LIFT parter + 12 etaje

PROIECT DSD

|  |  |  |
| --- | --- | --- |
|  | Studenți: Onaci Andreea-Maria și Potra Paul-Antonio |  |
|  |  |  |
|  | Coordonator: Blaj Ileana |  |

2022

1. Specificația proiectului

Să se proiecteze un automat care comandă un lift într-un hotel cu P + 12 etaje. Liftul trebuie să răspundă solicitărilor persoanelor aflate în interior și cererilor exterioare (sus, jos) care apar pe parcurs de la ușile aflate la fiecare nivel. Ordinea de onorare a cererilor ține cont de sensul de mers (urcare sau coborâre). Se onorează cererile în ordinea etajelor, indiferent de unde provin ele (lift sau exterior). Liftul are o intrare care sesizează depășirea greutății maxime admise și nu pornește în acest caz. Plecarea nu are loc dacă ușile nu sunt închise. Ușile trebuie să stea deschise un interval de timp programabil. Ușile nu se închid dacă există vreo persoană în ușă. Viteza liftului va fi selectabilă între două valori: 1 sau 3 secunde / etaj. Se consideră că în momentul inițial liftul se găsește la parter, cu ușile deschise.

Sistemul proiectat este un automat de tip Mealy.

1. Schema bloc cu componentele principale

date\_out

clk

cathode

reset

ușa

ok

sensor

floor

dir

buton

reset – intrare care resetează liftul la setările inițiale

ușa – intrare care se activează când ușile sunt închise

ok – intrare care se activează când se poate memora etajul la care dorim să ajungă liftul

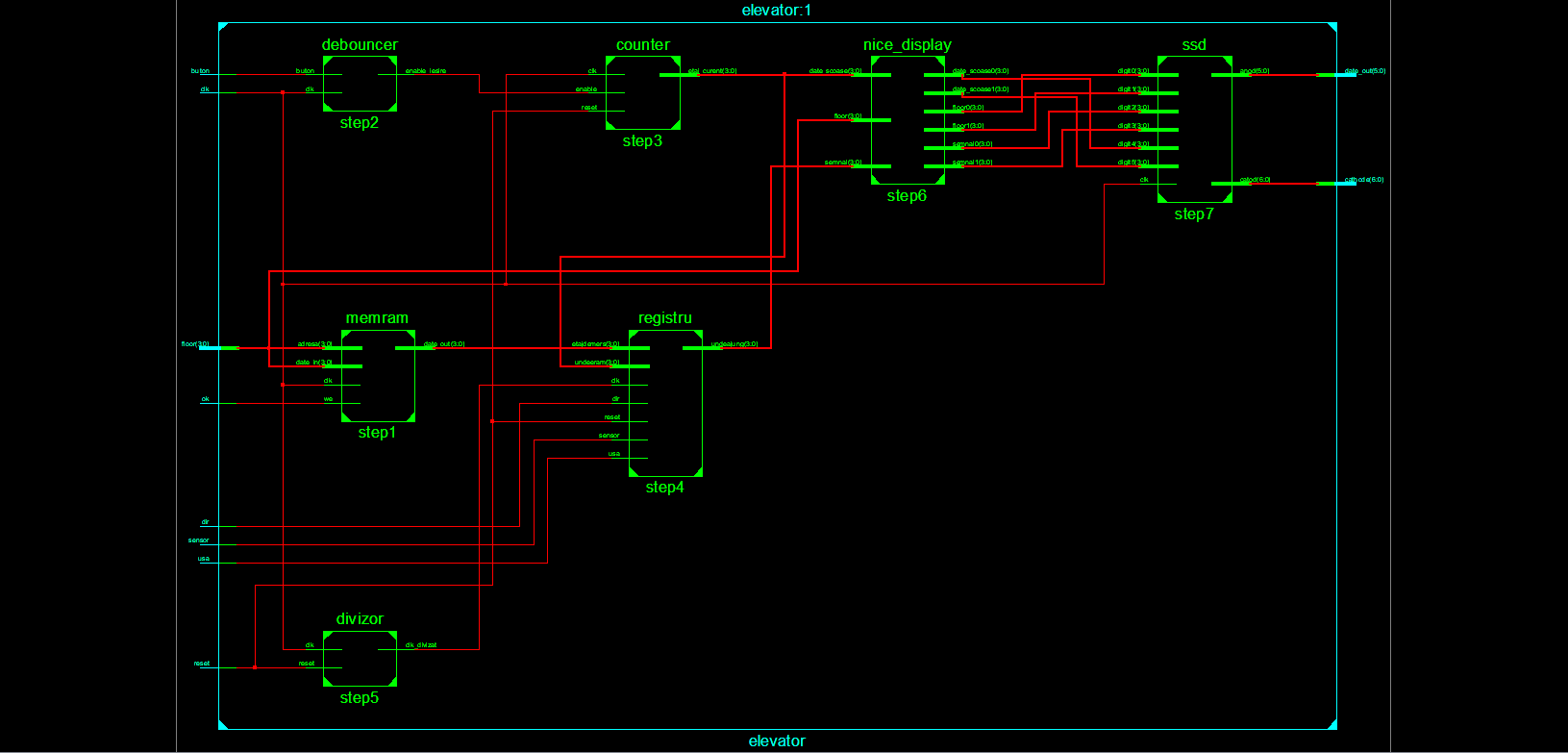
sensor – intrare activă dacă nu există persoane în ușă și greutatea este bună

floor – intrare pe 4 biți care reprezintă valoarea etajului unde dorim să ajungă liftul

dir – intrare care determină direcția în care dorim să meargă liftul (pentru urcare valoarea e 1 și pentru coborâre e 0)

buton – intrare pentru a activa counter-ul liftului

**Schema logică**



iesire\_db – iesire pentru debouncer și intrare ca enable pentru counter-ul liftului

clk\_div – clock-ul divizat necesar pentru a vedea pe plăcuță cum se modifică etajele

date\_scoase – reprezintă etajul la care se află liftul, parcursul dintre etajul la care se afla inițial și etajul final

datee – semnal care reprezintă etajul de la care pleacă liftul

floor0 – semnal care reprezintă cifra unităților etajului la care pleacă liftul

floor1 – semnal care reprezintă cifra zecilor etajului la care pleacă liftul

date\_scoase0 - semnal care reprezintă cifra unităților etajului de la care pleacă liftul

date\_scoase1 - semnal care reprezintă cifra zecilor etajului de la care pleacă liftul

semnal0 – semnal care reprezintă cifra unităților etajului curent

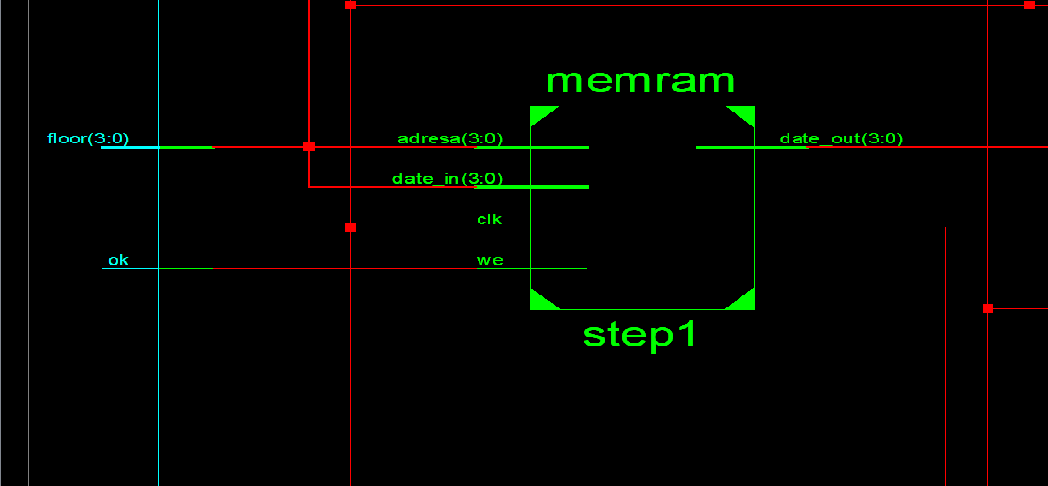
semnal1 – semnal care reprezintă cifra zecilor etajului curent

1. Proiectare și implementare

La început, liftul se află în stare de staționare. În momentul în care nu există persoane în ușă, greutatea este bună, ușile sunt închise, iar liftul primește o comandă, liftul urcă/coboară în funcție de direcție. După ce liftul ajunge la etajul comandat, acesta intră din nou în stare de staționare.

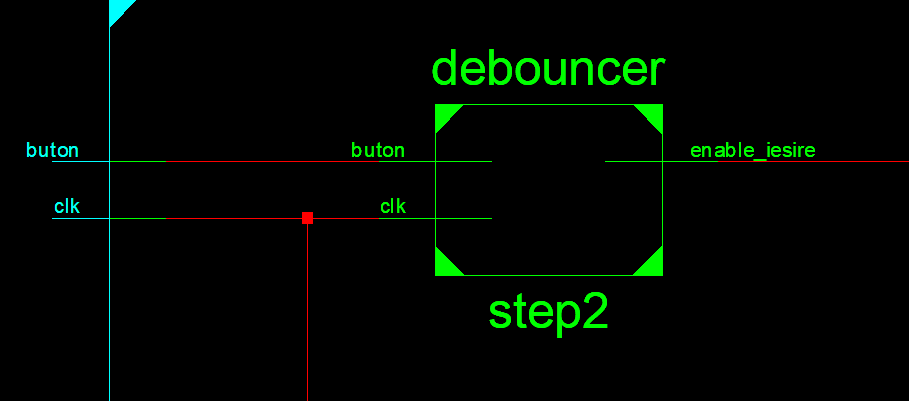
Memoria RAM

Etajul dorit este introdus în RAM. Dacă cererile de comandă sunt bune și se îndeplinesc condițiile de introducere a etajului, memoria RAM memorează în „datee” valoarea etajului la care trebuie să ajungă liftul.



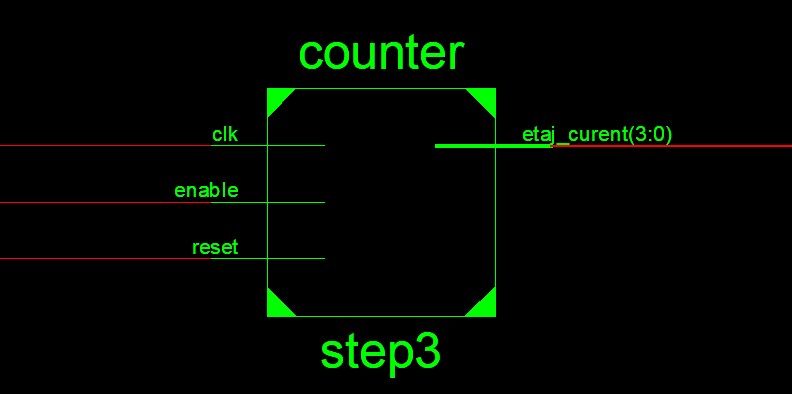
Debouncer

Debouncer-ul asigură valoarea etajului inițial. Precum pentru floor se dă din plăcuță valoarea etajului inițial (switch-uri), pentru etajul inițial se stabilește valoarea prin apăsarea unui buton și verificarea valorii dorite pe afișor (anozii 5 și 6).



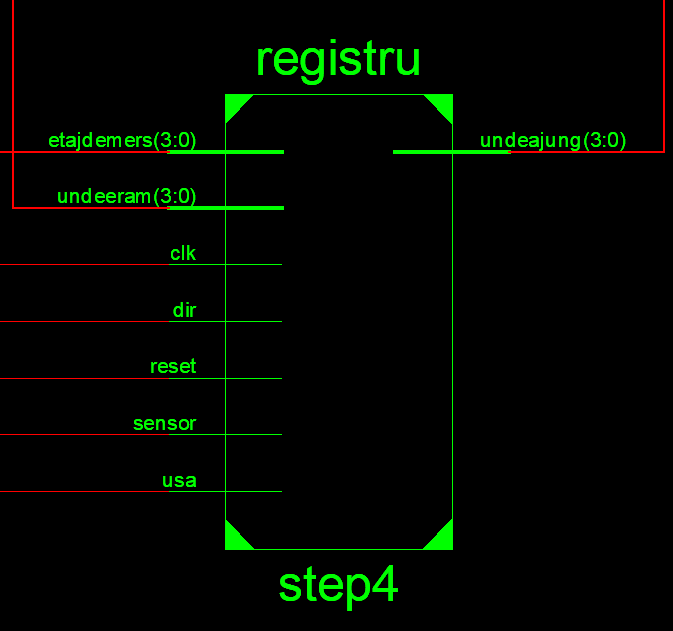
Counter

Numărătorul determină etajul unde eram, practic etajul inițial. Dacă reset-ul este 1, atunci liftul se resetează și se află la parter. Dacă reset-ul este 0, liftul urcă, căutând etajul unde avem o comandă.



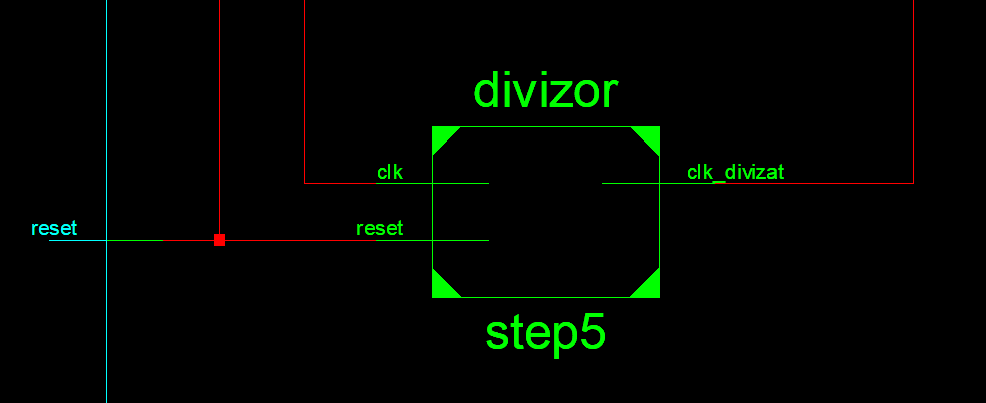
Registru

Registrul asigură de fapt întreaga funcționare a liftului. Dacă reset-ul este 1, etajul unde trebuie să ajungem este parterul. Dacă reset-ul este 0 și condițiile pentru ca liftul să efectueze comanda nu sunt îndeplinite, atunci liftul rămâne la etajul unde se află în stare de staționare până când comenzile sunt îndeplinite. Dacă comenzile sunt îndeplinite, verificăm cu ajutorul input-ului „dir” direcția unde dorim să mergem. Dacă dir este 1 și etajul unde ne aflăm este mai mic decât etajul unde vrem să ajungem, atunci liftul urcă cu câte un etaj. Dacă dir este 0 și etajul unde ne aflăm este mai mare decât etajul unde vrem să ajungem, atunci liftul coboară cu câte un etaj.



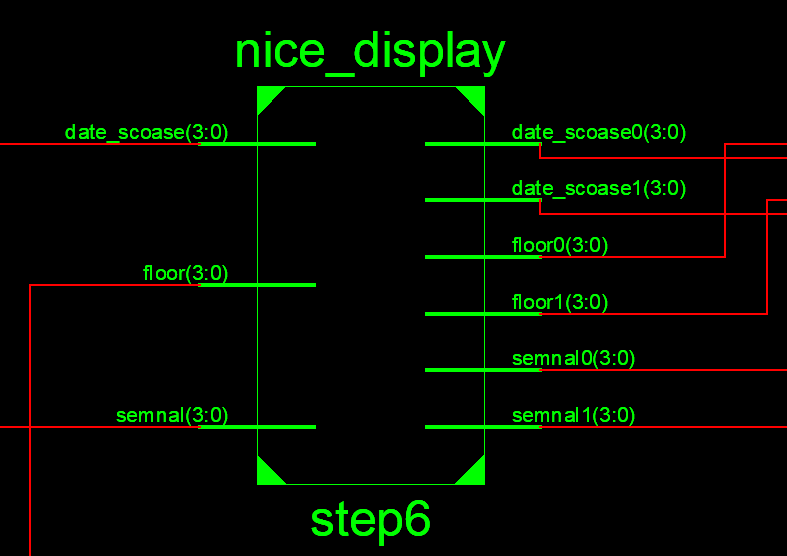
Divizor de frecvență

Divizorul plăcii este de 50 MHz, de aceea avem nevoie de un clock divizat pentru a putea încetini timpul de execuție al operațiunilor și pentru a putea vedea cu ochiul liber ce se întâmplă pe afișoare.



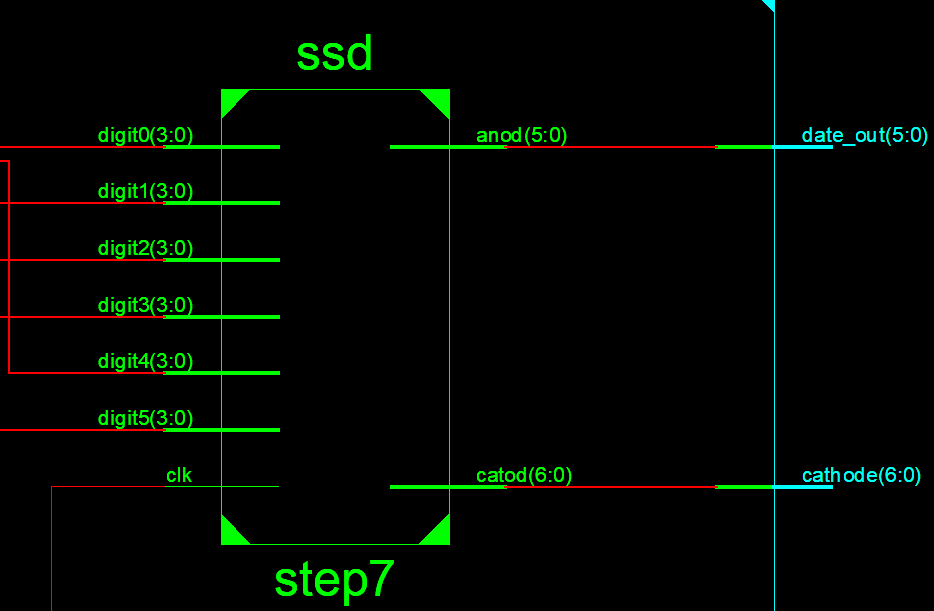
Nice\_display

Această componentă asigură afișarea în zecimal a etajelor. Date\_scoase este alcătuit din două cifre: date\_scoase0 care reprezintă cifra unităților pentru etajul curent, date\_scoase1 reprezintă cifra zecilor pentru etajul curent (0 pentru etaj format dintr-o singură cifră), semnal este format din semnal1 care reprezintă cifra unităților pentru etajul inițial, semnal0 reprezintă cifra zecilor pentru etajul inițial (0 pentru etaj format dintr-o singură cifră), floor este alcătuit din floor0 care reprezintă cifra unităților pentru etajul unde dorim să ajungem și floor1 care reprezintă cifra zecilor pentru etajul unde dorim să ajungem.



SSD

Componenta SSD afișează pe primii 6 anozi valorile cerute. Pe primul anod se afișează cifra unităților a etajului unde dorim să ajungem, pe al doilea cifra zecilor a etajului final, pe al treilea cifra unităților a etajului curent, pe al patrulea anod cifra zecilor a etajului unde ne aflăm, pe anodul al cincilea cifra unităților a etajului inițial și pe al șaselea anod cifra zecilor a etajului la care ne-am aflat inițial.



1. Utilizare și rezultate

(poză cu plăcuța)

*case dir is*

*when '0' => if iesire > etajdemers then*

*iesire <= iesire - 1;*

*end if;*

*when '1' => if iesire < etajdemers then*

*iesire <= iesire + 1;*

*end if;*

*when others => null;*

*end case;*

În acest case, pentru direcția 0, registrul verifică dacă semnalul ieșire care reprezintă etajul curent este mai mare decât etajul unde dorim să ajungem, caz în care liftul urmează să coboare. În cazul în care direcția este 1, liftul verifică dacă etajul curent este mai mic decât etajul unde este plasată comanda și în acest caz, liftul urcă.

*case semnal is*

*when "1100" => semnal0 <= "0010"; semnal1 <= "0001"; -- 12*

*when "1011" => semnal0 <= "0001"; semnal1 <= "0001"; -- 11*

*when "1010" => semnal0 <= "0000"; semnal1 <= "0001"; -- 10*

*when others => semnal0 <= semnal; semnal1 <= "0000"; -- < 10*

*end case;*

Acest case asigură afișarea etajului curent în decimal, aparținând componentei nice\_display. Pentru 12, semnal0 (care reprezintă cifra unităților) este „0010” în binar, cifră care în decimal este 2 și semnal1 (care reprezintă cifra zecilor) este „0001” în binar, cifră care în decimal este 1. Analog pentru valorile 11 și 10. Pentru valori de o cifră, cifra zecilor este 0, iar cifra unităților este etajul curent dat de registru.

1. Posibilități de dezvoltare

Liftul poate fi utilizat într-o clădire care are 12 etaje, ajutând la deplasarea persoanelor de la un etaj la altul. Avantajul liftului proiectat de noi este faptul că reține etajul unde s-a aflat anterior și poate efectua comenzi în lanț, fără a fi nevoie să îl resetăm de fiecare dată când dorim să dăm o nouă comandă (cu excepția primelor două comenzi, dacă prima comandă este de urcare și a doua de coborâre, liftul se va reseta automat, neținând cont de etajul unde am dori să ne oprim la coborâre, acesta va coborî direct la parter, acest lucru putând fi îmbunătățit într-o dezvoltare ulterioară).

Liftul poate fi de asemenea îmbunătățit. În momentul în care liftul primește o altă comandă în drum (de exemplu liftul se afla inițial la etajul 3 și dorește să urce la 11, dar în drum primește comandă la 8, el să se oprească la 8 pentru a prelua persoana de la etajul respectiv dacă aceasta alege sensul ca fiind de urcare). O altă îmbunătățire este introducerea unui sistem vocal și butoane cu sistem tactil de tip braille. O altă idee de îmbunătățire ar fi introducerea unui buton care să anuleze comanda înainte ca ușile să apuce să se închidă în cazul în care introducem un etaj greșit.

1. Justificarea soluției alese

Am utilizat 7 componente.

Memoria RAM este necesară pentru a putea memora etajul la care dorim să ajungem.

Debouncer-ul este folosit pentru a putea da cu ajutorul unui buton etajul de unde dorim să pornim la început, pentru testarea liftului, fiind folosit doar pentru prima comandă, deoarece pentru următoarele, etajul inițial devine etajul unde am ajuns la pasul anterior.

Counter-ul l-am utilizat pentru a putea număra etajul inițial și a-l putea afișa pe SSD.

Registrul asigură funcționalitatea liftului. Am folosit un registru în regim de numărare pentru a ține minte etajul curent și a-l putea afișa pe SSD.

Divizorul asigură clock-ul pentru a putea vedea etajul curent pe afișor.

Nice\_display asigură afișarea în zecimal a valorilor.

SSD este afișorul cu 7 segmente care utilizează 6 anozi din cei 8 disponibili.

Am ales să implementăm această metodă cu doar 7 componente pentru a fi un lift eficient care îndeplinește toate comenzile, iar în cazul unei implementări ulterioare, acesta ar fi ieftin datorită numărului mic de componente necesare.