ВОПРОСЫ ТЕОРИИ И ПОСТРОЕНИЯ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ
(Вычислительные системы)
1979 год Выпуск 80

УДК 681.3.066: 519.682.1

ОПЕРАЦИОННАЯ СИСТЕМА СУММА

С.Г.Седухин

Однородная вычислительная система (ОВС) СУММА [1-2] ориентирована на решение задач сбора и обработки данных и управления в реальном времени. Организация функциональной целостности вычислительной системы СУММА возложена на операционную систему, которая дополняет аппаратное обеспечение и координирует работу основных элементов системы.

Вычисления на системе СУММА реализуются как система параллельных асинхронных взаимодействующих процессов. Взаимодействие выражается в управлении, синхронизации и коммуникации между процессами. Операционная система организует согласованную работу параллельных процессов как на уровне одной элементарной машины, так и на уровне системы элементарных машин.

§ 1. Архитектура ОВС СУММА

Функционально-конструктивным элементом ОВС является э л е м е н т а р н а я м а ш и н а (ЭМ). В состав ЭМ ОВС СУММА входит мини-ЭВМ "Электроника-1001" и системное устройство [3]. Системное устройство подключается к мини-ЭВМ как специализированное внешнее устройство, а к системным устройствам соседних ЭМ через стандартные двунаправленные линии связи.

Наращиваемая регулярная структура, образованная системными устройствами и линиями связи, представляет собой с е т ь с в я з и ОВС. Сеть связи служит для обмена управляющей информацией и данными между ЭМ системы.

Каждая ЭМ системы помечена р а б о ч и м и д е н т и ф и к а т о р о м, являющимся ее адресом при обменах через сеть связи ОВС СУММА. Рабочий идентификатор (адрес) ЭМ назначается операционной системой и хранится в регистре признака системного устройства*) данной ЭМ. Любая ЭМ системы может быть отмечена одновременно несколькими адресами, которые определяют степень участия данной ЭМ в системных обменах. Структура сети связи ОВС СУММА позволяет организовать асинхронный режим передачи данных между ЭМ системы.

Рис. I.

Формат сообщений, предназначенных для передачи через сеть связи вычислительной системы СУММА, стандартизован операционной системой. Каждое сообщение оформляется операционной системой в выходной пакет (рис.I), состоящий из служебной и информационной частей. В служебную часть пакета входят: номер пакета, длина пакета, контрольная сумма, управляющее и адресное поля. Информационная часть пакета, а также контрольная сумма, номер и длина пакета, предназначены для занесения в оперативную память адресуемых в адресном поле пакета ЭМ системы. Адресное и управляющее поля пакета используются для настройки сети связи.

*) Регистр признака системного устройства позволяет непосредственно адресовать до двенадцати элементарных машин.

При передаче данных через сеть связи используется волновой (лавинный) поиск ЭМ-абонентов [4] (рис.2, цифрами обозначено расстояние ЭМ от передатчика), т.е. тех ЭМ, которым предназначен данный передаваемый пакет. Волновой поиск состоит в автоматическом заполнении сети связи (но не памяти элементарных машин) одним пакетом, в адресном поле которого указан идентификатор искомой (искомых) ЭМ системы. Произвольное системное устройство в сети связи, приняв пакет по одному из направлений, размножает пакет по оставшимся направлениям, не отвлекая на это транзитные ЭМ (т.е. те ЭМ, рабочие идентификаторы которых не совпадают с заданным в пакете), гарантируя тем самым, что пакет придет к требуемой (требуемым) ЭМ.

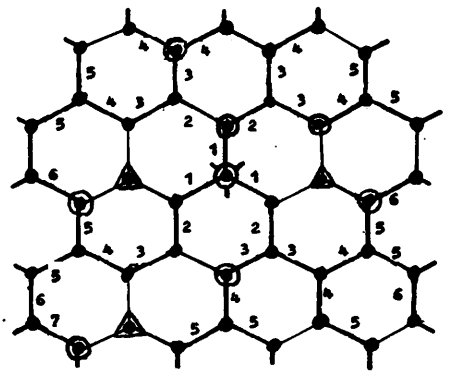


Рис.2.

Выбор метода передачи данных обусловили следующие требования к системе СУММА:

1) живучесть в условиях повреждений;

2) саморегулирующийся поиск при меняющейся конфигурации сети связи;

3) независимость адресации ЭМ для сети связи;

4) эффективность сети для небольшого числа ЭМ;

5) контроль сети связи во время функционирования системы.

Перечислим ряд достоинств волнового поиска, отвечающих вышеперечисленным требованиям.

а) Поиск вызываемой ЭМ-абонента происходит автоматически, и ЭМ могут сохранять свои рабочие идентификаторы при любых перемещениях в сети связи, что представляется удобным при диспетчировании и контроле системы.

б) Поиск характеризуется тенденцией выбирать самый короткий путь через сеть, обходя неисправные участки.

в) Поиск искомой ЭМ, выбор направлений и собственно передача данных производятся одновременно.

г) Выбор пути основывается на сложившемся состоянии и конфигурации сети связи в момент поиска, результатом чего является быстрая адаптация поиска. Поиск мгновенно приспосабливается к разрушениям и подключению к сети новых ЭМ.

д) Высокая стойкость к отказам и частичному разрушению: если даже остается всего один возможный путь для связи двух ЭМ-абонентов, этот путь будет найден.

е) Выбор пути происходит только по сети связи, отвлекая только требуемые ЭМ, что увеличивает эффективность системы.

ж) Требования к системе контроля значительно снижены, так как сеть - саморегулирующаяся и отсутствуют списки направлений, нуждающиеся в сохранении и коррекции наподобие сетей связи ЭВМ [4].

§ 2. Операционная система СУММА

2.1. Система параллельных процессов. Операционная обстановка вычислительной системы СУММА представляется системой асинхронных взаимодействующих процессов. П р о ц е с с о м называется последовательная реализация некоторого алгоритма в реальном времени. Процесс реализуется некоторым ресурсом системы (процессором, каналом, системным устройством и т.п.), существует наряду с другими процессами и при реализации взаимодействует с ними.

В системе различаются асинхронные процессы, реализуемые аппаратурой и программами. Программный процесс является последовательным исполнением запрограммированных на машинном языке действий аппаратуры процессора ЭМ. С каждым внешним устройством (накопитель на магнитной ленте, пишущая машинка, системное устройство и другие) связан соответствующий программный процесс, так называемый драйвер внешнего устройства, осуществляющий взаимодействие соответствующего внешнего устройства с ЭМ.

Внутри операционной системы процесс представлен структурой данных, характеризующей состояние процесса в любой момент времени. Перед началом процесса и тогда, когда он должен быть по какой-то причине прерван, переменные, описывающие его состояние, объединяются в структуру данных, называемую "в е к т о р о м с о с т о я н и я" и сохраняемую до тех пор, пока процесс не начнется (или не возобновится) Основными переменными, описывающими

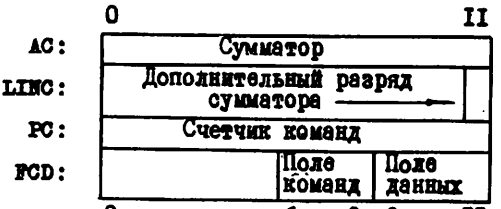


Рис.3.

состояние процесса и входящими в "вектор состояния" данного процесса, являются центральные регистры процессора ЭМ (рис.3).

По отношению к операционной системе любой процесс может находиться в одном из характерных состояний: потенциальном, реальном, ждущем, действующем, текущем, прерванном. Операционная система исполняет необходимые переходы процесса из состояния в состояние под воздействием самого процесса или других процессов вычислительной системы. При этом "вектор состояния" исключается или включается в некоторую очередь "векторов состояния", определяемую для каждого состояния.

2.2. Ядро операционной системы. Для обеспечения режима мультипрограммирования ЭМ устанавливаются те программные средства (процедуры), которыми будут пользоваться процессы. Эти процедуры образуют монитор управления процессами, который реализуется как непрерываемая часть операционной системы. Такое программное обеспечение называют ядром операционной системы [5,6]. Ядро реализует механизмы останова/запуска, синхронизации/коммуникации процессов, а также такое планирование времени процессора ЭМ, при котором каждый из процессов может работать (логически) так, будто он имеет свой собственный процессор. Иными словами, ядро операционной системы осуществляет динамические переходы процессов, реализующихся в данной ЭМ из одного состояния в другое.

Для запуска процесса, т.е. для перевода процесса из потенциального состояния в реальное, должен существовать некий текущий процесс, который, задавая "вектор состояния" запускаемого процесса, реализует процедуру START ядра операционной системы. При выполнении данной процедуры "вектор состояния" запускаемого процесса помещается в очередь прерванных (готовых к исполнению на процессоре) процессов, которые требуют процессорного времени для своего развития. Практически не существует ограничения на число процессов, которые могут быть определены (запущены) для операционной системы. Ограничение накладывается системой

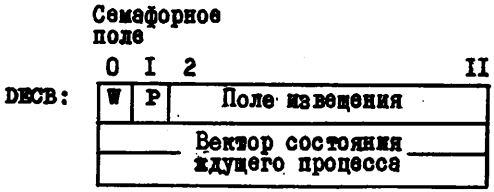
для того, чтобы уменьшить "накладные" расходы, связанные с управлением. Быстродействие процессора, конечно, ограничивает число процессов, которые могут быть активными и успешно реализоваться в течение заданного отрезка времени. Очевидно, что если бы ЭМ имела несколько центральных процессоров, то они могли бы быть загружены "векторами состояний" процессоров, взятых из очереди прерванных процессов, что повышало бы общую производительность вычислительной системы, правда, до определенного уровня [7].

Находящийся в текущем состоянии реальный процесс может перейти в потенциальное состояние, используя процедуру STOP ядра операционной системы. При этом операционная система удаляет "вектор состояния" данного процесса из области своего действия. Таким образом, наличие в ядре операционной системы процедур запуска (START) и остановки (STOP) процесса позволяет создавать и уничтожать программные процессы в пределах одной ЭМ.

Потенциальное состояние процесса отображено в ядре операционной системы максимально допустимой глубиной мультипрограммирования ЭМ, что отражается на допустимых размерах соответствующих очередей.

Гармоничное взаимодействие процессов регулируется посредством определенных общих синхронизирующих процедур, входящих в состав ядра операционной системы. На различных уровнях вычислительной системы управление асинхронными процессами (аппаратными и программными) осуществляется по-разному. Синхронизация процессора ЭМ и внешних устройств реализуется через систему прерывания. При реализации режима разделения времени может применяться генератор временных прерываний (таймер). Каждый раз, когда возникает временное прерывание, процессору назначается новый процесс из очереди прерванных процессов. На более высоких уровнях системы для управления асинхронными процессами используются программные средства.

Принципиально существуют две основные формы синхронизации в системе совместно протекающих процессов, одна в целях выборки общих данных, а другая - в целях коммуникации. В основе синхронизации процессов лежит аппарат работы с событиями, описываемый в терминах концепции семафоров [8]. Каждому событию в системе процессов сопоставляется соответствующий семафор так,что,с одной стороны, некоторый процесс может ожидать определенное событие, а с другой - процесс, в ведении которого находится событие,- сигнализировать ждущему процессу о наступлении ожидаемого события.



W - индикатор ожидания
P - индикатор извещения

Рис.4 .

Синхронизация процессов, реализующихся в пределах одной ЭМ системы СУММА, осуществляется с помощью процедур ядра операционной системы WAIT (ждать) и POST (известить), работающих с таблицей управления событием [9] (рис.4). Каждому событию в системе параллельных процессов сопоставляется своя таблица управления, содержащая семафорное поле из индикаторов ожидания W и извещения P , поля извещения и поля "вектора состояния".

Таблица управления событием обеспечивает связь между процессом, ждущим определенное событие, и процессом, извещающим о завершении данного события. При начальном определении таблицы управления событием индикаторы W и P должны быть сброшены.

Процесс может ожидать определенного события в системе процессов, указывая соответствующую таблицу управления при использовании процедуры WAIT . При этом если событие ранее не произошло (P = 0), то "вектор состояния" данного процесса переводится операционной системой в очередь ждущих процессов. В этом случае говорят, что процесс находится в ждущем состоянии.

Действующий программный процесс, совершив существенное для системы процессов событие, переводит процесс, ждущий этого события (W = I), в действующее состояние, задавая соответствующую таблицу управления событием при использовании процедуры POST .При этом "вектор состояния" ждущего процесса переводится операционной системой в очередь прерванных процессов. Заметим, что для обеспечения более полной информации о событии, извещающий процесс не только сигнализирует о наступлении события, но и извещает ждущий процесс о том, как произошло событие (каков исход события). Для этого извещающий процесс задает (при входе в процедуру POST) определенный "код извещения", который помещается в поле извещения соответствующей таблицы управления событием.

Отметим также, что синхронизация действий операционной системы (межмашинный обмен, ввод/вывод и др.) и функциональных процес-

сов, по запросам которых работает операционная система, осуществляется по этому же механизму, т.е. синхронизация процессов операционной системы и функциональных (прикладных) процессов осуществляется по единому правилу.

Если программный процесс исполняется процессором ЭМ,т.е. "вектор состояния" данного процесса содержится на регистрах процессора, то считается, что процесс находится в текущем состоянии. Все остальные действующие процессы образуют очередь прерванных (готовых к исполнению) процессов на реализацию к процессору ЭМ.Очередь хранит "векторы состояний" процессов, находящихся в прерванном состоянии,и в настоящей разработке ядра операционной системы представляет собой стек.

Текущий программный процесс может быть прерван аппаратным процессом (внешним устройством) через систему прерывания ЭМ и переведен операционной системой в прерванное состояние.Прерванный процесс активизируется (переводится в текущее состояние) ядром операционной системы,которое выбирает соответствующий "вектор состояния" из очереди прерванных процессов на основе приоритетного правила "последним пришел - первым обслужен" (LIFO). Диаграмма переходов программного процесса из состояния в состояние в мультипрограммном режиме работы ЭМ приведена на рис.5.

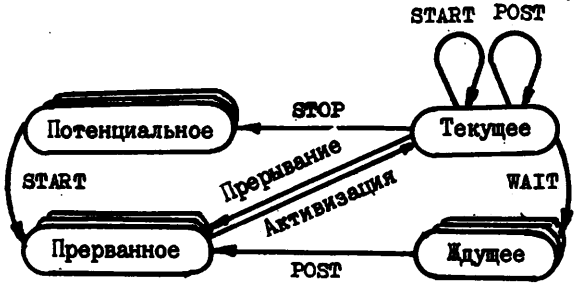


Рис. 5

При взаимодействии процессы обычно передают информацию. В силу этого, кроме операций "синхронизации", требуются операции "коммуникации", которые выполняют и синхронизацию, и передачу данных. В работах [10,11] показано, что с логической точки зрения для коммуникации процессов, реализующихся в единой среде, достаточно средств синхронизации, обеспечиваемых семафорами. Вследствие этого, операции коммуникации не включены в ядро операционной системы на уровне ЭМ. Однако для осуществления коммуникации процессов , реализующихся в разных ЭМ однородной вычислительной системы СУММА,

т.е. реализующихся в различной среде, только средств синхрониза - ции становится недостаточно. Коммуникация процессов, реализующихся в разных ЭМ системы, будет подробно рассмотрена при изложении распределенного монитора взаимодействия процессов, а пока отметим только, что для задания взаимодействий между процессами, активных в различных ЭМ системы, на уровне ядра операционной системы определен так называемый LAM-запрос (см. § 3), который фиксируется ядром в о ч е р е д и з а п р о с о в на дистанционное взаимодействие данной ЭМ. С этой очередью запросов работает распределенный монитор взаимодействия процессов.

Задаваемый функциональным процессом LAM-запрос позволяет процессу данной ЭМ произвести: а) коммуникацию с произвольным процессом, реализующимся в любой ЭМ системы путем чтения/записи информации из/в любой ЭМ системы, и используя при этом б) синхронизацию с процессом, реализующимся в этой ЭМ, а также осуществить в) запуск процесса в произвольной ЭМ однородной вычислительной системы СУММА.

На уровне ЭМ ядро операционной системы "скрывает" физические устройства ввода/вывода, так как логический режим их работы эквивалентен программному процессу. Драйвер внешнего устройства, являясь программным процессом, обеспечивает взаимодействие функциональных (прикладных) процессов с устройством ввода/вывода.

Структура ядра операционной системы приведена на рис.6. Ядро операционной системы состоит из набора модулей, которые используются функциональными процессами, а также драйверов внешних устройств, обеспечивающих логический режим работы ввода/вывода. Функционирование модулей ядра непосредственно следует из вышеописанного.

Таким образом, ядро операционной системы ОВС СУММА является средой, в которой выполняются в мультипрограммном режиме программы и ввод/вывод, управляемые однородно как параллельные асинхронные процессы данной ЭМ.

2.3. Распределенный монитор взаимодействия процессов. Для реализации взаимодействий между процессами, активных в различных ЭМ системы, необходимо осуществить обмен сообщениями через сеть связи между данными ЭМ. Коммуникация, или обмен сообщениями между удаленными процессами, требует выработки специальных соглашений относительно процедуры управления сетью связи и процедуры обмена сообщениями между удаленными процессами. Процедуры обмена сообщениями принято называть п р о т о к о л а м и [4].

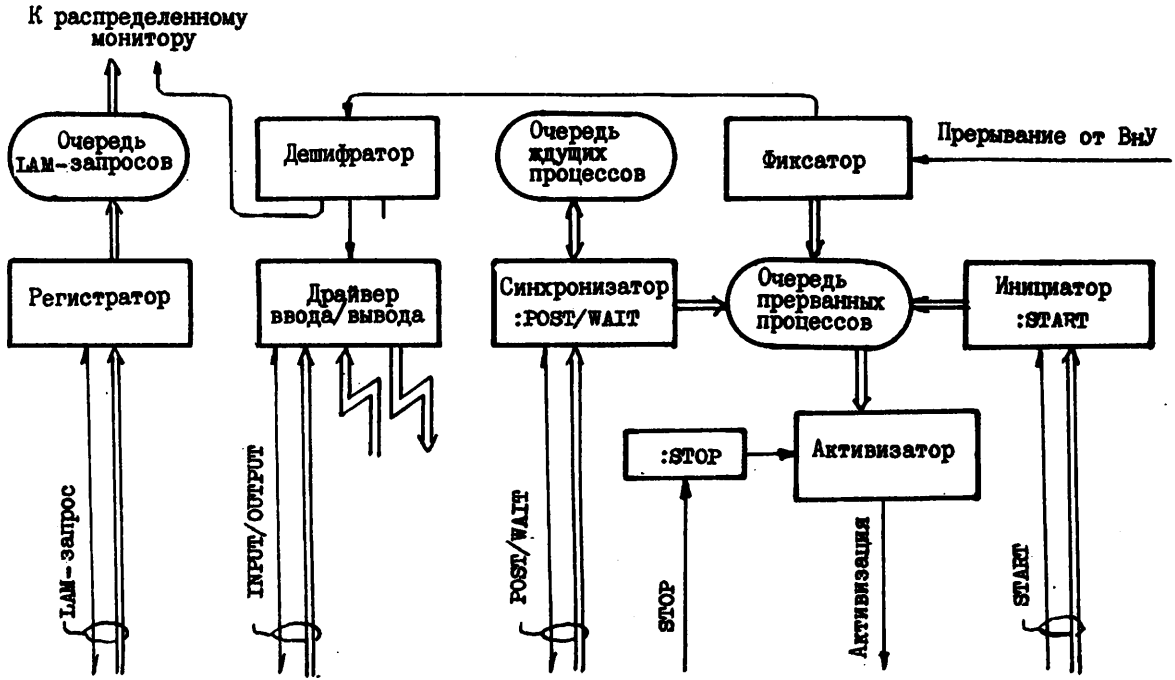


Рис. 6. Структура ядра операционной системы

Процедуры управления сетью связи и обмена сообщениями между удаленными, активными в разных ЭМ процессами реализуются в распределенном мониторе взаимодействия процессов. Распределенный монитор строится над ядром операционной системы и является, так же как и ядро, резидентом в каждой ЭМ системы. Данный монитор осуществляет согласованное взаимодействие удаленных процессов, реализуя режим параллельной обработки [7].

Анализ централизованных и децентрализованных дисциплин управления показал [12], что невозможно отдать предпочтение чисто централизованному или чисто децентрализованному управлению сетью связи ОВС при решении задач сбора и обработки данных, управления и регулирования в реальном времени, т.е. тогда, когда запросы на сеть связи ОВС со стороны функциональных (прикладных) процессов возникают в заранее непредсказуемые моменты времени и неизвестен тип взаимодействия по данному запросу (например, какие ЭМ будут участвовать в данном взаимодействии).

Распределенный монитор операционной системы СУММА реализует частично децентрализованный принцип управления сетью связи ОВС, при котором сеть связи рассматривается как общий ресурс системы. Для управления доступом функциональных процессов к сети связи ядро операционной системы каждой ЭМ системы создает очередь запросов (LAM-запросы) на данный ресурс, но в каждый момент времени распределенный монитор операционной системы СУММА разрешает доступ к сети связи только одной ЭМ системы, которая, реализуя протоколы обмена сообщениями между ЭМ, обслуживает свою очередь LAM-запросов.

Адресом ЭМ при обменах сообщениями через сеть связи ОВС СУММА является рабочий идентификатор ЭМ. Кроме рабочего идентификатора, ЭМ, управляющая в данное время сетью связи (т.е. обслуживающая свою очередь LAM-запросов), помечена идентификатором управления, хранящегося, так же как и рабочий идентификатор, в регистре признака системного устройства. Данная ЭМ системы является управляющей ЭМ, а остальные ЭМ системы, вовлекаемые в обмен через сеть связи, являются управляемыми ЭМ [5].

При функционировании ОВС СУММА любая ЭМ системы может быть в один момент времени управляющей, а в другой - управляемой. Процедуру, реализующую конкурс ЭМ системы на управление сетью связи ОВС и передачу статуса управляющей (передачу идентификатора управ-

ления) конкурирующим, требующим сети связи ЭМ, осуществляет процесс АРБИТР распределенного монитора операционной системы (рис.7).

Данная процедура построена таким образом, что управляющей может стать только одна ЭМ системы, даже если одновременно несколько ЭМ предложили себя в качестве управляющей. При арбитраже (выявлении новой управляющей ЭМ) процедура АРБИТР управляющей ЭМ исходит из относительных приоритетов каждой ЭМ системы.Приоритет ЭМ в рассматриваемой операционной системе представляет собой текущий размер очереди LAM-запросов в данной ЭМ. Таким образом, ЭМ, накопившая в очереди большее число запросов на сеть связи, считается самой приоритетной ЭМ на статус управляющей.

Переход на другое приоритетное правило выявления управляющей ЭМ достигается простым перепрограммированием части принятия решения процедуры АРБИТР. Если в данной разработке операционной системы управляющая ЭМ обслуживает в с ю свою очередь LAM-запросов, то возможна процедура обслуживания только ч а с т и очереди LAM-запросов, например, до некоторого среднего числа запросов по всем очередям LAM-запросов системы. Очевидно, возможны и другие варианты,описываемые в теории массового обслуживания.

Как только некоторой ЭМ передается статус управляющей, распределенный монитор операционной системы [9] данной ЭМ начинает обслуживание своей очереди LAM-запросов на обмен через сеть связи, реализуя протоколы обмена сообщениями между ЭМ системы. Сеть связи при этом считается закрытой для доступа управляемым ЭМ системы, которые, работая в мультипрограммном режиме, наполняют свои собственные очереди LAM-запросами от функциональных (прикладных) процессов, отвлекаясь только на осуществление дистанционных взаимодействий, требуемых со стороны управляющей ЭМ.

После обслуживания всех LAM-запросов из собственной очереди операционная система управляющей ЭМ отказывается от функций управляющей, реализуя процедуру арбитража распределенного монитора. Если после проведения арбитража выясняется, что претенденты на обмен через сеть связи ОВС отсутствуют (т.е. отсутствуют LAM-запросы во всей системе), управляющая ЭМ "открывает" сеть связи посылкой специального сообщения всем ЭМ системы. При этом распределенный монитор управляющей ЭМ ожидает (не расходуя процессорного времени) появления претендентов на сеть связи. Первый появившийся LAM-запрос в системе ЭМ "закрывает" сеть связи посылкой также специального сообщения всем ЭМ системы (в том числе и управляющей)

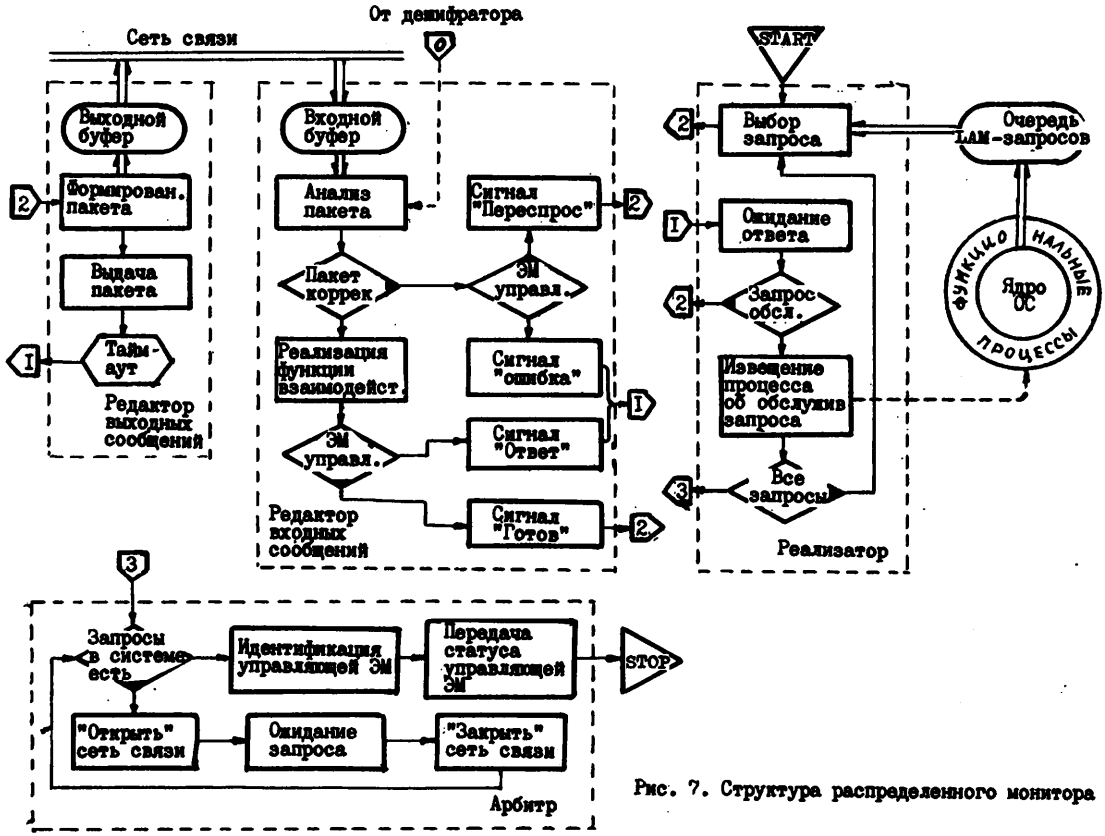


Рис. 7. Структура распределенного монитора ОС

и выводит распределенный монитор управляющей ЭМ на новый цикл арбитража для выявления появившихся претендентов.

Отметим, что адресация ЭМ на сети связи системы произвольная (необходимо только, чтобы ЭМ отличались по рабочим идентификаторам) и что управляющая ЭМ отвлекает на дистанционное взаимодействие только те ЭМ, адреса которых указаны в LAM-запросе. При этом управляющая ЭМ **не знает** фактического расположения вызываемых ЭМ на сети связи, а поиск ЭМ и осуществление передачи сообщений реализуется **автоматически** сетью связи.

На уровне распределенного монитора операционной системы сеть связи рассматривается как **сеть с коммутацией пакетов** [4]. При этом сообщения, определяемые LAM-запросом, преобразуются при обслуживании запроса распределенным монитором (но не самим функциональным процессом) в определенный формат, называемый пакетом. Пакет рассматривается как блок данных, который имеет смысл только на уровне распределенного монитора операционной системы. Блок имеет определенную форматную структуру, включающую информационную и управляющую части, формируемые модулем распределенного монитора РЕДАКТОРОМ ВЫХОДНЫХ СООБЩЕНИЙ.

Для осуществления корректного обмена между ЭМ системы заранее устанавливается максимальная длина пакета. Сообщения, длина которых больше установленной длины пакета, передаются распределенным монитором в виде нескольких пакетов. При этом принимающая и передающая ЭМ имеют входные и выходные буфера для приема и передачи пакетов соответственно.

Реализацию протоколов коммутации пакетов и регистрацию ошибок при обменах ЭМ через сеть связи осуществляют модули распределенного монитора операционной системы РЕАЛИЗАТОР и РЕДАКТОР ВХОДНЫХ/ВЫХОДНЫХ СООБЩЕНИЙ. Отметим, что несмотря на то, что в LAM-запросе функционального процесса можно задать множество вовлекаемых в обмен ЭМ-абонентов, операционная система для обеспечения надежности обмена обслуживает данный запрос через парные взаимодействия с ЭМ-абонентами в порядке, указанном в LAM-запросе. Эффективность использования парной схемы обмена между ЭМ системы и трансляционной схемы обмена в зависимости от длины пакета рассмотрена в работе [13].

Управляемая ЭМ, работая в мультипрограммном режиме, отвлекается сетью связи от обслуживания функциональных (прикладных) процессов только тогда, когда в ее входной буфер поступит пакет, пе-

реданный управляющей ЭМ. После приема пакета РЕДАКТОР ВХОДНЫХ СООБЩЕНИЙ управляемой ЭМ проверяет входной пакет на наличие ошибок, и, если ошибок не обнаружено, отрабатывается соответствующая функция взаимодействия, заданная в принятом пакете. Функция вызывающего процесса (породившего LAM-запрос) по взаимодействию с вызываемыми процессами, реализующимися в различных ЭМ системы, определяет:

а) над д а н н ы м и вызываемых процессов :

- чтение (READ) информации из определяемой в запросе области оперативной памяти указанных ЭМ-абонентов в определяемую также в запросе область памяти вызывающего процесса;

- запись (WRITE) информации из определяемой в запросе области памяти вызывающего процесса в оперативную память указанных ЭМ-абонентов, в которых реализуются вызываемые процессы;

б) над с о с т о я н и е м вызываемых процессов:

- извещение (POST) вызываемых процессов, реализующихся в указанных в запросе ЭМ-абонентах, о завершении определенного события вызывающим процессом;

в) по у п р а в л е н и ю:

- запуск (RUN) программных процессов в указанных ЭМ-абонентах с определяемого в запросе вызывающего процесса "вектора состояния".

Кроме этого, имеются две служебные функции взаимодействия по управлению сети связи, а именно - "открыть" и "закрыть" сеть связи.

Синхронизация вызывающего процесса с действиями операционной системы по обслуживанию LAM-запроса на взаимодействие осуществляется через таблицу управления событием (дистанционным взаимодействием), определяемой в LAM-запросе вызывающего процесса.

Таким образом, распределенный монитор осуществляет следующие функции:

- частично децентрализованное управление сетью связи;

- реализацию протоколов коммутации пакетов при обмене сообщениями между ЭМ системы через сеть связи;

- взаимодействие между процессами, реализующихся на разных ЭМ системы.

§3. Входной язык операционной системы

Входной язык операционной системы СУММА MACRO-8.0 представляет собой модификацию языка MACRO-8 [I4]. Модификация MACRO-8 С достигнута за счет определения некоторых макрокоманд (макросов) и

стандартных идентификаторов. Язык MACRO-8C позволяет создавать функциональные программы для вычислительной системы СУММА с учетом операционной системы (см. §2). Средства входного языка позволяют описывать произвольные схемы взаимодействий между ветвями параллельной программы, существующих для операционной системы в виде процессов. Взаимодействия процессов (ветвей) описываются на языке макрокомандами управления, синхронизации и коммуникации. Макроассемблер MACRO-8C используется для трансляции исходной программы, написанной на символическом языке в функциональную программу, готовую к реализации на вычислительной системе СУММА.

3.I. Системные макрокоманды для связи с ядром. Для связи функциональных (прикладных) процессов с ядром в языке MACRO-8 C определены макрокоманды по запуску (START) и остановке (STOP) процессов, по ожиданию (WAIT) и по извещению (POST) смежных процессов, а также макрокоманды ввода/вывода информации для связи процессов с драйверами ввода/вывода (READER/PUNCH) .

Макрокоманда запуска процесса на языке MACRO-8C описывается как

START AC, LINC, PC, PCD ,

где AC, LINC, PC,PCD представляют "вектор состояния" запускаемого процесса (см. рис.3).

Остановка процесса осуществляется по макрокоманде STOP , которая останавливает текущий процесс, не останавливая процессора ЭМ.

Макрокоманды синхронизации процессов в пределах данного ядра операционной системы описываются на входном языке как

WAIT FD, DECB
POST, FD, DECB, ICD,

где FD и DECB - поле данных и идентификатор таблицы управления (см. рис.4) определенного события; DECB, ∅; ∅; ∅ ;
ICD - идентификатор задаваемого кода извещения.

Макрокоманды функциональных процессов на ввод READER и вывод PUNCH символьной информации описываются на языке MACRO-8C следующим образом:

READER FD, BUF, COD, FD1, DECB
PUNCH FD, BUF, COD, FD1, DECB ,

где FD и BUF - поле данных и идентификатор приемного/передаваемого буфера символьной информации соответственно;

COD - код последнего принимаемого/передаваемого символа;

FDI,DECB - поле данных и идентификатор таблицы управления событием (вводом/выводом) для синхронизации процесса с драйверами ввода/вывода.

3.2. Описание LAM- запроса. Общая форма LAM-запроса на взаимодействие функционального процесса с вызываемыми процессами,реализующимися в различных ЭМ системы,на входном языке MACRO-8C выглядит следующим образом:

LAM FUNC, FD, PARAM, ABON, DECB .

Операнд FUNC указывает функцию взаимодействия, которая определяется стандартными идентификаторами READ (читать), WRITE (писать), POSTF (известить) и PUSK (пустить).

Операнд FD определяет идентификатор поля данных памяти,в которой определяются все остальные операнды,относящиеся к данному LAM-запросу. Формат FD: $\emptyset\ \emptyset$ D $\emptyset$, где $1 \le D \le 7$.

Операнд PARAM специфицирует идентификатор списка исходных параметров задаваемой в запросе функции взаимодействия. Структура списка исходных параметров конкретизируется ниже.

Операнд ABON определяет список рабочих идентификаторов ЭМ, вовлекаемых во взаимодействие данным запросом. Формат списка:

ABON, N; ID1; ID2;... ; IDJ; ...; IDN ,

где N - мощность списка (число рабочих идентификаторов в списке), $1 \le N \le 11$, IDJ - рабочий идентификатор J-й ЭМ-абонента.

Заметим,что LAM-запрос обслуживается в порядке перечисления ЭМ в списке ABON и что операционная система отмечает в списке ABON рабочие идентификаторы тех ЭМ, взаимодействие с которыми не было осуществлено (взводится нулевой бит IDJ). Функциональный процесс может построить дальнейшую логику своего поведения, исходя из этой информации.

Операнд LAM-запроса DECB специфицирует адрес таблицы управления событием (дистанционным взаимодействием). На таблице управления DECB происходит синхронизация функционального процесса, выдавшего LAM-запрос, с действиями распределенного монитора , осуществляющего обслуживание запроса [5].

После регистрации LAM-запроса в очереди функциональный процесс, вообще говоря, продолжается дальше. Для того чтобы узнать

об обслуживании данного LAM-запроса, процесс должен реализовать макрокоманду WAIT с той же таблицей управления,что и в соответствующем LAM-запросе. При этом после разблокировки,анализируя код извещения таблицы управления взаимодействием, функциональный процесс узнает о том, как был обслужен системой данный запрос.Код извещения в таблице управления дистационным взаимодействием может быть задан операционной системой следующим образом:

0 - данный LAM-запрос полностью обслужен;
4 - переполнение входного буфера;
5 - несовпадение контрольной суммы;
6 - неправильно задана функция взаимодействия;
7 - несостоятельность функции взаимодействия;
I2 - коней таймаута.

При этом коды извещения 4,5,6,7,I2 имеют отношение только к операционной системе и для функционального процесса означают неудачу при обслуживании запроса. ЭМ, взаимодействия с которыми не произошло, отмечаются в списке ABON.

Если в LAM-запросе указывается функция чтения данных (READ), то исходные параметры, заданные списком PARAM , должны быть описаны в следующем формате:

PARAM, FD1; ADRIN; DL1; FD2; ADROT; DL2 ,

где FD1,ADRIN - поле данных и идентификатор приемного буфера в данной ЭМ; DL1- длина приемного буфера; FD2, ADROT - поле данных и имя буфера данных читаемого в ЭМ-абонентах; DL2 - длина читаемого буфера.

Для функции чтения должно выполняться условие $DL1 \geq DL2 \times N$, где N - мощность списка ABON.

Если задается функция записи данных (WRITE) , то исходные параметры должны быть описаны в аналогичном формате с функцией чтения с учетом того, что

ADRIN - идентификатор буфера приема данных в ЭМ-абонентах,рабочие идентификаторы которых определены списком ABON исходного запроса; ADROT - идентификатор передаваемого буфера информации из данной ЭМ.

Для функции записи должно выполняться условие $DL1 \geq DL2$.

В том случае, когда определяется LAM-запрос на извещение процессов (POST),исходные параметры задаются списком

PARAM, COD: FD: DECB ,

где COD - задаваемый код извещения, $0 \le COD \le 1777_8$; FD, DECB - поле данных и идентификатор таблицы управления событием, определенные на множестве вовлекаемых во взаимодействие процессов, активных в различных ЭМ-абонентах.

Если в LAM-запросе задается функция запуска процесса (PUSK), то список параметров определяет "вектор состояния" запускаемых процессов:

PARAM, AC; LINC; PC; FCD ,

где AC, LINC, PC, FCD - "вектор состояния" запускаемых процессов в ЭМ, рабочие идентификаторы которых указаны в списке ABON."Вектор состояния" задается аналогично "вектору состояния" в макрокоманде запуска процесса (START) ядра операционной системы.

3.3. Стандартные идентификаторы. При модификации макроассемблера MACRO-8C команды обращения к системному устройству [3] определены как стандартные идентификаторы. Содержательное значение и мнемоника команд приведены в таблице.

Таблица

Код команды	Мнемокод	Команда
6I6I	SSF	пропуск, если флаг СУ = I
6I62	CLD	очистка приоритета ЭВМ
6I63	DAI	занесение в регистр признаков
6I64	STP	установка приоритета ЭВМ
6I65	DAA	занесение в регистр адреса СУ
6I66	RDB	требование разрыва данными
6I67	DAJ	занесение в регистр настройки
6I72	SCF	очистка флага СУ и регистра настройки
6I73	LAJ	считывание регистра настройки
6I75	LAA	считывание регистра адреса СУ
6I76	LAB	считывание буферного регистра
6I77	LAI	считывание регистра признаков

Вместе с этим были определены стандартные аргументы макроса LAM-запроса: READ = 1, WRITE = 2, POST = 3, PUSK = 4 .

Заключение

Составными частями операционной системы СУММА являются ядро, организующее происхождение процессов на уровне ЭМ, и распределенный монитор, осуществляющий взаимодействие процессов на уровне системы ЭМ. Предложенные способ организации функционирования и структура операционной системы могут использоваться при построении многомашинных вычислительных систем и сетей при их ориентации на решение задач распределенной обработки информации, регулирования и управления в реальном времени.

Автор выражает благодарность И.Н.Кашуну и Л.А.Седухиной за помощь в реализации операционной системы на ОВС СУММА.

Литература

I. ЕВРЕИНОВ Э.В., ХОРОШЕВСКИЙ В.Г. Однородные вычислительные системы.-Новосибирск: Наука Сиб.отд-ние, 1979. -319 с.

2. Вычислительная система СУММА/ Афанасьев В.П., Еремин А.П., Желтов М.П., Ильин М.Н., Седухин С.Г., Томилов Ю.Ф., Хорошевский В.Г., Шум Л.С. -В кн.: Вычислительные системы. Вып.60. Вопросы теории и проектирования вычислительных систем. Новосибирск, 1974, с. 153-169.

3. Системное устройство однородной вычислительной системы СУММА/ Афанасьев В.П., Ильин М.Н., Седухин С.Г., Шум Л.С. -В кн.: Однородные вычислительные системы и среды. Материалы IУ Всесоюзной конференции. Киев, Наукова думка, 1975, с.47-48.

4. ДЭВИС Д., БАРБЕР Д. Сети связи для вычислительных машин.-М.: Мир, 1976. -680 с.

5. СЕДУХИН С.Г., ЖЕЛТОВ М.П., КАШУН И.Н. Ядро операционной системы СУММА.-В кн.: Вычислительные системы. Вып. 70.Вопросы теории и проектирования вычислительных систем. Новосибирск, 1977, с. 113-129.

6. HANSEN P.B. The Nucleus of a Multiprogramming System. - Comm.ACM,1970,v.13, N 4,p.238-240,250.

7. Мультипроцессорные системы и параллельные вычисления. Под ред. Ф.Г.Энслоу, пер. с анг., М., Мир, 1979. -383 с.

8. ДИЙКСТРА Э. Взаимодействие последовательных процессов. -В кн.: Языки программирования. М., 1972, с.9-87.

9. СЕДУХИН С.Г. Организация взаимодействий в однородной вычислительной системе СУММА. -В кн.: Однородные вычислительные системы и среды. Материалы IУ Всесоюзной конференции, Киев, 1975, с. 63-64.

10. ЦИКРИТЗИС Д., БЕРНСТЕЙН Ф. Операционные системы. Пер. с англ. М.: Мир, 1977, 335 с.

11. DIJKSTRA E.W. The Structure of the THE Multiprogramming System.-Comm.ACM,1968,v.11, N 5,p.341-347.

I2. ВОРОБЬЕВ В.А., СЕДУХИН С.Г., КАШУН И.Н. Исследование децентрализованных дисциплин взаимодействий между элементарными машинами однородной вычислительной системы.-В кн.: Однородные вычислительные системы и среды.Материалы IУ Всесоюзной конференции, Киев, I975, с.36-38.

I3. СЕДУХИН С.Г. К анализу режимов передачи данных в вычислительной системе СУММА.-Настоящий сборник, с. 8I-84.

I4. PROGRAMMING Languages. PDP-8 Family Computer, Dec.,1970.

Поступила в ред.-изд.отд.
29 июня I979 года