



澳門城市大學
Universidade da Cidade de Macau
City University of Macau

計算機科學導論



主講人 |

姓名 張琪

Name Zhang Qi


澳門城市大學

City University of Macau



第十一章 數字邏輯基礎

本章學習要點：

- 1 數字系統的基本概念
 - 2 常用計數制及其轉換
 - 3 帶符號二進制數的代碼表示
 - 4 邏輯代數的基本概念
 - 5 邏輯代數的基本定理和規則
 - 6 複合邏輯
- 



11.1 數字系統的基本概念

● 1.1.1 數字系統概述

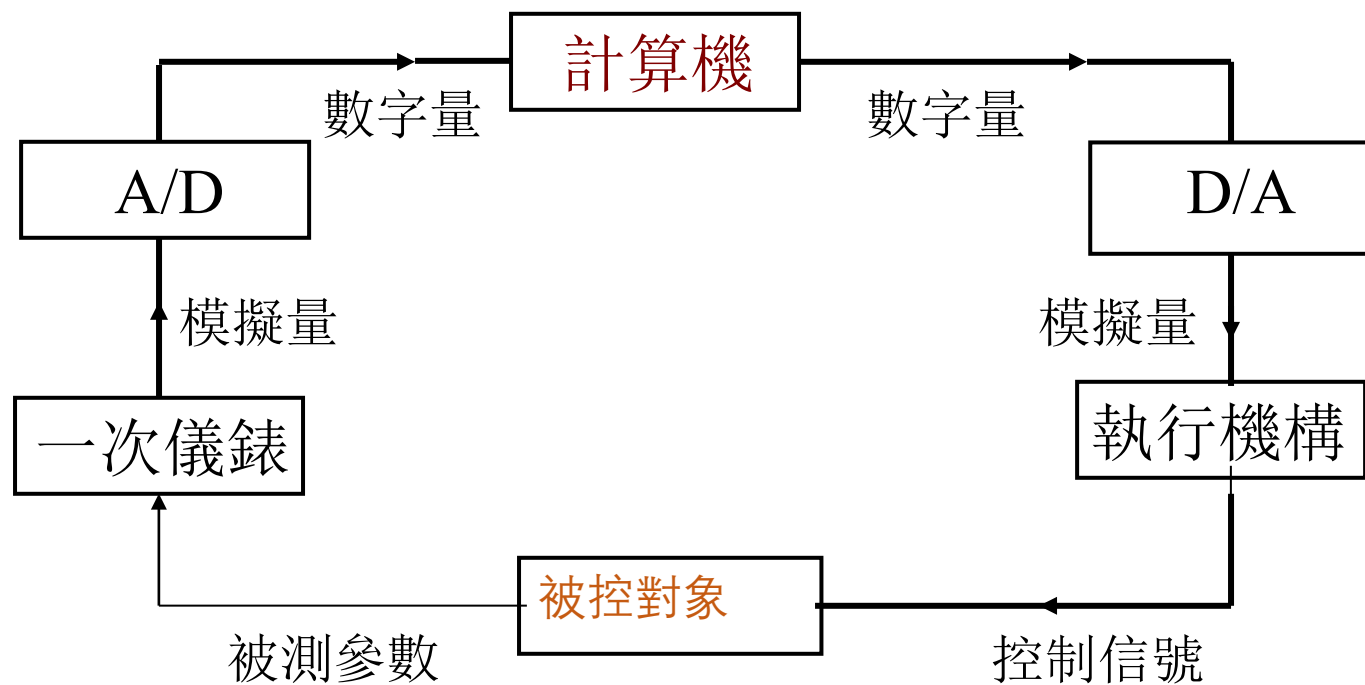
- 一、數字信號
- 若信號的變化在時間上和數值上都是離散的，或者說斷續的，則稱為離散信號。離散信號的變化可以用不同的數字反映，所以又稱為數字信號，簡稱為數字量
- 例如，學生成績記錄，工廠產品統計, 電路開關的狀態等
- 數字系統中處理的是數字信號，當數字系統要與模擬信號發生聯繫時，必須經過模/數(A/D)轉換和數/模(D/A)轉換電路，對信號類型進行變換



11.1 數字系統的基本概念

● 1.1.1 數字系統概述

- 一、數字信號
- 例如,某控制系統框圖如下：



11.1 數字系統的基本概念

● 1.1.1 數字系統概述

● 二、數字電路

- 用來處理數字信號的電子綫路稱為數字電路。由于數字電路的各種功能是通过邏輯運算和邏輯判斷來實現的，所以數字電路又稱為數字邏輯電路或者邏輯電路
- 數字邏輯電路具有如下特點：
 - (1) 電路的基本工作信號是二值信號。它表現為電路中電壓的“高”或“低”、開關的“接通”或“斷開”、晶體管的“導通”或“截止”等兩種穩定的物理狀態
 - (2) 電路中的半導體器件一般都工作在開、關狀態
 - (3) 電路結構簡單、功耗低、便于集成製造和系列化生產；產品價格低廉、使用方便、通用性好
 - (4) 由數字邏輯電路構成的數字系統工作速度快、精度高、功能強、可靠性好
- 由于數字邏輯電路具有上述特點，所以，數字邏輯電路的應用十分廣泛

11.1 數字系統的基本概念

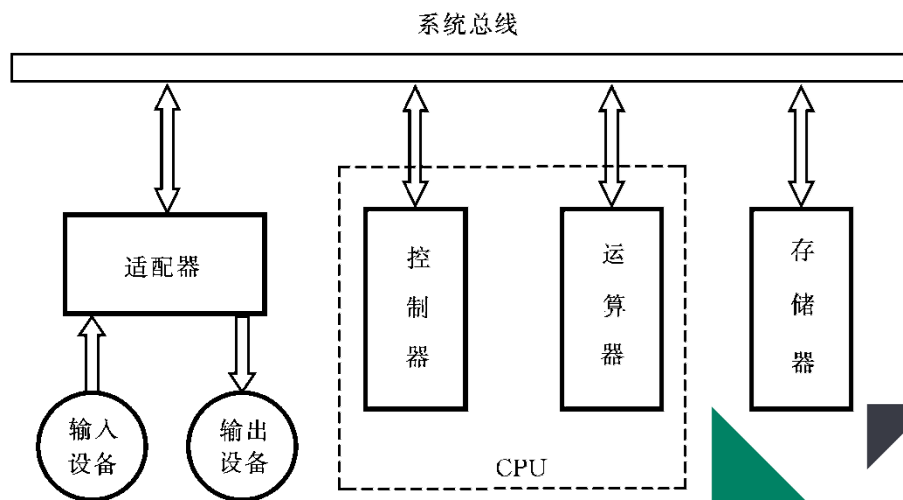
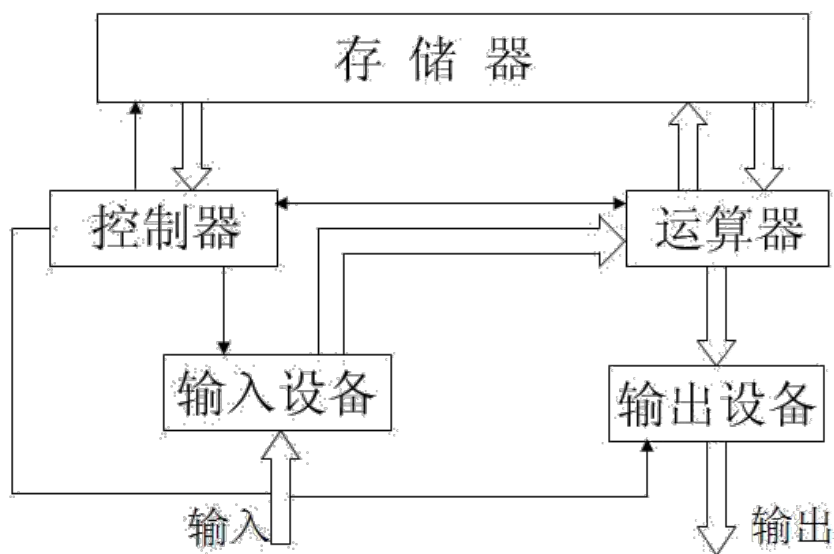
● 1.1.1 數字系統概述

● 三、數字系統

● 1. 典型的數字系統——數字計算機

● 數字計算機是一種能夠自動、高速、精確地完成數值計算、數據加工和控制、管理等功能的數字系統

● 結構框圖如下：



11.1 數字系統的基本概念

● 1.1.1 數字系統概述

● 三、數字系統

● 2. 計算機的發展

- 數字計算機從1946年問世以來，其發展速度是驚人的。根據組成計算機的主要元器件的不同，至今已經歷了四代。具體如下表所示
- 發展趨勢：速度↑、功能↑、可靠性↑、體積↓、價格↓、功耗↓

數字計算機的劃代

劃代	主要元器件	生產時間	國家
第一代	電子管	1946年	美國
第二代	晶體管	1958年	美國
第三代	小規模集成電路	1964年	美國
第四代	中、大規模集成電路	1971年	美國

11.1 數字系統的基本概念

● 1.1.1 數字系統概述

- 你瞭解組成各代計算機的主要元器件嗎？不妨看一下有關圖片

電子管



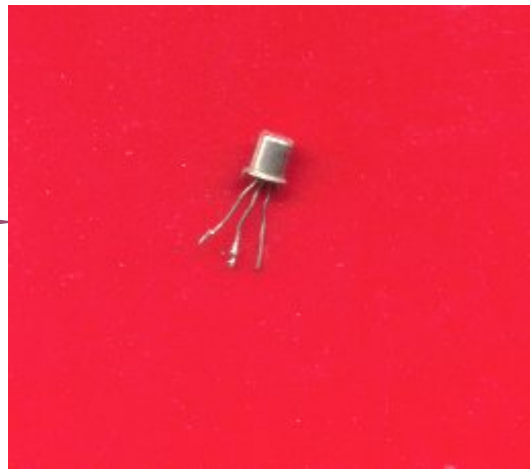
電子管是第一代計算機的主要元器件

11.1 數字系統的基本概念

● 1.1.1 數字系統概述

- 你瞭解組成各代計算機的主要元器件嗎？不妨看一下有關圖片

晶體管



晶體管是第二代計算機的主要元器件

11.1 數字系統的基本概念

● 1.1.1 數字系統概述

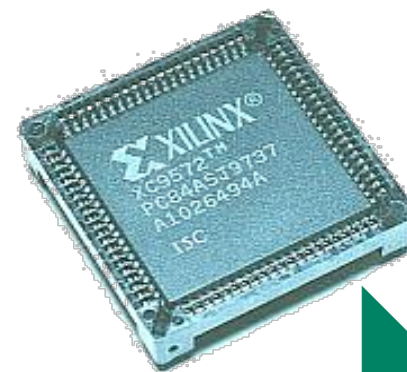
- 你瞭解組成各代計算機的主要元器件嗎？不妨看一下有關圖片

小規模
集成電路



小規模集成電路是第三代計算機的主要元器件

大規模
集成電路



中大規模集成電路的出現，導致了第四代計算機問世

11.1 數字系統的基本概念

● 1.1.1 數字系統概述

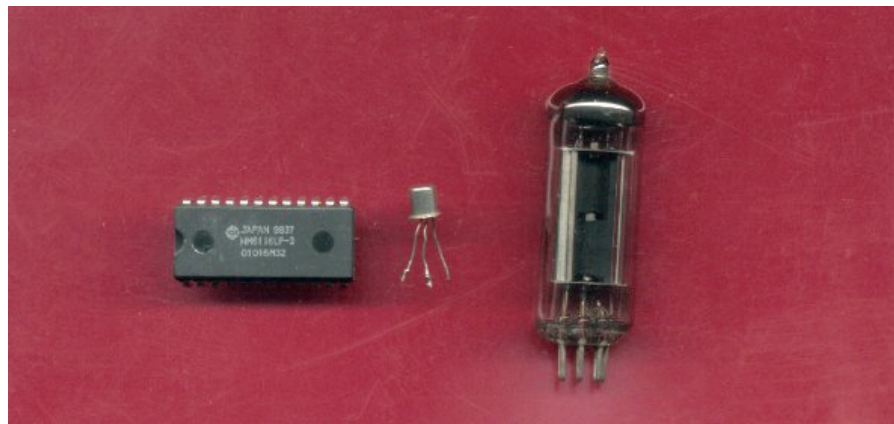
- 廣泛使用的微型計算機、單片機是建立在超大規模集成電路基礎上的。其CPU的集成規模如何？
- 以 PC機CPU芯片80×86系列為例：
- 500個晶體管串起來，才能繞頭髮絲一周

型號	集成度
8086	2.9萬個晶體管
80286	13.5萬個晶體管
80386	32萬個晶體管
80486	120萬個晶體管
80586	320萬個晶體管
...	...

11.1 數字系統的基本概念

● 1.1.1 數字系統概述

- 發展趨勢：速度↑、功能↑、可靠性↑、體積↓、價格↓、功耗↓



想一想！
比較一下！





11.1 數字系統的基本概念

- 1.1.1 數字系統概述

- 3. 數字系統的定義

- 什麼是數字系統？數字系統是一個能對數字信號進行加工、傳遞和存儲的實體，它由實現各種功能的數字邏輯電路相互連接而成





11.1 數字系統的基本概念

● 1.1.2 數字電路的分類

- 1.按功能分類
- 根據一個電路是否具有記憶功能，可將數字邏輯電路分為組合邏輯電路和時序邏輯電路兩種類型
- 組合邏輯電路：如果一個邏輯電路在任何時刻的穩定輸出僅取決于該時刻的輸入，而與電路過去的輸入無關，則稱為組合邏輯 (Combinational Logic) 電路
- 由于這類電路的輸出與過去的輸入信號無關，所以不需要有記憶功能。例如，一個“多數表決器”，表決的結果僅取決于參予表決的成員當時的態度是“贊成”還是“反對”，因此屬組合電路





11.1 數字系統的基本概念

● 1.1.2 數字電路的分類

● 1.按功能分類

- **時序邏輯電路**：如果一個邏輯電路在任何時刻的穩定輸出不僅取決於該時刻的輸入，而且與過去的輸入相關，則稱為時序邏輯 (Sequential Logic) 電路
- 由於這類電路的輸出與過去的輸入相關，所以要用電路中記憶元件的狀態來反映過去的輸入信號。例如，一個統計串行輸入脈衝信號個數的“計數器”，它的輸出結果不僅與當時的輸入脈衝相關，還與前面收到的脈衝個數相關，因此，計數器是一個時序邏輯電路
- 時序邏輯電路按照是否有統一的時鐘信號進行同步，又可進一步分為同步時序邏輯電路和异步時序邏輯電路





11.1 數字系統的基本概念

- 1.1.2 數字電路的分類

- 2. 按規模分類

- 隨著半導體技術和工藝的發展，出現了數字集成電路，集成電路發展十分迅速
- 數字集成電路按照集成度的高低可分為小規模（SSI）、中規模（MSI）、大規模（LSI）和超大規模（VLSI）幾種類型





11.1 數字系統的基本概念

- 1.1.3 數字邏輯電路的研究方法
- 2.按規模分類
- 對數字系統中邏輯電路的研究有兩個主要任務：一是分析，二是設計
- 邏輯分析：研究一個已有邏輯電路的邏輯功能和性能
- 邏輯設計：根據提出的邏輯功能，在給定條件下構造出實現預定功能的邏輯電路稱為邏輯設計，或者邏輯綜合
- 注意：邏輯電路分析與設計的方法隨著集成電路的迅速發展在不斷發生變化



11.1 數字系統的基本概念

● 1.1.3 數字邏輯電路的研究方法

● 傳統法

- 傳統法：傳統方法是建立在小規模集成電路基礎之上的，它以技術經濟指標作為評價一個設計方案優劣的主要性能指標，設計時追求的目標是如何使一個電路達到最簡
- 如何達到最簡呢？在組合邏輯電路設計時，通過邏輯函數化簡，盡可能使電路中的邏輯門和連線數目達到最少。而在時序邏輯電路設計時，則通過狀態化簡和邏輯函數化簡，盡可能使電路中的觸發器、邏輯門和連線數目達到最少
- 注意：一個最簡的方案並不等於一個最佳的方案
- 以邏輯代數作為基本理論的方法始終是最基本的方法



11.1 數字系統的基本概念

- 1.1.3 數字邏輯電路的研究方法
- 採用中、大規模集成組件進行邏輯設計的方法
- 由于中、大規模集成電路的不斷發展，使芯片內部容納的邏輯元器件越來越多，因而，實現某種邏輯功能所需要的門和觸發器數量已不再成為影響經濟指標的突出問題
- 如何採用各種廉價的中、大規模集成組件去構造滿足各種功能的邏輯電路，尋求經濟合理的方案？必須注意：
 - 充分瞭解各種器件的邏輯結構和外部特性，做到合理選擇器件
 - 充分利用每一個已選器件的功能，用靈活多變的方法完成各類電路或功能模塊的設計
 - 盡可能減少芯片之間的相互連線





11.1 數字系統的基本概念

- 1.1.3 數字邏輯電路的研究方法
- 用PLD進行邏輯設計的方法
- 各類可編程邏輯器件(PLD)的出現，給邏輯設計帶來了一種全新的方法。人們不再用常規硬綫連接的方法去構造電路，而是借助豐富的計算機軟件對器件進行編程燒錄來實現各種邏輯功能，給邏輯設計帶來了極大的方便
- 電子設計自動化 (EDA)
- 面對日益複雜的集成電路芯片設計和數字系統設計，人們不得不越來越多地借助計算機進行輔助邏輯設計。目前，已進入電子設計自動化階段，不少人認為EDA技術已成為計算機科學中的一個獨立的學科

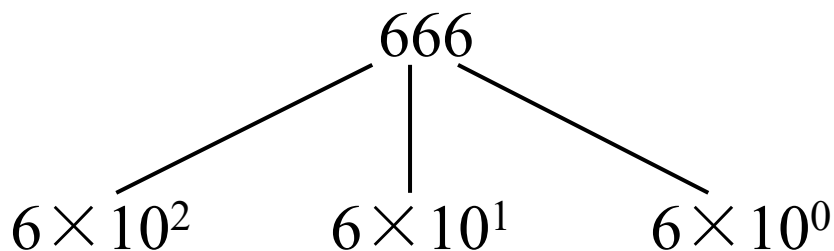


11.2 常用計數制及其轉換

● 1.2.1 進位計數制

- 數制是人們對數量計數的一種統計規律。日常生活中廣泛使用的是十進制，而數字系統中使用的是二進制
- 一、十進制
- 十進制中採用了0、1、...、9共十個基本數字符號，進位規律是“逢十進一”。當用若干個數字符號并在一起表示一個數時，處在不同位置的數字符號，其值的含意不同

如



- 同一個字符6從左到右所代表的值依次為600、60、6

11.2 常用計數制及其轉換

● 1.2.1 進位計數制

- 數制是人們對數量計數的一種統計規律。日常生活中廣泛使用的是十進制，而數字系統中使用的是二進制
- 二. R進制
- 廣義地說，一種進位計數制包含著基數和位權兩個基本的要素：
- **基數**：指計數制中所用到的數字符號的個數。在基數為 R 計數制中，包含 0 、 1 、...、 $R-1$ 共 R 個數字符號，進位規律是“逢 R 進一”。稱為 R 進位計數制，簡稱 R 進制
- **位權**：是指在一種進位計數制表示的數中，用來表明不同數位上數值大小的一個固定常數。不同數位有不同的位權，某一個數位的數值等于這一位的數字符號乘上與該位對應的位權。 R 進制數的位權是 R 的整數次幂

11.2 常用計數制及其轉換

● 1.2.1 進位計數制

- 數制是人們對數量計數的一種統計規律。日常生活中廣泛使用的是十進制，而數字系統中使用的是二進制
- 二． R進制
- R進制的特點可歸納如下：
- (1) 有0、1、...、 $R-1$ 共 R 個數字符號；
- (2) “逢 R 進一”，“10”表示 R ；
- (3) 位權是 R 的整數次幂，第 i 位的權為 R^i ($-m \leq i \leq n-1$)

11.2 常用計數制及其轉換

● 1.2.1 進位計數制

- 數制是人們對數量計數的一種統計規律。日常生活中廣泛使用的是十進制，而數字系統中使用的是二進制
- 三、二進制
- 基數 $R=2$ 。二進制數中只有0和1兩個基本數字符號，進位規律是“逢二進一”。二進制數的位權是2的整數次冪
- 任意一個二進制數 N 可以表示成

$$\begin{aligned}(N)_2 &= (K_{n-1}K_{n-2}\dots K_1K_0.K_{-1}K_{-2}\dots K_{-m})_2 \\&= K_{n-1} \times 2^{n-1} + K_{n-2} \times 2^{n-2} + \dots + K_1 \times 2^1 + K_0 \times 2^0 \\&\quad + K_{-1} \times 2^{-1} + K_{-2} \times 2^{-2} + \dots + K_{-m} \times 2^{-m} \\&= \sum_{i=-m}^{n-1} K_i 2^i\end{aligned}$$

其中： n —整數位數； m —小數位數； K_i —為0或者1， $-m \leq i \leq n-1$

11.2 常用計數制及其轉換

● 1.2.1 進位計數制

● 三、二進制

例如，一個二進制數1011.01可以表示成：

$$(1011.01)_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2}$$

二進制數的運算規則如下：

加法規則 $0+0=0$ $0+1=1$
 $1+0=1$ $1+1=0$ (進位為1)

減法規則 $0-0=0$ $1-0=1$
 $1-1=0$ $0-1=1$ (借位為1)

乘法規則 $0 \times 0=0$ $0 \times 1=0$
 $1 \times 0=0$ $1 \times 1=1$

除法規則 $0 \div 1=0$ $1 \div 1=1$

11.2 常用計數制及其轉換

● 1.2.1 進位計數制

● 三、二進制

- 例如，二進制數 $A=11001$, $B=101$, 則 $A+B$ 、 $A-B$ 、 $A \times B$ 、 $A \div B$ 的運算為

$$\begin{array}{r} 11001 \\ + 101 \\ \hline \end{array}$$

$$11110$$

$$\begin{array}{r} 11001 \\ \times 101 \\ \hline \end{array}$$

$$\begin{array}{r} 11001 \\ 00000 \\ + 11001 \\ \hline \end{array}$$

$$1111101$$

$$\begin{array}{r} 11001 \\ - 101 \\ \hline \end{array}$$

$$10100$$

$$101$$

$$101$$

$$101$$

$$\begin{array}{r} 11001 \\ -101 \\ \hline \end{array}$$

$$101$$

$$101$$

$$-101$$

$$0$$



11.2 常用計數制及其轉換

● 1.2.1 進位計數制

● 三、二進制

- 二進制的優點：運算簡單、物理實現容易、存儲和傳送方便、可靠
- 因為二進制中只有0和1兩個數字符號，可以用電子器件的兩種不同狀態來表示一位二進制數。例如，可以用晶體管的截止和導通表示1和0，或者用電平的高和低表示1和0等。所以，在數字系統中普遍採用二進制
- 二進制的缺點：數的位數太長且字符單調，使得書寫、記憶和閱讀不方便
- 因此，人們在進行指令書寫、程序輸入和輸出等工作時，通常採用八進制數和十六進制數作為二進制數的縮寫



11.2 常用計數制及其轉換

● 1.2.1 進位計數制

● 四、八進制

- 基數 $R=8$ 。八進制數中有0、1、...、7共8個基本數字符號，進位規律是“逢八進一”。八進制數的位權是8的整數次幂

任意一個八進制數N可以表示成

$$\begin{aligned}(N)_8 &= (K_{n-1}K_{n-2}\dots K_1K_0 . K_{-1}K_{-2}\dots K_{-m})_8 \\&= K_{n-1} \times 8^{n-1} + K_{n-2} \times 8^{n-2} + \dots + K_1 \times 8^1 + K_0 \times 8^0 \\&\quad + K_{-1} \times 8^{-1} + K_{-2} \times 8^{-2} + \dots + K_{-m} \times 8^{-m} \\&= \sum_{i=-m}^{n-1} K_i 8^i\end{aligned}$$

其中：n—整數位數；m—小數位數；

K_i —0~7中的任何一個字符， $-m \leq i \leq n-1$ 。

11.2 常用計數制及其轉換

● 1.2.1 進位計數制

● 五、十六進制

- 基數 $R=16$ 。十六進制數中有 0、1、...、9、A、B、C、D、E、F 共 16 個數字符號，其中，A ~ F 分別表示十進制數的 10 ~ 15。進位規律為“逢十六進一”。十六進制數的位權是 16 的整數次幂

任意一個十六進制數 N 可以表示成

$$\begin{aligned}(N)_{16} &= (K_{n-1}K_{n-2}\dots K_1K_0.K_{-1}K_{-2}\dots K_{-m})_{16} \\ &= K_{n-1} \times 16^{n-1} + K_{n-2} \times 16^{n-2} + \dots + K_1 \times 16^1 + K_0 \times 16^0 \\ &\quad + K_{-1} \times 16^{-1} + K_{-2} \times 16^{-2} + \dots + K_{-m} \times 16^{-m} \\ &= \sum_{i=-m}^{n-1} K_i 16^i\end{aligned}$$

其中： n —整數位數； m —小數位數； K_i —表示 0 ~ 9、A ~ F 中的任何一個字符， $-m \leq i \leq n-1$ 。

11.2 常用計數制及其轉換

● 1.2.1 進位計數制

- 十進制數 0 ~ 15 及其對應的二進制數、八進制數、十六進制數如下表所示

十進制數與二、八、十六進制數對照表

十進制	二進制	八進制	十六進制	十進制	二進制	八進制	十六進制
0	0000	00	0	8	1000	10	8
1	0001	01	1	9	1001	11	9
2	0010	02	2	10	1010	12	A
3	0011	03	3	11	1011	13	B
4	0100	04	4	12	1100	14	C
5	0101	05	5	13	1101	15	D
6	0110	06	6	14	1110	16	E
7	0111	07	7	15	1111	17	F

11.2 常用計數制及其轉換

● 1.2.2 數制轉換

- 數制轉換是指將一個數從一種進位制轉換成另一種進位制。從實際應用出發，要求掌握二進制數與十進制數、八進制數和十六進制數之間的相互轉換
- 一、二進制數與十進制數之間的轉換
- 二進制數轉換為十進制數
- 將二進制數表示成按權展開式，并按十進制運算法則進行計算，所得結果即為該數對應的十進制數

$$\begin{aligned}(10110.101)_2 &= 1 \times 2^4 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^{-1} + 1 \times 2^{-3} \\ &= 16 + 4 + 2 + 0.5 + 0.125 \\ &= (22.625)_{10}\end{aligned}$$



11.2 常用計數制及其轉換

● 1.2.2 數制轉換

- 一、二進制數與十進制數之間的轉換
- 十進制數轉換為二進制數
- 將二進制數表示成按權展開式，并按十進制運算法則進行計算，所得結果即為該數對應的十進制數
- 方法：基數乘法
- 十進制數轉換成二進制數時，應對整數和小數分別進行處理
 - 整數轉換——採用“除2取餘”的方法
 - 小數轉換——採用“乘2取整”的方法



11.2 常用計數制及其轉換

● 1.2.2 數制轉換

- 一、二進制數與十進制數之間的轉換
- 十進制數轉換為二進制數

例如， $(35)_{10} = (?)_2$

			余數	
2	3	5		
2	1	7 1	(K_0)
2		8 1	(K_1)
2		4 0	(K_2)
2		2 0	(K_3)
2		1 0	(K_4)
		0 1	(K_5)

低位
↑
高位

即 $(35)_{10} = (100011)_2$

11.2 常用計數制及其轉換

● 1.2.2 數制轉換

- 一、二進制數與十進制數之間的轉換
- 十進制數轉換為二進制數

例如， $(0.323)_{10} = (?)_2$ (保留4位小數)。

高位

↓

0.323
× 2
0.646
× 2
1.292
× 2
0.584
× 2
1.168
× 2
0.336

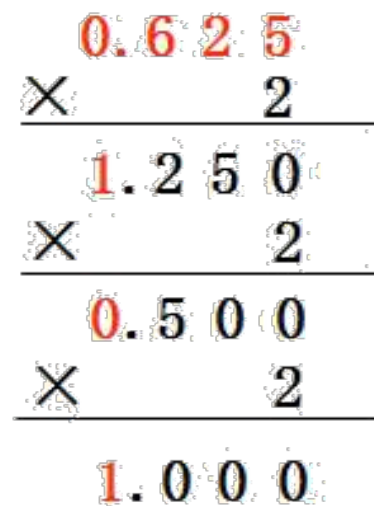
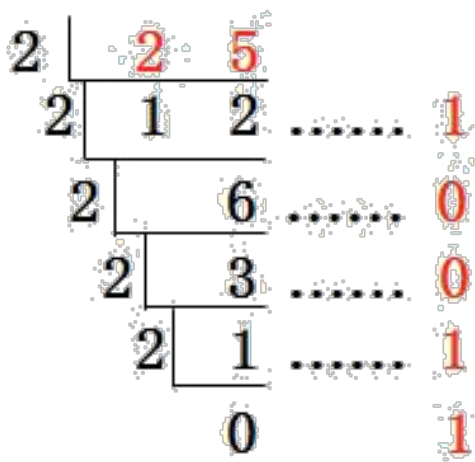
低位

即 $(0.323)_{10} = (0.0101)_2$

11.2 常用計數制及其轉換

● 1.2.2 數制轉換

- 一、二進制數與十進制數之間的轉換
- 十進制數轉換為二進制數
- 若一個十進制數既包含整數部分，又包含小數部分，則需將整數部分和小數部分分別轉換，然後用小數點將兩部分結果連到一起



即 $(25.625)_{10} = (11001.101)_2$

11.3 帶符號二進制數的代碼表示

- 爲了標記一個數的正負，通常在數的前面用 “+” 號表示正數，用 “-” 號表示負數。在數字系統中，符號和數值一樣是用0和1來表示
- 一般將數的最高位作爲符號位，用0表示正，用1表示負
- 其格式爲
- $X_f X_{n-1} X_{n-2} \dots X_1 X_0$
- 通常將用 “+” 、 “-” 表示正、負的二進制數稱爲符號數的真值，而把將符號和數值一起編碼表示的二進制數稱爲機器數或機器碼
- 常用的機器碼有原碼、反碼和補碼三種

11.3 帶符號二進制數的代碼表示

● 11.3.1 原碼

- 原碼：符號位用0表示正，1表示負；數值位保持不變。原碼表示法又稱為符號—數值表示法
- 一、小數原碼

設二進制小數 $X_1 = +0.x_{-1}x_{-2}\cdots x_{-m}$, $X_2 = -0.x_{-1}x_{-2}\cdots x_{-m}$, 原碼為

$$[X_1]_{\text{原碼}} = 0.x_{-1}x_{-2}\cdots x_{-m}, \quad [X_2]_{\text{原碼}} = 1.x_{-1}x_{-2}\cdots x_{-m}$$

例如，若 $X_1 = +0.0011$, $X_2 = -0.0011$
則 $[X_1]_{\text{原碼}} = 0.0011$, $[X_2]_{\text{原碼}} = 1.0011$

注意：小數“0”的原碼有正負之分， $+0.0\cdots 0$ 和 $-0.0\cdots 0$ 分別表示成
 $0.0\cdots 0$ 和 $1.0\cdots 0$ 。

11.3 帶符號二進制數的代碼表示

● 11.3.1 原碼

- 原碼：符號位用0表示正，1表示負；數值位保持不變。原碼表示法又稱為符號—數值表示法
- 二、整數原碼

設二進制整數 $X_1 = +x_{n-1}x_{n-2} \cdots x_0$ ， $X_2 = -x_{n-1}x_{n-2} \cdots x_0$ ，原碼為

$$[X_1]_{\text{原碼}} = 0x_{n-1}x_{n-2} \cdots x_0, \quad [X_2]_{\text{原碼}} = 1x_{n-1}x_{n-2} \cdots x_0$$

例如，若 $X_1 = +1001$ ， $X_2 = -1001$

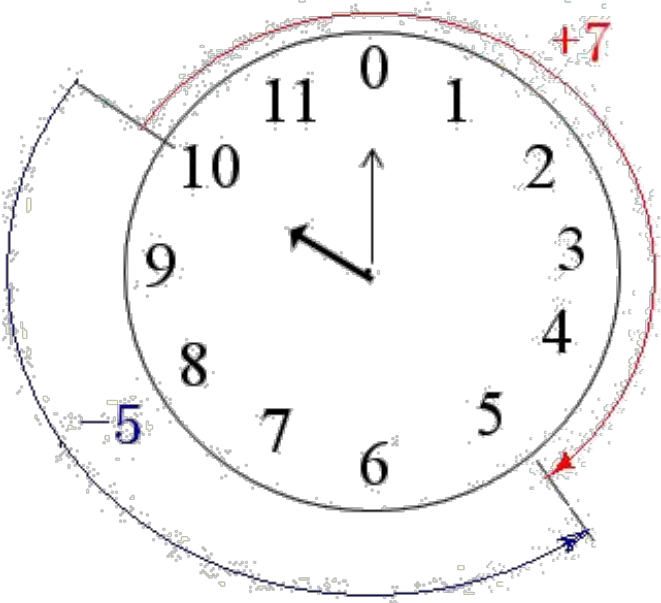
則 $[X_1]_{\text{原碼}} = 01001$ ， $[X_2]_{\text{原碼}} = 11001$

注意：整數“0”的原碼同樣有正負之分， $+00 \cdots 0$ 和 $-00 \cdots 0$ 的原碼分別表示為 $00 \cdots 0$ 和 $10 \cdots 0$ 。

11.3 帶符號二進制數的代碼表示

● 11.3.1 原碼

- 原碼的優點：簡單易懂,求取方便
- 缺點：加、減運算不方便
- 當進行兩數加、減運算時，要根據運算及參加運算的兩個數的符號來確定是加還是減；如果是做減法，還需根據兩數的大小確定被減數和減數，以及運算結果的符號。顯然，這將增加運算的複雜性
- 如何克服原碼的缺點呢？請看一種處理問題的方法



當要將時針從10點調至5點時，可順調7格（+7），也可反調5格（-5），即對12進制而言 $10-5 \equiv 10+7$ 。這裏， $5+7=12$ ，通常稱5和7對12進制而言互補

爲了克服原碼的缺點，引入了反碼和補碼

11.3 帶符號二進制數的代碼表示

● 11.3.2 反碼

- 帶符號二進制數的反碼表示：
- 符號位——用0表示正，用1表示負
- 數值位——正數反碼的數值位和真值的數值位相同；而負數反碼的數值位是真值的數值位按位變反

一、小數反碼

設二進制小數 $X_1 = +0.x_1x_2\cdots x_m$ ， $X_2 = -0.x_1x_2\cdots x_m$ ，反碼為

$$[X_1]_{\text{反碼}} = 0.x_1x_2\cdots x_m, \quad [X_2]_{\text{反碼}} = 1.\bar{x}_1\bar{x}_2\cdots\bar{x}_m$$

例如，若 $X_1 = +0.0011$ ， $X_2 = -0.0011$

則 $[X_1]_{\text{反碼}} = 0.0011$ ， $[X_2]_{\text{反碼}} = 1.1100$

小數“0”的反碼有正、負之分， $+0.0\cdots 0$ 和 $-0.0\cdots 0$ 分別用 $0.0\cdots 0$ 和 $1.1\cdots 1$ 表示

11.3 帶符號二進制數的代碼表示

● 11.3.2 反碼

- 帶符號二進制數的反碼表示：
- 符號位——用0表示正，用1表示負
- 數值位——正數反碼的數值位和真值的數值位相同；而負數反碼的數值位是真值的數值位按位變反

二、整數反碼

設二進制整數 $X_1 = +x_{-1}x_{-2}\cdots x_{-m}$ ， $X_2 = -x_{-1}x_{-2}\cdots x_{-m}$ ，反碼為

$$[X_1]_{\text{反碼}} = 0x_{-1}x_{-2}\cdots x_{-m} \quad , \quad [X_2]_{\text{反碼}} = 1\bar{x}_{-1}\bar{x}_{-2}\cdots\bar{x}_{-m}$$

例如，若 $X_1 = +0001$ ， $X_2 = -0001$

則 $[X_1]_{\text{反碼}} = 00001$ ， $[X_2]_{\text{反碼}} = 11110$

整數“0”的反碼也有兩種形式，即00...0和11...1

11.3 帶符號二進制數的代碼表示

● 11.3.2 反碼

- 帶符號二進制數的反碼表示：
- 符號位——用0表示正，用1表示負
- 數值位——正數反碼的數值位和真值的數值位相同；而負數反碼的數值位是真值的數值位按位變反
- 采用反碼進行加、減運算時，無論進行兩數相加還是兩數相減，均可通過加法實現
- 加、減運算規則如下：

$$[X_1 + X_2]_{\text{反}} = [X_1]_{\text{反}} + [X_2]_{\text{反}}$$

$$[X_1 - X_2]_{\text{反}} = [X_1]_{\text{反}} + [-X_2]_{\text{反}}$$

- 運算時，符號位和數值位一樣參加運算。當符號位有進位產生時，應將進位加到運算結果的最低位，才能得到最後結果

11.3 帶符號二進制數的代碼表示

● 11.3.2 反碼

例如，已知 $X_1 = +0.1110$ ， $X_2 = +0.0101$ ，求 $X_1 - X_2 = ?$

解：求 $X_1 - X_2$ 可通過反碼相加實現。運算如下：

$$[X_1 - X_2]_{\text{反}} = [X_1]_{\text{反}} + [-X_2]_{\text{反}} = 0.1110 + 1.1010$$

$$\begin{array}{r} 1 1 0 \\ + 1 0 0 0 \\ \hline 1 0 0 0 0 \\ + 1 \\ \hline 0 0 0 \end{array}$$

即 $[X_1 - X_2]_{\text{反}} = 0.1001$ 。由于結果的符號位為0，表示是正數，故 $X_1 - X_2 = +0.1001$

11.3 帶符號二進制數的代碼表示

● 11.3.3 補碼

- 帶符號二進制數的補碼表示：
- 符號位——用0表示正，用1表示負；
- 數值位——正數補碼的數值位與真值相同；負數補碼的數值位是真值的數值位按位變反，并在最低位加1

一、小數補碼

設二進制小數 $X_1 = +0.x_{-1}x_{-2}\cdots x_{-m}$ ， $X_2 = -0.x_{-1}x_{-2}\cdots x_{-m}$ ，則其補碼為

$$[X_1]_{\text{補碼}} = 0.x_{-1}x_{-2}\cdots x_{-m} \quad , \quad [X_2]_{\text{補碼}} = 1.\bar{x}_{-1}\bar{x}_{-2}\cdots\bar{x}_{-m} + 2^{-m}$$

例如，若 $X_1 = +0.1110$ ， $X_2 = -0.1110$ ，則 X_1 和 X_2 的補碼為

$$[X_1]_{\text{補碼}} = 0.1110$$

$$[X_2]_{\text{補碼}} = 1.0001 + 0.0001 = 1.0010$$

注意：小數“0”的補碼只有一種表示形式，即 $0.0\cdots 0$ 。

11.3 帶符號二進制數的代碼表示

● 11.3.3 補碼

- 帶符號二進制數的補碼表示：
- 符號位——用0表示正，用1表示負；
- 數值位——正數補碼的數值位與真值相同；負數補碼的數值位是真值的數值位按位變反，并在最低位加1

二、整數補碼

設二進制整數 $X_1 = +x_{n-1}x_{n-2}\cdots x_0$ ， $X_2 = -x_{n-1}x_{n-2}\cdots x_0$ ，其補碼為

$$[X_1]_{\text{補碼}} = 0x_{n-1}x_{n-2}\cdots x_0, \quad [X_2]_{\text{補碼}} = 1\bar{x}_{n-1}\bar{x}_{n-2}\cdots\bar{x}_0 + 2^0$$

例如，若 $X_1 = +1010$ ， $X_2 = -1010$ ，則 X_1 和 X_2 的補碼為

$[X_1]_{\text{補碼}} = 01010$ （正數補碼的數值位與真值的數值位相同）

$[X_2]_{\text{補碼}} = 10101 + 1 = 10110$ （負數補碼的數值位是真值的數值位按位變反，并在最低位加1）

整數“0”的補碼也只有一種表示形式，即00...0

11.3 帶符號二進制數的代碼表示

● 11.3.3 補碼

- 帶符號二進制數的補碼表示：
- 符號位——用0表示正，用1表示負；
- 數值位——正數補碼的數值位與真值相同；負數補碼的數值位是真值的數值位按位變反，并在最低位加1
- 采用補碼進行加、減運算時，可以將加、減運算均通過加法實現
運算規則如下：

$$[X_1 + X_2]_{\text{補}} = [X_1]_{\text{補}} + [X_2]_{\text{補}}$$

$$[X_1 - X_2]_{\text{補}} = [X_1]_{\text{補}} + [-X_2]_{\text{補}}$$

運算時，符號位和數值位一樣參加運算，若符號位有進位產生，則應將進位丟掉後才能得到正確結果。

11.3 帶符號二進制數的代碼表示

● 11.3.3 補碼

例： 已知 $X_1 = -1001$ ， $X_2 = +0011$ ，求 $X_1 - X_2 = ?$

解： 采用補碼求 $X_1 - X_2$ 的運算如下：

$$[X_1 - X_2]_{\text{補}} = [X_1]_{\text{補}} + [-X_2]_{\text{補}} = 10111 + 11101$$

$$\begin{array}{r} 1\ 0\ 1\ 1\ 1 \\ +\ 1\ 1\ 1\ 0\ 1 \\ \hline \text{丟掉 } \underline{1}\ 1\ 0\ 1\ 0\ 0 \end{array}$$

即 $[X_1 - X_2]_{\text{補}} = 10100$ 。由于結果的符號位為1，表示是負數，故 $X_1 - X_2 = -1100$

注意：補碼還原成真值時，對數值位變反加1

顯然，采用補碼進行加、減運算最方便

思考題

- 電子設計自動化（EDA）有什麼作用，請簡要說明。
- 什麼是組合邏輯電路？請簡要說明。
- 什麼是原碼，反碼，補碼？請簡要說明。

休息一下

Take a break

11.4 邏輯代數的基本概念

- 邏輯代數是數字系統設計的理論基礎和重要數學工具！
- 邏輯代數是從哲學領域中的邏輯學發展而來的
- 1847年,英國數學家喬治·布爾提出了用數學分析方法表示命題陳述的邏輯結構，幷成功地將形式邏輯歸結為一種代數演算，從而誕生了著名的“布爾代數”。
- 1938年，克勞德·向農將布爾代數應用于電話繼電器的開關電路，提出了“開關代數”。
- 隨著電子技術的發展，集成電路邏輯門已經取代了機械觸點開關，故“開關代數”這個術語已很少使用。爲了與“數字系統邏輯設計”這一術語相適應，人們更習慣于把開關代數叫做邏輯代數

11.4 邏輯代數的基本概念

● 11.4.1 基本概念

- 邏輯代數 L 是一個封閉的代數系統，它由一個邏輯變量集 K ，常量 0 和 1 以及“或”、“與”、“非”三種基本運算所構成，記為 $L = \{K, +, \cdot, -, 0, 1\}$ 。該系統應滿足下列公理

● 公理 1 交換律

- 對於任意邏輯變量 A 、 B ，有
- $A + B = B + A$; $A \cdot B = B \cdot A$

● 公理 2 結合律

- 對於任意的邏輯變量 A 、 B 、 C ，有
- $(A + B) + C = A + (B + C)$
- $(A \cdot B) \cdot C = A \cdot (B \cdot C)$

11.4 邏輯代數的基本概念

● 11.4.1 基本概念

● 公理 3 分配律

- 對於任意的邏輯變量 A 、 B 、 C ，有
- $A + (B \cdot C) = (A + B) \cdot (A + C)$ ；
- $A \cdot (B + C) = A \cdot B + A \cdot C$

● 公理 4 0-1 律

- 對於任意邏輯變量 A ，有 \square
- $A + 0 = A$ ； $A \cdot 1 = A$
- $A + 1 = 1$ ； $A \cdot 0 = 0$

● 公理 5 互補律

- 對於任意邏輯變量 A ，存在唯一的 \overline{A} ，使得
- $\overline{\overline{A}} + A = 1$ $\overline{\overline{A}} \cdot A = 0$

11.4 邏輯代數的基本概念

● 11.4.2 邏輯變量及基本邏輯運算

- 一．變量
- 邏輯代數和普通代數一樣，是用字母表示其值可以變化的量，即變量。所不同的是：
- 1．任何邏輯變量的取值只有兩種可能性——取值0或取值1
- 在普通代數中，變量的取值可以是任意實數，而邏輯代數是一種二值代數系統
- 2．邏輯值0和1是用來表徵矛盾的雙方和判斷事件真偽的形式符號，無大小、正負之分
- 在數字系統中，開關的接通與斷開，電壓的高和低，信號的有和無，晶體管的導通與截止等兩種穩定的物理狀態，均可用1和0這兩種不同的邏輯值來表徵

11.4 邏輯代數的基本概念

● 11.4.2 邏輯變量及基本邏輯運算

● 二．基本邏輯運算

- 描述一個數字系統，必須反映一個複雜系統中各開關元件之間的聯繫，這種相互聯繫反映到數學上就是幾種運算關係
- 邏輯代數中定義了“或”、“與”、“非”三種基本運算
- 1．“或”運算
- 如果決定某一事件是否發生的多個條件中，只要有一個或一個以上條件成立，事件便可發生，則這種因果關係稱之為“或”邏輯
- 例如，用兩個開關并聯控制一個燈的照明控制電路

11.4 邏輯代數的基本概念

● 11.4.2 邏輯變量及基本邏輯運算

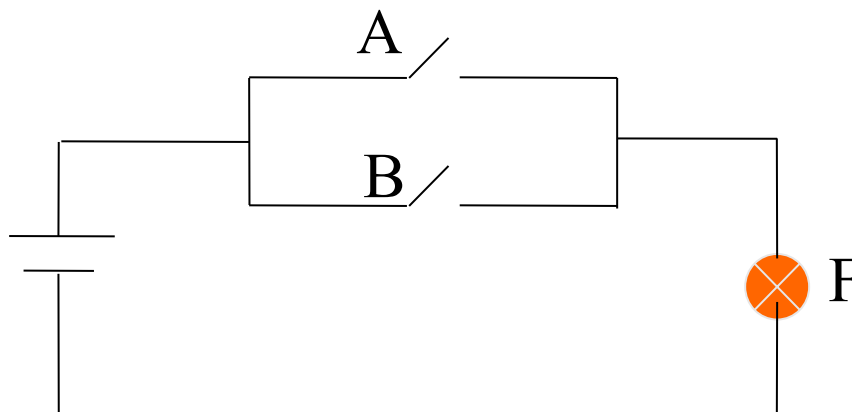
● 二．基本邏輯運算

● 1．“或”運算

- 如果決定某一事件是否發生的多個條件中，只要有一個或一個以上條件成立，事件便可發生，則這種因果關係稱之為“或”邏輯

- 例如，用兩個開關并聯控制一個燈的照明控制電路

- 開關A和B并聯控制燈F。可以看出，當開關A、B中有一個閉合或者兩個均閉合時，燈F即亮。因此，燈F與開關A、B之間的關係是“或”邏輯關係



并聯開關電路

11.4 邏輯代數的基本概念

● 11.4.2 邏輯變量及基本邏輯運算

● 二．基本邏輯運算

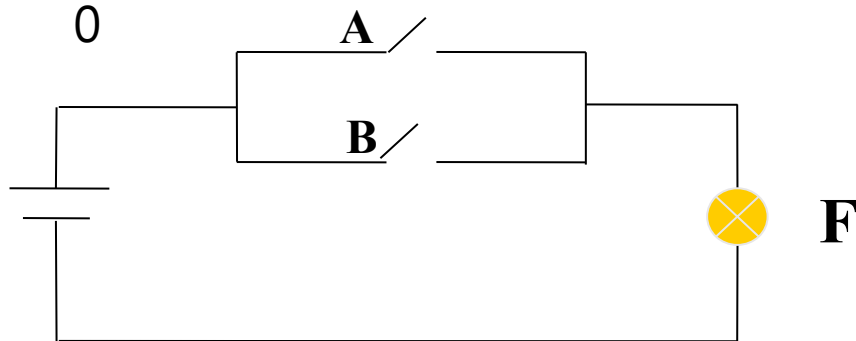
● 1．“或”運算

● 邏輯代數中，“或”邏輯用“或”運算描述。其運算符號為“+”，有時也用“ \vee ”表示。兩變量“或”運算的關係可表示為

● $F = A + B$ 或者 $F = A \vee B$ 讀作“F等于A或B”

● 假定開關斷開用0表示，開關閉合用1表示；燈滅用0表示，燈亮用1表示，則燈F與開關A、B的關係如下表所示

● 即：A、B中只要有一個為1，則F為1；僅當A、B均為0時，F才為0



并聯開關電路

“或”運算表

A	B	F
0	0	0
0	1	1
1	0	1
1	1	1



11.4 邏輯代數的基本概念

- 11.4.2 邏輯變量及基本邏輯運算

- 二．基本邏輯運算

- 1．“或”運算

- “或”運算的運算法則：

- $0 + 0 = 0$ $1 + 0 = 1$

- $0 + 1 = 1$ $1 + 1 = 1$

- 實現“或”運算關係的邏輯電路稱為“或”門



11.4 邏輯代數的基本概念

● 11.4.2 邏輯變量及基本邏輯運算

● 二．基本邏輯運算

● 2．“與”運算

- 如果決定某一事件發生的多個條件必須同時具備，事件才能發生，則這種因果關係稱之為“與”邏輯。
- 在邏輯代數中，“與”邏輯關係用“與”運算描述。其運算符號為“ \cdot ”，有時也用“ \wedge ”表示。兩變量“與”運算關係可表示為
- $F = A \cdot B$ 或者 $F = A \wedge B$
- 即：若A、B均為1，則F為1；否則，F為0

“與” 運算表

A	B	F
0	0	0
0	1	0
1	0	0
1	1	1

11.4 邏輯代數的基本概念

● 11.4.2 邏輯變量及基本邏輯運算

● 二．基本邏輯運算

● 2．“與” 運算

● 假定在開關串聯電路中，開關閉合狀態用1表示，斷開狀態用0表示，燈亮用1表示，燈滅用0表示，則電路中燈F和開關A、B之間的關係即上表所示的“與”運算關係

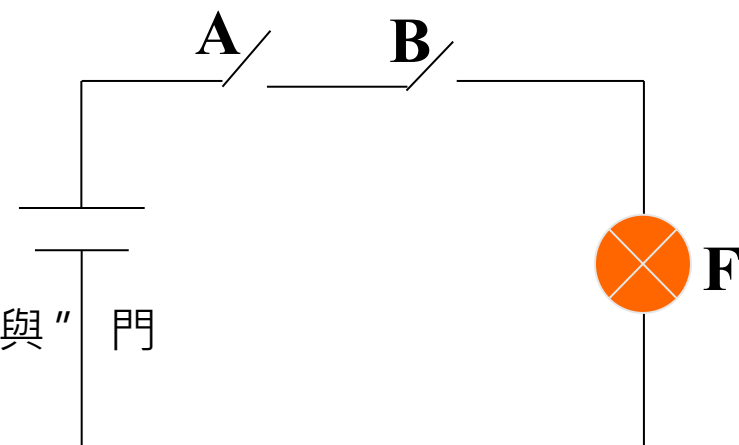
● “與” 運算的運算法則:□

● $0 \cdot 0 = 0$ $1 \cdot 0 = 0$

● $0 \cdot 1 = 0$ $1 \cdot 1 = 1$

● 實現“與”運算關係的邏輯電路稱為“與”門

● “與”邏輯關係如右表所示



串聯開關電路

11.4 邏輯代數的基本概念

● 11.4.2 邏輯變量及基本邏輯運算

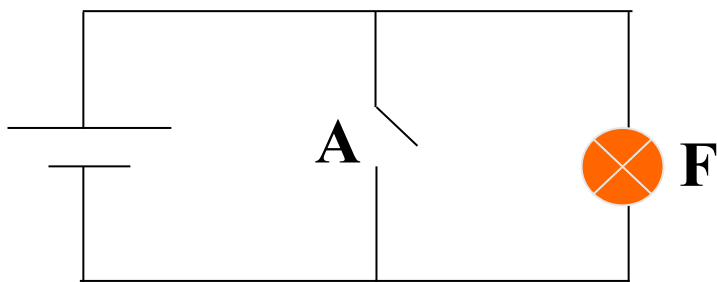
● 二．基本邏輯運算

● 3．“非” 運算

● 如果某一事件的發生取決於條件的否定，即事件與事件發生的條件之間構成矛盾，則這種因果關係稱為“非”邏輯

● 在邏輯代數中，“非”邏輯用“非”運算描述。其運算符號為“ \neg ”，有時也用“ \neg ”表示。“非”運算的邏輯關係可表示為或者 $F = \neg A$ ，讀作“F等於A非”

● 即：若A為0，則F為1；若A為1，則F為0



開關與燈并聯電路

“非”運算表

A	F
0	1
1	0



11.4 邏輯代數的基本概念

● 11.4.3 邏輯函數

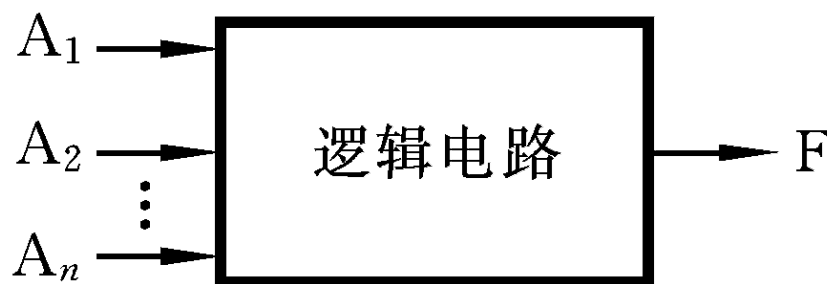
- 一、邏輯函數的定義
- 邏輯代數中函數的定義與普通代數中函數的定義類似，即隨自變量變化的因變量。但和普通代數中函數的概念相比，邏輯函數具有如下特點
- 1．邏輯函數和邏輯變量一樣，取值只有0和1兩種可能
- 2．函數和變量之間的關係是由“或”、“與”、“非”三種基本運算決定的



11.4 邏輯代數的基本概念

● 11.4.3 邏輯函數

- 一、邏輯函數的定義
- 從數字系統研究的角度看，邏輯函數的定義如下：
- 設某一邏輯電路的輸入邏輯變量為 A_1, A_2, \dots, A_n ，輸出邏輯變量為 F ，如下圖所示



- 圖中， F 被稱為 A_1, A_2, \dots, A_n 的邏輯函數，記為 $F = f(A_1, A_2, \dots, A_n)$
- 邏輯電路輸出函數的取值是由邏輯變量的取值和電路本身的結構決定的
- 任何一個邏輯電路的功能都可由相應的邏輯函數完全描述，因此，可借助抽象的代數表達式對電路加以分析研究

11.4 邏輯代數的基本概念

● 11.4.3 邏輯函數

- 二、邏輯函數的相等
- 邏輯函數和普通代數中的函數一樣存在是否相等的問題。
- 什麼叫做兩個邏輯函數相等呢？
- 設有兩個相同變量的邏輯函數
- $F1 = f1(A_1, A_2, \dots, A_n)$
- $F2 = f2(A_1, A_2, \dots, A_n)$
- 若對應于邏輯變量 A_1, A_2, \dots, A_n 的任何一組取值， $F1$ 和 $F2$ 的值都相同，則稱函數 $F1$ 和 $F2$ 相等，記作 $F1 = F2$

11.4 邏輯代數的基本概念

● 11.4.3 邏輯變量及基本邏輯運算

- 三、邏輯函數的表示法
- 如何對邏輯功能進行描述？
- 常用的方法有邏輯表達式、真值表、卡諾圖3種
- 1、邏輯表達式
- 邏輯表達式是由邏輯變量和“或”、“與”、“非”3種運算符以及括號所構成的式子。例如

$$F = f(A, B) = \overline{A}B + A\overline{B}$$

- 該邏輯表達式描述了一個兩變量的邏輯函數F。函數F和變量A、B的關係是：當變量A和B取值不同時，函數F的值為“1”；取值相同時，函數F的值為“0”

11.5 邏輯代數的基本定理和規則

● 11.5.1 基本定理

- 根據邏輯代數的公理，可以推導出邏輯代數的基本定理
- 常用的有 8 組定理

● 定理 1

- $0 + 0 = 0$ $1 + 0 = 1$ $0 \cdot 0 = 0$ $1 \cdot 0 = 0$

- $0 + 1 = 1$ $1 + 1 = 1$ $0 \cdot 1 = 0$ $1 \cdot 1 = 1$



11.5 邏輯代數的基本定理和規則

- 11.5.1 基本定理

- 根據邏輯代數的公理，可以推導出邏輯代數的基本定理
- 常用的有 8 組定理

- 定理 2

- $A + A = A$; $A \cdot A = A$

- 定理 3

- $A + A \cdot B = A$; $A \cdot (A + B) = A$



11.5 邏輯代數的基本定理和規則

● 11.5.1 基本定理

- 根據邏輯代數的公理，可以推導出邏輯代數的基本定理
- 常用的有 8 組定理

$$\square \square 4 \quad A + \bar{A} \cdot B = A + B \quad A \cdot (\bar{A} + B) = A \cdot B$$

$$\begin{aligned} \square \square \quad A + \bar{A}B &= (A + \bar{A}) \cdot (A + B) && \text{公理3} \\ &= 1(A + B) && \text{公理5} \\ &= A + B && \text{公理4} \end{aligned}$$

$$\text{定理 5} \quad \overline{\overline{A}} = A$$

$$\text{证明} \quad \text{令} \quad \overline{A} = X$$

$$\text{因而} \quad \bar{A} \cdot X = 0 \quad \bar{A} + X = 1$$

$$\text{但是} \quad \bar{A} \cdot A = 0 \quad \bar{A} + A = 1$$

$$\text{根据公理5的唯一性有:} \quad X = A$$

11.5 邏輯代數的基本定理和規則

● 11.5.1 基本定理

- 根據邏輯代數的公理，可以推導出邏輯代數的基本定理
- 常用的有 8 組定理

$$\text{定理6} \quad \overline{A+B} = \overline{A} \cdot \overline{B} \quad \overline{A \cdot B} = \overline{A} + \overline{B}$$

$$\text{定理7} \quad A \cdot B + A \cdot \overline{B} = A \quad (A+B) \cdot (A+\overline{B}) = A$$

$$\square \square 8 \quad A \cdot B + \overline{A} \cdot C + B \cdot C = A \cdot B + \overline{A} \cdot C$$

$$\square A+B \square \cdot (\overline{A}+C \square \cdot (B+C) = (A+B \square \cdot (\overline{A}+C \square$$

11.5 邏輯代數的基本定理和規則

● 11.5.1 基本定理

- 根據邏輯代數的公理，可以推導出邏輯代數的基本定理
- 常用的有 8 組定理

定理7 $A \cdot B + A \cdot \bar{B} = A$ $(A + B) \cdot (A + \bar{B}) = A$

证明	$A \cdot B + A \cdot \bar{B} = A \cdot (B + \bar{B})$	公理3
	$= A \cdot 1$	公理5
	$= A$	公理4

11.5 邏輯代數的基本定理和規則

● 11.5.2 重要規則

- 3條重要規則:代入規則、反演規則、對偶規則
- 代入規則:任何一個含有變量 A 的邏輯等式,如果將所有出現 A 的位置都代之以同一個邏輯函數 F ,則等式仍然成立
- 反演規則:若將邏輯函數表達式 F 中所有的“ \cdot ”變成“ $+$ ”,“ $+$ ”變成“ \cdot ”;“ 0 ”變成“ 1 ”,“ 1 ”變成“ 0 ”;原變量變成反變量,反變量變成原變量。幷保持原函數中的運算順序不變,則所得到的新的函數為原函數 F 的反函數
- 對偶規則:如果將邏輯函數表達式 F 中所有的“ \cdot ”變成“ $+$ ”,“ $+$ ”變成“ \cdot ”,“ 0 ”變成“ 1 ”,“ 1 ”變成“ 0 ”,幷保持原函數中的運算順序不變,則所得到的新的邏輯表達式稱為函數 F 的對偶式,幷記作 F'

11.6 複合邏輯

- 實際應用中廣泛採用“與非”門、“或非”門、“與或非”門、“異或”門等門電路。這些門電路輸出和輸入之間的邏輯關係可由3種基本運算構成的複合運算來描述，通常將這種邏輯關係稱為複合邏輯，相應的邏輯門則稱為複合門

- 一、與非邏輯

- 與非邏輯是由與、非兩種邏輯複合形成的，可用邏輯函數表示為

$$F = \overline{A \cdot B \cdot C \dots}$$

- 邏輯功能：只要變量A、B、C、...中有一個為0，則函數F為1；僅當變量A、B、C、...全部為1時，函數F為0。
- 實現與非邏輯的門電路稱為“與非”門

11.6 複合邏輯

● 一、與非邏輯

由定理 $\overline{A \cdot B} = \overline{A} + \overline{B}$ 不難看出，“與”之“非”可以產生“或”的關係。因此，實際上只要有了與非邏輯便可實現與、或、非3種基本邏輯。

以兩變量與非邏輯為例：

與： $F = \overline{\overline{A \cdot B \cdot 1}} = \overline{\overline{A \cdot B}} = A \cdot B$

或： $F = \overline{\overline{A \cdot 1 \cdot B \cdot 1}} = \overline{\overline{A \cdot B}} = A + B$

非： $F = \overline{A \cdot 1} = \overline{A}$

由于與非邏輯又可實現3種基本邏輯，所以，只要有了與非門便可組成實現各種邏輯功能的電路，通常稱與非門為通用門。采用與非邏輯可以減少邏輯電路中門的種類，提高標準化程度

11.6 複合邏輯

● 二、或非邏輯

或非邏輯是由或、非兩種邏輯複合形成的，可用邏輯

函數表示為 $F = \overline{A + B + C + \dots}$

邏輯功能：只要變量A、B、C...中有一個為1，則函數F為0；僅當變量A、B、C...全部為0時，函數F為1

實現或非邏輯的門電路稱為“或非”門

同樣，由定理 $\overline{A + B} = \overline{A} \cdot \overline{B}$ 可知，“或”之“非”可以產生“與”的關係。因此，只要有了或非邏輯也可以實現與、或、非3種基本邏輯。以兩變量或非邏輯為例：

與：
$$F = \overline{\overline{A + 0} + \overline{B + 0}} = \overline{\overline{A} + \overline{B}} = A \cdot B$$

或：
$$F = \overline{\overline{A + B} + 0} = \overline{\overline{A + B}} = A + B$$

非：
$$F = \overline{A + 0} = \overline{A}$$

或非門同樣是一種通用門

11.6 複合邏輯

● 三、與或非邏輯

1. 異或邏輯

異或邏輯是一種兩變量邏輯關係，可用邏輯函數表示為

$$F = A \oplus B = \overline{A}B + A\overline{B}$$

邏輯功能：變量A、B取值相同，F為0；變量A、B取值相異，F為1。實現異或運算的邏輯門稱為“異或門”

根據異或邏輯的定義可知：

$$A \oplus 0 = A \quad A \oplus 1 = \overline{A}$$

$$A \oplus A = 0 \quad A \oplus \overline{A} = 1$$

當多個變量進行異或運算時，可用兩兩運算的結果再運算，也可兩兩依次運算

11.6 複合邏輯

● 三、與或非邏輯

與或非邏輯是由3種基本邏輯複合形成的，邏輯函數表達式的形式為 $F = \overline{AB + CD + \dots}$

邏輯功能：僅當每一個“與項”均為0時，F才能為1，否則F為0。實現與或非功能的門電路稱為“與或非”門

顯然，可以僅用與或非門組成實現各種功能的邏輯電路。但實際應用中這樣做一般會很不經濟，所以，與或非門主要用來實現與或非形式的函數。必要時可將邏輯函數表達式的形式變換成與或非的形式

11.6 複合邏輯

● 四、异或邏輯

1. 异或邏輯

异或邏輯是一種兩變量邏輯關係，可用邏輯函數表示為

$$F = A \oplus B = \overline{A}B + A\overline{B}$$

邏輯功能：變量A、B取值相同，F為0；變量A、B取值相異，F為1。實現异或運算的邏輯門稱為“异或門”

根據异或邏輯的定義可知：

$$A \oplus 0 = A \quad A \oplus 1 = \overline{A}$$

$$A \oplus A = 0 \quad A \oplus \overline{A} = 1$$

當多個變量進行异或運算時，可用兩兩運算的結果再運算，也可兩兩依次運算

11.6 複合邏輯

- 四、异或邏輯

- 特性：在進行异或運算的多個變量中，若有奇數個變量的值為1，則運算結果為1；若有偶數個變量的值為1，則運算結果為0

例如， $F = A \oplus B \oplus C \oplus D$

$= (A \oplus B) \oplus (C \oplus D)$ (兩兩運算的結果再運算)

$= [(A \oplus B) \oplus C] \oplus D$ (兩兩依次運算)

思考題

- 基本邏輯運算有哪些？請列舉幾種，并進行說明。
- 什麼是與非邏輯？請簡要說明。
- 什麼是異或邏輯？請簡要說明。

休息一下

Take a break



感謝觀賞

Thank you for listening.