Tema 1: Monitorizare temperatură

Responsabili: Ionuţ Pascal, Ana-Maria Creţan
Deadline soft (fără penalizări): 15.11.2020, 23:59
Deadline hard(cu penalizări) : 22.11.2020, 23:59
Data publicării: 05.11.2020
Data ultimei actualizări: 05.11.2020
Istoric modificări:

- 05.11.2020
    - Publicare temă

Obiective

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

- divizarea problemei generale şi organizarea ei în module cu o funcţionalitate specifică;
- implementarea unui algoritm dat.

Descriere şi cerinţe

Implementaţi în Verilog un **circuit combinaţional** care are ca scop simularea monitorizării în cadrul unei sere şi afişarea ei într-o formă vizuală (ieşire scalată cu afişaj de tip LED). Circuitul va extrage de fiecare dată temperaturile senzorilor activi, va efectua media lor aritmetică şi va afişa temperatura aproximată într-o manieră codată, pretabilă ca semnal de activare pentru fiecare bec de tip LED, emiţând o alarmă în cazul în care temperatura este în afara valorilor admise.

Valorile de intrare ale tuturor senzorilor sunt reprezentate pe 8 biţi şi sunt concatenate într-un vector de biţi, astfel: [7:0] - valoarea primului senzor, [15:8] - valoarea celui de-al doilea senzor, *etc*. Fiecare senzor are un bit de `enable` care marchează faptul că valoarea trimisă de senzor este validă şi poate fi citită.

Ieşirea modulului va fi aproximată la întregul cel mai apropiat şi poate fi reprezentată doar în intervalul [19:26], orice valoare în afara acestui interval activând un semnal de alarmă. Aceasta este codificată într-un mod scară, astfel:

Temperatură aproximată	Ieşire codată	Alarmă
19	8'b0000_0001	0
20	8'b0000_0011	0
21	8'b0000_0111	0
22	8'b0000_1111	0
23	8'b0001_1111	0
24	8'b0011_1111	0
25	8'b0111_1111	0
26	8'b1111_1111	0

Temperatură aproximată	Ieşire codată	Alarmă
<19	8'b0000_0001	1
>26	8'b1111_1111	1

Notă:

- 18.8 va fi aproximat la 19, deci nu va exista semnal de alarmă
- 18.4 va fi aproximat la 18, deci va exista semnal de alarmă

Un exemplu detaliat este prezentat în anexă.

Implementare

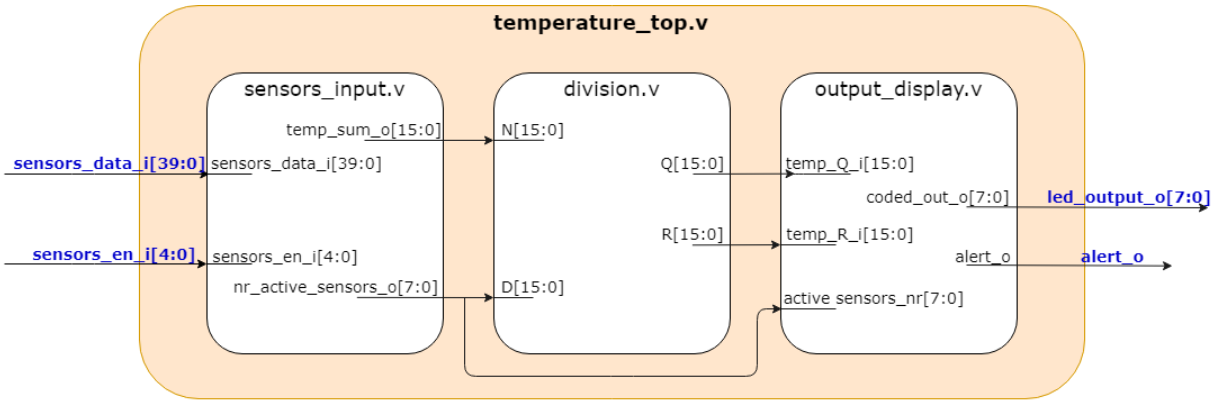

Fig1. Schema bloc

Pentru rezolvarea temei este necesară împărţirea problemei în 3 module distincte, cu funcţionalităţi specifice, instanţiate şi conectate la nivel superior într-un modul de top. Schema bloc şi conexiunile modulelor sunt prezentate în *Fig1. Schema bloc*. Modulele trebuie să respecte interfeţele prezentate mai jos, cu următoarele menţiuni:

- ieşirile pot fi declarate de tip registru;
- modulele pot fi parametrizate cu maxim 1 parametru, dacă se consideră necesar.

temperature_top

Modulul are rolul de a instanţia blocurile şi a realiza conexiunile necesare funcţionării corespunzătoare. Modulul trebuie să respecte următoarea interfaţă:

```
module temperature_top (
    output [7:0] led_output_o,
    output       alert_o,
    input [39:0] sensors_data_i,
    input  [4:0] sensors_en_i);
```

Descrierea semnalelor folosite de acest modul este următoarea:

- `led_output_o` - ieşirea codificată pentru aprinderea individuală a fiecărui led
- `alert_o` - semnalul de alertă, activ dacă ieşirea aproximată este în exteriorul [19:26]
- `sensors_data_i` - valorile de temperatură ale senzorilor, concatenate într-un vector
- `sensors_en_i` - bit de enable al senzorilor; dacă `sensors_en_i`[i] este 1, atunci valoarea indicată este semnificativă

sensors_input

Modulul are rolul de a interoga informaţia primită de la senzori precum şi gradul lor de disponibilitate şi a furniza suma valorilor şi numărul senzorilor activi modulului `division` spre realizarea mediei aritmetice. Modulul trebuie să respecte următoarea interfaţă:

```
module sensors_input (
    output [15:0] temp_sum_o,
    output  [7:0] nr_active_sensors_o,
    input  [39:0] sensors_data_i,
    input   [4:0] sensors_en_i);
```

Descrierea semnalelor folosite de acest modul este următoarea:

- `temp_sum_o` - suma temperaturilor valide
- `nr_active_sensors_o` - numărul senzorilor activi
- `sensors_data_i` - valorile de temperatură ale senzorilor, concatenate într-un vector
- `sensors_en_i` - bit de enable al senzorilor;

division

Modulul are rolul de a efectua operaţia de împărţire cu cât şi rest a două numere naturale reprezentate pe 16 biţi [Resurse], cu datele provenite de la modulul `sensors_input` şi a le furniza modulului `output_display` pentru aproximare şi codificare. Modulul trebuie să respecte următoarea interfaţă:

```
module division (
    output [15:0] Q,
    output [15:0] R,
```

```
        input  [15:0] N,
        input  [15:0] D);
```

Descrierea semnalelor folosite de acest modul este următoarea:

- `N` - deîmpărţit
- `D` - împărţitor
- `Q` - cât (N / D)
- `R` - rest (N % D)

Operaţiile determinate de operatorii `/` şi `%` nu sunt, în general, sintetizabile în Verilog.

output_display

Modulul are rolul de a aproxima valoarea medie a temperaturii la întregul cel mai apropiat şi de a afişa informaţia codificată. Modulul trebuie să respecte următoarea interfaţă:

```
    module output_display (
        output [7:0] coded_out_o,
        output       alert_o,
        input [15:0] temp_Q_i,
        input [15:0] temp_R_i,
        input  [7:0] active_sensors_nr);
```

Descrierea semnalelor folosite de acest modul este următoarea:

- `coded_out_o` - semnalul codificat pentru ieşire
- `alert_o` - semnalul de alertă, activ dacă ieşirea aproximată este în exteriorul [19:26]
- `temp_Q_i` - câtul împărţirii reprezentând media aritmetică
- `temp_R_i` - restul împărţirii reprezentând media aritmetică
- `active_sensors_nr` - numărul de senzori activi pentru care s-a efectuat media

Bonus

Implementaţi modulul astfel încât el să accepte un număr variabil de senzori la intrare. Se garantează faptul că numărul maxim de senzori este 200 şi că el nu se va modifica pe parcursul rulării. Interfaţa intrare - ieşire **nu** poate fi modificată, cu excepţia declarării variabilelor output de tip reg. Numărul parametrilor acceptaţi pentru interfaţa modulelor este limitat la 1.
Hint: Pentru implementare se poate folosi operatorul `+:` pentru selectarea a unui număr constant de biţi cu un punct de start variabil.

Notare

- +6 pct: implementarea modulului **division**; sunt testate toate combinaţiile valide de operanzi pe 16 biţi, numere exprimate fără semn;
- +4 pct: implementarea corectă a întregului ansamblu, pentru 5 senzori de intrare; vor fi stimulate intrările astfel încât să fie acoperite toate combinaţiile posibile la ieşire.
- +2 pct **bonus:** implementarea admite un număr generic de senzori ca semnal de intrare.
- -12 pct: folosirea construcţiilor nesintetizabile din Verilog (`while`, `repeat`, `for` cu număr variabil de iteraţii, operatorii / şi %, instrucţiuni de întârziere etc.);
- -1 pct: lipsa fişierului README;
- -0.5 pct: pentru fiecare zi de întârziere; tema poate fi trimisă cu maxim 7 zile întârziere faţă de termenul specificat în enunţ (faţă de deadline-ul soft).
- -0.5 pct: folosirea incorectă a atribuirilor continue (assign), blocante (=) şi non-blocante (<=).
- -0.5 pct: indentare haotică
- -0.2 pct: lipsa comentariilor utile
- -0.1 pct: comentarii inutile (ex. wire x; // semnalul x)
- -0.2 pct: diverse alte probleme constatate în implementare (per problemă)

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

Precizări

- Arhiva temei (de tip **zip**) trebuie să cuprindă în rădăcina sa (*fără alte directoare*) **doar**:
  - fişierele sursă (extensia .v);
  - fişierul README.
- Arhiva **nu** trebuie să conţină fişiere de test, fişiere specifice proiectelor etc.
- Fişierului README va conţine minim:
  - numele şi grupa;
  - prezentarea generală a soluţiei alese (ex: descrierea de nivel înalt a algoritmului folosit);
  - explicarea porţiunilor complexe ale implementării (poate fi făcută şi în comentarii);
  - alte detalii relevante.
- Vmchecker ne permite să revenim la orice soluţie încărcată de voi; cereţi revenirea la cea mai convenabilă soluţie trimisă (punctaj teste automate + depunctare întârziere) printr-un mail titularului de laborator.
- Tema trebuie realizată individual; folosirea de porţiuni de cod de la alţi colegi sau de pe Internet (cu excepţia site-ului de curs şi a resurselor puse la dispoziţie în conţinutul temei) poate fi considerată **copiere** şi va fi penalizată conform *regulamentului*.
- Aduceţi-vă aminte de **recomandările** prezentate în Tema 0.
- Se pot implementa mici module de test pentru fiecare bloc pentru a-i verifica, separat, funcţionalitatea. Ele **nu** trebuie încărcate pe platforma vmchecker.

Resurse

- Division Algorithms
- Tutoriale Xilinx ISE
- Verilog 2000 Standard (4. Indexed Vector Part Selects)

Anexă - Exemplu date

Considerăm un sistem format din 5 senzori, având valorile prezentate în tabel:

	temperatură senzor 4	**temperatură senzor 3**	**temperatură senzor 2**	**temperatură senzor 1**	**temperatură senzor 0**
Valoare decimală	19	18	21	25	20
Valoare binară	00010011	00010010	00010101	00011001	00010100
Senzor status	1	1	1	0	1

Astfel, valorile obţinute pentru vectorii de intrare vor fi:

```
sensors_data_i[39:0]    = 40'b00010011_00010010_00010101_00011001_00010100;
sensors_en_i[4:0]       =  5'b11101;
```

În urma operaţiei efectuate de blocul `sensors_input`, valorile de ieşire ale acestuia vor fi:

```
temp_sum_o[15:0]        = 78; // 19+18+21+20
nr_active_sensors_o[7:0]=  4; //  1+1+1+0+1
```

După efectuarea mediei aritmetice cu ajutorul blocului `division`, valorile de ieşire ale acestuia vor fi:

```
Q = 19; // 78 / 4
R = 2;  // 78 % 4
```

Ţinând cont de aproximarea necesară (78 / 4 = 19.5) efectuată de blocul `output_display`, ieşirea blocului va fi:

```
led_output_o[7:0] = 8'b0000_0011; // val 20, conform codificării prezentate
alert_o           = 0;      // valoarea temp se află în range
```