

工业控制领域

Cordic算法

在现代工业控制应用场景中存在大量坐标变换和超越函数的计算，Cordic算法凭借其硬件友好的特性，成为工业控制领域实现高效实时运算的基石算法。但是在通用处理器中实现Cordic算法面临着性能瓶颈问题，本文采用隼瞻科技ArchitStudio™工具，为Cordic算法定制RISC-V拓展指令实现硬件加速，最终在少量硬件开销的前提下获得27倍的性能收益，为领域专用的处理器架构的快速探索提供了可能性。

一、背景介绍

在工业自动化的核心场景中，如永磁同步电机（PMSM）的矢量控制（FOC）、工业机器人的实时轨迹规划、并网逆变器的高精度锁相以及多轴伺服驱动系统，都需要在微秒级的时间内完成大量三角函数、反三角函数及坐标变换运算。这些运算的实时性与精度直接决定了设备的能效、响应速度与控制稳定性。

传统通用处理器在处理这类超越函数时往往依赖软件数学库，其延迟与功耗难以满足高性能工业控制的需求。而CORDIC（坐标旋转数字计算机）算法凭借其独特的“移位-相加”迭代结构，无需硬件乘法器即可高精度计算三角函数、双曲函数、对数及开方等，成为工业控制领域实现高效实时运算的基石性算法。其硬件友好的特性，使得它能够以极低的资源开销，在FPGA、ASIC及专用微控制器中实现，为工业设备的小型化、高效化与高可靠性提供了关键支撑。

二、问题与挑战

CORDIC算法的核心流程简洁而优雅。以计算余弦和正弦值的“旋转模式”为例，其预先定义一系列微小角度 $\theta_i = \arctan(2^{-i})$ ，通过旋转迭代的方式逼近目标角度 θ 。每次迭代仅需要进行以下操作：

- 根据当前剩余角度 z_i 的符号，决定旋转方向 $d_i = \text{sign}(Z_i)$
- 执行坐标更新： $X_{i+1} = X_i - d_i * (y_i * 2^{-i})$ $Y_{i+1} = Y_i + d_i * (X_i * 2^{-i})$
- 更新剩余角度： $Z_{i+1} = Z_i - d_i * \theta_i$

CORDIC算法核心代码如下：

```
1 int32_t cordic(int32_t theta_fixed) {
2     int32_t x = FIXED_K, y = 0, z = theta_fixed;
3     for (int i = 0; i < ITERATIONS; ++i) {
4         int32_t d = (z >= 0) ? 1 : -1;
5         int32_t x_new = x - d * (y >> i);
6         int32_t y_new = y + d * (x >> i);
7         int32_t z_new = z - d * cordic_table[i];
8         x = x_new;
9         y = y_new;
10        z = z_new;
11    }
12    return x;
13 }
```