УДК 004.318

М. Е. Барских[1], зав. сектором, e-mail: barskikh@cs.niisi.ras.ru,
С. Г. Бобков[1, 2], д-р техн. наук, зам. директора, e-mail: bobkov@cs.niisi.ras.ru
[1] НИИ системных исследований РАН,
[2] НИЯУ МИФИ

Исследование влияния динамического предсказания ветвлений на производительность перспективных микропроцессоров НИИСИ РАН

Проведено исследование влияния различных способов предсказания команд условного перехода на производительность микропроцессоров с RISC-архитектурой КОМДИВ. Проведены сравнительные исследования влияния параметров схемы динамического предсказания на ее точность. Подготовлена методика проведения подобных исследований и обоснована корректность реализации блока.

Ключевые слова: *суперскалярный процессор, динамическое предсказание переходов, моделирование*

Введение

Основными факторами сдерживания производительности микропроцессоров являются рост потребляемой мощности и снижение надежности. Во многом данные параметры определяются архитектурой микропроцессоров. В связи с этим в настоящее время все большую популярность завоевывают микропроцессоры с RISC-архитектурой. Производительность систем, построенных на базе RISC-процессоров, практически не уступает процессорам с архитектурой x86 при существенно меньшей стоимости и потребляемой мощности (для примера в табл. 1 приведено сравнение характеристик систем с процессорами ARM и с процессорами Sandy Bridge компании Intel).

Производительность микропроцессоров с RISC-архитектурой во многом определяется возможностями параллельного выполнения нескольких инструкций. Возможность одновременного выполнения инструкций решается, в частности, механизмом предсказания переходов. Суть метода заключается в прогнозировании направления ветвления программы, запросе и начале выполнения команд по адресу перехода. Повышение эффективности данного метода является актуальным для любого современного микропроцессора. Это объясняется тем, что в обычных программах порядка 15 % [1] инструкций являются командами ветвления, которые в простейших случаях реализации архитектуры приводят к остановке конвейера. В современных микропроцессорах глубина конвейера достигает нескольких десятков тактов [2—4], и остановка конвейера приводит к потере этих тактов и к соответственному снижению производительности. Базовые подходы к механизму предсказания ветвления одинаковые для всех RISC-микропроцессоров — это статическое и динамическое предсказания ветвления. Однако реализация этой функции для разных микропроцессоров разная и занимает в случае динамического предсказания, как правило, от двух до пяти тактов, что становится сопоставимым с самим вычислением и приводит к соответствующему повышению потребляемой мощности. Поэтому оптимизация блока предсказания ветвления для каждого микропроцессора проводится индивидуально.

Можно дать следующее определение модулю предсказания условных переходов (*branch prediction unit*, BPU) — устройство, входящее в состав микропроцессоров, имеющих конвейерную архитектуру, которое определяет направление ветвлений (предсказывает, будет ли выполнен условный переход) в исполняемой программе. Предсказание ветвлений позволяет осуществить предварительную выборку инструкций и данных из памяти и начать выполнение инструкций, находящихся после условного перехода, до того, как он будет выполнен. Предсказатель переходов является неотъемлемой частью всех современных суперскалярных микропроцессоров, так как в большинстве случаев (точность предсказания переходов в современных процессорах превышает 90 %) позволяет оптимально использовать вычислительные ресурсы процессора.

Таблица 1

Сравнительные характеристики процессоров с архитектурами RISC и Intel

Микропроцессор	Производительность, Гфлопс	Число адресных пространств	Объем накристальной памяти, Мбайт	Пропускная способность памяти, Гбайт/с	Пропускная способность коммуникационных каналов, Гбайт/с	Стоимость, $	Мощность, Вт
2×Sandy Bridge	370	1	44	136	56	>3000	>350
8×ARM	256	8	16	102	40	<200	<100

Существуют два основных метода предсказания переходов: статический и динамический.

Статические методы предсказания ветвлений являются наиболее простыми. Суть этих методов состоит в том, что переходы различных типов либо выполняются всегда, либо не выполняются никогда. При реализации статических предсказаний предполагается, что любой переход "назад" (т. е. переход на младшие адреса) является циклом и должен выполняться, а любой переход "вперед" (т. е. на старшие адреса) не выполняется. Динамические методы, широко используемые в современных процессорах, подразумевают анализ истории ветвлений, обновляемой по результатам фактического выполнения команд переходов.

Поэтому для применения в разрабатываемых в НИИСИ перспективных микропроцессорах с MIPS-подобной архитектурой КОМДИВ [5, 6] был разработан блок динамического предсказания ветвления. Целью данной работы была верификация этого блока, т. е. проверка, что он "работает" и "работает хорошо". В качестве критериев использовались следующие характеристики: точность предсказания переходов (команд ветвления) и производительность ядра процессора в виде среднего числа выполняющихся команд за такт (*instruction per cycle*, IPC). Проверка того, что блок динамического предсказания переходов "работает", заключалась в подтверждении того, что точность предсказания и значение IPC выше, чем в случае статического предсказания и, тем более, без него. Так как существует множество параметров, влияющих на работу блока динамического предсказания, следующей целью работы было определение лучшего их сочетания. В этом случае можно говорить о том, что блок "работает хорошо", т. е. мы получаем максимальную точность предсказаний при одинаковых аппаратных затратах на реализацию этого блока. Вместе с тем, погоня только за точностью предсказания может приводить к необоснованному усложнению блока и, соответственно, к росту потребляемой мощности. Поэтому использование в качестве критерия IPC позволяет определить момент, когда усложнение блока уже практически не влияет на рост производительности и на данной реализации можно остановиться.

Выбор схемы динамического предсказания

Выбор схемы предсказания основывался на анализе решений, применявшихся в коммерческих процессорах. К сожалению, в последнее время технических данных об используемых схемах предсказания немного. Это связано с тем, что выбор оптимальной схемы — очень кропотливое и долгое занятие, а результат является коммерческой тайной, так как позволяет увеличить производительность процессора. При этом улучшения тем выше, чем длиннее конвейер и чем большее окно выполнения он имеет. Кроме этого, в последнее время в процессорах используются не просто схемы предсказаний направления перехода, а схемы, предсказывающие целевой адрес перехода и использующие буфер целевых адресов (*branch target buffer*, BTB). Это необходимо в случае, когда только запрос инструкций занимает несколько тактов и необходимо предсказывать не только направление, но и адрес перехода для конвейерной выборки инструкций. Однако сложность BTB-буфера сопоставима со сложностью кэш-памяти инструкций. Наиболее полное перечисление используемых механизмов предсказаний переходов в микропроцессорах x86 приведены в [7].

Проведенный анализ показал, что наиболее оптимальной из известных схем динамического предсказания переходов с точки зрения сложности и объема схемы, потребления питания и получаемых результатов предсказания является двухуровневая гибридная схема динамического предсказания переходов, реализованная в микропроцессорах Alpha 21264 [8, 9], использованная затем в процессорах POWER4 [10] и POWER5 [11]. Для дополнительного снижения потребления питания в предлагаемой нами схеме локального предсказания не используется память локальной истории переходов, а сам счетчик предсказания — двухразрядный. Общая логическая схема блока представлена на рис. 1.

Для обеспечения возможности анализировать различные реализации блока предсказаний схема сделана конфигурируемой. Конфигурационные регистры, доступные в том числе и программно, позволяют изменять:

- тип предсказания (динамическое, статическое и отключено);
- вид динамического предсказания;
- алгоритм обновления истории;
- размер памятей истории;
- варианты адресации (выбора данных) памятей истории.

В результате блок может быть переконфигурирован для исследования большого числа вариантов схем динамического предсказания, в частности, для проведения исследований схемы gskew [12], которая разрабатывалась для микропроцессора Alpha EV8 [13].

Благодаря относительно короткому конвейеру в семь стадий (из которых запрос инструкций со-

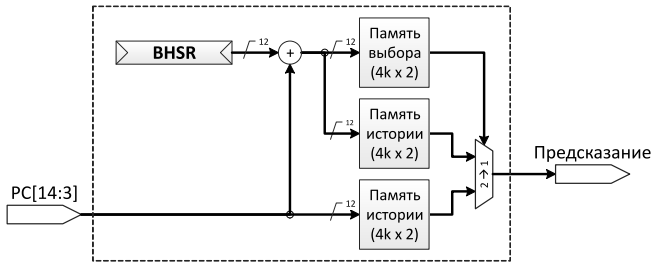

Рис. 1. Общая схема работы блока динамического предсказания (BHSR — branch history shift register, сдвиговый регистр истории переходов; PC — program counter, программный счетчик)

ставляет две стадии) в такой схеме нет необходимости использовать BTB. Расчет адреса перехода делается в момент получения инструкций из кэш-памяти, в следующем такте выставляется новый запрос по предсказанному (или следующему, если предсказания не было) адресу. Дополнительно с блоком динамического предсказания, для команд безусловных регистровых переходов сделан буфер возврата на восемь адресов.

Данные о точности предсказаний и производительности накапливаются в специальных регистрах, доступных для чтения.

Конвейер процессора и запрос инструкций

На рис. 2 показаны две первые стадии конвейера процессора, в которых происходит запрос инструкций из кэш-памяти. Параллельно с запросом инструкций выполняется расчет предсказания перехода и рассчитывается целевой адрес перехода. Для расчета адреса смещения берутся из кода полученных инструкций (для условных и не регистровых безусловных переходов) или из стека возврата (для регистровых безусловных переходов). Это было возможно, так как кэш-память инструкций работает на сдвинутом на половину периода тактовом сигнале.

В части запроса инструкций процессор имеет две особенности, которые необходимо было учитывать при разработке: у всех инструкций перехода MIPS-архитектуры есть слот задержки, который выполняется после этой инструкции; и в случае попадания в кэш инструкций буфер автоматически получит инструкции по текущему и "инкрементированному" запросу. Наличие слота задержки и его обязательное выполнение приводят к тому, что если он не был получен в текущем запросе, следующий запрос будет по адресу слота задержки, а запрос по предсказанному адресу будет отложен.

Возможность получения двух групп инструкций по одному запросу обусловливает необходимость контролировать момент выставления предсказанного запроса для отмены записи в буфер инструкций, полученных от предыдущего запроса. Эти ситуации контролируются конечным автоматом запросов, который получает информацию от блока предсказаний. Кроме этого, автомат запросов отслеживает отмену запросов, связанных с исключительными ситуациями, или восстановление после неправильного предсказания. Однако из-за унификации поведения буфера инструкций в этих двух ситуациях и благодаря одинаковому механизму работы при предсказании любого типа ветвлений (условного, безусловного или вероятного (*branch likely*) перехода), сложность машины состояний по сравнению с версией 1890ВМ6 даже уменьшилась.

Так как на вычисление адреса остается половина такта, что недостаточно для полного 64-разрядного сумматора, адрес вычисляется в два этапа (такта). На первом такте вычисляются инкрементированный и декрементированный адреса 48-битной базы, общей для всех получаемых инструкций, и 16-битные смещения базового адреса для каждой из четырех получаемых инструкций. Результаты сохраняются в регистры и используются на следующем такте, когда из кэш-памяти поступают считанные инструкции. Младшие 16 бит всех инструкций суммируются с соответствующими смещениями базы (адреса запроса) и на основании знака смещения инструкции и флагов переносов двух сумматоров смещений выбирается соответствующая старшая часть базового адреса.

Стек возврата представляет собой кольцевой буфер на восемь записей для сохранения последних адресов, генерируемых командами jump & link (ими делается переход в подпрограммы в MIPS-архитектуре). Чтение из него происходит командами безусловных регистровых переходов, которые делают возврат из подпрограмм. От "классического" стека стек возвратов отличается тем, что он может быть считан, когда в нем нет записей, и записан, когда он полон (так как выполнение инструкций ничем не ограничено).

При записи в заполненный стек затирается самая первая запись, соответствующая самому старому адресу возврата, и работа продолжается в нормальном режиме (буфер инструкций даже не имеет информации об этом событии). При получении команды регистрового перехода целевой адрес для нее берется из стека возврата, и команда считается предсказанной. После фактического выполнения команды рассчитанный адрес перехода сравнивается с предсказанным, и в случае ошибки инструкции перезапрашиваются. В случае пустого стека дальнейшее выставление запросов блокируется до фактического выполнения команды перехода и вычисления его адреса уже на стадии выполнения.

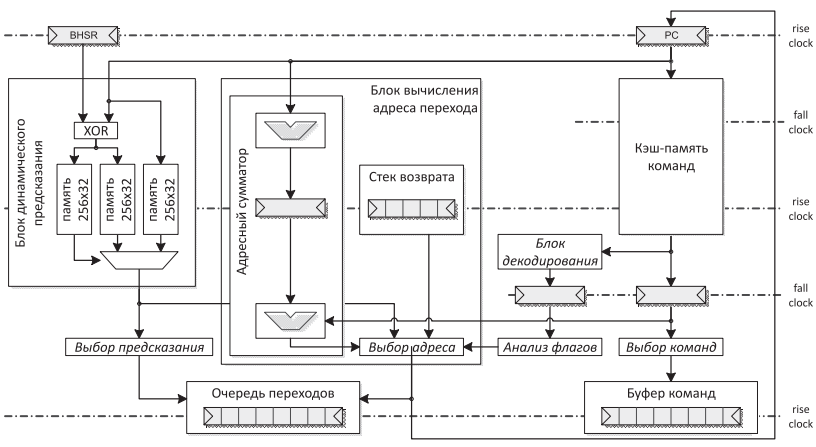

Рис. 2. Стадии запроса инструкций и предсказания перехода (первые два такта конвейера)

Выбор тестов и моделирование

Для моделирования была взята RTL модель системы на кристалле (в состав которой входят системный контроллер и ядро процессора). К модели подключена память DDR2, соотношение частот при моделировании: 500 МГц для ядра процессора и 250 МГц для памяти DDR2. Моделирование проводилось на аппаратном ускорителе Palladium XP фирмы Cadence. После компиляции проект имел реальную частоту 1,7 МГц. Это позволило моделировать тесты длительностью примерно 5 млн инструкций, что достаточно как для начального накопления истории переходов, так и для оценки работы схемы предсказаний.

Принципиальная схема предсказания переходов, реализованная для предлагаемого блока, с управляющей логикой, с помощью которой происходило конфигурирование схемы, показана на рис. 3. Для моделирования были отобраны программы, выполняющие различные операции (табл. 2), структура тестов (пропорции команд в тесте по группам) представлена на рис. 4 (см. третью сторону обложки). В качестве тестов были использованы как широко известные тесты производительности (Coremark, Dhrystone и др.), так и программы, реализующие прикладные алгоритмы (gZip, MatrixMult). Выбор тестов обусловлен необходимостью иметь возможность ограничивать их длительность без ущерба для вычислительного ядра программы, т. е. чтобы функциональность программы не пострадала. В отличие от тестов Spec, они были небольшими, не требовали поддержки со стороны операционной системы (что не только замедляет моделирование, но и вносит погрешность из-за накопления истории предсказаний при выполнении кода ОС) и наиболее хорошо подходили для целей исследовательского тестирования.

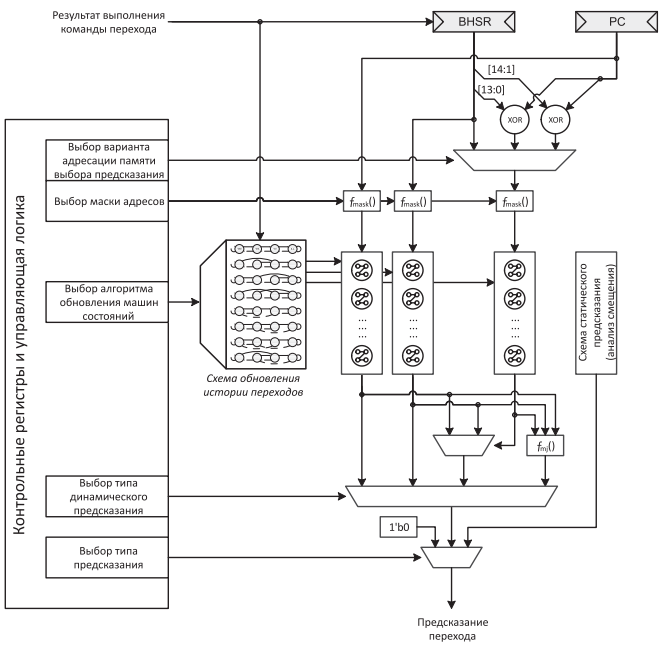

Рис. 3. Схема блока динамического предсказания с управляющей логикой, использовавшаяся при моделировании

Таблица 2

Тесты, использовавшиеся при моделировании

Тест	Описание	Длительность теста	
		Число команд	%
Coremark	Тест производительности Coremark	6 740 566	10,41
Dhrystone	Тест производительности Dhrystone	4 470 429	6,90
FFT	"Бабочка" Фурье двойной точности	5 732 961	8,85
Fibonacci	Поиск N-го числа Фибоначчи	5 607 102	8,66
gZip	Алгоритм архиватора gzip	6 408 230	9,89
Hanoi	Алгоритм перемещения "Ханойской башни"	4 998 254	7,72
Heapsort	Алгоритм пирамидальной сортировки	5 325 662	8,22
MatrixMult	Умножение вещественных матриц двойной точности различными алгоритмами	9 832 118	15,18
Queens	Решение задачи о расстановке N ферзей на шахматной доске размером $N \times N$	13 639 321	21,06
Whetstone	Тест производительности вещественной арифметики Whetstone	2 012 127	3,11
Всего		64 766 630	100,00

Анализ полученных результатов

В первую очередь, для анализа работоспособности блока предсказания были промоделированы схемы статического и динамического предсказаний, а также (для сравнения) работа процессора без предсказаний, когда переход считается никогда не выполняющимся. Точность предсказания определялась как отношение правильно предсказанных инструкций ветвления к их общему числу. Для случая, когда предсказание было выключено, все инструкции ветвления считались "не предсказанными", и точность работы в этом случае является отношение невыполненных инструкций перехода к их общему числу. Значение IPC есть отношение числа выполненных команд (без учета отмененных из-за неправильного предсказания) к длительности всего теста в тактах.

Полученные результаты представлены на рис. 5. На основании этих данных можно говорить о том, что реализованная схема динамического предсказания работает: точность предсказания (по сравнению со статическим методом) выросла на 36 %, что дало увеличение IPC на 5 %. Небольшой (относительно увеличения точности) рост IPC объясняется, в первую очередь, наличием тестов вещественной арифметики (см. рис. 4 на третьей стороне обложки), в которых длительность выполнения веществен-

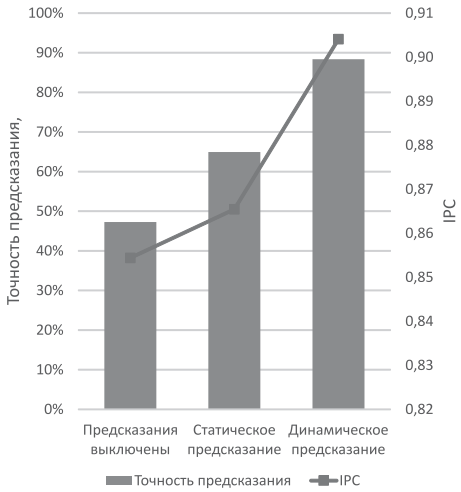

Рис. 5. Точность статического и динамического предсказаний

ных команд вносит существенный вклад в длительность всего теста и мало зависит от точности предсказаний. Вместе с тем, это показатель того, насколько сложно добиться реального увеличения производительности.

Для анализа собственно схемы динамического предсказания были промоделированы различные виды ее реализации: две схемы одноуровневого предсказания, различающиеся способом адресации памяти истории переходов (biModal) и (gShare), схема мажоритарного предсказания (majority) и исходная схема с использованием памяти выбора предсказания (choice) (см. рис. 1 и рис. 6). Полученные результаты коррелируют с известными из литературы [12, 14, 15]: одноуровневое предсказание имеет наименьшую точность, требуя при этом только одного банка памяти. Мажоритарная схема работает несколько лучше, однако требует уже три банка памяти, но в этом случае нужно реализовывать схему с выбором предсказания, которая дает лучший результат при тех же аппаратных затратах.

Влияние размера памяти на точность предсказания для этих четырех видов динамического предсказания показано на рис. 7. Размер памяти с помощью маски ограничивался от 16 байт (размера строки физического блока памяти) до полного размера в 4 Кбайт. На графике видно, что точность схемы с выбором предсказания выше при любом размере памятей, и не растет при увеличении размера памяти выше 512 байт. Некоторое уменьшение точности при размерах памяти свыше 512 байт объясняется потерями на накопление истории, так как для большего объема памяти требуется больше времени на сбор истории переходов, но оно еще не компенсируется увеличением точности работы схемы выбора.

В целом, собранные данные на рис. 6 и 7 коррелируют с приводимыми в источниках [16, 17] и подтверждают правильность реализации как самих схем, так и управляющей логики, которой эти схемы выбирались.

Основываясь на этих результатах, для дальнейшего анализа была выбрана схема предсказания с выбором (*choice*). В исходной реализации память выбора предсказания адресовалась значением, полученным сложением по модулю 2 (XOR) программного счетчика и сдвигового регистра глобальной истории переходов (*branch history shift register*, BHSR). Кроме такой реализации в качестве адреса можно использовать значение самого программного счетчика и регистра BHSR или изменить функцию их комбинирования (например, для функции XOR не учитывать значение последнего перехода). Такой анализ должен показать, можно ли улучшить реализацию схемы предсказания изменением адресации ее памятей. По результатам моделирования, приведенным на рис. 8, лучший результат показала схема, в которой в качестве адреса памяти выбора предсказания используется значение программного счетчика. Поэтому для дальнейшего использования в схеме выбран такой механизм адресации.

При описаниях схем динамического предсказания в работе [18] встречается анализ изменений алгоритма обновления двухбитного конечного автомата, лежащего в основе схемы предсказания.

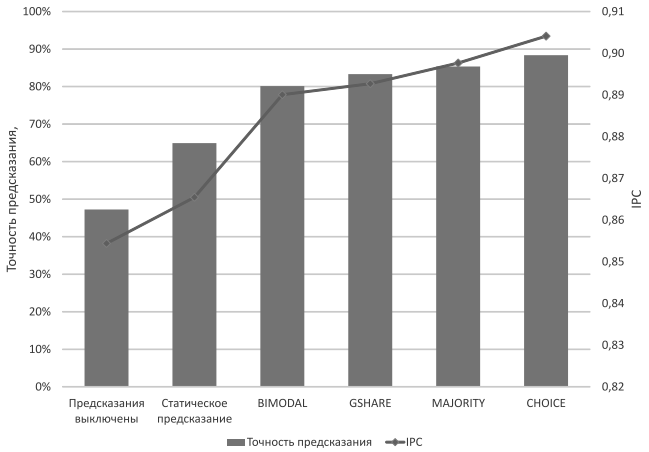

Рис. 6. Точность разных видов динамического предсказания

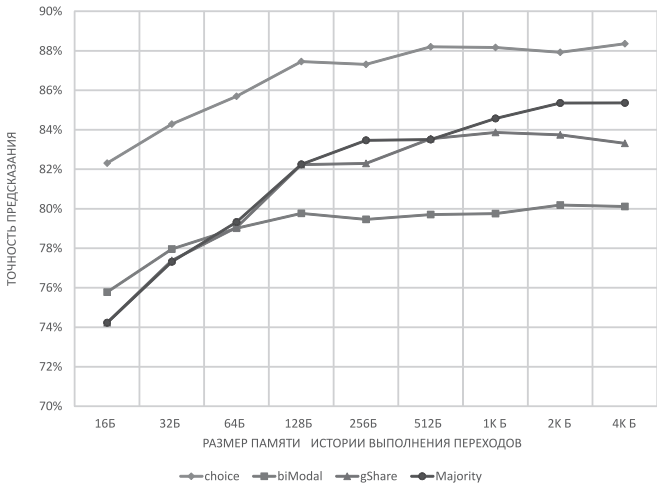

Рис. 7. Точность предсказания в зависимости от размера памятей

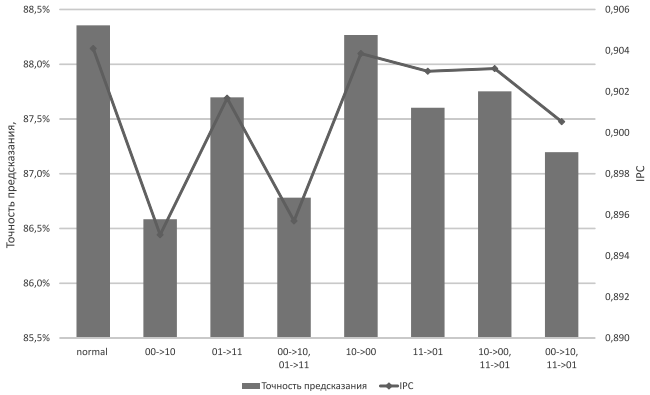

Рис. 8. Точность предсказания при различных вариантах адресации памяти выбора результата

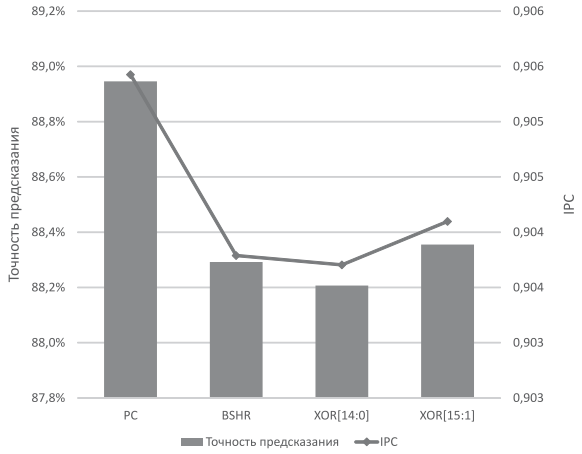

Рис. 9. Точность предсказания при различных алгоритмах обновления машин состояний

Анализ начального состояния истории не проводился, так как его невозможно контролировать в микросхеме, но статистика изменения точности предсказаний при различных алгоритмах обновления значения двухбитного конечного автомата была собрана, результаты показаны на рис. 9.

В среднем, классическая реализация двухбитного счетчика оказалась лучшей для большинства тестов, а попытка предсказывать переход как "выполняемый" после первого его реального выполнения приводила даже к худшим результатам, чем предсказание перехода "невыполняемым" после первой же невыполненной команды ветвления. Хотя были тесты, для которых изменение логики обновления истории приводило к улучшению точности (и, самое главное, к увеличению IPC), для остальных тестов эти же изменения сказывались отрицательно. Возможно, стоит проверить тесты, для которых в литературе указывается лучшая стратегия обновления истории, но это может быть результатом отдельного исследования.

Заключение

В результате проведенного исследования была обоснована корректность реализации блока динамического предсказания в микропроцессоре КОМДИВ, а также проанализировано влияние различных параметров на точность работы схемы динамического предсказания. Проведенное исследование показывает, что реализованная гибридная схема динамического предсказания является максимально производительной при данных аппаратных затратах. Единственное изменение, внесенное в схему по результатам моделирования, касалось адресации памяти выбора предсказания: оно было изменено с функции XOR на адресацию памяти счетчиком команд. Никакие изменения других параметров не приводили к увеличению точности работы схемы.

Кроме того, была собрана и проанализирована статистика по изменению производительности при применении той или иной схемы. На основании этих данных можно обоснованно выбирать для использования разные схемы динамического предсказания в случае необходимости экономии аппаратных ресурсов. Например, влияние памяти на точность предсказания ограничивается 512 байтами, т. е. в случае необходимости ее можно уменьшить без существенной потери точности работы схемы и производительности.

Также было проанализировано влияние различных факторов на работу схемы предсказания. Так, изменение вариантов адресации памяти выбора предсказания или алгоритмов обновления машин состояний истории переходов может давать эффект для определенных программ и может быть полезно при оптимизации микропроцессора для решения именно этого класса задач. Кроме этого, ожидается, что на точность будет оказывать влияние число параллельно работающих памятей и способ выбора конечного предсказания, что является темой для следующих исследований.

Список литературы

1. **Huang Y., Zha Z., Chen M. and Zhang L.** Moby: A Mobile Benchmark Suite for Architectural Simulators // IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS). 2014. P. 45—54.

2. **Intel** Atom Processor 330. 2008. URL: http://ark.intel.com/products/35641/Intel-Atom-Processor-330-1M-Cache-1 60-GHz-533-MHz-FSB

3. **Cortex-A** Series Programmer's Guide, rev. 4.0, 2014. URL: http://infocenter.arm.com/help/index.isp?topic=/com.arm.doc.den0013d/index.html

4. **Halfhill T. R.** P5600 Extends MIPS Performance. Microprocessor Report, 2013.

5. **Бобков С. Г.** Архитектура перспективных высокопроизводительных микропроцессоров // Программные продукты и системы. 2012. № 3. С. 63—68.

6. **Бобков С. Г.** Импортозамещение элементной базы вычислительных систем // Вестник Российской академии наук. 2014. Т. 84, № 11. С. 1010—1016.

7. **The microarchitecture** of Intel, AMD and VIA CPUs: An optimization guide for assembly programmers and compiler makers. URL: http://www.agner.org/optimize/microarchitecture.pdf (последнее изменение 19.02.2014).

8. **Kessler R. E.** The Alpha 21264 Microprocessor // IEEE Micro. 1999. Vol. 19, No. 2. P. 24—36.

9. **Compaq** Computer Corp., Alpha 21264 Microprocessor Hardware Reference Manual, 1999.

10. **Tendler J. M., Dodson J. S., Fields J. S., Le H., Sinharoy B.** POWER4 System Microarchitecture // IBM Journal of Research and Development. 2002. Vol. 46, No 1. P. 5—25.
11. **Sinharoy B., Kalla R. N., Tendler J. M., Eickemeyer R. J., and Joyner J. B.** POWER5 System Microarchitecture // IBM Journal of Research and Development. 2005. Vol. 49, No 4/5. P. 505—521.
12. **Michaud P., Seznec A., and Uhlig R.** Trading conflict and capacity aliasing in conditional branch predictors // Proceedings of the 24th Annual International Symposium on Computer Architecture (ISCA-97). 1997. P. 292—303.
13. **Seznec A., Felix S., Krishnan V., Sazeides Y.** Design Tradeoffs for the Alpha EV8 Conditional Branch Predictor // 29th IEEE-ACM International Symposium on Computer Architecture (ISCA-02). 2002. P. 295—306.
14. **McFarling S.** Combining Branch Predictors // Technical Report TN-36, Digital Western Research Laboratory, 1993.
15. **Eden A. N. and Mudge T. N.** The YAGS Branch Prediction Scheme // 31st annual ACM/IEEE international symposium on Microarchitecture. 1998. P. 69—77.
16. **Yeh T.-Y. and Partt Y. N.** A Comparison of Dynamic Branch Predictors that Use Two Levels of Branch History // in 24th annual ACM/IEEE International Symposium on Microarchitecture. 1993. P. 257—266.
17. **Stark J., Evers M., and Patt Y. N.** Variable Length Path Branch Prediction // 8th international conference on Architectural support for programming languages and operating systems. 1998. P. 170—179.
18. **Shen J. P., Lipasti M. H.** Modern Processor Design: Fundamentals of Superscalar Processors // Waveland Press, Inc., USA, 2013.

M. E. Barskikh[1], Head of section, e-mail: barskikh@cs.niisi.ras.ru,
S. G. Bobkov[1, 2], Doctor of Science, Deputy Director, e-mail: bobkov@cs.niisi.ras.ru
[1] Russian Academy of Sciences, Scientific Research Institute for System Analysis
[2] National Research Nuclear University MEPhI (Moscow Engineering Physics Institute)

Investigation of the Influence of Dynamic Branch Prediction to Perspective Microprocessors Performance from NIISI RAS

The effect of different dynamic branch prediction methods on the perspective microprocessors performance with RISC architecture KOMDIV is investigated. The simulation was performed on the System-on-Chip RTL-model with system controller and CPU core, which gives cycle-accuracy results. The dynamic prediction block was made configurable to be able to control the various prediction schemes. For the test cases a well-known benchmarks (Coremark, Dhrystone, etc.), and a real application programs (gZip, MatrixMult and other) have been used. The comparative studies of various effects of dynamic branch prediction schemes were conducted with the analysis of hardware consumption. Based on these data, choice of combined prediction scheme (with memory to prediction choice) for use is substantiated in the article. Besides, size and memory addressing function influence were collected and analyzed for the prediction accuracy. Based on this data modification in the memory addressing algorithm were amended: program counter has been selected directly to select memory instead of XOR function. Additional analysis of 2-bit FSM update algorithms showed the possibility of improving the accuracy and, most importantly, the performance for certain tests only, but, in general, the classic algorithm 2-bit counter is the best choice. This research confirmed the gain from dynamic branch prediction used in processors with a short pipeline and architecture KOMDIV for the feasible application of it for the serial chips.

Keywords: *superscalar processor, dynamic branch prediction, simulation*

References

1. **Huang Y., Zha Z., Chen M. and Zhang L.** Moby: A Mobile Benchmark Suite for Architectural Simulators. *IEEE International Symposium on Performance Analysis of Systems and Software* (ISPASS), 2014, pp. 45—54.
2. **Intel** *Atom Processor* 330, 2008. URL: http://ark.intel.com/products/35641/Intel-Atom-Processor-330-1M-Cache-1_60-GHz-533-MHz-FSB
3. **Cortex-A** *Series Programmer's Guide,* rev. 4.0. 2014. URL: http://infocenter.arm.com/help/index.isp?topic=/com.arm.doc.den0013d/index.html
4. **Halfhill T. R.** P5600 Extends MIPS Performance. *Microprocessor Report*, 2013.
5. **Bobkov S. G.** Arkhitektura perspektivnykh vysokoproizvoditel'nykh mikroprotsessorov. *Programmnye produkty i sistemy*, 2012, no. 3, pp. 63—68.
6. **Bobkov S. G.** Importozameshchenie elementnoi bazy vychislitel'nykh system. *Vestnik Rossiiskoi Akademii Nauk*, 2014, vol. 84, no. 11, pp. 1010—1016.
7. **The microarchitecture** *of Intel, AMD and VIA CPUs: An optimization guide for assembly programmers and compiler makers.* URL: http://www.agner.org/optimize/microarchitecture.pdf (last update 19.02.2014).
8. **Kessler R. E.** The Alpha 21264 Microprocessor. *IEEE Micro*, 1999, vol. 19, no. 2, pp. 24—36.
9. **Compaq** Computer Corp., *Alpha 21264 Microprocessor Hardware Reference Manual*, 1999.
10. **Tendler J. M., Dodson J. S., Fields J. S., Le H., Sinharoy B.** POWER4 System Microarchitecture. *IBM Journal of Research and Development*, 2002, vol. 46, no. 1, pp. 5—25.
11. **Sinharoy B., Kalla R. N., Tendler J. M., Eickemeyer R. J., and Joyner J. B.** POWER5 System Microarchitecture. *IBM Journal of Research and Development*, 2005, vol. 49, no. 4/5, pp. 505—521.
12. **Michaud P., Seznec A., and Uhlig R.** Trading conflict and capacity aliasing in conditional branch predictors. *Proceedings of the 24th Annual International Symposium on Computer Architecture (ISCA-97),* 1997, pp. 292—303.
13. **Seznec A., Felix S., Krishnan V., Sazeides Y.** Design Tradeoffs for the Alpha EV8 Conditional Branch Predictor. *29th IEEE-ACM International Symposium on Computer Architecture (ISCA-02),* 2002, pp. 295—306.
14. **McFarling S.** Combining Branch Predictors. *Technical Report TN-36, Digital Western Research Laboratory*, 1993.
15. **Eden A. N. and Mudge T. N.** The YAGS Branch Prediction Scheme. *31st annual ACM/IEEE international symposium on Microarchitecture*, 1998, pp. 69—77.
16. **Yeh T.-Y. and Patt Y. N.** A Comparison of Dynamic Branch Predictors that Use Two Levels of Branch History". *24th annual ACM/IEEE International Symposium on Microarchitecture*, 1993, pp. 257—266.
17. **Stark J., Evers M., and Patt Y. N.** Variable Length Path Branch Prediction. *8th international conference on Architectural support for programming languages and operating systems*, 1998, pp. 170—179.
18. **Shen J. P., Lipasti M. H.** Modern Processor Design: Fundamentals of Superscalar Processors. *Waveland Press, Inc.*, USA, 2013.