Эмулятор EM1.
Задачи, возможности и результаты

Баклановский М.В., старший преподаватель кафедры системного программирования СпбГУ, baklanovsky@mail.ru

Лагунов Л.Л., старший разработчик ПО ООО «Софтком», leonid.lagunov@softcom.su

Сибиряков А.Е., старший преподаватель департамента математики, механики и компьютерных наук УрФУ, a.sibiryakov@urfu.ru

Ханов А.Р. , старший преподаватель кафедры системного программирования СпбГУ, st036451@student.spbu.ru

Аннотация

Рассмотрены основные подходы, использованные при разработке эмулятора EM1. Предложены способы эмуляции внешних устройств и многопроцессорных систем. Описана методика тестирования эмуляторов и процессоров на соответствие с документацией.

Ключевые слова

Внешние устройства, симуляция, тестирование эмуляторов, эмуляторы, эмуляция процессоров.

Введение

Эмулятором называют программу, которая имитирует поведение реального вычислителя. Эмулируются наборы API-вызовов, инструкции целевых процессоров, микроархитектура вычислителей либо отдельные логические элементы [1,2].

Известно большое количество эмуляторов для популярных процессоров [3,4,5,6]. Практически все они (даже популярный эмулятор qemu) обладают рядом серьезных недостатков:

- добавление новых процессоров требует существенных

изменений/добавлений в коде и не всегда возможно в разумные сроки даже при наличии в открытом доступе исходных текстов и соответствующей лицензии;
- скорость работы эмулируемых программ обычно значительно ниже чем на реальных вычислителях (иногда до сотен раз);
- многие разработки не являются отечественными и могут содержать недокументированные фрагменты кода.

Разработчики редких вычислителей, таких как DSP-ядра, процессоры для микроконтроллеров, иногда включают эмуляторы в коммерческие пакеты поставки. В отсутствие таковых приходится писать новый эмулятор "с нуля". При этом такая программа может не обладать всеми необходимыми возможностями по трассировке и отладке и потребовать существенных трудовых затрат. Неэффективная реализация (обычно на ЯВУ) не позволяет выполнять эмулируемые программы с достаточной скоростью. Немаловажную роль здесь играет и точность эмуляции - эмулятор может не соответствовать документации, а также не достаточно точно эмулировать микроархитектуру (например такты процессора), что может быть существенным при решении некоторых задач, работающих с аппаратурой. Известны попытки предложить универсальные решения данных проблем.

Система WInter [6] предназначена для разработки эмуляторов целых вычислительных систем. В ней поддержано несколько известных процессоров для микроконтроллеров, а также процессор общего назначения Intel 8086. Однако все возможности отладки были реализованы внутри собственной графической оболочки, что не позволяет подключать сторонние программы. Кроме того, проект устарел и не поддерживается с 2003 года.

Еще одной такой попыткой можно считать известный эмулятор qemu. Однако его архитектура не подходит для тонкой эмуляции микроархитектуры: работы конвейеров, таймеров, шин обмена данными. Поскольку реализация кодов инструкций производится на языке C++, такой эмулятор будет медленным.

Большая часть современных работ на данную тему посвящена эмуляции вычислительных систем на FPGA, что лишает их возможности работы на компьютерах общего назначения

Многие эмуляторы разрабатывается для уже существующих гостевых аппаратных платформ и не содержат встроенных механизмов низкоуровневого тестирования на соответствие между документацией, эмулятором и целевой вычислительной системой.

Постановка задачи

Важной [7] прикладной задачей представляется разработка мультиэмулятора, обладающего следующими характеристиками:

- быстрая (2-3 месяца) разработка эмуляторов для еще не выпущенных процессоров и включение их в мультиэмулятор;
- возможность начать полноценную разработку прикладного ПО параллельно с производством процессоров, значительно уменьшив при этом время готовности решений;
- использование дополнительных возможностей, предоставляемых эмуляцией, при отладке и оптимизации разрабатываемого ПО;
- высокая скорость выполнения эмулируемых программ;
- максимально точная эмуляция времени выполнения;
- использование внешних по отношению к пространству вычислений средств отладки и трассировки;
- встроенный механизм тестирования эмулятора;
- простое решение для эмуляции внешних устройств;
- API для интеграции в различные графические среды;
- эмуляция многопроцессорных систем, состоящих в том числе и из процессоров разных архитектур;
- собственная хорошо документированная разработка, позволяющая вносить существенные изменения и дополнения в алгоритм работы эмуляторов.

Основные решения

Создание нового эмулятора каждый раз "с нуля" предполагает серьезные затраты на разработку, поэтому при создании очередного эмулятора для EM1 используются готовые, заранее разработанные модули:

- парсеры команд и мнемоник;
- поддержка протокола GDB;
- API для работы с консольной или графической оболочкой;
- инструменты интеграции, отладки и трассировки.

Описание каждого эмулируемого процессора и правила разбора опкодов инструкций сохраняются в специализированной базе данных [8]. Обработчики инструкций разрабатываются на языке ассемблера хост-компьютера (Intel x64) в ручном или полуавтоматическом режиме. Исполняемый файл эмулятора автоматически генерируется с использованием данных из БД, модулей и обработчиков инструкций. Для

увеличения скорости работы парсеров при разработке эмулятора для каждого очередного процессора выполняется тестирование различных вариантов разбора, в финальную сборку попадает самый скоростной вариант.

EM1 содержит встроенный GDB-сервер. В качестве клиентов могут использоваться популярные средства отладки, поддерживающие протокол GDB, например - Visual Studio Code, IDA Pro, GDB-Multiarch, PEDA, Angr, Radare2. Подключение к эмулятору выполняется по сети с любого компьютера в Интернете.

Предлагаемая система тестирования эмуляторов и микропроцессоров (ТЭМП) содержит тестовую базу, разрабатываемую в соответствии с документацией на процессор. ТЭМП использует специализированный язык описания тестов, позволяющий выполнять тесты в пакетном режиме. Эмулятор перед сдачей в эксплуатацию проходит полное тестирование. Позднее такое же тестирование пройдут образцы процессора. Это даст нам уверенность в том, что программы на процессоре будут выполняться точно так же, как на эмуляторе.

EM1 эмулирует не только алгоритмы исполнения инструкций, но и времена их выполнения в тактах процессора. До момента готовности образцов процессора время выполнения каждой инструкции вычисляется в соответствии с документацией. В дальнейшем в ходе тестирования процессоров выполняется дополнительное изучение времен выполнения инструкций и их комбинаций, исследуется влияние конвейеризации и вносятся изменения в соответствии с полученными результатами.

Достигнутая точность эмуляции позволяет выполнять на эмуляторе разработки практически любой сложности, включая, например, компиляторы и другое сложное ПО. Используемый при разработке ПО язык программирования большого значения не имеет, достаточно применять кросс-компилятор, максимально поддерживающий возможности процессора.

Эмуляция внешних устройств является одной из сложнейших задач при создании эмуляторов. Основная причина состоит в том,, что разработчики процессоров при разработке SoC чаще всего пользуются готовыми устройствами внешних производителей. Документация на устройства не всегда доступна в полном объеме и часто не содержит информации, достаточной для разработки полноценных драйверов. В случае разработки в таких условиях драйвер устройства для эмулятора с большой вероятностью не сможет в дальнейшем использоваться при работе с реальной аппаратурой и подобная эмуляция устройств становится

неэффективной.

Вместо эмуляции работы устройств предлагается выполнять эмуляцию API-функций, работающих с устройствами. Основные преимущества такого подхода:

- разработка прикладного ПО может выполняться при отсутствии ОС;
- функции реализуются один раз для каждого конкретного устройства и в дальнейшем переиспользуются;
- в случае если эти функции по логике работы совпадают с API той ОС, которая будет использована, затраты на доработку ПО под реальный вычислитель могут быть сведены к нулю;

К эмулятору прилагаются библиотеки для специальной сборки исполняемых файлов. Библиотеки содержат в том числе и необходимые для работы ПО функции (например printf и malloc). Собранный с такими библиотеками исполняемый файл работает одинаково как под эмулятором так и на "голом" железе.

Один запущенный эмулятор процессора (ЭП) эмулирует один процессор. На одном или нескольких компьютерах в сети могут быть запущены несколько эмуляторов одного и того же или разных процессоров. Отдельный эмулятор может быть запущен в режиме эмуляции многопроцессорного вычислителя. Такой эмулятор многопроцессорной системы (ЭМПС) связывает между собой эмуляторы различных (в т.ч. и по архитектуре) процессоров и обеспечивает обмен информацией между ними используя общую память. Каждый ЭП имеет несколько интерфейсов - Init, Cons и Link - обеспечивающих возможности подключения. Схема сборки многопроцессорного эмулятора приведена на Рис.1.

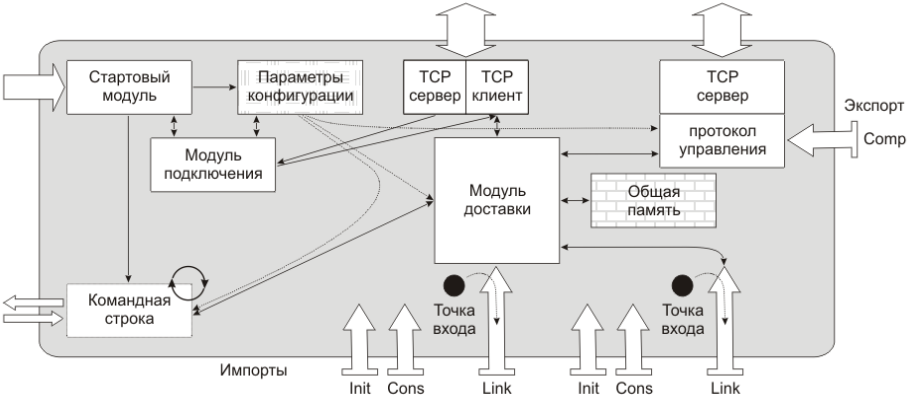

Рис. 1. Схема сборки многопроцессорного эмулятора.

ЭМПС позволяет отлаживать и оптимизировать одновременную (параллельную) работу ПО на разных процессорах. При этом каждый ЭП может быть запущен в отдельном потоке на отдельном ядре или на отдельном компьютере.

Внешняя по отношению к вычислительному пространству гостевого процессора эмуляция предоставляет уникальные возможности для отладки и трассировки, отсутствующие в реальных процессорах, например:

- точки останова без внесения изменений в секции кода (в т.ч. и в ПЗУ);
- управление отладкой в зависимости от состояния или изменения областей данных (в любом количестве);
- наблюдение за служебными регистрами эмулируемого процессора даже если они не доступны пользователю в user mode.

Разработанный механизм наблюдения за процессом вычислений позволяет не только отлаживать код, работающий в привилегированных режимах, но и отлаживать программы-отладчики.

Заключение

К настоящему времени с использованием предложенной технологии разработаны эмуляторы процессоров из семейств ARM, STM32, MIPS32, КОМДИВ64, DSP TMS320C6x.

Скорость работы эмуляторов настолько высока что, например, известный тест LINPACK в эмуляторе MIPS32 на компьютере с далеко не

современным процессором Intel PentiumG 4го поколения выполняется примерно в 6 раз быстрее чем на реальном микрокомпьютере Onion Omega2+.

EM1 работает в среде ОС Linux (Ubuntu, Arch, Debian, Astra и др.) или Windows 64 (в т.ч. под WSL).

Режимы внешней трассировки и отладки в сочетании с ЭМПС позволяют разрабатывать и оптимизировать программные решения со сложной синхронизацией.

Реализована концепция “одного бинарника”, когда любой исполняемый файл одинаково работает как на эмуляторе, так и на процессоре без ОС.

Все разработки выполнены "с нуля" без использования готовых (чужих) программ или их фрагментов. Основной язык программирования — ассемблер.

Литература

1. A. Svensson, "Software primitives for emulation of multiprocessor architectures," Twenty-Third Annual Hawaii International Conference on System Sciences, 1990, pp. 48-56 vol.1, doi: 10.1109/HICSS.1990.205098.
2. Речистов Г.С. и др., https://atakua.org/w/images/simulation-lectures-latest.pdf [дата просмотра 25.04.2022].
3. https://en.wikipedia.org/wiki/List_of_emulators [дата просмотра 25.04.2022].
4. https://en.wikipedia.org/wiki/List_of_computer_system_emulators [дата просмотра 25.04.2022].
5. QEMU’s documentation, https://www.qemu.org/docs/master/ [дата просмотра 25.04.2022].
6. Интегрированная среда разработки программного обеспечения встроенных систем Winter, https://newit.gsu.by/ru/winter/ [дата просмотра 25.04.2022].
7. Баклановский М.В., Оносовский В.В., Терехов А.Н., Тимохин Д.В., Мартынов В.И., Милюченко С.Г., Нестеренко А.Н., Сгонников А.С., Сгонников Д.С., “Инструментальная среда разработки программно-аппаратных комплексов”, XLV академические чтения по космонавтике, Москва, 30 марта – 02 апреля 2021 года, стр. 523-526.
8. Баклановский М.В., Сибиряков А.Е, Нестеренко А.Н., Дмитриев В.С., Сгонников Д.С., “Эмуляторы многопроцессорных вычислительных систем и отладка на уровне системы”, XLV академические чтения по космонавтике, Москва, 30 марта – 02 апреля 2021 года, стр. 517-519.