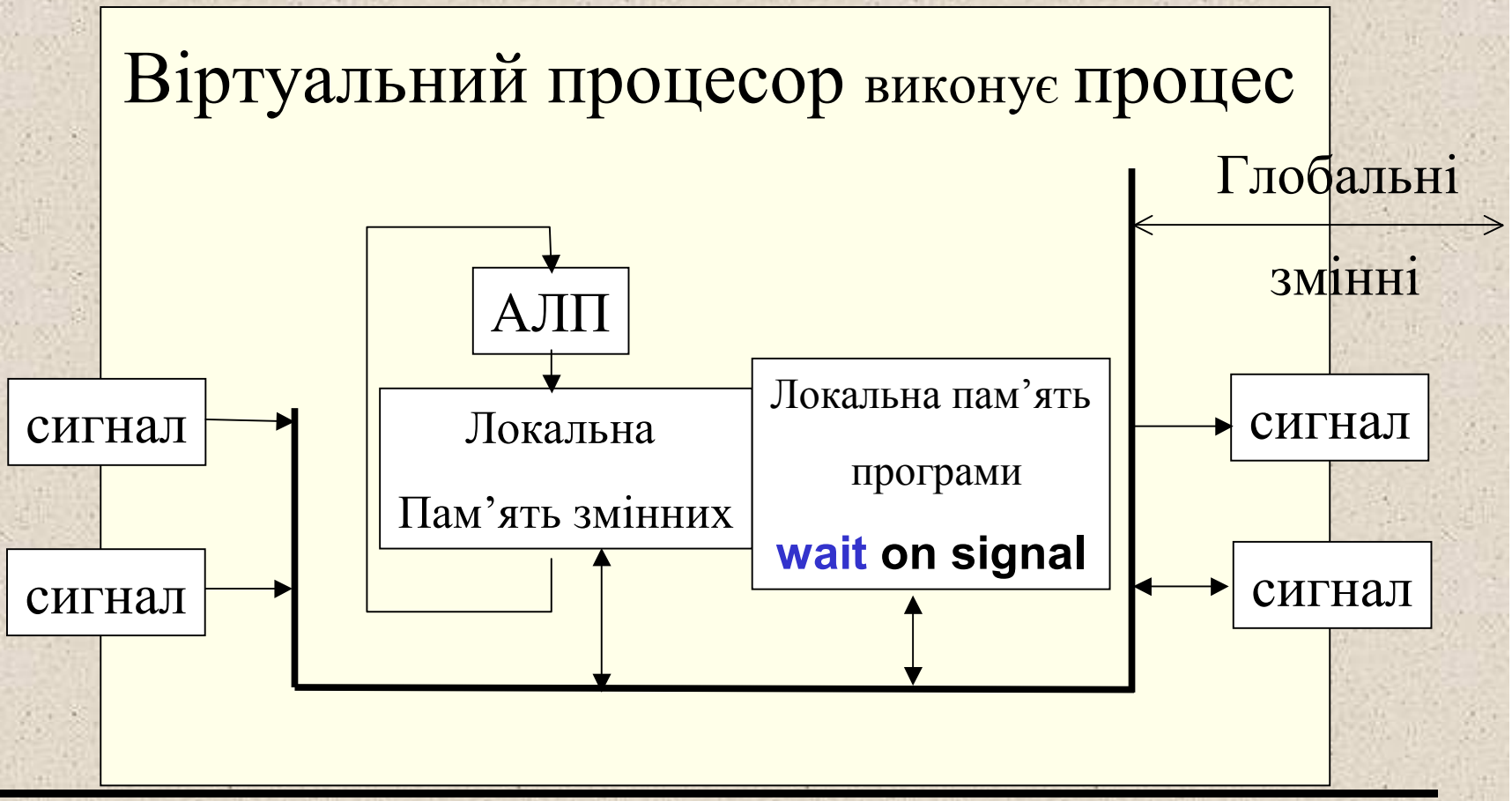
Бібліотека IEEE для проектування пристроїв

Процес і послідовні оператори

- Оператор **process** визначає незалежний послідовний процес , який представляє собою поведінку деякої частини проекта.



Оператор присвоювання змінній

- Змінні зберігають значення таке саме як і сигнал, але можуть бути використані тільки всередині ПРОЦЕСА. Вони не можуть передавати дані між процесами
- На відміну від сигналу, значення, яке присвоєне змінній, доступне негайно

```
process (A,B,C)
  variable c,d: bit:='0';
begin
    d := B or c;       -- d обчислюється і присвоюється,
  -- значення змінної c – з попереднього запуску процеса,
              --  змінна c   не має відношення до сигналу C
    c := d and B;     -- d приймає участь в обчисленні
    A<= c xor B;      -- нове значення c приймає участь
end process;          -- в цей момент присвоюється A
```

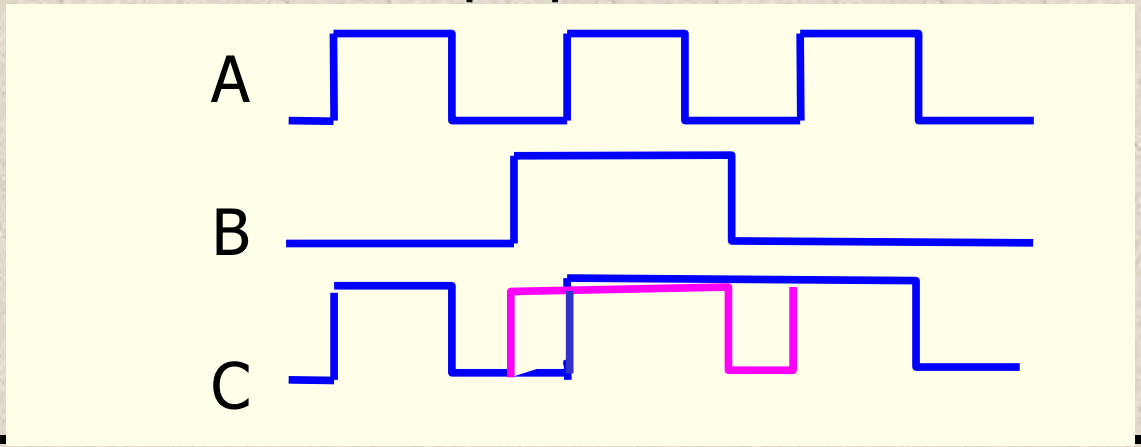
Список чутливості

При моделюванні логічних схем в список чутливості необхідно вносити **всі вхідні сигнали**, інакше моделювання може відрізнятись від бажаного.

Наприклад, процес

```
process(A, B )  begin
    c<= A or B;
  end process;
```

При моделюванні дає графіки:

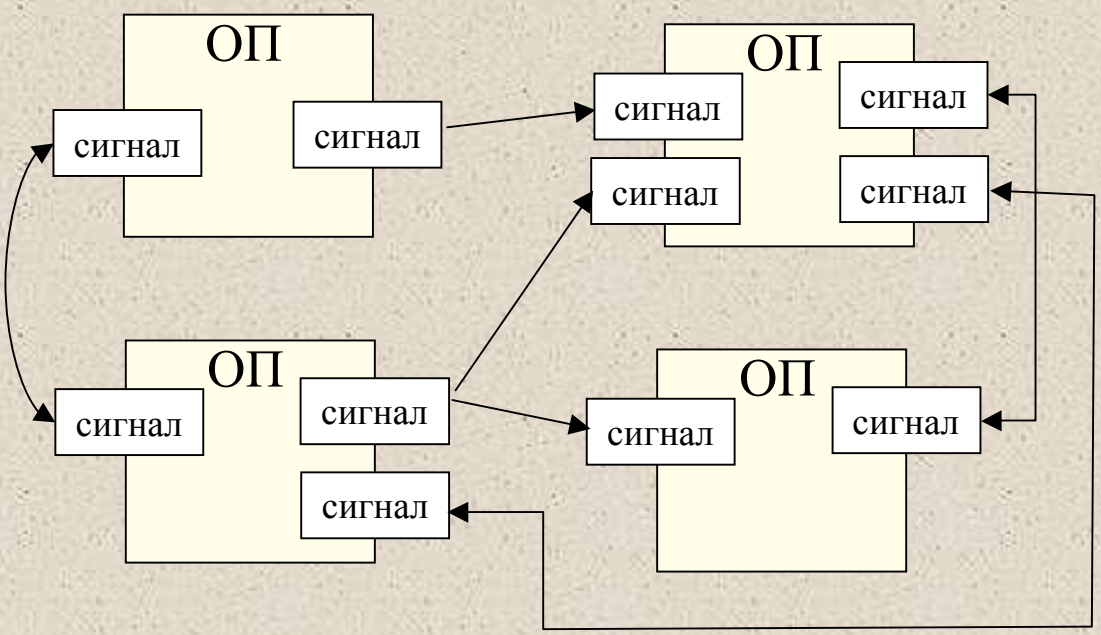


Обчислювальна модель для реалізації VHDL

Консоль

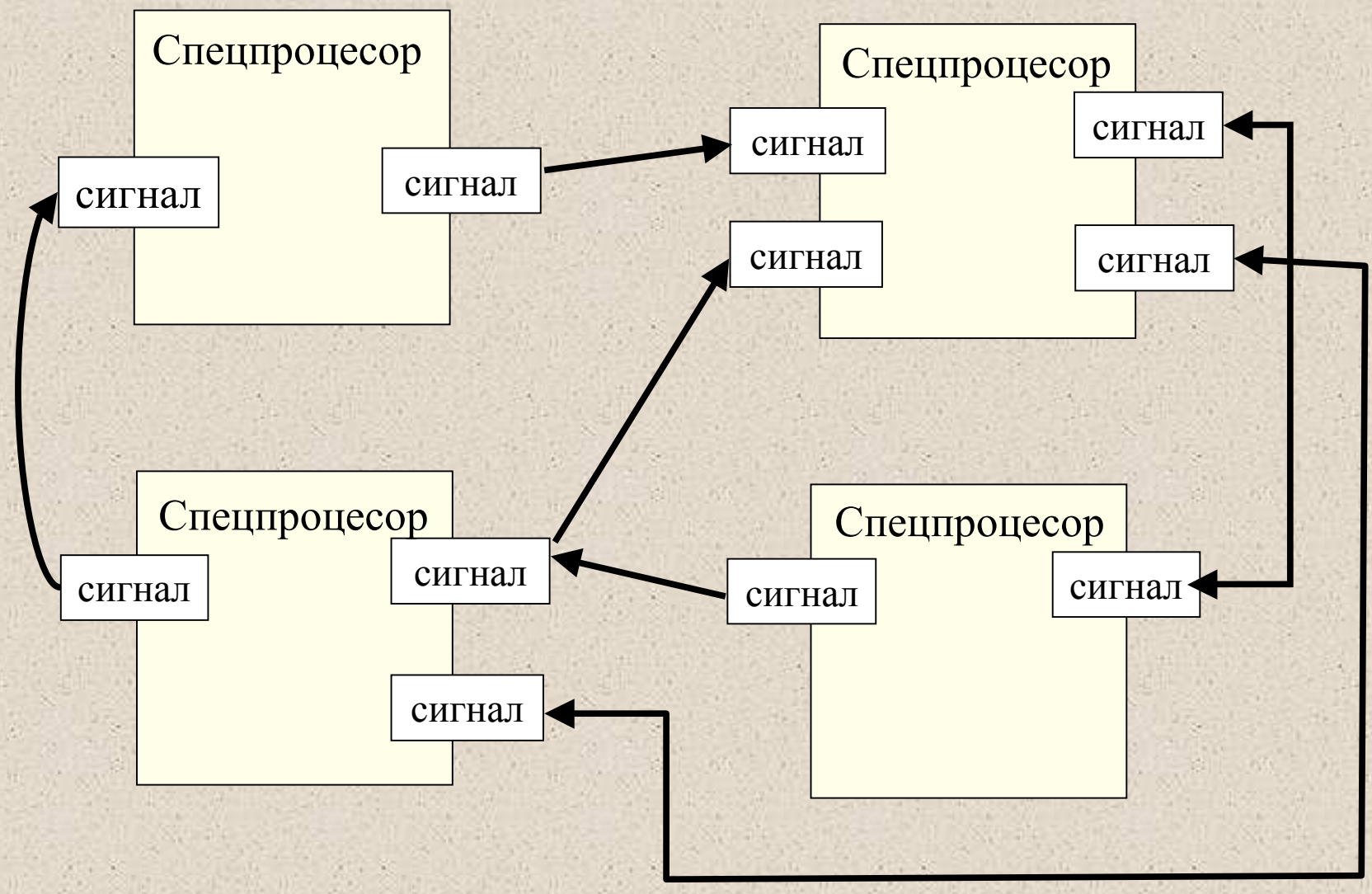
Дискова пам’ять

Пам’ять глобальних змінних



Аппаратна модель для реалізації VHDL

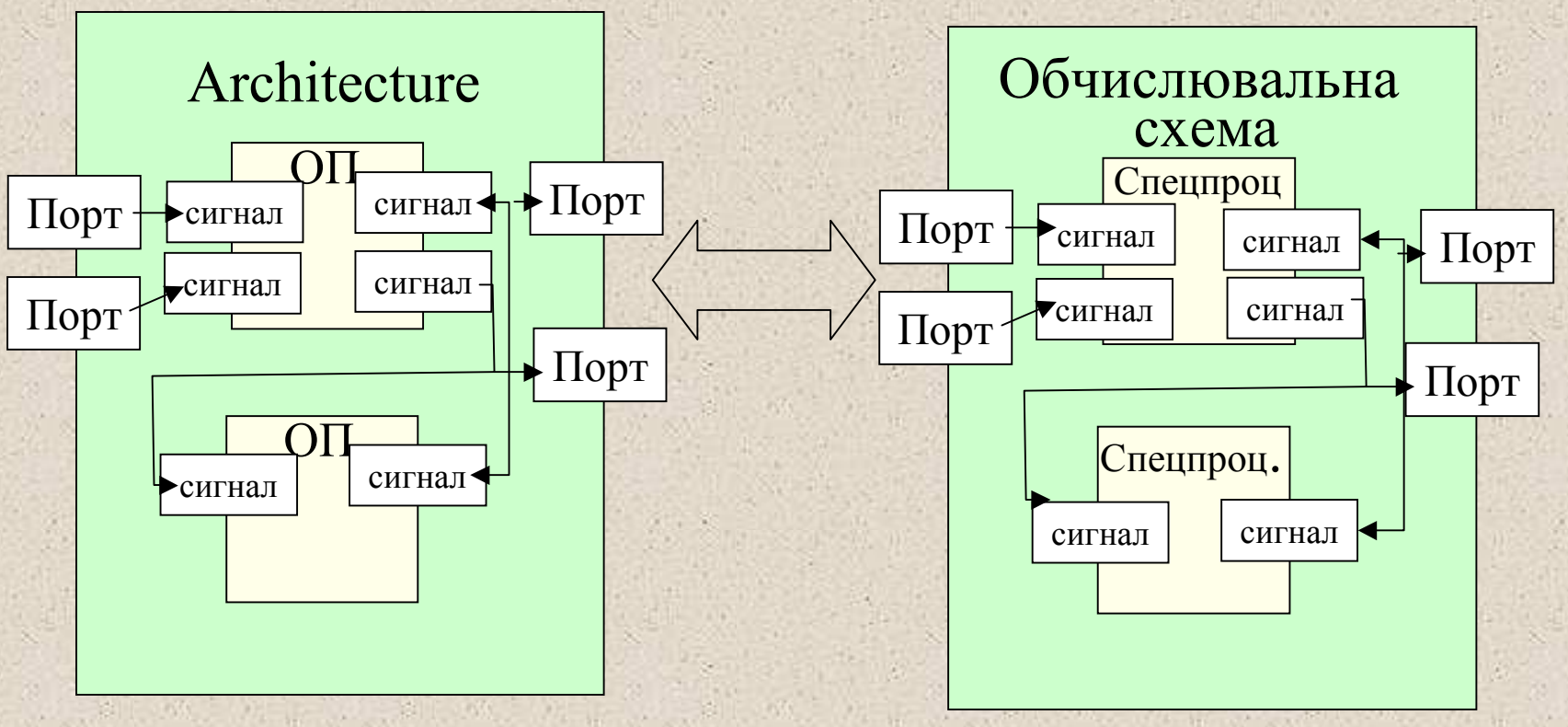
Аппаратна модель для реалізації VHDL



Відображення алгоритмів на VHDL

Програмна модель

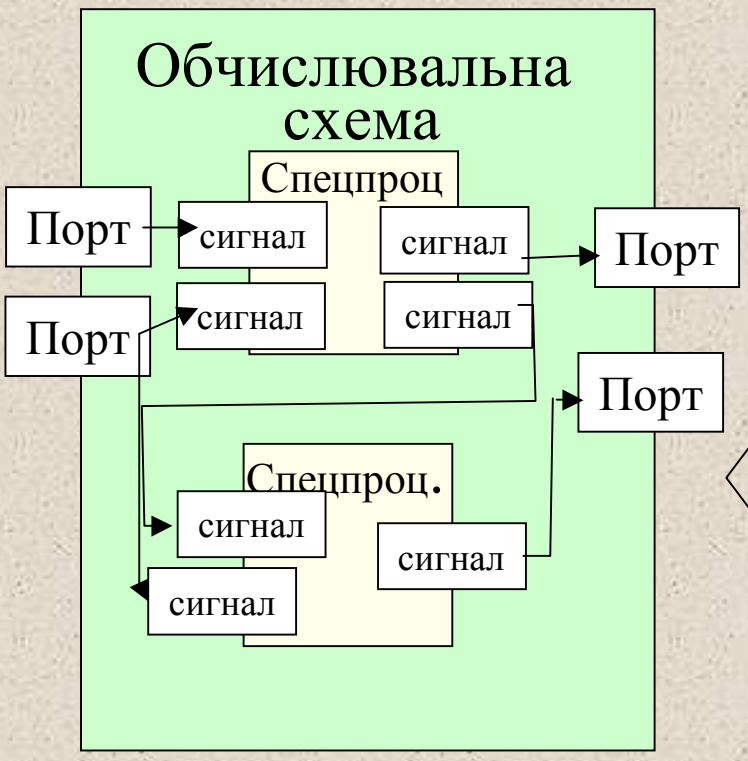
Апаратна модель



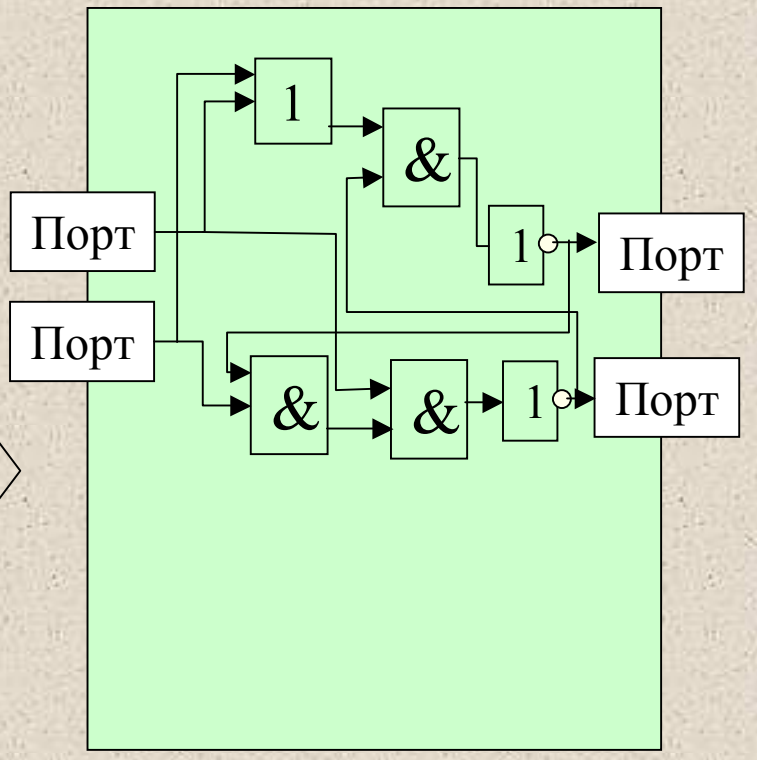
Синтез логічної схеми

Відображення в вентилі

Апаратна модель



Логічна схема

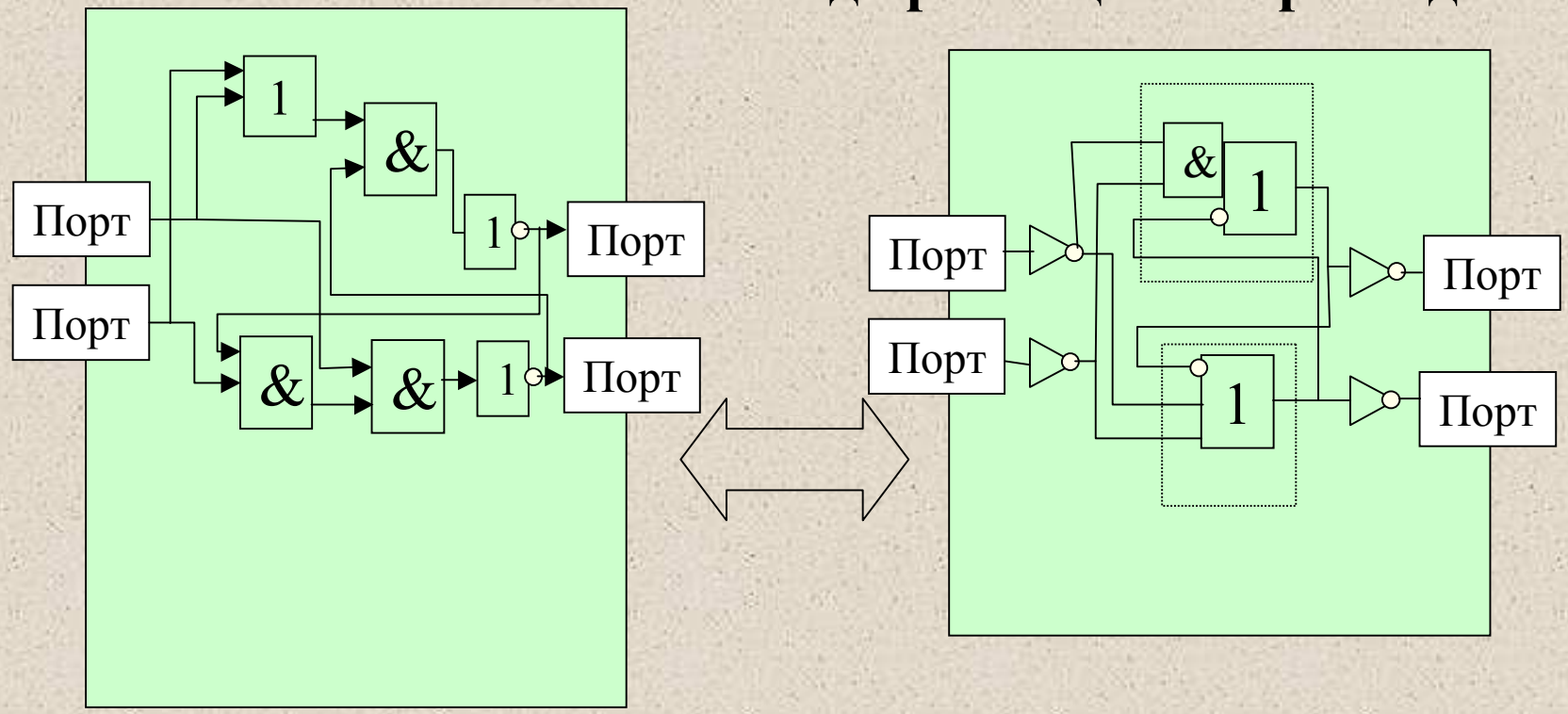


Синтез логічної схеми

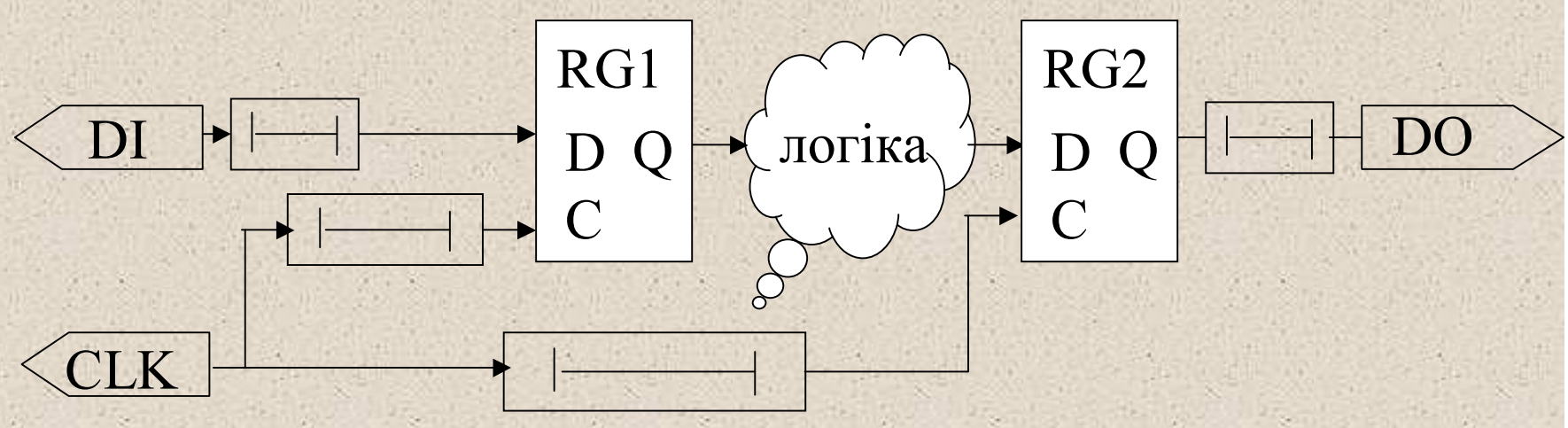
Оптимізація і адаптація до елементної бази (технології)

Логічна схема

Схема, яка готова до розміщення і розводки



Загальні вимоги до проектування систем на кристалі.

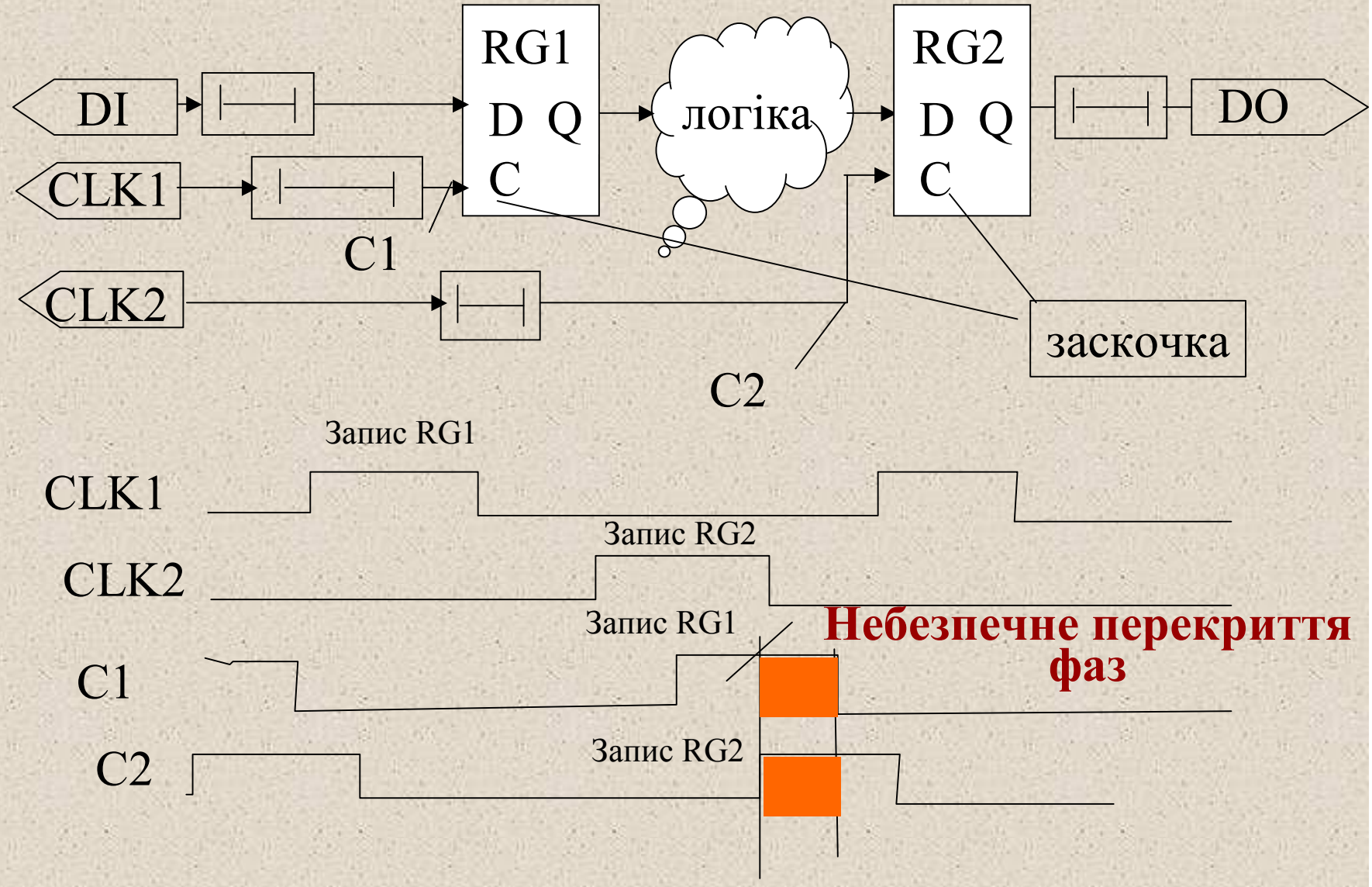


Затримки у лініях з’єднання можуть бути більшими за затримки в логічних схемах і тригерах.

Поведінка схеми у кристалі

має бути передбачуваною

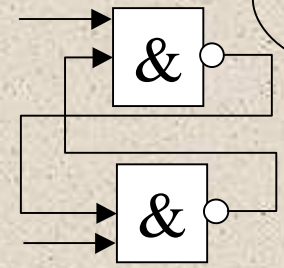
Принцип двохтактної синхронізації.



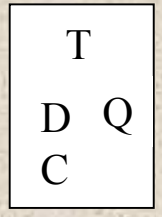
Проектування з асинхронними тригерами

Ненавмисна заскочка

Спроектована заскочка

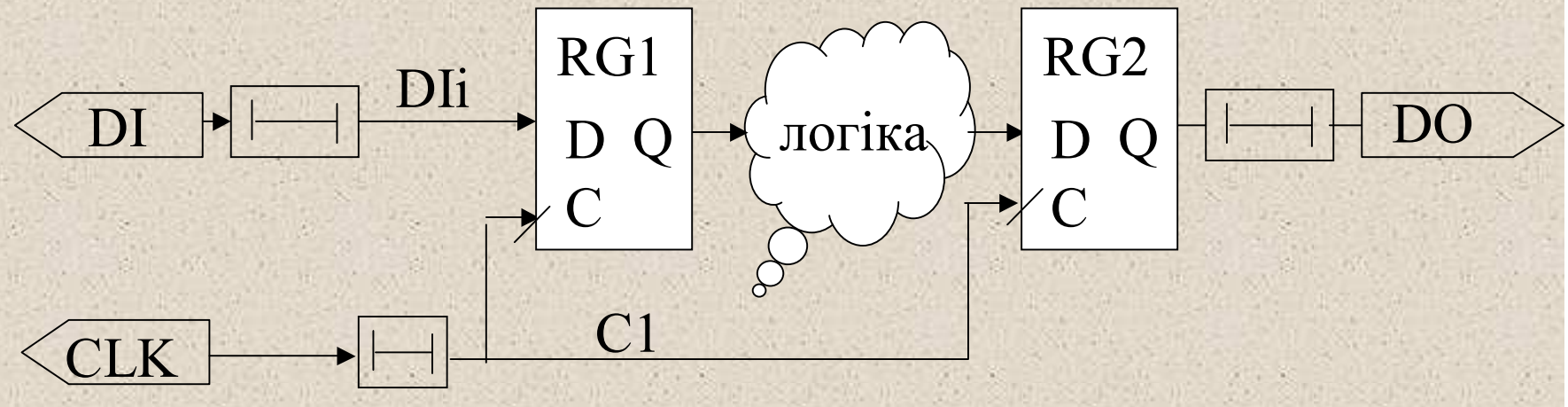


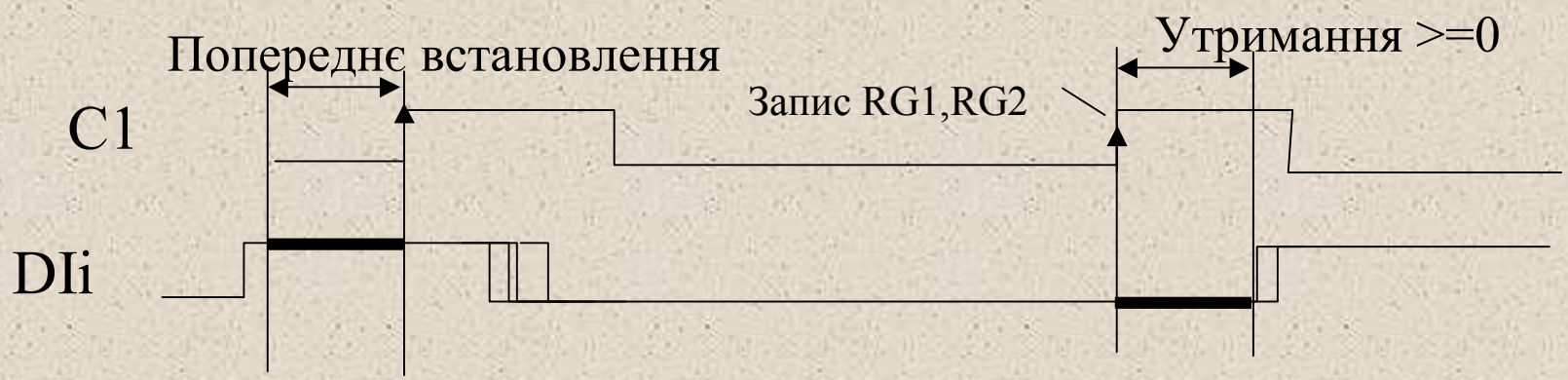
Заскочка з бібліотеки



- Без особливої потреби
  не треба проектувати з асинхронними тригерами
- За потребою слід використати заскочку з бібліотеки і забезпечити відсутність перекриття фаз синхронізації

Принцип однотактної синхронізації





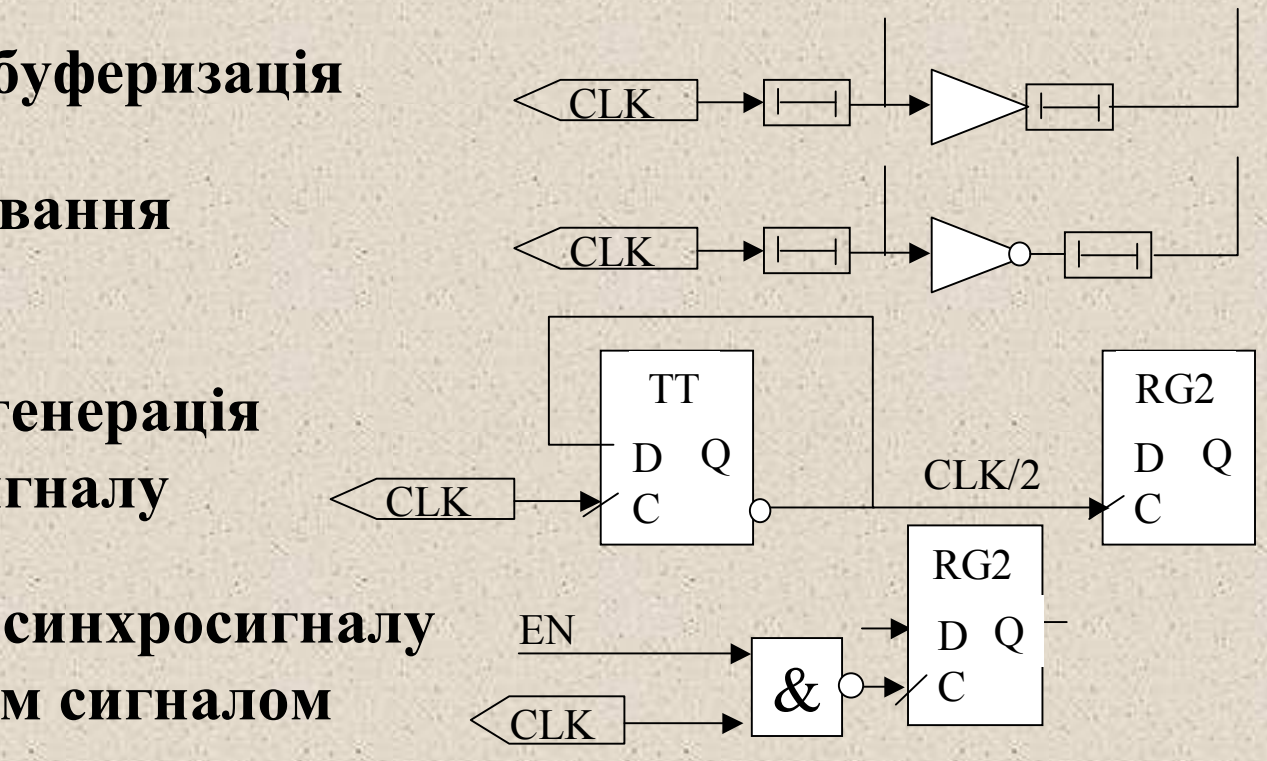
Перекручування однотактної синхронізації

Додаткова буферизація

Інвертування

Внутрішня генерація синхросигналу

Змішування синхросигналу з логічним сигналом



- Слід утримуватись від перекручування синхронізації
- При необхідності використовуйте глобальні буфери і мережу синхронізації
- Бажано звести кількість синхросерій до 1.

5. Конструкції VHDL для синтезу.

Конструкція підтримується	Конструкція не підтримується
Пакет	
IEEE.Std_Logic_1164	IEEE.TextIO
IEEE.Std_Logic_Unsigned	Пакети користувача (з конструкціями, що не підтримуються і конструкціями типу: - глобальні змінні - відкладені константи - фізичні типи - багатовимірні масиви (не завжди)
IEEE.Std_Logic_Signed	
IEEE.Std_Logic_Arith	
IEEE.Numeric_Bit (не завжди)	
IEEE.Numeric_Std(не завжди)	
Пакети користувача (з дозволеними конструкціями)	

Конструкції VHDL для синтезу.

Конструкція підтримується | **Конструкція не підтримується**

Класи об’єктів

константи

сигнали

змінні

файли

Сигнал, змінна, generic

підтримуються;

ініціалізація generic;

```
generic(a:integer:=0)
```

ініціалізація сигнала або змінної;

```
signal a: bit:='0';
(не завжди)
```

тип сигналу Register, Bus;

константа, ініційована в формальному параметрі або у зв”язуванні порту

```
port map(a=>'1'); (не завжди)
MY_FUNC(b:=to_integer("0100"));
```

Конструкції VHDL для синтезу.

Конструкція підтримується	Конструкція не підтримується
Типи та підтипи	
Integer	фізичні типи,
перелічувані	Time,
Boolean	Real
Bit, Bit_Vector	багатовимірні масиви
Std_Logic, Std_Logic_Vector	
одномірний масив	
Record – з обмеженнями	Record

Конструкції VHDL для синтезу.

Конструкція підтримується | **Конструкція не підтримується**

Оператори

логічні and, or, xor, not
порівняння =,/=,<,>
конкатенація &
арифметичні +, -, *, abs
- якщо задіяно відпов. пакет
/, **, mod,rem - для степеня 2

арифметичні /, **, mod,rem - для довільних операндів

Присвоювання

сигналу, змінній,
умовне сигналу,
вибіркове сигналу,
присвоювання агрегату

```
(a,b,c)<=data(1 to 3);
```

опис AFTER ігнорується

Конструкції VHDL для синтезу.

Конструкція підтримується	Конструкція не підтримується
Підпрограми	
Підтимуються, якщо об”явлені в пакетах або у декларації архітектури	З операторами WAIT, функція NOW функція вирішення не з бібліотеки IEEE
Атрибути	
'base, 'left, 'right, 'high, 'low, 'range, 'reverse_range, 'length 'event, 'stable - в операторах if, wait. Атрибути, які визначені фірмою –виробником ПЛІС	решта стандартних атрибутів, атрибути користувача

Конструкції VHDL для синтезу.

Конструкція підтримується	Конструкція не підтримується
Послідовні оператори	
Wait, присвоювання сигналу, присвоювання змінній, CALL, return, if, case, loop, next, exit, null	опис AFTER, transport, оператор assert ігноруються, індекс для loop - не цілий

Конструкції VHDL для синтезу.

Конструкція підтримується	Конструкція не підтримується
Паралельні оператори	
process	process - список чутливості ігнорується
Присвоювання сигналу	guarded, transport - ігноруються
процедура	процедура з кількома WAIT
вставка компонента	вставка компонента з перетвореннями типів, агрегатами, константами у зв’язуванні портів
generate	guarded, register block
block	block з port та generic
	CONFIGURATION

Пакет std_logic_1164

Базовий тип даних для моделювання ситуацій в лініях зв'язку і логічних схемах (STD_LOGIC) :

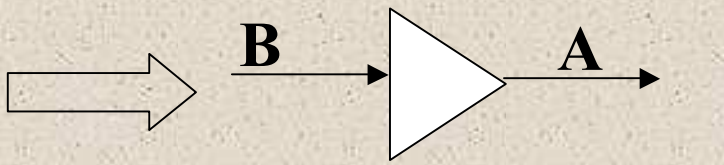
'**U**' - не ініціалізований
'**X**' - сильний невідомий
'**W**'- слабий невідомий
'**Z**' - високий імпеданс
'**H**' - слабий 1
'**L**' - слабий 0
'**-**' - байдуже

елементи 'U', 'X', 'W' и '-' кодують не логічні стани а стани сигналу в процесі моделювання, що відрізняються від правильного.

Кілька джерел сигналу і функція вирішення.

Кожне присвоювння сигналу відображається в джерело сигналу

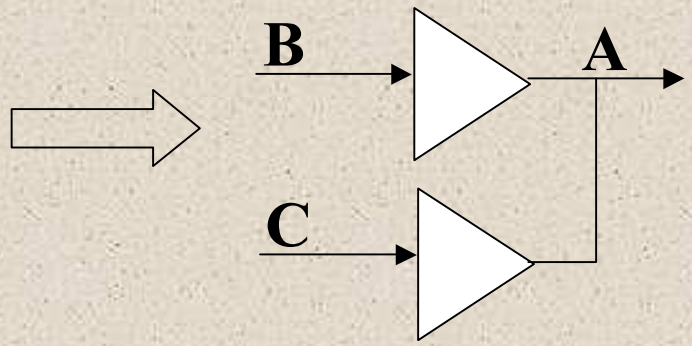
A<=B;



Багатократне присвоювання

A<=B;

A<=C;



дозволене тільки для сигналів, над типом яких задана

функція вирішення.

Кілька джерел сигналу і функція вирішення.

Std_ULogic - тип без функції вирішення;

Std_Logic - підтип типу Std_ULogic з функцією вирішення :

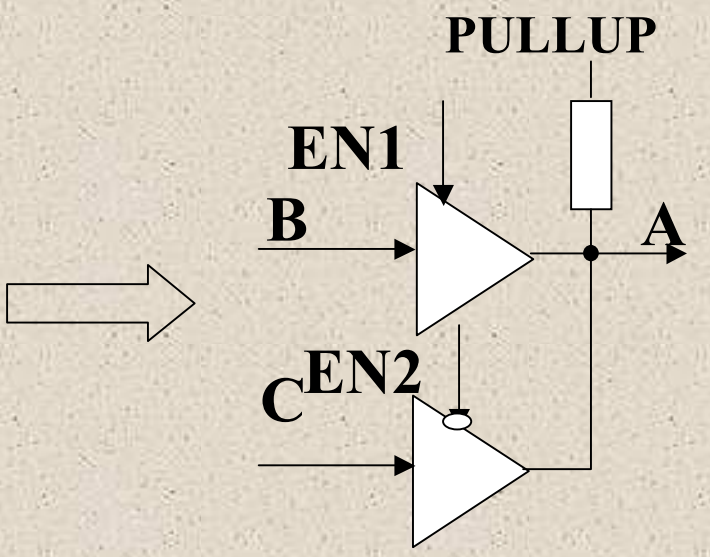
```
SUBTYPE std_logic IS resolved std_ulogic;
FUNCTION resolved (s:std_ulogic_vector) RETURN std_ulogic;

CONSTANT resolution_table : stdlogic_table := (
   ---------------------------------------------------
   --|  U    X    0    1    Z    W    L    H    -     |   |
   ---------------------------------------------------
     ( 'U', 'U', 'U', 'U', 'U', 'U', 'U', 'U', 'U' ), -- | U |
     ( 'U', 'X', 'X', 'X', 'X', 'X', 'X', 'X', 'X' ), -- | X |
     ( 'U', 'X', '0', 'X','0', '0', '0', '0', 'X' ), -- | 0 |
     ( 'U', 'X', 'X', '1', '1', '1', '1', '1', 'X' ), -- | 1 |
     ( 'U', 'X', '0', '1', 'Z', 'W', 'L', 'H', 'X' ), -- | Z |
     ( 'U', 'X', '0', '1', 'W', 'W', 'W', 'W', 'X' ), -- | W |
     ( 'U', 'X', '0', '1', 'L', 'W', 'L', 'W', 'X' ), -- | L |
     ( 'U', 'X', '0', '1', 'H', 'W', 'W', 'H', 'X' ), -- | H |
     ( 'U', 'X', 'X', 'X', 'X', 'X', 'X', 'X', 'X' )  -- | - |
   );
```

Кілька джерел сигналу і функція вирішення

Для синтезу дозволяються кілька джерел сигналу,
якщо ці джерела керуються таким чином,
щоб функція вирішення при будь-яких вхідних даних
не давала як результат значення X або W,
напр., коли одночасно сигналу присвоюється
значення 0 і 1 , L і H.

```
 Signal A: Std_Logic;
. . .
A<=B when EN1='1' else 'Z';
A<=C when EN2='0' else 'Z';
UP : PULLUP port map(O=>A);
. . .
Architecture synt of PULLUP is
begin
          O<='H';
End;
```



Функції пакета **std_logic_1164**

В пакеті визначені логічні функції, що перезавантажуються, такі як

and, or, not, nand, nor, xor, xnor.

Логіка цих функцій у відношенні до елементів типу std_ulogic аналогічна логіці функції вирішення, тобто при помилкових сигналах – операндах результат буде 'U' або 'X'.

Наприклад:

'X' or '1' = '1', 'U' and '0' = '0',

'-' nand '0' = '1'.

Функції пакета **std_logic_1164**

функції перетворення типів:

```
function To_bit ( s : std_ulogic; xmap : bit := '0') return bit;

function To_bitvector(s:std_logic_vector;xmap: bit := '0') return
    bit_vector;
constant EL:std_logic_vector(0 to 8):="01HLZWXU-";
To_bitvector(EL,'1') ="0110111111", To_bitvector(EL) ="011000000".

function To_StdULogic( b : bit ) return std_ulogic;

function To_StdLogicVector(b:bit_vector) return
    std_logic_vector;
```

Функції пакету **std_logic_1164**

Для моделювання синхронних схем пам'яті булевські функції Rising_edge і Falling_edge фронту і спаду сигналу,

Наприклад, оператори

C<=Rising_edge(CLK);

C<= CLK='1' and CLK'event;

Виконують однакову дію - присвоюють **C true** , коли в циклі моделювання відбувся фронт сигналу **CLK**

Пакети std_logic_arith, std_logic_signed і std_logic_unsigned.

Визначені типи unsigned і signed як вектори елементів типу std_logic

unsigned - позитивні двійкові цілі числа

signed - двійкові цілі числа зі знаком в доповнюючому коді.

Функції '+', '-', '*', abs,

Всі функції порівняння для операндів типу unsigned, signed, small_int та integer

Вектори операндів у виразах можуть мати різні діапазони

Пакети std_logic_arith, std_logic_signed і std_logic_unsigned.

constant A:signed(4 **downto** 0) :="11100"; --*число -4*

constant B:unsigned(3 **downto** 1):="101"; --*число 5*

A+7 = "00011" , A+B = "00001", A-B = "10111",

B*3 = "01111", (A = B) = false, (A<B) = true.

Функції зсуву

SHR(A,B-4) = "11110", --*вправо*

SHL(A,B-3) = "10000". *–вліво*

Функції розширення розрядної сітки

EXT("10",5) = "00010", -- *без знака*

SXT("10",5) = "11110". --*зі знаком*

Пакети std_logic_arith, std_logic_signed і std_logic_unsigned.

Перетворення типів

Функції:

Conv_std_logic_vector

Conv_integer, Conv_unsigned Conv_signed

```
constant A:signed(4 downto 0) := Conv_signed (-4)  ;
                                          --число -4
constant B:unsigned(3 downto 1):= Conv_unsigned (5);
                                          --число 5
constant C:std_logic_vector(4 downto 0) :=
          Conv_std_logic_vector(-4, 5)  ;   --число -4
```

Пакети numeric_bit і numeric_std

numeric_bit - базовий тип bit,

numeric_std – базовий тип std_ulogic.

Функції numeric_std перезавантажують функції з пакета std_logic_arith, а також арифметичні, логічні функції и функції зсуву з бібліотеки STD, включаючи "/", **mod**, **rem**

Особливі функції:

Std_match("10101X", "1-1-1-") = true.

To_01, To_integer,

To_signed, To_unsigned, To_stdlogicvector.