MOUSA AL-QAWASMI
1/3/20

LAB 2 TUTORIAL

PROGRAM COUNTER AND REGISTER SET DESIGN

OVERVIEW

In this lab we will implement, test, and simulate:

1. The 32-bit and 1-bit Register Set required for the 32-bit CPU.
2. The 32-bit Program Counter (PC) required for the 32-bit CPU.

PROCEDURE

1. 1-bit Register

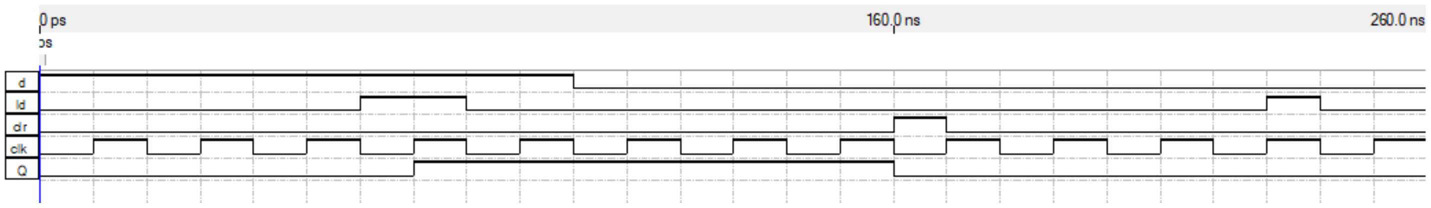
Please follow the following instructions to implement the 1-bit register:

1. Create a new folder "COE608" in your working directory.
2. Create a new folder "Lab2" in your "COE608" folder.
3. Create a new folder "register1" in your "Lab2" folder.
4. Open the Quartus ii software, and using the new project wizard, create a new project "register1" in your "../COE608/Lab2/register1" folder.

   MAKE SURE THAT YOU CHOOSE THE "EP4CE115F29C7" DEVICE IN THE PROJECT WIZARD.
5. Create a new VHDL file in your "register1" project (File > New > VHDL File).
6. Type the following in the Text Editor and save the file as "register1.vhd":

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

entity register1 is
port(
        d       :   in  std_logic;
        ld      :   in  std_logic;
        clr     :   in  std_logic;
        clk     :   in  std_logic;
        Q       :   out std_logic
        );
end register1;

architecture Behavior of register1 is
begin
        process (ld, clr, clk)
        begin
                if clr = '1' then
                        Q <= '0';
                elsif ((clk'event and clk = '1') and (ld = '1')) then
                        Q <= d;
                end if;
        end process;
end Behavior;
```

7. Start the compiler. Fix any errors and re-compile. Once the compiler compiles without any errors, move to the next step.
8. Create a new University Program VWF (File > New > University Program VWF).
9. Simulate "register1.vhd" and make sure that your simulation results match the results shown below.
10. Save the VWF file as "register1.vwf" in your "../COE608/Lab2/register1" folder and take a screenshot of your results.

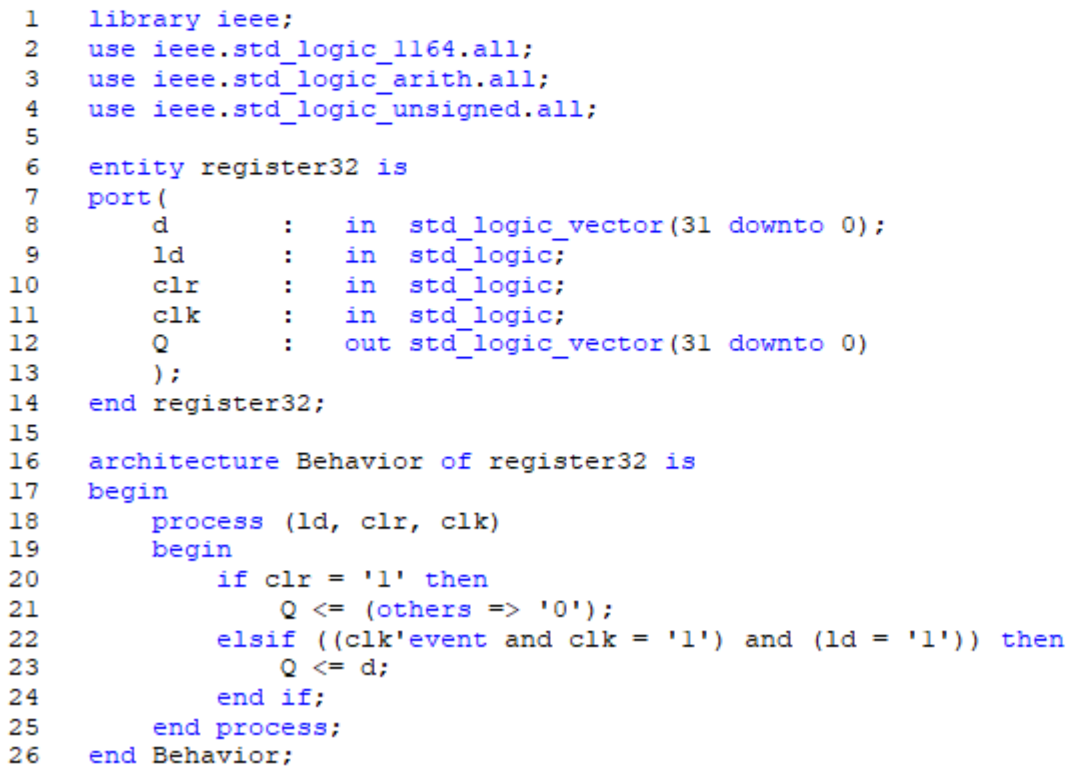


2. 32-bit Register

Please follow the following instructions to implement the 32-bit register:

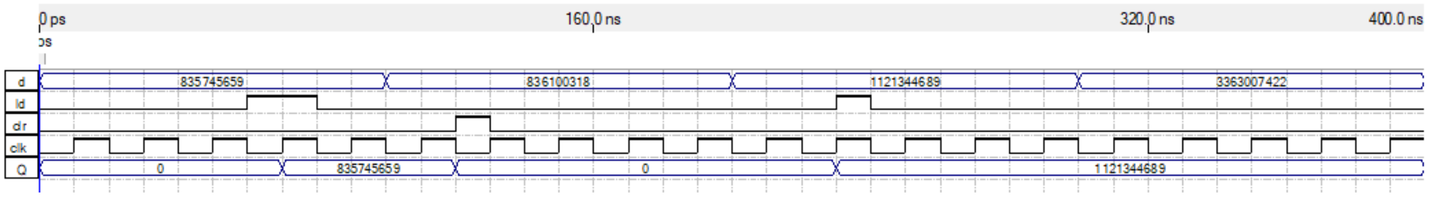
1. Create a new folder “register32” in your “../COE608/Lab2” folder.
2. Open the Quartus ii software, and using the new project wizard, create a new project “register32” in your “../COE608/Lab2/register32” folder.

   MAKE SURE THAT YOU CHOOSE THE “EP4CE115F29C7” DEVICE IN THE PROJECT WIZARD.
3. Create a new VHDL file in your “register32” project (File > New > VHDL File).
4. Type the following in the Text Editor and save the file as “register32.vhd”:



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

entity register32 is
port(
    d       :   in  std_logic_vector(31 downto 0);
    ld      :   in  std_logic;
    clr     :   in  std_logic;
    clk     :   in  std_logic;
    Q       :   out std_logic_vector(31 downto 0)
    );
end register32;

architecture Behavior of register32 is
begin
    process (ld, clr, clk)
    begin
        if clr = '1' then
            Q <= (others => '0');
        elsif ((clk'event and clk = '1') and (ld = '1')) then
            Q <= d;
        end if;
    end process;
end Behavior;
```

5. Start the compiler. Fix any errors and re-compile. Once the compiler compiles without any errors, move to the next step.
6. Create a new University Program VWF (File > New > University Program VWF).
7. Simulate "register32.vhd" and make sure that your simulation results match the results shown below.
8. Save the VWF file as "register32.vwf" in your "../COE608/Lab2/register32" folder and take a screenshot of your results.



3. Program Counter

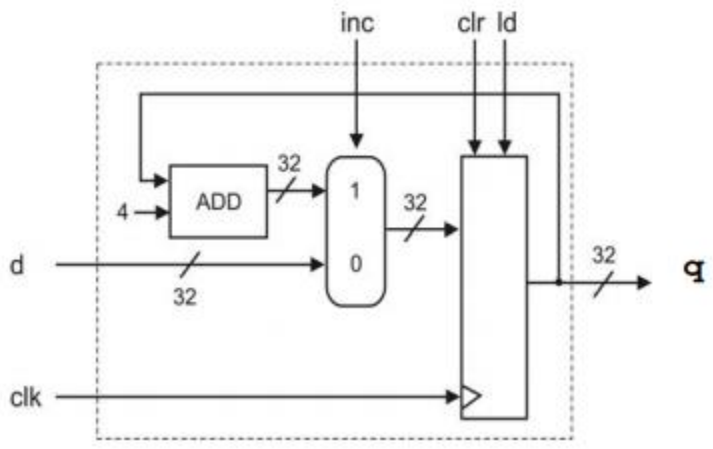
The schematic of the Program Counter is shown in the figure below. As shown in the figure, the Program Counter consists of:

1. An Add block to increment the PC by 4 during instruction execution.
2. A 2 to 1 Multiplexer.
3. A 32-bit Register.

Please follow the following instructions to implement the Program Counter:

4. Create a new folder "PC" in your "..COE608/Lab2" folder.
5. Open the Quartus ii software, and using the new project wizard, create a new project "PC" in your "../COE608/Lab2/PC" folder.

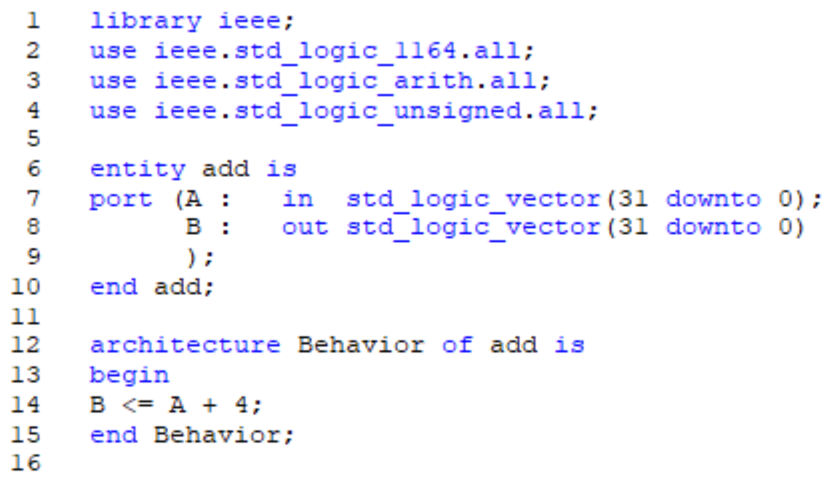
   MAKE SURE THAT YOU CHOOSE THE "EP4CE115F29C7" DEVICE IN THE PROJECT WIZARD.



i. Add

Please follow the following instructions to implement the Add block in your PC project:

1. Create a new VHDL file in your "PC" project (File > New > VHDL File).
2. Type the following in the Text Editor and save the file as "add.vhd":



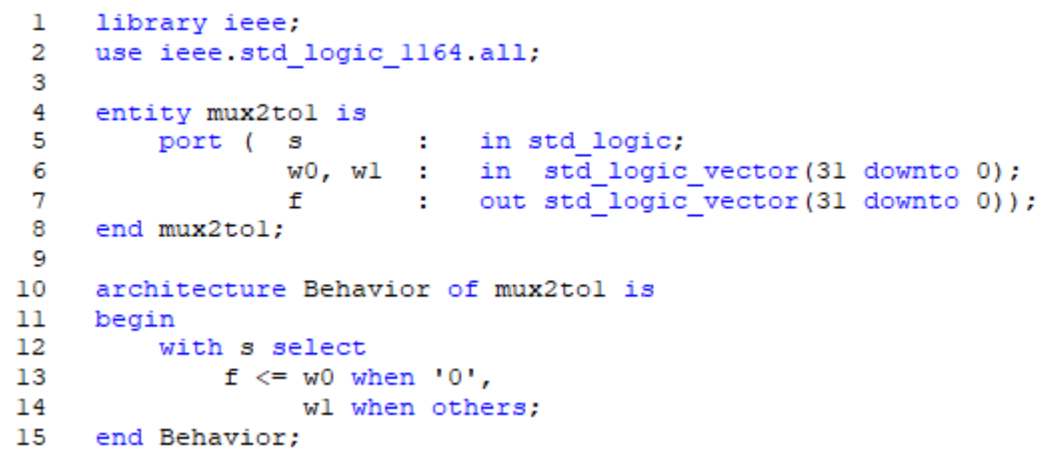
```vhdl
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity add is
port (A :   in  std_logic_vector(31 downto 0);
      B :   out std_logic_vector(31 downto 0)
      );
end add;

architecture Behavior of add is
begin
B <= A + 4;
end Behavior;

```

3. **Set "add.vhd" as the top-level entity.** Start the compiler. Fix any errors and re-compile. Once the compiler compiles without any errors, move to the next step.
4. Simulate your design, and make sure that "add.vhd" functions as intended before moving to the next step.

ii. <u>2 to 1 Multiplexer</u>

Please follow the following instructions to implement the 2 to 1 Multiplexer in your PC project:

1. Create a new VHDL file in your “PC” project (File > New > VHDL File).
2. Type the following in the Text Editor and save the file as “mux2to1.vhd”:



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

entity mux2to1 is
    port (  s        :   in std_logic;
            w0, w1   :   in  std_logic_vector(31 downto 0);
            f        :   out std_logic_vector(31 downto 0));
end mux2to1;

architecture Behavior of mux2to1 is
begin
    with s select
        f <= w0 when '0',
             w1 when others;
end Behavior;
```

3. **Set “mux2to1.vhd” as the top-level entity.** Start the compiler. Fix any errors and re-compile. Once the compiler compiles without any errors, move to the next step.
4. Simulate your design, and make sure that “mux2to1.vhd” functions as intended before moving to the next step.

iii. <u>32-bit Register</u>

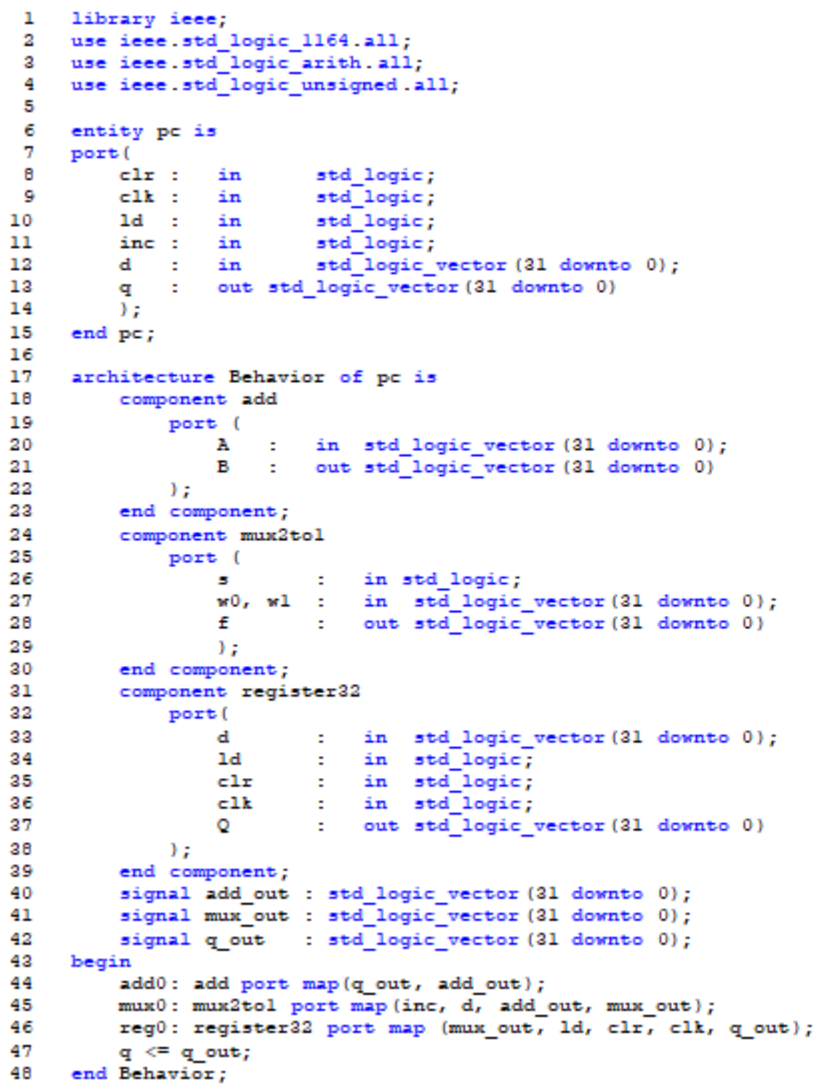
**** Please follow the following instructions to implement the 32-bit Register in your PC project:

1. Open the “register32.vhd” file located in your “../COE608/Lab2/register32” folder.
2. Save the VHDL file as “register32.vhd” in your **“../COE608/Lab2/PC”** folder.
3. **Set “register32.vhd” as the top-level entity.** Start the compiler. Fix any errors and re-compile. Once the compiler compiles without any errors, move to the next step.
4. Simulate your design, and make sure that “register32.vhd” functions as intended before moving to the next step.

iv. Overall PC Implementation

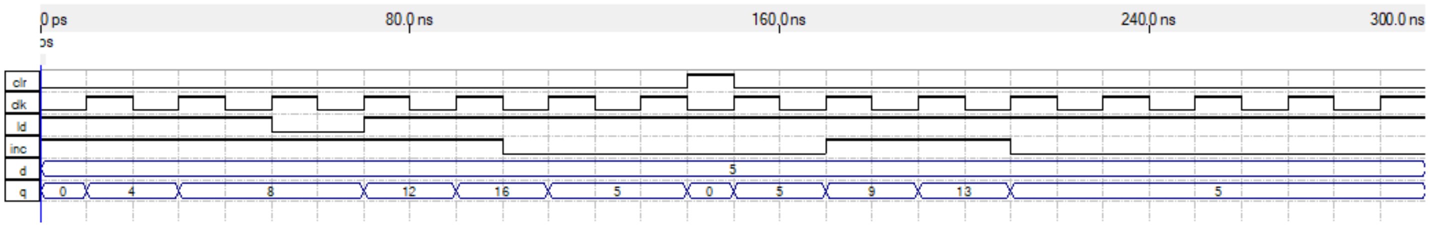
Please follow the following instructions to implement the overall Program Counter:

1. Create a new VHDL file in your "PC" project (File > New > VHDL File).
2. Type the following in the Text Editor and save the file as "pc.vhd":



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

entity pc is
port(
    clr :   in      std_logic;
    clk :   in      std_logic;
    ld  :   in      std_logic;
    inc :   in      std_logic;
    d   :   in      std_logic_vector(31 downto 0);
    q   :   out std_logic_vector(31 downto 0)
    );
end pc;

architecture Behavior of pc is
    component add
        port (
            A   :   in  std_logic_vector(31 downto 0);
            B   :   out std_logic_vector(31 downto 0)
        );
    end component;
    component mux2tol
        port (
            s       :   in std_logic;
            w0, w1  :   in  std_logic_vector(31 downto 0);
            f       :   out std_logic_vector(31 downto 0)
            );
    end component;
    component register32
        port(
            d       :   in  std_logic_vector(31 downto 0);
            ld      :   in  std_logic;
            clr     :   in  std_logic;
            clk     :   in  std_logic;
            Q       :   out std_logic_vector(31 downto 0)
        );
    end component;
    signal add_out : std_logic_vector(31 downto 0);
    signal mux_out : std_logic_vector(31 downto 0);
    signal q_out   : std_logic_vector(31 downto 0);
begin
    add0: add port map(q_out, add_out);
    mux0: mux2tol port map(inc, d, add_out, mux_out);
    reg0: register32 port map (mux_out, ld, clr, clk, q_out);
    q <= q_out;
end Behavior;
```

3. Start the compiler. Fix any errors and re-compile. Once the compiler compiles without any errors, move to the next step.
4. Create a new University Program VWF (File > New > University Program VWF).
5. Simulate "pc.vhd" and make sure that your simulation results match the results shown below.
6. Save the VWF file as "pc.vwf" in your "../COE608/Lab2/PC" folder and take a screenshot of your results.


0 ps
80.0 ns
160.0 ns
240.0 ns
300.0 ns
clr
clk
ld
inc
d
q