Terminology

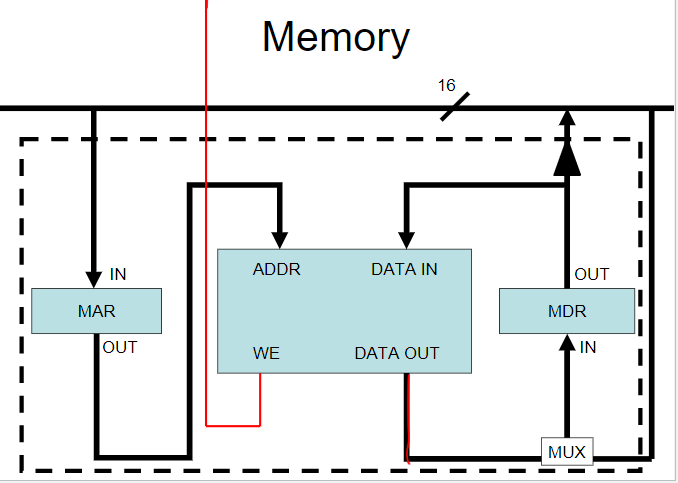
**Patt: Circuitsim**

Gated D Latch Register (with level-triggered setting)  
Master-Slave Flip-flop D Flip-Flop -or- Register (with edge-triggered setting)

Machine Instructions as Data?

* The von Neumann model leads us to treat machine instructions as just another data representation!
* Think unsigned integer, twos-complement integer, ASCII, IEEE-754 floating point, …, and machine instructions

Memory

* Take memory address off the bus to the Memory Address Register (MAR) so it is not all on the bus. This is tied to the address block, which then links to the Memory Data Register (MDR) where data can be read back out of the memory.
* Can store data in one cycle and then use the same bus to send or receive data
* 

Finite state machine is combinational logic with a register added. Determines what is the new state and what control lines come out. Register connected to both the input and output

PC – Program counter, points to next memory address we are going to run in our program.

The Program Counter holds? the address of the next instructions to be executed

The Instruction register hold? The instruction currently being executed

Machines do not need the same addressability and instruction size

When you execute an instruction in machine code, a cycle begins

Fetch (fetches the instruction from the memory using PC) -> Decode -> Evaluate Address [optional] -> Fetch Operands [optional] -> Execute (and, add) [optional] -> Store Result [optional]

Program Counter Circuit  
 PCMUX, increment by 1

Increment of 1 is default increment, used commonly in loops

Instructions look just like data – its all interpretation

Three basic kinds of instructions:

-computational instructions (ADD, AND, …)

-data movement instructions (LD, ST, …)

-control instrctinos (JMP, BRnz, …)

Six basic phases of instructions processing

F -> D -> EA -> OP -> EX -> S

-Not all phases needed by every instruction

-Phases may take variable number of machine cycles