Digital Design Practical Exercise 2
Combinational logic design & testbench

Yêu cầu báo cáo:

Trong báo cáo nên có các phần sau:

- Source code VHDL của các phần thực hành (những phần phải viết code)
- Kết quả chạy mô phỏng, giải thích kết quả (dưới dạng dễ hiểu và dễ thấy)
- Trả lời các câu hỏi trong phần hướng dẫn
- Liên kết đến các phần lý thuyết đã học và những kiến thức mà các bạn thấy

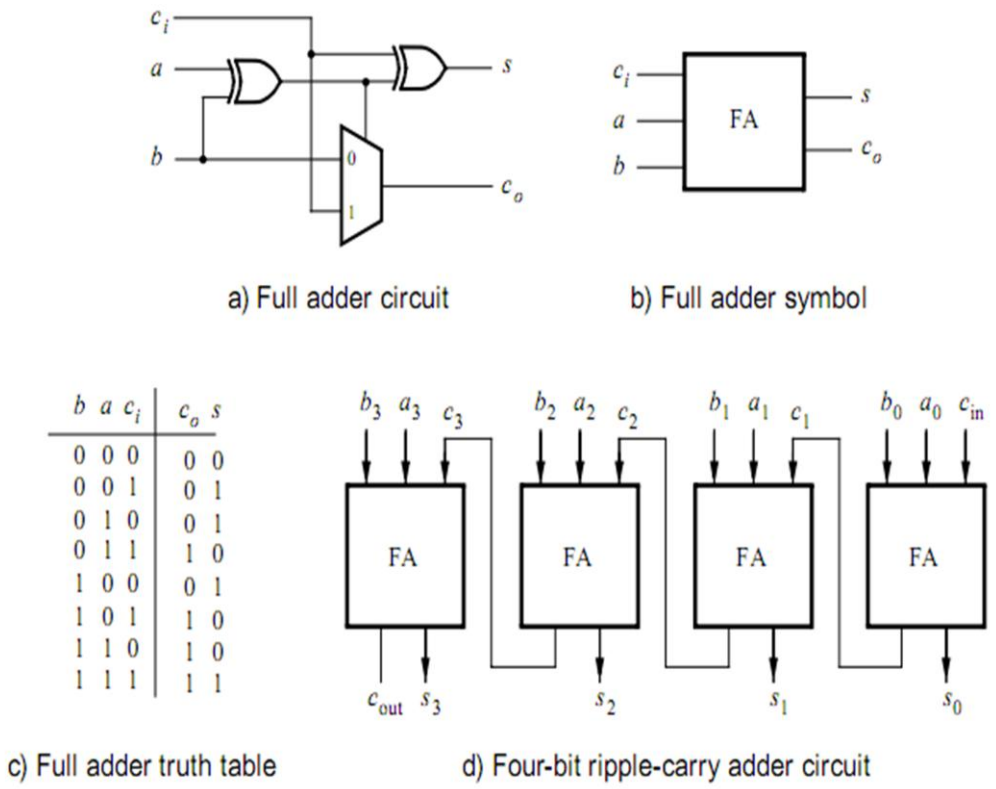
Chuẩn bị

Tạo thư mục lab2 trong ổ C.
Chuyển thư mục làm việc trong modelsim đến thư mục này bằng cách dùng lệnh:

```
cd C:/lab2
```

Phần 1

Trong Hình 1 là bảng chân lý và mạch điện của bộ cộng Full Adder (FA) cho 2 số 1-bit. Bộ cộng cho hai số n-bit có thể được mô hình hóa bằng cách sử dụng n bộ cộng 2 số 1-bit. Trong phần thực hành này, chúng ta sẽ thiết kế bộ cộng 1-bit sau đó sử dụng bộ cộng này để thiết kế bộ cộng n-bit với n là 1 số nguyên dương bất kỳ.



b	a	c_i	c_o	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Hình 1. (a) mạch điện của bộ cộng 1-bit; (b) ký hiệu của bộ cộng 1-bit; (c) bản chân lý của bộ cộng 1-bit; (d) bộ cộng 4-bit được tạo ra từ bộ cộng 1-bit.

1. Sử dụng tập tin *full_adder.vhd*, hoàn thành phương trình logic cho các đầu ra s và co.

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY full_adder IS

  PORT (
    a  : IN  STD_LOGIC;
    b  : IN  STD_LOGIC;
    ci : IN  STD_LOGIC;
    s  : OUT STD_LOGIC;
    co : OUT STD_LOGIC);

END ENTITY full_adder;

ARCHITECTURE dataflow OF full_adder IS

BEGIN  -- ARCHITECTURE dataflow

  -- please complete the following lines with the correct logic function
  -- of 's' and 'co'
  s  <= '0';
  co <= '0';

END ARCHITECTURE dataflow;
```

2. Biên dịch bộ *full_adder* và mô phỏng thiết kế này sử dụng các lệnh force như trong bài thực hành số 1.
3. Sử dụng tập tin *full_adder_tb.vhd*, hoàn thành chương trình kiểm tra bộ *full_adder* bằng cách hoàn thành các giá trị đầu vào và đầu ra tương ứng.

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

--------------------------------------------------------------------------------

ENTITY full_adder_tb IS

END ENTITY full_adder_tb;

--------------------------------------------------------------------------------

ARCHITECTURE test OF full_adder_tb IS

  -- component ports
  SIGNAL a       : STD_LOGIC;
  SIGNAL b       : STD_LOGIC;
  SIGNAL ci      : STD_LOGIC;
  SIGNAL s       : STD_LOGIC;
  SIGNAL co      : STD_LOGIC;
  CONSTANT delay : TIME := 10 NS;

  COMPONENT full_adder IS
    PORT (
      a  : IN  STD_LOGIC;
      b  : IN  STD_LOGIC;
      ci : IN  STD_LOGIC;
      s  : OUT STD_LOGIC;
      co : OUT STD_LOGIC);
  END COMPONENT full_adder;
BEGIN  -- ARCHITECTURE test

  -- component instantiation
  DUT : full_adder
    PORT MAP (
      a  => a,
      b  => b,
      ci => ci,
      s  => s,
      co => co);

  -- waveform generation
  WaveGen_Proc : PROCESS
  BEGIN
    -- insert signal assignments here
    -- start a test by assigning the input signal
    a  <= '0';
    b  <= '0';
    ci <= '0';
    WAIT FOR delay;  -- wait for the inputs propagate to outputs
    ASSERT s = '0' AND co = '0' REPORT "Test failed" SEVERITY ERROR;
    -- end a test

  END PROCESS WaveGen_Proc;
```

```
END ARCHITECTURE test;
```

4. Biên dịch và chạy mô phỏng kịch bản kiểm tra bộ *full_adder*.

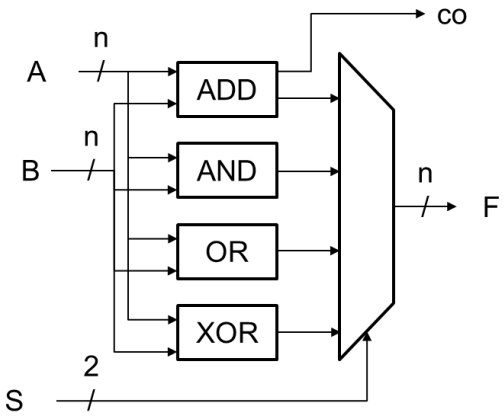
5. Bộ cộng hai số n-bit có thể được thiết kế bằng cách sử dụng các bộ cộng 1-bit đã được thiết kế và kiểm tra. Để thiết kế bộ cộng n-bit với số n thay đổi tùy thuộc vào nhu cầu của người sử dụng, chúng ta phải sử dụng khai báo *GENERIC* của VHDL để tham số hóa tham số n và sử dụng *FOR-GENERATE* để tạo ra cấu trúc phần cứng như trong Hình(d). Sử dụng tập tin adder_nbit.vhd để hoàn thành thiết kế bộ cộng n-bit từ các bộ cộng 1-bit sử dụng cấu trúc *FOR-GENERATE*.

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY adder_nbit IS

  GENERIC (
    DATA_WIDTH : INTEGER := 32);

  PORT (
    cin  : IN  STD_LOGIC;
    a    : IN  STD_LOGIC_VECTOR(DATA_WIDTH-1 DOWNTO 0);
    b    : IN  STD_LOGIC_VECTOR(DATA_WIDTH-1 DOWNTO 0);
    sum  : OUT STD_LOGIC_VECTOR(DATA_WIDTH-1 DOWNTO 0);
    cout : OUT STD_LOGIC);

END ENTITY adder_nbit;

ARCHITECTURE structural OF adder_nbit IS

  SIGNAL cout_temp : STD_LOGIC_VECTOR(DATA_WIDTH-1 DOWNTO 0);
  SIGNAL cin_temp  : STD_LOGIC_VECTOR(DATA_WIDTH-1 DOWNTO 0);

  COMPONENT full_adder IS
    PORT (
      a  : IN  STD_LOGIC;
      b  : IN  STD_LOGIC;
      ci : IN  STD_LOGIC;
      s  : OUT STD_LOGIC;
      co : OUT STD_LOGIC);
  END COMPONENT full_adder;
BEGIN  -- ARCHITECTURE structural

  adder_nbit : FOR i IN 0 TO DATA_WIDTH-1 GENERATE
    -- to be completed
    full_adder_2 : full_adder
      PORT MAP (
        a  => '0',
        b  => '0',
        ci => '0',
        s  => OPEN,
        co => OPEN);
```

```
  END GENERATE adder_nbit;
  cin_tmp <= (OTHERS => '0');           -- to be completed
  cout    <= '0';                       -- to be completed
  sum     <= (OTHERS => '0');           --to be removed
END ARCHITECTURE structural;
```

6. Viết kịch bản kiểm tra cho bộ cộng n-bit.
7. Phương pháp thiết kế sử dụng bộ cộng 1-bit để thiết kế bộ cộng n-bit được gọi là phương pháp gì và phương pháp này được sử dụng khi nào?
8. Nêu sự khác biệt giữa miêu tả phần cứng của bộ *full_adder* và bộ *adder_nbit*.

Phần 2



Hình 2 Bộ ALU với 3 đầu vào S, A, B và hai đầu ra F và CO.

Trong phần này, chúng ta sẽ thiết kế một bộ tính toán số học và logic (ALU) của 2 số A và B với 4 phép toán. Một phép toán số học: phép cộng. Ba phép toán logic là AND, OR, XOR. Kết quả của các phép toán có thể được lựa chọn để đưa ra đầu ra bằng hai bit S[1:0] theo như bảng sau:

S(1)	S(0)	ALU's Operation
0	0	(F, CO) = A + B
0	1	F = A and B
1	0	F = A or B
1	1	F = A xor B

Hai số A và B có thể được cấu hình với số bit bất kỳ.

1. Viết miêu tả phần cứng của bộ ALU nêu trên sử dụng ngôn ngữ VHDL.
2. Viết kịch bản kiểm tra cho bộ ALU nêu trên sử dụng ngôn ngữ VHDL.
3. Nêu sự khác nhau giữa phong cách thiết kế trong Phần 1 và Phần 2 mà bạn đã sử dụng. Ưu điểm và nhược điểm của các phong cách thiết kế này là gì? Chúng được sử dụng khi nào?