

Arhitektura informaciono-komunikacionih sistema

## Little Man Computer - simulator

Mihajlo Krunic 2020/0366

Elektrotehnički fakultet

28. jun 2022.

# Sadržaj

<b>1</b>	<b>Uvod – o Little Man Computer-u</b>	<b>2</b>
1.1	Arhitektura . . . . .	2
1.2	Instrukcioni set . . . . .	2
1.3	Instrukcioni ciklus . . . . .	3
1.4	Instrukcije grananja . . . . .	3
<b>2</b>	<b>Primeri programa</b>	<b>5</b>
2.1	Sabiranje dva broja . . . . .	5
2.2	Maksimum dva broja . . . . .	5
2.3	Množenje 2 broja . . . . .	6
<b>3</b>	<b>Asemblerski jezik</b>	<b>7</b>
3.1	Simbolički kodovi i oznake . . . . .	7
<b>4</b>	<b>Simulator</b>	<b>9</b>
4.1	Uvod . . . . .	9
4.2	Rad i uputstvo . . . . .	9
<b>5</b>	<b>Zaključak</b>	<b>13</b>

# 1 Uvod – o Little Man Computer-u

Model LMC (little man computer) je razvijen 1960-ih godina od strane Stjuarta Madnika i Džona Donovana sa MIT-a, gde je bio korišćen kao nastavno sredstvo na kursevima računarskih nauka. Zasnovan na jednostavnoj fon Nojmanovoj arhitekturi, implementira sve osnove funkcionalnosti modernih računara. Na njemu se može programirati u mašinskom kodu (mada je to decimalni a ne binarni) ili asemblerskom kodu.

## 1.1 Arhitektura

LMC se sastoji od odvojene poštanske prostorije, koja je predstavljena debelom linijom koja okružuje model na dijagramu. Unutar poštanske sobe nalazi se više stavki.

Prvenstveno, u sobi se nalazi 100 poštanskih sandučića, obeleženih brojevima (adresama) od 00 do 99. U ovom sistemu, adresa svakog sandučića se može predstaviti sa dve decimalne cifre. Predviđeno je da svako sanduče sadrži papirić na kojem je napisan trocifren decimalan broj.

Sledeća stavka u prostoriji je kalkulator (akumulator). On se koristi za unos i privremeno skladištenje brojeva, kao i za njihovo sabiranje i oduzimanje. Nije predviđena mogućnost rada sa negativnim brojevima, kao ni sa brojevima koji imaju više od tri cifre.

Treća stavka je dvocifren brojač (brojač instrukcija), koji može da se inkrementira ili da mu se dodeli specifična vrednost. Dugme za resetovanje brojača se nalazi izvan poštanske prostorije.

Poslednju stavku predstavlja sam mali čovek (little man) koji će obavljati sve zadatke vezane za rad ovog računara.

Pored dugmeta za resetovanje, interakcija između LMC-a i spoljašnje sredine se sprovodi kroz ulaznu i izlaznu korpu. Korisnik izvan poštanske prostorije komunicira sa malim čovekom ubacujući papirić sa trocifrenim brojem u korpu, koju će mali čovek pročitati u odgovarajućem trenutku. Slično, mali čovek može ispisati trocifren broj na papiriću i ubaciti ga u izlaznu korpu. Važno je napomenuti da se sve naredbe malom čoveku dostavljaju kao trocifreni decimalni brojevi.

## 1.2 Instrukcioni set

Svaka instrukcija se sastoji od tri cifre. Prva cifra predstavlja operacioni kod, koji definiše koju instrukciju treba izvršiti, dok druga i treća predstavljaju dvocifrenu adresu sandučića, koji je operand ove instrukcije.

Definisane su sledeće instrukcije (Tabela 1.1):

Numerički kod	Simbolički kod	Instrukcija	Opis
1xx	ADD	ADD (sabiranje)	Vrednost sačuvana u sandučiću xx se sabira sa vrednošću u akumulatoru. Sadržaj sandučića nije promenjen i rad akumulatora nije definisan za instrukcije koje dovode do zbira većeg od 999.
2xx	SUB	SUBTRACT (oduzimanje)	Vrednost sačuvana u sandučiću xx se oduzima od vrednosti u akumulatoru. Sadržaj sandučića nije promenjen i rad akumulatora nije definisan za instrukcije koje dovode do negativnih rezultata manjih od -999, ali je stanje negativnog rezultata sačuvano.
3xx	STA	STORE (čuvanje)	Vrednost akumulatora se čuva u sandučiću xx. Sadržaj akumulatora je nepromenjen dok je sadržaj sandučića promenjen.
5xx	LDA	LOAD (učitavanje)	Vrednost sačuvana u sandučiću xx se unosi u akumulator.
6xx	BRA	BRANCH (bezuslovni skok)	Vrednost program counter-a se podešava na adresu xx, tj. sledeća izvršena instrukcija se nalazi u sandučiću xx.
7xx	BRZ	BRANCH IF ZERO (uslovni skok nulta vrednost)	Ako je vrednost akumulatora 0 i stanje kalkulatora nije negativno, program counter se podešava na vrednost xx. U suprotnom se ne dešava ništa.
8xx	BRP	BRANCH IF POSITIVE (uslovni skok pozitivna vrednost)	Ako je vrednost akumulatora pozitivna ili 0 i stanje kalkulatora nije negativno, program counter se podešava na vrednost xx. U suprotnom se ne dešava ništa.
901	INP	INPUT (ulaz)	Učitavanje vrednosti iz ulazne korpe i čuvanje iste u akumulatoru.
902	OUT	OUTPUT (izlaz)	Ispisivanje vrednosti iz akumulatora u izlaznu kutiju.
000	HLT	HALT (zaustavljanje)	Zaustavljanje izvršavanja programa.
	DAT	DATA (definisanje podataka)	Asemblerska instrukcija koja učitava podešenu vrednost u sledeće dostupno sanduče. DAT se može koristiti sa oznakama da bi se deklarise promenjive.

Tabela 1.1: Instrukcije LMC-a

## 1.3 Instrukcioni ciklus

Koraci koje mali čovek obavlja kako bi sproveo instrukciju se naziva instrukcioni ciklus. Ovaj ciklus, koji je sličan za sve instrukcije, se može podeliti na dva dela: dohvaćanje instrukcije i izvršavanje instrukcije.

Prilikom dohvaćanja instrukcije, koje je identično za sve instrukcije, mali čovek saznaje koju instrukciju treba da izvrši. Za početak, mali čovek pamti trenutnu vrednost program counter-a. Posle toga, odlazi na adresu sandučića koja odgovara toj vrednosti i čita trocifreni broj sadržan u njemu. Taj trocifreni broj predstavlja instrukciju koju će izvršiti.

Izvršni deo instrukcije je različit za svaku instrukciju. Njihove pojedinosti su opisane u Tabeli 1.1 Sledi opis izvršavanja instrukcije LOAD:

Mali čovek prvenstveno dohvata instrukciju. Kako bi je izvršio, on odlazi na adresu sandučića koja je operand instrukcije. Dalje, čita trocifreni broj koji se nalazi u tom sandučiću i unosi ga u akumulator. Na kraju, odlazi do program counter-a i inkrementira ga. Ovim je mali čovek obavio jedan instrukcioni ciklus i spreman je da započne sledeći.

## 1.4 Instrukcije grananja

Ove instrukcije kao operand imaju adresu sledeće instrukcije koju je potrebno izvršiti. One

omogućavaju kontrolisanje toka programa, kao i implementaciju skokova i petlji. U arhitekturi LMC-a su implementirane tri instrukcije grananja:

### Bezuslovno grananje (skok) - operacioni kod 6

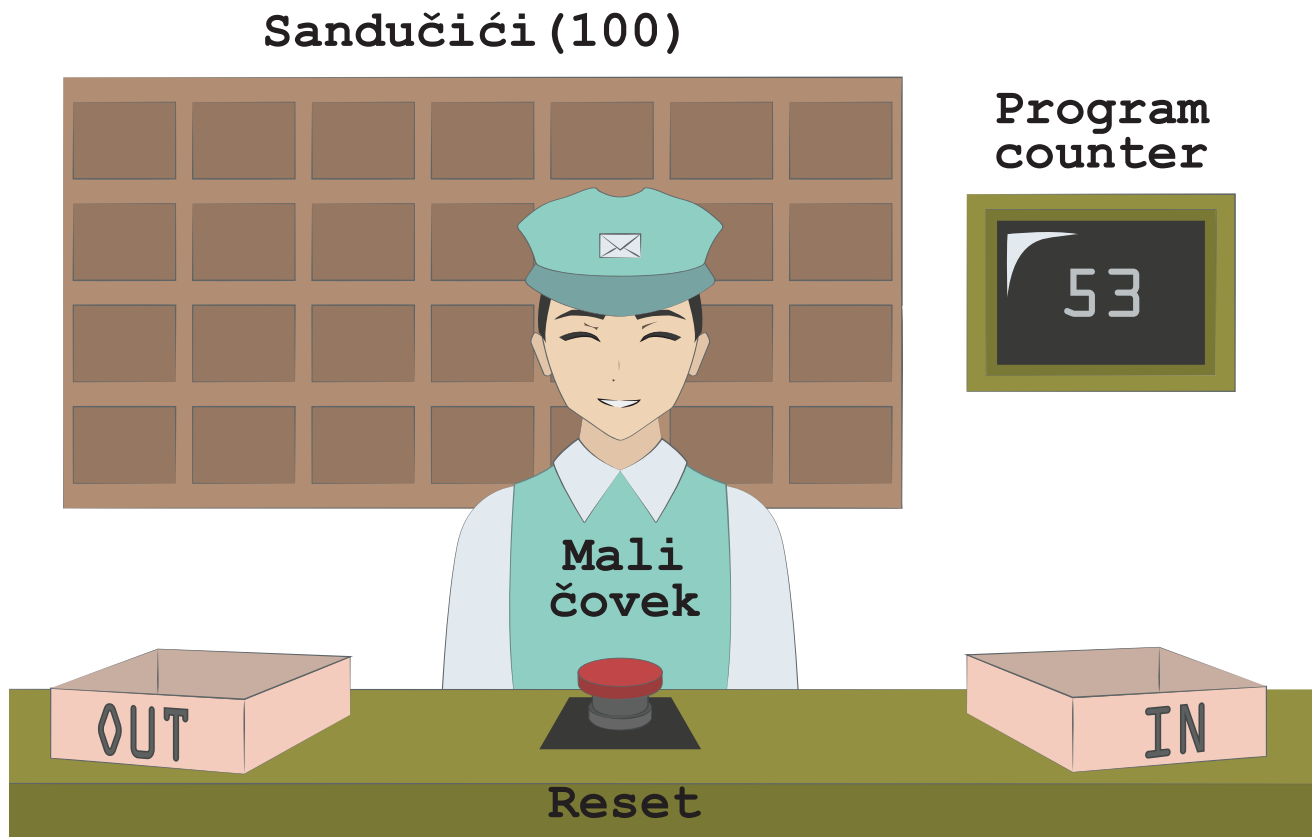
Ova instrukcija govori malom čoveku da ode do program counter-a i da promeni njegovu vrednost na adresu koja predstavlja operand ove instrukcije. To znači da će sledeća izvršena instrukcija biti baš ta koja se nalazi zabeležena kao operand instrukcije grananja.

### Grananje pri vrednosti 0 - operacioni kod 7

Mali čovek će pri izvršavanju ove instrukcije otići do akumulatora i proveriti da li je vrednost jednaka nuli. Ako jeste, otići će do program counter-a i promeniti njegovu vrednost na adresu koja predstavlja operand ove instrukcije. Ako nije, samo će inkrementirati program counter.

### Grananje pri pozitivnoj vrednosti - operacioni kod 8

Mali čovek će pri izvršavanju ove instrukcije otići do akumulatora i proveriti da li je vrednost pozivina (oznaka za negativne brojeve nije aktivna). Ako jeste, otići će do program counter-a i promeniti njegovu vrednost na adresu koja predstavlja operand ove instrukcije. Ako nije, samo će inkrementirati program counter. Nulu smatramo pozivnom vrednošću.



Slika 1.1: Grafički prikaz LMC arhitekture  
Grafiku izradile Bojana Milosavljević i Simonida Srdić

## 2 Primeri programa

### 2.1 Sabiranje dva broja

```
0 901    #učitavanje sa ulaza u akumulator
1 306    #čuvanje sadržaja akumulatora u sandučić 06
2 901    #učitavanje sa ulaza u akumulator
3 106    #sabiranje sadržaja akumulatora sa sadržajem sandučića 06
4 902    #ispis sadržaja akumulatora
5 000    #kraj programa
```

Ovaj program sabira dva broja sa ulaza, sabira ih i ispisuje zbir.

### 2.2 Maksimum dva broja

```
0 901    #učitavanje sa ulaza u akumulator
1 312    #čuvanje sadržaja akumulatora u sandučić 12
2 901    #učitavanje sa ulaza u akumulator
3 313    #čuvanje sadržaja akumulatora u sandučić 13
4 212    #oduzimanje sadržaja sandučića 12 od sadržaja akumulatora
5 809    #grananje na pozitivnoj vrednosti akumulatora (skok na lokaciju 09)
6 512    #učitavanje sadržaja sandučića 12 u akumulator
7 902    #ispis sadržaja akumulatora
8 611    #bezuslovni skok na lokaciju 11
9 513    #učitavanje sadržaja sandučića 13 u akumulator
10 902    #ispis sadržaja akumulatora
11 000    #kraj programa
```

Ovaj program pronalazi maksimum od dva broja sa ulaza i ispisuje ga.

## 2.3 Množenje 2 broja

```
0 901    #učitavanje sa ulaza u akumulator
1 316    #čuvanje sadržaja akumulatora u sandučić 13
2 901    #učitavanje sa ulaza u akumulator
3 317    #čuvanje sadržaja akumulatora u sandučić 17
4 519    #učitavanje sadržaja sandučića 19 u akumulator
5 116    #sabiranje sadržaja akumulatora sa sadržajem sandučića 16
6 319    #čuvanje sadržaja akumulatora u sandučić 19
7 517    #učitavanje sadržaja sandučića 17 u akumulator
8 218    #oduzimanje sadržaja sandučića 18 od sadržaja akumulatora
9 317    #čuvanje sadržaja akumulatora u sandučić 17
10 804   #grananje na pozitivnoj vrednosti akumulatora (skok na lokaciju 04)
11 519   #učitavanje sadržaja sandučića 19 u akumulator
12 216   #oduzimanje sadržaja sandučića 16 od sadržaja akumulatora
13 319   #čuvanje sadržaja akumulatora u sandučić 19
14 902   #ispis sadržaja akumulatora
15 000   #kraj programa
16 0
17 0
18 1     #konstanta 1
19 0
```

Ovaj program množi dva broja sa ulaza i ispisuje proizvod.

## 3 Asemblerski jezik

Asemblerski jezik je programski jezik koji mašinski kod, specifičan za konkretnu procesorsku arhitekturu (u ovom slučaju LMC), predstavlja u ljudima čitljivom obliku. Veza između simboličkih asemblerskih instrukcija i mašinskih je 1-1. Iako LMC koristi mali broj simboličkih kodova, pogodnost korišćenja istog je vidna zato što programer ne mora da pamti neimenovane numeričke kodove. Još jedan vid olakšanja predstavljaju oznake, koje se mogu koristiti kao adrese instrukcija (u slučaju grananja) ili kao promenjive (korišćenjem naredbe DAT).

### 3.1 Simbolički kodovi i oznake

Za početak, dat je jednostavan primer programa čiji je zadatak da sabere dva broja sa ulaza(2.1), koristeći asemblerske oznake i simboličke kodove:

```
0      INP
1      STA broj
2      INP
3      ADD broj
4      OUT
5      HLT
6
7 broj  DAT
```

Ovaj kod se asblira u kod koji je naveden u odeljku 2.1. U ovom slučaju, promenljivoj broj dodeljen je sandučić broj 06, a to je definisano naredbom DAT.

Sledeći primer je program koji ispisuje maksimalni od dva broja sa ulaza(2.2):

```
0      INP
1      STA broj1
2      INP
3      STA broj2
4      SUB broj1
5      BRP poz
6      LDA broj1
7      OUT
8      BRA kraj
9 poz   LDA broj2
```



```

10      OUT
11 kraj  HLT
12
13 broj1  DAT
14 broj2  DAT

```

U ovom programu su pored oznaka za promenljive korišćene i oznake lokacija (poz i kraj). Te oznake se mogu naći ispred bilo koje izvršne naredbe asemblerskog koda.

Poslednji primer je program koji izračunava proizvod dva broja sa ulaza(2.3):

```

0      INP
1      STA broj1
2      INP
3      STA broj2
4 petlja LDA total
5      ADD broj1
6      STA total
7      LDA broj2
8      SUB jedan
9      STA broj2
10     BRP petlja
11     LDA total
12     SUB broj1
13     STA total
14     OUT
15     HLT
16
17 broj1  DAT
18 broj2  DAT
19 jedan  DAT 1
20 total  DAT 0

```

U ovom programu možemo videti primer realizacije for petlje, koja broj2 puta dodaje broj1 na samog sebe.

# 4 Simulator

## 4.1 Uvod

Simulator je implementiran u programskom jeziku Python(3.10). Dostupne funkcionalnosti su prevođenje sa asemblerskog koda na LMC mašinski(učitavanjem koda iz tekstualnog fajla), pronalaženje i signaliziranje grešaka u kodu, kao i izvršavanje i praćenje izvršavanja u realnom vremenu.

Simulator je realizovan uz pomoć dve klase sa različitim kompetencijama, klasom *LittleManComputer* i klasom *ProgramParser*. Prva klasa je odgovorna za izvršavanje programa i prikaz korisničkog interfejsa, dok druga klasa tumači i prevodi asemblerski kod u niz instrukcija koje se čuvaju u memoriji LMC-a.

Celokupan izvorni kod je dostupan na github repozitorijumu: [https://github.com/misadeks/little\\_man\\_computer\\_simulator](https://github.com/misadeks/little_man_computer_simulator)

## 4.2 Rad i uputstvo

Pokretanjem *main.py* fajla, prikazan je sledeći interfejs:

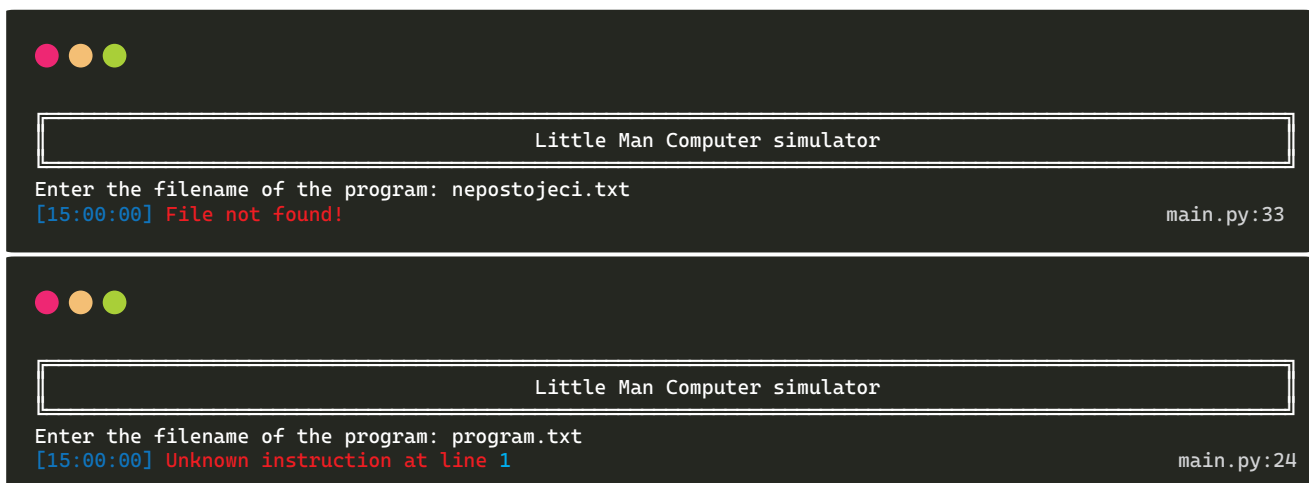


```

    Little Man Computer simulator

Enter the filename of the program: sabiranje_dva_ulaza.txt
Enter the clock speed (instructions/s): 1
```

U slučaju greške u vidu nepostojeće datoteke programa ili greške u prevođenju (nepoznata instrukcija, nepoznata lokacija, nepoznata promenljiva i neoznačena DAT instrukcija), biće prikazane ovakve poruke:



```

    Little Man Computer simulator

Enter the filename of the program: nepostojeci.txt
[15:00:00] File not found! main.py:33

    Little Man Computer simulator

Enter the filename of the program: program.txt
[15:00:00] Unknown instruction at line 1 main.py:24
```

Unosom traženih parametara (datoteka programa i brzine takta instrukcija) program prevodi i priprema izvršavanje programa, čiji je sledeći interfejs:

Program started!

Program	Memory	Parameters	Output																																																																																										
<div> <div>INP</div> <div>STA broj1</div> <div>INP</div> <div>ADD broj1</div> <div>OUT</div> <div>HLT</div> <div>broj1 DAT</div> </div>	<table border="1"> <tr><td>901</td><td>306</td><td>901</td><td>106</td><td>902</td><td>000</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	901	306	901	106	902	000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<div> <div>Accumulator = 0</div> <div>Program Counter = 0</div> <div>Clock = 1.0 ins/s</div> <div>Instruction Name:</div> <div>INPUT</div> </div>	
901	306	901	106	902	000	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				

Input a number:

U ovom slučaju, prikazano je izvršavanje programa datog u odeljku 2.1, sabiranje dva broja sa ulaza. Sledeće slike predstavljaju izvršavanje istog programa instrukciju po instrukciju.

Program started!

Program	Memory	Parameters	Output																																																																																										
<div> <div>INP</div> <div>STA broj1</div> <div>INP</div> <div>ADD broj1</div> <div>OUT</div> <div>HLT</div> <div>broj1 DAT</div> </div>	<table border="1"> <tr><td>901</td><td>306</td><td>901</td><td>106</td><td>902</td><td>000</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	901	306	901	106	902	000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<div> <div>Accumulator = 10</div> <div>Program Counter = 1</div> <div>Clock = 1.0 ins/s</div> <div>Instruction Name:</div> <div>STORE</div> </div>	
901	306	901	106	902	000	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				



Program started!

Program	Memory	Parameters	Output																																																																																										
<div>INP</div> <div>STA broj1</div> <div>INP</div> <div>ADD broj1</div> <div>OUT</div> <div>HLT</div> <div>broj1 DAT</div>	<table><tr><td>901</td><td>306</td><td>901</td><td>106</td><td>902</td><td>000</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	901	306	901	106	902	000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<div>Accumulator = 10</div> <div>Program Counter = 2</div> <div>Clock = 1.0 ins/s</div> <div>Instruction Name:</div> <div>INPUT</div>	
901	306	901	106	902	000	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				

Input a number: 20



Program started!

Program	Memory	Parameters	Output																																																																																										
INP STA broj1 INP ADD broj1 OUT HLT  broj1 DAT	<table><tr><td>901</td><td>306</td><td>901</td><td>106</td><td>902</td><td>000</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	901	306	901	106	902	000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Accumulator = 20 Program Counter = 3 Clock = 1.0 ins/s  Instruction Name: ADD	
901	306	901	106	902	000	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				
0	0	0	0	0	0	0	0	0	0																																																																																				



Program started!

Program	Memory										Parameters	Output
INP STA broj1 INP ADD broj1 OUT HLT  broj1 DAT	901	306	901	106	902	000	0	0	0	0	Accumulator = 30 Program Counter = 4 Clock = 1.0 ins/s  Instruction Name: OUTPUT	
	0	0	0	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0	0	0	0		



Program	Memory										Parameters	Output
INP STA broj1 INP ADD broj1 OUT HLT  broj1 DAT	901	306	901	106	902	000	10	0	0	0	Accumulator = 30 Program Counter = 5 Clock = 1.0 ins/s  Instruction Name: HALT	30
	0	0	0	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0	0	0	0		

Program halted!

Kraj izvršavanja programa je označen porukom *Program halted!*, dok se izlaz programa može videti u desnom odeljku *Output*. Parametri LMC-a se mogu pratiti sve vreme tokom izvršavanja programa u odeljku *Parameters*.

## 5 Zaključak

Arhitektura LMC-a predstavlja uprošćeni model rada svakog računara i predstavlja odličnu osnovu za izučavanje generalne arhitekture računara, kao i uvod u programiranje na asemblerskim jezicima. Uz određene modifikacije i dodatke (proširenje instrukcijskog seta i implementacija višeadresnih instrukcija), LMC može postati napredniji i korisniji kao nastavno sredstvo.

Jednostavnost ovog modela doprinosi tome da skoro svakome može da se približi kako rade moderni računari na zanimljiv i razumljiv način, dok simulator omogućava studentima da svoje programe isprobaju i da se uvere u to kako bi se oni izvršavali.

# Bibliografija

- [1] I. Englander, The architecture of computer hardware, systems software, and networking: An information technology approach, 5th ed. Nashville, TN: John Wiley & Sons, 2014.
- [2] W. Yurcik and H. Osborne, “A crowd of Little Man Computers: visual computer simulator teaching tools,” in Proceeding of the 2001 Winter Simulation Conference (Cat. No.01CH37304), 2002.
- [3] <https://www.101computing.net/LMC/>