МАТЕМАТИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ
(Вычислительные системы)
1979 год Выпуск 78

УДК 681.142.4

ОПЕРАЦИОННАЯ СИСТЕМА МИНИМАКС

В.Д.Корнеев

Введение

Вычислительная система МИНИМАКС представляет собой модульный многопроцессорный комплекс с программно перестраиваемыми связями между модулями. Основу модулей, называемых элементарными машинами, составляют мини-ЭВМ М-6000. Объединение элементарных машин в единую многопроцессорную вычислительную установку осуществляется с помощью специального системного коммутатора, представляющего собой двухмерную решетку, в каждом узле которой находится элементарный модуль межмашинной связи. Последний имеет непосредственную связь с одной мини-ЭВМ и с четырьмя соседними элементарными модулями коммутатора. Взаимодействие между элементарными машинами и разбиение системы на автономно работающие подсистемы обеспечиваются программированием соответствующих путей и непересекающихся графов в двухмерной решетке системного коммутатора.

Минимальное число элементарных машин в системе - 2, максимальное - 64.

Каждый процессор имеет доступ к любой ячейке памяти всей системы, однако "механизм" доступа к памяти других элементарных машин (ЭМ) отличается от "механизма" доступа к памяти своей машины, т.е. оперативная память всей системы по отношению к каждому процессору является иерархической. Внешние устройства системы подключаются через стандартные средства сопряжения ЭМ.

Важным свойством системы МИНИМАКС является стандартизированность (однородность) ее аппаратурных модулей. Однородность опре-

деляется относительно свойств, присущих всем ЭМ. Ее определяют одинаковый интерфейс между ЭМ и единый формат команд.

Вместо выражения "система МИНИМАКС" будем иногда употреблять просто МИНИМАКС или система, а вместо - операционная система для МИНИМАКСа - операционная система МИНИМАКС.

Операционная система МИНИМАКС представляет собой совокупность программных модулей (блоков), реализующих интерфейс между пользователями или их программами и ресурсами всей системы. Каждый блок является супервизором, выполняющим задания определенного типа и управляющим выделенными ему ресурсами. Модули распределены по двум уровням иерархии принятия решений и доступа к обобщенным ресурсам и физически рассредоточены по элементарным машинам системы. "Погруженные" в однородную аппаратурную часть эти модули образуют совокупность функционально неоднородных программно-аппаратурных модулей системы МИНИМАКС.

Операционная система обеспечивает следующие основные режимы функционирования.

Режим параллельной обработки, при котором все ресурсы системы монопольно используются для решения одной сложной задачи, представленной параллельной программой. При этом возможны как централизованные, так и децентрализованные схемы взаимодействий между процессами, происходящими в разных ветвях параллельной программы. Этот режим представляет определенные возможности для организации высоконадежных вычислений, связанных с продолжением счета при выходе из строя отдельных ЭМ системы.

Режим автономной работы, при котором в системе выполняются последовательные программы с требованиями на ресурсы, не превышающие возможностей одной ЭМ.

Режим смешанного функционирования, при котором на одной подсистеме выполняется режим параллельной обработки, а на другой - режим автономной работы.

Операционная система МИНИМАКС использует основную управляющую систему и супервизор реального времени АСВТ-М в качестве своих составных частей.

Для реализованной в данный момент версии операционной системы МИНИМАКС [I] рекомендуются следующие области применений:

- автоматизированные системы управления технологическими процессами;

- информационно-справочные центры;

- системы реального времени с фиксированным циклом регистрации данных и выдачи управляющих сигналов;

- специализированные вычислительные центры коллективного пользования.

§I. Структура и функциональные возможности

I.I. О с о б е н н о с т и с т р у к т у р ы. Основной структурной единицей ресурса системы МИНИМАКС является элементарная машина (ЭМ), состоящая из ЭВМ и модуля межмашинной связи. В качестве ЭВМ используется вычислительный комплекс, организованный на базе АСВТ-М и включающий процессор М-6000.

Взаимодействие между ЭМ осуществляется по двухмерным связям I и одномерным связям 2 (замкнутым в кольцо), а между ЭВМ и модулем межмашинной связи - по связям 3 (рис.I). Взаимодействия по связям I и 2 осуществляются по правилам, установленным командами модуля межмашинной связи [2], а по связям 3 - по правилам, принятым для работы ЭВМ с внешними устройствами.

Различаются взаимодействия двух типов: управление и обмен, процесс выполнения которых разбивается на три этапа. На первом этапе формируется область связности между участвующими во взаимодействии ЭМ, на втором - выполняется взаимодействие, на третьем - производятся действия, разрешающие использовать ЭМ в других взаимодействиях.

М о д у л ь м е ж м а ш и н н ы х с в я з е й состоит из следующих основных блоков: настройки, обмена и местного управления.

Блок настройки расшифровывает в 3-разрядном регистре настройки | СФI | СФ2 | ω | код, указывающий, а) готова (согласна) ли данная ЭМ участвовать во взаимодействии по связям I (разряд ω = I - да, ω =0 - нет) и б) по какому из четырех направлений осуществляется прием данных (направления задаются двумя разрядами СФI и СФ2).

Блок обмена: а) принимает информацию по связям I с направления, определяемого значениями СФI и СФ2; б) передает информацию из ЭМ по этим связям во все четыре направления.

В системе может быть образовано несколько подсистем, работающих независимо. В каждый момент времени в подсистеме может быть

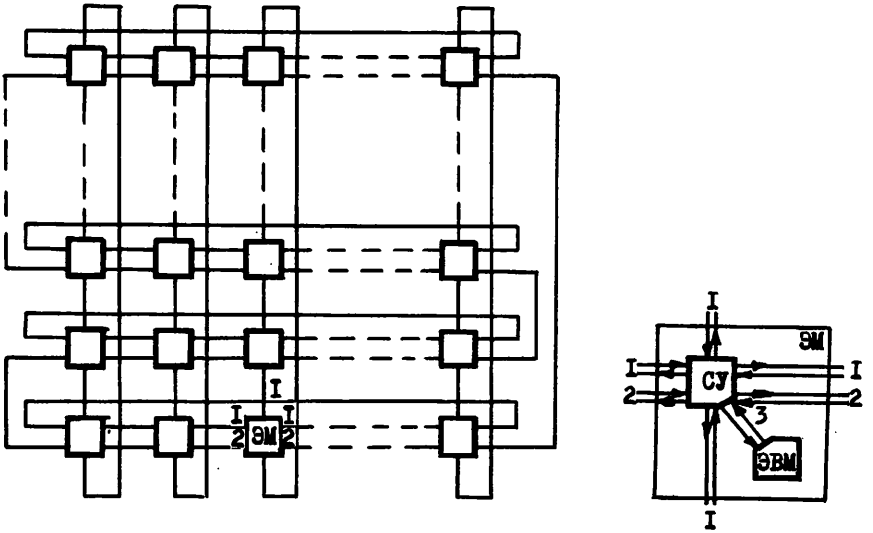


Рис. I

только одна передающая ЭМ, все остальные либо принимающие, либо транзитные. В транзитных ЭМ $\omega = 0$ и в передаче данных участвуют только модули межмашинных связей. Процессоры этих ЭМ не отвлекаются от их текущей работы.

Одной из главных функций блока местного управления является контроль по четности передаваемой информации.

I.2. Я з ы к с и с т е м ы представляет собой расширение МНЕМОКОДа М-6000 специальными системными командами, позволяющими задавать взаимодействия между процессами, выполняющимися в разных ЭМ системы. Системные команды реализуются блоками модуля межмашинной связи и программным модулем (драйвером), входящим в состав операционной системы. Это расширение является в МИНИМАКСе системным языком первого уровня.

Язык позволяет задавать:

- передачу информации между ЭМ, которая выполняется по сигналам запрос-ответ, что обеспечивает обмен информацией при любых рассогласованиях скоростей работы ЭМ;

- синхронизацию, выравнивание во времени работы ЭМ;

- выполнение обобщенной условной и безусловной передач управления в нескольких машинах подсистемы;

- изменение топологии связей и степени участия ЭМ во взаимодействиях;

- выставление в модуле межмашинной связи специального признака "запрос связи" и проверку наличия этого признака из любой другой ЭМ подсистемы;

- приведение межмашинной связи в исходное состояние.

Алгоритмы, реализующие системные команды, записаны на МНЕМОКОДе и оформлены в виде девяти подпрограмм.

I) Команда НАСТРОЙКА СИСТЕМЫ. Вызов подпрограммы, реализующей команду, имеет вид:

```
JSB WKN
DEF AMH
OCT ⟨DMH⟩,
```

где WKN - имя (входная метка) подпрограммы (в записях всех вызовов подпрограмм, реализующих системные команды, идентификатор в первом слове является именем соответствующей подпрограммы); AMH - метка начального адреса массива настройки; ⟨DMH⟩ - длина массива настройки, т.е. количество кодов настройки в массиве. Код настройки - четырехразрядное слово | СФI | СФ2 | ω | ϕ |.

Команда осуществляет изменение регистров настройки в любом заданном множестве модулей межмашинной связи. Участие модуля межмашинной связи в настройке определяется признаком ϕ в коде настройки, первые три разряда которого записываются в регистр настройки данного модуля.

Настройка системных каналов выполняется по одномерным связям 2 последовательной выдачей в канал кодов настройки. В каждый момент в подсистеме может быть только одна ЭМ, выполняющая команду НАСТРОЙКА СИСТЕМЫ.

2) Команда НАСТРОЙКА МОДУЛЯ МЕЖМАШИННОЙ СВЯЗИ. Вызов соответствующей подпрограммы имеет вид:

```
JSB WSN
OCT ⟨KIH⟩,
```

где ⟨KIH⟩ - код настройки, записываемый в регистр настройки собственного модуля межмашинной связи. Команда осуществляет изменение регистра настройки и одновременно может выполняться в любом множестве элементарных машин подсистемы.

3) Команда ЗАПРОС СВЯЗИ. Вызов подпрограммы:

```
{ JSB ZSW      или  { JSB ZSW
{ DEF  AV           { OCT  0 ,
```

где AV - метка фоновой программы; 0 - специальный признак.

Команда устанавливает регистр R_z в своем модуле межмашинной связи в состояние "запрос", который воспринимается блоками операционной системы (см. п.2.4), ведающими в подсистеме централизованным распределением системных каналов.

При выполнении команды реализуются два типа ожидания ответа от блоков операционной системы на поставленный запрос: динамический останов процессора - указывается признаком нуль; переход (ответвление) на выполнение фоновой программы - указывается входной меткой AV.

4) Команда ПРОВЕРКА СВЯЗИ. Вызов подпрограммы:

```
JSB PSW
OCT ⟨НЕМ⟩
JMP  M ,
```

где ⟨НЕМ⟩ - порядковый в подсистеме номер ЭМ, у которой проверяется содержимое регистра запроса R_z; M - входная метка программы, которой передается управление, если R_z находится в состоянии "запрос", иначе управление получает команда,следующая за данным вызовом подпрограммы.

5) Команда СИНХРОНИЗАЦИЯ. Вызов подпрограммы:

```
{JSB SINX      или  {JSB SIN X
{DEF AV1            {OCT  0 ,
```

где AV1 - метка фоновой программы; 0 - специальный признак.

Команда реализует групповую синхронизацию (выравнивание во времени) процессов, происходящих в разных ЭМ подсистемы. Синхронизация заключается в одновременном прерывании работы процессов в элементарных машинах подсистемы, участвующих в синхронизации, и передаче в каждой ЭМ управления на команду, следующую за вызовом подпрограммы. Для участия в синхронизации необходимо в регистр настройки - разряд ω - и в регистр синхронизации R_c выставить единицы.

При выполнении команды реализуются два типа ожидания синхронизации процессов: динамический останов процессора - указывается

признаком 0 ; ответвление на фоновую программу - указывается входной меткой AV1 .

6) Команды ПРИЕМ, ПЕРЕДАЧА, ФИКТИВНЫЙ ПРИЕМ реализуются подпрограммой, имеющей одну общую входную метку. Вызов подпрограммы:

```
JSB FPPP
OCT ⟨PR⟩
DEF ADR
OCT ⟨DM⟩,
```

где ⟨PR⟩ - код типа команды: при ⟨PR⟩ = 0 - фиктивный прием, ⟨PR⟩ = 1 - прием, ⟨PR⟩ = 2 - передача; ADR - метка начального адреса принимаемого или передаваемого массива в зависимости от типа команды, при фиктивном приеме этот параметр может быть произвольным, так как ЭМ принимаемую информацию в память не записывает; ⟨DM⟩ - длина обмениваемого массива.

Команда осуществляет групповой обмен данными между программами, реализующимися на подсистеме. В каждый момент времени в подсистеме только одна ЭМ может выполнять передачу данных; прием данных может одновременно выполняться либо всеми ЭМ, либо выделенным их подмножеством (но не менее одной), участвующим в обмене. Принимающие ЭМ одновременно получают одну и ту же информацию от передающей.

Фиктивный прием данных используется элементарными машинами для определения момента окончания обмена данными, выполняемого другими машинами подсистемы.

7) Команда ОБОБЩЕННЫЙ БЕЗУСЛОВНЫЙ ПЕРЕХОД (ОБП). Вызов подпрограммы:

```
JSB EXEK
OCT ⟨KY⟩,
```

где ⟨KY⟩ - код управляющего слова, ⟨KY⟩ = i (i=1,2,3,4,5) соответствует разным типам обобщенного безусловного перехода.

Команда реализует одновременное прерывание в подсистеме процессоров всех ЭМ с $\omega = 1$ и передачу этим ЭМ управляющего слова, по которому они определяют тип выполняемой работы.

8) Команда ОБОБЩЕННЫЙ УСЛОВНЫЙ ПЕРЕХОД (ОУП). Вызов подпрограммы:

```
JSB OUT
JMP M1 ,
```

где М1 - метка, на которую передается управление при выполнении обобщенного условия, характеризуемого значением обобщенного признака Ω. Значение Ω равно конъюнкции признаков f_y от всех элементарных машин подсистемы, участвующих в обобщенном условном переходе. В каждой ЭМ перед выполнением команды ОУП признак f_y записывается на регистр В .

Обобщенный условный переход заключается в одновременной передаче управления во всех машинах на метку М1 или на команду,следующую за вызовом подпрограммы, если обобщенное условие не выполняется. Программы, участвующие в обобщенном условном переходе,перед выполнением команды ОУП синхронизируются. Во время ожидания синхронизации процессоры элементарных машин находятся в динамическом останове.

9) Команда СБРОС РЕГИСТРОВ В МОДУЛЕ МЕЖМАШИННОЙ СВЯЗИ. Вызов подпрограммы:

JSB SBR .

Команда осуществляет приведение модуля межмашинной связи в исходное состояние. Используется в основном при реализации других системных команд и после сбойных ситуаций.

Таким образом, системный язык первого уровня, реализованный программно-аппаратурным способом, имеет необходимые средства для запуска, прерывания и синхронизации процессов,происходящих в разных машинах системы. Главные особенности этого языка:

- наличие средств управления структурой связи между элемен - тарными машинами и активностью машин в системных взаимодействиях;
- эффективная аппаратурная поддержка команд одновременного (группового) доступа многих процессов к одной и той же информации;
- наличие команды СИНХРОНИЗАЦИЯ, обеспечивающей выравнивание во времени большой группы процессов, реализуемых в разных ЭМ системы.

§2. Операционная система

2.I. О б щ а я х а р а к т е р и с т и к а. Операционная система МИНИМАКС представляет собой совокупность асинхронно-взаимодействующих блоков (модулей), распределенных по двум уровням организационной иерархии. Совокупность блоков нижнего уровня представляет собой внутренний концентр, над которым надстроены блоки

второго уровня. Последние, в свою очередь, являются базой для следующего уровня иерархии. Каждый модуль операционной системы является специализированным супервизором с многоуровневой структурой. Такой подход позволяет произвести определенную классификацию вычислительных процессов, происходящих в системе, и путем учета особенностей процессов разных типов повысить эффективность их взаимодействий.

Блоки нижнего уровня называются диспетчерами элементарных машин. Они рассредоточены по всем машинам, и в оперативной памяти каждой ЭМ имеется один диспетчер элементарной машины. Блоки верхнего уровня называются старшими диспетчерами; они располагаются в оперативной памяти выделенных машин, называемых диспетчерскими. Количество старших диспетчеров в системе равно количеству автономно работающих подсистем. Каждый диспетчер ЭМ и старший диспечтер реализуются средствами одной ЭМ и имеют стандартизированную структуру, представляющую собой пару последовательно выполняемых подблоков, первый из которых распознает мультипрограммную ситуацию, в частности, контролирует правильность запуска блока, второй - реализует алгоритмы управления.

Диспетчер ЭМ включает в себя подпрограммы, реализующие системные команды (см. п.I.2), обработчик системных прерываний, обработчик аварийных ситуаций, библиотеку модулей системных операторов и основную управляющую систему АСВТ-М. Старший диспетчер включает в себя ядро, супервизор реального времени АСВТ-М и системный загрузчик.

Каждый диспетчер ЭМ управляет прохождением задач или их ветвей на одной машине и обеспечивает взаимодействие процессов, реализующихся в разных ЭМ, а именно:

- инициирует процессы в старшем диспетчере;
- обрабатывает прерывания процессора;
- осуществляет групповые взаимодействия между процессами;
- управляет вводом/выводом для индивидуальных внешних устройств.

Старший диспетчер управляет прохождением задач на всей подсистеме:

- проверяет запросы от диспетчеров ЭМ;
- управляет каналами межмашинных связей;
- управляет аппаратом событий для индивидуальных взаимодействий процессов;

- инициирует автоматический запуск задач в соответствии с их временными характеристиками (начальной задержкой и периодом между двумя последовательными выполнениями);

- обрабатывает различные особые ситуации, которые могут возникнуть в процессе работы системы;

- обеспечивает централизованную загрузку параллельных программ в систему.

В ОВС имеется развитая система синхронизации процессов, обеспечивающая как индивидуальные, так и групповые схемы взаимодействий.

Для обеспечения взаимодействий процессов имеется эффективная система управления структурой связи между ЭМ и их активностью в системных взаимодействиях.

Для обеспечения устойчивого (надежного) функционирования системы имеются специальные средства, позволяющие выявлять глобальные тупиковые ситуации при взаимодействии процессов и выполнять перезапуск вычислений со специальных точек возврата при сбоях аппаратуры.

Операционная система МИНИМАКС обеспечивает автоматическое управление большим количеством внешних объектов и позволяет организовать высокопроизводительные, высоконадежные и экономичные вычисления.

Высокопроизводительные вычисления могут достигаться параллельным выполнением на различных ЭМ разных задач или разных частей одной задачи.

Высоконадежные вычисления могут обеспечиваться:

- продолжением вычислений при выходе ЭМ из строя на подсистемах меньших размеров;

- дублированием вычислений на разных подсистемах;

- использованием специальных программных средств для осуществления повторных запусков программы с контрольных точек.

Экономичные вычисления могут достигаться:

- подбором состава оборудования и блоков операционной системы;

- специализацией отдельных подсистем на определенные виды работ;

- организацией коллективного доступа к системе путем одновременного доступа к разным элементарным машинам;

- постепенным наращиванием количества ЭМ в системе.

2.2. Процессы и их взаимодействия. При функционировании системы выделяются три типа процессов, связанных соответственно с реализацией I) обрабатывающих пользовательских программ; 2) блоков диспетчеров ЭМ и 3) блоков старшего диспетчера.

Процессы первого типа (обрабатывающих программ) могут запускать только процессы второго типа (диспетчеров ЭМ); любые процессы первого типа могут взаимодействовать между собой только через инициацию процессов либо второго, либо второго и третьего типов.

Процессы второго типа могут запускать процессы второго и третьего типов; взаимодействовать между собой они могут как непосредственно, так и через процессы третьего типа.

Процессы третьего типа (старшего диспетчера) могут запускать процессы только второго типа и взаимодействовать между собой путем передачи управления друг другу.

Процессы старшего диспетчера являются самыми приоритетными, они могут прервать любой процесс. Процессы диспетчеров ЭМ являются более приоритетными по отношению к процессам обрабатываемых программ.

Процесс с большим приоритетом запускает процесс с меньшим приоритетом без ожиданий (активно), прерывая, если нужно, другие процессы. Процесс с меньшим приоритетом запускает процесс с большим приоритетом путем запроса, т.е. пассивно.

Процессы первого типа делятся на основные и вспомогательные. Основные процессы порождаются (оформляются) в системе в реаультате выполнения этапа генерации программного обеспечения, вспомогательные порождаются процессами диспетчеров ЭМ по запросам основных процессов. Основной и его вспомогательный процессы связаны с одной и той же обрабатывающей программой.

Основной процесс может находиться в трех состояниях: пассивном (П-состояния) и двух активных (А-состояниях): A-REAL и A-WAITE . Заметим, что П-состояние существует с момента порождения до момента первой инициации обрабатывающей программы, т.е. до момента перехода в состояние A-REAL ; А-состояние существует в течение всего времени режима автоматический счет. Процесс находится в состоянии A-REAL ,если выполняются команды обрабатывающей программы и при этом нет события, которого ждет процесс. Процесс находится в состоянии

A-WAITE , если он ожидает некоторого события и счетчик адреса команд в его векторе состояния не изменяется. Из A-REAL процесс может перейти только в A-WAITE , и наоборот. Исключением является момент завершения режима автоматический счет, - когда все процессы переходят в П-состояние.

Вспомогательный процесс может находиться только в состояниях A-REAL и A-WAITE . Порождается такой процесс в тот момент, когда основной процесс находится в состоянии A-WAITE , уничтожается в результате определенных прерываний или передачи управления процессу "динамический останов". В каждый момент с основным процессом может быть связан только один вспомогательный. Переход из A-REAL в A-WAITE происходит в результате прерываний от индивидуальных внешних устройств, а из A-WAITE в A-REAL - после завершения обработки таких прерываний основной управляющей системой АСВТ-М.

С каждым основным процессом связываются: его имя, системные адреса запуска (т.е. относительные номера ЭМ и адреса в памяти этих ЭМ), адреса точек возврата, значение вектора состояния вспомогательного процесса в момент его уничтожения или перехода в состояние A-WAITE , значения регистров А, В, Е, РП, а также поля синхронизации (см. п.2.4).

В вектор состояния вспомогательного процесса входят адреса запуска (уничтожения) и стандартные регистры А, В, Е, РП мини-ЭВМ М-6000.

Вспомогательные процессы организуются для совмещения с ч е т а по обрабатывающей программе с о ж и д а н и е м - основным процессом определенных событий в системе.

Процессы второго типа существуют в течение всего режима "автоматический счет" и могут иметь максимум 4 разных состояния: П-состояние, A-REAL ,A-WAITE и а в а р и й н о е. Первые три аналогичны вышерассмотренным. Можно отметить особенность состояния A-REAL , состоящую в том, что процесс переходит в это состояние путем передачи управления на стандартную входную метку. Исключение могут составлять процессы обработки прерываний индивидуальных внешних устройств, если основная управляющая система АСВТ-М разрешает новое прерывание до завершения обработки предыдущего. Аварийным называется состояние, в котором оказывается процесс второго типа после того, как он не смог (после трехкратной попытки) выполнить межмашинное взаимодействие, запустить внешнее устройство, дождаться за отведенное время определенного события,а также в ре-

зультате сбоя процессора во время выполнения процесса. Переход в аварийное состояние сопровождается остановкой процессоров и выдачей сообщения оператору.

С процессами второго типа связываются его имя (входная метка) и адреса точек возврата, обеспечивающие оператору возможность перезапуска системы и перевода процессов из аварийного состояния в одно из А-состояний.

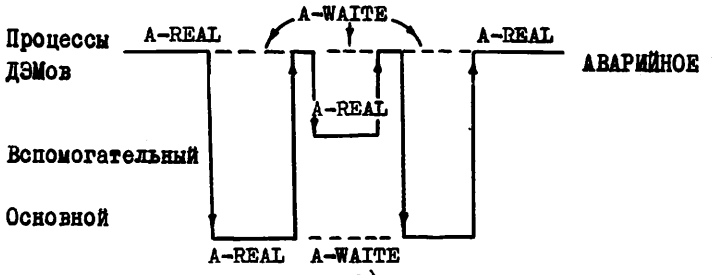
Состояния процессов третьего типа аналогичны состояниям процессов второго типа. Однако переход из A-REAL в A-WAITE возможен только по инициативе самого процесса, а переход из A-WAITE в A-REAL - по инициативе процессов второго типа, выполняющихся в той же ЭМ, где и процессы третьего типа. Основными мультипрограммными ситуациями, которые приводят ко второму из указанных переходов, являются обработка прерываний от таймера и попытка выполнить оператор индивидуальных взаимодействий (см. п.2.3) в программе, находящейся в той же ЭМ, где и старший диспетчер.

Типы процессов и особенности их взаимодействий иллюстрируются на рис. 2.

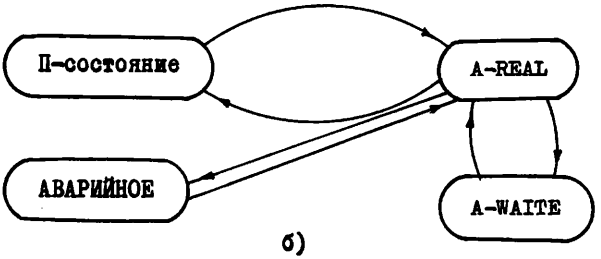
Имеются два уровня языковых средств для задания взаимодействий между процессами: системный язык первого уровня, описанный в п.I.2, и системные языки второго уровня, представляющие собой МНЕМОКОД, ФОРТРАН, АЛГОЛ, расширенные набором системных операторов. Подпрограммы, реализующие эти операторы, объединены в библиотеку диспетчеров элементарных машин. Системный язык первого уровня рассчитан в основном на разработчиков системного математического обеспечения, а языки второго уровня - на разработчиков обрабатывающих программ.

2.3. С и с т е м н ы е о п е р а т о р ы [3] расширяют языки АЛГОЛ, ФОРТРАН, МНЕМОКОД до языков, предназначающихся для записи параллельных алгоритмов. Они так же, как и системные команды, обеспечивают обмен информацией между процессами, синхронизацию процессов по определенным событиям, прерывание одних процессов и инициацию других, но в отличие от системных команд не требуют знаний особенностей аппаратуры.

Среди системных операторов выделены два функционально полных набора, отличающихся числом процессоров, одновременно участвующих во взаимодействии, задаваемом этим оператором, а также используемым аппаратом событий. Системные операторы первого набора называ-



а)



б)

в)

Рис.2. Типы процессов и особенности их взаимодействий: а) взаимодействие процессов обрабатывающих программ и диспетчеров ЭМ; б) диаграмма переходов между состояниями процесса; в) иерархия процессов.

ются операторами индивидуальных взаимодействий; с помощью одного такого оператора можно запустить только один процесс и передать информацию только одному процессу. Системные операторы второго набора называются операторами групповых взаимодействий; с помощью одного такого оператора можно запустить группу процессов в разных ЭМ и передать информацию группе процессов.

В основе аппарата событий второго набора лежат системные команды СИНХРОНИЗАЦИЯ и ОБОБЩЕННЫЙ УСЛОВНЫЙ ПЕРЕХОД, позволяющие определять момент выхода на разные копии одного и того же системного оператора группы процессов.

В реализации операторов первого набора принимают участие диспетчеры ЭМ и старший диспетчер подсистемы; в реализации операторов второго набора - только диспетчеры ЭМ.

Системные операторы индивидуальных взаимодействий [4]. Процесс, вышедший на такой оператор, устанавливает регистр R_z в своем модуле межмашинной связи в состояние "запрос". После обнаружения этого запроса старшим диспетчером последнему передается запрос на взаимодействие, состоящий из массива необходимых параметров, и после получения разрешения от старшего диспетчера выполняется взаимодействие, задаваемое системным оператором.

ПИСАТЬ - IW(Individual Writing) - предназначен для передачи данных процессу, реализующемуся в другой ЭМ. Вызывающая последовательность на языке ФОРТРАН:

```
CALL IW(I,A1, A2,N1,A,N,B,P)
GOTO  M ,
```

где IW - имя подпрограммы, реализующей системный оператор; I - имя набора семафорных переменных; A1 - параметр, выделяющий подмножество семафоров, по которым проверяется условие синхронизации; A2 - параметр, определяющий упорядоченную последовательность степеней P и V операций (см. п.2.4.); N1 - номер ветви, которой пересылается массив; A - имя пересылаемого массива; N - размер массива; B - адрес в ветви N1 , в которую пересылается массив A; P - признак ответвления на вспомогательный процесс (при P=1 ответвление есть, при P=0 - нет); M - метка ответвления на вспомогательный процесс.

Оператор GOTO служит для возможного совмещения вычислений с ожиданием, если возникают задержки при выполнении системного взаи-

модействия. Задержки обусловлены занятостью общих ресурсов или невыполнением условия синхронизации. В случае, если нет необходимости в ответвлении на вспомогательный процесс, оператор GOTO не употребляется.

ЧИТАТЬ - IR(Individual Reading) - предназначен для чтения данных из любой ветви параллельного процесса. Вызывающая последовательность на языке ФОРТРАН:

```
CALL IR(I,A1,A2,N1,A,N,B,P)
GOTO  M,
```

где IR - имя подпрограммы; A - имя массива, в который надо принять данные; B - начальный адрес данных в ветви N1 .Остальные параметры имеют тот же смысл, что и для макрооператора IW .

НЕСТРУКТУРИРОВАННЫЙ ПЕРЕХОД - UG(Unstructured GOTO) - предназначен для порождения и уничтожения процессов, происходящих в других ветвях. Вызывающая последовательность на языке ФОРТРАН:

```
CALL UG(I,A1,A2,N1,B,P)
GOTO  M,
```

где UG - имя подпрограммы неструктурированного перехода; B - адрес передачи управления в ветви N1 .

ЗАПИСАТЬ ПАСПОРТ СОБЫТИЙ - IWS(Individual Writing of Semafors) - предназначен для извещения других процессов о происходящих событиях через семафорные переменные, а также для управления этими переменными. Вызывающая последовательность:

```
CALL IWS(I,A1,A2,P)
GOTO  M          ,
```

где IWS - имя подпрограммы, реализующей оператор.

ПРОЧИТАТЬ ПАСПОРТ СОБЫТИЙ - IRS(Individual Reading of Semafors) - предназначен для чтения значений семафорных переменных.Вызывающая последовательность:

```
CALL IRS(I,A,P)
GOTO  M  ,
```

где IRS - имя подпрограммы, реализующей оператор; A - адрес, по которому записываются прочитанные значения семафорных переменных.

ЗАДАТЬ ПАСПОРТ СОБЫТИЙ - INP (Individual Name Pasport) - предназначен для задания множества семафорных переменных и иден-

тификации этого множества именем. Вызывающая последовательность:

```
CALL INP(D,I,P)
GOTO M ,
```

где INP- имя подпрограммы, реализующей оператор; D- параметр,характеризующий максимальное значение, которое могут принимать семафорные переменные. Значения каждой из задаваемых семафорных переменных $Q_i \leq 2^D-1$.

УДАЛИТЬ ПАСПОРТ СОБЫТИЙ - IAP (Individual Abort Pasport) - предназначен для удаления множества семафорных переменных с указанным именем. Вызывающая последовательность:

```
CALL IAP(I,P)
GOTO M ,
```

где IAP - имя подпрограммы, реализующей оператор.

В последних пяти операторах неперечисленные параметры аналогичны параметрам оператора ПИСАТЬ.

Системные операторы групповых взаимодействий [5]. Процесс,вышедший на такой оператор, переходит в состояние ожидания.Это ожидание длится до тех пор, пока не наступит событие, определяемое моментом выхода на копии такого же оператора всех процессов параллельной программы. После этого выполняется взаимодействие, задаваемое системным оператором.

ТРАНСЛЯЦИОННЫЙ ОБМЕН - BEXC (N2,N1,N,A,B) - предназначен для передачи информации из одной ветви параллельной программы во все остальные. Через N2 обозначается идентификатор данного оператора, отличающий его от остальных. Для параллельных программ групповые операторы, реализующие одно и то же взаимодействие, должны иметь одинаковые значения N2 во всех ветвях; N1 - номер передающей ветви; A и B - идентификаторы передаваемого и принимаемого массивов; N - количество передаваемых слов.

ДИФФЕРЕНЦИРОВАННЫЙ ОБМЕН- DEXC(N2,N1,N,A,B,Z1,Z2,..., ZP) - предназначен для передачи информации из одной ветви в некоторые выделенные; Z1,..., ZP - номера принимающих ветвей.

ОБМЕН СДВИГОМ - MEXC(N2,X,A,B) - предназначен для передачи (сдвига) информации из каждой ветви в соседнюю с бóльшим (X = 0) или мéньшим (X = 1) относительным номером, т.е. сдвигу информации по ветвям "вправо или влево".

ОБОБЩЕННЫЙ УСЛОВНЫЙ ПЕРЕХОД - GCP(N2,F,M) - предназначен для изменения естественного порядка реализации операторов во всех ветвях при выполнении обобщенного условия, равного конъюнкции от значений F в каждой ветви; M - метка, на которую передается управление, если указанное обобщенное условие истинно.

ТРАНСЛЯЦИОННО-ЦИКЛИЧЕСКИЙ ОБМЕН - CBEXC(N2,N,A,B) - представляет собой трансляционный обмен в цикле по всем ветвям.

КОЛЛЕКТОРНЫЙ ОБМЕН - CEXC(N2,N1,N,A,B) - предназначен для передачи информации из всех ветвей в одну выделенную с относительным номером N1 .

СИНХРОНИЗАЦИЯ SNH (N2,P)

GOTO M или SNH(N2, P) - предназначена для синхронизации (выравнивания во времени) ветвей параллельной программы. Действия оператора совпадают с действиями одноименной системной команды, но реализация отличается дополнительным контролем операционной системы за правильностью выполнения параллельной программы.

Здесь M - входная метка вспомогательного процесса; при P = 0 оператор GOTO отсутствует.

Для взаимодействия основного и вспомогательного процессов в рамках одной ЭМ в состав системных операторов введен оператор MW1, предназначенный для передачи управления на продолжение вспомогательного процесса, который был приостановлен в ветви при выполнении последнего системного оператора.

2.4. С т а р ш и й д и с п е т ч е р. В нем выделяются три главных компонента: супервизор реального времени АСВТ-М, ядро и системный загрузчик.

Супервизор реального времени АСВТ-М осуществляет запуск ядра старшего диспетчера и обрабатывающих пользовательских программ, реализуемых в диспетчерской ЭМ в соответствии с их временными характеристиками. Все управляемые программы оформлены согласно требованиям супервизора реального времени АСВТ-М. Возможна такая структура обрабатывающих программ, при которой в диспетчерской ЭМ находится только их резидентная ветвь (оформленная как управляемая супервизором программа), в других ЭМ системы расположены остальные ветви, взаимодействующие с резидентной и между собой посредством системных команд и операторов.

Системный загрузчик [6] является автономным модулем старшего диспетчера. Он осуществляет загрузку программ в систему централи-

зованно, через выделенную машину; сначала вводит ветвь параллельной программы в одну из машин, настраивает ее на память, которую она будет занимать в системе, а затем в абсолютном формате посылает в заданную машину. Системный загрузчик устанавливает непосредственную связь между памятями ветвей параллельной программы; это повышает скорость взаимодействия между ветвями во время их выполнения и тем самым уменьшает время реакции системы при работе в реальном масштабе времени.

Ядро старшего диспетчера осуществляет управление межмашинными связями в системном коммутаторе, выполняет прием и анализ запросов от диспетчеров ЭМ на реализацию системных взаимодействий, производит обслуживание этих запросов, управляет аппаратом событий, связанных с синхронизацией индивидуальных взаимодействий процессов.

Управление межмашинными связями в системном коммутаторе заключается в программировании путей в двухмерной решетке коммутатора с целью обеспечения взаимодействий между элементарными машинами подсистемы.

Программирование указанных путей заключается в настройке модулей межмашинных связей, расположенных вдоль пути в двухмерной решетке системного коммутатора, соединяющего машины, участвующие во взаимодействии. Это обеспечивает доступ любого процессора подсистемы к любой области оперативной памяти всех ЭМ подсистемы.При программировании путей используется команда НАСТРОЙКА СИСТЕМЫ.

Проверка наличия, прием и анализ запросов от диспетчеров ЭМ выполняется с помощью системных команд ПРОВЕРКА ЗАПРОСА СВЯЗИ, НАСТРОЙКА СИСТЕМЫ, ПРИЕМ и ПЕРЕДАЧА. При обнаружении в какой-либо ЭМ "запроса" программируется путь к этой ЭМ; с помощью системной команды ОБП в ней прерывается работа процессора и ей передается тип задания, показывающий, что старший диспетчер готов принимать запрос. Запросы поступают от диспетчеров ЭМ в виде массива параметров, необходимых для осуществления требуемого системного взаимодействия.

Имеются семь типов запросов, соответствующих семи типам операторов индивидуальных взаимодействий; число в первом слове указывает на тип системного оператора, которому соответствует данный запрос: I - соответствует оператору IW; 2 - оператору IR;3 - оператору UG; 4 - оператору IWS; 5 - оператору IRS ; 6 - оператору INP; 7 - оператору IAP:

1	2	3	4	5	6	7
± ⟨I⟩	± ⟨I⟩	± ⟨I⟩	± ⟨I⟩	± ⟨I⟩	± ⟨I⟩	± ⟨I⟩
⟨A1⟩	⟨A1⟩	⟨A1⟩	⟨A1⟩		⟨D⟩	
⟨A2⟩	⟨A2⟩	⟨A2⟩	⟨A2⟩			
⟨A21⟩	⟨A21⟩	⟨A21⟩	⟨A21⟩			
⟨N1⟩	⟨N1⟩	⟨N1⟩				
⟨A⟩	⟨A⟩			⟨A⟩		
⟨N⟩	⟨N⟩					
⟨B⟩	⟨B⟩	⟨B⟩				

Смысл и назначение остальных параметров в запросе описан в п. 2.3.

Управление аппаратом событий для синхронизации индивидуальных взаимодействий осуществляется на основе определенных операций над семафорными переменными. При этом каждый процесс асинхронно выходит на ожидание требуемого ему индивидуального события и асинхронно оповещается о наступлении события.

Для синхронизации индивидуальных взаимодействий каждая задача, реализующаяся на системе, задает с помощью специального оператора (см. п.2.3) одно или несколько наборов семафорных переменных. Каждый набор идентифицируется именем и представляется в виде упорядоченной последовательности переменных: $Q_1\ Q_2\ \ldots\ Q_z$.

В запросе на взаимодействие указываются имя набора семафорных переменных и функция A1 , выделяющая подмножество из данного набора. На переменных этого подмножества задается логическая функция - конъюнкция или дизъюнкция, - определяемая в запросе знаками "+" или "-" соответственно.

Условие синхронизации считается выполненным, если $\bigwedge_{i \in A} Q_i > 0$ или $\bigvee_{i \in A} Q_i > 0$, где A - множество выделенных семафоров; значения $Q_i^{max} \geq Q_i \geq 0 \quad \forall i \in \{1, \ldots, z\}$.

После выполнения синхронизации и реализации связанного с ней запроса осуществляется преобразование всего набора семафорных переменных. Вид преобразования определяется параметрами A2 и A21.

Над семафорами определены P- и V -операции:
$P(Q_i) = \hat{Q}_i + 1$ и $P(Q^{max}) = \hat{Q}_i^{max}$, где $\hat{Q}_i$ - значение семафора Q_i,

$\hat{Q}_i^{max}$ – максимальное значение, которое может принимать семафор Q_i; $V(Q_i) = \hat{Q}_i - 1$ и $V(Q_i^0) = 0$, где Q_i^0 – нулевое значение семафора Q_i.

Степени P- и V -операций определяются следующим образом:

$$P^n(Q_i) = P(P(\ldots(P(Q_i))\ldots) = \hat{Q}_i + n, \qquad \hat{Q}_i + n \le \hat{Q}_i^{max};$$

$$V^n(Q_i) = V(V(\ldots(V(Q_i))\ldots) = \hat{Q}_i - n, \qquad \hat{Q}_i - n \ge 0;$$

$$P^n(Q_i^{max}) = Q_i^{max}, V^n(Q_i^0)=0, P^0(Q_i) = \hat{Q}_i, V^0(Q_i) = \hat{Q}_i.$$

Функция, определяемая параметрами A2 и A21, есть упорядоченная последовательность степеней P- и V-операций, количество которых равно числу семафоров в заданном наборе. Элемент последовательности i действует только на семафор с индексом i .Например:

$$A2,A21 = \underset{\downarrow}{P_1^{n_1}}\ \underset{\downarrow}{V_2^{n_2}}\ \underset{\downarrow}{V_3^{n_3}}\ \underset{\downarrow}{P_4^{n_4}} \ldots \underset{\downarrow}{V_z^{n_z}}$$
$$Q_1 \quad Q_2 \quad Q_3 \quad Q_4 \qquad Q_z$$

Блок, реализующий управление аппаратом событий, в которое входят проверка условия синхронизации и выполнение преобразований над семафорами, функционирует по принципу монитора Хоара [7].

О б с л у ж и в а н и е з а п р о с о в, п о с т у п а ю щ и х от диспетчеров ЭМ. При невыполнении условия синхронизации запрос ставится в очередь, при выполнении – реализуются следующие действия:

I. Запрос от операторов ПИСАТЬ и ЧИТАТЬ. В этом случае:

I) Программируется путь в двухмерной решетке коммутатора к ЭМ, диспетчер которой указан в запросе.

2) При помощи системных команд ОБОБЩЕННЫЙ БЕЗУСЛОВНЫЙ ПЕРЕХОД, ПРИЕМ и ПЕРЕДАЧА передаются управляющее слово, характеризующее тип задания, и информационная часть задания, состоящая из двух параметров ⟨Адр В⟩, ⟨N⟩, взятых из запроса. Тип задания записывается в регистр слова состояния. Диспетчер ЭМ, получив задание, ждет разрешения на выполнение этого задания.

3) Программируется путь к ЭМ, от диспетчера которой получен запрос, и при помощи команды ОБП передается управляющее слово, указывающее, что условие синхронизации, связанное с запросом, выполнено. После этого диспетчер ждет разрешения на выполнение своего запроса.

4) Программируется путь между двумя ЭМ,ожидающими взаимодействия,что и является разрешением для диспетчера ЭМ начать взаимодействие.

Если запрос обращен к ЭМ,в которой находится старший диспетчер, то последний с помощью диспетчера из этой же ЭМ реализует обмен. Аналогично, если запрос получен от диспетчера из ЭМ, в которой находится старший диспетчер, то он с помощью диспетчера ЭМ реализует обмен.

П. Запрос от оператора НЕСТРУКТУРИРОВАННЫЙ ПЕРЕХОД. В этом случае:

I) Программируется путь к ЭМ, диспетчер которой указан в запросе.

2) При помощи системных команд ОБП, ПРИЕМ и ПЕРЕДАЧА пересылаются тип задания и адрес, по которому диспетчер ЭМ передает управление.

Если запрос обращен к диспетчеру из ЭМ, в которой находится старший диспетчер, то последний передает управление по указанному адресу. Если запрос получен от диспетчера из ЭМ со старшим диспетчером, то старший диспетчер с помощью диспетчера ЭМ своей машины передает тип задания и адрес диспетчеру требуемой ЭМ.

Ш. Запросы от операторов, связанных с действиями над семафорами. В этом случае:

I) Старший диспетчер выполняет указанное в запросе действие.

2) Передаются значения семафоров в ЭМ, от которой поступил запрос, если запрос исходил от оператора IRS.

2.5. Д и с п е т ч е р ЭМ. В нем выделяются 5 основных компонентов: I) обработчик системных прерываний; 2) обработчик аварийных ситуаций; 3) драйвер модуля межмашинных связей; 4) основная управляющая система АСИТ-М и 5) библиотека модулей системных операторов.

О б р а б о т ч и к системных прерываний, наряду со стандартными прерываниями М-6000, обслуживает прерывания ЭМ, связанные с выполнением системных команд. Получив управление, обработчик системных прерываний запоминает вектор состояния (регистры А, В, РП, Е, РТА) процесса в точке прерывания и анализирует тип прерывания. Тип прерывания распознается по слову состояния из регистра слова состояния. Имеются два типа системных прерываний: I) по команде СИНХРОНИЗАЦИЯ; 2) по команде ОБОБЩЕННЫЙ БЕЗУСЛОВНЫЙ ПЕРЕХОД.

При прерывании по синхронизации диспетчеры ЭМ во всех машинах, ожидающих этого события и находящихся в состоянии динамического останова либо выполнения вспомогательного процесса, передадут управление команде, следующей за командой СИНХРОНИЗАЦИЯ.

При прерывании по ОБП в регистр слова состояния этой командой записывается код управляющего слова, характеризующий тип задания для диспетчера ЭМ. Имеются пять типов заданий, обозначенных соответственно числами I,2,3,4,5, в соответствии с которыми диспетчер ЭМ выполняет требуемые действия. Перед выполнением требуемых заданий диспетчер ЭМ запоминает вектор состояния прерванного процесса.

I. По заданию первого типа диспетчер ЭМ принимает от старшего диспетчера информационную часть задания и передает управление по адресу, указанному в задании.

2. По заданию второго типа диспетчер ЭМ передает управление команде, следующей за системной командой ЗАПРОС СВЯЗИ, реализующейся в данный момент и находившейся в стадии ожидания ответа от старшего диспетчера.

3. По заданию третьего типа диспетчер ЭМ передает управление на продолжение выполнения системного оператора, который находился в стадии ожидания условия синхронизации.

4. По заданию четвертого типа диспетчер ЭМ принимает от старшего диспетчера информационную часть задания, в котором указаны адрес и длина массива, находившегося в прерванной ЭМ.Диспетчер ЭМ передает указанный массив затребовавшему его процессу. После передачи восстанавливаются состояние модуля межмашинной связи и состояние процесса, которое они имели до прерывания процессора старшим диспетчером.

5. По заданию пятого типа диспетчер ЭМ выполняет действия, аналогичные описанным в предыдущем пункте, только осуществляется прием массива, адрес и длина которого указаны в задании.

О б р а б о т ч и к аварийных ситуаций состоит из двух частей: анализатора работы модуля межмашинной связи и анализатора значений параметров операторов групповых и индивидуальных взаимодействий.

Первый анализатор распознает и распечатывает на терминал причину прерывания процессора по сигналу "Ошибка ввода/вывода" при обращении к модулю межмашинной связи и причину ошибок, возникших

в процессе реализации системных команд драйвером модуля межмашинной связи.

Второй анализатор распознает и распечатывает на терминал причину:

- ошибки в запросах операторов индивидуальных взаимодействий;

- несоответствия значений параметров в ветвях при реализации операторов групповых взаимодействий.

Обработчик аварийных ситуаций участвует в организации устойчивых (надежных) вычислений. Это участие связывается с применением RPT-операторов в пользовательских программах [8]. Эти RPT-операторы задают точки возврата, на которые обработчик прерываний передает управление в случае сбойной ситуации.

Д р а й в е р модуля межмашинной связи реализует системные команды (см. п.I.2), обеспечивающие функционирование системного коммутатора и всего комплекса как единой многопроцессорной установки.

О с н о в н а я управляющая система АСВТ-М обеспечивает загрузку перемещаемых программ, управление вводом/выводом в ветвях параллельных программ. Основная управляющая система скомпонована модульно и компонуется для каждой конкретной конфигурации технических средств.

Б и б л и о т е к а модулей системных операторов (см.п.2.3.) представляет собой автономный набор подпрограмм, реализующих групповые и индивидуальные схемы взаимодействий между ветвями параллельных программ. Модули библиотеки загружаются в систему вместе с обрабатывающими (пользовательскими) программами.

§4. Операционная система и языковая иерархия

Операционную систему МИНИМАКС можно представить в виде нескольких уровней (слоев) программного обеспечения. Первый слой программного обеспечения вместе с нижележащей аппаратурой обеспечивает выполнение определенного множества команд, которые можно рассматривать как внутренний язык системы. Добавление второго слоя определяет систему с внутренним языком более высокого уровня и т.д. В системе МИНИМАКС можно выделить четыре уровня языковой иерархии.

П е р в ы й уровень состоит из МНЕМОКОДа, расширенного системными командами: ПРИЕМ, ПЕРЕДАЧА, СИНХРОНИЗАЦИЯ, ОБОБЩЕННЫЙ УСЛОВНЫЙ ПЕРЕХОД, СБРОС РЕГИСТРОВ В МОДУЛЕ МЕЖМАШИННОЙ СВЯЗИ.

Этот уровень обеспечивает децентрализованное функционирование элементарных машин подсистемы, выполняющих режим параллельной обработки, и является языком для записи групповых схем взаимодействий между ветвями параллельной программы.

Второй уровень состоит из МНЕМОКОДа, расширенного системными командами: НАСТРОЙКА СИСТЕМЫ, ЗАПРОС СВЯЗИ, ПРОВЕРКА ЗАПРОСА СВЯЗИ, ОБОБЩЕННЫЙ БЕЗУСЛОВНЫЙ ПЕРЕХОД. Этот уровень обеспечивает запуск и прерывание процессов, управление структурой связи между ЭМ, выполнение запросов на взаимодействия с другими процессами и проверку наличия этих запросов, обеспечивает средства децентрализованного взаимодействия между ветвями параллельной программы.

Третий уровень состоит из языков АЛГОЛ, ФОРТРАН, МНЕМОКОД, расширенных системными операторами групповых взаимодействий: ТРАНСЛЯЦИОННЫЙ ОБМЕН, ОБМЕН СДВИГОМ, ДИФФЕРЕНЦИРОВАННЫЙ ОБМЕН, КОЛЛЕКТОРНЫЙ ОБМЕН, ТРАНСЛЯЦИОННО-ЦИКЛИЧЕСКИЙ ОБМЕН, СИНХРОНИЗАЦИЯ, ОБОБЩЕННЫЙ УСЛОВНЫЙ ПЕРЕХОД.

Третий уровень, аналогично первому, обеспечивает децентрализованное функционирование элементарных машин подсистемы, выполняющих режим параллельной обработки. Но с точки зрения языковой иерархии является расширением языков более высокого уровня для задания групповых схем взаимодействий между ветвями.

Четвертый уровень состоит из языков АЛГОЛ, ФОРТРАН, МНЕМОКОД, расширенных системными операторами индивидуальных взаимодействий: ПИСАТЬ, ЧИТАТЬ, НЕСТРУКТУРИРОВАННЫЙ ПЕРЕХОД, ЗАДАТЬ ПАСПОРТ СОБЫТИЙ, ЗАПИСАТЬ В ПАСПОРТ СОБЫТИЙ, ЧИТАТЬ ПАСПОРТ СОБЫТИЙ, УДАЛИТЬ ПАСПОРТ СОБЫТИЙ.

Этот уровень обеспечивает централизованное функционирование элементарных машин подсистемы, выполняющих режим параллельной обработки. В отличие от языка второго уровня, он является расширением языков программирования высокого уровня для задания асинхронных схем взаимодействий между модулями параллельной программы, реализующейся на подсистеме.

При использовании указанных расширенных языков высокого уровня пользователю необязательно знать аппаратурные особенности взаимодействия ЭМ в системе.

§5. Вместо заключения

В предыдущих разделах операционная система МИНИМАКС описывалась, по существу, с разных точек зрения, т.е. в ней выделялись особенности взаимодействия процессов, уровни принятия решений, языковая иерархия и др. В [9] операционные системы подобного класса предлагается рассматривать как пятерку конечных множеств $\langle S, \Phi, G, \Gamma, U \rangle$, где S - множество программных блоков; Φ - множество алгоритмов распознавания мультипрограммной ситуации и алгоритмов управления; G - множество параметров, определяющих мультипрограммную ситуацию; Γ - граф, задающий отношение между блоками при доступе к общим параметрам мультипрограммной ситуации; U - множество векторов инициации блоков из S.

Такой взгляд на указанную операционную систему не противоречит упомянутым выше точкам зрения и позволяет их рассматривать в рамках единого подхода, а также служит основой для аналитических исследований определенных свойств операционных систем. Рассмотрим эти множества.

Множество S состоит из двух подмножеств S_1 и S_2, соответствующих двум типам программных блоков. Каждый блок из S_1 есть диспетчер элементарной машины; из S_2 - старший диспетчер. Блоки из S_1 рассредоточены по всем машинам, и в оперативной памяти каждой машины системы имеется один блок. Каждый блок из S_2 располагается в одной из ЭМ, называемой диспетчерской. Количество блоков в S_2 равно количеству автономно работающих подсистем.

Множество Φ состоит из следующих алгоритмов: настройки системы; запроса на взаимодействие; настройки модуля межмашинной связи; групповой синхронизации процессов, реализующихся в разных ЭМ подсистемы; группового обмена данными; обобщенного безусловного и условного переходов; приведения модуля межмашинной связи в исходное состояние; управления системными каналами; приема и анализа запросов от процессов; управления аппаратом событий при индивидуальных взаимодействиях процессов; реализации запросов; обеспечения связи ЭВМ с внешними устройствами; запуска програми в соответствии с их временными характеристиками.

К алгоритмам распознавания мультипрограммной ситуации относятся: распознавание системных прерываний и директив от старшего диспетчера, а также распознавания аварийных прерываний.

Множество G определяет мультипрограммную ситуацию для блоков операционной системы и связано с реализацией взаимодействующих

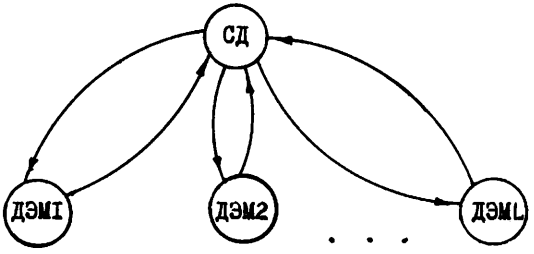
друг с другом процессов. К указанному множеству относятся следующие параметры: регистр R_z предназначен для выставления признака запроса алгоритмом запроса; R_c - для записи признака синхронизации, выставляемого алгоритмом синхронизации; $R_н$ - для настройки модуля межмашинной связи (в $R_н$ засылается направление приема информации и участие ЭМ во взаимодействии); регистр СС - для анализа состояния модуля межмашинной связи и считывания типа задания, посылаемого старшим диспетчером; триггер T_y - для инициирования прерывания работы процессора через ячейку прерываний, связывающую управляющий канал модуля межмашинной связи с ЭВМ; триггер T_D - для инициирования прерывания работы процессора через ячейку прерываний, связывающую информационный канал модуля межмашинной связи с ЭВМ.

Все вышеперечисленные регистры находятся в каждом модуле межмашинной связи.

Диспетчеры элементарных машин имеют доступ к перечисленным ресурсам, которые находятся только в "своей" ЭМ, причем этот доступ не зависит от доступа других блоков операционной системы к некоторым из этих ресурсов.

Старший диспетчер имеет индивидуальный доступ по записи одновременно ко всему множеству регистров $\{R_н\}$ либо к любому его подмножеству. К регистрам настройки доступ по записи имеют алгоритмы настройки системы. К множеству всех регистров $\{R_z\}$ старший диспетчер имеет индивидуальный доступ по чтению. Этот доступ осуществляет алгоритм проверки запросов; считывание осуществляется с каждого R_z в отдельности. Старший диспетчер имеет доступ по записи ко всему множеству триггеров $\{T_y\}$ и регистров {СС} либо к любому их подмножеству. Этот доступ осуществляет алгоритм обобщенного безусловного перехода, причем запись может производиться одновременно в требуемые регистры.

Отношение между блоками при доступе к общим параметрам мультипрограммной ситуации можно представить в виде следующего графа Г:



Множество векторов U инициирует запуск процессов, реализующихся в системе. Каждый вектор состоит из заданного множества компонент (в общем случае разного для каждого вектора). Компонентами являются адреса ячеек прерываний ЭВМ от внешних устройств и от схем контроля процессора, а также входные метки подпрограмм, реализующих некоторые алгоритмы из Ф . Каждому блоку из S соответствует свой вектор из U. Диспетчеру элементарной машины соответствует вектор U = (KBY,KBD,ZSW,SINX,WSN,SBR,OUP,EXEK,FPPP,WKN,PSW), где KBY - ячейка прерываний процессора, связывающая управляющий канал модуля межмашинной связи с ЭВМ; KBD связывает информационный канал модуля межмашинной связи с ЭВМ; остальные параметры - входные метки подпрограмм. Старшему диспетчеру соответствует вектор U = (CD,CD1) , компонентами которого являются входные метки.

Литература

1. КОРНЕЕВ В.Д. Супервизор реального времени однородной вычислительной системы МИНИМАКС. Отчет ИМ СО АН СССР, 1978.

2. ДИМИТРИЕВ Ю.К., МИРЕНКОВ Н.Н., ХОРОШЕВСКИЙ В.Г. и др. Однородная вычислительная система из мини-машин. -В кн.: Вычислительные системы, вып. 51, Новосибирск, 1972, с.127-145.

3. КЕРБЕЛЬ В.Г., КОЛОСОВА Ю.И., КОРНЕЕВ В.Д., МИРЕНКОВ Н.Н. Средства программирования системы МИНИМАКС.-В кн.: Вычислительные системы, вып.60. Вопросы теории и построения вычислительных систем. Новосибирск, 1974, с. 143-153.

4. КЕРБЕЛЬ В.Г., КОРНЕЕВ В.Д., КРЫЛОВ Э.Г. Языки для описания параллельных процессов. Операторы индивидуальных взаимодействий. Отчет ИМ СО АН СССР, 1977.

5. КОЛОСОВА Ю.И. Языки для описания параллельных процессов. Операторы групповых взаимодействий. Отчет ИМ СО АН СССР, 1977.

6. КРЫЛОВ Э.Г. Загрузка параллельных программ в однородную вычислительную систему МИНИМАКС. -Настоящий сборник, с.31-39.

7. HOARE C.A.R., PERROT R.H. Operating Systems Techniques.- New York, Acaxdemic Press, 1972,p.36-53.

8. КОРНЕЕВ В.Д. Программные средства для организации отказоустойчивых вычислений в системе МИНИМАКС. -Настоящий сборник, с. 40-53.

9. МИРЕНКОВ Н.Н. Однородные вычислительные системы. Структурная организация операционных систем. -Новосибирск, Б.и.,1977.- 48 с.-(Препринт/ИМ СО АН СССР; ОВС-01,ОВС-02).

Поступила в ред.-изд.отд.
21 мая 1979 года