数字逻辑复习笔记

Chapter 1 二进制

控制复杂性的艺术

"3Y" 原则

> **Hierarchy (层次化)**

把系统划分为若干模块，然后进一步划分每个模块直到这些模块可以很容易理解

> **Modularity（模块化）**

所有模块有定义好的功能和接口，以便于他们之间可以很容易地相互连接而不会产生意想不到的副作用

> **Regularity（规整化）**

在模块之间寻求一致，通用模块可以重复使用多次，以便减少设计不同模块的数量

数制

十进制、二进制、八进制和十六进制的转换

二进制加减法

数字抽象

噪声容限

> NM_L = V_IL - V_OL NM_H = V_OH - V_IH

静态约束 static discipline

牺牲使用任意模拟电路元件的自由来换取数字电路的简单可靠 同一个逻辑系列(logic family)的元器件遵循相同的静态约束，逻辑门之间保持兼容的逻辑电平

Chapter 2 组合逻辑设计

组合电路的概念

- 每一个电路元件本身就是组合电路
- 每一个**电路结点**或者是**一个电路**的输入，或者是连接到外部电路的一个输出端
- 电路不包含回路；经过电路的每条路径最多只能经过每个电路结点一次

布尔表达式

与或式（SOP,sum of products）（先与后或）、最小项（minterm）

对应真值表为TRUE的行; Eg：AB+AB'

或与式（POS,products of sums）（先或后与）、最大项(maxterm)

对应真值表为FALSE的行; Eg：(A+B)(A+B')

布尔代数

布尔代数公理（Axioms）

对偶式

单变量定理

> **T1 同一性定理**
>
> **T2 零元定理**
>
> **T3 重叠定理**
>
> **T4 回旋定理**
>
> **T5 互补定理**

多变量定理

> **交换律**
>
> **结合律**
>
> **分配律**
>
> **吸收律**
>
> **合并律**
>
> **一致律**
>
> ***德摩根定理***
>
> 所有项的乘积的补等于每个项的补的和，对偶表达为：所有项的和的补等于每个项的补的乘积 吸收、合并和一致律，允许消除冗余变量

由德摩根定理到'推气泡'法

把门电路中的逆变器看作'气泡'，从输入端推气泡到输出端，或者把气泡从输出端推回输出端，同时与门和或门相互转换，逻辑等价

展开蕴含项 AB=AB(C+C')

4输入优先级电路（priority circuit）

X（contention） 与 Z（floating）

非法值 X

X 在**电路中**表示出现非法的或者未知的值，会在比如一个节点同时被1和0驱动的时候出现，这种情况称为"竞争"（contention）是必须避免的错误。

需要注意的是，**电路**中的X和**真值表**中的 X 代表不同的含义，不能混淆。后者指的是在真值表中的**无关项**，表示不重要的值。

浮空值 Z

浮空值Z代表结点既没有被低电平也没有被高电平驱动，他可能是 0 也可能是 1 ，也可能 0 和 1 之间的电压，取决于系统先前的状态。浮空不代表错误，但是很可能是因为输入端忘记接入电压，可能会引发错误。

应用：旧的计算机系统中，不同芯片经常被连接到共享总线（sharebus）上使用，通过三态缓冲器（tristates）与共享存储器总线进行连接。在某时刻只允许一个芯片的使能信号有效，其他芯片的输出必须为浮空。不过现代计算机用的是点到点链路直接连接而不是使用共享总线了。

卡诺图（Karnaugh Map）

卡诺图通过格雷码的顺序把各个真值组合放在一张表里，通过把相邻的1方格圈起来化简。 Karnaugh Map 化简逻辑：

- 用最少、而且最大的圈来圈出所有的1
- 每个圈中的方格都必须有1
- 必须是矩形，边长是2^n（1，2，4等）
- 卡诺图是没有边缘的，可以上下相通，左右相通
- 同一个1可以多次被圈出
- 如果算上 X，可以圈出更多1，那 X 也可以被圈起来

小结

逻辑代数或者卡诺图这两种逻辑化简方法。logic synthesizer（逻辑综合）干的事其实就是化简电路。 ##组合逻辑模块

Multiplexer 复用器（2.8.1）

一种最常用的组合逻辑电路，根据选择（select）信号的值，从多个输入中选择一个作为输出。复用器又称mux。

复用器可以用与或门的两级逻辑设计，也可以用三台缓冲器构成。

8：1复用器可以转换成4：1乃至2：1复用器，只需要重新设计真值表让某些输出从0/1转变为由某个值的输入。

译码器d ecoder（2.8.2）

N个输入和2^N个输出，每个输出取决于输入的组合。输出是独热（one-hot），给定时间恰好只有一个输出为高电平。

译码器会被应用到只读存储器（ROM）。

时序

上升沿

从低电平到高电平被称为上升沿。同样的有下降沿。timing diagram（时序图）描绘了电路的transient response（瞬间响应）。

传播延迟和最小延迟

propagation delay（传播延迟tpd）

当输入改变直到一个或者多个输出达到他们的最终值所经历的最长时间，对应者critical path

contamination delay（最小延迟tcd）

当一个输入发生变化直到任何一个输出开始改变的最短时间，对应者short path 组合电路的tpd是critical path上所有原件的tpd之和,tcd也是所有元件tcd之和。

控制关键路径和数据关键路径

(不考，其实就是critical path是从输入信号到输出信号还是控制信号到输出信号的差别)

glitch（毛刺）或者hazard（冒险）

毛刺通常不会导致什么问题。产生原因：用卡诺图表示理解就是圈到圈之间没有扣在一起

Chapter 3 时序逻辑设计

introduction

Outputs of sequential logic depend on current and prior input values – it has memory.

State elements store state

- Bistable circuit
- SR Latch
- D Latch
- D Flip-flop

Latches and Flip-Flops

Bistable Circuit

可以存1位信息 但是没啥用

SR (Set/Reset) Latch（3.2.1）

- S = 1, R = 0:

> then Q = 1 and Q = 0
>
> Set the output

- S = 0, R = 1:

> then Q = 0 and Q = 1
>
> Reset the output

- S = 0, R = 0:

> then Q = Qprev
>
> Memory!

- q – S = 1, R = 1:

> then Q = 0, Q' = 0
>
> Invalid State Q' ≠ NOT Q

D Latch（3.2.2）

D锁存器避免了S和R同时有效造成的奇怪情况

- Two inputs: CLK, D CLK: controls when the output changes

D (the data input): controls what the output changes to

When CLK = 1, D passes through to Q (transparent)

D Flip-Flop（3.2.3）

D触发器由两个背靠背的D锁存器构成，第一个L1是主锁存器，L2次锁存器。 CLK先0后为1的时候，L1和L2轮流阻塞（期间D到N1,再从N1到Q）。因此只有在clock-edge（时钟沿）,D才能被复制到Q。

tips:电路符号中的三角形代表时钟输入，当不需要输出的时候经常被省略。

Register 寄存器

一个N位寄存器由共享一个CLK的N个触发器构成，这样寄存器的所有位将被同时更新。寄存器是大多数时序电路的关键组件。

Enabled Flip-Flops 带使能端的触发器

- Inputs: CLK, D, EN 增加一个EN输入，确定在时钟沿是否载入数据。

tips：具体原理可以使用Mux把EN作为一个输入，也可以把CLK和EN作为一个新的AND门的输入，但是后者可能会造成时钟延迟，在时钟上执行逻辑不是一个好主意，一般不推荐使用。

Resettable Flip-Flops

Two types:

- Synchronous(同步): resets at the clock edge only 只需要把R和D作为一个新的AND门的输入，输出到D Flip-Flops
- Asynchronous（异步）: resets immediately when Reset = 1 需要改变Flip-Flops的内部结构。

Synchronous Sequential Logic Design

- Breaks cyclic paths by inserting registers
- Registers contain state of the system
- State changes at clock edge: system synchronized to the clock

Two common synchronous sequential circuits

- Finite State Machines (FSMs)
- Pipelines

Finite State Machine（FSM）有限状态机

Consists of state register and combinational.

Two types of finite state machines differ in output logic:

- Moore FSM: outputs depend only on current state,outputs labeled in each state
- Mealy FSM: outputs depend on current state and inputs,arcs indicate input/output

factoring FSM

FSM Design Procedure

1. Identify inputs and outputs
2. Sketch state transition diagram
3. Write state transition table
4. Select state encodings
5. For Moore machine:
6. Rewrite state transition table with state encodings
7. Write output table
8. For a Mealy machine:
9. Rewrite combined state transition and output table with state encodings
10. Write Boolean equations for next state and output logic
11. Sketch the circuit schematic

逆向工程：从电路图到状态转移图

Timing

- Flip-flop samples D at clock edge
- D must be stable when sampled
- Similar to a photograph, D must be stable around clock edge
- If not, metastability can occur

Dynamic discipline

- Setup time（建立时间）: t setup = time before clock edge data must be stable (i.e. not changing)
- Hold time（保持时间）: thold = time after clock edge data must be stable
- Aperture time（孔径时间）: ta = time around clock edge data must be stable (ta = tsetup + thold)
- Propagation delay: tpcq = time after clock edge that the output Q is guaranteed to be stable (i.e., to stop changing)
- Contamination delay: tccq = time after clock edge that Q might be unstable (i.e., start changing)

Setup Time Constraint

- Tc ≥ tpcq + tpd + tsetup
- tpd ≤ Tc – (tpcq + tsetup)
- (tpcq + tsetup): sequencing overhead

Hold Time Constraint

- thold ≤ tccq + tcd
- tcd ≥ thold - tccq

如果有时钟偏移tskew，加在建立时间或者保持时间那一侧

亚稳态、同步器分辨时间略（不会考吧？？没印象学过）

Parallelism

- Token（任务）: Group of inputs processed to produce group of outputs
- Latency（延迟）: Time for one token to pass from start to end
- Throughput（吞吐量）: Number of tokens produced
- **Spatial parallelism:** Ben asks Allysa P. Hacker to help, using her own oven
- **Temporal parallelism**: While first batch is baking, he rolls the second batch, etc.

Chapter 4 硬件描述语言

```
module sillyfunction(input logic a,b,c,
                     output logic y):
    assign y = ~a & ~b & ~c|
               a & ~b & ~c|
```

```
                a & ~b &  c;
endmodule
```

2:1复用器

```
module mux2(input logic[3:0] d0,d1,
    input logic s,
    output logic[3:0] y):
    assign y = s?d1:d0;
endmodule
```

Chapter 5 数字模块

算术电路

加法

1-bit adder

half adder

full adder

multibit adders(Carry=Propagate adder,CPAs,进位传播加法器)

ripple-carry adder（行波进位加法器）slow

N个全加器串联起来，其中一级的Cout就是下一节的Cin,这是模块化和规整化的应用范例。但是当N比较大的时候，运行速度会慢下来。行波进位加法器的延迟 t_ripple = Nt_FA 直接随位数的增加而增加

Carry-lookahead adder,CLA (先行进位加法器) fast

把加法器分解成若干块，同时增加电路，当每块一有进位的时候就快速确定此块的输出进位，不需要等待进位行波通过一块内的所有加法器，而是先行通过该块。

两个信号：P（传播）、G（产生）

- Example: 4-bit blocks (G3:0 and P3:0) : G3:0 = G3 + P3 (G2 + P2 (G1 + P1G0) P3:0 = P3P2 P1P0
- Generally, Gi:j = Gi + Pi (Gi-1 + Pi-1 (Gi-2 + Pi-2Gj) Pi:j = PiPi-1 Pi-2Pj Ci = Gi:j + Pi:jCj-1

For N-bit CLA with k-bit blocks: tCLA = tpg + tpg_block + (N/k – 1)tAND_OR + ktFA

- tpg : delay to generate all Pi, Gi
- tpg_block : delay to generate all Pi:j, Gi:j

- tAND_OR : delay from Cin to Cout of final AND/OR gate in k-bit CLA block An N-bit carry-lookahead adder is generally much faster than a ripple-carry adder for N > 16

（前缀加法器）fastesr (似乎没有学？)

tPA = tpg + log2N(tpg_prefix) + tXOR

- tpg: delay to produce Pi Gi (AND or OR gate)
- tpg_prefix: delay of black prefix cell (AND-OR gate)

减法

改变减数的符号然后做加法 改变二进制补码的符号就是反转所有的位，然后加1 Y=A-B=A+B'+1

比较器

equality comparator（相等比较器）

N位相等比较器就是把N个与或门作为一个与门的输入

magnitude comparator

计算A-B的值，检查结果的符号位

Arithmetic/Logic Unit（算术逻辑单元ALU）

SLT (set if less than) :当A < B的时候Y=1,否则Y=0。

zero extend unit（零拓展单元）STL 的输出本来只有1位，通过零拓展变N位输出

shifter（移位器） and rotator（循环移位器）

- 逻辑移位器 LSL<<或者LSR>> 以0填充空位
- 算术移位器 算术左移（ASL）和逻辑左移一样，但是算术右移（ASR>>>)会把原来数据的最高位填充在新数据的最高有效位（msb）上
- 循环移位器 循环移动数字，从一端移走的位重新填充到另一端的空位上

一个N位移位器可以使用N个N位的复用器构造而成。

数制

定点数

fixed-pointnotation定点表示法

小数二进制和十进制的转换！

浮点数

和科学计数法一样，浮点数包含符号（sign），尾数（mantissa,M），基数（base,B）和阶码（exponent,E）
32位浮点数：1（符号）+8（阶码）+23（尾数） 尾数的第一位（二进制小数点的左端）总是为一，因此不需要存储它，称为隐含前导0（implicant leading 0）。

偏置阶码是原始阶码加上一个常数偏置。32位浮点数使用的是127.对于阶码7，偏置编码就是7+127=134=1000_0110【2】

时序电路模块

计数器

N位二进制计数器（binary counter）

shift register 移位寄存器

移位寄存器可以用N个触发器串联而成，一个相关的电路是并行到串行（parrallel-to-serial）转换器。 移位寄存器可以看作串行到并行转换器。

带并行加载的移位寄存器

通过load信号和lock选择

存储器阵列（memory array）

DRAM（动态随机访问存储器），SRAM（静态随机访问存储器），ROM（只读存储器）。 RAM是易失的，ROM是非易失的。

位单元（bit cell）

wordline， bitline

使用存储器阵列的逻辑

用于执行逻辑的存储阵列称为查找表（LookUp Table,LUT）

逻辑阵列

可编程逻辑阵列(PLA)

PLA以与或形式实现两级组合逻辑。 AND 阵列每一行中的点表示组成蕴含项的项。OR阵列中的点说明输出函数包含了哪些蕴含项。

ROM可以看作PLA的一种特殊情况，一个2^M字N位的ROM就是一个M×2^M×N位的PLA。

现场可编程逻辑门阵列（Field Programmable Gate Arrary,FPGA）是一个可重构 的门阵列。