Опис асемблера

Студент: Душан Дамљановић ментор: Саша Стојановић

Садржај

[Увод 3](#_Toc461973422)

[Упутство за превођење и покретање програма 3](#_Toc461973423)

[Опис решења 4](#_Toc461973425)

[Опис релокација 5](#_Toc461973427)

# Увод

Овај асемблер је настао као као одговор на задатак из предмета Системски Софтвер са Електротехниког факултета Универѕитета у Београду.

Од прилога, поред овог документа се могу наћи поставка задатка у документу ss\_19\_jun\_proj\_postavka.pdf као и опис процесора за који је асемблер прављен у документу ss\_19\_jun\_proj\_prilog.pdf.

У директоријуму src се налази изворни код решења, док се извршна верзија програма (.exe) налази у директоријуму bin. У истом директоријуму се налазе и тест фајлови за тестирање програма.

Поставка задатка је она из испитног рока за јун 2019. године.

Програм је написан на језику C++ за оперативни систем Linux.

Превођење и покретање под овим оперативним системом је описано у даљем тексту.

# Упутство за превођење и покретање програма

За превођење програма се користи g++ преводилац.

У случају да немате наведени компајлер на Linux систему под којим радите, могуће је извршити инсталазију из командне линије укуцавањем следећих команди:

sudo add-apt-repository ppa:ubuntu-toolchain-r/test

sudo apt-get update

sudo apt-get install g++

Програм се преводи и покреће из командне линије.

Потребно је лоцирати се у директоријуму у коме се налази иворни код решења. За превођење је потом потребно укуцати следећу команду:

g++ -o asemler -std=c++0x Instrukcija.cpp relokacija.cpp Sekcija.cpp Simbol.cpp Parser.cpp drugiProlaz.cpp Source.cpp -g

Преведени програм покреће се следећом командом, уз навођење фајла који представља параметар функције main:

./asemler input.txt

# Опис решења

Асемблер који је дат у овом решењу је двопролазни и имплементиран је на начин који је дат на предавањима из предмета Системски Софтвер.

У првом пролазу, асемблер чита ред по ред из улазног фала и исти ред обрађује на следећи начин. Проверава да ли постоје глобални симболи и убаци их у табелу симбола. За сваки симбол чувају се следеће информације: назив, редни број, информација да ли је локални или глобални, информација да ли је симбол апсолутан или релативан, вредност и секција којој припада. За све увезене глобалне симболе назив секције је „УНД“. Након тога, асемблер тражи лабелу, и уколико постоји, додаје је у табелу као локални симбол. У случају наиласка на нову секцију, асемблер је убацује у табелу симбола и ажурира вредности бројача локације и тренутне секције. У случају директиве или помераја асемблер рачуна и ажурира вредност помераја. У првом пролазу обрађена је већина грешака у улазу до којих може доћи, као што су неправилни формати иструкција и директива, неправилне употребе апсолутних и релативних симбола, неправилност израза итд.

Такође, усваја се слично правило за изразе директива .skip, .align и .equ. Како у сва три случаја морамо у датом тренутку да срачунамо вредност израза, дозвољено је користити само цифре у датим изразима.

Као резултат првог пролаза, асемблер има попуњену табелу симбола.

Пре другог пролаза, асемблер се уверава да у табели не постоје симболи са недефинисаном секцијом који нису екстерни и да је претпоставка о апсолутним симболима тачна.

У другом пролазу, асемблер итерира кроз фајл тражећи инструкције и директиве, како би за исте генерисао машински код и евентуално релокације.

Све инструкције се декодирају у машинске речи дужине од 1 до 7 бајтова, у складу са документом ss\_19\_jun\_proj\_prilog.pdf. Асемблер ће их претворити у машинске речи. Свака секција има свој машински код у формату „little-endian”. У случају коришћења симбола у инструкцији који није дефинисан у тренутној секцији, генерисаће се релокација. Секције, поред машинског кода, садрже и табелу релокација, у којој се налазе све њене релокације.

Претпостављају се одређена правила адресирања. У операнду одредишта инструкције не сме се користити непосредно адресирање. Претпоставља се, такође, да се инструкције које захтевају операнде морају позивати уз информацију да ли се ради о бајту или речи, и да сам операнд мора бити компатибилан са том информацијом.

Директиве су дефинисане у складу са поставком задатка. Све директиве генеришу машински код, а све директиве осим .skip и .align (које мењају померај у односу на вредност аргумента, па стога тај аргумент не може бити симбол) могу генерисати и релокације, уколико се у аргументу јавља симбол из друге секције. Ти аргументи могу бити у форми израза, где је ниво имплементације такав да израз може садржати највише два симбола, и то само у случају да се измежу њих налази минус. Такође, важи правило да, уколико се у изразу налази симбол, он мора бити први операнд у изразу. Ово се не односи на апсолутне симболе који се могу наћи на било ком месту у изразу.

Резултати другог пролаза су табела релокација и машински код.

Након другог пролаза, ради се упис табеле симбола, табеле релокација и машинског кода у фајл output.txt по стандарду задатка 9 из V3\_Konstrukcija\_asemblera.ppt.

Уколико било где у асемблеру дође до грешке, текст ће се исписати на конзоли и извршавање ће бити прекинуто.

## 

## Опис релокација

У систему постоје две врсте релокација.

Релативне релокације се извршавају ако за инструкције које захтевају промену програмског бројача (дакле, када се користи адресирање помоћу операнда ‘$’ или регистарско индирекно адресирање са померајем када користимо регистар r7 или pc).

Апсолутне релокације се извршавају у преосталим случајевима, односно када је померај константан.