Instruksi-instruksi Keluarga MCS51
Operasi Aritmatika

ADD

ADD A,Rn

Siklus	Jumlah Byte	Instruksi							
1	1	ADD A,Rn							
Flag		C	AC	F0	RS1	RS0	OV		P
		X	X				X		X

Tambahkan Akumulator A dengan Rn di mana n = 0…7 dan simpan hasil di Akumulator A

Contoh:

```
Add     A,R7
```

Isi dari R7 akan ditambahkan dengan akumulator A dan hasilnya disimpan di Akumulator A

ADD A,direct

Siklus	Jumlah Byte	Instruksi							
1	2	ADD A,direct							
Flag		C	AC	F0	RS1	RS0	OV		P
		X	X				X		X

Tambahkan Akumulator A dengan data di alamat memori tertentu secara langsung.

Contoh:

```
Add     A,00H
```

Isi dari Akumulator A akan ditambahkan dengan isi dari memori RAM Internal di alamat 00H

ADD A,@Ri

Siklus	Jumlah Byte	Instruksi							
1	1	ADD A,@Ri							
Flag		C	AC	F0	RS1	RS0	OV		P
		X	X				X		X

Tambahkan Akumulator A dengan data yang berada di alamat Ri (ditunjuk oleh Ri) dan hasilnya disimpan di Akumulator A. Ri adalah Register Index di mana pada MCS51 adalah berupa R0 atau R1

Contoh:

```
Add     A,@R0
```

Isi dari Akumulator A akan ditambahkan dengan isi dari memori RAM Internal yang ditunjuk oleh R0. Apabila R0 berisi 05H maka, isi dari alamat 05H akan dijumlahkan dengan Akumulator A dan hasilnya disimpan di Akumulator A

ADD A,#data

Siklus	Jumlah Byte	Instruksi
1	2	ADD A,#data

Flag	C	AC	F0	RS1	RS0	OV		P
	X	X				X		X

Tambahkan Akumulator A dengan sebuah konstanta dan hasilnya disimpan dalam akumulator A.

Contoh:

```
Add     A,#05H
```

Isi Akumulator A ditambah dengan data 05H dan hasilnya disimpan dalam Akumulator A

ADDC

ADDC A,Rn

Siklus	Jumlah Byte	Instruksi
1	1	ADDC A,Rn

Flag	C	AC	F0	RS1	RS0	OV		P
	X	X				X		X

Tambahkan Akumulator A dengan Rn di mana n = 0…7 dan simpan hasil di Akumulator A

Contoh:

```
Addc    A,R7
```

Isi dari R7 akan ditambahkan dengan akumulator A beserta carry flag dan hasilnya disimpan di Akumulator A. Apabila carry flag set maka hasil yang tersimpan di Akumulator A adalah A + R7 + 1.

ADDC A,direct

Siklus	Jumlah Byte	Instruksi
1	2	ADDC A,direct

Flag	C	AC	F0	RS1	RS0	OV		P
	X	X				X		X

Tambahkan Akumulator A dan carry flag dengan data di alamat memori tertentu secara langsung.

Contoh:

```
Addc    A,00H
```

Isi dari Akumulator A akan ditambahkan dengan isi dari memori RAM Internal di alamat 00H beserta carry flag dan hasilnya disimpan di Akumulator A, Apabila carry flag set maka hasil yang tersimpan di Akumulator A adalah A + isi alamat 00H + 1

Siklus	Jumlah Byte	Instruksi
1	1	ADDC A,@Ri

Flag	C	AC	F0	RS1	RS0	OV		P
	X	X				X		X

Tambahkan Akumulator A beserta carry flag dengan data yang berada di alamat Ri (ditunjuk oleh Ri) dan hasilnya disimpan di Akumulator A. Ri adalah Register Index di mana pada MCS51 adalah berupa R0 atau R1

Contoh:

```
Add     A,@R0
```

Isi dari Akumulator A beserta carry flag akan ditambahkan dengan isi dari memori RAM Internal yang ditunjuk oleh R0. Apabila R0 berisi 05H maka, isi dari alamat 05H akan dijumlahkan dengan Akumulator A beserta carry flag dan hasilnya disimpan di Akumulator A

Siklus	Jumlah Byte	Instruksi							
1	2	ADDC A,#data							
Flag		C	AC	F0	RS1	RS0	OV		P
		X	X				X		X

Tambahkan Akumulator A beserta carry flag dengan sebuah konstanta dan hasilnya disimpan dalam akumulator A.
Contoh:

Addc A,#05H

Isi Akumulator A beserta carry flag ditambah dengan data 05H dan hasilnya disimpan dalam Akumulator A. Apabila carry flag set maka hasil di Akumulator A adalah A + 5H + 1.

SUBB

SUBB A,Rn

Siklus	Jumlah Byte	Instruksi							
1	1	SUBB A,Rn							
Flag		C	AC	F0	RS1	RS0	OV		P
		X	X				X		X

Lakukan pengurangan data di Akumulator A dengan Rn (n = 0…7) dan simpan hasilnya di Akumulator A
Contoh:

Subb A,R0

Data di akumulator A beserta carry flagnya dikurangi dengan isi R0 dan hasilnya disimpan di Akumulator A

SUBB A,direct

Siklus	Jumlah Byte	Instruksi							
1	2	SUBB A,direct							
Flag		C	AC	F0	RS1	RS0	OV		P
		X	X				X		X

Lakukan pengurangan data di Akumulator A dengan data di memori tertentu yang ditunjuk secara langsung.
Contoh:

Subb A,00H

Data di Akumulator A beserta carry flagnya dikurangi dengan data di alamat 00H dari RAM Internal dan hasilnya disimpan di Akumulator A

SUBB A,@Ri

Siklus	Jumlah Byte	Instruksi							
1	1	SUBB A,@Ri							
Flag		C	AC	F0	RS1	RS0	OV		P
		X	X				X		X

Lakukan pengurangan data di Akumulator A beserta carry flag dengan data yang ditunjuk oleh Ri (Register Index) di mana Ri dapat berupa R0 atau R1
Contoh:

SUBB A,@R0

Data di Akumulator A beserta carry flagnya dikurangi dengan data yang ditunjuk oleh R0 dan hasilnya disimpan di Akumulator A

SUBB A,#data

Siklus	Jumlah Byte	Instruksi							
1	2	SUBB A,#data							
Flag		C	AC	F0	RS1	RS0	OV		P
		X	X				X		X

Lakukan pengurangan data di Akumulator A beserta carry flag dengan sebuah konstanta dan hasilnya disimpan di Akumulator A
Contoh:

SUBB A,#05H

Data di Akumulator A beserta carry flag dikurangi dengan data 05H dan hasilnya disimpan di Akumulator A

INC

INC A

Siklus	Jumlah Byte	Instruksi							
1	1	INC A							
Flag		C	AC	F0	RS1	RS0	OV		P

Tambahkan nilai Akumulator A dengan 1 dan hasilnya disimpan di Akumulator A

INC Rn

Siklus	Jumlah Byte	Instruksi							
1	1	INC Rn							
Flag		C	AC	F0	RS1	RS0	OV		P

Tambahkan nilai Rn (n= 0…7) dengan 1 dan hasilnya disimpan di Rn tersebut

INC direct

Siklus	Jumlah Byte	Instruksi
1	2	INC direct

Flag	C	AC	F0	RS1	RS0	OV		P

Tambahkan data yang di RAM Internal yang alamatnya ditunjuk secara langsung dengan 1 dan hasilnya disimpan di alamat tersebut.
Contoh:

```
Inc     00H
```

Data di alamat 00H ditambah dengan 1 dan hasilnya disimpan di alamat 00H.

INC @Ri

Siklus	Jumlah Byte	Instruksi							
1	1	INC @Ri							
Flag		C	AC	F0	RS1	RS0	OV		P

Tambahkan data yang alamatnya ditunjuk oleh Ri (Register Index) dengan 1 dan simpan hasilnya di alamat tersebut.
Contoh:

```
Inc     @R1
```

Data di alamat yang ditunjuk oleh R1 dan hasilnya disimpan di alamat tersebut, apabila R1 berisi 10H maka data di alamat 10H ditambah dengan 1 dan simpan kembali di alamat 10H.

INC DPTR

Siklus	Jumlah Byte	Instruksi							
2	1	INC DPTR							
Flag		C	AC	F0	RS1	RS0	OV		P

Tambahkan nilai DPTR dengan 1 dan simpan hasilnya di DPTR

DEC

DEC A

Siklus	Jumlah Byte	Instruksi							
1	1	DEC A							
Flag		C	AC	F0	RS1	RS0	OV		P

Lakukan pengurangan pada nilai Akumulator A dengan 1 dan hasilnya disimpan di Akumulator A

DEC Rn

Siklus	Jumlah Byte	Instruksi							
1	1	DEC Rn							
Flag		C	AC	F0	RS1	RS0	OV		P

Lakukan pengurangan pada nilai Rn (n= 0…7) dengan 1 dan hasilnya disimpan di Rn tersebut

DEC direct

Siklus	Jumlah Byte	Instruksi							
1	2	DEC direct							
Flag		C	AC	F0	RS1	RS0	OV		P

Lakukan pengurangan pada data yang di RAM Internal yang alamatnya ditunjuk secara langsung dengan 1 dan hasilnya disimpan di alamat tersebut.
Contoh:

```
Dec     00H
```

Data di alamat 00H dikurangi dengan 1 dan hasilnya disimpan di alamat 00H.

DEC @Ri

Siklus	Jumlah Byte	Instruksi							
1	1	DEC @Ri							
Flag		C	AC	F0	RS1	RS0	OV		P

Lakukan pengurangan pada data yang alamatnya ditunjuk oleh Ri (Register Index) dengan 1 dan simpan hasilnya di alamat tersebut.
Contoh:

```
DEC    @R1
```

Data di alamat yang ditunjuk oleh R1 dan hasilnya disimpan di alamat tersebut, apabila R1 berisi 10H maka data di alamat 10H dikurangi dengan 1 dan simpan kembali di alamat 10H.

MUL AB

Siklus	Jumlah Byte	Instruksi							
4	1	MUL AB							
Flag		C	AC	F0	RS1	RS0	OV		P
							X		

Lakukan perkalian antara Akumulator A dan Register B, hasil dari perkalian disimpan di Akumulator A, untuk byte rendah dan di Register B untuk byte tinggi. Bila hasil perkalian lebih dari 255 (0FFH) maka flag overflow akan set, sedangkan carry akan selalu clear.
Contoh:

```
Mov    A,#50H
Mov    B,#0A0H
Mul    AB
```

Hasil perkalian dari 50H atau 80 desimal dengan 0A0H atau 160 desimal adalah 3200H atau 12800, maka nilai yang disimpan di Akumulator A adalah 00H dan di Register B adalah 32H. Sedangkan Flag Overflow akan set karena hasil dari perkalian lebih besar daripada 255 atau (0FFH)

DIV AB

Siklus	Jumlah Byte	Instruksi							
4	1	DIV AB							
Flag		C	AC	F0	RS1	RS0	OV		P
							X		

Lakukan pembagian antara Akumulator A dan Register B, hasil dari pembagian akan disimpan di akumulator A dan sisa pembagian disimpan di Register B. Flag Overflow dan Carry akan selalu clear

Flag Overflow akan set apabila isi dari Register B adalah 00 di mana hal ini menandakan bahwa proses pembagian tidak mungkin dilakukan karena hasil pembagian antara suatu bilangan dengan 0 adalah tidak berhingga.

Contoh:

```
Mov   A,#0FB
Mov   B,#12H
Div   AB
```

Hasil dari pembagian ini adalah 0DH dengan sisa 11H, maka 0DH akan tersimpan di Akumulator A sebagai hasil bagi dan 11H tersimpan di Register B sebagai sisa bagi.

DA A

Siklus	Jumlah Byte	Instruksi							
1	1	DA A							
Flag		C	AC	F0	RS1	RS0	OV		P
		X	X						

Mengubah data di Akumulator A menjadi bentuk BCD. Instruksi ini akan mengubah sebuah 8 bit data pada Akumulator A dalam bentuk hexa menjadi 2 digit di mana digit pertama pada nibble atas berupa bit 7…4 dari Akumulator A dan digit kedua adalah nibble bawah berupa bit 3…0 dari Akumulator A. Pada dasarnya instruksi ini akan menambahkan 6 apabila nibble rendah berada di atas 9 atau bit AC set dan menambahkan 6 apabila nibble tinggi berada di atas 9 atau bit Carry Set. Instruksi ini biasa digunakan sesudah instruksi ADD.

Contoh 1:

```
Add   A,#88H
DA    A
```

Nilai akumulator A sebelumnya adalah 99H, hasil penjumlahan 99H dan 88H adalah 21H dengan bit AC set dan Carry set karena hasil penjumlahan 9 dan 8 baik di nibble tinggi maupun nibble rendah adalah lebih besar dari 9. Karena bit AC set dan bit carry set maka hasil penjumlahan kedua nibble tersebut masing-masing ditambah 6 dan menghasilkan 87H dengan carry flag set.

Contoh 2:

```
Add   A,#02H
DA    A
```

Nilai Akumulator A sebelumnya adalah 79H, hasil penjumlahan dengan 02H adalah 7BH dengan bit AC set karena hasil penjumlahan pada nibble rendah lebih besar dari 9 dan

setelah melewati instruksi DA A, maka 7BH akan ditambah dengan 6H sehingga menjadi 81H.
Intinya, apabila kondisi:

- Bit AC set, Bit Carry clear = Akumulator A ditambah 06H
- Bit AC clear, Bit Carry clear = Akumulator A ditambah 00H
- Bit AC clear, Bit Carry set = Akumulator A ditambah 60H
- Bit AC set, Bit Carry set = Akumulator A ditambah 66H

Instruksi DA A memang bukan sebuah instruksi yang mengubah bilangan heksa menjadi bentuk desimal dengan mudah, namun instruksi ini berfungsi untuk membuat operasi penjumlahan secara BCD sehingga hasil penjumlahan dari 79H dan 02H bukan menjadi 7BH melainkan menjadi 81H.

Operasi Logika

ANL

Melakukan operasi AND antara dua buah variabel dan menyimpannya di variabel tujuan. Apabila variabel tujuan adalah akumulator, maka variabel yang lain dapat menggunakan register (Rn), alamat langsung, tak langsung atau immediate data, apabila variabel tujuan adalah alamat langsung, maka variabel yang lain dapat menggunakan akumulator atau immediate data

ANL A,Rn

Siklus	Jumlah Byte	Instruksi							
1	1	ANL A,Rn							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi AND antara akumulator A dan Rn (R0…R7) dan hasilnya disimpan di akumulator A

ANL A,direct

Siklus	Jumlah Byte	Instruksi							
1	2	ANL A,direct							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi AND antara akumulator A dan alamat langsung dan hasilnya disimpan di akumulator A.
Contoh:

```
ANL   A,05H
```

Akumulator A di AND dengan data di alamat 05H dan hasilnya disimpan di akumulator A

ANL A,@Ri

Siklus	Jumlah Byte	Instruksi							
1	1	ANL A,@Ri							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi AND antara akumulator A dan data yang ditunjuk oleh Register Index (R0 atau R1) serta hasilnya disimpan di akumulator A.
Contoh:

```
ANL  A,@R0
```

Akumulator A di AND dengan data yang ditunjuk oleh R0, misalkan R0 berisi 50H, maka akumulator A di AND dengan data yang tersimpan di alamat 50H dan hasilnya disimpan di akumulator A.

ANL A,#data

Siklus	Jumlah Byte	Instruksi							
1	2	ANL A,#data							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi AND antara akumulator A dan immediate data serta hasilnya disimpan di akumulator A

ANL direct,A

Siklus	Jumlah Byte	Instruksi							
1	2	ANL direct,A							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi AND antara alamat langsung dengan akumulator A serta hasilnya disimpan di alamat langsung tersebut.
Contoh:

```
ANL  07H,A
```

Data di alamat 07H di AND dengan akumulator A dan hasilnya kembali disimpan di alamat 07H

ANL direct,#data

Siklus	Kode Operasi	Instruksi							
2	3	ANL direct,#data							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi AND antara alamat langsung dengan immediate data serta hasilnya disimpan di alamat langsung tersebut.

ORL

Melakukan operasi OR antara dua buah variabel dan menyimpannya di variabel tujuan. Apabila variabel tujuan adalah akumulator, maka variabel yang lain dapat menggunakan register (Rn), alamat langsung, tak langsung atau immediate data, apabila variabel tujuan adalah alamat langsung, maka variabel yang lain dapat menggunakan akumulator atau immediate data

ORL A,Rn

Siklus	Jumlah Byte	Instruksi							
1	1	ORL A,Rn							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi OR antara akumulator A dan Rn (R0…R7) dan hasilnya disimpan di akumulator A

ORL A,direct

Siklus	Jumlah Byte	Instruksi							
1	2	ORL A,direct							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi OR antara akumulator A dan alamat langsung dan hasilnya disimpan di akumulator A.
Contoh:

```
ORL   A,05H
```

Akumulator A di OR dengan data di alamat 05H dan hasilnya disimpan di akumulator A

ORL A,@Ri

Siklus	Jumlah Byte	Instruksi							
1	1	ORL A,@Ri							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi OR antara akumulator A dan data yang ditunjuk oleh Register Index (R0 atau R1) serta hasilnya disimpan di akumulator A.
Contoh:

```
ORL   A,@R0
```

Akumulator A di OR dengan data yang ditunjuk oleh R0, misalkan R0 berisi 50H, maka akumulator A di OR dengan data yang tersimpan di alamat 50H dan hasilnya disimpan di akumulator A.

ORL A,#data

Siklus	Jumlah Byte	Instruksi							
1	2	ORL A,#data							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi OR antara akumulator A dan immediate data serta hasilnya disimpan di akumulator A

ORL direct,A

Siklus	Jumlah Byte	Instruksi
1	2	ORL direct,A

Flag	C	AC	F0	RS1	RS0	OV		P

Melakukan operasi OR antara alamat langsung dengan akumulator A serta hasilnya disimpan di alamat langsung tersebut.
Contoh:

ORL 07H,A

Data di alamat 07H di OR dengan akumulator A dan hasilnya kembali disimpan di alamat 07H

ORL direct,#data

Siklus	Jumlah Byte	Instruksi							
2	3	ORL direct,#data							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi OR antara akumulator A dan immediate data serta hasilnya disimpan di akumulator A

XRL

Melakukan operasi EXOR antara dua buah variabel dan menyimpannya di variabel tujuan. Apabila variabel tujuan adalah akumulator, maka variabel yang lain dapat menggunakan register (Rn), alamat langsung, tak langsung atau immediate data, apabila variabel tujuan adalah alamat langsung, maka variabel yang lain dapat menggunakan akumulator atau immediate data

XRL A,Rn

Siklus	Jumlah Byte	Instruksi							
1	1	XRL A,Rn							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi EXOR antara akumulator A dan Rn (R0…R7) dan hasilnya disimpan di akumulator A

XRL A,direct

Siklus	Jumlah Byte	Instruksi							
1	2	XRL A,direct							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi EXOR antara akumulator A dan alamat langsung dan hasilnya disimpan di akumulator A.
Contoh:

XRL A,05H

Akumulator A di EXOR dengan data di alamat 05H dan hasilnya disimpan di akumulator A

XRL A,@Ri

Siklus	Jumlah Byte	Instruksi							
1	1	XRL A,@Ri							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi EXOR antara akumulator A dan data yang ditunjuk oleh Register Index (R0 atau R1) serta hasilnya disimpan di akumulator A.
Contoh:

XRL A,@R0

Akumulator A di EXOR dengan data yang ditunjuk oleh R0, misalkan R0 berisi 50H, maka akumulator A di EXOR dengan data yang tersimpan di alamat 50H dan hasilnya disimpan di akumulator A.

XRL A,#data

Siklus	Jumlah Byte	Instruksi							
1	2	XRL A,#data							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi EXOR antara akumulator A dan data yang ditunjuk oleh Register Index (R0 atau R1) serta hasilnya disimpan di akumulator A.
Contoh:

XRL A,@R0

Akumulator A di EXOR dengan data yang ditunjuk oleh R0, misalkan R0 berisi 50H, maka akumulator A di EXOR dengan data yang tersimpan di alamat 50H dan hasilnya disimpan di akumulator A.

XRL direct,A

Siklus	Jumlah Byte	Instruksi							
1	2	XRL direct,A							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi OR antara alamat langsung dengan akumulator A serta hasilnya disimpan di alamat langsung tersebut.
Contoh:

ORL 07H,A

Data di alamat 07H di OR dengan akumulator A dan hasilnya kembali disimpan di alamat 07H

XRL direct,#data

Siklus	Jumlah Byte	Instruksi							
2	3	XRL direct,#data							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan operasi EXOR antara alamat langsung dengan immediate data serta hasilnya disimpan di alamat langsung tersebut.

CLR A

Siklus	Jumlah Byte	Instruksi							
1	1	CLR A							
Flag		C	AC	F0	RS1	RS0	OV		P

Memberikan nilai 0 pada 8 bit Akumulator A

CPL A

Siklus	Jumlah Byte	Instruksi							
1	1	CPL A							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan komplemen pada setiap bit dalam akumulator A.
Contoh :
Bila nilai akumulator A adalah 55H atau 01010101b, maka setelah terjadi proses komplemen nilai akumulator A berubah menjadi AAH atau 10101010b.

RL A

Siklus	Jumlah Byte	Instruksi							
1	1	RL A							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pergeseran ke kiri 1 bit untuk setiap bit dalam akumulator A
Contoh:
Nilai Akumulator A adalah 05H atau 00000101b, setelah dilakukan proses pergeseran maka nilai Akumulator A akan berubah menjadi 00001010b atau 0AH.

RLC A

Siklus	Jumlah Byte	Instruksi							
1	1	RLC A							
Flag		C	AC	F0	RS1	RS0	OV		P
		X							

Melakukan pergeseran ke kiri 1 untuk setiap bit dalam akumulator A diikuti dengan carry flag. Bit ke 7 akan masuk ke carry flag dan bit carry flag akan masuk ke bit 0.
Contoh:
Nilai Akumulator A adalah 05H atau 00000101b dan carry flag set, setelah dilakukan proses pergeseran maka nilai Akumulator A akan berubah menjadi 00001011b atau 0BH dengan carry flag clear.

RR A

Siklus	Jumlah Byte	Instruksi

1	1	RR A							
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**		**P**

Melakukan pergeseran ke kanan 1 bit untuk setiap bit dalam akumulator A
Contoh:
Nilai Akumulator A adalah 05H atau 00000101b, setelah dilakukan proses pergeseran maka nilai Akumulator A akan berubah menjadi 10000010b atau 0AH.

RRC A

Siklus	**Jumlah Byte**	**Instruksi**							
1	1	RRC A							
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**		**P**
		X							

Melakukan pergeseran ke kanan 1 untuk setiap bit dalam akumulator A diikuti dengan carry flag. Bit ke 7 akan masuk ke carry flag dan bit carry flag akan masuk ke bit 0.
Contoh:
Nilai Akumulator A adalah 05H atau 00000101b dan carry flag set, setelah dilakukan proses pergeseran maka nilai Akumulator A akan berubah menjadi 00001011b atau 0BH dengan carry flag clear.

SWAP A

Siklus	**Jumlah Byte**	**Instruksi**							
1	1	SWAP A							
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**		**P**

Melakukan operasi penukaran nibble tinggi dan nibble rendah di akumulator A
Contoh:
Isi akumulator A adalah 51H, setelah instruksi SWAP A dilakukan maka data 5 di nibble tinggi akan ditukar dengan data 1 di nibble rendah menadi 15H

Transfer Data

MOV

Melakukan pemindahan data dari variabel pada kode operasi kedua dan disimpan di variabel pada kode operasi pertama.

MOV A,Rn

Siklus	**Jumlah Byte**	**Instruksi**							
1	1	Mov A,Rn							
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**		**P**

Melakukan pemindahan data dari Rn (R0…R7) menuju ke akumulator A

MOV A,direct

Siklus	**Jumlah Byte**	**Instruksi**

1	2	Mov A,direct							
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**		**P**

Melakukan pemindahan data dari alamat langsung ke akumulator A

Mov A,@Ri

Siklus	**Jumlah Byte**	**Instruksi**							
1	1	Mov A,@Ri							
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**		**P**

Melakukan pemindahan data dari alamat yang ditunjuk oleh Register Index (R0 atau R1) menuju ke akumulator A

Mov A,#data

Siklus	**Jumlah Byte**	**Instruksi**							
1	2	Mov A,#data							
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**		**P**

Melakukan pemindahan data dari immediate menuju ke akumulator A
Contoh:

```
Data    EQU    05H
        Mov    A,#Data
```

Konstanta Data yang dideklarasikan sebagai 05H dipindah ke akumulator A sehingga nilai akumulator A menjadi 05H

Mov Rn,A

Siklus	**Jumlah Byte**	**Instruksi**							
1	1	Mov Rn,A							
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**		**P**

Melakukan pemindahan data dari akumulator A menuju ke Rn (R0…R7)

Mov Rn,direct

Siklus	**Jumlah Byte**	**Instruksi**							
2	2	Mov Rn,direct							
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**		**P**

Melakukan pemindahan data dari alamat langsung menuju ke Rn (R0…R7)
Contoh:

```
        Mov    R7,10H
```

Data di alamat 10H dipindah ke dalam R7

Mov Rn,#data

Siklus	Jumlah Byte	Instruksi							
1	2	Mov Rn,#data							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan data dari immediate menuju ke Rn (R0…R7)
Contoh:

Mov R7,#05H

Data 05H dipindah ke dalam R7

Mov direct,A

Siklus	Jumlah Byte	Instruksi							
1	2	Mov direct,A							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan data dari akumulator A menuju ke alamat langsung
Contoh:

Mov 10H,A

Data di akumulator A dipindah ke alamat 10H

Mov direct,Rn

Siklus	Jumlah Byte	Instruksi							
2	2	Mov direct,Rn							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan data dari Rn (R0…R7) menuju ke alamat langsung

Mov direct,direct

Siklus	Jumlah Byte	Instruksi							
2	3	Mov direct,direct							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan data dari alamat langsung menuju ke alamat langsung.
Contoh:

Mov P1,P2

P1 adalah register yang terletak di alamat 90H dan P2 adalah register yang terletak di alamat A0H, oleh karena itu instruksi di atas adalah sama dengan Mov 90H,0A0H. Artinya, data di alamat 90H atau di Port 1, dipindah ke alamat 0A0H atau Port 2

Mov direct,@Ri

Siklus	Jumlah Byte	Instruksi							
2	2	Mov direct,@Ri							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan data dari alamat yang ditunjuk oleh Register Index (R0 atau R1) ke alamat langsung
Contoh:

```
Mov    05H,@R0
```

Bila R0 sebelumnya berisi 20H, maka nilai atau data yang tersimpan di alamat 20H akan dipindah ke alamat 05H.

Mov direct,#data

Siklus	Jumlah Byte	Instruksi							
2	3	Mov direct,#data							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan data dari immediate ke alamat langsung.

Mov @Ri,A

Siklus	Jumlah Byte	Instruksi							
1	1	Mov @Ri,A							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan data dari akumulator A menuju ke alamat yang ditunjuk oleh Register Index (R0 atau R1).

Mov @Ri,direct

Siklus	Jumlah Byte	Instruksi							
2	2	Mov @Ri,direct							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan data dari alamat langsung menuju ke alamat yang ditunjuk oleh Register Index (R0 atau R1)

Mov @Ri,#data

Siklus	Jumlah Byte	Instruksi							
1	2	Mov @Ri,#data							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan data immediate menuju ke alamat yang ditunjuk oleh Register Index (R0 atau R1)

Mov DPTR,#data16

Siklus	Jumlah Byte	Instruksi							
2	3	Mov DPTR,#data16							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan data immediate 16 bit menuju ke DPTR.
Contoh:

```
Mov    DPTR,#2000H
```

Data 2000H dalam bentuk 16 bit dipindah ke alamat Register DPTR yang terdiri dari DPH dan DPL

Movc A,@A+DPTR

Siklus	Jumlah Byte	Instruksi							
2	1	Movc A,@A+DPTR							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan data dari memori program yang ditunjuk oleh DPTR dengan indeks akumulator A menuju ke akumulator A. Pada instruksi ini, mikrokontroler akan membaca memori program, yaitu sebuah memori yang ada dalam Flash PEROM AT89C51 atau pada sebuah komponen memori di luar AT89C51 yang di mana input RD dari memori tersebut terhubung dengan PSEN. Sinyal PSEN aktif pada saat instruksi ini dijalankan.
Contoh:

```
Mov    A,#50H
Mov    DPTR,#2000H
Movc   A,@A+DPTR
```

Data yang terletak di komponen memori di luar AT89C51 dan terletak pada alamat 2000H + 50H akan dibaca dan hasilnya disimpan di akumulator A

Movc A,@A+PC

Siklus	Jumlah Byte	Instruksi							
2	1	Movc A,@A+PC							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan data dari memori program yang ditunjuk oleh PC (Program Counter) dengan indeks akumulator A menuju ke akumulator A. Pada instruksi ini, mikrokontroler akan membaca memori program, yaitu sebuah memori yang ada dalam Flash PEROM AT89C51 atau pada sebuah komponen memori di luar AT89C51 yang di mana input RD dari memori tersebut terhubung dengan PSEN. Sinyal PSEN aktif pada saat instruksi ini dijalankan.
Contoh:

```
Mov    A,#50H
Movc   A,@A+PC
```

Apabila pada saat itu Program Counter sedang berada di alamat 2100H, maka data yang terletak di komponen memori di luar AT89C51 dan terletak pada alamat 2100H + 50H akan dibaca dan hasilnya disimpan di akumulator A

Movx A,@Ri

Siklus	Jumlah Byte	Instruksi
2	1	Movx A,@Ri

Flag	C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan data dari memori eksternal (komponen memori di luar AT89C51) ke akumulator A
Jalur alamat dan data dari memori tersebut terhubung melalui Port 0 dan proses pengiriman alamat serta pengambilan data dilakukan secara bergantian (multiplex)
Alamat dari memori terdiri dari 8 bit sehingga kapasitas maksimal dari memori adalah 256 byte.
Contoh:

```
Mov   R0,#50H
Movx  A,@R0
```

Data di alamat 50H dari memori eksternal dipindah ke akumulator A.

Movx A,@DPTR

Siklus	Jumlah Byte	Instruksi							
2	1	Movx A,@DPTR							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan data dari memori eksternal yang ditunjuk oleh DPTR menuju ke akumulator A. Berbeda dengan MOVC, pada saat instruksi ini dijalankan, sinyal yang muncul adalah sinyal RD (bukan PSEN), sehingga memori eksternal di mana input RDnya terhubung dengan RD dari AT89C51 yang terbaca.
Contoh:

```
Mov   DPTR,#2500H
Movx  A,@DPTR
```

Data yang terletak di alamat 2500H dipindah ke akumulator A

Movx @Ri,A

Siklus	Jumlah Byte	Instruksi							
2	1	Movx @Ri,A							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan data dari akumulator A ke memori eksternal yang ditunjuk oleh Register Indeks.
Jalur alamat dan data dari memori tersebut terhubung melalui Port 0 dan proses pengiriman alamat serta pengiriman data dilakukan secara bergantian (multiplex)
Alamat dari memori terdiri dari 8 bit sehingga kapasitas maksimal dari memori adalah 256 byte.
Contoh:

```
Mov   R0,#60H
Movx  @R0,A
```

Data di akumulator A dipindah ke alamat yang ditunjuk oleh R0 pada memori eksternal.

Movx @DPTR,A

Siklus	Jumlah Byte	Instruksi

2	1	Movx @DPTR,A							
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**		**P**

Melakukan pemindahan data dari akumulator A ke alamat yang ditunjuk oleh DPTR pada memori eksternal.

Contoh:

```
Mov   A,#55H
Mov   DPTR,#3000H
Movx  @DPTR,A
```

Data 55H pada akumulator A dipindah ke alamat 3000H yang terletak pada memori eksternal

Push direct

Siklus	**Jumlah Byte**	**Instruksi**							
2	2	Push direct							
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**		**P**

Melakukan penyimpanan data dari suatu register atau memori ke dalam stack. Lokasi data dalam stack tersebut ditunjuk oleh stack pointer. Pada saat instruksi ini dijalankan, nilai dari stack pointer akan bertambah satu dan register atau memori yang di PUSH akan masuk ke dalam alamat yang ditunjuk oleh stack pointer tersebut.

Contoh:

```
Push   A
```

Isi dari SP sebelumnya adalah 09H, maka SP akan bertambah 1 menjadi 0AH dan nilai akumulator A akan tersimpan di alamat 0AH.

Pop direct

Siklus	**Jumlah Byte**	**Instruksi**							
2	2	Pop direct							
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**		**P**

Melakukan pengambilan data dari dalam stack ke suatu register atau memori. Lokasi data dalam stack tersebut ditunjuk oleh stack pointer. Pada saat instruksi ini dijalankan, data di alamat yang ditunjuk oleh stack pointer akan dipindah ke register atau memori dilanjutkan dengan pengurangan nilai stack pointer dengan 1.

Contoh:

```
Pop    B
```

Isi dari SP sebelumnya adalah 08H, maka data yang tersimpan di alamat 08H akan dipindah ke Register B dilanjutkan dengan pengurangan nilai SP menjadi 07H.

XCH A,Rn

Siklus	**Jumlah Byte**	**Instruksi**
1	1	Xch A,Rn

Flag	C	AC	F0	RS1	RS0	OV		P

Menukar data yang tersimpan di akumulator A dengan Rn (R0…R7)

XCH A,direct

Siklus	Jumlah Byte	Instruksi						
1	2	Xch direct						
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**	**P**

Menukar data yang terseimpan di akumulator A dengan alamat langsung.
Contoh:

XCH A,70H

Data di akumulator A ditukar dengan data di alamat 70H dari RAM Internal.

XCH A,@Ri

Siklus	Jumlah Byte	Instruksi						
1	1	Xch A,@Ri						
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**	**P**

Menukar data yang tersimpan di akumulator A dengan alamat yang ditunjuk oleh Register Index (R0 atau R1)
Contoh:

XCH A,@R0

Data di akumulator A ditukar dengan data di alamat yang ditunjuk oleh R0

XCHD A,@Ri

Siklus	Jumlah Byte	Instruksi						
1	1	XCHD A,@Ri						
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**	**P**

Menukar data nibble rendah dari akumulator A dengan nibble rendah dari data yang ditunjuk oleh Register Indeks (R0 atau R1). Nibble tinggi tidak berubah.
Contoh:

XCHD A,@R0

Bila sebelumnya akumulator A berisi 15H, R0 berisi 20H dan isi dari RAM internal alamat 20H adalah 41H, maka setelah instruksi ini dijalankan isi dari akumulator A berubah menjadi 11H dan isi RAM Internal di alamat 20H menjadi 45H.

Manipulasi Bit

CLR C

Siklus	Jumlah Byte	Instruksi						
1	2	CLR C						
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**	**P**
		X						

Clear Carry Flag atau mengubah bit Carry Flag menjadi 0.

CLR bit

Siklus	Jumlah Byte	Instruksi							
1	2	CLR Bit							
Flag		C	AC	F0	RS1	RS0	OV		P

Clear bit atau mengubah bit-bit pada RAM Internal ataupun register yang dapat dialamati secara bit (bit addressable) menjadi 0.
Contoh:

```
Clr     P1.2
```

SETB C

Siklus	Jumlah Byte	Instruksi							
2	3	SETB C							
Flag		C	AC	F0	RS1	RS0	OV		P
		X							

Set Carry Flag atau mengubah bit Carry Flag menjadi 1

SETB bit

Siklus	Jumlah Byte	Instruksi							
2	2	SETB bit							
Flag		C	AC	F0	RS1	RS0	OV		P

Set bit atau mengubah bit-bit pada RAM Internal maupun register yang dapat dialamat secara bit (bit addressable) menjadi 1
Contoh:

```
Setb    A.7
```

Bit ke 7 dari akumulator A diubah menjadi 1, bila sebelumnya nilai akumulator A adalah 02H atau 00000010b maka setelah instruksi ini dijalankan, nilai akumulator A akan menjadi 82H atau 10000010b.

CPL C

Siklus	Jumlah Byte	Instruksi							
2	2	CPL C							
Flag		C	AC	F0	RS1	RS0	OV		P
		X							

Melakukan komplemen pada bit carry flag, apabila sebelumnya bit carry flag adalah 0 maka setelah instruksi ini dijalankan maka bit carry flag akan berada pada posisi set atau 1 demikian pula sebaliknya.

CPL bit

Siklus	Jumlah Byte	Instruksi
1	1	CPL bit

Flag	C	AC	F0	RS1	RS0	OV		P

Melakukan komplemen pada bit pada register atau memori yang dapat dialamati secara bit (bit addressable), apabila sebelumnya bit pada memori atau register tersebut adalah 0 maka setelah instruksi ini dijalankan maka bit pada memori atau register tersebut akan berada pada posisi set atau 1 demikian pula sebaliknya.

ANL C,bit

Siklus	Jumlah Byte	Instruksi							
2	2	ANL C,bit							
Flag		C	AC	F0	RS1	RS0	OV		P
		X							

Melakukan operasi AND antara bit carry flag dan bit pada register atau memori yang dapat dialamati secara bit (bit addressable)

ANL C,/bit

Siklus	Jumlah Byte	Instruksi							
2	2	ANL C,/bit							
Flag		C	AC	F0	RS1	RS0	OV		P
		X							

Melakukan operasi AND antara bit carry flag dengan komplemen dari bit pada register atau memori yang dapat dialamati secara bit (bit addressable)
Contoh:

```
ANL  C,/A.7
```

Bila sebelumnya bit carry flag adalah 0 dan nilai akumulator A adalah 80H atau 10000000b, maka 0 pada bit carry flag akan di AND dengan 1 pada bit ketujuh akumulator A dan menghasilkan 0. Hasil AND ini disimpan pada bit carry flag, sehingga nilai carry flag sesudah instruksi ini dijalankan adalah tetap 0.

ORL C,bit

Siklus	Jumlah Byte	Instruksi							
2	2	ORL C,/bit							
Flag		C	AC	F0	RS1	RS0	OV		P
		X							

Melakukan operasi OR antara bit carry flag dengan komplemen dari bit pada register atau memori yang dapat dialamati secara bit (bit addressable)
Contoh:

```
ORL  C,/A.7
```

Bila sebelumnya bit carry flag adalah 0 dan nilai akumulator A adalah 80H atau 10000000b, maka 0 pada bit carry flag akan di OR dengan 1 pada bit ketujuh akumulator A dan menghasilkan 0. Hasil OR ini disimpan pada bit carry flag, sehingga nilai carry flag sesudah instruksi ini dijalankan adalah tetap 0.

MOV C,bit

Siklus	Jumlah Byte	Instruksi							
1	2	MOV C,bit							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan dari bit pada register atau memori yang dapat dialamati secara bit (bit adressable) ke bit carry flag.

Contoh:

```
Mov    C,A.0
```

Apabila nilai akumulator A adalah 01H atau 00000001 maka bit ke 0 dari akumulator A, yaitu 0 akan dipindah ke bit carry flag sehingga nilai dari bit ini adalah 0.

MOV bit,C

Siklus	Jumlah Byte	Instruksi							
2	2	MOV bit,C							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pemindahan dari bit carry flag ke bit pada register atau memori yang dapat dialamati secara bit (bit adressable).

Contoh:

```
Mov    A.1,C
```

Apabila nilai akumulator A adalah 01H atau 00000001b dan nilai bit carry flag adalah 1 atau set maka nilai 1 pada bit carry flag akan dipindah ke bit ke 1 dari akumulator A sehingga nilai akumulator A akan berubah menjadi 03H atau 00000011b.

JC rel

Siklus	Jumlah Byte	Instruksi							
2	2	JC rel							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan lompatan ke suatu alamat yang didefinisikan apabila carry flag set. Apabila carry flag clear maka program akan menjalankan instruksi selanjutnya.

Contoh:

```
      Jc      Alamat1
      Mov     A,#05H
Alamat1:
      Mov     R1,#00H
```

Apabila carry flag set, maka program akan lompat label alamat 1 dan menjalankan instruksi Mov R1,#00H, namun bila carry flag clear maka program akan menjalankan instruksi Mov A,#05H terlebih dahulu sebelum menjalankan instruksi di label alamat 1.

JNC rel

Siklus	Jumlah Byte	Instruksi
2	2	JNC rel

Flag	C	AC	F0	RS1	RS0	OV		P

Melakukan lompatan ke suatu alamat yang didefinisikan apabila carry flag clear. Apabila carry flag set maka program akan menjalankan instruksi selanjutnya.

Contoh:

```
        Jnc     Alamat1
        Mov     A,#05H
Alamat1:
        Mov     R1,#00H
```

Apabila carry flag clear, maka program akan lompat label alamat 1 dan menjalankan instruksi Mov R1,#00H, namun bila carry flag set maka program akan menjalankan instruksi Mov A,#05H terlebih dahulu sebelum menjalankan instruksi di label alamat 1.

JB bit,rel

Siklus	Jumlah Byte	Instruksi							
2	3	JB bit,rel							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan lompatan ke suatu alamat yang didefinisikan apabila bit dari register atau memori yang dapat dialamati secara bit (bit addressable) set. Apabila bit tersebut clear maka program akan menjalankan instruksi selanjutnya.

Contoh:

```
        Jb      P1.0,Alamat1
        Mov     A,#05H
Alamat1:
        Mov     R1,#00H
```

Apabila bit tersebut set, maka program akan lompat label alamat 1 dan menjalankan instruksi Mov R1,#00H, namun bila bit tersebut clear maka program akan menjalankan instruksi Mov A,#05H terlebih dahulu sebelum menjalankan instruksi di label alamat 1.

JNB bit,rel

Siklus	Jumlah Byte	Instruksi							
2	3	JNB bit,rel							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan lompatan ke suatu alamat yang didefinisikan apabila bit dari register atau memori yang dapat dialamati secara bit (bit addressable) clear. Apabila bit tersebut set maka program akan menjalankan instruksi selanjutnya.

Contoh:

```
        Jb      P1.0,Alamat1
        Mov     A,#05H
Alamat1:
        Mov     R1,#00H
```

Apabila bit tersebut clear, maka program akan lompat label alamat 1 dan menjalankan instruksi Mov R1,#00H, namun bila bit tersebut set maka program akan menjalankan instruksi Mov A,#05H terlebih dahulu sebelum menjalankan instruksi di label alamat 1.

JBC bit,rel

Siklus	Jumlah Byte	Instruksi							
2	3	JBC bit,rel							
Flag		C	AC	F0	RS1	RS0	OV		P

Sama dengan instruksi Jb bit,rel , namun terdapat proses clear pada bit tersebut sesudah lompatan dilakukan

Contoh:

```
JBC   A.7,Alamat1
```

Apabila bit ketujuh dari akumulator A set, maka lompat ke alamat 1 dan sekaligus mengubah kondisi bit ketujuh dari akumulator A menjadi clear.

Percabangan

ACALL addr11

Siklus	Jumlah Byte	Instruksi							
2	2	ACALL Addr11							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan lompatan ke suatu subroutine yang ditunjuk oleh alamat pada addr11. Lompatan yang dapat dilakukan berada di area sebesar 2K byte.

Proses yang terjadi pada saat instruksi ini dikerjakan adalah sebagai berikut:

- Data pada Program Counter + 2 yang merupakan alamat program saat kembali dari subroutine disimpan ke dalam stack
- Stack pointer bertambah 2 kali
- Melakukan lompatan ke alamat yang ditunjuk oleh addr11 dengan mengisi Program Counter dengan alamat tersebut. Alamat yang diisikan ke Program Counter hanya 11 bit sehingga lompatan maksimum hanya sebesar 2K byte.

Contoh:

```
2000   Acall   Lompatan1
       ……
       ……
Lompatan1
2100   Mov     A,#00H
```

Data pada Program Counter + 2 yaitu 2002H disimpan pada stack di mana byte tinggi disimpan di alamat yang ditunjuk oleh SP+1 dan byte rendah disimpan di alamat yang ditunjuk oleh SP+2. Apabila sebelumnya posisi SP berada di alamat 10H maka byte tinggi, 20H akan disimpan di alamat 11H dan byte rendah, 02H disimpan di 12H. Kemudian data 11 bit pada alamat lompatan 1 dipindah ke Program Counter.

LCALL addr16

Siklus	Jumlah Byte	Instruksi

2	3	LCALL Addr16							
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**		**P**

Melakukan lompatan ke suatu subroutine yang ditunjuk oleh alamat pada addr16. Lompatan yang dapat dilakukan berada di area sebesar 64K byte.
Proses yang terjadi pada saat instruksi ini dikerjakan adalah sebagai berikut:

- Data pada Program Counter + 2 yang merupakan alamat program saat kembali dari subroutine disimpan ke dalam stack
- Stack pointer bertambah 2 kali
- Melakukan lompatan ke alamat yang ditunjuk oleh addr16 dengan mengisi Program Counter dengan alamat tersebut. Alamat yang diisikan ke Program Counter adalah 16 bit sehingga lompatan maksimum dapat mencapai 64K byte.

Contoh:

```
2000   Lcall   Lompatan1
       ……
       ……
Lompatan1
3000   Mov     A,#00H
```

Data pada Program Counter + 2 yaitu 2002H disimpan pada stack di mana byte tinggi disimpan di alamat yang ditunjuk oleh SP+1 dan byte rendah disimpan di alamat yang ditunjuk oleh SP+2. Apabila sebelumnya posisi SP berada di alamat 10H maka byte tinggi, 20H akan disimpan di alamat 11H dan byte rendah, 02H disimpan di 12H. Kemudian data 16 bit pada alamat lompatan 1 dipindah ke Program Counter yaitu 3000H.

RET

Siklus	**Jumlah Byte**	**Instruksi**							
2	1	RET							
Flag		**C**	**AC**	**F0**	**RS1**	**RS0**	**OV**		**P**

Melakukan lompatan ke alamat yang disimpan dalam SP dan SP-1. Instruksi ini biasa digunakan pada saat kembali dari subroutine yang dipanggil dengan instruksi ACALL atau LCALL.
Proses yang terjadi adalah sebagai berikut:

- Isi dari alamat yang ditunjuk oleh stack pointer dipindah ke Program Counter nibble tinggi
- Stack pointer berkurang 1
- Isi dari alamat yang ditunjuk oleh stack pointer dipindah ke Program Counter nibble rendah
- Stack pointer berkurang 1

Contoh:

```
2000   Lcall   Lompatan1
2002   ……..…
       ……
       ……
Lompatan1
```

3000 Mov A,#00H
3002 Ret

Saat instruksi RET dijalankan maka data 20H di stack pointer dipindah ke Program Counter nibble tinggi, dan data 02H di stack pointer –1 dipindah ke Program Counter nibble rendah, sehingga isi dari Program Counter menjadi 2002H dan otomatis program akan menjalankan instruksi di alamat 2002H.

RETI

Siklus	Jumlah Byte	Instruksi							
2	1	RETI							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan lompatan ke alamat yang disimpan dalam SP dan SP-1 dan mengembalikan kondisi flag-flag interrupt agar interrupt berikutnya dengan prioritas yang sama dapat dilakukan.

Pada saat terjadi interrupt, data pada PC+2 yang merupakan alamat tempat program harus kembali setelah proses interrupt selesai dilakukan tersimpan dalam SP dan SP-1, maka setelah instruksi RETI dijalankan, alamat pada SP dan SP-1 dipindah ke Program Counter dan program melompat ke alamat tempat akhir instruksi yang sedang dijalankan saat interrupt terdeteksi.

AJMP addr11

Siklus	Jumlah Byte	Instruksi							
2	2	AJMP Addr11							
Flag		C	AC	F0	RS1	RS0	OV		P

Absolute Jump, melompat dan menjalankan program yang berada di alamat yang ditentukan oleh addr11. Proses yang terjadi adalah, 11 bit dari alamat yang ditentukan oleh addr11 dipindah ke Program Counter sehingga program akan langsung menjalankan instruksi yang ada pada alamat tersebut.

Contoh:

```
        AJMP  Lompatan1
        Mov   A,#05H
Lompatan1:
        Mov   R0,#00H
```

Program akan melompat ke alamat lompatan 1 dan menjalankan instruksi Mov R0,#00H.tanpa melalui instruksi MOV A,#05H

LJMP addr16

Siklus	Jumlah Byte	Instruksi							
2	3	LJMP Addr16							
Flag		C	AC	F0	RS1	RS0	OV		P

Long Jump, melompat dan menjalankan program yang berada di alamat yang ditentukan oleh addr16. Proses yang terjadi adalah, 16 bit dari alamat yang ditentukan oleh addr16

dipindah ke Program Counter sehingga program akan langsung menjalankan instruksi yang ada pada alamat tersebut.
Contoh:

```
        LJMP  Lompatan2
        Mov   A,#05H
Lompatan2:
        Mov   R0,#00H
```

Program akan melompat ke alamat lompatan 2 dan menjalankan instruksi Mov R0,#00H.tanpa melalui instruksi MOV A,#05H

SJMP rel

Siklus	Jumlah Byte	Instruksi							
2	2	SJMP rel							
Flag		C	AC	F0	RS1	RS0	OV		P

Short Jump, melakukan lompatan ke alamat yang ditentukan oleh *rel* dengan lompatan maksimum sebesar 128 byte.

JMP @A+DPTR

Siklus	Jumlah Byte	Instruksi							
2	1	JMP @A+DPTR							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan lompatan ke alamat yang dihasilkan oleh penjumlahan antara DPTR dan akumulator A.
Contoh:

```
        Mov   A,#05H
        Mov   DPTR,#2000H
        JMP   A,@A+DPTR
```

Saat instruksi JMP A,@A+DPTR terjadi maka terjadi lompatan ke alamat 2000H + 5H yaitu 2005H dan menjalankan instruksi yang ada di alamat tersebut.

JZ rel

Siklus	Jumlah Byte	Instruksi							
2	2	JZ rel							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan lompatan ke alamat yang ditentukan apabila akumulator A adalah 00H dan langsung meneruskan instruksi dibawahnya bila akumulator A tidak 00H.
Contoh:

```
        JZ    Lompat1
        MOV   A,#07H
Lompat1:
        MOV   B,#00H
```

Apabila nilai akumulator A tidak 00H maka program akan langsung meneruskan instruksi dibawahnya yaitu MOV A,#07H dan program akan menjalankan instruksi di alamat Lompat1 yaitu MOV B,#00H apabila nilai akumulator A adalah 00H.

JNZ rel

Siklus	Jumlah Byte	Instruksi							
2	2	JNZ rel							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan lompatan ke alamat yang ditentukan apabila akumulator A adalah bukan 00H dan langsung meneruskan instruksi dibawahnya bila akumulator A adalah 00H.
Contoh:

```
        JNZ   Lompat1
        MOV   A,#07H
Lompat1:
        MOV   B,#00H
```

Apabila nilai akumulator A adalah 00H maka program akan langsung meneruskan instruksi dibawahnya yaitu MOV A,#07H dan program akan menjalankan instruksi di alamat Lompat1 yaitu MOV B,#00H apabila nilai akumulator A adalah bukan 00H.

CJNE

Instruksi ini melakukan perbandingan antara data tujuan dan data sumber serta melakukan lompatan ke alamat yang ditentukan apabila hasil perbandingan tidak sama.
Bentuk perintah:

```
        CJNE  data tujuan, data sumber, alamat lompatan
```

Carry flag akan set apabila data tujuan lebih kecil dari data sumber.

CJNE A,direct,rel

Siklus	Jumlah Byte	Instruksi							
2	3	CJNE A,direct,rel							
Flag		C	AC	F0	RS1	RS0	OV		P
		X							

Melakukan perbandingan antara akumulator A dan alamat langsung serta melakukan lompatan ke alamat yang ditentukan apabila hasil perbandingan tidak sama.
Contoh:

```
        CJNE  A,00H,lompat1
```

Program akan menuju ke alamat lompat 1 apabila data akumulator A tidak sama dengan data yang ada pada alamat 00H.

CJNE A,#data,rel

Siklus	Jumlah Byte	Instruksi							
2	3	CJNE A,#data,rel							
Flag		C	AC	F0	RS1	RS0	OV		P
		X							

Melakukan perbandingan antara akumulator A dan data immediate serta melakukan lompatan ke alamat yang ditentukan apabila hasil perbandingan tidak sama.
Contoh:

```
CJNE  A,#00H,lompat1
```

Program akan menuju ke alamat lompat 1 apabila data akumulator A tidak sama dengan data 00H..

CJNE Rn,#data,rel

Siklus	Jumlah Byte	Instruksi							
2	3	CJNE Rn,#data,rel							
Flag		C	AC	F0	RS1	RS0	OV		P
		X							

Melakukan perbandingan antara Rn (R0…R7) dan data immediate serta melakukan lompatan ke alamat yang ditentukan apabila hasil perbandingan tidak sama.
Contoh:

```
CJNE  R1,#00H,lompat1
```

Program akan menuju ke alamat lompat 1 apabila data pada R1 tidak sama dengan data yang ada pada alamat 00H.

CJNE @Ri,#data,rel

Siklus	Jumlah Byte	Instruksi							
2	3	CJNE @Ri,#data,rel							
Flag		C	AC	F0	RS1	RS0	OV		P
		X							

Melakukan perbandingan antara data yang terletak pada alamat yang ditunjuk oleh Register Index (R0 atau R1) dan data immediate serta melakukan lompatan ke alamat yang ditentukan apabila hasil perbandingan tidak sama.
Contoh:

```
CJNE  @R1,#00H,lompat1
```

Program akan menuju ke alamat lompat 1 apabila data di alamat yang ditunjuk oleh R1 tidak sama dengan data 00H.

DJNZ Rn,rel

Siklus	Jumlah Byte	Instruksi							
2	2	DJNZ Rn,rel							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pengurangan pada Rn (R0…R7) dengan 1 dan lompat ke alamat yang ditentukan apabila hasilnya bukan 00.
Apabila hasilnya telah mencapai 00, maka program akan terus menjalankan instruksi di bawahnya.
Contoh:

```
Tunggu:
        DJNZ  R7,Tunggu
        RET
```

Selalu melakukan lompatan ke alamat tunggu dan mengurangi R7 dengan 1 selama nilai R7 belum mencapai 00

DJNZ direct,rel

Siklus	Jumlah Byte	Instruksi							
2	3	DJNZ direct,rel							
Flag		C	AC	F0	RS1	RS0	OV		P

Melakukan pengurangan pada data di alamat yang ditunjuk secara langsung dengan 1 dan lompat ke alamat yang ditentukan apabila hasilnya bukan 00.
Apabila hasilnya telah mencapai 00, maka program akan terus menjalankan instruksi di bawahnya.
Contoh:

```
Tunggu:
        DJNZ  07H,Tunggu
        RET
```

Selalu melakukan lompatan ke alamat tunggu dan mengurangi data pada alamat 07H dengan 1 selama nilai pada data yang berada pada alamat 07H belum mencapai 00

NOP

Siklus	Jumlah Byte	Instruksi							
1	1	NOP							
Flag		C	AC	F0	RS1	RS0	OV		P

Instruksi ini berfungsi untuk melakukan tundaan pada program sebesar 1 cycle tanpa mempengaruhi register-register maupun flag.