УДК 004.315

Исследование способов построения блока динамического предсказания ветвлений для перспективных микропроцессоров разработки НИИСИ РАН

М.Е. Барских

Институт системных исследований РАН, barskikh@cs.niisi.ras.ru

***Аннотация* — В статье описан анализ вариантов реализации блока динамического предсказания ветвления. Проведены сравнительные исследования влияния параметров схемы на её точность. Для выбора оптимальной реализации исследовалось влияние различных схем на производительность микропроцессоров с RISC архитектурой.**

***Ключевые слова* — суперскалярный процессор, динамическое предсказание переходов, моделирование, комбинированная схема, мажоритарная схема.**

I. Введение

Команды ветвления в программе создают зависимости по управлению, определяя порядок выполнения инструкций в конвейере. Поэтому новые инструкции не могут запрашиваться до тех пор, пока не будет вычислено условие перехода и, соответственно, не будет определен адрес следующей инструкции для её выборки и выполнения. Решением данной проблемы является использование предсказания переходов, самой простой реализацией которого является статическое предсказание.

Недостатком его является точность, которая сильно зависит от исполняемой программы. Например, для программ с большим количеством циклов статическое предсказание может показать точность выше 90%, а в случае анализа данных его точность может быть порядка 50%, т.е. не выше случайного угадывания. Поэтому практически во всех современных процессорах используется динамическое предсказание ветвлений, которое основывается на анализе накопленной истории выполнения команд ветвления.

II. Принципиальные схемы динамического предсказания

Схемы предсказания переходов используют двухбитные счетчики истории с различными вариантами обновления. Отчет [1] был первой работой по систематизации многоуровневых схем динамических предсказаний. Представленные в этой работе схемы двухуровневых предсказаний вместе с терминологией затем использовались для реализации в коммерческих процессорах. Однако термин «многоуровневые схемы» в современных проектах понимается как применение нескольких уровней таблиц, выходы первых из которых используются как адреса следующих, и только в последней таблице содержится двухбитный код, отвечающий за предсказание направления перехода.

Одноуровневые схемы представляют собой память (**Branch History Table, BHT**), хранящую значения двухбитного насыщающегося счетчика (**saturated count**), адресуемую различными способами. Базовые схемы предсказаний, показанные на рис. 1, следующие:

1) **bimodal** – таблица истории адресуется счетчиком команд (**program counter**, **PC**).

2) **Global history** – таблица истории адресуется значением регистра глобальной истории перехода (**Branch History Shift Register, BHSR**).

3) Схемы, использующие различные функции смешения значений PC и BHSR: **gSelect** – совместное использование (конкатенация) младших бит PC и BHSR, когда биты объединяются, образуя адрес BHT; **gShare** – объединение PC и BHSR по функции XOR.

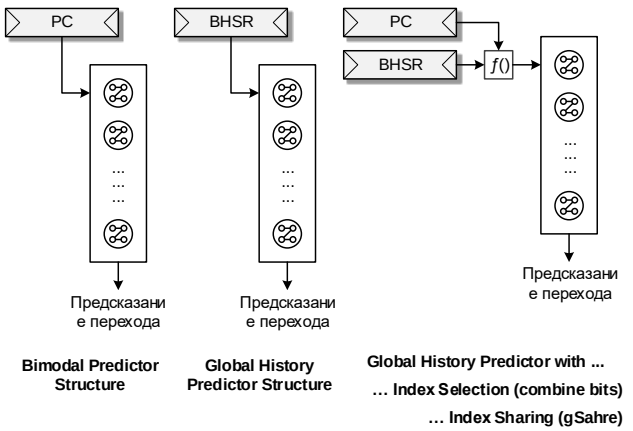

Рис. 1. Одноуровневые схемы предсказания переходов

Эти схемы используются как базовые для более сложных, но имеющих большую точность алгоритмов [2] (рис. 2):

1) **Combining** (**комбинированная схема**) – схема содержит два блока предсказаний, которые формируют каждый свой результат (P_1 и P_2 на рисунке), и дополнительную таблицу P_{ch}, обеспечивающую выбор предсказания для дальнейшего использования. Таблица выбора (как и основные таблицы BHT в плечах схемы)

содержит двухбитные счетчики, но отличается алгоритмом их обновления и, возможно, схемой адресации.

2) **Majority (ассиметричная схема)** – схема с нечетным количеством блоков предсказания, где результат выбирается по мажоритарному принципу: как предсказали большинство блоков. Главная идея такой схемы в том, чтобы отдельные блоки имели разные хэш-функции для адресации своих таблиц предсказаний. В таком случае они будут работать независимо и давать репрезентативный результат конечного предсказания.

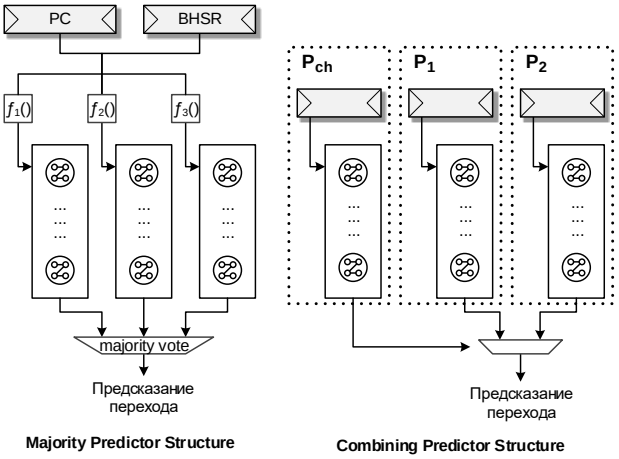

Рис. 2. Комбинированные схемы предсказаний

Еще одной схемой, часто использующейся в современных процессорах, является схема **Local history (с локальной историей)**. В такой схеме глобальная история выполнения команд ветвления хранится для каждого PC, в свою очередь адресуя каждая свою BHT с историей (см. рис. 3). Количество таблиц BHT равно 2^n, где n – количество бит истории выполнения переходов по этому адресу. Очевидным недостатком такой схемы является объем памяти, требуемый для хранения BHT. Поэтому в реальных процессорах ее реализация может значительно отличаться от показанной.

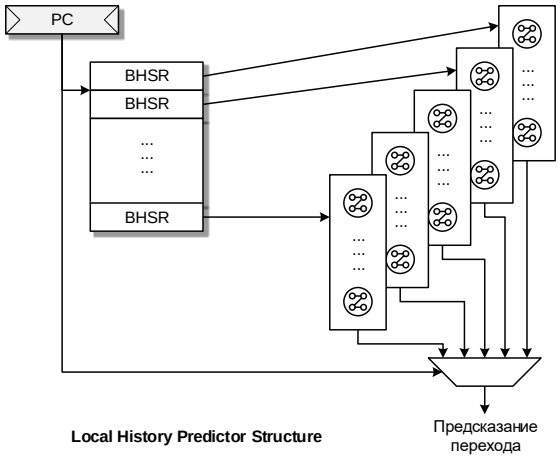

Рис. 3. Схема с локальной историей

Помимо описанных схем существует множество их вариантов, отличающихся функциями адресации, составом используемых базовых схем, механизмов обновления истории. Кроме того, было предложено много оригинальных схем, таких как согласованная схема [3]; YAGS-схема [4]; схемы с использованием глобальной истории переменной длины [5] и многие другие. Однако при описании конкретных реализаций в коммерческих процессорах используются в основном описанные выше схемы.

III. Схемы динамического предсказания, используемые в современных процессорах

В настоящее время использование динамического предсказания переходов в процессорах стало почти обязательным. Применяемые в коммерческих процессорах схемы представляют собой комбинацию из описанных в разделе I. Однако точных технических данных об их реализации не много.

A. INTEL

INTEL заявляет об использовании локальной схемы предсказания переходов, не раскрывая размер и организацию самих таблиц BHT [6]. Предсказание ветвлений комбинируется со схемами предсказания циклов и непрямых переходов. Современные ядра используют BHSR размером до 32 бит, но количество, объем и организация таблиц BHT неизвестны. Результаты тестов определяют наличие нескольких алгоритмов предсказания различных типов инструкций переходов [7].

B. AMD

В ядре Bulldozer используется комбинированная схема локального и глобального предсказаний. Глубина регистра BHSR составляет 12 бит, но размер и количество таблиц BHT для локальной схемы не описаны. В ядрах Bobcat и Jaguar глубина BHSR увеличена до 26 бит, но более подробное описание схемы не приводится [8].

C. IBM

В процессорах архитектуры POWER5 используется комбинированная схема, общая для обоих потоков инструкций ядра. Вариант схемы gShared использовал для адресации также и адреса предыдущих запросов. Размер BHT составлял 8K записей. В архитектуре POWER6 размер таблиц был увеличен до 16K записей. В POWER7 структура предсказаний осталась комбинированной, размеры таблиц кроме BHT глобальной истории (осталась 16K) уменьшены до 8K записей. Таблица локальной истории напрямую адресуется программным счетчиком, для остальных таблиц используются функции с использованием регистра BHSR глубиной 21 бит, который уникален для каждого потока инструкций. Функции адресации таблиц не публикуются [9].

D. ALPHA

Процессор Alpha 21264 имел, наверное, самое подробное описание блока предсказания ветвления в документации. Блок был построен по комбинированной схеме, для локального предсказателя использовалась относительно небольшая таблица BHT на 1024 записи, но в качестве истории при этом использовался трех-

битный счетчик. Глобальный предсказатель и блок выбора предсказания, оба размером 4096 записей, имели «стандартную» двухбитную историю и адресовались только регистром BHSR глубиной 12 бит [10].

В процессоре Alpha 21464 (EV8) планировалось улучшить блок предсказания, используя в одном плече отдельную мажоритарная схему. Предсказание bimodal использовалось, чтобы уменьшить интерференцию, возникающую при использовании глобальной истории. Обновление истории происходило только при неправильном общем предсказании с промежуточной проверкой результата. Это было сделано для того, чтобы не загонять все счетчики в режим насыщения, т.к. это негативно сказывается на предсказаниях других инструкций ветвления, использующих те же записи [11].

E. ARM

Предсказания ветвлений стали применяться в процессорах ARM начиная с ядра ARM11. Блок предсказаний состоял из одной таблицы BHT на 128 записей. Ядро Cortex-R использует схему gShare, размер BHT увеличен вдвое до 256 записей. Следующее увеличение объема BHT было сделано только для процессора Cortex-A8: размер памяти увеличен до 4K записей.

В процессорах Cortex-A9 размер таблиц значительно увеличен и может быть изменен при лицензировании ядра. Таблица BHT может иметь от 1K до 16K записей и использует «классический» 2-битный счетчик истории. Однако она также осталась одна, многоуровневое предсказание не используется [12]. В 64-разрядных версиях предсказания остались такими же простыми: в Cortex A53 используется таблица PTH на 3072 записи [13].

F. MIPS

Процессор R10000 использовал схему предсказания bimodal с таблицей BHT размером 512 записей, которая была изменена на gShare в R12000. В ядрах MIPS64 20K и MIPS32 1004K использовалась схема bimodal с таблицей BHT на 256 и 512 записей соответственно, а сам блок мог предсказывать одновременно до 2 команд ветвления. MIPS32 74K и 1074K использовали мажоритарную схему с BHT на 512 записей [14]. После покупки MIPS компанией Imagination открытых публикаций об используемых схемах динамического предсказания не приводилось, но в презентациях компании указывалась мажоритарная схема.

G. Краткий анализ используемых схем

INTEL и AMD вынуждены использовать сложные схемы предсказания, только в общих чертах описывая их состав. Для регистра глобальной истории указывается только его глубина, но никакой информации о том, как он используется при формировании адреса таблиц истории не приводится. Однако из его размера (32 бита) можно сделать вывод о том, что частично используются его старшие биты.

Процессоры Alpha и IBM используют комбинированные схемы, причем Alpha пыталась улучшить ее работу добавлением в одно ее плечо мажоритарной схемы, а IBM, поскольку технология позволяет, пошло по пути увеличения размера таблиц. ARM ограничилось использованием схемы gShare, компенсируя ее относительную простоту размером таблицы (до 16K записей). А MIPS, начав с использования bimodal схемы в ранних процессорах, остановился на мажоритарной схеме.

В результате проведенного анализа можно выделить несколько направлений, по которым нужно проводить оптимизацию схемы динамического предсказания при ее разработке:

1) проанализировать производительность наиболее часто применяемых схем: bimodal, gShare, комбинированной и мажоритарной для выбора оптимальной реализации.

2) Проанализировать влияние размера памяти для хранения таблиц BHT на точность работы схемы для обоснования его выбора.

3) Проанализировать влияние размера регистра глобальной истории, поскольку использование более ранней истории может положительно сказаться на точности предсказания.

IV. Блок динамического предсказания переходов, разрабатываемый для процессора 1890ВМ8

A. Требования к блоку предсказания переходов

На основе анализа существующих реализаций, описанных выше, блок должен реализовывать мажоритарную или комбинированную схему предсказания. Использование этих схем позволит отдельно проанализировать также схемы bimodal и gShare. Блок должен работать за 1 такт, обеспечивая одновременное предсказание двух команд условных переходов (из-за наличия в архитектуре MIPS слота задержки), формируя целевой адрес перехода для первой предсказанной инструкции ветвления.

B. Методика моделирования

Обычно в качестве модели для анализа схем динамического предсказания ветвления используются два метода: анализ трассы выполнения программы и использование потактовой С-модели. В первом случае используется модель предсказания ветвления, которая анализирует трассу выполнения программы, полученную на эмуляторе, выдавая предсказание и обновляя свою историю согласно логике своей работы. Во втором случае используется потактовая программная модель процессора, в которую добавляется блок предсказания, и на которой проводится запуск тестов для сбора статистики.

Оба метода имеют свои достоинства и недостатки. Анализ трассы наиболее простой для использования и быстрый с точки зрения времени моделирования метод, однако наименее точный: предсказание и обновление истории происходит для каждой инструкции ветвления, нет никакой информации об окне выполнения, задержках в запросах инструкций и данных. Ис-

пользование потактовой модели точнее, поскольку она содержит информацию об инструкциях, результат выполнения которых еще неизвестен, и о задержках, позволяя получить информацию о получившейся производительности процессора. Но основной вопрос состоит в точности и доступности этой потактовой модели.

Для разрабатываемого процессора существует покомандная С-модель, позволяющая получить трассу выполнения программы для анализа. Однако поскольку целью работы был не только анализ точности работы самой схемы предсказания, но его влияние на производительность и выбор оптимального варианта реализации, такой метод не давал нужной информации. Поэтому был выбран вариант модификации непосредственно RTL-модели процессора. С одной стороны, этот метод наиболее труден, но, учитывая необходимость реализации динамического предсказания, в данном случае был оправдан.

C. Структура разрабатываемого блока динамического предсказания переходов

При разработке блок сделан конфигурируемым с помощью конфигурационных регистров, доступных программно. Можно изменять следующие параметры:

1) вид предсказания (динамическое, статическое и отключено).

1) Тип динамического предсказания (мажоритарная схема, комбинированная схема, bimodal, gShare, см. рис. 4).

2) Алгоритм обновления двухбитного счетчика истории;

3) Размер памятей истории.

4) Варианты адресации памятей истории.

5) Размер глобальной истории (BHSR).

В результате блок может быть переконфигурирован для исследования большого числа вариантов схем динамического предсказания как при моделировании, так и в прототипе на ПЛИС.

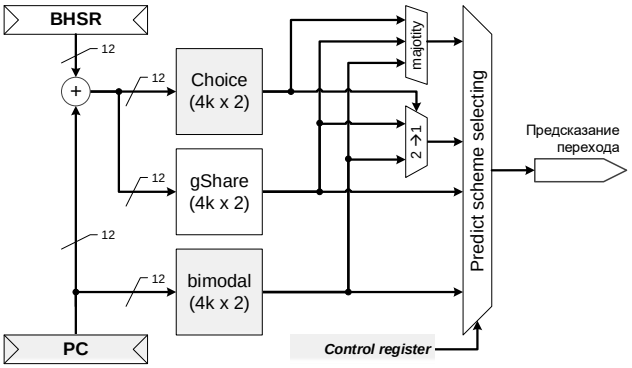

Рис. 4. Принципиальная схема выбора типа динамического предсказания

Из-за необходимости обеспечить работу блока за один такт локальная схема предсказания не использовалась. Схемы предсказания bimodal и gShare использовались для валидации собранных данных и самого метода анализа. Для ускорения сбора статистики количество счетчиков производительности было увеличено до 8, чтобы получать данные за один проход теста.

При моделировании использовались тесты производительности Coremark, Dhrystone, Whetstone; прикладные задачи умножения матриц, быстрого преобразования Фурье, архивирования, компиляции и сортировки. Длительность программ при моделировании составляла от 2 до 8 миллионов инструкций, для прототипа в ПЛИС – от 16 до 26 миллионов. Разница в полученных значениях между моделированием и прототипом составляет менее двух процентов.

V. Результаты моделирования

Для анализа работоспособности блока динамического предсказания и оценки эффективности его работы были промоделированы три варианта работы: с отключенными предсказаниями (все инструкции ветвления считались не предсказанными), со статическим и с динамическим предсказанием. Точность предсказания определялась как отношение правильно предсказанных инструкций ветвления к их общему количеству; значение IPC (instruction per cycle, количество команд за такт) – это отношение количества выполненных команд к длительности всего теста в тактах.

Результаты сравнения видов предсказания показаны на рис. 5. На графике видно, что рост точности динамического предсказания по сравнению со статическим на 35% дает увеличение IPC только на 10%. Это показывает важность контроля IPC при анализе различных вариантов схемы предсказания, т.к. попытки любыми способами улучшить только точность работы схемы, присутствующие в различных статьях, могут приводить к непропорционально низкому увеличению IPC – не выше погрешности измерений. Поэтому в данной статье для поиска оптимального решения оба параметра рассматриваются совместно.

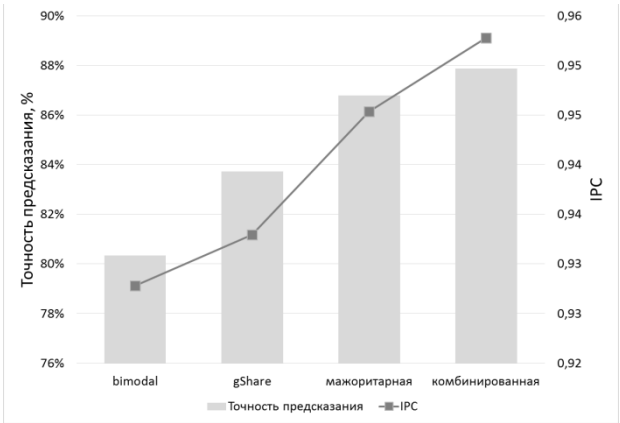

Рис. 5. Точность разных видов предсказаний и их влияние на IPC

Анализ типов схем динамического предсказания показан на рис. 6. Схемы предсказания bimodal и gShare имеют наименьшую точность, требуя при этом только одного банка памяти размером 8Кб (4К записей двухбитных счетчиков истории). Мажоритарная и комбинированная схемы работают несколько лучше:

требуя три банка памяти общим объемом 24Кб, они дают прирост точности предсказания по сравнению со схемой gShare на 3,5% и 4,7% соответственно. Но при этом рост IPC не так значителен, только 1,3% и 2%. Т.е. рост объема схемы более чем в 3 раза (память плюс логика хранения и обновления истории) дает увеличение производительности на проценты.

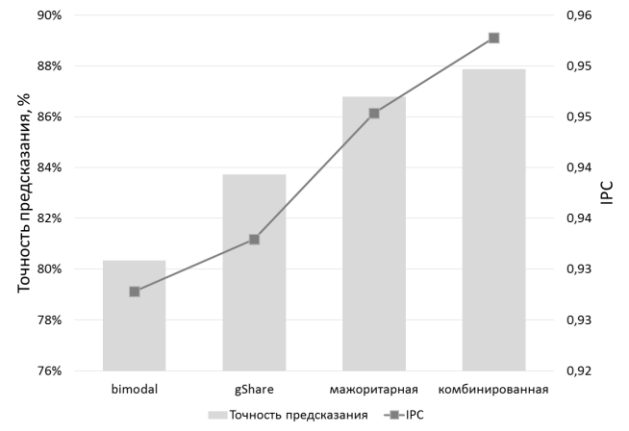

Рис. 6. Точность разных типов предсказаний

Влияние размера памяти на точность предсказания для этих четырех видов динамического предсказания показано на рис. 7. Размер памяти ограничивался от 16 до 4К записей маской на старшие разряды адреса. На графике видно, что точность комбинированной схемы выше остальных при любом размере памятей, а точность мажоритарной схемы выше gShare только начиная с размера памяти в 1К записей.

Для моделирования различных вариантов адресации памятей, использующих различную глубину глобальной истории, регистр BHSR был увеличен до 32 бит. Адрес памяти “gShare” (рис. 4) всегда формируется функцией XOR между 12 младшими битами PC и 12 битами глобальной истории, начиная с разрядов 0, 4, 8, 12, 16 и 20 (параметры *GS_ADDRESS_XX*). Для адресации памяти “choice” дополнительно используются варианты адресации непосредственно младшими 12 битами регистром BHSR и PC (параметры *CH_ADDRESS_GH* и *CH_ADDRESS_PC* соответственно) и функцией XOR, аналогичной для памяти “gShare”.

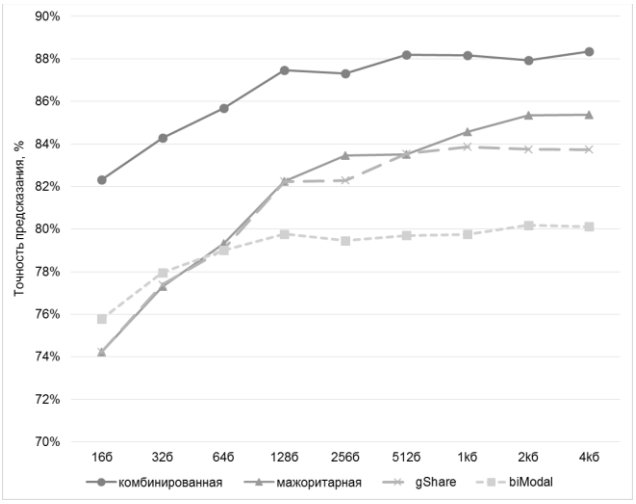

Рис. 7. Точность предсказания в зависимости от размера памятей

Для каждой точки пересечения этих параметров были собраны данные о точности работы, показанные на рис. 8. Из графика видно, что точность работы схемы падает при использовании в функции адресации более старой глобальной истории. Моделирование отдельных тестов с этими настройками схемы предсказа-

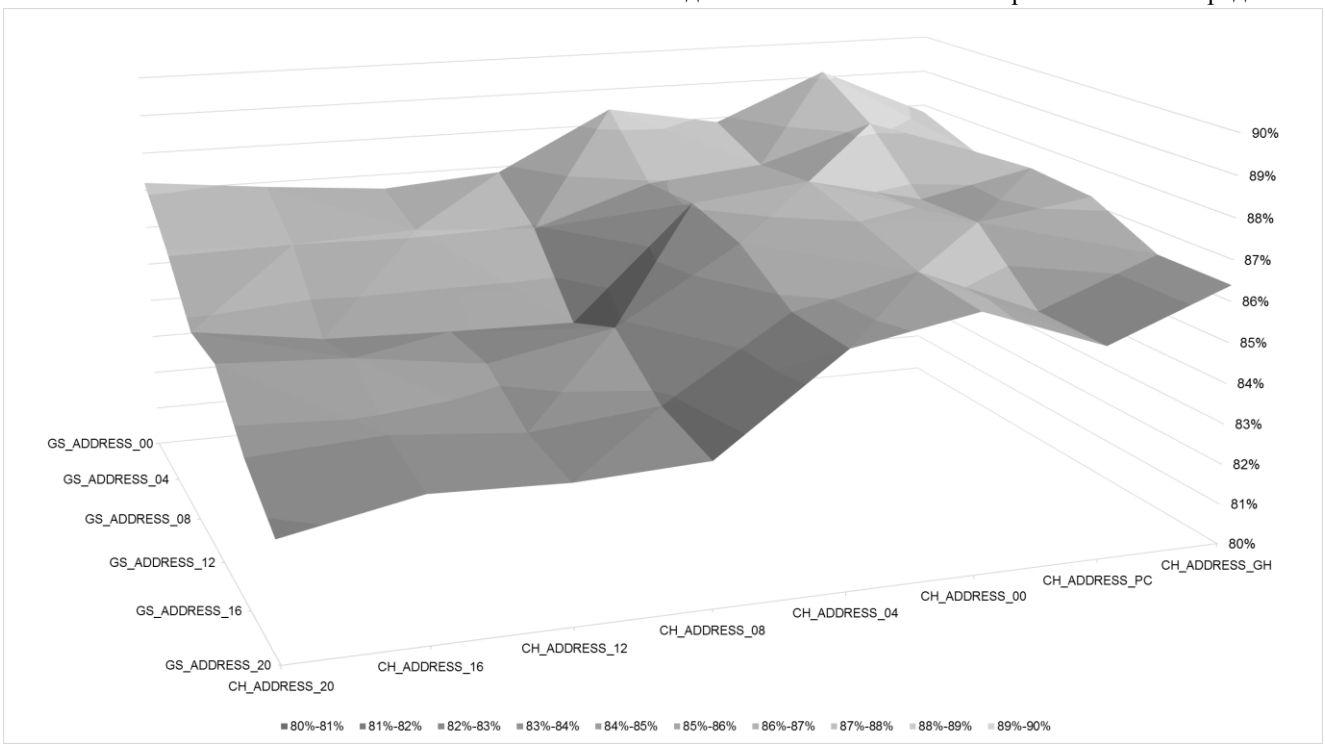

Рис. 8. Точность работы комбинированной схемы предсказания при различных функциях адресации памятей BHT

ния показало, что это связано с коротким конвейером и, соответственно, небольшим окном выполнения, в котором может находиться не более двух команд ветвления. В итоге для оптимальной работы блока достаточно хранить историю последних 20 команд условного перехода.

Другой особенностью работы схемы предсказания является то, что комбинированная схема лучше функционирует при адресации памяти выбора плеча предсказания (“choice” на рис. 4) значением PC. Т.е. выбор предсказания не зависит от предыдущей истории, и изменение адресации с исходной функции XOR (PC, BHSR) на адресацию только значением PC дает выигрыш в точности предсказания на 1,5%.

В некоторых статьях приводятся данные об изменении алгоритма работы двухбитного счетчика истории: вместо «классического» счетчика с насыщением предлагается использовать модифицированный – с пропуском некоторых состояний при определенных условиях. Собранная для предлагаемых вариантов статистика показана на рис. 9. Счетчик с насыщением отмечен «normal», для остальных вариантов отмечены переключения из начального в конечное состояние с пропуском промежуточного. Из графика видно, что любое изменение алгоритма обновления истории отрицательно сказывается на точности работы блока предсказания, поэтому для реализации в окончательном варианте был выбран счетчик с насыщением.

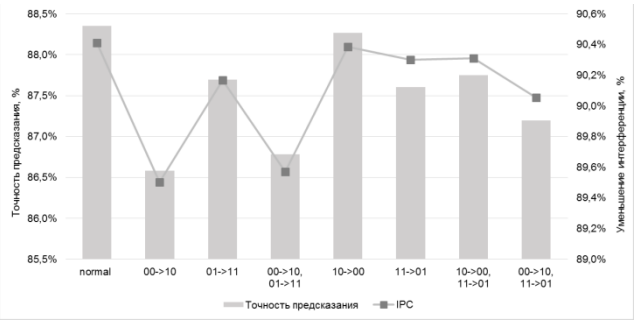

Рис. 9. Влияние изменения счетчика истории на точность предсказания переходов

Кроме абсолютных величин точности предсказания и IPC для различных функций адресации памятей была проанализирована частота обращений к различным ячейкам памяти. Разброс этих значений позволит оценить влияние выбранной функции на интерференцию, когда различные команды ветвления, находящиеся по различным адресам и имеющие различную историю выполнения, тем не менее отображаются на одну и ту же запись таблицы BHT. Чем меньше среднее абсолютных значений отклонений точек данных от среднего, тем равномернее используется память и тем устойчивее схема предсказания с выбранной функцией адресации к изменениям в исполняемом коде программ.

Полученная зависимость показана на рис. 10. Данные были получены моделированием тех же тестов, что и для ПЛИС, но меньшей длительности. Значения регистра BHSR использовались, начиная с 0, 4, 8 и 12 бит, т.к. при остальных параметрах падает точность работы схемы (см. рис. 8), лучшие параметры адресации перечислены в табл. 1.

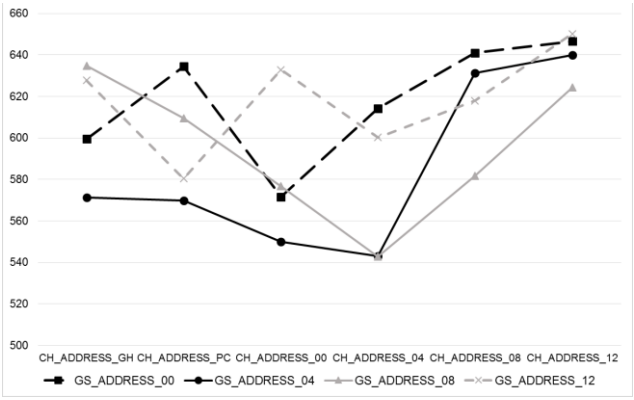

Рис. 10. Средний разброс количества обращений в ячейки BHT при разных функциях адресации

Для обоснования выбора параметров для использования в конечном варианте схемы динамического предсказания было оценено ухудшение точности относительно лучшего случая и уменьшение интерференции относительно него же (см. рис. 11). Из графика видно, что несмотря на большее падение точности для параметров функции адреса CH_ADDRESS_04/GS_ADDRESS_04 и CH_ADDRESS_04/GS_ADDRESS_08, значительно уменьшается разброс частоты обращений к различным записям таблиц BHT. На основании полученных данных для использования была выбрана схема адресации памяти “choice” – PC xor BHSR[15:4]; памяти “gShare” – PC xor BHSR[19:8].

Таблица 1

Лучшие параметры адресации памятей BHT для уменьшения интерференции

Функция адресации		**Точность предсказания**
“choice”	**“gShare”**	
PC	xor BHSR[11:0]	89,15 %
xor BHSR[11:0]	xor BHSR[15:4]	87,25 %
xor BHSR[15:4]	xor BHSR[15:4]	88,95 %
xor BHSR[15:4]	xor BHSR[19:8]	87,06 %

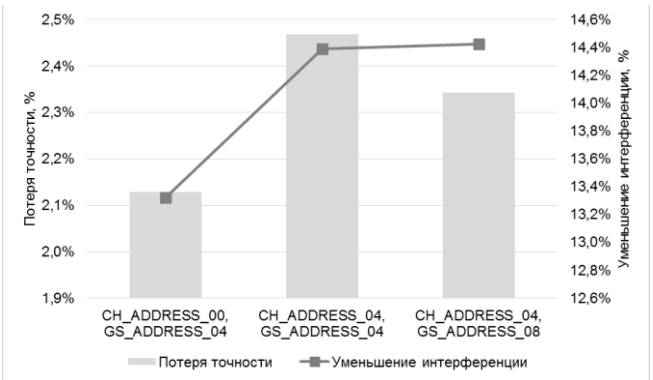

Рис. 11. Относительное ухудшение точности при уменьшении интерференции

VI. Заключение

В статье проведен анализ работы блока динамического предсказания ветвлений. Первоначальная реализация была выбрана на основании анализа схем динамического предсказания, известных из научных статей и применяющихся в коммерческих процессорах. Реализованная схема была сделана конфигурируемой для анализа точности работы различных ее вариантов и их влияния на производительность процессора.

Показавшая лучшие результаты комбинированная схема была проанализирована при изменении функций адресации памятей выбора и одного из плеч схемы. Для этого были собраны данные о количестве обращений к записям таблиц, оценен их разброс и проведено исследование качества разрабатываемой схемы. Показано, что при уменьшении точности предсказания можно добиться лучшей стабильности работы схемы.

Таким образом, схема динамического предсказания ветвления была оптимизирована для работы в составе процессора 1890ВМ8.

Литература

[1] S. McFarling. Combining Branch Predictors, Technical Report TN-36, 1993.

[2] P. Michaud, A. Seznec, and R. Uhlig. Trading Conflict and Capacity Aliasing in Conditional Branch Predictors // 24th Annual International Symposium on Computer Architecture. Denver, USA. 1997. pp. 292-303.

[3] E. Sprangle, R. S. Chappell, M. Alsup, and Y. N. Patt. The Agree Predictor A Mechanism for Reducing Negative Branch History Interference // 24th Annual International Symposium on Computer Architecture. Denver, USA. 1997. pp. 284-291.

[4] H. Vandierendonck. Improving the YAGS Branch Predictor, ELIS Technical Report, August 2005 // ELIS Technical Report. 2005., Ghent University, Ghent, BE, 2005-003, 2005.

[5] Y-C Maa, M-H Yen, Y-T Wang. Evaluating and improving variable length history branch predictors // International Computer Symposium (ICS). Tainan, TW. 2010. pp. 656-663.

[6] G. Hinton, D. Sager и др.. The Microarchitecture of the Pentium4 Processor // Intel Technology Journal, Vol. 1, Dec 2001. pp. 1-13.

[7] Agner Fog. The microarchitecture of Intel, AMD and VIA CPUs: An optimization guide for assembly programmers and compiler makers // Software optimization resources. 2016. URL: http://www.agner.org/optimize/microarchitecture.pdf (дата обращения: 10.02.2016).

[8] Software Optimization Guide for AMD Family 16h Processors // AMD. 2013. URL: http://developer.amd.com/wordpress/media/2012/10/SOG_16h_52128_PUB_Rev1_1.pdf (дата обращения: 10.02.2016).

[9] B. Sinharoy, R. Kalla, W. J. Starke и др. IBM POWER7 multicore server processor // IBM Journal of Research and Development, Vol. 55, No. 3, Jun 2011. pp. 1-29.

[10] Compaq. Alpha 21264 Microprocessor Hardware Reference Manual. 1999.

[11] A. Seznec, S. Felix, V. Krishnan, Y. Sazeides. Design tradeoffs for the alpha EV8 conditional branch predictor // 29th Annual International Symposium on Computer Architecture. Anchorage, AK. 2002. pp. 295-306.

[12] ARM. Cortex A9 Technical Reference Manual (revision: r4p1) // arm.com. 2012. URL: http://infocenter.arm.com/help/topic/com.arm.doc.ddi0388i/DDI0388I_cortex_a9_r4p1_trm.pdf (дата обращения: 10.02.2016).

[13] ARM. ARM® Cortex®-A72 MPCore Processor Technical Reference Manual (Revision r0p2) // www.arm.com. 2015. URL: http://infocenter.arm.com/help/topic/com.arm.doc.100095_0002_03_en/cortex_a72_mpcore_trm_100095_0002_03_en.pdf (дата обращения: 10.02.2016).

[14] MIPS Technologies, Inc. MIPS32® 1074K™ CPU Family Software User's Manual. 2011. 395 pp.

Research ways to design a dynamic branch prediction unit for promising microprocessor development by SRISA RAS

M.E. Barskikh

Scientific Research Institute for System Analysis RAS, barskikh@cs.niisi.ras.ru

***Keywords* — superscalar microprocessor, dynamic branch prediction, combining scheme, majority scheme, branch prediction performance.**

Abstract

The article describes the analysis of the dynamic branch prediction unit implementation options. A comparative study of the influence of circuit parameters on its accuracy has been demonstrated. We studied the effect of different schemes on the performance of microprocessors with RISC architecture to select the optimal implementation.

Branch and jump instructions in the program flow create control dependence, which determines the order of instruction execution in the pipeline. Therefore, using of dynamic branch prediction in modern processors becomes a required option. Combined or majority schemes which are based on combination of basic *bimodal* and *gShare* branch predictions are the most commonly used in commercial combined processors. Global branch history shift register (BHSR) is often implemented as 32-bit length to use earlier branch history in addressing of the branch history table (BHT).

Several areas of optimization have been allocated based on the analysis of known implementations. Optimi-

zation of dynamic prediction scheme in its development was carried out for selected areas and parameters. We evaluated performance of the most frequently used schemes: bimodal, gShare, combined, and majority; the effect of memory capacity for storing BHT-tables on the accuracy of the scheme; the impact of the global history register size.

The paper presents the branch prediction unit accuracy analysis and CPU performance. Investigation of instruction traces is not used, because it has low accuracy and does not provide information about the performance. Instead of that, we modified the processor RTL-model, which allows customizing branch prediction unit. For simulation, we used the following benchmarks: Coremark, Dhrystone and Whetstone, application programs of matrix multiplication, fast Fourier transformations, archiving, compilation and sorting.

Simulation analysis showed that separate bimodal and gShare prediction schemes have the lowest accuracy, but they require only a single 8Kb memory bank. The majority and combination prediction schemes work much better with memory requirement at 3 times more: they provide accuracy increase 3.5% and 4.7% respectively (comparing with the gShare scheme). However, at the same time, the IPC growth is only 1.3% and 2%. The size of the memory effect on prediction accuracy was analyzed as well. Memory size is limited from 16 to 4K entries by masking of the most significant bit address. These data allow selecting branch prediction implementation depending on limitations on its scope. The best choice is combination scheme, but we can use gShare scheme with a single 1Kb memory bank if necessary.

To simulate various memory addressing options, BHSR register was extended to 32 bits. The addresses of gShare and choice memories in the combined scheme are always formed XOR function between 12 lower PC bit and 12-bit global history, starting with the digits 0, 4, 8, 12, 16 and 20. For each point of these parameter intersections data about prediction accuracy have been collected, it is presented in this work. The data show that the accuracy of the scheme drops when the oldest global history in addressing functions is used. The best accuracy was obtained with using PC to address the choice memory and XOR function with the newest global history to address the gShare memory.

In addition, the absolute values of prediction accuracy and IPC for different memory addressing functions have been collected and the frequency of references to different BHT entries has been analyzed. The range of these values allows evaluating the influence of the selected addressing function on interference: when various different location branch instructions having different execution history, however, are mapped on the same BHT entry. The smaller range is more uniform memory used and it is more stable prediction scheme (with the selected addressing) for changes in the executable program code.

For explanation of the accuracy degradation choice, the addressing parameters on the best case was estimated and the reduction of interference with respect to it was done as well. It is shown that despite a slight drop in accuracy we can achieve better stability of the scheme. Finally, the addressing scheme of the choice and gShare memories to be used in CPU were selected based on the data obtained in the analysis. Thus, the dynamic branch prediction scheme has been optimized for using in the processor 1890VM8.

References

[1] S. McFarling. Combining Branch Predictors, Technical Report TN-36, 1993.

[2] P. Michaud, A. Seznec, and R. Uhlig. Trading Conflict and Capacity Aliasing in Conditional Branch Predictors // 24th Annual International Symposium on Computer Architecture. Denver, USA. 1997. pp. 292-303.

[3] E. Sprangle, R. S. Chappell, M. Alsup, and Y. N. Patt. The Agree Predictor A Mechanism for Reducing Negative Branch History Interference // 24th Annual International Symposium on Computer Architecture. Denver, USA. 1997. pp. 284-291.

[4] H. Vandierendonck. Improving the YAGS Branch Predictor, ELIS Technical Report, August 2005 // ELIS Technical Report. 2005., Ghent University, Ghent, BE, 2005-003, 2005.

[5] Y-C Maa, M-H Yen, Y-T Wang. Evaluating and improving variable length history branch predictors // International Computer Symposium (ICS). Tainan, TW. 2010. pp. 656-663.

[6] G. Hinton, D. Sager и др.. The Microarchitecture of the Pentium4 Processor // Intel Technology Journal, Vol. 1, Dec 2001. pp. 1-13.

[7] Agner Fog. The microarchitecture of Intel, AMD and VIA CPUs: An optimization guide for assembly programmers and compiler makers // Software optimization resources. 2016. URL: http://www.agner.org/optimize/microarchitecture.pdf (accessed: 10.02.2016).

[8] Software Optimization Guide for AMD Family 16h Processors // AMD. 2013. URL: http://developer.amd.com/wordpress/media/2012/10/SOG_16h_52128_PUB_Rev1_1.pdf (accessed: 10.02.2016).

[9] B. Sinharoy, R. Kalla, W. J. Starke и др. IBM POWER7 multicore server processor // IBM Journal of Research and Development, Vol. 55, No. 3, Jun 2011. pp. 1-29.

[10] Compaq. Alpha 21264 Microprocessor Hardware Reference Manual. 1999.

[11] A. Seznec, S. Felix, V. Krishnan, Y. Sazeides. Design tradeoffs for the alpha EV8 conditional branch predictor // 29th Annual International Symposium on Computer Architecture. Anchorage, AK. 2002. pp. 295-306.

[12] ARM. Cortex A9 Technical Reference Manual (revision: r4p1) // arm.com. 2012. URL: http://infocenter.arm.com/help/topic/com.arm.doc.ddi0388i/DDI0388I_cortex_a9_r4p1_trm.pdf (accessed: 10.02.2016).

[13] ARM. ARM® Cortex®-A72 MPCore Processor Technical Reference Manual (Revision r0p2) // www.arm.com. 2015. URL: http://infocenter.arm.com/help/topic/com.arm.doc.100095_0002_03_en/cortex_a72_mpcore_trm_100095_0002_03_en.pdf (accessed: 10.02.2016).

[14] MIPS Technologies, Inc. MIPS32® 1074K™ CPU Family Software User's Manual. 2011. 395 pp.