riscv-check

Маленькое приложение для большой задачи

Автор: Зернов Семён Николаевич
Научный руководитель: Кирилл Константинович Смирнов
Научный руководитель: Владимир Александрович Кутуев

Санкт-Петербургский государственный университет

28 июля 2023г.

План

- Описание проекта
- Цели и задачи
- Аналоги
- Технологический стек
- Особенности реализации

Описание проекта

riscv-check — утилита для анализа кодогенерации

```
chansons@chansons:~/Repositories/riscv-check$ python3 riscv_check.py --help
usage: riscv_check.py [-h] [--format {csv,default}] [--params PARAMS] compiler march mabi opt_level

Compiler options

positional arguments:
  compiler              Compiler name
  march                 Architecture parameter
  mabi                  ABI parameter
  opt_level             Optimization level

options:
  -h, --help            show this help message and exit
  --format {csv,default}
                        output as type of formatted output
  --params PARAMS       forwards parameters directly to compiler
```

Постановка задачи

Цель работы — создать автоматическую систему анализа кодогенерации
Задачи:

- Разработать архитектуру приложения
  - shell compatible
  - Лёгкая для расширения
- Выбрать технические средства для реализации

Аналоги

- **llvm-lit**[1] (внутреннее приложение для тестирования LLVM и Clang)
- **gcc backend**[2] (используются отдельные тесты внутри модуля кодогенерации)

[1] `https://llvm.org/docs/CommandGuide/lit.html`
[2] `https://patchwork.ozlabs.org/project/gcc/list/?series=263719`

Технические средства

- Python 3.10+ (реализация логики)
- C (транслируемый в ассемблер язык)
- CI
  - pylint (статический анализатор)
  - mypy (проверка типов)

Расширяемость

- Утилита shell-дружелюбна

```
chansans@chansans:~/Repositories/riscv-check$ python3 riscv_check.py "/home/chansans/Tools/sc-dt_2022.12-sp1/llvm/bin/clang" --params="--target=
riscv64" rv64gv_zba_zbb_zbc_zbs lp64d 3 --format=csv > clang.txt
chansans@chansans:~/Repositories/riscv-check$ python3 riscv_check.py --format=csv riscv64-unknown-linux-gnu-gcc rv64gv_zba_zbb_zbc_zbs lp64d 3
> gcc.txt
```

Рис.: Перенаправим вывод

```
chansans@chansans:~/Repositories/riscv-check$ diff -y --suppress-common-lines gcc.txt clang.txt
{'compiler': 'riscv64-unknown-linux-gnu-gcc', 'march': 'rv64g | {'compiler': '/home/chansans/Tools/sc-dt_2022.12-sp1/llvm/bin
7,zbb,rev8,rev8_32_manual,False                             | 7,zbb,rev8,rev8_32_manual,True
8,zbb,rev8,rev8_64_manual,False                             | 8,zbb,rev8,rev8_64_manual,True
10,zbb,sext.b,sext.b32,False                                | 10,zbb,sext.b,sext.b32,True
12,zbb,sext.h,sext.h32,False                                | 12,zbb,sext.h,sext.h32,True
39,zbs,binv,binv32,False                                    | 39,zbs,binv,binv32,True
42,zbs,bset,bset32,False                                    | 42,zbs,bset,bset32,True
50,zbs,bclr,bclr32,False                                    | 50,zbs,bclr,bclr32,True
51,zbs,bclr,bclr64,False                                    | 51,zbs,bclr,bclr64,True
52,zbs,bext,bext64,False                                    | 52,zbs,bext,bext64,True
53,zbs,bext,bext32,False                                    | 53,zbs,bext,bext32,True
54,zbc,clmulh,clmulh64_builtin,False                        | 54,zbc,clmulh,clmulh64_builtin,True
55,zbc,clmulh,clmulh32_builtin,False                        | 55,zbc,clmulh,clmulh32_builtin,True
56,zbc,clmulr,clmulr64_builtin,False                        | 56,zbc,clmulr,clmulr64_builtin,True
57,zbc,clmulr,clmulr32_builtin,False                        | 57,zbc,clmulr,clmulr32_builtin,True
58,zbc,clmul,clmul64_builtin,False                          | 58,zbc,clmul,clmul64_builtin,True
59,zbc,clmul,clmul32_builtin,False                          | 59,zbc,clmul,clmul32_builtin,True
64,zba,sh2add,sh2add32,False                                | 64,zba,sh2add,sh2add32,True
67,zba,sh3add,sh3add32,False                                | 67,zba,sh3add,sh3add32,True
69,zba,shladd,shladd32,False                                | 69,zba,shladd,shladd32,True
```

Рис.: diff

Расширяемость

- Утилита независима от bitmanip расширения
- Есть возможность добавить произвольное число тестов
- Попробуем добавить атомарную операцию amoadd.d [3]

```
chansons@chansons:~/Repositories/riscv-check$ mkdir tests/atomics
chansons@chansons:~/Repositories/riscv-check$ mkdir tests/atomics/amoadd.d
chansons@chansons:~/Repositories/riscv-check$ vim tests/atomics/amoadd.d/my_test.c
```

Рис.: Создание теста для инструкции amoadd.d

```c
#include <stdint.h>
#include <stdatomic.h>

int64_t test(_Atomic int64_t count, int64_t n) {
    return atomic_fetch_add_explicit(&count, n, memory_order_relaxed);
}
```

Рис.: Имплементация amoadd.d

[3] https://msyksphinz-self.github.io/riscv-isadoc/html/rv64a.html#amoadd-d

Расширяемость

```
amoadd.d
TEST #1 PASSED: my_test is generated

Total: Passed 55/70 tests
```

Рис.: Результат

Расширяемость

- Прямая возможность добавить свой формат

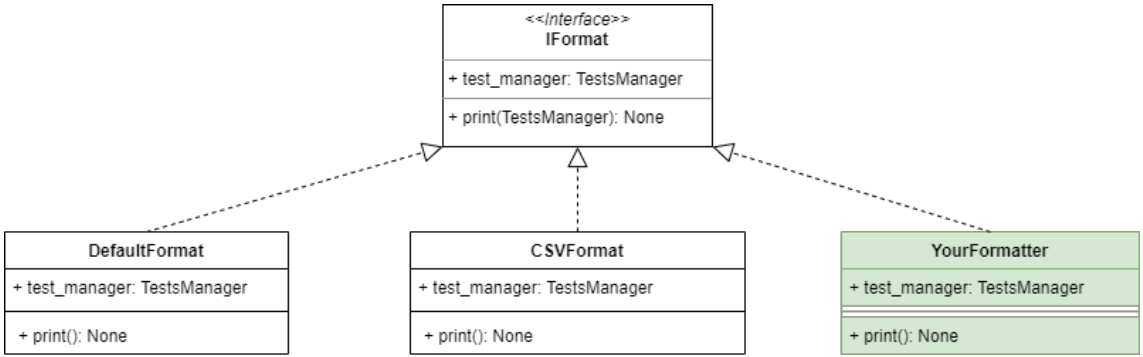

Рис.: Иерархия классов

Результаты

- Поддержка bitmanip в современных компиляторах (gcc и clang[4]) сильно отличается

```
slli.uw
TEST #1 FAILED: slli.uw is not supported

add.uw
TEST #1 PASSED: add.uw is generated

sh2add
TEST #1 FAILED: sh2add32 is not supported
TEST #2 PASSED: sh2add64 is generated

sh3add
TEST #1 PASSED: sh3add64 is generated
TEST #2 FAILED: sh3add32 is not supported

sh1add
TEST #1 PASSED: sh1add64 is generated
TEST #2 FAILED: sh1add32 is not supported

Total: Passed 35/69 tests
```

Рис.: gcc 13.1.0

```
add.uw
TEST #1 PASSED: add.uw is generated

sh2add
TEST #1 PASSED: sh2add32 is generated
TEST #2 PASSED: sh2add64 is generated

sh3add
TEST #1 PASSED: sh3add64 is generated
TEST #2 PASSED: sh3add32 is generated

sh1add
TEST #1 PASSED: sh1add64 is generated
TEST #2 PASSED: sh1add32 is generated

Total: Passed 54/69 tests
```

Рис.: clang из Syntacore SW Tools

[4] https://syntacore.com/page/products/sw-tools

Репозиторий

Рис.: Репозиторий (ссылка)