Uvod u IA-32 arhitekturu

Milan Banković

10. 10. 2007.

Sadržaj I

Sadržaj II

- Pristup argumentima i lokalnim podacima
- Očuvanje sadržaja registara

Osnovne napomene

- Dizajnirana od strane Intel–a, sedamdesetih i osamdesetih godina
- CISC arhitektura
- Unazad kompatibilna
- Neravnomeran skup instrukcija, mali broj registara
- Veliki broj hardverskih komponenti konstruisan za ovu arhitekturu
- Ogromna količina softvera napisana za ovu arhitekturu
- Dominantna pozicija na tržištu mikroprocesora za personalne računare
- Arhitektura je preuzeta od strane drugih proizvođača (pre svih AMD)

Istorijski pregled

- 4004, 8008, 8080 mikroprocesori
- 8086 procesor (8-bitni)
- 80286 procesor (16-bitni)
- 80386 procesor (32-bitni)
- 80486 procesor (32-bitni). Integrisan FPU.
- 80586 procesor – Pentium
- Pentium MMX procesor – MMX instrukcije
- Pentium Pro i Pentium II – 6. generacija
- Pentium III i Pentium IV procesori

Osnovni koncepti i terminologija

- Nivoi izvršavanja: sistemski vs. aplikativni
- Linearna vs. segmentna organizacija memorije
- *byte, word, doubleword, quadword, double quadword*
- *little-endian* arhitektura
- Podaci ne moraju biti poravnati, ali se preporučuje.
- Značenje podacima u memoriji daju instrukcije koje se nad njima primenjuju.

Tipovi podataka

- Neoznačeni celi brojevi
- Označeni celi brojevi
- Realni brojevi
- Pokazivači
- Stringovi

Registri opšte namene

- EAX, EBX, ECX, EDX
- Moguć pristup nižim delovima registra (npr. AX, AL, AH)
- ESI, EDI, ESP, EBP.
- EFLAGS registar. Flegovi.
- EIP registar.
- Segmentni registri: CS, DS, SS, ES, FS, GS

Adresiranje operanada

- Osnovna formula: $base + scale * index + displacement$
- *base* je bazna adresa (sadržaj nekog od registara)
- *scale* je konstanta (2,4, ili 8)
- *index* je celobrojni indeks (sadržan u nekom registru)
- *displacement* je pomeraj, i predstavljen je konstantom (ili labelom).
- Svaka od ovih komponenti se može izostaviti. Tako dobijamo razne vrste adresiranja.

Izgled memorije

- Zona sa instrukcijama – *text* zona
- Zona sa inicijalizovanim podacima – *data* zona
- Zona sa neinicijalizovanim podacima – *bss* zona
- Zona sistemskog steka
- Zona dinamičke memorije (*heap*).

Sistemski stek

- *LIFO* struktura.
- Procesorska podrška (registar ESP čuva adresu vrha steka)
- Podaci se postavljaju na stek instrukcijom PUSH
- Podaci se skidaju sa steka instrukcijom POP
- Stek raste ka ”nižim” adresama

Pozivanje funkcija

- Poziv funkcije je identičan bezuslovnom skoku, s tim što se pamti adresa povratka
- Pozivajuća struktura postavlja povratnu adresu na stek, a zatim skače na adresu početka pozvane procedure (instrukcija CALL)
- Pozvana procedura skida sa vrha steka povratnu adresu i nakon toga skače na tu adresu (instrukcija RET)
- Neophodno je obezbediti da pre pozivanja funkcije RET skinemo sa steka sve eventualne podatke koji su prilikom izvršavanja funkcije postavljani na stek, kako bi se na vrhu steka nalazila upravo povratna adresa.

Argumenti, povratna vrednost, lokalni podaci

- Argumenti se prenose preko steka, tako što ih postavlja pozivajuća procedura. Pozivajuća procedura je odgovorna i za skidanje argumenata sa steka.
- Konvencija u C–u je da se argumenti na stek postavljaju u obrnutom poretku.
- Povratna vrednost procedure se ostavlja u EAX registru, ako je podatak ceobrojnog ili pokazivačkog tipa.
- Lokalni podaci se postavljaju na stek, nakon ulaska u proceduru. Odgovornost pozvane procedure je i da ih ukloni pre poziva instrukcije RET.

Pristup argumentima i lokalnim podacima

- Pokazivač steka se može menjati, pa se zato treba zapamtiti pozicija vrha steka po ulasku u funkciju.
- Ova adresa se čuva u EBP registru, dok se prethodna vrednost EBP postavlja na stek.
- Nakon ovoga argumenati funkcije počinju od adrese $EBP + 8$, prema višima adresama, dok su lokalni podaci smešteni od adrese $EBP - 4$ prema nižim adresama.
- Deo steka od pozicije na koju pokazuje EBP do vrha steka naziva se *stack frame*
- Automatsko obeležavanje *stack frame*–a – instrukcije ENTER i LEAVE

Očuvanje sadržaja registara

- Pozvana procedura može promeniti sadržaj registara, i time promeniti podatke pozivajuće procedure.
- Pozvana procedura postavlja na stek vrednosti registara koje ima nameru da menja, a koji mogu biti potrebni pozivajućoj proceduri.
- Intrukcijama PUSHA i POPA mogu se brzo sačuvati svi registri opšte namene.
- Po C konvencijama, neophodno je sačuvati EBX, EDI, ESI i EBP.