

# Tema 1 - Baggage drop

---

- Responsabili: Ionuț Pascal, Ana-Maria Crețan
- Deadline soft (fără penalizări): **14.11.2021**, ora **23:59**
- Deadline hard (cu penalizări): **21.11.2021**, ora **23:59**
- Data publicării: **03.11.2021**
- Data ultimei actualizări: 03.11.2021, 00:00
- Istoric modificări:
  - 03.11.2021
    - Publicare temă + checker offline
  - 06.11.2021 22:03
    - Adaugare mențiune suplimentară în enunț și în barem relativ la permisiunea folosirii operatorului ' / '

## Obiective

---

Tema are ca scop familiarizarea cu noțiunile limbajului Verilog studiate în cadrul primelor laboratoare: module, construcții de limbaj, blocul always, prin:

- divizarea problemei generale și organizarea ei în module cu o funcționalitate specifică;
- implementarea unui algoritm dat într-o manieră sintetizabilă.

## Descriere și cerințe

---

Implementați în Verilog un **circuit combinațional** care are ca scop simularea aruncării automate a unui container cu provizii într-un câmp de luptă de către un elicopter către o echipă de pușcași marini, în momentul când acesta se află în perimetrul autorizat. Pentru a nu fi neutralizat, acesta trebuie să ajungă la sol într-un timp limită  $t$ , calculat cu ajutorul formulei  $t = \sqrt{\text{height}} / 2$

Circuitul va extrage de fiecare dată înălțimea curentă la care se află elicopterul de la cei 4 senzori prezenți pe burta aeronavei (plasați în perechi pentru a asigura redundanța măsurărilor), va efectua media aritmetică a celor activi și va afișa informația pe un display cu 4 elemente **7Seg**, emițând o alarmă în momentul în care pachetul este desprins.

Valorile de intrare ale tuturor senzorilor sunt reprezentate pe 8 biți. Senzorul este considerat având o valoare validă în momentul în care aceasta este diferită de 0. Timpul limită este exprimat pe 16 biți, în virgulă fixă, cu virgula între bitul 8 și bitul 7. Condiția de validare a aruncării automate este reprezentată de un semnal de intrare pe 1 bit.

Ieșirea modulului este reprezentată de 4 module **7Seg**, care vor afișa următoarele tipuri de mesaje:

- **COLD** - dacă elicopterul nu se află în aria de aruncare ( semnalul de validare nu este activ )
- **\_HOT** - dacă elicopterul se află în aria de aruncare, dar timpul limită este prea mare
- **DROP** - dacă elicopterul a aruncat pachetul.

În momentul în care semnalul DROP este activ, alarma va fi și ea activă.

Un exemplu detaliat este prezentat în [anexă](#).

## Implementare

---

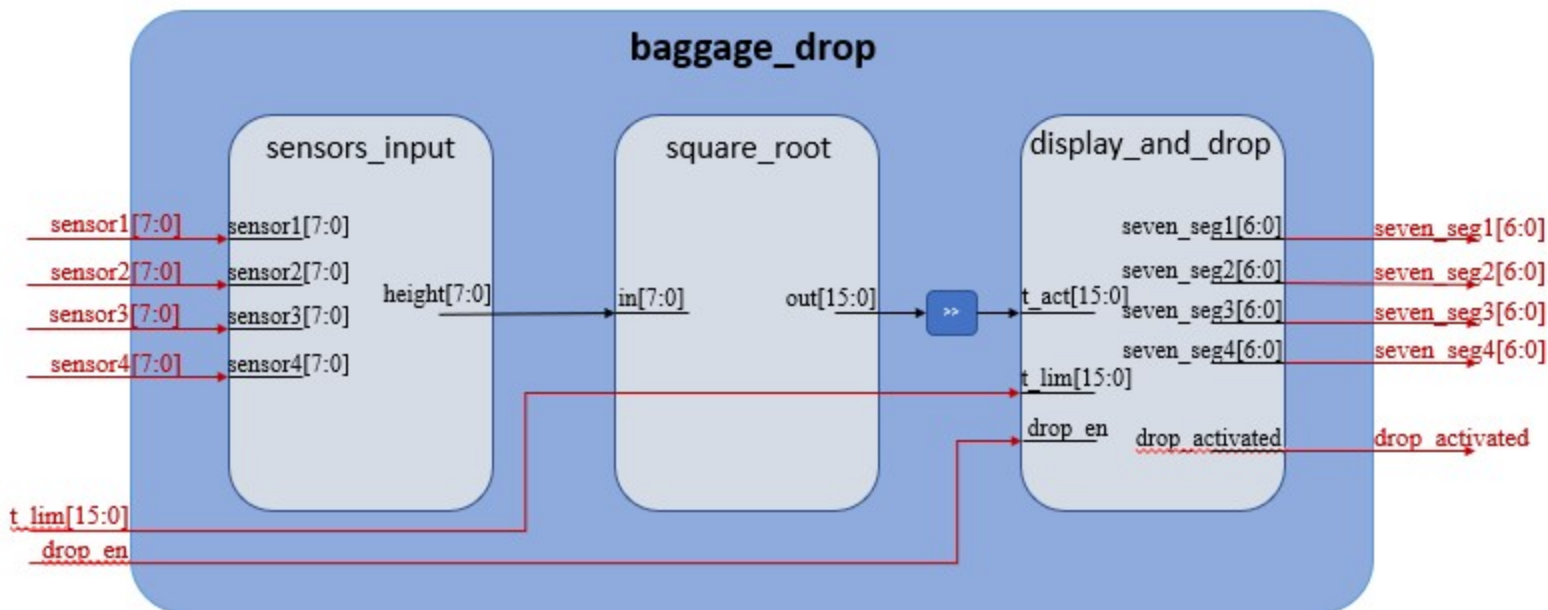
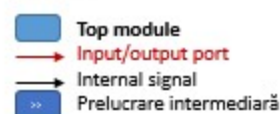


Fig1. Schemă bloc

#### Legend



Pentru rezolvarea temei este necesară împărțirea problemei în 3 module distincte, cu funcționalități specifice, instanțiate și conectate la nivel superior într-un modul de top. Schema bloc și conexiunile modulelor sunt prezentate în Fig1. Schemă bloc. Modulele trebuie să respecte interfețele prezentate mai jos, cu următoarele mențiuni:

- ieșirile pot fi declarate de tip registru;

## baggage\_drop

Modulul are rolul de a instanția blocurile și a realiza conexiunile necesare funcționării corespunzătoare, precum și prelucrare minimală a semnalelor intermediare ce nu necesită un modul separat.

Modulul trebuie să respecte următoarea interfață:

```
module baggage_drop (
    output [6 : 0] seven_seg1,
    output [6 : 0] seven_seg2,
    output [6 : 0] seven_seg3,
    output [6 : 0] seven_seg4,
    output [0 : 0] drop_activated,
    input [7 : 0] sensor1,
    input [7 : 0] sensor2,
    input [7 : 0] sensor3,
    input [7 : 0] sensor4,
    input [15 : 0] t_lim,
    input drop_en);
```

Descrierea semnalelor folosite de acest modul este următoarea:

- **seven\_seg\*** - modulele 7Seg pe care va fi afișat mesajul corespunzător
- **drop\_activated** - semnalul de alarmă ce se activează când se respectă condiția de lansare a pachetului
- **sensor\*** - intrările ce reprezintă cele 4 înălțimi, măsurate individual
- **t\_lim** - timpul limită de coborâre a pachetului
- **drop\_en** - activează posibilitatea lansării pachetului

## sensors\_input

Modulul are rolul de a interoga informația primită de la senzori precum și gradul lor de disponibilitate și a furniza suma aproximată la cel mai apropiat întreg a valorilor modulului square\_root spre calcularea rădăcinii pătrate.

Dacă un senzor este 0, acesta și perechea lui se exclud din calcul. Perechile de senzori sunt 1-3 și 2-4.

Se garantează faptul că nu există senzori din ambele perechi cu valoarea 0 în același timp.

Modulul trebuie să respecte următoarea interfață:

```
module sensors_input (
    output [7 : 0] height,
    input [7 : 0] sensor1,
    input [7 : 0] sensor2,
    input [7 : 0] sensor3,
    input [7 : 0] sensor4);
```

Descrierea semnalelor folosite de acest modul este următoarea:

- **height** - înălțimea, calculată folosind media senzorilor
- **sensor\*** - intrările ce reprezintă cele 4 înălțimi, măsurate individual

## square\_root

Modulul are rolul de a calcula rădăcina pătrată a unui număr natural, reprezentat pe 8 biți [Resurse], cu datele provenite de la modulul sensors\_input și a o furniza modulului drop\_and\_display în format virgulă fixă, pentru determinarea validității ieșirii și afișarea informațiilor necesare.

Modulul trebuie să respecte următoarea interfață:

```
module square_root (
    output [15:0] out,
    input [7:0] in );
```

Descrierea semnalelor folosite de acest modul este următoarea:

- **out** - rădăcina pătrată a intrării, exprimată în format virgulă fixă pe 16 biți, cu virgula fixă între bitul 8 și bitul 7.
- **in** - numărul pentru care se calculează rădăcina pătrată

Operațiile folosind operatori / și % nu sunt permise în rezolvarea temei, cu excepția operației de împărțire ( / ) la puteri ale lui 2.

## display\_and\_drop

Modulul are rolul de a determina existența condiției de aruncare a pachetului și de a afișa mesajul corespunzător.

Modulul trebuie să respecte următoarea interfață:

```
module display_and_drop (
    output [6 : 0] seven_seg1,
    output [6 : 0] seven_seg2,
    output [6 : 0] seven_seg3,
    output [6 : 0] seven_seg4,
    output [0 : 0] drop_activated,
    input [15: 0] t_act,
    input [15: 0] t_lim,
    input drop_en);
```

Descrierea semnalelor folosite de acest modul este următoarea:

- **seven\_seg\*** - modulele 7Seg pe care va fi afișat mesajul corespunzător
- **drop\_activated** - semnalul de alarmă ce se activează când se respectă condiția de lansare a pachetului (drop\_en - activ și t\_act ≤ t\_lim)
- **t\_act** - timpul curent posibil de coborâre a pachetului
- **t\_lim** - timpul limită de coborâre a pachetului
- **drop\_en** - activează posibilitatea lansării pachetului

# Bonus

---

Implementați modulul **square\_root** folosind una dintre metodele: Newton-Raphson, Goldschmidt, CORDIC sau dezvoltarea în serie Taylor.

Pentru acordarea bonusului fișierul README trebuie să conțină o descriere detaliată a implementării uneia dintre metodele prezentate mai sus.

# Notare

---

- +6 pct: implementarea modului **square\_root**; sunt testate toate combinațiile valide de operanzi pe 8 biți, numere exprimate fără semn;
- +1.5 pct: implementarea corectă a modului **sensors\_input**;
- +1.5 pct: implementarea corectă a modului **display\_and\_drop**;
- +1 pct: implementarea corectă a întregului ansamblu (modulul **top**);
- +2 pct **bonus**: implementarea unui algoritm special;
- +1 pct: fiecare bug găsit în implementarea de referință (se acordă primei persoane care-l semnalează);
  
- -12 pct: folosirea construcțiilor nesintetizabile din Verilog (while, repeat, for cu număr variabil de iterații, etc);
- -10 pct: folosirea operatorilor / , % ( excepție împărțirea la puteri ale lui 2. ex: x / 2, x / 8, etc);
- -7 pct: folosirea tipului real din Verilog, folosirea unui Lookup Table ( memorie ROM ) pentru implementarea modului **square\_root**;
- -1 pct: lipsa fișierului README;
- -1 pct: **indentare haotică** (incluzând **spațiere inutilă**);
- -0.5 pct: pentru fiecare zi de întârziere; tema poate fi trimisă cu maxim 7 zile întârziere față de termenul specificat în enunț (față de deadline-ul soft);
- -0.5 pct: folosirea incorectă a atribuirilor continue ( assign ), blocante ( = ) și non-blocante ( <= );
- -0.2 pct: lipsa comentariilor utile;
- -0.1 pct: comentarii inutile
- -0.2 pct: diverse alte probleme constatate în implementare (per problemă)

Punctajul inițial al checker-ului nu reprezintă punctajul final al temei. Acesta va fi acordat de către asistent, în urma analizei individuale a fiecărei implementări.

Dacă tema primește 0 puncte pe platforma vmchecker, se pot acorda maxim 2 pct pe ideea implementării, la latitudinea asistentului. Ideea și motivele pentru care nu funcționează trebuie **documentate** temeinic în README și/sau comentarii. Temele care au **erori** de compilare vor fi notate cu 0 puncte.

# Precizări

---

- Arhiva temei (de tip **zip**) trebuie să cuprindă în rădăcina sa (*fără alte directoare*) **doar**:
  - fișierele sursă (extensia .v);
  - fișierul README.
- Arhiva **nu** trebuie să conțină fișiere de test, fișiere specifice proiectelor etc.
- Fișierului README va conține minim:
  - numele și grupa;
  - prezentarea generală a soluției alese (ex: descrierea de nivel înalt a algoritmului folosit);
  - explicarea porțiunilor complexe ale implementării (poate fi făcută și în comentarii);
  - alte detalii relevante.
- Vmchecker ne permite să revenim la orice soluție încărcată de voi; cereți revenirea la cea mai convenabilă soluție trimisă (punctaj teste automate + depunțare întârziere) printr-un mail titularului de laborator.
- Tema trebuie realizată individual; folosirea de porțiuni de cod de la alți colegi sau de pe Internet (cu excepția site-ului de curs și a resurselor puse la dispoziție în conținutul temei) poate fi considerată **copiere** și va fi penalizată conform regulamentului.

- Aduceți-vă aminte de **recomandările** prezentate în Tema 0.
- Există module de test pentru fiecare bloc pentru a-i verifica, separat, funcționalitatea.
- Scenariul este o ficțiune iar problema fizică este simplificată.

## Resurse

- Tester
- **Wikipedia** - Square root Algorithms [[https://en.wikipedia.org/wiki/Methods\\_of\\_computing\\_square\\_roots](https://en.wikipedia.org/wiki/Methods_of_computing_square_roots)]
- **Wikipedia** - 7 segments display [[https://en.wikipedia.org/wiki/Seven-segment\\_display](https://en.wikipedia.org/wiki/Seven-segment_display)]
- **Algorithm Non-Restoring** - Implementation of Fixed and Floating Point Square Root Using Nonrestoring Algorithm [<http://www.ijcee.org/papers/767-ET030.pdf>]
- **Newton-Raphson, SRT** - Cost/Performance Tradeoff of n-Select Square Root Implementations [<https://yamin.cis.k.hosei.ac.jp/papers/ACAC2000.pdf>]
- **Goldschmidt** - Improving Goldschmidt Division, Square Root, and Square Root Reciprocal [<http://perso.ens-lyon.fr/jean-michel.muller/IEEETCJul2000-2pdf.pdf>]
- **CORDIC** - Square-root based on CORDIC [[https://www.convict.lu/Jeunes/Math/square\\_root\\_CORDIC.htm](https://www.convict.lu/Jeunes/Math/square_root_CORDIC.htm)]
- Verilog 2000 Standard [[https://sutherland-hdl.com/papers/2001-SNUG-presentation\\_Verilog-2000\\_standard.pdf](https://sutherland-hdl.com/papers/2001-SNUG-presentation_Verilog-2000_standard.pdf)]
- Ghidul studentului la AC [<https://ocw.cs.pub.ro/courses/ac-is/studentguide>]
- Utilizarea vmchecker [<https://ocw.cs.pub.ro/courses/ac-is/tutoriale/6-vmchecker-utilizare>]
- Debugging folosind Xilinx ISE [<https://ocw.cs.pub.ro/courses/ac-is/tutoriale/3-ise-debug>]

## Anexă - Exemplu date

Considerăm un sistem format din 4 senzori, având valorile prezentate în tabel. Cele 3 variante prezentate analizează pe rând un caz de calcul al înălțimii, considerând perechile care se anulează datorită unei valori nule.

Sensor1	Sensor2	Sensor3	Sensor4	Height
140	138	139	140	139.25 -> <b>139</b>
140	<b>0</b>	139	<b>140</b>	139.5 -> <b>140</b>
<b>140</b>	138	<b>0</b>	140	139 -> <b>139</b>

Astfel, în cazul exemplului 2, senzorul 2 are valoare nulă, deci acesta și perechea lui vor fi excluși de la calcularea mediei. Aceasta se va calcula doar din perechea (1,3):  $(140+139) / 2 = 139,5$ .

În urma introducerii rezultatelor în modulul square\_root care calculează rădăcina pătrată, obținem:

```
in = 8'd139 -> out = 16'b0000_1011_1100_1010;
    Unde 0000_1011 -> partea întreagă (11)
        1100_1010 -> partea zecimală (0,7890625)
    -----
    Se obține 11,7890625 vs 11,7898261(real)

in = 8'd140 -> out = 16'b0000_1011_1101_0101;
```

Înainte de a introduce rezultatul în modulul de afișare, trebuie calculat timpul final prin împărțirea la 2, folosind formula prezentată în temă. Vom analiza exemplul 1, al cărei înălțime aproximată este 139:

```
out = 16'b0000_1011_1100_1010 -> t_act = 16'b0000_0101_1110_0101 (5,89453125)
```

Considerând **t\_act** calculat anterior, există mai multe situații pentru ultimul modul, prezentate în următorul tabel:

t_act	t_lim	drop_en	Drop_activated	Mesaj
16'b0000_0101_1110_0101	16'b0000_0111_0100_0101	0	0	COLD
16'b0000_0101_1110_0101	16'b0000_0111_0100_0101	1	1	<b>DROP</b>
16'b0000_0101_1110_0101	16'b0000_0100_1110_010	1	0	_HOT

1. t\_act este sub t\_lim, dar condiția de lansare nu este îndeplinită
2. t\_act este sub t\_lim și condiția de lansare este îndeplinită
3. t\_act este mai mare decât t\_lim, chiar dacă condiția de lansare este îndeplinită

Notă: Mesajul COLD afișat pe cele 4 display-uri are următoarea ordine:

C-seven\_seg1 O-seven\_seg2 L-seven\_seg3 D-seven\_seg4

Mesajele în 7Seg sunt următoarele, unde un led aprins este reprezentat de semnalul 1:

