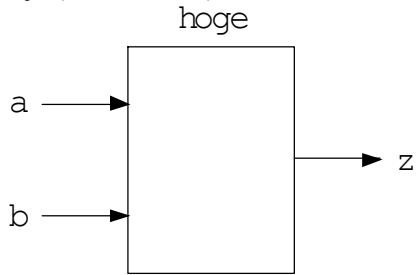


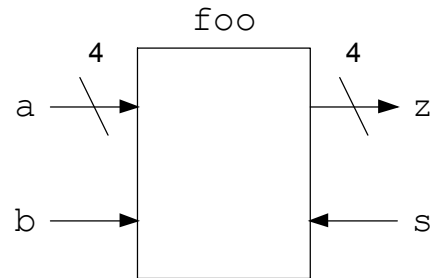
番号	氏名
----	----

1. 次のような入出力を持つ回路を VHDL で記述する際のエンティティ記述を完成させよ。なお回路の名称はブロック図の上に記されている。(10点×2)

(1)



(2)



entity

entity

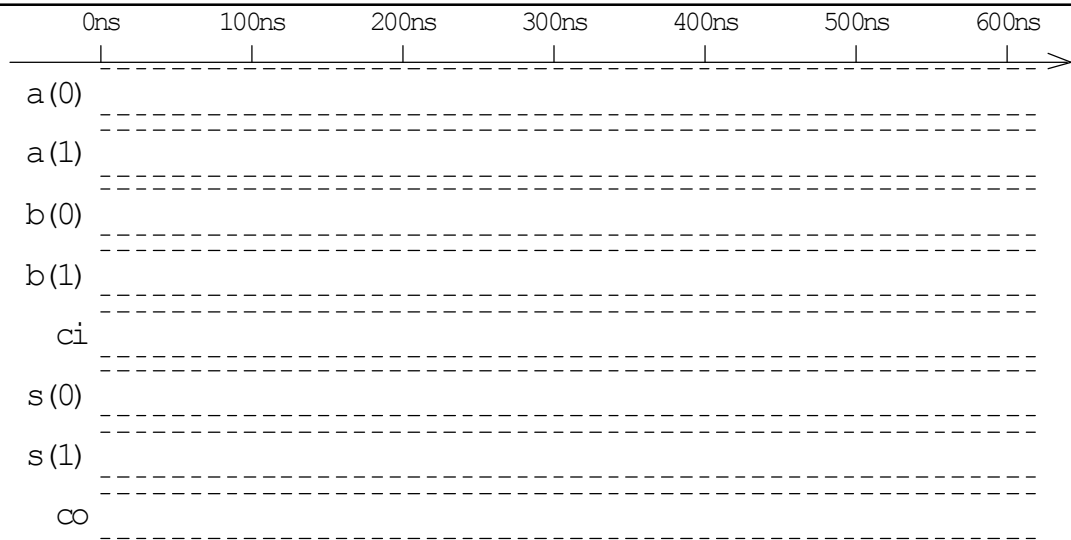
2. 全加算器を2段接続したリプルキャリー加算器 `adder2` に、次のようなテストベンチを与えた場合のシミュレーション結果のタイミングチャートを図中に示せ。なお `adder2` の機能は正しいと仮定し、論理ゲートなどの遅延は考慮しなくてよい。(20点)

```

library ieee;
use ieee.std_logic_1164.all;

entity test_adder2 is
end test_adder2;

architecture arch of test_adder2 is
  component adder2
    port (
      a, b: in std_logic_vector(1 downto 0);
      ci: in std_logic;
      co: out std_logic;
      s: out std_logic_vector(1 downto 0));
  end component;
  for i0: adder2 use entity work.adder4(arch);
  signal a, b: std_logic_vector(1 downto 0);
  signal ci: std_logic;
  signal co: std_logic;
  signal s: std_logic_vector(1 downto 0);
begin
  i0: adder2 port map (a=>a, b=>b, ci=>ci, co=>co, s=>s);
  process begin
    a <= "00"; b <= "00"; ci <= '0'; wait for 100ns;
    a <= "11"; b <= "00"; ci <= '0'; wait for 100ns;
    a <= "11"; b <= "01"; ci <= '0'; wait for 100ns;
    a <= "11"; b <= "00"; ci <= '1'; wait for 100ns;
    a <= "11"; b <= "11"; ci <= '0'; wait for 100ns;
    a <= "11"; b <= "11"; ci <= '1'; wait for 100ns;
    wait;
  end process;
end arch;
    
```



3. 教科書や授業で扱った回路の VHDL 記述を参考に、指定された機能をもつ回路の VHDL 記述を、欠けている箇所を補って完成させよ。(20点×2)

(1) 4ビットプライオリティエンコーダ enc4

※入力の優先度は a(3)が最も高く、以下順に a(0)が最も低い

※1つ以上の入力が1の場合は出力 x の最上位は1となり、それ以外では0となる。なお x の桁数は必要最小数とする

```

library ieee;
use ieee.std_logic_1164.all;

entity enc4 is
  port (
    a: in std_logic_vector(3 downto 0);
    x: out std_logic_vector(  downto 0)
  );
end enc4;

architecture arch of enc4 is
begin
  process ( ) begin
    if (a( ) = '1')
      then x <= "    ";
    elsif (a( ) = '1')
      then x <= "    ";
    elsif (a( ) = '1')
      then x <= "    ";
    elsif (a( ) = '1')
      then x <= "    ";
    else
      x <= "    ";
    end if;
  end process;
end arch;

```

(2) 4ビット桁上げ先見加算器 cla4

```

library ieee;
use ieee.std_logic_1164.all;

entity cla4 is
  port (
    a, b: in std_logic_vector(3 downto 0);
    ci: in std_logic; -- 桁上げ入力
    co: out std_logic; -- 桁上げ出力
    x: out std_logic_vector(3 downto 0) --和
  );
end cla4;

architecture arch of cla4 is
  signal g0, g1, g2, g3; -- 生成項
  signal p0, p1, p2, p3; -- 伝搬項(XOR型)
  signal c0, c1, c2, c3; -- 各桁の桁上げ
begin
  g0 <= a(0) and b(0); p0 <= a(0) xor b(0);
  g1 <=          p1 <=
  g2 <=          p2 <=
  g3 <=          p3 <=
  c0 <=
  c1 <=
  c2 <=
  c3 <=
  co <= c3;
  s(0) <= p0; s(1) <= p1;
  s(2) <= p2; s(3) <= p3;
end arch;

```

4. 以下のような VHDL 記述による回路 hoge を図 4-1 のように接続した回路に対して、図 4-2 のような入力を与えた場合に得られる出力を図 4-2 中に示せ。(20点)

```

library ieee;
use ieee.std_logic_1164.all;

entity hoge is
  port (
    a, b, s, in std_logic;
    x: out std_logic
  );
end hoge;

architecture arch of hoge is
begin
  process (a, b, s) begin
    case s is
      when '0' => x <= a;
      when '1' => x <= b;
      when others => x <= 'X';
    end case;
  end process;
end arch;

```

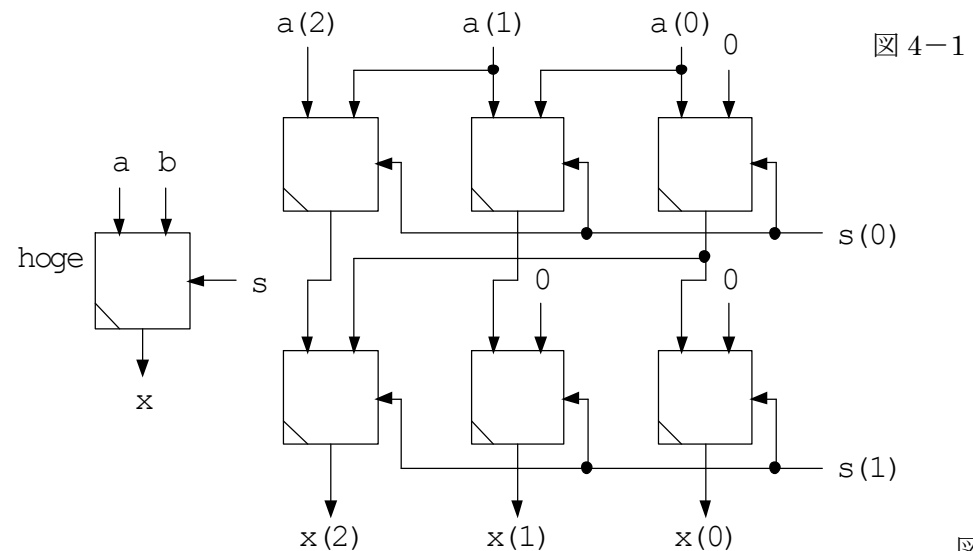


図 4-1

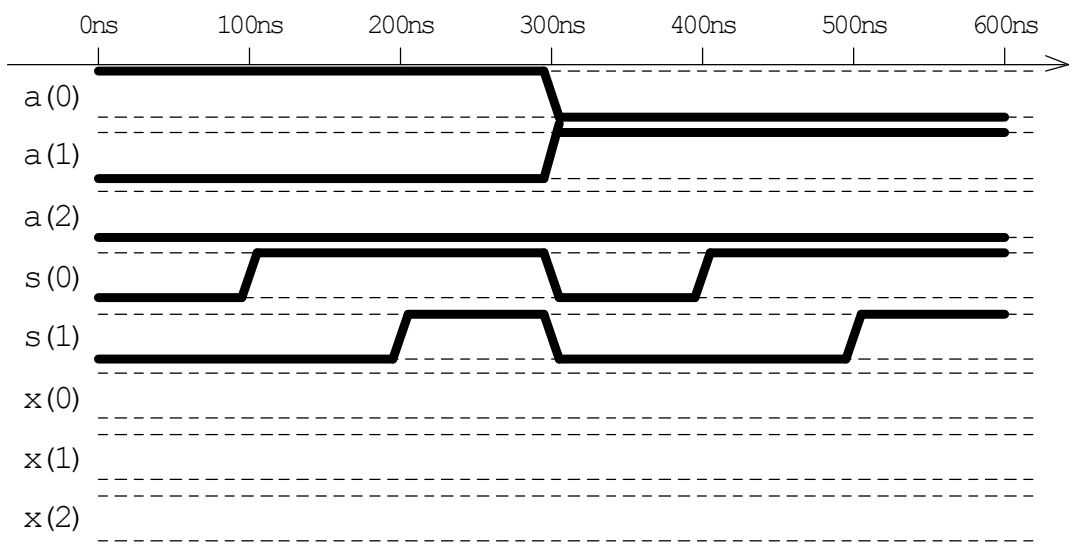


図 4-2