

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
Гарькавий Д.В.
25.10.2022

Київ-2022

Хід роботи

1. В Simulink реалізувати підсистему, що розраховує функцію:

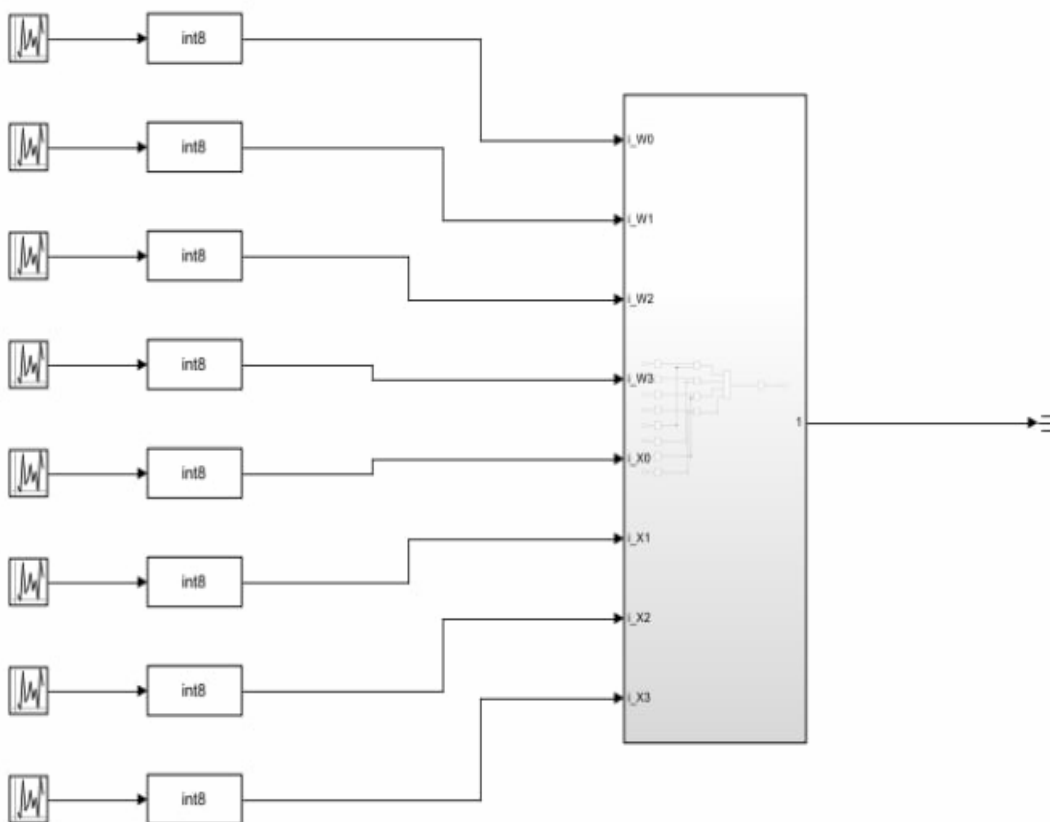
$$Y = W0*X0 + W1*X1 + W2*X2 + W3*X3$$

Типи даних входів: int8

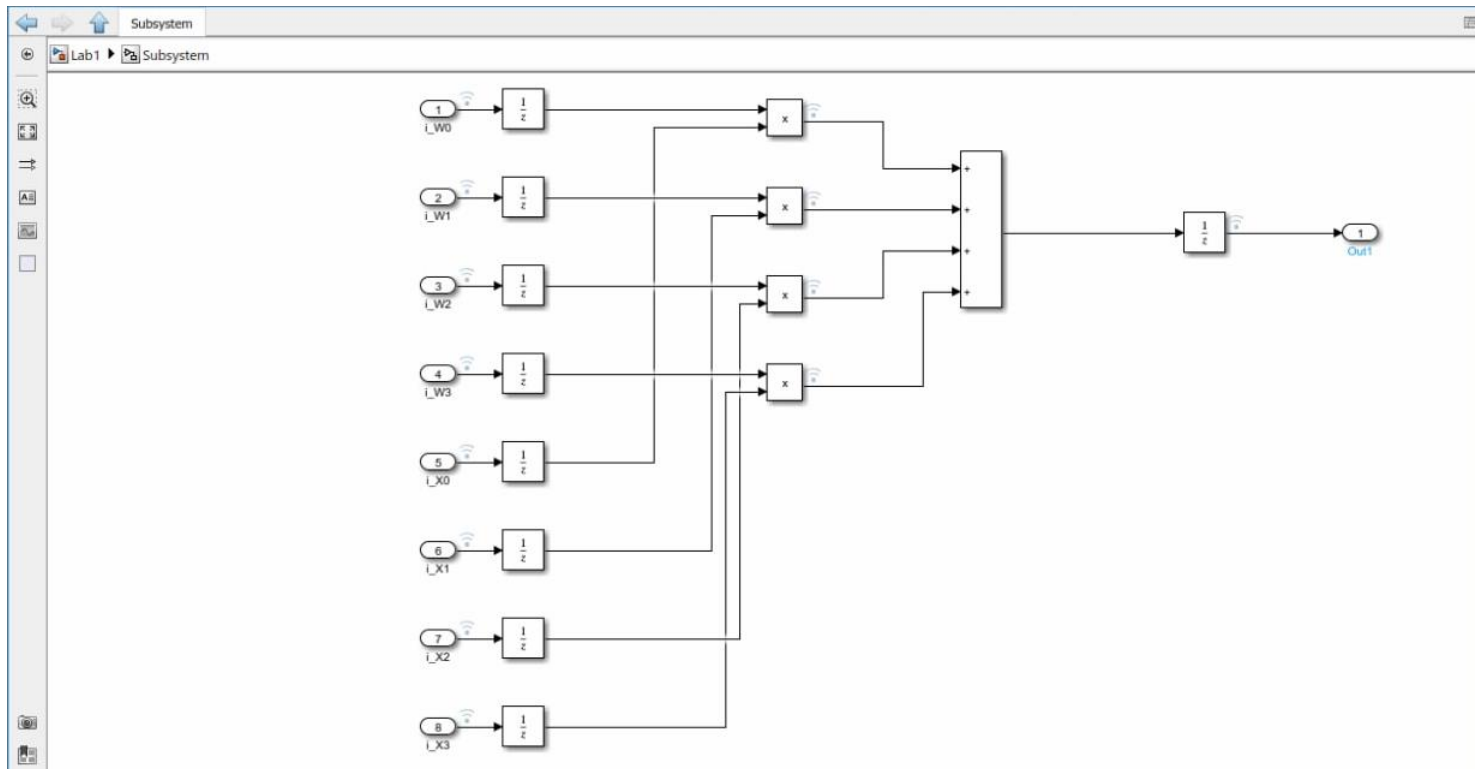
Тип даних виходу: int16

На входах і виході поставити регістри (блок затримки на 1 такт)

Схема:

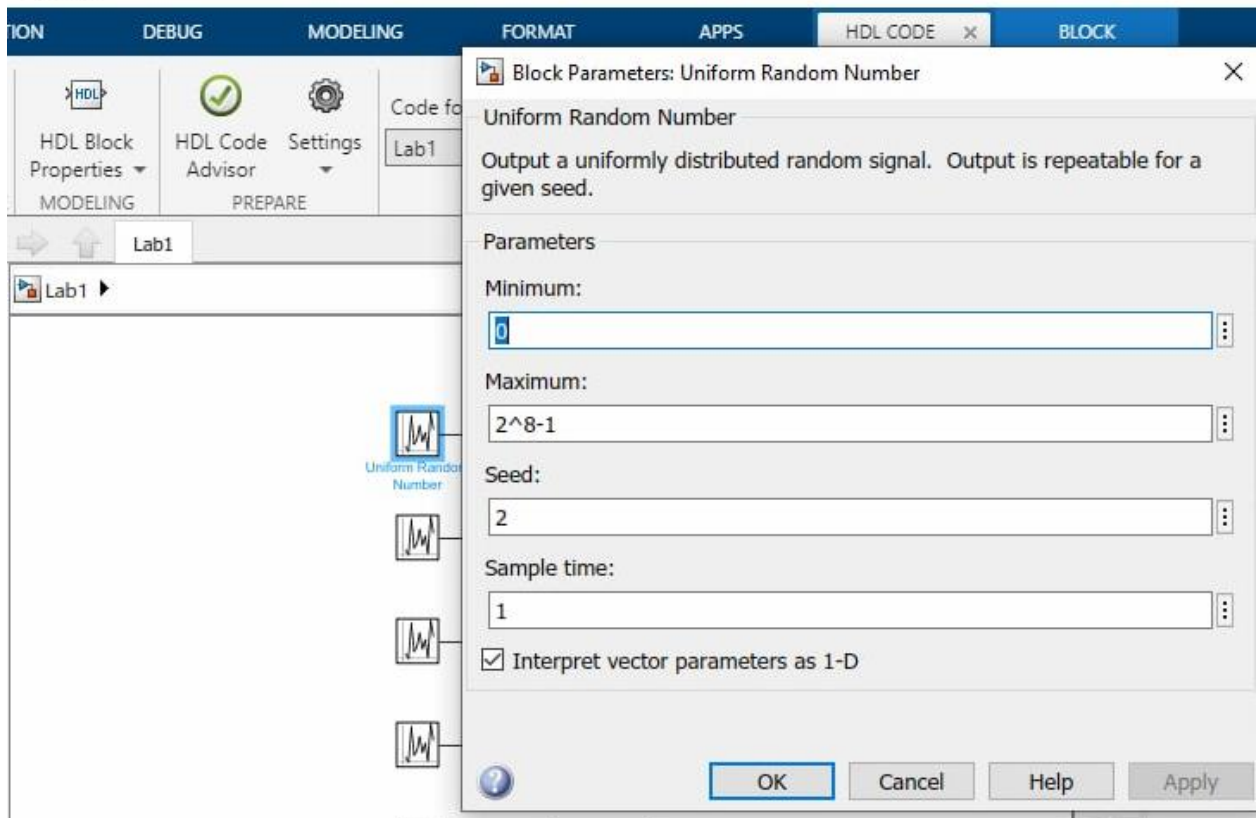


Блок Subsystem(PROCESSING_UNIT):



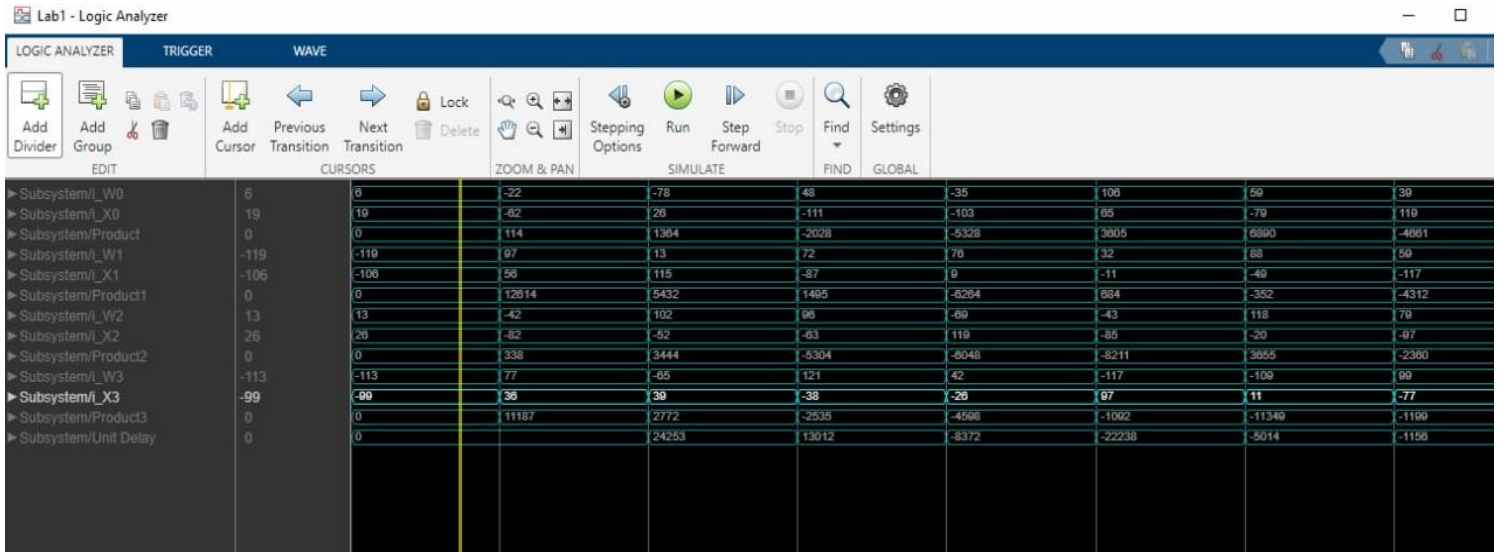
Налаштування одного з блоків “Uniform Random Number” (seed = 2, згідно мого варіанту, надалі збільшую значення кожного на 1) інші параметри вказані, як у методичних матеріалах:

- Simulink



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

Приклад результату:



The screenshot shows the Logic Analyzer interface with a table of signal values. The table has 10 columns representing different signals and 10 rows representing different time steps. The signals are: Subsystem/I_W0, Subsystem/I_X0, Subsystem/Product, Subsystem/I_W1, Subsystem/I_X1, Subsystem/Product1, Subsystem/I_W2, Subsystem/I_X2, Subsystem/Product2, Subsystem/I_W3, Subsystem/I_X3, Subsystem/Product3, and Subsystem/Unit Delay. The values are displayed in decimal format.

Signal	Row 1	Row 2	Row 3	Row 4	Row 5	Row 6	Row 7	Row 8	Row 9	Row 10
Subsystem/I_W0	6	19	0	-119	-106	0	13	26	0	-113
Subsystem/I_X0	19	0	0	-119	-106	0	13	26	0	-113
Subsystem/Product	0	0	114	1304	-2028	-5328	3805	8880	-4861	
Subsystem/I_W1	-119	-106	0	13	26	0	13	26	0	-113
Subsystem/I_X1	-106	0	0	13	26	0	13	26	0	-113
Subsystem/Product1	0	0	12614	5432	1495	-8284	884	-352	-4312	
Subsystem/I_W2	13	26	0	13	26	0	13	26	0	-113
Subsystem/I_X2	26	0	0	13	26	0	13	26	0	-113
Subsystem/Product2	0	0	338	3444	-5304	-6048	-8211	3055	-2360	
Subsystem/I_W3	-113	-99	38	38	-38	-26	97	11	-77	
Subsystem/I_X3	-99	38	38	-38	-38	-26	97	11	-77	
Subsystem/Product3	0	0	11187	2772	-2535	-4568	-1002	-11340	-1109	
Subsystem/Unit Delay	0	0	24253	13012	-8372	-22238	-5014	-1156		

Перевірка:

$6 * 19 + (-119) * (-106) + 13 * 26 + (-113) * (-99) = 114 + 12614 + 338 + 11187 = 24253$, як і на скріншоті результату.

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).

Згенерував код, який має наступний вигляд:

```

1 // -----
2
3 File Name: hdlsrc\Lab1\Subsystem.v
4 Created: 2022-10-25 20:22:07
5
6 Generated by MATLAB 9.12 and HDL Coder 3.20
7
8 // -----
9
10 -- Rate and Clocking Details
11 -----
12 Model base rate: 1
13 Target subsystem base rate: 1
14
15 //
16 Clock Enable Sample Time
17 -----
18 ce_out 1
19 -----
20
21 //
22 output signal Clock Enable Sample Time
23 -----
24 out1 ce_out 1
25 -----
26
27 // -----
28
29 //
30 -----
31
32 Module: Subsystem
33 Source Path: Lab1/Subsystem
34 Hierarchy Level: 0
35 -----
36
37
38 `timescale 1 ns / 1 ns
39
40 module LAB1M
41     (i_CLK,
42      i_RST_N,
43      i_CLK_EN,
44      i_w0,
45      i_w1,
46      i_w2,
47      i_w3,
48      i_x0,
49      i_x1,
50      i_x2,
51      i_x3,
52      ce_out,
53      out1);
54

```

```

55 L
56 input i_CLK;
57 input i_RST_N;
58 input i_CLK_EN;
59 input signed [7:0] i_w0; // int8
60 input signed [7:0] i_w1; // int8
61 input signed [7:0] i_w2; // int8
62 input signed [7:0] i_w3; // int8
63 input signed [7:0] i_x0; // int8
64 input signed [7:0] i_x1; // int8
65 input signed [7:0] i_x2; // int8
66 input signed [7:0] i_x3; // int8
67 output ce_out;
68 output signed [15:0] out1; // int16
69
70
71 wire enb;
72 reg signed [7:0] Unit_Delay8_out1; // int8
73 reg signed [7:0] Unit_Delay7_out1; // int8
74 reg signed [7:0] Unit_Delay6_out1; // int8
75 reg signed [7:0] Unit_Delay5_out1; // int8
76 reg signed [7:0] Unit_Delay4_out1; // int8
77 wire signed [15:0] Product_out1; // int16
78 reg signed [7:0] Unit_Delay3_out1; // int8
79 wire signed [15:0] Product1_out1; // int16
80 wire signed [15:0] Add_stage2_add_temp; // sfix16
81 wire signed [16:0] Add_op_stage1; // sfix17
82 reg signed [7:0] Unit_Delay2_out1; // int8
83 wire signed [15:0] Product2_out1; // int16
84 wire signed [15:0] Add_stage3_add_cast; // sfix16
85 wire signed [15:0] Add_stage3_add_temp; // sfix16
86 wire signed [17:0] Add_op_stage2; // sfix18
87 reg signed [7:0] Unit_Delay1_out1; // int8
88 wire signed [15:0] Product3_out1; // int16
89 wire signed [15:0] Add_stage4_add_cast; // sfix16
90 wire signed [15:0] Add_out1; // int16
91 reg signed [15:0] Unit_Delay_out1; // int16
92

```

```

92
93
94 assign enb = i_CLK_EN;
95
96 always @(posedge i_CLK or posedge i_RST_N)
97 begin : Unit_Delay8_process
98     if (i_RST_N == 1'b1) begin
99         Unit_Delay8_out1 <= 8'sb00000000;
100     end
101     else begin
102         if (enb) begin
103             Unit_Delay8_out1 <= i_w0;
104         end
105     end
106 end
107
108
109
110 always @(posedge i_CLK or posedge i_RST_N)
111 begin : Unit_Delay7_process
112     if (i_RST_N == 1'b1) begin
113         Unit_Delay7_out1 <= 8'sb00000000;
114     end
115     else begin
116         if (enb) begin
117             Unit_Delay7_out1 <= i_w1;
118         end
119     end
120 end
121
122
123
124 always @(posedge i_CLK or posedge i_RST_N)
125 begin : Unit_Delay6_process
126     if (i_RST_N == 1'b1) begin
127         Unit_Delay6_out1 <= 8'sb00000000;
128     end
129     else begin
130         if (enb) begin
131             Unit_Delay6_out1 <= i_w2;
132         end
133     end
134 end
135
136
137
138 always @(posedge i_CLK or posedge i_RST_N)
139 begin : Unit_Delay5_process
140     if (i_RST_N == 1'b1) begin
141         Unit_Delay5_out1 <= 8'sb00000000;
142     end
143     else begin
144         if (enb) begin
145             Unit_Delay5_out1 <= i_w3;
146         end
147     end
148 end
149
150
151
152 always @(posedge i_CLK or posedge i_RST_N)
153 begin : Unit_Delay4_process
154     if (i_RST_N == 1'b1) begin
155         Unit_Delay4_out1 <= 8'sb00000000;
156     end
157     else begin
158         if (enb) begin
159             Unit_Delay4_out1 <= i_x0;
160         end
161     end
162 end
163
164
165
166 assign Product_out1 = Unit_Delay8_out1 * Unit_Delay4_out1;
167
168
169
170 always @(posedge i_CLK or posedge i_RST_N)
171 begin : Unit_Delay3_process
172     if (i_RST_N == 1'b1) begin
173         Unit_Delay3_out1 <= 8'sb00000000;
174     end
175     else begin
176         if (enb) begin
177             Unit_Delay3_out1 <= i_x1;
178         end
179     end
180 end
181
182
183
184 assign Product1_out1 = Unit_Delay7_out1 * Unit_Delay3_out1;
185
186
187
188 assign Add_stage2_add_temp = Product_out1 + Product1_out1;
189 assign Add_op_stage1 = {Add_stage2_add_temp[15], Add_stage2_add_temp};
190
191
192

```

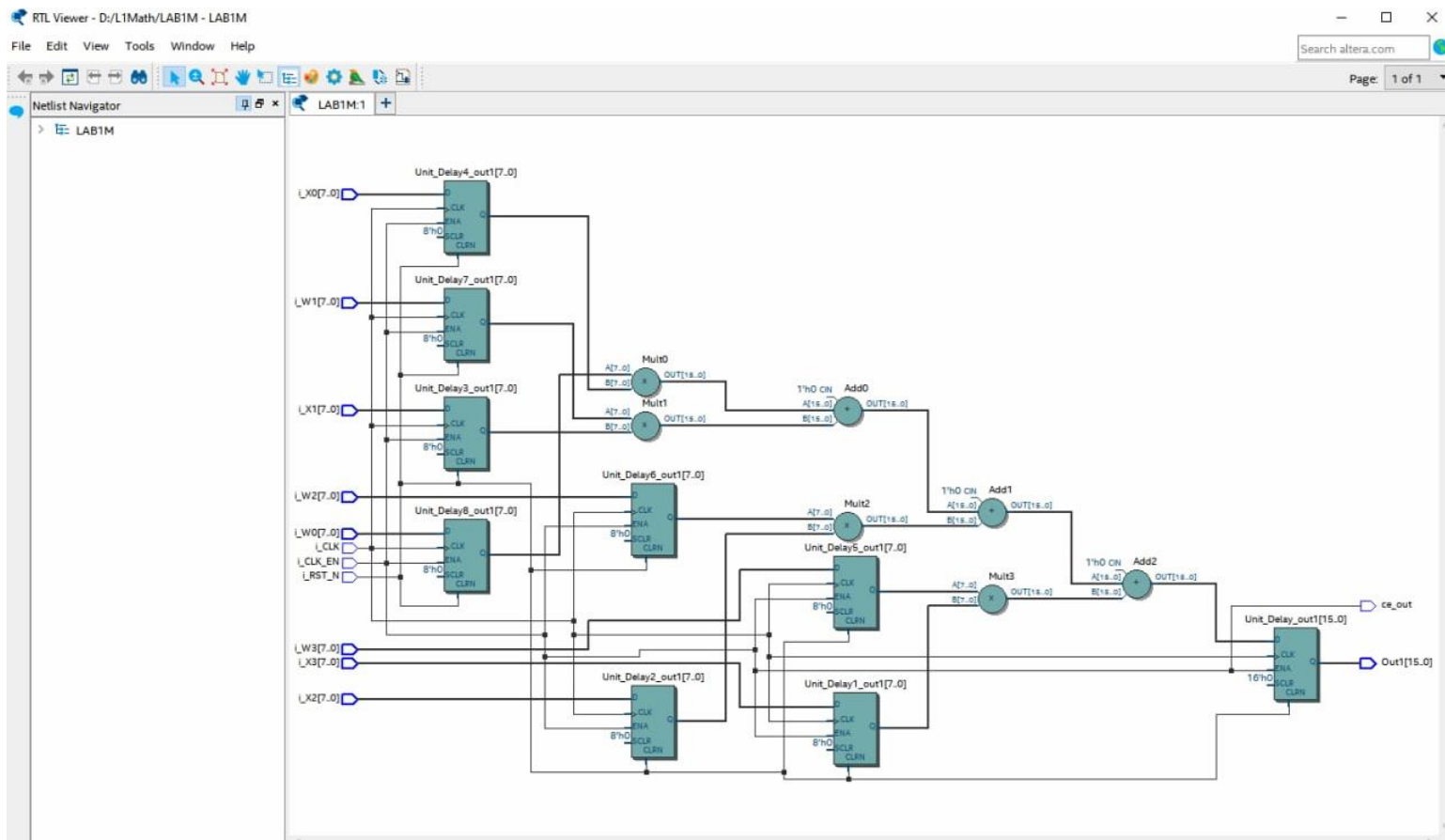


```

192
193 always @(posedge i_CLK or posedge i_RST_N)
194 begin : Unit_Delay2_process
195 if (i_RST_N == 1'b1) begin
196 Unit_Delay2_out1 <= 8'sb00000000;
197 end
198 else begin
199 if (enb) begin
200 Unit_Delay2_out1 <= i_X2;
201 end
202 end
203 end
204
205
206
207 assign Product2_out1 = Unit_Delay6_out1 * Unit_Delay2_out1;
208
209
210
211 assign Add_stage3_add_cast = Add_op_stage1[15:0];
212 assign Add_stage3_add_temp = Add_stage3_add_cast + Product2_out1;
213 assign Add_op_stage2 = {{2{Add_stage3_add_temp[15]}}}, Add_stage3_add_tem
214
215
216
217 always @(posedge i_CLK or posedge i_RST_N)
218 begin : Unit_Delay1_process
219 if (i_RST_N == 1'b1) begin
220 Unit_Delay1_out1 <= 8'sb00000000;
221 end
222 else begin
223 if (enb) begin
224 Unit_Delay1_out1 <= i_X3;
225 end
226 end
227 end
228
229
230
231 assign Product3_out1 = Unit_Delay5_out1 * Unit_Delay1_out1;
232
233
234
235 assign Add_stage4_add_cast = Add_op_stage2[15:0];
236 assign Add_out1 = Add_stage4_add_cast + Product3_out1;
237
238
239
240 always @(posedge i_CLK or posedge i_RST_N)
241 begin : Unit_Delay_process
242 if (i_RST_N == 1'b1) begin
243 Unit_Delay_out1 <= 16'sb0000000000000000;
244 end
245 else begin
246 if (enb) begin
247 Unit_Delay_out1 <= Add_out1;
248 end
249 end
250 end
251
252
253
254 assign out1 = Unit_Delay_out1;
255
256 assign ce_out = i_CLK_EN;
257
258 endmodule // subsystem
259
260

```

Результат виклику RTL Viewer:



Висновок

На цій лабораторній роботі я проглянувши відео-знайомство з середовищем Matlab створив підсистему і перевінив її на справність, також просимулювавши її в Matlab і синтезувавши її в Quartus Prime можу стверджувати про правильність виконання.