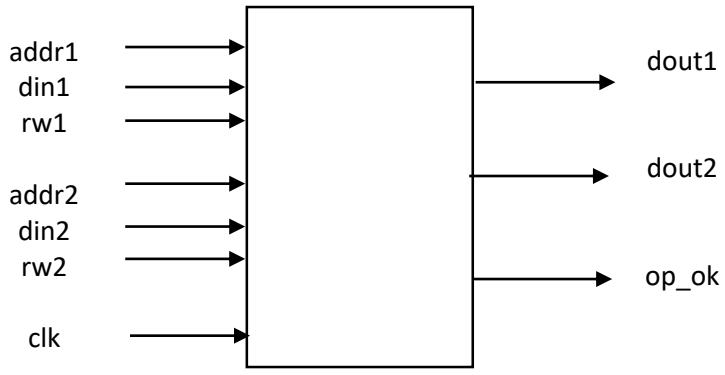
۱. یک شمارنده قابل سنتز طراحی و مدلسازی کنید که دو سیگنال ساعت مختلف داشته باشد و در لبه بالارونده هرکدام از سیگنال‌های ساعت یک واحد بشمارد. [۲۰ نمره]

۲. یک حافظه دو درگاهه به صورت شکل زیر در نظر بگیرید. حافظه با لبه بالارونده سیگنال ساعت کار می‌کند. [۲۰ نمره]



این حافظه دو خط آدرس، دو خطا داده ورودی، دو خط rw ورودی و دو خط داده خروجی دارد. هر مسیر به صورت مجزا و موازی کار می‌کند. هر گاه یک خواندن و یک نوشتن در یک حافظه رخ دهد اول نوشتن رخ دهد و سپس خواندن انجام شود. هر گاه دو نوشتن در یک آدرس رخ دهد، اولویت با خط اول است. در همه حالات خروجی op_ok برابر ’1‘ است و تنها در حالتی که دو نوشتن با داده مختلف در یک آدرس رخ میدهد این خروجی ’0‘ است.

۳. به سوالات زیر پاسخ کوتاه دهید. [۲۰ نمره]

الف. چگونه با وجود ترتیبی بودن عملیات در پراسس، عملکرد واقعی آن موازی است؟

ب. چرا حلقه‌هایی که تعداد تکرار آنها ثابت نیست، قابل سنتز نیستند؟

ج. کدام نوع WAIT قابل سنتز است؟ چه تفاوتی با سایر حالات دارد که باعث شده قابل سنتز باشد؟

د. فرق BUFFER با INOUT چیست؟

۴. یک MUX8x1 را در قالب یک ENTITY مدل کنید. سپس با نمونه‌گیری از حداقل تعداد لازم از آن یک FullAdder یک بیتی طراحی نمایید. [۱۵ نمره]

۵. با استفاده از پراسس بدون لیست حساسیت یک لچ بدون ریست مدل کنید. [۱۵ نمره]

۶. کد زیر قرار است یک ضرب کننده را مدل کند، اما اشکالات متنی و محتوایی دارد. کد را اصلاح کنید. [۲۰ نمره]

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

entity mult is
   port(n1, n2 : in std_logic_vector(1 downto 0);
          pd : out std_logic_vector(3 downto 0));
end mult;

architecture test of mult is
  signal n1_reg: std_logic_vector(2 downto 0);
  signal pd_reg: std_logic_vector(5 downto 0);
begin
   process(n1, n2)
   begin
     n1_reg <= '0' & n1;
     pd_reg <= "0000" & n2;

     for i in 1 to 3 loop
        if pd_reg(0)='1' then
              pd_reg(5 downto 3) <= pd_reg(5 downto 3) + n1_reg(2 downto 0);
        end if;
        pd_reg(5 downto 0) := '0' and pd_reg(5 downto 1);
     end loop;
     pd <= pd_reg;
   end process;
end test;
```