```vhdl
--------------------------------------------------------------------------------
--
-- Title       : Pipeline_tb
-- Design      : Pipeline
-- Author      : Aaron Lin and Hang Chen
-- Company     : Stony Brook University
--
--------------------------------------------------------------------------------
--
-- File        : c:\my_designs\Pipeline\Pipeline\src\Pipeline_tb.vhd
-- Generated   : Fri May  1 18:01:50 2020
-- From        : interface description file
-- By          : Itf2Vhdl ver. 1.22
--
--------------------------------------------------------------------------------
--
-- Description : Testbench for the top level design pipeline. This loads the binary formatted program into the Instruction Buffer
--  and provides the clock for
--
--------------------------------------------------------------------------------
library work;
use work.all;
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.NUMERIC_STD.all;
use std.textio.all;
use ieee.std_logic_textio.all;

entity Pipeline_tb is
end Pipeline_tb;

--}} End of automatically maintained section

architecture Pipeline_tb of Pipeline_tb is
signal Instruction_Index : std_logic_vector(5 downto 0);
signal CLK, W_En_inWB, Forward_Mux1, Forward_Mux2, Forward_Mux3: std_logic;
signal Instruction_inIF, Instruction_inID, Instruction_Data : std_logic_vector(24 downto 0);
signal ALU_Output, Input_rs3, Input_rs2, Input_rs1, Data_inWB : std_logic_vector(127 downto 0);
signal ALUop_inID, ALUop_inEX : std_logic_vector(7 downto 0);
constant period : time := 20ns;
begin

    uut : entity Pipeline port map (Instruction_inIF => Instruction_inIF, Instruction_inID => Instruction_inID, ALUop_inID => ALUop_inID,
        ALUop_inEX => ALUop_inEX, Data_inWB => Data_inWB, Instruction_Index => Instruction_Index, Instruction => Instruction_Data,
        CLK => CLK, ALU_Output => ALU_Output, W_En_inWB => W_En_inWB, Input_rs3 => Input_rs3, Input_rs2 => Input_rs2, Input_rs1 => Input_rs1,
        Forward_Mux1 => Forward_Mux1, Forward_Mux2 => Forward_Mux2, Forward_Mux3 => Forward_Mux3);
    store : process
    file F : TEXT open READ_MODE is "ESE345_Assembler.txt";
    variable L : LINE;
    variable temp : std_logic_vector(24 downto 0);
    variable count : unsigned(5 downto 0) := "000000";
    begin
        if not ENDFILE(F) then
            READLINE(F, L);
            READ(L, temp);
            Instruction_Data <= temp;
            Instruction_Index <= std_logic_vector(count);
            count := count + 1;
        end if;
        wait for period;
    end process;

    clk_process : process
    begin
        CLK <= '0';
        wait for period/2;
        CLK <= '1';
        wait for period/2;
    end process;
end Pipeline_tb;
```