ВОПРОСЫ ТЕОРИИ И ПОСТРОЕНИЯ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ
(Вычислительные системы)
1977 год Выпуск 70

УДК 681.31:681.323

ОДНОРОДНЫЕ ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ НА БАЗЕ МИКРОПРОЦЕССОРНЫХ БОЛЬШИХ ИНТЕГРАЛЬНЫХ СХЕМ

О.Л.Бандман, Э.В.Евреинов, В.В.Корнеев, В.Г.Хорошевский

§1. Введение

Поиск путей создания высокопроизводительных универсальных вычислительных средств [1,2] породил концепцию "идеальной вычислительной машины". В ее основе лежат два принципа: модульность структуры и максимально возможная одновременность выполнения операторов реализуемого алгоритма. Каждый модуль "идеальной машины" способен выполнить любой оператор алгоритма, а характер соединения модулей исключает потери, связанные с распределением операторов между ними, подготовкой данных для операторов и определением моментов начала их выполнения.

Примерами [2] реализованных модульных машин являются матричные, ассоциативные и магистральные вычислительные системы. Во всех трех типах систем размещение операторов и организация связей между ними производится из одного управляющего центра. Такая архитектура обусловливает высокую эффективность использования модулей только при реализации алгоритмов специальных классов. Кроме того, надежность и живучесть таких систем определяется управляющим устройством.

Модульные машины, в основу которых положена идея однородных вычислительных систем (ОВС) [1,3] свободны от этих недостатков. Главная особенность ОВС состоит в том, что модулем ее является универсальная вычислительная машина (элементарная машина - ЭМ), способная самостоятельно принимать решения относительно размещения операторов и транспортировки данных между ними. Средства

управления операторами содержатся в каждой элементарной машине системы. Этим обеспечивается максимально возможная загрузка модулей при относительно малых потерях времени на реализацию управляющих функций и создаются условия для самодиагностики и самовосстановления при отказах. Разработанные Институтом математики СО АН СССР совместно с промышленными организациями опытно-промышленные экземпляры вычислительных систем, такие как "Минск-222" [4], МИНИМАКС [5], СУММА [6], дали убедительные подтверждения возможности построения на основе ОВС высокопроизводительных и надежных вычислительных средств.

Изменения в элементной базе [7], связанные с появлением микропроцессорных серий больших интегральных схем (БИС), позволяют надеяться на экономически оправданную реализацию мощных вычислительных средств, построенных по идеологии ОВС.

Возможности, которые предоставляют большие интегральные схемы, заключаются в основном в их функциональной гибкости и дешевизне. Гибкость обусловлена микропрограммным управлением с программируемой и микропрограммируемой памятью микропрограмм, что дает возможность ограничить число типов необходимых БИС. Дешевизна БИС позволяет допускать необходимую аппаратурную избыточность, которая неизбежна при реализации управляющих средств в каждой ЭВМ.

Кроме указанных возможностей, микропроцессорная реализация накладывает и ряд ограничений. Основные из них обусловлены I) слабой вычислительной мощностью микропроцессоров и 2) ограниченным числом выводов с кристалла. Первое влияет на архитектуру системы, так как требует средств обеспечения интенсивных взаимодействий. Второе оказывает влияние на структуру, ограничивая количество системных связей.

Современное состояние теории и практики программно-аппаратного управления параллельными вычислениями позволяет поставить задачу создания вычислительной системы, удовлетворяющей следующим требованиям.

I. Система должна допускать разбиение на самостоятельно функционирующие подсистемы (вплоть до одной элементарной машины), сохраняющие основные свойства всей системы, а также допускать неограниченный рост числа элементарных машин.

2. Выход из строя некоторой совокупности элементарных машин не должен приводить к отказу системы в целом. Система должна быть самодиагностируемой и самовосстанавливающейся.

3. Структуры данных (разрядность обрабатываемых векторов, связь между данными и их идентификация) должны определяться пользователем и эффективно представляться в системе.

4. Операционная система должна предоставлять любой из известных режимов работы: пакетный, разделения времени, реального времени и т.д. По возможности операционная система должна реализоваться аппаратно. Особенно это касается механизмов принятия решений на основе данных из разных элементарных машин.

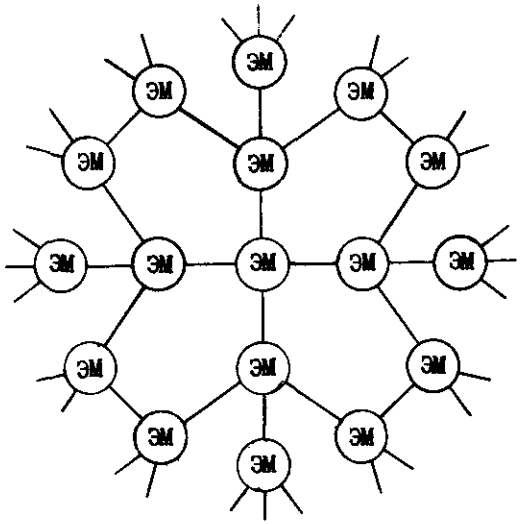
5. Увеличение числа элементарных машин в подсистеме должно приводить к появлению новых качеств: возможности интерпретации параллельных алгоритмов, организации табличных и групповых вычислений, а также пространственной обработки информации.

6. Внутренним языком элементарной машины должен быть язык высокого уровня, удобный для написания операционной системы.

Можно предположить, что проекты, реализованные на существующих серийных БИС, будут удовлетворять только первым четырем требованиям. Однако современное состояние технологии уже сейчас позволяет создать необходимые специальные БИС для реализации ОВС, соответствующей всем шести пунктам. При увеличении уровня интеграции БИС и их дальнейшем удешевлении перечисленные требования будут удовлетворяться все более полно.

В работе излагается один из возможных подходов к построению ОВС на основе микропроцессорных БИС. Этот подход предусматривает возможность реализации различных вариантов, получающихся путем перераспределения функций между программной и аппаратной частями системного управления.

Структурной особенностью предлагаемого подхода является выделение аппаратного ядра элементарной машины, допускающего реализацию в виде однокорпусной БИС – коммутационного модуля. Этот модуль образует шинную структуру элементарной машины, управляет ею и осуществляет соединения ее с шинными структурами соседних элементарных машин, организуя тем самым межмашинные связи (ММС). К образуемой коммутационным модулем (КМ) шинной структуре подключаются процессорные кристаллы (μС) элементарной машины (рис.I). Это могут быть как микропроцессоры с традиционной структурой (типа *Intel* – 8080 или *Intel* – 3000), так и специальные БИС, выполняющие функции аппаратной реализации операционной системы. Простейший вариант элементарной машины состоит из микропроцессора, кон-



мутационного модуля и памяти.В таком варианте все функции операционной системы реализуются программно. Не меняя архитектуры ОВС,введение в структуру дополнительных БИС, реализующих функции операционной системы, улучшает надежностно-временные характеристики. Таким образом, обеспечивается возможность построения целого спектра вычислительных средств с одной и той же архитектурой,но разными характеристиками.

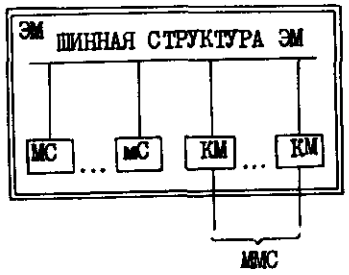


Рис. I

Архитектурной особенностью предлагаемого подхода является способ временной и пространственной организации вычислительных процедур. Методологическую основу способа составляет представление вычислений в виде асинхронных взаимодействующих процессов. Техническая реализация способа заключается во введении в каждую элементарную машину как средств управления процессами, так и средств, обеспечивающих взаимодействие процессов, протекающих в разных машинах.

§2. Архитектура однородной вычислительной системы

1. Архитектурные особенности микропроцессорной ОВС.

Однородная вычислительная система является "большим коллективом вычислителей", эффективная и согласованная работа которых может быть организована на основе следующих условий:

I) вычислительные процедуры задаются в виде набора операторов и отношений между ними;

2) готовность оператора к выполнению завершает выполнение всех операторов,динамически предшествующих ему;

3) вычисления должны быть однозначными, т.е. результат не должен зависеть от порядка реализации готовых к выполнению операторов;

4) после завершения вычислительной процедуры на входах оператора не должны оставаться неиспользованные значения [8,9].

Эффективная реализация такого способа вычислений требует соответствующей программно-аппаратной организации, при которой особое внимание уделяется механизмам определения готовности операторов к выполнению.

Наряду с вычислениями по программе пользователя элементарная машина должна выполнять функции узла коммутации, а также функционировать как самостоятельная единица системы самодиагностики и самовосстановления, обеспечивая живучесть ОВС [10,11].

Таким образом, операционная обстановка в ЭМ характеризуется многопрограммным режимом с заранее непредсказуемыми моментами взаимодействия программно-аппаратных средств. Это обусловливает необходимость введения в ЭМ единого механизма, обеспечивающего замещение программных фрагментов, традиционное для мультипрограммирования, и передачу значений от одних операторов другим, расположенным в той же самой ЭМ или в других элементарных машинах.Базой такого механизма являются понятия процесса и системы взаимодействующих процессов. Указанные понятия составляют основу современной вычислительной техники [12]. Они являются более равнозначным представлением (по сравнению с понятием ал-

горитм) того, что происходит при реализации вычислений на цифровых машинах.

Описание необходимого преобразования данных и функционирования вычислительных машин в терминах процессов имеет структурированный характер, обусловленный единообразием определения всех процессов ОВС. Заметим, что под структурированием здесь понимается программирование с использованием ряда полезных эвристических ограничений, увеличивающих наглядность описаний и отладки программ.

Программным и аппаратным средствам ОВС соответствуют процессы-программы и процессы-устройства.

2. О п е р а ц и о н н а я с и с т е м а ОВС.

Главной функцией операционной системы является у п р а в л е н и е п р о ц е с с а м и ОВС.

На множестве процессов ОВС определены следующие операции управления процессами: п о р о ж д е н и е, у н и ч т о ж е н и е, в з а и м о д е й с т в и е. Порождение процесса-потомка осуществляется процессом, называемым процессом-родителем, и состоит в присвоении порожденному процессу имени, а также в выделении части ресурсов процесса-родителя. Уничтожение процесса происходит путем исключения его имени из списка имен процессов системы и возврата его ресурсов процессу-родителю. Взаимодействие процессов включает взаимную синхронизацию и обмен информацией.

Операции управления процессами выделяются в я д р о о п е р а ц и о н н о й с и с т е м ы. Ядро реализуется набором программно-аппаратных средств, которыми обладает каждая элементарная машина. Сосредоточение функций управления процессами в ядре позволяет строить операционные системы некоторым стандартным образом, в виде наборов программ, осуществляющих связь с внешним миром и определяющих назначение ресурсов и момент начала работы операторов.

Ядро операционной системы ЭМ, наряду со средствами управления процессами содержит средства для организации связи между ядрами разных ЭМ. Отражением этого факта является то, что имена процессов, загруженных в ОВС, состоят из имени (номера) ЭМ, используемого для управления каналами межмашинного обмена, и собственно имени процесса, используемого для организации управления процессами.

Кроме ядра элементарная машина может иметь некоторые служебные программы, расширяющие возможности операционной системы и повышающие ее эффективность. Некоторые программы, не относящиеся к ядру, но являющиеся наиболее важными для функционирования системы, могут находиться в нескольких элементарных машинах. К ним относятся, например, программы, обеспечивающие реконфигурацию системы при отказах и сбоях.

ОПЕРАЦИОННАЯ СИСТЕМА ОВС	
Управление процессами (ядро ЭМ)	Взаимодействие процессов Межмашинные обмены
Управление ресурсами	Распределение ресурсов Реконфигурация и самовосстановление Самодиагностика
Управление данными	Файловая система Локальные диспетчеры Базовая операционная система

Рис. 2

При образовании в ОВС подсистемы происходит с л и я н и е о п е р а ц и о н н ы х с и с т е м э л е м е н т а р н ы х м а ш и н. При этом может быть образована операционная система любой требуемой конфигурации. Простейшая конфигурация состоит из ядра и базовой операционной системы, функции которой заключаются в управлении внешними устройствами и в организации файловой системы ОВС. Процессами базовой операционной системы являются локальные диспетчеры, осуществляющие поиск массивов по именам (без указания физических номеров внешних устройств).

Структура операционной системы ОВС представлена на рис. 2. Порядок блоков отражает степень необходимости их аппаратной реализации. Наиболее важным для аппаратной реализации представляется ядро, затем программы распределения ресурсов и связи с внешним миром.

3. О п е р а т о р ы у п р а в л е н и я п р о ц е с с а м и.

С каждым процессом связана таблица, содержащая информацию об имени процесса, его состоянии, ресурсах, а также ряд служебных признаков и указателей. Совокупность таблиц всех процессов, загруженных в ЭМ, образует список процессов.

Процессы имеют фиксированную структуру, включающую входной монитор данных, тело процесса и выходной монитор. Входной монитор определяет готовность структуры данных для обработки телом процесса. Например, в случае, если телом процесса является команда сложить два числа, то роль входного монитора сводится к определению момента получения обоих этих чисел. Выходной монитор, используя информацию о нормальном или ошибочном завершении тела процесса, формирует извещение о завершении работы.

Процесс может находиться в одном из следующих состояний: п а с с и в н о м, т е к у щ е м и ж д у щ е м. Пассивное состояние процесса-программы означает, что указанная программа отсутствует в памяти той ЭМ, на которой она должна реализоваться. Пассивному состоянию процесса-устройства соответствует устройство, не готовое к работе (например, отключенное). Процесс находится в текущем состоянии, если он исполняется. Ждущее состояние процесса обусловлено отсутствием условий, необходимых для перехода процесса в текущее состояние.

В з а и м о д е й с т в и я м е ж д у п р о ц е с с а м и и переходы процессов из одного состояния в другое происходят посредством управляющих взаимодействий: ПОРОДИТЬ {список подписков таблиц процессов; указатель выхода при нормальном завершении процессов; указатель выхода по ошибке}, УНИЧТОЖИТЬ{ список имен подписков процессов; указатель выхода при нормальном завершении процесса; указатель выхода по ошибке}, ЖДАТЬ{ список имен подписков процессов; условие блокировки; указатель выхода по ошибке}, ИЗВЕСТИТЬ{ список имен подписков процессов; условие разблокировки; указатель выхода по ошибке}.

Событие, состоящее в генерации ПОРОДИТЬ $\{\alpha, \beta, \gamma\}$, вызывает передачу управляющих взаимодействий ПОРОДИТЬ $\{\alpha_i, \beta, \gamma\}$ в элементарные машины, идентификаторы которых совпадают с указанными в подписках α_i, $\alpha = \{\alpha_1, \ldots, \alpha_k\}$. Подписок α_i содержит таблицы процессов, которые должны быть расположены в одной элементарной машине. Попытка породить в ЭМ подписок процессов с именем, совпадающим с уже имеющимся, вызывает передачу управления по указателю выхода по ошибке.

При отсутствии подписка процессов с именем, совпадающим с вновь порождаемым, таблицы процессов, составляющие подписок, заносятся в список процессов. Путем одноразового использования ПОРОДИТЬ

$\{\alpha_i, \beta, \gamma\}$ в ЭМ порождается столько процессов, сколько указано в подписке α_i. Фактически этим вводится механизм неявного порождения процессов.

Появление управляющего воздействия УНИЧТОЖИТЬ $\{\alpha, \beta, \gamma\}$ (по аналогии с ПОРОДИТЬ $\{\alpha, \beta, \gamma\}$) вызывает передачу в соответствующие ЭМ управляющего воздействия УНИЧТОЖИТЬ $\{\alpha_i, \beta, \gamma\}$. При наличии в ЭМ подписка процессов с именем, заданным в α_i, все упоминания о процессах подписка α_i уничтожаются с освобождением занимаемых этими процессами ресурсов. Если при этом оказывается, что подписок пуст, то уничтожается упоминание о нем.

В случае, если в ЭМ отсутствует хотя бы один из процессов, упомянутых в подписке α_i, управление передается по указателю выхода по ошибке.

Управляющее воздействие ЖДАТЬ $\{\alpha, \beta, \gamma\}$ передается в элементарные машины, содержащие процессы с именами из α_i. В каждом из этих процессов проверяется условие β. Если оно не выполнено, то рассматриваемый процесс в качестве условия блокировки получает β и переводится в ждущий. В таблицу процесса заносится информация, позволяющая ему продолжиться после выполнения условия β. В случае, если процесс с именем, указанным в подписке α_i, отсутствует, управление передается на γ, т.е. на выход по ошибке.

Генерация управляющего воздействия ИЗВЕСТИТЬ $\{\alpha, \beta, \gamma\}$ вызывает передачу управляющих воздействий ИЗВЕСТИТЬ $\{\alpha_i, \beta, \gamma\}$ в соответствующие ЭМ. Случай отсутствия в ЭМ процесса с именем, указанным в α_i, рассматривается как сбой, и управление передается по указателю выхода по ошибке. Иначе, условие разблокировки β поступает в процесс.

В качестве у с л о в и й б л о к и р о в к и п р о ц е с с а в ждущем состоянии могут быть следующие: занят процессор, занято внешнее устройство, занят буфер, устройство не готово к работе, не выполнено логическое условие и т.д. В силу введения процессов-устройств и процессов-программ, все условия блокировки есть не что иное, как логические функции над некоторым набором переменных, принадлежащих разным процессам.

Переходы процессов из состояния в состояние происходят при появлении управляющих воздействий. При этом состояние текущих процессов приводится в соответствие с появившимся управляющим воздействием.

После проверки условий ожидания и перевода процессов, в которых эти условия выполнены, в очередь готовых, назначаются текущие процессы.

Число текущих процессов в ЭМ определяется количеством процессоров, входящих в нее.

4. С и н х р о н и з а ц и я п а р а л л е л ь н ы х п р о ц е с с о в.

Одной из основных функций операционной системы, связанных с управлением процессами, является с и н х р о н и з а ц и я и х в з а и м о д е й с т в и й. Необходимость в синхронизации возникает, во-первых, в связи со стремлением нескольких процессов получить доступ к одним и тем же программно-аппаратным ресурсам и, во-вторых, в связи с обеспечением межпроцессорных коммуникаций. В первом случае задача решается с помощью мониторов [8,13], во втором – путем создания некоторого ресурса между взаимодействующими процессами [14,15].В обоих случаях необходимо введение механизмов блокировки-разблокировки процессов, которые обеспечивали бы бесконфликтность в любых ситуациях и были бы инвариантны к конкретным условиям взаимодействий. Основу таких механизмов составляет известная концепция семафоров [16,17]. Она заключается в выделении специальных переменных – СЕМАФОРОВ (s) и операций над ними: $P(s)$ и $V(s)$, которые управляют структурами типа "очередь". Семафоры с целочисленными значениями и дисциплиной обслуживания $fifo$ (первый пришел – первый обслужился) очереди ждущих процессов являются достаточным средством для организации произвольных дисциплин обработки очередей [5].

Построение программ, занимающихся составлением расписаний на пользование ресурсами [13], требует введения в качестве условных переменных значений мощности, списков готовых к выполнению и ждущих процессов. В этой же связи появляется необходимость учитывать события, являющиеся комбинацией событий над семафорами, т.е. функций над множествами семафоров и переменных [19,20].

Анализ существующих систем синхронизации параллельных процессов показывает, что достаточно полной, простой и наглядной моделью являются так называемые с е т и П е т р и [19,20]. Выбор этой модели для описания взаимодействий процессов оправдывается еще и тем обстоятельством, что она удобна для аппаратной реализации управления процессами. Это свойство важно, так как семафоры (метки), используемые для синхронизации, вырабатываются в разных элементарных машинах, и поэтому аппаратная реализация функций над ними существенно улучшит эффективность синхронизации.

В соответствии с моделью сети Петри, семантика управляющих воздействий ЖДАТЬ $\{\ldots\}$ и ИЗВЕСТИТЬ $\{\ldots\}$ описывается следующим образом:

$$\underline{when}\ L_k = \alpha \wedge \underset{i\in A}{\&}(s_i>0)\ \underline{do}\ L_k \leftarrow \alpha';\ \underset{i\in B}{s_i \leftarrow s_i-1};\ \underset{i\in C}{s_i \leftarrow s_i+1}\ \underline{od},$$

где A, B, C – подмножества множества семафоров $\{s_1,\ldots,s_m\}$; $A \cap C = \phi$, $B \subseteq A$, L_k – счетчик команд процесса k.

§3. Структура и функционирование однородной вычислительной системы

1. С т р у к т у р н ы е и ф у н к ц и о н а л ь н ы е о с о б е н н о с т и э л е м е н т а р н о й м а ш и н ы.

Архитектура однородной вычислительной системы дает представление о ее функционировании как о реализации системы асинхронных процессов. При этом процессы, составляющие операционную систему ОВС, и процессы пользователей системы порождаются, уничтожаются и взаимодействуют друг с другом посредством одного и того же набора операций управления процессами. Выделение совокупности процессов, образующих функциональный элемент системы – элементарную машину, основывается на учете технологических и системных факторов, а также на свойствах алгоритмов вычислительных процедур.

Технологические особенности элементной базы лимитируют структуру элементарной машины классом вычислительных устройств с динамической программируемостью управления преобразованием данных. В указанном классе устройств каждый акт преобразования данных сопровождается выборкой соответствующей команды из памяти устройства. Технологические ограничения обусловливают более быстрое взаимодействие инструкций, расположенных внутри одной ЭМ по сравнению со взаимодействием инструкций, находящихся в разных ЭМ.

Процесс, представляя собой набор вычислительных операторов и операций управления процессами, требует определенных ресурсов для своей реализации. Тем самым число микроинструкций, составляющих вычислительные операторы, количество процессов, а также число семафоров и конструкций над ними, имеют свои предельные значе-

ния, обусловленные структурными характеристиками элементарной машины.

Указанные предельные значения не должны быть слишком малыми. Анализ вычислительных процедур [21] показал, что большинство из них эффективно представляется в виде параллельных алгоритмов посредством методики крупноблочного распараллеливания. Получающиеся при этом параллельные программы состоят из вычислительных операторов и схем обмена между операторами. Эффективная реализация интенсивных взаимодействий инструкций, составляющих оператор, требует размещения их в одной ЭМ. Следовательно, ресурсов ЭМ должно хватать для реализации вычислительных операторов широкого класса задач.

Кроме того, ресурсы ЭМ должны допускать существование по крайней мере двух процессов: вычислительного процесса пользователя и процесса операционной системы, осуществляющего межмашинные взаимодействия.

Аппаратно-программная реализация ядра операционной системы элементарной машины предполагает наличие средств задания и исполнения входных и выходных мониторов, а также тел процессов.

Выбор сетей Петри в качестве синхронизирующих примитивов определяет структуру мониторов и тел процессов [20] как наборов инструкций следующего вида:

$$\underline{when}\ L_k = \alpha\ \underline{do}\ L_k \leftarrow g(E);\ E \leftarrow h(E)\ \underline{od}$$

$$\underline{when}\ L_k = \alpha \wedge \&_{i\in A}(s_i > 0);\ \underline{do}\ s_i \leftarrow s_i - 1;\ _{i\in B}\ s_i \leftarrow s_i + 1;\ _{i\in C}$$
$$L_k \leftarrow \alpha';\ E \leftarrow h(E)\ \underline{od}$$

где L_k – счетчик команд процесса k ($k \in \{1, \dots \theta\}$, θ – максимально возможное количество загруженных в ЭМ процессов), $h(E)$ и $g(E)$ – преобразования состояния E памятей элементарных машин, задаваемые инструкцией; $\alpha, \alpha' \in E$, $\alpha \neq \alpha'$.

Порождение процесса сводится к созданию условий для проверки предикатов инструкций процесса, заданию областей определения функций $g(E)$ и $h(E)$, а также выделению и начальной установке счетчика команд и подмножеств семафоров A, B, C для каждой инструкции второго вида.

Адекватность вложения вышеперечисленных данных процесса в структуру элементарной машины и характер функционирования ЭМ при реализации подмножеств асинхронных процессов определяют эффективность элементарной машины в целом.

Множество E элементов памяти системы включает в себя регистры, оперативные памяти и внешние устройства элементарных машин. В связи с этим в процессе выполнения элементарной машиной инструкций может потребоваться ввод-вывод данных по каналам обмена с внешними устройствами и другими элементарными машинами.

Таким образом, любое внешнее устройство, подключенное хотя бы к одной ЭМ, является обобществленным. Любая машина однородной вычислительной системы может получить к нему доступ и управлять работой этого внешнего устройства. Эффективное функционирование ОВС обусловливает подключение внешних устройств и организацию межмашинных взаимодействий посредством канала ввода-вывода. Это связано, во-первых, с тем, что вся система может рассматриваться как процессор (подсистема вплоть до одной ЭМ) и внешнее устройство (остальная часть системы). Во-вторых, режим работы системы предполагает транзитные передачи; желательно, чтобы они не затрагивали обрабатывающей части (процессора) ЭМ. В-третьих, асинхронность работы модулей и внешних устройств вызывает необходимость буферизации данных. В-четвертых, функции расчета путей между взаимодействующими ЭМ достаточно сложны, чтобы быть выполненными простой аппаратурой.

Необходимость структурированного распределения ресурсов системы приводит к введению подсистем с предоставлением последним всех внутренних ресурсов. Выделение подсистем реализуется установкой границ [22], исключающих возможность вмешательства извне. В то же время алгоритмы планирования, реконфигурации системы и диагностики основываются на подобном вмешательстве. В связи с этим возникает необходимость в двух режимах работы межмашинных связей: нормальном и с преодолением границ. В первом режиме невозможно войти в ЭМ, выставившую признак границы по направлению, связывающему ее с ЭМ, инициирующей передачу. Во втором режиме признак границы игнорируется; ЭМ, выставившая его при поступлении в нее информации, переключается (посредством прерывания) на программу, предусматривающую обработку ситуаций, связанных с преодолением границы. Эта программа обязательно входит в ядро операционной системы.

Исследования алгоритмов [2I] показали, что методика крупно-блочного распараллеливания вводит массовые взаимодействия, при которых одни и те же данные передаются большому числу процессов, реализующих то же задание, что и процесс, выдавший данные. В этой связи возникает необходимость иметь аппаратно-программные средства, позволяющие выделять и задавать режимы функционирования совокупностей межмашинных связей и внешних устройств. Указанные средства осуществляют сбор сигналов готовности к приему очередного кода от заранее установленных внешних устройств и межмашинных связей. По получении всех сигналов вырабатывается обобщенный сигнал готовности, поступающий в предварительно указанное передающее внешнее устройство.

Соответствие принимающих устройств передающим носят название соединительных функций или матриц коммутации [I]. Указанные матрицы, будучи заданными в каждой элементарной машине, определяют разбиение системы на подмножества взаимодействующих машин. В условиях непредсказуемости таких подмножеств и существования транзитных передач [23] необходимо иметь аппарат, позволяющий определять матрицы коммутаций в динамике вычислительного процесса. Основу такого аппарата составляют алгоритмы задания адресов элементарных машин на структуре ОВС, описываемой графом, вершинам которого соответствуют элементарные машины, а дугам – связи между ними. Определение путей между взаимодействующими машинами сводится к соответствующей экстремальной задаче поиска пути на графе. В общем случае решение этой задачи трудоемко, поэтому способ задания адресации следует выбирать исходя из соображений простоты расчета коммутаций.

2. Структура однородной вычислительной системы.

Выбор графа межмашинных связей определяется конструктивными ограничениями, степенью сложности алгоритма межмашинных взаимодействий, структурной живучестью, структурной коммутируемостью, максимальной и средней задержками при транзитных передачах информации [24,25].

В случае, если конкретная область применения не накладывает жестких требований на граф межмашинных связей, наилучшими структурами ОВС являются однородные графы, обладающие минимальным диаметром и минимальным средним диаметром. Под средним диаметром

понимается математическое ожидание расстояния между парой вершин при равновесном выборе пар. Широкий класс таких графов составляют дистанционно-транзитивные графы [26], все минимальные циклы которых имеют максимально возможную длину для данных N (число вершин) и ρ (степень вершины), т.е. максимально возможный обхват, минимальный диаметр и минимальный средний диаметр.

Большой интерес представляют также семейства графов с заданной величиной обхвата t. Такие графы описывают структуры ОВС, допускающие наращивание числа машин в системе без коренной ломки имеющихся межмашинных соединений. Частным случаем указанных графов при $t = 4$ являются КАИС-структуры [25].

Степень вершины ρ существенно зависит от технологически допустимой пропускной способности одного направления межмашинной связи, а также алгоритмов межмашинных взаимодействий. Последние могут использовать временное, пространственное и пространственно-временное представления сети связи ОВС для межмашинных взаимодействий. Естественно, что с возрастанием ρ последние две формы разделения сети связи существенно усложняются, так как растет число возможных вариантов соединения взаимодействующих ЭМ.

Выбор точек подключения обобществленных внешних устройств осуществляется [27] на основе максимизации вероятности доступа к заданному типу внешних устройств и минимизации времени обращения.

3. Идентификация элементарных машин.

Операции порождения, уничтожения и взаимодействия процессов, выполняемые в динамике вычислительного процесса, требуют эффективной реализации алгоритмов распределения ресурсов системы. Взаимодействия между процессами, расположенными в различных ЭМ, происходят посредством каналов межмашинных связей. Управление каналами обусловило задание в имени процесса идентификатора элементарной машины, на которой этот процесс реализуется.

Основные требования к системе идентификации элементарных машин вытекают из необходимости эффективного функционирования выделенных совокупностей ЭМ. Сюда относятся, во-первых, минимизация отношения адресной информации к той, которой обмениваются ЭМ, во-вторых, минимизация числа транзитных ЭМ, в-третьих, максимизация числа одновременно осуществляемых обменов.

В качестве системы адресации предлагается следующая. У элементарной машины допускается несколько имен, определяющих ее принадлежность к определенным группам. Имена ЭМ записываются в группы регистров, содержащих, во-первых, код (в дальнейшем называемый цветом), определяющий принадлежность ЭМ некоторому подмножеству, выделенному посредством этого цвета, во-вторых, упорядоченную последовательность ярусных номеров, принимающих значения из множества $\{0,1,\ldots,2^{\alpha}-1\}$, в-третьих, сопоставленные каждому ярусному номеру функции, указывающие на связь между номерами внутри яруса.

ЭМ адресуется ярусным номером нулевого ранга непосредственно к τ ЭМ, $\tau \in \{1,2,\ldots,2^{\alpha}\}$, образующим связный подграф структуры ОВС. Каждому ребру, инцидентному ЭМ, сопоставлена ярусная функция ранга 0, которая принимает значение I при аргументе $x \in \{0,1,\ldots,2^{\alpha}-1\}$, если это ребро лежит на кратчайшем пути, связывающем данную ЭМ и ЭМ с ярусным номером ранга 0, равным x.

Подмножество из $\tau \in \{1,2,\ldots,2^{\alpha}\}$ ЭМ, адресуемых непосредственно, имеет ярусный номер ранга I, приписанный всем ЭМ этого подмножества. Множество из $\mathcal{R}_0$ ($\mathcal{R}_0 \in \{1,2,\ldots,2^{\alpha}\}$) таких подмножеств адресуется относительно друг друга ярусными номерами ранга I, принимающими значения из совокупности $\{0,1,2,\ldots,\mathcal{R}_0\}$. В этом случае каждому направлению передачи из рассматриваемой ЭМ соответствует ярусная функция ранга I, принимающая значение единицы при аргументе x, если указанное направление лежит на кратчайшем пути из данной ЭМ в ЭМ с равным x ярусным номером ранга I.

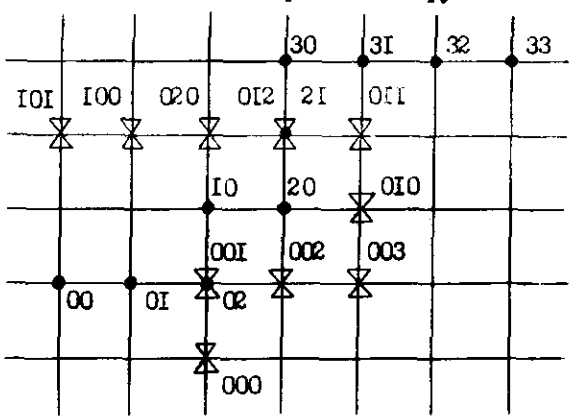


Рис. 3

Аналогично вводятся ярусные номера и ярусные функция следующих рангов. На рис.3 приведены две подсистемы с разными цветами, на которых заданы для каждой ЭМ ярусные номера (● - подсистема I, X - подсистема II, ■ - подсистема I∩II подсистема Д).

Рассмотренная выше система адресации инвариантна к структуре ОВС и является относительной [22], позволяющей создавать подсистемы с заданным количеством и взаимным положением ЭМ. Выделение подсистем фиксированного цвета дает возможность задавать число рангов адресации в зависимости от числа ЭМ, т.е. определять необходимый уровень адресной информации (длину адресной посылки). Указанная система адресации позволяет также при оптимально выбранном значении α осуществлять поиск требуемых ЭМ по кратчайшим путям.

В основу а л г о р и т м а о п р е д е л е н и я т р а н з и т н ы х м а ш и н могут быть положены следующие соображения. Пусть i - наибольший ранг, ярусный номер которого отличен от соответствующего ярусного номера ЭМ, выполняющей в данный момент роль транзитной. Тогда в качестве направления передачи из рассматриваемой транзитной ЭМ выбирается то, на котором ярусная функция ранга i равна I при аргументе, равном ярусному номеру ранга i искомой ЭМ. Таким образом, начиная с наибольших рангов, происходит процесс, выравнивающий ярусные номера. Машина, в которой произойдет совпадение ярусных номеров, и будет искомой.

4. С и с т е м н ы е к о м а н д ы.

Элементарная машина как составная часть ОВС предусматривает наличие в ней команд, отличающихся от предполагаемых, используемых в процессорах (микропроцессорах). Это связано, во-первых, с необходимостью программного включения-отключения питания ЭМ и его изменения с целью профилактического тестирования, во-вторых, с организацией взаимодействий процессов, расположенных как в одной, так и в разных ЭМ, в-третьих, с необходимостью организации специальных режимов функционирования выделенных совокупностей ЭМ.

Включение питания ЭМ совмещается с ее начальной установкой, при которой очищаются таблицы процессов. После этого ЭМ готова к работе в системе. Команды включения-отключения и изменения питания ЭМ могут приходить с консоли модуля, из него самого, а также из соседних модулей.

Выбор в качестве формы синхронизации сетей Петри обусловливает команду настройки, определенную на множестве семафоров ЭМ. Семафоры идентифицируются именем ЭМ, в которой они расположены, и собственным именем внутри ЭМ. Форма команды настройки есть $\Omega\{\alpha_0 \alpha_1 \ldots \alpha_k * \alpha_{k+1} \ldots \alpha_n * \beta_0 \beta_1 \ldots \beta_m\}\{\alpha_{k+1} \ldots \alpha_n\}$

$\{\beta_0 \dots \beta_m\}$ - подмножества семафоров A, B и C соответственно (см. §2, п.4).

В конструкции $if\ L_k = \alpha \wedge \&_{i\in A}\ (s_i > 0)\ then\ do\ L_k \leftarrow \alpha';\ s_i \underset{i\in B}{\leftarrow} s_i - 1;\ s_i \underset{i\in C}{\leftarrow} s_i + 1\ od$ обозначим логическое выражение $L_k = \alpha \wedge \&_{i\in A}\ (s_i > 0)$ чрез Ω_ℓ, где ℓ - индекс занятой межмашинной связи. Область значений $\ell \in \{0,1,\dots,L\}$ определяется конструктивными характеристиками ЭМ; значение $\ell = 0$ указывает на то, что все семафоры принадлежат одной ЭМ и межмашинной связи не требуется. Команда ОУП $\Omega^\ell \alpha$ по значению $\Omega^\ell = 1$ осуществит переход к команде с адресом α. Выполнение команды ОУП $\Omega^\ell \alpha$ при значении $\Omega^\ell = 0$ вызывает появление события ЖДАТЬ $\{*_\ell, \Omega^\ell, \gamma\}$, означающего постановку текущего процесса в ожидание $\Omega^\ell = 1$.

Для управления сетью связи, образующейся при объединении ЭМ в ОВС, предусмотрены команды НАД $\{\beta_0\,\beta_1 \dots \beta_k\}\alpha\gamma$, НАС $\{\beta_0\,\beta_1 \dots \beta_k\}\alpha\gamma$ и ОБМ $\alpha\gamma$. Команда НАД $\{\beta_0\,\beta_1 \dots \beta_k\}\alpha\gamma$ выполняет адресную настройку, состоящую в том, что данные α принимают ЭМ, адреса которых определяются $\{\beta_0\,\beta_1 \dots \beta_k\}$ по следующему правилу. Адрес β_0 задает ЭМ, которая должна в результате выполнения НАД $\{\beta_0\,\beta_1 \dots \beta_k\}\alpha\gamma$ получить адрес с цветом, указанным в β_1, и ярусные номера 00...0 для всех рангов, число которых совпадает с числом рангов β_1. Ярусный номер нулевого ранга β_1 задает допустимую область значений ярусных номеров создаваемой адресации. Ярусные номера адресов $\beta_2 \dots \beta_k$ указывают на исключаемые подмножества адресов. Так, например, задание в качестве β_j, $j \in (2,k)$, ярусных номеров 01 в системе адресации, использующей четыре ярусных номера, означает удаление из формируемой системы адресации множества адресов $01\rho q$, где ρ и q пробегают все допустимые значения.

Адресации цветов, указанных в $\beta_2 \dots \beta_k$, подлежат уничтожению в процессе выполнения НАД $\{\beta_0 \dots \beta_k\}\alpha\gamma$, если в этом возникает необходимость. Последняя вызывается отсутствием свободных мест в массиве адресов ЭМ, включаемых в строящуюся систему адресов. Если в процессе настройки затрагивается адресация цвета $i \in \{\beta_2 \dots \beta_k\}$, то происходит полное ее стирание, причем дальнейшее выполнение НАД $\{\beta_0 \dots \beta_k\}\alpha\gamma$ задерживается до конца указанного процесса. Если команда НАД $\{\beta_0\,\beta_1 \dots \beta_k\}\alpha\gamma$ в силу каких-либо причин не может быть выполнена, то результатом ее работы будет переход на γ с вектором $\{\beta_0^0 \dots \beta_k^0\}$, указывающим на реально построенную систему адресации.

Естественно, что никакого изменения адресов не происходит при совпадении цветов β_0 и β_1, а также если все требуемые ярусные номера уже присутствуют в адресации рассматриваемого цвета. В этом случае выполнение НАД $\{\beta_0 \dots \beta_k\}\alpha\gamma$ сводится к передаче данных α.

Команда НАС $\{\beta_0 \dots \beta_k\}\alpha\gamma$ позволяет передать данные α всем ЭМ, адреса которых указаны в $\{\beta_0\,\beta_1 \dots \beta_k\}$. В отличие от НАД $\{\dots\}$ эта команда не производит никаких изменений в адресации. В случае, если хотя бы один из указанных в $\{\beta_0 \dots \beta_k\}$ адресов отсутствует в системе, произойдет переход на γ с идентификацией отсутствующих адресов.

В обеих командах настройки данные α несут в себе имя передаваемого массива данных, а в случае коммутации сообщений сами данные и указания на способ передачи (коммутация каналов, коммутация сообщений).

Выполнение команд настройки с указанием на коммутацию каналов завершается построением пути передачи данных между ЭМ с адресом β_0 и всеми остальными упомянутыми соответствующим образом в $\{\beta_0 \dots \beta_k\}$.

Указанный скоммутированный канал разрушается после выполнения команды ОБМ $\alpha\gamma$, когда передан весь массив данных, упомянутых в команде настройки, создавшей этот канал.

Следующая группа системных команд связана с организацией с и н х р о н н о - п р о г р а м м н о г о р е ж и м а [28] и представлена командами настройки СПР $\{\beta_0\,\beta_1 \dots \beta_k\}\alpha$ и настройки конфигурации арифметическо-логического устройства САР $\{\beta_0\,\beta_1 \dots \beta_k\}\alpha$.

СПР $\{\beta_0\,\beta_1 \dots \beta_k\}\alpha$ - настройка подмножества процессов с именами $\beta_0 \dots \beta_k$ для работы в синхронно-программном режиме, когда процесс β_0 выдает команды, а процессы $\beta_1 \dots \beta_k$ выполняют их над собственными данными.

Синхронно-программный режим предполагает наличие обязательных трех семафоров в каждом из процессов, выполняющихся в нем. Указанные подмножества семафоров задаются α. В соответствии с этим линии $\Omega^{\alpha_0}, \Omega^{\alpha_1}, \Omega^{\alpha_2}$ служат для распространения сигналов: ПЕРЕХОДА НЕТ, КОМАНДА ВЫДАНА, КОМАНДА ИСПОЛНЕНА. Последний из них запускает очередную команду, которая выдается каналом и совмещается с сигналом КОМАНДА ВЫДАНА. Основную трудность при реализации програм

в синхронно-программном режиме представляют команды ветвления. В ЭВМ используются два вида таких команд: $skip\gamma$ и $jump\gamma$. Команда $skip\gamma$ осуществляет передачу управления либо следующей команде, либо через одну, в зависимости от того, выполнено или нет условие γ; $jump\gamma$ – команда безусловного перехода на γ.

Выполнение команды $skip\gamma$ подмножеством машин, работающих в синхрорежиме, вызывает проверку условия γ в каждой из них. Если оно выполнено хотя бы в одной ЭМ, об этом сообщается по Ω^{α_0} и происходит выбор следующей команды. В противном случае выбор команды $jump$, с необходимостью следующей за $skip\gamma$, не происходит, и выдается команда через одну после выполнившейся.

Если необходимо выполнить команду $jump\gamma$, следующую за $skip\beta$, анализируется адрес текущей команды (A), а именно адрес $jump\gamma$ и значение γ. Если $\gamma > A$, то выбирается следующая команда. Если $\gamma < A$, то γ заносится в текущий адрес канала и начинается выборка команд γ.

Каждый процесс синхронно-программного режима имеет свой текущий адрес, который постоянно сравнивается с адресом выдаваемой на шину команды, и только при их совпадении процесс начинает выполнение команды.

Подготовка работы со словами произвольной разрядности в синхронно-программном режиме производится при помощи команды САР $\{\beta_0 \dots \beta_k\}\alpha$, которая производит настройку ЭМ $\{\beta_0 \dots \beta_k\}$ для работы в системной арифметике α. При этом из d-разрядных регистров (сумматоров) составляются $(k+1)$ d-разрядные регистры (сумматоры).

Данные значения α могут обеспечить другие режимы групповой обработки данных. Например, можно задать настройку совокупности ЭМ $\{\beta_0 \dots \beta_k\}$ на работу в режиме обобщенных подстановок [29].

Для реализации межмашинных взаимодействий, предписываемых системными командами, элементарная машина должна иметь информационные линии, связывающие ее с другими ЭМ. Посредством информационных линий или путем аппаратного выделения образуются линии, задающие следующие подсистемы связи:

– линии Ω^{ℓ} ($\ell = 0,1,\dots,4$), служащие для выработки значения предиката;

– линии переноса, предназначенные для организации обработки векторов увеличенной размерности;

– линии расширения регистров и передачи их значений;

– информационные линии, по которым производится обмен данными и выдача команд в синхронно-программном режиме.

5. Функционирование однородной вычислительной системы.

Функционирование элементарной машины определяется внешней и внутренней дисциплинами [24] ОВС, задающими алгоритмы обслуживания внутренних потоков заявок на программно-аппаратные средства, процедуры устранения последствий отказов, организацию самодиагностики.

Локальные диспетчеры посредством устройств ввода-вывода осуществляют связь с внешним миром и формируют файлы заданий. Программы расписания ресурсов имеют иерархическую структуру, определяемую подмножествами ЭМ, управляемыми блоками уровней иерархии. Указанная иерархия порождается как система процессов базовой операционной системы (см. §2). Эффективное использование ресурсов системы предполагает их деление в соответствии с имеющейся в системе ситуацией и требованием представления для порождаемого процесса наилучших условий его реализации. В соответствии с этим операторы алгоритмов с трансляционной (коллекторной) схемой обмена [21] следует размещать в машинах, максимально близких к передающей (принимающей). Парносмежная (или параллельно-конвейерная) схема обмена требует размещения операторов в машинах, расположенных на гамильтоновом цикле структуры ОВС. Трансляционно-циклическая и нестационарная схемы обмена наилучшим образом реализуются на подграфе структуры ОВС с минимальным диаметром и минимальным средним диаметром.

Основной ресурс системы, разделяемый между процессами, есть память. Элементарная машина содержит 2^{m+n} ячеек памяти, разбитой на блоки по 2^m ячеек в каждом. Ячейка памяти идентифицируется номером блока α ($\alpha = 0,1,2,\dots,2^n-1$) и номером в блоке β ($\beta = 0,1,2,\dots,2^m-1$). Загружаемые в ЭМ программы записываются в виртуальных адресах, состоящих из виртуального номера блока и номера в блоке.

Каждый процесс определен на некотором подмножестве блоков памяти, связанных друг с другом посредством ссылок на ЭМ, которой принадлежит блок и номер блока в этой ЭМ. Таким образом, все блоки, выделенные для реализации некоторого функционально связанного подмножества процессов, образуют списковую структуру, позво-

ляющую рассматривать совокупность оперативных памятей ЭМ как единый ресурс – память системы. Элементарные машины, блоки памяти которых объединяются в единый ресурс – списковую структуру, имеют единый цвет адресации.

Идентификация блоков памяти задается тем, что в регистр, соответствующий блоку, записывается индекс защиты, указатель допустимых способов обработки данного блока, виртуальный номер блока, а также ссылки на блок, предшествующий данному и следующий за данным в списке, которому блок принадлежит.

Обращение к блоку памяти является корректным, если оно соответствует способу обработки блока, указанному в регистре блока. При попытке осуществить некорректное обращение к блоку памяти процесс прерывается и управление передается процессу, породившему его.

Резервирование блоков памяти производится оператором ВЫДЕЛИТЬ {цвет адресации; список ресурсов; убираемый индекс защиты; присваиваемый индекс защиты; дефицит}, где список ресурсов ::= адрес ЭМ, количество блоков | список ресурсов.

Оператор ВЫДЕЛИТЬ {...} осуществляет распределение ресурсов системы, задавая посредством адресации элементарных машин взаимное расположение (конфигурации) программных фрагментов. Так, если не указана ЭМ, в которой должно резервироваться требуемое число блоков памяти, то эти блоки резервируются в ближайшей (в смысле расстояния на графе связи) ЭМ, имеющей требуемый свободный ресурс. При этом указанная ЭМ с необходимостью получает адрес с цветом, совпадающий с тем, который указан в операторе ВЫДЕЛИТЬ { ... }. Все ссылки на ЭМ производятся посредством адресов этого цвета.

Оператор ВЫДЕЛИТЬ {...} появляется в одной из ЭМ, указанных в списке ресурсов. Количество блоков памяти ЭМ с индексом защиты, равным убираемому индексу защиты, сравнивается с требуемым от этой ЭМ количеством блоков, указанным в списке ресурсов. Если количество имеющихся блоков меньше требуемого, то разница между ними добавляется к дефициту, а все имеющиеся блоки путем оформления соответствующих ссылок и изменения индекса защиты переходят в создаваемый резерв памяти. Если количество имеющихся блоков больше или равно заданному в операторе ВЫДЕЛИТЬ {...}, то требуемое число блоков переходит в создаваемый резерв памяти.

После выполнения в модуле всех вышеперечисленных действий, если список ресурсов еще не исчерпан, соответствующим образом измененный оператор ВЫДЕЛИТЬ {...} (в котором удалено из списка ресурсов имя ЭМ, выполняющей оператор) передается в следующую в списке ресурсов элементарную машину.

Если список ресурсов исчерпан, то обратным проходом по созданному резерву памяти формируется картина действительного ее заполнения, которая передается процессу, производящему резервирование.

Установка способа обработки блока и виртуального номера блока происходит при загрузке данных в блок памяти.

Устранение последствий отказов базируется на представлении вычислительных процедур в структурированной форме, а именно как системы асинхронных процессов с дискретными событиями. Операторы управления процессами имеют точки выхода по ошибочным ситуациям. В связи с этим создается возможность проследить ход реализации вычислений и внести необходимые коррективы.

Элементарная машина является единицей самодиагностики ОВС. Самодиагностика подразумевает определение моментов пуска диагностических процедур и выделение исправной части системы. Первая из этих функций реализуется при помощи диагностического процесса, который может быть запущен в любой ЭМ либо внешним воздействием от аналогичного процесса другой ЭМ, либо собственным воздействием по истечении заданного времени от момента предыдущего запуска. Отрезок времени выбирается исходя из того, что прежде чем в данную ЭМ снова придет запускающее воздействие, оно должно обойти все другие ЭМ системы (заданную часть из них).

Запуск диагностического процесса ведет к реализации процедуры построения диагностического синдрома [10]. Указанная процедура на основании предположения о максимально возможном числе δ неисправных ЭМ вырабатывает коллективное мнение о неисправности какой-либо ЭМ ($\delta \le \left[\frac{N-1}{2}\right]$, где N – число ЭМ в ОВС, $[x]$ – целая часть x). Число ЭМ коллектива, выносящего решение, больше допустимого числа неисправных ЭМ, и каждая из машин коллектива имеет мнение о других его членах как об исправных.

6. Возможные реализации элементарной машины.

Анализ показал, что при существующих технико-экономических показателях БИС возможно создание однокорпусного специализированного микропроцессора – коммутационного модуля, реализующего канал ввода-вывода с непосредственным, минуя память, взаимодействием вне-

шних устройств и межмашинных связей. Введение в этот модуль механизмов задания и преодоления границ, а также соединительных функций позволяет достаточно эффективно чисто программными методами реализовать адресацию элементарных машин и механизмы выполнения системных команд на традиционных микропроцессорах аналогично тому, как это делается на мини-ЭВМ [I8,30]. Получаемая таким образом система имеет вид, изображенный на рис.I.

Повышение эффективности системы возможно путем введения в элементарную машину вместо универсальных микрокомпьютеров микропроцессоров, специализированных на задание адресации и выполнение системных взаимодействий. Указанные микропроцессоры будут реализовывать линии выработки Ω^{ℓ}, переносов и расширения регистров.

§4. Заключение

Наборы микропроцессорных БИС, выпускаемые в массовом количестве и широко используемые для построения микро-ЭВМ, поставили на повестку дня создание на их основе высокопроизводительных и надежных вычислительных систем. Такие системы невозможно строить без учета особенностей, обусловленных спецификой реализации на БИС, к которым прежде всего можно отнести малое число типов схем, и как следствие построение систем путем многократного их повторения. Получающиеся при этом модульные системы требуют высокой степени распараллеливания программ, в связи с чем возрастает доля межмодульных обменов. Достижение высокой эффективности ОВС требует аппаратных затрат на согласование параллельных процессов. Ограничения по числу выводов усложняют межмодульные коммутации, что ведет к необходимости использования шинных структур, образуемых специализированными для этого кристаллами.

В настоящей работе поставлены задачи и намечены пути построения ОВС на базе микропроцессорных БИС. Предлагаемый здесь подход показывает возможность эффективного использования преимущества БИС и преодоления трудностей, связанных с технологическими ограничениями, при создании высокопроизводительных вычислительных средств.

Литература

I. ЕВРЕИНОВ Э.В., КОСАРЕВ Ю.Г. Универсальные однородные вычислительные системы высокой производительности. Новосибирск, "Наука", I966.

2. Мультипроцессорные системы и параллельные вычисления. Под ред. Энслоу Ф.Г. М., "Мир", I976.

3. ЕВРЕИНОВ Э.В., ХОРОШЕВСКИЙ В.Г. Однородные вычислительные системы. -В кн.: Вычислительные системы. Вып.58. Новосибирск, I974, с. 32-6I.

4. ЕВРЕИНОВ Э.В., ЛОПАТО Г.П. Универсальная вычислительная система "Минск-222". -В кн.: Вычислительные системы. Вып.23. Новосибирск, I966, с. I3-20.

5. ВИНОКУРОВ В.Г., ДИМИТРИЕВ Ю.К., ЕВРЕИНОВ Э.В., КОСТЕЛЯНСКИЙ В.М., ЛЕХНОВА Г.М., МИРЕНКОВ Н.Н., РЕЗАНОВ В.В., ХОРОШЕВСКИЙ В.Г. Однородная вычислительная система из мини-машин. -В кн.: Вычислительные системы. Вып. 5I. Новосибирск, I972, с.I27-I45.

6. АФАНАСЬЕВ В.П., ГОЛОДОК В.Л., ГОРЯЧКИН В.И., ЕРЕМИН А.П., ЖЕЛТОВ М.П., ИЛЬИН М.Н., СЕДУХИН С.Г., ТОМИЛОВ Ю.Ф., ХОРОШЕВСКИЙ В.Г., ШУМ Л.С. Вычислительная система СУММА. -В кн.: Вычислительные системы. Вып.60. Вопросы теории и построения вычислительных систем. Новосибирск, I974, с. I53-I69.

7. Микропроцессоры и системы обработки данных. Обзор. Хвостанцев М.А. -"Зарубежная радиоэлектроника", I975, №9, ,с.3I-35.

8. HANSEN Per Brinch. Structural multiprogramming.-"Communs. ACM",1972,v.15,N 7,p.547-579.

9. RUMBAUGH J.E. A parallel asynchronous computer architecture for data flow programms.- Project MAC Massachusetts Institute of Technology. May 1975.

I0. PREPARATA F.P., METZE G., CHIEN R.T. On the connection Assignment problem of diagnosable system. - "IEEE Trans. on EC",1967, v.EC 16,N 6,p.848-853.

II. FORBES R.E., RUTHERFORD D.H., STEIGLITZ C.B., TUNG I. A self-diagnosable computer. - In: 1965 Fall Joint Computer Conf. AFIPS Proc.,1965,v.27, p.1073-1086.

I2. HABERMANN A.N. Operating system structure foundations of compueter Science.Mathemat.Centre Tracte 63. Amsterdam,1975.

I3. HOARE C.A.R. Monitors: an operating system: structuring concept . - "Communs.ACM", 1974,v.17,N 10,p.549-558.

I4. HANSEN Per Brinch. The Nucleus of a multiprogramming system . - "Communs.ACM", 1970,v.13, N 4,p.238-250.

I5. HABERMANN A.N. Synchronization of Communicating processes. - "Communs.ACM", ,1972,v.15,N 3,p.171-177.

I6. DIJKSTRA Edsger W. The structure of the "The" - Multiprogramming System . - "Communs.ACM", 1968,v.11, N 5,p.341-347.

I7. DIJKSTRA E.W. Cooperating sequential processes. Programming languages. Ed.by F.Genuys.London-New York,Acad.Press,1968.

I8. RADUE J.E., MULLINS J.M. Solving synchronization problems using semaphores. - "Software: practice and experience",1975,v.5, p.51-64.

19. SUHAS S.Patil. Limitations and capabilities of Dijkstra's semaphore primitives for coordination among processes. Project MAC. Computation structures Group Memo 57. February 1971. Mass. Inst.of Techn.

20. LIPTON R.J., SNYDER L., LALCSTEIN Y. A comparative study of models of parallel computation. – In: 15 Annual Symp. on Sch. and Automata Theory,1974, p.145-155.

21. КОСАРЕВ Ю.Г. О схемах обмена между ветвями параллельных алгоритмов. –В кн.: Вычислительные системы. Вып. 51. Новосибирск, 1972, с. 70-76.

22. ШУМ Л.С. О функциональной организации однородных вычислительных систем. –В кн.: Вычислительные системы. Вып. 39. Новосибирск, 1970, с. 81-89.

23. КОРНЕЕВ В.В. Анализ сетей передачи данных однородных вычислительных систем. –В кн.: Вычислительные системы. Вып.63. Теория однородных вычислительных систем. Новосибирск, 1975,с.86-102.

24. ВОРОБЬЕВ В.А., КОРНЕЕВ В.В. Некоторые вопросы теории структур однородных вычислительных систем. –В кн.: Вычислительные системы. Вып.60. Вопросы теории и построения вычислительных систем. Новосибирск, 1974, с.3-16.

25. КОРНЕЕВ В.В. О макроструктуре однородных вычислительных систем. –В кн.: Вычислительные системы. Вып.60. Вопросы теории и построения вычислительных систем. Новосибирск, 1974, с.17-34.

26. BIGGS N. Algebraic Graph Theory. Cambridge University Press,1974.

27. КОРНЕЕВ В.В. О подключении внешних устройств к однородной вычислительной системе. –В кн.: Вычислительные системы. Вып. 63. Теория однородных вычислительных систем, 1975, с.103-108.

28. ДИМИТРИЕВ Ю.К. О способе реализации ветвящихся программ при работе однородной вычислительной системы в синхронно-программном режиме. –В кн.: Вычислительные системы. Вып. 60. Вопросы теории и построения вычислительных систем. Новосибирск, 1974, с.103-114.

29. КОРНЕЕВ Ю.Н., ПИСКУНОВ С.В., СЕРГЕЕВ С.Н. Алгоритмы обобщенных подстановок и вопросы их интерпретации сетями автоматов и однородными машинами. –"Изв. АН СССР. Техническая кибернетика", 1971, № 6, с. 131-142.

30. СЕДУХИН С.Г., ЖЕЛТОВ М.П., КАШУН И.Н. Ядро операционной системы СУММА. –Настоящий сборник, с. 113-129.

Поступила в ред.-изд.отд.
16 апреля 1977 года