a1.4 Aritmetičke operacije u binarnom numeričkom sistemu

Aritmetička operacija sabiranja

Analogno principu korišćenom u decimalnom sistemu sabiranja, kod sabiranja binarnih brojeva (prirodni binarni kod!) sabiranje se vrši po pozicijama bitova, pri čemu se operacija sabiranja započinje od bita na najnižoj poziciji. Ako je zbir cifara veći od jedinice onda se vrši prenos jedinice na sledeću višu poziciju. Pravila sabiranja su sledeća:

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 10 (0 + prenos 1 na višu poziciju)

Primer:

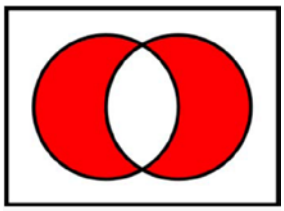
	N_2	N_{10}
	0000 1001	9
+	0000 0011	3
	0000 1100	12

Kombinaciona tablica sabirača:

A	B	S	C_izl
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

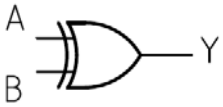
$$S = \overline{A}B + A\overline{B} = A \oplus B$$

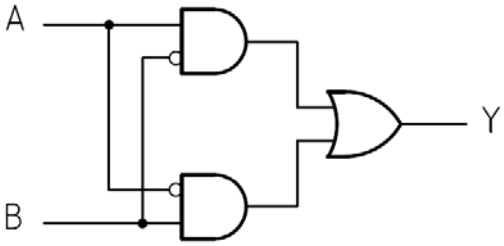
$$C_{izl} = AB$$

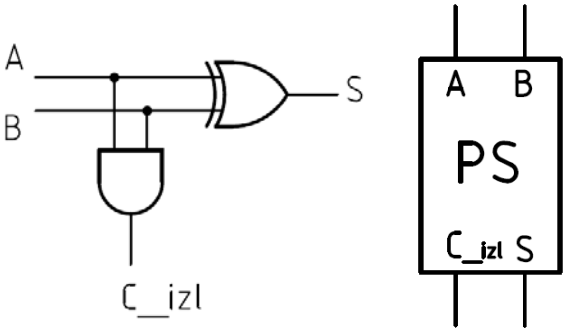


Simbol $\oplus$ (XOR) označava binarnu operaciju koja se naziva ISKLJUČIVO ILI (EKSKLUZIVNO ILI odnosno ekskluzivna disjunkcija ... jedan od dva tenisera će dobiti meč, ali ne oba!, simbol PLUS dolazi od toga da je ekskluzivna disjunkcija je sabiranje po modulu 2).

A	B	Y=A⊕B
0	0	0
0	1	1
1	0	1
1	1	0



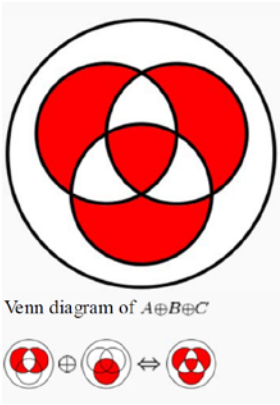




Prethodno navedena kombinaciona tablica se odnosi na slučaj sabiranja jednocifarnog binarnog broja, a funkcionalni modul koji je izvodi naziva se polusabirač (PS), (ili nepotpuni sabirač ako bi želeli da budemo do kraja jezički precizni). Da bi se operacija sabiranja realizovala sa većim brojem bitova neophodno je da se uzme u obzir bit prenosa sa niže pozicije višecifarnog binarnog broja. Tako se dolazi do potpunog sabirača (S).

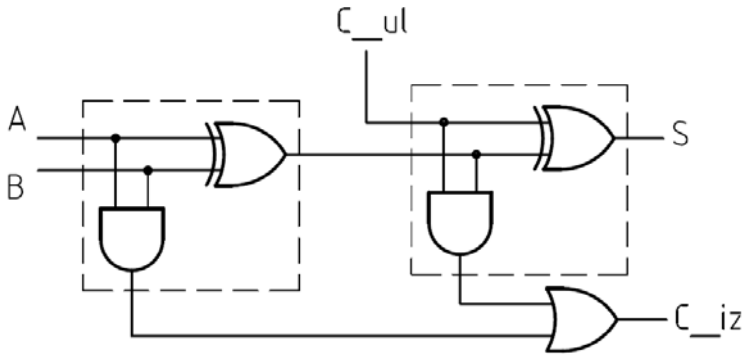
Kombinaciona tablica potpunog sabirača:

A	B	C_ul	S	C_izl
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

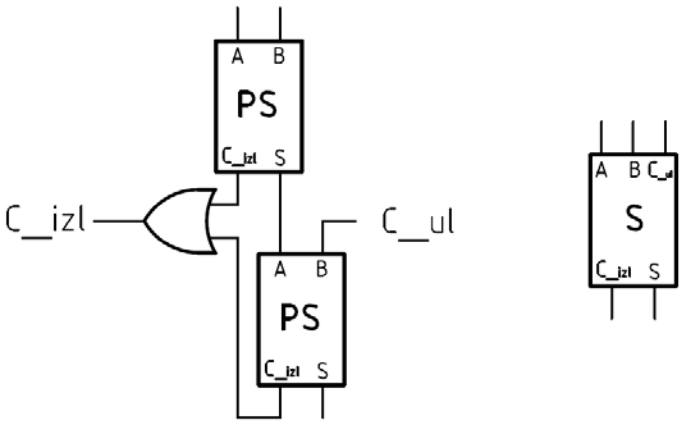


$$S = A\overline{B}\overline{C}_{ul} + \overline{A}B\overline{C}_{ul} + \overline{A}\overline{B}C_{ul} + ABC_{ul} = A \oplus B \oplus C_{ul}$$

$$C_{izl} = AB\overline{C}_{ul} + A\overline{B}C_{ul} + \overline{A}BC_{ul} + ABC_{ul} = (A \oplus B) \cdot C_{ul} + AB$$



Može se pokazati da se funkcija potpunog sabirača ostvaruje kombinacijom dva polusabirača. Ovo je vrlo važno za praktičnu realizaciju funkcije aritmetičkog sabiranja binarnim sklopovima.



Primer četvorobitnog sabirača dobijen kaskadnom vezom četiri jednobitna potpuna sabirača (u ovom primeru se uvodi pojam REGISTAR koji će kasnije biti detaljnije objašnjen, a u ovom slučaju ga treba razumeti kao vektorski prostor, odnosno uredjenu binarnu grupu u kojoj se čuvaju vrednosti promenljivih):

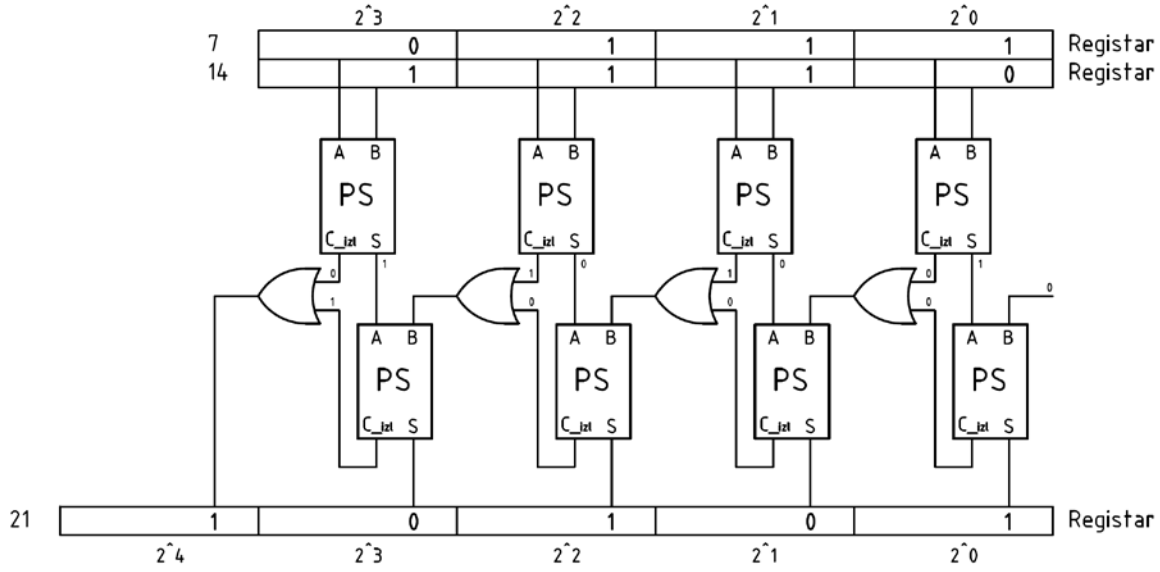


Figure 10: Binary Addition Machine (Wandel)
https://www.youtube.com/watch?v=GcDshWmhF4A

pogledati takodje: https://www.youtube.com/watch?v=3NJ7Fr6VrPU

Koncept kibernetike, odnosno apstraktnog opisa neke funkcije koja se kasnije fizički realizuje u okviru neke tehnologije. U ovom konkretnom slučaju radi se o mehaničkoj realizaciji koja nas vodi ka rešenju mehaničkog sabirača, koji je u ovom primeru izveden korišćenjem rešenja binarnog sabirača dva višebitna broja pomoću jednostavne drvene konstrukcije. Ovaj primer pokazuje kako se u mehatronici sreću apstraktni svet kibernetskih principa i realni svet tehnologije. Treba uočiti da jedan kibernetski konstrukt, u ovom slučaju sabirač digitalno iskazanog broja može da se realizuje kroz praktično neograničen broj tehnologija i tehničkih rešenja. Danas tehnologija digitalnog računara počiva na tehnologiji poluprovodničkih aktivnih elemenata. Ovaj primer pokazuje kako je moguće jednu te isti kibernetski kostrukt relizovati čisto mehaničkom tehnologijom. Takodje i pnemumatikom, hidraulikom, ili tehnologijom koja je bazirana na hemijskim mehanizmima, ili kvantnim fenomenima, što nas vodi ka kvantnom kompjuteru. Čovekov mozak kibernetske konstrukte relalizuje koristeći biološke konstrukte specijalizovanih ćelija koje nazivamo neuronima. Nezavisno od tehnologije, kibernetski konstrukt kao apstraktna matematička tvorevina ostaje nepromenjen. Izučavajući kibernetiku, zapravo se izučavaju tehnološke invarijante, odnosno principi funkcionisanja informacionih mašina koje ne zavise od usvojene tehnologije, pa su time oni i vremenski neosetljivi (tehnologije se menjaju, kibernetski principi ne!).

Aritmetička operacija oduzimanja

Aritmetička operacija oduzimanja bi mogla da se izvede na način analogan decimalnom oduzimanju, pri čemu važe sledeća pravila:

0 - 0 = 0

1 - 1 = 0

1 - 0 = 1

0 - 1 = 1 + pozajmica 10

Ovaj postupak se naziva direktnim postupkom oduzimanja.

Primer:

	N_2	N_{10}
	1001 0110	150
-	0110 1010	106
	0010 1100	44

kolona 2^0 0 - 0 = 1

kolona 2^1 1 - 1 = 0

kolona 2^2 1 - 0 = 1

kolona 2^3 pošto se ne može oduzeti jedinica od nule, to se iz kolone 2^4 pozajmljuje 1 i prebacuju u kolonu 2^3 kao 10. sada je u ovoj koloni

10 – 1 = 1

kolona 2^4 pošto je zbog pozajmice sada u ovoj koloni 0 sledi:

0 - 0 = 0

kolona 2^5 ovde se mora izvršiti pozajmljivanje i to iz kolone 2^7 pošto je u koloni 2^6 nula. Uzeta jedinica iz kolone 2^7 ima vrednost 10 u koloni 2^6, ali kada se od iz ove kolone pozajmi 10 za kolonu 2^5 u njoj će ostati 1 a u koloni 2^5 10. Prema tome u koloni 2^5 važi:

10 - 1 = 1

kolona 2^6 imajući u vidu prethodne pozajmice, ovde je:

1 - 1 = 0

kolona 2^7 imajući u vidu prethodne pozajmice, ovde je:

0 - 0 = 0

Ipak, iz tehnoloških razloga, postupak direktnog binarnog oduzimanja se kod savremenih računara ne koristi. Umesto toga, oduzimanje se sprovodi po proceduri dodavanja negativnog broja, odnosno komplementnom aritmetikom. Drugo, u opštem slučaju, ovim se otvara pitanje zapisa takozvanih označenih celobrojnih brojeva (signed integers).

Negativni broj se može formirati na tri načina:

1. sa predznakom u vidu binarne cifre 1 koja se nalazi na poziciji najznačajnijeg bita,
2. primenom prvog komplementa i
3. primenom drugog komplementa.

Kod savremenih računara negativni broj se iskazuje drugim komplementom jer to omogućava niz praktičnih koristi.

Pod terminom ’prvi komplement’ podrazumeva se komplement, odnosno dopuna do najveće cifre brojnog sistema. U slučaju binarnog brojnog sistema najveća cifra je 1, tako da je u ovom slučaju prvi komplement zapravo komplement jedinice, odnosno dopuna do jedinice.

Pod terminom ’drugi komplement’ podrazumeva se komplement do osnove brojnog sistema. U slučaju binarnog brojnog sistema osnova je 2, tako da u ovom slučaju drugi komplement jeste komplement dvojke, odnosno dopuna do dvojke.

Postupak formiranja drugog komplementa uključuje prvo obrazovanje prvog komplementa koji podrazumeva komplementiranje svih bitova pozitivnog broja, na primer 0000 0011 ($=3_{10}$) se prevodi u 1111 1100 ($= -3_{10}^1$). Zatim se dobijenom prvom komplementu dodaje jedinica, odakle se on prevodi u sledeći oblik: 1111 1101($= -3_{10}^2$).

Primer formiranja zapisa negativnog broja drugim komplementom i operacije oduzimanja 7 – 4 sabiranjem broja 7 sa drugim komplementom broja 4:

a) formiranje negativne vrednosti broja 4 prema drugom komplementu:

	0000 0100	broj 4 izražen prirodnim binarnim kodom

	1111 1011	prvi komplement broja 4
+	0000 0001	dodaje se 1

	1111 1100	drugi komplement broja 4

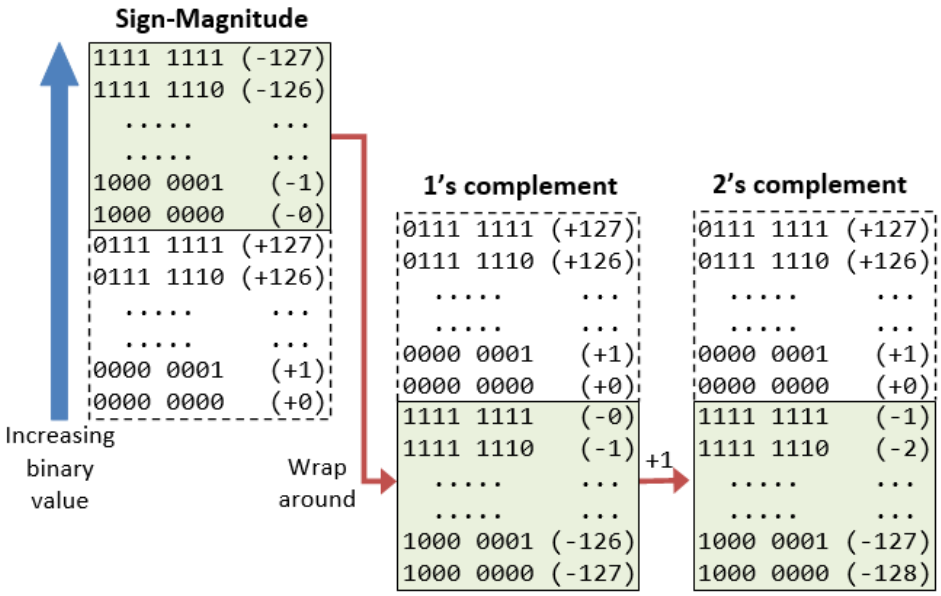
b) sabiranje broja 7 i broja - 4 izraženog preko drugog komplementa.

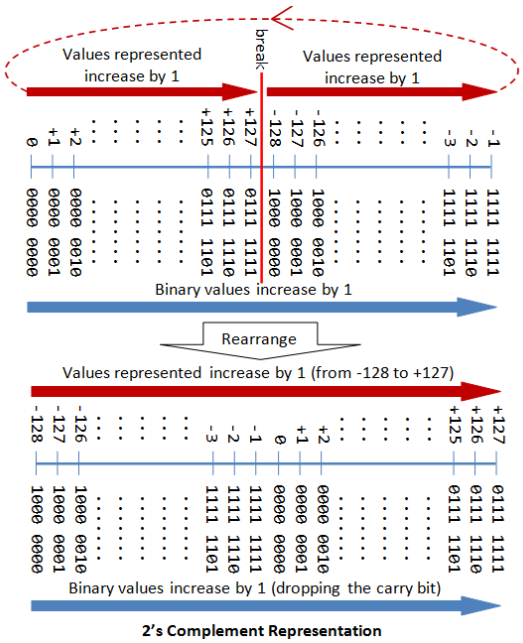
	N_2	N_{10}	

	0000 0111	7	
+	1111 1100	- 4	drugi komplement za 4

	0000 0011	3	

Prenos jedinice na najvišem bitu ovde nije potreban!

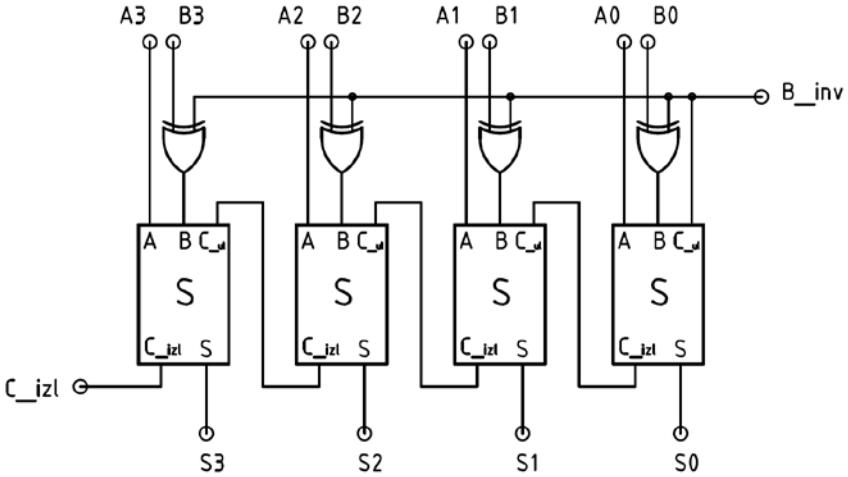




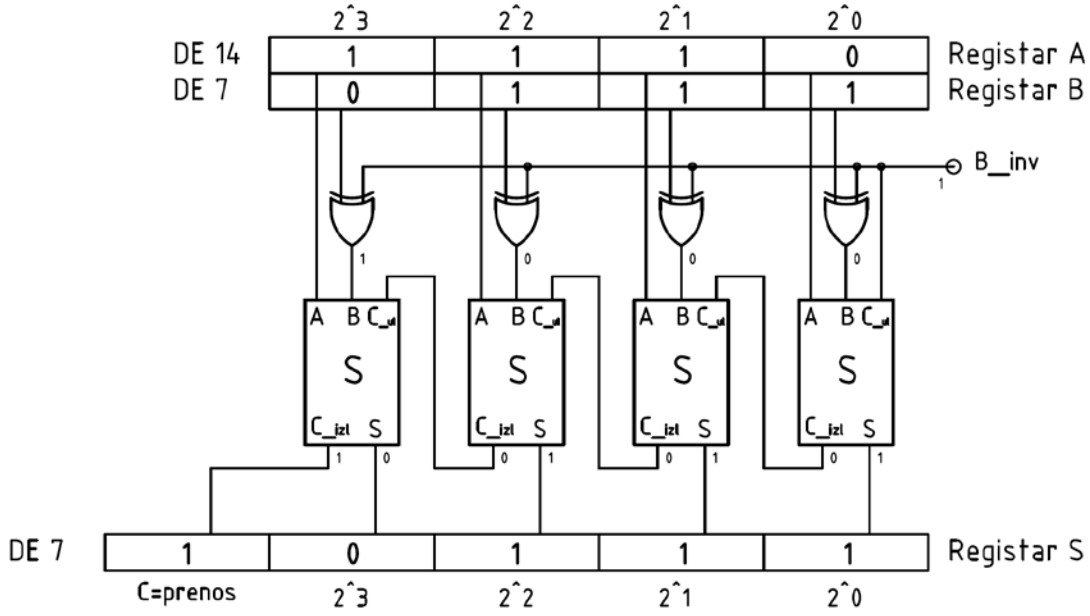
Fizička realizacija operacije oduzimanja primenom sabirača može da se ostvari korišćenjem EX-ILI logičkog kola. Prvo se pomoću niza EX-ILI logičkih kola formira prvi komplement binarne grupe B, korišćenjem komandnog kanala B_inv. Ukoliko je B_inv = 1 binarna grupa B se invertuje:

B	B_inv	Y=B⊕B_inv
0	0	0
1	0	1
0	1	1
1	1	0

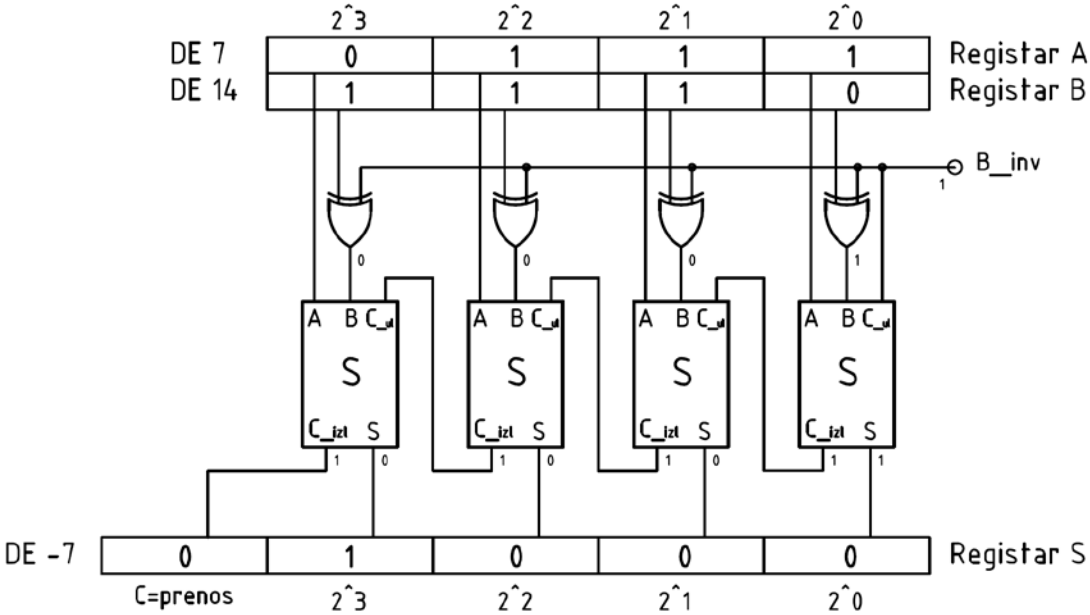
Formiranje drugog komplementa zahteva dodavanje jedinice. Taj zahtev može da se ostvari korišćenjem C_ul na najnižoj poziciji sabirača, na koji se dovodi sadržaj sa B_inv. Dalje se prikazuje dogradjena mreža četvorobitnog sabirača koja omogućava formiranje drugog komplementa sabirka B i time omogućava oduzimanje, odnosno izvodjenje operacije A – B = S.



Funkcija mreže sabirača na primeru decimalnog oduzimanja 14 – 7 = 7 ostvaruje se kao što je prikazano na sledećoj slici. Na poziciji najznačajnijeg bita u rezultatu se pojavljuje cifra 1. Ona nema numerički značaj, već ukazuje na pozitivni znak dobijene razlike.



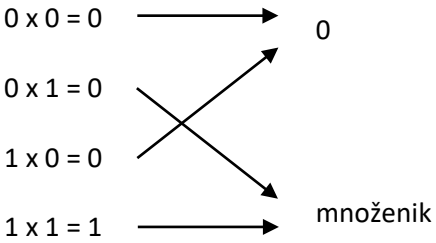
Prethodni primer se odnosi na slučaj kada je rezultat pozitivan, odnosno kada je B manje od A. Za ovaj slučaj je karakteristično da je bit na najvišoj poziciji jednak 1 uvek kada je rezultat operacije oduzimanja pozitivan. Medjutim, u slučaju da je B veće od A, rezultat će biti negativan, u obliku prvog komplementa, pri čemu će bit na najvišoj poziciji biti jednak 0. Dakle, 0 označava negativni znak i služi za proveru.



Aritmetička operacija množenja i deljenja

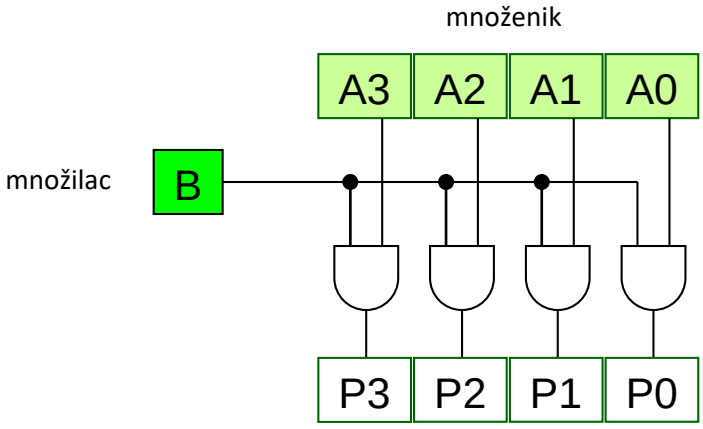
Izvodjenje operacija množenja i deljenja je daleko složenije od sabiranja ili oduzimanja. Načelno, ove operacije se svode na sukcesivno izvodjenje niza operacija sabiranja, pri čemu za razliku od samog sabiranja koje se u osnovi obavlja primenom logičkih kola, ovde postoji i problem memorisanja medjuvrednosti, njihovog sabiranja, komplementiranja i problem pomeranja binarnog sadržaja. Praktično, množenik se redom množi svakom cifrom množioca posebno, pa se dobijeni parcijalni proizvodi sabiraju, vodeći pri tome računa o pozicijama cifara u množiocu, kao i kod decimalnog množenja.

Množenje binarnog broja množiocem koji ima samo jednu cifru je vrlo jednostavno i izvodi se pomoću **I** logičkog operatora sa dva ulaza, odnosno važi:



Očigledno je da se gore navedena pravila svode samo na dva slučaja: množenje nulom – rezultat je nula i množenje jedinicom – rezultat je množenik.

Logička šema množenja višecifarnog množenika (A) jednocifarnim množiocem (B):

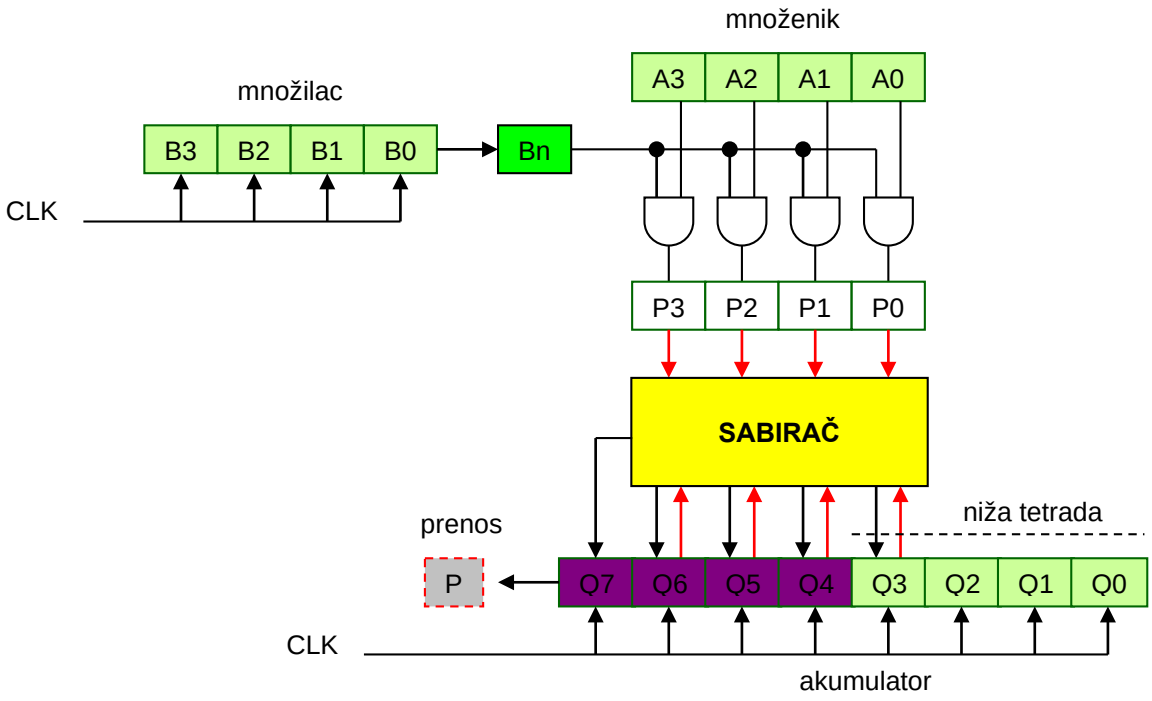


Ovakav generički modul moguće je dalje koristiti za svaku cifru množioca posebno. Važno je znati da se parcijalni proizvodi ne memorišu odvojeno već se oni neposredno posle formiranja sabiraju, odnosno sukcesivno akumuliranju. Pri tome se lokalne sume pomeraju u zavisnosti od pozicije cifre množioca.

Primer: množenik A=1010 i množilac B=1011

```
          1 0 1 0          : množenik
    x     1 0 1 1          : množilac
-------------------------------------------
          1 0 1 0
         1 0 1 0
        0 0 0 0
       1 0 1 0
-------------------------------------------
       1 1 0 1 1 1 0       : proizvod
```

Logička šema množenja višecifarnog množenika (A) višecifarnim množiocem (B) primenom modula sabirača i pomeračkog registra Q:



algoritam:

resetuje se Q, podešava se SW brojač na n (dužina reči procesora)

početak petlje:

pomera se množilac udesno kroz carry sistemski registar; dalje se ispituje c i:

1: kada je bit c (carry) = 0 samo se pomera sadržaj Q

2: kada je bit c (carry) = 1 formira se parcijalni proizvod, pomera se sadržaj Q i sabira parcijalni proizvod i pomereno Q

dekrementira se brojač

ispituje se tekuća vrednost brojača; ako je jednaka 0 preskače se sledeća instrukcija

idi na početak petlje

kraj

Za prethodno navedeni primer, množenje primenom prethodnog algoritma se izvodi na sledeći način:

		0 0 0 0	Qo početno stanje akumulatora
1 0 1 0 x	**1 0 1 1 =**		
c=1		1 0 1 0	P1 prvi parcijalni proizvod
	prepis P1 u akumulator	1 0 1 0	Q1
c=1		1 0 1 0	P2 drugi parcijalni proizvod
	pomeraj akumulatora udesno	1 0 1 0	Q1 pomereno
	suma Q1 pomereno i P2	1 1 1 1 0	Q2
c=0	pomeraj akumulatora udesno	1 1 1 1 0	Q2 pomereno = Q3
c=1		1 0 1 0	P4
	pomeraj akumulatora udesno	1 1 1 1 0	Q3 pomereno
	suma Q3 pomereno i P4	1 1 0 1 1 1 0	Q4
	proizvod A * B	1 1 0 1 1 1 0	Q4

Konkretni detalji realizacije operacije množenja, pre svega u smislu definisanja algoritma množenja uslovljeni su specifičnostima funkcije karakterističnih digitalnih modula, koji su neophodni za praktičnu realizaciju ove operacije. Kod digitalnih sistema uvek je nužno razmatrati fizičku realizaciju neke operacije paralelno sa razmatranjem principijelnih pitanja za njeno izvodjenje.

Iz prethodnog primera očigledno je da je postupak množenja značajno složeniji u poredjenju sa postupkom koji se koristi za aritmetičku operaciju sabiranja ili oduzimanja. Da bi se realizacija ove operacije ubrzala koriste se specijalni digitalni moduli koji se nazivaju množači. Množači se sastoje iz polja sabirača, koji su tako medjusobno povezani da formiraju kombinacionu logičku mrežu koja aritmetičku operaciju binarnog množenja realizuje kao i operaciju binarnog sabiranja – u jednom prolazu.

a1.5 Operacije nad binarnim grupama

Bulove binarne operacije se mogu izvoditi nad binarnim grupama. One su vrlo značajne zar rad digitalnih računarskih sistema jer se u izvesnim situacijama one koriste kao pomoćne operacije u izvodjenju aritmetičkih i logičkih operacija, ili za ostvarivanje nekih specijalnih funkcija manipulacije podacima (manipulacija bitovima unutar binarne grupe, na primer manipulacija bitovima u okviru jednog bajta).

Logički proizvod brojeva

Primenom logičkog proizvoda (I operator) nad dve binarne grupe sledi da će na pozicijama na kojima su oba bita imala jediničnu vrednost rezultat biti takodje jedinični, dok će na ostalim pozicijama rezultujući bitovi imati nultu vrednost.

Primer:

```
-----------------------------------------

        0101 0101           bajt

I       0000 1111           maska

-----------------------------------------

        0000 0101
```

Ova operacija se koristi za maskiranje, odnosno **brisanje jednog dela binarnog sadržaja**. U prethodnom primeru briše se viša tetrada a zadržava niža.

Logički zbir brojeva

U ovom slučaju primenom logičkog zbira brojeva daje u rezultatu broj koji ima jedinične vrednosti bitova na svim pozicijama na kojim oba bita imaju različite vrednosti od nulte. Ovom operacijom se **upisuje željeni sadržaj** u prazne delove binarnog bloka.

Primer:

```
-----------------------------------------

        0101 0101           bajt

ILI     1001 0000           bajt koji se upisuje

-----------------------------------------

        1101 0101
```

U navedenom primeru se vidi da je na poziciji više tetrade upisana željena vrednost 1001, dok je niža tetrada zadržala originalnu vrednost.

Logički operacija ekskluzivnog ILI

Ova operacija se primenjuje kada se želi **poredjenje dva broja**. Ukoliko su dva broja identična, primenom ekskluzivno ILI operacije rezultat će biti 0.

Primer 1:

```
-----------------------------------------
        0101 0101          bajt 1
Ex ILI  1001 0000          bajt 2
-----------------------------------------
        1100 0101          bajt 1 različit od bajta 2
-----------------------------------------
        0101 0101          bajt 1
Ex ILI  0101 0101          bajt 2
-----------------------------------------
        0000 0000          bajt 1 jednak bajtu 2
```

U navedenim primerima, prvi primer se odnosi na slučaj kada su dve binarne grupe različite a u drugom primeru je naveden slučaj kada su dve binarne grupe identične. Očigledno je da se EXILI operator može iskoristiti kao **komparator**.

Pomeranje binarnog sadržaja ulevo i udesno

Pomeranje sadržaja udesno kod decimalnih sistema ostvaruje se množenjem sadržaja osnovnom sistema, odnosno brojem 10. Budući da se radi o binarnom sistemu, pomeranje sadržaja ulevo postiže se njegovim **binarnim množenjem** sa 2 odnosno množiocem (0010_2):

```
        1 0 1 0        : množenik – broj koji se pomera
  x     0 0 1 0        : množilac 2(10)
-----------------------------------------
        0 0 0 0
      1 0 1 0
-----------------------------------------
      1 0 1 0 0        : proizvod -  broj pomeren za jednu cifru ulevo
```

Binarnim deljenjem sa 2 sadržaj se pomera udesno. Ovo važi samo u slučaju kada najviši ili najniži bitovi ne 'ispadaju' preko granice binarne grupe.

U zavisnosti kako je rešen problem ispadajućih bitova postoje tri karakteristična varijante ove operacije:

1. logičko pomeranje – kada se ispadajući bit definitivno gubi,
2. rotirajuće pomeranje – kada se bit koji je ispao preko jedne granice vraća peko druge granice na suprotnoj strani i
3. aritmetičko pomeranje – kada se bit koji je ispao čuva u posebnom jednobitnom registru; na ovaj način se omogućava čuvanje znaka ili održava opšta primenljivost deljenja ili množenja dvojkom.