Управление вводом/выводом в операционных системах

Лектор: доц., к.т.н.,
доцент кафедры ИСТАС
Иванов Н.А.

2020 г.

Особенности управления вводом/выводом в операционных системах

Управление операциями ввода/вывода считается одним из самых важных и в то же время одним из сложных разделов процесса разработки операционной системы.

Сложность объясняется наличием большого числа различных по типу и назначению устройств ввода/вывода, работу которых должна поддерживать операционная система.

Основная задача разработчиков подсистемы управления вводом/выводом

Разработчикам операционной системы необходимо решить трудную задачу:

организовать эффективное управление различными внешними устройствами,

обеспечив удобный и эффективный виртуальный интерфейс устройств ввода/вывода, дающий возможность прикладным программистам легко считывать или сохранять данные, не обращая внимания на особенности отдельных устройств и проблемы распределения устройств между выполняющимися процессами.

Основные требования к подсистеме ввода/вывода

Система ввода/вывода должна быть универсальной, объединяющей в одной модели все разнообразие внешних устройств от простой мыши и клавиатур до принтеров, графических дисплеев, дисковых накопителей и даже сетей.

Важно обеспечить доступ к внешним устройствам для множества параллельно выполняющихся процессов, сведя к минимуму вероятность возникновения проблем, связанных с использованием последними устройств ввода/ вывода.

Определяющий принцип при проектировании подсистемы управления вводом/выводом

Любые операции по управлению вводом/выводом объявляются привилегированными и могут выполняться только кодом самой операционной системы

Почему вводом-выводом должна управлять операционная система

Причины, по которым нельзя разрешать каждой отдельной пользовательской программе обращаться к внешним устройствам непо средственно

Необходимость избавить программы ввода-вывода от ошибок.

Необходимость разрешать возможные конфликты в доступе к устройствам ввода-вывода.

Необходимость увеличить эффективность использования ресурсов ввода-вывода.

Почему вводом-выводом должна управлять операционная система

Необходимость избавить программы ввода-вывода от ошибок

Ошибки в программах ввода-вывода могут привести к краху всех вычислительных процес сов, так как часть операций ввода-вывода требуются самой операционной системе.

В ряде операционных систем системный ввод-вывод имеет существенно более высокие привилегии, чем ввод-вывод задач пользователя.

Поэтому системный код, управляющий операциями ввода-вывода, очень тщательно отлаживается и оптимизируется для повышения надежности вычислений и эффективности использования оборудования.

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ
СТРОИТЕЛЬНЫЙ УНИВЕРСИТЕТ

Почему вводом-выводом должна управлять операционная система

Необходимость разрешать возможные конфликты в доступе к устройствам ввода-вывода

Задача 1 →  ← Задача 2

```
bo Assembler  Version 4.1 02/03/98 21:23:43   Page 1
_6_1.asm
;———Prg_6_1.asm————————
;Программа преобразования двузначного шестнадцатеричного числа
;в символьном виде в двоичное представление.
;Вход: исходное шестнадцатеричное число из двух цифр,
;вводится с клавиатуры.
;Выход: результат помещается
;в регистр al.
;————————————————
0000data segment para public "data" ;сегмент данных
0000 82 A2 A5 A4 A8 E2 A5 message db "Введите две шестнадцатеричные
цифры,$"
20 A4 A2 A5 20 E8   A5+
E1 E2 AD A0 A4 D5   A0+
E2 A5 E0 A8 E7 AD   EB+
A5 20 E6 A8 E4 E0   EB+
2C 24
0025 data ends
0000 stk  segment   stack
0000 0100*(3F)  db  256 dup ("?")       ;сегмент стека
```

Пусть две параллельно выполняющиеся программы пытаются вывести на печать результаты своей работы. Если не предусмотреть внешнего управления устройством печати, то в результате мы можем получить абсолютно нечитаемый текст, так как каждая программа будет время от времени выводить свои данные, перемежающиеся с данными от другой программы.

Еще один пример: одной программе необходимо прочитать данные с одного сектора магнитного диска, а другой - записать результаты в другой сектор того же накопителя. Если операции ввода-вывода не будут отслеживаться каким-то третьим (внешним) процессом-арбитром, то после позиционирования магнитной головки для первой задачи может тут же прийти команда позиционирования головки для второй задачи, и обе операции ввода-вывода не смогут выполниться корректно.

Почему вводом-выводом должна управлять операционная система

Необходимость увеличить эффективность использования ресурсов ввода-вывода

Например, у накопителя на магнитных дисках время подвода головки чтения/записи к необходимой дорожке и время обращения к определенному сектору могут значительно (до тысячи раз) превышать время пересылки данных.

В результате, если задачи по очереди обращаются к цилиндрам, далеко отстоящим друг от друга, то полезная работа, выполняемая накопителем, может быть существенно снижена.

Операционная система может упорядочить проведение операций ввода-вывода с целью уменьшения времени перемещений головок чтения/записи.

Поддержка принципа «ввод/вывод – работа ОС» наличие двух режимов работы

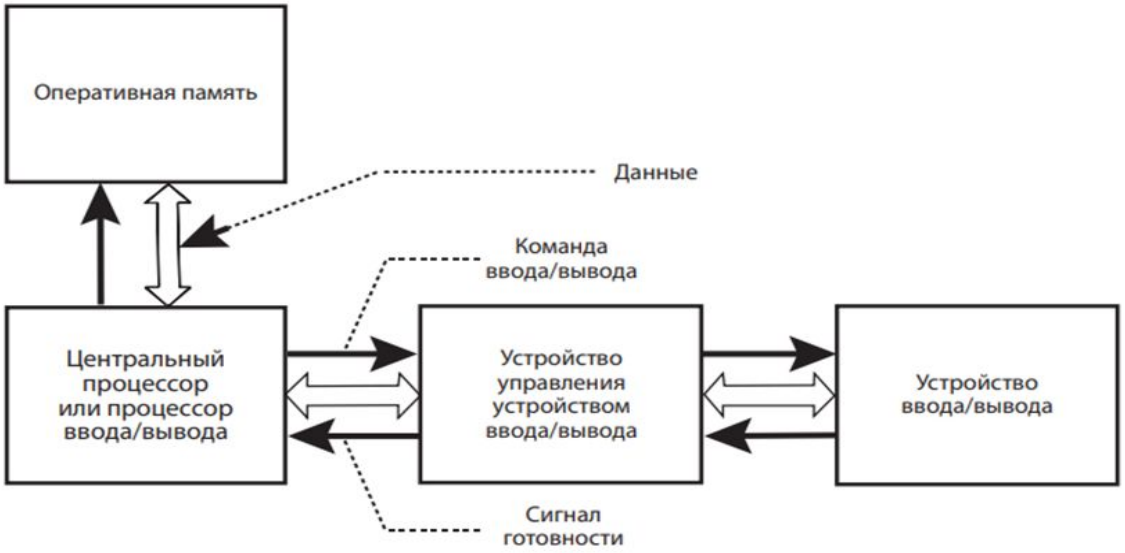
Режим пользователя	Режим супервизора /привилегированный режим /режим ядра
выполнение команд ввода/вывода запрещено	выполнение команд ввода/вывода разрешено
Обращение к командам ввода/вывода в прикладных программах вызывает прерывание, и управление через механизм прерываний передается коду операционной системы	

Режимы ввода/вывода

Режим обмена с опросом готовности устройства ввода/вывода

Режим обмена с прерываниями

Управление вводом/выводом с опросом готовности (Синхронный режим ввода-вывода)



Пусть центральный процессор посылает устройству управления команду, требующую, чтобы определенное внешнее устройство выполнило некоторое действие.

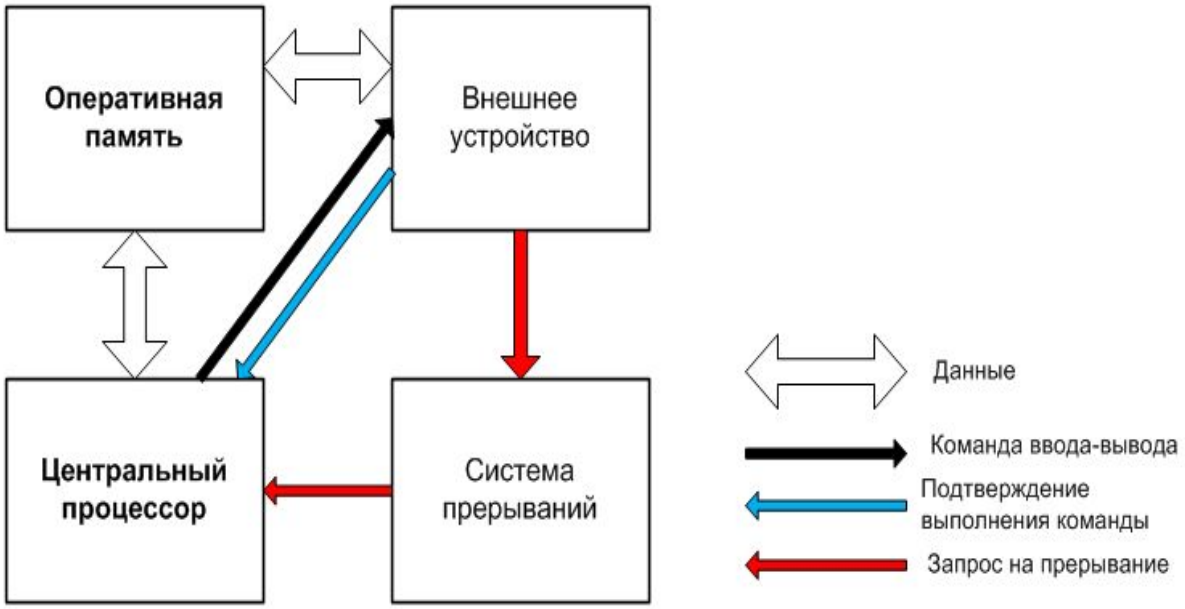
Устройство управления внешним устройством (УУВУ) исполняет команду, преобразуя сигналы, понятные ему и центральному процессору, в сигналы, понятные внешнему устройству.

После выполнения команды внешнее устройство выдает *сигнал готовности*, который сообщает процессору о том, что можно передавать очередную команду для продолжения обмена данными.

Из-за того, что быстродействие внешнего устройства на несколько порядков ниже быстродействия центрального процессора, последнему приходится долго ожидать сигнала готовности, постоянно опрашивая соответствующую линию интерфейса на наличие или отсутствие этого сигнала. Длительное ожидание связано с тем, что нет смысла отправлять внешнему устройству очередную команду, не дождавшись ответа об исполнении предыдущей команды.

В режиме опроса готовности процессор, обрабатывая код драйвера, управляющего процессом обмена данными с ВУ, как раз и выполняет в цикле команду «проверить наличие сигнала готовности», нерационально расходуя процессорное время, которое можно было бы потратить на решение других задач.

Управление вводом/выводом по прерываниям (Асинхронный режим ввода-вывода)



Центральный процессор посылает внешнему устройству команду, требующую, чтобы определенное внешнее устройство выполнило некоторое действие, и на время забывает об этом устройстве, перейдя на обработку другой программы.

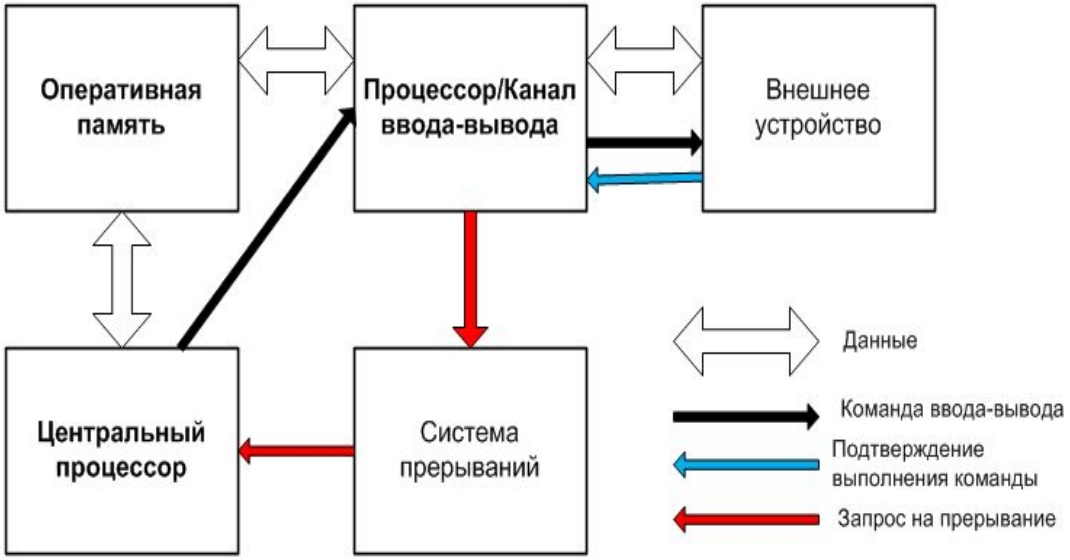
Выполнив операцию ввода/вывода, внешнее устройство должно оповестить процессор об этом, однако напрямую оно к процессору обратиться не может.

Для организации оповещения внешнее устройство через систему прерываний выдает запрос на прерывание работы процессора. Именно такой запрос и является для центрального процессора сигналом о выполнении операции ввода/вывода.

Для того чтобы процессор, выдав внешнему устройству очередную команду на прием или передачу данных, не потерял этого устройства после переключения на выполнение других программ, может быть запущен отсчет времени, в течение которого устройство обязательно должно выполнить команду и прислать сигнал запроса на прерывание. Максимальный интервал времени, в течение которого внешнее устройство или его контроллер должен выдать сигнал запроса на прерывание, принято называть ***тайм-аутом***.

Если после выдачи устройству очередной команды прошло больше времени, чем было указано в тайм-ауте, а устройство так и не ответило, то делается вывод о том, что связь с устройством утеряна и им больше невозможно управлять. Задача, запросившая утерянную операцию ввода/вывода, получают соответствующее диагностическое сообщение.

Управление вводом/выводом по прерываниям с использованием каналов вода-вывода



В описанном выше случае центральный процессор хотя и получает возможность заниматься основной работой – вычислениями, но он всё-таки вынужден время от времени прерывать свою работу, чтобы отправить внешнему устройству очередную команду на ввод-вывод или проанализировать сложившуюся во время выполнения очередной команды ситуацию. С ростом числа внешних устройств, задействованных в работе приложений, таких отвлечений процессора становится всё больше и больше и эффективность его работы значительно снижается.

Для решения указанной проблемы было предложено включить в состав вычислительной системы специализированные процессоры - *каналы ввода-вывода*, на которые была возложена задача управления работой внешних устройств.

Канал ввода-вывода взаимодействует с устройством ввода-вывода на протяжении выполнения *операции ввода-вывода*. Операция ввода-вывода или *канальная программа* состоит из отдельных *канальных команд*, которые канал поручает выполнять внешнему устройству. Канал контролирует результат выполнения каждой такой команды, требуя от устройств ввода-вывода передачи служебной информации о состоянии обмена.

При завершении канальной программы или при возникновении ситуации, не позволяющей выполнить ту или иную канальную команду, канал ввода-вывода через систему прерываний информирует об этом центральный процессор. При такой организации ввода-вывода роль центрального процессора состоит в инициализации операции ввода-вывода и получении сообщений от канала ввода-вывода о её завершении или аварийном прекращении.

Использование блокирующих (синхронных) и неблокирующих (асинхронных) передач

Большинство операций физического ввода-вывода выполняется асинхронно - процессор начинает передачу и переходит на другую работу, пока не наступает прерывание.
Пользовательские программы намного легче писать, если операции ввода-вывода блокирующие - после команды READ программа автоматически приостанавливается до тех пор, пока данные не попадут в буфер программы.
ОС выполняет операции ввода-вывода асинхронно, но представляет их для пользовательских программ в синхронной форме.

Организация программного обеспечения ввода-вывода

Основная идея организации программного обеспечения ввода-вывода состоит в разбиении его на несколько уровней, причем нижние уровни обеспечивают экранирование верхних уровней от особенностей аппаратуры, а те, в свою очередь, обеспечивают удобный интерфейс для пользователей.

Приложения, не имея возможности напрямую взаимодействовать с устройствами ввода-вывода, при необходимости выполнения операций ввода-вывода обращаются с запросами к операционной системе.

При этом используются универсальные библиотечные функции ввода-вывода системы программирования, которые транслируются в тот или иной системный вызов.

Независимый от устройств слой операционной системы

Большая часть программного обеспечения ввода-вывода должна быть *независимой от устройств*.

Типичными функциями для независимого от устройств слоя являются:

- обеспечение общего интерфейса к драйверам устройств,
- именование устройств,
- обеспечение независимого размера блока,
- буферизация,
- распределение памяти на блок-ориентированных устройства
- распределение и освобождение выделенных устройств,
- уведомление об ошибках.

Ключевую роль в подсистеме ввода-вывода играет *супервизор ввода-вывода* - компонента операционной системы, осуществляющая управление вводом-выводом.

Супервизор ввода-вывода и его задачи

1. Получает запросы на ввод-вывод от прикладных задач или от других модулей операционной системы.
2. Вызывает соответствующие распределители каналов и контроллеров, планирует ввод-вывод (определяет очередность предоставления устройств ввода-вывода задачам, затребовавшим эти устройства). Запрос на ввод-вывод либо тут же выполняется, либо ставится в очередь на выполнение.
3. Инициирует операции ввода-вывода (передает управление соответствующим драйверам) и в случае управления вводом-выводом с использованием прерываний предоставляет процессор диспетчеру задач с тем, чтобы передать его первой задаче, стоящей в очереди на выполнение.
4. При получении сигналов прерываний от устройств ввода-вывода идентифицирует эти сигналы и передает управление соответствующим программам обработки прерываний.
5. Осуществляет передачу сообщений об ошибках, если таковые происходят в процессе управления операциями ввода-вывода.
6. Посылает сообщения о завершении операции ввода-вывода запросившей эту операцию задаче и снимает ее с состояния ожидания ввода-вывода, если задача ожидала завершения операции.

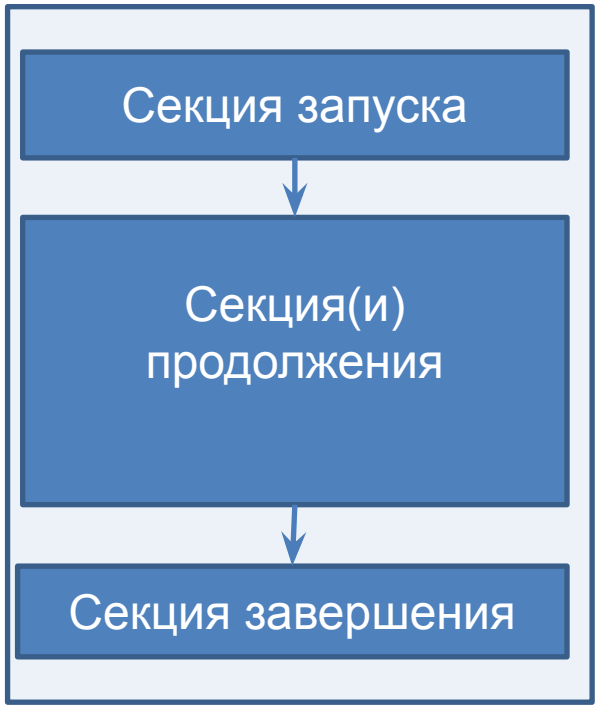
Драйверы устройств

Весь ***зависимый*** от устройства код помещается в драйвер устройства.
Каждый драйвер управляет устройствами одного типа или, может быть, одного класса.
В операционной системе только драйвер устройства знает о конкретных особенностях какого-либо устройства.
Например, только драйвер диска имеет дело с дорожками, секторами, цилиндрами, временем установления головки и другими факторами, обеспечивающими правильную работу диска.
Драйвер устройства принимает запрос от модулей программного слоя и решает, как его выполнить.

Работа драйвера устройств

Если драйвер был свободен во время поступления запроса, то он начинает выполнять запрос немедленно. Если же он был занят обслуживанием другого запроса, то вновь поступивший запрос присоединяется к очереди уже имеющихся запросов, и он будет выполнен, когда наступит его очередь.
Первый шаг в реализации запроса ввода-вывода состоит в преобразовании его из абстрактной формы в конкретную.
Например, при обработке типичным запроса на чтение n блоков данных с магнитного носителя для дискового драйвера выполняется преобразование номеров блоков в номера цилиндров, головок, секторов, проверку, работает ли мотор, находится ли головка над нужным цилиндром.
Другими словами, драйвер должен решить, какие операции контроллера нужно выполнить и в какой последовательности.
После передачи команды контроллеру драйвер должен решить, блокировать ли себя до окончания заданной операции или нет.
Если операция занимает значительное время, как при печати некоторого блока данных, то драйвер блокируется до тех пор, пока операция не завершится, и обработчик прерывания не разблокирует его. Если команда ввода-вывода выполняется быстро (например, прокрутка экрана), то драйвер ожидает ее завершения без блокирования.

Упрощенная структура драйвера устройств



Секция запуска инициирует операцию ввода-вывода. Эта секция запускается для включения устройства ввода-вывода или просто для инициализации очередной операции ввода-вывода.

Секция продолжения осуществляет основную работу по передаче данных, является основным обработчиком прерывания. Поскольку используемый интерфейс может потребовать для управления вводом-выводом несколько последовательностей управляющих команд, а сигнал прерывания у устройства, как правило, только один, то после выполнения очередной секции продолжения супервизор прерываний при следующем сигнале готовности должен передать управление другой секции. Это делается путем изменения адреса обработки прерывания после выполнения очередной секции, а если имеется только одна секция продолжения, она сама передает управление в ту или иную часть кода подпрограммы обработки прерывания.

Секция завершения обычно выключает устройство ввода-вывода или просто завершает операцию.

Обработка ошибок ввода-вывода

Важным вопросом для программного обеспечения ввода-вывода является обработка ошибок.
Принцип работы: «Ошибки следует обрабатывать как можно ближе к аппаратуре. И только если нижний уровень не может справиться с ошибкой, он сообщает об ошибке верхнему уровню».

Пример.
Если контроллер обнаруживает ошибку чтения, то он должен попытаться ее скорректировать. Многие ошибки могут исчезать при повторных попытках выполнения операций ввода-вывода, например, ошибки, вызванные наличием пылинок на головках чтения или на диске. Если же контроллеру не удается решить проблему, то исправлением ошибок должен заняться драйвер устройства.

Организация программного обеспечения ввода-вывода

Ключевым принципом является независимость от устройств.
Вид программы не должен зависеть от того, читает ли она данные с гибкого диска или с жесткого диска.

Очень близкой к идее независимости от устройств является идея единообразного именования, то есть для именования устройств должны быть приняты единые правила.

Информация, необходимая для управления вводом-выводом

Для управления операциями ввода-вывода и отслеживания состояния всех ресурсов, занятых в обмене данными, операционная система должна иметь соответствующие информационные структуры отображающие следующую информацию:

- состав устройств ввода-вывода и способы их подключения;
- аппаратные ресурсы, закрепленные за имеющимися в системе устройствами ввода-вывода;
- логические (символьные) имена устройств ввода-вывода, используя которые вычислительные процессы могут запрашивать те или иные операции ввода-вывода;
- адреса размещения драйверов устройств ввода-вывода и области памяти для хранения текущих значений переменных, определяющих работу с этими уст ройствами
- области памяти для хранения информации о текущем состоянии устройства ввода-вывода и параметрах, определяющих режимы работы устройства;
- данные о текущем процессе, который работает с данным устройством;
- адреса тех областей памяти, которые содержат данные, собственно и участвующие в операциях ввода-вывода (получаемые при операциях ввода данных и выводимые на устройство при операциях вывода данных).

Такими структурами являются системные *таблицы ввода-вывода*.

Основные системные таблицы ввода-вывода

Для различных операционных систем количество, назначение и состав каждой таблицы могут сильно отличаться.
В некоторых операционных системах вместо таблиц создаются списки, хотя использование статических структур данных для организации ввода-вывода, как правило, приводит к более высокому быстродействию.

Исходя из принципа управления вводом-выводом исключительно через супервизор операционной системы и учитывая, что драйверы устройств ввода-вывода используют механизм прерываний для установления обратной связи центральной части с внешними устройствами, можно сделать вывод о необходимости наличия по крайней мере **трех** системных таблиц:

1. *таблица оборудования* (Unit Table)
2. *таблица виртуальных логических устройств* (Device Reference Table, DRT)
3. *таблица прерываний (Interrupt Table)*

Таблица оборудования (Unit Table)

строка таблицы — блок управления устройством ввода-вывода (Unit Control Block, UCB), как правило, содержит следующую информацию об устройстве:

- тип устройства, его конкретная модель и характеристики устройства;
- способ подключения устройства (через какой интерфейс, к какому разъему, какие порты и линия запроса прерывания используются и т. д.);
- номер и адрес канала (и подканала), если такие используются для управления устройством;
- информация о драйвере, который должен управлять этим устройством, адреса секции запуска и секции продолжения драйвера;
- информация о том, используется или нет буферизация при обмене данными с устройством, «имя» (или просто адрес) буфера, если такой выделяется из системной области памяти;
- установка тайм-аута и ячейки для счетчика тайм-аута;
- состояние устройства;
- поле указателя на список задач, ожидающих устройство.

Таблица виртуальных логических устройств (Device Reference Table, DRT)

Предназначена для реализации принципа виртуализации устройств ввода-вывода — принципа независимости от устройства, за счет установления связи между виртуальными (логическими) устройствами и реальными устройствами, описанными посредством таблицы оборудования.

Позволяет супервизору перенаправить запрос на ввод-вывод из приложения в те программные модули и структуры данных, которые (или адреса которых) хранятся в соответствующем элементе первой таблицы.

Логическое устройство	Номер строки таблицы оборудования (физическое устройство)

Таблица прерываний (Interrupt Table)

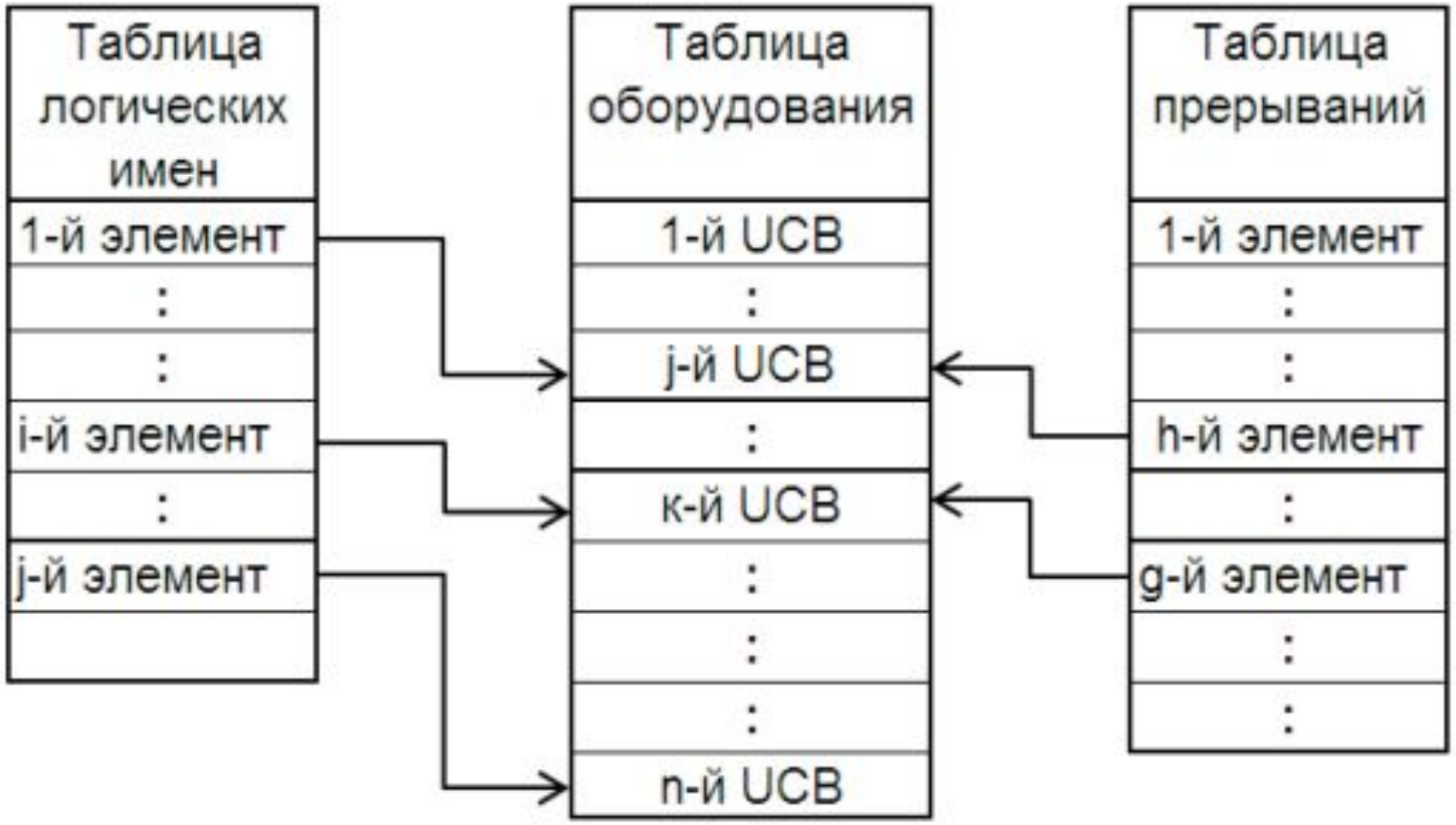
Необходима для организации обратной связи между центральным процессором и устройствами ввода-вывода.

Указывает для каждого сигнала запроса на прерывание тот элемент UCB, который сопоставлен данному устройству. Каждое устройство либо имеет свою линию запроса на прерывание, либо разделяет линию запроса на прерывание с другими устройствами, но при этом имеется механизм второго уровня адресации устройств ввода-вывода.

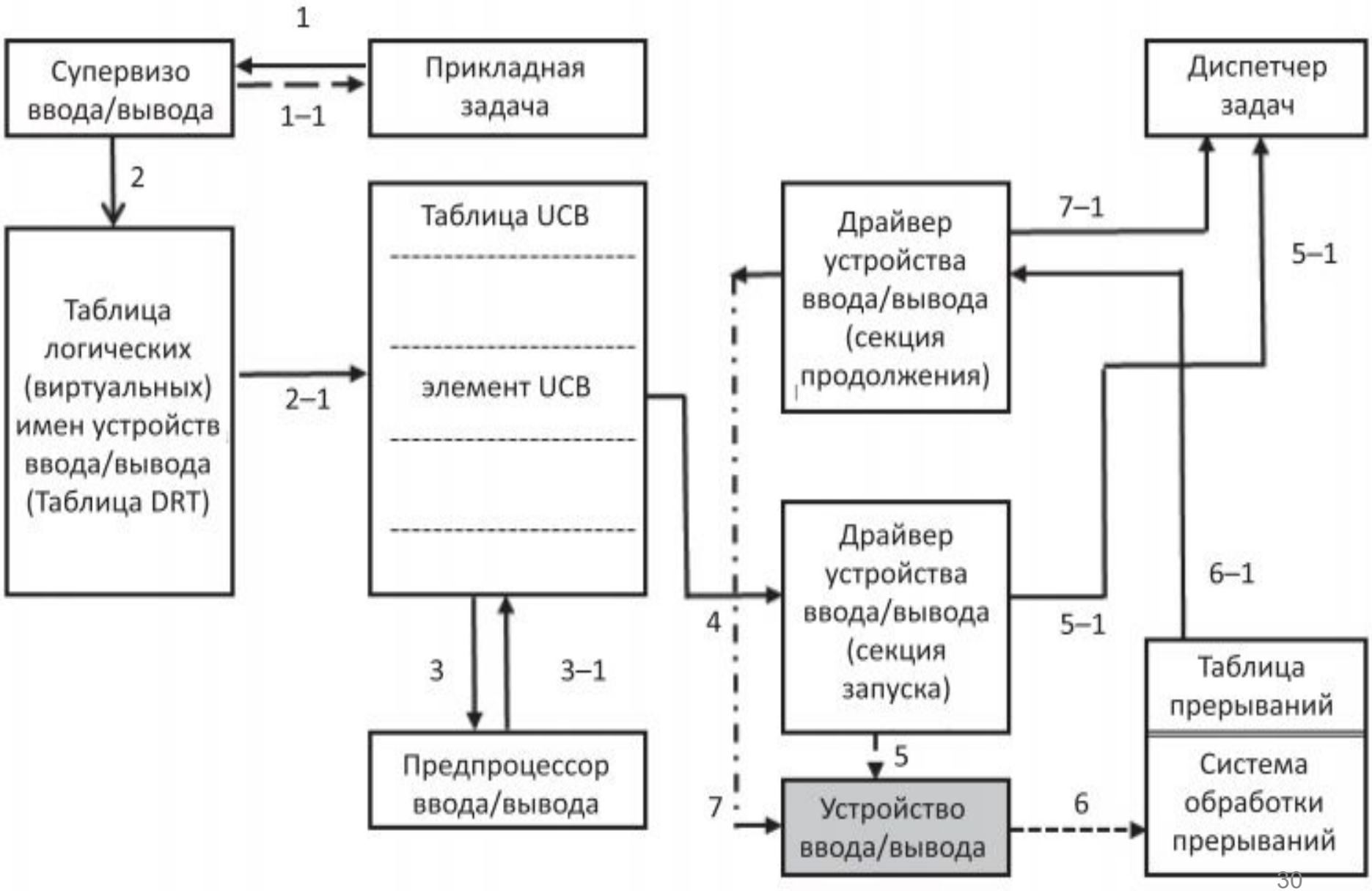
Таким образом, *таблица прерываний отображает связи между сигналами запроса на прерывания и самими устройствами ввода-вывода*.

Номер запроса на прерывание (Interrupt request, IRQ)	**Номер строки таблицы оборудования (физическое устройство, UCB)**

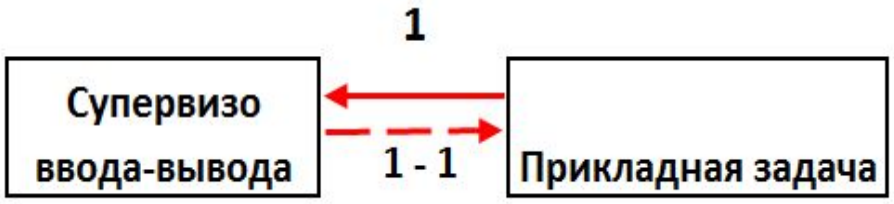
Взаимосвязь системных таблиц ввода-вывода



Процесс управления вводом/выводом

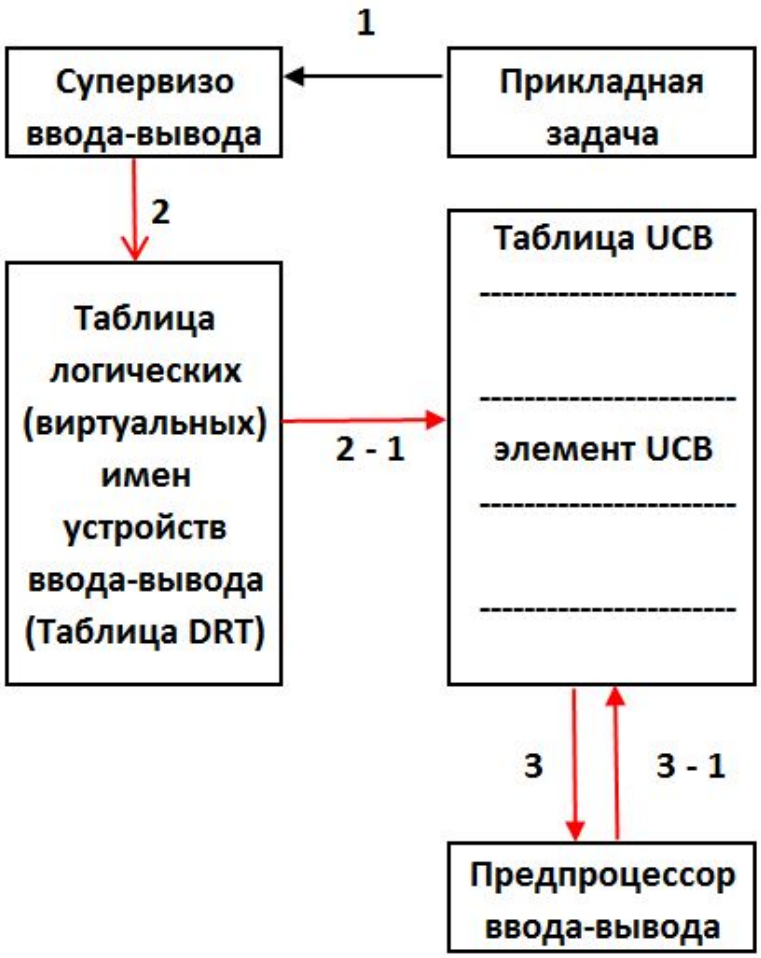


Процесс управления вводом/выводом



Программа, для дальнейшего выполнения которой требуется выполнить операцию ввода/вывода, направляет соответствующий запрос супервизору задач (шаг 1).
Этот запрос оформляется в виде системного вызова конкретной функции API. Вызов сопровождается некоторыми параметрами, уточняющими требуемую операцию.
Супервизор ввода/ вывода проверяет системный вызов на соответствие принятым в операционной системе спецификациям и в случае ошибки возвращает задаче соответствующее сообщение (шаг 1-1).

Процесс управления вводом/выводом

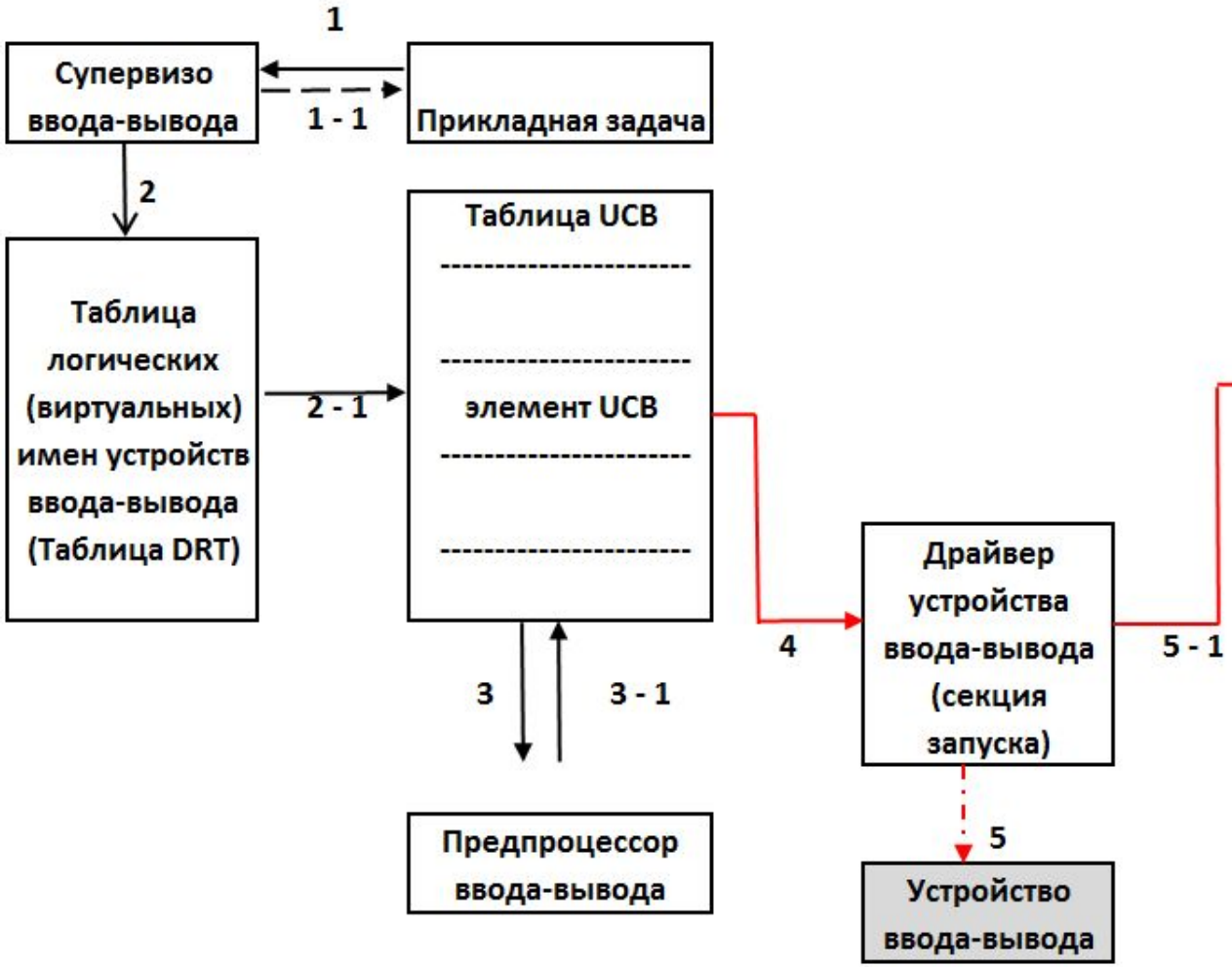


Если же запрос корректен, то супервизор ввода/вывода по логическому (виртуальному) имени с помощью таблицы DRT (шаг 2) находит соответствующий элемент UCB в таблице оборудования (шаг 2-1).

Если устройство уже занято, то номер дескриптора задачи, запрос которой обрабатывается супервизором ввода/вывода, помещается в дескриптор устройства в список задач, ожидающих это устройство.

Если же устройство свободно, то супервизор ввода/вывода определяет из UCB тип устройства и при необходимости запускает препроцессор (шаг 3), позволяющий получить последовательность управляющих кодов и данных, которую сможет правильно понять и отработать устройство (шаг 3-1).

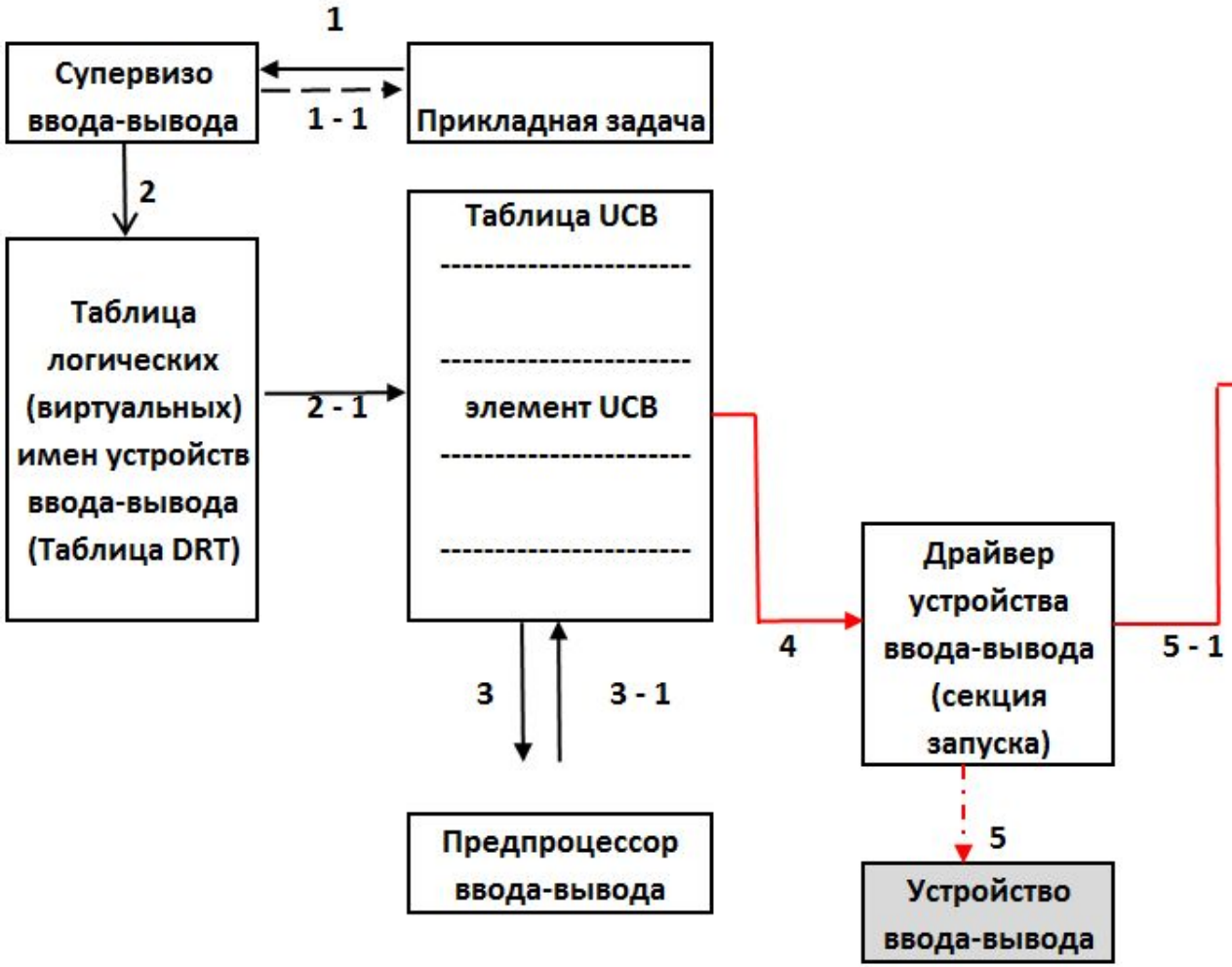
Процесс управления вводом/выводом



Когда «программа» управления операцией ввода/вывода, созданная предпроцессором ввода/вывода, будет готова, супервизор ввода/вывода передает управление соответствующему драйверу на секцию запуска (шаг 4).

Драйвер инициализирует операцию управления (шаг 5), обнуляет счетчик тайм-аута и передает управление диспетчеру задач (шаг 5-1) с тем, чтобы он поставил на процессор готовую к исполнению задачу.

Процесс управления вводом/выводом



Когда «программа» управления операцией ввода/вывода, созданная предпроцессором ввода/вывода, будет готова, супервизор ввода/вывода передает управление соответствующему драйверу на секцию запуска (шаг 4).

Драйвер инициализирует операцию управления (шаг 5), обнуляет счетчик тайм-аута и передает управление диспетчеру задач (шаг 5-1) с тем, чтобы он поставил на процессор готовую к исполнению задачу.

Процесс управления вводом/выводом

Система работает своим чередом, и когда устройство ввода/вывода отработает посланную ему команду, оно выставляет сигнал запроса на прерывание (шаг 6), по которому через таблицу прерываний управление передается на секцию продолжения (шаг 6-1).

Получив новую команду, внешнее устройство вновь начинает ее обрабатывать (шаг 7), а управление процессором снова передается диспетчеру задач (шаг 7-1), и процессор продолжает выполнять полезную работу. Таким образом, получается параллельная обработка задач, на фоне которой процессор осуществляет управление операциями ввода/вывода.