北京大学

硕士学位论文

PKUnity-3（SK）系统芯片功耗管理部件的设计与实现

姓名：杨锟

申请学位级别：硕士

专业：计算机系统结构

指导教师：王克义

20090601

摘　要

随着集成电路制造工艺的发展，集成电路的集成密度和性能迅速提高，加之移动和分布计算以及片上系统的广泛应用，功耗已经成为继成本、功能、面积、性能之后的关键设计特性。

本文从CMOS集成电路的功耗组成出发，详细分析了常用的功耗管理技术。以PKUnity-3（SK）系统芯片为平台，设计实现了由时钟生成模块和功耗管理模块组成的功耗管理部件，主要内容包括：

1．分析 PKUnity-3（SK）系统芯片的时钟需求，设计系统的时钟结构，完成时钟生成模块的 RTL 实现并以减小时钟偏差为目标对时钟生成模块进行逻辑优化，为与时钟控制相关的功耗管理技术的实现提供了基础。

2．基于 PKUnity-3（SK）系统芯片的时钟结构，实现动态变频技术，允许系统根据工作负荷量动态地调整时钟频率，从而降低芯片的动态功耗。

3．设计功耗管理模块的软件可编程寄存器与状态机，通过对功能模块的时钟信号以及系统芯片的全局时钟电路进行控制，允许系统根据工作负荷量在全集工作、子集工作、处理器空闲、系统休眠几种工作模式中切换。

4．重新设计时钟生成模块。新的时钟生成模块牺牲了一些灵活性，设计复杂度也有所提高，但是具有更为平衡的电路结构，为后端实现提供了一个较为理想的起点。

综上所述，本文通过时钟生成模块和功耗管理模块的设计与优化，探索实现了适合PKUnity-3（SK）系统芯片的功耗管理技术，对该芯片在65nm工艺下的功能升级版本的低功耗设计也有一定的借鉴意义。

关键词：系统芯片，功耗管理，时钟生成，动态变频，时钟门控

Design and Implementation of Power Management Unit in PKUnity-3(SK) SoC

Yang Kun (Computer Architecture)
Directed by Prof. Wang Keyi

Abstract

With the development of fabrication technologies, the density and speed of integrated circuit has greatly increased, and power has become one of the most important design concerns as cost, functionality, area and performance.

In this paper, we begin with the source of power dissipation of CMOS integrate circuits and the widely-used power management techniques. Power Management Unit composed of clock generator and power manager is designed and implemented. Our major work includes:

1. Designed the clock generation and distribution network for PKUnity-3(SK) SoC based on its clock requirements, completed RTL implementation and logic optimization of clock generator, which provides a robust platform for the clock-related power management techniques.

2. Implemented dynamic frequency scaling(DFS) in PKUnity-3(SK) SoC. The operating system is able to adjust clock frequencies according to the current workload to decrease dynamic power dissipation.

3. Designed both the register interface and state machine of power manager. By controlling the input clocks of function modules or the global clock network, the system is able to switch from normal mode(Fully-On) to low power mode, such as Partially-On, IDLE or Sleep.

4. Redesigned clock generator. The new design is less flexible and more complex, but the circuit is more balanced, providing a better start point for back-end implementation.

In summary, we developed a robust clock generator and employed several effective power management techniques in PKUnity-3(SK) SoC. The design methods mentioned in this paper work well for upgraded PKUnity-3(SK) SoC using 65nm process.

Keywords: SoC, power management, clk generation and distribution, dynamic frequency scaling, clock gating

图片目录

表格目录

北京大学学位论文原创性声明和使用授权说明

原创性声明

本人郑重声明： 所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品或成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本声明的法律结果由本人承担。

论文作者签名：杨锟 日期：2019 年 6 月 9 日

学位论文使用授权说明

（必须装订在提交学校图书馆的印刷本）

本人完全了解北京大学关于收集、保存、使用学位论文的规定，即：

- 按照学校要求提交学位论文的印刷本和电子版本；
- 学校有权保存学位论文的印刷本和电子版，并提供目录检索与阅览服务，在校园网上提供服务；
- 学校可以采用影印、缩印、数字化或其它复制手段保存论文；
- 因某种特殊原因需要延迟发布学位论文电子版，授权学校□一年/□两年/□三年以后，在校园网上全文发布。

（保密论文在解密后遵守此规定）

论文作者签名：杨锟 导师签名：

日期：2019 年 6 月 9 日

版权声明

第1章 绪论

1.1.功耗管理的重要性

随着集成电路制造工艺的发展，单芯片上集成的晶体管以极快的速度增加。1965年，Intel公司的创始人之一戈登•摩尔（Gordon Moore）博士在总结存储器芯片的增长规律时阐述了著名的"摩尔定律"：在每一平方英寸硅晶圆上的晶体管数量每隔12月翻一番；后来，根据发展情况将这一论述改为：晶体管的数目每18个月翻一番。单位面积上晶体管数量的迅速增长使得单个晶体管的成本快速下降，从而带来了功能强大的集成电路，且其速度与性能不断提升。根据美国半导体工业协会（SIA）2006年更新的国际半导体技术蓝图中的数据，集成电路在未来10年内仍将沿着摩尔定律的方向继续前进。到2020年，器件的特征尺寸将减小到14nm，微处理器芯片的集成度将达到每平方厘米353.91亿个晶体管，ASIC的集成度将达到每平方厘米71.92亿个晶体管，而单个ASIC芯片能够集成的晶体管数目将超过617.07亿个[1][2]。

而同时，晶体管密度的增加也导致了芯片功耗的快速增长。随着半导体器件特征尺寸的减小，单位面积上集成的晶体管数目快速增加。特征尺寸的下降带来了驱动电压的下降，器件速度的提升和成本的下降；但是晶体管集成密度的上升会导致功耗和电流密度的增加。

以微处理器芯片为例来分析电路功耗与工艺进步之间的关系：随着VLSI工艺的进步，器件的负载电容减小，处理器的工作电压降低，单个器件完成相同工作所消耗的能量是减小的。所以当采用新的工艺实现原有的体系结构时，最大功耗就会下降。但是，体系结构设计者往往会利用VLSI的进步实现更复杂的功能，电路集成度的提高和时钟主频的提升给功耗带来了负面的影响，从而导致新一代体系结构的处理器的功耗密度继续上升。

电路功耗作为集成电路的一项基本物理属性是无法消除的。而且，随着集成电路功耗密度的快速增长，它已经成为设计人员考虑的首要因素之一。功耗问题的提出，首先是基于功耗所造成的危害，它主要表现在以下几方面：

- 增加处理器与计算机系统的制造成本

处理器功耗的上升带来了制造成本的增加：首先是芯片的封装成本，其次影响到计算系统的散热成本。当处理器的功耗上升时，封装所使用的材料与工艺需要随之改进，其成本也不断上升。例如成本很低的塑料封装散热能力为 1W，稍好的塑料封装散热能力可达到 2W，陶瓷封装具有更强的散热能力，但成本也更高。在处理器功耗上升的同时，计算机系统需要增加额外的装置帮助芯片散热。目前个人电脑中使用的微处理器普遍需要专门的风扇进行风冷散热。在温度更高的情况下，还需要使用水冷、半导体散热甚至液氨散热的装置。而这些附加装置不但增加了计算机系统的设计复杂度，也提高了整机成本。

- 降低系统可靠性

电路功耗增加导致的另一个负面效应是电路工作稳定性的下降。随着时钟频率的提高，高速切换的电流会显著的降低电路可靠性。同时，功耗增加导致芯片温度上升，会引起诸如硅片连线故障、封装故障、电学参数漂移、电迁移等故障。一般来说，衬底温度每升高 10℃，系统可靠性将下降 50%，同时性能损失 3%。

- 限制处理器性能的提高

功耗急剧增加使得处理器温度升高，严重影响到系统的稳定性。为了保障计算系统稳定运行，人们只能采用降低频率的方法来降低温度。通过提高运行频率来提高处理器性能的方法，在电路功耗超过目前封装技术所能承受范围的状况下，已经变得难以持续进行。2005 年，Intel 公司放弃 4GHz Pentium4 处理器就是一个典型的案例。由于目前高频电路中的功耗问题，处理器的研发不得不考虑以多核架构来弥补无法提升频率带来的性能停滞。

- 限制高性能计算机的设计

今天的高性能服务器机群功耗是相当惊人的。例如，一台 1U 机架设计的基于 AMD Opteron 或 Intel Xeon 处理器的服务器，大约消耗 300W~400W 的功率；由这样的 24 台机器安装而成的一个机架功耗将能达到 7.2kW~9.6kW。有数据显示，Dell PowerEdge 刀片系统可以在一个 42U 的机柜中聚集多达 60 片刀片服务器，这样在一个机柜中达到了惊人的 30.4kW。对于高性能科学计算集群而言，功耗开销更是惊人，例如地球模拟器的最大功耗达到 12MW，未来的 petaflops 系统的功耗可能达到 100MW。电路功耗的快速增长令整个计算系统的用电与散热开销日益成为严重负担。

- 限制移动计算设备的使用

移动计算在现代信息社会的地位越来越重要，而目前电池供电时间短成为移动计算设备使用的主要瓶颈，这一日趋严重的形势对芯片的低功耗设计也提出了更高的要求。移动计算设备的供电目前都是通过电池来进行。在过去的 30 年中，电池的容量仅增长了 2～4 倍。即使采用新的电池技术，预计电池寿命的增长也只有 40%左右。而计算需求、芯片频率与规模的增长远远超过这个速度。如果只依赖电池技术的发展，必然导致移动计算停滞不前。研究低功耗高性能的芯片成为移动计算领域的迫切需求。

综上所述，功耗问题不仅是嵌入式和移动计算领域才面临的问题，它对桌面计算、高性能计算都带来了严重的影响，必须通过有效的方法对其进行控制[2]。

1.2. 论文研究背景

本文的主要工作是基于北大众志 PKUnity－3（SK）系统芯片设计并实现由时钟生成模块和功耗管理模块组成的功耗管理部件。本节简要介绍该系统芯片。

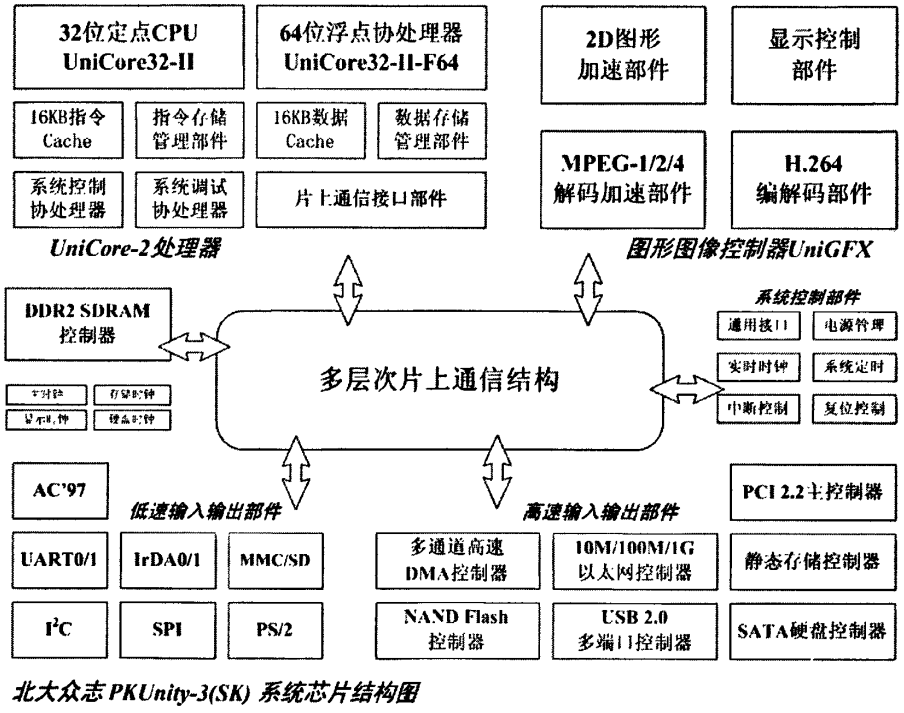


图 1-1 PKUnity-3（SK）系统芯片的结构框图

PKUnity－3（SK）系统芯片是应用于北大众志第三代网络计算机的高性能SoC产品。如图 1-1 所示，该芯片采用北大众志 UniCore－2 微处理器核心，基于第二代具有完全自主知识产权的 UniCore32 体系结构技术设计，片内集成具有8级流水线的第二代定点微处理器核，包括 32KB 一级 CACHE 系统。

该系统芯片中共有两条总线：UniBus64（64 位主存专用通道）和 MemBus32（32 位存储系统总线）。为了充分利用 DDR 的带宽，提高 UniCore－2 与外部存储器的访问速度和吞吐量，在 PKUnity－3（SK）系统芯片中使用了 64 位主存专用通道（UniBus64）来连接 UniCore－2 处理器和 DDR2 SDRAM 控制器的一个从设备接口。另一条总线，32 位存储系统总线（MemBus32）承担寄存器配置交易以及外围模块访问存储器的交易。两条总线均采用标准 AHB2.0 总线规范。

该系统芯片集成 533MHz 数据率的 DDR2 SDRAM 存储控制器、静态存储控制器、PCI2.2 桥接器、10M/100M/1000M 以太网控制器、多端口 USB 2.0 OTG 控制器、SATA 硬盘控制器、MPEG 视频解码引擎，2D 图形加速器等部件。

该产品主要应用于高性能网络计算机终端、网络安全设备、IPTV 终端设备和数字家电系统，是一款即可适应低端桌面又能符合高端嵌入式应用的高性能计算平台[3]。

1.3. 论文主要工作和组织结构

1.3.1. 论文主要工作

本文的主要工作包括：

1．分析 PKUnity-3（SK）系统芯片的时钟需求，设计系统的时钟结构，完成时钟生成模块的 RTL 实现并以减小时钟偏差为目标对时钟生成模块进行逻辑优化，为与时钟控制相关的功耗管理技术的实现提供了基础。

2．基于 PKUnity-3（SK）系统芯片的时钟结构，实现动态变频技术，允许系统根据工作负荷量动态地调整时钟频率，从而降低芯片的动态功耗。

3．设计功耗管理模块的软件可编程寄存器与状态机，通过对功能模块的时钟信号以及系统芯片的全局时钟电路进行控制，允许系统根据工作负荷量在全集工作、子集工作、处理器空闲、系统休眠几种工作模式中切换。

4．重新设计时钟生成模块。新的时钟生成模块牺牲了一些灵活性，设计复杂度也有所提高，但是具有更为平衡的电路结构，为后端实现提供了一个较为理想的起点。

1.3.2. 论文的组织结构

本文首先介绍了 PKUnity－3（SK）系统芯片和基于这一芯片的系统级功耗管理的选题背景，后续章节的主要内容为：

第 2 章：简单介绍了功耗管理的基本概念以及常用的功耗管理技术。

第 3 章：首先介绍了时钟生成模块的设计，这既包括时钟生成模块的总体结构和各子模块的 RTL 实现，也包括为了实现时钟信号之间的平衡而对时钟生成模块进行的逻辑优化。随后，介绍了如何对于这一时钟生成模块应用动态变频技术。根据待变频时钟的不同地位，动态变频的实现形式也有所不同，一为硬件（状态机）控制的动态变频，一为软硬件协同控制的动态变频。

第 4 章：介绍了时钟门控技术在 PKUnity－3（SK）系统芯片中的应用。包括用 EDA 工具在寄存器堆级别插入时钟门控逻辑，通过软件可编程寄存器对各模块的输入时钟进行门控，以及通过功耗管理模块（Power Manager）控制系统工作模式，进行系统级的时钟门控。

第 5 章：介绍了对时钟生成模块的优化。为了在前端设计阶段就尽量消除各时钟信号之间的时钟偏差，本章将时钟生成模块的三级时钟结构压缩为一级，使时钟路径更为平衡。

第 6 章：总结本文的工作，分析目前 PKUnity－3（SK）系统芯片所采用的功耗管理技术的局限性，提出了后续的工作设想。

第 2 章 功耗管理的基本理论

2.1. CMOS 电路的功耗组成

功耗由动态功耗和静态功耗两部分组成。动态功耗是指电路中的信号值发生变化所引起的能量消耗，而静态功耗是指信号的值没有发生变化时电路所消耗的能量。

2.1.1. 动态功耗

动态功耗包括由充放电电容引起的功耗和由直接通路电流引起的功耗，前者称为翻转功耗（switching power），后者称为短路功耗（internal power）。

动态功耗的首要来源是翻转功耗（switching power），即逻辑门的输出逻辑值改变时，Vdd 对该输出电容充电所需要的能量。

如图 2-1 所示，每当电容C_L通过 PMOS 管充电时，它的电压从 0 升至V_{DD}，此时从电源吸取了一定数量的能量。该能量的一部分消耗在 PMOS 器件中，而其余则存放在负载电容上。在由高至低的翻转期间，这一电容被放电，于是存放的能量被消耗在 NMOS 管中。

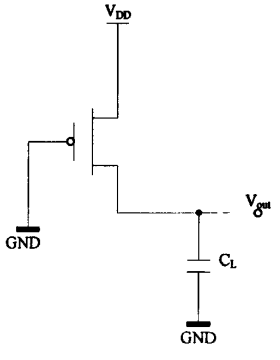


图 2-1 翻转功耗示意图

考虑由低至高的翻转。假设输入波形具有为零的上升和下降时间，即 NMOS 和 PMOS 器件不会同时导通。在这一翻转期间，从电源中取得的能量为$C_L V_{DD}^2$。其中，$C_L V_{DD}^2/2$的能量在充电阶段存放在电容C_L上，$C_L V_{DD}^2/2$的能量被 PMOS

管消耗。在放电阶段，电荷从电容上移去，存放在电容上的能量（$C_L V_{DD}^2/2$）消耗在 NMOS 器件中。总之，每一个开关周期（由 L→H 和 H→L 翻转组成）都需要一个固定数量的能量，并且该能量与 PMOS 和 NMOS 器件的尺寸无关。如果器件每秒通断 $f_{0\to1}$ 次，则功耗等于：$P_{dyn} = C_L V_{DD}^2 f_{0\to1}$。

其中，$f_{0\to1}$ 称为开关活动性（switching activity）。一个电路的开关活动性与输入信号的本质及统计特性有关：如果输入信号保持不变，则不会发生任何开关，于是动态功耗为零。反之，迅速变化的信号会引起许多次开关及功耗。此外，整个电路的拓扑结构以及要实现的功能也是影响开关活动性的因素。

上述翻转功耗的公式也可以写为：$P_{dyn} = C_L V_{DD}^2 f_{0\to1} = C_L V_{DD}^2 P_{0\to1} f = C_{EFF} V_{DD}^2 f$。

其中，f 代表输入发生变化事件的最大可能的速率（一般为时钟速率），而 $P_{0\to1}$ 是时钟变化事件在该门的输出端引起 0→1 变化事件的概率。$C_{EFF} = P_{0\to1} C_L$ 称为等效电容，它代表了每时钟周期发生开关的平均电容。

可以看到，降低 V_{DD} 对功耗的影响呈二次方关系。因此，降低电源电压是广泛应用的一种低功耗设计技术。此外，如果减小电源电压引起的性能降低不能被接受，减少等效电容也可以达到降低功耗的目的。这可以通过减少它的两个方面来实现：实际电容及开关活动性。

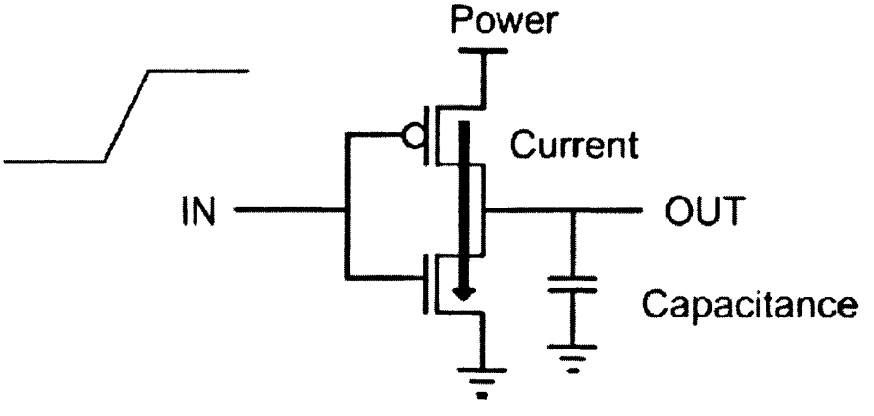


图 2-2 短路功耗示意图

在上述关于翻转功耗的分析中，曾假设输入波形具有为零的上升和下降时间。在实际设计中，这一假设是不成立的。如图 2-2 所示，输入信号非无穷大的

斜率造成了开关过程中V_{DD}和GND之间在短期内出现一条直流通路，此时NMOS和PMOS管同时导通。假设所形成的电流脉冲近似为三角形并且反相器的上升和下降响应是对称的，则每个开关周期消耗的能量为：$E_{dp} = t_{sc}V_{DD}I_{peak}$。平均功耗为：$P_{dp} = t_{sc}V_{DD}I_{peak}f = C_{sc}V_{DD}^2 f$。

与翻转功耗类似，直流通路引起的功耗与开关活动性成正比。t_{sc}代表两个器件同时导通的时间。I_{peak}出器件的饱和电流决定，直接正比于晶体管的尺寸，并与输入和输出斜率之比密切相关。

短路功耗可以通过使输入和输出信号的上升/下降时间匹配来达到最小。在整个电路层次上，这意味着所有信号的上升/下降时间应当保持在一定范围内不变。

此外，电源电压降低时短路功耗的影响将变小。极端情形下，当$V_{DD} < V_{Tn} + |V_{Tp}|$时，PMOS与NMOS器件不会同时导通，短路功耗完全消除。当阈值电压以比电源电压低的速率下降时，短路功耗在深亚微米工艺中将变得较不重要[1]。

对于一个给定的反相器尺寸，当负载电容太小时，动态功耗主要来自短路功耗。对于非常大的负载电容值，几乎所有的功耗都用来充电和放电负载电容，大部分动态功耗与翻转功耗有关，而只有很小一部分出自短路功耗。在本文对动态功耗的分析中，主要关注翻转功耗。

2.1.2. 静态功耗

一个电路的静态功耗（也称为稳态功耗）可用下列关系来表示：$P_{stat} = I_{stat}V_{DD}$。式中，$I_{stat}$是在没有开关活动存在时在电源两条轨线之间流动的电流。

理想情况是CMOS反相器的静态电流为零，因为PMOS和NMOS器件在稳态工作状况下决不会同时导通。实际情况是，总会有泄漏电流流过位于晶体管源（或漏）与衬底之间的反相偏置的二极管结。

结的泄漏电流是由热产生的载流子引起的。它们的数值随结温而增加，并且呈指数关系。为了保证小的泄漏电流，需要保持电路的工作温度较低，即限制电路的功耗或使用能支持有效散热的封装。

泄漏电流的一个越来越突出的来源是晶体管的亚阈值电流。一个 MOS 管，即使在V_{GS}小于阈值电压时也可以有一个漏－源电流。阈值电压越是接近 0V，则在$V_{GS}=0V$时的泄漏电流越大，因而静态功耗也就越大。为了抵消这一效应，器件的阈值电压一般应当保持足够高。标准工艺的特征值V_T从未小于 0.5~0.6V，有时甚至还相当大。

随亚微米工艺尺寸的缩小，电源电压降低。如果在降低电源电压的同时保持阈值电压不变，性能将严重下降，特别是当V_{DD}接近于$2V_T$时。解决这一性能问题的一个方法是同时降低这一器件的阈值电压。从而减小由降低电源电压造成的性能损失。但是，阈值电压的最低值是由所允许的亚阈值漏电流的数量所决定的。因此，阈值电压的选择代表了在性能和静态功耗之间的权衡取舍。随着工艺尺寸继续缩小，电源电压继续降低，进而迫使阈值电压更为降低，亚阈值导电将成为功耗的主要来源[1]。

综上所述，CMOS 器件的总功耗可以表示成三部分的和：

$$P_{tot}=P_{dyn}+P_{dp}+P_{stat}=(C_LV_{DD}^2+V_{DD}I_{peak}t_{sc})f_{0\to1}+V_{DD}I_{leak}$$

在典型的 CMOS 电路中翻转功耗是占主导地位的因素。短路功耗可以通过细心的设计控制在限定范围之内，因此不应当成为问题。在 130nm 工艺下，泄漏电流引起的静态功耗可以忽略，但对于 65nm 以下的工艺，情况会有所不同。

2.2. 功耗管理的基本技术

常见的功耗管理技术可以分为两类，一类是在设计时间实现的，比如选择电源电压、逻辑优化；一类是在运行时间动态改变的，比如时钟门控、动态电源电压调节、待机模式下切断电源[1]。本节将逐一介绍这些功耗管理技术。

2.2.1. 选择电源电压

降低电源电压能够使翻转功耗呈二次方地降低，因而是降低动态功耗的最有效的方法。在过去的 15 年里，随着 CMOS 器件制造工艺的进步，电源电压V_{DD}已经由 5V 降低到 3.3V，以及 2.5V，进而降低到 1.2V。在 2009 年，高性能设备将

使用 1.0V 的电源电压，而低功耗设备甚至使用 0.8V 的电源电压。

不过，随着电源电压的降低，CMOS 门的延时增加，导致电路的性能下降。解决这一问题的一个方法是同时降低器件的阈值电压，从而减小由降低电源电压造成的性能损失。此外，也可以通过逻辑和结构的优化等其他方法来补偿速度的损失。

一种方法是采用并行结构。将两个相同的功能单元并联，每个单元可以以原来一半的速率工作，同时又能保持原有的数据流通量。所以，电路的速度要求放宽，电源电压可以被降低。并行结构并不是补偿性能损失的惟一方法。运用流水线也可以达到同样的目的，这里不再赘述。显然，这两种方法都是牺牲面积来换取低功耗。

降低电源电压虽然非常简单直接，但是它在同等程度地降低所有逻辑门的功耗的同时，也同等程度地增加了它们的延时。更好的办法是有选择地降低某些门上的电源电压，即采用多种电源电压。一方面，电路的工作速度由延迟较大的关键路径决定，因此，降低较快路径上的门和完成计算较早的门的电源电压不会影响电路的性能。另一方面，降低驱动大电容的门的电源电压，在增加同样延迟的时候能够得到最大的收益。

多电源电压在今天的集成电路中已经常采用。为兼容起见，I/O 配备了一个单独的电源电压，逻辑核心则采用较低电压的电源。可以把这一方法扩展用于降低电路的功耗。例如，每一个模块都可以从两个（或更多个）电源中选择一个最合适的电压，甚至可以逐个门来指定多个电源电压[1][4]。

2.2.2. 逻辑优化

等效电容是实际电容和开关活动性的乘积，可以通过优化电路逻辑和结构降低活动性而无需以性能为代价。

例如，如果几个操作通过多路开关共享单个硬件单元，不但增加了实际电容，还增加了开关活动性，从而导致功耗增加。为了实现低功耗的设计目标，应避免过度复用资源，通过资源分配降低开关活动性。

又如，动态故障或毛刺（glitching）是加法器和乘法器等复杂结构中功耗的主要来源。在这样的一些结构中，当信号路径的长度之间存在很大差别时就有可

能引起寄生的瞬态过程。为了减少毛刺引入的开关活动性，应选择具有均衡信号路径的结构[1]。

2.2.3. 时钟门控

时钟网络所消耗的功率在电路的动态功耗中占有相当大的比例。首先，时钟树上的时钟缓冲器具有最高的翻转概率。其次，为了保证尽量小的时钟延迟，一般选择驱动能力大的缓冲器作为时钟缓冲器。并且，对于一个触发器，即使输入输出的逻辑值未发生变化，内部节点也将由于时钟信号的变化而产生翻转，从而产生动态功耗。一般来说，时钟信号的缓冲器所消耗的功率能够占到电路的总动态功耗的50%以上。

直观地，在电路不需要时钟信号时将时钟信号关断可以降低这部分动态功耗，这一技术称为时钟门控。

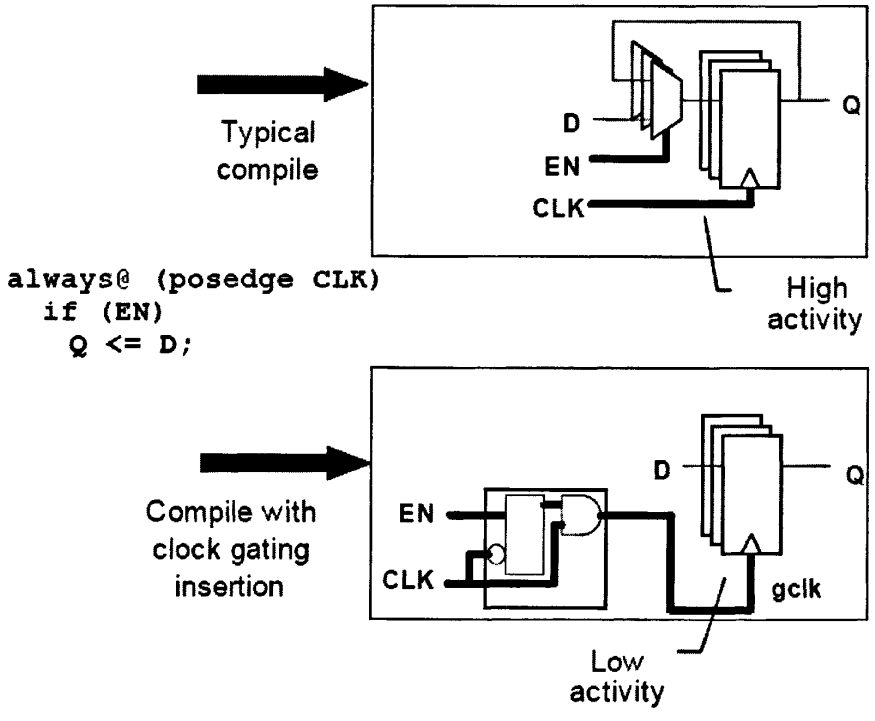


图 2-3 采用时钟门控技术的寄存器实现

如图 2-3 所示，如果某寄存器的更新依赖于使能信号 EN，可以将此 EN 信号作为多路选择器的选择信号，也可以将此 EN 信号作为时钟信号的门控使能信号。两种实现方法具有完全相同的逻辑功能，但后者所消耗的动态功耗更小[4][5]。

2.2.4. 动态电源电压调节

固定地降低电源电压是以性能为代价来减少每一操作的能耗并延长电池的寿命。这一性能方面的损失常常不能被接受，特别是在以等待时间（latency）为约束条件的应用中尤为明显。但我们注意到并不是持续不断地要求最高性能，因此仍然可以设法使功耗显著降低。例如，考虑一个便携式应用中的通用处理器，如笔记本电脑、电子管理器和手机。这类处理器所执行的计算功能主要分为三类：高强度计算任务、低速功能和闲置模式操作。高强度计算和短等待时间的任务要求处理器满流通量计算以满足实时约束。MPEG 视频和音频压缩器就是这样的例子。而如文本处理、数据输入和存储备份等低流通量和长等待时间的任务则在宽松得多的限定时间内完成工作，因此只需要微处理器最大流通量的一小部分。这时，提早完成计算没有任何回报，相反如果一个任务完成太早还会被认为浪费了能量。最后，便携式处理器有很大一部分时间处于闲置状态，等待着使用者的操作或外部的启动。总之，对一个移动处理器所要求的计算流通量和等待时间在不同的时刻会有很大的不同。

当执行较低负荷的工作时降低时钟频率能够降低功率，但却不能节省能量——因为每个操作仍然在较高的电压下进行。但如果同时降低电源电压和频率则能量就可以降低。为了在高负荷时保持所要求的流通量和低负荷时的最小能量，电源电压和频率都必须根据当时正在执行的应用要求动态地变化。这一技术称为动态电压调节（dynamic voltage scaling，DVS）。它的工作原则是一个功能应当总是在满足时间约束的最低电源电压下工作[1][6][7]。

2.2.5. 动态阈值调节

与动态改变电源电压相类似，动态调节晶体管的阈值电压也非常有吸引力。对要求低等待时间的计算，应当把阈值降低到它的最小值；对于低速计算可以增加阈值；而对于待机模式则应当把它设置在最高可能的值，以使漏电电流最小。

与动态电源电压调节技术一样，可变阈值电压技术也以反馈环为基础，它可以设置为实现以下各个目的：

- 可以降低待机模式时的漏电

- 可以在电路正常工作期间补偿芯片各处的阈值偏差
- 可以根据性能要求调整电路的数据流通量以降低工作功耗和漏电功耗

2.2.6. 待机模式下切断电源

闲置模式代表了动态功耗管理空间中的一个极端情况。由于没有发生任何工作切换，所有的功耗都来自漏电——假设合适的时钟和输入的门控选通在适当的位置上。为了降低待机期间漏电，可以用大尺寸的休眠晶体管在电路处于休眠状态时切断电源轨线。如图所示，这可以通过只在电源线上加一个电源开关来实现，或者在电源线和地线上都加上开关。

与简单的时钟选通不同，关断电源会删除功能块内寄存器的状态。在某些应用中，这是可以接受的，比如当再次唤醒一个功能块后执行的是一个全新的任务时。否则，如果需要保存原来的状态，可以把所有的寄存器都连至没有门控的电源轨线上，也可以利用操作系统把所有寄存器的状态都保存到一个非易失性存储器中。

2.3. 小结

本章介绍了功耗管理的基本理论。首先介绍了动态功耗与静态功耗的产生原因，其次介绍了常用的功耗管理技术，这些技术可以分为面向动态功耗的技术和面向静态功耗的技术，也可以分为设计时间的功耗管理技术和运行时间的动态功耗管理技术。其中，运行时间动态采用的面向动态功耗的功耗管理技术是本文讨论的重点。

第 3 章 动态变频技术的设计与实现

电路的动态功耗的首要来源是翻转功耗，即逻辑门的输出逻辑值改变时，电源对该输出电容充电所需要的能量。一方面，该功耗分量与时钟频率成线性关系，随时钟频率的降低而降低；另一方面，时钟频率降低后，电源电压又可随之降低，导致功耗以二次方的速度进一步降低。因此，根据系统的工作负荷量动态地调整时钟频率与电源电压是一种有效的低功耗设计技术[8][9][10][11]。SK 系统芯片采用了动态变频（Dynamic Frequency Scaling）技术，至于动态变电压（Dynamic Voltage Scaling）技术，由于设计难度和复杂度较高，暂时未予实现。

动态变频技术的实现与系统的时钟结构密切相关，同时，由于时钟偏差是限制数字系统的性能的重要因素，因此，时钟生成模块的设计与实现是系统芯片设计的一个突出难点和重点。本章介绍 SK 系统芯片中的时钟生成模块以及基于该时钟生成模块的动态变频技术的实现。3.1 节介绍时钟生成模块的 RTL 设计，包括其总体结构和各子模块的具体设计；3.2 节介绍以减小时钟偏差为目标对时钟生成模块进行的逻辑优化；3.3 节介绍不同级别的时钟信号的动态变频的具体实现；3.4 节总结本章内容。

3.1. 时钟生成模块的 RTL 设计

时钟生成模块的设计与实现是系统芯片设计的一个突出难点和重点。首先，系统芯片中往往集成了十几个功能模块，具有异常复杂的时钟需求；其次，必须保证时钟偏差尽量小，避免从根本上限制系统的性能；并且，时钟分布网络的结构直接影响到动态变频技术的实现方式。本节将详细介绍 SK 系统芯片中时钟生成模块的 RTL 设计[12][13][14]。

3.1.1. 时钟生成模块的总体结构

SK 系统芯片基于 AMBA 总线架构，集成了 UniCore－2 CPU、DDR2 SDRAM 控制器、十/百/千兆以太网控制器、H.264 编/解码器、图形图像控制器等多个功能模块，其结构框图如图 1-1 所示。

UniCore－2 处理器时钟 mclk 的频率配置范围为 125M～1GHz。64 位高速总线时钟 bclk64 同步于处理器时钟，并且频率不高于处理器时钟的频率；32 位低速总线时钟 bclk32 也同步于处理器时钟，并且频率不高于 250MHz。显然，两个总线时钟都可以由处理器时钟分频得到。一般地，bclk64 为处理器时钟的一分频时钟，而 bclk32 为处理器时钟的二分频或者三分频时钟。

除此以外，各功能模块也有复杂多样的时钟需求。

首先，32 位低速总线上的若干功能模块都需要同步于总线时钟 bclk32 的时钟作为核心工作时钟，但它们的频率需求又各不相同。比如，NAND Flash 控制器要求其工作时钟 nand_clk 的频率配置范围为 6～50MHz，MPEG 视频解码引擎要求其工作时钟 ve_clk 的频率配置范围为 30～150MHz，H.264 编码器和解码器分别要求其工作时钟 he_clk 和 hd_clk 的频率在 60～125MHz 和 60～100MHz 的范围内可配。直观地，这些时钟都可以由总线时钟 bclk32 分频得到，不同的频率需求对应着不同的分频比。具体地说，NAND Flash 控制器时钟 nand_clk 是总线时钟 bclk32 的 3～16 分频，MPEG 视频解码引擎时钟 ve_clk 是总线时钟 bclk32 的 3～16 分频，H.264 编码器和解码器时钟 he_clk 和 hd_clk 分别是总线时钟 bclk32 的 2～3 分频和 3～5 分频。

此外，在上述功能模块的设计中，核心工作时钟和总线时钟被视为同步不同频的时钟。在 UniCore－2 微处理器的设计中，处理器时钟和高速总线时钟以及低速总线时钟也被视为同步不同频的时钟。为了保证跨越这种同步时钟域的信号交互的正确性，时钟生成模块还需要对每一对这样的时钟生成相位指示信号，指示高速时钟的哪一个上升沿是与低速时钟的上升沿对齐的。

类似地，64 位高速总线上的二维图形加速器的核心工作时钟 ge_clk 也可以由相应的总线时钟 bclk64 分频得到。不同的是，在二维图形加速器的设计中，核心工作时钟 ge_clk 和总线时钟 bclk64 被视为完全异步的时钟，所有跨越这两个时钟域的交互路径都是按照跨异步时钟域设计方法进行设计的，所以，时钟生成模块只需对 bclk64 分频生成 ge_clk，而不需要为这对时钟信号生成相位指示信号。

根据上面的分析，本文设计了如图 3-1 所示的时钟生成模块。

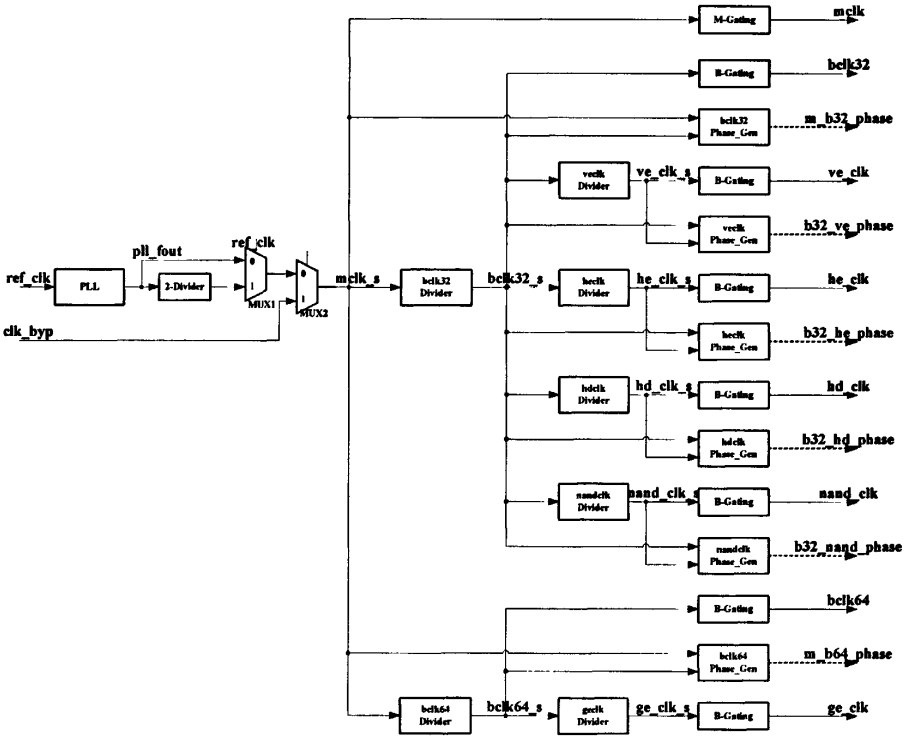


图 3-1 时钟生成模块的结构框图

图 3-1 中，mclk_s 是第一级时钟源，它可以来自 PLL 的输出时钟 pll_fout，也可以来自片外晶振时钟 clk_byp。SK 系统芯片中使用的 PLL（PG13A1LV3）的输出频率范围为 250M～1GHz，为了方便低频调试，在以 PLL 的输出时钟为 mclk_s 的来源时，除了可以选择该输出时钟作为 mclk_s 以外，也允许选择该输出时钟的二分频时钟作为 mclk_s。

bclk32_s 与 bclk64_s 是第二级时钟源，均由 mclk_s 分频得到。同时，相位生成器将根据快时钟 mclk_s 与慢时钟 bclk32_s/bclk64_s 的分频关系生成 mclk_s 相对于 bclk32_s/bclk64_s 的相位指示信号。

其它时钟（ve_clk_s、he_clk_s、hd_clk_s、nand_clk_s、ge_clk_s）是第三级时钟源，由 bclk32_s 或 bclk64_s 分频得到。同样地，相位生成器将根据 bclk32_s 或 bclk64_s 与第三级时钟的分频关系生成它们之间的相位指示信号。

最后，上述各时钟源（以_s 结尾的时钟信号）经过时钟门控电路后输出，作为处理器、总线以及各功能模块的输入时钟信号。时钟门控电路有两种，控制处理器时钟的时钟门控电路 M-Gating 以 mclk_en 为门控使能信号，控制总线时钟以及模块工作时钟的时钟门控电路 B-Gating 以 bclk_en 为门控使能信号。做这样

的区分主要是为了在应用时钟门控技术进行功耗管理时具有更大的设计空间。比如，将处理器时钟与总线时钟的控制权分离，使 SK 系统芯片可以定义 IDLE 与 SLEEP 两种低功耗工作模式，根据系统的工作情况选择合适的工作模式，最大限度地节省系统功耗。

3.1.2. 时钟分频器的 RTL 设计

为了满足各种频率需求，本文设计了支持 1～16 分频的时钟分频器 Divider。

不管是偶数分频还是奇数分频，都可以简单地用 Moore 状态机实现。不过，对于奇数分频，这种实现方法无法保证输出时钟具有 50%的占空比。为了得到 50%占空比的分频时钟，本文采用了正交时钟异或的分频器设计方法[15]。

这种分频器的电路结构如图 3-2 所示。

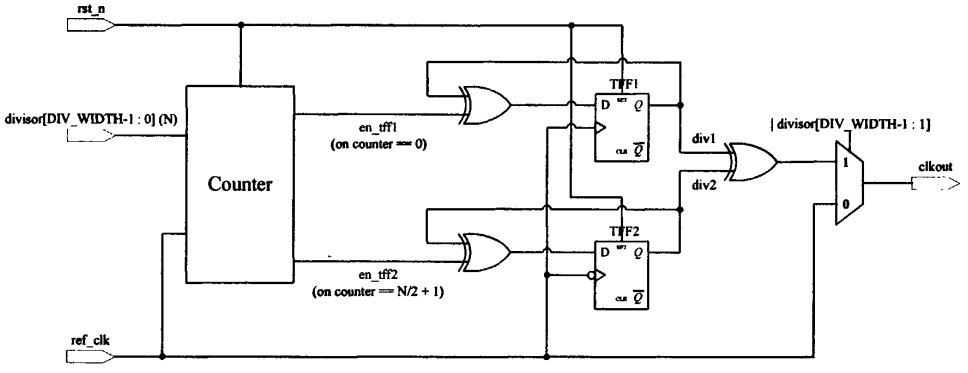


图 3-2 时钟分频器的结构框图

当分频系数 N 为 1 时，目标频率即输入时钟的频率，多路选择器将选择把输入时钟 ref_clk 输出，这样，输出时钟 clkout 与输入时钟 ref_clk 同频同相。

当分频系数 N 为大于 1 的奇数时，令计数器在输入时钟 ref_clk 的驱动下从 0 到 N－1 循环计数。当计数到 0 时， 将 T 触发器 TFF1 的翻转使能信号 en_tff1 置 1；当计数到 N/2+1 时，将 T 触发器 TFF2 的翻转使能信号 en_tff2 置 1。其中，TFF1 以输入时钟 ref_clk 的上升沿为触发沿，而 TFF2 以输入时钟 ref_clk 的下降沿为触发沿。这样，TFF1 和 TFF2 两个 T 触发器分别输出频率为目标频率的二分之一的时钟信号 div1 与 div2，并且这两个时钟信号的相位差为 90 度。最后，将这两个时钟信号异或起来，就得到了具有 50%占空比的输出时钟 clkout。该时钟的频率是输入时钟 ref_clk 的频率的 1/N，并且与输入时钟 ref_clk 同相。以三

分频为例，波形图如图 3-3 所示。

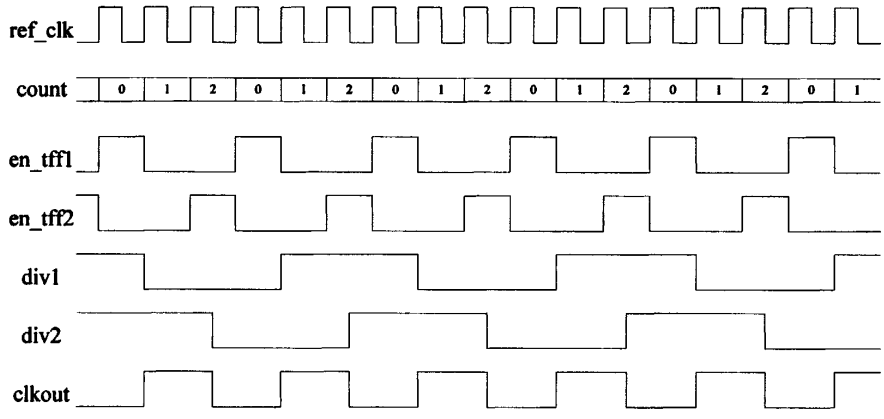


图 3-3 N=3 时分频器内部信号的波形图

当分频系数 N 为偶数时，令计数器在输入时钟 ref_clk 的驱动下从 0 到 N/2－1 循环计数。同样地，当计数到 0 时，将 T 触发器 TFF1 的翻转使能信号 en_tff1 置 1；当计数到 N/2+1 时，将 T 触发器 TFF2 的翻转使能信号 en_tff2 置 1。这样，TFF1 将输出目标频率的时钟信号 div1，而由于计数器不可能计数到 N/2+1，TFF2 的翻转使能信号 en_tff2 将恒为 0，导致 TFF2 输出的时钟信号 div2 恒为高电平。将这两个时钟信号异或起来，异或门的输出信号与时钟信号 div1 完全相同，也就是目标频率的时钟信号。该时钟的频率是输入时钟 ref_clk 的频率的 1/N，并且与输入时钟 ref_clk 同相。以四分频为例，波形图如图 3-4 所示。

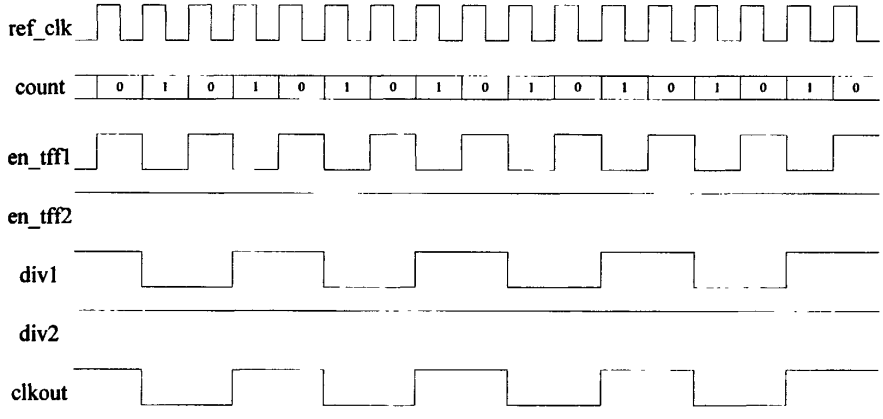


图 3-4 N=4 时分频器内部信号的波形图

3.1.3. 相位生成器的 RTL 设计

频率不同但是存在固定相位关系的两个时钟域之间的信号交互，属于同步时钟域之间的信号交互。这种跨时钟域路径不需要使用同步电路来避免亚稳态的出现，但是需要使用相位指示信号对信号的传递时机进行控制[16][17][18]。

相位指示信号 phase 的含义是：假设高速时钟的频率为低速时钟频率的整数倍，并且两个时钟的相位差为零，那么，在高速时钟域中采样 phase 信号，如果采样值为高电平，意味着高速时钟的当前上升沿与低速时钟的上升沿是对齐的。

当信号从高速时钟域进入低速时钟域时，必须保证高速信号的值在低速时钟采样之前不会发生多次改变。否则，低速时钟采样到的值可能已不是我们想要传递的值。为了解决这个问题，要求高速时钟域的最后一级寄存器仅在采样到高电平的相位指示信号时才改变其输出值。而当信号从低速时钟域进入高速时钟域时，由于从低速时钟域出发的时序路径往往是延迟较大的长路径，跨时钟域信号难以在一个高速时钟周期内到达高速时钟域的采样寄存器，因此，要求高速时钟域的第一级寄存器仅在采样到高电平的相位指示信号时才去采样来自低速时钟域的信号值。这样，跨时钟域信号只需在一个低速时钟周期内到达高速时钟域的采样寄存器即可。

以总线时钟 bclk32 和 NAND Flash 控制器的工作时钟 nand_clk（由总线时钟 bclk32 分频得到）为例，分频比为 3 时相位指示信号的波形以及两个时钟域之间的采样关系如图 3-5 所示。在 t1、t2、t3、t4 与 t5 时刻，两个时钟的上升沿对齐，高速时钟域的寄存器可以更新输出值或者采样输入数据值。

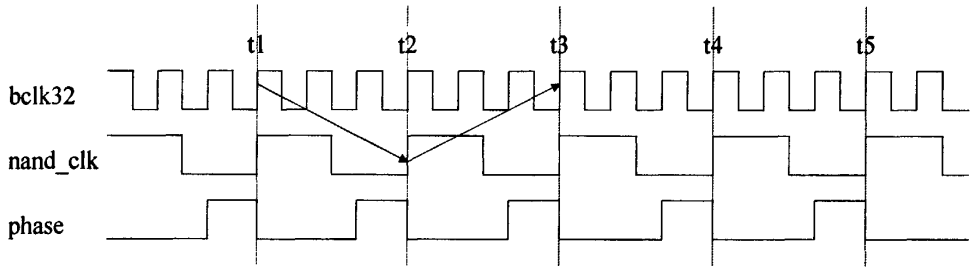


图 3-5 相位指示信号的波形与同步时钟域之间的采样关系

对于 SK 系统芯片，处理器时钟 mclk 与高速总线时钟 bclk64 以及低速总线时钟 bclk32 之间，低速总线时钟 bclk32 与 NAND Flash 控制器时钟 nand_clk、

H.264 编码器时钟 he_clk、H.264 解码器时钟 hd_clk 以及 MPEG 视频解码引擎时钟 ve_clk 之间都需要这种相位指示信号。为此，本文设计了用来为同步时钟对生成相位指示信号的相位生成器（Phase_Gen）。

相位生成器根据输入的高速时钟 fast_clk、低速时钟 slow_clk 以及低速时钟相对于高速时钟的分频系数 divisor，生成上文定义的相位指示信号 phase。当 divisor 为 1 时，两个时钟的频率相同，相位指示信号恒为高电平。当 divisor 大于 1 时，当同步于低速时钟 slow_clk 的复位信号 rstn_sync_sclk 被撤消后，令计数器在高速时钟 fast_clk 的驱动下从 divisor-1 到 0 循环计数，用高速时钟 fast_clk 的上升沿采样计数器的计数值，如果计数值为 1，则输出高电平的相位指示信号，否则，输出低电平的相位指示信号。此设计的电路结构如图 3-6 所示。

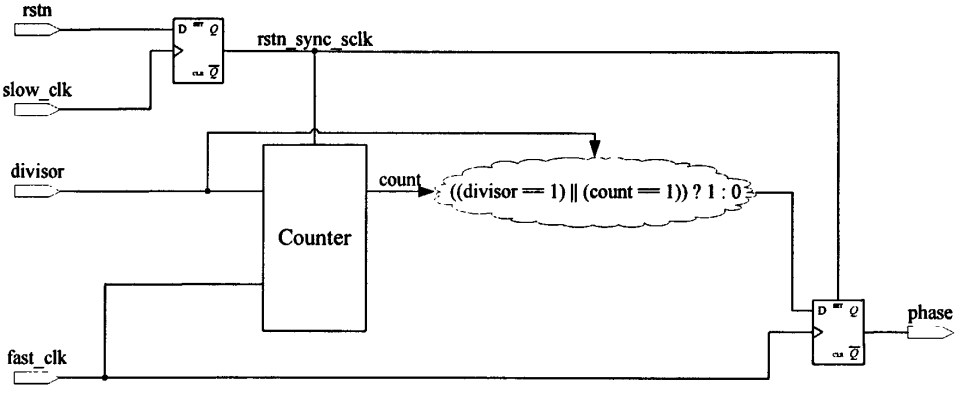


图 3-6 相位生成器的结构框图

以 divisor=3 为例，波形图如图 3-7 所示。

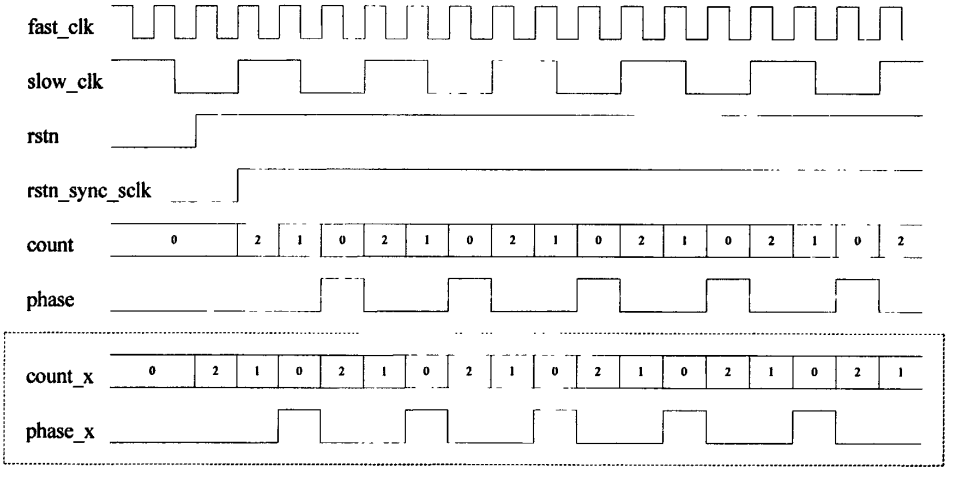


图 3-7 N=3 时相位生成器内部信号的波形图

值得注意的是，计数器的复位信号必须使用同步于慢速时钟的复位信号 rstn_sync_sclk，而不应使用同步于快速时钟的复位信号 rstn。如果计数器在 rstn 撤消的时刻就开始循环计数，生成的相位指示信号是不符合其定义的，图 3-7 中的 count_x 与 phase_x 示出了这种错误的波形。

3.2. 时钟生成模块的逻辑优化

在数字集成电路的设计中，时钟信号的作用日益重要。随着对芯片速度要求的提升，对时钟频率的要求也愈来愈高，而同步时钟信号的平衡对于保证电路的时序性能至关重要，因此，系统芯片设计中对时钟生成模块的输出时钟的质量有较为严格的要求。为了尽量消除时钟偏差、保证时钟生成模块的输出时钟的平衡，本文使用 Design Compiler、Astro、PrimeTime 等 EDA 工具，对时钟生成模块的逻辑结构进行了优化，本节将详细介绍这些逻辑优化技术[19]。

3.2.1. 时钟偏差对系统性能的影响

表征时钟信号质量的指标有时钟频率、时钟偏差（clock skew）、频率波动（jitter）、上升和下降时间、噪声和毛刺等。在频率目标固定、时序路径优化余地有限的情况下，时钟偏差是制约电路时序性能的重要因素，考虑如图 3-8 所示的时序路径。

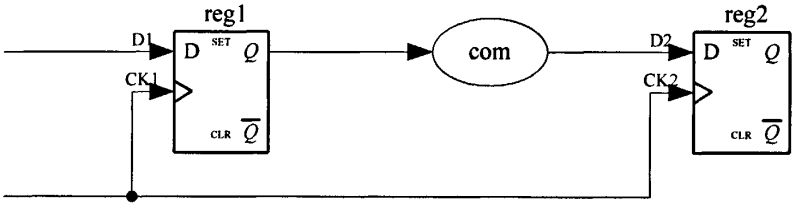


图 3-8 时钟偏差对电路时序性能的影响

如果时钟信号 CK1 与 CK2 之间不存在时钟偏差，则图 3-8 中的时序路径需满足：$T_{ck\to q}+T_{com}+T_{setup}\le T_{period}$。其中，$T_{ck\to q}$ 为寄存器 reg1 的内部延迟，T_{com} 为寄存器 reg1 与 reg2 之间的组合逻辑的延迟，T_{setup} 为寄存器 reg2 的建立时间要求，T_{period} 为时钟周期。如果时钟信号 CK1 与 CK2 之间存在时钟偏差 $T_{skew}=t_{ck2}-t_{ck1}$，

上式需修正为$T_{ck\to q}+T_{com}+T_{setup}\le T_{period}+T_{skew}$。在频率目标固定的情况下，如果时钟偏差$T_{skew}$为负，即 CK2 的触发沿$t_{ck2}$早于 CK1 的触发沿$t_{ck1}$，对组合逻辑的延迟的要求将变得更为严格。如果组合逻辑的延迟优化余地有限，则只能通过提高T_{period}，即降低时钟频率，来保证不发生建立时间违例（setup violation）。

根据上面的分析，减小时钟偏差是保证电路时序性能的重要手段。这既包括减小同一时钟树上各叶子节点之间的偏差，也包括减小两个同步时钟的时钟树的根节点之间的偏差。其中，同一时钟树上的延迟差异主要依赖 EDA 工具进行平衡。而由于各时钟树的根节点均来源于时钟生成模块的输出，因此时钟树之间的延迟差异直接取决于时钟生成模块的输出时钟之间的延迟差异。这样，平衡时钟延迟、降低时钟偏差就成为时钟生成模块设计和实现的重要目标。

在 3.1 节所述的时钟生成模块内部，主要存在以下几种类型的时钟偏差。

1）多时钟偏差。即各同步时钟之间的偏差。

2）沿偏差。同一时钟信号的上升沿和下降沿的不平衡称为沿偏差。如果设计中同时使用了一个时钟的上升沿和下降沿，并且上升沿触发的寄存器和下降沿触发的寄存器之间存在时序路径，则需要平衡该时钟信号的沿偏差。产生沿偏差的原因之一是逻辑单元的上升延迟和下降延迟不相等。以 Artisan 标准单元库中的反相器 CLKINVX1 为例，该反相器是上升延迟和下降延迟较为对称的逻辑单元，但是当负载很大时二者仍然相差 2～3%，其他逻辑门的上升延迟和下降延迟的差异甚至能够达到 50%。

3）路径偏差。当时钟信号由两条或两条以上的时钟路径汇聚而成时，这些路径称为重汇聚路径。如果重汇聚路径的结构不对称，将导致时钟信号产生路径偏差。

4)逻辑偏差。如果逻辑单元(如 Artisan 标准单元库中的异或门 CLKXOR2X1）的各输入端到输出端的延迟不对称，将导致时钟信号产生逻辑偏差。

针对上述几种类型的时钟偏差，本文采用反相器推前、异或门重构、寄存器复制、构建反相器链等多种技术对时钟路径的延迟进行平衡，消除了时钟生成模块的固有时钟偏差。

3.2.2. 面向时钟平衡的逻辑优化

时钟生成模块中，最为关键的子模块是时钟分频器 Divider，它包含了复杂的分频控制逻辑，并且存在时钟信号的重汇聚路径。时钟分频器的简化结构如图 3-9 所示，由于我们主要关注时钟信号的传播路径，因此图中省略了控制信号的生成逻辑。

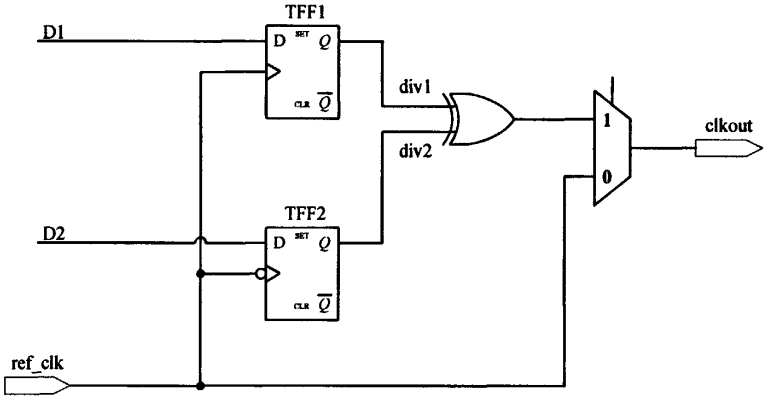


图 3-9 时钟分频器的简化结构框图

首先，TFF1 为上升沿触发的寄存器，而 TFF2 为下降沿触发的寄存器，它们具有不同的内部延迟（$T_{ck\to q}$），导致它们的输出时钟 div1 与 div2 之间存在偏差。这个偏差将使奇数分频时钟的占空比偏离 50%。为此，将时钟分频器做如图 3-10 所示的改进。

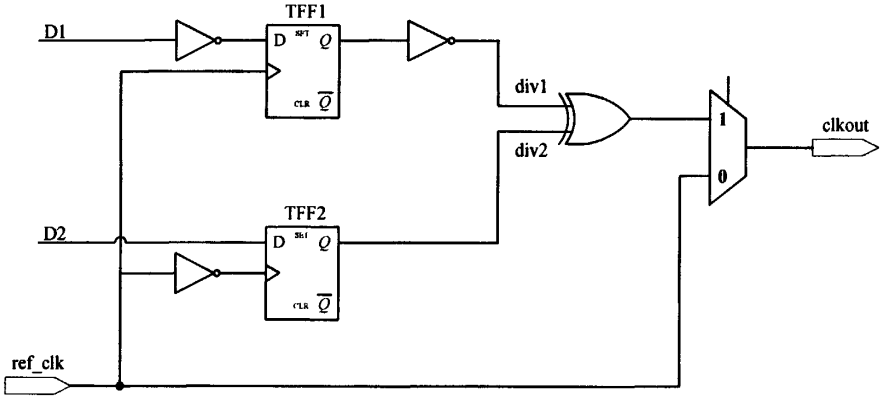


图 3-10 反相器推前后的时钟分频器

图 3-10 中，用上升沿触发的寄存器替换下降沿触发的寄存器，并在该寄存器的时钟输入端前插入一个反相器。这样，寄存器 TFF1 与 TFF2 的类型完全相

同，具有相同的内部延迟，并且 div2 这条时钟路径的逻辑行为与原设计一致。

由于在 div2 时钟路径上插入了反相器，导致 div1 与 div2 两条时钟路径出现较大的延迟差异。因此，在寄存器 TFF1 的数据输入端和输出端各插入一个反相器，在保证逻辑正确性的同时，平衡了 div1 与 div2 两条时钟路径上的延迟。

经过上述改进，可以保证到达异或门的 A、B 输入端的两个时钟信号的偏差基本被消除。但是，对于图 3-10 中的异或门，如果使用 Artisan 标准单元库中的异或门 CLKXOR2X1，A、B 输入端到输出端的延迟差异较大，并且一个输入端到输出端的延迟依赖于另一个输入端上的逻辑值；如果令逻辑综合工具自动使用简单逻辑门搭建异或逻辑，也往往导致 A、B 输入端到输出端的延迟存在差异。这种逻辑偏差将使奇数分频时钟的占空比偏离 50%。为此，手动搭建异或门如图 3-11 所示，其中的与非门选用了输入端到输出端的延迟基本平衡的与非门 NAND2X1。

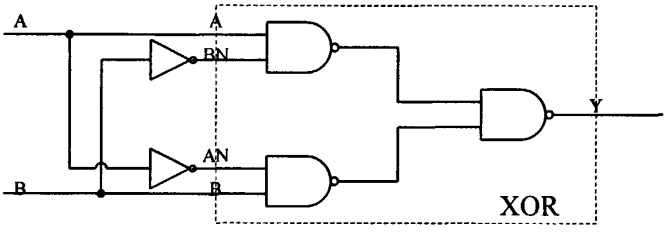


图 3-11 手动重构的平衡异或门

在如图 3-11 所示的重构的异或门中，存在两个反相器，这引入了新的延迟差异。为此，复制寄存器 TFF1 为 TFF1_n，复制寄存器 TFF2 为 TFF2_n，分别生成 AN 与 BN，从而保证到达异或门的 A、B、AN、BN 输入端的时钟信号的延迟基本平衡。图 3-12 示出了改进后的时钟分频器的结构。

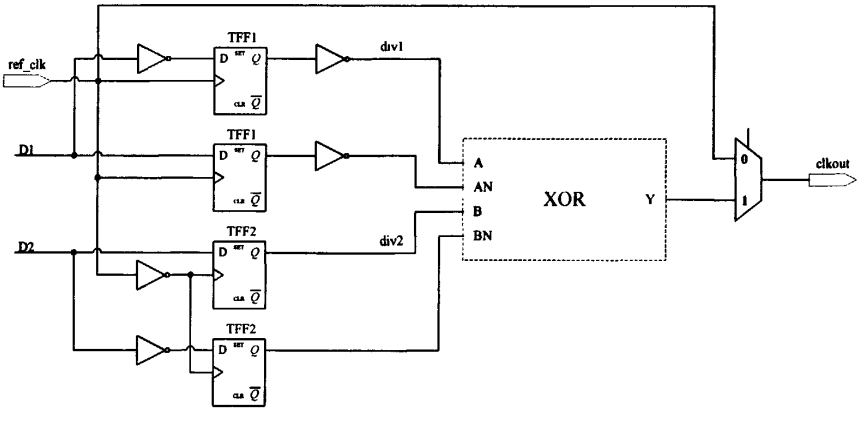


图 3-12 寄存器复制后的时钟分频器

下面考虑时钟生成模块的总体结构。如图 3-1 所示，从 mclk_s 到 mclk 不经历任何时钟分频器，从 mclk_s 到 bclk32/bclk64 经历 1 个时钟分频器，从 mclk_s 到 ve_clk、ge_clk 等第三级时钟则需经历 2 个时钟分频器。这样，mclk 与 bclk32、bclk64 之间，bclk32、bclk64 与 ve_clk、he_clk、hd_clk、nand_clk、ge_clk 之间的延迟差异较大，导致了较大的时钟偏差。为了实现这些同步时钟信号之间的平衡，需要在延迟较小的时钟路径上插入延迟元件。其中，延迟元件可以使用缓冲器，也可以使用反相器。由于反相器对可以通过沿互补来减小沿偏差，本文采用反相器链来提供特定的延迟，从而消除时钟偏差。以 mclk 与 bclk32 为例，插入反相器链后的电路结构如图 3-13 所示。

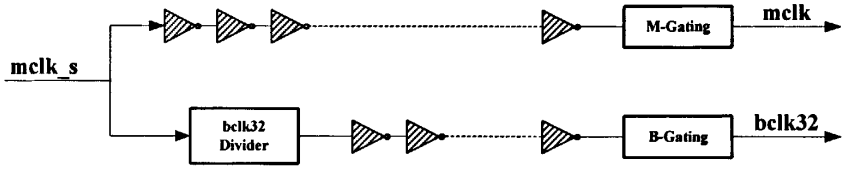


图 3-13 用反相器链调整不同级时钟之间的延迟差异

除了需要平衡同步时钟信号之间的偏差之外，时钟信号与相应的相位指示信号之间的偏差也需要平衡。仍以 mclk 与 bclk32 这对同步时钟为例，为了消除时钟信号与相位指示信号之间的偏差，应插入图 3-14 中阴影所示的反相器链。

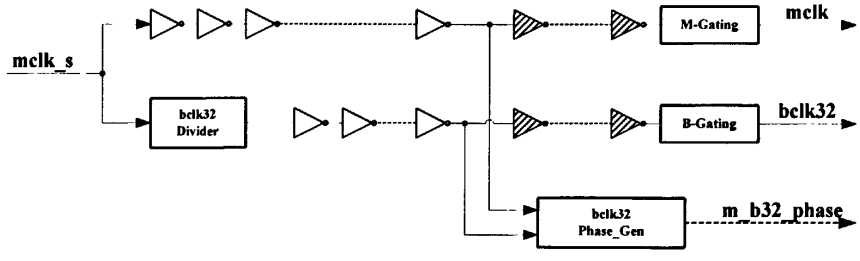


图 3-14 用反相器链调整时钟信号与相位指示信号之间的延迟差异

此外，考虑时钟分频器的两种工作模式。如图 3-2 所示，1 分频时，多路选择器将选择输入时钟 ref_clk 作为输出时钟；而 2～16 分频时，多路选择器将选择异或门的输出作为输出时钟。这两条时钟路径存在较大的延迟差异，也同样需要使用反相器链调整延迟。插入反相器链后的时钟分频器的电路结构如图 3-15 所示。

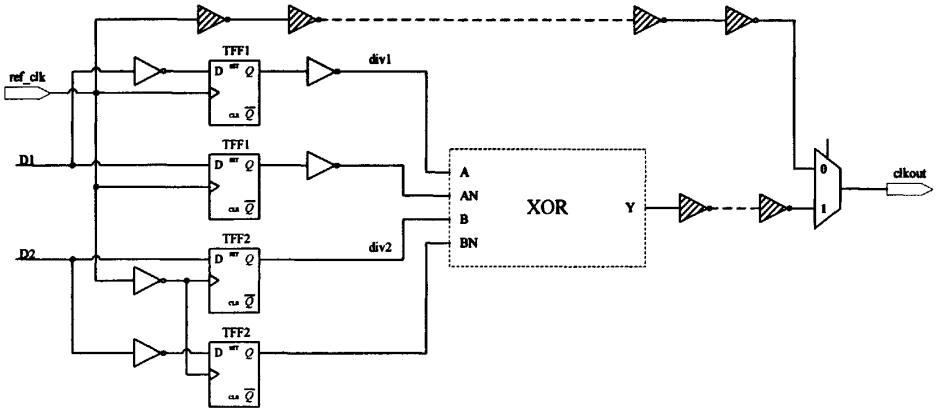


图 3-15 用反相器链调整时钟分频器内部的延迟差异

为了在保证逻辑正确性的同时尽量减小时钟偏差，上述反相器链需按如下原则构建：

1）反相器链需包含偶数个反相器；

2）每对反相器（第 2n-1 个反相器与第 2n 个反相器）类型相同；

3）相邻的反相器对（第 2n 个反相器与第 2n+1 个反相器）类型差别尽可能小，即反相器链的类型变化尽量平坦；

4）最后一对反相器选用标准单元库中驱动能力最大的反相器 CLKINVX20，满足时钟信号的高扇出要求。

通过上述的逻辑优化，时钟生成模块的固有时钟偏差被消除，各输出时钟信号基本平衡，避免了系统时序性能的恶化。

3.3. 动态变频技术的实现

基于上文所述的时钟生成模块，SK 系统芯片实现了动态变频技术来节省系统的动态功耗。由于不同的时钟信号在系统时钟结构中的地位不同，其动态变频的实现方法也不尽相同。本节详细介绍各级别时钟信号的动态变频的具体实现方法。

3.3.1. 处理器时钟与总线时钟的动态变频

对于 SK 系统芯片的时钟生成模块，处理器时钟、总线时钟以及各功能模块的工作时钟依次具有父子依赖关系。因此，当处理器时钟和总线时钟变频时，必

须对其本身及其后续子时钟的分频器和相位生成器进行适当的控制，避免在时钟信号上产生毛刺或者生成错误的相位指示信号。

首先，当处理器时钟和总线时钟变频时，应该通过时钟门控电路 M-Gating 与 B-Gating 关闭处理器时钟、总线时钟以及由总线时钟分频得到的各功能模块的工作时钟，避免变频过程中的质量较差的时钟信号被使用。

其次，处理器时钟和总线时钟变频通常涉及到改变 PLL 的配置参数，变频期间，处理器时钟的二分频器、总线时钟以及各功能模块的工作时钟的可配置分频器都应该被置入复位状态，待 PLL 的输出时钟稳定以后，再逐级将分频器的复位撤消。即，首先将处理器时钟的二分频器的复位撤消，待处理器时钟稳定以后，将总线时钟的分频器的复位撤消，最后，待总线时钟稳定以后，将各功能模块工作时钟的分频器的复位撤消。这样，就避免了时钟分频器因输入时钟的频率和相位发生变化而进入不确定的状态。

最后，当任意时钟的分频器被置入复位状态时，与其相关的所有相位生成器也应被置入复位状态，直到该时钟重新稳定以后，才撤消与其相关的相位生成器的复位，从而保证相位生成器不会因输入时钟的频率和相位发生变化而进入不确定的状态，避免生成错误的相位指示信号。

基于上述控制原则，本文设计了如图 3-16 所示的状态机来实现处理器时钟与总线时钟的动态变频。

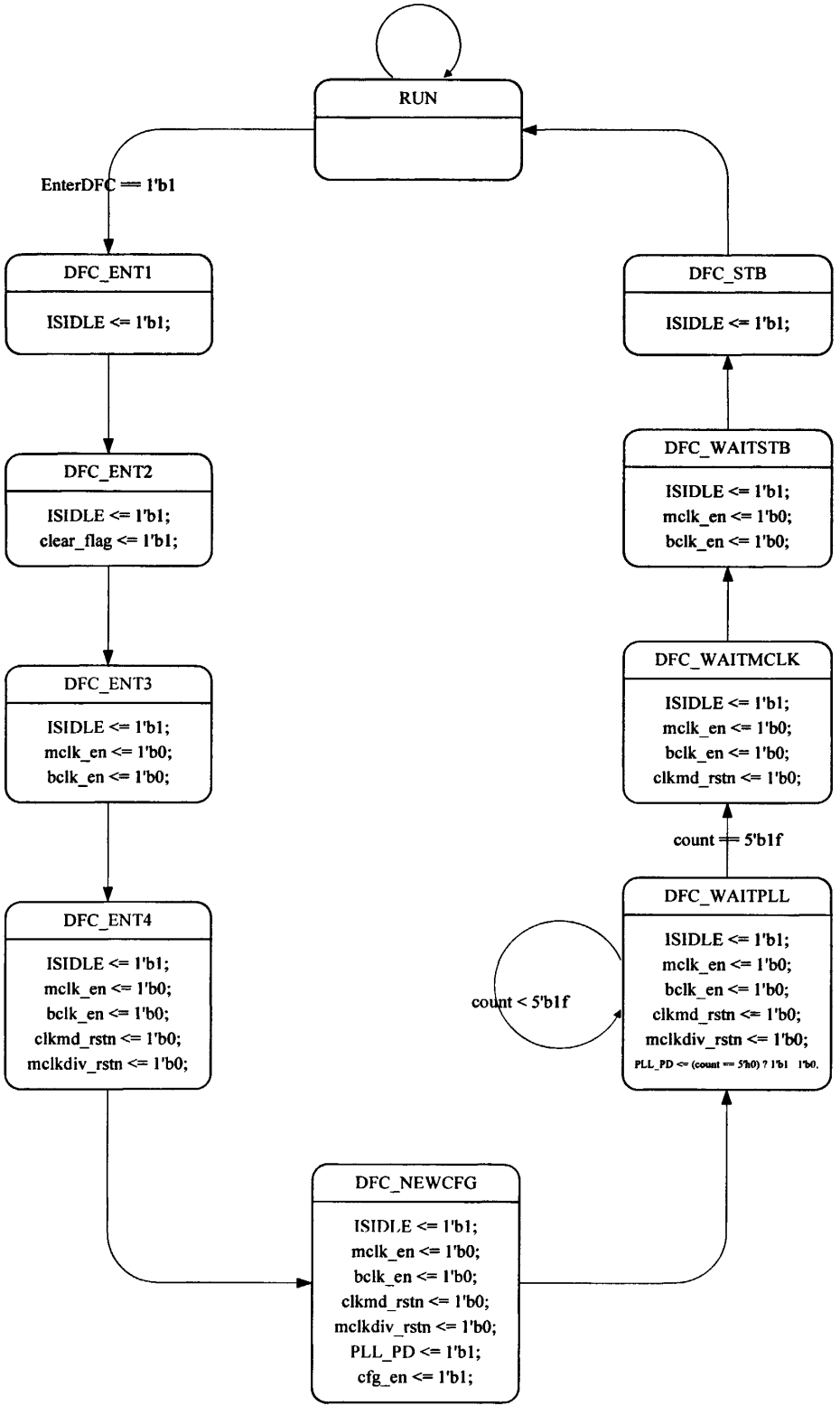


图 3-16 实现处理器时钟与总线时钟的动态变频的状态机

下面详细描述图 3-16 所示的状态转移关系以及每一个状态的输出与动作。

- RUN

系统完成上电启动流程后，进入此状态。在此状态下，功耗管理模块（Power Manager）不动作，系统处于正常工作模式。

如果接到处理器时钟变频或者总线时钟变频的命令，则转入 DFC_ENT1 状态。否则，继续处于此状态。

- DFC_ENT1

从此状态开始， ISIDLE 信号被置为高电平，标志着系统处于变频工作模式。

1 个 RTC_clk 周期后自动转入 DFC_ENT2 状态。

- DFC_ENT2

将清零信号 clear_flag 置为高电平，使输出变频命令的 PMCR 寄存器归零，避免变频完成后再重复启动变频流程。由于寄存器清零的动作需要在总线时钟的驱动下完成，所以此状态下尚不能够关闭时钟。

1 个 RTC_clk 周期后自动转入 DFC_ENT3 状态。

- DFC_ENT3

在此状态下，状态机输出低电平的门控使能信号 mclk_en、bclk_en，通过时钟门控电路 M-Gating 与 B-Gating 关闭处理器时钟、总线时钟以及模块工作时钟。

1 个 RTC_clk 周期后自动转入 DFC_ENT4 状态。

- DFC_ENT4

将时钟生成模块中包括处理器时钟的二分频器在内的所有时钟分频器和相位生成器置入复位状态，即停止系统内部时钟的生成。

1 个 RTC_clk 周期后自动转入 DFC_NEWCFG 状态。

- DFC_NEWCFG

将 PLL 置入 Power Down 模式，即停止 PLL 输出时钟的生成。由于在 DFC_ENT3 状态中已经通过时钟门控电路将时钟生成模块的各输出时钟关断，所以，在 DFC_ENT4 状态以及此状态下停止时钟的生成不会导致这些时钟信号上出现毛刺。

SK 系统芯片中的 PLL（PG13A1LV3）要求：改变 PLL 的配置参数之前需要将其置入 Power Down 模式至少 500ns，改变配置参数之后，应维持 PLL 继续处

于 Power Down 模式至少 500ns[20]。根据这一要求，在此状态下，状态机输出高电平的参数改变使能信号 cfg_en。该使能信号的含义是：当且仅当使能信号 cfg_en 为高电平时，才允许新的配置参数被传递到 PLL 的相应输入端上。

1 个 RTC_clk 周期后自动转入 DFC_WAITPLL 状态。

- DFC_WAITPLL

32 个 RTC_clk 周期后自动转入 DFC_WAITMCLK 状态。

第 1 个 RTC_clk 周期中仍维持 PLL 处于 Power Down 模式，即满足上述的改变配置参数后 PLL 继续处于 Power Down 模式至少 500ns 的要求。从第 2 个 RTC_clk 周期开始，令 PLL 回到正常工作模式。由于 1 个 RTC_clk 周期约为 30.52 μs，所以第 32 个 RTC_clk 周期时，PLL 已经进入正常工作模式约 0.9ms，超过了 PLL 重新锁定到新的频率所需要的时间 $T_{RDY} = 0.5ms$。此时，PLL 的输出时钟已经是稳定有效的，可以开始逐级恢复时钟。

- DFC_WAITMCLK

撤消处理器时钟的二分频器的复位。在离开此状态时，处理器时钟已经是稳定有效的。

1 个 RTC_clk 周期后自动转入 DFC_WAITSTB 状态。

- DFC_WAITSTB

当状态机到达此状态时，处理器时钟已经是稳定有效的。在此状态下，状态机将撤消时钟生成模块的复位。这首先导致第二级时钟 bclk32 与 bclk64 的时钟分频器的复位被撤消，随后，bclk32 与 bclk64 的相位生成器以及第三级时钟 ve_clk、he_clk、hd_clk、nand_clk 与 ge_clk 的时钟分频器的复位被撤消，最后，第三级时钟的相位生成器的复位被撤消。在离开此状态时，处理器时钟、总线时钟以及模块工作时钟的时钟信号和相位指示信号都是稳定而正确的。

1 个 RTC_clk 周期后自动转入 DFC_STB 状态。

- DFC_STB

按照上述的逐级恢复流程，当状态机到达此状态时，包括处理器时钟、总线时钟和模块工作时钟在内的所有时钟信号以及各同步时钟对的相位指示信号都已经正确地生成。在此状态下，状态机输出高电平的门控使能信号 mclk_en、bclk_en，通过时钟门控电路 M-Gating 与 B-Gating 打开各时钟。

1 个 RTC_clk 周期后自动转入 RUN 状态。

3.3.2. 模块工作时钟的动态变频

在如图 3-1 所示的时钟生成模块中，各功能模块的工作时钟（ve_clk、he_clk、hd_clk、nand_clk 和 ge_clk）属于第三级时钟。它们由总线时钟 bclk32 或者 bclk64 分频得到，仅被输出到各自对应的功能模块和相关的相位生成器，不再分频产生任何时钟。这些时钟动态变频的流程与处理器时钟和总线时钟动态变频的流程不同，既不需要将 PLL 置入 Power Down 模式，也不需要关闭处理器时钟、总线时钟等父时钟。因此，对于模块工作时钟的动态变频，本文没有为其设计状态机，而是基于软硬件协同控制的思想，采用了定时复位的实现方法。

在 SK 系统芯片的功耗管理模块（Power Manager）中定义有两个软件可编程的寄存器[21][22][23]。

一个寄存器是时钟门控寄存器（PCGR），该寄存器通过控制各时钟的门控使能信号的值来控制各时钟信号的打开与关断。寄存器的每一位对应一个时钟信号。如果某一位为高电平，则对应的时钟被门控电路关断，否则，该时钟被打开。该寄存器的详细描述如表 3-1 所示。

表 3-1 时钟门控寄存器

位	描述
0	DDR2 SDRAM 控制器的 64 位总线接口时钟的门控使能位
1	显示控制器的 64 位总线接口时钟的门控使能位
2	DDR2 SDRAM 控制器的 32 位总线接口时钟的门控使能位
3	保留
4	PCI 桥接器的总线接口时钟的门控使能位
5	DMA 控制器的总线接口时钟的门控使能位
6	以太网控制器的总线接口时钟的门控使能位
7	USB OTG 控制器的总线接口时钟的门控使能位
8	SATA 硬盘控制器的总线接口时钟的门控使能位
9	保留
10	MPEG 视频解码引擎的总线接口时钟的门控使能位

11	NAND Flash 控制器的总线接口时钟的门控使能位
12	H.264 编码器的总线接口时钟的门控使能位
13	显示控制器的总线接口时钟的门控使能位
14	H.264 解码器的总线接口时钟的门控使能位
15	MPEG 视频解码引擎的工作时钟的门控使能位
16	H.264 编码器的工作时钟的门控使能位
17	H.264 解码器的工作时钟的门控使能位
18	NAND Flash 控制器的工作时钟的门控使能位
19	图形加速器的工作时钟的门控使能位
20	显示控制器的工作时钟的门控使能位
21	PCI 桥接器的工作时钟的门控使能位
22	以太网控制器的发送时钟的门控使能位
23	以太网控制器的接收时钟的门控使能位
24	USB OTG 控制器的工作时钟的门控使能位
25	SATA 硬盘控制器的工作时钟的门控使能位
31~26	保留

另一个相关的寄存器是软件复位寄存器（SW_RESET），该寄存器可以控制系统内部各功能模块的复位信号。寄存器的每一位对应一个模块。如果某一位为高电平，则对应的模块被置入复位状态，否则，该模块正常工作。该寄存器的详细描述如表 3-2 所示。

表 3-2 软件复位寄存器

位	描述
0	64 位 AHB 总线的复位控制位
1	32 位 AHB 总线的复位控制位
2	DDR-II 存储控制器的复位控制位
3	PCI 桥接器的复位控制位
4	DMA 控制器的复位控制位
5	以太网控制器的复位控制位
6	USB OTG 控制器的复位控制位

7	SATA 硬盘控制器的复位控制位
8	APB 总线的复位控制位
9	MPEG 视频解码引擎的复位控制位
10	微处理器的复位控制位
11	NAND Flash 控制器的复位控制位
12	静态存储控制器的复位控制位
13	ICE 调试器的复位控制位
14	H.264 编码器的复位控制位
15	H.264 解码器的复位控制位
16	图形加速器的复位控制位
17	显示控制器的复位控制位
18	ve_clk 的相位生成器的复位控制位
19	he_clk 的相位生成器的复位控制位
20	hd_clk 的相位生成器的复位控制位
21	nand_clk 的相位生成器的复位控制位
22	ve_clk 的时钟分频器的复位控制位
23	he_clk 的时钟分频器的复位控制位
24	hd_clk 的时钟分频器的复位控制位
25	nand_clk 的时钟分频器的复位控制位
26	vga_clk 的时钟分频器的复位控制位
27	ge_clk 的时钟分频器的复位控制位
31~28	保留

其中，针对时钟分频器和相位生成器的复位控制位都是定时撤消的。当时钟分频器的复位控制位被置为高电平时，经过 32768 个 bclk32 周期（约 130μs）后，该位自动清零；当相位生成器的复位控制位被置为高电平时，经过 65536 个 bclk32 周期（约 260μs）后，该位自动清零。

当希望改变模块工作时钟的频率时，需按如下控制流程操作：

1）对时钟门控寄存器编程，将待变频的时钟所对应的门控使能位置为高电平，关闭该时钟信号。

2）将新的分频系数写入分频比配置寄存器的相应位中。

3）对软件复位寄存器编程，将待变频的时钟的时钟分频器和相位生成器所对应的复位控制位置为高电平。这样，该时钟分频器和相位生成器均被置入复位状态，时钟信号和相位指示信号的生成同时停止。在此复位期间，新的分频系数被传递到该时钟分频器和相位生成器的输入端。32768 个 bclk32 周期（约 130μs）后，时钟分频器的复位被撤消，开始输出稳定有效的时钟信号。再经历 32768 个 bclk32 周期（约 130μs）后，相位生成器的复位也被撤消，开始输出正确的相位指示信号。

4）对时钟门控寄存器编程，将待变频的时钟所对应的门控使能位置为低电平。这样，该时钟信号被时钟门控电路打开，驱动相应的功能模块正常工作。

在上述控制流程中，软件触发的门控与硬件计时控制的复位相配合，完成了模块工作时钟的动态变频。采用这种软硬件协同控制的实现方法，既避免了时钟信号上出现毛刺，也避免了相位指示信号因输入时钟的频率和相位发生变化而错误地偏移。

3.4. 小结

本章首先介绍了 SK 系统芯片中时钟生成模块的 RTL 设计，包括其总体结构和各子模块的具体设计。由于时钟偏差是限制数字系统的性能的重要因素，因此，继续介绍了为减小时钟偏差而采用的反相器推前、异或门重构、寄存器复制、构建反相器链等逻辑优化技术。最后，详细介绍各级别时钟信号的动态变频的具体实现方法，这主要包括功耗管理模块（Power Manager）中与处理器时钟和总线时钟动态变频相关的状态机设计以及与模块工作时钟动态变频相关的软件接口。

第 4 章 时钟门控技术的设计与实现

时钟网络所消耗的功率在电路的动态功耗中占有相当大的比例（一般在 50%以上）。一方面，时钟节点具有最高的翻转概率和较大的扇出；另一方面，对于一个触发器，即使输入输出的逻辑值未发生变化，内部节点也将由于时钟信号的变化而发生翻转，从而产生动态功耗。因此，在不需要的时候关闭时钟，减少不必要的电路翻转，是降低电路的动态功耗的有效途径。

本章介绍时钟门控技术及其在 SK 系统芯片中的应用。4.1 节介绍时钟门控技术的基本概念；4.2 节介绍细粒度的时钟门控的实现，即如何使用 EDA 工具为寄存器堆插入时钟门控电路；4.3 节介绍粗粒度的时钟门控的实现，即如何对功能模块的时钟信号甚至系统芯片的时钟电路进行控制；4.4 节总结本章内容。

4.1. 时钟门控技术简介

对于共享同一时钟信号和同样的控制信号（包括置位信号 set、复位信号 reset、载入使能信号 load-enable、翻转使能信号 toggle）的一组寄存器，如果不采用时钟门控技术，逻辑综合工具将使用反馈环路和多路选择器实现其逻辑功能。图 4-1 就是这样实现的一组寄存器。

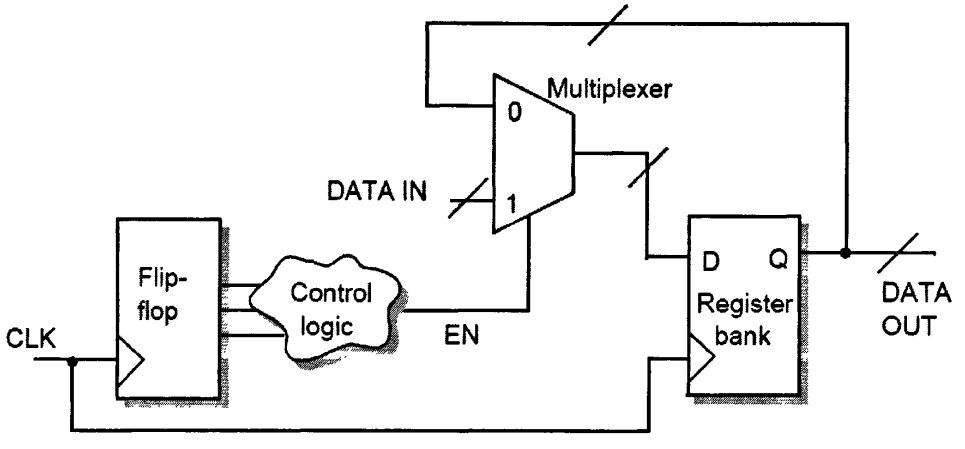


图 4-1 用多路选择器实现的寄存器

在图 4-1 中，当载入使能信号 EN 为 0 时，多路选择器将选择各寄存器的输出 Q 送至各自的输入端 D，寄存器堆（register bank）维持原值；当载入使能信

号 EN 为 1 时，多路选择器将选择新的数据值 DATAIN 送至各寄存器的输入端 D，寄存器堆的内容被更新。

上述结构在工作中存在一定的功耗浪费。假设在连续的多个时钟周期中，载入使能信号 EN 均为 0，那么寄存器堆一直维持原值。这期间，CLK 时钟节点反复翻转所消耗的功耗以及寄存器内部节点所消耗的功耗都是不必要的。

从降低功耗的角度考虑，可以采用时钟门控技术实现同样的逻辑功能。通过在寄存器堆的时钟端插入时钟门控电路，可以避免寄存器堆在连续的多个时钟周期中反复载入同样的数据值，从而避免不必要的功率消耗。采用时钟门控技术的寄存器堆的实现结构如图 4-2 所示。

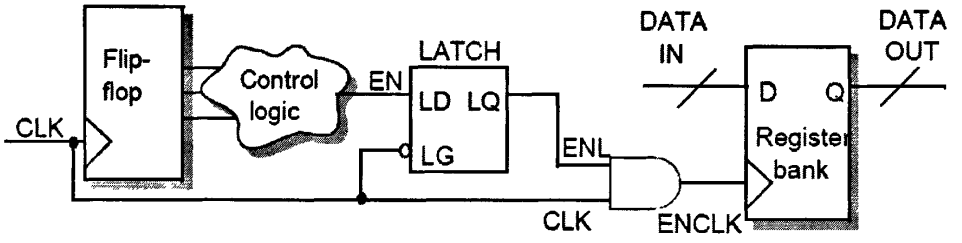


图 4-2 用时钟门控电路实现的寄存器

图 4-2 中，当载入使能信号 EN 为 1 时，寄存器堆的时钟输入 ENCLK 与时钟信号 CLK 一致，数据输入 DATAIN 将在 ENCLK 的上升沿被载入寄存器堆；当载入使能信号 EN 为 0 时，寄存器堆的时钟输入 ENCLK 被与门关断，由于没有有效的时钟触发沿，寄存器堆将维持原值。这样，时钟门控技术在保证逻辑正确性的同时避免了时钟信号 ENCLK 的不必要翻转，从而节省了电路的动态功耗[24][25]。

图 4-2 的时钟门控电路中使用了低电平触发的 LATCH。如果没有该 LATCH，载入使能信号 EN 直接和时钟信号 CLK 相与得到 ENCLK。那么，在 CLK 为高电平期间，使能信号 EN 的值的变化或者毛刺都将通过与门直接传递到 ENCLK 上，导致该时钟信号上出现毛刺。而插入低电平触发的 LATCH 后，在 CLK 为高电平期间，如果使能信号 EN 的值发生变化，这一变化将推迟到 CLK 为低电平时才被传递到与门输入端；如果使能信号 EN 上出现毛刺，这一毛刺将被完全屏蔽。因此，通过使用低电平触发的 LATCH，可以保证寄存器堆的时钟输入 ENCLK 上不会出现毛刺。图 4-3 所示的波形图反映了上述分析。

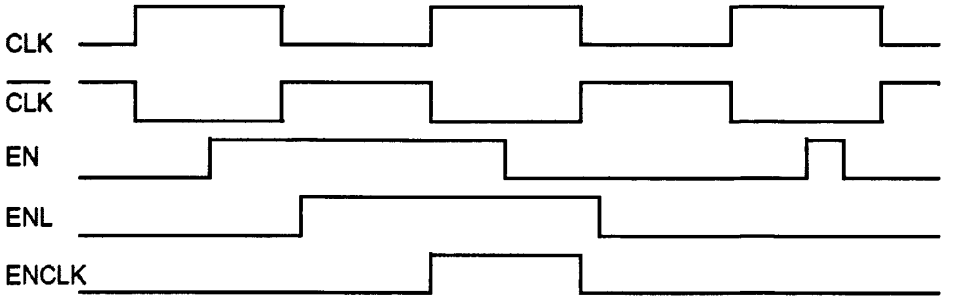


图 4-3 时钟门控电路的波形图

图 4-3 所示的时钟门控电路并不是时钟门控技术的唯一实现方式。除与门以外，与非门、或门、或非门等同样可以用来对时钟信号进行门控。值得注意的是，使用其它类型的逻辑门时，LATCH 的类型可能需要随之变化，或者，需要插入反相器或缓冲器来调整时序。例如，图 4-4 中，目标寄存器仍为上升沿触发的寄存器，或门配合反相器同样实现了对时钟信号 CLK 的门控。

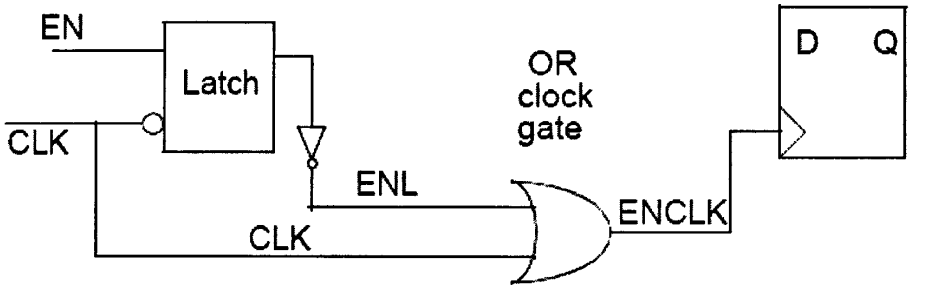


图 4-4 用或门实现的时钟门控电路

除了上述自由实现的时钟门控电路以外，还可以选用集成的时钟门控单元。如果选用自由实现的时钟门控电路，则设计流程将在以下几个方面变得复杂：

- 在物理实现阶段，需要保证 LATCH 和与门的输入时钟之间的时钟偏差尽量小，否则，可能导致时钟信号 ENCLK 上出现毛刺；
- 时钟树生成（clock tree synthesis，CTS）时，需要将 LATCH 的时钟输入端定义为 non stop pin；
- 静态时序分析（static timing analysis，STA）时，需要给定与门的 setup 时间与 hold 时间要求。

而如果选用集成的时钟门控单元，几乎不会给设计流程带来工作量和复杂度的增加：

- 作为一个标准库单元，LATCH 和与门的输入时钟之间的时钟偏差已经被

优化到近乎为零；

- 时钟树生成时,EDA 工具将自动处理该时钟门控单元，而不需要任何特别的指定；
- 标准库为时钟门控单元提供了 setup 时间与 hold 时间的检查模型，静态时序分析时，EDA 工具可以自动检查其时序。

根据上述分析，一般推荐选用集成的时钟门控单元，避免增加设计流程的工作量和复杂度。在 SK 系统芯片的设计中,无论是寄存器堆级的时钟门控，还是功能模块级甚至系统级的时钟门控，都选用集成的时钟门控单元来实现。其中，前者选用了 TLATNTSCAX16，后者选用了 TLATNCAX8。

4.2. 细粒度时钟门控的实现

目前的综合工具，如 Synopsys 的 Design Compiler 系列工具已经具有插入门控时钟的功能。EDA 工具插入的时钟门控电路作用于共享同一时钟信号和同样的控制信号（包括置位信号 set、复位信号 reset、载入使能信号 load-enable、翻转使能信号 toggle）的寄存器堆，因此称为寄存器堆级的时钟门控，是时钟门控技术在较细粒度上的应用。

以 Design Compiler 为例，要在逻辑综合阶段为满足时钟门控技术的应用条件的寄存器堆插入时钟门控电路，需要按如下步骤操作：

- 设置时钟门控电路的实现风格。这包括：

首先，明确目标寄存器堆是上升沿触发的还是下降沿触发的。

其次，确定是选用集成的时钟门控单元还是选用自由实现的时钟门控电路。如果选用前者，需指定集成的时钟门控单元的名字。如果选用后者，需明确具体的实现方式，即使用何种类型的二输入逻辑门（与门，与非门，或门，或非门），逻辑门的输入时钟信号上是否插入反相器或缓冲器，逻辑门的输出时钟信号上是否插入反相器或缓冲器。

- 设置时钟门控技术的应用条件。这包括：

首先，如果寄存器堆的载入使能信号 load-enable 恒为 1，说明寄存器堆在每一个时钟周期都要载入新的数据值，没有必要应用时钟门控技术。只有载入使能信号不恒为 1 时，该寄存器堆才可以应用时钟门控技术。这一条件不需要人为设

置，Design Compiler 将自动检查。

其次，只有共享相同时钟信号和控制信号的寄存器的数目高于一定的下限时，才值得对该寄存器堆应用时钟门控技术。这一下限值可以人为设定。

以 SK 系统芯片为例，在逻辑综合阶段，使用了如下命令来设置时钟门控技术的实现风格和应用条件：

```
set_clock_gating_style
    -positive_edge_logic "integrated:typical/TLATNTSCAX16"
    -minimum_bitwidth 4
    -enhanced_min_bitwidth 4
    -max_fanout 8
```

根据上述命令，如果共享相同时钟信号和控制信号的上升沿触发的寄存器的数目大于 4，Design Compiler 将使用集成的时钟门控单元 TLATNTSCAX16 实现这组寄存器的逻辑功能。并且，每个时钟门控单元最多驱动 8 个寄存器，如果该组寄存器的数目大于 8，Design Compiler 将会通过复制时钟门控单元来保证其扇出不超过 8。

- 使用 analyze 命令分析 RTL 设计。
- 用 elaborate 命令分析 RTL 设计时使用-gate_clock 选项或者在 elaborate 后使用 insert_clock_gating 命令，此时，Design Compiler 将检查上述的时钟门控技术的应用条件，对满足条件的寄存器堆插入时钟门控电路。
- 使用 report_clock_gating 命令输出关于时钟门控的详细报告。
- 使用 propagate_constraints 命令将 setup 时间与 hold 时间约束传递到当前设计。

按上述步骤操作，即在我们的设计中实现了时钟门控技术[26]。

值得注意的是，除了打开与关断时钟以外，对时钟信号进行门控的逻辑门不应改变时钟信号的形状，即不应导致时钟信号上出现毛刺。4.1 节中已经介绍了如何使用 LATCH 避免毛刺。而在物理实现后的静态时序分析阶段，需要进一步针对这一问题做时序检查。

以如图 4-5 所示的时钟门控电路为例，输入到与门的时钟使能信号 ENL 相对于原始时钟信号 CLK 有 setup 时间与 hold 时间的要求。使能信号 ENL 应在时

钟信号 CLK 由低电平变为高电平之前一定时间稳定，这一时间称为 setup 时间。并且，使能信号 ENL 应维持这一稳定值直到时钟信号 CLK 由高电平变为低电平之后一定时间，这一时间称为 hold 时间。如果使能信号 ENL 不满足上述定义的 setup 时间与 hold 时间要求，将有可能导致时钟信号在高电平时被门控逻辑切断，从而出现毛刺。

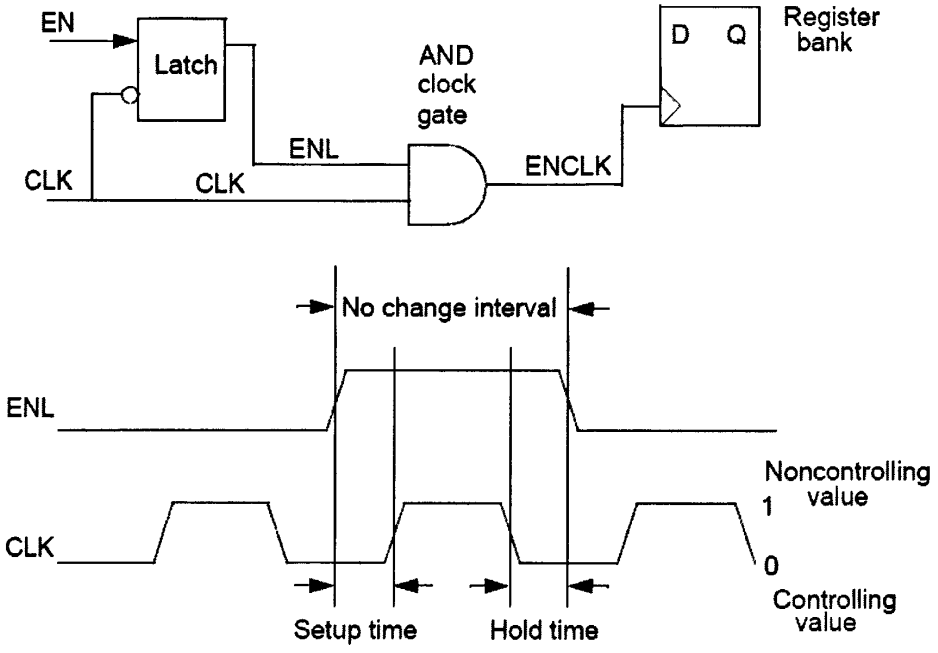


图 4-5 时钟门控电路的时序要求

对于集成的时钟门控单元，标准单元库中已经定义了 setup 时间与 hold 时间的检查模型，静态时序分析工具将自动处理该时钟门控单元，不需要人为设置。对于自由实现的时钟门控电路，可以使用 set_clock_gating_check –setup/-hold 命令来设置 setup 时间与 hold 时间要求[27]。

4.3. 粗粒度时钟门控的实现

除了针对寄存器堆应用时钟门控技术以外，还可以针对整个功能模块应用时钟门控技术。即，时钟生成模块为各功能模块提供互相独立的时钟，哪个模块不工作，就将哪个模块对应的时钟网络关闭。更进一步地，甚至可以针对整个系统应用时钟门控技术，在处理器空闲或者系统休眠时，选择性地关闭处理器时钟、总线时钟以及各功能模块的工作时钟。这分别称为模块级的时钟门控和系统级的

时钟门控，是时钟门控技术在较粗粒度上的应用[28][29][30][31]。本节将详细介绍粗粒度时钟门控技术在SK系统芯片中的实现。

4.3.1. 模块级时钟门控

首先，SK 系统芯片的时钟生成模块为各功能模块提供了互相独立的时钟，并显式地在每个时钟信号上插入了时钟门控电路（选用集成的时钟门控单元TLATNCAX8来实现）。

其次，在SK系统芯片的功耗管理模块（Power Manager）中定义有一个软件可编程的时钟门控寄存器，该寄存器通过控制各时钟的门控使能信号的值来控制各时钟信号的打开与关断。寄存器的每一位对应一个时钟信号。如果某一位为高电平，则对应的时钟被门控电路关断，否则，该时钟被打开。该寄存器的详细描述如表 3-1 所示。

这样，如果当前工作模式下不需要某一模块的工作，或者在一段时间内某一模块一直处于空闲状态，就可以通过对此时钟门控寄存器编程，关闭该模块的时钟网络。

例如，SK 系统芯片中的 NAND Flash 和 SATA 硬盘都可以作为独立的存储设备使用。如果 NAND Flash 的容量能够满足存储需求，那么就不需要使用 SATA 硬盘，可以对时钟门控寄存器的第 8 位和第 25 位写 1，分别关闭 SATA 硬盘控制器的总线时钟和工作时钟。又如，如果当前工作模式使用 PCI 桥接器上的网卡芯片而不使用片内集成的以太网控制器，可以对时钟门控寄存器的第 6 位、第 22 位和第 23 位写 1，分别关闭片内以太网控制器的总线时钟、发送时钟和接收时钟。

通过关闭空闲模块的输入时钟，不但消除了时钟网络上不必要的功率消耗，而且降低了空闲模块内部的寄存器和逻辑门的活性，大大节省了系统的动态功耗。

4.3.2. 系统级时钟门控——IDLE

当处理器处于空闲状态时，可以将系统置入 IDLE 工作模式。该工作模式下，处理器时钟被时钟门控电路关断，其它时钟（总线时钟和模块工作时钟等）不受

影响。并且，系统芯片的供电没有变化，cache 的内容、主存的内容以及所有设备内部的寄存器的内容均不会丢失。任意中断信号可以将系统从 IDLE 工作模式唤醒至正常工作模式。

一个典型的场景是，除了接收网络上的数据包以外，系统没有任何动作。此时，在以太网控制器接收数据包并通过内嵌 DMAC 将数据包存入内存缓冲区期间，不需要任何处理器动作，OS 可以通过将系统置入 IDLE 工作模式（即关闭处理器时钟）来降低功耗。SK 系统芯片中的以太网控制器支持多达 256 个接收内存缓冲区，在 256 个接收缓冲区全部被填满以后，以太网控制器将发出接收上溢中断，该中断导致系统离开 IDLE 工作模式而恢复到正常工作模式。在正常工作模式下，处理器对以太网控制器的中断请求进行处理，之后，可以再次进入 IDLE 工作模式，而以太网控制器则继续接收后续的网络数据包。

本文设计了如图 4-6 所示的状态机来实现上面定义的 IDLE 工作模式。

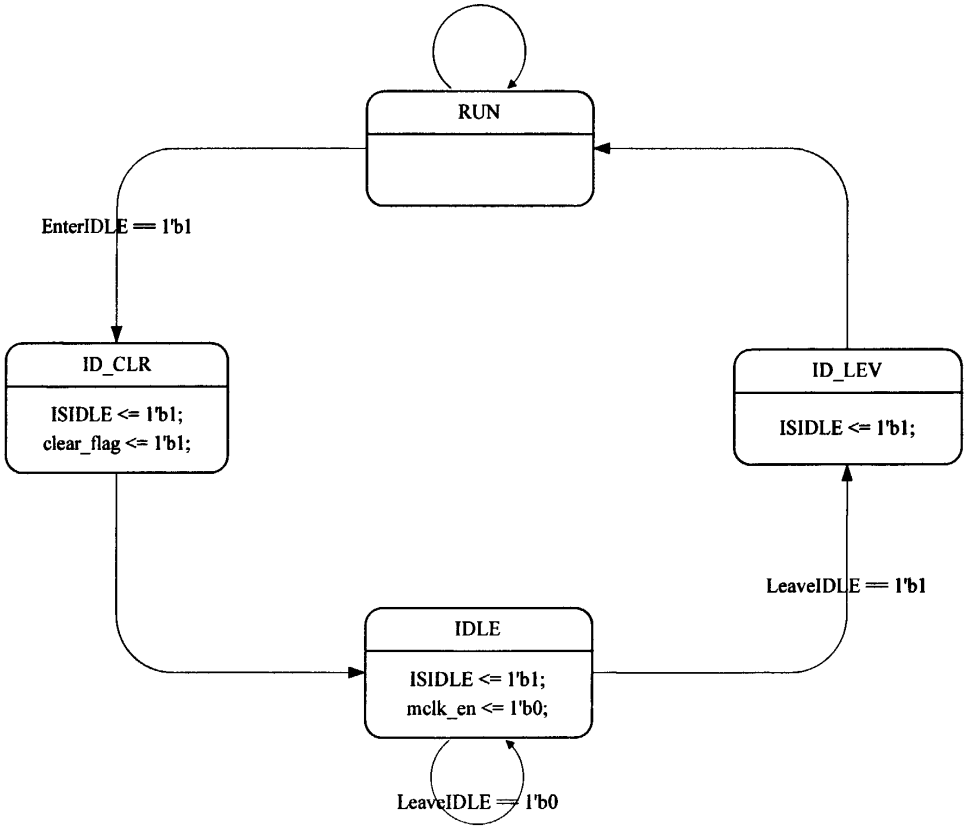


图 4-6 实现 IDLE 流程的状态机

下面详细描述图 4-6 所示的状态转移关系以及每一个状态的输出与动作。

- RUN

系统完成上电启动流程后，进入此状态。在此状态下，功耗管理模块（Power Manager）不动作，系统处于正常工作模式。

如果接到进入 IDLE 工作模式的命令，则转入 ID_CLR 状态。否则，继续处于此状态。

- ID_CLR

从此状态开始，ISIDLE 信号被置为高电平，标志着系统处于 IDLE 工作模式。

将清零信号 clear_flag 置为高电平，使输出“EnterIDLE”命令的 PMCR 寄存器归零，避免退出 IDLE 工作模式后再重复启动 IDLE 流程。

一个 RTC_clk 周期后自动转入 IDLE 状态。

- IDLE

在此状态下，状态机将输出低电平的门控使能信号 mclk_en，通过图 3-1 中所示的时钟门控电路 M-Gating 关闭处理器时钟。

如果接到离开 IDLE 工作模式的命令，则转入 ID_LEV 状态。否则，继续处于此状态。

- ID_LEV

ISIDLE 信号仍为高电平，同时，处理器时钟的门控使能信号 mclk_en 恢复到高电平，处理器时钟被打开。

一个 RTC_clk 周期后自动转入 RUN 状态。

4.3.3. 系统级时钟门控——SLEEP

如果处理器和各外围设备都没有工作任务，即整个系统处于空闲状态，可以将其置入 SLEEP 工作模式。该工作模式下，处理器时钟、总线时钟和模块工作时钟等全部被关断，并且，可以选择是否将 PLL 置入 Power Down 模式来进一步降低功耗。如果将 PLL 置入 Power Down 模式，将再节省约 10mW 的功耗，但是，PLL 的重新锁定会导致唤醒时间增加 1ms 左右。当唤醒时间的增加不能忍受时，应选择不关闭 PLL 的 SLEEP 工作模式。

SLEEP 工作模式下，系统芯片的供电没有变化，cache 的内容、主存的内容以及所有设备内部的寄存器的内容均不会丢失。RTC 定时唤醒信号以及来自通

用输入输出接口（GPIO）的唤醒信号都可以将系统从 SLEEP 工作模式唤醒至正常工作模式。

本文设计了如图 4-7 所示的状态机来实现不关闭 PLL 的 SLEEP 工作模式。

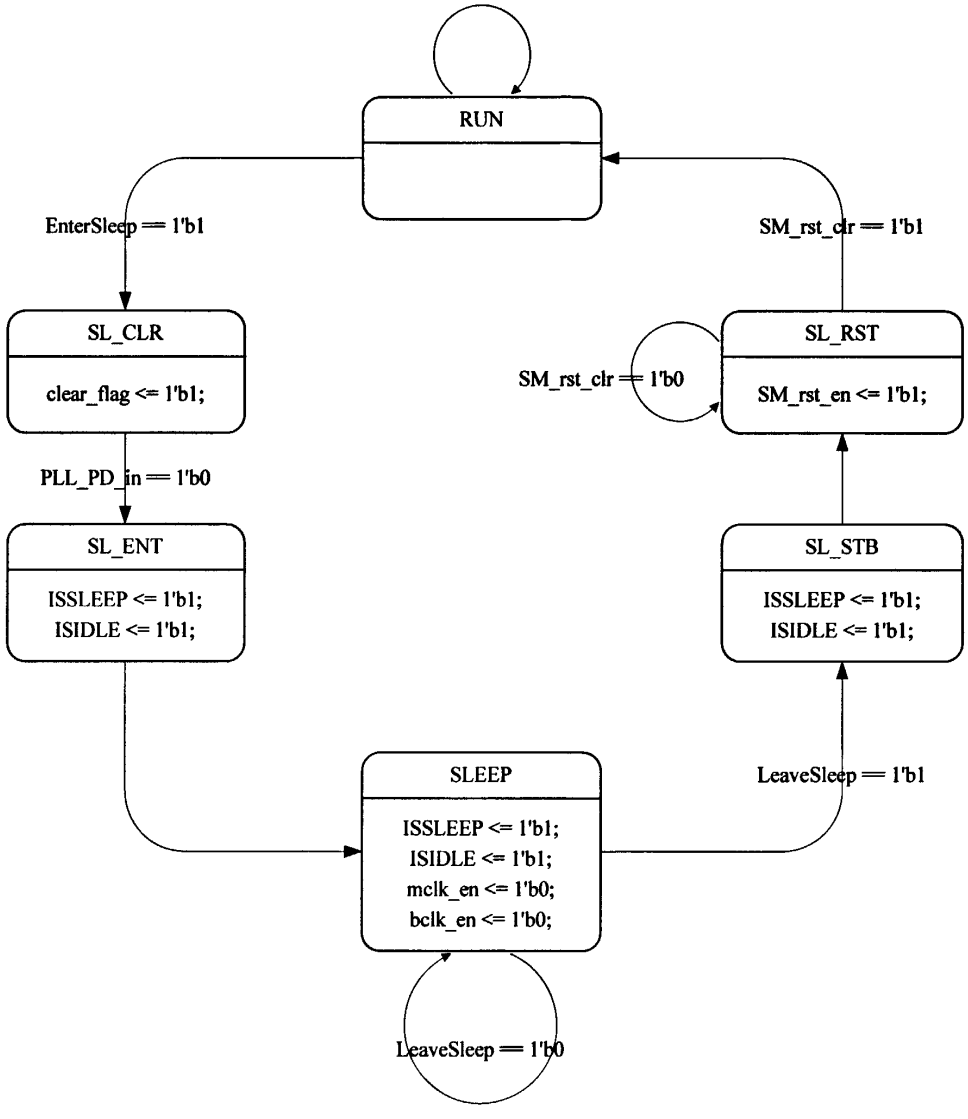


图 4-7 实现不关闭 PLL 的 SLEEP 流程的状态机

下面详细描述图 4-7 所示的状态转移关系以及每一个状态的输出与动作。

- RUN

系统完成上电启动流程后，进入此状态。在此状态下，功耗管理模块（Power Manager）不动作，系统处于正常工作模式。

如果接到进入 SLEEP 工作模式的命令，则转入 SL_CLR 状态。否则，继续

处于此状态。

- SL_CLR

将清零信号 clear_flag 置为高电平，使输出“EnterSLEEP”命令的 PMCR 寄存器归零，避免退出 SLEEP 工作模式后再重复启动 SLEEP 流程。由于寄存器清零的动作需要在总线时钟的驱动下完成，所以此状态下尚不能够关闭时钟。

当 PLL_PD_in 信号为低电平时，表示用户选择了不关闭 PLL 的 SLEEP 工作模式，1 个 RTC_clk 周期后自动转入 SL_ENT 状态。

- SL_ENT

从此状态开始， ISSLEEP 信号和 ISIDLE 信号均被置为高电平，标志着系统处于 SLEEP 工作模式。

1 个 RTC_clk 周期后自动转入 SLEEP 状态。

- SLEEP

在此状态下，状态机输出低电平的门控使能信号 mclk_en、bclk_en，通过图 3-1 中所示的时钟门控电路 M-Gating、B-Gating 关闭处理器时钟、总线时钟以及模块工作时钟。

如果接到离开 SLEEP 工作模式的命令，则转入 SL_STB 状态。否则，继续处于此状态。

- SL_STB

ISSLEEP 信号和 ISIDLE 信号仍为高电平，同时，门控使能信号 mclk_en、bclk_en 恢复到高电平，处理器时钟、总线时钟以及模块工作时钟均被打开。

1 个 RTC_clk 周期后自动转入 SL_RST 状态。

- SL_RST

将全局复位信号拉低，计时 2^{22} 个 mclk_s 周期后（约 7ms）撤消该复位并自动转入 RUN 状态。该全局复位信号将影响除功耗管理模块（Power Manager）、实时时钟（RTC）、通用输入输出接口（GPIO）以外的所有模块。

关闭 PLL 的 SLEEP 工作模式涉及到时钟生成的停止与重新启动，因此略为复杂一些。本文设计了如图 4-8 所示的状态机来实现关闭 PLL 的 SLEEP 工作模式。

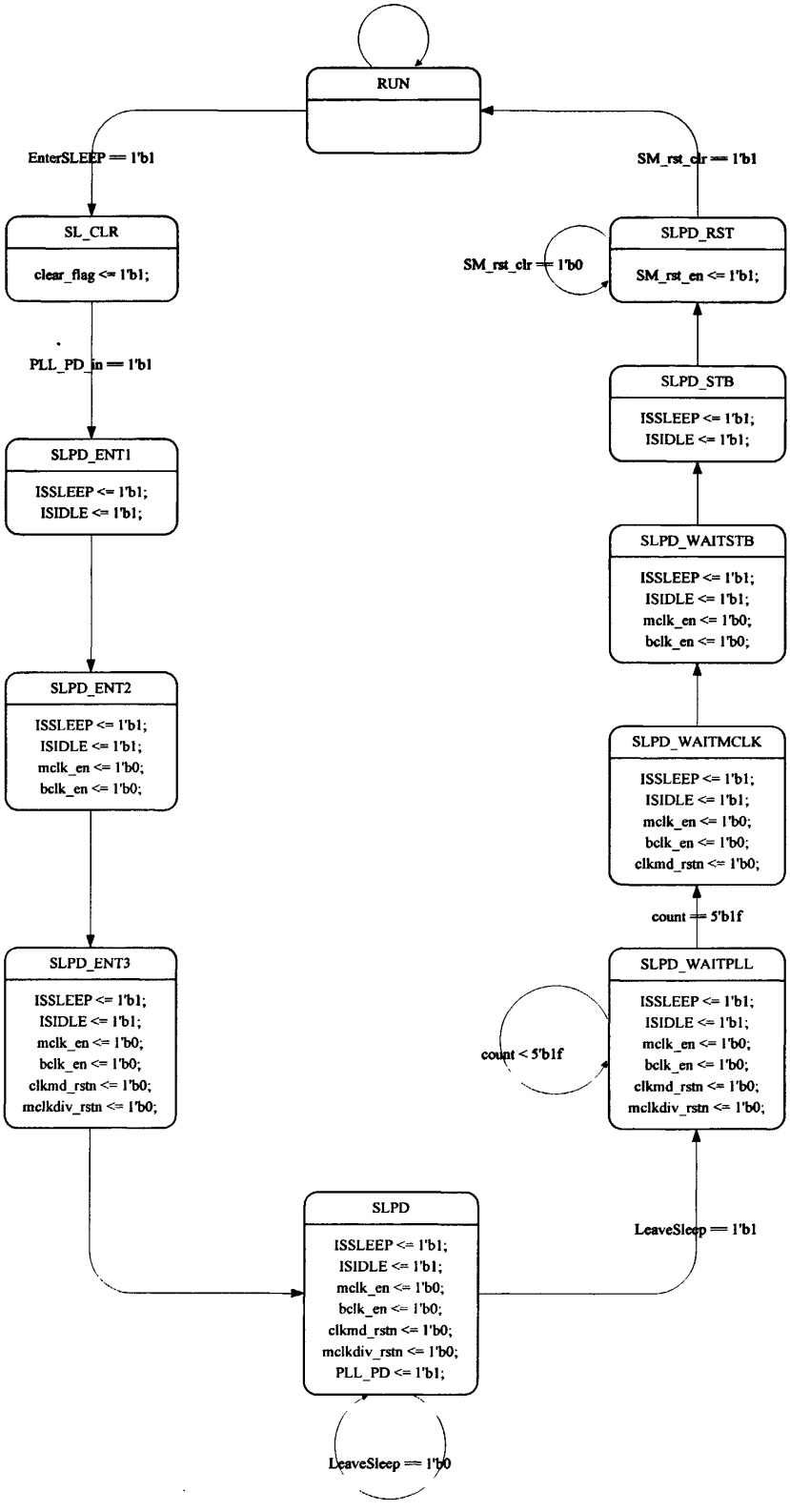


图 4-8 实现关闭 PLL 的 SLEEP 流程的状态机

下面详细描述图 4-8 所示的状态转移关系以及每一个状态的输出与动作。

- RUN

系统完成上电启动流程后，进入此状态。在此状态下，功耗管理模块（Power Manager）不动作，系统处于正常工作模式。

如果接到进入 SLEEP 工作模式的命令，则转入 SL_CLR 状态。否则，继续处于此状态。

- SL_CLR

将清零信号 clear_flag 置为高电平，使输出"EnterSLEEP"命令的 PMCR 寄存器归零，避免退出 SLEEP 工作模式后再重复启动 SLEEP 流程。由于寄存器清零的动作需要在总线时钟的驱动下完成，所以此状态下尚不能够关闭时钟。

当 PLL_PD_in 信号为高电平时，表示用户选择了关闭 PLL 的 SLEEP 工作模式，1 个 RTC_clk 周期后自动转入 SLPD_ENT1 状态。

- SLPD_ENT1

从此状态开始， ISSLEEP 信号和 ISIDLE 信号均被置为高电平，标志着系统处于 SLEEP 工作模式。

1 个 RTC_clk 周期后自动转入 SLPD_ENT2 状态。

- SLPD_ENT2

在此状态下，状态机输出低电平的门控使能信号 mclk_en、bclk_en，通过图 3-1 中所示的时钟门控电路 M-Gating、B-Gating 关闭处理器时钟、总线时钟以及模块工作时钟。

1 个 RTC_clk 周期后自动转入 SLPD_ENT3 状态。

- SLPD_ENT3

将时钟生成模块中包括处理器时钟的二分频器在内的所有时钟分频器和相位生成器置入复位状态，即停止系统内部时钟的生成。

1 个 RTC_clk 周期后自动转入 SLPD 状态。

- SLPD

将 PLL 置入 Power Down 模式，即停止 PLL 输出时钟的生成。由于在 SLPD_ENT2 状态中已经通过时钟门控电路将时钟生成模块的各输出时钟关断，所以，在 SLPD_ENT3 状态以及此状态下停止时钟的生成不会导致这些时钟信号

上出现毛刺。

如果接到离开 SLEEP 工作模式的命令，则转入 SLPD_WAITPLL 状态。否则，继续处于此状态。

- SLPD_WAITPLL

令 PLL 从 Power Down 模式回到正常工作模式。

32 个 RTC_clk 周期后自动转入 SLPD_WAITMCLK 状态。由于 1 个 RTC_clk 周期约为 30.52 μs，所以 32 个 RTC_clk 周期后，PLL 已经进入正常工作模式约 0.9ms，超过了 PLL 重新锁定到新的频率所需要的时间 $T_{RDY} = 0.5ms$。此时，PLL 的输出时钟已经是稳定有效的，可以开始逐级恢复时钟。

- SLPD_WAITMCLK

撤消处理器时钟的二分频器的复位。在离开此状态时，处理器时钟已经是稳定有效的。

1 个 RTC_clk 周期后自动转入 SLPD_WAITSTB 状态。

- SLPD_WAITSTB

当状态机到达此状态时，处理器时钟已经是稳定有效的。在此状态下，状态机将撤消时钟生成模块的复位。这首先导致第二级时钟 bclk32 与 bclk64 的时钟分频器的复位被撤消，随后，bclk32 与 bclk64 的相位生成器以及第三级时钟 ve_clk、he_clk、hd_clk、nand_clk 与 ge_clk 的时钟分频器的复位被撤消，最后，第三级时钟的相位生成器的复位被撤消。在离开此状态时，处理器时钟、总线时钟以及模块工作时钟的时钟信号和相位指示信号都是稳定而正确的。

1 个 RTC_clk 周期后自动转入 SLPD_STB 状态。

- SLPD_STB

按照上述的逐级恢复流程，当状态机到达此状态时，包括处理器时钟、总线时钟和模块工作时钟在内的所有时钟信号以及各同步时钟对的相位指示信号都已经正确地生成。在此状态下，状态机输出高电平的门控使能信号 mclk_en、bclk_en，通过图 3-1 中所示的时钟门控电路 M-Gating、B-Gating 打开各时钟。

1 个 RTC_clk 周期后自动转入 SLPD_RST 状态。

- SLPD_RST

将全局复位信号拉低，计时 2^{22} 个 mclk_s 周期后（约 7ms）撤消该复位并自

动转入 RUN 状态。该全局复位信号将影响除功耗管理模块（Power Manager）、实时时钟（RTC）、通用输入输出接口（GPIO）以外的所有模块。

4.4. 小结

本章首先介绍了时钟门控技术的基本原理和电路实现，比较了自由实现的时钟门控电路和集成的时钟门控单元。随后，进一步介绍了时钟门控技术在 SK 系统芯片中的应用情况：一方面，总结了使用 EDA 工具插入时钟门控电路的方法，介绍了时钟门控电路的时序要求；另一方面，详细描述了粗粒度的时钟门控的实现，这主要包括功耗管理模块（Power Manager）中与模块级时钟门控相关的软件接口以及与系统级时钟门控相关的状态机设计。

第 5 章 时钟生成模块的优化

对于第 3 章中描述的时钟生成模块，一方面，不同的时钟路径上级联的时钟分频器数目不同，导致输出时钟 mclk 与 bclk32、bclk64 之间，以及 bclk32、bclk64 与 ve_clk、he_clk、hd_clk、nand_clk、ge_clk 之间具有较大的延迟差异；另一方面，时钟分频器内部的各条重汇聚路径之间也存在较大的延迟差异。为了消除上述延迟差异导致的时钟偏差，需要采用反相器推前、异或门重构、寄存器复制、构建反相器链等多种技术来平衡各时钟路径。这为后端设计流程增加了若干工作量和复杂度。为了消除时钟生成模块的固有时钟偏差，为后端实现提供一个较为理想的起点，本章对时钟生成模块的 RTL 设计进行了优化，使其具有更为平衡的电路结构。5.1 节介绍优化后的时钟生成模块的总体结构，5.2 节介绍时钟分频器中的核心分频模块 clk_gen 的设计，5.3 节介绍时钟分频器中的控制模块 clk_ctrl 的设计，5.4 节总结本章内容。

5.1. 优化的时钟生成模块的结构

考虑如图 3-1 所示的时钟生成模块，各输出时钟信号都源自多路选择器 MUX2 的输出时钟 mclk_s，但是，不同输出时钟所经历的时钟分频器的数目却不尽相同。从 mclk_s 到第一级时钟 mclk 不经历任何时钟分频器，从 mclk_s 到第二级时钟 bclk32 和 bclk64 经历一个时钟分频器，而从 mclk_s 到 ve_clk、ge_clk 等第三级时钟则需经历两个时钟分频器。为了消除这一延迟差异，可以将三级的分频结构压缩至一级：所有时钟信号仍然源自同一时钟，并且，各时钟都只经历一次分频。

图 5-1 为改进的时钟生成模块的结构框图。其中，各输出时钟的源头时钟是 PLL 的输出时钟 pll_fout 和片外晶振时钟 clk_byp 的二选一输出时钟 clk_src。包括 mclk、bclk32、bclk64、ve_clk、he_clk、hd_clk、nand_clk、ge_clk 在内的所有时钟信号都直接由 clk_src 分频得到，整个时钟生成模块中不存在时钟分频器的级联结构。

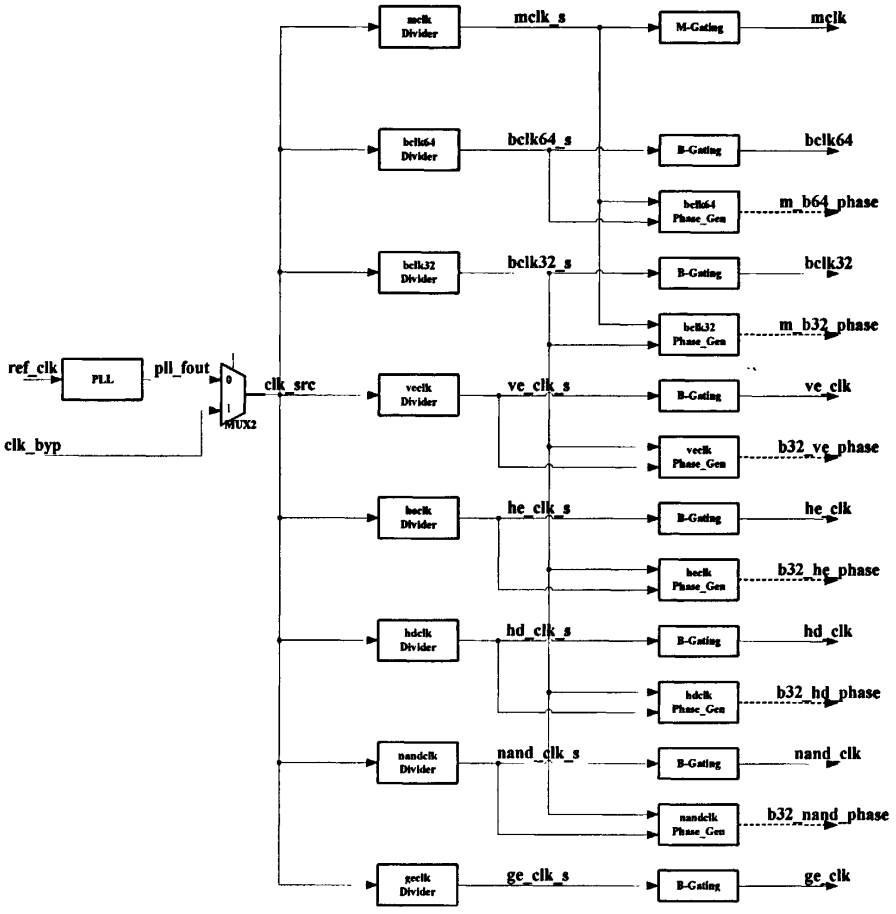


图 5-1 改进的时钟生成模块的结构框图

对于时钟生成模块的这一结构，从源头时钟到各输出时钟之间都只经历一个时钟分频器，各条时钟路径的延迟差异大大减小，更易于实现平衡。

其次，为了消除时钟分频器内部的各条重汇聚路径之间的延迟差异，本文也对时钟分频器进行了重设计，牺牲了 50%占空比等特性，来换取更为平衡的电路结构。如图 5-2 所示，此时钟分频器由 clk_gen 和 clk_ctrl 两个子模块组成。其中，clk_gen 是产生分频时钟 clk_out 的核心模块；clk_ctrl 则根据分频系数 divisor、门控使能信号 clk_en 以及复位信号 rst_n 等向 clk_gen 模块输出控制信号 hold_n、no_div 和 toggle。5.2 节与 5.3 节将详细描述这两个子模块的具体设计。

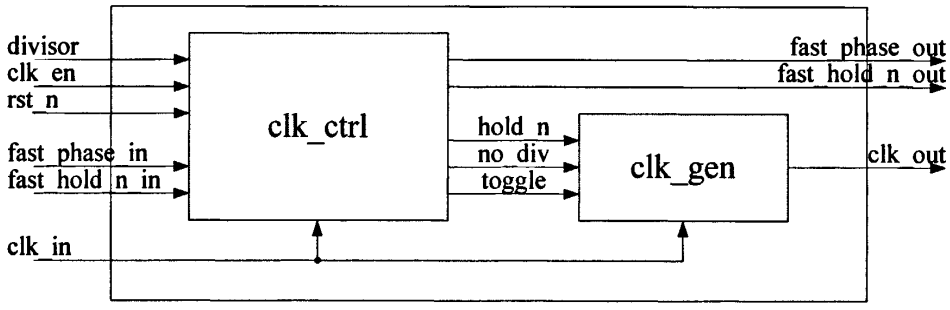


图 5-2 改进的时钟分频器的结构框图

最后，值得注意的是，在 UniCore-2 微处理器的设计中，处理器时钟 mclk 和高速总线时钟 bclk64 以及低速总线时钟 bclk32 被视为同步不同频的时钟；在 MPEG 视频解码引擎、H.264 编/解码器、NAND Flash 控制器这些功能模块的设计中，核心工作时钟 ve_clk、he_clk、hd_clk、nand_clk 和总线时钟 bclk32 也被视为同步不同频的时钟。为了保证跨越这种同步时钟域的信号交互的正确性，时钟生成模块需要对每一对这样的时钟生成相位指示信号。但是，在改进的时钟生成模块中，由于分频关系的改变，各同步时钟对不再是直接的分频与倍频关系，而是通过源头时钟 clk_src 间接地联系着。因此，相位指示信号的生成将变得略为复杂。

对于时钟分频器，输入的分频系数应为输出时钟相对于输入时钟（在改进的时钟生成模块中，所有时钟分频器的输入时钟均为 clk_src）的分频比；而对于相位生成器，输入的分频系数应为相应的同步时钟对（fast_clk 与 slow_clk）之间的频率比。因此，根据对软件可编程寄存器中存放的分频系数的不同定义和解释，存在下述两种设计方法。

一种方法是维持原来的分频系数的定义，即：mclk_div 表示 mclk 相对于 clk_src 的分频系数；bclk32_div 表示 bclk32 相对于 mclk_s 的分频系数；bclk64_div 表示 bclk64 相对于 mclk_s 的分频系数；veclk_div / heclk_div / hdclk_div / nandclk_div 表示 ve_clk / he_clk / hd_clk / nand_clk 相对于 bclk32_s 的分频系数；geclk_div 表示 ge_clk 相对于 bclk64_s 的分频系数。这样，相位生成器的输入分频系数直接来自软件可编程寄存器，但时钟分频器的输入分频系数需要由 1～3 个分频系数相乘得到。

另一种方法是将软件可编程寄存器中存放的分频系数定义为各时钟相对于源头时钟 clk_src 的分频比。这种情况下，时钟分频器的输入分频系数直接来自

软件可编程寄存器，但相位生成器的输入分频系数则需要由两个分频系数相除得到。

考虑到除法的实现代价比较高，本文采用了第一种设计方法，即将软件可编程寄存器中存放的分频系数定义为同步时钟对之间的频率比，时钟分频器的输入分频系数由 1～3 个这样的相对频率比相乘得到。例如，bclk64 的时钟分频器的输入分频系数为：bclk64_div * mclk_div；而 ge_clk 的时钟分频器的输入分频系数为 ge_clk * bclk64_div * mclk_div。

5.2. 分频模块 clk_gen 的 RTL 设计

图 5-2 所示的时钟分频器中，clk_gen 模块是产生分频时钟 clk_out 的核心模块。其电路结构如图 5-3 所示。

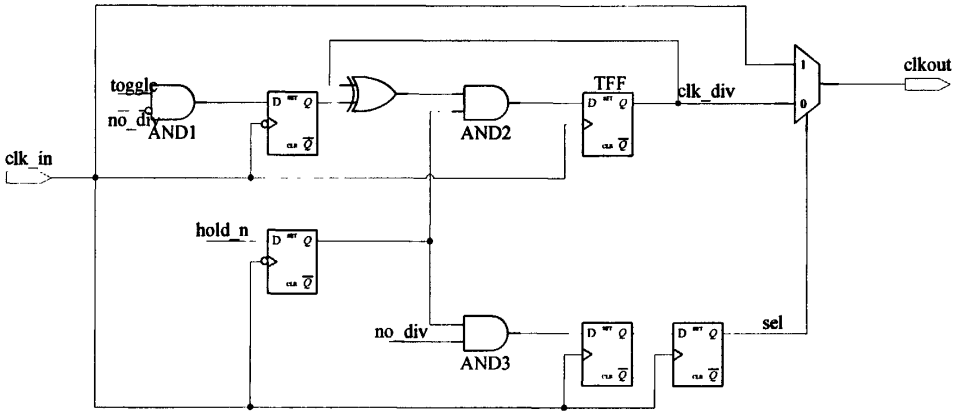


图 5-3 分频模块 clk_gen 的结构框图

其中，clk_in 为输入的参考时钟，clk_out 为输出的分频时钟，hold_n 为低电平有效的时钟关断信号，no_div 为高电平有效的分频废止信号，而 toggle 为触发器 TFF 的翻转使能信号，通过控制 toggle，可以控制输出时钟 clk_out 与输入时钟 clk_in 之间的频率比。

具体分析图 5-3 所示电路的逻辑功能：

- hold_n 为低电平时，与门 AND2 和 AND3 均输出低电平，不论 no_div 和 toggle 为何值，clk_out 都恒为低电平，即输出时钟被关断。
- hold_n 为高电平时，如果 no_div 为高电平，选择控制信号 sel 为高电平，多路选择器选择将输入时钟 clk_in 输出到 clk_out；而如果 no_div 为低电

平，选择控制信号 sel 为低电平，多路选择器选择将分频时钟 clk_div 输出到 clk_out。

- toggle 为高电平时，触发器 TFF 的值发生翻转，否则，触发器 TFF 维持原来的值。因此，通过控制 toggle，可以控制分频时钟 clk_div 与输入时钟 clk_in 之间的频率比。

根据上述分析，表 5-1 列出了在各种工作状态下输入控制信号与输出时钟的情况。

表 5-1 分频模块 clk_gen 的三种工作状态

	hold_n	no_div	toggle	clk_out
时钟关断	0	—	—	0
一分频	1	1	—	clk_in
X 分频(X>1)	1	0	1	~clk_out

可以看到，图 5-3 所示的电路中存在两条重汇聚路径，一是输入时钟 clk_in 经过多路选择器汇聚到输出时钟 clk_out，这条路径的延迟即为多路选择器的输入到输出的延迟；一是输入时钟 clk_in 经过触发器 TFF 和多路选择器汇聚到输出时钟 clk_out，这条路径的延迟即为触发器 TFF 的内部延迟（$T_{ck \to q}$）与多路选择器的输入到输出的延迟的和。其中，触发器的内部延迟为 0.3ns 左右，所以，上述重汇聚路径的延迟差异较小，插入少量的反相器或缓冲器即可实现平衡。

3.2 节中曾经提到，不论是使用标准单元库中的异或门，还是令逻辑综合工具自动用简单逻辑门搭建异或逻辑，从异或门的 A、B 输入端到输出端的延迟都存在较大的差异，导致时钟信号出现偏差。在此 clk_gen 模块中，虽然也使用了异或门，但是，该异或门并不在时钟信号的路径上，因此不会引入时钟偏差，不需任何特殊的处理。

5.3. 控制模块 clk_ctrl 的 RTL 设计

如图 5-2 所示，clk_gen 模块在 hold_n、no_div 和 toggle 的控制下产生分频时钟 clk_out；控制模块 clk_ctrl 则根据分频系数 divisor、门控使能信号 clk_en 以及复位信号 rst_n 等产生 clk_gen 模块所需要的控制信号 hold_n、no_div 和 toggle。

具体地说，clk_ctrl 模块将实现如下功能：

1）如果门控使能信号 clk_en 为低电平或者复位信号 rst_n 为低电平，则输出低电平的 hold_n 信号，将时钟分频器的输出时钟关断。

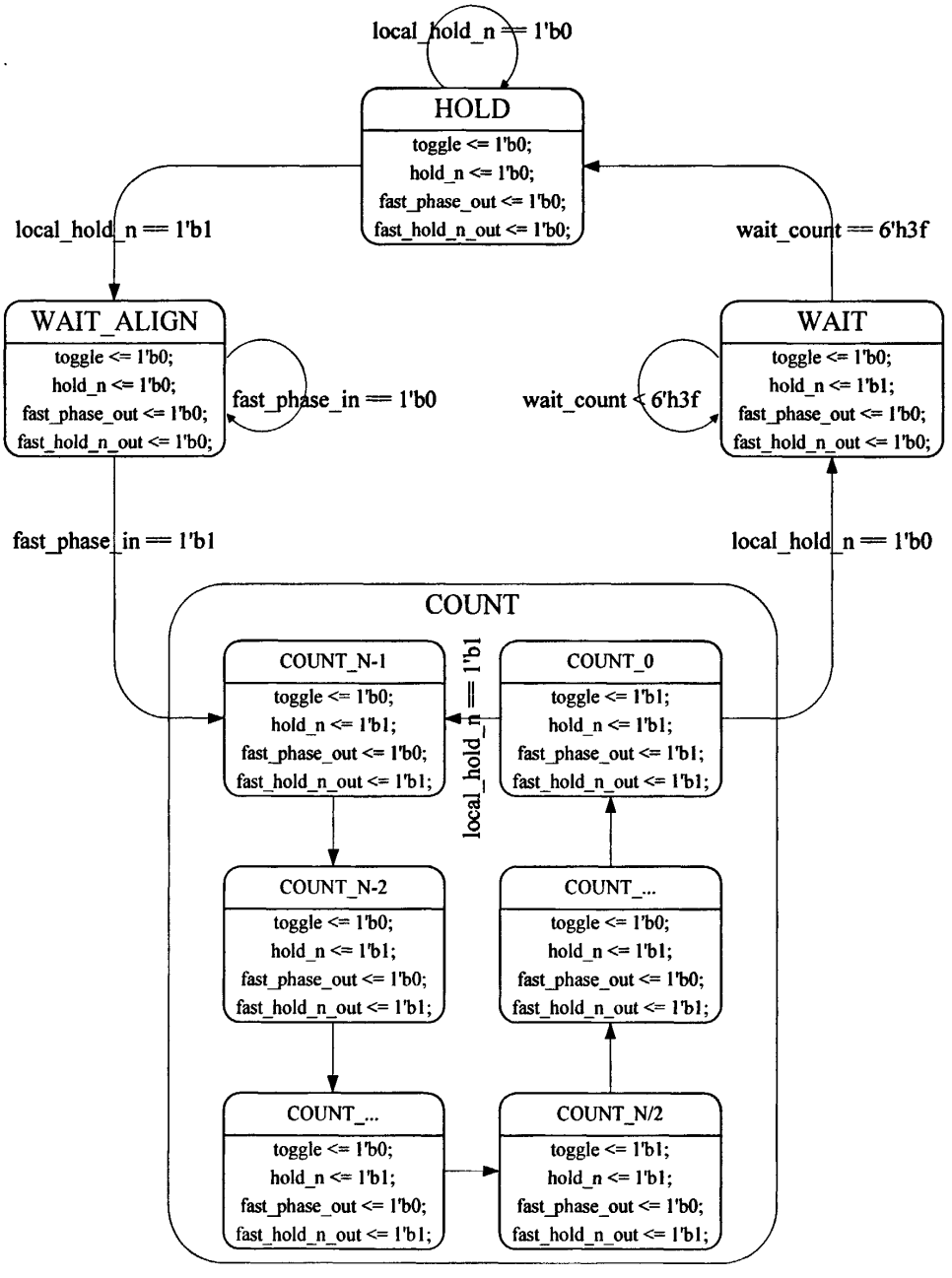
2）如果分频系数 divisor 等于 1，则输出高电平的 no_div 信号，废止分频逻辑，以原始输入时钟 clk_in 为输出时钟。如果分频系数 divisor 大于 1，则输出低电平的 no_div 信号，以分频时钟 clk_div 为输出时钟。

3）根据分频系数 divisor 的值，输出合理的翻转使能信号 toggle，使输入时钟 clk_in 与分频时钟 clk_div 的频率比为 divisor。

4）如果输出时钟 clk_out 与某一快时钟之间存在同步交互路径，需要保证这一同步时钟对的相位差恒为零。以 ve_clk 的时钟分频器为例，由于在视频解码引擎中存在核心工作时钟 ve_clk 与低速总线时钟 bclk32 的交互路径，因此，不论经历了多少次变频或门控，输出时钟 ve_clk 的上升沿应始终与低速总线时钟 bclk32 的上升沿对齐。为此，对于每一对同步时钟，快时钟的时钟分频器应向慢时钟的时钟分频器输出一个 fast_phase 信号，指明快时钟的上升沿与二者共同的参考时钟 clk_src 的上升沿的相对位置。在慢时钟的时钟分频器中，clk_ctrl 模块根据此信号控制翻转使能信号 toggle 的生成。

5）如果输出时钟 clk_out 与某一慢时钟之间存在同步交互路径，简单起见，在输出时钟 clk_out 被关断时，也将该慢时钟关断。为此，对于每一对同步时钟，快时钟的时钟分频器应向慢时钟的时钟分频器输出一个 fast_hold_n 信号，指明快时钟是否被关断，在慢时钟的时钟分频器中，clk_ctrl 模块根据此信号控制时钟关断信号 hold_n 的生成。

根据上述分析，no_div 的生成逻辑为：no_div = !(| divisor[DIV_WIDTH-1:1])。toggle、hold_n、fast_phase_out 的生成逻辑略为复杂一些，本文用图 5-4 所示的状态机来实现。



其中，local_hold_n <= rst_n && clk_en && fast_hold_n_in;

图 5-4 控制模块 clk_ctrl 中的状态机

下面详细描述图 5-4 所示的状态转移关系以及每一个状态的输出与动作。

- COUNT

此状态对应时钟分频器的正常工作状态。作用于本时钟分频器的输出时钟的

时钟关断信号 hold_n 与作用于相关慢时钟的时钟关断信号 fast_hold_n_out 均为高电平，即不关闭任何时钟。

设分频系数 divisor 等于 N。在此状态下，计数器在输入时钟 clk_in（亦即源头时钟 clk_src）的驱动下从 N-1 到 0 循环计数，当计数到 N/2 和 0 时输出高电平的翻转使能信号 toggle。这样,clk_gen 模块将根据 toggle 信号生成预期频率的分频时钟。当 N 为偶数时，该分频时钟具有 50%的占空比；当 N 为奇数时，该分频时钟的占空比为$1/2+1/2N$。对于 SK 系统芯片，除 DDR2 SDRAM 控制器以外，不存在任何下降沿触发寄存器。因此，对于除 DDR2 SDRAM 控制器的工作时钟（由另一个 PLL 单独生成）以外的时钟信号，占空比偏离 50%是没有问题的。

此外，当计数到 0 时，状态机同时输出高电平的 fast_phase_out 信号，向相关慢时钟的时钟分频器指明本时钟的上升沿与二者共同的源头时钟 clk_src 的上升沿的相对位置。该信号也就是本时钟与 clk_src 时钟之间的相位指示信号。

- WAIT

如果门控使能信号 clk_en 为低电平，或者复位信号 rst_n 为低电平，或者相关快时钟的时钟分频器通过 fast_hold_n_in 信号指明该快时钟即将被关断，那么，clk_ctrl 模块应输出低电平的时钟关断信号 hold_n，将本时钟分频器的输出时钟关断,并且，通过 fast_hold_n_out 信号指示相关慢时钟的时钟分频器关闭输出时钟。

考虑到完成慢时钟的关断需要更长的时间，在此状态下，令作用于相关慢时钟的时钟关断信号 fast_hold_n_out 为低电平，而令作用于本时钟的时钟关断信号 hold_n 为高电平。即暂时保留本时钟分频器的输出时钟，而首先去关闭相关慢时钟。

64 个输入时钟（clk_src）周期后,可以认为相关慢时钟已经被关断，状态机自动转入 HOLD 状态。

- HOLD

在此状态下，作用于本时钟的时钟关断信号 hold_n 与作用于相关慢时钟的时钟关断信号 fast_hold_n 均为低电平，本时钟分频器的输出时钟以及相关慢时钟均被关断。

如果门控使能信号 clk_en 与复位信号 rst_n 均为高电平，并且相关快时钟未被关断，则状态机转入 WAIT_ALIGN 状态，否则，继续处于此状态。

- WAIT_ALIGN

此状态是重新开始生成分频时钟之前的预备状态。

当本时钟分频器的输出时钟与某一快时钟之间存在同步交互路径时，需要保证这一对同步时钟信号的相位差恒为零，即本时钟的上升沿始终与相关快时钟的上升沿对齐。为此，恢复分频时钟的生成时必须参考该相关快时钟的相位指示信号。只有当来自相关快时钟的时钟分频器的相位指示信号 fast_phase_in 为高电平时，才允许转入 COUNT 状态重新开始生成分频时钟。

在上述状态机的控制下，控制模块 clk_ctrl 根据分频系数 divisor、门控使能信号 clk_en、复位信号 rst_n 以及相关快时钟的相位指示信号 fast_phase_in 与关断指示信号 fast_hold_n_in 产生分频模块 clk_gen 所需要的控制信号 hold_n、no_div 和 toggle，使其生成符合预期的分频时钟 clk_out。

值得注意的是，时钟关断信号 hold_n 的变化有可能导致时钟信号上出现毛刺，为此，应通过软硬件协同的操作流程避免毛刺被传递出去。这种软硬件协同控制的思想在已经在 3.3 节和 4.3 节介绍过，这里不再赘述。

5.4. 小结

为了消除时钟生成模块的固有时钟偏差，为后端实现提供一个较为理想的起点，本文对时钟生成模块的 RTL 实现进行了优化，使其具有更为平衡的电路结构。本章详细介绍了时钟生成模块的改进设计，包括改进的总体时钟结构以及时钟分频器中分频模块 clk_gen 与控制模块 clk_ctrl 的设计。可以看到，新的时钟生成模块中包含复杂的控制逻辑，也不能保证输出时钟具有 50%的占空比，这是平衡电路结构所需要付出的代价。

第6章 总结与展望

本文首先介绍了集成电路功耗管理领域的基本概念和常用技术，随后详细阐述了PKUnity－3（SK）系统芯片中实现的功耗管理技术。

由于时钟分布系统的结构与动态变频和时钟门控等功耗管理技术的实现密切相关，并且时钟生成模块的电路结构是否平衡直接影响到整个系统的时序性能，因此，设计具有平衡结构的时钟生成模块显得尤为重要。本文首先介绍了一种时钟生成模块的设计，包括RTL实现以及后端设计阶段为了平衡该时钟生成模块内部的各条时钟路径而进行的逻辑优化。随后，介绍了一种改进的时钟生成模块的设计。这一改进的设计在RTL实现阶段就考虑了同步时钟路径之间的平衡，从而减轻了后端设计的工作量和复杂度。

基于上述时钟生成模块，本文介绍了动态变频与时钟门控这两种功耗管理技术在PKUnity－3（SK）系统芯片中的具体实现。功耗管理模块（Power Manager）是实现这两种低功耗设计技术的核心控制模块，本文详细介绍了该模块的状态机设计以及与功耗管理相关的寄存器接口定义。

在PKUnity－3（SK）系统芯片中，主要采用了动态变频技术和时钟门控技术来降低系统的功耗，而未对电源电压进行任何控制。根据第2章的分析，降低电源电压是降低动态功耗的最有效的手段。此外，随着CMOS制造工艺的进步，静态功耗在系统总功耗中所占的比重越来越大，待机模式下关闭电源电压对降低功耗的影响将越来越不可忽视。基于上述考虑，在后续工作中，将着重研究动态变压变频技术以及局部电源电压的切断与恢复。为此，首先需要选择满足需求的Power IC芯片；其次需要设计功能增强版的功耗管理部件，通过控制Power IC芯片来实现对系统芯片的各电压域的电源电压的控制；尤其重要的是，需要对系统芯片进行合理的电压域划分，并在两个电压域的交界处进行电平转换和隔离处理，这种处理以及相应的验证工作是动态变压变频和切断局部电源电压的难点。

参考文献

[1] Jan M. Rabaey, Anantha Chandrakasan, Borivoje Nikolic. Digital Integrated Circuits A Design Perspective. Prentice Hall. 2003.

[2] 陈杰，基于贝叶斯分类的动态功耗建模与评测研究，北京大学博士研究生学位论文，2008。

[3] 北京大学微处理器研究开发中心，北大众志 CPU 及系统芯片的研究进展与展望，2009。

[4] Michael Keating, David Flynn, Robert Aitken, Alan Gibbons, Kaijian Shi. Low Power Methodology Manual For System-on-Chip Design. Springer. 2008.

[5] 唐杉，徐强，王莉薇，数字 IC 设计——方法、技巧与实践，机械工业出版社，2006。

[6] Luca Benini, Alessandro Bogliolo, Giovanni De Micheli. A Survey of Design Techniques for System-Level Dynamic Power Management. IEEE Transactions on Very Large Scale Integration(VLSI) Systems. 2000.

[7] Osman S. Unsal, Israel Koren. System-Level Power-Aware Design Techniques in Real-Time Systems. Proceedings of the IEEE. 2003.

[8] Cedric Lichtenau, Mathew I Ringler, Thomas Pfluger, et al. PowerTune: Advanced Frequency and Power Scaling on 64b PowerPC Microprocessor. IEEE International Solid-State Circuits Conference. 2004.

[9] Kevin J. Nowka, Gary D. Carpenter, Eric W. MacDonald, et al. A 32-bit PowerPC System-on-a-Chip With Support for Dynamic Voltage Scaling and Dynamic Frequency Scaling. IEEE Journal of Solid-State Circuits. 2002.

[10] Bishop Brock, Karthick Rajamani. Dynamic Power Management for Embedded Systems. Proceedings of the IEEE SoC Conference. 2003.

[11] Sung-Mo Steve Kang. Elements of Low Power Design for Integrated Systems. ACM. 2003.

[12] Michael D. Ciletti. Advanced Digital Design with the Verilog HDL. Prentice Hall. 2003.

[13] 徐振林等译，Verilog HDL 硬件描述语言，机械工业出版社，2000。

[14] Michael Keating, Pierre Bricaud. Reuse Methodology Manual For System-on-a-Chip Designs. KluWer Academic Publishers. 2000.

[15] Mohit Arora. Clock Dividers Made Easy. SNUG Boston. 2002.

[16] Mike Stein. Crossing the abyss: asynchronous signals in a synchronous world. www.edn.com. 2003.

[17] Clifford E. Cummings. Synthesis and Scripting Techniques for Designing Multi-Asynchronous Clock Designs. SNUG San Jose. 2001.

[18] Clifford E. Cummings, Don Mills, Steve Golson. Asynchronous & Synchronous Reset

Design Techniques – Part Deux. SNUG Boston. 2003.

[19] 许浒，王逵，方昊，吴费维，时钟产生模块的分支平衡设计，SNUG，2008。

[20] Taiwan Semiconductor Manufacturing Company, Ltd. PG13A1LV3-1000MHz Clock Generator Datasheet. www.tsmc.com. 2007.

[21] 北京大学微处理器研究开发中心，北大众志功耗管理部件用户手册，2009。

[22] 北京大学微处理器研究开发中心，北大众志功耗管理部件设计规格说明书，2009。

[23] ARM Limited. AMBA Specification(Rev 2.0). www.arm.com. 1999.

[24] Luca Benini, Giovanni De Micheli. System-Level Power Optimization: Techniques and Tools. ACM Transactions on Design Automation of Electronic Systems. 2000.

[25] Synopsys, Inc. Power Compiler User Guide. solvnet.synopsys.com. 2005.

[26] Synopsys, Inc. Design Compiler User Guide. solvnet.synopsys.com. 2005.

[27] Synopsys, Inc. PrimeTime User Guide. solvnet.synopsys.com. 2005.

[28] Jonathan Owen, Maurice Steinman. NorthBridge Architecture of AMD's Griffin Microprocessor Family. IEEE Computer Society. 2008.

[29] Advanced Micro Devices, Inc. AMD Geode CS5535 Companion Device Data Book. www.amd.com. 2005.

[30] National Semiconductor Corporation. Geode CS5535 I/O Companion Multi-Function South Bridge. www.national.com. 2003.

[31] Intel Corporation. Intel PXA27x Processor Family Developer's Manual. www.intel.com. 2004.

致 谢

本篇论文的完成，首先要衷心感谢我的导师王克义教授。攻读硕士研究生的三年中，王老师不但对我的学习和工作给予指导和启发，生活上也颇多关怀照顾。王老师严谨求实的治学态度、丰富渊博的专业知识和平易近人的处事作风让我受益菲浅。非常感谢他对我的指导和帮助。

感谢程旭教授，程老师献身科研的无畏精神和强烈的社会责任感对我的人生观和价值观产生了深刻的影响，他对事业的执着和专注激励着我不断进取。感谢佟冬老师，佟老师在工程实践中对我的悉心指导和严格要求使我的专业水平得到很大的提高，在我遇到挫折时的嘘寒问暖更让我感激不已。感谢易江芳老师在学业上给予的指导和生活上给予的关心。特别感谢陆俊林老师一直以来的帮助和鼓励，他追求完美的作风和认真踏实的态度是我做人做事的榜样，非常感谢他给予的许多珍贵意见和建议。

感谢冯毅老师。他带领我脚踏实地地完成一个个工程项目，他严谨求实的科研作风和精益求精的工作态度直接影响了我的职业素养的养成。更感谢冯毅老师对我的职业发展提出的许多中肯意见和建议，他的深邃洞察力与高远见地为我拨云见雾，帮助我选定了适合自己的职业道路。我在研究生阶段的点滴成长离不开冯毅老师的无私帮助，能够受教于他是我的荣幸。

感谢实验室的各位师兄师姐。黄萍萍师姐对我的学习、工作和生活都给予了无微不至的关怀，使我迅速地融入了实验室这个大家庭。黄侃师兄在若干工程项目中对我不厌其烦地指导和点拨，他对专业知识的把握和灵活的思维方式令我钦佩不已。刘丹师姐引导我开始对功耗管理领域的研究和实践，帮助我确定了本文的研究方向。陈守亮师兄、郭睿师姐、周政师兄、赵雅师姐、曹玉婷师姐、崔静师姐、张涛师兄更是在工程实践、职业规划、日常生活等诸多方面给予了真诚的帮助和关心。

感谢刘洋同学在功耗管理项目中和本文的撰写过程中给予的帮助，功耗管理部件在 SK 系统芯片中的成功实现有他的一份功劳。

感谢 06 级的所有硕士和博士们。三年来，我们同甘共苦，并肩前进，你们的勤奋与拼搏带给我活力，你们的亲切和热心让我感受到温暖，和大家一起奋斗的岁月是我无比美好的回忆。

感谢北京大学微处理器研究开发中心的所有老师和同学们，尤其是 SoC 组的兄弟姐妹们，很幸运能够成为这个富有魅力的团队的一员。

深深地感谢我的父母，你们无私的奉献和无尽的关爱是我努力奋斗的最大动力。