一种RISC地址产生器生成算法的设计与实例化

车德亮,赵 宁
(西安微电子技术研究所,陕西 西安 710054)

摘 要:提高功能部件的并行性是开发高性能微处理器的基本途径。在RISC处理器中设计独立的地址产生器可实现算术运算与地址运算并行处理,从而提高RISC处理器的性能。文中根据现今RISC处理器中常用的寻址方式,提出了一种RISC地址产生器生成算法并进行了实例化。实例化结果可作为IP核应用到RISC处理器的设计中。

关键词:RISC处理器;并行性;寻址方式;地址产生器
中图分类号:TP302 **文献标识码**:A **文章编号**:1005-3751(2006)01-0023-04

Design and Realization of an Address Generator Algorithm for RISC

CHE De-liang, ZHAO Ning
(Xi'an Microelectronics Technology Institute, Xi'an 710054, China)

Abstract: *It is an essential way for developing high performancemicroprocessor to improve parallelism of function components. The RISC processorwith AG(Address Generator, AG) can realize the operations of arithmetic and address in parallel, which improve the performance of RISC processor. This paper proposes an algorithm of AG and give an instance for this algorithm, which is accordingwith address mode of modern RISC processor. The instance of AG can be applied as an IP core used in design of RISC processor.*

Key words: RISC processor; parallelism; addressing modes; address generator

0 引 言

如果微处理器ALU既用于算术运算也用于地址运算,那么数据处理与地址产生只能以串行方式进行,这样制约了微处理器处理速度[1~3]。在微处理器中设计独立的地址产生器AG(Address Generator,AG),可以提高处理器性能。以含有片内cache的三级流水线微处理器为例:三级流水为取址级、译码级、执行级,每级的执行时间为t。连续执行n条算术指令,其中每条指令中都有一个操作数需要间接寻址。

微处理器使用ALU产生指令中操作数的地址时,流水线逻辑结构与时空图如图1所示。流水线效率

$$\eta_1 = S_{used1}/S_{total1} = 4n/3(2n+2) \quad (1)$$

微处理器使用独立AG产生指令中操作数的地址时,流水线逻辑结构与时空图如图2所示。流水线效率

$$\eta_2 = S_{used2}/S_{total2} = n/(n+2) \quad (2)$$

指令条数$n \to \infty$时,加速比$S_{peedup} = \lim(\eta_2/\eta_1) = 1.5$。说明使用独立的AG可使流水线效率提高到1.5倍。

当然,使用独立的AG来提高流水线的效率是以微处

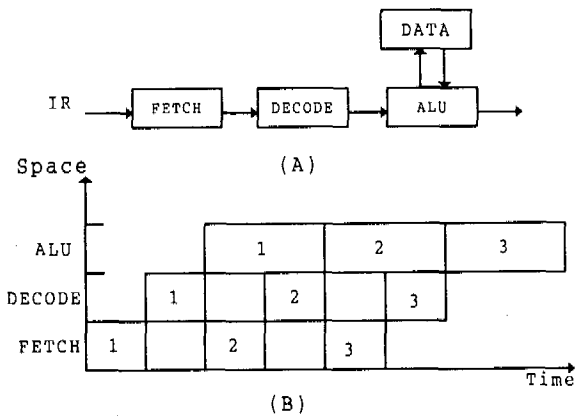


图1 ALU产生操作数地址的流水线结构与时空图

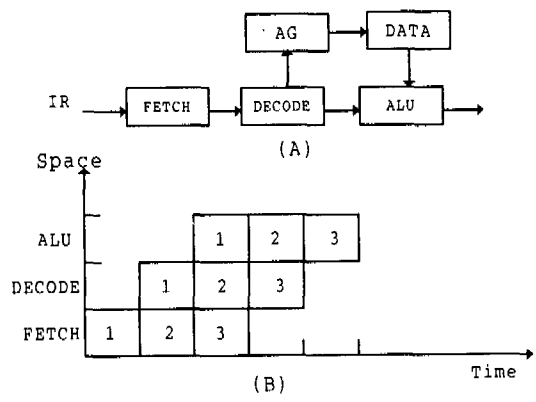


图2 AG产生操作数地址的流水线结构与时空图

收稿日期:2005-04-14
基金项目:国防预研项目(41308010203)
作者简介:车德亮(1975—),男,湖南湘潭人,博士研究生,研究方向为微处理器设计、嵌入式计算机系统结构。

理器的面积和功耗为代价的。如今微电子技术突飞猛进，工艺尺寸不断缩小，使得面积和功耗都不断降低。通过增加硬件模块提高微处理器速度成为开发高速微处理器的主要途径之一。

在微处理器中加入硬件地址产生器，提高微处理器性能的方法，已在多种处理器中使用。如 LSMPP[1] SIMD 处理器内设置了独立的数据地址产生器和程序地址产生器，数据地址产生器是 MPP 阵列数据传送与数据读取可以并行执行的基础；程序地址产生器缩短转移指令的转移时间。超标量 RISC 处理器 INTEL80860[4] 内只设置了程序地址产生器，作用是缩短转移指令的转移时间。TMS320C3x DSP[5] 处理器内设置了数据地址和程序地址计算合并的产生器，它是 TMS320C3x 实现并行指令的基础。

文中根据 RISC 处理器常用寻址方式，设计了一种地址产生器生成算法，并给出了实例设计。本算法有较强的普适性，其实例设计也可作为 IP 核应用到 RISC 处理器的设计中。

1 RISC 处理器常用寻址方式分类

1.1 基址加偏移量类寻址方式

RISC 处理器中常用的直接寻址（也称立即数寻址）、间接寻址、PC 相对寻址，结果地址都可用式(3)表示。这里把能用式(3)表示的寻址方式统称为基址加偏移量类寻址方式。

$$\text{Address} = (\text{reg}) \pm \text{disp} \tag{3}$$

其中，(reg)表示寄存器中存放的基址；disp 表示偏移量；Address 表示寻址方式产生的目标地址。

1.2 位翻转类寻址方式

数字信号处理是 RISC 处理器应用的主要方向之一。为了能够更好地支持数字信号处理算法如 DCT、FFT 等，位翻转寻址方式已成为现代高性能微处理器支持的重要寻址方式之一[5~7]。

N 点离散傅里叶变换的计算公式如式(4)与(5)所示。基 2FFT 是最常见的一种快速傅里叶变换算法，也是最容易硬件实现的 FFT 算法，8 点的基 2FFT 的一般计算流图如图 3 所示。

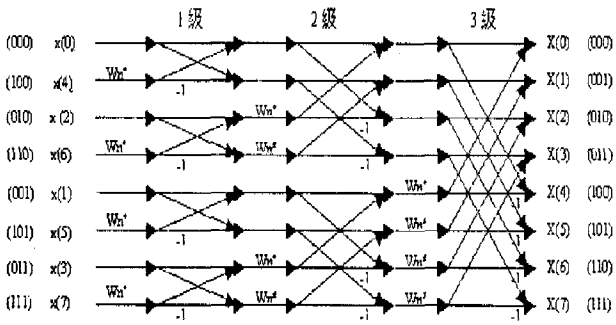


图 3 8 点 FFT 计算流图

$$X_k = \sum_{m=0}^{N-1} x_m W_N^{mk} \tag{4}$$

$$W_N^{mk} = e^{-j\frac{2\pi mk}{N}} \tag{5}$$

$k = 0,1,\cdots,N-1$

从图 3 中可以看出，8 点的基 2FFT 输入数据按混序访问使用，结果数据才能以顺序方式产生。数据混序访问方式的地址可用式(6)表示。把数据地址能用式(6)表示的寻址方式称为位翻转类寻址方式。

$$\text{Address} = H((\text{reg})) \tag{6}$$

其中，(reg)表示寄存器中存放的数据地址；函数 H 表示混序访问方式；Address 表示寻址方式产生的目标地址。

构造不同形式的函数 H，位翻转寻址可有多种方式实现。

1.3 循环类寻址方式

数字信号处理算法如卷积运算、相关运算等需要在存储器中设置一个长度为 L 的循环缓冲器。循环缓冲器作为一个滑动窗口保存将要处理的数据，数据读取/存放根据缓冲器指针。缓冲区指针(reg)以一定步长 Step 改变，当(reg)变化到循环缓冲器的结束位置或超过结束位置时，缓冲区指针将指向(reg) mod L 位置。循环缓冲器数据访问地址可用式(7)表示。把数据地址能用式(7)表示的寻址方式称为循环类寻址方式。

$$\text{Address} = [(\text{reg}) \pm \text{step}] \bmod L \tag{7}$$

其中，(reg)表示寄存器中存放的缓冲区指针；step 表示(reg)改变的步长；Address 表示寻址方式产生的目标地址。

2 地址产生器生成算法

地址产生器生成算法是根据 RISC 处理器常用寻址方式类产生，用 C 语言描述如下：

```
int AG_ algorithm (type,reg,disp,step,L)
int type,reg,disp,step,L
/* type 寻址方式类别标识 */
/* reg 在此表示寄存器存储值 */
{ int Address;
  Switch (type)
  { case 1:Address = reg ± disp;
      break;
  case 2:Address = H(reg);
      break;
  case 3:Address = [reg ± step]mod L;
  }
  Return(Address);
}
```

3 地址产生器生成算法的实例

3.1 基址加偏移量类寻址方式的实例

根据地址产生器生成算法，物理实现基址加偏移量类寻址方式时可直接用加法器完成，可以通过选择不同结构的加法器，来优化基址加偏移量类寻址方式的开销。

物理实现位翻转类寻址时，主要是构造函数 H。本实

例构造函数 H 时,折衷物理实现时速度、面积、功耗的要求,制定以下规则:

[规则 1]:位翻转类数据表长度记为 L,并且 $L = 2^n$($n \in$ 自然数)。

[规则 2]:数据表的首地址最低 k 位为零,k 满足关系式 $2^k > L$($k \in$ 自然数)。

如果地址长度为 24 位,规则 2 的实例如表 1 所示。

表 1　规则 2 的实例

L	K	位翻转类寻址起始地址
32	6	$xxxxxxxxxxxxxxxxxx000000_2$
1024	11	$xxxxxxxxxxxxx00000000000_2$

利用规则 1 和规则 2,可以方便地构造出函数 H,如式(8)所示。

$$H((reg)) = (reg)\text{“+”}L/2 \tag{8}$$

其中,(reg)指向数据表某个地址;“+”操作是逆向进位加法,即加法进位信号由高位向低位传递。例如:

$L = 2^4$, $L/2 = 2^3$,

$(reg) = (01101000)_2$,

那么位翻转寻址下个地址为:

$$\begin{aligned} Address &= (reg)\text{“+”}L/2 \\ &= (01101000)_2\text{“+”}(1000)_2 \\ &= (01100100)_2 \end{aligned}$$

如果从数据表首地址$(01100000)_2$开始,连续 15 次位翻转类寻址和递增基址加偏移量类寻址,地址最后 4 位的变化情况如表 2 所示。

表 2　位翻转类和递增基址加偏移量类寻址比较

递增访问数据序号	递增基址加偏移量类后 4 位地址	位翻转类后 4 位地址	位翻转类访问数据序号
0	0000	0000	0
1	0001	1000	8
2	0010	0100	4
3	0011	1100	12
4	0100	0010	2
5	0101	1010	10
6	0110	0110	6
7	0111	1110	14
8	1000	0001	1
9	1001	1001	9
10	1010	0101	5
11	1011	1101	13
12	1100	0011	3
13	1101	1011	11
14	1110	0111	7
15	1111	1111	15

本实例构造的函数 H 实现的位翻转类寻址运算开销与基址加偏移量类寻址运算开销相同,主要与物理实现时的加法器结构有关。因此可以通过选择不同结构的加法器优化位翻转类寻址运算速度、面积、功耗。

3.2　循环类寻址方式的实例

物理实现循环类寻址方式时,地址产生器生成算法中的循环类寻地址产生式需要进行实例化。为了优化物理实现结果,对循环类寻址方式作如下规定:

[规则 3]:循环类循环缓冲区长度记为 L(L∈自然数)。

[规则 4]:循环缓冲区首地址形式满足规则 2。

[规则 5]:循环缓冲区指针的改变步长满足$|Step| \leqslant L$。

根据规则 3,4,5,实例化的循环类地址产生过程用 C 语言描述如下:

```
{IF (Start≤(reg) + step<End)
  Address=(reg)+step;
  IF ((reg) + step≥End)
  Address=(reg)+step-L;
  IF ((reg)+step< Start)
  Address=(reg) + step+L;
}
```

循环缓冲区起始地址(Start)和结束地址(End),可由式(9)与(10)产生。

$$Start = (reg)\ \&\ Lt \tag{9}$$

$$End = Start \mid L \tag{10}$$

其中,“&”表示按位与操作;“|”表示按位或操作。Lt(L template)称为起始地址模板,根据规则 4 所规定的循环缓冲区首地址特点,L 与 Lt 以二进制表示且位长相同。模板的形成算法描述如下:从 L_{MSB}开始找到第一个 L_i 位不为零时,将 Lt_i 至 Lt_{LSB}位的值置为 0,将 Lt_{MSB} 至 Lt_{i+1} 位的值置为 1。

例如 $L = 32 = (000020)_h = (0000,0000,0000,0000,0010,0000)_2$

$(reg) = (87FE06)_h = (1000,0111,1111,1110,0000,0110)_2$

那么 $Lt = (FF\ FF\ C0)_h = (1111,1111,1111,1111,1100,0000)_2$

$Start = (reg)\ \&\ Lt = (87FE00)_h = (1000,0111,1111,1110,0000,0000)_2$

$End = Start \mid L = (87FE20)_h = (1000,0111,1111,1110,0010,0000)_2$

从实例化的循环类地址产生过程用 C 语言描述中可以发现,由于需要进行判断才能确定最后的地址表达式,因此,循环类寻址延迟时间在三类寻址延迟时间中最长,优化循环类地址产生路径延迟成为优化地址产生器整体性能的重要途径。

3.3　地址产生器生成算法实例的时间复杂度

根据地址产生器生成算法实例,图 4 描述了地址产生器的逻辑结构。

从图 4 中可以看出,循环寻址类计算路径是地址产生器的关键路径(图 4 中虚线所示)。关键路径上的时间延

迟 t_p 如式(11)。

$$t_p = 2t_{mux} + \max(t_{adder}, t_{Lt} + t_{\&} + t_{|}) + t_{cmp} + t_{reform} \quad (11)$$

其中，t_{mux}表示选择器的延迟；t_{adder}表示表示加法器的延迟；t_{Lt}表示 Lt 产生逻辑延迟；$t_{\&}$ 表示位与操作延迟；$t_{|}$ 表示位或操作延迟；t_{cmp}表示比较逻辑延迟；t_{reform}表示修改逻辑延迟。t_{mux}，$t_{\&}$，$t_{|}$基本为一常数，数量级与门延迟(t_g)相同。对于地址位数为 24 位的地址产生器：$t_{adder} >> t_g$；$t_{adder} > t_{Lt} + t_{\&} + t_{|}$；$t_{cmp}$、$t_{reform}$数量级与 t_{adder}相同。因此，式(11)可近似表达为式(12)。

$$t_p \approx 3t_{adder} \quad (12)$$

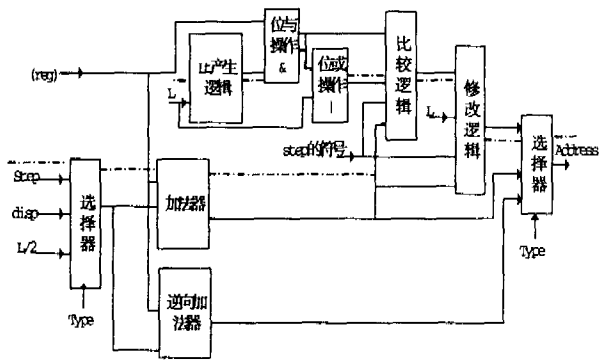


图 4 地址产生器的逻辑结构

表 3 给出了地址产生器使用 3 种不同加法器结构的时间复杂度。

表 3 不同加法器组成的地址产生器时间复杂度

加法器结构	加法器时间复杂度	地址产生器时间复杂度
Ripple	$O(n)$	$O(n)$
COSA	$O(\log_2 n)$	$O(\log_2 n)$
CLA	$O(\log n)$	$O(\log n)$

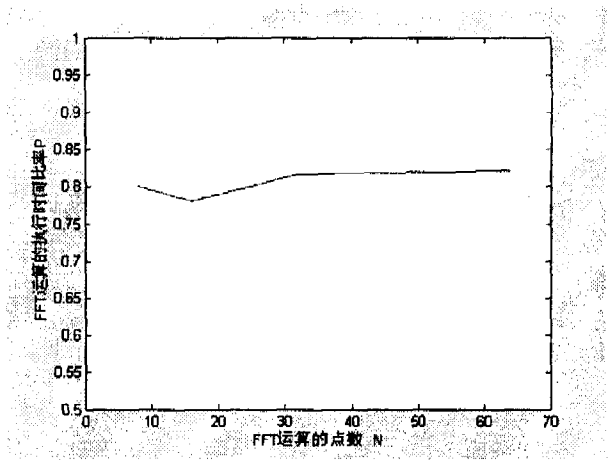
4 实验结果与分析

LS-RISC 处理器具有 4 级流水线(取址、译码、取操作数、执行)，其指令都为单周期指令(周期为 T)。LS-RISC 未采用地址产生器结构时，对其执行 FFT 运算和卷积运算的执行时间归一化，采用文中设计的地址产生器执行 FFT 运算和卷积运算的相对执时间曲线如图 5 所示。

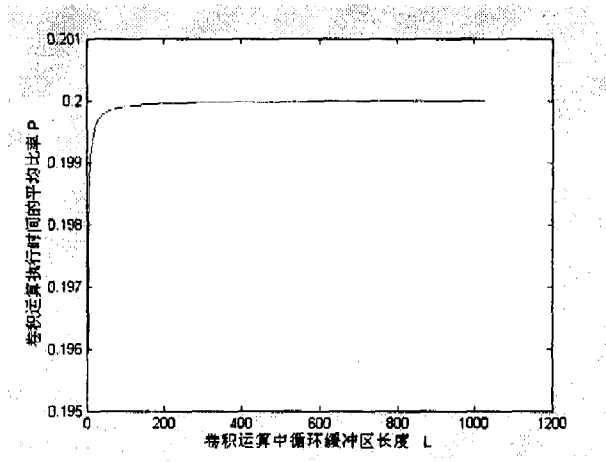
图 5 显示的数据表明，LS-RISC 采用文中设计的地址产生器时，FFT 运算、卷积运算平均执行速度比未采用地址产生器时的执行速度分别提高了 19%、80%。

5 结 论

文中给出的地址产生器生成算法有较强的普适性，生成算法实例实现方便、实用性强，可作为 IP 核应用到 RISC 处理器的设计中。地址产生器的时间复杂度可根据使用不同加法器结构进行调整，以满足不同 RISC 系统设计的需要。



(a) FFT 运算的执行时间



(b) 卷积运算平均执行时间

图 5 采用文中提出的地址产生器的相对执行时间曲线

参考文献：

[1] 沈绪榜. MPP 嵌入式计算机设计[M]. 北京：清华大学出版社，1999. 96-100.

[2] 沈绪榜. RISC 及后编译技术[M]. 北京：清华大学出版社，1994. 32-40.

[3] 蒋安平. 专用 32 位浮点 RISC 的数据路径研究[D]. 西安：西安微电子技术研究所，1997.

[4] 李三立，李亚民. RISC 单发射多发射体系结构[M]. 北京：清华大学出版社，1994. 132-140.

[5] Ma Y, Wanhammar L. A hardware efficient control of memory addressing for high-performance FFT processors[J]. IEEE Trans Signal Processing, 2000, 48: 917-921.

[6] Ma Y. An effective memory addressing scheme for FFT processors[J]. IEEE Trans Signal Processing, 1999, 47: 907-911.

[7] 马余泰. FFT 处理器地址快速生成方法[J]. 计算机学报，1994, 17(7): 505-512.