Asemblersko programiranje za x86_64 arhitekturu

Nikola Milev

Poslednja izmena: 30/11/2017

Sintaksa

Na kursu emo koristiti Intelovu neprefiksnu sintaksu.
Svaki red asemblera moe biti:

- Prazan red: prazni redovi se ignoriu
- Direktiva
- Komentar: Pri nailasku na simbol #, ostatak linije se ignorie
- Linije koje nisu ni prazne ni direktive smatraju se instrukcijama
- Svaka linija moe poeti labelom

Labele

Definicija labele sastoji se iz identifikatora iza kog se navodi simbol :. Identifikator mora poeti slovom ili simbolom _, dok moe sadrati slova, simbol _ i cifre. Labele se prilikom prevoenja programa prevode u memorijske adrese. Labele mogu oznaavati adrese podataka, kao i instrukcija.

Direktive

Direktive poinju simbolom . i imaju specijalno znaenje.

- *.intel_syntax noprefix* – Oznaava se da se koristi Intelova neprefiksna sintaksa
- *.globl identifikator* ili *.global identifikator* – Navodi se da je *identifikator* globalni simbol
- *.data* – Poinje se sekcija inicijalizovanih podataka
- *.text* – Poinje se sekcija koda
- *.asciz* – Kreira se ASCI niska na ijem se kraju automatski navodi terminirajua nula
- *.byte* – Kreira se jedan ili niz bajtova; lanovi 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 simboliku oznaku. Opti oblik instrukcije sa dva operanda je: *kod op*1, *op*2. Naini zadavanja operanada:

- Registarski operandi: navodi se simbolika 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. Opti sintaksni oblik je: $[B + S * I + D]$. B je bazna adresa, D je pomeraj, I je indeks, dok je I veliina "elementa". Svaki od navedenih elemenata moe se izostaviti i tada se dobijaju specijalni sluajevi:
  - $[B]$ – Bazno adresiranje
  - $[B + D]$ – Bazno adresiranje sa pomerajem
  - $[B + S * I]$ – Indeksno adresiranje

Registri

U toku kursa, koristie se uglavnom registri opte namene. Njih ima 16:
rax, *rbx*, *rcx*, *rdx*, *rsi*, *rdi*, *rsp*, *rbp*, *r*8, ..., *r*15 Navedeni registri imaju veliinu 8 bajtova (64 bita). Mogu je pristup nia 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* slui za uvanje trenutnog okvira steka.
- Registar *rsp* slui za uvanje trenutnog vrha steka. Pri izvravanju instrukcije *pop*, vrednost registra *rsp* automatski se uveava za veliinu operanda, dok se pri instrukciji *push* vrednost registra *rsp* automatski umanjuje za veliinu operanda.

Instrukcije

Instrukcije za rad sa stekom

- *push op* – Na vrh steka smeta se vrednost sadrana u operandu. Registar *rsp* uveava se za veliinu operanda. Na stek je mogue smestiti samo dvobajtne i osmobajtne podatke.

Instrukcije transfera

-

Aritmetike instrukcije

Logike instrukcije

Instrukcije poreenja

Instrukcije kontrole toka

Konvencije za pozivanje funkcija

Konvencije za pisanje funkcija

Mozda ova dva u jedan

Prevoenje