Библиотека программирования гетерогенных встраиваемых архитектур

Григорий Киргизов

Научный руководитель: Я.А. Кириленко

Рецензент: Д.А. Мордвинов

Кафедра Системного Программирования
Санкт-Петербургский Государственный Университет

Введение

Проблемы разработки ПО для гетерогенных встраиваемых систем:

- Необходимость использования различных наборов инструментов
- Постоянный обмен сообщениями между процессорами
- Настройка под определенные условия среды
- Настройка под различные периферийные устройства

Предыдущая работа

К.И. Мелентьев, Р. Белков, Я.А. Кириленко
”Система программирования кибернетических гетерогенных архитектур с использованием LLVM”,
SEIM-2017

- Показана жизнеспособность идеи
- Основные архитектурные решения оставлены неизменными

Основные моменты

- Идея — динамическая кодогенерация, компиляция и загрузка кода на периферийные процессоры
- DSL — ключевая часть системы
  - Транслируется в LLVM IR
- LLVM — обеспечивает компиляцию в машинный код
  - Для широкого спектра платформ

Поставленные задачи

- Проанализировать существующую реализацию
- Перепроектировать систему
- Реализовать согласно новой архитектуре
  - Подсистему DSL
  - Библиотеку программирования гетерогенных встраиваемых архитектур
- Провести апробацию

Обзор

- Texas Instruments MultiCore SDK
  - Для ARM+DSP устройств (архитектуры TI Keystone)
- LLVM ORC (On-Request Compiler)
  - Подсистема LLVM для JIT компиляции
- Delite
  - Фреймворк разработки статически компилируемых DSL (компиляция в C++/CUDA/Scala код)
  - Нацелен на параллельные вычисления (кластеры и CPU+GPU архитектуры)

Проблемы существующей реализации

- Высокая связность компонентов системы
  - DSL может быть независимым
- Смешение стадий работы с кодом
- Пренебрежение инструментами LLVM
  - Неэффективная генерация LLVM IR
  - Отсутствие оптимизаций генерируемого кода

DSL

DSL как средство описания вычислений

- Позволяет манипулировать выражениями C-подобного языка

Особенности:

- Разделение этапов построения выражений и определения функций
- Обобщенные (generic) DSL функции
- Специальные загруженные функции
- Независимость от остальных частей системы

Пример кода на DSL

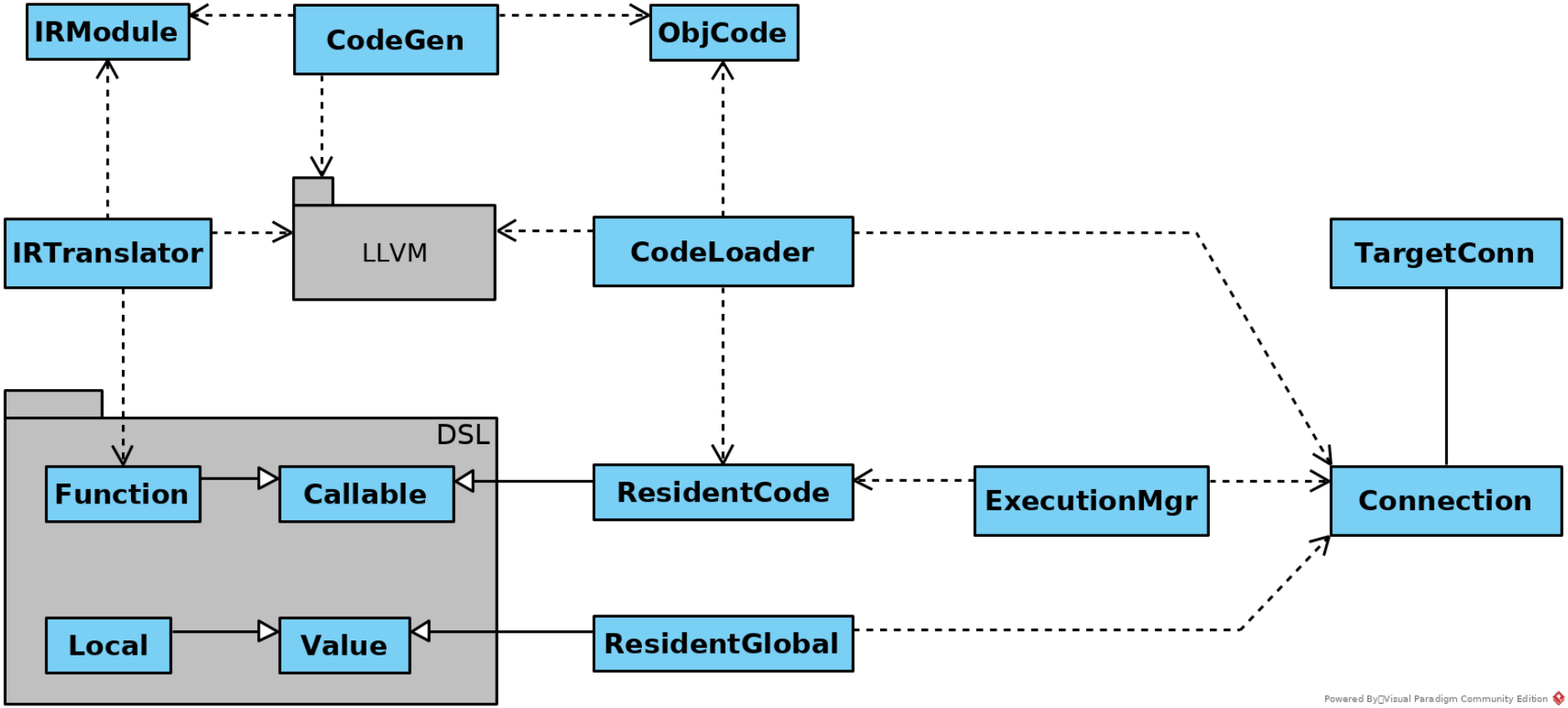
```
1   auto factorial_gen = [](auto n){
2      Var<int> res{1}; // local variable
3      return (
4          While(n > Lit{0}, (
5              res *= n,
6              n -= 1
7          )),
8          res // last value is returned
9      );
10  };
11  auto factorial = make_dsl_fun<unsigned>(factorial_gen);
```

Библиотека программирования

Компоненты:

- Модуль компиляции
- Модуль загрузки кода
- Модуль исполнения удаленного кода
- Связующий модуль
  - Работает по определенному командному протоколу
  - Поддержка новой платформы — это поддержка протокола в прошивке

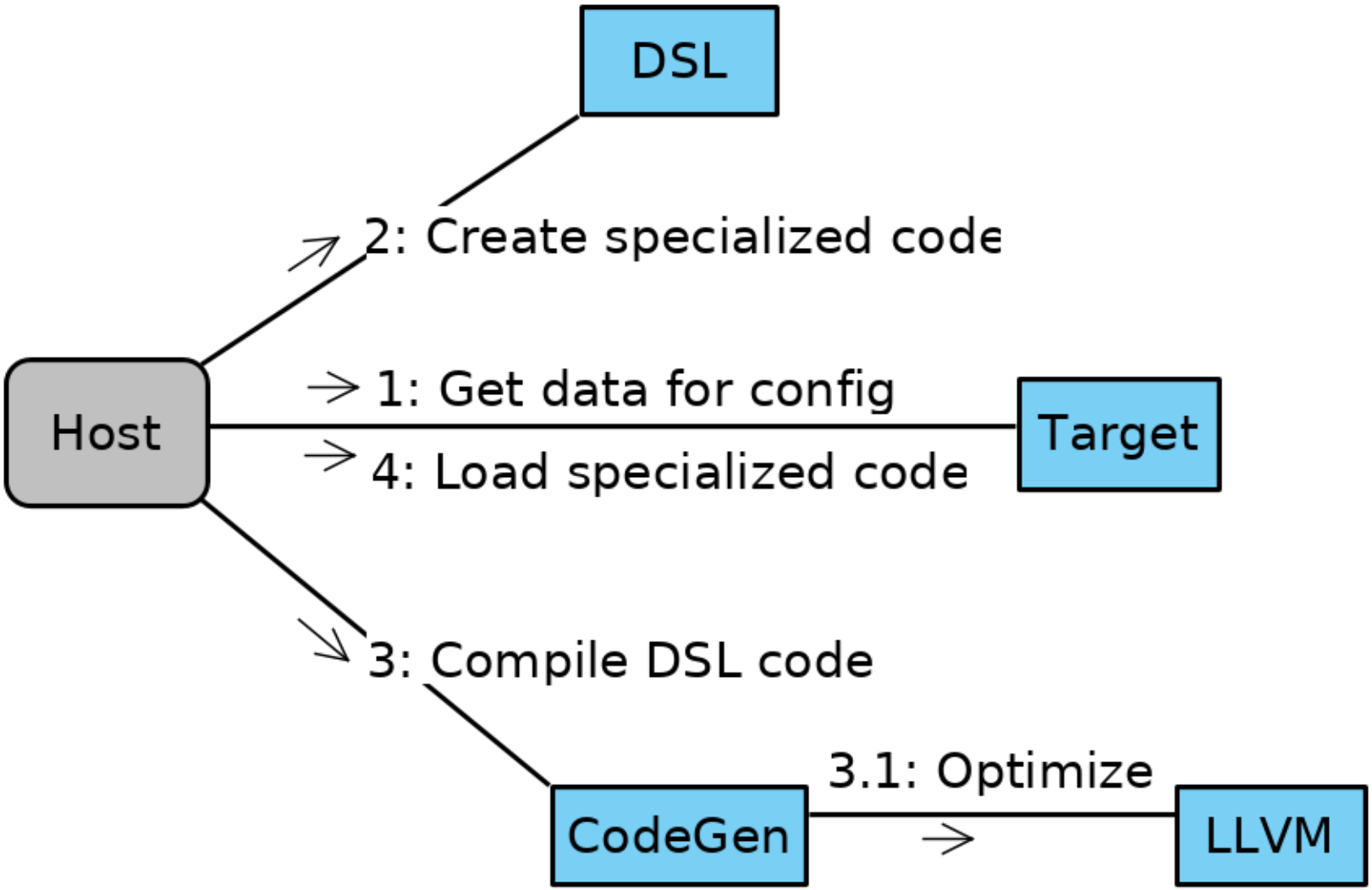
Архитектура системы



Динамическая специализация

- Динамическая настройка встроенных систем
  - Условия среды
  - Конфигурация системы
- Динамические оптимизации
  - Специализация DSL кода для определенных значений
  - LLVM избавляется от констант и генерирует более эффективные инструкции

Динамическая специализация



Апробация

Тестовые конфигурации (host ▶ target):

- `x86_64-linux` ▶ `armv7a-linux`
- `x86_64-linux` ▶ `armv7em-none`
- `armv7a-linux` ▶ `armv7em-none`

Результаты

- Выполнено перепроектирование системы
- Проведена реализация системы
  - Переиспользуемый DSL для метапрограммирования
  - Библиотека программирования гетерогенных встраиваемых архитектур
- Проведена апробация системы
  - Реализована прошивка для семейства микроконтроллеров stm32f4 (armv7em)

По результатам работы принята статья и проведен доклад на конференции SYRCoSE 2018