8051 Instruction Set Summary

Rn Register R7-R0 of the currently selected Register Bank.
Data 8-bit internal data location's address. This could be an internal Data RAM location (0-127) or a SFR [i.e. I/O port, control register, status register, etc. (128-255)].
@Ri 8-bit Internal Data RAM location (0-255) addressed indirectly through register R1 or R0.
#data 8-bit constant included in instruction.
#data16 16-bit constant included in instruction.
addr16 16-bit destination address. Used by LCALL and LJMP. A branch can be anywhere within the 64k byte Program Memory address space.
addr11 11-bit destination address. Used by ACALL and AJMP. The branch will be within the same 2k byte page of Program Memory as the first byte of the following instruction.
rel Signed (two's component) 8-bit offset byte. Used by SJMP and all conditional jumps. Range is –128 to +127 bytes relative to first byte of the following instruction.
bit Direct Addressed bit in Internal Data RAM or Special Function Register.

Instruction	Flag			Instruction	Flag		
	C	OV	AC		C	OV	AC
ADD	X	X	X	CLR C	O		
ADDC	X	X	X	CPL C	X		
SUBB	X	X	X	ANL C,bit	X		
MUL	O	X		ANL C,/bit	X		
DIV	O	X		ORL C,bit	X		
DA	X			ORL C,/bit	X		
RRC	X			MOV C,bit	X		
RLC	X			CJNE	X		
SETB C	1						

Note that operations on SFR byte address 206 or bit addresses 209-215 (i.e. the PSW or bits in the PSW) will also affect flag settings.

Mnemonic	Description	Byte	Cycle
Arithmetic operations			
ADD A,Rn	Add register to accumulator	1	1
ADD A,direct	Add direct byte to accumulator	2	1
ADD A,@Ri	Add indirect RAM to accumulator	1	1
ADD A,#data	Add immediate data to accumulator	2	1
ADDC A,Rn	Add register to accumulator with carry flag	1	1
ADDC A,direct	Add direct byte to A with carry flag	2	1
ADDC A,@Ri	Add indirect RAM to A with carry flag	1	1
ADDC A,#data	Add immediate data to A with carry flag	2	1
SUBB A,Rn	Subtract register to accumulator with borrow	1	1
SUBB A,direct	Subtract direct byte to A with carry borrow	2	1
SUBB A,@Ri	Subtract indirect RAM to A with carry borrow	1	1
SUBB A,#data	Subtract immediate data to A with carry borrow	2	1
INC A	Increment accumulator	1	1
INC Rn	Increment register	1	1
INC direct	Increment direct byte	2	1
INC @Ri	Increment indirect RAM	1	1
DEC A	Decrement accumulator	1	1
DEC Rn	Decrement register	1	1
DEC direct	Decrement direct byte	2	1
DEC @Ri	Decrement indirect RAM	1	1
INC DPTR	Increment data pointer	1	2
MUL AB	Multiply A and B -> [B hi]:[A lo]	1	4
DIV AB	Divide A by B -> A=result, B=remainder	1	4
DA A	Decimal adjust accumulator	1	1
CLR A	Clear accumulator	1	1

Mnemonic	Description	Byte	Cycle
CPL A	Complement accumulator	1	1
RL A	Rotate accumulator left	1	1
RLC A	Rotate accumulator left through carry	1	1
RR A	Rotate accumulator right	1	1
RRC A	Rotate accumulator right through carry	1	1
SWAP A	Swap nibbles within the accumulator	1	1
Logic operations			
ANL A,Rn	AND register to accumulator	1	1
ANL A,direct	AND direct byte to accumulator	2	1
ANL A,@Ri	AND indirect RAM to accumulator	1	1
ANL A,#data	AND immediate data to accumulator	2	1
ANL direct,A	AND accumulator to direct byte	2	1
ANL direct,#data	AND immediate data to direct byte	3	2
ORL A,Rn	OR register to accumulator	1	1
ORL A,direct	OR direct byte to accumulator	2	1
ORL A,@Ri	OR indirect RAM to accumulator	1	1
ORL A,#data	OR immediate data to accumulator	2	1
ORL direct,A	OR accumulator to direct byte	2	1
ORL direct,#data	OR immediate data to direct byte	3	2
XRL A,Rn	Exclusive OR register to accumulator	1	1
XRL A,direct	Exclusive OR direct byte to accumulator	2	1
XRL A,@Ri	Exclusive OR indirect RAM to accumulator	1	1
XRL A,#data	Exclusive OR immediate data to accumulator	2	1
XRL direct,A	Exclusive OR accumulator to direct byte	2	1
XRL direct,#data	Exclusive OR immediate data to direct byte	3	2
Boolean variable manipulation			
CLR C	Clear carry flag	1	1
CLR bit	Clear direct bit	2	1
SETB C	Set carry flag	1	1
SETB bit	Set direct bit	2	1
CPL C	Complement carry flag	1	1
CPL bit	Complement direct bit	2	1
ANL C,bit	AND direct bit to carry flag	2	2
ANL C,/bit	AND complement of direct bit to carry	2	2
ORL C,bit	OR direct bit to carry flag	2	2
ORL C,/bit	OR complement of direct bit to carry	2	2
MOV C,bit	Move direct bit to carry flag	2	1
MOV bit,C	Move carry flag to direct bit	2	2
Program and machine control			
ACALL addr11	Absolute subroutine call	2	2
LCALL addr16	Long subroutine call	3	2
RET	Return from subroutine	1	2
RETI	Return from interrupt	1	2
AJMP addr11	Absolute jump	2	2
LJMP addr16	Long jump	3	2
SJMP rel	Short jump (relative address)	2	2
JMP @A+DPTR	Jump indirect relative to the DPTR	1	2
JZ rel	Jump if accumulator is zero	2	2
JNZ rel	Jump if accumulator is not zero	2	2
JC rel	Jump if carry flag is set	2	2
JNC rel	Jump if carry flag is not set	2	2
JB bit,rel	Jump if bit is set	3	2
JNB bit,rel	Jump if bit is not set	3	2
JBC bit,rel	Jump if direct bit is set and clear bit	3	2
CJNE A,direct,rel	Compare direct byte to A and jump if not equal	3	2

Mnemonic	Description	Byte	Cycle
CJNE A,#data,rel	Compare immediate to A and jump if not equal	3	2
CJNE Rn,#data,rel	Compare immed. to reg. and jump if not equal	3	2
CJNE @Rn,#data,rel	Compare immed. to ind. and jump if not equal	3	2
DJNZ Rn,rel	Decrement register and jump in not zero	2	2
DJNZ direct,rel	Decrement direct byte and jump in not zero	3	2
NOP	No operation	1	1
Data transfer			
MOV A,Rn	Move register to accumulator	1	1
MOV A,direct*)	Move direct byte to accumulator	2	1
MOV A,@Ri	Move indirect RAM to accumulator	1	1
MOV A,#data	Move immediate data to accumulator	2	1
MOV Rn,A	Move accumulator to register	1	1
MOV Rn,direct	Move direct byte to register	2	2
MOV Rn,#data	Move immediate data to register	2	1
MOV direct,A	Move accumulator to direct byte	2	1
MOV direct,Rn	Move register to direct byte	2	2
MOV direct,direct	Move direct byte to direct byte	3	2
MOV direct,@Ri	Move indirect RAM to direct byte	2	2
MOV direct,#data	Move immediate data to direct byte	3	2
MOV @Ri,A	Move accumulator to indirect RAM	1	1
MOV @Ri,direct	Move direct byte to indirect RAM	2	2
MOV @Ri,#data	Move immediate data to indirect RAM	2	1
MOV DPTR,#data16	Load data pointer with a 16-bit constant	3	2
MOVC A,@A+DPTR	Move code byte relative to DPTR to accumulator	1	2
MOVC A,@A+PC	Move code byte relative to PC to accumulator	1	2
MOVX A,@Ri	Move external RAM (8-bit addr.) to A	1	2
MOVX A,@DPTR	Move external RAM (16-bit addr.) to A	1	2
MOVX @Ri,A	Move A to external RAM (8-bit addr.)	1	2
MOVX @DPTR,A	Move A to external RAM (16-bit addr.)	1	2
PUSH direct	Push direct byte onto stack	2	2
POP direct	Pop direct byte from stack	2	2
XCH A,Rn	Exchange register to accumulator	1	1
XCH A,direct	Exchange direct byte to accumulator	2	1
XCH A,@Ri	Exchange indirect RAM to accumulator	1	1
XCHD A,@Ri	Exchange low-order nibble indir. RAM with A	1	1

*) MOV A,ACC is not a valid instruction

jne A,#data,@ *(jump if A ! = data)*	cjne A,#data,@				
je A, #data,@ *(jump if A == data)*	add A,#low(–data) jz @	*or*	ne:	cjne A,#(data),ne jmp @ ...	
ja, jnbe A,#data,@ *(jump if A > data)*	add A,#low(–data–1) jc @	*or*	ne:	cjne A,#(data+1),ne jnc @	
jae, jnb A,#data,@ *(jump if A >= data)*	add A,#low(–data) jc @	*or*	ne:	cjne A,#(data),ne jnc @	
jb, jnae A,#data,@ *(jump if A < data)*	add A,#low,(–data) jnc @	*or*	ne:	cjne A,#(data),ne jc @	
jbe, jna A,#data,@ *(jump if A <= data)*	add A,#low(–data–1) jnc @	*or*	ne:	cjne A,#(data+1),ne jc @	
switch A <,==,> #data *(no A modification)*	cjne A,#data,ne ... jc is_below jnc is_above		ne:	; execute code if A==data ; jump if A<data ; jump if A>data or exec. code	

This paper was created by Štěpán Matějka alias Mates for anybody who needs it. Mates, Prague – Czech Republic 1998,2002.

Enjoy It! Mates

8051 Instruction Set Table

Mates (c) 1995, 2002

	0000 .0	0001 .1	0010 .2	0011 .3	0100 .4	0101 .5	0110 .6 @R0	0111 .7 @R1	1000 .8 R0	1001 .9 R1	1010 .A R2	1011 .B R3	1100 .C R4	1101 .D R5	1110 .E R6	1111 .F R7	
0	0 1c 1	1 2c 2	2 2c 3	3 1c 1	4 1c 1	5 1c 1	6 1c 1	7 1c 1	8 1c 1	9 1c 1	10 1c 1	11 1c 1	12 1c 1	13 1c 1	14 1c 1	15 1c 1	INC
	NOP	AJMP 0A8	LJMP A16	RR A	INC A	INC A8	INC @R0	INC @R1	INC R0	INC R1	INC R2	INC R3	INC R4	INC R5	INC R6	INC R7	
1	16 2c 3	17 2c 2	18 2c 3	19 1c 1	20 1c 1	21 1c 1	22 1c 1	23 1c 1	24 1c 1	25 1c 1	26 1c 1	27 1c 1	28 1c 1	29 1c 1	30 1c 1	31 1c 1	DEC
	JBC BIT,AR	ACALL 0A8	LCALL A16	RRC A	DEC A	DEC A8	DEC @R0	DEC @R1	DEC R0	DEC R1	DEC R2	DEC R3	DEC R4	DEC R5	DEC R6	DEC R7	
2	32 2c 3	33 2c 2	34 2c 1	35 1c 1	36 1c 2	37 1c 2	38 1c 1	39 1c 1	40 1c 1	41 1c 1	42 1c 1	43 1c 1	44 1c 1	45 1c 1	46 1c 1	47 1c 1	ADD
	JB BIT,AR	AJMP 1A8	RET	RL A	ADD A,#D	ADD A,A8	ADD A,@R0	ADD A,@R1	ADD A,R0	ADD A,R1	ADD A,R2	ADD A,R3	ADD A,R4	ADD A,R5	ADD A,R6	ADD A,R7	
3	48 2c 3	49 2c 2	50 2c 1	51 1c 1	52 1c 2	53 1c 2	54 1c 1	55 1c 1	56 1c 1	57 1c 1	58 1c 1	59 1c 1	60 1c 1	61 1c 1	62 1c 1	63 1c 1	ADDC
	JNB BIT,AR	ACALL 1A8	RETI	RLC A	ADDC A,#D	ADDC A,A8	ADDC A,@R0	ADDC A,@R1	ADDC A,R0	ADDC A,R1	ADDC A,R2	ADDC A,R3	ADDC A,R4	ADDC A,R5	ADDC A,R6	ADDC A,R7	
4	64 2c 2	65 2c 2	66 1c 2	67 1c 3	68 1c 2	69 1c 2	70 1c 1	71 1c 1	72 1c 1	73 1c 1	74 1c 1	75 1c 1	76 1c 1	77 1c 1	78 1c 1	79 1c 1	ORL
	JC AR	AJMP 2A8	ORL A8,A	ORL A8,#D	ORL A,#D	ORL A,A8	ORL A,@R0	ORL A,@R1	ORL A,R0	ORL A,R1	ORL A,R2	ORL A,R3	ORL A,R4	ORL A,R5	ORL A,R6	ORL A,R7	
5	80 2c 2	81 2c 2	82 1c 2	83 1c 3	84 1c 2	85 1c 2	86 1c 1	87 1c 1	88 1c 1	89 1c 1	90 1c 1	91 1c 1	92 1c 1	93 1c 1	94 1c 1	95 1c 1	ANL
	JNC AR	ACALL 2A8	ANL A8,A	ANL A8,#D	ANL A,#D	ANL A,A8	ANL A,@R0	ANL A,@R1	ANL A,R0	ANL A,R1	ANL A,R2	ANL A,R3	ANL A,R4	ANL A,R5	ANL A,R6	ANL A,R7	
6	96 2c 2	97 2c 2	98 1c 2	99 1c 3	100 1c 2	101 1c 2	102 1c 1	103 1c 1	104 1c 1	105 1c 1	106 1c 1	107 1c 1	108 1c 1	109 1c 1	110 1c 1	111 1c 1	XRL
	JZ AR	AJMP 3A8	XRL A8,A	XRL A8,#D	XRL A,#D	XRL A,A8	XRL A,@R0	XRL A,@R1	XRL A,R0	XRL A,R1	XRL A,R2	XRL A,R3	XRL A,R4	XRL A,R5	XRL A,R6	XRL A,R7	
7	112 2c 2	113 2c 2	114 2c 2	115 2c 1	116 1c 2	117 2c 3	118 1c 2	119 1c 2	120 1c 2	121 1c 2	122 1c 2	123 2c 2	124 1c 2	125 1c 2	126 1c 2	127 1c 2	MOV
	JNZ AR	ACALL 3A8	ORL C,BIT	JMP @A+DPTR	MOV A,#D	MOV A8,#D	MOV @R0,#D	MOV @R1,#D	MOV R0,#D	MOV R1,#D	MOV R2,#D	MOV R3,#D	MOV R4,#D	MOV R5,#D	MOV R6,#D	MOV R7,#D	
8	128 2c 2	129 2c 2	130 2c 2	131 2c 1	132 4c 1	133 2c 3	134 2c 2	135 2c 2	136 2c 2	137 2c 2	138 2c 2	139 2c 2	140 2c 2	141 2c 2	142 2c 2	143 2c 2	MOV
	SJMP AR	AJMP 4A8	ANL C,BIT	MOVC A,@A+PC	DIV AB	MOV A8,A8	MOV A8,@R0	MOV A8,@R1	MOV A8,R0	MOV A8,R1	MOV A8,R2	MOV A8,R3	MOV A8,R4	MOV A8,R5	MOV A8,R6	MOV A8,R7	
9	144 2c 3	145 2c 2	146 2c 2	147 2c 1	148 1c 2	149 1c 2	150 1c 1	151 1c 1	152 1c 1	153 1c 1	154 1c 1	155 1c 1	156 1c 1	157 1c 1	158 1c 1	159 1c 1	SUBB
	MOV DPTR,#D16	ACALL 4A8	MOV BIT,C	MOVC A,@A+DPTR	SUBB A,#D	SUBB A,A8	SUBB A,@R0	SUBB A,@R1	SUBB A,R0	SUBB A,R1	SUBB A,R2	SUBB A,R3	SUBB A,R4	SUBB A,R5	SUBB A,R6	SUBB A,R7	
A	208 2c 2	161 2c 2	162 1c 2	163 2c 1	164 4c 1	165	166 2c 2	167 2c 2	168 2c 2	169 2c 2	170 2c 2	171 2c 2	172 2c 2	173 2c 2	174 2c 2	175 2c 2	MOV
	ORL C,$\overline{\text{BIT}}$	AJMP 5A8	MOV C,BIT	INC DPTR	MUL AB		MOV @R0,A8	MOV @R1,A8	MOV R0,A8	MOV R1,A8	MOV R2,A8	MOV R3,A8	MOV R4,A8	MOV R5,A8	MOV R6,A8	MOV R7,A8	
B	176 2c 2	177 2c 2	178 1c 2	179 1c 1	180 2c 3	181 2c 3	182 2c 3	183 2c 3	184 2c 3	185 2c 3	186 2c 3	187 2c 3	188 2c 3	189 2c 3	190 2c 3	191 2c 3	CJNE
	ANL C,$\overline{\text{BIT}}$	ACALL 5A8	CPL BIT	CPL C	CJNE A,#D,AR	CJNE A,A8,AR	CJNE @R0,#D,AR	CJNE @R1,#D,AR	CJNE R0,#D,AR	CJNE R1,#D,AR	CJNE R2,#D,AR	CJNE R3,#D,AR	CJNE R4,#D,AR	CJNE R5,#D,AR	CJNE R6,#D,AR	CJNE R7,#D,AR	
C	192 2c 2	193 2c 2	194 1c 2	195 1c 1	196 1c 1	197 1c 2	198 1c 1	199 1c 1	200 1c 1	201 1c 1	202 1c 1	203 1c 1	204 1c 1	205 1c 1	206 1c 1	207 1c 1	XCH
	PUSH A8	AJMP 6A8	CLR BIT	CLR C	SWAP A	XCH A,A8	XCH A,@R0	XCH A,@R1	XCH A,R0	XCH A,R1	XCH A,R2	XCH A,R3	XCH A,R4	XCH A,R5	XCH A,R6	XCH A,R7	
D	208 2c 2	209 2c 2	210 1c 2	211 1c 1	212 1c 1	213 2c 3	214 1c 1	215 1c 1	216 2c 2	217 2c 2	218 2c 2	219 2c 2	220 2c 2	221 2c 2	222 2c 2	223 2c 2	DJNZ
	POP A8	ACALL 6A8	SETB BIT	SETB C	DA A	DJNZ A8,AR	XCHD A,@R0	XCHD A,@R1	DJNZ R0,AR	DJNZ R1,AR	DJNZ R2,AR	DJNZ R3,AR	DJNZ R4,AR	DJNZ R5,AR	DJNZ R6,AR	DJNZ R7,AR	
E	224 2c 1	225 2c 2	226 2c 1	227 2c 1	228 1c 1	229 1c 2	230 1c 1	231 1c 1	232 1c 1	233 1c 1	234 1c 1	235 1c 1	236 1c 1	237 1c 1	238 1c 1	239 1c 1	MOV
	MOVX A,@DPTR	AJMP 7A8	MOVX A,@R0	MOVX A,@R1	CLR A	MOV A,A8	MOV A,@R0	MOV A,@R1	MOV A,R0	MOV A,R1	MOV A,R2	MOV A,R3	MOV A,R4	MOV A,R5	MOV A,R6	MOV A,R7	
F	240 2c 1	241 2c 2	242 2c 1	243 2c 1	244 1c 1	245 1c 2	246 1c 1	247 1c 1	248 1c 1	249 1c 1	250 1c 1	251 1c 1	252 1c 1	253 1c 1	254 1c 1	255 1c 1	MOV
	MOVX @DPTR,A	ACALL 7A8	MOVX @R0,A	MOVX @R1,A	CPL A	MOV A8,A	MOV @R0,A	MOV @R1,A	MOV R0,A	MOV R1,A	MOV R2,A	MOV R3,A	MOV R4,A	MOV R5,A	MOV R6,A	MOV R7,A	

A8 ..... address 8 bits
A16 ... address 16 bits
AR .... relative address 8 bits
BIT ... bit's address
D ..... data 8 bits
D16 ... data 16 bits

8051 Instruction Set Table

Mates (c) 1995, 2002

	0000 .0	0001 .1	0010 .2	0011 .3	0100 .4	0101 .5	0110 .6 @R0	0111 .7 @R1	1000 .8 R0	1001 .9 R1	1010 .A R2	1011 .B R3	1100 .C R4	1101 .D R5	1110 .E R6	1111 .F R7	
0	0 1c 1 NOP	1 2c 2 AJMP 0A8	2 2c 3 LJMP A16	3 1c 1 RR A	4 1c 1 INC A	5 1c 1 INC A8	6 1c 1 INC @R0	7 1c 1 INC @R1	8 1c 1 INC R0	9 1c 1 INC R1	10 1c 1 INC R2	11 1c 1 INC R3	12 1c 1 INC R4	13 1c 1 INC R5	14 1c 1 INC R6	15 1c 1 INC R7	INC
1	16 2c 3 JBC BIT,AR	17 2c 2 ACALL 0A8	18 2c 3 LCALL A16	19 1c 1 RRC A	20 1c 1 DEC A	21 1c 1 DEC A8	22 1c 1 DEC @R0	23 1c 1 DEC @R1	24 1c 1 DEC R0	25 1c 1 DEC R1	26 1c 1 DEC R2	27 1c 1 DEC R3	28 1c 1 DEC R4	29 1c 1 DEC R5	30 1c 1 DEC R6	31 1c 1 DEC R7	DEC
2	32 2c 3 JB BIT,AR	33 2c 2 AJMP 1A8	34 2c 1 RET	35 1c 1 RL A	36 1c 2 ADD A,#D	37 1c 2 ADD A,A8	38 1c 1 ADD A,@R0	39 1c 1 ADD A,@R1	40 1c 1 ADD A,R0	41 1c 1 ADD A,R1	42 1c 1 ADD A,R2	43 1c 1 ADD A,R3	44 1c 1 ADD A,R4	45 1c 1 ADD A,R5	46 1c 1 ADD A,R6	47 1c 1 ADD A,R7	ADD
3	48 2c 3 JNB BIT,AR	49 2c 2 ACALL 1A8	50 2c 1 RETI	51 1c 1 RLC A	52 1c 2 ADDC A,#D	53 1c 2 ADDC A,A8	54 1c 1 ADDC A,@R0	55 1c 1 ADDC A,@R1	56 1c 1 ADDC A,R0	57 1c 1 ADDC A,R1	58 1c 1 ADDC A,R2	59 1c 1 ADDC A,R3	60 1c 1 ADDC A,R4	61 1c 1 ADDC A,R5	62 1c 1 ADDC A,R6	63 1c 1 ADDC A,R7	ADDC
4	64 2c 2 JC AR	65 2c 2 AJMP 2A8	66 1c 2 ORL A8,A	67 1c 3 ORL A8,#D	68 1c 2 ORL A,#D	69 1c 2 ORL A,A8	70 1c 1 ORL A,@R0	71 1c 1 ORL A,@R1	72 1c 1 ORL A,R0	73 1c 1 ORL A,R1	74 1c 1 ORL A,R2	75 1c 1 ORL A,R3	76 1c 1 ORL A,R4	77 1c 1 ORL A,R5	78 1c 1 ORL A,R6	79 1c 1 ORL A,R7	ORL
5	80 2c 2 JNC AR	81 2c 2 ACALL 2A8	82 1c 2 ANL A8,A	83 1c 3 ANL A8,#D	84 1c 2 ANL A,#D	85 1c 2 ANL A,A8	86 1c 1 ANL A,@R0	87 1c 1 ANL A,@R1	88 1c 1 ANL A,R0	89 1c 1 ANL A,R1	90 1c 1 ANL A,R2	91 1c 1 ANL A,R3	92 1c 1 ANL A,R4	93 1c 1 ANL A,R5	94 1c 1 ANL A,R6	95 1c 1 ANL A,R7	ANL
6	96 2c 2 JZ AR	97 2c 2 AJMP 3A8	98 1c 2 XRL A8,A	99 1c 3 XRL A8,#D	100 1c 2 XRL A,#D	101 1c 2 XRL A,A8	102 1c 1 XRL A,@R0	103 1c 1 XRL A,@R1	104 1c 1 XRL A,R0	105 1c 1 XRL A,R1	106 1c 1 XRL A,R2	107 1c 1 XRL A,R3	108 1c 1 XRL A,R4	109 1c 1 XRL A,R5	110 1c 1 XRL A,R6	111 1c 1 XRL A,R7	XRL
7	112 2c 2 JNZ AR	113 2c 2 ACALL 3A8	114 2c 2 ORL C,BIT	115 2c 1 JMP @A+DPTR	116 1c 2 MOV A,#D	117 2c 3 MOV A8,#D	118 1c 2 MOV @R0,#D	119 1c 2 MOV @R1,#D	120 1c 2 MOV R0,#D	121 1c 2 MOV R1,#D	122 1c 2 MOV R2,#D	123 2c 2 MOV R3,#D	124 1c 2 MOV R4,#D	125 1c 2 MOV R5,#D	126 1c 2 MOV R6,#D	127 1c 2 MOV R7,#D	MOV
8	128 2c 2 SJMP AR	129 2c 2 AJMP 4A8	130 2c 2 ANL C,BIT	131 2c 1 MOVC A,@A+PC	132 4c 1 DIV AB	133 2c 3 MOV A8,A8	134 2c 2 MOV A8,@R0	135 2c 2 MOV A8,@R1	136 2c 2 MOV A8,R0	137 2c 2 MOV A8,R1	138 2c 2 MOV A8,R2	139 2c 2 MOV A8,R3	140 2c 2 MOV A8,R4	141 2c 2 MOV A8,R5	142 2c 2 MOV A8,R6	143 2c 2 MOV A8,R7	MOV
9	144 2c 3 MOV DPTR,#D16	145 2c 2 ACALL 4A8	146 2c 2 MOV BIT,C	147 2c 1 MOVC A,@A+DPTR	148 1c 2 SUBB A,#D	149 1c 2 SUBB A,A8	150 1c 1 SUBB A,@R0	151 1c 1 SUBB A,@R1	152 1c 1 SUBB A,R0	153 1c 1 SUBB A,R1	154 1c 1 SUBB A,R2	155 1c 1 SUBB A,R3	156 1c 1 SUBB A,R4	157 1c 1 SUBB A,R5	158 1c 1 SUBB A,R6	159 1c 1 SUBB A,R7	SUBB
A	208 2c 2 ORL C,$\overline{\text{BIT}}$	161 2c 2 AJMP 5A8	162 1c 2 MOV C,BIT	163 2c 1 INC DPTR	164 4c 1 MUL AB	165	166 2c 2 MOV @R0,A8	167 2c 2 MOV @R1,A8	168 2c 2 MOV R0,A8	169 2c 2 MOV R1,A8	170 2c 2 MOV R2,A8	171 2c 2 MOV R3,A8	172 2c 2 MOV R4,A8	173 2c 2 MOV R5,A8	174 2c 2 MOV R6,A8	175 2c 2 MOV R7,A8	MOV
B	176 2c 2 ANL C,$\overline{\text{BIT}}$	177 2c 2 ACALL 5A8	178 1c 2 CPL BIT	179 1c 1 CPL C	180 2c 3 CJNE A,#D,AR	181 2c 3 CJNE A,A8,AR	182 2c 3 CJNE @R0,#D,AR	183 2c 3 CJNE @R1,#D,AR	184 2c 3 CJNE R0,#D,AR	185 2c 3 CJNE R1,#D,AR	186 2c 3 CJNE R2,#D,AR	187 2c 3 CJNE R3,#D,AR	188 2c 3 CJNE R4,#D,AR	189 2c 3 CJNE R5,#D,AR	190 2c 3 CJNE R6,#D,AR	191 2c 3 CJNE R7,#D,AR	CJNE
C	192 2c 2 PUSH A8	193 2c 2 AJMP 6A8	194 1c 2 CLR BIT	195 1c 1 CLR C	196 1c 1 SWAP A	197 1c 2 XCH A,A8	198 1c 1 XCH A,@R0	199 1c 1 XCH A,@R1	200 1c 1 XCH A,R0	201 1c 1 XCH A,R1	202 1c 1 XCH A,R2	203 1c 1 XCH A,R3	204 1c 1 XCH A,R4	205 1c 1 XCH A,R5	206 1c 1 XCH A,R6	207 1c 1 XCH A,R7	XCH
D	208 2c 2 POP A8	209 2c 2 ACALL 6A8	210 1c 2 SETB BIT	211 1c 1 SETB C	212 1c 1 DA A	213 2c 3 DJNZ A8,AR	214 1c 1 XCHD A,@R0	215 1c 1 XCHD A,@R1	216 2c 2 DJNZ R0,AR	217 2c 2 DJNZ R1,AR	218 2c 2 DJNZ R2,AR	219 2c 2 DJNZ R3,AR	220 2c 2 DJNZ R4,AR	221 2c 2 DJNZ R5,AR	222 2c 2 DJNZ R6,AR	223 2c 2 DJNZ R7,AR	DJNZ
E	224 2c 1 MOVX A,@DPTR	225 2c 2 AJMP 7A8	226 2c 1 MOVX A,@R0	227 2c 1 MOVX A,@R1	228 1c 1 CLR A	229 1c 2 MOV A,A8	230 1c 1 MOV A,@R0	231 1c 1 MOV A,@R1	232 1c 1 MOV A,R0	233 1c 1 MOV A,R1	234 1c 1 MOV A,R2	235 1c 1 MOV A,R3	236 1c 1 MOV A,R4	237 1c 1 MOV A,R5	238 1c 1 MOV A,R6	239 1c 1 MOV A,R7	MOV
F	240 2c 1 MOVX @DPTR,A	241 2c 2 ACALL 7A8	242 2c 1 MOVX @R0,A	243 2c 1 MOVX @R1,A	244 1c 1 CPL A	245 1c 2 MOV A8,A	246 1c 1 MOV @R0,A	247 1c 1 MOV @R1,A	248 1c 1 MOV R0,A	249 1c 1 MOV R1,A	250 1c 1 MOV R2,A	251 1c 1 MOV R3,A	252 1c 1 MOV R4,A	253 1c 1 MOV R5,A	254 1c 1 MOV R6,A	255 1c 1 MOV R7,A	MOV

A8 ..... address 8 bits
A16 ... address 16 bits
AR .... relative address 8 bits
BIT ... bit's address
D ..... data 8 bits
D16 ... data 16 bits

Code | Cycles | Bytes
INSTRUCTION