截短 Reed-Solomon 码译码器的 FPGA 实现

张 玲,张 立,何 伟
(重庆大学 通信工程学院,重庆 400030)

摘 要:提出了一种改进的 BM 算法,并在此基础上提出了一种大量采用并行结构的截短 RS 码译码器的实现方式。验证表明,该算法能显著提高基于 FPGA 的 RS 译码器的速度并简化其电路结构。

关键词:RS 译码器;关键方程;BM 算法;FPGA;并行结构
中图分类号:TN919 **文献标志码**:A

An implementation for shortened Reed-Solomon decoder on FPGA

ZHANG Ling, ZHANG Li, HE Wei
(Department of Communication Engineering, Chongqing University, Chongqing 400030, China)

Abstract: This paper discusses an algorithm for RS decoder, especially an improved BM(berlekamp-massey) method for the KE (key equation) solution is introduced,which can simplify the decoding process to a great extent. Based on this algorithm,the paper presents an architecture for its implementation on FPGA,in which parallel structures are largely used in order to achieve the speed as high as possible.

Key words: RS decoder; key equation; BM method; FPGA; parallel structure

RS 码(Reed-Solomon)是一种有很强纠错能力和很高编码效率的多进制 BCH 码,特别是纠正突发错误的能力很强,并且其码字构造简单,有严格的代数结构,便于用硬件实现,目前已广泛应用于通信领域中。RS 码译码算法中对关键方程的求解是最重要和耗时最多的部分,采用传统 BM 算法[1,2]来求解,其结构较为复杂,不利于用硬件实现。对此,本文提出了一种改进的 BM 算法,该算法便于用循环方式实现,能有效简化 RS 译码器的结构,并在此基础上提出了一种大量采用并行结构的截短 RS 码译码器的硬件电路实现方式,相比软件译码方式,其速度优势十分明显,适合应用于嵌入式系统中。

1 RS 码的解码算法流程及实现结构

RS 码是一种基于有限域(如 GF,Galois field,伽罗华域)的纠错码,其码字由两部分组成:数据码字和纠错码字。其中数据码字包含了有用的信息;纠错码字则是附加在数据码字之后,相当于附加的信息,用于对数据码字可能出现的错误的检测和纠正,码流所包含的总码字数就是数据码字与纠错码字个数之和。

RS 解码算法包括四个步骤:(1)根据接收码字计算伴随式;(2)求解关键方程[1];(3)用钱搜索[1]方法寻找错误位置,用 Forney 方法[1]计算错误值;(4)将错误位置的接收码字与错误值相加,得到纠错后的码字。其总体流程如图 1 所示。需要说明的是,解码过程中的计算都是基于有限域的,其基本计算单元是有限域的乘法器和加法器[3,4]。

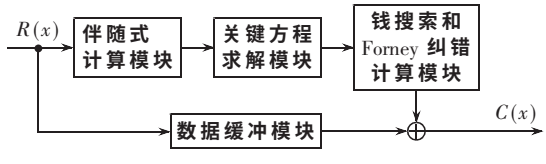

图 1 RS 码解码流程总体框图

1.1 伴随式的计算及电路结构实现

计算伴随式是译码算法的第一步,需要计算的伴随式个数是码流的可纠错码字数 t 的 2 倍。设发送多项式为 $C(x)$,码流在传输过程中产生的错误多项式为 $E(x)$,则接收多项式可表示为:

$$R(x)=C(x)+E(x)=\sum_{i=0}^{n-1} Y_i x_i \tag{1}$$

为了确定 $E(x)$,需要得到错误位置 x_i 及错误值 Y_i。设具有 (n,m,t) 形式的 RS 码所在伽罗华域的本原元为 α,其 $2t$ 个伴随式 $(S_1,S_2,\cdots,S_{2t})$ 是通过把 $\alpha^i(i\in 2t)$ 代入

到接收多项式 $R(x)$ 中计算而得到。

在实现时，伴随式 S 的计算可以用如下公式实现[5]：

$$s_i=R(\alpha^i)=\sum_{j=0}^{n-1}R_j\alpha^{ij}=(\cdots((R_{n-1}\alpha^i+R_{n-2})\alpha^i+R_{n-3})\alpha^i+\cdots+R_1)\alpha^i+R_0 \tag{2}$$

由式(2)可以看出，该计算式的结构非常规则，其基本单元是 $R_{n-k}\alpha^i+R_{n-k-1}$，其中 R_{n-k} 就是 RS 码流中的第 $n-k$ 个码字，以计算第 i 个伴随式 s_i 为例，这种循环迭代的实现结构如图 2 所示。

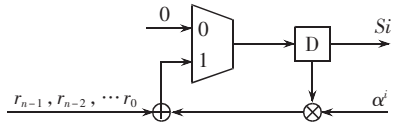

图 2 伴随式计算电路的结构

为了提高速度，本设计采用了 8 个并行的计算模块，分别计算所需要的 8 个伴随式，每次计算均在 1 个时钟周期内完成。这样，经过 26 个时钟周期后，8 个伴随式的值即可并行输出，传递给关键方程求解模块。

1.2 改进 BM 算法对关键方程的求解及电路实现结构

关键方程是以 $2t$ 个伴随式为系数构成的一个齐次线性方程组：

$$\begin{bmatrix} St+1 & St & St-1 & \vdots & S1 \\ St+2 & St+1 & St & \vdots & S2 \\ St+3 & St+2 & St+1 & \vdots & S3 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ S2t & S2t-1 & S2t-2 & \vdots & St \end{bmatrix}\begin{bmatrix}1\\ \sigma1\\ \sigma2\\ \cdots\\ \sigma t\end{bmatrix}=\begin{bmatrix}0\\0\\0\\ \cdots\\0\end{bmatrix} \tag{3}$$

记：

$$\Sigma=[1\quad \sigma1\quad \sigma1\quad \cdots\quad \sigma t]^{\mathrm{T}} \tag{4}$$

则对关键方程的求解，就是设法求得由错误位置多项式 $\sigma(x)$ 的各项系数构成的向量 Σ。笔者提出的改进 BM 算法相比传统的 BM 算法而言，其优点是便于采用循环迭代结构实现，硬件实现复杂度减小，而计算效率却得以提高。

改进 BM 算法的流程可以表示为：

初始化：$\Lambda^{(0)}(x)=1$，$B^{(0)}(x)=1$，$L_0=0$

下面的迭代流程用于计算 $\sigma(x)$：

$$\Delta_r=\sum_{j=0}^{r-1}\Lambda_j^{(r-1)}S_{r-j} \tag{5}$$

$$\delta_r=\begin{cases}1, \text{if}(\Delta_r\neq 0 \text{ and } 2L_r\leqslant r-1)\\ 0, \text{others}\end{cases} \tag{6}$$

$$L_r=\delta_r\times(r-L_{r-1})+(1-\delta_r)\times L_{r-1} \tag{7}$$

$$\begin{bmatrix}\Lambda^{(r)}(x)\\ B^{(r)}(x)\end{bmatrix}=\begin{bmatrix}1 & \Delta_r\times x\\ \delta_r\times\Delta_r^{-1} & (1-\delta_r)\times x\end{bmatrix}\times\begin{bmatrix}\Lambda^{(r-1)}(x)\\ B^{(r-1)}(x)\end{bmatrix} \tag{8}$$

$r=1,2,\cdots 2t$，经过 $2t$ 次迭代，可以得到：

$$\Lambda(x)=\Lambda^{(2t)}(x)=1+\Lambda_1x+\Lambda_2x+\cdots+\Lambda_tx^t \tag{9}$$

其中，$\Lambda(x)$ 即为前文所述的错误位置多项式 $\sigma(x)$，Δ_r、$B(x)$ 以及 L 均为中间变量，Δ_r^{-1} 表示 Δ_r 的倒数，或称为逆元素，$\Lambda^{(r-1)}(x)$ 表示上一次迭代中得到的 $\Lambda(x)$ 的值。有限域上求逆的计算可以通过 LUT 来实现，用一个片上 ROM 事先存入所有元素的逆值，根据输入元素的值，在 ROM 输出端给出对应的逆值。

在计算出错误位置多项式 $\sigma(x)$（即图 3 中所示 $\Lambda(x)$）之后，关键方程求解模块将利用它的值来计算错误值多项式 $\omega(x)$ 的值，$\omega(x)$ 的系数 ωi 为：

$$\omega i=\sum_{j=0}^{i}Sj\sigma i-j \tag{10}$$

在实现关键方程求解电路时，采用了划分时钟的方式，将系统时钟 sys_clock 依次划分为 pclk0、pclk1、…、pclk4，共 5 个节拍。其中 pclk0 和 pclk1 用于驱动 Δ_r 计算模块；pclk2 用于驱动 Δ_r 求逆模块；pclk3 用于判断 δ_r 和 L 的值，并寄存上一次迭代中变量 $B(x)$ 的值以用于后面的计算；pclk4 则用于计算错误位置多项式 $\Lambda(x)$ 以及中间变量 $B(x)$ 的值，该模块需要将错误位置多项式的值反馈到输入端以进行迭代计算。

1.3 钱搜索和 Forney 算法及电路实现结构

求出向量 Σ 后，由钱搜索方法得出错误个数和错误位置，最后使用 Forney 算法将错误值与错误码字相加，得到正确的码字。

对于一段在 $GF(2^m)$ 域上的 RS 码流，若其包含码字为 2^m-1 个，则称该码流为非截短 RS 码，否则称之为截短 RS 码[4]。以基于 $GF(2^8)$ 的 RS 码为例，在该伽罗华域

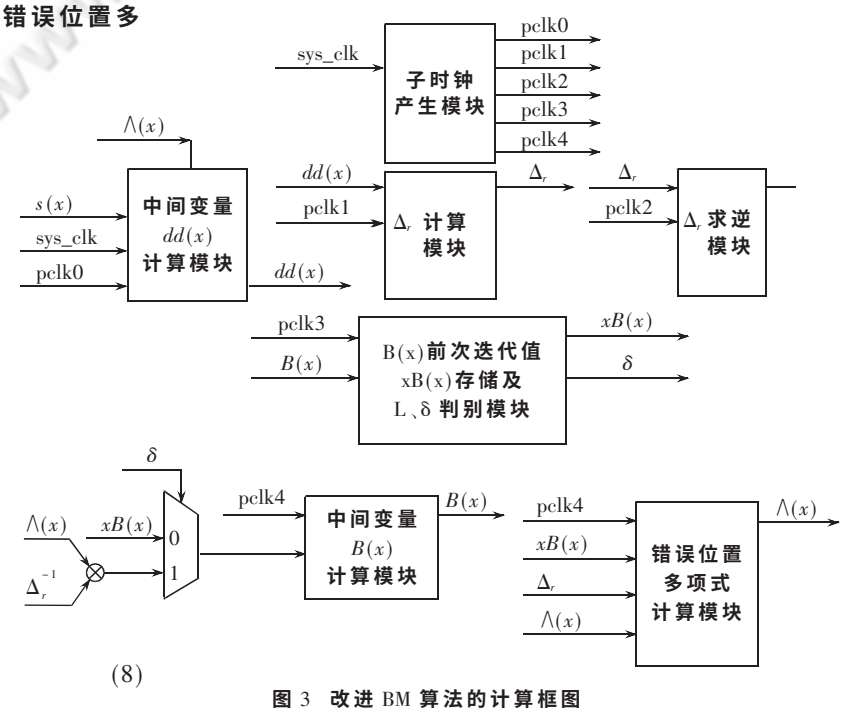

图 3 改进 BM 算法的计算框图

上的非截短 RS 码流应该包含 255 个码字，这里所讨论的(26,16,4)型码只有 26 个码字,故为截短码。

对于包含 j 个码字的截短 RS 码,可以将其理解为包含 j 个有用码字和 α^m-1-j 个 0 元素的非截短 RS 码。设非截短 RS 码为(n,k),当缩短 p 个码字时,可表示为$(n-p,k-p)$,则该截短码的接收序列可以表示为:

$$r(x)=r_{n-p-1}x^{n-p-1}+r_{n-p-2}x^{n-p-2}+\cdots+r_1x+r_0 \tag{11}$$

在对截短 RS 码应用钱搜索方法时，检查该码流中的第 l 位码字是否出错,实际上就是计算 $\alpha^{-(n-p-1)}=\alpha^{(p+1)}$ 是否为错误位置多项式 $\sigma(x)$的根[4]。

如果第 r_{n-l} 位码字有错,则根据 Forney 算法,其差错值 Y_{n-l} 按如下公式计算:

$$\begin{aligned}Y_{n-l}&=\frac{\omega(\alpha^{p+l})}{\alpha^{p+l}\times\sigma'(\alpha^{p+l})}\\&=\frac{1+\omega_1\alpha^{p+l}+\omega_2\alpha^{2(p+l)}+\cdots+\omega_t\alpha^{t(p+l)}}{\alpha^{p+l}\times(\sigma_1+\sigma_3\alpha^{2(p+l)}+\cdots+\sigma_{t-1}\alpha^{(t-2)(p+l)})}\\&=\frac{1+\omega_1\alpha^{p+l}+\omega_2\alpha^{2(p+l)}+\cdots+\omega_t\alpha^{t(p+l)}}{\sigma_1\alpha^{p+l}+\sigma_3\alpha^{3(p+l)}+\cdots+\sigma_{t-1}\alpha^{(t-1)(p+l)}}\end{aligned} \tag{12}$$

观察式(12)可以发现,其分子和分母多项式的结构相似,将每一项系数计算出之后,将各个系数相加,便可得出 $\omega(\alpha^{p+l})$和 $\alpha^{p+l}\times\sigma'(\alpha^{p+l})$这两个结果。以计算 $\omega(\alpha^{p+l})$为例,其电路结构如图 4 所示。由于有限域上的加法运算是按位异或,不存在进位问题,所以图 4 所示 5 个计算单元可以并行计算,最后同时将计算结果送至输出端的加法器。

2 RS 译码器实现结果的验证及分析

在 FPGA 上进行实现之前，将设计所用到的解码算法在 matlab6.5 上进行了验证。结果表明,对包含 4 个错误码字的(26,16,4)截短 RS 码可以进行准确无误的差错和纠错。为了验证在 FPGA 上实现的结果是否正确,在 Quartus Ⅱ 6.0 的波形仿真文件中输入了同样的码流,如图 5 所示,其中用椭圆所圈为错误码字。

在 Quartus Ⅱ 6.0 环境下的时序仿真结果显示，该译码器伴随式计算模块耗时不超过 30 个时钟周期，而关键方程求解模块消耗的时间最多,在 100 个时钟周期左右;钱搜索和 Forney 纠错模块同时进行,其总消耗时间约 50 个时钟周期。由此也可以看出,关键方程的求解是决定 RS 译码器速度的最重要步骤。系统时钟频率设为 100 MHz,经过 184 个时钟周期(1.84 μs)后可得到译码结果,与软件译码方式相比,其耗时非常少。图 6 中的 judge 信号显示了输出码字对应于输入码字的错误位置，当该信号为高电平时表示对应位置的输入码字有错,并进行了纠正。

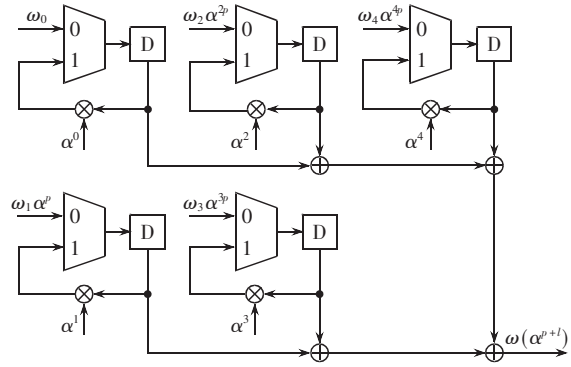

图 4 计算 $\omega(\alpha^{p+l})$的电路结构

该方案已经应用到了基于 Altera Cyclone Ⅱ 型 FPGA 的二维条码识读器中。

参考文献

[1] 张小平.RS 码迭代译码算法分析[J].现代电子技术,2005,28(2):95-98.

[2] 王进祥,张乃通,来逢昌,等.流水线结构 RS(255,233)译码器的 VLSI 设计[J].计算机研究与发展,2000,37(1):61-65.

[3] 刘志伟,邹雪城,刘政林,等.串行迭代结构在 RS 解码器设计中的应用[J].计算机与数字工程,2003,31(5):36-40.

[4] 王伟,徐辉,邹火儿.一种 RS 码译码器的硬件实现方法[J].无线电通信技术,2004,30(1):9-10.

[5] 张国华,王菊花.一种高速 Reed-Solomon 译码器的实现结构与 Simulink 建模仿真[J].空间电子技术,2004,1(2):17-22.

图 5 输入码字

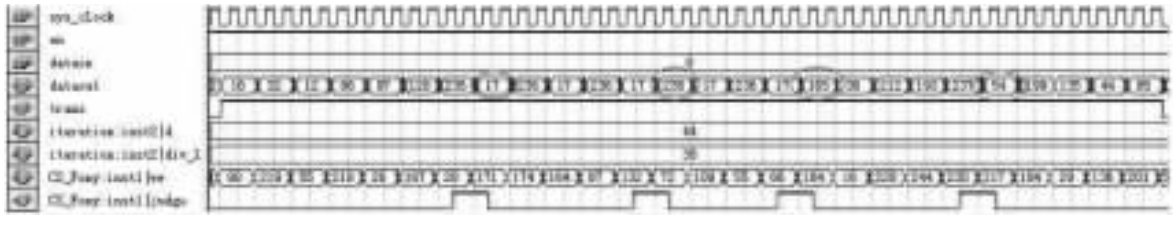

图 6 在 Quartus Ⅱ 6.0 中仿真得到的纠错之后的码字

(收稿日期:2008-12-16)