DOI:10. 13382/j. jemi. B2508192

基于查找表和并行免缩放迭代的混合 CORDIC 算法

魏学静　孙　皓　彭　宇　刘连胜
(哈尔滨工业大学电子与信息工程学院　哈尔滨　150080)

摘　要:坐标旋转数字计算机(CORDIC)因其简易的硬件实现,在电子测量、雷达探测和图像处理等诸多领域得到了广泛应用。高基数和并行 CORDIC 能够有效降低 CORDIC 迭代延迟,能够满足实时性要求较高的需求。但是,两者均引入了可变缩放因子,这增加了缩放因子的计算复杂度,引起额外的资源消耗。相比而言,免缩放 CORDIC 算法消除了可变的缩放因子。然而,现有的大多数免缩放 CORDIC 算法虽然能保持一定精度并支持较大收敛范围,但是在资源消耗和延迟指标上仍有待提升。因此,提出了一种结合查找表(LUT)和并行免缩放迭代的混合 CORDIC 算法及其计算结构设计,采用具有更少非零项的近似角度确定查找表与并行免缩放迭代角度边界的方法,扩展了并行免缩放迭代所支持的角度收敛范围;提出将并行迭代分为二并行迭代和四并行迭代来平衡每级迭代的计算复杂度,以保证整体设计性能的方法,LUT 用于将位于区间($-\pi/2,\pi/2$)的大角度输入快速折叠至二并行迭代所支持的角度收敛范围;然后执行二并行免缩放迭代,使得剩余角度进入到四并行免缩放迭代所支持的角度收敛范围;最后执行四并行免缩放迭代并输出 CORDIC 迭代结果。所提出的设计使用 Verilog 硬件描述语言实现,并在现场可编程门阵列(FPGA)上完成验证和性能评估。实验结果表明,与已有相关研究相比,本文提出的设计在保持相同精度和收敛范围的同时,资源消耗降低 23. 1%,延迟降低 22. 1%。

关键词: 免缩放 CORDIC;FPGA;低延迟;低资源消耗;并行迭代
中图分类号: TN98　　**文献标识码**: A　　**国家标准学科分类代码**: 510. 40

Hybrid CORDIC algorithm based on LUT and parallel scaling-free iterations

Wei Xuejing　Sun Hao　Peng Yu　Liu Liansheng
(School of Electronics and Information Engineering, Harbin Institute of Technology, Harbin 150080, China)

Abstract: The coordinate rotation digital computer (CORDIC) algorithm has the feature of simple hardware implementation. It has been widely applied in various fields, such as electronic measurement, radar detection, and image processing. High-radix and parallel CORDIC effectively reduce CORDIC iteration latency to meet the real-time requirements. However, both approaches introduce a variable scaling factor, increasing the computational complexity and results in additional resource consumption. In comparison, scaling-free (SF) CORDIC algorithm eliminates the variable scaling factor. However, the most existing SF-CORDIC algorithms still require improvements in resource consumption and latency performance, while maintaining acceptable accuracy and supporting a wide convergence range. Therefore, this article proposes a hybrid CORDIC algorithm and its computing architecture design, which combines the look-up table (LUT) and parallel SF iterations. A method is proposed to determine the angle boundary between the LUT and parallel SF iterations using approximate angles with fewer non-zero terms, which extends the convergence range supported by the parallel SF iterations; furthermore, a method is proposed to divide the parallel SF iterations into two-parallel and four-parallel SF iterations to balance the computational complexity of each iteration stage, ensuring the overall design performance. Specifically, the LUT is used to rapidly fold a large-angle input located in the range ($-\pi/2,\pi/2$) into the convergence range supported by the two-parallel SF iterations. Then, the two-parallel SF iterations are performed to bring the residual angle into the convergence range supported by the four-parallel SF iterations. Finally, the four-parallel SF iterations are performed and the CORDIC iteration results are output. The proposed design is implemented in Verilog hardware description language and validated on field-programmable gate array (FPGA). Experimental results

收稿日期: 2025-02-26　　Received Date: 2025-02-26

demonstrate that, compared with the existing designs, the proposed design reduces resource consumption by 23.1% and latency by 22.1%, while maintaining comparable accuracy and convergence range.

Keywords: scaling-free CORDIC; field-programmable gate array; low latency; low resource consumption; parallel iterations

0 引　言

随着当代科学技术的发展,多种工业场景对各类测试测量设备和电子系统的响应速率和精度提出了更高要求。数字信号处理算法作为这些设备和系统的重要组成部分,在保证计算精度的前提下提升其计算的实时性是有必要的。例如在基于带宽交织的实时数据采集系统中,要求对频率为 GHz 以上的信号进行高实时低延迟的数字信号处理,相关数字信号处理算法包括但不限于如数字本振的正余弦函数信号生成和三参数正弦拟合算法中用于相差估计的快速傅里叶变换(fast Fourier transform, FFT)[1-2]。而在高速高精度的相位敏感光时域反射仪中,需要对大量数据进行实时正交解调以使得反射仪准确快速获取目标信息[3]。而其余如数字干涉图实时相位计算[4]、高精度光栅测量系统的误差分离和补偿计算[5]和现代通信系统[6]中,都需要实时高效的数字信号处理算法实现。

坐标旋转数字计算机(coordinate rotation digital computer, CORDIC)算法由 Volder 于 1959 年提出,因其简单的硬件计算结构,被广泛用于各类数字信号处理算法中的复杂函数计算,如正余弦函数、双曲函数、开根号和乘除法等[7-8],一定程度上提升了数字信号处理算法的计算速度。然而,传统 CORDIC 算法的收敛速度仍较慢,为达到更高精度的结果输出需要执行多次迭代,不适用于对实时性要求很高的场合。为降低延迟,多种 CORDIC 算法已被提出,如分支 CORDIC 算法[9]、并行 CORDIC 算法[10]和高基数 CORDIC 算法[11-12]。尽管这些算法减少了迭代次数,实现了低延迟,但引入了可变缩放因子,使得缩放因子计算复杂度增加,从而导致资源消耗较高的问题。

较早的免缩放(scaling-free, SF)CORDIC 算法通过泰勒级数近似消除了缩放因子的计算和补偿,但其收敛范围较小。为解决该问题,Maharatna 等[13]提出了一种改进的免缩放 CORDIC 算法以扩展收敛范围。然而该算法并未显著减少迭代次数以实现低延迟,同时引入了两个额外的缩放因子,且精度较低。为扩展收敛范围并保持较高的计算精度,一些设计在处理大角度时采用了更高阶的泰勒级数近似[14]并扩展到了双曲函数的免缩放 CORDIC[15],但这些设计通常需要更高的资源消耗。Moroz 等[16]提出了一种结合查找表(look-up table, LUT)、免缩放迭代和乘法操作的混合 CORDIC 算法。在免缩放迭代部分,并未采用并行设计来进一步减少迭代次数;在后半段迭代中使用了乘法运算,显著增加了资源开销;同时每级迭代资源的分配不平衡使用该设计整体性能不高。Wu 等[17]提出了一种改进免缩放 CORDIC 算法用于信号源的信号生成。该算法结合传统 CORDIC 迭代和免缩放 CORDIC 迭代,通过对输入角度进行 Booth 编码以减少迭代次数,但是这引入了较为复杂的编码电路。Xue 等[18]则是结合传统 CORDIC 和免缩放迭代,并运用角度折叠技术提出了另一种改进免缩放 CORDIC 算法,但是 CORDIC 迭代结果精度较低。另一种采用混合基数的不完全免缩放因子 CORDIC 算法[19]通过结合传统 CORDIC、高基数设计和免缩放迭代,扩展了收敛范围并减少了迭代次数。然而,该算法引入了 4 次传统 CORDIC 迭代,仍然需要额外的缩放因子计算电路,同时传统 CORDIC 迭代部分限制了延迟和资源消耗的进一步优化降低。

综上,已有相关研究在免缩放 CORDIC 的收敛角度范围、精度和延迟等方面做出了许多改进工作,但是在资源消耗方面考虑较少,同时延迟方面仍存在改进提升空间。本文针对现有免缩放 CORDIC 在资源消耗和延迟等方面的问题,提出基于查找表和并行免缩放迭代的混合 CORDIC 算法,并给出其计算结构实现。此外,为了实现资源消耗和延迟指标的进一步优化,引入近似角度用于确定查找表与并行免缩放迭代的角度边界,并将这些近似角度作为并行免缩放迭代过程中的旋转角,这有利于扩展并行免缩放迭代所支持的角度范围并降低并行迭代过程的迭代延迟和资源消耗。同时,本次设计将并行免缩放迭代细分为二并行免缩放迭代和四并行免缩放迭代以平衡每级迭代的资源消耗,保证所提出设计的整体性能。

1 相关理论

1.1 传统 CORDIC 算法

在圆形坐标系中,从角度 α 旋转到角度 β 如图 1 所示,其中旋转角度是 θ。角度 α 与角度 β 的坐标关系由式(1)可得。

$$\begin{bmatrix} x_\beta \\ y_\beta \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \cdot \begin{bmatrix} x_\alpha \\ y_\alpha \end{bmatrix} \tag{1}$$

CORDIC 算法将角度 θ 分解为不同角度的线性组合,如式(2)所示。

$$\theta = \sum_{i=0}^{n-1} d_i \cdot \theta_i \tag{2}$$

式中：n 表示 CORDIC 旋转迭代的次数；d_i 表示旋转方向且 $d_i \in \{-1, +1\}$。

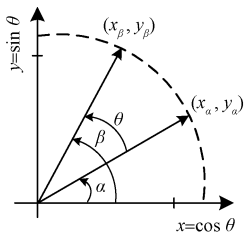


图 1　CORDIC 圆坐标旋转

Fig. 1　CORDIC rotation in circular coordinates

CORDIC 的关键在于将角度 θ_i 取值为 $\tan^{-1}(2^{-i})$。由此，CORDIC 的第 $i+1$ 次迭代由式(3)可得。

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = K_i \cdot \begin{bmatrix} 1 & -d_i \cdot 2^{-i} \\ d_i \cdot 2^{-i} & 1 \end{bmatrix} \cdot \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$
$$z_{i+1} = z_i - d_i \cdot \theta_i$$
$$K_i = \cos(\tan^{-1}(d_i \cdot 2^{-i})) = 1/\sqrt{1 + (d_i \cdot 2^{-i})^2} \tag{3}$$

其中，K_i 对应当前迭代的缩放因子。从 x_i 到 x_{i+1} 的计算被称为 x 路径，同理从 y_i 到 y_{i+1} 的计算被称为 y 路径，而从 z_i 到 z_{i+1} 的计算被称为剩余角度 z 路径。由式(3)可以发现计算 x_{i+1} 和 y_{i+1} 只需要进行移位和加减法操作。

1.2　免缩放因子 CORDIC 算法

免缩放 CORDIC 算法使用正余弦函数的泰勒级数近似消除了可变缩放因子，被称为免缩放操作。当 $\theta_i = 2^{-i}$ 时，对应的泰勒级数如式(4)所示。

$$\sin(2^{-i}) = 2^{-i} - \frac{2^{-3i}}{3!} + \frac{2^{-5i}}{5!} - \frac{2^{-i7}}{7!} + \cdots$$
$$\cos(2^{-i}) = 1 - \frac{2^{-2i}}{2!} + \frac{2^{-4i}}{4!} - \frac{2^{-6i}}{6!} + \cdots \tag{4}$$

基本的免缩放 CORDIC 只使用泰勒级数的第 1 项，结合式(1)和(4)，得到的免缩放 CORDIC 的迭代公式如式(5)所示，其中 $d_i \in \{0, +1\}$，同式(2)中的 d_i 符号含义一致，表示旋转方向，但只存在不旋转($d_i = 0$)或者逆时针旋转($d_i = +1$)两种选择。

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} 1 - 2^{-(2i+1)} & -d_i \cdot 2^{-i} \\ d_i \cdot 2^{-i} & 1 - 2^{-(2i+1)} \end{bmatrix} \cdot \begin{bmatrix} x_i \\ y_i \end{bmatrix} \tag{5}$$

旋转角度可被表示为 $b$bit 的二进制定点数，式(5)只适用于 $i \geqslant \lceil (b - \log_2(3!))/3 \rceil$ 的迭代[13]，这限制了免缩放 CORDIC 的收敛范围。相较于式(3)，式(5)没有缩放因子 K_i，但需要额外的两个移位操作和减法操作。当 $i \geqslant \lceil (b - \log_2(2!))/2 \rceil$ 时，项 $2^{-(2i+1)} < 2^{-b}$，已经不能采用 b bit 的有符号二进制数表示，所以此时式(5)可化简为式(6)，相比于式(5)，没有额外的两个移位操作和两个减法操作。

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & -d_i \cdot 2^{-i} \\ d_i \cdot 2^{-i} & 1 \end{bmatrix} \cdot \begin{bmatrix} x_i \\ y_i \end{bmatrix} \tag{6}$$

1.3　CSD 编码和有效字长 WL_E

当旋转角度 $\theta_i > 2^{\lceil (b-\log_2(3!))/3 \rceil}$ 时，如果要在保证精度的前提下进行免缩放 CORDIC 迭代，就要采用式(4)中更高阶数的泰勒级数项，但这需要更多的移位和加减法操作，体现出资源与 CORDIC 迭代精度的矛盾。CSD 编码以最少的非零项表示数值，从而有效降低资源消耗。数值 A 可以被表示为式(7)，这是一种三态表示形式。CSD 编码的核心特点时避免了连续非零位，从而有效降低了计算中的移位操作和加减法操作。

$$A = \sum_{i=0}^{b-1} a_i \cdot 2^{-i}, a_i \in \{-1, 0, +1\} \tag{7}$$

有限字长 WL_E 是作为旋转系统的旋转误差指标，由式(8)计算得到。其中 ε_s 表示正弦函数 $\sin\theta$ 的定点量化误差，而 ε_c 则表示余弦函数 $\cos\theta$ 的真实值与定点量化值之间的误差。为了实现准确的量化，对于一个 $b$bit 有符号数，需要保证 $WL_E \geqslant b$。

$$WL_E(\theta) = -\log_2(\sqrt{\varepsilon_s^2 + \varepsilon_c^2}) + 1.5 \tag{8}$$

表 1 为 $\theta \in \{2^0, 2^{-1}, \cdots, 2^{-15}\}$ 对应的正余弦函数值的 16 bit 定点数表示。

表 1　角度 2^{-i} 的 CSD 编码结果及对应近似角的 16 bit 字长表达

Table 1　The CSD encoding results of the angle 2^{-i} and it's modified angle with 16-bit word length

iter(i)	CSD Method			Modified Angles			
	θ/rad	$\sin\theta$	$\cos\theta$	θ/rad	$\sin\theta$	$\cos\theta$	WL_E
0	1	$1-2^{-3}-2^{-5}-2^{-9}-2^{-11}+2^{-13}+2^{-15}$	$2^{-1}+2^{-5}+2^{-7}+2^{-10}+2^{-12}+2^{-15}$	1.065 39	$1-2^{-3}$	$2^{-1}-2^{-6}-2^{-12}$	25.1
1	0.5	$2^{-1}-2^{-6}-2^{-8}-2^{-10}-2^{-14}$	$1-2^{-3}+2^{-9}+2^{-11}+2^{-13}+2^{-15}$	0.505 34	$2^{-1}-2^{-6}-2^{-12}$	$1-2^{-3}$	19.5
2	0.25	$2^{-2}-2^{-9}-2^{-11}-2^{-13}-2^{-15}$	$1-2^{-5}+2^{-13}+2^{-15}$	0.252 65	2^{-2}	$1-2^{-5}-2^{-11}$	17.5
3	0.125	$2^{-3}-2^{-11}+2^{-13}+2^{-15}$	$1-2^{-7}$	0.125 31	2^{-3}	$1-2^{-7}-2^{-15}$	23.5
4	0.062 5	$2^{-4}-2^{-15}$	$1-2^{-9}$	0.062 53	2^{-4}	$1-2^{-9}$	20.5
5~7	2^{-i}	2^{-i}	$1-2^{-(2i+1)}$	2^{-i}	2^{-i}	$1-2^{-(2i+1)}$	—
8~15	2^{-i}	2^{-i}	1	2^{-i}	2^{-i}	1	—

表 1 中的“CSD-Method”一列呈现了各角度正余弦函数值的 CSD 编码结果,但仍存在较多非零项。为解决该问题,Changela 等[20]依据 CSD 编码和 WL_E,选择了一些调整后的近似角度作为原有对应角度的替代。这些调整后的近似角度的 CSD 编码结果具有更少非零项,这代表着更少的移位和加减法操作。这些近似角度如表 1 中“Modified Angles”。

2　基于 LUT 和并行免缩放迭代的混合 CORDIC 算法及其计算结构

2.1　基于 LUT 和并行免缩放迭代的混合 CORDIC 算法

为了叙述方便,采用 $FIX(s,bit,frac)$ 来对一个定点数格式的数据进行表示。此处“s”表示数据是有符号数($s=1$)或者无符号数($s=0$);“bit”表示定点数的总 bit 位数;“$frac$”则用于表示定点数的小数位数。此外,定义表达式 $Tnum(\theta,b,f)$ 用于表示正余弦函数 $\sin(\theta)$ 和 $\cos(\theta)$ 经过 CSD 编码后的非零项之和(此处“b”表示有符号数位宽,“f”表示数据的小数位宽)。例如在表 1 中,角度 $\theta=0.125$(小数位数为 15 bit)的正弦值 $\sin(0.125)$ 经过 CSD 编码后,非零项为 4,其余弦值 $\cos(0.125)$ 经过 CSD 编码后,非零项为 2,一共为 6 个非零项,所以 $Tnum(0.125,16,15)=6$。

如图 2 所示,CORDIC 迭代被分为 3 个部分,包括查找表部分、二并行度免缩放迭代和四并行度免缩放迭代。输入角度的范围是 $[0,\pi/2)$,表示格式 $FIX(0,b,b-1)$,其二进制表示形式为:

$$\theta=\sum_{i=0}^{b-1}a_i2^{-i},a_i\in\{0,1\} \tag{9}$$

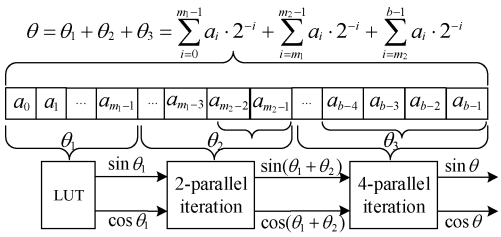


图 2　提出算法的总体框图

Fig. 2　Framework of the proposed algorithm

输入角度 θ 被分为 3 部分,如式 (10) 所示,表示依次被查找表、二并行度免缩放迭代和四并行度免缩放迭代 3 个阶段进行处理。确定式 (10) 中的参数 m_1 和 m_2 是本次设计的关键。

$$\theta=\theta_1+\theta_2+\theta_3$$

$$\theta_1=\sum_{i=0}^{m_1-1}a_i\cdot2^{-i};\theta_2=\sum_{i=m_1}^{m_2-1}a_i\cdot2^{-i};\theta_3=\sum_{i=m_2}^{b-1}a_i\cdot2^{-i} \tag{10}$$

根据式 (1),并行迭代的总体表达如式 (11) 所示,表示将第 k^{th} 次迭代及其之后的 $m-1$ 次迭代合并为一次迭代;z_{k+m} 是 z 路径中的剩余角度。

$$\begin{bmatrix}x_{k+m}\\y_{k+m}\end{bmatrix}=\left(\prod_{i=k}^{k+m-1}\begin{bmatrix}\cos\theta_i & -d_i\sin\theta_i\\ d_i\sin\theta_i & \cos\theta_i\end{bmatrix}\right)\cdot\begin{bmatrix}x_k\\y_k\end{bmatrix}$$

$$z_{k+m}=z_k-\sum_{i=k}^{k+m-1}\theta_i,d_i\in\{0,1\} \tag{11}$$

1)四并行度免缩放迭代

四并行度免缩放迭代完成式 (10) 对应的旋转角度 θ_3,其中 $i\in\{m_2,m_2+1,\cdots,b-1\}$。合并多次迭代为单次迭代有助于减少迭代次数从而减少 CORDIC 整体的迭代延迟。虽然更高并行度的迭代合并是可行的,但这会导致合并后的单次并行迭代具有较高的计算复杂度,容易成为数字系统中的关键路径,降低设计的整体运行频率,反而增加一定的延迟。根据研究工作表明,四并行度迭代能表现出较好性能,因此在本部分中,采用四并行度迭代方式。从第 m_2 开始,接下来的迭代以 4 次迭代为一组合并为单次四并行度迭代,其中 $m_2=\lceil(b-\log2(2!))/2\rceil$。根据式(6)和 (11),单次四并行度迭代表达式如式(12)所示。其中$<2^{-b}$ 的项已经被移除,因为这一部分小于采用 $FIX(0,b,b-1)$ 格式所能表示的最小范围,故移除。根据式(12)所示,单次四并行度迭代的 x 路径和 y 路径各自需要 4 次移位操作和 4 次加减法操作。在式 (12)中,旋转角度为 $d_k2^{-k}+d_{k+1}2^{-(k+1)}+d_{k+2}2^{-(k+2)}+d_{k+3}2^{-(k+3)}$,这对应角度 z_k 的 bit 位 $z_k[b-1-k:b-1-(k+3)]$,在实现时,并不需要减法操作,故不消耗资源。

$$\begin{bmatrix}x_{k+4}\\y_{k+4}\end{bmatrix}=\begin{bmatrix}1 & -\sum_{i=k}^{k+4-1}d_i2^{-i}\\ \sum_{i=k}^{k+4-1}d_i2^{-i} & 1\end{bmatrix}\cdot\begin{bmatrix}x_k\\y_k\end{bmatrix}$$

$$z_{k+4}=z_k-\sum_{i=k}^{k+4-1}d_i2^{-i} \tag{12}$$

2)二并行度免缩放迭代

在该迭代部分,要求旋转角 θ_i 正弦函数 $\sin\theta_i$ 表达式至少有一个非零项,余弦函数 $\cos(\theta_i)$ 表达式至少有两个非零项。根据式 (11)的并行合并方式,单次并行迭代的旋转角 $d_{m_2-(2n+1)}\theta_{m_2-(2n+1)}+d_{m_2-(2n+1)}\theta_{m_2-(2n+2)}$ 需要更多的非零项进行表示,这意味着需要更多的移位操作和加减法操作,导致更多的资源消耗。旋转方向 $d_{m_2-(2n+1)}$ 和 $d_{m_2-(2n+2)}$ 可以取数值“0”表示不进行旋转,也可以取数值“1”表示进行旋转,因此根据组合排列,单次二并行度迭代可能旋转的角度为 0、$\theta_{m_2-(2n+1)}$、$\theta_{m_2-(2n+2)}$ 和 $\theta_{m_2-(2n+1)}+\theta_{m_2-(2n+2)}$。在这些旋转角度中,“0”表示不旋转,是无效旋转角度,而其余 3 个角度都是有效的旋转角度。为了

平衡每一级迭代的资源消耗并提升设计的整体性能，参考四并行度迭代，要求 3 个旋转角度 $\theta_{m_2-(2n+1)}$、$\theta_{m_2-(2n+2)}$ 及 $\theta_{m_2-(2n+1)}+\theta_{m_2-(2n+2)}$ 对应的迭代旋转过程的非零项不超过 5，如式（13）所示。

$$\begin{cases} Tnum(\theta_{m_2-(2n+1)},b,b-1)\leqslant 5 \\ Tnum(\theta_{m_2-(2n+2)},b,b-1)\leqslant 5 \\ Tnum(\theta_{m_2-(2n+1)}+\theta_{m_2-(2n+2)},b,b-1)\leqslant 5 \end{cases} \tag{13}$$

为扩展二并行度免缩放迭代支持的角度范围并降低资源消耗和延迟，采用非 2^{-i} 角度的近似角度是可行的，这些近似角度的选取基于有限字长 $WL_E(\theta)\geqslant b$ 和经过 CSD 编码非零项不超过 5 的原则（满足式（13））。具体确定如算法 1 所示，其中，A 和 B 确定搜索近似角度的区间为 $[A\cdot 2^{b-1},B\cdot 2^{b-1}]$；$b$ 是式（13）中表示角度的二进制位宽。

算法 1：搜索满足非零项不超过 5 的近似角度

```
0  input A,B,b;
1  output angleList[ ];
2  for i=A;i<=B;i++{
3    angle = i*2^(-(b-1))
4    sin_value = sin(angle); cos_value = cos(angle);
5    sin_fix_value = FIX(sin_value); cos_fix_value = FIX(cos_value);
6    sin_fix_csd_value=CSD(sin_fix_value);
     cos_fix_csd_value=CSD(cos_fix_value);
7    cos_NoZeroTerms_num = getNoZeroTerms(sin_fix_csd_value);
     sin_NoZeroTerms_num = getNoZeroTerms(cos_fix_csd_value);
8    angle_NoZerosTerms_num=
       cos_NoZeroTerms_num+sin_NoZeroTerms_num;
9    angle_real = arctan(sin_fix_value/ cos_fix_value);
10   sin_error=sin(angle_real)-sin_fix_value;
     cos_error=cos(angle_real)-cos_fix_value;
11   WLe=-log2(sqrt(sin_error^2+ cos_error^2))+1.5;
12   if(Wle>=b && angleNoZerosTerms<=5)
     Append Angle in AngleList[ ];
13 }
```

算法 1 的第 5 步是对第 4 步正余弦结果进行定点化处理；第 6 步完成对定点数的 CSD 编码；第 7 步和第 8 步统计正余弦数值经过 CSD 编码后二进制的非零项个数。从第 3 步到第 8 步是表达式 $Tnum(\theta,b,b-1)$ 的实现过程。由于第 5 步对角度 *angle* 的正余弦函数值进行了定点量化，定点量化结果的实际对应的角度并非角度 *angle*，而是第 9 步计算的角度 *angle_real*。第 9 步到第 10 步是根据式（8）求取有限字长 WL_E，其中 *sin_error* 和 *cos_error* 分别对应式（8）中的 ε_s 和 ε_c。此外，选取的近似角度需要满足 CORDIC 的迭代收敛过程。总之，确定 m_1 的过程是从 m_2-1 向前逐对合并直到不满足式（13）为止。

3）查找表设计

查找表通过一次访存等效地将输入的角度 $\theta\in[0,\pi/2)$ “折叠”到区间 $[0,2^{-(m_1-1)})$，等效实现了一次大角度范围的“迭代”，弥补了免缩放迭代角度小的问题。将支持的角度 $[0,\pi/2)$ 等间隔分为若干份，间隔为 $2^{-(m_1-1)}$，形成的角度合集为 $\{0,2^{-(m_1-1)},\cdots,k\times 2^{-(m_1-1)}\}$，而 $k=\lfloor(\pi/2)\div 2^{-(m_1-1)}\rfloor$。在查找表中，存储了角度合集对应的 $k+1$ 组正余弦函数值。

4）迭代收敛过程

旋转角度的选择基于旋转角度取值所定义的边界条件。在单次迭代中，假设旋转角集合为 $\{0,E_1,E_2,\cdots,E_{n-1}\}$。图 3 所示为一次迭代过程中旋转角选择，当某一角度 A 位于区间 $[E_i,E_{i+1})$，则选取的旋转角度为 E_i，而剩余角度 $A-E_i$ 将位于区间 $[0,E_{i+1}-E_i)$。

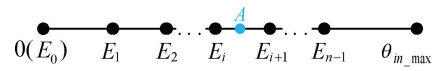


图 3 迭代收敛的角度选择与收敛示意图

Fig. 3 The rotation angle selection and convergence diagram

表 2 五级流水线结构中（16 bit）使用的旋转角度

Table 2 The angles used in the 5-stage pipeline architecture with 16-bit

Stage	θ_i/rad	$\theta_i\cdot 2^{15}$	sin-expression	cos-expression	WL_E	Selection conditions
1	$k_1\cdot 2^{-3}$	$k_1\cdot 4096$	from LUT-sin	from LUT-cos	—	$0\leqslant\theta\leqslant\pi/2$
2	0.093 87	3 076	$2^{-4}+2^{-5}$	$1-2^{-8}-2^{-11}$	18.1	$0.093\ 87\leqslant z_1$
	0.062 53	2 049	2^{-4}	$1-2^{-9}$	20.5	$0.062\ 53\leqslant z_1<0.093\ 87$
	2^{-5}	1 024	2^{-5}	$1-2^{-11}$	24.5	$2^{-5}\leqslant z_1<0.062\ 53$
3	2^{-5}	1 024	2^{-5}	$1-2^{-11}$	24.5	$2^{-5}\leqslant z_1$
	$2^{-6}+2^{-7}$	768	$2^{-6}+2^{-7}$	$1-2^{-12}-2^{-15}$	26.2	$(2^{-6}+2^{-7})\leqslant z_1<2^{-5}$
	2^{-6}	512	2^{-6}	$1-2^{-13}$	28.5	$2^{-6}\leqslant z_1<(2^{-6}+2^{-7})$
	2^{-7}	256	2^{-7}	$1-2^{-15}$	32.5	$2^{-7}\leqslant z_1<2^{-6}$
4	$k_4\cdot 2^{-11}$	$k_4\cdot 16$	$k_4\cdot 2^{-11}$	1	—	$2^{-11}\leqslant z_1<2^{-7}$
5	$k_5\cdot 2^{-15}$	k_5	$k_5\cdot 2^{-15}$	1	—	$2^{-15}\leqslant z_1<2^{-11}$

对于输入角度可能位于图 3 中的任意一个区间，所以本次迭代后输出的剩余角度范围可由式（14）表示。为了确保 CORDIC 迭代的收敛性，前一次迭代的输出范围不应超出当前迭代的输入范围，同时当前迭代的输出范围不应超出下一次迭代支持的输入范围。确保迭代收敛的实质就是保证设计的 CORDIC 经过多次迭代能够覆盖所支持输入角度范围内的任意角度。

$$[0,\max\{E_1-E_0,E_2-E_1,\cdots,\theta_{in\_max}-E_{n-1}\}) \quad (14)$$

2.2　基于 LUT 和并行免缩放迭代的混合 CORDIC 算法的计算结构实现

在该小节给出所提出算法 16 bit 实现的计算结构设计。根据 2.1 所提出的算法，16 bit 实现对应的计算结构被分为 5 级流水线进行，如图 4 所示，其中 x 路径和 y 路径采用的数据格式为 $FIX(1,17,16)$，在最后输出阶段截断最低 1 bit，输出结果的数据格式为 $FIX(1,16,15)$。而对于可支持的输入角度范围可以通过使用 1 bit 符号位从 $[0,\pi/2)$ 扩展到 $(-\pi/2,\pi/2)$，因此输入角度 θ 的数据格式为 $FIX(1,17,15)$。在 Stage−1 阶段，输入角度 θ 完成符号检测和绝对值取值后，其数据格式将变为 $FIX(0,16,15)$，这与提出的算法是一致的。

图 4 中，Stage-1 阶段执行查找表 LUT 步骤，Stage-2 和 Stage-3 阶段执行二并行度免缩放迭代，而 Stage-4 和 Stage-5 阶段执行四并行度免缩放迭代。每个阶段的旋转角度和判定方法如表 2 所示，其中近似角度 0.093 87 和 0.062 53 经过算法 1 和迭代收敛过程确定。为了后续叙述的简易性，采用 $\theta_i\cdot 2^{15}$（整数）替代实际上角度值 θ_i（小数，弧度制）来描述整个迭代过程。

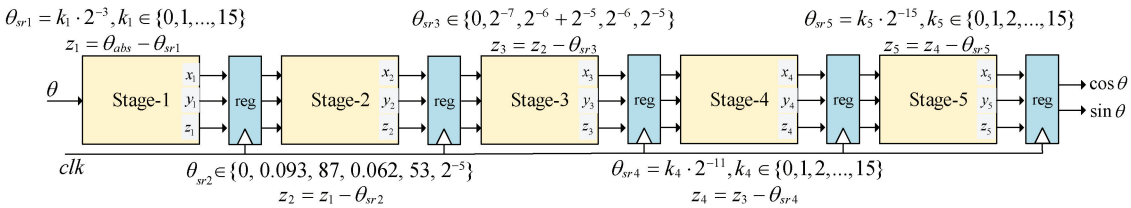
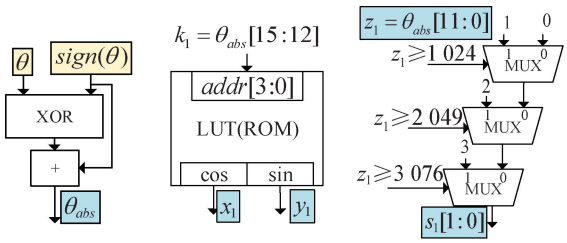


图 4　16 bit 字长的五级流水线计算结构框图

Fig. 4　5-stage pipeline computing structure diagram with 16-bit word length

1）Stage-1 阶段

第 1 阶段使用查找表将输入范围在 $(-\pi/2,\pi/2)$ 的输入角度映射到 $(0,2^{-3})$。在 $[0,\pi/2)$ 内以 2^{-3} 为步长构建查找表。该查找表存储了 $0,2^{-3},2\times2^{-3},\cdots,12\times2^{-3}$ 等 13 个角度对应的正余弦函数值并采用 $FIX(1,17,16)$ 进行量化。图 5 是 Stage-1 的计算结构示意图。



(a) 角度θ绝对值的计算结构
(a) Computing structure for absolute value of angle θ

(b) 查找表输出x_1和y_1
(b) The LUT for output x_1 and y_1

(c) 输出z_1和输出选择符号s_1的计算结构
(c) Computing structure for output z_1 and output selection signal s_1

图 5　Stage-1 的计算结构

Fig. 5　Computing architecture for stage-1

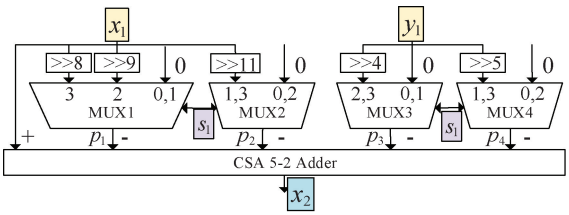
图 5(a) 中，输入角度 θ 与其符号位 $sign(\theta)$ 进行异或运算，得到的结果再与 $sign(\theta)$ 进行加法运算得到输入角度 θ 的绝对值 θ_{abs}。图 5(b) 中，查找表由 ROM 实现，并将输入角度绝对值 θ_{abs} 的 bit 位映射为查找表的查找地址：$addr[3:0]=k_1=\theta_{abs}[15:12]$。经过 Stage-1 后的剩余角度 z_1 等于 $\theta_{abs}[11:0]$，这直接从绝对值 θ_{abs} 的 bit 位取得而不需要常规的减法器进行运算。在图 5(c) 中，剩余角度 z_1 与 Stage-2 阶段的旋转角度进行比较从而得到旋转角度选择型号 $s_1[1:0]$。

2）Stage-2 阶段

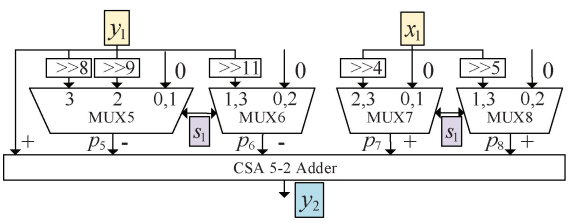
在 Stage-1 阶段完成后，剩余角度 $z_1=\theta_{abs}[11:0]$ 位于区间 $[0:4095)$ 且旋转角度选择信号是 $s_1[1:0]$。图 6(a) 是计算输出数据 x_2 的计算结构图，其中包含了 4 个中间输出，分别是 p_1（多路选择器 MUX1 的输出）、p_2（多路选择器 MUX2 的输出）、p_3（多路选择器 MUX3 的输出）及 p_4（多路选择器 MUX4 的输出）。这 4 个中间输出均由选择信号 $s_1[1:0]$ 进行选择而输出的。将这 4 个中间输出与输入数据 x_1 进行相加从而得到 x 路径的输出数据 x_2，为了减少路径延迟，此处采用了 5 输入 2 输出的进位保留加法器（carry-save adder，CSA），写为 CSA5-2，然后由一个超前进位加法器得到输出数据 x_2。选择信号的执行过程是：例如当 $s_1=1$，被选中的旋转角度是表 2 中的 1 024（$2^{-5}\times2^{15}=1\,024$），而对应的部分积是 $p_1=0,p_2=x_2\gg11,p_3=0,p_4=y_1\gg5$，这对应着的 x 路径的计算式如式（15）所示。

$$x_2=x_1-p_1-p_2-p_3-p_4=x_1(1-2^{-11})-y_1(2^{-5}) \quad (15)$$

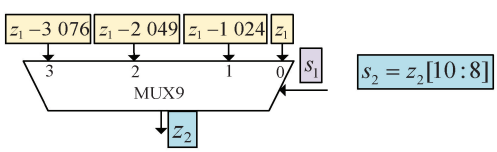
图 6(b)是输出数据 y_2 的逻辑结构实现,与图 6(a)是相似的。因为在 Stage-2 中的旋转角度 3 076 和 2 049 都是选取的近似角度,分别对应角度 3 072($(2^{-4}+2^{-5})\times 2^{15}=3\,072$)和角度 2 048($2^{-4}\times 2^{15}=2\,048$)。角度 3 076 和 2 049 在该阶段并不直接对应角度 z_1 的第 11 和 10 两个 bit 位,故需要使用一个减法器来计算剩余角度 z_2。在 Stage-3 阶段中,使用的旋转角度都是 2^{-i} 或者 $2^{-i}+2^{-i-1}$ 等角度,而非采用近似角度。这些角度能够直接对应角度 z_2 的第 8 到第 10 bit 位,因此该阶段的输出角度选择信号 $s_2[2:0]=z_2[10:8]$。根据 2.1 节中迭代收敛过程,剩余角度 z_2 位于区间[0:1 026]。



(a) 输出数据x_2的计算结构
(a) Computing architecture for output x_2



(b) 输出数据y_2的计算结构
(b) Computing architecture for output y_2



(c) 输出剩余角度z_2和输出选择信号s_2的计算结构
(c) Computing architecture for output z_2 and s_2

图 6　Stage-2 的计算结构

Fig. 6　Computing architecture for stage-2

3) Stage-3 阶段

在 Stage-3 阶段,输入角度 z_2 位于区间[0:1 026]。然而(1 026−768)>255,这超出了后续迭代所支持的输入角度范围[0:255]。因此,根据 2.1 节中迭代收敛过程,该阶段引入了第 4 个旋转角度 1 024,以保证位于区间[0:1 026]的任意输入角度 z_2 经过 Stage-3 的迭代,其输出角度 z_3 能位于区间[0:255],这与 Stage-4 阶段所支持的输入角度匹配。图 7 是输出数据 x_3 的计算结构,与 Stage-2 阶段相似,同样包含 p_1 ~ p_4 等中间输出。输入数据 x_2 与中间输出相加可得输出数据 x_3。

图 7 中的多路选择器 MUX1~MUX3 由选择控制信号 s_2 进行控制。该级所需的 4 个旋转角度如表 2 所示,对应了输出剩余角度 z_2 的第 8 到第 10 bit 位置 $z_2[10:8]$。例如,角度 768 对应 $z_2[9:8]=2'b11$。因此 $z_3=z_2[7:0]$ 可以直接截取 z_2 的低 8 bit 得到,而无需 Stage-2 阶段中的减法操作。输出选择信号 s_3 等于 $z_3[7:4]$。

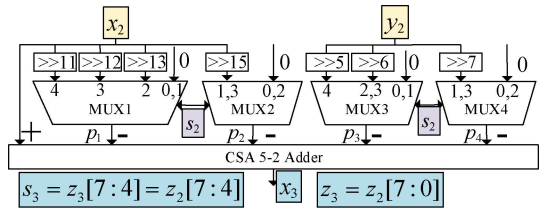


图 7　Stage-3 中输出 x_3、z_3 和 s_3 的计算结构

Fig. 7　Computing architecture for output x_3, z_3 and s_3 in stage-3

4) Stage-4 和 Stage-5 阶段

在 Stage-4 和 Stage-5 阶段,完成对剩余角度 $z_3[7:0]$ 的迭代。在 Stage-4 阶段,旋转角度为 $k_4\cdot 2^{-11}$,而 $k_4=s_3[3:0]=z_3[3:0]$。图 8(a)是 Stage-4 输出数据 x_4 的计算结构。旋转选择信号 $s_3[3:0]$ 的每一个 bit 位控制一个 2 选 1 的数据选择器。整体的旋转角度 $k_4\cdot 2^{-11}$ 便是由图 8(a)中的 4 个数据选择器实现。在图 8(b)是 Stage-5 输出数据 x_5 的计算结构,除了移位器的移位位数,整体结构与 Stage-4 阶段的计算结构一致。在 Stage-5 阶段,旋转角度 $k_5\cdot 2^{-15}$ 将由旋转选择信号 $s_4=z_3[3:0]$ 决定。图 8 仅是输出数据 x_4 和 x_5 的计算结构,而输出数据 y_4 和 y_5 的计算结构与该结构整体一致。

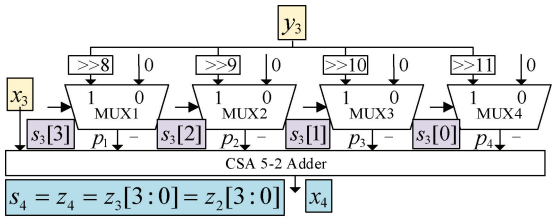
在最后输出免缩放 CORDIC 迭代结果时,$\cos\theta$ 在$(-\pi/2,\pi/2)$区间内始终为正数,所以始终有 $\cos\theta=x_5$。而输出 $\sin(\theta)$ 时受到输入角度 θ 符号 bit 位影响,若符号 bit 位等于“0”,则 $\sin\theta=y_5$;反之,符号 bit 位等于“1”,则 $\sin\theta=-y_5$。

3　实验结果与分析

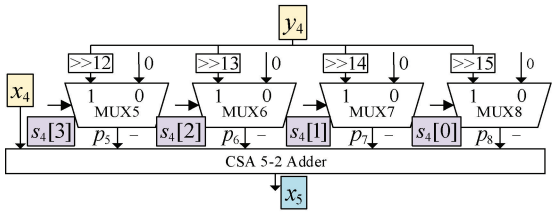
采用硬件描述语言 Verilog 实现所提出混合 CORDIC 算法的计算结构,并采用 AMD Vivado2019.3 和 ISE14.7 软件对其进行实验综合评估。

3.1　误差分析

在 CORDIC 迭代过程中,主要有两个误差来源:1)定点量化的量化误差;2)在移位和加减法操作中的截尾误差。为评估本文设计的误差性能,在区间$[0,\pi/2)$内等间隔采样 2^{15} 个数据并进行 $FIX(1,17,15)$ 的定点量化。将 2^{15} 个数作为输入角度 θ,然后将输出结果 $b(i)$ 与 C 语言软件计算的双精度浮点数结果 $a(i)$ 进行对比,采用误差 bit 位进行表示,计算如式(16)所示。



(a) Stage-4中输出x_4、s_4和z_4的计算结构
(a) Computing architecture for output x_4, s_4 and z_4 in stage-4



(b) Stage-5中输出x_5的计算结构
(b) Computing architecture22 for output x_5 in stage-5

图 8　Stage-4 中输出 x_4 和 stage-5 中输出 x_5 的计算结构

Fig. 8　Computing architecture for output x_4 in stage-4 and x_5 in stage-5

$$bit\_error\_position(i) = -\log_2(b(i) - a(i)) \tag{16}$$

误差结果如图 9 所示,其中最大的误差 bit 位出现在输出结果的第 13 bit 位置处,而最主要的误差 bit 位出现在 14 bit 或者优于 14 bit 位,也就是误差在 $2^{-14} = 6.1 \times 10^{-5}$ 附近。这表明以格式 $FIX(1,16,15)$ 量化的输出结果的后两个 bit 是误差主要占有的 bit 位。

在 CORDIC 中,另一种常使用的误差评价指标是均方根误差(root mean square error, RMSE),其计算方式是将仿真结果 $b(i)$ 与软件双精度浮点数计算结果 $a(i)$ 进行比较,计算表达式如式(17)所示。

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(a(i) - b(i))^2} \tag{17}$$

表 3 为本文设计与现有同类设计在均方根误差指标上的对比,其优于传统 CORDIC 和文献[19],同时与文献[20]的设计相当。

表 3　各类 CORDIC 设计的 RMSE 指标对比

Table 3　RMSE comparison of various CORDIC design

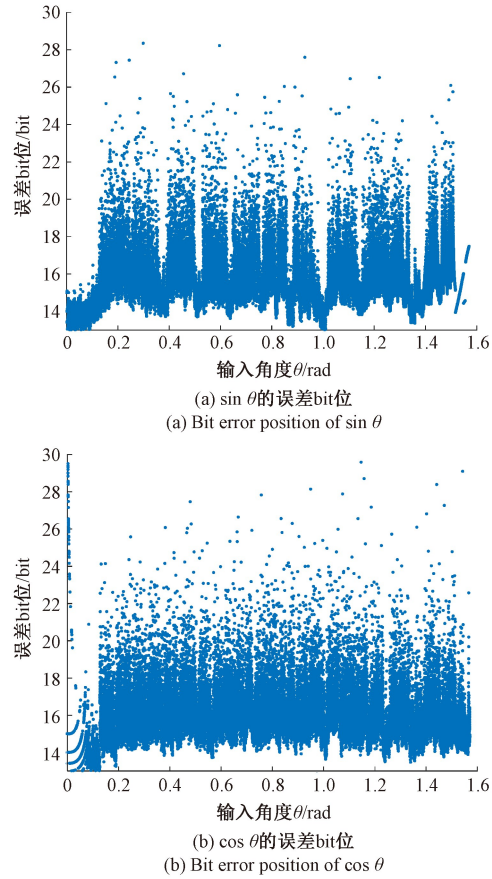
CORDIC Algorithm	sin-RMSE	cos-RMSE
传统 CORDIC	4.90×10^{-5}	4.90×10^{-5}
MRVSF[19]	4.90×10^{-5}	6.10×10^{-5}
NASSF[20]	3.21×10^{-5}	3.12×10^{-5}
本文 CORDIC	2.92×10^{-5}	3.52×10^{-5}

3.2　FPGA 实现的性能分析

为了确保尽可能公平的比较,所提出的设计与现有同类研究工作在相同系列的 FPGA(分别为 Virtex-7 和 Spartan-3E)上实现。对比指标包括资源消耗(LUTs、触发器(flip-flop, FF)或 Slices 资源)、最大工作频率(maximum frequency, Fmax)和延迟(latency)等,其中延迟等于:

$$Latency = Stage/F_{max} \tag{18}$$

表 4 所示结果均基于 Virtex-7 系列的 FPGA 实现,其中传统 CORDIC 和本次提出的设计输入角度均采用 $FIX(1,17,15)$ 量化到 15 bit 小数位,文献[19]采用 $FIX(1,16,14)$ 量化到 14 bit 小数位;在 x 路径和 y 路径均采用 $FIX(1,17,16)$ 进行计算,最后输出时截断最低 1 bit,输出正余弦结果的格式为 $FIX(1,16,15)$。如表 4 所示,与传统 CORDIC 和文献[19]相比,所提出的设计分别减少了 37.2%和 23.1%的资源消耗(LUT+寄存器),并降低了 52.9%和 22.1%的延迟。传统 CORDIC 具有更简单的迭代架构,可实现更高的最大工作频率,但由于收敛速率较慢,迭代次数多,便导致了更高的资源消耗和延迟。文献[19]在免缩放因子迭代旋转之前使用了 4 次传统 CORDIC 迭代,因此相较于本文所提出的设计,其资源消耗和延迟更大。



(a) sin θ的误差bit位
(a) Bit error position of sin θ

(b) cos θ的误差bit位
(b) Bit error position of cos θ

图 9　所提出的免缩放 CORDIC 的误差 bit 位

Fig. 9　The bit error position of the proposed SF-CORDIC

表 4 基于 FPGA Virtex-7 实现结果比较

Table 4 Comparison of implementation results on FPGA Virtex-7

Parameters	传统 CORDIC	MRVSF[19]	本文
LUTs	717	753	640
Registers	688	395	242
Fmax/MHz	350	253	233
Stage	16	7	5
Latency/ns	45.7	27.6	21.5
Range	(−1.73,1.73)	(−π/2,π/2)	(−π/2,π/2)

表 5 所示结果均基于 Spartan-3E 系列的 FPGA 实现,其中文献[13-14]的输入角度格式分别为 *FIX*(1,16,14) 和 *FIX*(1,12,9);在 x 路径和 y 路径的中间计算数据和输出正余弦及结果数据格式均为 *FIX*(1,16,15)。本文提出的设计仍然保持输入角度 *FIX*(1,17,15),中间计算数据 *FIX*(1,17,16) 和输出正余弦结果 *FIX*(1,16,15) 的格式。如表 5 所示,与文献[13-14]相比,所提出的设计分别减少了 54.0% 和 23.2% 的 Slices 资源消耗,并降低了 75.6% 和 25.7% 的延迟。文献[13]采用多个小角度 SF-CORDIC 迭代来实现大角度旋转,而后续迭代仅使用二并行迭代方式,较多的迭代次数导致更高的资源消耗和更大的延迟。文献[14]采用更多的项来逼近大角度旋转,从而增加了资源消耗。此外,这也导致大角度迭代的延迟增加,进而降低了最大工作频率并增加了总延迟。相比于所提出的设计,文献[13-14]支持更大的输入角度范围。

表 5 基于 FPGA Spartan-3E 实现结果比较

Table 5 Comparison of implementation results on FPGA Spartan-3E

Parameters	MVSFA[13]	ATSSF[14]	本文
LUTs	—	—	992
Registers	—	—	227
Slices	1 174	703	540
Fmax/MHz	69	75	101
Stage	14	5	5
Latency/ns	202.9	66.7	49.5
Range	(−π,π)	(−π,π)	(−π/2,π/2)

4 结 论

本文结合查找表和并行免缩放迭代,设计了一种低延迟低资源消耗的混合 CORDIC 算法。所提出的算法利用 LUT 高效快速地将输入的大角度折叠至小角度范围,从而减少迭代次数。在小角度范围内,采用并行迭代进一步降低迭代次数。其中,采用具有更少非零项的近似角度确定了查找表阶段与并行免缩放迭代阶段的角度边界,扩展了并行免缩放迭代所支持的角度收敛范围,并且这些近似角度作为迭代过程中使用的旋转角,进一步减少了资源消耗和每级迭代的延迟。同时,本次设计将并行迭代划分为二并行免缩放迭代和四并行免缩放迭代,每级并行迭代对应的旋转向量(正余弦数值)具有数量相同的非零项,这使得每级并行免缩放迭代的计算复杂度基本相同,可保证所设计 CORDIC 的整体性能。此外,针对于所提出的算法,本文提出了相应的计算结构并基于 FPGA 进行综合实验评估。实验结果表明,与本文同类的设计相比,所提出的设计在保证相同的收敛范围(−π/2,π/2)和几乎一致的精度情况下,资源消耗减少了 23.1%,延迟降低了 22.1%。

在后续研究中,将考虑针对于双曲函数、指数计算和开根计算等更为复杂计算的高效 CORDIC 设计实现。而对于本文设计所使用的进位保留加法器 CSA 可以考虑采用更加先进的加法器以实现更低延迟低资源消耗的高效 CORDIC。

参考文献

[1] 赵禹, 叶芃, 孟婕, 等. 基于带宽交织采样架构的 80 GSps 超宽带实时采集系统[J]. 仪器仪表学报, 2024, 45(5): 147-156.
ZHAO Y, YE P, MENG J, et al. Ultra-wideband data acquisition system with 80 GSps based on bandwidth interleaved architecture[J]. Chinese Journal of Scientific Instrument, 2024, 45(5): 147-156.

[2] QIAO Y, PENG Y, LIU L, et al. A multi-frequency phase difference estimation algorithm with time-frequency analysis[C]. 2024 IEEE International Instrumentation and Measurement Technology Conference (I2MTC). 2024: 1-6.

[3] 李沛峰, 刘丽, 王宇, 等. 基于 FPGA 的 Φ-OTDR 系统数字正交解调方法[J]. 电子测量与仪器学报, 2024, 38(5): 19-28.
LI P F, LIU L, WANG Y, et al. Φ-OTDR system digital quadrature demodulation method based on FPGA[J]. Journal of Electronic Measurement and Instrumentation, 2024, 38(5): 19-28.

[4] 马峻, 毛露露, 陈宏, 等. 数字干涉图实时相位提取 PSM-IP 核的 FPGA 实现[J]. 电子测量与仪器学报, 2019, 33(12): 71-79.
MA J, MAO L L, CHEN H, et al. FPGA implementation of real-time phase extraction PSM-IP core for digital interferograms[J]. Journal of Electronic Measurement and Instrumentation, 2019, 33(12): 71-79.

[5] 李润泽, 王涛, 牛群峰, 等. 基于误差分析与改进 CORDIC 算法的反射式光栅位移测量方法[J]. 仪器仪表学报, 2024, 45(11): 215-223.
LI R Z, WANG T, NIU Q F, et al. A reflective grating

displacement measurement method based on error analysis and the improved CORDIC algorithm [J]. Chinese Journal of Scientific Instrument, 2024, 45(11): 215-223.

[6] DATTA D, DUTTA H S. Design and implementation of digital down converter for WiFi network [J]. IEEE Embedded Systems Letters, 2024, 16(2): 122-125.

[7] 赵创, 张为. 基于 HCORDIC 的浮点运算协处理器的设计[J]. 电子测量与仪器学报, 2020, 34(11): 58-65.
ZHAO CH, ZHANG W. Design of floating pointarithmetic coprocessor based on HCORDIC [J]. Journal of Electronic Measurement and Instrumentation, 2020, 34(11): 58-65.

[8] CHEN H, QUAN L, LIU W. HGH-CORDIC: A high-radix generalized hyperbolic coordinate rotation digital computer[C]. 2024 IEEE 31st Symposium on Computer Arithmetic (ARITH). 2024: 88-95.

[9] LI K, FANG H, MA Z, et al. A low-latency CORDIC algorithm based on pre-rotation and its application on computation of arctangent function [J]. Electronics, 2024, 13(12): 2338.

[10] ZHU B, LEI Y, PENG Y, et al. Low latency and low error floating-point sine/cosine function based TCORDIC algorithm [J]. IEEE Transactions on Circuits and Systems I: Regular Papers, 2017, 64(4): 892-905.

[11] CHANGELA A, ZAVERI M, LAKHLANI A. ASIC implementation of high performance Radix-8 CORDIC algorithm [C]. 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI). Bangalore: IEEE, 2018: 699-705.

[12] CHANDRA INGUVA S, SEVENTILINE J B. Implementation of FPGA design of FFT architecture based on CORDIC algorithm [J]. International Journal of Electronics, 2021, 108(11): 1914-1939.

[13] MAHARATNA K, BANERJEE S, GRASS E, et al. Modified virtually scaling-free adaptive CORDIC rotator algorithm and architecture [J]. IEEE Transactions on Circuits and Systems for Video Technology, 2005, 15(11): 1463-1474.

[14] WADKAR S S, DAS B P, MEHER P K. Low latency scaling-free pipeline CORDIC architecture using augmented taylor series [C]. 2019 IEEE International Symposium on Smart Electronic Systems (iSES) (Formerly iNiS). Rourkela, India: IEEE, 2019: 312-315.

[15] VERMA A, KIYAWAT K, DAS B P, et al. An Efficient scaling-free folded hyperbolic CORDIC design using a novel low-complexity power-of-2 taylor series approximation [J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2023, 31(8): 1167-1177.

[16] MOROZ L, NAGAYAMA S, MYKYTIV T, et al. Simple hybrid scaling-free CORDIC solution for FPGAs [J]. International Journal of Reconfigurable Computing, 2014, 2014.

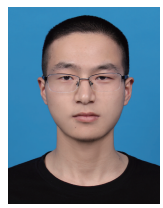
[17] WU C, ZHOU B, FANG J. Research on improved CORDIC algorithm for high accuracy DDS signal source [C]. 2015 4th International Conference on Computer Science and Network Technology (ICCSNT). 2015: 1351-1355.

[18] XUE Y, MA Z. Design and implementation of an efficient modified CORDIC algorithm[C]. 2019 IEEE 4th International Conference on Signal and Image Processing (ICSIP). 2019: 480-484.

[19] CHANGELA A, ZAVERI M, VERMA D. Mixed-radix, virtually scaling-free CORDIC algorithm based rotator for DSP applications[J]. Integration, 2021, 78: 70-83.

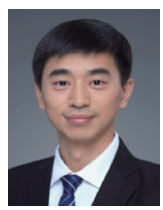
[20] CHANGELA A, ZAVERI M, KUMAR Y. A new angle set-based absolute scaling-free reconfigurable cordic algorithm[J]. Circuits, Systems, and Signal Processing, 2023, 42(12): 7404-7432.

作者简介



魏学静,2022 年于哈尔滨工业大学获得学士学位,现为哈尔滨工业大学硕士研究生,主要研究方向为 FFT 算法及其数字电路的实现与优化。
E-mail: wxj_hit006@163.com

Wei Xuejing received his B. Sc. degree from Harbin Institute of Technology in 2022. He is currently a M. Sc. candidate from Harbin Institute of Technology. His main research interests include implementation and optimization of the FFT algorithm and its digital circuit.



刘连胜(通信作者),2006 年、2008 年和 2017 年于哈尔滨工业大学分别获得学士、硕士和博士学位,现为哈尔滨工业大学教授,主要研究方向为电子测量、基于 FPGA 的高能效计算技术、故障预测与健康管理等。
E-mail: lianshengliu@hit.edu.cn

Liu Liansheng (Corresponding author) received his B. Sc., M. Sc., and Ph. D. degrees all from Harbin Institute of Technology in 2006, 2008, and 2017, respectively. He is currently a professor and a Ph. D. advisor at Harbin Institute of Technology. His main research interests include electronic measurement, FPGA-based energy-efficient computing technology, fault prognostics and health management, etc.