# Simulacija GNU Assemblera

## Nacin prevodjenja programa

Prevodjenje programa vrsi se iz terminala tako sto se pozicioniramo u folder sa izvornim kodom i komandom ***make*** pokrenemo recepte koji su zapisani unutar Makefile-a koji je deo projekta.

## Opis resenja

Resenje prvog zadatka obuhvata leksicka analiza zadatog ulaznog fajla, tokenizacija odgovarajucih leksickih grupacija, prevod odgovarajucih tokena i upis u izlazni fajl. Za leksicku analizu koriscen je Flex lekser. Tokeni na koje lekser razvrstava leksicke grupacije su sledeci:

END;

GLOBAL;

EXTERN;

SECTION;

BREAK\_LNE;

SEPARATOR;

DIRECTIVE;

LABEL;

INSTRUCTION;

LITERAL;

REG;

SYMBOL;

SPEC\_CHAR;

SIGN;

Nakon leksicke analize, prolazi se kroz tokene instrukciju po instrukciju sve do END tokena. Pri nailasku na razlicite tokene, definisano je razlicito ponasanje programa.

Kljucne komponente programa su tabela simbola VarTable, tabela sa relokacionim zapisima RelTable, tabela obracanja unapred FLink, i prevedeni binarni zapis Binary.

Tabela simbola sluzi za upisivanje svih simbola koji se koriste u programu upisanom u input file-u, kako simbola definisanih labelama i direktivom .equ, tako i naziva sekcija. Ulazi u tabeli simbola sadrze sledeca polja:

string name – ime simbola

valid – da li je symbol razresen, validan. Koristi se prilikom definisanja simbola koji zavise od drugih simbola, kao na primer pri .equ direktivi, za zastitu od redefinisanja vec definisanog simbola, kao i za proveru validnosti ulaznog fajla, da li su svi simboli koji su trebali da budu definisani definisani. Ovo se odnosi samo na labele i .equ directive, simboli sekcija ne koriste ovo polje.

section – broj sekcije u kojoj je symbol definisan, za sekcije taj broj je jednak id-ju same sekcije

value – vrednost simbola

visibility – oznacava da li je symbol lokalan ili globalan, svi simboli koji oznacavaju sekcije su lokalni. Eksterni symbol je automatski globalan i validan. Definisanje eksternog simbola je greska.

id – oznacava id simbola, automatski se generise i dodeljuje. U toku prevodjenja moze se desiti da se id promeni(pri definisanju nove sekcije nakon definisanja nekih simbola prethodne sekcije, da bi se odrzao raspored simbola slican primeru sa casa), sto potencijalno vuce i druge promene u drugim komponentama sistema.

externVar – flag koji oznacava da je symbol eksterni. Takve simbole nije dozvoljeno definisati, oni su automatski validni.

fLink – tabela obracanja unapred. Svaki symbol ima praznu tabelu fLink, ali je popunjavaju samo oni simboli koji se koriste pre nego sto su postali validni. Ako se symbol a koristi ispred njegovog definisanja, on ce se upisati u tabelu simbola, ali nece biti validan, u fLink ce se upisati gde se on koristi, gde ucestvuje. Kad symbol postane validan, onda se krece u razresavanje svih fLink zapisa.

links – koristi se pri proveri da li je symbol definisan .equ direktivom validan, posto on moze biti zavisan od vise simbola. Tek kada se svi linkovi od kojih zavisi ovaj simbol povezu, on postaje validan, i tek onda moze da ucestvuje u daljem razresavanju simbola. Svi linkovi su povezani kada je links == linked. Povecavaju ga zavisnosti od drugih simbola, koji nisu poznati u trenutku prevodjenja .equ direktive

linked - koristi se pri proveri da li je symbol definisan .equ direktivom validan, posto on moze biti zavisan od vise simbola. Tek kada se svi linkovi od kojih zavisi ovaj simbol povezu, on postaje validan, i tek onda moze da ucestvuje u daljem razresavanju simbola. Svi linkovi su povezani kada je links == linked. Povecavaju ga simboli koji se ugrade u tu .equ direktivu kada sami postanu validni.

size – koristi se za prepoznavanje da li je u pitanju symbol koji oznacava sekciju(size = 0), ili neki drugi symbol(size = -1). Prvobitno je plan bio da size za sekcije sadrzi velicinu sekcija, ali se od toga odustalo zbog bespotrebnosti zbog toga sto velicina moze lako da se zakljuci bez ove promenljive na osnovu velicine binary tabele. Ostala je kao legacy samo za razliku sekcija od ostalih simbola.

binary – tabela koja sadrzi binary prevod. Ironicno, ispisuje se u hex obliku. Koriste je samo sekcije.

Relocation Table – koristi se kao pomoc linkeru, radi ugradnje odredjenih vrednosti simbola nakon procesa linkovanja. Govori gde i kako linker treba da ugradi vrednosti koje proizvede. Koriste je samo sekcije.

idGenerator – generator id-jeva.

Tabela sa relokacionim zapisima se koristi kao pomoc linkeru, radi ugradnje odredjenih vrednosti simbola nakon procesa linkovanja. Govori gde i kako linker treba da ugradi vrednosti koje proizvede. Ulazi u tabeli relokacionih zapisa sadrze sledeca polja:

offset – prikazuje na kojoj “adresi” odnosno offsetu od pocetka binary-ja treba linker da upise odredjenu vrednost.

addrType – moze biti apsolutna i pc relativna. PC relativna se koristi kod instrukcija koje su PC relativne.

varTableEntry – koristi se da prikaze koji je simbol cija se vrednost nakon linkovanja upisuje. Ovde je upisan broj ulaza u varTable(tabelu simbola), koji je jednak id-ju simbola.

sectionOrId – odredjuje da li se koristi id simbola ili se koristi sekcija u kojoj je taj simbol definisan(true – section, false – id). Kod lokanlnih simbola, posto se trenutna vrednost simbola upisuje u binary u toku asembliranja, u relokacionu tabelu se u promenljivu varTableEntry upisuje ulaz u tabeli simbola gde se taj simbol nalazi, i true u sectionOrId, jer se upisuje vrednost na koju linker pomeri sekciju u kojoj je simbol definisan, a ne nova vrednost citavog simbola. Za globalne promenljive upisuje se upravo citava nova vrednost simbola nakon linkovanja I u tom slucaju sectionOrId = false.

variable – odredjuje da li se zapis u relokacionoj tabeli odnosi na simbol u tabeli simbola(varTable) ili na binary. Ako je variable – true, to bi znacilo da se odnosi na simbol, i da je offset u stvari id simbola. Koristi se kod simbola koji su definisani pomocu .equ diretkive.

Binary table je niz bajtova koji nastaje asembliranjem programa zadatog sa ulaza.

Tabela obracanja unapred(fLink) sluzi za slucajeve kada se negde koristi simbol koji jos uvek nije poznat. Svaki simbol sadrzi fLink, i to mu govori da li se negde pominjao ranije. Ako se pominjao, u trenutku spoznavanja simbola, kad postane validan, krece se u razresavanje na mestima gde se on pominjao. Po nekoj analogiji bih rekao da je to “relokaciona tabela koju koristi sam assembler”. Ulazi u tabeli fLink sadrze sledeca polja:

addr – adresa odnosno offset u binary tabeli gde se upisuje simbol koji koriscen negde ranije, kad postane validan.

sign – znak sa kojim se simbol pominjao, dal se dodaje ili se oduzima.

size – da li zauzima jedan ili dva B, 0 – 1B, 1 – 2B

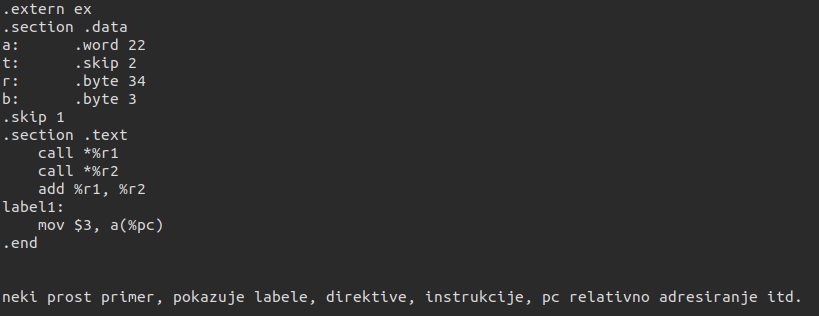
variable – ukoliko se zapis u fLinku odnosi na .equ direktivu, odnosno ako je ovaj simbol ucestvovao u izgradnji nekog drugog simbola uz pomoc .equ directive i to pre nego sto je bio poznat(validan). U ovom slucaju, addr se odnosi na ulaz u tabelu simbola(id simbola).

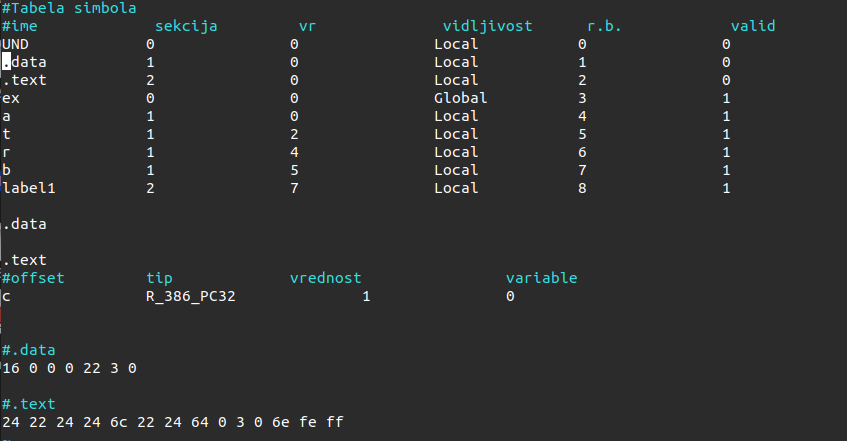
pcRel – koristi se da bi znali dal je u pitanju pcRel adresiranje, jer se u tom slucaju vrednost promenljive dodaje na mesto gde se upisuje. (dodaje se vrednost u slucaju da je simbol lokalan na pomeraj koji je vec upisan u kod instrukcije za taj operand pri pc relativnom adresiranju).

section – koristi se da bi se pri razresavanju fLinka za neki simbol znalo u binary koje sekcije se to razresenje upisuje. Ako se u jednoj sekciji simbol pominje, a on je lokalni simbol druge sekcije, kad on postane poznat(validan), mora se znati u kojoj sekciji se ispravka vrsi. Ovo se odnosi na ove fLink-ove kojima je variable = false.

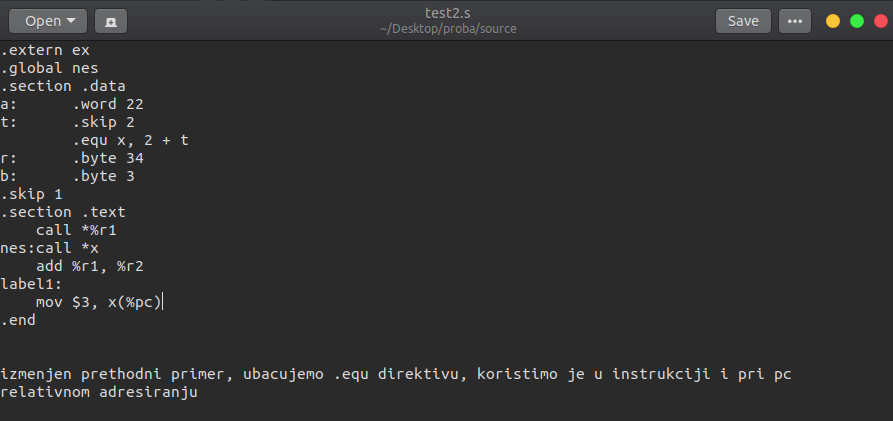
## Test primeri i izlazi

test1.s

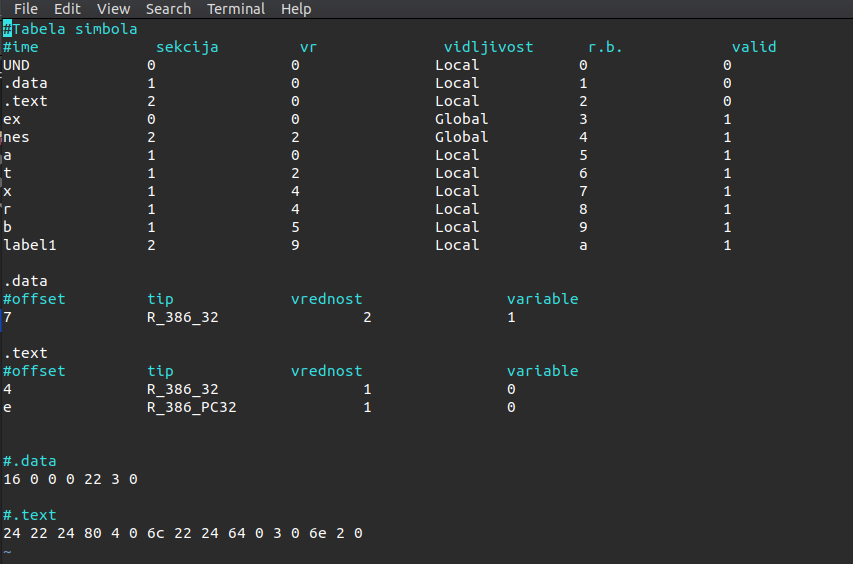


Izlaz test1.o

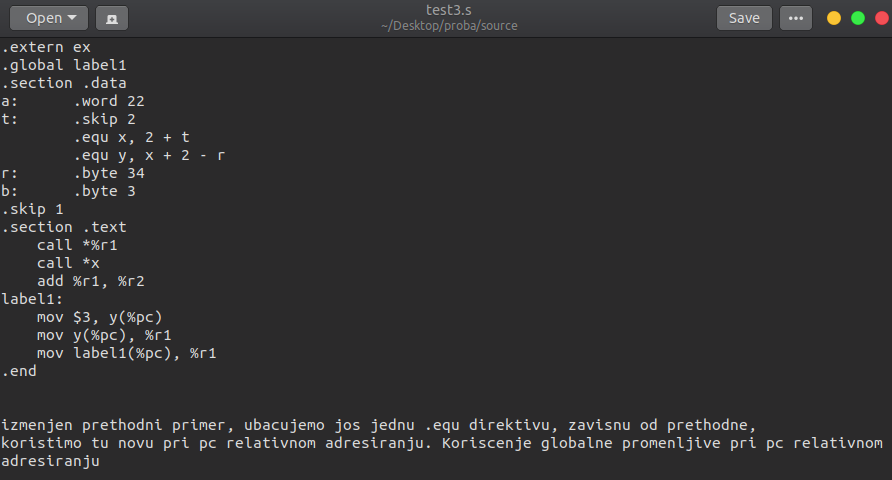
test2.s



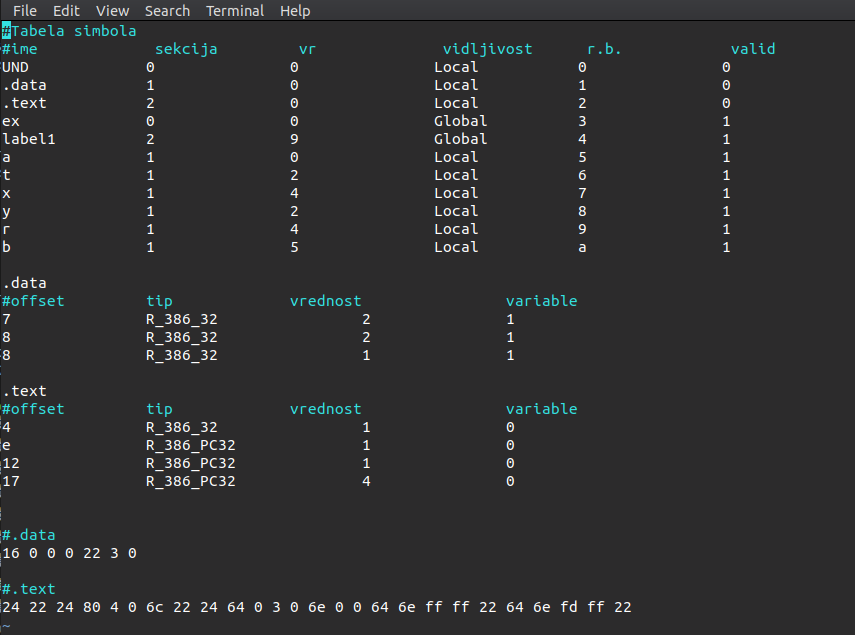
Izlaz test2.o



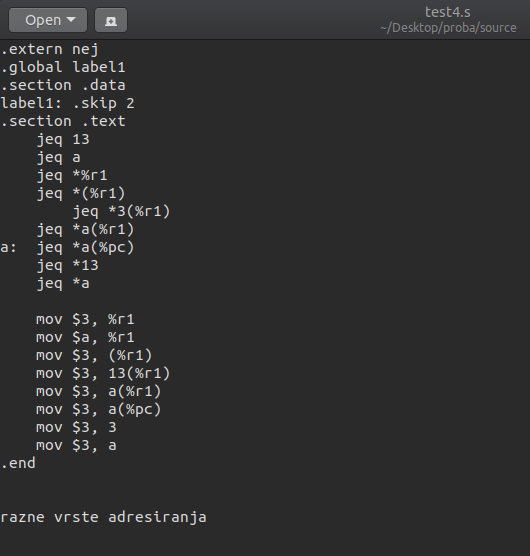
test3.s



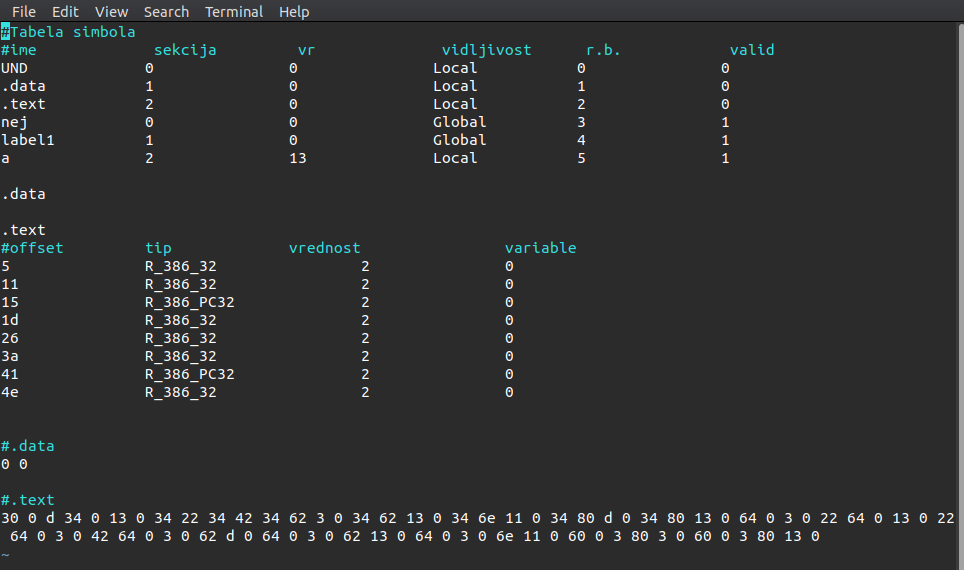
Izlaz test3.0



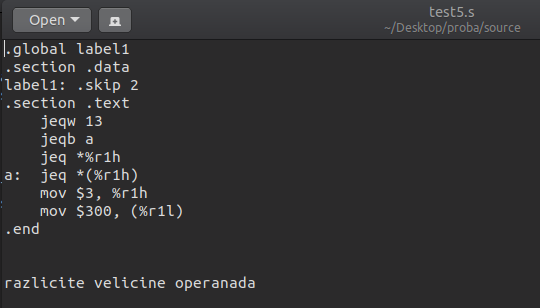
test4.s



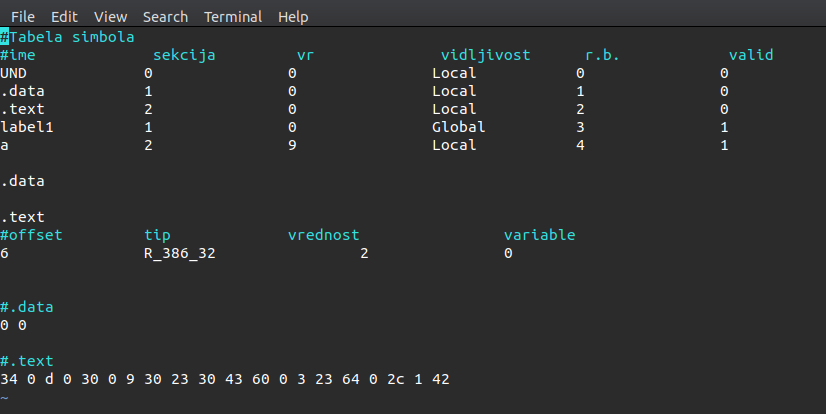
Izlaz test4.o



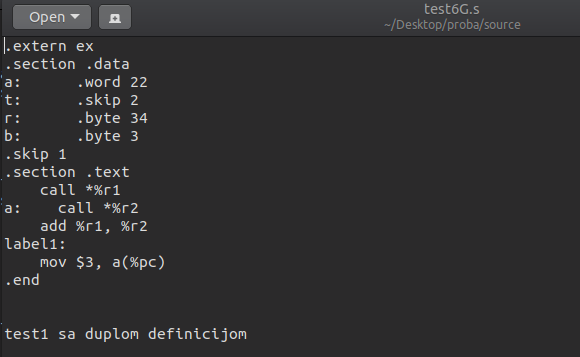
test5.s



Izlaz test5.o



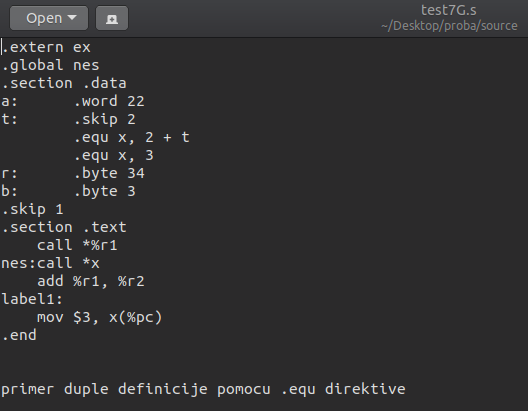
test6G.o



Izlaz



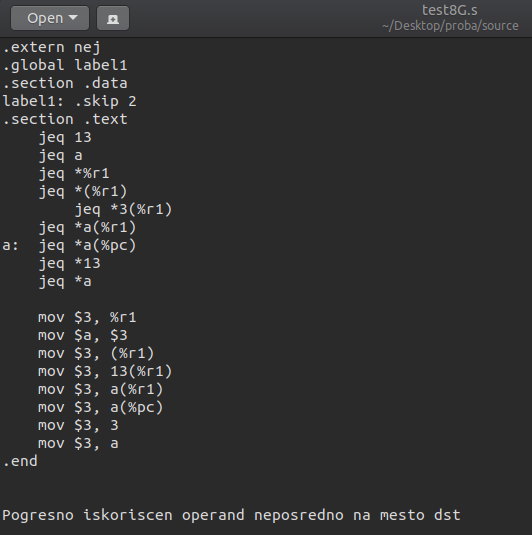
test7G.o



Izlaz



test8G.o



Izlaz

