# 摘要

随着移动互联网的迅速发展，SOC作为便携式智能终端的信息处理能力和待机时间的要求也日趋提高。存储器作为SOC的重要组成，一般而言面积占整个SOC芯片面积的一半以上。尤其在高性能处理器中，SRAM占芯片的面积已经超过80%。存储器的性能对SOC芯片的影响很大。

如今的SRAM设计中，越来越多的低功耗技术和高速技术被提出，比如DVFS技术、电荷泵技术等。在诸多优化技术带来诸多便利的同时，SRAM的设计周期相应也延长了，尤其在仿真环节中，相比于原本就耗时的SRAM设计仿真周期而言，这些优化技术无疑也增加了仿真周期和设计周期。

本文主要内容和研究对象是：第一，介绍机器学习，详细阐述了最小二乘法回归算法；第二，介绍了标准单元库，分析了十个标准单元的时序，通过电路参数建模实现时序预测和通过时序建模实现时序预测；第三，介绍SRAM的电路结构、基本原理和特征化，主要对表征中的时序和功耗进行分析，并通过时序建模实现时序预测、按温度划分功耗实现功耗预测，分析并指出本方法在SRAM设计周期中对于设计周期和仿真周期的提升效果；第四，对本文的工作进行了总结，并对未来的工作进行了展望。

关键词：静态随机存取存储器 最小二乘法 编译器 表征 预测

作者：佘一奇

指导教师：张立军

# Abstract

With the rapid development of mobile Internet, the information processing capacity and standby time requirement of SOC as a portable intelligent terminal is also increasing. As an important part of SOC, memory accounts for more than half of the total SOC chip area. Especially in high-performance processors, SRAM accounts for over 80% of the chip area. The performance of memory has a great influence on the SOC chip.

Nowadays, more and more low-power technologies and high-speed technologies have been put forward in SRAM design, such as DVFS technology, charge pump technology and so on. While many optimization technologies have brought many conveniences, the design cycle of SRAM has been extended correspondingly, especially in the simulation link. Compared to the time consuming SRAM design simulation cycle, these optimization techniques undoubtedly increase the simulation cycle and design cycle.

The main content and research object of this paper are: Firstly, this paper introduced machine learning, and described the least square regression algorithm in detail. Part two introduced the standard cell library, and analyzed the timing of ten standard cells, and realized sequential prediction and through time series modeling to realize time series prediction through the modeling of circuit parameters. Then this paper introduced the circuit structure, basic principle and characteristic of SRAM. In this part, the time series and power consumption in the characterization are analyzed. The time series prediction was realized and the power consumption was divided according to the temperature division. Finally, the improvement of the design cycle and the simulation cycle in the SRAM design cycle is also pointed out. The last part summarized the work of this paper and forecasted the future work.

Keywords: Static Random Access Memory, Least Squares, Compiler, Characterization, Prediction.

Written by She Yiqi

Supervised by Zhang Lijun

**目录**

[摘要 1](#_Toc511119394)

[Abstract 2](#_Toc511119395)

[**第一章 绪论** 1](#_Toc511119396)

[1.1 半导体集成电路介绍 1](#_Toc511119397)

[1.1.1 存储器及SRAM介绍 1](#_Toc511119398)

[1.1.2 SRAM设计优化技术 2](#_Toc511119399)

[1.2 SRAM仿真工作存在的优化可能 3](#_Toc511119400)

[1.3论文的主要内容 4](#_Toc511119401)

[第二章 最小二乘法 5](#_Toc511119402)

[2.1机器学习 5](#_Toc511119403)

[2.1.1 机器学习的定义 5](#_Toc511119404)

[2.1.2 机器学习算法分类 5](#_Toc511119405)

[2.2最小二乘法回归 6](#_Toc511119406)

[2.2.1 线性回归模型 6](#_Toc511119407)

[2.2.2 最小二乘法 6](#_Toc511119408)

[2.3 算法实现 8](#_Toc511119409)

[第三章 28nm标准单元库时序预测 9](#_Toc511119410)

[3.1 标准单元库 9](#_Toc511119411)

[3.1.1 标准单元方法 9](#_Toc511119412)

[3.1.2 标准单元库 10](#_Toc511119413)

[3.1.3 标准单元库流程 11](#_Toc511119414)

[3.2 时序 13](#_Toc511119415)

[3.2.1 Liberty库 13](#_Toc511119416)

[3.2.2 数据收集 15](#_Toc511119417)

[3.3 时序预测 18](#_Toc511119418)

[3.3.1 电路参数方法 18](#_Toc511119419)

[3.3.2 时序方法 20](#_Toc511119420)

[3.4 本章小结 22](#_Toc511119421)

[第四章 SRAM编译器 23](#_Toc511119422)

[4.1 SRAM基本原理 23](#_Toc511119423)

[4.1.1 SRAM架构 23](#_Toc511119424)

[4.1.2 SRAM存储单元 24](#_Toc511119425)

[4.1.2 SRAM Compiler 25](#_Toc511119426)

[4.2 Memory Characterization 29](#_Toc511119427)

[4.2.1 仿真工作 29](#_Toc511119428)

[4.2.2 特征 30](#_Toc511119429)

[4.2.3 INSTANCE 31](#_Toc511119430)

[4.2.4 IO 33](#_Toc511119431)

[4.2.5 PVT Corners 33](#_Toc511119432)

[4.3 本章小结 35](#_Toc511119433)

[第五章 SRAM表征数据预测 36](#_Toc511119434)

[5.1 时序 36](#_Toc511119435)

[5.1.1 时序数据 36](#_Toc511119436)

[5.1.2 时序预测 42](#_Toc511119437)

[5.2功率 51](#_Toc511119438)

[5.2.1 功率数据 51](#_Toc511119439)

[5.2.2 功率预测 52](#_Toc511119440)

[5.3 本章小结 58](#_Toc511119441)

[第六章 总结与展望 59](#_Toc511119442)

[6.1 论文总结 59](#_Toc511119443)

[6.2 工作展望 59](#_Toc511119444)

[参考文献 61](#_Toc511119445)

[攻读学位期间本人出版或公开发表的论著、论文 62](#_Toc511119446)

[附录 63](#_Toc511119447)

# 第一章 绪论

## 1.1 半导体集成电路介绍

随着物联网和移动互联网的迅捷发展，集成电路芯片被广泛应用于各种与日常生活密切相关的领域，比如个人电脑，智能手机，可穿戴设备等。其中手机的出货量已经超过71亿部。正是六十多年来集成电路的高速发展为基础，才有如今辉煌的电子信息产业链[1]。

2017年全球集成电路芯片市场的销售额已经达到4000亿美元，与之相关的电子信息产业的产值达到了15万亿美元，已经超过2017年全球GDP的五分之一。

国际半导体产业协会（SEMI）发布的各家产业研究机构对2018年半导体产业营收的预测成长率为7～8%。在全世界集成电路行业高度竞争和发展中，我国的集成电路行业也有着很好的表现。其中，中国在智能手机为代表的通信模块方面和移动终端的AP方面已占据全球市场份额的17%左右。然而，在集成电路芯片应用的其他领域中，国产芯片所占有市场份额几乎没有。从国内的集成电路芯片发展的规模来看，我国集成电路芯片产业正在进入一个快速的发展期，近年来保持着近20%的增长率并持续增长，而且在2017年集成电路产业的产值已达到5000多亿[2]。即使在这种情况下，国内的集成电路芯片才只有30%左右能满足自身的需求，其余均依赖于进口，集成电路芯片的进口额已经超过了石油，成为了我国的需求量第一的进口商品，这种现状亟待依赖于我国集成电路行业的迅速发展来解决[3]。从集成电路行业与其密切影响的电子信息行业的发展来看，不论是中国还是全球，发展集成电路行业都具有非常重要的意义。

### **1.1.1 存储器及SRAM介绍**

图1-1 近年来SOC中存储器的使用情况

存储器作为现在超大规模集成电路（Very Large Scale Integration Circuit，VSLI）中非常重要的组成部分，随着近几年来半导体工艺技术的迅速发展，存储器在SOC中占据着越来越重要的比重，2014年的存储器占有SOC的面积已经达到了94%。与此同时，SRAM作为存储器家族的重要成员，也具有十分重要的意义。

存储器家族的成员繁多，通常都有多种分类方式。通常较为主流的分类方式是按照存取方式的不同进行分类，主要分为随机存取存储器（Random Access Memory，RAM）、顺序存取存储器（Serial Access Memory）和按内容寻址存储器（Content Access Memory）。

其中，随机存取存储器还可以根据断电后是否能够继续保存信息而分类，主要分为易失性存储器（Volatile Random Access Memory，VM）和非易失性存储器（Non-Volatile Random Access Memory，NVM）两类。通常易失性存储器又分为静态随机存取存储器（Static Random Access Memory，SRAM）和动态随机存取存储器（Dynamic Random Access Memory，DRAM）。

如今SOC处理器的主频已经达到4GHZ，其片外的大容量主存储器的频率只有600MHZ左右[4]。SRAM凭借面积小，容量合适，速度快和很好的鲁棒性等优势，作为SOC芯片的存储设备。SRAM的频率可以达到逻辑电路的频率，也就是说完全可以满足处理器的主频需求。SRAM在待机（Standby）的状态下功耗较低，具有更高的带宽，可以满足系统对带宽的需求[2]。SRAM还拥有更好的工艺缩放性，可以随着工艺节点的缩小得到更好的集成度和性能。SRAM在拥有这些其他存储器不可比拟的优势的同时，也意味着SRAM具备的设计难度和要求。

### 1.1.2 SRAM设计优化技术

戈登·摩尔在1973年曾经提出了，一个硅片上晶体管的数量每隔18个月就会翻一番，后来这一推测成为了著名的摩尔定律[5]。在过去集成电路迅速发展的几十年中，集成电路的迭代更新一直印证着摩尔定律。平均一颗芯片上的晶体管数量一直保持大约每两年翻一番。然而随着SOC中晶体管数量的剧增，能耗问题、散热问题成为了SRAM设计的首要难题。其中，低功耗技术已经成为SRAM设计的首要考虑因素。

目前工业界常用的低功耗设计非常多，主要有电源关断（Power Shut-off）、门控时钟（Clock Gating）、多阈值电压（Multi-Vth）、多电源电压（Multi-Supply Voltage）、动态电压频率调节（DVFS，Dynamic Voltage and Frequency Scaling）、衬底电压偏置（Substrate Biasing）等。

其中，电源关断技术主要通过使能信号去控制一块区域的电源，去实现降低芯片整体的泄漏功耗。当使能信号处于无效状态的时候，控制的整个区域均不供电，因此不产生泄漏功耗。然而需要注意的是，电源关断方法在实现有效降低泄漏功耗的同时，需要引入更多的开关电路和控制电路，同时在电源系统中增加控制电路可能会让使能信号在开关的瞬间对附近区域的逻辑器件产生电压波动。因此考虑使用电路关断技术的同时，需要仔细考虑电源关断技术对电路产生的影响和变化进行评估。

相对于直接关断部分区域的电源的电源关断技术而言，门控时钟设计主要通过使能信号去控制时钟的传播，去实现降低芯片工作时的动态功耗。同样的，门控时钟技术也需要像电源关断技术一样的需要引入新的设计单元。门控时钟技术为电路带来了更多的信号控制的寄存器数量，与此同时控制层次也相应越高。门控时钟也是较为常用的降低动态功耗的设计手段。

多阈值电压技术则主要通过使用不同阈值电压的器件，来实现芯片泄漏功耗的降低。

多电源电压技术主要根据芯片的不同区域的电压需求，选择不同的电源电压值，从而实现降低动态功耗与静态功耗。多电源电压技术是静态的电压调整方法，不同的电源电压代表着不同的电路工作功耗和性能。

多电源电压、动态电压调节（Dynamic Voltage Scaling，DVS）以及动态电压频率调节技术均是通过调节电源电压来实现低功耗。

动态电压频率调整技术（DVFS）是根据芯片处于不同工作场景时，所需要运行的应用程序对计算能力的不同需要，动态的调整芯片的不同模块的工作频率和工作电压，从而达到降低功耗的目的[11]。

DVFS和DVS都是具有自适应性的技术，它们通过统计处理器的利用率来进行工作：当处理器的负载较小的时候，会降低工作频率和工作电压；当处理器处于额定负荷工作状态时，系统就会处于最大的工作频率和工作电压条件下。这些低功耗技术己被广泛应用于商用微处理器中，并具有着显著的节能特性[6-10]。

文献12中3D图形处理中有三个独立的模块，每个模块有不同的性能需求，将不同的三个模块应用DVFS技术可以实现每个功率区域的高效的能耗减少。最终，GPU系统产生相同输出结果的同时，应用DVFS技术的电源管理比正常工作的电源管理实现节省功率约65%。

## 1.2 SRAM仿真工作存在的优化可能

在各种层次的低功耗技术广泛应用在SRAM设计的时候，在带来显著减少功耗的同时，也给SRAM设计提出了相应的附属要求，以DVFS为例，不同模块动态调整工作电压和工作频率，对于设计的要求，尤其是仿真的要求提高了很多。通常SRAM编译器都有几千个实例，对于某个实例都要在不同的PVT（Process、Voltage、Temperature）进行仿真。DVFS使得SOC的电压节点增加了很多。以TSMC 28nm HPCPLUS为例，标准电压是0.9v，相对应的SRAM的签出需要6个PVT的仿真。电压节点从0.72v到1.08v时，以90mv为间隔，一共需要30个corner的签出。

以50个Intel（R） Xeon® 的4核CPU，其频率为3.60GHZ，以及足够的EDA licenses，Memory Compiler的表征仿真需要约340小时，基于机器资源和时间成本的考虑，以及更多优化技术的加入，未来SRAM Compiler表征所需的仿真将会呈现阶梯式的增长。本课题的出发点就是提出利用机器学习算法，通过已知的仿真数据构建模型，去预测未知的仿真数据，从而实现节省机器资源和减少时间成本的目的，进而优化整个SRAM Compiler的设计效率。

## 1.3论文的主要内容

本文主要基于当今微电子行业对于机器学习的应用现状与前景，总结了机器学习在存储器设计中应用的可行性与可靠性，确立了本文的课题：基于机器学习的28nm工艺SRAM Compiler特征化。围绕这一主题，基于SRAM编译器的设计现状，针对仿真过程的耗时长、占用资源比重大等不足，提出了应用机器学习算法简化仿真，并先后在标准单元库和SRAM Compiler的设计中应用，以期对未来的机器学习应用于集成电路设计的重要环节中有一定的指导意义。

本文的主要内容如下：

第一章为绪论，主要介绍了SRAM Compiler的研究背景、意义，系统阐述了机器学习应用于SRAM Compiler的应用现状和发展前景，提出了SRAM Compiler仿真工作方面存在的耗时长、资源占比大等问题，并针对这一问题提出了优化方法。

第二章首先简单的介绍了机器学习及其发展现状与应用前景，主要阐述了回归算法，着重分析了最小二乘法的算法原理和应用场景。

第三章介绍了标准单元库及标准单元时序，并通过最小二乘算法，拟合已有的时序数据，预测未知的时序数据。

第四章介绍了SRAM Compiler的设计流程，着重描述了表征的三个模块：时序、功耗和电容，并对三个模块分别使用最小二乘法进行数据的预测。

第五章为总结与展望，对本文的工作进行了总结，并对本课题的研究进行了展望。

# 第二章 最小二乘法

## 2.1机器学习

### 2.1.1 机器学习的定义

机器学习（Machine Learning）是一门人工智能的学科，机器学习主要讨论的内容是在“从有限观察发现观测数据中暗含的各种关系的数据分析”和“从有限观察概括特定问题世界模型的机器学习”的方法。机器学习领域的主要研究对象是人工智能。其研究热点是在经验学习中改进具体算法的性能。

机器学习主要应用于语音和手写识别、数据挖掘、生物特征识别、计算机视觉、医学诊断、自然语言处理、医学诊断、生物特征识别、搜索引擎、检测信用卡欺诈、证券市场分析、DNA序列测序、战略游戏和机器运用等领域。

### 2.1.2 机器学习算法分类

机器学习有很多算法，分类方式也有多种。主要的分类方式有基于学习策略的分类、按应用领域分类、基于所获取知识的表示形式分类、学习形式分类和综合分类等。其中最为普遍接受的是按学习形式分类。

通常的，机器学习按学习形式分类：监督学习（Supervised Learning）、非监督学习（Unsupervised Learning）和强化学习等几种学习形式。具有应用场景较多的是监督学习和非监督学习。

监督学习主要利用一组已知类型的训练数据样本，通过对样本数据的特征（Feature）和已知目标（Target）来推断一个功能的机器学习任务，并在未知的输入样本时也给出正确的输出结果，从而达到预测未知的目的；非监督学习是指数据样本中没有给出正确的输出结果信息。无监督学习希望从数据中挖掘的正是这一类信息，常见的例子有关联规则挖掘、聚类、离群点检测等等；强化学习是智能系统从环境到行为映射的学习，以使奖励信号（强化信号）函数值最大，强化学习不同于连接主义学习中的监督学习，主要表现在强化学习中由环境提供的强化信号是对产生动作的好坏作一种评价（通常为标量信号）。

其中根据输出结果是离散还是连续，监督学习可以分为分类问题和回归问题两大类。他们在文字、语音、图像识别、垃圾邮件分类与拦截、网页检索、股票预测等方面有着广泛应用。本课题使用监督学习中的回归算法：最小二乘法。回归分析中，如果只包括一个自变量和一个因变量，且二者的关系可用一条直线近似表示，这种回归分析称为一元线性回归分析。如果回归分析中包括两个或两个以上的自变量，且因变量和自变量之间是线性关系，则称为多元线性回归分析。

## 2.2最小二乘法回归

### 2.2.1 线性回归模型

考虑到本课题中数据的多维性，这里用多元回归方程表示：

（2.1）

式中：*θi*称为假设模型即映射输入特征*x*与结果*y*的线性函数*h*的参数（Parameters），为了简化表示，我们在输入特征中加入*X0* =1 ，于是得到：

（2.2）

参数*θ*和输入特征*x*都为矢量，*n*是输入的特征*x*的个数（不包含*x0*）。

使得预测值*h(x)*尽可能接近*y*，为了达到这个目的，我们对于每一个参数*θ*，定义一个**代价函数（Cost Function）**用来描述*h(x(i))'*与对应的*y(i)'*的接近程度：

（2.3）

于是我们的目标就变为了调整*θ*使得代价函数*J(θ)*取得最小值，方法有梯度下降法，最小二乘法等。

### 2.2.2 最小二乘法

最小二乘法通过最小化误差的平方和寻找数据的最佳函数匹配。利用最小二乘法可以简便地求得未知的数据，并使得这些求得的数据与实际数据之间误差的平方和为最小。最小二乘法还可用于曲线拟合。其他一些优化问题也可通过最小二乘法来表达。最小二乘法是一种直观的直接利用矩阵运算可以得到*θ*值的算法。

假设函数*f*是将*m\*n*维矩阵映射为一个实数的运算，即

（2.4）

并且定义对于矩阵*A*，映射*f*(*A*)对*A*的梯度为：

（2.5）

因此该梯度为*m\*n*的矩阵。

　 另外，对于矩阵的迹的梯度运算，有如下规则：

（2.6）

（2.7）

（2.8）

（2.9）

　 下面，我们将测试集中的输入特征x和对应的结果y表示成矩阵或者向量的形式，有：

（2.10）

（2.11）

对于预测模型有：

， （2.12）

即：

， （2.13）

于是可以得到：

（2.14）

所以可以得到：

（2.15）

于是，我们就将代价函数*J*(*θ*)表示为了矩阵的形式，就可以用上述提到的矩阵运算来得到梯度：

（2.16）

令上述梯度为0，得到等式：

， （2.17）

于是得到*θ*的值：

。 （2.18）

这就是最小二乘法得到的假设模型中参数的值。

### 2.2.3 算法实现

本课题所有工作都是在CentOS release 6.8（Final）平台完成，Linux内核版本为2.6.32。本文中SRAM批量仿真工作由C shell语言完成，仿真数据的采集与计算由Python语言编写实现，最小二乘法算法的回归与预测由Python和C++语言实现。

## 2.3 本章小结

本章首先介绍了机器学习的应用现状，并针对按学习方式指出了监督学习的回归算法。从解决多元回归问题的方式入手，阐述了最小二乘法用于回归问题的原理，并进行了详细的推理。最后说明了本课题中所有使用的算法实现方式。

# 第三章 28nm标准单元库时序预测

## 3.1 标准单元库

### 3.1.1 标准单元方法

在半导体设计中，标准单元方法（Standard Cell Methodology）是一种用于专用集成电路（Application Specific Integrated Circuits，ASICs）的数字逻辑的设计方法。标准单元方法通过将一个低级的集成电路逻辑单元的版图封装成一个抽象的逻辑表示形式，比如非门等。这就可以让电路设计师更专注于数字设计，而其他的设计师更专注于物理设计。随着半导体制造业的进步，标准单元方法使得成千上万的逻辑功能相对简单的电路，得以缩放到复杂的百万门级的SOC上。

标准单元是通过晶体管通过组合互联构成具有布尔逻辑函数，比如AND、OR、XOR、XNOR等，存储单元，比如Flip-Flop和Latch。最简单的标准单元比如NAND、NOR、XOR等布尔逻辑功能，尽管更为常用的是一些复杂的布尔逻辑单元：全加器，D触发器等。标准单元的布尔逻辑功能就是它的逻辑视图，其逻辑行为通常表现为真值表、布尔代数方程和状态转换表。其中布尔代数方程通常用于组合逻辑，状态转换表通常用于顺序逻辑。

通常，标准单元的初始设计开发在晶体管级，以晶体管网表或电路的视图的形式。网表是晶体管、晶体管连接、端口与外部电路连接的节点描述。电路图一般都是通过一些计算机辅助设计（Computer aided design，CAD）和电子设计自动化（Electronic design automation，EDA）的程序提供的图形化界面（Graphical User Interface，GUI）来表示。

标准单元除了有数字逻辑的表示，版图（layout）作为最接近实际产品的视图，是由基础层（layer）组成的，基础层对应晶体管的不同结构，布线层（wiring layers）和通过层（via layer）相互连接，然后连接晶体管端口形成的。

最后，布局和线路（PNR）工具可以用来组合的一切，合成的超大规模集成电路的布局，以自动化的方式，从更高层次设计的网表和平面图。

此外，还可以使用许多其他CAD工具来验证单元格视图和模型的其他方面。和其他文件可以创建支持各种工具，利用标准细胞过多的其他原因。所有支持标准单元变化而创建的所有文件被统称为标准单元库。

### 3.1.2 标准单元库

表3-1 标准单元库的分类

|  |  |  |
| --- | --- | --- |
| 标准单元库 | 组合逻辑单元 | Inverter反相器  NAND、NOR 与非、或非门  AND、OR 与、或逻辑  BUFFER 缓冲器  MUX 多路选择器  XOR、XNOR 异或、同或逻辑  AOI、OAI 与或非、或与非  ADD 加法器包括全加器、半加器  Clock Cell 包括Clock BUFFER、Clock Inverter、Clock NAND等 |
| 时序逻辑单元 | Flip-Flop 触发器  LATCH 锁存器 |
| 特殊类型单元 | FILL cells 填充单元  Tie High、Tie Low cells 电源连接单元  Delay cells 信号延迟单元  二极管单元  Tap cells 阱连接单元  FDCAP cells 去耦合单元 |
| 低功耗辅助设计单元 | Clock Gating 单元  Power Gating单元  Isolation Cell 隔离单元  Retention Cell 信号保持单元  Level Shift 电平转换单元  Always-on 单元  Well Bias Cell 阱电位偏置单元 |

一般地，标准单元库通常会包含以下四类单元：组合逻辑单元、时序逻辑单元、特殊类型单元和PMK（Power Management Kit）低功耗辅助设计逻辑单元。

除了不同工艺尺寸会有不同的标准单元库之外，相同工艺尺寸还会有不同的多套标准单元库用来供用户选择，它们在不同的电路密度和速度上有着相应的优势。例如，中芯国际（SMIC）深亚微米的工艺制程包括了三套标准单元库：一套高性能的超高速VHS（Very High Speed）单元库、一套密度和速度优化的高速HS（High Speed）单元库和一套高密度HD（High Density）单元库。

### 3.1.3 标准单元库流程

 传统标准单元库建立的流程如图：

图3-1 标准单元库建立流程

1. 制定设计方案：在设计设计28nm标准单元之前，我们首先要制定设计方案。主要包括：标准单元库的Track数、需要设计的单元库器件、MOSFET中PMOS与NMOS的宽度比、在版图中版面两单元间的中心距距离（Pitch）、单元库器件的驱动能力等。
2. 单元电路设计：合理地设计出所需要的全部标准单元的器件尺寸和电路结构，同时单元的时序和功能必须通过SPICE等仿真工具的相关验证。
3. 单元版图设计：绘制出所有库单元的版图，并使其满足面积最小，同时所有的电路单元保持高度一致。版图验证主要通过DRC（Design Rule Check）和LVS（Layout Versus Schematic）这两步的验证。其次，还需要根据Foundry所提供的Design Rule进行物理特性的验证。
4. 库模型建立：一套完整的标准单元库不仅需要包含电路图和版图，还需要包含各种符合国际标准认证的EDA（Electronics Design Automation）工具与可识别的库模型文件。后端设计中，需要使用各种EDA工具以及利用Foundry提供的文件，对标准单元库进行仿真和验证。通常，这些库文件主要描述了单元的时序信息、网表信息和版图信息三个方面。

表3-2 库模型文件分类

|  |  |  |
| --- | --- | --- |
| 时序信息文件 | .lib | 标准单元的Synopsis的时序模型，描述了标准单元的时间、功率特性 |
| .db | .lib的二进制格式文件 |
| .tlf | 标准单元的TT、FF、SS工艺角的时序库文件 |
| 版图信息文件 | .lef | 供布局布线使用的版图文件格式，具有工艺信息和各个单元的几何特性的文件，是生成milkyway库所必需的 |
| .gdsii | 包含版图信息的文件 |
| 网表信息文件 | .cdl | 标准单元网表，用于LVS检查 |
| .v | Verilog语言格式文件 |
| .vhd | VHDL语言格式文件 |
| 符号库文件 | .slib | 标准单元的符号库文件 |

1. 库验证：标准单元库文件必须通过Foundry可制造性的验证。与此同时还必须确保标准单元库模型的正确性，还需要确保应用库模型设计出来的电路在制造出来后时序、功耗、面积等都能满足要求。
2. Release：标准单元库在设计和验证之后，还需要撰写标准单元库的设计报告、数据手册、使用手册等相关的文档，以方便满足Foundry的使用，并将相关资料整理打包，最终发布。

## 3.2 时序

### 3.2.1 Liberty库

LIB时序库主要用于描述每个标准单元的传输延迟的时间参数、基本的逻辑功能等，其中传输延迟的时间参数是通过测试或仿真工作的提取获得。LIB库文件主要是由Synopsys公司开发的，专门用于对物理单元的时序以及功耗方面的信息进行描述的关键库文件。一般情况下，LIB文件由标准单元的具体信息和库的基本属性两个部分构成。

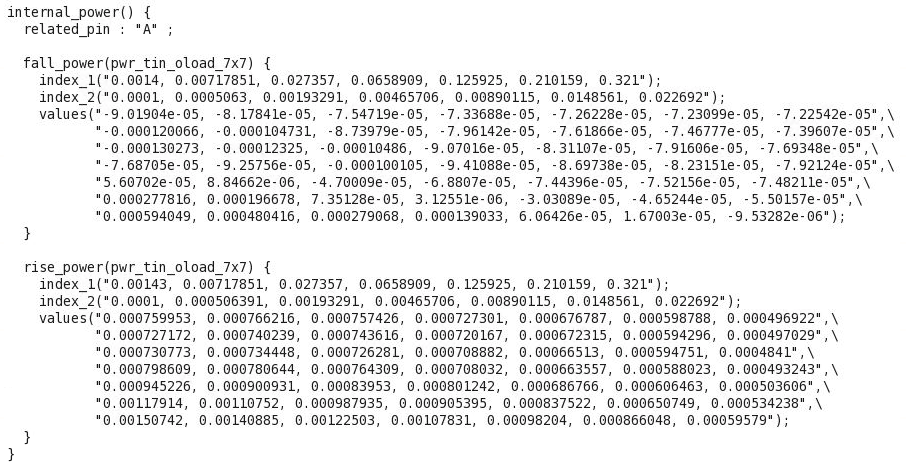
 基本属性主要描述了标准单元库的基本信息，主要包含PVT corner（Process、Voltage、Temperature）、文件的版本号、生成日期、电路属性、器件类型、以及时序功耗的查找表等。主要用于标准单元的功能参数库可以进行电路的Verilog/VHDL的数字仿真，从而确定所设计电路的功能正确与否，并检查异步时序电路中的竞争和冒险等工作。

图3-2 LIB文件中功耗数值查询表

LIB文件的主要部分是每个标准单元的功能和有关的延迟参数，主要包括亚阈值电流、输入输出负载以及时序功耗的数值查询表。

如图是UMC 28nm工艺的TT1.05V25C下的标准单元库的LIB文件的INV的功耗查询表。

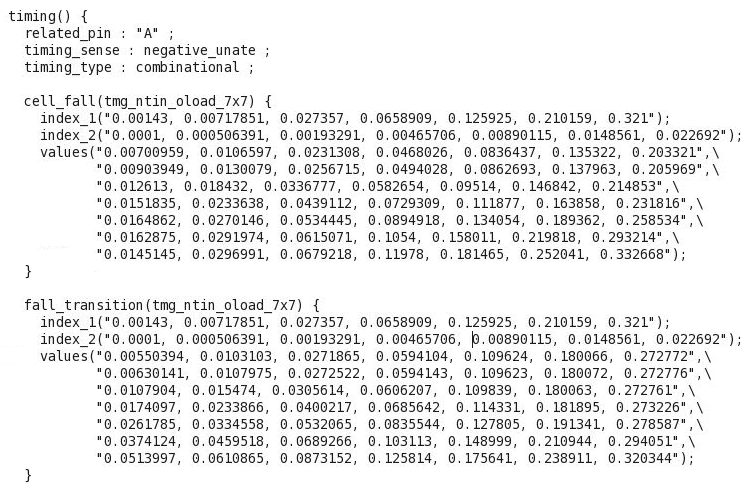
其中rise\_power表示管脚A从0变为1的内部功耗，但不包含开关功耗，开关功耗有电容乘以电压的平方单独运算；index\_1表示输入的传输时间；index\_2表示输出端电容；values就是当输入输出取不同值时，相对应的内部功耗值。其中横坐标是index\_1，纵坐标是index\_2。

图3-3 LIB文件中时序数值查询表

如图是UMC 28nm工艺的TT1.05V25C下的标准单元库的LIB文件的INV的时序查询表（部分）。

其中timing就是标准单元的时序描述；cell\_fall是下降时间；fall\_transition是传输时间；index\_1是输入的传输时间；index\_2是输出端的电容；values是7×7的时序查询表，横坐标是index\_1，纵坐标是index\_2。

LIB文件包含的时序信息是非常繁多的，详见公式3.1：

（3.1）

以UMC28nmHLP工艺的标准单元库而言，LIB数由以下元素组成。

表3-3 LIB文件数的影响因素详表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 阈值电压 | 沟道长度  （纳米） | 时序模型 | 工艺角 | 电压  （V） | 温度  （℃） |
| RVT  LVT  HVT | 30  35  40 | NLDM  CCS  ECSM | FF  FFG  TT  SS  SSG | 0.855  0.90  0.945  1.05  1.15 | -40  0  25  85  125 |

其中，时序模型分为三种：

1. NLDM：非线性延时电压源，适用于40nm以上，LIB文件较小；
2. CCS：复合电流源模型，由Synopsys公司开发，适用于40nm工艺以下，时序更为精准，LIB文件较大；
3. ECSM：有效电流源模型，有Cadence公司开发，适用于40nm工艺以下，时序更为精准，LIB文件中等。

由此可以看出，一套标准单元库会包含数量繁多的LIB文件，大约有1425个LIB文件。与此同时，单个LIB文件中具有整个标准单元库的所有电路的时序信息，当阈值电压为HVT，沟道长度是30nm时，就有939个标准单元电路。

由此可见，对于标准单元库而言，LIB文件的数量是非常多的。繁多的LIB文件数也就代表着仿真工作的繁多，也就意味着繁多的仿真时长和需求很大的机器资源。本文以标准单元为例，提出了通过机器学习算法对部分标准单元的时序进行回归，并对未知标准单元的时序进行预测的方法。

### 3.2.2 数据收集

由于制造工艺的偏差，同一晶圆的芯片在同一环境下都有不同的工作状态。以MOSFET晶体管为例，尽管他们也有类似的电阻和电容，会有不同的状态，如FNFP、SNSP、TNTP、FNSP、SNFP，不同的阈值电压等参数。除了制作工艺的偏差，MOSFET晶体管的性能也受到电压和温度的影响。一般来说，仿真的时候会选择不同的温度节点和工作电压节点。

表中有325套Process Corner。

表3-4 PVT的选择范围

|  |  |
| --- | --- |
| 工艺 | FNFP、SNSP、TNTP、FNSP、SNFP |
| 电压（V） | 0.84 0.875 0.91 0.945 0.98 1.015 1.05 1.085 1.12 1.155 1.19 1.225 1.26 |
| 温度（℃） | -40 -0 25 85 125 |

本课题选取了十个电路分别是：INV、AND2、AND3、NOR2B、NOR2、AOI211、XOR3、ADDF、DFFNQ、DFFNSRPQ。

本章仿真均使用基于Bsim4仿真模型的H-Spice仿真工具，Slew设置为1ns。

以D触发器为例，输入端是D，输出端是Q。路径延时（Path Delay）定义为从输入波形的50%到输出波形的90%或者从输入波形的50%到输出波形的10%，以确保输出波形的电势能稳定达到目标电位。

图3-5描述了setup time是时钟达到之前波形需要保持一段时间，hold time是时钟达到之后波形需要保持的一段时间。

以DFFs为例，setup time是从D端到节点m的传输路径时间，hold time有两个值：一个是节点m到节点s的传输路径时间，另一个是D端到节点nm的传输路径时间。Hold time选取二者的最大值。

图3-4 传输路径延时：从输入波形50%到输出波形10%的传输路径延时

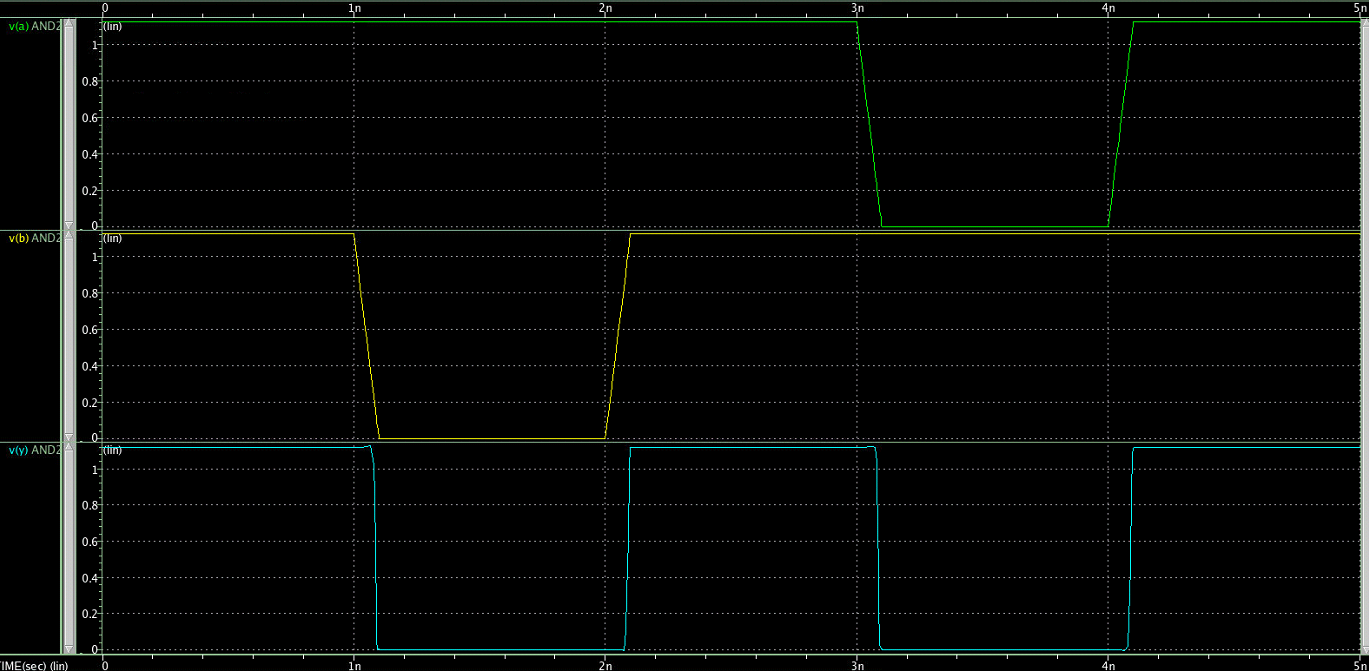
图3-5 D触发器的setup time和hold time

图3-6 D触发器的逻辑电路图

本课题中的10个标准单元电路分别有不同数量的路径延迟，此外D触发器还拥有建立时间和保持时间。

对于单输入电路而言，以INV为例，当输入端是上升沿时，输出端是下降沿，具有一个路径延时；当输入端是下降沿时，输出端是上升沿，还具有一个路径延时。

对于多输入电路而言，当某一输入端翻转，而其他输入端保持电平不变，且输出端也翻转的时候，才具有一个路径延时。

图3-7 AND2输入输出波形示意图

以AND2为例，时序包含四种时序：

1. 当输入A为1时，输入B从0翻转到1时，输出Y从0翻转到1。输入B的50%到输出端Y的90%的传输路径延时。
2. 当输入A为1时，输入B从1翻转到0时，输出Y从1翻转到0。输入B的50%到输出端Y的10%的传输路径延时。
3. 当输入B为1时，输入A从0翻转到1时，输出Y从0翻转到1。输入A的50%到输出端Y的90%的传输路径延时。
4. 当输入B为1时，输入A从1翻转到0时，输出Y从1翻转到0。输入A的50%到输出端Y的10%的传输路径延时。

下表是本课题中所选用的10个电路相对应的传输路径延时的数量：

表3-5 10个电路的传输路径延时

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| INV | AND2 | AND3 | NOR2B | NOR2 | AOI211 | XOR3 | ADDF | DFFNQ | DFFNSRPQ |
| 2 | 4 | 6 | 4 | 4 | 16 | 24 | 36 | 3 | 3 |

## 3.3 时序预测

本课题提出了两种方法实现预测标准电路的路径延时：电路参数方法和时序方法。

### 3.3.1 电路参数方法

作为集成电路的最小组成单元，金属氧化物半导体效应晶体管（MOSFET）的各种电路特性表现在电路的特性上。MOSFET根据载流子的极性不同，可以分为NMOSFET和PMOSFET。

MOSFET晶体管都是有漏极（Drain）、栅极（Gate）、源级（Source）与衬底（Body）。由于每个终端都影响着MOSFET的电路特性，且不同MOSFET的逻辑组合又具备不同的电路特性，所以本课题选择了MOSFET的一些电路参数作为最小二乘法回归学习的特征。

表3-6 电路参数表及其含义

|  |  |
| --- | --- |
| Ivin (A) | 信号上升或下降时VIN电压的电流的峰值 |
| Ivdd (A) | 信号上升或下降时VDD电压的电流的峰值 |
| Ivss (A) | 信号上升或下降时VSS 电压的电流的峰值 |
| Ion (A) | 信号上升或下降后MOS管的电流的平均数值 |
| Ipeak (A) | 信号上升或下降后MOS管的电流的峰值 |
| Vth (V) | 信号上升或下降时MOS管的阈值电压 |

以INV为例，共有325组仿真数据，每组数据有9个电路参数作为拟合的特征（Feature）和一个路径延时作为目标（Target）。表中所示是五组INV的特征和目标值。

表3-7 五组INV的电路参数与延时

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ivin1  (10-6A) | Ivdd  (10-6A) | Ivss  (10-6A) | Vth1  (V) | Vth2  (V) | Ion1  (10-9A) | Ion2  (10-9A) | Ipeak1  (10-5A) | Ipeak2  (10-6A) | Delay  (10-11s) |
| 5.868  4.419  7.076  6.802  5.461 | 1.345  1.020  9.112  1.614  1.422 | 4.226  3.226  11.97  5.830  3.938 | 0.5129  0.2629  0.2456  0.4511  0.5768 | 0.5063  0.3713  0.3713  0.5922  0.4257 | 4.357  1.663  5.097  2.508  15.85 | 0.9851  2.142  4.475  0.6069  4.057 | 1.650  1.064  2.466  1.996  1.358 | 4.074  3.672  13.06  6.239  3.789 | 3.146  2.527  1.859  1.456  4.058 |

本次学习使用10折交叉验证（10-Fold Cross Validation），每次学习使用32组仿真数据作为训练集，另外的293组仿真数据作为测试集。

本次学习选择了拟合优度作为拟合程度的度量，拟合优度（Goodness of Fit）是指回归模型对观测值的拟合程度。度量拟合优度的统计量是可决系数（determination coefficients）R2。可决系数的值越接近1，说明回归直线对观测值的拟合程度越好；反之，可决系数的值越小，说明回归直线对观测值的拟合程度越差。

（3.2）

可决系数的表达式如上，其中是回归平方，是总平方和。是表示因变量与所有自变量之间的总体关系。拟合优度的可决系数是无量纲系数，有确定的取值范围（0-1），便于对不同资料回归模型的拟合优度进行比较。

表3-8 十个标准电路延时训练集与测试集的拟合优度

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| R2 | INV | AND2 | AND3 | NOR2B | NOR2 | AOI211 | XOR3 | ADDF | DFFNQ | DFFNSRPQ |
| Train | 0.968 | 0.973 | 0.968 | 0.982 | 0.989 | 0.960 | 0.989 | 0.995 | 0.996 | 0.971 |
| Test | 0.962 | 0.930 | 0.952 | 0.973 | 0.983 | 0.924 | 0.962 | 0.962 | 0.965 | 0.940 |

本次学习还选择了准确度（Accuracy）作为预测时序准确度的评价标准。准确度和相对误差一般由公式表示：

（3.3）

（3.4）

其中，是准确度，是实际相对误差，是绝对误差，L为真实值。相对误差一般表示为百分数。相对误差越接近0，表示预测的时序的准确度越好，预测值与真实值越接近。同样的，准确度越接近1，表示预测的时序的准确度越好，预测值与真实值越接近。

表3-9 十个标准电路延时训练集与测试集的预测准确度

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| δ | INV | AND2 | AND3 | NOR2B | NOR2 | AOI211 | XOR3 | ADDF | DFFNQ | DFFNSRPQ |
| Train | 0.836 | 0.941 | 0.968 | 0.985 | 0.935 | 0.887 | 0.940 | 0.991 | 0.993 | 0.982 |
| Test | 0.859 | 0.796 | 0.952 | 0.958 | 0.771 | 0.779 | 0.754 | 0.954 | 0.955 | 0.958 |

电路参数方法可以取得很好的预测效果，与此同时，所有的电路参数中，Ivdd、Ivss在电路中只有一个；Ivin的数量对应输入端的数量；Vth、Ion、Ipeak每一个MOS管都有。对于小电路，电路参数不是很多，然而，对于DFF这种复杂一些的电路，则有三十多个MOS管，电路参数方法会有大概一百个电路参数作为学习的特征。

电路参数方法对于标准单元仿真周期，我们采用时间成本和机器资源成本来进行评价。

以UMC28nmHLP为例，通常以50个核来进行仿真，大致需要24小时。采用本方法后，以50个核来进行仿真数据的采集以及时序数据预测，大致需要17小时。

### 3.3.2 时序方法

从Process corner来看，一般而言，时序都随着Process、Voltage、Temperature的变化呈现相对应的变化。

以DFF为例，本方法以其他标准单元电路的时序作为学习的特征，DFF的时序作为目标。其中其他标准单元电路分别是INV、AND2、AND3、NOR2B、NOR2、AOI211、XOR3和ADDF。如图是5个process corner下的DFF的时序，分别是传输路径延时、setup time和hold time。

本学习也是用10折交叉验证，表列出了DFF的时序拟合优度。表列出了DFF的准确度。可决系数的值越接近1，说明回归直线对观测值的拟合程度越好，同样的，准确度越接近1，表示预测的时序的准确度越好，预测值与真实值越接近。

表3-10 DFFs的传输路径延时、setup time和hold time在5个corner下时序

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Process corner | DFFNQ | | | DFFNSRPQ | | |
| Path delay  [10-11s] | Setup time  [10-11s] | Hold time  [10-11s] | Path delay  [10-11s] | Setup time  [10-11s] | Hold time  [10-11s] |
| tt1p05v0c  ff1p26v125c  ss0p84vn40c  sf1p05v25c  fs1p05v85c | 7.766  4.378  72.50  7.999  7.915 | 8.880  4.760  65.69  9.969  7.918 | 5.857  3.634  49.26  5.928  6.556 | 9.717  5.059  95.01  9.793  10.05 | 11.92  6.276  92.43  12.46  12.24 | 8.879  5.396  72.91  8.312  10.67 |

表3-11 DFFs的传输路径延时、setup time和hold time的拟合优度

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| R2 | DFFNQ | | | DFFNSRPQ | | |
| Path delay | Setup time | Hold time | Path delay | Setup time | Hold time |
| Training set | 0.9991 | 0.9996 | 0.9989 | 0.9990 | 0.9996 | 0.9990 |
| Testing set | 0.9984 | 0.9987 | 0.9980 | 0.9979 | 0.9992 | 0.9983 |

表3-12 DFFs的传输路径延时、setup time和hold time的预测准确度

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| δ | DFFNQ | | | DFFNSRPQ | | |
| Path delay | Setup time | Hold time | Path delay | Setup time | Hold time |
| Training set | 0.9923 | 0.9907 | 0.9772 | 0.9981 | 0.9946 | 0.9763 |
| Testing set | 0.9912 | 0.9821 | 0.9874 | 0.9901 | 0.9846 | 0.9854 |

从上表可以看出，时序方法可以取得相当好的预测效果。

与电路参数方法相比较，时序方法所需要的特征数目较少，只需要其他电路的时序即可。不需要每个电路的每个MOS管的电路参数。

时序方法对于标准单元仿真周期，我们采用时间成本和机器资源成本来进行评价。

以UMC28nmHLP为例，通常以50个核来进行仿真，大致需要24小时。采用本方法后，以50个核来进行仿真数据的采集以及时序数据预测，大致需要12小时。

## 3.4 本章小结

本章详细阐述了标准单元方法和标准单元库流程，着重讲述了时序库LIB文件，分析了标准单元库仿真的周期时长。并通过电路参数方法和时序方法两种方法，使用最小二乘回归算法预测了标准单元电路的时序，其中电路参数方法的预测时序的准确度能达到94.58%和87.36%；时序方法的预测时序的准确度能达到98.82%和98.68%。并从时间成本和机器资源成本对两种方法进行了评估，电路参数方法节约时间大致有30%，时序方法节约时间大致有50%。

# 第四章 SRAM编译器

本章以TSMC 28nm SRAM为例，选择六管单元作为设计方案，介绍了SRAM编译器，然后详细分析SRAM编译器的表征（Characterization），提出预测三种仿真数据的方法，最后对预测的结果和预测方法进行了评价。

## 4.1 SRAM基本原理

### 4.1.1 SRAM架构

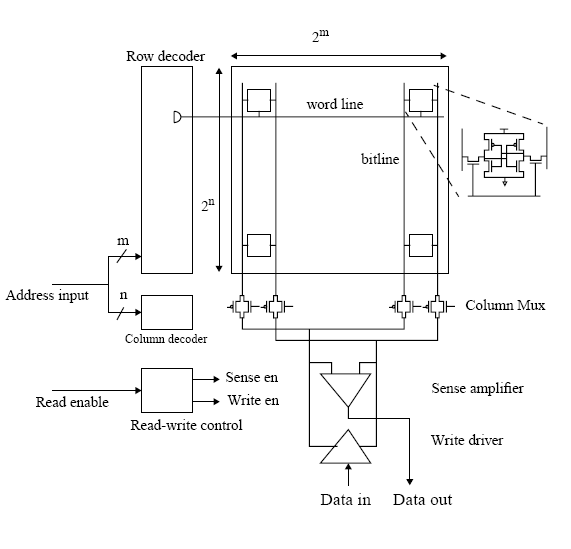
TSMC 28nm SRAM架构如图4-1。

图4-1 SRAM电路框架图

SRAM即静态随机存取存储器，SRAM主要由存储单元阵列（Core Cell Array）、地址译码器（Decoder）、灵敏放大器（Sense Amplifier）、控制电路（Control circuit）和缓冲/驱动电路（FFIO）。

如上图4-1，存储单元阵列有2m行和2n列，分别表示存储阵列有2m根字线（Word line）构成，每个字深存储2n根位线（Bit line）。每个SRAM包含2m×2n个存储单元。每个存储单元与一根字线和一组反相的位线相连接，并通过它们进行对存储单元的读写操作。

译码器分为行译码器（Row decoder）和列译码器（Column decoder）。行译码器接收地址信号，选中一根字线触发阵列的一行，在进行读操作的时候，字线驱动位线，反相的位线通过灵敏放大器读出存储单元的数据。

通常为了节省功耗和减少数据读取时间，存储器中的字数会远远多于每个字存储的位数，这样会使得存储器变得长而窄，因为位线过长会增加数据访问的速度。

### 4.1.2 SRAM存储单元

图4-2 SRAM 6管单元原理图

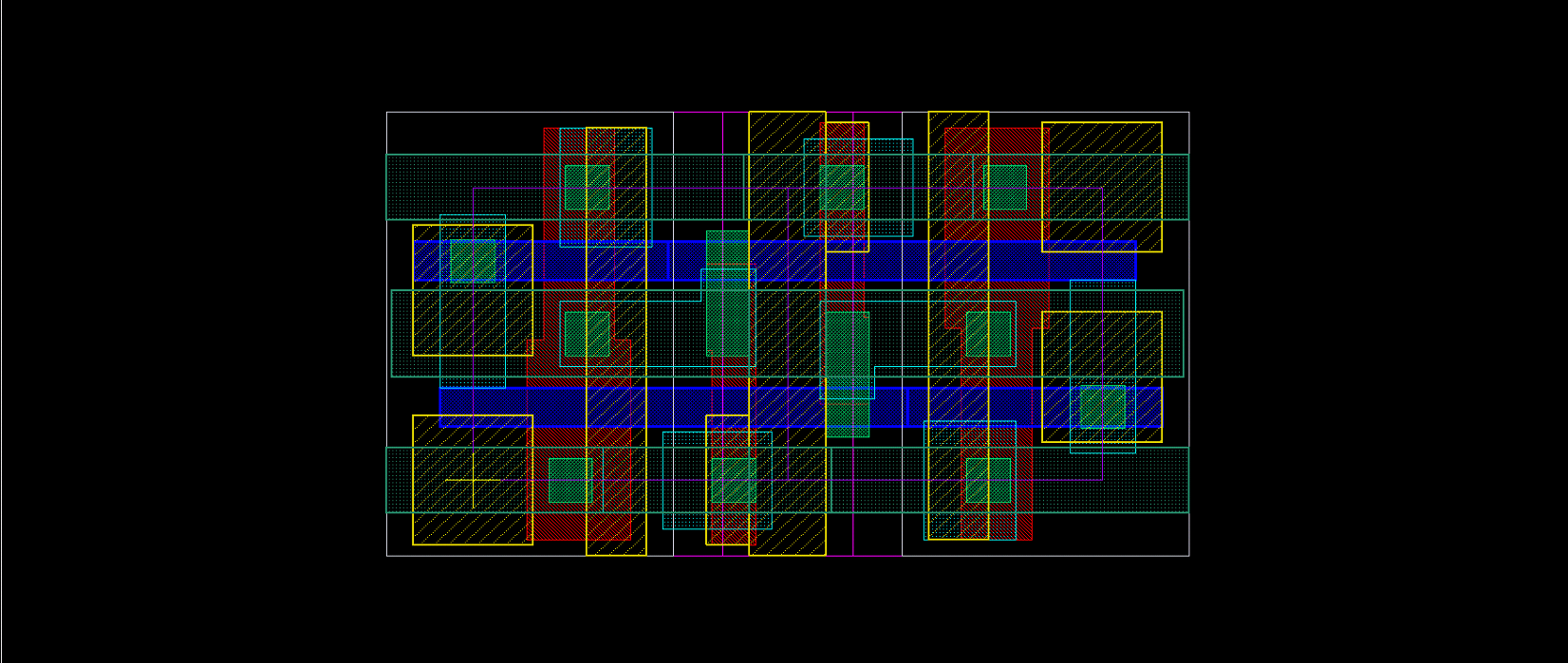
图中所示为6管SRAM单元，存储单元使用一根字线（Word Line）、两根反相的位线（bit和bit\_b）。单元中包括了一对交叉耦合的反相器，并且每根位线链接了一个存取晶体管。一对互补的数据存储在交叉耦合的反相器上。如果数据受到轻微的干扰，由回路构成的正反馈将使数据恢复到VDD或GND。选中这个单元的字线就可以对这个单元进行读出或写入。

图4-3 TSMC 28nm 6管单元的版图

如图4-3为TSMC 28nm SRAM Compiler的存储单元的版图。

### 4.1.2 SRAM Compiler

一般而言，Memory Compiler属于IPs的一种，目前部分Design House和Foundry都可以向客户提供配套的Memory Compiler。Memory Compiler是设计一套完整的存储器单元模块（SRAM、Flash等）。设计中包含不同大小的存储器单元模块，尽量做到单元模块的面积和速度最优。用户可以通过简单的界面定义实现目标要求的存储器单元模块的GDSII文件和网表文件等。由于用户不能更改电路的内部结构和版图，所以面积和速度就是Memory Compiler的竞争力所在。

常用SRAMs是全定制设计，全定制设计以存储器密度、速度、功耗和良率为主要设计目标，全定制设计需要相应较长的设计周期。能在较短时间内完成满足设计需求的SRAM，就意味着在高度竞争的市场中具有强有力的竞争力。SRAM Compiler就是一种解决方法，SRAM Compiler不仅能节约成本，而且能在很大程度的缩短设计周期。

SRAM Compiler是一种能生成不同存储器配置的计算机智能化程序。SRAM Compiler主要功能包括：产生存储器网表（netlist）、模拟最坏情况下读写路径以产生时序信息、为在原理图中取代SRAM而建立相应的符号、建立带有时序信息的模拟模型、产生对应SRAM的物理版图、建立电源栅结构等。通常SRAM Compiler用来为诸如门阵列或标准单元电路的专用集成电路（ASICs）生成存储器。

#### 4.1.2.1 SRAM Compiler设计流程

图4-4 SRAM Compiler设计流程图

SRAM Compiler通过调用设计电路结构中的子模块的版图来产生存储器模块版图，所以不同的电路设计结构得到的版图会对SRAM Compiler的面积有很大的影响。

如图4-4是SRAM Compiler的设计流程，发开一套SRAM Compiler的主要流程如下：根据客户的需求以及市场信息，提出对SRAM的设计需求，主要是SRAM所具备的一些功能和特征，并对存储器单元进行评估，并检查是否满足存储器设计的速度和面积要求；然后对SPEC的配置进行电路图设计及预仿真，其次是页元（Leaf cell）的版图设计，通过DRC和LVS规则对电路图和版图进行优化调整；使用SRAM Compiler的Tiling Engine来产生存储器模板（Memory Instance）版图，最后是对不同配置的Instance进行表征（Characterization），主要通过时序（Timing）、功耗（Power）和电容（Capacitance）来体现。

在Memory Characterization环节中，主要是仿真工作和仿真数据收集。实际设计中，仿真工作的时间几乎占据了全部的Memory Characterization环节。因为仿真工作的数量由SPEC配置和PVT共同决定。而SPEC与PVT参数包含了特征、MUX、Bank、Word、Bit、Input Slew、Output loading、Process Corner、Voltage以及Temperature等。参数的繁多决定了仿真工作的数量的繁多。

#### 4.1.2.2 SRAM Compiler整合流程

图4-5 SRAM Compiler整合流程图

1. Leaf cell的版图和电路在通过DRC和LVS之后，通过Tiling Engine的脚本生成不同instance以及instance的版图文件（LEF、GDSII文件）和电路描述文件（Verilog语言文件、SPICE文件）；
2. Characterization环节通过设计者给出的Instance列表对instance抽象成CP（Critical Path），通过Word、Bit，Feature的变化来改变CP中的各个参数，模拟对应Instance的行为，节省了仿真时间，同时还不是准确性。针对不同instance的SPEC配置文件的变量参数进行配置，通过脚本自动生成仿真文件，并进行仿真；
3. 仿真完成后，通过测量描述语言对仿真结果文件进行数据抽取，生成Rawdata文件，并通过计算文件生成最后的LIB文件。
4. 最后将LIB文件、LEF文件、GDSII文件、Verilog文件和SPICE文件整合，编写Complier实现程序，使得用户可以通过自定义instance的大小与功能自动生成SRAM及其网表文件、版图文件与表征数据库等。

通过Compiler工具中的各种引擎模块调用设计中相应的设计输入文件，如SPEC配置文件、Compiler的物理单元库（GDS），网表单元库（Netlist）、Memory Development Language（MDL）语言脚本等文件，具体需要的文件见图；Memory Compiler的输出文件结构如图，大致有GDSII、LEF、SPICE和Verilog文件等。

图4-6 Memory Compiler输入输出文件结构

## 4.2 存储器表征

Memory Characterization是SRAM Compiler设计中非常重要的环节，主要是instance的仿真工作以及仿真数据的收集。其中仿真数据主要是时序和功耗。

Characterization的仿真工作具有一定的重复性，可以通过脚本的实现来生成Test bench，进行仿真、数据检查以及数据收集。Characterization减少了设计者的仿真时间，同时也可以减少一些人为的错误。

仿真的工作量主要由SPEC配置和PVT共同决定，SPEC配置有FEATURE、MUX、BANK、WORD、BIT、INPUT SLEW和OUTPUT LOADING等。

### 4.2.1 特征

本课题SRAM Compiler所采用的工艺是TSMC28nm的HPC+工艺版本，本课题SRAM Compiler所采用的功能如下：

1. Synchronous read and write operations
2. Column Mux options for the best aspect ratio
3. Fully customized layout density available in 0.9V ±10%
4. Three power management modes: Light Sleep, Deep Sleep and Shut Down to reduce static power
5. Bit write and global write operations available
6. Extra margin control option
7. Special test modes externally bypass read and write self-timing circuits
8. Redundancy inside to repair defective cell

表4-1 TSMC28nm SRAM Compiler具备的特征

|  |  |
| --- | --- |
| 1 | 具有deep sleep模式 |
| 2 | 具有shut down模式 |
| P | 具有light sleep功能 |
| B | 具有Bit write功能 |
| I | 具有IO redundancy功能 |
| M | 具有BIST功能 |
| E | 具有HVT Periphery功能 |
| L | 具有LVT Periphery功能 |

用户可以通过定义不同的word、bit和mux来优化存储器，从而实现面积、速度和功耗的最佳平衡。以上的特征都被集成到Memory Compiler中，用来实现系统应用和用户需求。在设定一些参数之后，Compiler就可以在很短的时间内生成memory instance的data sheet、GDSII Layout、LVS netlist和PR model等。

通常，Memory Compiler所提供的全部功能，用户在自定义生成SRAM的时候，根据需求选择其中几个或者全部的功能。这就要求在进行仿真的时候，针对不同的特征组合进行不同的仿真，这也是仿真工作量繁多的重要原因。

以下为需要进行仿真的特征组合（字母组合）：

无特征：-

一个特征：1, 2, b, i, m, p

两个特征:12, 1b, 1i, 1m, 1p, 2b, 2i, 2m, 2p, bi, bm, bp, im, ip, mp, 12b, 12i, 12m, 12p, 1bi, 1bm, 1bp, 1im, 1ip, 1mp

三个特征：2bi, 2bm, 2bp, 2im, 2ip, 2mp, bim, bip, bmp, imp, 12bi, 12bm, 12bp, 12im, 12ip, 12mp, 1bim, 1bip

四个特征：1bmp, 1imp, 2bim, 2bip, 2bmp, 2imp, bimp

五个特征：12bim, 12bip, 12bmp, 12imp, 1bimp, 2bimp

六个特征：12bimp

共计64组特征。针对每一组特征都有相应的仿真工作需要完成。

### 4.2.2 实例

用户可以通过定义不同的word、bit、mux和bank来定义需求的存储器。

通常随着存储阵列容量的变大，那么字线和位线就会变得更长。增加字线和位线的长度会使电容变大，从而导致延时和功耗的增加。因此，通常会将大容量的存储阵列分割成为多个较小的存储阵列，这样的较小的存储阵列就被称为体（Bank）。

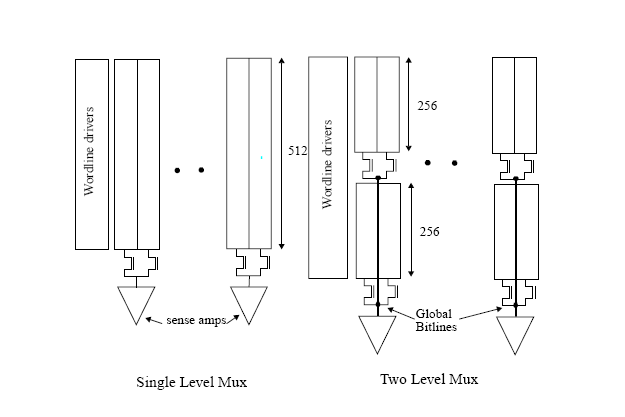
通常为了节省功耗和减少数据读取时间，存储阵列中的字数会远远多于每个字存储的位数，这样会使得存储器变得长而窄，因为位线过长会增加数据访问的速度。

图4-7 Mux示意图

通常采用的另一个做法就是减少位线的长度。如图将一根正常的位线分割成两部分，每根位线的长度与其所驱动的单元数都减半，从而减少了位线的负载。这样子分割位线的做法叫做Column Mux。其中每根位线的长度被分成几份，Mux就等于几，且Mux总是2的幂值。在具体的SRAM Compiler设计中，会提供很多不同的字线、位线和Mux来供用户进行选择。

本课题的SRAM Compiler是小面积的存储器，设计中不分Bank，因此Bank的值为1。

Mux根据SPEC文件定义为2和4。当Mux取2时，Word最小取32，最大取512，步长为4，Bit最小取8，最大取144，步长为2；当Mux取4时，Word最小取64，最大取1024，步长为8，Bit最小取8，最大取72，步长为1。

表4-2 字长根据Mux不同的选择范围

|  |  |
| --- | --- |
| Word Range | |
| Column Mux=2 | 32,36,40,…508,512 |
| Column Mux=4 | 64,72,80,…1016,1024 |

表4-3 位线根据Mux不同的选择范围

|  |  |
| --- | --- |
| Bit Range | |
| Column Mux=2 | 8,10,12,…142,144 |
| Column Mux=4 | 8,9,10,…71,72 |

当Mux选定2或者4的时候，选择不同的Word和Bit的值即可确定Memory Instance的大小。从上表可以看出，当Mux=2的时候，Word可取120个值，Bit可以取140个值；当Mux=4的时候，Word可以取120个值，Bit可以取64个值。由此可计算出本课题的SRAM Compiler的Instance个数共有24480个。理论上，用户自定义生成的Memory Instance的可以是24480中的任何一种，这就要求每一个Memory Instance都有相应的仿真数据，供用户查询。仿真阶段完成所有的Instance的仿真是不可能实现的，通常设计公司会提供比较典型的几个Instance的仿真数据。而客户需求所提供的Instance之外的仿真数据的时候，就需要设计公司重新进行仿真。

### 4.2.3 输入压摆和输出负载

在IO部分中，电路的端口主要有A、CK和DO。输入端口和时钟端口的slew的不同对电路的表征有不同的影响，在输出端口的负载的大小对电路的表征也有不同的影响。

在SPEC文件中，input slew和output loading都设置七个值。其中A作为数据输入的端口，采用input slew的值，称之为Data slew，CK作为时钟，也采用input slew的值，称之为CK slew，DO作为输出端，选有output loading的值，称之为load。

本课题中，input slew和output loading的值如下表：

表4-4：input slew和output loading数值表

|  |  |  |
| --- | --- | --- |
| Input slew | Data slew | 0.004，0.050，0.100，0.200，0.500，0.750，1.000 |
| CK slew | 0.004，0.050，0.100，0.200，0.500，0.750，1.000 |
| Output loading | Load | 0.001，0.020，0.050，0.100，0.250，0.300，0.350 |

对于instance而言，Data slew、CK slew和Load选择不同的数值的时候，都有不同的电路表征。也就是说对于某个确定的Instance而言，在IO中就有343种状态。

### 4.2.4 工艺角

在不同批次的晶圆和同批次的不同晶圆之间，MOSFETs的电路性能的变化都很大。为了规范电路设计，需要从晶片中提取相对应的电路器件模型，比如：片上NMOS和PMOS的测试会显示出不同的延时。在确保有效的工作状态下，对各种工艺角和各种温度条件下的电路进行仿真。工艺工程师们会在保证器件在电路性能处于在某个范围内，对有效芯片进行严格控制预期的参数变化。最终从仿真结果来对工艺角的选取制定标准。并根据这一标准对电路设计进行规范。这一做法可以在很大程度上减轻电路设计的困难。

通常提供给设计师的性能范围只适用于数字电路并以工艺角（Process Corner）的形式给出。通常表示为工艺、电压和温度的连续组合形式，比如TT0P9V125C。

图4-8 PVT Corners的示意图

图4-8表示了PVT Coners的思想：把有效的NMOS和PMOS晶体管的速度范围限制在由四个角所确定的矩形内。这四个角分别是：快速NMOSFET和快速PMOSFET（FNFP），慢速NMOSFET和慢速PMOSFET（SNSP），快速NMOSFET和慢速PMOSFET（FNSP），慢速NMOSFET和快速PMOSFET（SNFP）。比如具有较薄的栅氧和较低阈值电压的晶体管在图中就落在快速角的附近。

从载流子迁移率（Carrier Mobility）的角度来看，载流子迁移率指的是载流子在单位电场的作用下漂移的平均速度。工艺角的命名方式是两个字母组合，前一个字母表示NMOS的工艺角，后一个字母表示PMOS的工艺角。工艺角的类型通常有：快速（Fast）、慢速（Slow）和正常（Typical）。以TT为例，就是指正常速度的NMOS管和正常速度的PMOS管。

通常的，单一器件的载流子迁移率分布呈现正态分布，均值为TT，最小极值为SS，最大极值为FF。一般而言，在NMOS和PMOS管的迁移率分布范围中，TT、SS、FF这三个工艺角能覆盖大约±3 Sigma，即约99.73% 的范围。

然而，不同的工艺偏差也有很多情况。比如：掺杂浓度、制造时的温度控制、刻蚀程度等情况。再考虑到同一个晶圆上不同区域的非一致性，以及不同晶圆之间不同物理环境的因素。这些随机性事件的发生，只能使用统计学的方法才能够对工艺偏差进行有效的评估，才能覆盖到合理的范围。

本课题采用3 Corner Model，它们是TNTP，SNSP和FNFP。

设计除了要满足上述的Process Corner外，还需要满足电压与温度等条件, 形成的组合称为PVT (process, voltage, temperature)。

表4-5：PVT配置表

|  |  |
| --- | --- |
| Process | TT、SS、FF |
| Voltage/V | 0.72、0.81、0.9、0.99、1.08 |
| Temperature/℃ | -40、25、125 |

如表，电压设定数值分别是中位电压的80%、90%、100%、110%和120%。设计时还需要找到最好最坏情况，时序分析中将最好的条件(Best Case)定义为速度最快的情况, 而最坏的条件(Worst Case)则相反。

根据不同的需要，会有不同的PVT组合，以下是一些标准的分析条件：

1. WCS（Worst Case Slow）：slow process、high temperature、lowest voltage
2. TYP（Typical）：typical process、nominal temperature、nominal voltage
3. WCL（Worst Case Cold）：slow process、lowest temperature、lowest voltage
4. BCF（Best Case Fast）：fast process、lowest temperature、high voltage
5. ML（Maximal Leakage）：fast process、high temperature、high voltage
6. TL（Typical Leakage）：typical process、high temperature、nominal voltage

前四个的PVT组合主要用于时序分析，后两者PVT组合通常用于功耗分析。与前面说设计师通常只会提供一部分Instance的仿真数据一样，设计师通常也只会提供小部分PVT的仿真数据。当用户需求超出了最初的设计范围时，需要重新进行仿真工作。

## 4.3 本章小结

本章主要介绍了SRAM的基本框架和基本电路结构，然后描述了SRAM Compiler的设计流程与主要文件类型；着重对Compiler的表征环节进行了阐述，主要分析了Instance生成所受的因素影响。并详细对特征、word、bit、mux、bank、input slew、output loading和PVT corners进行详细解释，间接地表达了memory instance数量与仿真工作量的繁多。

# 第五章 SRAM表征数据预测

## 5.1 时序

### 5.1.1 时序数据

#### 5.1.1.1 读操作时序

SRAM有三种工作状态：保持（Standby）、写入（Write）和读出（Read）。

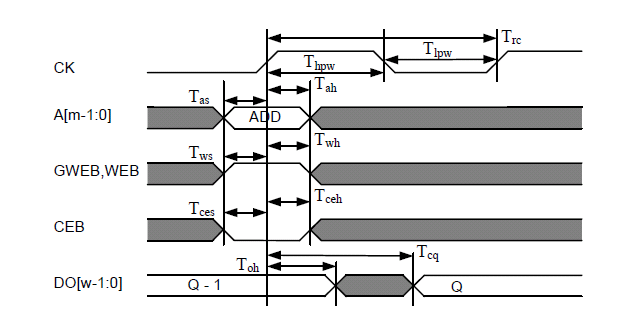
SRAM的数据保持时，芯片使能信号CEB处于无效状态，无论其他信号是什么状态，都不会对存储单元进行任何操作。

图5-1：读操作波形示意图

SRAM读操作时序如图，其中WEB为写使能，CK是时钟信号，CEB为片选中信号，A为地址信号，DO为数据输出信号。

表5-1 读操作时序

|  |  |
| --- | --- |
| **Parameter** | **Symbol** |
| Read cycle time | Trc |
| Data access time at the CK rising edge | Tcq |
| Output data hold time after the CK rising edge | Toh |
| Address setup time before the CK rising edge | Tas |
| Address hold time after the CK rising edge | Tah |
| GWEB,WEB setup time after the CK rising edge | Tws |
| GWEB,WEB hold time after the CK rising edge | Twh |
| CEB setup time before the CK rising edge | Tces |
| CEB setup time after the CK rising edge | Tceh |
| Clock high/low pulse widths | Thpw/Tlpw |

SRAM读取数据的时候，芯片使能信号CEB处于低电平（有效状态），写使能信号处于高电平（无效状态），地址信号A在时钟信号CK的上升沿到来的前后都保持一段时间，输出信号DO在延时之后会得到相应的数据读出。其中在时钟信号上升沿到来的前后都需保持一段时间的信号分别有地址信号A、写操作使能信号和芯片使能信号CEB。这三个信号在时钟信号上升沿到来之前保持的时间就是建立时间（Setup time），比如：Tas、Tws和Tces。相应的在时钟信号上升沿到来之后信号保持的时间就是保持时间（Hold time），比如Tah、Twh和Tceh。除此之外时钟信号CK处于高电平的时间为Thpw，处于低电平的时间为Tlpw，时钟信号的周期为Trc。输出信号DO从时钟信号上升沿到输出信号开始翻转的时间为Toh，而从时钟信号上升沿到输出信号翻转结束的时间为Tcq。其中Tcq越小说明SRAM工作越快。

对于SRAM的延时Tcq，它主要分布在三个方面：首先是译码延时，其次是存储阵列上的延时，另外还有时钟电路、放大电路、输出驱动等延时。

（5.1）

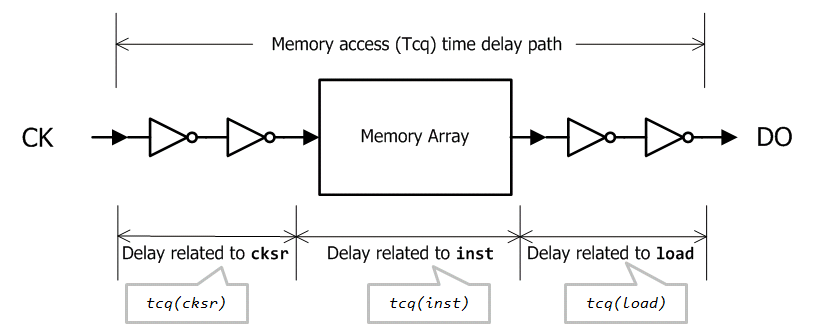
 由公式可以看出，Tcq主要有三部分组成：CKsr的延时、阵列的延时和读出路径的延时，其中cksr是Clock slew rate，load是output loading。

图5-2 Tcq组合示意图

其中tcq(cksr)只取最大instance和最小loading时的值；tcq(inst)只取最小cksr和最小loading时的值，tcq（load）取最小cksr和最大inst的值。

因为CK Slew和output loading都有7个数值，所以每一个instance的每一个时序都有7×7的查询表。

#### 5.1.1.2 写操作时序

图5-3 写操作波形示意图

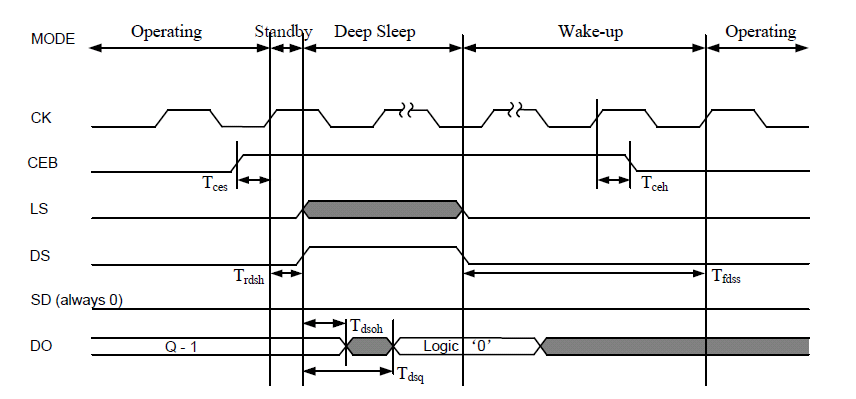
表5-2 写操作时序表

|  |  |
| --- | --- |
| **Parameter** | **Symbol** |
| Write cycle time | Twc |
| Address setup time before the CK rising edge | Tas |
| Address hold time after the CK rising edge | Tah |
| GWEB,WEB setup time after the CK rising edge | Tws |
| GWEB,WEB hold time after the CK rising edge | Twh |
| Input data setup time before the CK rising edge | Tds |
| Input data hold time after the CK rising edge | Tdh |
| CEB setup time before the CK rising edge | Tces |
| CEB setup time after the CK rising edge | Tceh |
| Output data invalid after the CK rising edge | Twdx |
| Output data valid after the CK rising edge | Twdv |
| Clock high/low pulse widths | Thpw/Tlpw |

SRAM写操作时序如图，其中WEB为写使能，CK是时钟信号，CEB为片选中信号，A为地址信号，DI为数据输入信号，DO为数据输出信号。

SRAM写数据的时候，芯片使能信号CEB处于低电平（有效状态），写使能信号处于低电平（有效状态），地址信号A在时钟信号CK的上升沿到来的前后都保持一段时间，输出信号DO在延时之后会得到相应的数据读出。其中在时钟信号上升沿到来的前后都需保持一段时间的信号分别有地址信号A、写操作使能信号、数据输入信号DI和芯片使能信号CEB。这四个信号在时钟信号上升沿到来之前保持的时间就是建立时间（Setup time），比如：Tas、Tws、Tds和Tces。相应的在时钟信号上升沿到来之后信号保持的时间就是保持时间（Hold time），比如Tah、Twh、Tdh和Tceh。除此之外时钟信号CK处于高电平的时间为Thpw，处于低电平的时间为Tlpw，时钟信号的周期为Twc。输出信号DO从时钟信号上升沿到输出信号开始翻转的时间为Twdx，而从时钟信号上升沿到输出信号翻转结束的时间为Twdv。

#### 5.1.1.3 电源模式时序

图5-4 Light sleep模式下电路各端口的波形示意图

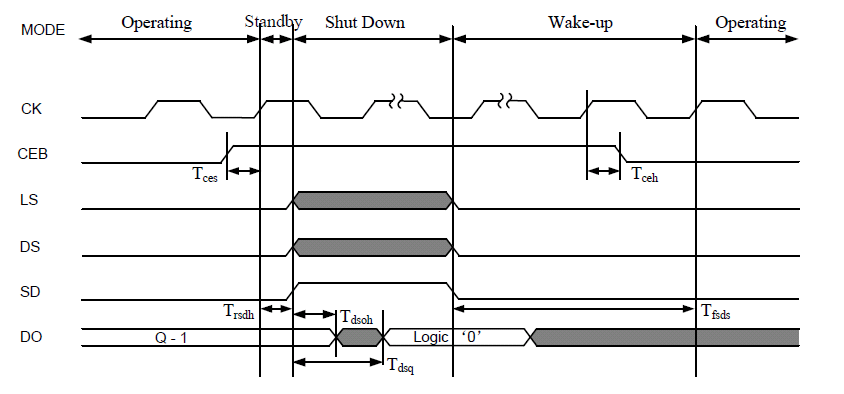
图5-5 Deep sleep模式下电路各端口的波形示意图

图5-6 Shut down模式下电路各端口的波形示意图

表5-3电源模式时序表

|  |  |
| --- | --- |
| **Parameter** | **Symbol** |
| LS fall setup time before CK rises | Tflss |
| LS rise setup time before CK rises | Trlsh |
| DS fall setup time before CK rises | Tfdss |
| DS rise setup time before CK rises | Trdsh |
| DS rise to output hold previous data time | Tdsoh |
| DS rise to all output fall to logic ‘0’ delay | Tdsq |
| SD fall setup time before CK rises | Tfsds |
| SD rise setup time before CK rises | Trsdh |
| SD rise to output hold previous data times | Tsdoh |
| SD rise to all output fall to logic ‘0’ delay | Tsdq |

在Light sleep模式，SRAM可以在不到一个完整的时钟周期之内恢复（wake up），但其静态功耗也是三种模式中最大的；Shut down模式具有最少的静态功耗，但其恢复时间是三种模式中最长的；Deep sleep模式的静态功耗和恢复时间都是介于两者之间。

以W32B8M2B1为例，当ck slew = 0.004，load = 0.001，data slew = 0.004时，Tcq在45个corners下的值。

图5-7 全corners组合下Tcq的时序柱状图

本课题分别从读操作时序、写操作时序和电源模式中选择20个时序，分别是Tcq、Toh、Troslp、Tfoslp、Tsdq、Tsdoh、Tdsq、Tdsoh、Tas、Tah、Tds、Tdh、Tceh、Tces、Tws、Twh、Thpw、Tlpw、Trc、Twc。

### 5.1.2 时序预测

#### 5.1.2.1 相同实例预测

本方法可以实现通过某一确定Instance的CK Slew、DATA Slew和LOAD的时序预测其他所有的CKSR、DATASR和LOAD组合的时序。

本方法使用CK slew=0.004，Data slew=0.004，Load=0.001（组合1）的时序，去预测了其他不同CK slew、Data slew和Load组合的时序。目标组合一共有7组，分别是：

组合2：CK slew=0.004，load=0.001，data slew=0.500；

组合3：CK slew=0.004，load=0.500，data slew=0.004；

组合4：CK slew=0.004，load=0.500，data slew=0.500；

组合5：CK slew=0.500，load=0.001，data slew=0.004；

组合6：CK slew=0.500，load=0.001，data slew=0.500；

组合7：CK slew=0.500，load=0.500，data slew=0.004；

组合8：CK slew=0.500，load=0.500，data slew=0.500。

本次预测采用8组corners作为训练集：FG1P08VN40C、FG0P9V25C、FG1P08V25C、TT0P9VN40C、TT0P9V125C、SG0P81VN40C、SG0P99VN40C和SG0P9V125C。其他的37组Corners作为测试集。

以CK slew=0.500，load=0.500，data slew=0.500为目标组合，并对数据集进行最小二乘回归拟合，如下式5.2和5.3：

（5.2）

（5.3）

其中是训练集的特征：组合1的8 corners×20 timing的数据集， 是训练集的目标：组合8的8corners×20timing的数据集。通过上述公式求得。并通过式5.2得到的回归模型，对测试集数据进行预测，并对测试集的目标数据与预测数据进行对比，其测试集的预测时序与仿真真实时序的对比图如图5-8。

图5-8 CK slew=0.500，load=0.500，data slew=0.500时的

测试集corners的Tcq的仿真真实值与预测值及其预测准确度

从图中可以看出37组Corners下Tcq的仿真值和预测值相差很小，可以从预测的准确度来观察仿真值和预测值的差异。

例中对测试集的37组数据的准确度求平均，得到本组Tcq预测的平均准确度为99.83%。

图5-9 测试集corners的全时序仿真数据的预测平均准确度

上图为目标组合的测试集所有时序的平均准确度。可以从图中观察到七个目标组合的测试集在37组PVT Corners下的时序的平均准确度都处于97%以上，其中组合3和组合5的Tds的预测准确度约为97%，除此之外所有的平均准确度都达到98%以上，而且大部分的时序的预测平均准确度可以达到99%。由此可以得出结论：本方法在TSMC28nm工艺下的同一Instance，通过某一确定的Input slew和output loading的时序仿真数据，实现其他Input slew和output loading组合的时序仿真数据的预测，可以预测得到准确度很高的时序仿真数据，同时可以很有成效的减少仿真工作量，节省仿真时间和仿真资源。

#### 5.1.2.2 不同实例预测

本课题中采选用了4个instance，他们分别是W32B8M2B1、W32B144M2B1、W512B8M2B1和W512B144M2B1。Instance的形状与阵列的大小密切相关，从版图中我们可以看出存储阵列几乎占据了SRAM版图98%的面积。4.1中讲到通常阵列会被设计成行数较少、列数较多的结构，因为位线的负载是通过并联的，过长的位线会使得SRAM读写的速度变得很慢。相同的存储容量的SRAM会因为其字线和位线的不同具有不同的读写速度，也就是时序。

Instance的形状与字数和位数线性相关，从版图上看Instance总是矩形，其宽通常为Word/Mux，其长通常为Bit/Mux。

表5-4 4个Instance的特征、存储容量

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inst | W32B8M2B1 | W32B144M2B1 | W512B8M2B1 | W512B144M2B1 |
| Word×Bit | 16×16 | 16×288 | 256×16 | 256×288 |
| 特征 | 面积最小 | 低长 | 高窄 | 面积最大 |
| 存储容量 | 0.4MB | 4.5MB | 4MB | 72MB |

本时序预测方法通过某个Instance的全corner来预测另一个instance的时序。可以实现相同CK slew、Data slew、Load的时候，通过单个instance的全corner时序以及目标instance的6、8组corners的时序，可以实现预测其他corners的时序。

##### 方法一：

图5-10 W32B144M2B1的测试集目标Tcq的仿真值、预测值以及预测准确度

通过W32B8M2B1和W32B144M2B1的8组corners时序数据作为训练集，其中W32B8M2B1的8组corners时序数据作为训练集的特征（Feature），而W32B144M2B1的8个corners时序数据作为训练集的目标（Target）。W32B8M2B1和W32B144M2B1的剩余37组时序数据作为测试集，其中W32B8M2B1的37组时序数据作为测试集的特征，W32B144M2B1的37组时序数据作为测试集的目标。使用最小二乘法对训练集特征和目标进行拟合，然后测试集的特征作为输入，得到测试集的目标预测值，并与测试集的目标真实值进行对比。

本方法中的训练集的8组Corners分别是FG1P08VN40C、FG0P9V25C、FG1P08V25C、TT0P81V25C、TT0P99V25C、SG0P72VN40C、SG0P9V25C和SG1P08V125C。

如图是本次预测的W32B144M2B1的测试集的Tcq的仿真值、预测值以及预测准确度。对39组Tcq的预测准确度进行求平均值，得到本次学习的平均预测准确度为99.69%。从表中也可以看出所有Tcq的预测准确度均处于99%以上，预测效果很好。

图5-11 三个目标Instances测试集的全时序预测平均准确度

上图为W32B144M2B1、W512B8M2B1和W512B144M2B1的测试集的所有时序的平均准确度折线图。从折线图中可以观察到三组Instance的测试集在37组PVT Corners下的时序的平均准确度都处于97%以上，其中W32B144M2B1和W512B144M2B1的Tas的预测平均准确度约为97%，W512B8M2B1的Tds的预测平均准确度约为97%，除此之外所有的平均准确度都达到98.5%以上，而且大部分的时序的预测平均准确度可以达到99%。

##### 方法二：

本方法通过选择W32B8M2B1和W32B144M2B1的6组corners时序数据作为训练集，其中W32B8M2B1的6组corners时序数据作为训练集的特征，而W32B144M2B1的6个corners时序数据作为训练集的目标。W32B8M2B1和W32B144M2B1的剩余39组时序数据作为测试集，其中W32B8M2B1的39组时序数据作为测试集的特征，W32B144M2B1的39组时序数据作为测试集的目标。使用最小二乘法对训练集特征和目标进行拟合，然后测试集的特征作为输入，得到测试集的目标预测值，并与测试集的目标真实值进行对比。

本方法中的训练集的6组Corners分别是FG1P08VN40C、FG1P08V25C、TT0P81V25C、TT0P99V25C、SG0P72VN40C和SG0P72VN40C。

图5-12 W32B144M2B1的测试集目标Tcq的仿真值、预测值以及预测准确度

如图是本次预测的W32B144M2B1的测试集的Tcq的仿真值、预测值以及预测准确度。对39组Tcq的预测准确度进行求平均值，得到本次学习的平均预测准确度为99.68%。从表中也可以看出所有Tcq的预测准确度均处于99%以上，预测效果很好。

图5-13 三个目标Instances测试集的全时序预测平均准确度

上图为W32B144M2B1、W512B8M2B1和W512B144M2B1的测试集的所有时序的平均准确度折线图。从折线图中可以观察到三组Instance的测试集在39组PVT Corners下的时序的平均准确度都处于95%以上，其中W32B144M2B1和W512B144M2B1的Tas、Tws的预测平均准确度约为97%，W512B8M2B1的Tds的预测平均准确度约为95%，除此之外所有的平均准确度都达到98.5%以上，而且大部分的时序的预测平均准确度可以达到99%。

##### 方法三：

通过W32B8M2B1和W32B144M2B1的8组corners时序数据作为训练集，其中W32B8M2B1的8组corners时序数据作为训练集的特征，而W32B144M2B1的8个corners时序数据作为训练集的目标。W32B8M2B1和W32B144M2B1的剩余37组时序数据作为测试集，其中W32B8M2B1的37组时序数据作为测试集的特征，W32B144M2B1的37组时序数据作为测试集的目标。使用最小二乘法对训练集特征和目标进行拟合，然后测试集的特征作为输入，得到测试集的目标预测值，并与测试集的目标真实值进行对比。

本方法中的训练集的8组Corners分别是FG0P81VN40C、FG0P99VN40C、FG0P9V25C、TT0P9VN40C、TT0P9V125C、SG0P81VN40C、SG0P99VN40C和SG0P9V125C。

图5-14 W32B144M2B1的测试集目标Tcq的仿真值、预测值以及预测准确度

如图是本次预测的W32B144M2B1的测试集的Tcq的仿真值、预测值以及预测准确度。对37组Tcq的预测准确度进行求平均值，得到本次学习的平均预测准确度为99.64%。从表中也可以看出所有Tcq的预测准确度均处于98.8%以上，预测效果很好。

图5-15 三个目标Instances测试集的全时序预测平均准确度

上图为W32B144M2B1、W512B8M2B1和W512B144M2B1的测试集的所有时序的平均准确度折线图。从折线图中可以观察到三组Instance的测试集在37组PVT Corners下的时序的平均准确度都处于96%以上，其中W32B144M2B1和W512B144M2B1的Tas的预测平均准确度约为97.5%，W512B8M2B1的Tds的预测平均准确度约为96%，除此之外所有的平均准确度都达到98.5%以上，而且大部分的时序的预测平均准确度可以达到99%。

##### 方法比较

在以上三个方法中，都是通过某一确定Input slew和output loading的Instance的时序仿真数据，以及目标Instance的部分corners的时序仿真数据，通过最小二乘法进行回归训练，并通过训练模型进行测试集的预测。

在以上三个方法中，方法一是用8个corners进行训练集范围内部预测，方法二使用6个corners进行训练集范围内部预测，方法三通过8个corners进行全corners范围预测。三种方法的Tcq预测准确度的最小值分别是99.05%、99.05%、98.77。三种方法的所有时序的预测平均准确度的最小值分别是96.99%、95.31%、96.24%。

从Tcq的准确度和全时序的平均准确度来看，方法一适用于有较多corners仿真数据的目标Instance，且已有corners是处于PVT corners的边缘，并对预测时序数据有非常高准确度和精确度的情况；方法二适用于有较少corners仿真数据的目标Instance，且已有corners是处于PVT corners的边缘，并对预测时序数据有比较高的准确度和精确度的情况；方法三适用于有较多corners仿真数据的目标Instance，且已有corners不是处于PVT corners的边缘，并对预测时序数据有非常高准确度和精确度的情况。

由此可以得出结论：本方法在TSMC28nm工艺下某一确定的Input slew和output loading的某一Instance的时序仿真数据，实现其他Instance的时序仿真数据的预测，提供了三种使用不同情况的预测方法，且三种方法都可以得到准确度很高的时序仿真数据。

以本套SRAM Compiler为例，以50个Intel（R） Xeon® 频率为3.60MHZ的4核CPU，其全corners时序表征仿真工作量需要大致340天。在本时序预测方法应用于仿真工作中，本套SRAM Compiler全部仿真数据获得大概为120小时。缩减了约65%的仿真时间。由此可见，本时序预测方法可以很有成效的减少仿真工作量，同时节省仿真时间和仿真资源。

## 5.2功率

### 5.2.1 功率数据

图5-16 CK、A、DI的波形示意图

如图，读电流的测量时间从CK的下降沿到下一个下降沿，在读周期中只有DI、A、CK在翻转，其他的Pin都是standby，且A和DI应该变化两次。

写电流的测量时间从CK的下降沿到CK的下一个下降沿，在写周期中只有DI、A、CK在翻转，其他的Pin都处于standby，且A和DI都翻转两次。

在light sleep mode、deep sleep mode和shut down mode下，所有的Pin都保持原值，然后在分别在light sleep、deep sleep和shut down的最后几个周期，选择两个周期测量。

在功耗中主要需要采集的功率数据主要有：

1. dc：standby状态下的电流的平均值
2. ac\_read：读操作电流平均值
3. ac\_write：写操作电流平均值
4. ls：light sleep模式下电流平均值
5. ds：deep sleep模式下电流平均值
6. sd：shut down模式下电流平均值

通常为了节省功耗和减少数据读取时间，存储器中的字数会远远多于每个字存储的位数，这样会使得存储器变得长而窄，因为位线过长会增加数据访问的速度。字深的长度和位长的大小对功率也有同样的影响。本方法考虑到样本的多样性，以及数据的差异性，采用4个形状各异的instance作为训练集，如图上上：W32B8M2B1、W32B144M2B1、W512B8M2B1和W512B144M2B1分别是字长的极大值和极小值、位深的极大值和极小值的组合，也就代表了不同字长、位深对于功耗大小响应的极致体现。

本方法共采集了18个instance的全corner仿真数据，通过程序处理和运算得到每组仿真数据分别为6个参数。共得到18组45×6的数据集。鉴于功率参数的复杂性，本方法的训练集采用4个instance的数据集，以其他14个instance的数据作为测试集，分别进行了14次回归学习和预测。分别是W80B8M2B1、W128B8M2B1、W192B8M2B1、W256B8M2B1、W320B8M2B1、W384B8M2B1、W448B8M2B1、W80B144M2B1、W128B144M2B1、W192B144M2B1、W256B144M2B1、W320B144M2B1、W384B144M2B1、W448B144M2B1等14个Instances。

因为功率参数受驱动电压波动的影响会有很明显的波动，通常在仿真阶段SRAM Compiler会选择三种电压作为仿真基准电压，分别是HVT（High Voltage Threshold）、SVT（Standard Voltage Threshold）和LVT（Low Voltage Threshold）。通常功耗会随着阈值电压的增加而减少，就是相同条件下，HVT的功耗最小，LVT的功耗最大。

### 5.2.2 功率预测

以SVT为阈值电压进行TSMC28nm工艺SRAM Compiler全特征的功耗仿真。

图5-17 SVT下W512B8M2B1\_12BIMP的全corners的DC电流值分布图

从图中可以看出图表中根据不同温度的DC数值在不同的量级，变化差异较大，故在此预测方法中对数据进行按温度分组进行回归学习与预测。

本方法将不同的温度分开进行学习。以W80B8M2B1为目标例，

#### 5.2.2.1 125℃

本次回归学习中，训练集Instances的功率数据作为特征，测试集W80B8M2B1的功率数据作为目标。其中训练集Instances的5组corners的数据作为训练集的特征，另外10组corners的数据作为测试集的特征；测试集W80B8M2B1的5组corners的数据作为训练集的目标，另外10组数据作为测试集的目标。使用最小二乘法对训练集特征和目标进行回归拟合，然后将测试集的特征作为输入，得到测试集的目标预测值，并与测试集的目标真实值进行对比。

本方法中的训练集的5组Corners分别是FG0P72V125C、FG1P08V125C、TT0P9V125C、SG0P72V125C、SG1P08V125C。

图5-18 W80B8M2B1的测试集目标DC的仿真值、预测值以及预测准确度

如图是本次预测的W80B8M2B1的测试集的DC的仿真值、预测值以及预测准确度。对10组DC的预测准确度进行求平均值，得到本次学习的平均预测准确度为99.38%。从表中也可以看出所有DC的预测准确度均处于98.5%以上，预测效果很好。

图5-19 14个目标Instances测试集的全功率预测平均准确度

上图为目标Instances测试集所有功率数据的平均准确度折线图。从折线图中可以观察到14组Instances的测试集在10组PVT Corners下功率数据的平均准确度都处于95.5%以上，其中所有Instances在AC\_read和AC\_write处的预测平均准确度约为95.5%-96.5%，在DC和LS处的预测平均准确度约为99.6%，在DS和SD处的预测平均准确度分布在97%-99.5%之间。

#### 5.2.2.2 25℃

本次回归学习中，训练集Instances的功率数据作为特征，测试集W80B8M2B1的功率数据作为目标。其中训练集Instances的5组corners的数据作为训练集的特征，另外10组corners的数据作为测试集的特征；测试集W80B8M2B1的5组corners的数据作为训练集的目标，另外10组数据作为测试集的目标。使用最小二乘法对训练集特征和目标进行回归拟合，然后将测试集的特征作为输入，得到测试集的目标预测值，并与测试集的目标真实值进行对比。

本方法中的训练集的5组Corners分别是FG0P72V25C、FG1P08V25C、TT0P9V25C、SG0P72V25C、SG1P08V25C。

如图是本次预测的W80B8M2B1的测试集的DC的仿真值、预测值以及预测准确度。对10组DC的预测准确度进行求平均值，得到本次学习的平均预测准确度为98.44%。从表中也可以看出所有DC的预测准确度均处于97.5%以上，预测效果很好。

图5-20 W80B8M2B1的测试集目标DC的仿真值、预测值以及预测准确度

图5-21 14个目标Instances测试集的全功率预测平均准确度

上图为目标Instances测试集所有功率数据的平均准确度折线图。从折线图中可以观察到14组Instances的测试集在10组PVT Corners下功率数据的平均准确度都处于96%以上，其中所有Instances在AC\_read和AC\_write处的预测平均准确度约为99.6%，在DC和LS处的预测平均准确度约为98.5%～99.5%，在DS和SD处的预测平均准确度分布在96%～99.5%之间。

#### 5.2.2.3 -40℃

本次回归学习中，训练集Inst ances的功率数据作为特征，测试集W80B8M2B1的功率数据作为目标。其中训练集Instances的5组corners的数据作为训练集的特征，另外10组corners的数据作为测试集的特征；测试集W80B8M2B1的5组corners的数据作为训练集的目标，另外10组数据作为测试集的目标。使用最小二乘法对训练集特征和目标进行回归拟合，然后将测试集的特征作为输入，得到测试集的目标预测值，并与测试集的目标真实值进行对比。

本方法中的训练集的5组Corners分别是FG0P72VN40C、FG1P08VN40C、TT0P9VN40C、SG0P72VN40C、SG1P08VN40C。

图5-22 W80B8M2B1的测试集目标DC的仿真值、预测值以及预测准确度

如图是本次预测的W80B8M2B1的测试集的DC的仿真值、预测值以及预测准确度。对10组DC的预测准确度进行求平均值，得到本次学习的平均预测准确度为99.81%。从表中也可以看出所有DC的预测准确度均处于99.5%以上，预测效果很好。

图5-23 14个目标Instances测试集的全功率预测平均准确度

上图为目标Instances测试集所有功率数据的平均准确度折线图。从折线图中可以观察到14组Instances的测试集在10组PVT Corners下功率数据的平均准确度都处于99.2%以上，其中所有Instances在AC\_read和AC\_write处的预测平均准确度约为99.7%，在DC和LS处的预测平均准确度约为99.5%～99.9%，在DS和SD处的预测平均准确度分布在99.2%～99.9%之间。

从以上SVT作为阈值电压的功耗数据预测中，针对Process corners的温度不同进行了分组学习。W80B8M2B1中功耗DC在125℃、25℃、-40℃时的最小预测准确度分别是98.5%、97.5%、99.5%；由此我们可以推测本预测方法在预测W80B8M2B1的6个功耗参数的准确度都很高，并以此为据，进行了14个目标Instances的仿真，分别得到125℃、25℃、-40℃时每组的6个功耗数据的平均预测准确度，其最小平均预测准确度分别是95.5%、96%、99.2%。从预测平均准确度的结果来看，本方法应用于TSMC28nm工艺的SRAM Compiler的表征功耗数据的预测具有很高的准确度，表明了通过最小二乘法进行回归学习与预测功耗数据的可行性与准确性。

以本套SRAM Compiler为例，以50个Intel（R） Xeon® 频率为3.60MHZ的4核CPU，其全corners功耗表征仿真工作量需要大致720小时。在本功耗预测方法应用于仿真工作中，本套SRAM Compiler全部仿真数据获得大概需要200小时。缩减了约72%的仿真时间。由此可见，本时序预测方法可以很有成效的减少仿真工作量，同时节省仿真时间和仿真资源。

## 5.3 本章小结

本章首先介绍了读操作、写操作和电源模式的时序含义，提出了两种时序预测方法：相同Instances预测、不同Instances预测。其中相同Instances预测可以实现预测相同Instances不同Input slew和output loading的时序数据；不同Instance预测提供三种方法实现相同Input slew和output loading的不同Instances的时序数据预测。且以上方法都能达到97%和95.5%以上的预测平均准确度。并且分析了应用本方法后本SRAM Compiler的时序仿真能节省65%的仿真时间。

本章的第二部分介绍了功率参数的含义，提出了基于温度分类的功率参数分类方法，且本方法能达到至少95.2%、96%、99.2%的预测平均准确度。并分析了应用本方法后本SRAM Compiler的功耗仿真能节省72%的仿真时间。

# 第六章 总结与展望

## 6.1 论文总结

现代SOC应用场景的不断拓展，对SOC不同的需求被提出，存储器作为SOC的重要部分，也对存储器提出了极高的要求，其中使用的SRAM在此背景下需要不断提升以适用于不同的需求，在各种高速、低功耗技术不断被提出的状态下，其对本身就很复杂耗时的SRAM表征环节提出了更高的要求。

本文以UMC28nmHLP工艺的标准单元库为例，提出了两种应用于标准单元库的时序预测方法：电路参数预测方法和时序预测方法。使用最小二乘回归算法预测了标准单元电路的时序，其中电路参数方法的预测时序的准确度能达到87.36%；时序方法的预测时序的准确度能达到98.68%。并从时间成本和机器资源成本对两种方法进行了评估，电路参数方法节约时间大致有30%，时序方法节约时间大致有50%。

以标准单元库时序预测的可行性为出发点，本文还提出了针对SRAM Compiler表征的数据预测方法。

本文还介绍了SRAM的基本框架和基本电路结构，针对影响表征仿真工作量的所有参数特征、word、bit、mux、bank、input slew、output loading和PVT corners等进行了详细的阐述，针对不同仿真的需求：相同Instance不同Input slew和output loading之间的预测、相同input slew和output loading下不同Instance之间的预测。对两种需求分别提出了一种和三种预测方法，其中针对不同instance之间预测的三种方法分别具有不同的应用场景。从预测结果来看，相同Instance预测方法能达到最低97%的平均准确度；不同Instance预测方法分别能达到最低96.99%、95.31%、96.24%的平均准确度。从时间成本个机器成本进行了评估，本时序预测方法能节省仿真时间大约67%。

最后对SRAM Compiler表征的功耗数据也提出了一种基于温度分类的功率参数分类方法，三个温度段的预测方法能达到至少95.2%、96%、99.2%的预测平均准确度。并分析了应用本方法后本SRAM Compiler的功耗仿真能节省73%的仿真时间。

## 6.2 工作展望

本文主要的工作集中在SRAM Compiler表征的时序与功耗的预测，本文的不足以及后续工作包括：

1. 本课题时序数据都是选取全特征的Instance，未对单个或多个Instance特征进行仿真与预测工作，接下来会对这部分进行研究。
2. 本课题功耗数据中只针对SVT下的功耗数据进行了预测工作，将来需要对HVT和LVT下的功耗数据进行预测研究。
3. 本文采用TSMC28nmHPC+工艺的SRAM Compiler进行了表征数据预测，接下来还需要针对不同工艺的SRAM Compiler进行表征数据的预测。

# 参考文献

[1] 移动智能终端暨智能硬件白皮书. 中国信息通信研究院, 2016.

[2] 关立军. 基于28nm工艺低电压SRAM单元电路设计[D].安徽大学,2017.

[3] 冯海玉. 中国集成电路市场的结构性分析[J].中国集成电路,2016,25(08):14-16+20. [4] Zhang K. Embedded Memories for Nano-Scale VLSIs [M]. Springer US, 2009.

[5] Moore G E. The microprocessor: engine of the technology revolution [M]. ACM, 1997.

[6] Narendra S, De V, Borkar S, et al. Full-chip subthreshold leakage power prediction and reduction techniques for sub-0.18-μm CMOS[J]. IEEE Journal of Solid-State Circuits, 2004, 39(3):501-510.

[7] De V, Borkar S. Technology and design challenges for low power and high performance [microprocessors][C] International Symposium on Low Power Electronics and Design, 1999. Proceedings. IEEE, 1999:163-168.

[8] Burd T, Brodersen R. Design Issue for Dynamic Voltage Scaling [J]. In Proceedings of International Symposium on Low Power Electronics and Design (ISLPED, 2001:9 - 14.

[9] Zhai B, Blaauw D, Sylvester D, et al. The limit of dynamic voltage scaling and insomniac dynamic voltage scaling [J]. IEEE Transactions on Very Large Scale Integration Systems, 2005, 13(11):1239-1252.

[10] Burd T D, Pering T A, Stratakos A J, et al. Dynamic voltage scaled microprocessor system[J]. Solid-State Circuits, IEEE Journal of, 2000, 35(11):1571-1580.

[11] Cho Y. Simultaneous optimization of battery-aware voltage regulator scheduling with dynamic voltage and frequency scaling[C] ACM/IEEE International Symposium on Low Power Electronics and Design. IEEE, 2010:309-314.

[12] Lee J, Nam B G, Yoo H J. Dynamic Voltage and Frequency Scaling (DVFS) scheme for multi-domains power management[C] Solid-State Circuits Conference, 2007. ASSCC '07. IEEE Asian. IEEE, 2008:360-363.

[13] Dutta U, Soni M K, Pattanaik M. A Review of NBTI Degradation and its Impact on the Performance of SRAM [J]. International Journal of Modern Education & Computer Science, 2016.

[14] Liu Z, Kursun V. Characterization of a Novel Nine-Transistor SRAM Cell [J]. IEEE Transactions on Very Large Scale Integration Systems, 2008, 16(4):488-492.

[15] Jagasivamani M, Dong S H. Development of a low-power SRAM compiler [M]. 2000.

[16] El-Thakeb A T, Elhamid H A, Mostafa H, et al. Performance evaluation of finFET based SRAM under statistical VT variability[C] International Conference on Microelectronics. IEEE, 2015:88-91.

[17] Hentrich D, Oruklu E, Saniie J. Performance evaluation of SRAM cells in 22nm predictive CMOS technology[C] IEEE International Conference on Electro information Technology. IEEE, 2009:470-475.

[18] Gupta S, Gupta K, Pandey N. Performance evaluation of SRAM cells for deep submicron technologies[C] Computational Intelligence on Power, Energy and Controls with Their Impact on Humanity. IEEE, 2017:292-296.

[19] 丁海涛. SRAM设计和Compiler技术[D]. 合肥工业大学, 2003.

[20] 谢凯翔. SRAM失效分析[J]. 中国集成电路, 2017, 26(10):70-73.

[21] She Y Q, Zhang L J, Zheng J B, et al. Standard Cell Library Characterization of 28nm Process Based on Machine Learning [J]. 2017(CST).

[22] 柏娜. 超低能耗亚阈值SRAM电路设计[D]. 东南大学, 2011.

[23] 朱贾峰. 低电压SRAM关键技术研究与实现[D]. 东南大学, 2014.

[24] 黎轩. 低电压SRAM关键模块研究与编译器设计[D]. 中国科学技术大学, 2015.

[25] 王旭. 低功耗SRAM电路的设计技术研究[D]. 上海交通大学, 2015.

[26] 李二亮. 基于Mix-IS算法的SRAM设计及良率分析[D]. 苏州大学, 2015.

[27] 胡玉青. 应用于Cache的65nm高速SRAM设计[D]. 苏州大学, 2016.

[28] Hocking R R. A Biometrics Invited Paper. The Analysis and Selection of Variables in Linear Regression [J]. Biometrics, 1976, 32(1):1-49.

[29] Wei L, Chin H C, Cheng S L, et al. Performance evaluation of 14 nm FinFET-based 6T SRAM cell functionality for DC and transient circuit analysis[J]. Journal of Nanomaterials, 2014, 2014(9):105.

[30] Cook R D. Detection of Influential Observation in Linear Regression [J]. Technometrics, 2000, 42(1):65-68.

[31] Montgomery D C, Peck E A, Vining G G. Introduction to linear regression analysis. 4th ED [J]. Journal of Applied Statistics, 1992, 40(12):2775-2776.

Tanaka H, Uejima S, Asai K. Linear Regression Analysis with Fuzzy Model [J]. Systems Man & Cybernetics IEEE Transactions on, 1982, 12(6):903-907.

[32] Wold S, Ruhe A, Wold H, et al. The Collinearity Problem in Linear Regression. The Partial Least Squares (PLS) Approach to Generalized Inverses [J]. Siam J.sci.comput, 1984, 5(3):735-743.

[33] Elkan C. The Foundations of Cost-Sensitive Learning[C] Seventeenth International Joint Conference on Artificial Intelligence. 2001:973--978.

# 攻读学位期间本人出版或公开发表的论著、论文

She Y Q, Zhang L J, Zheng J B, et al. Standard Cell Library Characterization of 28nm Process Based on Machine Learning[J]. 2017(CST).

# 附录

符号 含义

A Address 地址

DI Data input 数据输入

DO Data output 数据输出

CK Clock

CEB Chip enable

WEB Write enable

GWEB Global write enable

EMC Extra margin control

EMCE Extra margin control enable

LS Light sleep

DS Deep sleep

SD Shut down