**Sumator și Multiplicator**

În entitatea principală (muxSelectie.vhd) am realizat un multiplexor 2:1 care permite alegerea efectuării operației de adunare cu ajutorul sumatorului carry save, sau a operației de înmulțire pe baza algoritmului Wallace Tree. Putem alege care operație dorim să fie efectuată, pe baza semnalului „sel”, corespunzător pin-ului W2 (al 12-lea switch). Atunci când acest semnal de selecție este 0, se va efectua înmulțirea, iar când este 1, adunarea. De asemenea, primele 12 switch-uri sunt folosite pentru numerele introduse de utilizator (în binar) – algoritmul de adunare folosește toți cei 12 biți (switch-urile 11-0), iar la înmulțire sunt folosiți doar primii 8 (switch-urile 7-0). Pentru ieșire am folosit astfel: la adunare: primele 4 leduri pentru calcularea sumei (led3-0), iar ledul 8 pentru cout; la înmulțire: primele 8 leduri (led7-0) pentru afișarea produsului.

O imagine care conține diagramă, text, Plan, linie

Descriere generată automat

1. **Carry Save Adder pe 4 biți**

Am ales să implementez sumatorul cu salvarea transportului, deoarece este eficient pentru adunarea a mai mult de două numere, reducând timpul de propagare al semnalelor de transport. Pe intrarea acestui sumator se regăsesc cele 3 numere pe 4 biți, x, y și z care trebuie adunate. De aceea, se folosesc 4 full addere independente, semnalele de transport nefiind propagate între aceste sumatoare. Deci, toți biții sumă și toți biții de transport pot fi generați în paralel, ceea ce îmbunătățește viteza de adunare.

În progamul meu: prov\_carry = transportul parțial, prov\_sum = suma parțială și se calculează astfel:

prov\_carry = Xi xor Yi xor Zi

prov\_sum = XiYi + XiZi + YiZi

Am încercat să exemplific funcționarea sumatorului pentru numerele 9, 8 și 3:

O imagine care conține text, scris de mână, hârtie, cerneală

Descriere generată automat

Schema generată în Vivado:

O imagine care conține text, diagramă, Plan, linie

Descriere generată automat

Am efectuat pe plăcuța Basys3, 5 teste, explicând corespondența fiecărui bit:

* x: 3 = 0011, y: 8 = 1000, z: 9 = 1001=> sum = 0100, cout = 1 -> 20

O imagine care conține text, electronice, Inginerie electronică, Componentă electronică

Descriere generată automat

* x: 15 = 1111, y: 13 = 1101, z: 3 = 0011=> sum = 1111, cout = 1 -> 31

O imagine care conține text, electronice, Inginerie electronică, circuit

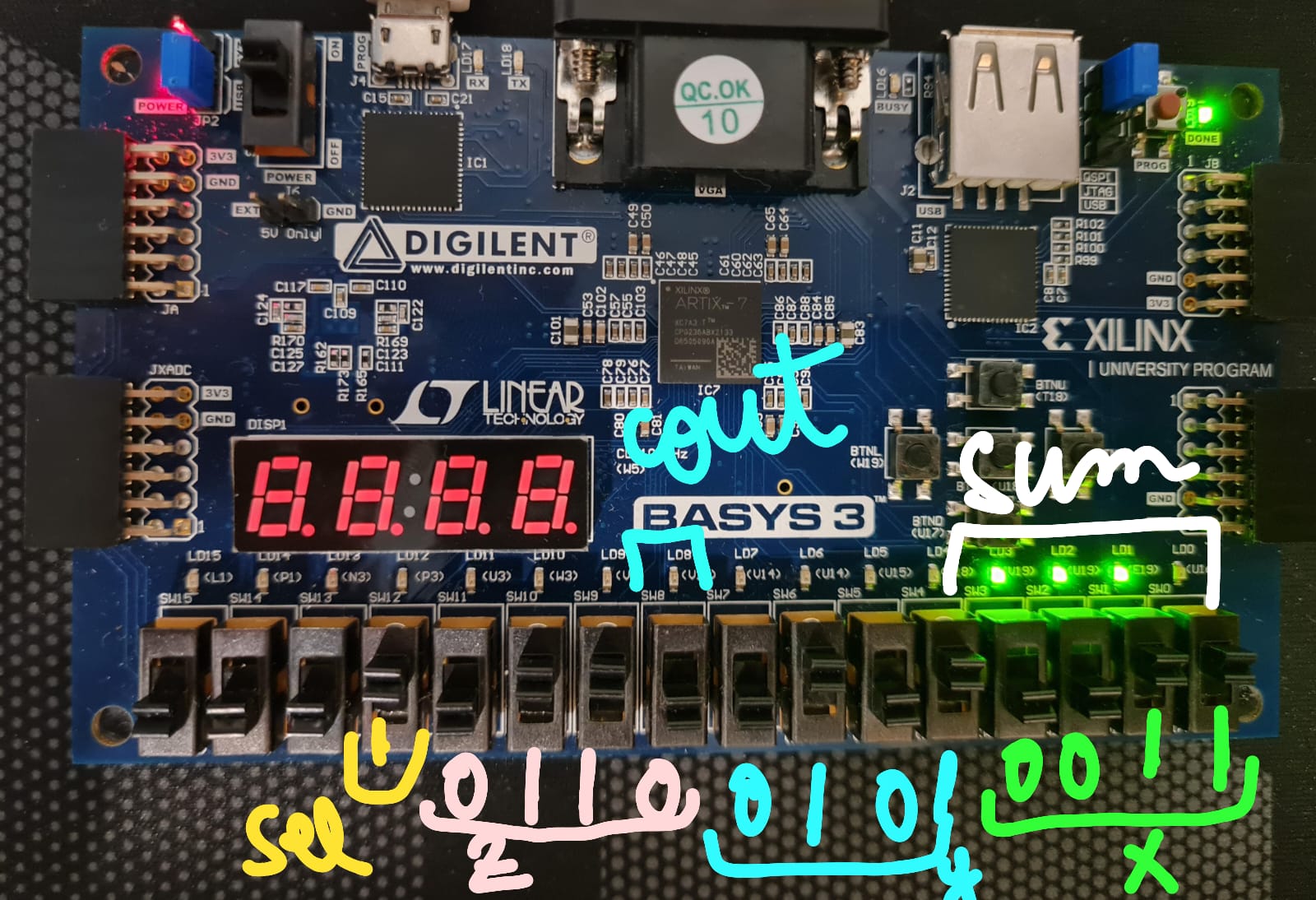
Descriere generată automat

* x: 14 = 1110, y: 6 = 0110, z: 2 = 0010=> sum = 0110, cout = 1 -> 22

O imagine care conține text, electronice, Inginerie electronică, circuit

Descriere generată automat

* x: 3 = 0011, y: 5 = 0101, z: 6 = 0110=> sum = 1110, cout = 0 -> 14



* x: 7 = 0111, y: 8 = 1000, z: 1 = 0001=> sum = 0000, cout = 1 -> 16

O imagine care conține text, electronice, circuit, Inginerie electronică

Descriere generată automat

1. **Multiplicatorul Wallace Tree**

Prin utilizarea unui arbore în înmulțirea a două numere de n biți se poate reduce timpul de execuție la O(log n), față de timpul obișnuit în care se efectuează această operație (prin adunarea a n produse parțiale): O(n).

O imagine care conține diagramă, Plan, text, hartă

Descriere generată automat

Nu folosesc semnal de ceas pentru realizarea acestui algoritm de multiplicare, dar după ce trec la următorul bit din înmulțitor, fac o shiftare a biților manuală. Câteva explicații suplimentare se găsesc mai jos:

O imagine care conține text, scris de mână, tablă albă de scris, cerneală

Descriere generată automat

Am efectuat 5 teste pe plăcuța Basys3:

* x: 4 = 0100, y: 5 = 0101 => p = 0001 0100 = 20

O imagine care conține text, electronice, Inginerie electronică, circuit

Descriere generată automat

* x: 11 =1011, y: 13 = 1101 => p = 1000 1111 = 143

O imagine care conține electronice, circuit, Inginerie electronică, text

Descriere generată automat

* x: 14 = 1110, y: 3 = 0011 => p = 0010 1010 = 42

O imagine care conține text, electronice, Inginerie electronică, circuit

Descriere generată automat

* x: 6 = 0110, y: 4 = 0100 => p = 0001 1000 = 24

O imagine care conține text, electronice, Inginerie electronică, circuit

Descriere generată automat

* x: 15 = 1111, y: 14 = 1110 => p = 1101 0010 = 210

