Asemblersko programiranje za x86_64 arhitekturu

Nikola Milev

Poslednja izmena: 01/12/2017

Sintaksa

Na kursu ćemo koristiti Intelovu neprefiksnu sintaksu.
Svaki red asemblera može biti:

- Prazan red: prazni redovi se ignorišu
- Direktiva
- Komentar: Pri nailasku na simbol #, ostatak linije se ignoriše
- Linije koje nisu ni prazne ni direktive smatraju se instrukcijama
- Svaka linija može početi labelom

Labele

Definicija labele sastoji se iz identifikatora iza kog se navodi simbol :. Identifikator mora početi slovom ili simbolom _, dok može sadržati slova, simbol _ i cifre. Labele se prilikom prevođenja programa prevode u memorijske adrese. Labele mogu označavati adrese podataka, kao i instrukcija.

Direktive

Direktive počinju simbolom . i imaju specijalno značenje.

- *.intel_syntax noprefix* – Označava se da se koristi Intelova neprefiksna sintaksa
- *.globl identifikator* ili *.global identifikator* – Navodi se da je *identifikator* globalni simbol
- *.data* – Počinje se sekcija inicijalizovanih podataka
- *.text* – Počinje se sekcija koda
- *.asciz* – Kreira se ASCI niska na čijem se kraju automatski navodi terminirajuća nula
- *.byte* – Kreira se jedan ili niz bajtova; članovi niza razdvojeni su zapetom
- *.word* – Kreira se jedan ili niz slogova od 2 bajta
- *.long* – Kreira se jedan ili niz slogova od 4 bajta
- *.quad* – Kreira se jedan ili niz slogova od 8 bajtova

Instrukcije

Instrukcija se sastoji od koda operanda i operan(a)da. Svaki kod instrukcije ima svoju simboličku oznaku. Opšti oblik instrukcije sa dva operanda je: $kod\ op1, op2$. Načini zadavanja operanada:

- Registarski operandi: navodi se simbolička oznaka registra
- Neposredni operandi: direktno se navodi vrednost sa kojom se radi
- Memorijski operandi: navodi se adresa na kojoj se nalazi vrednost sa kojom se radi. Opšti sintaksni oblik je: $[B + S * I + D]$. B je bazna adresa, D je pomeraj, I je indeks, dok je I veličina "elementa". Svaki od navedenih elemenata može se izostaviti i tada se dobijaju specijalni slučajevi:
  - $[B]$ – Bazno adresiranje
  - $[B + D]$ – Bazno adresiranje sa pomerajem
  - $[B + S * I]$ – Indeksno adresiranje

Registri

U toku kursa, koristiće se uglavnom registri opšte namene. Njih ima 16:
rax, *rbx*, *rcx*, *rdx*, *rsi*, *rdi*, *rsp*, *rbp*, *r*8, ..., *r*15 Navedeni registri imaju veličinu 8 bajtova (64 bita). Moguć je pristup niža 4 bajta:
eax, *ebx*, *ecx*, *edx*, *esi*, *edi*, *esp*, *ebp*, *r*8*d*, ..., *r*15*d*
Registri *rbp* i *rsp* imaju specijalnu svrhu pri radu sa stekom:

- Registar *rbp* služi za čuvanje trenutnog okvira steka.
- Registar *rsp* služi za čuvanje trenutnog vrha steka. Pri izvršavanju instrukcije *pop*, vrednost registra *rsp* automatski se uvećava za veličinu operanda, dok se pri instrukciji *push* vrednost registra *rsp* automatski umanjuje za veličinu operanda.

Instrukcije

Instrukcije za rad sa stekom

- *push op* – Na vrh steka smešta se vrednost sadržana u operandu. Registar *rsp* umanjuje se za veličinu operanda. Na stek je moguće smestiti samo dvobajtne i osmobajtne podatke.

pop op – Sa vrha steka uklanja se vrednost i smešta se u operand. registar *rsp* uvećava se za veličinu operanda. Sa steka je moguće ukloniti samo dvobajtne i osmobajtne podatke.

Instrukcije transfera

- *mov op*1, *op*2 – U prvi operand smešta se vrednost drugog operanda.
- *lea op*1, *op*2 – U drugi operand smešta se adresa drugog operanda.

Aritmetičke instrukcije

- Sabiranje: *add op*1, *op*2 – sabira argumente i rezultat smešta u prvi argument
- Oduzimanje: *sub op*1, *op*2 – oduzima argumente i rezultat smešta u prvi argument
- Neoznačeno množenje: *mul op* – množi sadržaj registra *rax* operandom *op* i rezultat se smešta u *rdx* : *rax*
- Označeno množenje: *imul op* – množi sadržaj registra *rax* operandom *op* i rezultat se smešta u *rdx* : *rax*
- Proširivanje znaka: cdqe – proširivanje registra *eax* na *rax*
- Proširivanje znaka: cdq – proširivanje registra *rax* na *rdx* : *rax*
- Neoznačeno deljenje: *div op* – neoznačeno deli sadršaj registara *rdx* : *rax* operandom *op*; količnik se nalazi u *rax* dok se ostatak pri deljenju nalazi u *rdx*
- Označeno deljenje: *idiv op* – neoznačeno deli sadršaj registara *rdx* : *rax* operandom *op*; količnik se nalazi u *rax* dok se ostatak pri deljenju nalazi u *rdx*
- Negiranje: *neg op*
- Uvećanje operanda za 1: *inc op*
- Umanjenje operanda za 1: *dec op*

Logičke instrukcije

- Konjunkcija: *and op*1, *op*2
- Disjunkcija: *or op*1, *op*2
- Ekskluzivna disjunkcija: *xor op*1, *op*2
- Negacija: *not op*
- Šiftovanje (pomeranje) u levo: *shl op*1, *op*2 – pomera prvi argument za broj mesta koji je sadržan u drugom argumentu
- Šiftovanje (pomeranje) u desno:
  - Logičko: *shr op*1, *op*2
  - Aritmetičko: *sar op*1, *op*2

Instrukcije poređenja

- *cmp* – poređenje operanada oduzimanjem; ne menja operande nego $rflags$ registar
- *test* – bitovsko poređenje operanada konjukcijom; ne menja operande nego $rflags$ registar

Instrukcije kontrole toka

- Bezuslovni skok: *jmp op* – bezuslovni skok na adresu
- Poziv funkcije: *call op* – bezuslovni skok uz pamćenje povratne adrese na steku
- *ret* – skidanje adrese sa steka i skok na nju
- *jz op* – Skok na adresu ukoliko je rezultat prethodne instrukcije nula
- *je op* – Skok na adresu ukoliko je rezultat prethodnog poređenja jednako (ekvivalentno sa *jz*)
- *jnz op* – Skok na adresu ukoliko rezultat prethodnog poređenja nije nula
- *jne op* – Skok na adresu ukoliko rezultat prethodnog poređenja nije jednako (ekvivalentno sa *jnz*)
- *ja op* – Skok ukoliko je rezultat prethodnog poređenja veće (neoznačeni brojevi)
- *jb op* – Skok ukoliko je rezultat prethodnog poređenja manje (neoznačeni brojevi)
- *jae op* – Skok ukoliko je rezultat prethodnog poređenja veće ili jednako (neoznačeni brojevi)
- *jae op* – Skok ukoliko je rezultat prethodnog poređenja manje ili jednako (neoznačeni brojevi)
- *jg op* – Skok ukoliko je rezultat prethodnog poređenja veće (označeni brojevi)
- *jl op* – Skok ukoliko je rezultat prethodnog poređenja manje (označeni brojevi)
- *jge op* – Skok ukoliko je rezultat prethodnog poređenja veće ili jednako (označeni brojevi)
- *jle op* – Skok ukoliko je rezultat prethodnog poređenja manje ili jednako (označeni brojevi)
- Postoje i negacije navedenih instrukcija: *jna*, *jnb*, *jnae*, *jnbe*, *jng*, *jnl*, *jnge*, *jnle*

Konvencije za pozivanje i pisanje funkcija

- Poziv se vrši instrukcijom *call op*
- Prenos parametara: Celobrojni parametri prenose se redom (s leva na desno) u registrima: *rdi*, *rsi*, *rdx*, *rcx*, *r*8, *r*9. Ukoliko funkciji prenosimo više od 6 argumenata, tada se preostali smeštaju na stek, redom s desna na levo.
- Povratna vrednost nalazi se u *rax* registru.
- Registri koji pripadaju pozvanoj funkciji:
  - *rax*, *rdi*, *rsi*, *rdx*, *rcx*, $r8 - r10$
- Registri koji pripadaju pozivajućoj funkciji:
  - *rbx*, *rbp*, $r12 - r15$
- Ukoliko u našem programu pozivamo neku funkciju, neophodno je da u trenutku poziva adresa vrha steka (sadržana u registru *rsp*) bude deljiva sa 16.
- Pisanje prologa: instrukcija *enter n*, 0 gde je *n* broj bajtova koji odvajamo za lokalne promenljiva
- Pisanje epiloga: instrukcija *leave*, koja vraća stek na stanje u trenutku ulaska u funkciju.
- Povratak iz funkcije: instrukcija *ret*

Prevođenje