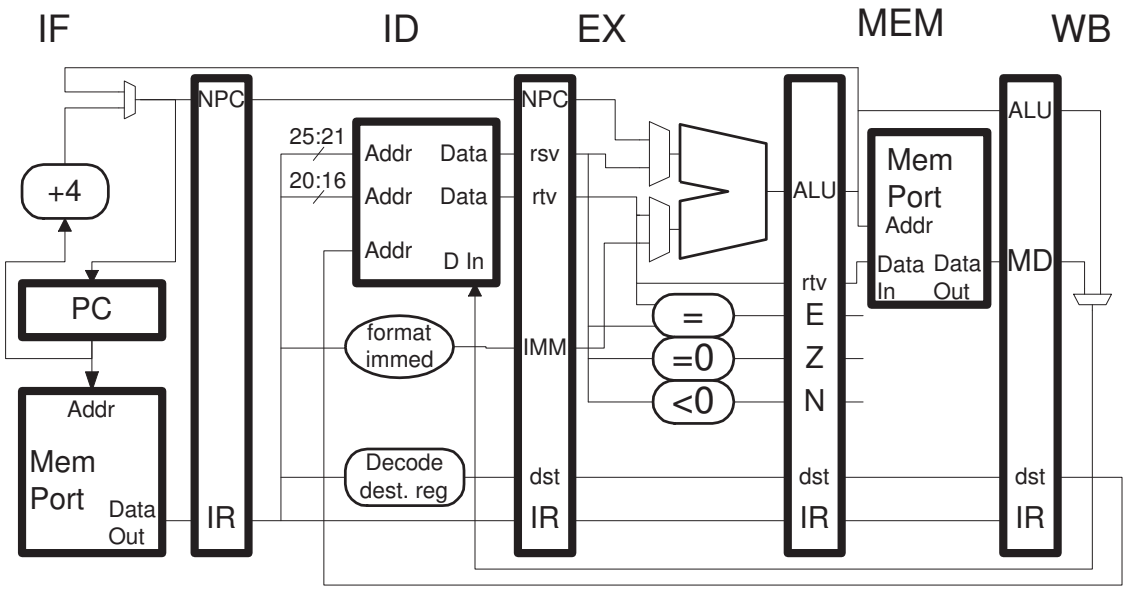
LSU EE 4720 **Homework 2** **Due: 17 September 2010**

Problem 1: Consider the execution of the code fragments below on the illustrated implementation.



- A value written to the register file can be read from the register file in the same cycle. (For example, if instruction A writes `r1` in cycle x (meaning A is in `WB` in cycle x) and instruction B is in `ID` in cycle x, then instruction B can read the value of `r1` that A wrote.)
- As one should expect, the illustrated implementation will execute the code correctly, as defined by MIPS-I, stalling and squashing as necessary.

```
LOOP:
 lw r3, 0(r1)
 add r4, r4, r3
 bne r1, r2  LOOP
 addi r1, r1, 4
 xor r7, r8, r3
 sw r4, 16(r5)
```

(a) Show a pipeline execution diagram for this code running for at least two iterations.

- Carefully check the code for dependencies, including dependencies across iterations.
- Base timing on the illustrated implementation, pay particular attention to how the branch executes.

(b) Find the CPI for a large number of iterations.

(c) How much faster would the code run on an implementation similar to the one above, except that it resolved the branch in `EX` instead of `ME`? Explain using the pipeline execution diagram above, or using a new one. An answer similar to the following would get no credit because "should run faster" doesn't say much: *A resolution of a branch in* `EX` *occurs sooner than* `ME` *so the code above should run faster.*. Be specific, and base your answer on a pipeline diagram.

Problem 2: *Apologies in advance to those tired of the previous problem.* Consider the execution of the code below on the implementation from the last problem. The code is only slightly modified.

(*a*) Show a pipeline execution diagram for this code, and compute the CPI for a large number of iterations. It should be faster.

```
LOOP:
 add r4, r4, r3
 lw r3, 0(r1)
 bne r1, r2  LOOP
 addi r1, r1, 4
 add r4, r4, r3
 sw r4, 16(r5)
```

(*b*) How much faster would the code above run on the implementation that resolves branches in `EX` (from the previous problem)?

(*c*) Suppose that due to critical path issues, the resolve-in-`EX` implementation had a slower clock frequency. Let $\phi_{\texttt{ME}}$ be the clock frequency of the resolve-in-`ME` implementation (the one illustrated), and $\phi_{\texttt{EX}}$ be the clock frequency of the resolve-in-`EX` implementation. Find $\phi_{\texttt{EX}}$ in terms of $\phi_{\texttt{ME}}$ such that both implementations execute the code fragment above in the same amount of time. That is, find a clock frequency at which the benefit of a smaller branch penalty is neutralized by the lower clock frequency *on the code fragment above*.