

坂井修一 著作 大纲

电子技术基础：时序组合 挺好，题目✓

数字与模拟 デジタルとアナログ

数字化特点: 处理速度快、记忆容易、传输容易、成本低、兼容性强、沉用性高

进数 Numeral system

Binary, Quaterary, Octal, Decimal, Hexadecimal

进制转换 $B \rightarrow + X_{n-1} \cdot B^{n-1} + X_{n-2} \cdot B^{n-2} + \dots + X_1 \cdot B + X_0$

$+ \rightarrow B$: 短除法, 除B取余, 倒序排列

$B \rightarrow =$: 几进制: 1位 $\rightarrow$ 36bit $+ =$: 1位 $\rightarrow$ 46bit

一般: 以 $=/+$ 为中介

小数: $B^{-1}, B^{-2} \dots$

$(0.6)_{10} \rightarrow (2), \begin{matrix} 0.6 \times 2 = 1.2 \dots 1 \\ 0.2 \times 2 = 0.4 \dots 0 \\ \vdots \end{matrix} \Rightarrow (0.6)_{10} = (0.100\dots)_2$
可有无限循环

有记号数

最高bit 作符号 $+5: 0.00101$

位数扩张: 整数前/小数后加0不改变

小数: 符号绝对值 <1 , 小数点前 0正 1负 $-0.75 = (1.11000)_2$

原码

n bit 整数: $-2^{n-1} \leq x \leq 2^{n-1}-1$

n bit 小数: $-1 + 2^{-(n-1)} \leq x \leq 1 - 2^{-(n-1)}$ $-1 < x < 1$

零: 0.0000/1.0000

反码

1の補数 正数=原码, 负数: 除符号位外逐位取反 零: 0.0000/1.1111

补码

2の補数 正数=原码, 负数: 反码末位+1 零: 0.0000

取反+1=原码

n bit 整数: $-2^{n-1} \leq x \leq 2^{n-1}-1$ 零只有0.00..., 1.00... 分给 -2^{n-1}

n bit 小数 $-1 \leq x < 1$

整数x的2の補

$x > 0, x$ $x < 0, 2^n - |x|$ $x_{\text{补}} = 1,110 \rightarrow 2^4 - |0,0110| = (14)_{10} = (1110)_2$ 非有符号数, 数字相同

1の補, $x < 0, 2^n - 1 - |x|$

小数.....

加法 (补码)

溢出 例: $0,011 + 0,101 \rightarrow 1,000 \rightarrow (-8)_{10} \times$ (正+正, 负+负) 发生溢出时符号位相反

$-3 + -6 = -9: 1,101 + 1,010 \rightarrow 0,111 \rightarrow (7)_{10} \times$

$-3 + -2 = -5: 1,101 + 1,110 = 1,011$

减法: $A \begin{cases} A > 0: \text{仅改符号位} \rightarrow -A \\ A < 0: \text{取反} + 1 \rightarrow -A \end{cases}$

$$\begin{array}{r} 1101 \\ + 1110 \\ \hline 11011 \end{array}$$

未溢出, 抛弃多余位
若位数不齐, bit扩张

乘法

$n \times n \rightarrow 2n$ bit

bit扩张: 在符号位与数值之间, 正数补0, 负数补1

正x负, 错位相加时, 符号位所得那一步错位相减

负x正, 负数bit扩张

负x负, 同上。

$$\begin{array}{r} 5 \times (-3) = -15 \\ \begin{array}{r} 0101 \\ \times 1101 \\ \hline 0000101 \\ 0000000 \\ 0001010 \\ 0001001 \\ \hline 00011001 \\ -00101 \\ \hline 1110001 \end{array} \end{array}$$

$$\begin{array}{r} -5 \times 3 \\ \begin{array}{r} 1011 \\ \times 0011 \\ \hline 111011 \\ 111011 \\ \hline 000000 \\ 000000 \\ \hline 1110001 \end{array} \end{array}$$

除法

无符号: $0 \div 1 = 0, 1 \div 1 = 1$

$$\begin{array}{r} 000111 \\ 110 \overline{) 101010} \\ \underline{110} \\ 1001 \\ \underline{110} \\ 110 \end{array}$$

移位: 正数: 左移末位添0, 右移首位添0.
负数: 左移末位添0, 右移首位添1

计算机定点数

算术移位: 有符号数, 符号位不移

左移: 逐位左移, 补0
右移: 补0
补: 左移补0, 右移补1

原: 10110101 $\xrightarrow{\text{左移一位}}$ $01011010 = (10b)_{10}$ 在0
右: 10110101 $\xrightarrow{\text{右移一位}}$ $11011010 = (26b)_{10}$ 在1
补: 10110101 $\xrightarrow{\text{右移一位}}$ 11011010 在1

有符号时的通用计算步骤:

1. 将被除数扩张至 $2n-1$ bit, 记作 $D0$ (遵守之前提过的bit扩张规则)
2. 将除数左移位 $n-1$ bit, 记作 $D1$
3. 商的初始值设置为 $Q=0$
4. 当 $D0$ 与 $D1$ 同符号或 $D0=0$ 时, 更新 $D0=D0-D1$; 若不满足前面的条件则更新 $D0=D0+D1$
5. 新 $D0$ 与 $D1$ 同符号或新 $D0=0$ 时, Q 的最右位改成1; 若不满足前面的条件则 Q 的最右位改成0
6. 将 $D1$ 右移位1 bit, 将 Q 左移位1 bit
7. 重复第(4)~(6)步 $n-1$ 次后, 再最后进行一次第(4)和第(5)步
8. Q 的最右位是0时, 更新 $D0=D0+D1$
9. 商为 Q , 余数为 $D0$, 且此时余数的符号与除数相同

$$① 5 \div 3 = 1 \cdots 2 \quad 0101 \div 0011 = 0001 \cdots 0010$$

$$n=4 \quad 2n-1=7 \quad n-1=3$$

$$② D_0 = 0000101, D_1 = 0011000$$

$$④ \begin{array}{r} 0000101 \\ - 0011000 \\ \hline 1101101 = D_0 \end{array} \quad Q=0$$

$$⑤ Q=0 \quad ⑥ D_1 = 0001100 \quad Q=00$$

⑦重复④~⑥

$$ii) D_0 = D_0 + D_1 = 1111001, D_1 = 0001100 \quad Q=00$$

$$Q=000, D_1 = 0000110$$

$$iii) D_0 = D_0 + D_1 = 1111111, D_1 = 0000110, Q=000$$

$$Q=0000, D_1 = 0000011$$

$$iv) D_0 = D_0 + D_1 = 0000010 \quad Q=0001 \text{ 商} \quad \text{余数 } 0000010$$

逻辑函数

输入端理变数, 返回端理变数, 二进制 $\rightarrow$ 二进制 $F(x_0, x_1, \dots, x_{m-1}) \rightarrow (y_0, y_1, \dots, y_{n-1}), x_i, y_i \in \{0, 1\}$

真理值表

x_0	x_1	$F(x_0, x_1)$
0	0	0
0	1	1
1	0	0
1	1	0

布尔代数 MIL图

$F = A \cdot B = C \cdot \bar{D}$

AND 与门 $F = X \cdot Y$  $X \cdot 0 = 0, X \cdot 1 = X, X \cdot X = X, X \cdot \bar{X} = 0$

OR 或门 $F = X + Y$  $X + 0 = X, X + 1 = 1, X + X = X, X + \bar{X} = 1$

NOT 非门 $F = \bar{X}$  $F = X \cdot \bar{Y}$

NAND 与非门 $F = \overline{X \cdot Y}$ 

NOR 或非门 $F = \overline{X + Y}$ 

XOR 异或门 $F = X \oplus Y = \bar{X} \cdot Y + X \cdot \bar{Y}$  同为0 异为1

EQ 同或门 $F = X \odot Y = \overline{X \oplus Y} = \bar{X} \cdot \bar{Y} + X \cdot Y$ 

★ 加号, 乘号优先级同级
 $X + Y \cdot Z = (X + Y) \cdot (X + Z), X \cdot (Y + Z) = X \cdot Y + X \cdot Z$
 ド・モルガンの法則 $\overline{X + Y} = \bar{X} \cdot \bar{Y}, \overline{X \cdot Y} = \bar{X} + \bar{Y}$

化简原则: ① $X \cdot Y + X \cdot \bar{Y} = X$ ② $X + X \cdot Y = X$ ③ $(X + \bar{Y}) \cdot Y = X \cdot Y$ ④ $(X + Y) \cdot (X + \bar{Y}) = X$
 ⑤ $X \cdot (X + Y) = X$ ⑥ $X \cdot \bar{Y} + Y = X + Y$ 左 = $X \cdot \bar{Y} + (Y + X \cdot Y) = Y + X \cdot (Y + \bar{Y}) = X + Y$

蕴含律 ① $X \cdot Y + Y \cdot Z + \bar{X} \cdot Z = X \cdot Y + \bar{X} \cdot Z$
 $= X \cdot Y + Y \cdot Z (X + \bar{X}) + \bar{X} \cdot Z = X \cdot Y + Y \cdot Z (1 + Z) + \bar{X} \cdot Z (1 + Y) = X \cdot Y + \bar{X} \cdot Z$
 ② $(X + Y) \cdot (Y + Z) \cdot (\bar{X} + Z) = (X + Y) \cdot (\bar{X} + Z)$
 $= (X + Y) \cdot (Y + Z + X \cdot \bar{X}) \cdot (\bar{X} + Z) = (X + Y) \cdot (Y + Z + X) \cdot (Y + Z + \bar{X}) \cdot (\bar{X} + Z)$
 $= [X + Y + (0 \cdot Z)] \cdot [\bar{X} + Z + (0 \cdot Y)] = (X + Y) \cdot (\bar{X} + Z)$

eg: $ab\bar{e} + ac\bar{e}f + a\bar{d}\bar{e} + b\bar{c} + bf + \bar{c}\bar{d} + \bar{d}\bar{f}$
 $= ac\bar{e}f + b\bar{c} + bf + \bar{c}\bar{d} + \bar{d}\bar{f}$
 $LHS = ac\bar{e}f + a\bar{e}(b + \bar{d}) + (b + \bar{d})(\bar{c} + \bar{f})$
 $= ac\bar{e}f + a\bar{e}(b + \bar{d}) + (b + \bar{d}) \cdot \bar{c}\bar{f}$
 $= a\bar{e}\bar{c}\bar{f} + (b + \bar{d}) \cdot \bar{c}\bar{f}$
 $= a\bar{e}\bar{c}\bar{f} + (b + \bar{d})(\bar{c} + \bar{f}) \cdot \bar{c}\bar{f} \cdot 1$

展开与因式分解

$(X + Y) \cdot (\bar{X} + Z) = XZ + \bar{X} \cdot Y$ $X \cdot Y + \bar{X} \cdot Z = (X + Z) \cdot (\bar{X} + Y)$

对偶 将F中所有 {., +, 0, 1} 替换为 {+, ., 1, 0}. $F \rightarrow F^D$
 $F = F', \text{ 则 } F^D = F'^D$

多输入因子

AND: 全1为1 OR: 全0为0 NAND: 全1为0, NOR: 全0为1
 XOR: 奇数1为1 EQ: 偶数1为0 N为奇: $EQ = XOR$; N为偶: $EQ = NOT \text{ XOR}$

完备性

$N=2, \{AND, OR, NOT\}$ 是完备集合
 $F(x_0) = Y, F(x_0) = NOT(Y)$ ✓
 假设M输入 $F(x_0, x_1, \dots, x_{m-1}) = Y$ 可用AND, OR, NOT表示
 M+1输入 $F(x_0, x_1, \dots, x_M) = OR(AND(F'(x_0, \dots, x_{M-1}, 0), NOT(x_M)), \dots, F'(x_0, \dots, x_{M-1}, 1), x_M)$ 可用三者表示口
 其中 $F'(x_0, \dots, x_{m-1}, 0) = F(x_0, \dots, x_{m-1})|_{x_m=0}$
 $F'(x_0, \dots, x_{m-1}, 1) = F(x_0, \dots, x_{m-1})|_{x_m=1}$
 ∴ RHS 可用 {NOT, AND, OR} 表示

其他完备集合

{OR, NOT} : $X \cdot Y = \overline{\overline{X} + \overline{Y}}$ 用 OR, NOT 替换 AND

{AND, NOT} : $X + Y = \overline{\overline{X} \cdot \overline{Y}}$

{NAND} : $\text{NOT}(X) = \text{NAND}(X, X)$, $\text{OR}(X, Y) = \text{NAND}(\overline{X}, \overline{Y})$

{NOR} : $\text{NOR} \rightarrow \text{NOT} \cdot \text{AND}$ $\text{NOT}(X) = \text{NOR}(X, X)$, $\text{AND}(X, Y) = \text{NOR}(\overline{X}, \overline{Y})$

標準形

变量/リテラル/Literal $X, \overline{X}$

積項·和項 $X \cdot Y \cdot Z, X + Y + Z$

最小項·最大項 n 变量逻辑函数中, n 个变量以原/非的形式在积项, 或项中仅出现一次

Minterm: $X \cdot Y \cdot Z, \overline{X} \cdot Y \cdot Z \dots$

Maxterm: $(X + Y + Z), (X + \overline{Y} + Z) \dots$

加法標準形 最小项的逻辑和,

$X \oplus Y = X \cdot \overline{Y} + \overline{X} \cdot Y$, $X \odot Y = X \cdot Y + \overline{X} \cdot \overline{Y}$

所有 F 都可用加法标准形表示。利用 $F(x_0, x_1, \dots, x_n) = \text{OR}(\text{AND}(F'(x_0 \dots x_{n-1}, 0), \text{NOT}(x_n)), \text{AND}(F'(x_0 \dots x_{n-1}, 1), x_n))$ 反复拆分

$$F(x_0, x_1, \dots, x_{n-1}) = \overline{x_0} \cdot \overline{x_1} \dots \overline{x_{n-1}} \cdot F(0, 0, \dots, 0) + \overline{x_0} \cdot \overline{x_1} \dots \overline{x_{n-1}} \cdot F(1, 0, \dots, 0) + \dots + \overline{x_0} \cdot x_1 \dots x_{n-1} \cdot F(1, 1, \dots, 1)$$

找输出为1的输入 ($F(\dots) = 1$), 则该项保留。

↓

加法标准形 = 输出为1的最小项之逻辑和。

MIL 图: 先过 AND, 再过 OR

表 2.12 4 输入 1 输出逻辑函数的真值表

X	Y	Z	W	O
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

(1 0 1 1)₂ = $1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 11$

(2) (3) (4) (6) (8) (10) (11) (15)

$0 = \overline{X} \overline{Y} \overline{Z} \overline{W} + \overline{X} \overline{Y} Z \overline{W} + \overline{X} Y \overline{Z} \overline{W} + \overline{X} Y Z \overline{W} + X \overline{Y} \overline{Z} \overline{W} + X \overline{Y} Z \overline{W} + X \overline{Y} \overline{Z} W + X \overline{Y} Z W$

$0 = \sum (2, 3, 4, 6, 8, 10, 11, 15)$

注意变量排序。

乘法標準形 最大项的逻辑和

利用 $F(x_0, \dots, x_{n-1}) = \overline{x_{n-1}} F(x_0 \dots 0) + x_{n-1} \cdot F(x_0 \dots 1)$

$= [\overline{x_{n-1}} + F(x_0, x_1, \dots, x_{n-2}, 1)] \cdot [x_{n-1} + F(x_0, x_1, \dots, 0)]$

$F(x_0, \dots, x_{n-1}) = (\overline{x_0} + \overline{x_1} + \dots + \overline{x_{n-1}} + F(1, 1, \dots, 1))$

$\cdot (\overline{x_0} + \overline{x_1} + \dots + \overline{x_{n-1}} + F(1, 0, \dots, 1))$

$\vdots (\overline{x_0} + \overline{x_1} + \dots + \overline{x_{n-1}} + F(1, 0, \dots, 1))$

$\cdot (\overline{x_0} + \dots + \overline{x_{n-1}} + F(0, \dots, 0))$

MIL 图: 先 OR, 再 AND

$(X + Y)(\overline{X} + Z) = XZ + \overline{X}Y$

找输出为0的输入 ($F() = 0$) 若 $F(\dots) = 1$, 则 $X+1=1$, 舍弃 X
 eg. $0 = \pi(0, 1, 5, 7, \dots)$ 结果与 $\Sigma(2, 3, 4, 6, \dots)$ 加法标准形互补.

数字电路 デジタル回路

组合电路: 输出取决于输入, 与自身状态无关

时序电路: 输出与输入和自身状态有关
 { Moore: 仅与状态有关
 { Mealy: 状态+输入

组合电路

化简: I/O真值表 $\rightarrow$ 加法标准形 $\rightarrow$ 化简逻辑函数 $\rightarrow$ MIL

减少延迟、元件数量 $\rightarrow$ 减少输入个数, 逻辑门个数

串行元件, 元件I/O个数.

逻辑相邻项: 仅有一个输入相反的最小项

布尔代数化简 $x_1 \dots x_i \dots x_n + x_1 \dots \bar{x}_i \dots x_n = x_1 \dots x_{i-1} x_{i+1} \dots x_n$ 消去 x_i

① $A+A=A$ ② $A+\bar{A}=1$ ③ $A \cdot 1=A$

eg. $\bar{x} \cdot y + x \cdot \bar{y} + x \cdot y = \bar{x} \cdot y + x \cdot \bar{y} + x \cdot y + x \cdot y = x+y$

$\bar{x} \cdot yz + x \cdot yz + \bar{x} \cdot \bar{y}z + x \cdot \bar{y}z = yz + \bar{x} \cdot \bar{y}z + x \cdot \bar{y}z = xy + \bar{x}z$

卡诺图化简 2~6输入 逻辑相邻项在图中相邻

① N输入, 2^N 格子.

2输入

XY	0	1
0		
1		

3输入

YZ	00	01	11	10
0				
1				

4输入

ZW	00	01	11	10
00				
01				
11				
10				

5输入

$X=1$

WV	00	01	11	10
00				
01				
11				
10				

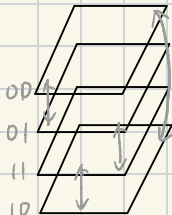
$X=0$

WV	00	01	11	10
00				
01				
11				
10				

或

CD	E				E			
AB	00	01	11	10	10	11	01	00
00								
01								
11								
10								

6输入



或 4^3 立方体, 对面相邻

- 卡诺图, 圈出相邻的值为1的项. 圈尽可能大; 个数尽可能少, 允许重叠, 不可包含.
- 每一个卡诺圈对应一个AND积项.

4 输入

ZW \ XY	00	01	11	10
00	0	1	1	0
01	0	0	0	0
11	0	1	1	0
10	0	1	1	0

→ 消Z: $F = \bar{X}\bar{Y}Z + \bar{X}\bar{Y}Z + \bar{X}\bar{Y}Z = \bar{X}\bar{Y}Z$

④ 消去原则: 某圈中, 某个变量 0, 1 都有, 则该变量可消, 即与该变量无关
圈大小为2, 可消去1个

→ 消Z, 消Y $XYZW = 1 \times 1 \times 1 \times 1 \leftarrow$ 简写

主项



○: 必选项
○: 多选项 } 3个主项



○: 可选主项 (=选-)

Quine-McCluskey 化简 QMC

Step1. 将逻辑函数写成加法标准形Σ形式, 并且将代表各最小项的序号数字写成二进制形式。

Step2. 将二进制形式的最小项按照含1的个数分组, 含1个1的最小项分入第1组中。这个分组叫做第一次分组。

Step3. 将连续的两个组(第1组和第i+1组)中的所有元素两两比较, 若有相邻的两个最小项, 则将这两项合并(0与1都出现过的位置用*代替)。找到相邻对象的二进制最小项后用\$符号标记。

Step4. 由step3得到的新的合并后的项中, 再次按照含1的个数进行分组。这叫做第二次分组。

Step5. 重复step3和step4, 直到不能再发现逻辑相邻的项。注意这期间*和*算作相同。这一步结束后, 没有\$印记的项就是主项。

Step6. 以主项为行, 加法标准形的最小项为列制作表格, 当主项含有最小项时, 在对应的格子中记上(>)印记。

Step7. 找到只含有一个>印记的列, 将这个>印记改成⊙。此时对应的主项叫做必须项(essential term)。紧接着将含有⊙行的所有>都改成○。

Step8. 将含有⊙和○的行与列从step6做成的表中全部删除。

Step9. 从删减后的表中, 选择最简单的主项组合, 但需要保证每一列都至少有一个1个>。

Step10. 将step7中得到的必须项和step9中选出的主项写成逻辑和形式, 即简化结果。

$$F = \Sigma(0, 2, 8, 10, 14)$$

$$\Sigma(0000, 0010, 1000, 1010, 1110)$$

$$\begin{array}{l} 01 \quad 0000 \quad \$ \\ 11 \quad 0010 \quad \$ \\ 11 \quad 1000 \quad \$ \\ 21 \quad 1010 \quad \$ \\ 31 \quad 1110 \quad \$ \\ 00*0, *000, *010, 10*0, 1*10 \end{array}$$

$$\begin{array}{l} 0 \quad 00*0 \quad \$ \\ *000 \quad \$ \\ 1 \quad *010 \quad \$ \\ 10*0 \quad \$ \\ 2 \quad 1*10 \quad \$ \\ *0*0, *0*0, \\ 0 \quad *0*0 \end{array}$$

$$00000001000101110$$

$$\begin{array}{l} *110 \quad 0 \quad \odot \\ *0*0 \quad \odot \quad \odot \quad \odot \quad 0 \end{array}$$

$$F = *10 + *0*0 = XZ\bar{W} + \bar{Y}\bar{W}$$

e.g. 0~15内 4bit 二进制数, 属于FIB 返回1.

$$0 \sim 15: 1 \quad 2 \quad 3 \quad 5 \quad 8 \quad 13$$

$$\Sigma(0001, 0010, 0011, 0101, 1000, 1101)$$

ZW \ XY	00	01	11	10
00	0	1	1	0
01	0	1	0	0
11	0	1	1	0
10	0	1	1	0

$$\bar{X}\bar{Y}W + \bar{X}\bar{Y}Z + YZ\bar{W} + X\bar{Y}Z\bar{W}$$

0~9内

Don't care 输出: 任取 0/1 作为输出, "x" 表示

卡诺图降维

每次去除一个变量的化简

AB \ CD	00	01	11	10
00			1	
01	1			1
11		1		
10			1	1

去B

ACD	00	01	11	10
0		B	B	1
1	B	B	1	B

多输出组合电路化简

当一个逻辑函数有多输出时，我们当然可以对每个输出进行单独的化简。但是每个输出往往存在共用项，所以单独对每个输出进行化简不能达到最佳的化简效果。

次输出：P，主项 PQ，P；等输出P的输出的表中没有同样的卡诺图。

化简步骤：

Step1. 先将多输出逻辑函数所有输出项进行积项排列组合，然后求出各输出组合的主项（求法：1.卡诺图里一个卡诺圈即一个主项；2.见QM法）。

Step2. 以主项为行，加法标准形的主项为列制作表格，当主项含有最小项时，在对应的格子中记上（>）印记。注意这里主项包含最小项指的是单独某个输出值的主项与此输出值的最小项之间的关系。

Step3. 找出只含有一个>的列，将该列中的>改成⊙。此时对应的主项叫做必须项。然后将该输出项下有⊙的行中的>全部改成○。

Step4. 将含有⊙和○的列从step2做成的表中全部删除。若某一行每格都是⊙或○则将该行删除。

Step5. 从删减后的表中，选择最简单的主项组合，但需要保证每一列都至少有一个1个>。

Step6. 将step3中得到的必须项和step5中选出的主项写成逻辑和形式，即简化结果。

① 先选可复用的

② 再选简单的

$$P = \sum(2, 5, 6, 7, 13, 15)$$

$$Q = \sum(2, 6, 10, 11, 14, 15)$$

$$R = \sum(5, 10, 11, 13, 15)$$

首先，对输出进行积项排列组合：

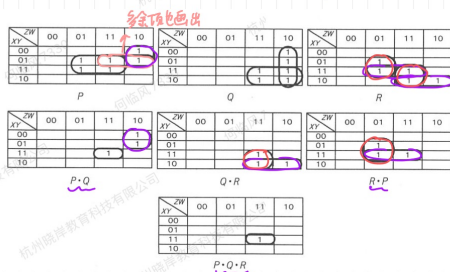
$$P \cdot Q = \sum(2, 6, 15)$$

$$Q \cdot R = \sum(10, 11, 15)$$

$$R \cdot P = \sum(5, 13, 15)$$

$$P \cdot Q \cdot R = \sum(15)$$

接着画出各积项的卡诺图，并画出所有可能的卡诺圈，包括多余的



常见组合电路

加法器

HA

半加器：输入中无下位的进位

全加器 FA 输入：X Y C_{in} 输出：S C_{out}

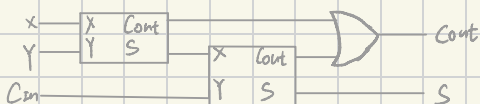
$$S = X \oplus Y \oplus C_{in}, C_{out} : \text{有2个1以上}$$

$$\text{则 } S = \Sigma(1, 2, 4, 7), C_{out} = \Sigma(3, 5, 6, 7), S, C_{out} = \Sigma(17) \text{ 多输出化简}$$

$$\text{化简得: } S = \bar{x}\bar{y}C_{in} + \bar{x}y\bar{C}_{in} + x\bar{y}\bar{C}_{in} + xyC_{in}$$

$$C_{out} = yC_{in} + xC_{in} + xy$$

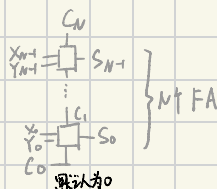
全加器可由2个半加器构成。



行波进位加法器：N bit 运算

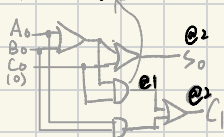
Ripple Carry Adder 延迟高。

$$C_{out} \rightarrow C_{in} \text{ (F-4)}$$



延迟分析：

AND $\Rightarrow$ 只要有一输入为0，不用等另一输入。



第一个加法器 (N=1)

$$\begin{aligned} \text{例: } C_0=0, S_0@2, S_1@3, S_2@5 \dots \\ C_1@2, C_2@4, C_3@6 \dots \\ N=1 \quad N=2 \end{aligned}$$

Critical data path: 延迟最大路径。N bit RCA 为 $2NT$

溢出判断 同号相加，符号位变化 (N-1 位)

X_{N-1}	Y_{N-1}	C_{N-1}	S_{N-1}	C_N	即结果符号位
0	0	0	0	0	✓
0	0	1	1	0	] overflow
1	1	0	0	1	
1	1	1	1	1	✓

$$\text{溢出} = C_{N-1} \oplus C_N$$

最高位加入 XOR 门即可

超前进位加法器

$$\text{Carry Lookahead Adder } C_{i+1} \text{ (进位)} = X_i Y_i + (X_i + Y_i) C_i$$

$$C_0 = 0, C_1 = X_0 Y_0$$

$$C_2 = X_1 Y_1 + (X_1 + Y_1) C_1 = X_1 Y_1 + (X_1 + Y_1) X_0 Y_0$$

⋮

$$C_i = \sum_{j=1}^i \{ X_j Y_j \cdot \prod_{k=j+1}^i (X_k + Y_k) \}$$

进位产生信号：当前位即可进位 $G_i = X_i \cdot Y_i$

进位传递信号：前一位加进位方可进位 $P_i = X_i \oplus Y_i$

$\rightarrow P_i G_i$ 仅与 $X_i Y_i$ 有关，可同时输出
 $C_n @ 3 \quad S_n @ 4$

$$S_i = P_i \oplus C_i, C_{i+1} = X_i Y_i + (X_i + Y_i) C_i = X_i Y_i + P_i C_i = G_i + P_i C_i$$

CLA 级联 cascading

2x2bit → 4bit, 4x4bit → 16bit

选取第3块为例, 总体 G_3, P_3

$$\begin{cases} \text{该块进位} \begin{cases} \text{最高位 } g = 1 \quad (g_2 = 1) \\ \text{最高位 } p = 1, \text{前一位进位 } g = 1 \quad (g_2 = 1, p_3 = 1) \end{cases} \\ \text{所有内部 } p_i = 1 \end{cases} \left. \begin{matrix} G_3 = 1 \\ P_3 = 1 \end{matrix} \right\}$$

$$(N \text{ bit}, \frac{N}{k} \text{ 块}) t_{CLA} = t_{pg} + t_{pg-block} + (\frac{N}{k} - 1) t_{AND-OR} + k t_{FA}$$

t_{pg} : 块内部 pg 信号时间

$t_{pg-block}$: 块产生总 pg 时间

t_{AND-OR} : 每块位总累计产生进位 (串行)

t_{FA} : FA 延迟。 (最后一块利用 ACA 算出 $S_0 \sim S_{k-1}$)



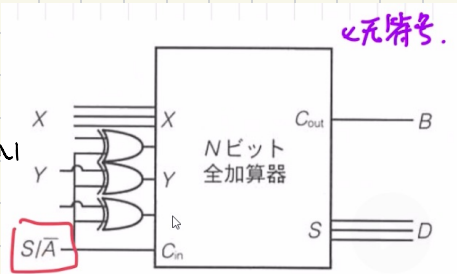
减法器 (无符号数)

利用 Adder 实现

$S/A = 0$ 表示 add

$= 1$ 表示 sub

对 Y 取反并以 C_{in} 输入



乘法器

利用 Adder 实现 (无符号数)

取 AND 后错位相加。

译码器

Decoder 二进制 → 十进制

3-8 译码器: 3bit 输入 8bit 输出 (A 表 0~7)

5-32: 用 2-4, 3-8 构成:

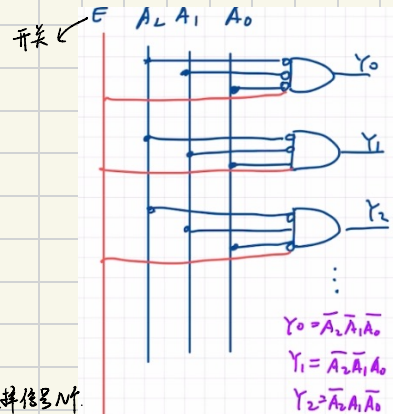
分为 4 组, 每组用 3-8.

编码器

十 → 二.

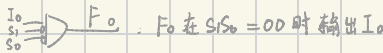
输入存在优先级, 优先级最高的被转码

比如回输入 $Y_2, Y_1, Y_0, Y_3 = 1, 111 \rightarrow 11$



多路选择器

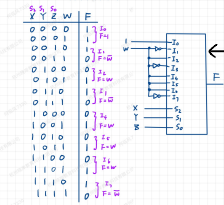
multiplexer 多个输入选择 1 个输出。输入 2^N 个, 选择信号 N 个。



用一个 4-MUX 汇聚 4 个 4-MUX, 形成 16-MUX.

N -MUX: 从 N 个输入中选择

用 8-MUX 实现 210, 1, 2, 3, 4, 5, 6, 7, 12, 13, 14



任意 $N+1$ 输入逻辑函数可用 2^N 输入 MUX 实现: 选择 N 个作为选择信号

MUX 输入数不够: 降维

多路分配器

即译码器

比较器

$$1 \text{ bit } \begin{cases} A < B: \bar{A} \cdot B \\ A = B: A \cdot B + \bar{A} \cdot \bar{B} \\ A > B: A \cdot \bar{B} \end{cases}$$

$$N \text{ bit } \begin{cases} A = B: \overline{A_i \oplus B_i}, \forall i \end{cases}$$

, $A > B$: 从高位到低位, 前j位相等, $A_j = 1, B_j = 0$

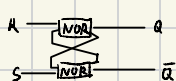
奇偶生成校验器

Nbit 一串二进制, Parity: 偶数1为0, 奇数1为1 $\xrightarrow{1 \text{ bit flip}}$ New parity = 1 不能修正, 仅检测 1 bit flip.
 添上 Parity 位, 共 N+1 位, 新 Parity 必为 0.



SR Latch 锁存器

存在自我反馈



稳态: $Q=1, \bar{Q}=0 / Q=0, \bar{Q}=1$

① $S=1, R=0 \rightarrow Q=1, \bar{Q}=0$

② $S=0, R=1 \rightarrow Q=0, \bar{Q}=1$

③ $S=R=0$ UB, 由①翻转S, 不影响, 由②翻转R, 不影响

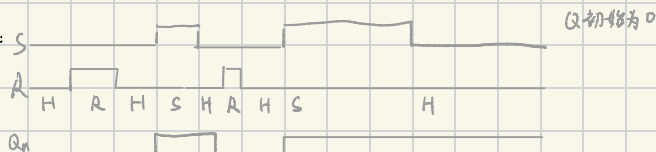
$(S,R) = (0,0)$, hold 保持上-状态 $(S,R) = (1,0)$, set $Q=1$ $(S,R) = (0,1)$, reset $Q=0$

④ $S=R=1 \rightarrow Q=\bar{Q}=0$, 不合法。禁止 $(S,R) = (1,1)$ 即不希望有这样的输入

Qnext	Q	R	S
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1

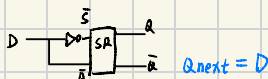
$Q_{next} = \bar{R}Q + S$

动作图:



D Latch delay, 延迟输出, 消除 UB 状态

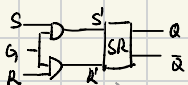
硬件延迟导致 hazard (如 (1,1))



$Q_{next} = D$

Gated Latch

增加 G 作为开关

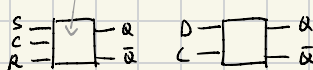


$G=0, (S',R') = (0,0)$, hold

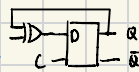
$G=1, (S',R') = (S,R)$ 即 SR-latch

触发器 Flip-flop 引入时钟信号, 保证 hazard 时 Gate = 0

也叫同步触发器



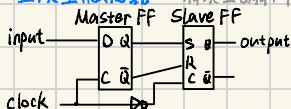
空翻:



改变输入, 一个时钟周期内状态多次震荡

主从型触发器

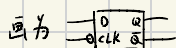
解决空翻问题



主、从有效周期相反

clock = 1, 主有效从无效, hold

clock = 0, 主无效从有效, input 翻转不影响主FF, 从FF从主FF中继承信号

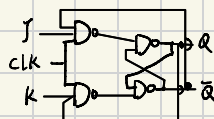


JK FF

解决 (S,R) 中 (1,1) 的限制 jump/kill

Q _n \ J K	00	01	11	10
0	0	1	0	0
1	1	1	0	0

$$Q_n = J\bar{Q} + \bar{K}Q$$



$(J, K) = (0, 1)$, kill $\rightarrow Q = 0, \bar{Q} = 1$ 稳态

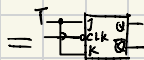
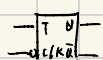
Q, Q-bar 又作输入

$(J, K) = (1, 1)$ 时, race condition 不断翻转

主从型 JK FF



T 触发器



$$Q_n = Q\bar{T} + \bar{Q}T$$

边沿触发器

主从触发器的硬件延迟不能 > 时钟周期

△ JK 主从 FF 一次翻转问题: ① $Q = 0$, J 变化引起 Q 主一次 flip, K 不起作用 ② $Q = 1$, Viso versa

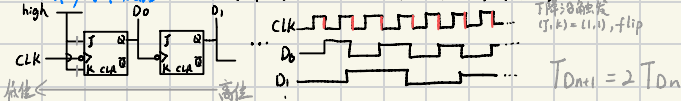
由时钟信号上升/下降时触发。



时序电路

异步式计数器

记录周期数。前一个FF的Q为下一个FF的时钟信号

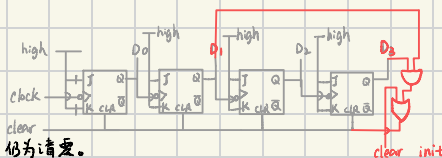


类似 Ripple Carry Adder, 延迟高 $N\text{bit} \rightarrow N\text{倍延迟}$

非 2^N-1 计数器 $0 \sim M-1$

① 大于 M 的 2^N ② 到 M 清零.

$M=0$	$N=4 \rightarrow 4$ 进制计数器, 到 10 清零 (1010) ₂
-------	---



caveat: 由于延迟, 会有一段时间达到M仍为清爱。

异步倒序计数器改为上升沿触发

正序计数器 + 倒序 + 2-MUX : 正/倒序切换



X-M计数器

① 到 $M+1$ 时重置为 X ② $0 \sim M-X$ 输出加 X 无延迟优势

与组合回路结合

e.g. 初始 5T 输出 1, 后面输出 0

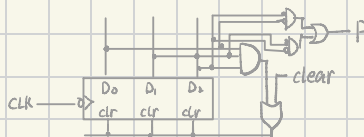
0	1	2	3	4	5
1	1	1	1	0	0

$(5)_{10} = (101)_2$ 即 D_2, D_0



eg. 循环输出 1110000 1110000... 0~6计数器 + 36it 输入 16it 输出组合电路

真值表 $\rightarrow$ 卡诺图 $\rightarrow F = \overline{D_3} \overline{D_0} + \overline{D_3} \overline{D_1}$

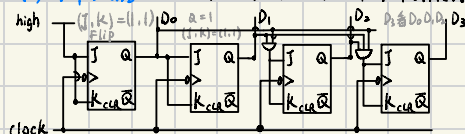


同步式计数器

低位全为1时,下一个时钟周期翻转到1

延迟小,无短脉冲问题

每一个JKFF接受同一个CLK



非 $2^n - 1$ 计数器

$M-1$ 的下一个时钟周期使全部 $(j, k) = (0, 1)$, 置 0。

e.g. 0~9计数器:

9 10 实际 $\rightarrow$ 即仅修改 D_3, D_1 即可。

到9后, 下次使 $(j, k)_{D_2} = (1, 1)$, $(j, k)_{D_1} = (0, 0)$

橙对9: $D_3 \cdot D_0$

对于 D_1 : $D_3 D_0 = 11$, hold. $D_3 D_0 = 01$, flip $\Rightarrow D_1\text{-flip} = D_0 \cdot \overline{D_3}$

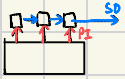
$$\text{Zf } D_3: D_2 D_1 D_0 = 111 / D_3 D_0 = 11, \text{ flip} \Rightarrow D_3\text{-flip} = D_2 D_1 D_0 + D_3 D_0$$

同步式倒序计数器 ① 下位全0, flip ② 可切换

对于非 2^n-1 进:

移位寄存器 相当于延迟输出

并入串出 Parallel Input Serial Output

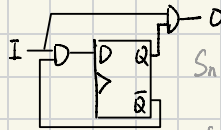


状态迁移图 Mealy 图 $\longleftrightarrow$ 时序回路

A/B $\rightarrow$ A: 输入, B: 输出

解析 回路 $\rightarrow$ Mealy 图

1. n 个触发器 $\rightarrow 2^n$ 个状态
2. 写每一个触发器的下一状态表达式, 输出表达式
3. 状态迁移表
4. 画图描



用Q来表示S(状态, 对于每一个FF)

$$S_n = Q_n = D = I \cdot \bar{Q} = \bar{S} \cdot I, 0 = I \cdot Q = S \cdot I$$

S	I	S _n	Q
0	0	0	0
0	1	1	0
1	0	0	1
1	1	0	1

连续2个1输入后, 交替输出0, 1