(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 102122964 B
(45)授权公告日 2014.07.02

(21)申请号 201110081379.X

(22)申请日 2011.03.31

(73)专利权人 西安电子科技大学
地址 710071 陕西省西安市太白南路 2 号

(72)发明人 宫丰奎 彭克蓉 葛建华

(51)Int.Cl.
H03M 13/15(2006.01)
H04L 1/00(2006.01)

(56)对比文件
US 6141787 A,2000.10.31, 全文.
CN 1344439 A,2002.04.10, 全文.

审查员 郭从征

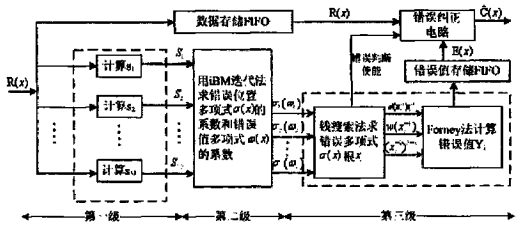
权利要求书2页 说明书8页 附图4页

(54)发明名称

一种基于 FPGA 的高速 RS 编译码器实现方法

(57)摘要

本发明公开了一种基于 FPGA 的高速 RS 编译码器实现方法，包括高速 RS(244,212) 编码器的 FPGA 实现与高速 RS(244,212) 译码器的 FPGA 实现，高速 RS 编码器基于多项式除法的电路，高速 RS 译码器基于三级流水线结构，采用双时钟驱动，时钟 i_clk 与反向时钟 i_clk180，同时，在普通 GF 域乘法器的基础上，提出三种基本运算单元，常系数 GF 域乘加器，常系数 GF 域乘法器以及两时钟周期控制的 GF 域乘法器，不仅大大提高了运算速度，还降低了硬件复杂度，本发明支持吞吐率高，纠正突发错误能力强，可满足多方面的应用。



CN 102122964 B

1. 一种基于 FPGA 的高速 RS 编译码器实现方法，该方法包括高速 RS(244,212) 编码器的 FPGA 实现与高速 RS(244,212) 译码器的 FPGA 实现，其特征在于：

所述的高速 RS(244,212) 编码器的 FPGA 实现方法，对每组串行输入的 212 个信息码元 $\{m_1, m_2, \cdots, m_{212}\}$ 编码得到 32 个校验码元 $\{p_1, p_2, \cdots, p_{32}\}$，具体步骤是：(1) 开关 K1 接 a 口，闭合开关 K2，在时钟 i_clk 的驱动下，输入的第 k 个信息码元 m_k 在使能信号的控制下输入编码器，与寄存器 D0 进行 GF 域相加运算得 S_{md}，k 表示码元序号，初始为 1，同时，编码器将输入信息码元 m_k 直接输出；(2) 在时钟 i_clk180 的驱动下，S_{md} 与寄存器 D1 进行常系数 GF 域乘加运算，结果存储至寄存器 D0；(3) 在时钟 i_clk 的驱动下，S_{md} 同时与寄存器 D2 ～ D31 进行常系数 GF 域乘加运算，结果分别存储至寄存器 D1 ～ D30，并将 S_{md} 存储于寄存器 D31 中；(4) 判断信息码元是否输入完毕，如果是，执行步骤 (5)，否则，输入第 k+1 个信息码元 m_{k+1} 到编码器，返回步骤 (1)；(5) 所有信息码元计算完成后，断开开关 K2，K1 接 b 口，在时钟 i_clk 的驱动下，串行输出寄存器 D0 ～ D31 的值作为检验码元，则该组信息码元编码完毕；

所述的高速 RS(244,212) 译码器的 FPGA 实现方法，采用双时钟驱动及三级流水线结构，包括伴随式计算模块、求解关键方程模块、错误值获取模块、信息存储 FIFO、错误值存储 FIFO 以及纠正错误电路，译码后的数据由所述纠正错误电路输出；

所述的双时钟驱动，对硬件实现比较复杂的部分模块采用反向时钟驱动，使其达到高速实现目的；

所述的三级流水线结构，其中第一级为伴随计算模块，第二级为求解关键方程模块，第三级为错误值获取模块以及纠正错误电路；

所述伴随式计算模块，通过每组 244 个数据码元 $\{c_1, c_2, \cdots, c_{244}\}$ 的串行输入，计算出伴随多项式的系数 $\{S_1, S_2, \cdots, S_{32}\}$，总共分为 32 个子模块，分别对应 32 个伴随多项式的系数，其结果并行输入求解关键方程模块，用于求解错误位置多项式和错误值多项式的系数；

所述的求解关键方程模块，根据所输入的伴随多项式系数 $\{S_1, S_2, \cdots, S_{32}\}$ 和 RiBM 算法来进行迭代运算，将得到的求错误位置多项式的系数 $\{\sigma_1, \sigma_2, \cdots, \sigma_{32}\}$ 和错误值多项式的系数 $\{\omega_1, \omega_2, \cdots, \omega_{32}\}$，以及错码数目，并行输入错误值获取模块，用于判断是否译码正确和计算错误位置以及错误值；

所述的错误值获取模块，包括钱搜索模块：将钱搜索法得到错误位置多项式的根 x_i，以及错误值计算中该根相对应的错误值多项式的值 $w(x_i^{-1})$ 的计算，综合于一个模块之中，同时得到错误位置多项式的根 x_i、错误值多项式的值 $w(x_i^{-1})$ 、 $\sigma'(x_i^{-1})x_i^{-1}$，以及常数项因子 $(x_i^{-1})^{111}$，其中 i 表示错误位置多项式的根的序号，初始为 1；错误值计算模块：得采用钱搜索模块得到的相应值计算出错误值；同时错误值获取模块将计算出的错误值，输出给错误值存储 FIFO 中存储起来；另外，钱搜索模块还输出判断使能信息，用于控制纠正错误电路；

所述的信息存储 FIFO，用于把输入的数据码元按先后顺序存储到 FPGA 芯片的 FIFO 单元，待错误值获取模块计算完成后，根据错误情况将数据依次输出到纠正错误电路中；

所述的错误值存储 FIFO，用于把错误值获取模块的输出数据，按先后顺序存储到 FPGA 芯片的 FIFO 单元中，待获取错误值模块计算完成后，根据错误情况将所计算的错误值依次

输出到纠正错误电路中；

所述的纠正错误电路，是采用错误值获取模块输出的判断使能信息，控制信息存储FIFO和错误值存储FIFO，当判断译码器已正确译码时，则将两者存储的信息顺序输出，进行GF域相加运算，将结果作为已译码字输出，否则，直接将信息存储FIFO的信息作为输出。

2. 根据权利要求1所述的一种基于FPGA的高速RS编译码器实现方法，其特征在于：所述的常系数GF域乘加运算在普通GF域乘法器的基础上，确定其中一个变量的值，同时，针对这个确定的变量，对乘法器进行改进。

3. 根据权利要求1所述的一种基于FPGA的高速RS编译码器实现方法，其特征在于：所述的常系数GF域乘加运算使GF域乘法和GF域加法同时进行，而仅在常系数GF域乘法器的结构上增加了一个异或运算。

4. 根据权利要求1所述的一种基于FPGA的高速RS编译码器实现方法，其特征在于：两时钟周期控制的所述常系数GF域乘加运算，将普通GF域乘法器结构分成两个子结构，分段进行运算。

一种基于 FPGA 的高速 RS 编译码器实现方法

技术领域

[0001] 本发明属于通信中信道编译码装置领域，特别涉及一种可用于卫星高速信号处理的基于 FPGA 的高速 RS(244,212) 编译码器实现方法。

背景技术

[0002] RS 码，由 Reed 和 Solomon 应用 Mattson-Solomon(MS) 多项式于 1960 年构造出来，是一类有很强纠错能力的多进制 BCH 码，它既能纠正随机错误又能纠正突发错误，这种良好的特性使它特别适用于信道干扰非常复杂的通信系统中。所谓的复杂信道干扰的情况，就是指信道中出现的错误类型在某一时刻可能是突发错误也可能是随机错误，但是在某一确定时刻只能有一种类型的错误。RS 码不仅是一种很好的纠正随机错误的码，又是一种接近最佳的纠正突发错误的码。RS 码广泛应用于工程实际之中，在数字传输系统中，如果功率受限且通信质量要求高（如深空通信、潜水通信、移动通信等），一般都采用以 RS 码为外码的级联码，在深空探测中，往往需要传输大量珍贵的遥测数据和工程数据，或者需要实时地传送清晰的动态图像数据，使用符合 CCSDS 标准的 RS 纠错码技术，可以确保所传送数据的可靠性。

[0003] RS 的编码实现较为简单，在 RS 编码速度上，Zhigang Ren 提出一种改进，选择 Altera 公司的 Cyclone III(EP3C25Q240C8)，最大编码时钟频率为 262.26MHz。

[0004] 而在 RS 码的译码过程中，由于译码方法中关键方程的求解，采用了多次迭代的方法，本身实现就比较复杂，又因为译码方法的工程实现也比较困难，从而导致其工程实现成本较高，且难以达到理想的译码速度，所以，一种 RS 码是否能在实际中得到应用，很大程度上取决于译码算法是否可以简单、快速、经济。此前，Xilinx 公司和法国 MATRA MARCONI 公司生产过满足 CCSDS 标准的 (255,223)RS 码的译码芯片，其中，Xilinx 公司的译码芯片的两个输入码块之间的时间间隔不小于 405 个时钟周期，法国 MATRA MARCON 公司生产的译码芯片的最大数据通过率也不超过 100Mbits/s，另外，专利 CN100384116C 中提出一种符合 CCSDS 标准的高速 RS 译码芯片，采用 Xilinx 公司的 xcv600e-6hq240c 作为实现芯片，其数据通过率大于 400Mbit/s，资源使用小于 18 万系统门，所以，减少输入码块之间的间隔时钟周期和提高译码器部分的数据通过率是近期工程实现的迫切要求。

发明内容

[0005] 本发明的目的在于克服上述已有技术的不足，提出一种基于 FPGA 的高速 RS 编译码器实现方法，以提高其通用性以及其所支持的数据吞吐率，满足不同场景的通信需求。

[0006] 为了实现上述目的，本发明的技术方案如下：

[0007] 本发明方法的技术原理包括 RS 编码原理与 RS 译码原理。

[0008] 1. RS 编码原理

[0009] 令 α 为伽罗华域 $GF(2^m)$ 中的本原元，可纠 t 个错误的 RS 码的生成多项式形式为：

[0010] $$g(x)=\prod_{j=b}^{b+2t-1}(x-\alpha^{sj})=\sum_{i=0}^{32}g_i x^i \quad (1)$$

[0011] 其中 b 称为偏移量，s 称为步进因子。

[0012] 注：由于缩短循环码的生成多项式与原码是相同的，所以这不影响生成多项式的形式。

[0013] 以 RS 系统码 (n，k) 为例，其中，码字的前 k 位是信息位，k+1 到 n 位是检验位，且

[0014] $C(x) = m(x)x^{n-k}+r(x)$ (2)

[0015] 首先，令需要编码的序列为：

[0016] $m = (m_{k-1}, m_{k-2}, Lm_1, m_0,)$ (3)

[0017] 那么，编码多项式为：

[0018] $m(x) = m_{k-1}x^{k-1}+m_{k-2}x^{k-2}+L+m_1x+m_0$ (4)

[0019] 其中 $k = n-2t$。

[0020] 在系统码形式下，2t 个校验位恰是信息多项式 $x^{2t}m(x)$ 除以生成多项式 g(x) 后得到的余式 $r(x) = r_{2t-1}x^{2t-1}+r_{2t-2}x^{2t-2}+L+r_1x+r_0$ 的系数。即：

[0021] $m(x)x^{2t} = q(x)g(x)+r(x)$ (5)

[0022] 此时，编码器输出的码字多项式为：

[0023] $C(x) = m(x)x^{2t}+r(x) = q(x)g(x)$ (6)

[0024] 因此，RS 码的编码问题就是以 g(x) 为模的除法问题。

[0025] 另外，本发明实现例所涉及的 RS(244,212) 码，每组码字包含 244 个码元，其中前 212 个为信息码元 $\{m_1, m_2, L, m_{212}\}$，后 32 个为校验码元 $\{p_1, p_2, L, p_{32}\}$。

[0026] 其中本原多项式采用：

[0027] $F(x) = x^8+x^7+x^2+x+1$ (7)

[0028] 生成多项式：

[0029] $$G(x)=\prod_{j=112}^{143}(x-\alpha^{11j})=\sum_{i=0}^{32}g_i x^i$$

[0030] $$=x^{32}+g_1x^{31}+g_2x^{30}+g_3x^{29}+g_4x^{28}+g_5x^{27}+g_6x^{26}+g_7x^{25}+g_8x^{24}$$

[0031] $$+g_9x^{23}+g_{10}x^{22}+g_{11}x^{21}+g_{12}x^{20}+g_{13}x^{19}+g_{14}x^{18}+g_{15}x^{17}+g_{16}x^{16}+g_{15}x^{15}$$

[0032] $$+g_{14}x^{14}+g_{13}x^{13}+g_{12}x^{12}+g_{11}x^{11}+g_{10}x^{10}+g_9x^9+g_8x^8+g_7x^7+g_6x^6 \quad (8)$$

[0033] $$+g_5x^5+g_4x^4+g_3x^3+g_2x^2+g_1x+1$$

[0034] $$=x^{32}+91x^{31}+127x^{30}+86x^{29}+16x^{28}+30x^{27}+13x^{26}+235x^{25}+97x^{24}$$

[0035] $$+165x^{23}+8x^{22}+42x^{21}+54x^{20}+86x^{19}+171x^{18}+32x^{17}+113x^{16}+32x^{15}$$

[0036] $$+171x^{14}+86x^{13}+54x^{12}+42x^{11}+8x^{10}+165x^9+97x^8+235x^7+13x^6$$

[0037] $$+30x^5+16x^4+86x^3+127x^2+91x+1$$

[0038] 2. RS 译码原理

[0039] 首先，令发送端传输的码字多项式为：

[0040] $C(x) = c_{n-1}x^{n-1}+c_{n-2}x^{n-2}Lc_1x+c_0$ (9)

[0041] 令接收端的接收序列多项式为：

[0042] $R(x) = r_{n-1}x^{n-1}+r_{n-2}x^{n-2}Lr_1x+r_0$ (10)

[0043] 令差错图样多项式为：

[0044] $$E(x)=e_v x^{l_v}+e_{v-1}x^{l_{v-1}}L+e_2 x^{l_2}+e_1 x^{l_1} \quad (11)$$

[0045] 其中x^{l_i}为错误位置数，该位置的错误值为 e_i。

[0046] 那么：

[0047] $$R(x)=C(x)+E(x) \quad (12)$$

[0048] RS 码的传统译码方法与一般的线性码相同，分为以下三步：

[0049] 1) 由接收到的码字 R(x) 计算伴随式 S(x)。

[0050] 2) 由伴随式 S(x) 确定错误图样 E(x)。

[0051] 3) 计算 R(x)-E(x) = C(x)，得到译码器输出码字 C(x)；若译码器不能得到 E(x)，则译码失败，此时译码器指出 R(x) 中有错误，但不能纠正。

[0052] 其中第 2) 步又可以分为求错误位置多项式 σ(x) 和错误值多项式 ω(x) 两步。伴随式 S(x) 和 σ(x) 以及 ω(x) 之间应满足关键方程：

[0053] $$S(x)\sigma(x)=\omega(x)\bmod x^{2t} \quad (13)$$

[0054] 译码器的结构具体包括伴随多项式的计算、错误位置多项式的计算、钱搜索和错误值计算四个模块，伴随多项式计算模块生成多项式 S(x)，它可用于关键方程求解模块求解关键多项式 $S(x)\sigma(x)=\omega(x)\bmod x^{2t}$，其中，我们可以用 ME 算法或 BM 算法来解关键方程，得到错误位置多项式 σ(x) 和错误值多项式 ω(x)。然后，通过钱搜索模块和 Forney 算法模块，这两个多项式可用来得到错误位置和对应的错误值，在译码器的输出端进行纠错。另外，FIFO 存贮器是根据这些模块的时延来缓存接收信号的。其中，决定译码器复杂度的主要因素在于求解关键多项式。

[0055] 3. 技术方案

[0056] 为实现高速目的，本发明所述的一种基于 FPGA 的高速 RS 编译码器实现方法包括高速 RS(244,212) 编码器的 FPGA 实现与高速 RS(244,212) 译码器的 FPGA 实现，采用双时钟驱动来实现。

[0057] 实现高速 RS(244,212) 编码器的 FPGA 方法如下：

[0058] 高速 RS(244,212) 编码器采用双时钟驱动模式工作，提高了时钟利用率，从而提高了编码速度，且资源消耗较低。对每组串行输入的 212 个信息码元 $\{m_1, m_2, L, m_{212}\}$ 编码得到 32 个校验码元 $\{p_1, p_2, L, p_{32}\}$，具体实现方法包括如下步骤：

[0059] (1) 开关 K1 接 a 口，闭合开关 K2，在时钟 i_clk 的驱动下，输入的第 k 个信息码元 m_k 在使能信号的控制下输入编码器，与寄存器 D0 进行 GF 域相加运算得 S_{md}，k 表示码元序号，初始为 1，同时，编码器将输入信息码元 m_k 直接输出；

[0060] (2) 在时钟 i_clk180 的驱动下，S_{md} 与寄存器 D1 进行常系数 GF 域乘加运算，结果存储至寄存器 D0；

[0061] (3) 在时钟 i_clk 的驱动下，S_{md} 同时与寄存器 D2 ～ D31 进行常系数 GF 域乘加运算，结果分别存储至寄存器 D1 ～ D30，并将 S_{md} 存储于寄存器 D31 中；

[0062] (4) 判断信息码元是否输入完毕，如果是，执行步骤 (5)，否则，输入第 k+1 个信息码元 m_{k+1} 到编码器，返回步骤 (1)；

[0063] (5) 所有信息码元计算完成后，断开开关 K2，K1 接 b 口，在时钟 i_clk 的驱动下，串行输出寄存器 D0 ～ D31 的值作为检验码元，则该组信息码元编码完毕；

[0064] 本发明中，所述的高速 RS(244,212) 编码器的 FPGA 方法采用基于多项式除法的电

路，调用了 32 个常系数 GF 域乘加器，通过反相驱动时钟 i_clk180 控制其中一个常系数 GF 域乘加器，使其达到高速编码的效果。

[0065] 实现高速 RS(244,212) 译码器的 FPGA 方法如下：

[0066] 本发明所述高速 RS(244,212) 译码器的 FPGA 方法采用双时钟驱动，具有三级流水线结构，包括伴随式计算模块、求解关键方程模块、错误值获取模块（包括钱搜索，错误值计算，能否纠错判断）、信息存储 FIFO、错误值存储 FIFO 以及纠正错误电路，译码后的数据由纠正错误电路输出。

[0067] 所述的双时钟驱动，在普通 RS 译码的流程上，对硬件实现较复杂的子模块，采用反向时钟驱动，使其达到高速运算的同时，也降低了硬件电路的复杂度。

[0068] 所述的三级流水线结构，其中第一级为伴随计算模块，第二级为求解关键方程模块，第三级为错误值获取模块以及纠正错误电路等，它在很大程度上提高了译码速度，减少了两输入码块之间的间隔时钟周期。

[0069] 所述伴随式计算模块，通过每组 244 个数据码元 $\{c_1, c_2, L, c_{244}\}$ 的串行输入，计算出伴随多项式的系数 $\{s_1, S_2, L, S_{32}\}$，总共分为 32 个子模块，分别对应 32 个伴随多项式的系数，其结果并行输入求解关键方程模块，用于求解错误位置多项式和错误值多项式的系数。

[0070] 所述的求解关键方程模块，根据所输入的伴随多项式系数 $\{S_1, S_2, L, S_{32}\}$ 和 RiBM 算法来进行迭代运算，将得到的错误位置多项式的系数 $\{\sigma_1, \sigma_2, L, \sigma_{32}\}$ 和错误值多项式的系数 $\{\omega_1, \omega_2, L, \omega_{32}\}$，以及错码数目，并行输入错误值获取模块，用于判断是否译码正确和计算错误位置以及错误值，其中两组多项式的系数之间是串行输出的。

[0071] 所述的错误值获取模块，包括钱搜索模块：将钱搜索法得到错误位置多项式的根 x_i，以及错误值计算中该根相对应的错误值多项式的值 $w(x_i^{-1})$ 的计算，综合于一个模块之中，同时得到错误位置多项式的根 x_i、错误值多项式的值 $w(x_i^{-1})$、$\sigma'(x_i^{-1})x_i^{-1}$ 以及常数项因子 $(x_i^{-1})^{b-1}$，在不影响速度的前提下，大大降低了错误值计算模块所需要的硬件资源和时钟周期；错误值计算模块：采用钱搜索模块得到的相应值计算出错误值。同时错误值获取模块将计算出的错误值，输出给错误值存储 FIFO 中存储起来。另外，钱搜索模块还输出判断使能信息，用于控制纠正错误电路。

[0072] 所述的信息存储 FIFO，用于把输入的数据码元按先后顺序存储到 FPGA 芯片的 FIFO 单元，待错误值获取模块计算完成后，根据错误情况将数据依次输出到纠正错误电路中；

[0073] 所述的错误值存储 FIFO，用于把错误值获取模块的输出数据，按先后顺序存储到 FPGA 芯片的 FIFO 单元中，待获取错误值模块计算完成后，根据错误情况将所计算的错误值依次输出到纠正错误电路中；

[0074] 所述的纠正错误电路，是采用错误值获取模块输出的判断使能信息，控制信息存储 FIFO 和错误值存储 FIFO，当判断译码器已正确译码时，则将两者存储的信息顺序输出，进行 GF 域相加运算，将结果作为已译码字输出，否则，直接将信息存储 FIFO 的信息作为输出。

[0075] 本发明中，所述的常系数 GF 域乘法器在普通 GF 域乘法器的基础上，确定其中一个变量的值，同时，针对这个确定的变量，对乘法器进行改进，大大降低其基本运算单元的复杂度，从而提高了运算速度。

[0076] 本发明中，所述的常系数 GF 域乘加器使 GF 域乘法和 GF 域加法操作同时进行，而仅在常系数 GF 域乘法器的结构上增加了一个异或运算，非常易于硬件实现，同时也大大提高了运算速度。

[0077] 本发明中，所述的两时钟周期控制 GF 域乘法器，将普通 GF 域乘法器结构分成两个子结构，分段进行运算，以一个时钟的时延得到了速度的提高。

[0078] 与现有技术相比，本发明的有益效果是：

[0079] 1、采用双时钟驱动，高速 RS 编译码器的数据通过率高，如用 Xilinx 公司的 xc4vlx160-12ff1148 芯片进行综合仿真以及静态时序分析，其中，编码器最高可工作于 560MHz 的时钟，其数据吞吐量约为 4.48Gbit/s，译码器最高可工作于 370MHz 的时钟，其数据吞吐量约为 2.96Gbit/s。另外，其数据资源使用率也不高，如下表所列。

[0080]

	Occupied Slice	Slice Flip Flops	4 input LUT	IOB	GCLK
RS 编码器	283	313	463	22	2
RS 译码器	3269	3387	5926	21	2

[0081] 2、高速 RS 译码器采用三级流水线结构，降低了输入码块之间的间隔时钟周期，同时将钱搜索和错误值的部分计算结合于一个模块，提高了流水线结构的效率。

[0082] 3、提出三种基本运算单元，常系数 GF 域乘加器，常系数 GF 域乘法器，以及两时钟周期控制的 GF 域乘法器，几种乘法器结合运用，不仅大大提高了运算速度，还降低了硬件复杂度。

[0083] 4、整个编译码过程均在一片 FPGA 芯片中实现，工作性能稳定可靠，同时能够容易移植。

[0084] 5、针对不同的实际应用需求，可以在保持外部接口不变的情况下灵活地修改内部电路设计，以便于和不同的设备进行衔接。

[0085] 6、不但可以纠正随机错误，而且具有很强的纠正突发错误的功能。

[0086] 本发明针对不同环境均可应用，具有很强的选择性，采用正反双时钟驱动，在很大程序上提高了数据吞吐量，且又避免了消耗资源过大的缺点，使硬件实现的复杂度大幅降低。此外由于在硬件实现中，均采用最简单的硬件单元，非常易于硬件实现。

附图说明

[0087] 图 1 为本发明的高速 RS 编码器的实现示意图；

[0088] 图 2 为本发明的高速 RS 译码器的三级流水线结构示意图；

[0089] 图 3 为本发明采用的伴随式计算子模块示意图；

[0090] 图 4 为本发明采用的 RiBM 算法的流程示意图；

[0091] 图 5 为本发明采用的钱搜索计算的结构框图；

[0092] 图 6 为本发明提出的常系数 GF 域乘加器的电路结构示意；

[0093] 图 7 为本发明提出的常系数 GF 域乘法器的电路结构示意；

[0094] 图 8 为本发明提出的高速 RS 编码器的仿真时序图；

[0095] 图 9 为本发明提出的高速 RS 译码器的仿真时序图。

具体实施方式：

[0096] 为了使本发明的技术手段、创作特征与达成目的易于明白理解，以下结合具体实施例进一步阐述本发明。

[0097] 本发明所述的一种基于 FPGA 的高速 RS 编译码器实现方法包括高速 RS(244,212) 编码器的 FPGA 实现与高速 RS(244,212) 译码器的 FPGA 实现，采用反向时钟来实现。

[0098] 实现高速 RS(244,212) 编码器的 FPGA 方法如下：

[0099] 所述高速 RS(244,212) 编码器采用双时钟驱动模式工作，对每组串行输入的 212 个信息码元 $\{m_1, m_2, L, m_{212}\}$ 编码得到 32 个校验码元 $\{p_1, p_2, L, p_{32}\}$，在不增加硬件复杂度的情况下，提高了时钟利用率，从而提高了编码速度，且资源消耗较低，如图 1，具体实现方法包括如下步骤：

[0100] (1) 开关 K1 接 a 口，闭合开关 K2，在时钟 i_clk 的驱动下，输入的第 k 个信息码元 m_k 在使能信号的控制下输入编码器，与寄存器 D0 进行 GF 域相加运算得 S_{md}，k 表示码元序号，初始为 1，同时，编码器将输入信息码元 m_k 直接输出；

[0101] (2) 在时钟 i_clk180 的驱动下，S_{md} 与寄存器 D1 进行常系数 GF 域乘加运算，结果存储至寄存器 D0；

[0102] (3) 在时钟 i_clk 的驱动下，S_{md} 同时与寄存器 D2 ～ D31 进行常系数 GF 域乘加运算，结果分别存储至寄存器 D1 ～ D30，并将 S_{md} 存储于寄存器 D31 中；

[0103] (4) 判断信息码元是否输入完毕，如果是，执行步骤 (5)，否则，输入第 k+1 个信息码元 m_{k+1} 到编码器，返回步骤 (1)；

[0104] (5) 所有信息码元计算完成后，断开开关 K2，K1 接 b 口，在时钟 i_clk 的驱动下，串行输出寄存器 D0 ～ D31 的值作为检验码元，则该组信息码元编码完毕；

[0105] 其中，所述的高速 RS(244,212) 编码器的 FPGA 方法采用基于多项式除法的电路，调用了 32 个常系数 GF 域乘加器，通过反相驱动时钟 i_clk180 控制其中一个常系数 GF 域乘加器使达到高速编码的效果，其具体编码时序仿真图如图 7 所示。

[0106] 实现高速 RS(244,212) 译码器的 FPGA 方法如下：

[0107] 本发明所述高速 RS(244,212) 译码器的 FPGA 方法采用双时钟驱动，具有三级流水线结构框图如图 2 所示，包括伴随式计算模块、求解关键方程模块、错误值获取模块（包括钱搜索，错误值计算，能否纠错判断）、信息存储 FIFO、错误值存储 FIFO 以及纠正错误电路，译码后的数据由纠正错误电路输出，其具体译码时序仿真图如图 9 所示。

[0108] 所述的双时钟驱动，在普通 RS 译码的流程上，对硬件实现较复杂的子模块，采用反向时钟驱动，使其达到高速运算的同时，也降低了硬件电路的复杂度。

[0109] 所述的三级流水线结构，其中第一级为伴随计算模块，第二级为求解关键方程模块，第三级为错误值获取模块以及纠正错误电路等，它在很大程度上提高了译码速度，减少了两输入码块之间的间隔时钟周期。

[0110] 所述伴随式计算模块，通过每组 244 个数据码元 $\{c_1, c_2, L, c_{244}\}$ 的串行输入，计算出伴随多项式的系数 $\{S_1, S_2, L, S_{32}\}$，总共分为 32 个子模块，分别对应 32 个伴随多项式的系数，其结果并行输入求解关键方程模块，用于求解错误位置多项式和错误值多项式的系数。

[0111] 所述的求解关键方程模块，根据所输入的伴随多项式系数 $\{S_1, S_2, L, S_{32}\}$ 和 RiBM 算法来进行迭代运算，将得到的错误位置多项式的系数 $\{\sigma_1, \sigma_2, L, \sigma_{32}\}$ 和错误值多项式

的系数 $\{\omega_1, \omega_2, L, \omega_{32}\}$，以及错码数目，并行输入错误值获取模块，用于判断是否译码正确和计算错误位置以及错误值，其中两组多项式的系数之间是串行输出的，参看图 4。

[0112] 所述的错误值获取模块，包括钱搜索模块：将钱搜索法得到错误位置多项式的根 x_i，以及错误值计算中该根相对应的错误值多项式的值 $w(x_i^{-1})$ 的计算，综合于一个模块之中，同时得到错误位置多项式的根 x_i、错误值多项式的值 $w(x_i^{-1})$ 、 $\sigma'(x_i^{-1})x_i^{-1}$ 以及常数项因子 $(x_i^{-1})^{b-1}$，在不影响速度的前提下，大大降低了错误值计算模块所需要的硬件资源和时钟周期；错误值计算模块：采用钱搜索模块得到的相应值计算出错误值。同时错误值获取模块将计算出的错误值，输出给错误值存储 FIFO 中存储起来。另外，钱搜索模块还输出判断使能信息，用于控制纠正错误电路。

[0113] 所述的信息存储 FIFO，用于把输入的数据码元按先后顺序存储到 FPGA 芯片的 FIFO 单元，待错误值获取模块计算完成后，根据错误情况将数据依次输出到纠正错误电路中；

[0114] 所述的错误值存储 FIFO，用于把错误值获取模块的输出数据，按先后顺序存储到 FPGA 芯片的 FIFO 单元中，待获取错误值模块计算完成后，根据错误情况将所计算的错误值依次输出到纠正错误电路中；

[0115] 所述的纠正错误电路，是采用错误值获取模块输出的判断使能信息，控制信息存储 FIFO 和错误值存储 FIFO，当判断译码器已正确译码时，则将两者存储的信息顺序输出，进行 GF 域相加运算，将结果作为已译码字输出，否则，直接将信息存储 FIFO 的信息作为输出。

[0116] 其中，所述的常系数 GF 域乘法器在普通 GF 域乘法器的基础上，确定其中一个变量的值，同时，针对这个确定的变量，对乘法器进行改进，如图 7 所示。

[0117] 其中，所述的常系数 GF 域乘加器使 GF 域乘法和 GF 域加法同时进行，仅在常系数 GF 域乘法器结构的基础上增加了一个异或运算，如图 6 所示。

[0118] 其中，所述的两时钟周期控制的 GF 域乘法器，将普通 GF 域乘法器结构分成两个子结构，分段进行运算。

[0119] 在上述的技术方案中，各部分运算电路均有限域中进行，RS 编译码的复杂度很大程度上取决于基本运算单元，传统的 RS 码编码器中乘法器非常复杂，需要两个多项式相乘，再除以本原多项式取模，其乘法运算不是速度慢就是太复杂，本发明中大部分模块均采用常系数来设计乘法器，或改进而成的常系数 GF 域乘加器，将 GF 域乘法和加法同时进行，均只需要进行少量的异或运算即可，大大降低了乘法器实现的复杂度。对于无法采用常系数 GF 域乘法器的模块，为了提高速度，可采用多个时钟将普通的 GF 域乘法器分块处理，此处采用两个时钟，将 GF 域乘法器分为两块进行处理，仅增加上了一个时钟的时延，就达到了很好的效果。

[0120] 本发明采用双时钟驱动，高速 RS 编译码器的数据通过率高，如用 Xilinx 公司的 xc4vlx160-12ff1148 芯片进行综合仿真以及静态时序分析，其中，编码器最高可工作于 560MHz 的时钟，其数据吞吐量约为 4.48Gbit/s，译码器最高可工作于 370MHz 的时钟，其数据吞吐量约为 2.96Gbit/s；高速 RS 译码器采用三级流水线结构，降低了输入码块之间的间隔时钟周期，同时将钱搜索和错误值的部分计算结合于一个模块，提高了流水线结构的效率；提出三种基本运算单元，常系数 GF 域乘加器，常系数 GF 域乘法器，以及两时钟周期控制

的 GF 域乘法器，几种乘法器结合运用，不仅大大提高了运算速度，还降低了硬件复杂度；整个编译码过程均在一片 FPGA 芯片中实现，工作性能稳定可靠，同时能够容易移植；针对不同的实际应用需求，可以在保持外部接口不变的情况下灵活地修改内部电路设计，以便于和不同的设备进行衔接；不但可以纠正随机错误，而且具有很强的纠正突发错误的功能。

[0121] 本发明针对不同环境均可应用，具有很强的选择性，其用途非常广泛，采用正反双时钟驱动，在很大程序上提高了数据吞吐量，且又避免了消耗资源过大的缺点，使硬件实现的复杂度大幅降低；此外由于在硬件实现中，均采用最简单的硬件单元，非常易于硬件实现。

[0122] (1) 如图 1 所示，是本发明提出的高速 RS 编码器的实现框图，其中，$Mg_1 \sim Mg_{16}$ 分别对应为常系数为 g1 ～ g16 的常系数 GF 域乘加器 CMS(Constant Multiply and Sum)，串行输入的 212 个信息码元 $\{m_1, m_2, L, m_{212}\}$，开关 K1 接 a 口，闭合开关 K2，在时钟 i_clk 的驱动下，输入的第 k 个信息码元 m_k 在使能信号的控制下输入编码器，与寄存器 D0 进行 GF 域相加运算得 S_{md}，k 表示码元序号，初始为 1，同时，编码器将输入信息码元 m_k 直接输出；在时钟 i_clk180 的驱动下，S_{md} 与寄存器 D1 进行常系数 GF 域乘加运算，结果存储至寄存器 D0；在时钟 i_clk 的驱动下，S_{md} 同时与寄存器 D2 ～ D31 进行常系数 GF 域乘加运算，结果分别存储至寄存器 D1 ～ D30，并将 S_{md} 存储于寄存器 D31 中；判断信息码元是否输入完毕，如果是，断开开关 K2，K1 接 b 口，在时钟 i_clk 的驱动下，串行输出寄存器 D0 ～ D31 的值作为检验码元，则该组信息码元编码完毕，否则，输入第 k+1 个信息码元 m_{k+1} 到编码器，继续编码操作；

[0123] (2) 如图 2 所示，是本发明提出的高速 RS 译码器的三级流水线结构框图，第一级为伴随式计算模块，由图 3 所示的伴随式计算子模块组成，第二级为求关键方程模块，其具体流程图如图 4 所示，第三级为错误值获取模块，其中错误值获取模块分为钱搜索模块，如图 5 和 Forney 计算错误值模块，其中，Csi 为伴随多项式模块中第 i 个子模块用到的常系数 GF 域乘法器 CM(Constant Multiplier) 的系数，$Cc_1 \sim Cc_{16}$ 为钱搜索模块用到的常系数 GF 域乘法器的系数，此外，图 5 中的所有加法器都由时钟 i_clk 驱动，所有 CM 都由时钟 i_clk180 驱动，在使能信号为高电平时，信号 $\{\sigma_1, \sigma_2, L, \sigma_{32}\}$ 输入，开关同时置于 a 端，在使能信号为低电平时，开关则同时置于 b 端。

[0124] (3) 如图 6 所示，为节省资源量和降低复杂度，本发明中所采用的常系数 GF 域乘加器，图所示为常系数为 g1 的常系数 GF 域乘加器，如图 7 所示为本发明例中所采用常系数为 21 的常系数 GF 域乘法器，其中 d0 ～ d7 对应为所要进行乘法运算的八比特数据，a0 ～ a7 对应为所要进行加法运算的八比特数据，n0 ～ n7 为所得到的结果的八比特数据。由图可以看出，所采用的均为最简单的异或门，复杂度非常低。

[0125] (4) 如图 8 和图 9 所示，为高速 RS 编码器和高速 RS 译码器的典型时序仿真图，可见其流水线的编译码方式，其中 i_clk 为正向时钟，i_clk180 为反向时钟，i_rst 为该模块的复位信号，i_data_ena 为数据输入的有效使能信号，iv_data 为输入的数据码元，ov_data 为输出的数据码元，o_data_ena 为数据输出的有效使能信号。

[0126] 所述的基本运算单元均在 GF 域内进行。

[0127] 上述步骤描述了本发明的优选实例，显然本领域技术人员通过参考本发明的优选实例和附图可以对本发明做出各种修改和替换，这些修改和替换都应落入本发明的保护范围之内。

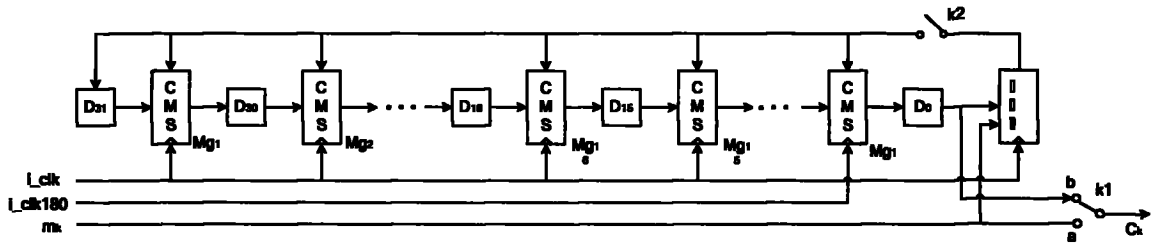

k2
CMS
Mg1
Mg2
b
k1
a

图 1

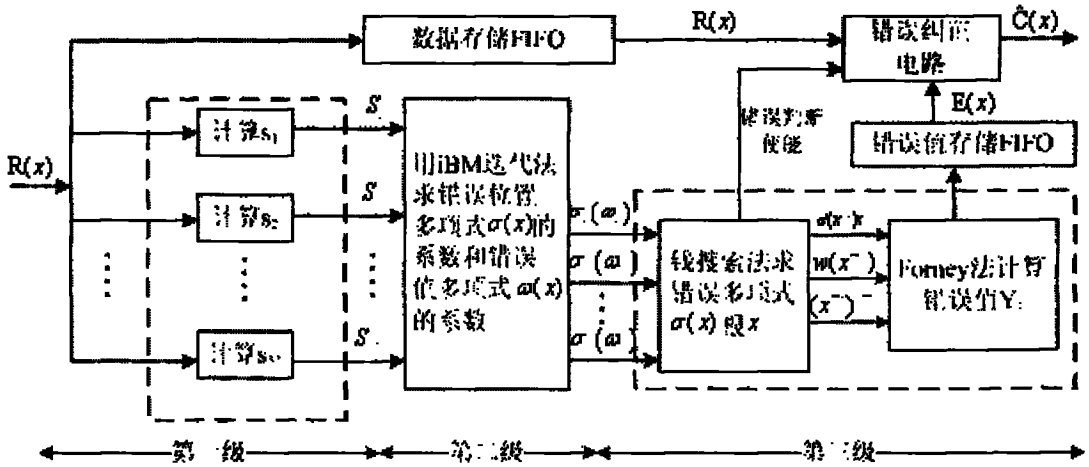

数据存储FIFO
R(x)
错误纠正电路
Ĉ(x)
E(x)
错误判断使能
错误值存储FIFO
S
计算S1
计算S2
计算Sn
利iBM迭代法求错误位置多项式σ(x)的系数和错误值多项式ω(x)的系数
钱搜索法求错误多项式σ(x)根x
Forney法计算错误值Yi
第一级
第二级
第三级

图 2

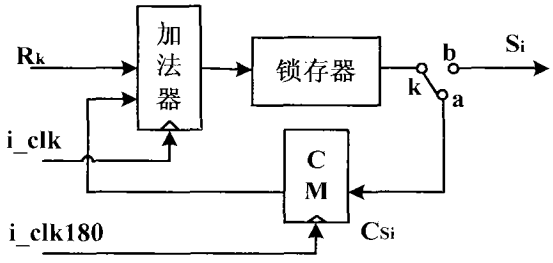

Rk
加法器
锁存器
b
k
a
Si
i_clk
CM
i_clk180
Csi

图 3

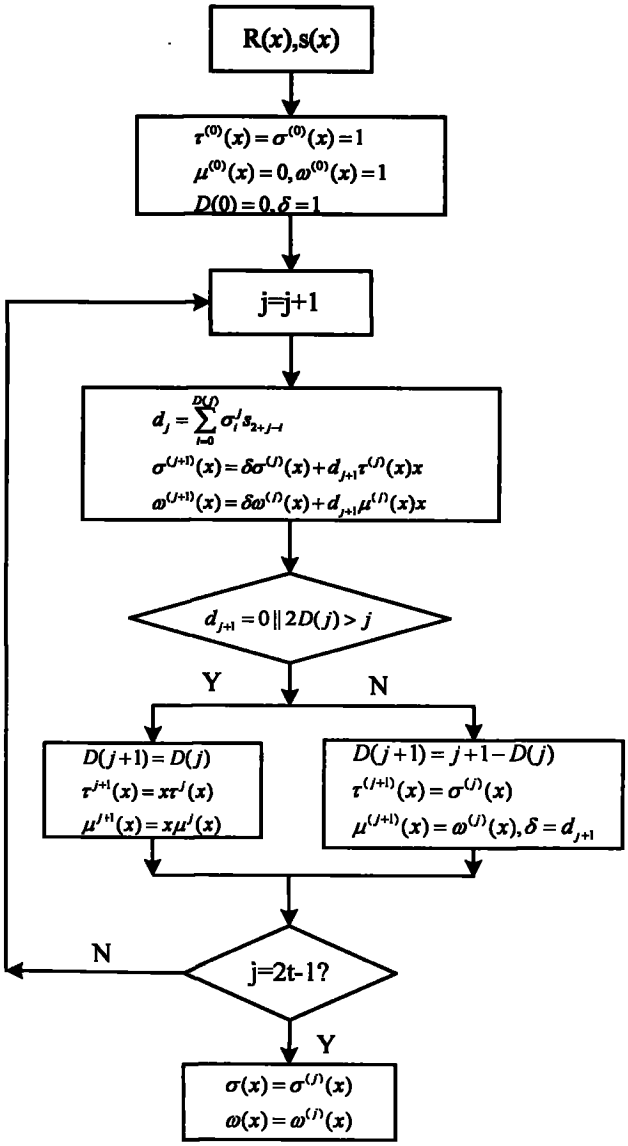

R(x),s(x)
$\tau^{(0)}(x)=\sigma^{(0)}(x)=1$
$\mu^{(0)}(x)=0,\omega^{(0)}(x)=1$
$D(0)=0,\delta=1$
j=j+1
$d_j=\sum_{i=0}^{D(j)}\sigma_i^j s_{2+j-i}$
$\sigma^{(j+1)}(x)=\delta\sigma^{(j)}(x)+d_{j+1}\tau^{(j)}(x)x$
$\omega^{(j+1)}(x)=\delta\omega^{(j)}(x)+d_{j+1}\mu^{(j)}(x)x$
$d_{j+1}=0\|2D(j)>j$
Y
N
$D(j+1)=D(j)$
$\tau^{j+1}(x)=x\tau^j(x)$
$\mu^{j+1}(x)=x\mu^j(x)$
$D(j+1)=j+1-D(j)$
$\tau^{(j+1)}(x)=\sigma^{(j)}(x)$
$\mu^{(j+1)}(x)=\omega^{(j)}(x),\delta=d_{j+1}$
N
j=2t-1?
Y
$\sigma(x)=\sigma^{(j)}(x)$
$\omega(x)=\omega^{(j)}(x)$

图 4

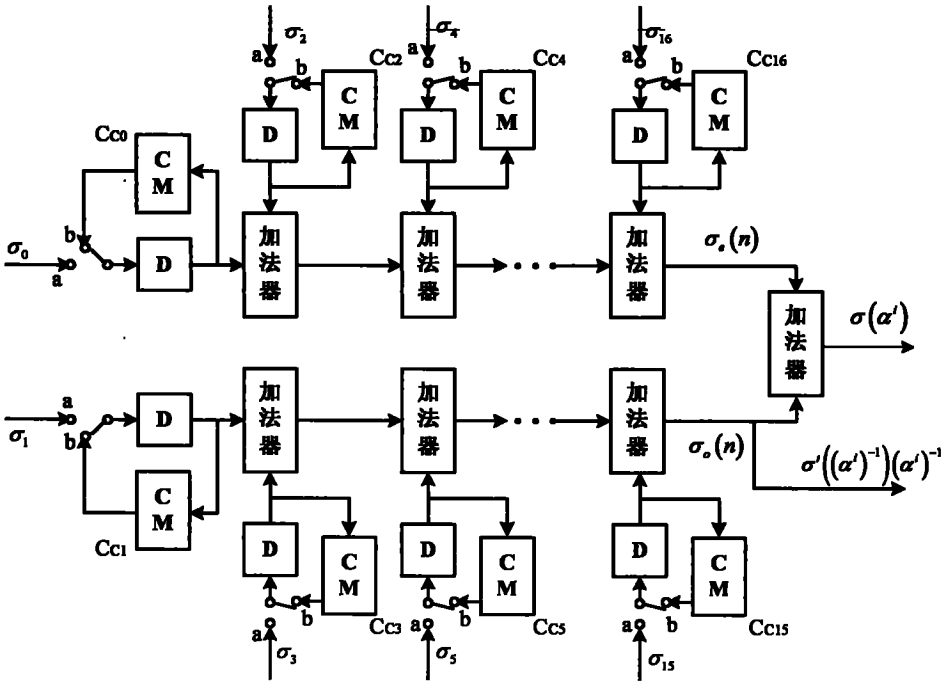

C M
D
加法器
Cc0
Cc1
Cc2
Cc3
Cc4
Cc5
Cc15
Cc16

图 5

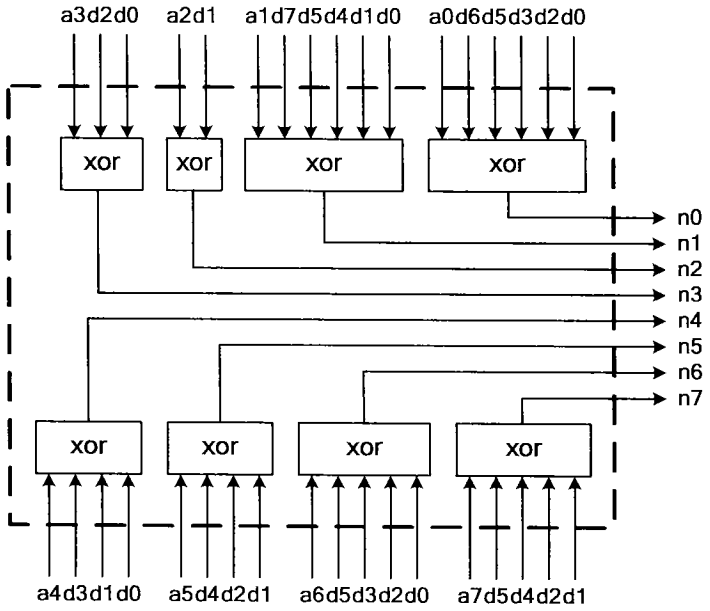

a3d2d0
a2d1
a1d7d5d4d1d0
a0d6d5d3d2d0
xor
n0
n1
n2
n3
n4
n5
n6
n7
a4d3d1d0
a5d4d2d1
a6d5d3d2d0
a7d5d4d2d1

图 6

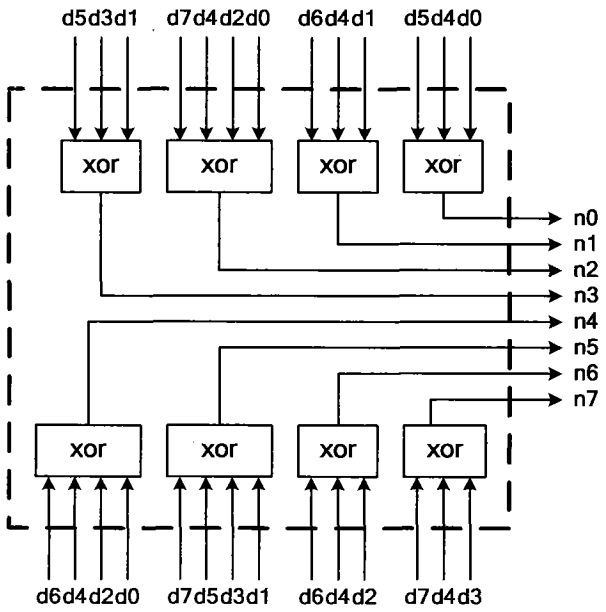

d5d3d1
d7d4d2d0
d6d4d1
d5d4d0
xor
xor
xor
xor
n0
n1
n2
n3
n4
n5
n6
n7
xor
xor
xor
xor
d6d4d2d0
d7d5d3d1
d6d4d2
d7d4d3

图 7

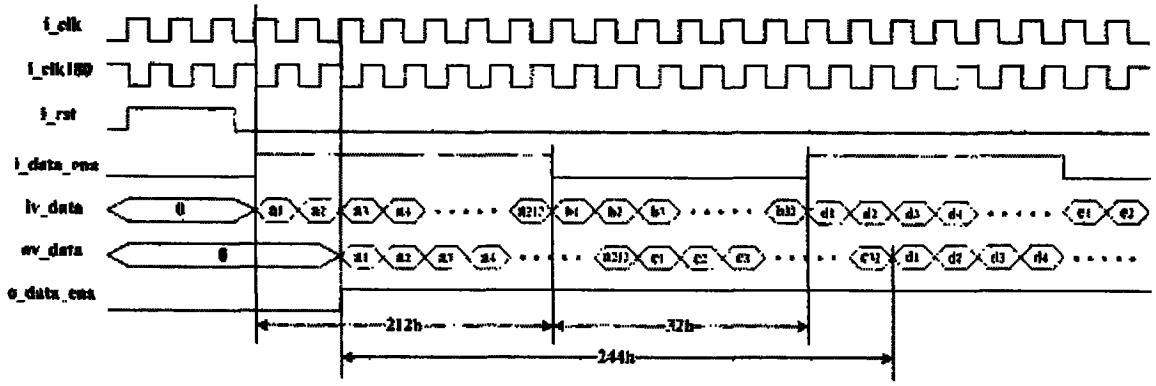

i_clk
i_clk180
i_rst
i_data_ena
iv_data
ov_data
o_data_ena
212b
32b
244b

图 8

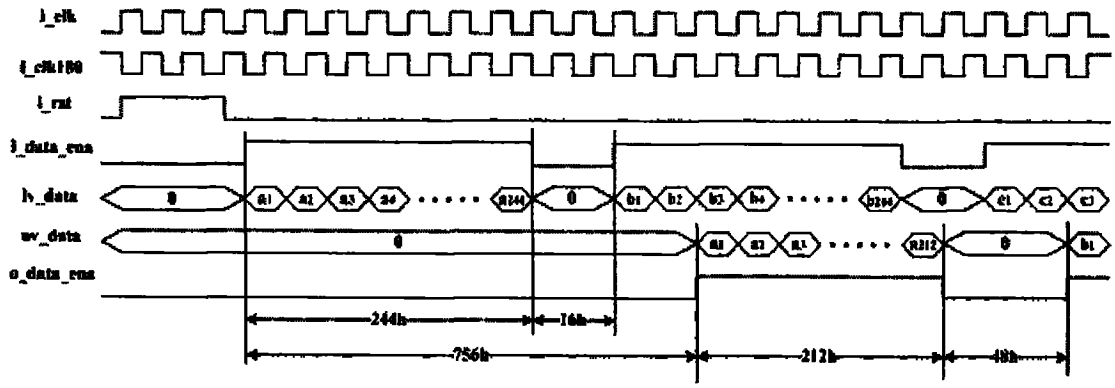

i_clk
i_clk180
i_rst
i_data_ena
iv_data
ov_data
o_data_ena
244b
16b
756b
212b
48b

图 9