Аналитические модели надежности кэш-памяти

О.В. Мамутова

Аннотация. Кэш-память процессорной системы уязвима к одиночным сбоям в условиях воздействия ионизирующего излучения. На этапе разработки архитектуры системы необходимо проводить сравнительный анализ различных конфигураций кэш-памяти. В качестве средства для быстрого получения подобных оценок предложена аналитическая модель, описывающая ход вычислительного процесса с помощью модели независимых обращений. Рассмотрены случаи кэш-памяти, не реализующей мер повышения надежности, кэш-памяти с помехоустойчивым кодированием без и с исправлением обнаруженной ошибки, а также кэш-памяти с просеиванием. Определен показатель уязвимости как критерий оценки надежности процессорной системы в присутствии ошибок. Проведенный анализ подтвердил, что существующие для кэш-памяти эффекты самовосстановления не способны обеспечить необходимый уровень маскирования ошибок и требуется использование дополнительных мер повышения надежности. В целом, предложенный подход к аналитической оценке надежности позволяет провести более детальный анализ по сравнению с традиционными подходами статистического моделирования.

Ключевые слова: надежность, процессор, кэш-память, одиночный сбой, аналитическая модель, вычислительная нагрузка, уязвимость, просеивание, помехоустойчивое кодирование, размер памяти, ошибка, исправление, маскирование.

Введение

Кэш-память процессорной системы – это буферная память между процессором и основной памятью, уменьшающая среднее время обращений процессора в основную память. Обычно кэш-память занимает значительную часть процессорной системы.

Традиционно в теории надежности оценка безотказности систем проводится только на основе данных о вероятности отказов элементов, составляющих эту систему, и не учитывает возможность попадания ошибочных данных в вычислительный процесс и соответствующую вероятность информационного отказа.

Однако кэш-память является одним из самых уязвимых узлов процессорной системы для систем в критических применениях, для которых характерно возникновение одиночных сбоев в ячейках памяти (в англоязычной литературе *SEU – Single Event Upset*). Такие сбои являются одной из основных причин информационных отказов современных систем. При возникновении одиночного сбоя не происходит невосстанавливаемый отказ всего кристалла или ячейки памяти, а только инвертируются значения отдельных битов данных. Неверные данные хранятся в памяти до ближайшей операции записи. Но за время присутствия неверных данных в памяти может произойти их чтение, которое изменит ход вычислительного процесса в зависимости от семантической значимости прочитанных данных и, как следствие, приведет к информационному отказу всей системы.

В свете уязвимости процессорной системы к одиночным сбоям в кэш-памяти актуальна задача оценки надежности исполнения программ в таких условиях. В представленной работе поставлена задача разработки аналитических моделей поведения кэш-памяти при наличии ошибок с целью получения быстрых результатов при сравнительном анализе различных реализаций кэш-памяти.

1. Состояние вопроса

1.1. Методы оценки надежности

Имеющиеся результаты проведенных исследований показывают существенную зависимость оценки работы кэш-памяти при наличии ошибок от характера исполняемой вычислительной нагрузки. Действительно, распределение обращений в память во времени относительно момента возникновения сбоя, а также семантический аспект ошибочных данных определяют влияние возникшей вследствие сбоя ошибки на ход вычислительного процесса.

Для подобного анализа надежности используют статистические методы внесения неисправностей, суть которых состоит в многократном моделировании исполнения программы в присутствии ошибки, специально внедренной в случайный момент времени в случайный адрес в памяти. Примеры таких подходов: программа, моделирующая работу кэш-памяти для процессора *LEON* [1], эмуляция для процессора *PIC18* [2], внесение неисправностей на основе симулятора для архитектуры *SPARC* [3], ускоренная процедура анализа влияния ошибки в программе на лету [4], внесение неисправностей на базе симулятора для многопроцессорных систем [5].

Моделирование внесения неисправностей требует создания специального окружения по внесению неисправностей для каждой исследуемой системы. Главным недостатком подобного анализа является необходимость большого числа проходов моделирования для получения статистически значимых результатов. Аналитическая оценка вероятности информационного отказа процессорной системы вследствие возникновения ошибки в кэш-памяти позволит существенно сократить время проведения сравнительного анализа различных реализаций кэш-памяти на ранних этапах проектирования.

1.2. Предпосылки использования аналитической модели

Для перехода к аналитической модели можно воспользоваться стохастической моделью обращений процессора в кэш-память, когда адрес, характер и время до очередного обращения представлены случайными величинами. Такая модель позволяет рассматривать последовательность обращений процессора в кэш-память любого типа как поток событий, где случаен характер и адрес обращения. Подобные модели называются моделями независимых обращений (в англоязычной литературе – *Independent Reference Model, IRM*) и широко используются для анализа систем хранения при отсутствии данных о трассах - последовательностях обращений. Действительно, сбор достаточного количества подробных трасс для анализа вариантов организации кэш-памяти для различных архитектур процессора представляет собой неразрешимую задачу. Основанием для использования модели независимых обращений является наличие точной информации о временных характеристиках обращений в кэш-память и во внешнюю память.

Однако в реальных системах наблюдается пространственная и временная локальность обращений процессора в основную память. Это значит, что есть такие области памяти, обращения к которым происходят чаще. Известно, что определенная локальность наблюдается и в потоке обращений в кэш-память [7]. Однако с целью упрощения модели можно допустить, что обращения в кэш-память происходят случайно и равномерно по всем адресам. Тогда при размере кэш-памяти N слов при очередном обращении равновероятно обращение к любому адресу в целочисленном диапазоне $[0; N)$. В случае ярко выраженной локальности обращений в кэш-память можно отдельно рассматривать участки массива данных, для которых равномерная модель обращений оказывается приемлемой.

1.3. Исходные данные для построения математической модели

Для дальнейшего построения аналитической модели необходимо формальное описание используемой стохастической модели обращений процессора в кэш-память [8]. Традиционная модель дополнена коэффициентом, учитывающим увеличение времени доступа к кэш-памяти при реализации помехоустойчивого кодирования.

Принцип работы кэш-памяти заключается в прозрачном для процессора замещении обращений в медленную основную память на обращения в быструю, но меньшую по объему кэш-память, хранящую последние запрошенные

процессором данные. *Кэш-попаданием* называется обращение процессора, когда запрашиваемые данные находятся в кэш-памяти. *Кэш-промахом* называется обращение процессора, когда запрашиваемые процессором данные необходимо загрузить из основной памяти [6].

Используются два независимых пространства элементарных исходов: (запись, чтение) с вероятностью записи p_{store} и (кэш-промах, кэш-попадание) с вероятностью кэш-промаха p_{miss}. Например, вероятность того, что очередное обращение будет чтением с кэш-попаданием, равна $(1 - p_{store})(1 - p_{miss})$.

Общепринятый эмпирический закон «корень из двух» говорит о том, что вероятность кэш-промахов p_{miss} имеет степенную зависимость от размера кэш-памяти N, со степенью около –0,5. Необходимо учитывать, что эта зависимость меняется для разных размеров строки кэш-памяти. Далее размер строки считается неизменным и используется предложенный в работе [9] расширенный вариант этого эмпирического закона, с двумя параметрами k и a, где последний принимает значения в диапазоне [0,3; 0,7]:

$$p_{miss}(N) = k(N+1)^{-a}. \quad (1)$$

Определив возможные исходы случайных событий, получаем среднее время обращения процессора в память t_0:

$$t_0 = m \cdot t_{hit}(N) + t_{miss\_penalty} p_{miss}(N), \quad (2)$$

где $t_{miss\_penalty}$ – средний штраф за кэш-промах, зависящий от быстродействия памяти верхнего уровня, $t_{hit}(N)$ – время обращения в кэш-память при кэш-попадании, зависящее от размера кэш-памяти, m – коэффициент увеличения времени доступа в кэш-память при использовании помехоустойчивого кодирования.

Далее считается, что контроллер кэш-памяти непрерывно осуществляет обращения к кэш-памяти, т.е. среднее время между обращениями процессора к памяти равно t_0 – среднему времени обращения в память. Действительно, можно представить, что такое предположение верно для большого класса систем, когда память является узким местом по быстродействию, например, для систем обработки данных, непрерывно считывающих данные из/в память. В противном случае для возможности использования модели независимых обращений требуется коррекция выражения для среднего времени обращения процессора в память t_0.

2. Модель надежности кэш-памяти

Для борьбы с ошибками, возникающими в кэш-памяти, используют помехоустойчивое кодирование и просеивание. Реже используют повторное исполнение из контрольной точки и троирование, поскольку в обоих случаях велики аппаратные затраты. Далее описана и формализована общепринятая модель надежности, которая позволяет рассмотреть вероятность информационного отказа (далее – отказ) процессорной системы при возникновении ошибки в одном из слов кэш-памяти для кэш-памяти без и с помехоустойчивым кодированием.

Пусть объем кэш-памяти равен N слов. Пусть вследствие одиночного сбоя в кэш-памяти появляется слово с ошибкой. Ошибка **остается незамеченной** (в англоязычной литературе *latent*) до тех пор, пока не происходит обращение к этому слову или строке со словом. Когда же обращение к слову происходит, возможны три исхода: запись с кэш-попаданием, кэш-промах, чтение с кэш-попаданием.

Для кэш-памяти характерно **самовосстановление**, возникающее в ходе обращений процессора в двух случаях: 1) если происходит запись с кэш-попаданием, 2) если происходит кэш-промах. Во втором случае строка кэш-памяти размером *block_size*, содержащая рассматриваемое слово, перезаписывается безошибочными данными из внешней памяти верхнего уровня. Вероятность самовосстановления для рассматриваемого слова с ошибкой при очередном обращении процессора:

$$p_{correct}(N) = \frac{p_{store}\left(1 - p_{miss}(N)\right) + p_{miss}(N) \cdot block\_size}{N} \quad (3)$$

Если в системе не используются меры повышения надежности для борьбы с ошибками, то при чтении с кэш-попаданием ошибка попадает в вычислительный процесс и в зависимости от семантической значимости прочитанных

данных может исказить ход исполнения программы процессора и привести к **необнаруженному отказу** всей системы. Семантическая значимость данных в том или ином слове кэш-памяти полностью определяется текущей вычислительной нагрузкой. Например, ошибка в коде команды наверняка приводит либо к исполнению неверной команды, либо к исключительной ситуации. Ошибка же в данных может лишь незначительно повлиять на результат исполнения программы. Рассматривая худший случай, будем считать, что искажение данных, получаемых процессором из памяти, недопустимо и приводит к **информационному отказу**. Получаемая в результате пессимистичная оценка надежности системы может быть скорректирована масштабирующим коэффициентом при наличии результатов анализа *ACE*-битов (*ACE* – *Architecturally Correct Execution*) – битов, содержащих информацию, повреждение которой повлияет на полезный результат исполняемой программы [11].

Пусть в системе используется помехоустойчивое кодирование. Тогда в случае чтения с кэш-попаданием, если используемый код позволяет исправить накопившиеся в слове ошибки, происходит **маскирование** – передача процессору исправленных данных. Дополнительно может быть проведено **исправление** – обратная запись этих данных в память.

Если накопленные ошибки не могут быть обнаружены, то процессор получает незамаскированные данные с ошибкой. Если же число ошибок в слове может быть обнаружено, но не может быть исправлено, то такая ситуация вызывает исключение и должна быть обработана дополнительно. Например, для кэш-памяти со сквозной записью в памяти верхнего уровня всегда хранится копия данных без ошибки, поэтому возможно исправление. Но для упрощения модели будем считать, что в общем случае исправление невозможно. Поэтому обе ситуации, когда используемое помехоустойчивое кодирование не позволяет исправить ошибки, будем считать информационными отказами.

Вероятность чтения с кэш-попаданием для рассматриваемого слова с ошибкой при очередном обращении процессора:

$$p_{read}(N) = \frac{(1 - p_{store})(1 - p_{miss}(N))}{N}. \quad (4)$$

3. Аналитическая оценка надежности кэш-памяти

В качестве критерия надежности кэш-памяти будем использовать *показатель уязвимости* (в англоязычной литературе – *vulnerability factor*, *VF*), который определим как вероятность того, что возникшая вследствие одиночного сбоя ошибка приведет к отказу. Рассчитаем показатель уязвимости для разных конфигураций надежной памяти: без помехоустойчивого кодирования, с помехоустойчивым кодированием без исправления ошибки в памяти и с исправлением ошибки в памяти, а также с просеиванием.

3.1. Без помехоустойчивого кодирования

Отказ возникает, если с момента возникновения ошибки до ближайшего чтения не происходит самовосстановление. Условное графическое пояснение такой ситуации представлено на Рис. 1: звездочкой на оси времени t показан момент возникновения ошибки – начало отсчета; вертикальная стрелка показывает момент чтения с кэш-попаданием, адресующего слово с ошибкой; штриховка сверху оси времени показывает отсутствие операций чтения с кэш-попаданием; и штриховка снизу оси времени показывает отсутствие самовосстановления.

Получается, что отказ произойдет на i-ом обращении, если i-ое обращение окажется чтением с кэш-попаданием, при условии, что все предыдущие i–1 обращений не были ни чтением с кэш-попаданием, ни самовосстановлением. Вероятность чтения с кэш-попаданием p_{read} определяется по формуле (4). Вероятность самовосстановления $p_{recover}$ определяется по формуле (3). Для очередного обращения события чтения с кэш-попаданием и самовосстановления являются несовместными. Поэтому вероятность того, что очередное обращение не является ни чтением с кэш-попаданием, ни самовосстановлением, равна $1 - p_{read} - p_{recover.}$

В выбранной модели обращений процессора в память все обращения к памяти являются независимыми событиями. Вероятность появ-

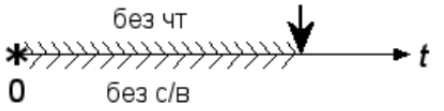


Рис. 1. Отказ для кэш-памяти с самовосстановлением

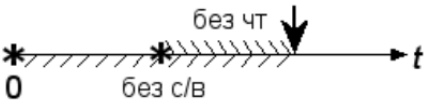


Рис. 2. Отказ для кэш-памяти с маскированием

ления последовательности независимых событий равна произведению вероятностей отдельных событий. Поэтому вероятность отказа на i-ом обращении определяется выражением

$$p(x=i) = p_{read}\left(1 - p_{read} - p_{recover}\right)^{i-1}.$$

Для упрощения расчетов произведем переход от дискретного времени к непрерывному. Для этого воспользуемся разложением в ряд Тейлора и при условии, что $x << 1$, $ix << 1$ и i – достаточно большое число, получим приближенное равенство $(1+x)^i \cong e^{ix}$. Тогда вероятность возникновения отказов в момент времени $T = it_0$, можно записать следующим образом:

$$p(T) = p_{read} e^{-(p_{read}+p_{recover})T/t_0}. \qquad (5)$$

Пусть время наблюдения от момента возникновения ошибки равно t. Тогда интегрируя имеющееся выражение по времени, получаем вероятность того, что появившаяся ошибка приведет к отказу за время наблюдения. Это определенный ранее показатель уязвимости.

$$VF(t) = \int_0^t p(T)\,dT / t_0 = \frac{p_{read}(N)}{p_{read}(N) + p_{recover}(N)} \times$$
$$\times \left(1 - e^{-\left(p_{read}(N)+p_{recover}(N)\right)t/t_0}\right). \qquad (6)$$

3.2. С помехоустойчивым кодированием и маскированием

Если слово содержит одну ошибку, то для этого слова при чтении с кэш-попаданием происходит только маскирование, и ошибка продолжает оставаться в памяти. Отказ возникает в том случае, если после возникновения второй ошибки в слове происходит чтение с кэш-попаданием, и при этом за все время после возникновения первой ошибки не произошло самовосстановление (Рис. 2).

Для описания процесса возникновения ошибок во времени можно воспользоваться общепринятым показательным распределением. Пусть интенсивность простейшего потока возникновения ошибок в рассматриваемом слове равна λ. Тогда функция распределения времени возникновения ошибки записывается как $F(x) = 1 - e^{-\lambda x}$ и плотность распределения как $f(t) = \lambda e^{-\lambda t}$. Тогда вероятность возникновения ошибки между обращениями $(j-1)$ и j будет равна:

$$p_{err}(j) = F(jt_0) - F((j-1)t_0) = e^{-\lambda(j-1)t_0} - e^{-\lambda j t_0}.$$

Усреднив вероятность безотказной работы при наличии повторной ошибки за i шагов, получаем вероятность отказа на i-ом обращении, когда происходит чтение с кэш-попаданием:

$$p(i) = p_{read}\sum_{j=1}^{i} p_{err}(j)\left(1-p_{recover}\right)^{j}\left(1-p_{recover}-p_{read}\right)^{i-j};$$

и при переходе к непрерывному времени вероятность отказа в момент времени $T = it_0$ равна (аналогично переходу к формуле (5)):

$$p(T) = p_{read}\int_0^T f(t)\, e^{-\frac{p_{recover}}{t_0}t}\, e^{-\frac{p_{read}+p_{recover}}{t_0}(T-t)}\,dt. \qquad (7)$$

Выражение для показателя уязвимости в случае реализации помехоустойчивого кодирования без исправления ошибки в памяти имеет вид:

$$VF(t) = \int_0^t p(T)\,dT / t_0 = p_{read}\frac{\lambda t_0}{p_{read} - \lambda t_0} \times$$
$$\times \left(\frac{1 - e^{-\frac{p_{recover}+\lambda t_0}{t_0}t}}{p_{recover} + \lambda t_0} - \frac{1 - e^{-\frac{p_{recover}+p_{read}}{t_0}t}}{p_{read} + p_{recover}} \right). \qquad (8)$$

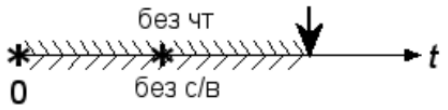


Рис. 3. Отказ для кэш-памяти с исправлением ошибки в памяти

3.3. С помехоустойчивым кодированием и исправлением

Отказ произойдет, если между моментом появления первой ошибки и следующим чтением с кэш-попаданием появится еще одна ошибка, при этом за все время не произойдет ни одного самовосстановления (Рис. 3).

Запишем вероятность отказа в момент T сразу в непрерывном времени:

$$p(T) = p_{read}\left(1-e^{-\lambda T}\right)e^{-\frac{p_{read}+p_{recover}}{t_0}T}.$$

Тогда получаем выражение для показателя уязвимости в случае реализации помехоустойчивого кодирования с исправлением:

$$VF(t) = \int_0^t p(T)\,dT/t_0 =$$

$$= p_{read}\left(\frac{1-e^{-\frac{p_{read}+p_{recover}}{t_0}t}}{p_{read}+p_{recover}} - \frac{1-e^{-\frac{p_{read}+p_{recover}+\lambda t_0}{t_0}t}}{p_{read}+p_{recover}+\lambda t_0}\right).$$

3.4. С помехоустойчивым кодированием и просеиванием

Просеивание – это процедура периодического опроса адресов массива памяти с исправлением обнаруженных ошибок. Считаем, что чтение с кэш-попаданием для слова с ошибкой только маскирует, но не исправляет эту ошибку. Пусть период просеивания равен T_s и пусть первая ошибка в рассматриваемом слове возникает в течение этого периода, в момент t_e. Если в слове накопились 2 ошибки, отказ произойдет или при чтении с кэш-попаданием в ходе выполнения программы, или при очередном опросе процедуры просеивания:

1. после возникновения первой ошибки к концу периода просеивания произошла еще одна ошибка, но не произошло ни одного самовосстановления и после второй ошибки не было ни одного чтения – так как две ошибки не могут быть исправлены (Рис. 4 а);

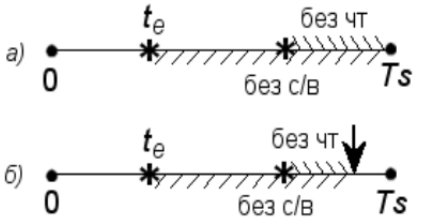


Рис. 4. Отказ для кэш-памяти с просеиванием

2. после возникновения второй ошибки произошло чтение, но после первой ошибки до чтения не было ни одного самовосстановления (Рис. 4 б).

Используя выражение (7), получаем вероятность исходов из первой группы:

$$p_{f1}(T_s,t_e) = \int_0^{T_s-t_e} f(t)e^{-\frac{p_{recover}}{t_0}t}e^{-\frac{p_{read}+p_{recover}}{t_0}(T_s-t_e-t)}dt,$$

и используя выражение (8), получаем вероятность исходов из второй группы:

$$p_{f2}(T_s-t_e) = \frac{p_{read}\lambda t_0}{p_{read}-\lambda t_0}\times$$

$$\times\left(\frac{1-e^{-\frac{p_{recover}+\lambda t_0}{t_0}(T_s-t_e)}}{p_{recover}+\lambda t_0} - \frac{1-e^{-\frac{p_{recover}+p_{read}}{t_0}(T_s-t_e)}}{p_{read}+p_{recover}}\right).$$

Считая, что первая ошибка может возникнуть в любой момент интервала T_s, интегрируем сумму вероятностей нежелательных исходов по t_e и получаем выражение для показателя уязвимости в случае реализации периодического просеивания:

$$VF = \frac{1}{T_s}\int_0^{T_s}\left(p_{f1}(T_s-t_e)+p_{f2}(T_s-t_e)\right)dt_e. \quad (9)$$

Для сокращения изложения итоговые выражения с результатами интегрирования опущены.

4. Анализ надежности кэш-памяти

Проанализируем зависимость показателя уязвимости от размера кэш-памяти N и времени

Табл. 1. Значения параметров при расчёте надёжности кэш-памяти

Вероятность записи, p_{store}	Параметры степенной функции вероятности промаха в выражении (1), k и a соответственно		Коэффициент увеличения времени обращения в кэш-память при использовании помехоустойчивого кодирования в выражении (2), m	Штраф при кэш-промахе, $t_{miss\_penalty}$, (с)	Размер строки кэш-памяти, *block_size*, (слов)	Интенсивность простейшего потока возникновения ошибок в слове в выражении (8), λ	Период просеивания в выражении (9), T_s, (с)
0,3	2	0,5	1,95	10^{-7}	16	10^{-4}	5

наблюдения t. Для этого определим значения остальных параметров предложенных моделей.

Соотношение в исполняемой программе между командами загрузки (*load*) и записи (*store*) данных в память зависит от типа вычислительной нагрузки, от системы команд процессора и процесса компиляции. В работе [13] на примере набора тестовых программ SPEC CPU2006 для двух разных процессоров с архитектурой RISC и CISC, исполняющих как целочисленные программы, так и программы с плавающей запятой, получена примерно одинаковая оценка процента команд записи от 13% до 18%. Для целей анализа далее использована величина $p_{store} = 30\%$.

Форма степенной зависимости $p_{miss}(N)$ вероятности кэш-промахов от размера кэш-памяти определяется в выражении (1) двумя параметрами: k и a. Величина параметра a выбрана как среднее из допустимого диапазона. Величина k выбрана так, чтобы обеспечить допустимое значение вероятности (не более 1) на выбранном диапазоне изменения N.

Также введен коэффициент m увеличения времени доступа в кэш-память в зависимости от используемого помехоустойчивого кодирования. Использование помехоустойчивого кодирования может увеличивать время доступа почти в два раза [10]. Величина параметра m выбрана близкой к этому значению.

Время обращения в кэш-память при кэш-попадании t_{hit} зависит от размера кэш-памяти, и такая зависимость может быть получена экспериментально. В представленных результатах использованы аппроксимированные степенной зависимостью значения для кэш-памяти прямого отображения с 16-ти байтными строками (*block_size*) при проектной норме 90 нм (результат получен с помощью модели *CACTI* http://quid.hpl.hp.com:9081/cacti). Величина t_{hit} при этом меняется в диапазоне от нескольких единиц до нескольких десятков нс.

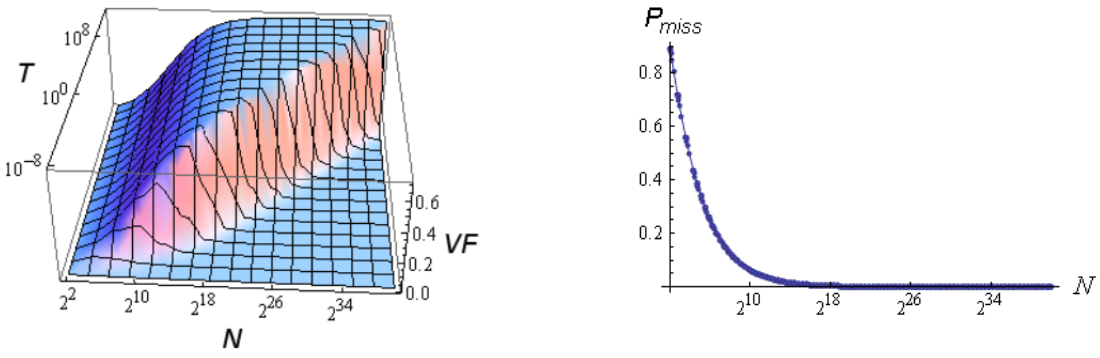
Средний штраф за кэш-промах $t_{miss\_penalty}$ обычно составляет от 10 до 100 тактов в зависимости от памяти верхнего уровня. Для расчетов выбрано значение 100 нс.

Все использованные величины параметров модели приведены в Табл. 1.

Для памяти с самовосстановлением значение показателя уязвимости определяется двумя процессами: самовосстановление при кэш-промахах/записи и чтение ошибочных значений при кэш-попаданях. Зависимость показателя уязвимости от времени наблюдения (T) и размера кэш-памяти (N), полученная по формуле (6) и представленная на Рис. 5, имеет вид треугольной призмы.

При малых размерах кэш-памяти N вероятность кэш-промахов p_{miss} велика, значит, велика вероятность самовосстановления, следовательно, показатель уязвимости VF мал. С увеличением размера кэш-памяти число кэш-промахов уменьшается, следовательно, показатель уязвимости растет. Но с дальнейшим увеличением размера памяти падает вероятность чтения ячейки с ошибкой за ограниченное время наблюдения, и показатель уязвимости вновь уменьшается.

Если рассмотреть процесс во времени, видно, что при фиксированном размере памяти показатель уязвимости возрастает до максимума, стремящегося к выражению $1 - p_{store}$ (Рис. 5 а, значение 0,7 по оси VF). Действительно, чем дольше ошибка хранится в памяти, тем выше вероятность обращения к слову с ошибкой, но отказ при чтении произойдет, только если это чтение с кэш-попаданием. Поэтому чем больше величина N,



а) б)

Рис. 5. Результаты моделирования

а) график изменения показателя уязвимости VF для кэш-памяти с самовосстановлением в зависимости от размера кэш-памяти и времени наблюдения
б) график зависимости вероятности кэш-промахов pmiss от размера кэш-памяти

тем позже наступает этот момент. При этом при меньших значениях N рост наблюдается раньше, но не до максимального значения.

Ожидаемо, что при больших размерах кэш-памяти и небольшом времени наблюдения показатель уязвимости мал, поскольку вероятность обращения к слову с ошибкой мала. Также можно сделать заключение, что *дополнительные меры повышения надежности не требуются только для малых размеров кэш-памяти, когда вероятность кэш-промаха велика и ошибка исправляется за счет самовосстановления*.

Для целей анализа диапазон изменения N и T на Рис. 5 намеренно превышает реально возможные значения. Так, в реальности размер кэш-памяти составляет от 2^4 до 2^{18} слов, а время наблюдения – от 10^{-2} до 10^8 с. *Поэтому фактически показатель уязвимости никогда не будет зависеть от времени наблюдения, а только от размера кэш-памяти*. Получаемый график зависимости представлен на Рис. 6.

Для вариантов памяти с помехоустойчивым кодированием на графиках, аналогичных представленному на Рис. 5, при больших значениях N и T также можно наблюдать ступень в виде треугольной призмы. Но для реалистичных значений аргументов, как и в предыдущем случае, показатель уязвимости не зависит от времени наблюдения. Также, по определению, от времени не зависит показатель уязвимости для кэш-памяти с просеиванием. На Рис. 7 представлены зависимости для этих трех конфигураций: с помехоустойчивым кодированием ($VF1$), с помехоустойчивым кодированием

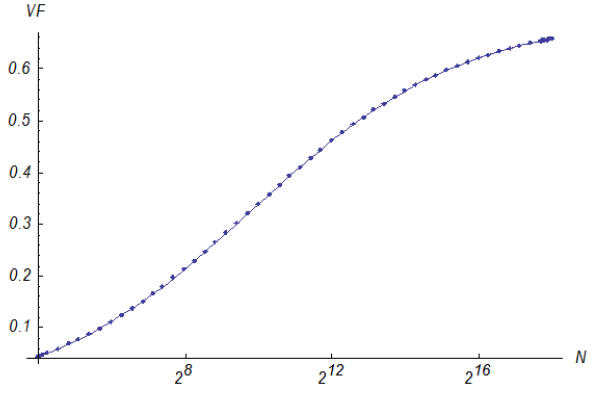


Рис. 6. Зависимость показателя уязвимости от размера кэш-памяти

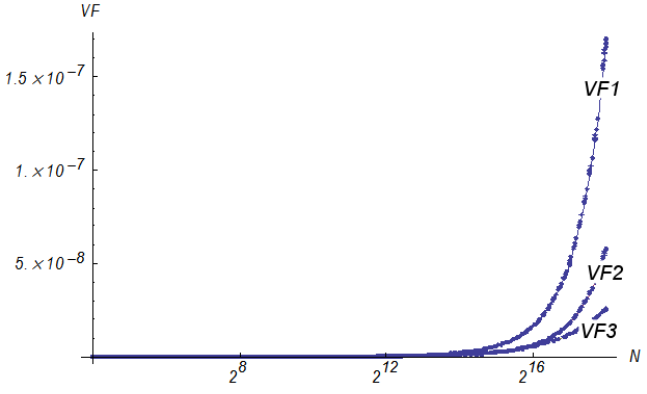


Рис. 7. Зависимость показателя уязвимости от размера кэш-памяти для трех конфигураций кэш-памяти с помехоустойчивым кодированием

и исправлением обнаруженной ошибки (*VF*2) и с просеиванием (*VF*3). На представленном графике зависимость *VF*3 получена для небольшого периода просеивания (5 с). Для длительных периодов просеивания кривая *VF*3 стремится к кривой *VF*1. При сравнении Рис. 6 и Рис. 7 видно, что использование помехоустойчивого кодирования с обнаружением одной ошибки дает выигрыш по надежности в семь порядков.

Заключение

Решение задачи оценки надежности процессорной системы в условиях возникновения ошибок в кэш-памяти обычно решается статистическими методами внесения неисправностей, требующими больших инженерных затрат на создание экспериментального окружения и временных затрат на проведение экспериментов для получения статистически значимых результатов. В отличие от статистических методов моделирования представленный в статье подход обеспечивает быструю аналитическую оценку надежности процессора с кэш-памятью с различными вариантами реализации методов повышения надежности в условиях возникновения одиночных сбоев.

Точность получаемых оценок определяется используемой в качестве исходных данных аналитической характеристикой исполняемой вычислительной нагрузки, определяющей вероятность кэш-промахов в зависимости от параметров кэш-памяти – размера и типа. Остальными входными данными для построенной модели являются размер кэш-памяти, размер строки кэш-памяти, вероятность операции записи, интенсивность возникновения ошибок, временные характеристики доступа в кэш-память и во внешнюю память.

Представленный подход может быть использован как инструмент прогнозирования надежности выбранной конфигурации кэш-памяти при заданных условиях работы, который в отличие от экспериментальных подходов позволяет провести анализ влияния входных данных на получаемые результаты.

Литература

1. Faure F., Velazko R., Violante M., Rebaudengo M., Reorda M.S. Impact of data cache memory on the Single Event upset-induced error rate in microprocessors // IEEE transactions on nuclear science, Vol. 50, No. 6, December 2003 — IEEE, 2003. — Pp. 2101-2106.
2. Garcia-Valderas M., Portela-Garcia M., Lopez-Ongil C., Entrena L. In-depth analysis of digital circuits against soft errors for selective hardening // 15th IEEE International On-Line Testing Symposium IOLTS 2009 — IEEE, 2009. — Pp. 144-149.
3. Rebaudengo M. An accurate analysis of the effects of soft errors in the instruction and data caches of a pipelined microprocessor // Design, automation and test in Europe conference and exhibition, 2003 — Washington, DC, USA: IEEE Computer society, 2003. — Pp. 602-607.
4. Li X., Adve S.V., Bose P., Rivers J.A. Online Estimation of Architectural Vulnerability Factor for Soft Errors // ISCA '08 Proceedings of the 35th Annual International Symposium on Computer Architecture — Washington, DC: IEEE Computer Society, 2008. — Pp. 341-352.
5. Beltrame G., Bolchini C., Fossati L., Miele A., Sciuto D. A framework for reliability assessment and enhancement in multi-processor Systems-on-Chip // 22nd IEEE international symposium on defect and fault tolerance in VLSI systems — IEEE Computer society, 2007. — Pp. 132-140.
6. Fricker C., Robert F. An Analytical cache model, Research Report RR-1496, INRIA Rocquencourt, France, 1991. — 28 p.
7. Agarwal A., Hennessy J., Horowitz M. An Analytical Cache Model // ACM Transactions on Computer Systems, Vol. 7, No. 2, 1989. — Pp. 184-215.
8. Мелехин В.Ф., Павловский Е.Г. Вычислительные машины. — М.: Издательский центр "Академия", 2013. — 384 с.
9. Hartstein A., Srinivasan V., Puzak T.R., Emma P.G. On the Nature of Cache Miss Behavior: Is it $\sqrt{2}$? // Journal of Instruction-Level Parallelism, No. 10, 2008. — Pp. 1-22.
10. Novac O., Vlădutiu M., Vari Kakas St., Novac M., Gordan M. A Comparative Study of Simulation Program for Cache Memory Performance Assessment // Journal of Computer Science and Control Systems, Vol. 2, Iss. 2 — Oradea, Romania: Editura Universitatii din Oradea, 2009. — Pp. 39-42.
11. Isen C., John L.K., John E. A Tale of Two Processors: Revisiting the RISC-CISC Debate. In Computer Performance Evaluation and Benchmarking, Lecture Notes in Computer Science, Volume 5419, 2009. — Pp. 57-76.
12. Li J.-F., Huang Y.-J. An Error Detection and Correction Scheme for RAMs with Partial-Write Function // IEEE International Workshop on Memory Technology, Design, and Testing (MTDT'05) 2005. — Pp. 115–120.

Мамутова Ольга Вячеславовна. Старший преподаватель Санкт-Петербургского государственного политехнического университета Петра Великого (СПбПУ). Окончила СПбПУ в 2008 году. Автор 30 печатных работ. Область научных интересов: архитектуры и надежность вычислительных систем, программируемая логика, реконфигурируемые аппаратно-программные системы. E-mail: mamoutova@kspt.icc.spbstu.ru