Professor Brendan Morris, SEB 3216, brendan.morris@unlv.edu

CPE100: Digital Logic Design I

Final Review

Logistics

- Tuesday Dec 10th
  - 13:00-15:00 (1-3pm)
  - 2 hour exam
- Chapters 1-3, 5.1-5.2.3, 5.4
  - Responsible for all material covered in class
    - Exclude 3.5.3-3.5.6

- Closed book, closed notes
- No calculators
- Must show work and be legible for credit

Preparation

- Read the book (2nd Edition)
  - Then, read it again
- Do example problems
  - Use both Harris and Roth books
- Be sure you understand homework solutions

- Come visit during office hours for questions

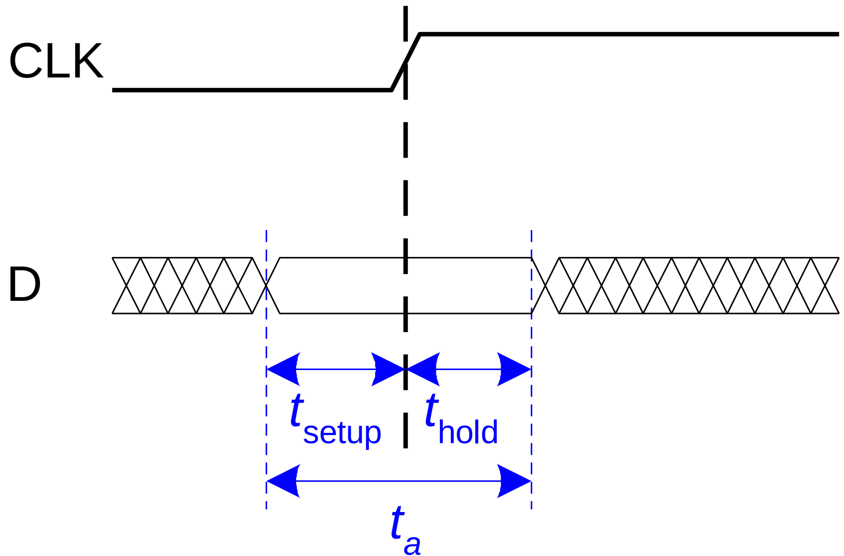
- Exam Advice: Be sure to attempt all problems.
  - Partial credit can only be given for something written on the page
  - Don't spend too much time thinking (don't get stuck)
  - Be sure to read questions completely

Main New Material

- Synchronous timing
  - How can you ensure dynamic discipline is respected
- Parallelism
  - How can you increase operational speed
- Synchronous building blocks
  - Full-adder, counter

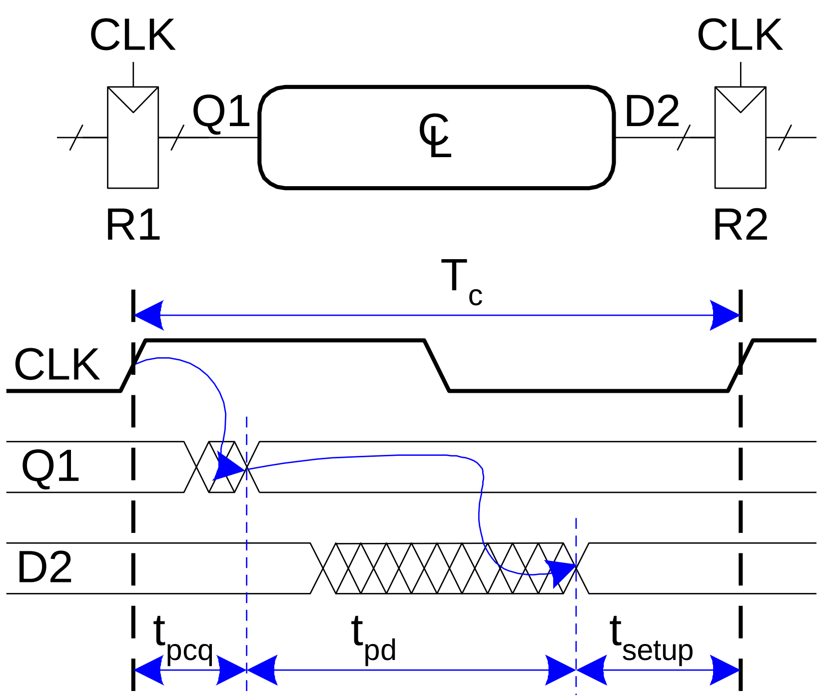
Ch 3.5.2 Synchronous Timing

- Data input must be stabled when sampled at rising clock edge
- Setup time: t_{setup} = time before clock edge data must be stable (i.e. not changing)
- Hold time: t_{hold} = time after clock edge data must be stable

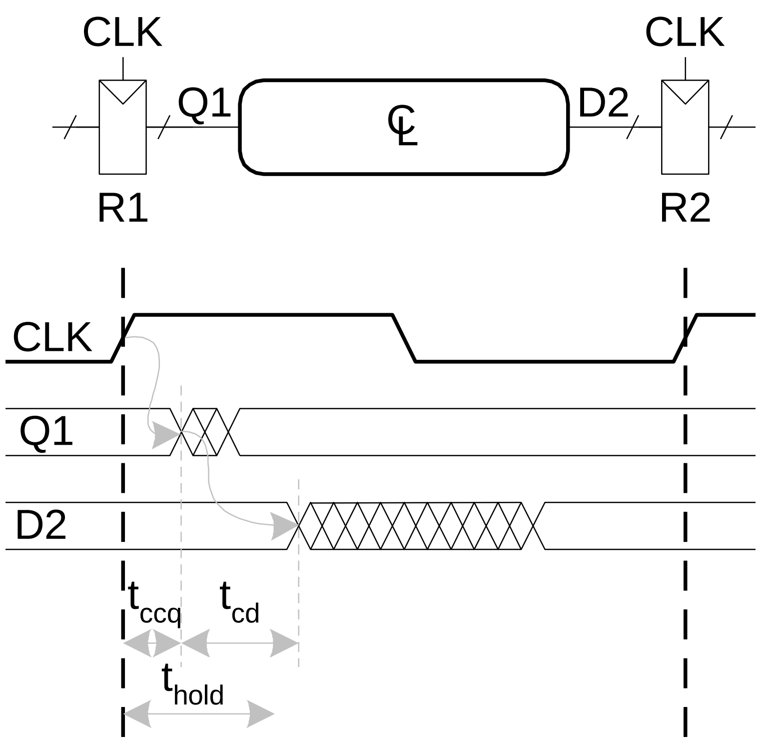


Timing Constraints

- **Setup Timing**
- D_2 cannot change t_{setup} before next clock cycle
- $T_c \geq t_{pcq} + t_{pd} + t_{setup}$



- **Hold Timing**
- D_2 cannot change until after t_{hold} of the current clock cycle
- $t_{hold} < t_{ccq} + t_{cd}$

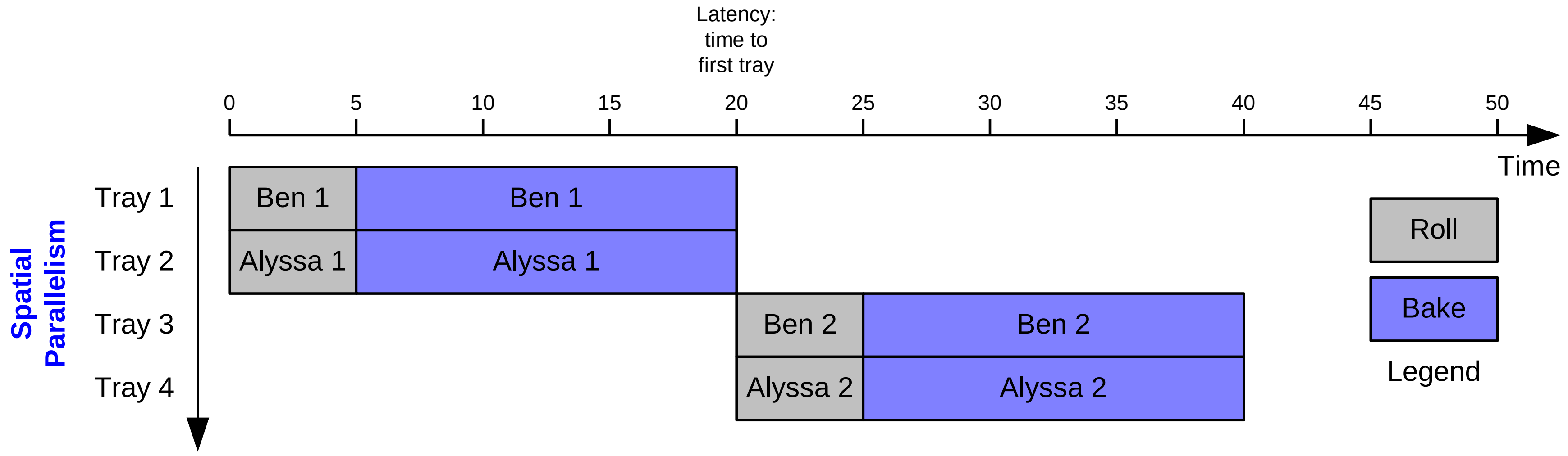


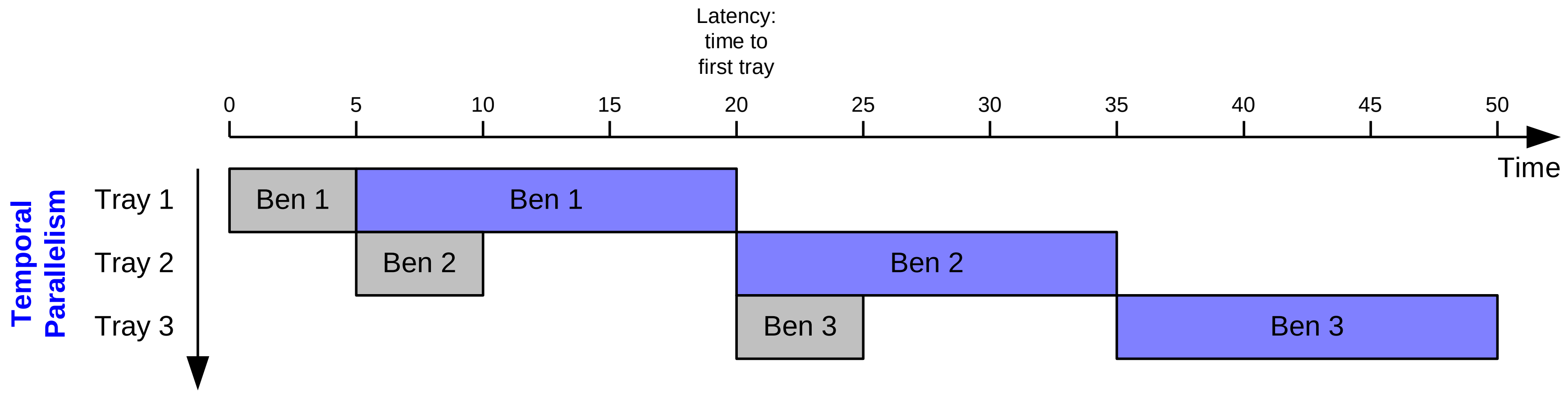
Ch 3.6 Parallelism

- Goal is to increase throughput of circuit
- Two types:
  - Spatial – duplicate hardware to do same task at once
  - Temporal – break task into smaller stages for pipelining (assembly line)
    - Note: this requires no dependencies between stages

- Token – group of inputs processed to produce group of outputs
- Latency – time for a single token to complete
- Throughput – # tokens produced per unit time

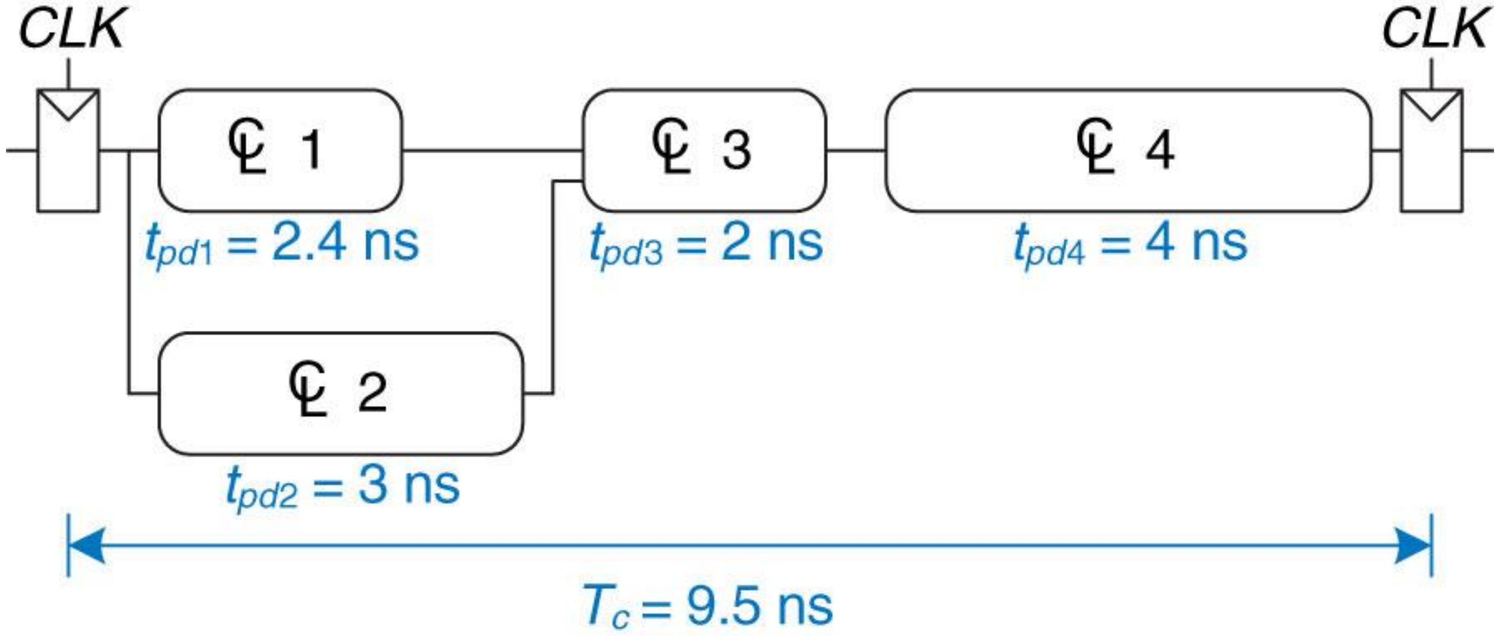
Parallelism Timeline





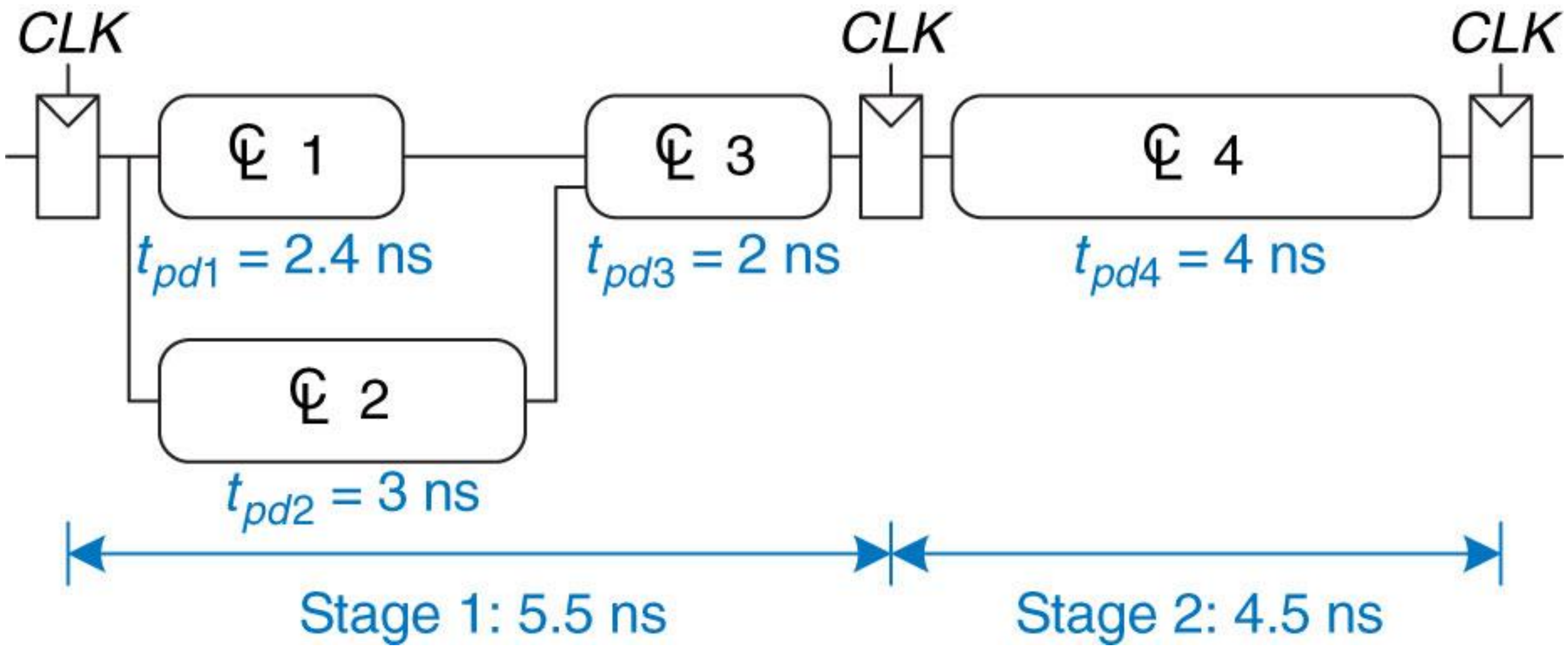
Pipeline Example

- $t_{pcq} = 0.3, t_{setup} = 0.2$ ns
- Identify critical path



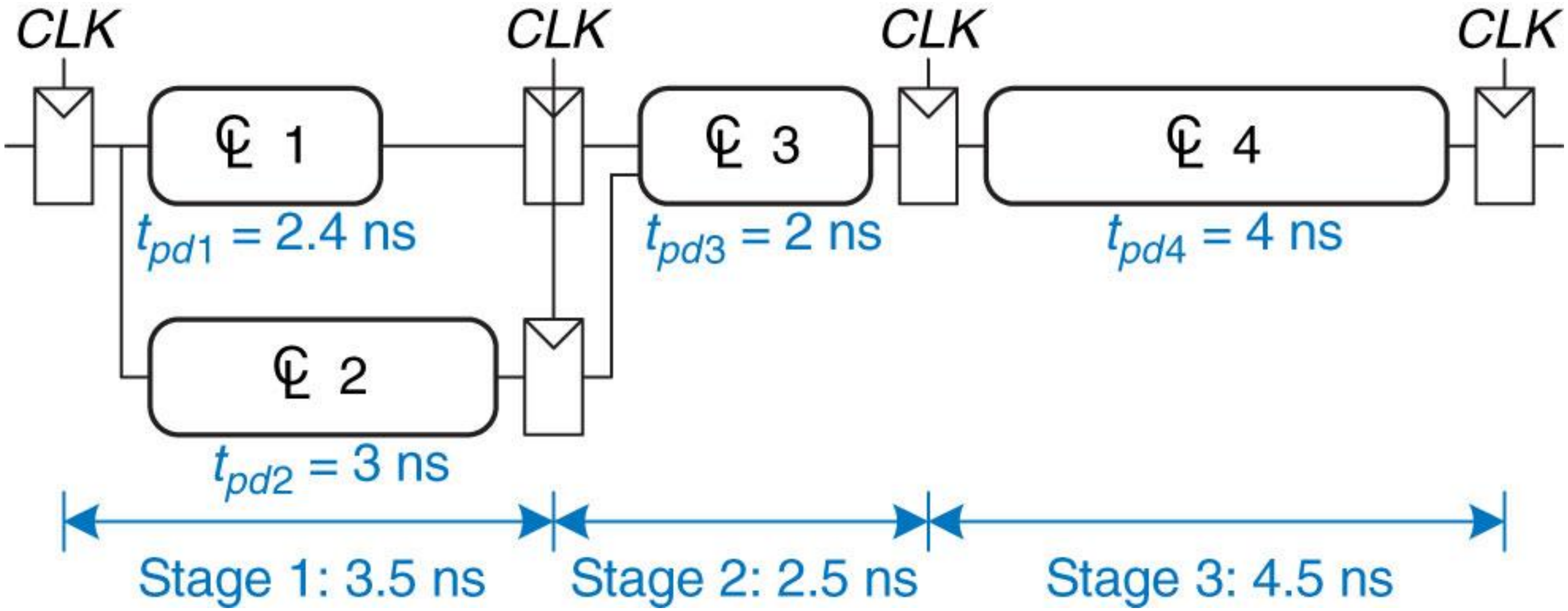
Pipeline Example II

- Two-stage pipeline



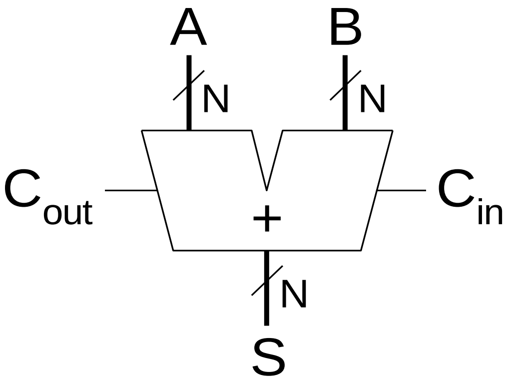
Pipeline Example III

- Three-stage pipeline



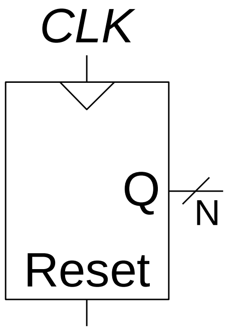
Ch 5 Digital Building Blocks

- Emphasis on full adder (FA) and counter
- Understand ripple adder
  - $t_{ripple} = Nt_{FA}$

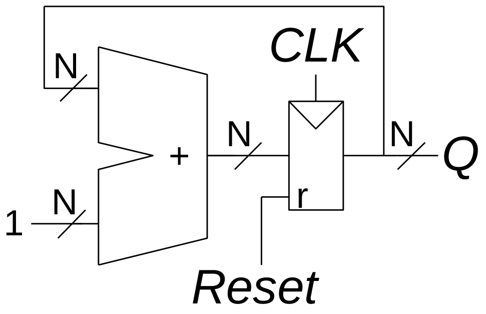


- Counter increments stored value each clock cycle

Symbol

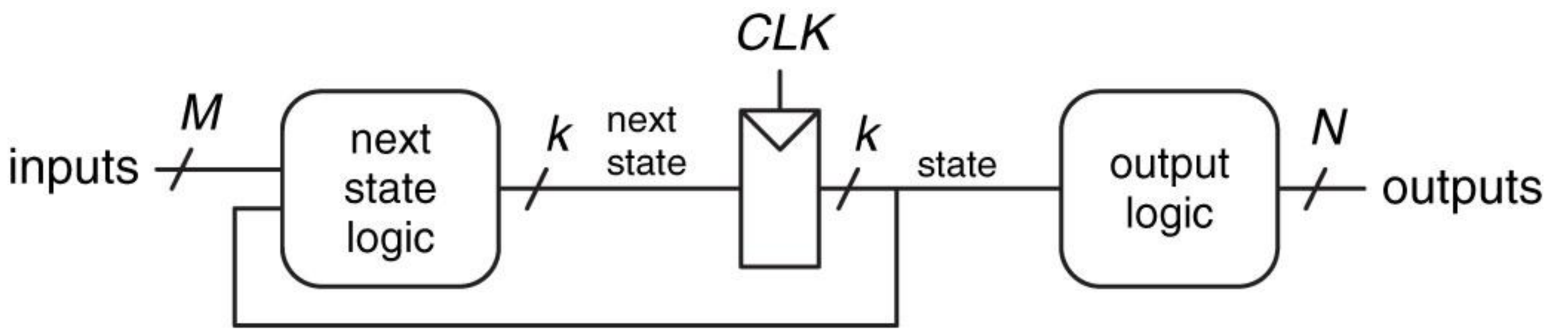


Implementation



Chapter 3.4 Finite State Machine

- Technique for representing synchronous sequential circuit
  - Consists of combinational logic and state register
  - Moore machine – output only dependent on state (not inputs)



Chapter 3.4 FSM Design Steps

1. **Identify inputs and outputs**
2. **Sketch state transition diagram**
3. Write state transition table
4. Select state encodings
5. Rewrite state transition table with state encodings
6. Write output table
7. Write Boolean equations for next state and output logic
8. Sketch the circuit schematic

Chapter 3.4 FSM Examples

- Given problem description, give state transition diagram
- Given state transition diagram, encode state and provide next state/output equations
- Given FSM circuit, describe what system does and give state transition/output tables

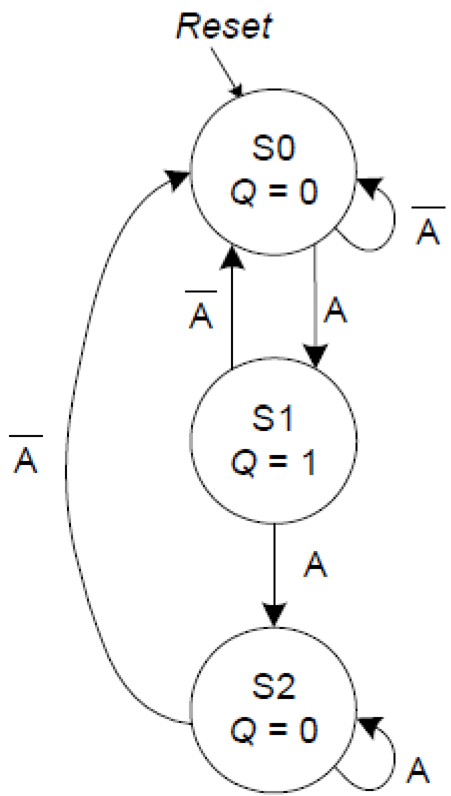
Chapter 3.4 FSM Examples

- Design and edge detector circuit. The output should go HIGH for one cycle after the input makes a 0 → 1 transition.

- Single input: A

FSM Example

- State transition diagram



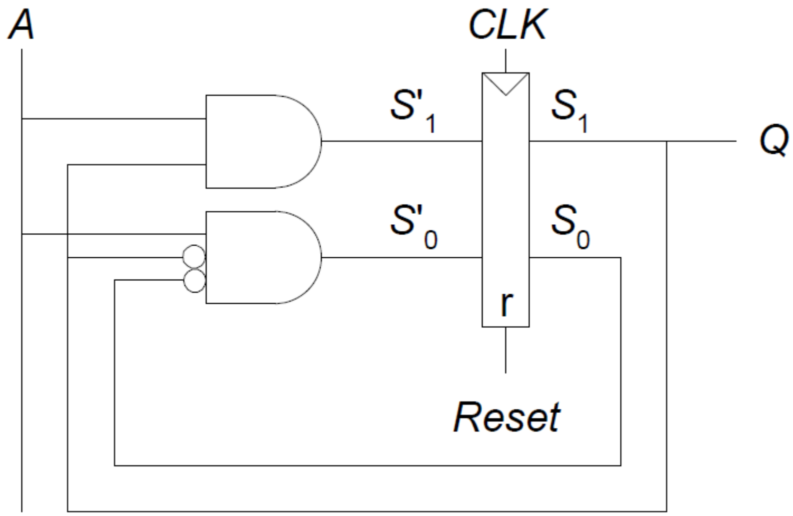
- State/Output Tables
  - Use binary state encoding

$S_1 S_0$	A	$S'_1 S'_0$	Q
00	0	00	0
00	1	01	0
01	0	00	1
01	1	10	1
10	0	00	0
10	1	10	0

FSM Example

- Equations
- $S_1' = AS_1 + AS_0$
- $S_0' = A\overline{S_1}\,\overline{S_0}$
- $Q = S_1$

- Circuit diagram



Ch 1.4 Number Systems

- General number representation
  - N-digit number $\{a_{N-1}a_{N-2}\ldots a_1a_0\}$ of base R in decimal
    - $a_{N-1}R^{N-1}+a_{N-2}R^{N-2}+\cdots+a_1R^1+a_0R^0$
    - $=\sum_{i=0}^{N-1}a_iR^i$
  - What is range of values?

- Should be very familiar with common bases such as 2, 10, 16
  - Be able to convert between bases

Number Example

- What is 10110_2 in (unsigned) decimal?
- Convert 10110_2 to base 6

Binary Addition

- Understand signed number representation (unsigned, two’s complement, sign-mag)
- Addition
    - Potential for overflow – know how and when occurs
- Subtraction
    - Find negative of number and do addition
- Zero/sign extension – when should you use which?

Binary Addition Example

- Assume 4-bit 2's complement and indicate if overflow occurs

- Add $-8\ +\ 4$

Ch 1.5 Logic Gates

- Know circuit symbols and associated truth tables
  - NOT/BUF, AND/OR, NAND/NOR, XOR, XNOR

- Be able to determine output from gate level circuit schematic
  - Both give truth table and provide Boolean equation

Ch 2.3-2.4 Boolean Equations

- Sum-of-product (SOP) minterm form
- Product-of-sum (POS) maxterm form
- Simplify using axioms/theorems
- Example

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

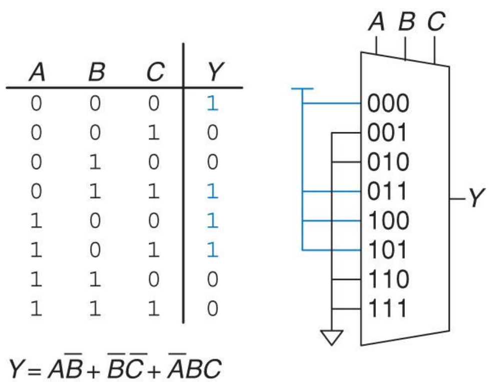
Ch 2.7 Kmap

- Convert truth table to Kmap and draw bubbles to maximally cover ones
  - Be sure to know how to include don’t cares
  - Know up to 5-input function
- Example

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

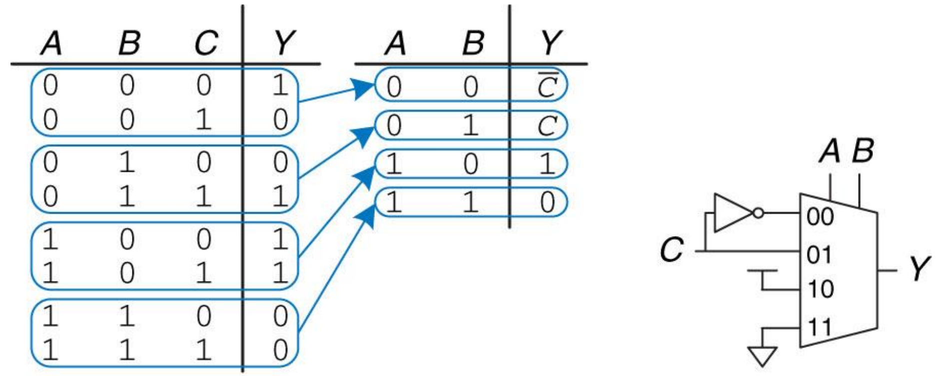
Ch 2.8 Mux/Decoder

- Know how to do logic with mux or decoder
- Mux



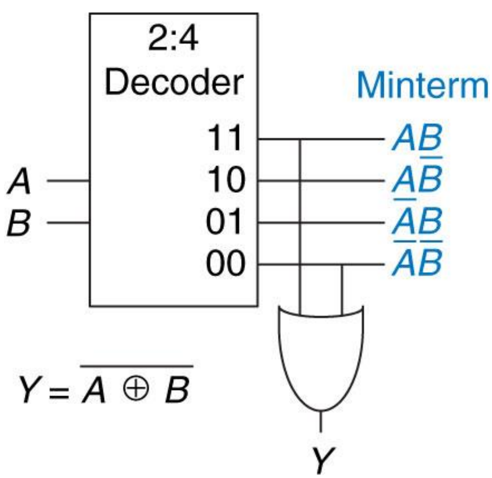
$Y = A\overline{B} + \overline{B}\,\overline{C} + \overline{A}BC$

(a) (b)



(a) (b) (c)

- Decoder



$Y = \overline{A \oplus B}$

Ch 2.9 Combinational Timing

- Delay for input to cause a change in output
- Propagation delay t_{pd} is longest time to see output change
- Contamination delay t_{cd} is shortest time to see output change

Ch 3.2 Sequential Elements

- Sequential elements store “state” – have memory
- Need to know the operation of different devices
  - SR latch, D latch, D flip-flop
- Should also understand the internal circuitry for these elements
  - Given a sequential circuit design you can explain operation
  - Given a description of operation, build a circuit using sequential building blocks.

Sequential Element Example

- How does the following work?