Chapter 1

Basic

1.1 N-base

1.2 N-base to Decimal

$$\text{Decimal} = \sum_{i=-m}^{n-1} k_i \cdot N^i$$

N 为相应进制的 base, k_i 为 i-th 的系数. 整数部分有 n 位, 小数部分有 m 位.

$$(1101.011)_B = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3}$$

1.3 Bin to Oct or Hex

Oct 取三位 Bin. (若不足三位之倍数则在最高位和最低位补齐).

$$(3D.BE)_H = \underset{3}{0011}\ \underset{D}{1101}\,.\,\underset{B}{1011}\ \underset{E}{1110} = (11\,1101.1011\,111)_B$$

Table 1.1: Oct to Bin

Bin	Oct
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Table 1.2: Hex to Bin

Hex	Bin
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

Hex 取四位 Bin. (若不足四位之倍数则在最高位和最低位补齐).

$$(1001.1101)_B = 001\,001.110\,100 = \underset{1}{001}\,\underset{1}{001}\,.\,\underset{6}{110}\,\underset{4}{100} = (11.64)_O$$

1.4 Dec to n-base

1.4.1 整数

短除法, 除到余数为零.

$2 \lfloor \underline{25}$

$2 \lfloor \underline{12}$ remainder 1

$2 \lfloor \underline{6}$ remainder 0

$2 \lfloor \underline{3}$ remainder 0

$2 \lfloor \underline{1}$ remainder 1

0 remainder 1

得到的数字, 下方为高位, 上方为低位. (最后一次运算为最高位) 即结果为 $(1\,1001)_B$

1.4.2 小数

加倍取整数为当前位, 然后取其小数作为下一位之运算.

$$0.125 \times 2 = 0.25_{0} \tag{1.1}$$

$$0.25 \times 2 = 0.5_{0} \tag{1.2}$$

$$0.5 \times 2 = 1.0_{1} \tag{1.3}$$

即小数部分为 0.001 (最后一次运算为最低位)

1.5 compliment

1.5.1 1's compliment (反码)

按位取反, 最高位保留一个符号位. (0 为正, 1 为负)

1.5.2 2's compliment (补码)

负数的情况下, 1's compliment 加 $(0001)_B$ 即可 (正数不变)

1.6 Encoding

1.6.1 BCD Code

8421 BCD

8421 就是按照十进制拆数, 每位数用对应的二进制码表示 +

Table 1.3: Dec to 8421

Dec	8421
10	0001 0000
11	0001 0001
12	0001 0010
20	0010 0010
21	0010 0001

0 至 9 与 Binary 相同.

Excess-3 (XS-3)

就是 8421 每一位加三.

- Find the decimal equivalent of the given binary number.
- Add +3 to each digit of decimal number.
- Convert the newly obtained decimal number back to binary number to get required excess-3 equivalent.

Table 1.4: Dec, 8421 and Excess-3

Dec	8421	Excess-3
0	000	011
1	001	100
2	010	101
3	011	110
4	100	111
5	101	1000
6	110	1001
7	111	1010
8	1000	1011
9	1001	1100

Grey Code

Table 1.5: Bin to Gray

Bin	Gray
000	000
001	001
010	011
011	010
100	110
101	111
110	101
111	100

$$G(n) = B(n+1)\,XOR\,B(n)$$

$$G(n) = B(n+1) + B(n)$$

自低位至高位运算即可，无需考虑进位.

$$B(n) = -B(n+1) + G(n)$$

自高位至低位运算即可，无需考虑借位.

Chapter 2

Boolean Algebra

2.1 Operator

2.1.1 AND

$$A \cdot B$$

A	B	$A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

Table 2.1: AND Truth Table

2.1.2 OR

$$A + B$$

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

Table 2.2: OR Truth Table

2.1.3 Exclusive OR/XOR

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Table 2.3: XOR Truth Table

From the truth table we can get that.

$$A \oplus B = A'B + AB'$$

2.1.4 Exclusive NOR/XNOR

A	B	$A \odot B$
0	0	1
0	1	0
1	0	0
1	1	1

Table 2.4: XNOR Truth Table

From the truth table we can get that.

$$A \odot B = A'B' + AB$$

2.1.5 NOT

$$A'$$

2.1.6 与或非

俩与门加一或非门 (two AND then a NOR) 四个输入变量

2.2 Formula

2.2.1 代数公理

A 和 0, A 和 1

$$A \cdot 0 = 0$$
$$A \cdot 1 = A$$
$$A + 0 = A$$
$$A + 1 = 1$$

A 和 A, A 和 NOT A

$$A \cdot A = A$$
$$A + A = A$$
$$A \cdot A' = 0$$
$$A + A' = 1$$

2.2.2 等同律

$$\begin{aligned} A + A' \cdot B &= A \cdot 1 + A' \cdot B \\ &= A \cdot (1 + B) + A' \cdot B \\ &= A + AB + A'B \\ &= A + (A' + A) \cdot B \\ &= A + B \end{aligned}$$

2.2.3 吸收律

$$\begin{aligned} A(A + B) &= A + AB \\ &= A \cdot (1 + B) && \text{ONE OR anything is always ONE} \\ &= A \end{aligned}$$

2.2.4 De Morgan's law

$$(AB)' = A' + B'$$
$$(A + B)' = A' \cdot B'$$

2.2.5 分配律

$$A + B \cdot C = (A + B) \cdot (A + C) \qquad \text{AND distributive law} \tag{2.1}$$

$$A \cdot (B + C) = AB + AC \qquad \text{OR distributive law} \tag{2.2}$$

2.2.6 A something OR NOT A something else

(A AND $Other_1$) OR (NOT A AND $Other_2$) OR ($Other_1$ AND $Other_2$ AND $Other_3$...)

$$AB + A'C + BC = AB + A'C \qquad \text{舍去最后一项}$$

$$AB + A'C + BCD = AB + A'C$$

$$AB + A'C + BCDE = AB + A'C \qquad \text{不管有多少项相乘}$$

两个乘积项分别包含 A 和 A' 两个因子, 其余因子正好组成第三个乘积项或其一部分, 这一项则可以被消去.

2.2.7 对偶性 (Duality)

如果两个逻辑式子相等, 则他们的对偶式亦相等
求 Duality, 就是把 OR 变 AND, 把 AND 变 OR. (注意和求反区别)

$$A + B \cdot C = (A + B) \cdot (A + C) \qquad \text{AND distributive law}$$

$$(A + B \cdot C)^D = [(A + B) \cdot (A + C)]^D$$

$$A \cdot (B + C) = AB + AC \qquad \text{OR distributive law}$$

2.3 Truth Table to Boolean Expression

右边取 1 的组合写下来做 SOP 即可.

2.4 逻辑表达式化为标准形式

2.4.1 最小项

即 SOP
写 K-map, (四变量时左边为前俩, 上边为后俩), (三变量时左边为第一, 上边为后俩). (标最小项时还是一行一行过去的) 然后按照二进制顺序排列即可得到最小项.
~~说白了最小项就是按照二进制顺序排列.~~ 更准确地说, 最小项就是把二进制转换成对应的十进制.

例子 $AB'CD'EF$ 按照原变量为 1, 反变量为 0 可以得到 $(101011)_B$ 转换为十进制数为 $(43)_D$ 则其编号就是 43. $AB'CD'EF$ 有 6 个变量, 即一共有 $2^6 = 64$ 项

2.4.2 最大项

即 POS.
最小项于最大项为互补关系. 即 $m_i' = M_i$, $M_i' = m_i$
列个真值表或者画个卡诺图, 已知最小项 (为 1 的), 剩下的 (为 0 的) 就是最大项了

$$Y = \sum_i m_i = \prod_{k \neq i} M_k$$

2.5 K-map

写 K-map, (四变量时左边为前俩, 上边为后俩), (三变量时左边为第一, 上边为后俩).
别忘了四角是可以互通的 (或者上边俩角, 或者下边俩角, 但是对角线是不能互通的).

2.6 SOP and POS

2.6.1 SOP 和 NAND

2.6.2 POS 和 NOR

Chapter 3

CMOS 门电路

FET P channel[1] is pointing out.
P 沟道指出去

3.1 CMOS NOT

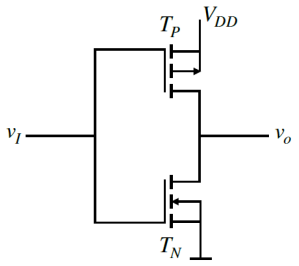

Figure 3.1: CMOS NOT

T channel Serializing N channel is NOT Gate.
T 串 N

[1] NPN is not pointing in

3.2 CMOS NAND

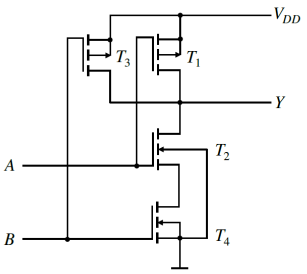

Figure 3.2: CMOS NAND

T channels Serializing, P channels Paralleling together are NAND gate.
T 相串 P 相并为 NAND

3.3 CMOS NOR

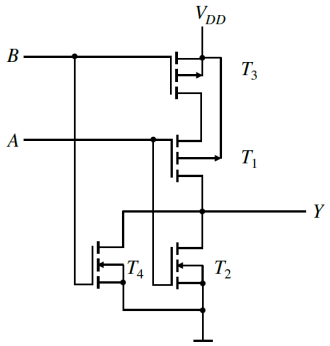

Figure 3.3: CMOS NOR

T channels Paralleling, P channels Serializing together are NOR gate.
P 相串 T 相并为 NOR

3.4 OD Gate

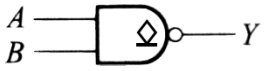

Figure 3.4: OD Gate

漏极开路门, 必须有个上拉电阻.

3.5 Transmission Gate

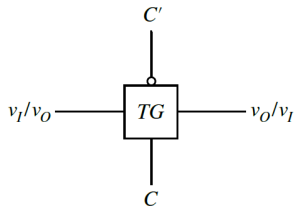

Figure 3.5: Transmission

传输门, 就一电子开关.

上下分别为 NOT C 和 C, 若下边的 C 为 1(上边 NOT C 为 0) 则左右导通, 反之截止.

3.6 三态门

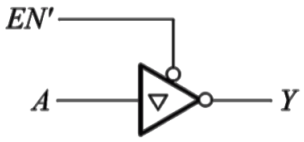

Figure 3.6: 三态门

其有三种状态: 高电平, 低电平, 高阻态

就一非门, 只不过多了一个高阻态.

当 $EN' = 0$ 时为正常非门 (低电平有效, 有效就是作为非门正常工作)

$EN' = 1$ 为高阻态.

3.7 噪声容限

$$V_{NH} = V_{(\text{OH min})} - V_{(\text{IH min})}$$
$$V_{NL} = V_{(\text{IL max})} - V_{(\text{OL max})}$$

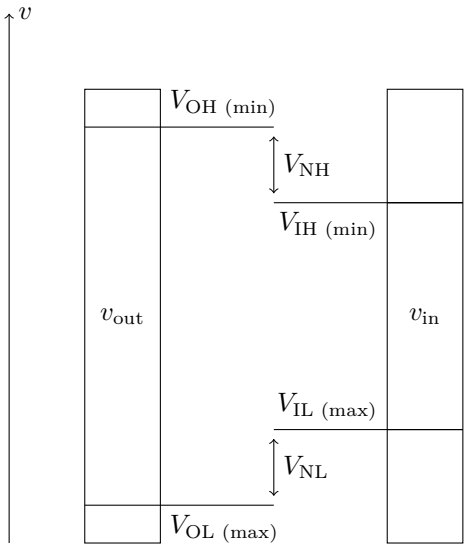

Figure 3.7: 噪声容限

3.8 CMOS 门注意事项

- 不允许出现悬空端
- NAND 接 1 ($1 \cdot A = A$)
- NOR 接 0 ($0 + A = A$)
- TTL 可悬空, 若悬空则为高电平

Chapter 4

组合逻辑电路

- 逻辑抽象
- 真值表
- 写出表达式
- 化简表达式
- 画出门电路图

4.1 常用模型

4.1.1 Full Adder

被加数, 加数, Carry in, Sum, Carry out.

半加器不包含 Carry in.

Table 4.1: Truth table of Full Adder

A	B	C_{in}	Sum	C_{out}
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

4.1.2 Full Subtractor

被减数, 减数, Borrow in, Difference, Borrow out.

半减器不包含 Borrow in.

Table 4.2: Truth table of Full Subtractor

Minuend	Subtrahend	B_{in}	Diff	B_{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

4.2 竞争冒险 (race hazard)

4.2.1 Race

两个输入端信号同时向相反方向变化, 变化的时间有差异时为 Race.

4.2.2 Hazard

两个输入端变化方向相反时, 由 Race 产生的可能出现的. 由于**电路中出现延时而引起的**.

当一个门电路的输入公式在某些情况下可以化为两个互补信号的 AND ($A \cdot A'$) 或 OR ($A + A'$) 时就有可能出现 race hazard.

消除方法

- 接入滤波电容
- 引入选通脉冲
- 增加冗余项, 使得最终结果不可能出现两个互补信号.

增加冗余项

$$Y = AB + A'C$$

当 $B=C=1$ 时, $Y=A+A'$. 若增加一个冗余项, 变为

$$Y=AB+A'C+BC$$

当 $B=C=1$ 时, $Y=A+A'+1=1$. 避免了最终结果的两个互补变量的出现.

Chapter 5

集成逻辑电路

5.1 Encoder

好像不考
所谓编码器 (74LS148)
注意输入是高电平有效还是低电平有效, 注意输出是输出原变量还是反变量.
2^n 个输入端口, n 个输出端口. 将输入的输入端口序号用二进制进行编码. (同时只能有一个有效输入)

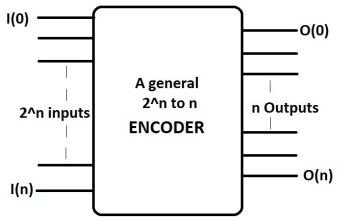

Figure 5.1: encoder

Table 5.1: Truth Table of 8 to 3 Encoder

INPUT								OUTPUT		
I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0	O_2	O_1	O_2
0	0	0	0	0	0	0	0	X	X	X
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

5.1.1 优先译码器

正常来说只能同时有一个输入, 输入为对应的输入端口序号的二进制编码.
优先编码器可以同时运行多个输入, 按照 $I_7 > I_6 > I_5 > I_4 > I_3 > I_2 > I_1 > I_0$ 的优先级来决定输入, 即输入端口序号大者优先被输出.

5.2 Decoder

所谓译码器 (74LS138)
n 个输入, 2^n 个输出. 注意看输出为低电平还是高电平.

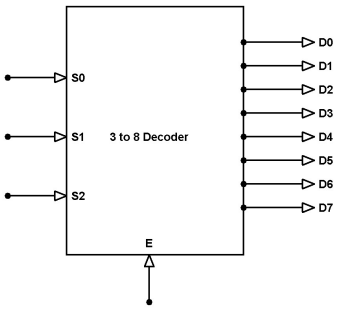

Figure 5.2: 3 to 8 decoder

Table 5.2: Truth Table of 2 to 4 decoder

Enable	Inputs		Outputs			
E	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	x	x	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

5.2.1 利用 Decoder 和 NAND 门来表示 SOP

一个 SOP 表达式可以用 Decoder[1]和 NAND 门来表示.

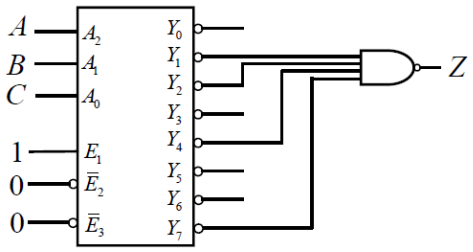

Figure 5.3: Decoder 与 NAND 表示 SOP

上图表示 $Z = A'B'C + A'BC' + AB'C' + ABC$, 化为最小项为 $Z = m_1 + m_2 + m_4 + m_7$.

5.3 Multiplexer

可以理解为是一个 decoder 用作电子开关. 有 2^n 个输入和 1 个输出, n 个控制端.

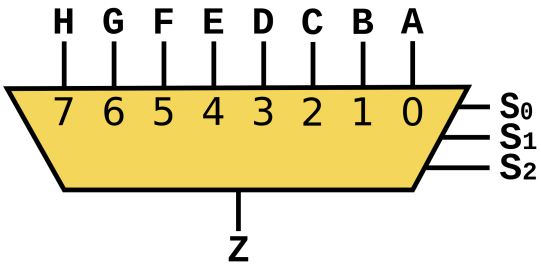

Figure 5.4: 3:8 MUX

[1]这里的 Decoder 输出为反变量, 若输出原变量则是用 OR 门

Table 5.3: Truth table of 2:4 MUX

Selection Lines		Output
S_1	S_0	Z
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

$$Z = S_1' \cdot S_0' \cdot I_0 + S_1' \cdot S_0 \cdot I_1 + S_1 \cdot S_0' \cdot I_2 + S_1 \cdot S_0 \cdot I_3$$

5.3.1 MUX 与三变量表达式

三变量的话列出真值表就好了, 然后将真值表的输出填入到输入端口.

例子 用 MUX 实现 $F = A \oplus B \oplus C$.
先列出真值表

Table 5.4: $F = A \oplus B \oplus C$ 的真值表

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

换句话说, 用 SOP 表示, 将 SOP 有的项转换成**最小项所对应的端口**的值为 1.

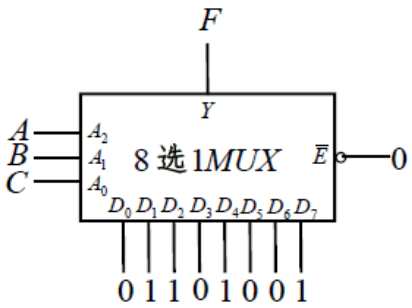

Figure 5.5: MUX 与三变量表达式

5.3.2 MUX 与四变量表达式

从外面额外引入第四个变量. SOP 中的 P 可以被拆成 $A{\cdot}BCD$ 之类的玩意~~(就肯定又一个变量是做控制端的)~~. 这么描述也不太对…反正肯定有第四个变量是从外面引入的.

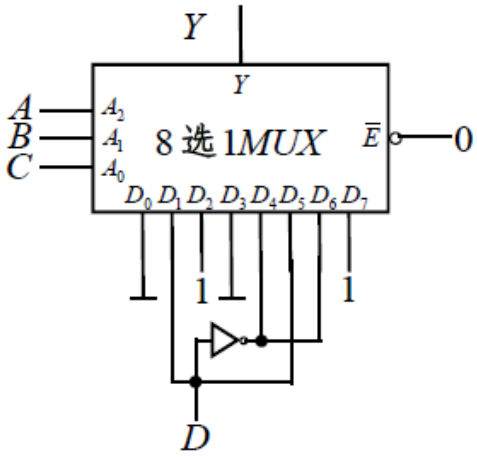

Figure 5.6: 四变量表达式的 MUX 的例子

ABC 决定输入的端口, 而 D 决定输入的端口的信号内容.

- $\mathbb{D}_0$ 接地为 0. SOP 表达式肯定没 $A'B'C'$ 这项
- $\mathbb{D}_1, \mathbb{D}_5$ 和 D 相连. 即 $A'B'C \cdot D, AB'C \cdot D$
- $\mathbb{D}_4, \mathbb{D}_6$ 和 D' 相连. 即 $AB'C' \cdot D', ABC' \cdot D'$
- $\mathbb{D}_2, \mathbb{D}_7$ 恒为 1. 即 $A'BC', ABC$

最后结果为

$$Y = A'B'C \cdot D + AB'C \cdot D + AB'C' \cdot D' + ABC' \cdot D' + A'BC' + ABC$$

5.3.3 数据选择器

就是 MUX 的别称, 没什么特别的.

5.4 Demultiplexer

不考

5.4.1 数据分配器

Demux 别称.

Chapter 6

Latch and Flip Flop

6.1 锁存器 (Latch)

Latch 不怎么考, 就是没有 CLK 的 FF, 一遇到输入信号就改变状态.

6.1.1 SR Latch

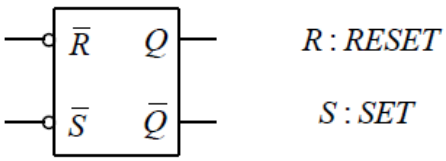

Figure 6.1: SR Latch

6.2 Flip Flop

有 CLK 的 Latch.
FF 的输出状态取决于输入信号和电路的原始状态.
FF 的 CP 端如果有 ▷ 则表示 rising edge (上升沿), 若在前面加一个 ∘ 表示 NOT 的话则为 falling edge (下降沿)

6.2.1 D FF

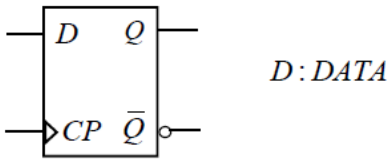

Figure 6.2: D FF

$$Q_{\text{next}} = D$$

6.2.2 SR FF

S for set (set to 0), R for reset (set to 1).
注意看输入是高电平有效还是低电平有效.

Table 6.1: Truth Table of SR FF

S	R	Q_{next}
0	0	Q
0	1	0
1	0	1
1	1	unknown

$$\begin{cases} Q_{\text{next}} = S + R' \cdot Q \\ S \cdot R = 0 \qquad \text{(S 与 R 不能同时取 1)} \end{cases}$$

6.2.3 JK FF

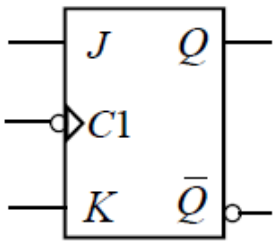

Figure 6.3: JK FF

$$Q_{\text{next}} = J \cdot Q' + K' \cdot Q$$

J 相当于 Set, K 相当于 Reset.

Table 6.2: Truth Table of JK FF

J	K	Q_{next}
0	0	Q
0	1	0
1	0	1
1	1	Q'

Table 6.3: Truth Table for T FF

T	Q_{next}
0	Q
1	Q'

6.2.4 JK 触发器转换

若 $J = K(= T)$, 可完成 T FF 的逻辑功能, $J = K'(= D)$ 可以完成 D FF 的逻辑功能

6.2.5 T FF

T for toggle.

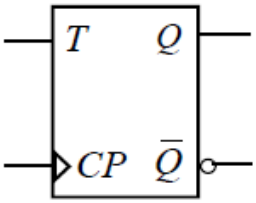

Figure 6.4: T FF

$$Q_{\text{next}} = T \cdot Q' + T' \cdot Q$$

6.2.6 触发方式

主从触发

边缘触发

6.2.7 带异步置位/复位的 FF

当异步复位端有效时, 输出 Q 被强制转换为 0.
置位端则使得 Q 被强制转换为 1.
由于是异步, 所以无视时钟 CLK 直接输出直接变化.

6.3 FF 之间的转换

- 写出两种 FF 的特性方程
- 对比转换电路的表达式
- 用门电路和目标 FF 画出转换电路

6.3.1 D FF to JK FF

~~Realise $J(D)$ 和 $K(D)$ (目标 FF 的输入变量为当前 FF 的输入变量的函数)~~ 只需要找到它们之间的关系即可, 不一定谁是谁的函数.

$$Q_{\text{next}} = D = D \cdot (Q + Q') = D \cdot Q + D \cdot Q'$$

$$Q_{\text{next}} = J \cdot Q' + K' \cdot Q \qquad \text{RHS 相等}$$

$$\begin{cases} J(D) = D \\ K(D) = D' \qquad \text{because } K' = D \end{cases}$$

6.3.2 JK FF to T FF

目标: ~~$T(J, K)$~~

$$Q_{\text{next}} = J \cdot Q' + K' \cdot Q$$
$$Q_{\text{next}} = T \cdot Q' + T' \cdot Q$$

$$\begin{cases} J(T) = T \\ K(T) = T' \end{cases}$$

6.3.3 D FF to T FF

$$\begin{cases} Q_{\text{next}} & = D \\ Q_{\text{next}} & = T \cdot Q' + T' \cdot Q \end{cases}$$

$$D(T, Q) = T \cdot Q' + T' \cdot Q = T \oplus Q$$

Chapter 7

时序逻辑电路分析

7.1 设计理论

由组合逻辑电路和存储电路构成, 又被称作状态机.

- Mealy 米利型 (输出和输入变量有关)
- Moore 摩尔型 (输出和输入变量无关)

描述方式:

- 驱动方程 (FF 的输入表达式/输入变量) (不包括 CLK)
- 状态方程 (将驱动方程代入到 FF 的特性方程得到)
- 输出方程 (就是输出变量的表达式)
- 状态真值表 (输入 | 现态 | 次态 | 输出)[1]
- 状态转换表 (次态/现态的输出)[2] (时钟 | 现态 | 输出)[3]
- 状态转换图

将输入变量以及出台带入状态方程和输出方程, 求得次态和初态的输出, 将其作为新的初态循环. 结束之后需要检查是否包含所有的结果.

[1]有时候没有输入或者输出可不填写
[2]Mealy 型电路
[3]异步

7.2 同步

7.2.1 Moore 型

例题 分析图中的电路, 描述其功能. [4]
摩尔型电路输出方程与输入无关.

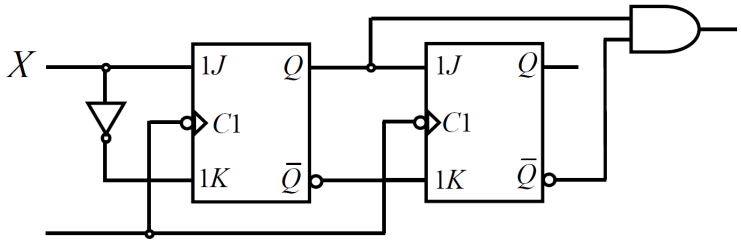

Figure 7.1: 摩尔型电路例题

▷ 前边有 ○ 故为 falling edge.

驱动方程 就是 JK FF 的输入.

$$\begin{cases} J_1 = X \\ K_1 = X' \end{cases} \quad \begin{cases} J_2 = Q_1 \\ K_2 = Q_1' \end{cases}$$

输出方程 右上角的 AND gate 大概就是输出.

$$Z = Q_1 \cdot Q_2'$$

状态方程 将驱动方程 JK FF 的特征方程代入 $Q^\star = J \cdot Q' + K' \cdot Q$ 可以得到

$$\begin{aligned} Q_1^* &= X \cdot Q_1' + (X')' \cdot Q_1 \\ &= X \cdot Q_1' + X \cdot Q_1 \\ &= X \cdot (Q_1' + Q_1) \\ &= X \\ Q_2^* &= Q_1 \cdot Q_2' + Q_1 \cdot Q_2 \\ &= Q_1 \cdot (Q_2' + Q_2) \\ &= Q_1 \end{aligned}$$

状态转换表 ~~状态转换表又被称作状态转移真值表~~如果是摩尔型[5]的电路则两者相同. 真值表顾名思义就是按照真值表二进制排列顺序列出来的东西 (区分那么多干啥, 两者似乎差不多)

[4] 左下角空着的是时钟 CLK, 右上角是输出 Z
[5] 输出和输入变量无关

得先假定一个输入, 在此我们先假设输入 $X=0$
还得假设一个初态, 在此假设 $Q_1, Q_2=0$
在此直接按照二进制顺序排列.

Table 7.1: 状态转换表/状态真值表

X	Q_1	Q_2	Q_1^*	Q_2^*	Z
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	0	0
1	1	0	1	1	1
1	1	1	1	1	0
1	0	1	1	0	0

状态转换图 如下图. (状态/输出) 为一个节点, 箭头指向次态, 箭头上方为输入.

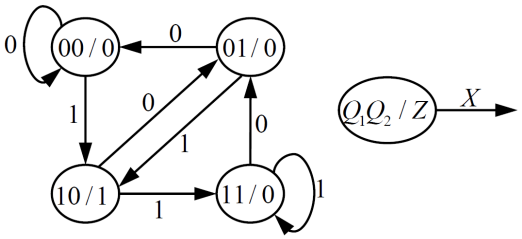

Figure 7.2: 状态转换图

7.2.2 Mealy 型

米利型电路输出方程与输入无关.

例题 分析下列电路

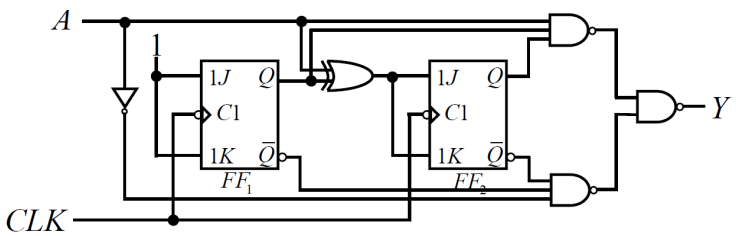

Figure 7.3: 米利型例题

驱动方程

$$\begin{cases} J_1 = 1 \\ K_1 = 1 \end{cases} \quad \begin{cases} J_2 = Q_1 \oplus A \\ K_2 = Q_1 \oplus A \end{cases}$$

输出方程

$$\begin{aligned} Y &= [(AQ_1Q_2)' \cdot (A'Q_1'Q_2')']' \\ &= AQ_1Q_2 + A'Q_1'Q_2' \end{aligned}$$

状态方程

$$\begin{aligned} Q_1^* &= J_1 \cdot Q_1' + K' \cdot Q_1 \\ &= Q_1' \\ Q_2^* &= (Q_1 \oplus A) \cdot Q_2' + (Q_1 \oplus A)' \cdot Q_2 \\ &= (Q_1 \oplus A) \cdot Q_2' + (Q_1 \odot A) \cdot Q_2 \end{aligned}$$

XOR 取反为 XNOR

$$\begin{aligned} A \oplus B &= A'B + AB' \\ (A \oplus B)' = A \odot B &= AB + A'B' \end{aligned}$$

状态真值表 (输入 | 现态 | 次态 | 输出) 如下表

Table 7.2: 状态真值表

A	Q1	Q2	Q1*	Q2*	Z
0	0	0	1	0	1
0	0	1	1	1	0
0	1	0	0	1	0
0	1	1	0	0	0
1	0	0	1	1	0
1	1	0	1	0	0
1	1	1	0	0	0
1	0	1	0	1	1

状态转换表 米利型的状态转换表与状态真值表不同, 表中的元素为 (次态/现态的输出)

Q_1Q_2 \ A	$Q_1^{n+1}Q_2^{n+1}$ / Y	
	0	1
00	10 / 1	11 / 0
01	11 / 0	10 / 0
10	01 / 0	00 / 0
11	00 / 0	01 / 1

Figure 7.4: 状态转换表

状态转换图　节点里面是状态, 输入和**现态**的输出被放到箭头上 (输出不再放到节点中, 因为和输入有关)

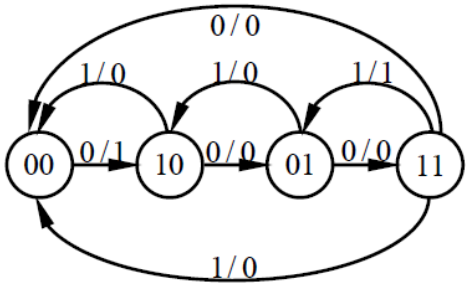

Figure 7.5: 状态转换图

7.2.3　计数器

状态转换图长这样

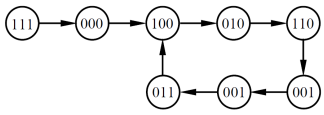

Figure 7.6: 计数器的状态转换图

圈里的闭环有几项就是几进制 (即模为几) 计数器, 该图为六进制计数器, 又被称作模 6 计数器.

位数

三位二进制加法计数器. 一个循环有 $2^3 = 8$ 个数 $0 \sim 7$. 即模为 8.

计数器设计

例子 构造一个模 10 计数器需要多少个状态? 需要几个 FF?
模 10 计数器需要记 10 个数, 所以就要 10 个状态.
一个 FF 可以表示 1 位二进制数, 所以把模转换为二进制. $(9)_D = (1001)_B$[6]

7.2.4 同步电路设计

给状态转换图设计时序电路

- 列写状态转移真值表 (根据状态转换图)
- ~~利用次态 K-map~~ 根据状态转换真值表直接列出次态各个 $\mathbb{Q}$ 的 K-map, 得到**状态方程**和输出方程
- 导出激励方程
- 画出电路图

给出描述设计时序电路

- 逻辑抽象 [7]
- 画状态转换图
- 画状态转换真值表 (可选)
- 补全真值表, 画出完整的状态转换图 (可选)
- ~~利用次态 K-map~~ 根据状态转换真值表直接列出次态各个 $\mathbb{Q}$ 的 K-map 以列出状态转换方程 (驱动/激励方程)
- 检查是否可以自启动, 若不能自启动者返回上一步
- 画出时序逻辑电路图

7.3 异步

7.4 集成计数器

四位同步十六进制计数器:74LS161/74LVC161.
四位十进制计数器: 74HC390

[6] 为啥用 9? 因为从 $0 \sim 9$ 就是 10 个数.
[7] 比如设计一个按照自然二进制变化的同步进制加法器, 那就要知道是一个 000_B 到 100_B 的循环 $0 \sim 4$

7.4.1 简介

以 74161 为例

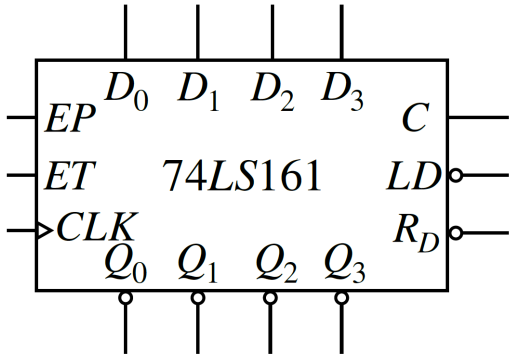

Figure 7.7: 74161

CLK	R'_D	LD'	EP	ET	工作状态
×	0	×	×	×	置 0（异步）
↑	1	0	×	×	预置数（同步）
×	1	1	0	1	保持（包括 C）
×	1	1	×	0	保持（$C = 0$）
↑	1	1	1	1	计数

Figure 7.8: 工作状态

端口讲解

- $\mathbb{D}$ 为置位 (输入)
- $\mathbb{Q}$ 为输出
- C/RCO 进位输出端
- EP/P 高电平有效时启用计数器
- ET/T 高电平有效时启用计数器 (高优先级)
- $R'_D/(CLR)'$ 异步清零端
- LD' 同步置数端, 若有效将会在下一时间周期将 $\mathbb{Q}$ 设置为 $\mathbb{D}$ 端口的数

功能

- +1 计数

- 同步预置
- 异步清零 (优先级最高)
- 计数保持 ($EP = 0, ET = 1$ 时不清除进位输出端)

7.4.2 置零法 (异步清零法)

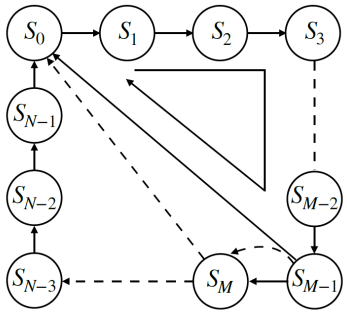

Figure 7.9: CLR' 逻辑表示

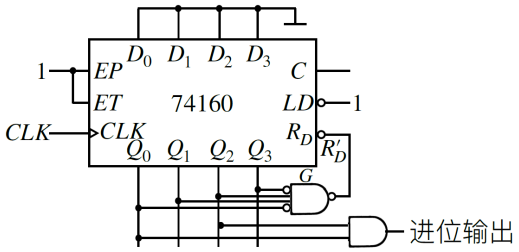

Figure 7.10: CLR' 电路图

计数初值一定为 0
令 $CLR' = 0$ 的最小 $\mathbb{Q}$ 值再减去 1, 就是它的**最终计数值**.
图中 $Q_3Q_2Q_1Q_0 = 0110_B = 6_D$, 即模 6 的计数器[8]. (一位六进制计数器)
但是其开始结束为 $0000_B \sim 0101_B$

[8]其最终计数值为 $6 - 1 = 5$, 即 $0 \sim 5$

7.4.3 置位法 (同步预置法)

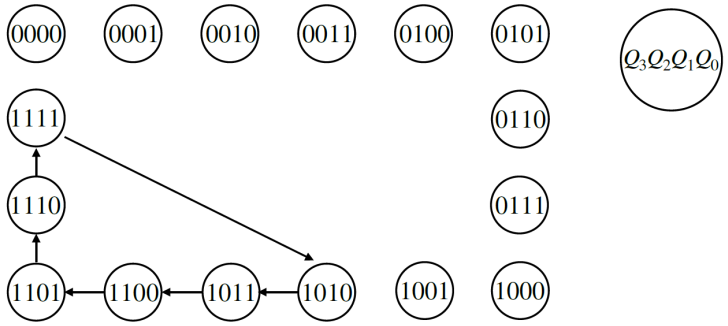

Figure 7.11: LD' 逻辑表示

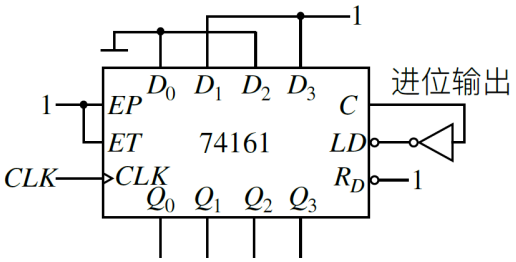

Figure 7.12: LD' 电路图

进位端接同步预置端
其计数初值为 $\mathbb{D}$ ($D_3D_2D_1D_0$)
可以令 $LD'=0$ 的最小 $\mathbb{Q}$ 值, 就是其计数终值. (若 $Q_3Q_2Q_1Q_0$ 不接, **进位端接同步预置端**, 则计数终值为 1111_B)

例子 下面是又一个例子 (进位端接同步预置端)

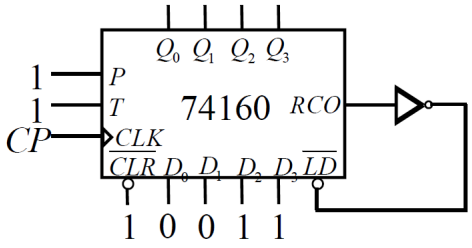

Figure 7.13: 同步预置接进位端

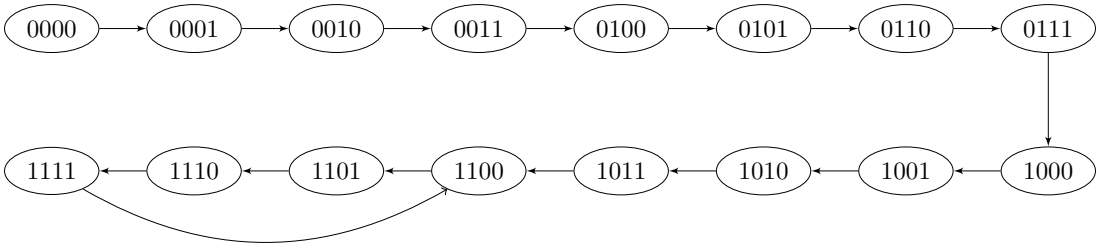

Figure 7.14: 同步预置接进位端的状态转换图

该图得到的是四位计数器.
由 $D_3D_2D_1D_0$ 为起始, 1111_B 结束

7.4.4 多个计数器串接

所需位数是否为质数 (Prime)

Chapter 8

脉冲的产生

数字脉冲信号就是矩形波.

8.1 触发器电路

8.1.1 施密特触发器

V_{T+} 和 V_{T-} T for Trigger.
施密特触发器是单调的.
施密特触发器可以用于脉冲整形, 波形变换, 鉴幅.

8.1.2 单稳态电路

单稳态 Toggler 特点是只有一个稳态, 剩下的是暂稳态. 在外界触发信号的作用下, Toggler 的状态能从稳态变化到暂稳态, 但是保持一段时间[1]之后就回自行变回稳态.

8.1.3 多谐振荡器

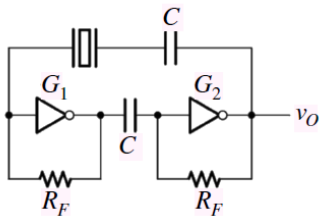

Figure 8.1: 石英晶体多谐振荡器

[1]维持暂稳态的时间长短取决于电路内部参数

多谐振荡器是一种自激振荡器, 在接通电源之后不需要外加触发信号即可产生矩形脉冲.

8.2 555 Timer

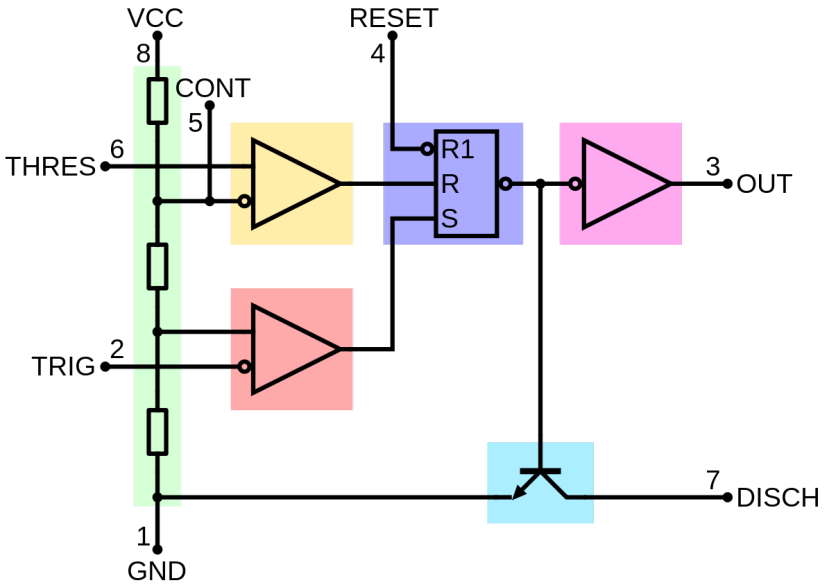

Figure 8.2: 555

由

- 两个比较器
- SR Latch
- 集电极开路的放电三极管

三部分组成.

8.2.1 构成施密特触发器

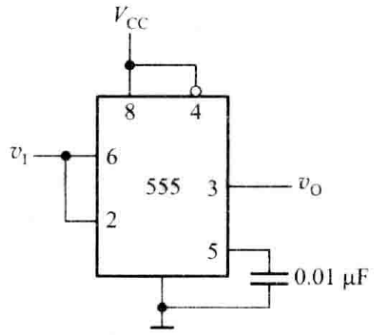

Figure 8.3: 555 施密特 (无外接 V_{CO})

V_{in} 只接 6 和 2 脚
若参考电压无外接电压 V_{CO} 供给 (5 端)

$$\begin{cases} V_{T+} = \frac{2}{3} \cdot V_{CC} \\ V_{T-} = \frac{1}{3} \cdot V_{CC} \end{cases}$$

若参考电压由外接电压 V_{CO} 供给 (5 端), 通过改变 V_{CO} 的值可以改变回差电压的大小.

$$\begin{cases} V_{T+} = V_{CO} \\ V_{T-} = \frac{1}{2} \cdot V_{CO} \end{cases}$$
$$\Delta V_T = \frac{1}{2} \cdot V_{CO}$$

8.2.2 构成单稳态电路

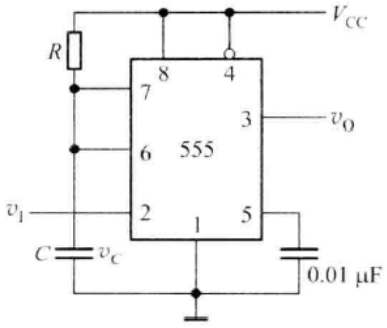

Figure 8.4: 555 单稳态

V_{in} 接 2 脚
高电平为暂稳态 (持续一段时间后掉回稳态), 低电平为稳态.

$V_{in} \leq \frac{1}{3} \cdot V_{CC}$, T_w 为暂稳态的持续时间. V_{in} 的低电平持续时间必须小于 T_w.

$$\ln(3) \approx 1.1$$

$$T_w = 1.1R \cdot C$$

输入脉冲宽度 (暂稳态的持续时间) 只和外接的电阻和电容有关

8.2.3 构成多谐振荡器

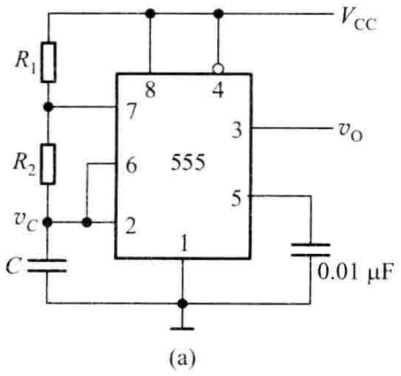

Figure 8.5: 555 多谐振荡器

多谐振荡器可以由施密特触发器得到, 所以将 555 TImer 接程施密特触发器之后, 再将 v_0 经过 RC 积分电路接回输入端, 即可得到多谐振荡器.
压根就没有 V_{in}, 由电路内部自行震荡. 有俩电阻 (R_1, R_2), 一电容 C, 一大小为 $0.01\mu F$ 的电容.
电容充电时, C 两端电压逐渐增大, 输出高电平. (反之为输出低电平)
充放电路径上经过的电阻, 将他们相加之后与电容相乘, 再乘以 0.7 得到其持续时间.

$$ln(2) \approx 0.7$$

高电平持续时间 (充电)

$$T_{\text{charge}} = 0.7 \cdot (R_1 + R_2) \cdot C$$

其充电路径是左边那一串电阻

低电平持续时间 (放电)

$$T_{\text{discharge}} = 0.7 \cdot R_2 \cdot C$$

放电路径是 2 脚到 7 脚 (一般是 R_2)

振荡周期 $T = T_{\text{charge}} + T_{\text{discharge}}$
占空比 $= \frac{T_H}{T} = \frac{T_{\text{charge}}}{T_{\text{charge}}+T_{\text{discharge}}}$