

引用格式:林青,杜高明,戴吴骏,等. 一种高性能极化码 Fast-SSC 译码器设计[J]. 微电子学与计算机,2021,38(2):7-13. [LIN Q, DU G M, DAI W J, et al. Design of a high-performance fast-SSC decoder for polar codes[J]. Microelectronics & Computer, 2021,38(2):7-13.]

一种高性能极化码 Fast-SSC 译码器设计

林 青,杜高明,戴吴骏,王晓蕾

(合肥工业大学 微电子设计研究所 教育部 IC 网上合作研究中心,安徽 合肥 230601)

摘 要: 为了进一步降低无线通信系统中信道编码极化码 Fast-SSC 译码器的译码延迟和硬件资源消耗,本文提出了两种对 Fast-SSC 译码算法的优化方法,基于 FPGA 硬件实现了码长 $N=1\ 024$ 优化的 Fast-SSC 译码器. 基于 Fast-SSC 译码树上的四种特殊结点 Rate0、Rate1、Rep 和 SPC,提出分别以 Rate0 结点和 Rep 结点为基础进行组合型特殊结点的拓展,使译码树上的特殊结点类型由原来的四种增加至十种以上,进一步加快了搜索译码树的速度;提出存储器资源复用的方法,相同译码树深度的结点使用相同的存储空间. 综合结果表明最大工作频率为 169 MHz,译码延迟为 $1.71\ \mu\text{s}$,数据吞吐率为 599 Mbps,存储器消耗为 14.8 Kbit,有效提升了 Fast-SSC 译码器的整体性能.

关键词: 极化码;Fast-SSC;低延迟;高性能;特殊结点

中图分类号: TN47

文献标识码: A

文章编号: 1000—7180(2021)02—0007—07

Design of a high-performance fast-SSC decoder for polar codes

LIN Qing, DU Gao-ming, DAI Wu-jun, WANG Xiao-lei

(Hefei University of Technology Institute of VLSI Design, IC Design Web-cooperation
Research Center of MOE, Hefei 230601, Anhui, China)

Abstract: In order to further reduce the decoding delay and hardware resource consumption of the Fast-SSC decoder of channel-coded polarized code in wireless communication system, this paper proposes two optimization methods for the Fast-SSC decoding algorithms, and realizes the fast-SSC decoder optimized with code length $N=1\ 024$ based on FPGA hardware. Based on the four special nodes Rate0, Rate1, Rep, and SPC in the fast-SSC decoding tree, the expansion of combined special nodes based on the Rate0 node and the Rep node respectively is proposed, which makes the special nodes in the decoder tree increase from the original four to more than ten, further speeding up the speed of searching decoder tree. A method of multiplexing memory resources is proposed in which nodes with the same decoding tree depth use the same storage space. The comprehensive results show that the maximum operating frequency is 169 MHz, the decoding delay is $1.71\ \mu\text{s}$, the data throughput is 599 Mbps, and the memory consumes 14.8 Kbit, which effectively improves the overall performance of the Fast-SSC decoder.

Key words: polar codes; Fast-SSC; low latency; special nodes

1 引言

目前,极化码作为新一代的信道编码技术已经成功应用于 5G 通信系统的 eMBB 应用场景,而且存在巨大的潜力适用于未来无线通信系统的各种应用场景^[1].当码长趋近无穷大时,极化码以数学方法被严格证明可以达到香农极限^[2].在极化码构造方法确定和保证译码性能不变的前提下,如何降低译码器的译码延迟和硬件资源消耗一直以来被作为研究的重点^[3].

极化码串行抵消(Successive Cancellation, SC)译码算法可以获得非常出色的译码性能,但是其串行译码的特点造成的高延迟与错误比特的传递问题却与无线通信系统中高速信息传输标准相悖.因此,文献[4]提出的简化 SC(Simplify SC, SSC)译码算法以 SC 译码算法为基础定义了两种特殊节点 Rate0 节点和 Rate1 节点,初步对 SC 译码树进行了简化.文献[5]基于 SSC 译码算法的基础上提出了 Fast-SSC 译码算法,另定义了两种基本特殊节点 Rep(Repetition)节点和 SPC(Single-Parity-Check)节点,如此译码树上的特殊节点类型增加至四种.随着码长的不断增大,Fast-SSC 译码树上的特殊节点的类型及数量都会显著增加,从而加快了搜索译码树的速度.为了增加译码树上特殊节点的类型,文献[6]提出基于四种节点的组合型特殊节点的可能性,文献[7]提出通过固定比特与信息比特的交换来设置特殊节点的方法,增加了组合型特殊节点数量.

但是,通过研究发现文献[6]和文献[7]中提出的组合型特殊节点类型依然具有可以继续扩展潜力.比如,组合型特殊节点 RepSPC 为右边的孩子结点时,左边的孩子结点为 Rate0 结点或 Rep 结点时,此时也可以再次拓展组合型的特殊结点 Rate0RepSPC 或 Rate0RepSPC.码长 N 越大,组合型特殊节点的类型及数量也就越多,从而进一步简化 Fast-SSC 译码树.另外,在硬件实现特殊结点译码器时,存在部分扩展后的特殊结点可以复用相同的译码器,较大程度降低了硬件资源的消耗.对于 Fast-SSC 译码器硬件架构的设计,通过分析文献[8]提出的 Tree 型译码结构和 Line 型译码结构,Line 型译码结构基于搜索译码树的译码器设计更加简单,而且可以大量减少计算内核 PE 的数量.另外,为了减少存储器资源的消耗,本文提出基于 Line 型译码结构提出相同译码树深度的结

点复用相同的存储空间的设计方法.

基于以上提出的改进方法,从降低译码延迟和硬件资源消耗的角度出发,本文提出一种高性能的 Fast-SSC 译码器架构,基于 FPGA 平台设计实现,并将实验结果与理论分析结果进行相互的印证.

2 Fast-SSC 算法原理

2.1 译码树搜索规则

给定极化码(N, K),其中 N 为码长, K 为信息的数量,则可构建最大深度为 $\log_2 N$ 的满二叉树,每层结点的子码长度为 $N_v = 2^{dv}$,其中 dv 为结点在译码树上的深度. SC 译码算法需要遍历整颗译码树,信道输入的初始对数似然值序列 α 从根结点开始向下传递,直到根结点获得两个孩子结点返回的估计比特序列才结束一次译码.如图 1 所示,一个极化码(8, 4)的译码器满二叉树.图中的白色叶子结点表示冻结比特的判决结点,黑色叶子结点表示信息比特的判决结点.

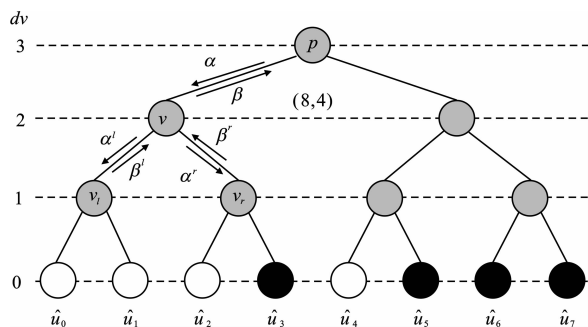


图1 码长 $N=8$ 的 SC 译码树

对于译码树中的任意结点 v ,首先当前搜索结点 v 从其父节点接收维度为 N_v 的 LLR 向量 $\alpha = \{\alpha_0, \alpha_1, \dots, \alpha_{N_v-1}\}$,计算完成后输出维度为 $N_v/2$ 的 LLR 向量 $\alpha' = \{\alpha'_0, \alpha'_1, \dots, \alpha'_{N_v/2-1}\}$ 至其左边孩子结点 v_l ,向量 α' 的计算公式如下

$$\alpha'[i] = \text{sign}(\alpha[i]) \text{sign}(\alpha[i + \frac{N_v}{2}]) \min(|\alpha[i]|, |\alpha[i + \frac{N_v}{2}]|) \quad (1)$$

当左边孩子结点 v_l 译码完成后,结点 v 接收到其左边孩子结点 v_l 返回的估计比特构成的维度为 $N_v/2$ 向量 $\beta = \{\beta_0, \beta_1, \dots, \beta_{N_v/2-1}\}$,结点 v 紧接着传递维度为 $N_v/2$ 的 LLR 向量 $\alpha'' = \{\alpha''_0, \alpha''_1, \dots, \alpha''_{N_v/2-1}\}$ 至其右边孩子结点 v_r ,向量 α'' 的计算公式如下

$$\alpha''[i] = \alpha[i + \frac{N_v}{2}] + (1 - 2\beta[i])\alpha[i] \quad (2)$$

当右边孩子结点译码完成后,结点 v 接收到其右边孩子结点 v_r 返回的估计比特构成的维度为 $N_v/2$ 向量 $\beta' = \{\beta'_0, \beta'_1, \dots, \beta'_{N_v/2-1}\}$ 时,此时结点 v 则可以计算返回的估计比特构成的维度为 N_v 向量 $\beta = \{\beta_0, \beta_1, \dots, \beta_{N_v-1}\}$, 向量 β 的计算公式如下

$$\beta[i] = \begin{cases} \beta[i] \oplus \beta'[i], & \text{if } i < \frac{N_v}{2} \\ \beta'[i - \frac{N_v}{2}], & \text{otherwise} \end{cases} \quad (3)$$

2.2 特殊结点族

Fast-SSC 译码算法包含的四类基本特殊结点 Rate0、Rate1、Rep 和 SPC. 特殊结点的码长和数量能够在很大程度上影响译码速度,并且每种特殊结点都有其独特的属性. 三种码长 $N=16$ 的译码树如图2所示,其中图(a)是 SC 译码算法的译码树,图(b)是 SSC 译码算法的译码树,图(c)是 Fast-SSC 译码算法的译码树. 通过对比可知,SSC 译码树上有 Rate0 结点和 Rate1 结点,初步对 SC 译码树进行剪枝, Fast-SSC 译码树上则多了 Rep 结点和 SPC 结点,译码树则变得更加简单.

○ 信息位
● 冻结位

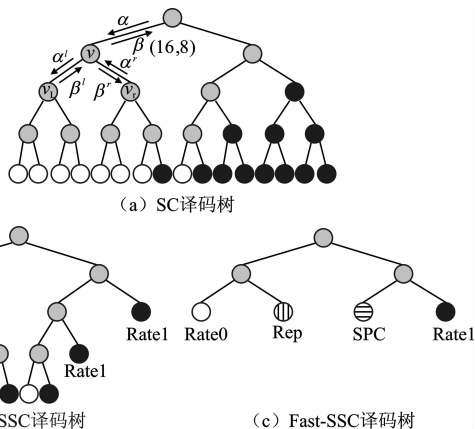


图2 译码树分类

Rate0 结点. 给定译码树上的结点 v , 若其作为根结点的子码树的叶子结点全部被选定为冻结比特的全零向量, 则该结点为 Rate0 结点;

Rate1 结点. 给定码长 N_v 的结点 v , 其所表示的子译码树的叶子结点都对应的是信息比特, 则称该结点为 Rate1 结点. 文献[4]中给出了 Rate1 结点的硬判决公式如下

$$\beta[i] = h(\alpha[i]) = \begin{cases} 0, & \text{if } \alpha[i] \geq 0 \\ 1, & \text{otherwise} \end{cases} \quad (4)$$

Rep 结点. Rep 结点码率为 $1/N_v$, 有且仅有一个信息比特位于最后一个叶子结点, 其余 $N_v - 1$ 个叶子结点为冻结比特, 一般取全零序列. Rep 结点接收

其父结点对数似然值序列 α , 返回估计比特序列 β . β 的硬判决公式如下

$$\beta[i] = \begin{cases} 0, & \sum_j \alpha[j] \geq 0 \\ 1, & \text{otherwise} \end{cases} \quad (5)$$

由上式可知, 估计比特序列 β 只有两种可能结果, 全0序列或全1序列. 仅有的一个信息比特的数值将会被重复 $N_v - 1$ 次, 这就是 Rep 结点之所以被称为重复码的原因.

SPC 结点. SPC 结点码率为 $(N_v - 1)/N_v$, 有且仅有一个冻结比特位于第一个叶子结点, 其余 $N_v - 1$ 个叶子结点为信息比特. 此时, 采用奇偶校验的方式来进行译码判决. 首先对接收的对数似然值序列进行硬判决, 得到初步的估计比特序列, 判决公式如下

$$h[i] = \begin{cases} 0, & \text{when } \alpha[i] \geq 0 \\ 1, & \text{otherwise} \end{cases} \quad (6)$$

提取上式的初步判决比特序列的奇偶校验位

$$parity = \bigoplus_{i=0}^{N_v-1} h[i] \quad (7)$$

然后提取输入对数似然比序列 α 中最低可靠度 LLR 的索引

$$j = \underset{i}{\operatorname{argmin}} |\alpha[i]| \quad (8)$$

最后对输入信息序列中最低可靠度的所判决结果进行奇偶校验, 若校验位与初步判决中的信息位不一致, 则翻转对应的信息位, 即可得到可靠性最高的估计比特序列

$$\beta[i] = \begin{cases} h[i] \oplus parity, & \text{when } i = j \\ h[i], & \text{otherwise} \end{cases} \quad (9)$$

3 特殊结点类型扩展

3.1 时序分析

前节中介绍了 Fast-SSC 译码算法包含的四类基本特殊结点 Rate0、Rate1、Rep 和 SPC. 特殊结点的码长和数量能够在很大程度上影响译码速度, 并且每种特殊结点都有其独特的属性. 遍历 SC 译码树的理论周期为 $2(N-1)$, 遍历特殊结点表示的子码树的理论周期为 $2(N_v-1)$, 其中 $N_v = 2^{d_v}$, 搜索 Fast-SSC 译码树的理论周期为

$$T = 2(N-1) - \sum_{d_v=1}^{\log N} (m_{\text{Rate0}} + m_{\text{Rate1}} + m_{\text{Rep}} + m_{\text{SPC}}) \cdot (2^{d_v+1} - 2) \quad (10)$$

式中, m_{Rate0} 、 m_{Rate1} 、 m_{Rep} 和 m_{SPC} 表示译码树上每层四种特殊结点数量, 以译码树上最大深度的特殊结点为准, 不能包含其孩子结点. 比如码长 N_v 的 SPC

结点,左边孩子结点为码长 $N_v/2$ 的 SPC 结点,右边孩子结点为码长 $N_v/2$ 的 Rate1 结点.

3.2 基于 Rate0 结点的简化译码

在译码过程中遇到 Rate0 结点是不需要译码的,而且在译码树的最大深度的 Rate0 结点会大概率出现在左边孩子结点. 一般在设计过程中会预先存储 Rate0 结点的硬判决的全零序列,当译码其右边的兄弟结点时它们的父结点直接读取全零序列从而计算所得右边孩子结点输入对数似然值序列. 给定合并特殊结点的码长 N_v ,合并的这类结点以增加 N_v-1 个全加器的代价可以进一步简化 Fast-SSC 译码树. 在保证低计算复杂度的情况下,基于 Rate0 结点简化译码的常见的几种结点可以归纳为表 1 中的几种类型.

表 1 基于 Rate0 结点的组合型特殊结点	
结点类型	描述
01	左边孩子为 Rate0 结点,右边孩子为 Rate1 结点.
001	左边孩子为 Rate0 结点,右边孩子为 01 结点.
0SPC	左边孩子为 Rate0 结点,右边孩子为 SPC 结点.
00SPC	左边孩子为 Rate0 结点,右边孩子为 0SPC 结点.
0Rep1	左边孩子为 Rate0 结点,右边孩子为 Rep1 结点.

假设所有的特殊结点都在一个时钟周期内完成译码,给定 001 结点的码长 N_v ,上表中 001 结点相比于 01 结点又多使用 $v-1$ 个加法器,而相比于未合并之前的译码多使用 $(3N_v-4)/2$ 个加法器. 同理,00SPC 结点相比于 0SPC 结点也是同样的. 基于 Rate0 结点的组合的灵活性较强,尤其在低码率条件下,Rate0 结点的数量会更多,组合也更具有多样性. 构造方法一定,考虑到码长和码率,特殊节点的分布和数量都会相应的变化,但是基于上述组合型特殊结点的规律具有一致性.

3.3 基于 Rep 结点的简化译码

Rep 结点的特点在增加有限处理单元的条件下也可以简化译码搜索过程. Rep 结点的估计比特序列只有两种结果,非 0 即 1,在穷举式搜索方法中适用性较强. 因此,当 Rep 结点是左边孩子结点时,且右边孩子结点同样为特殊结点,它们的父结点也属于组合型特殊结点. 特殊结点的合并原则是基于信道极化原理的,所以应该充分考虑合并的特殊结点的存在性. 另一方面,合并的特殊结点综合了两个孩子结点的计算复杂度,因此不同的组合方式也会造

成实际硬件实现中关键路径不同程度的变化,进而影响译码器的工作频率和数据吞吐率. 常用的基于 Rep 结点的组合型特殊结点可以归纳为表 2 中几种类型.

表 2 基于 Rep 结点的组合型特殊结点	
结点类型	描述
RepSPC	左边孩子为 Rep 结点,右边孩子为 SPC 结点.
Rep1	左边孩子为 Rep 结点,右边孩子为 Rate1 结点.
Rep0SPC	左边孩子为 Rep 结点,右边孩子为 0SPC 结点.
Rep01	左边孩子为 Rep 结点,右边孩子为 01 结点.
Rep001	左边孩子为 Rep 结点,右边孩子为 001 结点.
RepRep1	左边孩子为 Rep 结点,右边孩子为 Rep1 结点.
RepRepSPC	左边孩子为 Rep 结点,右边孩子为 RepSPC 结点.

4 硬件架构设计

4.1 整体硬件架构

Fast-SSC 译码器的硬件实现方案的整体硬件架构如图 3 所示. 主要包括 LLRs 计算模块、Beta 计算模块、存储模块、特殊结点译码器模块和控制器模块. 其中,LLRs 计算模块完成当前译码结点 LLRs 序列的迭代计算;Beta 计算模块完成返回值估计比特 β 的计算;存储器模块包括 Channel LLRsRAM、Internal LLRs RAM 和 Beta RAM;特殊结点译码器模块包括特殊结点类型拓展后的 Fast-SSC 译码树

上所有的特殊结点译码器,具体根据特殊结点属性进而调用对应的特殊结点译码器;控制器模块主要实现对译码树的有序搜索以及控制存储器模块的读写操作.

Fast-SSC 译码树的搜索过程可以大致分为三类运算操作,分别是普通结点的 LLR 迭代计算、特殊结点的译码计算和返回值估计比特 β 的计算. 为了方便译码器的设计,需要记录每个结点的属性,然后按照译码树上结点的搜索顺依次存储结点的属性信息. 结点类型拓展后的 Fast-SSC 译码树上可以大致分为两种类型结点:普通 LLRs 计算结点类和特殊结点类. 在本设计中令每个结点拥有各自的属性,包括结点的深度 dv 、左边孩子结点/右边孩子结点 i_s 和结点的类型. 在搜索译码树的过程中会首先读取每个结点的属性信息,然后根据结点的属性信息

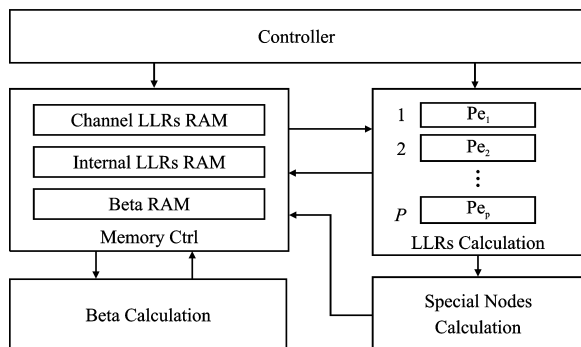


图 3 Fast-SSC 译码器的整体硬件架构

完成相应的结点操作。

4.2 LLRs 计算模块

对于标准 Fast-SSC 译码树,当搜索到码树不同的深度时所需要激活计算内核 PE 的数量是不同的,主要取决于当前译码结点的输入软信息向量 α 的维度。因此,综合考虑译码周期和 PE 资源的使用率,在译码树的不同深度复用相同数量的 PE 计算单元。若设置 PE 计算单元的总数量为 P ,令 $m = 2^{s-1}/P, m=1, 2, 2^2, 2^3, \dots$, 码树最大深度为 $\log_2 N$ 。以标准 SC 译码器为例,完成译码所需要的额外计算周期的总数量为

$$T_{\text{extra}} = 2 \sum_{\lambda=0}^{\log m} (2^\lambda - 1) = 2(2m - \log m - 2) \quad (11)$$

因此为了适当减少译码器的资源消耗,所需要的 PE 的数量满足条件 $2^{s-1} > P$ 。对于本设计码长 $N = 1024$ 的译码器,设置 $P = 32$ 能够较好平衡资源使用和译码延时。LLRs 计算模块如图 4 所示。

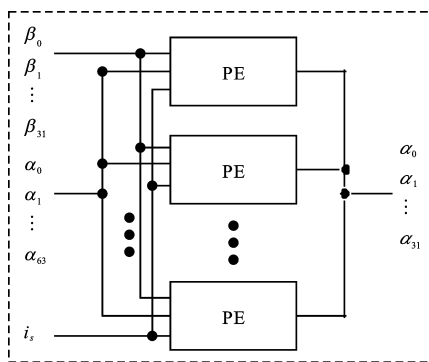


图 4 LLRs 计算模块

4.3 特殊结点译码器模块

本设计中, Fast-SSC 译码树上包含除了 Rate0、Rate1、Rep 和 SPC 四种基本结点,另外还包括组合的 6 中特殊结点 Rep1、RepSPC、01、001、0SPC、

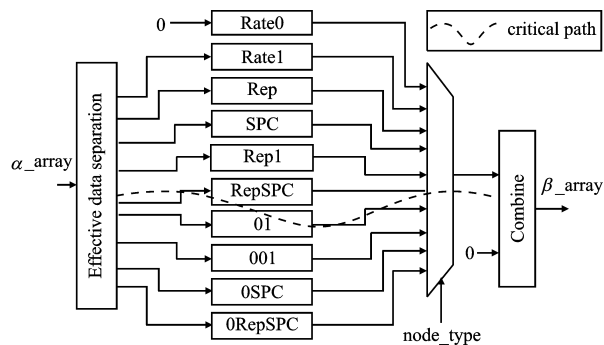


图 5 特殊结点译码器模块

0RepSPC。基于四种基本结点的特殊结点的拓展具有较强的灵活性,但是在设计硬件实现中必须考虑译码器的工作频率和硬件资源消耗,所以在本文的设计中增加 6 种以上所述拓展的特殊结点。

在图 5 中的特殊结点中, Rep1 结点是左边孩子结点 Rep 和右边孩子结点 Rate1 的组合型特殊结点。01 结点的左边孩子结点类型为 Rate0, 右边孩子结点类型为 Rate1。当 01 结点和 Rep1 结点的码长 N_v 相同时, 01 结点是 Rep1 结点的左边孩子结点 Rep 的判决比特序列为 0 时的特殊情况, 给定任意限制范围内的码长 N_v , 01 结点译码器和 Rep1 结点译码器可以使用同一个电路单元, 如图 6 所示。

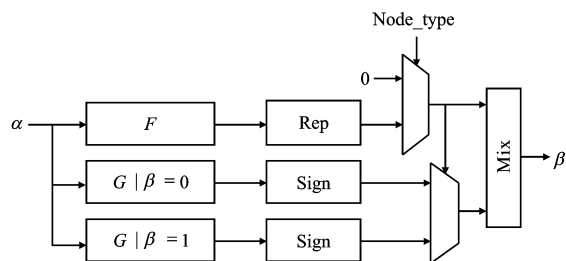


图 6 RepSPC 结点译码器和 0SPC 结点译码器结构图

同理, 给定任意限制范围内的码长 N_v , 001 结点和 0RepSPC 结点也可以使用相同的结点译码器, 如图 7 所示。

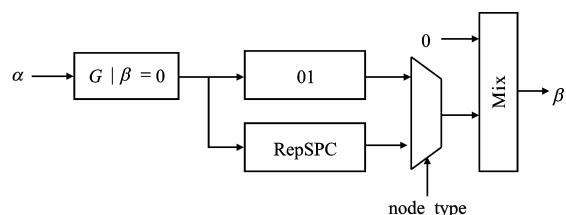


图 7 001 结点译码器和 0RepSPC 结点译码器结构图

4.4 存储器模块

存储器模块主要包括 Channel LLRs RAM、Internal LLRs RAM 和 Beta RAM, 其中 Channel

LLRs RAM 负责存储信道传输的 LLR 信息, Internal LLRs RAM 负责存储译码树除了根结点以外的所有结点之间传输的 LLR 信息, Beta RAM 负责存储判决估计比特 β . 存储器模块将根据控制器传递来的指令信息控制所有 RAM 的读写操作. 存储模块的设置对简化遍历译码树的过程十分重要, 在整个译码器的硬件实现中存储资源的消耗占比最大. 为了减少存储器资源的使用, 译码树深度相同的结点完全可以复用相同的存储空间. 因此, Channel LLRs 采用一片双口 RAM 存储; Internal LLRs 采用一片双口 RAM 存储; 估计比特值 β 则采用两片双口 RAM 存储, 分别存储左边孩子的返回值 β_l 和右边孩子的返回值 β_r . 存储器模块的结构图如图 8 所示.

4.5 Beta 计算模块

返回值估计比特 β 的计算规则如公式(3) 所示. 根据译码树的搜索规则, 每当得到左边孩子结点的估计比特 β_l 和右边孩子结点的估计比特 β_r , 即可计

算它们父结点的估计比特 β . 软信息 α 由译码树的根结点开始向下传递直到叶子结点, 计算得到叶子结点的硬判决信息 β . 然后硬判决信息 β 则是由叶子结点开始向上传递直到根结点, 即得到完整码长的硬判决估计比特序列 β . 左边孩子结点和右边孩子结点各采用一片双口 RAM 来存储. 在进行返回值估计比特 β 的计算时, 需要从两片双口 RAM 中相同的地址中同时分别读取数据.

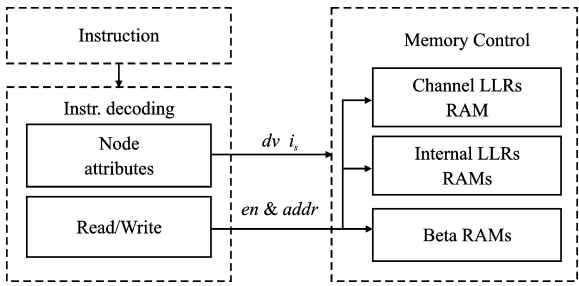


图 8 存储器模块结构图

表 3 优化的 Fast-SSC 译码器的 FPGA 综合结果对比

设计	[5]	[9]	[10]	[11]	本文
算法	Fast-SSC	Pipelined combinational SC	Fast-SSC	Fast-SSC-Flip	Fast-SSC
(N,K)	(1 024,512)	(1 024,512)	(1 024,512)	(1 024,512)	(1 024,512)
P	64	64	64	—	32
Tech. (nm)	40	40	40	40	28
LUTs	6 126	190 127	14 300	3 645	3 367
Registers	1 223	22 928	1216	—	1 902
RAM (bits)	23 592	13 312	18 350	17 244	14 848
f _{max} (MHz)	99. 76	—	89. 57	204. 67	169
Cycles(clks)	266	—	213	—	310
TP (Mbps)	384	1 240	430. 6	140. 6	599
Latency(us)	2. 67	—	2. 38	—	1. 71
AreaEfficiency (TP/KLUT)	62. 68	6. 52	30. 07	38. 57	177. 9

4.6 控制器模块

OC	读操作	LLRs计算	写操作	
	读操作	Beta计算	写操作	
RC	读操作	Beta计算	写操作	
SC	读操作	LLRs计算	特殊结点译码	写操作

图 9 三种运算操作的数据流向

控制器模块的主要功能是控制译码树的搜索流程. 遍历 Fast-SSC 译码树的过程中, 需要准确的识别每个结点的属性信息才可以完成相关的操作. 在本文

的译码器设计中, 将译码过程中的运算操作分为三种类型, 分别是普通结点的 LLR 迭代计算(OC)、特殊结点的译码计算(SC)和返回值估计比特 β 的计算(RC). 按照译码树的搜索规则可知, 以上三种运算类型的数据流可以表示为如图 9 所示, 其中 OC 和 RC 运算操作都在第一步读取 RAM 中的数据, 第二步计算模块完成相应的操作, 第三步将计算结果写入 RAM; SC 运算操作则需要如图 9 所示的四步完成操作, 其中读取的 LLR 数据经过 LLRs 计算模块处理完成后, 再将计算结果直接输入特殊结点译码器模块. 在搜索译码树的过程中, 如果计算的结

点码长 $N_v > 64$, 则读操作需要连续的多个时钟周期才能读取完成所有数据, 否则读操作在一个时钟周期内完成。

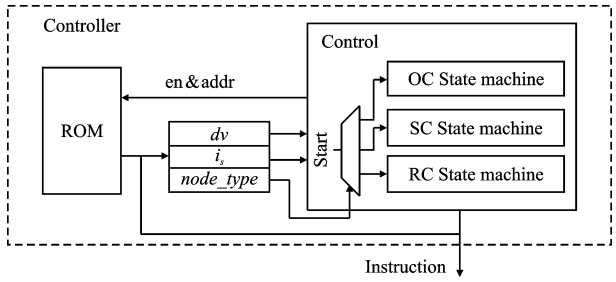


图 10 控制器结构图

为了便于控制译码流程, 在本设计中使译码树上的每个结点拥有各自的属性, 包括结点的深度 dv 、左边孩子结点/右边孩子结点 i_i 和结点的类型 $node_type$ 。因此, 需要将整颗译码树上的所有结点的属性信息按照译码树的搜索顺序预先写入存储器中。只有完成当前译码结点的运算操作之后, 再读取下一个结点的属性信息, 依次类推, 直到最后一个结点的运算操作完成。存储器模块的读操作和写操作的也由控制器模块的指令进行控制。如图 10 所示, 三种运算操作类型的读操作和写操作分别采用状态机设计。在每一个时钟周期内控制器都会更新控制指令, 从而使各个模块之间的数据通路能够完全按照译码树的遍历规则进行数据的有效传输。

5 实验结果与分析

表 3 展示了针对极化码(1 024, 512)的 Fast-SSC 译码器的基于 Xilinx Artix-7 平台综合结果。当基本计算内核 PE 的数量设置为 $P = 32$, Fast-SSC 译码器需要 310 个时钟周期, 最大工作频率为 169 MHz, 译码延迟仅为 $1.71 \mu s$, 相比于文献[4]和文献[10]提升了约 30% 左右。资源使用率指标 TP/KLUT 为 177.9, 相比与文献[5][9][10][11]分别提升了 183.8%、2629%、491.6% 和 361.2%。 , 由此可见本文所提出的优化设计方案大量节省了硬件资源的消耗。译码器的数据吞吐率为 599 Mbps, 相比与文献[5][10][11]分别提升了 56%、39.1% 和 326%。

通过以上对本文所提出的 Fast-SSC 译码器的实现结果的分析与对比可知, 相比于其他的设计方案, 虽然并行度 P 缩减了一倍, 但是译码延迟、数据吞吐率和资源消耗都提升显著, 因此提升了 Fast-

SSC 译码器的整体性能, 从而也验证了本文优化的 Fast-SSC 译码器的设计方案具有实际应用的潜力。

6 结束语

本文首先提出设置组合型特殊结点对 Fast-SSC 译码树剪枝的方法, 使得译码树上的特殊结点类型增加至十种以上。然后提出部分组合型特殊结点可以复用同一个的结点译码器方法, 减少了硬件资源的消耗。最后为了减少存储器资源的使用, 提出相同码树深度的结点复用相同存储空间的方法。因此, 本文设计了优化的 Fast-SSC 译码器, 并基于 FPGA 平台实现, 与目前的研究工作相比, 译码延迟优化约 30%, 数据吞吐率至少优化 39.1%, 硬件资源使用率 TP/KLUT 至少优化 183.8%。

参考文献:

[1] EGILMEZ K Z B, XIANG L P, MAUNDER R G, et al. The development, operation and performance of the 5G polar codes[J]. IEEE Communications Surveys & Tutorials, 2020, 22(1): 96-122. DOI: 10.1109/COMST.2019.2960746.

[2] ARIKAN E. Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels[J]. IEEE Transactions on Information Theory, 2009, 55(7): 3051-3073. DOI: 10.1109/TIT.2009.2021379.

[3] ZHANG C, PARHI K K. Latency analysis and architecture design of simplified SC polar decoders[J]. IEEE Transactions on Circuits & Systems II Express-Briefs, 2014, 61(2): 115-119. DOI: 10.1109/TCSIL.2013.2291065.

[4] ALAMDAR-YAZDI A, KSCHISCHANG F R. A simplified successive-cancellation decoder for polar codes[J]. IEEE Communications Letters, 2011, 15(12): 1378-1380. DOI: 10.1109/LCOMM.2011.101811.111480.

[5] SARKIS G, GIARD P, VARDY A, et al. Fast polar decoders: algorithm and implementation[J]. IEEE Journal on Selected Areas in Communications, 2014, 32(5): 946-957. DOI: 10.1109/JSAC.2014.140514.

[6] GIARD P, SARKIS G, THIBEAULT C, et al. A 638 Mbps low-complexity rate 1/2 polar decoder on FPGAs[C]//Proceedings of 2015 IEEE Workshop on Signal Processing Systems (SiPS). Hangzhou: IEEE, 2015: 1-6. DOI: 10.1109/SiPS.2015.7345007.

能在 20 min 之内,且 N、E、U 三方向误差低于 2 cm,甚至为毫米级.动态模式下测站的定位误差在分米级,满足了实时 PPP 定位需求且性能较为稳定.

参考文献:

[1] ZHOU F. Theory and methodology of multi-GNSS undifferenced and uncombined precise point positioning [D]. Shanghai: East China Normal University, 2018.

[2] YIN Q Q, LOU Y D, YI W T. Comparison and analysis of IGS real-time products[J]. Journal of Geodesy and Geodynamics, 2012, 32(6): 123-128. DOI: 10.14075/j. jgg. 2012. 06. 019.

[3] SONG W W. Research on real-time clock offset determination and real-time precise point positioning[D]. Wuhan: Wuhan University, 2011.

[4] WANG L, LI Z S, GE M R, et al. Validation and assessment of multi-GNSS real-time precise point positioning in simulated kinematic mode using IGS real-time service[J]. Remote Sensing, 2018, 10(2): 337. DOI:10.3390/rs10020337.

[5] KAZMIERSKI K, SO SNICA K, HADAS T. Quality

assessment of multi-GNSS orbits and clocks for real-time precise point positioning [J]. GPS Solutions, 2018, 22(1): 11. DOI:10.1007/s10291-017-0678-6.

[6] STÜRZE A, MERVART L, WEBER G, et al. The new version 2.12 of BKG Ntrip Client (BNC)[C]//EGU General Assembly Conference Abstracts, EGU, 2016.

[7] ZHANG L P, DANG Y M, GU S Z, et al. BDS satellite real-time orbit determination and SSR correction generation[J]. Bulletin of Surveying and Mapping, 2017(12): 1-5. DOI: 10.13474/j.cnki.11-2246.2017.0368.

[8] ZHANGX H, LI X X, GUO F. Satellite clock estimation at 1 Hz for realtime kinematic PPP applications [J]. GPS Solutions, 2011, 15(4): 315-324. DOI: 10.1007/s10291-010-0191-7.

作者简介:

刘亚男, (1995-), 硕士研究生. 研究方向为卫星导航算法. E-mail: liuya@ime.ac.cn

何文涛男, (1982-), 博士, 研究员. 研究方向为卫星导航接收机信号处理算法, 超大规模集成电路设计.

(上接第 13 页)

[7] GIARD P, BALATSOUKAS-STIMMING A, SARKIS G, et al. Fast Low-complexity decoders for low-rate polar codes [J]. Journal of Signal Processing Systems, 2018, 90(5): 675-685. DOI:10.1007/s11265-016-1173-y.

[8] LEROUX C, TAL I, VARDY A, et al. Hardware architectures for successive cancellation decoding of polar codes [C]//Proceedings of 2011IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Prague: IEEE, 2011: 1665-1668. DOI:10.1109/ICASSP.2011.5946819.

[9] DIZDAR O, ARIKAN E. A high-throughput energy-efficient implementation of successive cancellation decoder for polar codes using combinational logic[J]. IEEE Transactions on Circuits and Systems I: Regular Papers, 2016, 63(3): 436-447. DOI: 10.1109/TCSI.2016.2525020.

[10] ERCAN F, CONDO C, GROSS W J. Reduced-memory high-throughput fast-SSC polar code decoder architecture [C]//Proceedings of 2017 IEEE International Workshop on Signal Processing Systems (SiPS). Lori-

ent: IEEE, 2017: 1-6. DOI: 10.1109/SiPS.2017.8110014.

[11] ZENG J, ZHOU Y C, LIN J, et al. Hardware implementation of improved fast-SSC-flip decoder for polar codes [C]//Proceedings of 2019 IEEE Computer Society Annual Symposium on VLSI (ISVLSI). Miami, FL, USA: IEEE, 2019. DOI: 10.1109/ISVLSI.2019.00109.

作者简介:

林青男, (1992), 硕士研究生. 研究方向为 5G 移动通信信道编码和集成电路设计.

杜高明(通讯作者)男, (1977), 博士, 副教授. 研究方向为多核体系结构、片上网络体系结构、多核/NoC 性能分析. E-mail: 12345@163.com

戴吴骏男, (1994), 硕士研究生, 研究方向为 5G 移动通信信道编码和集成电路设计.

王晓蕾女, (1978), 博士, 副教授. 研究方向为集成电路设计.