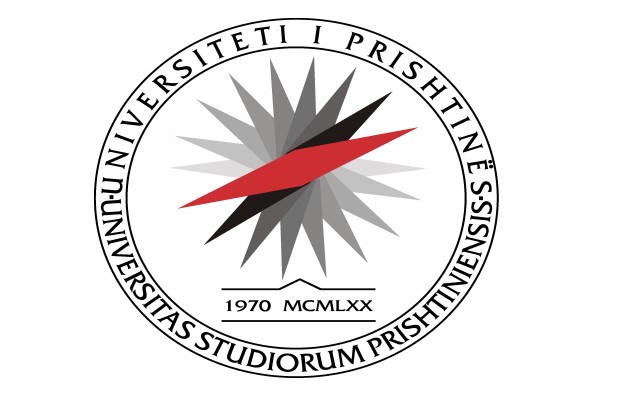
**UNIVERSITETI I PRISHTINËS**

**Fakulteti i Inxhinierisë Elektrike dhe Kompjuterike Departamenti i Kompjuterikës**



**Lënda: Arkitektura e Kompjuterëve**

**16-bit CPU**

**Profesori i lendes:** Valon Raca

Yll Berisha

Maj 2022

# Përmbajtja

1. **Hyrje………………………………………………….…..................................................................................3**
2. **Pjesët për dizajnimin e CPU………………………………………………………………………………………………....4**
3. **Testimet……………………………………………………………………………………………………………………………….5 4. Konkluzioni…………………………………………………………………………………………………………………………..8**

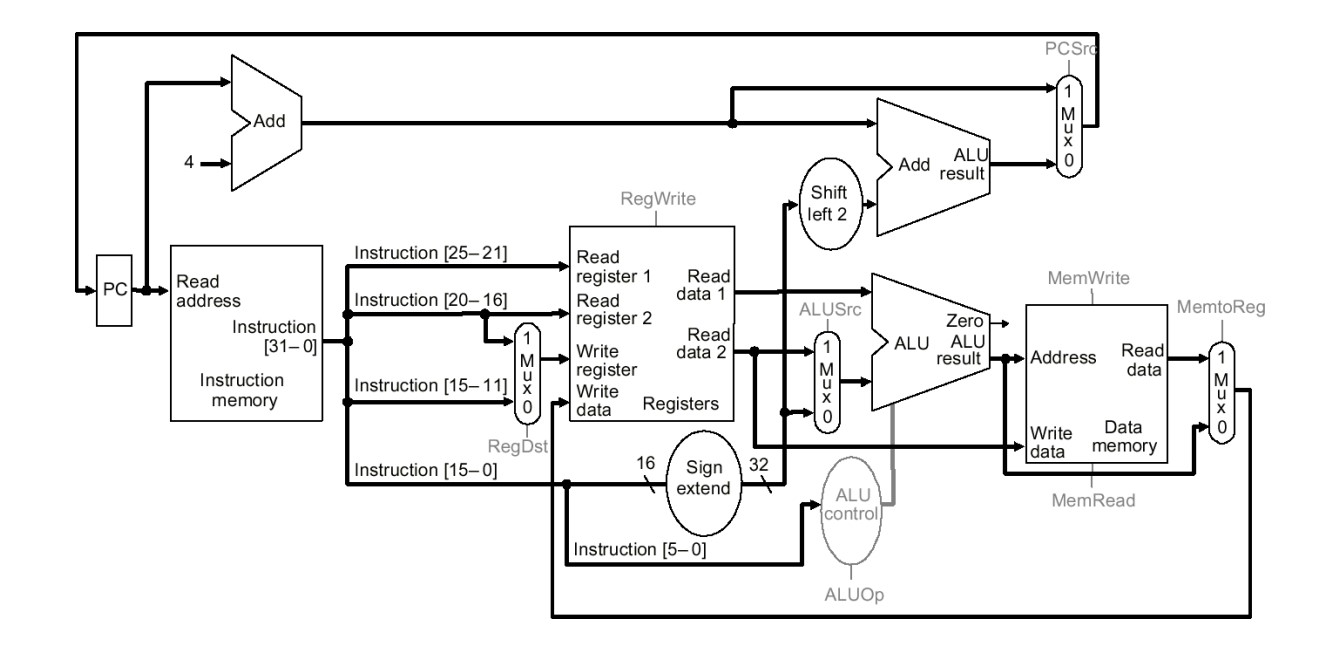
# HYRJE

Qëllimi i detyrës është dizajnimi i CPU-së 16-bitëshe që është e implementuar në Single-Cycle Datapath me anë të Verilog-ut. Një CPU 16-bitëshe nënkupton një procesor me një grup instruksionesh 16-bitësh dhe një databus të dhënash(framework softuerik e përqëndruar te të dhënat për shpërndarjen dhe menaxhimin e tyre në kohë reale në sistemet inteligjente të shpërndara) me gjerësi 16 bit. E kemi paraqitur CPU-ne te ndare ne pjese kryesore:

* Krijimi i Datapath – Këtu kemi paraqitur komponentët e procesorit që përformojnë operacione aritmetike si dhe ruajnë të dhëna.
* Dizajnimi i një Control Unit (CU) - Këtu kemi paraqitur komponentët e procesorit që e komandojnë datapath-in, memorien, pajisjet me input dhe output duke i’u përshtatur instruksioneve të memories.

Elementet e nevojshme per krijimin e CPU 16-biteshe:

* **Datapath** - Datapath është hardueri që i kryen të gjithë operacionet e kërkuara, për shembull, ALU, regjistrat dhe internal buses. Kontrolli është hardueri që i tregon shtegut të të dhënave se çfarë duhet të bëj, për sa i përket ndërrimit, zgjedhjes së funksionimit, lëvizjes së të dhënave midis komponentëve ALU, etj.
* **Control Unit (CU) -** Njësia e kontrollit të njësisë qendrore të përpunimit rregullon dhe integron operacionet e kompjuterit. Përzgjedh dhe merr udhëzime nga memoria kryesore në sekuencën e duhur dhe i interpreton ato në mënyrë që të aktivizojë elementët e tjerë funksionalë të sistemit në momentin e duhur.
* **ALU 16-bitëshe(Arithmetic logic unit)** - Pjesa e kompjuterit që kryen të gjitha llogaritjet aritmetike.
* **ALU Control(Arithmetic logic unit Control)** - Specifikon se çfarë operacioni kryen ALU-ja.
* **Instruction Memory** - Një njësi memorike për të ruajtur udhëzimet e një programi dhe udhëzimet të furnizimit me të dhëna nga një adresë.
* **Program Counter (PC)** - Është një regjistër që mban adresën e instruksioneve të tashme.
* **Register file** – Secili regjistër mund të lexohet ose shkruhet duke e specifikuar numrin e atij regjistri.
* **Data Memory** - Zbaton funksionalitetin për leximin dhe shkrimin e të dhënave tek memoria.
* **Multiplexer** - Është pajisje që zgjedh midis disa sinjaleve hyrëse analoge ose digjitale dhe e përcjell atë në një linjë të vetme daljeje.



# Pjeset per dizajnimin e CPU

**Alu\_16bit –** Përbëhet nga 16 ALU-ja 1bitëshe me anë të metodës Ripple-carry ku cout i njërës ALU është hyrje për ALU-në e ardhshme. Kjo është përgjegjëse për kryerjen e operacioneve si andi, ori, add, addi, sub etj. Ajo përkrah numra me gjatësi 16 bitëshe. Fillimisht e kemi krijuar një ALU 1-bitëshe e cila i përkrah veprimet e lartpërmendura, pastaj janë lidhur 16 ALU-ja 1 bitëshe për kompletim të ALU-së 16bitëshe. Brenda kësaj ALU-je ne kemi një multiplekser 2 me 1 ku ai 8 me 1 është përgjegjës për zgjedhjen e veprimit(pra njërit nga veprimet lartë).

**Alu\_control –** Përmban modulin alu\_control() me hyrjet: ALUOp,Funct kurse dalja është: ALU\_Cnt. Në Alu Control permes ALUOp përcaktojme se cfarë operacione dëshirojmë të kryejmë. Në rastin tonë daljet jane 3bitëshe. ALU control është përgjegjëse që të ja përcaktojë ALU-së 16 bitëshe bitat të cilët e specifikojnë veprimin që ka për tu kryer në ALU. Pra disa prej këtyre bitave ndahen për multiplekserin e specifikuar më lart, disa prej tyre për invertim të hyrjes B.

**CU** – Hyrjet jane: Opcode Daljet: RegDst, MemToReg, MemWrite, Branch, MemRead, RegWrite, ALUOp , ALUSrc. Kjo paraqitet si stateless dhe është pjesë përgjegjëse për të gjitha veprimet brenda CPU-së. Të gjitha fushat e saj përvec ALUop e cila shërben për specifikimin e operacionit që do kryhet në ALU, pjesa tjetër ka vetëm një bit, ngase secila prej tyre është për të treguar vërtetësi psh: MemRead për vlerën 1 lejon të lexojmë në memorie, MemWrite të shkruajmë në memorie, Branch për kërcime me beq, MemtoReg i cili shërben si bit selektues tek multiplekseri 2 me 1 i cili zgjedh se a kemi të bëjme me lw apo me ndonjë operacion tjetër të formatit r, RegWrite për vlerën 1 lejon qe te shkruajme në regjistër edhe ALUsrc i cili është bit selektues tek multiplekseri 2 me 1 i cili shërben për të marë vlerën imediate apo vlerën e read data 2, varësisht nga lloji i instruksionit i apo r.

**Mux\_2ne1** – përmban modulin mux\_2to1() Hyrjet: input0, input1, S Dalja: multi2\_out

**Mux\_8ne1** – përmban modulin mux\_4to1() Hyrjet: input0, input1, input2, input3, input4, input5, S Dalja: multi8\_out. Jane multiplekserat qe na ndihmojne për të zgjedhur operacionet e kerkuara.

**DataMemory** –

Hyrjet: Adresa,WriteData, MemWrite,MemRead,Clock.

Dalja: ReadData. Data Memory është Read-Write(mundeson shkrim dhe lexim). Hyrje 16 bitëshe për përcaktimin e adresës së fjalës 2 bajtëshe që lexohet/shkruhet. Hyrje 16 bitëshe për fjalën që shkruhet. Dalje 16 bitëshe për fjalën që lexohet. Me anë të $readmemh për të inicializuar memorien nga fajlli dataMem.mem që përmban vlera hex të gjitha zero që tregojnë se është e shprazur.

Data Memory është e ngjajshme me instruction memory, përpos se këtu mundemi me shkrujt të dhëna. Shërben për instruksionet e formatit i, dy veprimet që lidhen me data memory janë sw dhe lw.

**InstrMemory** –

Hyrjet: clk,Hyrja Dalja: dalja. Instruction Memory është vetëm Read-Only(vetëm për Instruksione). Hyrje 16 bitëshe nga PC për përcaktimin e adresës së instruksionit. Dalje 16 bitëshe për leximin e instruksionit. InstructionMemory është në heksadecimal dhe është 128bajt, ku dhjetë bajtat e parë janë të rezervuar kurse nga bajti i 10 kemi shkruar instuksionet per ekzekutim nga memoria. Me anë të $readmemh për të inicializuar memorien nga fajlli instrMem.mem që përmban vlera hex

**RegisterFile** –

Hyrjet: RS,RT,RD,WriteData, RegWrite, Clock. Daljet: ReadRS,ReadRD. Numri i regjistrave do të jetë 8 (regjistri $zero dhe 7 të tjerë për përdorim të përgjithshëm). Regjistrat do të jenë të gjerë 16 bit. Register File ka tre hyrje tre bitëshe për përcaktimin e regjistrave RS, RT, RD. Register file ka një hyrje 16 bitëshe për të shkruar në regjistrin RD. Register File ka dy dalje 16 bitëshe për të lexuar të dhënat nga regjistrat e përcaktuar në RS dhe RD. WD - Te dhenat per tu shkruar ne RegisterFile.

- Ka të implementuar dy multiplekserë(8 me 1 pasi kemi 8 regjistra dhe 2 me 1) për zgjedhjen e rs dhe rtsë dhe një dekoder i cili vendos në cilin regjistër me shkrujt, pastaj në atë regjistër vendosen të dhënat që vijnë në writedata.

Mod & SLTI & XOR &ADDI

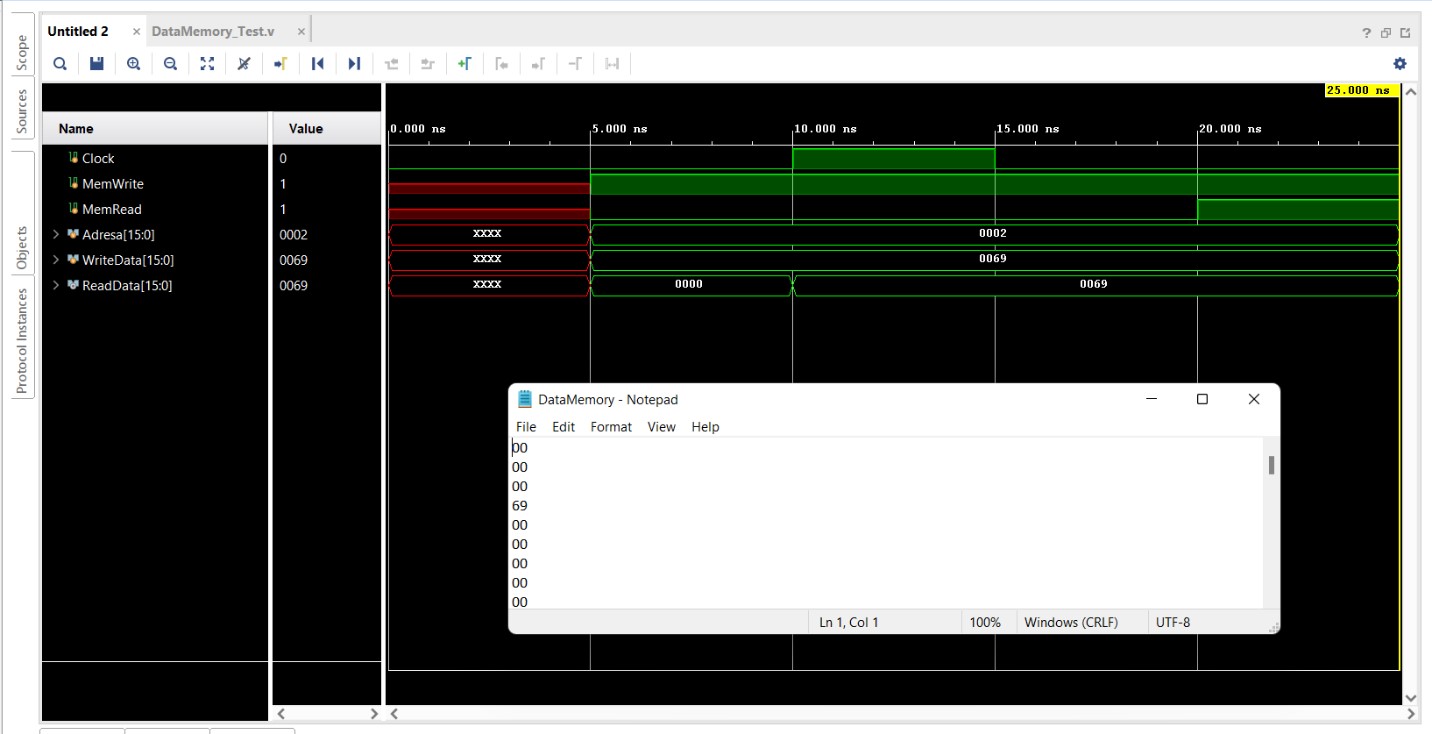
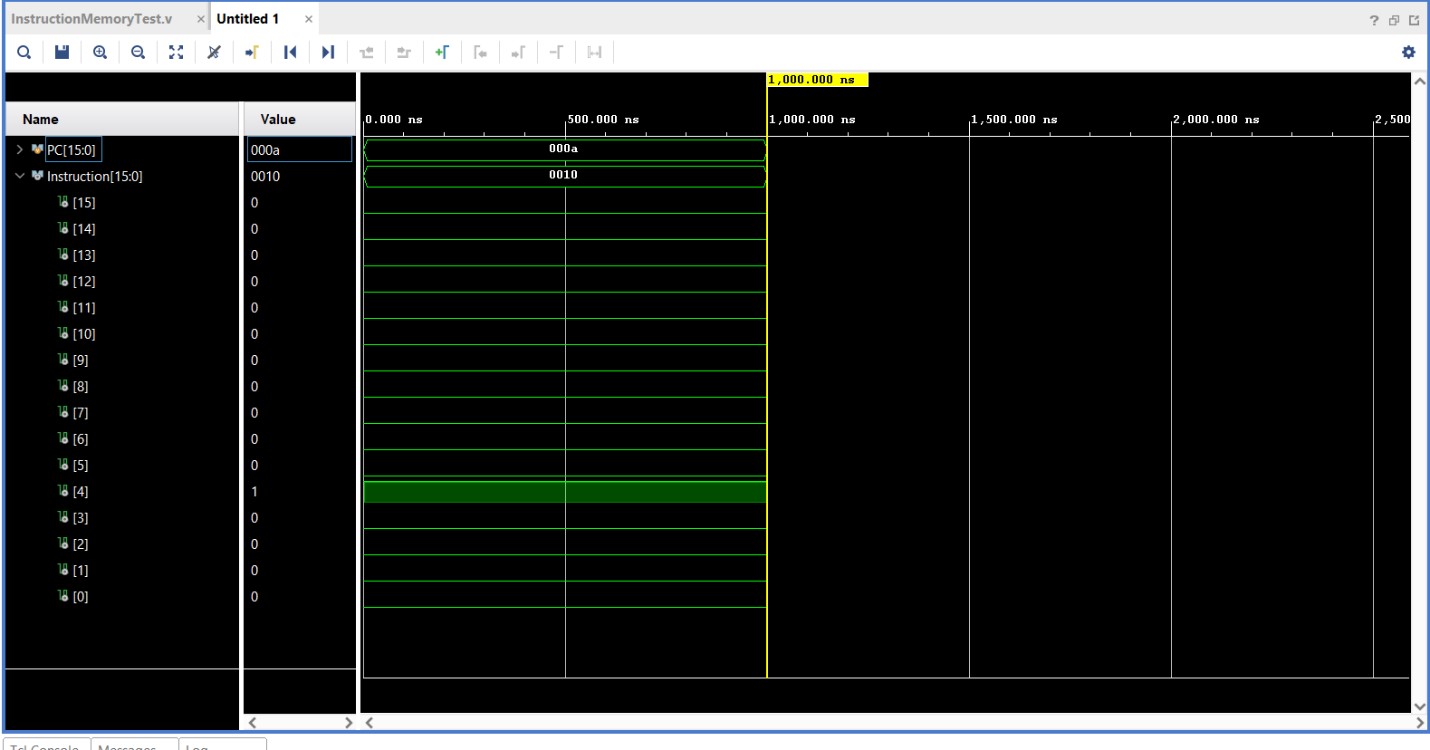
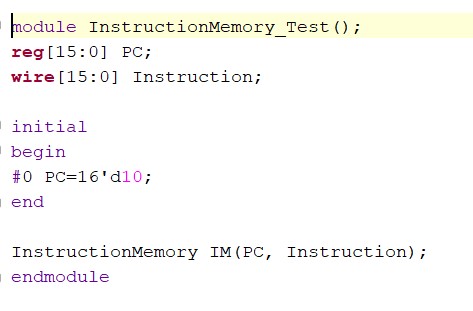
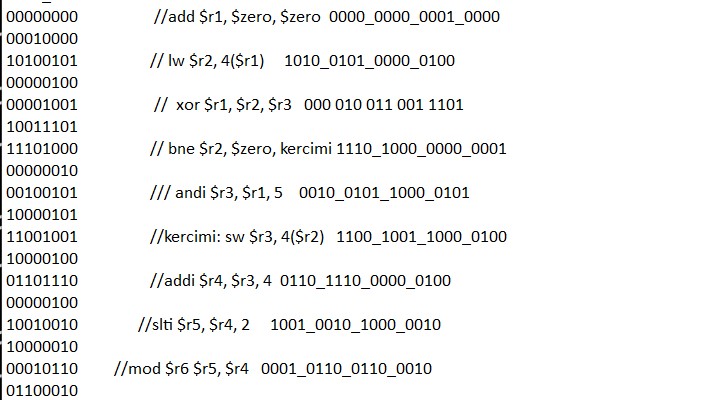
Si kerkese e pjeses se detyrave bonus po ashtu kemi realizuar Instruksionet

* Mod -eshte thirrur ne ALU si funksion I jashtem ku mod eshte krijuar me ane te nje vlere dhe divisor-it te tij ku si dalje na jepet mbetja.
* SLTI-(SET LESS THAN EQUAL) edhe ky instruksion eshte krijuar nga jashte dhe eshte thirrur Brenda ALU.
* XOR-exclusive or – eshte operator logjik Boolean-Jep True nëse saktësisht një (por jo të dyja) nga dy kushtet është true.
* ADDI realizohet te mbledhesi njejte si add por me ane te mux2ne1 qe paraqitet para saj ne skeme aty vendoset a do te merret si vlere fillestare vlera immediate a po jo.

# Ekzekutimi

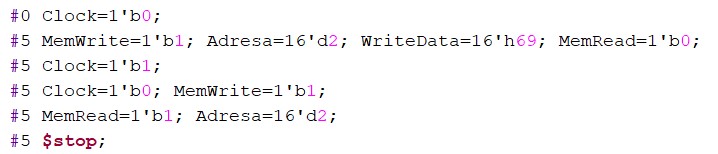
**Instruction Memory Test:**

Në vijim është paraqitur ekzekutimi i testimit te Instruction Memory, ku kemi marrë instruksionet prej adresës 11 pasi që 10 të parat janë të rezervuara.



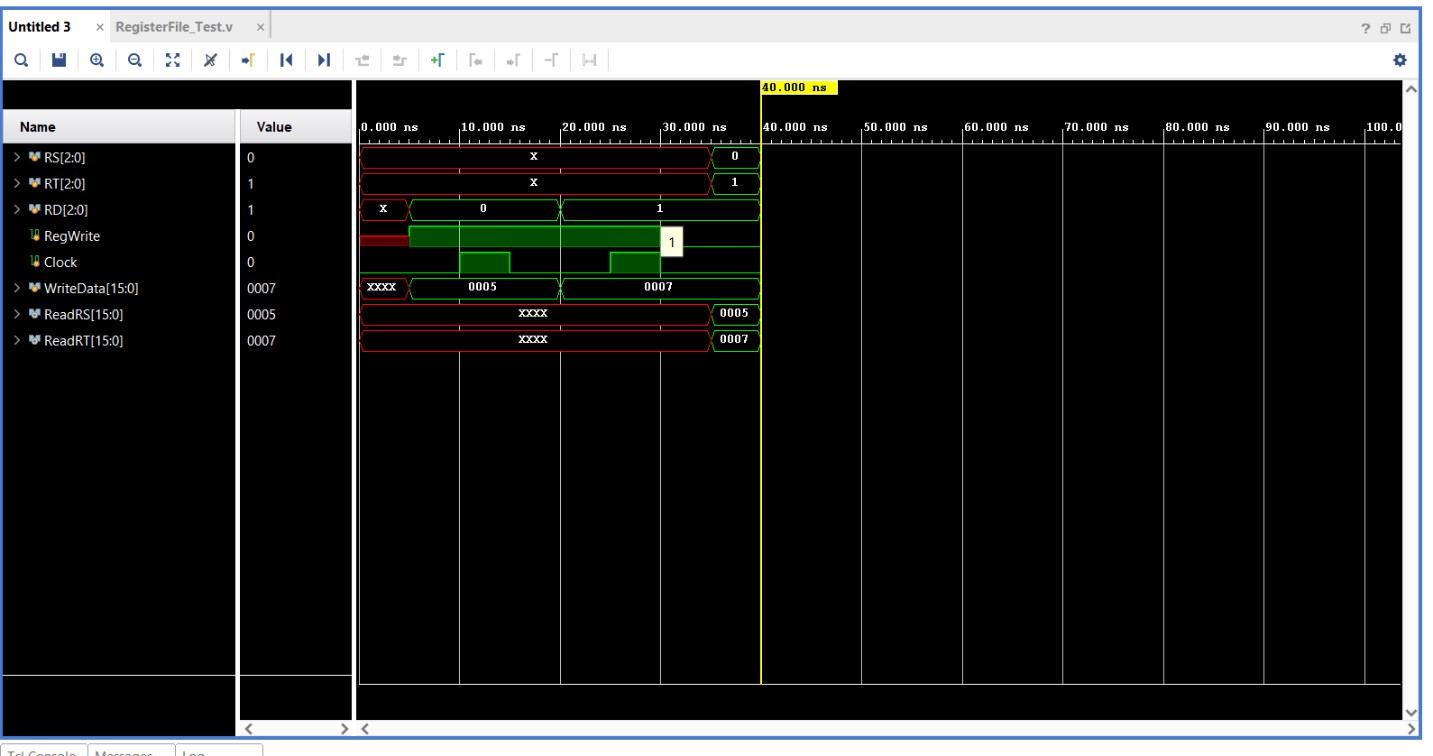
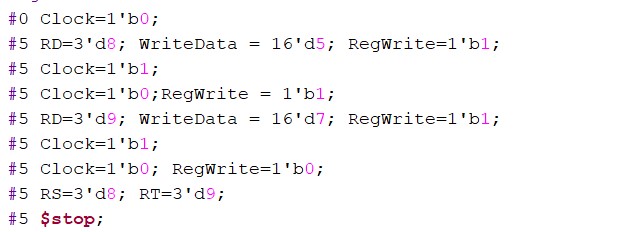
**Data Memory Test:**

Testimin e Data Memory e kemi bërë duke shënuar një numër në memorie, dhe pas ekzekutimit shihet që numri jonë është ruajtur në memorie.



**Register File Test:**

Tek Register File për testim kemi shënuar vlerat 5 dhe 7 , verejmë se kur dëshirojm të shkruajmë ndonjë vlerë RegWrite duhet të jetë 1, dhe në fund i kemi lexuar këto vlera, me poshtë janë paraqitur edhe Screenshots që e ilustrojnë këtë.



# Konkluzioni

Lloji më i rëndësishëm i dizajnit të harduerit është arkitektura e CPU-së të kompjuterit.

Dizajni i procesorit përcakton se çfarë programi mund të funksionoj në kompjuter dhe cilët komponentë të tjerë harduerikë janë të suportuar. Nga përvoja me këtë detyrë, mund të themi se ka qenë një sfidë e cila na ka ndihmuar shumë për ta kuptuar aspektin praktik të një procesori që punon sipas parimit të Single Cycle.