

基于 IEEE802.16e 的 LDPC 编码器设计与实现

陈志凯, 韩泽耀

(上海交通大学 微电子学院, 上海 200240)

摘要: 提出了一种基于 IEEE802.16e 的具有线性编码复杂度的 LDPC 码的硬件编码器结构, 并且在 TSMC 的 0.18 μm 工艺库的最恶劣情况下, 通过 Design Compiler 工具综合可以达到 385MHz 的速度。

关键词: LDPC 码 IEEE802.16e 编码器 硬件实现

低密度奇偶校验码 LDPC 码作为前向纠错码广泛应用于通信和存储领域。最近, LDPC 码被 IEEE802.16e 标准选择作为其信道编码技术^[1]。LDPC 码在码长足够长时性能接近香农限^[2]。LDPC 码比 Turbo 码有更好的渐进特性, 并且拥有一类广阔的译码算法——消息传递算法, 使其能够在编码的性能与复杂度之间折中选择。但是对于 LDPC 码来说, 一个需要面对的主要问题是来自于自身表面的高编码复杂度, LDPC 码编码器具有和码长成二次方的时间复杂度, 而 Turbo 码则可在线性时间内编码。

Richardson 和 Urbanke 提出了一种新的编码算法^[3], 在很大程度上减轻了随机构造的 LDPC 码在编码上的巨大运算量需求, 使得 LDPC 码编码器能够在和码长成线性关系的时间内编码。

本文基于 Richardson 和 Urbanke 提出的编码算法, 针对 IEEE802.16e 中对 LDPC 码校验矩阵的规定, 提出了一种特定的实现 LDPC 编码器的方法。

1 基于 IEEE802.16e 的 LDPC 码的快速编码算法

1.1 IEEE802.16e 中对 LDPC 码校验矩阵的约定

LDPC 码由 $m \times n$ 的矩阵 H 定义, 其中 m 是校验位的位数, n 是码长的位数。各种码长和码率的校验矩阵 H 都是由基本矩阵 H_b 膨胀得到的, 其膨胀因子 $z_f = n/24$ 。在基本矩阵 H_b 中, 用 -1 表示 $z_f \times z_f$ 的全 0 矩阵, 用一个循环移位因子 $p(i, j)$ 表示单位阵循环移位后的矩阵。每种码率的基本矩阵 H_b 划分为如下三部分:

$$H_b = [(H_{b1})_{m_b \times K_b} \mid (h_b)_{m_b \times 1} \mid (H_{b2})_{m_b \times (m_b - 1)}]$$

其中, H_{b1} 对应的是信息位, h_b 和 H_{b2} 对应的是校验位。 H_b 都是针对最大的码长 ($n=2304$) 定义的。

基本矩阵中的移位集 $\{p(i, j)\}$ 用来决定相同码率不同码长的移位大小, 对于码率为 1/2、3/4A、3/4B、2/3B 和 5/6 的各种码, $p(i, j)$ 由下式决定:

$$p(i, j) = \begin{cases} p(i, j) & p(i, j) \leq 0 \\ \left[\frac{p(i, j)z_f}{z_0} \right] & p(i, j) > 0 \end{cases}$$

而对于码率为 2/3A 的各种码, $p(i, j)$ 由下式决定:

$$p(i, j) = \begin{cases} p(i, j) & p(i, j) \leq 0 \\ \text{mod}(p(i, j)z_f) & p(i, j) > 0 \end{cases}$$

1.2 基于 IEEE802.16e 的 LDPC 码的快速编码算法

为了对 LDPC 码进行有效编码, 矩阵 H_b 被分割成如下形式:

$$H_b = \begin{pmatrix} A_{(m_b-1) \times K_b} & B_{(m_b-1) \times 1} & T_{(m_b-1) \times (m_b-1)} \\ C_{1 \times K_b} & D_{1 \times 1} & E_{1 \times (m_b-1)} \end{pmatrix}$$

设膨胀因子为 z , 码字 $x = (k_b, p_1, p_2)$, 其中 $k_b = (u_1, u_2, \dots, u_{k_b})$, $p_1 = (p_1)_{1 \times z}$, $p_2 = (v_1, v_2, \dots, v_{m_b-1})$, 而 $u_i (i=1, \dots, k_b)$ 、 $v_j (j=1, \dots, m_b-1)$ 为 $1 \times z$ 向量。

又设矩阵 $F = \begin{pmatrix} I & 0 \\ -ET^{-1} & 0 \end{pmatrix}$, 由于 LDPC 码都有:

$$H_b x^T = 0 \quad (1)$$

对 (1) 式两边左乘矩阵 F 可得:

$$FH_b x^T = 0 \Rightarrow \begin{pmatrix} A & B & T \\ -ET^{-1}A + C & -ET^{-1}B + D & 0 \end{pmatrix} \begin{pmatrix} k_b^T \\ p_1^T \\ p_2^T \end{pmatrix} = 0$$

$$\Rightarrow \begin{cases} Ak_b^T + Bp_1^T + Tp_2^T = 0 \\ (-ET^{-1}A + C)k_b^T + (-ET^{-1}B + D)p_1^T = 0 \end{cases}$$

又由于 IEEE802.16e 标准中的 LDPC 码的校验矩阵 $(-ET^{-1}B + D) = 1$ 总是成立, 所以可以得到:

$$p_1^T = (ET^{-1}A + C)k_b^T \quad (2)$$

$$p_2^T = T^{-1}(Ak_b^T + Bp_1^T) \quad (3)$$

由式 (2)、(3) 可知, 其编码过程可以分为如下六个步骤:

- (1) 计算 $f_1 = Ak_b^T$ 和 $f_2 = Ck_b^T$;
- (2) 计算 $f_3 = T^{-1}f_1$ 和 $f_4 = Ef_3$;
- (3) 计算 $p_1 = f_4 + f_2$;
- (4) 计算 $f_5 = Bp_1^T$;
- (5) 计算 $f_6 = f_1 + f_5$;
- (6) 计算 $p_2 = T^{-1}f_6$ 。

由上述分析可知, LDPC 编码算法的计算量主要在于计算 $f_1 = Ak_b^T$, 而 A 是稀疏矩阵并且矩阵中的元素不

是0矩阵就是单位阵循环移位后的矩阵,所以 $f_1=Ak_b^T$ 可以通过将对应的 k_b^T 循环移位得到,计算复杂度和码长成线性关系。 f_2, f_3 可以用同样方法得到。

对于计算 $f_3=T^{-1}f_1$ 和 $p_2=T^{-1}f_6$,本文采用前向置换的方法得到,下面以计算 $f_3=T^{-1}f_1$ 为例推导计算方法(注意推导是针对二进制而言的)。

设 $f_3=(u_1, u_2, \dots, u_{m_b-1})$, $f_1=(v_1, v_2, \dots, v_{m_b-1})$, 因 IEEE802.16e 标准中的 LDPC 码校验矩阵 T 都是双对角阵。则有:

$$f_3 = T^{-1}f_1 \Rightarrow Tf_3 \Rightarrow f_1 \Rightarrow \begin{cases} u_1 = v_1 \\ u_1 + u_2 = v_2 \\ u_2 + u_3 = v_3 \\ \vdots \\ u_{m_b-2} + u_{m_b-1} = v_{m_b-1} \end{cases}$$

$$\Rightarrow \begin{cases} u_1 = v_1 \\ u_2 = v_2 + v_1 \\ u_3 = v_3 + v_2 + v_1 \\ \vdots \\ u_{m_b-1} = v_{m_b-1} + \dots + v_3 + v_2 + v_1 \end{cases}$$

这样,计算 $f_3=T^{-1}f_1$ 和 $p_2=T^{-1}f_6$ 在硬件上就可以简单用‘异或门’来实现。

2 LDPC 编码器硬件结构和具体实现方法

硬件编码器应按照前面的步骤计算 p_1 和 P_2 ,为了使编码器有最快的速度和吞吐量,本设计充分利用了并行执行和硬件流水的方法。硬件编码器的总体结构如图1所示。

本设计把编码过程划分为两级流水的主要依据是平衡各级的执行时间,使得两级的空闲时间尽量小。编码器主要是针对码长 $n=2304$ 、码率 $r=0.5$ 的 LDPC 码设计的,但也适用于其他码率和码长。码长为2304、码率为0.5的 LDPC 码校验矩阵如图2所示,图中-1表示 96×96 全0矩阵,其他元素表示 96×96 的单位阵循环移位后的矩阵,元素值表示循环移位值。

根据前面所述的矩阵分割方法,矩阵 A 有最多的非0元素,所以计算 $f_1=Ak_b^T$ 花的时间最多;而 B 、 C 、 D 、 E 所含的非0元素很少,所以计算 $f_2=Ck_b^T$ 、 $f_4=Ef_3$ 、 $f_5=Bp_1^T$ 花的时间最少;而对于前向置换操作,由于 T 矩阵的维

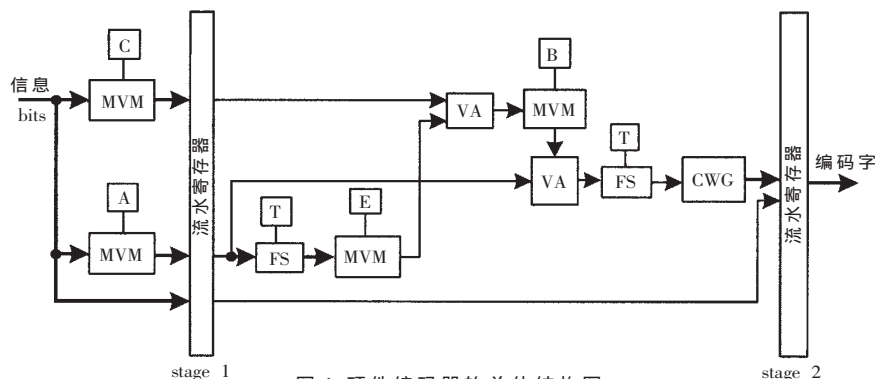


图1 硬件编码器的总体结构图

-1	94	73	-1	-1	-1	-1	55	83	-1	-1	70	-1	-1	-1	-1	-1	-1	-1	-1
-1	27	-1	-1	-1	22	79	9	-1	-1	-1	12	-1	00	-1	-1	-1	-1	-1	-1
-1	-1	-1	24	22	81	-1	33	-1	-1	-1	0	-1	00	-1	-1	-1	-1	-1	-1
61	-1	47	-1	-1	-1	-1	65	25	-1	-1	-1	-1	00	-1	-1	-1	-1	-1	-1
-1	-1	39	-1	-1	-1	84	-1	-1	41	72	-1	-1	-1	-1	00	-1	-1	-1	-1
-1	-1	-1	46	40	-1	82	-1	-1	-1	79	0	-1	-1	-1	00	-1	-1	-1	-1
-1	-1	95	53	-1	-1	-1	-1	14	18	-1	-1	-1	-1	-1	-1	00	-1	-1	-1
-1	11	73	-1	-1	-1	2	-1	47	-1	-1	-1	-1	-1	-1	-1	00	-1	-1	-1
12	-1	-1	-1	83	24	-1	43	-1	-1	-1	51	-1	-1	-1	-1	-1	00	-1	-1
-1	-1	-1	-1	94	-1	59	-1	-1	70	72	-1	-1	-1	-1	-1	-1	-1	00	-1
-1	-1	7	65	-1	-1	-1	-1	39	49	-1	-1	-1	-1	-1	-1	-1	-1	-1	00
43	-1	-1	-1	-1	66	-1	41	-1	-1	-1	26	7	-1	-1	-1	-1	-1	-1	-1

图2 码长 $n=2304$ 码率 $R=0.5$ 的 LDPC 校验矩阵

数比较大,所以计算 $f_3=T^{-1}f_1$ 、 $p_2=T^{-1}f_6$ 花的时间也会相对较多。因此根据以上分析,把计算 $f_1=Ak_b^T$ 和 $f_2=Ck_b^T$ 划在第一级,把剩余部分划在第二级。

编码器的几个主要部件的功能是矩阵乘法(MVM)、前向置换(FS)、矢量加法(VA)和码字生成(CWG)。下面主要介绍矩阵乘法和前向置换。

2.1 矩阵乘法(MVM)

由于 IEEE802.16e 标准中的 LDPC 码校验矩阵的特殊性,矩阵乘法可以用桶形移位器(barrel shifter)来实现,同时要充分利用校验矩阵的稀疏性,以减少计算量。矩阵乘法结构如图3所示。

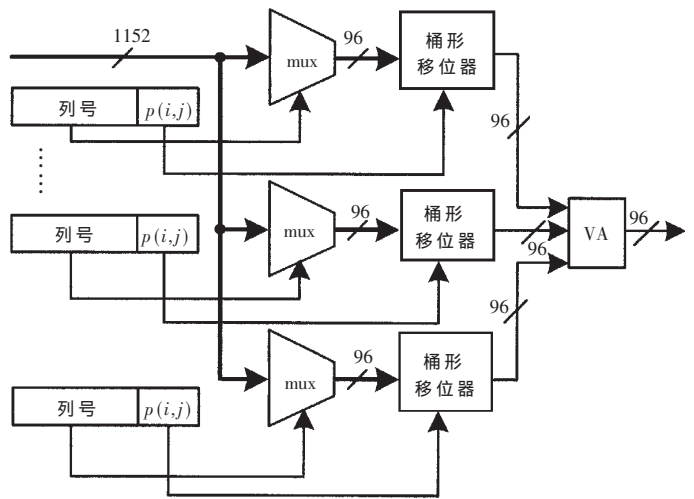


图3 矩阵乘法结构图

在计算矩阵乘法时,编码器按行并行执行。校验矩阵中的每个非0元素都按行分别存储在一个11位的寄存器中,其存储格式如图4所示。

寄存器的第0位~第6位共7位用来存储校验矩阵中的循环移位值,第7

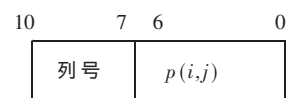
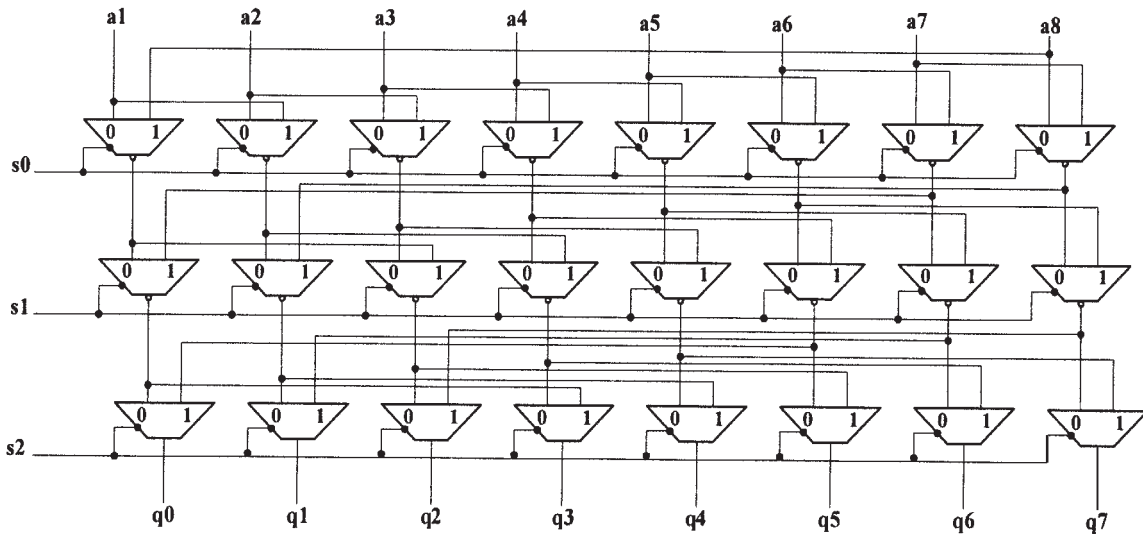


图4 校验矩阵非0元素存储格式

图5 $n=8$ 桶形移位器结构

位~第10位共4位用来存储校验矩阵中非0元素所在列的列号。因为LDPC码的校验矩阵是稀疏矩阵,矩阵中大部分都是0,所以通过这种方法能够高效地存储矩阵。如果LDPC码编码器挂在一个SOC中,软件员可以通过配置寄存器的值使得编码器能够适应各种码长和码率的编码。

通过这样的存储机制,在执行矩阵某一行及信息位乘法时,并行执行这一行中的非0元素和信息位相乘,其过程是:首先利用寄存器的高4位选择信息位中相应的矢量,之后把它输入到桶形移位器中,通过寄存器中的低7位控制移位位数,就完成了一次非0元素和信息位相乘操作;然后把这一行中的所有非0元素和信息位相乘所得结果做一次‘异或’操作(即矢量相加VA),这样就完成了矩阵一行和信息位相乘。编码器中矩阵乘法行和行之间是并行执行的。

在本设计的矩阵乘法模块中,桶形移位器的设计是矩阵乘法设计中最关键的部分。由于Synopsys公司的DesignWare库中的桶形移位器最优的最大位数是64位,对于96位的桶形移位器则没有优势。本设计基于TSMC公司0.18 μm 工艺库设计的桶形移位器是通过把移位器中的主要元件MUX指定映射为MXI2元件并且采用编码的方式进一步减小移位器的延时,使得 n 位的移位器延时为 $\lceil \log_2 n \rceil \times T_{MXI2}$ 。其中, $\lceil x \rceil$ 表示不小于 x 的最小整数, T_{MXI2} 表示MXI2门的延时。

图5是8位桶形移位器结构,对于 $n=96$ 的桶形移位器也有相同的结构,不过 $n=96$ 的移位器的级数为7级。

基于上述结构设计的96位桶形移位器通过design compiler综合以后比基于DesignWare的桶形移位器的设计有5%的速度优势。

2.2 前向置换(FS)

考虑等式 $f_3 = T^{-1}f_1$,其中, T 是双对角的稀疏矩阵, f_3

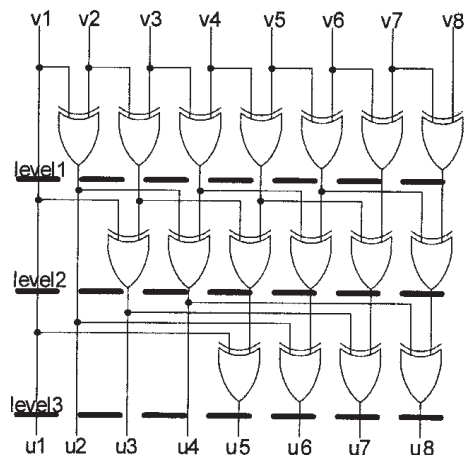
和 f_1 为矢量,设 $f_3 = (u_1, u_2, \dots, u_{m_b-1})$, $f_1 = (v_1, v_2, \dots, v_{m_b-1})$ 。

解决这个问题的一种比较直接的方法是先求矩阵的逆矩阵,然后计算。但矩阵的逆运算是一个很复杂的运算,需要大量的处理时间,而且求逆矩阵破坏了矩阵的稀疏性。

本文采用前面所提出的前向置换的算法,计算只需要‘异或’操作就可以实现,如下式:

$$\begin{cases} u_1 = v_1 \\ u_1 = v_2 \oplus v_1 \\ u_3 = v_3 \oplus v_2 \oplus v_1 \\ \vdots \\ u_{m_b-1} = v_{m_b-1} \oplus \dots \oplus v_3 \oplus v_2 \oplus v_1 \end{cases}$$

但是,如果不加以改进直接使用‘异或’门来实现,将会引入 $(m_b-1) \times T_{2\text{input_xor}}$ 的延时,而且这个延时在流水线的两级都存在(矢量加法也是‘异或’门实现的),这将极大地影响整个编码器的速度。经测试,前向置换操作和矢量加法运算引入的延时占总延时的56.54%。因此本编码器设计了一种编码‘异或’树,使得实现的延时只有 $\log_2(m_b-1) \times T_{2\text{input_xor}}$ 。图6是当 $m_b=9$ 时,编码‘异或’

图6 $m_b=9$ 时编码异或树结构图

树的结构图。

由上述可知,对于码长为 2 304 的 LDPC 码,理论上采用这种编码异或树的设计可以使前向置换操作的延时减少 84.38%,但是面积却增加了 5 倍左右。不过对于高速应用场合,这点面积的代价是可以接受的。

计算 $p_2=T^{-1}f_6$ 与计算 $f_3=T^{-1}f_1$ 采用同样的方法。

已有的 LDPC 码硬件编码器^[4-6]是通过校验矩阵得到生成矩阵后直接做矩阵乘法进行编码的,由稀疏的校验矩阵得到的是密集的生成矩阵,这种实现方法没有充分利用校验矩阵的稀疏性,所以它的计算复杂度和码长成 2 次方关系。而本文的实现方法是:从校验矩阵编码就使用 Richadson 和 Urbanke 的快速编码算法,充分利用了校验矩阵的稀疏性,编码复杂度与码长成线性关系;此外,还设计了专用的编码桶形移位器和编码‘异或’树,使得编码器可以应用在高速应用场合;同时采用并行加流水的结构,提高了编码器的吞吐量。

本设计用 Verilog 实现了码长为 $n=2\ 304$,码率为 $r=0.5$ 的 LDPC 码硬件编码器,通过 VCS 仿真得到了编码

后的码字,然后使用 matlab 计算 $HC^T=0$ 成立,从而验证了功能的正确性。用 Synopsys 公司的 Design Compiler 工具,使用 TSMC 0.18 μm 库单元中的 slow 元件并且包括了线上延时后得到了速度和面积指标,具体数据如表 1 所示。

参考文献

[1] Air interface for fixed broadband wireless access systems. IEEE802.16e-2005.

[2] MACKEY D J C. Good error-correcting codes based on very sparse matrices. IEEE Trans Inform Theory, 1999, (45)399-431.

[3] RICHARDSOM T,URBANKE R. Efficient encoding of low-density parity-check codes. IEEE Trans Inform Theory, 2001,47(2):638-656.

[4] HOWLAND C, BLANKSBY A. A 220mW 1Gb/s 1024-bit rate-1/2 low density parity check code decoder In Proc. IEEE Custom Integrated Circ Conf.2001:293-296.

[5] MEHTA G, LEE H. An FPGA implementation of the graph encoderdecoder for regular ldpc codes. In CRL Technical Report.Communications Research Laboratory. University of Pittsburgh, 2002.

[6] ZHANG T,PARHI K.VLSI implementation-oriented (3,k)-regular low-density parity-check codes. In Proc. IEEE Workshop on Signal Processing Systems (SiPS): Design and Implementation, 2001:25-36.

(收稿日期:2006-07-21)

表 1 基于 IEEE802.16e 标准的 LDPC 码编码器的速度和面积指标

平台	速度(MHz)	面积(gate)
码长 $n=2304$, 码率 $r=0.5$ 的采用两级流水的 LDPC 硬件编码器	385	372 121

(上接第 118 页)

表 1 演示速度调整

调整项目	演示速度(帧/秒)	速度提升比(%)
12MHz 晶振	1	
24MHz 晶振	1.2	20
演示界面软件代码优化	3	150
ARM 运行时打开 Cache	8	167
调整 DMA 中 FIFO 大小	15	88
调整 DMA 搬运结构及方式	28	86
硬件上进一步改进 FIFO 大小	30(预计)	7
ARM 不从 Flash 取指改从 SRAM 取指	30 以上(预计)	

是放在 Flash 中,如果改成从 SRAM 中取指,估计效果会更加理想。

从以上分析可见,ARM 在整个设计中所起的主要作

用是控制图像的输入输出,以及循环控制 DSP Core 的运行停止等状态;DSP Core 的主要作用是处理运算应用程序,计算小目标识别程序。这样既分工又合作,能够充分发挥 ARM 的控制功能以及 DSP Core 的数字运算处理功能。

与此同时,由于 ARM 在整个设计当中主要起到一些辅助的控制作用,ARM922T 的一些扩展 DSP 运算功能没有用到,如果综合考虑到成本和性价比等因素,可以考虑采用 ARM7 硬核、NIO S 或其他形式的软核替代。

参考文献

[1] FURBER S,田泽,于敦山.ARM SOC 体系结构.盛世敏,译.北京:北京航空航天大学出版社,2002.

[2] CSCHWIND M. FPGA prototyping of a RISC processor core for embedded applications. IEEE Transactions on Very Large Scale Integration(VLSI)Systems, 2001,9(2).

[3] Hardware Reference Manual Version 3.1. www.altera.com. 2002-11.

(收稿日期:2006-08-14)