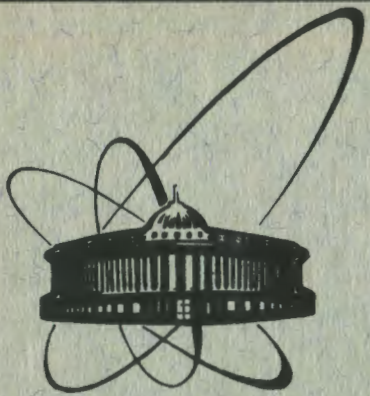


сообщения
объединенного
института
ядерных
исследований
дубна

Р11-91-47

А.П. Сапожников

ПРИНЦИПЫ ОРГАНИЗАЦИИ И СТРУКТУРА ОС МКБ-8601

1991

Весной 1990 года в ЛВТА ОИЯИ завершилось изготовление макетного образца МКБ-8601 [1,2], интегральной ЭВМ ряда БЭСМ-6. Параллельно работам по проектированию аппаратуры велась разработка операционной системы (ОС) МКБ. Использовалась кросс-система [3,4], созданная на БЭСМ-6 специально для этого проекта. К моменту вывода БЭСМ-6 из эксплуатации макетный образец МКБ-8601 был уже в состоянии выполнять программы кросс-системы в режиме эмуляции БЭСМ-6, так что все работы по ОС были перенесены непосредственно на обьектную ЭВМ без заметного перерыва. Настоящая публикация посвящена описанию основных понятий и принципов, заложенных в новой ОС. Хочется надеяться, что она откроет целую серию публикаций на эту тему. Необходимым условием является, конечно, развертывание хотя бы мелкосерийного производства МКБ-8601.

Как и для всех операционных систем, поддерживающих мультипрограммный режим работы ЭВМ, для ОС МКБ основной структурной единицей является ПРОЦЕСС. Процесс-это работа, производимая последовательным процессором при выполнении программы с ее данными.[7]. При этом в любой момент времени каждый процессор занят ровно одним процессом, а каждый процесс использует не более чем один процессор. Таким образом, можно выделить 3 возможных состояния процесса:

- активен, т.е. занимает один из N процессоров ЭВМ;
- готов захватить процессор;
- не готов из-за ожидания какого-то внешнего события.

В каждом процессоре МКБ есть однобайтовый регистр, содержащий номер текущего активного процесса. Поэтому всего в системе может быть одновременно не более 256 процессов. Они пронумерованы от 0 до 255. Процессы бывают двух типов :

- диспетчерские. Они постоянно живут в системе, поэтому за каждым из них можно навсегда закрепить свой номер;
- пользовательские. Эти процессы имеют свое начало и конец. При рождении пользовательского процесса он получает

номер из числа свободных. По завершении процесса номер освобождается. Согласно этому критерию к пользовательским следует отнести и все служебные процессы, запускаемые с операторской консоли.

Каждый процесс имеет свое собственное виртуальное адресное пространство размером **2**20** слов, разделенное на 1024 виртуальных страницы. Страницы **0-511** предназначены для размещения программ, работающих в режиме пользователя, и их данных. Диспетчерские процессы эту память не используют. Страницы **512-1021** предназначены для ОС, причем диспетчерские процессы используют их "напрямую", т.к. "знают" распределение памяти в ОС. Пользовательские процессы получают доступ к этим страницам косвенно, через аппарат экстракодов. Часть из этих страниц используется как буферная область при работе с файлами. Страницы **1022-1023** представляют из себя информационное поле, хранящее текущее состояние процесса, а также стек его рабочих переменных.

Архитектура ОС и принципы ее организации определяются способами взаимодействия процессов, а также перечнем диспетчерских процессов и описанием их функционирования.

Процесс обработки прерываний

В каждом из возможных **N** процессоров системы МКБ-8601 аппаратура для обслуживания прерываний идентична. Все процессоры работают асинхронно, поэтому для каждого процессора требуется отдельный процесс обработки прерываний. Поскольку программы хранятся в общей памяти, то программный код у всех этих **N** процессов (за ними закреплены номера от **0** до **N-1**) общий. Каждый из этих процессов работает в режиме блокировки внешних прерываний на своем процессоре. Для него зарезервирована одна из 32 групп рабочих регистров процессора (0-я). Процессы обработки прерываний имеют абсолютно высший приоритет, т.к. активизируются аппаратно. Все другие процессы запускаются программно, их приоритеты вычисляются динамически и почти не зависят от номера процесса (при прочих равных условиях предпочтение отдается процессу с меньшим номером). Все работы с очередями процессов инициируются специальным программным прерыванием, поэтому все функции

планирования загрузки процессоров системы МКБ локализованы именно в блоке реакции на прерывания.

Ждущий процесс

Поскольку любой процессор всегда должен быть чем-то занят, для него должен существовать особый процесс, единственно необходимой функцией которого является постоянная готовность использовать процессор. Для таких ждущих процессов зарезервированы номера от **N** до **2N-1**, а также одна (1-я) группа рабочих регистров в каждом процессоре. Ждущие процессы имеют, естественно, абсолютно низший приоритет.

Об управлении процессами

Перечисленные выше процессы жестко привязаны каждый к своему процессору. Привязка обусловлена использованием внутренних регистров процессора, а точнее, - необходимостью статического резервирования этих регистров. Все остальные процессы могут "кочевать" с одного процессора на любой другой. Очевидно, что такие перемещения процесса должны происходить не по инициативе его самого, а планироваться с учетом эффективности системы в целом. Основные принципы разделения времени, отработанные в диспетчере ОС "Дубна" [5,6] на БЭСМ-6, мы стремились сохранить и в МКБ-8601. Наличие в системе нескольких процессоров учитывается, по существу, только в программе активации процесса, логика же работы всех процессов не зависит от числа процессоров и их состояния.

Все готовые процессы в системе связаны в список, упорядоченный по убыванию приоритетов (процессы обработки прерываний сюда не входят, т.к. они активизируются аппаратно !). Смена активного процесса на процессоре осуществляется в двух случаях :

- при поступлении в список готовых нового процесса с приоритетом большим, чем у текущего;
- при исчерпании текущим процессом своего кванта времени;

Смена процессов на процессоре осуществляется, как уже говорилось выше, процессом обработки прерываний этого процессора. В первом случае она инициируется специальным программным прерыванием, выдаваемым программой, установившей готовность нового процесса. Во втором случае смена процессов инициируется прерыванием от таймера. Кроме того, необходимость обьявить готовым некоторый (обычно диспетчерский) процесс выявляется и при обслуживании внешних прерываний.

Из списка готовых выбирается первый процесс, не привязанный к "чужому" процессору и не ждущий обслуживания запроса на память. Принадлежащая ему группа рабочих регистров процессора становится активной. При необходимости группа динамически выделяется, и в нее копируется содержимое из информационного поля процесса. Исполнение команды "возврат из прерывания" завершает процедуру смены процессов.

Если же вновь поступивший в список готовых процесс имеет приоритет меньший, чем у текущего, соседнему процессору посылается специальное прерывание с тем, чтобы он, если имеет возможность, "подхватил" этот новый процесс.

Снятие готовности процесса производится только по его собственной инициативе с помощью программного прерывания, т.е. активный процесс сам "замораживает" свое состояние. Процесс обслуживания прерываний исключает его из списка готовых, после чего процессор отдается другому процессу. Готовность, возникающая вновь при наступлении ожидаемого события, приводит к продолжению процесса с прерванного места.

При исчерпании кванта времени у текущего процесса корректируется динамический приоритет. Если полученное значение динамического приоритета превысило предел, установленный для процессов данного класса, то необходим полный пересчет приоритетов в классе. Это сравнительно длинная работа, поэтому для ее проведения активируется специальный процесс.

Процесс пересчета приоритетов

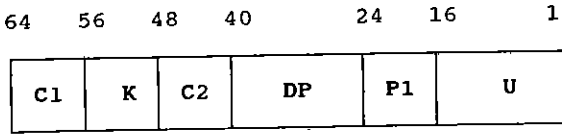
За ним закреплен номер **2N**. Этот диспетчерский процесс осуществляет нормирование характеристик использования времени всеми процессами заданного класса. После этого заново пересчитываются их приоритеты. Список готовых процессов пе-

рестраивается в соответствии с новыми значениями приоритетов. Закончив эту работу, процесс пересчета приоритетов снимает свою готовность.

С точки зрения системы разделения времени процессы делятся на классы, так что представитель класса с большим номером безусловно приоритетнее любого представителя класса с меньшим номером. Класс процесса не изменяется в ходе его работы. Перечень классов, сложившийся еще в ОС "дубна" на БЭСМ-6, таков :

0 - ждущие процессы; 1 - резервные задачи;
2 - задачи пользователей; 3 - служебные процессы;
4 - диспетчерские процессы;

Каждому классу **K** соответствует своя величина учетного кванта времени **Nk**, не изменяющаяся во время работы системы. Глобальным параметром системы разделения времени является **E** - минимальный квант, равный времени подкачки одной страницы. Процессорное время **T** выделяется процессам квантами **T=P*Q*E**, где **Q**-нормированный в классе **K** квант процесса, **P**-внешний приоритет процесса. Таким образом, время делится между процессами прямо пропорционально их внешним приоритетам.
64-разрядное слово приоритета процесса имеет следующую структуру:



K - класс процесса; **P1** = 256-<номер процесса>;
DP - динамический приоритет процесса. **DP=(Nk-U)/Q** ;
U - характеристика использования процессорного времени. Сюда добавляется **Q** или его доля, если текущий квант использован не полностью. При **U>Nk** **U**-характеристики пересчитываются у всех процессов данного класса **K**;
C1,C2 - счетчики семафоров, захваченных процессом. При входе (выходе) из т.н. критического интервала [7] увеличивается (уменьшается) на 1 либо **C1**, либо **C2**, в зависимости от типа семафора. Это наиболее простой способ временного увеличения приоритета у процесса, находящегося в критическом интервале.

Эту работу осуществляет диспетчерский процесс с номером **2N+1**. Так же, как это было и на БЭСМ-6, выделение физической памяти процессам производится страницами по запросам. Точно так же применяется бронирование резервной страницы. Реализация здесь намного проще, чем на БЭСМ-6. Во-первых, физическая память МКБ однородна (на БЭСМ-6 для системных запросов можно было использовать только 0-й сегмент!). Во-вторых, в МКБ ведется аппаратный, вернее, микропрограммный подсчет приоритетов физических страниц памяти и отслеживание изменений их содержимого.

Запрос страницы производится, так же как и в БЭСМ-6, при возникновении прерывания по защите памяти. Процесс, которому понадобилась страница, помечается признаком блокировки активации, но не исключается из списка готовых (он не виноват, что конкуренты потеснили его в памяти!). Запрос содержит номер процесса, номер требуемой виртуальной страницы и тип работы. По номеру страницы определяется ее тип, он и задает стратегию поиска страницы.

1 - личные (Private) страницы процесса. Все личные страницы образуют т.н. файл укачки процесса. Сюда же входит и информационное поле процесса. Файл укачки создается при запуске процесса и уничтожается после завершения процесса. Такая организация обеспечивает дешевый способ консервации задач перед запланированной остановкой ЭВМ.

2 - буферные (Buffer) страницы. Являются "окнами" в файлы, с которыми работает процесс. По запросу буферной страницы необходимый обмен осуществляется непосредственно с тем файлом, в который "смотрит окно".

3 - общие (Common) страницы, разделяемые одновременно несколькими процессами. Такие страницы отображаются на файлы общего доступа. Пока этот тип страниц используется только для включения самой ОС в адресное пространство всех процессов.

Запрошенная страница считывается с диска на резервную страницу физической памяти. Резервная страница отдается процессу-клиенту, у которого сразу же после этого снимается блокировка активации. Затем определяется новая физическая страница, подлежащая укачке с целью новообразования резерва. Она отнимается у своего владельца. Ее тип определяет место ее записи на диске. Откачка производится на фоне работы процесса, подавшего запрос на память. После завершения откачки процесс управления памятью снимает свою готовность, а также блокировку активации у всех процессов, ждущих своей очереди подать запрос на память.

Кроме подкачки страницы, основного типа запроса на память, введена еще превентивная укачка страницы на свое место по прямому запросу процесса-владельца. Это может оказаться полезным для задач, требующих длительного счета.

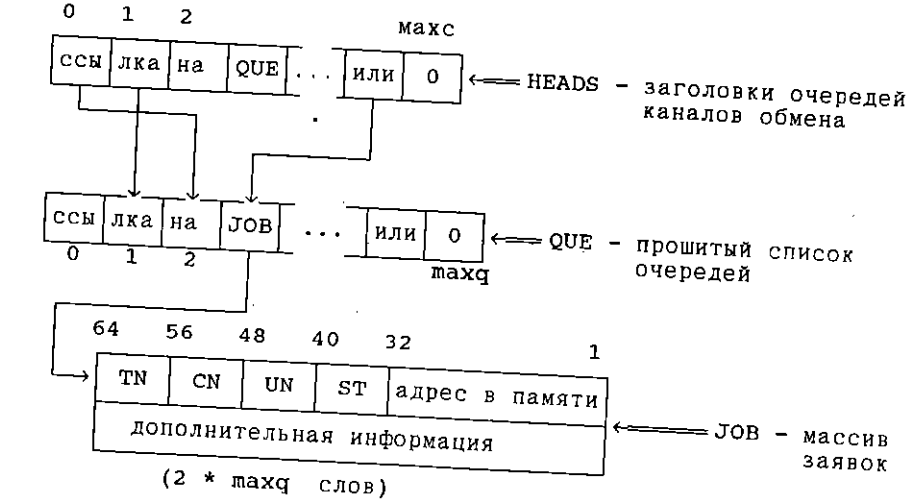
Управление дисками

Осуществляется диспетчерским процессом с номером **2N+2**. Постановка заявки на обмен в общую очередь делается от имени самого процесса-клиента. При этом, так же как и в БЭСМ-6, податель заявки дезактивируется не сразу, а только если обратится к странице, занятой в обмене. Собственно же процесс управления дисками активизируется прерыванием по концу обмена. Он занимается реакцией на возможные ошибки, декомпрессией обменной страницы, продвижкой очереди и активизацией, если это необходимо, процесса, подавшего обмен.

Конструктивно оба блока, постановки в очередь и продвижки ее, хотя и исполняются от имени разных процессов, выполнены в виде единой программы. Это, на наш взгляд, один из наиболее существенных элементов стиля программирования драйверов, сложившегося на БЭСМ-6 еще в незапамятные времена, и не утратившего своей прелести и ныне.

На физическом уровне управление всеми внешними устройствами МКБ осуществляется пульт-процессором, реализованным на базе микропроцессора **INTEL-8086**. Очередь заявок ведется в памяти, доступной как ПП, так и ЦП. Разработан протокол, обеспечивающий синхронизацию доступа к очереди при совместной работе обоих процессоров. По существу, это обьединение

нескольких (по числу каналов) очередей, организованное с помощью метода прошитых списков.



Здесь TN-номер процесса, CN-номер канала,
UN-номер устройства на канале, ST-статус обмена. Дополнительная информация специфична для каждого обменного канала. Поступающая в систему заявка на обмен попадает в свободную позицию J массива JOB. С каждым каналом обмена C=0,1,...,maxc связана своя очередь заявок. C-й байт HEADS является "головой" C-й очереди. Он содержит 0, если очередь пуста, иначе - индекс начала соответствующей очереди в QUE. 0-й байт QUE является "головой" списка свободных позиций в массиве JOB. Если L-й байт QUE (L=1,2,...,maxq) содержит код J > 0, это означает, что J-я заявка из JOB следует в своей очереди обменов непосредственно за L-й. J=0 означает, что L-я заявка является последней в своей очереди. Такая, нетрадиционная для БЭСМ-6, структура данных, чрезвычайно просто реализуется в МКБ-8601, имеющей адресацию байтов.

Анализатор сообщений, поступающих с консоли

Это диспетчерский процесс с номером **2N+3**. Он активизируется при получении от пульт-процессора прерывания по приему строки с операторской консоли. Производится анализ директивы, адресованной ОС, и ее параметров. При необходимости за-

пускаются служебные процессы или задачи пользователей. Текст директивы, повлекшей запуск процесса, передается этому процессу в его информационном поле. Что касается вывода на консоль, то он производится от имени процесса-отправителя, без дезактивации его.

Итак, мы перечислили все существующие в ОС МКБ диспетчерские процессы. Процессы-пользователи структурно устроены аналогично : у них то же виртуальное адресное пространство, та же структура информационного поля. Методы синхронизации и коммуникации процессов мы намерены изложить в последующих публикациях.

Л И Т Е Р А Т У Р А :

1.Емелин.И.А., Кадыков В.М., Левчановский Ф.В., Попов М.Ю., Сапожников А.П., Сапожникова Т.Ф., Силин И.Н. Принципы организации и архитектура процессора-эмулятора МКБ-8601. Дубна, ОИЯИ, Б1-11-88-442, 1988.

2.Емелин.И.А., Кадыков В.М., Левчановский Ф.В., Попов М.Ю., Сапожников А.П., Сапожникова Т.Ф., Силин И.Н. Архитектурные особенности МКБ-8601, интегральной ЭВМ ряда БЭСМ-6. Дубна, ОИЯИ, Р11-91-43, Дубна, 1991.

3.Давыдов А.Л., Сапожников А.П. Ассемблер МКБ-8601. Дубна, ОИЯИ, Б3-11-88-774. 1988.

4.Сапожникова Т.Ф. Программный эмулятор спецпроцессора МКБ-8601. Дубна, ОИЯИ, Б2-11-88-891. 1988.

5.Силин И.Н. Операционная система "Дубна" и проблемы создания математического обеспечения мощных ЭВМ. Автореферат диссертации. Дубна, ОИЯИ, 11-9248. 1975.

6.Силин И.Н., Федюнькин Е.Д., Универсальный алгоритм разделения времени. - Программирование, N4, 1980, стр.40-54.

7.А.Шоу. Логическое проектирование операционных систем. М., "Мир". 1981.

8.Тюрин В.Ф. Операционная система ДИСПАК. М., "Наука", 1985.

Рукопись поступила в издательский отдел
24 января 1991 года.