- MyCPU -

Microcode 2.3 Reference

Dennis Kuschel
Corintostraße 21
28279 Bremen
Germany

dennis_k@freenet.de
http://www.mycpu.eu

Architecture	8 bit data, 16 bit addresses, similar to Harvard
Maximum Speed	8 MHz, depends on speed of EPROMs
Program Memory	max. 64 kb addressable ROM
Data Memory	max. 64 kb addressable RAM or IO
Registers	hardware level: 5 general purpose registers, 1 constant register software level: 1 accumulator, 2 index registers, 1 stack pointer
Arithmetic	high speed ALU, maximum execution time = 3 cycles (375ns at 8MHz) logical speed: 0.8 million multiplications per second (faster than a 8051!)
Interrupts	1 mask-able hardware interrupt, 1 software interrupt, system reset
Stack	256 byte in conventional data memory (0100h-01FFh), programmable
Addressing Modes	14 Addressing Modes: immediate 16bit, immediate 8bit, immediate zeropage, direct absolute, direct zeropage, direct absolute plus index, indirect absolute, indirect zeropage, indirect absolute plus index, indirect zeropage plus index, indirect plus index absolute, absolute pointer 16bit, absolute pointer 8bit, immediate registers
Instruction Set	256 OP-Codes, average 6.9 cycles per instruction (normal program code) 9 OP-Codes are still undefined and can be defined by the user: 77h, 87h, 97h, A7h, B7h, C7h, D7h, E7h, F7h

Mnemonic	Function	Description	OP	Flags	Bytes	Cycles
CLA	0 -> A	clear Accu	2C	V, Z	1	3
CLX	0 -> X	clear X-Register	2D	V, Z	1	3
CLY	0 -> Y	clear Y-Register	2E	V, Z	1	3
LDA #data	data -> A	load Accu immediate	30	V, Z	2	4
LDA (ZP)	((ZP)) -> A	load content of RAM indirectly addressed by 16bit pointer stored in zeropage into Accu	4B	V, Z	2	13
LDA (ZP),X	((ZP)+X) -> A	load content of RAM indirectly addressed by 16bit pointer stored in zeropage plus X into Accu	33	V, Z	2	14
LDA (ZP),Y	((ZP)+Y) -> A	load content of RAM indirectly addressed by 16bit pointer stored in zeropage plus Y into Accu	34	V, Z	2	14
LDAA abs	(abs) -> A	load content of RAM addressed by abs (16bit) into Accu	32	V, Z	3	8
LDA abs,X	(abs+X) -> A	load content of RAM addressed by abs (16bit) plus X into Accu	35	V, Z	3	10
LDA abs,Y	(abs+Y) -> A	load content of RAM addressed by abs (16bit) plus Y into Accu	36	V, Z	3	10
LDA ZP	(ZP) -> A	load content of RAM addressed by ZP (8bit) into Accu	31	V, Z	2	6
LDAC (ZP),X	[(ZP)+X] -> A	load content of ROM indirectly addressed by 16bit pointer stored in zeropage plus X into Accu	39	V,Z	2	14
LDAC (ZP),Y	[(ZP)+Y] -> A	load content of ROM indirectly addressed by 16bit pointer stored in zeropage plus Y into Accu	3A	V,Z	2	14
LDAC abs,X	[abs+X] -> A	load content of ROM addressed by abs (16bit) plus X into Accu	37	V, Z	3	10
LDAC abs,Y	[abs+Y] -> A	load content of ROM addressed by abs (16bit) plus Y into Accu	38	V, Z	3	10
LDCC abs,X	scratch = A [abs+X] -> A X = scratch	This OP does the same like PHA / LDAC abs,X / PLX. Usefull for CRC16 calculation	4C		3	10
LDX #data	data -> X	load X-Register immediate	50	V, Z	2	4
LDXA abs	(abs) -> X	load content of RAM addressed by abs (16bit) into X-Register	52	V, Z	3	8
LDX abs,Y	(abs+Y) -> X	load content of RAM addressed by abs (16bit) plus Y into X-Register	53	V, Z	3	10
LDX ZP	(ZP) -> X	load content of RAM addressed by ZP (8bit) into X-Register	51	V, Z	2	6
LDY #data	data -> Y	load Y-Register immediate	57	V, Z	2	4
LDYA abs	(abs) -> Y	load content of RAM addressed by abs (16bit) into Y-Register	59	V, Z	3	8
LDY abs,X	(abs+X) -> Y	load content of RAM addressed by abs (16bit) plus X into Y-Register	5A	V, Z	3	10
LDY ZP	(ZP) -> Y	load content of RAM addressed by ZP (8bit) into Y-Register	58	V, Z	2	6
LPA	(X+256*Y) -> A XY=XY+1	load content of RAM addressed by X- and Y-Register into Accu and increment 16 bit pointer	EA	V, Z	1	10

Mnemonic	Function	Description	OP	Flags	Bytes	Cycles
LPT #data	data%256 -> X data/256 -> Y	store 16 bit number immediate into X- and Y-Register	6C		3	6
LPTA abs	(abs) -> X (abs+1) -> Y	load 16 bit number from RAM addressed by abs (16 bit) into X- and Y-Register	6D		3	14
LPT ZP	(ZP) -> X (ZP+1) -> Y	load 16 bit number from RAM addressed by ZP (8 bit) into X- and Y-Register	5F		3	10
MOV ZP,#data	data -> (ZP)	load data immediate to zeropage	48		3	8
MOV ZP1,ZP2	(ZP2) -> (ZP1)	copy content of RAM addressed by ZP2 (8bit) into RAM addressed by ZP1 (8bit)	47		3	11
STA (ZP),X	A -> ((ZP)+X)	store Accu into RAM indirectly addressed by 16bit pointer stored in zeropage plus X	43		2	14
STA (ZP),Y	A -> ((ZP)+Y)	store Accu into RAM indirectly addressed by 16bit pointer stored in zeropage plus Y	44		2	14
STAA abs	A -> (abs)	store Accu into RAM addressed by abs (16bit)	42		3	8
STA abs,X	A -> (abs+X)	store Accu into RAM addressed by abs (16bit) plus X	45		3	10
STA abs,Y	A -> (abs+Y)	store Accu into RAM addressed by abs (16bit) plus Y	46		3	10
STA ZP	A -> (ZP)	store Accu into RAM addressed by ZP (8bit)	41		2	6
STXA abs	X -> (abs)	store X-Register into RAM addressed by abs (16bit)	55		3	8
STX abs,Y	X -> (abs+Y)	store X-Register into RAM addressed by abs (16bit) plus Y	56		3	10
STX ZP	X -> (ZP)	store X-Register into RAM addressed by ZP (8bit)	54		2	6
STYA abs	Y -> (abs)	store Y-Register into RAM addressed by abs (16bit)	5C		3	8
STY abs,X	Y -> (abs+X)	store Y-Register into RAM addressed by abs (16bit) plus X	5D		3	10
STY ZP	Y -> (ZP)	store Y-Register into RAM addressed by ZP (8bit)	5B		2	6
STZ abs STZA abs	0 -> (abs)	store zero into RAM addressed by abs (16bit)	2F		3	8
SAX	A <-> X	swap Accu and X-Register	28		1	4
SAY	A <-> Y	swap Accu and Y-Register	29		1	4
SPA	A -> (X+256*Y) XY=XY+1	store Accu into RAM addressed by X- and Y-Register and increment 16 bit pointer	FA		1	8
SPTA abs	X -> (abs) Y -> (abs+1)	store 16 bit number stored in X- and Y-Register into RAM addressed by abs (16 bit)	6E		3	14
SPT ZP	X -> (ZP) Y -> (ZP+1)	store 16 bit number stored in X- and Y-Register into RAM addressed by ZP (8 bit)	6F		3	10

Mnemonic	Function	Description	OP	Flags	Bytes	Cycles
SWP	A = A*16 + A/16	swap high- and low-nibble of Accu	40	V, Z	1	11
SXY	X <-> Y	swap X- and Y-Register	2A		1	4
TAX	A -> X	transfer Accu into X-Register	20	V, Z	1	3
TAY	A -> Y	transfer Accu into Y-Register	22	V, Z	1	3
TSX	S -> X	transfer Stackpointer into X-Register	26	V, Z	1	3
TXA	X -> A	transfer X-Register into Accu	21	V, Z	1	3
TXS	X -> SP	transfer X-Register into Stackpointer	27	V, Z	1	3
TXY	X -> Y	transfer X- into Y-Register	24	V, Z	1	3
TYA	Y -> A	transfer Y-Register into Accu	23	V, Z	1	3
TYX	Y -> X	transfer Y- into X-Register	25	V, Z	1	3

Mnemonic	Function	Description	OP	Flags	Bytes	Cycles
ADC #data	A = A + data + C	add immediate data to Accu	80	C, V, Z	2	4
ADC (ZP),X	A = A + ((ZP)+X) + C	add content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus X to Accu	83	C, V, Z	2	14
ADC (ZP),Y	A = A + ((ZP)+Y) + C	add content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus Y to Accu	84	C, V, Z	2	14
ADCA abs	A = A + (abs) + C	add content of RAM addressed by abs (16bit) to Accu	82	C, V, Z	3	8
ADC abs,X	A = A + (abs+X) + C	add content of RAM addressed by abs (16bit) plus X to Accu	85	C, V, Z	3	10
ADC abs,Y	A = A + (abs+Y) + C	add content of RAM addressed by abs (16bit) plus Y to Accu	86	C, V, Z	3	10
ADC ZP	A = A + (ZP) + C	add content of RAM addressed by ZP (8bit) to Accu	81	C, V, Z	2	6
ADCC abs,X	A = A + [abs+X] + C	add content of ROM addressed by abs (16bit) plus X to Accu	88	C, V, Z	3	10
ADCC abs,Y	A = A + [abs+Y] + C	add content of ROM addressed by abs (16bit) plus Y to Accu	89	C, V, Z	3	10
AND #data	A = A AND data	logical AND with Accu and immediate data	D0	V, Z	2	4
AND (ZP),X	A = A AND ((ZP)+X)	logical AND with Accu and content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus X	D3	V, Z	2	14
AND (ZP),Y	A = A AND ((ZP)+Y)	logical AND with Accu and content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus Y	D4	V, Z	2	14
ANDA abs	A = A AND (abs)	logical AND with Accu and content of RAM addressed by abs (16bit)	D2	V, Z	3	8
AND abs,X	A = A AND (abs+X)	logical AND with Accu and content of RAM addressed by abs (16bit) plus X	D5	V, Z	3	10
AND abs,Y	A = A AND (abs+Y)	logical AND with Accu and content of RAM addressed by abs (16bit) plus Y	D6	V, Z	3	10
AND ZP	A = A AND (ZP)	logical AND with Accu and content of RAM addressed by ZP (8bit)	D1	V, Z	2	6
ANDC abs,X	A = A AND [abs+X]	logical AND with Accu and content of ROM addressed by abs (16bit) plus X	D8	V, Z	3	10
ANDC abs,Y	A = A AND [abs+Y]	logical AND with Accu and content of ROM addressed by abs (16bit) plus Y	D9	V, Z	3	10
CAX	A - X	compare Accu with X-Register	68	C, V, Z	1	5
CAY	A - Y	compare Accu with Y-Register	69	C, V, Z	1	5
CMP #data	A - data	compare Accu with immediate data	70	C, V, Z	2	5
CMP (ZP),X	A - ((ZP)+X)	compare Accu with content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus X	73	C, V, Z	2	15
CMP (ZP),Y	A - ((ZP)+Y)	compare Accu with content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus Y	74	C, V, Z	2	15
CMPA abs	A - (abs)	compare Accu with content of RAM addressed by abs (16bit)	72	C, V, Z	3	8

Mnemonic	Function	Description	OP	Flags	Bytes	Cycles
CMP abs,X	A - (abs+X)	compare Accu with content of RAM addressed by abs (16bit) plus X	75	C, V, Z	3	11
CMP abs,Y	A - (abs+Y)	compare Accu with content of RAM addressed by abs (16bit) plus Y	76	C, V, Z	3	11
CMP ZP	A - (ZP)	compare Accu with content of RAM addressed by ZP (8bit)	71	C, V, Z	2	6
CMPC abs,X	A - [abs+X]	compare Accu with content of ROM addressed by abs (16bit) plus X	78	C, V, Z	3	11
CMPC abs,Y	A - [abs+Y]	compare Accu with content of ROM addressed by abs (16bit) plus Y	79	C, V, Z	3	11
CPX #data	X - data	compare X-Register with immediate data	60	C, V, Z	2	5
CPX ZP	X - (ZP)	compare X-Register with content of RAM addressed by ZP (8bit)	61	C, V, Z	2	6
CPXA abs	X - (abs)	compare X-Register with content of RAM addressed by abs (16bit)	62	C, V, Z	3	8
CPY #data	Y - data	compare Y-Register with immediate data	64	C, V, Z	2	5
CPY ZP	Y - (ZP)	compare Y-Register with content of RAM addressed by ZP (8bit)	65	C, V, Z	2	6
CPYA abs	Y - (abs)	compare Y-Register with content of RAM addressed by abs (16bit)	66	C, V, Z	3	8
CXY	X - Y	compare X-Register with Y-Register	6A	C, V, Z	1	6
DEC	A = A - 1	decrement Accu	9B	V, Z	1	3
DEC (ZP),X	((ZP)+X) = ((ZP)+X) - 1	decrement the content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus X	9E	V, Z	2	16
DEC (ZP),Y	((ZP)+Y) = ((ZP)+Y) - 1	decrement the content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus Y	9F	V, Z	2	16
DECA abs	(abs) = (abs) - 1	decrement the content of RAM addressed by abs (16bit)	8D	V, Z	3	10
DEC abs,X	(abs+X) = (abs+X) -1	decrement the content of RAM addressed by abs (16bit) plus X	8E	V, Z	3	12
DEC abs,Y	(abs+Y) = (abs+Y)-1	decrement the content of RAM addressed by abs (16bit) plus Y	8F	V, Z	3	12
DEC ZP	(ZP) = (ZP) - 1	decrement the content of RAM addressed by ZP (8bit)	8C	V, Z	2	8
DEX	X = X - 1	decrement X-Register	AB	V, Z	1	3
DEY	Y = Y - 1	decrement Y-Register	BB	V, Z	1	3
DIV #data	A = A / data	divide Accu by immediate data	B0	V, Z	2	4
DIV (ZP),X	A = A / ((ZP)+X)	divide Accu by content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus X	B3	V, Z	2	14
DIV (ZP),Y	A = A / ((ZP)+Y)	divide Accu by content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus Y	B4	V, Z	2	14
DIVA abs	A = A / (abs)	divide Accu by content of RAM addressed by abs (16bit)	B2	V, Z	3	8
DIV abs,X	A = A / (abs+X)	divide Accu by content of RAM addressed by abs (16bit) plus X	B5	V, Z	3	10
DIV abs,Y	A = A / (abs+Y)	divide Accu by content of RAM addressed by abs (16bit) plus Y	B6	V, Z	3	10
DIV ZP	A = A / (ZP)	divide Accu by content of RAM addressed by ZP (8bit)	B1	V, Z	2	6

Mnemonic	Function	Description	OP	Flags	Bytes	Cycles
DIVC abs,X	A = A / [abs+X]	divide Accu by content of ROM addressed by abs (16bit) plus X	B8	V, Z	3	10
DIVC abs,Y	A = A / [abs+Y]	divide Accu by content of ROM addressed by abs (16bit) plus Y	B9	V, Z	3	10
EOR #data	A = A EOR data	logical AND with Accu and immediate data	F0	V, Z	2	4
EOR (ZP),X	A = A EOR ((ZP)+X)	logical exclusive OR with Accu and content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus X	F3	V, Z	2	14
EOR (ZP),Y	A = A EOR ((ZP)+Y)	logical exclusive OR with Accu and content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus Y	F4	V, Z	2	14
EORA abs	A = A EOR (abs)	logical exclusive OR with Accu and content of RAM addressed by abs (16bit)	F2	V, Z	3	8
EOR abs,X	A = A EOR (abs+X)	logical exclusive OR with Accu and content of RAM addressed by abs (16bit) plus X	F5	V, Z	3	10
EOR abs,Y	A = A EOR (abs+Y)	logical exclusive OR with Accu and content of RAM addressed by abs (16bit) plus Y	F6	V, Z	3	10
EOR ZP	A = A EOR (ZP)	logical exclusive OR with Accu and content of RAM addressed by ZP (8bit)	F1	V, Z	2	6
EORC abs,X	A = A EOR [abs+X]	logical exclusive OR with Accu and content of ROM addressed by abs (16bit) plus X	F8	V, Z	3	10
EORC abs,Y	A = A EOR [abs+Y]	logical exclusive OR with Accu and content of ROM addressed by abs (16bit) plus Y	F9	V, Z	3	10
INC	A = A + 1	increment Accu	6B	V, Z	1	3
INC (ZP),X	((ZP)+X) = ((ZP)+X) + 1	increment the content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus X	9C	V, Z	2	16
INC (ZP),Y	((ZP)+Y) = ((ZP)+Y) + 1	increment the content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus Y	9D	V, Z	2	16
INCA abs	(abs) = (abs) + 1	increment the content of RAM addressed by abs (16bit)	7D	V, Z	3	10
INC abs,X	(abs+X) = (abs+X) +1	increment the content of RAM addressed by abs (16bit) plus X	7E	V, Z	3	12
INC abs,Y	(abs+Y) = (abs+Y)+1	increment the content of RAM addressed by abs (16bit) plus Y	7F	V, Z	3	12
INC ZP	(ZP) = (ZP) + 1	increment the content of RAM addressed by ZP (8bit)	7C	V, Z	2	8
INX	X = X + 1	increment X-Register	7B	V, Z	1	3
INY	Y = Y + 1	increment Y-Register	8B	V, Z	1	3

Mnemonic	Function	Description	OP	Flags	Bytes	Cycles
MOD #data	A = A % data	Accu modulo immediate data	C0	V, Z	2	8
MOD (ZP),X	A = A % ((ZP)+X)	Accu modulo content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus X	C3	V, Z	2	18
MOD (ZP),Y	A = A % ((ZP)+Y)	Accu modulo content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus Y	C4	V, Z	2	18
MODA abs	A = A % (abs)	Accu modulo content of RAM addressed by abs (16bit)	C2	V, Z	3	12
MOD abs,X	A = A % (abs+X)	Accu modulo content of RAM addressed by abs (16bit) plus X	C5	V, Z	3	14
MOD abs,Y	A = A % (abs+Y)	Accu modulo content of RAM addressed by abs (16bit) plus Y	C6	V, Z	3	14
MOD ZP	A = A % (ZP)	Accu modulo content of RAM addressed by ZP (8bit)	C1	V, Z	2	10
MODC abs,X	A = A % [abs+X]	Accu modulo content of ROM addressed by abs (16bit) plus X	C8	V, Z	3	14
MODC abs,Y	A = A % [abs+Y]	Accu modulo content of ROM addressed by abs (16bit) plus Y	C9	V, Z	3	14
MUL #data	A = (A * data) % 256 if (C=1) X = (A * data) / 256	multiply Accu with immediate data	A0	V, Z	2	4 / 6
MUL (ZP),X	A = (A * ((ZP)+X)) % 256 if (C=1) Y = (A * data) / 256	multiply Accu with content of RAM indirectly addressed by 16bit pointer stored in zeropage plus X	A3	V, Z	2	14 / 16
MUL (ZP),Y	A = (A * ((ZP)+Y)) % 256 if (C=1) {see MUL #}	multiply Accu with content of RAM indirectly addressed by 16bit pointer stored in zeropage plus Y	A4	V, Z	2	14 / 16
MULA abs	A = (A * (abs)) %256 if (C=1) {see MUL #}	multiply Accu with content of RAM addressed by abs (16bit)	A2	V, Z	3	8 / 10
MUL abs,X	A = (A * (abs+X)) % 256 if (C=1) Y = (A * data) / 256	multiply Accu with content of RAM addressed by abs (16bit) plus X	A5	V, Z	3	10 / 12
MUL abs,Y	A = (A * (abs+Y)) % 256 if (C=1) {see MUL #}	multiply Accu with content of RAM addressed by abs (16bit) plus Y	A6	V, Z	3	10 / 12
MUL ZP	A = (A + (ZP)) % 256 if (C=1) {see MUL #}	multiply Accu with content of RAM addressed by ZP (8bit)	A1	V, Z	2	6 / 8
MULC abs,X	A = (A + [abs+X]) % 256 if (C=1) Y = (A * data) / 256	multiply Accu with content of ROM addressed by abs (16bit) plus X	A8	V, Z	3	10 / 12
MULC abs,Y	A = (A + [abs+Y]) % 256 if (C=1) X = (A * data) / 256	multiply Accu with content of ROM addressed by abs (16bit) plus Y	A9	V, Z	3	10 / 12
ORA #data	A = A OR data	logical OR with Accu and immediate data	E0	V, Z	2	4
ORA (ZP),X	A = A OR ((ZP)+X)	logical OR with Accu and content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus X	E3	V, Z	2	14
ORA (ZP),Y	A = A OR ((ZP)+Y)	logical OR with Accu and content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus Y	E4	V, Z	2	14

Mnemonic	Function	Description	OP	Flags	Bytes	Cycles
ORAA abs	A = A OR (abs)	logical OR with Accu and content of RAM addressed by abs (16bit)	E2	V, Z	3	8
ORA abs,X	A = A OR (abs+X)	logical OR with Accu and content of RAM addressed by abs (16bit) plus X	E5	V, Z	3	10
ORA abs,Y	A = A OR (abs+Y)	logical OR with Accu and content of RAM addressed by abs (16bit) plus Y	E6	V, Z	3	10
ORA ZP	A = A OR (ZP)	logical OR with Accu and content of RAM addressed by ZP (8bit)	E1	V, Z	2	6
ORAC abs,X	A = A OR [abs+X]	logical OR with Accu and content of ROM addressed by abs (16bit) plus X	E8	V, Z	3	10
ORAC abs,Y	A = A OR [abs+Y]	logical OR with Accu and content of ROM addressed by abs (16bit) plus Y	E9	V, Z	3	10
SBC #data	A = A - data - 1 + C	subtract immediate data from Accu	90	C, V, Z	2	4
SBC (ZP),X	A = A - ((ZP)+X) - 1 + C	subtract content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus X from Accu	93	C, V, Z	2	14
SBC (ZP),Y	A = A - ((ZP)+Y) - 1 + C	subtract content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus Y from Accu	94	C, V, Z	2	14
SBCA abs	A = A + (abs) - 1 + C	subtract content of RAM addressed by abs (16bit) from Accu	92	C, V, Z	3	8
SBC abs,X	A = A + (abs+X) - 1 + C	subtract content of RAM addressed by abs (16bit) plus X from Accu	95	C, V, Z	3	10
SBC abs,Y	A = A + (abs+Y) - 1 + C	subtract content of RAM addressed by abs (16bit) plus Y from Accu	96	C, V, Z	3	10
SBC ZP	A = A + (ZP) - 1 + C	subtract content of RAM addressed by ZP (8bit) from Accu	91	C, V, Z	2	6
SBCC abs,X	A = A + [abs+X] - 1 + C	subtract content of ROM addressed by abs (16bit) plus X from Accu	98	C, V, Z	3	10
SBCC abs,Y	A = A + [abs+Y] - 1 + C	subtract content of ROM addressed by abs (16bit) plus Y from Accu	99	C, V, Z	3	10
SHL	A = A * 2	shift left Accu. highest bit will be stored in carry.	CB	C, V, Z	1	3
SHL (ZP),X	((ZP)+X) = ((ZP)+X) * 2	shift left the content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus X. highest bit will be stored in carry.	CC	C, V, Z	2	16
SHL (ZP),Y	((ZP)+Y) = ((ZP)+Y) * 2	shift left the content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus Y. highest bit will be stored in carry.	CD	C, V, Z	2	16
SHLA abs	(abs) = (abs) * 2	shift left the content of RAM addressed by abs (16bit). highest bit will be stored in carry.	AD	C, V, Z	3	10
SHL abs,X	(abs+X) = (abs+X) * 2	shift left the content of RAM addressed by abs (16bit) plus X. highest bit will be stored in carry.	AE	C, V, Z	3	12
SHL abs,Y	(abs+Y) = (abs+Y) *2	shift left the content of RAM addressed by abs (16bit) plus Y. highest bit will be stored in carry.	AF	C, V, Z	3	12

Mnemonic	Function	Description	OP	Flags	Bytes	Cycles
SHL ZP	(ZP) = (ZP) * 2	shift left the content of RAM addressed by ZP (8bit). highest bit will be stored in carry.	AC	C, V, Z	2	8
SHR	A = A / 2	shift right Accu. lowest bit will be stored in carry.	DB	C, V, Z	1	3
SHR (ZP),X	((ZP)+X) = ((ZP)+X) / 2	shift right the content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus X. lowest bit will be stored in carry.	CE	C, V, Z	2	16
SHR (ZP),Y	((ZP)+Y) = ((ZP)+Y) / 2	shift right the content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus Y. lowest bit will be stored in carry.	CF	C, V, Z	2	16
SHRA abs	(abs) = (abs) / 2	shift right the content of RAM addressed by abs (16bit). lowest bit will be stored in carry.	BD	C, V, Z	3	10
SHR abs,X	(abs+X) = (abs+X) / 2	shift right the content of RAM addressed by abs (16bit) plus X. lowest bit will be stored in carry.	BE	C, V, Z	3	12
SHR abs,Y	(abs+Y) = (abs+Y) /2	shift right the content of RAM addressed by abs (16bit) plus Y. lowest bit will be stored in carry.	BF	C, V, Z	3	12
SHR ZP	(ZP) = (ZP) / 2	shift right the content of RAM addressed by ZP (8bit). lowest bit will be stored in carry.	BC	C, V, Z	2	8
ROL	A = A * 2 + C	rotate left Accu. highest bit will be stored in carry.	EB	C, V, Z	1	3
ROL (ZP),X	((ZP)+X) = ((ZP)+X) * 2 + C	rotate left the content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus X. highest bit will be stored in carry.	FC	C, V, Z	2	16
ROL (ZP),Y	((ZP)+Y) = ((ZP)+Y) * 2 + C	rotate left the content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus Y. highest bit will be stored in carry.	FD	C, V, Z	2	16
ROLA abs	(abs) = (abs) * 2 + C	rotate left the content of RAM addressed by abs (16bit). highest bit will be stored in carry.	DD	C, V, Z	3	10
ROL abs,X	(abs+X) = (abs+X) * 2 + C	rotate left the content of RAM addressed by abs (16bit) plus X. highest bit will be stored in carry.	DE	C, V, Z	3	12
ROL abs,Y	(abs+Y) = (abs+Y) * 2 + C	rotate left the content of RAM addressed by abs (16bit) plus Y. highest bit will be stored in carry.	DF	C, V, Z	3	12
ROL ZP	(ZP) = (ZP) * 2 + C	rotate left the content of RAM addressed by ZP (8bit). highest bit will be stored in carry.	DC	C, V, Z	2	8
ROR	A = A / 2 + C * 80h	rotate right Accu. lowest bit will be stored in carry.	FB	C, V, Z	1	3
ROR (ZP),X	((ZP)+X) = ((ZP)+X) / 2 + C *80h	rotate right the content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus X. lowest bit will be stored in carry.	FE	C, V, Z	2	16

Mnemonic	Function	Description	OP	Flags	Bytes	Cycles
ROR (ZP),Y	((ZP)+Y) = ((ZP)+Y) / 2 + C *80h	rotate right the content of RAM indirectly addressed by 16 bit pointer stored in zeropage plus Y. lowest bit will be stored in carry.	FF	C, V, Z	2	16
RORA abs	(abs) = (abs) / 2 + C * 80h	rotate right the content of RAM addressed by abs (16bit). lowest bit will be stored in carry.	ED	C, V, Z	3	10
ROR abs,X	(abs+X) = (abs+X) / 2 + C * 80h	rotate right the content of RAM addressed by abs (16bit) plus X. lowest bit will be stored in carry.	EE	C, V, Z	3	12
ROR abs,Y	(abs+Y) = (abs+Y) /2 + C * 80h	rotate right the content of RAM addressed by abs (16bit) plus Y. lowest bit will be stored in carry.	EF	C, V, Z	3	12
ROR ZP	(ZP) = (ZP) / 2 + C * 80h	rotate right the content of RAM addressed by ZP (8bit). lowest bit will be stored in carry.	EC	C, V, Z	2	8

Mnemonic	Function	Description	OP	Flags	Bytes	Cycles
BIT ZP, #data	(ZP) AND data	logical AND with RAM (ZP = 8bit address) and immediate data	5E	V, Z	3	8
CLC	C = 0	clear the carry flag	04	C	1	3
CLI	I = 0	disable interrupt	02	I	1	3
CLZ	if Z = 1 then Z = 0	clear the zero flag	06	Z	1	3 / 4
FLG ZP	Z=1 if (ZP) is zero, V=1 if bit7 is set	set flags according to the content of the RAM addressed by ZP (8bit)	3C	V, Z	2	6
FLS ZP	Z=1 if (ZP) is zero, V=1 if bit6 is set	set flags according to the content of the RAM addressed by ZP (8bit)	3D	V, Z	2	16
PHP	Flags -> (SP) SP = SP - 1	push flags (C, V, Z) to stack	0B		1	6
PLP	SP = SP + 1 (SP) -> Flags	pull flags (C, V, Z) from stack	0F	C, V, Z	1	7
SEC	if C = 0 then C = 1	set the carry flag	05	C	1	3
SEI	I = 1	enable interrupt	03	I	1	3
SEZ	if Z = 0 then Z = 1	set the zero flag	07	Z	1	3 / 4
JNC abs	if C=0 then PC=abs	jump if carry flag is not set	16		3	4 / 7
JNV abs	if V=0 then PC=abs	jump if sign flag is not set	14		3	4 / 7
JNZ abs	if Z=0 then PC=abs	jump if zero flag is not set	18		3	4 / 7
JPC abs	if C=1 then PC=abs	jump if carry flag is set	17		3	4 / 7
JPV abs	if V=1 then PC=abs	jump if sign flag is set	15		3	4 / 7
JPZ abs	if Z=1 then PC=abs	jump if zero flag is set	19		3	4 / 7

Mnemonic	Function	Description	OP	Flags	Bytes	Cycles
INS #value	SP = SP + value	increment stackpointer by value	DA		2	5
PHA	A -> (SP) SP = SP - 1	push Accu to stack	08		1	6
PHP	Flags -> (SP) SP = SP - 1	push flags (C, V, Z) to stack	0B		1	6
PHR	A -> (SP) X -> (SP-1) Y -> (SP-2) SP = SP - 3	push Accu, X- and Y-Register to stack	7A		1	12
PHX	X -> (SP) SP = SP - 1	push X-Register to stack	09		1	6
PHY	Y -> (SP) SP = SP - 1	push Y-Register to stack	0A		1	6
PLP	SP = SP + 1 (SP) -> Flags	load flags (C, V, Z) from stack	0F	C, V, Z	1	7
PLA	SP = SP + 1 (SP) -> A	load Accu from stack	0C	V, Z	1	7
PLR	(SP+1) -> Y (SP+2) -> X (SP+3) -> A SP = SP + 3	load Accu, X- and Y-Register from stack	8A		1	13
PLX	SP = SP + 1 (SP) -> X	load X-Register from stack	0D	V, Z	1	7
PLY	SP = SP + 1 (SP) -> Y	load Y-Register from stack	0E	V, Z	1	7
POP ZP	SP = SP + 1 (SP) -> (ZP)	load data from stack and write it into RAM addressed by ZP (8bit)	AA		2	11
PUSA #	data / 256 -> (SP) data % 256 -> (SP-1) SP = SP -2	push immediate data to stack	CA		3	13
PUSH #	data -> (SP) SP = SP - 1	push immediate data to stack	BA		2	7
PUSH ZP	(ZP) -> (SP) SP = SP - 1	push content of RAM addressed by ZP (8bit) to stack	9A		2	10
SBK abs,#	(abs & FF00h) -> (SP) SP = SP - 1 data -> (abs & FF00h)	Save bank-select register content to stack and set new value. Register must be at a page boundary.	3E		3	14
RBK abs	SP = SP + 1 (SP) -> (abs & FF00h)	Restore bank-select register. Register must be at a page boundary.	3F		2	11

Mnemonic	Function	Description	OP	Flags	Bytes	Cycles
BRK	flags -> (SP) PC(hi) -> (SP-1) PC(lo) -> (SP-2) SP = SP - 3 PC = (0006h)	execute break interrupt	00		1	19
DLY	do Y=Y-1 while Y<>0	delay program execution	3B	V, Z	1	9*Y
DXJP abs	X = X - 1, if (X<>0) then PC=abs	Decrement X and jump if X is not zero.	49	V, Z	3	7
DYJP abs	Y = Y - 1, if (Y<>0) then PC=abs	Decrement Y and jump if Y is not zero.	4A	V, Z	3	7
JMP (abs)	PC = (abs)	jump to indirect address. the pointer is stored in RAM	11		3	14
JMP (abs,X)	PC = (abs+X)	jump to indirect address stored in RAM at position abs+X	12		3	17
JMP abs	PC = abs	jump to direct address	10		3	7
JMPC (abs,X)	PC = [abs+X]	jump to indirect address stored in ROM at position abs+X	13		3	17
JNC abs	if C=0 then PC=abs	jump if carry flag is not set	16		3	4 / 7
JNV abs	if V=0 then PC=abs	jump if sign flag is not set	14		3	4 / 7
JNZ abs	if Z=0 then PC=abs	jump if zero flag is not set	18		3	4 / 7
JPC abs	if C=1 then PC=abs	jump if carry flag is not set	17		3	4 / 7
JPV abs	if V=1 then PC=abs	jump if sign flag is not set	15		3	4 / 7
JPZ abs	if Z=1 then PC=abs	jump if zero flag is not set	19		3	4 / 7
JSR (abs)	PC-1 / 256 -> (SP) PC-1 %256 ->(SP-1) SP = SP - 2 PC = (abs)	jump to indirectly addressed subroutine. the pointer to the subroutine is stored in RAM.	1B		3	20
JSR abs	PC-1 / 256 -> (SP) PC-1 %256 ->(SP-1) SP = SP - 2 PC = abs	jump to direct addressed subroutine	1A		3	11
NOP		no operation	01		1	3
RTB	(SP+1) -> PC(lo) (SP+2) -> PC(hi) (SP+3) -> flags SP = SP + 3	return from break interrupt	1D		1	13
RTI	(SP+1) -> PC(lo) (SP+2) -> PC(hi) (SP+3) -> flags SP = SP + 3 I = 1	return from interrupt and enable interrupt again	1E		1	13
RTS	(SP+1) -> PC(lo) (SP+2) -> PC(hi) SP = SP + 2 PC = PC + 1	return from subroutine	1F		1	12
SKA	PC = PC + 1	skip one byte	4D		1	3
SKB	PC = PC + 2	skip two bytes	4E		1	4
SKC	PC = PC + 3	skip three bytes	4F		1	5
TSTC abs,X	A AND [abs+X]	logical AND with Accu and content of ROM addressed by abs (16bit) plus X	1C	V, Z	3	10

Mnemonic	Function	Description	OP	Flags	Bytes	Cycles
reserved	reserved	Reserved for future use, use at your own risk!	2B			
reserved	reserved	Reserved for future use, use at your own risk!	63			
reserved	reserved	Reserved for future use, use at your own risk!	67			
			77			
			87			
			97			
			A7			
			B7			
			C7			
			D7			
			E7			
			F7			

Please use this table to write down your own OP-Code definitions.