DOCUMENTAȚIE

Proiectarea unei unități de calcul în virgulă flotantă: inmultirea

NUME STUDENT : IRIMIE DARIA-ALEXIA

GRUPA : 30238

# CUPRINS

[1. Introducere 3](#_Toc117899591)

[1.1 Context 3](#_Toc117899592)

[1.2 Specificații 3](#_Toc117899593)

[1.3 Obiective 4](#_Toc117899594)

[2. Proiectare 5](#_Toc117899595)

[2.1 Numerele în virgulă flotantă 5](#_Toc117899596)

[2.2 Design 6](#_Toc117899597)

[2.2.1 Descrierea flowchart-ului 6](#_Toc117899599)

2.2.2 Design-ul complet .……………………………………………………………………....7

[3. Implementare 10](#_Toc117899600)

[4. Manual de utilizare 12](#_Toc117899601)

5. Testare ……………..………………………………………………………………………12

[6. Concluzii 17](#_Toc117899602)

[7. Bibliografie 17](#_Toc117899603)

# Introducere

# Context

Acest proiect implementează un multiplicator pentru numere în virgulă mobilă, bazat pe reprezentarea standard IEEE 754 a numerelor în virgulă mobilă în format binar.

Numarul în virgulă mobilă este un format de reprezentare a numerelor reale care le împarte în trei componente principale: semnul, exponentul și mantisa. Reprezentarea IEEE 754 a numerelor în virgulă mobilă utilizează aceste componente pentru a permite reprezentarea unui interval larg de valori cu precizie variabilă.

În cadrul acestui proiect, se lucrează cu numere de tip float pe 32 de biți. Aceste numere sunt reprezentate în formatul IEEE 754, unde 1 bit este alocat pentru semn, 8 biți pentru exponent și 23 de biți pentru mantisă. Multiplicatorul implementat aici primește două astfel de numere și furnizează rezultatul produsului lor, respectând specificațiile IEEE 754. Proiectul este structurat în mai multe module care gestionează operațiile de înregistrare a multiplicandului și a multiplicatorului, efectuarea operației de înmulțire, și rotunjirea rezultatului pentru a se încadra în formatul IEEE 754. Multiplicatorul are și funcționalitatea de resetare și semnalizare a finalizării operației.

# Specificații

* Reprezentare IEEE 754: Proiectul utilizează formatul IEEE 754 pentru reprezentarea numerelor în virgulă mobilă pe 32 de biți, cu 1 bit pentru semn, 8 biți pentru exponent și 23 de biți pentru mantisă.
* Multiplicare în virgulă mobilă: Scopul principal al proiectului este să efectueze operația de multiplicare între două numere în format IEEE 754 și să furnizeze rezultatul în același format.
* Operare pe 32 de biți: Proiectul este optimizat pentru operațiile pe numere în virgulă mobilă reprezentate pe 32 de biți conform standardului IEEE 754.
* Utilizarea modulelor auxiliare: Proiectul este divizat în module distincte pentru a gestiona înregistrarea multiplicandului și a multiplicatorului, efectuarea operației de înmulțire, și manipularea rezultatului pentru a respecta formatul IEEE 754.
* Resetare și semnalizare: Multiplicatorul include funcționalitatea de resetare, care aduce sistemul într-o stare inițială, și semnalizarea finalizării operației.

# Obiective

* Eficiență în spațiu: Proiectul este proiectat pentru a utiliza eficient resursele disponibile, astfel încât să ofere o implementare compactă și optimizată.
* Performanță: Multiplicatorul trebuie să furnizeze rezultate precise și eficiente într-un timp rezonabil pentru operațiile de multiplicare în virgulă mobilă.
* Conformitate IEEE 754: Rezultatele furnizate de multiplicator trebuie să respecte standardul IEEE 754 pentru a asigura compatibilitatea cu alte dispozitive și software care utilizează acest standard.
* Ușurința în integrare: Proiectul trebuie să fie ușor de integrat în sistemele mai mari care utilizează operații de înmulțire în virgulă mobilă.

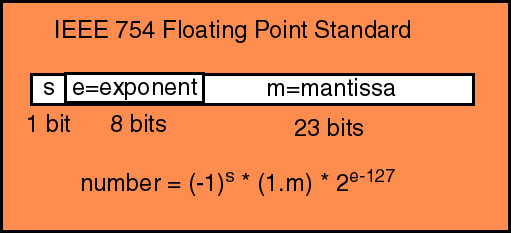
# PROIECTARE

# Numerele în virgulă flotantă

În concordanță cu standardul IEEE, numerele în virgulă flotantă cu simplă precizie sunt reprezentate astfel folosind:

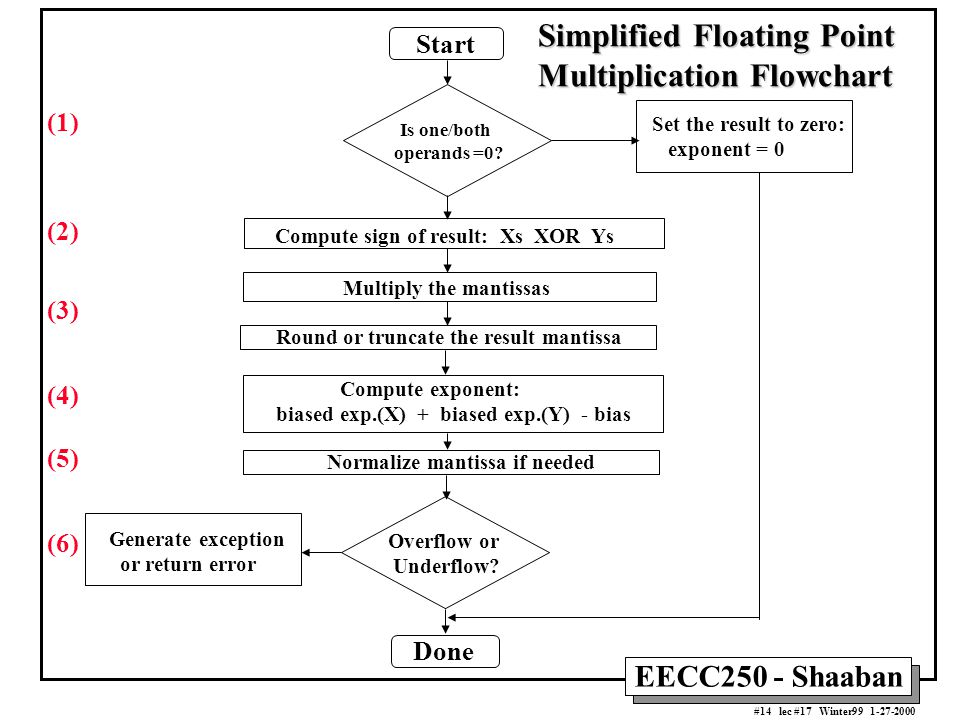
* 1 bit pentru semn
* 8 biți pentru exponent
* 23 de biți pentru mantisă

Numărul se calculează cu ajutorul următoarei formule:



# Design

Înainte să vorbim despre design–ul propriu–zis al proiectului, trebuie făcută o analiză a operatiei.



# Descrierea flowchart-ului

Pentru această operație vom avea nevoie de o componentă de înmulțire a mantiselor și de un sumator pentru exponenți.

Pentru înmulțirea mantiselor, concatenam 1 ca cel mai semnificativ bit pentru ambele mantise și trimitem rezultatele la înmulțitor. Rezultatul va fi astfel pe 48 de biți.

Avem astfel 2 cazuri:

- dacă primul bit de la rezultat este 1 atunci luăm următorii 23 de biți și normalizăm rezultatul prin adăugarea unui 1 la exponent

- dacă bitul cel mai semnificativ este 0, luăm următorii 23 de biți.

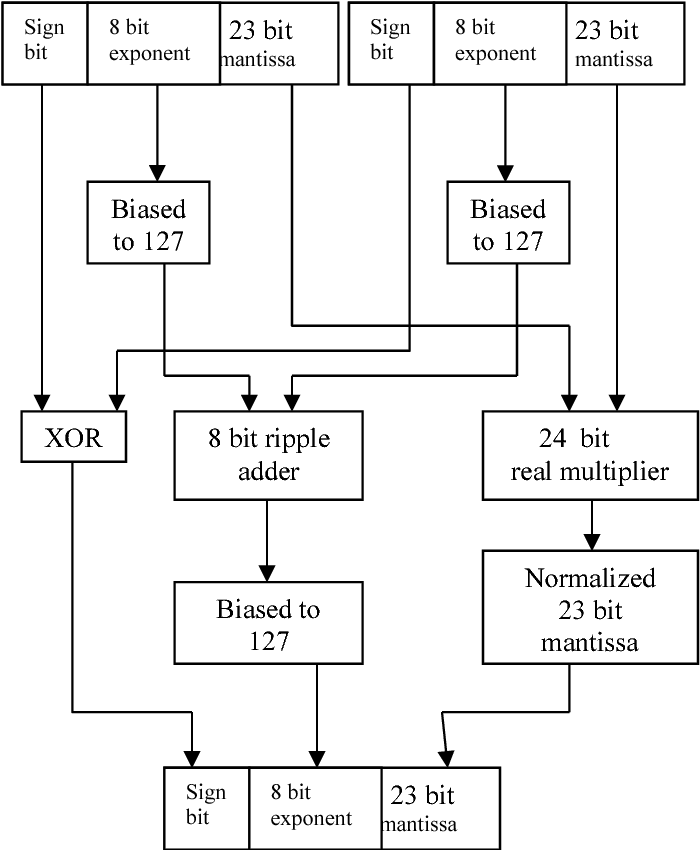
Pentru a realiza adunarea exponenților se va folosi un sumator pe 8 biți, iar pentru aflarea semnului operației, se va folosi operația XOR între semnele numerelor.

În ceea ce privește componenta de înmulțire, se va folosi tehnica de Shift and Add.

# Design-ul complet

Întreaga componentă de înmulțire a numerelor în virgulă flotantă este descrisă de urmatoarea diagramă.

La înmulțire, dacă ambii operanzi sunt foarte mari, adunarea exponenților va cauza overflow, dând un rezultat foarte mic, iar dacă sunt foarte mici, se va obține un rezultat foarte mare.



În ceea ce privește componenta de înmulțire, se va folosi tehnica de Shift and Add, descrisă de figura de mai jos:

A diagram of a flowchart

Description automatically generated

Schema completă în mediul vivado este următoarea :

A diagram of a computer

Description automatically generated

# Implementare

* Desfacerea Operanzilor:
* Semn: În primul rând, se desface semnul din reprezentarea în virgulă mobilă a ambilor operanzi (input1 și input2).
* Exponent: Se extrage exponențialul de 8 biți al fiecărui operand pentru a putea efectua operații ulterioare.
* Mantisa: Mantisa de 23 de biți a fiecărui operand este separată, reprezentând partea fraccională a numărului.
* Extinderea Mantisei:
* Mantisa operandului input1 este extinsă la 48 de biți prin adăugarea a 24 de biți de zero la stânga.
* Mantisa operandului input2 este extinsă la 24 de biți prin adăugarea unui bit de '1' la stânga.
* Multiplicarea:
* Se utilizează un bloc specializat numit Multiplicator pentru a realiza operația de înmulțire a mantiselor extinse ale operanzilor (mantissaInput1\_WITHONE și mantissaInput2\_WITHONE).
* Multiplicatorul produce un rezultat într-un vector numit multiplicationResult și semnalează când operația de înmulțire este finalizată prin semnalul done.
* Calculul Exponenților:
* Se calculează noul exponențial al rezultatului prin adunarea exponențialilor celor doi operanzi și ajustarea corespunzătoare a rezultatului în funcție de bitul semnificativ obținut în urma înmulțirii.
* Rotunjirea Mantisei:
* Mantisa rezultatului este rotunjită în funcție de valoarea bitului semnificativ obținut în urma înmulțirii, respectând regulile standardului IEEE 754.
* Formarea Rezultatului Final:
* Rezultatul final este compus din semnul rezultatului, noul exponențial și mantisa rezultatului rotunjit. În cazurile speciale în care unul dintre operanzi este zero, rezultatul final este setat la zero.

**Shift and Add Method:**

Metoda "shift and add" este o tehnică utilizată în operațiile de înmulțire și împărțire în reprezentarea în virgulă mobilă. Această tehnică este eficientă și ușor de implementat în hardware, oferind o soluție pentru efectuarea acestor operații.

**Procesul de Înmulțire:**

* Shift (Deplasare): La fiecare pas al procesului de înmulțire, mantisa operandului multiplicat (multiplicand) este deplasată la stânga cu un bit (shift left). Acest lucru corespunde multiplicării mantisei cu 2 la fiecare pas.
* Adunare (Add): Dacă bitul de cel mai la dreapta al multiplicatorului este '1', atunci mantisa operandului multiplicator (multiplier) este adunată la rezultatul parțial. Altfel, nu se efectuează nicio adunare.

**Motivația Alegerii Metodei:**

* Eficiență în Hardware: Această metodă este prietenoasă cu hardware-ul și este ușor de implementat în circuite digitale.
* Sintetizare Rapidă: Tehnica shift and add poate fi sintetizată într-un număr redus de porți logice și registre, ceea ce face procesul eficient.
* Paralelism Redus: În timp ce poate exista o limitare a paralelismului în comparație cu alte metode, cum ar fi algoritmul Booth, aceasta poate fi compensată prin optimizări.

# MANUAL DE UTILIZARE

Se foloște mediul de dezvoltare Vivado.

**Utilizarea Circuitului:**

Furnizați valorile numerelor de intrare (input1 și input2) în formatul IEEE 754 (32 de biți fiecare). Stabiliți semnalul load pentru a încărca valorile de intrare în circuit. Furnizați semnalul de ceas (clk) pentru sincronizare. Așteptați semnalul done pentru a indica finalizarea operației de multiplicare. Citiți rezultatul din ieșirea output, care este reprezentat în formatul IEEE 754.

# Testare

S-au realizat testbench-uri pentru componenta de înmulțire, simulate în Vivado :

A screen shot of a computer

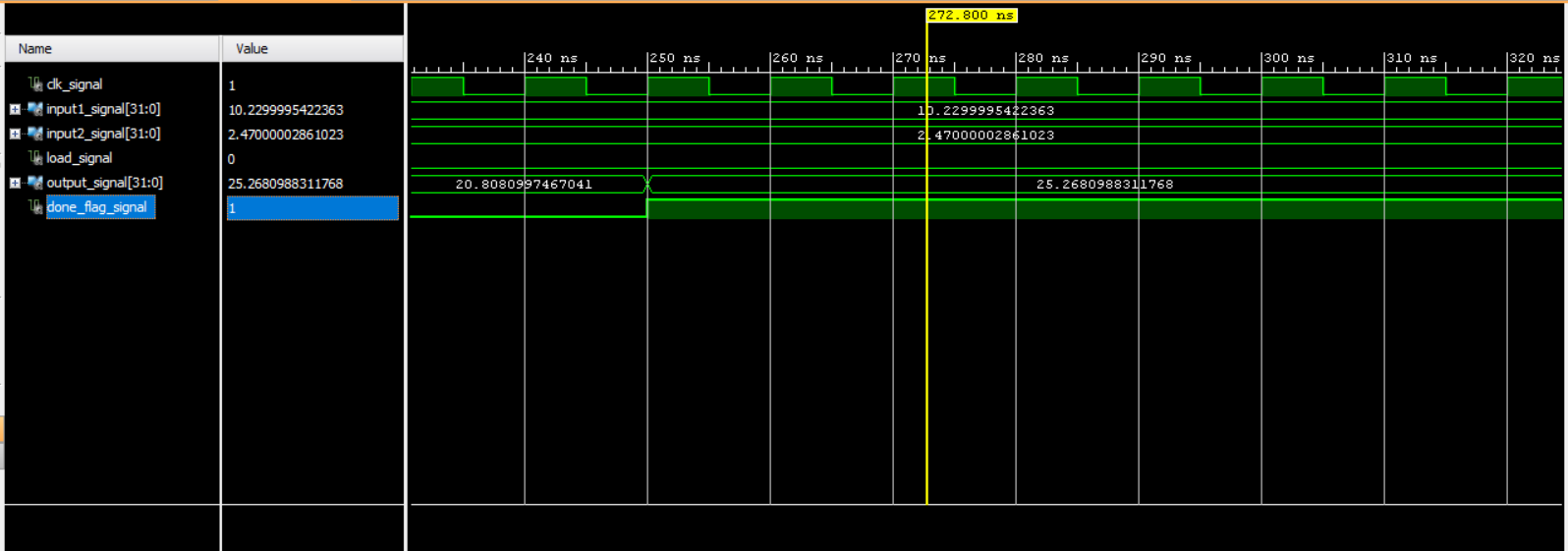
Description automatically generated

Valori simulate :

input1\_signal <= x"c123ae14"; -- -10.23

input2\_signal <= x"c01e147b"; -- -2.47

(positive)



Valori simulate :

input1\_signal <= x"4123ae14"; --10.23

input2\_signal <= x"401e147b"; -- 2.47

(negative)

A screen shot of a computer

Description automatically generated

Valori simulate :

input1\_signal <= x"c123ae14"; -- -10.23

input2\_signal <= x"401e147b"; -- 2.47

(input1 cu valoare negative, iar input2 cu valoare pozitiva)

A screen shot of a computer

Description automatically generated

Valori simulate :

input1\_signal <= x"c123ae14"; -- 10.23

input2\_signal <= x"401e147b"; -- -2.47

(input1 cu valoare pozitiva, iar input2 cu valoare negativa)

# Concluzii

În final, proiectul de implementare a unității de înmulțire pentru numere în virgulă mobilă aduce o soluție eficientă și flexibilă pentru operațiile matematice într-un sistem digital. Standardul IEEE 754, care definește reprezentarea și operațiile numerelor în virgulă mobilă, impune cerințe stricte pentru precizie și corectitudine. Proiectul urmărește aceste cerințe, asigurând rezultate consistente și fiabile în cadrul operațiilor de înmulțire.

Implementarea modulară a unității, folosind registre specializate pentru multiplicand, multiplicator și produs, permite o gestionare eficientă a datelor și a operațiilor specifice înmulțirii. Elementele de control și logica de decizie asigură o coordonare precisă a acestor operații, contribuind la obținerea unor rezultate corecte în diverse scenarii de utilizare. De asemenea, detaliile tehnice precum modul de gestionare a exponenților, rotunjirea mantiselor și operațiile de shift aduc un plus de complexitate și acuratețe unității de înmulțire. Aceste caracteristici permit adaptarea proiectului la cerințele specifice ale aplicațiilor care necesită manipularea numerelor în format IEEE 754.

Într-un final, prin integrarea tuturor acestor elemente, proiectul reușește să ofere o soluție completă și performantă pentru operațiile de înmulțire în virgulă mobilă, aducând beneficii semnificative în domeniile în care precizia și eficiența operațiilor numerice sunt cruciale.

# Bibliografie

<https://www.geeksforgeeks.org/ieee-standard-754-floating-point-numbers/>

<https://www.geeksforgeeks.org/multiplying-floating-point-numbers/?ref=gcse>

<https://www.rfwireless-world.com/Tutorials/floating-point-tutorial.html>

<https://electronics.stackexchange.com/questions/20791/what-is-the-sequence-and-how-it-is-going>

<https://www.semanticscholar.org/paper/Implementation-of-Single-Precision-Floating-Point-Chaitanya/cfd118aa92262dc032b3e2ac4ecd12715aa1f947>

<https://allaboutfpga.com/>

<http://lslwww.epfl.ch/pages/teaching/cours_lsl/sl_info/FPMultiplier.pdf>