清華大學
Tsinghua University

网络处理加速指令集

路由器实验团队

2020年10月

主要内容

Contents

- 自定义指令的意义
- 自定义指令的方法
- 自定义指令的使用
- 字节序转换加速
- Internet校验和计算加速

自定义指令的意义

- 某些操作常用，但软件实现繁琐，如何优化？
- 软硬件协同设计，CPU中引入新指令进行加速
  - 各类SIMD指令集：加速通用数据运算（shuffle等）
  - Intel AESNI指令集：加速密码学运算
  - Intel VNNI指令集：加速卷积运算（深度学习）
- 联合实验的重大教学意义
  - 培养同学们软硬件协同设计的系统能力

自定义指令的方法

- 寻找可用的编码空间
  - 如RISC-V指令集编码中的OPCODE custom-0~3
- 确定指令功能
  - 支持寄存器或立即数
- 实现自定义指令译码逻辑
- 实现相应指令功能
  - 计算功能可合并在ALU中实现
  - 可增加专用功能器件

inst[4:2] inst[6:5]	000	001	010	011	100	101	110	111 (> 32b)
00	LOAD	LOAD-FP	*custom-0*	MISC-MEM	OP-IMM	AUIPC	OP-IMM-32	48b
01	STORE	STORE-FP	*custom-1*	AMO	OP	LUI	OP-32	64b
10	MADD	MSUB	NMSUB	NMADD	OP-FP	*reserved*	*custom-2/rv128*	48b
11	BRANCH	JALR	*reserved*	JAL	SYSTEM	*reserved*	*custom-3/rv128*	≥ 80b

Table 19.1: RISC-V base opcode map, inst[1:0]=11

自定义指令的使用

- 未经修改的汇编器无法识别自定义指令
- 简单方法：使用.word在汇编中嵌入指令机器码
  - 可用GCC宏拼接

```
#define CUSTOMX(X, rd, rs1, rs2, funct, option)                          \
  CUSTOMX_OPCODE(X) | ((rd) << (7)) | ((option) << (7 + 5)) |            \
      ((rs1) << (7 + 5 + 3)) | ((rs2) << (7 + 5 + 3 + 5)) |              \
      ((funct) << (7 + 5 + 3 + 5 + 5))

#define ROCC_INSTRUCTION_RAW_R_R_R(x, rd, rs1, rs2, funct, option)       \
  .word CUSTOMX(x, ##rd, ##rs1, ##rs2, funct, option)

#define VSETVLI(rd, rs1, imm)                                            \
  ROCC_INSTRUCTION_RAW_R_R_R(0, rd, rs1, EXTRACT(imm, 5, 0),             \
                             EXTRACT(imm, 6, 5), 0b110)
```

- 高级方法：修改工具链，添加支持
  - 如修改binutils或基于RISC-V LLVM
  - https://llvm.org/docs/ExtendingLLVM.html
  - https://llvm.org/docs/WritingAnLLVMBackend.html

字节序转换加速

- 网络字节序为大端序，而RISC-V为小端序，处理时需要转换
- 模拟实现代码冗余，可加入新指令转换字节序

```
00000000 <my_ntohs>:
   0:   00851793                slli    a5,a0,0x8
   4:   00855513                srli    a0,a0,0x8
   8:   00a7e533                or      a0,a5,a0
   c:   01051513                slli    a0,a0,0x10
  10:   01055513                srli    a0,a0,0x10
  14:   00008067                ret

00000018 <my_ntohl>:
  18:   01855693                srli    a3,a0,0x18
  1c:   01851793                slli    a5,a0,0x18
  20:   00851713                slli    a4,a0,0x8
  24:   00d7e7b3                or      a5,a5,a3
  28:   00ff06b7                lui     a3,0xff0
  2c:   00d77733                and     a4,a4,a3
  30:   00e7e7b3                or      a5,a5,a4
  34:   00010737                lui     a4,0x10
  38:   f0070713                addi    a4,a4,-256 # ff00
  3c:   00855513                srli    a0,a0,0x8
  40:   00e57533                and     a0,a0,a4
  44:   00a7e533                or      a0,a5,a0
  48:   00008067                ret
```

Internet校验和计算加速

- Internet校验和（如IP分组头部校验和、UDP及TCP校验和等）采用反码加法，补码机器上计算略微繁琐
- 可加入新指令计算反码加法

其他网络处理加速指令

- 讨论：还有哪些其他网络处理可以引入新指令来加速？

参考资料

- 更多位运算加速指令请见
  - “B” Standard Extension for Bit Manipulation
  - https://github.com/riscv/riscv-bitmanip

本周任务

- 为本组CPU设计并加入网络处理加速指令
  - 可选，算作《计算机组成原理》实验6扩展部分

清華大學
Tsinghua University

谢谢