Introduction to Digital Design

Week 7: Clock, Latches, and Flip-Flops

Yao Zheng
Assistant Professor
University of Hawai'i at Mānoa
Department of Electrical Engineering

Overview

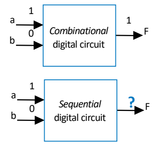
- Sequential circuit
  - Output depends not just on present inputs but on past sequence of inputs.
- SR Latch
  - Feedback circuit for bit storage.
  - Race condition and level-sensitive latch.
- D Latch and D Flip-Flop
  - D Latch: inserted inverter ensures R always opposite of S
  - D Flip-Flop: bit storage that stores on clock edge.
- Clock signal
  - Flip-flop to generates periodic pulsing signal.
- Basic register

2

3.1

Introduction

- Sequential circuit
  - Output depends not just on present inputs (as in combinational circuit), but on past sequence of inputs
    - Stores bits, also known as having "state"
  - Simple example: a circuit that counts up in binary
- This chapter will:
  - Design a new building block, a **flip-flop**, to store one bit
  - Combine flip-flops to build multi-bit storage – **register**
  - Describe sequential behavior with **finite state machines**
  - Convert a finite state machine to a **controller** – sequential circuit with a register and combinational logic



Must know sequence of past inputs to know output

3

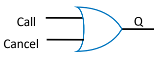
Note: Slides with animation are denoted with a small red "a" near the animated items

3.2

Storing One Bit – Flip-Flops

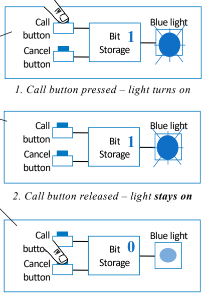
Example Requiring Bit Storage

- Flight attendant call button
  - Press call: light turns on
    - ***Stays on*** after button released
  - Press cancel: light turns off
    - Stays off after button released
  - Logic gate circuit to implement this?



Doesn't work. Q=1 when Call=1, but doesn't stay 1 when Call returns to 0

Need some form of "feedback" in the circuit



1. Call button pressed – light turns on

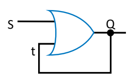
*2. Call button released – light **stays on***

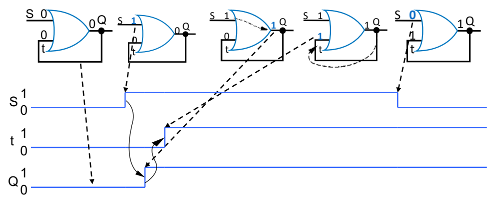
3. Cancel button pressed – light turns off

4

First attempt at Bit Storage

- Need some sort of feedback
  - Does circuit on the right do what we want?
    - No: Once Q becomes 1 (when S=1), Q stays 1 forever – no value of S can bring Q back to 0

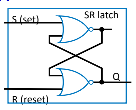


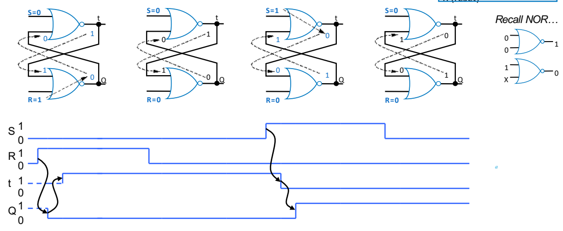


5

Bit Storage Using an SR Latch

- Does the circuit to the right, with cross-coupled NOR gates, do what we want?
  - Yes! How did someone come up with that circuit? Maybe just trial and error, a bit of insight...

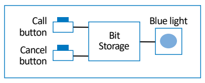


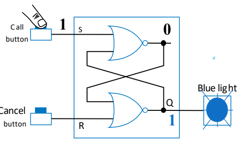


6

Example Using SR Latch for Bit Storage

- SR latch can serve as bit storage in previous example of flight-attendant call button
  - Call=1 : sets Q to 1
    - Q stays 1 even after Call=0
  - Cancel=1 : resets Q to 0
- But, there's a problem...

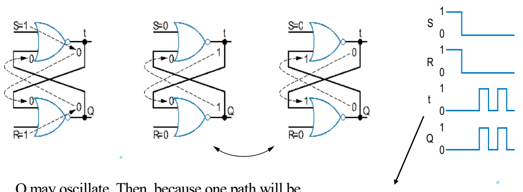




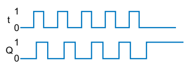
7

Problem with SR Latch

- Problem
  - If S=1 and R=1 simultaneously, we don't know what value Q will take



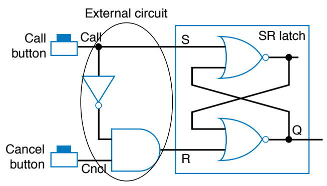
Q may oscillate. Then, because one path will be slightly longer than the other, Q will eventually settle to 1 or 0 – but we don't know which. Known as a *race condition.*



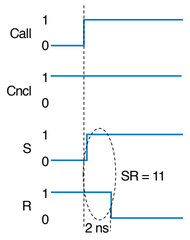
8

Problem with SR Latch

- Designer might try to avoid problem using external circuit
  - Circuit should prevent SR from ever being 11
  - But 11 can occur due to different path delays



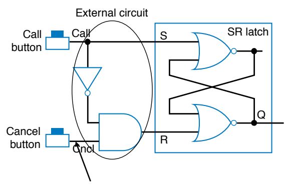
Assume 1 ns delay per gate. The longer path from Call to R than from Call to S causes SR=11 for short time – could be long enough to cause oscillation



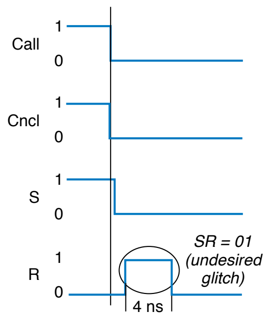
9

Problem with SR Latch

- Glitch can also cause undesired set or reset



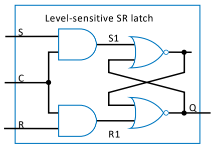
Suppose this wire has 4 ns delay

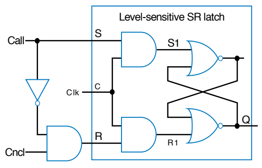


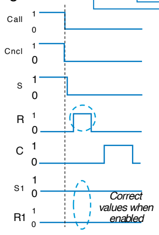
10

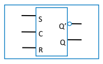
Solution: Level-Sensitive SR Latch

- Add enable input "C"
- Only let S and R change when C=0
  - Ensure circuit in front of SR never sets SR=11, except briefly due to path delays
  - Set C=1 after time for S and R to be stable
  - When C becomes 1, the stable S and R value passes through the two AND gates to the SR latch's S1 R1 inputs.









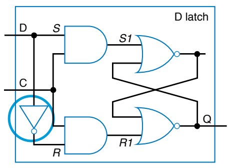
Level-sensitive SR latch symbol

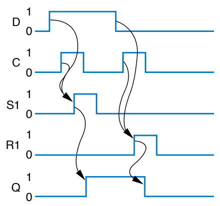
Glitch on R (or S) doesn't affect R1 (or (S1)

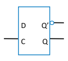
11

Level-Sensitive D Latch

- SR latch requires careful design to ensure SR=11 never occurs
- D latch relieves designer of that burden
  - Inserted inverter ensures R always opposite of S





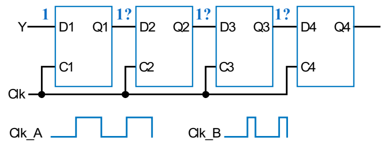


D latch symbol

12

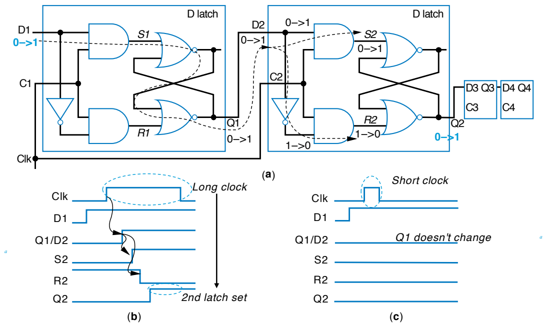
Problem with Level-Sensitive D Latch

- D latch still has problem (as does SR latch)
  - When C=1, through how many latches will a signal travel?
  - Depends on how long C=1
    - Clk_A – signal may travel through multiple latches
    - Clk_B – signal may travel through fewer latches



13

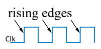
Problem with Level-Sensitive D Latch



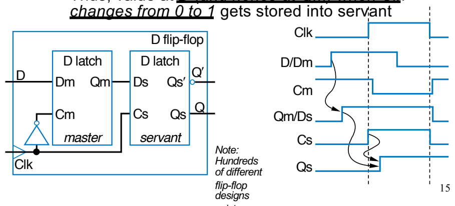
14

D Flip-Flop

Can we design bit storage that only stores a value on the *rising* edge of a clock signal?



- ***Flip-flop***: Bit storage that stores on clock *edge*
- One design – master-servant
  - Clk = 0 – master enabled, loads D, appears at Qm. Servant disabled.
  - Clk = 1 – Master disabled, Qm stays same. Servant latch enabled, loads Qm, appears at Qs.
  - Thus, value at *D (and hence at Qm) when Clk changes from 0 to 1* gets stored into servant

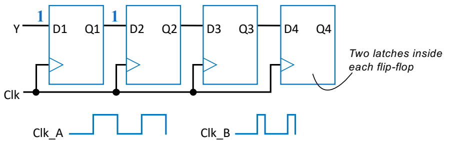


Note: Hundreds of different flip-flop designs exist

15

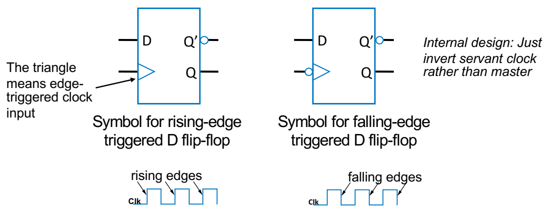
D Flip-Flop

- Solves problem of not knowing through how many latches a signal travels when C=1
  - In figure below, signal travels through exactly one flip-flop, for Clk_A or Clk_B
  - Why? Because on *rising edge* of Clk, all four flip-flops are loaded simultaneously – then all four no longer pay attention to their input, until the next rising edge. Doesn't matter how long Clk is 1.



16

D Flip-Flop



D Q'
Q
The triangle means edge-triggered clock input
D Q'
Q
Internal design: Just invert servant clock rather than master
rising edges
Clk
falling edges
Clk

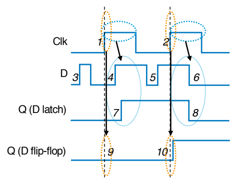
Symbol for rising-edge triggered D flip-flop

Symbol for falling-edge triggered D flip-flop

17

D Latch vs. D Flip-Flop

- Latch is level-sensitive
  - Stores D when C=1
- Flip-flop is edge triggered
  - Stores D when C changes from 0 to 1
- Saying "level-sensitive latch" or "edge-triggered flip-flop" is redundant
- Comparing behavior of latch and flip-flop:



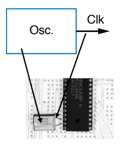
Latch follows D while Clk is 1

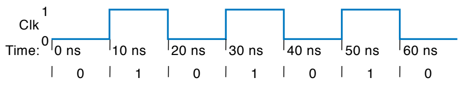
Flip-flop only loads D during Clk rising edge

18

Clock Signal

- Flip-flop Clk inputs typically connect to one clock signal
  - Coming from an oscillator component
  - Generates periodic pulsing signal
    - Below: "Period" = 20 ns, "Frequency" = 1/20 ns = 50 MHz
    - "Cycle" is duration of 1 period (20 ns); below shows 3.5 cycles





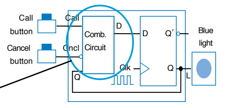
Period/Freq shortcut: Remember 1 ns → 1 GHz

Freq.	Period
100 GHz	0.01 ns
10 GHz	0.1 ns
1 GHz	**1 ns**
100 MHz	10 ns
10 MHz	100 ns

19

Flight-Attendant Call Button Using D Flip-Flop

- D flip-flop will store bit
- Inputs are Call, Cancel, and present value of D flip-flop, Q
- Truth table shown below



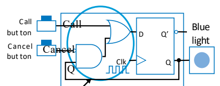
Call	Cancel	Q	D
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Preserve value: if Q=0, make D=0; if Q=1, make D=1

Cancel -- make D=0

Call -- make D=1

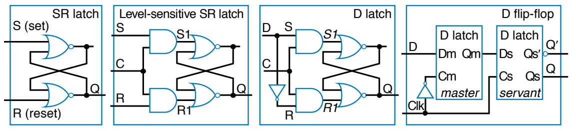
Let's give priority to Call -- make D=1



Circuit derived from truth table, using Chapter 2 combinational logic design process

20

Bit Storage Summary



Feature: S=1 sets Q to 1, R=1 resets Q to 0. *Problem:* SR=11 yields undefined Q, other glitches may set/reset inadvertently.

Feature: S and R only have effect when C=1. An external circuit can prevent SR=11 when C=1. *Problem:* avoiding SR=11 can be a burden.

Feature: SR can't be 11. *Problem:* C=1 for too long will propagate new values through too many latches; for too short may not result in the bit being stored.

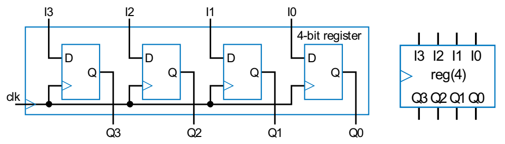
Feature: Only loads D value present at rising clock edge, so values can't propagate to other flip-flops during same clock cycle. *Tradeoff:* uses more gates internally, and requires more external gates than SR– but transistors today are more plentiful and cheaper.

- We considered increasingly better bit storage until we arrived at the robust D flip-flop bit storage

21

Basic Register

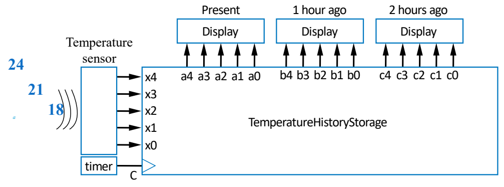
- Typically, we store multi-bit items
  - e.g., storing a 4-bit binary number
- ***Register***: multiple flip-flops sharing clock signal
  - From this point, we'll use registers for bit storage
    - No need to think of latches or flip-flops
    - But now you know what's inside a register



22

Example Using Registers: Temperature Display

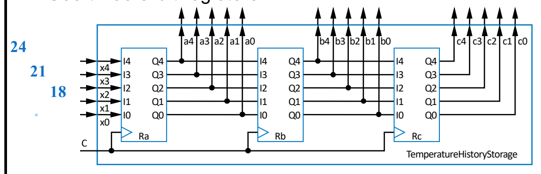
- Temperature history display
  - Sensor outputs temperature as 5-bit binary number
  - Timer pulses C every hour
  - Record temperature on each pulse, display last three recorded values

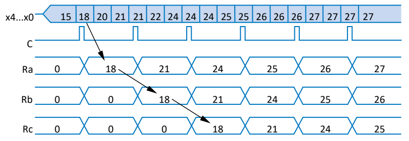


23

Example Using Registers: Temperature Display

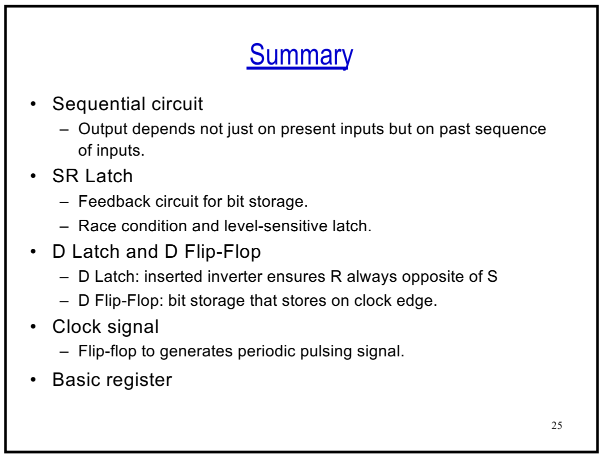
- Use three 5-bit registers





Note that registers only loaded on rising clock edges

24



Summary

- Sequential circuit
  - Output depends not just on present inputs but on past sequence of inputs.
- SR Latch
  - Feedback circuit for bit storage.
  - Race condition and level-sensitive latch.
- D Latch and D Flip-Flop
  - D Latch: inserted inverter ensures R always opposite of S
  - D Flip-Flop: bit storage that stores on clock edge.
- Clock signal
  - Flip-flop to generates periodic pulsing signal.
- Basic register

25