4.4 INSTRUCTION SET

8051 Microcontroller have set of instruction to perform different operations. There are five group of instruction which are listed below.

- Arithmetic Instructions
- Logic Instructions
- Data Transfer Instructions
- Branch Instructions
- Bit-oriented Instructions

1.ARITHMETIC INSTRUCTION:

Arithmetic instructions perform several basic operations such as addition, subtraction, division, multiplication etc. After execution, the result is stored in the first operand.

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 from A with borrow	1	1
SUBB A,direct	Subtract direct byte from A with borrow	2	1
SUBB A,@Ri	Subtract indirect RAM from A with borrow	1	1
SUBB A,#data	Subtract immediate data from A with 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	1	4
DIV AB	Divide A by B	1	4
DA A	Decimal adjust accumulator	1	1

[Source: Mohamed Ali Mazidi, Janice Gillispie Mazidi, Rolin McKinlay, "The 8051Microcontroller and Embedded Systems: Using Assembly and C", Second Edition, Pearson Education,2011]

1. ADD A,Rn;

Adds the register Rn to the accumulator

Description:

Instruction adds the register Rn (R0-R7) to the accumulator. After addition, the result is stored in the accumulator

Before execution:

A=2Eh R4=12h

After execution:

A=40h R4=12h

2. ADD A,@Ri-

Adds the indirect RAM to the accumulator.

Ri: Register R0 or R1

Description:

Instruction adds the indirect RAM to the accumulator. Address of indirect RAM is stored in the Ri register (R0 or R1). After addition, the result is stored in the accumulator.

Register address:

R0=4Fh

Before execution:

A= 16h SUM= 33h

After execution :

A= 49h

3. ADDA,#DATA

Data: constant within 0-255 (0-FFh)

Description:

Instruction adds data (0-255) to the accumulator. After addition, the result is stored in the accumulator.

ADD A,#33h

Before execution:

A= 16h

After execution:

A= 49h)

4. SUBB A,direct-

Subtracts the direct byte from the accumulator witha borrow

Direct: arbitrary register with address 0-255(0-FFh)

Description:

Instruction subtracts the direct byte from the accumulator with a borrow. If the higher bit is subtracted from the lower bit then the carry flag is set. As it is direct addressing, the direct byte can be any SFRs or general-purpose register with address 0-7Fh. (0-127 dec.). The result is stored in the accumulator.

SUBB A,Rx

Before execution:

A=C9h, DIF=53h, C=0

After execution:

A=76h, C=0

5. INC A - Increments the accumulator by1

Description:

This instruction increments the value in the accumulator by 1. If the accumulator includes the number 255, the result of the operation will be 0.

Before execution:

A=E4h

After execution:

A=E5h

6. DEC A - Decrements the accumulator by1

Description: Instruction decrements the value in the accumulator by 1. If there is a 0 in the accumulator, the result of the operation is FFh. (255 dec.)

Syntax: DEC A;

Byte: 1 (instruction code);

STATUS register flags:

No flags are affected;

Before execution:

A=E4h

After execution:

A=E3h

7. DIV AB - Divides the accumulator by the registerB

Description:

Instruction divides the value in the accumulator by the value in the B register. After division the integer part of result is stored in the accumulator while the register contains the remainder. In case of dividing by 1, the flag OV is set and the result of division is unpredictable. The 8-bit quotient is stored in the accumulator and the 8-bit remainder is stored in the B register.

Before execution:

A=FBh (251dec.) B=12h (18 dec.)

After execution:

A=0Dh (13dec.) B=11h (17dec.)

13·18 + 17 =251

8. DA A - Decimal adjust accumulator

Description: Instruction adjusts the contents of the accumulator to correspond to a BCD number after two BCD numbers have been added by the ADD and ADDC instructions. The result in form of two 4-digit BCD numbers is stored in the accumulator.

Before execution:

A=56h (01010110) 56BCD

B=67h (01100111)67BCD

After execution:

A=BDh (10111101)

After BCD conversion:

A=23h (00100011), C=1 (Overflow)

(C+23=123) = 56+67

9. MUL AB - Multiplies A andB

Description: Instruction multiplies the value in the accumulator with the value in the B register. The low-order byte of the 16-bit result is stored in the accumulator, while the high byte remains in the B register. If the result is larger than 255, the overflow flag is set. The carry flag is not affected.

Before execution:

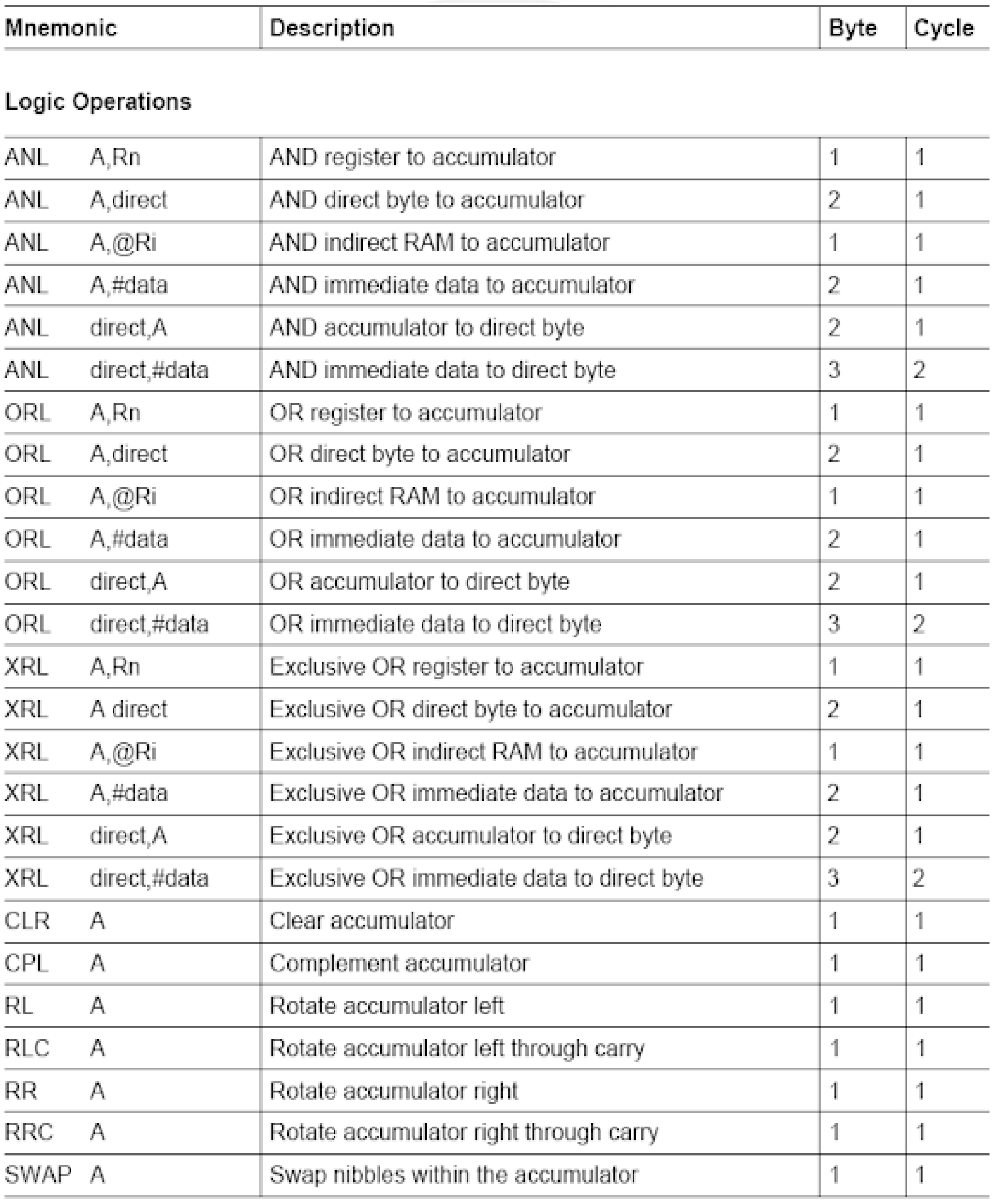
A=80 (50h) B=160 (A0h)

After execution:

A=0 B=32h A·B=80·160=12800 (3200h)

2.LOGICAL INSTRUCTION OF 8051 MICRO-CONTROLLER

Logic instructions perform logic operations upon corresponding bits of two registers. After execution, the result is stored in the first operand.



Mnemonic		Description	Byte	Cycle
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
CLR	A	Clear accumulator	1	1
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

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Mazidi, Janice Gillispie Mazidi, Rolin McKinlay]

ANL A,Rn - AND register to the accumulator A: accumulator Rn: any R register(R0-R7)

Instruction performs logic AND operation between the accumulator and Rn register. The result is stored in the accumulator.

Syntax:

ANL A,Rn

Before execution:

A= C3h (11000011 Bin.) R5= 55h (01010101 Bin.)

After execution:

A= 41h (01000001 Bin.)

ORL A,Rn - OR register to the accumulator Rn: any R register (R0-R7)

Instruction performs logic OR operation between the accumulator and Rn register. The result is stored in the accumulator.

Syntax:

ORLA,Rn

Before execution:

A= C3h (11000011 Bin.) R5= 55h (01010101 Bin.)

After execution:

A= D7h (11010111 Bin.)

XRL A,Rn - Exclusive OR register to accumulator Rn: any R register (R0-R7)

Instruction performs exclusive OR operation between the accumulator and the Rn register. The result is stored in the accumulator.

Syntax:

XRL A,Rn

Before execution:

A= C3h (11000011 Bin.) R3= 55h (01010101 Bin.)

After execution**:**

A= 96h (10010110 Bin.)

CLR A - Clears the accumulator A: accumulator

Instruction clears the accumulator.

Syntax:

    CLR A

After execution:

A=0

CPL A - Complements the accumulator

Instruction complements all the bits in the accumulator (1==>0, 0==>1).

Syntax:

    CPL A

Before execution:

A=(00110110)

After execution:

A=(11001001)

RL A - Rotates the accumulator one bit left A: accumulator

Eight bits in the accumulator are rotated one bit left, so that the bit 7 is rotated into the bit 0 position.

Syntax:

    RL A

Before execution:

A= C2h (11000010 Bin.)

After execution:

A=85h (10000101 Bin.)

RR A - Rotates the accumulator one bit right

A: accumulator All eight bits in the accumulator are rotated one bit right so that the bit 0 is rotated into the bit 7 position.

Syntax:

    RR A

Before execution:

A= C2h (11000010Bin.)

After execution:

A= 61h (01100001Bin.)

RLC A - Rotates the accumulator one bit left through the carry flag A: accumulator

All eight bits in the accumulator and carry flag are rotated one bit left. After this operation, the bit 7 is rotated into the carry flag position and the carry flag is rotated into the bit 0 position.

Syntax:

RLC A

Before execution:

A= C2h(11000010 Bin.) C=0

After execution:

A= 85h(10000100Bin.) C=1

RRC A - Rotates the accumulator one bit right through the carry flag A: accumulator

All eight bits in the accumulator and carry flag are rotated one bit right. After this operation, the carry flag is rotated into the bit 7 position and the bit 0 is rotated into the carry flag position.

Syntax:

RRC A

Before execution:

A= C2h(11000010 Bin.) C=0

After execution:

A= 61h(01100001Bin.) C=0

SWAP A - Swaps nibbles within the accumulator

A: accumulator

A nibble refers to a group of 4 bits within one register (bit0-bit3 and bit4-bit7). This instruction interchanges high and low nibbles of the accumulator.

Syntax:

SWAPA

Before execution:

A=E1h (11100001)bin.

After execution:

A=1Eh (00011110)bin.

3. DATA TRANSFER INSTRUCTION OF 8051

These instructions are used to copy the content of source operand to Destination operand

Mnemonic		Description	Byte	Cycle
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 with accumulator	1	1
XCH	A,direct	Exchange direct byte with accumulator	2	1
XCH	A,@Ri	Exchange indirect RAM with accumulator	1	1
XCHD	A,@Ri	Exchange low-order nibble indir. RAM with A	1	1

*) MOV A,ACC is not a valid instruction

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Mazidi, Janice Gillispie Mazidi, Rolin McKinlay]

MOV A,Rn - Moves the Rn register to the accumulator

The instruction moves the Rn register to the accumulator. The Rn register is not affected.

Syntax

MOV A,Rn

Beforeexecution:

R3=58h

After execution:

R3=58h A=58h

MOV A,@Ri - Moves the indirect RAM to the accumulator

Instruction moves the indirectly addressed register of RAM to the accumulator. The register address is stored in the Ri register (R0 or R1). The result is stored in the accumulator. The register is not affected.

Syntax:

MOV A,@Ri

Register Address SUM=F2h R0=F2h

Before execution:

SUM=58h

After execution:

A=58h SUM=58h

MOV A,#data - Moves the immediate data to the accumulator

Instruction moves the immediate data to the accumulator.

Syntax:

MOV A, #28

After execution:

A=28h

MOV direct,@Ri - Moves the indirect RAM to the direct byte

Instruction moves the indirectly adressed register of RAM to the direct byte. The register is not affected.

Syntax:

MOV Rx,@Ri

Register Address SUM=F3

Before execution:

SUM=58h R1=F3

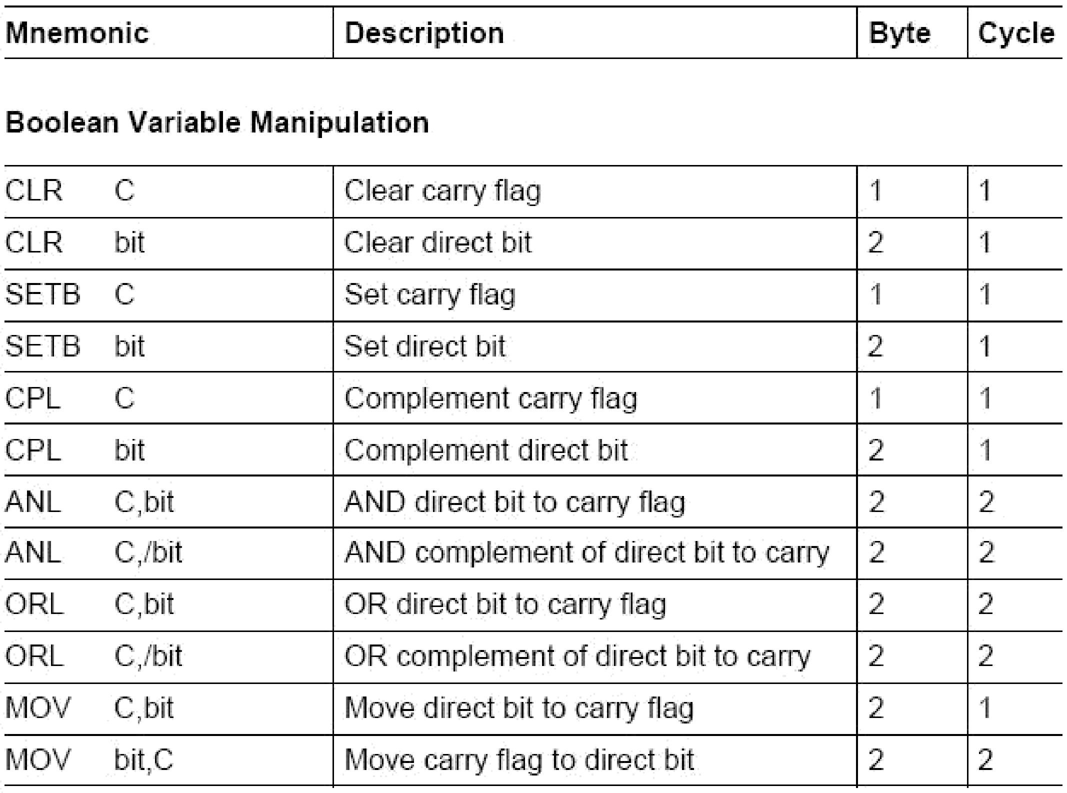
After execution:

SUM=58h TEMP=58h

MOVC A,@A+DPTR - Moves the code byte relative to the DPTR to the accumulator Instruction first adds the 16-bit DPTR register to the accumulator. The result of addition is then used as a memory address from which the 8-bit data is moved to the accumulator.

4. BIT-ORIENTED INSTRUCTIONS

Similar to logic instructions, bit-oriented instructions perform logic operations. The difference is that these are performed upon single bits.



Mnemonic	Description	Byte	Cycle
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

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Mazidi, Janice Gillispie Mazidi, Rolin McKinlay , pg.no.29]

ANL C,bit - AND direct bit to the carry flag C: Carry flag

Bit: any bit of RAM,

Instruction performs logic AND operation between the direct bit and the carry flag.

Syntax:

ANL C,bit

Before execution:

ACC= 43h (01000011 Bin.)C=1

After execution:

ACC= 43h(01000011 Bin.) C=0

CLR C - clears the carry flag

C: Carry flag, Instruction clears the carry flag. After execution: C=0

CLR bit - clears the direct bit

Bit: any bit of RAM, Instruction clears the specified bit.

Syntax:

CLR P0.3

Before execution:

P0.3=1 (input pin)

After execution:

P0.3=0 (output pin)

CPL bit - Complements the direct bit

Bit: any bit of RAM, Instruction complements the specified bit of RAM (0==>1, 1==>0).

Syntax:

CPL P0.3

Before execution:

P0.3=1 (input pin)

After execution:

P0.3=0 (output pin)

CPL C - Complements the carry flag

C: Carry flag, Instruction complements the carry flag (0==>1, 1==>0).

Syntax:

CPL C

Before execution:

C=1

After execution:

C=0

MOV bit,C - Moves the carry flag to the direct bit C: Carry flag, Bit: any bit ofRAM Instruction moves the carry flag to the direct bit. After executing the instruction, the carry flag is not affected.

Syntax:

MOVP1.2,C

After execution:

If C=0 P1.2=0 If C=1 P1.2=1

MOV C,bit - Moves the direct bit to the carryflag C: Carry flag, Bit: any bit of RAM Instruction moves the direct bit to the carry flag. After executing the instruction, the bit is not affected.

Syntax:

MOV C, P1.4

After execution:

If P1.4=0 C=0 If P1.4=1 C=1

SETB C - Sets the carry flag

C: Carry flag, Instruction sets the carry flag.

Syntax:

SETB C

After execution: C=1

SETB bit - Sets the direct bit

Bit: any bit of RAM

Instruction sets the specified bit. The register containing that bit must belong to the group of the so called bit addressable registers.

Syntax:

SETB P0.1

Before execution:

P0.1 = 34h (00110100) pin 1 is configured as an output

After execution:

P0.1 = 35h (00110101) pin 1 is configured as an inputs

5. BRANCH INSTRUCTION OF 8051 CONTROLLER

There are two kinds of branch instructions:

Unconditional jump instructions:

upon their execution a jump to a new location from where the program continues execution is executed.

Conditional jump instructions:

a jump to a new program location is executed only if a specified condition is met. Otherwise, the program normally proceeds with the next instruction.

Mnemonic		Description	Byte	Cycle
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 iump	3	2
SJMP	rel	Short jump (relative addr.)	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 direct bit is set	3	2
JNB	bit,rel	Jump if direct 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
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	@Ri,#data,rel	Compare immed. to ind. and jump if not equal	3	2
DJNZ	Rn,rel	Decrement register and jump if not zero	2	2
DJNZ	direct,rel	Decrement direct byte and jump if not zero	3	2
NOP		No operation	1	1

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Mazidi, Janice Gillispie Mazidi, Rolin McKinlay]