面向64位RISC-V的基础数学库自动化移植

Automatic Porting of Basic Mathematics Library for 64-bit RISC-V

数学工程与先进计算国家重点实验室

汇报人：曹浩

邮箱：atao728208@126.com

OUTLINE

提纲

00 引入

01 成果简介

02 设计与实现

03 移植函数

04 总结与展望

基础数学库作为处理器配套软件的重要组成部分，是高性能计算机平台上科学计算、工程数值计算和专用软件开发所必备的最基础、最核心的软件之一。

Performance Libraries

USA (English) Sign In

/ Tools

Intel® Math Kernel Library

AMD | Developer Central

China / AMD.com SEARCH

HOME | TOOLS & SDKS | RESOURCES | COMMUNITY | ECOSYSTEM

Home > **Building with ACML**

CMSIS-DSP Version 1.8.0

CMSIS DSP Software Library

General | Core(A) | Core(M) | Driver | DSP | NN | RTOS v1 | RTOS v2 | Pack | Build | SVD | DAP | Zone

Main Page | Usage and Description | Reference

Search

CMSIS-DSP

Data Structures | Macros | Typedefs | Enumerations | Functions

arm_math.h File Reference

The GNU C Library (glibc)

What is glibc? | Get Started | Get Involved | Sources | Documentation | Report a bug

What is glibc?

The GNU C Library project provides *the* core libraries for the GNU system and GNU/Linux systems, as well as many other systems that use Linux as the kernel. These libraries provide critical APIs including ISO C11, POSIX.1-2008, BSD, OS-specific APIs and more. These APIs include such foundational facilities as `open`, `read`, `write`, `malloc`, `printf`, `getaddrinfo`, `dlopen`, `pthread_create`, `crypt`, `login`, `exit` and more.

基于申威平台的国产基础数学库

总体介绍

通用函数库

IEEE 754 标准

基础函数库 | SIMD扩展函数库 | Fortran函数库

三角函数 | 双曲函数 | 指数函数 | 对数函数 | 数值运算函数 | 数值处理函数 | 判断类函数 | 贝塞儿函数 | 误差函数

浮点异常控制 | 精度控制 | 性能优化

基础算法 | SIMD算法

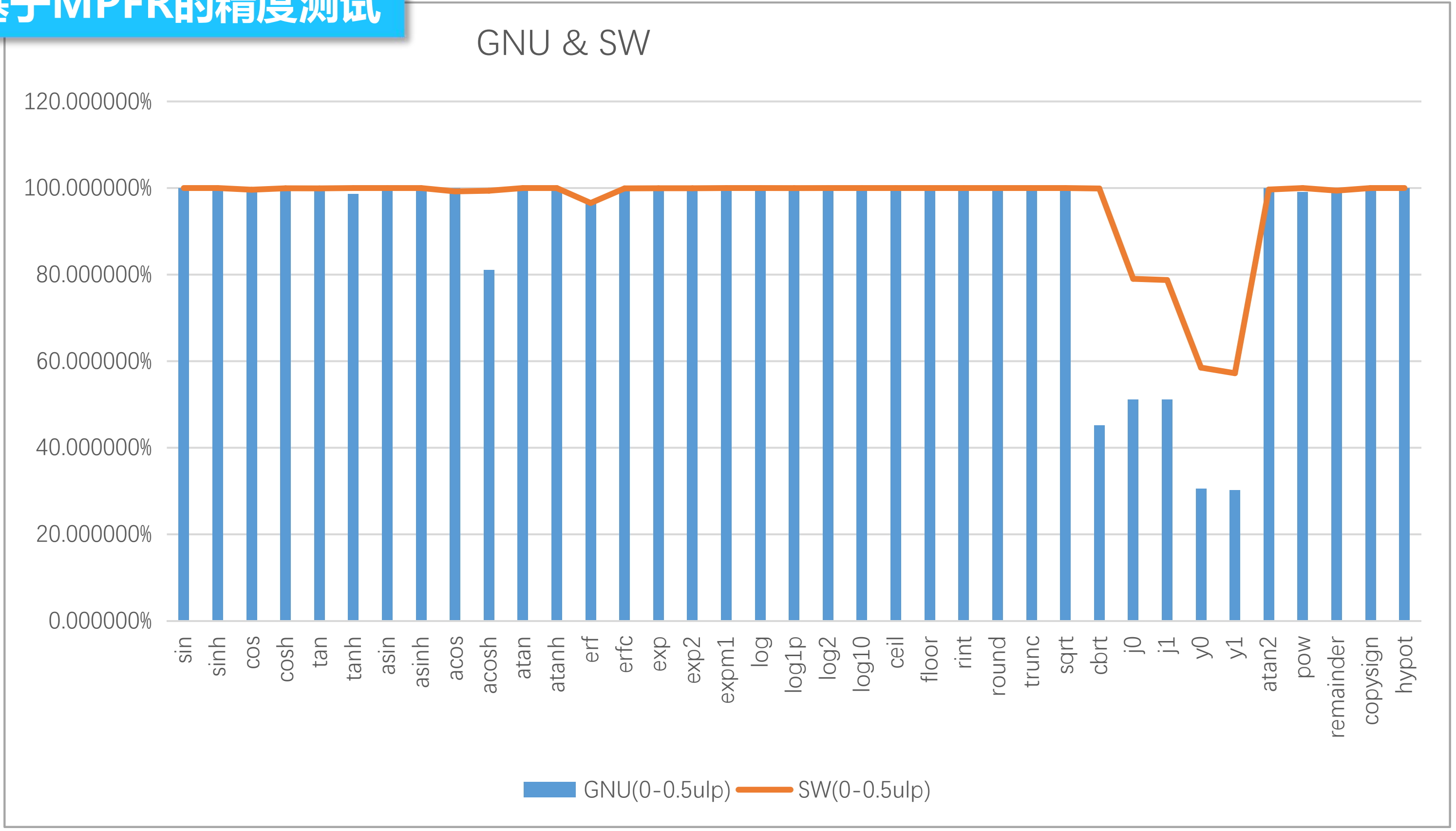
ISO C99 函数接口规范

基于申威平台的国产基础数学库

精度方面

基础数学函数精度与GNU相比平均提高5.02%

基于MPFR的精度测试

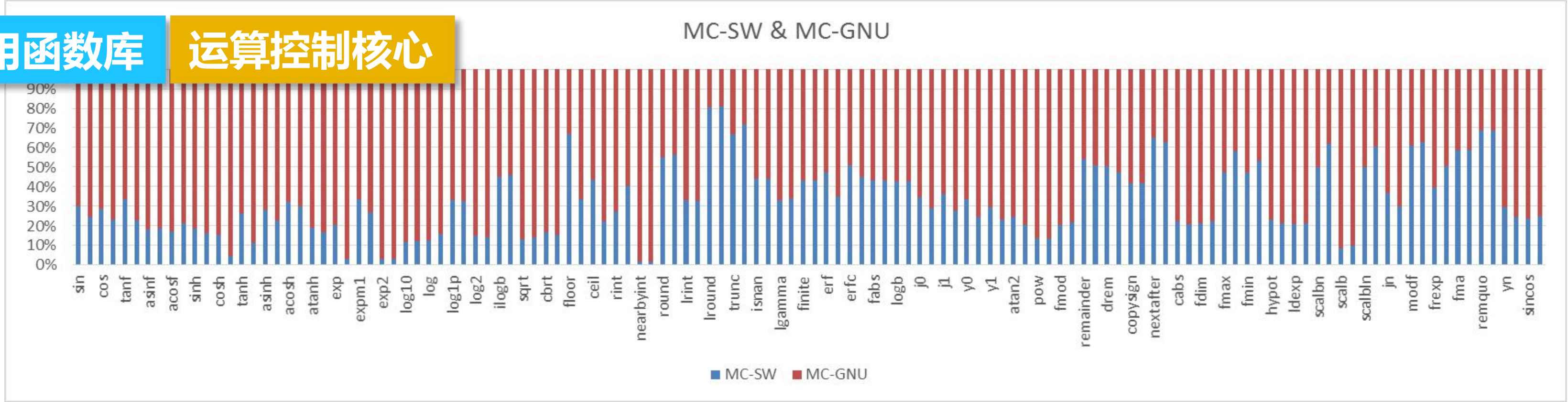


基于申威平台的国产基础数学库

性能方面

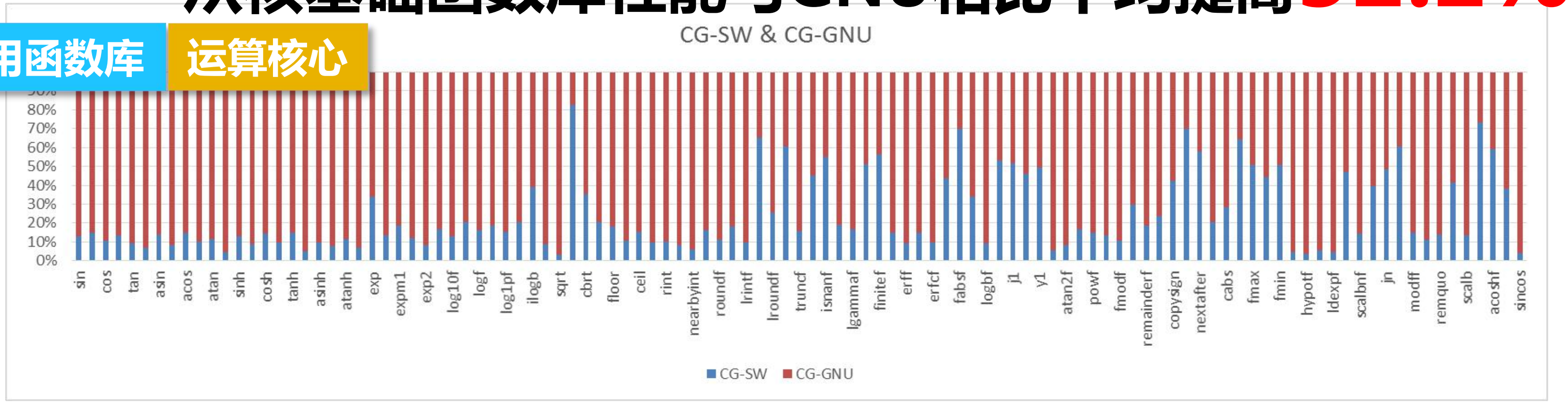
主核基础函数库性能与GNU相比平均提高33.6%

通用函数库 运算控制核心



从核基础函数库性能与GNU相比平均提高52.2%

通用函数库 运算核心



基于申威平台的国产基础数学库

课题应用

清华大学

1、WRF模式云微物理过程。核心代码段使用LDM版本快速数学库，一个迭代步的节拍数从560万下降到170万，提高3倍。

2、某应用课题。与用户深入沟通应用和函数对LDM使用的权衡方案，针对关键函数分别设计了1KB、2KB、4KB多个版本的算法实现供用户选择。

3、某应用课题。针对核心代码循环中反复密集调用函数的情况，定制开发了动态LDM版本，确保函数的LDM使用不会对应用造成影响，且性能与高效的静态LDM版本相当。

西电

4、某应用课题。针对应用需求，开发支持复数的Fortran函数，为用户提供规范接口和高效的函数实现。

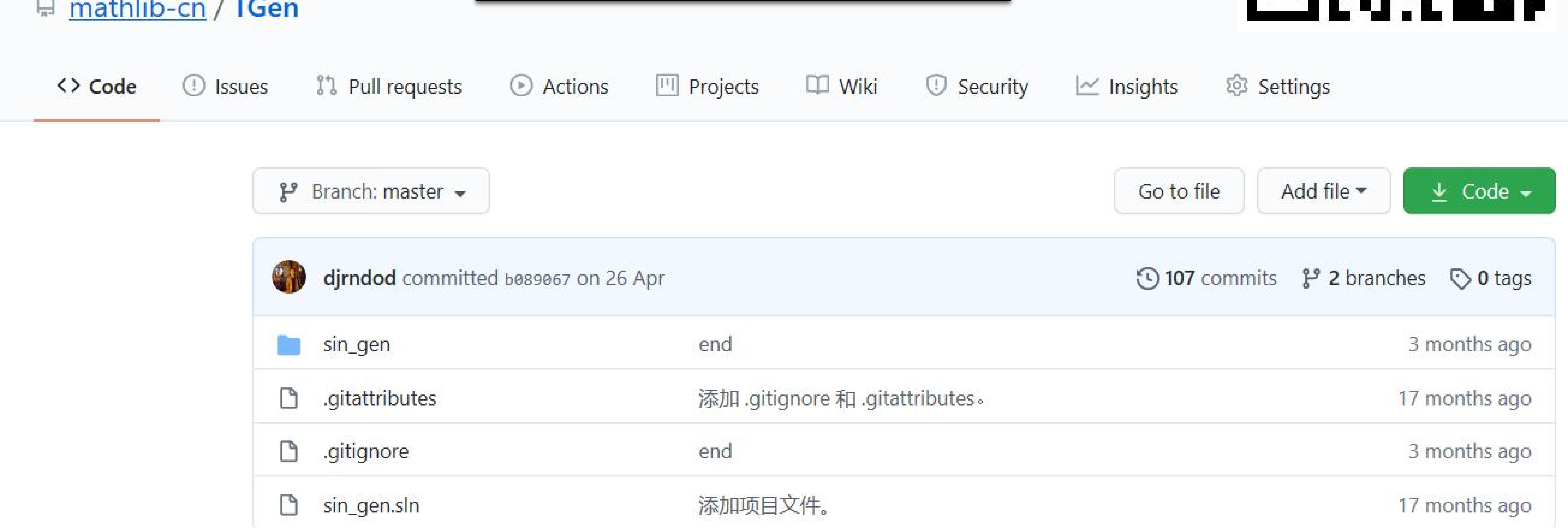
过程所

5、纳米线热导率的大规模分子动力学模拟。针对应用课题对函数执行效率和精度的特殊需求，重新设计了支持细粒度精度优化的函数算法和局部流水调度技术，开发了适当降低精度（误差<=3ULP）的高效定制版本，与过程所联合申报GB2016。

基于申威平台的国产基础数学库

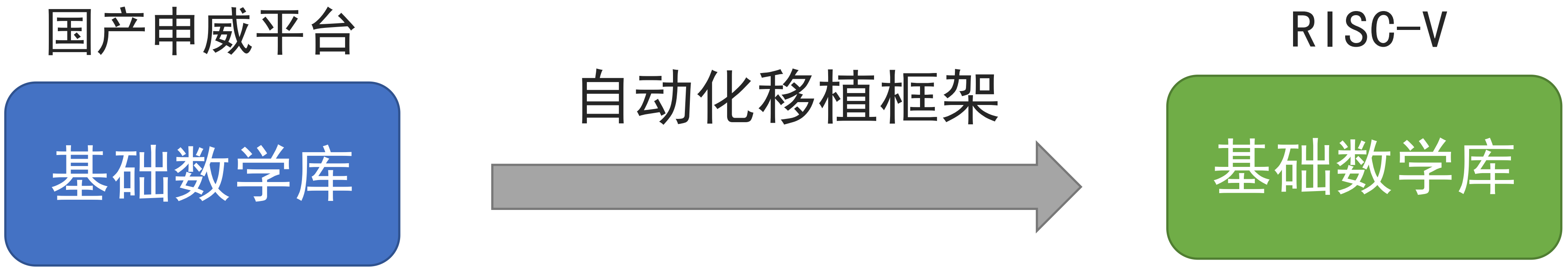
TGen

https://github.com/mathlib-cn/TGen



01 加强RISC-V生态建设

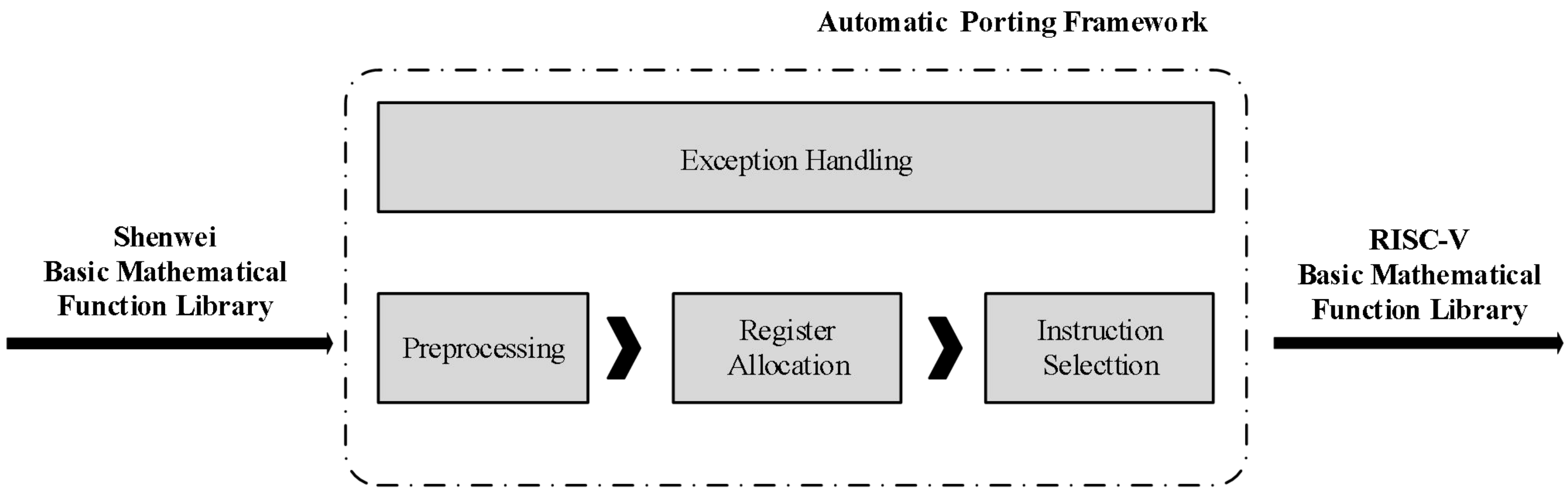
02 准确高效移植数学库



1 成 果 简 介

成果简介

01 设计了一个自动化移植框架，使得申威基础数学库通过移植平台就能实现在64位RISC-V环境上运行

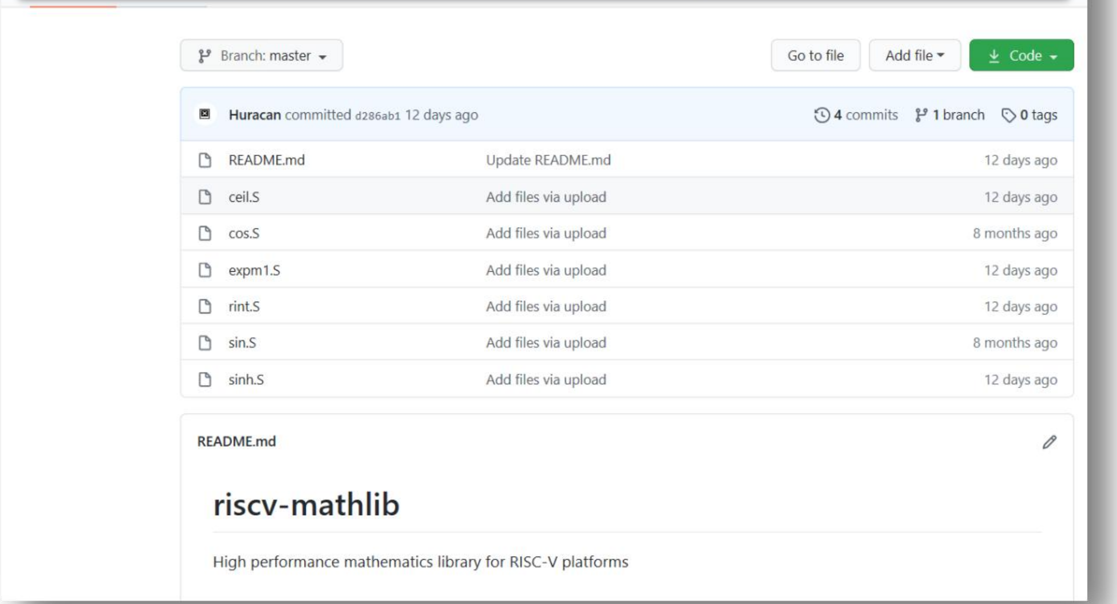


成果简介

02

完成了对申威基础数学库典型超越函数和数值计算函数的移植，移植函数功能正确，且相较于GLIBC性能有所提升

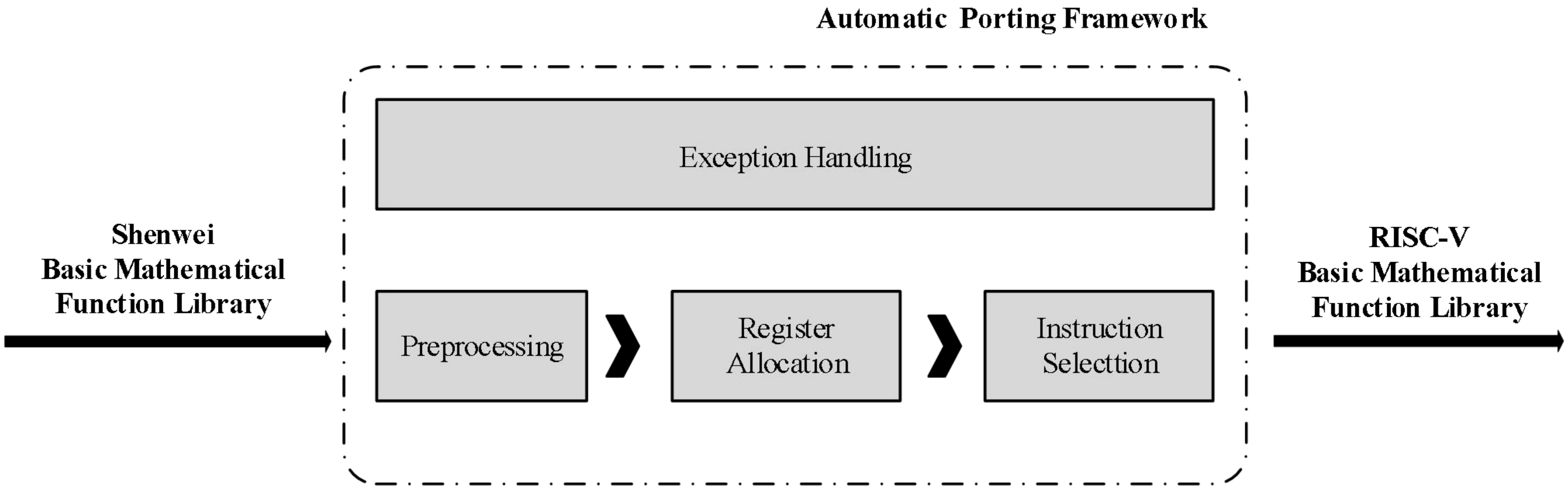
https://github.com/mathlib-cn/riscv-mathlib



2 设 计 与 实 现

设计与实现

自动化移植框架的组成



设计与实现

移植工作流程示例

Code#1

```
# Shenwei Mathlib
# sin
$sindata:
        .dword 0x3fe45f306dc9c883
        .dword 0x42c8000000000000
sin:
        fmv.x.d a0,fa0
        ldi a0,$sindata
        ...
```

Preprocessing

Code#2

```
$sindata:
        .quad 0x3fe45f306dc9c883
        .quad 0x42c8000000000000
sin:
        fmv.x.d a0,fa0
        ldi a0,$sindata
        ...
```

Register Allocation

Code#3

```
$sindata:
        .quad 0x3fe45f306dc9c883
        .quad 0x42c8000000000000
sin:
        fmv.x.d v0,$f16
        ldi a1,$sindata
        ...
```

Instruction Selecttion

Code#4

```
$sindata:
        .quad 0x3fe45f306dc9c883
        .quad 0x42c8000000000000
sin:
        fimovd $f16,v0
        lui a1,%hi($sindata)
        addi a1,a1,%lo($sindata)
        ...
```

寄存器分配模块

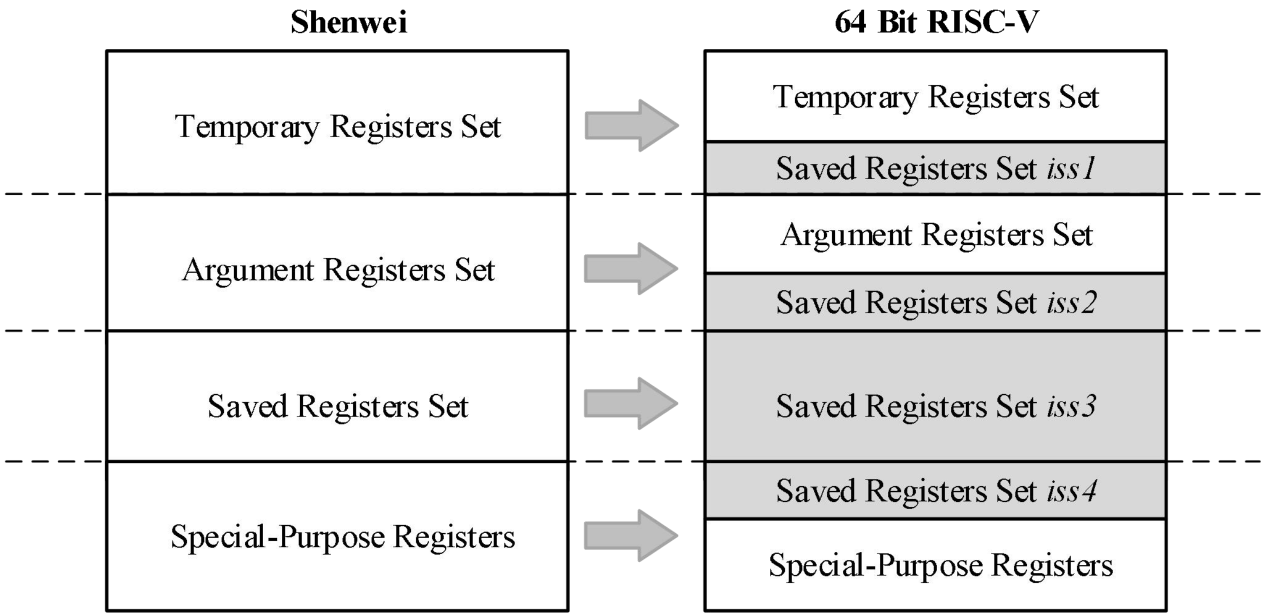
基于全局的主动式寄存器分配方法（Global Active Register Allocation Method,GARA Method）

指令选择模块

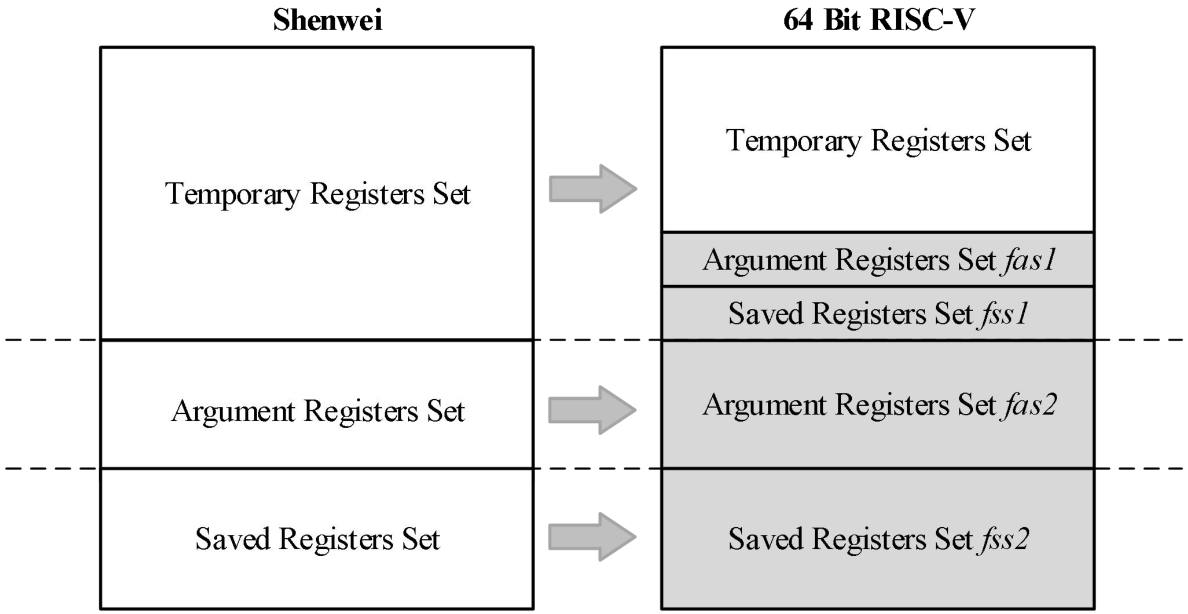
基于层次的指令选择策略（Hierarchical Instruction Selection Strategy，HIS Strategy）

设计与实现

寄存器分配模块



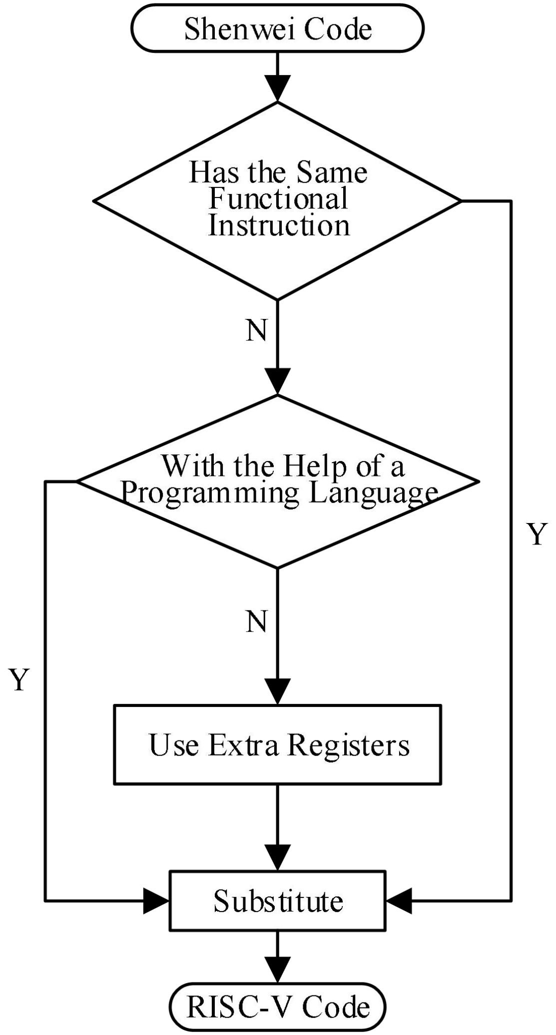
整数寄存器分配示意图



浮点寄存器分配示意图

设计与实现

指令选择模块




```
# PYTHON
```

```
# Shenwei
…
ldi t12,17666(t12)
…
```

```
# RISC-V
…
sd s8,80(sp)
li s8,17666
add s11,s11,s8
ld s8,80(sp)
…
```

```
        else: return False
```

3 移 植 函 数

移植函数

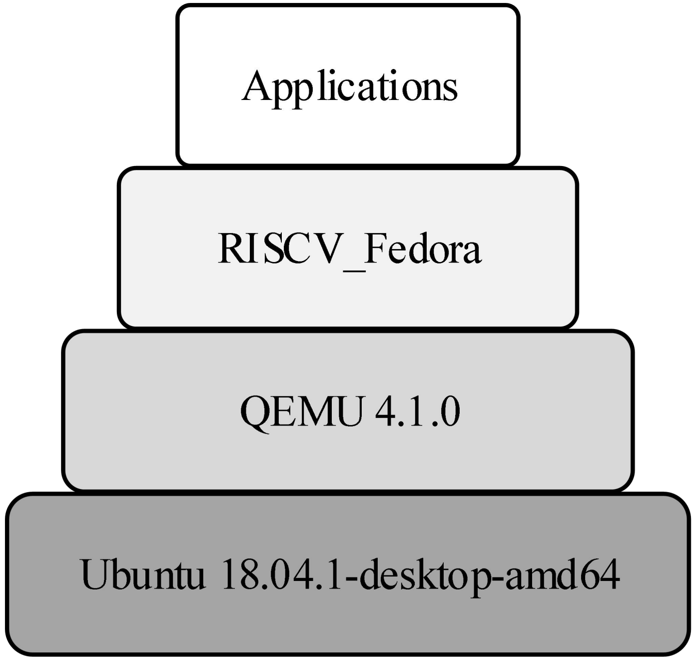
典型函数的选取

Code Type \ Function	Basic Operations											Extended Operations				
	Arithmetic	Logical	Shift	Compare	Convert	Byte operations	Sign Copy	Register transfer	MAD/FMAD	Conditional Selection	R/W FPCR	Branh	Load/Store	System Call	Privileged	Miscellaneous
sin	○	○	○	○	○	×	○	○	○	○	×	○	○	×	×	×
cos	○	○	○	○	○	×	○	○	○	○	×	○	○	×	×	×
sinh	○	○	○	○	×	×	×	○	○	○	×	○	○	×	×	×
expm1	○	○	○	○	×	×	×	○	○	○	×	○	○	×	×	×
rint	○	○	○	○	×	×	○	○	×	×	×	○	○	×	×	×
ceil	○	○	○	○	×	×	○	○	×	×	×	○	○	×	×	×

移植函数

测试样本&测试环境

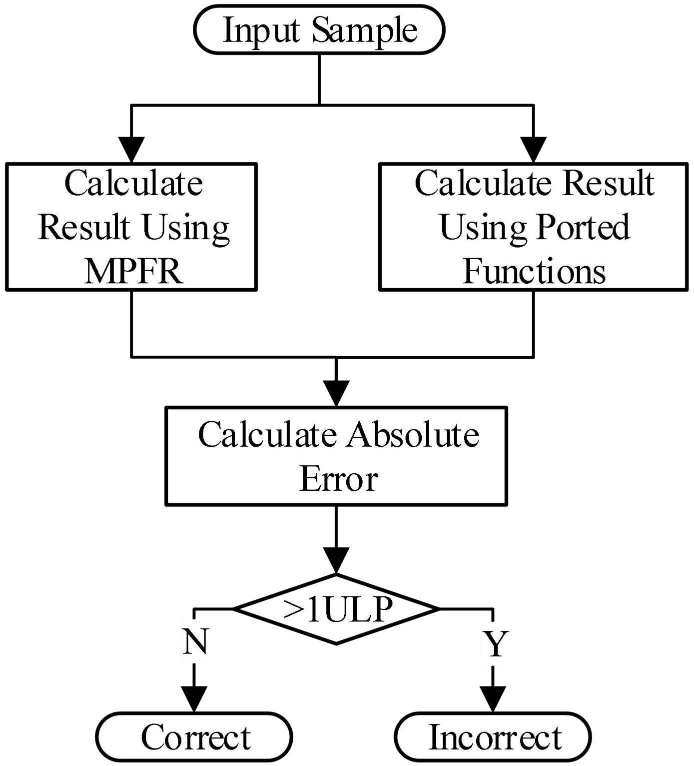
测试样本总共包含8,380,416个64位双精度浮点数，其浮点指数能够全部覆盖，最大限度地保证了测试浮点数的覆盖率。



测试环境

移植函数

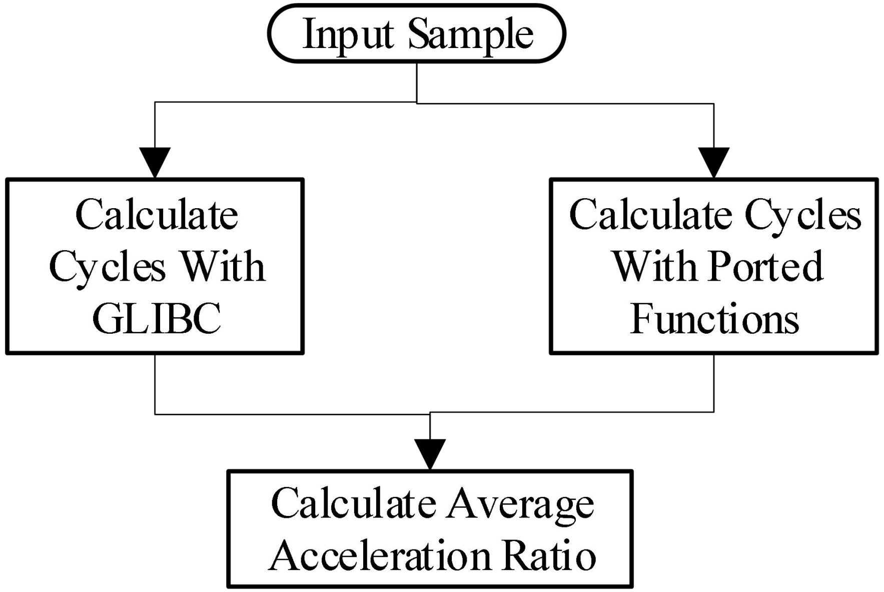
正确性测试



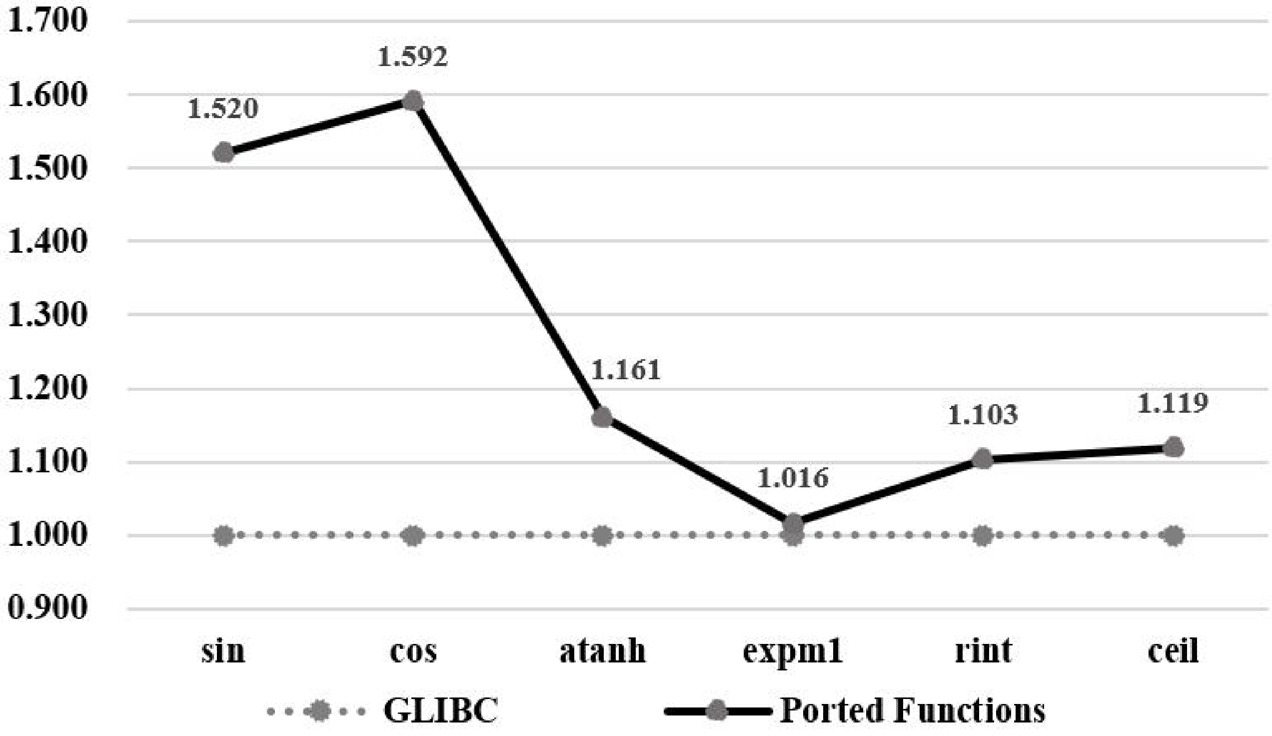
ULP Functions	=1.0	<1.0	>1.0
sin	0.55%	99.45%	0.00%
cos	0.55%	99.45%	0.00%
sinh	0.00%	100.00%	0.00%
expm1	0.00%	100.00%	0.00%
rint	0.00%	100.00%	0.00%
ceil	0.00%	100.00%	0.00%

移植函数

性能测试



Input Sample
Calculate Cycles With GLIBC
Calculate Cycles With Ported Functions
Calculate Average Acceleration Ratio



1.700
1.600
1.500
1.400
1.300
1.200
1.100
1.000
0.900
1.520
1.592
1.161
1.016
1.103
1.119
sin
cos
atanh
expm1
rint
ceil
GLIBC
Ported Functions

4 总 结 与 展 望

总结与展望

1

当前只完成了典型函数的移植，验证了移植的可行性

下一步，将通过自动化框架实现整个基础数学库的移植

2

移植函数性能仍有提升空间

下一步，计划在自动化框架中加入指令调度模块

3

当前工作主要在模拟环境中进行

下一步，当相关条件具备的时侯，研究在真实硬件环境下的验证实验

感　谢　各　位　专　家

敬　请　批　评　指　正