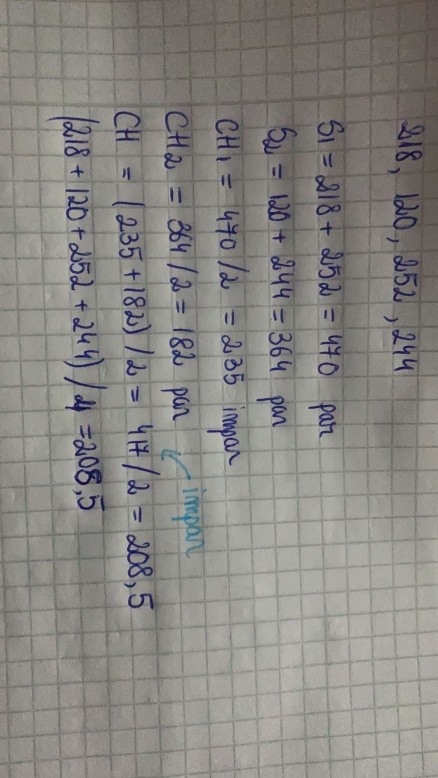
Ionica Madalina-Cristina 332AB, Tema 1

SENSORS\_INPUT

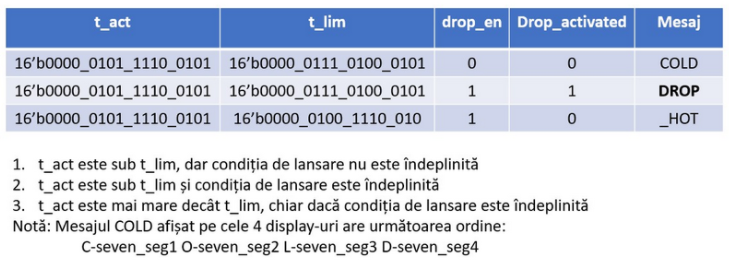
Pentru modulul sensors\_input am ales toate cazuile posibile in functie de valoarea senzorilor. Astfel am ales pe rand cazurile in care fiecare dintre senzori avea valoarea 0. In cazurile in care senzorii 1 sau 3 aveau valoarea 0, calculam suma senzorilor 2 si 4, la fel si pentru 2 sau 4, calculam suma celor de la 1 si 3, asa cum este precizat in anexa. Dupa calcularea sumei, am evaluat LSB-ul, adica primul bit de la dreapta la stanga. Daca acesta era 0, atunci suma era doar impartita la 2 pentru ca suma era un numar par, deci impartirea avea sa rezulte exact (impartire prin shiftare la dreapta), altfel adaugam un 1 pentru a efectua rotunjire, suma fiind impara. Conditiile le-am evaluat folosind operatorul de conditionare.

Ultimul caz a fost cel al celor 4 senzori, toti diferiti de 0, unde am calculat doua sume partiale, pe care ulterior le-am aproximat dupa acelasi principiu. Dupa aproximare ca sa obtin media celor patru numere, am facut media sumelor partiale. Totusi am reusit sa trec doar 970/1000 de cazuri pentru ca modul meu de rezolvare exclude cazul unor numere precum 218, 120, 252, 244, unde ultima medie nu mai este aproximata, iar diferenta dintre rezultate este +1. Nu am stiut cum ar trebui sa rezolv acest caz fara a pierde teste ok pentru celelalte cazuri.



DISPLAY\_AND\_DROP

La modul acesta am folosit operatorul conditional, din nou pentru a stabili cazurile mentionate in anexa. Pe baza cazurilor am reusit sa implementez



Apoi am scris mesajele pe cele 4 segmente asemnea exercitiului 2 din laboratorul 3.

SQUARE\_ROOT

Modulul square\_root l-am facut pe baza algoritmului codric oferit la resurse pe care l-am transpus in Verilog ([Square-root CORDIC (convict.lu)](https://www.convict.lu/Jeunes/Math/square_root_CORDIC.htm)). Am observant ca baza 128 propusa nu era potrivita pentru numerele ce trebuiau testate in cazul meu. Am mai observant si faptul ca baza era calculate prin 2^baza, iar numarul de iteratii din for era chiar puterea lui 2. Asa ca pentru ca numarul out din anexa era scris in baza 16, am ales aceeasi 2^16 facand primul bit din stanga 1, pentru a calcula aceasta putere. In plus fata de algoritmul propus, am shiftat la stanga cu 16 pentru a ajunge la MSB ca sa obtin rezultatul in virgula fixa, conform testelor. In continuare, am respectat algoritmul propus pentru cordic.

BAGGAGE\_DROP

Pentru Top\_Module am creat legaturile intre cele 3 module rezolvate anterior, asemnea schemei din tema. Legatura am creat-o prin instantiere, asemenea acestui tutorial ([Intel Quartus: Connecting Modules in Verilog - YouTube](https://www.youtube.com/watch?v=VatK4m3m430&t=192s)) de interconectare a modulelor. In plus am declarat doua variabile de tip wire pentru a crea conexiunea intre modulul 1 si 2, respectiv 2 si 3, cu numarul de biti necesari, 8 si 16, pe care i-am dedus in functie de numarul de biti pe care erau definite anterior, dar si in functie de erorile pe care le-am primit cand am uitat sa introduc si dimensiunea pentru wire.