احمد فروغی علی نوروزی

عنوان آزمایش: طراحی حافظه RAM & ROM

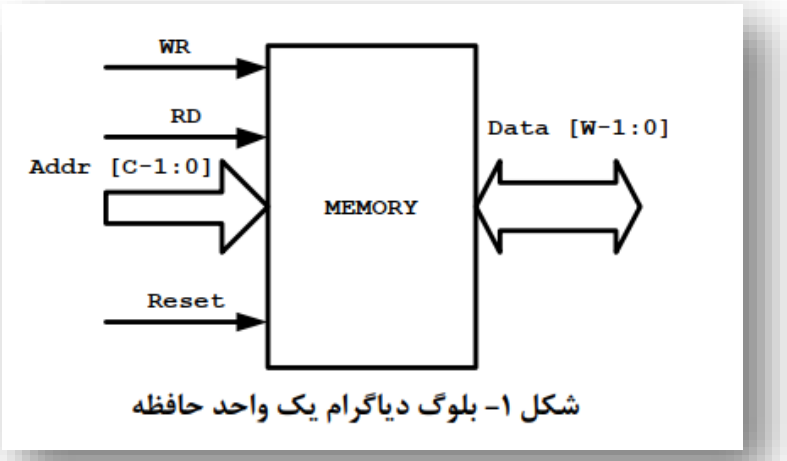
هدف از آزمایش: آشنایی با انواع حافظه، طراحی و پیاده سازی آنهاست.

شرح آزمایش: یکي از بخش های اصلي در مدارهاي ديجيتال، بخش حافظه است. مدلسازي نامناسب حافظه علاوه بر اينکه ممکن است زمان شبيه سازي را طولاني کند، در هنگام سنتز نيز در صورت توصيف نادرست باعث استفاده غيرمعمول از منابع سختافزاري ميشود. در اين آزمايش هدف اصلي آشنايي با نحوه مدل کردن انواع حافظه ها با زبان توصيف سخت افزار است.

طراحی حافظه RAM:

یک حافظه RAM مطابق با شکل زیر طراحي کنيد. پارامترهاي اين بلوک به شرح زير ميباشد:

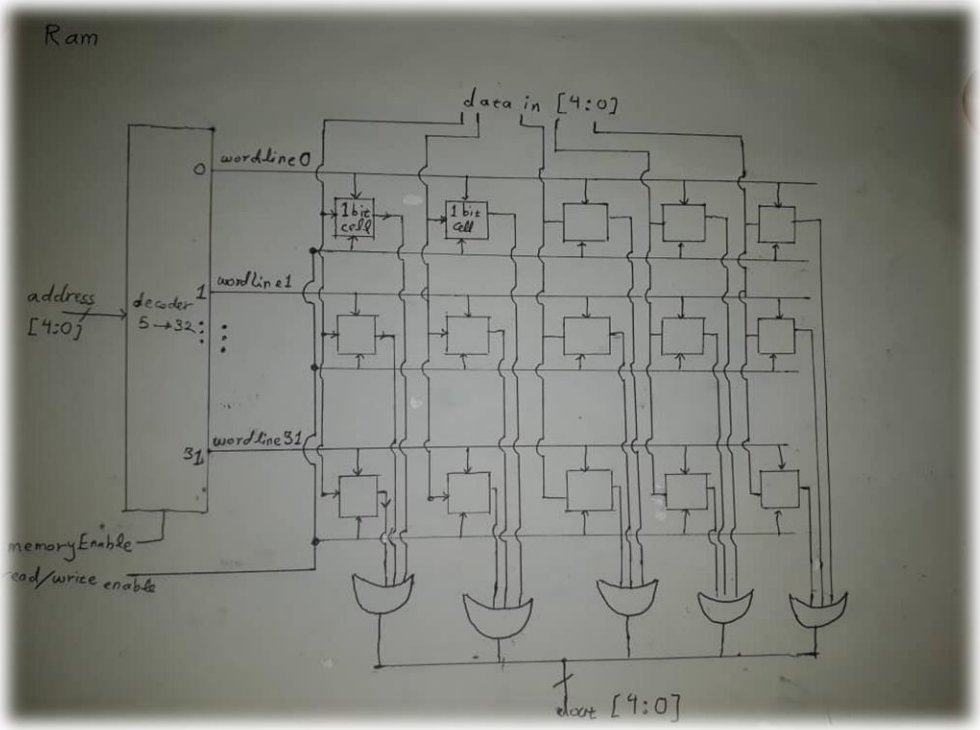
- عرض حافظه W
- تعداد خانه های حافظه D
- عرض درگاه Address برابر با lg(D) میباشد که فعلا آن را هم در کد با پارامتر C نشان میدهیم.



شکل ۱- بلوک دیاگرام یک واحد حافظه

احمد فروغی علی نوروزی

دیاگرام:



روش انجام آزمایش:

ابتدا مانند آنچه در آزمایش های قبل دیدیم به طراحی خود کامپوننت اصلی RAM میپردازیم. برای اینکار ابتدا Port های آن را تعریف میکنیم:

```
entity Ram is
    Port ( clk : in  STD_LOGIC;
           rst : in  STD_LOGIC;
           write_en : in  STD_LOGIC;
           read_en : in  STD_LOGIC;
           address : in  STD_LOGIC_VECTOR(4 downto 0);
           dout : out  STD_LOGIC_VECTOR(4 downto 0);
           din : in  STD_LOGIC_VECTOR(4 downto 0));
end Ram;
```

سپس یک آرایه ۳۲ بیتی (ردیف های آرایه) تعریف کرده که هر ردیف شامل ۵ بیت میباشد:

```
-- Depth of memory: 32
-- Width of memory: 5bit
-- Address bits: 5

Type Ram_Array is array (31 downto 0) of STD_LOGIC_VECTOR(4 downto 0);

Signal Memory: Ram_Array;
```

درنهایت Process را ایجاد میکنیم و با توجه به آنکه طراحی ما Single Port است (یعنی همزمان نمیتوان خواند و نوشت – یکی از اعمال در یک زمان قابل انجام است) خواهیم داشت:

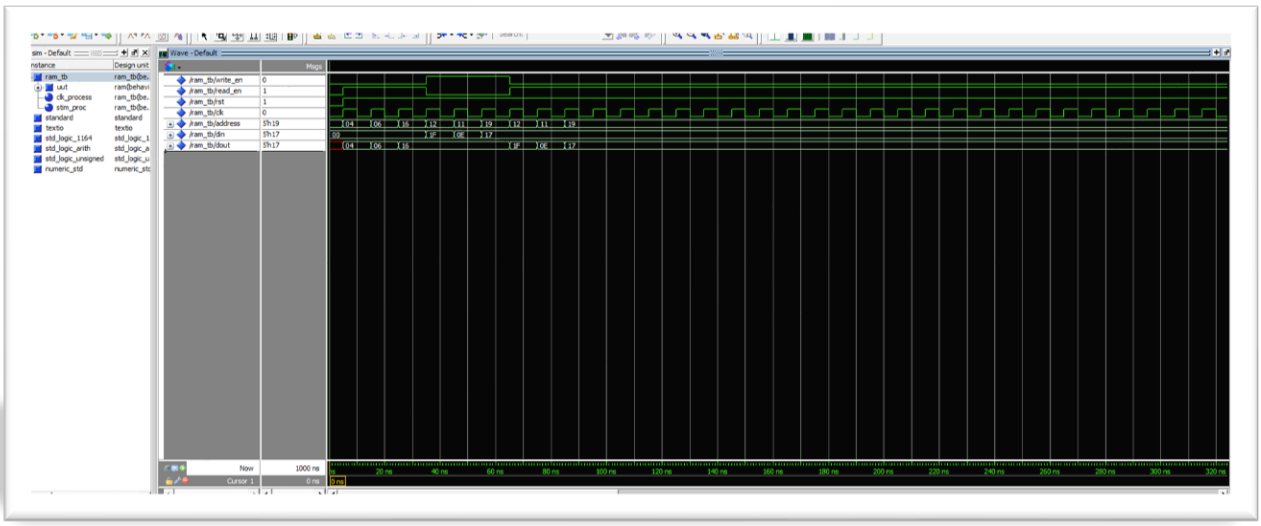
```
process(clk, rst)
begin
    if (rst = '0') then
        for i in 0 to 31 loop
            Memory(i) <= std_logic_vector(to_unsigned(i,5));
        end loop;
    elsif (clk'event and clk = '1') then
        if (write_en = '1' and read_en = '0') then

            Memory(to_integer(unsigned(address))) <= din;
        elsif ( read_en = '1' and write_en = '0') then

            dout <= Memory( to_integer( unsigned(address) ));
        end if;
    end if;
end process;
```

نکته مهم در این بخش ان است که در هنگام ریست (مطابق دستورکار آزمایشگاه) بایستی دیتای داخل هر خانه معادل شماره همان خانه باشد.

درنهایت با استفاده از یک Test Bench کامپوننت خود را تست میکنیم (ضمیمه – نتیجه زیر)

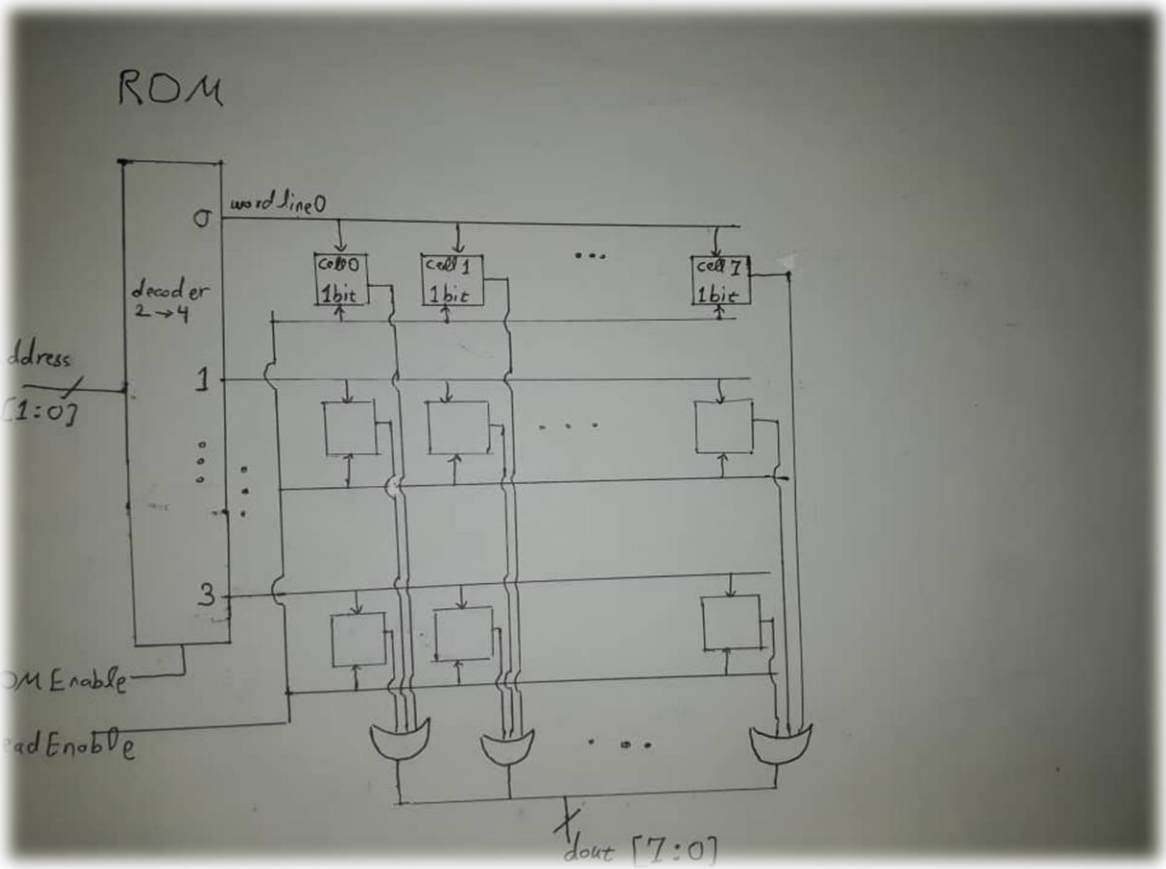


احمد فروغی علی نوروزی

طراحی حافظه ROM:

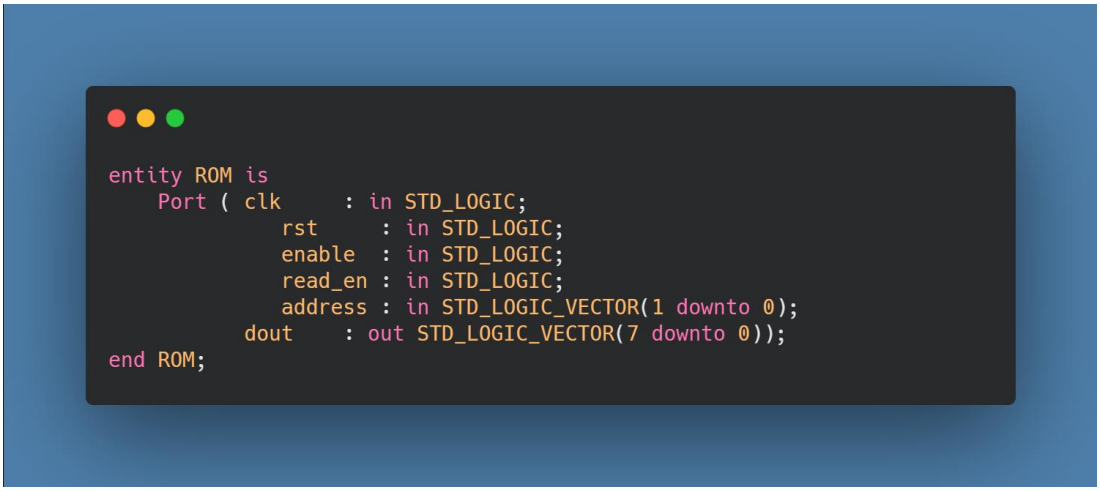
با تغییر اندکی در طراحی قبل حافظه ROM را میسازیم.

نکته مهم در اینجا آن است که در حافظه ROM برخلاف حافظه RAM تنها خوانش اطلاعات از پیش تعیین شده را داریم بنابراین ساخت آن با تغییر اندکی صورت میگیرد:



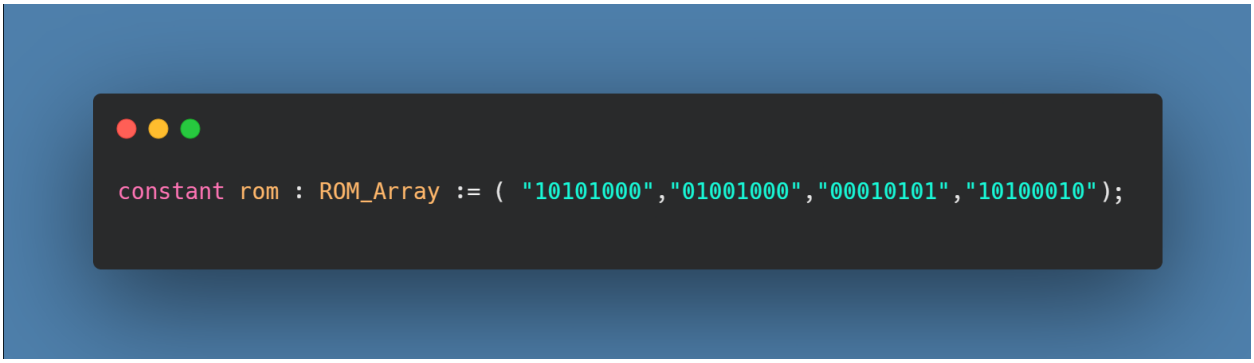
روش انجام آزمایش:

ابتدا در ماژول اصلی پورت ها را تعریف میکنیم



```
entity ROM is
    Port ( clk     : in STD_LOGIC;
           rst     : in STD_LOGIC;
           enable  : in STD_LOGIC;
           read_en : in STD_LOGIC;
           address : in STD_LOGIC_VECTOR(1 downto 0);
           dout    : out STD_LOGIC_VECTOR(7 downto 0));
end ROM;
```

سپس مقدایر Constant (ثابت)، آرایه ROM را ست میکنیم:



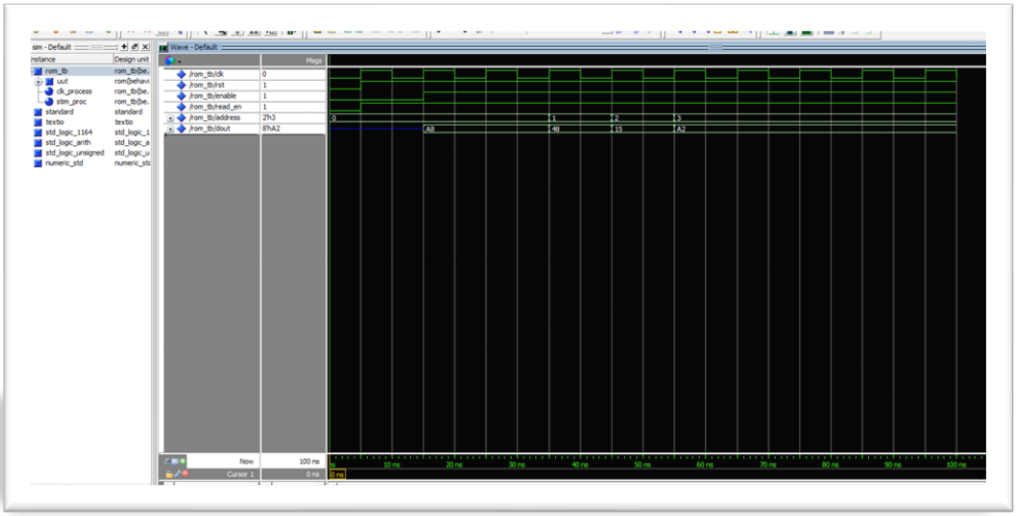
```
constant rom : ROM_Array := ( "10101000","01001000","00010101","10100010");
```

و در نهایت Process را ایجاد میکنیم که دو حالت ریست و لبه بالارونده Clock را مدنظر قرار میدهد. در صورتی که درحالت ریست باشیم خروجی ما High-Z خواهد بود.

```
process(clk)
    begin
        if(rst='0') then
            dout <= "ZZZZZZZZ";
        elsif(clk'event and clk='1') then
            if( enable = '1') then
                if(read_en = '1') then
                    dout <= rom( to_integer(unsigned(address)));
                else
                    dout <= "ZZZZZZZZ";

                end if;
            end if;
        end if;
end process;
```

و درنهایت با استفاده از کد فوق و اجرای Test Bench (موجود در ضمیمه – پیش از این بررسی شده است) ماژول (کامپوننت) خود را بررسی میکنیم.



موفق باشید