Лабораторна робота № 2

Опис регістрової пам'яті за допомогою мови VHDL

1. Мета роботи:

Оволодіти знаннями та практичними навиками проектування пристроїв пам'яті, таких як регістровий ОЗП (FM). Практична робота також допомогає отримати навики програмування та налагодження опису RAM на мові VHDL.

2. Завдання для виконання практичної роботи:

1. Вивчити основні елементи мови VHDL і правила опису логічних схем.
2. Зробити опис цифрових пристроїв згідно приведених нижче прикладів за допомогою текстового редактора САПР QUARTUS II.
3. Виконати симуляцію роботи схем. Перевірити правильність роботи пристрою по часових діаграмах.
4. Відповісти на контрольні запитання, оформити звіт про виконану роботу.

3. Теоретичні відомості.

Блок FM (Fast Memory – швидка пам'ять) призначений для швидкого доступу до N_R – розрядним словам за кількома довільними адресами. В практичній роботі передбачається, що N_R змінюється від 4 до 16 біт і об'єм пам'яті *M* рівний від 8 до 32 регістрів. Кожний із каналів доступу до FM має свою адресну шину. Кількість каналів доступу рівна 2 або 3 і вони позначаються буквами B, D і Q. Один із каналів призначений для запису, а решта - для читання. Канали можуть бути двохнаправленими, тобто використовуватись як для запису, так і для читання. В практичній роботі двохнаправлений канал рекомендується виконати на основі шини з трьома станами, як в практичній роботі 8.

При проектуванні мікросхем, якщо об'єм RAM невеликий, як у випадку FM, то пам'ять набирають із окремих тригерів. Запис даних в модуль FM завжди виконується по фронту синхросерії або сигналу запису, тобто вхід модуля можна розглядати як вхід синхронного регістру. Прочитане дане видається сразу ж після подачі адресу в FM.

4. Приклади опису FM

Розглянемо приклад проектування трьохканальної FM об'ємом 8 шістнадцятирозрядних слів. Така FM може мати таке оголошення об'єкта:

```
use work.CNetwork.all;
entity FM is    port(CLK:in BIT;            -- синхровхід
    WR:in BIT;                              -- сигнал запису
    AB:in BIT_VECTOR(2 downto 0);           -- адрес каналу B
    AD:in BIT_VECTOR(2 downto 0);           -- адрес каналу D
    AQ:in BIT_VECTOR(2 downto 0);           -- адрес каналу Q
    Q: in BIT_VECTOR(15 downto 0);          -- дане каналу Q
    B: out BIT_VECTOR(15 downto 0);         -- дане каналу B
    D: out BIT_VECTOR(15 downto 0));        -- дане каналу D
end FM;
```

4.1. Поведінкова модель FM

Поведінкова модель FM багато в чому похожа на поведінкову модель RAM, описану в практичній роботі 8. Різниця полягає в тому, що дві паралельні операції читання і запис виконуються за трьома різними адресами, а регістр адресу і трьохстановий буфер - відсутні.

```
architecture BEH of FM is
        type MEM8X16 is array(0 to 7) of BIT_VECTOR(15 downto 0);
        signal addr,do: BIT_VECTOR(15 downto 0);
```

```
begin
   FM8:process(CLK,AD,AB)        -- блок регістрової пам'яті --
       variable RAM: MEM8xl6;
       variable addrq,addrd,addrb:natural;
     begin
       addrq:= BIT_TO_INT(AQ);
       addrd:= BIT_TO_INT(AD);
       addrb:= BIT_TO_INT(AB);
       if CLK='l' and CLK'event then
                if WR = '1' then
                  RAM(addrq): = Q;  -- запис
                end if;
      end if;
       B<= RAM(addrb);       -- читання каналу B
       D<= RAM(addrd);       -- читання каналуD
     end process;
   end BEH;
```

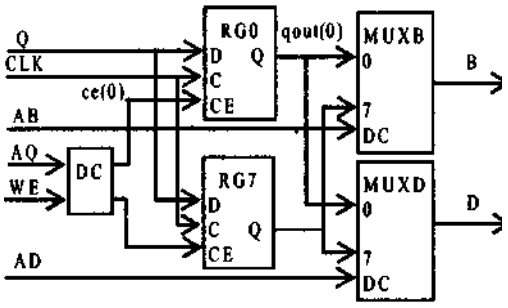
Дана програма відноситься до моделей, описаних стилем синтезу. Компілятор - синтезатор виконує цю пам'ять на окремих тригерах.

4.2. Структурна модель FM

Розглянемо проектування FM на базі PLMT і тригерів. Блок FM повинен мати в своєму складі 8 регістрів розміром 16 біт, 2 мультиплексора зчитаних даних по каналам B, D і дешифратор запису по каналу Q (див. рис.).

Дешифратор DC складається із восьми PLM, кожна із яких декодує три розряди адресу. Перша із них описується наступним чином при D=WE, C=A_2, B= A_1, A=A_0:.

```
architecture PLM_DC0 of PLM_4 is
begin
   Y<=(D and not C and not B and not A) after td;
end PLM_DC0;
```



Структура блоку FM

Решта PLM дешифратора - PLM_DC1, ... , PLM_DC7 - описуються аналогічно. Кожний із мультиплексорів складається із шістьнадцяти однобітних восьмивходових мультиплексорів. Восьмивходовий мультиплексор можна представити як один двохвходовий і два чотирьохвходові мультиплексори. Чотирьохвходовий мультиплексор описаний в практичній роботі 8 як PLM з архітектурою PLM_6(PLM_MUX). Двохвходовий мультиплексор має наступну архітектуру при C = D_1, B= D_0, A=A_0:

```
   architecture PLM_MUX of PLM_3 is
begin
     Y<=(B and not A)        -- 0-й вхід
```

```
        or (C and A)            -- 1-й вхід
      after td;                 -- затримка елемента
end PLM_MUX;
```

Структурная модель восьмивходового мультиплексора описывается в следующем объекте.

```
entity MUX8 is port(D0,D1,D2,D3,D4,D5,D6,D7: in BIT; -- входи даних
                A: in BIT_VECTOR(2 downto 0);       -- адрес
                Q: out BIT);                        -- вихід даного
end MUX8;
architecture STR of MUX8 is
    signal mux0,mux1:BIT;
 begin
    U_MUX0: entity PLM_6(PLM_MUX)
        portmap(F=>D3,E=>D2,D=>D1,C=>D0,B=>A(l),A=>A(0),Y=>mux0);
    U_MUX1: entity PLM_6(PLM_MUX)
        portmap(F=>D7,E=>D6,D=>D5,C=>D4,B=>A(l),A=>A(0),Y=>muxl);
    U_MUX3: entity PLM_3(PLM_MUX)
        portmap(C=>mux1,B=>mux0,A=>A(2),Y=>Q);
 end STR;
```

Тіло архітектури регістрової пам'яті виглядає так:

```
architecture STR of FM is
        type FMARR is array(7 downto 0, 15 downto 0) of BIT;
        signal y: FMARR;
        signal ce:BIT_VECTOR(7 downto 0);
        constant gnd:BIT:='0';
        component MUX8 is port(D0,D1,D2,D3,D4,D5,D6,D7: in BIT;   -- входи
                        A: in BIT_VECTOR (2 downto 0);            -- адрес
                        Q: out BIT);                              -- вихід даного
        end component;
        component FDRE is port (Q:out BIT;                         -- тригер
                        D,C,CE,R: in   BIT);
        end component;
begin           -- дешифратор адреса (компоненти U_DC2,..., U_DC6 пропущені)
        U_DC0: entity PLM_4(PLM_DC0)
        port map(D=>WR,C=>AQ(2),B=>AQ(1),A=>AQ(0),Y=>ce(0));
        …

        U_DC7: entity PLM_4(PLM_DC7)
        port map(D=>WR,C=>AQ(2),B=>AQ(1),A=>AQ(0),Y=>ce(7));
    --- масив регістрів ---
U_FM: for i in 0 to 7 generate
        U_RG: for j in 0 to 15 generate
                U_TT: FDRE port map (D=>Q(j),   -- вхідне дане
                        C =>CLK,                -- синхросигнал
                        CE=>ce(i),              -- дозвіл запису
                        R =>gnd,                -- скидання не використовується
                        Q=>y(i,j));             -- виходи тригерів
                end generate;
end generate;

--- вихідні мультиплексори ---
```

```
U_MUX: for i in 0 to 15 generate
   MUXD:MUX8 port map(D0=>y(0,i),D1=>y(1,i),D2=>y(2,i),D3=>y(3,i),
              D4=>y(4,i),D5= >y(5,i),D6= >y(6,i),D7=>y(7,i),
              A=>AD,                          -- адрес
              Q=>D(i));                       -- вихід каналу D
    MUXB:MUX8 port map(D0=>y(0,i),D1=>y(l,i),D2=>y(2,i),D3=>y(3,i),
              D4=>y(4,i),D5=>y(5,i),D6=>y(6,i),D7=>y(7,i),
              A=>AB,                          -- адрес
              Q= >B(i));                      -- вихід каналу B
     end generate;
end STR;
```

Для зв'язку виходів тригерів FM з входами мультиплексорів використовується сигнал ***y*** типу двохмірний масив розмірами 8x16.

5. Питання до практичної роботи..

Яке функціональне призначення FM?
Які елементи використовуються при проектуванні FM?
Запропонуйте 5 способів задання дешифратора на 4 виходи.
Якими способами задають FM в проектах на VHDL?
Від чого і наскільки залежатьт апаратні затрати FM?
Чому в структурному проекті FM вставка дешифратора виконана як вставка об'єкта, а вставка тригера - як вставка компонента?
Чому всі тригери FM потрібно підключати до одного джерела синхросерії?

6. Рекомендована література.

1. Сергиенко А.М., Корнейчук В.И. Микропроцессорные устройства на программируемых логических ИС. -К.: «Корнійчук», 2005. -108 с.