Chapter 5: Synchronous Digital Systems (SDS)

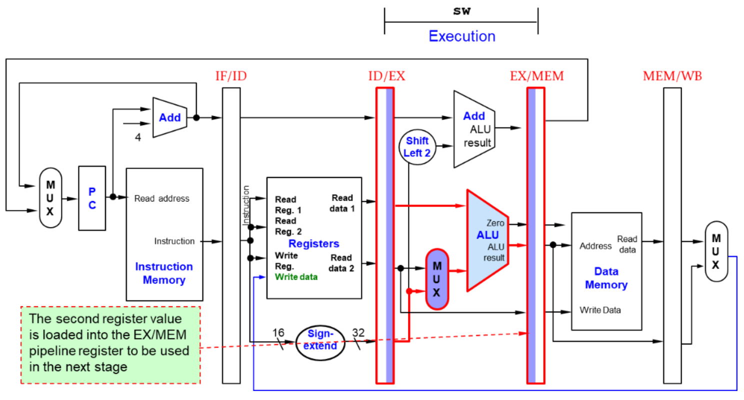
- SDS system that runs on our computers.
- It's digital because electric signals are interpreted as either 1 (asserted) or 0 (unasserted). It's synchronous because all the operations are coordinated by a clock - a device in the computer that alternates between 1 and 0 at regular intervals, the *clock period*.
- They are two critical elements of SDS:
    - Combinational Logic Element - output is only a func of the input
        - Used to perform operations on inputs
    - State Element - stores a value for an indeterminate amt. of time
        - Used to store information and control the flow of info between combinational logic blocks

5.1 Registers

- State element that is mostly used in the circuit
- On rising edge of the clock, the input *d* is sampled and transferred the output *q*
    - At other times, input *d* is ignored
- Three critical timing requirements for registers:
    - *Setup Time* - how long the input must best able before the rising edge of the clock for the register properly read it
    - *Hold Time* - how long the input must best able after the rising edge of the clock for the register properly read it
    - *Clock-to-Q Time* - how long after the rising edge of the clock it takes for input to appear at the register output\

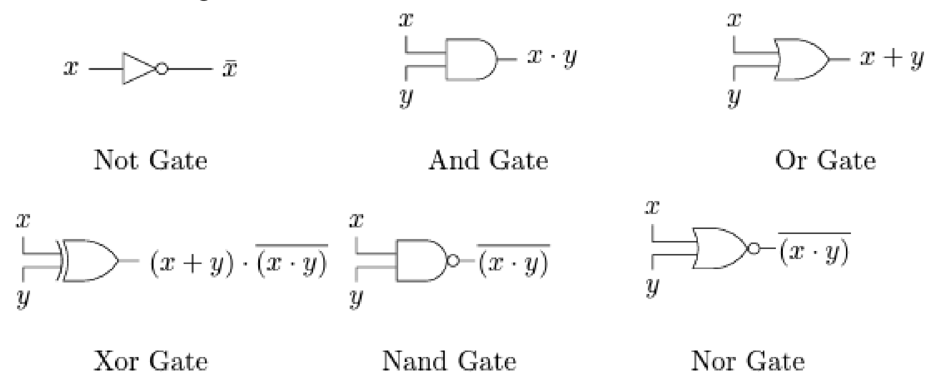
5.1.1 Pipelining

- This is where registers become useful! Pipelining a circuit means placing registers after the combinational logic block
    - This stops delay times from adding up because now intermediate quantities must be stored in the register before being passed to the next block
        - This allows for higher clock frequencies because we are no longer limited to the logic delays, only by our registers



The second register value is loaded into the EX/MEM pipeline register to be used in the next stage

5.2 Combinational Logic Elements



- This is the basic logic gates used in combinational logic.
- Using Boolean Algebra, we can simplify complicated logic statements/circuits
- Data multiplexor (mux):

$$\begin{array}{c|c}
x\cdot\bar{x}=0 & x+\bar{x}=1 \\
x\cdot 0=0 & x+1=1 \\
x\cdot 1=x & x+0=x \\
x\cdot x=x & x+x=x \\
x\cdot y=y\cdot x & x+y=y+x \\
(xy)z=x(yz) & (x+y)+z=x+(y+z) \\
x(y+z)=xy+xz & x+yz=(x+y)(x+z) \\
xy+x=x & (x+y)x=x \\
\overline{xy}=\bar{x}+\bar{y} & \overline{x+y}=\bar{x}\bar{y}
\end{array}$$

    - Mux takes in *n* different sources of data and selects one of those sources to output by using select inputs
        - Ex: if a mux has 3 select bits, then it can choose from 8 different streams of data

5.3 Timing

- Important because combinational logic blocks have propagation delays (the output doesn't change instantaneously with input)
- When used in conjunction with registers (which what their own timing requirements), if not careful: we could build a circuit which product inaccurate outputs
- When adding whether or not a circuit will do what it's designed to do, we need to look at two special paths:
    - Longest CL Path – the longest path of combinational logic blocks between 2 registers
    - Shortest CL Path – the shortest path of combinational logic blocks between 2 registers

- It allows us to analyze things like maximum clock rate to achieve the max hold time our registers can have
    - $t_{delay} = t_{setup} + t_{clk2q} + t_{longestCL}$
        - This formula tells us the max amount of time it takes for an input to propagate from a register to another register.
        - When clock rises the first register will read the input and output it takes tClk2q later
        - The input will then go in thru the combinational logic elements. Since only looking for max of time, we will consider the longestCL path
        - The output needs to be stable for tSetup in order for the designation register to read it properly. As long as our clock period is longer than the max delay, our circuit will run properly
    - $t_{hold} = t_{clk2q} + t_{shortestCL}$
        - This formula tells us the max hold time which registers can have
        - When clock rises the first register will read the input and output it takes tClk2q later
        - The input will then go in thru the combinational logic elements. Hold time is how long the input needs to be stable after the rising edge of the clock, so if the comb. logic works incredibly fast, we will need faster hold time bc otherwise the output will change faster than the hold time of the output register
        - This is why we look at the shortestCL path. As long as our registers have hold time which is less than the max-hold, our circuit will work properly