Homework 4

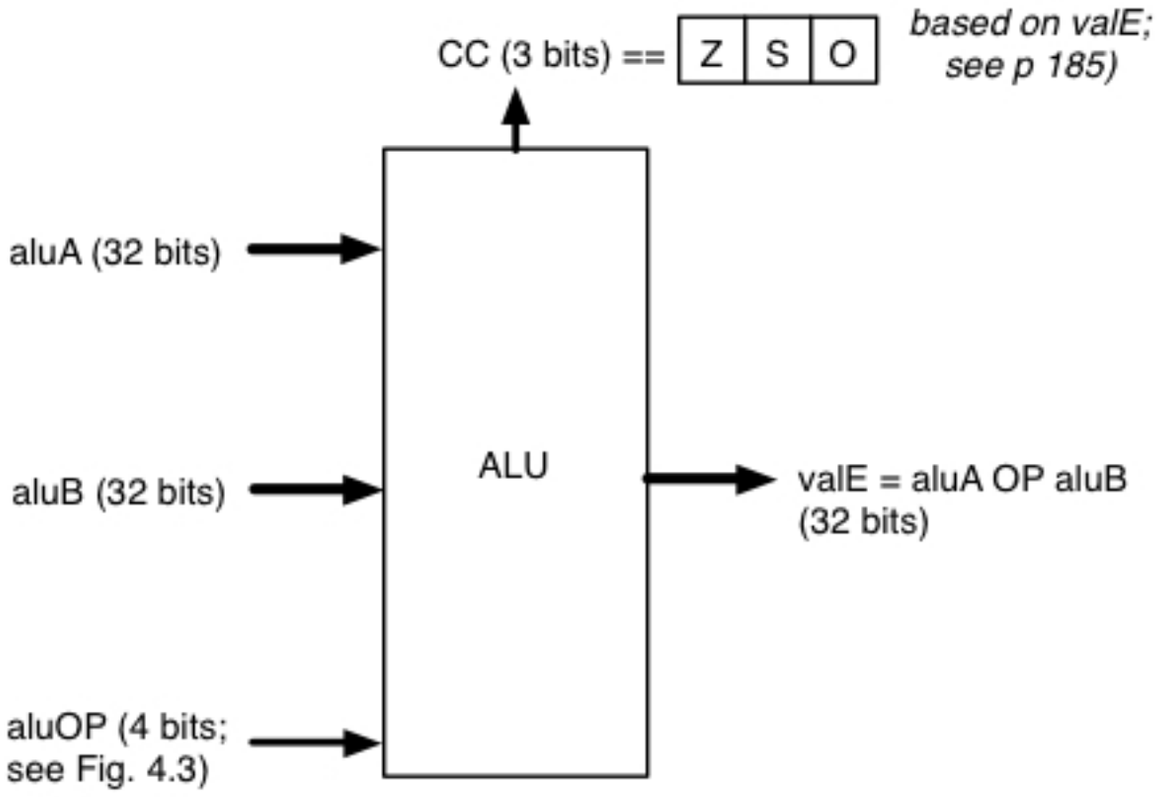
Due Apr 27 by 11:59pm **Points** 100 **Submitting** a file upload **File Types** circ, pdf, txt, zip, and gz
Available after Apr 18 at 12am

In this assignment, you will apply principles we've been covering in class to build a few more pieces of your Y86.

Q1: ALU (20pts)

(Exercising combinatorial logic and binary arithmetic)

Build the Arithmetic Logic Unit for your Y86. It will fit better into your circuit if it looks like this:



Don't use any internal registers; the outputs should follow directly from whatever the inputs are.

In accordance with our policy, you may use baked-in addition (since we've built adders in class) and equality-testing, but you may not use the baked-in subtractors or greater-than/less-than components (unless of course you build one first).

Memory Shims

(Exercising RAM principles and FSM/sequential logic design)

As we discussed in class:

- RAM is typically byte-addressable
- But modern CPUs (including the textbook's development of the Y86) want to access RAM on other granularities.

In these next questions, you'll build shims that glue the behavior the Y86 expects to byte-oriented LogiSim RAM.

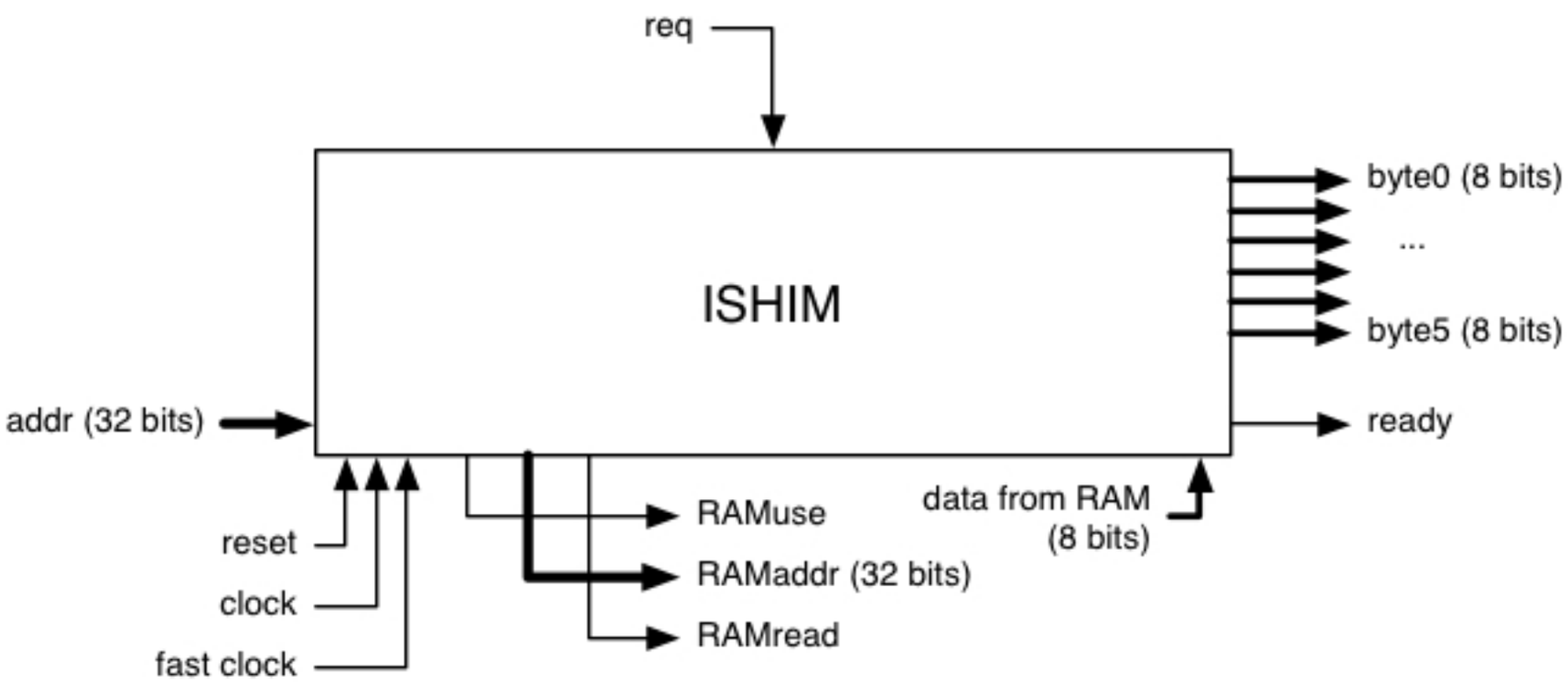
Use LogiSim's byte-wide RAM with "separate load and store ports." (Try 256 or 1024 bytes.) Feel free to make use of the "str," "sel," and "ld" inputs---since Monday's demos will fulfill our principle of "don't use things until you see how they are built." (Yes.... in your Y86, these shims will be working with the *same* RAM.)

The shims will have three common inputs:

- an active-high "reset"---to force everything to a clean, known initial statee
- a "clock," driving everything in your Y86
- and (as a secret weapon for this homework), a "fast clock"--- e.g., 16 pulses for every pulse the rest of the system sees.

Q2: Instruction Memory Shim (40pts)

When getting instructions, the Y86 will want to fetch six consecutive bytes from a given address. Build a shim that glues this behavior to byte-oriented LogiSim RAM.



At clock rise, when "req" is asserted at and "addr" contains N, the shim will fetch the bytes at addresses N+0, N+1, ...N+5, store them in internal registers, and make their contents available as output.

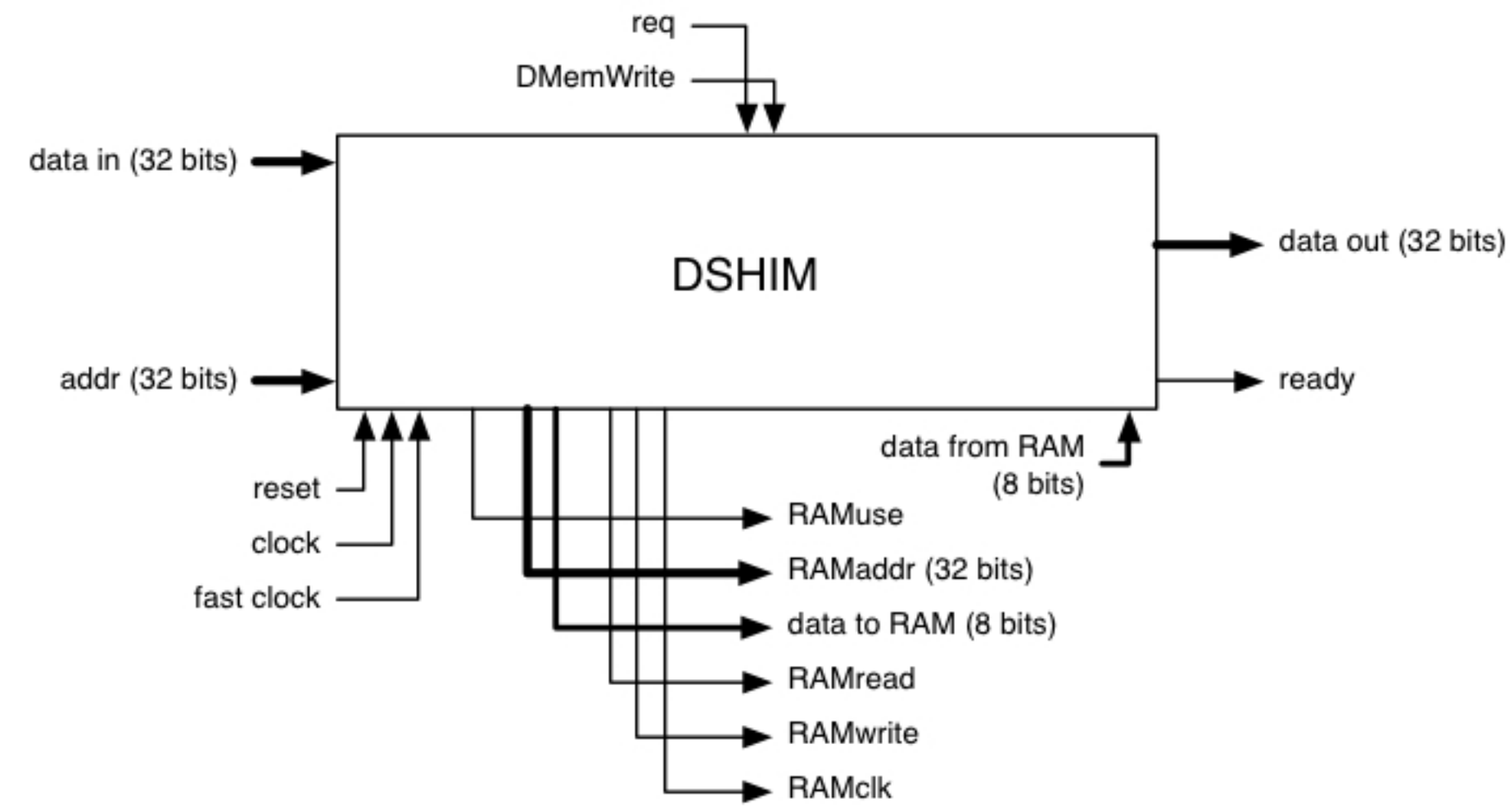
When the shim's byte outputs are stable and valid, the shim asserts "ready."

The shim works with RAM by asserting "RAMuse" when it's doing something with RAM and "RAMread" when it would like to read from "RAMaddr."

If the shim is not working with RAM, RAMaddr should be tri-stated.

Q2: Data Memory Shim (40pts)

When working with data, the Y86 will want to either read or write four consecutive bytes from a given address. Build a shim that glues this behavior to byte-oriented LogiSim RAM.



At clock rise, when "req" is asserted at and "addr" contains N, the shim will begin operation.

If "DMemWrite" is asserted, the shim will write the 4-byte word at "data in" to RAM locations N+0,...,N+3 (little endian). If "DMemRead" is asserted, the shim will fetch the little-endian 4-byte word at N+0,...,N+3, store them in an internal register, and output this value at "data out." When this output is stable and valid, the shim asserts "ready."

The shim works with RAM by asserting "RAMuse" when it's doing something with RAM, "RAMread" when it would like to read from "RAMaddr"; and "RAMWrite" when it would like to write there. The right will happen at the rising edge of "RAMclk."

If the shim is not working with RAM, RAMaddr should be tri-stated.

hw 5 rubric		
Criteria	**Ratings**	**Pts**
Q1: Correctness		16 pts
Q1: Clarity, Simplicity, Neatness		4 pts
Q2 - ISHIM: Core Functionality		24 pts
Q2 - ISHIM: Testing Subcircuit		2 pts
Q2 - ISHIM: Subcircuit Appearance		2 pts
Q2 - ISHIM: Clarity, Simplicity, Neatness		6 pts
Q2 - ISHIM: The Details		6 pts
Q2 - DSHIM: Core Functionality		24 pts
Q2 - DSHIM: Testing Subcircuit		2 pts
Q2 - DSHIM: Subcircuit Appearance		2 pts
Q2 - DSHIM: Clarity, Simplicity, Neatness		6 pts
Q2 - DSHIM: The Details		6 pts
		Total Points: 100