Обслуживание ввода-вывода

Лекция 7

Организация побайтного ввода-вывода

Во многих ОС имеется два типа устройств:

- устройства ввода-вывода блоками
- устройства неструктурированного или посимвольного ввода-вывода.

Примером блочных устройств ввода-вывода являются диски и ленты.

К устройствам посимвольного (побайтного) ввода вывода относятся практически все остальные устройства, в том числе дисплеи, клавиатуры и сетевое оборудование. Устройства ввода-вывода блоками могут также иметь интерфейс работы посимвольного ввода-вывода. В последнем случае для работы с устройством существуют два драйвера:

- блочный
- посимвольный.

Организация ввода-вывода с использованием каналов ввода-вывода

Каналы ввода-вывода (англ. IOC - input-output channel), далее КВВ, и интерфейсы обеспечивают взаимодействие центральных устройств машины и периферийных устройств.

КВВ — самостоятельные в логическом отношении устройства, которые работают под управлением собственных программ, находящихся в памяти.

В современных машинах КВВ называют *периферийными процессорами* или *процессорами ввода-вывода*.

КВВ и интерфейсы выполняют следующие функции:

1. Позволяют иметь машины с переменным составом периферийных устройств.
2. Обеспечивают параллельную работу периферийных устройств как между собой, так и по отношению к процессору.
3. Обеспечивают автоматическое распознавание и реакцию процессора на различные ситуации, возникающие в периферийных устройствах.

Организация ввода-вывода с использованием каналов ввода-вывода

Существует 3 вида КВВ:

1. *Мультиплексный канал.*

Сам канал быстродействующий, но обслуживает медленное периферийное устройство. При этом, подключившись к одному устройству, подаёт одно машинное слово, и после этого подключается к другому.

2. *Селекторный канал.*

Канал быстродействующий и обслуживает быстрые устройства. При этом подключившись к одному устройству, передаёт всю информацию, и после этого подключается к другому устройству.

3. *Блок-мультиплексорный канал.*

Подключившись к одному устройству, передаёт часть информации. После этого подключается к другому устройству.

Канальная программа

Команда канальной программы, или управляющее слово канала CCW, записывается в основной памяти по целочисленной границе двойного слова и имеет длину 64 бита.

При этом *команды канальной программы* не обязательно должны располагаться в одной области основной памяти: отдельные части программ могут быть в различных областях памяти. Адрес данных (биты 8 - 31) в команде Переход в канале воспринимается как адрес команды канала CCW, на которую осуществляется переход, поэтому он должен быть выровнен на целочисленную границу двойного слова. Другие поля команды игнорируются. Команда Переход в канале не может быть первой командой канальной программы, а две таких команды не могут выполняться друг за другом.

Каждый оператор CCW, записанный в исходной программе, при трансляции порождает одну *команду канальной программы*. Операторы CCW записываются в той последовательности, в какой должны располагаться команды канальной программы, и размещаются в программе обычно в области констант и данных. После этого центральный процессор освобождается для выполнения другой работы, а канал извлекает из основной памяти ЭВМ *команды канальной программы* одну за другой и выполняет их.

Канальная программа

Канал имеет свою систему канальных команд (CCW), из которых составляется канальная программа. В процессе обмена данными канал считывает *команды канальной программы* из ОП и обеспечивает их выполнение. CCW, записанный в исходной программе, при трансляции порождает одну команду канальной программы. Операторы CCW записываются в той последовательности, в какой должны располагаться *команды канальной программы*, и размещаются в программе обычно в области констант и данных.

В методе доступа EXCP внешними устройствами при организации ввода-вывода непосредственно управляет канал под воздействием канальной программы (см. разд. Канальная программа представляет собой одну или несколько последовательно связанных между собой канальных команд CCW. *Команды канальной программы* записываются в основной памяти но целочисленной границе двойного слова.

Канал, как и центральный процессор, имеет свою систему команд. Одна канальная программа определяет одну операцию обмена и записывается в последовательных ячейках основной памяти машины. В процессе обмена данными канал считывает *команды канальной программы* и обеспечивает их выполнение.

Канальная программа

Для реализации такого совмещения необходимо разделить функции управления центральным процессором и внешними устройствами. При этом устройства управления вводом-выводом получают только общие указания от ЦП и затем организуют работу внешних устройств автономно. Операция обмена информацией выполняется по схеме: оперативная память-канал ввода-вывода - контроллер (устройство управления внешними устройствами) - внешние устройства. Передача информации организуется ЦП и управляется с помощью команд ввода-вывода, *команд канальной программы*, приказов.

Накопитель на магнитной ленте ЕС-5010 подключается через УВУ ЕС-5511. Информация на ленту записывается с плотностью 8 или 32 бит / мм и считывается при прямом и обратном движении ленты. Лента содержит девять дорожек. Переключение с одной плотности на другую автоматическое, по *командам канальной программы*.

Канальная программа

Обмен данными между основной памятью и внешними устройствами осуществляется с помощью каналов. Канал представляет собой специализированное обрабатывающее устройство (процессор ввода-вывода), предназначенное для работы с устройствами ввода-вывода; как и центральный процессор, он имеет свою систему команд. Каждое из этих устройств - центральный процессор и канал - может выполнять только те команды, которые относятся к системе команд данного устройства. Из канальных команд (управляющих слов канала CCW) составляется канальная программа. Одна канальная программа определяет одну операцию обмена и записывается в последовательных ячейках основной памяти машины. В процессе обмена данными канал считывает *команды канальной программы* из основной памяти и обеспечивает их выполнение.

Вовлечение ОС в управление вводом-выводом

Подсистема ввода-вывода ОС при обмене данными с внешними устройствами должна решать ряд общих задач:

- Организация параллельной работы устройства ввода-вывода и процессора;
- Согласование скоростей обмена и кэширования данных;
- Разделение устройств и данных между процессами;
- Обеспечение удобного логического интерфейса между устройствами и остальной частью системы;
- Поддержка широкого спектра драйверов с возможностью простого включения в систему нового драйвера;
- Динамическая загрузка и выгрузка драйверов;
- Поддержка файловых систем;
- Поддержка синхронных и асинхронных операций ввода-вывода.

Очередь запросов на ввод-вывод

Ввод – это считывание данных с носителей информации в оперативную память. *Вывод* – перенос данных из ОП на носители информации.

Аппаратура различных ЭВМ существенно отличается по техническим и функциональным характеристикам, часто возникает потребность менять её количество и состав. В составе любой ОС имеется специальная подсистема управления аппаратурой ввода-вывода, избавляющая пользователя от необходимости знания множества деталей взаимодействия между программами и ПУ. Основной задачей этой подсистемы в мультипрограммном режиме является организация двусторонней высокоскоростной передачи данных между ОП и ПУ с целью достижения максимального перекрытия во времени работы аппаратуры ввода-вывода и ЦП. При этом реализуется принцип независимости от устройств, подразумевающий унифицированный интерфейс для доступа к различным по своим физическим характеристикам ПУ.

Очередь запросов на ввод-вывод

Несмотря на различия в подсистемах управления вводом-выводом, все ОС включают следующую концепцию: устройства ввода-вывода рассматриваются как совокупность аппаратурных процессоров, способных работать параллельно относительно друг друга и относительно ЦП. На таких процессорах развиваются внешние процессы, взаимодействующие между собой и с программными процессами, при этом скорости развития внешних и программных процессов могут различаться на порядок.

Система управления вводом-выводом (СУВВ) представляет собой один или несколько системных процессов, обеспечивающих информационное и управляющее взаимодействие между внутренними и внешними процессами. Через эту подсистему происходит инициация, управление и уничтожение внешних процессов. С точки зрения программных процессов пользователей СУВВ представляет собой программный интерфейс с необходимыми для них ПУ. В рамках этого интерфейса пользователь формирует запросы на выполнение следующих действий в отношении ПУ:

- операции чтения и записи данных в отношении адресуемого ПУ;
- операции управления устройством;
- операции по проверке состояния устройства.

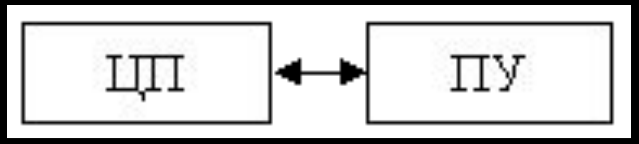
Очередь запросов на ввод-вывод

Большинство компонентов СУВВ «невидимы» для пользователя.

В зависимости от степени автономности от ЦП можно выделить два типа управления ПУ.

Прямой метод основан на непосредственной связи ЦП и ПУ и предполагает наличие в составе команд процессора специальных команд по инициированию работы, проверке готовности, останову, записи информации и т.д.

Методы управления периферийными устройствами



Прямой

Косвенный

Очередь запросов на ввод-вывод

Косвенный метод состоит в том, что между ЦП и ПУ помещается канал – специальный процессор, который фактически управляет вводом-выводом. С ЦП снимаются несвойственные ему функции по управлению ПУ, остаются лишь функции управления каналом. ЦП только инициирует ввод-вывод, а затем может выполнять свои программы (до момента окончания процесса ввода-вывода). При этом ЦП, канал и ПУ по мере развития внешнего процесса работают параллельно.

Для синхронизации параллельной работы ЦП и канала используют различные средства. В простейшем случае это флажок, в других случаях ЦП может быть доступна расширенная статусная информация о состоянии канала, контроллера и устройства. Такие средства предполагают некоторую периодичность проверок занятости канала со стороны ЦП.

Более совершенным механизмом является использование прерываний. Канал через систему прерываний прерывает работу ЦП всякий раз при завершении операции ввода-вывода или при возникновении ошибки. Здесь сигнал прерывания является по смыслу синхронизирующим, т.к. используется для оповещения определенного программного процесса о событии, которое произошло при работе канала или ПУ (например, при завершении печати страницы на принтере, ошибке записи на диск и т.д.).

Очередь запросов на ввод-вывод

При возникновении прерывания ЦП временно «отвлекается» от основной работы. В соответствии с централизованной схемой управления ПУ после определения причины прерывания управление передается системной программе управления вводом-выводом – *супервизору ввода-вывода*. При оповещении через прерывание о событии в некотором внешнем процессе супервизор ввода-вывода планирует и осуществляет через канал дальнейшие действия по организации ввода-вывода (обновление данных, инициирование следующей операции и т.д.).

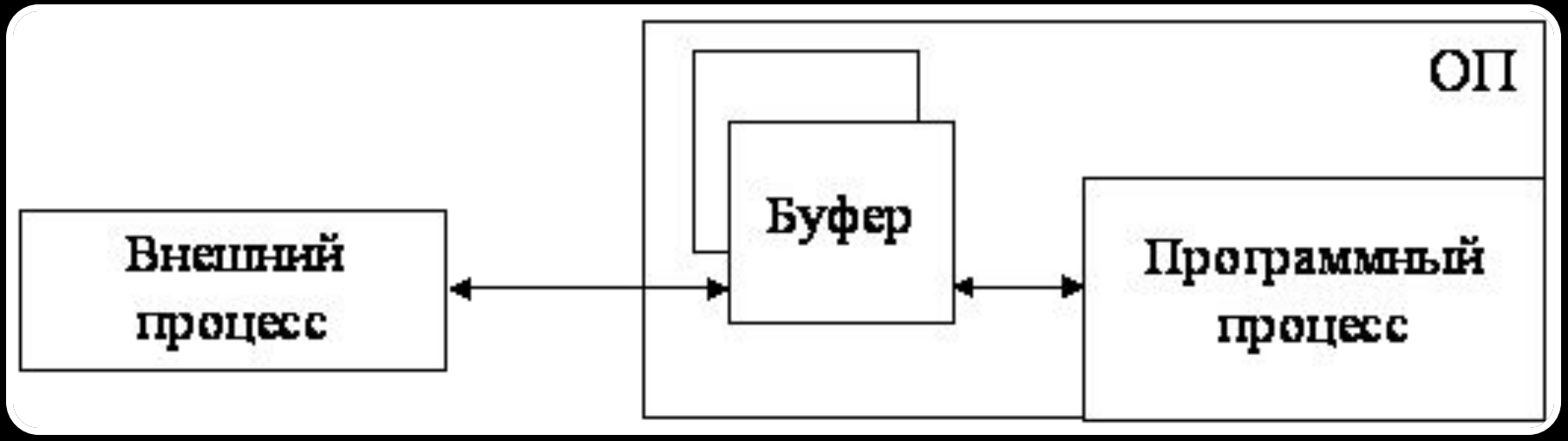
Помимо управления прерываниями супервизор ввода-вывода выполняет и другие функции, из которых, в первую очередь, рассмотрим сглаживание эффекта несоответствия скоростей между программными и внешними процессами с помощью одного или нескольких буферов, роль которых выполняют непрерывные области оперативной памяти (рис. 3).

Через буфер данные либо посылаются от некоторого программного процесса к адресуемому внешнему, либо от внешнего процесса передаются программному. На супервизор ввода-вывода возлагаются функции выделения и уничтожения буферов в оперативной памяти, определения их количества, размеров и назначения (для ввода или для вывода).

Супервизор ввода-вывода производит синхронизацию программных и внешних процессов, взаимодействующих через буфер – устраняет возможность одновременного обращения этих процессов к буферу.

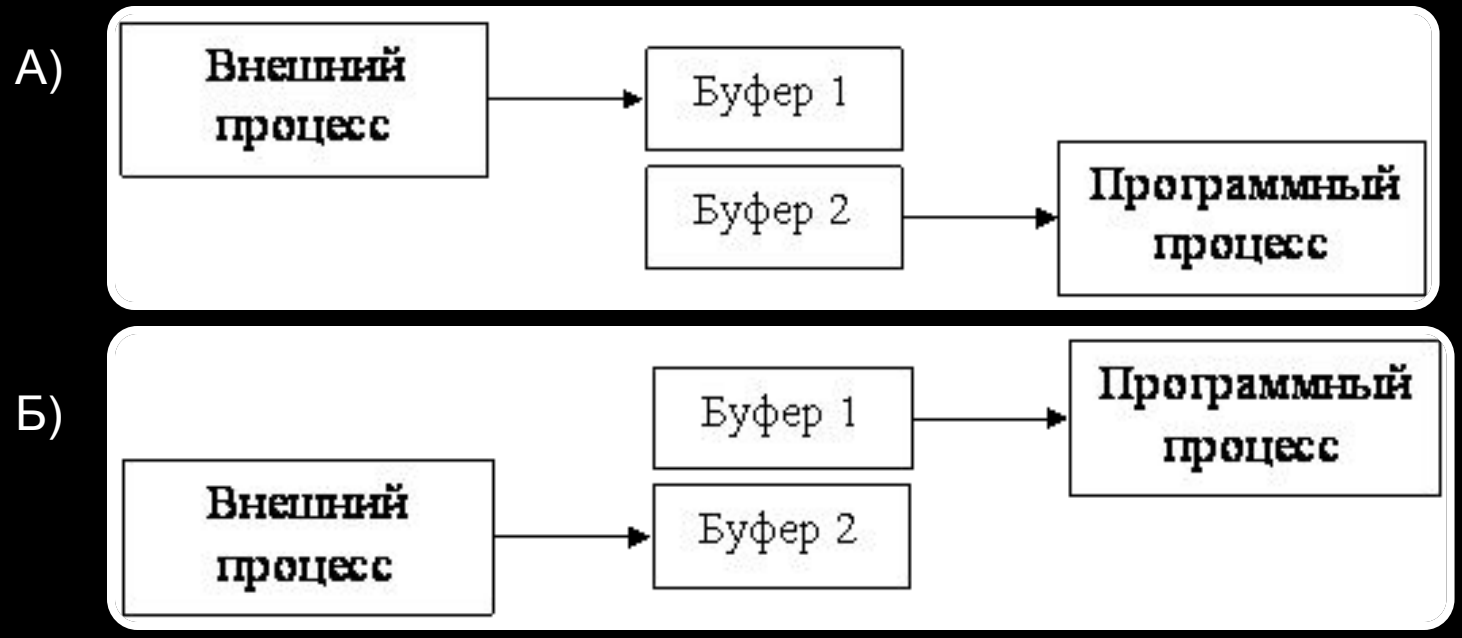
Очередь запросов на ввод-вывод

Использование буферов для организации информационного взаимодействия внешнего и программного процессов.



Очередь запросов на ввод-вывод

Для устранения задержек в ожидании наполнения буфера используется несколько буферов. Например, с точки зрения временных затрат для операции чтения рационально использовать два буфера. Пока один из них наполняется, другой в это время освобождается.



Поочередное использование двух буферов.

Очередь запросов на ввод-вывод

Определение размера буфера требует компромисса между эффективным использованием оперативной памяти и внешних запоминающих устройств (ВЗУ) при обмене данными между ними. Каналу и ВЗУ более выгодны большие порции информации, которые можно пересылать за одну операцию ввода-вывода, реализуемую в канале, и хранить непрерывными участками на внешних носителях устройств. Чем длинней блок информации, тем дольше канал работает автономно от ЦП и тем реже его «тревожит». Длинные блоки выгоднее для ВЗУ, так как в этом случае уменьшается число межблочных промежутков и количество памяти для их фиксации. Что касается ОП, то здесь требования противоположны – чем меньше размер блока (равный размеру буфера), тем лучше, так как больше памяти достанется программным процессам.

Не всегда выгодна наиболее быстрая передача данных из заполненного буфера внешнему процессу. Иногда такую передачу искусственно задерживают на достаточно продолжительные интервалы времени, формируя тем самым программный аналог *кэш-памяти* при работе с ВЗУ. Физическая запись на ВЗУ заполненных буферов осуществляется только тогда, когда возникает необходимость в свободном буфере, а его в текущий момент нет. Создаваемая таким образом задержка предполагает возможность обращения со стороны программных процессов к информации, временно хранимой в буфере. Если бы эта информация была «сброшена» на ВЗУ сразу после заполнения буфера, то обращение к ней потребовало бы обращения к устройству.

Очередь запросов на ввод-вывод

Другая важная функция супервизора ввода-вывода состоит в том, что по отношению к программным процессам он выступает в роли распределителя ресурсов, которыми в данном случае являются периферийные устройства. В соответствии с этим он должен воспринимать и интерпретировать запросы от программных процессов на использование ПУ и обеспечивать реализацию определенной стратегии распределения ПУ. Как 1правило, в запросах на ввод-вывод указываются символические имена или номера устройств. Супервизор ввода-вывода производит сопоставление символических имен устройств с конкретными их адресами, вызов и подготовку к работе соответствующих системных программ для обслуживания канала (канальных программ). В данном случае он выступает как транслятор запросов на ввод-вывод. При возникновении одновременно нескольких запросов от различных программных процессов супервизор помещает их в очередь к каналу, связанному с данным устройством. При освобождении канала супервизор извлекает из очереди запрос в соответствии с принятым правилом обслуживания очереди, после чего инициируется работа канала по программе, определенной в запросе.

Алгоритм обработки прерываний по вводу-выводу

Во всех компьютерах предусмотрен механизм, с помощью которого различные устройства (ввода-вывода, памяти) могут прервать нормальную работу процессора .Прерывания в основном предназначены для повышения эффективности работы. Например, большинство устройств ввода-вывода работают намного медленнее, чем процессор. Предположим, что процессор передает данные на принтер по схеме, показанной рис.1. После каждой операции процессор вынужден делать паузу и ждать, пока принтер не примет данные. Длительность этой паузы может быть в сотни и даже тысячи раз больше длительности цикла команды, в которой участвуют обращения к памяти. Подобное использование процессора является неэффективным.

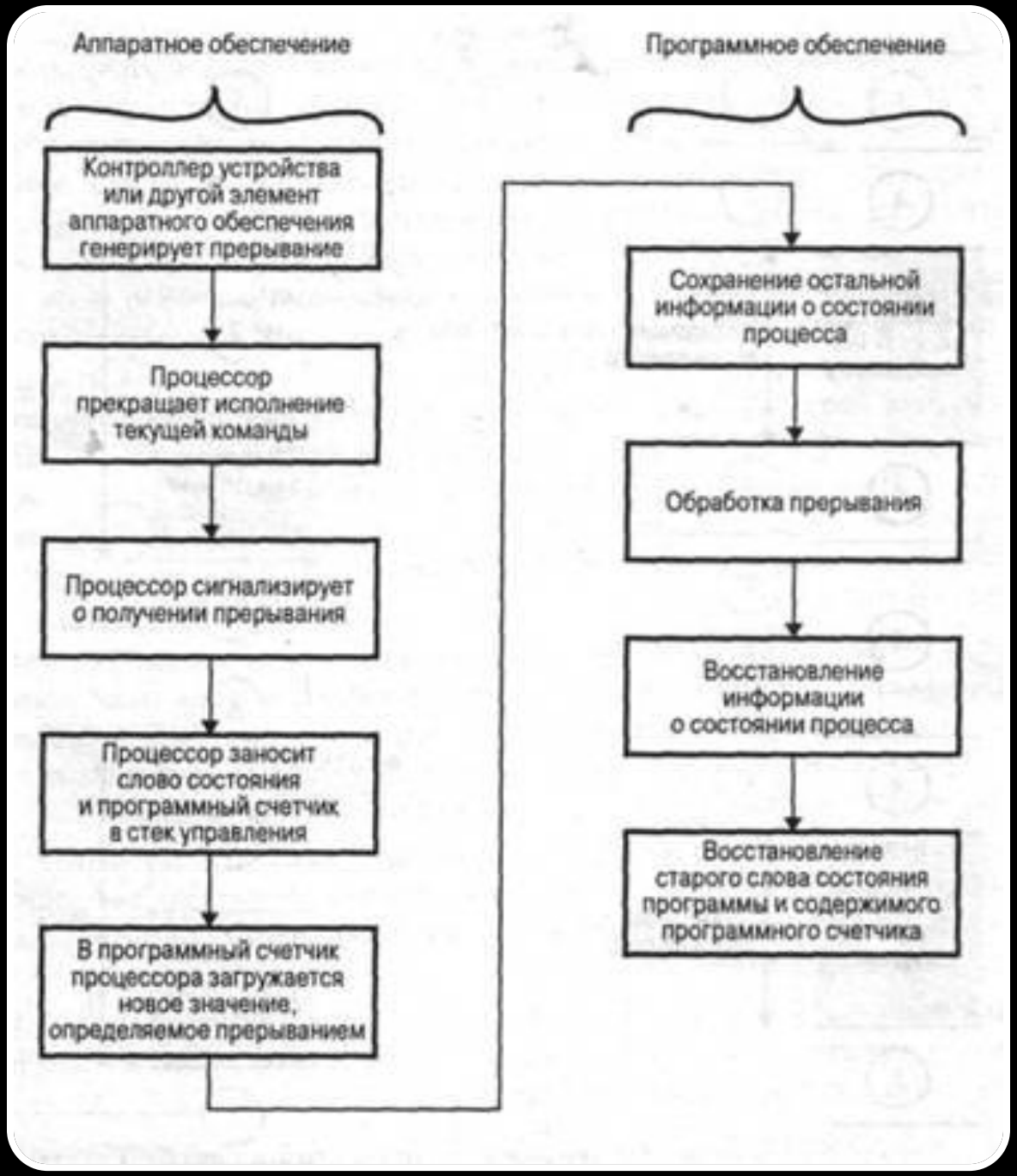
Программное прерывание	Генерируется в некоторых ситуациях, возникающих в результате выполнения команд. Такими ситуациями могут быть арифметическое переполнение, деление на ноль, попытка выполнить некорректную команду и ссылка на область памяти, доступ к которой пользователю запрещен
Прерывание по таймеру	Генерируется таймером процессора. Это прерывание позволяет операционной системе выполнять некоторые свои функции периодически, через заданные промежутки времени
Прерывание ввода-вывода	Генерируется контроллером ввода-вывода. Сигнализирует о нормальном завершении операции или о наличии ошибок
Аппаратное прерывание	Генерируется при возникновении таких аварийных ситуаций, как, например, падение напряжения в сети или ошибка контроля четности памяти

Алгоритм обработки прерываний по вводу-выводу

Прерывание вызывает ряд событий, которые происходят как в аппаратном, так и в программном обеспечении. На рисунке показана типичная последовательность этих событий. После завершения работы устройства ввода-вывода происходит следующее:

- Устройство посылает процессору сигнал прерывания.
- Перед тем как ответить на прерывание, процессор должен завершить исполнение текущей команды.
- Процессор производит проверку наличия прерывания, обнаруживает его и посылает устройству, приславшему это прерывание, уведомляющий сигнал об успешном приеме. Этот сигнал позволяет устройству снять свой сигнал прерывания.

Далее в программный счетчик процессора загружается адрес входа программы обработки прерываний, которая отвечает за обработку данного прерывания. В зависимости от архитектуры компьютера и устройства операционной системы может существовать как одна программа для обработки всех прерываний, так может быть и своя программа обработки для каждого устройства и каждого типа прерываний. Если для обработки прерываний имеется несколько программ, то процессор должен определить, к какой из них следует обратиться. Эта информация может содержаться в первоначальном сигнале прерывания; в противном случае для получения необходимой информации процессор должен по очереди опросить все устройства, чтобы определить, какое из них отправило прерывание.


Аппаратное обеспечение
Программное обеспечение
Контроллер устройства или другой элемент аппаратного обеспечения генерирует прерывание
Процессор прекращает исполнение текущей команды
Процессор сигнализирует о получении прерывания
Процессор заносит слово состояния и программный счетчик в стек управления
В программный счетчик процессора загружается новое значение, определяемое прерыванием
Сохранение остальной информации о состоянии процесса
Обработка прерывания
Восстановление информации о состоянии процесса
Восстановление старого слова состояния программы и содержимого программного счетчика

Алгоритм обработки прерываний по вводу-выводу

Как только в программный счетчик загружается новое значение, процессор переходит к следующему циклу команды, приступая к ее извлечению из памяти. Так как команда извлекается из ячейки, номер которой задается содержимым программного счетчика, управление переходит к программе обработки прерываний. Исполнение этой программы влечет за собой следующие операции.

Содержимое программного счетчика и слово состояния прерываемой программы уже хранятся в системном стеке. Однако это еще не вся информация, имеющая отношение к состоянию исполняемой программы. Например, нужно сохранить содержимое регистров процессора, так как эти регистры могут понадобиться обработчику прерываний. Поэтому необходимо сохранить всю информацию о состоянии программы. Обычно обработчик прерываний начинает свою работу с записи в стек содержимого всех регистров.

Теперь обработчик прерываний может начать свою работу. В процесс обработки прерывания входит проверка информации состояния, имеющая отношение к операциям ввода-вывода или другим событиям, вызвавшим прерывание. Сюда может также входить пересылка устройствам ввода-вывода дополнительных инструкций или уведомляющих сообщений.

После завершения обработки прерываний из стека извлекаются сохраненные ранее значения, которые вновь заносятся в регистры, возобновляя таким образом то состояние, в котором они пребывали до прерывания.

Последний этап — восстановление из стека слова состояния программы и содержимого программного счетчика. В результате следующей будет выполняться команда прерванной программы.

Алгоритм обработки прерываний по вводу-выводу

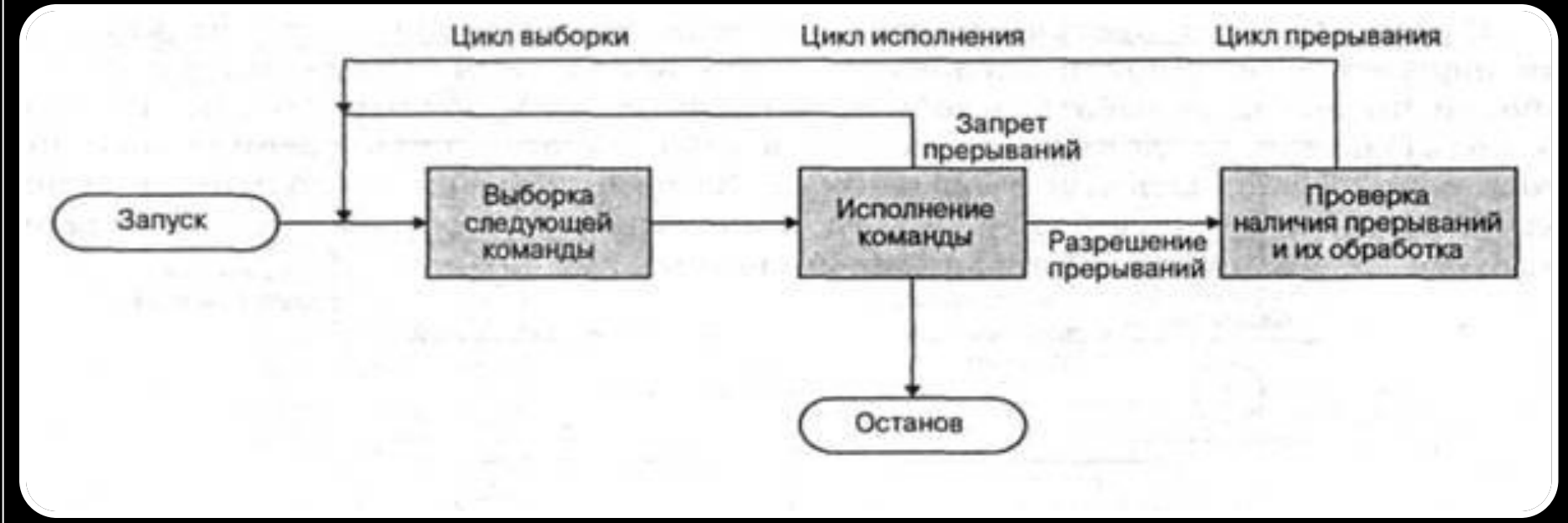
Чтобы согласовать прерывание с программой, в цикл команды добавляется цикл. В цикле прерывания процессор проверяет наличие сигналов прерываний, свидетельствующих о происшедших прерываниях. При поступлении прерывания процессор приостанавливает работу с текущей программой и выполняет *обработчик прерываний*.

Обработчики прерываний обычно входят в состав операционной системы. Как правило, эти программы определяют природу прерывания и выполняют необходимые действия. Например, в используемом примере обработчик должен определить, какой из контроллеров ввода-вывода сгенерировал прерывание; кроме того, он может передавать управление программе, которая должна вывести данные на устройство ввода-вывода. Когда обработчик прерываний завершает свою работу, процессор возобновляет выполнение программы пользователя с того места, где она была прервана.

Процесс включает в себя некоторые непроизводительные затраты.

Для определения природы прерывания и принятия решения о последующих действиях обработчик прерываний должен выполнить дополнительные команды. Тем не менее, ввиду того что для ожидания завершения операций ввода-вывода потребовался бы сравнительно большой отрезок времени, с помощью прерываний процессор можно использовать намного эффективнее.

Алгоритм обработки прерываний по вводу-выводу



Поддержка широкого спектра драйверов с возможностью простого включения в систему нового драйвера

Достоинством подсистемы ввода-вывода операционной системы является разнообразие устройств, поддерживаемых данной ОС.

Для создания драйверов необходимо наличие удобного и открытого интерфейса между драйверами и другими компонентами ОС.

Драйвер взаимодействует, с одной стороны, с модулями ядра ОС, а с другой стороны – с контроллерами внешних устройств. Драйвер имеет два интерфейса:

1. DKI (driver kernel interface)
2. DDI (driver device interface).

Динамическая загрузка и выгрузка драйверов

Другой проблемой работы с устройствами ввода-вывода является проблема включения драйвера в состав работающей ОС – динамическая загрузка/выгрузка драйверов.

Способность системы автоматически загружать и выгружать из оперативной памяти требуемый драйвер повышает универсальность ОС.

Альтернативой динамической загрузке драйверов при изменении текущей конфигурации внешних устройств является повторная компиляция кода ядра с требуемым набором драйверов. Пример – некоторые версии UNIX.

Поддержка файловых систем

Внешняя память вычислительной системы представляет собой периферийные устройства, на которых хранится большая часть пользовательской информации и системных данных.

Для организации хранения информации на внешних носителях используется файловая модель.

Для обеспечения доступа к данным используется специальный программный слой, обеспечивающий поддержку работы с конкретной файловой системой – драйверы файловой системы.

Для обеспечения возможности работы с несколькими файловыми система применяется подход, основанный применении специального слоя, с которым взаимодействуют приложения ОС – например, слой VFS (virtual file system) в некоторых версиях UNIX.

Поддержка синхронных и асинхронных операций ввода-вывода

Операции ввода-вывода по отношению к программному приложению выполняются в синхронном или асинхронном режимах.

Синхронный режим – приложение приостанавливает свою работу и ждет отклика от устройства.

Асинхронный режим – приложение продолжает работу, параллельно с ожиданием отклика от устройства.

Операционные системы для разных приложений должны обеспечить синхронную и асинхронную работу с устройствами.

Многослойная модель подсистемы ввода-вывода

Менеджеры ввода-вывода

Для координации работы драйверов в подсистеме ввода-вывода выделяется специальный модуль, называемый менеджером ввода-вывода.

Верхний слой менеджера составляют системные вызовы ввода-вывода, которые получают запросы от приложений и переадресуют их определенным драйверам.

Нижний слой реализует взаимодействие с контроллерами внешних устройств, экранируя драйверы от особенностей аппаратной платформы компьютера.

Еще одна функция менеджера ввода-вывода – организация взаимодействия модулей ввода-вывода с модулями других подсистем (управление процессами, виртуальной памятью и т. д.).

Специальные файлы

Для унификации операций и структуризации программного обеспечения ввода-вывода устройства рассматриваются как некоторые специальные (виртуальные) файлы.

Такой подход позволяет использовать общий набор базовых операций ввода-вывода для любых устройств, экранировать специфику устройства.

Например, в операционных системах семейства UNIX, специальные файлы помещаются в каталог /dev. При появлении нового устройства администратор имеет возможность создать новую запись с помощью команды mknod.