CAPITOLUL 4

CIRCUITE ARITMETICE COMBINAȚIONALE

În acest capitol se descriu diferite tipuri de sumatoare și circuite combinaționale de înmulțire. Dintre sumatoare, se prezintă sumatorul elementar, sumatorul cu propagarea succesivă a transportului, sumatorul cu anticiparea transportului, sumatorul cu salvarea transportului și sumatorul zecimal. Dintre circuitele combinaționale de înmulțire, se prezintă circuitul de înmulțire matriceală și arborele Wallace. La sfârșitul capitolului se descrie modul de implementare pe o placă de dezvoltare a unui sumator cu anticiparea transportului și a unui circuit de înmulțire matriceală.

4.1 Sumatoare

4.1.1 Sumatorul elementar

Un sumator elementar adună trei intrări de câte un bit. Două din intrări sunt biții care trebuie adunați, notați cu x_i, y_i, iar cealaltă intrare este transportul de la bitul din poziția mai puțin semnificativă, notat cu T_i. Ieșirile sunt bitul sumă S_i și transportul către bitul din poziția mai semnificativă, T_{i+1}.

Ieșirile sumatorului elementar au următoarele expresii booleene:

$$S_i = x_i \oplus y_i \oplus T_i \tag{4.1}$$

$$T_{i+1} = x_i y_i + (x_i + y_i)\ T_i \tag{4.2}$$

unde:

$$x_i \oplus y_i = x_i \overline{y}_i + \overline{x}_i y_i \tag{4.3}$$

Sumatorul elementar este unul din blocurile de bază ale sumatoarelor mai complexe. Un caz particular de sumator elementar este *semisumatorul elementar*, care

adună două intrări de câte un bit (nu există intrare de transport) și generează un bit sumă și un bit de transport.

După cum semisumatoarele elementare și sumatoarele elementare se utilizează pentru realizarea sumatoarelor, semiscăzătoarele elementare și scăzătoarele elementare se utilizează pentru realizarea scăzătoarelor. Un *semiscăzător elementar* scade două intrări de câte un bit și generează un bit diferență și un bit de împrumut. Dacă bitul descăzut este mai mic decât bitul scăzător, bitul de împrumut va fi setat la 1, iar în caz contrar, bitul de împrumut va fi setat la 0. Un *scăzător elementar* are trei intrări și două ieșiri. Două din intrări sunt cei doi biți care trebuie scăzuți, descăzutul și scăzătorul, iar cealaltă intrare este împrumutul de la bitul din poziția mai puțin semnificativă. Ieșirile sunt bitul diferență și împrumutul către bitul din poziția mai semnificativă.

4.1.2 Sumatorul cu propagarea succesivă a transportului

Unul din algoritmii de bază pentru adunare este algoritmul de adunare cu propagarea succesivă a transportului. Acest algoritm poate fi implementat în mod simplu cu sumatoare elementare. Principul este similar cu adunarea obișnuită. Fie $x_3 x_2 x_1 x_0$ și $y_3 y_2 y_1 y_0$ două numere binare de câte patru biți. Pentru adunarea acestor numere, se adună x_0 și y_0 pentru a determina cifra cea mai puțin semnificativă a sumei. Dacă rezultă un transport, acesta se adună cu x_1 și y_1 pentru a determina următoarea cifră mai semnificativă a sumei. Acest proces continuă până când se adună x_3 și y_3. Dacă rezultă un transport final, acesta va deveni cifra cea mai semnificativă a sumei.

O posibilitate de implementare a acestui proces este de a se conecta mai multe sumatoare elementare în serie, câte un sumator elementar pentru fiecare bit al numerelor care trebuie adunate. Figura 4.1 prezintă un sumator cu propagarea succesivă a transportului pentru adunarea a două numere binare de câte patru biți. După cum se ilustrează în figură, ieșirea de transport a unui sumator elementar este conectată la intrarea de transport a următorului sumator elementar. Intrarea de transport a sumatorului din poziția cea mai puțin semnificativă (T_0) este setată la valoarea logică 0. De aceea, pentru această poziție se poate utiliza un semisumator elementar.

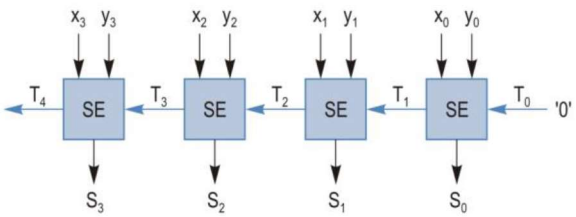

Figura 4.1. Schema bloc a unui sumator de patru biți cu propagarea succesivă a transportului.

Sumatorul cu propagarea succesivă a transportului este un sumator paralel, deoarece operanzii sunt aplicați la intrare în același timp. Numele acestui sumator

provine de la faptul că transportul trebuie să se propage succesiv prin toate sumatoarele elementare înainte de a se cunoaște rezultatul final. Deși acest tip de sumator este simplu de realizat și are un cost redus, este unul din cele mai lente sumatoare. Aceasta deoarece transportul trebuie să se propage de la poziția bitului cel mai puțin semnificativ la poziția bitului cel mai semnificativ. Acest tip de sumator este utilizat uneori ca o celulă de adunare pentru realizarea sumatoarelor de dimensiuni mai mari.

Sumatorul cu propagarea succesivă a transportului poate fi utilizat și ca scăzător, deoarece scăderea a două numere binare se poate executa prin adunarea complementului față de doi (C2) al scăzătorului la descăzut. De exemplu, fiind date două numere binare $X = 01001$ și $Y = 00011$, scăderea $X - Y$ poate fi executată prin adunarea numărului X la complementul față de doi al numărului Y:

```
X          0 1001+
Y(C2)      1 1101
X-Y     ① 0 0110
```

Transportul de la poziția cea mai semnificativă a rezultatului se neglijează. În același mod se poate realiza și un *scăzător*. În acest caz, fiecare bit al scăzătorului este scăzut din bitul corespunzător al descăzutului pentru a forma un bit diferență. Dacă bitul descăzutului este 0 și bitul scăzătorului este 1, se împrumută 1 de la următoarea poziție mai semnificativă.

4.1.3 Sumatorul cu anticiparea transportului

Sumatorul cu anticiparea transportului crește viteza operației de adunare prin reducerea timpului necesar pentru generarea semnalelor de transport. În cazul acestui sumator, intrarea de transport necesară pentru un etaj este generată în mod direct, utilizând semnale de la toate etajele precedente, în loc de a se aștepta propagarea lentă a transporturilor de la un etaj la altul.

Figura 4.2 prezintă schema bloc a unui sumator cu anticiparea transportului pentru adunarea a două numere de câte patru biți. În această figură, blocurile de transport generează intrările de transport pentru sumatoarele elementare. Intrările fiecărui bloc de transport sunt doar numerele care trebuie adunate și intrarea de transport inițială T_0.

Expresiile booleene ale fiecărui bloc de transport pot fi definite pe baza ecuației transportului de ieșire al unui sumator elementar:

$$T_{i+1} = x_i y_i + (x_i + y_i)\ T_i \tag{4.4}$$

De exemplu, ecuațiile booleene ale semnalelor de transport T_1 și T_2 se pot obține din ecuația (4.4) pentru $i = 0$, respectiv $i = 1$:

$$T_1 = x_0 y_0 + (x_0 + y_0)\ T_0 \tag{4.5}$$

$$T_2 = x_1 y_1 + (x_1 + y_1)\ T_1 = x_1 y_1 + (x_1 + y_1)\ [x_0 y_0 + (x_0 + y_0)\ T_0] \tag{4.6}$$

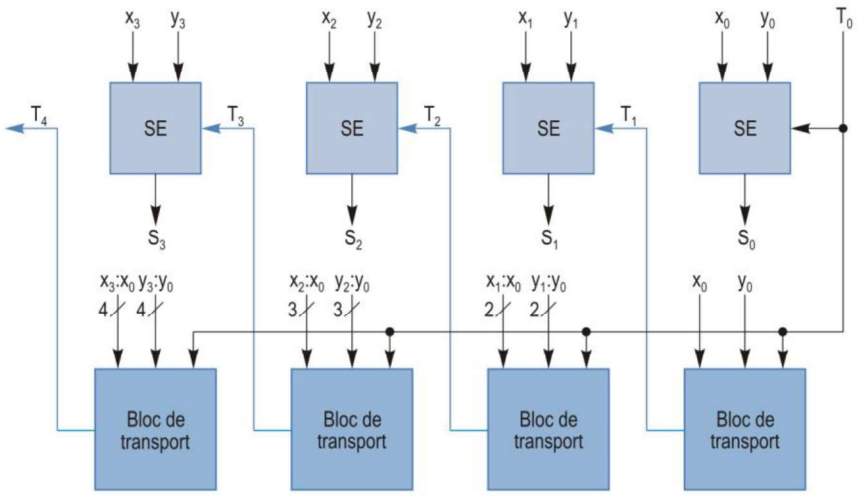

Figura 4.2. Schema bloc a unui sumator de patru biți cu anticiparea transportului.

Pentru simplificarea expresiei fiecărui semnal de transport T_i, se introduc două funcții, notate cu g și p. Funcția g se referă la *generarea* transportului de către un bloc de transport, iar funcția p se referă la *propagarea* intrării de transport la ieșirea de transport a blocului. Aceste funcții sunt definite astfel:

$$g_i = x_i\, y_i \tag{4.7}$$
$$p_i = x_i + y_i \tag{4.8}$$

Denumirea de generare a transportului provine din faptul că etajul i generează un semnal de transport cu valoarea logică '1' (T_{i+1} = '1') independent de semnalul T_i dacă x_i și y_i au ambele valoarea logică '1' ($x_i\, y_i$ = '1'). Etajul i propagă semnalul de transport T_i, deci T_{i+1} va avea valoarea logică '1' ca răspuns la T_i având valoarea logică '1', dacă x_i sau y_i are valoarea logică '1' ($x_i + y_i$ = '1').

Astfel, ecuația (4.1) a sumei pentru sumatorul elementar poate fi scrisă în funcție de g_i și p_i sub forma următoare:

$$T_{i+1} = g_i + p_i\, T_i \tag{4.9}$$

Utilizând aceste notații, ecuațiile booleene pentru semnalele de transport ale sumatorului cu anticiparea transportului de patru biți se pot exprima astfel:

$$T_1 = g_0 + p_0\, T_0 \tag{4.10}$$
$$T_2 = g_1 + p_1 g_0 + p_1\, p_0\, T_0 \tag{4.11}$$
$$T_3 = g_2 + p_2 g_1 + p_2\, p_1 g_0 + p_2\, p_1\, p_0\, T_0 \tag{4.12}$$
$$T_4 = g_3 + p_3 g_2 + p_3\, p_2\, g_1 + p_3\, p_2\, p_1 g_0 + p_3\, p_2\, p_1\, p_0\, T_0 \tag{4.13}$$

Figura 4.3 prezintă schema bloc a sumatorului cu anticiparea transportului implementat pe baza acestor ecuații. Fiecare sumator de un bit produce semnalele de

propagare a transportului pe bit *p* și de generare a transportului pe bit *g* în locul ieșirilor de transport. Generatorul de transport anticipat convertește cele patru seturi de semnale *p*, *g* în intrările de transport necesare pentru cele patru sumatoare. Fiecare sumator de un bit produce și bitul corespunzător al sumei, în același mod ca și un sumator elementar.

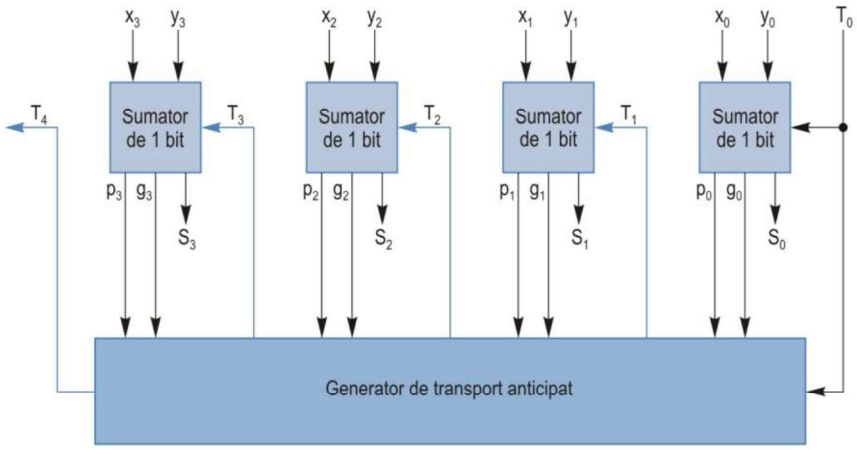

Figura 4.3. Schema bloc modificată a unui sumator de patru biți cu anticiparea transportului care utilizează un generator de transport anticipat.

Pentru a reduce complexitatea circuitelor necesare generării semnalelor de transport, se limitează numărul de intrări ale porților și numărul de porți alimentate de acestea la o anumită valoare, în funcție de tehnologia utilizată pentru implementare. Aceasta necesită adăugarea unor nivele logice suplimentare la circuitul cu anticiparea transportului. De exemplu, dacă în cazul precedent se limitează numărul de intrări la patru, va fi necesară o întârziere mai mare pentru generarea semnalului de transport T_4. Pentru a realiza limitarea amintită, se definesc două noi funcții, de *generare a transportului pe grup* $G_{i,k}$ și de *propagare a transportului pe grup* $P_{i,k}$ pentru blocul biților de la i până la k. De exemplu, pentru blocul biților de la 0 până la 3, ecuațiile booleene ale celor două funcții se pot scrie astfel:

$$G_{0,3} = g_3 + p_3 g_2 + p_3 p_2 g_1 + p_3 p_2 p_1 g_0 \tag{4.14}$$

$$P_{0,3} = p_3 p_2 p_1 p_0 \tag{4.15}$$

Cu aceasta, pentru semnalul de transport T_4 se obține:

$$T_4 = G_{0,3} + P_{0,3}\, T_0 \tag{4.16}$$

Această ecuație are o formă similară cu ecuația (4.10) și poate fi obținută prin înlocuirea semnalelor de generare a transportului pe bit și de propagare a transportului pe bit cu semnalele corespunzătoare de generare a transportului pe grup și de propagare a transportului pe grupul de biți de la 0 până la 3.

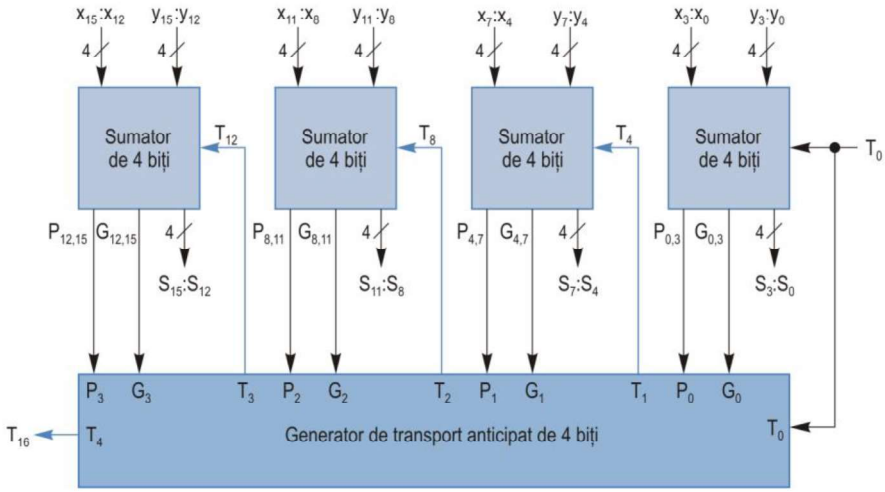

Figura 4.4. Schema bloc a unui sumator de 16 biți format din sumatoare de patru biți conectate prin semnale de transport generate anticipat.

Sumatorul cu anticiparea transportului de patru biți se poate extinde la un sumator de dimensiune mai mare. Sumatoarele de un bit se pot înlocui cu sumatoare pentru grupuri de patru biți, după cum se ilustrează în figura 4.4. Fiecare etaj al sumatorului produce semnalele de propagare a transportului pe grup P și de generare a transportului pe grup G, iar un generator de transport anticipat convertește cele patru seturi de semnale P și G în intrările de transport necesare pentru cele patru etaje.

În figura 4.4, semnalul T_8 are o ecuație care este similară cu ecuația (4.11):

$$T_8 = G_{4,7} + P_{4,7}\, G_{0,3} + P_{4,7}\, P_{0,3}\, T_0 \tag{4.17}$$

unde:

$$G_{4,7} = g_7 + p_7\, g_6 + p_7\, p_6\, g_5 + p_7\, p_6\, p_5\, g_4 \tag{4.18}$$

$$P_{4,7} = p_7\, p_6\, p_5\, p_4 \tag{4.19}$$

Ecuațiile booleene pentru semnalele de transport T_{12} și T_{16} sunt similare cu ecuațiile (4.12), respectiv (4.13).

Sumatorul cu anticiparea transportului are o viteză mai ridicată decât sumatorul cu transport succesiv. Totuși, acest sumator necesită un spațiu suplimentar în circuitul integrat datorită complexității mai ridicate.

4.1.4 Sumatorul cu salvarea transportului

Adunarea cu salvarea transportului este o altă tehnică care poate fi utilizată pentru creșterea vitezei operației de adunare atunci când trebuie adunate mai mult de două numere. În timp ce tehnica de adunare cu anticiparea transportului reduce timpul

necesar pentru generarea semnalelor de transport, adunarea cu salvarea transportului reduce timpul de propagare al semnalelor de transport. Un sumator cu salvarea transportului (SST) de *n* biți este o simplă colecție de *n* sumatoare elementare independente. Intrările acestui sumator sunt reprezentate de trei numere de câte *n* biți care trebuie adunate. Fiecare sumator elementar generează câte un bit sumă și un bit de transport, care formează un cuvânt sumă *S* de *n* biți, respectiv un cuvânt de transport *T* de *n* biți.

Spre deosebire de sumatorul cu propagarea succesivă a transportului, într-un sumator cu salvarea transportului fiecare sumator elementar funcționează independent unul de celălalt, iar semnalele de transport nu sunt propagate între sumatoarele elementare. Astfel, viteza de adunare este substanțial mai ridicată, deoarece toți biții sumă și toți biții de transport pot fi generați în paralel. Pentru obținerea rezultatului final, cuvântul sumă și cuvântul de transport trebuie adunate cu un sumator obișnuit, numit și *sumator cu propagarea transportului* (SPT). Acest termen indică un sumator care nu este un sumator cu salvarea transportului, care utilizează propagarea succesivă a transportului, anticiparea transportului, sau o altă tehnică.

Exemplificăm funcționarea unui sumator cu salvarea transportului pentru adunarea a patru numere *X*, *Y, Z* și *W*. Sumatorul cu salvarea transportului generează mai întâi o sumă a primelor trei numere și un transport salvat. Presupunând că $X = 5$ (0101), $Y = 3$ (0011), $Z = 4$ (0100) și $W = 1$ (0001), suma și transportul salvat vor fi:

```
X                       0101
Y                       0011
Z                       0100
Suma                    0010
Transportul salvat      1010
```

În următoarea etapă, se adună suma, al patrulea număr (*W*) și transportul salvat, generând o nouă sumă și un nou transport salvat:

```
Suma                    0010
W                       0001
Transportul salvat      1010
Noua sumă               1001
Noul transport salvat   0100
```

În ultima etapă, se utilizează un sumator cu anticiparea transportului pentru a aduna noua sumă și noul transport salvat, rezultatul fiind 13 (1101).

```
Noua sumă               1001
Noul transport salvat   0100
Rezultat                1101
```

Pe baza acestor etape, se poate realiza un sumator cu operanzi multipli. Un asemenea sumator este utilizat adesea la circuitele de înmulțire pentru acumularea produselor parțiale. De exemplu, figura 4.5 prezintă o schemă bloc pentru adunarea a patru numere. Acest circuit cuprinde două sumatoare cu salvarea transportului, fiecare fiind format dintr-o serie de sumatoare elementare. În ultima etapă, se utilizează un sumator cu anticiparea transportului pentru obținerea sumei finale.

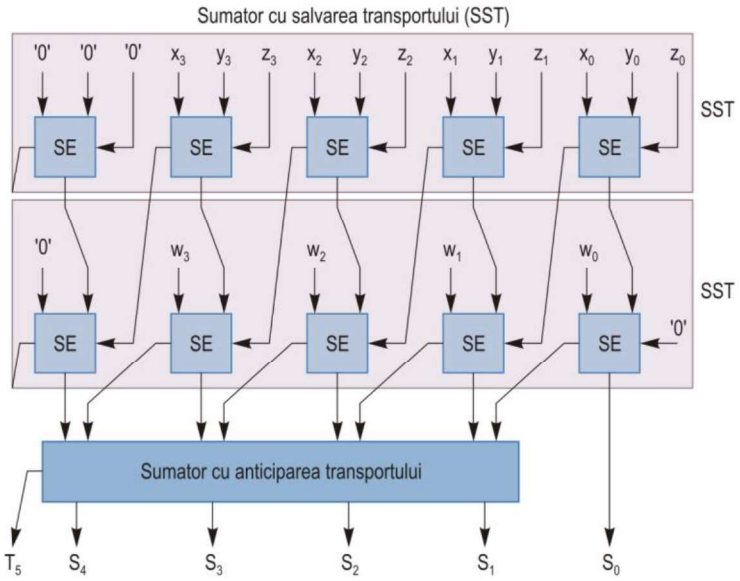

Figura 4.5. Schema bloc a unui sumator pentru adunarea a patru numere de câte patru biți utilizând sumatoare cu salvarea transportului.

Figura 4.6 ilustrează modul în care se poate utiliza un singur sumator cu salvarea transportului pentru adunarea unui set de numere. Mai întâi, se aplică trei numere la intrările *X*, *Y* și *Z*. Suma și transportul generate în urma adunării acestor numere se memorează în bistabile D, se aplică din nou la intrările *X* și *Y* în următorul ciclu de ceas și se adună cu al patrulea număr, care se aplică la intrarea *Z*. Acest proces continuă până când se adună toate numerele.

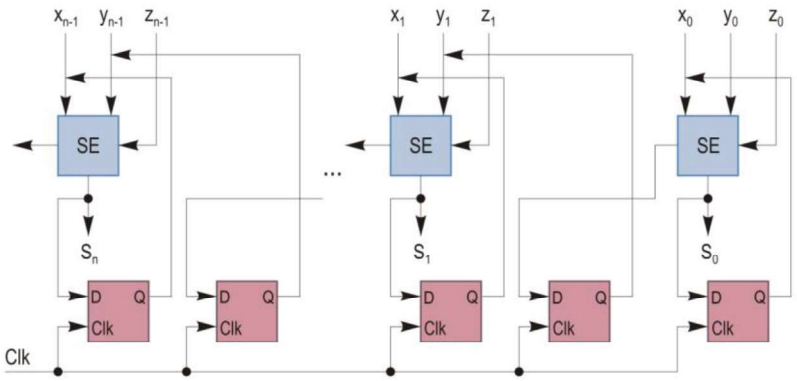

Figura 4.6. Schema bloc a unui sumator de *n* biți cu salvarea transportului pentru adunarea unui set de numere.

4.1.5 Sumatorul zecimal

Există cazuri în care numerele sunt reprezentate în zecimal, într-un cod binar zecimal BCD (*Binary Coded Decimal*). În asemenea cazuri, în locul conversiei numerelor în binar şi utilizarea unui sumator binar pentru adunarea lor, este mai eficient să se utilizeze un sumator zecimal. Un sumator zecimal adună două cifre BCD în paralel şi generează o sumă în cod BCD. Deoarece o cifră BCD are valori cuprinse între 0 şi 9, ori de câte ori suma depăşeşte valoarea 9 sau se generează un transport către cifra următoare, rezultatul este corectat prin adunarea valorii 6 la rezultat. De exemplu, considerăm următoarea adunare a două numere zecimale:

```
245+
474
6B9+       Suma intermediară
060
719        Suma zecimală finală
```

Suma cifrelor 4 şi 7 este mai mare decât 9; în acest caz, cifra sumă intermediară este corectată prin adunarea la aceasta a valorii 6.

Figura 4.7 prezintă un sumator zecimal bazat pe două sumatoare de câte patru biţi. Pe lângă cele două sumatoare de patru biţi, sumatorul mai cuprinde un număr de porţi pentru corecţia cifrelor sumă intermediare care sunt egale sau mai mari decât 10 (1010_2). Corecţia este realizată şi atunci când există un transport în urma adunării celor două cifre zecimale. Transportul de intrare T_{in} şi transportul de ieşire T_{out} pot fi utilizate pentru realizarea unui sumator zecimal pentru numere de mai multe cifre.

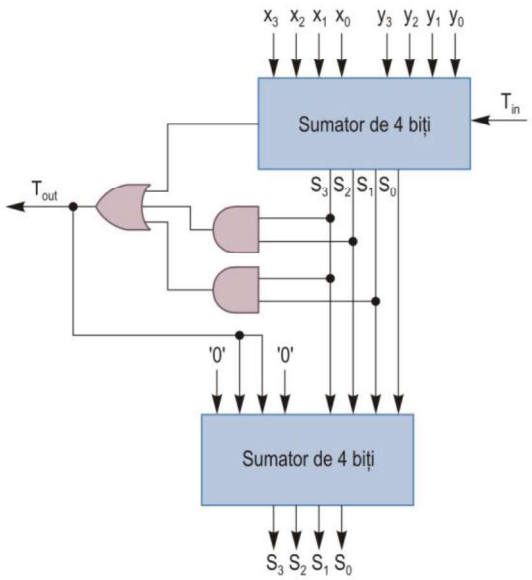

Figura 4.7. Schema bloc a unui sumator zecimal.

4.2 Circuite combinaționale de înmulțire

4.2.1 Înmulțirea matriceală

Comparativ cu circuitele secvențiale de înmulțire, circuitele combinaționale de înmulțire conțin o logică suplimentară care permite calcularea produsului într-o singură etapă. Un asemenea circuit de înmulțire este format dintr-o matrice de elemente combinaționale simple, fiecare din acestea implementând o operație de adunare și deplasare pentru un bit sau pentru un număr redus de biți.

Considerăm înmulțirea a două numere binare $X = x_{n-1}\ldots x_1x_0$ și $Y = y_{n-1}\ldots y_1y_0$. Pentru simplitate, presupunem că X și Y sunt numere întregi fără semn. Produsul P se poate exprima sub forma:

$$P = X * \left(\sum_{i=0}^{n-1} 2^i * y_i\right) \tag{4.20}$$

Această ecuație poate fi scrisă sub forma:

$$P = \sum_{i=0}^{n-1} 2^i * \left(\sum_{j=0}^{n-1} x_i * y_j * 2^j\right) \tag{4.21}$$

Fiecare din cei n^2 termeni produs de un bit x_i*y_j se poate calcula cu ajutorul unei porți ȘI cu două intrări, deoarece produsul aritmetic și cel logic coincid pentru un bit. Astfel, o matrice de $n \times n$ porți ȘI cu două intrări poate calcula toți termenii x_i*y_j simultan. Acești termeni sunt însumați cu ajutorul unei matrice de $n(n\text{-}1)$ sumatoare elementare, dintre care un număr de n sumatoare elementare pot fi înlocuite prin semisumatoare elementare. Circuitul rezultat este similar cu un sumator bidimensional cu transport succesiv. Deplasările implicate de factorii 2^i și 2^j în ecuația (4.21) sunt implementate prin deplasarea spațială a sumatoarelor pe direcția x și y.

Considerăm înmulțirea numerelor de câte patru biți $X = x_3x_2x_1x_0$ și $Y = y_3y_2y_1y_0$. Această înmulțire se efectuează după cum urmează:

				x_3	x_2	x_1	x_0 *
				y_3	y_2	y_1	y_0
0	0	0	0	x_3*y_0	x_2*y_0	x_1*y_0	x_0*y_0
0	0	0	x_3*y_1	x_2*y_1	x_1*y_1	x_0*y_1	0
0	0	x_3*y_2	x_2*y_2	x_1*y_2	x_0*y_2	0	0
0	x_3*y_3	x_2*y_3	x_1*y_3	x_0*y_3	0	0	0
P_7	P_6	P_5	P_4	P_3	P_2	P_1	P_0

Biții produsului final au expresiile booleene din ecuațiile (4.22) – (4.28), în care produsele aritmetice x_i*y_j sunt înlocuite cu produsele logice $x_i \cdot y_j$.

$$P_0 = x_0 \cdot y_0 \tag{4.22}$$

$$P_1 = x_1 \cdot y_0 + x_0 \cdot y_1 \tag{4.23}$$

$$P_2 = x_2 \cdot y_0 + x_1 \cdot y_1 + x_0 \cdot y_2 \tag{4.24}$$

$$P_3 = x_3 \cdot y_0 + x_2 \cdot y_1 + x_1 \cdot y_2 + x_0 \cdot y_3 \quad (4.25)$$

$$P_4 = x_3 \cdot y_1 + x_2 \cdot y_2 + x_1 \cdot y_3 \quad (4.26)$$

$$P_5 = x_3 \cdot y_2 + x_2 \cdot y_3 \quad (4.27)$$

$$P_6 = x_3 \cdot y_3 \quad (4.28)$$

Figura 4.8 prezintă o schemă posibilă pentru generarea și adunarea produselor parțiale. Dreptunghiurile notate cu x_iy_j reprezintă termeni produs de câte un bit generați cu porți ȘI. Semnalele de transport din fiecare rând de sumatoare elementare sunt conectate ca și la un sumator cu propagarea succesivă a transportului. Primul rând de sumatoare elementare adună primele două produse parțiale. Următoarele rânduri de sumatoare elementare adună câte un produs parțial la suma rezultată.

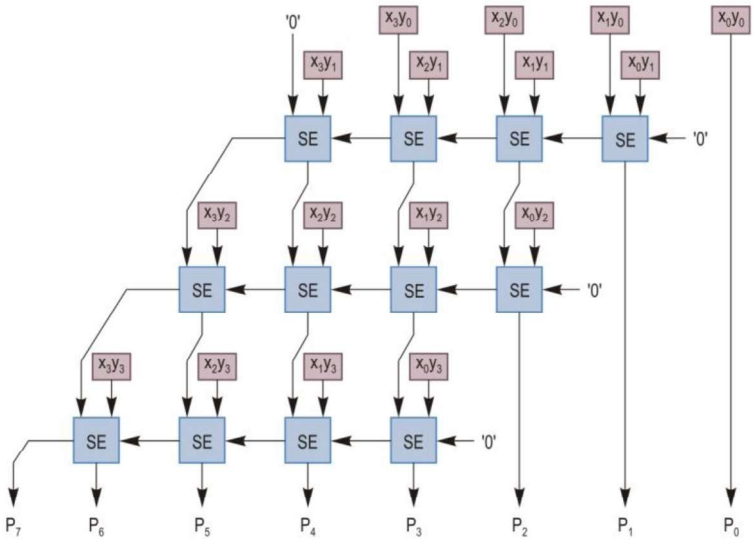

Figura 4.8. Matrice de sumatoare elementare pentru înmulțirea a două numere fără semn de câte patru biți.

Funcția de adunare și funcția logică ȘI necesară circuitului de înmulțire matriceală pot fi combinate într-o singură celulă, după cum se ilustrează în figura 4.9, în care celula este notată cu *M*.

Celula din figura 4.9 implementează următoarea expresie aritmetică, în care ***plus*** indică operatorul de adunare:

$$T_{out}S = a \; \textbf{\textit{plus}} \; xy \; \textbf{\textit{plus}} \; T_{in} \quad (4.29)$$

La intrarea *a* se conectează un bit al produsului parțial din linia precedentă de celule. Un circuit de înmulțire pentru $n \times n$ biți poate fi realizat prin utilizarea a n^2 celule de acest tip, deși unele celule de la periferia matricei vor avea intrările setate

la '0'. Timpul de execuție al operației de înmulțire este determinat de timpul de propagare al transportului pentru cazul cel mai defavorabil. Ignorând diferențele dintre celulele interne și cele periferice, acest timp este $(2n-1)t_p$, unde t_p este timpul de propagare al celulei de bază.

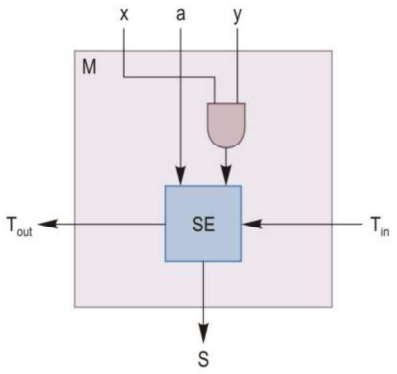

Figura 4.9. Celulă pentru înmulțirea matriceală a numerelor fără semn.

Pentru creșterea vitezei operației de înmulțire, circuitele de înmulțire matriceală pot fi realizate și prin utilizarea sumatoarelor cu salvarea transportului (SST). Viteza circuitelor realizate astfel este crescută deoarece timpul necesar adunării produselor parțiale este mai redus, propagarea transportului între sumatoarele elementare din același rând fiind eliminată. Propagarea transportului este amânată până la ultimul etaj al circuitului.

Figura 4.10 prezintă schema bloc a unui circuit de înmulțire matriceală pentru înmulțirea a două numere de câte opt biți utilizând sumatoare cu salvarea transportului. Dreptunghiurile notate cu Xy_0, Xy_1, ..., Xy_7 reprezintă produse parțiale de opt biți generate cu porți ȘI. Această structură necesită un număr de șase sumatoare cu salvarea transportului și un sumator cu propagarea transportului (SPT). Ieșirea sumă S și ieșirea de transport T a unui sumator cu salvarea transportului sunt aplicate la intrările următorului sumator cu salvarea transportului. Conexiunile de transport sunt deplasate la stânga cu o poziție pentru a corespunde propagării normale a transportului. Această schemă poate fi implementată în mod eficient într-un circuit VLSI datorită structurii sale uniforme.

Circuitul de înmulțire combinațional cu structura din figura 4.10 este practic pentru valori moderate ale numărului de biți n. Pentru valori mari ale lui n, numărul necesar de sumatoare cu salvarea transportului poate fi excesiv. Tehnica de adunare cu salvarea transportului poate fi utilizată totuși dacă sumatorul este partiționat în k segmente de câte m biți fiecare. În acest caz, sunt generate doar m produse parțiale, iar acestea sunt adunate utilizând sumatoare cu salvarea transportului. Procesul este repetat de k ori, sumele rezultate fiind acumulate.

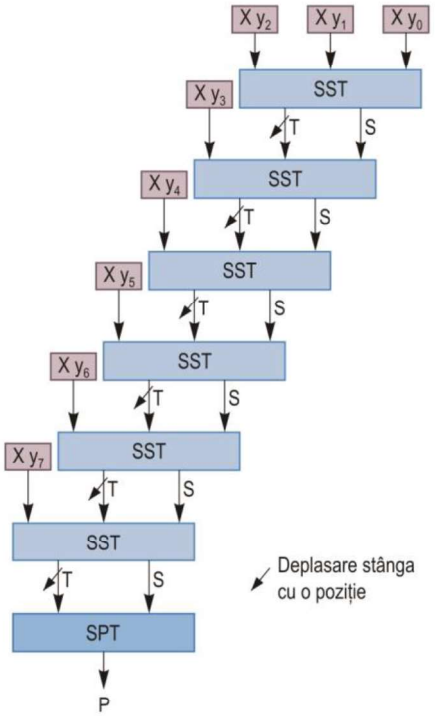

Figura 4.10. Circuit de înmulțire matriceală utilizând sumatoare cu salvarea transportului.

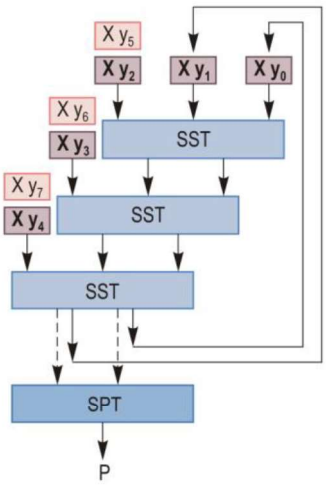

Figura 4.11. Circuit de înmulțire matriceală cu două treceri.

Figura 4.11 ilustrează o structură cu două treceri. La prima trecere, se efectuează înmulțirea celor cinci biți mai puțin semnificativi (intrările utilizate la prima trecere sunt indicate prin caractere aldine). Rezultatul obținut după prima trecere este transferat apoi la intrarea circuitului pentru a fi combinat cu următoarele trei produse parțiale. Rezultatul final este calculat apoi cu un sumator cu propagarea transportului.

Spre deosebire de circuitele de înmulțire matriceală anterioare, care sunt complet combinaționale, circuitul de înmulțire matriceală cu două treceri necesită, în mod normal, un semnal de ceas. Totuși, dacă matricea din figura 4.11 are o dimensiune suficient de mare astfel încât nici un semnal nu se poate propaga de la intrare la ieșire în timpul necesar primului sumator pentru stabilizarea ieșirilor sale, este posibilă evitarea utilizării unui semnal de ceas.

4.2.2 Arborele Wallace

În modul cel mai simplu, înmulțirea a două numere de câte n biți se poate efectua prin adunarea a n produse parțiale, după cum s-a prezentat anterior. Circuitele de înmulțire prezentate până acum execută înmulțirea într-un timp de ordinul valorii n a numărului de biți, $O(n)$. Acest timp poate fi redus la $O(\log n)$ prin utilizarea unui arbore. Arborele cel mai simplu ar combina perechi de produse parțiale, reducând numărul produselor parțiale de la n la $n/2$. Aceste produse parțiale ar fi adunate apoi două câte două, obținându-se suma finală după un număr de $\log_2 n$ etape. Acest arbore binar simplu nu poate fi implementat însă utilizând sumatoare elementare, care generează două ieșiri din trei intrări și nu o ieșire din două intrări.

Chris S. Wallace a arătat că produsele parțiale pot fi adunate într-un mod rapid și economic utilizând nivele multiple de sumatoare cu salvarea transportului, organizate într-o structură numită *arbore Wallace*. Presupunând că toate produsele parțiale sunt generate simultan, în primul nivel al arborelui produsele sunt grupate câte trei și se utilizează câte un sumator cu salvarea transportului pentru adunarea produselor din fiecare grup. Astfel, problema adunării a n produse parțiale se reduce la problema adunării a $2n/3$ produse parțiale. În al doilea nivel, cele $2n/3$ produse parțiale rezultate sunt grupate din nou câte trei și sunt adunate prin sumatoare cu salvarea transportului. Acest proces continuă până când rămân numai două produse de adunat. Pentru adunarea acestora, se utilizează un sumator cu propagarea transportului (de multe ori, un sumator cu anticiparea transportului). Deoarece fiecare nivel reduce numărul termenilor care trebuie adunați cu un factor de 1,5, înmulțirea poate fi terminată într-un timp proporțional cu $\log_{1,5} n$.

Figura 4.12 prezintă schema bloc pentru înmulțirea a două numere de câte opt biți utilizând un arbore Wallace. Primele șase produse parțiale $X y_i$ sunt grupate câte trei și sunt adunate cu două sumatoare cu salvarea transportului în primul nivel al arborelui. Cele două produse parțiale rămase sunt adunate în al doilea nivel cu sumele și transporturile de la primul nivel. Sumele și transporturile rezultate sunt grupate în continuare, iar suma și transportul final sunt adunate cu un sumator SPT.

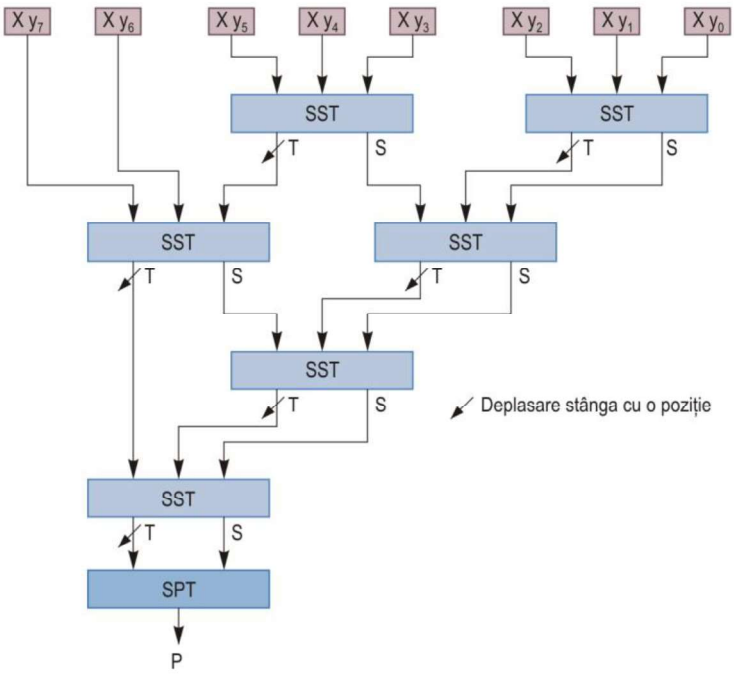

Figura 4.12. Circuit de înmulțire pentru numere de opt biți utilizând un arbore Wallace.

De multe ori, metoda arborelui Wallace este combinată cu alte metode, cum este metoda Booth (prezentată în capitolul 5, secțiunea 5.1.3), pentru a obține circuite rapide de înmulțire. Metoda Booth se utilizează pentru a genera produsele parțiale, iar apoi se utilizează un arbore Wallace pentru adunarea lor. De exemplu, presupunem înmulțirea a două numere de câte opt biți, $X = x_7 \ldots x_1x_0$ și $Y = y_7 \ldots y_1y_0$. Figura 4.13 prezintă o structură posibilă pentru înmulțirea acestor numere.

În primul nivel al arborelui din figura 4.13, cele opt blocuri generează produsele parțiale prin utilizarea metodei Booth. În cazul numerelor de opt biți, biții înmulțitorului care se utilizează pentru generarea fiecărui produs parțial sunt $y_0$0, y_1y_0, y_2y_1, y_3y_2, y_4y_3, y_5y_4, y_6y_5, y_7y_6. În funcție de cei doi biți, produsul parțial va fi 0 (dacă biții sunt 00 sau 11), X (dacă biții sunt 01) sau $-X$ (dacă biții sunt 10). Produsele parțiale sunt însumate apoi printr-un set de sumatoare cu salvarea transportului aranjate sub forma unui arbore Wallace.

Pentru reducerea circuitelor necesare în structura precedentă, de multe ori produsul final se obține prin mai multe treceri prin arborele Wallace. De exemplu, într-un circuit cu două treceri, în prima trecere se adună cele patru produse parțiale care rezultă din cei patru biți mai puțin semnificativi ai înmulțitorului. În a doua trecere, suma parțială și transportul se aplică la intrarea unui sumator din primul nivel al arborelui pentru a fi adunate cu următoarele patru produse parțiale.

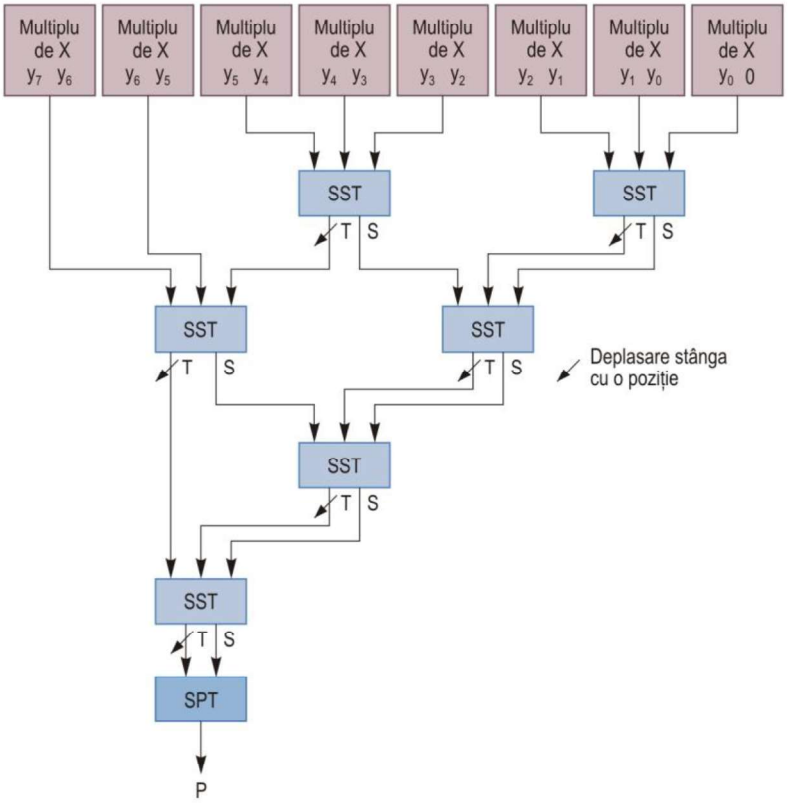

Figura 4.13. Circuit de înmulțire care combină metoda Booth cu un arbore Wallace.

Aplicații

4.1 Răspundeți la următoarele întrebări:

a. Care este principiul sumatorului cu anticiparea transportului?

b. Scrieți ecuațiile ieșirilor P și G ale unui sumator de doi biți utilizat pentru sumatorul de opt biți cu anticiparea transportului pe grupe de doi biți din figura 4.14. Mai întâi, scrieți ecuațiile funcțiilor de generare a transportului pe bit pentru biții 0 și 1 (g_0, g_1) și a funcțiilor de propagare a transportului pe bit pentru biții 0 și 1 (p_0, p_1). Utilizând aceste funcții, scrieți apoi ecuația funcției de generare a transportului pe grupul de biți 0, 1 ($G_{0,1}$) și a funcției de propagare a transportului pe grupul de biți 0, 1 ($P_{0,1}$).

c. Scrieți ecuațiile semnalelor de transport T_2, T_4, T_6 și T_8 pentru sumatorul de opt biți cu anticiparea transportului pe grupe de doi biți din figura 4.14. Utilizați

funcțiile de generare a transportului și de propagare a transportului pe grupe de doi biți.

d. Care este principiul arborelui Wallace?

4.2 Descrieți în limbajul VHDL sumatorul de opt biți cu anticiparea transportului pe grupe de doi biți din figura 4.14. Mai întâi, creați un modul VHDL pentru un sumator de doi biți cu anticiparea transportului. Porturile de intrare ale acestui modul sunt *X*(1:0), *Y*(1:0) și *Tin*, iar porturile de ieșire sunt *S*(1:0), *P* și *G*. Creați apoi un modul VHDL pentru sumatorul de opt biți, cu porturile de intrare *X*(7:0), *Y*(7:0), *Tin* și porturile de ieșire *S*(7:0), *Tout*. În acest modul, instanțiați direct entitatea sumatorului de doi biți și scrieți ecuațiile semnalelor de transport T_2, T_4, T_6 și T_8, care sunt ieșiri ale generatorului de transport anticipat de patru biți.

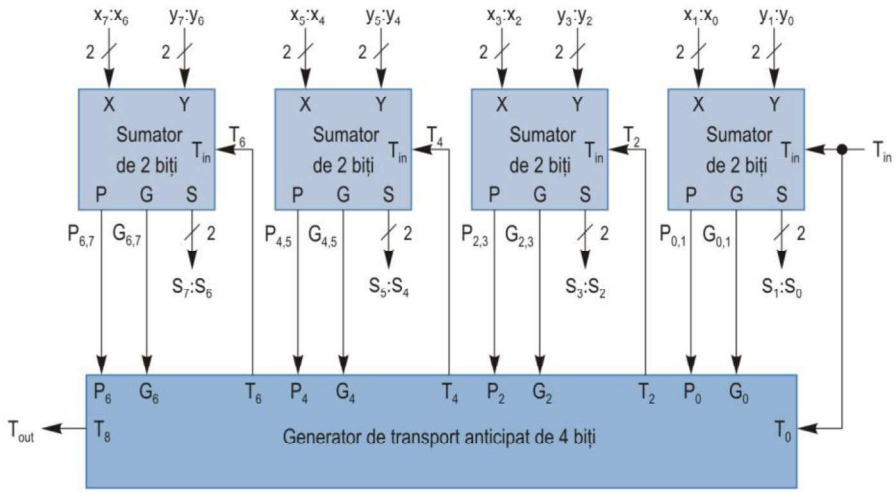

Figura 4.14. Schema bloc a unui sumator de opt biți cu anticiparea transportului pe grupe de doi biți.

Creați un fișier al bancului de test pentru sumatorul de opt biți. Generați mai întâi vectori de intrare cu valori apropiate de 0 (de exemplu, cu valori între 0 și 2), apoi cu valori apropiate de cele maxime (de exemplu, cu valori între 253 și 255) și cu valori oarecare între valorile minime și maxime. De fiecare dată, verificați dacă ieșirile generate sunt corecte. Simulați funcționarea sumatorului cu simulatorul Vivado.

Observație

Pentru generarea vectorilor de intrare ai sumatorului, utilizați funcția de conversie **CONV_STD_LOGIC_VECTOR**, disponibilă în pachetul **STD_LOGIC_ARITH** din biblioteca IEEE. Această funcție convertește o valoare întreagă specificată

de primul parametru al funcției într-o valoare de tip **STD_LOGIC_VECTOR**, pe un număr de biți specificat de al doilea parametru al funcției. De exemplu, funcția se poate utiliza pentru conversia indecșilor din buclele `for .. loop` în vectori de test.

4.3 Implementați pe o placă de dezvoltare sumatorul de opt biți cu anticiparea transportului creat pentru aplicația 4.2. În continuare se presupune utilizarea plăcii de dezvoltare Nexys4 DDR sau Nexys A7-100T, dar se poate utiliza orice placă cu 16 comutatoare, un buton și un afișaj cu șapte segmente conținând patru sau mai multe cifre. Perifericele plăcii vor fi utilizate în felul următor:

- Primul operand (*X*) va fi introdus de la primele opt comutatoare SW15..SW8 (din partea stângă).
- Al doilea operand (*Y*) va fi introdus de la celelalte opt comutatoare SW7..SW0 (din partea dreaptă).
- Semnalul de resetare (necesar doar pentru multiplexorul afișajului cu șapte segmente) va fi generat prin butonul BTND.
- Starea comutatoarelor va fi afișată pe primele patru cifre (din partea stângă) ale afișajului cu șapte segmente.
- Pe a cincea cifră a afișajului cu șapte segmente se va afișa 0, iar pe a șasea cifră se va afișa transportul de ieșire T_{out} (0 sau 1).
- Suma va fi afișată pe ultimele două cifre (din partea dreaptă) ale afișajului cu șapte segmente.

Parcurgeți următoarele etape pentru implementarea sumatorului.

1. Adăugați la proiect fișierul sursă al multiplexorului pentru afișajul cu șapte segmente displ7seg creat pentru aplicația 1.3 din capitolul 1.
2. Creați un fișier sursă VHDL pentru modulul principal al sumatorului. Porturile de intrare ale acestui modul vor fi *Clk*, *Rst*, *X*(7:0) și *Y*(7:0) (porturile *Clk* și *Rst* vor fi necesare doar pentru multiplexorul afișajului cu șapte segmente). Porturile de ieșire ale modulului principal vor fi *An*(7:0) și *Seg*(7:0), necesare afișajului cu șapte segmente.
3. În fișierul sursă al modulului principal, instanțiați entitatea sumatorului de opt biți. Aplicați valoarea logică '0' la intrarea de transport T_{in} a sumatorului și declarați semnalele care trebuie conectate la porturile sale de ieșire. Declarați un semnal *Data* care se va conecta la portul cu același nume al multiplexorului pentru afișajul cu șapte segmente. Asignați la acest semnal valorile care trebuie afișate, în modul specificat anterior. Instanțiați apoi entitatea multiplexorului cu șapte segmente displ7seg.

4. Adăugați la proiect fișierul de constrângeri al plăcii de dezvoltare, care a fost utilizat pentru proiectul ceasului de timp real din capitolul 1, și deschideți pentru editare acest fișier. Ștergeți caracterele # de la începutul liniilor corespunzătoare comutatoarelor SW[0] .. SW[15] și modificați numele porturilor pentru ca acestea să corespundă cu intrările *Y* și *X* ale sumatorului. Comentați liniile corespunzătoare butoanelor, cu excepția butonului BTND (la care a fost conectat portul *Down* al ceasului de timp real) și modificați numele portului din *Down* în *Rst*.
5. Realizați elaborarea proiectului și corectați erorile, dacă există.
6. Setați opțiunile pentru sinteza și implementarea proiectului. Pentru sinteză, selectați strategia de rulare *Flow_RuntimeOptimized*, iar pentru implementare selectați strategia de rulare *Flow_Quick*.
7. Realizați sinteza și implementarea proiectului, după care generați fișierul cu șirul de biți pentru configurarea circuitului FPGA.
8. Conectați placa de dezvoltare la un port USB al calculatorului și verificați funcționarea sumatorului.

4.4 Modificați descrierea sumatorului de opt biți cu anticiparea transportului pe grupe de doi biți creat pentru aplicația 4.2 astfel încât să utilizați instrucțiunea `for generate` la instanțierea sumatorului de doi biți și generarea semnalelor de transport.

Observație

Recapitulați instrucțiunea `for generate`, care este prezentată în Anexa C, *Proiectarea structurală în limbajul VHDL*, secțiunea C.6.1.

4.5 Descrieți în limbajul VHDL sumatorul de 16 biți cu anticiparea transportului pe grupe de patru biți din figura 4.4. Simulați funcționarea sumatorului cu simulatorul Vivado.

4.6 Descrieți în limbajul VHDL un circuit de înmulțire matriceală pentru două numere de câte opt biți fără semn. Creați mai întâi un modul VHDL pentru un sumator elementar. Creați apoi modulul VHDL pentru circuitul de înmulțire, cu porturile de intrare *X*(7:0), *Y*(7:0) și portul de ieșire *P*(15:0). Utilizați semnalele notate ca în figura 4.15.

Definiți semnalele *PP*, *S* și *T* ca tablouri de vectori. Utilizați instrucțiunea `for generate` pentru generarea produselor parțiale, pentru inițializarea semnalelor de la marginile matricei și pentru generarea matricei de sumatoare elementare. Creați un fișier al bancului de test pentru circuitul de înmulțire. Generați vectori de intrare cu valori apropiate de 0, vectori cu valori apropiate de cele maxime și vectori cu valori oarecare între valorile minime și maxime, verificând de fiecare

dată dacă ieșirile generate sunt corecte. Simulați funcționarea circuitului de înmulțire cu simulatorul Vivado.

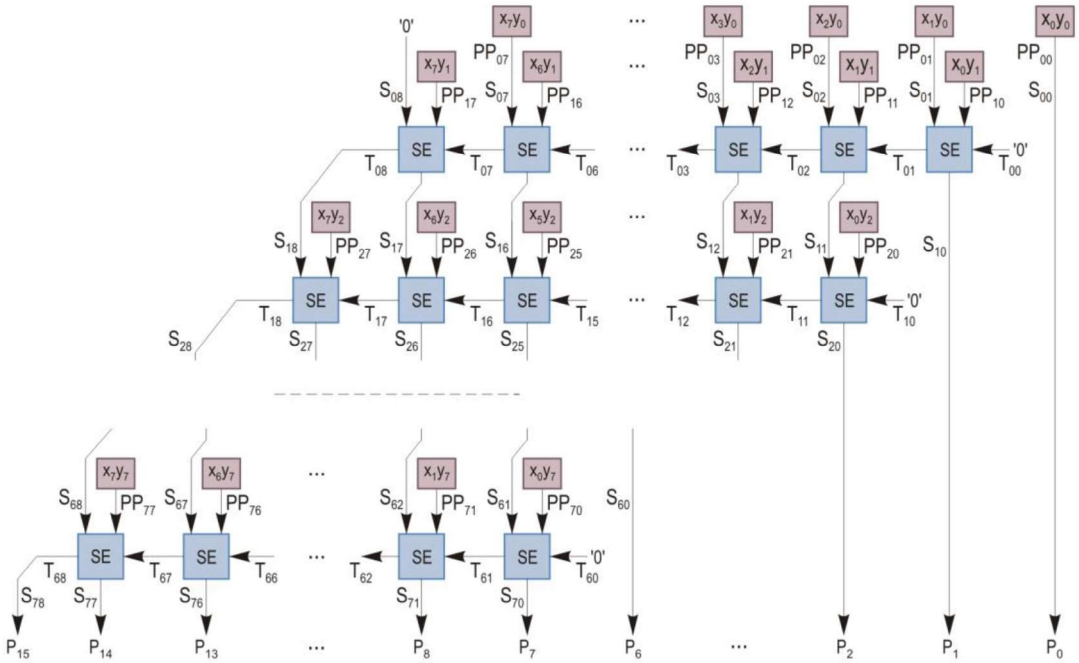

Figura 4.15. Denumirea semnalelor utilizate de circuitul de înmulțire matriceală pentru numere de câte opt biți.

4.7 Implementați pe o placă de dezvoltare circuitul de înmulțire matriceală de opt biți creat pentru aplicația 4.6. În continuare se presupune utilizarea plăcii de dezvoltare Nexys4 DDR sau Nexys A7-100T, dar se poate utiliza orice placă cu 16 comutatoare, un buton și un afișaj cu șapte segmente conținând patru sau mai multe cifre. Perifericele plăcii vor fi utilizate în felul următor:

- Deînmulțitul *X* va fi introdus de la primele opt comutatoare SW15..SW8 (din partea stângă).
- Înmulțitorul *Y* va fi introdus de la celelalte opt comutatoare SW7..SW0 (din partea dreaptă).
- Semnalul de resetare (necesar doar pentru multiplexorul afișajului cu șapte segmente) va fi generat prin butonul BTND.
- Starea comutatoarelor va fi afișată pe primele patru cifre (din partea stângă) ale afișajului cu șapte segmente.

- Produsul va fi afișat pe ultimele patru cifre (din partea dreaptă) ale afișajului cu șapte segmente.

Parcurgeți următoarele etape pentru implementarea circuitului de înmulțire.

1. Adăugați la proiect fișierul sursă al multiplexorului pentru afișajul cu șapte segmente `displ7seg` creat pentru aplicația 1.3 din capitolul 1.
2. Creați un fișier sursă VHDL pentru modulul principal al circuitului de înmulțire. Porturile de intrare ale acestui modul vor fi *Clk*, *Rst*, *X*(7:0) și *Y*(7:0) (porturile *Clk* și *Rst* vor fi necesare doar pentru multiplexorul afișajului cu șapte segmente). Porturile de ieșire ale modulului principal vor fi *An*(7:0) și *Seg*(7:0), necesare afișajului cu șapte segmente.
3. În fișierul sursă al modulului principal, instanțiați entitatea circuitului de înmulțire de opt biți. Declarați un semnal *Data* care se va conecta la portul cu același nume al multiplexorului pentru afișajul cu șapte segmente. Asignați la acest semnal valorile care trebuie afișate, în modul specificat anterior. Instanțiați apoi entitatea multiplexorului cu șapte segmente `displ7seg`.
4. Adăugați la proiect fișierul de constrângeri al plăcii de dezvoltare, care a fost utilizat în proiectul sumatorului de opt biți cu anticiparea transportului, creat pentru aplicația 4.3.
5. Realizați elaborarea proiectului și corectați erorile, dacă există.
6. Setați opțiunile pentru sinteza și implementarea proiectului. Pentru sinteză, selectați strategia de rulare *Flow_RuntimeOptimized*, iar pentru implementare selectați strategia de rulare *Flow_Quick*.
7. Realizați sinteza și implementarea proiectului, după care generați fișierul cu șirul de biți pentru configurarea circuitului FPGA.
8. Conectați placa de dezvoltare la un port USB al calculatorului și verificați funcționarea circuitului de înmulțire.

4.8 Modificați descrierea circuitului de înmulțire matriceală creat pentru aplicația 4.6 astfel încât să utilizați tehnica de adunare cu salvarea transportului la adunarea produselor parțiale. În ultimul etaj al circuitului, utilizați un sumator cu propagarea succesivă a transportului. Creați un fișier al bancului de test și simulați funcționarea circuitului de înmulțire cu simulatorul Vivado.

4.9 Descrieți în limbajul VHDL un circuit de înmulțire matriceală pentru două numere de câte opt biți fără semn. Utilizați celula de înmulțire din figura 4.9, fără alte componente sau porți logice. Pentru însumarea produselor parțiale utilizați tehnica de adunare cu propagarea succesivă a transportului. Creați un fișier al bancului de test și simulați funcționarea circuitului de înmulțire cu simulatorul Vivado.

4.10 Descrieți în limbajul VHDL sumatorul zecimal din figura 4.7. Utilizați acest sumator pentru a realiza un sumator zecimal pentru două numere de câte patru cifre. Creați un fișier al bancului de test și simulați funcționarea sumatorului zecimal cu simulatorul Vivado.