数字电路简易版

代数

最小项法

任何的表达式都可以分解成最小项的**和**，最小项指所有输入（或其反）的**积**，共有2^n个最小项，如三个输入A,B,C的最小项有：

$$\begin{array}{cccc} ABC & A\bar{B}C & AB\bar{C} & A\bar{B}\bar{C} \\ \bar{A}BC & \bar{A}\bar{B}C & \bar{A}B\bar{C} & \bar{A}\bar{B}\bar{C} \end{array}$$

两者等价的定义就是两者真值表相同。在已知某元件真值表的情况下，可根据最小项法来找到一个运算式，使该运算式的真值表与之完全相同，这样，就保证了该运算式和该元件的等价

处理元件

毕竟是简易版数字电路，几个简单的就不纠结了，略去

编码器

编码器就是将n个输入分别编上对应的数码，比如8个输入端（代表8件事），3个输出端口，这3个端口的$0,1$数码组合就可分别表示2^3个输入

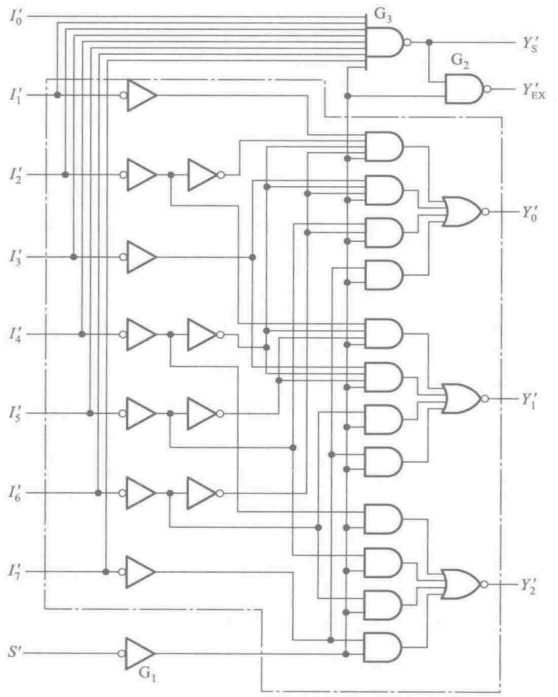
如有7个输入事件，要被3个输出端的数码编码，但不小心有好几个事件同时输入了，那输

出的数码究竟该表示哪个事件？这得引入优先级才能解决，比如对于事件$I_1 \to I_7$，输出是Y_0, Y_1, Y_2三个端口，设计如下的对应关系：

$$\begin{cases} Y_2' = ((I_4+I_5+I_6+I_7)S)' \\ Y_1' = ((I_2I_4'I_5'+I_3I_4'I_5'+I_6+I_7)S)' \\ Y_0' = ((I_1I_2'I_4'I_6'+I_3I_4'I_6'+I_5I_6'+I_7)S)' \end{cases}$$

那么I_7有最高的优先级，因为在“或”关系中，它是单独拎出来的，拥有“一票赞成权”，一旦I_7输入（$=1$），那么别的几项不用看了，或的结果一定是1，同理，I_6是优先级排第二的……

该电路参考下图（只用看虚线框中部分）



译码器

译码器做相反的事，下图中，$V_{cc}=5V$对于输入A_0, A_1, A_2而言，低电平为$0V$，取反后高电平为$3V$，下图就利用了二极管的钳制效应，当A_0, A_1, A_2呈现某一种输入方式时，输出端只有一个Y_i会变成高电平（指$3+0.7=3.7V$），十分巧妙

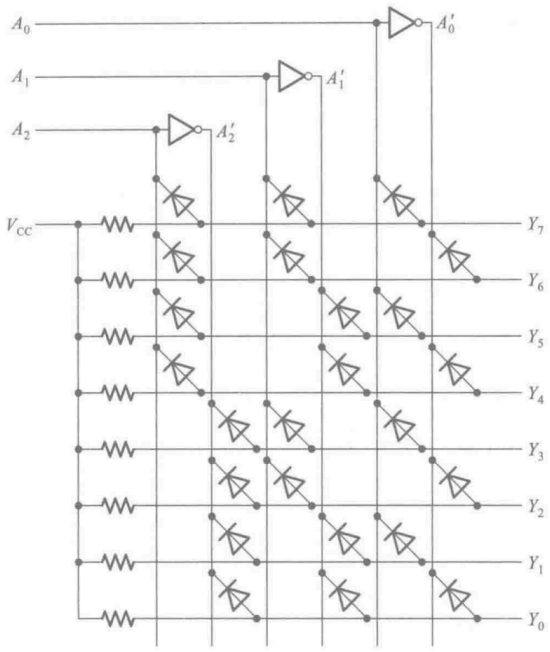
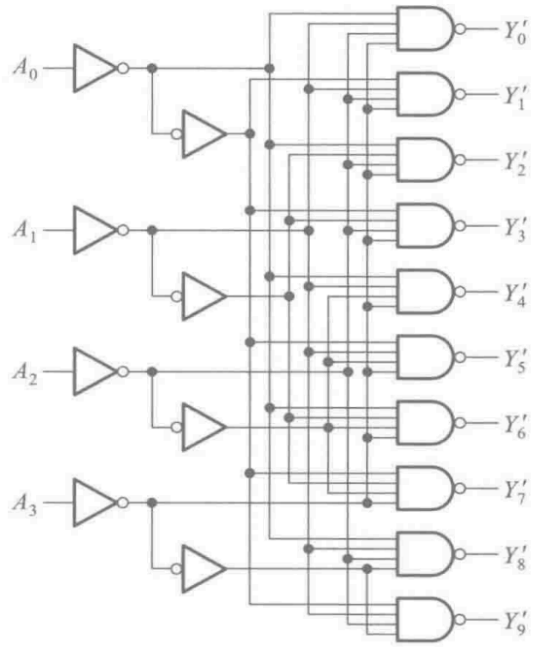


图 4.4.6　用二极管与门阵列组成的 3 线-8 线译码器

还有BCD码的译码器，神奇的是对于无效的编码如10-15（1010-1111）会自动拒绝翻译，下图实现：



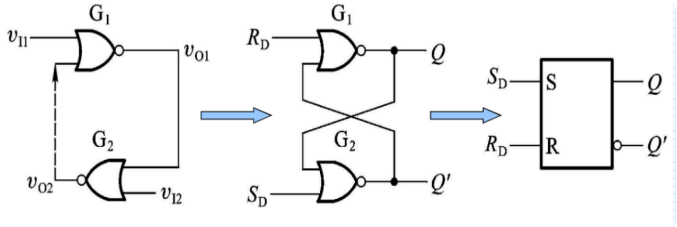
还有七段（加上小数点就是八段）的显示器，本质也与译码器一样。至于像液晶手表之类的数字显示原理稍提一句：液晶在通电时电离，生成正离子碰撞各个液晶分子，破坏了整齐排列，入射光线很少反射出来，就造成浑浊现象。不通电时就整齐排列呈透明状

选择器

给出地址（比如00-11）以及“使能状态”（0或1），输入有D0-D3共4个端口，在使能是低电平0的情况下，输出就等于地址所对应端口的输入

记忆元件

锁存器



（用与非门构建的电路同理）

1. 当$R_D = 0$，$S_D = 1$时，得到的$Q' = 0$（这一点根据的是S_D，别根据R_D判断，因为看R_D是判断不出来的，或者说是不稳定的），从而$Q = 1$，此时的状态称为1状态

2. 当$R_D = 1$，$S_D = 0$，得到的$Q = 0$（同理要根据R_D判断），从而$Q' = 1$，此时的状态称为0状态

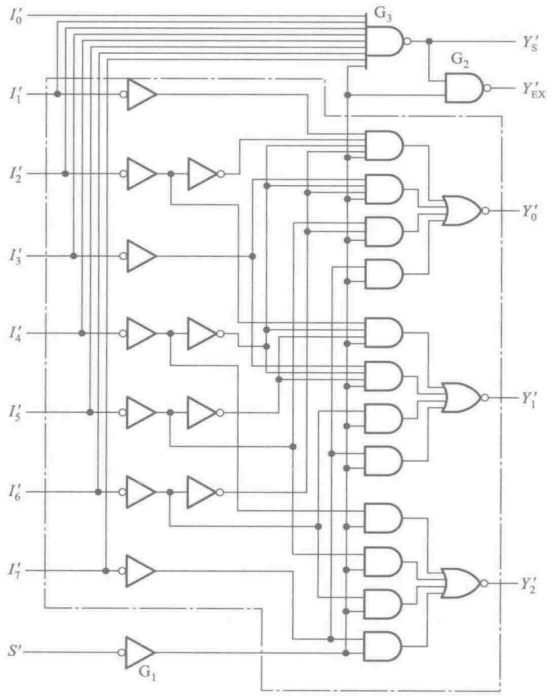
3. 当$R_D = 1$，$S_D = 0$时，原有的状态不改变

4. $R_D = 1$且$S_D = 1$的输入是违法的（因为当它们同时变回00后，不清楚锁存的状态）

S_D	R_D	Q	Q^*
0	0	0	0
0	0	1	1
1	0	0	1
1	0	1	1
0	1	0	0
0	1	1	0
1	1	0	0①
1	1	1	0①

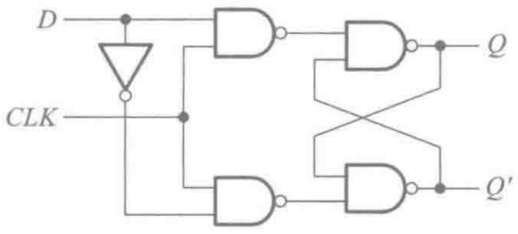
这张表中的Q可不是上文的Q，$Q*$更不是Q'！这里的$Q = 1$指的是旧状态处于1状态，而$Q*$指的则是下一刻的新状态（其含义不是高低电平）

电平触发器



在前面锁存的基础上加入了一个CLK控制，一旦CLK为0，S与R都是没用的，不会更新右侧锁存住的内容，在CLK为1的情况下才可操作，回到锁存器的情形

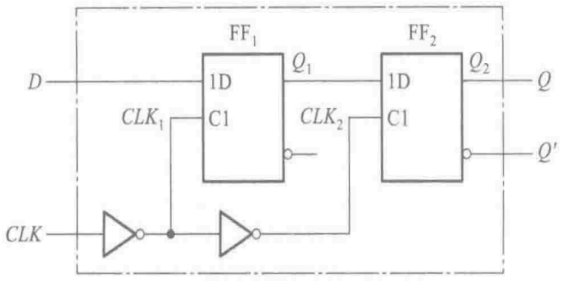
D型锁存器



这就是把上一个情况更改成单信号输入而已

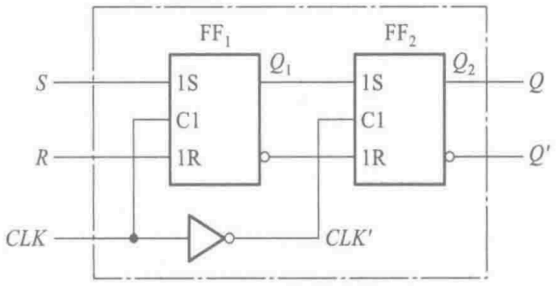
边沿触发器

edge-triggered，触发器状态的更新仅仅取决于CLK信号上升（下降）**那个瞬间**输入信号的状态，被称为上升沿（下降沿），实现电路：



图中，当CLK上升沿（从0到1时），由于CLK_1是由1到0，瞬间断电，直接把FF_1这个D型锁存器固定在上一刻的状态Q_1，今后D怎么乱变都无所谓，仍是Q_1；另外，此时CLK_2被开通，可以接受输入Q_1，这样FF_2就被状态Q_1更新，而直到下一次CLK变化来到之前，D的变化不会对Q_1造成影响，因此不会对Q_2造成影响，达到目的

脉冲触发器

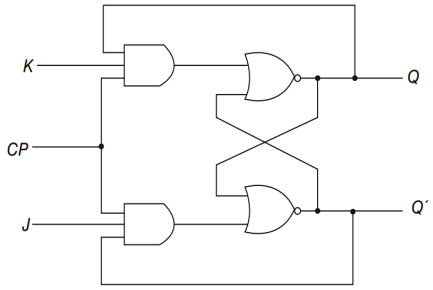


当CLK在高位时，允许更新FF_1，而FF_2按兵不动。但是当CLK一旦下降后，就会把FF_1最后时刻存下的状态传给FF_2，开始改变Q_2。因此，输出端的改变取决于下降沿时刻FF_1的状态。此外，同之前的SR锁存的规则，S与R同时取1是违法的，因为如果真的两个都是1，在CLK断电后，不清楚状态是什么。但是，如果能够把“S与R同时取1”设计成“使$Q^* = Q'$”（即状态取反）这一功能就好了！可让它从一个违法的行为变成一件有益社会的事：

JK触发器

JK电平触器

先来看这个（也可直接跳过，看后面那个大的版本）：



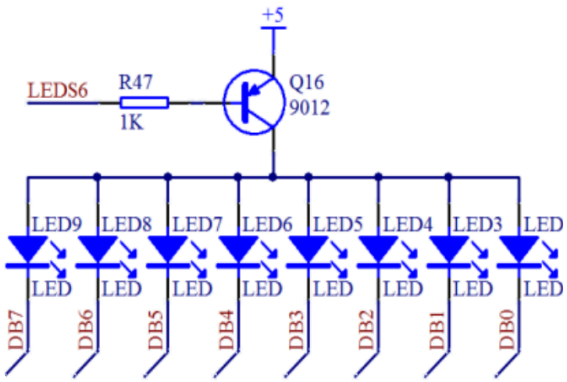
下面是其特性表：

Q	J	K	Q(t+1)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

可以发现，都是1的情况已经很好转化成取反了！

JK脉冲触发

再变成Master-Slave（主从型）赋予边沿触发的特质，成为最终版：



因为输入端比较复杂，要想清楚这个，还是建议按Q的状态分类讨论一下，比如：

1. $Q=0$时，如果$J=1,K=0$，在$CLK=1$的时候更新FF_1，易知由于Q的作用，G_2一定与成0，取反后传出1，而输入G_1的三者全是1，与后取反传出0，这样就把FF_1的锁存部分更新成1状态了。之后下降沿时，就会把这个1状态更新到FF_2那里去，导致$Q=1$

2. $Q=1$时，如果$J=1,K=0$，在$CLK=1$的时候更新FF_1，这时G_1与G_2都与成0，取反后传出1，即给FF_1的锁存部分输入两个1，但输入两个1（不是违法的！要与前面的违法区分开，因为这里有取反，相当于之前的输入两个0，是合法的）并不改变FF_1锁存的状态，因此最后FF_2的更新还是受之前锁存的FF_1支配，相当于和上次更新一样，那结果也和上次一样是$Q=1$ （注意，这一条结论还可以这样理解：因为G_1和G_2门此时都被封锁了，所以信息无法传播，自然不会更新Q）

综合以上两条，得出$J=1,K=0$的作用是把输出更新为状态1的结论

再看看$J=K=1$时：

1. $Q=0$时，如果$J=K=1$，在$CLK=1$的时候更新FF_1，此时G_1门开通，G_2门封锁，将FF_1更新成1状态，这样下降沿时就会把1状态存到FF_2中，使得$Q=1$

2. $Q=1$时，如果$J=K=1$，在$CLK=1$的时候更新FF_1，此时G_2门开通，G_1门封锁，将FF_1更新成0状态，这样下降沿时就会把0状态存到FF_2中，使得$Q=0$

综合上述两者，得到$J = K = 1$的作用是取反！

当然其他两种情况同理，总之现在的逻辑就是：J=K=0时保持，J=1,K=0时更新为状态1，J=0,K=1时更新为状态0，J=K=1时更新为当前的反状态

触发器总结

“触发方式”分为电平、边沿、脉冲三者，“逻辑功能”分为SR，D，JK等，不要混起来这两个独立的方面！

电平触发就是最简单的，与$CLK = 1$这段时间里的整个历史有关，且输出会随着输入的变化动态更新

边沿触发是最保险的，真正的只和瞬间值有关，刚才的例子中只有D边沿触发器能做到这一点！

脉冲触发，与$CLK = 1$这段时间里的整个历史仍有关（注意这点），但是输出的更新只在那一个边沿的瞬间而已

按照逻辑功能分，SR与JK都是双输入的逻辑：SR有三个合法，一个违法的逻辑，而JK就是四个都合法的逻辑（多出来的就是J=K=1时的取反功能）。D则是单端输入，要么更新为1，要么更新为0，逻辑更简单

组成更大的记忆元件

寄存器

Register，像这样把一些触发器拼起来就行了：

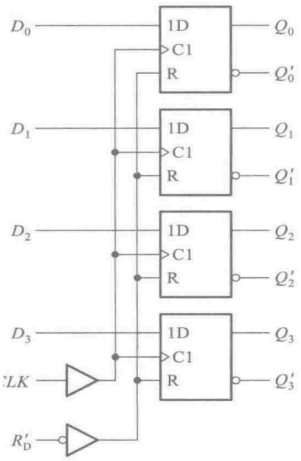


图 5.4.2　74HC175 的逻辑图

存储器

RAM (random access memory)

1. 静态 SRAM
2. 动态 DRAM

ROM (read only memory)，可以放一些程序

下面仅举SRAM的例子

SRAM

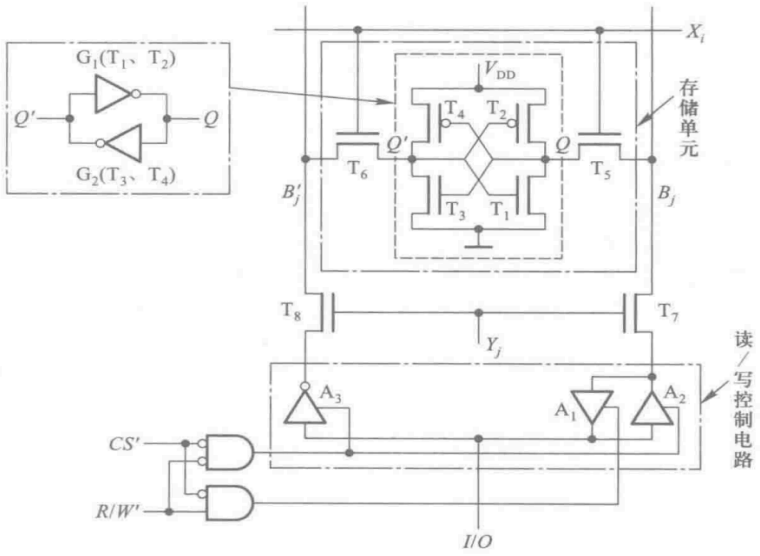


图 5.5.3　六管 CMOS 静态存储单元

把许多的SR锁存器排成一个矩阵，设计成只有其所在行与所在列同时上电后才能对其读写。当输入行地址与列地址（两个编码）后，这两个编码分别经过行译码器与列译码器处理，分别为某一行与某一列通上电，这样该行该列的那个小单元（如上图）就被选中了，允许进行读写操作，别的小单元都不会被读写

为什么只有所在行、列都通电时才能工作呢？见上图的X_i与Y_j：X_i通电后T_5, T_6这两个CMOS开关才会闭合，解开第一道封印；Y_j通电后T_7, T_8这两个CMOS开关才会闭合，解开第二道封印。封印同时解除后，内部的存储单元（锁存器）才与读/写控制电路相通

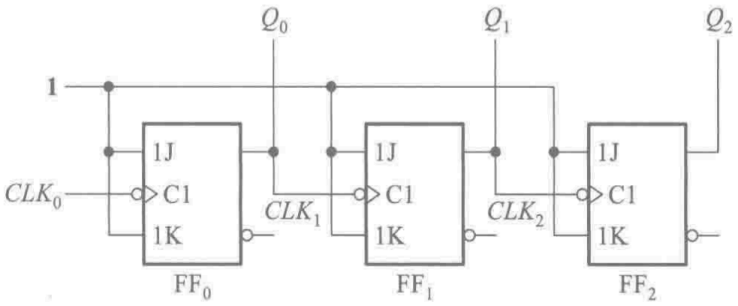
为简明起见，把图中存储单元的小虚线框直接用指向它的那个虚线框代替（其实就是一个锁存器）

如何实现读的操作？$CS' = 0, R/W' = 1$，此时读/写控制电路中只有A_1接通，存储单元里Q的电平会接通到I/O端（别忘了单向特性，反着来是不行的），读出内容

如何实现写的操作？$CS' = 0, R/W' = 0$，此时读/写控制电路中只有A_2, A_3接通，I/O端的电平会按照D触发器的逻辑（单输入），接到Q与Q'，复写存储的内容

当$CS' = 1$时，A_1, A_2, A_3三个门全部关上了，禁止了对内部的一切操作

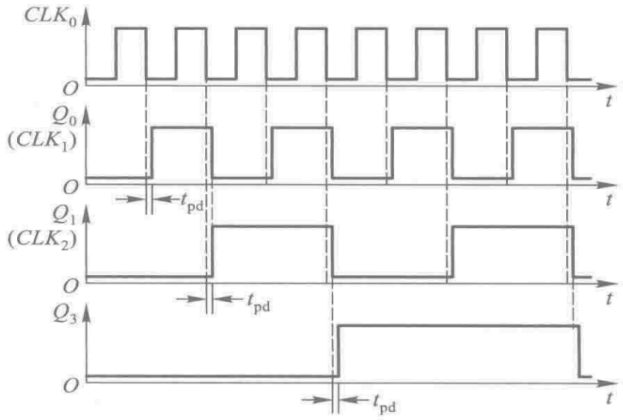
计数功能



举个上图简单的例子，利用的就是JK的逻辑来计数

开始时最左边的FF_0会在第一次更新时由0变1（因为J=K=1逻辑是取反，且初始状态=0），不过因为CLK_1也即Q_0刚刚还是0，所以后面的FF_1按兵不动；到第二次更新到来时，由于已经$Q_0 = 1$了，FF_1就会更新取反，Q_1由0变1，但是由于Q_1刚刚还是0，所以FF_2还是按兵不动，此外，FF_0也要取反又由1到0了……

可以得到下图：



这不就是二进制进位计数嘛！左边的寄存器是低位，右边是高位。基于这种结构，显然可以构建出各种2的幂次倍数的进制计数器，那10进制这种怎么办呢？只要在输出的状态满足“1010”时会自动清零就行了，打断后面的几个计数。实现方法：可以放置一个与门，实现“输出$= a\bar{b}c\bar{d}$”，只有在$a\bar{b}c\bar{d} = 1\bar{0}1\bar{0}$这个最小项的时候（即等于十），与门启动输出1，开启reset端口清零计数器就行了