(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 102118173 A
(43)申请公布日 2011.07.06

(21)申请号 201110029300.9

(22)申请日 2011.01.27

(71)申请人 牛毅
地址 100094 北京市海淀区邓庄南路 9 号

(72)发明人 牛毅 马忠松 傅得立

(74)专利代理机构 中国航天科技专利中心 11009
代理人 安丽

(51)Int.Cl.
H03M 13/11(2006.01)

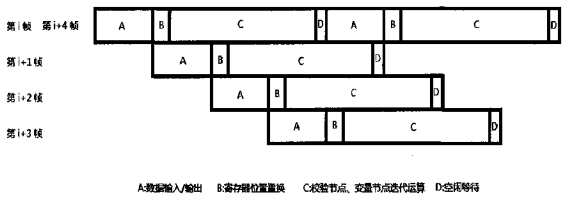
权利要求书 1 页 说明书 6 页 附图 2 页

(54)发明名称

一种 LDPC 及其缩短码的高速译码方法

(57)摘要

本发明公开了一种 LDPC 及其缩短码的高速译码方法，本发明以 CCSDS 为标准对 (8176,7154) 码型，在较低成本的硬件平台上实现了对高速数据流的译码，解决了长码长的低密度奇偶校验码在高码率下的译码问题。



CN 102118173 A

1. 一种 LDPC 及其缩短码的高速译码方法，其特征在于通过以下步骤进行实现：

(1) 对译码数据的第 i 个数据帧进行判断：

若输入为 LDPC 数据，则按所使用的输出数据寄存器的地址将 LDPC 数据中帧头数据的原码进行存储；并由输出数据寄存器输出前一运算周期中数据帧的帧头数据；

若输入为 LDPC 的缩短码数据，则按所使用的输出数据寄存器的地址将 LDPC 数据中帧头和填充数数据的原码进行存储；并由输出数据寄存器输出前一运算周期中数据帧的帧头和填充数数据；

(2) 按照所使用的中间数据寄存器和迭代结果数据寄存器的地址存储第 i 个数据帧中的有效数据；并由步骤 (1) 中所述的输出数据寄存器输出前一运算周期中的译码数据；

(3) 第 i 个数据帧存储完毕后，对第 i 个数据帧进行校验节点和变量节点的迭代运算操作；同时对输入的第 i+1 个数据帧从步骤 (1) 开始利用不同的输出数据寄存器、中间数据寄存器和迭代结果数据寄存器进行操作；

(4) 当对第 i 个数据帧的操作完成并输出后，重新采用对第 i 个数据帧进行操作的数据寄存器、中间数据寄存器和迭代结果数据寄存器对新到达的数据帧进行操作；

在上述步骤中，同时对输入的所有数据帧进行操作，对一个数据帧进行存储后操作所用时间等于对该单个数据帧之后输入的其余各数据帧进行存储所用时间的累加和。

2. 根据权利要求 1 所述的一种 LDPC 及其缩短码的高速译码方法，其特征在于：所述步骤 (3) 中的校验节点和变量节点的迭代运算操作分别由校验节点运算模块和变量节点模块实现，并可在连续到达的数据帧中进行复用。

3. 根据权利要求 1 所述的一种 LDPC 及其缩短码的高速译码方法，其特征在于：同时进行操作的数据帧为 4 个；当完成对第 i 个数据帧的操作后，第 i+4 个输入数据帧重新采用对第 i 个数据帧进行操作的数据寄存器、中间数据寄存器和迭代结果数据寄存器。

一种 LDPC 及其缩短码的高速译码方法

技术领域

[0001] 本发明属于编译码技术领域，涉及一种 LDPC 及其缩短码的高速译码装置。

背景技术

[0002] LDPC 码最早在 20 世纪 60 年代由 Gallager 在他的博士论文中提出，但限于当时的技术条件，缺乏可行的译码算法，此后的 35 年间基本上被人们忽略，其间由 Tanner 在 1981 年推广了 LDPC（低密度奇偶校验码）码并给出了 LDPC 码的图表示，即后来所称的 Tanner 图。1993 年 Berrou 等人发现了 Turbo 码，在此基础上，1995 年前后 MacKay 和 Neal 等人对 LDPC 码重新进行了研究，提出了可行的译码算法，从而进一步发现了 LDPC 码所具有的良好性能，迅速引起强烈反响和极大关注。目前已广泛应用于深空通信、光纤通信、卫星数字视频和音频广播等领域。LDPC 码已成为第四代通信系统 (4G) 强有力的竞争者，而基于 LDPC 码的编码方案已经被下一代卫星数字视频广播标准 DVB-S2 采纳。

[0003] LDPC 信道编码技术首次应用于我国航天领域是在 2010 年 10 月 1 日发射升空的嫦娥二号。其下行数据速率是 12M/S，也就是 LDPC 编译码器均工作在 12MHz 的频率下。随着我国空间技术的发展，空间探测中各种有效载荷需要下传的数据量会越来越大，这就需要具有良好纠错能力的 LDPC 码能够有效的适应高码速率的要求。

[0004] 当前 LDPC 译码的硬件实现多是基于码长较短的码型（几十到几百比特的码长），且运算速率较低（十几兆到几十兆），而且由于硬件资源的限制，实现功能较强的 LDPC 译码器需要性能很好的芯片、甚至需要针对特定算法专门设计，硬件实现平台的成本很高，不利于大规模应用。

发明内容

[0005] 本发明的技术解决问题是：克服现有技术的不足，提供了一种 LDPC 译码装置及方法。本发明以 CCSDS 为标准对 (8176,7154) 码型，在较低成本的硬件平台上实现了对高速数据流的译码，解决了长码长的低密度奇偶校验码在高码率下的译码问题。

[0006] 本发明的技术解决方案：

[0007] 通过以下步骤进行实现：

[0008] (1) 对译码数据的第 i 个数据帧进行判断：

[0009] 若输入为 LDPC 数据，则按所使用的输出数据寄存器的地址将 LDPC 数据中帧头数据的原码进行存储；并由输出数据寄存器输出前一运算周期中数据帧的帧头数据；

[0010] 若输入为 LDPC 的缩短码数据，则按所使用的输出数据寄存器的地址将 LDPC 数据中帧头和填充数数据的原码进行存储；并由输出数据寄存器输出前一运算周期中数据帧的帧头和填充数数据；

[0011] (2) 按照所使用的中间数据寄存器和迭代结果数据寄存器的地址存储第 i 个数据帧中的有效数据；并由步骤 (1) 中所述的输出数据寄存器输出前一运算周期中的译码数据；

[0012] (3) 第 i 个数据帧存储完毕后，对第 i 个数据帧进行校验节点和变量节点的迭代运算操作；同时对输入的第 i+1 个数据帧从步骤 (1) 开始利用不同的输出数据寄存器、中间数据寄存器和迭代结果数据寄存器进行操作；

[0013] (4) 当对第 i 个数据帧的操作完成并输出后，重新采用对第 i 个数据帧进行操作的数据寄存器、中间数据寄存器和迭代结果数据寄存器对新到达的数据帧进行操作；

[0014] 在上述步骤中，同时对输入的所有数据帧进行操作，对一个数据帧进行存储后操作所用时间等于对该单个数据帧之后输入的其余各数据帧进行存储所用时间的累加和。

[0015] 所述步骤 (3) 中的校验节点和变量节点的迭代运算操作分别由校验节点运算模块和变量节点模块实现，并可在连续到达的数据帧中进行复用。

[0016] 同时进行操作的数据帧为 4 个；当完成对第 i 个数据帧的操作后，第 i+4 个输入数据帧重新采用对第 i 个数据帧进行操作的数据寄存器、中间数据寄存器和迭代结果数据寄存器。

[0017] 本发明与现有技术相比具有如下优点：

[0018] (1) 采用 2 个校验节点模块和 16 个变量节点模块的半并行数据处理方式，使资源占用和运算效率取得平衡优化。相对于完全串行的处理方式，可以使得在缓存 3 帧数据的情况下迭代运算次数增加一倍；相对于完全并行的处理方式，可以大量节省芯片资源占用，使得在较低端芯片上实现此规模的译码成为可能。另外，以此种方法为基础，为相邻两帧数据分配独立的运算单元，在少量增加硬件资源支出的情况下可以进一步提高系统的工作频率。根据实际工程需求，对时序控制部分作适当修改，还可以在降低工作频率百分之五十的情况下再增加一倍的迭代次数，提供更优的译码效果。

[0019] (2) 采用半并行的数据处理方式也会产生大量的中间数据。如果在对码长较短 LDPC 码进行译码的情况下，可以采用 FPGA 的片内寄存器资源进行临时存储。但对 (8176, 7154) 这种码长较长的 LDPC 码进行译码，片内的寄存器资源就会不足，而且运算中频繁的对不同位置的寄存器进行读写访问也会降低系统的工作频率。基于此，设计中采用了 BlockRam 作为中间数据的存储，有效利用 FPGA 内部的专用电路从而大量节省了逻辑和布线资源，解决了以上问题。另外，由于 BlockRam 寻址读写的特点，也增加了设计的可移植性。

[0020] (3) 多层流水线结构的控制方式实际上是对半并行的数据处理方式的一种优化。从数据流状态上看，这种并行结构是动态的，在提高系统工作频率的同时，解决了译码数据流输入、输出的连贯问题。

附图说明

[0021] 图 1 为本发明结构示意图；

[0022] 图 2 为 Min-sum 算法校验矩阵 Tanner 图；

[0023] 图 3 为校验节点运算示意图；

[0024] 图 4 为本发明工作流程图。

具体实施方式

[0025] 下面就结合附图对本发明做进一步描述。

[0026] LDPC 码的译码算法有：

[0027] 1. SPA(Sum-Product Algorithm)，Probability-domain；

[0028] 2. SPA，Log-domain；

[0029] 3. Min-sum(Log-domain SPA 的简化算法)，及其各种修正版本；

[0030] 4. BF(Bit-Flipping)；

[0031] 5. MLG(Majority-Logic)，只对循环 LDPC 码可以应用。

[0032] 以上所列的算法，复杂性和误码性能依次降低。其中，Probability-domainSPA 计算太过复杂，BF 算法和 MLG 算法不能提供良好的误码性能，实际中均不宜采用。可以使用的几种算法中，修正的 Min-sum 算法误码性能可以接近 SPA，又有相对低的复杂度，因此，经综合考虑本设计采用了修正的 Min-sum 算法。

[0033] Min-sum 算法译码迭代实施方式：

[0034] $$H=\begin{bmatrix}1&1&1&1&0&0&0&1&0&0\\1&0&0&0&1&1&1&0&0&0\\0&1&0&0&1&0&0&1&1&0\\0&0&1&0&0&1&0&1&0&1\\0&0&0&1&0&0&1&0&1&1\end{bmatrix}$$

[0035] Min-sum 算法基于 LDPC 码校验矩阵 H 的 Tanner 图进行迭代译码。我们以如下校验矩阵为例来说明它的运算过程。H 的 Tanner 图如图 2 所示。想象 c_i 和 f_j 分别为校验节点和变量节点的运算单元，c_i 和 f_j 之间的连线为双向通道，记从 c_i 和 f_j 的连线（即 c_i 的输出，f_j 的输入）为 q_{ij}；从 f_j 到 c_i 的连线（即 f_j 的输出，c_i 的输入）为 r_{ji}。

[0036] Min-sum 首先初始化 f_j 的输入 q_{ij}。q_{ij} 经 f_j 运算后输出 r_{ji}，再经 c_i 运算后输出更新 q_{ij}，完成一次迭代。直到解出正确的码子，或者达到预先设定的最大迭代次数，结束整个迭代过程，解码完毕。具体步骤如下：

[0037] 1. 初始化 q_{ij}：接收到的含噪声 (Var = σ)^2 的码子 $\gamma = (\gamma_0, \gamma_1, \cdots, \gamma_9)$，将 $q_i = 2\gamma_i/\sigma^2$ 作为原始输入，并将所有 q_{ij} 初始化为：

[0038] $q_{ij} = q_i = 2\gamma_i/\sigma^2$

[0039] 2. 运算单元 f_j：f_0 为 5 输入 5 输出，其余的 f_j 均为 4 输入 4 输出。输出与输入的关系为：

[0040] $$r_{ji} = \prod_{i' \neq i} sign(q_{ij}) \cdot \min_{i' \neq i} \left| q_{i'j} \right|$$

[0041] 即每个输出的符号为除去相应的输入 q_{ij} 外，其他输入的符号的乘积；每个输出的大小为：

[0042] $$r_{21} = \prod (sign(q_{42}), sign(q_{72}), sign(q_{82})) \cdot \min (|q_{42}|, |q_{72}|, |q_{82}|)$$

[0043] 3. 运算单元 c_i：c_7 为 3 输入 3 输出的运算单元，其余所有的 c_i 均为 2 输入 2 输出。具体关系说清楚，，输入与输出的关系为：

[0044] $$Q_i = \sum_{j'} r_{j'i} + q_i$$

[0045] $q_{ij} = Q_i - r_{ij}$

[0046] 即每个输出为除去相应的输入外，其他输入的和，再加上该运算单元 c_i 所对应的原始输入 $q_i = 2\gamma_i/\sigma^2$。中间结果 Q_i 为下一步判断码子做准备。以 c_7 为例，见图 3，输出到 f_0 的 q_{70} 为：

[0047] $Q_7 = r_{07}+r_{27}+r_{37}+2\gamma_7/\sigma^2$

[0048] $q_{70} = Q_7-r_{07} = r_{27}+r_{37}+2\gamma_7/\sigma^2$

[0049] 4. 计算码子与判断停止条件：步骤 2 和 3 完成了一次迭代，此时用 Q_i 作判决码子：

[0050] $$c_i=\begin{cases}1, Q_i<0\\0, Q_i>0\end{cases}$$

[0051] 如果 $c_iH^T = 0$，即译码成功，或者达到了预先设定的最大迭代次数，则停止迭代，否则，继续步骤 2 和 3。

[0052] NASA 在 2005 年 9 月提出了一种由有限几何方法构造的 (8176,7154)LDPC 码，并于 2006 年 5 月将其确定为取代 R-S 加卷积码的常规信道编码标准。CCSDS(The Consultative Committee for Space Data Systems，太空数据系统咨询委员会）于 2006 年 8 月发表的技术文件推荐该码为近地应用的标准编码（参考：《CCSDS 131.1-O-1，Aug 2006》）。

[0053] 该码有如下特点：

[0054] 1、有限几何码，可在 BER = 10^{-10} 时不出现 error-floor；

[0055] 2、有限几何码，有很快的迭代译码收敛速度；

[0056] 3、准循环码，系统码，可用逻辑电路实现，不需要实数运算；

[0057] 4、码率 7/8，不明显增加传输带宽。

[0058] 基于以上特点，本发明采用了由有限几何方法构造的 (8176,7154)LDPC 码。

[0059] 硬件实现采用 xilinx 公司的可编程逻辑器件，充分利用其中的 Ram 资源代替查找表。采用流水线的控制方式，复用算法中关键的逻辑功能部分，有效的提高工作频率，降低了资源占用率。

[0060] 译码采用 Min-sum 算法进行。其核心是校验节点运算和变量节点运算，主要问题是在对码长比较长的 LDPC 码进行译码的时候，要进行的运算量很大，同时产生大量的中间数据需要存储。以一帧长 8176bits 为例，进行一次迭代就需要进行 1022 次校验节点运算和 8176 次变量节点运算，而完成一帧数据的译码需要进行 10 迭代。

[0061] 根据 LDPC 校验矩阵的特点，采用了以下方法实现了对 100Mbps 数据流的连续译码：

[0062] 2 组校验节点运算并行，和串行结构相比，降低一半校验节点运算周期；

[0063] 16 组变量节点运算并行，和串行结构相比，降低一半变量节点运算周期；

[0064] 流水线控制运算时序，提高了数据的处理能力，提高系统工作速度；

[0065] 两帧数据复用一组校验节点运算模块和变量节点运算模块，降低了资源占用率；

[0066] 利用 FPGA 内部 Ram 取代寄存器存储中间数据；

[0067] 按照功能划分，本发明的译码装置包括数据转换和控制模块、数据输出模块、n 个中间数据寄存器、n 个迭代结果寄存器、n 个输出数据寄存器、n/2 个校验节点运算模块、n/2 个变量节点运算模块。具体结构如图 1 所示，以下分别说明各部分的实现功能：

[0068] 数据转换和控制模块

[0069] LDPC 译码的控制核心，完成对原始输入数据的格式转换，控制中间数据寄存器、迭代结果寄存器和输出数据寄存器的输入选择、地址选择、读写操作；分配数据流走向；控制所有功能单元的时序。

[0070] 数据存储模块

[0071] 数据存储模块包括中间数据寄存器、迭代结果寄存器、输出数据寄存器三部分。为迭代运算提供临时的存储空间，当前帧完成运算后分配给下一帧使用，是数据交换的节点。

[0072] 中间数据寄存器、迭代结果寄存器、输出数据寄存器采用 blcok ram 进行实现。blcok ram 是 FPGA 中的特定资源，可以实现双口的同时读取，且不占用 FPGA 中的其他查找表资源 (look-up table)。

[0073] 校验节点运算模块

[0074] 完成校验节点运算。在 CCSDS 建议的 (8176,7154)LDPC 码下，其核心是 32 个 7bits 数据输入，计算其中的最小值和次小值，把次小值赋给原来最小值位置的数据，把最小值赋给其余数据，输出还是 32 个 7bits 数据。可以根据数据转换和控制模块给出的数据输入选择信号对输入的两组 32 个 7bits 数据中的一组进行计算。

[0075] 在具体实现时，校验节点运算模块由包括 4 个子模块，其作用是在 8 个输入数据中选择其中的最小值和次小值输出，并输出最小值数据的位置。

[0076] 变量节点运算模块

[0077] 完成变量节点运算。同样 CCSDS 建议的 (8176,7154)LDPC 码，根据计算输入的 5 个 7bits 位宽数据，计算输入数据的和并输出；计算和值与各位置数据的差并输出；缩短数值过大的数据。

[0078] 数据输出控制模块

[0079] 多路选择器，根据数据转换和控制单元给出的输入数据选择信号依次连续输出 4 帧译码后数据。

[0080] 如图 4 所示，为本发明装置工作流程图。译码装置前端的解调器输出的同步数据流可以是单比特（对应硬判决译码），也可以是多比特量化（对应软判决译码）。数据进入译码装置后，时序控制模块先对数据格式进行转换，将原始数据（编码数据）的原码输出给中间数据寄存器和将补码输出给迭代结果寄存器。并根据 CCSDC 建议的校验矩阵的结构特点，将一帧的数据存储在 32 个中间数据寄存器和 16 个迭代结果寄存器内。当所有数据写入完毕后，对中间数据寄存器和迭代结果寄存器的地址进行置位，开始同时将 32 个中间数据寄存器对应地址的数据读取至校验节点运算模块，待校验节点运算完毕后，用此结果更新之前输入的数据。采用两组校验节点运算模块同时工作的方式，一共对 1022 组数据进行校验节点运算，完成对一帧数据的一次校验运算更新。然后再对 32 个中间数据寄存器和 16 个迭代结果寄存器的地址进行置位，同时将 32 个中间数据寄存器和 16 个迭代结果寄存器对应地址的数据读取至变量节点运算模块，待变量节点运算完毕后，用此结果更新之前输入的数据。采用 16 个变量节点运算模块同时工作的方式，一共对 8176 组数据进行变量节点运算，完成对一帧数据的一次变量运算更新。

[0081] 以上是对一帧数据进行一次完整的迭代运算的过程，经过 10 次这样的迭代过程，一帧数据的译码结束。为了达到较高的速率，同时采用了并行、缓存、流水线处理的方法进行数据处理。当第一帧原始数据读取完毕开始进行处理的同时，第二帧原始数据开始读入，

存储到第二组 32 个中间数据寄存器和 16 个迭代结果寄存器内，第三、四帧原始数据的读入过程相同，即在原始数据的输入过程中是第一至四帧依次进入流水线结构，每帧数据各自占用一组中间数据寄存器和迭代结果寄存器。在第一帧数据进行译码的同时，从第一帧数据所进行的操作看，另外三帧数据相当于缓存在中间数据寄存器和迭代结果寄存器内。实际上从整个数据流的循环处理过程来看，任何一帧数据的译码过程都与另外三帧数据的缓存相对应。接下来，第二、三、四帧数据依次进入迭代译码的流水线过程进行处理。当数据流源源不断地输入时，不仅每帧内部的校验节点运算和变量节点运算是并行的，而且帧和帧之间数据处理也是并行的。可以看出在任一时刻，都会有一帧数据处于输入 / 输出状态，另外三帧数据处于迭代译码状态。从时序控制的角度看，每帧数据的处理过程均由状态机控制，从原始数据的输入到译码结果的输出，共约 50 个状态；以四帧数据为一个周期作循环译码，处理连续数据流需要经历 200 种状态组合。

[0082] 本发明未详细说明部分属本领域技术人员公知常识。

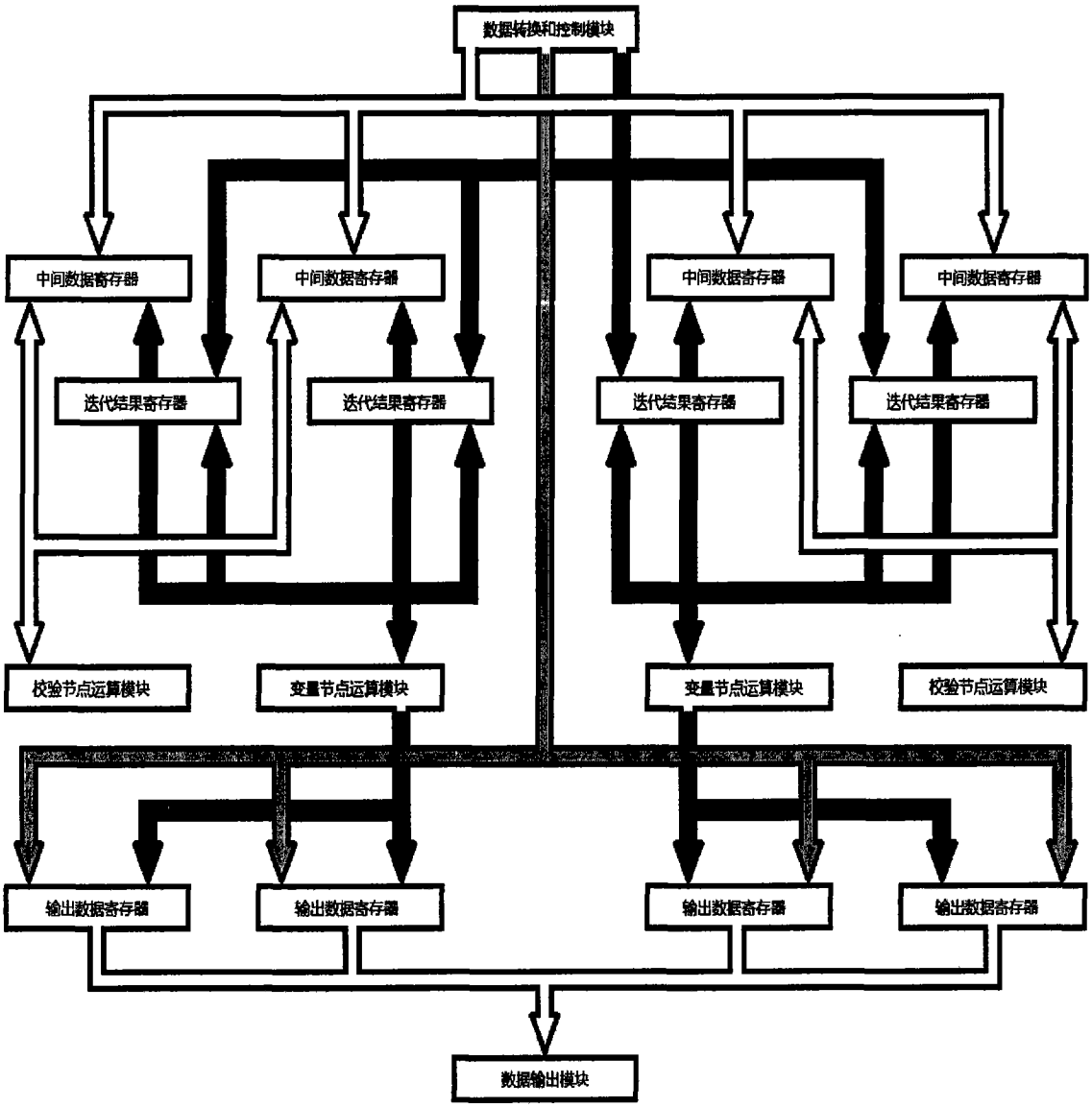

数据转换和控制模块
中间数据寄存器
中间数据寄存器
中间数据寄存器
中间数据寄存器
迭代结果寄存器
迭代结果寄存器
迭代结果寄存器
迭代结果寄存器
校验节点运算模块
变量节点运算模块
变量节点运算模块
校验节点运算模块
输出数据寄存器
输出数据寄存器
输出数据寄存器
输出数据寄存器
数据输出模块

图 1

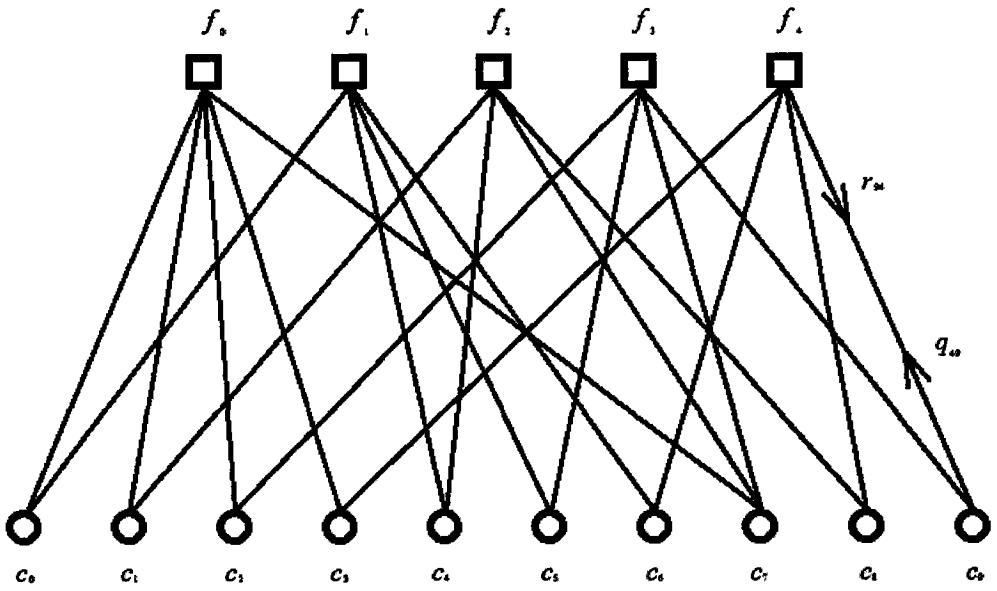


图 2

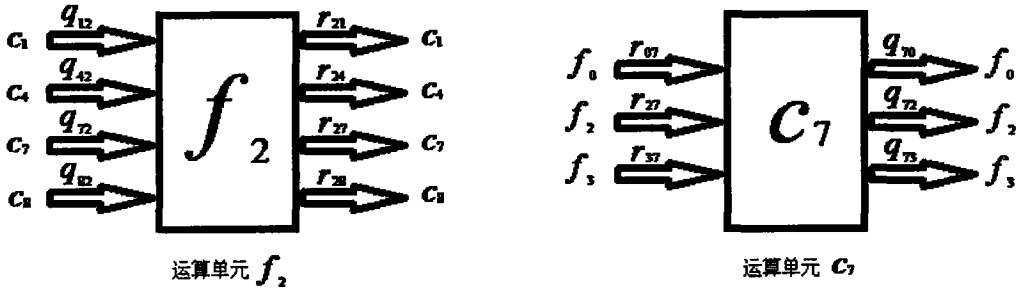

运算单元 f_2
运算单元 C_7

图 3

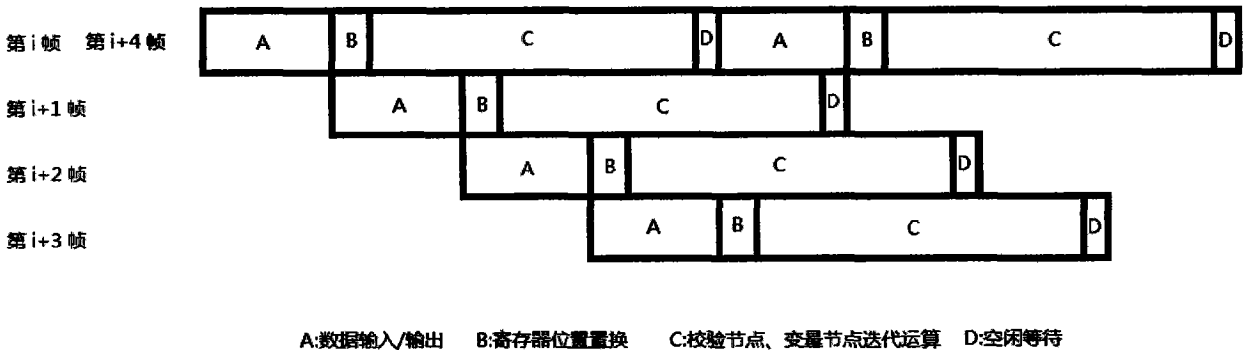

第 i 帧 第 i+4 帧
第 i+1 帧
第 i+2 帧
第 i+3 帧
A
B
C
D
A:数据输入/输出 B:寄存器位置置换 C:校验节点、变量节点迭代运算 D:空闲等待

图 4