

Univerzitet u Novom Sadu  
Fakultet tehničkih nauka  
Departman za energetiku, elektroniku i telekomunikacije  
Katedra za elektroniku

# **MIKROPROCESORSKA ELEKTRONIKA**

**Beleške sa predavanja  
#02**

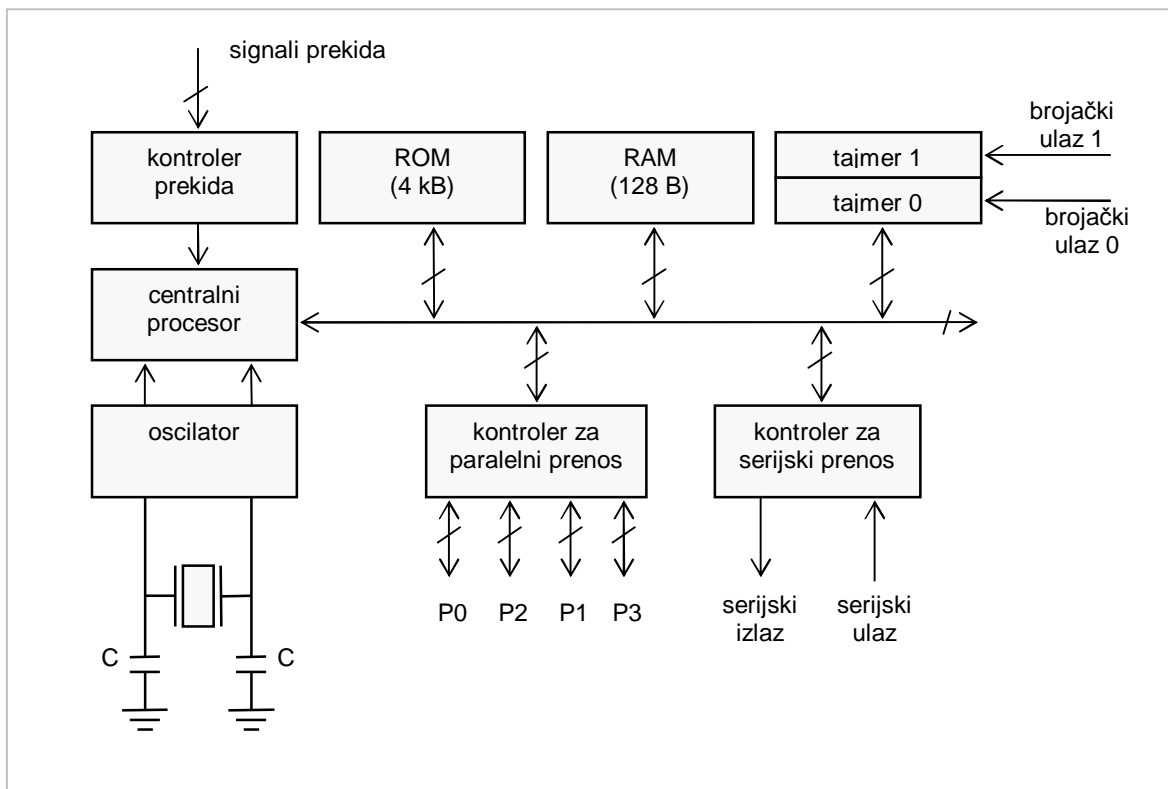
Pripremio: prof. dr Veljko Malbaša  
Novi Sad, oktobar 2007. godine

## Mikrokontroler 8051

Mikrokontroleri su složena integrisana kola koja pored mikroprocesora poseduju i druge funkcionalne jedinice kao što su memorije, kontroleri za paralelni i serijski prenos podataka, tajmeri, pa čak i AD i DA konvertori.

Mikrokontroler 8051 jedna je od retkih mikroproceskih komponenata koja je na tržištu blizu 30 godina i još uvek se koristi. U međuvremenu su se pojavile različite verzije ovog mikrokontrolera, tako da projektanti mogu da izaberu konfiguraciju koja je pogodna za datu primenu.

Opšti blok dijagram mikrokontrolera 8051 dat je na sledećoj slici.



Mikrokontroler je organizovan oko interne magistrale na koju su vezane sve funkcionalne jedinice. Pored 8-bitnog centralnog procesora, na internu magistralu vezani su interni ROM, interni RAM, dva tajmera, kontroler za paralelni prenos i kontroler za serijski prenos podataka. Kontroler prekida prihvata spoljne signale prekida i na osnovu njih generiše signal prekida centralnog procesora. Ugrađeni interni oscilator, na koji se spolja priključuje kvarcni kristal i dva kondenzatora, generiše interne sinhronizacione signale za centralni procesor.

### ROM

U osnovnoj verziji, mikrokontroler ima interni ROM kapaciteta 4 kB koji se koristi za smeštanje programa. Postoje verzije mikrokontrolera koje imaju interni ROM većeg kapaciteta. Umesto ROM-

a često se u mikrokontroler ugrađuje EPROM, koji je veoma pogodan u postupku razvoja mikrorračunarskog sistema. Nek verzije mikrokontrolera 8051 imaju ugrađeni EEPROM ili FLASH memorije za smeštanje programa.

Ukoliko kapacitet internog ROM-a nije dovoljan za datu primenu, projektant može u sistem dodati spoljni ROM, što je objašnjeno kasnije u ovom tekstu.

## **RAM**

U osnovnoj verziji, mikrokontroler ima interni RAM kapaciteta 128 bajtova koji se koristi za smeštanje podataka i promenljivih. Postoje verzije mikrokontrolera koje imaju interni RAM većeg kapaciteta. Ukoliko kapacitet internog RAM-a nije dovoljan, projektant može u sistem dodati spoljni RAM, što je objašnjeno kasnije u ovom tekstu.

Zanimljivo je da interni RAM obuhvata i specijalizovane registre, što je objašnjeno kasnije u ovom tekstu.

## **Tajmeri**

Mikrokontroler ima dva 16-bitna tajmera/brojača koji se koriste za brojanje spoljnih impulsa ili deljenje internog sinhronizacionog signala čime se generišu vremenska kašnjenja za potrebe rada u realnom vremenu. Programabilni registri tajmera nalaze se u memorijskom prostoru mikrokontrolera. Viši bajt registra tajmera 0 označen je sa *TH0* a niži bajt sa *TL0*. Slično su bajtovi registra tajmera 1 označeni sa *TH1* i *TH1*. Upravljački registar tajmera označen je sa *TMOD*.

## **Kontroler za paralelni prenos podataka**

Mikrokontroler ima ugrađen kontroler za paralelni prenos podataka sa 4 (četiri) 8-bitna ulazno-izlazna porta označena sa *P0*, *P1*, *P2* i *P3*. Programabilni registri za prenos podataka preko paralelnih portova nalaze se u memorijskom adresnom prostoru mikrokontrolera. Pošto mikrokontroler nema spoljne linije za prenos adresnih signala i signala podataka, paralelni portovi mogu da se programiraju tako da se koristi za adresnu magistralu i magistralu podataka. Portovi 0 i 2 koriste se za prenos 16-bitne adrese, a korišćenjem vremenskog multipleksa port 0 koristi se i za magistralu podataka.

## **Kontroler za serijski prenos podataka**

Kontroler za serijski prenos podataka podržava dupleksni način rada, sa serijskim izlaznim i serijskim ulaznim signalima. Programabilni registri kontrolera nalaze se u memorijskom adresnom prostoru mikrokontrolera. Upravljački registar serijskog kontrolera označen je sa *SCON* a registar podataka sa *SBUF*.

## **Kontroler prekida**

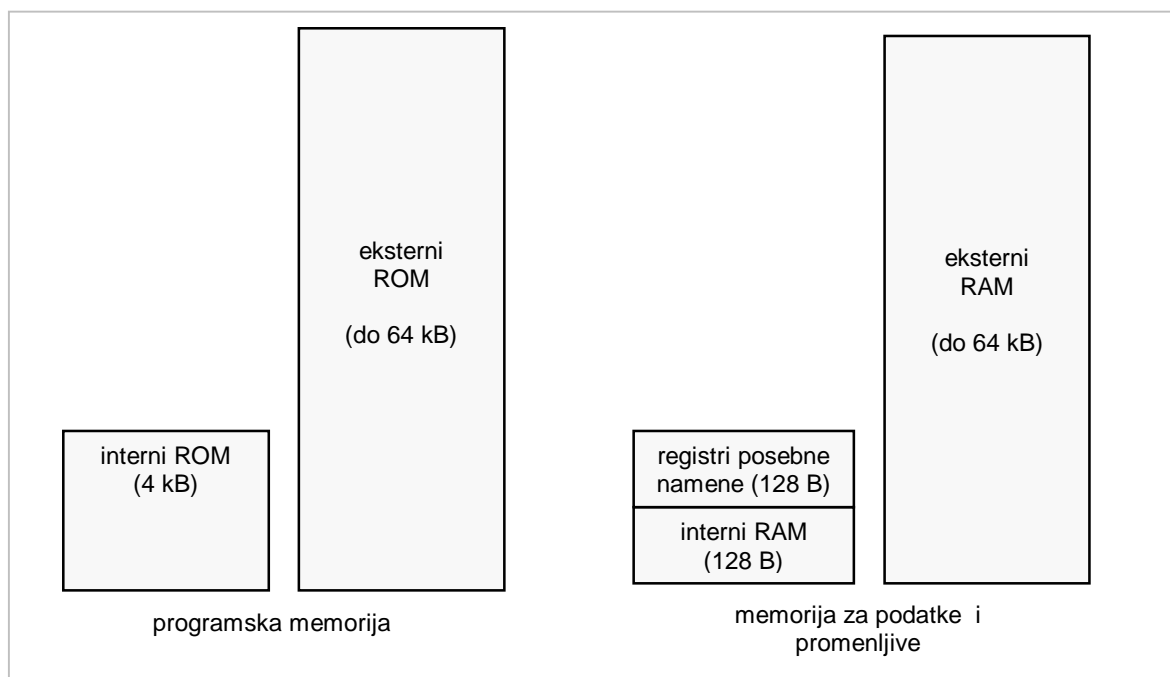
Kontroler prekida prihvata spoljne signale prekida, rešava pitanje prioriteta signala prekida i prosleđuje signal prekida mikroprocesoru. Programabilni registri kontrolera prekida nalaze se u memorijskom adresnom prostoru mikrokontrolera.

## **Memorijski adresni prostor**

Mikrokontroler ima internu memoriju tipa ROM i RAM. Memorija tipa ROM koristi se za smeštanje programa i još se naziva *programska memorija*. Memorijal tipa RAM koristi se za smeštanje podataka i promenljivih. Kao što je rečeno, projektant može po potrebi dodati spoljnu memoriju tipa ROM i/ili tipa RAM. Na ovaj način dolazimo do 4 memorijska bloka koji se nalaze u mikrokontroleru ili su spolja priključeni na mikrokontroler:

- interna programska memorija,
- interna memorija za podatke i promeljive,
- spoljna programska memorija i
- spoljna memorija za podatke i promeljive.

Memorijskih adresni prostor ilustrovan je na sledećem dijagramu.



U donjih 128 bajtova interne memorije za podatke i promenljive smešteni su registri opšte namene, zatim memorijska zona koja može da se adresira po bitovima i slobodna memorijska zona, kao što pokazuje sledeća slika.

Registri opšte namene označeni su sa  $R0$  do  $R7$ . Zanimljivo je da mikrokontroler ima četiri grupe registara opšte namene, ali u datom trenutku aktivna je samo jedna grupa. Aktivnu grupa registara opšte namene određuju dva bita indikatorskog registra.

Memorijska zona sa pristupom pojedinim bitima smeštena je od adrese  $20$  do adrese  $2F$  (heksadecimalno) i ima ukupno  $16 \times 8 = 128$  bita. Ova zona koristi se za predstavljanje prekidačkih promenljivih, koje mogu biti u stanju 0 ili 1. Posebne instrukcije koriste se za adresiranje i pristup pojedinim bitima.

Slobodna memorijska zona nalazi se od adrese  $30$  do  $7F$  (heksadecimalno) i projektant može da je koristi po sopstvenom izboru.

adresa (hex)	namena	opis
7F	slobodna memorijska zona	memorijska zona koja se može proizvoljno koristiti
30 2F	memorijska zona sa pristupom pojedinim bitima	memorijska zona sa (16 x 8) bita koji se mogu pojedinačno adresirati
20 1F	<i>R0 do R7</i>	skup registara opšte namene (3)
18 17	<i>R0 do R7</i>	skup registara opšte namene (2)
10 0F	<i>R0 do R7</i>	skup registara opšte namene (1)
08 07	<i>R0 do R7</i>	skup registara opšte namene (0)
00		

Registri posebne namene smešteni su u interni memorijski adresni prostor kapaciteta 128 B, od adrese 80 do adrese FF, kao što pokazuje slika. Treba imati u vidu da nisu zauzete sve memorijske lokacije, tako na primer, u lokacijama sa adresama 80, 81, 82 i 83 (u heksadecimalnom brojnem sistemu), smešteni su registar *P0* za podatke paralelnog porta 0, pokazivač steka *SP*, donji bajt registra *DPTR* i gornji bajt registra *DPTR*, respektivno. Zatim se memorijske lokacije sa adresama 84 do 86 ne koriste, pa se u lokaciji sa adresom 87 nalazi registar *PCON* za upravljanje potrošnjom energije.

Na slici su prikazane memorijske lokacije koje se koriste za smeštanje registara, a nisu prikazane lokacije koje se ne koriste.

Treba naglasiti da su na slici prikazani registri osnovne verzije mikrokontrolera 8051. Proširene verzije mikrokontrolera imaju dodatne registre i oni su smešteni u lokacijama koje nisu prikazane na slici. Na primer, neke verzije mikrokontrolera imaju 3 tajmera (označene sa 0, 1 i 2). Niži i viši bajtovi tajmera 0 i 1 smešteni su u istim memorijskim lokacijama kao i kod osnovne verzije mikrokontrolera, dok su niži i viši bajtovi tajmera 2 smešteni u memorijskim lokacijama sa adresama *CC* i *CD*, respektivno.

adresa (hex)	naziv registra	namena registra
FF		
F0	B	registar B
E0	A	akumulator
D0	PSW	indikatorski registar
B8	IP	registar za prioritet prekida
B0	P3	paralelni port 3
A8	IE	registar za dozvolu prekida
A0	P2	paralelni port 2
99	SBUF	registar podataka kontrolera za serijski prenos
98	SCON	upravljački registar kontrolera za serijski prenos
90	P1	paralelni port 1
8D	TL1	niži bajt tajmera 1
8C	TH0	viši bajt tajmera 0
8B	TH1	viši bajt tajmera 1
8A	TL0	niži bajt tajmera 0
89	TMOD	registar za vrstu rada tajmera
88	TCON	upravljački registar tajmera
87	PCON	registar za upravljanje potrošnjom energije
83	DPH	gornji bajt registra DPTR
82	DPL	donji bajt registra DPTR
81	SP	pokazivač steka
80	P0	paralelni port 0

## Registri

U ovom odeljku ukratko su opisani registri centralnog procesora, ostali registri opisani su u odeljcima koji se bave ulazno-izlaznim operacijama mikrokontrolera.

Akumulatorski registar *A* ima 8 bita i koristi se kod obavljanja aritmetičkih, logičkih i ulazno izlaznih instrukcija. Mikrokontroler 8051 je akumulatorska mašina, što znači da uvek jedan od operandi mora biti u akumulatoru i rezultat obavljanja neke operacije uvek se smešta u akumulator. Na primer, sledeće instrukcije sabiraju dva broja, koja se nalaze u lokacijama sa labelama *x* i *y*, i rezultat smeštaju u lokaciju sa labelom *z*:

```

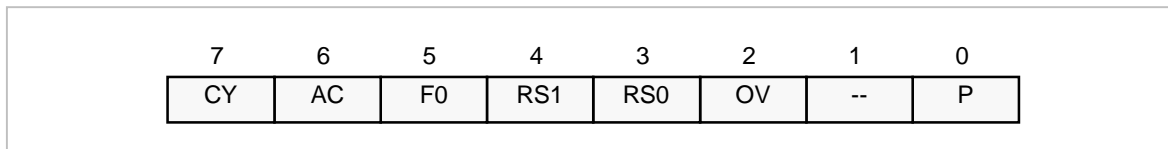
MOV  A, x    ;    A ← x
ADD  A, y    ;    A ← A + y
MOV  z, A    ;    z ← A

```

Osmobitni registar *B* koristi se kao pomoćni akumulator. Pokazivač steka *SP* ima 8 bita i koristi se kod pristupak steku koji je smešten u internoj memoriji za podatke i promenljive. Registar *DPTR*

(Data Pointer) ima 16-bita i koristi se kod nekih indeksnog adresiranja. Programski brojač *PC* ima 16 bita i sadrži adresu naredne instrukcije koja treba da se izvrši.

Indikatorski registar *PSW (Program Status Word)* ima format prikazan na sledećoj slici.



Bit 0, predstavlja bit parnosti i označen je sa *P*. Posle svake instrukcije sadržaj bita *P* pokazuje da li je broj jedinica u akumulatoru paran ili neparan:

*broj jedinica u A je paran* :  $P \leftarrow 0$

*broj jedinica u A je neparan* :  $P \leftarrow 1$

Bitovi *OV*, *CY* i *AC* predstavljaju bit za prekoračenje, prenos i pomoćni prenos, respektivno. Bit *F0* projektant može da koristi ja proizvoljan način.

Bitovi *RS1* i *RS0* određuju aktivni skup registara opšte namene *R0* do *R7* prema sledećoj tabeli:

R0	R1	skup registara opšte namene koji je aktivan
0	0	0
0	1	1
1	0	2
1	1	3

U daljem tekstu objašnjene su osnovne vrste adresiranja koje koristi mikrokontroler 8051. Treba imati u vidu da su neki specifični načini adresiranja primenjeni sa ciljem da se koristi što je moguće manji broj bajtova izvršnog programa. Tako se, na primer, svuda gde je moguće koristi 8-bitna umesto 16-bitna adresa operanda kako bi se instrukcija predstavila sa 2 umesto sa 3 bajta.

## Registarsko inherentno adresiranje

Registarsko inherentno (engleski *Register Inherent*) adresiranje koristi se za adresiranje registara *A*, *DPTR*, *PWS*, *SP*, *PC* i registara opšte namene. Ovi registri adresiraju se neposredno preko koda operacije što znači da za instrukcije koje pristupaju ovim registraima nije neophodan dodatni bajt za adresiranje registra. Na primer, sledeće instrukcija koriste registarsko indirektno adresiranje:

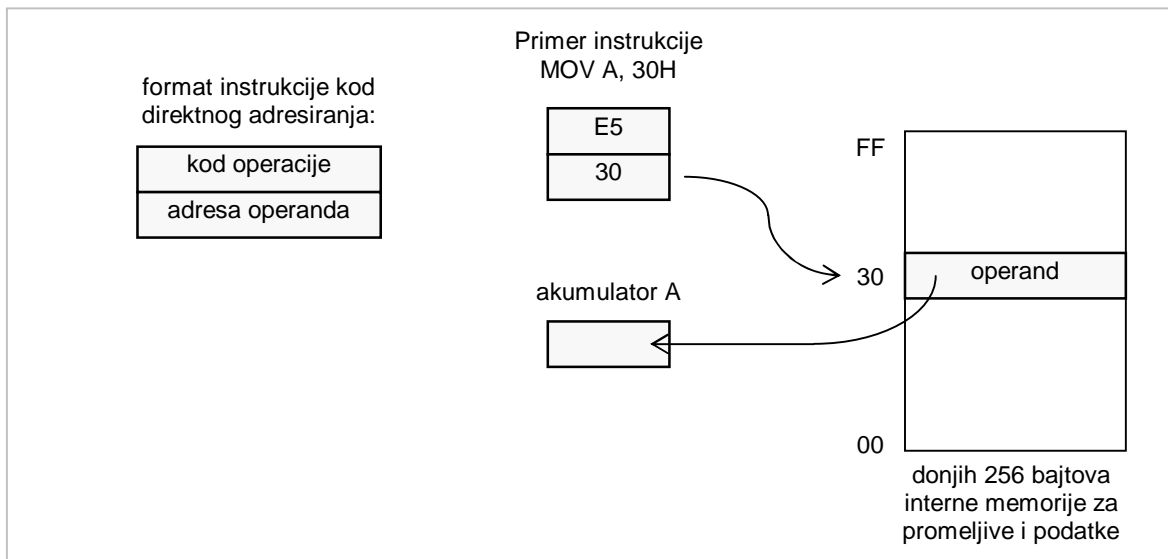
*INC R1* ;  $R1 \leftarrow R1 + 1$   
*MOV A, R2* ;  $A \leftarrow R2$

## Direktno adresiranje

Direktno adresiranje koristi 8-bitnu adresu za pristup operandu koji se nalazi u internoj memoriji za promenljive i podatke. Format instrukcije sastoji se iz polja (prvi bajt) koji sadrži kod operacije i polja (drugi bajt) koji sadrži adresu operanda, kao što to pokazuje sledeća slika (levo). Na desnoj strani grafički je prikazan tok prenosa adrese operanda i operanda u primeru instrukcije:

*MOV A, 30H ;       $A \leftarrow M[30H]$*

Treba primetiti da u navedenoj instrukciji, registar *A* adresira se primenom registerskog inherentnog adresiranja, a operand koji treba smestiti u registar *A* dobija se direktnim adresiranjem.



Kod adresiranja registara posebne namene mogu se koristiti njihve adrese ili simbolička imena. Tako se, na primer, sledeće dve instrukcije prevode i izvršavaju na isti način:

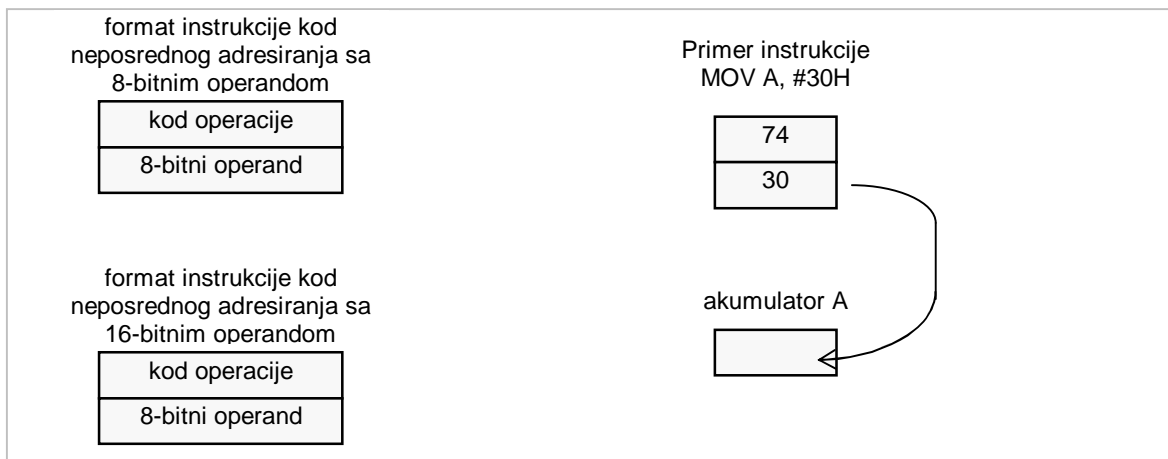
*MOV A, B0H ;       $A \leftarrow M[B0H]$*   
*MOV A, P3 ;       $A \leftarrow M[B0H]$ , jer se P3 nalazi na adresi B0H*

### Neposredno adresiranje

Naposredno (engleski Immediate) adresiranje koristi se za smeštanje 8-bitne ili 16-bitne konstante u registar. Dvobajtna (16-bitna) konstanta koristi se jedino za rad sa registrom *DPTR*, dok svi ostali registri koriste jednobajtni (8-bitni) operand. Neposredno adresiranje označava se znakom #.

Operand (u ovom slučaju konstanta) je deo instrukcije, kao što to prikazuje format instrukcije na sledećoj slici.





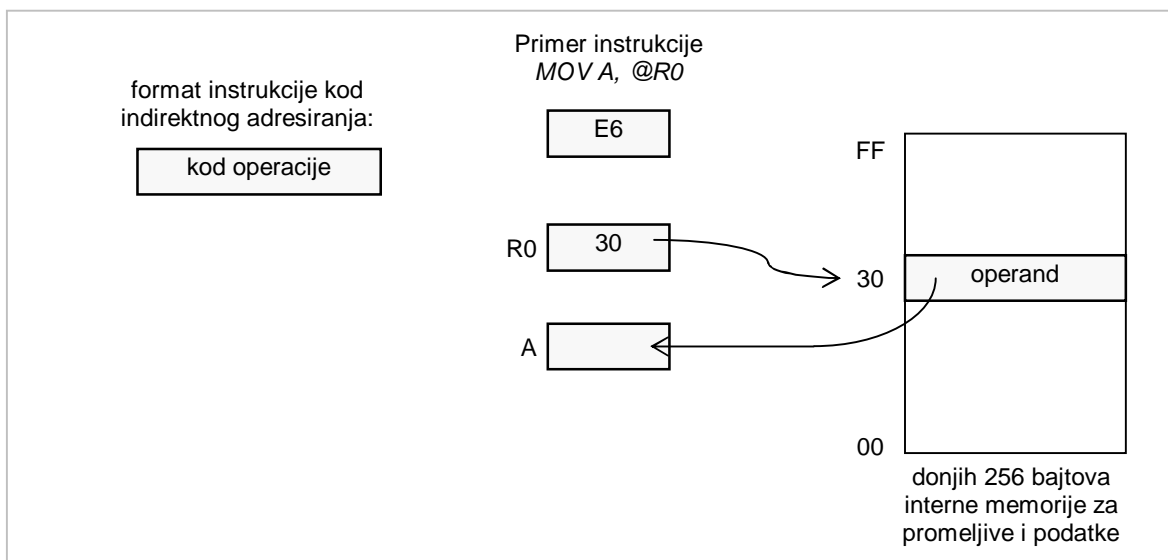
Primeri instrukcija koje koriste neposredno adresiranje:

```
MOV  A, #30H      ;      A ← 30H
ADD  A, #30H      ;      A ← A + 30H
MOV  DPTR, #1330H ;      DPTR ← 1330H
```

## Indirektno adresiranje

Kod indirektnog adresiranja, adresa operanda se nalazi u jednom od registara *R0*, *R1* ili *DPTR*. Registri *R0* ili *R1* koriste se za 8-bitnu adresu, a registar *DPTR* za 16-bitnu adresu. Indirektno adresiranje označava se znakom @.

Operand se nalazi u memorijskoj lokaciji sa adresom koja je u registru *R0*, *R1* ili *DPTR*, kao što je ilustrovano na sledećoj slici.



Primeri instrukcija koje koriste indirektno adresiranje:

```
MOV  A, @R0      ;      A ← M[R0]
MOV  @R1, A      ;      M[R1] ← A
MOVX A, @DPTR    ;      A ← M[DPTR], M[DPTR] je u spljnoj memoriji
```

Poslednja instrukcija sa simboličkim imenom *MOVX* označava da se operand nalazi u spoljnoj memoriji.

## Indeksirano adresiranje

Indeksirano adresiranje koristi bazni registar (programski brojač *PC* ili *DPTR*) na koji se dodaje odstojanje i tako dobija adresa operanda koji kod ove vrste adresiranja mora biti u programskoj memoriji. Primeri instrukcija koje koriste indeksirano adresiranje:

$$\begin{array}{lll} \text{MOVC } A, @A + \text{DPTR} & ; & A \leftarrow M[\text{DPTR} + A] \\ \text{MOVC } A, @A + \text{PC} & ; & A \leftarrow M[\text{PC} + A] \end{array}$$

Slovo *C* u simboličkom imenu instrukcije (*MOVC*) označava pristup programskoj memoriji.

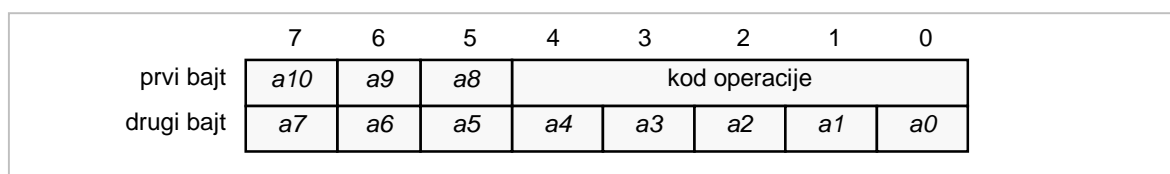
## Relativno adresiranje

Relativno adresiranje koristi se kod uslovnih programskih skokova kod kojih je programski skok u opsegu od  $-128$  do  $+127$  bajtova relativno u odnosu na programski brojač. Mašinska instrukcije sastoji se iz dva bajta, prvi bajt sadrži kod operacije, a drugi bajt relativno odstojanje. Na primer, sledeće instrukcija uslovnog programskog skoka koriste relativno adresiranje:

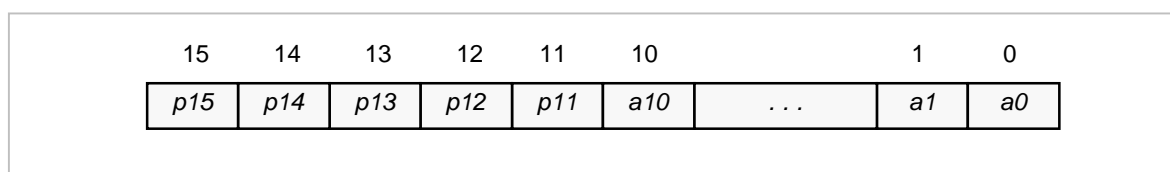
$$\begin{array}{lll} \text{JZ} & \text{labela1} & ; \text{prelazak na instrukciju 'labela1' ako je sadržaj A jednak 0} \\ \text{JC} & \text{kraj} & ; \text{prelazak na instrukciju 'kraj' ako je bit za prenos C = 1} \end{array}$$

## Apsolutno adresiranje

Apsolutno adresiranje koristi se samo kod instrukcija *ACALL* (poziv potprograma sa apsolutnim adresiranjem) i *AJMP* (bezuslovni programski skok sa apsolutnim adresiranjem). Format ove dve instrukcije prikazan je na sledećem dijagramu.



Donjih pet bita prvog bajta sadrži kod operacije, a gornja tri bita označena su sa *a8* do *a10*. Drugi bajt sadrži bite označene sa *a0* do *a7*. Efektivna adresa dobija se tako što se na bite *a0* do *a10* doda gornjih pet bita, *p11* do *p15*, iz programskog brojača, kao što pokazuje sledeća slika.



## Dugačko adresiranje

Dugačko (Long) adresiranje koristi se kod instrukcija *LCALL* (poziv potprograma sa dugačkim adresiranjem) i *LJMP* (bezuslovni programski skok sa dugačkim adresiranjem). Mašinska instrukcija sastoji se iz tri bajta, prvi bajt sadrži kod operacije a sledeća dva bajta 16-bitnu efektivnu adresu. Na primer sledeće instrukcije koriste dugačko adresiranje:

*ACALL potp1* ; poziv potprograma 'potp1' koji može biti smešten bilo gde u programskoj memoriji  
*AJMP kraj* ; prelazak na instrukciju 'kraj' koja može biti bilo gde u programskoj memoriji

## Adresiranje bita

Bitovima u registara posebne pristupa se navođenjem njihovih simboličkih imena. Na primer, bit za prenos *C* indikatorskog registra pristupa se navođenjem njegovog simboličkog imena:

*SETB C* ;  $C \leftarrow 1$   
*CLR C* ;  $C \leftarrow 0$   
*CPL C* ;  $C \leftarrow \neg C$  (invertovanje bita za prenos)

Direktno adresiranje bita koristi se kod pristupa bitovima u zoni interne memorije sa adresama od *20* do *2F* (heksadecimalno) i pristupu bitima većine registara posebne namene. Svaki bit u zoni interne memorije za promenljive i podatke, od adrese *20* do *2F* (heksadecimalno), ima svoju adresu kao što prikazuje sledeća slika.

	7	6	5	4	3	2	1	0
2F	7F	7E	7D	7C	7B	7A	79	78
2E	77	76	75	74	73	72	71	70
...	...							
21	0F	0E	0D	0C	0B	0A	09	08
20	07	06	05	04	03	02	00	01

Sledeće instrukcije koriste direktno adresiranje bitova.

*SETB PSW.7* ;  $C \leftarrow 1$   
*SETB 72H* ; bit 72H (bit broj 2 u lokaciji sa adresom 2EH)  $\leftarrow 1$