摘要

随着近年来半导体制造工艺的不断进步及电路的大规模化，集成电路设计中遇到了诸如功耗，时钟偏移及连线延迟增大等难题。这时异步电路由于其低功耗，高性能及避免时钟偏移等独特优势引起了设计者的关注。对异步电路设计技术及单元电路的深入研究具有重要的理论意义及实践价值。

论文针对目前集成电路设计中遇到的挑战，介绍了异步集成电路的独特优势、发展历程及相关 EDA 工具。文中对异步电路设计技术的时序模式、编码方式及信号协议作了研究，重点探讨了四种异步握手协议。设计实现了异步 C 基本单元的单轨与双轨等多种结构，对各种电路做了详细的仿真及优化，获得了良好的电路性能。通过对比分析了各种结构的适用环境；分析探讨了 merge, fork 及 join 等异步握手基本单元；设计实现了异步单双轨及混合加法器，并通过仿真对每种结构的特点和应用环境进行了分析研究。

关键词： 异步电路　低功耗　握手协议　C-单元

Abstract

With a fast development in semiconductor fabrication technology, IC has the tendency of larger scale and higher speed. But simultaneously, some critical problems in IC design emerged, such as the large power dissipation, the clock-skew and the increasing percentage of the wire delay in the delay of system. Since asynchronous design has advantages of low-power, high-performance and no clock skew, there has been a revival of interest on asynchronous circuit research. The research on the asynchronous design methodologies and standard cells become very necessary.

The critical problems in IC design emerged at present were introduced in this thesis, and then the prominent advantage, the history and EDA tools of the asynchronous logic were summarized. The asynchronous design methodologies, includes the asynchronous sequence mode, the coding mode and the signaling protocol, were explained. Four handshake protocols were studied as an emphasis. Various single-rail and double-rail implementations of the c-element were designed. After careful simulations, the performance of every implementation was analyzed and optimized. The suited environments of every implementation were discussed. The merge, fork and join cells used in handshake circuit were analyzed. The single-rail, double-rail and hybrid asynchronous adders were implemented in the paper, and the characters of every implementation were also discussed.

Keywords: **asynchronous circuit** **low power** **handshake protocol** **c-element**

创新性声明

本人声明所呈交的论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果；也不包含为获得西安电子科技大学或其它教育机构的学位证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

本人签名：李俊志　　日期：2006.1.20

关于论文使用授权的说明

本人完全了解西安电子科技大学有关保留和使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权单位属西安电子科技大学。本人保证毕业离校后，发表论文或使用论文工作成果时署名单位仍然为西安电子科技大学。学校有权保留送交论文的复印件，允许查阅和借阅论文；学校可以公布论文的全部或部分内容，可以允许采用影印、缩印或其他复制手段保存论文。(保密的论文在解密后应遵守此规定)

本学位论文属于保密，在____年解密后适用于本授权书。

本人签名：李俊志　　日期：2006.1.20

导师签名：[illegible]　　日期：2006.1.20

第一章 绪论

1.1 目前集成电路设计中面临的挑战

随着目前半导体制造工艺及电路设计技术的不断进步，集成电路朝着电路规模更大，速度更快的方向飞速发展。在此同时，也带来了一系列急需解决的难题，包括：电路大规模化带来的电路功耗增大；全局时钟引起的时钟偏移；工艺尺寸减小带来的连线延迟等[1]。

在过去，集成电路设计者最关心的电路参数是面积、速度、性能、成本和可靠性，然后才是电路的功耗的问题。然而近年来由于个人电脑，手机、PDA 及消费数码等便携式电子器件的广泛应用及电路大规模化带来的芯片散热问题，电路的功耗已经和面积、速度一样，成为设计者考虑的首要因素。与之相对应，超大规模集成电路和系统的设计重点也渐渐地由高速度转向低功耗设计，低功耗设计方法成为现在超大规模集成电路设计中一个热门的课题[2][3]。

同时全局时钟的存在带来的时钟偏移及深亚微米工艺下连线延迟的增加也为集成电路设计带来了很多比较棘手的问题，大大增加了电路设计的复杂性和时序验证的困难。在规模较大的集成电路中分布一个高频的全局时钟难度越来越大，有的电路已开始采用“混合时序系统”的设计方法，但在同步系统中，由于各个模块时钟相位和频率的不同，如何将各个模块连接起来也是比较麻烦的问题。

针对这些问题现在已有许多的解决方案，包括降低特征尺寸、设法降低电路翻转率，优化算法等。这些方案可以在一定程度上降低电路的功耗，改善电路性能。但很多情况下，效果都是很有限的，而且大都是速度与功耗的折衷考虑。这种情况下，异步电路由于其独特的优势引起了设计者的重视。

1.2 异步电路设计技术发展状况

同步电路是当今集成电路设计的主流，大多数的数字电路系统都采用同步设计形式，在一个全局时钟的控制下进行工作。与之相反，异步电路不采用全局时钟，各模块通过局部握手信号进行模块间的通讯。由于异步电路能较好的解决在现今 IC 设计中的许多难题，最近对它的研究成了集成电路设计中的一个新课题。在过去的十年中，对异步电路设计的研究重点已经从原来的纯理论研究转向 VLSI 电路和系统中异步电路的实际应用研究。工业界的项目和学术界的研究都证明了异步电路设计的可行性及由此带来的很多电路性能上的改进，尤其是电路功耗的

降低。事实上，1998 年，市场上就已经出现了一些完整的异步 IC 的芯片[4]。

异步电路采用了一种数据驱动的控制机制，它通过使用大量本地握手信号来完成整个电路的时序控制[5]。正是此种控制机制的采用决定了异步电路独特的优势：

1) 低功耗。同步电路在整体时钟控制下工作，时钟工作频率必须满足最大负荷的要求，造成功耗浪费。异步电路则由数据驱动，仅在需要处理数据时才消耗能量，具有低功耗的潜力
2) 潜在高性能。同步电路工作时需要考虑电路的最坏情况延迟，而异步电路的性能则由电路的平均延迟决定，理论上比同步电路可以达到更高的速度。
3) 避免时钟偏移。异步电路用大量本地时序控制信号取代整体时钟，从根本上避免了时钟偏移问题。
4) 模块化特性。异步电路使用握手信号进行模块内部与模块间的通信，采用相同握手协议的电路模块可以方便地直接互连，而且单独模块也能够很方便地优化升级。
5) 电磁兼容性好。异步电路辐射频谱含能量少且分散性非常好。

异步集成电路应用中，最突出的两个优点是低功耗和潜在的高性能，但这些优点不是无条件的。异步集成电路使用本地握手信号进行时序控制，需要增加一些电路模块来完成这些工作。这些附加的电路模块往往会对功耗和性能产生负面影响。因此，发挥异步集成电路低功耗和高性能的特点需要合适的应用对象，比如，在待机很频繁的场合容易实现低功耗；在平均性能与最差性能相差较大的情况下，有利于实现高性能[6]。

虽然目前已有一些异步处理器如 AMULE、MiniMIPS、微控制器异步 80C51、异步 PIC 微控制器（复旦大学）等出现，但由于异步集成电路现在还没有统一的设计方法和成熟系统的 EDA 工具，现有的异步集成电路的设计方法，大部分是时针对集成电路中某些局部问题的解决方案。而且已经实现的异步微处理器都不同程度地采用了全定制设计流程，自动化程度低，开发时间长。而当前流行的同步集成电路设计一般采用基于标准单元库的半定制设计流程，具有方法简单、工具成熟、自动化程度高等优点。如何用基于标准单元库的半定制流程实现异步集成电路设计，是异步集成电路设计中的难题之一。首先需要解决的问题就是建立异步标准单元库，供前端综合工具和后端布局布线工具的使用。现有的标准单元库一般只针对同步集成电路设计，采用静态 CMOS 工艺，缺乏对异步集成电路设计中特殊电路结构（如动态电路、C 单元）的支持。因此对异步电路常用单元电路的研究就显得非常重要。

1.3 论文的主要工作方向及结构

本文主要介绍了异步电路的独特优势，对异步逻辑的时序模式、编码方式及通讯协议做了深入的探讨研究。设计实现了异步 C 单元、Merge、fork、join 及异步加法器等基本单元，并对各种结构进行了详细的仿真及优化。

文章的结构主要是根据作者在论文过程中的工作顺序安排的。本文大概可以分为五章：

第一章为绪论，简单介绍了目前集成电路设计遇到的难题及异步电路设计技术的发展状况；第二章主要是针对目前 IC 设计中面临的功耗，时钟偏移与工艺尺寸减小等带来的问题，阐述了异步电路的独特优势，并解释了为什么异步电路设计方法可以解决这些难题；第三、四章是本文的重点。第三章主要研究了异步电路的设计技术，包括异步时序模式、单双轨信号编码方式与四相及两相信号协议等。对四种异步握手通信协议作了重点探讨。阐述了异步集成电路发展的历史，相关的 EDA 工具及异步微处理器的发展现状；第四章是对异步电路中常用基本单元电路的研究：设计实现了异步 C 单元的单轨与双轨多种实现结构，从延时、功耗等参数入手对各种结构进行了详细的仿真和优化，并通过对比，分析了各种结构的性能特点及适合的应用环境；研究了 merge, fork, join 等异步握手电路基本单元，对各种单元的结构及功能进行了探讨；实现了单轨、双轨及混合异步加法器，分析探讨了每种实现结构的工作方式及特点，同样地也对各自的应用环境作了讨论；第五章为总结与展望，主要是对本文的工作做了总结，并对异步电路以后的发展作了展望。

第二章 IC 设计面临的挑战及异步电路的特点

随着社会需求和半导体制造工艺的发展，集成电路朝着电路规模更大，速度更快的方向飞速发展。在此同时，集成电路的设计中也出现了一系列急需解决的问题，包括：电路的功耗增大：全局时钟引起的时钟偏移；工艺尺寸减小带来的问题等。异步逻辑电路采用了不同与同步电路的设计方法，它不存在全局时钟，而是采用了一种数据驱动的控制机制，各个模块互相之间通过握手信号进行通讯。通过异步逻辑来设计电路是解决上述这些问题的较好的手段[7]。

2.1 IC 设计面临的挑战

2.1.1 集成电路的功耗问题

在过去，集成电路设计者最关心的电路参数是电路的面积，速度，性能，成本和可靠性，电路的功耗是其次的考虑因素。然而近几年来，情况发生了很大的变化。电路功耗成了与面积，速度同样重要的一个参数。与之相对应，超大规模集成电路和系统的设计重点也渐渐地由高速度转向低功耗设计。低功耗设计方法成为现在超大规模集成电路设计中一个很热门的课题[8][9]。

2.1.1.1 低功耗设计的必要性

A. 便携式电子器件的广泛应用

低功耗设计最显为人知的目的是用于电池供电的便携式器件。便携式器件在近几年得到了迅猛的发展，已经深入到人们的日常生活中，例如便携式个人电脑，手机、PDA，手掌游戏机，个人数字低高保真音响，全球卫星定位系统（GPS），便携式数字摄像机和数字相机等等。这些电子器件给人们的生活带来了极大的方便，用户通常要求产品体积小，重量轻，电池可使用尽量长的时间。然而这些便携式器件供电电源的发展速度远远不及电子系统的发展速度，这就使得电路的低功耗设计变得很重要 [10][11]。

B. 散热问题

随着半导体制造工艺的发展，在将来非便携式器件也有可能不得不应用低功耗技术。由于集成电路制造的特征尺寸以每年 12%的速度递减，在芯片的单位面子上可集成的管子数大大增加，芯片上可集成的电路规模也将变大，因此实际芯片消耗的功耗就会增加。“SCALING LAW”告诉我们，每当芯片面积下降 1/2，

芯片的功耗将增加 1.8 倍[1]。

电路功耗的增加引起芯片上热量也不断增加，芯片的散热就成了问题。解决散热问题所花费的代价越来越大。同时芯片温度的上升也会导致电路性能和可靠性的下降。为了避免这些问题，在高性能系统中将必须使用到类似与在便携式器件中用到的低功耗技术。

C. 环境保护的需要

美国环境保护局提出了一个个人电脑的低功耗标准，估计能减少相当于五百辆汽车排放的二氧化碳引起的环境污染。同时也可节约大量的资金。

2.1.1.2 CMOS 集成电路中功耗的来源

A. 电容充放电动态功耗

与双极型工艺大部分的功耗是静态功耗不同，CMOS 电路所消耗的功耗主要是电容的动态充放电功耗。若用α表示每个时钟内输出翻转（包括 0→1 和 1→0）的次数（也称为翻转活动率），f表示电路的时钟频率，C_L表示电路的负载电容，V_{DD}表示电路的电源电压，电容充放电动态功耗的大小是：

$$P = 0.5\alpha * C_L * V_{DD}^2 * f \tag{2-1}$$

由于在深亚微米(deep submicron)集成电路中电容动态充放电功耗是电路总功耗中最大的一部分，因此大部分的低功耗方法集中在降低动态功耗上。

从动态功耗的数学表达式我们可以看出，决定动态功耗大小的主要有三个参数：负载电容、电源电压和时钟频率。对于负载电容，我们可以在设计电路时调整结构或者通过增加 buffer 的方法使得每一级输出不要带过大的负载；对于电源电压，由于集成电路工艺的发展，电源电压已经可以降到晶体管的阈值附近。同时随着特征尺寸的减小，电路的电容将会减小。这几种方法可以降低电路的功耗。

然而电路的发展方向是速度越来越快，所以f是肯定会增大的，这样又增大了电路的功耗。由于全局时钟的负载相当大，且以很高的频率一直在翻转，全局时钟的功耗是数字系统动态功耗中很大的一部分。据估计，全局时钟的功耗可能多达整个系统功耗的 40%。随着工艺的发展，电路规模和时钟的频率还将不断提高，全局时钟消耗功耗过大将成为系统降低功耗的一个大阻碍。

B. 短路电流功耗、漏电流功耗及静态偏置功耗

这三种功耗只是 CMOS 电路功耗中的小部分。以下分别解释：

短路电流功耗是当 P 管与 N 管都导通时，从电源到地的通路上的短路电流所引起的功耗。对大多数的集成电路来说，短路电流功耗大约占电路动态功耗的 5%-10%；

漏电流功耗有两个来源，一是在 MOS 管衬底与源极，漏极所形成的寄生二极管中的反偏电流产生的功耗；二是 MOS 管闭合时沟道中仍存在着的亚阈值电流引起的功耗。随着集成电路电源电压和阈值电压减小的趋势，亚阈值电流占总电流的份额将会变得更显著；

C. 静态偏置功耗

集成电路由 NMOS 转向 CMOS 的趋势使大多数电路的静态偏置功耗趋向于 0，但仍有某些静态偏置在相应的电路对电路功耗的降低及面积的减少具有很大的好处。因此，在某些复杂的逻辑电路功耗中，有时会采用具有静态偏置功耗的 Pseudo-NMOS，它在功耗、面积上都比 CMOS 具有更大的优势[12]。

2.1.1.3 现有的低功耗设计方法简介

从电路设计的工艺级、晶体管级、电路级到结构算法级上，都有对应的低功耗设计方法。各个设计级别上节省的功耗的方法应该组合起来，才能最大限度的降低电路功耗。

A. 工艺级的低功耗方法和晶体管级的低功耗方法

工艺级的低功耗方法主要是降低特征尺寸。随着特征尺寸的减少，电路的电容将会减少。粗略的估计显示，每个门每次的翻转所耗费的功耗与特征尺寸的三次方成正比。同时，特征尺寸的减小也使得在单个芯片上可集成的管子数增加。这就使许多面积很大的、原本不得不分割在好几个芯片上的电路可集成在单个芯片上。由于芯片内部的连线比起芯片之间的连线，功耗将会小许多，且速度也快，因此，该电路的总功耗将会降低很多。

晶体管级的低功耗方法主要是设计合理的晶体管宽长比，建立一个低功耗的标准单元库。一方面，我们希望标准单元的输出变化尽可能的快，这样能减小输出变化时，中间值引起的短路电流功耗。另一方面，我们又希望晶体管的尺寸尽可能的小，以减小电容充放电引起的动态功耗。因此，低功耗电路中晶体管的最佳尺寸应是以上两者的权衡。

B. 电路级的低功耗设计方法

电路级的低功耗设计方法很多，针对各种不同的电路有不同的策略。但总体思想是降低电路翻转率，主要的电路级低功耗设计方法有：

1) 逻辑优化，将电路工作时的最典型的情况设计成具有较低的翻转率和较低的功耗；
2) 并行处理，将一个模块的工作有 N 个速度较慢的模块同时来处理，功耗将会减少 N 倍；
3) 预计算，即预先进行那些可能直接决定结果的电路的工作，以避免一些对最

后结果没有影响的无效的操作引起的功耗；

4) 采用门控时钟，它能比较粗略地将一些此时不需要工作的电路与驱动它的全局时钟断开，节省了此时这部分电路中锁存器的翻转和组合电路的无效翻转；

5) 采用“SLEEP”模式，若是观察到电路的工作负载很小，则可降低时钟频率，降低电源电压，让电路在满足要求的情况下尽可能的以低功耗模式工作。若是观察到电路被长时间闲置，则可进入“SLEEP”模式，切断时钟，将动态功耗降到0。此方法的缺点是变化时钟频率和电源电压的过程较为复杂，不方便；

6) 采用异步逻辑来实现电路，去除全局时钟。

C. 结构级及算法级的低功耗设计方法

结构级及算法级的低功耗设计方法对集成电路的功耗的降低是各个级别中最显著和有效的。对微处理器和微控制器来说，结构级上的低功耗方法是设计一个具有低功耗特性的体系结构，而算法级上的低功耗方法是设计一个具有低功耗特性的指令集。

2.1.2 集成电路中全局时钟引起的问题

A. 全局时钟消耗的功耗过大

由于全局时钟的负载相当大，且以很高的频率一直在抖动，全局时钟所消耗的功耗是数字系统动态功耗中很大的一部分。据统计，全局时钟的功耗可能多达整个系统功耗的40%。随着工艺的发展，电路规模和时钟的频率还将不断提高，全局时钟消耗功耗过大将成为系统降低功耗的一个大阻碍。

B. 时钟偏移问题(Clock Skew)

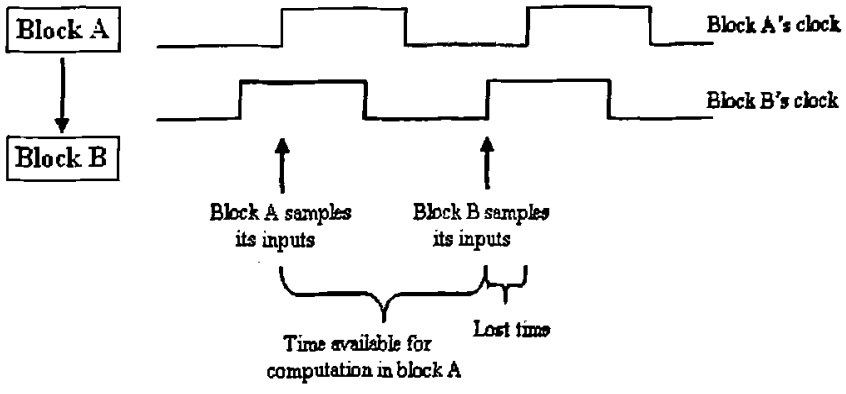


图 2.1 时钟偏移(Clock Skew)所损失的计算时间

在同步系统中，各个模块对时钟线的负载各不相同，并且通常处在时钟线上不同的位置。这使得时钟信号到达各个模块的延迟时间不同，各个模块的所获得时钟信号就各不相同。这就是“时钟偏移”(Clock Skew)。

如图 2.1，若一个模块与另外一个时钟信号略早于它的模块进行通讯，则此模块必须在那个较早的时钟到达前将输出数据就准备好，这样，后一个模块才能读取到有效的输出。因此第一个模块的实际计算时间要比一个时钟周期略短，存在一个时间上损失。

2.1.3 集成电路中工艺尺寸减小

随着半导体工艺的发展，尤其是深亚微米工艺的发展，集成电路逻辑门上的延迟不断降低，连线上的延迟占全部延迟的比重越来越大，众所周知，现在连线延迟超过门延迟成为电路的主导延迟。0.18 微米以下铝工艺和 0.13 微米以下铜工艺中连线延迟都超过了门延迟。90 纳米时，连线延迟甚至占总延迟的 75%。

除了整体延迟外，在 0.18 微米以下工艺中，信号之间的耦合作用会产生信号完整性(Signal Integrity)问题。延迟是连线负载和线驱动的函数。在 0.25 微米以上时，主要连线电容是和地之间的耦合电容，它和连线的长度成比例，线长增加一倍，那么耦合电容就增加一倍。到了 0.18 微米时，特定的一个连线和它邻近的连线之间的耦合电容变成了主要耦合电容，电容取决于局部连线的几何形状，在很多的情况下，取决于临近的连线。

同时，由于连线的电容的增大，在规模较大的集成电路中分布一个高频的全局时钟越来越难，有的电路已开始采用“混合时序系统”的设计方法，即在整个芯片中，不采用单一时钟频率的全局时钟，而是用几个频率不同的局部时钟。在同步系统中，由于各个模块时钟相位和频率的不同，如何将各个模块连接起来也是比较麻烦的问题。

随着工艺尺寸的减小和电路规模的增加，芯片中会出现 IR 压降问题。IR 压降主要是分布在电源线和地线网络中的电阻产生的。因为电阻随着尺寸的减小而增加，当电源电压降低或连线加长时，IR 压降会更加恶化。一项针对 0.18 微米以下芯片设计的研究表明：大约 20%的第一次投片失败仅仅是因为 IR 压降过大[13]。

此外工艺尺寸的减小还出现了一些涉及复杂性、物理效应和可制造性的新的技术问题，诸如串扰和耦合、电迁移、数模整合、系统信号传输等。

2.2 异步电路的特点

从集成电路设计之初到现在，同步设计因为概念简单、设计方便、有成熟的

EDA 工具的支持等因素，一直是集成电路设计的主流方案。当今大多数的数字系统也是用同步电路来设计的。但随着电路规模的不断扩大和制造工艺的进一步发展，电路设计中出现了许多对同步方法来说比较棘手的难题。因此异步电路设计方法引起了设计者的重视，由原来的纯粹理论探索转向了实际应用研究。在过去的十几年中，工业界的项目和学术界的研究都证明了异步电路设计的可行性及异步电路与同步电路比较很多电路性能上的改进。

与同步电路相比较，异步电路设计有着一系列的优势。包括：1)低功耗；2)潜在高性能；3)避免时钟偏移；4)模块化特性；5)更好的电磁兼容性[3][14]。

2.2.1 异步电路具有低功耗的特点

A. 异步电路中没有全局时钟消耗的功耗

随着特征尺寸的减小和电路规模的增加，同步电路中全局时钟的负载越来越大，消耗在时钟线上的功耗也越来越大。异步电路不需要全局时钟，各模块之间用负载较小的局部握手信号相互通讯，因此减少了消耗在时钟上的功耗，也大大降低了电路的功耗。

B. 异步逻辑速度可更快

无论是同步还是异步逻辑，电路通常都是由几个被寄存器单元划分开的组合电路块组成的，也就是通常所说的“寄存器-云”结构。其中组合电路的速度与寄存器的速度在同步电路和异步电路中大致相等。但在同步电路中，全局时钟的速度必须满足组合逻辑块中延时最长的一块在最慢的输入数据时的处理速度。这样就导致较快的模块在时钟周期的部分时间里是处在闲置状态。

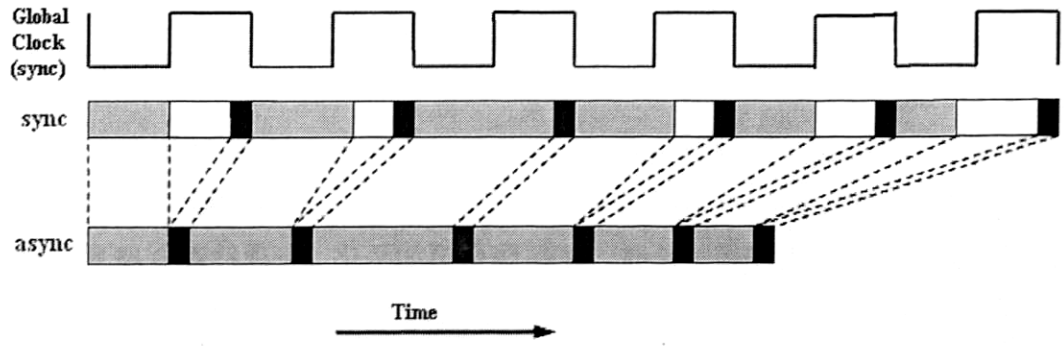


图 2.2 同步逻辑与异步逻辑的速度比较

. 图 2.2 的上半部分即是以上的这种情况。其中灰色部分表示组合逻辑运算的延迟时间，黑色表示寄存器的延迟，白色区域表示此模块处于闲置状态。由图可见，全局时钟的周期必须根据电路中最慢的速度来设置。

而在异步电路中，没有全局时钟，用各模块各自给出的握手信号来表示此模块已开始或结束运算。因此，如上图的下半部分所示，只要此模块已完成此时的工作，它就可立即进行下一个任务的工作，而无须进入闲置状态。因此，在异步电路的工作中，各个周期之间是么有闲置时间的。这就提高了整个电路系统的运行速度。

异步电路速度快的特点在电路的速度与输入数据有关的情况下更为明显。此时，同步电路的全局时钟大于最慢情况的数据的延迟，而此类数据实际上只占所有输入数据的很小一部分。异步电路的操作速度则是所有输入情况的一个平均值。

由于异步电路比相应的同步电路速度快，我们可以降低异步电路的电源电压，使得两者的操作速度相等，电压的降低无疑将大幅度地减少异步电路消耗的功耗。

C. 异步电路中无效的翻转大大减少。

在同步电路中，电路是这样工作的：全局时钟的触发沿触发了所有的触发器，电路中的数据发生变化，组合电路开始工作。这其中，有的组合电路的这个时钟周期的输出结果对整个电路的功能没有影响，属于无效工作；有的组合电路在形成有效输出前要经过好几次的变化，产生许多毛刺，但在下一个触发沿来到之前，输出已稳定为有效输出。这些无效翻转对电路的功能没有影响，但大大增加了电路消耗的功耗。

而在异步电路中，各模块只有在需要工作时才工作，否则处在闲置状态，并不进行无谓的翻转，并且异步电路中有的毛刺会引起电路的误操作，因此设计者在设计时要尽量去除电路里的毛刺。这样毛刺引起的无效翻转也大大减少。这使得异步电路工作时电路里的无效翻转大大减少，功耗也相应降低。

虽然同步电路也可采用通过“门控时钟”来避免无效的翻转，即用一个门来控制不将全局时钟信号输给不需工作的模块，但这也引起了一些问题。首先，电路中增加的门增大了电路规模，加大了时钟线上的负载；同时进行时钟控制的门引入的延迟将使得电路中的时钟偏移（Clock Skew）的问题更加严重，全局时钟引起的问题更加显著。

D. 异步电路面积更小

异步电路在有些应用中可显著减少电路的面积，这使得电路的负载电容减小，动态充放电引起的功能功耗也随之减少。由于异步电路是与输入的“平均情况”有关，同步电路的速度是与输入的“最坏情况”有关，因此在某些情况下，异步采用面积较小但速度较慢的设计方案和同步电路采用的面积较大，速度较快的方案实际性能是相同的。此时同步情况下采用的设计方案在输入的最坏情况下的性能比异步电路采用的方案要好，但输入最坏情况只占所有情况的很小一部分，同步电路因此损失掉的大量的面积无疑是不合算的。

E. 异步电路更适合变需求系统

异步电路的低功耗优势在变需求系统中表现得尤为明显。变需求系统在现今使用的非常广泛，例如大部分的电子计算机，随着用户进行操作的任务不同，它在运算需求上有着一个很大的波动。在某些应用中，微处理器常常对某些特定的事件做出反应，开始工作，过一段时间，工作完毕后，微处理器又进入闲置状态。

在这种情况下，同步微处理器采用了粗粒度的方法来减少系统的功耗。如笔记本电脑，它检测用户的使用情况，并在不同的情况下通过降低时钟频率或关断时钟进入不同的“睡眠状态”。由于全局时钟的转换是个很复杂的过程，故不得不采用此粗粒度方法。

异步电路的情况就完全不同。在没有任务时，异步电路中握手信号（Request，Acknowledge）就不变化，与之相对应，电路也停止翻转，消耗功耗为 0。与同步相比，它对系统运算需求的变化反应显然更迅速，是细粒度方法。

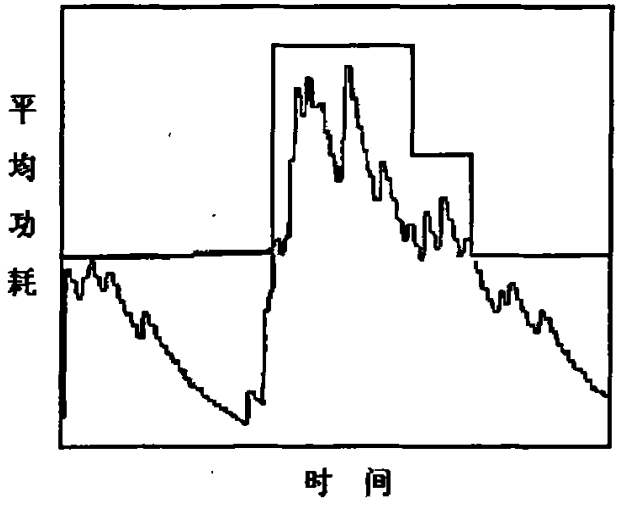


图 2.3 同步电路与异步电路在变需求系统中

图 2.3 显示了同步和异步微处理器在变需求系统中的不同表现。其中的粗线表示同步电路。细线表示异步电路。从图中可见，同步的实现方法中，对变需求系统得适应只能是通过粗略变化时钟速度，而异步电路的功耗大致与系统需求成正比。两条曲线之间的区域即是异步电路可比同步电路节省的功耗的大小。

F. 异步电路更适合可变电源电压系统

当系统的工作负载较小时，可降低电源电压来减小系统的工作速度，使得系统在提供足够性能的同时，大幅度降低电路的功耗，在同步电路中，降低电压是粗粒度的，且必须考虑降低电压后最大工作频率也将相应降低，电路是否还能正常工作在原来的时钟频率下。否则调节工作时钟频率也将是一个较复杂的过程。

而在异步电路中，情况就简单得多，无须考虑最大工作频率是否已被降低到时钟频率以下的问题。

2.2.2 异步电路避免时钟偏移

由于异步电路不用时钟机制来控制各部分的工作，模块之间通过握手协议进行通讯，它避免了一系列全局时钟引起的问题。首先，各模块之间进行通讯的握手信号是局部信号，它的负载远远小于全局时钟的负载；其次，它避免了同步电路中的“时钟偏移（Clock Skew）”问题。

2.2.3 异步电路的延迟无关性

异步电路可被设计成“SELF-TIMED”型，即由一些带冗余的表达式来表示数据[15][16]。此表达式不仅能表示数据的值，数据有效性的时序信息也包含在内。因此若采用此设计方式，无须计算连线的延迟有多大，直接判断数据是否已为有效值，即可知道是否应开始新的工作。因此，无论连线的延迟有多大，电路都能正常工作。

另外，异步电路可自然地应用于“混合时序系统”中。采用“全局异步，局部同步”的方法，可将“同步的小岛分布在异步的海洋里”，即各个时钟频率不同的同步模块用异步握手信号来连接。这方便地解决了时钟相位，频率不同的模块的连接问题。

第三章 异步电路设计方法研究

3.1 异步设计方法发展历程

A. 理论基础创建

50年代是组合逻辑和有限状态机时序电路时代。异步电路的研究开始是从分析时序电路的输入约束条件开始的，这也是当时开关理论研究领域中的一部分。Huffman首先指出，为了使一个时序电路能分辨出输入的变化，要求这个电路的输入信号之间必须有一个最小的时间间隔。也就是说，存在两个时间段d_1 和d_2，其中$d_1<d_2$。当输入信号之间间隔小于d_1 时，不能被电路分辨出来；当间隔大于d_2时，才可以分辨；当间隔大于d_1而小于d_2时，会导致时序电路功能的不确定。基于这种分析出现的一类电路形式称为Huffman电路。

同一时期，Muller提出了另外的一类电路形式，与现在的异步电路形式极为接近。实际上，他提出了使用完成信号，对于Muller 电路，仅仅在完成信号有效后，输入信号才允许发生变化[17][18]。

B. 应用探索

Huffman和Muller的先导工作，激起了开关电路领域对异步电路的广泛研究。其中，Unger的工作是最有创造性的。他给出了设计单输入变化异步时序电路的详细方法，并提出多输入变化电路设计时，需要考虑的一些因素。他的工作对随后异步电路的实用化产生了很大的影响。例如，几个早期的主流计算机MU-5和Atlas[19]是完全用异步电路实现的异步系统。60 年代，一项重要的工作是Macromodule Project，它有力地证明了异步电路模块在组合构成系统时模块化带来的优势。当时创建了一组数字模块，用这组模块不仅可以很快地组建成通用计算机，而且可以用于构建专用计算机。这个计划的研究成果给后来大量模块化的设计方法奠定了重要的基础。

另一项有价值的开创性工作来自Chuck Seitz。他提出了使用Petri网作为设计和分析异步电路的体系。其后，他在Utah大学和CalTech的教学过程，引发一大批学生对异步电路产生了浓厚的兴趣。他的影响促成了世界上首台异步Dataflow计算机和包含异步硬件的商用系统——Evans & SutherlandLDS-1（首台专用图形计算机）的出现。

C. VLSI时代异步系统的研究

70 年代后期到 80 年代，由于工艺技术的发展，数字集成电路的设计规模从LSI 向 VLSI 发展。同时，设计方法也面临着很多挑战，结束了半导体行业提供单

个逻辑单元，由电路系统设计者利用这些单元进行逻辑设计，然后构成电路系统的设计流程。这一时期，由于同步集成电路的设计模型简单，设计方法统一，CAD工具日趋成熟，绝大部分设计、研究都集中在同步集成电路上，而异步电路的研究还仅仅局限于一些特殊模块和理论研究。

D. EDA时代

随着工艺技术的持续发展，最小线宽逐渐减小，同时单芯片的尺寸却逐渐增大。特别是进入深亚微米以后，连线延迟的影响越来越大，使同步设计中的时钟skew问题越来越难处理；芯片的密度和规模的增加，功耗问题给电池供电和散热都提出了更高的要求。从80年代后期到90年代中后期，异步电路被重新引起重视。异步电路的复兴，由研究设计方法开始。设计方法的自动化为异步电路的实现提供有力的支持，出现了Locally-clocked machine、Tagram、3D-machine Micropipeline等较实用的设计模型和方法。利用这些方法设计出的异步集成电路AMULET Ⅰ、Ⅱ、Ⅲ验证了异步集成电路低功耗的优点[20]，RAPPID则是异步集成电路在高性方面的应用。

E. SOC 时代

超大规模和低功耗是21世纪的一个突出问题。超大规模要求集成电路设计必须依靠EDA工具来完成，低功耗则要求采用新的系统和电路结构。异步集成电路在这方面的优势是功耗效率高、电路组织形式灵活，但主要的问题是缺乏固定的设计流程和设计工具。SOC设计是以IP模块为基础的，这些模块有MCU、DSP、MPEG 解码器、ASIC模块等。根据任务要求的不同模块往往使用不同的时钟频率，如何有效地互连这些模块，使其能正确、有效地工作，是SOC设计中主要的问题。在异步集成电路系统中，各子模块的关系仅仅是接口互连关系，要求互相连接的接口符合异步系统要求的时序规范，而模块内部的时序关系与步长可以独立决定。异步电路SOC设计为IP互连提供了潜在的有效连接方式。

3.2 异步逻辑的设计形式

异步电路的含义非常广泛。首先，它指的是“非同步电路”，即电路中没有全局时钟来协调各部分的操作。其次，他包含了很多种的设计形式，每一种虽然都属于异步逻辑形式，但在设计上彼此差别很大。这一节简单地介绍了这些逻辑形式和它们的一些特征。

3.2.1 时序模式

每种设计形式都是建立在一定的时序模式上的。时序模式表明了设计者在设

计时所遵循的对系统内信号时序上可做的假设条件，在此条件下，设计者设计的电路时能正常工作的。

根据电路延迟，异步电路可被分为三类基本的时序模式：

A. 延迟无关电路(Delay-insensitive circuits)：

延时无关电路采用了无限连线延时模型。通过放松该电路的延时模型，人们还引入了准延时无关电路(Quasi-insensitive circuits)[21]。

在延时无关模式中，逻辑门和连线上的延迟都没有限制，因此，传输控制信号以及数据都需要使用请求确认机制。通用的电路模块，如单输出的逻辑门，由于不具有延时无关性，都不能用于延时无关电路设计。能够使用的电路模块一般只有缓冲器和 C-element 等有限的电路单元。为实现复杂的电路结构，通常使用基于别的延时假设的电路模块。而在电路模块之间的接口电路则是延时无关的。

延时无关电路可以实现平均效率的处理速度，但由于需要引入比较复杂的控制电路，增加的面积很大。同时，在实现小规模电路时，往往由于控制电路的过于复杂而变得不够经济。因此，延时无关电路更适合用于系统模块之间的接口电路。

B. 速度无关型(Speed-independent circuits)：

Muller 提出了一种速度无关电路模型[22][23]。在此模式下，假定每个门在其输出端具有一个无限惯性延时单元，但同时认为互连线的延时可以忽略。文献[22]给出了速度无关的定义，该性质可以一般地理解为电路中的任何潜在可能的信号变化除非自身发生了相应跳变，其他的电路信号的变化都不能消除该信号的变化趋势。需要指出的是，速度无关特性并不是一个电路的性质，而是电路在某种特定环境下的性质。因此，进行电路设计和验证时，都必须结合相应的电路环境来验证电路的速度无关性。

C. 有限延迟型：

在此模式下，每个门的延迟都被假设有一个可知范围内的延迟，电路只有在每个逻辑门的延迟时间处在此范围内时才能正常工作。近年提出的定时电路(Timed circuit)基于更为接近实际电路行为的有限延时模型[24]，能获得比速度独立电路和同步数字电路更好的性能。定时电路的每个门的延时都具有一个量化的上限和下限，这样就能对实际电路进行更准确的分析和优化，从而省去了由于保守的延时模型而引入的额外开销。该种电路设计的主要问题在于，加入每个门具体的延时信息之后，电路的信息将会变得非常复杂，从而使得该类异步电路的综合和验证变得更加困难。因此，如何提出一种有效描述和处理电路信息的描述方式，是此类电路需要解决的主要问题；同时，定时信息的要求对电路的生产和测试都会提出较高的要求。

从“延迟无关型”到“速度无关型”再到“有限延迟型”，对于设计者，在

设计电路时自由度是逐渐递增的。“延迟无关型”一定是“速度无关型”的，“速度无关型”一定也可工作在“有限延迟型”的工作条件下，但，反之则不可以。

但是，从“延迟无关型”到 “速度无关型”再到“有限延迟型”，对于设计者，在如何构成电路上的自由度却是逐渐递减的。在“延迟无关型”中，只要将各个功能正确的元件连接起来，则一定能正常工作；在“有限延迟型”中，设计者只能使用那些延迟时间是符合设计中的假设的逻辑门。

如何折中地选择一个异步电路的时序模型在设计中很重要。一个实用主义的设计者将会在电路的各个部分根据不同的特性和需要选择一些时序模式的组合方式。

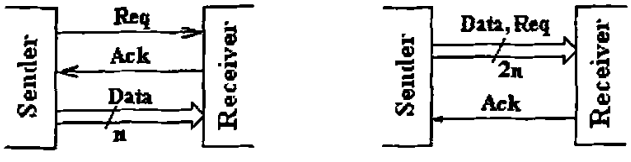
3.2.2 信号编码方式

信号编码方式指数据信息在异步逻辑中以何种方式来表达[25]。通常的一步逻辑编码方式由两种：

A. 单轨编码方式(Single-rail)

一个采用单轨编码方式的电路对信息进行编码的时候用传统的电平编码方式。信息的每一比特需要一根信号线。如果信息是数值型的，则通常的编码方式是将高电平对应为逻辑“1”，低电平定义为逻辑“0”。有的文献中，单轨也称为“bundled-data”，它是出于对数据信号和握手信号之间的时序关系而定义的，而“single-rail”则是表明一根信号线承载每一比特的信息。

图 3.1a)为单轨编码方式，通信传输通过与发送方数据绑定的 Req 信号与接受方发出的 Ack 信号变化来逐步完成。



a) Single-rail (bundled-data) channel　　b) Dual-rail channel

图 3.1 单轨与双轨编码方式

B. 双轨编码方式(Dual-rail)

一个采用双轨编码方式的电路对每一位信息 d 进行编码需要二根信号线。一根信号线 d.t 表示逻辑“1”，另一根 d.f 表示逻辑“0”。通过两信号线的组合值来表

示数据的具体值，见表 3.1。

在任何的一次通讯中，逻辑“0”线和逻辑“1”线不能同时发生变化。（因为信号不可能同时为“1”和“0”）。即在每次的通讯中，代表每一位信息的两根信号线中的一根将发生一次变化。因此，可以通过检测数据线中每一位信息的两根信号线中

表 3.1 双轨数据编码

	d.t	d.f
Empty(“E”)	θ	0
Valid “0”	0	1
Valid “1”	1	0
Not used	1	1

所发生的变化，来判断数据信息是否已准备完毕，是否处在有效状态。可见，双轨编码方式的时序信息是隐藏在数据信号中的，两根信号线的数值能指示数据信息的有效性。通信过程中发送方不需要再另外绑定一个 Req 信号来表示数据的有效性。这使得双轨编码方式具有很好的延时无关性(Delay-insensitivity)。

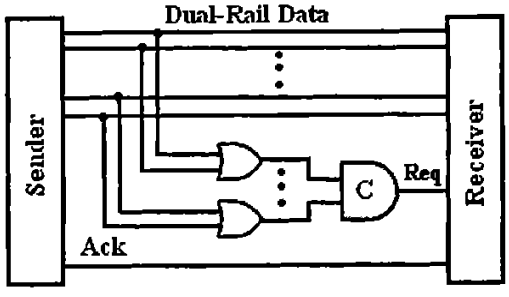


图 3.2 Dual-rail Logic

图 3.2 是一个双轨逻辑的例子，每组双轨信号都会经过一个或门，然后所有的信号输入 C-element，闲置状态时，或门的输出都为 0，则 c-element 的输出为 0；进入工作状态后，各双轨信号由(0,0)变为(0,1)或(1,0)，由于各信号到达稳定的时间不一样，当所有信号都达到稳定状态后，所有的或门的输出都为 1，则 C-element 的输出也变为 1，由此触发接受端，表明输入数据有效，可以开始接受。其他的实际应用中，电路的结构可以根据需要进行相应的调整。

3.2.3 输出指示方式

输出指示方式指用何种方式来表示此模块的工作与运算已结束且模块输出值已为有效值。异步逻辑的输出指示模式可分为两种：

A. Self-Timed 逻辑

采用 Self-Timed 逻辑[26]的异步电路使用带冗余的逻辑表达式，其中的一些输出值的组合表示此输出数值是有效值，另一些输出值的组合表示此时输出无效，数据还未被准备完毕。因此在"Self-Timed"逻辑中，通过判断输出值，即可知道此时输出数据是否有效，即输出有效与否的判据是隐藏在数据中的。Self-Timed 逻辑通常采用双轨编码的方式，则通过一位信息的两根信号线的数值，可判断此时信息是否有效。

B. Bundled-Data 逻辑

Bundled-Data[27]逻辑采用传统的逻辑表达式，但是另包括一根附加的控制线，由此线来指示输出有效的时刻。通常，控制线上使用延迟匹配，即控制线上信号的变化要通过的延时不小于模块中组合逻辑功能块的延迟。Bundled-Data 逻辑通常采用单轨编码的方式。

利用 Self-Timed 逻辑，可设计"延迟无关型"的异步电路，它对任何逻辑门和线上的延迟都不敏感，电路的有效输出始终是正确的（当然速度较慢的门和线会引起有效输出的延迟）。由于功能性与性能的完全分离，此方法设计的电路的功能正确与否与实现的工艺是无关的，并很容易实现电路功能的正确性的证明。因此，大多数的完美主义者偏好用此方法。然而，此方法采用的冗余结构耗费太大，它所花费的芯片面积几乎是标准单轨编码逻辑的两倍。而 Bundled-Data 逻辑中所采用的延迟匹配方法不能设计出"延迟不敏感型"的电路，它所设计的电路是与工艺有关的，并且在设计延迟匹配时需要用到如详细的 SPICE 模型等的仔细的工程。但是它的优点是与相对应的同步电路比起来面积增加的不多。因此在现阶段，Bundled-Data 受到大多是设计复杂异步电路的设计师的偏爱。但随着半导体制造工艺的发展，深亚微米工艺会使电路中连线上的延迟更大，片内的延迟更难控制，故"延迟不敏感型"的冗余编码方法将会在未来有更大的应用。

3.2.4 通讯协议

异步逻辑的信号协议[25]指异步通讯时信号所遵循的协议及协议所表示的含义，包括两种：

A. 四相通讯协议

在四相通讯协议中，其中的两相是处在通讯的活跃状态。另外两相中，握手信号及传输的数据回复到预定义的状态。图 3.3a)是四相通讯的一个例子。

四相通讯的过程包括：1)由传送方（Sender）准备好数据并将信号 Request 置高；2)接受方（Receiver）做出响应接受数据，然后将信号 Acknowledge 也置为高电平；3)传送方观察到信号的变化，表示通讯已经成功，将 Request 变为低，

表示传送方已经不再处在活跃状态；4)接受方也将信号 Acknowledge 重新变低，完成一次通讯。

B. 两相通讯协议

在两相通讯中，信息是通过信号电平的转换或改变来传输的。图 3.3b)是两相通讯协议的一个

两相通讯：顾名思义，在一次通讯中包含着两相：1)传送方准备数据并发送，此相由Request信号变化一次来表示结束;2)接受方接受数据，此相由Acknowledge信号变化一次来结束。信号的电平不包含任何信息，只有电平的转换才有意义，电平转换一次表明此逻辑线上的值为"1"。同时，上升沿和下降沿具有相同意义，没有什么区别。

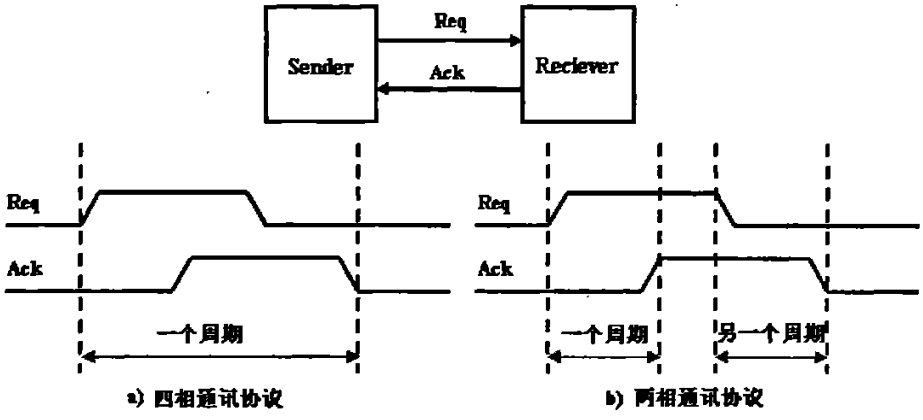


图 3.3 四相与两相通讯协议

四相协议中，标志通讯过程的是电平值。每次通讯过程中 Req 与 Ack 握手信号都要"归零"，这个过程有着不必要的时间和能耗浪费。而两相协议则避免了这种情况，它关注的是电平的跳变，而且"0→1"与"1→0"的电平变化是没有区别的，都可以表示一个信号事件。所以理论上讲，两相通讯协议要比四相协议更能实现较高的速度，更具有低功耗的特性[28]。

3.2.5 异步电路握手协议

异步电路中，模块间的通讯是通过握手协议来完成的。而握手协议可以通过两种编码方式与两种通讯协议的组合构成。很明显这种组合方式可以有四种：四相单轨协议、两相单轨协议、四相双轨协议、两相双轨协议。

四相单轨协议、两相单轨协议由于其工作方式比较简单，比较接近于同步电

路中的通讯协议，对于设计者来说比较熟悉，是比较容易理解的。图 3.4 给出了四相及两相单轨握手协议的图解。

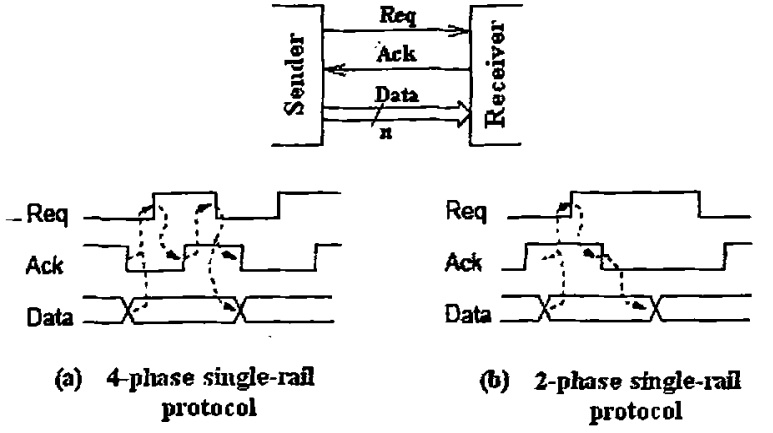


图 3.4 四相及两相单轨握手协议

对于四相双轨协议与两相双轨协议，图 3.5 是一个两信道的简单例子。

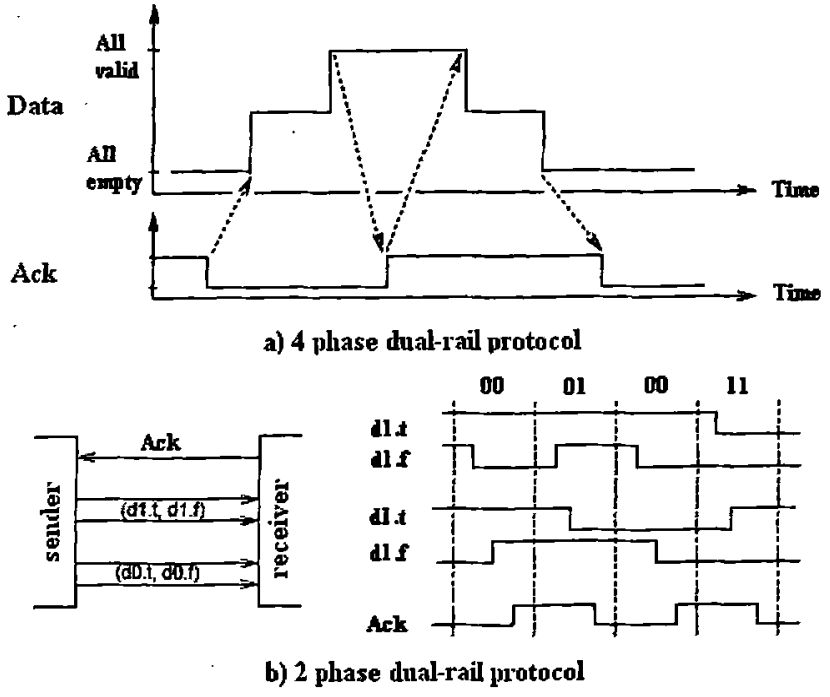


图 3.5 四相及两相双轨握手协议

图 3.5a)四相双轨协议中，“All valid”表示所有信道中数据都已经处于有效状态，“All empty”表示所有信道中数据都无效。二者状态之间存在一些部分有效的中间状态。只有所有的信道都有效并完成数据传输后，Ack 才会变高，同样地，

只有所有信道都无效后，Ack 信号才会归零，完成一次数据传输。两个有效数据不能连续出现，必须以一个无效数据间隔，在这方面存在速度上的损失，不过具有双轨传输延时无关性的特点[25]。

两相双轨握手协议如图 3.5b)。每个信道数据编码由逻辑线的一次跳变（“0→1”与“1→0” 具有相同意义）来表示“1”，然后每条信道的数据通过对两条逻辑线 d.t 与 d.f 的双轨编码来完成。数据传输中没有无效数据，两个有效数据可以连续传输。因此相对四相协议，两相协议速度更快一些。

图中出于简化考虑，以两条信道为例，多条信道的情况可以类推得出。

另外需要解释的是，我们上面介绍的握手通讯协议都是设定发送方(sender)为主动方，即发送方发起了信道上的数据传输，称这种方式为“push channel”。相反地也存在另一种情况，即接受方主动要求发送方传输数据，相应地我们称这种方式为“pull channel”[25]。这两种情况中工作方式在根本上是一致的，不同的只是 Req 与 Ack 信号的方向正好是相反的。数据的有效性通过发送方到接受方的 Ack 信号来标示。

3.3 异步电路的计算机辅助设计软件

在计算机辅助设计方面，同步电路已经发展得相当成熟和完善，所以当前的电路设计软件大都基于同步电路。而由于异步电路在理论上还没有完全解决，所以当前异步电路的设计软件尽管有数十种，但是基本上还处于实验室阶段，还需要不断地完善。虽然如此，当前的异步电路的计算机辅助设计还是有很多成功的案例。

自上世纪 50 年代以来，异步电路的计算机辅助设计的研究工作就在不断进行，其中主要的异步电路计算机辅助的成功设计有：

在上世纪五六十年代，异步设计用于很多早期的大型计算机，其中有在伊利诺伊大学，Muller 及其同事使用速度无关（Speed-independent）设计方法设计的 ILLIAC 和 ILLIAC II。在 1989 年，加利福尼亚理工学院的研究者率先设计处理完整的异步微处理器，该设计完全是准延迟不敏感（Quasi-Delay Insensitive，简称 QDI），即除了特殊的等时分支，不用考虑延迟电路就能运行正确。在 1994 年，曼彻斯特大学 AMULET 小组完成了 AMULET1，这是第一个与现有的同步处理器（ARM 微处理器）编码兼容的异步处理器。同年，飞利浦为数字便携磁带机制作了一个错误校正芯片，并在 1998 年制作出了异步 80C51 微控制器，这两个设计均在功耗上大幅度优于同等功能的同步电路，但在芯片面积上却增加不多。在 1995-1999 年，英特尔完成了 RAPPID 工程，这是一个对奔腾 II 32 位 MMX 指

令集的完全异步的指令长度解码器，和同样功能的同步设计相比，速度提高了 3 倍，而功耗仅为 1/2。

表 3.2 异步电路设计软件分类

实现功能	软件名称
综合设计平台	ACK、Balsa、Tangram、TAST、HORN、Persia、Rainbow
仿真	LARD、VHDL++、Testify、Fsimac
相关设计工具	Butler、NCL、XDL、HOP、Synchronized Transitions、AVEmap、USC-PET
逻辑综合	3D、ATACS、MINIMALIST、Petrify、SIS、CASCADE、ConfRes、DESI、Optimist、AC-TSPC、ASSASSIN、DGC、FORCAGE、Ludwig、MATE、MVSIS、SYN
前端设计软件	digpn、Pipefitter、VSTGL、digg、Punf、VL2STG
形式验证	CADP、Clp、CCS(CWB)、Firemaps、Transyst、Veraci、CBMC、MAGIC、SyMP、BMC、SMV、BDD、CSML、MCB、Word-level SMV、Verus、AVER、SPHINX、TIMVER、VERDECT、Versify、Circal System

当前异步电路的相关软件有数十种，它们分别完成异步电路设计不同层次的功能[29]。表3.2给出了它们主要功能的基本分类[30]。按照上述分类我们对异步电路的设计软件简单介绍，对现在应用比较广泛的几种软件会给予较详细的说明。

A. 综合设计平台

这类软件中，Balsa使用较为广泛，而且可以免费下载。Balsa软件是一个异步电路的综合系统，其使用的语言也称为Balsa，编译方式采用语法直接编译（syntax-directed compilation）。经过Balsa编译的代码可以生成由很多模块构成的网络图，网络的元件都是使用握手协议通信。元件之间的连线定义为信道（channel），信道的参数有控制（control）和数据（data），其中数据参数不是必须的。信道的两端分别为主动端口（active port）和被动端口(passive port)，主动端口为起始端，发送请求（req），而被动端口响应（ack）。图3.6为Balsa的设计流程[31]。

信道的分类：牵引信道（pull channel）的信号流向是从主动端口到被动端口；推动信道（push channel）的信号流向是从被动端口到主动端口。

电路的Balsa描述使用Balsa-c编译为一个中间breeze描述。大多数Balsa工具和breeze握手中间文件相关。Balsa描述的末端工具实现中可使用breeze文件，但是breeze文件也包括过程和类型定义，源文件允许使用breeze作为Balsa的包描述格

式。

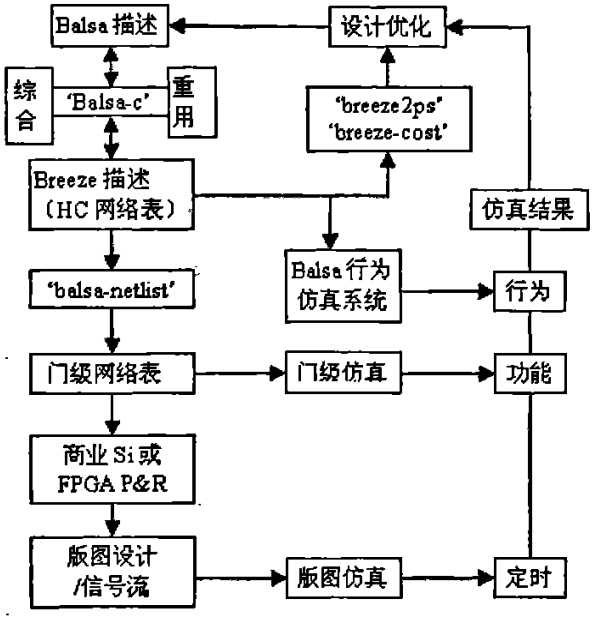


图 3.6 Balsa 的设计流程

ACK是一个高级的综合工具，在程序级描述系统（包括数据通路描述），可自动地编译控制和数据通路电路的相互连接规范。使用标准的Verilog仿真工具（Veriwell、Verilog-XL等），数据通路综合使用Synopsys，复杂的门级综合使用Cadence，末端设计现在也使用Cadence。这个软件还在完善中，尚未提供下载。

Tangram是Philips公司开发的综合工具，其使用握手电路作为其核心表达形式和交互结构，结构上和Balsa近似。Philips公司用其开发出了异步80C51微控制器、DDC芯片和智能卡等一系列IC，是一种比较成功的商业化软件。

B. 仿真软件

LARD是一种异步电路设计的硬件描述语言，也可以描述同步电路。相比VHDL，LARD的设计更为异步情况考虑。首先LARD比VHDL多了信道通讯模块，这一部分对握手电路的描述相当重要，可以更为清楚地描述其内部关系；其次，尽管VHDL可以有很多并行的进程，但是他们之间必须建立关系，如使用一个同步时钟，这样就成为了同步电路，所以使用VHDL描述异步电路相当麻烦；再次，VHDL语言复杂，不容易建立异步仿真。Tangram解决了上述问题，但其缺乏一些高级编程能力，而且其属于Philips公司，没有开源。

异步设计中，著名的AMULETⅢ就是使用LARD语言进行建模的。使用LARD仿真的步骤如下：仿真抽象模型，以决定基本结构；使用商业CAD软件或Balsa建模。

VHDL++是一个基于VHDL的异步设计语言，增加了信道之间的同步信息传输和进程中信息传输声明的并行执行。

C. 相关设计工具

NCL和传统的布尔逻辑不同，布尔逻辑必须加入一个时钟以决定输入和输出。无时钟的NCL逻辑则是数据驱动的，指令在使能时即执行，数据在有效后就输出。NCL使用离散域值门，这种门仅可以识别特定的值的组合。

NCL用于延迟不敏感电路，优点是使用VHDL语言且结合商业软件比较好，缺陷是使用NCL逻辑会多使用3-5倍的面积，且功耗将会增加。设计采用以下步骤[32]：写出RTL级的VHDL语言描述，并加入延迟定义：使用VHDL或Verilog写出综合前的测试平台；仿真综合前的网络表，可以使用ModelTech；使用Synopsys公司的dc_shell综合网络表,使用Theseus Logic, Inc的工具ncl_shell进行设计优化；为综合后的网络表修改综合前的测试平台，并进行测试；使用orphan_check命令验证设计是否延迟不敏感；使用Silicon Ensemble进行布线。

HOP是一种硬件描述语言，用于VLSI设计，现在已较少见。XDL（eXtended model for Delay-Insensitive systems）是一个针对延迟无关电路的描述和实现的理论框架。AVEmap是一个用于异步BM和one-hot domino电路的版图生成工具，其接受未绘图的电路描述，使用库中的元件产生平均性能最优的版图电路。

D. 逻辑综合

这类软件主要包括综合功能，有的也附有实现网络表和仿真等其他功能。

3D软件使用XBM模型进行综合，输入要求是NIG，综合的目标是伪静态非对称CMOS复杂门（pseudo-state asymmetric CMOS complex gate）,所以基本电路模型采用Huffman电路模型。 其综合结果可以为逻辑门的结构（由Kudva et al设计），也可以为广义C单元的结构。使用广义C单元结构更有利于电路的功能，可以比逻辑门结构节约40%的面积且提高30%的速度。

ATACS是一个对赋时电路综合、分析和验证的工具[33]，其接受VHDL语言、握手描述、Petri网、BM有限状态机、TEL结构和状态图输入。它支持几种综合算法，包括有效显式状态方法、BDD隐式方法和一个避免状态空间探测和从自由选择Petri网表达的电路简单生成的直接综合方法。ATACS也支持分析和验证。利用给定延迟分布的随机分析，得到系统的平均情形性能。对应验证，设计者可提供在时间控制状态空间中选定的时间约束。如果电路已提供，也要校验是否有竞争冒险现象。最后，在状态空间探测网络安全和检查死锁，并提供图型错误路径。

MINIMALIST是一个针对BM异步有限状态机的可扩展的综合和验证模型，采用启发式的异步综合算法，如最优状态分配、二值无冒险逻辑最小化和可测试性综合，即Huffman电路的综合算法。 该软件相对于以前的软件3D和UCLOCK较早

期的版本，在功能上和设计效果上均有较大提高。

Petrify的综合对象是Petri网和异步控制器，其读入一个Petri网，再将其转换为一个较为简单的类似Petri网的描述。Petrify读入的Petri网也可以解释为使用STG描述的异步控制器，其可以解决完全编码问题，而生成速度无关电路，故其使用了Muller电路模型；Petrify也可以综合成赋时电路。如果要综合成赋时电路，设计者要提供相关的时间约束。Petrify可以综合成不同的网络表，如布尔等式、广义C单元或给定库中的门，而且会检测出电路中的置位和复位点及其初始化信息。但是，Petri不能进行数据通路的综合，而且不是总能得到电路的网络表，有一定局限性。

SIS包括多种综合算法，延迟模型包括无界延迟二级逻辑门、有界线延迟和门延迟模型，接受STG描述，输出可以转换为Verilog和VHDL的门级网络表。设计流程为：检查描述性质→状态分配→综合成为初始逻辑网络表→逻辑网络表的约束优化→映射的指定的标准单元库→插入延迟环节使电路无冒险。该软件的模块化比较好，容易加入新的算法，输出接口标准化。

E. 前端设计软件

Di2pn可以自动地将设计描述转化为Petri网，故可用于使用Petri综合的软件的前端设计。其可以输出文字表达的Petri网，这样可以用于Petrify;也可以输出ASTG形式,这样可用于SIS.类似的软件还有pn2dot和pn2lola，用于转换为发Petri网的其他形式。

Digg类似上面的di2pn，它是将DI-Algebra转换为状态图，用于状态图综合的软件。

Pipefitter用于微流水线（micro-pipelined）异步电路的综合，使用有匹配延迟的同步数据路径和支持并行、串行和选择的4相控制单元。其使用一部分Verilog语言（不支持完整的Verilog语言）作为输入，得到综合脚本和一些Verilog网络表：一个控制单元和多个数据路径上的寄存器。同时得到一个描述功能的STG，可以用于使用STG综合的软件，如Petrify。所以该软件既可以用于综合，也可以用于前端设计。

VSTGL是一个可视化的STG的生成、编辑和仿真软件，可以用于Petrify的前端设计，结合Petrify可以得到设计的基本流程：使用VSTGL手工生成STG→写入后缀为.g的文件，用于记录STG的信息→生成可视化的STG图→检查生成的STG图→校正后缀为.g的文件→使用Petrify分析→修正可能的错误。

F. 形式验证

CADP可以进行编译、交互仿真、形式验证和对使用ISO语言LOTOS编写的代码进行描述测试。验证的过程一般首先将系统从非形式化的描述转换为形式化的描述，这样可以检查出不相容性和未覆盖的问题；然后使用CADP进行仿真、实现

和验证形式描述。

CCS用于描述2相和4相异步系统中的控制信号，基于门级或者更高级的描述。支持CCS的平台为CWB（Concurrency Work Bench）,在该平台上进行语法测试，可以得到最简等价状态机，然后进行功能测试。曾用于描述和验证AMULET微处理器。

CV是一个对VHDL描述的验证软件，采用符合模型验证。分为cva（VHDL编译器）和cvc（模型验证）可以进行完整的模型验证，流程图如图3.7[34]。

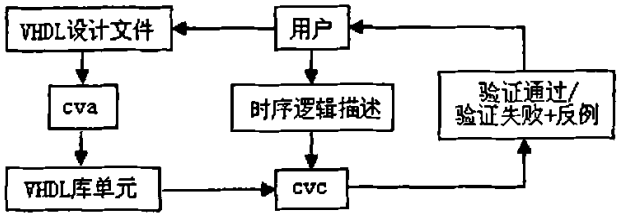


图 3.7 CV 的验证流程

3.4 异步微处理器的设计现状

表 3.3 是近几年中设计的异步微处理器的基本情况。可以看出，近几年中，对异步微处理器的研究很多，且各个国家都有，这说明用异步逻辑来实现微处理器的优越性已被大家普遍认可，现正处在设计的实践和研究阶段。同时，表中所实现的微处理器的指令集大多数是自定义的，这对用户来说使用起来不方便，不能普及，因此大多数的异步微处理器还未能代替市场上的现有微处理器[35-38]。

表 3.3 异步微处理器设计的现状

型号	设计者	设计形式	指令集
CAP	CALTECH,USA	四相/双轨协议 延迟不敏感型	自定义 16 位， RISC-like
AMULET1	University of Manchester, UK	两相，Bundled-Data	ARM
AMULET1		四相，Bundled-Data	ARM
NSR	University of Utah, USA	两相，Bundled-Data	自定义 16 位， RISC-like
Fred		两相，Bundled-Data	基于 88100
CFPP	Ivan Sutherland, USA	两相，Bundled-Data	SPARC
Hades	University of Herlfordhire	Unspecified	自定义

ECSTAC	University of Adelaide, Australia	Fundamental Mode	自定义变字长指令
TITAC	Tokyo Institute of Tech, Japan	两相，双轨， Quasi 延迟不敏感型	自定义 8 位
ST-RISC	Israel Institute of Tech, Israel	双轨，延迟不敏感型	自定义
FAM		四相，双轨，延迟不敏感型	自定义 RISC-like
STRAP	Stanford University, USA	不同的同步时钟，混合时序系统	MIPS-X
SCALP	University of Manchester, UK	四相，Bundled-Data	自定义

表中也给出了各种异步微处理器的设计形式。从中可看出，异步微处理器的设计大致可分为两类：

A. 异步设计采用保守且谨慎的设计形式，适合进行正规的综合和验证，但它所实现的电路结构较简单，如 CAP，TITA，CST-RISC；

B. 异步设计采用更大胆的设计形式，或非正规的设计方法，但是它所实现的电路的结构较复杂，如 AMULET，NSR，Fred，CFPP，Hades，ECSTAC 和 STRIP。

3.5 异步电路所面临的挑战

虽然异步集成电路具有很多的优点，但它在超大规模集成电路应用方面，还面临着许多挑战。发挥异步集成电路的优点需要合适的应用系统。异步集成电路应用中，最突出的两个优点是低功耗和潜在的高性能，但这优点不是无条件的。异步集成电路没有整体时钟，使用本地握手信号进行时序控制，需要增加一些电路模块来完成这些工作。这些附加的电路模块往往会对功耗和性能产生负面影响。因此,发挥异步集成电路低功耗和高性能的特点需要合适的应用对象，比如，在待机很频繁的场合容易实现低功耗；在平均性能与最差性能相差较大的场合，有利于实现高性能。异步集成电路设计没有统一的设计方法和特别成熟的 EDA 工具。现有的异步集成电路的设计方法，大部分是针对集成电路中某些局部问题的解决方法。

超大规模集成电路设计的一个重要特征是 EDA 工具的广泛应用，同步集成电路的 EDA 工具已经相当成熟，而异步集成电路设计的工具仅仅局限于某些部分的算法研究和自动化[39]。

因此，在集成电路发展到超大规模和SOC时代后，主要的问题是解决面向超大规模集成电路的异步集成电路设计方法，以及选择合适的领域发挥它的优点。

第四章 异步电路基本单元

由于异步电路相对同步电路具有的独特优势，目前已有一些异步处理器如AMULE、MiniMIPS、微控制器异步80C51、异步PIC微控制器（复旦大学）等出现，Philips公司的异步80C51已经进入实际应用[40]。

但由于异步集成电路现在还没有统一的设计方法和成熟系统的EDA工具，现有的异步集成电路的设计方法，大部分是时针对集成电路中某些局部问题的解决方案。而且已经实现的异步微处理器都不同程度地采用了全定制设计流程，自动化程度低，开发时间长。而当前流行的同步集成电路设计一般采用基于标准单元库的半定制设计流程，具有方法简单、工具成熟、自动化程度高等优点。

如何用基于标准单元库的半定制流程实现异步集成电路设计，是异步集成电路设计中的难题之一。首先需要解决的问题就是建立异步标准单元库，供前端综合工具和后端布局布线工具的使用。现有的标准单元库一般只针对同步集成电路设计，采用静态CMOS工艺，缺乏对异步集成电路设计中特殊电路结构（如动态电路、C单元）的支持。因此对异步电路常用单元电路的研究就显得非常重要。

异步集成电路使用的标准单元，既包括同步集电路标准单元库包含的逻辑门、寄存器、多路器等逻辑单元，还包含一些异步逻辑需要的C单元、S单元等异步逻辑单元。这些单元可以分为：输入输出（组合逻辑）类、LATCH类、数据通路类和握手电路类。

输入输出类单元中的逻辑门主要包括传统的反相器、与门、或门、异或门等。LATCH类单元的特点是具有存储的功能，从晶体管电路结构看，存在一些反馈环路，异步集成电路中使用的LATCH类单元，根据功能和应用位置可分为两类：用于接口的单元和用于系统基本模块内部的单元。数据通路类单元主要完成数据运算和数据流控制，也分为异步数据通路和异步多路器两类。握手电路类单元主要完成握手信号的控制，按其结构应该属于宏单元，因为握手电路类单元是基于逻辑门而不是晶体管的。

4.1 异步电路 C 单元

C 单元（C-Element）是异步集成电路中一个普遍应用的特殊单元。C 单元的接口类似于普通的逻辑门，但是其主要特点是本身并不是一个完全的组合逻辑，其中包含一个类似寄存器功能的存储结点。由于其存储结点仅保存自身的工作状态，一般把 C 单元归于逻辑门类。

C 单元是异步控制电路中最常用的基本单元之一，它由 Muller 提出[41]，因此被称作“Muller C 单元”。C 单元有两个输入 a 与 b，一个输出 c。它的逻辑行为可以如下描述：若两个输入都为 0(1)，那么输出也变为 0(1)，否则输出保持。为保证 C 单元的正常运行还设定：一旦两个输入都变为 0(1)，那么在输出做出响应之前输入将不可以再发生变化。输出 c 也可以通过两个输入 a、b 与输出的前态 c' 的布尔函数表达式得到：

$$c = c' \cdot (a+b) + a \cdot b \tag{4-1}$$

图 4.1 给出了 C 单元的状态转换图及示意图。

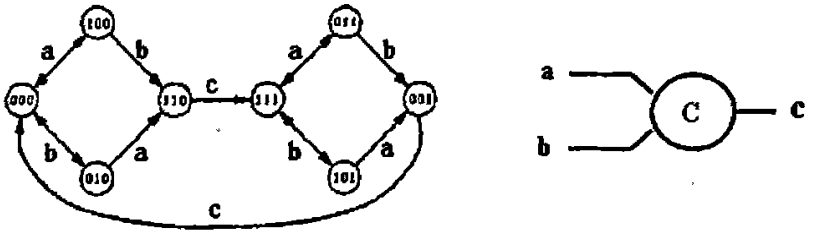


图 4.1 C 单元的状态转换图与示意图

C 单元可以通过单轨和双轨电路结构实现。下文中介绍了四种单轨(single-rail)及四种双轨(dual-rail)电路实现结构。双轨结构使用了“Differential logic and an Inverter Latch”结构，文中简称“DIL”[42]。每种单轨或双轨电路结构各有其特点，各有其适合的应用环境。

4.1.1 C 单元单轨实现

这里我们主要设计实现了四种经典的 C 单元单轨电路结构。

4.1.1.1 C 单元四种单轨结构

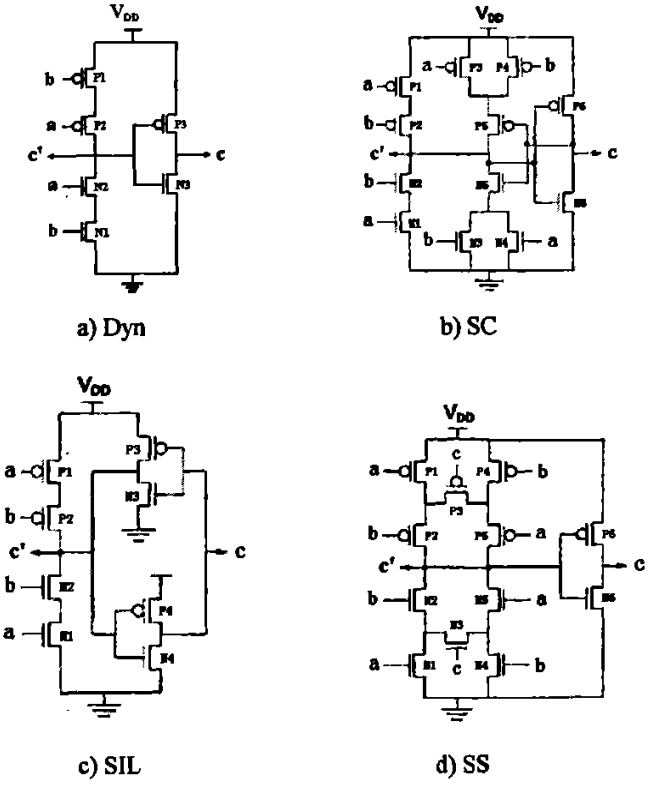
图 4.2a)给出的电路结构是单轨基本动态(dyn)C 单元，它组成了 C 单元的静态实现形式的基本功能部分，本身没有对输出 c 进行锁存的电路结构。而 C 单元各种静态实现电路正是在输出的锁存结构上有所不同。

C 单元单轨静态的电路结构中有两种不同的晶体管：完成输出转换功能的称作 switcher，而提供必要反馈以保持输出状态的晶体管称为 keeper。Switcher 的尺寸大小与动态的 C 单元相似，而 keeper 的大小一般采用最小尺寸，这样可以尽量的减小它们对电路的附带影响[43]。

图 4.2b)SC 是 C 单元的一个传统静态实现，此电路是由 Sutherland 提出的。电路中 Keepers 包括 N3、N4、N5、P3、P4 与 P5，它们都采用最小尺寸。图 4.2c)所示 SIL 结构由 Martin 提出，此结构采用一个弱反馈反相器来保持输出状态。由

Van Berkel 提出的 C 单元结构如图 4.2d)，Keeper 包括 N3 与 P3 管，但同时主体结构晶体管也参与了输出状态的保持。此结构对两个输入是对称的，而且它的反馈结构保证 C′ 点任何时候都能通过上拉下拉网络连接到电源或地，使得对称 C 单元具有更好的抗噪能力。

在 SC、SIL、与 MSIL 结构中，主体晶体管 NMOS 的宽为 Wn=W，PMOS 的宽为 Wp=rW。而 SS 实现中，Wn=W/2，Wp=rW/2，这样 SS 就与其他结构有相同的上拉和下拉电阻。SC 与 SS 电路是无比的[44]，对管子的大小没有太严格的限制。但 SIL 并非是无比的，在节点 C′ 处存在竞争现象。



a) Dyn, dynamic; b) SC, conventional pull-up pull-down; c) SIL, with an Inverter latch; d) SS, symmetric

a) 动态 C 单元； b) 传统上拉下拉结构 C 单元； c) 反相锁存 C 单元； d) 对称 C 单元

图 4.2 C 单元单轨动态与静态结构

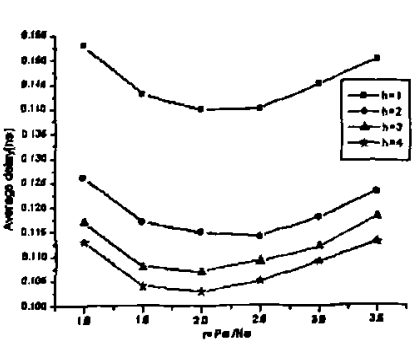
4.1.1.2 C 单元单轨电路设计实现

异步电路 C 单元的设计实现流程中，我们首先用 Cadence 电路设计工具 Composer 完成电路设计，并调用 Hspice 对电路进行模拟；然后进行版图设计，并运用 Dracula 对版图进行 DRC 和 LVS 检查；最后用参数提取工具提取版图中

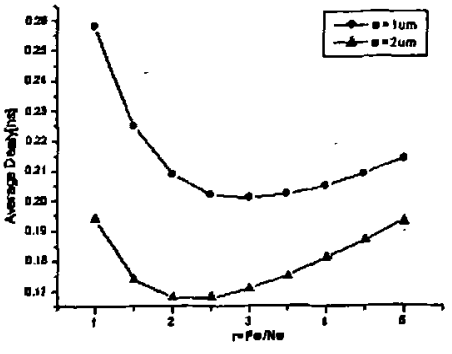
的各种参数，并进行更详细的 Hspice 模拟，以得到更精确的性能参数。

电路模拟过程中使用 CSMC 0.35μm 工艺库文件，因此设定各种电路中所有管子的长度为 0.35μm。对 Dyn 结构，我们设定上拉 P1、P2 管与下拉 N1、N2 管的大小尺寸之比为 r，上拉下拉管子与输出反相器 N3、P3 管的大小之比为 k。图 4.3a)给出了仿真结果，可以看出，电路的平均传输延迟随管子增大逐渐减小，这里我们选定 k=4、r=2，此时平均延时为 0.103ns。

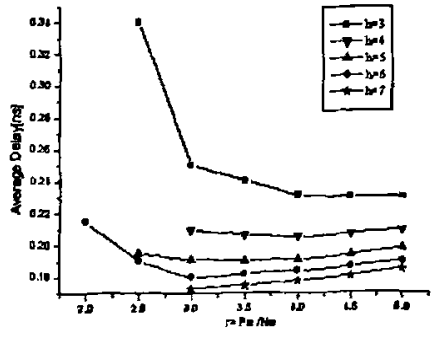
静态结构中我们设定 keeper 的宽度都为模型库中限制的最小尺寸 0.4μm，同样设定上拉 P 管与下拉 N 管的大小之比为 r。对 SC 电路，锁存结构晶体管 N3、N4、N5、P3、P4 与 P5 都采用最小尺寸。设定下拉 N 管的宽为 w，图 4.3b)是 w 值分别为 1μm 与 2μm 情况下电路的仿真结果。图中可以看出 w=2μm 时电路的延时性能相对要好一些，可以实现 0.165ns 的最小平均延时值，我们选取 w=2μm、r=2.5 作为 SC 电路的实际参数。



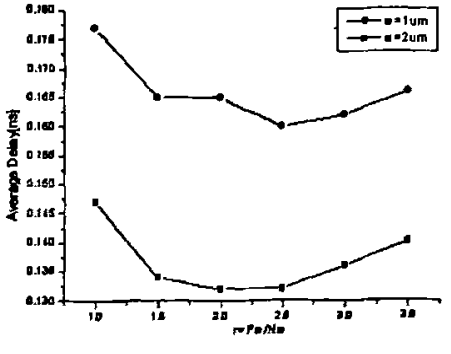
(a) Dyn



(b) SC



(c) SIL



(d) SS

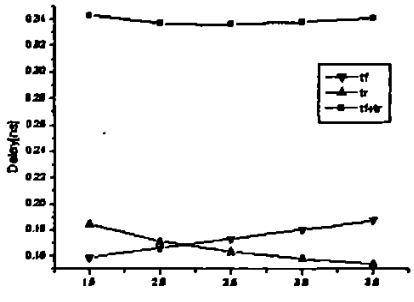
图 4.3 不同参数对电路平均延时性能的影响

SIL 电路中我们设定 N3 管的宽为 0.4μm，N1、N2、N4 管的宽为 N3 管的 h 倍，即 0.4hμm，那么 P1、P2、P4 管的宽为 0.4rhμm。从图 4.3c)中可以看出，r 与 h 的值越小，传输延迟越大，更甚导致电路功能失常，选择较大的 r 与 h 值有

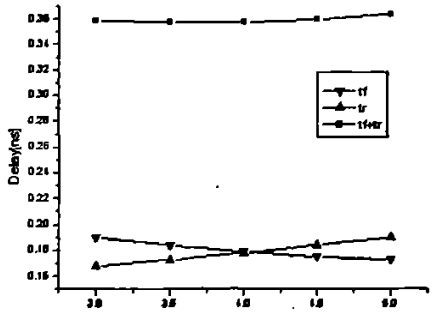
益于电路的性能。综合考虑选取 r=3，h=6 为 SIL 的最终参数，此时平均传输延时为 0.176ns。

对 SS 电路，管子大小只影响电路的性能，不会影响到电路的功能。这里同样设定下拉 N 管的宽为 w。管子的大小为其他几种电路的一半,但对输出反相器 N6 与 P6，为保证电路的负载能力，它们采用与其它电路相同的尺寸。图 4.3d)给出了不同参数下电路的仿真结果。很明显，w=2μm、r=2 时，平均延时为 0.132ns，电路的延时性能为最好。

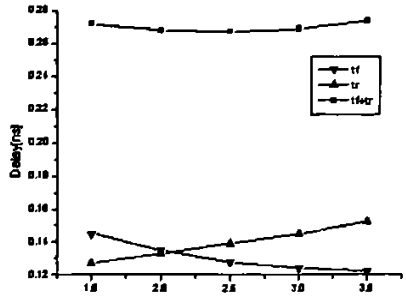
对三种静态电路进行详细模拟，选定合适的电路参数后，我们发现电路的上升和下降延时有一定的差异。为达到二者之间的平衡，可以在固定电路其它管子大小的情况下，对输出反相器的 N 管大小进行调节。图 4.4 即输出反相器 N 管对上升下降延时影响的仿真结果。可以看出 SC 电路在 N6 管宽为 2.4μm 时达到平衡，此时延时为 0.17ns。SIL 在输出反相 N4 管宽为 4μm 时，上下延时平衡在 0.178ns 处。而 SS 电路可以实现 0.135ns 的上下延时平衡值，为静态电路所能实现的最小值，此时 N6 管宽为 2.2μm。



a) SC



b) SIL



c)SS

图 4.4 输出反相器 N 管对上下延时的影响

表 4.1 对三种静态 C 单元电路的参数进行了比较。SIL 电路在面积上有着较大的优势，只相当于 SC 与 SS 的 61%和 70%，并且它的输入电容只为 0.001pf。但由于弱反馈回路的存在，它的功耗比较大。SC 与 SS 电路在功耗和延时上都要

相对小一些，但同时二者都存在比较大的输入电容。SS 单元可以实现 0.132ns 到 0.165ns 的平均延时，此电路具有很好的综合性能。

通过对各种结构仿真结果和电路参数的详尽对比，可以看出：反相锁存 C 单元在面积上有明显的优势，传统结构 C 单元与对称 C 单元则具有相对较小的功耗和平均延时。对称 C 单元本身具有对称结构，并且引入的锁存结构也最为简单，使得它能实现最小的平均延时和功耗。所以在微流水线实现中，若主要考虑面积因素，可以选用反相锁存 C 单元。若更关心电路性能和功耗，则对称 C 单元更为合适。

表 4.1 三种静态 C 标准单元比较

	平均延时(ns)	面积(μm^2)	输入电容(pf)	功耗(pW/Hz)
SC	0.165	162.06	0.003	1.87
SIL	0.176	99.46	0.001	2.24
SS	0.132	142.13	0.003	1.62

4.1.2 C 单元双轨实现

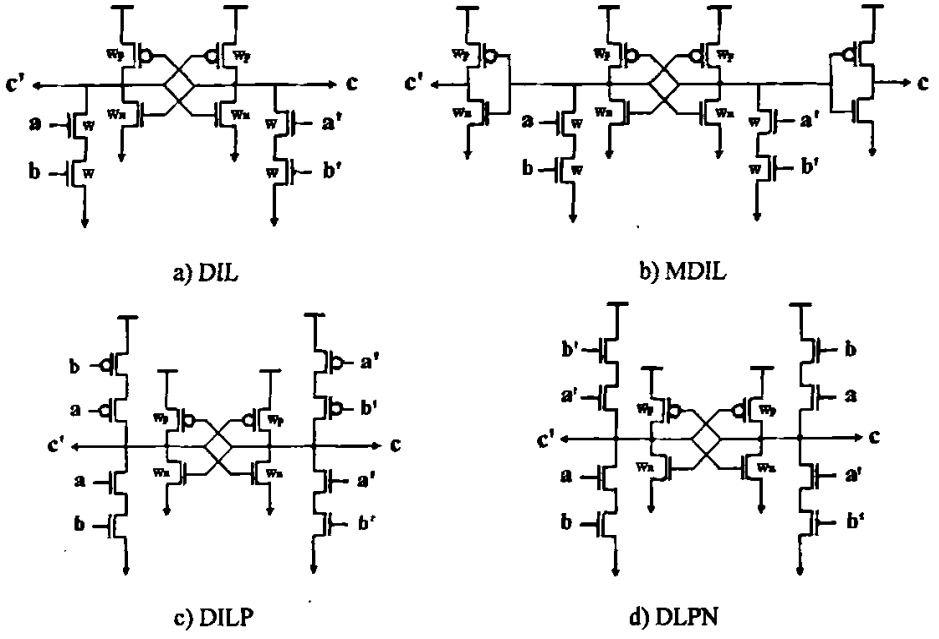


图4.5 C单元双轨电路结构

图 4.5a)所示 DIL 是 C 单元双轨实现电路结构。它由两组 NMOS 下拉管和一

个反相锁组成。反相锁中的 NMOS 管在转换过程中是惰性的，因此将它设定为最小尺寸。而 PMOS 管在转换过程中则是工作的，它们的大小关系到电路的延时特性。另外为保证电路的功能正确实现，下拉管的阻抗要小于锁存 PMOS 器件。

由于转换过程中输出有竞争现象，DIL 结构存在比较大的延时和能耗敏感性。减弱这个敏感性的一个途径是在 DIL 电路输出与负载之间加入一对反相器，如图 4.5b)MDIL。此电路的运作与 DIL 电路是相似的，只是在上升延时上有所改善。另外一个改进 DIL 上升延时特性的方法是在原始的 DIL 电路基础上加入了两支上拉 PMOS 管，如图 4.5c)DILP。与 DIL 对比这种结构中输出转换独立进行，反向锁也仅仅用来完成锁存功能。但同时这种电路存在较高的输入电容，考虑到能耗问题，对结构进一步该进，将上拉 P 管换作 N 管，如图 4.5d)DILN。在微流水线环境下，DILN 电路比 DILP 电路有着更好的应用前景。但 DILN 有个缺陷就是它的上升与下降时间不对称。

一定范围内增大锁存结构的尺寸有助于增大电路可实现的最大频率值。但由于大尺寸锁存结构对工艺实现和能耗都有不利影响，而且频率上受益也非常有限，所以这种大尺寸的改变一定情况下是不可取的。实际的改进方法是在 DILN 结构的输出端加上一对反相器，称这种结构为 MDILN。

双轨结构电路有着自身的优势，但同时应该注意：双轨结构两个互补输出之间存在一个产生时间上的差异。同步电路中，由于存在时钟而且没有反馈，这种差异不会带来很大的麻烦。但在异步电路中，这种时间的差异常会导致门电路不能对其输入产生正确的响应，导致电路功能失效。

4.1.3 C 单元单轨与双轨电路特性比较

对C单元各种单轨和双轨实现结构，图4.6给出了Energy-Frequency对比仿真结果[42]。图中各个数据点都是取自于不同的尺寸下，延曲线从左向右，电路的尺寸逐渐增大。很明显，外围的曲线代表了更好的Energy-Frequency平衡。0.45GHz是动态电路结构所能达到的最大频率值，此点处的虚线是大多单轨电路曲线的渐进线，双轨电路似乎也不能跨过这个界限，但对DILN与MDILN进一步优化，可以提升它的频率性能。SC、SIL与MSIL电路的曲线都向渐进线接近，但SS结构的曲线在相对低一些的频率值时达到顶点，这是由SS电路的对称结构决定的。

SS 结构的曲线是最接近动态结构曲线的，原因是 SS 电路引入最为简单的锁存结构。而 SIL 电路 keepers 中包含一个反相器，它会影响到输出转换。正因这点，SIL 电路的曲线距动态实现最远。MSIL 电路相对 SIL 在这方面有所改进。

考虑到 Energy-Frequency 的平衡，在频率要求低于 0.39GHz 的情况下，SS 电路是静态电路中的首选。从图中可以看出：SS 电路可以在能耗 12.6pJ 时实现

0.375GHz 的频率。而相同的频率值时，SC 与 MSIL 电路功耗为 22pJ，MDILN 为 27pJ，MDIL 为 33pJ，SIL 为 42pJ。如果设计者要求超过 0.4GHz 的频率指标，那么他可以选取 MDILN 电路。但同时考虑到双轨电路存在的互补输出时间分歧问题，选用这种电路结构时应该保持谨慎。

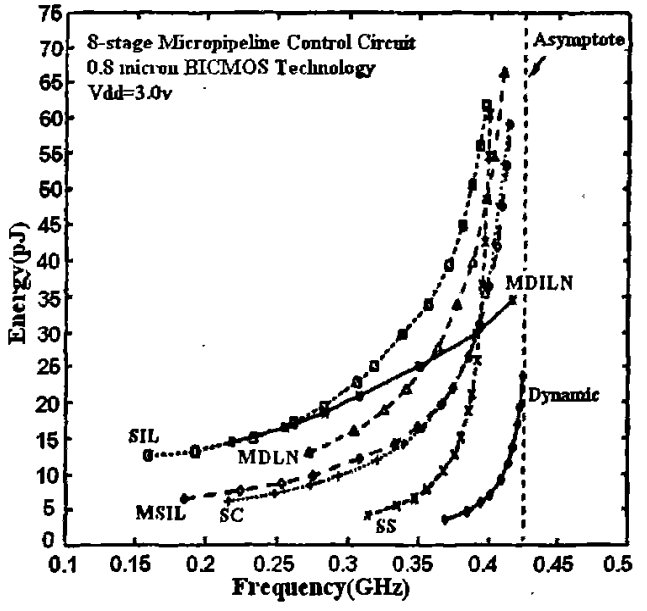


图 4.6 微流水线环境下 C 单元单轨与双轨电路仿真结果

4.1.4 结论

本章中我们介绍了 C 单元的单轨与双轨等八种电路结构，其中包括四种单轨和四种双轨结构。对每种结构的特点和优化途径都作了研究，并且基于仿真结果对各种结构作了详尽的对比。

每种电路结构各有其特点和优势，实际中可以根据具体的要求来选用不同的结构。单轨电路中，反向锁存 C 单元在面积上有明显的优势，在更关心面积的环境下可以采用此种结构。对称 C 单元本身具有对称结构，并且引入最为简单的锁存结构，它是低功耗耗和高性能设计中的较好选择。一些双轨电路结构能够实现比单轨结构更高的频率，在要求高频的情况下可以采用。但由于互补输出分歧问题的存在，在选用双轨结构时应保持谨慎。

4.2 异步握手电路基本单元

在实现异步握手协议时，也有一些常用的基本单元。这里我们主要介绍三种异步握手电路基本单元：Fork, Join 及 Merge 单元。三个基本单元组成结构中都用

到了异步 C 单元。Fork 与 Join 的单元结构比较简单，而 Merge 电路中用到了多个 C 单元、或门及 Mux 来实现，相对复杂一些[25]。

作为握手电路，我们同样利用这三种基本电路实现四种不同的握手协议，图 4.7 为此三种结构的四相单轨及双轨结构。这里我们主要以这两种结构为例，对三个异步握手电路基本单元做研究。

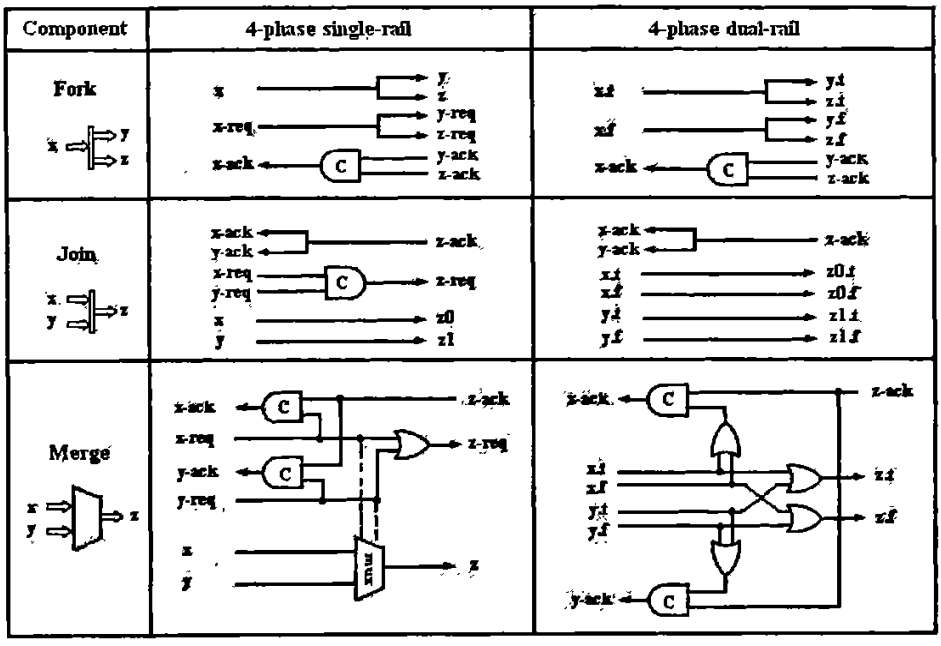


图 4.7 异步握手电路基本单元：Fork, Join, Merge

出于简化考虑，上图的 fork 单元设定只有两个输出信道，join 和 merge 单元也只假定存在两个输入。而且，所有的信道设定为 1-bit 信道。当然我们可以把它类推到更多的输入输出，也可以扩展到 n-bit 信道，但基本的工作方式是一致的。

A. 四相 fork, join 及 merge 单元

Fork 单元是一个单输入多输出结构，它将输入数据复制为多个输出。而 join 单元则是多输入单输出结构。这两种单元常常用于静态数据流结构中。

Fork 单元输出端的两个 Ack 信号通过 C 单元得到一个输入 Ack 信号。相似地，四相单轨 join 单元也利用 C 单元将输入端的 Req 信号合并为输出端的一个 Req 信号。四相双轨 join 与 fork 单元不包含任何主动请求信号，这是因为 Req 信号通过数据来编码完成。

这里我们主要从工作机理上对二者做探讨，具体的实现结构可以有多种选择：比如，fork 单元可以将输入数据分裂，这样它与 join 单元就显得更加对称。各种不同的选择仅仅是在输入数据到输出的转换结构上有不同。从控制机制的角度来看，它们都是相同的：join 单元同步化多个输入信道，而 fork 单元同步化多个输

出信道。

Merge 单元的结构相对稍微精细一些。输入信道的握手信号是相互独立的，merge 简单地传输输入握手信号到输出端。

四相单轨 merge 由一个异步控制电路与一个由输入 Req 信号控制的多路复用器组成。控制电路可以如下解释：

输入端的 Req 信号相互独立，简单地通过或运算来得到输出端的 Req 信号。对于每个输入信道，若输入信道载有有效数据，C 单元将对输出信道的 Ack 信号做出响应，产生一个 Ack 信号。举个例子：当 x-req 与 z-ack 都变高，驱动 x-ack 的 C 单元将置高。而当 x-req 变低（表示数据传输完成），作为响应 z-ack 信号也变低，这就足够使得 C 单元置位。对驱动 y-ack 的 C 单元，情况是一样的。

四相双轨 merge 单元大致是相似的。由于 Req 信号在数据中编码，两个输出信号 z.t 与 z.f 都用到了或门来实现。若输入信道存在有效数据，作为输出信道 Ack 信号的响应，产生一个输入信道 Ack 信号。例子中假定了 1-bit 宽的信道，所以后面用到了或门，但对 n-bit 宽的信道来说，则需要用到一个完成检测器。

B. 两相 fork, join and merge 单元

两相单轨 fork 与 join 单元与相应的四相单轨单元是相同的（假定所有信号起始都为低）。两相单轨 merge 结构比较复杂，与四相单轨有些不同。观测单独的 Req 或 Ack 信号，它们不停的上升或下降交替跳变。输入端的 Req 信号跳变转换与作为响应的输出端 Req 信号跳变没有任何必然关系；同样地，输出端的 Ack 信号与作为响应的输入端的 Ack 信号之间也没有任何关系。因此需要用到一些存储单元来保持电路产生的 Req 与 Ack 信号。这增加了电路结构复杂性。

4.3 异步加法器

异步加法器也是异步设计中一个最为基本的标准单元。同样地，它也可以通过单轨与双轨编码的方式实现[45]。本节中主要对异步加法器的各种独特的实现形式作了探讨。各种结构各有其特点，各有优缺点，考虑到不同的环境可以采用不同的结构。

4.3.1 异步单轨加法器

图 4.8a)为用多路器实现的单轨异步一位全加器。这种加法器由独特的数据部分及控制部分组成。加法器的数据通路在 Sum 输出端产生求和结果，在 Cout 输出端产生进位信号。表 4.2 为进位输出真值表。加法器的控制结构是用来控制在什么时候进位输出可由环境获知。当在输入端 A、B 数据准备就绪后，nStart 输

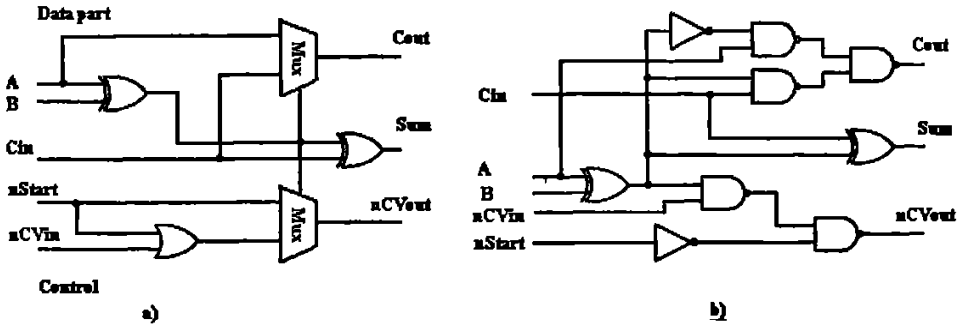


图 4.8 单轨异步 1 位全加器的实现：a）用多路器；b）用逻辑门

入端将产生一个零电平激活的 nStart 信号。如果 A、B 的值相同，nStart 信号将被传输到 nCVout 端，否则，nCVin 与 nStart 信号经由或门及多路器传输到 nCVout 输出端。此加法器控制部分采用四相信号协议。图 4.8b)为电路的逻辑门实现结构，工作方式与前者完全相同。

表 4.2 一位全加器的进位输出真值表

Input		Output
A	B	Cout
0	0	0
0	1	Cin
1	0	Cin
1	1	1

图 4.9 所示为八位异步单轨全加器。此结构中，所有一位全加器串型连接，即前一个一位全加器的进位输出 Cout 及进位有效输出端 nCVout 连接到下个一

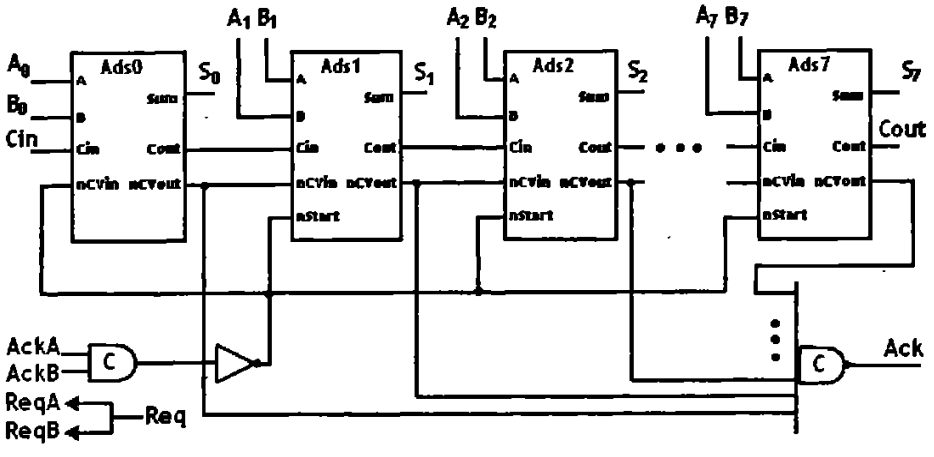


图 4.9 八位异步单轨全加器

位全加器的进位输入 Cin 及进位有效输入端 nCVin。加法器的控制输出 Ack 是由八输入对称 C 单元的“反”产生的，这个 C 单元的各个输入分别接到八个一位全加器的 nCVout 端。最后一个全加器的进位输出信号就是整个八位全加器的进位输出。全局的 nStart 信号输给所有一位全加器。第一个全加器(Ads0)没有 nStart 输入端，全局的 Start 信号接在它的 nCVin 输入端。全局的 nStart 信号需要存在一定的延时以保证 Ads0 的 Cout 信号达到稳定状态。

求和运算的请求是由加法器的 Req 输入端的环境来发送的。当 A、B 输入端数据就绪，二输入 C 单元的 AckA、AckB 两个确认信号置高。通过 C 单元及反相器后，一个零激活的低电平全局 nStart 信号就会传输给所有 1 位加法器的相应输入端，加法器开始运算。当八输入 C 单元的 Ack 输出端出现上升信号时，就表明此求和过程完成。一旦求和结果可知，在 Req 输入端的请求信号就重返为 0，AckA、AckB 确认信号也都置位为 0。二输入 C 单元也就被复位，随之，全局 nStart 信号变为高。当加法器的 Ack 输出端的确认信号被复位时，整个握手过程完成。

这里我们对 nCVout 的布尔关系式做一讨论：

$$nCVout_1 = \overline{\overline{nStart * hs_1 * \overline{nStart}}} = nStart * hs_1 + nStart = nStart \tag{4-2}$$

$$nCVout_i = nStart * hs_i + nStart = nStart, i = 2,3\cdots,7 \tag{4-3}$$

可以看出，控制逻辑中存在着冗余逻辑，在对单元的测试中必须去除这个冗余逻辑，需要一个特殊的测试模式来完成[46][47]。

为增加可测性，可以对单轨结构进行改进，如图 4.10。

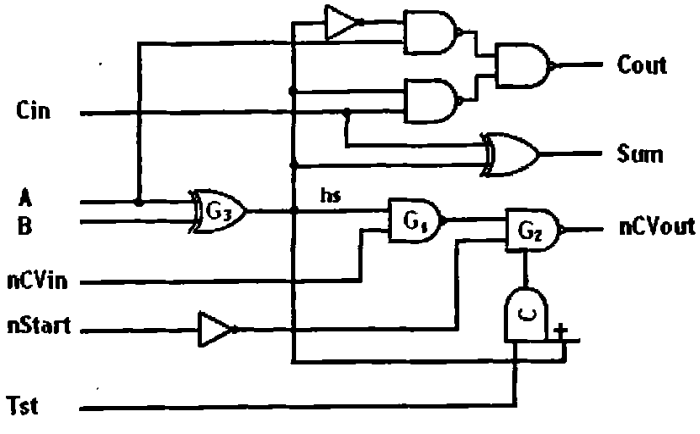


图 4.10 可测异步单轨加法器

此种结构有两种工作模式：正常工作模式和测试模式。模式的选择是通过信号 Tst 来确定的。当 Tst 为高电平时，加法器工作于测试模式，当 T_{st} 为低电平时，就处于正常工作模式下。图 4.11a)为 G_2 门的管级结构，它可作为两输入与非门也

可作为反相器来使用，这取决于它的控制端。若 Om 为低，G_2 就表现为一个两输入与非门。否则，G_2 变为一个反相器。图 4.11b)为可测单轨全加器的控制逻辑。

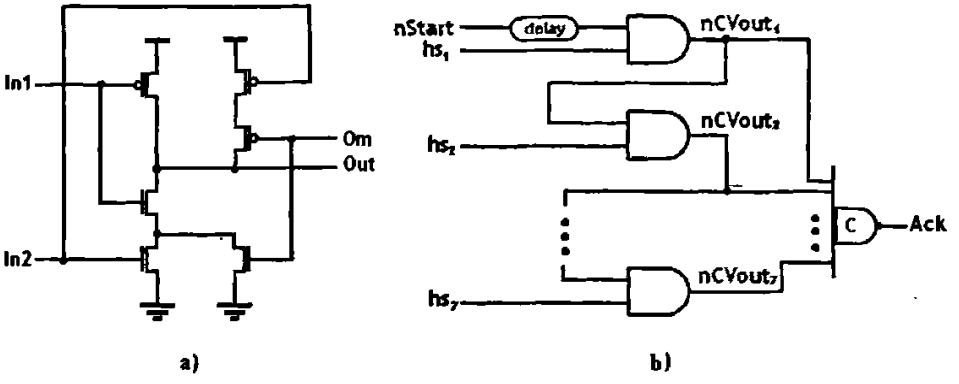


图 4.11 a)可测全加器 G_2 的内部结构；b)可测单轨全加器的控制逻辑

测试模式中，去除了控制部分的冗余逻辑，增加了其可测性。但由于使用了 C 单元，所以在面积上有所增加。

4.3.2 异步双轨加法器

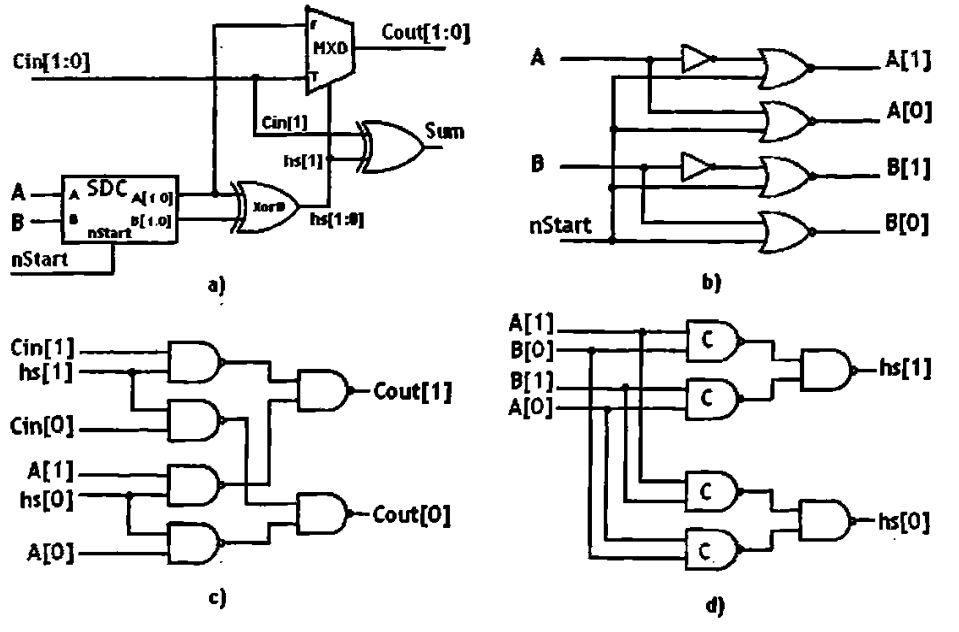


图 4.12 a)一位双轨异步全加器；b)单轨双轨转换电路；c)双轨异步多路器；d)双轨异或门

图 4.12a)为双轨异步一位全加器电路图。它包含一个单轨到双轨的数据转换

模块（SDC），双轨异或门及一个双轨多路器。单双轨转换器将从 A、B 输入的单轨数据转换成双轨数据形式。图 4.12b)给出了用逻辑门实现的转换模块。当 nStart 信号为高电平时，转换模块的各输出都将保持为低电平。如要完成数据转换，则应使 nStart 信号设置为低电平。双轨多路器及异或门的结构分别如图 4.12c)及 d)所示。在设计异或门时，对称的 C 单元的使用可以保证它的延时无关性。加法运算的单轨形式结果 Sum 通过 Cin[1]与 hs[1]两者的异或运算得到。

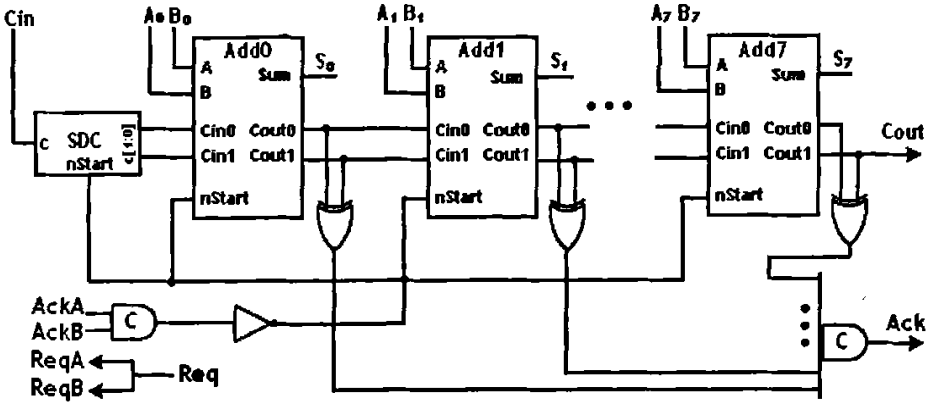


图 4.13 八位异步双轨全加器

图 4.13 为双轨异步八位加法器的实现电路。加法器的输入与输出均为单轨编码。在输入端 A、B 及 Cin 数据就绪后，AckA 与 AckB 处的确认信号均被设置为高电平。这样，nStart 信号变为低，电路开始进行求和。如果所有一位全加器的进位输出 Cout[1]与 Cout[0]不一样，那么通过八输入 C 单元后输出 Ack 处出现上升电平转换，那么就表明运算完成，数据输出有效。最终的进位输出是从最后一个全加器的 Cout[7]得到的。当加法器输出成功传输后，环境的握手请求信号 Req 重返为 0，确认信号 AckA、AckB 随即变为低，nStart 信号变高。最后，加法器的所有输出都变为 0，这使得八位全加器的 Ack 信号也变低，完成一个四相通信过程。

4.3.3 异步混合加法器

之所以称为“混合型”，是因为：加法器中的某些模块利用双轨输入数据；同时，混合型加法器具有类似于单轨加法器的控制逻辑。SDC 转换模块将 A、B 单轨数据转换成双轨形式。双轨异或门(XorD)的输出单轨多路器(MX)的控制端，MX 的两输入分别连到加法器的 Cin 及转换模块的 A[1]输出上。加法器的控制部分利用双轨异或门的两输入来产生一个高电平激活的进位信号 CVout。例如：当

hs[0] =1，输入 A、B 一致， 使得 CVout 输出变为高，就表明求和过程已完成。当输入 A、B 不一致，hs[1] =1 时，C 单元就处于等待信号状态。当输入 CVin 为高时，C 单元的输出就为高。随即会在 CVout 输出端产生一个上升信号。

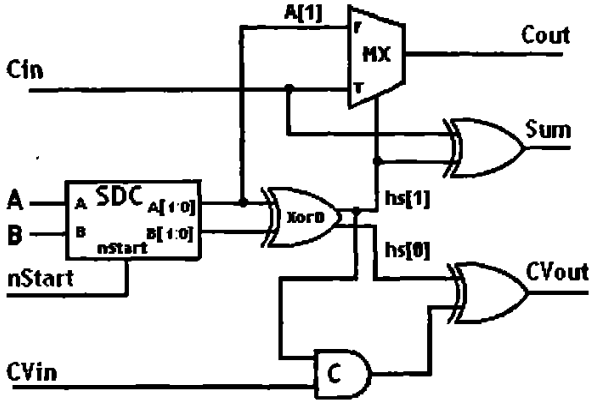


图 4.14 混合型异步加法器

混合型八位加法器的设计与图 4.9 所示的单轨加法器类似。在加法器的输入端准备好数据后，nStart 信号被设置为 0，数据转换为双轨形式。当多输入对称 C 单元的输出端 Ack 处出现上升信号时，就表明求和过程已完成。当 nStart 信号重返为 1 时，加法器的数据与控制输出都将被复位。为了使混合型加法器的进位输出重返为 0，所有一位全加器的控制通道中的 C 单元都必须复位（见图 4.14）。如果此时所有 $hs_i[1](i=0, 1,...,7)$都为高，那么加法器中的控制部分的所有单元的复位只能从第一个 1 位全加器开始顺序完成，复位的速度比较慢。显然，这是混合型加法器中的一个最坏情况。

最后我们对以上这三种加法器做一对比。可以看出，单轨结构所用门数最少，结构最为简单，速度最快。但单轨控制结构中存在冗余逻辑，可测性比较差。可测单轨结构优化了它的可测性，但面积上有所增加。双轨及混合型加法器的可测性比较好，但面积比较大，性能也相对比较差。混合型的结构在面积上是单轨与双轨形式的折中，但它的速度是三者中最慢的。因此，在对面积和性能要求比较高，并且具有完备测试模式的情况下可以使用单轨加法器结构。而在测试工作非常重要，而对性能和面积要求不是很高的情况下可以采用双轨或混合结构。

4.3.4 各种结构仿真结果及性能对比

我们使用 CHRT 0.35μm 工艺库文件实现了单轨、双轨及混合异步加法器结构，图 4.15、4.16 与 4.17 为三种结构的仿真波形。

通过仿真结果我们对以上几种结构做了对比。表 4.3 为性能参数对比。

表 4.3 三种加法器性能比较

	单轨	双轨	混合
平均延时(ns)	0.32	0.57	0.68
功耗(pW/Hz)	2.61	5.34	4.72

可以看出单轨结构所用门数最少，结构最为简单，平均延时最小(0.32ns)，功耗也最小(2.61pW/Hz)。但单轨控制结构中存在冗余逻辑，可测性比较差。可测单轨结构优化了它的可测性，但面积上有所增加。双轨及混合型加法器的可测性比较好，但结构要复杂的多，面积大。从表中看出它们的延时分别为单轨结构的 178%与 213%，功耗分别为单轨的 205%与 181%。混合型加法器在规模上是单轨与双轨结构的折中，但它的速度是三者中最慢的。因此，在对面积和性能要求比较高，并且具有完备测试模式的情况下可以使用单轨加法器结构。而在测试工作非常重要，而对性能和面积要求不是很高的情况下可以采用双轨或混合结构。

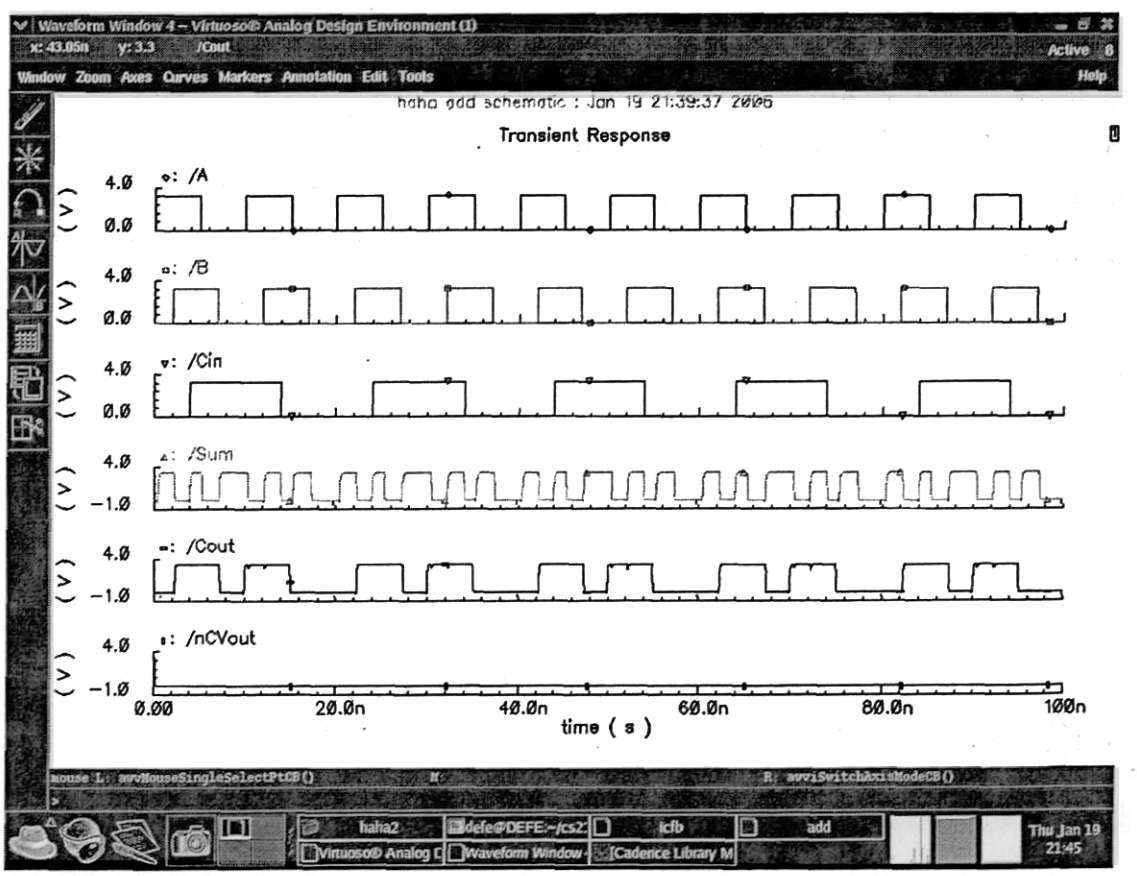


图 4.15 异步单轨加法器仿真波形

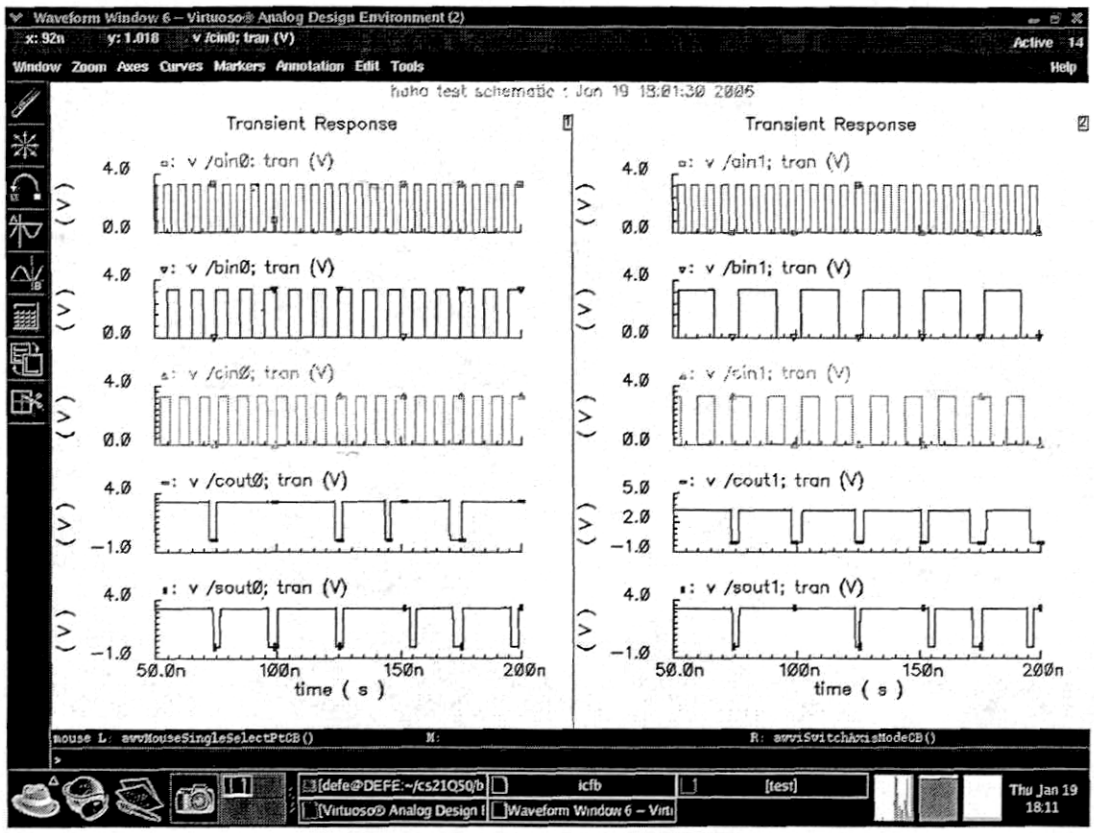


图 4.16 异步双轨加法器仿真波形

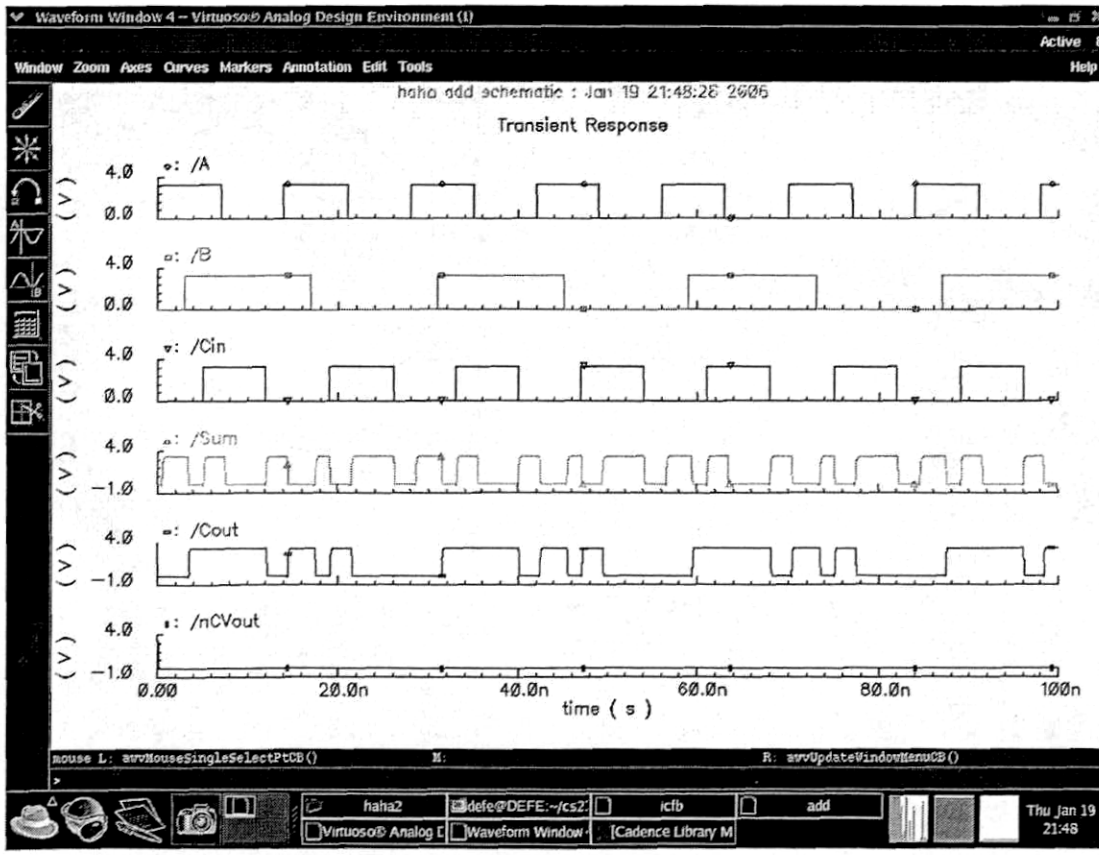


图 4.17 异步混合加法器仿真波形

第五章 总结与展望

本文主要介绍了异步电路的独特优势，对异步逻辑的时序模式、编码方式及通讯协议做了深入的探讨研究。设计实现了异步 C 单元、Merge、fork、join 及异步加法器等基本单元，并对各种结构进行了详细的仿真及优化。论文主要工作和结果归纳如下：

1)简单介绍了近年来集成电路随半导体制造工艺的不断进步而出现的诸如功耗、时钟偏移与连线延迟比重增加等难题；从异步电路的低功耗、高性能、无时钟偏移等独特优势入手，解释了为什么异步电路可以解决这些问题。

2）研究了异步电路的设计技术，包括异步时序模式、单双轨信号编码方式与四相及两相信号协议等。对四种异步握手通信协议作了重点探讨。阐述了异步集成电路发展的历史，相关的 EDA 工具及异步微处理器的发展现状。

3）设计实现了异步 C 单元的单轨与双轨多种实现结构，从延时、功耗等参数入手对各种结构进行了详细的仿真和优化，并通过对比，分析了各种结构的性能特点及适合的应用环境。

4）研究了 merge, fork, join 等异步握手电路基本单元，对各种单元的结构及功能进行了探讨；实现了单轨、双轨及混合异步加法器，分析探讨了每种实现结构的工作方式及特点，同样地也对各自的应用环境作了讨论。

本文的工作中对异步电路设计技术的研究还缺乏更加深入的理解和探索；对异步 C 单元、异步加法器等单元电路的设计有待实际应用的检验；另外由于时间有限，对异步其它基本单元的研究工作还有欠缺。对异步电路设计技术及基本单元电路的研究工作还有待以后进一步继续。

异步电路在低功耗、高性能及无时钟偏移等各方面的独特优势，已经越来越多地引起了设计者的关注。对异步逻辑设计技术及异步基本单元电路的研究具有很大的理论意义及实用价值。虽然目前由于异步电路设计的复杂性及缺乏成熟的 EDA 设计工具，异步电路的应用还非常有限，很多应用只是在部分模块或者模块互联上。但我们有理由相信，随着异步设计理念的不断发展，随着对异步电路设计技术及单元电路研究的不断完善，随着异步电路 EDA 工具的不断成熟，异步集成电路将会带来更大的理论及实际应用价值，甚至完成数字集成电路设计的一场革命。

致 谢

本论文是在导师杨银堂教授的关心和悉心指导下完成的。杨老师渊博的学术素养、对科学前沿发展的准确把握，以及求实创新的科研精神令我受益匪浅，他严谨求实的治学作风，一丝不苟的敬业精神给我留下了深刻的印象，从杨老师那里，不仅使我学到了做学问的思想方法，更使我明白了许多做人做事的道理，并将让我终身受益。

感谢课题组所有老师在本论文完成过程中的大力支持和帮助。对李跃进老师、柴常春老师、汪家友老师、傅俊兴老师、刘毅老师和贾护军老师在学习和生活中给予我的大力支持，在此表示真诚的感谢。

由衷感谢徐阳扬师兄在论文工作中对我的无私帮助和指导。对丁瑞雪、邓斌、韩茹、徐宏才、于建旺、宋久旭等同窗好友在日常的学习、生活以及论文的写作过程中所给予的帮助表示最诚挚的谢意。

尤其感谢我的父母、妹妹及所有亲人朋友，他们在我的成长中倾注了无数心血，每一步的成长都离不开他们的谆谆教导和鼎力支持。

最后，再次向所有关心、爱护、帮助过我的人们表示最衷心的谢意!!

参考文献

[1] Sakurai. "Design Challenges for 0.1um and Beyond". ASP-DAC 2000

[2] Van Berkel, etc. Scanning the Technology-Application of Asynchronous Circuit. Proceedings of the IEEE. Feb.1999. Vol.87, No.2. 223-233

[3] Scott Hauck. Asynchronous Design Methodologies: An Overview. Proceedings of the TEEE. Jan. 1995. Vol.83, No.1. 69-93

[4] G.Gopalakrishanan. Guest Editor's Introduction to the Special Issue on A sunchronous Systems, Integration. The VLSI Journal. 1993. 233-239

[5] Kinniment, D. Asynchronous design methods. IEE Colloquium. May.1998. 2/1 - 2/4

[6] 赵冰，黑勇，仇玉林. 异步集成电路设计的研究与进展. 半导体技术. May.2004. vol.29. No.5. 10-15

[7] Mark.B; Josephs, etc, Modeling and Design of Asynchronous Circuits, Proceedings of the IEEE, Jun.1994. Vol.29, No.6. 663-670

[8] Dake Liu; Chister Svenssen. Power Consumption Estimation in CMOS VLSI Chips. IEEE Journal of Solid-State Circuits. Jun.1994. 663-670

[9] A.P.Chandrankasan, etc. Low Power Digital VLSI Design. IEEE Journal of Solid-State Circuit. Apr.1992. Vol.27, No.4. 473-484

[10] P. B. Endecott. SCALP: A Superscalar Asynchronous Low-Power Processor, PhD dissertation. University of Manchester. 1996

[11] P. B. Endecott, "Processor Architecture for Power Efficient and Asynchronous Implementation", Master Thesis, University of Manchester, 1993

[12] 俞颖. 异步低功耗微控制器芯片的研究与设计. 2003

[13] 周乾. 椭圆曲线加密算法的异步电路硬件实现. May.2003

[14] Josephs M.B.; Nowick S.M.; VanBerkel C.H. Modeling and Design of Asynchronous Circuits. Proceedings of the IEEE. Feb.1999. Vol 87, Issue 2. 234-242

[15] Greenstreet, M.R.; De Alwis, B. How to achieve worst-case performance (self-timed circuit design). Asynchronous Circuits and Systems, 2001. Seventh International Symposium. Mar.2001. 206-216

[16] David, I.; Ginosar, R.; Yoeli, M. An efficient implementation of Boolean functions as self-timed circuits. Computers, IEEE Transactions. Jan.1992. Vol.41, Issue

1. 2-11

[17] MULLER D E, BARTKY W S. A theory of asynchronous circuits Ⅰ. Digital Computer Laboratory 75.University of Illinois, 1956

[18] MULLER D E, BARTKY W S. A theory of asynchronous circuits Ⅱ. Digital Computer Laboratory 75.University of Illinois, 1957

[19] DAVIS A L, NOWICK S M. Asynchronous circuit design: motivation, background and methods. Asynchronous Digital Circuit Design. British Computer Society, 1995. 1-49

[20] GARSIDE J D, BAINBRIGE W J. ANULET3i: an asynchronous system on chip[A]. Proc of 6th Int Symp on Advanced Research in Asynchronous Circuits and Systems[C], Israel, 2000, 162-175

[21] Martin A.J. Programming in VLSI from communicating processes to delay-insensitive circuits. 1990. 1-64

[22] Muller D.E., Bartky W.C. A theory of asynchronous circuits. Hardward University. 1959

[23] Kondratyev A., Kishinevsky M., Yakovlev A. Hazard-free implementation of speed-independent circuits. IEEE Trans CAD. 1998. 17(9):749-771

[24] Myers C.J. Computer aided synthesis and verification of gate-level timed circuits. 1995

[25] Jens Spars; Steve Furber. Principles of asynchronous circuit design-A system perspective. Boston/London. Kluwer Academic Publishers. Sep.2001. 9-15

[26] Tugsiriavisut, S.; Beerel, P.A. Control circuit templates for asynchronous bundled-data pipelines. Design, Automation and Test in Europe Conference and Exhibition, 2002. Mar.2002. 1098

[27] Poole, N.R. Self-timed logic circuits. Electronics & Communication Engineering Journal. Dec.1994. Vol 6. Issue 6. 261-270

[28] Nielsen, L.S.; Sparso, J. Designing asynchronous circuits for low power: an IFIR filter bank for a digital hearing aid. Proceedings of the IEEE. Vol.87, Issue 2. Feb.1999. 268-281

[29] 陶冶，赵千川，异步电路的计算机辅助设计软件，计算机工程与应用，2005，Vol.27，84-88

[30] D A Edwards, W B Toms. Design, Automation and Test for Asynchronous Circuits and System. Information Society Technologies(IST) Program Concerted Action Thematic Network Contract, IST-1999-29119, 2nd Edition, 2003-02

[31] Dung Edwards, Andrew Bardsley, Lilian Janin et al.Balsa: A Tutorial Guide. Version V3.4.1, 2004-05

[32] http://www.theseus.com/FramesTechDesFlo.htm

[33] http://www.asyn.ece.utah.edu/tools/autacsman.html

[34] http://www-2.cs.cmu.edu/-cmuvhdl/

[35] Martin, A.J.; Nystrom, M.; Wong, C.G. Three generations of asynchronous microprocessors. Design & Test of Computers, IEEE. Nov.-Dec.2003. Vol.20, Issue 6. 9-17

[36] Endecott, P.B. Superscalar instruction issue in an asynchronous microprocessor. Computers and Digital Techniques, IEE Proceedings. Sep.1996. Vol.143, Issue 5. 266-272

[37] Werner, T.; Akella, V. Asynchronous processor survey. Computer. Nov.1997. Vol.30, Issue 11. 67-76

[38] Furber, S.B.; Day, P.; Garside, J.D. The design and evaluation of an asynchronous microprocessor. IEEE International Conference. Oct.1994. 217-220

[39] STEVEN M N. Automatic synthesis of burst-mode asynchronous controller [D]. Ph D Thesis, Stanford University, 1993

[40] VAN GAGELDONK H, VAN BERKEL K. An asynchronous low-power 80C51 microcontroller[A]. Proc of 4th, Int Symp on Advance Research in Asynchronous Circuits and System[C]. San Diego, 1998. 96-107.

[41] Wuu T Y.; Vrudhula S B K. A design of a fast and area efficient multi-input Muller C-element. Very Large Scale Integration (VLSI) Systems, IEEE Transactions. June.1993,Volume 1,Issue 2,215–219

[42] Shams M.; Ebergen J C.; Elmasry M I. Optimizing CMOS Implementation of the C-element. 1997 IEEE International Conference on 12-15 Oct.1997, 700–705

[43] Shams M.; Ebergen J C.; Elmasry M I. A comparison of CMOS implementations of an asynchronous circuits primitive: the C-element. Low Power Electronics and Design. Aug.1996, 93–96

[44] Shams M.; Ebergen J C.; Elmasry M I. Modeling and comparing CMOS implementations of the C-element. Very Large Scale Integration (VLSI) Systems, IEEE Transactions. Dec.1998,Volume 6,Issue 4, 563–567

[45] J. D. Garside, A CMOS VLSI implementation of an asynchronous ALU, IFIP WG 10.5 Working Conference on Asynchronous Design Methodologies, Manchester, 1993

[46] O. A. Petlin, C. Farmsworth, S. B. Furber, Design For Testability of an Asynchronous Adder, Design and Test of Asynchronous Systems, IEE Colloquium on, Feb.1996, 5/1-5-9

[47] J. D. Garside, A CMOS VLSI implementation of an asynchronous ALU in Asynchronous Design Methodologies, IFIP Transactions, 1993, vol.A-28, 181–207

科研成果

一、参与的科研项目

[1] 国防科技预研基金，FPGA 设计

[2] 国家自然科学基金项目，低功耗异步微处理器设计技术研究

二、发表论文

[1] 李俊杰，杨银堂，徐阳扬，异步电路 C 单元设计，第十四届全国半导体集成电路、硅材料学术会议，录用

[2] 李俊杰，杨银堂，徐阳扬，异步电路基本单元设计，西安电子科技大学研究生学术年会，录用