Counters

- Asynchronous / Synchronous
- Up / Down
- Using T / D FF
- Other features: Enable, Parallel load, Modulo

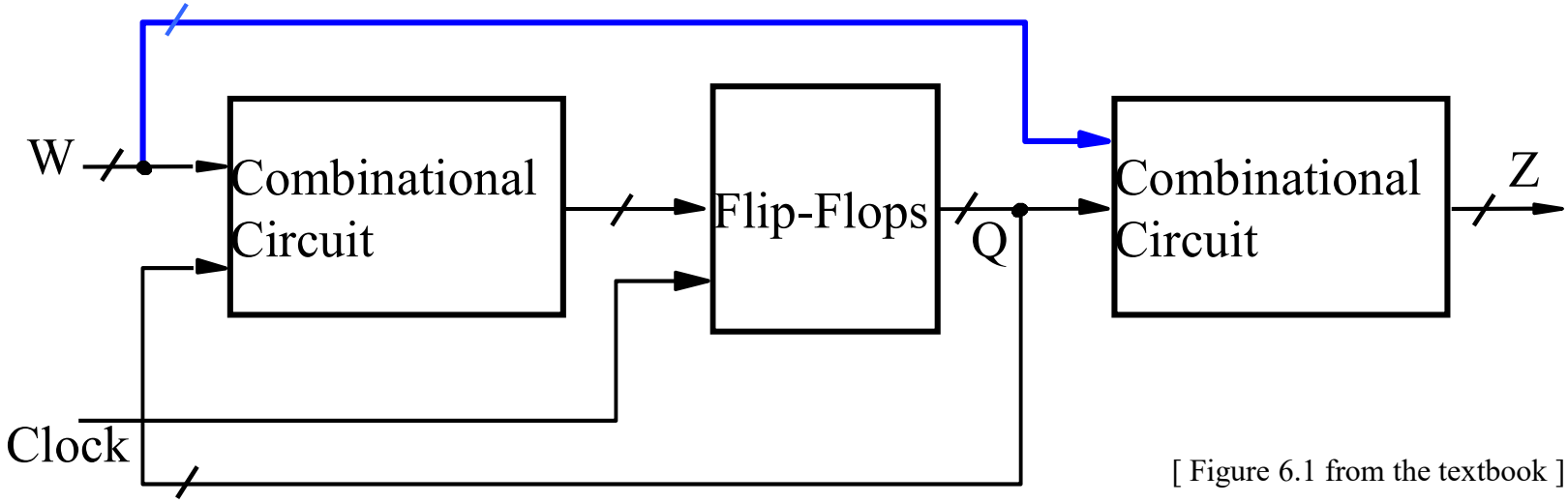
- Remark: *Synchronous counters can be designed as finite state machines*

Timing Analysis of Sequential Circuit (1)

- Key constraints
  - Input to each FF should have been stable for setup time before each clock edge
    - Impact max. clock frequency
  - Input to each FF shall not change before the end of hold-time period
    - Determines whether there is hold-time violation
- Principles
  - Need to consider *signal-delay along all the paths that start and stop at some flip-flops*
    - Starting and ending flip-flops can be different
  - The path with the longest delay, including setup time of ending FF, determines the *highest clock frequency*
  - The path with the shortest delay, not including setup time of ending FF, determines whether there is *any hold-time violation*

Timing Analysis of Sequential Circuit (2)

- Clock frequency, Clock period (clock cycle time)
- For next state logic,
  - Next state logic propagation delay (worst case delay)
    <= Clock period - FF C-to-Q prop delay - FF set-up time
  - Next state logic contamination delay (best case delay)
    >= FF hold time



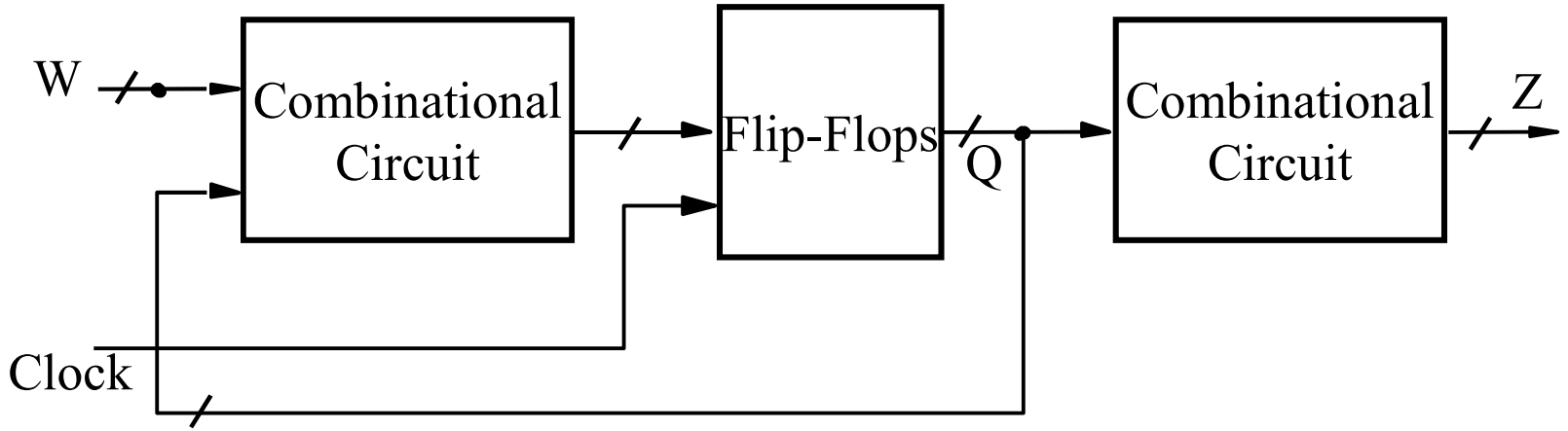
[Figure 6.1 from the textbook]

Sequential Circuit / Finite State Machine

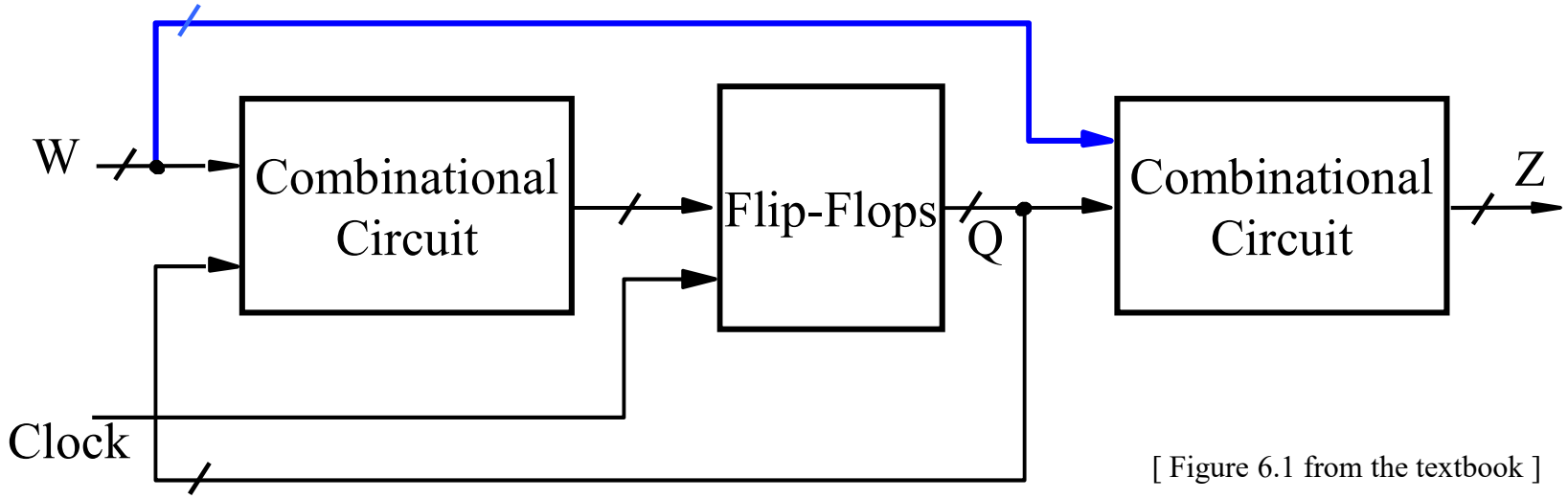
- Output: Moore / Mealy
  - State is different from output
  - For some Moore machine, output logic is trivial
    (i.e., use state directly as output)
- Input: with / without
  - No difference betwee Moore and Mealy machines if no input
- FF: D / T / JK

Overall structure of a Sequential Circuit

Moore machine (outputs depend on current state, but not current inputs)



Mealy machine (outputs depend on both current state and current inputs)



[Figure 6.1 from the textbook]

Designing a Sequential Circuit

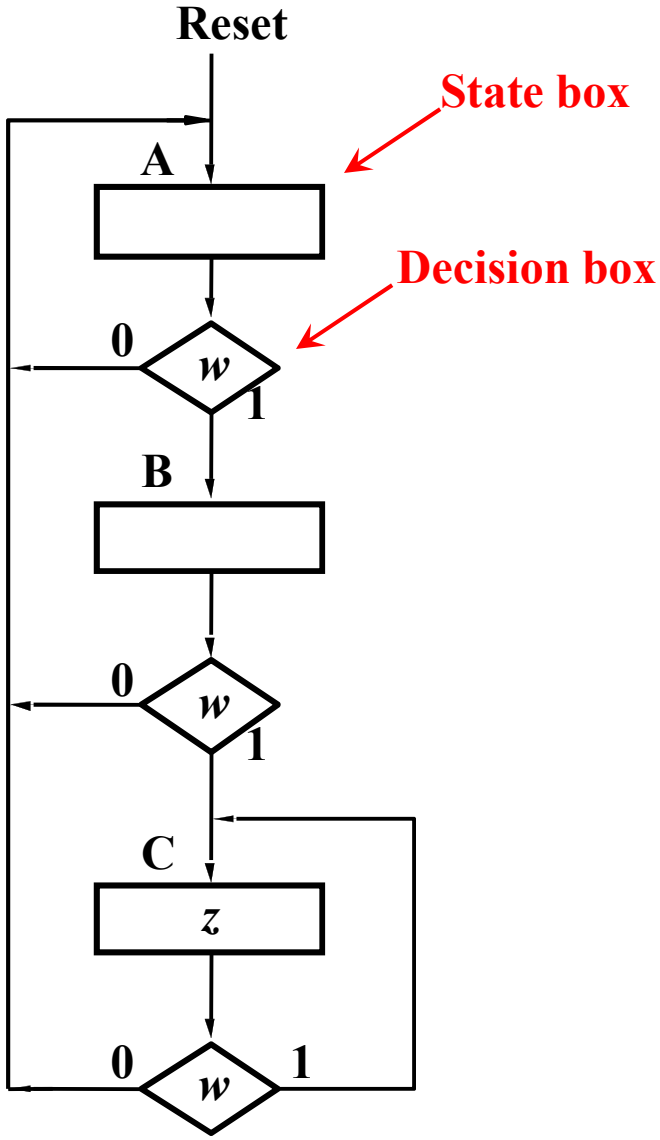
1. Determine number of states you need and what each state represents
2. Draw a state diagram
   - It has an initial state
   - It has other states to keep track of various activities
   - It has some transitions
3. Generate a state table
4. Write state-assigned table
   - Needs state assignment, i.e., the code used for each state
   - State assignment is a complex process
   - For the time being assume straightforward combinations
5. Derive logic expressions for next state variables and for outputs
   - You should simplify the expressions
6. Implement the circuit

Specialized FSM

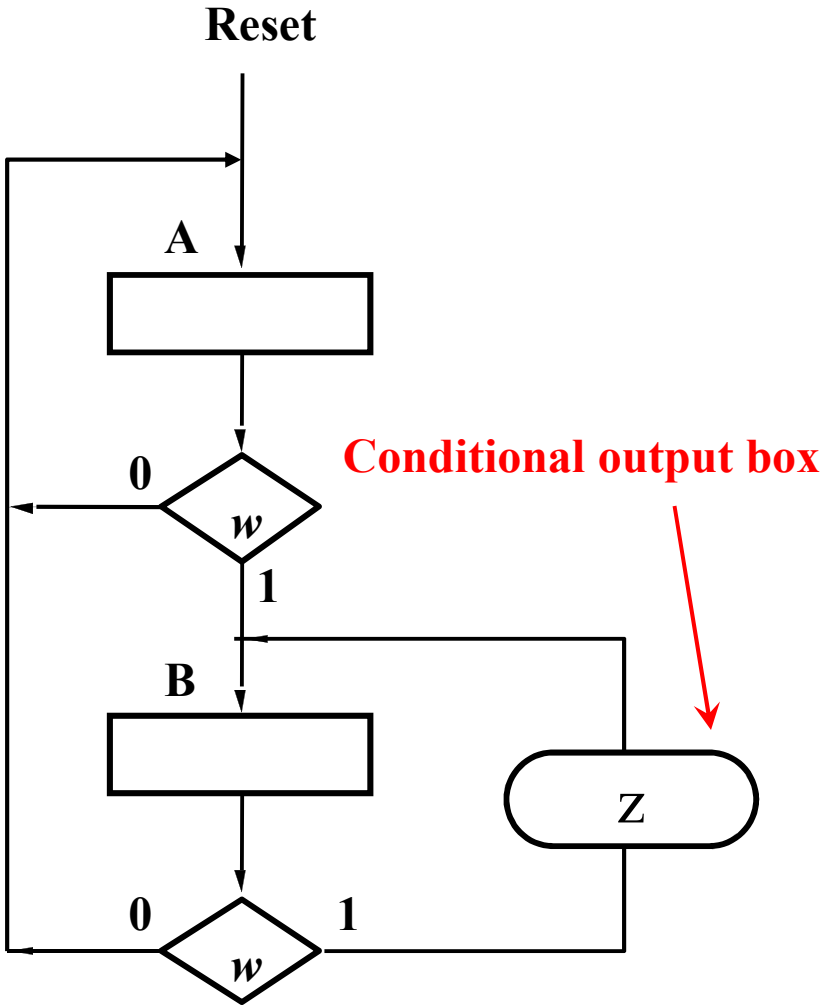
- Examples:
  - Pattern detector
  - Serial adder
  - Counter
  - Arbiter
  - Controller
  - Divisible by N detector
  - Vending machine

ASM Chart

- Moore machine



Mealy machine



Digital System Design

- Datapath
  - Store, manipulate and transfer data
  - Register file
  - ALU
  - Bus: Implementation using tri-state buffer or MUX
  - Memory
  - …
- Control
  - Finite state machine
  - For a given datapath, how to design a control to perform a certain operation?