COMP3211/9211 Computer Architecture
2004 S2 Laboratory Exercise 3 (VHDL3)
Instruction Fetch Unit (Draft 29/08)

1 Goal

The goal of this exercise is to introduce datapath design by considering the problem of implementing an Instruction Fetch unit for a processor. The successful implementation of this component also serves as a foundation for implementing this component in Assignment 1.

2 Specification

You are required to implement and test the design of the Instruction Fetch Unit illustrated in Figure 1. This unit fetches the next sequential instruction from instruction memory if the current instruction is a non-branching instruction. However, if the current instruction is a branching instruction, then if the branch condition is satisfied, a signed offset is added to the next program counter value to obtain the next instruction address. Otherwise, if the branch condition is not satisfied, the next instruction in sequence is fetched.

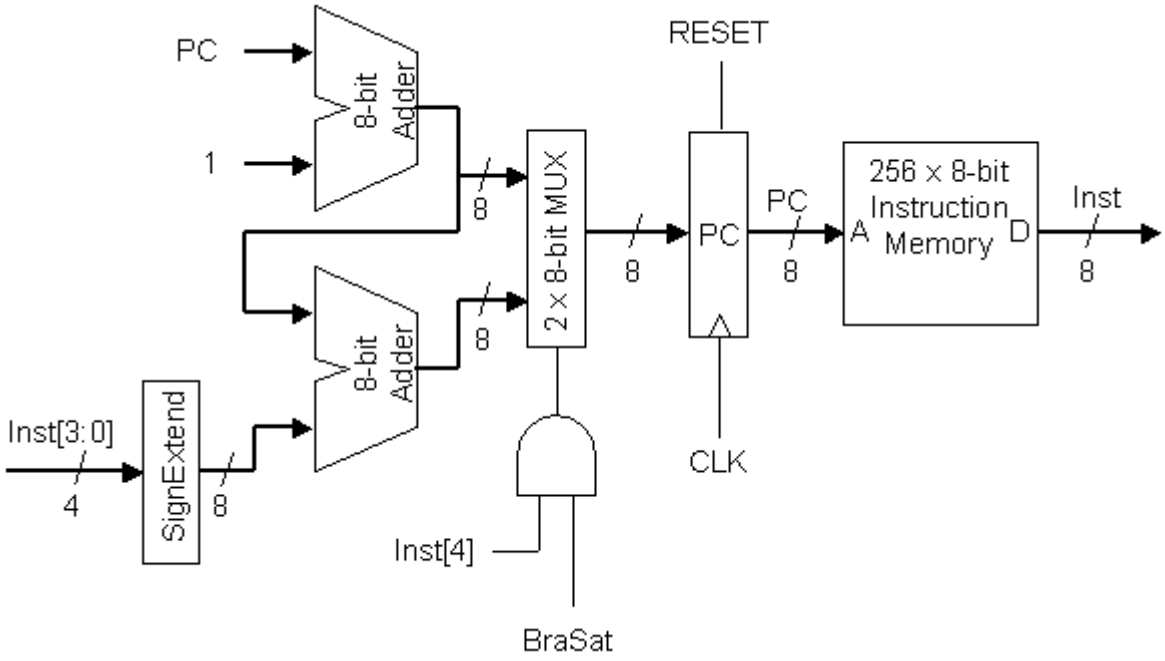

Figure 1. Instruction Fetch Unit

In order to implement the Instruction Fetch Unit, it is recommended that you make use of previously supplied or constructed models to implement the Instruction Memory, the 8-bit Adders, and the Program Counter (PC) Register. You will need to modify the existing components and implement a suitable sign extension unit and multiplexer.

The Instruction Fetch Unit is largely self-contained. However, the unit requires some additional external signals. These are a clock signal, `CLK`, an asynchronous `RESET` signal that clears the PC register when asserted, and a control signal, `BraSat`, which is asserted by the user to simulate "satisfaction" of the branch condition.

Instructions are to be stored as 8-bit words of the form `xxxx yyyy`. The most significant 4 bits of the instruction word represent the opcode. Only two codes are to be used: 0000, representing non-branching instructions, and 0001 for conditional branch instructions. The least significant 4 bits have no meaning for non-branching instructions and encode a signed 2's complement offset that is to be added to the next PC value if the `BraSat` input is high.

3 Procedure

1. Build and test a VHDL model of a SignExtend unit as needed by the Instruction Fetch Unit (IFU).
2. Build and test a VHDL model of a 2 x 8-bit MUX as used by the IFU.
3. Modify the register and memory components studied in Lab 1 to suit the purposes of the IFU.
4. Load the memory with at least 16 instruction words that will allow you to test both forwards and backwards branching.
5. Define a suitable entity for the IFU.
6. Describe a structural implementation of the IFU using components from above and your adder from Lab 2.
7. Test your IFU.

4 Hand-in

There is no hand-in for this lab. Instead, you are to demonstrate your test from Step 7. of the above procedure. You may do this during your 4th lab session. Successful demonstration of a meaningful IFU test is worth 1 mark of your final assessment.