ИНСТИТУТ ПРИКЛАДНОЙ МАТЕМАТИКИ
имени М.В. Келдыша
РОССИЙСКОЙ АКАДЕМИИ НАУК

В.М. Михелев

ВИКТОР
Архитектура многопроцессорного комплекса

МОСКВА
2002

УДК 681.31

Резюме

Описывается вариант тредовой архитектуры для многопроцессорных вычислительных комплексов. Рассматриваются проблемы связи процессоров с расщепленной памятью и автоматизации синхронизации при параллельных вычислениях.

Abstract.

A variant of computer multithreading architecture is described. It has a good look at problem of connection processors and a shared memory as well as synchronization of parallel execution.

Введение

В описанной в предлагаемой работе архитектуре многопроцессорного комплекса развиваются идеи и уточняются понятия, изложенные ранее автором в работах [1,2]. Идеологически близкими можно считать работы по Dynamic Multithreading Processor [3], Trace Processor [4,5] и Tera computer system [6]. По уже установившейся традиции [1,2] этот вариант архитектуры получил новое название, точнее отражающее текущее состояние работы – **ВИКТОР** – **ВИ**ртуальные **К**омпьютеры с **Т**редовой **ОР**ганизацией вычислений. Рассмотрим некоторые основные понятия и введем соответствующую трминологию.

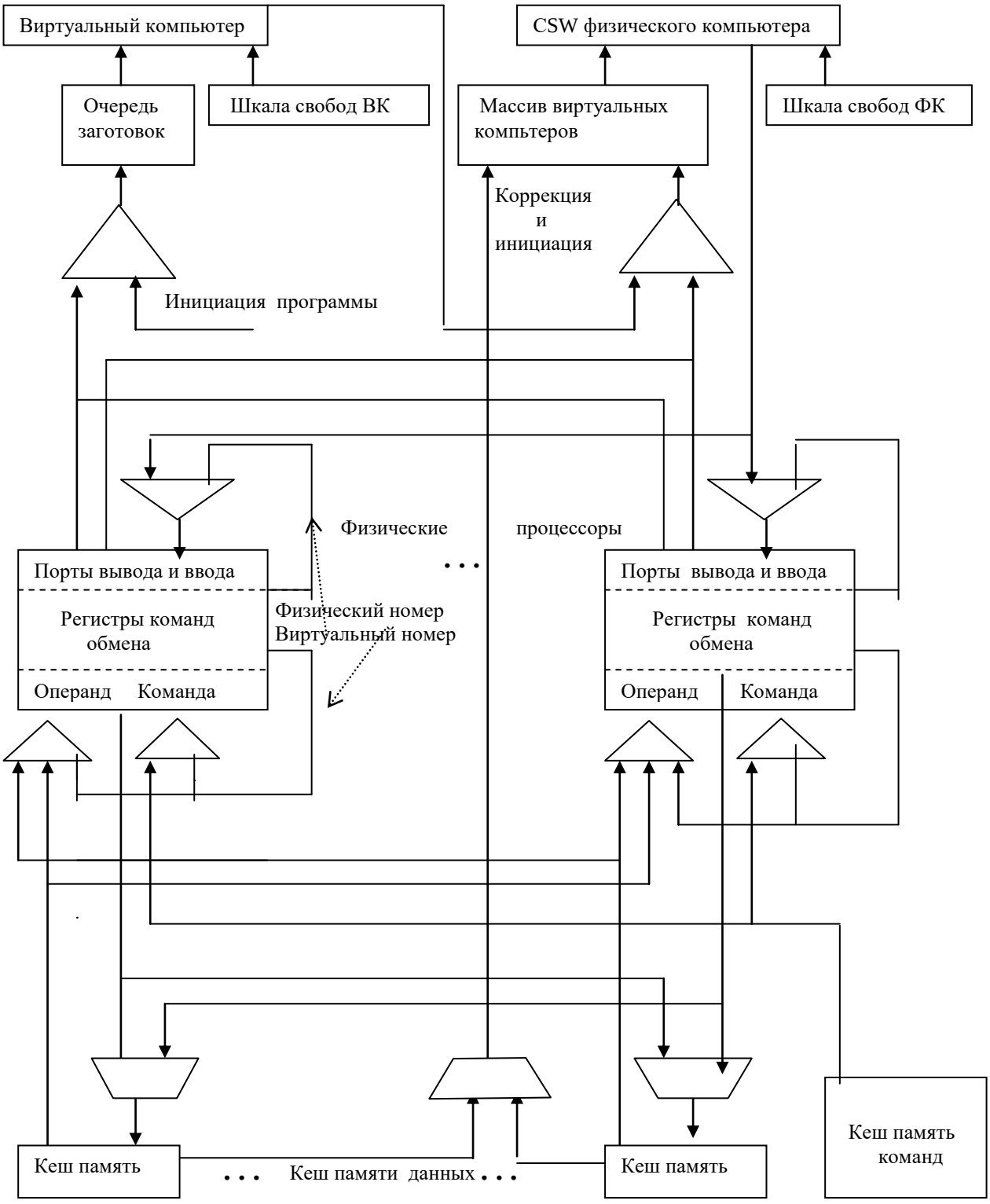
Текст программы на языке компьютеров, входящих в состав комплекса, содержит специальные команды, которые инициируют выполнение *трасс* – последовательностей команд, заканчивающихся командами останова. Каждая трасса выполняется на своем *виртуальном компьютере* и в свою очередь может инициировать новые трассы. Считается, что виртуальный компьютер может находиться как в активном состоянии так и в состоянии ожидания. В активное состояние он может перейти лишь при наличии свободного *физического компьютера*. Физический компьютер выполняет команды *треда* – последовательности команд трассы от конца предыдущего треда или начала трассы до команды прерванной, например, из-за отсутствия требуемого *тега* или адреса в кеш памяти при чтении или записи. Прерывание освобождает физический компьтер и переводит в ждущее состояние соответствующий ему виртуальный, передавая при этом информацию, необходимую для выполнения следующего треда. Обратный переход в активное состояние происходит после выполнения условий прерывания. Поскольку физических компьютеров меньше чем виртуальных, то они практически всегда загружены.

В многопроцессорных комплексах существуют несколько проблем, способ решения которых существенно влияет на их эксплуатационные характеристики. Во-первых, специальные команды, вводимые для организации параллельных вычислений не должны существенно осложнять создание трансляторов с языков высокого уровня. Архитектура **ВИКТОР**, по-видимому, удовлетворяет этому требованию. Во-вторых, связь большого количества процессоров с общей кеш памятью не должна серьезно увеличивать время доступа. В предлагаемой архитектуре эта проблема решается за счет расщепления памяти и использования так называемой “выборки с привилегиями”. И, в-третьих, должно быть обеспечен быстрый обмен информацией между физическими и виртуальными компьютерами. Здесь это достигается за счет использования ассоциативной памяти с возможностью прямого доступа.

Работа выполнена при поддержке Российского Фонда Фундаментальных Исследований.

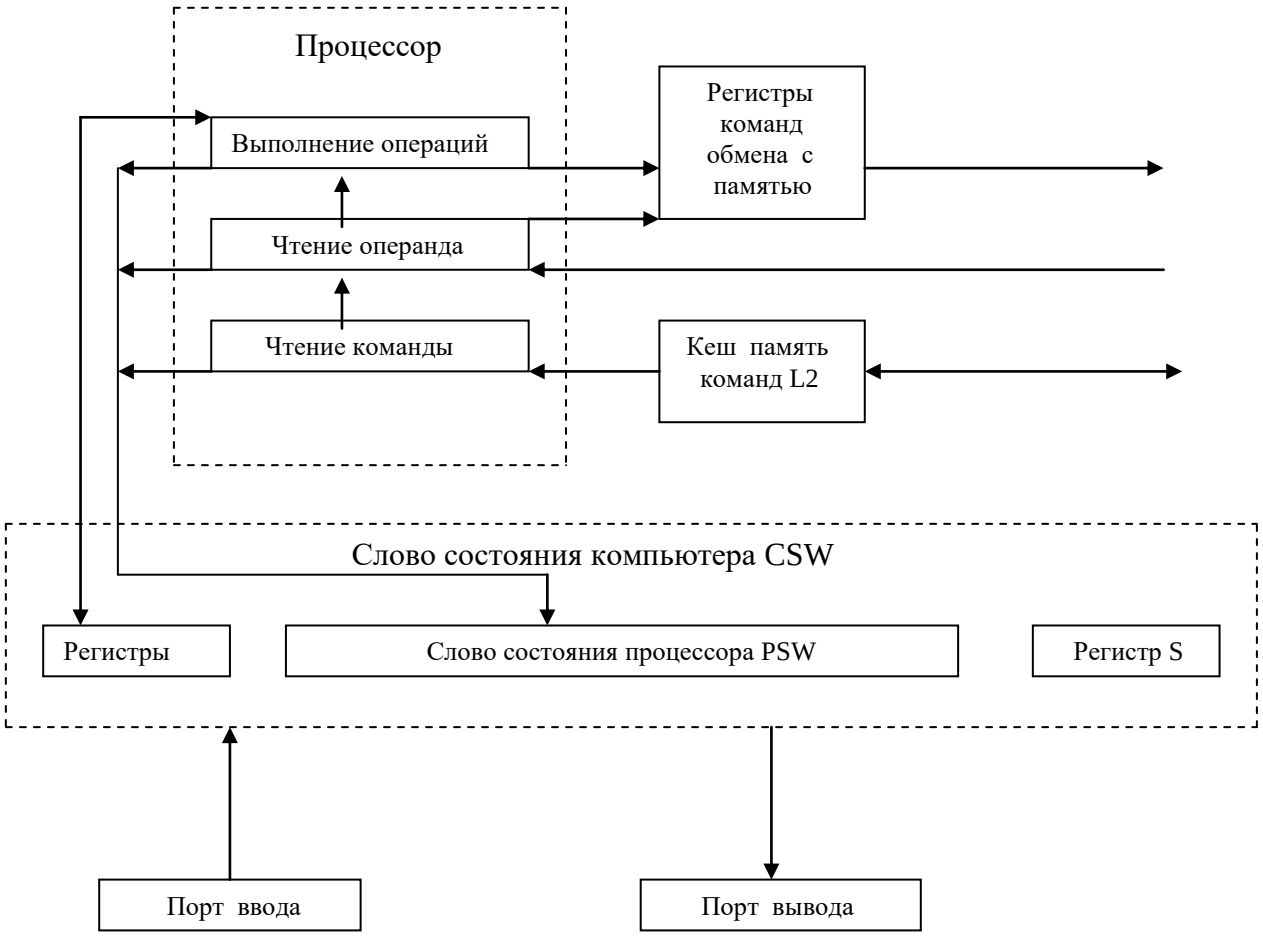
Элементы архитектуры

Совокупность физических компьтеров, массив виртуальных компьютеров, очередь заготовок, кеш памяти и система связи между ними образуют вычислительную среду. Соответствующая схема представлена на рисунке.



1. Компьютер.

Рассмотрим компоненты отдельного компьютера, схема которого изображена на рисунке.



1.1. Процессор – устройство выполнения операций. Используется трехуровневый конвеерный (pipeline) способ выполнения, состоящий из выборки команды, чтения операнда и собственно выполнения. Компоненты процессора в процессе работы занимают состояния, соответствующие уровню выполнения действия, например, ожидание операнда. В случае прерывания эти состояния запоминаются в слове состояния процессора – PSW. Процессор имеет физический и виртуальный номера. Последний приписывается процессору в момент, когда в компьютер из памяти виртуальных компьютеров загружается CSW – слово состояния компьютера.

Слово состояния процессора содержит

- текущий виртуальный номер процессора,
- его статус (свободен или занят),
- счетчик команд,
- код условия,
- состояния компонент процессора.

Ниже приведен формат команды процессора. Команда состоит из четырех полей и может содержать один или два операнда.

Операция	Индикатор	Регистр	Адрес/Регистр/Литерал

Регистр – поле, содержащее один из регистров S,A,B,C,D,E,F,G,H. Регистр А обычно используется в качестве базы программы, регистр В – базы данных, а регистр S – выполняет функцию первого операнда и регистра результата в однооперандных командах.

Индикатор – индикатор способа адресации. От способа адресации зависит, в частности, и количество заданных в команде операндов. Команды с индикаторами **a,c,d,w –** однооперандные, а с индикаторами **b**,**r,l –** двухоперандные.

a - признак автоматического базирования. Если это адрес в команде переходов или ветвлений, то в качестве базы берется регистр А, если адрес данных, то регистр В. Исполнительный адрес вычисляется следующим образом (База)+(Регистр)+Адрес. В командах с индикатором **a** первым операндом служит результат предыдущей выполненной операции, а вторым – операнд по адресу.

b – также считается признаком автоматического базирования, но в этом случае Регистр используется как первый операнд, а исполнительный адрес вычисляется как (База)+Адрес.

c – исполнительный адрес вычисляется как (Регистр)+Адрес

d – косвенная адресация в командах, использующих дескриптор. Адрес самого дескриптора задается аналогично **a.** Дескриптор – три элемента памяти. В первом элементе задается адрес трассы, которой передаются данные, во втором – база той части программы, которой трасса принадлежит, а в третьем – база соответствующих программе данных. При передаче данных по дескриптору устанавливаются базы вызываемой программы, то есть содержимое второго элемента, если он не нулевой устанавливается в регистр **A**, а содержимое третьего регистра заносится в регистр **B.**

w – при выполнении команды с таким индикатором адресуемый объект рассматривается как рабочий, используемый для реализации рандеву двух операндов. Исполнительный адрес вычисляется как в случае **a**.

l – в поле Адрес записывается константное или адресное выражение.

r – оба операнда заданы в регистрах: первый в поле Регистр, второй – в адресной части команды. Результат остается в первом операнде. Команды с таким индикатором не изменяют регистр **S** как регистр результата. Однако, он может использоваться в качестве операнда.

1.2. Регистры команд обмена с памятью используются для задания способа обмена, номера кеш памяти и адреса операнда. Ниже приведен формат команды.

Занятость	Команда	Виртуальный номер процессора	Адрес в кеш	Значение

По-существу, номер кеш памяти задается выбором соответствующего регистра. Бит занятости определяет занят регистр или свободен, то есть можно его использовать для обмена или следует подождать. Команда определяет способ обмена – чтение, запись и так далее. Виртуальный номер процессора указывает какому процессору следует возвратить результат.

1.3. Кеш память команд второго уровня используется для накопления команд трассы, исполняемой на данном процессоре. В случае необходимости команды подкачиваются из основной кеш памяти команд.

1.4. Слово состояния компьтера – CSW - состоит из слова состояния процессора – PSW, локальных регистров и регистра результата. Регистры именуются

S – регистр результата,

A,B,C.D.E,F,G,H – регистры баз и промежуточных результатов.

1.5. Порт ввода – регистр, через который передается компьютеру его CSW из массива виртуальных компьютеров.

1.6. Порт вывода – регистр, через который в случае прерывания CSW передается виртуальному процессору с номером, заданным в PSW. В случае команд разветвления, организующих новые трассы, заготовка для нового виртуального процессора через порт вывода передается в очередь заготовок.

2. Элементы памяти.

Объекты в памяти адресуются с точность до слова. Каждое слово сопровождается тегом, указывающим на наличии в памяти значения и допустимости его изменения.

Значение тега:

00 – объект не содержит значения (Cl),
01 – разрешена запись (Wr),
10 – разрешено чтение (Rd),
11 – значение в процессе изменения (Up).

При обращении к кеш памяти происходит проверка тега на соответствие требованию команды обращения к памяти. Если условие не выполняется, то команда возвращается выдавшему ее процессору, который инициирует прерывание, то есть заканчивает выполнение треда. CSW передается виртуальному компьютеру, и физический компьютер освобождается.

По-существу, тег Up служит тегом блокировки - при изменение объекта командами одной трассы доступ к объекту из другой трассы должен быть

заблокирован. Примерами могут служить изменение части слова или накопление суммы операндов, взятых из разных трасс.

3. Очередь заготовок.

Очередь заготовок – это совокупнось CSW, которым уже присвоены адреса трасс, но не заданы виртуальные компьтеры, на которых они должны вычисляться. При появлении свободного виртуального компьтера первому элементу очереди присваивается его номер и он переписывается в массив виртуальных компьтеров. Занесение в очередь происходит либо в начале выполнения программы, когда в очередь ставится CSW с адресом первой трассы, либо командами, организующими новые трассы – fork, cfork и gnr.

4. Массив виртуальных компьютеров.

Это память, сдержащая CSW компьютеров, принимающих участие в вычислении. Каждый виртуальный компьтер занят вычислением переданной ему трассы программы и может находиться в активном состоянии или в состоянии ожидания. В первом случае его CSW было передано одному из физических компьютеров, который занят вычислениями очередного треда. Во втором - виртуальный компьтер ожидает либо освобождения физического процессора, либо инициации (коррекции) при изменении состояния памяти, связанной с подкачкой, записью или изменением тега. Запись в массив виртуальных компьютеров происходит либо из очереди заготовок, либо из портов вывода компьютеров. В первом случае это означает, что к вычислениям подключается новая трасса, а во втором, что при вычислении возникло прерывание и для начала вычисления очередного треда требуется коррекция и инициация соответствующего CSW. Освобождение виртуального процессора происходит в результате выполнения команды stop – команды, завершающей выполнение трассы.

5. Кеш память данных.

Это расщепленная кеш память, то есть набор кеш памятей с непересекающимися адресами. В качестве способа расщепления может рассматриваться, например, эквивалентность по модулю. Режим работы памяти определяется заданной командой обмена, установленной на регистре команд обмена процессора. Поскольку кеш памяти в один и тот же такт могут быть переданы задания от разных компьютеров, то может возникнуть конфликтная ситуация. Для разрешения конфликта используются устройства, в которых реализуется так называемый “выбор с привилегиями”. На схеме вычислительной системы соответствующие усройства изображены в виде трапеций. Каждый элемент кеш памяти состоит из пяти полей.

Поясним значения полей.

Блокировка выталкивания запрещает выталкивание значения в основную память до того как предыдущее вытолкнутое значение не будет записано в основную память. Это существенно поскольку передача команд основной памяти происходит через кольцевой буфер.

Объект не использовался – поле, необходимое для оптимизации процесса обмена, запрещающее выталкивание подкаченного, но не использованного значения.

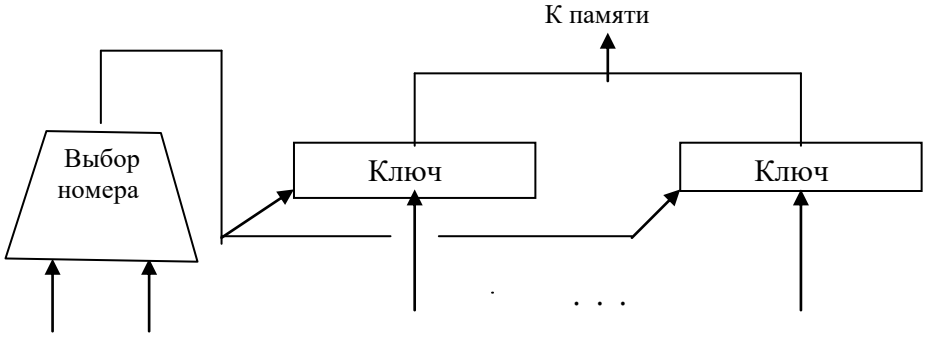
Тег - тег текущего значения - rd, wr, up.

Номер страницы – номер страницы в общей памяти, в которой должно храниться текущее значение ячейки кеш памяти, когда оно выталкивается. Размер страницы – общий размер кеш памяти.

Значение – текущее значение.

6. Выбор с привилегиями

Это устройство используется для разрешения конфликта в случае выбора процессора при обращении к памяти или кеш памяти при выборе команды коррекции в массиве виртуальных компиляторов.

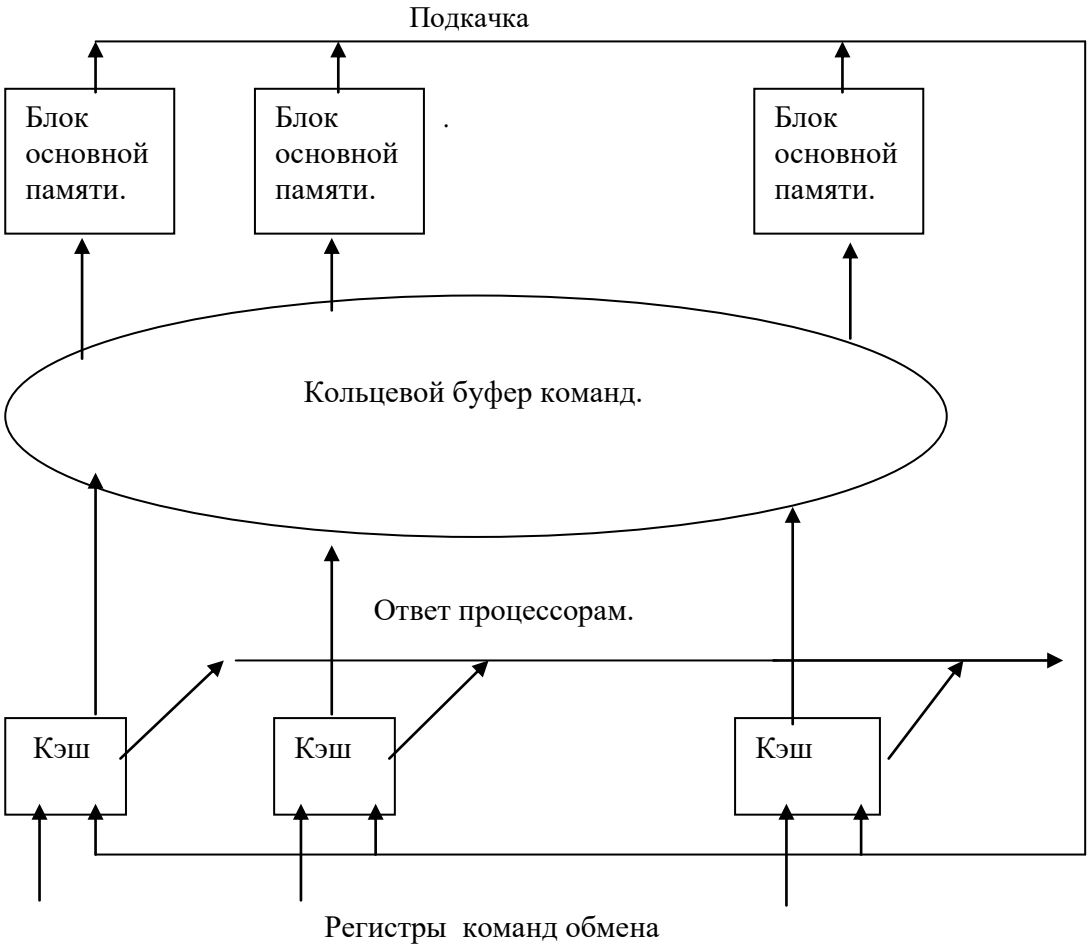


.

Рассмотрим, например, обращение к памяти. В этом случае устройство работает следующим образом. В комплесе зафиксировано, что привилегии уменьшаются по мере увеличения физического номера. В силу этого устройство выбирает регистр команд обмена процессора с наивысщей привилегией среди регистров процессоров готовых к обмену.

7. Память данных.

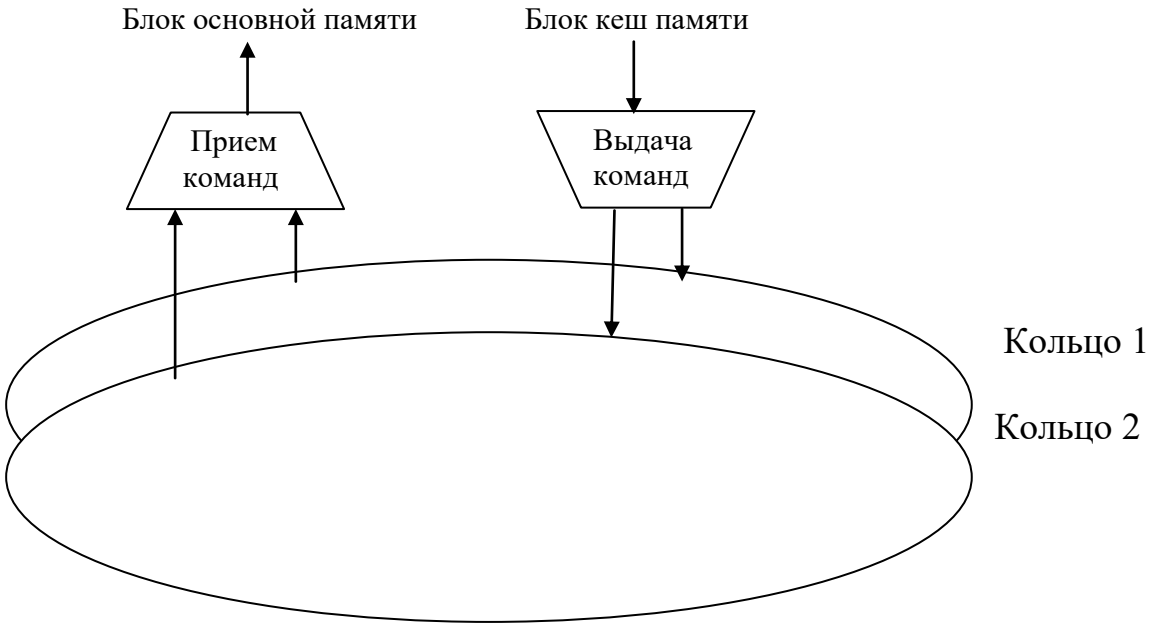
Память данных состоит из нескольких блоков с непересекающимися адресами. Рассмотрим один из возможных способов организации такой памяти – памяти с кольцевым буфером. Ниже представлена соответствующая схема.



Когда при обращении к кэш памяти происходит “промах”, то есть объект в кэш отсутствует, то команда обращения к памяти передается в кольцевой буфер. В нем команда циркулирует до тех пор, пока необходимый блок памяти не завершит предыдущее задание и не начнет выполнять данную команду.

Команда обращения к кэш может быть как выполненной, так и не выполненной. Последнее бывает в двух случаях – несовпадение тегов и “промах”. И в том и в другом случае происходит прерывание выполнения треда командой ответа с одновременной записью CSW в память виртуальных процессоров.

Очевидно, что размер кольца увеличивает время передачи команды из блока кеш памяти к блоку основной памяти, увеличивая тем самым время выборки. Нетрудно видеть, что можно уменьшить в несколько раз длину кольца при сохранении количества блоков памяти. Для этого нужно лишь заменить одно кольцо несколькими, усложнив схемы выбора и приема команды из кольца. Ниже приведен пример двойного кольца.



8. Команды процессора

Ниже приведен список команд, используемых для решения тестовых примеров на модели. В дальнейшем список этот будет расширен. Все команды делятся на семь групп.

1. Выполняющие действия над двумя операндами. Значение результата остается в первом операнде.

add, sub, mult, div, mod – арифметические команды с целыми аргументами.

or, and, not – логические команды.

eq, ge, ne, le, lt, gt – команды, устанавливающие отношения между операндами. Команды вырабатывают код условия – true или false.

2. Распараллеливающие вычисления. Команды этой группы позволяют распараллеливать вычисления путем создания новой трассы. Адрес трассы, значения регистров и код условия – заготовка – сначала заносится в очередь заготовок. При наличии свободного виртуального компьтера очередная заготовка присваивается этому компьютеру.

fork - разветвление Команда используется для создания трассы по указанному в команде адресу.

cfork - разветвление по условию.

gnr – генератор значений. Команда используется для параллельного выполнения команд тела цикла путем создания трасс с параметрами, изменяющимися в интервале – <значение первого операнда>... 0. Команда выполняется следующим образом. Если значение больше или равно нулю, то соэдается новая трасса, адрес которой в программе на единицу больше адреса команды gnr и значение уменьшается на единицу. Когда значение становится отрицательным, то выполняется команда по адресу, заданному во втором операнде команды gnr.

3. Передающие управление. Команды выполняются внутри одной трассы.

br - безусловная передача управления команде по адресу.

cond – передача управления по условию. Если условие true, то следующей выполняется команда по адресу, если false, то текстуально следующая.

4. Команды, организующие обращение к процедурам.

call - по адресу в команде записывается дескриптор возврата. Первый операнд – адрес точки возврата.

par – запись шаблона дескриптора параметров. Первый операнд – адрес процедуры записывается во второе слово дескриптора. В третье слово дескриптора записывается регистр D.

5. Команды, присваивающие значение. Напомним, что в одноперандных командах первый операнд – регистр S.

rdv – первому операнду присваивается значение второго.

rdup – аналогично rdv. Второму же операнду, если он в памяти, присваивается тег up, защищающий его от несвоевременного чтения командой из другой трассы.

wrv – второму операнду присваивается значение первого. Если второй операнд - объект в памяти, то ему присваивается тег rd.

wrup – снимается блокировка, установленная rdup. Далее аналогично wrv.

crd – чтение кода условия.

cwr – запись кода условия.

push - значение регистров C,D,E,F,G,H,S сохраняются в области, адрес которой задается во втором операнде.

pop – восстанавливает значения регистров из области, адрес которой задается во втором операнде.

6. Команда останова.

stop - прекращение выполнения треда.

7. Пустая команда.

emp – не выполняет никаких действий.

Примеры и приемы программирования.

Все приведенные ниже примеры записываются на языке ассемблера. Поля в предложениях ассемблера называются:

Опер – операция,
И – индикатор способа адресации,
Р – идентификатор регистра,
Адр - поле адреса.

1. Перемножение матриц.

Рассмотрим следующий пример.

```
MM,NN,RR: array[1..20] of integer;
          Sum: integer;

For I in 1..20
  Loop
    For j in 1..20
      Loop  Sum:=0;
        For k in 1..20
              Loop Sum+=MM[i,k]*NN[k,j]
          End loop;
        RR[i,j]:=Sum
    End loop
  End loop
```

В приведенном ниже примере вторая команда gnr порождает трассы, каждая из которых вычисляет один элемент матрицы результата.

```
        Опер  И   Р   Адр      Комментарий

        rdv   l   S   19       19 => S
        gnr   a       end      Конец перемножения матриц.   →

        mult  l   S   20
        wtv   r   S   D        i*20 => D
        rdv   l   S   19
        gnr   a       Line                                  →

        wrv   r   S   E        j => E
        rdv   l   F   0        k:=0 => F
        rdv   l   C   0        Сумма C=0
Cycle   eq    r   F   20
```

```
        cond  a        Summa
        rdv   r    H   F


        add   r    H   D          i*20+k => H
        rdv   a    H   MM
        wrv   r    S   G          MM[j,k] => G
        rdv   r    H   20
        mult  r    H   F
        add   r    H   E          k*20+j => H
        rdv   a    H   NN         NN[k,j] => S
        mult  r    S   G          MM[i,k]*NN[k,j] => S
        add   r    C   S
        add   l    F   1          k:=k+1
        br    a        Cycle

Summa   add   r    D   E          i*20+j => E
        Rdv   r    S   C
        wrv   a    D   RR         Запись значения в RR[I,j]
        stop

Line    stop

End     stop
```

В приведенном примере строки из пробелов отделяют части программы, соответствующие трассам. Стрелки → указывают на продолжение трассы. В комментариях => означает “содержится в регистре”.

2. Упорядочение массива целых чисел.

Рассмотрим пример, демонстрирующий защиту объекта, который изменяется одниой трассой, от несвоевременного доступа со стороны другой трассы. С этой целью рассмотрим задачу упорядочения массива целых чисел методом пузырька. Допустим, что задача решается независимо, но одновременно сразу несколькими процессорами и используется стандартный алгоритм сортировки. Понятно, что задача будет решаться быстрее, поскольку процессоры будут помогать друг другу. Для определенности выберем 10 виртуальных процессоров и массив MM из 1000 слов.

```
        rdv   l   D    1          D = 0 – признак конца сортировки
        rdv   l   S    10-1
        gnr   a        End        Генерация 10 трасс →
```

```
Cycle eq     l   D    0
       cond  a        Stp       Окончание сортировки
       rdv   l   D    0
       rdv   l   E    0         E – номер элемента в массиве
       rdup  a   E    MM        Чтение с блокировкой обращения
       wrv   r   S    F         B F – левый элемент пары

Check add    l   E    1
      rdup   a   E    MM
      wrv    r   S    G         B G – правый элемент пары
      le     r   F    G
      cond   a        Right
      wrv    r   G    H         Обмен значениями
      rdv    r   F    G
      wrv    r   H    F
      rdv    l   D    1         Сортировка не закончена
Rignt rdv    r   S    F
      sub    l   E    1
      wrup   a   E    MM
      add    l   E    1
      rdv    r   F    G
      lt     l   E    1000-1
      cond   a        Check
      rdv    r   S    G
      wrup   a   E    0         Новое значение левого  элемента пары
      br     a        Cycle
Stp   stop

End   stop
```

В этом примере трасса, содержащая команду gnr, передает данные десяти другим трассам, каждая из которых на своем собственном компьютере выполняет сортировку массива.

3. Вызов процедуры.

Вызов процедуры при распараллеливании вычислений отличается тем, что любая передача параметра приводит к началу выполнения новой трассы в вызываемой программе. Второе отличие – процедуры должны быть реентерабельными. Это означает наличие собственной локальной памяти для каждого вызова процедуры. Если это условие не соблюдено, то потребуется дополнительная синхронизация для установления последовательности вызова процедур.

Очевидно также, что в случае процедуры-функции команда, принимающая значение, должна иметь индикатор **w**, чтобы обеспечить синхронизацию операндов. Кроме того, необходимо обеспечить восстановление тех значений регистров, которые они имели до обращения к процедуре.

Вообще говоря, команды, реализующие вызов и возврат из процедуры, должны выполнять соглашение о связи вызывающей и вызываемой процедур.

Один из возможных вариантов такого соглашения реализован в приведенном ниже примере. В дальнейшем будем считать, что

- все процедуры адресуются с нуля, то есть значение базы (регистр **A**) процедуры устанавливается на ее начало,
- адреса меток задаются по отношению к началу процедуры, а адрес самой процедуры устанавливается загрузчиком по отношению к началу памяти,
- перед каждым вызовом процедуры ей выделяется локальная память данных, база которой заносится в регистре **D,**
- локальная память начинается с области сохранения регистров, за которой следуют дескриптор возврата и дескриптор параметров,
- параметр передается по значению командой ветвления.

Рассмотрим следующий простой пример.

```
Proc RlSign(x,y) returns integer   (* Вызываемая процедура *)
  If x*y>0
    then return 0
    else  return 1
  end if;

… p*RlSign(m,n) …                  (* вызов RlSign *)
```

Пусть аргумент m записан в локальной памяти вызывающей процедуры по адресу MM, аргумент n – по адресу NN, а p – по адресу PP. Как обычно, локальная память данных в вызывающей процедуре базируется регистром B. Будем считать, что размер слова в этой памяти равен четырем байтам.

Локальная память данных начинается областью сохранения регистров – C,D,E,F,G,H,S, состоящей из семи слов. За ней следуют дескриптор возврата и шаблон дескриптора передачи параметров. В шаблоне в момент передачи параметра первое слово заменяется адресом входа, по которому этот параметр передается.

Ниже приведены фрагменты текста программы на ассемблере.

```
RlSign     proc                    Тело процедуры
RlSign#1 nop                       Вход для m
RlSign#2 mult  w        WRlSign    Вход для n
                                   WRlSign– объект для синхронизации,
```

```
                                        который находится в локальной
        gt    l   S   0                 памяти данных процедуры RlSign.
        pop   c   B   0                 Восстановление регистров
        rdv   l   S   0
        cond  d   D   0                 D снова указывает на начало области
        rdv   l   S   1
        br    d   D   0
endp
```

Передача параметров и обращение к процедуре RlSign.

```
        push   c  D   0          Сохранение регистров
        rdv    l  S   Rt         Адрес возврата Rt  => S
        call   c  D   28         Запись дескриптора возврата
        rdv    l  S   RlSign     База процедур
        par    c  D   40         Запись шаблона для дескриптора параметра
        rdv    l  S   RlSign#1
        wrv    c  D   40
        rdv    a      MM
        fork   d  D   40           Передача первого параметра
        rdv    l   S  RlSign#2
        wrv    c   D  40
        rdv    a      NN
        fork   d  D   40           Передача второго параметра
        rdv    a      PP
Rt      mult   w      Wmlt        Wmlt – объект для синхронизации, который
                                     расположен  а локальной памяти  данных.
               …
```

Заключение

Основное внимание при разработке рассмотренной выше архитектуры многопроцессорного комплекса было уделено вопросам быстрого обмена с памятью, инициации прерванных в процессе обращения к памяти вычислений и вопросам синхронизации. Вопросы прерывания как системные, так и пользовательские, а также многозадачность не были предметом данной работы. Их разработка предполагается в дальнейшем.

Литература

1. В.М. Михелев
   ПРОК Архитектура многопроцессорного комплекса,
   препринт ИПМ им. Келдыша РАН, 2000, N 59.

2. В.М. Михелев
   ПЕДАНТ Архитектура многопроцессорного комплекса,
   препринт ИПМ им. Келдыша РАН, 2001, N 81.

3. H. Akkary, M. Driscoll
   A Dynamic Multithreading Processor.
   31st Annual ACM/IEEE International Simposium on Microarchitecture,
   Nov. 1998

4. E. Rotenberg, Q. Jacson, Y. Sazeides, and J.Smith,
   Trace Processor.
   30st Annual ACM/IEEE International Simposium on Microarchitecture,
   Dec. 1997

5. James E. Smith and Sriram Vajapeyam
   Trace Processor: Moving to Fourth-Generation Microarchitectures,
   IEEE Computer, Vol. 30, No.9, September 1997.

6. R. Alverson, D. Callaham, D. Cummings, B. Koblenz, A Portenfueld,
   B. Smith.
   The Tera computer system.
   Proceedings of the ACM International Conference on Supercomputing,
   June 1990.

Оглавление