EXAM 1

E1:

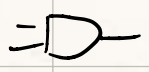
正 P switch/MOS  正常通 加电压不通

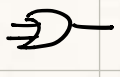
负 N switch/MOS  正常不通 加电压通

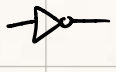
SD不通叫floating

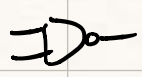
E2:

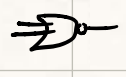
AND 与 | OR 或 | NOT 非 | NAND (not and) 与非 | NOR (not or) 或非


ab


a+b


a'


(ab)'


(a+b)'

XOR (exclusive OR) 异或

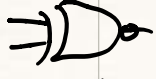
独有


a⊕b

异为1，同为0

ab	f
00	0
01	1
10	1
11	0

XNOR (exclusive not OR) 同或


a⊙b

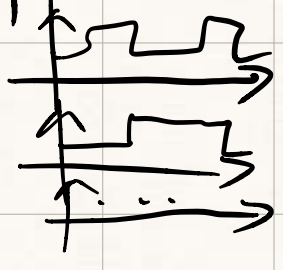
同为1，异为0

ab	f
00	1
01	0
10	0
11	1

真值表 True table

xy	S
00	0
01	0
10	0
11	1

时序图 Timing diagrams



combinational circuit 组合逻辑电路

输出只与输入有关

sequential circuit 时序逻辑电路

输出与前时间状态、有关

product term 乘积项 ab

sum-of-products 乘积和 ab+cd, ab

Sum of minterms 最小项求和

y=ab+a' 拆成最小项和

=ab+a'b+a'b'

minterms numbers 最小项编号

ab'c = 101 为真，则 m_5

a'b'c' = 000 为真，则 m_0

$f(a,b,c)=a'b'c'+ab'c+abc=m_0+m_5+m_7$

$=\sum(0,5,7)$

Sum of product 和积式

$A'B+AB'$

外和 内积

product of sum 积和式

外积 内和

$(A+B)(A'+B')$

Sigma最小项之和缩写，用 sum of product 外和内积

如 $Out = \sum(1,3,5)$

$= m_1 + m_3 + m_5 = A'B'C + A'BC + AB'C$

pi 零项之积缩写，用 product of sum 外积内和

如 $Out = \prod(0,2,4)$

$= (A+B+C)(A+B'+C)(A'+B+C)$

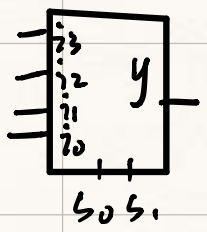
000 且 010 且 100 时

Out才为零，条件并列，以AND积组成

DeMorgan's Law 德摩根定律 折磨写

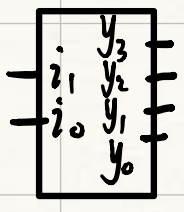
$(a+b)' = a'b'$ $(ab)' = a'+b'$

E3: 多路选择器 multiplexor (Mus) 缩写



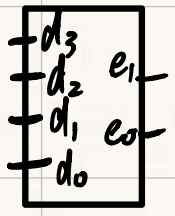
S_1	S_0	y
0	0	i_0
0	1	i_1
1	0	i_2
1	1	i_3

(解)译码器 Decoders



i_1	i_0	y_3	y_2	y_1	y_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	0	0	0	0

编码器 Encoders



d_3	d_2	d_1	d_0	e_1	e_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

*优先编码器 priority encoders

$d_3 > d_2 > d_1 > d_0$

有$d_3 d_2 d_1 d_0$中同时1时大的编码

☆

编码 $d_3 d_2 d_1 d_0 = 0000$ 是 unknown

因为必有一个d是1

d_3	d_2	d_1	d_0	e_1	e_0
0	0	0	1	0	0
0	0	1	0	0	1
0	0	1	1	0	1
0	1	0	0	1	0
0	1	0	1	1	0
0	1	1	0	1	0
0	1	1	1	1	0
⋮				⋮	

香农展开 Shannon's expansion

$$f(x,y,z) = x' \cdot f(0,y,z) + x \cdot f(1,y,z)$$

$$= x' \cdot [y' f(0,0,z) + y f(0,1,z)] + \dots$$

例 $f(x,y,z) = yz + xyz' + x'y'z$

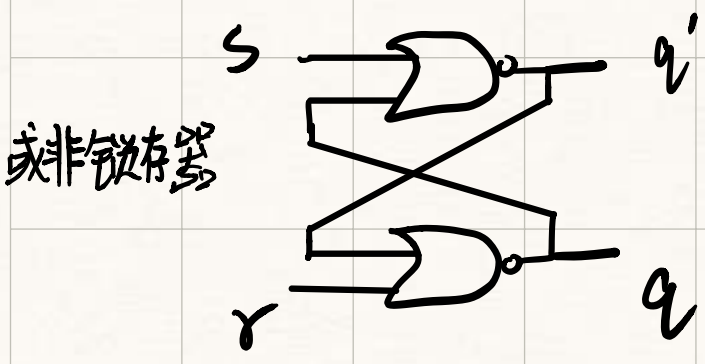
$$= x f(1,y,z) + x' \cdot f(0,y,z)$$

$$= \dots \; x(yz + yz') + x'(yz + y'z)$$

使用香农展开可抽取变量

#EXAM 2

E4: SR锁存器 SR Latch



或非锁存器

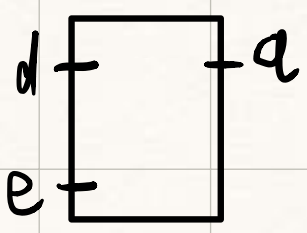
'set' 'reset'

s	r	q
0	0	Previously-stored bit 保持
0	1	0 ("Reset") 清0
1	0	1 ("Set") 置1
1	1	Unknown 振荡

为了解决RS锁存器带来的问题（RS不能同时为1），在此基础上，添加两个与门和一个非门，即可避免这种情况。升级版电路名字就叫D锁存器。

但是D锁存器同样存在它的问题，那就是无法去除输入的毛刺（换句话说，对毛刺很敏感）。可以看到当E端为0的时候，R端也会恒为0，S端则等于D端输入，亦即是此时输出直接等于输入。所以在E=0的时候，输出完全跟随输入（哪怕输入存在毛刺/抖动，这在电路中十分常见！！！）

D锁存器 D Latch

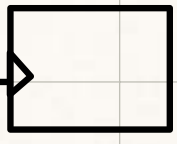


'enable'

e	d	q
0	0	Previously-stored bit (d is ignored) } 保持不收d
0	1	Previously-stored bit (d is ignored)
1	0	0 (d is stored) 0
1	1	1 (d is stored) 1

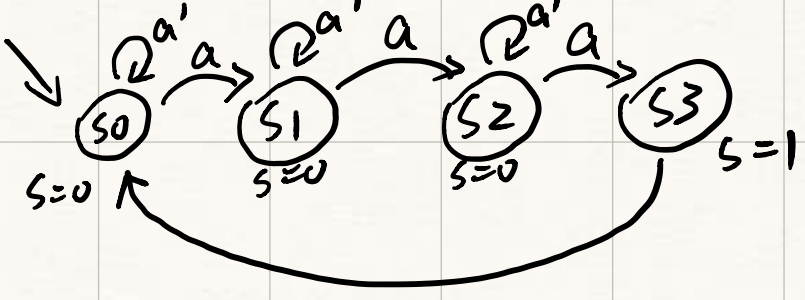
时钟 clock

上升沿 rising edge



状态机FSM (Finite state Machine)

由寄存器和逻辑组成，按照控制信号预先设置的状态，进行转移的数学模型



可由 状态图 写 状态表 (时序)

Q^* 次态 (下一次) next　　Q 现态 present

	Q	a	Q^*
S_0	00	0	00
	00	1	01
S_1	01	0	01
	01	1	10
S_2	10	0	10
	10	1	11
S_3	11	0	00
	11	1	00

可再写出公式 $Q_1^* =$ ...

$Q_2^* =$ ...

组合逻辑电路 combinational circuit

时序逻辑电路 sequential circuit

E5: 补码 complement

原码"1,0"取反,值加1

符号数 signed number { 0打头作正数 / 1打头作负数

Overflow 溢出

指正加正或负加负超出进位数

对4位符号数(-8~7)

0010 + 0011 = 5 ok

2 3

0111 + 0001 = 1000 overflow

7 1 -8

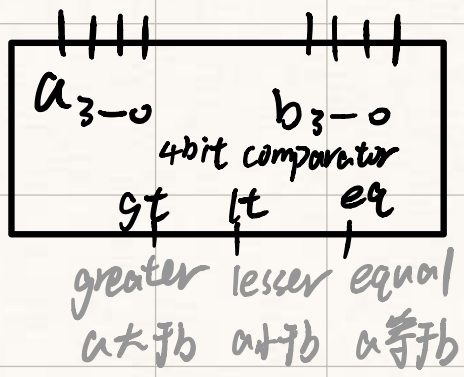
1111 + 1111 = (1)1110 ok

-1 -1 -2

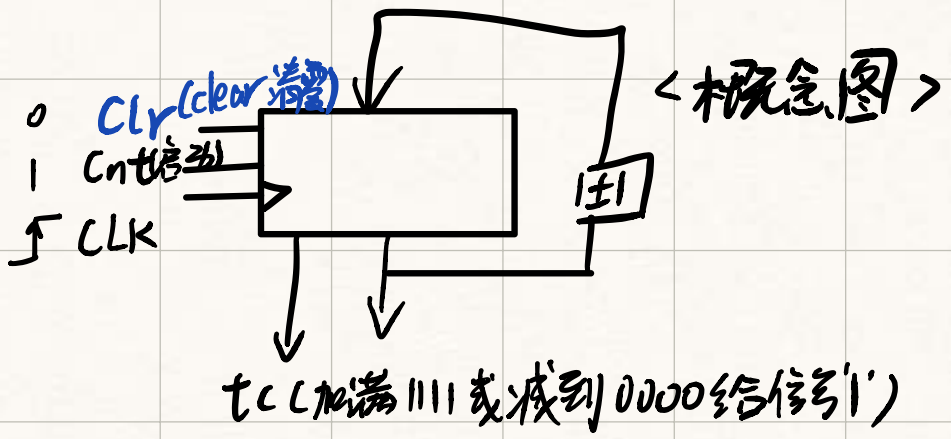
正加负一定不会溢出 1000 + 0111 = 1111 ok

-8 7 -1

E6 比较器 Comparators

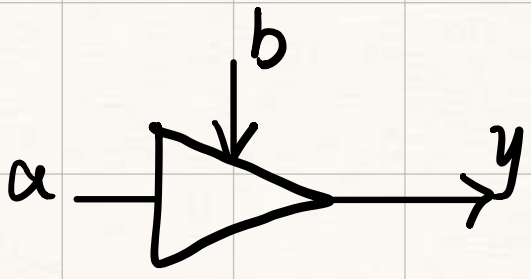


加/减 计数器 Up/down counters



tc(加满1111或减到0000给信号'1')

E7：三态门 three-state buffer



当 b=1 时通路

$y = a$

当 b=0 时断路

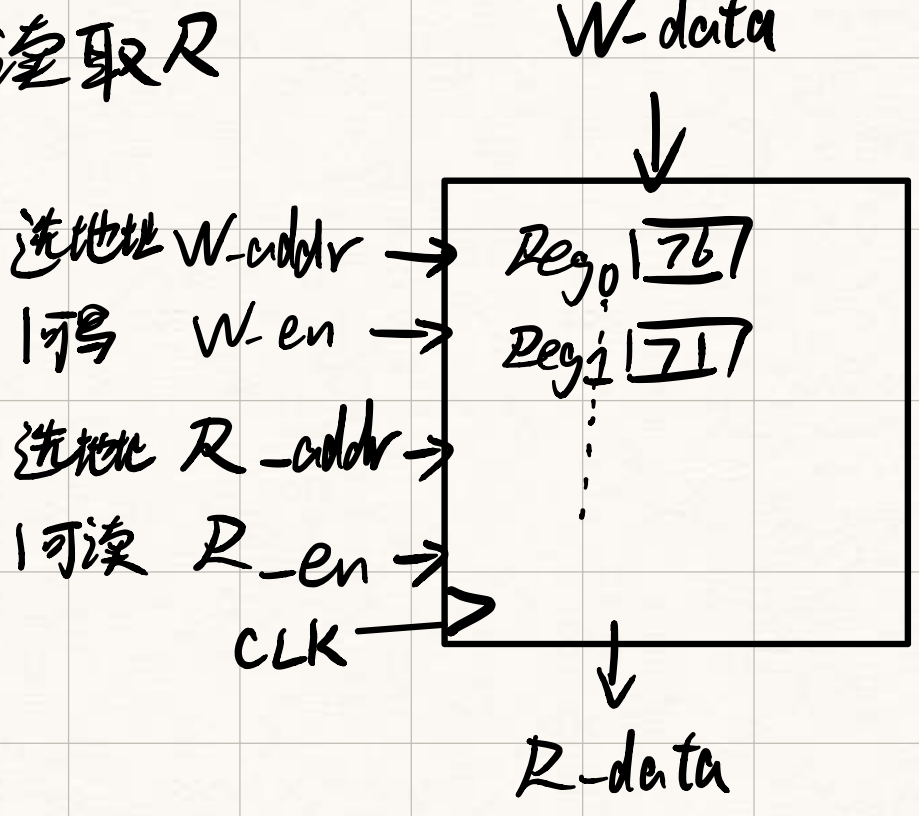
$y = z$ 高阻态、

high impedance

E8 寄存器 Register

Writing 写入 W

Reading 读取 R



EXAM #3 week8-14

- 香农展开 Shannon's expansion

$n1(0,0,x)=0$ $n0(0,0,x)=x$ $y=S1'S0$

$n1(0,1,x)=x$ $n0(0,1,x)=0$

$n1(1,0,x)=x$ $n0(1,0,x)=0$

$n1(1,1,x)=0$ $n0(1,1,x)=0$

使用香农展开可抽取变量

- 补码 complement 原码"1,0"取反,值加1得补码

符号数 signed number {0打头作正数 1打头作负数

● Overflow 溢出

指 正加正 或负加负超出进位数

对4位符号数(-8~7)

0010 + 0011 = 5 ok

2 3

0111 + 0001 = 1000 overflow

7 1 -8

1111 + 1111 = (1)1110 ok

-1 -1 -2

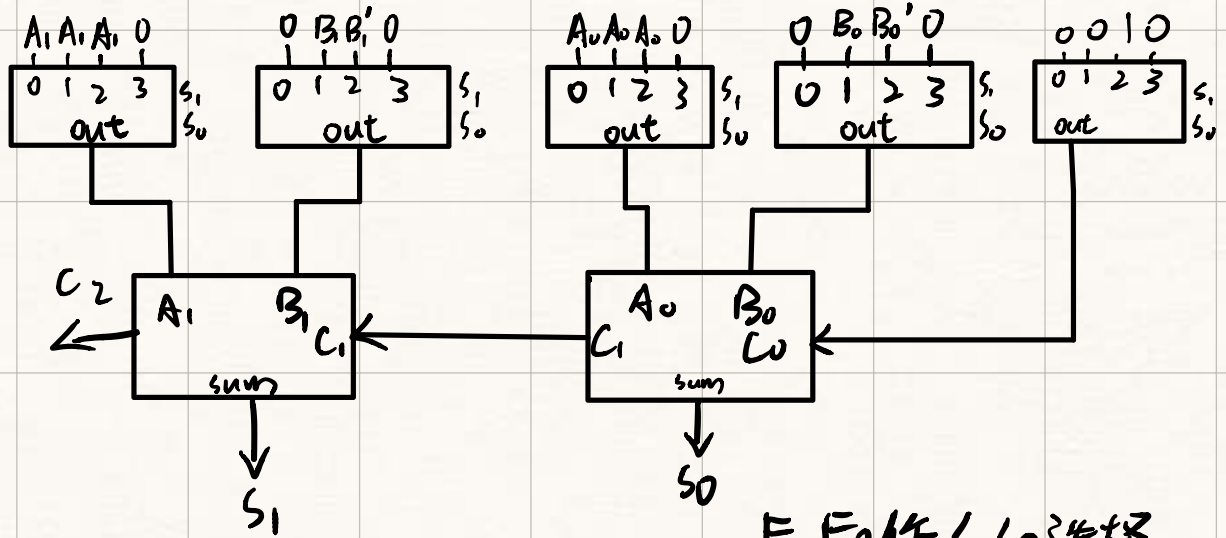
正加负一定不会溢出 1000 + 0111 = 1111 ok

-8 7 -1

● MUX实现多功能计算:ALU

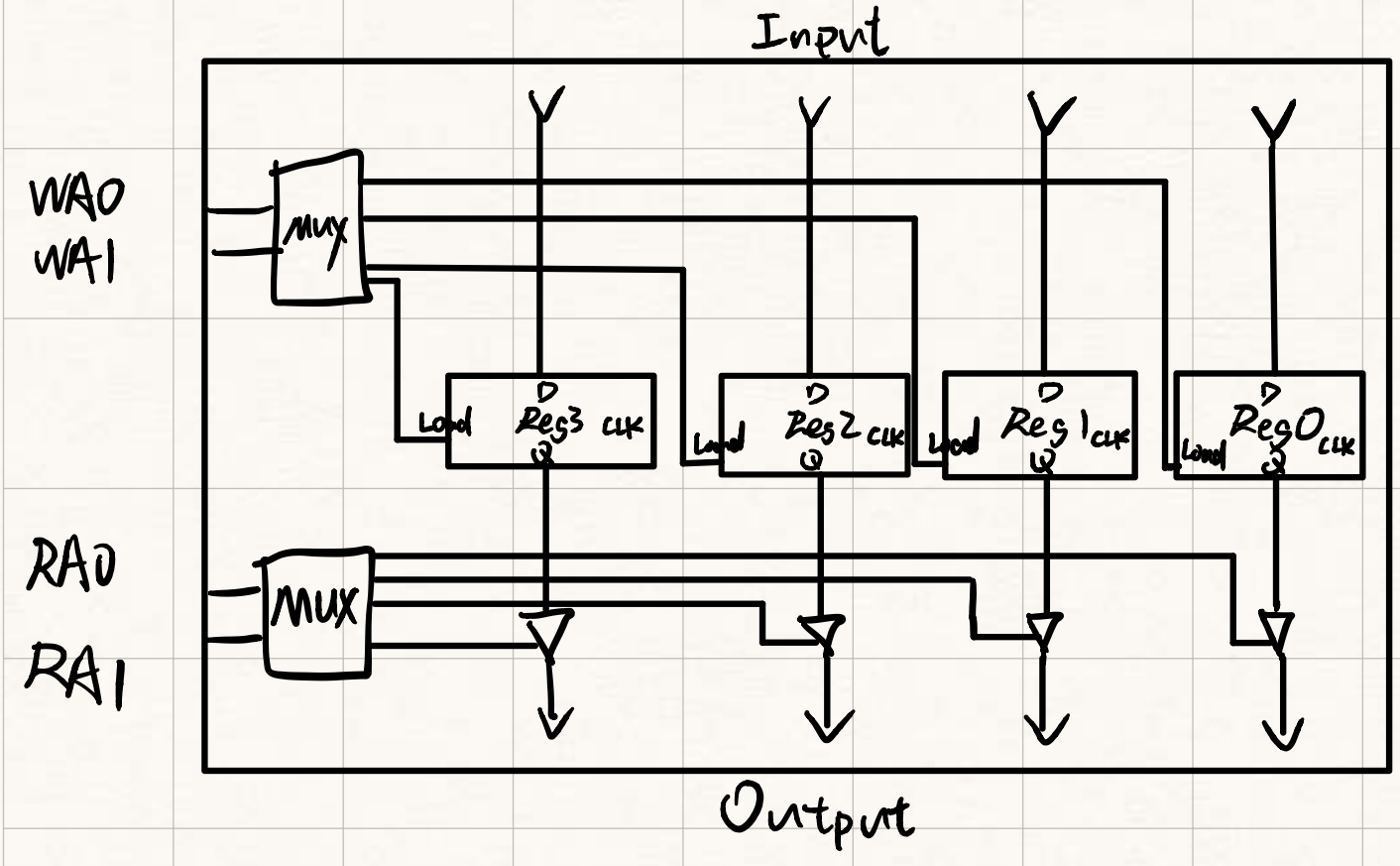
F_1	F_0	A_1A_0	B_1B_0	进位 C_0	$A_1A_0+B_1B_0+C_0$	
0	0	A_1A_0	00	0	A_1A_0	Pass A
0	1	A_1A_0	B_1B_0	0	$A_1A_0+B_1B_0$	A加B
1	0	A_1A_0	$(B_1B_0)'$	1	$A_1A_0-B_1B_0$	A减B
1	1	00	00	0	00	00

$V_0 \rightarrow V_{19}$ input



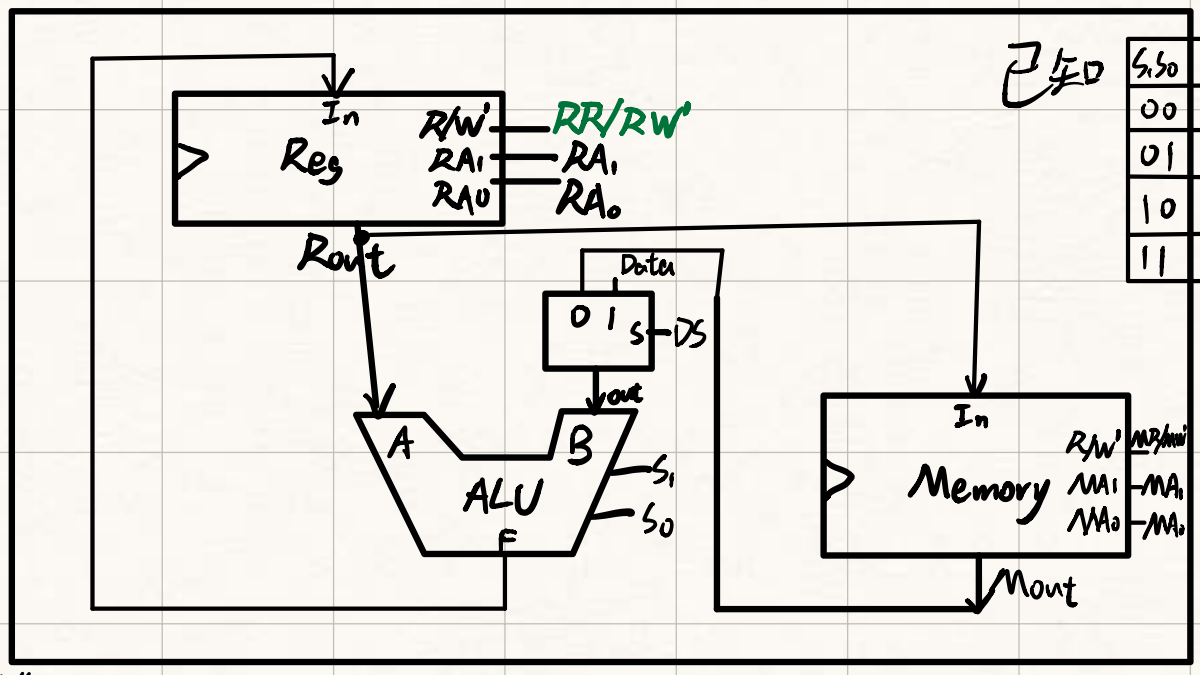
F_1F_0作S_1S_0选择

Register Files 4位寄存器



Write写 Read读 Input输入内容

● Controlling Register Transfers 特殊功能寄存器？



A_1A_0选址

R/W'给0 加载 In(t)，
给1 不变，
} 并输出原值 (下一个CLK)

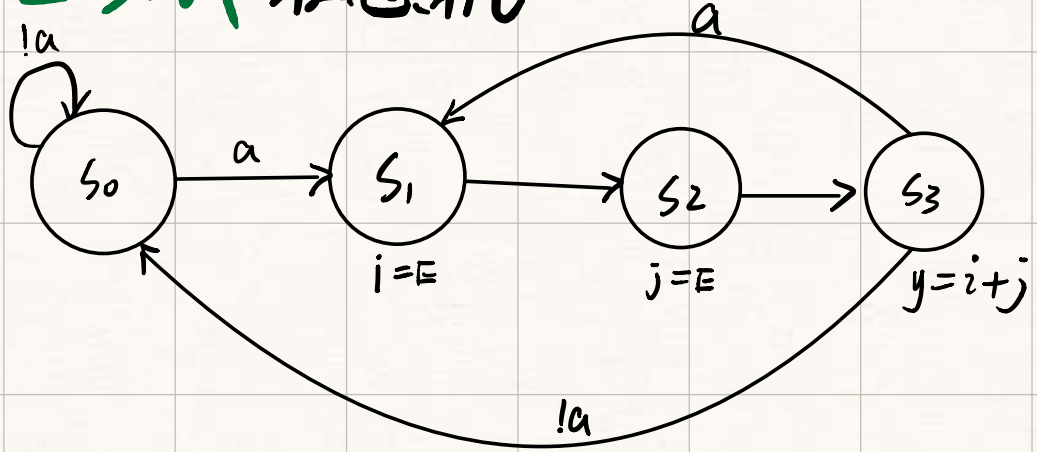
R0 < R0+M1 ：R0+M1给R0

选址 RA=00 MA=01

读写 RR/RW'=0 写读 MR/MW'=1 只读

DS=0 S1S0=00 选加功能

● HLSM 状态机



i，j 用寄存器，加载上一个状态的E值(Cinput)

y同理

id_i，id_j，id_y即对应 reg 使能端 enable

S_3 1 S_1 1 S_2 1

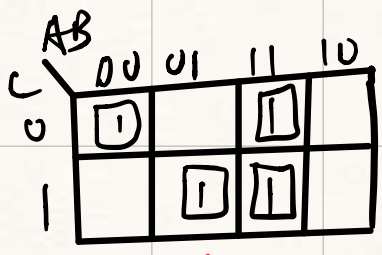
S_0

● 蕴含项那茬

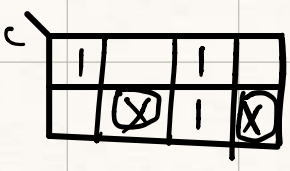
- An **implicant** of a function is a term that covers only minterms in the function's on-set
- A **prime implicant** of a function is an implicant that cannot have a literal removed without becoming a non-implicant
- A function's **minimal cover** is an expression for the function, having the fewest terms, each with the fewest literals
- An **essential prime implicant** is the only prime implicant to cover a particular minterm in a function's on-set

28

minterms 最小项：每个单元格

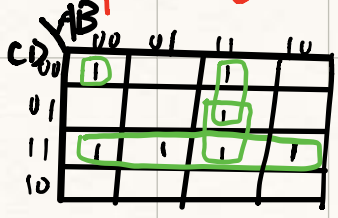


Don't cares 无关项：可有可无的"X"值



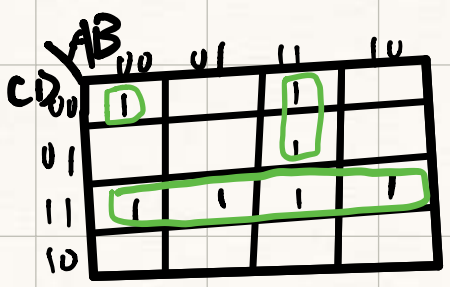
implicant 蕴含项："圈"，每种可能的圈就是一个它

prime implicant 主要，质蕴含项：不能和其他圈合并成更大的主要圈



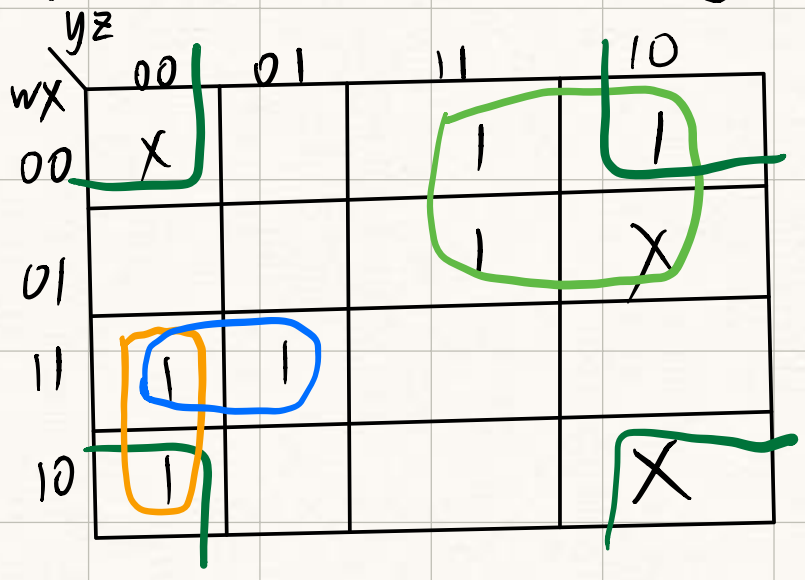
4个圈

essential prime implicant 实质本源蕴含项：至少有一个独有项的最大



3个圈

例：已知 $F(w,x,y,z)=\Sigma(2,3,7,8,12,13)$ $D(w,x,y,z)=\Sigma(0,6,10)$



Prime Implicant	Essential or not?
w'y 大圈	essential
x'z' 4角大圈	not essential 可消失
wy'z'	not essential 可消失
wxy'	essential

sum of products：最简 prime 组合 + 最少 not essential

$$F=w'y+x'z'+wxy'$$

● Moore 和 Mealy 状态机的转化

Moore 圈上才变化，等CLK

Mealy 箭头上变化，立即

Moore to Mealy

moore

Present	Next state		Output
	in=0	in=1	
S0	S5	S2	0
S1	S0	S5	1
S2	S0	S5	0
S3	S4	S3	1
S4	S4	S3	0
S5	S1	S0	1

↓

Mealy

Present	in=0		in=1	
	Next	Output	Next	Output
S0	S5	1	S2	0
S1	S0	0	S5	1
S2	S0	0	S5	1
S3	S4	0	S3	1
S4	S4	0	S3	1
S5	S1	1	S0	0

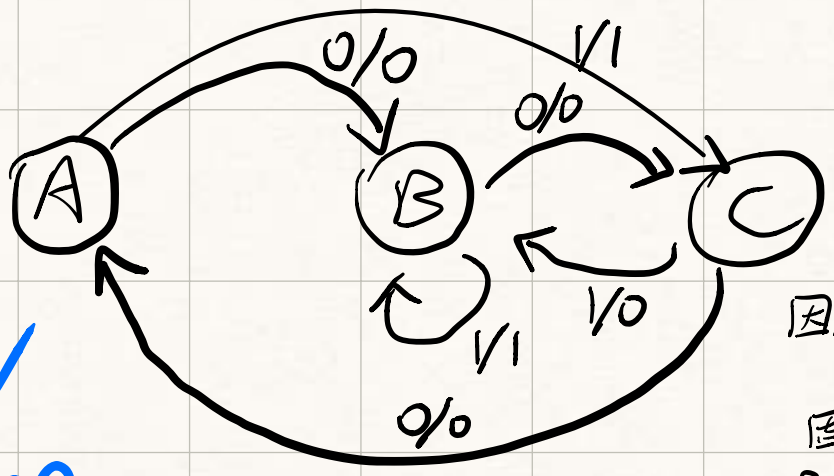
Mealy to Moore

Mealy

已知

Present	in=0		in=1	
	Next	output	Next	output
A	B	0	C	1
B	C	0	B	1
C	A	0	B	0

in/out

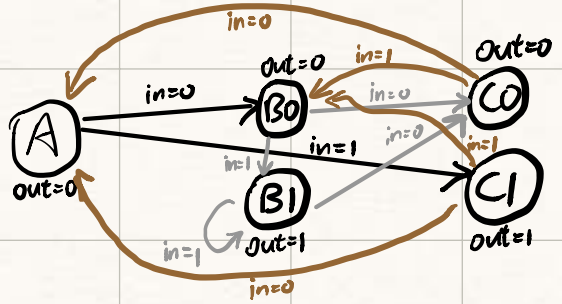


因从不同态到B/C output 不一致

因分裂为B1,B0 C1,C0态，B0表Output为零的B，同理C

Moore

Present	Next State		Output
	in=0	in=1	
A	B0	C1	0
B0	C0	B1	0
B1	C0	B1	1
C0	A	B0	0
C1	A	B0	1



● Partition 分区法?

已知

Present	x=0 Next, Output	x=1 Next, Output
S_0	S_2 0	S_1 0
S_1	S_4 1	S_1 1
S_2	S_3 1	S_2 1
S_3	S_3 0	S_0 0
S_4	S_4 0	S_0 0

合并化简：1、同样output值？

2、next state到自己或同一状态？(同次下)

3、再基于分组再化简

1、$G_1=\{S_0,S_3,S_4\}, G_2=\{S_1,S_2\}$

2、$G_1=\{S_0\}, G_2=\{S_3,S_4\}, G_3=\{S_1\}, G_4\{S_2\}$

由$S_3=S_4$合并，S_1与S_2也同next state

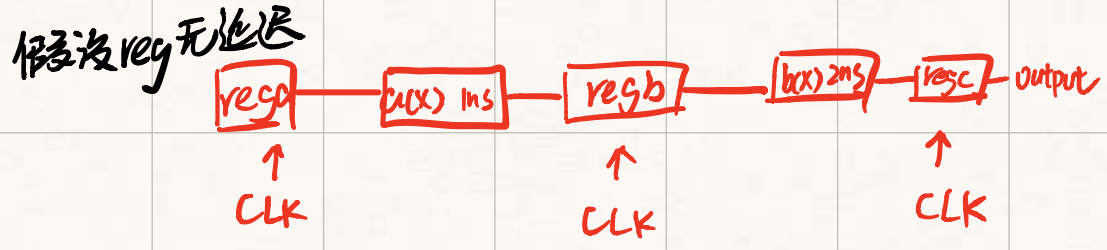
3、$G_1=\{S_0\}, G_2=\{S_1,S_2\}, G_3=\{S_3,S_4\}$

- 迟迟 in pipelining 流水线

Minimum clock period 最小时钟长度(要求)
即 ≥ 最大延时部件

Throughput 吞吐量: 新output更新时间

Latency 延迟 遍历流程所需时间



Minimum clock period : 2ns
Throughput : 2ns
Latency : 4ns 2↑CLK, → 4ns
X2个动作

- Carry Look ahead 超前进位加法器
加快加法计算

$C_1 = G_0 + P_0 C_0$

$C_2 = G_1 + P_1 C_1$

$C_3 = G_2 + P_2 \cdot C_2$

$\vdots$

$G_i = a_i \cdot b_i$ 与

$P_i = a_i + b_i$ 或