|  |  |
| --- | --- |
| etf_logo_2010_cir | **УНИВЕРЗИТЕТ У БЕОГРАДУ**  **ЕЛЕКТРОТЕХНИЧКИ ФАКУЛТЕТ**  Управљање софтверским пројектима – 2017/2018.  Булевар краља Александра 73, ПФ 35-54, 11120 Београд, Србија  телефон: 011/3218-321, [dekanat@etf.bg.ac.rs](mailto:dekanat@etf.bg.ac.rs) |

ПРОЈЕКТНИ ЗАДАТАК - АСЕМБЛЕР

Системски софтвер

Студент:

Лукић Тања, 2016/0626

Београд, август 2019.

Садржај

[О пројектном задатку 3](#_Toc17286759)

[Инсталација неопходних алата за коришћење и евентуалну модификацију пројекта 3](#_Toc17286760)

[Упутство за покретање и превођење 4](#_Toc17286761)

[Опис решења 4](#_Toc17286762)

[Ограничења у синтакси 5](#_Toc17286763)

# О пројектном задатку

Израда решења пројектног задатка урађена је под оперативним системом Linux на х86 архитектури у програмском језику С++. Потребно је било осмислити синтаксу асемблерског језика (у складу са задатим ограничењима у тексту и прилогу самог пројекта), а након тога и написати двопролазни асемблер. Као улазни параметри самог програма наводе се два .txt фајла, један у ком се налази асемблерски код који је неопходно превести и други у који се смешта резултат превођења. Асемблер у првом пролазу попуњава табелу симбола, док у другом пролазу формира предметни програм и табелу релокативних записа.

# Инсталација неопходних алата за коришћење и евентуалну модификацију пројекта

Пројекат је писан у NetBeans окружењу уз коришћење g++ компајлера (верзија 4.9).

Да би се инсталирала одговарајућа верзија компајлера неопходно је у терминалу редом уносити следеће команде:

* **sudo add-apt-repository ppa:ubuntu-toolchain-r/test**
* **sudo apt-get update**
* **sudo apt-get install gcc-4.9 g++-4.9**
* **sudo update-alternatives –install /usr/bin/gcc gcc /usr/bin/gcc-4.9 60 –slave /usr/bin/g++ g++ /usr/bin/g++-4.9**

За проверу инсталације и верзије компајлера у терминалу је довољно унети:

* gcc --version

Да би се инсталира одговарајућа верзија окружења неопходно је прво скинути инсталациони фајл са сајта (обратити пажњу да треба скинути фајл прилагођен Linux оперативним системима), а затим у терминалу редом унети следеће команде:

* **chmod +x netbeans-8.2-cpp-linux-x86.sh**
* **sh ./netbeans-8.2-cpp-linux-x86.sh**

# Упутство за покретање и превођење

Програм асемблер (аssembler – извршни фајл) је се покреће тако што се у терминалу поред извршног фајла наводи следећа линија:

* **asembler -o izlaz.txt ulaz.txt**

где izlaz.txt представља назив излазног фајла, а ulaz.txt назив улазног фајла у ком се налази асемблерски код који је потребно превести.

Ако би се преводио фајл ulazniTekst.txt, а резултат смештао у izlazniTekst.txt команда која би се унела у командни прозор гласила би ./assembler asembler -o izlazniTekst.txt ulazniTekst.txt .

# Опис решења

Улазни фајл се прво парсира тако што се сваки ред улазног фајла је представљен листом string-ова. Један string је заправо једна реч (све што се налази између бланко знакова и зареза) из улазног фајла, а листа string-ова је заправо један ред из улазног фајла. Цео улазни фајл се пампти као листа листи string-ова (листа string-ова је један ред улазног фајла, листа листи string-ова су сви редови улазног фајла), у даљем тексту садржај.

Након тога садржај пролази кроз функцију FirstRun() која симулира рад првог пролаза улазног асемблера. Функција пореди садржај са могућим комбинацијама које су раније предвиђене као могуће у синтакси асемблера. Функција FirstRun() позива помоћне функције које испитују веродостојност унетог садржаја и уколико га препознају као веродостојан и исправан обрађују га (повећава бројач локација, убацује симболе у табелу симбола, препознаје почетак нове секције...), а уколико примете неку грешку (не препознају инструкцију, препознају грешку у адресирању инструкција, препознају грешку у броју параметара инструкције...) избацују одговарајућу поруку и прекидају рад читавог програма.

Као табела симбола користи се посебна класа која представлју листу симбола (постоји посебна класа која дефинише сваки симбол). Класа симбола у себи има поља која говоре у којој секцији је дефинисан симбол (und ако није дефинисан), о вредности симбола (-1 ако није дефинисан), да ли је симбол глобалан (-1 ако је локалан, 1 ако је глобалан), да ли је симбол дефинисан унутар фајла (-1 ако није, 1 ако јесте).

У табелу симбола се убацују и секције, поље са именом симбола и секције у којој се дефинише симбол су исти, поља која одређују вредност симбола, глобалност и дефинисаност су све постављене на -1. Као додатна поља за опис секција јављају се величина секције и права приступа у секцијама. Та поља су у случају обичних симбола постављена на -1 и und респективно.

Уколико се у току обраде садржаја унутар функције FirstRun() препозна коришћење .equ директиве и дефинисање било чега што није константа, симбол који се дефинише и израз којим се симбол дефинише се убацују у табелу неизрачунљивих симбола. Табела неизрачунљивих симбола се разрешава на крају првог пролаза, прецизније између првог и другог. Уколико програм не успе да разреши све симболе из табеле пријављује грешку, а у супротном наставља даље са извршавањем и прелази на други пролаз.

Са преласком на обраду садржаја у другом пролазу, прелази се на коришћење функције SecoundRun(). Дата функције поново чита садржај и позива помоћне функције које освнову њега поново повећавају бројач локација, испитују одређене грешке које нису видљиве у првом пролазу (да ли је симбол извежен са .extern дефинисан у посматраном фајлу...) и или формирају одговарајући предметни програм за дату инструкцију (уз уколико је потребно формирање одговарајућег релокационог записа) или пријављују грешку и завршавају рад читавог програма.

За формирање одговарајућех предметног програма користи се посебна класа која саржи листу string-ова(у дањем тексту предметни програм) која заправо представља кодовање једне инструкције (један string је један бајт). За сваку секцију која треба да буде присутна у крајњем фајлу (предвиђено да све изузев .bss секције буду присутне) формира се посебна инстаца ове класе, свака инстанца поред предметног програма садржи и назив секције којој припада дати предметни програм. Релокациони записи се формирају креирањем инстаци такође засебне класе, а припадност секцијама се разрешава постојањем посебног поља унутар релокационог записа који говори о секцији унутар које је формиран тај релокациони запис.

На крају се пролази кроз табелу симбола (листу свих симбола), кроз листу свих инстанци класе за формирање предметног програма и записе унутар њих уписује у крајњи фајл, исто се ради и са релокационим записима.

Као резулат формира се излазни фајл који у себи има табелу симбола, предметни програм по секцијама и табелу релокативних записа.

# Ограничења у синтакси

* сваки фајл мора да се заврши са .end директивом нако које не сме да стоји ништа, ни празан ред
* уколико се користи непосредно адресирање неопходно је као суфикс на инструкције додати ‘b’ или ‘w’ и тиме одредити величину операнада
* уколико се користи регистарско директно адресирање или регистарско индиректно без помераја у оба операнда неопходно је као суфикс на инструкције додати ‘b’ како би се инструкција исправно кодовала
* уколико се користе инструкције скокова дозвољено је са њима корисити само меморијско директно адресирање или рс релативно
* дестинациони операнд не може бити одређен непосредним адресирањем
* приликом коришћења аритметичких операција дестинациони операнд мора бити одређен регистарским директним адресирањем
* ако се користи xchg инструкција оба операнда морају да користе регистарско директно адресирање

# Тестови

|  |  |
| --- | --- |
| ulaz.txt | izlazTest.txt |
| .global a,b,c  .extern d  .data  .skip 9  tanja:  .align 5  .text  addb r0,r1  addb r1, 0x25  addw r0, 0x0036  nikola:  movw sp[0], 0x1234  mov r6[8], \*1234  movb r6[8], r2  mov r3[368], r6[8]  mov r5[nikola], &tanja  add r4, nikola  mul r5, $a  .end | Ime | Sekcija| ofset|Global| Definisan| Redni broj | Velicina sekcije1 | Flagovi sekcija  .data .data und -1 -1 1 10 110  .text .text und -1 -1 2 52 101  a und und 1 -1 3 -1 und  b und und 1 -1 4 -1 und  c und und 1 -1 5 -1 und  d und und 1 -1 6 -1 und  tanja .data 09 -1 1 7 -1 und  nikola .text 0C -1 1 8 -1 und  Predmetni program po sekcijama  .data  00 00 00 00 00 00 00 00 00  00  .text  28 20 22  28 22 00 25  2C 20 00 36 00  24 4C 00 34 12  24 8C 08 00 A0 34 12  20 6C 08 24  24 86 70 01 8C 08 00  24 8A 0C 00 00 09 00  2C 28 A0 0C 00  3C 2A 8E FE FF  Relokacioni zapisi  25 R\_386\_32 2 .text  28 R\_386\_32 1 .text  2D R\_386\_32 2 .text  32 R\_386\_PC32 3 .text |

Напомена: тестирано је како програм кодује различите типове адресирања операнада унутар инструкција.

|  |  |
| --- | --- |
| relokacioniZapisiTest.txt | izlazRelokacijeTest.txt |
| .global a,b  .data  a: .skip 8  .align 13  d:  .word 1234  .text  movb r0, r1  b: movb r2,r3  add r4, $b  mul $a, r5  c: jmp b  jne $c  add r6, $d  cmp c,d  cmp b,a  .end | Tabela simbola  Ime | Sekcija| ofset|Global| Definisan| Redni broj | Velicina sekcije1 | Flagovi sekcija  .data .data und -1 -1 1 15 110  .text .text und -1 -1 2 43 101  a .data 00 1 1 3 -1 und  b .text 03 1 1 4 -1 und  d .data 0D -1 1 5 -1 und  c .text 10 -1 1 6 -1 und  Predmetni program po sekcijama  .data  00 00 00 00 00 00 00 00  00 00 00 00 00  D2 04  .text  20 20 22  20 24 26  2C 28 8E FE FF  3C 8E FD FF 2A  9C A0 00 00  AC 8E 0E 00  2C 2C 8E 0B 00  4C A0 10 00 A0 0D 00  4C A0 00 00 A0 00 00  Relokacioni zapisi  09 R\_386\_PC32 4 .text  0D R\_386\_PC32 3 .text  12 R\_386\_32 4 .text  1B R\_386\_PC32 1 .text  1F R\_386\_32 2 .text  22 R\_386\_32 1 .text  26 R\_386\_32 4 .text  29 R\_386\_32 3 .text |

Напомена: тестирано је како програм формира записе о релокацијама.

|  |  |
| --- | --- |
| tns.txt | izlazTNS.txt |
| .global a,b,c  .data  .byte  .equ d,a+6  a: .equ c, 5  .equ k, l  .skip 5  .equ l, 5+6  .align 7  .equ b, k+c  .end | Tabela simbola  Ime | Sekcija| ofset|Global| Definisan| Redni broj | Velicina sekcije1 | Flagovi sekcija  .data .data und -1 -1 1 7 110  a .data 01 1 1 2 -1 und  b .data 10 1 -1 3 -1 und  c .data 05 1 1 4 -1 und  d .data 07 -1 1 5 -1 und  l .data 0B -1 1 6 -1 und  k .data 0B -1 1 7 -1 und  Predmetni program po sekcijama  .data  00  00 00 00 00 00  00  Relokacioni zapisi |

Напомена: тестирано је како програм разрешава вредности симбола користећи табелу неизрачунљивих симбола.