COE608 Computer Organization and Architectures
Winter 2017

Lab 2: Program Counter and Register Set Design

Due Date: Week 4 (Before the Start of your Lab Session)

Objectives

- To implement, simulate and test the Register Set required for the 32-bit CPU.
- To implement, simulate and test the 32-bit Program Counter (PC) required for the 32-bit CPU.

1. Register Design

The processor you are designing in this course requires several registers (A, B, AR, IR, C and Z), which are either 32-bit or 1-bit. The symbol for generic n-bit register is illustrated in Figure 1.

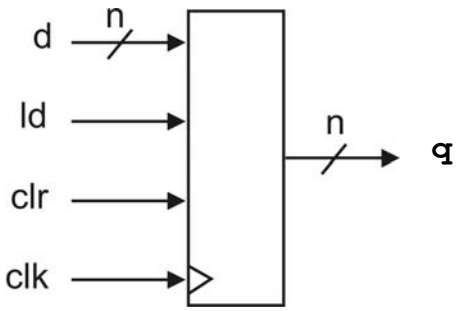

Figure 1: Block Diagram for n-bit Register.

Main inputs of the register include clock (clk), clear (clr), load/enable (ld) signals and an n-bit data (d). The n-bit output is denoted by (q).

You need to implement a 1-bit and a 32-bit register using VHDL. The entity for both registers must follow the format of the entity declaration shown in next page for the 32-bit register. The entity names of these registers should be register1 and register32.

```
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); -- input.
        ld      : IN STD_LOGIC;     -- load/enable.
        clr     : IN STD_LOGIC;     -- async. clear.
        clk     : IN STD_LOGIC;     -- clock.
        Q       : OUT STD_LOGIC_VECTOR(31 DOWNTO 0)); -- output.
END register32;

ARCHITECTURE description OF register32 IS
BEGIN

-- You fill in what goes here!!!!

END description;
```

2. Program Counter Design

The program counter (PC) is a key component of the processor, which is a 32-bit register with some additional control signals. The PC is illustrated in Figure 2.

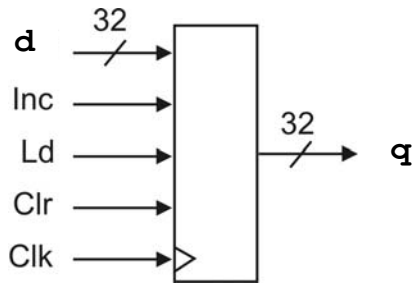

Figure 2: Program Counter

The PC has a 32-bit output q and several inputs. Inputs include the clock (clk) and clear (clr) to reset the PC to zero. Also included is a 32-bit input (d) used to set the PC to non-zero values. A load/enable (ld) input load the program counter with input (d) at the rising clock edge. Finally, an increment (inc) input is required for incrementing the PC by 4 during instruction execution.

The internals of the PC are illustrated in Figure 3.

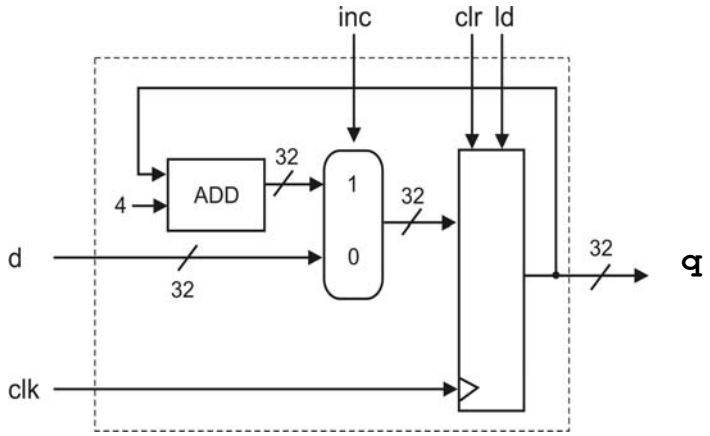

Figure 3: 32-bit Program Counter Internal Structure

The *inc* input is used to select between d or pc+4. The clr, ld and clk are directly connected to an internal 32-bit register, which is part of the PC comes from. You need to implement a 32-bit PC using VHDL. The entity declaration for PC is given below, which must be followed.

```
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       : INOUT STD_LOGIC_VECTOR(31 DOWNTO 0));
END pc;

ARCHITECTURE description OF pc IS
BEGIN

-- you fill in what goes here!!!

END description;
```

Notice that the PC output is declared as "INOUT". This is necessary since, looking at the internal block diagram for the PC, the signal is used both as input and output.

3. What to Hand In

You should hand in the following before your lab starts in week four in order to receive credit for the laboratory:

- A hard-copy listing of all source listings of your VHDL code
- Waveform simulations of all your registers and your program counter. Use a grid size of 100ns, and fit to one page, landscape format.

Demonstrate the implementation and simulation of Registers and PC. Your lab instructor may quiz you about your VHDL implementation for this lab.