Building a Simple MIPS Datapath

Tom Kelliher, CS 220

Nov. 11, 2011

1 Administrivia

Announcements

Assignment

Read 4.4.

From Last Time

Project day.

Outline

1. Quick introduction to digital logic.
2. Overview of the MIPS implementation.

Coming Up

Completing the datapath.

2 Quick Introduction to Digital Logic

(Register for CS 240 for the complete introduction.)

2.1 Combinational Logic

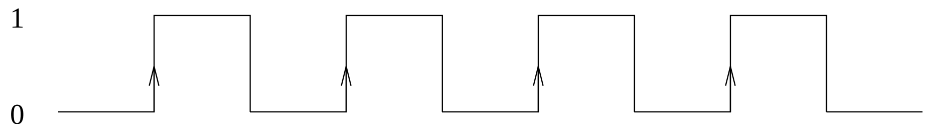
1. Basic logic gates: Inverter, AND, OR.
2. One bit full adder:
   (a) From truth table to gates.

   (b) Multiple bit adders.
3. 2-1 mux:
   (a) What is it and why do we need it?

   (b) From truth table to gates.

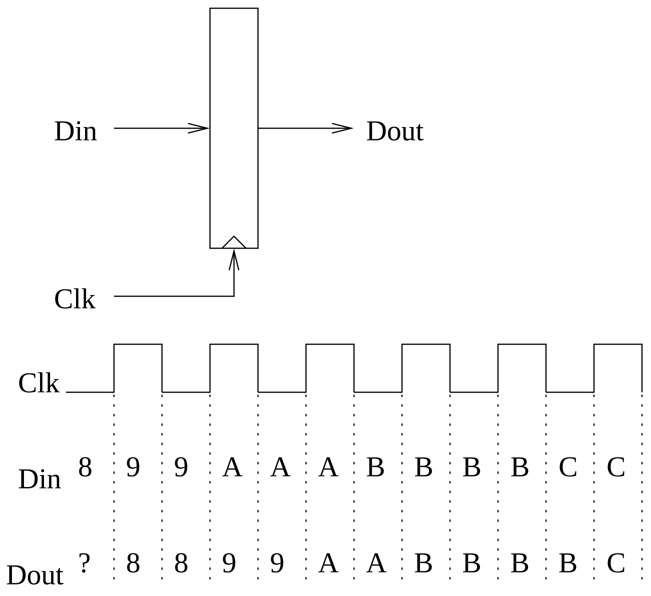
2.2 Sequential Logic and Examples

1. The clock signal.

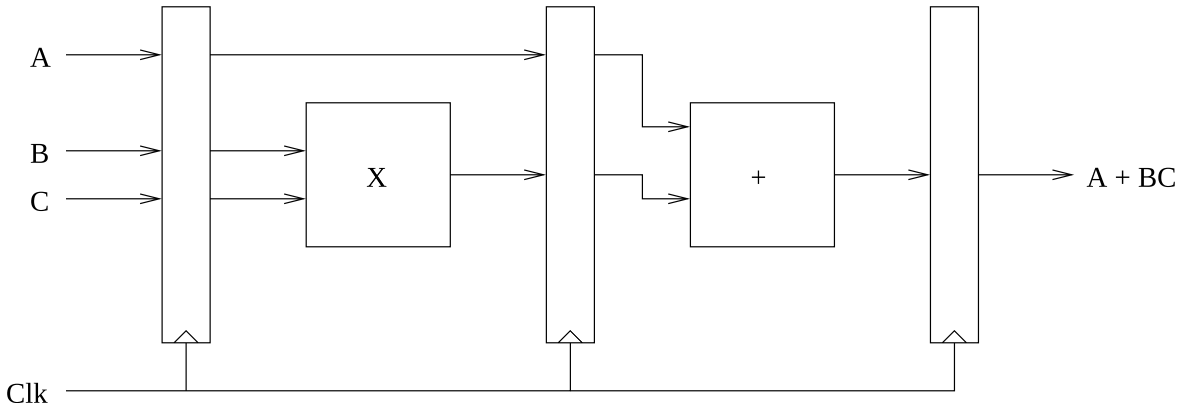


   Sequential elements are *rising edge sensitive*.

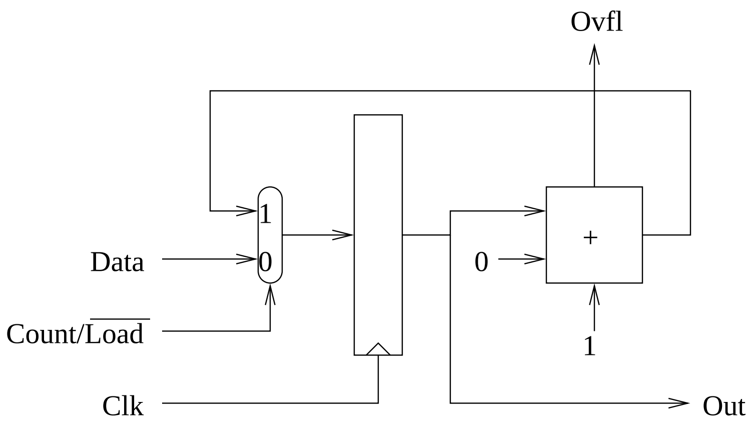
2. Clocking registers.

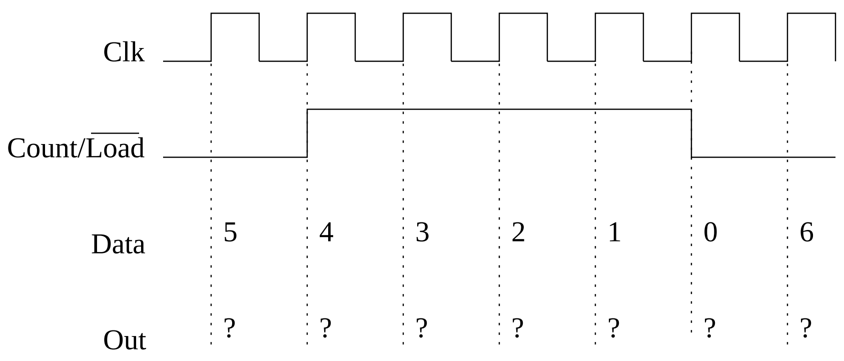


3. A simple pipeline.



4. A loadable counter.





5. Design exercise: four entry register file. Components: loadable register, 4-1 mux, decoder with enable.

3 Overview of the MIPS Implementation

Instruction set subset we'll consider:

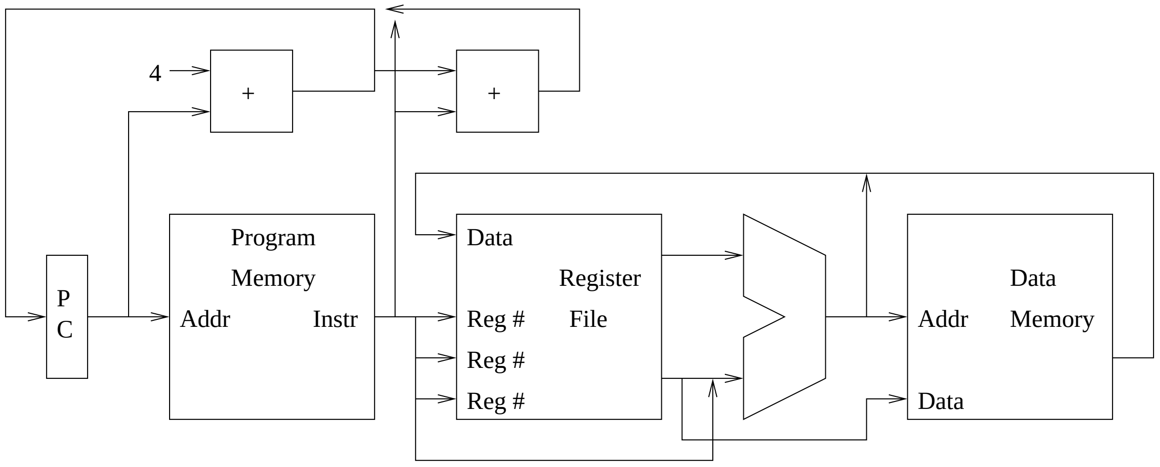
1. `lw`/`sw`.
2. `add`, `sub`, `and`, `or`, `slt`.
3. `beq`.
4. `j`.

The general instruction cycle:

1. Instruction fetch.
2. Instruction decode.
3. Register fetch.
4. Operate.
5. Register store or memory operation.

How do the steps of the cycle fit each of the three instruction classes: arithmetic-logic, memory reference, branch?

A high level view of the implementation, in view of the instruction cycle:



1. Datapath only.

2. Is everything we need for our instructions here?

3. Why two memories?