Instruksi Mikrokontroler

Mikrokontroler MCS-51 memiliki 256 perangkat instruksi. Seluruh instruksi dapat dikelompokkan dalam 4 bagian yang meliputi instruksi 1 byte sampai 4 byte. Apabila frekuensi clock mikrokontroler yang digunakan adalah 12 Mhz, kecepatan pelaksanaan instruksi akan bervariasi dari 1 sampai 4 mikrodetik.

Instruksi yang dimiliki oleh mikrokontroler MCS-51 pada dasarnya digolongkan menjadi :

- instruksi operasi aritmatika
- operasi logika
- transfer data
- operasi manipulasi boolean
- perintah percabangan.

ADD (Add Immediate Data)

Operand : A Akkumulator
: data - 256 <= data <= + 255
Format : ADD A, #data
Operasi : (A) <- (A) + data
Keterangan : instruksi ini menambah 8 bit data langsung ke dalam isi akumulator dan menyimpan hasilnya pada akumulator

ADD (Add Indirect Address)

Operand : A Akkumulator
: Rr Register 0 <= r <= 1
Format : ADD A, @Rr
Operasi : (A) <- (A) + ((Rr))
Keterangan : instruksi ini menambah isi data memori yang lokasinya ditunjukkan oleh nilai register r ke isi akumulator dan menyimpan hasilnya dalam akumulator
Contoh : ADD A, @R1

ADD (Add Register)

Operand : A Akkumulator
: Rr Register 0 - 7
Format : ADD A, Rr
Operasi : (A) <- (A) + (Rr)
Keterangan : instruksi ini menambah isi register r dengan isi akumulator dan menyimpan hasilnya dalam akumulator

Contoh : ADD A, R6

ADD (Add Memori)

Operand : A Akkumulator
: Alamat data 0 <= Alamat data <= 256
Format : ADD A, Alamat data
Keterangan : instruksi ini menambah isi alamat data ke isi kumulator dan menyimpan hasilnya dalam akumulator
Contoh : ADD A, 30H

ADDC (Add Carry Plus Immediate Data To Accumulator)

Operand : A Akkumulator
: data -256 <= data <= +255
Format : ADDC A, # data
Operasi : (A) <- (A) + (C) + data
Keterangan : instruksi ini menambah isi carry flag (0 atau 1) ke dalam isi akumulator.Data langsung 8 bit ditambahkan ke akumulator.
Contoh : ADDC A,#0AFH

ADDC (Add Carry Plus Indirect Address To Accumulator)

Operand : A Akkumulator
: data Register 0 <= r <= 1
Format : ADDC A, @Rr
Operasi : (A) <- (A) + (C) + ((Rr))
Keterangan : instruksi ini menambah isi carry flag (0 atau 1) dengan isi akumulator. Isi data memori pada lokasi yang ditunjukkan oleh register Rr ditambahkan dan hasilnya disimpan dalam akumulator
Contoh : ADDC A,@R1

ADDC (Add Carry Plus Register To Accumulator)

Operand : A Akkumulator
: data Register 0 <= r <= 7
Format : ADDC A, Rr
Operasi : (A) <- (A) + (C) + (Rr)
Keterangan : instruksi ini menambah isi carry flag dengan isi akumulator. Isi register r ditambahkan dan hasilnya disimpan dalam akumulator
Contoh : ADDC A,R7

ADDC (Add Carry Plus Memory To Accumulator)

Operand : A Akkumulator
: Alamat data 0 <= Alamat data <= 255
Format : ADDC A, Alamat data
Operasi : (A) <- (A) + (C) + (Alamat data)
Keterangan : instruksi ini menambah isi carry flag dengan isi akumulator. Isi dari alamat data tertentu ditambahkan pula dan hasilnya disimpan dalam akumulator
Contoh : ADDC A,30H

AJMP (Absolute Jump Within 2K byte Page)

Operand : Alamat kode
Format : AJMP alamat kode
Operasi : (PC) <- (PC) + 2
: (PC) 0-10 <- page address
Keterangan : instruksi ini meletakkan bagian bawah 11 bit dari pencacah program dengan 11 bit alamat yang dikodekan.

ANL (Logical AND Immediate Data to Accumulator)

Operand : A Akkumulator
: data -256 <= data <= +255
Format : ANL A, #data
Operasi : (A) <- (A) AND data
Keterangan : instruksi ini meng-AND kan data 8 bit secara langsung dengan isi akumulator
Contoh : ANL A,#00001000B

ANL (Logical AND Indirect Address to Accumulator)

Operand : A Akkumulator
: data Register 0 <= r <= 1
Format : ANL A, @Rr
Operasi : (A) <- (A) AND ((Rr))
Keterangan : instruksi ini meng-AND kan isi memori yang lokasinya ditunjukkan oleh isi register r dengan isi akumulator

Contoh : ANL A,@R0

ANL (Logical AND Register to Accumulator)

Operand : A Akkumulator

: data 0 <= Rr <= 7

Format : ANL A, Rr

Operasi : (A) <- (A) AND (Rr)

Keterangan : instruksi ini meng-AND kan isi register r dengan isi akumulator

ANL (Logical AND Memory to Accumulator)

Operand : A Akkumulator

: Alamat data 0 <= Alamat data <= 255

Format : ANL A, Alamat data

Operasi : (A) <- (A) AND (Alamat data)

Keterangan : instruksi ini meng-AND kan isi alamat data dengan isi akumulator

Contoh : ANL A,35H

ANL (Logical AND Bit to Carry Flag)

Operand : C Carry flag

: Alamat bit 0 <= alamat bit <= 255

Format : ANL C, Alamat bit

Operasi : (C) <- (C) AND (Alamat bit)

Keterangan : instruksi ini meng-AND kan isi alamat bit tertentu dengan isi carry flag. Jika keduanya 1 maka hasilnya 1, selainitu hasilnya 0. Hasilnya ditempatkan pada carry flag.

Contoh : ANL C, 37.3

ANL (Logical AND Complement of Bit to Carry Flag)

Operand : C Carry flag

: Alamat bit 0 <= alamat bit <= 255

Format : ANL C, /alamat bit

Operasi : (C) <- (C) AND NOT (alamat bit)

Keterangan : instruksi ini meng-AND kan hasil komplemen isi alamat bit tertentu dengan isi carry flag. Hasilnya ditempatkan pada carry flag. Isi alamt bit semula tidak berubah.

Contoh : ANL A,/40.5

ANL (Logical AND Immediate Data to Memory)

Operand : Alamat data 0 <= alamat data <= 255

: data -256 <= data <= +255

Format : ANL Alamat data, #data

Operasi : (Alamat data) <- (Alamat data) AND data

Keterangan : instruksi ini meng-AND kan data 8 bit secara langsung dengan isi alamat data tertentu. Hasilnya akan disimpan dalam memori data pada alamat tersebut.

Contoh : ANL 57H,#01H

ANL (Logical AND Accumulator to Memory)

Operand : Alamat data 0 <= Alamat data <= 255

: A Akumulator

Format : ANL Alamat data, A

Operasi : (Alamat data) <- (Alamat data) AND A

Keterangan : instruksi ini meng- AND kan isi akumulator dengan isi alamat data tertentu. Hasilnya disimpan dalam memori data pada alamat yang bersangkutan

Contoh : ANL 10H,A

CALL (Generic Call)

Operand : Alamat kode

Format : Call alamat kode

Keterangan : instruksi ini akan ditranslasikan ke ACALL atau LCALL

CJNE (Compare Indirect Address to Immediate Data)

Jump if Not Equal

Operand : Rr Register 0<= r <= 1

: data -256 <= data <= +255

: Alamat kode

Format : CJNE @Rr, #data, alamat kode

Operasi : (PC) <- (PC) + 3

: Jika ((Rr)) <> data, maka (PC) <- (PC) + offset relatif

: Jika ((Rr)) < data, maka (C) <- 1,

: lainnya (C) <- 0

Keterangan : instruksi ini akan membandingkan data langsung dengan lokasi memori yang dialamati oleh register r. Apabila tidak sama, eksekusi akan menuju ke alamat kode. Bila sama, instruksi selanjutnya yang akan dijalankan. Jika data langsung lebih besar dari isi alamat data tertentu, carry flag akan di - set menjadi 1.

Contoh : CJNE @R1, #01H, 0009H

CJNE (Compare Immediate Data to Accumulator)

Jump if Not Equal

Operand : A Akumulator

: data -256 <= data <= +255

: Alamat kode

Format : CJNE A, #data, Alamat kode

Operasi : (PC) <- (PC) + 3

: Jika (A) <> data, maka (PC) <- (PC) + offset relatif

: Jika (A) < data, maka (C) <- 1,

: lainnya (C) <- 0

Keterangan : instruksi ini akan membandingkan data langsung dengan isi akumulator. Apabila tidak sama, eksekusi akan menuju ke alamat kode. Bila sama, instruksi selanjutnya yang akan dijalankan. Jika data langsung lebih besar dari isi alamat data tertentu, carry flag akan di - set menjadi 1.

CJNE (Compare Memory to Accumulator)

Jump if Not Equal

Operand : A Akumulator

: Alamat data 0 <= Alamat data <= 255

: Alamat kode

Format : CJNE A, Alamat data, alamat kode

Operasi : (PC) <- (PC) + 3

: Jika (A) <> Alamat data, maka (PC) <- (PC) + offset relatif

: Jika (A) < Alamat data, maka (C) <- 1,

: lainnya (C) <- 0

Keterangan : instruksi ini akan membandingkan isi lokasi memori tertentu dengan isi akumulator. Apabila tidak sama, eksekusi akan menuju ke alamat kode. Bila sama, instruksi selanjutnya yang akan dijalankan. Jika data langsung lebih besar dari isi akumulator, carry flag akan di - set menjadi 1.

CJNE (Compare Immediate Data to Register)

Jump if Not Equal

Operand : Rr Register 0<= r <= 7

: data -256 <= data <= +255

: Alamat kode

Format : CJNE Rr, #data, Alamat kode

Operasi : (PC) <- (PC) + 3
: Jika (Rr) <> data, maka (PC) <- (PC) + offset relatif
: Jika (Rr) < data, maka (C) <- 1,
: lainnya (C) <- 0

Keterangan : instruksi ini akan membandingkan data langsung deng-an isi register r. Apabila tidak sama, eksekusi akan menuju ke alamat kode. Bila sama, instruksi selanjutnya yang akan dijalankan. Jika data langsung lebih besar dari isi register, carry flag akan di - set menjadi 1.

CLR (Clear Accumulator)

Operand : A Akumulator

Format : CLR A

Operasi : (A) <- 0

Keterangan : instruksi ini akan me-reset akumulator menjadi 00H

Contoh : CLR A

CLR (Clear Carry Flag)

Operand : C

Format : CLR C

Operasi : (A) <- 0

Keterangan : instruksi ini akan me-reset carry flag menjadi 00H

Contoh : CLR C

CLR (Clear Bit)

Operand : Alamat bit 0 <= Alamat bit <= 255

Format : CLR Alamat bit

Operasi : (Alamat bit) <- 0

Keterangan : instruksi ini akan me-reset alamat bit menjadi 00H

Contoh : CLR 40.5

CPL (Complement Accumulator)

Operand : A Akumulator

Format : CPL A

Operasi : (A) <-NOT (A)

Keterangan : instruksi ini akan mengkomplemen isi akumulator

Contoh : CPL A

CPL (Complement_Carry Flag)

Operand : CPL C

Format : CPL C

Operasi : (C) <-NOT (C)

Keterangan : instruksi ini akan mengkomplemen isi carry flag

Contoh : CPL C

CPL (Complement_Bit)

Operand : Alamat bit 0 <= Alamat bit <= 255

Format : CPL Alamat bit

Operasi : (Alamat bit) <-NOT (Alamat bit)

Keterangan : instruksi ini akan mengkomplemen isi suatu alamat bit

Contoh : CPL 33.7

DA (Decimal Adjust_Accumulator)

Operand : A Akumulator

Format : DA A

Keterangan : instruksi ini mengatur isi akumulator ke padanan BCD nya, setelah penambahan dua angka BCD. Jika auxiliary carry flag 1 atau isi nibble bawah (bit 0 - 3) dari akumulator lebih tinggi dari 9, isi akumulator akan ditambah 6. Jika carry flag di set sebelum atau sesudah penambahan atau isi nibble atas (bit 4 - 7) lebih tinggi dari 9, isi akumulator akan ditambah 60H.

DEC (Decrement Indirect Address)

Operand : Rr Register 0 <= r <= 1

Format : DEC @Rr

Operasi : ((Rr)) <-((Rr)) -1

Keterangan : instruksi ini akan mengurangi isi lokasi memori yang ditunjukkan oleh register r dengan 1. Hasilnya disimpan dalam lokasi tersebut.

DEC (Decrement Accumulator)

Operand : A Akumulator

Format : DEC A

Operasi : (A) <-(A) -1

Keterangan : instruksi ini akan mengurangi isi akumulator dengan 1. Hasilnya disimpan dalam akumulator

Contoh : DEC A

DEC (Decrement Register)

Operand : Rr Register 0 <= r <= 7

Format : DEC Rr

Operasi : (Rr) <-(Rr) -1

Keterangan : instruksi ini akan mengurangi isi register r dengan 1. Hasilnya disimpan dalam register r

Contoh : DEC R7

DEC (Decrement Memory)

Operand : Alamat data 0 <= Alamat data <= 255

Format : DEC Alamat data

Operasi : (Alamat data) <-(Alamat data) -1

Keterangan : instruksi ini akan mengurangi isi alamat data dengan 1. Hasilnya disimpan dalam alamat tersebut.

DIV (Divide Accumulator by B)

Operand : AB Pasangan register

Format : DIV AB

Operasi : (AB) <-(A)/(B)

Keterangan : instruksi ini membagi isi akumulator dengan isi register B. Kedua operand adalah bilangan integer tak bertanda. Akumulator berisi hasil bagi, register B berisi sisa pembagian

Contoh : MOV B, #5H

: DIV AB

DJNZ (Decrement Register And Jump If Not Zero)

Operand : Rr Register 0 <= r <= 7

: Alamat kode

Format : DJNZ Rr, Alamat kode

Operasi : (PC) <-(PC) + 2

: (Rr) <- (Rr) -1

: Jika (Rr) <> 0, maka (PC) <- (PC) + offset relatif

Keterangan : instruksi ini mengurangi register r dengan 1 dan me-nempatkan hasilnya pada register tertentu. Jika hasilnya sudah 0, instruksi selanjutnya yang akan dieksekusi. Jika belum 0, eksekusi akan menuju ke alamat kode.

DJNZ (Decrement Memory And Jump If Not Zero)

Operand : Alamat data 0 <= Alamat data <= 255
: Alamat kode

Format : DJNZ Alamat data, Alamat kode

Operasi : (PC) <-(PC) + 3
: (Alamat data) <- (Alamat data) -1
: Jika (Alamat data) <> 0, maka (PC) <- (PC) + offset relatif

Keterangan : instruksi ini mengurangi alamat data dengan 1 dan me-nempatkan hasilnya pada alamat tersebut. Jika hasilnya sudah 0, instruksi selanjutnya yang akan dieksekusi. Jika belum 0, eksekusi akan menuju ke alamat kode.

INC (Increment Indirect Address)

Operand : Rr Register 0 <= r <= 1

Format : INC @Rr

Operasi : ((Rr)) <-((Rr)) +1

Keterangan : instruksi ini akan menambah isi lokasi memori yang ditunjukkan oleh register r dengan 1. Hasilnya disimpan dalam lokasi tersebut.

INC (Increment Accumulator)

Operand : A Akumulator

Format : INC A

Operasi : (A) <-(A) +1

Keterangan : instruksi ini akan menambah isi akumulator dengan 1. Hasilnya disimpan dalam akumulator

Contoh : INC A

INC (Increment Data Pointer)

Operand : DPTR Data Pointer

Format : INC DPTR

Operasi : (DPTR) <-(DPTR) +1

Keterangan : instruksi ini akan menambah isi data pointer dengan 1. Hasilnya disimpan pada data poniter

Contoh : INC DPTR

INC (Increment Register)

Operand : Rr Register 0 <= r <= 7

Format : INC Rr

Operasi : (Rr) <-(Rr) +1

Keterangan : instruksi ini akan menambah isi register r dengan 1. Hasilnya disimpan dalam register tersebut

Contoh : INC R7

INC (Increment Memory)

Operand : Alamat data 0 <= Alamat data <= 255

Format : INC Alamat data

Operasi : (Alamat data) <-(Alamat data) +1

Keterangan : instruksi ini akan menambah isi alamat data dengan 1. Hasilnya disimpan dalam alamat tersebut.

Contoh : INC 37H

JB (Jump if Bit is Set)

Operand : Alamat bit 0 <= Alamat bit <= 255

: Alamat kode

Format : JB Alamat bit, alamat kode

Operasi : (PC) <- (PC) + 3

: Jika (Alamat bit) = 1, maka (PC) <- (PC) + offset relatif

Keterangan : instruksi ini akan menguji suatu alamat bit. Jika berisi 1, ekesekusi akan menuju alamat kode. Jika tidak, instruksi selanjutnya akan dieksekusi. Pencacah program akan dinaikkan pada instruksi selanjutnya. Jika pengujian berhasil, offset relatif akan ditambahkan ke pencacah program yang telah dinaikkan dan instruksi pada alamat ini akan dieksekusi.

JBC (Jump And Clear if Bit is Set)

Operand : Alamat bit 0 <= Alamat bit <= 255

: Alamat kode

Format : JBC Alamat bit, alamat kode

Operasi : (PC) <- (PC) + 3

: Jika (Alamat bit) = 1, maka (alamat bit) <- 0

: (PC) <- (PC) + offset relatif

Keterangan : instruksi ini akan menguji suatu alamat bit. Jika berisi 1, bit tersebut akan diubah menjadi 0 dan eksekusi akan menuju ke alamat kode. Jika berisi 0, instruksi selanjutnya yang akan dieksekusi.

JC (Jump if Carry is Set)

Operand : Alamat kode

Format : JCC Alamat kode

Operasi : (PC) <- (PC) + 2

: Jika (PC) = 1, maka (PC) <- (PC) + offset relatif

Keterangan : instruksi ini akan menguji isi carry flag. Jika berisi 1, eksekusi akan menuju ke alamat kode. Jika berisi 0, instruksi selanjutnya yang akan dieksekusi.

JMP (Generic Jump)

Operand : Alamat kode 0 <= Alamat kode <= 65535

Format : JMP Alamat kode

Keterangan : instruksi ini akan diubah menjadi SJMP, AJMP atau LJMP

JMP (Jump to Sum of Accumulator and Data Pointer)

Operand : A Akumulator

: DPTR Data Pointer

Format : JMP @A + DPTR

Operasi : (PC) <- (A) + (DPTR)

Keterangan : instruksi ini akan menambah isi akumulator dengan isi data pointer dan meloncat ke alamat kode sesuai hasil penjumlahan

Contoh : JMP @A + DPTR

JNB (Jump if Bit is Not Set)

Operand : Alamat bit

: Alamat kode

Format : JNB Alamat bit, alamat kode

Operasi : (PC) <- (PC) + 3

Jika (alamat bit) = 0, maka (PC) <- (PC) + offset relatif

Keterangan : instruksi ini akan menguji suatu alamat bit. Jika isinya 0, eksekusi akan menuju ke alamat kode. Jika isinya 1, instruksi selanjutnya yang akan dieksekusi.

JNC (Jump if Carry is Not Set)

Operand : Alamat kode

Format : JNC Alamat kode

Operasi : (PC) <- (PC) + 2
Jika (C) = 0, maka (PC) <- (PC) + offset relatif
Keterangan : instruksi ini akan menguji isi carry flag. Jika isinya 0, eksekusi akan menuju ke alamat kode. Jika isinya 1, instruksi selanjutnya yang akan dieksekusi.

JNZ (Jump if Accumulator is Not Zero)

Operand : Alamat kode
Format : JNZ Alamat kode
Operasi : (PC) <- (PC) + 2
Jika (A) <> 0, maka (PC) <- (PC) + offset relatif
Keterangan : instruksi ini akan menguji isi akumulator. Jika tidak sama dengan 0, eksekusi akan menuju ke alamat kode. Jika sama dengan 0, instruksi selanjutnya yang akan dieksekusi.

JZ (Jump if Accumulator is Zero)

Operand : Alamat kode
Format : JZ Alamat kode
Operasi : (PC) <- (PC) + 2
Jika (A) = 0, maka (PC) <- (PC) + offset relatif
Keterangan : instruksi ini akan menguji isi akumulator. Jika sama dengan 0, eksekusi akan menuju ke alamat kode. Jika tidak sama dengan 0, instruksi selanjutnya yang akan dieksekusi.

LCALL (Long Call)

Operand : Alamat kode 0 <= Alamat kode <= 65535
Format : LCALL Alamat kode
Operasi : (PC) <= (PC) + 3
: (SP) <= (SP) + 1
: ((SP)) <= (PC Low)
: (SP) <= (SP) + 1
: ((SP)) <= (PC high)
: (PC) <= Alamat kode
Keterangan : instruksi ini akan menyimpan pencacah program pada stack dan eksekusi akan menuju ke alamat kode.

LJMP (Long Jump)

Operand : Alamat kode 0 <= Alamat kode <= 65535
Format : LJMP Alamat kode
Operasi : (PC) <- Alamat kode

Keterangan : instruksi ini akan menuju alamat kode

MOV (Move Immediate Data to Indirect Address)

Operand : Rr Register 0 <= r <= 1
: data - 256 <= data <= + 255
Format : MOV @Rr, #data
Operasi : ((Rr)) <- data
Keterangan : instruksi ini akan memindahkan data 8 bit secara langsung ke lokasi memori yang ditunjukkan oleh isi register r
Contoh : MOV @R1, #01H

MOV (Move Accumulator to Indirect Address)

Operand : Rr Register 0 <= r <= 1
: A Akkumulator
Format : MOV @Rr, A
Operasi : ((Rr)) <- (A)
Keterangan : instruksi ini menambah isi akumulator ke lokasi memori yang ditunjukkan oleh isi register r.
Contoh : MOV @R0, A

MOV (Move Memory to Indirect Address)

Operand : Rr Register 0 <= r <= 1
: Alamat data 0 <= Alamat data <= 255
Format : MOV @Rr, alamat data
Operasi : ((Rr)) <- (alamat data)
Keterangan : instruksi ini memindahkan isi suatu alamat data ke lokasi memori yang ditunjukkan oleh isi register r.
Contoh : MOV @R1, 77H

MOV (Move Immediate Data to Accumulator)

Operand : A Akkumulator
: Data -256 <= data <= +255
Format : MOV A, #data
Operasi : (A) <- data
Keterangan : instruksi ini memindahkan data 8 bit secara langsung ke akumulator

Contoh : MOV A, #02H

MOV (Move Indirect Address to Accumulator)

Operand : A Akumulator
: Rr Register 0 <= r <= 1
Format : MOV A, @Rr
Operasi : (A) <- ((Rr))
Keterangan : instruksi ini memindahkan isi data memori yang lokasinya ditunjukkan oleh register r ke akumulator
Contoh : MOV A, @R0

MOV (Move Register to Accumulator)

Operand : A Akkumulator
: Rr Register 0<= r <= 7
Format : MOV A, Rr
Operasi : (A) <- (Rr)
Keterangan : instruksi ini memindahkan isi register r ke akumulator
Contoh : MOV A, #02H

MOV (Move Memory to Accumulator)

Operand : A Akkumulator
: Alamat data 0<= Alamat data <= 255
Format : MOV A, alamat data
Operasi : (A) <- (Alamat data)
Keterangan : instruksi ini memindahkan isi memori data pada suatu alamat ke akumulator
Contoh : MOV A, P3 ; pindahkan isi port 3 ke akumulator

MOV (Move Bit to Carry Flag)

Operand : C Carry Flag
: Alamat bit 0<= Alamat bit <= 255
Format : MOV C, alamat bit
Operasi : (C) <- (alamat bit)
Keterangan : instruksi ini memindahkan isi suatu alamat bit ke carry flag
Contoh : MOV C, P1.0

MOV (Move Immediate Data to Data Pointer)

Operand : Data Pointer
: Data 0 <= data <= 65535
Format : MOV DPTR, #data
Operasi : (DPTR) <- data
Keterangan : instruksi ini memindahkan data 16 bit secara langsung ke data pointer (DPTR)
Contoh : MOV DPTR, #0B19H

MOV (Move Memory to Memory)

Operand : Alamat1 0 <= Alamat 1 <= 255
: Alamat 2 0 <= Alamat 2 <= 255
Format : MOV alamat 1, alama 2
Operasi : (Alamat 1) <- (Alamat 2)
Keterangan : instruksi ini memindahkan isi memori alamat data sumber (alamat 2) ke alamat daa tujuan (alamat 1)
Contoh : MOV 13H, 12H

MOVC

(Move Code Memory Offset from Data Pointer to Accumulator)

Operand : A Akumulator
: DPTR Data Pointer
Format : MOVC A, @A + DPTR
Operasi : (A) <- ((A) + (DPTR))
Keterangan : instruksi ini menjumlahkan isi data pointer dengan isi akumulator. Hasil penjum-lahan merupakan alamat kode memori dan isinya akan dipindahkan ke akumulator.
Contoh : MOVC, @A + DPTR

MOVC

(Move Code Memory Offset from Program Counter to Accumulator)

Operand : A Akumulator
: PC Program Counter
Format : MOVC A, @A + PC
Operasi : (PC) <- (PC) + 1
: (A) <- ((A) + (PC))
Keterangan : instruksi ini menjumlahkan isi pencacah program yang telah dinaikkan dengan isi akumulator. Hasil penjumlahan tersebut digunakan sebagai alamat kode memori dan isinya dipindahkan ke akumulator.

Contoh : MOVC, @A + PC

MOVX

Move Accumulator to External Memory Addressed by Data Pointer

Operand : DPTR Data Pointer
: A Akumulator
Format : MOVX @DPTR, A
Operasi : ((DPTR)) <- (A)
Keterangan : instruksi ini akan memindahkan isi akumulator ke memori data eksternal (off chip) yang alamatnya ditunjukkan oleh isi data pointer.
Contoh : MOVX @DPTR, A

MOVX

(Move Accumulator to External Memory Addressed by Register)

Operand : Rr Register 0<= r <= 1
: A Akumulator
Format : MOVX @Rr, A
Operasi : ((Rr)) <- (A)
Keterangan : instruksi ini akan memindahkan isi akumulator ke me-mori data eksternal yang alamatnya ditunjukkan oleh register r dan SFR P2. P2 menampung byte atas alamat dan register r menampung byte bawah.
Contoh : MOV P2, #00H
: MOVX @R0, A

MOVX

Move External Memory Addressed by Data Pointer to Accumulator

Operand : A Akumulator
: DPTR Data Pointer
Format : MOVX A, @DPTR
Operasi : (A) <- ((DPTR))
Keterangan : instruksi ini akan memindahkan isi memori data eksternal yang alamatnya ditunjukkan oleh data pointer ke akumulator.
Contoh : MOVX A, @DPTR

MOVX

(Move External Memory Addressed by Register to Accumulator)

Operand : A Akumulator

: Rr Register 0 <= r <= 1

Format : MOVX A, @Rr

Operasi : (A) <- ((Rr))

Keterangan : instruksi ini akan memindahkan isi memori data eks-ternal yang alamatnya ditunjukkan oleh register r dan SFR P2 ke akumulator. P2 menampung byte atas alamat dan register r menampung byte bawah.

Contoh : MOV P2, #55H

: MOVX A, @R1

MUL (Multiply Accumulator by B)

Operand : AB

Format : MUL AB

Operasi : (AB) <- (A) * (B)

Keterangan : instruksi ini akan mengalikan isi akumulator dengan isi register pengali (B). Byte bawah hasil perkalian dimasukkan ke akumulator dan byte atas dimasukkan ke register pengali.

Contoh : MOV B, #10

: MUL AB

NOP (No Operation)

Operand : -

Format : NOP

Operasi : Tak ada operasi

Keterangan : instruksi ini tidak melakukan apa pun selama satu siklus

Contoh : NOP

ORL (Logical OR Immediate Data to Accumulator)

Operand : A Akkumulator

: data -256 <= data <= +255

Format : ORL A, #data

Operasi : (A) <- (A) OR data

Keterangan : instruksi ini meng-OR kan data 8 bit secara langsung dengan isi akumulator

Contoh : OR A,#00001000B

ORL (Logical OR Indirect Address to Accumulator)

Operand : A Akkumulator

: Rr Register 0 <= r <= 1

Format : ORL A, @Rr

Operasi : (A) <- (A) OR ((Rr))

Keterangan : instruksi ini meng-OR kan isi memori yang lokasinya ditunjukkan oleh isi register r dengan isi akumulator. Hasilnya disimpan di akumulator.

Contoh : ORL A,@R0

ORL (Logical OR Register to Accumulator)

Operand : A Akkumulator

: Rr 0 <= Rr <= 7

Format : ORL A, Rr

Operasi : (A) <- (A) OR (Rr)

Keterangan : instruksi ini meng-OR kan isi register r dengan isi akumulator. Hasilnya disim-pan di akumulator

Contoh : ORL A, R4

ORL (Logical OR Memory to Accumulator)

Operand : A Akkumulator

: Alamat data 0 <= Alamat data <= 255

Format : ORL A, Alamat data

Operasi : (A) <- (A) OR (Alamat data)

Keterangan : instruksi ini meng-OR kan isi alamat data dengan isi akumulator. Hasilnya disimpan di akumulator.

Contoh : ORL A,35H

ORL (Logical OR Bit to Carry Flag)

Operand : C Carry flag

: Alamat bit 0 <= alamat bit <= 255

Format : ORL C, Alamat bit

Operasi : (C) <- (C) OR (Alamat bit)

Keterangan : instruksi ini meng-OR kan isi alamat bit tertentu de-ngan isi carry flag. Hasilnya ditempatkan pada carry flag.

Contoh : ORL C, 46.2

ORL (Logical OR Complement of Bit to Carry Flag)

Operand : C Carry flag

: Alamat bit 0 <= alamat bit <= 255

Format : ORL C, /alamat bit

Operasi : (C) <- (C) OR NOT (alamat bit)

Keterangan : instruksi ini meng-OR kan hasil komplemen isi alamat bit tertentu dengan isi carry flag. Hasilnya ditempatkan pada carry flag. Isi alamt bit semula tidak berubah.

Contoh : ORL C,/25H.5

ORL (Logical OR Immediate Data to Memory)

Operand : Alamat data 0 <= alamat data <= 255

: data -256 <= data <= +255

Format : ORL Alamat data, #data

Operasi : (Alamat data) <- (Alamat data) OR data

Keterangan : instruksi ini meng-OR kan data 8 bit secara langsung dengan isi alamat data tertentu. Hasilnya akan disimpan dalam memori data pada alamat tersebut.

Contoh : ORL 57H,#01H

ORL (Logical OR Accumulator to Memory)

Operand : Alamat data 0 <= Alamat data <= 255

: A Akumulator

Format : ORL Alamat data, A

Operasi : (Alamat data) <- (Alamat data) OR A

Keterangan : instruksi ini meng- OR kan isi akumulator dengan isi alamat data tertentu. Hasilnya disimpan dalam memori data pada alamat yang bersangkutan

Contoh : ORL 10H,A

POP (Pop Stack to Memory)

Operand : Alamat data 0 <= Alamat data <= 255

Format : POP Alamat data

Operasi : (Alamat data) <- ((SP))

: (SP) <- (SP) - 1

Keterangan : instruksi ini menempatkan byte yang ditunjukkan oleh stack pointer ke suatu alamat data, kemudian mengurangi satu isi stack pointer.

Contoh : POP PSW

PUSH (Push Memory onto Stack)

Operand : Alamat data 0 <= Alamat data <= 255

Format : PUSH Alamat data

Operasi : (SP) <- (SP) + 1

: ((SP)) <- (Alamat data)

Keterangan : instruksi ini menaikkan stack pointer kemudian menyimpan isinya ke suatu alamat data pada lokasi yang ditunjukkan oleh stack pointer.

Contoh : PUSH 4DH

RET (Return from Subroutine) (Non Interrupt)

Operand : -

Format : RET

Operasi : (PC high) <- ((SP))

: (SP) <- (SP) - 1

: (PC low) <- ((SP))

: (SP) <- (SP) -1

Keterangan : instruksi ini dipakai untuk kembali dari suatu subroutine ke alamat terakhir saat subroutine dipanggil.

RETI (Return from Interrupt Routine)

Operand : -

Format : RETI

Operasi : (PC high) <- ((SP))

: (SP) <- (SP) - 1

: (PC low) <- ((SP))

: (SP) <- (SP) -1

Keterangan : instruksi ini dipakai untuk kembali dari suatu routine pelayanan interupsi.

RL (Rotate Accumulator Left)

Operand : A Akumulator

Format : RL A

Operasi :

Keterangan : instruksi ini memutar setiap bit dalam akumulator satu posisi ke kiri. Bit paling besar (MSB) bergerak ke bit paling kecil (LSB).

Contoh : RL A

RLC (Rotate Accumulator And Carry Flag Left)

Operand : A Akumulator

Format : RLC A

Operasi :

Keterangan : instruksi ini memutar bit bit dalam akumulator satu posisi ke kiri. Bit paling besar (MSB) bergerak ke dalam carry flag. Sedangkan isi carry flag menuju ke LSB

Contoh : RLC A

RR (Rotate Accumulator Right)

Operand : A Akumulator

Format : RR A

Operasi :

Keterangan : instruksi ini memutar setiap bit dalam akumulator satu posisi ke kanan. Bit paling kecil (LSB) bergerak ke bit paling besarl (MSB).

Contoh : RR A

RRC (Rotate Accumulator And Carry Flag Right)

Operand : A Akumulator

Format : RRC A

Operasi :

Keterangan : instruksi ini memutar bit bit dalam akumulator satu posisi ke kanan. Bit paling kecil (LSB) bergerak ke dalam carry flag. Sedangkan isi carry flag menuju ke MSB

Contoh : RRC A

SETB (Set Carry Flag)

Operand : C Carry Flag

Format : SETB C

Operasi : (C) <- 1

Contoh : SETB C

SETB (Set Bit)

Operand : Alamat bit 0 <= Alamat bit <= 255

Format : SETB Alamat bit

Operasi : (Alamat bit) <- 1

Keterangan : Instruksi ini akan men- set isi suatu alamat bit menjadi 1

Contoh : SETB 41.5

SUBB (Subtract Indirect Address from Accumulator with Borrow)

Operand : A Akumulator

: Rr Register 0 <= r <= 1

Operasi : (A) <- (A) - (C) - ((Rr))

Keterangan : instruksi ini akan mengurangkan isi akumulator de-ngan carry flag dan isi lokasi memori yang ditunjukkan oleh isi register r. Hasilnya disimpan dalam akumulator.

SUBB (Subtract Immediate Data from Accumulator with Borrow)

Operand : A Akumulator

: Data -256 <= data <= +255

Format : SUBB A, #data

Operasi : (A) <- (A) - (C) - data

Keterangan : instruksi ini akan mengurangkan isi carry flag dan data langsung dari isi akumulator. Hasilnya disimpan dalam akumulator.

Contoh : SUBB A, #0C1H

SUBB (Subtract Register from Accumulator with Borrow)

Operand : A Akumulator

: Rr Register 0<= r <= 7

Format : SUBB A, Rr

Operasi : (A) <- (A) - (C) - (Rr)

Keterangan : instruksi ini akan mengurangi isi akumulator dengan isi carry flag dan isi register r. Hasilnya disimpan dalam akumulator.

Contoh : SUBB A, R6

SUBB (Subtract Memory from Accumulator with Borrow)

Operand : A Akumulator

: Alamat data 0 <= Alamat data <= 255

Format : SUBB A, Alamat data

Operasi : (A) <- (A) - (C) - (Alamat data)

Keterangan : instruksi ini akan mengurangkan isi akumulator de-ngan isi carry flag dan isi suatu alamat data. Hasilnya disimpan dalam akumulator.

Contoh : SUBB A, 32H

SJMP (Short Jump)

Operand : Alamat data

Format : SJMP alamat kode

Operasi : (PC) <- (PC) + 2

: (PC) <- (PC) + offset relatif

Keterangan : instruksi ini akan menyebabkan operasi melompat ke alamat kode

SWAP (Exchange Nibbles in Accumulator)

Operand : A Akumulator

Format : SWAP A

Keterangan : instruksi ini akan mempertukarkan isi nibble bawah dengan nibble atas

Contoh : SWAP A

XCH (Exchange Indirect Address with Accumulator)

Operand : A Akumulator

: Rr Register 0 <= r <= 1

Format : XCH A, @Rr

Operasi : temp <- ((Rr))

: ((Rr)) <- (A)

: (A) <- temp

Keterangan : instruksi ini akan menukar isi lokasi memori yang ditunjukkan oleh isi register r dengan isi akumulator

Contoh : XCH A, @R0

XCH (Exchange Register with Accumulator)

Operand : A Akumulator

: Rr Register 0 <= r <= 7

Format : XCH A, Rr

Operasi : temp <- ((Rr))

: ((Rr)) <- (A)

: (A) <- temp

Keterangan : instruksi ini akan menukar isi register dengan isi akumulator

Contoh : XCH A, R6

XCH (Exchange Memory with Accumulator)

Operand : A Akumulator

: Alamat data 0<= Alamat data <= 255

Format : XCH A, alamat data

Operasi : temp <- (Alamat data)

: (Alamat data) <- (A)

: (A) <- temp

Keterangan : instruksi ini akan menukar isi suatu alamat data dengan isi akumulator

Contoh : XCH A, 37H

XCHD

(Exchange Low Nibbles of Indirect Address with Accumulator)

Operand : A Akumulator

: Rr Register 0 <= r <= 1

Format : XCHD A, @Rr

Operasi : temp <- $((Rr))_{0-3}$

: $((Rr))_{0-3}$ <- $(A)_{0-3}$

: $(A)_{0-3}$ <- temp

Keterangan : instruksi ini akan menukar isi nibble bawah dari lokasi memori yang alamatnya ditunjukkan oleh isi register r dengan isi nibble bawah akumulator.

Contoh : XCHD A, @R0

XRL (Logical XOR Immediate Data to Accumulator)

Operand : A Akkumulator

: data -256 <= data <= +255

Format : XRL A, #data

Operasi : (A) <- (A) XOR data

Keterangan : instruksi ini meng-XOR kan data 8 bit secara langsung dengan isi akumulator

Contoh : XOR A,#0FH

XRL (Logical XOR Indirect Address to Accumulator)

Operand : A Akkumulator

: Rr Register 0 <= r <= 1

Format : XRL A, @Rr

Operasi : (A) <- (A) XOR ((Rr))

Keterangan : instruksi ini meng-XOR kan isi memori yang lokasinya ditunjukkan oleh isi register r dengan isi akumulator. Hasilnya disimpan di akumulator.

Contoh : XRL A,@R0

XRL (Logical XOR Register to Accumulator)

Operand : A Akkumulator

: Rr Register 0 <= Rr <= 7

Format : XRL A, Rr

Operasi : (A) <- (A) XRL (Rr)

Keterangan : instruksi ini meng-XOR kan isi register r dengan isi akumulator. Hasilnya disim-pan di akumulator

Contoh : XRL A, R4