BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI
First Semester 2014-2015
CS F342 Computer Architecture

Lab Sheet - 8 **Topic: Pipeline Design**

Learning Objectives:

i) **Modeling 3-stage pipeline design shown below in Verilog**
ii) **Designing modules for each hardware component and some support modules occurring in the pipeline design.**
iii) **Implementation of the modules in Verilog**
iv) **Integrating these modules**
v) **Writing test bench**

Today we have to implement a simple 3-stage pipeline design described below using Verilog. To accomplish this, first identify the hardware components of the pipeline design and write separate modules for each component. Integrate these modules to realize the complete pipeline design circuit. Test the correctness of the complete implementation by supplying a sequence of instructions as input.

I. Modeling Pipeline Design in Verilog:

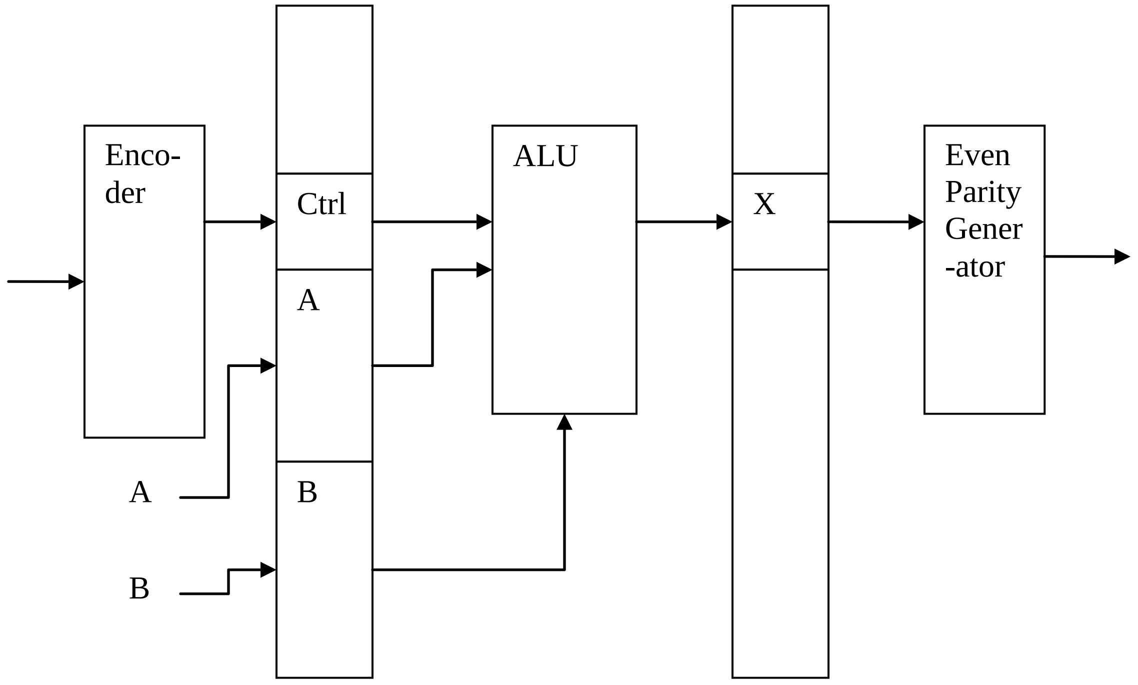

Fig.1. 3-Stage Pipeline Design Circuit

Description of 3-stage Pipeline Desgin:

It consist of three pipeline stages: fetch, execute and generate parity and two pipeline registers.
Let us consider, there are 8 instructions in ISA each require 3-bit opcode. The encoder takes **8-bit function code** of the instruction as input and produces a **3-bit op-code (or Ctrl) as output.** The ALU shown in the above design can perform 8 different operations on the 4-bit operands **A** and **B** based on the 3-bit **opcode (or Ctrl)** and produces a **4-bit output X**. The output X is given as input to the Even Parity Generator which genearates the parity.

1. ctrl = 3'b000 : add
2. ctrl = 3'b001 : sub
3. ctrl = 3'b010 : xor
4. ctrl = 3'b011 : or
5. ctrl = 3'b100 : and
6. ctrl = 3'b101 : nor
7. ctrl = 3'b110 : nand
8. ctrl = 3'b111 : xnor

In first stage (Fetch stage (F)), following operations are done:

1. 8-bit function code of the operation to be performed in the instruction is given as input to the encoder, encoder encodes the function code and outputs the corresponding 3-bit op-code. This op-code is stored in the first pipeline register.
2. Two 4-bit operands **A** and **B** are stored in first pipeline register which will be used in second stage.

In second stage (Execute stage (E)), following operations are carried out:

1. Op-code, operand A and operand B are read from the first pipeline register and given as input to the ALU.
2. Based on the op-code, ALU performs operation on two operands A and B.
3. Output of ALU is stored in second pipeline register.

In third stage (Generate Parity (GP)), following operations are done:

1. Output of ALU **X** is read from second pipeline register and given as input to Even Parity Generator.
2. Even parity generator generates parity of the input 4-bit number.

II. Designing modules for each hardware component

Write separate modules for encoder, ALU and Even Parity Generator. Input and output to each of the modules are already explained above. Write a module which will integrate these modules to realize the pipeline design circuit.
Make use of behavioural modeling for encoder, dataflow modeling for ALU and Even Parity Generator. Write a test bench to test different instructions given in ISA for any two operand values serially.

****