Domaći zadatak iz Sistemskog softvera

Asembler

Zorana Janković 0143/16 SI

1.Zadatak

Napisati dvoprolazni asembler za procesor opisan u prilogu. Svi potrebni argumenti

asemblera zadaju se kroz komandnu liniju prilikom pokretanja. Ulaz asemblera je tekstualni fajl u

skladu sa sintaksom opisanom u nastavku. Za potrebe leksičke analize ulaznog tekstualnog fajla

dozvoljeno je koristiti generatore leksera. Izlaz asemblera treba da bude predmetni program zapisan

u tekstualnom fajlu. Za potrebe učitavanja u emulator dozvoljeno je generisati i binarni fajl pored

tekstualnog.

Format predmetnog programa bazirati na školskoj varijanti ELF formata (tekstualni fajl kakav

je korišćen u zadatku 9 u prezentaciji V3\_Konstrukcija asemblera.ppt) i predložiti izmene u formatu u

skladu sa potrebama ciljne arhitekture (nove sekcije, novi tipovi zapisa o relokacijama, dodatna polja

u postojećim tipovima zapisa, novi podaci o predmetnom programu i slično).

Prilikom generisanja izlaza asemblera voditi se principima koje koristi GNU asembler. Sve

sekcije smeštaju se počev od nulte adrese. Sintaksa asemblera i ostali zahtevi:

- jedna linija izvornog koda sadrži najviše jednu asemblersku naredbu/direktivu,

- labela, koja se završava dvotačkom, se može naći na početku linije izvornog koda nakon

proizvoljnog broja belih znakova,

- labela može da stoji samostalno, bez prateće asemblerske naredbe/direktive u istoj liniji izvornog

koda, što je ekvivalentno tome da stoji u liniji izvornog koda sa prvim sledećim sadržajem,

- simboli se izvoze i uvoze navođenjem asemblerskih direktiva .global <ime\_simbola> i

.extern <ime\_simbola>, respektivno, pri čemu u okviru jedne direktive može da se navede i više

simbola razdvojenih zapetama,

- direktiva ekvivalencije .equ <ime\_simbola>, <vrednost> uvodi simbol sa datom vrednošću,

- celokupan izvorni kod je podeljen na sekcije definisane pomoću sledećih asemblerskih direktiva:

• .text – sekcija sa mašinskim kodom,

• .data – sekcija za podatke sa inicijalnim vrednostima,

• .bss – sekcija za podatke bez inicijalnih vrednosti,

• .section <ime\_sekcije> [, "<flegovi>"] – sekcija sa navedenim proizvoljnim

imenom (datu asemblersku direktivu je moguće navesti više puta sa različitim imenom),

- fajl sa izvornim kodom se završava (ostatak fajla se odbacuje tj. ne prevodi se) pomoću asemblerske

direktive .end,

- asemblerske direktive .byte, .word, .align i .skip imaju identične funkcionalnosti kao u okviru

GNU asemblera,

- ostatak sintakse, ukoliko nije definisan u prilogu, definisati po sopstvenom nahođenju.

Primer komande, kojom se pokreće asembliranje izvornog koda u okviru fajla ulaz1.s, dat je

u nastavku:

asembler -o ulaz1.o ulaz1.s

Rešenje:

Napisan je dvoprolazni asembler u jeziku c. U prvom prolazu se generise tabela simbola i proverava se sintaksna ispravnost. Ukoliko se posle prvog prolaza ustanovi da je doslo do greške obrada se prekida.

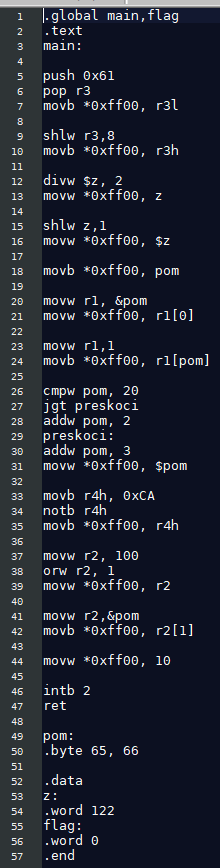
Posle prvog prolaza proverava se ispravnost tabele simbola.

U drugom prolazu se generise binarni sadrzaj za svaku sekciju kao i relokacioni zapisi, i takodje proverava ispravnost ulaznog programa. Ukoliko se posle drugog prolaza ustanovi da je doslo do greške obrada se prekida.

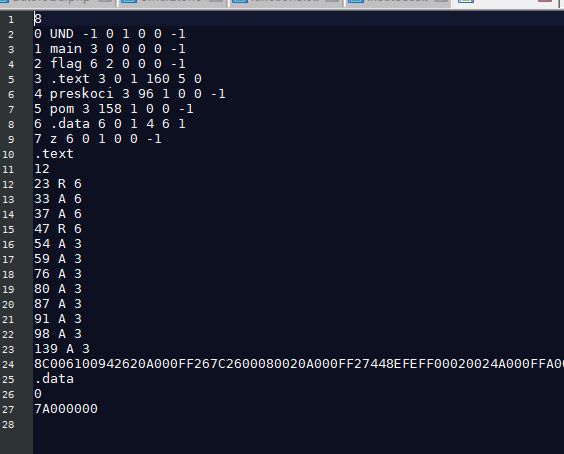
Posle drugog, ukoliko nije doslo do greske podaci se ispisuju u fajl u određenom formatu.

Komanda za prevođenje:

gcc asembler.c –o asembler

Primer ulaznog fajla: 

Primer izlaza za gore navedeni ulazni fajl:



Ceo sadržaj text sekcije nije mogao da stane u sliku.

Format izlaznog fajla:

U prvoj liniji se nalazi broj simbola u tabeli simbola i u toliko sledećih redova su zapisi za simbole.

Dalje se za svaku sekciju nalazi:

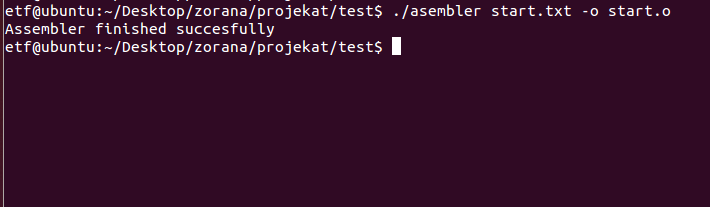
Ime sekcije

Broj relokacionih zapisa K

U K sledećih redova se nalaze relokacioni zapisi za trenutnu sekciju

Binarni sadržaj trenutne sekcije

Primer pokretanja:



Zadaju se ulazni i izlazni fajlovi. Fajl posle –o je izlazni.