## AVR Befehlssatz (Auszug)

Arithmetische und Logische Befehle	Mnomic	Onerand <sup>5</sup>	Beschreibung	Operation	Flags	Takte	Wörter <sup>4</sup>				
ADD				- polation	լ. ացշ	Tanto	1101101				
ADC											
SUB											
Subtraction einer Konstante			•								
SBC		-	<u> </u>		<del>  ' ' ' ' ' ' '   '   '   '   '   '   '</del>						
SBC1											
TNC			-			-					
DEC         Rd         Dekrement         Rd=Rd-1         Z,N,V,S         1         1           NEG         Rd         Zweierkomplement         Rd=Rd+1         Z,C,N,V,S         1         1           AND         Rd, Rr         Logische AND Verknüpfung         Rd=Rd∧R         2,N,V,S         1         1           AND1*         Rd, K         Logische AND Verknüpfung mit Konstante         Rd=Rd∧R         Z,N,V,S         1         1           AND1*         Rd, K         Logische AND Verknüpfung mit Konstante         Rd=Rd∧R         Z,N,V,S         1         1           GR         Rd, Rr         Logische Exclusiv-OR Verknüpfung         Rd=Rd⊕R         Z,N,V,S         1         1           EOR         Rd, Rr         Logische Exclusiv-OR Verknüpfung         Rd=Rd⊕R         Z,N,V,S         1         1           CCM         Rd         Seze Register auf 0         Rd=Rd⊕Rd         Z,N,V,S         1         1           SER*         Rd         Setze Register auf 0xFF         Rd=0xFF         NC-PC+k+1         2         1           Sprungbefehle und Vergleich         Sterze Register auf 0xFF         Rd=0xFF         NV,S         1         1           Sprungbefehle und Vergleich         Rd         Ac0xFF			•								
NE6					+ ' ' '						
AND											
ANDI®   Rd, K			·								
OR         Rd, Rr         Logische OR Verkrüpfung         Rd=Rd AR         Z,N,V,S         1         1           ORT¹         Rd, K         Logische OR Verkrüpfung mit Konstante         Rd=Rd WK         Z,N,V,S         1         1           EOR         Rd, Rr         Logische Exclusiv-OR Verknüpfung         Rd=RdeRd         Z,N,V,S         1         1           COM         Rd         Einerkomplement         Rd=Rd         Z,C,N,V,S         1         1           CLR         Rd         Setze Register auf         Rd=RdeRd         Z,N,V,S         1         1           SER¹         Rd         Setze Register auf         Rd=RdeRdeRd         Z,N,V,S         1         1           Sprungbefehle und Vergleich           RJMP         k         Relativer Subroutinen         PC=PC+k+1         2         1         3         2         1           Sprungbefehle und Vergleich           Sprungbefehle und Vergleich           Sprungbefehle und Vergleich           Setze Register auf 0xFF         Rd=0xFF         1         1         2         1           Sprungbefehle und Vergleich         Rd=0xBefe         2048         2         2048         2											
ORI¹         Rd, K         Logische OR Verknüpfung mit Konstante         Rd=RdvK         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=Rd         Z,C,N,V,S         1         1           CLR         Rd         Setze Register auf         Rd=Rd@Rd         Z,N,V,S         1         1           SER¹         Rd         Setze Register auf 0xFF         Rd=0xFF         1         1         1           SER¹         Rd         Setze Register auf 0xFF         Rd=0xFF         1         1         1           Sprungbefelle und Vergleich         RSMP         Rd=deliver Sprung (*2048 < k < 2048)											
EOR         Rd, Rr         Logische Exclusiv-OR Verknüpfung         Rd=Rd⊕Rr         Z,N,V,S         1         1           COM         Rd         Einerkomplement         Rd=Rd         Z,C,N,V,S         1         1         1           CLR         Rd         Setze Register auf 0xFF         Rd=Rd⊕Rd         Z,N,V,S         1         1           SER³ Rd         Setze Register auf 0xFF         Rd=Rd⊕Rd         Z,N,V,S         1         1           Sprungbefehle und Vergleich           ROBAL**         Relativer Sprung (-2048 < k < 2048)         PC=PC+k+1         2         1         2           Sprungbefehle und Vergleich           Weld Relativer Subroutinenaufruf (-2048 < k < 2048)         PC=PC+k+1         2         1         3         1           RCALL² k         Relativer Subroutinenaufruf (-2048 < k < 2048)         PC=PC+k+1         3         1         2         4         2         2         1         4         4         2         2         1         4         4         2         2         1         4         4         2         2         4         2         2         1         4         4         2         2         2											
COM         Rd         Einerkomplement         Rd=Rd         Z,C,N,V,S         1         1           CLR         Rd         Setze Register auf 0         Rd=Rd⊕Rd         Z,N,V,S         1         1           SEZE¹         Rd         Setze Register auf 0XFF         Rd=Rd⊕Rd         Z,N,V,S         1         1           Sprung befeite and Vergleich           RJMP         k         Relativer Sprung (-2048 < k < 2048)											
CLR         Rd         Setze Register auf 0         Rd=Rd⊕Rd         Z.N.V.S         1         1           SER¹         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=PC-k         3         2           RCALL²         k         Relativer Subroutinenaufruf         PC=PC-k         4         2           RET²         Rücksprung von Subroutine         PC=[Stack]         4         1           RET²         Rücksprung von Subroutine         PC=[Stack]         4         1           CP         Rd,Rr         Vergleich         Rd-Rr         Z,C,N,V,S,H         1         1           CPC         Rd,Rr         Vergleich mit Konstante         Rd-Rr-C         Z,C,N,V,S,H         1         1           CPI¹         Rd, K         Vergleich mit Konstante         Rd-Rr-C         Z,C,N,V,S,H         1         1           CPI¹         Rd, K         Vergleich mit Konstante         Rd-Rr-C         Z,C,N,V,S,H         1         1           SBRS³		-	• • • • • • • • • • • • • • • • • • • •	<del></del>	1 1 1						
SER¹         Rd         Setze Register auf 0xFF         Rd=0xFF         1         1         1           Sprungbefehle und Vergleich           RJMP         k         Relativer Sprung (-2048 < k < 2048)			·	1							
Sprungbefehle und Vergleich           RJMP         k         Relativer Sprung (-2048 < k < 2048)					Z,N,V,S		1				
RJMP   k				Rd=0xFF		1	1				
JMP							T				
RCALL²         k         Relativer Subroutinenaufruf (-2048 < k < 2048)			, ,				1				
CALL²         K         Absoluter Subroutinenaufruf         PC=k         4         2           RETT²         Rücksprung von Subroutine         PC=[Stack]         4         1           RETT²         Rücksprung von Interrupt-Routine         PC=[Stack]         1         4         1           CP         Rd,Rr         Vergleich         Rd-Rr         Z,C,N,V,S,H         1         1           CPC         Rd,Rr         Vergleich mit Konstante         Rd-K-C         Z,C,N,V,S,H         1         1           CPI¹         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           BRBS3         s,k         Sprung, wenn Status Flag s gesetzt         Ja:PC=PC+k+1         1/2         1           BRBC3³         k         Sprung, wenn Status Flag s gesetzt (Z=1)         Ja:PC=PC+k+1         1/2         1           BRNE³         k         Sprung, wenn gleich bzw. Null Flag gesetzt (Z=1)         Ja:PC=PC+k+1         1/2         1           BRNE³         k         Sprung, wenn übertrag Flag gesetzt (C=1)         Ja:PC=PC+k+1         1/2         1           BRC3³ <td></td> <td>k</td> <td></td> <td></td> <td></td> <td></td> <td>2</td>		k					2				
RET2         Rücksprung von Subroutine         PC=[Stack]         4         1           RET12         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           CPI1         Rd,K         Vergleich mit Konstante         Rd-K         Z,C,N,V,S,H         1         1           TST         Rd         Test auf Null oder Negativ         Rd-Rd ARd         Z,N,V,S         1         1           BRBS3         s,k         Sprung, wenn Status Flag s gesetzt         Ja:PC=PC+k+1         1/2         1           BRBC3         s,k         Sprung, wenn Status Flag s nicht gesetzt         Ja:PC=PC+k+1         1/2         1           BRE03         k         Sprung, wenn Ubertrag Flag gesetzt (Z=1)         Ja:PC=PC+k+1         1/2         1           BRNE3         k         Sprung, wenn Übertrag Flag gicht gicht gesetzt (C=0)         Ja:PC=PC+k+1         1/2         1           BRC3         k         Sprung, wenn Übertrag Flag nicht gesetzt (C=0)         Ja:PC=PC+k+1         1/2         1 </td <td></td> <td>k</td> <td>, ,</td> <td>PC=PC+k+1</td> <td></td> <td>3</td> <td>1</td>		k	, ,	PC=PC+k+1		3	1				
RETT²         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           CPI¹         Rd,K         Vergleich mit Konstante         Rd-K         Z,C,N,V,S,H         1         1           TST         Rd         Test auf Null oder Negativ         Rd-Rd ARd         Z,N,V,S         1         1           BRBS³         s,k         Sprung, wenn Status Flag s gesetzt         Ja:PC=PC+k+1         1/2         1           BRBC³         s,k         Sprung, wenn Status Flag s nicht gesetzt         Ja:PC=PC+k+1         1/2         1           BREQ³         k         Sprung, wenn gleich bzw. Null Flag gesetzt (Z=1)         Ja:PC=PC+k+1         1/2         1           BRNE³         k         Sprung, wenn Übertrag Flag gesetzt (C=1)         Ja:PC=PC+k+1         1/2         1           BRC3³         k         Sprung, wenn Übertrag Flag nicht gesetzt (C=0)         Ja:PC=PC+k+1         1/2         1           BRC3³         k         Sprung, wenn gleich oder höher (C=0)         Ja:PC=PC+k+1         1/		K	Absoluter Subroutinenaufruf	PC=k		4	2				
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           CPI¹         Rd,K         Vergleich mit Konstante         Rd-K         Z,C,N,V,S,H         1         1           TST         Rd         Test auf Null oder Negativ         Rd-Rd ARd         Z,N,V,S         1         1           BRBS³         S,k         Sprung, wenn Status Flag s gesetzt         Ja:PC=PC+k+1         1/2         1           BRBC³         S,k         Sprung, wenn Status Flag s nicht gesetzt         Ja:PC=PC+k+1         1/2         1           BRE0³         k         Sprung, wenn gleich bzw. Null Flag gesetzt (Z=1)         Ja:PC=PC+k+1         1/2         1           BRNE³         k         Sprung, wenn ungleich bzw. Null Flag gincht g.Z=0         Ja:PC=PC+k+1         1/2         1           BRC3³         k         Sprung, wenn Übertrag Flag gesetzt (C=1)         Ja:PC=PC+k+1         1/2         1           BRC3³         k         Sprung, wenn bleich oder höher (C=0)         Ja:PC=PC+k+1         1/2         1           BRS4³         k         Sprung, wenn kleiner (C=1)         Ja:PC=PC+k+1         1			Rücksprung von Subroutine	PC=[Stack]		4	1				
CPC         Rd,Rr         Vergleich mit Übertrag         Rd-Rr-C         Z,C,N,V,S,H         1         1           CPI¹         Rd,K         Vergleich mit Konstante         Rd-K         Z,C,N,V,S,H         1         1           TST         Rd         Test auf Null oder Negativ         Rd=Rd ARd         Z,N,V,S         1         1           BRBS³         s,k         Sprung, wenn Status Flag s gesetzt         Ja:PC=PC+k+1         1/2         1           BRBC³         s,k         Sprung, wenn Status Flag s nicht gesetzt         Ja:PC=PC+k+1         1/2         1           BREQ³         s,k         Sprung, wenn gleich bzw. Null Flag gesetzt (Z=1)         Ja:PC=PC+k+1         1/2         1           BRC³³         k         Sprung, wenn übertrag Flag gesetzt (C=0)         Ja:PC=PC+k+1         1/2         1           BRC³³         k         Sprung, wenn Übertrag Flag nicht gesetzt (C=0)         Ja:PC=PC+k+1         1/2         1           BRC³³         k         Sprung, wenn gleich oder höher (C=0)         Ja:PC=PC+k+1         1/2         1           BRS³³         k         Sprung, wenn kleiner (C=1)         Ja:PC=PC+k+1         1/2         1           BRL0³³         k         Sprung, wenn ber         Rd=R         1         1 <td></td> <td></td> <td>Rücksprung von Interrupt-Routine</td> <td>PC=[Stack]</td> <td>I</td> <td>4</td> <td>1</td>			Rücksprung von Interrupt-Routine	PC=[Stack]	I	4	1				
CPI¹         Rd,K         Vergleich mit Konstante         Rd-K         Z,C,N,V,S,H         1         1           TST         Rd         Test auf Null oder Negativ         Rd=Rd ARd         Z,N,V,S         1         1           BRBS³         S,k         Sprung, wenn Status Flag s gesetzt         Ja:PC=PC+k+1         1/2         1           BRBC³         S,k         Sprung, wenn Status Flag s nicht gesetzt         Ja:PC=PC+k+1         1/2         1           BREQ³         k         Sprung, wenn gleich bzw. Null Flag gesetzt (Z=1)         Ja:PC=PC+k+1         1/2         1           BRCS³         k         Sprung, wenn ungleich bzw. Null Flag nicht g.(Z=0)         Ja:PC=PC+k+1         1/2         1           BRCS³         k         Sprung, wenn übertrag Flag gesetzt (C=1)         Ja:PC=PC+k+1         1/2         1           BRCS³         k         Sprung, wenn übertrag Flag nicht gesetzt (C=0)         Ja:PC=PC+k+1         1/2         1           BRSH³         k         Sprung, wenn kleiner (C=0)         Ja:PC=PC+k+1         1/2         1           BRL0³         k         Sprung, wenn kleiner (C=1)         Ja:PC=PC+k+1         1/2         1           BRL0³         k         Sprung, wenn in geich oder höher (C=0)         Ja:PC=PC+k+1         1/		Rd,Rr	Vergleich	Rd-Rr	Z,C,N,V,S,H	1	1				
TST         Rd         Test auf Null oder Negativ         Rd=Rd∧Rd         Z,N,V,S         1         1           BRBS³         s,k         Sprung, wenn Status Flag s gesetzt         Ja:PC=PC+k+1         1/2         1           BRBC³         s,k         Sprung, wenn Status Flag s nicht gesetzt         Ja:PC=PC+k+1         1/2         1           BREQ³         k         Sprung, wenn gleich bzw. Null Flag gesetzt (Z=1)         Ja:PC=PC+k+1         1/2         1           BRNB³         k         Sprung, wenn ungleich bzw. Null Flag nicht g.(Z=0)         Ja:PC=PC+k+1         1/2         1           BRCS³         k         Sprung, wenn Übertrag Flag gesetzt (C=1)         Ja:PC=PC+k+1         1/2         1           BRCC³         k         Sprung, wenn Übertrag Flag nicht gesetzt (C=0)         Ja:PC=PC+k+1         1/2         1           BRC3³         k         Sprung, wenn büertrag Flag nicht gesetzt (C=0)         Ja:PC=PC+k+1         1/2         1           BRC3³         k         Sprung, wenn übertrag Flag nicht gesetzt (C=0)         Ja:PC=PC+k+1         1/2         1           BRC3³         k         Sprung, wenn kleiner (C=1)         Ja:PC=PC+k+1         1/2         1           BRC3³         k         Sprung, wenn kleiner (C=1)         Ja:PC=PC+k+1		Rd,Rr	Vergleich mit Übertrag	Rd-Rr-C	Z,C,N,V,S,H	1	1				
BRBS³ s,k Sprung, wenn Status Flag s gesetzt Ja:PC=PC+k+1 1/2 1 BRBC³ s,k Sprung, wenn Status Flag s nicht gesetzt Ja:PC=PC+k+1 1/2 1 BREQ³ k Sprung, wenn gleich bzw. Null Flag gesetzt (Z=1) Ja:PC=PC+k+1 1/2 1 BRNE³ k Sprung, wenn ungleich bzw. Null Flag nicht g.(Z=0) Ja:PC=PC+k+1 1/2 1 BRCS³ k Sprung, wenn übertrag Flag gesetzt (C=1) Ja:PC=PC+k+1 1/2 1 BRCS³ k Sprung, wenn Übertrag Flag gesetzt (C=1) Ja:PC=PC+k+1 1/2 1 BRCC³ k Sprung, wenn Übertrag Flag nicht gesetzt (C=0) Ja:PC=PC+k+1 1/2 1 BRSH³ k Sprung, wenn gleich oder höher (C=0) Ja:PC=PC+k+1 1/2 1 BRLO³ k Sprung, wenn kleiner (C=1) Ja:PC=PC+k+1 1/2 1 BRLO³ k Sprung, wenn kleiner (C=1) Ja:PC=PC+k+1 1/2 1 Datentransfer  MOV Rd,Rr Kopiere Register Rd=Rr 1 1 1 LDI¹ Rd, K Lade Konstante in Register Rd=K 1 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 LD Rd, Y+ Lade indirekt mittels Register Y, Predekrement Y=Y-1;Rd=(Y) 2 1 LD Rd, Y+ Lade indirekt mittels Register Y (R29:R28) Rd=(Y+q) 2 1 LD Rd, Z Lade Indirekt mittels Register Y Redekrement Y=Y-1;Rd=(Y) 2 1 LD Rd, Y+ Lade indirekt mittels Register Y mit Offset Rd=(Y+q) 2 1 LD Rd, Z Lade Indirekt mittels Register Y Rit Offset Rd=(Y+q) 2 1 LD Rd, Z Lade Indirekt mittels Register Z (R31:R30) Rd=(Z) 2 1		Rd,K	Vergleich mit Konstante	Rd-K	Z,C,N,V,S,H	1	1				
BRBC3         s,k         Sprung, wenn Status Flag s nicht gesetzt         Ja:PC=PC+k+1         1/2         1           BREQ3         k         Sprung, wenn gleich bzw. Null Flag gesetzt (Z=1)         Ja:PC=PC+k+1         1/2         1           BRNE3         k         Sprung, wenn ungleich bzw. Null Flag nicht g.(Z=0)         Ja:PC=PC+k+1         1/2         1           BRCS3         k         Sprung, wenn Übertrag Flag gesetzt (C=1)         Ja:PC=PC+k+1         1/2         1           BRCC3         k         Sprung, wenn Übertrag Flag nicht gesetzt (C=0)         Ja:PC=PC+k+1         1/2         1           BRCC3         k         Sprung, wenn Übertrag Flag nicht gesetzt (C=0)         Ja:PC=PC+k+1         1/2         1           BRCG3         k         Sprung, wenn Übertrag Flag gesetzt (C=0)         Ja:PC=PC+k+1         1/2         1           BRCG3         k         Sprung, wenn Übertrag Flag gesetzt (C=0)         Ja:PC=PC+k+1         1/2         1           BRCG3         k         Sprung, wenn Übertrag Flag gesetzt (C=0)         Ja:PC=PC+k+1         1/2         1           BRCG3         k         Sprung, wenn Übertrag Flag gister (C=0)         Ja:PC=PC+k+1         1/2         1           BRC4         A         Sprung, wenn übertrag Flag gister Kele         Rd=R<	TST	Rd	Test auf Null oder Negativ	Rd=Rd \( Rd	Z,N,V,S	1	1				
BREQ³ k Sprung, wenn gleich bzw. Null Flag gesetzt (Z=1) Ja:PC=PC+k+1 1/2 1 BRNE³ k Sprung, wenn ungleich bzw. Null Flag nicht g.(Z=0) Ja:PC=PC+k+1 1/2 1 BRCS³ k Sprung, wenn Übertrag Flag gesetzt (C=1) Ja:PC=PC+k+1 1/2 1 BRCC³ k Sprung, wenn Übertrag Flag gesetzt (C=0) Ja:PC=PC+k+1 1/2 1 BRCC³ k Sprung, wenn Übertrag Flag nicht gesetzt (C=0) Ja:PC=PC+k+1 1/2 1 BRSH³ k Sprung, wenn gleich oder höher (C=0) Ja:PC=PC+k+1 1/2 1 BRLO³ k Sprung, wenn kleiner (C=1) Ja:PC=PC+k+1 1/2 1  Datentransfer  MOV Rd,Rr Kopiere Register Rd=R LDI¹ 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 LD Rd, Y+ Lade indirekt mittels Register Y, Predekrement Y=Y-1;Rd=(Y) 2 1 LD Rd, Y+ Lade indirekt mittels Register Y, Predekrement Y=Y-1;Rd=(Y) 2 1 LD Rd, Y+ Lade indirekt mittels Register Y, Predekrement Y=Y-1;Rd=(Y) 2 1 LD 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	BRBS <sup>3</sup>	s,k	Sprung, wenn Status Flag s gesetzt	Ja:PC=PC+k+1		1/2	1				
BRNE³ k Sprung, wenn ungleich bzw. Null Flag nicht g.(Z=0) Ja:PC=PC+k+1 1/2 1 BRCS³ k Sprung, wenn Übertrag Flag gesetzt (C=1) Ja:PC=PC+k+1 1/2 1 BRCC³ k Sprung, wenn Übertrag Flag nicht gesetzt (C=0) Ja:PC=PC+k+1 1/2 1 BRSH³ k Sprung, wenn gleich oder höher (C=0) Ja:PC=PC+k+1 1/2 1 BRLO³ k Sprung, wenn gleich oder höher (C=0) Ja:PC=PC+k+1 1/2 1 BRLO³ k Sprung, wenn kleiner (C=1) Ja:PC=PC+k+1 1/2 1  Datentransfer  MOV Rd,Rr Kopiere Register  LDI¹ 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, Postinkrement Rd=(Y);Y=Y+1 2 1  LD Rd, -Y Lade indirekt mittels Register Y, Predekrement Y=Y-1;Rd=(Y) 2 1  LD Rd, Y+ Lade indirekt mittels Register Y, Predekrement Y=Y-1;Rd=(Y) 2 1  LD Rd, Y+ Lade indirekt mittels Register Y, Predekrement Y=Y-1;Rd=(Y) 2 1  LD Rd, Y+ Lade indirekt mittels Register Y, Predekrement Y=Y-1;Rd=(Y) 2 1  LD Rd, Y+ 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	BRBC <sup>3</sup>	s,k	Sprung, wenn Status Flag s nicht gesetzt	Ja:PC=PC+k+1		1/2	1				
BRCS3         k         Sprung, wenn Übertrag Flag gesetzt (C=1)         Ja:PC=PC+k+1         1/2         1           BRCC3         k         Sprung, wenn Übertrag Flag nicht gesetzt (C=0)         Ja:PC=PC+k+1         1/2         1           BRSH3         k         Sprung, wenn gleich oder höher (C=0)         Ja:PC=PC+k+1         1/2         1           BRL03         k         Sprung, wenn kleiner (C=1)         Ja:PC=PC+k+1         1/2         1           Datentransfer           MOV         Rd,Rr         Kopiere Register         Rd=Rr         1         1         1           LD11         Rd, K         Lade Konstante in Register         Rd=K         1         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, Y         Lade Indirekt mittels Register Y (R29:R28)         Rd=(Y)         2         1           LD         Rd, Y+         Lade indirekt mittels Register Y, Postinkrement         Rd	BREQ <sup>3</sup>	k	Sprung, wenn gleich bzw. Null Flag gesetzt (Z=1)	Ja:PC=PC+k+1		1/2	1				
BRCC³ k Sprung, wenn Übertrag Flag nicht gesetzt (C=0) Ja:PC=PC+k+1 1/2 1 BRSH³ k Sprung, wenn gleich oder höher (C=0) Ja:PC=PC+k+1 1/2 1 BRLO³ k Sprung, wenn kleiner (C=1) Ja:PC=PC+k+1 1/2 1  Datentransfer  MOV Rd,Rr Kopiere Register Rd=Rr 1 1 LDI¹ 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, Postinkrement Rd=(Y);Y=Y+1 2 1 LD Rd, -Y Lade indirekt mittels Register Y, Predekrement Y=Y-1;Rd=(Y) 2 1 LD Rd, -Y Lade indirekt mittels Register Y (R19:R28) 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 LDD Rd, Z Lade Indirekt mittels Register Z (R31:R30) Rd=(Z) 2 1	BRNE <sup>3</sup>	k	Sprung, wenn ungleich bzw. Null Flag nicht g.(Z=0)	Ja:PC=PC+k+1		1/2	1				
BRSH³ k Sprung, wenn gleich oder höher (C=0) Ja:PC=PC+k+1 1/2 1 BRL0³ k Sprung, wenn kleiner (C=1) Ja:PC=PC+k+1 1/2 1  Datentransfer  MOV Rd,Rr Kopiere Register Rd=K 1 1 1 LDI¹ Rd, K Lade Konstante in Register Rd=K 1 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 Rd=(Y);Y=Y+1 2 1 LD Rd, -Y Lade indirekt mittels Register Y, Predekrement 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 LDD Rd, Z Lade Indirekt mittels Register Z (R31:R30) Rd=(Z) 2 1	BRCS <sup>3</sup>	k	Sprung, wenn Übertrag Flag gesetzt (C=1)	Ja:PC=PC+k+1		1/2	1				
BRLO³ k Sprung, wenn kleiner (C=1) Ja:PC=PC+k+1 1/2 1  Datentransfer  MOV Rd,Rr Kopiere Register Rd=R  LDI¹ Rd, K Lade Konstante in Register Rd=K  LDS Rd, k Lade von Adresse k des Datenspeicher Rd=(k) 2  LD Rd, X Lade Indirekt mittels Register X (R27:R26) Rd=(X) 2  LD Rd, X+ Lade indirekt mittels Register X, Postinkrement Rd=(X);X=X+1 2  LD Rd, -X Lade indirekt mittels Register X, Predekrement X=X-1;Rd=(X) 2  LD Rd, Y Lade Indirekt mittels Register Y (R29:R28) Rd=(Y) 2  LD Rd, Y+ Lade indirekt mittels Register Y, Postinkrement Rd=(Y);Y=Y+1 2  LD Rd, -Y Lade indirekt mittels Register Y, Predekrement Rd=(Y);Y=Y+1 2  LD Rd, -Y Lade indirekt mittels Register Y, Predekrement Y=Y-1;Rd=(Y) 2  LD Rd, Y+q Lade indirekt mittels Register Y mit Offset Rd=(Y+q) 2  LD Rd, Z Lade Indirekt mittels Register Z (R31:R30) Rd=(Z) 2	BRCC <sup>3</sup>	k	Sprung, wenn Übertrag Flag nicht gesetzt (C=0)	Ja:PC=PC+k+1		1/2	1				
DatentransferMOVRd,RrKopiere RegisterRd=Rr11LDI¹Rd, KLade Konstante in RegisterRd=K11LDSRd, kLade von Adresse k des DatenspeicherRd=(k)22LDRd, XLade Indirekt mittels Register X (R27:R26)Rd=(X)21LDRd, X+Lade indirekt mittels Register X, PostinkrementRd=(X);X=X+121LDRd, -XLade indirekt mittels Register X, PredekrementX=X-1;Rd=(X)21LDRd, YLade Indirekt mittels Register Y (R29:R28)Rd=(Y)21LDRd, Y+Lade indirekt mittels Register Y, PostinkrementRd=(Y);Y=Y+121LDRd, -YLade indirekt mittels Register Y, PredekrementY=Y-1;Rd=(Y)21LDDRd, Y+qLade indirekt mittels Register Y mit OffsetRd=(Y+q)21LDRd, ZLade Indirekt mittels Register Z (R31:R30)Rd=(Z)21	BRSH <sup>3</sup>	k	Sprung, wenn gleich oder höher (C=0)	Ja:PC=PC+k+1		1/2	1				
MOVRd,RrKopiere RegisterRd=Rr11LDI¹Rd, KLade Konstante in RegisterRd=K11LDSRd, kLade von Adresse k des DatenspeicherRd=(k)22LDRd, XLade Indirekt mittels Register X (R27:R26)Rd=(X)21LDRd, X+Lade indirekt mittels Register X, PostinkrementRd=(X);X=X+121LDRd, -XLade indirekt mittels Register X, PredekrementX=X-1;Rd=(X)21LDRd, YLade Indirekt mittels Register Y (R29:R28)Rd=(Y)21LDRd, Y+Lade indirekt mittels Register Y, PostinkrementRd=(Y);Y=Y+121LDRd, -YLade indirekt mittels Register Y, PredekrementY=Y-1;Rd=(Y)21LDDRd, Y+qLade indirekt mittels Register Y mit OffsetRd=(Y+q)21LDRd, ZLade Indirekt mittels Register Z (R31:R30)Rd=(Z)21	BRL0 <sup>3</sup>	k	Sprung, wenn kleiner (C=1)	Ja:PC=PC+k+1		1/2	1				
LDI¹ Rd, K Lade Konstante in Register Rd=K 1 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 Pd=(Y);Y=Y+1 2 1  LD Rd, -Y Lade indirekt mittels Register Y, Predekrement Pd=(Y);Y=Y+1 2 1  LD 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											
LDI¹ 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  LDD Rd, Z Lade Indirekt mittels Register Z (R31:R30) Rd=(Z) 2 1	MOV	Rd,Rr	Kopiere Register	Rd=Rr		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 LDD Rd, Z Lade Indirekt mittels Register Z (R31:R30) Rd=(Z) 2 1	LDI <sup>1</sup>					1	1				
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  LDD Rd, Z Lade Indirekt mittels Register Z (R31:R30) Rd=(Z) 2 1	LDS		-	Rd=(k)		2	2				
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			` '			1				
LDRd, -XLade indirekt mittels Register X, PredekrementX=X-1;Rd=(X)21LDRd, YLade Indirekt mittels Register Y (R29:R28)Rd=(Y)21LDRd, Y+Lade indirekt mittels Register Y, PostinkrementRd=(Y);Y=Y+121LDRd, -YLade indirekt mittels Register Y, PredekrementY=Y-1;Rd=(Y)21LDDRd, Y+qLade indirekt mittels Register Y mit OffsetRd=(Y+q)21LDRd, ZLade Indirekt mittels Register Z (R31:R30)Rd=(Z)21	LD	Rd, X+	, ,	Rd=(X);X=X+1		2	1				
LDRd, YLade Indirekt mittels Register Y (R29:R28)Rd=(Y)21LDRd, Y+Lade indirekt mittels Register Y, PostinkrementRd=(Y);Y=Y+121LDRd, -YLade indirekt mittels Register Y, PredekrementY=Y-1;Rd=(Y)21LDDRd, Y+qLade indirekt mittels Register Y mit OffsetRd=(Y+q)21LDRd, ZLade Indirekt mittels Register Z (R31:R30)Rd=(Z)21	LD			· '			1				
LDRd, Y+Lade indirekt mittels Register Y, PostinkrementRd=(Y);Y=Y+121LDRd, -YLade indirekt mittels Register Y, PredekrementY=Y-1;Rd=(Y)21LDDRd, Y+qLade indirekt mittels Register Y mit OffsetRd=(Y+q)21LDRd, ZLade Indirekt mittels Register Z (R31:R30)Rd=(Z)21	LD		-	· · · · · ·		2	1				
LDRd, -YLade indirekt mittels Register Y, PredekrementY=Y-1;Rd=(Y)21LDDRd, Y+qLade indirekt mittels Register Y mit OffsetRd=(Y+q)21LDRd, ZLade Indirekt mittels Register Z (R31:R30)Rd=(Z)21	LD		• , ,	· '			1				
LDDRd, Y+qLade indirekt mittels Register Y mit OffsetRd=(Y+q)21LDRd, ZLade Indirekt mittels Register Z (R31:R30)Rd=(Z)21	LD		<u>-</u>	+ ','			1				
LD Rd, Z Lade Indirekt mittels Register Z (R31:R30) Rd=(Z) 2 1	LDD		-	. ,			1				
	LD	•	<u> </u>	· ''			1				
	LD	Rd, Z+	Lade indirekt mittels Register Z, Postinkrement	Rd=(Z);Z=Z+1		2	1				
	LD			· · · · ·			1				
	LDD		-				1				

## AVR Befehlssatz (Auszug)

Mnomic	Operand	Beschreibung	Operation	Flags	Takte	Wörter⁴		
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 <sup>2</sup>	Rr	Schreibt Wert auf den Stack	[Stack]=Rr		2	1		
POP <sup>2</sup>	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		
R0R	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 (R27 ist das Highbyte)

R29:R28 : Y-Register (R29 ist das Highbyte, unterstützt Displacment Adressierung)

R31:R30 : Z-Register (R31 ist das Highbyte, unterstützt Displacment-Adressierung, unterstützt Adressierung des Flash-Speichers)

## Wichtige IO Register

SREG	Statusregister	7:I	6:T	5:H	4:S	3:V	2:N	1:Z	0:C
		Interruptfreig.	Bitspeicher	Halbübertrag	Vorzeichen	2er Übertrag	Negativ	Null	Übertrag
SPL	Stackpointer Low	Untere 8-Bit d	Untere 8-Bit des 16 Bit Stackpointers						
SPH	Stackpointer High	Obere 8-Bit de	Obere 8-Bit des 16 Bit Stackpointers						
DDRx	Portrichtung	Für Port x[A,E	Für Port x[A,B,C,]: 0-Eingang, 1-Ausgang						
P0RTx	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							