Лабораторна робота № 1

Опис логічних схем за допомогою мови VHDL

1. Мета роботи:

Отримання основних навиків опису цифрових схем за допомогою мови опису апаратури VHDL. Виконати моделювання логічних схем за допомогою текстового редактора САПР QUARTUS II**.**

2. Завдання для виконання практичної роботи:

1. Вивчити основні елементи мови VHDL і правила опису логічних схем.
2. Зробити опис цифрових пристроїв згідно приведених нижче прикладів за допомогою текстового редактора САПР QUARTUS II.
3. Виконати симуляцію роботи схем. Перевірити правильність роботи пристрою по часових діаграмах.
4. Відповісти на контрольні запитання, оформити звіт про виконану роботу.

2.1. Синтез та дослідження напівсуматора

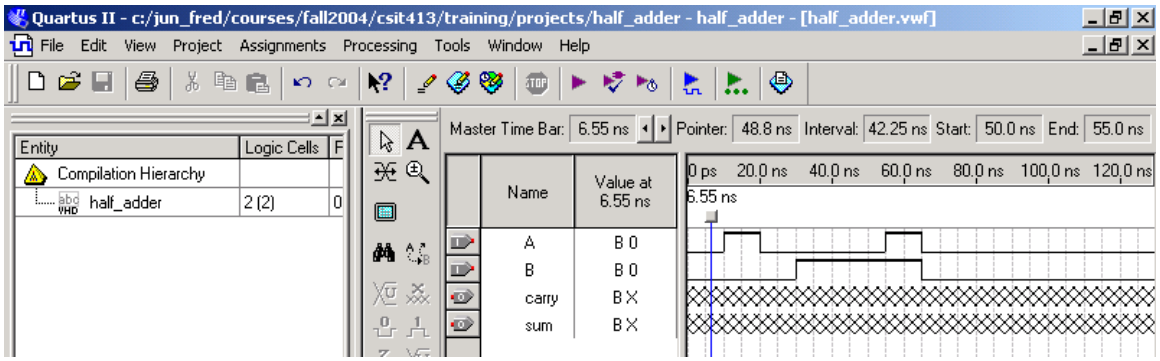
1. Вибрати ***File→ New→VHDL file***
2. Надрукувати наступний код у вікні редактора:

```
library ieee;
use ieee.std_logic_1164.all;
entity half_adder is
port (A,B:in std_logic;
sum,carry:out std_logic);
end half_adder;

--architecture section
architecture my_adder of half_adder is
begin
sum<=A xor B;
carry<= A and B;
end my_adder;
```

3. Зберегти файл як **“half_adder.vhd”** і вибрати створення новогу проекту на основі цього файла.
4. Запустити процес компіляції ***Processing→Start Compilation.***
5. Відслідковувати повідомлення в процесі виконання компіляції.
6. Вибрати ***File→New→Other Files→Vector Waveform Files.***
7. Отримуємо дві секції активного вікна. За допомогою лівої клавіші миші в лівій секції відкрити стовпці **“Name”** і **“Value at”.**
8. Натиснути праву клавішу миші і вибрати **“Insert Node or Bus…”** із контекстного меню за допомогою лівої клавіші миші.
9. Натиснути **“Node Finder”** в діалоговому вікні і вибрати опцію **“List”.**
10. Всі лінії Входів/Виходів (I/O) проекту відображуються в лівій стороні. Вибрати всіх їх і натиснути на знак **“>>”** інструментів вибору для пересилання їх всіх в праву частину.

11. Вибрати **“OK”**, діалогове вікно закривається, повертаємось назад в редактор часових діаграм.
12. Всі лінії входів/виходів (I/O) тепер видимі в стовпці **“Name”.** Вибрати кожну із них і відредагувати часові діаграми для отримання приведеного нижче вигляду. Звертаємо увагу, що періоди часу між вертикальними лініями зміни станів рівні 5ns.



13. Тепер вже все готово для запуску симуляції. Зберегти файл часових діаграм як **“half_adder.vwf”** і вибрати ***Processing→Generate Functional Simulation Netlist.***
14. Запустити симуляцію із меню ***Processing→Start Simulation*** або вибором іконки на панелі інструментів.
15. За повідомленням про успішне завершення в новому вікні відображаються результуючі часові діаграми. Перевірити, що сума і переповнення виконується правильно відповідно до значень вхідних сигналів A і B. Можете позбутися завад у вигляді коротких імпульсів на виході “sum” ***by bringing*** двох входів на нуль в різний час. Відредагуйте файл .vwf і повторіть кроки 13 і 14.

2.2. Синтез та дослідження суматора

1. Вибрати ***File→ New→VHDL file.***
2. Надрукувати наступний код у вікні редактора:

```
library ieee;
use ieee.std_logic_1164.all;
entity full_adder is
port (X,Y,Cin:in std_logic;
sum,Cout:out std_logic);
end full_adder;

--architecture section
architecture my_fulladder of full_adder is
signal S1,C1,S2,C2:std_logic;
component half_adder
port (A,B:in std_logic;
sum,carry:out std_logic);
end component;
begin
h_add0:half_adder port map(X,Y,S1,C1);
h_add1:half_adder port map(S1,Cin,sum,C2);
```

Cout<= C1 or C2;
end my_fulladder;

3. Зберегти файл як **“full_adder.vhd”** і вибрати створення нового проекту на основі цього файлу в тій же директорії, де збережений файл half_adder.vhd. Не звертайте увагу на попередження директорії.
4. Натисніть піктограму компіляції або виберіть ***Processing→Start Compilation***
5. Відслідковувати повідомлення в процесі виконання компіляції. Запам’ятайте найбільшу затримку розповсюдження від вхідного виводу до вихідного. (tpd= ?)
6. Вибрати ***File→New→Other Files→Vector Waveform Files***
7. Отримуємо дві секції активного вікна. За допомогою лівої клавіші миші в лівій секції відкрити стовпці **“Name”** і **“Value at”.**
8. Натиснути праву клавішу миші і вибрати **“Insert Node or Bus…”** із контекстного меню за допомогою лівої клавіші миші.
9. Натиснути **“Node Finder”** в діалоговому вікні і вибрати опцію **“List”.**
10. Всі лінії входів/виходів (I/O) проекту відображуються в лівій стороні. Вибрати всіх їх і натиснути на знак **“>>”** інструментів вибору для пересилання всіх їх в праву частину. Вибрати **“OK”**, діалогове вікно закривається, повертаємось назад в редактор часових діаграм.
11. Всі лінії Вхід/Вихід видимі тепер в стовпці **“Name”**. Вибрати вхідні лінії одна за одною і відредагувати часові діаграми вхідної секції згідно приведеної нижче таблиці істинності. Звертаємо увагу, що періоди часу між вертикальними лініями зміни станів рівні 5ns. Вхідні значення повинні зберігатись не менше 15ns. Необхідний час 10ns перед поданням нових значень для обнулення суматора перед приєднанням нових входів.

X	Y	Cin	Sum	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

12. Тепер ми вже готові до запуску симуляціїю. Зберігаємо файл часових діаграм як “full_adder.vwf” і вибираємо ***Processing→Generate Functional Simulation Netlist***

13. Запускаємо симуляцію ***Processing→Start Simulation*** або вибором відповідної піктограми.
14. За повідомленням про успішне завершення в новому вікні відображаються результуючі часові діаграми. Перевірити, що результати відповідають приведеній вище таблиці істинності. **Spikes in the output** внаслідок того факту, що ми подали всі входи в один і то й же час для спрощення. Однак запам’ятайте, що **Cin** був поданий на другий напівсуматор в той же час, що були подані X та Y на перший напівсуматор. Тому другий напівсуматор генерує проміжний результат, перш ніж на нього поступає результат першого напівсуматора.

2.3. Синтез та дослідження 4-х розрядного суматора

1. Вибрати ***File→ New→VHDL file*** і надрукувати наступний код у вікні редактора:

```
library ieee;
use ieee.std_logic_1164.all;

entity adderfour is
port (Cin:in std_logic;
x:in std_logic_vector(3 downto 0);
y:in std_logic_vector(3 downto 0);
s:out std_logic_vector(3 downto 0);
Cout:out std_logic);
end adderfour;

architecture compo of adderfour is
signal c1,c2,c3:std_logic;
component full_adder
port (X,Y,Cin:in std_logic;
sum,Cout:out std_logic);
end component;
begin
stage0:full_adder port map (Cin,x(0),y(0),s(0),c1);
stage1:full_adder port map (c1, x(1),y(1),s(1),c2);
stage2:full_adder port map (c2,x(2),y(2),s(2),c3);
stage3:full_adder port map (c3,x(3),y(3),s(3),Cout);
end compo;
```

Зберігаємо файл, як "adderfour.vhd" і вибираємо створення нового проекту на основі цього файлу в тій же директорії, де були збережені файли half_adder.vhd і full_adder files. Ігноруємо попередження директорії, а потім вибаємо ***Assignments→Settings→Files*** і добавляємо обидва файли в проект.

2. Натисніть піктограму компіляції або виберіть ***Processing→Start Compilation***
3. Відслідковувати повідомлення в процесі виконання компіляції. Запам'ятайте найбільшу затримку розповсюдження від вхідного виводу до вихідного. (tpd= ?)
4. Вибрати ***File→New→Other Files→Vector Waveform Files***
5. Отримуємо дві секції активного вікна. За допомогою лівої клавіші миші в лівій секції відкрити стовпці **"Name"** і **"Value at".**
6. Натиснути праву клавішу миші і вибрати **"Insert Node or Bus…"** із контекстного меню за допомогою лівої клавіші миші.
7. Натиснути **"Node Finder"** в діалоговому вікні і вибрати опцію **"List".**
8. Всі лінії входів/виходів (I/O) проекту відображуються в лівій стороні. Вибрати їх всіх і натиснути на знак **">>"** інструментів вибору для пересилання всіх їх в праву частину. Вибрати **"OK"**, діалогове вікно закривається, повертаємось назад в редактор часових діаграм.
9. Всі лінії Вхід/Вихід видимі тепер в стовпці **"Name"**. Вибираємо вхідні лінії як вектори і записуємо десяткові значення без знаку для формування вхідної секції таблиці істинності, приведеної нижче. Звертаємо увагу, що періоди часу між вертикальними лініями зміни станів рівні 5ns. Вхідні значення повинні зберігатись не менше 20ns. Необхідний час 15ns перед поданням нових значень для обнулення суматора перед приєднанням нових входів.

X	Y	Cin	Sum	Cout
2	2	0	4	0
2	2	1	5	0
7	3	0	10	0
7	3	1	11	0
9	2	0	11	0
9	2	1	12	0
8	7	0	15	1
8	7	1	0	1

10. Тепер ми вже готові до виконання симуляції. Зберігаємо файл часових діаграм як **"adderfour.vwf"** і вибираємо ***Processing→Generate Functional Simulation Netlist***

11. Запускаємо симуляцію: ***Processing→Start Simulation*** або вибором піктограми.
12. За повідомленням про успішне завершення в новому вікні відображаються результуючі часові діаграми. Перевірити, що результати відповідають приведеній вище таблиці істинності. **Spikes in the output** внаслідок того факту, що є нерівномірний перенос суматора і перенос розповсюджується повільніше від першого каскаду до останнього. Тому можливий некоректний проміжний результат на виході, поки на завершиться передавання найбільш затриманого сигналу.

2.4. Синтез та дослідження 16 розрядного суматора

1. Вибрати ***File→ New Project Wizard***.
2. Присвоїти ім'я проекту **myadder16**.
3. Вибираємо ***Tools → Megawizard plugin Manager***, Вибрати першу опцію **"Create a new custom megafunction variation".**
4. Із арифметичної категорії вибрати **LPM_ADD_SUB**. Можна проігнорувати вибір сімейства ПЛІС.Однако вибираємо генерування VHDL файлу і задаємо ім'я вихідного файлу як **addonly.vhd**.
5. В кількох наступних вікнах вибрати модуль «лише додавання», встановити розрядність шини даних 16, вибрати обидва входи як змінні (variable), добавити приєднання вхідного та вихідного переносів і відмовитись від **pipeline** функції.
6. Відкрити файл **addonly.vhd**, який згенерувався сервісом Мегамайстра. Переконайтеь, що ім'я проекту коректне і всі входи та виходи коректної розрядності.
7. Надрукуйте код, приведений нижче в VHDL файл з іменем **myadde16.vhd**. Він буде вершиною проекту, як уже відзначалось, коли ви створювали проект. Переоконайтесь, що цей файл є частиною проекту.

```
library ieee;
use ieee.std_logic_1164.all;

entity myadder16 is
port (carryin:in std_logic;
x,y:in std_logic_vector(15 downto 0);
s:out std_logic_vector(15 downto 0);
```

```
carryout:out std_logic);
end myadder16;

architecture struct of myadder16 is
component addonly
 PORT
 (
        dataa  : IN STD_LOGIC_VECTOR (15 DOWNTO 0);
        datab  : IN STD_LOGIC_VECTOR (15 DOWNTO 0);
        cin            : IN STD_LOGIC ;
        result  : OUT STD_LOGIC_VECTOR (15 DOWNTO 0);
        cout           : OUT STD_LOGIC
 );
 END component;

 begin
 adder_circuit: addonly port map (cin=>carryin, dataa=>x, datab=>y,
 result=>s, cout=>carryout);
 end struct;
```

8. Зкомпілюйте проект та виконайте його симуляцію редагуванням часових діаграм. Переконайтесь, що вибрані позначення ліній входів і виходів дійсно відповідають індивідуальним лініям. Відкрийте меню симуляції і виберіть функціональну симуляцію. Запишіть входи як: **FF00 + 00FF, 100A+ 0001, F0FF + 00FF** і переконайтесь, що виходи рівні **FFFF**, **100B, F1FE**.

4. Контрольні питання.

1. Що таке мова опису апаратури?
2. Назвіть мови опису апаратури, які існують. Чим вони відрізняються?
3. Назвіть основні елементи мови VHDL, дайте їх коротку характеристику.
4. Як описують логічні елементи в VHDL?

5. Рекомендована література.