

# Microcontroller Teil 1 Hardware

DI (FH) Andreas Pötscher

HTL Litec

Ein Pin ist Anschluss eines ICs, der mit der Platine verlötet wird.

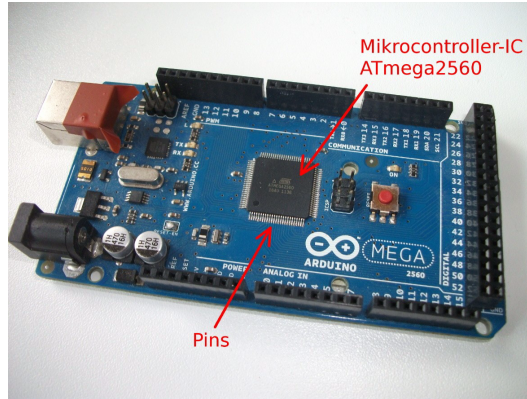


Figure 1: Pin eines Microcontrollers

*Die meisten Pins der AVR-Mikrocontroller sind so genannte GPIO-Pins: General Purpose Input Output Pins – Allgemein verwendbare Ein-/Ausgangs-Pins. Sie können entweder als digitale Eingänge oder als digitale Ausgänge verwendet werden.*

- ▶ Eine externe Schaltung legt eine Spannung von 0V oder 5V an den Pin.

- ▶ Eine externe Schaltung legt eine Spannung von 0V oder 5V an den Pin.
- ▶ Die Spannung wird vom Microcontroller als logisch 0 oder logisch 1 eingelesen.

- ▶ Eine externe Schaltung legt eine Spannung von 0V oder 5V an den Pin.
- ▶ Die Spannung wird vom Microcontroller als logisch 0 oder logisch 1 eingelesen.
- ▶ Verwendung für Taster, Sensoren, ...

- ▶ Der Microcontroller legt programmgesteuert eine Spannung von 0V oder 5V an den Pin.

- ▶ Der Microcontroller legt programmgesteuert eine Spannung von 0V oder 5V an den Pin.
- ▶ Die Spannungspegel wird vom Microcontroller als logisch 0 oder logisch 1 in der Software gesetzt.



- ▶ Der Microcontroller legt programmgesteuert eine Spannung von 0V oder 5V an den Pin.
- ▶ Die Spannungspegel wird vom Microcontroller als logisch 0 oder logisch 1 in der Software gesetzt.
- ▶ Verwendung für LEDs, Relais, Motor, ...

*Üblicherweise sind die meisten Pins eines Mikrocontrollers GPIO-Pins (General Purpose Input/Output). Sie können entweder als Eingänge oder als Ausgänge verwendet werden. Das im Mikrocontroller laufende Programm muss jeden verwendeten GPIO-Pin zu einem Ein- oder zu einem Ausgang programmieren. Das passiert in der Initialisierungsphase des Programms.*

- ▶ Microcontrollerpins sind üblicherweise zu Achtergruppen zusammengefasst.

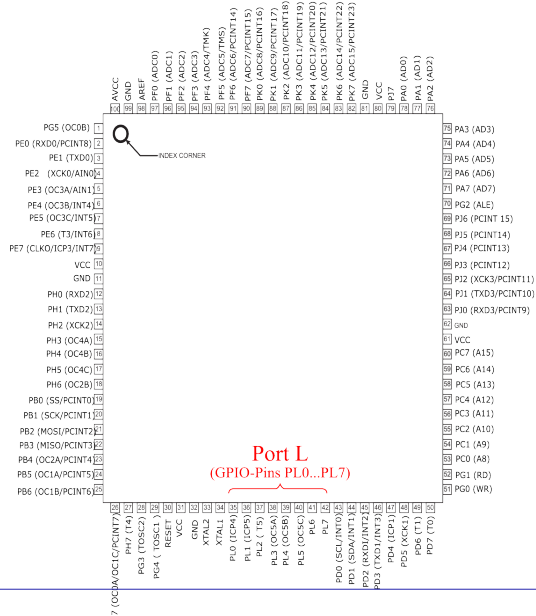
- ▶ Microcontrollerpins sind üblicherweise zu Achtergruppen zusammengefasst.
- ▶ Diese Achtergruppen bilden einen **Port**.

- ▶ Microcontrollerpins sind üblicherweise zu Achtergruppen zusammengefasst.
- ▶ Diese Achtergruppen bilden einen **Port**.
- ▶ Ports werden mit einem Buchstaben bezeichnet: PortA, PortB, ....

- ▶ Microcontrollerpins sind üblicherweise zu Achtergruppen zusammengefasst.
- ▶ Diese Achtergruppen bilden einen **Port**.
- ▶ Ports werden mit einem Buchstaben bezeichnet: PortA, PortB, ....
- ▶ Die Pins werden mit 0 bis 7 durchnummeriert.

- ▶ Microcontrollerpins sind üblicherweise zu Achtergruppen zusammengefasst.
- ▶ Diese Achtergruppen bilden einen **Port**.
- ▶ Ports werden mit einem Buchstaben bezeichnet: PortA, PortB, ....
- ▶ Die Pins werden mit 0 bis 7 durchnummeriert.
- ▶ Der Pin 0 von PortB hat dann die Bezeichnung PB0

# Beispiel Pinout





- ▶ GPIO Pins können auch alternative Funktion haben.

- ▶ GPIO Pins können auch alternative Funktion haben.
- ▶ Der Pin kann dann entweder als Eingang, Ausgang oder in seiner alternativen Funktion verwendet werden.

- ▶ GPIO Pins können auch alternative Funktion haben.
- ▶ Der Pin kann dann entweder als Eingang, Ausgang oder in seiner alternativen Funktion verwendet werden.
- ▶ Z.B hat der der PIN PE0 die alternative Funktion *RXD0* (Receive Data 0).

- ▶ GPIO Pins können auch alternative Funktion haben.
- ▶ Der Pin kann dann entweder als Eingang, Ausgang oder in seiner alternativen Funktion verwendet werden.
- ▶ Z.B hat der der PIN PE0 die alternative Funktion *RXD0* (Receive Data 0).
- ▶ Beim Arduino Mega werden diese als serielle Schnittstelle verwendet und können damit nicht als GPIO genutzt werden.

# Pin Mapping

## ARDUINO MEGA PINOUT DIAGRAM

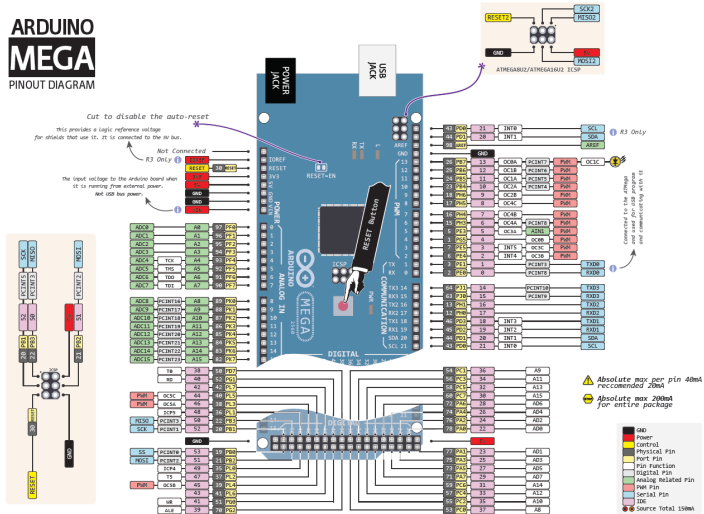


Figure 3: Arduino Mega Pinout Diagramm

*Wird vom Programm aus ein GPIO-Pin als Ausgang programmiert, so gibt der Mikrocontroller entweder einen High- oder einen Low-Pegel aus (5V gegen GND bzw. 0V gegen GND). Vom Programm aus kann der gewünschte Pegel vorgegeben werden, und zwar durch Schreiben einer 1 bzw. einer 0 in ein Bit in einem Special-Function-Register (oder durch Verwenden einer Bibliotheks-Funktion).*

# GPIO als Ausgang (High Pegel)

Bei einem High Pegel wird der GPIO über einen Schalttransistor mit **5V** verbunden.

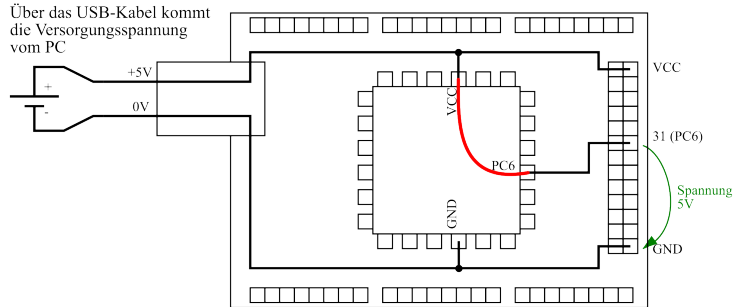


Figure 4: High Pegel

Bei einem Low Pegel wird der GPIO über einen Schalttransistor mit **GND** verbunden.

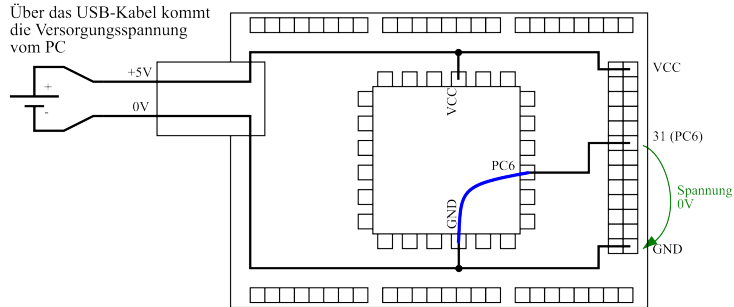


Figure 5: Low Pegel



- ▶ LEDs müssen immer mit einem Vorwiderstand betrieben werden (ca.  $330\ \Omega$ , um den LED-Strom auf ca. 10mA zu begrenzen).

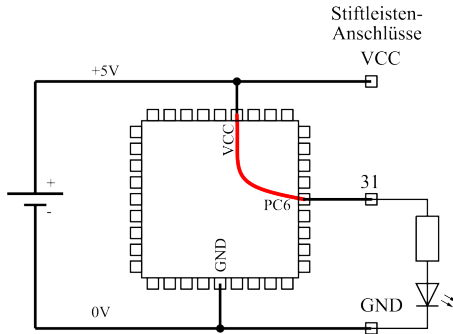
- ▶ LEDs müssen immer mit einem Vorwiderstand betrieben werden (ca.  $330\ \Omega$ , um den LED-Strom auf ca. 10mA zu begrenzen).
- ▶ Es gibt 2 Varianten eine LED anzuschließen.

- ▶ LEDs müssen immer mit einem Vorwiderstand betrieben werden (ca.  $330\ \Omega$ , um den LED-Strom auf ca. 10mA zu begrenzen).
- ▶ Es gibt 2 Varianten eine LED anzuschließen.
- ▶ In beiden Fällen muss der GPIO als Ausgang programmiert werden.

# LED an einem GPIO Pin Variante 1

Bei dieser Variante wird mit einer logischen **1** im Programm die LED **eingeschaltet** und mit einer logischen **0** **ausgeschaltet**.

LED eingeschaltet



LED ausgeschaltet

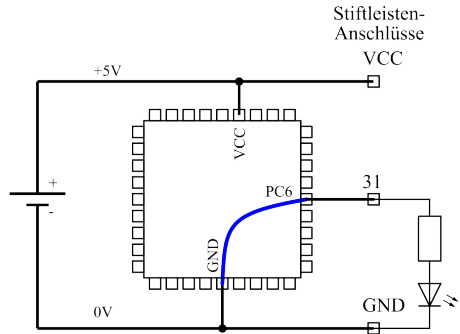
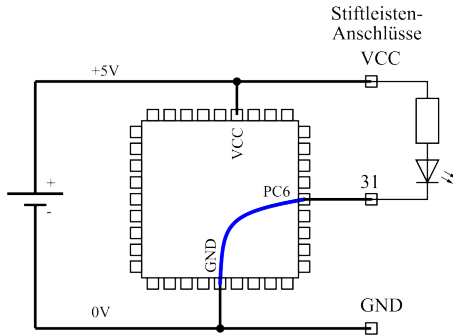


Figure 6: Erste Variante zum Anschließen von Leds

## LED an einem GPIO Pin Variante 2

Bei dieser Variante wird mit einer logischen **0** im Programm die LED **eingeschaltet** und mit einer logischen **1** **ausgeschaltet**.

LED eingeschaltet



LED ausgeschaltet

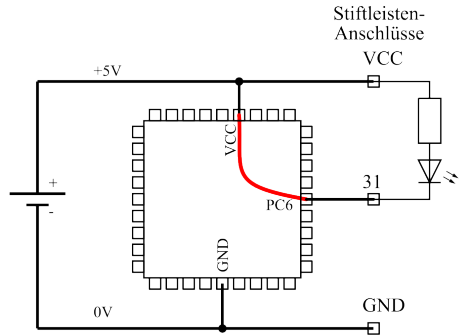


Figure 7: Zweite Variante zum Anschließen von Leds

**Beispiel** Schließen Sie die LEDs in beiden Varianten an den Microcontroller und starten Sie die Simulation.

*Wird ein GPIO-Pin als Eingang programmiert, muss eine externe Schaltung eine Spannung von 5V (High-Pegel) oder 0V (Low-Pegel) zwischen Pin und GND erzeugen. Meist geschieht das, indem die externe Schaltung den Eingangspin des Mikrocontrollers mit VCC oder mit GND verbindet. Der von der externen Schaltung angelegte Spannungspegel kann vom Programm eingelesen werden.*

- ▶ Ein verwendeter Eingang darf nicht potenzialfrei bleiben.



- ▶ Ein verwendeter Eingang darf nicht potenzialfrei bleiben.
- ▶ Das bedeutet es muss immer eine Verbindung entweder mit 5V oder GND hergestellt sein.

- ▶ Ein verwendeter Eingang darf nicht potenzialfrei bleiben.
- ▶ Das bedeutet es muss immer eine Verbindung entweder mit 5V oder GND hergestellt sein.
- ▶ Falls dies nicht der Fall ist, ist der Zustand undefiniert.

- ▶ Ein verwendeter Eingang darf nicht potenzialfrei bleiben.
- ▶ Das bedeutet es muss immer eine Verbindung entweder mit 5V oder GND hergestellt sein.
- ▶ Falls dies nicht der Fall ist, ist der Zustand undefiniert.
- ▶ Selbst kleinste elektromagnetische Felder können den Eingangszustand verändern.

- ▶ Ein verwendeter Eingang darf nicht potenzialfrei bleiben.
- ▶ Das bedeutet es muss immer eine Verbindung entweder mit 5V oder GND hergestellt sein.
- ▶ Falls dies nicht der Fall ist, ist der Zustand undefiniert.
- ▶ Selbst kleinste elektromagnetische Felder können den Eingangszustand verändern.
- ▶ Im Programm sieht es so aus als ob der Eingang zufällig 0 oder 1 ist.

- ▶ Wenn ein Taster als Eingang verwendet wird muss sicher gestellt werden, dass der Eingang nie potentialfrei ist.

- ▶ Wenn ein Taster als Eingang verwendet wird muss sicher gestellt werden, dass der Eingang nie potentialfrei ist.
- ▶ Dazu gibt es drei Möglichkeiten.

- ▶ Der Taster wird zwischen VCC und Eingangspin geschaltet.

- ▶ Der Taster wird zwischen VCC und Eingangspin geschaltet.
- ▶ Zusätzlich verbindet eine Widerstand (typischerweise  $10\text{ k}\Omega$ ) den Pin mit GND.



- ▶ Der Taster wird zwischen VCC und Eingangspin geschaltet.
- ▶ Zusätzlich verbindet eine Widerstand (typischerweise  $10\text{ k}\Omega$ ) den Pin mit GND.
- ▶ Wird der Taster gedrückt wird eine logische 1 im Programm gelesen.

- ▶ Der Taster wird zwischen VCC und Eingangspin geschaltet.
- ▶ Zusätzlich verbindet eine Widerstand (typischerweise  $10\text{ k}\Omega$ ) den Pin mit GND.
- ▶ Wird der Taster gedrückt wird eine logische 1 im Programm gelesen.
- ▶ Wird der Taster nicht gedrückt eine logische 0.

# Externer pull-down Widerstand

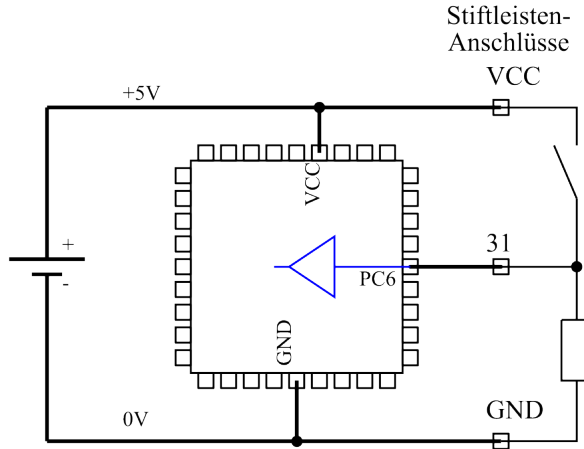


Figure 8: Taster an AVR Mikrocontroller mit pull-down Widerstand

**Beispiel** Schließen Sie den Taster mit externen pull-down Widerstand an den Microcontroller und starten Sie die Simulation.

- ▶ Der Taster wird zwischen GND und Eingangspin geschaltet.

- ▶ Der Taster wird zwischen GND und Eingangspin geschaltet.
- ▶ Zusätzlich verbindet eine Widerstand (typischerweise  $10\text{ k}\Omega$ ) den Pin mit VCC.

- ▶ Der Taster wird zwischen GND und Eingangspin geschaltet.
- ▶ Zusätzlich verbindet eine Widerstand (typischerweise  $10\text{ k}\Omega$ ) den Pin mit VCC.
- ▶ Wird der Taster gedrückt wird eine logische 0 im Programm gelesen.

- ▶ Der Taster wird zwischen GND und Eingangspin geschaltet.
- ▶ Zusätzlich verbindet eine Widerstand (typischerweise  $10\text{ k}\Omega$ ) den Pin mit VCC.
- ▶ Wird der Taster gedrückt wird eine logische 0 im Programm gelesen.
- ▶ Wird der Taster nicht gedrückt eine logische 1.



# Externer pull-up Widerstand

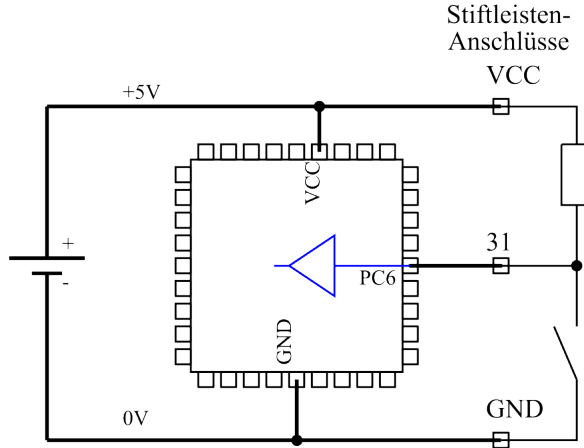


Figure 9: Taster an AVR Mikrocontroller mit pull-up Widerstand

**Beispiel** Schließen Sie den Taster mit externen pull-up Widerstand an den Microcontroller und starten Sie die Simulation.

# Interner pull-up Widerstand

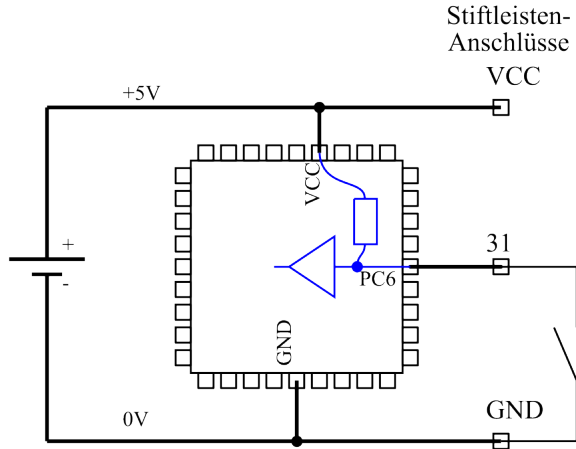


Figure 10: Taster an AVR Mikrocontroller mit internem pull-up Widerstand

**Beispiel** Schließen Sie den Taster mit internem pull-up Widerstand an den Microcontroller und starten Sie die Simulation.