Mnemonic	Opcode																32 bits	SREG								Syntax	Destination		Source		Description	Alias
	15			12				8				4				0		I	T	H	S	V	N	Z	C							
ADC	0	0	0	**1**	1	1	r	d	d	d	d	d	r	r	r	r				*	*	*	*	*	*	ADC Rd, Rr	Rd	0 ≤ d ≤ 31	Rr	0 ≤ r ≤ 31	Add with carry	
ADD	0	0	0	**0**	1	1	r	d	d	d	d	d	r	r	r	r				*	*	*	*	*	*	ADD Rd, Rr	Rd	0 ≤ d ≤ 31	Rr	0 ≤ r ≤ 31	Add without carry	
ADIW	1	0	0	1	0	1	1	0	K	K	d	d	K	K	K	K					*	*	*	*	*	ADIW Rd+1:Rd K	Rd	d ∈ {24,26,28,30}		0 ≤ K ≤ 63	Add immediate to word	
AND	0	0	1	0	0	0	r	d	d	d	d	d	r	r	r	r					*	0	*	*		AND Rd, Rr	Rd	0 ≤ d ≤ 31	Rr	0 ≤ r ≤ 31	Logical AND	
ANDI	0	1	1	1	K	K	K	K	d	d	d	d	K	K	K	K					*	0	*	*		ANDI Rd, K	Rd	16 ≤ d ≤ 31	K	0 ≤ K ≤ 255	Logical AND with immediate	
ASR	1	0	0	1	0	1	0	d	d	d	d	d	0	1	0	1					*	*	*	*	*	ASR Rd	Rd	16 ≤ d ≤ 31			Arithmetic shift right	
BCLR	1	0	0	1	0	1	0	0	**1**	s	s	s	1	0	0	0		?	?	?	?	?	?	?	?	BCLR s			s	0 ≤ s ≤ 7	Bit clear in SREG	
BLD	1	1	1	1	1	0	**0**	d	d	d	d	d	0	b	b	b										BLD Rd, b	Rd	0 ≤ d ≤ 31	b	0 ≤ b ≤ 7	Bit load from T flag in SREG to bit in register	
BRBC	1	1	1	1	0	**1**	k	k	k	k	k	k	k	s	s	s										BRBC s, k	s	0 ≤ s ≤ 7	k	-64 ≤ b ≤ 63	Branch if bit in SREG is cleared	
BRBS	1	1	1	1	0	**0**	k	k	k	k	k	k	k	s	s	s										BRBS s, k	s	0 ≤ s ≤ 7	k	-64 ≤ b ≤ 63	Branch if bit in SREG is set	
BREAK	1	0	0	1	0	1	0	1	1	0	0	1	1	0	0	0										BREAK					BREAK	
BREQ	1	1	1	1	0	0	k	k	k	k	k	k	k	0	0	1										BREQ k			k	-64 ≤ b ≤ 63	Branch if equal	BRBS 1, k
BRGE	1	1	1	1	0	1	k	k	k	k	k	k	k	1	0	0										BRGE k			k	-64 ≤ b ≤ 63	Branch if greater of equal (signed)	BRBC 4, k
BRHC	1	1	1	1	0	1	k	k	k	k	k	k	k	1	0	1										BRHC k			k	-64 ≤ b ≤ 63	Branch if half carry flag is cleared	BRBC 5, k
BRHS	1	1	1	1	0	0	k	k	k	k	k	k	k	1	0	1										BRHS k			k	-64 ≤ b ≤ 63	Branch if half carry flag is set	BRBS 5, k
BRID	1	1	1	1	0	1	k	k	k	k	k	k	k	1	1	1										BRID k			k	-64 ≤ b ≤ 63	Branch if global interrupt is disabled	BRBC 7, k
BRIE	1	1	1	1	0	0	k	k	k	k	k	k	k	1	1	1										BRIE k			k	-64 ≤ b ≤ 63	Branch if global interrupt is enabled	BRBS 7, k
BRLO	1	1	1	1	0	0	k	k	k	k	k	k	k	0	0	0										BRLO k			k	-64 ≤ b ≤ 63	Branch if lower (unsigned)	BRBS 0, k
BRLT	1	1	1	1	0	0	k	k	k	k	k	k	k	1	0	0										BRLT k			k	-64 ≤ b ≤ 63	Branch if less than (signed)	BRBS 4, k
BRMI	1	1	1	1	0	0	k	k	k	k	k	k	k	0	1	0										BRMI k			k	-64 ≤ b ≤ 63	Branch if minus	BRBS 2, k
BRNE	1	1	1	1	0	1	k	k	k	k	k	k	k	0	0	1										BRNE k			k	-64 ≤ b ≤ 63	Branch if not equal	BRBC 1, k
BRPL	1	1	1	1	0	1	k	k	k	k	k	k	k	0	1	0										BRPL k			k	-64 ≤ b ≤ 63	Branch if plus	BRBC 2, k
BRSH	1	1	1	1	0	1	k	k	k	k	k	k	k	0	0	0										BRSH k			k	-64 ≤ b ≤ 63	Branch if same or higher (unsigned)	BRBC 0, k
BRTC	1	1	1	1	0	1	k	k	k	k	k	k	k	1	1	0										BRTC k			k	-64 ≤ b ≤ 63	Branch if T flas is cleared	BRBC 6, k
BRTS	1	1	1	1	0	0	k	k	k	k	k	k	k	1	1	0										BRTS k			k	-64 ≤ b ≤ 63	Branch if T flas is set	BRBS 6, k
BRVC	1	1	1	1	0	1	k	k	k	k	k	k	k	0	1	1										BRVC k			k	-64 ≤ b ≤ 63	Branch if overflow flag is cleared	BRBC 3, k
BRVS	1	1	1	1	0	0	k	k	k	k	k	k	k	0	1	1										BRVS k			k	-64 ≤ b ≤ 63	Branch if overflow flag is set	BRBS 3, k
BSET	1	0	0	1	0	1	0	0	**0**	s	s	s	1	0	0	0		?	?	?	?	?	?	?	?	BSET s			s	0 ≤ s ≤ 7	Bit set in SREG	
BST	1	1	1	1	1	0	**1**	d	d	d	d	d	0	b	b	b			?							BST Rd, b	Rd	0 ≤ d ≤ 31	b	0 ≤ b ≤ 7	Bit store from bit in register to T flag in SREG	
CALL	1	0	0	1	0	1	0	k	k	k	k	k	1	1	**1**	k	✓									CALL k			k	0 ≤ k ≤ 4M	Long call to a subroutine	
CBI	1	0	0	1	1	0	**0**	0	A	A	A	A	A	b	b	b										CBI A, b	A	0 ≤ A ≤ 31	b	0 ≤ b ≤ 7	Clear bit in I/O register	
CBR	0	1	1	1	K	K	K	K	d	d	d	d	K	K	K	K					*	0	*	*		CBR Rd, K	Rd	16 ≤ d ≤ 31	K	0 ≤ K ≤ 255	Clears bit in register	ANDI Rd, ($FF – K)
CLC	1	0	0	1	0	1	0	0	1	0	0	0	1	0	0	0									0	CLC					Clear carry flag	BCLR 0
CLH	1	0	0	1	0	1	0	0	1	1	0	1	1	0	0	0				0						CLH					Clear half carry flag	BCLR 5
CLI	1	0	0	1	0	1	0	0	1	1	1	1	1	0	0	0		0								CLI					Clear global interrupt flag	BCLR 7
CLN	1	0	0	1	0	1	0	0	1	0	1	0	1	0	0	0							0			CLN					Clear negative flag	BCLR 2
CLR	0	0	1	0	0	1	d	d	d	d	d	d	d	d	d	d						0	0	0	1	CLR Rd	Rd	0 ≤ d ≤ 31			Clear register	EOR Rd, Rd
CLS	1	0	0	1	0	1	0	0	1	1	0	0	1	0	0	0						0				CLS					Clear sign flag	BCLR 4
CLT	1	0	0	1	0	1	0	0	1	1	1	0	1	0	0	0			0							CLT					Clear T flag	BCLR 6
CLV	1	0	0	1	0	1	0	0	1	0	1	1	1	0	0	0						0				CLV					Clear overflow flag	BCLR 3
CLZ	1	0	0	1	0	1	0	0	1	0	0	1	1	0	0	0								0		CLZ					Clear zero flag	BCLR 1
COM	1	0	0	1	0	1	0	d	d	d	d	d	0	0	0	0					*	0	*	*	1	COM Rd	Rd	0 ≤ d ≤ 31			One’s complement	
CP	0	0	0	1	0	**1**	r	d	d	d	d	d	r	r	r	r				*	*	*	*	*	*	CP Rd, Rr	Rd	0 ≤ d ≤ 31	Rr	0 ≤ r ≤ 31	Compare	
CPC	0	0	0	0	0	**1**	r	d	d	d	d	d	r	r	r	r				*	*	*	*	*	*	CPC Rd, Rr	Rd	0 ≤ d ≤ 31	Rr	0 ≤ r ≤ 31	Compare with carry	
CPI	0	0	1	1	K	K	K	K	d	d	d	d	K	K	K	K				*	*	*	*	*	*	CPI Rd, K	Rd	16 ≤ d ≤ 31	K	0 ≤ K ≤ 255	Compare with immediate	
CPSE	0	0	0	1	0	0	r	d	d	d	d	d	r	r	r	r										CPSE Rd, Rr	Rd	0 ≤ d ≤ 31	Rr	0 ≤ r ≤ 31	Compare and skip if equal	
DEC	1	0	0	1	0	1	0	d	d	d	d	d	1	0	1	0					*	*	*	*		DEC Rd	Rd	0 ≤ d ≤ 31			Decrement	
DES	1	0	0	1	0	1	0	0	K	K	K	K	1	0	1	1										DES K			K	0 ≤ K ≤ 255	Data Encryption Standard	
EICALL	1	0	0	1	0	1	0	**1**	0	0	0	*1*	1	0	0	1										EICALL					Extended indirect call to a subroutine	
EIJMP	1	0	0	1	0	1	0	**0**	0	0	0	*1*	1	0	0	1										EIJMP					Extended indirect jump	
ELPM (1)	1	0	0	1	0	1	0	1	1	1	0	1	1	0	0	0										ELPM					Extended load program memory	
ELPM (2)	1	0	0	1	0	0	0	d	d	d	d	d	0	1	1	**0**										ELPM Rd, Z	Rd	0 ≤ d ≤ 31			Extended load program memory	
ELPM (3)	1	0	0	1	0	0	0	d	d	d	d	d	0	1	1	**1**										ELPM Rd, Z+	Rd	0 ≤ d ≤ 31			Extended load program memory	

Mnemonic	Opcode																32 bits	SREG								Syntax	Destination		Source		Description	Alias
	15			12				8				4				0		I	T	H	S	V	N	Z	C							
EOR	0	0	1	0	0	1	r	d	d	d	d	d	r	r	r	r					*	0	*	*		EOR Rd, Rr	Rd	0 ≤ d ≤ 31	Rr	0 ≤ r ≤ 31	Logial exclusive OR	
FMUL	0	0	0	0	0	0	1	1	0	d	d	d	1	r	r	r								*	*	FMUL Rd, Rr	Rd	16 ≤ d ≤ 23	Rr	16 ≤ r ≤ 23	Factional multiply unsigned	
FMULS	0	0	0	0	0	0	1	1	1	d	d	d	0	r	r	r								*	*	FMULS Rd, Rr	Rd	16 ≤ d ≤ 23	Rr	16 ≤ r ≤ 23	Factional multiply signed	
FMULSU	0	0	0	0	0	0	1	1	1	d	d	d	1	r	r	r								*	*	FMULS Rd, Rr	Rd	16 ≤ d ≤ 23	Rr	16 ≤ r ≤ 23	Factional multiply signed with unsigned	
ICALL	1	0	0	1	0	1	0	**1**	0	0	0	*0*	1	0	0	1										ICALL					Indirect call to a subroutine	
IJMP	1	0	0	1	0	1	0	**0**	0	0	0	*0*	1	0	0	1										IJUMP					Indirect jump	
IN	1	0	1	1	**0**	A	A	d	d	d	d	d	A	A	A	A										IN Rd, A	Rd	0 ≤ d ≤ 31	A	0 ≤ A ≤ 63	Load an I/O location to register	
INC	1	0	0	1	0	1	0	d	d	d	d	d	0	0	1	1					*	*	*	*		INC Rd	Rd	0 ≤ d ≤ 31			Increment	
JMP	1	0	0	1	0	1	0	k	k	k	k	k	1	1	**1**	k	✓									JMP k			k	0 ≤ k ≤ 4M	Long jump	
LAC	1	0	0	1	0	0	1	d	d	d	d	d	0	1	1	0										LAC Rd	Rd	0 ≤ d ≤ 31			Load and clear	
LAS	1	0	0	1	0	0	1	d	d	d	d	d	0	1	0	1										LAS Rd	Rd	0 ≤ d ≤ 31			Load and set	
LAT	1	0	0	1	0	0	1	d	d	d	d	d	0	1	1	1										LAT Rd	Rd	0 ≤ d ≤ 31			Load and toggle	
LD (X,1)	1	0	0	1	0	0	**0**	d	d	d	d	d	*1*	*1*	0	0										LD Rd, X	Rd	0 ≤ d ≤ 31			Load indiect using register X	
LD (X,2)	1	0	0	1	0	0	**0**	d	d	d	d	d	*1*	*1*	0	1										LD Rd, X+	Rd	0 ≤ d ≤ 31			Load indiect using register X post incremented	
LD (X,3)	1	0	0	1	0	0	**0**	d	d	d	d	d	*1*	*1*	1	0										LD Rd, -X	Rd	0 ≤ d ≤ 31			Load indiect using register X pre decremented	
LD (Y,1)	1	0	0	0	0	0	**0**	d	d	d	d	d	*1*	0	0	0										LD Rd, Y	Rd	0 ≤ d ≤ 31			Load indiect using register Y	LDD Rd, Y+0
LD (Y,2)	1	0	0	1	0	0	**0**	d	d	d	d	d	*1*	*0*	0	1										LD Rd, Y+	Rd	0 ≤ d ≤ 31			Load indiect using register Y post incremented	
LD (Y,3)	1	0	0	1	0	0	**0**	d	d	d	d	d	*1*	*0*	1	0										LD Rd, -Y	Rd	0 ≤ d ≤ 31			Load indiect using register Y pre decremented	
LD (Z,1)	1	0	0	0	0	0	**0**	d	d	d	d	d	*0*	0	0	0										LD Rd, Z	Rd	0 ≤ d ≤ 31			Load indiect using register Z	LDD Rd, Z+0
LD (Z,2)	1	0	0	1	0	0	**0**	d	d	d	d	d	*0*	*0*	0	1										LD Rd, Z+	Rd	0 ≤ d ≤ 31			Load indiect using register Z post incremented	
LD (Z,3)	1	0	0	1	0	0	**0**	d	d	d	d	d	*0*	*0*	1	0										LD Rd, -Z	Rd	0 ≤ d ≤ 31			Load indiect using register Z pre decremented	
LDD (Y)	1	0	q	0	q	q	**0**	d	d	d	d	d	*1*	q	q	q										LDD Rd, Y+q	Rd	0 ≤ d ≤ 31			Load indiect using register Y with displacement	
LDD (Z)	1	0	q	0	q	q	**0**	d	d	d	d	d	*0*	q	q	q										LDD Rd, Z+q	Rd	0 ≤ d ≤ 31			Load indiect using register Z with displacement	
LDI	1	1	1	0	K	K	K	K	d	d	d	d	K	K	K	K										LDI, Rd, K	Rd	16 ≤ d ≤ 31	K	0 ≤ K ≤ 255	Load immediate	
LDS (1)	1	0	0	1	0	0	**0**	d	d	d	d	d	0	0	0	0	✓									LDS Rd, k	Rd	0 ≤ d ≤ 31	k	0 ≤ k ≤ 65535	Load direct	
LDS (2)	1	0	1	0	**0**	k	k	k	d	d	d	d	k	k	k	k										LDS Rd, k	Rd	16 ≤ d ≤ 31	k	0 ≤ k ≤ 127	Load direct	
LPM (1)	1	0	0	1	0	1	0	1	1	1	0	0	1	0	0	0										LPM					Load program memory	
LPM (2)	1	0	0	1	0	0	0	d	d	d	d	d	0	1	0	**0**										LPM Rd, Z	Rd	0 ≤ d ≤ 31			Load program memory	
LPM (3)	1	0	0	1	0	0	0	d	d	d	d	d	0	1	0	**1**										LPM Rd, Z+	Rd	0 ≤ d ≤ 31			Load program memory	
LSL	0	0	0	0	1	1	d	d	d	d	d	d	d	d	d	d				*	*	*	*	*	*	LSL Rd	Rd	0 ≤ d ≤ 31			Logical shift left	ADD Rd, Rd
LSR	1	0	0	1	0	1	0	d	d	d	d	d	0	1	1	0					*	*	0	*	*	LSR Rd	Rd	0 ≤ d ≤ 31			Logical shift right	
MOV	0	0	1	0	1	1	r	d	d	d	d	d	r	r	r	r										MOV Rd, Rr	Rd	0 ≤ d ≤ 31	Rr	0 ≤ r ≤ 31	Copy register	
MOVW	0	0	0	0	0	0	0	1	d	d	d	d	r	r	r	r										MOVW Rd+1:Rd, Rr+1:Rr	Rd	d ∈ {0,2,...,30}	Rr	r ∈ {0,2,...,30}	Copy register word	
MUL	1	0	0	1	1	1	r	d	d	d	d	d	r	r	r	r										MUL Rd, Rr	Rd	0 ≤ d ≤ 31	Rr	0 ≤ r ≤ 31	Multiply (unsigned)	
MULS	0	0	0	0	0	0	1	**0**	d	d	d	d	d	d	d	d										MULS Rd, Rr	Rd	16 ≤ d ≤ 31	Rr	16 ≤ r ≤ 31	Multiply (signed)	
MULSU	0	0	0	0	0	0	1	**1**	d	d	d	d	d	d	d	d										MULSU Rd, Rr	Rd	16 ≤ d ≤ 31	Rr	16 ≤ r ≤ 31	Multiply (signed with unsigned)	
NEG	1	0	0	1	0	1	0	d	d	d	d	d	0	0	0	1										NEG Rd	Rd	0 ≤ d ≤ 31			Two's complement (negate)	
NOP	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0										NOP					No operation	
OR	0	0	1	0	1	0	r	d	d	d	d	d	r	r	r	r					*	0	*	*		OR Rd, Rr	Rd	0 ≤ d ≤ 31	Rr	0 ≤ r ≤ 31	Logical OR	
ORI	0	1	1	0	K	K	K	K	d	d	d	d	K	K	K	K					*	0	*	*		ORI Rd, K	Rd	16 ≤ d ≤ 31	K	0 ≤ K ≤ 255	Logical OR with immediate	
OUT	1	0	1	1	**1**	A	A	r	r	r	r	r	A	A	A	A										OUT A, Rr	A	0 ≤ A ≤ 63	Rr	0 ≤ r ≤ 31	Store register to I/O location	
POP	1	0	0	1	0	0	**0**	d	d	d	d	d	1	1	1	1										POP Rd	Rd	0 ≤ d ≤ 31			Pop register from stack	
PUSH	1	0	0	1	0	0	**1**	r	r	r	r	r	1	1	1	1										PUSH Rr			Rr	0 ≤ r ≤ 31	Push register onto stack	
RCALL	1	1	0	**1**	k	k	k	k	k	k	k	k	k	k	k	k										RCALL k			k	-2K ≤ k ≤ 2K	Relative call to a subroutine	
RET	1	0	0	1	0	1	0	1	0	0	0	**0**	1	0	0	0										RET					Return from subroutine	
RETI	1	0	0	1	0	1	0	1	0	0	0	**1**	1	0	0	0										RETI					Return from interrupt service routine	
RJMP	1	0	0	**0**	k	k	k	k	k	k	k	k	k	k	k	k										RJMP k			k	-2K ≤ k ≤ 2K	Relative jump	
ROL	0	0	0	1	1	1	d	d	d	d	d	d	d	d	d	d				*	*	*	*	*	*	ROL Rd	Rd	0 ≤ d ≤ 31			Rotate left through carry	ADC Rd, Rd
ROR	1	0	0	1	0	1	0	d	d	d	d	d	0	1	1	1					*	*	*	*	*	ROR Rd	Rd	0 ≤ d ≤ 31			Rotate right through carry	
SBC	0	0	0	0	1	0	r	d	d	d	d	d	r	r	r	r				*	*	*	*	*	*	SBC Rd, Rr	Rd	0 ≤ d ≤ 31	Rr	0 ≤ r ≤ 31	Subtract with carry	
SBCI	0	1	0	0	K	K	K	K	d	d	d	d	K	K	K	K				*	*	*	*	*	*	SBCI Rd, K	Rd	16 ≤ d ≤ 31	K	0 ≤ K ≤ 255	Subtract immediate with carry	
SBI	1	0	0	1	1	0	**1**	0	A	A	A	A	A	b	b	b										SBI A, b	A	0 ≤ A ≤ 31	b	0 ≤ b ≤ 7	Set bit in I/O register	
SBIC	1	0	0	1	1	0	**0**	1	A	A	A	A	A	b	b	b										SBIC A, b	A	0 ≤ A ≤ 31	b	0 ≤ b ≤ 7	Skip if bit in I/O register is cleared	

Mnemonic	Opcode																32 bits	SREG							Syntax	Destination		Source		Description	Alias
	15			12				8				4				0		I	T	H	S	V	N	Z	C							
SBIS	1	0	0	1	1	0	**0**	1	A	A	A	A	A	b	b	b										SBIS A, b	A	0 ≤ A ≤ 31	b	0 ≤ b ≤ 7	Skip if bit in I/O register is set	
SBIW	1	0	0	1	0	1	1	1	K	K	d	d	K	K	K	K					*	*	*	*	*	SBIW Rd+1:Rd K	Rd	d ∈ {24,26,28,30}		0 ≤ K ≤ 63	Subtract immediate from word	
SBR	0	1	1	0	K	K	K	K	d	d	d	d	K	K	K	K					*	0	*	*		SBR Rd, K	Rd	16 ≤ d ≤ 31	K	0 ≤ K ≤ 255	Set bits in register	ORI Rd, K
SBRC	1	1	1	1	1	1	**0**	r	r	r	r	r	0	b	b	b										SBRC Rd, b	Rd	16 ≤ d ≤ 31	b	0 ≤ b ≤ 7	Skip if bit in register is cleared	
SBRS	1	1	1	1	1	1	**1**	r	r	r	r	r	0	b	b	b										SBRS Rd, b	Rd	16 ≤ d ≤ 31	b	0 ≤ b ≤ 7	Skip if bit in register is set	
SEC	1	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0									?	SEC					Set carry flag	BSET 0
SEH	1	0	0	1	0	1	0	0	0	1	0	1	1	0	0	0			?							SEH					Set half carry flag	BSET 5
SEI	1	0	0	1	0	1	0	0	0	1	1	1	1	0	0	0		?								SEI					Set global interrupt flag	BSET 7
SEN	1	0	0	1	0	1	0	0	0	0	1	0	1	0	0	0							?			SEN					Set negative flag	BSET 2
SER	1	1	1	0	1	1	1	1	d	d	d	d	1	1	1	1										SER Rd	Rd	16 ≤ d ≤ 31			Set all bits in register	LDI Rd, $FF
SES	1	0	0	1	0	1	0	0	0	1	0	0	1	0	0	0					?					SES					Set sign flag	BSET 4
SET	1	0	0	1	0	1	0	0	0	1	1	0	1	0	0	0		?								SET					Set T flag	BSET 6
SEV	1	0	0	1	0	1	0	0	0	0	1	1	1	0	0	0						?				SEV					Set overflow flag	BSET 3
SEZ	1	0	0	1	0	1	0	0	0	0	0	1	1	0	0	0								?		SEZ					set zero flag	BSET 1
SLEEP	1	0	0	1	0	1	0	1	1	0	0	0	1	0	0	0										SLEEP					Set circuit in sleep mode	
SPM	1	0	0	1	0	1	0	1	1	1	1	**0**	1	0	0	0										SPM					Store program memory	
SPM	1	0	0	1	0	1	0	1	1	1	1	**1**	1	0	0	0										SPM Z+					Store program memory	
ST (X,1)	1	0	0	1	0	0	**1**	r	r	r	r	r	*1*	*1*	0	0										ST X, Rr			Rr	0 ≤ r ≤ 31	Store indiect using register X	
ST (X,2)	1	0	0	1	0	0	**1**	r	r	r	r	r	*1*	*1*	0	1										ST X+, Rr			Rr	0 ≤ r ≤ 31	Store indiect using register X post incremented	
ST (X,3)	1	0	0	1	0	0	**1**	r	r	r	r	r	*1*	*1*	1	0										ST -X, Rr			Rr	0 ≤ r ≤ 31	Store indiect using register X pre decremented	
ST (Y,1)	1	0	0	0	0	0	**1**	r	r	r	r	r	*1*	0	0	0										ST Y, Rr			Rr	0 ≤ r ≤ 31	Store indiect using register Y	LDD Y+0, Rr
ST (Y,2)	1	0	0	1	0	0	**1**	r	r	r	r	r	*1*	*0*	0	1										ST Y+, Rr			Rr	0 ≤ r ≤ 31	Store indiect using register Y post incremented	
ST (Y,3)	1	0	0	1	0	0	**1**	r	r	r	r	r	*1*	*0*	1	0										ST -Y, Rr			Rr	0 ≤ r ≤ 31	Store indiect using register Y pre decremented	
ST (Z,1)	1	0	0	0	0	0	**1**	r	r	r	r	r	*0*	0	0	0										ST Z, Rr			Rr	0 ≤ r ≤ 31	Store indiect using register Z	STD Z+0, Rd
ST (Z,2)	1	0	0	1	0	0	**1**	r	r	r	r	r	*0*	*0*	0	1										ST Z+, Rr			Rr	0 ≤ r ≤ 31	Store indiect using register Z post incremented	
ST (Z,3)	1	0	0	1	0	0	**1**	r	r	r	r	r	*0*	*0*	1	0										ST -Z, Rr			Rr	0 ≤ r ≤ 31	Store indiect using register Z pre decremented	
STD (Y)	1	0	q	0	q	q	**1**	r	r	r	r	r	*1*	q	q	q										STD Y+q, Rr			Rr	0 ≤ r ≤ 31	Store indiect using register Y with displacement	
STD (Z)	1	0	q	0	q	q	**1**	r	r	r	r	r	*0*	q	q	q										STD Z+q Rr			Rr	0 ≤ r ≤ 31	Store indiect using register Z with displacement	
STS (1)	1	0	0	1	0	0	**1**	r	r	r	r	r	0	0	0	0	✓									LDS k, Rr	k	0 ≤ k ≤ 65535	Rr	0 ≤ r ≤ 31	Store direct	
STS (2)	1	0	1	0	**0**	k	k	k	r	r	r	r	k	k	k	k										LDS k, Rr	k	0 ≤ k ≤ 127	Rr	16 ≤ r ≤ 31	Store direct	
SUB	0	0	0	1	1	0	r	d	d	d	d	d	r	r	r	r					*	*	*	*	*	*	SUB Rd, Rr	Rd	0 ≤ d ≤ 31	Rr	0 ≤ r ≤ 31	Subtract without carry	
SUBI	0	1	0	1	K	K	K	K	d	d	d	d	K	K	K	K					*	*	*	*	*	*	SUBI Rd, K	Rd	16 ≤ d ≤ 31	K	0 ≤ K ≤ 255	Subtract immediate	
SWAP	1	0	0	1	0	1	0	d	d	d	d	d	0	0	1	0										SWAP Rd	Rd	0 ≤ d ≤ 31			Swap nibbles	
TST	0	0	1	0	0	0	d	d	d	d	d	d	d	d	d	d						*	0	*	*	TST Rd	Rd	0 ≤ d ≤ 31			Test for zero or minus	AND Rd, Rd
WDR	1	0	0	1	0	1	0	1	1	0	1	0	1	0	0	0										WDR					Watchdog reset	
XCH	1	0	0	1	0	0	1	d	d	d	d	d	0	1	0	0										XCH Z, Rd	Z		Rd	0 ≤ r ≤ 31	Exchange	