GOARCH=loong64: 昨天・今天・明天

×

王雪瑞

七牛云
研发工程师

目 录

第零部分

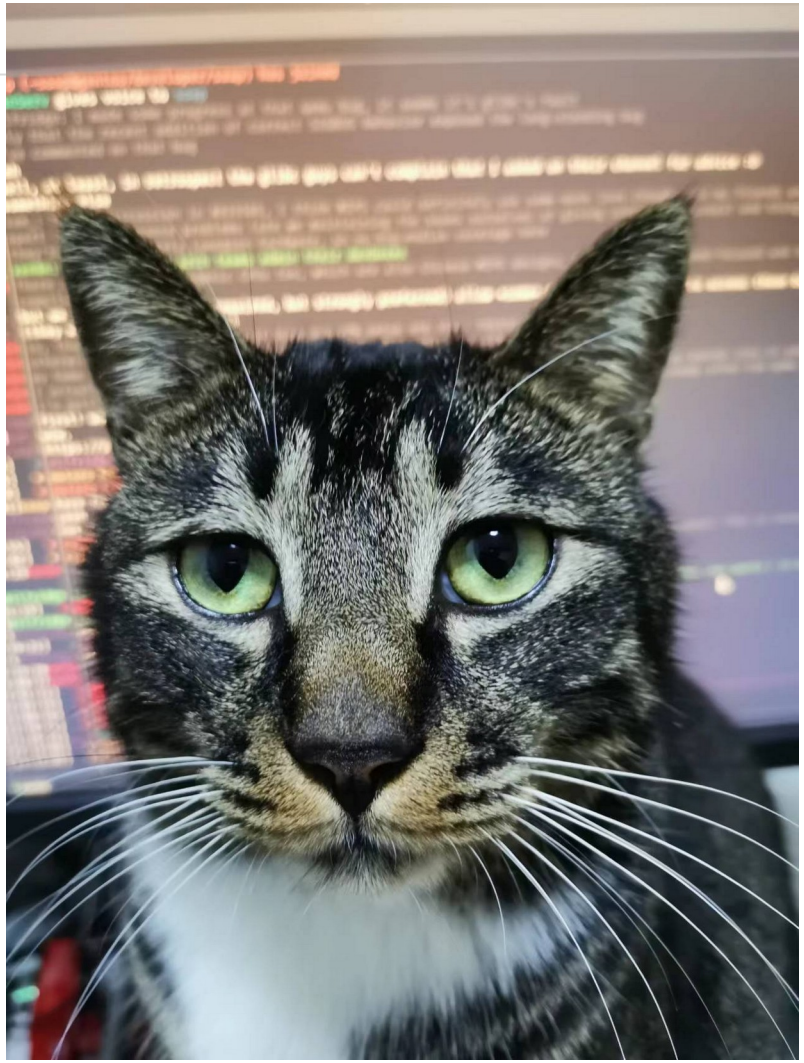
开场介绍

Metadata

- 本分享有一份文字版，信息更多，但是英文版~~因为我懒~~
- A text version of this presentation in English is also available over at my GitHub Gist space.

我是谁？LoongArch 又是啥？

- 七牛写业务的
  - 七牛现在是一家「一站式场景音视频服务商」
- https://github.com/xen0n
- https://blog.xen0n.name
- 生命不息，折腾不止 → FLOSS 用户、开发者
- @golang/loong64 member
- Linux/LoongArch port reviewer
- LLVM/QEMU/binutils/etc. contributor
- Gentoo developer
  - Gentoo Wiki User:Xen0n
  - Gentoo Wiki Project:LoongArch
- NOTE: 本人与龙芯公司，乃至国产化软硬件圈子利益都无关



我是谁？LoongArch 又是啥？

- LoongArch™ = 龙架构™，龙芯中科 (SH.688047) 开发的一种 RISC ISA
- 融合了实践经验的经典 ISA 设计，受了 MIPS、RISC-V 等的影响
- 之前说有打算转向开放标准 (但目前仍然 CC-BY-NC-ND 4.0)
- 官方资源
  - https://www.loongson.cn/
  - https://github.com/loongson/LoongArch-Documentation
  - https://github.com/loongson/la-abi-specs
- 非官方资源
  - https://github.com/loongson-community/docs
  - https://blog.xen0n.name/posts/tinkering/loongarch-faq/
  - https://areweloongyet.com 有周报看

第一部分

昨天

昔我往矣
楊柳依依

Go LoongArch port 的**上游历程**

- 2020.12.16: 首次从内部 GitLab 同步到外界
  - 花絮: 此时的 GOARCH 是 `larch64`，但公开资料大多已散佚

- 2021.05.18: 发起 https://go.dev/issue/46229
  - Bikeshedding: `GOARCH=loongarch64 or loong64?`
  - 跨公司/利益团体合作: KPI，共情，善意，坦率，沟通

- 2022.05.02: `@golang/loong64` team created
- 2022.05.12: port merged

过去时代的余晖

最早的 loong64 port 代码有明显的 mips64x 痕迹，目前仍有部分残留。

- 代码风格、组织方式不够现代 (Plan 9 时代，机翻味道)
- 少数用户可感知的部分之一: Go asm 语法
  - MOVV 是向量搬运吗?
  - ADDVU 是无符号运算吗?

好在: 大部分坑和褶皱仍然属于实现细节，可以重构。

第二部分

今天

今我來思

雨雪霏霏

本节**提纲**

- 周遭环境与版本现状
- 如何为 linux/loong64 适配
- go1.21 与 go1.22 周期，已经做完的 loong64 新鲜事

今夕**何夕**?

- 2023.06.10 此刻的周遭环境如何?
  - `go1.20.5` is latest
  - `go1.21` frozen
  - `x/sys`, `x/net` etc. are tagged
  - 其他相关项目: Linux 6.3, binutils 2.40, gcc 13.1, glibc 2.37, LLVM 16, ...

2023 年已将近过半了。没有提及的项目**默认不需适配即可工作**

如果并非如此，**请去相应上游开 issue 并摇人**

版本现状: **开发者侧**的生态情况

- 库 & 开发工具
  - x/sys, x/net: tagged 版本均可
  - golangci-lint: v1.51.0 (2023.02.02) 以来，上游已提供二进制包 #3459
  - dlv: #2773 由于缺乏 CI 支持而受阻

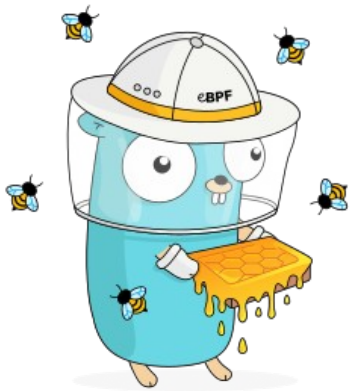
- CI/CD 工具
  - goreleaser: #3277 v1.11.0 (2022.08.29) 以来
  - gox: #166 遗憾的是: 上游不活跃了

版本现状: **用户侧**的生态情况

- 一些可工作的流行项目 (流行 = 我在用/用过/听说过)
  - fatedier/frp
  - Gitea
  - Helm
  - MinIO
  - Hugo
  - Kubernetes (`kubectl` etc.)
  - Syncthing
  - direnv
  - GitHub CLI
  - ...

版本现状: **用户侧**的生态情况

- 暂不工作的流行项目
  - 依赖 cilium/ebpf 的 (fixed in #975 by @zhaixiaojuan but not tagged yet)
    - Kubernetes (`kubelet` etc.)
    - Moby (Docker)
  - 其他没有注意到的



再次强调:

2023 年已将近过半了。没有提及的项目**默认不需适配即可工作**

如果并非如此，**请去相应上游开 issue 并摇人**

版本现状: port 技术状态

- ~~又不是不能用~~
- go1.20 周期的社区贡献
  - Native bit-rotates
  - Intrisification of Add64 and Sub64
  - Enable branchelim pass
  - All from Wayne Zuo @wdvxdr1123

如何为 linux/loong64 适配?

- √ 确保项目可在 go1.19+ 版本构建
- √ 确保关键依赖的版本足够
  - x/sys x/net: need tagged versions
  - go.etcd.io/bbolt: need ≥ 1.3.7
  - github.com/shirou/gopsutil/v3: need ≥ 3.23.2
- √ 使用常识 case-by-case 处理余下情况
  - Pure Go: 条件编译。字节序，对齐，地址空间大小，页大小 etc.
  - cgo: 化归为 C/C++ 移植问题
  - 其他构建系统: 化归为一般性的发行版打包问题

新鲜事: go1.21

- 更多构建模式: `buildmode={pie,c-archive,c-shared}`
- `cmd/asm`
  - LoongArch ELF psABI v2
  - 性能优化软柿子 perf +~1% & 硬核桃 perf +2.33% +0.29%!
- `cmd/compile`
  - 以 MASKEQZ 优化移位操作
  - 不要乘法高位结果就别计算
  - 不用非把除法的商和余数都算出来
  - All from Wayne Zuo too!
- 可靠性改进
  - 崩溃修复，神奇行为修复 e.g. 47627.7 会 overflow `int64` 吗?
  - 更多细节详见文字版

新鲜事: go1.22

- 更多构建模式: buildmode={shared,plugin}
- 库优化
  - hash/crc32 μbench perf +20%
  - runtime atomic ops: xchg μbench time -28.61%, And/Or μbench time -44.96%
- cmd/compile SSA 优化: tracker issue
  - math/bits.TrailingZeros perf +1.02%
  - FMA perf +2.72%
  - math/bits.Len μbench time -48.84%
  - math/bits.ReverseBytes μbench time -65.40%
  - math/bits.Reverse μbench perf -69.90%

汇编性能优化: **软柿子**

```
beq $zero, $zero, offset  // before
b   offset                // after (perf +0.24%)

beq  A, $zero, offset     // before
beqz A, offset            // after (perf-insensitive)

slli.d     B, A, 32       // before
srli.d     B, B, 32
bstrpick.d B, A, 31, 0    // after

slli.d  B, A, 48          // before
srai.d  B, B, 48
ext.w.h B, A              // after (perf +0.84%)
```

不積跬步
無以至千里

汇编性能优化: **硬核桃**

```
// src/internal/bytealg/indexbyte_loong64.s

        PCALIGN $16
loop:
        ADDV    $1, R4
        BEQ     R4, R5, notfound
        MOVBU   (R4), R8
        BNE     R6, R8, loop
```

LA464 中的 4 有什么深刻含义?

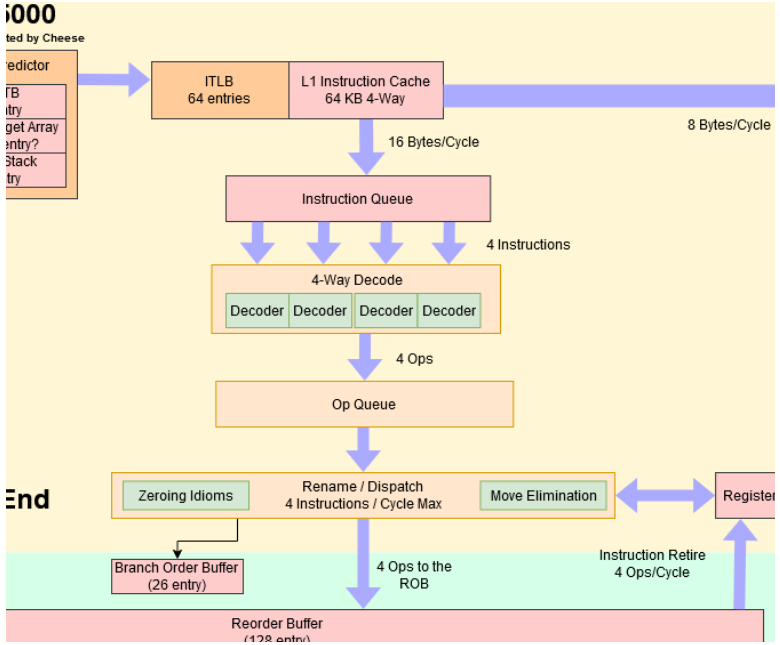


Image courtesy https://chipsandcheese.com/2023/04/09/loongsons-3a5000-chinas-best-shot/

豈敢定居
一月三捷

第三部分

明天

下阶段的**上游**工作

- 梳理、重构 **asm 后端**
  - 从机器可读的指令描述数据，自动生成相关代码
  - 整理汇编语法
    - 规范化助记符、操作数顺序等，去除特例
    - 需要保持兼容历史版本 x/sys 等
  - 更完善的指令支持: LA64 v1.00 查漏补缺, LA664 新用法, ...
- 实现 **regabi**
  - 预期又可 perf +~10%
- 其他
  - 把 g 从 $fp 挪到 $s8: 以便能始终维护帧指针，提升可调试性
  - 支持内部链接 (internal link)

下阶段的**社区**工作

- 拉人入伙！
  - 这是属于「我们」的高基线、高性能、高可维护性技术栈
  - 我们已在 bug trackers 恭候多时 ;-)
- 更多的外交走动
  - 与 Go 关系密切的标准化组织 e.g. CNCF
  - 援引了 Go 语言规范或具体实现的上游规范 e.g. OCI, CNI, ...

致敬幕后的人们

- 早期 (开始上游工作前) 的龙芯内部贡献者们
- @golang/loong64 team
- 社区贡献者们
  - Wayne Zuo @wdvxdr1123
  - Ben Shi @benshi001
- The entire Go team

没有贡献者们的付出，一切都不会发生
接触社区，理解社区，成为社区的一部分

互动环节

感谢您的参与！

咱上游见！