Instruction Set Summary ATMega328P(B)

Mnemonics	Operands	Description	Operation	Flags	#Clocks
ADD	Rd, Rr	Add two registers	Rd ← Rd + Rr	Z,C,N,VH	1
ADC	Rd, Rr	Add with carry two registers	Rd ← Rd + Rr + C	Z,C,N,V,H	1
ADIW	Rdl, K	Add immediate to word	Rdh: Rdl ← Rdh: Rdl + K	Z,C,N,V,S	2
AND	Rd, Rr	Logical AND registers	Rd ← Rd × Rr	Z,N,V	1
ANDI	Rd, K	Logical AND register and constant	Rd ← Rd × K	Z,N,V	1
ASR	Rd	Arithmetic shift right	Rd (n) ← Rd(n+1), n=0..6	Z,C,N,V	1
BCLR	s	Flag clear	SREG (s) ← 0	SREG (s)	1
BLD	Rd, b	Bit load from T to register	Rd (b) ← T	None	1
BRBC	s, k	Branch if status flag cleared	if (SREG(s) = 0) then PC←PC + k + 1	None	1/2
BRBS	s, k	Branch if status flag set	if (SREG(s) = 1) then PC←PC + k + 1	None	1/2
BRCC	k	Branch if carry cleared	if (C = 0) then PC ← PC + k + 1	None	1/2
BRCS	k	Branch if carry set	if (C = 1) then PC ← PC + k + 1	None	1/2
BREAK		Break	For on-chip debug only	None	N/A
BREQ	k	Branch if equal	if (Z = 1) then PC ← PC + k + 1	None	1/2
BRGE	k	Branch if greater or equal, signed	if (N ⊕ V= 0) then PC ← PC + k + 1	None	1/2
BRHC	k	Branch if half carry flag cleared	if (H = 0) then PC ← PC + k + 1	None	1/2
BRHS	k	Branch if half carry flag set	if (H = 1) then PC ← PC + k + 1	None	1/2
BRID	k	Branch if interrupt disabled	if (I = 0) then PC ← PC + k + 1	None	1/2
BRIE	k	Branch if interrupt enabled	if (I = 1) then PC ← PC + k + 1	None	1/2
BRLO	k	Branch if lower	if (C = 1) then PC ← PC + k + 1	None	1/2
BRLT	k	Branch if less than zero, signed	if (N ⊕ V= 1) then PC ← PC + k + 1	None	1/2
BRMI	k	Branch if minus	if (N = 1) then PC ← PC + k + 1	None	1/2
BRNE	k	Branch if not equal	if (Z = 0) then PC ← PC + k + 1	None	1/2
BRPL	k	Branch if plus	if (N = 0) then PC ← PC + k + 1	None	1/2
BRSH	k	Branch if same or higher	if (C = 0) then PC ← PC + k + 1	None	1/2
BRTC	k	Branch if T flag cleared	if (T = 0) then PC ← PC + k + 1	None	1/2
BRTS	k	Branch if T flag set	if (T = 1) then PC ← PC + k + 1	None	1/2
BRVC	k	Branch if overflow flag is cleared	if (V = 0) then PC ← PC + k + 1	None	1/2
BRVS	k	Branch if overflow flag is set	if (V = 1) then PC ← PC + k + 1	None	1/2
BSET	s	Flag set	SREG (s) ← 1	SREG (s)	1
BST	Rr, b	Bit store from register to T	T ← Rr (b)	T	1
CALL	k	Direct subroutine call	PC ← k	None	4
CBI	P, b	Clear bit in I/O register	I/O (P, b) ← 0	None	2
CBR	Rd, K	Clear bit(s) in register	Rd ← Rd × (0xFF – K)	Z,N,V	1
CLC		Clear carry	C ← 0	C	1
CLH		Clear half carry flag in SREG	H ← 0	H	1
CLI		Global interrupt disable	I ← 0	I	1
CLN		Clear negative flag	N ← 0	N	1
CLR	Rd	Clear register	Rd ← Rd ⊕ Rd	Z,N,V	1
CLS		Clear signed test flag	S ← 0	S	1
CLT		Clear T in SREG	T ← 0	T	1
CLV		Clear twos complement overflow	V ← 0	V	1
CLZ		Clear zero flag	Z ← 0	Z	1
COM	Rd	One's complement	Rd ← 0xFF – Rd	Z,C,N,V	1
CP	Rd, Rr	Compare	Rd – Rr	Z,N,V,C,H	1
CPC	Rd, Rr	Compare with carry	Rd – Rr – C	Z, N,V,C,H	1
CPI	Rd, K	Compare register with immediate	Rd – K	Z, N,V,C,H	1
CPSE	Rd, Rr	Compare, skip if equal	if (Rd = Rr) PC ← PC + 2 or 3	None	1/2/3
DEC	Rd	Decrement	Rd ← Rd – 1	Z,N,V	1

Instruction Set Summary ATMega328P(B)

Mnemonics	Operands	Description	Operation	Flags	#Clocks
EOR	Rd, Rr	Exclusive OR registers	Rd ← Rd ⊕ Rr	Z,N,V	1
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 unsigned	R1:R0 ← (Rd × Rr) << 1	Z,C	2
ICALL		Indirect call to (Z)	PC ← Z	None	3
IJMP		Indirect jump to (Z)	PC ← Z	None	2
IN	Rd, P	In port	Rd ← P	None	1
INC	Rd	Increment	Rd ← Rd + 1	Z,N,V	1
JMP	k	Direct jump	PC ← k	None	3
LD	Rd, X	Load indirect	Rd ← (X)	None	2
LD	Rd, X+	Load indirect and post-inc.	Rd ← (X), X ← X + 1	None	2
LD	Rd, – X	Load indirect and pre-dec.	X ← X – 1, Rd ← (X)	None	2
LD	Rd, Y	Load indirect	Rd ← (Y)	None	2
LD	Rd, Y+	Load indirect and post-inc.	Rd ← (Y), Y ← Y + 1	None	2
LD	Rd, – Y	Load indirect and pre-dec.	Y ← Y – 1, Rd ← (Y)	None	2
LD	Rd, Z	Load indirect	Rd ← (Z)	None	2
LD	Rd, Z+	Load indirect and post-inc.	Rd ← (Z), Z ← Z+1	None	2
LD	Rd, –Z	Load indirect and pre-dec.	Z ← Z – 1, Rd ← (Z)	None	2
LDD	Rd, Y+ q	Load indirect with displacement	Rd ← (Y + q)	None	2
LDD	Rd, Z+ q	Load indirect with displacement	Rd ← (Z + q)	None	2
LDI	Rd, K	Load immediate	Rd ← K	None	1
LDS	Rd, k	Load direct from SRAM	Rd ← (k)	None	2
LPM		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-inc	Rd ← (Z), Z ← Z+1	None	3
LSL	Rd	Logical shift left	Rd(n+1) ← Rd (n), Rd(0) ← 0	Z,C,N,V	1
LSR	Rd	Logical shift right	Rd (n) ← Rd(n+1), Rd(7) ← 0	Z,C,N,V	1
MOV	Rd, Rr	Move between registers	Rd ← Rr	None	1
MOVW	Rd, Rr	Copy register word	Rd+1:Rd ← Rr+1:Rr	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
NEG	Rd	Two’s complement	Rd ← 0x00 – Rd	Z,C,N,V,H	1
NOP		No operation		None	1
OR	Rd, Rr	Logical OR registers	Rd ← Rd v Rr	Z,N,V	1
ORI	Rd, K	Logical OR register and constant	Rd ← Rd v K	Z,N,V	1
OUT	P, Rr	Out port	P ← Rr	None	1
POP	Rd	Pop register from stack	Rd ← STACK	None	2
PUSH	Rr	Push register on stack	STACK ← Rr	None	2
RCALL	k	Relative subroutine call	PC ← PC + k + 1	None	3
RET		Subroutine return	PC ← STACK	None	4
RETI		Interrupt return	PC ← STACK	I	4
RJMP	k	Relative jump	PC ← PC + k + 1	None	2
ROL	Rd	Rotate left through carry	Rd(0) ← C,Rd(n+1) ← Rd (n), C ← Rd(7)	Z,C,N,V	1
ROR	Rd	Rotate right through carry	Rd(7) ← C, Rd (n) ← Rd(n+1),C ← Rd(0)	Z,C,N,V	1
SBC	Rd, Rr	Subtract with carry two registers	Rd ← Rd – Rr – C	Z,C,N,V,H	1
SBCI	Rd, K	Subtract with carry constant from reg.	Rd ← Rd – K – C	Z,C,N,V,H	1
SBI	P, b	Set bit in I/O register	I/O (P, b) ← 1	None	2
SBIC	P, b	Skip if bit in I/O register cleared	if (P(b)=0) PC ← PC + 2 or 3	None	1/2/3

Instruction Set Summary ATMega328P(B)

Mnemonics	Operands	Description	Operation	Flags	#Clocks
SBIS	P, b	Skip if bit in I/O register is set	if (P(b)=1) PC ← PC + 2 or 3	None	1/2/3
SBIW	Rdl, K	Subtract immediate from word	Rdh: Rdl ← Rdh: Rdl – K	Z,C,N,V,S	2
SBR	Rd, K	Set bit(s) in register	Rd ← Rd v K	Z,N,V	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 is set	if (Rr(b)=1) PC ← PC + 2 or 3	None	1/2/3
SEC		Set carry	C ← 1	C	1
SEH		Set half carry flag in SREG	H ← 1	H	1
SEI		Global interrupt enable	I ← 1	I	1
SEN		Set negative flag	N ← 1	N	1
SER	Rd	Set register	Rd ← 0xFF	None	1
SES		Set signed test flag	S ← 1	S	1
SET		Set T in SREG	T ← 1	T	1
SEV		Set twos complement overflow.	V ← 1	V	1
SEZ		Set zero flag	Z ← 1	Z	1
SLEEP		Sleep	(see specific descr. for sleep function)	None	1
SPM		Store program memory	(Z) ← R1:R0	None	–
ST	X, Rr	Store indirect	(X) ← Rr	None	2
ST	X+, Rr	Store indirect and post-inc.	(X) ← Rr, X ← X + 1	None	2
ST	– X, Rr	Store indirect and pre-dec.	X ← X – 1, (X) ← Rr	None	2
ST	Y, Rr	Store indirect	(Y) ← Rr	None	2
ST	Y+, Rr	Store indirect and post-inc.	(Y) ← Rr, Y ← Y + 1	None	2
ST	– Y, Rr	Store indirect and pre-dec.	Y ← Y – 1, (Y) ← Rr	None	2
ST	Z, Rr	Store indirect	(Z) ← Rr	None	2
ST	Z +, Rr	Store indirect and post-inc.	(Z) ← Rr, Z ← Z + 1	None	2
ST	–Z, Rr	Store indirect and pre-dec.	Z ← Z – 1, (Z) ← Rr	None	2
STD	Y+ q, Rr	Store indirect with displacement	(Y + q) ← Rr	None	2
STD	Z + q, Rr	Store indirect with displacement	(Z + q) ← Rr	None	2
STS	k, Rr	Store direct to SRAM	(k) ← Rr	None	2
SUB	Rd, Rr	Subtract two registers	Rd ← Rd – Rr	Z,C,N,V,H	1
SUBI	Rd, K	Subtract constant from register	Rd ← Rd – K	Z,C,N,V,H	1
SWAP	Rd	Swap nibbles	Rd(3..0) ← Rd(7..4),Rd(7..4) ← Rd(3..0)	None	1
TST	Rd	Test for zero or minus	Rd ← Rd × Rd	Z,N,V	1
WDR		Watchdog reset	(see specific descr. for WDR/timer)	None	1