Input/Output

Pinmode	pinMode(<pin>, INPUT/OUTPUT)	pinMode(21, INPUT);
	DDRx (x = Port) Bit auf 0: INPUT, 1: OUTPUT	DDRB |= (1 << DDB3); DDRB &= ~(1 << DDB3);
Output (write)	digitalWrite(<pin>)	digitalWrite(21);
	Wenn Pin auf OUTPUT: PORTx = 1 (HIGH) 0 (LOW)	PORTB |= (1 << P3);
Input (read)	Wenn Pin auf INPUT: PINx = 1 (HIGH) 0 (LOW)	if (PINB & (1 << PINB3)) { … }

Einzelne Bits adressieren: Manual Seite 96

PORT: Px<0-7>

DDR: DDx<0-7>

PIN: PINx<0-7>

Interrupts

Global aktivieren	sei()	
	SREG |= (1 << 7);	Manual Seite 13
Global deaktivieren	cli()	
	SREG &= ~(1 << 7);	
Spezielle Interrupts aktivieren/deaktivieren	EIMSK |= (1 << INTx)	EIMSK |= (1 << INT5); Manual Seite 111
Interrupt Flags	EIFR Register	Manual Seite 112
Steigende, oder fallende Flanke?	INT 0-3: EICRA INT 4-7: EICRB	Manual Seite 110 EICRB |= (1 << ISC40) | (1 << ISC41) In diesem Fall steigende Flanke für INT4
ISR definieren	ISR(vector) { … }	ISR(INT0_vect) { … }
Interrupt mit Arduino Library	attachInterrupt(<quelle>, <ISR name>, <mode>)	attachInterrupt(digitalPinToInterrupt(21), count, RISING); //count ist eine definierte Funktion

Verschiedene Interrupt-Vektoren für ISR:

INT0_vect // externer Interrupt 0

TIMER4_COMPA_vect // Timer 4 hat Vergleichswert in OCR4A erreicht

TIMER4_OVF_vect // Timer 4 ist übergelaufen

Timer (n ist der jeweilige Timer)		
Pulsweitenmodulation	TCCRnA	TCCR3A = 0x00;
Prescaler setzen und Timer starten	TCCRnB	TCCR3B = 0x00; TCCR3B |= (1 << CS30) | (1 << CS32); // Prescaler auf 16 (Wenn man keinen Prescaler benutzt, muss auch ein Bit gesetzt werden!!!) Manual Seite 157
Aktueller Zählerstand	TCNTn	TCNT4 = 0x00;
Output Compare Register	OCRnA OCRnB	OCR4A = 62500; ISR(TIMER4_COMPA_vect) { … } ISR wird aufgerufen, wenn Timer 62500 erreicht
Wert für Input Capture	ICRn	
Timer Interrupts aktivieren/deaktivieren	TIMSKn	TIMSK4 = (1 << OCIE4A); Manual Seite 161f.
Einzelne Interrupt Flags für Timer	TIFRn	