*TEHNIČKA ŠKOLA RUĐERA BOŠKOVIĆA ZAGREB*

GRAĐA RAČUNALA – Logisim

LV10 – ponavljanje

1. Projektirati memorijski modul ROM s 1024 memorijskih riječi duljine 8 bita. Koliki je kapacitet tog memorijskog modula? Upišite podataka na prve 4 memorijske lokacije, i simulirajte iščitavanje promjenom adrese.
   * Pin na kojem mijenjamo adrese (na ulazu A ROM modula) moramo postaviti na 10 bita (*Data Bits)*. Isto to učinimo za ROM modul (*Address Bit Width*) jer moramo postići kapacitet od 1 kB.
   * ROM modulu treba postaviti *Data Bits* na 8 bita (duljina

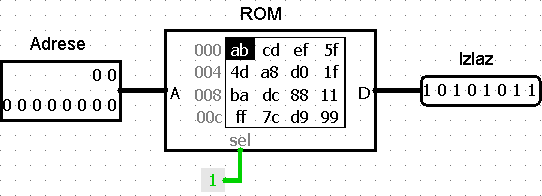
memorijskih riječi), te također pin Izlaz namjestimo na 8 bita

(obavezno uključiti na pinu Izlaz *Output?).*

* + Pomoću desnog klika na ROM modul odlazimo na *Edit Contents...*

te u HEX editoru upišemo sadržaj u ROM.

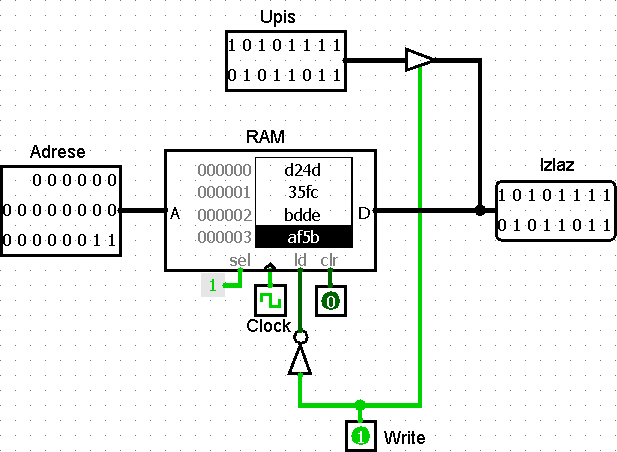
* + Zatim pomoću *Poke Tool-a* mijenjamo adrese na pinu A te kako ih mijenjamo na izlazu dobivamo binarnu kombinaciju upisane heksadecimalne vrijednosti na toj adresi.
  + Kapacitet ovog ROM memorijskog modula je 1kB jer je 210=1024 B.



1. Projektirati memorijski modul RAM s 4MB memorijskih riječi duljine 16 bita. Koliki je kapacitet ovog memorijskog modula? Upiši sadržaj na prve 4 memorijske lokacije, i simulirajte iščitavanje promjenom adrese.
   * Ponavljamo postupak iz prethodnog zadatka, samo što postavljamo duljinu memorijskih riječi na 16 bita, a 4MB realiziramo s 22 bita jer je 222=4.194.304 (približno 4MB).
   * Sada je kapacitet RAM memorije 33.554.432 bita, odnosno

4.194.304 byte-a (33.554.432/8=4.194.304 jer 1 Byte ima 8 bitova).

* + Sklop koji se nalazi na izlazu iz pina *Upis* je Controlled Buffer (16 bitni)
  + Vanjski upis podataka omogućujemo pomoću pina *Write* (ako je u ''1'' upis je omogućen, te se upisuje kada se vrijednost *Clock-a* dosegne ''1'', a kada je Write u ''0'' upis je onemogućen). Podatke upisujemo u pin *Upis*, a mijenjanjem *Adrese* pomoću *Poke Tool-* aodabiremo na koju memorijsku lokaciju želimo upisati novi sadržaj. Na pinu *Izlaz* dobivamo binarnu kombinaciju sadržaja na odabranoj adresi. Istovremeno pin *Sel* mora biti u ''1'', a *Clr* u ''0''.



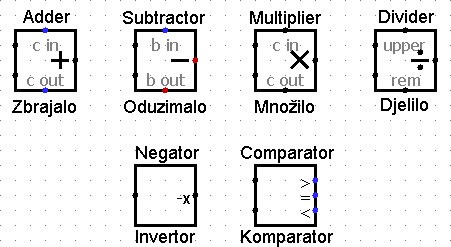
1. Pokrenuti program Logisim i otvoriti biblioteku s aritmetičkim

modulima. Ispisati sve aritmetičke elemente koje sadrži ta biblioteka,

nacrtati simbole i njihove pinove.

* + Iz biblioteke *Arithmetic* uzimamo aritmetičke module: zbrajalo,

oduzimalo, množilo, djelilo, invertor i komparator.

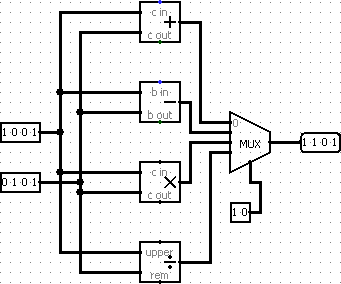


1. Postaviti na radnu površinu module za osnovne aritmetičke operacije i

proanalizirati njihove ulaze i izlaz. Povezati sva četiri modula u jednostavnu aritmetičku sekciju procesora i izlaz odabrati s odgovarajućim kombinacijskim modulom.

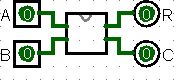
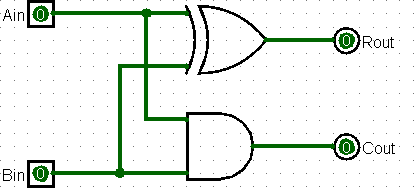
Potrebno je postaviti module koji obavljaju osnovne aritmetičke

operacije.

* + Svakom od tih modula u tablici atributa treba postaviti *Data Bits* na 4 bita. Multipleksoru treba postaviti *Select Bits* na 2 bita, a *Data Bits* na 4 bita. Pinove također postavljamo na 4 bita te sve spojimo kako je navedeno na slici ispod.
  + Pin koji je postavljen na 2 bita služi za biranje sklopa tj. operacije koju želimo izvesti pa je tako: 00-zbrajanje, 01-oduzimanje, 10- množenje, 11-dijeljenje. Na izlazu čitamo rezultat u binarnom zapisu. Na dva ulaza upisujemo binarnu kombinaciju brojeva čije vrijednosti želimo izračunati, no kako imamo skopove s 4 bita, nije moguće izračunati brojeve veće od 15.

1. Nacrtati poluzbrajalo i pomoću njega izvesti podsklop. Pomoću

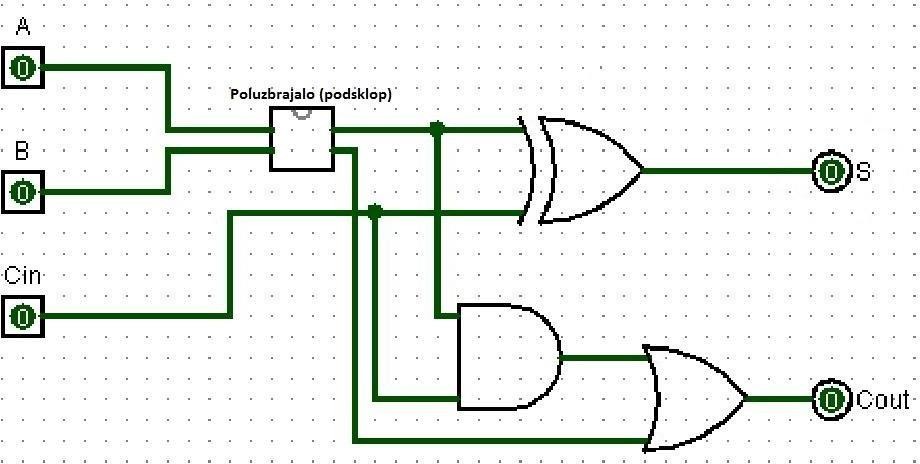
podsklopa poluzbrajala izvesti potpuno zbrajalo koje također pretvorite u podsklop. Pomoću podsklopa potpuno zbrajalo projektirati 8-bitno paralelno zbrajalo i simulirajte njegov rad.



* + Kako bi kreirali podsklop moramo dodati Circuit - ProjectAdd Circuit... kojeg nazovemo Poluzbrajalo. Kada smo dodali Circuit počinjemo slagati sklopove koji se nalaze na slici iznad. Potrebno je namjestiti logičke sklopove na 2 ulaza, pinove na 1 bit, a pinovima Rout i Count u tablici atributa omogućiti *Output.*
  + Sada je potrebno napraviti ponovno novi Circuit kojeg nazovemo Potpuno zbrajalo. Kada smo upisali ime, radna površina programa je prazna. U lijevom izborniku možemo vidjeti povećalo na podsklopu kojeg smo kreirali, *Potpuno zbrajalo*. Ako želimo ponovno uređivati prethodni sklop, kliknemo na njega u lijevom izborniku, i povećalo će se premjestiti na njega, a na radnoj površini programa će se pojaviti sklop. Osnovni sklop kojeg smo prethodno kreirali možemo izvući na radnu površinu i s njim raditi u izradi sljedećeg sklopa tako da kliknemo na njega i izvučemo ga radnu površinu (svrha podsklopova  omogućuju nam da

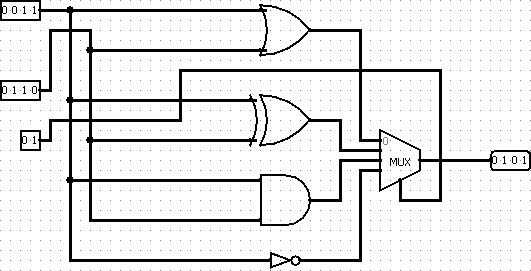
smanjimo broj elemenata na radnoj površini te da lakše spojimo na više mjesta isti sklop).

* + Možemo primijetiti da sklop više nije u onom obliku u kojem smo ga kreirali, već izgleda kao jedan sklop koji obavlja funkciju koju smo mu namijenili s ulazima i izlazima koje smo definirali pri njegovoj izradi.

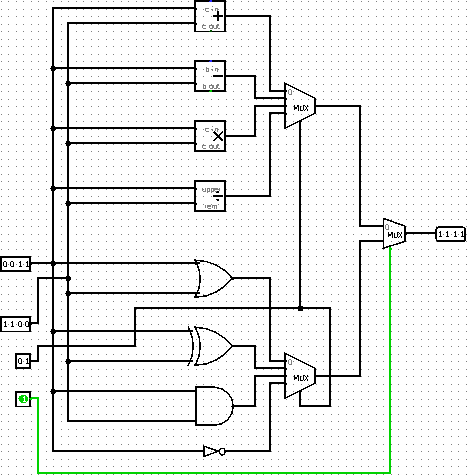


1. Projektirati logičku sekciju ALU pomoću osnovnih logičkih sklopova.

Simulirati njen rad.

* + Ponovimo postupak spajanja iz 4. zadatka. Umjesto aritmetičkih sklopova stavljamo logičke sklopove od kojih I, ILI i isključivo ILI imaju dva ulaza, a NE sklop standarno 1 ulaz. Svakom sklopu postavimo *Data Bits* na 4 bita.
  + Pin koji je postavljen na 2 bita služi za biranje sklopa tj. operacije koju želimo izvesti pa je tako: 00-log.funkcija ILI (OR), 01- log.funkcija isključivo ILI (XOR), 10-log.funkcija I (AND), 11- log.funkcija NE (NOT).

1. Povezati aritmetičku i logičku sekciju u jedan modul ALU i to projektirati kao podsklop. Modul treba izračunavati četverobitne podatke.
   * Sada je potrebno spojiti aritmetičku i logičku sekciju kako bi dobili ALU (Aritmetičko-logičku jedinicu). No, kako bi zajedno funkcionirali potrebno je dodati još jedan Multipleksor na izlazima te dvije sekcije.
   * Potrebno je u tablici atributa za MUX namjestiti *Select Bits* na 1 bit, a *Data Bits* na 4 bita čime omogućavamo prikaz 4 bitnog podatka na izlazu (i naravno ulaz 4 bitnog podatka na ulaz sklopa).
   * Kreiranje podsklopa je objašnjeno u 5. zadatku.



1. Napravite shemu s dva 2-bitna ulaza i 2-bitnim sklopovima za zbrajanje i oduzimanje. Oba ulaza su spojena i na zbrajalo i oduzimalo. Ukoliko zbrajalo šalje jedinicu kao bit prijenosa, neka svijetli zeleni LED, a ukoliko oduzimalo ima bit prijenosa, neka crveni LED svijetli.
2. Napravite shemu s tri 1-bitna ulaza: A, B i C koji daje izlaz po pravilu:

**( NOT A or B ) and C** .

1. Napravite sklop s jednim 8-bitnim ulazom i *splitterom* ga podijelite na dva 4-bitna signala i spojite ih na *i*-sklop. Ta dva 4-bitna ulaza i jedan 4-bitni izlaz spojite na *3 X 4 LED Matrix.* Sklop treba raditi dok je gumb stisnut, inače ni jedan LED ne treba svijetliti.
2. Napravite sklop sa osam LED elemenata različitih boja prema sljedećoj tablici

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ulaz | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| boja | bijela | ljubičasta | plava | zelena | žuta | narančasta | crvena | crna |

Dodajte tri 1-bitna ulaza koji zajedno predstavljaju brojeve od 0 do 7 i prema gornjoj tablici uključuju samo jedan LED, ali samo na rastući brid *clock*-a.

1. Napravite sklop s RAM sklopom kapaciteta 4MB s razdvojenim pinovima za pohranjivanje i čitanje podataka, pri čemu jedna memorijska lokacija zauzima 4 bajta. Dodajte gumb koji prazni memoriju ako se klikne. Dodajte i izlazni pin za prikaz odabranog podatka iz memorije.
2. U dokumentu *8051 Instruction Set* pronađite operacijske kodove za pribrajanje u akumulator iz registara. Dodajte 8-bitni ulaz kojim ćete „upisivati“ operacijski kod jedne naredbe i RAM na koju ćete na nultu adresu dodati tu naredbu. Dodajte i 9 registara koji predstavljaju akumulator i registre opće namjene R0 do R7. Za registre R0 do R7 dodajte također po jedan 8-bitni ulaz koji sprema vrijednost u njih. Napravite sklop koji će prepoznati radi li se o naredbi za zbrajanje. Ako se ne radi o naredbi za zbrajanje neka uključi crveni LED, a ukoliko se radi o naredbi za zbrajanje, neka se rezultat zbrajanja prikaže na 7-segmentnom displeju.