© 2012 г. Б.А. ЗЫРЯНОВ, д-р техн. наук,
Н.В. СТРЕЛЬЦОВ
(ОАО «Мультиклет», Екатеринбург)

ОБЕСПЕЧЕНИЕ ЖИВУЧЕСТИ МУЛЬТИКЛЕТОЧНОГО ПРОЦЕССОРА

Описываются особенности архитектуры мультиклеточных процессоров, позволяющие обеспечить их живучесть. Предлагается ряд аппаратных и программных решений для реализации этой возможности.

LIVENESS OF MULTICELLULAR PROCESSOR / B.A. Zyryanov, N.V. Streltsov (MultiClet corp., of. 133, Cheluskintsev str. 2, Ekaterinburg, 620014, Russia, E-mail: micron@fointec.ru, synputer@fointec.ru). There are described features of multicellular processors architecture which allow to ensure their liveness. There is proposed a set of hardware and software solutions for realization of this possibility.

1. Введение

Характерной особенностью мультиклеточной архитектуры является полная независимость объектного программного кода от количества клеток. Одна и та же программа, без перекомпиляции, может быть выполнена на любом количестве клеток. Эта особенность потенциально обеспечивает живучесть мультиклеточного процессора, т.е. возможности непрерывного исполнения программы при отказах его отдельных клеток (деградации процессора). Очевидно, что деградация процессора связана с потерей производительности и, соответственно, с увеличением времени решения задач. Но, для целого ряда встроенных применений, живучесть мультиклеточного процессора позволяет управляемому объекту выполнять основные функции, либо за счет снижения их качества, либо за счет отказа от решения второстепенных задач.

2. Постановка задачи

Структурно (см. рис.1), выпускаемый в настоящее время, мультиклеточный процессор представляет собой параллельную процессорную систему, состоящую из n клеток и общих ресурсов (блок регистров общего назначения(GPR), n-блочная память данных(DM), периферийные устройства). Каждая клетка, в свою очередь, состоит из простейшего процессорного устройства, включающего: устройство управления(CU), буфер команд(BUF), исполнительные устройства(EU), память программ(PM), и узла коммутации(SU). Узлы коммутации образуют общую коммутационную среду[1].

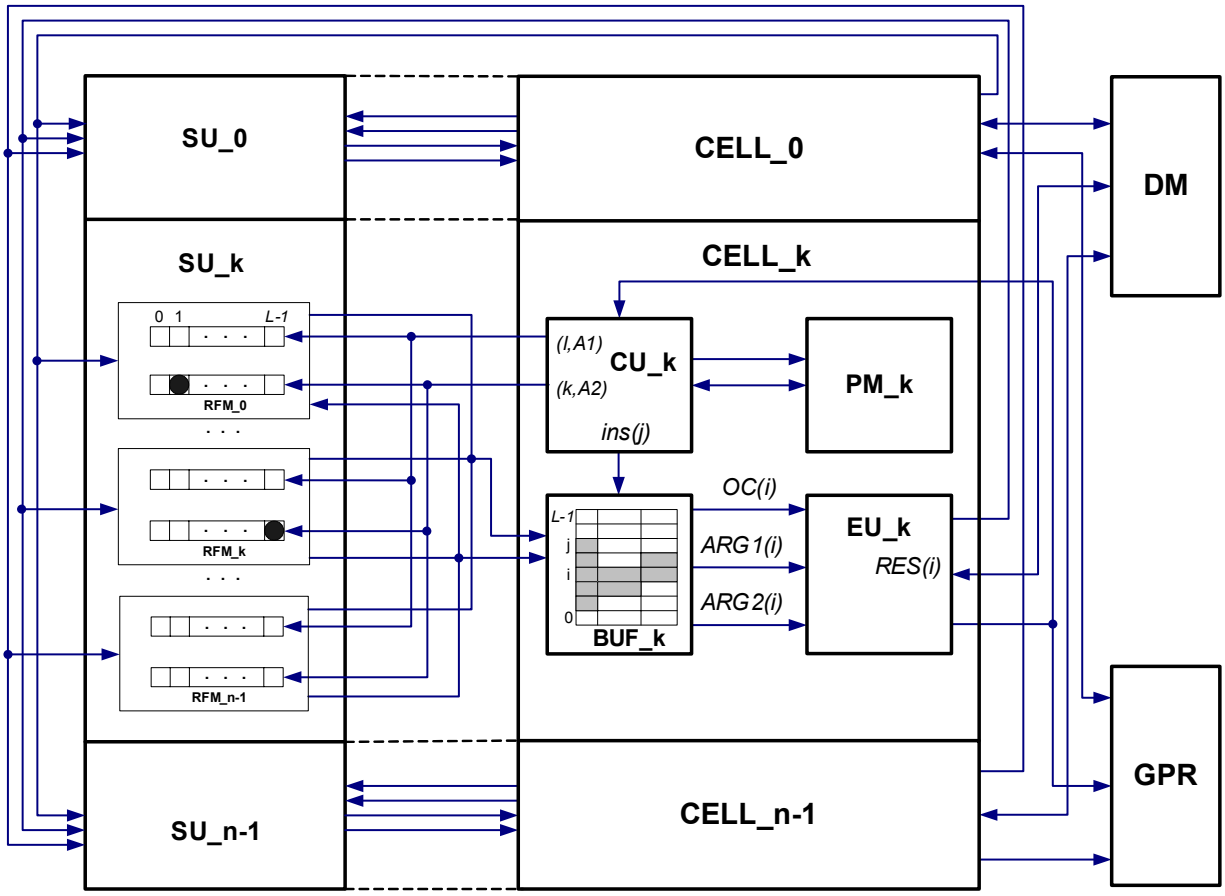

Рис.1. Структура мультиклеточного процессора

Информационный обмен между командами – непосредственный, реализованный, в отличие от потоковых процессоров, с использованием не адресной, а широковещательной рассылки результатов выполнения команд. При данном способе рассылки исполняемая программа представляет собой набор параграфов, каждый из которых является частично упорядоченной последовательностью команд, а именно, команда-источник результата размещен раньше команды-приемника. Непосредственные информационные связи допустимы только между командами одного параграфа и задаются в командах-приемниках ссылками на команды–источники результатов. Значение ссылки – это номер команды–приемника относительно команды– источника. Информационная связь между параграфами – опосредована, через память(GPR или DM).

Команды параграфа размещаются последовательно группами по n команд в РМ клеток с одного начального адреса. Начальная команда параграфа размещается в РМ нулевой клетки, следующая в РМ первой и т.д. Выбираются команды клетками также последовательно группами по n команд. Сначала одновременно выбираются первые n команд, потом вторые и т.д. Очередной выбранной группе присваивается очередное значение тега группы (T_g). Минимальное значение тега группы определяется максимальным значением ссылки (окном видимости). Выбранные команды размещаются в буферах клеток до исполнения. Каждая выбранная команда и, соответственно, ее результат именуются, а

именно, ее результату присваивается индивидуальный номер равный $T_g\&N_c$, где N_c – физический номер клетки, выбравшей данную команду – число в диапазоне от 0 до $n-1$. По значению ссылки формируются номера запрашиваемых результатов, которые сообщаются коммутационному узлу клетки. Для мультиклеточных процессоров с $n = 2**k$ и частично упорядоченным размещением команд (команда- источник размещается раньше команды-приемника), номер запрашиваемого результата определяется следующим образом:

$$N_r = ((T_g\&N_c)\text{-}P_{op}) mod (T_g^{max}) \quad (1)$$

где: N_r - номер запрашиваемого результата; T_g^{max} – максимальное значение тега группы; P_{op} – значение ссылки. Следует отметить, что младшие k бит N_r содержат физический номер клетки от которой должен придти запрашиваемый результат и который используется при выдаче запроса узлу коммутации на отбор результата (значения $A1$ или $A2$).

Каждый полученный результат поступает всем коммутационным узлам. Коммутационные узлы по номеру отбирают из общего, потока требуемые данной клетке результаты, и передают их процессорному устройству. Процессорное устройство исполняет команду, когда выполнены два условия – получены запрошенные результаты и выбраны все команды–приемники ее результата.

В общем случае вычислительный ресурс мультиклеточного процессора может использоваться с различными целевыми установками. Например, все ресурсы (все клетки) могут быть направлены на решение одной задачи или её фрагмента. Ресурсы также могут быть разделены между несколькими задачами, например для решения основной задачи управления и параллельного контроля целостности аппаратного контура, вычислительного процесса или оценки получаемых результатов[2]. Также эти два подхода могут быть использованы неоднократно (на различных интервалах) при решении одного и того же комплекса задач. Подобная динамическая реконфигурация и, как следствие, обеспечение живучести процессора возможны только при выполнении следующих условий:

1) Все общие ресурсы процессора, за исключением периферии, должны быть растиражированы и включены в состав клеток.
2) Должна быть аппаратно поддержана возможность формирования запросов к коммутационному узлу при $n \neq 2**k$.
3) РМ каждой клетки должна содержать все программное обеспечение, а не только его часть, и обеспечивать согласованную выборку команд при любом количестве клеток.
4) Должна быть аппаратно поддержана возможность синхронной выборки команд частью клеток, вне зависимости от состояния других.
5) Должно быть аппаратно поддержано управляемое тиражирование данных, поступающих в DM и GPR.

3. Реализация динамической реконфигурации

3.1. Структура реконфигурируемого мультиклеточного процессора

Структура реконфигурируемого мультиклеточного процессора приведена на рис.2. В результате тиражирования общих ресурсов(кроме периферии) каждая клетка превращается в функционально полное процессорное устройство, которое может работать в любом сочетании с другими клетками, т.е. в составе любой группы клеток, как единый мультиклеточный процессор. При этом каждая клетка имеет возможность записи в память всех других клеток, но чтение разрешено только из своих блоков памяти.

Реализация периферии существенно зависит от общей организации системы управления и необходимость ее тиражирования определяется, в первую очередь, не вычислительным ядром а особенностями системы в целом.

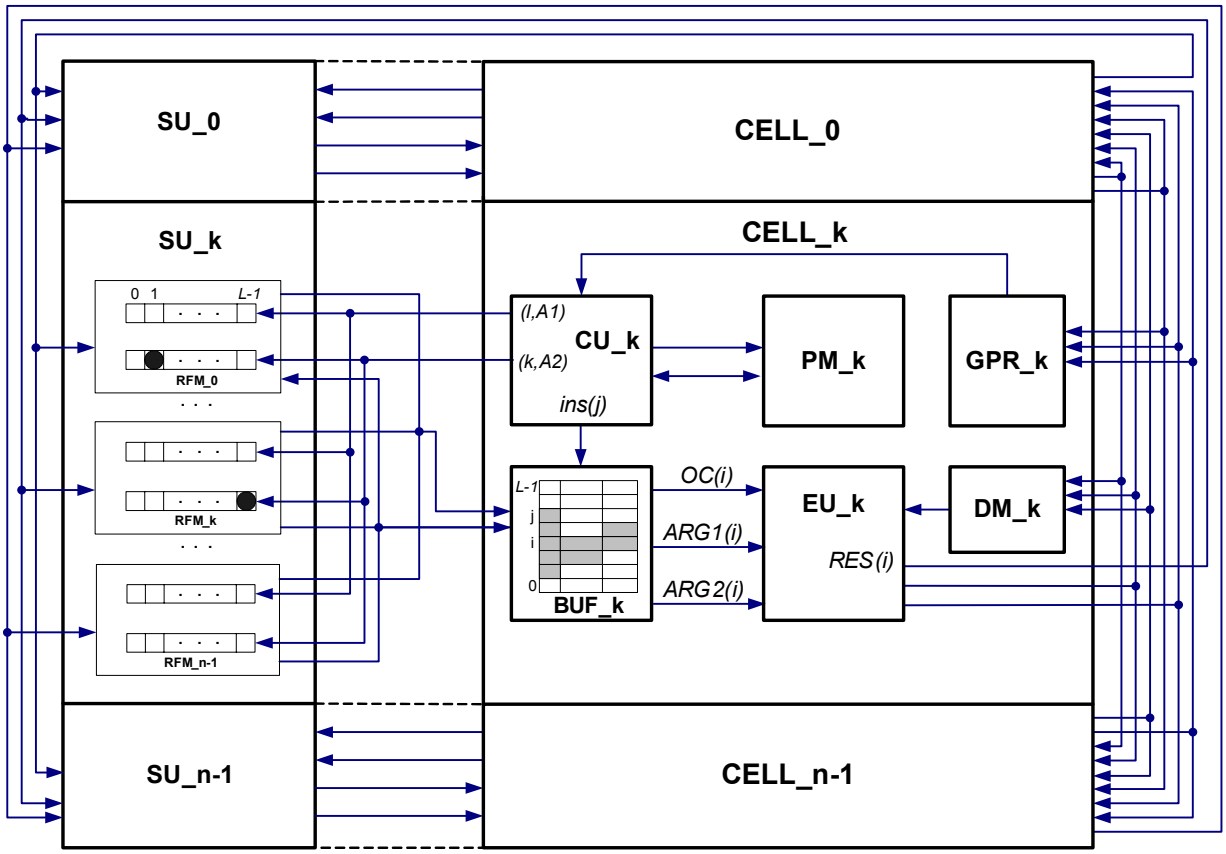

Рис.2. Структура реконфигурируемого мультиклеточного процессора.

3.2. Формирование номера запрашиваемого результата

При $n \neq 2^{**}k$ изменяется формула определения номера запрашиваемого результата, которая будет выглядеть следующим образом:

(2) $$N_r = (((T_g^{pre} + n_g) + N_{cg}) - P_{op}) mod (T_g^{max})$$

где: T_g^{pre} – предыдущее значение тега группы; n_g – количество клеток в группе; N_{cg} – логический номер клетки в группе, выбравшей команду запрашивающую данный результат– число в диапазоне от 0 до $n_g - 1$.

Следует отметить, что для приема результатов, устройству управления необходимо указать коммутационному узлу не только номера результатов, но и физические номера клеткок от которых они приходят (значения $A1$ и $A2$, соответственно, для первого и второго операндов). Переход от логического к физическому номеру обеспечивается таблицей соответствия, которая создается при формировании группы клеток. При этом логические номера присваиваются клеткам в порядке возрастания физических номеров

3.3. Организация выборки команд

Существующая схема выборки команд в мультиклеточном процессоре основана на одновременной выборке n клетками n команд. При этом каждая клетка на j-том шаге выбирает $N_c + j^* n$ команду параграфа. Соответственно, в РМ клетки содержится только последовательность выбираемых ею команд.

Для обеспечения динамической реконфигурации в РМ клетки необходимо хранить все программное обеспечение. Эта необходимость диктуется тем, что заранее не известно в составе какой группы и с каким логическим номером будет работать та или иная клетка. Например, если группа будет состоять из двух клеток, то эти клетки будут выбирать последовательно, каждую вторую команду параграфа. Клетка с номером N_{cg}=0 будет выбирать четные команды, а с номером N_{cg}=1– нечетные.В случае группы из трех клеток, клетки будут выбирать каждую третью команду. При различной длине командных слов подобная организация выборки практически не позволяет обеспечить требуемый темп выборки, а именно, n_g команд за такт.Этот темп может быть обеспечен при единой длине командных слов, либо при одновременной выборке n команд. В последнем случае РМ каждой клетки формируется как n-блочная память, в которой команды параграфа размещаются последовательно группами по n команд. Соответственно, при каждом обращении к РМ выбирается n команд, которые поступают на схему мультиплексирования. Из полученных n команд схема мультиплексирования выбирает одну и передает ее устройству управления (см. рис.3).

Номер выбираемой командыопределяется следующим образом:

(3) $$N_{pm} = (N_{cg} + j^* n_g) mod\ n$$

При этом очередная выборка группы команд выполняется тогда когда среди выбранных n команд нет больше исполняемых команд.

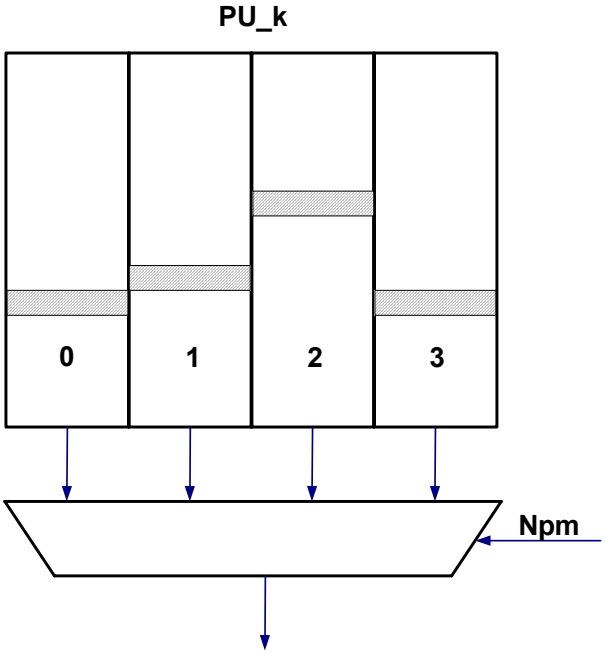

Рис.3. Схема выборки команд

3.4. ***Control****-группа*

Разделим все клетки мультиклеточного процессора на несколько непересекающихся подмножеств. Такое подмножество будем называть ***control***-группой.

Синхронизация выборки в ***control***-группе ограничена только составом ***control***-группы. Аппаратно она поддерживается регистрами связности по управлению(CCR).

Регистр связности по управлению входит в состав каждой клетки и имеет размер n бит. Если i-тая и j-тая клетка входят в одну ***control***-группу, то содержимое $CCR_j(i)=CCR_i(j)=$'1' для обоих клеток. Если они в разных control-группах, то $CCR_j(i)=CCR_i(j)=$'0'.

На рис.4 показаны CCR для мультиклеточного процессора состоящего из 4 клеток, разделенных на три ***control***-группы.

	3	2	1	0
Cell_0	**0**	**0**	**0**	**1**
Cell_1	**1**	**0**	**1**	**0**
Cell_2	**0**	**1**	**0**	**0**
Cell_3	**1**	**0**	**1**	**0**

Рис. 4. Пример содержимого CCR (клетки разбиты на 3 ***control***-группы:{0}{1,3}{2})

CCR позволяет каждой клетке выбрать из всей совокупности сигналов готовности к выборке только те, которые относятся к ее ***control***-группе.

Формируется содержимое *CCR* программно при реконфигурации. При этом архитектура мультиклеточного процессора не налагает каких-либо ограничений на количество ***control***-групп и количество клеток в ***control***-группе. Все клетки могут быть объединены в одну ***control***-группу. Может быть сформировано *n* ***control***-групп, в каждой будет по одной клетке. Конкретное распределение клеток по ***control***-группам определяется задачей, принципами организации вычислительного процесса и может неоднократно меняться в процессе вычислений. Клетка с нулевым *CCR*, т.е. не входящая ни в одну **control**-группу в вычислениях не участвует. Включение ее в вычислительный процесс возможно только при рестарте.

3.5. Регистр информационной связности

Регистр информационной связности (*ICR*) входит в состав каждой клетки и обеспечивает управляемое тиражирование данных, поступающих в DM и GPR. Имеет размер n бит. Если у некоторой клетки *ICR*(*i*)=’1’, то клетке с физическим номером *i* разрешена запись в память этой клетки. Если ’0’, то запись запрещена. Формируется содержимое *ICR* программно, при реконфигурации. При решении информационно не связанных задач, *ICR,* как правило, поразрядно совпадает с *CCR*. В остальных случаях этот регистр, как правило, разрешает запись от всех клеток, за исключением неработоспособных.

4. Организация процесса реконфигурации

Реконфигурация мультиклеточного процессора или его группы клеток может инициироваться программно, например, при плановом перераспределении ресурсов или по результатам сравнения двух просчетов. Она также может инициироваться аппаратно, если при работе в многоканальном режиме не совпадают данные записываемые в память. В последних двух случаях по результатам сравнения формируется, соответственно, программный или аппаратный сигнал на проведение самопроверки. Каждая клетка автономно проводит самопроверку, после которой проводится проверка коммутационной среды. Результаты проверки доступны всем клеткам. По результатам проверки клетками формируется работоспособное подмножество клеток, которое максимально по размеру и в которое включаются те клетки, которые считают друг друга работоспособными. В каждой клетке этого подмножества записывается новое состояние регистра связности по управленю на основании которого формируются таблицы соответствия логических и физических номеров и регистра информационной связности, в котором блокируются записи в память от неработоспособных клеток. Сформированное подмножество работает в этом составе до получения следующего сигнала на проведение самопроверки, после чего процесс повторяется. Если подмножества равны, то выбирается первое (с минимальным физическим номером клетки). Это относится и случаю, когда подмножества состоят из одной клетки.

.

5. Заключение

Современные подходы к созданию отказоустойчивых встроенных систем опираются на следующие методы: технологические (стойкая ЭБ), конструктивные, схемотехнические (резервирование на схемном уровне) и структурные (резервирование на приборном уровне). Возможности мультиклеточной архитектуры позволяют говорить о появлении еще одного – нового метода – системного, обеспечивающего не только традиционную отказоустойчивость, но и реализацию живучести на микропроцессорном уровне

СПИСОК ЛИТЕРАТУРЫ

1. *Стрельцов Н.В.* Архитектура и реализация мультиклеточных процессоров // Труды V Международной научной конференции «Параллельные вычисления и задачи управления» - Москва, 26-28 октября 2010. С. 1087-1104.
2. *Виноградов А.А.* Проблемные вопросы создания отечественных доверительных программно-аппаратных платформ АСУ критических инфраструктур // Сборник докладов конференции «Разработка отказоустойчивых микропроцессорных систем управления» - Москва, 31 мая – 01 июня 2012. С. 8.