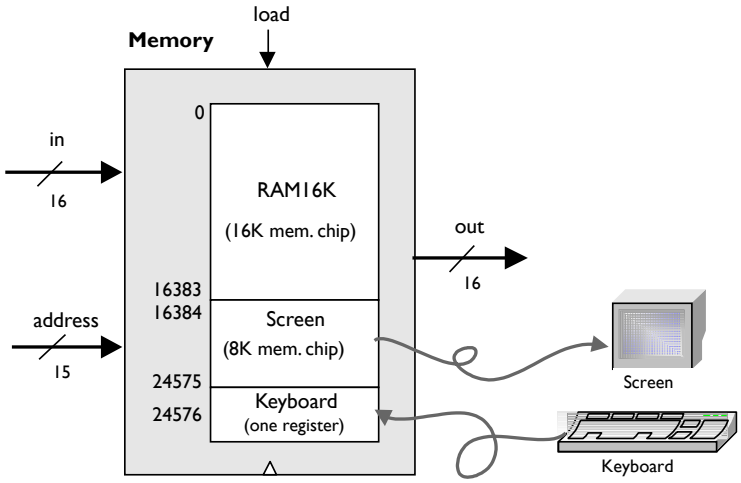
Tips on Building the CPU and Memory

Memory



Depending on the address, the input should be loaded into either the RAM or the Screen. If the address is less than 16384, the input goes to the RAM. If the address is greater than 16383 and less than 24576, the input foes to the RAM.

Consider bits 14 and 13 of the address. They have values of 16384 and 8192 respectively. There are four combinations of bits and from this we can determine which of RAM, Screen and Keyboard are selected.

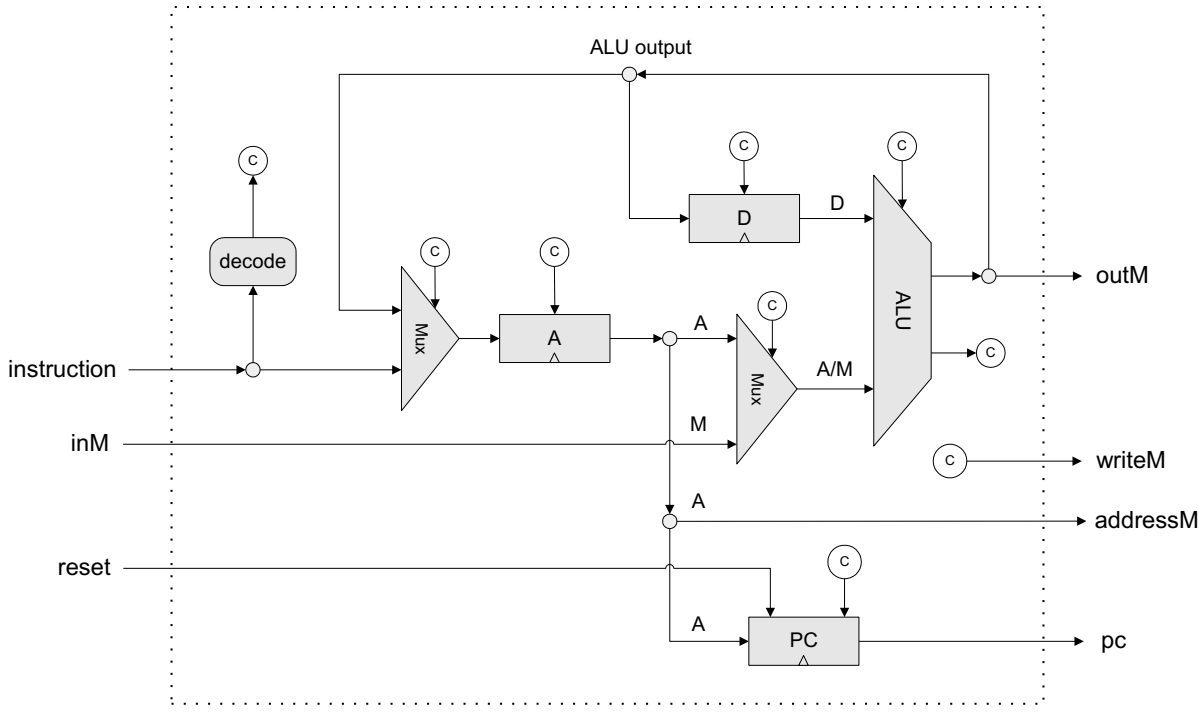
00 - address must be less than 8192 – RAM
01 - address must be less than 16384 – RAM
10 – address must be greater or equal to 16384 – Screen
11 – address must be greater or equal to 24576 - Keyboard

Suggestion:

Use a DMux4Way and bits 13 and 14 of the address to select whether to load the RAM or Screen (by piping ‘true’ to the relevant load bit).

Use a Mux4Way and bits 13 and 14 of the address to select whether to connect the RAM, Keyboard or Screen to the output.

CPU



Instruction Decoder

This is just simple combination logic that check the bits on the instruction bus. You can check bit 15 of the instruction to determine whether a A (0) or C instruction (1) is to be executed.

A NOT gate added to bit 15 can then be used to create the control signal that an A instruction is being decoded.

When it is a C instruction, the 'a' bit is used on the Mux to select between A and M on the ALU input. The six control bits can be feed to the control bits of the ALU.

The 3 destination bits and jump bits are discussed below.

It is sensible to AND each of the instruction bits with bit 15 before using in the rest of the circuit. Therefore, when it is an A-instruction (bit 15 = 0), the outputs of the C-instruction decoder will all be zero. Consider an A instruction, then bits 0 to 14 are all randomly 0 or 1 depending on the value in A. If these are connected to the ALU (via the control bits), the ALU will end up doing a random operation.

A-Instruction Implementation

There are two ways to load the A register. Either through an A-instruction (e.g @15) or by the destination being A e.g. (A = -1)

We can use a Mux to select where the input comes from (ALU output in the event of destination being A or instruction bus in the event of an A-instruction). The relevant destination bit **or** (not) bit 15 of the instruction should be used to set the load bit of the A register.

Destination Specification

The ALU output is connected to the A register input, D register input and outM (to RAM).

The destination bits control where the output of the ALU goes to. d1 selects the A register, d2 selects the D register and d3 the RAM. d1 and d2 therefore control the load bits of the relevant registers and d3 is the 'writeM' output of the CPU.

Jump Specification

When a jump occurs we load the value stored in A into the Program Counter. To check whether a jump occurs, we need to check whether a jump has been specified in the C instruction and whether the appropriate condition has been met.

For example, JGT means 'jump if greater than zero'. We therefore need to check the ALU output flag. The negative and zero flags have already been implemented. We can use these to create a positive flag (if not negative and not zero, it must be positive).

If the JGT bit is set (j3) and the positive flag is set, we must do a jump and can load the PC. You can create similar circuits for negative and zero and or them together to feed in to the load bit of the PC.

The reset of the CPU connects directly to the reset of the PC.

The PC should be put into increment mode by default (by setting the appropriate input to be true).