УДК 004.051
DOI: 10.15827/0236-235X.119.409-419

Дата подачи статьи: 17.04.17
2017. Т. 30. № 3. С. 409–419

СРАВНЕНИЕ ПРОИЗВОДИТЕЛЬНОСТИ ОТЕЧЕСТВЕННЫХ И ИМПОРТНЫХ МИКРОПРОЦЕССОРОВ

Н.Д. Байков, младший научный сотрудник, nknikita@niisi.ras.ru;

А.Н. Годунов, к.ф.-м.н., зав. отделом, nkag@niisi.ras.ru

*(Федеральный научный центр Научно-исследовательский институт системных исследований
РАН (ФНЦ НИИСИ РАН), Нахимовский просп., 36, корп. 1, г. Москва, 117218, Россия)*

В статье сравнивается производительность трех различных процессоров архитектуры MIPS – RM7000, XLP316 и 1890BM8Я и описываются их архитектурные особенности.

Для оценки производительности процессоров при помощи языков программирования C и Assembler строится и применяется методика тестирования, включающая в себя три последовательных этапа.

На первом этапе тестирования измеряется время выполнения отдельных процессорных инструкций при условии, что эти инструкции и необходимые для их работы данные лежат в кэш-памяти первого уровня. Инструкции делятся на несколько групп, для представителей каждой из которых в статье приводятся результаты. На втором этапе измеряется эффективность работы со структурой кэш-памяти первого и второго уровней. В данной статье приводятся только результаты измерений времени обращения в кэш-память второго уровня и в основную память. На третьем этапе сравниваются полученные измерения с теоретическими оценками, построенными на основе результатов первых двух этапов тестирования. Для этого используются синтетические тесты производительности.

Достоинством методики является независимость результатов измерений от используемого компилятора и операционной системы. Все измерения производятся в тактах процессора средствами специальных регистров сопроцессора.

Ключевые слова: тестирование, MIPS, производительность, микропроцессор.

В статье сравнивается производительность процессоров архитектуры MIPS [1] – RM7000 [2], XLP316 (далее – XLP) [3] и экспериментальной версии процессора 1890BM8Я (далее – BM8) [4]. Основной целью тестирования является выявление особенностей архитектуры, оказывающих влияние на производительность процессоров, а также классов задач, на которых процессоры показывают наилучшую производительность. Результаты тестирования содержат информацию о времени выполнения как отдельных процессорных инструкций, так и некоторых модельных задач. Предлагаемая методика тестирования имеет следующие особенности:

- все измерения проводились в тактах процессора с целью получить характеристику производительности, которая не зависит от тактовой частоты;
- было изучено влияние кэш-памяти первого и второго уровней на производительность;
- все тесты задействовали только одно ядро процессора;
- при разработке использовались языки программирования Assembler и C; для их трансляции в машинный код – один и тот же компилятор на всех трех процессорах.

Многие тесты производительности (Whetstone [5], Dhrystone [6] и др.) реализованы при помощи языков программирования высокого уровня. Как следствие, на результаты измерений влияют и оптимизирующие возможности компилятора. В данной статье производительность процессоров и влияние компилятора на нее изучаются отдельно друг от друга.

Каждый из тестируемых процессоров имеет уникальную и достаточно сложную архитектуру, и,

как следствие, состояния этих процессоров в каждый момент времени определяются большим количеством различных параметров, осуществить полный учет которых достаточно трудно. В данной статье рассматривается упрощенная модель, согласно которой состояние процессора определяется содержимым кэш-памяти процессора и последовательностью исполняемых им инструкций.

Технические характеристики

Основные параметры изучаемых процессоров приведены в таблице 1. Для всех трех процессоров функциональность ALU зависит от конвейера, на котором этот блок находится. Например, операции целочисленного умножения и деления выполняются только на F-конвейере для процессоров RM7000 и BM8 и только на ALU2 для процессора XLP. На процессоре RM7000 инструкции вещественной арифметики не вынесены на отдельный конвейер и обрабатываются F- и M-конвейерами.

Методика тестирования

Тестирование процессоров проводилось в три этапа. На первом этапе измерялось время выполнения процессорных инструкций при условии, что сами инструкции и все необходимые для работы данные уже находятся в соответствующих кэшах процессоров. Тестирование охватывало четыре группы инструкций: целочисленная арифметика, вещественная арифметика, передачи управления, чтение/запись в память. В статье приводятся результаты измерений для нескольких представителей каждой из указанных групп.

Таблица 1

Технические характеристики

Table 1

Technical specifications

Характеристика	Процессор		
	RM7000	XLP	BM8
Тактовая частота процессора	400,0 МГц	1,0 ГГц	700,0 МГц
Частота памяти	66,6 МГц	800,0 МГц	400,0 МГц
Количество ядер	1	4	2
L1 – кэш инструкций (iL1)	16 Кб (4-канальный)	64 Кб (2-канальный)	32 Кб (8-канальный)
L1 – кэш данных (dL1)	16 Кб (4-канальный)	32 Кб (2-канальный)	16 Кб (4-канальный)
L2 – кэш	256 Кб (4-канальный)	512 Кб (8-канальный)	512 Кб (4-канальный)
L3 – кэш	Отсутствует	4 Мб (16-канальный)	Отсутствует
Размер строки кэша в L1	32 байта	32 байта	32 байта
Размер строки кэша в L2	32 байта	64 байта	32 байта
Конвейеры выполнения инструкций	2 конвейера: ALU1/BR/FPU, ALU2/LSU/COP0/FPU	5 конвейеров: ALU0/LSU0, ALU1/LSU1, ALU2, ALU3/BR/COP0, FPU	3 конвейера: ALU1/BR, ALU2/LSU/COP0, FPU

Примечание. В таблице приняты следующие обозначения: FPU – обработка инструкций вещественной и комплексной арифметики, ALU – целочисленной арифметики, BR – передачи управления, LSU – чтения/записи в память, COP0 – работы с управляющим сопроцессором. Для первого и второго конвейеров процессоров RM7000 и BM8 также используются обозначения F и M соответственно.

На втором этапе тестирования изучалось влияние на производительность процессоров кэшей первого и второго уровней.

На третьем этапе сравнивалась производительность процессоров на следующих модельных задачах: функция копирования данных, метод Рунге–Кутты, метод Гаусса.

Для автоматизации процесса тестирования с помощью языков программирования Assembler и C была построена тестовая система. В ее функции входили подготовка начального состояния процессора для каждого теста, а также непосредственное измерение характеристик производительности процессоров.

Перед началом тестирования содержимое всех кэшей очищалось, в том числе и содержимое кэша третьего уровня на процессоре XLP. Инструкции тестовой системы размещались в кэше инструкций первого уровня и оставались там до окончания тестирования всех функций. Для инструкций каждой тестируемой функции и данных, с которыми она работает, независимо моделировались следующие ситуации: присутствие инструкций/данных в соответствующем кэше первого уровня, присутствие только в кэше второго уровня, отсутствие в кэшах всех уровней. После окончания тестирования каждой функции ее инструкции и данные удалялись из кэшей. Таким образом, в кэшах процессора одновременно могли находиться только инструкции тестовой системы и одной из тестируемых функций, а также данные, необходимые для ее работы. Кэши изучаемых процессоров организованы так, что в них можно одновременно разместить указанные инструкции и данные вне зависимости от адресов, которые они имеют в памяти, не вызывая при этом вытеснения строк (количества кэш-каналов достаточно, чтобы при возникновении конфликтов

между адресами функций или данных можно было разместить их в разных секциях соответствующего кэша). Загрузка инструкций тестовой системы и тестируемых функций в кэши происходила при помощи вызовов этих функций, а загрузка данных – при помощи операций чтения. Для исключения данных из кэшей использовалась инструкция cache. Проведенные операции позволили минимизировать влияние тестовой системы на результаты измерений и предотвратить вытеснение строк из кэшей в связи с их переполнением.

В качестве средств измерения использовались регистры процессора Performance Counter Control и Performance Counter, которые позволяют регистрировать различные события во время работы программы, такие как такты процессора, промахи в кэш-память процессора и т.д. При каждом выполнении тестируемой функции тестовая система опрашивала счетчик до и после ее выполнения и сохраняла разницу этих значений. Анализ полученных результатов производился в зависимости от этапа тестирования. Для проведения первых двух этапов конструировались семейства однотипных тестовых функций. Функции одного семейства последовательно повторяли заданную инструкцию или набор инструкций заданное количество раз и отличались друг от друга только количеством повторений. Кроме указанных инструкций, функции также могли содержать какие-либо дополнительные инструкции, которые требуются для корректной работы теста, например, первоначальной инициализации используемых регистров. В частности, для каждого такого семейства конструировалась пустая тестовая функция, которая состояла только из дополнительных инструкций и характеризовала величину накладных расходов. Оценка интересующих авторов факторов проводилась на основе се-

рии измерений для функций, принадлежащих одному семейству. На третьем этапе тестирования такие семейства уже не требовались и измерения проводились непосредственно для программных реализаций модельных задач.

Важной технической деталью является то, что процессор XLP поддерживает внеочередное исполнение инструкций. Каждый из его конвейеров оснащен очередью, вмещающей в себя до 16 инструкций. Если в данный момент времени нет свободного конвейера, который может выполнить текущую инструкцию, она помещается в очередь к одному из них, а процессор переходит к обработке следующей инструкции. Если при этом окажется, что есть свободный конвейер для выполнения этой инструкции и она не зависит по данным от уже находящихся в очередях инструкций, процессор немедленно приступит к ее обработке, тогда как предыдущая инструкция еще будет находиться в очереди. Эта архитектурная особенность XLP позволяет избежать простоя всех конвейеров из-за высокой нагрузки на один из них. Так как указанная особенность распространяется и на конвейер, ответственный за работу с управляющим сопроцессором, повторный опрос счетчика после возврата из тестируемой функции может произойти раньше, чем будет завершена обработка всех инструкций теста, которые на данный момент находятся в очереди другого конвейера, то есть результат измерений будет некорректным. Эту проблему удастся решить. Заметим, что только один из конвейеров процессора XLP способен обрабатывать инструкции работы с управляющим сопроцессором, то есть эти инструкции сохраняют порядок относительно друг друга. Добавляя загрузку результата работы тестируемой функции на один из временных регистров управляющего сопроцессора перед повторным опросом счетчика, формируем зависимость по данным между инструкциями и гарантируем, что опрос произойдет после вычисления этого результата. Наконец, для корректности итоговых результатов прерывания во время работы каждого теста были запрещены.

Результаты измерений

Рассмотрим процессорные инструкции. Прежде чем перейти к результатам измерений, необходимо уточнить, что будем понимать под длительностью выполнения инструкций. Тестируемые процессоры используют конвейерный способ обработки инструкций. Полное время прохождения инструкцией всех стадий конвейера не дает точного представления о производительности процессора, поскольку не учитывает параллелизм в выполнении инструкций. Более точно оценить вклад каждой отдельной инструкции в общее время выполнения программы позволяют следующие величины [7]:

- время освобождения блока выполнения конвейера (Repeat Rate), то есть количество тактов от

момента начала стадии выполнения инструкции до момента, когда конвейер сможет начать эту стадию для следующей инструкции;

- задержка результата выполнения инструкции (Latency), то есть количество тактов от момента начала стадии выполнения инструкции до момента, когда для следующих инструкций будет доступен результат выполнения.

Также существует понятие пропускной способности конвейера (Pipeline Throughput). На наборе из n инструкций это отношение количества инструкций n к полному времени их прохождения по конвейеру.

Стадия выполнения для инструкции может быть начата на подходящем свободном конвейере только в том случае, если доступны результаты всех инструкций, от которых она зависит. В противном случае возникает простой конвейера до получения необходимых данных. Для многих инструкций задержка результата оказывается больше времени освобождения блока выполнения. Поскольку при выполнении реальных задач нет никаких гарантий того, что результат работы таких инструкций не понадобится уже в следующей инструкции, для оценки производительности процессоров на первом этапе тестирования больше внимания уделяется измерению задержки результата. Для этого достаточно сделать инструкции тестируемых функций зависимыми.

Описание результатов начнем с инструкций целочисленной арифметики. Приведем результаты для трех инструкций из этой группы: сложение с постоянным значением `addi`, умножение `mul` и целочисленное деление при помощи пары инструкций `div` и `mflo`. Инструкция `mflo` считывает результат целочисленного деления `div` из специального LO-регистра, создавая зависимость по данным между соседними инструкциями `div`. Для других операций, например, выполняемых при помощи инструкций `andi`, `add`, `sub` и т.д., тестирование проводится аналогично.

Структура всех тестов одинакова и состоит из этапа инициализации, последовательности повторяющихся операций и возврата управления. Приведем пример исходного кода:

```
li v0, C1; // Инициализация v0
addi v0, v0, C2;
... // Повторяем вышестоящую инструкцию нужное
... // количество раз вручную, не используя
... // передачи управления
jr ra;
nop;
```

В таблице 2 указана продолжительность тестов в тактах процессора в зависимости от количества повторений выбранной операции целочисленной арифметики (где 0 соответствует пустой функции). Все указанные результаты воспроизводятся при повторном тестировании. Отметим, что время выполнения операции деления при помощи пары инструкций `div` и `mflo` не на всех процессорах является фиксированным. Здесь приведены результаты

Таблица 2

Инструкции целочисленной арифметики (такты)

Table 2

Integer arithmetic instructions (clock cycles)

Инструкция	Процессор	Количество повторений																
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
addi	RM7000	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
	XLP	29	33	33	33	33	34	35	36	37	38	39	40	41	42	43	44	45
	BM8	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
mul	RM7000	6	8	11	14	17	20	23	26	29	32	35	38	41	44	47	50	53
	XLP	34	33	35	38	41	44	47	50	53	56	59	62	65	68	70	73	76
	BM8	13	14	16	18	20	22	24	26	28	30	32	34	36	38	40	42	44
div + mflo	RM7000	6	44	84	124	164	204	244	284	324	364	404	444	484	524	564	604	644
	XLP	29	44	59	73	87	100	113	125	136	148	161	173	185	197	209	221	233
	BM8	13	24	34	44	53	62	70	78	101	124	147	170	193	216	239	262	285

частного случая начальных значений регистров v0 и t0.

На процессоре RM7000 зависимость количества тактов от количества операций линейна для инструкции addi, а для инструкции mul и пары инструкций div, mflo почти линейна, за исключением переходов от пустых функций. Действительно, с увеличением на единицу количества повторений addi длительность теста возрастает ровно на 1 такт во всех случаях. Для инструкции mul и пары инструкций div, mflo при переходе от пустых функций длительность теста возрастает на 2 и 38 тактов процессора соответственно, а при дальнейшем увеличении количества операций – на 3 и 40 тактов за каждое повторение соответствующей операции. Отличие первого приращения от остальных указывает на то, что освобождение блока выполнения от mul и mflo происходит быстрее, чем будет доступен их результат (для возврата управления их результат не требуется). Также было обнаружено, что на процессоре RM7000 задержка получения результата инструкции mul зависит от размера аргументов. Для 16-битных аргументов она равнялась 3 тактам, а для 32-битных – 4 тактам. На других процессорах подобной зависимости не наблюдалось.

Результаты процессоров BM8 и XLP аналогичны результатам процессора RM7000. Укажем лишь на некоторые особенности. На процессоре BM8 полученные значения задержки результата для инструкций addi и mul составили 1 и 2 такта соответственно, а задержка результата выполнения пары инструкций div и mflo варьировалась от 8 до 23 тактов.

На процессоре XLP полученные результаты проявляют линейную зависимость лишь после нескольких повторений операции. Для инструкций addi и mul наблюдаемые значения задержки составили 1 и 3 такта. Задержка на паре div и mflo, как и в случае процессора BM8, не была фиксированной и требовала от 11 до 15 тактов. Большая величина накладных расходов (то есть время выполнения пустой функции) на процессоре XLP по сравнению с остальными процессорами частично объясняется

влиянием на результат дополнительных операций с управляющим сопроцессором для предотвращения преждевременного опроса счетчика событий.

Таким образом, на операциях целочисленного сложения и деления лучшую производительность в тактах процессора показал процессор XLP, а на умножении – BM8.

Аналогичные тесты были проведены для инструкций сопроцессора вещественной арифметики. Тесты отличались только способом инициализации исходных регистров. Рассмотрим результаты для четырех видов инструкций, работающих с числами формата double: сложение add.d, умножение mul.d, деление div.d и смешанная операция умножения и сложения madd.d.

На процессоре XLP результаты измерений не воспроизводились. Уже в случае пустой функции результат измерений мог меняться от запуска к запуску. Например, для 5 повторений инструкции add.d в 50 запусках теста результат измерений изменялся от 57 до 116 тактов. На процессорах RM7000 и BM8 колебаний при измерениях не наблюдалось. Результаты этой группы тестов приведены в таблице 3 (для процессора XLP выбраны результаты одного из запусков).

Оценивать увеличение длительности теста при увеличении количества повторений инструкций на единицу на процессоре XLP будем в среднем. Получаем приближенные значения 5,9, 6,1, 36,2 и 10,2 такта для инструкций add.d, mul.d, div.d и madd.d, соответственно. На процессорах RM7000 и BM8 искомые величины могут быть вычислены точно. Не считая пустых функций, длительности тестов на указанных процессорах линейно зависят от количества инструкций. На процессоре RM7000 длительность теста увеличивается на 4, 5, 36 и 5 тактов за каждую инструкцию add.d, mul.d, div.d и madd.d соответственно, а на процессоре BM8 – на 3, 3, 19 и 5 тактов. Для тестируемых инструкций вещественной арифметики освобождение блока выполнения конвейера происходит не позже, чем будет доступен результат работы инструкций, поэтому полученные значения совпадают с временем задержки

Таблица 3

Инструкции вещественной арифметики (такты)

Table 3

Floating point arithmetic instructions (clock cycles)

Инструкция	Процессор	Количество повторений																
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
add.d	RM7000	9	14	18	22	26	30	34	38	42	46	50	54	58	62	66	70	74
	XLP	53	85	47	83	70	100	85	91	104	127	98	122	114	131	133	136	148
	BM8	17	21	24	27	30	33	36	39	42	45	48	51	54	57	60	63	66
mul.d	RM7000	9	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90
	XLP	71	45	92	56	70	64	110	80	84	103	93	118	133	135	126	135	168
	BM8	17	21	24	27	30	33	36	39	42	45	48	51	54	57	60	63	66
div.d	RM7000	9	46	82	118	154	190	226	262	298	334	370	406	442	478	514	550	586
	XLP	83	75	157	156	205	226	281	304	340	397	437	455	516	530	571	614	663
	BM8	17	37	56	75	94	113	132	151	170	189	208	227	246	265	284	303	322
madd.d	RM7000	10	16	21	26	31	36	41	46	51	56	61	66	71	76	81	86	91
	XLP	52	93	72	68	91	96	110	127	139	147	166	169	194	191	197	205	215
	BM8	17	24	29	34	39	44	49	54	59	64	69	74	79	84	89	94	99

результата этих инструкций. Таким образом, наименьшие значения задержки результата для группы инструкций вещественной арифметики наблюдались на процессоре BM8.

В отличие от тестов целочисленной арифметики переход от пустых функций длится дольше всех последующих переходов. Причиной оказываются инструкции dmtcl, для которых задержка результата выполнения больше времени освобождения блока выполнения конвейера. В ходе отдельных тестов было получено, что для освобождения блока выполнения конвейера от инструкции dmtcl на процессорах RM7000 и BM8 требуется 1 такт, а задержка результата равна 2 и 3 тактам соответственно.

Перейдем к тестированию инструкций передачи управления. Эффективность выполнения этих инструкций существенно зависит от используемых алгоритмов предсказания переходов и может меняться в зависимости от ситуации. Оценка эффективности проводилась в четырех тестах: простой цикл, вложенные передачи управления, перекрест-

ные передачи управления и безусловные передачи управления.

Результаты тестов представлены в таблице 4. Лучшие результаты в этих тестах показал процессор RM7000, которому требуется ровно 2 такта на выполнение одной передачи управления, тогда как процессорам XLP и BM8 требуется до 5 тактов процессора. Отдельно можно отметить только результаты процессора BM8 в первом тесте, которому после нескольких начальных итераций требуется только 1 такт на выполнение передачи управления.

Наконец, перейдем к изучению инструкций чтения/записи. Тест инструкций чтения требовал дополнительной подготовки. Буфер, из которого считывались данные, заполнялся двойными словами, каждое из которых содержало адрес следующего. Это позволило сформировать зависимость по данным между инструкциями, которая вынуждает процессор XLP выполнять инструкции последовательно.

Результаты обоих тестов воспроизводятся точно на процессорах RM7000 и BM8, а на процес-

Таблица 4

Инструкции передачи управления (такты)

Table 4

Branch instructions (clock cycles)

Тест	Процессор	Количество повторений																
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Тест 1	RM7000	8	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39
	XLP	14	27	29	31	34	35	37	39	41	43	45	47	49	51	53	55	57
	BM8	19	19	28	31	32	33	34	35	36	37	38	39	40	41	42	43	44
Тест 2	RM7000	9	13	17	21	25	29	33	37	41	45	49	53	57	61	65	69	73
	XLP	17	32	42	50	60	66	74	82	90	98	106	114	122	130	138	146	154
	BM8	25	35	45	55	65	75	85	95	105	115	125	135	145	155	165	175	185
Тест 3	RM7000	8	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39
	XLP	15	18	45	49	33	62	37	39	67	43	45	47	49	51	53	55	57
	BM8	19	25	28	34	37	43	46	52	55	61	64	70	73	79	82	88	91
Тест 4	RM7000	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37
	XLP	14	16	22	24	30	32	38	40	46	48	54	58	62	64	70	74	78
	BM8	12	15	20	23	28	31	36	39	44	47	52	55	60	63	68	71	76

соре XLP наблюдаются редкие и незначительные по величине (в пределах 1-2 тактов) колебания. Результаты измерений приведены в таблице 5.

На процессоре XLP была получена наибольшая величина накладных расходов. На инструкциях записи происходит скачок длительности теста при переходе от пустой функции к функции с одной инструкцией записи и далее в среднем длительность теста возрастает приблизительно на 1 такт процессора за каждую дополнительную инструкцию записи. На инструкциях чтения прирост длительности теста составил 4 такта за каждую инструкцию.

Перейдем к результатам процессоров RM7000 и BM8. На инструкциях чтения процессоры показали одинаковый результат: длительность теста возрастает на 2 такта процессора за каждую инструкцию чтения. При тестировании инструкций записи было обнаружено, что результаты RM7000 и BM8 уже не совпадают. На каждую инструкцию записи процессору RM7000 требуется ровно один такт, тогда как на процессоре BM8 зависимость количества тактов от количества инструкций становится нелинейной и в среднем требует более одного такта процессора.

Часто при выполнении инструкции чтения соседние инструкции не зависят от ее результата. Производительность в этом случае определяется временем освобождения блока выполнения конвейера. Измерения показали, что на это требуется 1 такт на всех трех процессорах. Также отметим, что на процессоре XLP имеются сразу два конвейера, способных обрабатывать инструкции чтения/записи, что также позволяет ускорить работу процессора.

Время обращения к кэш-памяти

Хотя влияние кэш-памяти на производительность складывается из многих факторов, ограничимся лишь измерением времени обращения к ней. Влияние других факторов (например, политики кэширования) изучаться не будет. Также отметим, что измеряемая величина не является только характеристикой производительности процессора, так как зависит также от используемой системы памяти.

В процессе тестирования существенных различий в длительности получения данных или инструкций из памяти обнаружено не было. Поэтому рассмотрим только результаты тестов для случая работы с инструкциями. Для измерений использовалось семейство тестовых функций из первого этапа тестирования для инструкции `addi`. При их работе чтение данных из памяти не происходит. Были рассмотрены два случая.

1. Инструкций нет в кэше инструкций первого уровня `iL1`, но они есть в кэше второго уровня `L2`.

2. Инструкций нет в кэшах `iL1` и `L2`. В случае процессора XLP инструкции также отсутствуют в кэше `L3`.

Поскольку инструкции и данные попадают в кэши строками, количество обращений из кэша `iL1` в кэш `L2` определяется количеством строк кэша `iL1`, требуемых для хранения функции. Тестовые функции задавались так, чтобы начало каждой из них соответствовало началу строки в кэшах первого и второго уровней. В таком случае количество требуемых для хранения функции строк кэша однозначно определяется по количеству инструкций в ней. Размер строки кэша первого уровня у всех трех процессоров одинаков – 32 байта (см. табл. 1). Поскольку на хранение одной инструкции выделяются 4 байта, в одной строке кэша помещаются ровно 8 инструкций, следовательно, для получения количества строк кэша необходимо округлить вверх результат деления на 8 количества инструкций, из которого состоит тестируемая функция.

Результаты первого случая приведены в таблице 6. На процессорах RM7000 и BM8 при переходе от 5 повторений инструкции `addi` к 6 повторениям происходят скачки в длительности тестов. Далее подобные скачки повторяются с периодом в 8 инструкций `addi`. Учитывая то, что, помимо инструкций `addi`, в теле тестовых функций присутствуют ровно 3 дополнительные инструкции (`li`, `jr` и `por`), получаем, что скачки происходят одновременно с увеличением количества строк кэша `iL1`, требуемых для хранения тестовой функции. На процессоре RM7000 зависимость монотонна, а величина скачка равна 9 тактам процессора. На процессоре BM8 монотонность нарушается. Тем не менее, скачок при увеличении количества строк кэша

Инструкции чтения/записи (такты)

Load/store instructions (clock cycles)

Инструкция	Процессор	Количество повторений																
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
sd	RM7000	7	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
	XLP	54	61	61	64	63	66	66	69	68	71	71	73	73	76	76	77	78
	BM8	18	19	20	22	23	25	26	28	29	31	32	34	35	37	38	40	41
ld	RM7000	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39
	XLP	49	53	57	60	64	68	72	76	80	84	88	92	96	100	104	108	112
	BM8	18	19	21	23	25	27	29	31	33	35	37	39	41	43	45	47	49

Таблица 5

Table 5

на процессоре VM8 оценим величиной в 4-5 тактов. На процессоре XLP наблюдаются пары скачков при переходе от 13 к 14 повторениям и от 14 к 15 повторениям и далее, как показало тестирование, с периодом в 16 инструкций. Размер скачков оценивается величиной в 5-6 тактов процессора.

Перейдем ко второму случаю. Его результаты приведены в таблице 7. Для процессора VM8 результаты отсутствуют, так как оценить длительность обращения в память в рамках используемой модели на данном процессоре не удалось. Анализ результатов проводится аналогично первому случаю. Отметим только, что на процессоре XLP размер строки кэша L2 равен 64 байтам. Как видно из таблицы 7, скачки в длительностях тестов на процессорах RM7000 и XLP происходят одновременно с увеличением количества строк кэша L2, требуемых для хранения функции. Размер этих скачков приблизительно равен 110–120 тактам для обоих процессоров.

В процессе тестирования для процессора XLP также были выявлены две важные особенности, которые позволяют существенно ускорить его работу при промахах в кэш-память процессора. Было обнаружено, что процессор XLP способен обрабатывать параллельно промахи в кэш-память для различных инструкций из очереди выполнения при условии, что известны адреса данных в памяти. Если требуется, например, последовательно прочитать данные из буфера, указанная особенность позволяет добиться существенного прироста к производительности. Также было обнаружено, что при выполнении инструкций запись обращений в память не происходит. Вместо этого процессор XLP помещает записываемые данные в специальную очередь на запись, из которой они впоследствии попадают в основную память. При этом запись данных в очередь осуществляется почти так же быстро, как если бы данные находились в кэше данных первого уровня.

Модельные задачи

Представим результаты измерений для трех модельных задач: функция копирования данных, метод Рунге–Кутты и метод Гаусса [8].

Начнем с задачи копирования данных из одного буфера в другой (назовем их Src и Dst) при помощи функции memcpy, поддерживаемой стандартами POSIX и C99. Адреса буферов в памяти были выровнены на границу строк кэшей первого и второго уровней, а кэши данных очищены перед началом тестирования. Существует несколько реализаций функции копирования (memcpy); в данном случае использовалась реализация, которая перемещает 4 двойных слова из буфера Src в буфер Dst за одну итерацию основного цикла. Инструкции этой функции были предварительно размещены в кэше инструкций первого уровня. Рассматривались не перекрывающиеся в памяти буферы размером от 1 Кб до 8 Мб в следующих случаях:

- данные обоих буферов отсутствуют в кэшах;
- буфер Src предварительно прочитан (то есть, если размер кэша больше размера буфера, весь буфер находится в этом кэше после прочтения);
- буфер Dst предварительно прочитан;
- оба буфера были прочитаны перед началом теста.

Результаты измерений представлены в таблице 8 (вместо общей длительности теста указывается время копирования 1 Кб). На RM7000 полученные данные хорошо согласуются с результатами первых этапов тестирования. Действительно, при отсутствии буферов в кэшах время копирования, главным образом, определяется количеством обращений в основную память. Количество таких обращений можно вычислить явно. Поскольку размер строки кэша второго уровня на процессоре RM7000 равен 32 байтам, а адрес используемых буферов выровнен на границу строки кэша, для хранения 1 Кб в кэш-памяти второго уровня на процес-

Таблица 6

Длительность обращения в кэш L2 (такты)

Table 6

The L2 cache access duration (clock cycles)

Процессор	Количество повторений																
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
RM7000	13	14	14	15	16	17	26	25	26	27	27	28	29	30	39	38	39
XLP	44	45	46	47	48	49	50	51	52	53	54	55	56	57	62	68	68
VM8	17	18	19	20	21	22	32	27	28	29	30	31	32	33	43	38	39

Таблица 7

Длительность обращения в память (такты)

Table 7

The memory access duration (clock cycles)

Процессор	Количество повторений																
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
RM7000	115	116	116	117	118	119	229	228	229	230	230	231	286	233	343	342	343
XLP	161	151	163	164	165	166	167	168	158	159	171	161	174	163	281	275	275

Таблица 8

Копирование памяти (такты/Кб)

Table 8

Memory copying (clock cycles/Kb)

Бу- фер	Процес- сор	Размер буфера											
		4Кб	8Кб	16Кб	32Кб	64Кб	128Кб	256Кб	512Кб	1Мб	2Мб	4Мб	8Мб
Src(-) Dst(-)	RM7000	7 773	7 776	7 776	7 776	7 776	7 780	8 654	9 092	9 312	9 422	9 477	9 504
	XLP	809	791	797	806	804	803	804	808	811	812	850	875
	BM8	1 666	1 702	1 689	1 676	1 676	1 668	1 671	1 911	2 032	2 091	2 240	2 263
Src(+) Dst(-)	RM7000	4 099	4 101	4 229	4 352	4 417	4 448	7 007	8 270	8 901	9 216	9 374	9 453
	XLP	233	224	222	223	276	302	359	431	478	503	663	806
	BM8	1 195	1 189	1 093	1 040	1 009	1 003	990	1 630	1 861	2 120	2 200	2 189
Src(-) Dst(+)	RM7000	4 093	4 098	4 280	4 378	4 430	4 454	7 011	8 271	8 901	9 216	9 374	9 453
	XLP	838	800	807	800	796	793	799	804	807	808	844	875
	BM8	983	961	988	1 003	1 005	1 011	1 010	1 579	1 956	2 122	2 202	2 243
Src(+) Dst(+)	RM7000	289	300	643	849	953	1 011	5 269	7 400	8 466	8 999	9 265	9 398
	XLP	230	225	224	276	298	311	332	449	484	504	708	798
	BM8	451	451	561	616	644	658	666	1 457	1 781	2 078	2 181	2 231

соре RM7000 требуются 32 строки. Поскольку работа идет сразу с двумя буферами, количество строк необходимо удвоить. Длительность чтения 1 строки из основной памяти на процессоре RM7000 оценивалась величиной приблизительно в 110 тактов процессора, отсюда получаем, что только для 64 обращений к основной памяти без учета времени выполнения инструкций потребовалось бы приблизительно 7 040 тактов. Полученная теоретическая оценка близка к данным таблицы 8 для буферов малых размеров. При увеличении размера копируемых буферов до 256 Кб начинается рост результатов измерений, связанный с переполнением кэша второго уровня и необходимостью вытеснения строк из кэшей.

Изучим также результаты процессора RM7000 для случая, когда оба буфера находятся в кэше данных первого уровня. В таком случае инструкции записи и независимые инструкции чтения, как показало тестирование на первом этапе, должны выполняться за 1 такт процессора. Для передачи управления требуются 2 такта. Учитывая, что передачи управления и инструкции целочисленной арифметики могут быть выданы на выполнение одновременно с инструкциями чтения/записи, получаем, что копирование 4 двойных слов можно осуществить за 9 тактов. Следовательно, на копирование 1 Кб требуется 288 тактов. Соответствующий результат измерений из таблицы 8 равен 289 тактам, то есть практически совпадает с предложенной оценкой.

Скачки в измерениях для процессоров XLP и BM8, как и в случае процессора RM7000, объясняются переполнением какого-либо из кэшей. Из данных таблицы 8 следует, что процессор BM8 в большинстве случаев показал большую производительность, чем процессор RM7000, но лучшие результаты в этом тесте были получены при тестировании процессора XLP, что особенно заметно при копировании буферов большого размера. Сравним результаты процессора XLP и процессора RM7000

в случае копирования 4 Кб при полном отсутствии данных в кэш-памяти. Хотя на втором этапе тестирования было получено, что количество тактов для обращения в основную память на рассматриваемых процессорах приблизительно совпадает, процессор XLP обладает рядом преимуществ, в числе которых вдвое больший размер строки кэша второго уровня, отсутствие обращений к основной памяти на инструкциях записи и параллельная обработка промахов в кэш-память. Поскольку процессор XLP обладает двумя LSU-блоками, в очередях на выполнение может находиться до 32 инструкций чтения/записи. В рассматриваемом случае этого достаточно для одновременного копирования 16 двойных слов, то есть двух строк кэша второго уровня на процессоре XLP. Таким образом, скорость копирования данных на процессоре XLP должна быть как минимум в 8 раз выше скорости копирования на процессоре RM7000, что подтверждают результаты измерений.

В случае работы с некешируемой памятью скорость копирования данных вне зависимости от общего размера буферов составила приблизительно 19 000, 36 250 и 5 800 тактов/Кб на процессорах RM7000, XLP и BM8 соответственно.

В качестве другой модельной задачи был рассмотрен метод Рунге–Кутты 4-го порядка точности. Количество обращений к памяти в этой задаче мало. Промежуточные результаты вычислений не сохранялись, что также снизило количество обращений к памяти. Метод Рунге–Кутты применялся к осциллятору Ван-дер-Поля, который задается следующей системой уравнений:

$$\begin{cases} \frac{dx}{dt} = y, \\ \frac{dy}{dt} = \mu(1 - x^2)y - x, \end{cases}$$

где x – координата точки; $y = dx/dt$ – скорость точки; μ – некий коэффициент. Уравнение рассматривалось на фиксированном по времени отрезке при

разном количестве шагов по времени. Для программной реализации поставленной задачи использовался язык программирования C. В данном случае изучалась работа функций, построенных с использованием ключей компилятора -O0 и -O3, то есть при отсутствии оптимизации и с высоким уровнем оптимизации. На всех трех процессорах использовался один и тот же компилятор и, как следствие, один и тот же объектный код. Все инструкции и данные, необходимые для работы программы, предварительно размещались в соответствующих кэшах процессоров.

Из таблицы 9 видно, что при отсутствии оптимизации полученные результаты на всех трех процессорах достаточно близки. Лучшую производительность в тактах процессора в этом случае показал процессор XLP. Использование оптимизации приводит к сокращению времени выполнения на всех тестируемых процессорах. Лучшую производительность в тактах процессора в этом случае показал уже процессор BM8.

Сравним объектный код программ, полученных при разных уровнях оптимизации. Так как каждый шаг метода Рунге–Кутты требует одних и тех же операций, в обоих случаях программа имеет структуру цикла. В зависимости от уровня оптимизации в цикле используются различные наборы инструкций. Укажем количество повторов каждой из них.

С ключом -O0

dmtcl – 156, ld – 94, dmfc1 – 68, mul.d – 34,
add.d – 24, sdc1 – 24, lui – 18, addiu – 17,
nop – 17, move – 16, sd – 10, jr – 8,
jal – 8, sub.d – 8, ldc1 – 4, lw – 3,
div.d – 2, sw – 2, bnez – 1, slti – 1.

С ключом -O3:

mul.d – 12, add.d – 10, madd.d – 8,
msub.d – 4, sub.d – 4, bnez – 1, addiu – 1.

За счет оптимизации количество используемых инструкций сократилось на порядок (с 515 до 40 инструкций). Компилятору удалось избавиться от инструкций перемещения данных между регистрами, обращений к памяти и большей части передач управления. Также сократилось количество инструкций вещественной арифметики.

Из таблицы 9 получаем, что при отсутствии оптимизации на выполнение одной итерации цикла требуется приблизительно 854 такта на процессоре RM7000, 706 тактов на XLP и 792 такта на BM8.

Так как все необходимые инструкции и данные уже находятся в соответствующих кэшах первого уровня, длительность теста определяется эффективностью выполнения отдельных процессорных инструкций, а также выигрышем от одновременного выполнения независимых инструкций на нескольких конвейерах. Последний в данном случае можно оценить. Сформулируем несколько утверждений относительно структуры программы:

- каждая инструкция вещественной арифметики зависит от пары инструкций dmtcl, которые загружают ее аргументы на регистры сопроцессора вещественной арифметики;
- результат каждой инструкции вещественной арифметики выгружается на регистр общего назначения при помощи инструкции dmfc1, то есть выполнение dmfc1 не начнется, пока не станет доступным результат вычислений;
- в слотах задержки всех инструкций передачи управления расположены инструкции por;
- для большей части остальных инструкций время освобождения блока выполнения конвейера происходит через 1 такт процессора, а дополнительных простоев из-за задержки результата не возникает.

Предположим, что все инструкции программы выполняются последовательно, если только это не инструкции передачи управления, параллельно с которыми могут выполняться только их слоты задержки. Тогда, учитывая результаты первого этапа тестирования и сделанные утверждения о структуре программы, можно формально оценить длительность такого выполнения одной итерации цикла. На процессоре RM7000 получаем:

Передачи управления:	17×2
Инструкции mul.d:	+34×5
Инструкции add.d:	+24×4
Инструкции sub.d:	+8×4
Инструкции div.d:	+2×36
Простой на dmtcl:	+(34+24+8+2)×(2-1)
Прочие инструкции:	+(515-17×2-34-24-8-2)×1
	= 885 тактов.

Разница между полученной оценкой и результатом измерений составила всего 31 такт процессора, следовательно, использование нескольких конвейеров на процессоре RM7000 практически не дает прироста к производительности из-за большого количества зависимостей по данным между соседними инструкциями. Аналогичные результаты по-

Метод Рунге–Кутты (такты)

Таблица 9

The Runge–Kutta method (clock cycles)

Table 9

Количество шагов	Ключ -O0			Ключ -O3		
	RM7000	XLP	BM8	RM7000	XLP	BM8
256	218 780	181 281	202 995	24 455	33 524	19 091
512	437 404	362 686	405 747	48 779	67 068	38 063
1,024	874 652	722 888	811 227	97 415	134 128	75 923

Таблица 10

Метод Гаусса (такты)

Table 10

The Gauss method (clock cycles)

Размер матрицы	Ключ -O0			Ключ -O3		
	RM7000	XLP	BM8	RM7000	XLP	BM8
32	1 120 314	856 320	1 218 764	330 119	161 709	233 262
64	8 502 498	6 221 187	9 330 940	2 361 472	1 043 660	1 652 070
128	66 270 288	48 281 943	72 851 443	17 711 872	7 999 142	12 369 516
256	610 119 486	377 854 617	575 599 138	267 932 792	60 719 264	73 889 891

Таблица 11

Результаты тестирования (такты)

Table 11

Test results (clock cycles)

Инструкция	RM7000	XLP	BM8
addi	1	1	1
mul	3–4	3	2
div + mflo	40	11–15	8–23
add.d	4	5,9	3
mul.d	5	6,1	3
div.d	36	36,2	19
madd.d	5	10,2	5
dmtcl	2	–	3
Передачи управления	2	2–4	1–5
sd	1	1	1–2
ld	2	4	2
Обращение в L2	9	5–6	4–5
Обращение в память	110–120	110–120	–
Копирование 1 Кб (данные находятся в кэше dL1)	289	229	451
Копирование 1 Кб (данные отсутствуют в кэш-памяти)	7 773–9 504	809–891	1 666–2 263
Итерация метода Рунге–Кутты (-O3)	95	130	74
Метод Гаусса (n=128, -O3)	17 711 872	7 999 142	12 369 516

лучаются на процессоре BM8. Для процессора XLP разница между такой оценкой и реальным результатом оказывается уже значительной, что, вероятно, является следствием особенностей его архитектуры, которые позволяют изменять порядок выполнения инструкций и за счет этого добиваться лучших результатов, несмотря на большее по сравнению с остальными процессорами время выполнения отдельных типов инструкций.

При высоком уровне оптимизации, как следует из таблицы 9, для выполнения одной итерации цикла на процессорах RM7000, XLP и BM8 требуется 95, 130 и 74 такта соответственно. Отметим, что процессор XLP в этом случае оказался медленнее RM7000 и BM8, а это хорошо согласуется с результатами первого этапа тестирования для инструкций вещественной арифметики.

Перейдем к результатам третьей задачи. В ней изучалась производительность процессоров при решении системы линейных уравнений методом Гаусса с выбором главного элемента по столбцу. Он уже содержит не только большое количество вычислений, но и большое количество обращений к памяти. Результаты измерений для этого теста приведены в таблице 10. Аналогично методу Рунге–Кутты рассматриваются результаты для двух уровней оптимизации.

Наибольшую производительность в этом тесте показал процессор XLP в случаях как отсутствия оптимизации, так и ее высокого уровня. Результаты процессора BM8 при высоком уровне оптимизации оказались близкими к результатам процессора XLP.

Заключение

Подведем итоги проведенного тестирования. Разработанная тестовая система показала свою эффективность на практике. За исключением нескольких экспериментов, была измерена длительность выполнения многих низкоуровневых операций. Благодаря этой информации обнаружено несколько технических особенностей архитектуры, влияющих на производительность при выполнении модельных задач. Основные результаты измерений (в тактах процессора) отражены в таблице 11.

Проведенные измерения также показали, что среди процессоров невозможно однозначно выбрать лидера по производительности. При определенных условиях каждый из трех процессоров может показать лучшую производительность, чем два других. Например, вычислительные задачи с небольшой долей обращений к памяти быстрее всего выполняются на процессоре BM8, даже если при-

нять во внимание тактовые частоты процессоров. Для копирования больших объемов данных, напротив, лучше всего подходит процессор XLP, на котором большая длительность загрузки строк в кэш-память компенсируется увеличением их размера и их одновременной загрузкой.

Литература

1. MIPS Architecture For Programmers, vol. III: The MIPS64 and microMIPS64 Privileged Resource Architecture, Revision 5.04, 2014.
2. PMC-Sierra. URL: <https://www.pmcs.com> (дата обращения: 23.03.2017).

3. NetLogic Microsystems. URL: <https://www.netlogicmicro.com> (дата обращения: 23.03.2017).
4. Бобков С.Г. Импортзамещение элементной базы вычислительных систем // Вестн. РАН. 2014. Т. 84. № 11. С. 1010–1016.
5. Curnow H.J. and Wichman B.A. a synthetic benchmark. *Computer Jour.*, 1976, vol. 19, iss. 1, pp. 43–49.
6. Reinhold P. Weicker. Dhrystone: a synthetic systems programming benchmark. *Communications of the ACM*, 1984, vol. 27, iss. 10, pp. 1013–1030, DOI: 10.1145/358274.358283.
7. Zargham M.R. *Computer architecture: single and parallel systems*. Upper Saddle River, NJ, Prentice Hall, 1996, 472 p.
8. Бахвалов Н.С., Жидков Н.П., Кобельков Г.М. Численные методы. М.: Бином. Лаборатория знаний, 2003. 632 с.

Software & Systems
DOI: 10.15827/0236-235X.119.409-419

Received 17.04.17
2017, vol. 30, no. 3, pp. 409–419

COMPARISON OF RUSSIAN AND FOREIGN MICROPROCESSOR PERFORMANCE

N.D. Baykov¹, Junior Researcher, nknikita@niisi.ras.ru

A.N. Godunov¹, Ph.D. (Physics and Mathematics), Head of Department, nkag@niisi.ras.ru

¹ Federal State Institution "Scientific Research Institute for System Analysis of the Russian Academy of Sciences" (SRISA RAS), Nakhimovsky Ave. 36/1, Moscow, 117218, Russian Federation

Abstract. This article provides performance comparison for three different MIPS processors (RM7000, XLP316 and 1890VM8YA) and describes their architecture details.

To compare processor performance, the testing technique is developed and implemented using C and Assembler. The technique consists of three consecutive stages.

At the first stage the authors measure processor instructions execution time providing that both instructions and data required are already stored in a primary cache. The instructions are divided into several groups. There are the results for members of each group. Primary and secondary cache efficiency is benchmarked on the second stage. The article provides the results for secondary cache and RAM access time only. The third stage uses synthetic performance tests. The obtained results are compared with theoretical estimations based on the results of the first two stages.

The advantage of the proposed technique is in its independence from compiler and operating system specified. All measurements are carried out in clock cycles using special purpose coprocessor registers.

Keywords: testing, MIPS, performance, microprocessor.

References

1. MIPS Architecture For Programmers. Vol. III: The MIPS64 and microMIPS64. Privileged Resource Architecture. Revision 5.04, 2014.
2. PMC-Sierra. Available at: <https://www.pmcs.com> (accessed March 23, 2017).
3. NetLogic Microsystems. Available at: <https://www.netlogicmicro.com> (accessed March 23, 2017).
4. Bobkov S.G. Import substitution of the element base of computer systems. *Vestnik Rossiyskoy akademii nauk* [Herald of the Russian Academy of Sciences]. 2014, vol. 84, iss. 11, pp. 1010–1016 (in Russ.).
5. Curnow H.J., Wichman B.A. A Synthetic Benchmark. *Computer Jour.* 1976, vol. 19, iss. 1, pp. 43–49.
6. Weicker R.P. Dhrystone: A Synthetic Systems Programming Benchmark. *Communications of the ACM*. 1984, vol. 27, iss. 10, pp. 1013–1030.
7. Zargham M.R. *Computer Architecture*. Prentice Hall Publ., 1996.
8. Bakhvalov N.S., Zhidkov N.P., Kobelkov G.M. *Chislennyye metody* [Numerical Methods]. Moscow, Binom. Laboratoriya znaniy Publ., 2003, 632 p.