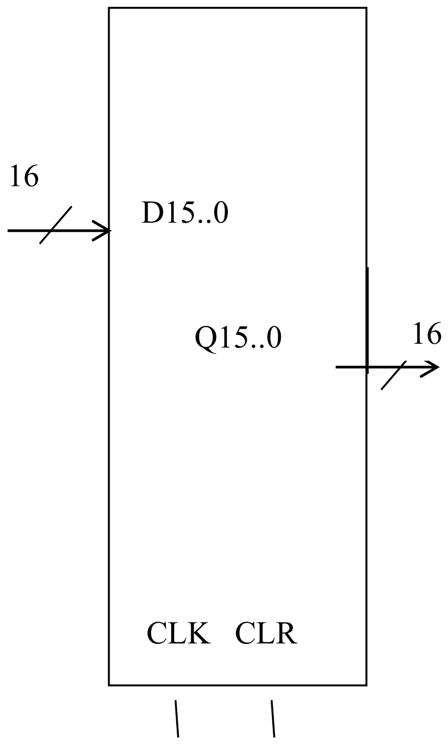
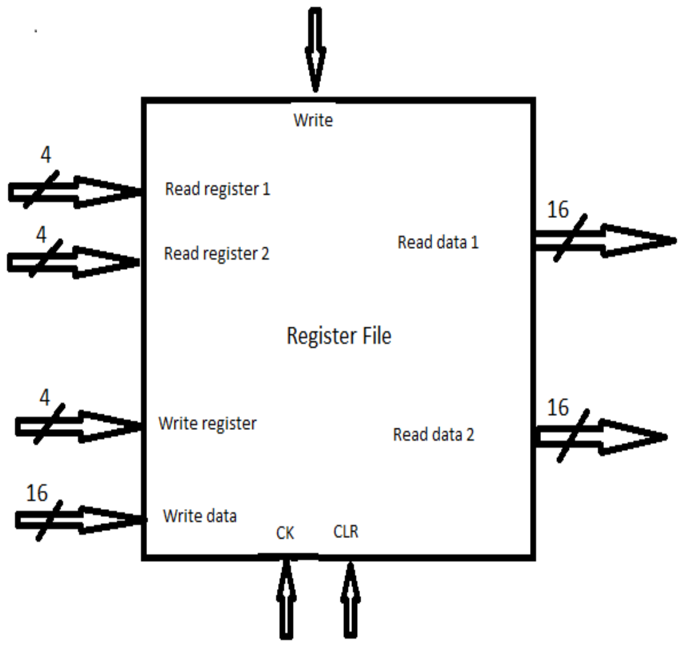
CS240 Laboratory 5
Sequential/Memory Circuits
and
Introduction to Datapath

Circuits using Flip-flops

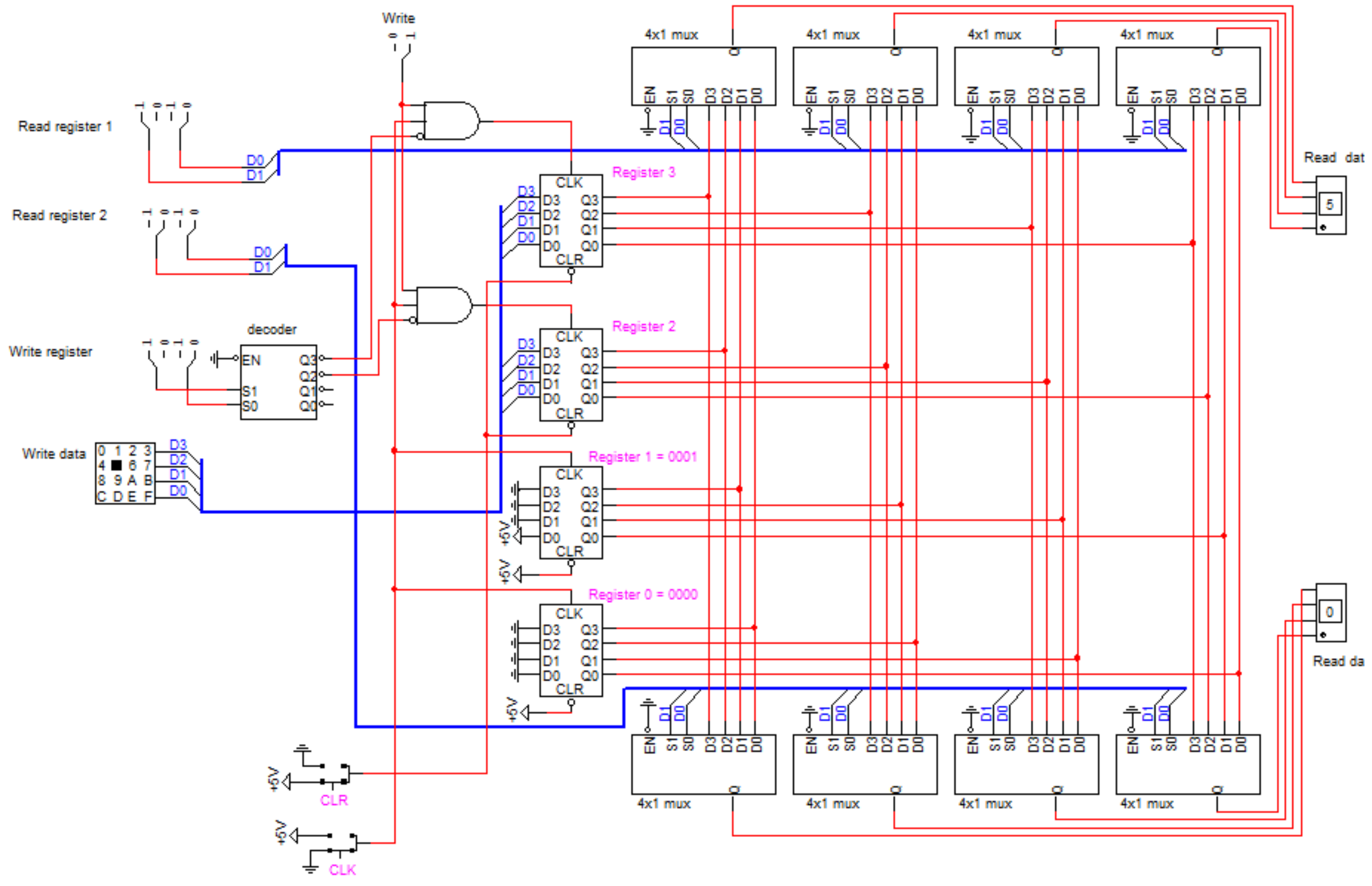
Register - n-bit memory, uses *n* flip-flops, and shared *clock* and *clear* inputs



Register File set of registers



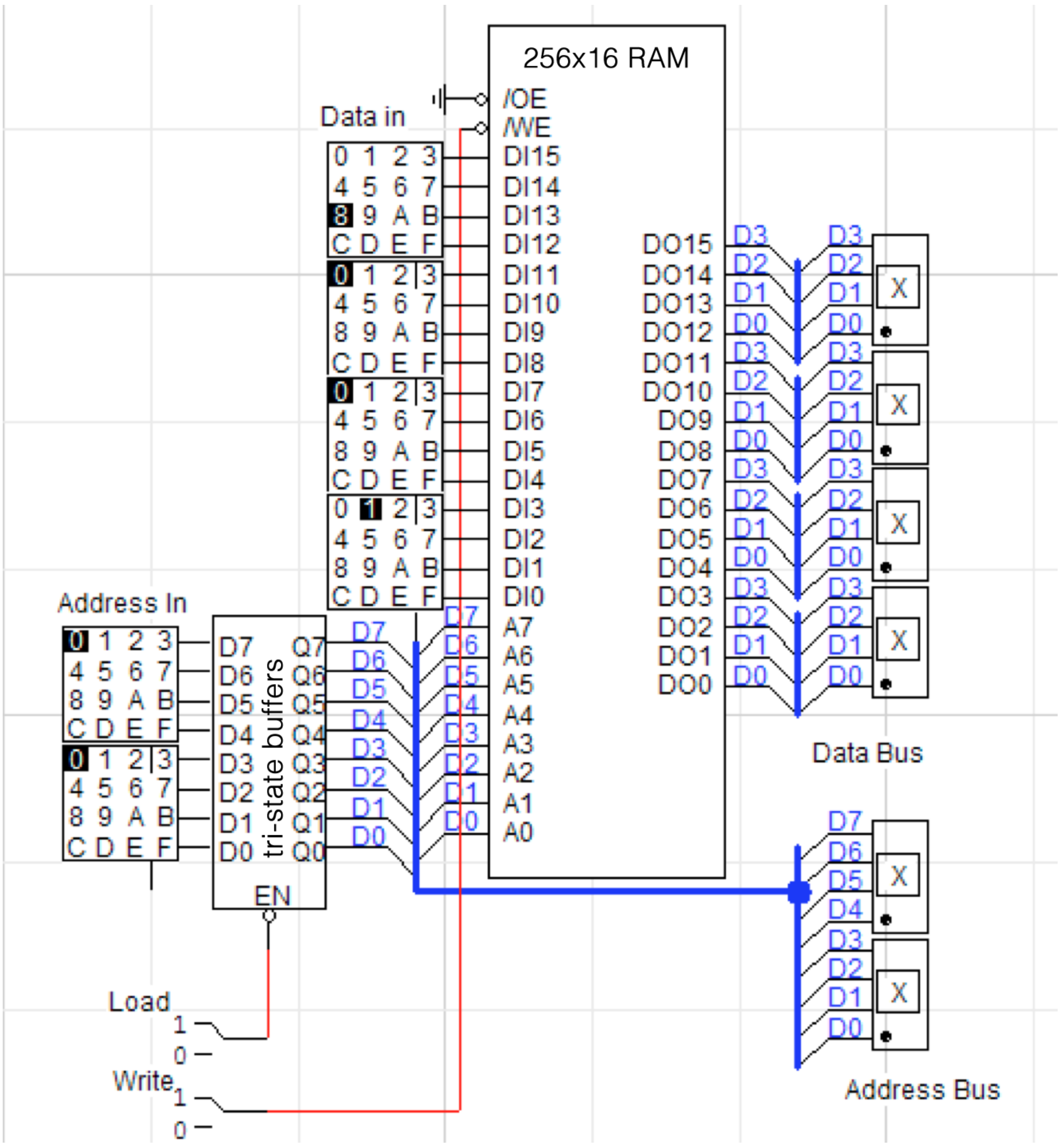
- **Write** is the write control signal.
- **Write register** is the number of a register to be written with a new value
- **Read register 1** and **2** indicate which 2 registers can be read at data ports **Read data 1** and **Read data 2** at any given time
- clear and clock (**CLR** and **CLK**) are shared by all the 16 registers.



- 2 sets of 4 x 1 multiplexers select which 2 registers are currently being output at the two read ports.

- A decoder uses the write register number to select which of the 4 registers will receive a new value on a write.

RAM memory contains multiple flip-flops, organized into n-bit words, where each word can be accessed through use of an address:

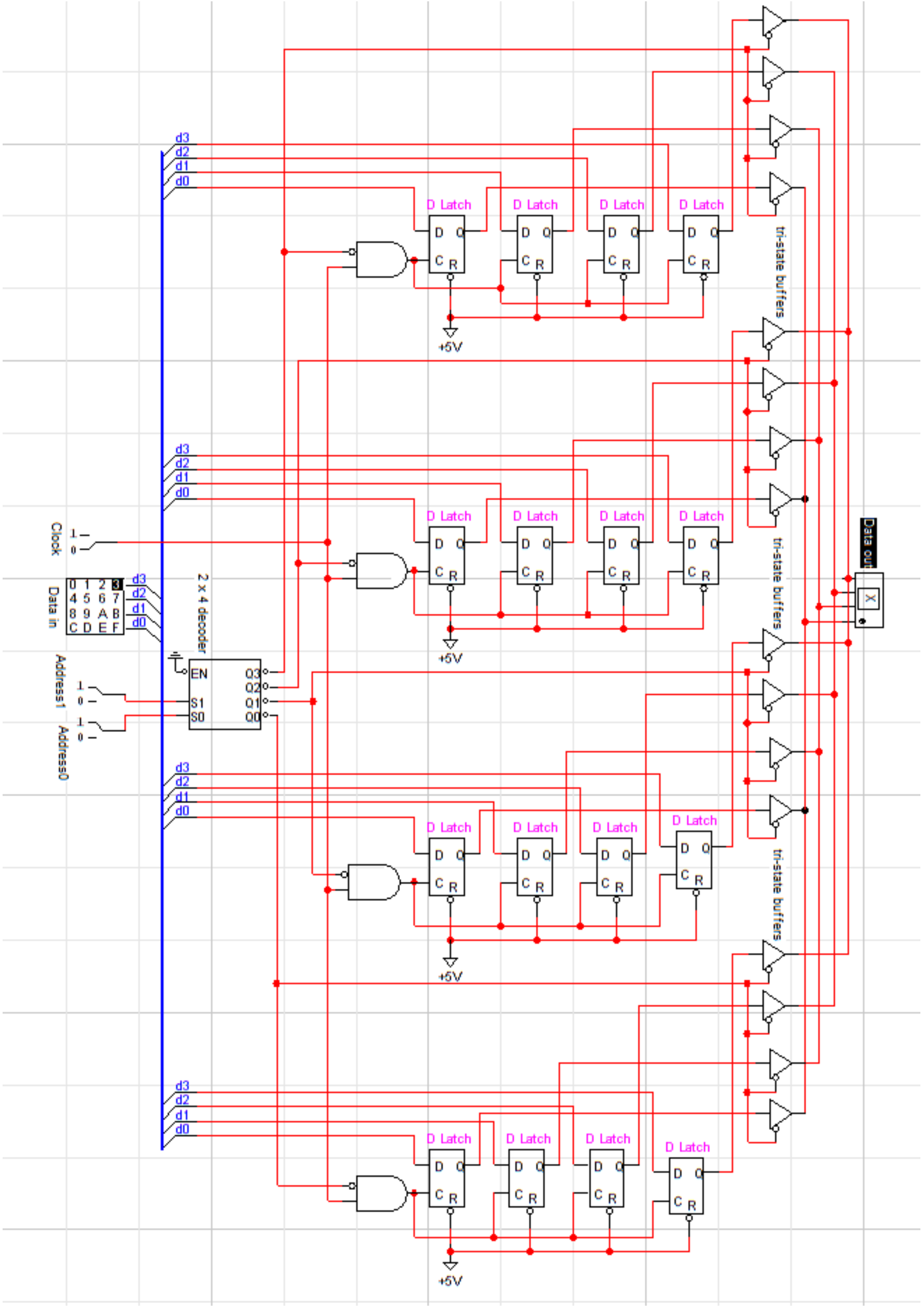


Set Load = 0 (enables tri-state buffers so that address lines are connected to memory)

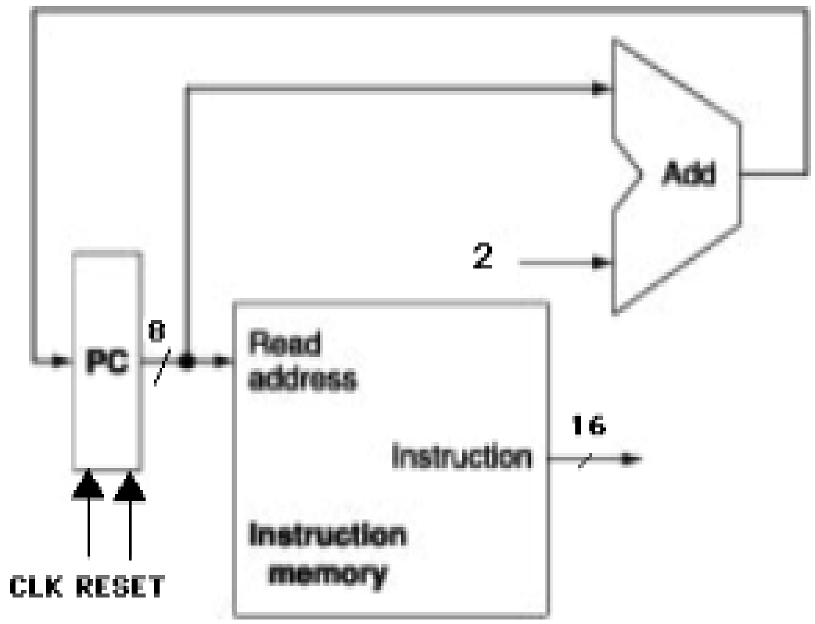
Set Addressin to location in memory

Set Datain to value to be stored at specified address

Set Write = 0 (data is stored in memory), then back to 1


Clock
Data in
0 1 2 3
4 5 6 7
8 9 A B
C D E F
d3
d2
d1
d0
2 x 4 decoder
EN
S1
S0
Q3
Q2
Q1
Q0
Address1
Address0
D Latch
D Q
C R
+5V
tri-state buffers
Data out
X

Instruction Fetch



1. The **Instruction memory** contains 256 8-bit values (bytes). Each byte has a unique 8-bit location, or **address**.

2. The Program Counter register (**PC**) is initialized to address 0 by a **reset**.

3. Address 0 is always the address of the first **instruction** in the program, which is stored in the Instruction memory. Each instruction is 16 bits in length, so it takes 2 bytes to store an instruction.

4. Address 0 is input to Instruction Memory, and the instruction stored at address 0 is produced at data outputs

5. The value of the PC is also an input to the adder, which adds 2 to the address to calculate the address of the instruction in memory (since the instruction is 2 bytes long, you need to move to an address 2 bytes away to get to the address of the next instruction in memory).

6. The output of the adder is applied to the inputs of the PC, so that when the PC is **clock**ed, it becomes the next address applied to the memory (and the next instruction stored in memory is then produced at the data outputs).

7. So, every time the PC is clocked, the next sequential instruction in memory is produced at the data outputs.

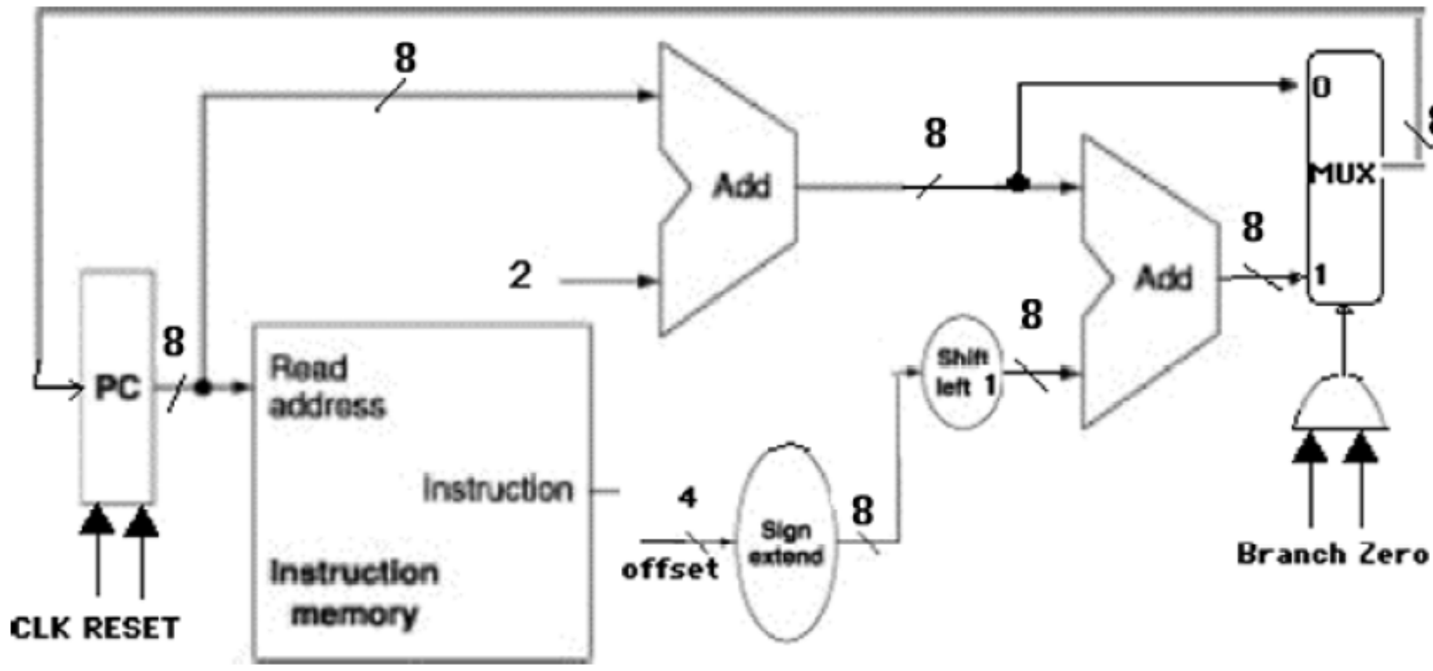
Branch Address

Not all programs execute in sequential order; a **conditional branch** instruction in the program can cause the PC to skip to an earlier or later address in memory. Conditional branching is how conditional statements and loops are implemented.

The next value of the PC (address of the next instruction to be executed) is either:

PC + 2 or
PC + 2 + (2*offset)

offset = number of instructions away from the current instruction.



The **offset** is 4 bits, so it must be **sign-extend**ed to 8 bits to be added to the PC.

A **2x8 multiplexer** circuit selects the next value of the PC. The value of the **Branch** and **Zero** bits are used to determine which is used:

The **Branch** control line = 1 if a branch instruction is being executed.

The **Zero** bit from the ALU is used to check whether the branching condition is met. If **Branch** = 1 and **Zero** = 1, then the next value of the PC will be the branch address; otherwise, it will simply be PC + 2.