**浅谈lower power flow （向大佬学习）**

[快乐的芯片工程师](javascript:void(0);) 2022-04-08 10:00

以下文章来源于志芯 ，作者Jack Xu

## 降低功耗的方法

降低静态功耗的方法：非关键路径HVT cell 替换、coarse grain（粗粒度）、fine grain（细粒度）、power shutdown、减少decap\_cell、独热编码、散热降温等。

降低动态功耗的方法：降压、power island、DVSF、非关键路径HVT cell 替换、clock gating、memory split、signal gating、transition time 约束、减小高速信号的走线长度、时钟的multi-slice等。

挑两个听的少的概念稍作阐述。

## coarse grain & fine grain

所谓的粗粒度和细粒度应该是一种前端的设计思想，举例说明，当然我说的不一定对。例如做人脸识别算法时，一种实现方法是将人脸划分为M个区域，然后比较每个区域的像素情况，假设像素信息是N个比特位，那么识别人脸信息就需要M\*N比特的寄存器去传输这个数据；另外一个实现方法就是分两步，第一步设别脸部某个特殊部位（例如鼻子或者眼睛），然后用该特殊部位的像素情况替代整体，从而识别独一的个体，此时只需要N比特的寄存器去传输这个数据。

再说个例子，就是识别一个人的手掌根本不需要知道手掌的全貌信息，只需要知道手掌上一个指头的指纹信息就行。

## DVSF

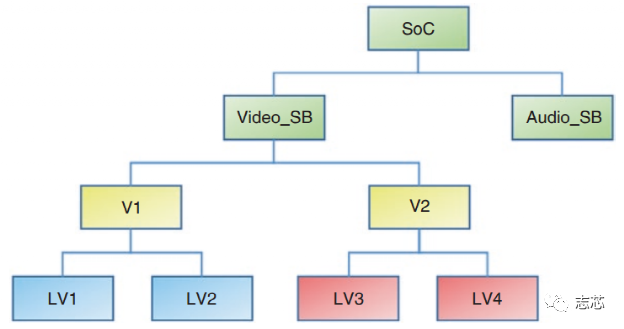
所谓的DVSF就是动态电压和频率调节。扩展多电压的原理，可以允许在芯片工作期间改变电压以匹配当前的工作负载。例如，笔记本电脑中的处理器芯片进行简单的电子表格计算时可以在较低的电压和较低的时钟频率下工作，从而节省功耗; 然后继进行3-D图像渲染时，在更高的电压和更高时钟频率下工作。芯片工作期间供电电压和工作频率的变化满足工作负载要求称为动态电压和频率调整。

芯片电源可以设计为多个级别，甚至是连续的电源范围。动态电压调节需要多级电源和逻辑模块确定用于给定任务的最佳电压电平。由于电源范围以及电压电平和工作频率的组合，设计，实现，验证和测试都特别具有挑战性。

动态电压调节可以与电源开关技术相结合。设计中的模块可以在多个电压电平下工作，以实现不同的性能要求，或在不需要时完全关闭。

## upf

假设某design的module 层次如下图所示。



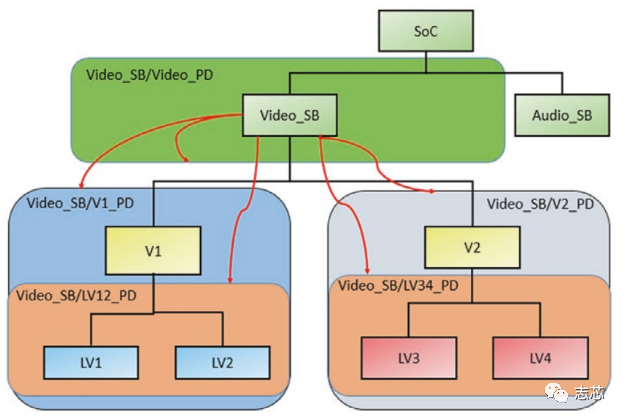
### 1设定实施约束的层次

**set\_design\_top SoC/Video\_SB #指定design\_top为Video\_SB**

**set\_scope . 　#指定命令作用的范围为当前层次即Video\_SB**

**set\_scope V1/LV1 #指定命令作用的范围为V1/LV1**

**2设定power\_domain**Video\_SB top层次下存在下面5个domain和sub\_domain。



Video\_SB/Video\_PD、Video\_SB/V1\_PD、Video\_SB/V2\_PD、Video\_SB/LV12\_PD、Video\_SB/LV34\_PD

设定power\_domain的步骤如下：

**1.set\_scope Video\_SB　　　#set the scope to Video\_SB**

**2.create\_power\_domain Video\_PD -include\_scope  #create Video\_PD power domain in current scope.**

**3.create\_power\_domain V1\_PD -elements {V1} 　 #creates V1‘s domain，its own power domain called V1\_PD**

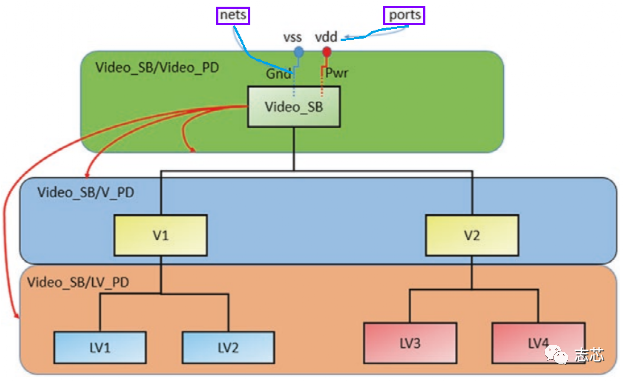
**4.create\_power\_domain LV12\_PD -elements {V1/LV1 V1/LV2}**

**5.create\_power\_domain V2\_PD -elements {V2}**

**6.create\_power\_domain LV34\_PD -elements {V2/LV3 V2/LV4}**

### 3描述power\_domain的电源网络

power domain中电源网络的定义如下图。



主要是通过下面四个进行电源网络的连接。

**create\_supply\_port VDD -direction in  
create\_supply\_port VSS -direction in**

**create\_supply\_net Pwr -domain Video\_PD  
create\_supply\_net Gnd -domain Video\_PD**

**connect\_supply\_net Pwr -ports (VDD)  
connect\_supply\_net Gnd -ports (VSS)**

**set\_domain\_supply\_net Video\_PD  
-primary\_power\_net Pwr  
-primary\_ground\_net Gnd**

下面，V\_PD domain 中的supply\_net复用Video\_PD domain中的supply\_net。需要-reuse进行描述

**create\_supply\_net Pwr -reuse -domain V\_PD  
create\_supply\_net Gnd -reuse -domain V\_PD**

同时，必须显示的申明net所属的power\_domain

**set\_domain\_supply\_net V\_PD  
-primary\_power\_net Pwr  
-primary\_ground\_net Gnd**

### 4创建power\_switch 图片

在综合时只会检查语法，不会有实际效果，但在综合时可以将该UPF指令读入系统并输出相应的UPF给PR工具，PR工具可以自动识别这些指令，到时才会真正添加Power Switch。

**create\_supply\_net VDDsw -domain V\_PD #定义switch产生的supply\_net。**

**create\_power\_switch SW -domain V\_PD  
-input\_supply\_port {pwin Pwr}  
-output\_supply\_prt {pwout VDDsw }  
-control\_port {swctrl sw\_ctrl}  
-on\_state {Pwon swctrl}  
-off\_state {Pwoff !swctrl}**

**set\_domain\_supply\_net V\_PD \ #V\_PD的primary\_power\_net为switch产生的net  
-primary\_power\_net VDDsw  
-primary\_ground\_net Gnd**

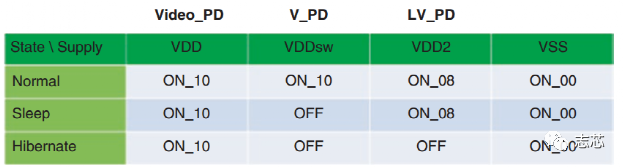
### 5设定Supply Port的值

add\_port\_state VDD -state {ON\_10 1.0} #VDD只有一种常开的状态，其电压值为1.0v

add\_port\_state VSS -state {ON\_00 0.0}

add\_port\_state SW/VDDsw -state {ON\_10 1.0} -state {OFF off} #switch开关电压VDDsw,存在ON和OFF两种状态。

### 6 Power State Table的描述



create\_pst PST1 -supplies {VDD, VDDsw, VDD2, VSS}

add\_pst\_state Normal -pst PST1 -state {ON\_10, ON\_10, ON\_08, ON\_00}  
add\_pst\_state Sleep -pst PST1 -state {ON\_10, OFF, ON\_08, ON\_00}  
add\_pst\_state *~~Normal~~* （Hibernate） -pst PST1 -state {ON\_10, OFF, OFF, ON\_00}

Hibernate == 冬眠

### 7 set retention快速恢复被关断domain的状态 图片

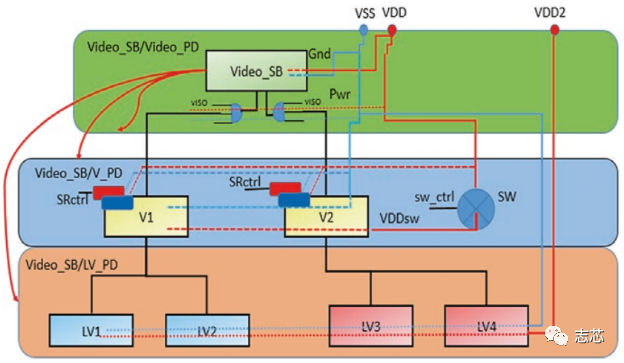
其中retention cell的电源和地为红色VDD和VSS

set\_retention V\_PD\_retention \ #所以V\_PD domain的状态都要能被恢复  
-domain V\_PD  
-retention\_power\_net Pwr  
-retention\_ground\_net Gnd

set\_retention\_control V\_PD\_retention  
-domain V\_PD  
-save\_signal {SRctrl posedge}  
-restore\_signal {SRctrl negedge}

### 8用isolation cell 隔离ON domain和OFF domain 图片

OFF domain 其输出连接至ON domain，所以其输出需要处理。



set\_isolation V\_PD\_isolation  
-domain V\_PD  
-applies\_to outputs  
-clamp\_value 0  
-isolation\_power\_net Pwr  
Isolation\_ground\_net Gnd

set\_isolation\_control V\_PD\_isolation  
-domain V\_PD  
-isolation\_signal vISO  
-isolation\_sense high  
-location parent

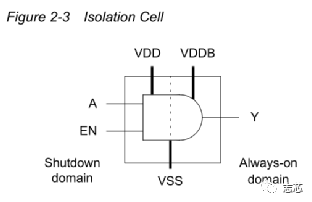
### 9 level shift cell 处理不同电压域之间信号的传递 图片图片

set\_level\_shifter LV\_PD\_LS-domain LV\_PD-threshold 0.1-applies\_to both-rule both-location self

## low power cell

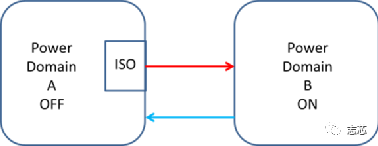
## isolation

isolation cell 一般用于隔离两个不同的 power domain ，其示意图如下：



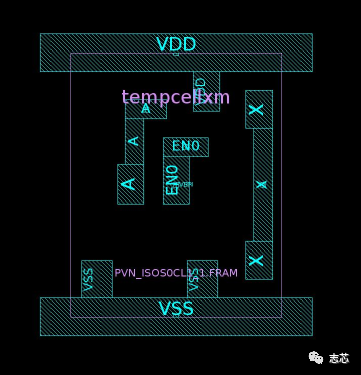
iso cell 有一个控制端 EN， 当 EN 无效时， A 端信号直接送到 Y 端，此时 iso cell 等效于一个buffer；当 EN 有效时，buffer 断开，Y 端保持固定的高电平或者低电平；上面这种 iso cell 有两组 power：primary power VDD 和 backup power VDDB，当 左边domain 关掉时， VDD off，此时就由 VDDB 供电，维持 Y 端的固定电平。

当信号从一个 off domain 穿到 另一个 domain （on 或者 off）时，如果 domain A 关掉，而 domain B 还在运行，就会出现 B 的input floating，此时 B 就可能会因为输入不定态而出现错误；所以当信号从 A 到 B 时，需要加 iso cell，保证在 A 关掉时，A 的输出信号维持在稳定的电平



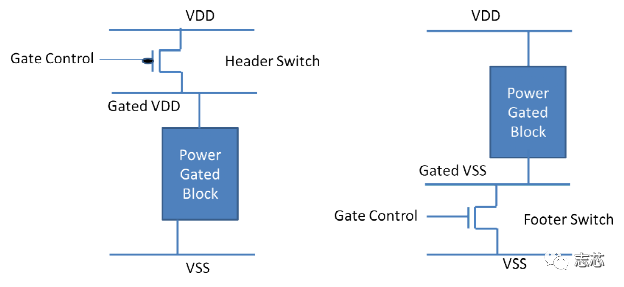
有个问题需要注意，当 domain A 关掉时， nwell 也断电了，那么其中的 iso cell 的 nwell 也断电了，那么iso cell 就可能出现漏电（？），这个问题有两种解法：一种是将 iso cell 的nwell 做成孤立的（与旁边的std cell 的nwell 隔开），然后连接到 backup power VDDB 上，保证nwell 不会断电；另一种更推荐的办法是将 iso cell 放到 domain B 中去，也就是在domain B 的input 加 iso cell，而 A 的output 不再需要 iso，这样就不用担心 nwell 断电的问题了。

常用的 iso cell 其实是不带 VDDB 的，仅有primary power，如下图，这种 iso cell 就只能放在 domain B 的input 端了。

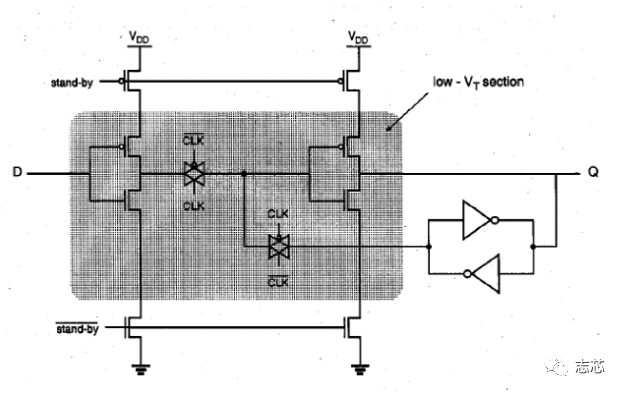


## **Power Switch**

**Power Switch**(电源开关单元)用于电源关断技术(Power Shut Off, PSO)，它通过关断芯片中暂时不需要的某个区域或者某个子模块的供电电压，来达到降低静态功耗的目的。也叫MTCMOS即Multi-threshold CMOS （多阈值电压CMOS）。常见的switc结构如下。

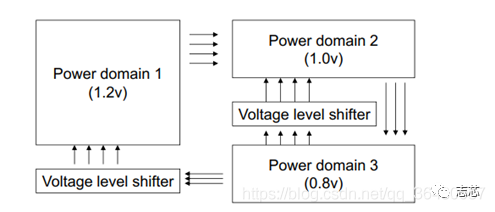


还有一种不常见的结构如下。



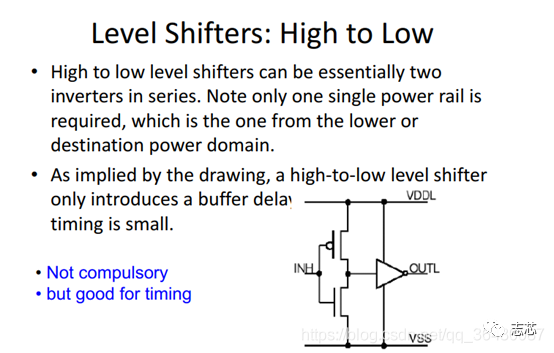
## level shifter

信号跨越不同级别电压模块进行驱动时，需要使用level shifter。



从高电压到低电压的level shifter，可以是两个反相器的串联。

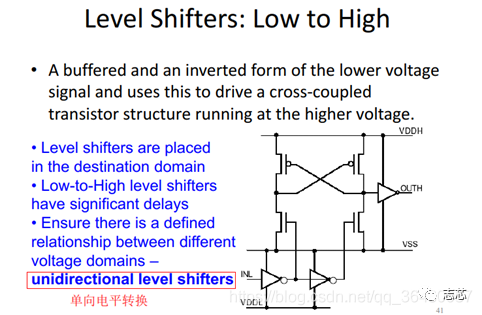
输入高电平区域信号INH，输出低电平区域信号OUTL，注意供应电源由低电压模块提供，即图中的VDDL。

从图中可以看出，H2L的level shifter只会引入一个buffer的延迟，对时序的影响较小。  


用低电压区域的信号invert和buffer之后得到的一对信号来驱动工作在高电压区域的交叉耦合晶体管结构。

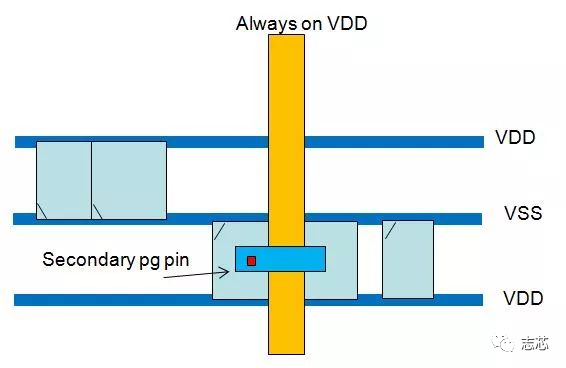
输入低电平区域信号INL，输出高电平区域信号OUTH，注意电源电压由高电压模块提供，level shifter也放置在高电压模块中。

L2H level shiter有明显的延迟。



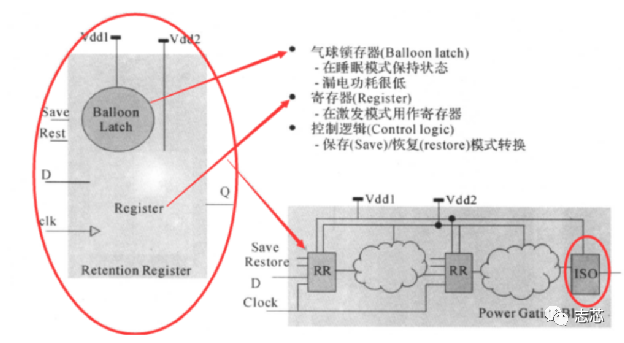
## always on buffer

它与普通cell的区别在于，多了一个**secondary power pin**，这个second power pin必须连在always on stripe上。



## retention reg

支持Retention的设计会将普通的寄存器，替换成带Retention的寄存器，它由主从两部分组成：常规寄存器（主）和Retention寄存器（从）。



工作过程如下：

（1）正常工作时，主寄存器工作，供电电压高，从寄存器仅保持一个低电压供电，其CLK被关掉。

（2）进入Low Power模式，主寄存器掉电，掉电前将其信息写入从寄存器，从寄存器完成对主寄存器的采样之后CLK被关掉，随后在Low Power模式期间一直保持低电压供电。

（3）重新工作时，主寄存器上电之后，从寄存器将其保存的内容写回主寄存器。主寄存器正常工作。从寄存器依然保持低电压供电，CLK关掉。

看起来多了一个Retention寄存器，似乎费电更多才是，那怎么做到省电的呢？

关键就在于，Retention寄存器只消耗很少的电来保存信息，从而使得主寄存器可以完全掉电，从而省电。这里就要说一说主从寄存器两者的区别：

主寄存器面积大，工作电压高，采用的晶体管阈值电压也低，工作电流大，这样可以保证该寄存器在正常工作时速度快，负载能力强。Retention寄存器，面积相对小一些，采用阈值电压高的晶体管，其驱动电压也会更低。因此，消耗的电量很少。

还要注意，包含retention逻辑的寄存器，因为有两个电压域，为防止电压互相冲击，需要插Isolation Cell做电压隔离。

# Magnet placement

在place的时候，对于level shift cell要考虑transition，这时候一般用magnet placement实现。

图片

本文思路框架出自本人，部分参考文献如下。

https://blog.csdn.net/qq\_36480087/article/details/111589349；作者：小猴子。

https://blog.csdn.net/gemengxia/article/details/108379016?spm=1001.2101.3001.6650.4&utm\_medium=distribute.pc\_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-4.no\_search\_link&depth\_1-utm\_source=distribute.pc\_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-4.no\_search\_link；作者：爱吃蛋挞的Dolly。

https://www.cnblogs.com/xiaoxie2014/p/10837155.html；作者：いつまでも。

https://www.cnblogs.com/xiaoxie2014/p/10837155.html；作者：hippomyl。

http://www.faadoengineers.com/online-study/post/ece/vlsi-design/1239/multiple-threshold-cmos-mtcmos-circuits。

图片