中国科学: 信息科学 2021 年 第 51 卷 第 5 期: 750–763
SCIENTIA SINICA Informationis
论文

三值光学计算机中并行 MSD 整数除法器的设计与实现

江家宝[1,2], 沈云付[1*], 陈迅雷[1], 王哲河[1], 刘拥[2], 陈丽萍[2]

1. 上海大学计算机工程与科学学院, 上海 200444
2. 巢湖学院信息工程学院, 巢湖 238000
* 通信作者. E-mail: yfshen@shu.edu.cn

收稿日期: 2019–10–31; 接受日期: 2019–12–05; 网络出版日期: 2021–03–30

安徽省教育厅自然科学基金重点项目 (批准号: KJ2017A452)、上海市科研计划专项 (批准号: 15700500400) 和国家自然科学基金面上项目 (批准号: 61073049) 资助

摘要 除法运算是基本四则运算之一, 如何进行快速除法一直是电子计算机、嵌入式系统和其他新型计算系统广受关注的问题. 充分发挥三值光学处理器位数众多、运算功能可重构、按位可分配等优势, 设计出高效并行 MSD (modified signed digit) 数除法器对提高大数据除法的运算效率、促进三值光学计算机 (ternary optical computer, TOC) 在数值计算领域的应用意义重大. 本文首次提出 MSD 数的符号判定算法, 并基于 SRT 算法首次提出利用一个并行无进位 SJ-MSD 加法器和一个 MSD 数比较器实现单组 MSD 整数除法或多组 MSD 整数并行除法方案 —— 并行 MSD 整数除法, 该算法对于被除数等长的多组与单组 MSD 整数除法需要的机器周期是相同的. 实验表明, 并行 MSD 整数除法方案是可行的, 它将有效地提高大数据处理效率并加速 TOC 进入数值计算等实际应用领域.

关键词 并行 MSD 整数除法器, SJ-MSD 加法器, 比较器, SRT 除法, 三值光学计算机

1 引言

在实现常用算术运算时, 把被减数和减数的相反数送入加法器进行运算便可实现减法运算, 利用软件方法, 通过一系列加 (减) 操作便可实现乘 (除) 运算. 因此, 在电子计算机中仅构造硬件加法器即可实现加减乘除运算. 随着硬件技术的发展, 为了提高效率, 硬件乘法器应运而生. 然而, 构建硬件除法器依然困难重重, 最大的问题是随着运算位数的增加, 除法器的硬件结构十分繁杂, 体积急剧膨胀, 电路复杂度急剧上升, 构造成本、构造难度和计算延时也随之快速上升. 因此, 寻求高效算法和新的快速硬件设备提高除法的效率备受计算机等相关领域学者的关注. 为了提高效率, 人们长期致力于除法实现方法的优化, 多种除法实现方法相继出现. 其中, 恢复余数和不恢复余数两种函数迭代法最为突出 [1~3]. 函数迭代法执行的时间复杂度较高. 恢复余数和不恢复余数是以加法和减法运算为基础的

引用格式: 江家宝, 沈云付, 陈迅雷, 等. 三值光学计算机中并行 MSD 整数除法器的设计与实现. 中国科学: 信息科学, 2021, 51: 750–763, doi: 10.1360/SSI-2019-0240
Jiang J B, Shen Y F, Chen X L, et al. Design and implementation of parallel MSD integer divider in ternary optical computer (in Chinese). Sci Sin Inform, 2021, 51: 750–763, doi: 10.1360/SSI-2019-0240

算法, 需要多个循环周期, 虽然实现方法简单, 但需要较多的循环次数才能达到需要的目标, 运算效率较低. 由 D. Sweeney, J.E. Robertson 和 T.D. Tocher 3 人独立提出的 SRT 除法算法是不恢复余数算法的扩展, 在电子计算机和嵌入式系统等领域得到了广泛应用, 最受关注. SRT 的商数取值集合与 SD (signed-digit) 冗余计数法的取值集合完全相同, SRT 通过移位操作实现被除数和除数的规格化. 因此, 与其他二进制除法方法相比, SRT 算法更易于实现. 由于 SRT 算法商值取值出现负值, 所以在迭代过程中或运算结束后, 需要将包含负值的商值转换为常规二进制数 [2~8]. SRT 算法实现除法过程中的多次迭代意味着很难完全依靠硬件实现长数据除法, 因此, 软硬件结合方法是除法实现的常用方案 [6]. TOC 在 MSD (modified signed digit) 数 (含标准二进制数) 的计算方面具有很大优势.

处理器众位数、可按位重构、可按位独立使用、处理器位易扩展以及并行性等特点使其更适用于快速处理大批量复杂运算的数据. 2017 年 3 月上海大学光学计算机中心研制出三值光学处理器 (ternary optical processor, TOP) SD16 的一个模块, 该模块有 192 个处理器位, 其任一处理器位都可独立使用, 并可按功能需求重构成一位三值 (二值) 逻辑运算器. 目前 SD16 可以加装 64 个模块, 可扩展成万位光学处理器. 随后我们在 SD16 的一个模块上重构出了 46 位 SJ-MSD 并行加法器 [9], 这标志着 MSD 并行加法器进入实际应用的阶段. 与此同时, 团队已着手进行乘法、除法等计算例程的设计与开发, 并设计了三值光学计算机应用开发平台, 它为三值光学计算机在符号处理、图像处理、神经网络和人工智能等数值计算领域发挥重要作用奠定了基础.

目前三值光学处理器没有专门构建硬件乘除法器, 乘除运算仍采用软件方式将其转化成一系列加 (减) 运算后利用加法器来实现, 但用三值光学处理器进行计算效率更高. MSD 数的符号集 {ī, 0, 1} 与 SRT 算法的商值符号集完全相同, 且在 TOP 中很容易用硬件实现 SRT 算法的迭代操作.

本文在对并行无进位 MSD 加法进行研究的基础上, 对基于 SRT 算法的 MSD 除法进行了深入研究. 第 2 节介绍了 MSD 数表示、MSD 数加法及 SRT 除法的相关研究. 第 3 节介绍了 MSD 数比较器原理和设计思想. 第 4 节介绍了基于 SRT 算法的 MSD 单数据整数除法与多数据并行整数除法技术, 该技术统称为 “并行 MSD 整数除法”. 第 5 节介绍了在 SD16 上构建由一个加法器和一个比较器组成的并行 MSD 整数除法器. 第 6 节介绍了并行 MSD 整数除法器同时计算多组数据的相关实验, 实现了多组除法数据的并行除法, 并进行了实验结果分析. 实验结果表明, 并行 MSD 整数除法有效地提高了大整数除法和批量数据除法的计算效率.

2 MSD 数加法与 SRT 除法的相关研究

在 TOC 上, 并行无进位 SJ-MSD 加法器的完成 [9,10] 为其乘除法器的实现提供了硬件支持. 本文中的 MSD 除法是以 TOC 中的三值逻辑变换 —— C 变换和并行无进位 SJ-MSD 加法为基础的.

2.1 MSD 数的表示

Avizienis 于 1961 年首次提出 MSD 数字系统 [11]. 任意数值的十进制数表达和 MSD 数表达的转换关系如下 [9,10]:

$$[x,y]_{10}=\sum_{i=0}^{w}(a_i\times 2^i)+\sum_{j=1}^{v}(b_j\times 2^{-j}), \tag{1}$$

其中 a_i, $b_j\in\{-1,1,0\}$, 且 $i=0,1,2,\ldots,w$; $j=1,2,\ldots,v$. 为了方便与 “减 1” 区分, 本文将 “−1” 改写为 “ī” 或 “u”.

表 1　SJ 变换 —— S_1, S_2, J_1, J_2 和 J_3 三值逻辑变换

Table 1　SJ transformation — S_1, S_2, J_1, J_2 and J_3 ternary logic transformation

Transformation S_1				Transformation S_2				Transformation J_1			Transformation J_2			Transformation J_3		
b	a			b	a			s_1	s_2		s_1	s_2		j_2	j_1	
	0	1	u		0	1	u		0	1		0	1		0	1
0	0	0	u	0	0	1	1	0	0	1	0	0	u	0	0	1
1	0	1	0	1	1	0	0	1	0	1	1	1	0	1	1	0
u	u	0	u	u	1	0	0	u	0	0	u	u	0	u	u	0

在 TOC 中, 我们用无光态、相互正交的垂直偏振光态与水平偏振光态依次表达 0, 1 和 u, 利用液晶对光状态的旋转和偏振片对光状态的选择透过实现 3 种光状态之间的转变, 从而实现各种运算. MSD 数中使用了冗余符号, 与常规二进制数表示相比, 它有如下特点.

(1) 除了数值 0 之外, 任意数值都有无数个 MSD 数表达, 如 $[-24]_{10}$ =$[\mathrm{u1u000}]_{\mathrm{MSD}}$ =$[\mathrm{u11u000}]_{\mathrm{MSD}}$ = $\cdots$; $[15.5]_{10} = [1111.1]_{\mathrm{MSD}} = [\mathrm{1u00u1.1}]_{\mathrm{MSD}} = \cdots$.

(2) 一个数值的 MSD 数逐位取反 (0 不变, 1 和 u 互相转换) 得到其相反数的 MSD 数表示. 如: $[-22.25]_{10} = [\mathrm{uu010.0u}]_{\mathrm{MSD}} = [\mathrm{u1u01u.11}]_{\mathrm{MSD}}$; $[22.25]_{10} = [\mathrm{110u0.01}]_{\mathrm{MSD}} = [\mathrm{1u10u1.uu}]_{\mathrm{MSD}}$.

(3) 常规二进制数是 MSD 数的不使用 u 符号的子集.

2.2　SJ-MSD 加法原理

SJ-MSD 加法由如表 1 所示的 S_1, S_2, J_1, J_2, J_3 三值逻辑变换 (SJ 变换) 和一组操作规则 (SJ 规则) 组成 [9,10].

约定: 本文的三值逻辑真值表以列优先方式取得变换值.

SJ 规则: 对两个 n 位 MSD 数 a 和 b 按照如下 3 步操作即可完成 $a+b$ 运算 [9,10].

步骤 1. 对 a 和 b 逐位同时进行 S_1 和 S_2 变换; S_1 变换结果左移 1 位, 第 0 位补 0 (右补 0), 得到 $n+1$ 位的 s_1 值; S_2 变换结果第 n 位补 0 (左补 0), 得到 $n+1$ 位的 s_2 值.

步骤 2. 对 s_2 和 s_1 的低 n 位逐位进行 J_1 变换; J_1 变换结果左移 1 位, 第 0 位补 0 (右补 0), 得到 $n+1$ 位的 j_1 值; 同时对 s_2 和 s_1 逐位进行 J_2 变换, 得到 $n+1$ 位的 j_2 值.

步骤 3. 对 j_1 和 j_2 逐位进行 J_3 变换, 得到 $n+1$ 位的最终结果: $j_3 = a + b$.

在 SD16 中构建 SJ-MSD 加法器时, 为了节约 SD16 处理器位资源从而提高处理器效率, 用 1 个处理器位同时实现一位的 J_1 变换和 J_2 变换. 这样, 完成 n 位 MSD 数 a 和 b 的加法只需要 $(4n+2)$ 个处理器位即可. 详细操作参见文献 [9].

2.3　不恢复余数迭代除法与 SRT 除法

在电子计算机中, 与恢复余数除法类似, 不恢复余数除法主要思想是: 在进行每一轮的迭代运算时, 首先做减法运算, 如果差值为非负, 商为 1; 如果差值为负数, 商为 0, 当前余数左移 1 位, 加上除数. SRT 除法仍采用减法操作进行试商. SRT-2 除法 (基数 $r=2$) 基本原理如下 [7,8].

假设 A 表示被除数, B 表示除数, 求商 $q = A/B$, 余数 $R = A - q \times B$; r 表示 SRT 算法的基数, 这里 $r=2$. 用 q_j 表示第 j 次试商得到的第 j 位 (自左向右) 商数, w_j 表示第 j 次试商后得到的部分余数. 商数 q_j 的选择是整个 SRT 算法实现过程的关键.

(1) 商数的各位值采用冗余计数法:

表 2 比较器真值表 C

Table 2 Comparator truth table C

b	a		
	0	1	u
0	0	1	u
1	1	1	u
u	u	1	u

对于 r 进制, SD 数的集合为 $\{-(r-1),-(r-2),\ldots,-1,0,1,\ldots,(r-2),(r-1)\}$.

当 $r=2$ 时, 该集合为 $\{-1,0,1\}$.

(2) 对于二进制数, 将除数和被除数规格化到 [0.5, 1) 域.

(3) 对于第 j 次迭代的商值 q_j 和部分余数 w_j 的取值分别由式 (2) 和 (3) 确定:

$$q_j=\begin{cases}-1, & \text{if } (w_{j-1}\leqslant -0.5);\\ 0, & \text{if } (-0.5<w_{j-1}<0.5);\\ 1, & \text{if } (0.5\leqslant w_{j-1});\end{cases}\tag{2}$$

$$w_j=\begin{cases}2(w_{j-1}+\mathrm{B}), & \text{if } (w_{j-1}\leqslant -0.5);\\ 2w_{j-1}, & \text{if } (-0.5<w_{j-1}<0.5);\\ 2(w_{j-1}-\mathrm{B}), & \text{if } (0.5\leqslant w_{j-1}).\end{cases}\tag{3}$$

因为商和余数可能出现负值, 所以在迭代运算过程中或运算完成时还需要对商值进行转换运算以得到正确标准的二进制商和余数.

3 MSD 数比较器设计

在利用 SRT 算法实现除法过程中, 涉及两个 MSD 数相加或相减, 根据和 (差) 值的正负选择不同的操作. 所以, 需要设计一个高效判断 MSD 数符号的算法 (简称为 “二叉比较法”) 以及实现该算法的判断部件 (简称为 “比较器”).

3.1 二叉比较法

假设定长 MSD 数 D 有 m 位, 记 $D=d_{m-1}d_{m-2}d_{m-3}d_{m-4}\cdots d_2d_1d_0$, 每位取值为 {0, 1, u}.

一个 MSD 数 D 的符号依据如表 2 所示的真值表给出, 可通过如下递归算法确定.

步骤 1. 若 $m=1$, 转步骤 5; 否则转步骤 2.

步骤 2. 从 D 的最高位开始, 依次对每相邻的两位 d_{j+1}, d_j 进行比较, 比较结果依据表 2 给出, 即 $C[d_{j+1}][d_j]$ 为比较结果.

步骤 3. 如 m 为奇数, 将最低位数 d_0 作为比较结果直接放入上述比较结果数据的尾部.

步骤 4. 将步骤 2 和 3 得到的数仍用数组 D 记录, 长度仍记为 m; 转步骤 1.

步骤 5. 最后获得的一位数据 d 就是 D 的符号位, 结束.

此时, 若 $d=0$, 则 D 为 0, 若 $d=\mathrm{u}$, 则 D 为负数, 若 $d=1$, 则 D 为正.

上述算法经过 $\lceil \log_2 m \rceil$ 次递归得到解决. 步骤 2 中每相邻两位的比较可并行操作, 因此二叉比较法的时间复杂度为 $O(\log_2 m)$.

3.2 三值光学比较器设计

对于长为 m 的 MSD 数 D, 在判定 D 的符号时, 为了提高并行性, 在 TOP 中根据表 2 中的 C 变换表重构一个 $\lceil m/2 \rceil$ 位比较器. 每轮比较时, 对 MSD 数 D, 每相邻两位 d_{2j+1} 和 d_{2j} $(i = 0, 1, \ldots, \lceil m/2 \rceil - 1)$ 进行比较; 其中奇数位置的数依次作为一位比较器的主光路输入 a, 而偶数位置的数作为控制光路的输入 b. 在每轮比较结束时, 将得到的比较结果仍记为 D, 作为新一轮比较的输入. 这样, m 位 MSD 数 D 的 $\lceil m/2 \rceil$ 次比较操作可经过一次并行计算完成. 由此可见, 在 SD16 中, 对于任意 m 位 MSD 数 D, 只需要 $\lceil \log_2 m \rceil$ 个机器周期即可判断出其符号特性. 在以后的 TOP 中可构建一个 2^T 位 (T 为正整数) 的常备比较器, 采用 T 层叠加结构构建. 这样, 判断任意位数不大于 2^T 位 MSD 数的符号特性只需要 1 个机器周期.

4 并行 MSD 整数除法算法

基于 SRT 除法原理, 设计如下单数据 MSD 整数除法算法和多数据并行 MSD 整数除法算法.

4.1 单数据 MSD 整数除法算法

对于 n 位 MSD 被除数 $A = a_0 a_1 a_2 \cdots a_{n-1}$ 和 m 位 MSD 除数 $B = b_0 b_1 b_2 \cdots b_{m-1}$, 求商 $q = A/B$, 余数 $R = A - q \times B$. 以下仍采用 2.3 小节的有关符号, 但我们将上述 SRT-2 除法的算法过程修改如下.

首先设 B 是正数, $w_0 = a_0$. 对 $j = 0, 1, 2, \ldots, n-1$, 有

$$q_{j+1} = \begin{cases} -1, & \text{if } (w_j \leqslant -B); \\ 0, & \text{if } (-B < w_j < B); \\ 1, & \text{if } (B \leqslant w_j). \end{cases} \tag{4}$$

当 $j<n-1$ 时,

$$w_{j+1} = \begin{cases} (w_j + B)\hat{}a_{j+1}, & \text{if } (w_j \leqslant -B); \\ w_j\hat{}a_{j+1}, & \text{if } (-B < w_j < B); \\ (w_j - B)\hat{}a_{j+1}, & \text{if } (B \leqslant w_j). \end{cases} \tag{5}$$

当 $j=n-1$ 时,

$$w_n = \begin{cases} w_{n-1} + B, & \text{if } (w_{n-1} \leqslant -B); \\ w_{n-1}, & \text{if } (-B < w_{n-1} < B); \\ w_{n-1} - B, & \text{if } (B \leqslant w_{n-1}). \end{cases} \tag{6}$$

上述式 (5) 中, 记号 $x\hat{}y$ 表示在数据 x 后面拼接一个数据 y. 那么, $q = q_0 q_1 q_2 \cdots q_{n-1}$ 是 A 除以 B 的商, 而 w_n 是 A 除以 B 所得的余数.

其次, 如果 B 是负数, 那么对上述式 (4)~(6) 确定商 q_j 及 w_{j+1} 的条件进行适当修改即可.

单数据 SRT 整数除法执行流程如图 1 所示.

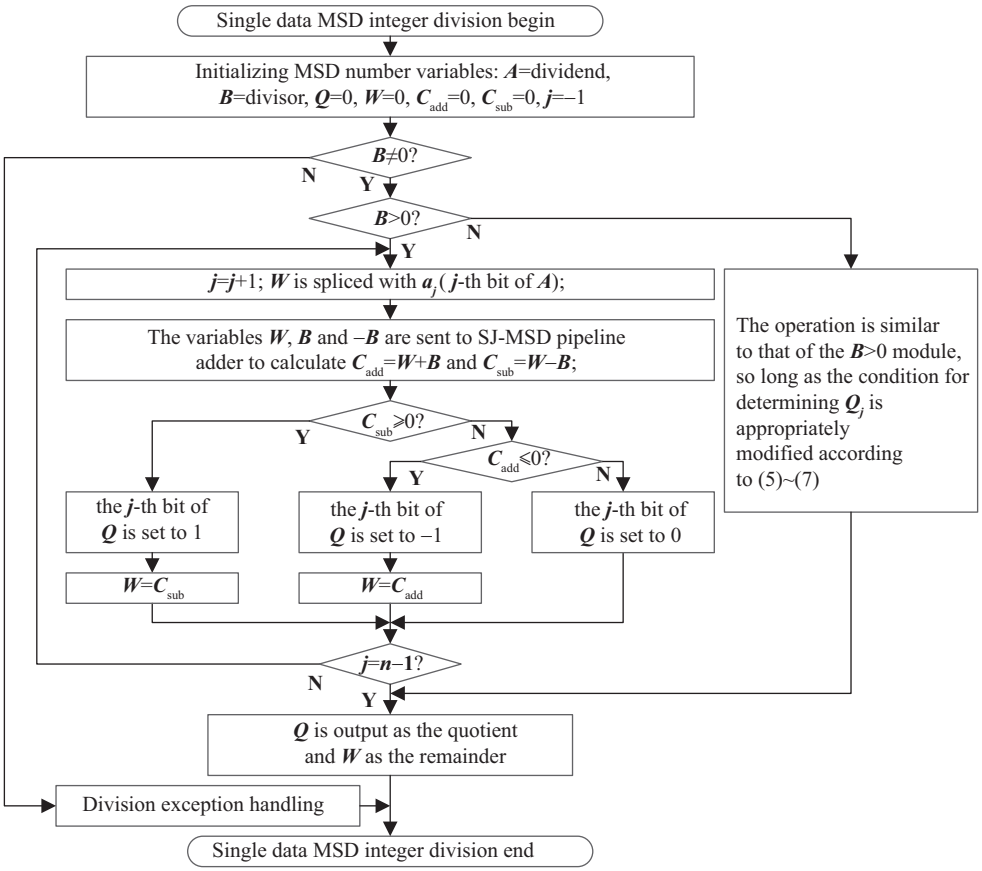

图 1 单数据 MSD 整数除法的执行流程

Figure 1 Execution flow chart of single data MSD integer division

4.2 多数据并行 MSD 整数除法

以下 $i=0,1,2,\ldots,k-1$, n 位 MSD 数的最高位为第 0 位, 最低位为第 $n-1$ 位. 对于 k 个 MSD 被除数 A_i (数据位数最大为 n) 和 k 个 MSD 除数 B_i (数据位数最大为 m), 即 k 组除法数据, 为了求商 $Q_i=A_i/B_i$ 和余数 $R_i=A_i-Q_i\times B_i$, 定义与初始化如下 MSD 数组与循环控制变量.

(1) MSD 数组 A, 数组元素 $A[i]$ 初始化为第 i 个被除数 A_i (位数不足 n 则高位补 0).

(2) MSD 数组 B, 数组元素 $B[i]$ 初始化为第 i 个除数 B_i (位数不足 m 则高位补 0).

(3) MSD 数组 N, 数组元素 $N[i]$ 初始化为第 i 个除数 B_i 的相反数.

(4) MSD 数组 Q, 数组元素 $Q[i]$ 用于存储第 i 个除法的商, 初始化为 0.

(5) MSD 数组 W, 数组元素 $W[i]$ 用于存储第 i 个除法的部分余数, 初始化为 0.

(6) MSD 数组 $C_{\rm add}$ 和 $C_{\rm sub}$, 数组元素 $C_{\rm add}[i]$ 和 $C_{\rm sub}[i]$ 用于存储 SJ-MSD 加法器输出.

(7) 循环控制变量 j, 初始化为 -1.

在 TOP 中重构一个 SJ-MSD 流水加法器和一个比较器, 每次计算 $C_{\rm add}[i]=W[i]+B[i]$ 和 $C_{\rm sub}[i]=W[i]+N[i]$ $(=W[i]-B[i])$ 时, 在 SJ-MSD 加法器中选取 $2k$ 个区域, 第 i 和第 $i+k$ 个区域分别用于流水计算 $C_{\rm add}[i]$ 和 $C_{\rm sub}[i]$, 在比较器中选取 $2k$ 个区域, 第 i 和第 $i+k$ 个区域分别用于判断 $C_{\rm add}[i]$ 和 $C_{\rm sub}[i]$ 的符号.

多数据并行 MSD 整数除法按照以下步骤执行.

步骤 1. 变量 j 增 1; 针对每个 i, $W[i]$ 与 $A[i]$ 的第 j 位拼接.

步骤 2. 针对每个 i, 把 $W[i]$ 和 $B[i]$ 作 SJ-MSD 加法器第 i 个区域的原始数据输入; 把 $W[i]$ 和 $N[i]$ 作 SJ-MSD 加法器第 $i+k$ 个区域的原始数据输入.

步骤 3. 启动 TOP 完成所有 k 个 SJ-MSD 加法运算 $W[i]+B[i]$ 及 k 个 SJ-MSD 加法运算 $W[i]+N[i]$; 并将 SJ-MSD 加法器第 i 个区域的输出, 即 $W[i]+B[i]$ 的和值, 存入 $C_{\mathrm{add}}[i]$ 中, SJ-MSD 加法器第 i+k 个区域的输出, 即 $W[i]+N[i]$ 的和值, 存入 $C_{\mathrm{sub}}[i]$ 中.

步骤 4. 针对每个 i, $C_{\mathrm{add}}[i]$ 和 $C_{\mathrm{sub}}[i]$ 分别送入比较器的第 i 和第 $i+k$ 个区域; 启动 TOP 完成符号判断操作; 判断结果为 $C_{\mathrm{add}}[i]$ 和 $C_{\mathrm{sub}}[i]$ 的最高位.

步骤 5. 针对每个 i, 作如下操作:

如果 $B[i]>0$, 那么,

如果 $C_{\mathrm{sub}}[i]\geqslant 0$, $Q[i]$ 的第 j 位置 1, $C_{\mathrm{sub}}[i]$ 替代 $W[i]$.

否则, 如果 $C_{\mathrm{add}}[i]\leqslant 0$, $Q[i]$ 的第 j 位置 u, $C_{\mathrm{add}}[i]$ 替代 $W[i]$.

否则, $Q[i]$ 的第 t 位置 0.

否则, 如果 $C_{\mathrm{sub}}[i]\leqslant 0$, $Q[i]$ 的第 j 位置 1, $C_{\mathrm{sub}}[i]$ 替代 $W[i]$.

否则, 如果 $C_{\mathrm{add}}[i]\geqslant 0$, $Q[i]$ 的第 j 位置 u, $C_{\mathrm{add}}[i]$ 替代 $W[i]$.

否则, $Q[i]$ 的第 t 位置 0.

步骤 6. 数组 Q 的 k 个元素分别作为 k 个除法数据的商输出, 数组 W 的 k 个元素分别作 k 个除法数据的余数输出; 除法结束.

说明: SJ-MSD 加法器完成一次加法运算需要 3 步操作, 需要 3 个机器周期, 步骤 3 并行完成计算 $W[i]+B[i]$ 和 $W[i]+N[i]$ 仅需要 3 个机器周期. 如果 TOP 资源受限, 为了提高 TOP 的效率, 可在加法器的第 i 个区域流水计算 $W[i]+B[i]$ 和 $W[i]+N[i]$, 这样, 需要 4 个机器周期完成 $W[i]+B[i]$ 和 $W[i]+N[i]$ 的并行计算.

5 并行 MSD 整数除法器的设计与实现

5.1 实验设备

主要实验设备是图 2 所示的 SD16 一号基本模块, 其像素排列如图 3 所示. SD16 的每个模块有排列成 24 行 24 列的 576 个像素. 图 3 中的每个小方格代表一个像素. 每行 3 个相邻的像素组成一个处理器位, 其编号从 0 到 191. 0 号处理器位在最低行的中部. 处理器左半侧的每个处理器位的 3 个像素依次是 w 像素、v 像素和 h 像素, 也称为 o 分支光路、v 分支光路和 h 分支光路, 右半侧相反. 每个像素都可以输出垂直偏振光态或水平偏振光态或无光态, 于是主光路的每个像素可以重构出 3 个最简运算基元, 故每个处理器位可以重构出 9 个最简运算基元. 根据降值设计理论 [12], 任意三值逻辑运算器的一位可以用不超过 6 个最简运算基元构成, 于是 SD16 每个处理器位都可以构成任意三值变换器的一位.

5.2 并行 MSD 整数除法器的处理器位分配

采用文献 [9] 的技术, 在一个处理器位上同时实现 J_1 变换和 J_2 变换, 则 4 个处理器位就可构成 SJ-MSD 加法器所需变换器各一位. 根据表 2 构建的比较器和 SJ-MSD 加法器的每一位可以任意选取处理器位来构建, 为了数据整理方便, 我们选取位号连续的处理器位构建比较器和 SJ-MSD 加法器.

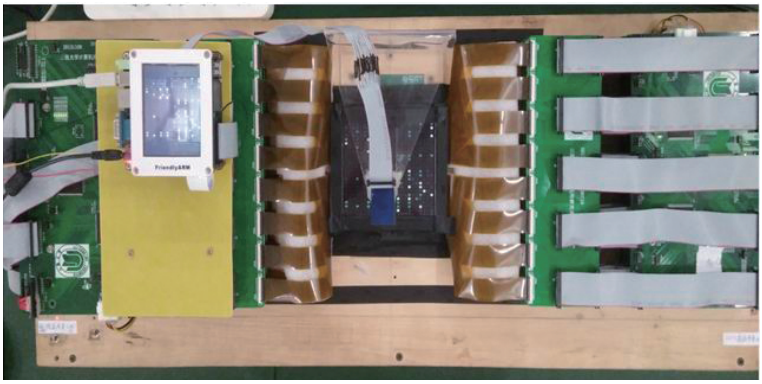

图 2 (网络版彩图) SD16 的 1 号机外形

Figure 2 (Color online) Appearance of machine 1 of SD16

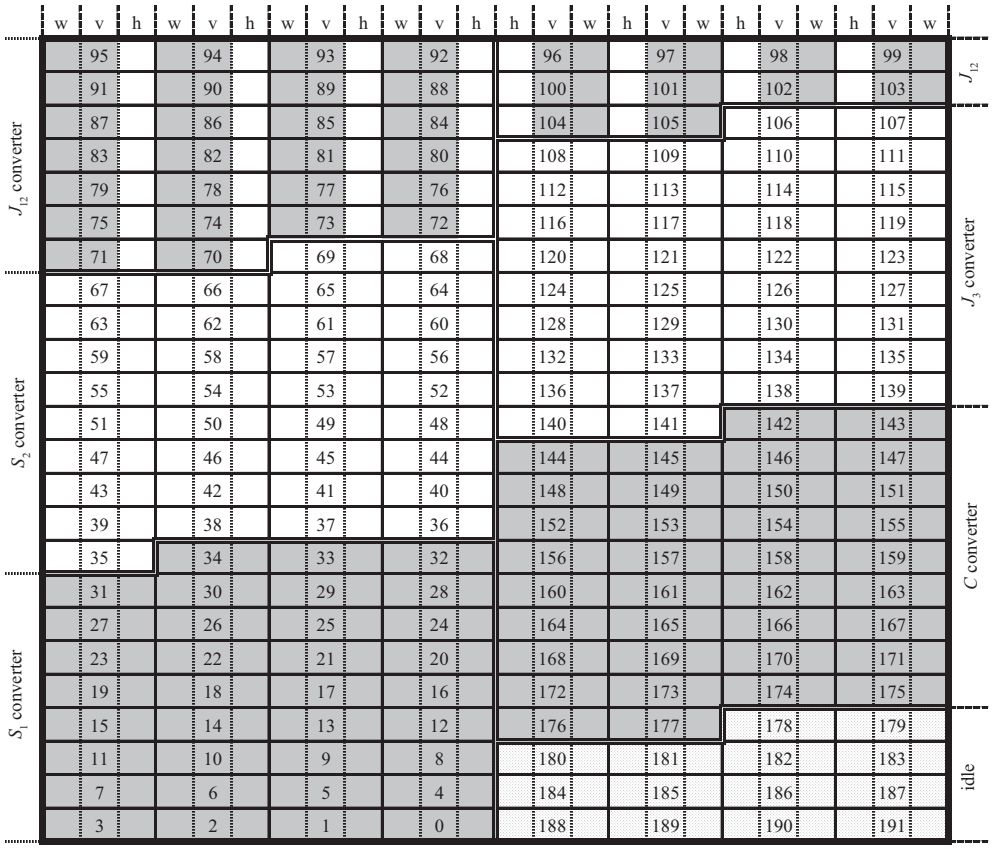

图 3 处理器像素排列与除法器处理器位分配

Figure 3 Processor pixel arrangement and divider processor bit allocation

于是从第 C_0 号处理器位开始构建 $k(m+3)-1$ 位 SJ-MSD 加法器和 $2k\lceil(m+3)/2\rceil$ 位比较器的设计如下.

(1) C_0~$C_{k(m+4)-2}$ 号处理器位构造成 $k(m+4)-1$ 位 S_1 变换器, w, v 和 h 像素的输出迭合成 S_1 输出.

(2) $\mathrm{C}_{k(m+4)-1}$~$\mathrm{C}_{2k(m+4)-3}$ 号处理器位构造成 $k(m+4)-1$ 位 S_2 变换器, w, v 和 h 像素的输出迭合成 S_2 输出.

(3) $\mathrm{C}_{2k(m+4)-2}$~$\mathrm{C}_{3k(m+4)-3}$ 号处理器位构造成 $k(m+4)$ 位 J_{12} 变换器, 即这些处理器位的 w 像素和 v 像素构造成 $k(m+4)$ 位 J_2 变换器, h 像素构造成 $k(m+4)$ 位 J_1 变换器. 相应地, 给 v 像素和 h 像素输入相同的数据 s_2; w 像素和 v 像素的输出迭合成 J_2 输出, 而 h 像素的输出为 J_1 输出.

(4) $\mathrm{C}_{3k(m+4)-2}$~$\mathrm{C}_{4k(m+4)-3}$ 号处理器位构造成 $k(m+4)$ 位 J_3 变换器, w, v 和 h 像素的输出迭合成 J_3 输出.

(5) $\mathrm{C}_{4k(m+4)-2}$~$\mathrm{C}_{2k\lceil(m+4)/2\rceil+4k(m+4)-3}$ 号处理器位构造成 $2k\lceil(m+4)/2\rceil$ 位比较器, w, v 和 h 像素的输出迭合成比较器输出.

显然, 上述一个并行 MSD 整数除法器需要的处理器位总数为 $k(4m+16+2\lceil(m+4)/2\rceil)-2$.

6 实验与分析

为验证并行 MSD 整数除法的正确性, 2019 年 8 月 22 日作者在 SD16 处理器 1 号模块上实现了 1 个并行 MSD 整数除法器, 随后对其进行了全面测试. 测试实验分 3 个阶段: 首先是实验准备阶段, 包括处理器位分配、实验规划、实验用例选择、实验用例理论图像准备和调试实验系统等工作. 然后是实验实施阶段, 包括重构除法器, 逐一把实验用例原始数据送入除法器, 逐一拍照处理器输出的计算结果图像, 将除法器输出结果与事先准备好的理论结果对比并记录对比情况等. 最后是实验分析阶段, 主要分析每一个实验用例的实验结果与理论结果的差异, 找出产生差异的原因, 排除错误, 完成补充实验, 得出最终实验结论等. 具体描述如下.

6.1 实验准备

(1) 并行 SRT 整数除法器的处理器位分配. SD16 处理器 1 号模块共有 192 个处理器位, 实验取被除数数据位数 $n=16$, 除数数据位数 $m=8$, 并行运算的除法数据个数 $k=3$. 由 5.2 小节可知, 实际需要的处理器位总数为 $(4m+16+2\lceil(m+4)/2\rceil)k-2=178<192$. 我们把 0~141 号处理器位重构成 35 位的 SJ-MSD 加法器, 其中 0~34 号处理器位为 S_1 变换器, 35~69 号处理器位为 S_2 变换器, 70~105 号处理器位为 J_{12} 变换器, 106~141 号处理器位为 J_3 变换器, 142~177 号处理器位为 36 位的比较器, 处理器位分配如图 3 所示. 由于 TOP 的处理器位数受限, 每次确定商 q_j $(j=0,1,2,\ldots,15)$ 时采取流水方式计算 $C_{\mathrm{add}}[i]=W[i]+B[i]$ 和 $C_{\mathrm{sub}}[i]=W[i]+N[i]$ $(i=0,1,2)$.

(2) 实验规划.

(i) 验证并行 MSD 整数除法器的计算正确性.

(ii) 验证一个并行 MSD 整数除法器能同时运算多组除法数据.

(3) 实验用例选择. 准备好的实验用例如表 3 所示, 其中, 用例 1~8 为随机数据, 位数不足的高位补 0 (用 ϕ 表达), 而用例 9~16 为边界值. 用例 17~19 为存在除数为 0 的用例.

(4) 实验用例理论图像准备. 根据并行 MSD 整数除法器的结构, 可以从理论上推测出除法器输出商和余数时光学处理器的输出状态, 实验前先将这些理论预测的光学处理器输出状态绘制成图像. 针对表 3 给出的实验用例, 共绘制处理器的理论输出图像 16 幅, 如图 4 所示, 图中 H 表示该像素输出水平偏振光, V 表示输出垂直偏振光, 无符号表示输出无光态.

(5) 按照第 4 节给出的操作步骤和算法准备好操控软件. 该软件将加法器的 3 步操作分离, 每一步操作都等待上位机发出步进指令后方进行, 这样做的目的是在光学处理器的每一个操作步之后留有

表 3 实验用例表

Table 3 Table of experiment cases

Case	Dividend 1	Divisor 1	Dividend 2	Divisor 2	Dividend 3	Divisor 3
1	11u0uuuu01001uu1	11u1u1u0	$\phi\phi\phi$1uuuu011u0u10	uuuuu101	uuu11111u10u1u0u	$\phi\phi$1101u1
2	ϕu110u1u1u101u01	u0u11010	u011u0u100u010u0	1uu01001	1u1000u010u11u0u	u0u1101u
3	u0100u0101u0u1u0	1u10u0u0	1u001u0u00u010u1	u1u1u1u1	u1u01u0101110u1u	u000u010
4	u1u0101u10u1uu01	01101001	$\phi\phi\phi\phi\phi\phi\phi\phi$u010u1u0	u1uuu011	111u0u011u0u11uu	10u0u1uu
5	1u10000uu0uu110u	u1u1u10u	u0100000000001uu1	ϕ11u0u10	$\phi\phi$1u0u0100001uu1	110u0u10
6	ϕ1uu01u100uu110u	u101u0u1	u0100001uu101uu1	10u111u1	101001u001101uu1	u00u100u
7	1u1u1010u110uu01	10u11010	1u0u100u0u0110u0	uu0u1001	1u1000u0100u0u01	1u1uuu1u
8	ϕ1u0u1u0100u01u0	1u0uu101	1u10u010u01u0u01	u1uu11u1	1u0101u1u0110u1u	u0u00000
9	0000000000000000	00000001	0000000000000001	00000001	000000000000000u	00000001
10	0000000000000000	0000000u	0000000000000001	0000000u	000000000000000u	0000000u
11	0000000000000000	11111111	0000000000000001	11111111	000000000000000u	11111111
12	0000000000000000	uuuuuuuu	0000000000000001	uuuuuuuu	000000000000000u	uuuuuuuu
13	0000000000000000	00000001	1111111111111111	00000001	uuuuuuuuuuuuuuuu	00000001
14	0000000000000000	0000000u	1111111111111111	0000000u	uuuuuuuuuuuuuuuu	0000000u
15	0000000000000000	11111111	1111111111111111	11111111	uuuuuuuuuuuuuuuu	11111111
16	0000000000000000	uuuuuuuu	1111111111111111	uuuuuuuu	uuuuuuuuuuuuuuuu	uuuuuuuu
17	1u10000uu0uu110u	00000000	u0100000000001uu1	u11u0u10	1u1u0u0100001uu1	110u1010
18	01uu01u100uu110u	u110u0u1	u0100001uu101uu1	00000000	101001u001101uu1	u00u110u
19	1u1u1010u110uu01	10u1u110	1u0u100u0u0110u0	uu0u1101	1u1000u0100u0u01	00000000

拍摄处理器输出图像的时间.

6.2 实验实施

实验的实施严格按下列步骤进行:

(1) 使用处理器重构指令, 在 SD16 处理器模块 1 上, 按图 3 所示构造出除法器.

(2) 将表 3 给出的实验用例 1 的输入数据送入光学处理器的数据输入端, 然后发出操控光学处理器运行的 “步进指令”.

(3) 当最后一次上商时, 拍摄光学处理器输出的图像 (如图 5).

(4) 将拍摄到的图像与实现准备好的理论图像对比. 若二者相同, 在实验记录表 4 中对应位置打 ✓ 号, 进行实验步骤 (5); 否则对照理论输入数据检查实际输入的数据是否有错误. 若输入有误, 则更正输入后重新拍照; 否则在实验记录表中对应位置打 ×, 并停止实验, 进入实验结果分析阶段.

(5) 对表 3 给出的后续实验例重复执行实验步骤 (2)∼(4), 直到 19 个实验用例运算结束. 作为示例, 图 4 和 5 分别给出了前 3 个用例运行结束时处理器理论输出图像和实际输出图像.

实验结果表明: 在计算实验用例的过程中, 并行 MSD 整数除法器的工作状态与理论预测一致.

6.3 实验结果分析

从光学处理器的输出图像或对应的理论图像都可以判读出并行 MSD 整数除法器所拥有的各个三值变换器输出数值. 为分析并行 MSD 整数除法器在 TOP 上的实施情况, 将实验过程中产生的用例

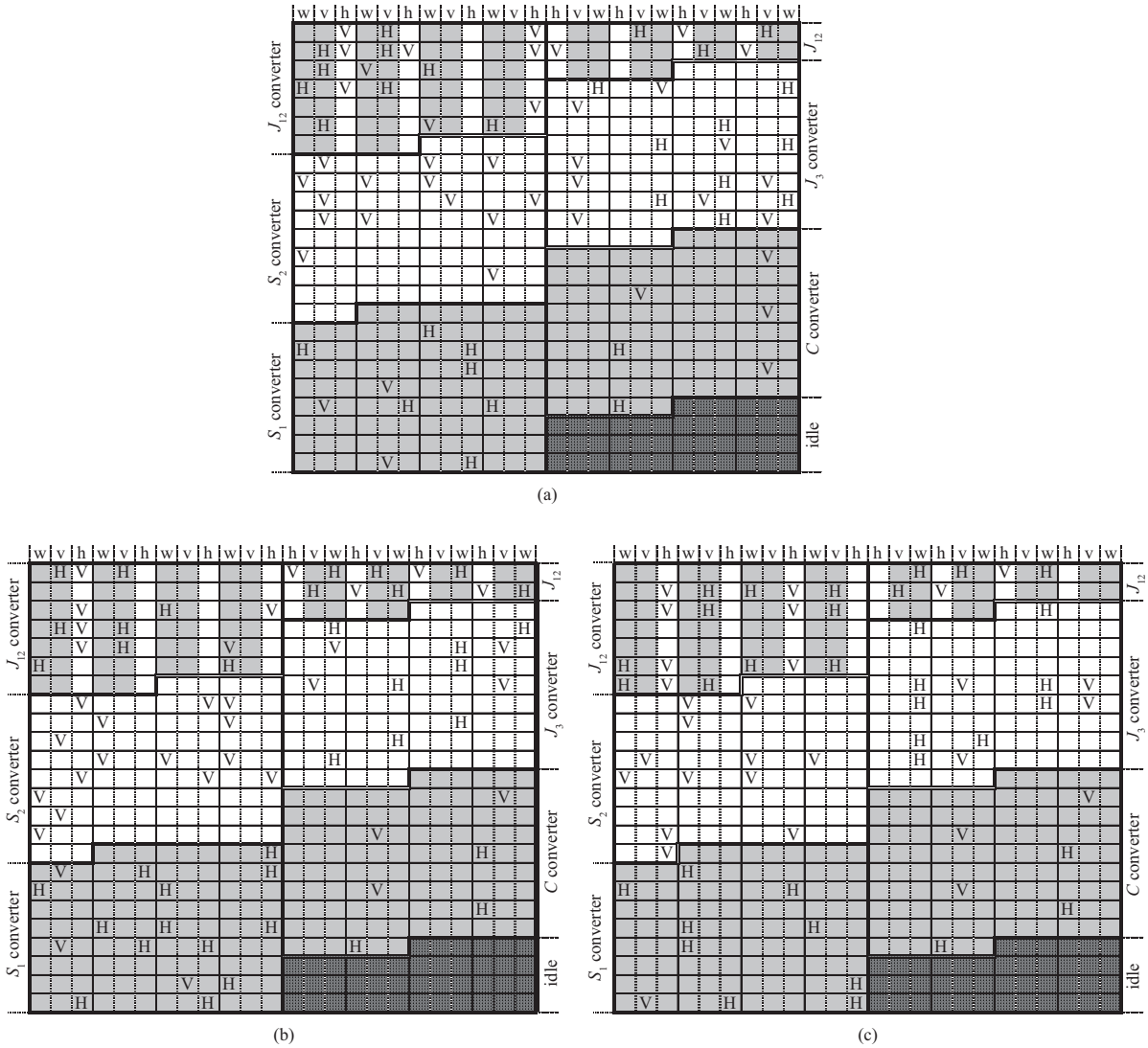

图 4　TOP 的理论输出图像

Figure 4　Theoretical output image of TOP. (a) Case 1; (b) Case 2; (c) Case 3

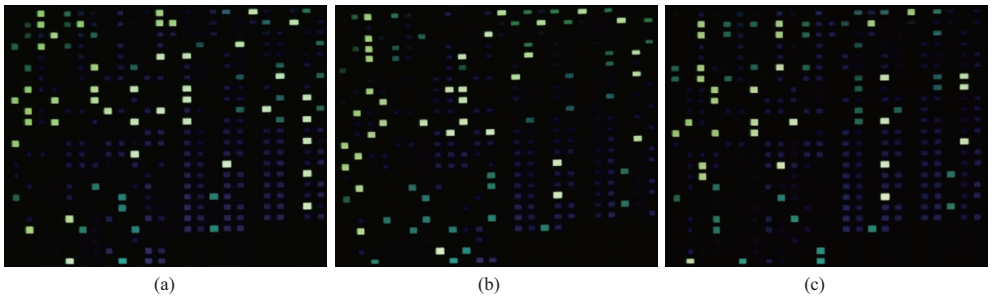

图 5　(网络版彩图) TOP 的实际实际图像

Figure 5　(Color online) Actual output image of TOP. (a) Case 1; (b) Case 2; (c) Case 3

表 4 实验结果记录表

Table 4 Record table of experimental results

Case	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Result	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

表 5 实验用例的计算结果表

Table 5 Calculation results table of experimental cases

Case	Quotient 1	Remainder 1	Quotient 2	Remainder 2	Quotient 3	Remainder 3
1	0000000011011010	0001u00000u1	0000000000000001u	01u1uu00001u	000000uuuu00u0u0	0000000u01uu
2	0000000000100011	00000u00u00u	000000u000u0uu00	0000000u1uu0	00000000u0uu00uu	0000001u00u0
3	0000000u00u00uu0	000000u000u0	00000000uu00u00u	000001u1u010	0000000010101101	000000000u0u
4	00000000uu0u00uu	000000u10000	0000000000000000	0000u010u1u0	0000001001001110	000000100u0u
5	0000000u000uuuu00	0000001u000u	0000000u00uuuu0uu	000000000u1u	0000000000010011	00000010u01u
6	000000000u0u0u0u00	000001u0100u	00000000uu000uuu	000000000000	0000000u00u0uuuu	00001u000u00
7	0000000010111100	0001uu000u01	000000000u000uu0	0001u00u0u00	0000000101011110	0000001u10u1
8	0000000010010110	000001u0uu00	00000000uuu000u0	0001u0u001u1	00000000u0000u00	0000010u0u1u
9	0000000000000000	000000000000	0000000000000001	000000000000	000000000000000u	000000000000
10	0000000000000000	000000000000	000000000000000u	000000000000	0000000000000001	000000000000
11	0000000000000000	000000000000	0000000000000000	000000000001	0000000000000000	00000000000u
12	0000000000000000	000000000000	0000000000000000	000000000001	0000000000000000	00000000000u
13	0000000000000000	000000000000	1111111111111111	000000000000	uuuuuuuuuuuuuuuu	000000000000
14	0000000000000000	000000000000	uuuuuuuuuuuuuuuu	000000000000	1111111111111111	000000000000
15	0000000000000000	000000000000	0000000100000001	000000000000	0000000u0000000u	000000000000
16	0000000000000000	000000000000	0000000u0000000u	000000000000	0000000100000001	000000000000
17	Because	the existing	divisor	is 0,	division is	terminated
18	Because	the existing	divisor	is 0,	division is	terminated
19	Because	the existing	divisor	is 0,	division is	terminated

1 至用例 16 的商与余数列在表 5 中.

分析表 3 和 5 数据可知, 并行 MSD 整数除法器的运算结果都是正确的. 在三值光学处理器上重构数据位较大的并行 MSD 整数除法器可同时运算多组除法数据, 从而能有效地提高批量除法数据的计算速度.

对于 k 组 n 位 MSD 数 A_i 除以 m 位 MSD 数 B_i ($i = 0, 1, 2, \ldots, k-1$), 需要进行 n 次定商操作. 每次定商操作, 需要做一次 MSD 数加法 $w_j + B$ 和一次 MSD 数减法 $w_j - B$. 采用同一个流水加法器完成两次 MSD 数的运算只需要 3 个机器周期 (TOP 资源受限时需要 4 个机器周期). 在完成这两次 MSD 数的加减运算后, 应判定和与差的符号, 这需要 $\lceil \log_2 m \rceil$ 个机器周期. 最后, 需要在部分余数后面进行一次拼数 a_{j+1} 操作, 需要 1 个机器周期. 因此完成一次定商操作, 需要 $4 + \lceil \log_2 m \rceil$ 个机器周期 (TOP 资源受限时需要 $5 + \lceil \log_2 m \rceil$ 个机器周期). 完成整个除法计算, 需要 $4n + \lceil \log_2 m \rceil n$ 个机器周期 (TOP 资源受限时需要 $5n + \lceil \log_2 m \rceil n$ 个机器周期). 例如 k 组 256 位 MSD 数的除法只需 $(4 + \lceil \log_2 256 \rceil) \times 256 = (4 + 8) \times 256 = 3072$ 个机器周期 (TOP 资源受限时需要 3328 个机器周期), 同理, k 组 1024 位 MSD 数的除法只需 14336 个机器周期 (TOP 资源受限时需要 15360 个机器周期).

多数据的并行 MSD 数除法跟单数据的 MSD 数除法只要被除数的数据长度一致 (取被除数长度

中最大的即可), 那么计算复杂性是相同的.

7 结论

为了充分发挥 TOC 处理器位众多、可重构、处理器位可分配、易于扩展等优势, 本文首次提出了 MSD 数的符号判定算法, 并提出了采用一个比较器和一个 SJ-MSD 加法器实现 MSD 整数除法运算方案 —— 并行 MSD 整数除法, 同时阐述了在 TOC 上具体实现一个并行 MSD 整数除法器并行计算多个除法数据的方法. 实验表明, 本文提出的并行 MSD 整数除法方案是正确可行的. 由此可见, 并行 MSD 整数除法将为 TOC 进入数值计算等实际应用领域起到积极推动作用.

参考文献

1 Obermann S F, Flynn M J. Division algorithms and implementations. IEEE Trans Comput, 1997, 46: 833–854

2 Liu H P. Research on high-performance arithmetic for floating-point division and the elementary functions. Dissertation for Ph.D. Degree. Beijing: Institute of Computing Technology, Chinese Academy of Sciences, 2003 [刘华平. 高性能浮点除法及基本函数功能部件的研究. 博士学位论文. 北京: 中国科学院研究生院 (计算技术研究所), 2003]

3 Hooman N. Architectures for floating-point division. Dissertation for Ph.D. Degree. Adelaide: Adelaide University of Australia, 2005

4 Atkins D E. The Theory and Implementation of SRT Division. Technical Report UIUCDCS-R-67-230, 1967

5 Tocher K D. Techniques of multiplication and division for automatic binary computers. Quart J Mech Appl Math, 1958, 11: 364–384

6 Xu Q, Jin Y, Shen Y F, et al. MSD iterative division algorithm and implementation technique for a ternary optical computer. Sci Sin Inform, 2016, 46: 539–550 [徐群, 金翊, 沈云付, 等. 三值光学计算机的 MSD 迭代除法算法和实现技术. 中国科学: 信息科学, 2016, 46: 539–550]

7 Liu J. Design of divider in high performance CPU. Dissertation for Master's Degree. Shanghai: Tongji University, 2007 [刘冀. 高性能 CPU 中除法器的设计. 硕士学位论文. 上海: 同济大学, 2007]

8 Yuan J H. Design and perforrmance improvement of high- perforrmance divider based on SRT algorithm. Dissertation for Master's Degree. Changsha: National University of Defense Technology, 2015 [苑佳红. 基于 SRT 算法高性能除法器设计及性能改进. 硕士学位论文. 长沙: 国防科学技术大学, 2015]

9 Jiang J B, Zhang X F, Shen Y F, et al. Design and implementation of SJ-MSD adder in ternary optical computer. Acta Electronica Sinica, 2021, 49: 275–285 [江家宝, 张晓峰, 沈云付, 等. 三值光学计算机中 SJ-MSD 加法器的设计与实现. 电子学报, 2021, 49: 275–285]

10 Jiang J B, Chen X L, Ouyang S. Hardware implementation of converting ternary optical computer MSD into standard binary data. J Nanjing Univ Sci Tech, 2016, 40: 278–284 [江家宝, 陈迅雷, 欧阳山. 三值光计算机 MSD 数转标准二进制数的硬件实现. 南京理工大学学报 (自然科学版), 2016, 40: 278–284]

11 Avizienis A. Signed-digit numbe representations for fast parallel arithmetic. IEEE Trans Electron Comput, 1961, 10: 389–400

12 Yan J Y, Jin Y, Zuo K Z. Decrease-radix design principle for carrying/borrowing free multi-valued and application in ternary optical computer. Sci China Ser F-Inf Sci, 2008, 51: 1415–1426

Design and implementation of parallel MSD integer divider in ternary optical computer

Jiabao JIANG[1,2], Yunfu SHEN[1*], Xunlei CHEN[1], Zhehe WANG[1], Yong LIU[2] & Liping CHEN[2]

1. *School of Computer Engineering and Science, Shanghai University, Shanghai* 200444*, China*;
2. *College of Information Engineering, Chaohu University, Chaohu* 238000*, China*
* Corresponding author. E-mail: yfshen@shu.edu.cn

Abstract Division is one of the four basic operations. It is a very important operation in the field of numerical calculation, such as numerical calculation, large integer decomposition and so on. How to divide quickly has been a hot issue in electronic computers, embedded systems and other new computing systems. It is of great significance to give full play to the advantages of ternary optical computer processors, such as a large number of bits, reconfigurable operation functions, bit allocation, etc. by designing the efficient parallel MSD number divider to improve the division operation efficiency of large data and promote the application of ternary optical computer in the field of numerical calculation. In this paper, the symbol decision algorithm for MSD numbers is proposed first time, and based on the SRT algorithm, the scheme for implementing a parallel division of multiple MSD integers using a parallel carry free SJ-MSD adder and an MSD comparator, which is called as parallel MSD integer division, is designed first time. For the corresponding parallel MSD integer divider, the machine cycles required to run multiple MSD integer division in parallel is the same as the machine cycles required for the operation of a single MSD integer division. Experiments show that the parallel SRT integer division scheme is feasible. It will effectively improve the efficiency of large data processing and accelerate TOC to enter practical applications such as numerical computation.

Keywords parallel MSD integer divider, SJ-MSD adder, comparator, SRT division, ternary optical computer (TOC)

Jiabao JIANG was born in 1968. He received a master's degree from Jiangnan University in 2007. Currently, he is studying for a doctorate at the School of Computer Engineering and Science, Shanghai University. He is currently an associate professor at Chaohu University. His research interests include ternary optical computer and embedded system.

Yunfu SHEN was born in 1960. He received a Ph.D. degree in science from Beijing Normal University, Beijing in 1996. He is currently a professor at Shanghai University. His research interests include reliability and operation units for ternary optical computers, formalization methods for software and hardware, and model checking.

Yong LIU was born in 1981. He received a master's degree from Hohai University, Nanjing, in 2012. Currently, he is an associate professor at Chaohu University. His research interests include ternary optical computer, embedded system, database system and data mining.

Liping CHEN was born in 1981. She received master's degree from Xi'an University of Technology in 2009. Currently, she is an instructor at Chaohu University. Her research interests include ternary optical computer, embedded system, machine learning and data mining.