

Темы от YADRO: TMS и RISC-V

Юрий Литвинов
y.litvinov@spbu.ru

15.09.2022

Test Management System

- ▶ Веб-приложение для управления тестовыми планами и результатами тестирования
- ▶ Аналог TestRail, Test It, TestLink и т.п.
- ▶ Нужно YADRO для работы QA-отдела, первый прототип по плану уже в декабре
- ▶ В основном на Python + PostgreSQL
- ▶ Нам достанется лишь небольшая часть проекта
- ▶ Основные фишки: тест-кейсы с настраиваемыми полями, группировка по сьютам и проектам, тэги, приоритеты, параметризация, конфигурации тестового окружения, тест-планы, учёт времени выполнения, учёт результатов с настраиваемыми полями, интеграция с Jira и системами модульного тестирования, репортинг, REST API

Конкретные задачи (1)

Пока что

- ▶ Отчёты о результатах тестирования
 - ▶ Скорее всего, отдельный сервис
 - ▶ HTML-документ со стилем и скриптами для удобного просмотра (речь о тысячах тестов)
 - ▶ Поддержка автотестов (см. Allure Framework)
- ▶ Система уведомлений
 - ▶ Скорее всего, отдельный сервис, но может оказаться плагином к сайту на Django
 - ▶ Письма, уведомления внутри системы, Telegram

Конкретные задачи (2)

- ▶ Свой фронтенд
 - ▶ У YADRO есть свой UI Framework, который они нам не дадут, и будут писать фронтенд на нём
 - ▶ Open source-версия на чём угодно, взаимодействие через REST API
 - ▶ Надо будет написать и заглушки сервера
 - ▶ Скорее всего, командная работа
- ▶ Интеграция с Jira
 - ▶ Больше фронтенд внутри самой системы
 - ▶ Скорее всего, потребуется устройство на стажировку

RISC-V

- ▶ Целое направление — инструментальная поддержка процесса разработки под RISC-V
- ▶ Актуально, поскольку открытая процессорная архитектура
- ▶ Много разного плана задач, и по мере погружения в тему будет ещё Больше
 - ▶ Кажется, что вплоть до кандидатской
- ▶ Направления:
 - ▶ Симуляция, Instruction-accurate и Cycle-accurate
 - ▶ Оптимизация библиотек и инструментов под RISC-V
 - ▶ Рантаймы языков программирования: Python, Go, Lua
 - ▶ Отладка и анализ трасс
 - ▶ IDE
- ▶ Нужно желание копать в чужом коде и непонятных штуках
 - ▶ Это интересно, и уникальные знания

Конкретные задачи (1)

Пока что

- ▶ Verilator: SCR1
 - ▶ SCR1 — открытое ядро Syntacore RISC-V
 - ▶ Verilator — тул, который по описанию процессора на Verilog генерирует для него симулятор
 - ▶ Надо запустить SCR1 на Verilator и прогнать на получившемся симуляторе тесты от Syntacore
 - ▶ Просто? Не думаю
- ▶ Сравнительный обзор симуляторов
 - ▶ Посмотреть на разные симуляторы, способные симулировать RISC-V
 - ▶ Научиться их запускать
 - ▶ Составить таблицу поддержки профилей
 - ▶ Сравнить производительность

Конкретные задачи (2)

▶ Оптимизация Verilator

- ▶ Известно, что некоторый код на Verilog порождает медленные симуляторы
- ▶ Например, сложение может быть расписано как страшная конструкция из транзисторов и Verilator не догадается заменить его на “+”
- ▶ Задача — найти такие случаи и предложить изменения в Verilator
- ▶ Много нетривиального кода на C++

Ещё направления

- ▶ IDE
 - ▶ Готовый инструментарий разработки под RISC-V с симулятором, отладкой изнутри IDE и т.п.
 - ▶ Рекомендации по портированию — например, подсветка инструкций с плавающей точкой, если их нет в профиле
- ▶ Анализ трасс в Verilator — печатать состояние процессора
- ▶ Cycle-accurate-симуляция в симуляторе SAIL для RISC V
 - ▶ Почитать про это статьи, повторить эксперимент Coyote
 - ▶ Перенести в SAIL
 - ▶ Код на OCaml!
- ▶ Оптимизация Linux под RISC-V
- ▶ JIT для SAIL
- ▶ Реализация расширений в SAIL и SPIKE
- ▶ Моделирование DDR
- ▶ ...