数字系统

1. 进制 Combinational 无记忆 Sequential 有记忆.

2. 逻辑门 → 组合逻辑 → 时序逻辑 → 微处理器

晶体管 → (记忆) 自锁

3. 门：输入、出；功耗；时延

或 ≥1 或非 异或 ⊕ (不一样输1).

非 x' 缓冲门(延时).

与 & 与非 =

$AB+AC+B'C=AB+B'C$ $\overline{A}\cdot\overline{B}=\overline{A+B}$ $AB+\overline{A}B=B$

$(A+B)(A+C)(B'+C)$ (分配律).

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

=

$\overline{A}+\overline{B}=\overline{AB}$ $(A+B)(A+C)=A+BC$

4. 微处理器，FPGAs，ASICs.

软件编 硬件编

5. 时钟 → 速率 → 效率.

6. 真值表求表达式：[电路设计] 反过来：电路分析.

最小项为1 for 求和. SOP. (电路图 → 真值表)

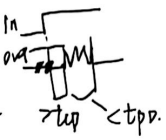
输出为1的，把输入画出那个指到1，相加.

7. 延时 propagation 传输延时 t_{pd}. 看输出

contamination 污染延时 t_{cd} 看输入.

经过 t_{pd}，保证输出稳定（看最长的线路）

输入在 t_{cd} 内绝对不变（最短线路）下限.



8. 表达式化成最简：卡诺图化简 00, 01, 11, 10.

① 画包含最多且为2^n的"1"的圈 矩形（首尾相连）

② 每画一个新圈，必须包含至少一个"1".

③ 所有"1"均被圈到.

9. 最小项 m eg. ABC 为1时有效 (都101时). $M+m=全集$

最大项 M eg. $\overline{A}+B+\overline{C}$ 为0时有效

⇒ SOP = Σ最小项 eg. $F=xyz+xy\overline{z}+y\overline{z}=(\cdots+\overline{x}y\overline{z}+xy\overline{z})$

$=\Sigma(3,5,6,7)$

POS = Π最大项 $=\Pi(1,2,4,0)(\overline{x}+y+\overline{z})(\cdots)\cdots$

有X(可0可1)：哪个好画圈.

化1不如化可以化0.

0. 模块(组合逻辑)

① Multiplexer 数据选择器 (n-1). X-to-1 Mux.

S=0 y=d0

S=1 y=d1

$Y=d_0\overline{S}+d_1S$.

$Y=\overline{S_0}\,\overline{S_1}\,d_0+\cdots$

② decoder 解码 $N\to 2^N$

③ Tri-state buffer 三态门.

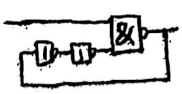
④ Addition 加法器 sub: $a-b=a+\overline{b}+1$.

编码器，译码器，比较器，加法器. 竞争冒险. 触发器

$2^N\to 1$

11. 门延时 → Glitches 毛刺 → Hazard 冒险.

检：可产生时钟信号（晶振）



不好：Static-0 Hazard 有 $(A+\cdots)(\overline{A}+\cdots)$

Static-1 hazard 有 $A\cdots+\overline{A}\cdots$

· 消除毛刺 ① Using K-map：(加冗余项)

画m个圈相邻有一条公共边 → 有毛 $(X+\overline{X})$. Solution → 画包括公共边的圈.

$y=\overline{x}d_0+xd_1=\overline{x}d_0+xd_1+d_0d_1$ 强行为1.

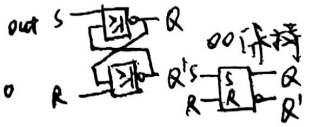
② 电容.

Verilog ① 结构型

② 数据流型. $y=(s'\&d_0)|(s\&d_1)$ 赋值

③ 行为级描述.

锁存器 latch (电平触发) S-R 锁存器

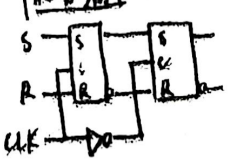


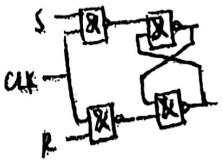
S'-R' 锁存器

D 锁存器 (边沿触发)

11 保持

触发器 flip-flop





CLK=1, $\overline{S}$, $\overline{R}$

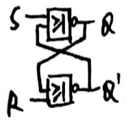
CLK=0, 1, 1 保

主从或 S-R 触发器

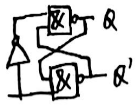
(有 CLK 控制从锁存变成了触发)

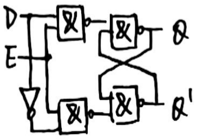
同为D触发器 SR 间断门.

锁存器 latch 电平触发

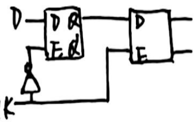
SR   00保

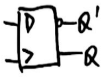
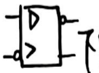
S'R'  11保

D  

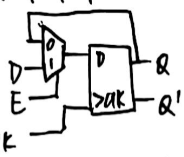
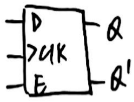
D latch with enable (Q=D)  E=0 保 E1 使能

触发器 flip-flop D-F-F 边沿触发

D.  CLK=0, 过第一个; =1, 过二个.

⇒  上升沿触发  下降沿. 补丁: violation 违规.

D flip-flop with enable

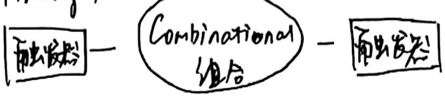
 

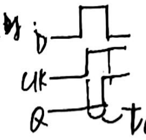
Clear: 异步清0 (与 CLK 无关)

异步计数器 分频系数 $\div 2^N$.

寄存器 register.

Timing.



触发器延时:  t_{cq} { C 污染 P 传输

 稳定区间

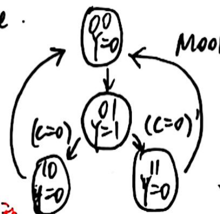
1. 第二个时钟上升沿不能来太快 (Comb 延时不太慢).
$t_{pcq} + t_{comb\text{-}pd} + t_{setup} \le T$. ← 降频.

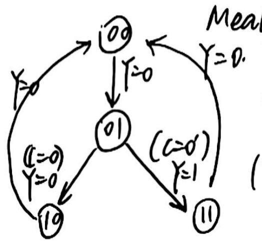
2. 第一个时钟上升沿不能来太慢 $+t_{skew}$ (T_1T_2 相差)(ΔT) $+2t_{jitter}$
$t_{hold} < t_{ccq} + t_{comb\text{-}cd}$

有限状态机 finite state machine.

状态图 state diagram

{ Moore: Y 仅与状态有关
{ Mealy (快, 不等时钟).





输入/输出. I/O

(状态数量少. 更小电路面积).

电路图 → Q_{next} 与 Q 关系 → Y=f(Q) 输出方程 → 绘制状态图 → 不易影响其他. 安全.
状态方程, 状态表 → 绘制状态图

Verilog code for FSM.

一段式
两段式 { 同步时序形式 always @(posedge clk) begin negedge clk
{ 组合电路形式 always begin @(*)

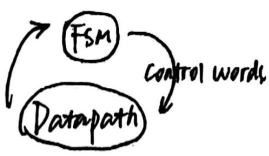
三段式: ① current → next_state { 非阻塞赋值 <=
② 对每状态输出
③ what's next_state { 阻塞 =

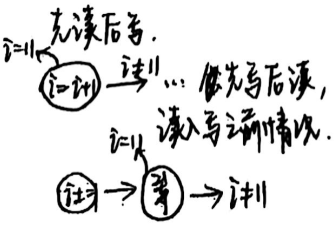
伪代码(功能) → datapath → control words → state diagram → output → circuit.

要考虑在状态图中不出现的状态 ($2^n - x$) 回到原始状态.

对 datapath, 有几个未知数就有几个 reg.

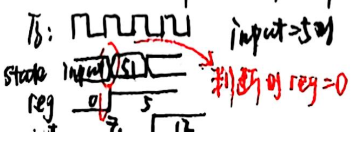
组合电路没有延时. 但是回到寄存器要等: 读不等, 写等 CLK

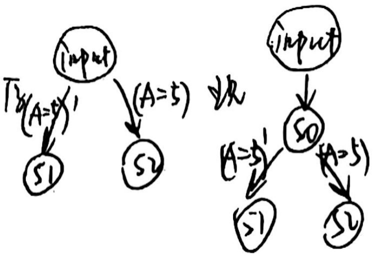




读 (写入 reg) 判断 写

若在 reg 后有一个判断条件:





Verilog

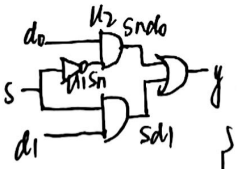
module 名 (结构型

input s, d0, d1, input [3:0] 4bit

output y 默认1bit

);

wire sn, sndo, sd1;



not u1 (sn, s); sn=~s

实例名. 门名称 顺序根据module里面in, out来.

and u2 (sndo, d0, s1);

and u3 (sd1, s, d1);

语句间并行.

...

endmodule

assign y=~a; 数据流型.

derive 推导.

equation 方程式.

{ unsigned 无符号
  singned 带符号

minus 减

implement 实现

xor 异或

nor 或非.

wire [31:0].

reg. 在always里面线

123='d123. 10进制 'h16进, 'o8进, 'b二进.

16'd5 (16位)

---

vivado 流程.

画 datapath

源文件 (design sources). + 仿真 → run sim.

↓

综合 run synthesis

↓

open Synthesized design

↓

约束文件 { 管脚 LVCMOS33 / 引脚 } → save constraints

↓

综合 Synthesis

↓

实现 run implementation

↓

Bitstream

↓

硬件.