5G LDPC码译码器实现

胡东伟*

(中国电子科技集团公司第五十四研究所 石家庄 050080)

摘 要：该文介绍了5G标准中LDPC码的特点，比较分析了各种译码算法的性能，提出了译码器实现的总体架构：将译码器分为高速译码器和低信噪比译码器。高速译码器适用于码率高、吞吐率要求高的情形，为译码器的主体；低信噪比译码器主要针对低码率、低信噪比下的高性能译码，处理一些极限情形下的通信，对吞吐率要求不高。分别对高速译码器和低信噪比译码器进行了设计实践，给出了FPGA综合结果和吞吐率分析结果。

关键词：5G移动通信；低密度奇偶校验码；译码器；FPGA

中图分类号：TN915 **文献标识码**：A **文章编号**：1009-5896(2021)04-1112-08

DOI: 10.11999/JEIT200046

On the Implementation of 5G LDPC Decoder

HU Dongwei

(54*th Institute of CETC*, *Shijiazhuang* 050080, *China*)

Abstract: This paper focuses on the Low-Density-Parity-Check (LDPC) decoder for 5G New Radio (NR) specification. After introducing the characteristics of the LDPC code in 5G NR, the performance of different decoding algorithms are compared, and then the overall architecture of the decoder is proposed. In the proposed architecture, the decoder is divided into high-speed decoder and high-performance decoder. The high-speed decoder is intended for high-rate and high throughput decoding, while the high-performance decoder is used for low-rate decoding under low Signal-to-Noise-Ratio (SNR) scenarios, which is for communications under extremely bad situations, and does not need a high throughput. The design is implemented on Field Programmable Gate Array (FPGA) and the results are shown.

Key words: 5G Mobile Communications; LDPC; Decoder; Field Programmable Gate Array (FPGA)

1 引言

2017年12月，第5代移动通信新无线(5G New Radio, 5G NR)标准[1]的发表，标志着第5代移动通信(5G)的正式诞生。相比于第4代移动通信标准——长期演进(Long Term Evolution, LTE)而言，5G采纳了很多的新技术。调制和编码是无线通信的两大支柱性技术。相比于LTE，5G的编码技术亦有着很多的看点。

5G的编码采用LDPC码和Polar码[2]。LDPC码具有较长的编码块长度，主要用于数据的传输；Polar码具有较短的编码块长度，主要用于信令的传输。Polar码作为一种全新的编码方式，受到了较大的关注。然而，5G的LDPC码具有更大的灵活性——它的最短信息比特块长度只有20 bit，最长可达8448 bit。如果将5G LDPC码应用在其他场合，则它不但可用于长码块的数据传输，亦可用于短码块的信令传输。

其次，5G LDPC码最低码率可低至1/5。这是目前进入实际使用的、最低码率的编码，具有非常接近香农极限的性能。这在深空通信、卫星通信中，具有重要意义。在深空通信中，性能的提高意味着更远的传输距离，或无需昂贵的低温射频前端[3,4]。

再次，5G LDPC码具有速率自适应的特点[5]。这是LDPC编码史上一次巨大的进步。传统上，由于LDPC码为分组编码，被视为难以实现速率自适应[6]。这是LDPC码相对于卷积码、Turbo码的一大劣势。5G LDPC码以其独特的编码方式，实现了速率自适应。这给了该码使用上的极大便利性和灵活性。

由于5G NR标准颁布的时间还不长，直接针对5G NR标准的LDPC译码器研究尚不多。文献[7]主要在译码的调度算法上进行了改进；文献[8]是一个采用归一化最小和(Normalized Minimum Sum, NMS)算法和分层调度的译码器完整ASIC实现方案；文献[9]给出了块并行架构和行并行架构的实现

收稿日期：2020-01-13；改回日期：2020-07-11；网络出版：2020-07-24
*通信作者： 胡东伟 hudw1980@sina.com

细节。与这些文献相比，本文所提架构具有以下特点：第一，基于SoC架构，采用基于处理器的调度方案，可实现更灵活的调度；第二，支持多种译码算法，特别地，本文硬件实现了log-bp译码算法；第三，在移位器的实现方法上，本文别具一格。

本文首先介绍5G LDPC码的编码器结构，然后比较该码在各种译码算法下的性能，并进一步分析各种译码算法的校验子通道实现结构和复杂度。在此基础上，提出了5G LDPC码译码器的整体实现结构，并给出了FPGA实现结果。

2　5G LDPC码的结构

5G LDPC码在一个基本编码器基础上，通过速率自适应打孔得来。

2.1　基本编码器结构

5G的基本编码器结构分为两种，分别由两个基本图(Base Graph, BG)来定义。两种基本编码器都是准循环编码。准循换子码块大小Z_c如表1所示。表1中Z即为可选择的准循换子码块长度Z_c。在编码过程中，需要用到循环右移单位矩阵。n阶循环右移单位矩阵为将$Z_c \times Z_c$的单位矩阵的每一行，循环右移n位所得。如图1给出了$Z_c = 4$时，1阶和2阶的循环右移单位矩阵的取值。

当采用基本图1时，信息码块长度为$22Z_c$，编码字长度为$66Z_c$，编码码率为1/3；当采用基本图2时，信息码块长度为$10Z_c$，编码字长度为$50Z_c$，编码码率为1/5。Z_c为表1中，使得$22Z_c$或$10Z_c$大于待编码信息码块长度的最小值。当信息码块的长度不是刚好为$22Z_c$或$10Z_c$时，在信息码块后补填充位0，并在编码完成后去除。填充后的信息码块记为$\boldsymbol{c}$。

编码方法为通用的分组码编码方式。设校验矩阵为$\boldsymbol{H}$，则校验位w须满足$\boldsymbol{H} \times \left[\boldsymbol{c}^{\mathrm{T}} \quad \boldsymbol{w}^{\mathrm{T}}\right]^{\mathrm{T}} = 0$。

校验矩阵$\boldsymbol{H}$由基本图定义。基本图1定义了一个46×68的基本矩阵$\boldsymbol{H}_{\mathrm{BG1}}$，基本图2定义了一个$42 \times 52$的基本矩阵$\boldsymbol{H}_{\mathrm{BG2}}$。将基本矩阵$\boldsymbol{H}_{\mathrm{BG1}}$或$\boldsymbol{H}_{\mathrm{BG2}}$的每个元素$n$, $n \neq -1$，替换为$Z_c \times Z_c$的n阶循环右移单位矩阵，-1则替换为全零矩阵，即得到校验矩阵$\boldsymbol{H}$。$\boldsymbol{H}_{\mathrm{BG1}}$和$\boldsymbol{H}_{\mathrm{BG2}}$的具体取值请参阅5G NR标准[1]。根据表1中不同的集指数，$\boldsymbol{H}_{\mathrm{BG1}}$和$\boldsymbol{H}_{\mathrm{BG2}}$有不同的取值。图2给出了集指数为1时$\boldsymbol{H}_{\mathrm{BG1}}$和$\boldsymbol{H}_{\mathrm{BG2}}$的图形表示。黑色处为全零元素。

编码完成后，得到的码字$\boldsymbol{d}^0 = \left[\boldsymbol{c}^{\mathrm{T}} \quad \boldsymbol{w}^{\mathrm{T}}\right]^{\mathrm{T}}$，长度分别为$68Z_c$(基本图1)或$52Z_c$(基本图2)。最后发送时，去掉$\boldsymbol{d}^0$的前$2Z_c$个元素，即$\boldsymbol{d} = \boldsymbol{d}^0(2Z_c + 1 : \mathrm{end})$，至此完成1/3或1/5码率的编码。

2.2　速率自适应打孔

5G的自适应打孔，可根据物理层帧结构可承载的比特数目，自适应截取编码后码字的发送比特数目，并可根据混合差错重传(HARQ)要求，截取编码后码字的不同段。记基本编码器完成后的码字长度为N，下层帧结构可承载的比特数目为N_E，版本号为$\mathrm{rv_{id}} \in \{0, 1, 2, 3\}$。将编码后码字放入一个长度为$N_c$的循环缓冲区内。如图3所示。$N_c$的长度由上层信令指定，$N_c \leq N$。当$N_c < N$时，只截取

表 1　准循换子码块长度取值

集指数 (i_{LS})	子码块大小集合 (Z)
0	{2, 4, 8, 16, 32, 64, 128, 256}
1	{3, 6, 12, 24, 48, 96, 192, 384}
2	{5, 10, 20, 40, 80, 160, 320}
3	{7, 14, 28, 56, 112, 224}
4	{9, 18, 36, 72, 144, 288}
5	{11, 22, 44, 88, 176, 352}
6	{13, 26, 52, 104, 208}
7	{15, 30, 60, 120, 240}

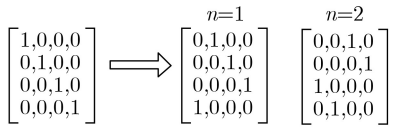
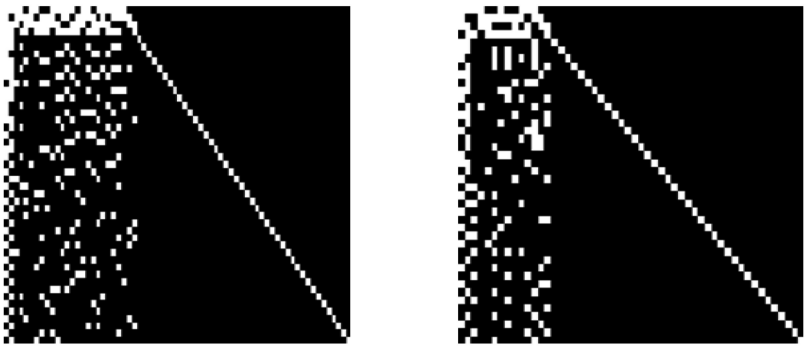


图 1 n阶循环右移单位矩阵示意图



(a) 基本图1　　(b) 基本图2

图 2 基本图1和基本图2的图形显示

编码块的前N_c个比特放入循环缓冲区。最后速率自适应打孔的输出，就是从该循环缓冲区内，从第N_s比特开始，取出N_E个比特。N_s由版本号$\mathrm{rv_{id}}$指定。N_c和N_s具体取值请参阅5G NR标准[1]。由这种速率自适应打孔方式可见，我们可实现任意速率的最终编码。这给我们的应用提供了极大的灵活性。

3 5G LDPC码的译码算法和性能分析

基本译码方式为，首先对打孔后的接收信号进行解打孔，然后利用标准的LDPC译码方式进行译码。表2的伪代码给出了译码过程[10–13]。其中y_i为接收数据，$L(q_i)$为变量节点对数似然比，$L(r_{ji})$为校验节点j向变量节点i传递的外信息，σ^2为信道噪声方差。

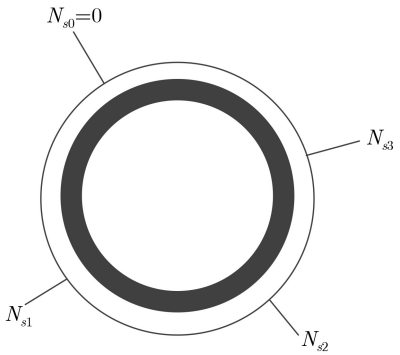


图 3 速率自适应打孔示意图

表 2 译码算法

初始化$\forall i,j$，$L^0(q_i)=2y_i/\sigma^2$，$L^0(r_{ji})=0$；
For 迭代次数l
For 每一个校验节点j
idx= 与校验节点j相连的所有变量节点坐标；
$L_{\mathrm{in}}=L^{l-1}(q_{\mathrm{idx}})-L^{l-1}(r_{j,\mathrm{idx}})$；//去掉自身产生的外信息
$L_{\mathrm{out}}=A\lg\mathrm{orithm}(L_{\mathrm{in}})$；//变量节点对校验节点的更新
$L^l(q_{\mathrm{idx}})=L_{\mathrm{in}}+L_{\mathrm{out}}$；//变量节点更新
$L^l(r_{j,\mathrm{idx}})=L_{\mathrm{out}}$；//外信息更新
End
$\forall i$, $d_i=L^l(q_i)>0$；
If $Hd==0$
Break；
end
end

上述计算过程的核心在于变量节点对校验节点的更新算法$A\lg\mathrm{orithm}(\bullet)$。不同的更新算法有不同的性能[14]。表3给出了4种不同的更新方式。表中$V_j\backslash i$表示与校验节点j相连的所有的、除去变量节点i以外的所有变量节点。α和β为常数，$\mathrm{sgn}(\bullet)$为符号函数。$\tanh(x)=(\mathrm{e}^x-\mathrm{e}^{-x})/(\mathrm{e}^x+\mathrm{e}^{-x})$，$\tanh^{-1}(\bullet)$为$\tanh(\bullet)$的逆函数。

图4给出了5G LDPC码在两种基本图下，在这4种译码算法下的性能。由图4可见，在所有码率下，log-bp(BP)译码算法的性能非常好，且较MSA, Offset MSA(OMSA), Normalized MSA (NMSA)的性能要好得多。MSA算法的性能最差，较log-bp算法有接近2 dB的差别。OMSA和NMSA算法性能介于BP算法和MSA算法性能之间。低码率时，OMSA算法性能较NMSA好；高码率时，NMSA算法性能较OMSA好。1/5码率时，log-bp算法译码门限可以达到$E_{\mathrm{b}}/N_0\approx 0$ dB (SNR $\approx$ −4 dB)。低码率时，OMSA算法性能较log-bp算法要差0.5 dB左右。这在深空通信、卫星通信等极限通信条件下，是不可忽略的。

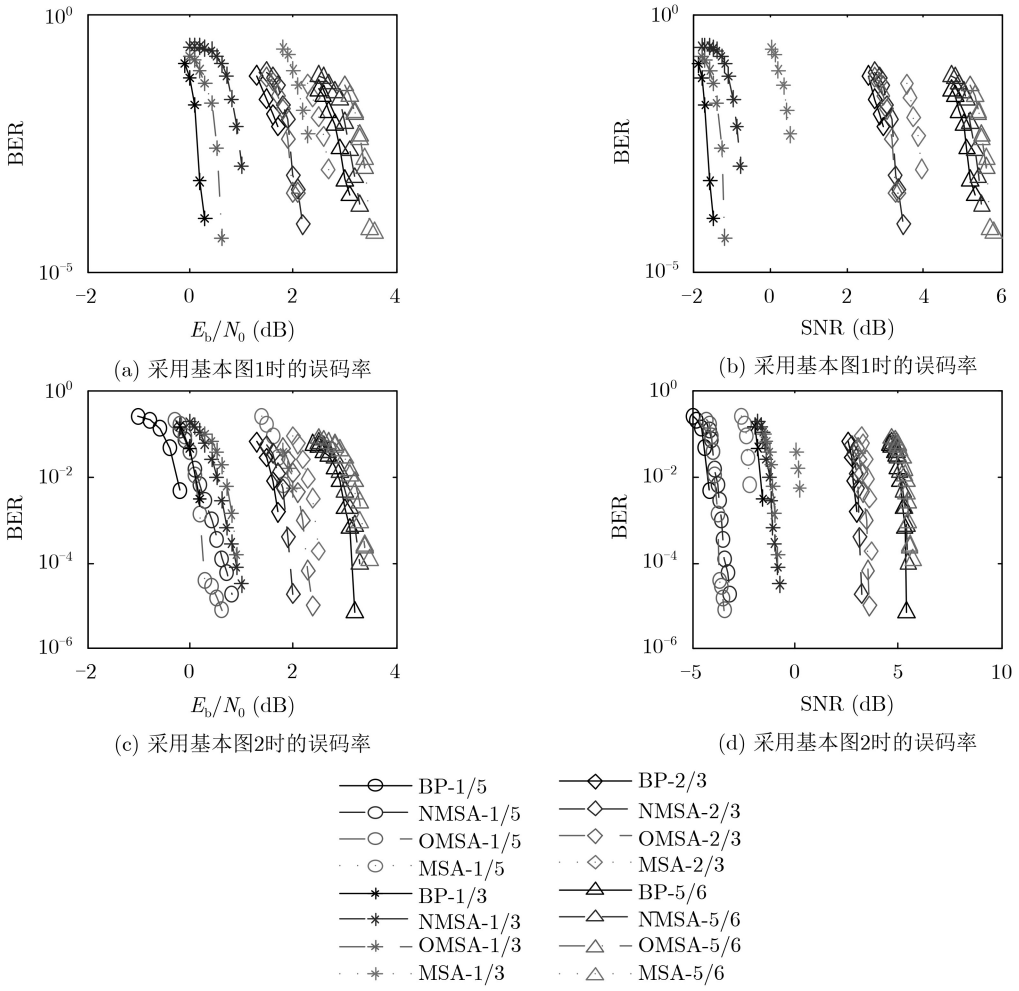
4 校验子通道结构和复杂度

典型的准循换LDPC码译码器为很多并行通道，每个通道执行一个校验子的计算，即表3中的计算。本节主要分析以上4种算法的通道结构和复杂度。

MSA, Offset MSA和Normalized MSA算法只涉及到取绝对值和大小比较操作，较简单。根据以上$\boldsymbol{H}_{\mathrm{BG1}}$和$\boldsymbol{H}_{\mathrm{BG2}}$的定义，5G LDPC码的每个校验子，最多与19个变量节点相连。因此，可设计这3种算法的校验子计算通道结构如图5所示。图5中，校验子通道连接在处理器的数据总线上。当计算某校验子时，处理器依次写入与该校验子相连的各个变量节点值。当写入第1个(地址为0)变量节点时，通道

表 3 变量节点对校验节点更新方式

序号	算法名称	更新方式
1	log-bp(BP)	$L_{\mathrm{out}}^{\mathrm{log-bp}}(r_{ji})=2\tanh^{-1}\left(\prod_{i'\in V_j\backslash i}\tanh\left(L_{\mathrm{in}}(q_{i'j})/2\right)\right)$
2	MSA	$L_{\mathrm{out}}^{\mathrm{MSA}}(r_{ji})=\prod_{i'\in V_j\backslash i}\mathrm{sgn}\left(L_{\mathrm{in}}(q_{i'j})\right)\bullet\min_{i'\in V_j\backslash i}\left(\left\lvert L_{\mathrm{in}}(q_{i'j})\right\rvert\right)$
3	Offset MSA(OMSA)	$L_{\mathrm{out}}^{\mathrm{Offset}}(r_{ji})=\prod_{i'\in V_j\backslash i}\mathrm{sgn}\left(L_{\mathrm{in}}(q_{i'j})\right)\bullet\max\left(\min_{i'\in V_j\backslash i}\left(\left\lvert L_{\mathrm{in}}(q_{i'j})\right\rvert\right)-\beta,0\right)$
4	Normalized MSA(NMSA)	$L_{\mathrm{out}}^{\mathrm{Norm}}(r_{ji})=\alpha\prod_{i'\in V_j\backslash i}\mathrm{sgn}\left(L_{\mathrm{in}}(q_{i'j})\right)\bullet\min_{i'\in V_j\backslash i}\left(\left\lvert L_{\mathrm{in}}(q_{i'j})\right\rvert\right)$



(a) 采用基本图1时的误码率　(b) 采用基本图1时的误码率

(c) 采用基本图2时的误码率　(d) 采用基本图2时的误码率

图 4 不同算法的性能

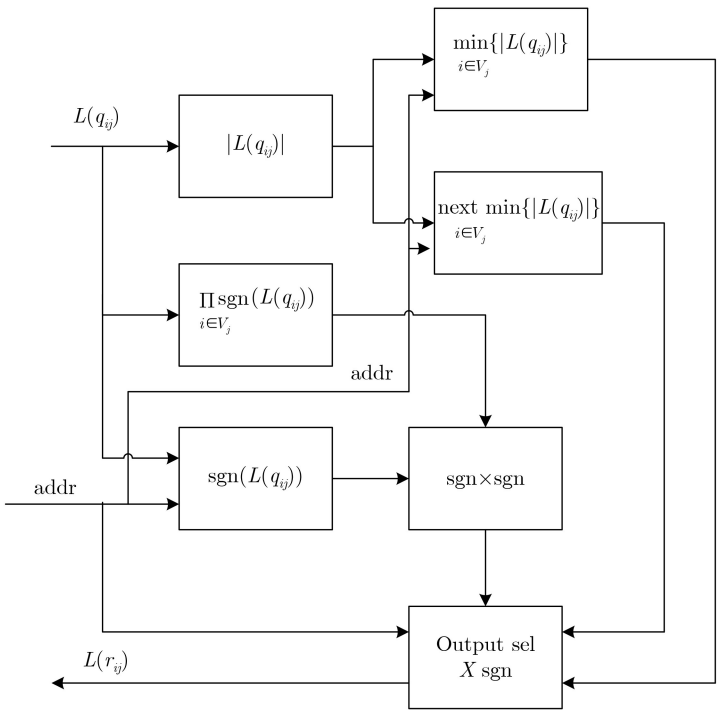


图 5 MSA, Offset MSA和Normalized MSA算法的校验子计算通道结构

复位。在写入的过程中，通道内部依次比较得出绝对值最小变量节点、绝对值次小变量节点以及他们的地址；并且，通道内计算所有节点的符号乘积，保存每个变量节点的符号。由于最多只有19个变量节点，符号位使用19个寄存器保存即可。当输入完毕后，处理器开始读出计算结果，即每个变量节点的外信息。读出时，根据地址选择符号位，选择绝对值最小或次小值，相乘合并为输出外信息值。与MSA算法相比，Offset MSA和Normalized MSA只是在输出时，做一点特别计算。通道结构可设计成可配置实现这3种算法，且α和β值可配置。

根据表3中的公式，log-bp算法计算较复杂。但是，仔细分析该计算公式可发现，log-bp算法的计算，具有迭代的性质。也就是，假如定义两个输入节点的log-bp计算为$\mathrm{BP}(l_1,l_2)$，即[15]

$$\mathrm{BP}(l_1,l_2)=2\tanh^{-1}(\tanh(l_1)\tanh(l_2)) \tag{1}$$

则有

$$\begin{aligned}\mathrm{BP}(l_1,l_2,l_3)&=2\tanh^{-1}(\tanh(l_1)\tanh(l_2)\tanh(l_3))\\&=\mathrm{BP}(\mathrm{BP}(l_1,l_2),l_3)\end{aligned} \tag{2}$$

于是，如果以$\mathrm{BP}(l_1,l_2)$为基本计算节点，则可构造如图6所示的计算过程。图中每个圆圈代表一个$\mathrm{BP}(l_1,l_2)$计算节点。计算分为前向迭代计算(记结果为t_i^1)和反向迭代计算(记结果为t_j^2)。最后校验节点对变量节点产生的外信息输出为前向、反向计算相遇处的计算结果，即

$$L^{\log-\mathrm{bp}}(r_i)=\mathrm{BP}\left(t_{i-1}^1,t_{i+1}^2\right) \tag{3}$$

根据$\tanh(\bullet)$函数的定义，式(1)中$\mathrm{BP}(l_1,l_2)$的计算可进一步简化为

$$\begin{aligned}\mathrm{BP}(l_1,l_2)&=\ln\left(\frac{1+\exp(l_1+l_2)}{\exp(l_1)+\exp(l_2)}\right)\\&=\mathrm{sgn}(l_1)\,\mathrm{sgn}(l_2)\ln\left(\frac{1+\exp(|l_1|+|l_2|)}{\exp(|l_1|)+\exp(|l_2|)}\right)\\&=\mathrm{sgn}(l_1)\,\mathrm{sgn}(l_2)\,\mathrm{Min}^*(l_1,l_2)\end{aligned} \tag{4}$$

其中，

$$\begin{aligned}\mathrm{Min}^*(l_1,l_2)=&\min(|l_1|,|l_2|)\\&+\ln(1+\exp(-(|l_1|+|l_2|)))\\&+\ln(1+\exp(-||l_1|-|l_2||))\end{aligned} \tag{5}$$

结合式(4)和式(5)，即得到基本计算节点$\mathrm{BP}(l_1,l_2)$的计算。式(5)中的后两项对数部分，可通过查表实现。由于函数$\ln(1+\exp(-x))$下降较快，仿真发现，这两个表格只需8个条目大小就足够。因此，这两个查找表可直接用组合逻辑实现。式(5)的实现复杂度不高。

然而，在式(3)的计算过程中，需要存储输入对数似然比值和前向迭代的计算结果。根据5G NR标准，每个校验子最多与19个变量节点相连，因此至少需要38个8位(假设每个变量位宽为8位)存储器来保存输入数据和前向计算结果。这造成了log-bp的实现复杂度较高。图7给出了log-bp算法的通道实现结构。图中Mem即为存储输入对数似然比和前向迭代计算结果的存储器。为方便处理器读取计算结果，输出采用一个存储器进行缓冲。表4给出了log-bp算法与MSA, Offset MSA和Normalized MSA算法的通道实现复杂度。表中的数值为在Altera Stratix IV EP4SGX290HF35C2 FPGA上的实现结果。

5 5G LDPC码译码器的整体架构设计和实现

5.1 架构设计的考量依据及设计方案

(1) 从图4的仿真性能来观察，在低信噪比时，MSA, Offset MSA和Normalized MSA算法性能较log-bp算法性能相差较大，因此log-bp算法的实现不可缺少，主要用于信噪比极低的极限情况下。

(2) Log-bp算法的前向、反向迭代过程计算较慢，因此，log-bp算法译码的吞吐率较低；相比较而言，采用MSA, Offset MSA和Normalized MSA算法的译码器，计算过程较快。所幸的是，从应用的角度来说，log-bp算法主要处理极限通信的情况，对吞吐率要求不高；而采用MSA, Offset MSA和Normalized MSA算法的译码器对吞吐率要求较高，但他们的通道结构简单，可采用更多的并行通道。

(3) 倘若一个变量节点被打孔，它的$L_{\mathrm{in}}(q_{i'j})$为零，根据表3中的计算公式，则该行校验子对所有变量节点产生的外信息均为零，因此该校验子的计算不用执行。由于5G LDPC可通过打孔实现任意速率的编码，因此译码时，需要有灵活地控制调度方案，适应不同码率下的译码。

根据以上设计考量依据，本文提出图8所示的译码器总体架构。图中虚线框内即为LDPC译码器，他们连接到一个处理器(DSP)的总线上。译码器内分为3大部分，log-bp译码加速器、高速译码加速器和数据存储器。其中log-bp译码加速器与其他两部分相互独立。Log-bp译码加速器为24个图7所示的并行通道。设置24个通道是因为，总线宽度为64位，每个通道输入输出数据位宽为8位。24个通道分为3组，每组8个通道。根据图7所示架构及延时，3组交错执行，刚好能保证数据不间断地写入，不间断地读出。

高速译码加速器可执行MSA, Offset MSA和

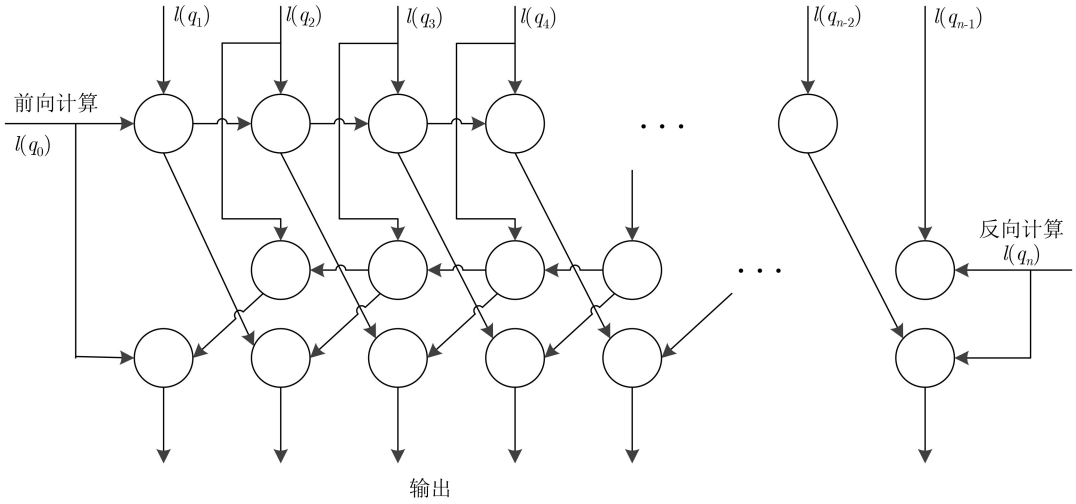


图 6 Log-bp算法的计算过程

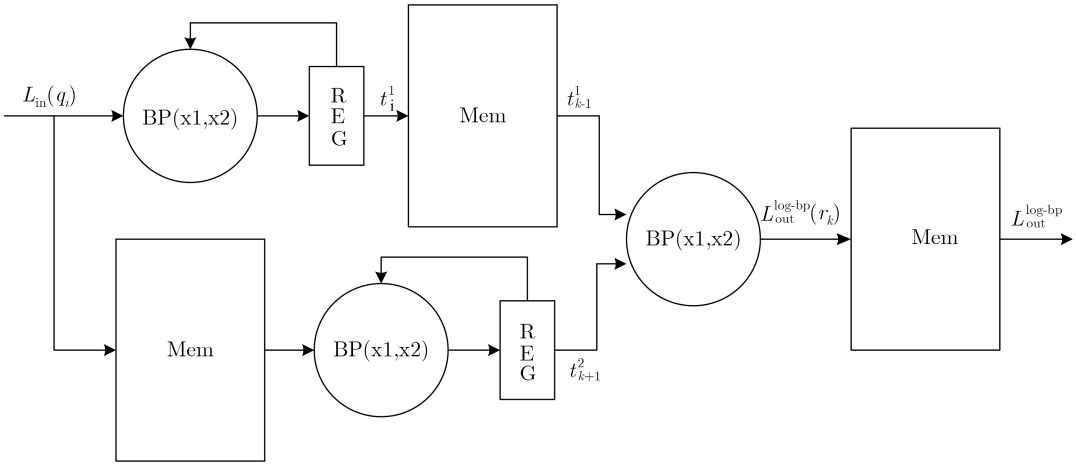


图 7 Log-bp算法通道实现结构

表 4　通道实现复杂度

序号	算法	组合逻辑	寄存器	存储器(bit)
1	log-bp	447	85	768
2	MSA, Offset MSA, Normalized MSA	222	58	0

Normalized MSA算法的译码，主要包括移位器和384个图5所示的并行通道[16,17]。图9给出了高速译码器的详细结构。在处理器调用高速译码之前，处理器首先(或通过外部DMA)将待译码数据放在数据存储器内，然后配置高速译码器循环块大小Z_c。Z_c的配置将使得高速译码器内只有Z_c个通道开启工作。然后，处理器向高速译码器写入一个控制命令，启动一个循环块的校验子计算。该控制命令包含的信息为数据存储器地址和移位器偏移量。高速译码器收到该命令后，按照控制命令的地址，一个时钟从数据存储器内读出Z_c个通道的输入数据对数似然比和外信息，进行循环移位、计算、反移位，然后将外信息结果写回该地址。整个过程耗费$12+2d$个时钟。d为校验子上连接的变量节点个数。写回完成后，处理器可控制进行下一块Z_c个校验子

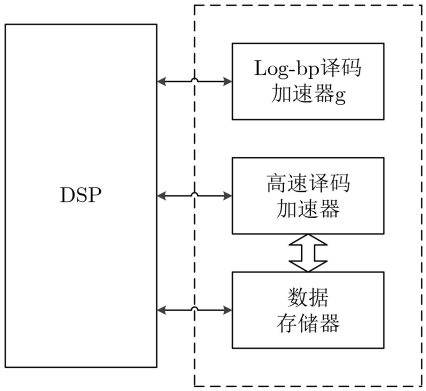


图 8 译码器总体架构

的译码。如此完成一次译码和多次译码迭代。在上述过程中，处理器很容易判断哪些数据已被打孔，从而直接忽略。数据存储器分为输入数据存储器和外信息存储器两组，每组的位宽均为3072 bit，可以保证一个时钟同时读取(或写入)384个通道的输入数据和(或)外信息。

图9中，384个通道的加速器只有Z_c个通道开启。移位器必须实现Z_c个通道数据的循环移位。根据表1，Z_c可以有多达51个取值，最大取值384。要实现51种情形下的任意循环移位，且循环移位范围可达到384，移位器的实现较复杂。本文推荐以下

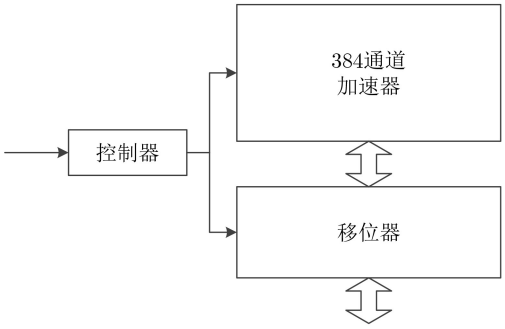
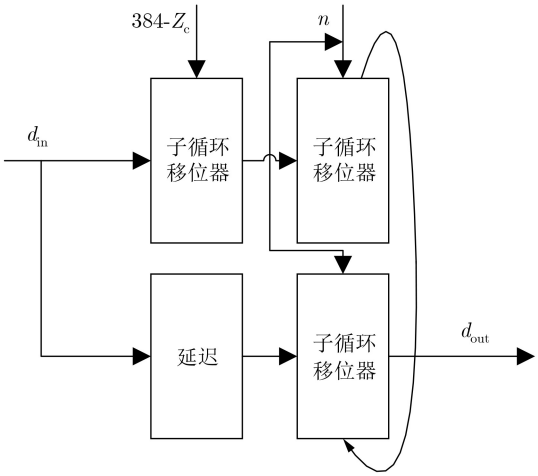
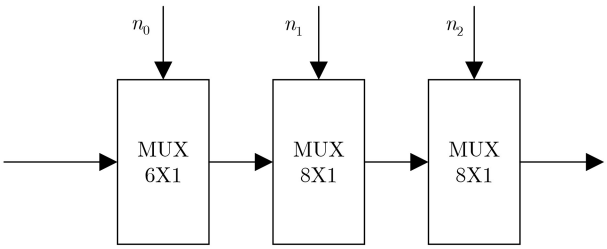


图 9 高速译码加速器架构



(a) 移位器总体架构



(b) 子循环移位器

图 10 移位器架构

图10(a)所示的移位器实现方案。图10(a)中，假设$Z_c < 384$，输入数据d_{in}的下部Z_c个数据有效，上部$384 - Z_c$个数据无效。输入数据首先同时输入子循环移位器和一个延迟器。子循环移位器向上移位$384 - Z_c$个数据，将d_{in}的有效数据抵到最顶端。然后，在第2级，移位后的d_{in}和延迟后的d_{in}联动，进行指定的n阶循环移位。这里的循环，是将图10中上部子循环移位器移出的数据，移入下部的子循环移位器的输入。最后，整个移位器的输出为下部子循环移位器的输出，其最低Z_c个数据有效。图10(b)给出了子循环移位器的实现。由于子循环移位器最大需要移位384位，这需要通过3级流水的多路器实现，每一级中间通过寄存器隔开(图中未示出寄存器)。3级流水多路器分别为6选1、8选1和8选1多路器。多路器的选择参数n_0, n_1和n_2通过n变换得来。$n = 64n_0 + 8n_1 + n_2$。图10所示的移位器，在控制器的控制下，既执行读出数据的正向移位，也执行写回数据的反向移位。

5.2 FPGA实现结果

表5给出了图8所示架构在FPGA上的实现结果，不包括DSP。从这些数据可以看到，高速译码

表 5 译码器的FPGA实现结果

序号	模块	组合逻辑	寄存器	存储器(bit)	最大时钟频率(MHz)	资源占用比例(%)
1	高速译码加速器(含存储器)	124457	50515	393216	112.74	74
2	log-bp译码加速器	10901	1937	18432	97.59	5

表 6 译码器吞吐率(Mbps)

序号	模块\码率	1/5	1/2	5/6	9/10
1	高速译码加速器(含存储器)	29.94	119.74	598.69	1077.60
2	log-bp译码加速器	2.84	11.37	56.85	102.34

加速器(含数据存储器)耗费的资源非常多，达到了EP4SGX290HF35C2 FPGA资源的74%。高速译码加速器耗费的资源是log-bp译码加速器的大约15倍。二者可达到的时钟速率均较高，可达到100 MHz左右。

5.3 吞吐率分析

表6给出了不同码率下，采用基本图2，译码器时钟为100 MHz，译码器平均迭代15次时的吞吐率。如果将译码器做成ASIC芯片，时钟速率可以成倍提高，相应地吞吐率也成倍提高。例如，当码率为9/10时，100 MHz时钟已经可以达到1 Gbps吞吐率。假若采用1 GHz时钟，则吞吐率将达到10 Gbps。这个吞吐率很高，甚至可以在光纤通信系统中使用[18]。

6　结束语

5G LDPC码具有低码率、高性能和速率自适应的特点，具有非常大的灵活性。除了用在地面移动通信系统中外，亦可以用在深空通信、卫星通信甚至光纤通信中。本文分析了5G LDPC码的构造和性能之后，提出其译码器的总体架构：将译码器分为高速译码器和低信噪比译码器，并进行了设计实践，给出了译码器的FPGA综合结果和吞吐率分析结果。本文的工作将对从事5G研发，或更广泛地，从事通信系统研发的人员，有一定的参考意义。

参 考 文 献

[1] 3GPP TS 38.212 v15.2.0(2018-06) Technical Specification Group Radio Access Network; NR; Multiplexing and Channel Coding (Release 15) [S]. 2017.

[2] BAE J H, ABOTABL A, LIN H P, *et al*. An overview of channel coding for 5G NR cellular communications[J]. *APSIPA Transactions on Signal and Information Processing*, 2019, 8: e17. doi: 10.1017/ATSIP.2019.10.

[3] 欧阳成. 深空通信中LDPC码编码方法研究[D]. [硕士论文], 西安电子科技大学, 2009.

[4] 王博. 低信噪比卫星通信中的编码与解调技术研究[D]. [硕士论文], 杭州电子科技大学, 2013.

[5] 朱文君. 实数域上的无速率码构造[D]. [硕士论文], 西安电子科技大学, 2015.

[6] HAMIDI-SEPEHR F, NIMBALKER A, and ERMOLAEV G. Analysis of 5G LDPC codes rate-matching design[C]. The IEEE 87th Vehicular Technology Conference, Porto, Portugal, 2018: 1-5. doi: 10.1109/VTCSpring.2018.8417496.

[7] 白薇. 5G通信系统中LDPC编译码器的设计与实现[D]. [硕士论文], 西安电子科技大学, 2018.

[8] 康丁文. 5G通信系统中高效LDPC译码技术研究[D]. [硕士论文], 西安电子科技大学, 2019.

[9] 黄福威. 5G-LDPC码编译码器设计与FPGA实现技术研究[D]. [硕士论文], 西安电子科技大学, 2019.

[10] 黄海艺. 低密度奇偶校验(LDPC)码改进译码算法研究[D]. [博士论文], 华南理工大学, 2013.

[11] 田宇. QC-LDPC码构造及其译码研究[D]. [硕士论文], 西安电子科技大学, 2015.

[12] 邓堤峡. 面向5G通信的LDPC码译码算法研究[D]. [硕士论文], 西安电子科技大学, 2018.

[13] ZHOU Yangcan, LIN Jun, and WANG Zhongfeng. Efficient approximate layered LDPC decoder[C]. 2017 IEEE International Symposium on Circuits and Systems (ISCAS), Baltimore, USA, 2017: 1–4. doi: 10.1109/ISCAS.2017.8050908.

[14] WU Xiaoning, JIANG Ming, ZHAO Chunming. Decoding optimization for 5G LDPC codes by machine learning[J]. *IEEE Access*, 2018, 6: 50179–50186. doi: 10.1109/ACCESS.2018.2869374.

[15] DIVSALAR D, DOLINAR S, JONES C R, *et al*. Capacity-approaching protograph codes[J]. *IEEE Journal on Selected Areas in Communications*, 2009, 27(6): 876–888. doi: 10.1109/JSAC.2009.090806.

[16] 安宁. 兼容DVB-S2X标准的全速率高速LDPC译码器设计与FPGA实现[D]. [硕士论文], 西安电子科技大学, 2016.

[17] 刘冬培. DVB-S2标准中LDPC码的编译码算法研究与实现[D]. [硕士论文], 国防科学技术大学研究生院, 2008.

[18] 袁建国. 高速超长距离光通信系统中超强FEC码型的研究[D]. [博士论文], 重庆大学, 2007.

胡东伟：男，1980年生，高级工程师，研究方向为无线通信理论和数字大规模集成电路设计.

责任编辑：余　蓉