

2023 全国大学生计算机系统能力大赛

编译系统挑战赛代码规模优化赛题技术方案

详情访问大赛技术平台：<https://compiler.educg.net>

一、评价方式的基本说明

第 1 条 大赛编译系统挑战赛代码规模（Code Size）优化赛题（以下简称“赛题”）

要求参赛队在确保程序功能正确且性能损失合理的前提下，综合使用汇编及链接时代码规模优化技术，尽可能减小面向特定目标平台的二进制代码规模。

第 2 条 参赛队必须完成面向 RISC-V 32 位平台基于 Global Pointer（GP）寄存器的代码规模优化，在此基础上，可根据需要综合采用各类汇编及链接时的代码规模优化技术，提交改进后的汇编器、链接器源代码，实现对给定汇编文件的代码规模优化能力。

第 3 条 除本技术方案特别要求、规定和禁止事项外，各参赛队可在给定的参考实现基础上，自行决定所采用的优化技术等细节。

二、初赛评分标准

第 4 条 比赛内容。在给定的参考实现源码基础上，开发面向 RISV-V 32 位架构的汇编器和链接器，在确保程序功能正确且性能损失合理的前提下，获得尽可能小的.text 代码段的二进制可执行文件。

1. 比赛提供 OpenEuler LLVM15 编译器源码（包括汇编器和链接器）作为参考实现，能够在运行 Ubuntu 18.04 操作系统的 Docker 容器上基于 GCC 编译运行。
2. 参赛队改进的链接器采用静态链接方式，允许参赛队针对大赛提供的库进行通用汇编、链接优化，不允许参赛队通过手写汇编或直接替换相似功能库等非汇编、链接的优化方式进行代码规模优化。
3. 基于参考实现源码及参赛队改进的汇编器、链接器源码得到的编译器，应能以面向 RISV-V 32 位架构的汇编文件作为输入，在确保程序功能正确且性能损失合理的前提下，经过代码规模优化后，输出面向目标 RISC-V 32 位架构的，.text 代码段的代码规模尽可能小的二进制可执行文件。

第 5 条 功能测试。基于参赛队提交的汇编器、链接器源码得到的编译器应能够对赛题提供的 RISC-V 32 位架构汇编格式的基准测试程序进行汇编和链接，正确生成二进制可执行文件，并在指定目标平台的模拟器中运行，根据每个汇编格式的基准测试程序所给出的输入数据，对比输出结果，计算功能正确性得分。若未能将每个基准测试程序正确汇编、链接输出可执行二进制文件，或所有测试点均未通过计 0 分；所有测试点都通过计 100 分；部分测试点通过的，按所通过测试点的比例计算功能测试得分。参赛队的最终功能测试成绩为每个基准测试程序功能测试分数的平均值。

第 6 条 性能测试。在通过功能测试的前提下，记录针对每个基准测试汇编、链接得到的可执行二进制文件在目标硬件平台的模拟器上运行的执行时间 T_{opt} ，作为性能评价依据；使用开源编译器工具链对每个基准测试程序进行汇编和链接并记录在目标硬件平台模拟器上运行的执行时间 T_{base} ，作为性能评价的基准；计算参赛队在每个基准测试程序上的性能分数 S_{perf} ，具体计分规则如下：

$$S_{perf} = \begin{cases} -10, & T_{base} < T_{opt} \times 80\% \\ 0, & T_{opt} \times 80\% \leq T_{base} \leq T_{opt} \\ 5, & T_{base} > T_{opt} \end{cases}$$

参赛队的最终性能测试成绩为每个基准测试程序的性能分数的平均值。

第 7 条 代码规模测试。在通过功能测试的前提下，用参赛队汇编器、链接器对每个基准测试程序汇编、链接得到的可执行二进制文件进行分析，统计可执行二进制代码段（.text）的代码规模 C_{opt} ，作为代码规模的评价依据；使用开源编译器工具链对每个基准测试程序进行汇编和链接得到的可执行二进制文件进行分析，统计可执行二进制代码段（.text）的代码规模 C_{base} ，作为代码规模评价的基准；计算参赛队在每个基准测试程序上的代码规模分数 S_{size} ，具体计分规则如下：

$$S_{size} = \frac{C_{base} - C_{opt}}{C_{base}} \times 500$$

参赛队的最终代码规模测试成绩为每个基准测试程序的代码规模测试分数的平均值。

第 8 条 本赛题分为初赛和决赛。初赛阶段共提供 3 组公开的 RISC-V 32 位的汇编基准测试程序。初赛成绩按如下权重计算：

1. 基准测试程序的功能测试成绩：50%
2. 基准测试程序的性能测试及代码规模测试成绩之和：50%

三、决赛评分标准

第 9 条 决赛阶段赛题要求和初赛保持一致，参赛队团队协作及现场答辩主要考察参赛队的团队协作及提交作品的代码完成度、技术创新性及代码规范性等因素，具体评分标准在决赛阶段另行公布。

第 10 条 决赛阶段追加 3 组不公开的测试程序，连同初赛阶段公开的 3 组基准测试程序，共 6 组测试程序，用于对参赛队的成绩评定。

第 11 条 决赛成绩按如下权重计算：

1. 基准测试程序的功能测试成绩：10%
2. 基准测试程序的性能测试及代码规模测试成绩之和：50%
3. 参赛队团队协作及现场答辩：40%

四、参赛作品提交

第 12 条 各参赛队在初赛和决赛阶段需要在大赛的竞赛平台提交如下内容：

1. 基于参考实现源码改进汇编器、链接器后的完整工程文件，并在竞赛平台中至少有一次完整通过性能测试和代码规模测试的记录和有效成绩。
2. 相关设计文档（PDF 格式），内容包括但不限于对功能、性能及创新性的说明、相关测试结果及与类似项目的对比分析、遇到的瓶颈问题和解决方法等。

第 13 条 参赛队在初赛阶段需要提供时长不超过 5 分钟的答辩视频（MP4 格式）。

第 14 条 如果需要使用第三方 IP 或者借鉴他人的部分源码，必须在设计文档和源代码的头部予以明确说明，并确保内容符合相关法律法规和开源协议之规定。允许参赛队部分代码参考或使用遵循开源协议的编译系统或软件模块。

第 15 条 参赛队必须严守学术诚信。一经发现存在代码抄袭或使用他人代码不做说明等学术不端行为，修改的代码重复率在 20% 以上，取消参赛队的参赛资格。

第 16 条 参赛队在赛后，按编译系统设计赛要求公开代码和文档。

五、竞赛平台与测试程序

第 17 条 赛题提供的竞赛平台和测试程序包括：

1. 代码托管平台，支持各参赛队的群体协作与版本控制。
2. 竞赛评测系统，根据参赛队的申请从代码托管平台获取指定版本，生成编译系统，并加载基准测试程序，自动进行功能、性能及代码规模测试。
3. 汇编语言基准测试程序，用于在 RISC-V 32 位目标硬件平台上对参赛队优化后的编译系统所生成的可执行文件进行评测。

六、软硬件系统规范

第 18 条 赛题使用华为公司提供的 OpenEuler LLVM15 编译器源码作为参考实现。参赛队基于该版本的 RISC-V 32 位后端进行优化开发，可使用扩展指令集“i”、“m”、“c”、“f”、“d”。

编译器源码路径：<https://gitee.com/openeuler/llvm-project> 分支 dev_15x

第 19 条 大赛指定的编译环境用于编译参赛队提交的源码，参数如下：

1. CPU：Intel x86，24 物理核心、开启超线程。
2. 内存：128GB LPDDR4-3200 SDRAM。
3. Docker 容器操作系统：Ubuntu 18.04。
4. 建议构建 LLVM 工具链使用标准 GCC 编译器，编译选项如下：
 - 1) `mkdir -p /home/compiler/llvm`
 - 2) `cmake -S llvm -B build -G "Unix Makefiles" -
DLLVM_ENABLE_PROJECTS="clang;lld" -
DLLVM_TARGETS_TO_BUILD="RISCV" -
DCMAKE_INSTALL_PREFIX="/home/compiler/llvm" -
DCMAKE_BUILD_TYPE="release"`
 - 3) `cmake --build build --target install -j16`

第 20 条 大赛指定的 RISC-V 32 位架构目标程序基准测试设备为 RISC-V 32 位仿真器（QEMU），主要参数如下：

1. 可运行 32 位 RISC-V 标准程序，支持“i”、“m”、“c”、“f”、“d”扩展

指令集。

2. 运行仿真器的操作系统版本为 Ubuntu18.04。

七、大赛网站

第 21 条 编译系统赛技术平台网址：<https://compiler.educg.net>。

第 22 条 大赛网站提供多种软件开发工具及设计资料，包括但不限于下列内容：

1. LLVM 工具链编译说明。
2. RISC-V 32 位 QEMU 使用说明文档。
3. 竞赛平台（代码托管平台、竞赛测试系统）使用文档。
4. 公开的汇编语言基准测试程序及其文档。
5. 参赛队远程调试及本地调试指南。

附：GP 优化简介

RISC-V 可以基于 Global Pointer (GP) 寄存器偏移 (Offset) 对 data 段数据进行快速存取，但允许的 Offset 范围有限（在 32 位指令系统中，Offset 一般为 12 位整型）。

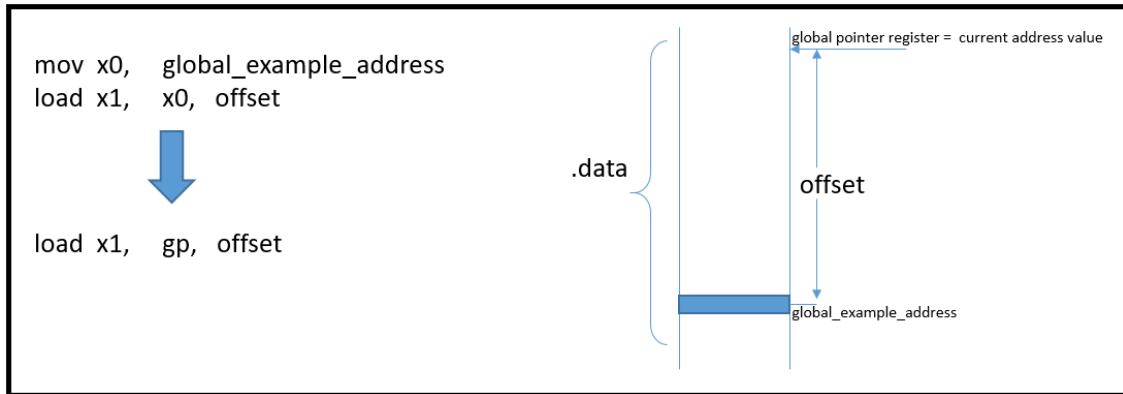


图 1. GP 工作原理

目前 GP 的默认值一般设置为 data 段起始地址，这个设置不能最大限度发挥 GP 寄存器的优势。本赛题要求参赛队必须实现基于汇编链接过程的 GP 寄存器地址值最优设置算法，在可以接受部分运行性能损失的情况下，达到尽可能多地减少代码规模。

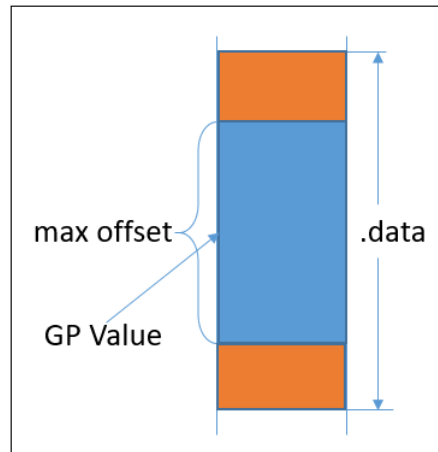


图 2. 设置 GP 值

其中：

1. 在汇编链接过程，可以使用任意 GP 相关的优化并完成 GP 寄存器地址值的设置。
2. 采用指令替换，将满足条件的 data 段存取指令替换为基于 GP 偏移方式的指令。