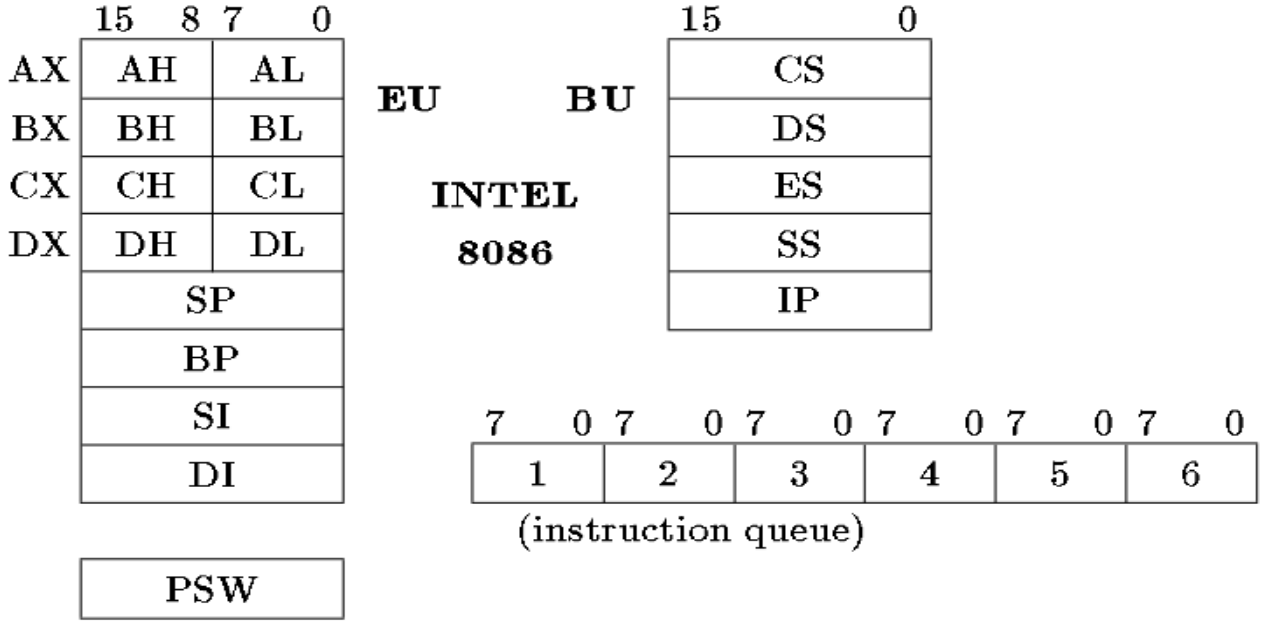
Precesor Intel 8086

Uvod

- Procesor 8086 jedan je od najvažnijih u istoriji računarstva, rodonačelnik Intelove 80x86 familije kojoj pripadaju: 8086, 80286, 80386, 80486, Pentium, Pentium II itd.
- Familija – kompatibilni procesori, mogućnost zadržavanja programa pri prelasku na novi računar

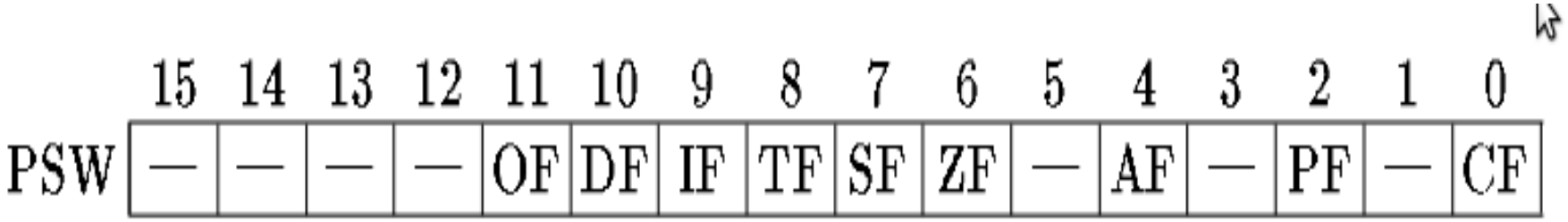
Osnovna arhitektura

- Dva glavna dijela 8086 procesora
  - Interfejs prema magistrali koji čita podatke i naredbe iz memorije
  - Izvršna jedinica koja dekodira i izvršava naredbe



Registri procesora 8086

- Oznake AX=AH:AL, slično za BX, CX, DX
- Registri opšte namjene su AX, AH, AL, BX, BH, BL, CX, CH, CL, DX, DH, DL, SP, BP, SI, DI
- Registar flagova, ProgramStatusWord



Memorija

- Magistrala ima 20 bita, može se adresirati do 2^{20} adresa koje sadrže 8 bita, što znači da je memorija kapaciteta 1MB, osnovne jedinice za rad sa memorijom su bajt i riječ (16 bita)
- Adresa (ukupno 16 bita) dijeli se na dva dijela
  - Adresa segmenta
  - Adresa bajta u segmentu (pomjeraj)
  - U heksadekadnoj notaciji adresa je SSSS:OOOO, na primjer 0040:0002 je adresa trećeg bajta u segmentu 40_{hex}
  - Adresa od 20 bita dobija se množenjem adrese segmenta sa 10$_{hex}$ i dodavanjem pomjeraja

Registri

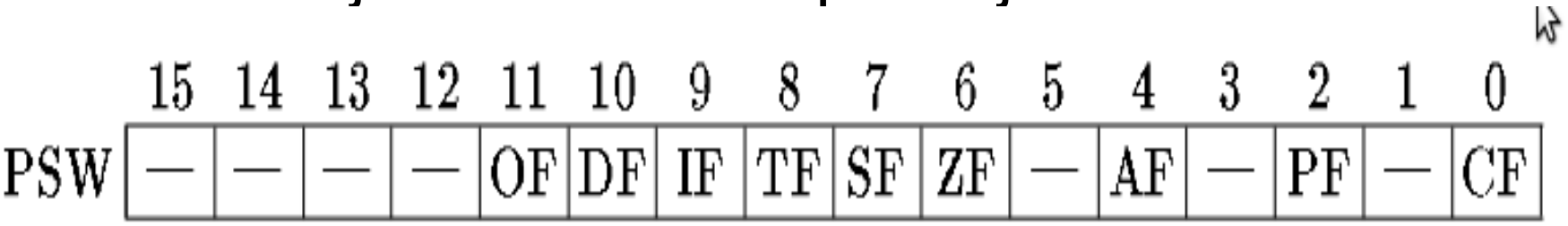
- Procesor 8086 sadrži 14 registara
  - Programski brojač, IP, pokazuje na narednu instrukciju koju procesor treba da izvrši
  - Statusni registar
  - Registri opšte namjene, 4 registra
  - Pokazivački registri, 2 registra
  - Indeksni registri, 2 registra
  - Segmentni registri, 4 registra

Statusni registar

- PSW je 16-bitni registar koji sadrži informacije o rezultatu posljednje izvršene instrukcije
- Flegovi
  - **O**verflow, prekoračenje kod aritmetičkih operacija
  - **D**irection, za kretanje unutar stringova, 0 ka višim, 1 ka nižim adresama
  - **I**nterrupt enable, reagovanje na spoljašnje prekide
  - **T**rap, izvršavanje programa instrukciju po instrukciju, poslije svake instrukcije generiše prekid INT 3

Statusni registar

- Flegovi
  - **S**ign, znak rezultata prethodne instrukcije
  - **Z**ero, rezultat posljednje operacije je nula
  - **A**uxiliary Carry, bit prenosa u BCD aritmetici
  - **P**arity, broj jedinica u rezultatu prethodne instrukcije je paran
  - **C**arry, bit prenosa sa najvišeg mjesta poslije izvršavanja aritmetičkih operacija



Registri opšte namjene

- Mogu se posmatrati kao 16-bitni ili kao par 8-bitnih registara, npr. AX=AH:AL
- AX i AL često se nazivaju akumulatorom, predstavlja podrazumijevani argument nekih instrukcija
- BX, Base Register, bazna adresa prilikom pristupa memoriji
- CX, Counter Regiter, čuva trenutnu vrijednost brojača, implicitno se umanjuje za rad sa petljama
- DX, implicitno se koristi za množenje i

Pokazivački registri

- Pristup podacima sa stek segmenta, dodatno je dozvoljena upotreba u aritmetičko-logičkim operacijama
  - SP, Stack Pointer, pokazivač na vrh steka, prilikom formiranja apsolutne adrese uz njega se podrazumijeva SS (Stack Segment) registar
  - BP, Base Pointer, pristup podacima sa steka, ali bez izmjene steka

Indeksni registri

- Koriste se implicitno prilikom rada sa stringovima, dodatno se mogu koristiti kao registri opšte namjene u aritmetičko-logičkim operacijama
  - SI, Source Index, indeks unutar izvornog stringa, mijenja se implicitno u zavisnosti od smjera, fleg D
  - DI, Destination Index, indeks unutar odredišnog stringa, mijenja se implicitno u zavisnosti od smjera, fleg D

Segmentni registri

- Omogućavaju segmentnu organizaciju memorije, svaki registar definiše blok od 64KB memorije
  - CS (Code Segment) čuva adresu bloka memorije u kojem se nalazi izvršni kod
  - DS (Data Segment) pokazuje na blok memorije za smještanje podataka
  - SS (Stack Segment) pokazuje na segment steka
  - ES (Extra Segment) dodatni segment podataka, koristi se za rad sa stringovima

Metode adresiranja

- Načini adresiranja
  - Neposredno adresiranje
  - Registarsko adresiranje
  - Registarsko indirektno adresiranje
  - Direktno adresiranje
  - Indeksirano adresiranje
  - Bazno adresiranje
  - Bazno indeksirano adresiranje
  - Bazno indeksirano relativno adresiranje

Neposredno adresiranje

- Neposredna dodjela konstante, na primjer:

  mov ah, 34h

  mov ax, 4563h

- Za 16-bitne konstante, u memoriju se prvo smješta njen niži, a zatim viši bajt

Registarsko adresiranje

- Svodi se na preuzimanje sadržaja registra i njegovo smještanje u odredište, na primjer:

```
mov ax, bx
mov si, dx
mov al, dl
mov ds, ax
mov ax, cs
```

Registarsko indirektno adresiranje

- Sadržaj registra, navedenog između uglastih zagrada, tumači se kao adresa na kojoj se nalaze podaci, prilikom upotrebe BX, SI, DI podrazumijeva se DS, za BP koristi se SS na primjer:

```
mov al, [bx]
mov al, [bp]
mov al, cs:[bx]
```

Direktno adresiranje

- Adresa na kojoj se nalaze podaci zadaje se direktno u uglastim zagradama, adresa je 16-bitna, podrazumijeva se DS ragistar, na primjer:

  mov al, [0234]h *(smisao: al ← c(ds:234))*

  mov [0045h], ax

  mov al, cs:[3452h]

Indeksirano adresiranje

- Efektivna adresa dobija se sabiranjem registra i navedene konstante, konstanta je u principu početna adresa bloka kome se pristupa, koristi se DS, na primjer:

  mov al, 20h[si] *(ili: mov al, [si+20h])*

  mov dl, 78[di]

  mov bh, ss:[di+120]

Bazno adresiranje

- Bazni registar je početak bloka, a konstanta relativni položaj elementa kome pristupamo, na primjer:

  mov al, 20h[bx] *(ili: mov al, [bx+20h])*

  mov dl, 120[bp]

  mov bh, cs:[bp+120]

Bazno indeksirano adresiranje

- Podvarijanta indeksiranog adresiranja, ulogu konstante preuzima bazni registar, na primjer:

  mov al, ds:[bp][si] *(ili mov al, ds:[bp+si])*

  mov dl, es:[bx][di]

Bazno indeksirano relativno dresiranje

- Pored baznog i indeksnog registra, koristi se konstanta pomjeraja, na primjer:

  mov al, ds:25h[bp][si] *(ili mov al, ds:[bp+si+25h])*

  mov dl, es:120[bx][di]

Instrukcije

- Instrukcije za prenos podataka
- Aritmetičke instrukcije
- Logičke instrukcije
- Instrukcije pomjeranja i rutiranja
- Upravljačke instrukcije
- Instrukcije za rad sa nizovima
- Instrukcije za rad sa status registrom
- Razne instrukcije

Instrukcije za prenos podataka

- Instrukcije prenosa opšte namjene
- Instrukcije za rukovanje stekom
- Instrukcije konverzije
- Instrukcije za rukovanje periferijom

Instrukcije prenosa opšte namjene

- Instrukcija MOV, dva operanda, prvi odredište, drugi izvor podataka koji treba preniijeti, moguće je vršiti prenos između registara, memorijskih lokacija, ili dodijeliti konstantu
  - MOV reg, {reg | mem | immed}
  - MOV mem, {reg | immed}
  - MOV {reg16 | mem16}, {CS | DS | ES | SS}
  - MOV {DS, ES, SS}, {reg16 | mem16}

Instrukcije prenosa opšte namjene (2)

- Instrukcija LEA, učitavanje efektivne adrese drugog operanda, prvi argument je neki od 16-bitnih registara izuzev segmentnih
- Instrukcije LDS i LES, drugi operand je adresa na kojoj se nalazi 32-bitni pokazivač, prvi operand je registar u koji se smješta ofset, segment se smješta u u jedan od segmentnih registara (DS kod LDS, ES kod LES)

Instrukcije prenosa opšte namjene (3)

- Instrukcija LAHF i SAHF obavljaju razmjenu sadržaja sa status registrom, bez operanada, podrazumijeva se da je AH prvi, a nižih 8 bita status registra drugi operand, LAHF smješta nižih 8 bita u AH, SAHF suprotno
- Instrukcija XCHG zamjenjuje sadržaje dva operanda, prvi operand može da bude memorijska lokacija, drugi obavezno registar, zamjena bez upotrebe treće, pomoćne, promjenljive

Instrukcije za rukovanje stekom

- Stek je procesorski podržana struktura podataka sa LIFO organizacijom, vrh steka je predstavljen parom SS:SP, stek raste ka nižim adresama
- Instrukcija stavljanja na stek PUSH, operand može biti 16-bitni registar ili memorijska lokacija
- Instrukcija uzimanja sa steka POP, operand određuje lokaciju za smještanje podatka, može da bude registar ili memorijska lokacija

Instrukcije konverzije

- Instrukcija CBW za prevođenje broja iz bajta u riječ, svodi se na prenošenje znaka, podrazumijeva da se podataka nalazi u AL
- Instrukcija CDW za prevođenje riječi u duplu riječ, ulaz je AX, izlaz DX:AX

Instrukcije za rukovanje periferijom

- Memorijski preslikan ulaz-izlaz vs instrukcije, periferijama odgovaraju posebne memorijske lokacije – portovi
- Za čitanje vrijednosti IN, vrijednost se smješta u AL ili AX što je prvi operand, drugi operand je 8-bitna konstanta ili DX koji predstavlja broj porta
- Za slanje podataka na odgovarajući port OUT

Aritmetičke instrukcije

- ALJ je 16-bitna, mogućnost rada u BCD aritmetici, svaka četiri bita su kod jedne decimalne cifre
- Otpakovani format – jedna cifra je u jednom 8-bitnom registru
- Spakovani format – dvije cifre u jednom 8-bitnom registru, ne mogu se direktno vršiti operacije množenja i dijeljenja

Instrukcije za sabiranje

- ADD za sabiranje dva operanda, rezultat se smješta u prvi, dozvoljene su kombinacije registara i memorijskih lokacija, drugi operand može biti konstanta
- ADC sabiranje sa prenosom
- INC uvećanje operanda (registar ili memorijska lokacija) za 1

Instrukcije za oduzimanje

- Instrukcija SUB za oduzimanje drugog operanda od prvog, a rezultat smješta u prvi, operandi moraju biti istih dužina

Instrukcija za poređenje

- Instrukcija CMP za poređenja dva operanda, nakon ove instrukcije uvijek se izvršava neka od instrukcija skoka

Instrukcija za komplement

- Nalaženje suprotne vrijednosti sa NEG sa jednim operandom čiju suprotnu vrijednost treba naći

Instrukcije za množenje

- Množenje neoznačenih brojeva MUL, jedan operand je u AX, drugi se eksplicitno navodi, ako je operand dužine 8 bita množi se na AL a rezultat smješta u AX, ako je operand riječ množi se sa AX a rezultat smješta u DX:AX
- Množenje označenih brojeva IMUL

Instrukcije za dijeljenje

- Dijeljenje neoznačenih brojeva DIV, u akumulatoru je dijeljenik, eksplicitno se navodi djelilac, ako je operand dužine bajta tada se AX dijeli sa operandom, količnik je u AL a ostatak u AH, kada je operand riječ, tada se DX:AX dijeli sa operandom, rezultat je u AX a ostatak u DX
- Dijeljenje označenih brojeva IDIV

Logičke instrukcije

- Instrukcija AND, logičko i nad operandima, rezultat se smješta u prvi
- Instrukcija OR za logičko ili
- Instrukcija XOR za ekskluzivno ili
- Instrukcija NOT za negaciju, ima jedan operand za ulaz i izlaz instrukcije

Instrukcije za pomjeranje

- Aritmetičko i logičko pomjeranje
- Instrukcije pomjeranja ulijevo SHL, SAL, prvi operand je vrijednost koju treba pomjeriti, drugi operand je 1 ili registar CL koji sadrži podatak o broju pomjeranja, prilikom pomjeranja nula se dovodi na najnižu poziciju, a bit sa najviše pozicije odlazi u bit prenosa statusnog registra
- Logičko udesno, SHR, bit sa najniže pozicije odlazi u bit prenosa, na najvišu poziciju dolazi nula

Instrukcije za rotiranje

- Rotacija ulijevo, RCL, preuzima se zatečeni bit prenosa, ROL, bez preuzimanja bita prenosa, kod obije instrukcije vrijednost sa najviše pozicije smješta se u bit prenosa, kod RCL u najniži bit upisuje se zatečeni prenos, kod ROL u najniži bit upisuje se vrijednost sa najviše pozicije
- Rotacije udesno, RCR, preuzima se zatečeni bit prenosa, ROR bez preuzimanja zatečenog prenosa

Instrukcije bezuslovnog skoka

- Instrukcija JMP sa jednim operandom koji predstavlja adresu na koju će se izvršiti skok
  - Kratki skok sa rasponom -128 do 127 u odnosu na instrukciju skoka
  - Skok u blizini, unutar istog segmenta
  - Daleki skok, van tekućeg segmenta
- Prilikom prevođenja asembler otkriva da li je skok unutar istog segmenta ili između dva različita segmenta

Instrukcije uslovnog skoka

- Označeni brojevi
  - JL, skok ako je manji
  - JLE, skok ako je manji ili jednak
  - JG, skok ako je veći
  - JGE, skok ako je veći ili jednak
- Neoznačeni brojevi
  - JA, skok ako je veći
  - JBE, skok ako je manji ili jednak
  - JCXZ, skok ako CX = 0

Instrukcije za poziv potprograma

- Pozivanje potprograma CALL, operand je adresa potprograma, implicitno koristi stek za smještanje povratne adrese, CALL near ili far
- Instrukcija RET/RETF za povratak iz potprograma, uzima adresu sa steka
- Instrukcija RET immed, brisanje sa steka

Instrukcije za petlje

- Podrazumijeva se da je brojačka promjenljiva u CX, njena vrijednost se smanjuje za 1, izlaz iz petlje ako je CX = 0
- Instrukcija LOOP, CX ← CX – 1 i skoči na operand ako je CX != 0, dakle sve dok je CX različito od 0, CX se smanjuje za 1 i ponavljaju se naredbe između LOOP i operanda
- Slične su LOOPE, LOOPZ, LOOPNE, LOOPNZ

Instrukcije za rukovanje prekidima

- Softverski prekid INT, operand je broj prekida, prilikom skoka na prekidnu rutinu na stek se smještaju CS, IP, PSW, dok CS, IP dobijaju odgovarajuće vrijednosti
- Povratak iz obrade prekida IRET

Instrukcije za rukovanje nizovima

- Implicitni argumenti su indeksni registri SI i DI, SI je vezan za DS, DI je vezan za ES, instrukcije za rad sa nizovima bajta imaju postfiks B, za rad sa nizovima riječi postfiks W
- LODSB, LODSW učitavanje elementa niza u akumulator
- STOSB, STOSW smještanje u niz
- MOVSB, MOVSW kopiranje elemenata
- CMPSB, CMPSW poređenje elemenata
- Dodavanje prefiksa REP ispred imena instrukcije za formiranje petlji

Razne instrukcije

- Instrukcije za rukovanje status registrom
  - STC, CLC, CMC, STD, CLD, STI, CLI
- Ostale instrukcije
  - NOP za genrisanje kratkih pauza
  - WAIT za usklađivanje procesora i numeričkog koprocesora
  - HLT za zaustavljanje

Program debug.exe

- Asembler i dibager, generiše izvršne .com programe, zauzimaju najviše jedan segment, izvršavanje počinje od (Hex) 0100
- Pokretanje programa sa MS-DOS prompta komandom debug + ENTER, pojavljuje se dibagerov prompt “-”

Program debug.exe (2)

- Naredbe
  - A [adresa] u memoriju se direktno upisuju naredbe ili podaci počev od adrese
  - D [opseg] daje na ekranu sadržaj oblasti memorije i to opseg + 128B
  - R prikazuje sadržaj registar, mogućnost postavljanja vrijednosti
  - G [adresa] pokretanje programa od naznačene adrese
  - T [adresa][broj] izvršava navedenni broj naredbi počev od zadate adrese