Asembler – dokumentacija

# Kratak pregled rada asemblera

Osnovna klasa je klasa Assembler. Pored nje, postoje još i klase Symbol\_table - tabela simbola, Section\_table - tabela sekcija, Relocation\_table - tabela relokacija, Constants\_table -tabela konstanti (simbola definisanih direktivom .equ) i Undefined\_symbol\_table – tabela nedefinisanih simbola.

U klasi Assembler nalaze se pokazivači na objekte tipa Relocation\_table, Constants\_table, i Undefined\_symbol\_table – postoji po jedan objekat tipa svake od ovih klasa, a u klasi Symbol\_table pokazivač na objekat tipa Section\_table.

Klasa Assembler sadrži skoro sve funkcije potrebne za obradu ulaznog fajla. Ostale klase sadrže definicije struktura koje predstavljaju ulaze u tabele tih klasa.

U main.cpp se dohvati ime ulaznog fajla, koje je prosleđeno kao jedini argument komandne linije, napravi objekat tipa Assembler i pokrene funkcija Assembler::assemble(). U toj funkciji se dohvata linija po linija ulaznog fajla i obrađuje. Prvo se poziva funkcija first\_pass(string line) klase Assembler i prosleđuje joj se pročitana linija. Ova funkcija vraća vrednost različitu od nule u slučaju greške, i na ekranu se ispisuje na kojoj liniji je greška. Posle prvog prolaza se prolazi kroz tabelu nedefinisanih simbola. Ako ova funkcija vrati vrednost različitu od nule, na ekranu se ispisuje informacija da je uočena greška u definiciji konstante. Ako greške nema, opet se dohvata linija po linija iz ulaznog fajla. Sada se linija obrađuje u funkciji second\_pass(string line). Kao i u prvom prolazu, ispisaće se kod linija sa greškom ako takva postoji. U funkciji second\_pass se vrši upis(konkatenacija) u stringove koji predstavljaju sekcije. Iz tih stringova će se na kraju drugog prolaza vršiti ispis u izlazni fajl. U izlaznom fajlu će se naći i tabela simbola, tabela konstanti, tabela relokacija i tabela sekcija.

Napomena: U celom kodu ispravnost koda je proveravana pomoću regularnih izraza (regex). Funkcija u kojoj je na taj način utvrđena greška vratiće odgovarajuću vrednost i informacija o grešci biće ispisana na ekranu.

# Prvi prolaz

Ako se naiđe na labelu, ona se posebno obrađuje i ako zadovoljava neke uslove(nije prethodno definisan simbol sa istim imenom, ime se sastoji od slova, brojeva i donje crte, a mora da počne slovom), dodaje se u tabelu simbola. Ako se simbol već nalazi u tabeli simbola, proverava se da li je to globalni simbol definisan sa .global i, ako jeste, postavlja mu se sekcija (na tekuću sekciju) i vrednost. Ulaz u tabeli simbola sadrži polja: name – ime simbola, index – redni broj ulaza, section – trenutna sekcija, offset – pomeraj u odnosu na početak sekcije (location\_counter tekuće sekcije u trenutku dodavanja u tabelu simbola), locality – indikator da li je simbol globalan ili lokalan .

U nastavku reda sa labelom ili u drugom redu može se naći jedan od tri tipa naredbi: instrukcija, direktiva (.align, .skip, .global, .extern, .byte, .word, .equ, .end) i sekcija(.section, .text, .data, .bss)

Pri procesiranju sekcije (funkcija process\_section), ako se sekcija nije nalazila u tabeli simbola biće dodata. Postavlja se broj trenutne sekcije(polje curr\_section u tabeli simbola). To je indeks ulaza sekcije u tabeli sekcija (1-text, 2-data, 3-bss...). Ako se uvodi nova sekcija, biti za prava pristupa moraju biti postavljeni, u suprotnom se prijavljuje greška. Nova sekcija se dodaje i u tabelu sekcija. Ulaz u tabeli sekcija sadrži ime sekcije, prava pristupa, location\_counter i broj ulaza sekcije u tabeli simbola.

Instrukcije su podeljene u nekoliko grupa: instrukcije bez operanada (halt, ret, iret), instrukcije sa jednim operandom kod kojih je dozvoljeno neposredno adresiranje (push) i kod kojih nije dozvoljeno (int, not, pop, jmp, jne, jeq, jgt, call, dst) i instrukcije sa dva operanda, kod kojih u drugom operandu nije dozvoljeno neposredno adresiranje (xchg, mov, add, sub, mul, div, cmp, or, xor, test, shl, shr). Instrukcije su dozvoljene samo u sekcijama koje imaju postavljen bit x. Ako to nije slučaj prijavljuje se greška. Na početku obrade instrukcije se utvrđuje sufiks (ako postoji, b ili w) i dužina operanada na osnovu adresiranja. Zatim se location\_counter tekuće sekcije uvećava za određenu vrednost koja zavisi od broja i veličine operanada. Ako je utvrđena greška (pogrešan broj opearanada za neku instrukciju, nedozvoljeno adresiranje ili bilo koja druga) na ekranu će se ispisati red sa greškom.

Kod direktiva .byte, .word, .skip i .align se u prvom prolazu samo povećava location\_counter za odgovarajuću vrednost trenutne sekcije. Ove direktive nisu dozvoljene izvan sekcije (u nedefinisanoj sekciji). Direktive .global i .extern ubacuju navedene globalne simbole u tabelu simbola. Ove dve direktive su dozvoljene samo na početku ulaznog fajla, izvan sekcije. Posle .end direktive se prestaje sa daljom obradom (sve napisano ispod .end se neće gledati).

Direktiva .equ sliži za definisanje konstante. Sve vrednosti koje se ovako definišu biće poznate pre drugog prolaza. Linker neće primenjivati nikakve relokacije. Dozvoljene vrste izraza su: druga konstanta definisana sa .equ, hex ili dec brojka, zbir ili razlika predhodno navedenog i razlika dva lokalna simbola (dve labele) iz iste sekcije. Dozvoljeno je kao operand staviti i konstantu/simbol koji još uvek nije definisan (definisan je niže u kodu). U .equ direktivi se pravi novi ulaz u tabeli konstanti za simbol koji se definiše, vrednost se postavlja na nulu i naznačava se da konstanta nije definisana (ulaz sadrži polja za ime i vrednost konstante, i indikator definisanosti), a izraz prosledjuje funkciji process\_expression, kao i ulaz u tabelu konstanti. Ova funkcija utvrđuje da li se radi o izrazu sa jednim ili dva operanda, zatim obrađuje operande i, ako je moguće, dodeljuje konstanti u tabeli konstanti vrednost i postavlja njen indikator definisanosti. Ako to nije moguće, ubacuje izraz u tabelu nedefinisanih simbola. Ova tabela biće razrešena posle prvog prolaza. Naravno, u sličaju bilo kakve greške, informacija o grešci će se ispisati na ekranu.

# Drugi prolaz

U drugom prolazu labela se preskače, a obrađuju se sekcije, instrukcije i direktive. Pre drugog prolaza se resetuju brojači lokacija svih sekcija, a kursor ulaznog fajla postavlja na početak.

Kod obrade sekcije sada se samo postavlja broj trenutne sekcije.

U obradi instrukcije vrši se upis koda instrukcije, a zatim i njenih operanada u tekuću sekciju. Sve greške biće prijavljene kao i do sada. To se radi u funkciji writeOp. Ovoj funkciji se prosleđuje operand, prepoznaje se adresiranje, utvrđuje ispravnost. Posle ispisa se povećava location\_counter, a ako je potrebno pravi se i relokacioni zapis.

Postoje dva tipa relokacionih zapisa: apsolutna i relativna relokacija. Relativna se primenjuje kod PC-relativnog adresiranja (i registarskog indirektnog adresiranja gde je registar r7, odnosno PC). Relokacioni zapis sadrži informacije o tipu relokacije, sekciji, adresi relokacije i indeks ulaza u tabeli simbola za simbol čiju adresu linker treba da prepravi i upiše).

Kod direktiva .word i .byte u tekuću sekciju se upisuje zadata vrednost, ako je odgovarajuća (za .byte mora biti u opsegu od -128 do 127), a kod .skip i .align se upisuju nule. Location\_counter se, naravno, povećava za odgovarajuću vrednost. Kod direktive .global utvrđuje se da li su navedeni globalni simboli definisani u kodu ispod, odnosno proverava se da li su stavljeni u neku sekciju. Ako nisu prijavljuje se greška. Za .extern i .equ se u drugom prolazu ne radi ništa. I ovde .end direktiva označava kraj, sve iza nje se zanemaruje.

# Prolaz kroz tabelu nedefinisanih simbola

Posle prvog prolaza kroz kod, vrši se prolaz kroz tabelu nedefinisanih simbola (Undefined\_symbol\_table), kako bi se razrešili simboli koji su referisani pre njihove definicije u direktivi .equ. Kroz ovu tabelu prolazi se redom od prvog do poslednjeg ulaza. Ukoliko neki izraz može da se razreši, upisuje se rezultat u tabelu konstanti i ulaz briše iz tabele simbola. Prolaz se ponavlja sve dok se tabela ne isprazni. Ako posle prolaza nema ni jedne promene u tabeli, prestaje se sa obradom i prijavljuje greška u definiciji konstante. To je znak da je programer načinio grešku (na primer, definicija u krug: .equ a, b; .equ b, a).