中国科学院大学
University of Chinese Academy of Sciences

CS101

Systems Thinking

Modularization-1:
Combinational Circuits and Sequential Circuits
组合电路与时序电路

zxu@ict.ac.cn
zhangjialin@ict.ac.cn

Outline

- What is systems thinking?
- Three objectives of systems thinking
- Abstraction
- Modularization 模块化　　抽象一点也不“抽象”
  - Modularization and modules
  - Combinational circuits 组合电路
    - Logic gates and combinational circuits
    - The information hiding principle
    - Adders
    - An adder-subtractor controlled by multiplexers
  - Sequential circuits 时序电路
    - Types of memory cells and the D flip-flop
    - General organization of sequential circuit
    - A serial adder example
  - Instruction Set and Instruction Pipeline 指令集与指令流水线
  - Software Stack 软件栈
- Seamless transition

These slides acknowledge sources for additional data not cited in the textbook

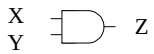
4.1 Modularization and modules
模块化与模块

- Modularization is a systems thinking method, similar to divide-and-conquer in algorithmic thinking　　模块化像算法思维的分治方法
  - Divide a system into multiple subsystems called modules 将系统划分成模块
  - Compose modules into a system (higher-level abstraction) 将模块组合成系统
    - 该系统成为一个更高级的抽象
- In a system with modularization
  - Two modules may be interconnected, but often do not overlap
  - 很多情况下，系统中的两个模块可以连接，但不重叠
- Modularization is a special form of abstraction where the information hiding principle is followed　模块是采用了信息隐藏原理的抽象
- Modularization, i.e., how to divide and compose a system, is an art, needing human imagination and creativity　　模块化是艺术，需要创造力
- Understand how modularization works via a design journey
  - of higher and higher hardware subsystem abstractions
    - from designing gates to designing an instruction pipeline
    - 从逻辑门到指令流水线的旅行，学习模块化概念（模块如何组合成为系统）

4.2.1 Logic gates and combinational circuit

- Logic gates: electronic circuits realizing Boolean operators

X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1



AND

与门

$Z = X \cdot Y$

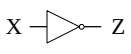
X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

OR

或门

$Z = X + Y$

X	Z
0	1
1	0



NOT

非门

$Z = \overline{X}$

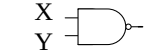
X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

XOR

异或门

$Z = X \oplus Y$

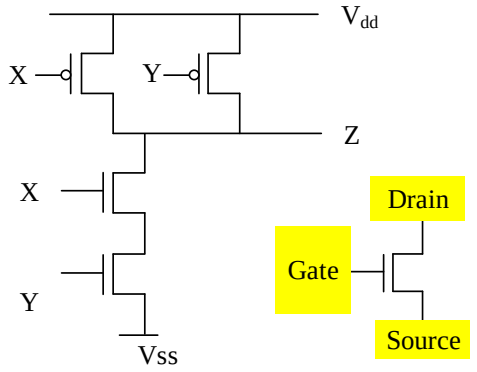
X	Y	Z
0	0	1
0	1	1
1	0	1
1	1	0



NAND

与非门

$Z = \overline{X \cdot Y}$



CMOS circuit for NAND

与非门的**CMOS**电路

每个逻辑门都有半导体电路实现

只需知道与非门的CMOS电路实现

作为例子

晶体管行为

Gate 为1（高电平）：导通

Gate为0（低电平）：断开

Vss: ground (0)

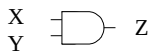
Vdd: high voltage (1)

X,Y都是高电平（1）时，下两个晶体管联通，上两个晶体管断开，Z连接到地（0）

4.2.1 Logic gates and combinational circuit

- Logic gates: electronic circuits realizing Boolean operators

X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1



AND

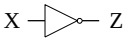
$Z = X \cdot Y$

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

OR

$Z = X + Y$

X	Z
0	1
1	0



NOT

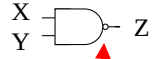
$Z = \overline{X}$

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

XOR

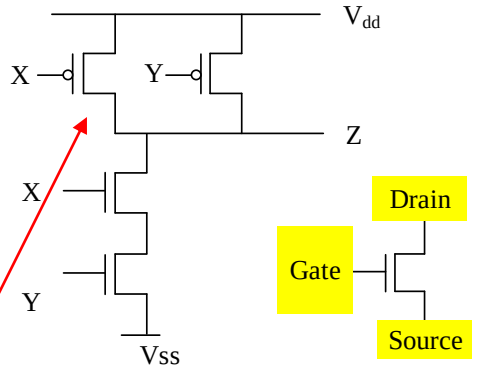
$Z = X \oplus Y$

X	Y	Z
0	0	1
0	1	1
1	0	1
1	1	0



NAND

$Z = \overline{X \cdot Y}$

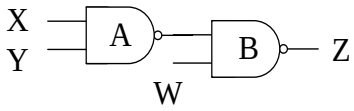


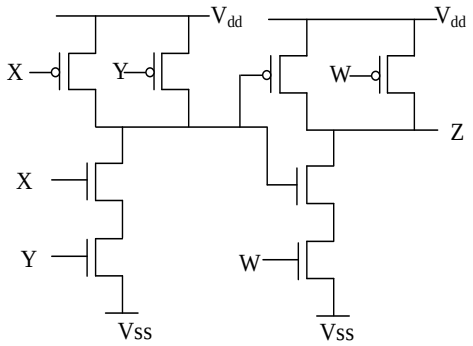
CMOS circuit for NAND

小圈表示**NOT**（非）

circle means NOT 高电平变成低电平，低电平变成高电平

$Z = \overline{(\overline{X \cdot Y}) \cdot W}$

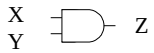




4.2.1 Logic gates and combinational circuit

- Logic gates: electronic circuits realizing Boolean operators

X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

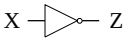


AND

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

OR

X	Z
0	1
1	0

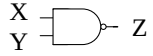


NOT

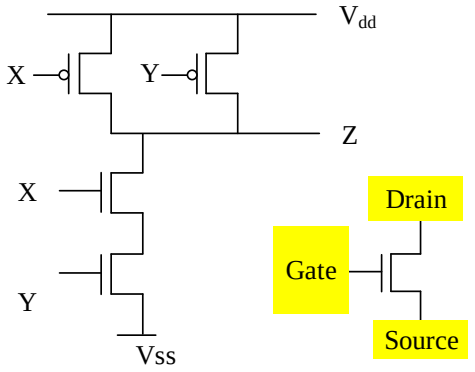
X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

XOR

X	Y	Z
0	0	1
0	1	1
1	0	1
1	1	0

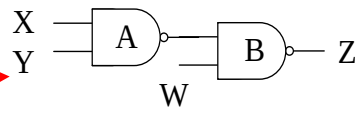


NAND



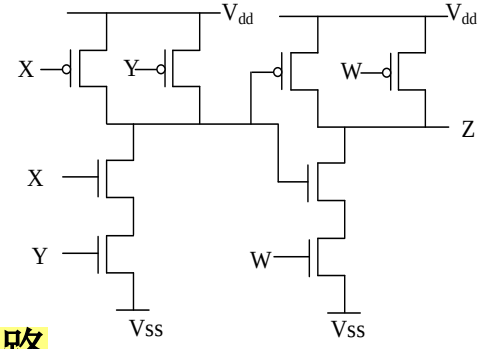
CMOS circuit for NAND

- A combinational circuit is comprised of interconnected gates without feedback wires 组合电路由逻辑门连接而成，没有反馈连线
  - 教科书：组合电路实现命题逻辑表达式
    - Any combinational circuit has a corresponding Boolean expression → $Z = \overline{(\overline{X \cdot Y}) \cdot W}$
    - Any Boolean expression has a corresponding combinational circuit
  - A combinational circuit can be shown as a logic diagram →



  - 不用更加繁杂的CMOS电路

使用逻辑线路图表示组合电路

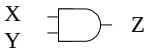


CMOS circuit

4.2.1 Logic gates and combinational circuit

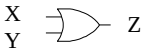
- Logic gates: electronic circuits realizing Boolean operators

X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1



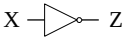
AND

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1



OR

X	Z
0	1
1	0

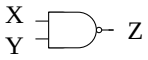


NOT

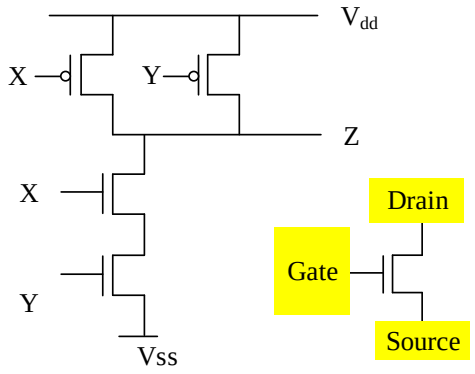
X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

XOR

X	Y	Z
0	0	1
0	1	1
1	0	1
1	1	0



NAND

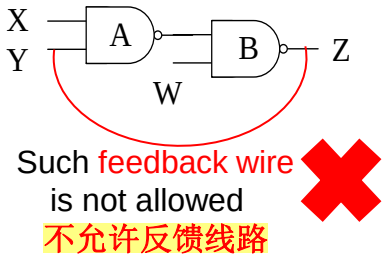


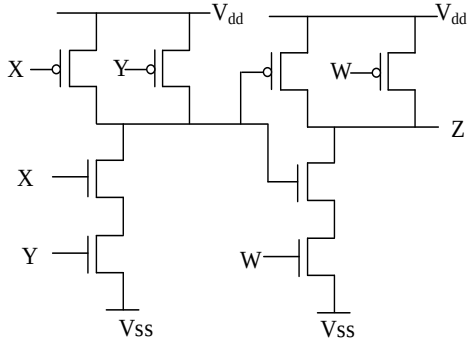
CMOS circuit for NAND

A combinational circuit is comprised of interconnected gates without feedback wires 组合电路由门电路连接而成，没有反馈连线

- Any combinational circuit has a corresponding Boolean expression
- Any Boolean expression has a corresponding combinational circuit
- A combinational circuit can be shown as a logic diagram

$$Z = \overline{\overline{(X \cdot Y)} \cdot W}$$



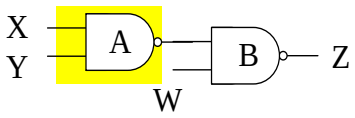


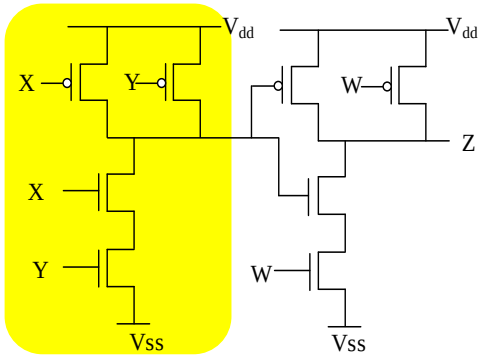
CMOS circuit

4.2.2 The information hiding principle 信息隐藏原理

- A module only exposes its interface and visible behaviors, but hides internal details and internal behavior
  - 模块仅暴露接口和可见行为，隐藏内部细节和内部行为
- Three types of abstractions are shown of the same combinational circuit
  一个组合电路的三种表示，它们逻辑等价
  - Boolean expression 布尔表达式
  - Logic diagram 逻辑线路图
  - CMOS circuit CMOS电路
  - 布尔表达式、逻辑线路图更简洁

$$Z = \overline{(X \cdot Y)} \cdot W$$

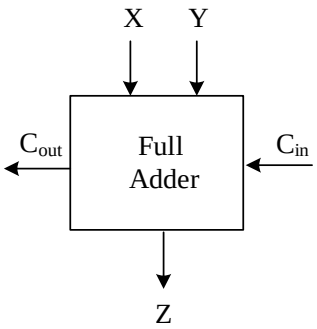




- 三个黄色区域表示同样的
  2-输入-1-输出与非门
- The three yellow areas
  are different abstractions for the same thing
  - a 2-input-1-output NAND gate
  - The Boolean expression and the logic diagram abstractions hide internal details of the CMOS implementation **CMOS电路最复杂，暴露了内部**
  - The former two are simpler and allow different implementations

4.2.3 Adders 加法器

- Add two unsigned numbers X and Y to generate result sum Z
  - Consider the carry-in bit C_{in} and the carry-out bit C_{out}
  - Use the same algorithm we use when doing addition by pen and paper
- For one bit, design a full adder 全加器
  - “全”：考虑进位输入（C_{in}）与进位输出（C_{out}），而不只是本位输入（**X,Y**）与本位输出（**Z**）
  - Here, “full” means the adder considers carry-in and carry-out bits
  - **1-minute quiz**: given X, Y and C_{in}, what are the expressions of Z and C_{out}?

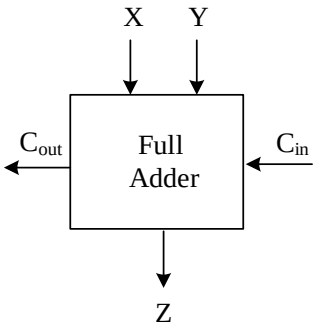


Full adder symbol

Full adder 全加器
代表了设计方法1：直接连接逻辑门

- Add two unsigned numbers X and Y to generate result sum Z
  - Consider the carry-in bit C_{in} and the carry-out bit C_{out}
  - Use the same algorithm we use when doing addition by pen and paper
- For one bit, design a full adder
  - Here, “full” means the adder considers carry-in and carry-out bits
  - **1-minute quiz**: given X, Y and C_{in}, what are the expressions of Z and C_{out}?
  - A: Derive the truth table from the manual addition method
    Then, derive the Boolean expressions for Z and C_{out}
  - $Z = X \oplus Y \oplus C_{in}$
  - $C_{out} = (X \cdot Y) + (X \oplus Y) \cdot C_{in}$

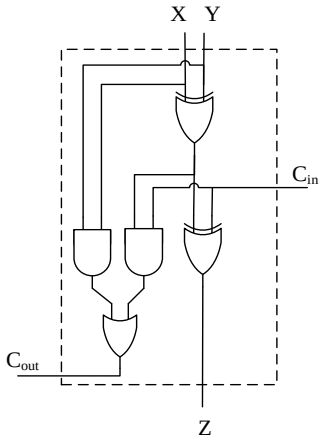
采用二进制加法原理导出真值表
继而导出输出Z和C_{out}的布尔表达式



Full adder symbol

Cin	X	Y	Z	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Truth table



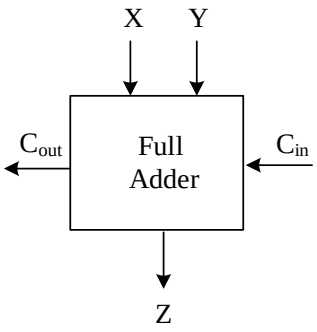
Implementation by gates

全加器是系统

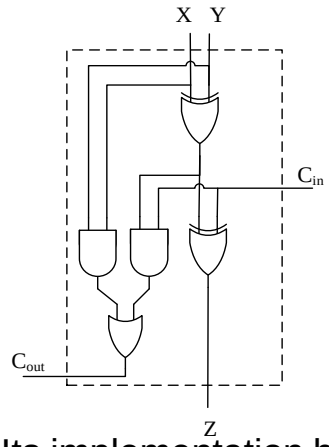
逻辑门是模块

Ripple-carry adder 波纹进位加法器
代表了设计方法2：串行连接多个模块

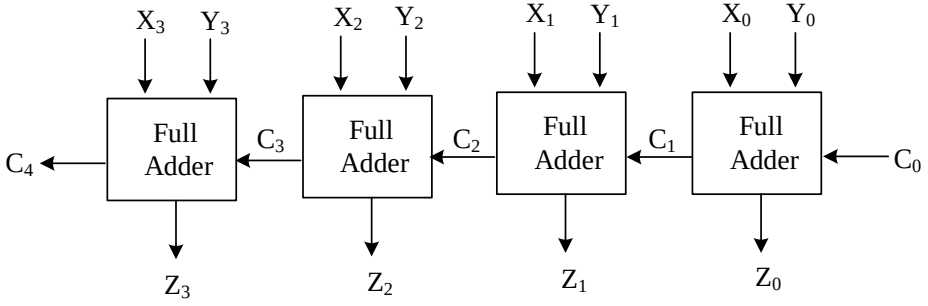
- Add two unsigned numbers X and Y to generate result sum Z
  - Consider the carry-in bit C_{in} and the carry-out bit C_{out}
  - Use the same algorithm we use when doing addition by pen and paper
- Design an n-bit ripple-carry adder (assuming n=4 in example)
  串联4个全加器（模块），实现一个4位波纹进位加法器（系统）
  - Use X+Y=1011+1001 = 10100 to verify that the 4-bit adder works correctly
  - 将模块组合成系统的抽象过程可能会丢失信息
  - **1-minute quiz**: How much time to do the addition? Use gate delay as the unit
    4位波纹进位加法器产生多少级门延迟？n位呢？



Full adder symbol
全加器是模块



Its implementation by gates

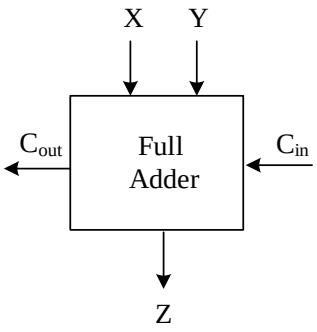


A 4-bit ripple-carry adder
4位波纹进位加法器是系统

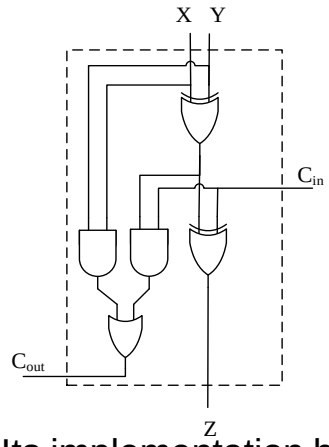
Ripple-carry adder

- For n-bit, design an n-bit ripple-carry adder (assuming n=4 in example)

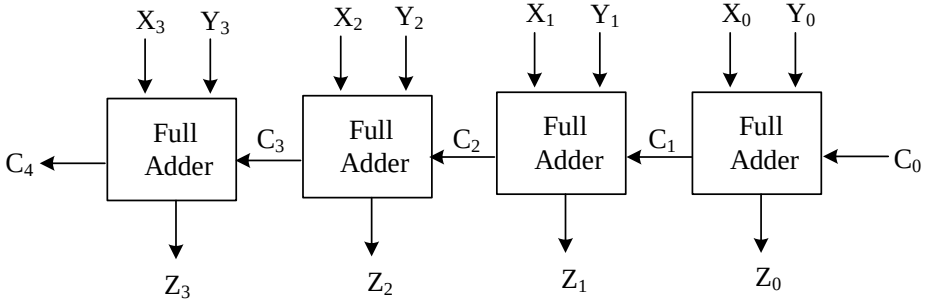
- 将模块组合成系统的抽象过程可能会丢失信息
  - **1-minute quiz**: How much time to do the addition? Use gate delay as the unit
  - 问题：**4**位波纹进位加法器产生多少级门延迟？**n**位呢？
  - Answer 1: total delay ~ **3n**; about 12 gate delays when n=4
    - Because each full adder needs 3 gate delays to generate carry-out
  - 答案：**4**位波纹进位加法器产生**4*3=12**级门延迟，因为每个全加器产生**3**级门延迟
  - Is this answer correct?



Full adder symbol



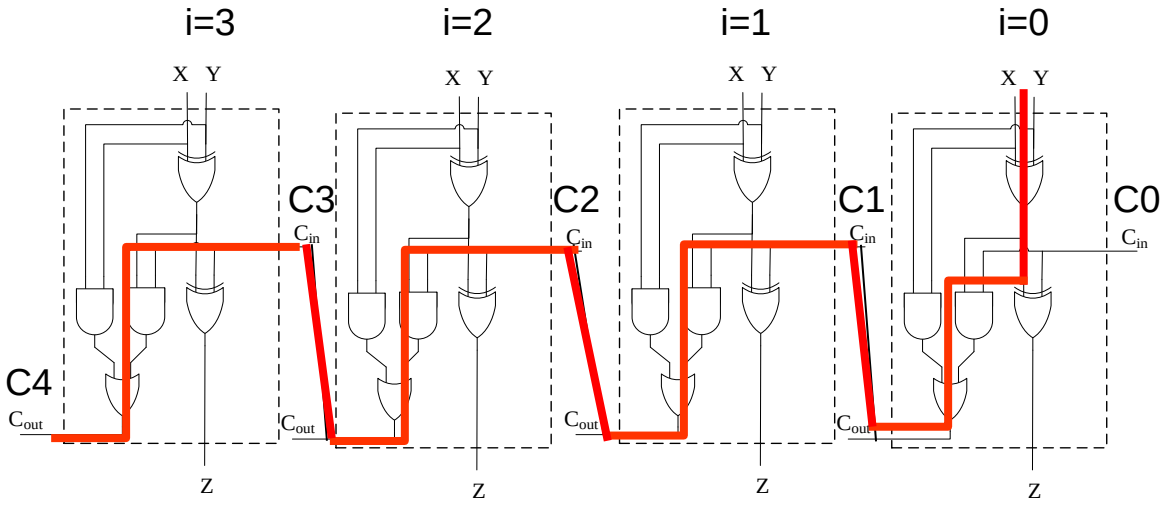
Its implementation by gates



A 4-bit ripple-carry adder

Ripple-carry adder

- Design an n-bit ripple-carry adder (assuming n=4 in example)
  - **1-minute quiz**: How much time to do the addition? Use gate delay as the unit
  - Answer 2: **2n+1**. There are 9 gate delays for n=4
    正确答案：**4位波纹进位加法器产生2*4+1=9级门延迟**
    因为当算出C_1时，$X_i \oplus Y_i$ 也已经被算出来了，$1 \le i < n$
    - Only C1 needs 3 gate delays, and another Ci needs only 2 additional gate delays.
    - Why? Because for $1 \le i < n$, $X_i \oplus Y_i$ is already generated when C_1 becomes available
      - Red line shows the longest path 红线展示了最长路径

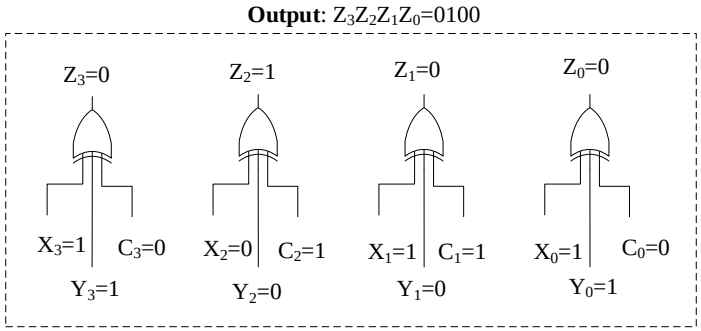


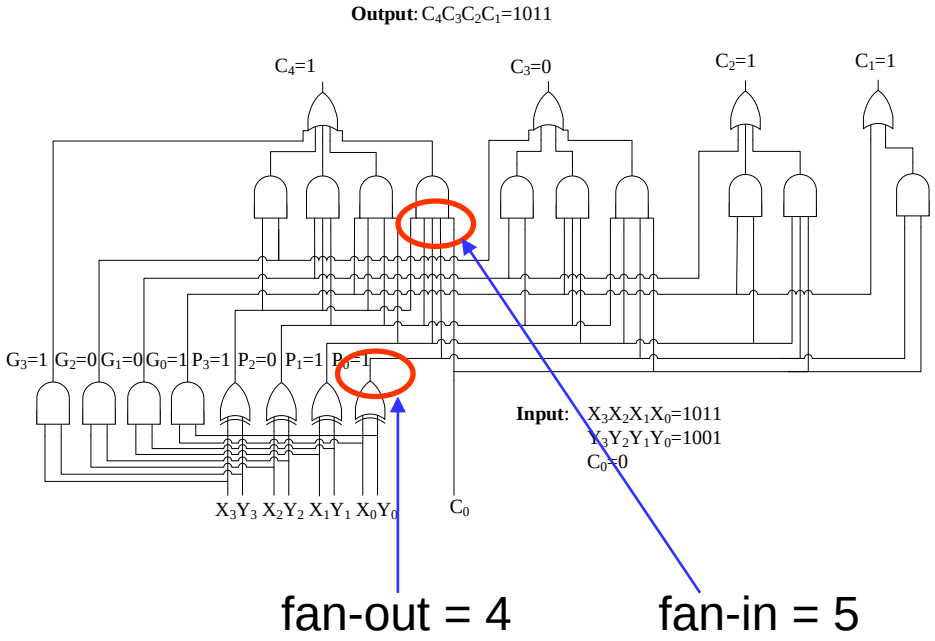
Expand to show the gate-level details

A much faster adder 并行进位的加法器

代表了设计方法3：并行连接，使用中间模块Gi和Pi

- Instead of compute the carry bits one by one in *n* steps, we can compute all the *n* carry bits in parallel in one step
  先算出全部进位比特值
  - This step needs 3 gate delays
- Afterwards, the *n* sum bits are computed in parallel in one step
  - This step needs 1 gate delay
- 4 gate delays in total
  只需4级门延迟　　4 vs. 2n-1
  - Compared to 2*n* -1 gate delays for ripple-carry adder
- Constrained by fan-in and fan-out in practice 扇入扇出限制
  - A gate can only safely receive a few inputs
  - A gate output cannot drive too many wires





4.2.4 A combinational circuit can compute multiple functions via selection by control signals

代表了设计方法4：使用多路复用器和控制信号（本例中的选择输入S）

- An adder-subtractor controlled by a multiplexer (MUX in short)
  - Control signal S to select addition (S=0) or subtraction (S=1)
  - Subtraction is adding negative, e.g., 5-5 = 5+(-5)
    - The negative value of a number X is the two's complement of X
  - Let X = Y = 5, i.e., $X_3X_2X_1X_0 = Y_3Y_2Y_1Y_0 = 0101$
    - Then, X – Y = 5 + (-5) = 5 + Two's complement of 5

→ $0101 + (\overline{0}\,\overline{1}\,\overline{0}\,\overline{1} + 0001)$

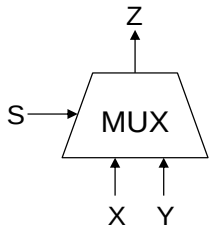
= 0101 + 1011

= 10000 = $C_4\, Z_3\, Z_2\, Z_1\, Z_0$

采用多路复用器的加减器，控制信号S选择做加法还是做减法

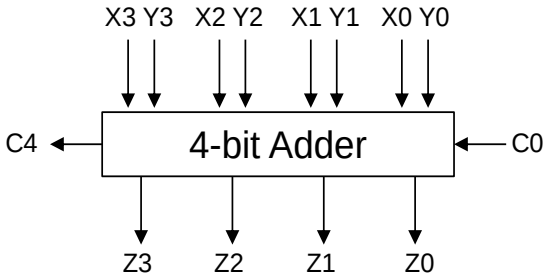
$$Z = \begin{cases} X & \text{if } S = 0 \\ Y & \text{if } S = 1 \end{cases}$$

S	X	Y	Z
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

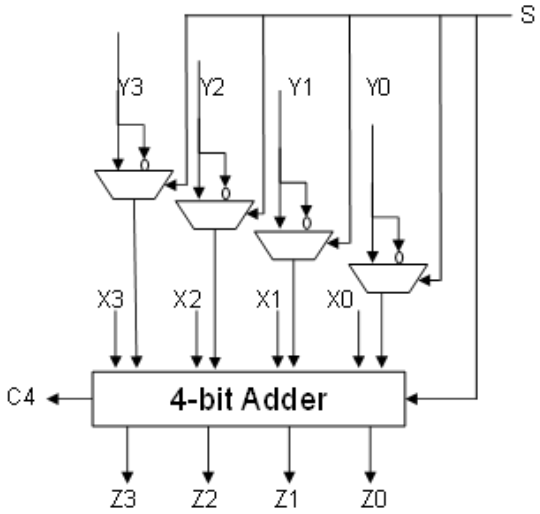


S	Z
0	X
1	Y

A multiplexer 多路复用器



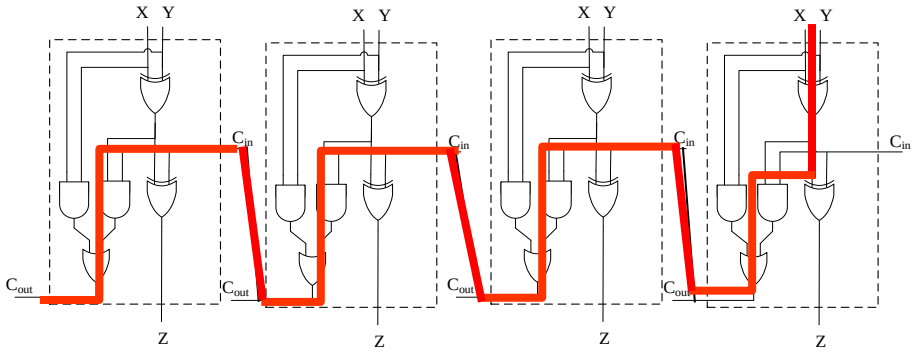
A two's complement 4-bit adder

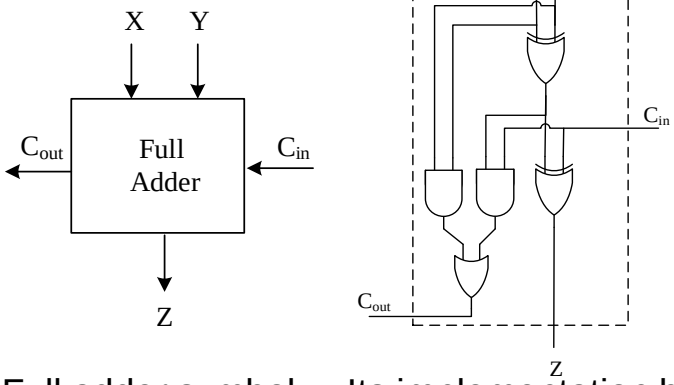


and an adder-subtractor

通过实例理解模块化的一般原理

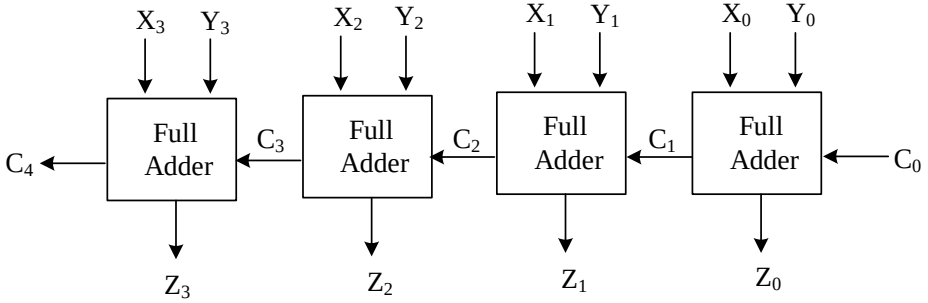
- 模块是采用信息隐藏原理的抽象
  - 全加器模块是一个组合电路抽象，隐藏了虚框内的逻辑电路细节
- 系统中的两个模块可以连接，但不重叠
  - 波纹进位加法器中的两个全加器可连接，但两个全加器的内部电路不重叠
- 模块化是艺术，需要人的创造力
  - 波纹进位加法器比较直接简单
  - 并行进位加法器则需要艺术：
    体现在中间表示P，G两种模块（留作练习）





Full adder symbol　　Its implementation by gates

全加器是**模块**



A 4-bit ripple-carry adder

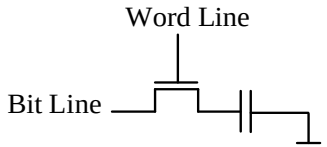
4位波纹进位加法器是**系统**

4.3 Sequential circuits

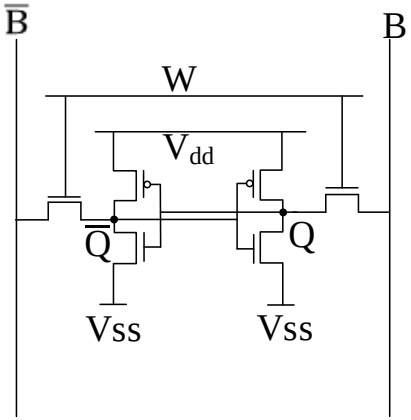
- **Sequential circuit = combinational circuit + state circuit**
  时序电路 = 组合电路 + 状态电路，以实现多步计算过程
  - With states, a system can execute multi-step computational processes
  - Each step computes two types of values
    - The current output values 当前输出
    - The next state values 下一状态
  - Both are computed from
    - The current input values 当前输入
    - The current state values 当前状态
- Correspondences to logic thinking abstractions 实现逻辑思维
  - A combinational circuit is equivalent to a Boolean expression
  - A sequential circuit is equivalent to an automaton
  - 组合电路实现布尔表达式；时序电路实现自动机
- States in hardware circuits are implemented by two types of basic circuits
  状态电路有两种实现方法：存储单元，触发器
  - Memory cells
  - flip-flops: logic circuits with feedback wires
    - We discuss only the D flip-flop

4.3.1 Types of memory technology 存储器类别

- Non-volatile memory (NVM): content is retained when power is off
  - 计算机下电后，非易失存储器中的内容不会丢失
  - ROM (read-only memory) 只读存储器　　例子：开机后读取数据与指令
  - Read-write NVM 可读可写的非易失存储器　　例子：U盘
- Volatile memory: content is lost when power is off
  - 计算机下电后，易失存储器中的内容会丢失
  - DRAM (dynamic random access memory) 动态随机访问存储器
    - Simple and inexpensive 成本低
    - Needs to constantly refresh its content (once every 7.8-128 μs)
    - Because the capacitor leaks electricity after charging
  - SRAM (static random access memory) 静态随机访问存储器
    - More complex but avoid refreshing overhead
    - Faster but more expensive than DRAM



DRAM cell
1 transistor and
1 capacitor
只需一个晶体管



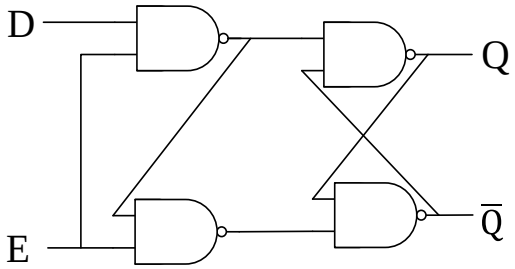
SRAM cell
6 transistors
需要6个晶体管

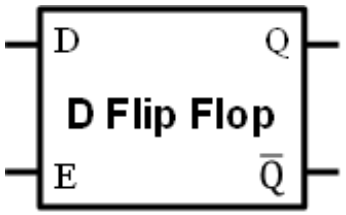
Type	Latency	Price $/GB
Register	100s ps	N/A
SRAM	100s ps ~ 10 ns	$100's ~1000s /GB
DRAM	10s ~ 100 ns	$2~4 /GB
NVM: Main Memory	100 ns ~ 10 μs	$6 / GB
NVM: SSD	10 μs ~ 1 ms	$0.1~0.2 / GB
Hard Disk: HDD	> A few ms	$0.02 / GB

4.3.2 D flip-flop D触发器

- We can also use gates to form state circuits
  - With feedback wires 触发器是 包含反馈线路的逻辑门电路
- A common type is the delay flip-flop, or D flip-flop
  - 2-input-2-output, where the 2 states (outputs) are negation of each other
    - Input D is data input; Input E (enable signal) is often the clock signal CLK
      状态Q值是数据输入D一个时钟周期前的值，即D延迟一周期后变成Q
  - Functionality: when enabled, Q is D delayed one clock cycle
    - When E=0, Q remains the same; when E=1, the next Q will be the current D

E	D	Q	Q_{next}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

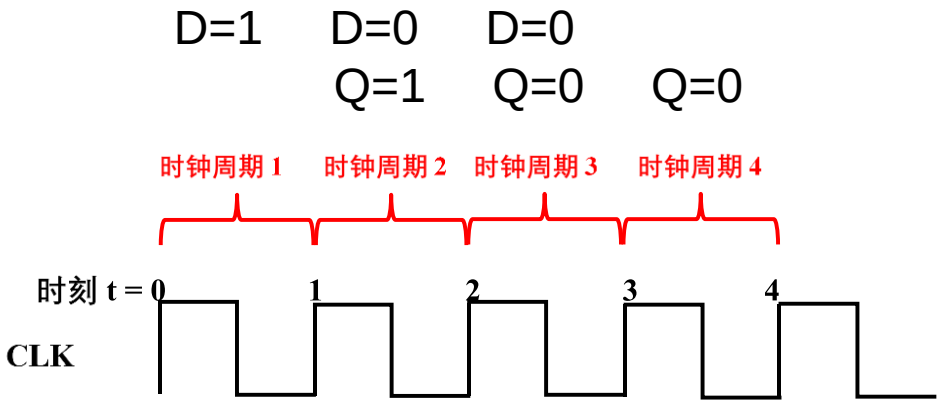




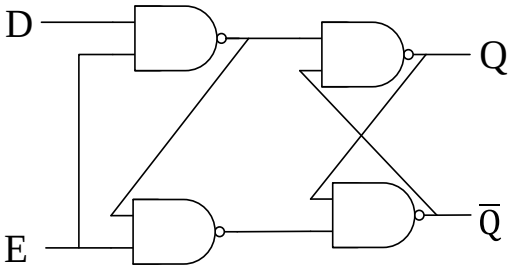
D flip-flop: Truth table Internal logic diagram Symbol

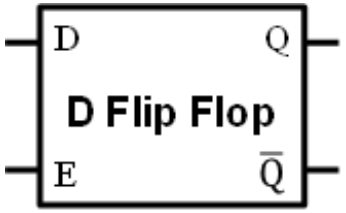
4.3.2 D flip-flop D触发器

- Input D is data input; **Input E (enable signal) is often the clock signal CLK**
- D是数据输入，E常常是时钟信号CLK
- 状态Q值是数据输入D一个时钟周期前的值，即D延迟一周期后变成Q



E	D	Q	Q_{next}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1



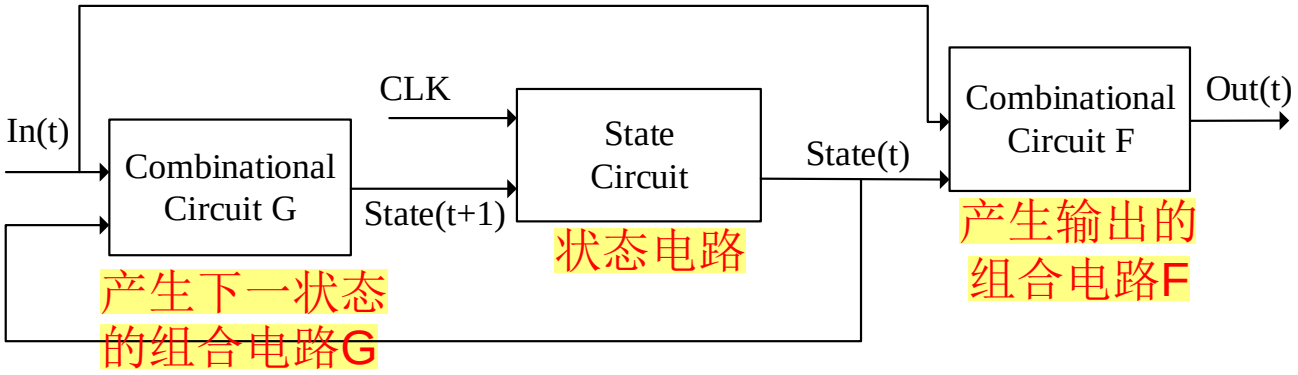


D flip-flop: Truth table　　Internal logic diagram　　Symbol

4.3.3 General organization of sequential circuits

时序电路的一般组成：两个组合电路，一个状态电路；时钟信号驱动

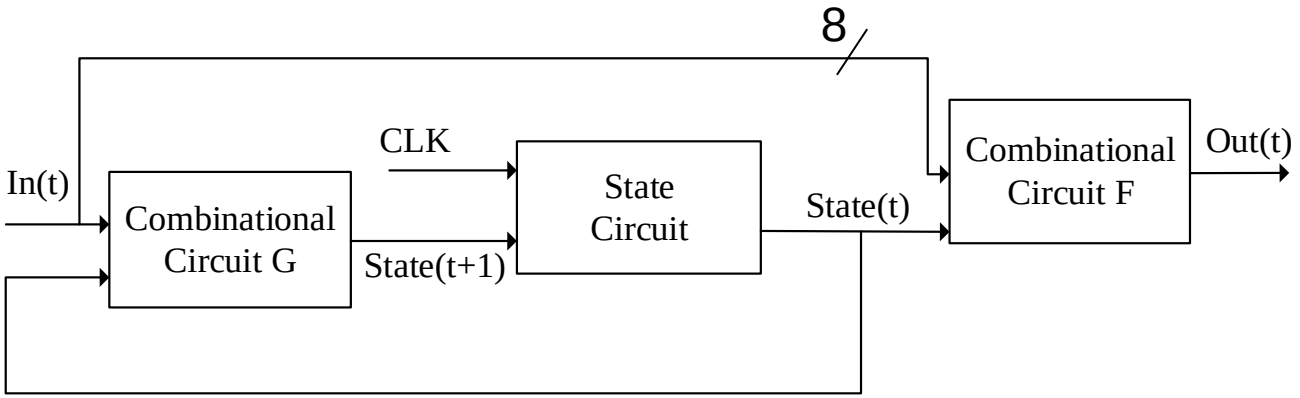
- Any sequential circuit can be organized as shown below
  - Comprised of two combinational circuits and a state circuit
  - Driven by a clock signal CLK
  - The state circuit consists of one or more D flip-flops
    - The output circuit F: Out(t) = **F**(In(t), State(t)) 输出电路
    - The next-state circuit G: State(t+1) = **G**(In(t), State(t)) 下一状态电路
- Logic diagram of a sequential circuit 时序电路的逻辑线路图
  - Also called synchronous sequential circuit
    - Because the sequential circuit is synchronized by the clock signal



4.3.3 General organization of sequential circuits

- Any sequential circuit can be organized as shown below
  - Comprised of two combinational circuits and a state circuit
  - Driven by a clock signal CLK
  - The state circuit consists of one or more D flip-flops
  - The output circuit F: Out(t) = F(In(t), State(t))
  - The next-state circuit G: State(t+1) = G(In(t), State(t))

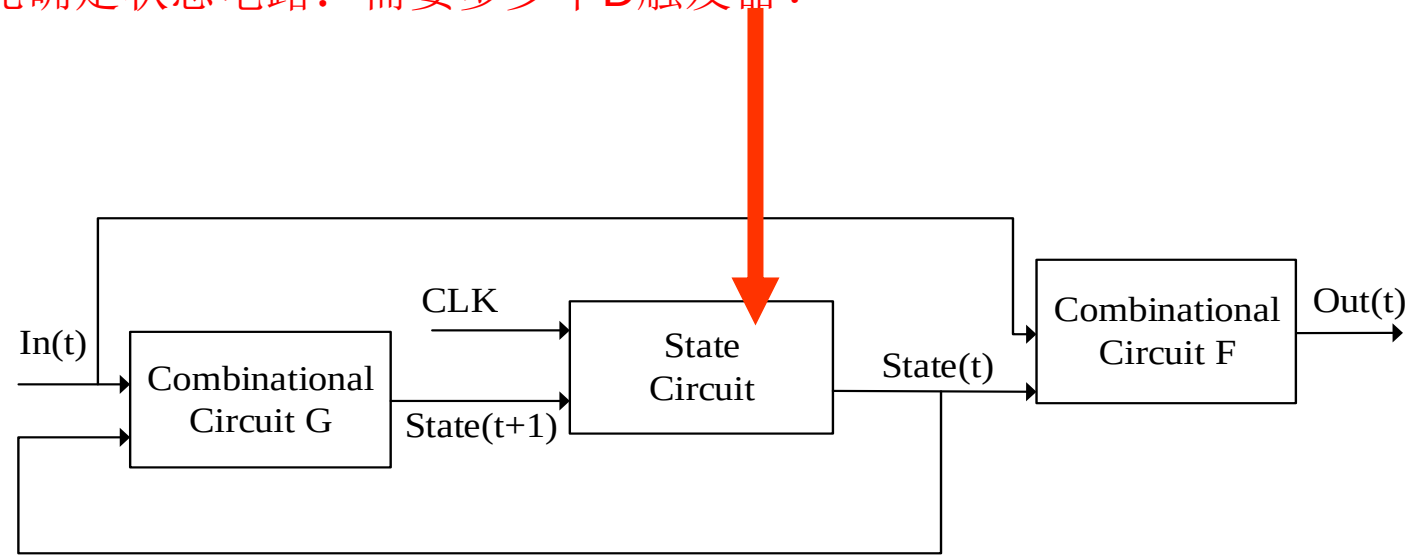
  - An input or output may have multiple bits （此处输入In是8条线）



We will go through the process of designing a 4-bit serial adder

4.3.4 Design a 4-bit serial adder
设计4位串行加法器

- Perform $Z_3Z_2Z_1Z_0 = X_3X_2X_1X_0 + Y_3Y_2Y_1Y_0$ in 4 steps
  需要四步（四个时钟周期）
  - Each step performs a full addition 每一步执行一个全加操作
- Verify: $11_{10} + 9_{10} = 1011_2 + 1001_2 = 10100_2 = 20_{10} = 4_{10}$ and overflow
  用一个实例验证： 输入X=1011，Y=1001；则输出Z=0100且产生溢出
- Design process 设计过程
  - Q: how many bits (or D flip-flops) are needed for the state circuit?
  - 首先确定状态电路：需要多少个D触发器？



Design process

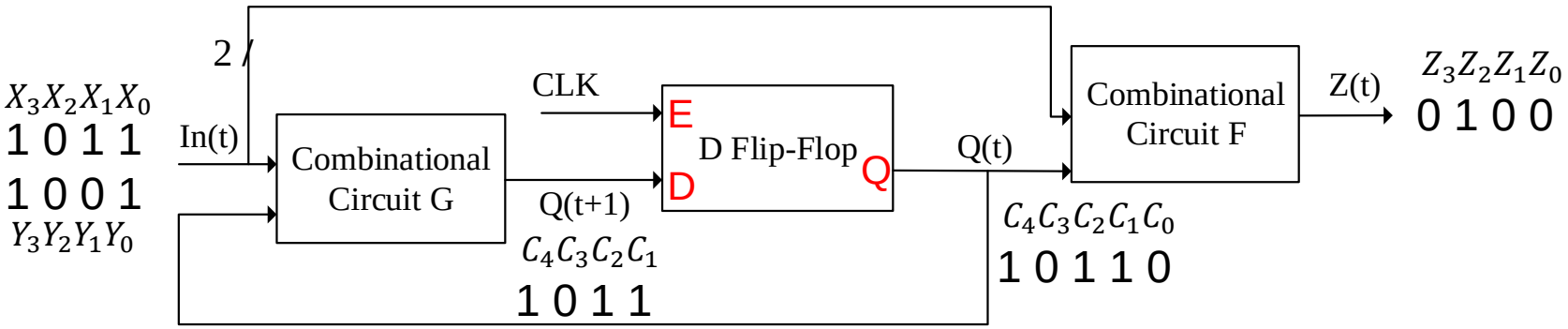
- Design process
  - Q: how many bits (or D flip-flops) are needed for the state circuit?
    - A: 1 bit to hold the carry bit. **Only one D flip-flop is needed**. Denote the state as Q.

    Clock signal as the Enable signal

    只需要一个 **D触发器，其状态Q表示当前进位，其Enable端连接时钟信号CLK**
  - Draw the logic diagram for the sequential circuit

    确定状态电路后，只需要画出组合电路**G、F**（确定它们的布尔表达式）
    - How?

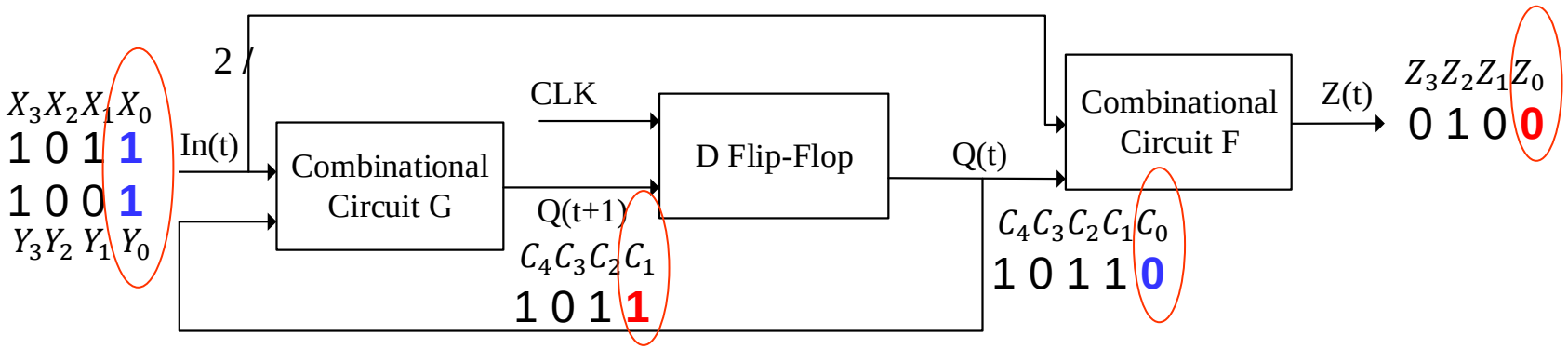
      使用加法原理理解串行加法器的时序电路
    - 某些同学更喜欢使用实例来理解



Design process

- Draw the logic diagram for the sequential circuit
  - 确定组合电路G、F的布尔表达式
  - How?
    使用实例和加法原理理解串行加法器的时序电路
    - Step 1: inputs in blue, outputs in red. Note that $C_0 = 0$

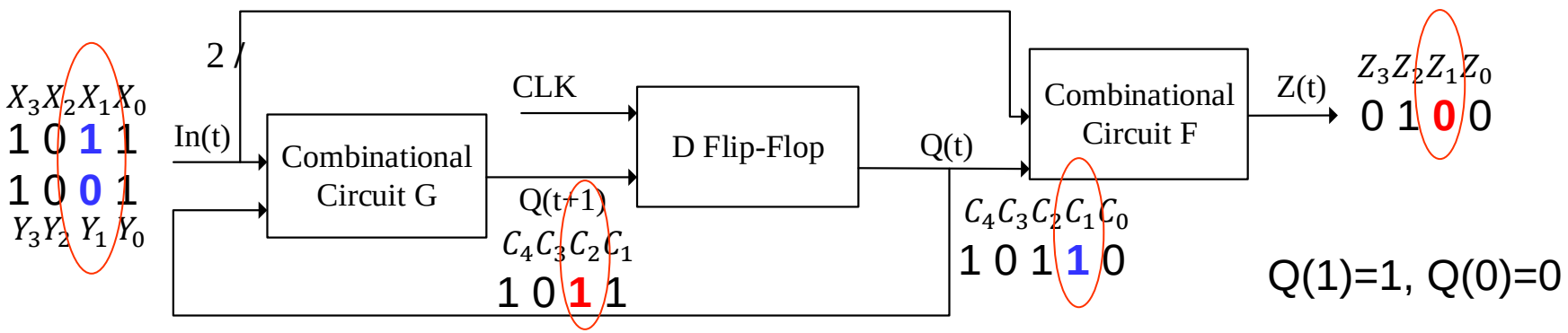
$$
\begin{array}{rr}
 & 1\,0\,1\,\mathbf{1} = X \\
+ & 1\,0\,0\,\mathbf{1} = Y \\
\hline
 & 1\,0\,1\,\mathbf{1}\,\mathbf{0} = C \\
 & 0\,1\,0\,\mathbf{0} = Z
\end{array}
$$



Design process

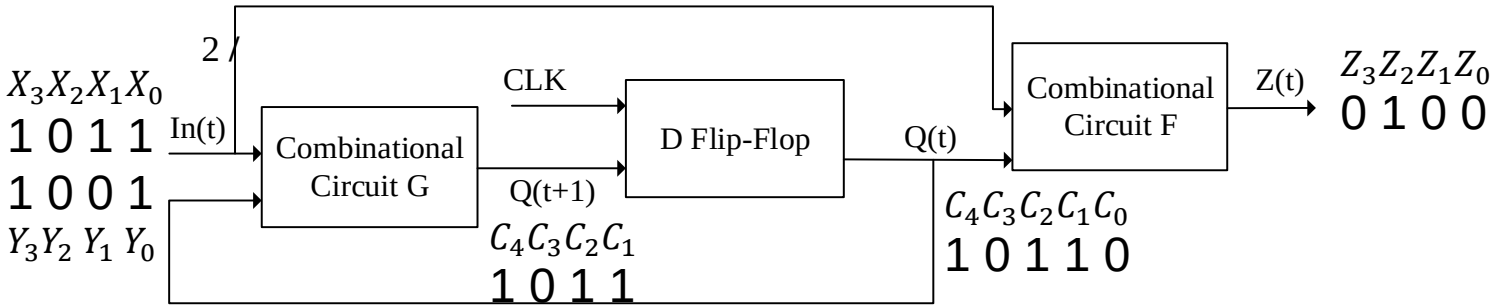
- Draw the logic diagram for the sequential circuit
  - 确定组合电路G、F的布尔表达式
  - How?
    使用实例和加法原理理解串行加法器的时序电路
    - Step 2: inputs in blue, outputs in red.

1 0 **1** 1 = X
\+ 1 0 **0** 1 = Y
1 0 **1** **1** 0 = C
0 1 **0** 0 = Z

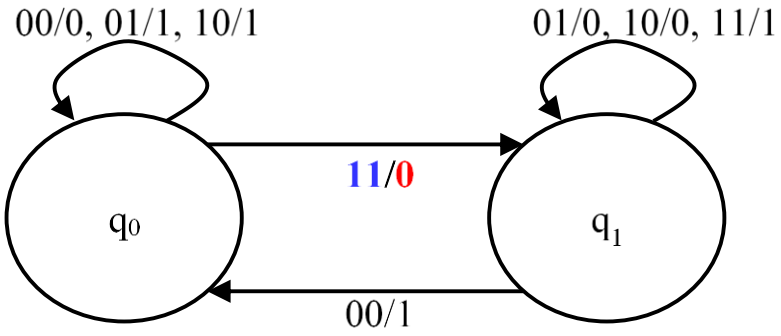


Design process

- Draw the logic diagram for the sequential circuit
  - Derive the state-transition diagram 画出时序电路的状态转换图



XY/Z



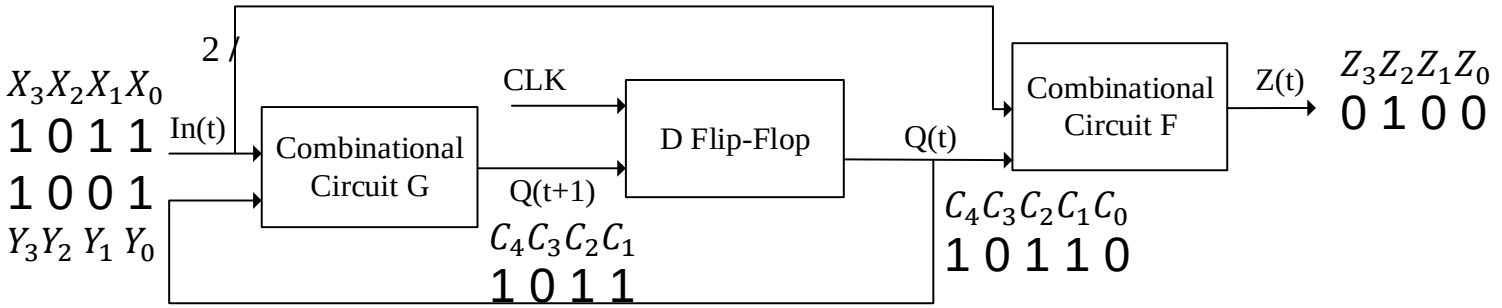
画出时序电路的状态转换图

q_0 denotes C=0
q_1 denotes C=1

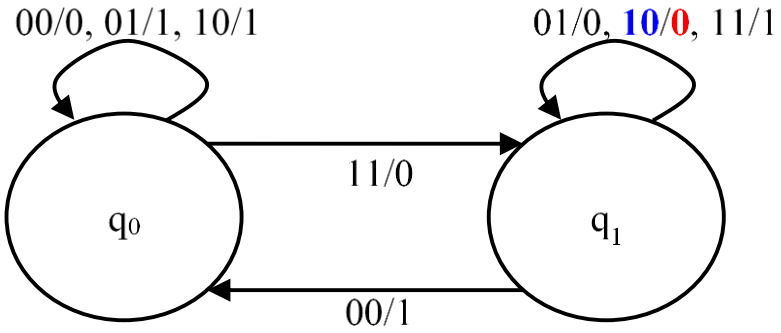
	1 0 1 **1** = X
+	1 0 0 **1** = Y
	1 0 1 **1** **0** = C
	0 1 0 **0** = Z

Design process

- Draw the logic diagram for the sequential circuit
  - Derive the state-transition diagram



XY/Z



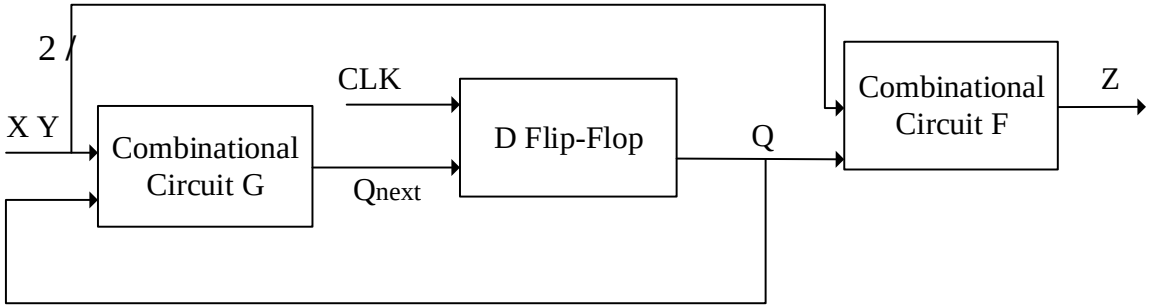
画出时序电路的状态转换图

q_0 denotes C=0
q_1 denotes C=1

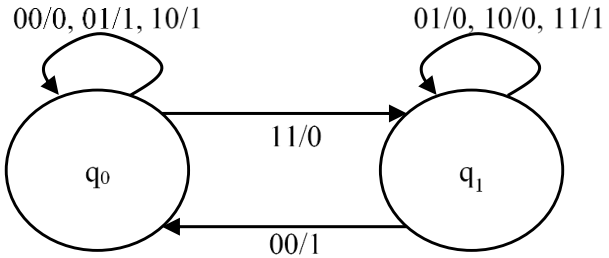
```
  1 0 1 1 = X
+ 1 0 0 1 = Y
-----------
1 0 1 1 0 = C
  0 1 0 0 = Z
```

Design process

- Derive the state-transition diagram (remember that Q denotes carry C)
- Derive the truth table and Boolean expressions for F, G



XY/Z



q_0 denotes C=0
q_1 denotes C=1

Q	X	Y	Z	Q_{next}
q_0	0	0	0	q_0
q_0	0	1	1	q_0
q_0	1	0	1	q_0
q_0	1	1	0	q_1
q_1	0	0	1	q_0
q_1	0	1	0	q_1
q_1	1	0	0	q_1
q_1	1	1	1	q_1

进而导出真值表
进而求出组合电路F和G
的布尔逻辑表达式

$$Z = F(X, Y, Q) = X \oplus Y \oplus C$$

$$Q_{next} = G(X, Y, Q) = (X \cdot Y) + (X \oplus Y) \cdot Q$$

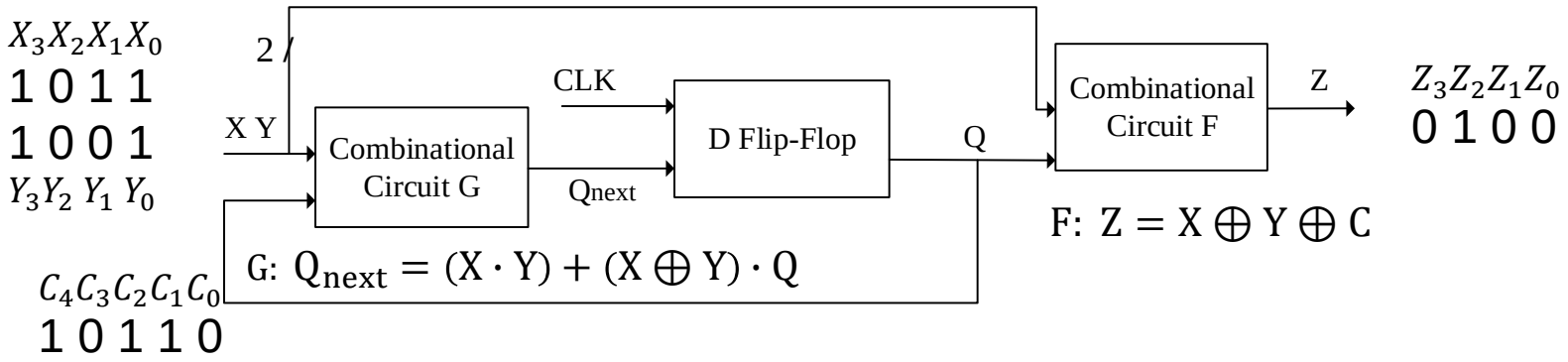
Design process

- Derive the state-transition diagram (remember that Q denotes carry C)
- Derive the truth table and Boolean expressions for F, G
- Verify the resulting sequential circuit,
  given $X_3X_2X_1X_0 = 1011$, $Y_3Y_2Y_1Y_0 = 1001$, $C_0 = 0$

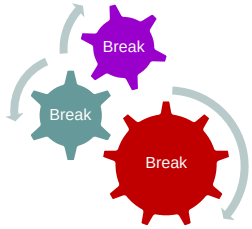
  - Step 1: $X_0 = 1, Y_0 = 1, C_0 = \mathbf{0}.\ Z_0 = \mathbf{0}, C_1 = \mathbf{1}$
  - Step 2: $X_1 = 1, Y_1 = 0, C_1 = \mathbf{1}.\ Z_1 = \mathbf{0}, C_2 = \mathbf{1}$
  - Step 3: $X_2 = 0, Y_2 = 0, C_2 = \mathbf{1}.\ Z_2 = \mathbf{1}, C_3 = \mathbf{0}$
  - Step 4: $X_3 = 1, Y_3 = 1, C_3 = \mathbf{0}.\ Z_3 = \mathbf{0}, C_4 = \mathbf{1}$

最后，用实例验证

- The final result is $Z_3Z_2Z_1Z_0 = 0100$, with $C_4 = 1$ indicating overflow



Take-Home Messages



- Modularization
  - Divide a system into multiple subsystems (modules), and compose modules into a system (higher-level abstraction), using information hiding
  - Understand modularization via a design journey
- Combinational circuits are circuits of gates without feedback wires
  - Equivalent to Boolean expression
    - How many combinational circuits are there, of n-input-1-output?
  - Boolean expressions and logic diagrams are simpler than CMOS circuits
  - Examples: adders, adder-subtractor with multiplexer
- Sequential circuits = state circuits + combinational circuits
  - State can be represented by a memory cell or a flip-flop
  - There are different types of memory, with different latencies and costs
    - Non-volatile memory (NVM): ROM (read-only memory), Read-write NVM
    - Volatile memory: DRAM, SRAM
  - A flip-flop, such as D flip-flop, is a circuit of gates with feedback wires
  - Design a serial subtractor to see how a sequential circuit generates the output and the next state from the inputs and the current state