NoC节点编码及路由算法的研究

杨晓强[1] 谭耀东[2] 朱宁洪[3] 韩俊刚[1,2]

(西安电子科技大学微电子学院 西安 710071)[1] (西安邮电学院计算机系 西安 710061)[2]

(西安科技大学计算机系 西安 710061)[3]

摘 要 NoC的设计和实现受到芯片的面积、功耗、深亚微米效应的限制。将拓扑结构和节点编码相结合,提出一种基于约翰逊码的二维平面编码。该编码隐含了Torus网络拓扑结构以及网络节点之间的连接关系并且有很好的扩展性,能够简化Torus拓扑结构上路由算法的实现和降低硬件成本。基于此编码和利用X-Y路由的路由确定性特点,提出改进X-Y路由,在中间节点只需要3或5个逻辑运算,降低路由的计算复杂性和硬件成本。最后,进行了节点结构设计。提出的编码不仅用于NoC的路由方面而且在NoC任务映射方面有重要应用。

关键词 片上网络,拓扑结构,节点编码,片上路由

中图法分类号 TP331.1 **文献标识码** A

Node Encoding and Routing Algorithm for Network on Chip

YANG Xiao-qiang[1] TAN Yao-dong[2] ZHU Ning-hong[3] HAN Jun-gang[1,2]

(School of Microelectronics, Xidian University, Xi'an 710071, China)[1]

(Department of Computer Science, Xi'an Institute of Post & Telecommunications, Xi'an 710061, China)[2]

(Department of Computer, Xi'an University of Science and Technology, Xi'an 710054, China)[3]

Abstract NoC design encounters the limitation of chip area, power assumption and the physical effects in deep sub-micron and nano-micron technology. By the combination of network topology with corresponding node coding, this paper proposed a two-dimensional plane encoding based on Johnson Code which implies the relation between neighbouring nodes, has a good scalable characteristic and can simplify the routing algorithms and implementations of NoC. Utilizing the code proposed and the deterministic property of X-Y routing, the improved algorithm for X-Y routing was also presented which was implemented with three or five logic operations in middle nodes. At last, the node structure of NoC was designed. The code proposed is useful not only in NoC routing but also in NoC task mapping.

Keywords NoC, Topology architecture, Node encoding, NoC routing

半导体技术的快速发展以及芯片上系统应用复杂度的不断增长,使得片上互连结构的吞吐量、功耗、延迟以及时钟同步等问题更加复杂。针对此状况,近几年提出了将通信机制与计算资源分离的片上网络(NoC, Network-on-Chip)[1]。NoC的核心思想是借鉴计算机网络的互连和通信技术,从体系结构上解决片上通信的瓶颈问题,充分利用分布式计算机系统的通讯方式,用路由和分组交换技术代替传统的总线通讯方式,使得NoC提供模块化、可升级、可靠性和高带宽通信结构[1-3]。NoC的设计和实现受到芯片的面积、功耗、深亚微米效应的影响和限制[1]。在NoC中采用路由算法实现路由时,要求设计出比宏观网络更加简单的路由器硬件结构来实现路由和交换等功能,以满足有限的硅片资源和低延时等VLSI实现要求[4]。

1 NoC拓扑结构与编码

Mesh结构具备硬件实现简单、网络扩展性好等方面的优点,被作为NoC主要的拓扑结构[1,4]。Torus结构是Mesh结构的一种改进,将Mesh结构中每个行、列的首、尾节点进行互连,从而消除了Mesh结构的边界效应。并行计算机采用了超立方体结构(Hypercube),节点采用了格雷码对其节点编码,能够实现简单的路由机制[5]。表1对Mesh, Torus和Hypercube 3种网络的特性进行了对比。低维网络(如二维格网)的成本有效性要比高维网络(如超立体)高得多。Hypercube结构一般不超过5维或6维[5,6],当节点数n超过5,节点与节点的链路关系越来越复杂,不仅在网络拓扑上描述链路越来越困难,同时链路数目过多,也导致了成本过高、网络扩展性差[6]、布线复杂,所以很少作为NoC的拓扑结构。

表1 N个节点的3种网络特性表

网络	直径	对分宽度	弧连通性	成本(链路数)
Mesh	$2(\sqrt{n}-1)$	$\sqrt{n}$	2	$2(n-\sqrt{n})$
Torus	$2\lfloor\sqrt{n}/2\rfloor$	$2\sqrt{n}$	4	2n
Hypercube	$\log_2 n$	n/2	$\log_2 n$	$n(\log_2 n)/2$

到稿日期:2008-04-16 本文受国家自然科学基金(90607008)资助。

杨晓强(1971-),男,博士生,主要研究方向为集成电路设计等,E-mail:xyangq@sina.com;谭耀东(1979-),男,硕士生,主要研究方向为集成电路设计等;朱宁洪(1969-),男,讲师,主要研究方向为多媒体技术等;韩俊刚(1943-),男,教授,主要研究方向为形式化验证和综合、人工智能、VLSI设计等。

Torus 既具有 Mesh 结构硬件实现简单、网络扩展性好等方面的优点，又兼有 Hypercube 结构相对于 Mesh 能减少最短路径距离和提高容错的能力。目前在 Torus 中，采用 X-Y 路由或 DyAD routing 路由是基于(x,y)节点坐标编码[7,8]，其路由机制相对于采用格雷编码超立方体结构的路由算法复杂。如果采用 Torus 结构的 NoC 借鉴 Hypercube 结构采用格雷编码简单的路由机制，更有利于 NoC 的硬件实现。但基于格雷码的网络的节点位数每增加一位，为了保证循环的特性，就需要将节点增加原来的一倍，这样对网络扩展的有效性有较大影响。当我们只需扩展跟原来网络相比很少的节点时，增加一位会导致过多的空节点，一方面可能浪费资源和空间，另一方面对路由也带了不利影响。为了解决上述问题，本文提出基于约翰逊码的二维平面编码对 Torus 拓扑网络的节点进行编码。

2 基于约翰逊码的二维编码

2.1 二进制循环单位距离码

定义 1 二进制循环单位距离码是自然数$(0,1,2,\cdots,n-2,n-1)$映射成由“0”和“1”组成的一组二进制数据，且满足：

(1)任意两个相邻的数据有且仅有一位不同(单位距离性质)；

(2)数据序列中第一个和最后一个的数据有且仅有一位不同(循环性质)。

定义 2 对于任意的一个自然二进制编码：$A_{n-1}A_{n-2}\cdots A_1A_0$ 对应的格雷编码(Gray Code)为：$A_{n-1}(A_{n-1}\oplus A_{n-2})\cdots(A_3\oplus A_2)(A_2\oplus A_1)(A_1\oplus A_0)$(其中“$\oplus$”代表位异或运算)。

定义 3 对于递增序列$(0,1,2,\cdots,p-1,p,p+1,\cdots,n-2,n-1)$，其中$n=2t,t=0,1,2,\cdots$，采用$m=n/2$位来进行编码，如果编码后的数据序列具有定义 1 的性质并且满足：① 当$p<m$时，则p的编码形式为$Q=F_{m-1}\cdots F_k\ T_{k-1}\cdots T_0$；其中$k=p$(“$F_{m-1}\cdots F_k$”代表为全“0”的序列部分，“$T_{k-1}\cdots T_0$”代表全“1”的序列部分，并且当$k=0$时，$Q$为$m$位的全“0”序列)；② 当$p\geqslant m$时，则$p$的编码形式为$Q=T_{m-1}\cdots T_k\ F_{k-1}\cdots F_0$；其中$k=p-m$(“$F_{k-1}\cdots F_0$”代表为全“0”的序列部分；“$T_{m-1}\cdots T_k$”代表全“1”的序列部分，并且当$p=m$时，$Q$为$m$位的全“1”序列)，称该序列的编码为二进制约翰逊编码(Johnson Code)。

由上述定义可以看出，格雷码和约翰逊码是二进制循环单位距离码。

2.2 格雷码、约翰逊编码与自然二进制码之间的转换

格雷码与自然二进制码之间的转换：

(1)依据定义 2 的映射关系将自然二进制码转换为格雷码。

(2)任意的一个二进制格雷码：$A_{n-1}A_{n-2}\cdots A_1A_0$ 对应的自然二进制编码为 $B_{n-1}B_{n-2}\cdots B_1B_0$(其中 $B_k=A_k\oplus B_{k+1}$，$0\leqslant k<n-1$ 且 $A_{n-1}=B_{n-1}$)。

通过上述的转换关系可以建立一个由n位组成的2^n个格雷码数据序列与n位组成的2^n个自然二进制数据序列的一一对应关系。

约翰逊编码与自然二进制码之间的转换：

(1)依据定义 3 的映射关系将自然二进制码转换为约翰逊编码。

(2) 任意的一个约翰逊编码$(A_{n-1}A_{n-2}\cdots A_p,\cdots A_2A_1A_0)$，存在一个$m=\lceil\log_2 2n\rceil$位的自然二进制码组来与之对应：

①当A_{n-1}为“0”时，对应的自然二进制码是把该约翰逊编码中“1”的个数相加所得的二进制码；

②当A_{n-1}的最高位为“1”时，对应的自然二进制码是由m加上该约翰逊编码中“0”的个数形成的二进制码。

2.3 基于约翰逊码的二维平面编码

将二进制循环单位距离码推广到二维平面上时，具备下述两个性质：

(1)在平面上任意相邻的两个节点编码有且仅有一位不同(单位距离性质)；

(2)在平面上任意数据行或列的首尾数据有且仅有一位不同(循环性质)。

由上述性质可以看出二进制循环单位距离码在二维平面上任意的行列的性质都满足在一维序列上的性质，同时二维平面上具备更多的一些性质：

(1)在平面上，任一个节点有 4 个相邻节点(当节点位数大于或等于 4 时)；

(2)在平面上，每行每列都存在一维的单位距离循环圈。

定义 4 在二维平面上对于任意的两个节点，如果它们两者的节点编码有且仅有相差一位时，两者称为是相邻的；

定义 5 在定义 3 的基础上，如果在二维平面上两个节点是相邻的，那么称它们之间存在(直接)链路。

定义 6 基于约翰逊码 2 维平面编码(2_d JCode)是一个二维循环单位距离码。2_d JCode=JCode_Y+JCode_X，$w=h+k$，其中，w,h,k 分别是 2_d JCode，JCode_Y 及 JCode_X 的 bit 长度。“+”是连接操作。并且满足：

(1) 在每一维上相邻的两个节点的编码有且仅有一位不同(单位距离性质)；

(2) 在每一维上第一个和最后一个编码有且仅有一位不同(循环性质)。

按照上述定义，可以构造出基于约翰逊码的二维平面编码：对于n位节点，把其中k位的$2k$个按自然二进制码顺序排列且转换成一维约翰逊码的二进制数，作为二维平面节点的横坐标；把剩下的$n-k$位的$2n-k$个数据按同样方式编码形成的二进制码，作为二维平面节点的纵坐标；然后，节点的行列坐标组合在一起，就构成了基于约翰逊码的二维平面编码。任意一个节点的坐标都可以表示为：$N[n-1,\cdots,k,k-1,\cdots,2,1,0]$，其中“$n-1,\cdots,k$”代表节点的纵坐标，“$k-1,\cdots,2,1,0$”代表节点的横坐标。图 1 给出一个 4×6 基于格雷码的二维平面编码图 (其中$n=5,k=3$)，前 2 位为纵坐标，后 3 位为横坐标。

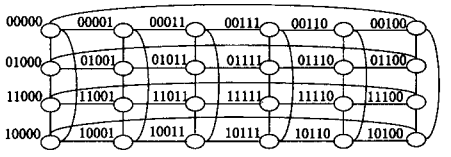

图 1 4×6 基于约翰逊码的二维平面编码 ($n=5,k=3$)

基于约翰逊码二维平面编码形成的网络拓扑是一个

$2k\times2h$ 的 Torus 网络，所有对于任意 n 位节点的二维约逊码形成的网络中，任意节点只存在 4 条直接链路，4 个相邻节点。

仿照基于约翰逊码的二维平面编码的造出方法，可以构造出基于格雷码的二维平面编码：对于 n 位节点，把其中 k 位的 2^k 个按自然二进制码顺序排列且转换成一维格雷码编码的二进制数，作为二维平面节点的横坐标；把剩下的 $n-k$ 位的 2^{n-k} 个数据按同样方式编码形成的二进制码作为二维平面节点的纵坐标；然后，节点的行列坐标组合在一起，就构成了基于格雷码的二维平面编码，任意一个节点的坐标都可以表示为：$N[n-1,\cdots,k,k-1,\cdots,2,1,0]$，其中“$n-1,\cdots,k$”代表节点的纵坐标，“$k-1,\cdots,2,1,0$”代表节点的横坐标。图 2 给出一个 4×8 基于格雷码的二维平面编码图（其中 $n=5,k=3$），前 2 位为纵坐标，后 3 位为横坐标。

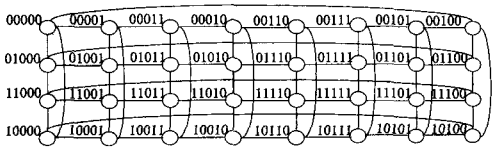

图 2　4×8 基于格雷码的二维平面编码（$n=5,k=3$）

基于格雷码的二维平面编码网络扩展性差，节点位数每增加一位，需要将节点增加原来的一倍。当只需扩展跟原来网络相比很少的节点时，增加一位会导致过多的空节点，一方面可能浪费资源和空间，另一方面对路由也带走了不利影响，降低网络扩展的有效性。当采用基于约翰逊码的二维平面编码对于规模为 $2k\times2h$ 的 Torus，每增加一位，节点个数只需增加 $4k$ 或 $4h$ 个。同时，基于约翰逊码的二维平面编码构成的 Torus（当 $k\geqslant2,h\geqslant2$ 时），任一节点只有 4 个相邻节点，使得整个网络的链路数大大减少，其成本有效性比二维格雷编码的 Hypercube 网络高。

3　基于约翰逊码的二维编码 NoC 路由

Torus 结构的路由算法比较多，有 X-Y 路由、X-Y 交错路由和 DyAD 路由等，比较简单而且常用的是 X-Y 路由算法[9,10]。Torus 结构与约翰逊编码相结合下的路由机制相对于一般情况下以 (x,y) 坐标标识节点的路由算法有以下几个方面优点。

(1)最短路径长度的计算。对于任意的 $2n\times2m$ 的 Torus 网络，则每个结点编码可以用（$N_{n+m-1}\cdots N_k\cdots N_m\ N_{m-1}\cdots N_i\cdots N_0$），用这种编码形成的节点地址 S 和 D 之间的最短路径可以表示为：$C=Haming(S\oplus D)$，其中“$\oplus$”标识位异或运算。“Haming”函数代表把 1 的个数相加的运算，即结果为海明距离。通过这种运算可以用简单的硬件机制实现，而采用 (x,y) 坐标标识节点，已知节点规模系数 n 和 m 后对节点之间求曼哈顿距离[10]。

(2)最短路径条数以及路径遍历节点链的计算。最短路径数 $P=MinPath(Haming(S\oplus D))$，其中 $MinPath$ 函数如下描述：

①当采用基于 X-Y 路由时：

if ($Haming(S_{m-1}\cdots S_i\cdots S_0\oplus D_{m-1}\cdots D_i\cdots D_0)=\ =m$)

$dh=2$; else $dh=1$;

if ($Haming(S_{n+m-1}\cdots S_k\cdots S_m\oplus D_{n+m-1}\cdots D_k\cdots D_m)=n$)

$dv=2$; else $dv=1$;

$d=dh*dv$，即 d 为最短路径条数。

②为了平衡网络流量，采用基于 X-Y 交错路由时：

If (($Haming(S_{m-1}\cdots S_i\cdots S_0\oplus D_{m-1}\cdots D_i\cdots D_0)=\ =j$) and

($Haming(S_{n+m-1}\cdots S_k\cdots S_m\oplus D_{n+m-1}\cdots D_k\cdots D_m)=i$))

$d=C_{j+i}$ or $d=C_{j+i}$。

(3)节点中路由信息的产生。用基于约翰逊码的二维编码进行片上网络节点的编码过程隐含全局的路由的信息，当前节点 N（$N_{n+m-1}\cdots N_k\cdots N_m\ N_{m-1}\cdots N_i\cdots N_0$）与目的节点 D 的距离为 $d=Haming(N\oplus D)$，当前节点的地址编码能够得到相邻的 4 个节点的地址编码为：

$left\_node\_id=\ N_{n+m-1}\cdots N_k\cdots N_m\overline{N}_0N_{m-1}\cdots N_i\cdots N_1$；

$right\_node\_id=N_{n+m-1}\cdots N_k\cdots N_m\ N_{m-2}\cdots N_i\cdots N_0\ \overline{N}_{m-1}$；

$up\_node\_id=\overline{N}_mN_{n+m-1}\cdots N_k\cdots N_{m+1}N_{m-1}\cdots N_i\cdots N_0$；

$down\_node\_id=N_{n+m-2}\cdots N_k\cdots N_m\overline{N}_{n+m-1}N_{m-1}\cdots N_i\cdots N_0$。

进一步可以得到周边 4 个节点与目的节点的距离：

$d\_left=Haming(left\_node\_id\oplus D)$；

$d\_right=Haming(right\_node\_id\oplus D)$；

$d\_up=\ Haming(up\_node\_id\oplus D)$；

$d\_down=Haming(down\_node\_id\oplus D)$。

基于约翰逊码的二维平面编码任一节点相邻 4 个节点的地址可以用简单的移位寄存器的实现，而对于采用 (x,y) 标识节点，求相邻节点地址时一方面跟坐标原点的定位有关，另一方面还需要判断节点是否在人为规定的边界节点（实际上 Torus 是没有边界节点的），再根据节点已知的 n 和 m，求出相邻节点地址。同时，采用基于约翰逊码的二维平面编码的 Torus 网络的节点标识可以动态变换，节点标识只是说明了节点之间的相对位置；而采用 (x,y) 坐标是说明了节点之间的绝对位置，坐标变换往往使得内部路由机制也需改动。

本文利用 X-Y 路由的确定性路由特点，对 X-Y 路由做了改进，降低中间节点上路由的计算复杂性。在数据包中增加 2 比特（设 bit_x 和 bit_y）的路由方向标识位，其中 bit_x 为 0 或 1 表示向左或向右，bit_y 为 0 或 1 表示向上或向下。算法设输出端口方向为 out_dir。基于约翰逊码的二维编码改进 X-Y 路由算法的伪代码为：

```
if(数据包在源节点上)
 { if(Haming(left_node_id ⊕ D)<= Haming(right_node_id ⊕ D))
       bit_x=1;
   else bit_x=0;
   if(Haming(up_node_id ⊕ D)<= Haming(down_node_id ⊕ D))
       bit_y=1;
   else bit_y=0;
   if(Nm-1…Ni…N0⊕ Dm-1…Di…D0！ =0)
       if(bit_x==1) out_dir=数据包向左发;
       else out_dir=数据包向右发;
   else if(Nm+n-1…Nk…Nm⊕ Dm+n-1…Dk…Dm！ =0)
           if(bit_y==1) out_dir=数据包向上发;
           else out_dir=数据包向下发;
       else out_dir=已经到达终点;
```

```
}
if(数据包在中间节点上)
    { if(N_{m-1}…N_i…N_0 ⊕ D_{m-1}…D_i…D_0 ! =0)
        if(bit_x==1) out_dir=数据包向左发;
        else out_dir=数据包向右发;
      else if(N_{m+n-1}…N_k…N_m ⊕ D_{m+n-1}…D_k…D_m ! =0)
        if(bit_y==1) out_dir=数据包向上发;
        else out_dir=数据包向下发;
      else out_dir=已经到达终点;
    }
```

X-Y 路由算法采用(x,y)标识节点，数据包所经过的每个节点上路由的实现都必须进行算术运算。在上面的路由算法中，中间的节点路由只进行简单的 3 个或 5 个逻辑运算(bit_x,bit_y 是否为 0 通过一个逻辑运算就能证实)，在源节点上多 4 个异或运算和 2 个比较运算。逻辑运算实现相对算术运算要简单得多。同样地可以描述基于 X-Y 的交错路由。

4 NoC 节点的设计

NoC 采用的 Torus 结构，节点的设计结构如图 3 所示。

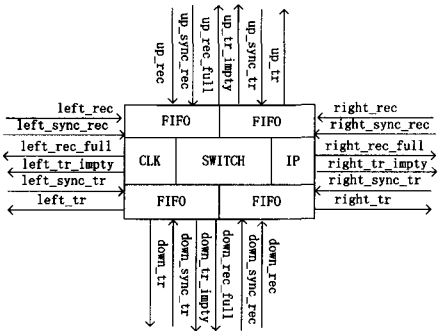

图 3　节点的结构

节点主要由 SWITCH、4 个方向的异步数据缓冲队列 FIFO，IP 以及时钟产生 CLK 等模块组成。其中节点发送 tr 和接收 rec 的数据为并行多位的数据帧格式，帧含有二维平面约翰逊码编址。发送和接收同步信号为 sync_tr 和 sync_rec。tr_full 和 rec_impty 信号表示数据缓冲队列接收已满或发送已空信号，overflow 表示发送有溢出。FIFO 写时钟用相邻节点的时钟进行写，读时钟用本节点的时钟进行读。每个节点内部有一个 IP 模块，IP 与交换单元 SWITCH 共用一个系统时钟，IP 核产生数据帧通过局部链路发给 SWITCH 单元进行转发。

每个节点与周围 4 个节点相连，相连节点之间存在双向传输链路，节点内部各模块之间共享同一个时钟，而节点之间的系统时钟可以不相同，即全局异步局部同步(GALS)。路由采用 3 节中给出的改进的 X-Y 路由方式。节点内部的控制属于集中式控制，由一个控制单元来决定节点的内部交换单元的消息的接收和转发。消息的接收和转发采用的是公平策略，通过轮询节点的 4 个方向的数据缓冲队列 FIFO 中是否存在数据，如果存在，则取数据，然后分析其目的地址和本节点地址决定是转发还是接收；如果不存在，则询问下一个队列，做同样的处理。当需要转发数据时，如果下一个节点的输入 FIFO 未满，则输入至该节点的 FIFO 中；如果已满，下一个节点输出一个高电平的 rec_full 信号至该节点。如果该节点再发送消息至下一个节点，则输出溢出，over_flow 信号输出为高电平，如图 3 所示。

结束语　本文分析了 NoC 的拓扑结构和路由算法，并从超大规模集成电路(VLSI)实现的角度探讨如何选择合适的拓扑结构和相应的节点编码方法。研究了 NoC 的节点编码，提出了基于约翰逊码的二维平面编码。该编码隐含了 Torus 网络拓扑结构以及网络节点之间的连接关系，能够简化片上网络的路由算法的设计和实现。基于该编码和利用 X-Y 路由的路由确定性特点，提出改进 X-Y 路由，降低路由的计算复杂性，并且设计相应的 NoC 节点结构。这种编码方式与 Torus 结构的结合，在 NoC 中具有广泛的应用前景。

参 考 文 献

[1] Bjerregaard T, Mahadevan S. A Survey of Research and Practice of Network-on-Chip. ACM, 2006: 3-15

[2] Nickray M, Dehyadgari M, Afzali-Kusha A. Power and delay optimization for network on chip. Circuit Theory and Design, 2005 (3): 273-276

[3] Rantala V. Network on Chip Routing Algorithms. TUCS Technical Report. No 779. 2006(8): 10-12

[4] Dally W, Towles B. Route packets, not wires: on-chip interconnection networks // Proc. the Design Automation Conference. 2001: 684-689

[5] Rao M V, Chalamaiah Dr N. Hypercube and Its Variant Networks: A Topological Evaluation // Proceedings of the Eighth International Conference on High-performance Computing in Asia-pacific Region. 2005

[6] Lee T-Y, Hsiung P-A, Chen S-J. TCN: Scalable Hierarchical Hypercubes // Proceedings of the Ninth In ernational Conference on Parallel and Dis ributed Systems, 2002

[7] Hu Jingcao. DyAD-Smart Routing for Networks-on-chip // Proc. 41'st ACM/IEEE Design Automation Conf. San Diego, CA, 2004

[8] Kim J, Park D, Theocharides T, et al. A low latency router supporting adaptivity for on-chip interconnects // Proc. of the Design Automation Conf. 2005: 559-564

[9] Bononi L, Concer N. Simulation and analysis of network on chip architectures: Ring, spidergon and 2D mesh // Proc. of the Design, Automation and Test in Europe. 2006

[10] 朱可静. xmesh：一个 xmesh-like 片上网络拓扑结构. 软件学报, 2007, 18(9): 2194-2204

(上接第 47 页)

[5] Berson D, Gupta R, Soffa M L. Integrated Instruction Scheduling and Register Allocation Techniques[C] // Proc. of the 11th International Workshop on Languages and Compilers for Parallel Computing. London: Springer-Verlag, 1998: 247-262

[6] Allen R, Kennedy K. Optimizing Compilers for Modern Architectures, a Dependence-based Approach[M]. San Francisco: Morgan Kaufmann, 2001: 511-547

[7] Leupers R. Compiler Design Issues for Embedded Processors [J]. IEEE Design and Test of Computers, 2002, 19(4): 51-58