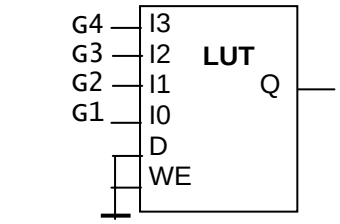
Лабораторна робота 3

Проектування АЛП на базі логічних таблиць

1 **Мета лабораторної роботи**:
оволодіти знаннями і навичками по проектуванню арифметико-логічних пристроїв (АЛП) для сучасних компьютерів. Ознайомитись з основами розробки проектів для програмованих логічних інтегральних схем (ПЛІС). Вивчити спосіб проектування логічних схем на базі логічних таблиць.

Теоретичні відомості

Роль основного логічного елемента в ПЛІС грає логічна таблицця (ЛТ) або look-up table (LUT), яка представляє собою однобітний ОЗП на 16 комірок (рис.1). В ЛТ за адресою G3,G2,G1,G0 записана одиниця, якщо код адреси представляє собою констітуенту одиниці заданої чотирьохвходової логічної функції. Наприклад, якщо за адресою 1,1,1,1 записана одиниця, а за рештою адрес - ноль, то ЛТ реалізує чотирьохвходову функцію І. На рис.1 показано приклад кодування функції Виключне АБО на чотири входа.



Q:=G4 xor G3 xor G2 xor G1

G4	G3	G2	G1	Q
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Рис.1. Логічна таблиця для виконання чотирьохвходової функції Виключне АБО

Тригери ЛТ входять в склад программуючого регістра зсуву ПЛІС і їхній початковий стан заповнюється в період конфігурування ПЛІС.

Під час перетворення проекту, представленого на рівні вентилів в проект на ПЛІС, його логічні схеми покриваються еквівалентними схемами на основі ЛТ. Такий процес називається відображенням (mapping).

2. Завдання для лабораторної роботи:

– розробити функціональну схему одного і-го розряду АЛП, що виконує задані функції;

– змоделювати роботу АЛП.

Результати виконання оформлюються у вигляді звіту (протоколу). Звіт повинен вміщувати:

- опис і рисунок заданого варіанта АЛП,
- хід проектування і схему АЛП,
- графіки сигналів, знятих при іспитах АЛП,
- висновки.

У всіх варіантах завдань необхідно обчислити деяку функцію Y=F(A,B).

Варіант завдання вибирається за номером студента в списку групи з наступної таблиці

№ завдання	Функції Y в залежності від F	№ завдання	Функції Y в залежності від F
1	A+B+C_0, A and B	12	A-B-C_0, A nand B
2	A+B+C_0, A or B	13	A-B-C_0, A nor B
3	A+B+C_0, A xor B	14	A-B-C_0 , A xor B
4	A+B+C_0, A xnor B	15	A-B-C_0, A xnor B
5	A+B+C_0, max(A,B)	16	A-B-C_0, max(A,B)
6	A+B+C_0, min(A,B)	17	A-B-C_0, min(A,B)
7	A+B+C_0, A-B-C_0	18	A-B-C_0, abs(A)
8	A+B+C_0, not A and B	19	A-B-C_0, not A and B
9	A+B+C_0, not A or B	20	A-B-C_0, not A or B
10	A+B+C_0, A xor not B	21	A-B-C_0, A xor not B
11	A+B+C_0, abs(A)	22	A-B-C_0, abs(B)

3. **Виконання роботи**

Спочатку складають таблиці істиності вихідних сигналів АЛП. Для кожного розряду АЛП складають окрему таблицю. Це необхідно для того, щоб спробувати виявити закономірність в побудові логічної схеми одного розряду в багаторозрядному АЛП.

На першому етапі синтезу за таблицями істиності записують Булевські рівняння для вихідних сигналів. Булевські рівняння використовують для накреслення функціональної схеми.

На другому етапі функціональну схему покривають схемою, що складається з 2, 3, 4-входових ЛТ і виконують функціональну схему, що складається тільки з ЛТ. Метою етапу є мінімізувати кількість ЛТ, що відповідає мінімізації апаратних витрат.

Моделі ЛТ беруть з пакета Unisim фірми Xilinx, що входить в склад САПР ActiveHDL. Це такі 2, 3, 4-вході моделі, як LUT2, LUT3 і LUT4, відповідно. Об′ява компонента ЛТ LUT4 виглядає як:

```
component LUT4  generic(  INIT : bit_vector := X"0000"  );
  port(O : out std_ulogic;
     I0 : in std_ulogic;
     I1 : in std_ulogic;
     I2 : in std_ulogic;
     I3 : in std_ulogic );
end    component;
```

Вставка компонента, що відповідає логічній функції `y = a and b and c and d` – “4-входова І” виглядає як:

```
U1:LUT4 generic map(X"8000")
 port map(y,     a,b,c,d);
```

Тут настроювальна константа `X"8000"` дорівнює 16-бітному вектору – вмісту ЛТ в 16-ричному вигляді, причому в 15-му розряді (за адресою 15) записана одиниця, а в решті розрядів (за рештою адрес) - ноль, тобто ЛТ дійсно реалізує чотирьохвходову функцію І.

На третьому етапі знаходять настроювальні константи для всіх ЛТ у відповідності з їхніми функціями і складають відповідну VHDL-програму.

Допускається пошук функціональної схеми з ЛТ і їхнього вмісту безпосередньо з таблиць істинності вихідних бітів АЛП.

Результуючу VHDL-програму тестують в САПР ActiveHDL.

Одержані графіки сигналів заносять в протокол лабораторної роботи.

4 **Приклад виконання роботи**

Розглянемо приклад проектування і- го розряду АЛП, що виконує функції

Y = max(A , B) при F=0 , Y = min(A, B) при F=1.

При цьому n-бітний АЛП повинен складатись з n таких розрядів. Функціонування АЛП полягає в тому, що спочатку з А віднімається В і потім аналізується біт переносу Cn з старшого розряду. Потім якщо Cn = 1, то A≥B і на вихід АЛП видається А при F=0, а інакше – В. При F=1 на вихід видаються при такій самій умові В і А відповідно.

Структура АЛП показана на рис. 2. Вона складається з n ступенів, кожна з яких вміщує 2 логічні таблиці LUT2 і LUT3.

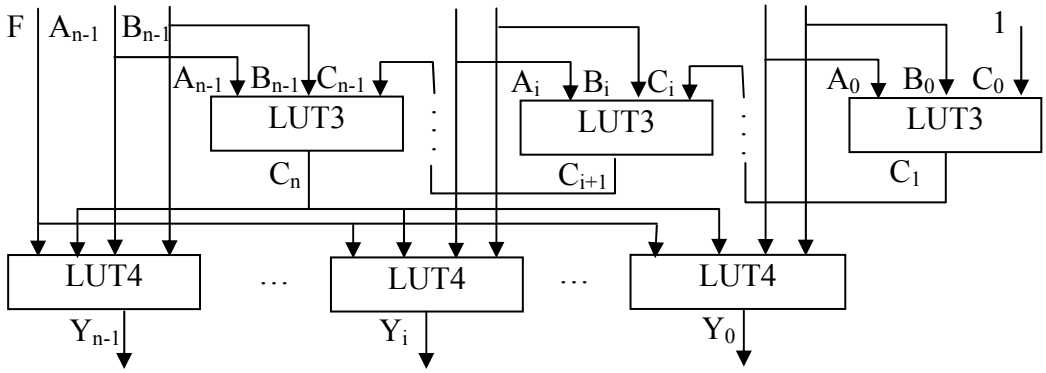


Рис.2. Структура АЛП

Логіка ЛТ виражається наступними таблицями істинності

Ai	Bi	Ci	C_{i+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

F	Cn	Ai	Bi	Yi
0	0	0	0	0
		0	1	0
		1	0	1
		1	1	1
	1	0	0	0
		0	1	1
		1	0	0
		1	1	1
1	0	0	0	0
		0	1	1
		1	0	0
		1	1	1
	1	0	0	0
		0	1	0
		1	0	1
		1	1	1

Перша і друга таблиці виражають логіку першої і другої ЛТ і-го розряду АЛП, відповідно. Відповідні настроювальні константи дорівнюють вмісту правих столбців таблиць (читаючи знизу-вгору):

“10110010” = X”B2” і “1100101010101100” = “CAAC” .

Об'єкт проекта виглядає наступним чином

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
library Unisim;
use unisim.all;
entity ALUi is
 port( a  : in STD_LOGIC;
       b  : in STD_LOGIC;
       ci : in STD_LOGIC;  --перенос з попереднього розряду
       cn : in STD_LOGIC;  --перенос з найстаршого розряду
       f  : in STD_LOGIC;
       co :out STD_LOGIC;
       y  :out STD_LOGIC  );
end ALUi;

architecture ALUi of ALUi is
 component LUT4 is generic( INIT : bit_vector := X"0000"  );
      port( O : out std_ulogic;
            I0 : in std_ulogic;
            I1 : in std_ulogic;
            I2 : in std_ulogic;
            I3 : in std_ulogic );
 end  component;
 component  LUT3 is generic( INIT : bit_vector := X"00" );
      port( O : out std_ulogic;
            I0 : in std_ulogic;
            I1 : in std_ulogic;
            I2 : in std_ulogic );
 end   component;

begin
 U1:LUT3 generic map(X"B2")
 port map(O=>co,I2=>a,I1=>b,I0=>ci);

 U2:LUT4 generic map(X"CAAC")
 port map(O=>y,I3=>f,I2=>cn,I1=>a,I0=>b);
end ALUi;
```