Inf2C Computer Systems

Tutorial 4, Week 9

1. **Multi-cycle processor**
   Discuss the steps in executing the `jal`, the `lw` and the `add` instruction in the multi-cycle datapath presented in Figure 1.

   Now assume that an access to the register file takes 6.5ns, an access to the memory takes 100ns and the ALU delay is 6ns. Assume also that the instruction's execution stalls when it waits for a resource to produce a result. If we were to optimize this simple processor, which should be the component we should spend our main design efforts on, what could we do to improve it?

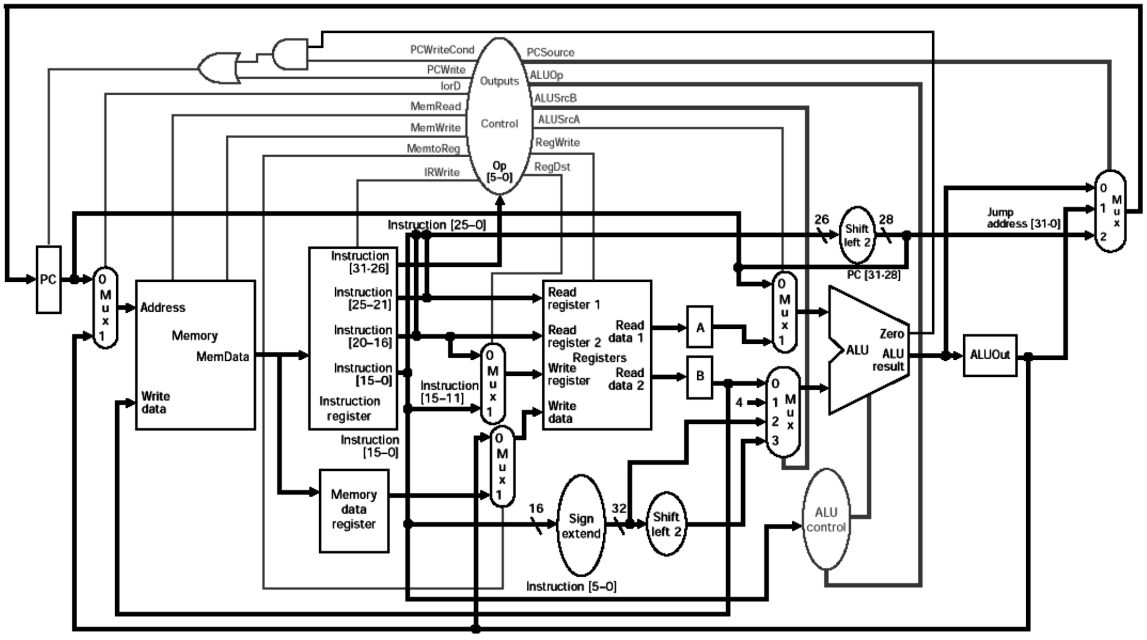

Figure 1: MIPS Datapath

2. **Direct-mapped cache**
Consider a simple direct-mapped cache with 4 lines, each containing 2 bytes of data and a tag field of appropriate size for 6-bit addresses. Assume that the cache is initially empty and that memory is byte-addressable.

For the following pattern of cache accesses:

Access Number	Address
1	1 0 1 1 0 0
2	0 0 1 0 1 1
3	0 1 1 0 0 1
4	0 1 0 0 0 1
5	1 0 1 1 0 0
6	0 1 1 0 0 1
7	1 0 0 1 1 0

draw up a table of form

Access Number	Tag	Index	Line 0	Line 1	Line 2	Line 3	Hit/Miss
1							
2							
3							
4							
5							
6							
7							

that summarises the contents of the cache *after* each access. In the *Tag* and *Index* columns, write the decimal values of the tag and index components of each access address. Then, in each row, if Line i is occupied with the memory block with tag t, write the decimal value of t in the column for Line i, and record in the Hit/Miss column an H or an M to indicate whether the access is a hit or a miss. If a line is empty after some access, leave the entry for that line and access number blank.

3. **Fully-associative cache**
Consider again a cache with 4 lines, each containing 2 bytes of data, and assume addresses are 6 bits wide, but this time assume the cache is fully-associative. With the same pattern of accesses as before, draw up a table of form

Access Number	Tag	Line 0	Line 1	Line 2	Line 3	Hit/Miss
1						
2						
3						
4						
5						
6						
7						

that summarises the contents of the cache *after* each access.

Assume that empty cache lines are filled in order and assume an LRU (least recently used) replacement policy. Note how the table would change if a FIFO (first in first out) replacement policy were used instead.