LECTURE #12: Registers

EEL 3701: Digital Logic and Computer Systems
Based on lecture notes by Dr. Eric M. Schwartz

Registers:
- Registers are arrays D flip-flops that are used to store data
  - Quick access memory for CPU’s (used for calculations)
  - I/O ports
- Registers perform the following operations:
  - Load: Clock data into the register.
  - Read: Get data from the register.
  - Do Nothing: Output remains the same.
  - Disable: Prevent the register from performing any functions.

Recall: We needed this register type to supply data to the ALU and store its result.

- Registers can have many other inputs:
  - Enable (E): Inhibits output by outputting all 0’s. (Internal Q unchanged.)
  - Memory Reset (MR)/Clear (CLR): All Q become 0.
  - Load Enable: Prevents Q from changing.
  - Clock Enable: Prevents clock pulses latching new data and changing Q.
  - Output Enable: Q enters a High-Z state.

Recall: We needed this register type to receive data from the sequential adder.
Note: The timing diagram above neglects propagation delay. If propagation delay is considered, the values are no longer in transition and “unknown” at the rising edges. For this reason, in ideal timing diagrams, we use the value just to the left of this transition point.

- There are a variety of formats for shift registers:
  - Serial In, Serial Out
  - Serial In, Parallel Out
  - Parallel In, Serial Out
  - Parallel In, Parallel Out (standard µProcessor register)

Note: “In” represents the “load” operation and “out” represents the “read” operation.

- There are multiple ways of shifting:
  - Logical Shift: Always shifts in a 0.
  - Arithmetic Shift:
    - Right shift maintains the value of the MSb (preserving sign).
    - Left shift brings in a 0.
  - Circular Shift:
    - Right shift moves the LSb to the MSb.
    - Left shift moves the MSb to the LSb.
Synchronous vs. Asynchronous:

**Asynchronous**

- \( t_a \rightarrow \) initiate sync load of \( D_i \)
- \( t_{apd} \rightarrow \) complete load of \( D_i \) after \( t_{PD} \)
- \( t_b \rightarrow \) initiate async reset of \( Q_i \)
- \( t_{bpd} \rightarrow \) complete async reset of \( D_i \) after \( t_{PD} \)
- \( t_c \rightarrow \) NO sync load of \( D_i \)

**Synchronous**

- \( t_a \rightarrow \) initiate sync load of \( D_i \)
- \( t_{apd} \rightarrow \) complete load of \( D_i \) after \( t_{PD} \)
- \( t_b \rightarrow \) sync reset is applied (but ignored)
- \( t_c \rightarrow \) initiate sync reset
- \( t_{cpd} \rightarrow \) complete sync reset after \( t_{PD} \)

---

Figure borrowed from EEL 3701 “Lecture 14: Registers, ALU, Asynch, Synch” by Dr. Eric M. Schwartz
(Figure 5.25 from Fundamentals of Computer Engineering: Logic Design and Microprocessors by Lam, O’Malley, and Arroyo)