МНОГОПРОЦЕССОРНАЯ СИСТЕМА С ОБЩЕЙ ПАМЯТЬЮ НА ОТЕЧЕСТВЕННЫХ СИГНАЛЬНЫХ ПРОЦЕССОРАХ

ДМИТРИЙ МАТЮНИН, инженер-программист 1-й категории, matyunin.d@milandr.ru, АО «ПКК Миландр»

В статье рассматривается возможность проектирования многопроцессорной системы с общей памятью на базе сигнальных процессоров со статической суперскалярной архитектурой 1967ВН028 (АО «ПКК Миландр»).

Рассмотрим несколько концепций, применяемых для проектирования систем с многопроцессорной архитектурой: симметричную (SMP) и ассиметричную (ASMP) мультипроцессорность. В системах, созданных на базе архитектуры симметричной мультипроцессорности, у каждого процессора появляется возможность выполнять любую задачу независимо от местонахождения данных обработки для этой задачи. В таких системах задачи перемещаются между процессорами, обеспечивая эффективное распределение нагрузки. Такая архитектура – один из простых способов для масштабирования системы, повышения переносимости исходного кода и скорости его разработки. Недостатком таких систем является ограничение числа процессоров в системе, т. к. при их увеличении возрастает нагрузка на общую шину, и вследствие этого общая производительность падает. Кроме того, недостатком можно считать определенную сложность инструментов разработки и средств (компилятор, операционная система, средства отладки), предоставляющих требуемый для реализации такой системы функционал. В концепции с ассиметричной мультипроцессорностью каждый процессор может использоваться для решения отличной от других процессоров задачи. Явным недостатком такой концепции является низкая масштабируемость системы, сложность обеспечения равномерного распределения общей нагрузки. Однако такой подход накладывает меньше требований к средствам разработки и отладки и, в общем, является более простой альтернативой симметричной мультипроцессорности.

Таким образом, в качестве архитектуры для многопроцессорной системы с общей памятью на базе процессора 1967ВН028 была выбрана именно архитектура с симметричной мультипроцессорностью. На рисунке 1 представлена структурная схема классической мультипроцессорной системы с общей памятью. Каждый процессор (ЦП) в такой системе обладает схожими характеристиками по производительности и схожим образом подключается к общей шине, соединяющей его с другими процессорами, общей памятью и устройствами ввода/вывода (УВВ).

Однако сигнальный процессор (ЦСП) 1967ВН028 имеет более сложную архитектуру из-за наличия у него внутренней (локальной) памяти и устройств ввода/вывода (порты связи, входы внешних прерываний и запросов к каналам прямого доступа к памяти на обслуживание внешних устройств). Дополнительно стоит отметить, что в связи с архитектурными особенностями данного процессора максимальное количество процессоров в одной многопроцессорной системе, или вычислительном кластере, может

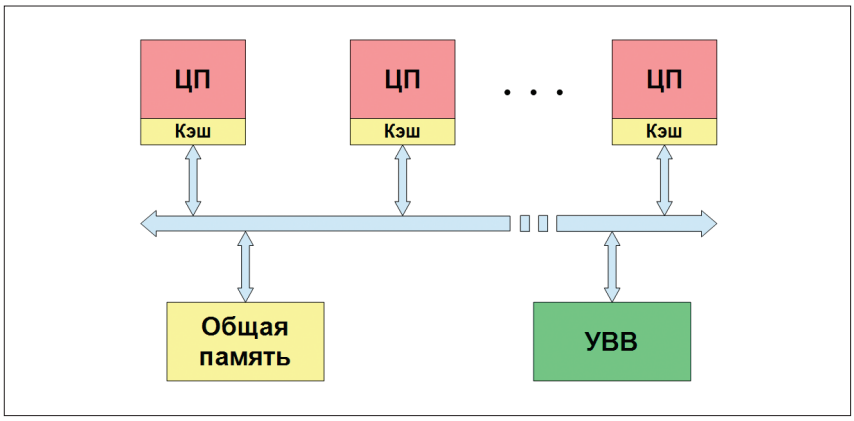

Рис. 1. **Структурная схема классической SMP-системы**

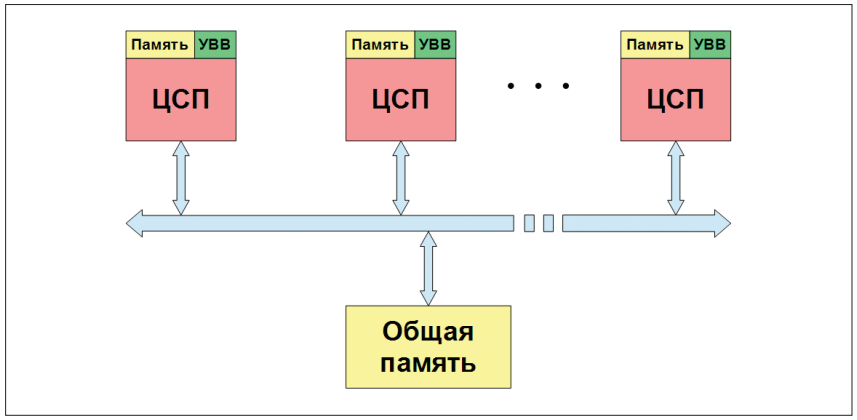

Рис. 2. **Структурная схема SMP-системы на базе сигнального процессора 1967ВН028**

составлять не более восьми единиц. Эти особенности налагают дополнительные требования к проектированию многопроцессорной системы с общей памятью и вносят в архитектуру определенный элемент асимметричности.

На рисунке 2 показана структурная схема SMP-системы на базе процессора 1967ВН028. В этой системе каждый процессор (DSP) имеет собственную внутреннюю память, устройства ввода/вывода и одинаково подключается, как и в случае с классической системой, к общей шине, связывающей его с общей памятью и другими процессорами.

Одним из основных способов для реализации работы такой нестандартной системы может послужить подход, основанный на использовании операционной системы (ОС), которая поддерживает выполнение программы в симметричном многопроцессорном режиме и предоставляет набор механизмов, обеспечивающих многопроцессорное взаимодействие. Такой ОС является операционная система реального времени для мультиагентных когерентных систем (ОСРВ МАКС, «АстроСофт») для многопроцессорных кластерных систем.

Эта операционная система предоставляет необходимый набор средств и инструментов для работы такой SMP-системы:

- POSIX Threads API (программный интерфейс приложения для стандарта POSIX-реализации потоков Pthreads);
- OpenMP API (программный интерфейс приложения, поддерживающий многопроцессорное программирование систем с общей памятью);
- PIPE API (программный интерфейс приложения механизма межпроцессорного взаимодействия);
- TRACE API (программный интерфейс приложения функций профилирования и трассировки пользовательских событий и событий операционной системы по интерфейсу JTAG или порту связи).

Главной особенностью операционной системы является то, что она работает в режиме с двумя контекстами выполнения ядра: один во внутренней памяти, другой – во внешней. Такая возможность обусловлена наличием в процессоре 1967ВН028 блока управления (sequencer) исполнения программы, позволяющего выполнять команды программы с хранением данных на стеке во внешней памяти. Контекст выполнения потоков Pthreads относится ко второму типу, т. е. такие потоки выполняются из внешней памяти и предназначены для конкурентных задач. Эти задачи мигрируют между процессорами в системе, что позволяет равномерно распределить общую нагрузку на систему. Однако для обеспечения корректного взаимодействия с устройствами ввода/вывода процессора категорически необходима возможность привязки задачи, обслуживающей данные устройства, к конкретному процессору. В операционной системе предусмотрено несколько механизмов для привязки задач к конкретному процессору:

1) привязка потока выполнения Pthread к конкретному процессору с помощью атрибута affinity; при этом контекст такого потока не изменяется (выполнение происходит по-прежнему из внешней памяти);
2) использование стандартного механизма операционной системы для создания локальных задач (tasks) с контекстом во внутренней памяти.

На взаимодействие между задачами, выполняющимися из внутренней памяти, и задачами, выполняющимися из внешней, накладываются некоторые ограничения: значение указателя на локальные данные, расположенных во внутренней памяти, нельзя использовать в исходном виде потоками на других процессорах. Для обеспечения прямого взаимодействия необходимо выполнять обмен данными только через общую память.

При разработке такой системы следует учитывать, что скорость внешней шины существенно проигрывает скорости внутренней: внешняя шина имеет разрядность 64 бит и максимальную частоту 100 МГц, внутренняя – четыре шины по 128 бит и частоту, равную частоте ядра, до 450 МГц. Это явным образом негативно отражается на пропускной способности внешней шины. Кроме того, следует учитывать, что на внешней шине находится более одного процессора. Тем самым вводятся дополнительные условия и требования при разработке программного обеспечения.

Чтобы снизить негативный эффект от частого использования внешней шины, применяется метод, когда совместно с внешней памятью активно используется внутренняя. Поскольку при миграции потока с одного процессора на другой прежняя локальная память окажется для него недоступной, требуется некоторая осторожность при таком подходе. Необходимо определить, когда и при каких задачах размещение данных для обработки во внутренней памяти действительно обеспечит необходимый прирост производительности системы, а когда данным условием можно пренебречь.

Было проведено тестирование производительности многопроцессорной системы, содержащей два процессора

Таблица 1. **Конфигурации запуска тестирования производительности системы**

Конфигурация	Количество процессоров	Используемая память
0	1	Локальная
1	2	
2	1	Общая (внешняя)
3	2	

Таблица 2. **Описание тестов производительности системы**

Тест №	Описание	Примеры
1	Большое количество тривиальных вычислений при множественном обращении к одним и тем же данным	Матричные операции, вычисление свертки, БПФ
2	Большое количество вычислений при небольшом обращении к данным	Табличные расчеты
3	Большое количество вычислений при минимальном обращении к данным	Расчет рядов числа π

Таблица 3. **Результаты тестирования производительности системы**

Конфигурация	Тест №1, мс	Тест №2, мс	Тест №3, мс
0	180	82	660
1	92	48	425
2	3780	82	657
3	3380	54	424

Таблица 4. **Относительный прирост производительности системы в процентах**

Конфигурация	Тест №1			Тест №2			Тест №3		
	0	1	2	0	1	2	0	1	2
1	+48 ,8			+41,5			+35,6		
2	–2000,0	–4008,7		0,0	–70,8		+0,5	–54,6	
3	–1777,8	–3573,9	+10,6	+34,1	–12,5	+34,1	+35,8	+35,8	+35,5

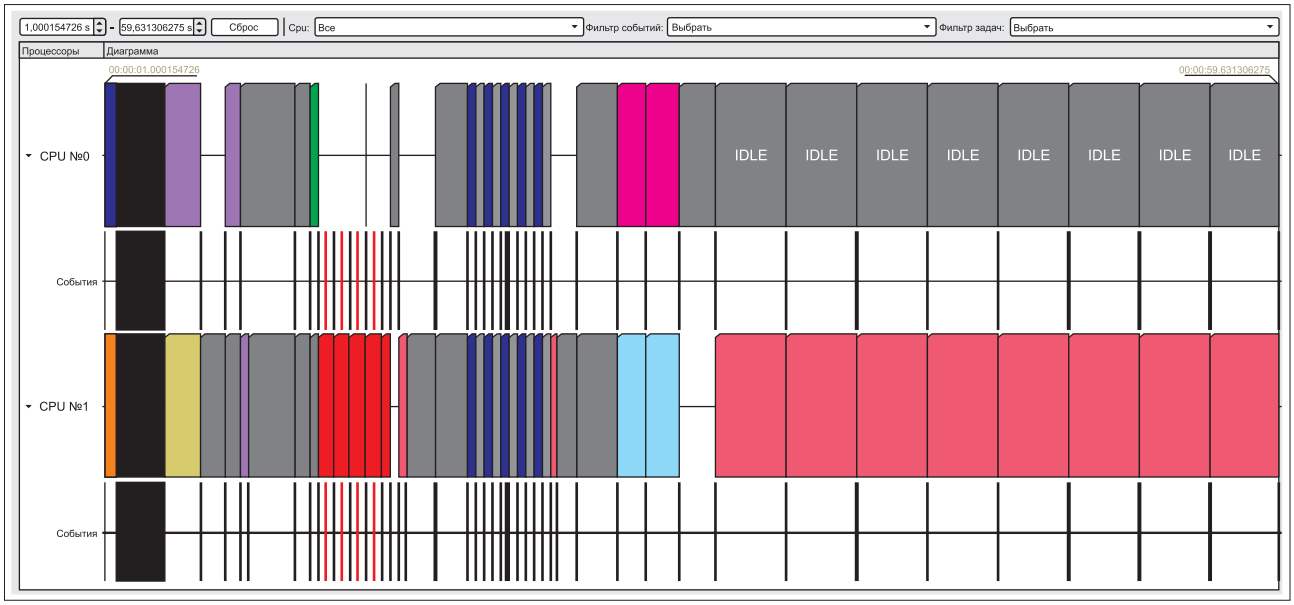

Рис. 3. **Диаграмма профилирования работы многопроцессорной системы**

1967ВН028. Тестирование проводилось при различных конфигурациях и требованиях доступа к общим исходным данным. В таблицах 1–2 приведены конфигурации запуска тестов и описание тестов, соответственно.

Результаты тестирования на производительность приведены в таблицах 3–4. По результатам первого теста можно сделать вывод, что при использовании многопроцессорного подхода для решения этой задачи почти достигается теоретический двукратный прирост производительности. Таким образом, для решения данных задач настоятельно рекомендуется размещать общие данные для обработки во внутренней памяти каждого процессора. Во втором тесте отчетливо видно схожее поведение по производительности, что и в первом тесте, достигается существенный прирост при увеличении количества процессоров. Однако размещение данных для вычислений в локальной памяти практически не влияет на прирост производительности. И, наконец, в третьем тесте поведение схоже со вторым тестом за исключением чуть более низкого показателя прироста производительности. Следовательно, при разработке аналогичной многопроцессорной системы необходимо внимательно анализировать тип решаемой задачи, специфику поведения алгоритма при работе с общими данными.

Можно привести несколько ситуаций происхождения данных. В случае, когда данные, участвующие в вычислениях, генерируются и изменяются синхронно на всех процессорах (например, поворотные коэффициенты для расчета БПФ с изменением размера окна), эти данные непременно следует размещать во внутренней памяти процессора. В случае же, когда источником данных является один из процессоров (например, данные были получены по порту связи), эти данные необходимо скопировать во внутреннюю память каждого процессора. Чтобы выполнить такую процедуру и обеспечить максимальную производительность, следует воспользоваться операцией широковещательной записи. Данный тип операции присутствует в процессоре 1967ВН028 и обеспечивает одновременное копирование данных с одного процессора на другие через общую шину в обход внешней памяти.

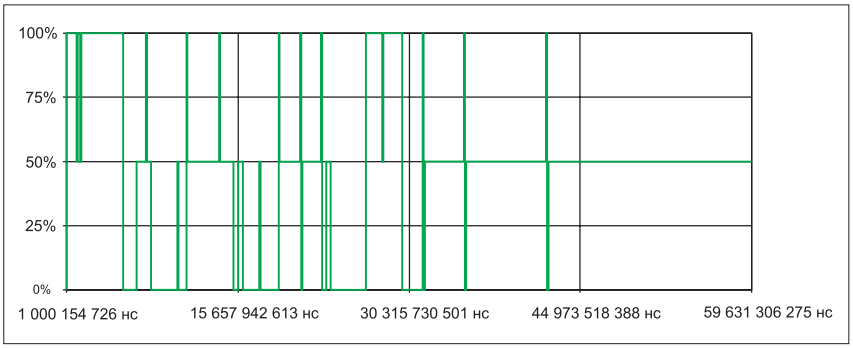

Рис. 4. **Диаграмма загруженности многопроцессорной системы**

Немаловажную роль в проектировании любых многопроцессорных систем играет наличие инструментов и средств для анализа ее работы, возможности быстрой отладки, тестирования и дальнейшей оптимизации производительности. В данном случае интегрированный в операционную систему механизм профилирования и трассировки дает такую возможность. На рисунке 3 приведена диаграмма профилирования двухпроцессорной системы с общей памятью. На ней отображены задачи, выполняющиеся на каждом процессоре, а также события, происходящие на каждом из них. Событиями могут выступать как системные сообщения (создание задачи, смена приоритета, освобождение семафора, ожидание захвата мьютекса и т.д.), так и пользовательские с произвольной информацией. На рисунке 4 показана временная диаграмма загрузки системы.

В заключение заметим, что представленный в данной статье подход проектирования многопроцессорных систем с общей памятью на базе процессора 1967ВН028 имеет свои преимущества. Во-первых, ОСРВ МАКС для многопроцессорных кластерных систем предоставляет большой набор средств и инструментов для ускорения разработки, отладки и тестирования таких. Во-вторых, обеспечивается определенный уровень масштабируемости системы. В-третьих, реализуется возможность кросс-платформенности разрабатываемого программного обеспечения. Однако из-за особенностей архитектуры процессора 1967ВН028 (наличие быстрой внутренней памяти и локальных устройств ввода/вывода) на разработчика накладываются дополнительные требования при проектировании.