Dokumentacija

**Opis projekta**

Projektni zadatak predstavlja jednoprolazni assembler za 16-bitni dvoadresni procesor sa Von-Neuman arhitekturom. Procesor poseduje osam opštenamenskih 16-bitnih registara označenih sa r gde može imati vrednosti od nula do sedam. Moguće je zasebno koristiti viših ili nižih osam bita svakog od opštenamenskih registara kao 8-bitni registar označen sa rh ili rl, respektivno. Registar r7 se koristi kao pc registar. Registar r6 se koristi kao sp registar. Ulaz asemblera je tekstualni fajl.

Jednoprolazni asembler učitava liniju po liniju ulaznog fajla generišući tokene. Tokeni se zatim obrađuju i na osnovu tipa tokena se vrši smeštanje podataka u tabelu simbola, relokacionu tabelu ili tabelu podataka. Korišćena i pomoćna tabela obraćanja unapred u kojoj su se čuvali podaci za simbole koji nisu bili definisani u trenutko obrade. Izlazni fajl sadrži tabelu simbola u kojoj je prikazan naziv simbola, njegova vrednost, da li je globalan ili lokalan i tip. Relokaciona tabela sadrži tip relokacije-relativni ili apsolutni, pomeraj i redni broj iz tabele simbola na koji se relokacija odnosi. Tabela podataka sadrži podatke koji će se smestiti u memoriju.

**Komande**

Za pokretanje programa koriste se sledeće naredbe:

*g++ -o program Asembler.cpp Asembler.h main.cpp Converter.cpp Converter.h Lexer.cpp Lexer.h Obracanja.cpp Obracanja.h Podaci.cpp Podaci.h Provere.cpp Provere.h Relokaciona.cpp Relokaciona.h TabelaSimbola.cpp TabelaSimbola.h*

*./program izlaz.txt ulaz.s*

**Sintaksa-dodatne napomene**

* Usled neadekvatnog unosa instrukcija ili operanada u komandnoj liniji će se ispisati greška
* Usled ispravnog unosa u izlaznom fajlu će se pojaviti tabela simbola, relokaciona tabela i tabela podataka
* Program može podržati rad sa dekadnim, heksadecimalnim i decimalnim brojevima
* Prefiks za rad sa heksadecimalnim brojevima je *0x,* a za decimalne *0b*
* Za definisanje nove sekcije se koristi sintaksa *.section .sekcija:*
* Dozvoljene sekcije su .data: .text: i .bss:
* Instrukcije treba pisati velikim slovima, a registri se označavaju malim slovima
* Za imenovanje labela nije dozvoljeno korišćenje specijalnih znakova
* Jedna linija izvornog koda sadrži najviše jednu asemblersku naredbu/direktivu
* Tipovi relokacija su *R\_386\_PC16* -relativnoI *R\_386\_16*-apsolutno
* Instrukcije se ne mogu nalaziti van *.text:* sekcije
* Dozvoljeni načini adresiranja
  + Neposredno
  + Registarsko direktno
  + Registarsko indirektno bez pomeraja
  + Registarsko direktno bez pomeraja
  + Memorijsko
  + Apsolutno
* U programu nije moguće navoditi komentare
* Ulazni fajl se mora završiti sa .end direktivom
* Ostatak sintakse je u skladu za postavkom zadatka

**Primeri korišćenja**

* Skokovi
  + JMP \*simbol1
  + JMP %r1
  + JMP (%r2)
  + JMP simbol2
  + JMP simbol3(%r3)
  + JMP simbol3(%pc)
* Jednoadresne instrukcije
  + PUSH %r1
  + PUSH simbol1
  + PUSH (%r1)
  + PUSH %psw
  + PUSH $simbol2
  + PUSH simbol3(%r3)
* Dvoadresne instrukcije
  + ADD simbol1, %r1
  + ADD $simbol2, (%r2)
  + ADD simbol3, simbol4(%pc)
* Equ direktiva
  + .equ simbol1, 1+simbol2-0x4F
  + .equ simbol3, -1+0b0110100-simbol5
* Byte i word direktive
  + .byte simbol1,simbol2,25
  + .word simbol3,0x1A
* Skip
  + .skip 15
  + .skip 0xF
  + .skip 0b1111

**Testovi**

**TEST 1:**

A screen shot of a computer

Description automatically generated

Izlaz:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

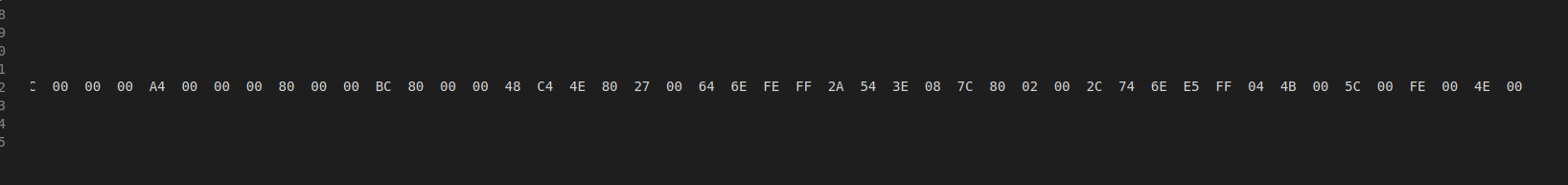
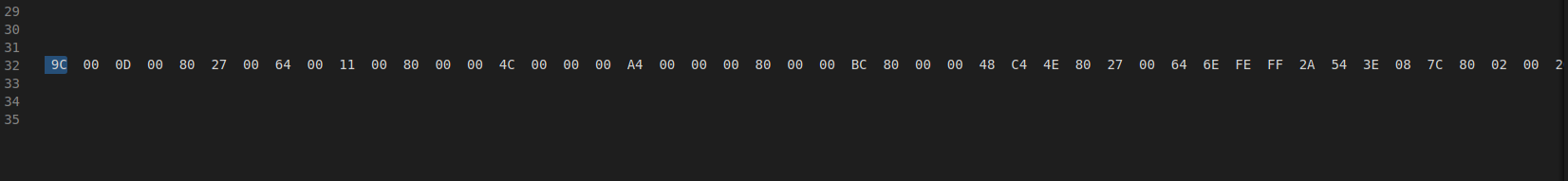
**TEST 2:**

**A black and red text

Description automatically generated**

Izlaz:**A screenshot of a computer

Description automatically generatedA close up of a screen

Description automatically generated**