Fakultet elektrotehnike i računarstva
Zavod za automatiku i računalno inženjerstvo

Arhitektura računala 1

Priprema za 1. laboratorijsku vježbu

Rezultat izrade pripreme je kratak izvještaj koji treba sadržavati:

- slike ekrana (*screenshot*) iz zadataka 4, 6 i 9
- odgovore na pitanja iz zadataka 7 i 9
- programski kod iz zadatka 10

Izvještaj treba izraditi u elektroničkom obliku i u **PDF** formatu predati na sustav **Moodle** najkasnije do roka naznačenog u obavijesti. Ime datoteke mora biti u formatu **JMBAG_ImePrezime.pdf** (npr. ako je moj matični broj 0036978123, predajem datoteku imena **0036978123_PeroPeric.PDF**). Datoteke u krivom formatu i pogrešno nazvane datoteke neće se uzimati u obzir.

1. Priprema

Instalirati distribuciju paketa ATLAS:

- upute su dostupne na stranici:

http://www.fer.unizg.hr/predmet/arh1/laboratorijske_vjezbe

2. Zadatak pripreme

Sljedeći program za FRISC ima nekoliko pogrešaka:

```
        ORG 0
         MOVE   50, R1
         MOVE   70, R2
         MOVE   5, R3

PETLJA   LD     R0, (R1)
         STORE  R0, (R2)
         ADD    R1, 4, R1
         ADD    R2, 4, R2
         SUB    R3, 1, R3
         JR_NZ  LOOP

         HALT

         ORG   50
         DW     1, 2, 3, 4, 5

         ORG   70
         DS    %D 20
```

Navedeni program kopira podatke iz jednog bloka memorije u drugi. Početni blok je na adresi 50_{16} i sadrži pet podataka (32-bitnih): 1, 2, 3, 4, 5. Odredišni blok je na adresi 70_{16}. Program u petlji kopira jedan po jedan podatak. Kao brojač petlje koristi se registar R3. Podatci se adresiraju registrima R1 (početni blok) i R2 (odredišni blok).

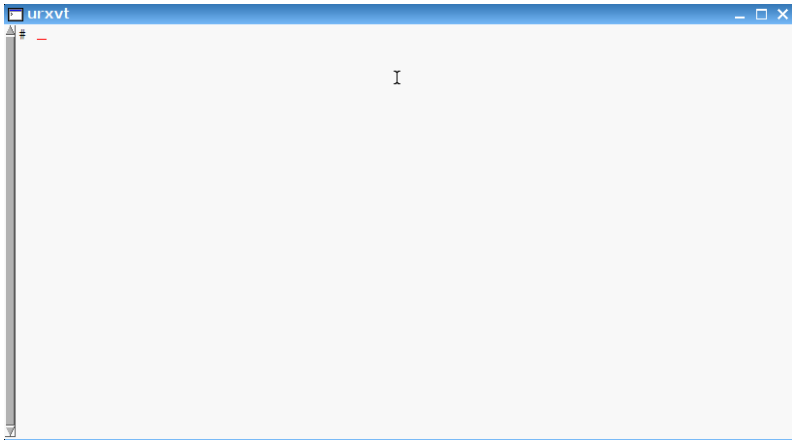
Pogreške treba pronaći i ispraviti. Program treba **asemblirati** (programom *xconas*), učitati u simulator (*xcompas*), **pokrenuti** i **provjeriti** da li daje očekivan **rezultat** (pregledom memorijskih lokacija 70_{16}-84_{16}).

Nakon toga treba modificirati izvorni program tako da veličina bloka ne bude poznata unaprijed/fiksno zadana. Novi program mora raditi ispravno za bilo koju veličinu bloka (veličina bloka nije poznata unaprijed), a kraj bloka je zaključen ništicom. Promijenite broj podataka u bloku na tri i zaključite ga ništicom. Ne zaboravite ponovno asemblirati promijenjeni program. Ponovite simulaciju i provjerite ispravnost rada programa.

3. Upute za obavljanje pripreme

3.1. Terminal

Nakon instalacije i pokretanja aplikacije (vidi upute za instalaciju distribucije ATLAS-a), pojavljuje se prozor terminala (*urxvt*):



Svi alati programskog paketa ATLAS pokreću se iz naredbene linije terminala.

Ako vam je ovo prvi put da se susrećete s Linuxovom komandnom linijom i ljuskom Bash (*Bourne Again SHell*), obavezno prođite kroz jedan od ovih vodiča:

- http://gd.tuwien.ac.at/linuxcommand.org/lts0030.php
- http://tech.icrontic.com/articles/ultimate_newbie_guide_gnulinux_shell

Sve naredbe iz vodiča možete isprobati u vlastitom prozoru terminala(*xterm, rxvt*).

3.2. Struktura direktorija

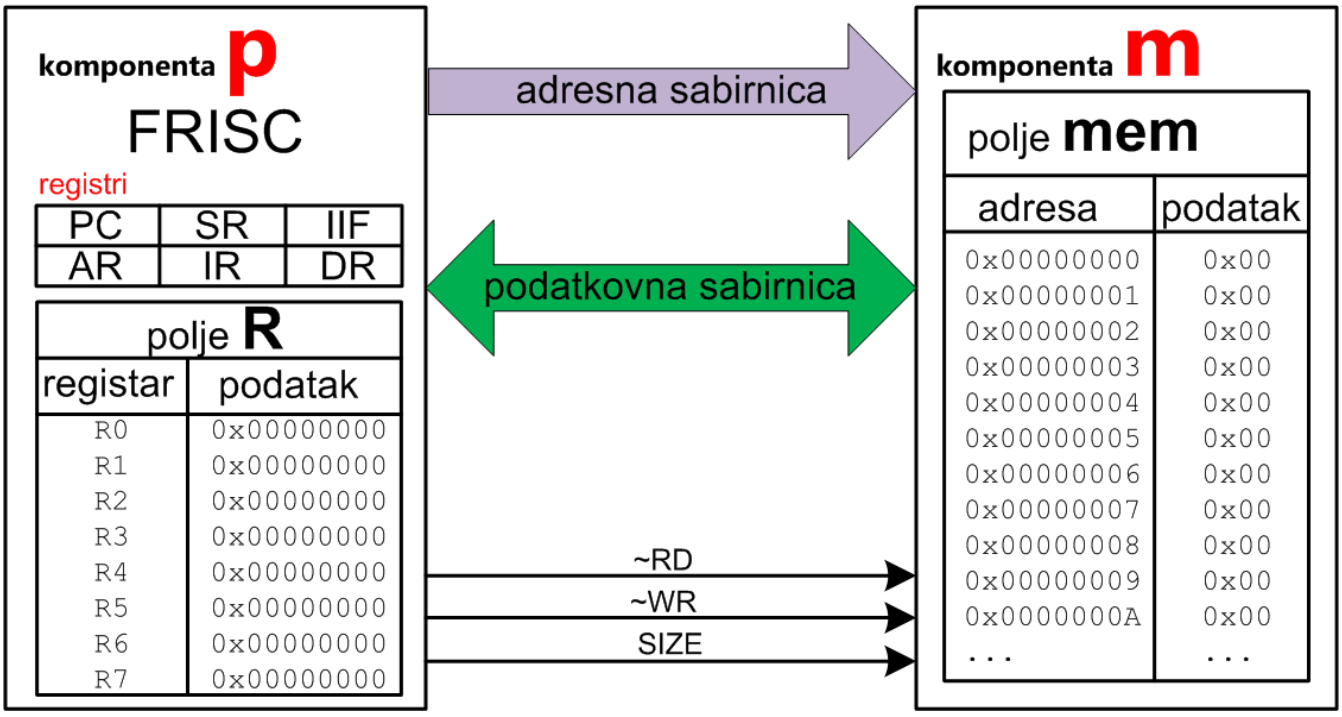
Hijerarhija Linux/Unix datotečne strukture sastoji se jednog primarnog (*root*) direktorija označenog sa *"/"*. Svi ostali direktoriji sadržani su u primarnom (*root*) direktoriju. Fizički diskovi (i/ili particije diskova, ako postoji više od jedne na disku) mapiraju se kao pod-direktoriji direktorija ***/mnt*** (*Puppy Linux*).

3.3. Simulacijski paket ATLAS

Programski paket ATLAS (***A**dvanced **T**ools and **L**anguages for Microprocessor **A**rhitecture **S**imulation* - http://staticweb.rasip.fer.hr/research/atlas/) namijenjen je simulaciji i proučavanju arhitekture i funkcionalnosti mikroprocesora.

ATLAS može simulirati bilo kakav mikroprocesor, ili bilo kakav digitalni sustav. Ponašanje svake komponente takvog sustava i njihova međusobna interakcija opisuje se posebnim jezicima (*COMDEL i SYSDEL*).

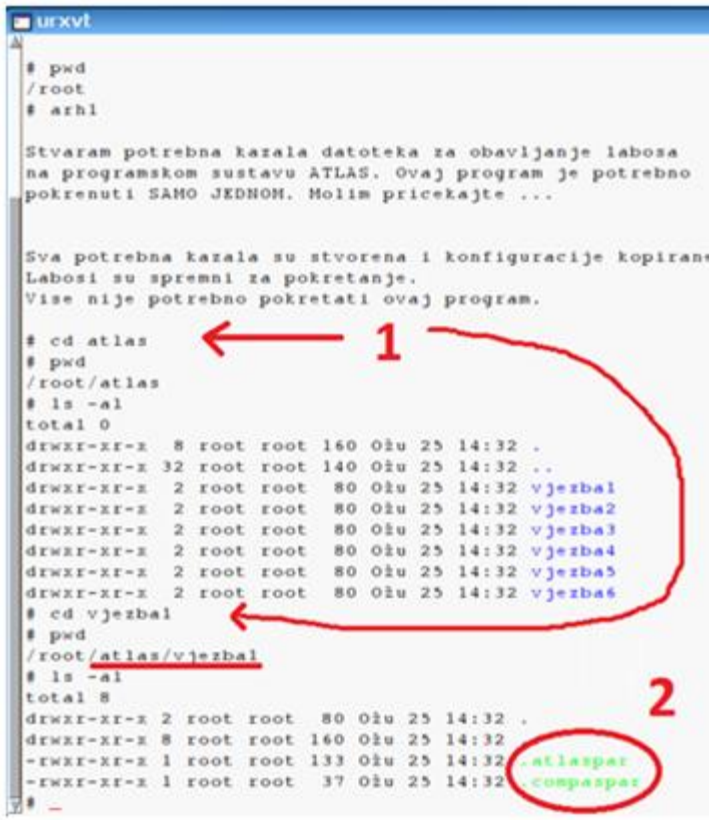
U kontekstu laboratorijskih vježbi treba znati da se naš računalni sustav opisan unutar ATLAS-a sastoji od komponenata ***p*** (procesor, tj. FRISC) i ***m*** (memorija), te od pomoćnih komponenata (*pu* i *decod*, nebitne za našu primjenu). Registri opće namjene modelirani su kroz polje ***R*** u modelu FRISC-a/komponenti ***p*** (ostali registri modelirani su pojedinačno), veličina memorije je 64kB, a podaci se nalaze u polju ***mem*** komponente ***m:***



3.4. Priprema konfiguracijskih datoteka

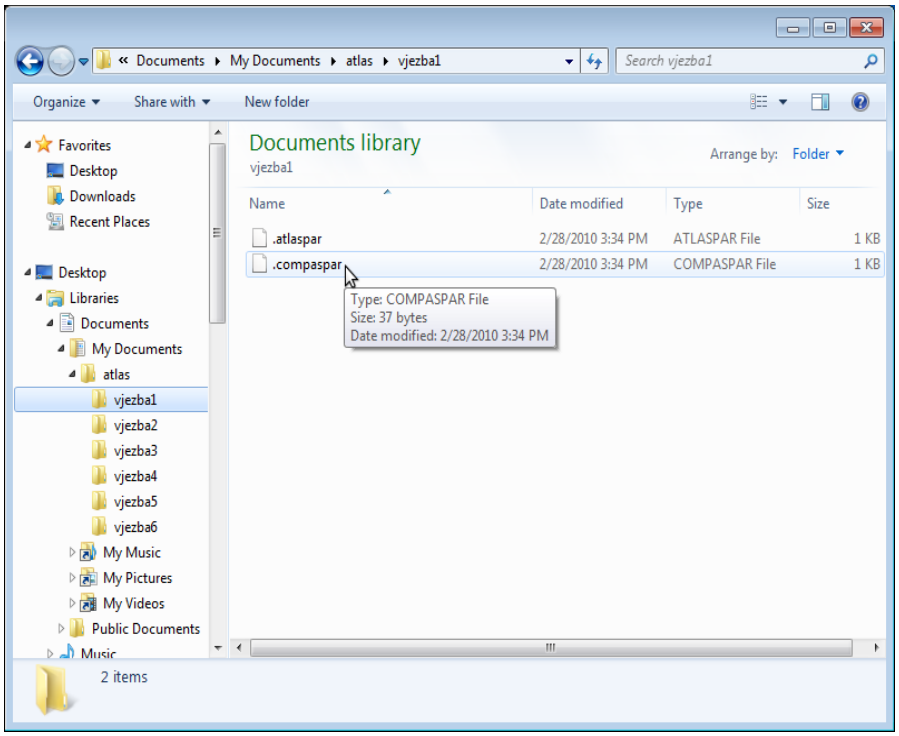
Da bi ATLAS-ovi alati ispravno radili, u direktoriju iz kojeg se pokreću moraju postojati valjane konfiguracijske datoteke: ***.atlaspar*** i ***.compaspar***. Konfiguracijske datoteke određuju koje će se definicije asemblerskog jezika koristiti za prevođenje programa, te koje će se definicije ATLAS-ovih komponenata koristiti pri simulaciji. Budući da se za svaku vježbu koriste drugačije postavke, pripremljena je skripta (***arh1***) koja u trenutnom direktoriju kreira direktorij ***atlas*** i pod-direktorije (***vjezba1, ... vjezba6***) za svaku vježbu, te u svaki od njih kopira odgovarajuće konfiguracijske datoteke.

Pri pokretanju konzole nalazite u */root* direktoriju vašeg korisničkog profila. Pokrenite skriptu ***arh1***, te se nakon toga pozicionirajte u jedan od direktorija *vjezbaX* (oznaka 1 na slici desno) i provjerite da li su konfiguracijske datoteke iskopirane (oznaka 2 na slici desno):



VAŽNO:

- Na Linux/Unix operacijskim sustavima datoteke čije ime počinje točkom implicitno su skrivene, da bi ih vidjeli naredbu ***ls*** treba pozvati s parametrom ***a*** (***ls -a***).
- Ako želite vježbe držati na nekoj drugoj lokaciji, **prije pokretanja skripte** *arh1* **pozicionirajte se** u direktorij gdje želite držati vježbe naredbom ***cd <staza>***



Skripta ***arh1*** u trenutnom direktoriju kreira sve potrebne direktorije za vježbe i kopira u njih potrebne konfiguracijske datoteke. Skriptu je potrebno pokrenuti samo jednom. U direktoriju *My Documents* (ili nekom drugom, ako ste se pozicionirali na neku drugu lokaciju) kreirat će se direktorij ***atlas*** i u njemu za svaku vježbu po jedan dodatni direktorij (***vjezba1, ... vjezba6***), te će se također u svaki od direktorija ***vjezba1-6*** prekopirati ispravne konfiguracijske datoteke (*.atlaspar* i *.compaspar*) za tu vježbu (slika lijevo).

Da bi obavljali pojedinu vježbu, pozicionirajte se u direktorij pripadne vježbe. Npr. za izradu prve vježbe u terminalu se pozicionirajte u direktorij ***atlas/vjezba1*** pomoću naredbe '***cd atlas/vjezba1***'. Sve daljnje radnje vezane uz pojedinu vježbu obavljate u njenom direktoriju (pokretanje programa ATLAS-a, spremanje vaših asemblerskih (mnemoničkih) programa itd.).

ZADATAK 1: U terminalu provjerite u kojem se direktoriju nalazite naredbom *pwd'*. Ispišite sadržaj direktorija naredbom *ls*. Pozicionirajte se u direktorij u kojem želite raditi naredbom '*cd ime_direktorija*' te pokrenite skriptu *arh1*. Zatim se naredbom *cd'* pozicionirajte u direktorij *atlas/vjezba1*.

3.5. Pisanje teksta programa, uređivač (editor) teksta

Za razliku od Windows tekstualnih datoteka, kod Linux/Unix tekstualnih datoteka svaki redak mora biti terminiran **samo znakom *LF*** (*line feed*, '\n', 0x0A, 10 dekadski). Kod Windows tekstualnih datoteka, svaki redak je terminiran znakovima ***CR* i *LF*** (CR+LF, 0x0D 0x0A). Za više informacija pročitajte http://en.wikipedia.org/wiki/Newline.

Da bi ATLAS alati ispravno preveli vaše programe, isti trebaju imati **kraj svakog retka označen samo znakom LF**. Budući da većina Windows uređivača teksta ne nudi takvu mogućnost, u instalaciju ATLASa uključen je uređivač teksta Geany („edit“ ikonica na desktopu) koji je unaprijed podešen da snima datoteke u ispravnom formatu (sa znakovima LF na kraju retka).

Dodatno, **svaki program** koji napišete **na kraju teksta** mora sadržavati **jedan prazan redak**.

ZADATAK 2: Pokrenite uređivač teksta,te pretipkajte ili prekopirajte tekst programa:

```
        ORG 0
        MOVE    50, R1
        MOVE    70, R2
        MOVE    5, R3

PETLJA  LD    R0, (R1)
        STORE   R0, (R2)
        ADD     R1, 4, R1
        ADD     R2, 4, R2
        SUB     R3, 1, R3
        JR_NZ   LOOP

        HALT

        ORG    50
        DW      1, 2, 3, 4, 5

        ORG    70
        DS     %D 20
```

Pravila pisanja mnemoničke datoteke

- Mnemonička datoteka je retkovno orijentirana - svaka naredba ili pseudo-naredba nalazi se u jednom retku.
- svaki program **na kraju teksta** mora sadržavati jedan **prazan redak.**
- Pseudo-naredbe i mnemoničke naredbe za FRISC se pišu VELIKIM slovima. Mnemoničke naredbe za ARM mogu se pisati i malim i velikim slovima.
- Komentar počinje znakom ';' i proteže se do kraja retka.
- Brojevi se pišu u heksadekadskoj bazi (osim ako sami drugačije ne zadate).
- Labele se pišu od prvog stupca, a ispred mnemoničkih naredaba mora stajati barem jedan znak razmaka.
- Pseudo-naredbe nisu naredbe za procesor. One upravljaju radom prevodioca.

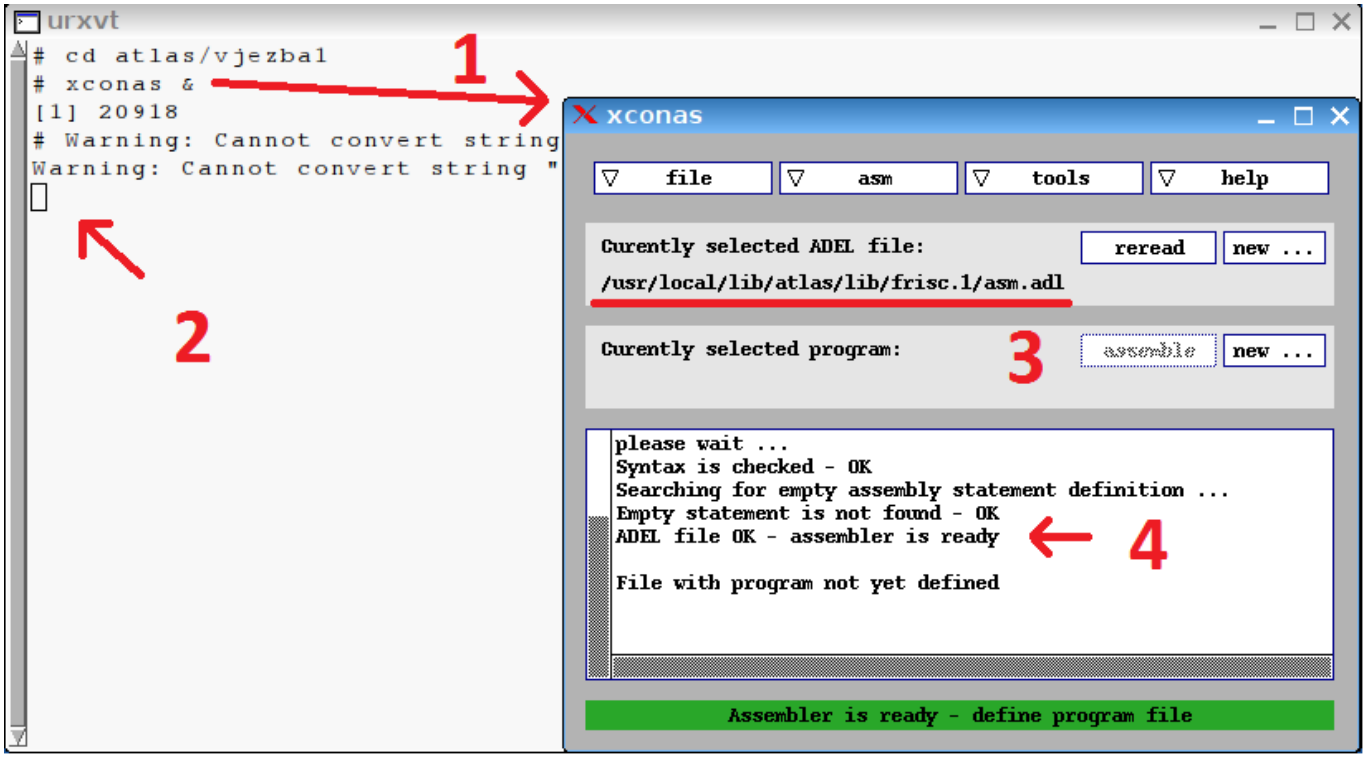
Spremite datoteku u direktorij *atlas/vjezba1*, obavezno s nastavkom ***.a*** (npr. ***vjezba0.a***).

[Navedeni program kopira podatke iz jednog bloka memorije u drugi. Početni blok je na adresi 50_{16} i sadrži pet podataka (32-bitnih): 1, 2, 3, 4, 5. Odredišni blok je na adresi 70_{16}. Program u petlji kopira jedan po jedan podatak. Kao brojač petlje koristi se registar R3. Podatci se adresiraju registrima R1 (početni blok) i R2 (odredišni blok).]

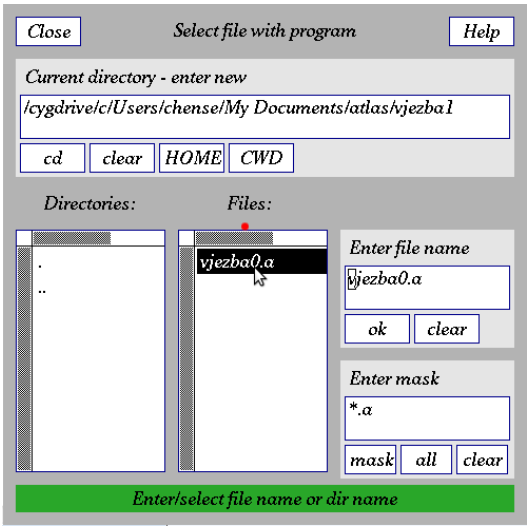
3.6. Asembliranje programa

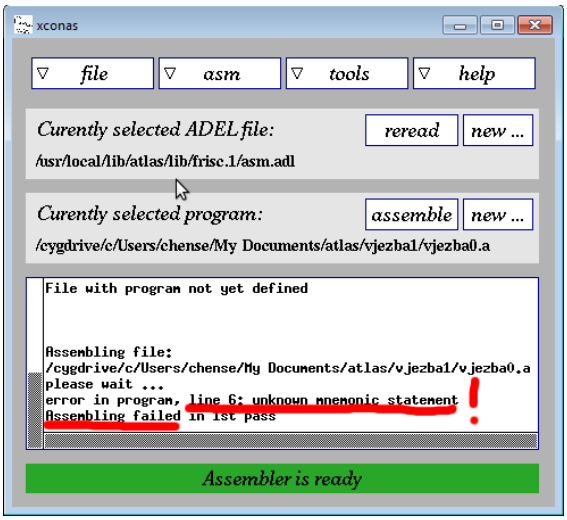
Naredbom *xconas* pokreće se asembler, a ako iza naredbe dodate znak '&' asembler će se pokrenuti u pozadini, pa ćete u terminalu dalje moći utipkavati naredbe ili pozivati druge programe.

Na slici dolje: *xconas* se pokreće u pozadini (1) i vraća prompt (2). Budući da se u direktoriju nalaze ispravne konfiguracijske datoteke, *xconas* odabire ispravnu ADEL datoteku (ona sadrži definiciju asemblerskog jezika FRISC-a) – oznaka 3 na slici – te je spreman za prevođenje datoteke s izvornim kodom (oznaka 4 na slici) :



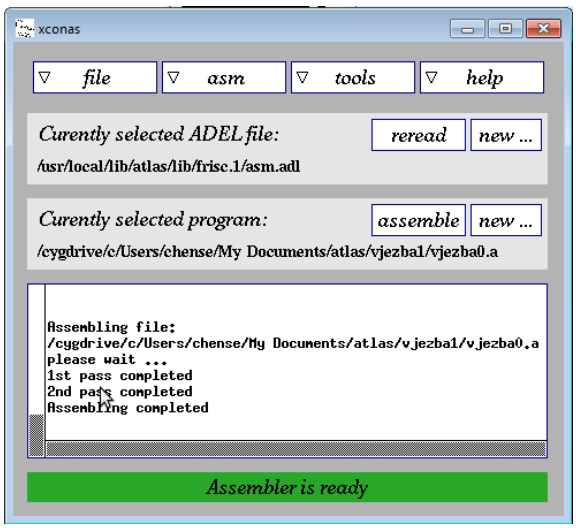
Nakon toga treba kliknuti na **donji** gumb *'new ...'* (slika iznad) te odabrati datoteku s izvornim kodom:





Nakon klika na gumb '*ok*' (slika iznad lijevo) datoteka s izvornim kodom se asemblira, te ako postoje greške ispisuju se poruke (slika iznad desno).

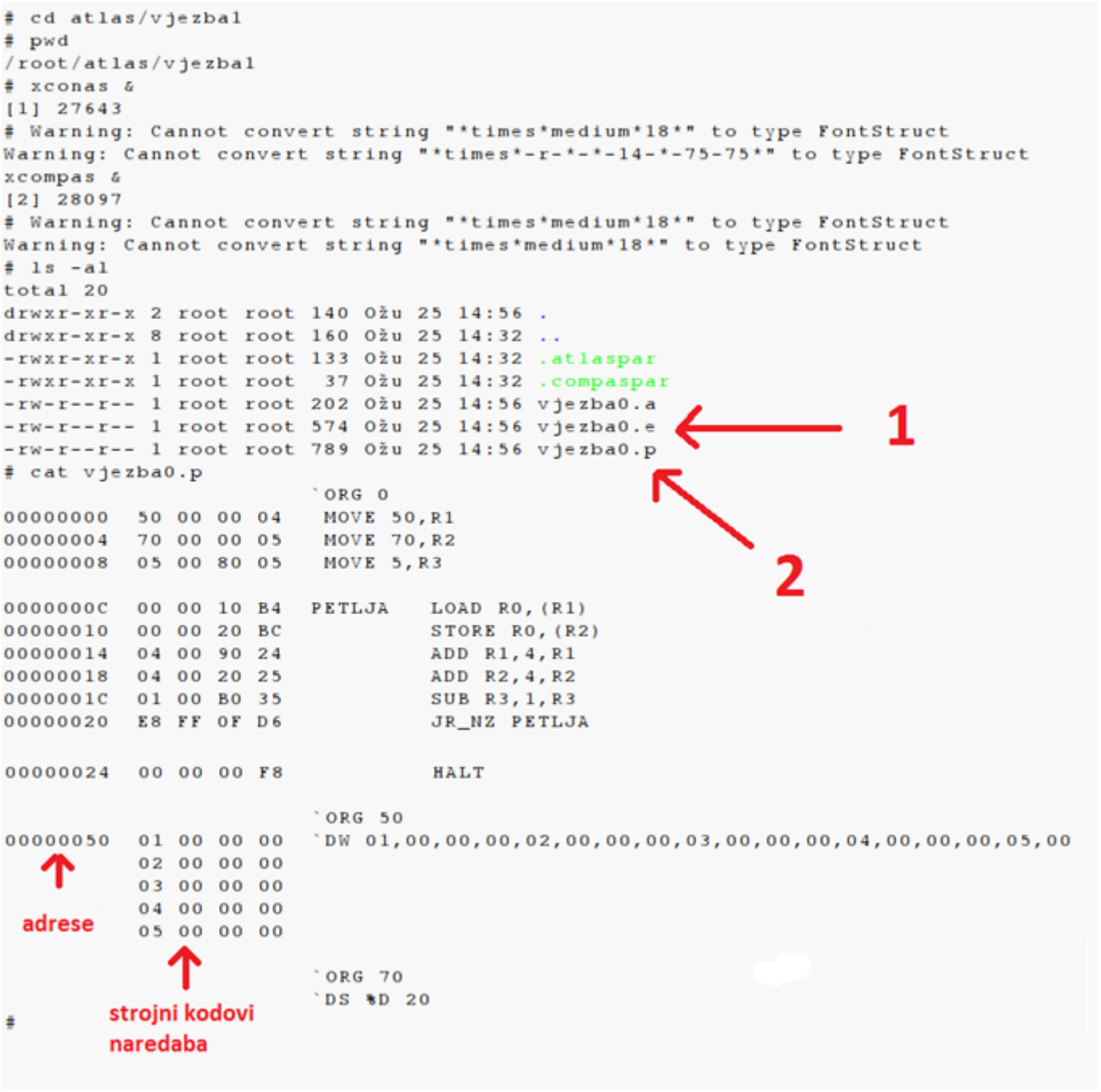
Nakon što pomoću uređivača teksta uklonite greške i spremite izmijenjenu datoteku, kliknite na gumb '*assemble*' – ako je sve u redu, dobit ćete sljedeći ispis:



ZADATAK 3: U terminalu provjerite nalazite li se u kazalu *atlas/vjezba1* (naredbom *'pwd'*). Pokrenite asembler *xconas* i asemblirajte program. Popravite sve greške u kodu te ponovite postupak asembliranja dok ne završi bez grešaka.

3.7. Pokretanje simulacije

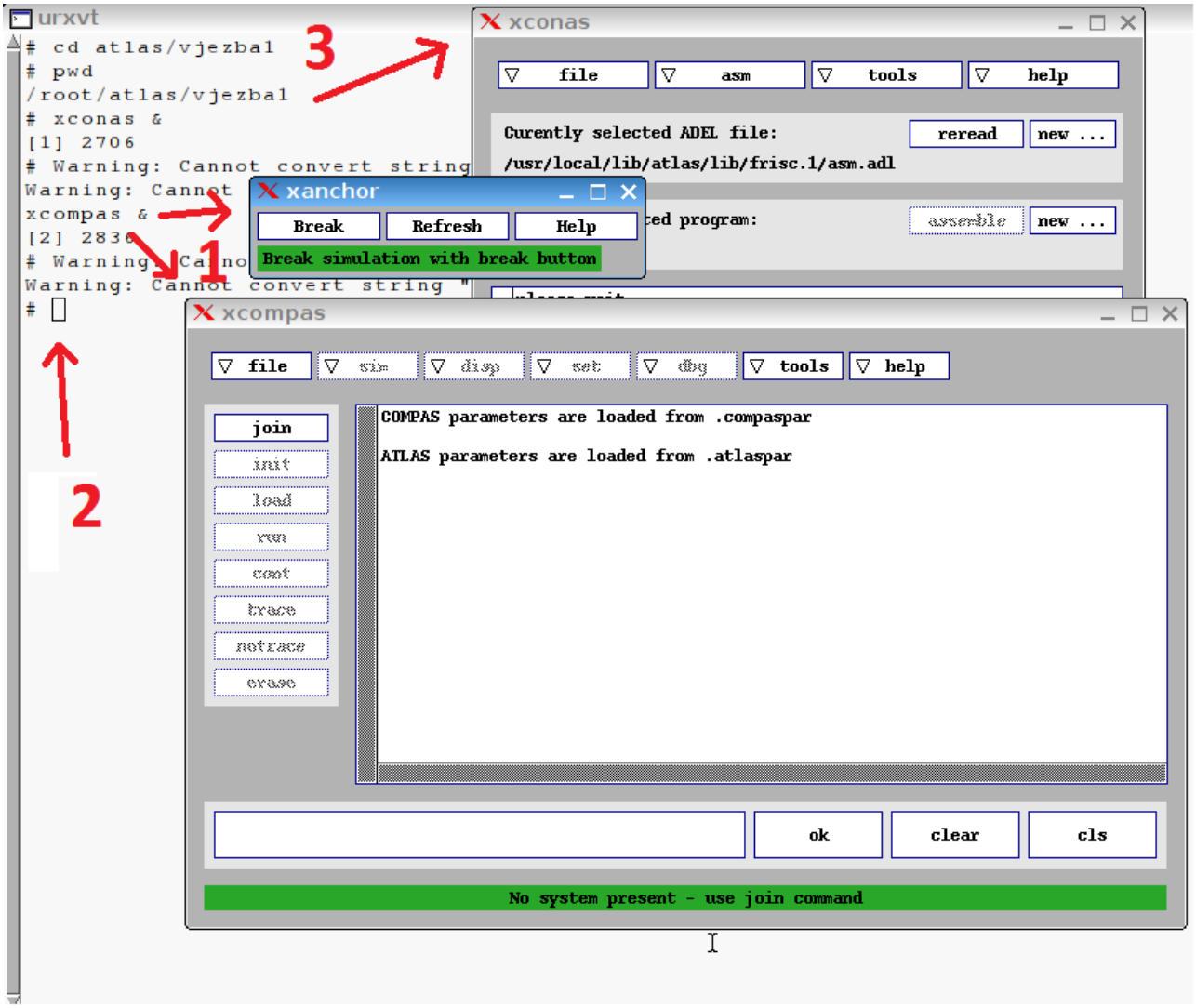
Asembler će nakon uspješnog asembliranja stvoriti datoteku sa strojnim kodom i ispisnu datoteku. Ove će datoteke imati isto ime kao i datoteka s mnemoničkim programom, ali će imati ekstenziju *.e*, odnosno *.p* (npr. mnemonički program iz datoteke *vjezba0.a* prevest će se u strojni kod u datoteci *vjezba0.e* – oznaka 1 na slici dolje – i u ispisnu datoteku *vjezba0.p* – oznaka 2 na slici dolje). Ako ispišemo sadržaj ispisne datoteke (UNIX naredba *cat* ispisuje datoteku na standardni izlaz), vidimo na koje je adrese asembler smjestio pojedine naredbe (prvi stupac), strojni kod naredbe (drugi stupac), te mnemonički zapis programa. Neke naredbe se prilikom asembliranja prilagođavaju procesoru pa je ispis nešto drugačiji nego originalna naredba (npr. *ORG* → *`ORG* ili *DW 1* → *`DW 01,00,00,00*).




```
# cd atlas/vjezba1
# pwd
/root/atlas/vjezba1
# xconas &
[1] 27643
# Warning: Cannot convert string "*times*medium*18*" to type FontStruct
Warning: Cannot convert string "*times*-r-*-*-14-*-75-75*" to type FontStruct
xcompas &
[2] 28097
# Warning: Cannot convert string "*times*medium*18*" to type FontStruct
Warning: Cannot convert string "*times*medium*18*" to type FontStruct
# ls -al
total 20
drwxr-xr-x 2 root root 140 Ožu 25 14:56 .
drwxr-xr-x 8 root root 160 Ožu 25 14:32 ..
-rwxr-xr-x 1 root root 133 Ožu 25 14:32 .atlaspar
-rwxr-xr-x 1 root root  37 Ožu 25 14:32 .compaspar
-rw-r--r-- 1 root root 202 Ožu 25 14:56 vjezba0.a
-rw-r--r-- 1 root root 574 Ožu 25 14:56 vjezba0.e
-rw-r--r-- 1 root root 789 Ožu 25 14:56 vjezba0.p
# cat vjezba0.p
                        `ORG 0
00000000   50 00 00 04   MOVE 50,R1
00000004   70 00 00 05   MOVE 70,R2
00000008   05 00 80 05   MOVE 5,R3

0000000C   00 00 10 B4   PETLJA   LOAD R0,(R1)
00000010   00 00 20 BC            STORE R0,(R2)
00000014   04 00 90 24            ADD R1,4,R1
00000018   04 00 20 25            ADD R2,4,R2
0000001C   01 00 B0 35            SUB R3,1,R3
00000020   E8 FF 0F D6            JR_NZ PETLJA

00000024   00 00 00 F8            HALT

                        `ORG 50
00000050   01 00 00 00  `DW 01,00,00,00,02,00,00,00,03,00,00,00,04,00,00,00,05,00
           02 00 00 00
           03 00 00 00
           04 00 00 00
           05 00 00 00

                        `ORG 70
                        `DS %D 20
#
```

ZADATAK 4: Nakon što uspješno asemblirate program, u terminalu ispišite sadržaj vaše ispisne datoteke naredbom '*cat*', kao na slici gore. Priložite sliku (*screenshot*) prozora terminala s ispisanim sadržajem datoteke.

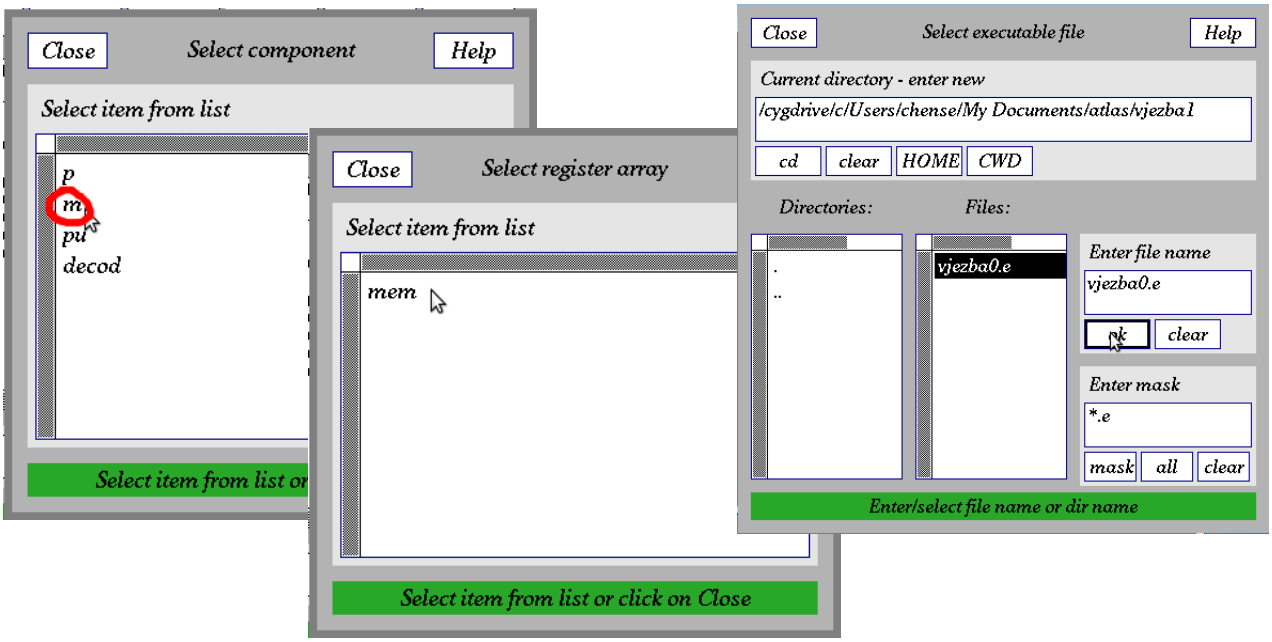
U terminalu pokrenite *xcompas*, također u pozadini – pojavljuju se prozori *xcompas* i *xanchor* (slika dolje, oznaka 1). Prompt se vraća, pa možemo i dalje koristiti terminal (slika dolje, oznaka 2). Program *xconas* je i dalje pokrenut u pozadini (slika dolje, oznaka 3):



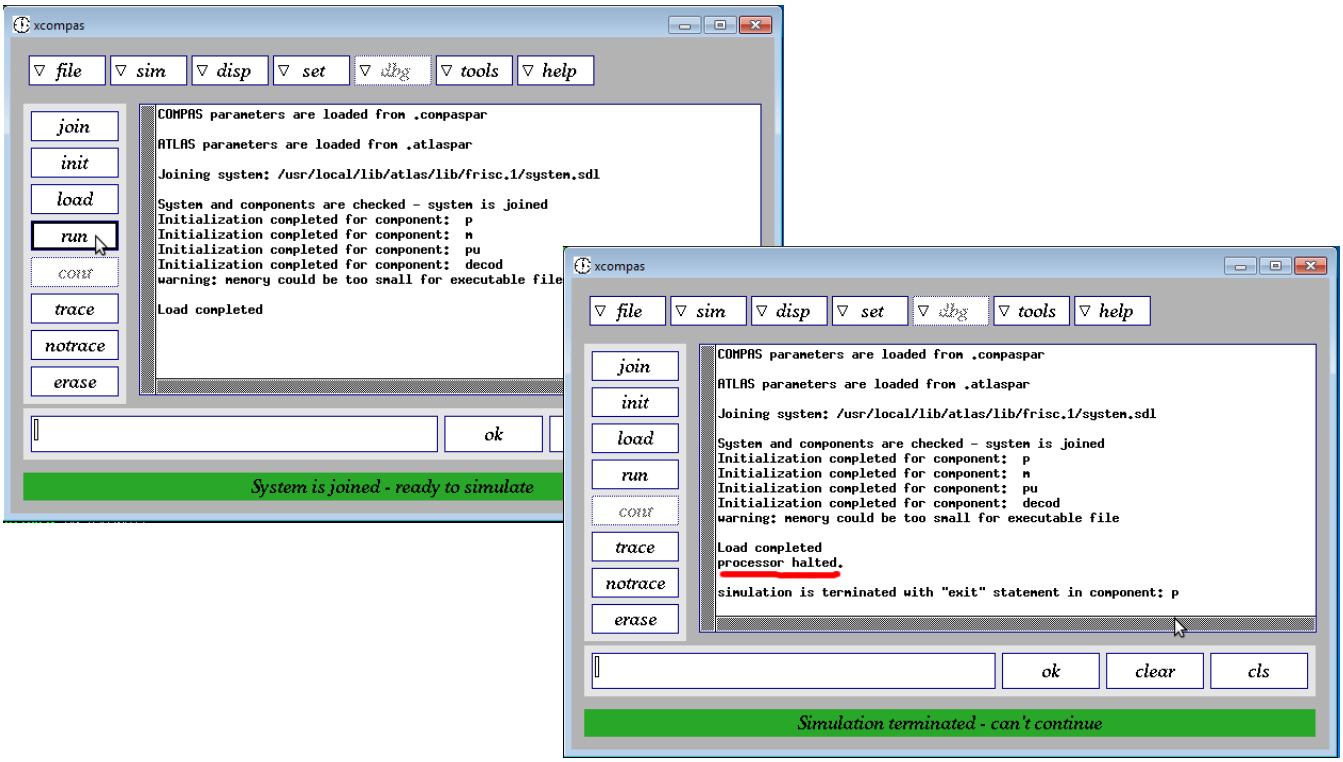
Prozor *xanchor* (slika gore) služi za zaustavljanje simulacije ako *xcompas* prestane reagirati (npr. ako program izvodi beskonačnu petlju).

Nakon pokretanja simulatora većina potrebnih akcija dostupna vam je pomoću gumbi na lijevoj strani glavnog prozora *xcompasa* ili iz izbornika. Na početku je uvijek potrebno **učitati model u simulator**. Učitavanje modela u simulator obavlja se gumbom ***join***. Model je dovoljno učitati **samo jednom** - na početku rada sa simulatorom, prije nego počnete sa simulacijama.

Prije svake runde simuliranja potrebno je **inicijalizirati model**. Prvi dio inicijalizacije se obavlja gumbom ***init***. Drugi dio inicijalizacije je **punjenje izvršnog programa u memoriju** što se obavlja gumbom ***load***. Pri tome treba odabrati gdje se stavlja program (*m*, *mem*) i ime datoteke sa strojnim kodom (datoteka s nastavkom *.e*, dobivena asembliranjem):

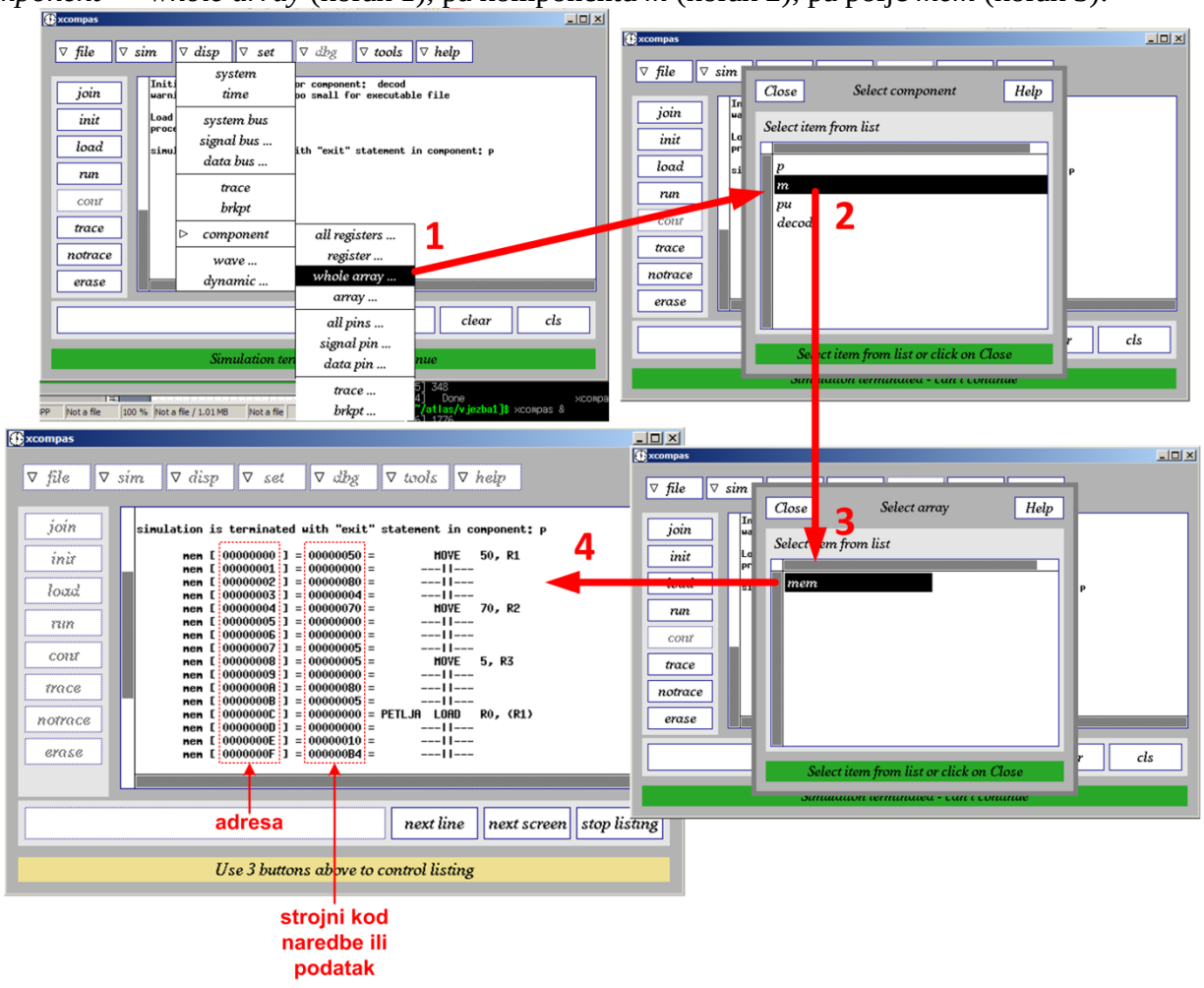


Ispravno pokretanje simulacije moguće je tek nakon uspješne inicijalizacije. Simulacija se pokreće gumbom *run* (slika lijevo), a kad završi *xcompas* ispiše poruku o završetku simulacije (slika desno):



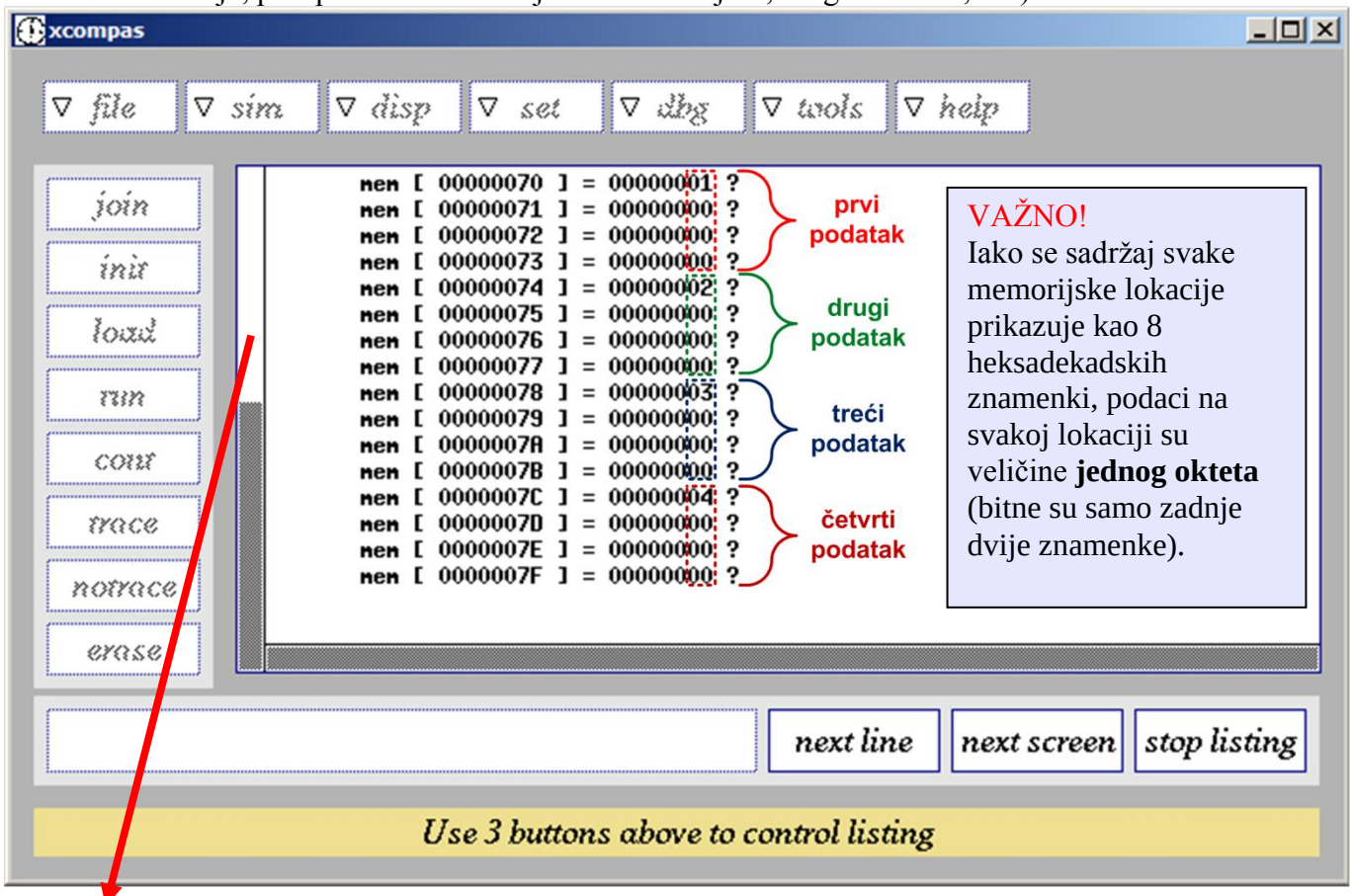
Nakon što je simulacija gotova (završena) ili zaustavljena, postupak simuliranja se može ponoviti (prvo ponoviti inicijalizaciju pa tek onda ponovno pokrenuti simulaciju).

Rezultate izvođenja programa treba provjeriti pregledom sadržaja memorije – odabrati *disp* -> *component* -> *whole array* (korak 1), pa komponentu *m* (korak 2), pa polje *mem* (korak 3):



Gumbima '*next line*', '*next screen*' i '*stop listing*' (slika gore, korak 4) upravljamo ispisom (možemo pregledavati sljedeće memorijske lokacije).

Sadržaj lokacija 70_{16}-84_{16} ako je program ispravno izvršio trebao bi biti sljedeći (svaki podatak zauzima 4 lokacije, prvi podatak označen je crvenom bojom, drugi zelenom, itd.):



POMAK (SCROLL)

Ako na traku za pomak (na bijeli dio trake) kliknete:

- **lijevom tipkom miša = ispis se pomiče dolje**
- **desnom tipkom = ispis se pomiče gore**

Ako srednjim gumbom miša "uhvatite" traku možete pomicati sadržaj gore-dolje u željenom smjeru

ZADATAK 5: Učitajte model gumbom *join*. Inicijalizirajte model gumbom *init*, a nakon toga napunite strojni kôd programa ***vjezba0.e*** u komponentu ***m*** u polje *mem* pomoću gumba *load* . Pokrenite simulaciju gumbom *run*. Simulacija bi trebala završiti porukom *simulation is terminated with "exit" statement in component: p.*

ZADATAK 6: Provjerite ispravnost rezultata izvođenja programa (tj. jesu li podaci preneseni u dio memorije s početnom adresom 70_{16}), tako da prikažete sadržaj memorijskih lokacija 70_{16}-84_{16} – u izborniku odaberite *disp -> component -> array*, upišite početnu adresu (upisuju se heksadekadske vrijednosti), te provjerite da li ispis odgovara očekivanjima. Priložite sliku (*screenshot*) prozora *xcompasa* s prikazanim memorijskim lokacijama.

ZADATAK 7: Na koji su način 32-bitni podaci smješteni u memoriji? Na kojoj adresi se nalazi najznačajniji oktet, a na kojoj najmanje značajan oktet? Kako se naziva takav smještaj?

3.8. *Traženje grešaka u programu – prekidne točke (break points), točke praćenja (trace points), ispis sadržaja registara i memorije*

Prekidna točka *(break point)* omogućuje zaustavljanje izvođenja mnemoničkog programa kada se dosegne određena naredba. Naredba na kojoj treba zaustaviti izvođenje se zadaje pomoću adrese te naredbe. Adresu naredbe možete saznati pregledom ispisne datoteke (istog imena kao i datoteka s programom, samo ima nastavak *.p*).

Želimo postaviti prekidnu točku na naredbu *LOAD R0, (R1)*. Unutar terminala ispišemo sadržaj ispisne datoteke (naredbom *cat*) te iščitamo adresu naredbe:

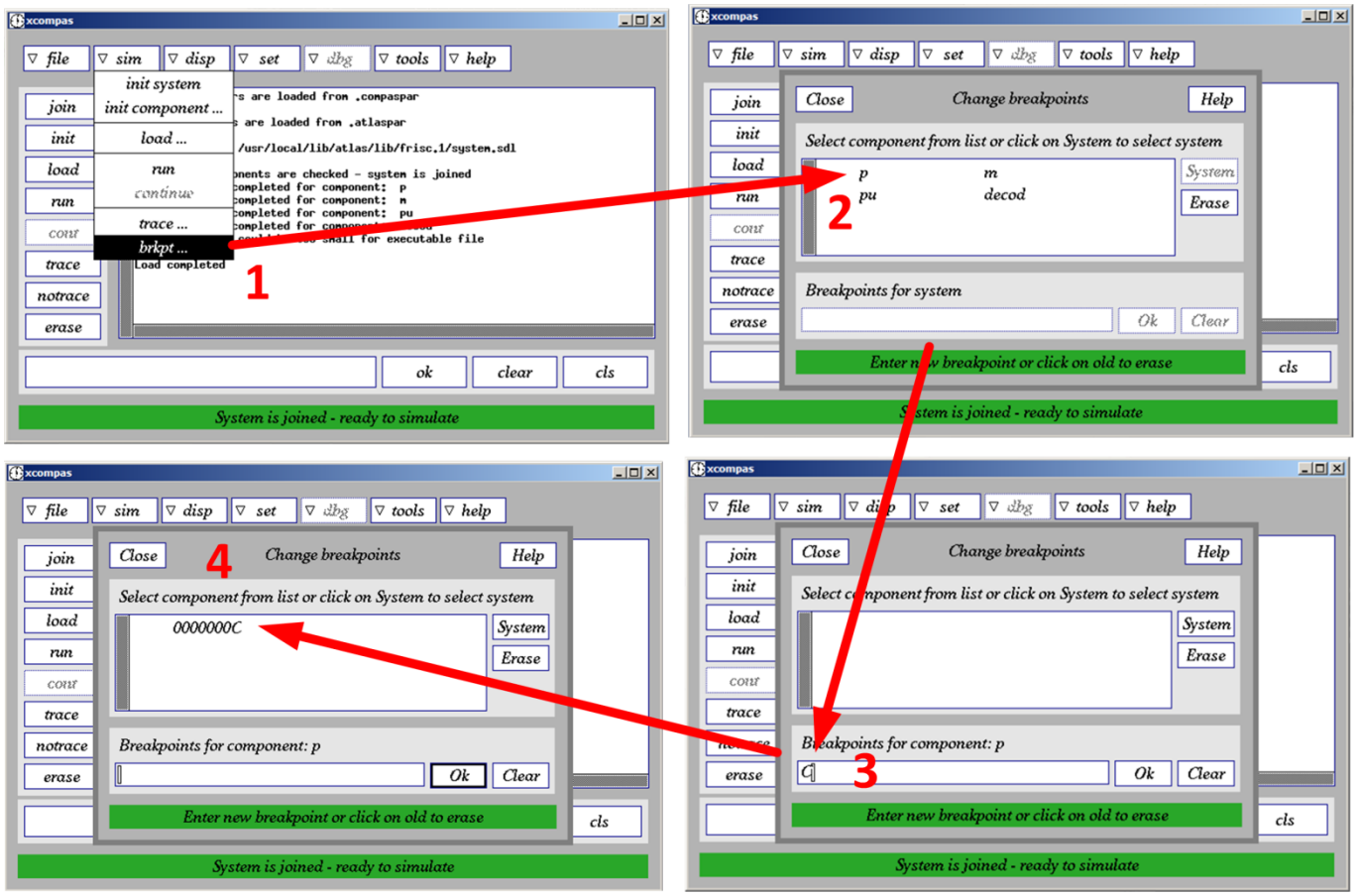
```
urxvt
# cd atlas/vjezba1
# ls -al
total 20
drwxr-xr-x 2 root root 140 Ožu 25 14:56 .
drwxr-xr-x 8 root root 160 Ožu 25 14:32 ..
-rwxr-xr-x 1 root root 133 Ožu 25 14:32 .atlaspar
-rwxr-xr-x 1 root root  37 Ožu 25 14:32 .compaspar
-rw-r--r-- 1 root root 201 Ožu 25 15:31 vjezba0.a
-rw-r--r-- 1 root root 538 Ožu 25 15:31 vjezba0.e
-rw-r--r-- 1 root root 789 Ožu 25 15:31 vjezba0.p
# cat vjezba0.p
```

adresa	strojni kod	tekst programa
		`ORG 0
00000000	50 00 80 04	MOVE 50,R1
00000004	70 00 00 05	MOVE 70,R2
00000008	05 00 80 05	MOVE 5,R3
0000000C	00 00 10 B4	PETLJA LOAD R0,(R1)
00000010	00 00 20 BC	STORE R0,(R2)
00000014	04 00 90 24	ADD R1,4,R1
00000018	04 00 20 25	ADD R2,4,R2
0000001C	01 00 B0 35	SUB R3,1,R3
00000020	E8 FF 0F D6	JR_NZ PETLJA
00000024	00 00 00 F8	HALT
		`ORG 50
00000050	01 00 00 00	`DW 01,00,00,00,02,00,00,00,03,00,00,00,04,00,00,00,05,00
	02 00 00 00	
	03 00 00 00	
	04 00 00 00	
	05 00 00 00	
		`ORG 70
		`DS %D 20

naredba LOAD smještena je na adresi 12_{10},tj C_{16}

#

Prekidne točke zadaju se pomoću stavke *brkpt* u izborniku *sim* (korak 1):



Odaberemo komponentu *p* (korak 2), zatim upišemo vrijednost *C* (korak 3), klik na *OK* i nakon toga se prekidna točka pojavljuje u listi, pa klik na *Close* (korak 4).

Nakon pokretanja simulacije ispisuje se poruka *'Simulation is suspended with breakpoint in component: p'*.

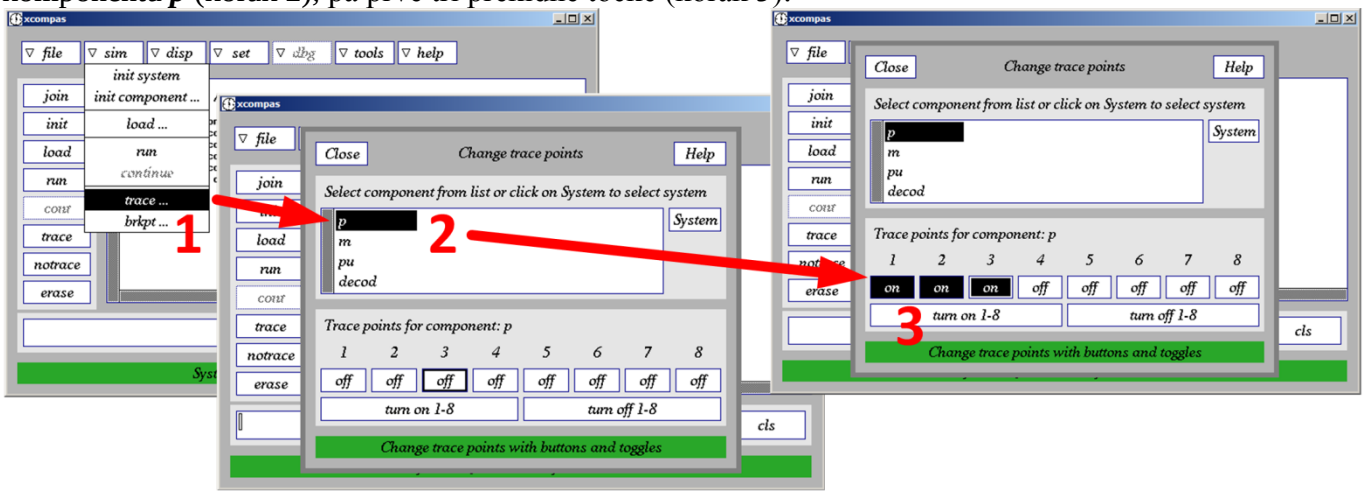
Točke praćenja omogućuju detaljnije praćenje tijeka simulacije, tj. praćenje promjena stanja modela tijekom simulacije.

- Uključivanje i isključivanje **svih** točaka praćenja u cijelom modelu obavlja se gumbima ***trace*** i ***notrace***.
- Uključivanje i isključivanje **pojedinih** točaka obavlja se pomoću stavke ***trace*** u izborniku ***sim*.**

Namjena pojedinih točaka definiranih za procesor FRISC:

- *trace 1*: ispis sadržaja registara procesora nakon svake naredbe.
- *trace 2*: ispis mnemoničkog oblika naredbe koja se upravo izvodi.
- *trace 3*: čekanje da korisnik potvrdi prelazak na izvođenje sljedeće naredbe (donjim gumbom *cont*).
- *trace 4-8*: ovisno o definiciji trenutne vježbe.

Ako za procesor želimo uključiti prekidne točke 1-3, odaberemo *sim* -> *trace* (korak 1), pa komponentu ***p*** (korak 2), pa prve tri prekidne točke (korak 3):



Pregledavanje stanja modela, tj. registara u procesoru, lokacija u memoriji, simuliranog vremena itd. ostvaruje se stavkama izbornika *disp*. **Prilikom predaje svake sljedeće vježbe bit će potrebno pokazati stanje određenih dijelova modela, te će se od vas zahtijevati suvereno vladanje opcijama.**

Promjena stanja modela (registri, memorijske lokacije itd.) ostvaruje se stavkama izbornika *set*. Izlazak iz programa ostvaruje se stavkom *quit* u izborniku *file*.

ZADATAK 8: Ponovite simulaciju s uključenim točkama praćenja (*trace*). Provodite simulaciju naredbu po naredbu (*init* -> *load* -> *trace* -> *run* -> *cont* -> *cont* -> *cont*...), analizirajte ispis i provjerite da li sve naredbe imaju očekivani utjecaj na registre procesora.

ZADATAK 9: Isključite točke praćenja te postavite prekidnu točku na naredbu *"LOAD R0, (R1)"*. Ponovno pokrenite simulaciju (*init* -> *load* -> *run*). Koja je vrijednost registra *PC* u trenutku kad je simulacija prekinuta prekidnom točkom? Provjerite pregledom registara procesora! Priložite sliku (*screenshot*) prozora *xcompasa* s prikazanim registrima procesora u tom trenutku.

ZADATAK 10: Prepravite program tako da veličina bloka ne bude poznata unaprijed. Novi program mora raditi za bilo koju veličinu bloka, pri čemu je blok zaključen ništicom. Promijenite broj podataka u bloku na tri i zaključite ga ništicom. Ne zaboravite ponovno asemblirati promijenjeni program. Ponovite simulaciju i provjerite ispravnost rada programa pregledom odredišnih memorijskih lokacija. Ako program ne funkcionira ispravno, pronađite greške koristeći prekidne točke i točke praćenja i popravite ih. Priložite programski kôd rješenja.

Nakon što ste gotovi s vježbom zatvorite simulator, asembler i editor (ukoliko to već prije niste učinili).

3.9. Česti problemi i pripadna rješenja

Ne mogu asemblirati program!!!

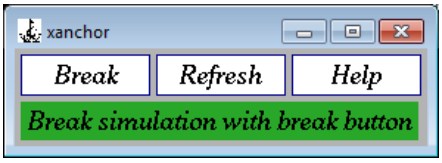
Vrlo česte greške su pisanje programa malim slovima, pisanje naredbe od prvog stupca gdje mogu biti samo labele, zamjena znamenke 0 velikim slovom o, zamjena znamenke 1 slovom l, kao i pisanje apostrofa ispred pseudo-naredaba (npr. ispred `ORG` – starije inačice koristile su naredbu `` `ORG ``, dok nova inačica koristi naredbu `ORG`).

Miš mi ne radi, ne mogu scrollati gore/dolje!!!

Scroll-anje (pomak) u programima conas i compas se najlakše ostvaruje tako da pozicionirate miša iznad trake za pomak i srednjim gumbom miša "uhvatite" traku te je pomičete gore-dolje u željenom smjeru. Drugi način pomicanja je da pozicionirate miš iznad trake za pomak te se pritiscima **lijeve tipke miša** pomičete prema **dolje** ili **desnom tipkom** prema **gore**.

Kako zaustaviti simulaciju? (imam beskonačnu petlju !!!)

Ponekad je potrebno zaustaviti simulaciju koja je u tijeku. To se može izvesti jedino pomoću gumba ***break*** u malom prozoru *xanchor* koji se otvorio zajedno s glavnim prozorom simulatora *xcompas*. Nakon zaustavljanja simulacije moguće ju je ponovno pokrenuti gumbom *cont* u *xcompasu*.



Razlikujte nastavljanje zaustavljene simulacije od pokretanja simulacije iznova, što ostvarujete sa *run* i prije čega obavezno morate inicijalizirati model.

Program xconas/xcompas mi ne želi otvoriti datoteku s programom!!!

Nekim korisnicima se pojavljuje problem kod otvaranja datoteka. Do njega dolazi jer ATLAS ima ograničenu duljinu putanje do datoteka s vašim mnemoničkim programima. Jedino rješenje je da cijeli direktorij *atlas* s pod-direktorijima za vježbe premjestite u neki drugi direktorij čija putanja ima kraće ime. I dalje sve morate raditi u direktoriju *atlas* i njegovim pod-direktorijima.

Ispis unutar xcompasa je ... čudan, pomaknut, nesuvisao ?!

Ako mišem kliknete u veliko središnje polje s tekstom programa xcompasa, sve poruke se ispisuju na mjestu gdje ste kliknuli mišem. Rješenje je izbjegavanje klikanja po tom polju, a ako ste već kliknuli, scrollajte polje s tekstom do kraja i kliknite ispod zadnjeg retka teksta čime ćete postaviti ispis da ubuduće kreće od te pozicije.