Elektrotehnički fakultet  
Univerzitet u Beogradu

DVOPROLAZNI ASEMBLER  
SISTEMSKI SOFTVER  
Projekat, septembar 2018.

Trifunović Nina  
br.ind: 2014/0221

SADRŽAJ:  
1. Opis problema  
2. Opis rešenja  
3. Uputstvo za prevođenje i pokretanje  
4. Test programi

**1. Opis problema:**

Potrebno je napisati dvoprolazni asembler za procesor napisan u postavci zadatka. Ulaz asemblera je tekstualni fajl u skladu sa sintaksom datom u postavci zadatka, a izlaz asemblera treba da bude predmetni program zapisan u tektualnom fajlu (nalik na ELF format objektnih fajlova). Prilikom generisanja izlaznog fajla treba se voditi principima GNU asemblera, s tim da se sekcije smeštaju jedna za drugom, počev od adrese koja se zadaje iz komandne linije prilikom pokretanja samog asemblera.

**2. Opis rešenja:**

Najpre se učitava ulazni fajl pozivanjem funkcije readFile(), koja za svaku liniju ulaznog fajla poziva funkciju parseLineIntoTokens(), čime se fajl deli na tokene za kasniju obradu. Zatim se kreće na samo asembliranje.  
 Najpre se poziva funkcija firstPass(), u kojoj je obrađen prvi prolaz asemblera. Cilj ovog prolaza jeste kreiranje tabele simbola, te se svaka labela (simbol) na koju se naiđe ubacuje u tabelu simbola, a za instrukcije i direktive u ovom prolazu neophodno je samo uvećati brojač lokacija (locCnt). Zatim se prelazi na drugi prolaz.  
 Sledeća funkcija koja se poziva jeste secondPass(), čiji je cilj implementacija drugog prolaza asemblera. U ovom prolazu popunjavaju se sadržaji svih sekcija koje u fajlu postoje (skup mogućih sekcija čine .text, .data, .rodata i .bss sekcije, i svaka od njih može se u ulaznom fajlu pojaviti najviše jednom, prema zahtevima projektnog zadatka). Jedino u .text sekciji mogu se naći instrukcije. Još jedan cilj ovog prolaza jeste po potrebi kreiranje relokacionih zapisa i popunjavanje tabela relokacionih zapisa za svaku od sekcija koja se pojavljuje u ulaznom fajlu, a u kojoj postoji potreba za istim. Ukoliko se u ovom prolazu naiđe na direktivu .global, a postoji simbol sa zadatim imenom, postavlja se isGlobal atribut na true, čime se označava da je taj simbol globalni i izvozi se. Ukoliko simbol ne postoji, ubacuje se u tabelu simbola, naznačava da on nije definisan u ulaznom fajlu, čime se označava da je taj simbol globalni i uvozi se.  
 Nakon što se završi drugi prolaz, samo asembliranje je završeno i ostaje jedino da se rezultati upišu u izlazni fajl, za šta se koristi funkcija writeToFile(). Najpre se ispisuje sadržaj tabele simbola, a zatim za svaku od sekcija koje su se pojavile u ulaznom fajlu sadržaj njene tabele relokacija, ukoliko su relokacije potrebne, a zatim i sadržaj samih sekcija. Time je u potpunosti završen proces asembliranja i kreiran je traženi objektni fajl.

**3. Uputstvo za prevođenje i pokretanje:**

Najpre je potrebno instalirati gcc i g++, za šta su korišćene sledeće komande:  
 1) sudo add-apt-repository ppa:ubuntu-toolchain-r/test  
 2) sudo apt-get update  
 3) sudo apt-get install gcc-4.9  
 4) sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-4.9 50  
 5) sudo apt-get install g++-4.9  
 6) sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-4.9 50

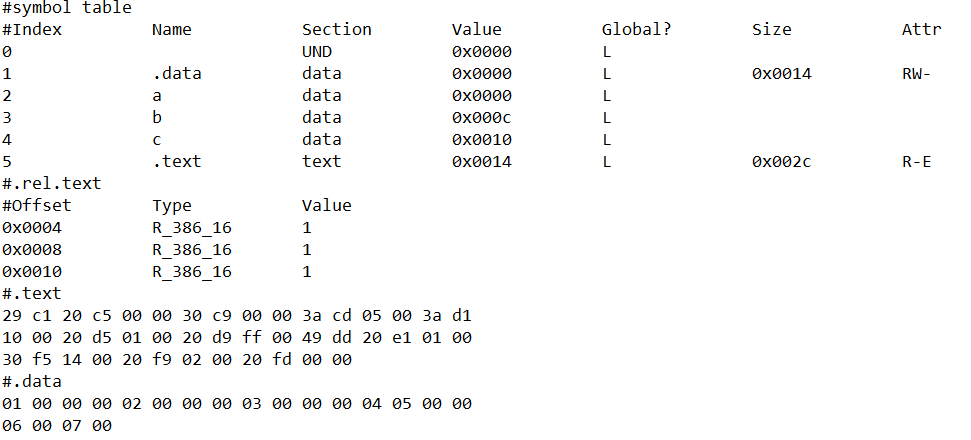
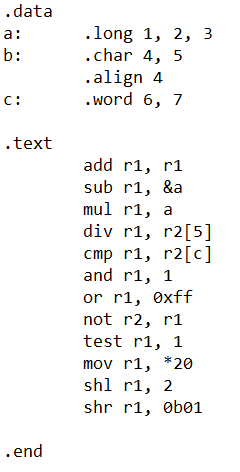
Zatim je neophodno prevesti program, za šta se koristi komanda:  
 **g++ -std=c++0x -g -o Assembler symbol.cpp section.cpp relocEntry.cpp parser.cpp main.cpp**.

Za samo pokretanje, pri čemu je kao argumente potrebno uneti putanju do ulaznog fajla, putanju do izlaznog fajla i početnu adresu od koje je potrebno smeštati sekcije, koristi se komanda:  
 **./Assembler inPath outPath startAddress**

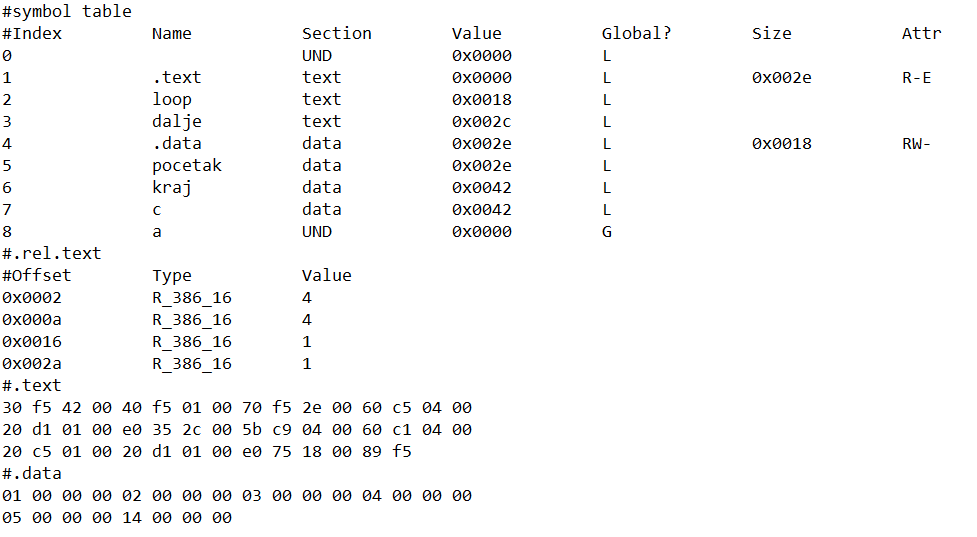
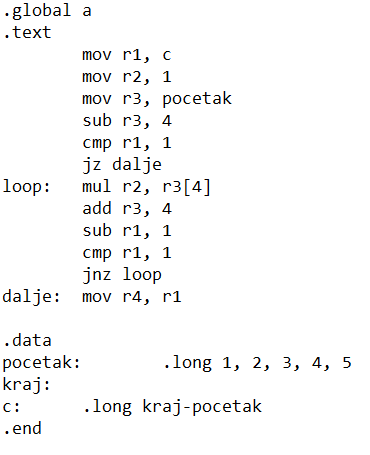
**4. Test programi:**

Postoje 4 test programa, sa očekivanim izlazima:

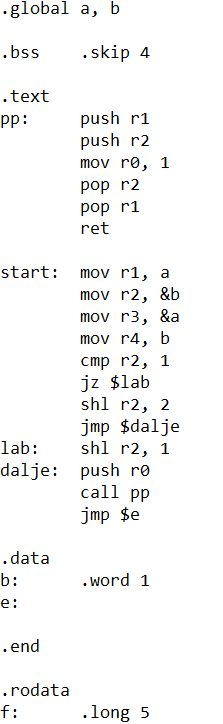
1. input1.txt (izlaz: output1.txt) – testira aritmetičko-logičke instrukcije i načine adresiranja u njima (&, registarsko indirektno sa pomerajem, neposredno, registarsko direktno, \*), obradu heksadecimalnih i binarnih vrednosti, kao i direktive .long, .word, .char i .align

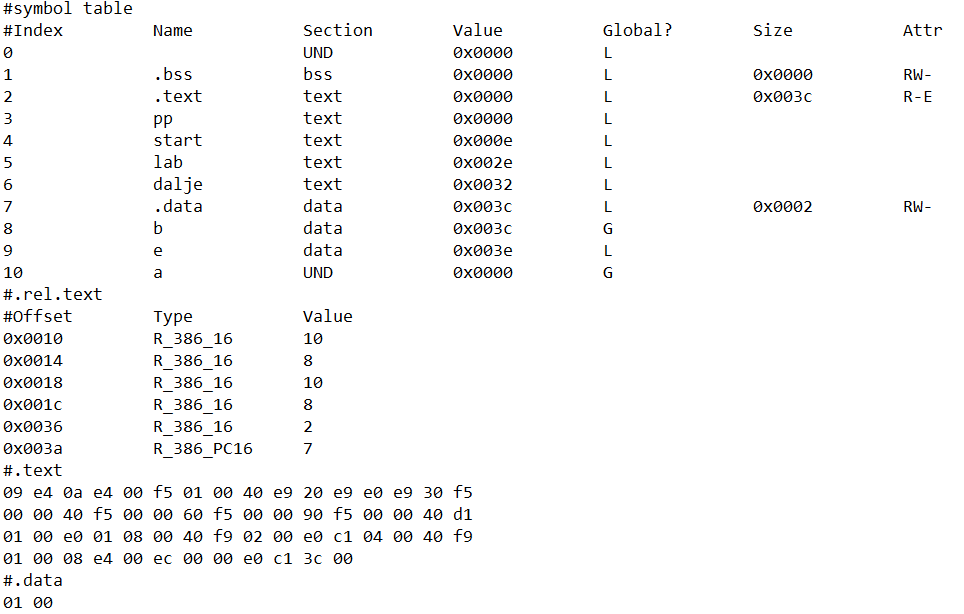


1. input2.txt (izlaz: output2.txt) – testira da li asembler dobro obrađuje direktivu .global za simbol koji se izvozi, računanje izraza prilikom definisanja vrednosti, kao i instrukcije skoka na labelu



1. input3.txt (izlaz: output3.txt) – testira da li asembler dobro obrađuje direktivu .skip, direktivu .global za izvoženje i uvoženje simbola, instrukcije push, pop, ret, call, kao i pc relativno adresiranje kod instrukcija skoka





1. input4.txt (izlaz: output4.txt) – testira da li asembler dobro kreira relokacione zapise kod instrukcija skoka sa pc relativnim adresiranjem, kada je labela na koju se skače u istoj sekciji, globalna i u nekoj drugoj sekciji (primer kao u zadatku 9 sa vežbi)

