**Proiect PSN**

**Calculator de buzunar**

Grupa 30212,semigrupa 2

Pantea Marius Nicusor

Cerință:

Să se proiecteze un calculator de buzunar cu operații aritmetice fundamentale (adunare, scădere, înmulțire, împărțire). Operațiile de înmulțire și împărțire se vor implementa folosind algoritmi specifici, nu operatorii limbajului. Operanzii sunt reprezentați pe 8 biți. Operanzii și operatorii vor fi introduși secvențial în formă zecimală. Se vor folosi afișajele cu 7 segmente de pe plăcuțele cu FPG.

1.Specificatii

- Stare inițială când numere A și B nu au fost încărcate

-Numere se pun pe swich-uri și se apasă pe un buton pentru a le încărca

-Se alege modul

-Se apasă pe alt buton si se executa instrucțiunea

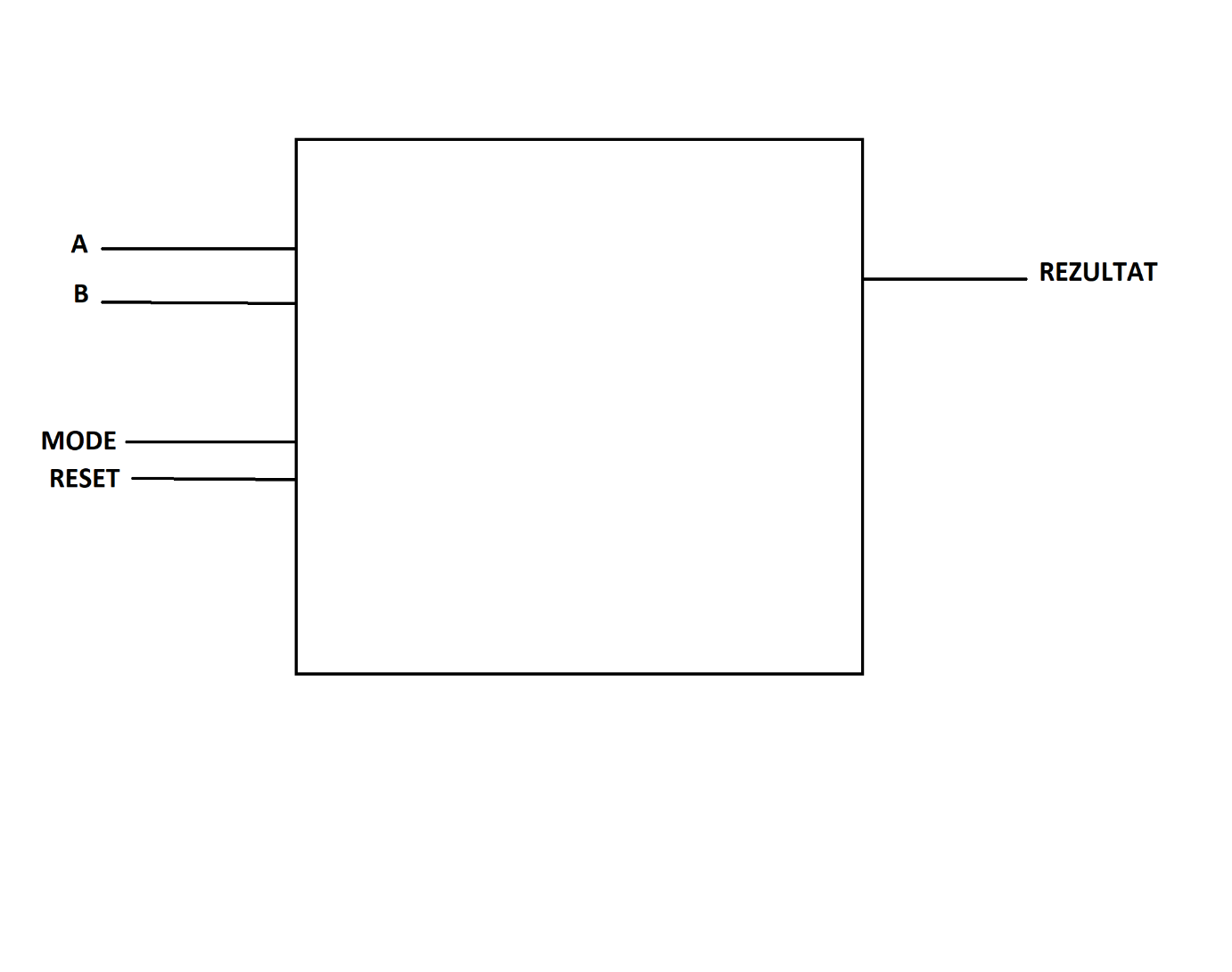
-rezultatul apărând pe afișajele 7 segment

-Va conține si un reset pentru a anula rezultatul de pe 7 segment (pentru a se încarcă alte valori si să se facă altă operație)

-Se va utiliza si un clock pentru încărcarea numerelor și pentru împărțire

**2.Proiectare**

2.1 Schema bloc



Acest prim pas poate fi complicat, există intrări si ieșiri care pot sa nu fie specificate in text dar pe care proiectantul sa le adauge pentru a putea implementa funcționalitatea pe placa.

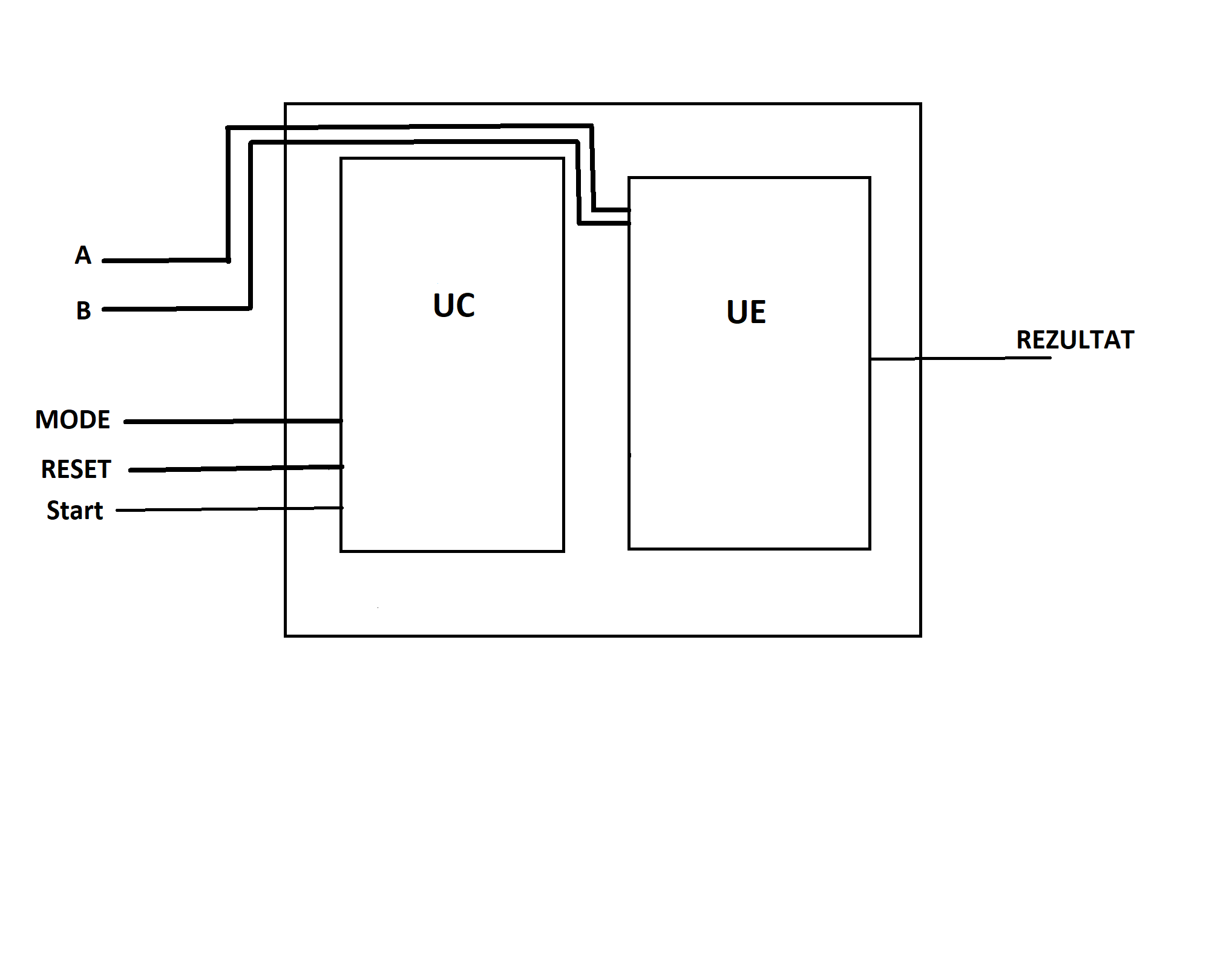
Proiectantul trebuie la acest pas sa identifice use case-urile proiectului , sa ia fiecare caz in parte pentru a tine cont de restricțiile plăcii. Oricând pot fi adăugate semnale de intrare si ieșire pe parcurs

2.2 Unitate de control si Unitate de execuție

Cutia neagră a sistemului trebuie descompusa mai departe pentru a putea găsi componente implementabile. Vom face o descompunere top-down a problemei pana când ajungem la circuite cunoscute ,iar apoi vom implementa bottom-up.

1.2.1 Maparea intrărilor și ieșirilor cutiei mari pe cele doua componente UC și UE

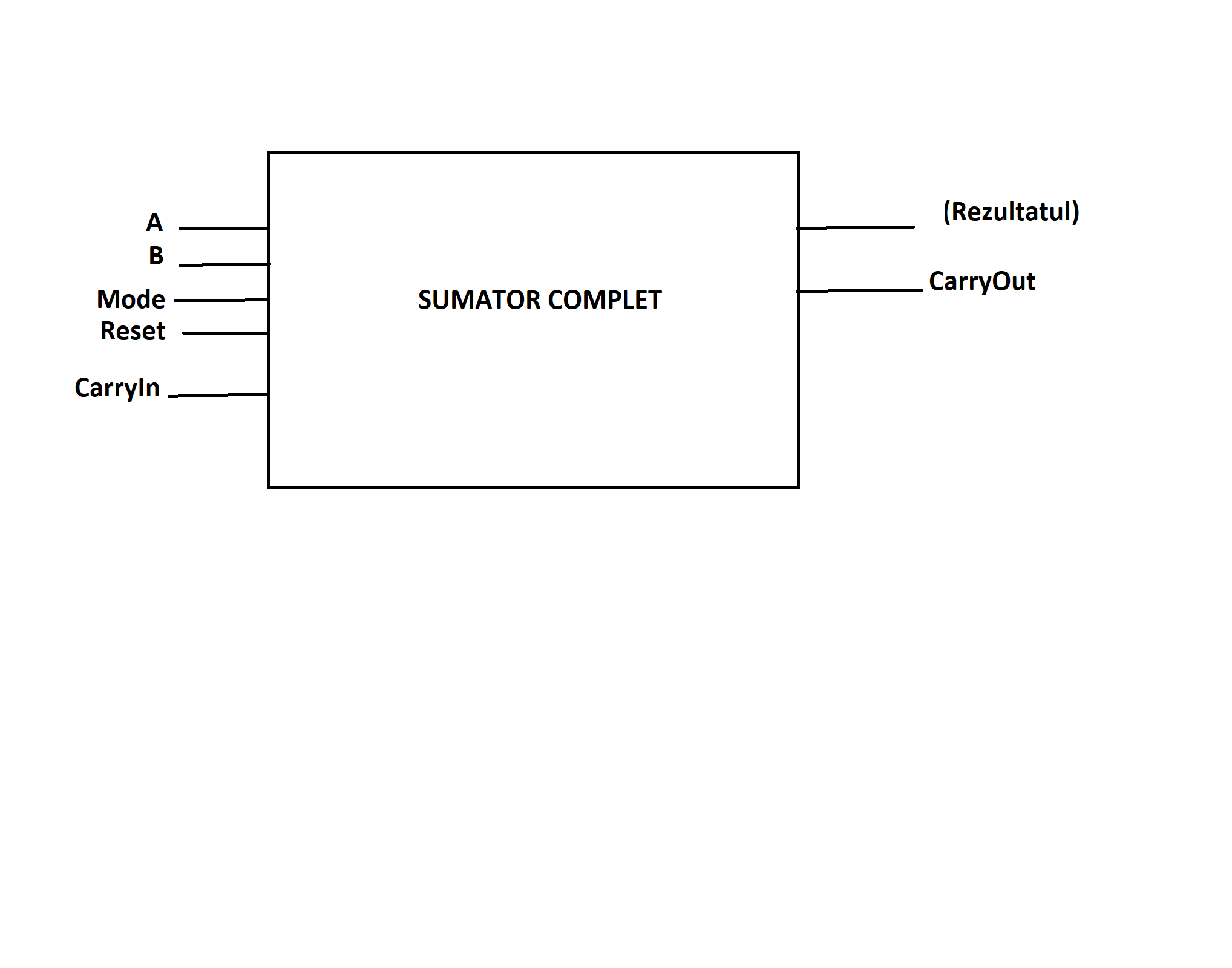
Putem împărți atât intrările cât și ieșirile în 2 categorii : de date si de control. Această separare este esențială la început.



|  |  |  |
| --- | --- | --- |
| Nume | Scop | Dimensiune |
| A,B | Operanzi | 8-biti |
| Mode | Modul  Pentru 000-invalid  Pentru 001-adunare  Pentru 010-scadere  Pentru 011-inmultire  Pentru 100-impartire  Pentru 101-unvalid  Pentru 110-incarcare A si B | 3-biti |
| Reset | Resetare | 1-bit |
| Rezultat | Se stochează rezultatul | 16-biti/8 biți(adunare, împărțire scădere) |
|  |  |  |

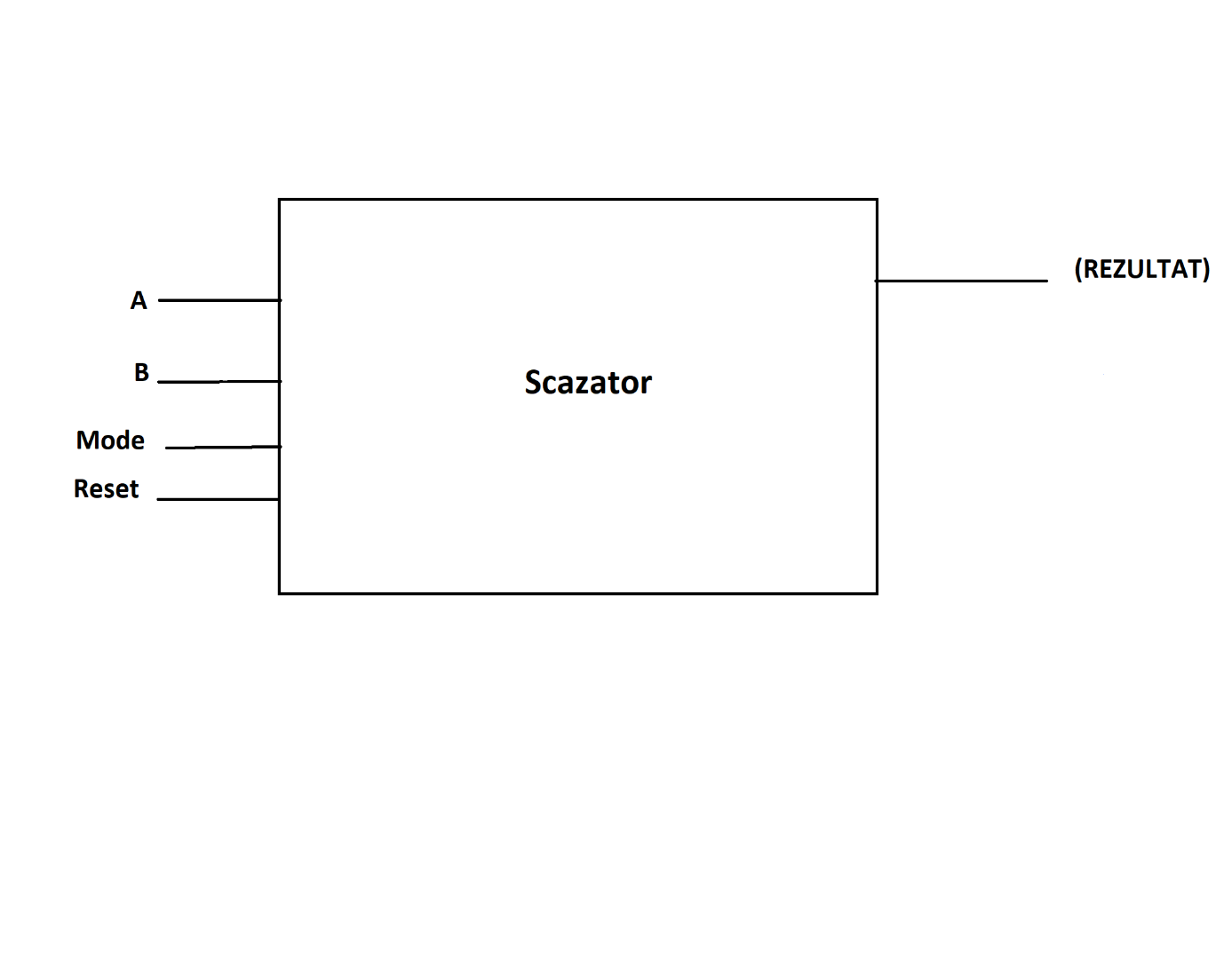
1.2.2 Determinarea resurselor (UE)

1.Sumator pentru realizarea adunări



2.Pentru realizarea scăderi in complement fata de 2,avem nevoie de un sumator care are o intrare pentru numărul A si cea de la B negata (deoarece avem nevoie de A doilea număr negata ),apoi in alt sumator se aduna rezultatul din primul si cu 1 (in baza 2) .

(Cin si Cout-ul se vor lega corespunzător)

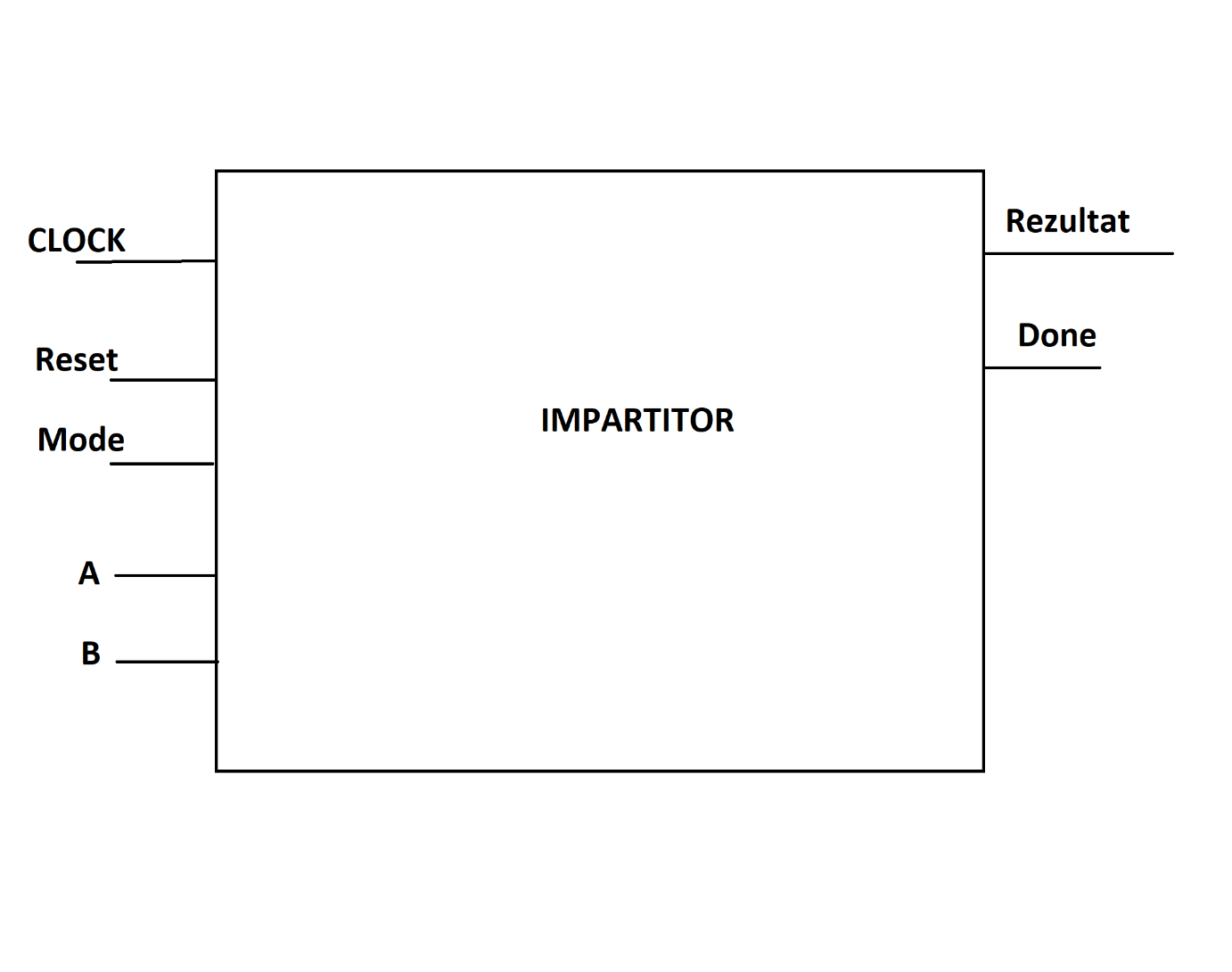


3.Pentru înmulțirea a doua numere, se va realiza un circuit ,care utilizează siftarea la stânga.

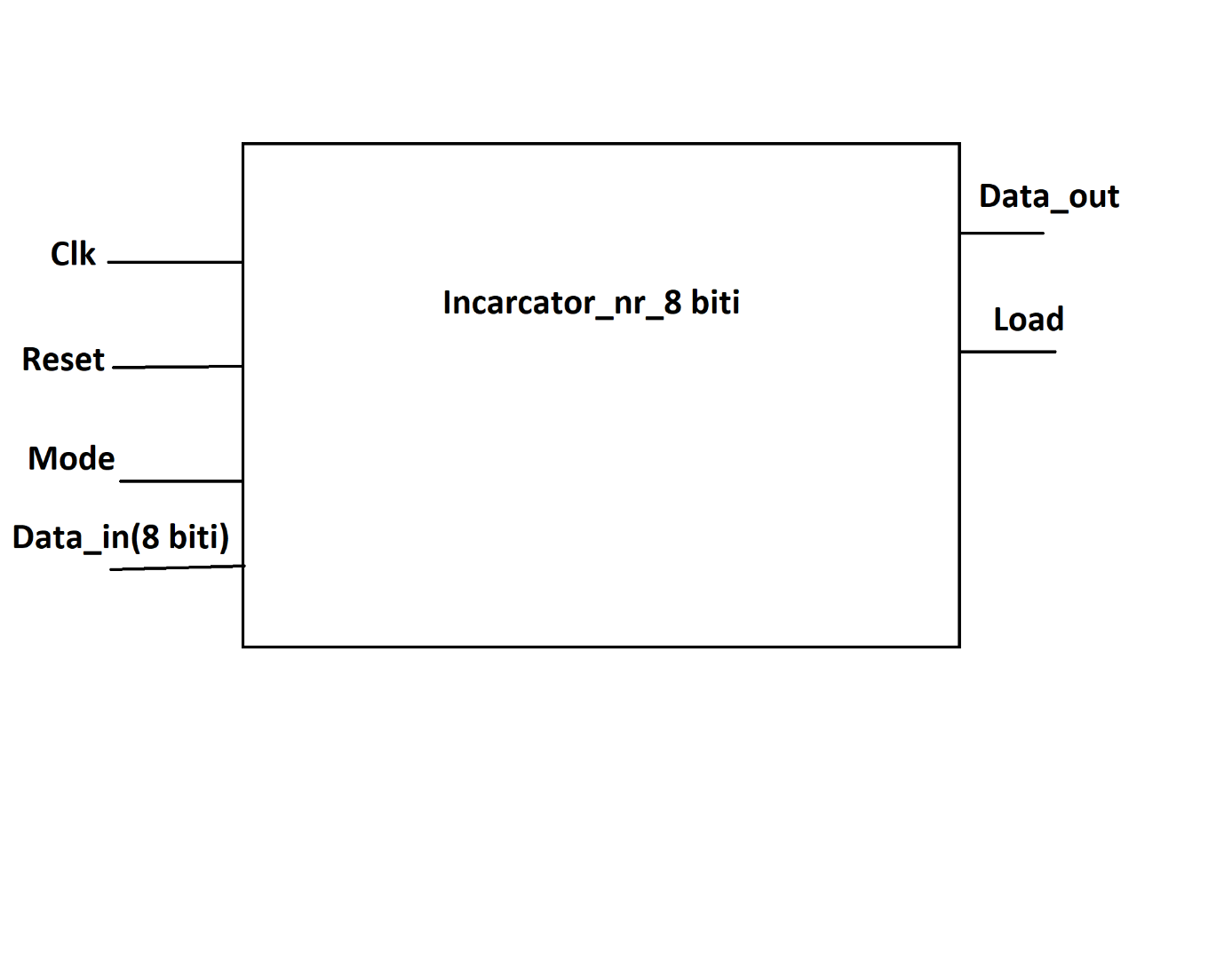
O imagine care conține text, captură de ecran, diagramă, linie

Descriere generată automat

4.Pentru împărțire



5.Un circuit care încarcă un număr pe 8 biți



6.Un circuit convertor BCD-7segment (afișări de la 0(0000) la F(1111))

2.2.3 Schema bloc a primei descompuneri

O imagine care conține diagramă, Desen tehnic, schiță, Plan

Descriere generată automat

2.2.4 Reprezentarea UC prin diagrama de stări (organigrama)

Diagrama de stări nu este un flow-chart ,ci reprezintă partea de control partea decizionala din orice algoritm si ea poate fi apoi implementata direct in VHDL daca e făcută corect.

-Stările sunt reprezentate prin .O stare reprezentativa un moment de timp ( o perioada )

Start

-Deciziile luate in fiecare stare sunt reprezentate prin romb.

-Ieșirile generate in fiecare stare sun reprezentate prin. In interiorul dreptunghiului se enumera ieșirile care sunt adevărate in acel moment.

En,Reset

O imagine care conține captură de ecran, alb și negru, cerc, proiectare

Descriere generată automat

3.Manual de utilizare

-Se aleg cele doua numere in binar si se pun pe intrările A si B

-Se alege modul de încărcare (Mode =110) , se apasă pe start (sa fie 1) si apoi pe clock. Numere au fost încărcate apărând pe ieșirile load a circuitelor de încărcare 1.

-Apoi se alege din cele 4 operații

1.Adunare (Mode=001) ,se apasă start si rezultatul se poate vede in binar pe ieșirea de la circuit sau la afișoarele 7-segment de la circuitul respectiv.

1.Scadere (Mode=010) ,se apasă start si rezultatul se poate vede in binar pe ieșirea de la circuit sau la afișoarele 7-segment de la circuitul respectiv.

1.Inmultire (Mode=011) ,se apasă start si rezultatul se poate vede in binar pe ieșirea de la circuit sau la afișoarele 7-segment de la circuitul respectiv.

1.Impartire (Mode=100) ,se apasă start apoi de doua ori pe clock(pe front ascendent) si rezultatul se poate vede in binar pe ieșirea de la circuit sau la afișoarele 7-segment de la circuitul respectiv. Se vede si pe ieșirea load a circuitului respectiv daca operația a fost efectuata cu succes.

Se apasă pe reset ,si se resetează tot ,dar la împărțire reset trebuie sa fi pe 1 si se apasă pe clock.

4.Justificarea soluției alese

Am ales aceasta soluție de implementare , fiind un mod destul de accesibil de realizare a acestui proiect , la sfatul profesorului de la laborator . O soluție prin utilizarea unui “ALU” mai complex cu mai multe component. Fiecare componenta cu atributul ei.

5.Posibilitati de dezvoltări ulterioare

- Implementare si împărțirii cu rest

-Dezvoltarea pentru un numere pe mai mulți biți

-Afișarea pe 7-segment sa fie in zecimal nu in baza 16

-Abordări mai complexe ,ale circuitelor componente, eficientizarea timpilor

