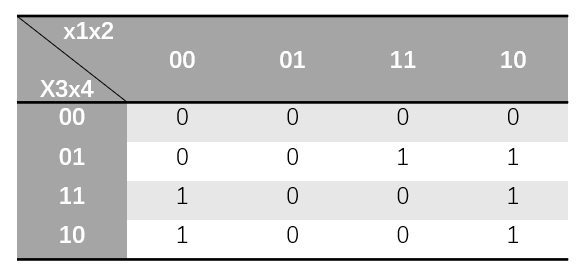
【笔试解析】2021年校招提前批（vivo芯片设计岗）

本期笔试经解析题目来源vivo2021校招提前批**硬件逻辑岗**整套题，题目不难，但是考察的点还是蛮多的，从数字电路到数字逻辑再到时序分析与约束和功耗覆盖率问题都有涉及，总的来说题目很不错，一个小时的做题时间对大部分同学来说也刚刚好，是一次练手的好题目。

vivo作为2021校招提前批的先行者，率先在6月8号开启了第一枪笔试，据说vivo的芯片设计岗是第一年公开校招，工作地点为杭州。

# 一、单选题

## 1、下图为组合逻辑Y=f(x1,x2,x3,x4)的真值表，请根据真值表选择Y的逻辑表达式为







**解析：**用卡诺图化简逻辑函数时可按如下步骤进行：

1）将函数化为最小项之和的形式。

2）画出表示该逻辑函数的卡诺图。

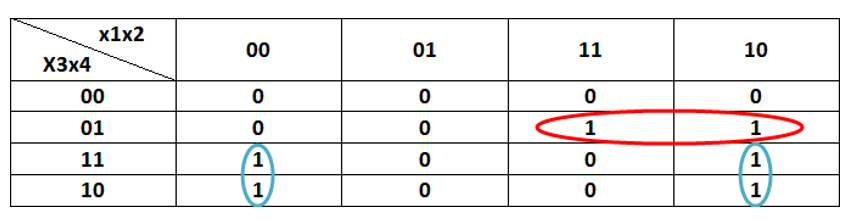
3）找出可以合并的最小项。

4）选取化简后的乘积项。选取的原则是：

①这些乘积项应包含函数式中所有的最小项（应覆盖卡诺图中所有的1）。

②所用的乘积项数目最少。也就是可合并的最小项组成的矩形组数目最少。

③每个乘积项包含的因子最少。也就是每个可合并的最小项矩形组中应包含尽量多的最小项。



红色两个合并结果为，蓝色的四个合并结果为，所以最后的化简结果为，所以答案选择D。

## 2、下面哪个不是循环关键字

A repeat     B forever

C fork        D while

**解析：**fork和join一般同时使用，在fork ... join中存在的语句是并行执行的。与之对应的是begin...end，在begin...end中存在的语句是顺序执行的。**所以答案选择C**。

## 3、netlist一般通过什么手段进行验证其正确性

A 随机验证      B RTL验证

C 形式验证      D 网表验证

**解析：**一般通过形式验证+静态时序分析来验证网表的正确性。

形式验证是验证范畴，它是从功能上（STA是时序上）对综合后的网表进行验证。常用的就是等价性检查方法，以功能验证后的HDL设计为参考，对比综合后的网表功能，它们是否在功能上存在等价性。这样做是为了保证在逻辑综合过程中没有改变原先 HDL描述的电路功能。形式验证工具有Synopsys的Formality。

StaticTiming Analysis（STA），静态时序分析，也属于验证范畴，它主要是在时序上对电路进行验证，检查电路是否存在建立时间（setup time）和保持时间（hold time）的违例（violation）。这个是数字电路基础知识，一个寄存器出现这两个时序违例时，是没有办法正确采样数据和输出数据的，所以以寄存器为基础的数字芯片功能肯定会出现问题。STA工具有Synopsys的Prime Time。**所以答案选择C**。

## 4、下列功耗措施哪个可以降低峰值功耗

A 静态模块级Clock Gating

B Memory Shut Down

C Power Gating

D 大幅度提高HVT比例

**解析：**首先峰值功耗是属于动态功耗中的短路功耗，即NMOS和PMOS同时导通所引起的峰值电流，最终带来的功耗。这个功耗和电源电压，时钟翻转率，以及峰值电流有关。

**（1）从动、静态功耗的角度分析选择A**

A选项显然降低时钟翻转率是降低的动态功耗，所以A选项正确。

B选项存储关闭。即不被访问的时候，关闭存储器，因而也是降低静态功耗。

C选项电源门控技术，即模块不工作的时候，关闭电源，模块睡眠，工作时候再启动电源，是降低静态功耗。

D选项即采用高阈值电压的晶体管，阈值电压增加的效果在于降低亚阈值漏电电流，因而降低静态功耗。

**（2）从短路功耗公式分析选D**

短路功耗公式为：Pshort = τAshortVdd = τAβ(Vdd-Vth)3 ，只跟Vth有关，而D选项中大幅提高HVT（High Voltage Value）带入短路功耗公式中会使短路功耗变小，从而降低动态功耗中的峰值功耗。**所以D选项正确**。

**扩展：**

LVT：低阈值这种库的漏电流较大，但是延迟较小。

SVT：标准阈值居于两者中间。

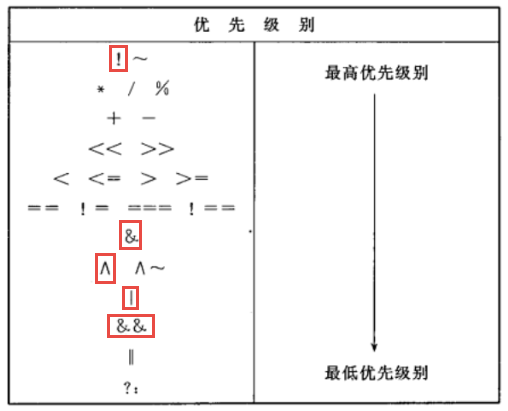
HVT：高阈值这种库的漏电流小，但是延迟大在关键路径上最好用LVT的库，考虑到降低功耗，最好用HVT的库。

## 5、以下Verilog运算符优先级由高到低正确的是

A !, &, ^, |, &&   B ^, !, &, |, &&

C !, |, &, &&, ^   D &, |, &&, ^, !

**解析：**参考下图可以知道**答案选择A**。



## 6、下面哪个不属于跨时钟域数据传递的基本方法

A 使用握手协议

B 使用多级触发器缓冲

C 信号通路上插入isolation

D 使用FIFO

**解析：**这题我们用排除法也是可以做出来的，A**握手、B打两拍、D异步FIFO**都是处理跨时钟域时数据传递的常用方法，**而C选项则是隔离单元**，用来跨power domain的，当一个power domain的电断了，isolation里还可以存储数据，断电不丢失数据。**所以答案选择C**。

## 7、下面哪种不属于验证覆盖率

A 翻转覆盖率        B 状态覆盖率

C 循环覆盖率        D 条件覆盖率

**解析：**给大家介绍一下常用的验证覆盖率有哪些：

语句覆盖率（statement coverage）：指的是程序的每一行代码是否被执行过。

条件覆盖率（condition coverage）：指的是每个条件中的逻辑操作数被覆盖的情况。

决策覆盖率（branch coverage）：指的是在if，case，while，repeat，forever，for 和loop语句中各个分支执行的情况。

事件覆盖率（event coverage）：用来记录某一个事件被触发的次数。

跳转（翻转）覆盖率（toggle coverage）：用来记录某个设计边界信号数据位的0/1跳转情况，例如从0到1，或者从1到0的跳转。

状态机覆盖率（finite stage machine coverage）：仿真器的覆盖率功能可以识别出设计中的状态机部分，记录各种状态被进入的次数，以及状态之间的跳转情况。

**由排除法可得答案选择C**。

## 8、逻辑电路低功耗设计中，无效的方法是

A 采用慢速设计         B 减少信号翻转

C 采用较慢速的时钟    D 提高阈值电压

**解析：**A选项说采用慢速设计并不一定会降低功耗，**所以A选项不正确**。

B选项减少信号翻转可以降低动态功耗。

C选项采用较慢速时钟也相对的降低了信号的翻转，所以也是降低动态功耗。

D选项即采用高阈值电压的晶体管，阈值电压增加的效果在于降低亚阈值漏电电流，因而降低静态功耗。

## 9、组合逻辑电路的冒险现象是由于（）引起的

A 电路未达到最简    B 电路存在延时

C 逻辑门类型不同    D 电路有多个输出

**解析：**由于逻辑门的延迟作用，竞争冒-险现象主要由两个输入信号同时向相反的逻辑电平跳变所生的。**所以答案选择B**。

## 10、X和Y 均为补码表示的二进制，其中X=10010.png"font-size: 15px;letter-spacing: 2px;">

A 110011.png"margin-left: 8px;margin-right: 8px;margin-bottom: 15px;line-height: 2em;">C 11110001.png"margin-left: 8px;margin-right: 8px;margin-bottom: 15px;line-height: 2em;">

**解析：**该题就是纯二进制加法计算，经程序员计算器验证（笔试答题时最好笔算）答案选择D。

## 11、假设一个3bit计数器（计数范围为0~6）工作在38M时钟域下，要把此计数器的值传递到另一个异步100M时钟域，以下不正确的是（）？

A 使用异步FIFO    B 锁存+握手信号

C 使用格雷码        D 使用DMUX电路

**解析**：本题答案为C，之所以不能使用格雷码做同步的原因是计数器的技术范围在0~6，如果是0-7就可以用了，这样格雷码就可以形成闭环（直接使用格雷码做跨时钟域需要2n个连续的计数才可以）。

## 12、D触发器：Tsetup=3ns，Thold=1ns，Tck2q=1ns，该D触发器最大可运行时钟频率是

A 1GHZ            B 250MHZ

C 500MHZ        D 200MHZ

**解析：**Tsetup + Tck2q 是整个路径上delay，最大频率和Thold无关，所以就是4ns，答案为B。

## 13、十进制数-1,用4位二进制表示的原码、补码、反码分别是（）

**A 1001B  0111B    1110B**

**B 1111B   0111B   1000B**

**C 1111B   1110B   1000B**

**D 1001B   1111B   1110B**

**解析：**负数最高位为符号位1，剩余位数按照二进制转化为1001B；负数取反码时符号位不变，其余位取反，为1110B；负数取补码为原码取反加1，为反码直接加1，为1111B。所以D选项正确。

## 14、无符号二进制除法1110111B/1001B的结果是

**A 商：1101B ， 余数：110B**

**B 商：1101B ， 余数：110B**

**C 商：1101B ，余数：10B**

**D 商：101B  ， 余数：10B**

**解析：**可以先转成十进制计算，再转换成二进制得出答案

1110111B= 119D，1001B = 9D

1110111B/1001B =119D/9D=13D...2D，

余数：13D = 1101B，商：2D = 10B，所以答案选择C。

## 15、在RTL设计阶段，降低功耗的常用设计方法是

**A 门级电路的功耗优化    B 门控时钟**

**C 降低电路漏电流          D 多阈值电压**

**解析：**这四个选项都是可以降低功耗的常用设计方法，但是有个前提，那就是在RTL设计阶段，也就是我们编写代码时可以控制的阶段，可以在代码中加入门控时钟，所以C选项正确。

## 16、时钟的占空比指的是（）

**A 时钟的变化速度**

**B 时钟的变化范围**

**C 低脉冲的持续时间与脉冲总周期的比值**

**D 高脉冲的持续时间与脉冲总周期的比值**

**解析**：时钟的占空比就是高电平（脉冲）所占总周期的比率，如占空比为0.5即高电平持续时间占整个周期的二分之一。所以D选项正确。

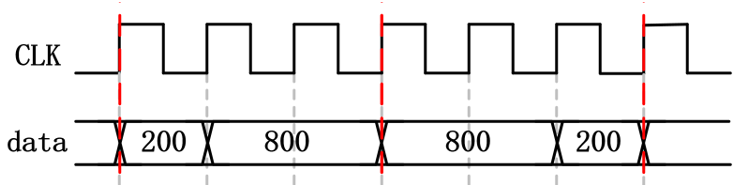
## 17、有一个FIFO设计，输入时钟100MHz，输出时钟80MHz，输入数据模式是固定的，其中1000个时钟中有800个时钟传输连续数据，另外200个空闲。请问为了避免FIFO下溢/上溢，最小深度是多少

**A 320             B 80**

**C 160             D 200**

**解析：**这种题目是比较常见的求解FIFO最小深度的问题。一般的情况都是写时钟快，读时钟慢，且为单位时间内的突发模式，解题的关键是要考虑背靠背问题。

在1000个时钟内突发800个数据，但是在这1000个时钟内突发800个数据的时间是不固定的，如下图所示，我们考虑最坏的情况，相当于在2000个时钟内连续突发1600个数据，因为写时钟快，读时钟慢，所以我们需要FIFO进行存储。在写入数据的时候我们同时还可以读取，而FIFO的最小深度就是计算突发1600个数据时写和读的差值，即保证FIFO不被写满。1600-(80MHz/100MHz)\*1600=320个，所以答案选A。

****

# 二、多选题

## 18、对解决亚稳态问题有效果的方法是

**A 用反应更快速的DFF**

**B 改善时钟质量，用边沿变化快速的时钟信号**

**C 降低时钟频率**

**D 引入同步机制，如加两级触发器**

**解析：**这四个选项都对解决亚稳态问题有效果，所以全部正确的。

## 19、如下verilog描述中哪个或者哪几个会生成寄存器？

**A always@(clk)**

**reg\_a <= reg\_b;**

**B always@(negedge clk)**

**reg\_a<= reg\_b;**

**C always@(posedgeclk)**

**reg\_a<= reg\_b;**

**D assign reg\_a =reg\_b;**

**解析：**A选项中虽然也有clk，但是不是沿触发的，而是用always实现的组合逻辑。B、C毫无疑问，无论是上升沿触发还是下降沿触发都会产生寄存器，所以我们可以得出结论能不能生成寄存器的关键条件是always敏感列表中的触发条件。而D选项也没有什么疑问了，用assign实现的组合逻辑赋值。所以答案选择B、C。

## 20、Supposedly there is a combination circuit between two registers driven by a clock. what will you do if the delay of the combinational circuit is greater than clock signal

**A to reduce clock frequency**

**B to increase clockfrequency**

**C to make it pipelining**

**D to make it multi\_cycle**

**解析：**先翻译下这段英文，大概意思是：假设一个时钟驱动的两个寄存器之间有一个组合电路，如果组合电路的延迟大于时钟信号怎么办。再看四个选项的翻译：A 降低时钟频率；B 增加时钟频率；C 插入流水线设计；D 做multi\_cycle约束。

在实际的工程中，经常会碰到数据被发起后，由于路径过长或者逻辑延迟过长要经过多个时钟周期才能到达捕获寄存器；又或者在数据发起的几个周期后，后续逻辑才能使用。组合电路的延迟大于时钟信号会导致不能每一拍都有结果，有可能会导致结果的混乱，为了让每一拍都尽可能采到结果可以选择降低时钟频率，也可以通过插入寄存器来降低两个寄存器之间组合逻辑的延迟。这时如果按照单周期路径进行时序检查，就会报出时序违规。所以我们在约束的时候需要设置多周期路径约束，即做multi\_cycle约束。所以答案选择A、C、D。

# 三、【简答】

## 21、解释setup和hold time violation ,并说明解决办法。

**解析：**setuptime violation为建立时间违例：即不满足时钟沿来到之前数据必须保持稳定的时间。hold time violation为保持时间违例：即不满足时钟沿来到之后数据必须保持稳定的时间。

**关于解决时序违例的方法请参考：[求职攻略|三步优化FPGA时序设计](http://mp.weixin.qq.com/s?__biz=Mzg5MDIwNjIwMA==&mid=2247488126&idx=1&sn=e951be4f5177463f7f3f22585aef0e4f&chksm=cfe170e9f896f9ff8425a202d680046c41253564d6e19f67f1da7cb8f391de0107c093d48316&scene=21" \l "wechat_redirect" \t "_blank)。**

## 22、请描述如下代码，实现加法；

**C = A + B；**

**A是21bit无符号数；**

**B是18位有符号数；**

**如何实现才能保证正确得到一个不溢出的有符号数C；**

**解析：**我们可以看到这是不等位宽无符号与有符号数的加法运算，需要将其位宽对补齐后运算，还要注意让C不能溢出。代码如下：

**方法一：**

**//----------------------------------------------**

**01module unsign\_sign(**

**02     input  wire               [20:0]  A,**

**03     input  wire   signed    [17:0]B,**

**04**

**05     output wire    signed    [22:0]C**

**06);**

**07**

**08assign C =$signed({1'b0, A})+$signed({{4{B[17]}}, B});**

**09**

**10endmodule**

**//----------------------------------------------**

**方法二：**

**//----------------------------------------------**

**01module unsign\_sign(**

**02     input  wire               [20:0]A,**

**03     input  wire   signed    [17:0]B,**

**04**

**05     output wire    signed    [22:0]C**

**06);**

**07**

**08assign C =$signed({1'b0,A})+ B;**

**09**

**10endmodule**

**//----------------------------------------------**

大家可能会有疑问，说为什么需要做强制有符号类型的转换，我们应该明确一点：只要参与运算的式子里，有一个数是无符号的，整个式子就会按照无符号计算。上述的两种方法均经过ModelSim仿真验证了其正确性（需要注意的是仿真时A以Unsigned型格式显示，B、C以Decimal型格式显示，所有的值都会以二进制补码的形式呈现）。

## 23、请找出下面异步同步电路当中存在的问题并改正

**reg reg\_aa,reg\_ab,reg\_ac reg\_ad;**

**reg reg\_ba,reg\_bb;**

**always@(posedge clk\_a)**

**reg\_ab <= reg\_aa;**

**always@(posedge clk\_a)**

**reg\_ad <= reg\_ac;**

**always@(posedge clk\_b)**

**reg\_ba <= reg\_ab & reg\_ad;**

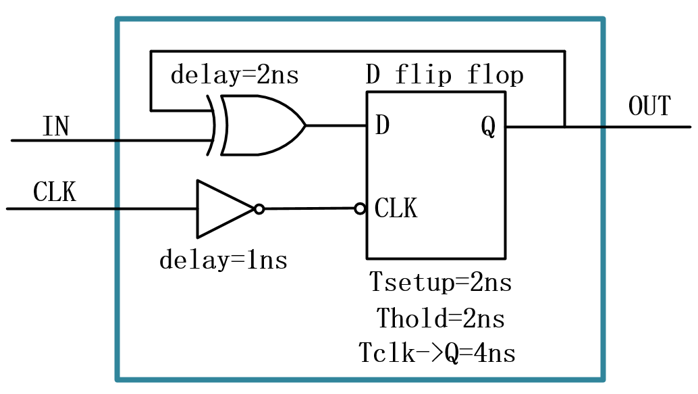
**always@(posedge clk\_b)**

**reg\_bb <= reg\_ba;**

**解析：**我们细心观察可以发现该题的时钟有两个clk\_a和clk\_b，这显然是一种跨时钟域的问题，我们先找到跨时钟域的位置，发现从reg\_ab & reg\_ad开始进入了时钟域clk\_b中，而用clk\_b时钟来采集在clk\_a时钟域下产生的reg\_ab和 reg\_ad数据在高速情况下肯定会大概率出现亚稳态现象。所以我们最好能让reg\_ab & reg\_ad在clk\_a时钟下进行，然后将其相与后的reg\_ba信号在clk\_b时钟下打两拍（默认Tclk\_a> Tclk\_b），然后再将打拍后的结果赋值给reg\_bb即可。

当然有些同学会说，能不能先把reg\_ab和 reg\_ad各在clk\_b下打两拍后再相与，虽然也可以，但是不如先在clk\_a中相与后再在clk\_b中打拍更方便。

## 24、下图中的电路，器件延时如图中标注，将框内电路作为一个寄存器，其有效setup time=（）ns, hold time=（）ns？

****

**解析：**有效建立时间：假设电路的有效Setup为Tsetup\_valid，对于D触发器而言，其本身的建立时间是2ns，也就是说数据必须在时钟有效沿到达之前2ns保持稳定，这样到达D端后就一定是稳定的数据了。这个电路的数据来自于IN，时钟来自于CLK。

考虑时钟路径延迟影响：电路有效建立时间Tsetup\_valid = Tsetup - 1ns = 1ns（因为数据需要提前1ns稳定下来）

考虑数据路径延迟影响：Tsetup\_valid= Tsetup - 1ns + 2ns = 3ns；（经过组合逻辑后的数据需要在时钟有效沿之前Tsetup时间稳定下来）

有效保持时间分析：和建立时间分析套路一致，对于D触发器而言，数据需要在时钟有效沿到来之后保持Thold时间。

考虑时钟延迟的影响：考虑到电路时钟对于触发器时钟早到1ns，所以电路有效保持时间Thold\_valid = Thold + 1ns = 3ns;

考虑路径延迟影响：数据需要经过一段组合逻辑之后才能保持稳定，因此电路的有效保持时间为：Thold\_valid = Thold + 1ns - 2ns = 1ns。

**END**