AVR Instruction Sets

Data Transfer Instructions

Mnemonics	Operands	Description	Operation	Flags
MOV	Rd, Rr	Move Between Registers	Rd ← Rr	None
MOVW	Rd, Rr	Copy Register Word	Rd + 1:Rd ← Rr + 1:Rr	None
LDI	Rd, K	Load Immediate	Rd ← K	None
LD	Rd, X	Load Indirect	Rd ← (X)	None
LD	Rd, X+	Load Indirect and Post-Inc.	Rd ← (X), X ← X + 1	None
LD	Rd, –X	Load Indirect and Pre-Dec.	X ← X – 1, Rd ← (X)	None
LD	Rd, Y	Load Indirect	Rd ← (Y)	None
LD	Rd, Y+	Load Indirect and Post-Inc.	Rd ← (Y), Y ← Y + 1	None
LD	Rd, –Y	Load Indirect and Pre-Dec.	Y ← Y – 1, Rd ← (Y)	None
LDD	Rd,Y+q	Load Indirect with Displacement	Rd ← (Y + q)	None
LD	Rd, Z	Load Indirect	Rd ← (Z)	None
LD	Rd, Z+	Load Indirect and Post-Inc.	Rd ← (Z), Z ← Z+1	None
LD	Rd, –Z	Load Indirect and Pre-Dec.	Z ← Z – 1, Rd ← (Z)	None
LDD	Rd, Z + q	Load Indirect with Displacement	Rd ← (Z + q)	None
LDS	Rd, k	Load Direct from SRAM	Rd ← (k)	None
ST	X, Rr	Store Indirect	(X) ← Rr	None
ST	X+, Rr	Store Indirect and Post-Inc.	(X) ← Rr, X ← X + 1	None
ST	–X, Rr	Store Indirect and Pre-Dec.	X ← X – 1, (X) ← Rr	None
ST	Y, Rr	Store Indirect	(Y) ← Rr	None
ST	Y+, Rr	Store Indirect and Post-Inc.	(Y) ← Rr, Y ← Y + 1	None
ST	–Y, Rr	Store Indirect and Pre-Dec.	Y ← Y – 1, (Y) ← Rr	None
STD	Y + q, Rr	Store Indirect with Displacement	(Y + q) ← Rr	None
ST	Z, Rr	Store Indirect	(Z) ← Rr	None
ST	Z+, Rr	Store Indirect and Post-Inc.	(Z) ← Rr, Z ← Z + 1	None
ST	–Z, Rr	Store Indirect and Pre-Dec.	Z ← Z – 1, (Z) ← Rr	None
STD	Z + q, Rr	Store Indirect with Displacement	(Z + q) ← Rr	None
STS	k, Rr	Store Direct to SRAM	(k) ← Rr	None
LPM		Load Program Memory	R0 ← (Z)	None
LPM	Rd, Z	Load Program Memory	Rd ← (Z)	None
LPM	Rd, Z+	Load Program Memory and Post-Inc.	Rd ← (Z), Z ← Z+1	None
SPM		Store Program Memory	(Z) ← R1:R0	None
IN	Rd, P	In Port	Rd ← P	None
OUT	P, Rr	Out Port	P ← Rr	None
PUSH	Rr	Push Register on Stack	Stack ← Rr	None
POP	Rd	Pop Register from Stack	Rd ← Stack	None

Branch Instructions

Mnem.	Oper.	Description	Operation	Flags
RJMP	k	Relative Jump	PC ← PC + k + 1	None
IJMP		Indirect Jump to (Z)	PC ← Z	None
JMP	k	Direct Jump	PC ← k	None
RCALL	k	Relative Subroutine Call	PC ← PC + k + 1	None
ICALL		Indirect Call to (Z)	PC ← Z	None
CALL	k	Direct Subroutine Call	PC ← k	None
RET		Subroutine Return	PC ← Stack	None
RETI		Interrupt Return	PC ← Stack	I
CPSE	Rd,Rr	Compare, Skip if Equal	if (Rd = Rr) PC ← PC + 2 or 3	None
CP	Rd,Rr	Compare	Rd – Rr	Z,N,V,C,H
CPC	Rd,Rr	Compare with Carry	Rd – Rr – C	Z,N,V,C,H
CPI	Rd,K	Compare Register with Immediate	Rd – K	Z,N,V,C,H
SBRC	Rr, b	Skip if Bit in Register Cleared	if (Rr(b)=0) PC ← PC + 2 or 3	None
SBRS	Rr, b	Skip if Bit in Register is Set	if (Rr(b)=1) PC ← PC + 2 or 3	None
SBIC	P, b	Skip if Bit in I/O Register Cleared	if (P(b)=0) PC ← PC + 2 or 3	None
SBIS	P, b	Skip if Bit in I/O Register is Set	if (P(b)=1) PC ← PC + 2 or 3	None
BRBS	s, k	Branch if Status Flag Set	if (SREG(s)=1) then PC←PC+k+1	None
BRBC	s, k	Branch if Status Flag Cleared	if (SREG(s)=0) then PC←PC+k+1	None
BREQ	k	Branch if Equal	if (Z = 1) then PC ← PC + k + 1	None
BRNE	k	Branch if Not Equal	if (Z = 0) then PC ← PC + k + 1	None
BRCS	k	Branch if Carry Set	if (C = 1) then PC ← PC + k + 1	None
BRCC	k	Branch if Carry Cleared	if (C = 0) then PC ← PC + k + 1	None
BRSH	k	Branch if Same or Higher	if (C = 0) then PC ← PC + k + 1	None
BRLO	k	Branch if Lower	if (C = 1) then PC ← PC + k + 1	None
BRMI	k	Branch if Minus	if (N = 1) then PC ← PC + k + 1	None
BRPL	k	Branch if Plus	if (N = 0) then PC ← PC + k + 1	None
BRGE	k	Branch if Greater or Equal,Signed	if (N and V= 0) then PC←PC + k +1	None
BRLT	k	Branch if Less Than Zero, Signed	if (N and V= 1) then PC←PC + k +1	None
BRHS	k	Branch if Half Carry Flag Set	if (H = 1) then PC ← PC + k + 1	None
BRHC	k	Branch if Half Carry Flag Cleared	if (H = 0) then PC ← PC + k + 1	None
BRTS	k	Branch if T Flag Set	if (T = 1) then PC ← PC + k + 1	None
BRTC	k	Branch if T Flag Cleared	if (T = 0) then PC ← PC + k + 1	None
BRVS	k	Branch if Overflow Flag is Set	if (V = 1) then PC ← PC + k + 1	None
BRVC	k	Branch if Overflow Flag is Cleared	if (V = 0) then PC ← PC + k + 1	None
BRIE	k	Branch if Interrupt Enabled	if (I = 1) then PC ← PC + k + 1	None
BRID	k	Branch if Interrupt Disabled	if (I = 0) then PC ← PC + k + 1	None

Bit and Bit-Test Instructions

Mnem.	Operan.	Description	Operation	Flags
SBI	P, b	Set Bit in I/O Register	I/O(P, b) ← 1	None
CBI	P, b	Clear Bit in I/O Register	I/O(P, b) ← 0	None
LSL	Rd	Logical Shift Left	Rd(n + 1) ← Rd(n), Rd(0) ← 0	Z,C,N,V
LSR	Rd	Logical Shift Right	Rd(n)←Rd(n+1), Rd(7)←0	Z,C,N,V
ROL	Rd	Rotate Left Through Carry	Rd(0)←C, Rd(n+1)←Rd(n), C←Rd(7)	Z,C,N,V
ROR	Rd	Rotate Right Through Carry	Rd(7) ← C, Rd(n) ← Rd(n + 1), C ← Rd(0)	Z,C,N,V
ASR	Rd	Arithmetic Shift Right	Rd(n) ← Rd(n + 1), n = 0..6	Z,C,N,V
SWAP	Rd	Swap Nibbles	Rd(3..0) ← Rd(7..4), Rd(7..4) ← Rd(3..0)	None
BSET	s	Flag Set	SREG(s) ← 1	SREG(s)
BCLR	s	Flag Clear	SREG(s) ← 0	SREG(s)
BST	Rr, b	Bit Store from Register to T	T ← Rr(b)	T
BLD	Rd, b	Bit load from T to Register	Rd(b) ← T	None
SEC		Set Carry	C ← 1	C
CLC		Clear Carry	C ← 0	C
SEN		Set Negative Flag	N ←1	N
CLN		Clear Negative Flag	N ← 0	N
SEZ		Set Zero Flag	Z ←1	Z
CLZ		Clear Zero Flag	Z ← 0	Z
SEI		Global Interrupt Enable	I ← 1	I
CLI		Global Interrupt Disable	I ← 0	I
SES		Set Signed Test Flag	S ← 1	S
CLS		Clear Signed Test Flag	S ← 0	S
SEV		Set Two's Complement Overflow	V ← 1	V
CLV		Clear Two's Complement Overflow	V ← 0	V
SET		Set T in SREG	T ← 1	T
CLT		Clear T in SREG	T ← 0	T
SEH		Set Half Carry Flag in SREG	H ←1	H
CLH		Clear Half Carry Flag in SREG	H ← 0	H

Arithmetic and Logic Instructions

Mnem.	Operands	Description	Operation	Flags
ADD	Rd, Rr	Add two Registers	Rd ← Rd + Rr	Z,C,N,V,H
ADC	Rd, Rr	Add with Carry two Registers	Rd ← Rd + Rr + C	Z,C,N,V,H
ADIW	Rdl, K	Add Immediate to Word	Rdh:Rdl ← Rdh:Rdl + K	Z,C,N,V,S
SUB	Rd, Rr	Subtract two Registers	Rd ← Rd – Rr	Z,C,N,V,H
SUBI	Rd, K	Subtract Constant from Register	Rd ← Rd – K	Z,C,N,V,H
SBC	Rd, Rr	Subtract with Carry two Registers	Rd ← Rd – Rr – C	Z,C,N,V,H
SBCI	Rd, K	Subtract with Carry Constant from Reg.	Rd ← Rd – K – C	Z,C,N,V,H
SBIW	Rdl, K	Subtract Immediate from Word	Rdh:Rdl ← Rdh:Rdl – K	Z,C,N,V,S
AND	Rd, Rr	Logical AND Registers	Rd ← Rd • Rr	Z,N,V
ANDI	Rd, K	Logical AND Register and Constant	Rd ← Rd • K	Z,N,V
OR	Rd, Rr	Logical OR Registers	Rd ← Rd v Rr	Z,N,V
ORI	Rd, K	Logical OR Register and Constant	Rd ← Rd v K	Z,N,V
EOR	Rd, Rr	Exclusive OR Registers	Rd ← Rd ⊕ Rr	Z,N,V
COM	Rd	One's Complement	Rd ← $FF – Rd	Z,C,N,V
NEG	Rd	Two's Complement	Rd ← $00 – Rd	Z,C,N,V,H
SBR	Rd, K	Set Bit(s) in Register	Rd ← Rd v K	Z,N,V
CBR	Rd, K	Clear Bit(s) in Register	Rd ← Rd • ($FF – K)	Z,N,V
INC	Rd	Increment	Rd ← Rd + 1	Z,N,V
DEC	Rd	Decrement	Rd ← Rd – 1	Z,N,V
TST	Rd	Test for Zero or Minus	Rd ← Rd • Rd	Z,N,V
CLR	Rd	Clear Register	Rd ← $00	Z,N,V
SER	Rd	Set Register	Rd ← $FF	None
MUL	Rd, Rr	Multiply Unsigned	R1:R0 ← Rd x Rr	Z,C
MULS	Rd, Rr	Multiply Signed	R1:R0 ← Rd x Rr	Z,C
MULSU	Rd, Rr	Multiply Signed with Unsigned	R1:R0 ← Rd x Rr	Z,C
FMUL	Rd, Rr	Fractional Multiply Unsigned	R1:R0 ← (Rd x Rr)<< 1	Z,C
FMULS	Rd, Rr	Fractional Multiply Signed	R1:R0 ← (Rd x Rr)<< 1	Z,C
FMULSU	Rd, Rr	Fractional Multiply Signed with Unsigned	R1:R0 ← (Rd x Rr)<< 1	Z,C

MCU Control Instructions

Mnemonics	Operands	Description	Operation	Flags
NOP		No Operation		None
SLEEP		Sleep	(see specific descr. for Sleep function)	None
WDR		Watchdog Reset	(see specific descr. for WDR/timer)	None
BREAK		Break	For On-Chip Debug Only	None

M. Mazidi, S. Naimi, and S. Naimi, The AVR Microcontroller and Embedded Systems Using Assembly and C, 2nd Edition, 2014.