Zadaća 1

Logički dizajn

Student: Bakir Činjarević, bcinjarevi1@etf.unsa.ba

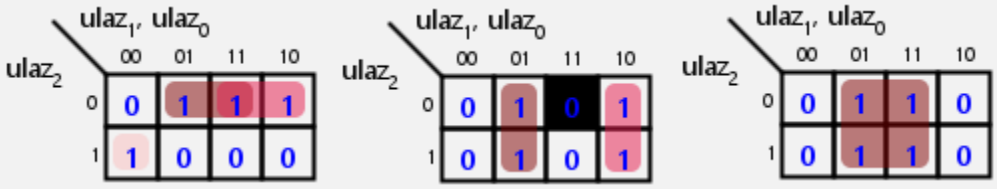
Zadatak 1: 8kk

Projektovati u Logisimu logičko kolo koje na ulazu prima oktalnu cifru, a na izlazu daje njen 8-komplement, koristeći minimalan broj osnovnih logičkih kola.

Oktalna cifra je definisana sa $\log_2 8 = 3$ bita, kao i njen komplement. Tablica istine izgleda ovako:

$x_2x_1x_0$	$y2$	$y1$	$y0$
000	0	0	0
001	1	1	1
010	1	1	0
011	1	0	1
100	1	0	0
101	0	1	1
110	0	1	0
111	0	0	1

K-mape za bitove y_2, y_1 i y_0, redom, su slijedeće:



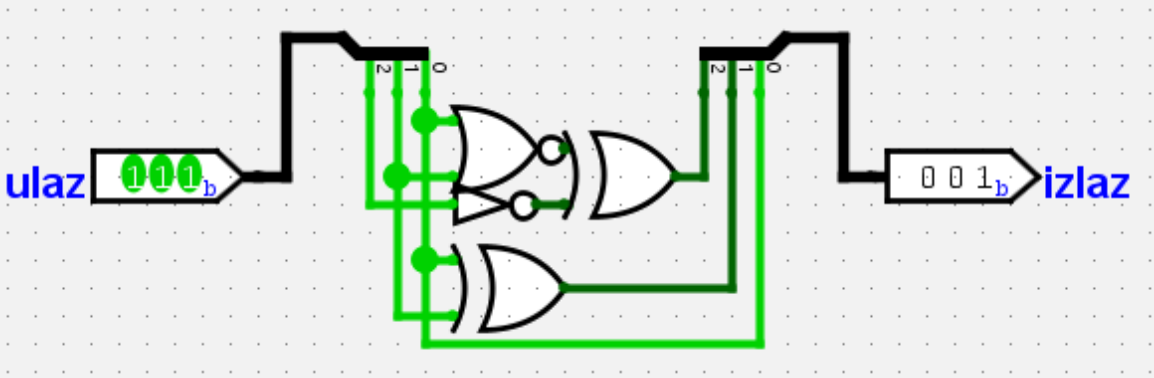
Te dobijamo izraze

1. $y_2 = \overline{x_2}x_0 + \overline{x_2}x_1 + x_2\overline{x_1x_0}$,
2. $y_1 = \overline{x_1}x_0 + \overline{x_0}x_1$,
3. $y_0 = x_0$

Međutim, možemo primijetiti da za k-mapu izlaza y_2 možemo prekriti XOR kolom za konture kao izraz $\overline{x_2} \oplus \overline{x_1}\,\overline{x_0}$. Dvostrukom negacijom izraza $\overline{x_1}\,\overline{x_0}$ dobijemo $\overline{\overline{x_1}\,\overline{x_0}} = \overline{x_1 \vee x_0} = x_1 \downarrow x_0$. Dakle, konačni izrazi za izlaz y su slijedeći:

1. $y_2 = \overline{x_2} \oplus (x_1 \downarrow x_0)$,
2. $y_1 = x_0 \oplus x_1$,
3. $y_0 = x_0$

Na kraju, kolo u Logisimu izgleda ovako:

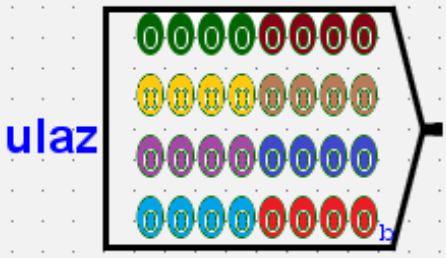


Ako smatramo kola XOR i NOR osnovnim, onda se broj kola može svesti na 4. XOR kolo u sebi krije dvije negacije, dva AND kola i jedan OR, a NOR kolo krije jedan OR i jednu negaciju; ako bismo razbili sklop samo na kola AND, OR i NOT, onda bi ukupan broj kola iznosio 11.

Zadatak 2: 32u1 multiplekser preko 4u1 multipleksera

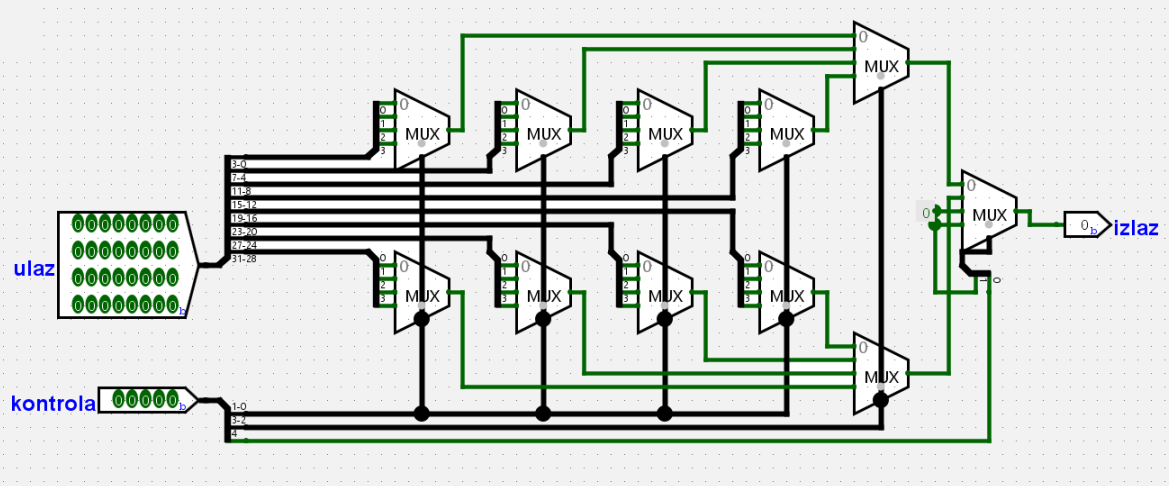
Projektovati multiplekser 32u1 koristeći multipleksere 4u1.

Tablica istine za 32 ulaza nije praktična. Logika kojom sam rješavao zadatak je pomoću slijedeće slike:



Kontrolnih signala je $\log_2 32 = 5$, obilježićemo ih sa $k_4k_3k_2k_1k_0$. Najniža dva kontrolna signala k_1k_0 diktiraju koji bit unutar neke četverobitne grupe se proslijeđuje kroz multiplekser. k_3k_2 određuju koja je od tih četverobitnih grupa unutar jedne polovice ulaza u pitanju, jer smo ograničeni na 4u1 multipleksere; inače bi $k_4k_3k_2$ zajedno određivali koju četverobitnu grupu proslijediti. Nakon što su k_3k_2 odabrali jednu od 4 četverobitne grupe, preostalo je bitu k_4 da odabere s koje polovice ulaza uzima grupu, tj. bira između više ili niže polovice.

Kolo izgleda ovako:



Na zadnjem multiplekseru, situacija se mogla riješiti jednim 2u1 multiplekserom. Zbog ograničenja zadatka, stavljen je 4u1 multiplekser sa viška bitima postavljenim na nulu, jer se nikada neće koristiti.

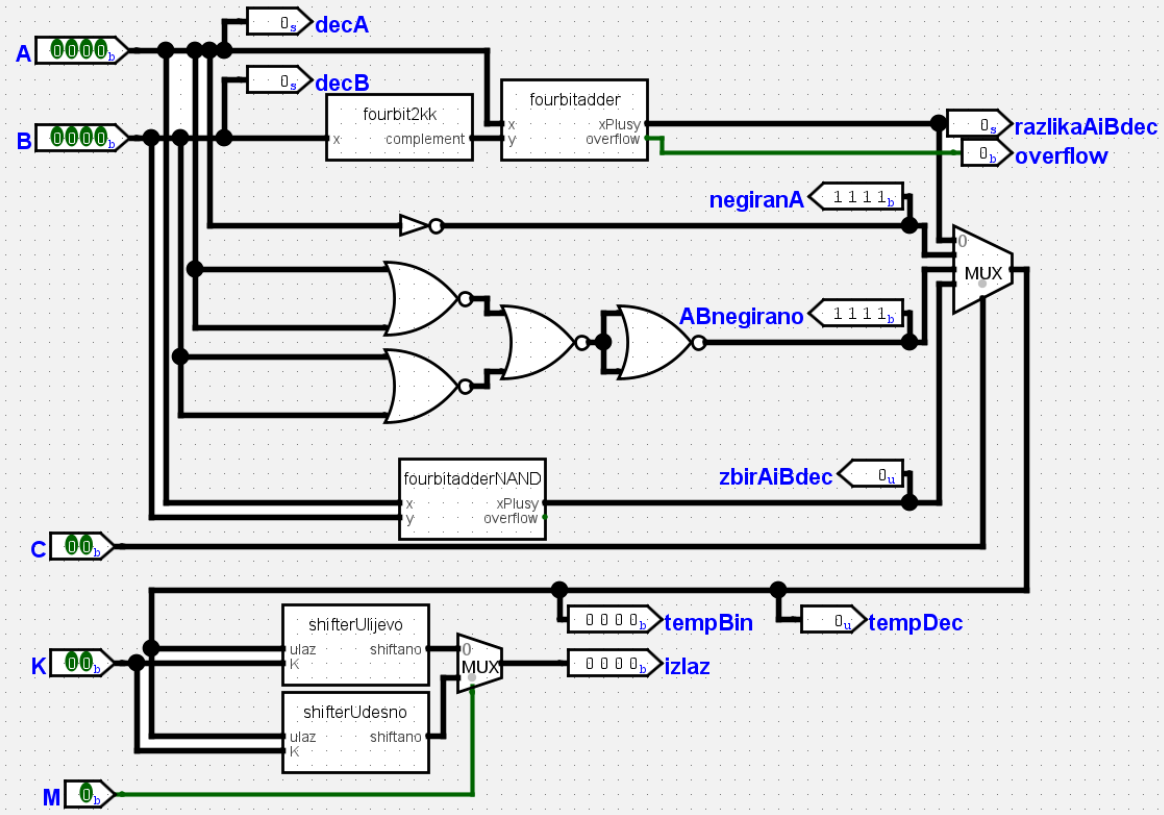
Zadatak 3: ALU

Projektovati u Logisimu 4-bitnu aritmetičko-logičku jedinicu (ALU) kao prostornu iterativnu strukturu. Signal C upravlja radom ALU-a, pri čemu ALU izvršava sljedeće operacije nad dva ulaza A i B:

1. *Ako je vrijednost kontrolnog signala C=0, ALU radi aritmetičko oduzimanje.*
2. *Ako je vrijednost kontrolnog signala C=1, ALU radi logičku negaciju prvog ulaza.*
3. *Ako je vrijednost kontrolnog signala C=2, ALU radi logičku NI operaciju: (AB)', korištenjem minimalnog broja NILI kola.*
4. *Ako je vrijednost kontrolnog signala C=3, ALU radi aritmetičko sabiranje korištenjem minimalnog broja NI kola.*

Izlaz iz ALU-a ide u shifter koji radi pomjeranje lijevo ili desno. Signal M upravlja stranom u koju ide pomjeranje, a signal K specificira za koliko mjesta se vrši operacija.

Očigledno je da operaciju bira kontrolni signal C sa ciframa $c_1 c_0$, što znači da se izbor izlaza može simulirati multiplekserom sa 4 četverobitna ulaza i izlazom. Na visokom nivou, sa pomoćnim prikazima brojeva, kolo izgleda ovako:



Pristupi pri operacijama:

1. $C = 0$: Aritmetičko oduzimanje se vrši sabiranjem A sa komplementom B, zajedno sa izlazom *overflow*, da znamo da li je tačan rezultat operacije.

2. $C = 1$: Logička negacija je u ovom slučaju trivijalna; ulaz A se proslijedi invertoru koji to za nas uradi na četverobitnom nivou.

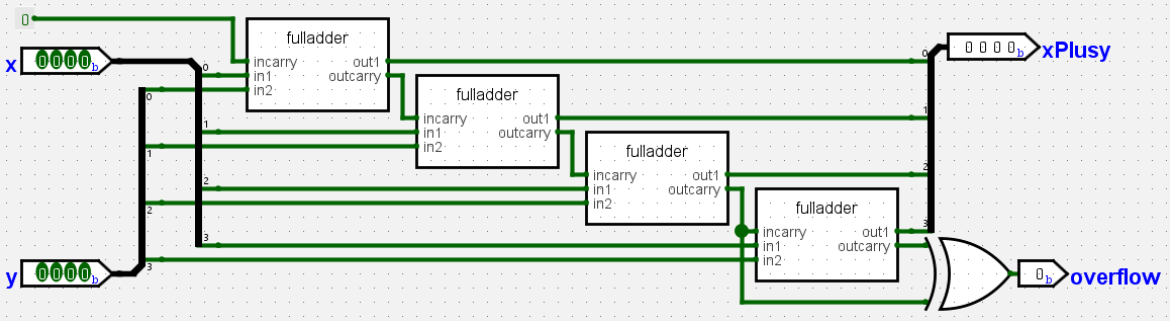
3. $C = 2$: Računanje izraza $\overline{AB}$ se može zapisati kao $\overline{AB} = \overline{A} \vee \overline{B} = \overline{(A \downarrow A) \downarrow (B \downarrow B)} = [(A \downarrow A) \downarrow (B \downarrow B)] \downarrow [(A \downarrow A) \downarrow (B \downarrow B)]$,

4. $C = 3$: Aritmetičko sabiranje preko NI kola je realizovano sa 4 puna sabirača iterativno nadovezanih.

Signal M bira hoće li se izlaz shiftat ulijevo odnosno udesno, što se opet svodi na odgovornost jednog multipleksera. U nastavku ćemo redom opisati komponente.

0.1 fourbitadder

Kolo je izvedeno preko punih sabirača izvedenih onako kao na predavanjima. Iterativno su povezani, i daju *overflow* signal u slučaju prekoračenja, kako bi se signalizirao netačan rezultat. U Logisimu:

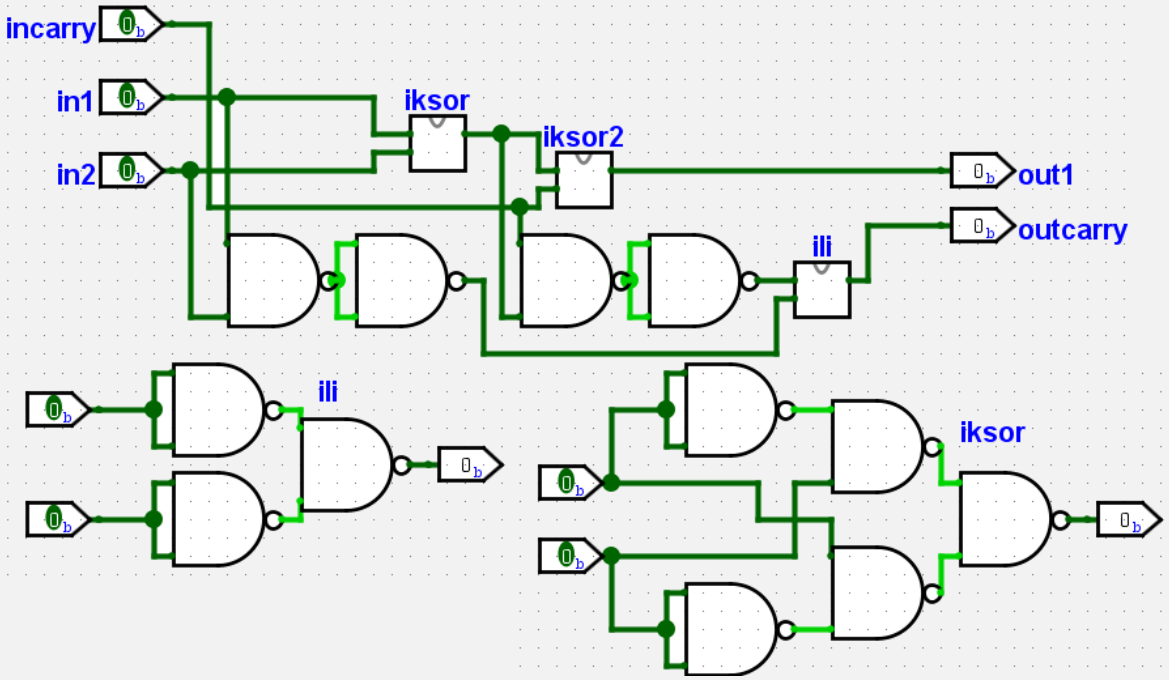


0.2 fourbit2kk

Realizovan kao aritmetičko sabiranje negacije ulaza sa jedinicom pomoću *fourbitadder* komponente.

0.3 fourbitadderNAND

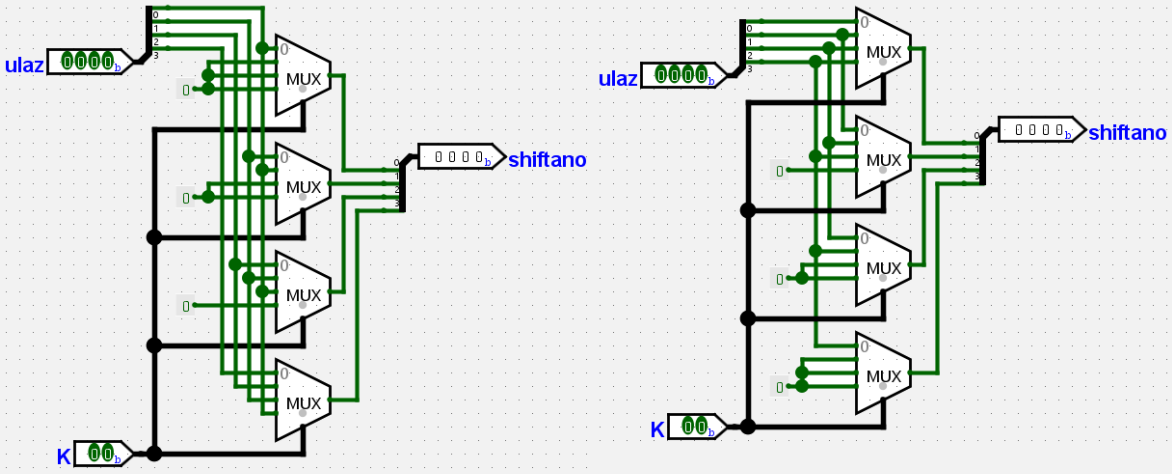
Realizovan na isti način kao *fourbitadder*, s tim što su striktno korištena NAND kola:



Na ovom crtežu vidimo implementaciju punog sabirača, OR i XOR kola preko NAND kola, i njihovu upotrebu u punom sabiraču. Komponenta *fourbitadderNAND* je realizovana na isti način kao *fourbitadder*, samo što se služi NAND punim sabiračima.

0.4 shifterUlijevo i shifterUdesno

Na slici možemo vidjeti komponente *shifterUlijevo* (lijevo) i *shifterUdesno* (desno).



U krajnjem sklopu, na osnovu signala M se multiplekserom bira koji shiftani izlaz će biti proslijeđen.