1. Befehlssatz

1.1 Symbole und Abkürzungen

Operand	Bedeutung
A	Akkumulator (E0)
adr	8 Bit - Adresse
adr11	11 Bit - Adresse
adr16	16 Bit - Adresse
AC	Auxiliary Carry (Hilfsübertrag)
B	Register B
bit	Bitadresse im internen RAM oder im SFR-Bereich (00-7F) (80-FF)
/bit	Invertierter Inhalt der Bitadresse (Komplement)
C	Carrybit
D	Kennzeichnung für ein 4 Bit - Digit (Nibble)
#data	8 Bit - Konstante
#data16	16 Bit - Konstante
direct	Adresse eines internen RAM- oder SFR-Platzes
DPTR	Datenpointregister
I	Interrupt
HB	Highbyte eines Datenwortes
LB	Lowbyte eines Datenwortes
LSB	Bit 0 eines Bytes
MSB	Bit 7 eines Bytes
MZ	Maschinenzyklen
P	Port
PC	Programmzähler
PSW	Programmstatuswort
rel	Signiertes 8 Bit - Offset für Sprungbefehle
@Ri	Adressregister für internes und externes RAM
Rn	Register 0 bis 7 der aktuellen Registerbank
SP	Stackpointer
SFR	Spezialfunktionsregister
T	Timer
TF	Timer - Flag
WB	Wortbreite in Bytes
■	Flag wird beeinflusst
--	Flag wird nicht beeinflußt
WB	Wortbreite des Befehls
MZ	Maschinenzyklen des Befehls
CY	Carry-Flag (D7)
OV	Overflow (Überlauf)- Flag (D2)
AC	Hilfscarry- Flag für BCD- Addition (D6)

1.2 Befehle

1.2.1 Datentransport

Mnemonischer Befehl OpCode	Operanden	Hex-Code	WB	MZ	Beeinflussung Zustandbits CY	OV	AC	Befehlsbeschreibung
MOV	Rn,#data	78-7F	2	1	--	--	--	Direktes Laden des Registers mit einer Konstanten
MOV	A,#data	74	2	1	--	--	--	Akku direkt mit Konstante laden
MOV	adr,#data	75	3	2	--	--	--	Internen Speicher mit Konstante laden
MOV	DPTR,#data	90	3	2	--	--	--	16Bit-Konstante in Datenpointer laden
MOV	@R0,#data	76	2	1	--	--	--	Konstante in internes RAM laden
MOV	@R1,#data	77	2	1	--	--	--	R0 bzw. R1 bestimmen die Ziel Adresse
MOV	A,Rn	E8-EF	1	1	--	--	--	Kopieren Registerinhalt in den Akku
MOV	Rn,A	F8-FF	1	1	--	--	--	Kopieren des Akkuinhaltes in ein Register
MOV	bit,C	92	2	2	--	--	--	Carry-Inhalt in angegebene Bitadresse kopieren
MOV	C,bit	A2	2	1	■	--	--	Der Inhalt Bitadresse in das Carry kopieren
XCH	A,Rn	C8-CF	1	1	--	--	--	Akku- und Registerinhalt austauschen
MOV	Rn,adr	A8-AF	2	2	--	--	--	internen Speicherplatz in ein Register kopieren
MOV	adr,Rn	88-8F	2	2	--	--	--	Registerinhalt in internen Speicherplatz kopieren
MOV	A,adr	E5	2	1	--	--	--	Inhalt interner Speicherplatze in den Akku kopieren
MOV	adr,A	F5	2	1	--	--	--	Inhalt Akku in einen internen Speicherplatz kopieren
MOV	adr,adr	85	3	2	--	--	--	Inhalt interner Speicherplatz in einen anderen kopieren
MOV	A,@R0	E6	1	1	--	--	--	Speicherinhalt des internen RAM in den Akku kopieren(R0
MOV	A,@R1	E7	1	1	--	--	--	bzw. R1 enthält die Quellenadresse)
MOV	@R0,A	F6	1	1	--	--	--	Akkuinhalt in Speicherplatz des internen RAM kopieren(R0
MOV	@R1,A	F7	1	1	--	--	--	bzw. R1 enthält die Zieladresse)
MOV	adr,@R0	86	2	2	--	--	--	Inhalt eines interner Speicherplatz in einen anderen
MOV	adr,@R1	87	2	2	--	--	--	kopieren(R0 bzw. R1 enthält die Quellenadresse)
MOV	@R0,adr	A6	2	2	--	--	--	Inhalt interner Speicherplatz in einen anderen kopieren
MOV	@R1,adr	A7	2	2	--	--	--	(R0 bzw. R1 enthält die Zieladresse)
POP	adr	D0	2	2	--	--	--	Speicherinhalt vom Stack holen
XCH	A,adr	C5	2	1	--	--	--	Internen Speicher mit dem Akkuinhalt tauschen
PUSH	adr	C0	2	2	--	--	--	Speicherinhalt auf den Stack schreiben
XCH	A,@R0	C6	1	1	--	--	--	Inhalt interner Speicherplatzes Akku austauschen
XCH	A,@R1	C7	1	1	--	--	--	(R0 bzw. R1 enthält die Zieladresse
XCHD	A,@R0	D6	1	1	--	--	--	Das LOW-Nibble eines Speicherplatzes im internen RAM gegen das LOW-Nibble des Akkus austauschen. Die HIGH-Nibble beider Speicher werden nicht verändert. (R0 bzw. R1 enthält die Zieladresse)
XCHD	A,@R1	D7	1	1	--	--	--	
MOVX	A,@R0	E2	1	2	--	--	--	Inhalt eines externen Speicherplatzes in den Akku kopieren
MOVX	A,@R1	E3	1	2	--	--	--	
MOVX	@R0,A	F2	1	2	--	--	--	Inhalt des Akkus in einen externen Speicherplatz kopieren.
MOVX	@R1,A	F3	1	2	--	--	--	
MOVX	A,@DPTR	E0	1	2	--	--	--	Inhalt eines externen Speicherplatzes in den Akku kopieren
MOVX	@DPTR,A	F0	1	2	--	--	--	Inhalt des Akkus in einen externen Speicherplatz kopieren
MOVC	A,@A+DPTR	93	1	2	--	--	--	Hole Konstante aus einer Tabelle im EEPROM.
MOVC	A,@A+PC	83	1	2	--	--	--	Hole Konstante aus einer Tabelle im EEPROM.
NOP		00	1	1	--	--	--	Keine Aktivität

1.2.2 Arithmetische Operationen

Mnemonischer Befehl OpCode	Operanden	Hex-Code	WB	MZ	Beeinflussung Zustandbits CY	OV	AC	Befehlsbeschreibung
INC	A	04	1	1	--	--	--	Inhalt des Akku um „1“ erhöhen
INC	Rn	08-0F	1	1	--	--	--	Inhalt des Registers um „1“ erhöhen
INC	adr	05	2	1	--	--	--	Inhalt intern. Speicherstelle um „1“ erhöhen
INC	DPTR	A3	1	2	--	--	--	Inhalt des Datenpointers um „1“ erhöhen
INC	@R0	06	1	1	--	--	--	Inhalt einer Speicherstelle im internen
INC	@R1	07	1	1	--	--	--	RAM um „1“ erhöhen
DEC	A	14	1	1	--	--	--	Inhalt des Akku um „1“ vermindern
DEC	Rn	18-1F	1	1	--	--	--	Inhalt des Registers um „1“ vermindern
DEC	adr	15	2	1	--	--	--	Inhalt interne Speciherst. um „1“ vermindern
DEC	@R0	16	1	1	--	--	--	Inhalt int. Speicherstelle um „1“ vermindern
DEC	@R1	17	1	1	--	--	--	
ADD	A,#data	24	2	1	■	■	■	Addition einer Konstante zum Akkuinhalt
ADDC	A,#data	34	2	1	■	■	■	Addition einer Konstante plus Carry
ADD	A,Rn	28-2F	1	1	■	■	■	Addition eines Registerinhaltes zum Akkuinhalt
ADDC	A,Rn	38-3F	1	1	■	■	■	Add. eines Registerinh. plus Übertrag zum Akkuinhalt
ADD	A,adr	25	2	1	■	■	■	Inhalt int. Speicherstelle zum Akkus addieren
ADDC	A,adr	35	2	1	■	■	■	Inhalt int. Speicherstelle plus CY zum Akku addieren
ADD	A,@R0	26	1	1	■	■	■	Inhalt einer Speicherstelle im internen
ADD	A,@R1	27	1	1	■	■	■	RAM zum Inhalt des Akkus addieren
ADDC	A,@R0	36	1	1	■	■	■	Inhalt einer Speicherstelle im internen RAM
ADDC	A,@R1	37	1	1	■	■	■	plus CY zum Akku addieren
SUBB	A,#data	94	2	1	■	■	■	Subtraktion Konstante plus Carry vom Akku
SUBB	A,adr	95	2	1	■	■	■	Subtrakt. Int. Speicherinhalt plus Carry vom Akku
SUBB	A,Rn	98-9F	1	1	■	■	■	Subtrakt. eines Registers plus Carry vom Akku
SUBB	A,@R0	96	1	1	■	■	■	Subtraktion eines Speicherinhaltesdes
SUBB	A,@R1	97	1	1	■	■	■	internen RAM plus Carry vom Akkuihalt
DA	A	D4	1	1	■	--	--	Dezimalkorrektur des Akku ***nur*** nach einer **BCD-Addition**
CLR	A	E4	1	1	--	--	--	Löschen des Akku-Inhaltes
CPL	A	F4	1	1	--	--	--	Komplementieren des Akku-Inhaltes
SWAP	A	C4	1	1	--	--	--	Vertausche die Nibbles des Akkus
MUL	AB	A4	1	4	■	■	--	Multipliziere den Akku B- Register
DIV	AB	84	1	4	■	■	--	Teile Akkuinhalt durch den B-Registerinhalt
RL	A	23	1	1	--	--	--	Rotiere Akku-Inhalt eine Stelle nach links
RLC	A	33	1	1	■	--	--	Rotiere Akku-Inhalt durch Carry nach links
RR	A	03	1	1	--	--	--	Rotiere Akku-Inhalt eine Stelle nach rechts
RRC	A	13	1	1	■	--	--	Rotiere Akku-Inhalt durch Carry nach rechts
SETB	C	D3	1	1	■	--	--	Setze das CY-Bit auf „1“
SETB	bit	D2	2	1	--	--	--	Setze das adressierte Bit auf „1“
CLR	C	C3	1	1	■	--	--	Setze das CY-Bit auf „0“
CLR	bit	C2	2	1	--	--	--	Setze das adressierte Bit auf „0“
CPL	C	B3	1	1	■	--	--	Komplementiere das CY-Bit
CPL	bit	B2	2	1	--	--	--	Komplementiere das adressierte Bit
ANL	A,#data	54	2	1	--	--	--	Bitweise UND-Verknüpfung Konstante und Akku
ANL	adr,#data	53	3	2	--	--	--	Bitweise UND-Verkn. Konstante und int. Speicherst., das Ergebnis steht in der Speicherst.
ANL	A,Rn	58-5F	1	1	--	--	--	Bitweise UND-Verknüpfung zwischen Akku und Register. *Ergebnis im Akku.*
ANL	A,adr	55	2	1	--	--	--	Bitweise UND-Verknüpfung zwischen Akku inter. Speicherst., *Ergebnis im Akku*
ANL	adr,A	52	2	1	--	--	--	Bitweise UND-Verknüpfung zwischen Akku und RAM-internem Speicher, *Ergebnis im Speicher*
ANL	C,bit	82	2	2	■	--	--	UND-Verknüpfung zwischen Carry und Bit
ANL	C,/bit	B0	2	2	■	--	--	UND-Verknüpfung zwischen Carry und invert. Bit. *Ergebnis jeweils im Carry-Bit*
ANL	A,@R0	56	1	1	--	--	--	Bitweise UND-Verknüpfung
ANL	A,@R1	57	1	1	--	--	--	Bitweise UND-Verknüpfung
ORL	A,#data	44	2	1	--	--	--	Bitweise ODER-Verknüpfung Akku und Konstante
ORL	adr,#data	43	3	2	--	--	--	Bitweise ODER-Verknüpfung Konstante und int. Speichest. *Das Ergebnis steht in der Speicherstelle*

Arithmetische Operationen (Fortsetzung)								
ORL	A,Rn	48-4F	1	1	--	--	--	Bitweise ODER-Verknüpfung zwischen Akku und Register. *Ergebnis im Akku.*
ORL	A,adr	45	2	1	--	--	--	Bitweise ODER-Verknüpfung zwischen Akku und RAM-internem Speicher, *Ergebnis im Akku*
ORL	adr,A	42	2	1	--	--	--	Bitweise ODER-Verknüpfung Akku und int. Speicherst., *Ergebnis im Speicher*
ORL	C,bit	72	2	2	■	--	--	ODER-Verknüpfung zwischen Carry Bit
ORL	C,/bit	A0	2	2	■	--	--	ODER-Verknüpfung zwischen Carry invertiertem Bit. *Ergebnis jeweils im Carry-Bit.*
ORL	A,@R0	46	1	1	--	--	--	Bitweise ODER-Verknüpfung
ORL	A,@R1	47	1	1	--	--	--	Bitweise ODER-Verknüpfung
XRL	A,#data	64	2	1	--	--	--	Bitweise EXKLUSIV-ODER-Verknüpfung
XRL	adr,#data	63	3	2	--	--	--	Bitweise EXKLUSIV-ODER-Verknüpfung Konst. und int. Speicherst. *Ergebnis in Speicherstelle.*
XRL	A,Rn	68-6F	1	1	--	--	--	Bitweise EXKLUSIV-ODER-Verknüpfung Akku und angeg. Register. *Ergebnis im Akku.*
XRL	A,adr	65	2	1	--	--	--	Bitweise EXKLUSIV-ODER-Verknüpfung zw. Akku und RAM-internem Speicher, *Ergebnis im Akku*
XRL	adr,A	62	2	1	--	--	--	Bitweise EXKLUSIV-ODER-Verknüpfung zw. Akku und RAM-internem Speicher, *Ergebnis im Speicher*
XRL	A,@R0	66	1	1	--	--	--	Bitweise EXKLUSIV-ODER-Verknüpfung
XRL	A,@R1	67	1	1	--	--	--	Bitweise EXKLUSIV-ODER-Verknüpfung

1.2.3 Sprungbefehle

Mnemonischer Befehl OpCode	Operanden	Hex-Code	WB	MZ	Beeinflussung Zustandbits CY	OV	AC	Befehlsbeschreibung
LJMP	adr16	02	3	2	--	--	--	Programmsprung im 64K-Block
SJMP	rel	80	2	2	--	--	--	relativer Programmsprung im Bereich -128 bis +127 zur nachfolgenden Befehlsadresse
AJMP	adr11	01-E1	2	2	--	--	--	Sprung im 2k-Block
JMP	@A+DPTR	73	1	2	--	--	--	Springe zur Adresse, die aus Akku- und Datenpointerinhalt gebildet wird.
JBC	bit,rel	10	3	2	--	--	--	Springe bei *gesetzem* Bit *und lösche es*
JB	bit,rel	20	3	2	--	--	--	Springe bei *gesetzem* Bit
JNB	bit,rel	30	3	2	--	--	--	Springe bei *gelöschtem* Bit
JC	rel	40	2	2	--	--	--	Springe bei *gesetztem* Carry-Bit
JNC	rel	50	2	2	--	--	--	Springe bei *gelöschtem* Carry-Bit
JZ	rel	60	2	2	--	--	--	Springe, wenn Akkuinhalt *gleich Null*
JNZ	rel	70	2	2	--	--	--	Springe, wenn Akkuinhalt *ungleich Null*
DJNZ	Rn,rel	D8-DF	2	2	--	--	--	Vermindere Register um Eins und springe, wenn der Rest ungleich Null
DJNZ	adr,rel	D8	3	2	--	--	--	Vermindere den Speicherinhalt im internen RAM um Eins und springe, wenn der Rest ungleich Null.
CJNE	A,#data,rel	B4-BF	3	2	■	--	--	Vergleiche Akku mit Konstante und verzweige bei Ungleichheit. Andernfalls fahre im Programm fort.
CJNE	Rn,#data,rel	B8-BF	3	2	■	--	--	Vergleiche Register mit Konstante und verzweige bei Ungleichheit, andernfalls fahre fort.
CJNE	A,adr,rel	B5	3	2	■	--	--	Vergleiche Akku- und Speicherinhalt und verzweige bei Ungleichheit.
CJNE	@R0,#data,rel	B6	3	2	■	--	--	Vergleiche den Inhalt des RAM-internen
CJNE	@R1,#data,rel	B7	3	2	■	--	--	Speichers mit der Konstante und verzweige bei Ungleichheit. (R0 bzw. R1 enthält die Quellenadresse)
LCALL	Adr16	12	3	2	--	--	--	Unterprogrammaufruf im 64k-Block
ACALL	adr11	11-F1	2	2	--	--	--	Unterprogrammaufruf im 2k-Block
RET		22	1	2	--	--	--	Ende Unterprogramm
RETI		32	1	2	--	--	--	Ende UP plus löschen des INT-Flags

Hexdezimale
Reihenfolge

0 h	NOP	
01-E1	AJMP	adr11
08-0F	INC	Rn
10h	JBC	bit,rel
11-F1	ACALL	adr11
12h	LCALL	Adr16
13h	RRC	A
14h	DEC	A
15h	DEC	adr
16h	DEC	@R0
17h	DEC	@R1
18-1F	DEC	Rn
20h	JB	bit,rel
22h	RET	
23h	RL	A
24h	ADD	A,#data
25h	ADD	A,adr
26h	ADD	A,@R0
27h	ADD	A,@R1
28-2F	ADD	A,Rn
2h	LJMP	adr16
30h	JNB	bit,rel
32h	RETI	
33h	RLC	A
34h	ADDC	A,#data

35h	ADDC	A,adr
36h	ADDC	A,@R0
37h	ADDC	A,@R1
38-3F	ADDC	A,Rn
3h	RR	A
40h	JC	rel
42h	ORL	adr,A
43h	ORL	adr,#data
44h	ORL	A,#data
45h	ORL	A,adr
46h	ORL	A,@R0
47h	ORL	A,@R1
48-4F	ORL	A,Rn
4h	INC	A
50h	JNC	rel
52h	ANL	adr,A
53h	ANL	adr,#data
54h	ANL	A,#data
55h	ANL	A,adr
56h	ANL	A,@R0
57h	ANL	A,@R1
58-5F	ANL	A,Rn
5h	INC	adr
60h	JZ	rel
62h	XRL	adr,A
63h	XRL	adr,#data

64h	XRL	A,#data
65h	XRL	A,adr
66h	XRL	A,@R0
67h	XRL	A,@R1
68-6F	XRL	A,Rn
6h	INC	@R0
70h	JNZ	rel
72h	ORL	C,bit
73h	JMP	@A+DPTR
74h	MOV	A,#data
75h	MOV	adr,#data
76h	MOV	@R0,#data
77h	MOV	@R1,#data
78-7F	MOV	Rn,#data
7h	INC	@R1
80h	SJMP	rel
82h	ANL	C,bit
83h	MOVC	A,@A+PC
84h	DIV	AB
85h	MOV	adr,adr
86h	MOV	adr,@R0
87h	MOV	adr,@R1
88-8F	MOV	adr,Rn
90h	MOV	DPTR,#data
92h	MOV	bit,C
93h	MOVC	A,@A+DPTR

94h	SUBB	A,#data
95h	SUBB	A,adr
96h	SUBB	A,@R0
97h	SUBB	A,@R1
98-9Fh	SUBB	A,Rn
A0	ORL	C,/bit
A2	MOV	C,bit
A3	INC	DPTR
A4	MUL	AB
A6	MOV	@R0,adr
A7	MOV	@R1,adr
A8-AF	MOV	Rn,adr
B0	ANL	C,/bit
B2	CPL	bit
B3	CPL	C
B4-BF	CJNE	A,#data,rel
B5	CJNE	A,adr,rel
B6	CJNE	@R0,#data,rel
B7	CJNE	@R1,#data,rel
B8-BF	CJNE	Rn,#data,rel
C0	PUSH	adr
C2	CLR	bit
C3	CLR	C
C4	SWAP	A
C5	XCH	A,adr
C6	XCH	A,@R0

C7	XCH	A,@R1
C8-CF	XCH	A,Rn
D0	POP	adr
D2	SETB	bit
D3	SETB	C
D4	DA	A
D6	XCHD	A,@R0
D7	XCHD	A,@R1
D8	DJNZ	adr,rel
D8-DF	DJNZ	Rn,rel
E0	MOVX	A,@DPTR
E2	MOVX	A,@R0
E3	MOVX	A,@R1
E4	CLR	A
E5	MOV	A,adr
E6	MOV	A,@R0
E7	MOV	A,@R1
E8-EF	MOV	A,Rn
F0	MOVX	@DPTR,A
F2	MOVX	@R0,A
F3	MOVX	@R1,A
F4	CPL	A
F5	MOV	adr,A
F6	MOV	@R0,A
F7	MOV	@R1,A
F8-FF	MOV	Rn,A