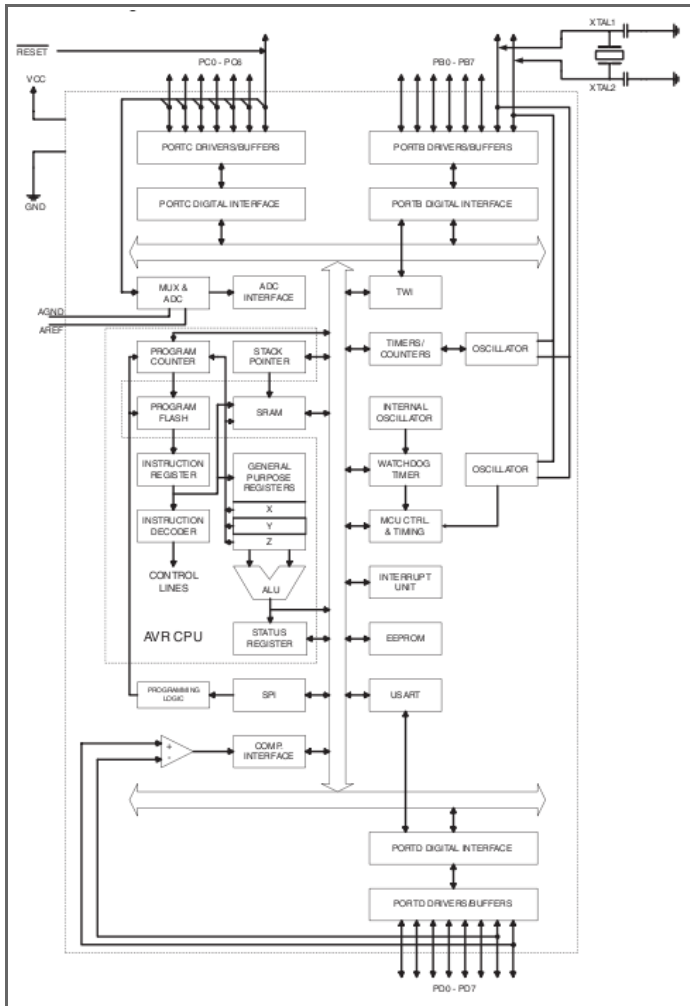


```
#include <avr/io.h>
void main (void) {
  DDRB = 0b00000001; //1==Ausgang
  PORTB= 0b00000000; //1== HIGH, 0 == LOW
  while(1); //endlosschleife
}
```



AVR-Mikrocontroller

Thaddäus Krönert
kroenert@cs.tu-berlin.de

PROJEKT
LABOR

```
#include <avr/io.h>
void main (void) {
  DDRB = 0b00000000; //Eingang ; 1==Ausgang
  PORTB= _BV(PB1); //Pullup ; 0==ohne Pullup

  int i; // int ist 8-bit gross auf einem Atmega

  do {
    i = PINB; //Lesen des gesamten PORTs B
    while(!(i & (_BV(PB1)))); //<- Alternative!
    while(1); //endlosschleife
  }
```

ATmega48/88/168/328 Arduino

reset (RESET) PC6	1	28	PC5	analog input 5
(RX) digital 0 (RXD) PD0	2	27	PC4	analog input 4
(TX) digital 1 (TXD) PD1	3	26	PC3	analog input 3
digital 2 PD2	4	25	PC2	analog input 2
(pwm) digital 3 PD3	5	24	PC1	analog input 1
digital 4 PD4	6	23	PC0	analog input 0
VCC	7	22	GND	
GND	8	21	AREF	
(XTAL1) PB6	9	20	AVCC	
(XTAL2) PB7	10	19	PB5 (SCK)	digital 13
(pwm) digital 5 PD5	11	18	PB4 (MISO)	digital 12
(pwm) digital 6 PD6	12	17	PB3 (MOSI)	digital 11 (pwm)
digital 7 PD7	13	16	PB2	digital 10 (pwm)
digital 8 PB0	14	15	PB1	digital 9 (pwm)

```
#include <avr/interrupt.h>

//Timer0 overflow vector-ISR:
ISR (TIMER0_OVF_vect)

{
  // insert interrupt-handler here
}
```

```
int main (void)
{
  // io-clk / 1024:
  TCCR0 |= (_BV(CS00) | _BV(CS02));
  // Interrupt bei overflow aktivieren:
  TIMSK |= _BV(TOIE0);
  sei();
  while (1);
}
```

Software:

- AVR-Studio (Windows, GUI)
- KontrollerLAB (Linux, GUI)
- avr-gcc (compiler, commandline)
- Arduino (Komplette IDE fuer Arduinos)

Programmer:

- USBasp (~5Euro zum selberloeten)
- Dragon (~60Euro , aber mit debugWire)

Evaluationboards:

- Steckbrett (Breadini)
- STK500 (teuer, aber von Atmel)
- stk500 aehnliche Nachbauten (noname)
- Arduino (~20Euro, viele Beispiele)