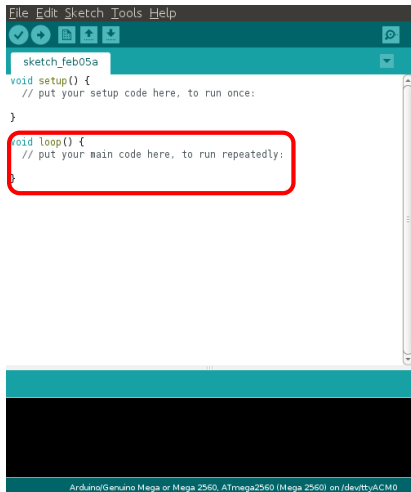
```
File Edit Sketch Tools Help
sketch_feb05a
void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}

Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on /dev/ttyACM0
```

- *setup()*
Wird nur einmal ausgeführt

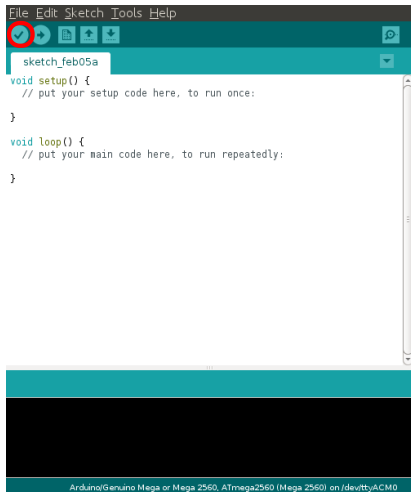
Wie entwickeln?



```
File Edit Sketch Tools Help
sketch_feb05a
void setup() {
  // put your setup code here, to run once:
}
void loop() {
  // put your main code here, to run repeatedly:
}
```

- *setup()*
Wird nur einmal ausgeführt
- *loop()*
Mehrfache Ausführung

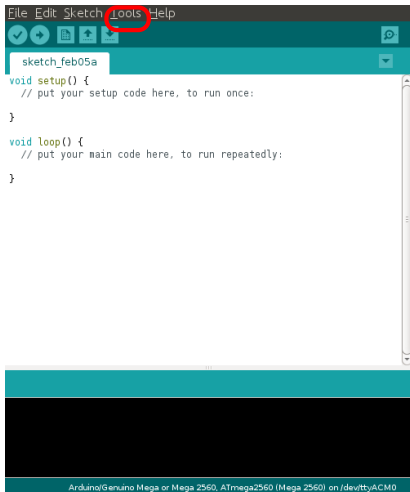
Wie entwickeln?



```
File Edit Sketch Tools Help
sketch_feb05a
void setup() {
  // put your setup code here, to run once:
}
void loop() {
  // put your main code here, to run repeatedly:
}
Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on /dev/ttyACM0
```

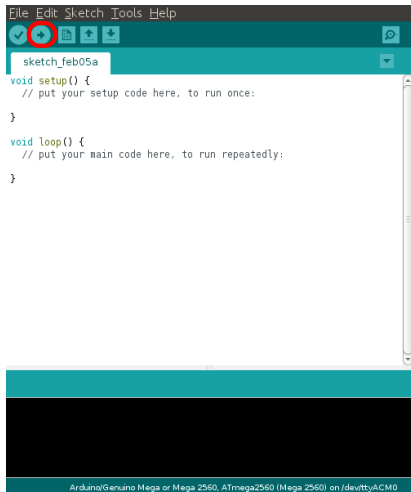
- *setup()*
Wird nur einmal ausgeführt
- *loop()*
Mehrfache Ausführung
- ✓
Sourcecode verifizieren

Wie entwickeln?



- *setup()*
Wird nur einmal ausgeführt
- *loop()*
Mehrfache Ausführung
- ✓
Sourcecode verifizieren
- *Tools*
Auswahl des
Entwicklungs-Boards und
des Ports

Wie entwickeln?



- *setup()*
Wird nur einmal ausgeführt
- *loop()*
Mehrfache Ausführung
- ✓
Sourcecode verifizieren
- *Tools*
Auswahl des
Entwicklungs-Boards und
des Ports
- →
Kompilieren und flashen

Port Einstellungen

- In Windows:
Das IDE erkennt das Arduino Board automatisch.
- In Linux:
Fehlermeldung:
"Error opening serial port..."

Port Einstellungen in Linux

① Verbinden des Arduino Boards mit ihrem Computer

② Öffnen des Terminals:

```
ls -l /dev/ttyACM*
```

Ausgabe:

```
crw-rw-- 1 root dialout 166, 0 Feb 6 10:53  
/dev/ttyACM0
```

Die "0" am Ende des ACM kann sich unterscheiden oder sogar mehrfache Einträge können zurück gegeben werden.

Group Owner: *dialout*

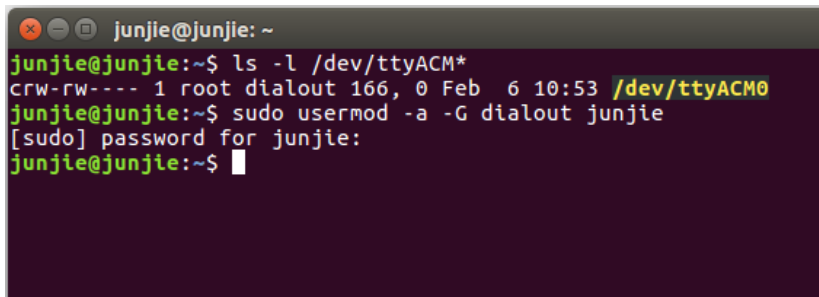
③ Hinzufügen des Benutzers zu der Gruppe *dialout*:

```
sudo usermod -a -G dialout <linux username>
```

<ref: <https://www.arduino.cc/en/Guide/Linux>>

Wie entwickeln?

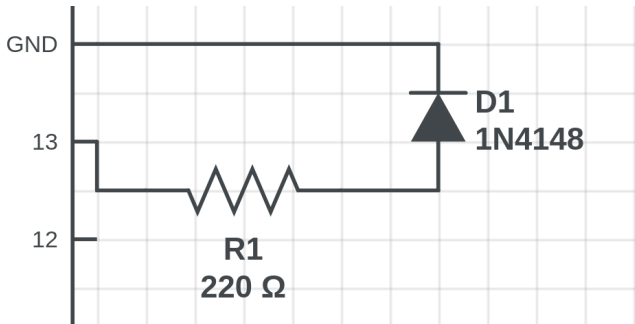
- Sie müssen sich einmal Aus- und Einloggen, damit die Änderungen übernommen werden.



```
junjie@junjie: ~  
junjie@junjie:~$ ls -l /dev/ttyACM*  
crw-rw---- 1 root dialout 166, 0 Feb  6 10:53 /dev/ttyACM0  
junjie@junjie:~$ sudo usermod -a -G dialout junjie  
[sudo] password for junjie:  
junjie@junjie:~$
```

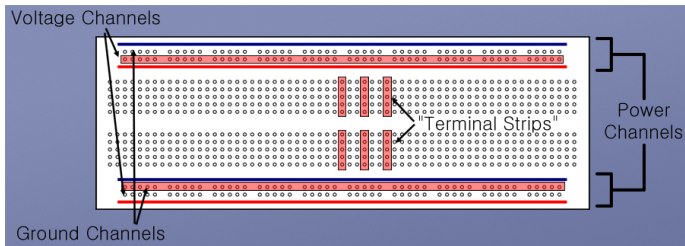
Hello World

- LEDs blinken lassen entspricht dem *Hello World* in Arduino
- Der Source code kann hier gefunden werden:
File → *Examples* → *01.Basics* → *Blink*
- Ein exemplarischer Ausschnitt des Schaltkreises: **Immer einen Widerstand verwenden!**



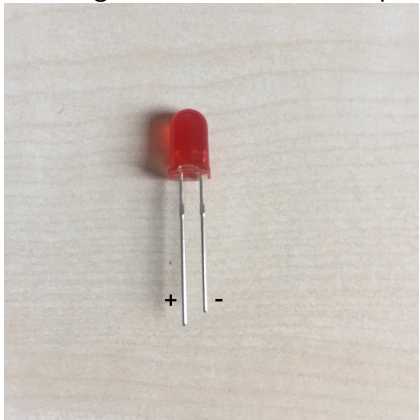
Schaltkreis *entwerfen*

- Zum schnellen & lötfreiem Testen \Rightarrow Breadboard



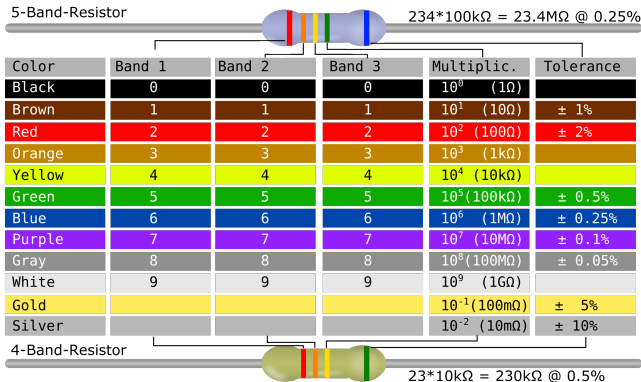
Schaltkreis *entwerfen*

- Zum schnellen & lötfreiem Testen \Rightarrow Breadboard
- Der längere Pin der Diode entspricht dem positiven Pol



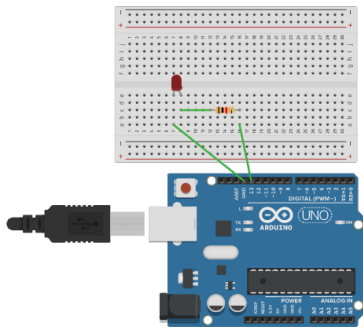
Schaltkreis *entwerfen*

- Zum schnellen & lötfreiem Testen \Rightarrow Breadboard
- Der längere Pin der Diode entspricht dem positiven Pol
- Kodierung der Werte der Widerstände anhand der Farben



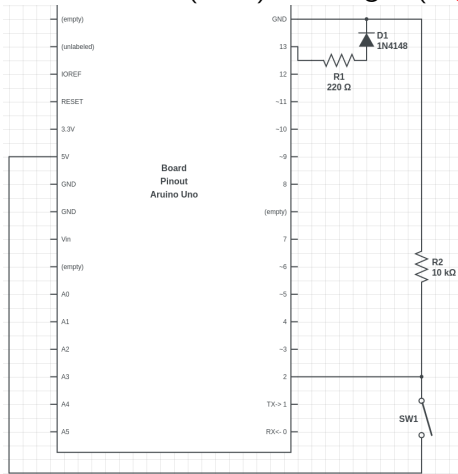
Schaltkreis *entwerfen*

- Zum schnellen & lötfreiem Testen \Rightarrow Breadboard
- Der längere Pin der Diode entspricht dem positiven Pol
- Kodierung der Werte der Widerstände anhand der Farben
- Verbinden aller Komponenten wie folgt



Ein Weiterführendes Beispiel

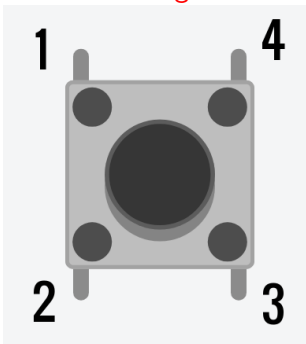
- Einen Schalter (SW1) hinzufügen (**Digital Input**)



Ein Weiterführendes Beispiel

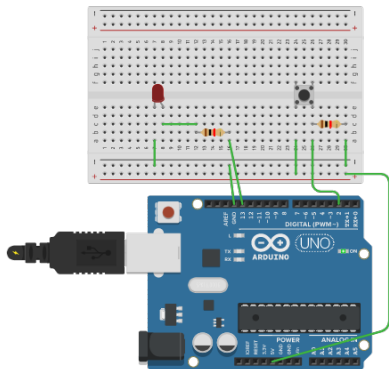
- Einen Schalter (SW1) hinzufügen (**Digital Input**)
- Beschaltung des Schalters:
Pin 1 und Pin 2 sind verbunden, Pin 3 und Pin 4 sind verbunden

Unterscheidung der Richtung und Position der Pins



Ein Weiterführendes Beispiel

- Einen Schalter (SW1) hinzufügen (**Digital Input**)
- Beschaltung des Schalters:
- Der komplette Schaltkreis ist wie folgt:



Ein Weiterführendes Beispiel - Source Code

```
const int ledPin = 13; // the number of the LED pin
const int buttonPin = 2; // the number of the
    button pin

int ledState = LOW; // ledState used to set the LED
int buttonState = 0;

void setup() {
    // set the mode of digital pin:
    pinMode(ledPin, OUTPUT);
    pinMode(buttonPin, INPUT);
}
```

Ein Weiterführendes Beispiel - Source Code

```
void loop() {  
  //check the state of the buttonPin  
  buttonState = digitalRead(buttonPin);  
  //give the output according to the input  
  if (buttonState == HIGH) {  
    digitalWrite(ledPin, HIGH);  
  } else {  
    digitalWrite(ledPin, LOW);  
  }  
}
```

Exercises

- Drei LEDs, LED_1 blinkt alle 3 Sekunden, LED_2 blinkt alle 5 Sekunden und eine weitere LED blinkt alle 7 Sekunden. Die 'Blinkdauer' beträgt eine Sekunde.
- Es werden drei Schalter / Knöpfe verschwendet um die RGB LEDs zu kontrollieren. Andere Knöpfe können andere Farben kontrollieren.

Hilfreiche Webseiten

- Project hub:
<https://create.arduino.cc/projecthub?f=1>
- Simulator:
Circuit in Tinkercad (basierend auf Arduino-UNO):
<https://www.tinkercad.com/circuits>
- Schaltkreise zeichnen:
z.B. <https://www.circuitlab.com/editor/#?id=6t78uk>

Fragen

Weitere Fragen?

