Logičke instrukcije i instrukcije pomeranja i rotiranja

Logičke instrukcije

Logičke instrukcije i instrukcije pomeranja i rotiranja

U logičke instrukcije seta instrukcija mikroprocesora 8086 spadaju:

AND, OR, XOR, NOT i TEST

Za ove četiri instrukcije sintaksa je ista:

instrukcija (AND, OR, XOR ili TEST) operand1, operand2

pri čemu operand1 i operand2 mogu biti:

Logičke instrukcije i instrukcije pomeranja i rotiranja

- REGISTAR i memorija,
- memorija i REGISTAR,
- REGISTAR i REGISTAR,
- memorija i eksplicitna vrednost
- REGISTAR i eksplicitna vrednost

Logičke instrukcije i instrukcije pomeranja i rotiranja

PRIMERI:

AND AL, [BX]
AND [BX], AL
AND AX, BX
AND [BX], 11011111b
AND AL, 11011111b

Logičke instrukcije i instrukcije pomeranja i rotiranja

- Sve instrukcije se izvršavaju "bit po bit".
- Za izvršavanje logičkih operacija koriste se već poznate tablice istinitosti za AND, OR i XOR operacije.
- Rezultat instrukcije se smešta u prvi operand.

Logičke instrukcije i instrukcije pomeranja i rotiranja

- Treba voditi računa da operandi imaju istu bitsku dužinu npr. memorijska lokacija i 8-bitni broj ili 8-bitni registar.
- Instrukcija TEST se izvršava tako da se obavi logičko I (AND) "bit po bit", ali se rezultat ne postavlja u prvi operand već se u zavisnosti od dobijene vrednosti postave bitovi stanja (flegovi) ZF, SF i PF.
- Instrukciju TEST obično sledi neka od instrukcija grananja koja se odnosi na pomenute bitove stanja.

Logičke instrukcije i instrukcije pomeranja i rotiranja

NOT

Instrukcija NOT ima sintaksu:

NOT operand1

i invertuje sadržaj "bit po bit".
Operand može biti registar ili memorijska lokacija.

Logičke instrukcije i instrukcije pomeranja i rotiranja

Instrukcije za pomeranje

Sintaksa instrukcija za pomeranje
SHL, SAL, SHR, SAR je:

instrukcija (SHL, SAL, SHR, SAR) operand1, operand2

pri čemu operand1 i operand2 mogu biti:

Logičke instrukcije i instrukcije pomeranja i rotiranja

- memorija i eksplicitna vrednost,
- REGISTAR i eksplicitna vrednost,
- memorija i CL
- REGISTAR i CL

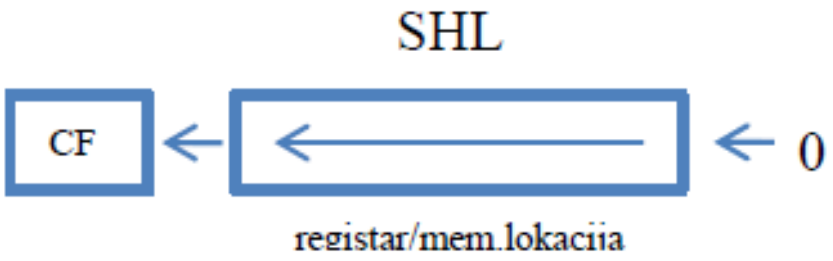
Logičke instrukcije i instrukcije pomeranja i rotiranja

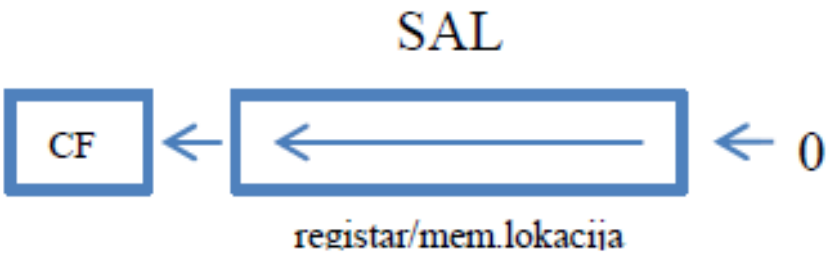
• Instrukcije za pomeranje vrše "bitsko pomeranje" ulevo ili udesno za odredjeni broj bitova.
Zajedničko za sve instrukcije pomeranja je:
• pomeranje se vrši preko flega prenosa CF,
• umesto pomerenog bita umeće se 0,
• pomeranje se vrši za registar ili memorijsku lokaciju koja se navodi kao prvi operand i
• pomeranje se vrši za broj bitova koji se navodi kao drugi operand i može biti eksplicitna vrednost ili broj upisan u CL registar.

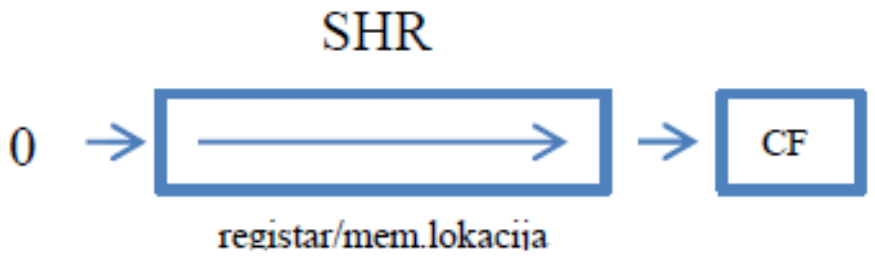
Logičke instrukcije i instrukcije pomeranja i rotiranja

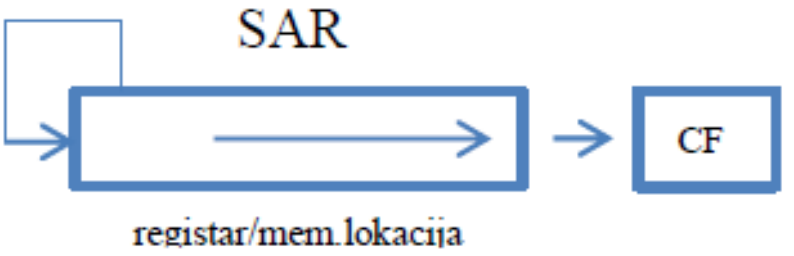
- Instrukcije imaju sledeće značenje:
- SHL - Shift Left - logičko pomeranje ulevo
- SHR - Shift Right - logičko pomeranje udesno,
- SAL - Shift Arithmetic Left - aritmetičko pomeranje ulevo i
- SAR - Shift Arithmetic Right - aritmetičko pomeranje udesno.

Logičke instrukcije i instrukcije pomeranja i rotiranja









Logičke instrukcije i instrukcije pomeranja i rotiranja

Kao što sa prethodnog slajda može zaključiti logičko pomeranje je takvo da se na mesto sa koga se pomeraju binarne cifre upisuju nule, a pomeranje se vrši preko CF flega.

Logičke instrukcije i instrukcije pomeranja i rotiranja

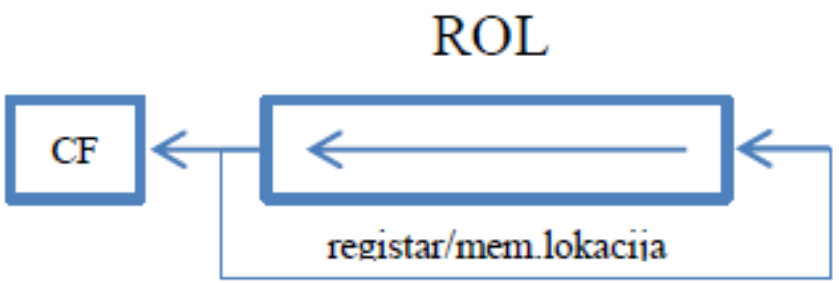
- Aritmetičko pomeranje ulevo je identično logičkom, dok se aritmetičko pomeranje udesno realizuje tako da se bit najveće težine pomera udesno, ali se istovremeno identična vrednost upisuje na mesto bita najveće težine (ne menja se vrednost MSB bita).

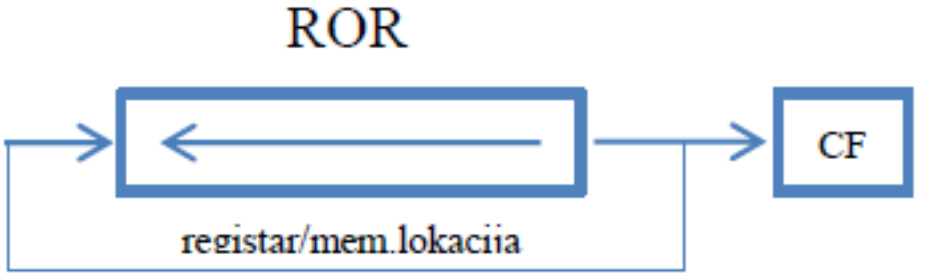
Logičke instrukcije i instrukcije pomeranja i rotiranja

Instrukcije za rotiranje

- Slično pomeranju i kod rotiranja postoje instrukcije za rotiranje ulevo ROL i RCL i za rotiranje udesno ROR i RCR.
- Instukcije ROL i ROR realizuju neposredno rotiranje pri čemu se bit koji se rotira upisuje i u CF fleg i u MSB (prvi bit zdesna) ili LSB (prvi bit sleva) bit.

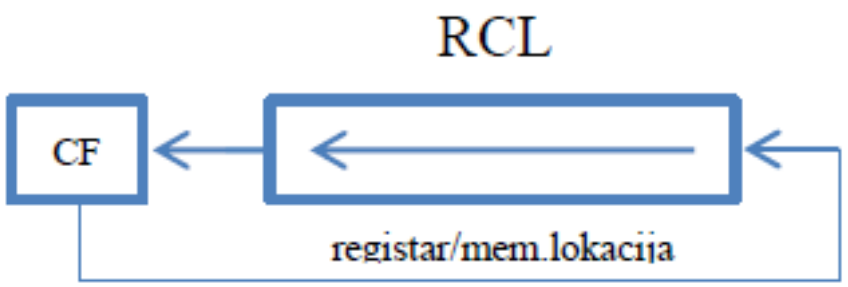
Logičke instrukcije i instrukcije pomeranja i rotiranja

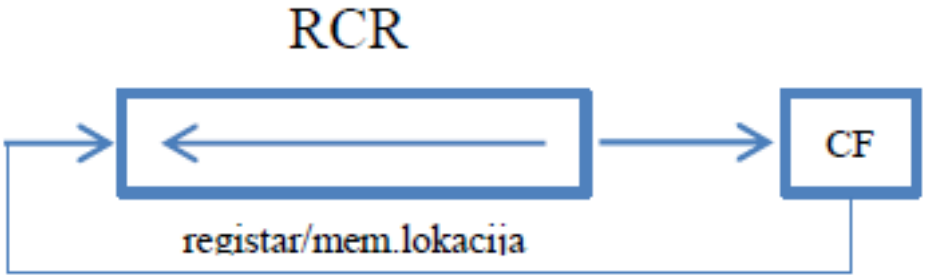




ulevo: MSB -> LSB i CF
udesno: LSB -> MSB i CF

Logičke instrukcije i instrukcije pomeranja i rotiranja





Instrukcije RCL i RCR takodje vrše rotiranje, ali preko CF flega.
ulevo: MSB -> CF a u sledećoj instrukciji CF -> LSB
udesno: LSB -> CF a u sledećoj instrukciji CF -> MSB

Logičke instrukcije i instrukcije pomeranja i rotiranja

Sintaksa instrukcija za rotiranje je:

instrukcija (ROL, ROR, RCL, RCR) operand1, operand2

pri čemu operand1 i operand2 mogu biti:

Logičke instrukcije i instrukcije pomeranja i rotiranja

- memorija i eksplicitna vrednost,
- REGISTAR i eksplicitna vrednost,
- memorija i CL i
- REGISTAR i CL

PRIMERI

ZADATAK 1: Napisati program u asembleru koji će:

- AND instrukcijom resetovati (postaviti na 0) 4 niža bita registra AL, a 4 viša ostaviti nepromenjena,
- OR instrukcijom setovati (postaviti na 1) 4 viša bita registra AL, a 4 niža bita ostaviti nepromenjena,
- XOR instrukcijom zameniti sve nule i jedinice u AX registru i
- NOT funkcijom zameniti sve nule i jedinice u AX registru.

PRIMERI

REŠENJE:

```
MOV AL, 10101101b
AND AL, 11110000b

MOV AL, 10101101b
OR AL, 11110000b

MOV AL, 10101101b
XOR AL, 11111111b

MOV AL, 10101101b
NOT AL
```

PRIMERI

ZADATAK 2: Napisati program u asembleru koji će demonstrirati:

- upotrebu AND instrukcije sa 16-bitnim registrom i neposrednom vrednošću kao operandima,
- upotrebu OR instrukcije sa 8-bitnim registrom i memorijskom lokacijom DS:BX = 2000:0001 kao operandima,
- upotrebu XOR instrukcije sa dva 16-bitna registra kao operandima.

PRIMERI

REŠENJE:

```
MOV AX, 1010110100001111b          ; punimo registar AX
AND AX, 1111000011110000b          ; logicko I sa neposr.vrednoscu
MOV CX, 2000h                      ; pripremamo DS registar
MOV DS, CX                         ; za adresiranje - adresa segmenta
MOV BX, 1h                         ; vrednost pomeraja
MOV DS:[BX], 10101100b             ; upisemo neku vrednost na lokaciju
OR DS:[BX], 11111111b              ; izvrsimo ILI lokacije i neposr.vredn.
MOV AX, 1111000011110000b
MOV BX, 1111111111111111b
XOR AX, BX
```

PRIMERI

ZADATAK 3: Napisati program u asembleru koji:

- postavlja vrednost registra AL na 10001101b,
- instrukcijom TEST AL, 00000000b uzrokuje postavljanje ZF i PF na 1 i
- instrukcijom TEST AL,10000000b uzrokuje postavljanje SF na 1

PRIMERI

REŠENJE:

```
MOV AL, 10001101b
TEST AL, 00000000b
TEST AL, 10000000b
```

PRIMERI

ZADATAK 4: Analizirati program instrukciju po instrukciju.
Posmatrati vrednost flega CF nakon svake instrukcije.

PRIMERI

```
MOV AX, 0F00Fh
MOV CL, 2

SHL AX, CL      ;0C03Ch, CF=1 (bit koji je ispao)
SAL AX, CL      ;0C03Ch, CF=1 (bit koji je ispao)
SHR AX, CL      ;03C03h, CF=1 (bit koji je ispao)
SAL AX, CL      ;0FC03h, CF=1 (bit koji je ispao)
ROR AX, CL      ;0FC03h, CF=1 (poslednji
ROL AX, CL      ;0C03Fh, CF=1 rotirani bit)
RCR AX, CL      ;0BC03h, CF=1 (poslednji
RCL AX, CL      ;0C03Dh, CF=1 izbačeni bit)
```

PRIMERI

ZADATAK 5: Napisati program u asembleru koji će izvršiti množenje osmobitnog broja sa 16 koristeći isključivo instrukcije za pomeranje.

PRIMERI

RESENJE:

```
MOV AX, 0000000011001011b
SHL AX, 4
```

PRIMERI

ZADATAK 6: Napisati program u asembleru koji će izvršiti množenje dva osmobitna broja koristeći isključivo instrukcije za pomeranje i sabiranje.

Izmnožiti brojeve:

$11001011_2 = 203_{10}$ i $01001011_2 = 75_{10}$

$203_{10} * 75_{10} = 15225_{10} = 3B79_{16}$

PRIMERI

RESENJE:

```
MOV AX, 0000000011001011b
MOV BX, AX
SHL BX, 3
ADD BX, AX
SHL BX, 2
ADD BX, AX
SHL BX, 1
ADD BX, AX
```

PRIMERI

ZADATAK 7: Napisati program u asembleru koji će izvršiti celobrojno deljenje 16-bitnog broja sa 16. Koristiti registar AX.

PRIMERI

RESENJE:

```
MOV AX, 0000000011001011b
SHR AX, 4
```

PRIMERI

ZADATAK 8: U registru AX se nalazi 16-bitni broj 0001100 1100 00110 koji predstavlja datum u formatu GGMMDD.
Prvih 7 bitova predstavlja godinu (0-99), sledeća 4 bita predstavljaju mesec (1-12), dok najnižih 5 bitova (0-31) predstavlja dane.
Napisati program u asembleru koji u registre BX, CX i DX redom smešta godine, mesece i dane.

PRIMERI

RESENJE:

```
MOV AX, 0001100110000110b

MOV BX, AX
AND BX, 1111111000000000b
SHR BX, 9

MOV CX, AX
AND CX, 0000000111100000b
SHR CX, 5

MOV DX, AX
AND DX, 0000000000011111b
```