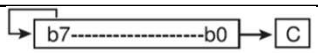
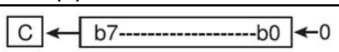
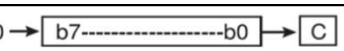
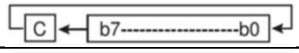
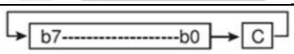
Instruction	Operands	Decription	Operation	Restrictions	Flags
ADC	Rd, Rs	Add w/Carry	Rd ← Rd + Rs + C	0 ≤ d, s ≤ 31	Z,C,N,V,S,H
ADD	Rd, Rs	Add	Rd ← Rd + Rs	0 ≤ d, s ≤ 31	Z,C,N,V,S,H
ADIW	Rd, K	Add Imm to Word	Rd ← Rd + 1:Rd + K	d ={24,26,28,30}, 0 ≤ K ≤ 31	Z,C,N,V,S
AND	Rd, Rs	AND	Rd ← Rd • Rs	0 ≤ d, s ≤ 31	Z,N,V,S
ANDI	Rd, K	AND w/Imm	Rd ← Rd • K	16 ≤ d ≤ 31, 0 ≤ K ≤ 255	Z, N,V,S
ASR	Rd	Arith Shift Right		0 ≤ d ≤ 31	Z,C,N,V
CBR	Rd,K	Complement bits in Reg	Rd ← Rd • ($FFh-K)	16 ≤ d ≤ 31, 0 ≤ K ≤ 255	Z, N,V,S
CLR	Rd	Clear Register	Rd ← Rd ⊕ Rd	0 ≤ d ≤ 31	Z, N,V,S
COM	Rd	Complement	Rd ← $FF - Rd	0 ≤ d ≤ 31	Z,C,N,V,S
DEC	Rd	Decrement	Rd ← Rd - 1	0 ≤ d ≤ 31	Z, N,V,S
EOR	Rd, Rs	Ex-OR	Rd ← Rd ⊕ Rs	0 ≤ d, s ≤ 31	Z, N,V,S
FMUL	Rd,Rs	Frac Mult UU	R1:R0 ← Rd x Rs	16 ≤ d, s ≤ 23	Z,C
FMULS	Rd,Rs	Frac Mult SS	R1:R0 ← Rd x Rs	16 ≤ d, s ≤ 23	Z,C
FMULSU	Rd,Rs	Frac Mult SU	R1:R0 ← Rd x Rs	16 ≤ d, s ≤ 23	Z,C
INC	Rd	Increment	Rd ← Rd + 1	0 ≤ d ≤ 31	Z, N,V,S
LD	Rd, X	Load Indirect	Rd ← (X)	0 ≤ d ≤ 31	
LD	Rd, X+	Load and post-increment X	Rd ← (X), X ← X+1		
LD	Rd, -X	Load and pre-decrement X	X ← X-1, Rd ← (X)		
LD	Rd, Y		Rd ← (Y)		
LD	Rd, Y+	Load and post-increment Y	Rd ← (Y), Y ← Y+1		
LD	Rd, -Y	Load and pre-decrement Y	Y ← Y-1, Rd ← (Y)		
LD	Rd, Z		Rd ← (X)		
LD	Rd, Z+	Load and post-increment Z	Rd ← (Z), Z ← Z+1		
LD	Rd, -Z	Load and pre-decrement Z	Z ← Z-1, Rd ← (Z)		
LDD	Rd, Y+q	Load w/Displacement	Rd ← (Y + q)		
LDD	Rd, Z+q		Rd ← (Z + q)		
LDI	Rd, K	Direct Load Imm	Rd ← K	16 ≤ d ≤ 31, 0 ≤ K ≤ 255	
LDS	Rd, k	Load Data Mem	Rd ← (k)	0 ≤ d ≤ 31, 0 ≤ k ≤ 65535	
LPM	Rd, Z	Indirect Load Program Mem	Rd ← (Z)		
LPM	Rd, Z+				
LPM			R0 ← (Z)	0 ≤ d ≤ 31	
LSL	Rd	Logical Shift Left		0 ≤ d ≤ 31	Z,C,N,V,H
LSR	Rd	Logical Shift Right		0 ≤ d ≤ 31	Z,C,N,V
MOV	Rd, Rs	Copy Register	Rd ← Rs	0 ≤ d ≤ 31	
MOVW	Rd, Rs	Copy Register Pair	Rd+1:Rd ← Rs+1:Rs	d,s = {0,2,...,30}	
MUL	Rd,Rs	Mult UU	R1:R0 ← Rd x Rs (UU)	0 ≤ d, s ≤ 31	Z,C
MULS	Rd,Rs	Mult SS	R1:R0 ← Rd x Rs (SS)	16 ≤ d, s ≤ 31	Z,C
MULSU	Rd,Rs	Mult SU	R1:R0 ← Rd x Rs (SU)	16 ≤ d, s ≤ 23	Z,C
NEG	Rd	Negate	Rd ← $00 - Rd	0 ≤ d ≤ 31	Z,C,N,V,S,H
OR	Rd, Rs	OR	Rd ← Rd v Rs	0 ≤ d, s ≤ 31	Z, N,V,S
ORI	Rd, K	OR w/Imm	Rd ← Rd v K	16 ≤ d ≤ 31, 0 ≤ K ≤ 255	Z, N,V,S
RJMP	k	Relative Jump	PC ← PC + k + 1	k -2K ≤ k < 2K	
ROL	Rd	Rotate Left Thru Carry		0 ≤ d ≤ 31	Z,C,N,V,H
ROR	Rd	Rotate Right Thru Carry		0 ≤ d ≤ 31	Z,C,N,V
SBC	Rd, Rs	Sub w/Carry	Rd ← Rd - Rs - C	0 ≤ d, s ≤ 31	Z,C,N,V,S,H
SBCI	Rd, K	Sub Imm w/Carry	Rd ← Rd - K - C	16 ≤ d ≤ 31, 0 ≤ K ≤ 255	Z,C,N,V,S,H
SBIW	Rd, K	Sub Imm from Word	Rd + 1:Rd ← Rd + 1:Rd - K	d ={24,26,28,30}, 0 ≤ K ≤ 31	Z,C,N,V,S
SBR	Rd,K	Set Bits in Reg	Rd ← Rd v K	16 ≤ d ≤ 31, 0 ≤ K ≤ 255	Z, N,V,S
SER	Rd	Set Register	Rd ← $FF	16 ≤ d ≤ 31	
ST	X, Rs	Store Indirect	(X) ← Rs	0 ≤ d ≤ 31	
ST	Z, Rs		(Z) ← Rs		

ST	Z+, Rs		(Z) ← Rs, Z ← Z+1		
ST	-X, Rs		X ← X-1, (X) ← Rd		
ST	-Z, Rs		Z ← Z-1, (Z) ← Rd		
ST	Y, Rs		(Y) ← Rs		
ST	X+, Rs		(X) ← Rs, X ← X+1		
ST	Y+, Rs		(Y) ← Rs, Y ← Y+1		
ST	-Y, Rs		Y ← Y-1, (Y) ← Rd		
STD	Y+q,Rs	Store w/Displacement	(Y+q) ← Rs	0 ≤ s ≤ 31, 0 ≤ q ≤ 63	
STD	Z+q,Rs		(Z+q) ← Rs		
STS	k, Rs	Store Direct	(k) ← Rd	0 ≤ d ≤ 31, 0 ≤ k ≤ 65535	
SUB	Rd, Rs	Sub	Rd ← Rd - Rs	0 ≤ d, s ≤ 31	Z,C,N,V,S,H
SUBI	Rd, K	Sub Imm	Rd ← Rd - K	16 ≤ d ≤ 31, 0 ≤ K ≤ 255	Z,C,N,V,S,H
SWAP	Rd	Swap Nibbles	Rd(3..0) ↔ Rd(7..4)	0 ≤ d ≤ 31	

Instruction	**Operands**	**Decription**	**Operation**	**Restrictions**	**Flags**
BSET	s	Flag Set	SREG(s) ← 1		SREG(s)
BCLR	s	Flag Clear	SREG(s) ← 0		SREG(s)
SEC		Set Carry	C ← 1		
CLC		Clear Carry	C ← 0		
SEN		Set Negative Flag	N ← 1		N
CLN		Clear Negative Flag	N ← 0		N
SEZ		Set Zero Flag	Z ← 1		
CLZ		Clear Zero Flag	Z ← 0		Z
SES		Set Signed Test	S ← 1		S
CLS		Clear Signed Test	S ← 0		S
SEV		Set Overflow (V)	V ← 1		V
CLV		Clear Overflow (V)	V ← 0		V
RJMP	k	Relative Jump	PC ← PC + k + 1	2K ≤ k < 2K	
CALL	k	Call Subroutine	PC ← k; SP = SP - 3	0 ≤ k < 4M	
RET		Subroutine Return	PC ← (SP); SP = SP + 3		
CPSE	Rd,Rr	Compare,skip if Equal	(Rd == Rr): PC ← PC + 2	0 ≤ d,r ≤ 31	
CP	Rd,Rr	Compare	Rd - Rr	0 ≤ d,r ≤ 31	Z,C,N,V,S,H
CPC	Rd,Rr	Compare w/Carry	Rd - Rr - C	0 ≤ d,r ≤ 31	Z,C,N,V,S,H
CPI	Rd,K	Compare w/Imm	Rd - K	0 ≤ d,r ≤ 31; 0 ≤ K ≤ 255	Z,C,N,V,S,H
TST	Rd	Test if Rd = 0	Rd • Rd	0 ≤ d ≤ 31	Z, N,V,S
BRBS	s, k	Branch if Status = s	(SREG(s) == 1): PC ← PC + k + 1	-64 ≤ k ≤ +63	
BRBC	s, k	Branch if Status = not(s)	(SREG(s) ==0): PC ← PC + k + 1	-64 ≤ k ≤ +63	
BREQ	k	Branch if Equal	(Z == 1): PC ← PC + k + 1	-64 ≤ k ≤ +63	
BRNE	k	Branch if Not Equal	(Z = 0): PC ← PC + k + 1	-64 ≤ k ≤ +63	
BRCS	k	Branch if Carry Set	(C = 1): PC ← PC + k + 1	-64 ≤ k ≤ +63	
BRCC	k	Branch if Carry Cleared	(C == 0): PC ← PC + k + 1	-64 ≤ k ≤ +63	
BRSH	k	Branch if Same or Higher	(C == 0): PC ← PC + k + 1	-64 ≤ k ≤ +63	
BRLO	k	Branch if Lower	(C == 1): PC ← PC + k + 1	unsigned; -64 ≤ k ≤ +63	
BRMI	k	Branch if Minus	(N == 1) : PC ← PC + k + 1	unsigned; -64 ≤ k ≤ +63	
BRPL	k	Branch if Plus	(N == 0) : PC ← PC + k + 1	-64 ≤ k ≤ +63	
BRGE	k	Branch if Greater or Equal	(N ⊕ V== 0): PC ← PC + k + 1	signed; -64 ≤ k ≤ +63	
BRLT	k	Branch if Less Than	(N ⊕ V== 1): PC ← PC + k + 1	signed; -64 ≤ k ≤ +63	
BRVS	k	Branch if V Flag is Set	(V == 1): PC ← PC + k + 1	-64 ≤ k ≤ +63	
BRVC	k	Branch if V Flag is Cleared	(V == 0): PC ← PC + k + 1	-64 ≤ k ≤ +63	
PUSH	Rr	Push Register on Stack	(SP) ← Rr; SP--	0 ≤ d ≤ 31	
POP	Rd	Pop Register from Stack	Rd ← (SP); SP++	0 ≤ d ≤ 31	