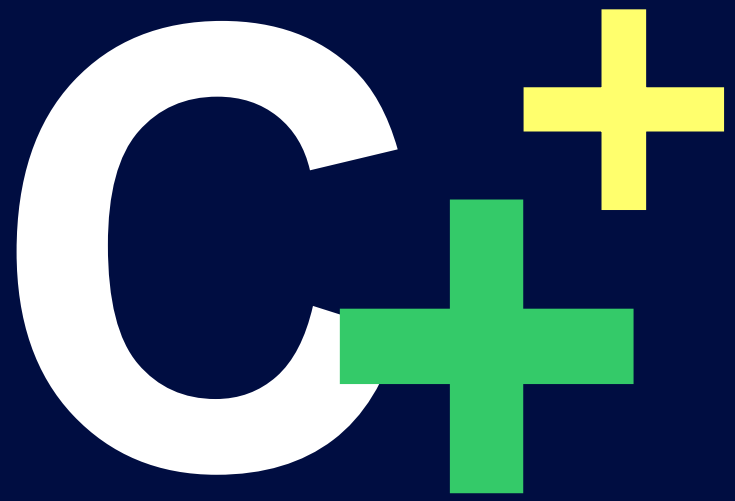


Создание среды разработки для C++

Взгляд изнутри

Профессиональная
конференция
для embedded-
разработчиков C++

МИР

МИЛАНДР
ГРУППА КОМПАНИЙ

Мастерская
инструментов
разработки

Отладчик на базе LLDB. Проблемы и решения.

Портирование для удалённой отладки barebone -системы

Илья Гозман

Июнь 2019

А почему LLDB?

- Написан на C++ (GDB написан на C)
- Лицензия MIT/BSD (GDB - GPLv3)
- Сравнительно новый проект – нет избытка легаси-кода, как в GDB
- Состоит из плагинов, что облегчает портирование
- Полное API на Python – помогает в тестировании
- Уже имеется портированный LLVM

Но всё оказалось не так просто

Специфика задачи

- Версия LLDB ограничена версией LLVM
- Необходимость удаленной и локальной отладки
- Отсутствие готового сервера отладки с поддержкой GDB RSP
- Сервер удаленной отладки должен работать под Windows
- Отладка на отдельных процессорах

Портирование lldb-server под Windows

- Из коробки не собирается
- Лишние сигналы, отсутствие инициализации и поллинга сокетов, неправильная работа с каналами (pipes)
- Нет гайдов портирования lldb-server – обычно используется собственный RSP-совместимый сервер отладки (например, debugserver под MacOS)
- Некоторые части не портированы

Решение: исправления + применение кода с апстрима

Невозможность запуска lldb -server напрямую на плате

- Целевая архитектура (lynx) отличается от архитектуры удаленной платформы (x86), где запускается lldb -server
- Компоненты lldb -server не поддерживают данный случай из коробки , и пытаются получить информацию о потоках/процессах через модуль удаленной платформы - т.е. средствами Windows
- При файловых операциях нужно работать именно с Windows
- Нет отдельных команд загрузки образа в память (появились в апстриме)

Решение: проверка архитектуры процесса при связанных с ним запросах, образ загружается при запуске

32-битная адресация памяти

- В LLDB используется только 8 -битная адресация
- В LLVM/LLDB используются свои классы для работы с данными, адресами и смещениями
- В отладочной информации (dwarf) - смещения только в 8 -битных байтах
- В тулчейне CM-LYNX- 32-битная адресация, затрагивает все инструменты
- Строковые константы в char32
- В lynx 32 -битная адресация памяти

Решение : адреса в 32-битных байтах, смещения - в 8-битных байтах, размер байта для адресов записан в описании архитектуры

Отсутствие сигналов (или прерываний) у драйвера аппаратного отладчика

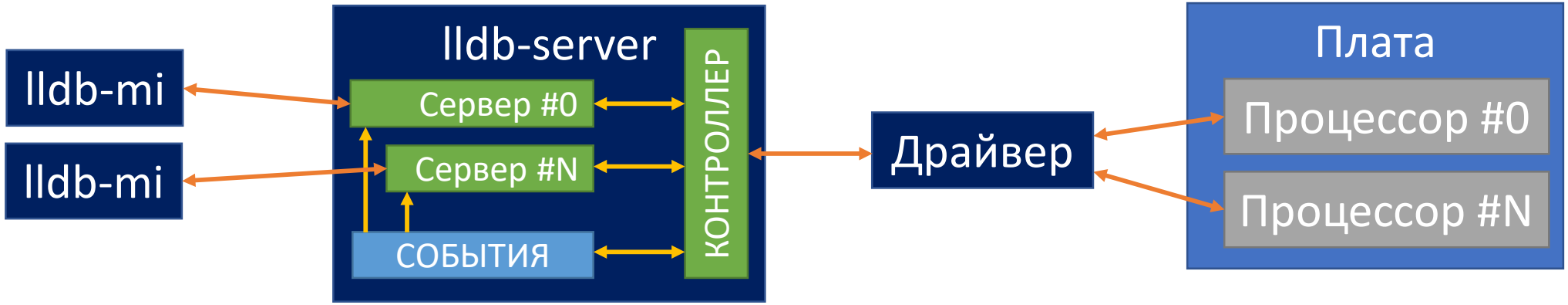
- LLDB предполагает получение событий от системы и возможность проверки их наличия без блокировки. Например, ptrace в unix - проверяет через pselect
- Драйвер поддерживает только получение статуса по запросу
- К системе опроса событий также привязаны события ввода/вывода сокетов и файлов – используется общий интерфейс
- Реализация опроса событий под Windows начала поддерживаться в LLDB 6.0

Решение: запрос статуса через определённый интервал, по таймеру, отдельно для каждого процесса

Драйвер поддерживает только одно активное подключение

- По умолчанию lldb-server делает форк для отладки каждого процесса
- По умолчанию lldb-server поддерживает только одно входящее соединение и завершается после его разрыва
- В драйвере есть команды переключения активного ядра

Решение: использование потоков для отдельных серверов, с общим потоком поллинга событий (Mainloop) и общим клиентом для драйвера
Бонус - избавились от каналов и форков, общий лог событий



Прямые команды к драйверу

- Команды, не требующие контекста отладки

Пример: инициализация драйвера, сброс (reset), тестирование JTAG, подключение к одному/нескольким процессорам, проверка доступности процессоров

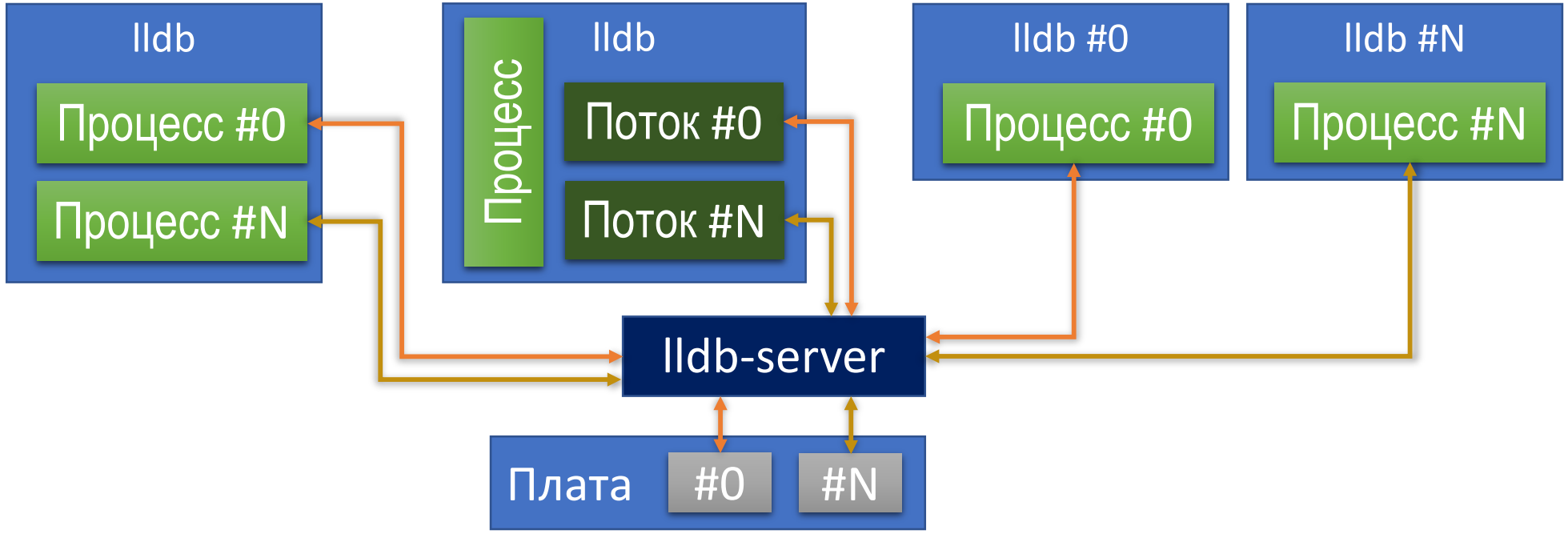
Решение: использование функции lldb для запуска shell-команд на удаленной платформе с зарезервированными командами, например "jem reset"

Бонус - удобно тестировать, подкладывая реальные скрипты с нужным именем

Отладка отдельных процессоров (SMP/AMP)

Рассматривались следующие варианты:

1. Один процесс на процессор
2. Один поток на процессор
3. Один lldb (клиент) на процессор



Отладка отдельных процессоров *VS* реальность

- №1: multi -target режим в LLDB имеет большие проблемы с отправкой событий от отдельных процессов и с потоками ввода/вывода
- №2: LLDB пока поддерживает только Process-centric режим, т.е. нет полного контроля за отдельными потоками (non -stop режим)
- №3: Усложняется синхронизация команд к разным процессорам, другое представление в IDE из-за нескольких клиентов

Решение (не идеальное):

Использование варианта №3, синхронизация на уровне IDE

Для SMP – также №2 с ограничениями

Вопросы?

Мастерская
инструментов
разработки

Илья Гозман

hi@mir.astrosoft.ru