8051 Instructions

8051 has about 111 instructions. These can be grouped into the following categories

1. Arithmetic Instructions
2. Logical Instructions
3. Data Transfer instructions
4. Boolean Variable Instructions
5. Program Branching Instructions

The following nomenclatures for register, data, address and variables are used while write instructions.
A: Accumulator

B: "B" register

C: Carry bit

Rn: Register R0 - R7 of the currently selected register bank

Direct: 8-bit internal direct address for data. The data could be in lower 128bytes of RAM (00 - 7FH) or it could be in the special function register (80 - FFH).

@Ri: 8-bit external or internal RAM address available in register R0 or R1. This is used for indirect addressing mode.

#data8: Immediate 8-bit data available in the instruction.

#data16: Immediate 16-bit data available in the instruction.

Addr11: 11-bit destination address for short absolute jump. Used by instructions AJMP & ACALL. Jump range is 2 kbyte (one page).

Addr16: 16-bit destination address for long call or long jump.

Rel: 2's complement 8-bit offset (one - byte) used for short jump (SJMP) and all conditional jumps.

bit: Directly addressed bit in internal RAM or SFR

Arithmetic Instructions

Mnemonics	Description	Bytes	Instruction Cycles
ADD A, Rn	A ⟵A + Rn	1	1
ADD A, direct	A ⟵A + (direct)	2	1
ADD A, @Ri	A ⟵A + @Ri	1	1
ADD A, #data	A ⟵A + data	2	1
ADDC A, Rn	A ⟵A + Rn + C	1	1
ADDC A, direct	A ⟵ A + (direct) + C	2	1
ADDC A, @Ri	A ⟵A + @Ri + C	1	1
ADDC A, #data	A ⟵A + data + C	2	1
DA A	Decimal adjust accumulator	1	1
DIV AB	Divide A by B A ⟵quotient B ⟵remainder	1	4
DEC A	A ⟵A -1	1	1
DEC Rn	Rn ⟵Rn - 1	1	1
DEC direct	(direct) ⟵(direct) - 1	2	1
DEC @Ri	@Ri ⟵@Ri - 1	1	1
INC A	A ⟵A+1	1	1
INC Rn	Rn ⟵Rn + 1	1	1
INC direct	(direct) ⟵(direct) + 1	2	1
INC @Ri	@Ri ⟵@Ri +1	1	1
INC DPTR	DPTR ⟵DPTR +1	1	2
MUL AB	Multiply A by B A ⟵low byte (A*B) B ⟵high byte (A* B)	1	4
SUBB A, Rn	A ⟵A - Rn - C	1	1
SUBB A, direct	A ⟵A - (direct) - C	2	1
SUBB A, @Ri	A ⟵A - @Ri - C	1	1
SUBB A, #data	A ⟵A - data - C	2	1

Logical Instructions

Mnemonics	Description	Bytes	Instruction Cycles
ANL A, Rn	A ⟵A AND Rn	1	1
ANL A, direct	A ⟵A AND (direct)	2	1
ANL A, @Ri	A ⟵A AND @Ri	1	1
ANL A, #data	A ⟵A AND data	2	1
ANL direct, A	(direct) ⟵(direct) AND A	2	1
ANL direct, #data	(direct) ⟵(direct) AND data	3	2
CLR A	A⟵ 00H	1	1
CPL A	A⟵A	1	1
ORL A, Rn	A ⟵A OR Rn	1	1
ORL A, direct	A ⟵A OR (direct)	1	1
ORL A, @Ri	A ⟵A OR @Ri	2	1
ORL A, #data	A ⟵A OR data	1	1
ORL direct, A	(direct) ⟵(direct) OR A	2	1
ORL direct, #data	(direct) ⟵(direct) OR data	3	2
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 Acumulator	1	1
XRL A, Rn	A ⟵A EXOR Rn	1	1
XRL A, direct	A ⟵A EXOR (direct)	1	1
XRL A, @Ri	A ⟵A EXOR @Ri	2	1
XRL A, #data	A ⟵A EXOR data	1	1
XRL direct, A	(direct) ⟵(direct) EXOR A	2	1
XRL direct, #data	(direct) ⟵(direct) EXOR data	3	2

Data Transfer Instructions

Mnemonics	Description	Bytes	Instr. Cycles
MOV A, Rn	A ⟵Rn	1	1
MOV A, direct	A ⟵(direct)	2	1
MOV A, @Ri	A ⟵@Ri	1	1
MOV A, #data	A ⟵data	2	1
MOV Rn, A	Rn ⟵A	1	1
MOV Rn, direct	Rn ⟵(direct)	2	2
MOV Rn, #data	Rn ⟵data	2	1
MOV direct, A	(direct) ⟵A	2	1
MOV direct, Rn	(direct) ⟵Rn	2	2
MOV direct1, direct2	(direct1) ⟵(direct2)	3	2
MOV direct, @Ri	(direct)⟵ @Ri	2	2
MOV direct, #data	(direct) ⟵#data	3	2
MOV @Ri, A	@Ri ⟵A	1	1
MOV @Ri, direct	@Ri ⟵(direct)	2	2
MOV @Ri, #data	@Ri ⟵data	2	1
MOV DPTR, #data16	DPTR ⟵data16	3	2
MOVC A, @A+DPTR	A ⟵Code byte pointed by A + DPTR	1	2
MOVC A, @A+PC	A ⟵Code byte pointed by A + PC	1	2
MOVC A, @Ri	A ⟵Code byte pointed by Ri 8-bit address)	1	2
MOVX A, @DPTR	A ⟵External data pointed by DPTR	1	2
MOVX @Ri, A	@Ri ⟵A (External data - 8bit address)	1	2
MOVX @DPTR, A	@DPTR ⟵A(External data - 16bit address)	1	2
PUSH direct	(SP) ⟵(direct)	2	2
POP direct	(direct) ⟵(SP)	2	2
XCH Rn	Exchange A with Rn	1	1
XCH direct	Exchange A with direct byte	2	1
XCH @Ri	Exchange A with indirect RAM	1	1
XCHD A, @Ri	Exchange least significant nibble of A with that of indirect RAM	1	1

Boolean Variable Instructions

Mnemonics	Description	Bytes	Instruction Cycles
CLR C	C-bit ⟵0	1	1
CLR bit	bit ⟵0	2	1
SET C	C ⟵1	1	1
SET bit	bit ⟵1	2	1
CPL C	C⟵ $\overline{\text{C-bit}}$	1	1
CPL bit	bit ⟵ $\overline{\text{bit}}$	2	1
ANL C, /bit	C ⟵C . $\overline{\text{bit}}$	2	1
ANL C, bit	C ⟵C. bit	2	1
ORL C, /bit	C⟵ C + $\overline{\text{bit}}$	2	1
ORL C, bit	C ⟵C + bit	2	1
MOV C, bit	C⟵ bit	2	1
MOV bit, C	bit ⟵C	2	2

Program Branching Instructions

Mnemonics	Description	Bytes	Instr. Cycles
ACALL addr11	PC + 2 ⟶(SP) ; addr 11⟶ PC	2	2
AJMP addr11	Addr11 ⟶PC	2	2
CJNE A, direct, rel	Compare with A, jump (PC + rel) if not equal	3	2
CJNE A, #data, rel	Compare with A, jump (PC + rel) if not equal	3	2
CJNE Rn, #data, rel	Compare with Rn, jump (PC + rel) if not equal	3	2
CJNE @Ri, #data, rel	Compare with @Ri A, jump (PC + rel) if not equal	3	2
DJNZ Rn, rel	Decrement Rn, jump if not zero	2	2
DJNZ direct, rel	Decrement (direct), jump if not zero	3	2
JC rel	Jump (PC + rel) if C bit = 1	2	2
JNC rel	Jump (PC + rel) if C bit = 0	2	2
JB bit, rel	Jump (PC + rel) if bit = 1	3	2
JNB bit, rel	Jump (PC + rel) if bit = 0	3	2
JBC bit, rel	Jump (PC + rel) if bit = 1	3	2
JMP @A+DPTR	A+DPTR ⟶PC	1	2
JZ rel	If A=0, jump to PC + rel	2	2
JNZ rel	If A ≠ 0 , jump to PC + rel	2	2
LCALL addr16	PC + 3 ⟶(SP), addr16 ⟶PC	3	2
LJMP addr 16	Addr16 ⟶PC	3	2
NOP	No operation	1	1
RET	(SP) ⟶PC	1	2
RETI	(SP) ⟶PC, Enable Interrupt	1	2
SJMP rel	PC + 2 + rel ⟶PC	2	2
JMP @A+DPTR	A+DPTR ⟶PC	1	2
JZ rel	If A = 0. jump PC+ rel	2	2
JNZ rel	If A ≠ 0, jump PC + rel	2	2
NOP	No operation	1	1