Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут»

Кафедра КЕОА

Лабораторна робота №1

з курсу: «Апаратні прискорювачі обчислень на мікросхемах програмованої логіки»

Виконав:

студент III-го курсу ФЕЛ

група ДК-02

Мачковський Д.В.

25.10.2022

Київ-2022

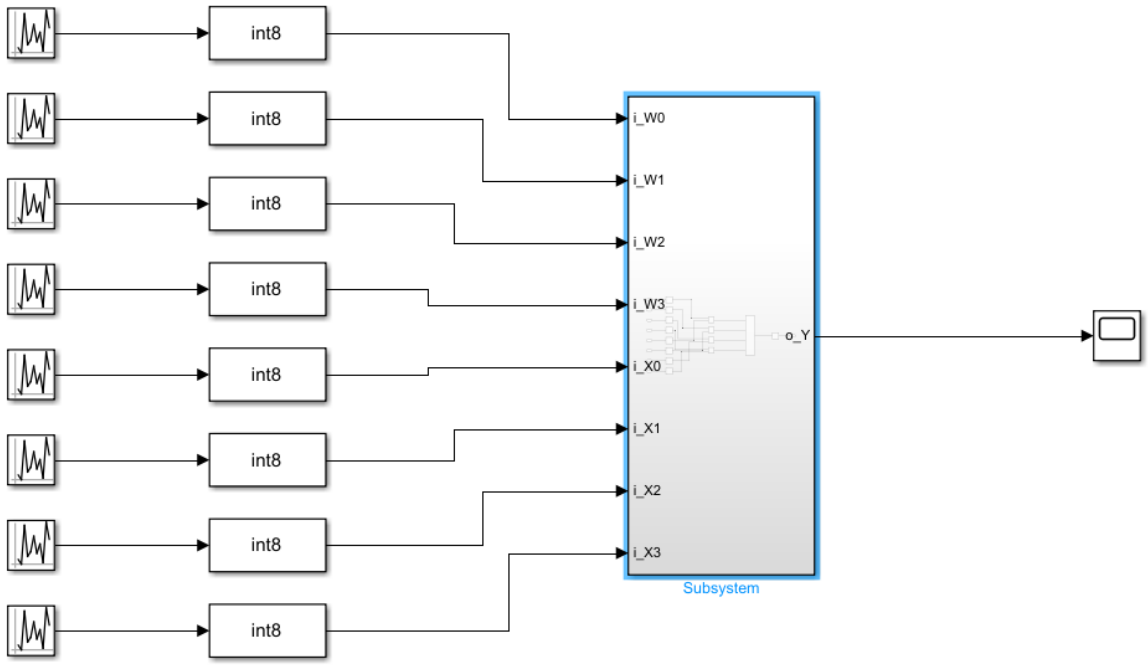
Хід роботи

1. В Simulink реалізувати підсистему, що розраховує функцію:

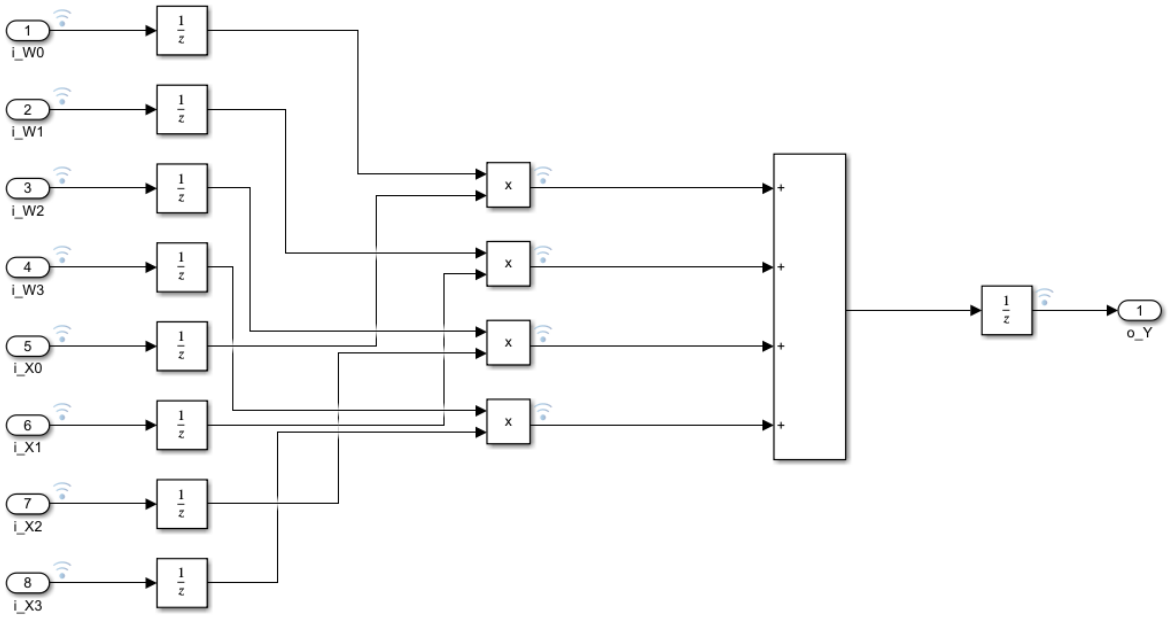
Y = W0*X0 + W1*X1 + W2*X2 + W*X3

Типи даних входів: int8
Тип даних виходу: int16
На входах і виході поставити регістри (блок затримки на 1 такт)

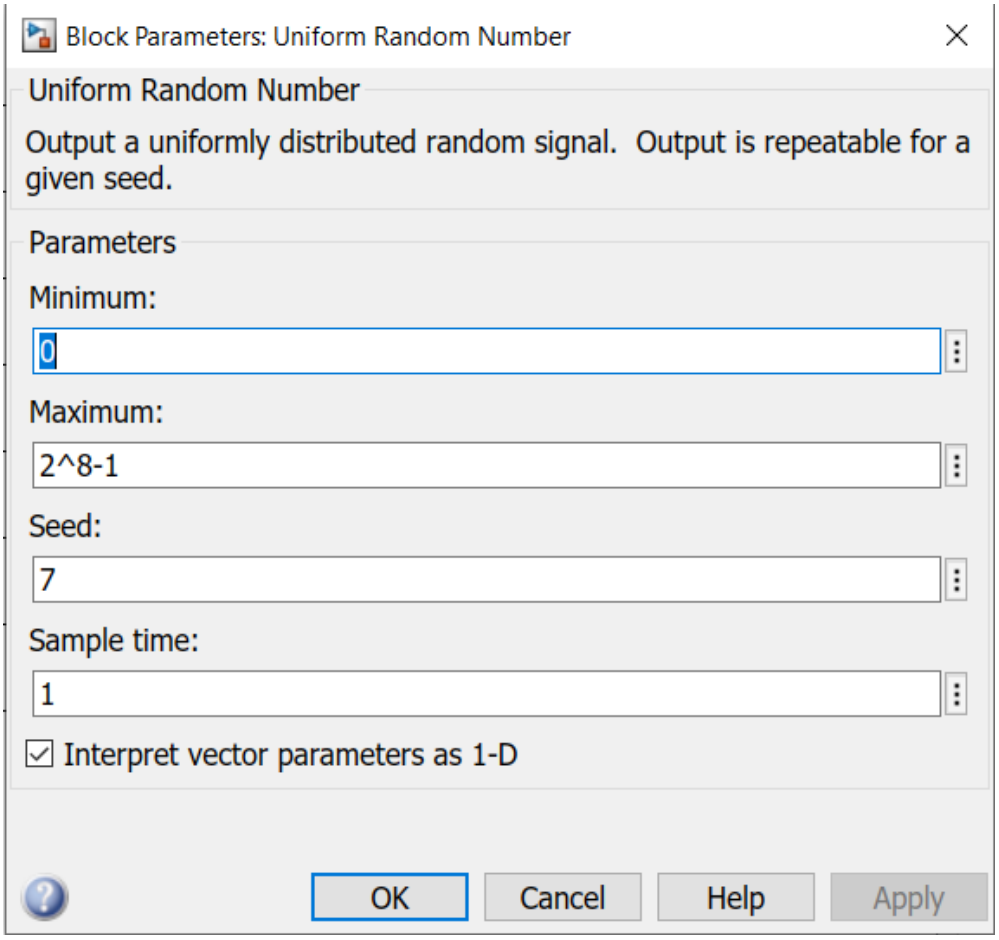
Схема має наступний вигляд:



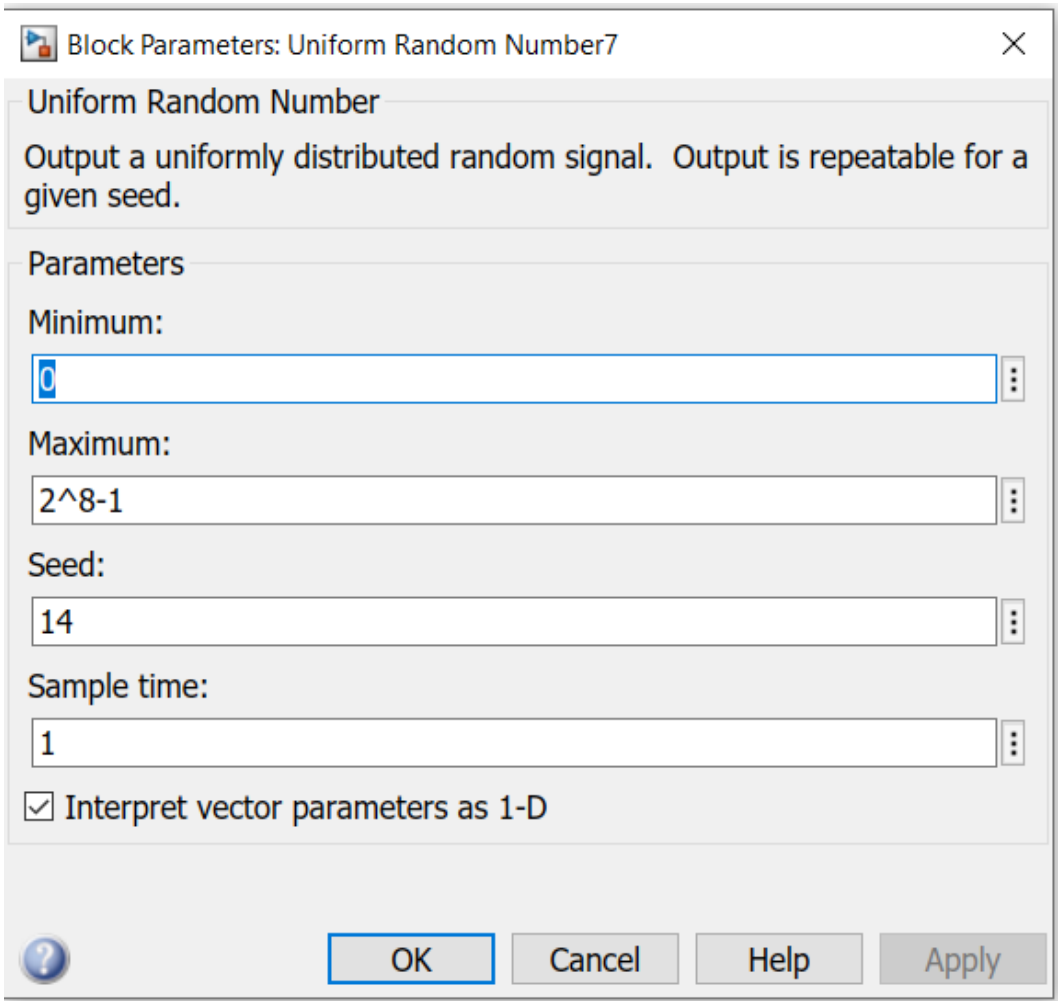
Вигляд всередині блоку Subsystem:



Налаштування першого Uniform Random number:



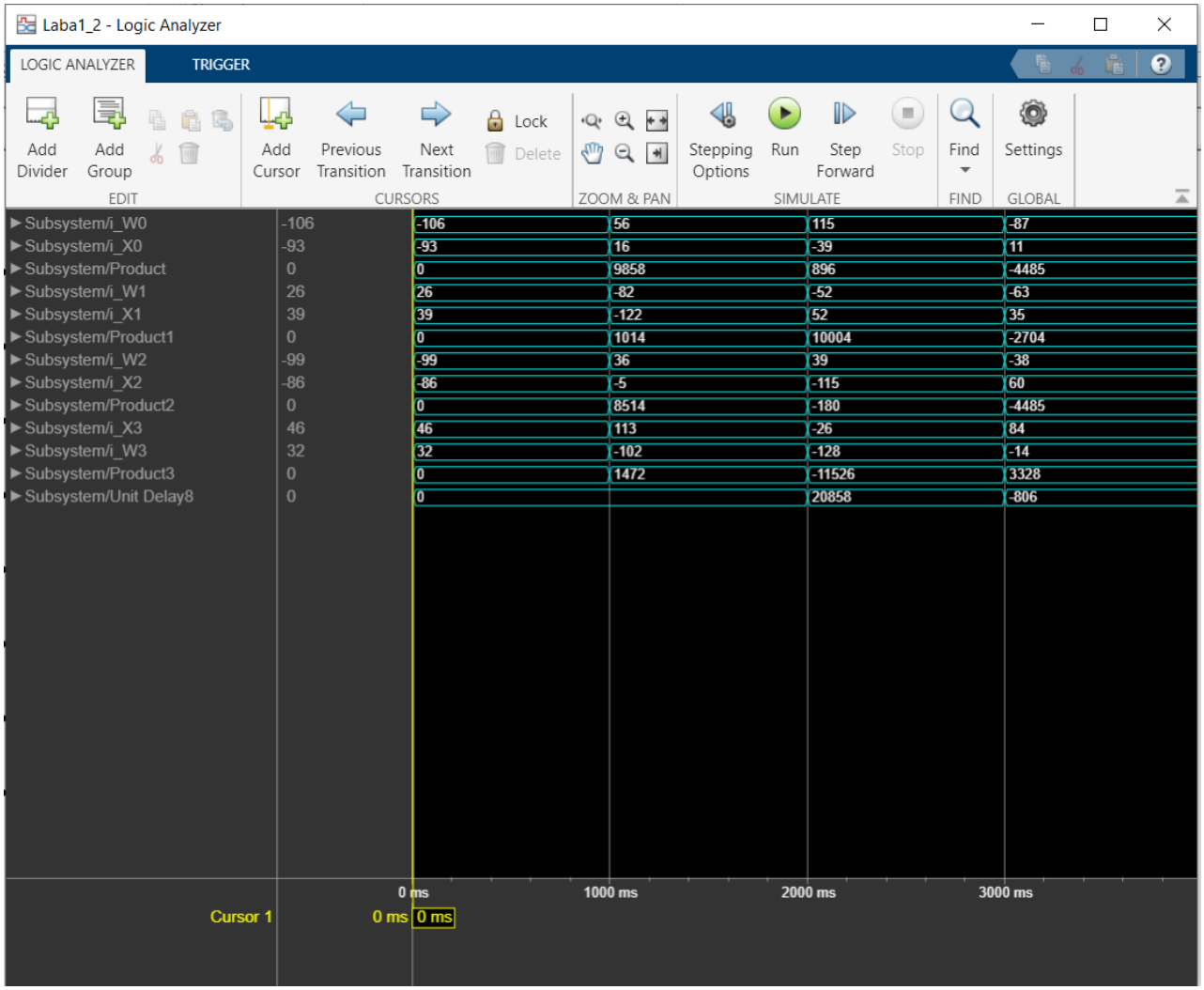
Налаштування останнього Uniform Random number:



Параметр seed починається з 7 і далі збільшується з кількістю uniform random number. Всі інші налаштування були задані на основі параметрів які вказані в методичних матеріалах і їх можна буде продивитися в надісланому проекті.

2. В логічному аналазаторі переглянути дані на входах і на виході створеної підсистеми у знаковому десятковому поданні (форматі).

Результат виглядає наступним чином:



Для контролю результатів були встановлені Log Select Signal після блоків множення також (Subsystem/Product), З чого можна прикладом побачити, що добуток Subsystem/_iW0 і Subsystem/i_X0 -106*-93 = 9858. Аналогічно перевіряються інші W і X. Далі йде сума чотирьох Subsystem/Product, який, перевіривши, дає правильні результати.

3. Додати у звіт згенерований код на Verilog та результат синтезу згенерованого коду в Quartus для створеної підсистеми (звіт по апаратним витратам, результат виклику RTL Viewer).

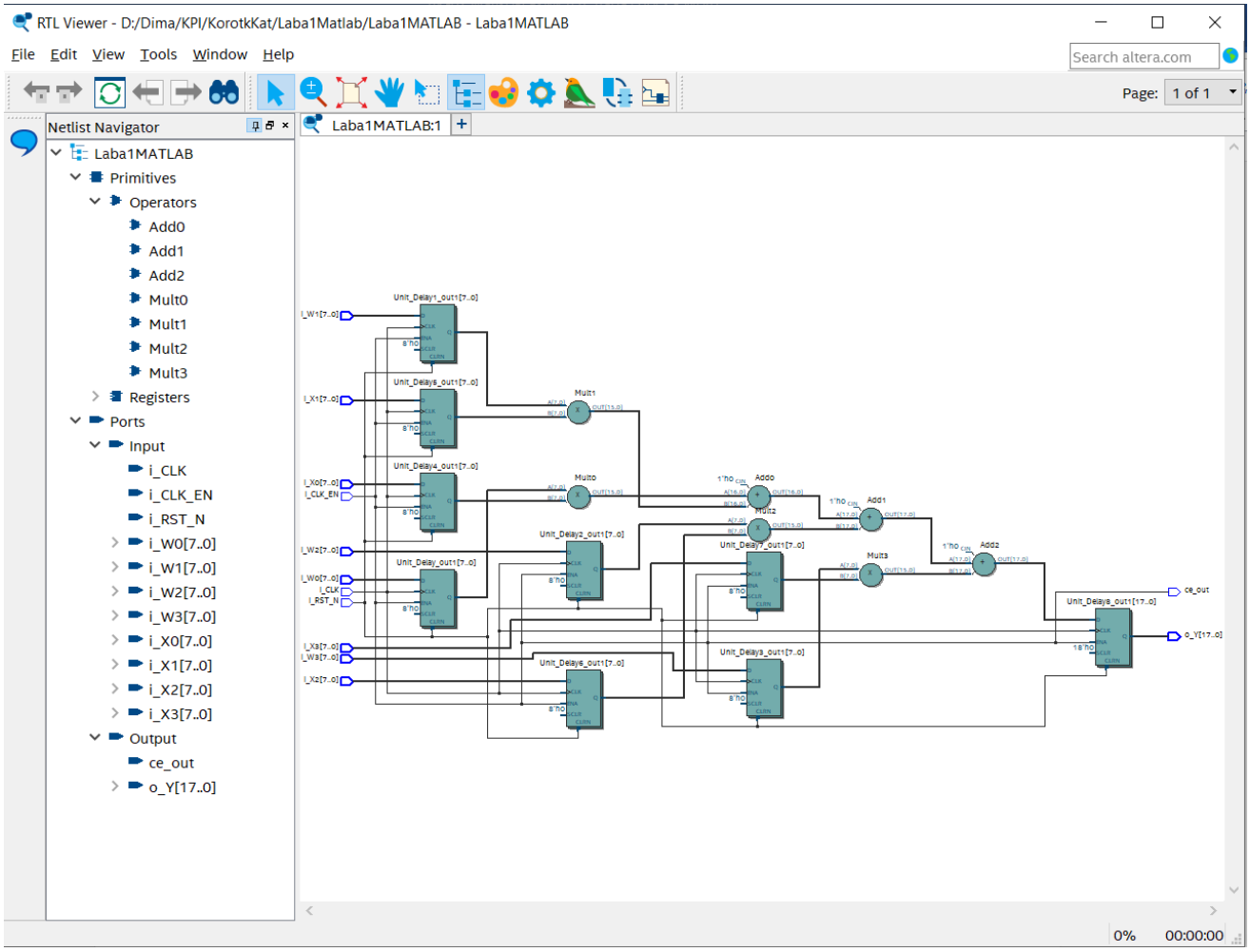
Згенерований Verilog код має наступний вигляд:

```
// -------------------------------------------------------------
//
// File Name: hdlsrc\Laba1_2\Subsystem.v
// Created: 2022-10-18 20:02:47
//
// Generated by MATLAB 9.12 and HDL Coder 3.20
//
//
// -- -----------------------------------------------------------
// -- Rate and Clocking Details
// -- -----------------------------------------------------------
// Model base rate: 1
// Target subsystem base rate: 1
//
//
// Clock Enable  Sample Time
// -- -----------------------------------------------------------
// ce_out        1
// -- -----------------------------------------------------------
//
//
// Output Signal                 Clock Enable  Sample Time
// -- -----------------------------------------------------------
// o_Y                           ce_out        1
// -- -----------------------------------------------------------
//
// -------------------------------------------------------------


// -------------------------------------------------------------
//
// Module: Subsystem
// Source Path: Laba1_2/Subsystem
// Hierarchy Level: 0
//
// -------------------------------------------------------------

`timescale 1 ns / 1 ns

module Laba1MATLAB
          (i_CLK,
           i_RST_N,
           i_CLK_EN,
           i_W0,
           i_W1,
           i_W2,
           i_W3,
           i_X0,
           i_X1,
           i_X2,
           i_X3,
           ce_out,
           o_Y);

  input   i_CLK;
  input   i_RST_N;
  input   i_CLK_EN;
  input   signed [7:0] i_W0;  // int8
  input   signed [7:0] i_W1;  // int8
  input   signed [7:0] i_W2;  // int8
  input   signed [7:0] i_W3;  // int8
  input   signed [7:0] i_X0;  // int8
  input   signed [7:0] i_X1;  // int8
  input   signed [7:0] i_X2;  // int8
  input   signed [7:0] i_X3;  // int8
  output  ce_out;
  output  signed [17:0] o_Y;  // sfix18


  wire enb;
  reg signed [7:0] Unit_Delay_out1;  // int8
  reg signed [7:0] Unit_Delay1_out1;  // int8
  reg signed [7:0] Unit_Delay2_out1;  // int8
  reg signed [7:0] Unit_Delay3_out1;  // int8
  reg signed [7:0] Unit_Delay4_out1;  // int8
  wire signed [15:0] Product_out1;  // int16
  reg signed [7:0] Unit_Delay5_out1;  // int8
  wire signed [15:0] Product1_out1;  // int16
  wire signed [16:0] Add_stage2_1;  // sfix17
  wire signed [16:0] Add_stage2_2;  // sfix17
  wire signed [16:0] Add_op_stage1;  // sfix17
  reg signed [7:0] Unit_Delay6_out1;  // int8
  wire signed [15:0] Product2_out1;  // int16
  wire signed [17:0] Add_stage3_1;  // sfix18
  wire signed [17:0] Add_stage3_2;  // sfix18
  wire signed [17:0] Add_op_stage2;  // sfix18
  reg signed [7:0] Unit_Delay7_out1;  // int8
  wire signed [15:0] Product3_out1;  // int16
  wire signed [17:0] Add_stage4_1;  // sfix18
  wire signed [17:0] Add_out1;  // sfix18
  reg signed [17:0] Unit_Delay8_out1;  // sfix18


  assign enb = i_CLK_EN;
```

```
96
97      always @(posedge i_CLK or negedge i_RST_N)
98        begin : Unit_Delay_process
99          if (i_RST_N == 1'b0) begin
100           Unit_Delay_out1 <= 8'sb00000000;
101         end
102         else begin
103           if (enb) begin
104             Unit_Delay_out1 <= i_W0;
105           end
106         end
107       end
108
109
110
111     always @(posedge i_CLK or negedge i_RST_N)
112       begin : Unit_Delay1_process
113         if (i_RST_N == 1'b0) begin
114           Unit_Delay1_out1 <= 8'sb00000000;
115         end
116         else begin
117           if (enb) begin
118             Unit_Delay1_out1 <= i_W1;
119           end
120         end
121       end
122
123
124
125     always @(posedge i_CLK or negedge i_RST_N)
126       begin : Unit_Delay2_process
127         if (i_RST_N == 1'b0) begin
128           Unit_Delay2_out1 <= 8'sb00000000;
129         end
130         else begin
131           if (enb) begin
132             Unit_Delay2_out1 <= i_W2;
133           end
134         end
135       end
136
137
138
139     always @(posedge i_CLK or negedge i_RST_N)
140       begin : Unit_Delay3_process
141         if (i_RST_N == 1'b0) begin
142           Unit_Delay3_out1 <= 8'sb00000000;
143         end
144         else begin
145           if (enb) begin
146             Unit_Delay3_out1 <= i_W3;
147           end
148         end
149       end
150
151
152
153     always @(posedge i_CLK or negedge i_RST_N)
154       begin : Unit_Delay4_process
155         if (i_RST_N == 1'b0) begin
156           Unit_Delay4_out1 <= 8'sb00000000;
157         end
158         else begin
159           if (enb) begin
160             Unit_Delay4_out1 <= i_X0;
161           end
162         end
163       end
164
165
166
167     assign Product_out1 = Unit_Delay_out1 * Unit_Delay4_out1;
168
169
170
171     always @(posedge i_CLK or negedge i_RST_N)
172       begin : Unit_Delay5_process
173         if (i_RST_N == 1'b0) begin
174           Unit_Delay5_out1 <= 8'sb00000000;
175         end
176         else begin
177           if (enb) begin
178             Unit_Delay5_out1 <= i_X1;
179           end
180         end
181       end
182
183
184
185     assign Product1_out1 = Unit_Delay1_out1 * Unit_Delay5_out1;
186
187
```

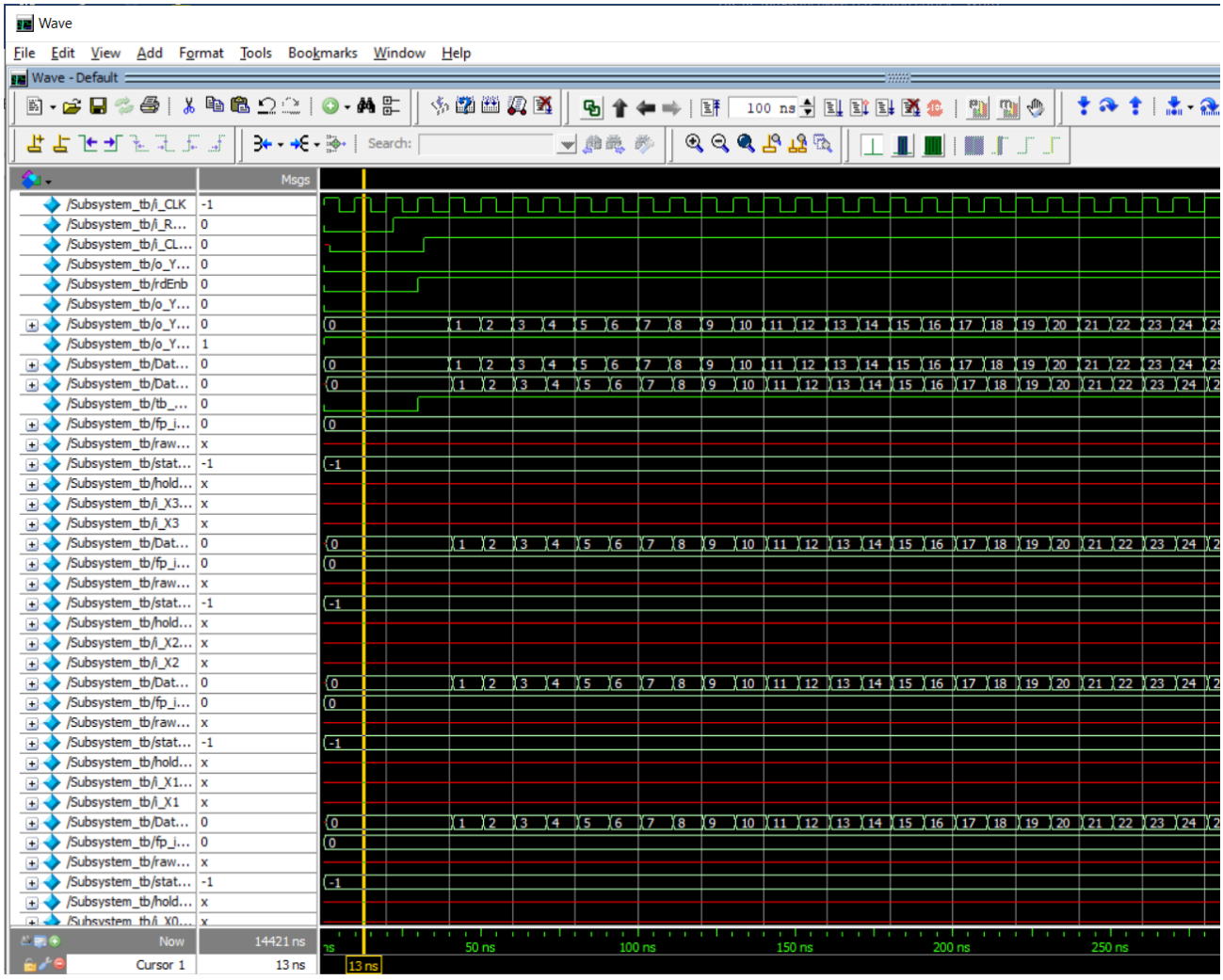
```
  assign Add_stage2_1 = {Product_out1[15], Product_out1};
  assign Add_stage2_2 = {Product1_out1[15], Product1_out1};
  assign Add_op_stage1 = Add_stage2_1 + Add_stage2_2;



  always @(posedge i_CLK or negedge i_RST_N)
    begin : Unit_Delay6_process
      if (i_RST_N == 1'b0) begin
        Unit_Delay6_out1 <= 8'sb00000000;
      end
      else begin
        if (enb) begin
          Unit_Delay6_out1 <= i_X2;
        end
      end
    end



  assign Product2_out1 = Unit_Delay2_out1 * Unit_Delay6_out1;



  assign Add_stage3_1 = {Add_op_stage1[16], Add_op_stage1};
  assign Add_stage3_2 = {{2{Product2_out1[15]}}, Product2_out1};
  assign Add_op_stage2 = Add_stage3_1 + Add_stage3_2;



  always @(posedge i_CLK or negedge i_RST_N)
    begin : Unit_Delay7_process
      if (i_RST_N == 1'b0) begin
        Unit_Delay7_out1 <= 8'sb00000000;
      end
      else begin
        if (enb) begin
          Unit_Delay7_out1 <= i_X3;
        end
      end
    end



  assign Product3_out1 = Unit_Delay3_out1 * Unit_Delay7_out1;



  assign Add_stage4_1 = {{2{Product3_out1[15]}}, Product3_out1};
  assign Add_out1 = Add_op_stage2 + Add_stage4_1;



  always @(posedge i_CLK or negedge i_RST_N)
    begin : Unit_Delay8_process
      if (i_RST_N == 1'b0) begin
        Unit_Delay8_out1 <= 18'sb000000000000000000;
      end
      else begin
        if (enb) begin
          Unit_Delay8_out1 <= Add_out1;
        end
      end
    end



  assign o_Y = Unit_Delay8_out1;

  assign ce_out = i_CLK_EN;

endmodule  // Subsystem
```

Результат синтезу в RTL Viewer:



5. Створити тестбенч в Matlab для створеної підсистеми і додати в звіт результат симуляції тестбенча в Modelsim/Questasim.

Результат симуляції створеного тестбенчу:


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/Subsystem_tb/i_CLK -1
/Subsystem_tb/i_R... 0
/Subsystem_tb/i_CL... 0
/Subsystem_tb/o_Y... 0
/Subsystem_tb/rdEnb 0
/Subsystem_tb/o_Y... 0
/Subsystem_tb/o_Y... 0
/Subsystem_tb/o_Y... 1
/Subsystem_tb/Dat... 0
/Subsystem_tb/Dat... 0
/Subsystem_tb/tb_... 0
/Subsystem_tb/fp_i... 0
/Subsystem_tb/raw... x
/Subsystem_tb/stat... -1
/Subsystem_tb/hold... x
/Subsystem_tb/i_X3... x
/Subsystem_tb/i_X3 x
/Subsystem_tb/Dat... 0
/Subsystem_tb/fp_i... 0
/Subsystem_tb/raw... x
/Subsystem_tb/stat... -1
/Subsystem_tb/hold... x
/Subsystem_tb/i_X2... x
/Subsystem_tb/i_X2 x
/Subsystem_tb/Dat... 0
/Subsystem_tb/fp_i... 0
/Subsystem_tb/raw... x
/Subsystem_tb/stat... -1
/Subsystem_tb/hold... x
/Subsystem_tb/i_X1... x
/Subsystem_tb/i_X1 x
/Subsystem_tb/Dat... 0
/Subsystem_tb/fp_i... 0
/Subsystem_tb/raw... x
/Subsystem_tb/stat... -1
/Subsystem_tb/hold... x
Now 14421 ns
Cursor 1 13 ns
50 ns
100 ns
150 ns
200 ns
250 ns
13 ns