Arithmetic and Logic Instructions

Mnemonic	Operands	Description	Operation	Flags	Cycles
ADD	Rd, Rr	Add without Carry	Rd = Rd + Rr	Z,C,N,V,H,S	1
ADC	Rd, Rr	Add with Carry	Rd = Rd + Rr + C	Z,C,N,V,H,S	1
ADIW	Rdl, K6	Add Immediate to Word	Rdh:Rdl = Rdh:Rdl + K6	Z,C,N,V,S	2
SUB	Rd, Rr	Subtract without Carry	Rd = Rd - Rr	Z,C,N,V,H,S	1
SUBI	Rd, K8	Subtract Immediate	Rd = Rd - K8	Z,C,N,V,H,S	1
SBC	Rd, Rr	Subtract with Carry	Rd = Rd - Rr - C	Z,C,N,V,H,S	1
SBCI	Rd, K8	Subtract with Carry Immedtiate	Rd = Rd - K8 - C	Z,C,N,V,H,S	1
SBIW	Rdl, K6	Subtract Immediate from Word	Rdh:Rdl = Rdh:Rdl - K6	Z,C,N,V,S	2
AND	Rd, Rr	Logical AND	Rd = Rd · Rr	Z,N,V,S	1
ANDI	Rd, K8	Logical AND with Immediate	Rd = Rd · K8	Z,N,V,S	1
OR	Rd, Rr	Logical OR	Rd = Rd | Rr	Z,N,V,S	1
ORI	Rd, K8	Logical OR with Immediate	Rd = Rd | K8	Z,N,V,S	1
EOR	Rd, Rr	Logical Exclusive OR	Rd = Rd EOR Rr	Z,N,V,S	1
COM	Rd	One's Complement	Rd = $FF - Rd	Z,C,N,V,S	1
NEG	Rd	Two's Complement	Rd = $00 - Rd	Z,C,N,V,H,S	1
SBR	Rd, K8	Set Bit(s) in Register	Rd = Rd | K8	Z,C,N,V,S	1
CBR	Rd, K8	Clear Bit(s) in Register	Rd = Rd · ($FF - K8)	Z,C,N,V,S	1
INC	Rd	Increment Register	Rd = Rd + 1	Z,N,V,S	1
DEC	Rd	Decrement Register	Rd = Rd - 1	Z,N,V,S	1
TST	Rd	Test for Zero or Negative	Rd = Rd · Rd	Z,C,N,V,S	1
CLR	Rd	Clear Register	Rd = 0	Z,C,N,V,S	1
SER	Rd	Set Register	Rd = $FF	None	1
MUL	Rd, Rr	Multiply Unsigned	R1:R0 = Rd * Rr	Z,C	2
MULS	Rd, Rr	Multiply Signed	R1:R0 = Rd * Rr	Z,C	2
MULSU	Rd, Rr	Multiply Signed with Unsigned	R1:R0 = Rd * Rr	Z,C	2
FMUL	Rd, Rr	Fractional Multiply Unsigned	R1:R0 = (Rd * Rr) << 1	Z,C	2
FMULS	Rd, Rr	Fractional Multiply Signed	R1:R0 = (Rd * Rr) << 1	Z,C	2
FMULSU	Rd, Rr	Fractional Multiply Signed with	R1:R0 = (Rd * Rr) << 1	Z,C	2
DES	K8	Data Encryption	If (H=0) then R15:R0 = Encrypt(R15:R0, K8)		

Branch Instructions

Mnemonic	Operands	Description	Operation	Flags	Cycles
RJMP	K	Relative Jump	PC = PC + K + 1	None	2
IJMP	None	Indirect Jump to (Z)	PC = Z	None	2
EIJMP	None	Extended Indirect Jump (Z)	STACK = PC + 1, PC(15:0) = Z, PC(21:16) = EIND	None	2
JMP	K	Jump	PC = K	None	3
RCALL	K	Relative Call Subroutine	STACK = PC + 1, PC = PC + K + 1	None	3/4*
ICALL	None	Indirect Call to (Z)	STACK = PC + 1, PC = Z	None	3/4*
EICALL	None	Extended Indirect Call to (Z)	STACK = PC + 1, PC(15:0) = Z, PC(21:16) = EIND	None	4*
CALL	K	Call Subroutine	STACK = PC + 2, PC = K	None	4/5*
RET	None	Subroutine Return	PC = STACK	None	4/5*
RETI	None	Interrupt Return	PC = STACK	I	4/5*
CPSE	Rd, Rr	Compare, Skip if equal	if (Rd == Rr) PC = PC + 2 or 3	None	1/2/3
CP	Rd, Rr	Compare	Rd - Rr	Z,C,N,V,H,S	1
CPC	Rd, Rr	Compare with Carry	Rd - Rr - C	Z,C,N,V,H,S	1
CPI	Rd, K8	Compare with Immediate	Rd - K	Z,C,N,V,H,S	1
SBRC	Rr, b	Skip if bit in register cleared	if(Rr(b) == 0) PC = PC + 2 or 3	None	1/2/3
SBRS	Rr, b	Skip if bit in register set	if(Rr(b) == 1) PC = PC + 2 or 3	None	1/2/3
SBIC	P, b	Skip if bit in I/O register cleared	if(I/O(P, b) == 0) PC = PC + 2 or 3	None	1/2/3
SBIS	P, b	Skip if bit in I/O register set	if(I/O(P, b) == 1) PC = PC + 2 or 3	None	1/2/3
BRBC	s, K	Branch if Status flag cleared	if(SREG(s) == 0) PC = PC + K + 1	None	1/2
BRBS	s, K	Branch if Status flag set	if(SREG(s) == 1) PC = PC + K + 1	None	1/2
BREQ	K	Branch if equal	If(Z == 1) PC = PC + K + 1	None	1/2
BRNE	K	Branch if not equal	If(Z == 0) PC = PC + K + 1	None	1/2
BRCS	K	Branch if Carry set	if(C == 1) PC = PC + K + 1	None	1/2
BRCC	K	Branch if Carry cleared	if(C == 0) PC = PC + K + 1	None	1/2
BRSH	K	Branch if same or higher	if(C == 0) PC = PC + K + 1	None	1/2
BRLO	K	Branch if lower	if(C == 1) PC = PC + K + 1	None	1/2
BRMI	K	Branch if minus	if(N == 1) PC = PC + K + 1	None	1/2
BRPL	K	Branch if plus	if(N == 0) PC = PC + K + 1	None	1/2
BRGE	K	Branch if greater than or equal (signed)	if(S == 0) PC = PC + K + 1	None	1/2
BRLT	K	Branch if less than (signed)	if(S == 1) PC = PC + K + 1	None	1/2
BRHS	K	Branch if half Carry flag set	if(H == 1) PC = PC + K + 1	None	1/2
BRHC	K	Branch if half Carry flag cleared	if(H == 0) PC = PC + K + 1	None	1/2
BRTS	K	Branch if T flag set	if(T == 1) PC = PC + K + 1	None	1/2
BRTC	K	Branch if T flag cleared	if(T == 0) PC = PC + K + 1	None	1/2
BRVS	K	Branch if overflow flag set	if(V == 1) PC = PC + K + 1	None	1/2
BRVC	K	Branch if overflow flag cleared	if(V == 0) PC = PC + K + 1	None	1/2
BRIE	K	Branch if interrupt enabled	if(I == 1) PC = PC + K + 1	None	1/2
BRID	K	Branch if interrupt disabled	if(I == 0) PC = PC + K + 1	None	1/2

(CP, CPI etc.)

Signed:

Rd > Rr	BRLT	Rr, Rd
Rd >= Rr	BRGE	Rd, Rr
Rd = Rr	BREQ	Rd, Rr
Rd <= Rr	BRGE	Rr, Rd
Rd < Rr	BRLT	Rd, Rr

Unsigned:

Rd > Rr	BRLO	Rr, Rd
Rd >= Rr	BRCC	Rd, Rr
Rd = Rr	BREQ	Rd, Rr
Rd <= Rr	BRSH	Rr, Rd
Rd < Rr	BRLO	Rd, Rr

Data Transfer Instructions

Mnemonic	Operands	Description	Operation	Flags	Cycles
MOV	Rd, Rr	Copy register	Rd = Rr	None	1
MOVW	Rd, Rr	Copy register pair	Rd + 1:Rd = Rr + 1:Rr, rd even	None	1
LDI	Rd, K8	Load Immediate	Rd = K8	None	1
LDS	Rd, K	Load Direct	Rd = (K)	None	2*
LD	Rd, X	Load Indirect	Rd = (X)	None	2*
LD	Rd, X+	Load Indirect and Post- Increment	Rd = (X), X = X + 1	None	2*
LD	Rd, -X	Load Indirect and Pre- Decrement	X = X - 1, Rd = (X)	None	2*
LD	Rd, Y	Load Indirect	Rd = (Y)	None	2*
LD	Rd, Y+	Load Indirect and Post- Increment	Rd = (Y), Y = Y + 1	None	2*
LD	Rd, -Y	Load Indirect and Pre- Decrement	Y = Y - 1, Rd = (Y)	None	2*
LDD	Rd, Y + K6	Load Indirect with displacement	Rd = (Y + K6)	None	2*
LD	Rd, Z	Load Indirect	Rd = (Z)	None	2*
LD	Rd, Z+	Load Indirect and Post- Increment	Rd = (Z), Z = Z + 1	None	2*
LD	Rd, -Z	Load Indirect and Pre- Decrement	Z = Z - 1, Rd = (Z)	None	2*
LDD	Rd, Z + K6	Load Indirect with displacement	Rd = (Z + K6)	None	2*
STS	K, Rr	Store Direct	(K) = Rr	None	2*
ST	X, Rr	Store Indirect	(X) = Rr	None	2*
ST	X+, Rr	Store Indirect and Post- Increment	(X) = Rr, X = X + 1	None	2*
ST	-X, Rr	Store Indirect and Pre- Decrement	X = X - 1, (X) = Rr	None	2*
ST	Y, Rr	Store Indirect	(Y) = Rr	None	2*
ST	Y+, Rr	Store Indirect and Post- Increment	(Y) = Rr, Y = Y + 1	None	2
ST	-Y, Rr	Store Indirect and Pre- Decrement	Y = Y - 1, (Y) = Rr	None	2
ST	Y + K6, Rr	Store Indirect with displacement	(Y + K6) = Rr	None	2
ST	Z, Rr	Store Indirect	(Z) = Rr	None	2
ST	Z+, Rr	Store Indirect and Post- Increment	(Z) = Rr, Z = Z + 1	None	2
ST	-Z, Rr	Store Indirect and Pre- Decrement	Z = Z - 1, (Z) = Rr	None	2
ST	Z + K6, Rr	Store Indirect with displacement	(Z + K6) = Rr	None	2
LPM	None	Load Program Memory	R0 = (Z)	None	3
LPM	Rd, Z	Load Program Memory	Rd = (Z)	None	3
LPM	Rd, Z+	Load Program Memory and Post- Increment	Rd = (Z), Z = Z + 1	None	3
ELPM	None	Extended Load Program Memory	R0 = (RAMPZ:Z)	None	3
ELPM	Rd, Z	Extended Load Program Memory	Rd = (RAMPZ:Z)	None	3
ELPM	Rd, Z+	Extended Load Program Memory and Post Increment	Rd = (RAMPZ:Z), Z = Z + 1	None	3
SPM	None	Store Program Memory	(Z) = R1:R0	None	-
ESPM	None	Extended Store Program Memory	(RAMPZ:Z) = R1:R0	None	-
IN	Rd, P	In Port	Rd = P	None	1
OUT	P, Rr	Out Port	P = Rr	None	1
PUSH	Rr	Push register on Stack	STACK = Rr	None	2
POP	Rd	Pop register from Stack	Rd = STACK	None	2

Bit and Bit- Test Instructions

Mnemonic	Operands	Description	Operation	Flags	Cycles
LSL	Rd	Logical shift left	Rd(n + 1) = Rd(n), Rd(0) = 0, C = Rd(7)	Z,C,N,V,H,S	1
LSR	Rd	Logical shift right	Rd(n) = Rd(n + 1), Rd(7) = 0, C = Rd(0)	Z,C,N,V,S	1
ROL	Rd	Rotate left through Carry	Rd(0) = C, Rd(n + 1) = Rd(n), C = Rd(7)	Z,C,N,V,H,S	1
ROR	Rd	Rotate right through Carry	Rd(7) = C, Rd(n) = Rd(n + 1), C = Rd(0)	Z,C,N,V,S	1
ASR	Rd	Arithmetic shift right	Rd(n) = Rd(n + 1), n = 0,...,6	Z,C,N,V,S	1
SWAP	Rd	Swap nibbles	Rd(3..0) = Rd(7..4), Rd(7..4) = Rd(3..0)	None	1
BSET	s	Set flag	SREG(s) = 1	SREG(s)	1
BCLR	s	Clear flag	SREG(s) = 0	SREG(s)	1
SBI	P, b	Set bit in I/O register	I/O(P, b) = 1	None	2
CBI	P, b	Clear bit in I/O register	I/O(P, b) = 0	None	2
BST	Rr, b	Bit store from register to T	T = Rr(b)	T	1
BLD	Rd, b	Bit load from register to T	Rd(b) = T	None	1
SEC	None	Set Carry flag	C = 1	C	1
CLC	None	Clear Carry flag	C = 0	C	1
SEN	None	Set negative flag	N = 1	N	1
CLN	None	Clear negative flag	N = 0	N	1
SEZ	None	Set Zero flag	Z = 1	Z	1
CLZ	None	Clear Zero flag	Z = 0	Z	1
SEI	None	Set interrupt flag	I = 1	I	1
CLI	None	Clear interrupt flag	I = 0	I	1
SES	None	Set signed flag	S = 1	S	1
CLN	None	Clear signed flag	S = 0	S	1
SEV	None	Set overflow flag	V = 1	V	1
CLV	None	Clear overflow flag	V = 0	V	1
SET	None	Set T flag	T = 1	T	1
CLT	None	Clear T flag	T = 0	T	1
SEH	None	Set half Carry flag	H = 1	H	1
CLH	None	Clear half Carry flag	H = 0	H	1
NOP	None	No operation	None	None	1
SLEEP	None	Sleep	See instruction manual	None	1
WDR	None	Watchdog Reset	See instruction manual	None	1

Rd: Destination register in the register file
Rr: Source register in the register file
b: Constant (0- 7), can be a constant expression
s: Constant (0- 7), can be a constant expression
P: Constant (0- 31/63), can be a constant expression

K: Constant, value range depending on instruction.
K6: Constant (0 - 63), can be a constant expression
K8: Constant (0 - 255), can be a constant expression
Rdl: R24, R26, R28, R30. For ADIW and SBIW instructions
X,Y,Z: Indirect address registers (X = R27:R26, Y = R29:R28, Z = R31:R30)