Mnomic	Operand[5]	Beschreibung	Operation	Flags	Takte	Wörter[4]
Arithmetische und Logische Befehle						
ADD	Rd, Rr	Addition ohne Übertrag	Rd=Rd+Rr	Z,C,N,V,S,H	1	1
ADC	Rd, Rr	Addition mit Übertrag	Rd=Rd+Rr+C	Z,C,N,V,S,H	1	1
SUB	Rd, Rr	Subtraktion ohne Übertrag	Rd=Rd-Rr	Z,C,N,V,S,H	1	1
SUBI[1]	Rd, K	Subtraktion einer Konstante	Rd=Rd-K	Z,C,N,V,S,H	1	1
SBC	Rd, Rr	Subtraktion mit Übertrag	Rd=Rd-Rr-C	Z,C,N,V,S,H	1	1
SBCI[1]	Rd, K	Subtraktion einer Konstante mit Übertrag	Rd=Rd-K-C	Z,C,N,V,S,H	1	1
INC	Rd	Inkrement	Rd=Rd+1	Z,N,V,S	1	1
DEC	Rd	Dekrement	Rd=Rd-1	Z,N,V,S	1	1
NEG	Rd	Zweierkomplement	Rd=$\overline{Rd}$ + 1	Z,C,N,V,S,H	1	1
AND	Rd, Rr	Logische AND Verknüpfung	Rd=Rd∧Rr	Z,N,V,S	1	1
ANDI[1]	Rd, K	Logische AND Verknüpfung mit Konstante	Rd=Rd∧K	Z,N,V,S	1	1
OR	Rd, Rr	Logische OR Verknüpfung	Rd=Rd∧Rr	Z,N,V,S	1	1
ORI[1]	Rd, K	Logische OR Verknüpfung mit Konstante	Rd=Rd∨K	Z,N,V,S	1	1
EOR	Rd, Rr	Logische Exclusiv-OR Verknüpfung	Rd=Rd⊕Rr	Z,N,V,S	1	1
COM	Rd	Einerkomplement	Rd=$\overline{Rd}$	Z,C,N,V,S	1	1
CLR	Rd	Setze Register auf 0	Rd=Rd⊕Rd	Z,N,V,S	1	1
SER[1]	Rd	Setze Register auf 0xFF	Rd=0xFF		1	1
Sprungbefehle und Vergleich						
RJMP	k	Relativer Sprung (-2048 < k < 2048)	PC=PC+k+1		2	1
JMP	k	Absoluter Sprung	PC=k		3	2
RCALL[2]	k	Relativer Subroutinenaufruf (-2048 < k < 2048)	PC=PC+k+1		3	1
CALL[2]	K	Absoluter Subroutinenaufruf	PC=k		4	2
RET[2]		Rücksprung von Subroutine	PC=[Stack]		4	1
RETI[2]		Rücksprung von Interrupt-Routine	PC=[Stack]	I	4	1
CP	Rd,Rr	Vergleich	Rd-Rr	Z,C,N,V,S,H	1	1
CPC	Rd,Rr	Vergleich mit Übertrag	Rd-Rr-C	Z,C,N,V,S,H	1	1
CPI[1]	Rd,K	Vergleich mit Konstante	Rd-K	Z,C,N,V,S,H	1	1
TST	Rd	Test auf Null oder Negativ	Rd=Rd∧Rd	Z,N,V,S	1	1
BRBS[3]	s,k	Sprung, wenn Status Flag s gesetzt	Ja:PC=PC+k+1		1/2	1
BRBC[3]	s,k	Sprung, wenn Status Flag s nicht gesetzt	Ja:PC=PC+k+1		1/2	1
BREQ[3]	k	Sprung, wenn gleich bzw. Null Flag gesetzt (Z=1)	Ja:PC=PC+k+1		1/2	1
BRNE[3]	k	Sprung, wenn ungleich bzw. Null Flag nicht g.(Z=0)	Ja:PC=PC+k+1		1/2	1
BRCS[3]	k	Sprung, wenn Übertrag Flag gesetzt (C=1)	Ja:PC=PC+k+1		1/2	1
BRCC[3]	k	Sprung, wenn Übertrag Flag nicht gesetzt (C=0)	Ja:PC=PC+k+1		1/2	1
BRSH[3]	k	Sprung, wenn gleich oder höher (C=0)	Ja:PC=PC+k+1		1/2	1
BRLO[3]	k	Sprung, wenn kleiner (C=1)	Ja:PC=PC+k+1		1/2	1
Datentransfer						
MOV	Rd,Rr	Kopiere Register	Rd=Rr		1	1
LDI[1]	Rd, K	Lade Konstante in Register	Rd=K		1	1
LDS	Rd, k	Lade von Adresse k des Datenspeicher	Rd=(k)		2	2
LD	Rd, X	Lade Indirekt mittels Register X (R27:R26)	Rd=(X)		2	1
LD	Rd, X+	Lade indirekt mittels Register X, Postinkrement	Rd=(X);X=X+1		2	1
LD	Rd, -X	Lade indirekt mittels Register X, Predekrement	X=X-1;Rd=(X)		2	1
LD	Rd, Y	Lade Indirekt mittels Register Y (R29:R28)	Rd=(Y)		2	1
LD	Rd, Y+	Lade indirekt mittels Register Y, Postinkrement	Rd=(Y);Y=Y+1		2	1
LD	Rd, -Y	Lade indirekt mittels Register Y, Predekrement	Y=Y-1;Rd=(Y)		2	1
LDD	Rd, Y+q	Lade indirekt mittels Register Y mit Offset	Rd=(Y+q)		2	1
LD	Rd, Z	Lade Indirekt mittels Register Z (R31:R30)	Rd=(Z)		2	1
LD	Rd, Z+	Lade indirekt mittels Register Z, Postinkrement	Rd=(Z);Z=Z+1		2	1
LD	Rd, -Z	Lade indirekt mittels Register Z, Predekrement	Z=Z-1;Rd=(Z)		2	1
LDD	Rd, Z+q	Lade indirekt mittels Register Z mit Offset	Rd=(Z+q)		2	1

Mnomic	Operand	Beschreibung	Operation	Flags	Takte	Wörter[4]
STS	k, Rr	Speichere in Adresse k des Datenspeichers	(k)=Rr		2	2
ST	X, Rr	Speichere indirekt mittels Register X	(X)=Rr		2	1
ST	X+, Rr	Speichere indirekt mittels Register X, Postinkrement	(X)=Rr;X=X+1		2	1
ST	-X, Rr	Speichere indirekt mittels Register X, Predekrement	X=X-1;(X)=Rr		2	1
ST	Y, Rr	Speichere indirekt mittels Register Y	(Y)=Rr		2	1
ST	Y+, Rr	Speichere indirekt mittels Register Y, Postinkrement	(Y)=Rr;Y=Y+1		2	1
ST	-Y, Rr	Speichere indirekt mittels Register Y, Predekrement	Y=Y-1;(Y)=Rr		2	1
STD	Z+q, Rr	Speichere indirekt mittels Register Z und Offset	(Z+q)=Rr		2	1
ST	Z, Rr	Speichere indirekt mittels Register Z	(Z)=Rr		2	1
ST	Z+, Rr	Speichere indirekt mittels Register Z, Postinkrement	(Z)=Rr;Z=Z+1		2	1
ST	-Z, Rr	Speichere indirekt mittels Register Z, Predekrement	Z=Z-1;(Z)=Rr		2	1
STD	Z+q, Rr	Speichere indirekt mittels Register Z und Offset	(Z+q)=Rr		2	1
IN	Rd, A	Lese Wert aus I/O Bereich	Rd=I/O(A)		1	1
OUT	A, Rr	Schreibe Wert im I/O Bereich	I/O(A)=Rd		1	1
PUSH[2]	Rr	Schreibt Wert auf den Stack	[Stack]=Rr		2	1
POP[2]	Rd	Lese Wert vom Stack	Rd=[Stack]		2	1
Bit-Operationen						
LSL	Rd	Logisches Schieben nach links	Rd=Rd<<1	Z,C,N,V,H	1	1
LSR	Rd	Logisches Schieben nach rechts	Rd=Rd>>1	Z,C,N,V	1	1
ROL	Rd	Rotieren nach links über das Übertragsbit	Rd=Rd<<1+C	Z,C,N,V,H	1	1
ROR	Rd	Rotieren nach rechts über das Übertragsbit	Rd=C:(Rd>>1)	Z,C,N,V	1	1
ASR	Rd	Arithmetisches Schieben nach rechts (Vorzeichen)	Rd=Rd>>1	Z,C,N,V	1	1
Sonstige						
NOP		Keine Operation			1	1
SEI		Setze globale Interrupt Freigabe	I = 1		1	1
CLI		Lösche globale Interrupt Freigabe	I = 0		1	1
LPM	Rd, Z	Lade durch Z adressierten Wert aus dem Flash	Rd=F(Z)		3	1
LPM	Rd, Z+	Lade Wert aus Flash, Postinkrement von Z	Rd=F(Z), Z=Z+1		3	1

1) Geht für Rd nur mit Register 16 bis 31
2) Stackpointer wird während der Operation verändert
3) Wenn die Bedingung zutrifft werden 2 Takte benötigt, ansonsten 1 Takt; k: -2048 bis 2048
4) Ein Wort besteht beim AVR Befehlssatz aus 16 Bit
5) Rd: Zielregister R0-R31 bzw. R16-R32, Rr: Operandregister R0-R31, s: 0-7, q: 0-63, K: 0-255

Register

R0-R15: Generelle Register, nicht mit Befehlen mit Konstanten verwendbar
R16-R31: Generelle Register
R27:R26 : X-Register
R29:R28 : Y-Register (unterstützt Displacment Adressierung)
R31:R30 : Z-Register (unterstützt Displacment-Adressierung, unterstützt Adressierung des Flash-Speichers)

Wichtige IO Register

SREG	Statusregister	7:I Interruptfreig.	6:T Bitspeicher	5:H Halbübertrag	4:S Vorzeichen	3:V 2er Übertrag	2:N Negativ	1:Z Null	0:C Übertrag
SPL	Stackpointer Low	Untere 8-Bit des 16 Bit Stackpointers							
SPH	Stackpointer High	Obere 8-Bit des 16 Bit Stackpointers							
DDRx	Portrichtung	Für Port x[A,B,C,...]: 0-Eingang, 1-Ausgang							
PORTx	Portausgabe	Port als Eingang: 0–Hochohmig, 1–Pullup Widerstand, als Ausgang: 0–Masse, 1-Versorgung							
PINx	Porteingang	Pegel, der am Pin anliegt: 0-Logisch 0 Pegel, 1-Logisch 1 Pegel							