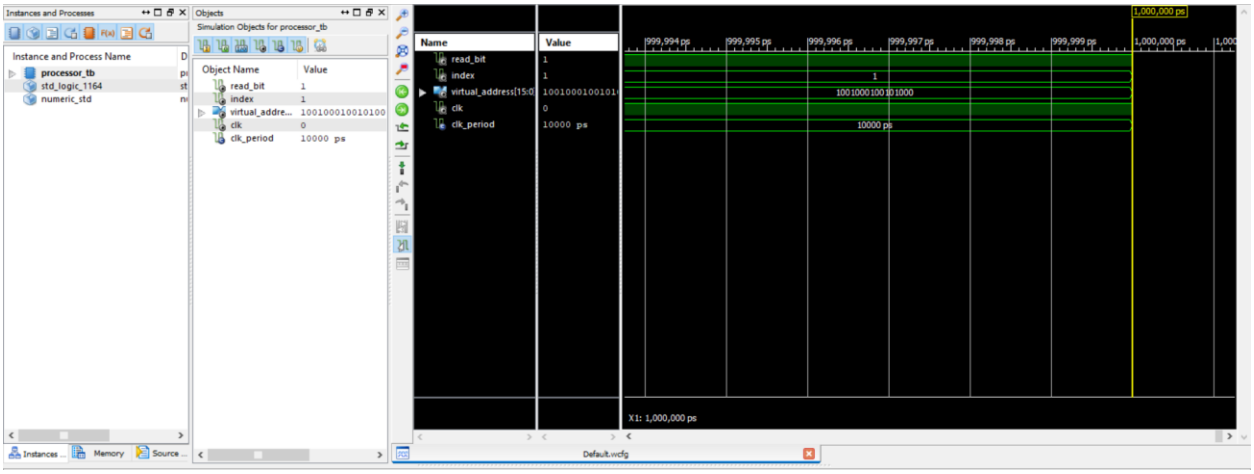
پروژه معماری کامپیوتر
ترم ۴۰۰۲

امیر فخارزاده ۹۹۵۲۱۴۸۷

سینا زمانی ۹۹۵۲۱۳۲۵

ارائه : ۱۴۰۱/۰۴/۱۹

Processor



تست بنچ به درستی ایندکس ۱ از پراسسور را به ما برگردانده است.

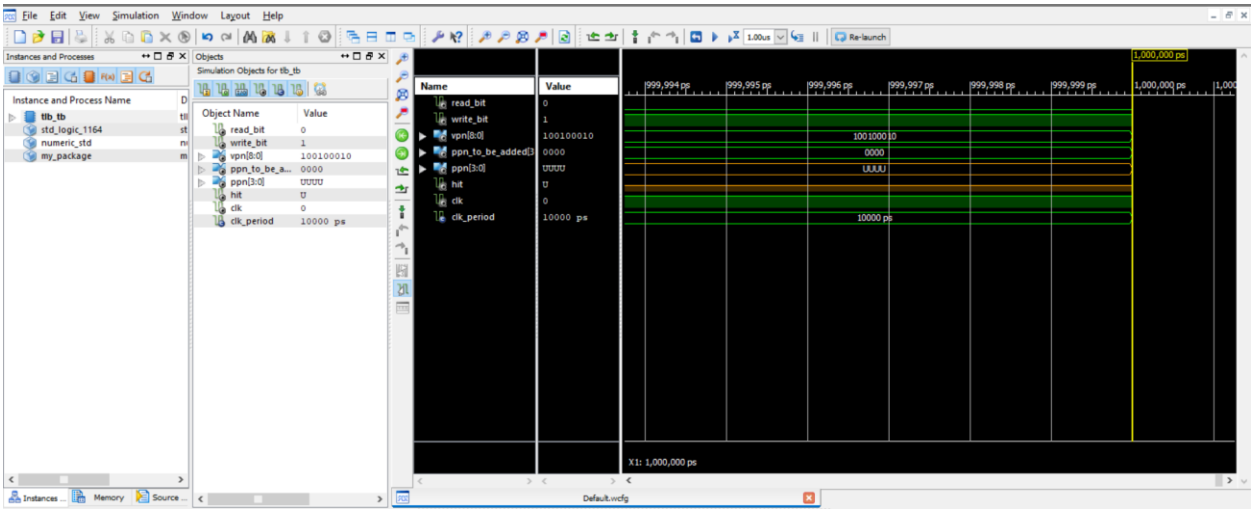
```
entity Processor is
   port (
      read_bit : in std_logic;
      index : in integer;
      virtual_address : out std_logic_vector(15 downto 0)
   );
end Processor;
```

و کدی که برای هنگام اجرا قرار است داشته باشیم در زیر آمده است:

```
process (index)
is
type virtual_address_array_type is array (0 to 4) of std_logic_vector(15 downto 0);
variable all_virtual_addresses : virtual_address_array_type := ("1011110111000110",
                                                                "1001000100101000",
                                                                "1000111010100011",
                                                                "0001011011010001",
                                                                "1011110001111100");
begin
   if read_bit = '1' then
      virtual_address <= all_virtual_addresses(index);
   end if;
end process;
```

همانطور که از تست بنچ مشخص است ایندکس ۱ به ما برگردانده شده است.

TLB



اینجا در حال پر کردن اطلاعات مربوط به ایندکس ۱ پروسسور میباشد.

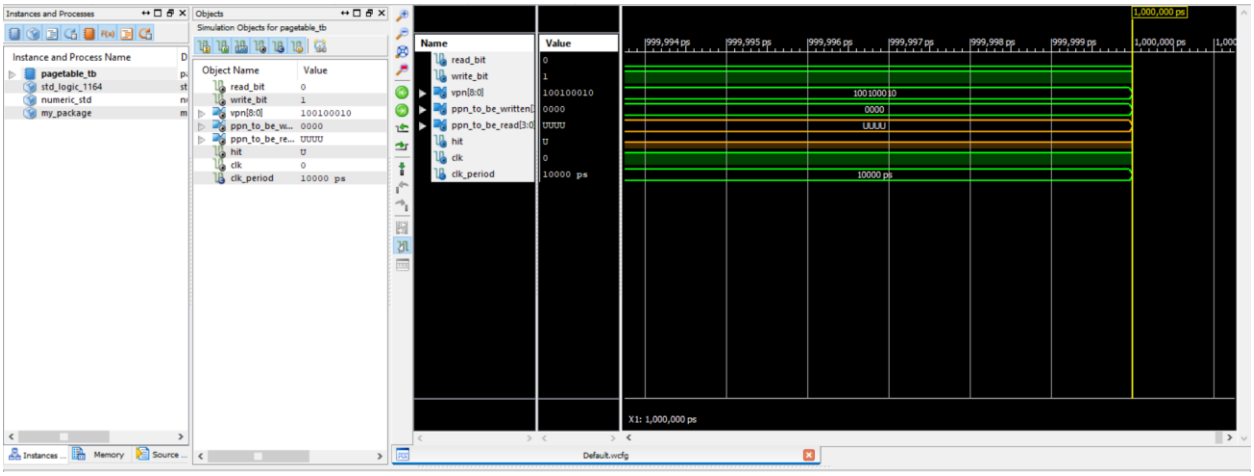
از آنجایی که در تست بنچ ها فایل های دیگر را دخل ندادیم تمامی تست بنچ ها خروجی به همین شکل دارند.

```
entity fully_associative_TLB is
    port (
        read_bit : in std_logic;
        write_bit : in std_logic;
        vpn : in std_logic_vector(8 downto 0);
        ppn_to_be_added : in std_logic_vector(3 downto 0);
        ppn : out std_logic_vector(3 downto 0);
        hit : out std_logic
    );
end fully_associative_TLB;
```

اما رفتار این انتیتی در قبال مقدار هایی که به آن داده میشود عبارت است از:

```
architecture TLB_behavioral of fully_associative_TLB is
signal TLB_ppns : TLB_ppns_type := (others => ( others => '0' ));
signal TLB_tags : TLB_tags_type := (others => ( others => '0' ));
signal TLB_valids : std_logic_vector(47 downto 0);
signal index : integer := 0;
begin
   process (vpn)
   is
   begin
      if read_bit = '1' then
         hit <= '0';
         for i in 0 to 47 loop
            if TLB_valids(i) = '1' and TLB_tags(i) = vpn then
               ppn <= TLB_ppns(i);
               hit<= '1';
               exit;
            end if;
         end loop;
      end if;
      if write_bit = '1' then
         TLB_valids(index) <= '1';
         TLB_tags(index) <= vpn;
         TLB_ppns(index) <= ppn_to_be_added;
         index <= (index + 1) mod 48;
      end if;
   end process;
end   TLB_behavioral;
```

Page Table



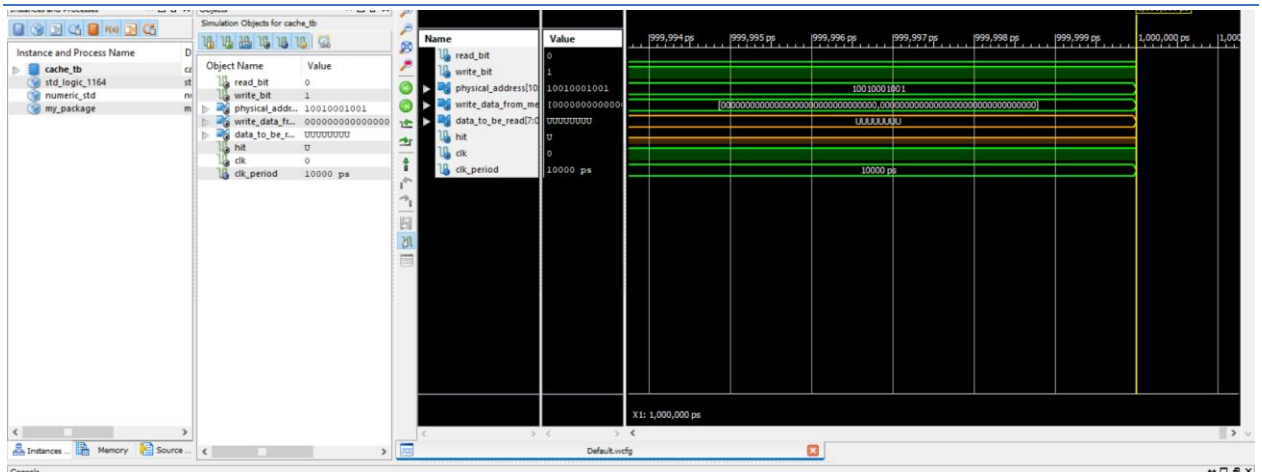
این قسمت هم مانند بخش قبلی چون پیاده سازی جدا جدا داشته است خروجی درستی به ما نمیتواند نشان بدهد.

```
entity Page_Table is
  port (
   read_bit : in std_logic;
    write_bit : in std_logic;
    vpn : in std_logic_vector(8 downto 0);
    ppn_to_be_written : in std_logic_vector(3 downto 0);
    ppn_to_be_read : out std_logic_vector(3 downto 0);
    hit : out std_logic
  );
end Page_Table;
```

اما هنگامی که مقادیر این انتیتی را پر میکنیم به شکل زیر رفتار میکند:

```
architecture Page_Table_behavioral of Page_Table is
signal Page_Table_ppns : Page_Table_ppns_type := (others => (others => '0'));
signal Page_Table_valids: std_logic_vector(0 to 511) := (others => '0');
begin
  process (vpn)
  is
  begin
    if read_bit = '1' then
      hit <= '0';
      if Page_Table_valids(to_integer(unsigned(vpn))) = '1' then
        ppn_to_be_read <= Page_Table_ppns(to_integer(unsigned(vpn)));
        hit <= '1';
      end if;
    end if;
    if write_bit = '1' then
      Page_Table_valids(to_integer(unsigned(vpn))) <= '1';
      Page_Table_ppns(to_integer(unsigned(vpn))) <= ppn_to_be_written;
    end if;
  end process;
end  Page_Table_behavioral;
```

Cache



تست بنچ بخش کش مانند بخش های قبلی، خروجی نامشخص بدلیل دخالت نداشتن بخش های دیگه.

```
entity Direct_Map_Cache is
    port (
        read_bit : in std_logic;
        write_bit : in std_logic;
        physical_address : in std_logic_vector(10 downto 0
        write_data_from_memory : in double_word_data_Type;
        data_to_be_read : out std_logic_vector(7 downto 0)
        hit : out std_logic
    );
end Direct_Map_Cache;
```

حال در مواجهه با داده این انتیتی رفتار زیر را از خود نشان میدهد:

```
architecture Direct_Map_Cache_behavioral of Direct_Map_Cache is
signal cache_valids : std_logic_vector(31 downto 0) := (others => '0');
signal cach_data : cach_data_blocks_type := (others => (others => (others => '0')));
signal chache_tags : chache_tag_data_type := (others => (others => '0'));
begin
   process (physical_address)
   is
      variable byte_offset : std_logic_vector(1 downto 0) := physical_address(1 downto 0);
      variable word_offset : std_logic := physical_address(2);
      variable index : std_logic_vector(4 downto 0) := physical_address(7 downto 3);
      variable tag : std_logic_vector(2 downto 0) := physical_address(10 downto 8);
      variable block_data : double_word_data_Type;
      variable word_data : std_logic_vector(31 downto 0);
   begin
      if read_bit = '1' then
         if cache_valids(to_integer(unsigned(index))) = '0' then
            cache_valids(to_integer(unsigned(index))) <= '1';
            hit <= '0';
         elsif chache_tags(to_integer(unsigned(index))) = tag then
            hit <= '1';
            block_data := cach_data(to_integer(unsigned(index)));
            if word_offset = '0' then
               word_data := block_data(0);
            else
               word_data := block_data(1);
            end if;
            data_to_be_read <= word_data(to_integer(unsigned(byte_offset)) * 8 + 7 downto to_integer(unsigned(byte_offset)) * 8);
         else
            hit <= '0';
         end if;
      end if;
      if write_bit = '1' then
         cache_valids(to_integer(unsigned(index))) <= '1';
         cach_data(to_integer(unsigned(index))) <= write_data_from_memory;
         chache_tags(to_integer(unsigned(index))) <= tag;
      end if;
   end process;

end   Direct_Map_Cache_behavioral;
```