TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Arvutitehnika instituut

Marvin Uku 120893 IASB

4-JÄRGULISE KOLMENDLOOGIKA LIITJA-LAHUTAJA REALISATSIOON FPGA-L

Bakalaureusetöö

Juhendaja: Peeter Ellervee
Professor / Ph.D

Tallinn 2017

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Marvin Uku

17.05.2017

Annotatsioon

Lõputöö eesmärgiks oli 4-järgulise kolmendloogika liitja-lahutaja realisatsioon fpga-l. Selle töö eesmärgiks oli luua kolmendsüsteemi liitja-lahutaja, mis suudaks liita nelja järguliseid kolmendsüsteemi arve. Samuti pidi liitja-lahutaja töötama Nexys-3 plaadil, andes lülititega sisendi ja võimaldama lugeda väljundit LED-idelt. Töö käigus töötati välja erinevad lülitid, mida kasutati liitja-lahutaja jaoks loogika loomiseks. Lülitite loomiseks on kasutatud töös välja toodud matemaatilisi valemeid. Lõputöös on kasutatud ModelSIM, Xilnx ISE ja Digilent Adapt tarkvara, vastavas järjekorras elementide kirjeldamiseks/simuleerimiseks, skeemi joonistamiseks ja simuleerimiseks, Nexys-3 treeningplaadi programmeerimiseks.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 83 leheküljel, 7 peatükki, 56 joonist, 17 tabelit.

Abstract

FPGA Implementation of 4-trit Ternary Logic Adder-Subtracter

The goal of this thesis was to create a 4-trit ternary logic adder and subtractor, also to implement it on FPGA. The program adds or subtracts 4-trit values using registry, switches and leds on the Nexsys-3 training board. Writing this thesis needed the creation of three-valued-logic operands. Expressions were written in VHDL using ModelSIM and simulated using Xilnx ISE Design Suite. Nexys-3 training board was programmeri with Digilent Adapt 2 software. This work is explained in figures and truth tables, also some mathematical formulas.

The thesis is in Estonian and contains 83 pages of text, 7 chapters, 56 figures, 17 tables.

Lühendite ja mõistete sõnastik

LED	Valgusdiood
FPGA	*Field-programmable gate array*, väliprogrammeeritav maatriks
LUT	*Lookup table*, otsingu tabel
TVT	Tõeväärtustabel
VHDL	Riistvara kirjeldamis keel

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Lõputöö eesmärgiks oli luua kolmendsüsteemi liitja kui ka lahutaja ning kasutada seda Nexsys3 arendusplaadil. Plaadil peab kasutaja andma kahendkoodis sisendi kasutades kaheksat lülitit ning seejärel näeb kasutaja summat lülitite üleval kuvatavatelt LED-idelt. Kahte nuppu kasutades saab kasutaja liita või lahutada kuni kaheksa järgulist positiivset või negatiivset arvu, salvestades ühe registrisse ja sisestades seejärel teise lülititel.

Kolmendsüsteem[1] on arvusüsteem alusega ’3’, järguväärtusteks on ’0’, ’1’, ’2’. Töös on kasutatud tasukaalustatud[2] kolmendsüsteemi, mille järguväärtused on $a_i \in \{-1, 0, +1\}$. Süsteemi polünoomvalemiks on $N = \sum_{i=-m}^{n-1} a_i 3^i$. Arvuformaadi pikkus on n järku, võimalikud 3^n kombinatsioonid jagunevad positiivsete ja negatiivsete arvude vahel kaheks.

Teisendamisel kümnendsüsteemist ja vastupidi, käib samamoodi nagu kahendsüsteemis. 10-nd süsteemist teisendamisel järkude arvu saab teada kui jagada teisendatav arv 3-ga, kolmendsüsteemist teisendamisel tuleb kõik tritid ehk järgud läbi käia.

$$10_3 = 1 * 3^1 + 0 * 3^0 = 3_{10}$$

$$-9_{10} = -1 * 3^2 + 0 * 3^1 + 0 * 3^0 = -100_3$$

Kolmendsüsteemis liitmisel kasutatakse kahte reeglit:

1. Kahe väärtuse ’1’ liitmisel, annab summa järgu väärtuseks ’-1’ ja ülekandeks ’1’.

2. Liites kaks ’-1’, saab summaks ’1’ ja ülekandeks väärtuse ’-1’.

Arendusplaadi jaoks on kirjutatud programm VHDL-is, algselt on kasutatud ModelSim tarkvara, kuid peale programmi valmimist, on kasutatud Xilinx ISE Design Suite-i.

Esimene mehaaniliselt arvutav masin [3], mis arvutas tasakaalustatud kolmendsüsteemis on loodud Thomas Fowleri[4] poolt, aastal 1840, mis oli tehtud täielikult puidust. See oli

Mõõtmetega 1800 x 900 x 300 millimeetrit.

Modernsem kolmendsüsteemis töötav arvuti, nimega „Setun“ [5][6] on loodud aastal 1958 Moskva Ülikooli poolt, loojateks olid Sergei Sobolev ja Nikolay Brusentsov. Neid arvuteid loodi kokku 50 aastast 1959 kuni 1965, kuna siis tootmini peatati. Operatiiset mälu oli arvuti 162 triti lisaks magneetilisel trummil oli tal lisaks 1944 triti, umbes 3.3 kilobaiti. 1965 ja 1970 vahel asendati Setun tavalise kahendsüsteemi arvutiga, mis maksis 2,5 korda rohkem kui Setun. 1970-ndal töötati välja uus mudel kolmendsüsteemi arvutist, nimega „Setun-70“.

2 Kolmendvalentne loogika

Kasutades mitmevalentset loogikat[1] tuli luua loogikalülitused, nende funktsioon ja samuti arvutada tõeväärtustabelid. Liitja-lahutaja on ülesehituselt sarnane kahendkoodis kasutatava liitjaga, kuid lülitused seal sees, sai loodud kasutades LUT-e. Järgnevas tabelis on teisendatud kolmendsüsteemi kümnendarvud, millega saab liita või lahutada plaadil.

Tabel 1. Kolmendsüsteemi tõeväärtustabel 4-järguni

y	x_4	x_3	x_2	x_1	y	x_4	x_3	x_2	x_1	y	x_4	x_3	x_2	x_1
-40	-	-	-	-	-13	0	-	-	-	14	+	-	-	-
-39	-	-	-	0	-12	0	-	-	0	15	+	-	-	0
-38	-	-	-	+	-11	0	-	-	+	16	+	-	-	+
-37	-	-	0	-	-10	0	-	0	-	17	+	-	0	-
-36	-	-	0	0	-9	0	-	0	0	18	+	-	0	0
-35	-	-	0	+	-8	0	-	0	+	19	+	-	0	+
-34	-	-	+	-	-7	0	-	+	-	20	+	-	+	-
-33	-	-	+	0	-6	0	-	+	0	21	+	-	+	0
-32	-	-	+	+	-5	0	-	+	+	22	+	-	+	+
-31	-	0	-	-	-4	0	0	-	-	23	+	0	-	-
-30	-	0	-	0	-3	0	0	-	0	24	+	0	-	0
-29	-	0	-	+	-2	0	0	-	+	25	+	0	-	+
-28	-	0	0	-	-1	0	0	0	-	26	+	0	0	-
-27	-	0	0	0	0	0	0	0	0	27	+	0	0	0
-26	-	0	0	+	1	0	0	0	+	28	+	0	0	+
-25	-	0	+	-	2	0	0	+	-	29	+	0	+	-
-24	-	0	+	0	3	0	0	+	0	30	+	0	+	0
-23	-	0	+	+	4	0	0	+	+	31	+	0	+	+
-22	-	+	-	-	5	0	+	-	-	32	+	+	-	-
-21	-	+	-	0	6	0	+	-	0	33	+	+	-	0
-20	-	+	-	+	7	0	+	-	+	34	+	+	-	+
-19	-	+	0	-	8	0	+	0	-	35	+	+	0	-
-18	-	+	0	0	9	0	+	0	0	36	+	+	0	0
-17	-	+	0	+	10	0	+	0	+	37	+	+	0	+
-16	-	+	+	-	11	0	+	+	-	38	+	+	+	-
-15	-	+	+	0	12	0	+	+	0	39	+	+	+	0
-14	-	+	+	+	13	0	+	+	+	40	+	+	+	+

3 Lülitused

Lõputöö jaoks on loodud järgnevad lülitid[7]. Nende loomiseks on kasutatud VHDL riistvara kirjeldamise keelt. Lülitustest on kasutatud *decoder*it sisendite ja väljundite jaoks, *modulo-3*, *accept anything*, *consensus*, inverterit ja registrit. Liitja koosneb kahest *modulo-3* lülitusest, kahest *consensus* ja ühest *accept anything* lülitusest.
Lahutaja koosneb samuti samadest lülitustest, kuid ühte sisendväärtust on inverteeritud. Liitja-lahutaja jaoks on lisaks kasutatud registrit ja inverterile on juurde lisatud valimise signaal.

3.1 Decoder

3.1.1 *Decoder* sisendite jaoks

Decoder sisendite jaoks teisendab sisse tulevad bitid tritiks. Sisend *decoder*i kood leheküljel 27, testpink leheküljel 45 ja signaaid leheküljel 58.

Tabel 2. Sisend *decoder*i TVT

x_1	x_2	y
0	0	0
0	1	-
1	0	+
1	1	0

3.1.2 *Decoder* väljundite jaoks

Decoder väljundi jaoks, mis teisendab välja mineva triti bitiks. Väljund *decoder*i kood leheküljel 28, testpink leheküljel 45 ja signaalid leheküljel 58.

Tabel 3. Väljund *decoder*i TVT

x	y_1	y_2
0	0	0
-	0	1
+	1	0
0	1	1

3.2 *XOR*

XOR kahendloogikast. *XOR*-i kood leheküljel 29, testpink leheküljel 46 ja signaalid leheküljel 58.

$$(a \oplus \mathrm{b}) = ((a\&\bar{b})V(\bar{a}\&b))$$

Tabel 4.*XOR*-i TVT

x_1	x_2	y
-	-	-
-	0	0
-	+	+
0	-	0
0	0	0
0	+	0
+	-	+
+	0	0
+	+	-

3.3 Minimum

Minimum lülitus ehk loogiline konjunktsioon või *AND* lülitus kahendloogikast. *Minimum*-i kood leheküljel 30, testpink leheküljel 47 ja signaalid leheküljel 58.

Tabel 5. *Minimum*i TVT

x_1	x_2	y
-	-	-
-	0	-
-	+	-
0	-	-
0	0	0
0	+	0
+	-	-
+	0	0
+	+	+

3.4 Maximum

Maximum lülitus ehk loogiline disjunktsioon või *OR* lülitus kahendloogikast. *Maximum*-i kood leheküljel 31, testpink leheküljel 48 ja signaalid leheküljel 58.

Tabel 6. *Maximum*i TVT

x_1	x_2	y
-	-	-
-	0	0
-	+	+
0	-	0
0	0	0
0	+	+
+	-	+
+	0	+
+	+	+

3.5 Inverter

Inversioon ehk loogiline eitus. Inverteri kood 36, testpink 53 ja signaalid 58.

Tabel 7. Inverteri TVT

x	y
-	+
0	0
+	-

3.6 Comparator

Comparator on lülitus, mis võrdleb kahte sisendi oma vahel. Kõrge väärtus ainult võrduse puhul. *Comparator*-i kood leheküljel 33, testpink leheküljel 50 ja signaalid leheküljel 58.

Tabel 8. *Comparator*i TVT

x_1	x_2	y
-	-	+
-	0	-
-	+	-
0	-	-
0	0	+
0	+	-
+	-	-
+	0	-
+	+	+

3.7 Modulo-3

Modulo-3 lülitus ehk siis sisendite summa arvutamiseks. *Modulo-3* kood leheküljel 34, testpink leheküljel 51 ja signaalid leheküljel 58.

$$(x_1 + x_2) = \left((x_1 = -)\&(x_2 = -)\right)V((x_1 = 0)\&(x_2 = 0))V((x_1 = +)\&(x_2 = +))$$

Tabel 9. *Modulo-3* TVT

x_1	x_2	y
-	-	+
-	0	-
-	+	0
0	-	-
0	0	0
0	+	+
+	-	0
+	0	+
+	+	-

3.8 Consensus

Consensus lülitus sama väärtuse kui kaks sisendit on võrdsed, kui erinevad, siis ’0’. *Consensus* kood leheküljel 35, testpink leheküljel 52 ja signaalid leheküljel 58.

$$(x_1 \boxtimes x_2) = (x_1\&x_2)V((x_1 \neq -)\&0)V((x_2 \neq -)\&0)$$

Tabel 10. *Consensus*e TVT

x_1	x_2	y
-	-	-
-	0	0
-	+	0
0	-	0
0	0	0
0	+	0
+	-	0
+	0	0
+	+	+

3.9 Accept anything

Accept anything on loodud 4-nd süsteemis oleva lülituse *gullibility* või *accept anything* põhjal, mis on tegelikult kahekordne *consensus*. *Accept anything* kood leheküljel 32, testpink leheküljel 49 ja signaalid leheküljel 58.

Tabel 11. *Accept anything* TVT

x_1	x_2	y
-	-	-
-	0	0
-	+	+
0	-	0
0	0	0
0	+	+
+	-	+
+	0	+
+	+	+

3.10 Liitja

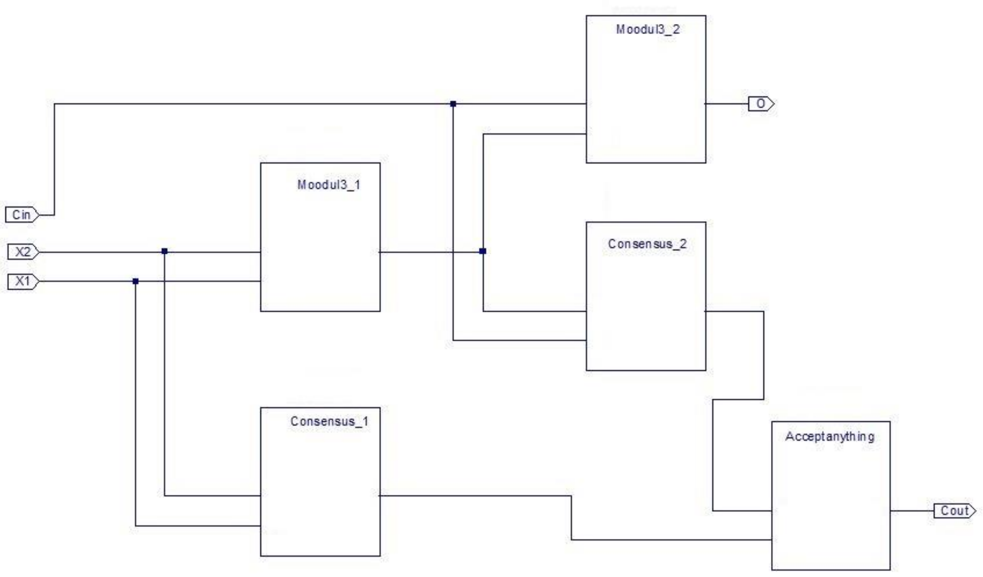
Sumaator on loodud *modulo-3*, *consensus* ja *accept anything* lülititest. Liitja kood leheküljel 37, testpink leheküljel 54 ja signaalid leheküljel 59.

Tabel 12. Liitja TVT

x_1	x_2	C_{in}	C_{out}	o	x_1	x_2	C_{in}	C_{out}	o	x_1	x_2	C_{in}	C_{out}	o
-	-	-	-	0	0	-	-	-	+	+	-	-	0	-
-	-	0	-	+	0	-	0	0	-	+	-	0	0	0
-	-	+	0	-	0	-	+	0	0	+	-	+	0	+
-	0	-	-	+	0	0	-	0	-	+	0	-	0	0
-	0	0	0	-	0	0	0	0	0	+	0	0	0	+
-	0	+	0	0	0	0	+	0	+	+	0	+	+	-
-	+	-	0	-	0	+	-	0	0	+	+	-	0	+
-	+	0	0	0	0	+	0	0	+	+	+	0	+	-
-	+	+	0	+	0	+	+	+	-	+	+	+	+	0

$$
\begin{aligned}
O = {} & C_{in}^{-}(x_1^{+}x_2^{-} + x_1^{0}x_2^{0} + x_1^{-}x_2^{+}) + C_{in}^{0}(x_1^{0}x_2^{0} + x_1^{-}x_2^{0} + x_1^{+}x_2^{+}) \\
& + C_{in}^{+}(x_1^{-}x_2^{-} + x_1^{+}x_2^{0} + x_1^{0}x_2^{+}) + C_{in}^{-}(x_1^{0}x_2^{-} + x_1^{-}x_2^{0} + x_1^{+}x_2^{+}) \\
& + C_{in}^{0}(x_1^{-}x_2^{-} + x_1^{+}x_2^{0} + x_1^{-}x_2^{+}) + C_{in}^{+}(x_1^{+}x_2^{-} + x_1^{0}x_2^{0} + x_1^{-}x_2^{-})
\end{aligned}
$$

$$C = x_1^+ x_2^+ C_{in}^+ + (x_1^+ x_2^0 C_{in}^- + x_1^+ x_2^+ C_{in}^-) + (x_1^- x_2^+ C_{in}^- + x_1^0 x_2^0 C_{in}^0) + (x_2^0 C_{in}^+ + x_2^+ C_{in}^+) \\ + (x_2^+ C_{in}^+ + x_2^+ C_{in}^0) + (x_1^0 C_{in}^+ + x_1^+ C_{in}^+) + x_1^+ C_{in}^0$$



Joonis 1. Liitja skeem

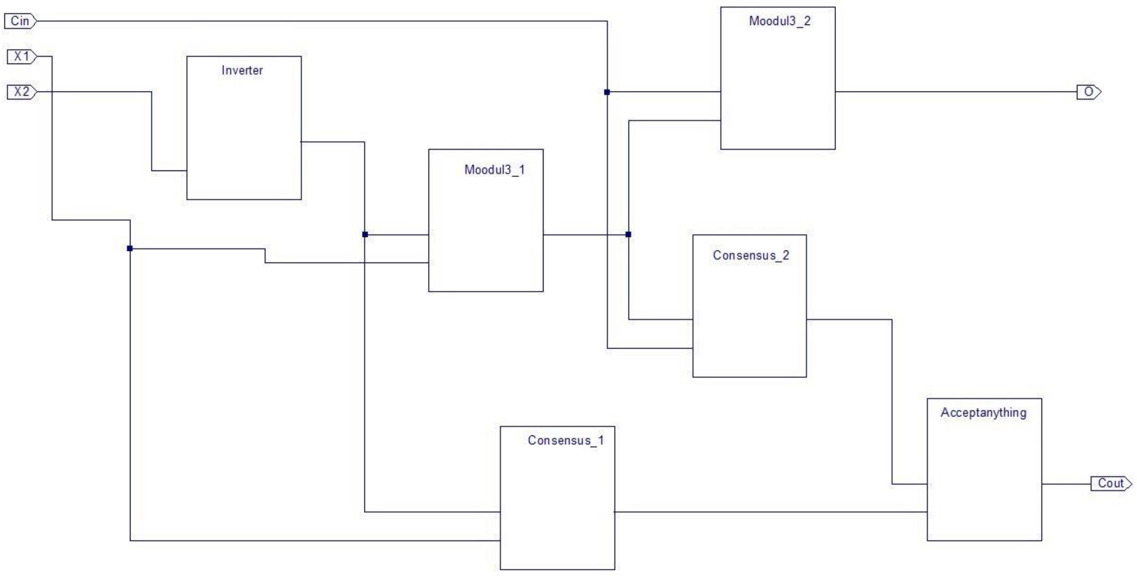
3.11 Lahutaja

Lahutaja on loodud inversiooni, *modulo-3*, *consensus* ja *accept anything* lülititest. Lahutaja kood leheküljel 37, testpink leheküljel 55 ja signaalid leheküljel 61.

$$O = C_{in}^-\left(\overline{x_1^+} x_2^- + \overline{x_1^0} x_2^0 + \overline{x_1^-} x_2^+\right) + C_{in}^0\left(\overline{x_1^0} x_2^0 + \overline{x_1^-} x_2^0 + \overline{x_1^+} x_2^+\right) \\ + C_{in}^+\left(\overline{x_1^-} x_2^- + \overline{x_1^+} x_2^0 + \overline{x_1^0} x_2^+\right) + C_{in}^-\left(\overline{x_1^0} x_2^- + \overline{x_1^-} x_2^0 + \overline{x_1^+} x_2^+\right) \\ + C_{in}^0\left(\overline{x_1^-} x_2^- + \overline{x_1^+} x_2^0 + \overline{x_1^-} x_2^+\right) + C_{in}^+\left(\overline{x_1^+} x_2^- + \overline{x_1^0} x_2^0 + \overline{x_1^-} x_2^-\right)$$

Tabel 13. Lahutaja TVT

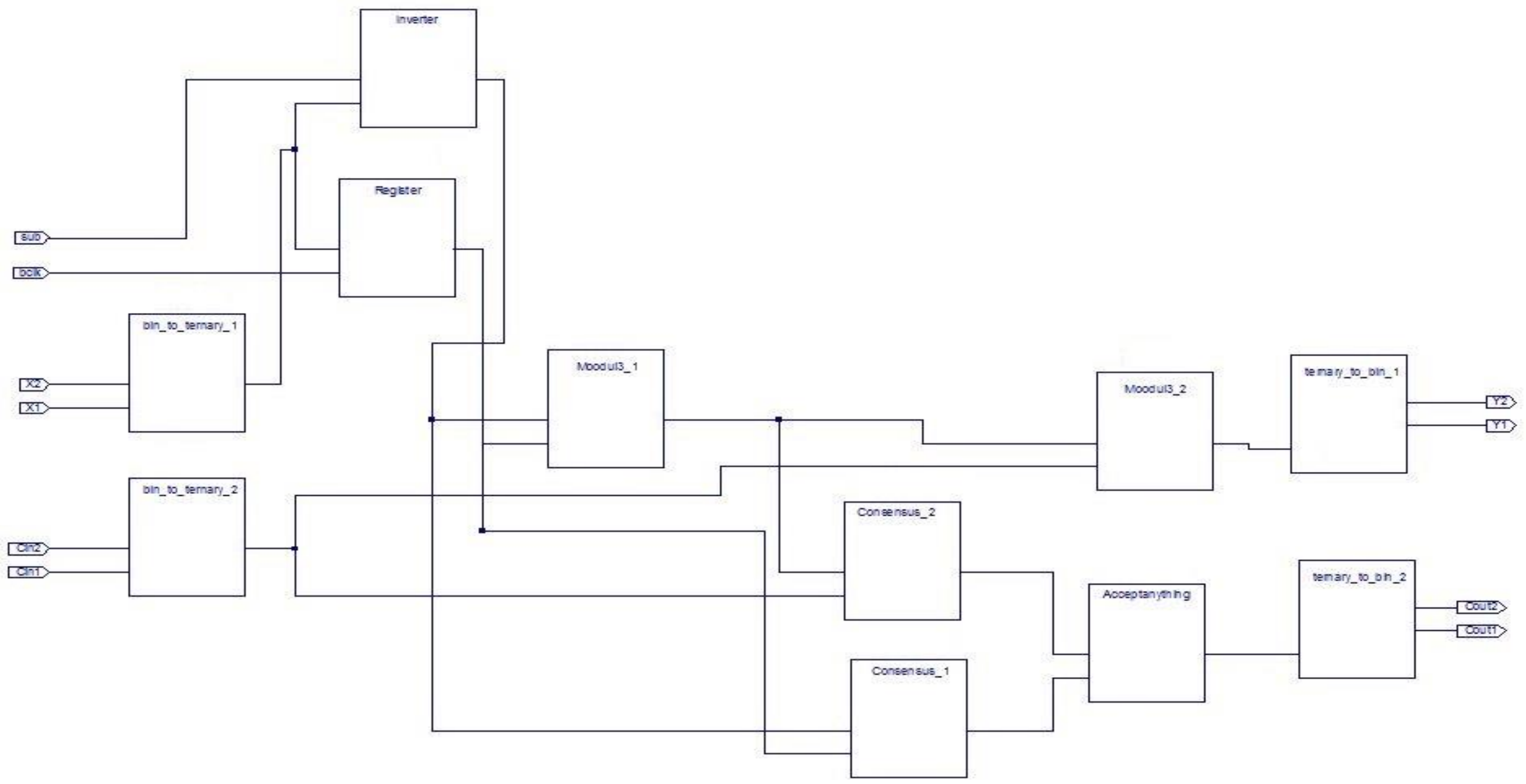
x_1	x_2	C_{in}	C_{out}	o	x_1	x_2	C_{in}	C_{out}	o	x_1	x_2	C_{in}	C_{out}	o
-	-	-	-	-	0	-	-	-	-	+	-	-	-	-
-	-	0	-	0	0	-	0	-	0	+	-	0	-	0
-	-	+	-	+	0	-	+	-	+	+	-	+	-	+
-	0	-	-	-	0	0	-	-	-	+	0	-	-	-
-	0	0	-	0	0	0	0	-	0	+	0	0	-	0
-	0	+	-	+	0	0	+	-	+	+	0	+	-	+
-	+	-	-	-	0	+	-	-	-	+	+	-	-	-
-	+	0	-	0	0	+	0	-	0	+	+	0	-	0
-	+	+	-	+	0	+	+	-	+	+	+	+	-	+



Joonis 2. Lahutaja skeem

3.12 Liitja-lahutaja

Liitja-lahutaja loomiseks on kasutatud inverteri, *modulo-3*, *consensus* ja *accept anything* lülitust. Lisaks on programmis lisatud sisendid bclk ja sub. Programm liidab koheselt, sisestatud väärtuse, kui bclk sisend on ’1’, siis registris oleva väärtusega, muidu iseendaga. Lahutamisel toimib programm samamoodi, kuid selle jaoks peab ka sisend sub olema ’1’. Samuti on lisatud *decoder*id sisendite jaoks kui ka väljundite jaoks. Liitja-lahutaja kood leheküljel 42, testpink leheküljel 56 ja signaalid leheküljel 62.



Joonis 3. Liitja-lahutaja skeem

Tabel 14. Liitja-lahutaja TVT 1

x1	x2	Cin1	Cin2	sub	bclk	y1	y2	Cout1	Cout2
0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	1	1	0	0	0
0	0	0	0	1	0	1	0	0	0
0	0	0	0	1	1	1	0	0	0
0	0	0	1	0	0	0	1	1	0
0	0	0	1	0	1	0	1	1	0
0	0	0	1	1	0	0	1	1	0
0	0	0	1	1	1	0	1	1	0
0	0	1	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0	0	0
0	0	1	0	1	0	0	0	0	0
0	0	1	0	1	1	0	0	0	0
0	0	1	1	0	0	1	0	0	0
0	0	1	1	0	1	1	0	0	0
0	0	1	1	1	0	1	0	0	0
0	0	1	1	1	1	1	0	0	0
0	1	0	0	0	0	0	1	1	0
0	1	0	0	0	1	0	1	1	0
0	1	0	0	1	0	0	1	1	0
0	1	0	0	1	1	0	1	1	0
0	1	0	1	0	0	0	0	1	0
0	1	0	1	0	1	0	0	1	0
0	1	0	1	1	0	0	0	1	0
0	1	0	1	1	1	0	0	1	0
0	1	1	0	0	0	1	0	0	0
0	1	1	0	0	1	1	0	0	0
0	1	1	0	1	0	1	0	0	0
0	1	1	0	1	1	1	0	0	0
0	1	1	1	0	0	0	1	1	0
0	1	1	1	0	1	0	1	1	0
0	1	1	1	1	0	0	1	1	0
0	1	1	1	1	1	0	1	1	0
1	0	0	0	0	0	0	0	0	0

Tabel 15. Liitja-lahutaja TVT 2

x1	x2	Cin1	Cin2	sub	bclk	y1	y2	Cout1	Cout2
1	0	0	0	0	1	0	0	0	0
1	0	0	0	1	0	0	0	0	0
1	0	0	0	1	1	0	0	0	0
1	0	0	1	0	0	1	0	0	0
1	0	0	1	0	1	1	0	0	0
1	0	0	1	1	0	1	0	0	0
1	0	0	1	1	1	1	0	0	0
1	0	1	0	0	0	0	1	0	0
1	0	1	0	0	1	0	1	0	0
1	0	1	0	1	0	0	1	0	0
1	0	1	0	1	1	0	1	0	0
1	0	1	1	0	0	0	0	0	0
1	0	1	1	0	1	0	0	0	0
1	0	1	1	1	0	0	0	0	0
1	0	1	1	1	1	0	0	0	0
1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	1	1	0	0	0
1	1	0	0	1	0	1	0	0	0
1	1	0	0	1	1	1	0	0	0
1	1	0	1	0	0	0	1	1	0
1	1	0	1	0	1	0	1	1	0
1	1	0	1	1	0	0	1	1	0
1	1	0	1	1	1	0	1	1	0
1	1	1	0	0	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0
1	1	1	0	1	0	0	0	0	0
1	1	1	0	1	1	0	0	0	0
1	1	1	1	0	0	1	0	0	0
1	1	1	1	0	1	1	0	0	0
1	1	1	1	1	0	1	0	0	0
1	1	1	1	1	1	1	0	0	0

4 Lülituste program

Lülitused on kirjutatud VHDL-is, kasutades ModelSIM tarkvara ning loodud vastavalt tõeväärtustabeli väärtustele, mis on saadud kasutades eelnevalt välja toodud loogikat.

4.1 Pakett

```
package ternary_logic is
        subtype ternary is bit_vector(0 to 1);
                constant Plus:  ternary := "01";
                constant Zero:  ternary := "00";
                constant Minus: ternary := "10";
end package ternary_logic;
```

Joonis 4. Paketi kood

4.2 Decoder

4.2.1 *Decoder* sisendite jaoks

Sisendite *decoder*-i loogika leheküljel 15, testpink leheküljel 44 ja signaalid leheküljel 58.

```
package ternary_logic is
        type ternary is (Minus, Zero, Plus);
end package ternary_logic;


use work.ternary_logic.all;

entity decoder_sisse is
        port ( x1, x2: in bit;
                y: out ternary);
end entity decoder_sisse;

architecture bhv of decoder_sisse is
begin
        process (x1, x2) begin
                case x1 is
                        when '0'=>
                                case x2 is
                                        when '0' => y <= Zero;
                                        when '1' => y <= Minus;
                                end case;
                        when '1'=>
                                case x2 is
                                        when '0' => y <= Plus;
                                        when '1' => y <= Zero;
                                end case;
                end case;
        end process;
end architecture bhv;
```

Joonis 5. Sisend *decoder*

4.2.2 *Decoder* väljundite jaoks

Väljundite *decoder*-i loogika leheküljel 16, testpink leheküljel 45 ja signaalid leheküljel 58.

```
package ternary_logic is
        type ternary is (Minus, Zero, Plus);
end package ternary_logic;

use work.ternary_logic.all;

entity decoder_valja is
        port ( x: in ternary;
                y1, y2: out bit);
end entity decoder_valja;

architecture bhv of decoder_valja is
begin
        process (x) begin
                case x is
                        when Plus=> y1<= '0';
                                    y2<= '1';
                        when Zero=> y1<= '0';
                                    y2<= '0';
                        when Minus=> y1<= '1';
                                    y2<= '0';
                        end case;
        end process;
end architecture bhv;
```

Joonis 6. Väljund *decoder*

4.3 XOR

XOR-i loogika leheküljel 16, testpink leheküljel 46 ja signaalid leheküljel 58.

```
package ternary_logic is
        type ternary is (Minus, Zero, Plus);
end package ternary_logic;

use work.ternary_logic.all;

entity iksor is
        port ( x1, x2: in ternary;
                y: out ternary );
end entity iksor;

architecture bhv of iksor is
begin
        process (x1, x2) begin
                case x1 is
                        when Plus=>
                                case x2 is
                                        when Plus => y <= Minus;
                                        when Zero => y <= Zero;
                                        when Minus => y <= Plus;
                                end case;
                        when Zero=>
                                case x2 is
                                        when Plus => y <= Zero;
                                        when Zero => y <= Zero;
                                        when Minus => y <= Zero;
                                end case;
                        when Minus=>
                                case x2 is
                                        when Plus => y <= Plus;
                                        when Zero => y <= Zero;
                                        when Minus => y <= Minus;
                                end case;
                        when others => y <= Zero;
                end case;
        end process;
end architecture bhv;
```

Joonis 7. *XOR* lülitus

4.4 Minimum

Minimum-i loogika leheküljel 17, testpink leheküljel 47 ja signaalid leheküljel 58.

```
package ternary_logic is
        type ternary is (Minus, Zero, Plus);
end package ternary_logic;

use work.ternary_logic.all;

entity min is
        port ( x1, x2: in ternary;
                y: out ternary );
end entity min;

architecture bhv of min is
begin
        process (x1, x2) begin
                case x1 is
                        when Plus=>
                                case x2 is
                                        when Plus => y <= Plus;
                                        when Zero => y <= Zero;
                                        when Minus => y <= Minus;
                                end case;
                        when Zero=>
                                case x2 is
                                        when Plus => y <= Zero;
                                        when Zero => y <= Zero;
                                        when Minus => y <= Minus;
                                end case;
                        when Minus=>
                                case x2 is
                                        when Plus => y <= Minus;
                                        when Zero => y <= Minus;
                                        when Minus => y <= Minus;
                                end case;
                        when others => y <= Zero;
                 end case;
        end process;
end architecture bhv;
```

Joonis 8. *Minimum* lülitus

4.5 Maximum

Maximum-i loogika leheküljel 17, testpink leheküljel 48 ja signaalid leheküljel 58.

```
package ternary_logic is
        type ternary is (Minus, Zero, Plus);
end package ternary_logic;

use work.ternary_logic.all;

entity max is
        port ( x1, x2: in ternary;
                y: out ternary);
end entity max;

architecture bhv of max is
begin
        process (x1, x2) begin
                case x1 is
                        when Plus=>
                                case x2 is
                                        when Plus => y <= Plus;
                                        when Zero => y <= Plus;
                                        when Minus => y <= Plus;
                                end case;
                        when Zero=>
                                case x2 is
                                        when Plus => y <= Plus;
                                        when Zero => y <= Zero;
                                        when Minus => y <= Zero;
                                end case;
                        when Minus=>
                                case x2 is
                                        when Plus => y <= Plus;
                                        when Zero => y <= Zero;
                                        when Minus => y <= Minus;
                                end case;
                                        when others => y <= Zero;
                        end case;
        end process;
end architecture bhv;
```

Joonis 9. *Maximum* lülitus

4.6 Accept anything

Accept anything-i loogikaleheküljel 20 , testpink leheküljel 49 ja signaalid leheküljel 58.

```
package ternary_logic is
        type ternary is (Minus, Zero, Plus);
end package ternary_logic;

use work.ternary_logic.all;

entity acceptany is
        port ( x1, x2: in ternary;
                y: out ternary );
end entity acceptany;

architecture bhv of acceptany is
begin
        process (x1, x2) begin
                case x1 is
                        when Plus=>
                                case x2 is
                                        when Plus => y <= Plus;
                                        when Zero => y <= Plus;
                                        when Minus => y <= Zero;
                                end case;
                        when Zero=>
                                case x2 is
                                        when Plus => y <= Plus;
                                        when Zero => y <= Zero;
                                        when Minus => y <= Minus;
                                end case;
                        when Minus=>
                                case x2 is
                                        when Plus => y <= Zero;
                                        when Zero => y <= Minus;
                                        when Minus => y <= Minus;
                                end case;
                        when others => y <= Zero;
                end case;
        end process;
end architecture bhv;
```

Joonis 10. *Accept anything* lülitus

4.7 Comparator

Comparator-i loogika leheküljel 18, testpink leheküljel 50 ja signaalid leheküljel 58.

```
package ternary_logic is
        type ternary is (Minus, Zero, Plus);
end package ternary_logic;

use work.ternary_logic.all;

entity comp is
        port ( x1, x2: in ternary;
                y: out ternary );
end entity comp;

architecture bhv of comp is
begin
        process (x1, x2) begin
                case x1 is
                        when Plus=>
                                case x2 is
                                        when Plus => y <= Plus;
                                        when Zero => y <= Minus;
                                        when Minus => y <= Minus;
                                end case;
                        when Zero=>
                                case x2 is
                                        when Plus => y <= Minus;
                                        when Zero => y <= Plus;
                                        when Minus => y <= Minus;
                                end case;
                        when Minus =>
                                case x2 is
                                        when Plus => y <= Minus;
                                        when Zero => y <= Minus;
                                        when Minus => y <= Plus;
                                end case;
                        when others => y <= Zero;
                end case;
        end process;
end architecture bhv;
```

Joonis 11. *Comparator* lülitus

4.8 Modulo-3

Modulo-3 loogika leheküljel 19, testpink leheküljel 51 ja signaalid leheküljel 58.

```
package ternary_logic is
        type ternary is (Minus, Zero, Plus);
end package ternary_logic;

use work.ternary_logic.all;

entity modulo_3 is
        port ( x1, x2: in ternary;
                y: out ternary );
end entity modulo_3;

architecture bhv of modulo_3 is
begin
        process (x1, x2) begin
                case x1 is
                        when Plus=>
                                case x2 is
                                        when Plus => y <= Minus;
                                        when Zero => y <= Plus;
                                        when Minus => y <= Zero;
                                end case;
                        when Zero=>
                                case x2 is
                                        when Plus => y <= Plus;
                                        when Zero => y <= Zero;
                                        when Minus => y <= Minus;
                                end case;
                        when Minus=>
                                case x2 is
                                        when Plus => y <= Zero;
                                        when Zero => y <= Minus;
                                        when Minus => y <= Plus;
                                end case;
                        when others => y <= Zero;
                end case;
        end process;
end architecture bhv;
```

Joonis 12. *Modulo-3* lülitus

4.9 Consensus

Consensus loogika leheküljel 19, testpink leheküljel 52 ja signaalid leheküljel 58.

```
package ternary_logic is
        type ternary is (Minus, Zero, Plus);
end package ternary_logic;

use work.ternary_logic.all;

entity cons is
        port ( x1, x2 : in ternary;
                y: out ternary );
end entity cons;

architecture bhv of cons is
begin
        process (x1, x2) begin
                case x1 is
                        when Plus=>
                                case x2 is
                                        when Plus => y <= Plus;
                                        when Zero => y <= Zero;
                                        when Minus => y <= Zero;
                                end case;
                        when Zero=>
                                case x2 is
                                        when Plus => y <= Zero;
                                        when Zero => y <= Zero;
                                        when Minus => y <= Zero;
                                end case;
                        when Minus=>
                                case x2 is
                                        when Plus => y <= Zero;
                                        when Zero => y <= Zero;
                                        when Minus => y <= Minus;
                                end case;
                end case;
        end process;
end architecture bhv;
```

Joonis 13. *Consensus* lülitus

4.10 Inverter

Inverteri loogika leheküljel 18, testpink leheküljel 53 ja signaalid leheküljel 58.

```
package ternary_logic is
        type ternary is (Minus, Zero, Plus);
end package ternary_logic;

use work.ternary_logic.all;

entity inv is
        port ( x: in ternary;
                y: out ternary );
end entity inv;

architecture bhv of inv is
begin
        process (x) begin
                case x is
                        when Plus   =>  y <= Minus;
                        when Minus  =>  y <= Plus;
                        when others =>  y <= Zero;
                end case;
        end process;
end architecture bhv;
```

Joonis 14. Inverter lülitus

4.11 Liitja

Liitja loogika leheküljel 20, testpink leheküljel 54 ja signaalid leheküljel 59.

```
package ternary_logic is
        type ternary is (Minus, Zero, Plus);
end package ternary_logic;

use work.ternary_logic.all;

entity fullAdder is
        port( x1, x2, Cin : in ternary;
                Cout, y : out ternary);
end entity fullAdder;

architecture bhv of fullAdder is
        component modulo_3 is
                port ( x1, x2: in ternary;
                        y: out ternary );
        end component;
        component acceptany is
                port ( x1, x2: in ternary;
                y: out ternary );
        end component;
        component cons_1 is
                port ( x1, x2 : in ternary;
                y: out ternary );
        end component;


signal: modulo_3_1_out, modulo_3_2_out , cons_1_out, cons_2_out, acceptany_out
: ternary;

begin

U1: modulo_3 port map (x1, x2, modulo_3_1_out);
U2: cons port map (x1, x2, cons_1_out);
U3: modulo_3 port map (Cin, modulo_3_1_out,  modulo_3_2_out);
U4: cons port map (modulo_3_1_out, Cin, cons_2_out);
U5: acceptany port map (cons_1_out, cons_2_out, acceptany_out);

end architecture bhv;
```

Joonis 15. Liitja programmi kood

4.12 Lahutaja

Lahutaja loogika leheküljel 21, testpink leheküljel 55 ja signaalid leheküljel 61.

```
package ternary_logic is
        type ternary is (Minus, Zero, Plus);
end package ternary_logic;

use work.ternary_logic.all;

entity fullSubtractor is
        port( x1, x2, Cin : in ternary;
                 Cout, o : out ternary);
end entity fullSubtractor;

architecture bhv of fullSubtractor is
        component modulo_3 is
                port ( x1, x2: in ternary;
                y: out ternary );
        end component;
        component acceptany is
                port ( x1, x2: in ternary;
                y: out ternary );
         end component;
        component cons is
                port ( x1, x2 : in ternary;
                y: out ternary );
        end component;
        component inv is
                port (x: in ternary;
                        y: out ternary);
        end component;

signal inv_out, modulo_3_1_out, cons_1_out, cons_2_out, acceptany_out,
modulo_3_2_out : ternary;
```

Joonis 16. Lahutaja programmikood 1

```
begin

U0: inv port map (x1, inv_out);
U1: modulo_3 port map (inv_out, x2, modulo_3_1_out);
U2: cons port map (inv_out, x2, cons_1_out);
U3: modulo_3 port map (Cin, modulo_3_1_out, modulo_3_2_out);
U4: cons port map (modulo_3_1_out, Cin, cons_2_out);
U5: acceptany port map (cons_1_out, cons_2_out, acceptany_out);

end architecture bhv;
```

Joonis 17. Lahutaja programmikood 2

4.13 Register

```
package ternary_logic is
        type ternary is (Minus, Zero, Plus);
end package ternary_logic;

use work.ternary_logic.all;

entity register1 is
        port( bclk : in bit;
                dir : in ternary;
                reg : out ternary);
end entity register1;

architecture bhv of register1 is
begin
        process (bclk) begin
                if bclk = '0' and bclk'event then
                        reg <= dir;
                end if;
        end process;
end architecture bhv;
```

Joonis 18. Registrisse salvestamine

4.14 Valikuga inverter

```
package ternary_logic is
        type ternary is (Minus, Zero, Plus);
end package ternary_logic;

use work.ternary_logic.all;

entity inv is
        port ( sub: in bit;
                x: in ternary;
                y: out ternary );
        end entity inv;

architecture bhv of inv is
begin
        process (sub, x) begin
                case sub is
                        when '1' =>
                                case x is
                                        when Plus   =>  y <= Minus;
                                        when Minus  =>  y <= Plus;
                                        when others =>  y <= Zero;
                                end case;
                        when  '0' =>
                                case x is
                                        when Plus   =>  y <= Plus;
                                        when Minus  =>  y <= Minus;
                                        when others =>  y <= Zero;
                                end case;
                end case;
        end process;
end architecture bhv;
```

Joonis 19. Valikuga inverteri programmikood

4.15 Liitja-lahutaja

Liitja-lahutaja loogika leheküljel 22, testpink leheküljel 56 ja signaalid leheküljel 62.

```
package ternary_logic is
        type ternary is (Minus, Zero, Plus);
end package ternary_logic;

use work.ternary_logic.all;

entity full_adder_sub is
        port( bclk, sub, x1, x2,Cin1, Cin2 : in bit;
                y1, y2, C1, C2  : out bit);
end entity full_adder_sub;

use work.ternary_logic.all;

architecture bhv of full_adder_sub is
        component fullAdder is
                        port (x1, x2, Cin: in ternary;
                                Cout, y: out ternary);
        end component;
        component inv is
                        port (sub: in bit;
                                x : in ternary;
                                y : out ternary);
        end component;
        component decoder_sisse is
                        port (x1, x2: in bit;
                                y : out ternary);
        end component;
        component decoder_valja is
                        port (x: in ternary;
                        y1, y2: out bit);
        end component;
        component register1 is
                        port (bclk : in bit;
                                dir : in ternary;
                                 reg : out ternary);
        end component;

        signal de0, Cin, r0, S0, Cout0, de0_inv : ternary;
```

Joonis 20. Liitja-lahutaja programm 1

```
        begin
                U6: decoder_sisse port map (x1, x2, de0);
                U7 : decoder_sisse port map (Cin1, Cin2, Cin);
                U8 : inv port map (sub, de0, de0_inv);
                U9: register1 port map (bclk, de0, r0);
                U10: fullAdder port map (r0, de0_inv, Cin, Cout0,  S0);
                U11: decoder_valja port map (S0, y1, y2);
                U12: decoder_valja port map (Cout0, C1, C2);
end architecture bhv;
```

Joonis 21. Liitja-lahutaja programm 2

5 Testimine

Üksikute lülituste testimiseks kasutati ModelSIM tarkvara, summatori, lahutaja ning liitja-lahutaja testimiseks kasutati Xilnx ISE tarkvara simulatsiooniks. Simuleerimiseks on ette antud sisendsignaalid tõeväärtustabelite järgi. Simulatsiooni testpink on kirjutatud VHDL-is.

5.1 *Decoder*

5.1.1 Sisendi *decoder*

Sisend *decoder*-i loogika leheküljel 15, kood leheküljel 27 ja signaalid leheküljel 58.

```
use work.ternary_logic.all;

entity test_decoder_sisse is
end entity test_decoder_sisse;

architecture bench of test_decoder_sisse is
        signal x1, x2: bit;
        signal y: ternary;
        component decoder_sisse
                port ( x1, x2: bit;
                        y: out ternary );
        end component;
begin

        x1 <= '0' after 0 ns,  '0' after 10 ns, '1' after 20 ns, '1' after 30 ns;
        x2 <= '0' after 0 ns, '1' after 10 ns, '0' after 20 ns, '1' after 30 ns;

        U1: decoder_sisse port map (x1, x2, y);
end architecture bench;
```

Joonis 22. Sisendi *decoder*i testpink

5.1.2 Väljundi *decoder*

Väljund *decoder*-i loogika leheküljel 16, kood leheküljel 28 ja signaalid leheküljel 58.

```
use work.ternary_logic.all;

entity test_decoder_valja is
end entity test_decoder_valja;

architecture bench of test_decoder_valja is
        signal x: ternary;
        signal y1, y2: bit;

        component decoder_valja
                port ( x: ternary;
                        y1, y2: out bit );
        end component;
begin

        x <= Plus after 0 ns, Zero after 10 ns, Minus after 20 ns;

        U1: decoder_valja port map (x, y1, y2);
end architecture bench;
```

Joonis 23. Väljundi *decoder*i testpink

5.2 XOR

XOR-i loogika leheküljel 16, kood leheküljel 29 ja signaalid leheküljel 58.

```
use work.ternary_logic.all;

entity test_iksor is
end entity test_iksor;

architecture bench of test_iksor is
        signal x1, x2: ternary;
        signal y: ternary;

        component iksor
                port ( x1, x2: in ternary;
                        y: out ternary );
        end component;

        begin

                x1 <= Minus after 0 ns, Minus after 10 ns, Minus after 20 ns, Zero
                after 30 ns, Zero after 40 ns, Zero after 50 ns, Plus after 60 ns, Plus
                after 70 ns, Plus after 80 ns;
                x2 <= Minus after 0 ns, Zero after 10 ns, Plus after 20 ns, Minus after
                30 ns, Zero after 40 ns, Plus after 50 ns, Minus after 60 ns, Zero after
                70 ns, Plus after 80 ns;

        U1: iksor port map (x1, x2, y);
end architecture bench;
```

Joonis 24. *XOR*-i testpink

5.3 Minimum

Minimum-i loogika leheküljel 17, kood leheküljel 30 ja signaalid leheküljel 58.

```
use work.ternary_logic.all;

entity test_min is
end entity test_min;


architecture bench of test_min is
        signal x1, x2: ternary;
        signal y: ternary;

        component min
                port ( x1, x2: in ternary;
                        y: out ternary );
        end component;

        begin

                x1 <= Minus after 0 ns, Minus after 10 ns, Minus after 20 ns, Zero
                after 30 ns, Zero     after 40 ns, Zero after 50 ns, Plus after 60 ns, Plus
                after 70 ns, Plus after 80 ns;
                x2 <= Minus after 0 ns, Zero after 10 ns, Plus after 20 ns, Minus after
                30 ns, Zero after 40 ns, Plus after 50 ns, Minus after 60 ns, Zero after
                70 ns, Plus after 80 ns;

        U1: min port map (x1, x2, y);
end architecture bench;
```

Joonis 25. *Minimum*i testpink

5.4 Maximum

Maximum-i loogika leheküljel 17, kood leheküljel 31 ja signaalid leheküljel 58.

```
use work.ternary_logic.all;

entity test_max is
end entity test_max;

architecture bench of test_max is
        signal x1, x2: ternary;
        signal y: ternary;

        component max
                port ( x1, x2: in ternary;
                        y: out ternary );
        end component;

        begin

                x1 <= Minus after 0 ns, Minus after 10 ns, Minus after 20 ns, Zero
                after 30 ns, Zero after 40 ns, Zero after 50 ns, Plus after 60 ns, Plus
                after 70 ns, Plus after 80 ns;
                x2 <= Minus after 0 ns, Zero after 10 ns, Plus after 20 ns, Minus after
                30 ns, Zero after 40 ns, Plus after 50 ns, Minus after 60 ns, Zero after
                70 ns, Plus after 80 ns;

        U1: max port map (x1, x2, y);
end architecture bench;
```

Joonis 26. *Maximum* testpink

5.5 Accept anything

Accept anything loogika leheküljel 20, kood leheküljel 32 ja signaalid leheküljel 58.

```
use work.ternary_logic.all;

entity test_acceptany is
end entity test_acceptany;

architecture bench of test_acceptany is
        signal x1, x2: ternary;
        signal y: ternary;

        component acceptany
                port ( x1, x2: in ternary;
                        y: out ternary );
        end component;

        begin

                x1 <= Minus after 0 ns, Minus after 10 ns, Minus after 20 ns, Zero
                after 30 ns, Zero after 40 ns, Zero after 50 ns, Plus after 60 ns, Plus
                after 70 ns, Plus after 80 ns;
                x2 <= Minus after 0 ns, Zero after 10 ns, Plus after 20 ns, Minus after
                30 ns, Zero after 40 ns, Plus after 50 ns, Minus after 60 ns, Zero after
                70 ns, Plus after 80 ns;

        U1: acceptany port map (x1, x2, y);

end architecture bench;
```

Joonis 27. *Accept anything* testpink

5.6 **C*omparator***

Comparator-i loogika leheküljel 18, kood leheküljel 33 ja signaalid leheküljel 58.

```
use work.ternary_logic.all;

entity test_comp is
end entity test_comp;

architecture bench of test_comp is
        signal x1, x2: ternary;
        signal y: ternary;

        component comp
                port ( x1, x2: in ternary;
                y: out ternary );
        end component;

        begin

                x1 <= Minus after 0 ns, Minus after 10 ns, Minus after 20 ns, Zero
                after 30 ns, Zero after 40 ns, Zero after 50 ns, Plus after 60 ns, Plus
                after 70 ns, Plus after 80 ns;
                x2 <= Minus after 0 ns, Zero after 10 ns, Plus after 20 ns, Minus after
                30 ns, Zero after 40 ns, Plus after 50 ns, Minus after 60 ns, Zero after
                70 ns, Plus after 80 ns;

        U1: comp port map (x1, x2, y);
end architecture bench;
```

Joonis 28. *Comparator*-i testpink

5.7 Modulo-3

Modulo-3 loogika leheküljel 19, kood leheküljel 34 ja signaalid leheküljel 58.

```
use work.ternary_logic.all;

entity test_modulo_3 is
end entity test_modulo_3;

architecture bench of test_modulo_3 is
        signal x1, x2: ternary;
        signal y: ternary;

        component modulo_3
                port ( x1, x2: in ternary;
                        y: out ternary );
        end component;

        begin

                 x1 <= Minus after 0 ns, Minus after 10 ns, Minus after 20 ns, Zero
                after 30 ns, Zero after 40 ns, Zero after 50 ns, Plus after 60 ns, Plus
                after 70 ns, Plus after 80 ns;
                x2 <= Minus after 0 ns, Zero after 10 ns, Plus after 20 ns, Minus after
                30 ns, Zero after 40 ns, Plus after 50 ns, Minus after 60 ns, Zero after
                70 ns, Plus after 80 ns;

        U1: modulo_3 port map (x1, x2, y);
end architecture bench;
```

Joonis 29. *Modulo-3* testpink

5.8 **C*onsensus***

Consensus loogika leheküljel 19, kood leheküljel 35 ja signaalid leheküljel 58.

```
use work.ternary_logic.all;

entity test_cons is
end entity test_cons;

architecture bench of test_cons is
        signal x1, x2: ternary;
        signal y: ternary;

        component cons
                port ( x1, x2: in ternary;
                        y: out ternary );
        end component;

        begin

                x1 <= Minus after 0 ns, Minus after 10 ns, Minus after 20 ns, Zero
                after 30 ns, Zero after 40 ns, Zero after 50 ns, Plus after 60 ns, Plus
                after 70 ns, Plus after 80 ns;
                x2 <= Minus after 0 ns, Zero after 10 ns, Plus after 20 ns, Minus after
                30 ns, Zero after 40 ns, Plus after 50 ns, Minus after 60 ns, Zero after
                70 ns, Plus after 80 ns;

        U1: cons port map (x1, x2, y);
end architecture bench;
```

Joonis 30. *Consensus* testpink

5.9 Inverter

Inverteri loogika leheküljel 18, kood leheküljel 36 ja signaalid leheküljel 58.

```
use work.ternary_logic.all;

entity test_inv is
end entity test_inv;

architecture bench of test_inv is
        signal x: ternary;
        signal y: ternary;

        component inv
                port ( x: in ternary;
                        y: out ternary );
        end component;

        begin

         x <= Minus after 0 ns, Zero after 10 ns, Plus after 30 ns;


        U1: inv port map (x, y);
end architecture bench;
```

Joonis 31. Invereteri testpink

5.10 Liitja

Liitja loogika leheküljel 20, kood leheküljel 37 ja signaalid leheküljel 59.

```
use work.ternary_logic.all;

entity test_fulladder is
end entity test_fulladder;

architecture bench of test_fulladder is
        signal x1, x2, Cin : ternary;
        signal Cout, y : ternary;

        component fullAdder is
                port ( x1, x2, Cin: in ternary;
                        Cout, y: out ternary);
        end component;

        begin

        x1 <= Minus after 0 ns, Minus after 5 ns, Minus after 10 ns, Minus after 15
        ns, Minus after 20 ns, Minus after 25 ns, Minus after 30 ns, Minus after 35
        ns, Minus after 40 ns, Zero after 45 ns, Zero after 50 ns, Zero after 55 ns,
        Zero after 60 ns, Zero after 65 ns, Zero after 70 ns, Zero after 75 ns, Zero
        after 80 ns, Zero after 85 ns, Plus after 90 ns, Plus after 95 ns, Plus after 100
        ns, Plus after 105 ns, Plus after 110 ns, Plus after 115 ns, Plus after 120 ns,
        Plus after 125 ns, Plus after 130 ns;

        x2 <= Minus after 0 ns, Minus after 5 ns, Minus after 10 ns, Zero after 15 ns,
        Zero after 20 ns, Zero after 25 ns, Plus after 30 ns, Plus after 35 ns, Plus after
        40 ns, Minus after 45 ns, Minus after 50 ns, Minus after 55 ns, Zero after 60
        ns, Zero after 65 ns, Zero after 70 ns, Plus after 75 ns, Plus after 80 ns, Plus
        after 85 ns, Minus after 90 ns, Minus after 95 ns, Minus after 100 ns, Zero
        after 105 ns, Zero after 110 ns, Zero after 115 ns, Plus after 120 ns, Plus after
        125 ns, Plus after 130 ns;

        Cin <= Minus after 0 ns, Zero after 5 ns, Plus after 10 ns,  Minus after 15 ns,
        Zero after 20 ns, Plus after 25 ns,  Minus after 30 ns, Zero after 35 ns, Plus
        after 40 ns,  Minus after 45 ns, Zero after 50 ns, Plus after 55 ns,  Minus after
        60 ns, Zero after 65 ns, Plus after 70 ns,  Minus after 75 ns, Zero after 80 ns,
        Plus after 85 ns,  Minus after 90 ns, Zero after 95 ns, Plus after 100 ns,
        Minus after 105 ns, Zero after 110 ns, Plus after 115 ns,  Minus after 120 ns,
        Zero after 125 ns, Plus after 130 ns;

        U6: fullAdder port map (x1, x2, Cin, Cout, y);
```

Joonis 32. Täisliitja testpink

5.11 Lahutaja

Lahutaja loogika leheküljel 21, kood leheküljel 38 ja signaalid leheküljel 61.

```
use work.ternary_logic.all;

entity test_fullAdderSubtractor is
end entity test_fullAdderSubtractor;

architecture bench of test_fullSubtractor is
        signal x1, x2, Cin : ternary;
        signal Cout, o : ternary;

        component fullSubtractor
                port (x1, x2, Cin: in ternary;
                        Cout, o: out ternary);
        end component;

        begin

        x1 <= Minus after 0 ns, Minus after 5 ns, Minus after 10 ns, Minus after 15
        ns, Minus after 20 ns, Minus after 25 ns, Minus after 30 ns, Minus after 35
        ns, Minus after 40 ns, Zero after 45 ns, Zero after 50 ns, Zero after 55 ns,
        Zero after 60 ns, Zero after 65 ns, Zero after 70 ns, Zero after 75 ns, Zero
        after 80 ns, Zero after 85 ns, Plus after 90 ns, Plus after 95 ns, Plus after 100
        ns, Plus after 105 ns, Plus after 110 ns, Plus after 115 ns, Plus after 120 ns,
        Plus after 125 ns, Plus after 130 ns;
        x2 <= Minus after 0 ns, Minus after 5 ns, Minus after 10 ns, Zero after 15 ns,
        Zero after 20 ns, Zero after 25 ns, Plus after 30 ns, Plus after 35 ns, Plus after
        40 ns, Minus after 45 ns, Minus after 50 ns, Minus after 55 ns, Zero after 60
        ns, Zero after 65 ns, Zero after 70 ns, Plus after 75 ns, Plus after 80 ns, Plus
        after 85 ns, Minus after 90 ns, Minus after 95 ns, Minus after 100 ns, Zero
        after 105 ns, Zero after 110 ns, Zero after 115 ns, Plus after 120 ns, Plus after
        125 ns, Plus after 130 ns;
        Cin <= Minus after 0 ns, Zero after 5 ns, Plus after 10 ns,  Minus after 15 ns,
        Zero after 20 ns, Plus after 25 ns,  Minus after 30 ns, Zero after 35 ns, Plus
        after 40 ns,  Minus after 45 ns, Zero after 50 ns, Plus after 55 ns,  Minus after
        60 ns, Zero after 65 ns, Plus after 70 ns,  Minus after 75 ns, Zero after 80 ns,
        Plus after 85 ns,  Minus after 90 ns, Zero after 95 ns, Plus after 100 ns,
        Minus after 105 ns, Zero after 110 ns, Plus after 115 ns,  Minus after 120 ns,
        Zero after 125 ns, Plus after 130 ns;

        U6: fullSubtractor port map (x1, x2, Cin, Cout, o);
end architecture bench;
```

Joonis 33. Lahutaja testpink

5.12 Liitja-lahutaja

Liitja-lahutaja loogika leheküljel 22, kood leheküljel 42 ja signaalid leheküljel 62.

5.12.1.1 Esimene test (x1=0)

```
use work.ternary_logic.all;

entity test_full_adder_sub is
end entity test_full_adder_sub;

architecture bench of test_full_adder_sub is

        signal x1, x2, Cin1, Cin2 : bit;
        signal y1, y2, C1, C2 : bit;
component full_adder_sub
                port( sub, bclk, x1, x2, Cin1, Cin2 : in bit;
                        y1, y2, C1, C2 : out bit);
        end component full_adder_sub;
        signal bclk: bit := '0';
        signal sub: bit := '0';
begin
        x1 <=  '0' after 0 ns;
        x2 <=  '0' after 0 ns, '1' after 80 ns, '0' after 160 ns;
        Cin1 <=  '0' after 0 ns, '1' after 40 ns, '1' after 50 ns, '1' after 70 ns, '0' after 80
        ns, '1' after 120 ns, '0' after 160 ns;
        Cin2 <=  '0' after 0 ns, '1' after 20 ns, '0' after 40 ns, '1' after 60 ns, '1' after 70
        ns, '0' after 80 ns, '1' after 100 ns, '0' after 120 ns, '1' after 140 ns,
        '0' after 160 ns;
        sub <=  '0' after 0 ns,  '1' after 10 ns,  '0' after 20 ns, '1' after 30 ns, '0' after 40
        ns, '1' after 50 ns, '0' after 60 ns, '1' after 70 ns, '0' after 80 ns,
        '1' after 90 ns, '0' after 100 ns, '1' after 110 ns, '0' after 120 ns, '1' after 130 ns,
        '0' after 140 ns, '1' after 150 ns, '0' after 160 ns;
        bclk <=  '0' after 0 ns,  '1' after 5 ns,  '0' after 10 ns,  '1' after 15 ns,  '0' after 20
        ns, '1' after 25 ns, '0' after 30 ns, '1' after 35 ns, '0' after 40 ns,
        '1' after 45 ns, '0' after 50 ns, '1' after 55 ns, '0' after 60 ns, '1' after 65 ns, '0'
        after 70 ns, '1' after 75 ns, '0' after 80 ns, '1' after 85 ns, '0' after 90 ns,
        '1' after 95 ns, '0' after 100 ns, '1' after 105 ns, '0' after 110 ns, '1' after 115 ns,
        '0' after 120 ns, '1' after 125 ns, '0' after 130 ns, '1' after 135 ns,
        '0' after 140 ns, '1' after 145 ns, '0' after 150 ns, '1' after 155 ns,
        '0' after 160 ns;

        U13: full_adder_sub port map (bclk, sub, x1, x2, Cin1, Cin2, y1, y2, C1,
        C2 );
end architecture bench;
```

Joonis 34. Liitja-lahutaja 1. testpink

5.12.1.2 Teine test (x1=1)

```
use work.ternary_logic.all;

entity test_full_adder_sub is
end entity test_full_adder_sub;

architecture bench of test_full_adder_sub is

        signal x1, x2, Cin1, Cin2 : bit;
        signal y1, y2, C1, C2 : bit;
component full_adder_sub
                port( sub, bclk, x1, x2, Cin1, Cin2 : in bit;
                        y1, y2, C1, C2 : out bit);
        end component full_adder_sub;
        signal bclk: bit := '0';
        signal sub: bit := '0';
begin
        x1 <=  '1' after 0 ns;
        x2 <=  '0' after 0 ns, '1' after 80 ns, '0' after 160 ns;
        Cin1 <=  '0' after 0 ns, '1' after 40 ns, '1' after 50 ns, '1' after 70 ns, '0' after 80
        ns, '1' after 120 ns, '0' after 160 ns;
        Cin2 <=  '0' after 0 ns, '1' after 20 ns, '0' after 40 ns, '1' after 60 ns, '1' after 70
        ns, '0' after 80 ns, '1' after 100 ns, '0' after 120 ns, '1' after 140 ns,
        '0' after 160 ns;
        sub <=  '0' after 0 ns,  '1' after 10 ns,  '0' after 20 ns, '1' after 30 ns, '0' after 40
        ns, '1' after 50 ns, '0' after 60 ns, '1' after 70 ns, '0' after 80 ns,
        '1' after 90 ns, '0' after 100 ns, '1' after 110 ns, '0' after 120 ns, '1' after 130 ns,
        '0' after 140 ns, '1' after 150 ns, '0' after 160 ns;
        bclk <=  '0' after 0 ns,  '1' after 5 ns,  '0' after 10 ns,  '1' after 15 ns,  '0' after 20
        ns, '1' after 25 ns, '0' after 30 ns, '1' after 35 ns, '0' after 40 ns,
        '1' after 45 ns, '0' after 50 ns, '1' after 55 ns, '0' after 60 ns, '1' after 65 ns, '0'
        after 70 ns, '1' after 75 ns, '0' after 80 ns, '1' after 85 ns, '0' after 90 ns,
        '1' after 95 ns, '0' after 100 ns, '1' after 105 ns, '0' after 110 ns, '1' after 115 ns,
        '0' after 120 ns, '1' after 125 ns, '0' after 130 ns, '1' after 135 ns,
        '0' after 140 ns, '1' after 145 ns, '0' after 150 ns, '1' after 155 ns,
        '0' after 160 ns;

        U13: full_adder_sub port map (bclk, sub, x1, x2, Cin1, Cin2, y1, y2, C1,
        C2 );
end architecture bench;
```

Joonis 35. Liitja-lahutaja 2. testpink

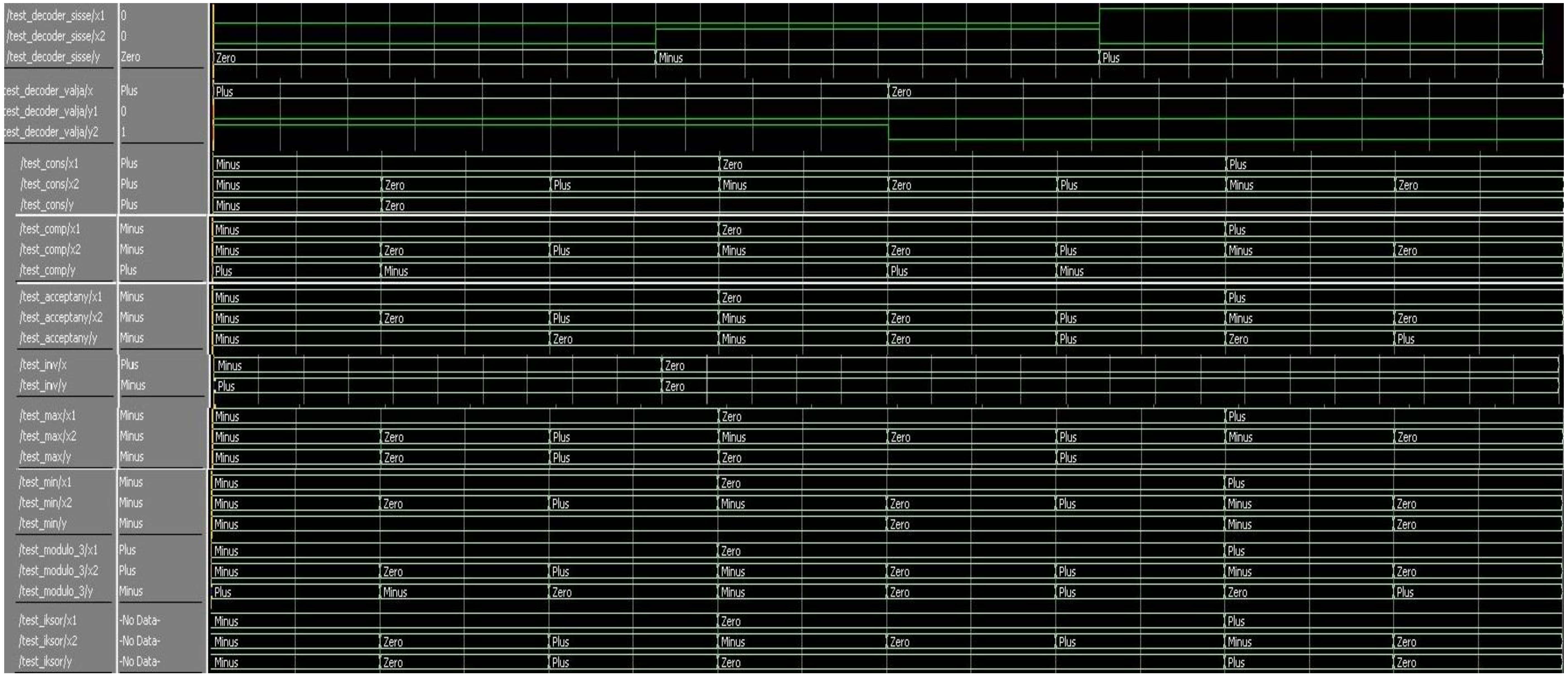
6 Signaalid

Signaalid on saadud VHDL-i koodi simuleerimisel ModelSIM tarkvaraga. Signaalide piltides on näha nii 3-nd süsteemi väljundeid kui ka 2-nd süsteemi väljundeid.

6.1 Eraldi seisvad lülitused

Signaalid 1-3kuuluvad sisendite *decoder*-ile – loogika leheküljel 15, kood leheküljel 27, testpink leheküljel 44;
Signaalid 4-6 väljundite *decoder*-ile – loogika leheküljel 16, kood leheküljel 28, testpinkleheküljel 45;

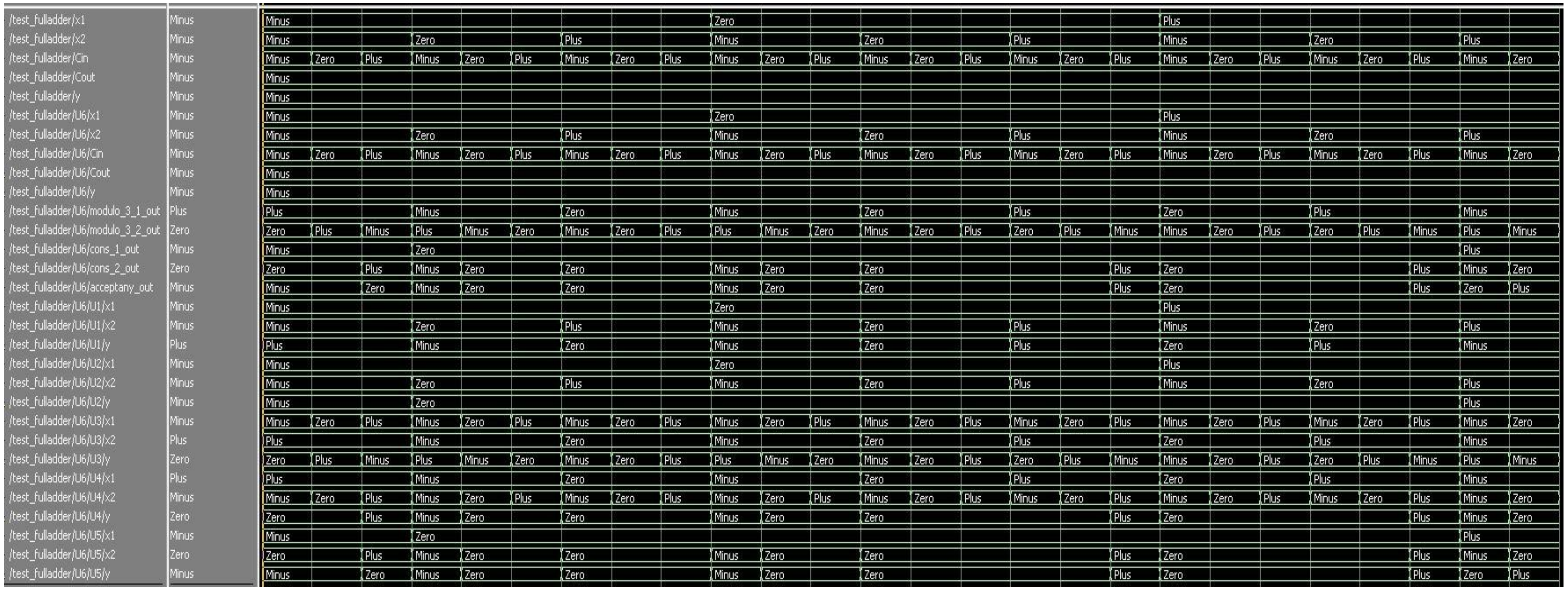
Signaalid 7-9 *consensus* – loogika leheküljel 19, kood leheküljel 35, testpink leheküljel 52;
Signaalid 10-12 *comparator* – loogika leheküljel 18, kood leheküljel 33, testpink leheküljel 50;
Signaalid13-15 *accept anything* – loogika leheküljel 20, kood leheküljel 32, testpink leheküljel 49;
Signaalid 16-17 inverter – loogika leheküljel 18, kood leheküljel 36, testpink leheküljel 53;
Signaalid 18-20 *maximum* – loogika leheküljel 17, kood leheküljel 31, testpink leheküljel 48;
Signaalid 20-22 *minimum* – loogika leheküljel 17, kood leheküljel 30, testpink leheküljel 47;
Signaalid 22-24 *modulo-3* – loogika leheküljel 19, kood leheküljel 34, testpink leheküljel 51;
Signaalid 25-27 *XOR* – loogika leheküljel 16, kood leheküljel 29, testpink leheküljel 46.



Joonis 36. Eraldi seivate lülituste signaalid

6.2 Liitja

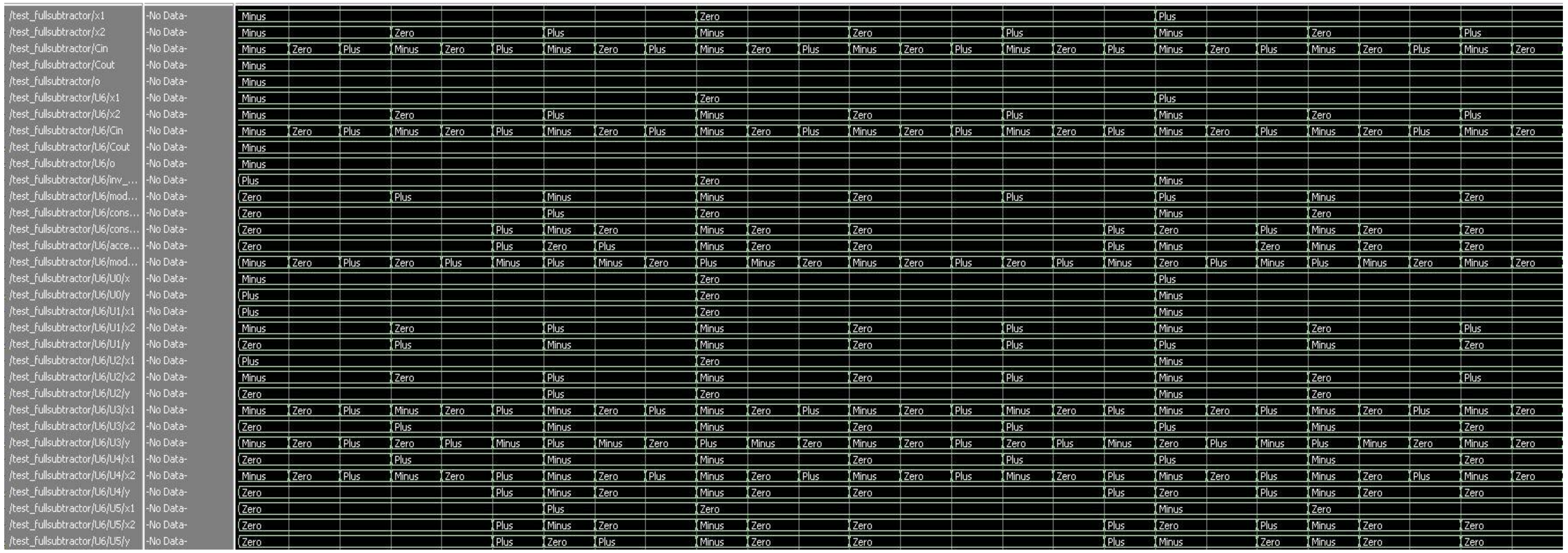
Liitja loogika leheküljel 20, kood leheküljel 37, testpink leheküljel 54


/test_fulladder/x1 Minus
/test_fulladder/x2 Minus
/test_fulladder/Cin Minus
/test_fulladder/Cout Minus
/test_fulladder/y Minus
/test_fulladder/U6/x1 Minus
/test_fulladder/U6/x2 Minus
/test_fulladder/U6/Cin Minus
/test_fulladder/U6/Cout Minus
/test_fulladder/U6/y Minus
/test_fulladder/U6/modulo_3_1_out Plus
/test_fulladder/U6/modulo_3_2_out Zero
/test_fulladder/U6/cons_1_out Minus
/test_fulladder/U6/cons_2_out Zero
/test_fulladder/U6/acceptany_out Minus
/test_fulladder/U6/U1/x1 Minus
/test_fulladder/U6/U1/x2 Minus
/test_fulladder/U6/U1/y Plus
/test_fulladder/U6/U2/x1 Minus
/test_fulladder/U6/U2/x2 Minus
/test_fulladder/U6/U2/y Minus
/test_fulladder/U6/U3/x1 Minus
/test_fulladder/U6/U3/x2 Plus
/test_fulladder/U6/U3/y Zero
/test_fulladder/U6/U4/x1 Plus
/test_fulladder/U6/U4/x2 Minus
/test_fulladder/U6/U4/y Zero
/test_fulladder/U6/U5/x1 Minus
/test_fulladder/U6/U5/x2 Zero
/test_fulladder/U6/U5/y Minus

Joonis 37. Liitja signaalid

6.3 Lahutaja

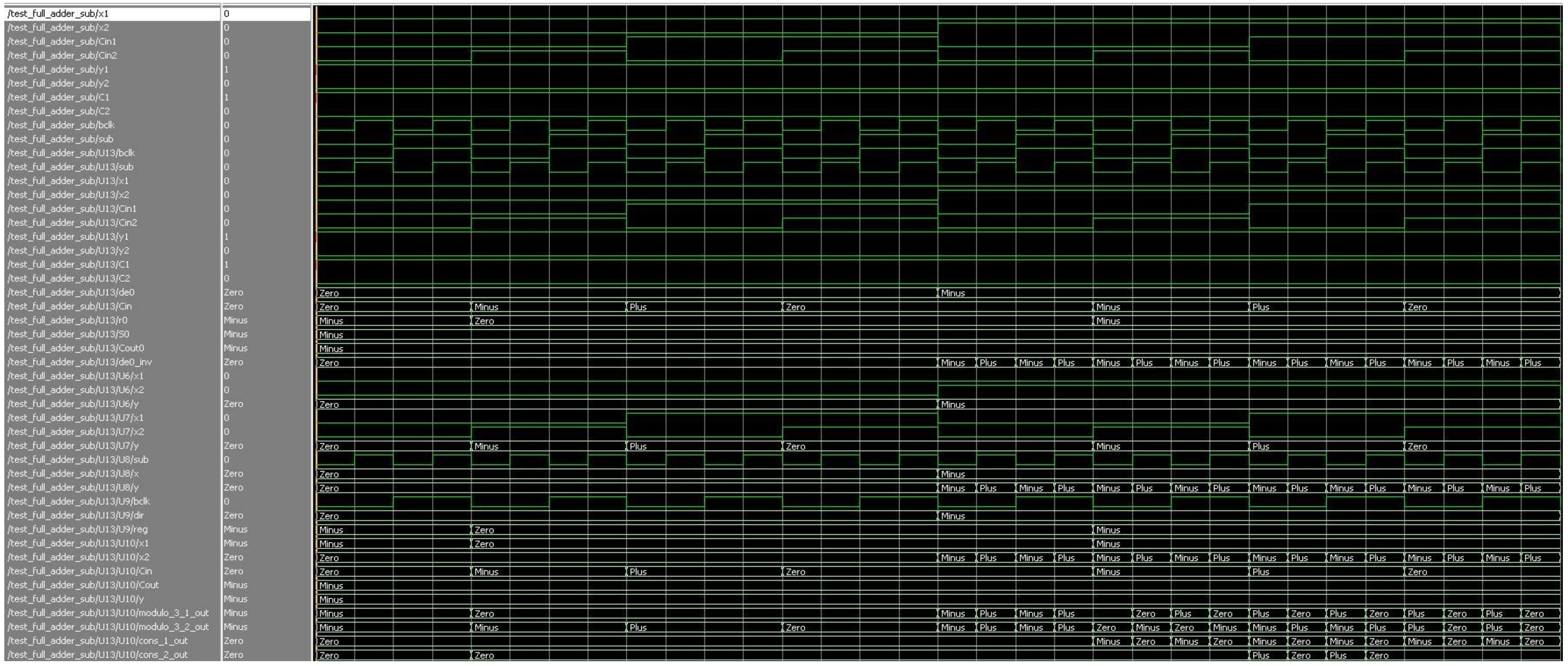
Lahutaja loogika leheküljel 21, kood leheküljel 38, testpink leheküljel 55.


/test_fullsubtractor/x1
/test_fullsubtractor/x2
/test_fullsubtractor/Cin
/test_fullsubtractor/Cout
/test_fullsubtractor/o
/test_fullsubtractor/U6/x1
/test_fullsubtractor/U6/x2
/test_fullsubtractor/U6/Cin
/test_fullsubtractor/U6/Cout
/test_fullsubtractor/U6/o
/test_fullsubtractor/U6/inv_...
/test_fullsubtractor/U6/mod...
/test_fullsubtractor/U6/cons...
/test_fullsubtractor/U6/cons...
/test_fullsubtractor/U6/acce...
/test_fullsubtractor/U6/mod...
/test_fullsubtractor/U6/U0/x
/test_fullsubtractor/U6/U0/y
/test_fullsubtractor/U6/U1/x1
/test_fullsubtractor/U6/U1/x2
/test_fullsubtractor/U6/U1/y
/test_fullsubtractor/U6/U2/x1
/test_fullsubtractor/U6/U2/x2
/test_fullsubtractor/U6/U2/y
/test_fullsubtractor/U6/U3/x1
/test_fullsubtractor/U6/U3/x2
/test_fullsubtractor/U6/U3/y
/test_fullsubtractor/U6/U4/x1
/test_fullsubtractor/U6/U4/x2
/test_fullsubtractor/U6/U4/y
/test_fullsubtractor/U6/U5/x1
/test_fullsubtractor/U6/U5/x2
/test_fullsubtractor/U6/U5/y
-No Data-
Minus
Zero
Plus

Joonis 38. Lahutaja signaalid

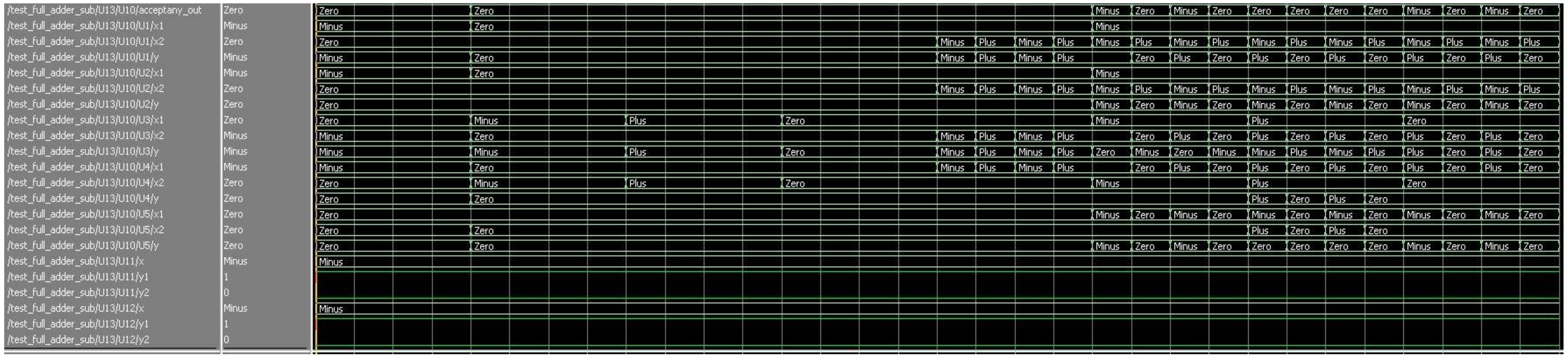
6.4 Liitja-lahutaja

Liitja-lahutaja loogika leheküljel 22, kood leheküljel 42, testpink leheküljel 56.



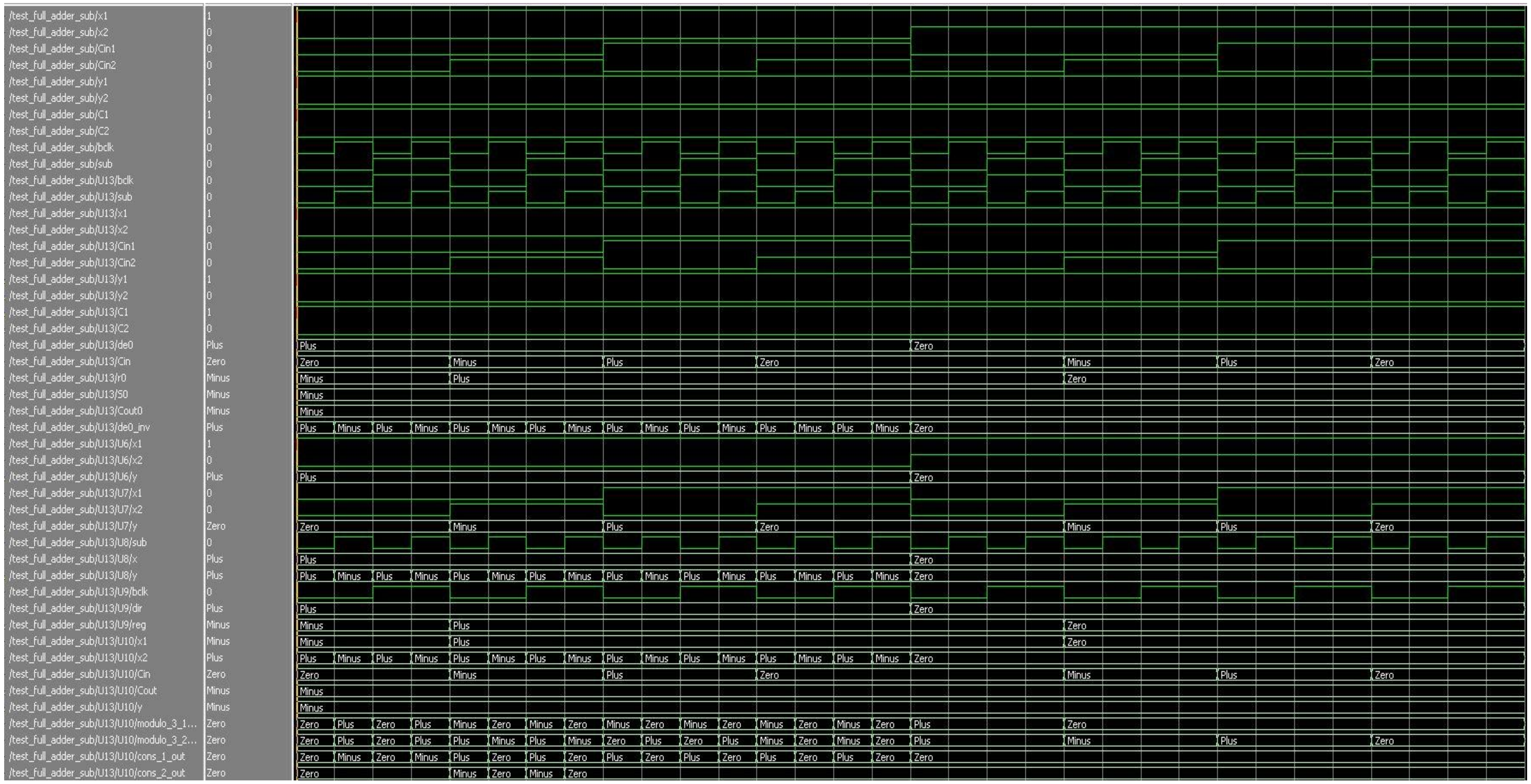
```
/test_full_adder_sub/x1                         0
/test_full_adder_sub/x2                         0
/test_full_adder_sub/Cin1                       0
/test_full_adder_sub/Cin2                       0
/test_full_adder_sub/y1                         1
/test_full_adder_sub/y2                         0
/test_full_adder_sub/C1                         1
/test_full_adder_sub/C2                         0
/test_full_adder_sub/bclk                       0
/test_full_adder_sub/sub                        0
/test_full_adder_sub/U13/bclk                   0
/test_full_adder_sub/U13/sub                    0
/test_full_adder_sub/U13/x1                     0
/test_full_adder_sub/U13/x2                     0
/test_full_adder_sub/U13/Cin1                   0
/test_full_adder_sub/U13/Cin2                   0
/test_full_adder_sub/U13/y1                     1
/test_full_adder_sub/U13/y2                     0
/test_full_adder_sub/U13/C1                     1
/test_full_adder_sub/U13/C2                     0
/test_full_adder_sub/U13/de0                    Zero
/test_full_adder_sub/U13/Cin                    Zero
/test_full_adder_sub/U13/r0                     Minus
/test_full_adder_sub/U13/S0                     Minus
/test_full_adder_sub/U13/Cout0                  Minus
/test_full_adder_sub/U13/de0_inv                Zero
/test_full_adder_sub/U13/U6/x1                  0
/test_full_adder_sub/U13/U6/x2                  0
/test_full_adder_sub/U13/U6/y                   Zero
/test_full_adder_sub/U13/U7/x1                  0
/test_full_adder_sub/U13/U7/x2                  0
/test_full_adder_sub/U13/U7/y                   Zero
/test_full_adder_sub/U13/U8/sub                 0
/test_full_adder_sub/U13/U8/x                   Zero
/test_full_adder_sub/U13/U8/y                   Zero
/test_full_adder_sub/U13/U9/bclk                0
/test_full_adder_sub/U13/U9/dir                 Zero
/test_full_adder_sub/U13/U9/reg                 Minus
/test_full_adder_sub/U13/U10/x1                 Minus
/test_full_adder_sub/U13/U10/x2                 Zero
/test_full_adder_sub/U13/U10/Cin                Zero
/test_full_adder_sub/U13/U10/Cout               Minus
/test_full_adder_sub/U13/U10/y                  Minus
/test_full_adder_sub/U13/U10/modulo_3_1_out     Minus
/test_full_adder_sub/U13/U10/modulo_3_2_out     Minus
/test_full_adder_sub/U13/U10/cons_1_out         Zero
/test_full_adder_sub/U13/U10/cons_2_out         Zero
```

Joonis 39. Liitja-lahutaja signaalid 1



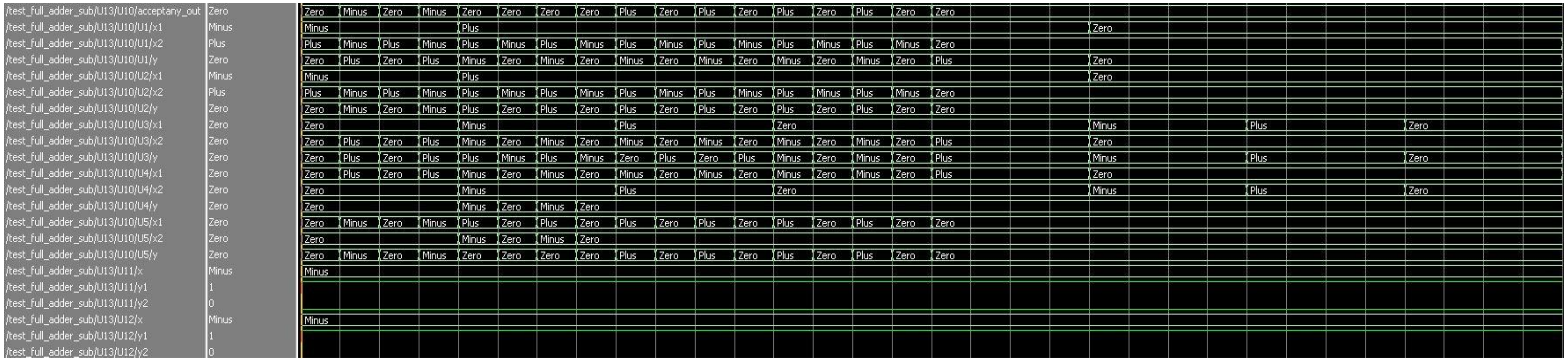
```
/test_full_adder_sub/U13/U10/acceptany_out   Zero
/test_full_adder_sub/U13/U10/U1/x1           Minus
/test_full_adder_sub/U13/U10/U1/x2           Zero
/test_full_adder_sub/U13/U10/U1/y            Minus
/test_full_adder_sub/U13/U10/U2/x1           Minus
/test_full_adder_sub/U13/U10/U2/x2           Zero
/test_full_adder_sub/U13/U10/U2/y            Zero
/test_full_adder_sub/U13/U10/U3/x1           Zero
/test_full_adder_sub/U13/U10/U3/x2           Minus
/test_full_adder_sub/U13/U10/U3/y            Minus
/test_full_adder_sub/U13/U10/U4/x1           Minus
/test_full_adder_sub/U13/U10/U4/x2           Zero
/test_full_adder_sub/U13/U10/U4/y            Zero
/test_full_adder_sub/U13/U10/U5/x1           Zero
/test_full_adder_sub/U13/U10/U5/x2           Zero
/test_full_adder_sub/U13/U10/U5/y            Zero
/test_full_adder_sub/U13/U11/x               Minus
/test_full_adder_sub/U13/U11/y1              1
/test_full_adder_sub/U13/U11/y2              0
/test_full_adder_sub/U13/U12/x               Minus
/test_full_adder_sub/U13/U12/y1              1
/test_full_adder_sub/U13/U12/y2              0
```

Joonis 40 . Liitja-lahutaja signaalid 2



```
/test_full_adder_sub/x1                      1
/test_full_adder_sub/x2                      0
/test_full_adder_sub/Cin1                    0
/test_full_adder_sub/Cin2                    0
/test_full_adder_sub/y1                      1
/test_full_adder_sub/y2                      0
/test_full_adder_sub/C1                      1
/test_full_adder_sub/C2                      0
/test_full_adder_sub/bclk                    0
/test_full_adder_sub/sub                     0
/test_full_adder_sub/U13/bclk                0
/test_full_adder_sub/U13/sub                 0
/test_full_adder_sub/U13/x1                  1
/test_full_adder_sub/U13/x2                  0
/test_full_adder_sub/U13/Cin1                0
/test_full_adder_sub/U13/Cin2                0
/test_full_adder_sub/U13/y1                  1
/test_full_adder_sub/U13/y2                  0
/test_full_adder_sub/U13/C1                  1
/test_full_adder_sub/U13/C2                  0
/test_full_adder_sub/U13/de0                 Plus
/test_full_adder_sub/U13/Cin                 Zero
/test_full_adder_sub/U13/r0                  Minus
/test_full_adder_sub/U13/S0                  Minus
/test_full_adder_sub/U13/Cout0               Minus
/test_full_adder_sub/U13/de0_inv             Plus
/test_full_adder_sub/U13/U6/x1               1
/test_full_adder_sub/U13/U6/x2               0
/test_full_adder_sub/U13/U6/y                Plus
/test_full_adder_sub/U13/U7/x1               0
/test_full_adder_sub/U13/U7/x2               0
/test_full_adder_sub/U13/U7/y                Zero
/test_full_adder_sub/U13/U8/sub              0
/test_full_adder_sub/U13/U8/x                Plus
/test_full_adder_sub/U13/U8/y                Plus
/test_full_adder_sub/U13/U9/bclk             0
/test_full_adder_sub/U13/U9/dir              Plus
/test_full_adder_sub/U13/U9/reg              Minus
/test_full_adder_sub/U13/U10/x1              Minus
/test_full_adder_sub/U13/U10/x2              Plus
/test_full_adder_sub/U13/U10/Cin             Zero
/test_full_adder_sub/U13/U10/Cout            Minus
/test_full_adder_sub/U13/U10/y               Minus
/test_full_adder_sub/U13/U10/modulo_3_1...   Zero
/test_full_adder_sub/U13/U10/modulo_3_2...   Zero
/test_full_adder_sub/U13/U10/cons_1_out      Zero
/test_full_adder_sub/U13/U10/cons_2_out      Zero
```

Joonis 41 . Liitja-lahutaja signaalid 3


```
/test_full_adder_sub/U13/U10/acceptany_out  Zero
/test_full_adder_sub/U13/U10/U1/x1          Minus
/test_full_adder_sub/U13/U10/U1/x2          Plus
/test_full_adder_sub/U13/U10/U1/y           Zero
/test_full_adder_sub/U13/U10/U2/x1          Minus
/test_full_adder_sub/U13/U10/U2/x2          Plus
/test_full_adder_sub/U13/U10/U2/y           Zero
/test_full_adder_sub/U13/U10/U3/x1          Zero
/test_full_adder_sub/U13/U10/U3/x2          Zero
/test_full_adder_sub/U13/U10/U3/y           Zero
/test_full_adder_sub/U13/U10/U4/x1          Zero
/test_full_adder_sub/U13/U10/U4/x2          Zero
/test_full_adder_sub/U13/U10/U4/y           Zero
/test_full_adder_sub/U13/U10/U5/x1          Zero
/test_full_adder_sub/U13/U10/U5/x2          Zero
/test_full_adder_sub/U13/U10/U5/y           Zero
/test_full_adder_sub/U13/U11/x              Minus
/test_full_adder_sub/U13/U11/y1             1
/test_full_adder_sub/U13/U11/y2             0
/test_full_adder_sub/U13/U12/x              Minus
/test_full_adder_sub/U13/U12/y1             1
/test_full_adder_sub/U13/U12/y2             0
```

Zero Minus Zero Minus Zero Zero Zero Zero Plus Zero Plus Zero Plus Zero Plus Zero Zero
Minus Plus Zero
Plus Minus Plus Minus Plus Minus Plus Minus Plus Minus Plus Minus Plus Minus Plus Minus Zero
Zero Plus Zero Plus Minus Zero Minus Zero Minus Zero Minus Zero Minus Zero Minus Zero Plus Zero
Minus Plus Zero
Plus Minus Plus Minus Plus Minus Plus Minus Plus Minus Plus Minus Plus Minus Plus Minus Zero
Zero Minus Zero Minus Plus Zero Plus Zero Plus Zero Plus Zero Plus Zero Plus Zero Zero
Zero Minus Plus Zero Minus Plus Zero
Zero Plus Zero Plus Minus Zero Minus Zero Minus Zero Minus Zero Minus Zero Minus Zero Plus Zero
Zero Plus Zero Plus Plus Minus Plus Minus Zero Plus Zero Plus Minus Zero Minus Zero Plus Minus Plus Zero
Zero Plus Zero Plus Minus Zero Minus Zero Minus Zero Minus Zero Minus Zero Minus Zero Plus Zero
Zero Minus Plus Zero Minus Plus Zero
Zero Minus Zero Minus Zero
Zero Minus Zero Minus Plus Zero Plus Zero Plus Zero Plus Zero Plus Zero Plus Zero Zero
Zero Minus Zero Minus Zero
Zero Minus Zero Minus Zero Zero Zero Zero Plus Zero Plus Zero Plus Zero Plus Zero Zero
Minus
Minus

Joonis 42. Liitja-lahutaja signaalid 4

7 Kokkuvõte

Töö käigus loodi kolmendsüsteemi liitja-lahutaja ning testiti FPGA Nexys-3 plaadil, millel on võimalik anda lülititega kahendkoodis sisendid ja vastavatele nuppudele vajutades, saada väärtus LED-idel. Väljundiks saab kas summa või vahe, samuti saab vaadata väärtust lisades ning vähendades ülekannet. Tööd oleks võimalik kindlasti edasi arendada, luues näiteks täieliku aritmeetika-loogikaploki ning samuti muuta realisatsiooni plaadil, näiteks kuvada väärtust 7-segmendilisel LED numberindikaatoril. Programm töötab täielikult, seega võib töö eesmärki lugeda õnnestunuks.

Kasutatud kirjandus

1. [WWW] https://en.wikipedia.org/wiki/Three-valued_logic (16.05.2017)

2. [WWW] https://en.wikipedia.org/wiki/Balanced_ternary (16.05.2017)

3. [WWW] https://en.wikipedia.org/wiki/Ternary_computer (16.05.2017)

4. [WWW] https://en.wikipedia.org/wiki/Thomas_Fowler_(inventor) (16.05.2017)

5. [WWW] https://en.wikipedia.org/wiki/Setun (16.05.2017)

6. [WWW] http://www.computer-museum.ru/english/setun.htm (16.05.2017)

7. [WWW] http://homepage.divms.uiowa.edu/~jones/ternary/ (16.05.2017)

8. Lensen, Harri, Kruus, Margus, Diskreetne matemaatika, 2012

9. Ariste, Andri, Loogikalülituste koostamise metoodika, 1978

10. [WWW] https://reference.digilentinc.com/_media/nexys:nexys3:nexys3_rm.pdf (16.05.2017)

11. [WWW] https://reference.digilentinc.com/_media/nexys:nexys3:nexys3_sch.pdf (16.05.2017)

12. [WWW] http://mini.pld.ttu.ee/~lrv/IAY0150/ (16.05.2017)

13. [WWW] http://www.ashenden.com.au/students-guide/SG.html (16.05.2017)

14. [WWW] http://www.ics.uci.edu/~jmoorkan/vhdlref/ (16.05.2017)

Lisa 1 – Tasakaalustatud 4-järgulise täisliitja-lahutaja VHDL

```
package ternary_logic is
  type ternary is (Minus, Zero, Plus);
end package ternary_logic;
use work.ternary_logic.all;
entity decoder_sisse is
  port ( x, y: in bit;
         a: out ternary);
end entity decoder_sisse;
architecture bhv of decoder_sisse is
begin
  process (x, y) begin
    case x is
       when '0'=>
                    case y is
                            when '0' => a <= Zero;
                            when '1' => a <= Minus;
                    end case;
       when '1'=>
                    case y is
                            when '0' => a <= Plus;
                            when '1' => a <= Zero;
                    end case;
    end case;
  end process;
end architecture bhv;
use work.ternary_logic.all;
entity inv is
  port ( sub: in bit;
           a: in ternary;
         o: out ternary );
end entity inv;
architecture bhv of inv is
begin
  process (sub, a) begin
    case sub is
          when '1' =>
                    case a is
```

Joonis 43. Tasakaalustatud 4-järgulise täisliitja-lahutaja kood 1

```
when Plus   =>  o <= Minus;
                                    when Minus  =>  o <= Plus;
                                    when others =>  o <= Zero;
                        end case;
            when  '0' =>
                        case a is
                                                when Plus   =>  o <= Plus;
                                                when Minus  =>  o <= Minus;
                                                when others =>  o <= Zero;
                        end case;
   end case;
  end process;
end architecture bhv;
-------------------------------------------
use work.ternary_logic.all;
entity modulo_3 is
  port ( a, b: in ternary;
         o: out ternary );
end entity modulo_3;
architecture bhv of modulo_3 is
begin
  process (a, b) begin
    case a is
    when Plus=>
                        case b is
                                    when Plus => o <= Minus;
                                    when Zero => o <= Plus;
                                    when Minus => o <= Zero;
                                    end case;
    when Zero=>
                        case b is
                                    when Plus => o <= Plus;
                                    when Zero => o <= Zero;
                                    when Minus => o <= Minus;
                                    end case;
    when Minus=>
                        case b is
                                    when Plus => o <= Zero;
                                    when Zero => o <= Minus;
                                    when Minus => o <= Plus;
                                    end case;
when others => o <= Zero;
    end case;
  end process;
end architecture bhv;
```

Joonis 44 . Tasakaalustatud 4-järgulise täisliitja-lahutaja kood   2

```
use work.ternary_logic.all;
entity cons is
  port ( a, b : in ternary;
         o: out ternary );
end entity cons;
architecture bhv of cons is
begin
  process (a, b) begin
    case a is
    when Plus=>
                    case b is
                                when Plus => o <= Plus;
                                when Zero => o <= Zero;
                                when Minus => o <= Zero;
                                end case;
    when Zero=>
                    case b is
                                when Plus => o <= Zero;
                                when Zero => o <= Zero;
                                when Minus => o <= Zero;
                                end case;
    when Minus=>
                    case b is
                                when Plus => o <= Zero;
                                when Zero => o <= Zero;
                                when Minus => o <= Minus;
                                end case;
    when others => o <= Zero;
    end case;
  end process;
end architecture bhv;
---------------------------------------------
use work.ternary_logic.all;
entity acceptany is
  port ( a, b: in ternary;
         o: out ternary );
end entity acceptany;
architecture bhv of acceptany is
begin
  process (a, b) begin
    case a is
    when Plus=>
                    case b is
                                when Plus => o <= Plus;
                                when Zero => o <= Plus;
                                when Minus => o <= Zero;
                                end case;
```

Joonis 45. Tasakaalustatud 4-järgulise täisliitja-lahutaja kood 3

```
when Zero=>
                case b is
                        when Plus => o <= Plus;
                        when Zero => o <= Zero;
                        when Minus => o <= Minus;
                        end case;
   when Minus=>
                case b is
                        when Plus => o <= Zero;
                        when Zero => o <= Minus;
                        when Minus => o <= Minus;
                        end case;
   when others => o <= Zero;
   end case;
  end process;
end architecture bhv;
-----------------------------------------
use work.ternary_logic.all;
entity decoder_valja is
  port ( a: in ternary;
       xm, xp: out bit);
end entity decoder_valja;
architecture bhv of decoder_valja is
begin
  process (a) begin
   case a is
                when Plus=> xp <= '1';
                           xm <= '0';

                when Zero=> xp <= '0';
                           xm <= '0';

                when Minus=> xp <= '0';
                            xm <= '1';
   end case;
  end process;
end architecture bhv;
use work.ternary_logic.all;
entity fullAdder is
   port( a, b, Cin : in ternary;
          Cout, S : out ternary);
end entity fullAdder;
architecture bhv of fullAdder is
component modulo_3 is
         port (a, b: in ternary;
             o: out ternary);
end component;
```

Joonis 46. Tasakaalustatud 4-järgulise täisliitja-lahutaja kood 4

```
component cons is
        port (a, b: in ternary;
            o: out ternary);
        end component;
component acceptany is
        port (a, b: in ternary;
            o: out ternary);
        end component;

        signal modulo_3_1, cons_1, cons_2 : ternary;
        begin
        U0 : modulo_3 port map (a, b, modulo_3_1);
        U1 : cons port map (a, b, cons_1);
        U2 : modulo_3 port map (Cin, modulo_3_1, S);
        U3 : cons port map (modulo_3_1, Cin, cons_2);
        U4 : acceptany port map (cons_1, cons_2, Cout);
end architecture bhv;
use work.ternary_logic.all;
entity register1 is
        port( bclk : in bit;
                 dir : in ternary;
                 reg : out ternary);
end entity register1;
use work.ternary_logic.all;
architecture bhv of register1 is
begin
        process (bclk) begin
                if bclk = '0' and bclk'event then
                        reg <= dir;
                end if;
        end process;
end architecture bhv;
use work.ternary_logic.all;
entity full_adder_sub is
  port( bclk, sub, S0p, S0m, S1p, S1m, S2p, S2m, S3p, S3m, Cip, Cim : in bit;
                l0m, l0p, l1m, l1p, l2m, l2p, l3m, l3p, lcp, lcm  : out bit);
end entity full_adder_sub;
use work.ternary_logic.all;
architecture bhv of full_adder_sub is
component fullAdder is
        port (a, b, Cin: in ternary;
            Cout, S: out ternary);
end component;
component inv is
        port (sub: in bit;
                a : in ternary;
             o : out ternary);
end component;
```

Joonis 47. Tasakaalustatud 4-järgulise täisliitja-lahutaja kood 5

```
component decoder_sisse is
        port (x, y: in bit;
            a : out ternary);
end component;
component decoder_valja is
        port (a: in ternary;
            xm, xp: out bit);
end component;
component register1 is
        port (bclk : in bit;
                 dir : in ternary;
                 reg : out ternary);
end component;

signal de0, de1, de2, de3, Cin, r0, r1, r2, r3,  S0, S1, S2, S3, Cout0, Cout1, Cout2,
Cout3, de0_inv, de1_inv, de2_inv, de3_inv : ternary;

begin

U5 : decoder_sisse port map (S0p, S0m, de0);
U6 : decoder_sisse port map (S1p, S1m, de1);
U7 : decoder_sisse port map (S2p, S2m, de2);
U8 : decoder_sisse port map (S3p, S3m, de3);
U9 : decoder_sisse port map (Cip, Cim, Cin);
U10 : inv port map (sub, de0, de0_inv);
U11 : inv port map (sub, de1, de1_inv);
U12 : inv port map (sub, de2, de2_inv);
U13 : inv port map (sub, de3, de3_inv);
U14: register1 port map (bclk, de0, r0);
U15: register1 port map (bclk, de1, r1);
U16: register1 port map (bclk, de2, r2);
U17: register1 port map (bclk, de3, r3);
U18: fullAdder port map (r0, de0_inv, Cin, Cout0, S0);
U19: fullAdder port map (r1, de1_inv, Cout0, Cout1, S1);
U20: fullAdder port map (r2, de2_inv, Cout1, Cout2, S2);
U21: fullAdder port map (r3, de3_inv, Cout2, Cout3, S3);
U22: decoder_valja port map (S0, l0m, l0p);
U23: decoder_valja port map (S1, l1m, l1p);
U24: decoder_valja port map (S2, l2m, l2p);
U25: decoder_valja port map (S3, l3m, l3p);
U26: decoder_valja port map (Cout3, lcp, lcm);

end architecture bhv;
```

Joonis 48. Tasakaalustatud 4-järgulise täisliitja-lahutaja kood 6

Lisa 2 – Tasakaalustatud 4-järgulise täisliitja-lahutaja test

```
entity test_full_adder_sub is
end entity test_full_adder_sub;

architecture bench of test_full_adder_sub is

        signal S0p, S0m, S1p, S1m, S2p, S2m, S3p, S3m, Cip, Cim : bit;
        signal l0m, l0p, l1m, l1p, l2m, l2p, l3m, l3p, lcp, lcm : bit;

        component full_adder_sub
                port( bclk, sub, S0p, S0m, S1p, S1m, S2p, S2m, S3p, S3m, Cip, Cim :
                in bit;
                l0m, l0p, l1m, l1p, l2m, l2p, l3m, l3p, lcp, lcm : out bit);
        end component full_adder_sub;

        signal bclk: bit := '0';
        signal sub: bit := '0';
        begin
        process begin
                S0p <= '0';
                S0m <= '0';
                S1p <= '0';
                S1m <= '1';
                S2p <= '0';
                S2m <= '0';
                S3p <= '1';
                S3m <= '0';
                Cip <= '1';
                Cim <= '0';
                bclk <= '0' after 0 ns, '1' after 10 ns, '0' after 20 ns;
                wait for 40 ns;
                S0p <= '1';
                S0m <= '0';
                S1p <= '1';
                S1m <= '0';
                S2p <= '1';
                S2m <= '0';
                S3p <= '1';
                S3m <= '0';
                Cip <= '0';
                Cim <= '1';
```

Joonis 49.Tasakaalustatud 4-järgulise täisliitja-lahutaja ja testpink 1

```
		bclk <= '0' after 0 ns, '1' after 50 ns, '0' after 60 ns;
		wait for 40 ns;

		S0p <= '0';
		S0m <= '0';
		S1p <= '0';
		S1m <= '1';
		S2p <= '0';
		S2m <= '0';
		S3p <= '1';
		S3m <= '0';
		Cip <= '1';
		Cim <= '0';

		sub <= '0' after 0 ns, '1' after 10 ns, '0' after 20 ns;
		bclk <= '0' after 0 ns, '1' after 10 ns, '0' after 20 ns;
		wait for 40 ns;

		S0p <= '1';
		S0m <= '0';
		S1p <= '1';
		S1m <= '0';
		S2p <= '1';
		S2m <= '0';
		S3p <= '1';
		S3m <= '0';
		Cip <= '0';
		Cim <= '1';

		sub <= '0' after 0 ns, '1' after 10 ns, '0' after 20 ns;
		bclk <= '0' after 0 ns, '1' after 50 ns, '0' after 60 ns;
		wait for 40 ns;
	end process;


	U26: full_adder_sub port map (bclk, sub, S0p, S0m, S1p, S1m, S2p, S2m, S3p,
	S3m, Cip, Cim, l0m,  l0p, l1m, l1p, l2m, l2p, l3m, l3p, lcp, lcm);
end architecture bench;
```

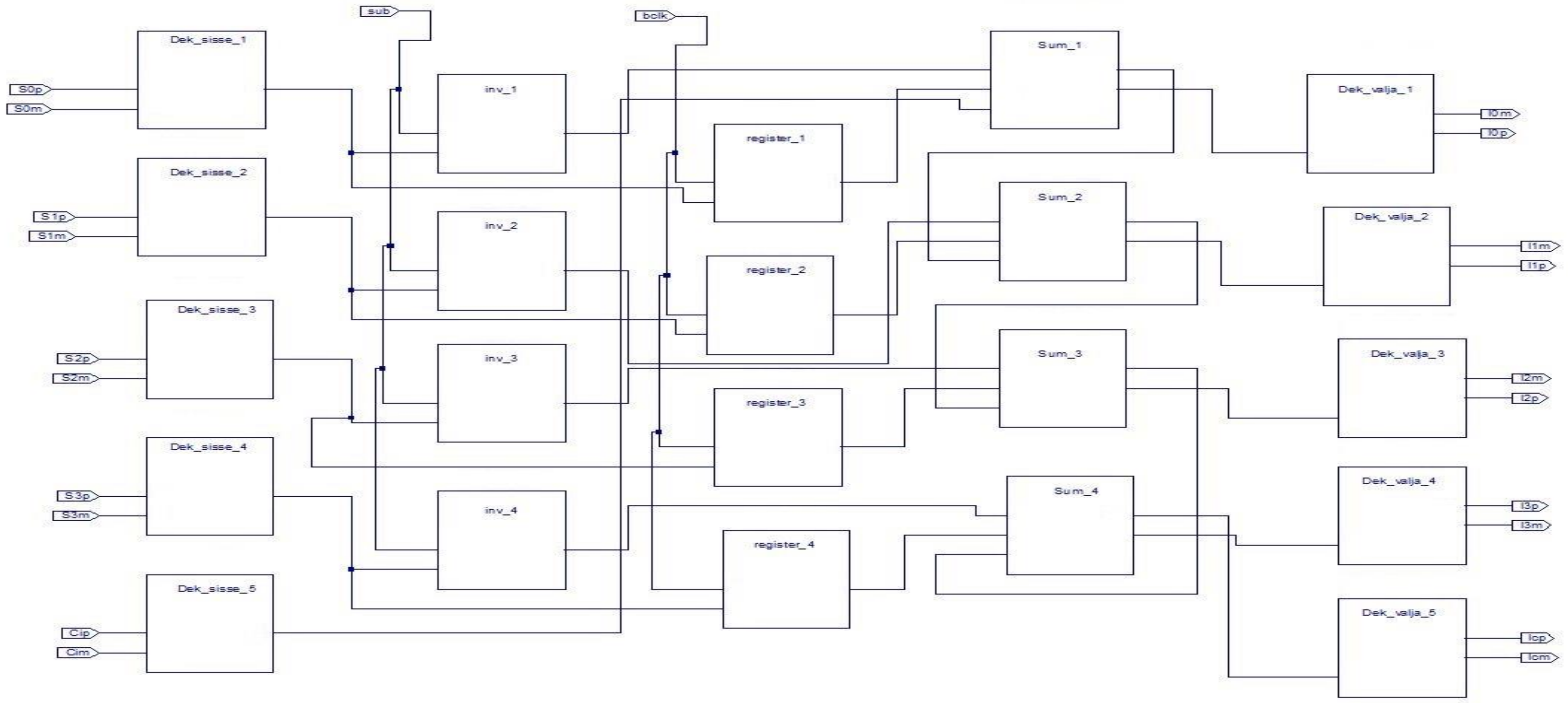
Joonis 50. Tasakaalustatud 4-järgulise täisliitja-lahutaja ja testpink 2

Lisa 3 – Tasakaalustatud 4-järgulise täisliitja-lahutaja Nexys-3 seadistus

```
NET "l0m"        LOC = "U16" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin
name = IO_L2P_CMPCLK,                    Sch name = LD0
NET "l0p"        LOC = "V16" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin
name = IO_L2N_CMPMOSI,                   Sch name = LD1
NET "l1m"        LOC = "U15" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin
name = IO_L5P,                           Sch name = LD2
NET "l1p"        LOC = "V15" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin
name = IO_L5N,                           Sch name = LD3
NET "l2m"        LOC = "M11" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin
name = IO_L15P,                          Sch name = LD4
NET "l2p"        LOC = "N11" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin
name = IO_L15N,                          Sch name = LD5
NET "l3m"        LOC = "R11" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin
name = IO_L16P,                          Sch name = LD6
NET "l3p"        LOC = "T11" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin
name = IO_L16N_VREF,                     Sch name = LD7
NET "S0m"         LOC = "T10" | IOSTANDARD = "LVCMOS33";   #Bank = 2,
Pin name = IO_L29N_GCLK2,                Sch name = SW0
NET "S0p"         LOC = "T9"  | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin
name = IO_L32P_GCLK29,                   Sch name = SW1
NET "S1m"         LOC = "V9"  | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin
name = IO_L32N_GCLK28,                   Sch name = SW2
NET "S1p"         LOC = "M8"  | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin
name = IO_L40P,                          Sch name = SW3
NET "S2m"         LOC = "N8"  | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin
name = IO_L40N,                          Sch name = SW4
NET "S2p"         LOC = "U8"  | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin
name = IO_L41P,                          Sch name = SW5
NET "S3m"         LOC = "V8"  | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin
name = IO_L41N_VREF,                     Sch name = SW6
NET "S3p"         LOC = "T5"  | IOSTANDARD = "LVCMOS33";   #Bank = MISC,
Pin name = IO_L48N_RDWR_B_VREF_2,        Sch name = SW7
NET "Cip"        LOC = "A8"  | IOSTANDARD = "LVCMOS33";   #Bank = 0, Pin
name = IO_L33N,                          Sch name = BTNU
NET "bclk"        LOC = "C4"  | IOSTANDARD = "LVCMOS33";   #Bank = 0, Pin
name = IO_L1N_VREF,                      Sch name = BTNL
NET "bclk" CLOCK_DEDICATED_ROUTE = FALSE;
NET "Cim"        LOC = "C9"  | IOSTANDARD = "LVCMOS33";   #Bank = 0, Pin
name = IO_L34N_GCLK18,                   Sch name = BTND
NET "sub"        LOC = "D9"  | IOSTANDARD = "LVCMOS33";   #Bank = 0, Pin
name = IO_L34P_GCLK19,                   Sch name = BTNR
```

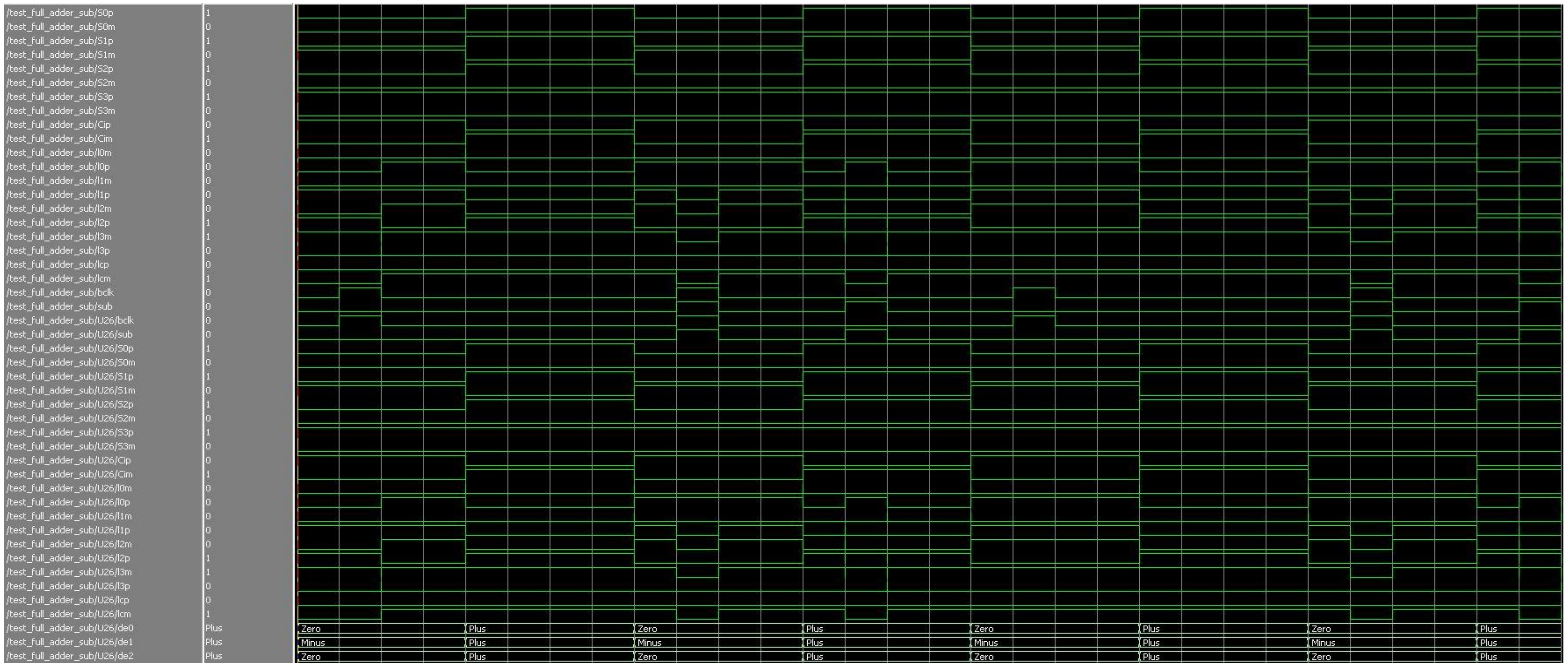
Joonis 51. Nexys-3 seadistus

Lisa 4 – Tasakaalustatud 4-järgulise täisliitja-lahutaja skeem



Joonis 52. Tasakaalustatud 4-järgulise täissumaator-lahutaja skeem

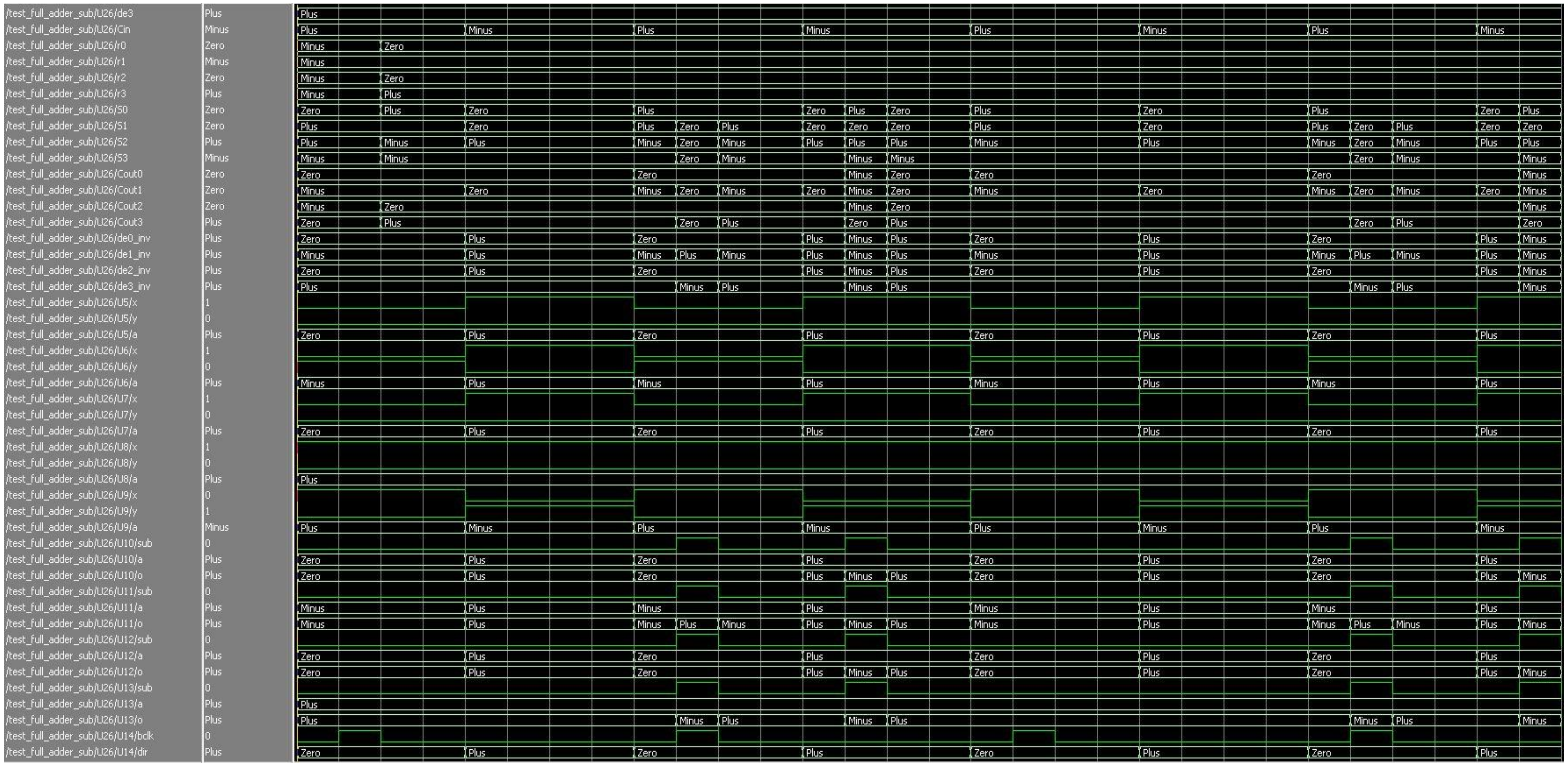
Lisa 5 – Tasakaalustatud 4-järgulise täisliitja-lahutaja signaalid valitud kombinisatsiooniga



```
/test_full_adder_sub/S0p          1
/test_full_adder_sub/S0m          0
/test_full_adder_sub/S1p          1
/test_full_adder_sub/S1m          0
/test_full_adder_sub/S2p          1
/test_full_adder_sub/S2m          0
/test_full_adder_sub/S3p          1
/test_full_adder_sub/S3m          0
/test_full_adder_sub/Cip          0
/test_full_adder_sub/Cim          1
/test_full_adder_sub/I0m          0
/test_full_adder_sub/I0p          0
/test_full_adder_sub/I1m          0
/test_full_adder_sub/I1p          0
/test_full_adder_sub/I2m          0
/test_full_adder_sub/I2p          1
/test_full_adder_sub/I3m          1
/test_full_adder_sub/I3p          0
/test_full_adder_sub/Icp          0
/test_full_adder_sub/Icm          1
/test_full_adder_sub/bclk         0
/test_full_adder_sub/sub          0
/test_full_adder_sub/U26/bclk     0
/test_full_adder_sub/U26/sub      0
/test_full_adder_sub/U26/S0p      1
/test_full_adder_sub/U26/S0m      0
/test_full_adder_sub/U26/S1p      1
/test_full_adder_sub/U26/S1m      0
/test_full_adder_sub/U26/S2p      1
/test_full_adder_sub/U26/S2m      0
/test_full_adder_sub/U26/S3p      1
/test_full_adder_sub/U26/S3m      0
/test_full_adder_sub/U26/Cip      0
/test_full_adder_sub/U26/Cim      1
/test_full_adder_sub/U26/I0m      0
/test_full_adder_sub/U26/I0p      0
/test_full_adder_sub/U26/I1m      0
/test_full_adder_sub/U26/I1p      0
/test_full_adder_sub/U26/I2m      0
/test_full_adder_sub/U26/I2p      1
/test_full_adder_sub/U26/I3m      1
/test_full_adder_sub/U26/I3p      0
/test_full_adder_sub/U26/Icp      0
/test_full_adder_sub/U26/Icm      1
/test_full_adder_sub/U26/de0      Plus
/test_full_adder_sub/U26/de1      Plus
/test_full_adder_sub/U26/de2      Plus
```

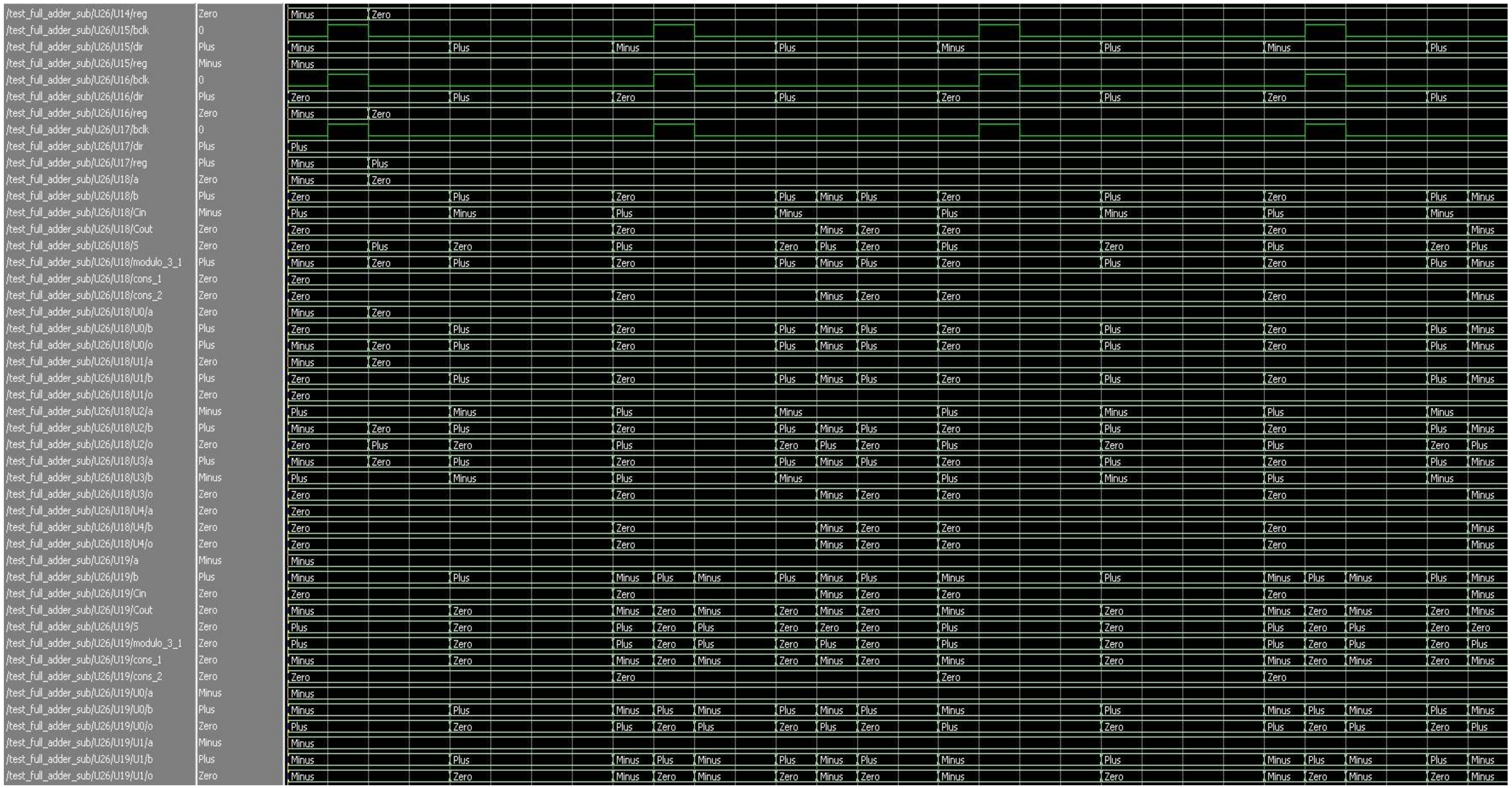
Zero Plus Zero Plus Zero Plus Zero Plus
Minus Plus Minus Plus Minus Plus Minus Plus
Zero Plus Zero Plus Zero Plus Zero Plus

Joonis 53. Tasakaalustatud 4-järgulise täisliitja-lahutaja signaalid valitud kombinatsiooniga 1



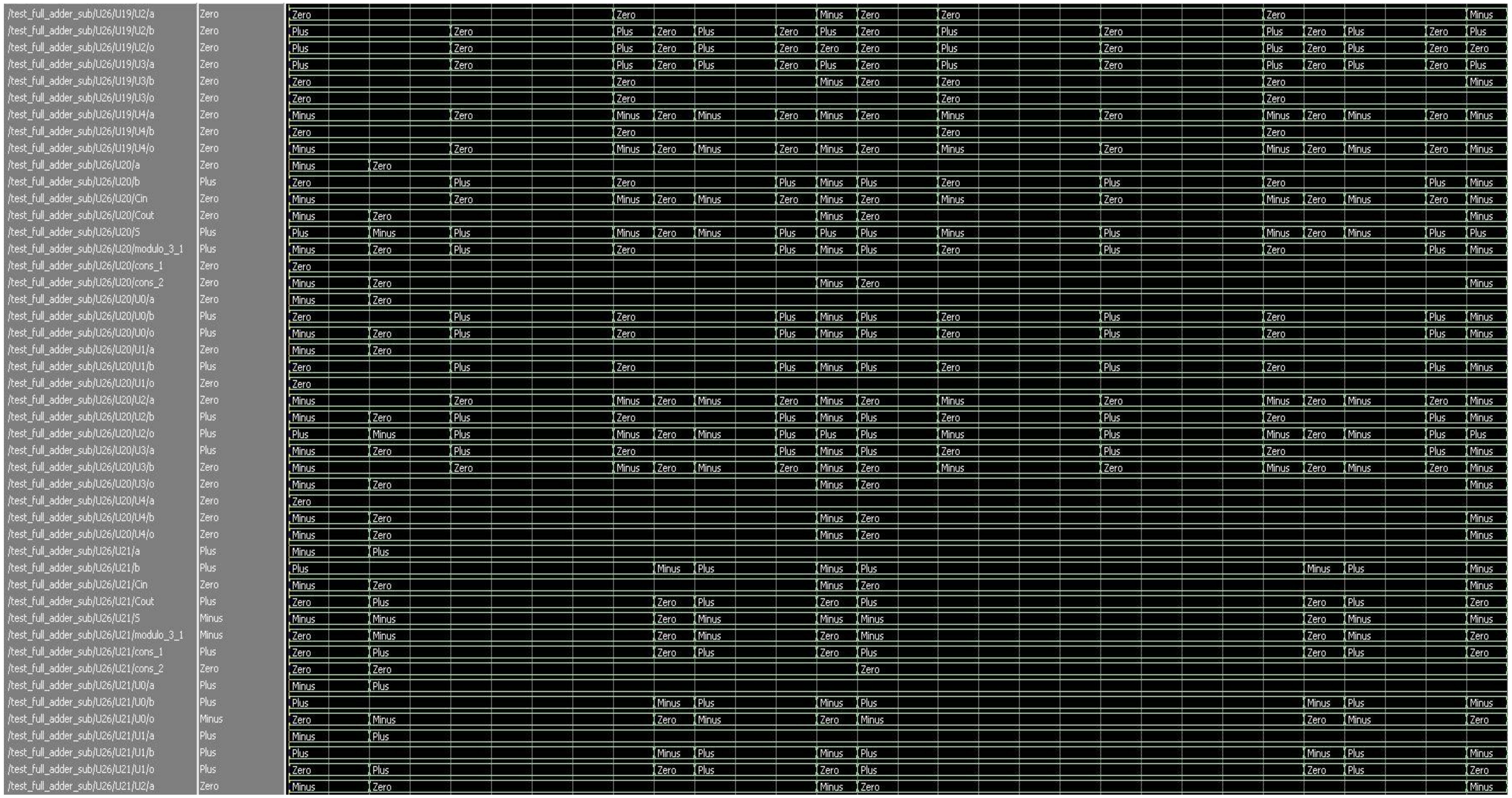
```
/test_full_adder_sub/U26/de3        Plus
/test_full_adder_sub/U26/Cin        Minus
/test_full_adder_sub/U26/r0         Zero
/test_full_adder_sub/U26/r1         Minus
/test_full_adder_sub/U26/r2         Zero
/test_full_adder_sub/U26/r3         Plus
/test_full_adder_sub/U26/S0         Zero
/test_full_adder_sub/U26/S1         Zero
/test_full_adder_sub/U26/S2         Plus
/test_full_adder_sub/U26/S3         Minus
/test_full_adder_sub/U26/Cout0      Zero
/test_full_adder_sub/U26/Cout1      Zero
/test_full_adder_sub/U26/Cout2      Zero
/test_full_adder_sub/U26/Cout3      Plus
/test_full_adder_sub/U26/de0_inv    Plus
/test_full_adder_sub/U26/de1_inv    Plus
/test_full_adder_sub/U26/de2_inv    Plus
/test_full_adder_sub/U26/de3_inv    Plus
/test_full_adder_sub/U26/U5/x       1
/test_full_adder_sub/U26/U5/y       0
/test_full_adder_sub/U26/U5/a       Plus
/test_full_adder_sub/U26/U6/x       1
/test_full_adder_sub/U26/U6/y       0
/test_full_adder_sub/U26/U6/a       Plus
/test_full_adder_sub/U26/U7/x       1
/test_full_adder_sub/U26/U7/y       0
/test_full_adder_sub/U26/U7/a       Plus
/test_full_adder_sub/U26/U8/x       1
/test_full_adder_sub/U26/U8/y       0
/test_full_adder_sub/U26/U8/a       Plus
/test_full_adder_sub/U26/U9/x       0
/test_full_adder_sub/U26/U9/y       1
/test_full_adder_sub/U26/U9/a       Minus
/test_full_adder_sub/U26/U10/sub    0
/test_full_adder_sub/U26/U10/a      Plus
/test_full_adder_sub/U26/U10/o      Plus
/test_full_adder_sub/U26/U11/sub    0
/test_full_adder_sub/U26/U11/a      Plus
/test_full_adder_sub/U26/U11/o      Plus
/test_full_adder_sub/U26/U12/sub    0
/test_full_adder_sub/U26/U12/a      Plus
/test_full_adder_sub/U26/U12/o      Plus
/test_full_adder_sub/U26/U13/sub    0
/test_full_adder_sub/U26/U13/a      Plus
/test_full_adder_sub/U26/U13/o      Plus
/test_full_adder_sub/U26/U14/bclk   0
/test_full_adder_sub/U26/U14/dir    Plus
```

Joonis 54. Tasakaalustatud 4-järgulise täisliitja-lahutaja signaalid valitud kombinatsiooniga 2



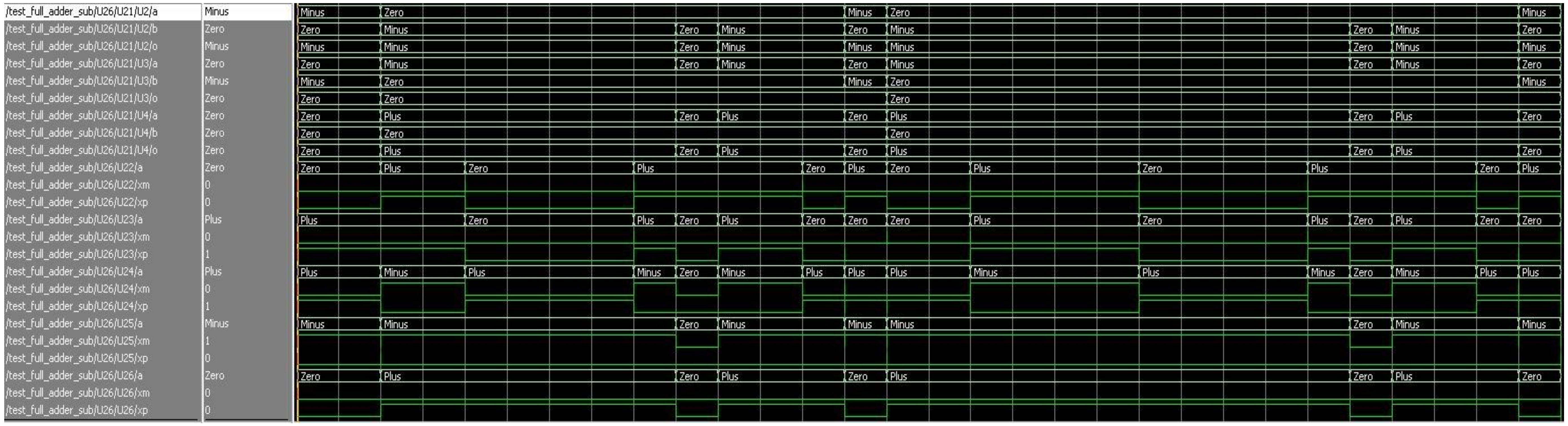
```
/test_full_adder_sub/U26/U14/reg          Zero
/test_full_adder_sub/U26/U15/bclk         0
/test_full_adder_sub/U26/U15/dir          Plus
/test_full_adder_sub/U26/U15/reg          Minus
/test_full_adder_sub/U26/U16/bclk         0
/test_full_adder_sub/U26/U16/dir          Plus
/test_full_adder_sub/U26/U16/reg          Zero
/test_full_adder_sub/U26/U17/bclk         0
/test_full_adder_sub/U26/U17/dir          Plus
/test_full_adder_sub/U26/U17/reg          Plus
/test_full_adder_sub/U26/U18/a            Zero
/test_full_adder_sub/U26/U18/b            Plus
/test_full_adder_sub/U26/U18/Cin          Minus
/test_full_adder_sub/U26/U18/Cout         Zero
/test_full_adder_sub/U26/U18/S            Zero
/test_full_adder_sub/U26/U18/modulo_3_1   Plus
/test_full_adder_sub/U26/U18/cons_1       Zero
/test_full_adder_sub/U26/U18/cons_2       Zero
/test_full_adder_sub/U26/U18/U0/a         Zero
/test_full_adder_sub/U26/U18/U0/b         Plus
/test_full_adder_sub/U26/U18/U0/o         Plus
/test_full_adder_sub/U26/U18/U1/a         Zero
/test_full_adder_sub/U26/U18/U1/b         Plus
/test_full_adder_sub/U26/U18/U1/o         Zero
/test_full_adder_sub/U26/U18/U2/a         Minus
/test_full_adder_sub/U26/U18/U2/b         Plus
/test_full_adder_sub/U26/U18/U2/o         Zero
/test_full_adder_sub/U26/U18/U3/a         Plus
/test_full_adder_sub/U26/U18/U3/b         Minus
/test_full_adder_sub/U26/U18/U3/o         Zero
/test_full_adder_sub/U26/U18/U4/a         Zero
/test_full_adder_sub/U26/U18/U4/b         Zero
/test_full_adder_sub/U26/U18/U4/o         Zero
/test_full_adder_sub/U26/U19/a            Minus
/test_full_adder_sub/U26/U19/b            Plus
/test_full_adder_sub/U26/U19/Cin          Zero
/test_full_adder_sub/U26/U19/Cout         Zero
/test_full_adder_sub/U26/U19/S            Zero
/test_full_adder_sub/U26/U19/modulo_3_1   Zero
/test_full_adder_sub/U26/U19/cons_1       Zero
/test_full_adder_sub/U26/U19/cons_2       Zero
/test_full_adder_sub/U26/U19/U0/a         Minus
/test_full_adder_sub/U26/U19/U0/b         Plus
/test_full_adder_sub/U26/U19/U0/o         Zero
/test_full_adder_sub/U26/U19/U1/a         Minus
/test_full_adder_sub/U26/U19/U1/b         Plus
/test_full_adder_sub/U26/U19/U1/o         Zero
```

Joonis 55. . Tasakaalustatud 4-järgulise täisliitja-lahutaja signaalid valitud kombinatsiooniga 3



```
/test_full_adder_sub/U26/U19/U2/a          Zero    Zero  Zero  Minus  Zero  Zero  Zero  Minus
/test_full_adder_sub/U26/U19/U2/b          Zero    Plus  Zero  Plus  Zero  Plus  Zero  Plus  Zero  Plus  Zero  Plus  Zero  Plus  Zero  Plus
/test_full_adder_sub/U26/U19/U2/o          Zero    Plus  Zero  Plus  Zero  Plus  Zero  Zero  Zero  Plus  Zero  Plus  Zero  Plus  Zero  Zero
/test_full_adder_sub/U26/U19/U3/a          Zero    Plus  Zero  Plus  Zero  Plus  Zero  Plus  Zero  Plus  Zero  Plus  Zero  Plus  Zero  Plus
/test_full_adder_sub/U26/U19/U3/b          Zero    Zero  Zero  Minus  Zero  Zero  Zero  Minus
/test_full_adder_sub/U26/U19/U3/o          Zero    Zero  Zero  Zero  Zero
/test_full_adder_sub/U26/U19/U4/a          Zero    Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus
/test_full_adder_sub/U26/U19/U4/b          Zero    Zero  Zero  Zero  Zero
/test_full_adder_sub/U26/U19/U4/o          Zero    Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus
/test_full_adder_sub/U26/U20/a             Zero    Minus  Zero
/test_full_adder_sub/U26/U20/b             Plus    Zero  Plus  Zero  Plus  Minus  Plus  Zero  Plus  Zero  Plus  Minus
/test_full_adder_sub/U26/U20/Cin           Zero    Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus
/test_full_adder_sub/U26/U20/Cout          Zero    Minus  Zero  Minus  Zero  Minus
/test_full_adder_sub/U26/U20/S             Plus    Plus  Minus  Plus  Minus  Zero  Minus  Plus  Plus  Plus  Minus  Plus  Minus  Zero  Minus  Plus  Plus
/test_full_adder_sub/U26/U20/modulo_3_1    Plus    Minus  Zero  Plus  Zero  Plus  Minus  Plus  Zero  Plus  Zero  Plus  Minus
/test_full_adder_sub/U26/U20/cons_1        Zero    Zero
/test_full_adder_sub/U26/U20/cons_2        Zero    Minus  Zero  Minus  Zero  Minus
/test_full_adder_sub/U26/U20/U0/a          Zero    Minus  Zero
/test_full_adder_sub/U26/U20/U0/b          Plus    Zero  Plus  Zero  Plus  Minus  Plus  Zero  Plus  Zero  Plus  Minus
/test_full_adder_sub/U26/U20/U0/o          Plus    Minus  Zero  Plus  Zero  Plus  Minus  Plus  Zero  Plus  Zero  Plus  Minus
/test_full_adder_sub/U26/U20/U1/a          Zero    Minus  Zero
/test_full_adder_sub/U26/U20/U1/b          Plus    Zero  Plus  Zero  Plus  Minus  Plus  Zero  Plus  Zero  Plus  Minus
/test_full_adder_sub/U26/U20/U1/o          Zero    Zero
/test_full_adder_sub/U26/U20/U2/a          Zero    Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus
/test_full_adder_sub/U26/U20/U2/b          Plus    Minus  Zero  Plus  Zero  Plus  Minus  Plus  Zero  Plus  Zero  Plus  Minus
/test_full_adder_sub/U26/U20/U2/o          Plus    Plus  Minus  Plus  Minus  Zero  Minus  Plus  Plus  Plus  Minus  Plus  Minus  Zero  Minus  Plus  Plus
/test_full_adder_sub/U26/U20/U3/a          Plus    Minus  Zero  Plus  Zero  Plus  Minus  Plus  Zero  Plus  Zero  Plus  Minus
/test_full_adder_sub/U26/U20/U3/b          Zero    Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus
/test_full_adder_sub/U26/U20/U3/o          Zero    Minus  Zero  Minus  Zero  Minus
/test_full_adder_sub/U26/U20/U4/a          Zero    Zero
/test_full_adder_sub/U26/U20/U4/b          Zero    Minus  Zero  Minus  Zero  Minus
/test_full_adder_sub/U26/U20/U4/o          Zero    Minus  Zero  Minus  Zero  Minus
/test_full_adder_sub/U26/U21/a             Plus    Minus  Plus
/test_full_adder_sub/U26/U21/b             Plus    Plus  Minus  Plus  Minus  Plus  Minus  Plus  Minus
/test_full_adder_sub/U26/U21/Cin           Zero    Minus  Zero  Minus  Zero  Minus
/test_full_adder_sub/U26/U21/Cout          Plus    Zero  Plus  Zero  Plus  Zero  Plus  Zero  Plus  Zero
/test_full_adder_sub/U26/U21/S             Minus   Minus  Minus  Zero  Minus  Minus  Minus  Zero  Minus  Minus
/test_full_adder_sub/U26/U21/modulo_3_1    Minus   Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero
/test_full_adder_sub/U26/U21/cons_1        Plus    Zero  Plus  Zero  Plus  Zero  Plus  Zero  Plus  Zero
/test_full_adder_sub/U26/U21/cons_2        Zero    Zero  Zero  Zero
/test_full_adder_sub/U26/U21/U0/a          Plus    Minus  Plus
/test_full_adder_sub/U26/U21/U0/b          Plus    Plus  Minus  Plus  Minus  Plus  Minus  Plus  Minus
/test_full_adder_sub/U26/U21/U0/o          Minus   Zero  Minus  Zero  Minus  Zero  Minus  Zero  Minus  Zero
/test_full_adder_sub/U26/U21/U1/a          Plus    Minus  Plus
/test_full_adder_sub/U26/U21/U1/b          Plus    Plus  Minus  Plus  Minus  Plus  Minus  Plus  Minus
/test_full_adder_sub/U26/U21/U1/o          Plus    Zero  Plus  Zero  Plus  Zero  Plus  Zero  Plus  Zero
/test_full_adder_sub/U26/U21/U2/a          Zero    Minus  Zero  Minus  Zero  Minus
```

Joonis 56. . Tasakaalustatud 4-järgulise täisliitja-lahutaja signaalid valitud kombinatsiooniga 4



```
/test_full_adder_sub/U26/U21/U2/a    Minus
/test_full_adder_sub/U26/U21/U2/b    Zero
/test_full_adder_sub/U26/U21/U2/o    Minus
/test_full_adder_sub/U26/U21/U3/a    Zero
/test_full_adder_sub/U26/U21/U3/b    Minus
/test_full_adder_sub/U26/U21/U3/o    Zero
/test_full_adder_sub/U26/U21/U4/a    Zero
/test_full_adder_sub/U26/U21/U4/b    Zero
/test_full_adder_sub/U26/U21/U4/o    Zero
/test_full_adder_sub/U26/U22/a       Zero
/test_full_adder_sub/U26/U22/xm      0
/test_full_adder_sub/U26/U22/xp      0
/test_full_adder_sub/U26/U23/a       Plus
/test_full_adder_sub/U26/U23/xm      0
/test_full_adder_sub/U26/U23/xp      1
/test_full_adder_sub/U26/U24/a       Plus
/test_full_adder_sub/U26/U24/xm      0
/test_full_adder_sub/U26/U24/xp      1
/test_full_adder_sub/U26/U25/a       Minus
/test_full_adder_sub/U26/U25/xm      1
/test_full_adder_sub/U26/U25/xp      0
/test_full_adder_sub/U26/U26/a       Zero
/test_full_adder_sub/U26/U26/xm      0
/test_full_adder_sub/U26/U26/xp      0
```

Joonis 57. . Tasakaalustatud 4-järgulise täisliitja-lahutaja signaalid valitud kombinatsiooniga 5

Lisa 6 – Täisliitja Karnaugh kaart

3-valentne Karnaugh kaart täisliitjas olevale summale

Tabel 16. Täisliitja summa Karnaugh kaart 3-nd süsteemis

X1 \ X2	-	0	+	-	0	+	-	0	+
-	-	0	+	0	+	-	+	-	0
0	0	+	-	+	-	0	-	0	+
+	+	-	0	-	0	+	0	+	-
	Cin = -			Cin = 0			Cin= +		

Lisa 7 – Täisliitja ülekande Karnaugh kaart

3- valentne Karnaugh kaart täisliitjas olevale ülekandele

Tabel 17. Täisliitja ülekande Karnaugh kaart 3-nd süsteemis

X1 \ X2	-	0	+	-	0	+	-	0	+
-	-	-	0	-	-	0	-	0	0
0	-	-	-	-	0	0	0	0	0
+	-	0	0	0	0	0	0	0	+
	Cin = -			Cin = 0			Cin= +		