

一种基于 RS(24,20)的编译码器设计

孙 健,张 辉,王宇飞,刘 明

(西安微电子技术研究所,陕西 西安 710054)

摘 要: 介绍了一种基于 RS(24,20)的编译码实现方案,分析了其设计原理,给出了具体实现过程,通过时序仿真表明,该译码器能实现最大的纠错能力,同时给出了一种具有普适作用的译码方案.

关键词: RS 编码;RS 解码;伽罗华域;电路设计

中图分类号: TN911

文献标识码: A

文章编号: 1000-7180(2016)12-0075-05

An Encoder and Decoder Design Based on RS(24,20)

SUN Jian,ZHANG Hui,WANG Yu-fei,LIU Ming

(Xi'an Microelectronics Technology Institute,Xi'an 710054,China)

Abstract: In this paper,the implementation of RS(24,20)encoding and decoding are discussed. The analysis and the simulation result indicate the ability of the correction and present a normal methodology to construct other RS encoding and decoding.

Key words: RS encoding,RS decoding,Galois Field,Circuit Design

1 引言

随着通信、存储技术的高速发展,各种应用对编码的纠错能力的要求越来越高.比如,海底通信系统传输数据的距离非常远,受衰落、噪声等的影响比较严重,为了保证远距离传输的误码率在允许的范围之内,必须采用很强的纠错编码;在卫星中使用的固态存储器,为了避免恶劣环境的影响(超高温、超低温,太空辐射等等),也需要采用纠错能力强的编码.

RS(Reed-Solomon)码具有同时纠突发错误和随机错误的能力,因而被广泛应用于卫星通信和高速数字系统中,作为提高数据传输和存储可靠性的重要手段,它是当今最有效、应用最广的差错控制编码方式之一^[1].目前 RS 码已经成为美国国家航天局(NASA)和欧洲航天局(ESA)在深空通信级联系统中采用的标准码.本文设计了 RS(24,20)的编译码的各个模块,分析了其设计原理,给出了具体实现过程和仿真结果,通过仿真分析该译码器可以实现预期功能,同时给出了一种具有普适作用的译码

方案.

2 RS 码的原理和性质

由于 RS 码的码元取自伽罗华域(Galois Field, GF),且编码、译码的所有运算都是在伽罗华域中进行的. RS(n,k)是建立在 $GF(2^m)$ 上的纠错码,其中 m 是一个符号的位数.

2.1 伽罗华域的运算

伽罗华域又称有限域,是指按域的构成规则构成的有限个元素的集合,要求满足加法和乘法两种运算的代数系统.如果域中有 p 个元素,则表示为 $GF(p)^{[2]}$.

2.1.1 加减法

在 GF 中,加减是等效的,元素用矢量表示,相应的码元进行异或,如 $GF(2^5)$ 中, $\alpha^1 + \alpha^6 = \alpha^1 - \alpha^6 = 00010 \text{ xor } 01010 = 01000 = \alpha^3$.

2.1.2 乘法

一般元素用幂表示,对指数进行加减后 mod $2^m - 1$,如 $GF(2^5)$ 中, $\alpha^{29} * \alpha^6 = \alpha^{35 \text{ mod } 31} = \alpha^4; \alpha^1 / \alpha^6 = \alpha^{(-5) \text{ mod } 31} = \alpha^{26}$.

2.2 RS 编码原理

设 RS(n, k) 码的码元取自 GF(2^m), 能纠正 $t = (n - k) / 2$ 个错误, 以下各符号及运算都定义在 GF(2^m) 中.

RS 码作为循环码, 有对应的生成多项式:

$$G(x) = (x - \alpha^{m_0})(x - \alpha^{m_0+1}) \cdots (x - \alpha^{m_0+2t+1}) \quad (1)$$

式中, $\alpha^{m_0}, \alpha^{m_0+1}, \dots, \alpha^{m_0+2t+1}$ 为 $G(x) = 0$ 的根, 根据编码要求需要从 GF(2^m) 中选出, 在本设计中 m_0 取值为 1. 要纠正 t 个错误, 就需要 $2t$ 个根.

设需要传输或存储的原始信息为 $(d_{k-1}, d_{k-2}, \dots, d_1, d_0)$, 构成信息码多项式:

$$D(x) = d_{k-1}x^{k-1} + d_{k-2}x^{k-2} + \cdots + d_1x + d_0 \quad (2)$$

用 $x^{n-k}D(x)$ 去除以 $G(x)$, 得到商 $Q(x)$ 和余数 $P(x)$, 即:

$$x^{n-k}D(x) = Q(x)G(x) + P(x) \quad (P(x) \text{ 的次数小于 } n - k) \quad (3)$$

由上式可得:

校验多项式:

$$P(x) = x^{n-k}D(x) \bmod G(x) = p_{n-k-1}x^{n-k-1} + p_{n-k-2}x^{n-k-2} + \cdots + p_1x + p_0 \quad (4)$$

码字多项式:

$$C(x) = Q(x)G(x) + P(x) = c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \cdots + c_1x + c_0 \quad (5)$$

这样, 码字多项式的系数 $(c_{n-1}, c_{n-2}, \dots, c_1, c_0)$ 就是 RS 码, 其中:

$$(c_{n-1}, c_{n-2}, \dots, c_{n-k}) = (d_{k-1}, d_{k-2}, \dots, d_0) \quad (6)$$

$$(c_{n-k-1}, c_{n-k-2}, \dots, c_0) = (p_{n-k-1}, p_{n-k-2}, \dots, p_0) \quad (7)$$

前 k 个系数正好是信息码, 后 $n - k$ 个系数正好是校验码. 编码的目的, 就是求出后 $n - k$ 个系数.

2.3 RS 译码原理

根据收到或读出的码字, 得到多项式:

$$R(x) = C(x) + E(x) = r_{n-1}x^{n-1} + r_{n-2}x^{n-2} + \cdots + r_1x + r_0 \quad (8)$$

式中, $E(x)$ 称为错误多项式:

$$E(x) = Y_e x^{i_e} + y_{e-1} x^{i_{e-1}} + \cdots + Y_i x^{i_1} \quad (9)$$

式中, $Y_e, Y_{e-1}, \dots, Y_1$ 称为错误值; 对应于 Y_j , 有 $x_j = \alpha^{i_j}$ 称为错误位置.

设伴随式 $s_i = R(\alpha^{m_0+i}) = C(\alpha^{m_0+i}) + E(\alpha^{m_0+i})$, 由于 α^{m_0+i} 是 $G(x) = 0$ 的根, $C(\alpha^{m_0+i}) = Q(\alpha^{m_0+i})G(\alpha^{m_0+i}) = 0$, 所以 $s_i = E(\alpha^{m_0+i})$. 如果 s_0 ,

$s_1, \dots, s_{2t-1}$ 都为 0, 就表示没有错误; 只要有一个不为 0, 就表示有错, 必须进行纠错.

$$\text{伴随多项式 } S(x) = s_0 + s_1x + \cdots + s_{2t-1}x^{2t-1} \quad (10)$$

$$\text{错误位置多项式 } \Lambda(x) = \prod_{j=1}^e (1 - X_j x) = 1 + \lambda_1 x + \lambda_2 x^2 + \cdots + \lambda_e x^e \quad (11)$$

($\Lambda(x) = 0$ 的根的倒数就是 X_j , 即错误位置)

错误值多项式为

$$\Omega(x) = w_0 + w_1x + w_2x^2 + \cdots + w_{e-1}x^{e-1} \quad (12)$$

那么, 定义关键方程, 通过求解这个方程, 得到所有 λ 和 w :

$$\Lambda(x) S(x) \equiv \Omega(x) \bmod x^{2t} \quad (13)$$

求出了 λ 后, 使用钱氏搜索 (Chien Search), 得到错误位置. 其中钱氏搜索的原理如下:

错误位置多项式为

$$\Lambda(x) = \prod_{j=1}^e (1 - X_j x) \quad (14)$$

由于该多项式定义在 GF 上, 元素是有限的, 只要列举该域中的所有元素, 代入 $\Lambda(x)$, 检查 $\Lambda(x)$ 是否等于 0, 就能确定哪些元素是根.

如果 $\Lambda(\alpha^i) = 0$, 那么 α^{-i} 一定是一个错误位置; 求出 $\alpha^{-i} = 1 / \alpha^i$, 来确定到底是哪一个位置出现了错误.

如果求得的错误位置数与错误位置多项式的度数不相等, 就表示错误数目超过了该 RS 码的纠错能力, 译码失败; 反之, 则表示译码成功.

求出错误位置后, 使用 Forney 公式求得对应的错误值:

$$Y_i = -\frac{X_i^{-(m_0-1)} \Omega(X_i^{-1})}{\Lambda'(X_i^{-1})} = -\frac{x^{m_0} \Omega(x)}{x \Lambda'(x)_{x=X_i^{-1}}} \quad (15)$$

式中, $\Lambda'(X)$ 为 $\Lambda(X)$ 的导函数.

最后将码字和错误值相加, 得到正确的数据.

3 RS(24, 20) 编译码的实现

根据 RS 原理, 可以根据 RS(24, 20) 得到以下设计参数如表 1 所示.

3.1 RS(24, 20) 编码的实现

RS 编码采用除法方式实现, 编码器的主要组成部分是一组线性反馈移位寄存器 (LFSR), 编码的电路结构图如图 1 所示, 该除法电路模拟了人工进行多项式除法的步骤: 被除数多项式的系数即 20

个码元的信息码从高次项开始依次输入,与上一个 Ri 寄存器的输出相加后,作为本次要求的商值;除数多项式的系数即 4 个码元的 $G(x)$ 系数(g_0, g_1, g_2, g_3)作为乘法器的输入,与本次的商值相乘后,再与前一个寄存器的输出相加(减),完成本次的加(减)法;依次下去,当最后一个信息码输入后,所有寄存器的输出则正好是余数多项式的系数,即校验码. 开关位于 k1 时,输出信息码,位于 k2 时输出校验码,输出的信息码和校验码共同组成编码后码元^[3-4].

表 1 RS(24,20)参数表

参数	说明
$n = 24$	RS 码长 120 bit
$k = 20$	信息码长 100 bit
$t = (n - k) / 2 = 2$	纠正 2 个错误
$F(x) = x^5 + x^2 + 1$	本原多项式
$G(x) = x^4 + a^{24}x^3 + a^{19}x^2 + a^{29}x + a^{10}$	生成多项式

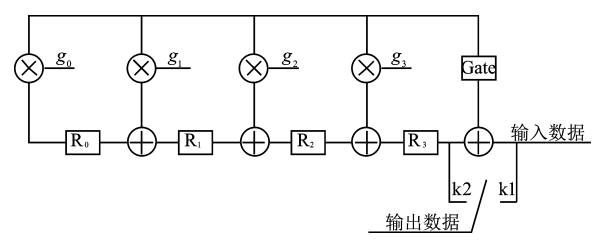


图 1 编码电路图

3.2 RS(24,20)译码的实现

根据 RS 译码原理,译码步骤如图 2 所示.

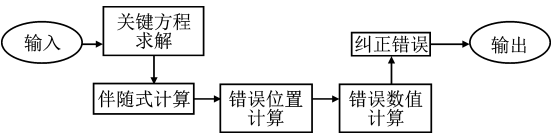


图 2 RS 译码步骤

据此构建了 RS(24,20)译码电路实现的结构框图,如图 3 所示.

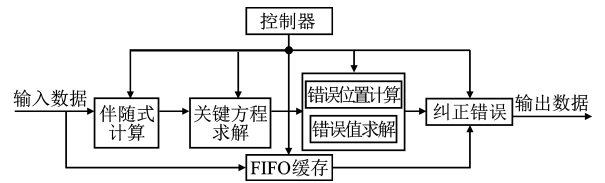


图 3 RS 译码电路结构图

译码器主要由 6 个模块组成:控制模块(Con-

troller)、FIFO 缓冲模块(FIFO Register, FIFO)、伴随式计算模块(Syndrome Computation, SC)、关键方程求解模块(Key-Equation Solver, KES)、错误位置和错误值计算模块(Chien Search and Error Evaluator, CSEE)、纠错模块(Error Correction, EC). 其中,控制器同步和控制码元的流动,是译码器的关键部分^[5].

收到或读出的码字共 24 个码元,从高次开始,依次将待纠正码元读入,一路传给 SC,另一路传给 FIFO;SC 完成计算后,将伴随式的值送往 KES;KES 使用改进的 Berlekamp Massey 算法求解出所有 λ 的值以及部分 $w(h)$ 的值(用于改进的 Forney 公式),送到 CSEE;从高次开始,CSEE 依次确定该位置是否错误位置,并求出相应的错误值(对于正确的位置,CSEE 输出的错误值是 0);同时,FIFO 也同步输出读入的码字,经过 EC 的处理(将原码字和错误值相加),得到正确的码字,完成纠错.

3.2.1 伴随式计算电路设计

用生成多项式 $G(x)$ 的各个根 $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ 带入下式计算伴随多项式,其中的乘法与加法运算均是 GF 乘法和加法.

$$\begin{aligned} s_i &= R(\alpha_i) \\ &= r_0 + r_1 x^1 + r_2 x^2 + \dots + r_{23} x^{23} \\ &= r_0 + x(r_1 + x(r_2 + \dots + x(r_{22} + x r_{23}))) \end{aligned} \tag{16}$$

根据伴随多项式计算的迭代关系,构建的电路实现结构如图 4 所示.

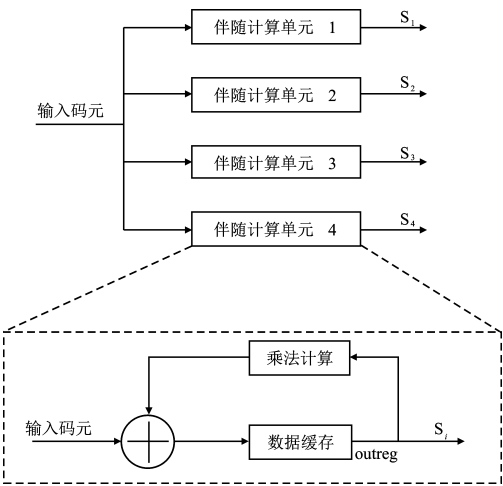


图 4 伴随式计算电路结构

每次运算最多只包括一次乘法和一次加法;以“伴随式计算单元 1”为例:开始时,outreg 为 0,乘以

α 后加上读入的 r_{23} , 存入寄存器; 接着, 数据缓存输出乘以 α , 再加上读入 r_{22} , 存入寄存器; 依此类推, 整个过程中保证码元的流动; 读入最后一个码元后, 计算完成将伴随式计算的值 S_1 保持。

要注意, 数据流有一个串并转换的过程: 串行输入系数, 并行输出 4 个伴随式的值。

加法的结果存在数据缓存寄存器中, 并作为一次乘法操作的一个乘数; 乘法的另一个乘数是常数, 在“伴随式计算单元 1”, “伴随式计算单元 2”, “伴随式计算单元 3”, “伴随式计算单元 4”中分别是 $\alpha^1, \alpha^2, \alpha^3, \alpha^4$; 乘法的结果用于下一个周期的加法操作。

由于乘法的一个乘数是固定的, 所以无需专门的 GF 乘法器, 只要利用多项式乘法求得乘积 (乘数是一个未知的多项式和一个常多项式), 再对本原多项式取模, 记录下余数多项式系数和两个乘数多项式系数的关系, 用组合逻辑即可实现。

3.2.2 关键方程求解电路设计

关键方程的计算采用改进的 Berlekamp Massey 算法求解, 利用伴随式计算的结果, 结合对关键方程的求解, 得到输出的 $\lambda_1, \lambda_2, w_0, w_1$, 用于后续 Chien 搜索提供必要的计算参数。

$$\begin{pmatrix} S_1 & S_2 \\ S_2 & S_3 \end{pmatrix} \begin{pmatrix} \lambda_2 \\ \lambda_1 \end{pmatrix} = \begin{pmatrix} S_3 \\ S_4 \end{pmatrix} \quad (17)$$

$$w_0 = \lambda_1 * S_1 \quad (18)$$

$$w_1 = \lambda_1 * S_2 + \lambda_2 * S_1 \quad (19)$$

3.2.3 错误位置与错误值计算电路设计

根据 RS 译码原理, 要完成最后一步就是必须确定错误值和错误位置多项式的系数, 从而确定错误值和错误位置。由于错误位置数是错误位置多项式 $\Lambda(x)$ 根的倒数, 因此可以通过 Chien 搜索采用验根的方式找出^[6]。RS(24, 20) 的错位位置多项式为

$$\Lambda(x) = 1 + \lambda_1 x + \lambda_2 x^2 \quad (20)$$

为了验证输入码元 α^i 是否为错误位置, 就是看在上式中 $\Lambda(\alpha^i)$ 是否为 0。再根据关键方程中求解出的错误位置多项式的系数 λ_1, λ_2 , 即可得到 Chien 电路结构图如图 5 所示。

其中 Degree_0, Degree_1, Degree_2 分别对应表示 $\Lambda(x) = 1 + \lambda_1 x + \lambda_2 x^2$ 中的 1, $\lambda_1 x, \lambda_2 x^2$, 共同完成 $x = 1, \alpha, \alpha^2, \alpha^3, \dots, \alpha^{31}$ 的计算, 在每一轮计算完成之后即可通过判断 $\Lambda(\alpha^i)$ 是否为 0, 判断 α^i 是不是一个错误位置, 并将错误值转换为伽罗华域表示。

在传输过程中, 如果出现错误, 则 $\Lambda(\alpha^i)$ 不为 0,

即可利用 Forney 算法完成错误值的计算, Forney 的算法结构图如图 6 所示。

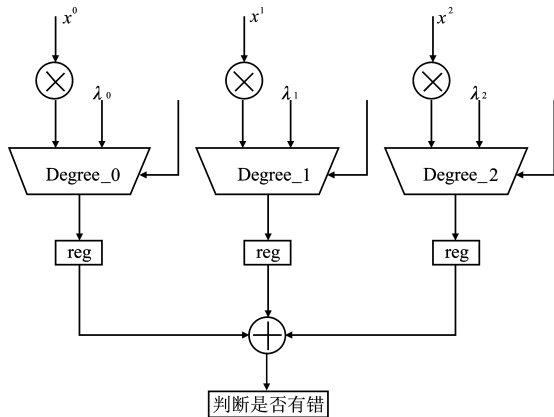


图 5 Chien 搜索电路结构图

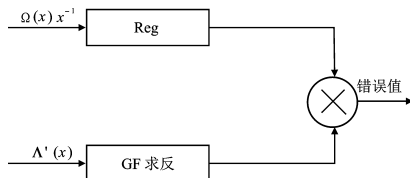


图 6 Forney 电路结构图

在具体应用时采用查找表的方法将输入的位置通过查表可以查找到每个码元的逆元。

3.2.4 纠错电路设计

通过 FIFO 缓存的待纠正数据与计算完的数据进行比较纠错, 在有错误时, 在计算完后同步输出缓存的数据, 如果没有错误的时候输出 FIFO 缓存中的数据。

3.2.5 电路结构

根据 RS 编码各部分电路结构的描述, 通过 Verilog 实现了 RS(24, 20) 译码器的设计, 同时得到 RS(24, 20) 译码器的电路结构如图 7 所示。

3.3 译码仿真结果

RS(24, 20) 译码器的最大纠错能力为 2 个码元, 本文中设定的错误位置为第 2 个码元和第 5 个码元。图 8 为针对最多两个码元的错误进行纠正的仿真结果。仿真结果显示第 2 个码元和第 5 个码元得到有效地纠正并输出了纠错后的编码序列, 有本仿真结果可知, 该设计的译码器能实现该结构编码的最大纠错能力。

4 结束语

本文设计了 RS(24, 20) 的编译码的各个模块, 分析了其设计原理, 给出了具体实现过程和仿真结果。对译码器的设计假定随机出现了两个误码的情况, 通过具体电路模块的分析计算, 验证了该译码模

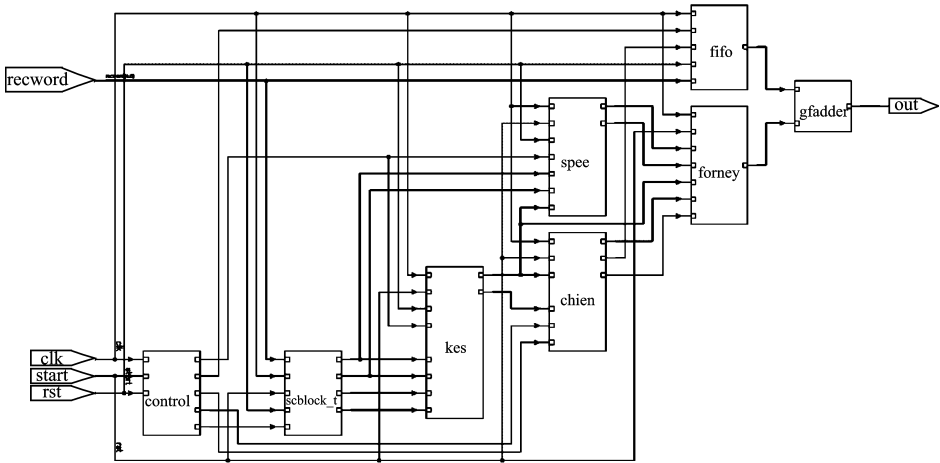


图 7 译码器电路结构图

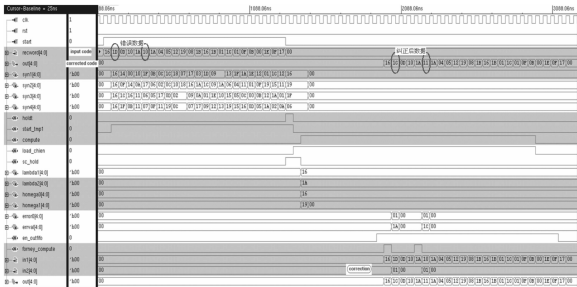


图 8 译码纠错仿真时序图

块的纠错性能. 并通过时序仿真分析该译码器可以实现该结构编码的最大纠错能力. 因此基于该方法设计的 RS(24,20)编译码电路正确, 并且该编译码器可以运用到对纠正突发误码有较高要求的现代无线通信系统中. 同时, 通过对 RS(24,20)译码电路结构的分析, 给出了一种具有普适作用的译码方案.

参考文献:

[1] 王新梅,肖国镇. 纠错码原理与方法[M]. 西安:西安电

子科技大学出版社,2003.

[2] Lin S, Costello D J. Error control coding[M]. Second Edition. New Jersey:Prentical Hall,2004.

[3] Hasan M A, Bharagava V K. Architecture for a Low Complexity Rate-adaptive Reed-Solomon Encoder[J]. IEEE Trans. on Computers,1995,44(7):938-942.

[4] 戴小红,潘志文. Reed-Solomon 编码器的设计与 FPGA 实现[J]. 现代电子技术,2006,3(3):119-121.

[5] 归绍升. 纠错码编码技术和应用[M]. 上海:上海交通大学出版社,1988.

[6] 严来金,李明. RS(255,223)译码器的设计与 FPGA 实现[J]. 微计算机信息,2005,21(1):5-7.

作者简介:

孙 健 男,(1983-),工程师. 研究方向为 SoC 验证. E-mail:sj_2003_ren@163.com

张 辉 男,(1982-),工程师. 研究方向为 SoC 验证.

王宇飞 男,(1983-),工程师. 研究方向为 SoC 验证.

刘 明 男,(1985-),工程师. 研究方向为 SoC 验证.

(上接第 74 页)

[4] 郑晓光,白国强. 一种基于 JTAG 的 SOC 测试电路设计及实现[J]. 微电子学与计算机,2009,26(1):187-189.

[5] IEEE Std 1149. 7. IEEE standard for reduced-pin and enhanced-functionality test access and boundary-scan architecture[S]. American:IEEE,2009.

[6] 陈寿宏,颜学龙,黄新. 基于 IEEE 1149. 7 标准的 CJTAG 测试设计方法研究[J]. 测控技术与仪器仪表,2013,39(1):79-82.

[7] 徐志磊. 基于 IEEE1149. 7 的新一代测试界接口实现与应用[J]. 信息技术,2010,34(8):164-166.

[8] 黄文君. 基于 IEEE 1149. 7 边界扫描技术的研究[D]. 桂林:桂林电子科技大学,2001.

[9] 陈光福,潘中良. 可测性设计技术[M]. 北京:电子工业出版社,1997.

作者简介:

颜学龙 男,(1962-),教授,硕士生导师. 研究方向为集成电路测试理论与测试仪器.

何正亮(通讯作者) 男,(1989-),硕士研究生. 研究方向为检测技术与自动化装置. E-mail:530894041@qq.com.

陈寿宏 男,(1981-),高级实验师,硕士生导师. 研究方向为集成电路测试理论与仪器.