Концепции процесса

Определение процесса

Процесс

- Впервые термин процесс для описания программы в ходе ее выполнения был использован в системе MULTICS (вторая половина 1960-х годов)
- Иногда заменяется термином «задача» (task) и наоборот

Процесс

Опр. Процесс (process) – логический объект, описывающий программу в стадии ее выполнения.

Адресное пространство процесса

- Область команд
- Область данных
- Область стека

Область команд

Опр. Область команд (text region) – часть адресного пространства процесса, содержащая инструкции, предназначенные для выполнения процессором.

Область данных

Опр. Область данных (data region) – часть адресного пространства процесса, содержащая данные (а не инструкции). Эта область доступна для изменений.

Область стека

Опр. Область стека (stack region) – область адресного пространства процесса, содержащая инструкции и данные для открытых процедур. Размер стека увеличивается в случае вызова вложенных процедур и уменьшается по их завершении.

Вопрос для самопроверки

- Правда ли, что термины «процесс» и «программа» являются синонимами? (Да/Нет)

Вопрос для самопроверки

- Правда ли, что термины «процесс» и «программа» являются синонимами? (Да/Нет)
- Нет. Процессом называется программа в стадии выполнения, тогда как сама программа представляет собой неодушевленный логический объект.

Вопрос для самопроверки

- Правда ли, что адресное пространство процесса поделено на непересекающиеся области? (Да/Нет)

Вопрос для самопроверки

- Правда ли, что адресное пространство процесса поделено на непересекающиеся области? (Да/Нет)
- Да. Каждая область адресного пространства предназначена для хранения данных, схожих по способу доступа (доступ запрещен, открыт, по принципу стека). Память поделена на области для того, чтобы система могла обеспечить соблюдение этих правил.

Концепции процесса

Состояния процесса

Основные состояния процесса

- Состояние выполнения
- Состояние блокировки
- Состояние готовности

Состояние выполнения

Опр. Состояние выполнения (running state) – состояние процесса, если ему в данный момент выделен процессор.

Состояние блокировки

Опр. Состояние блокировки (blocked state) – состояние процесса, в котором он ожидает наступления определенного события, например, завершения операции ввода/вывода. Процесс в этом состоянии не может использовать процессор, даже если он свободен.

Состояние готовности

Опр. Состояние готовности (ready state) – состояние процесса, в котором он сразу мог бы использовать процессор, предоставленный в его распоряжение.

Список готовых к выполнению процессов

Опр. Список готовых к выполнению процессов (ready list) – структура данных ядра, в которой в упорядоченном виде хранятся указатели на все готовые к выполнению процессы. Сортировка списка готовых к выполнению процессов обычно осуществляется в соответствии с их приоритетом.

Список заблокированных процессов

Опр. Список заблокированных процессов (blocked list) – структура данных ядра, содержащая указатели на все заблокированные процессы. Данный список не упорядочен по приоритету.

Вопрос для самопроверки

- Правда ли, что в каждый конкретный момент времени в системе может выполняться только один процесс? (Да/нет)

Вопрос для самопроверки

- Правда ли, что в каждый конкретный момент времени в системе может выполняться только один процесс? (Да/нет)
- Нет. В многопроцессорной системе в каждый конкретный момент времени может выполняться столько процессов, сколько процессоров установлено в системе.

Вопрос для самопроверки

- Верно ли, что список заблокированных процессов хранится в упорядоченном виде? (Да/Нет)

Вопрос для самопроверки

- Верно ли, что список заблокированных процессов хранится в упорядоченном виде? (Да/Нет)
- Нет. Список заблокированных процессов хранится в неупорядоченном виде, так как никакого приоритетного порядка разблокировки не предусматривается – она осуществляется в том порядке, в котором происходят ожидаемые процессами события.

Концепции процесса

Переходы процесса из состояния в состояние

Диспетчер

Опр. Диспетчер (dispatcher) – компонент операционной системы, отбирающий для запуска на процессоре первый процесс из списка готовых к выполнению процессов. Является частью планировщика процессов.

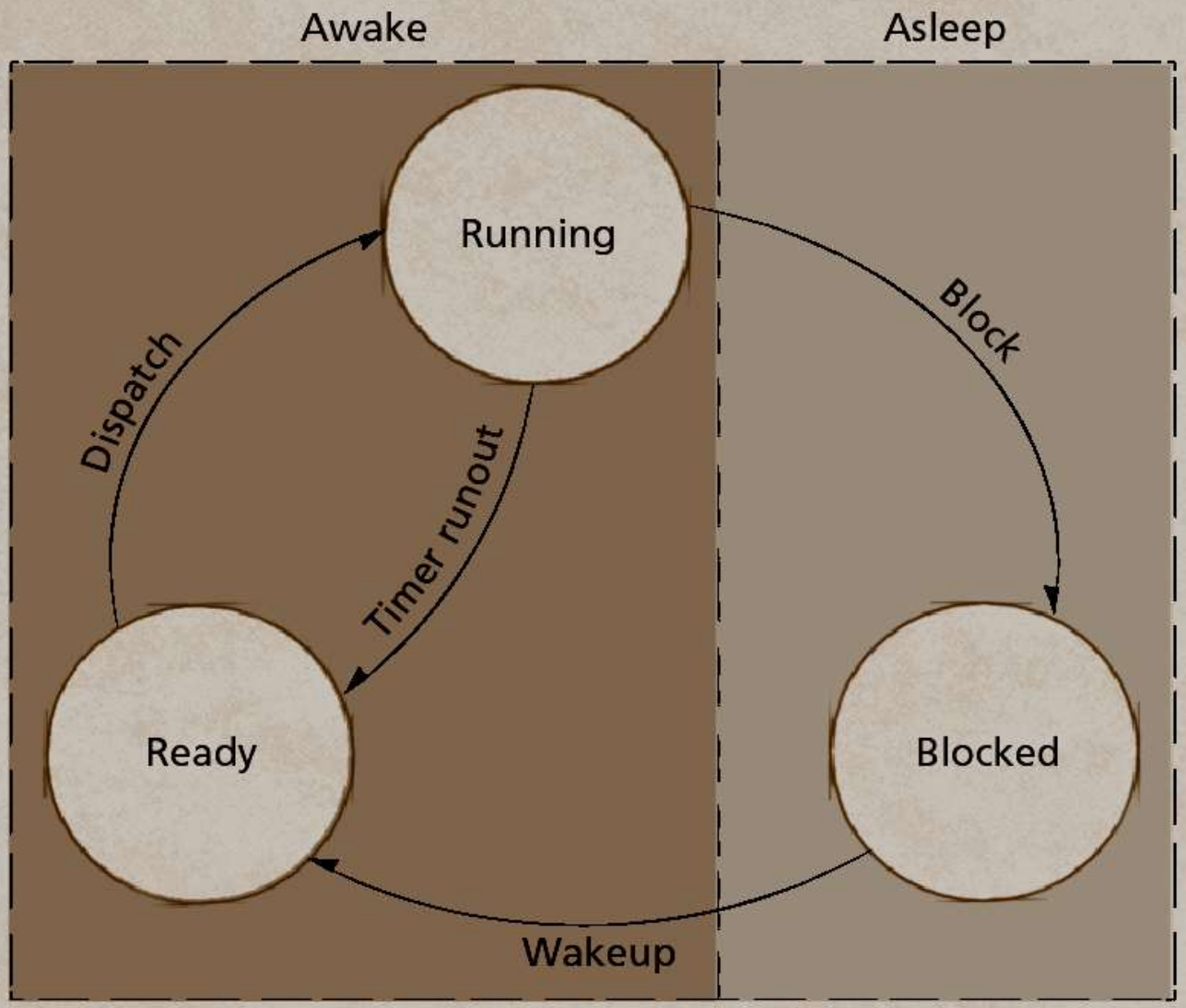
Квант

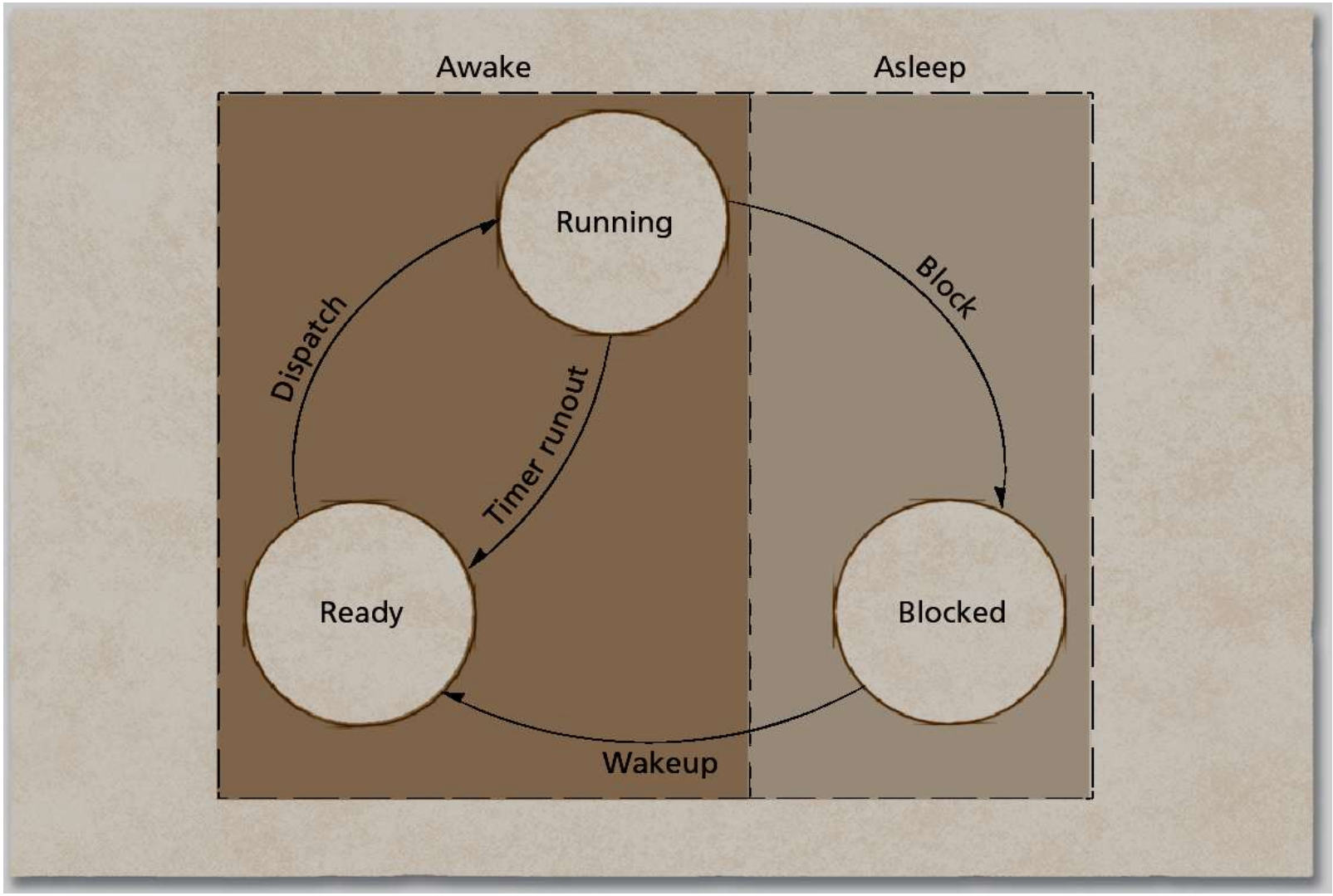
Опр. Квант (quantum) – промежуток времени, в течение которого процессор остается выделенным одному процессу. Использование квантов позволяет предотвратить монопольный захват процессора одним процессом.

Таймер прерываний

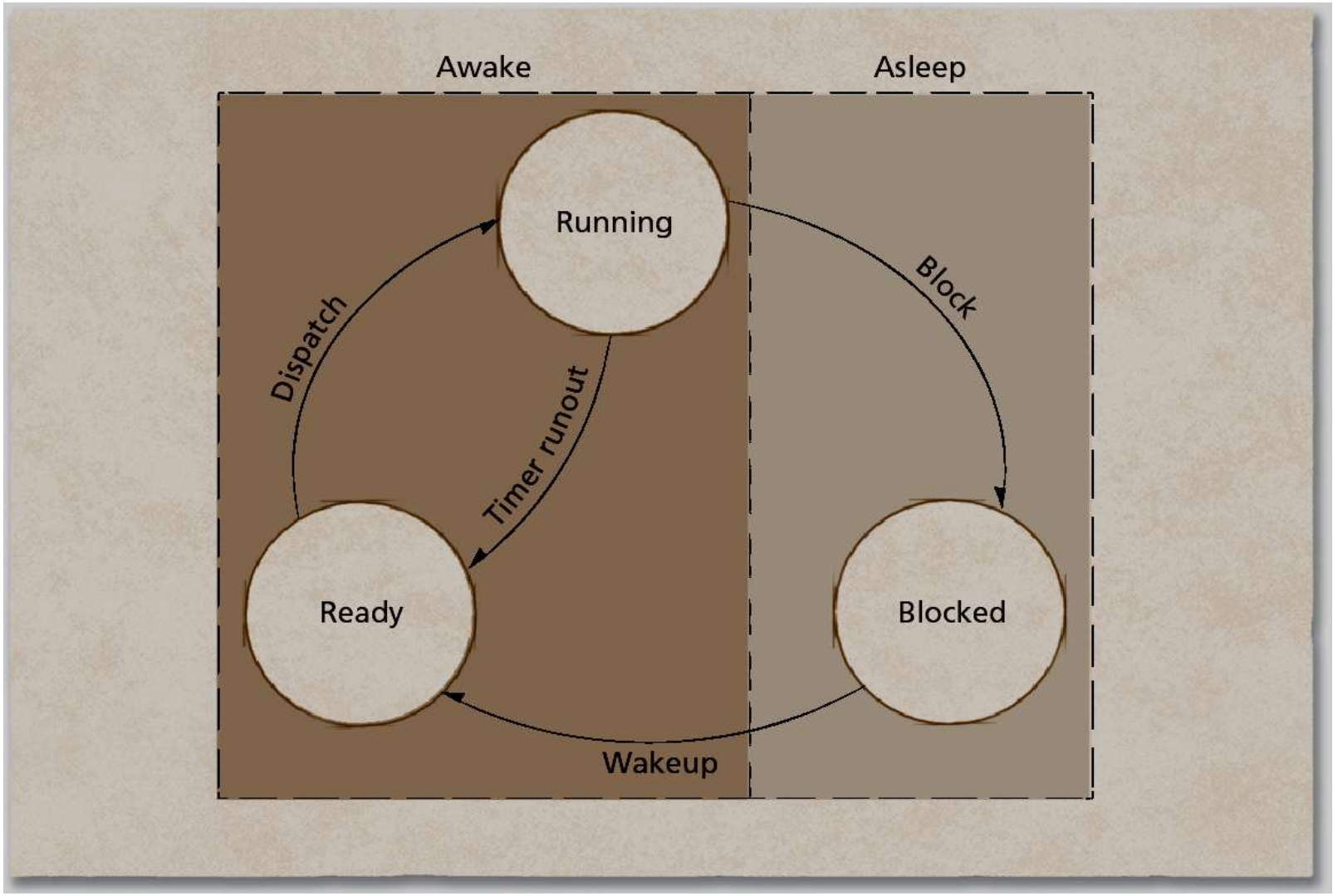
Опр. Таймер прерываний (interrupting clock) – аппаратно реализованный таймер, вырабатывающий сигналы прерывания через определенные промежутки времени (называемые квантами) для того чтобы не допустить монопольного захвата процессора одним процессом.

Диаграмма переходов процесса из состояния в состояние

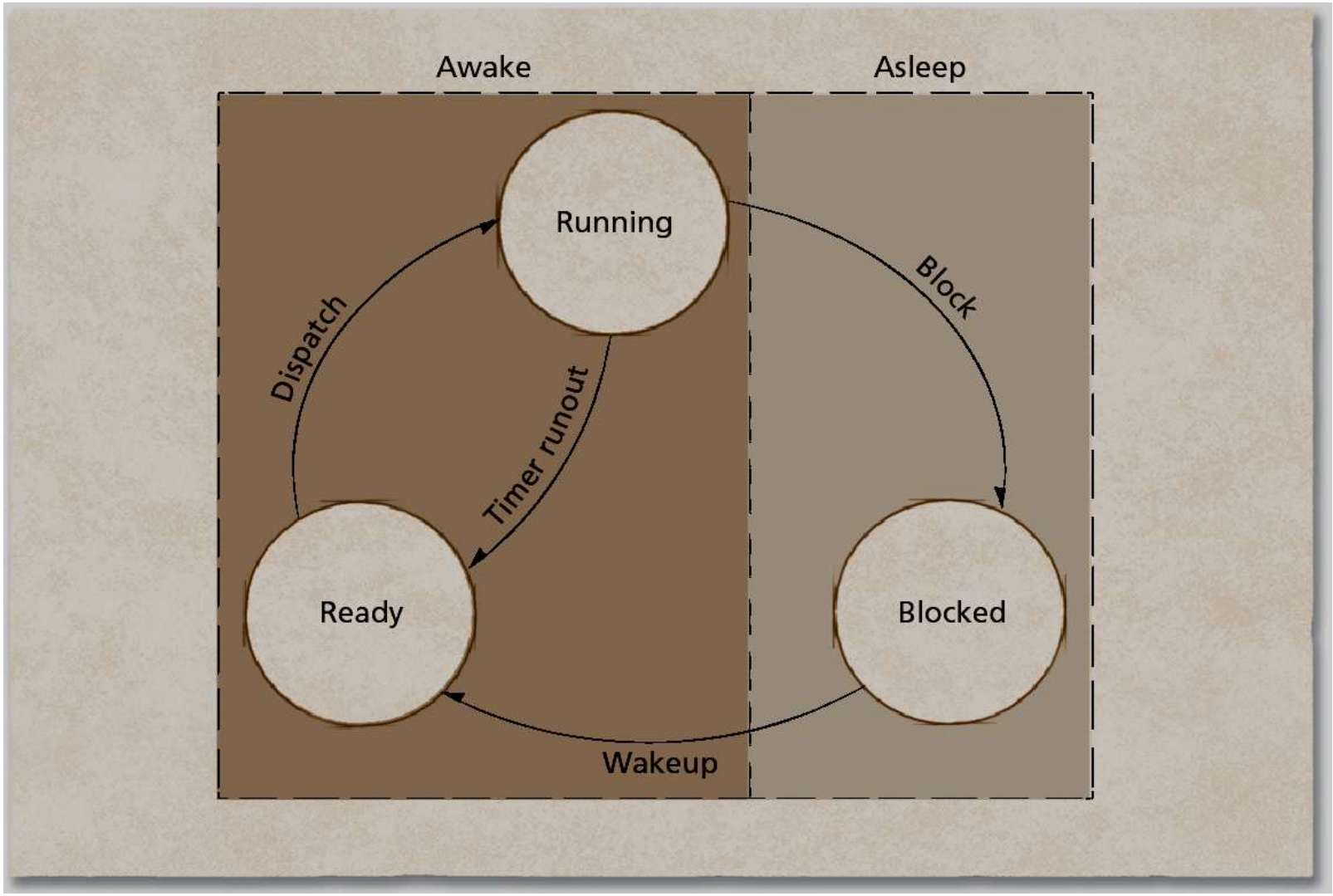




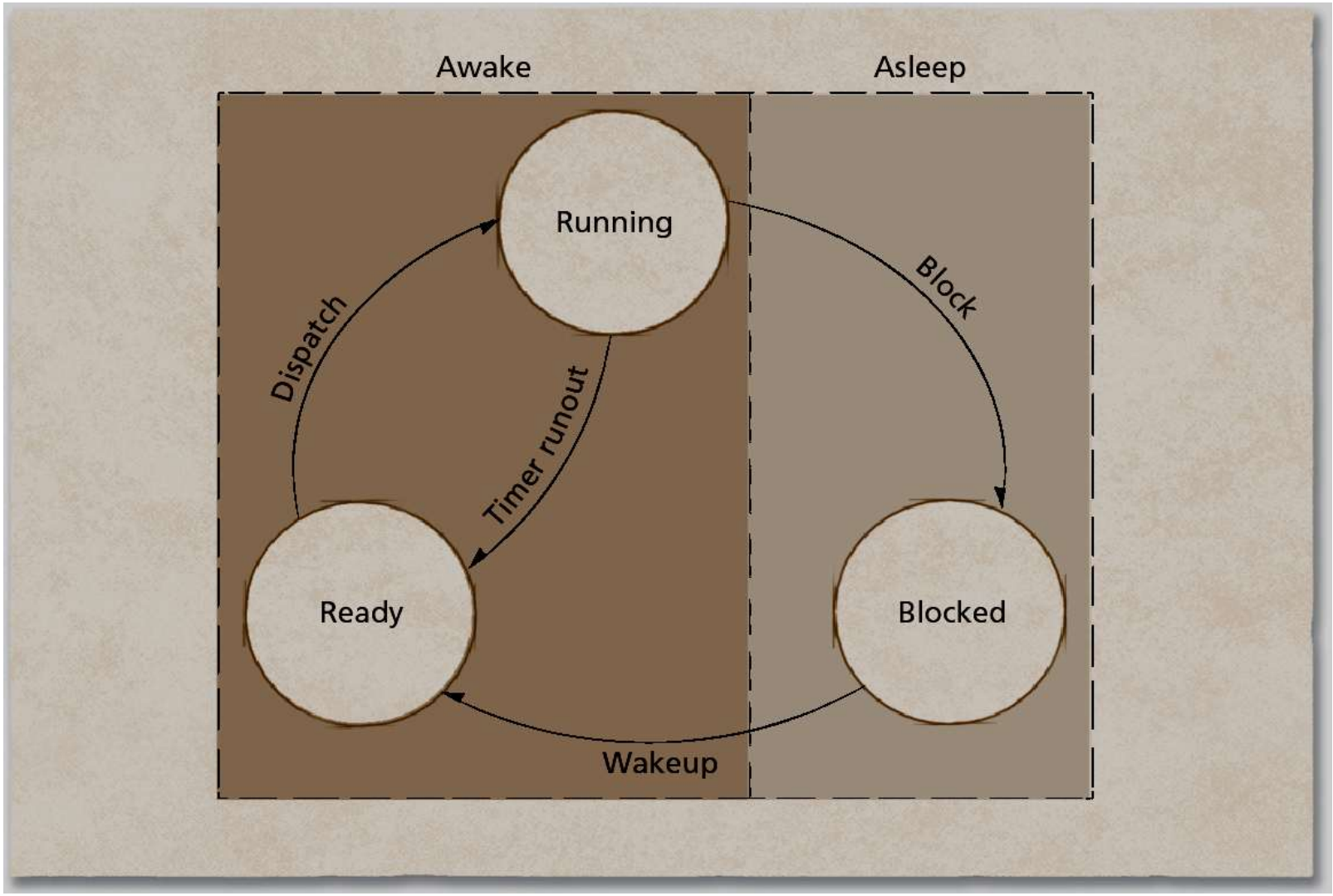
Awake – бодрствование; бодрствующие процессы постоянно соревнуются за процессорное время.



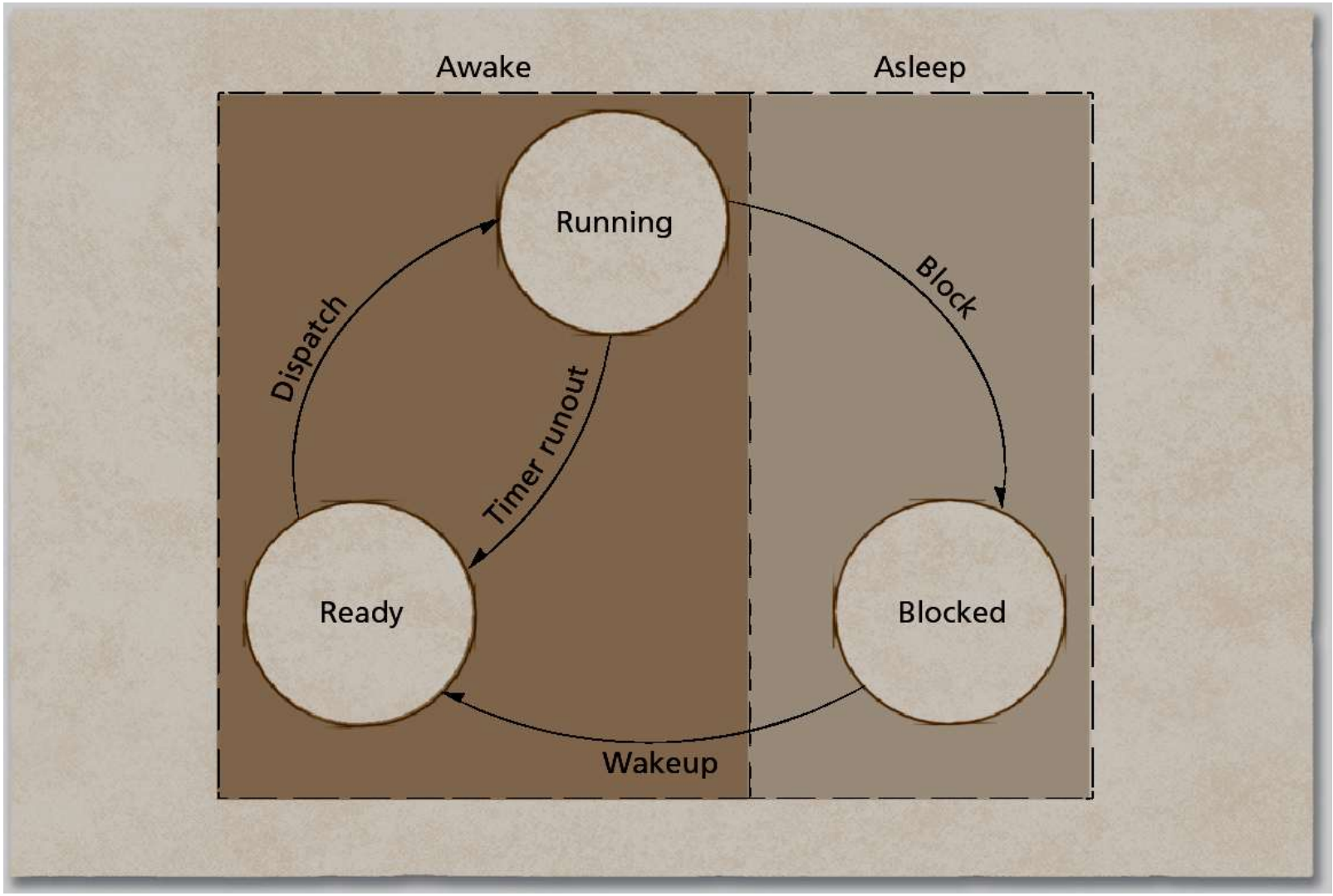
Asleep – спячка; спящий процесс не сможет использовать процессор, даже если тот снова станет доступен.



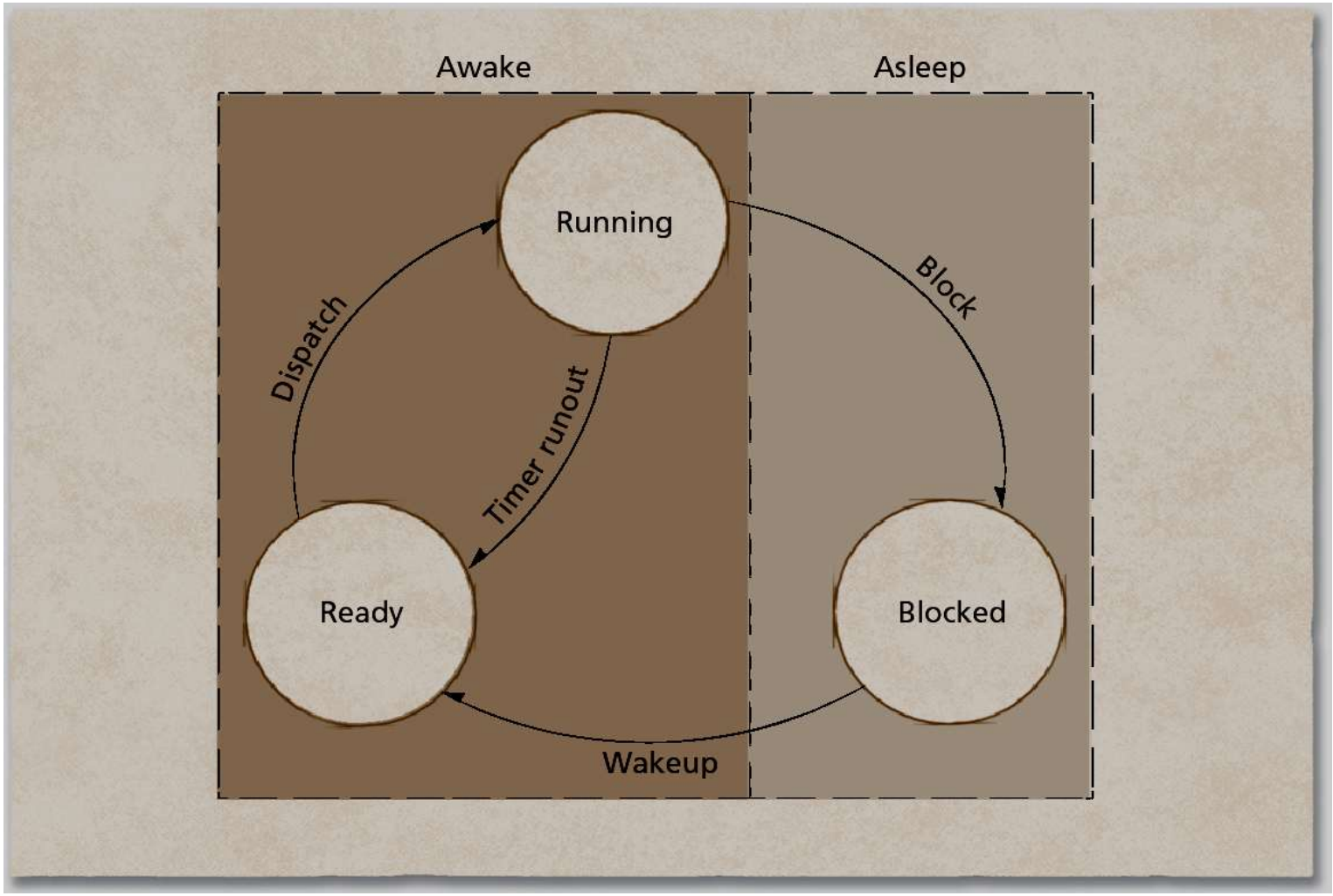
Dispatch – запуск; диспетчер планировщика процессов запускает на процессоре первый процесс из списка готовых к выполнению процессов.



Timer run out – истечение кванта; по истечении кванта планировщик процессов переводит процесс в состояние готовности из состояния выполнения.



Block – блокирование; процесс блокирует сам себя – если до истечения выделенного ему кванта времени он начнет операцию ввода/вывода, то добровольно освободит процессор.



Wakeup – пробуждение; по завершении операции ввода/вывода, ожидаемом процессом, планировщик процессов перемещает указатель на данный процесс из списка заблокированных в список готовых к выполнению процессов.

Вопрос для самопроверки

- Верно ли, что все переходы между состояниями процесса осуществляет планировщик процессов? (Да/Нет)

Вопрос для самопроверки

- Верно ли, что все переходы между состояниями процесса осуществляет планировщик процессов? (Да/Нет)
- Нет. Планировщик процессов осуществляет запуск, пробуждение и перевод процесса из состояния выполнения в состояние готовности по истечении кванта времени. Блокирование осуществляется самим процессом.

Вопрос для самопроверки

- Возможен ли переход процесса, из состояния готовности в состояние блокировки? (Да/Нет)

Вопрос для самопроверки

- Возможен ли переход процесса, из состояния готовности в состояние блокировки? (Да/Нет)
- Нет. Процесс не может заблокировать себя из состояния готовности, так как для того чтобы сгенерировать запрос ввода/вывода, он должен выполняться.

Вопрос для самопроверки

- Возможен ли переход процесса, из состояния блокировки в состояние выполнения? (Да/Нет)

Вопрос для самопроверки

- Возможен ли переход процесса, из состояния блокировки в состояние выполнения? (Да/Нет)
- Нет. По завершении операции ввода/вывода процесс не может быть сразу направлен на выполнение, так как в это время процессор может быть занят более приоритетным процессом.

Концепции процесса

Блоки управления процессами (дескрипторы процессов)

Таблица процессов

Опр. Таблица процессов (process table) – структура данных, в которой хранятся указатели на все процессы системы.

Идентификационный номер процесса

Опр. Идентификационный номер процесса (Process Identification Number, PID) – числовое значение, уникальным образом идентифицирующее процесс.

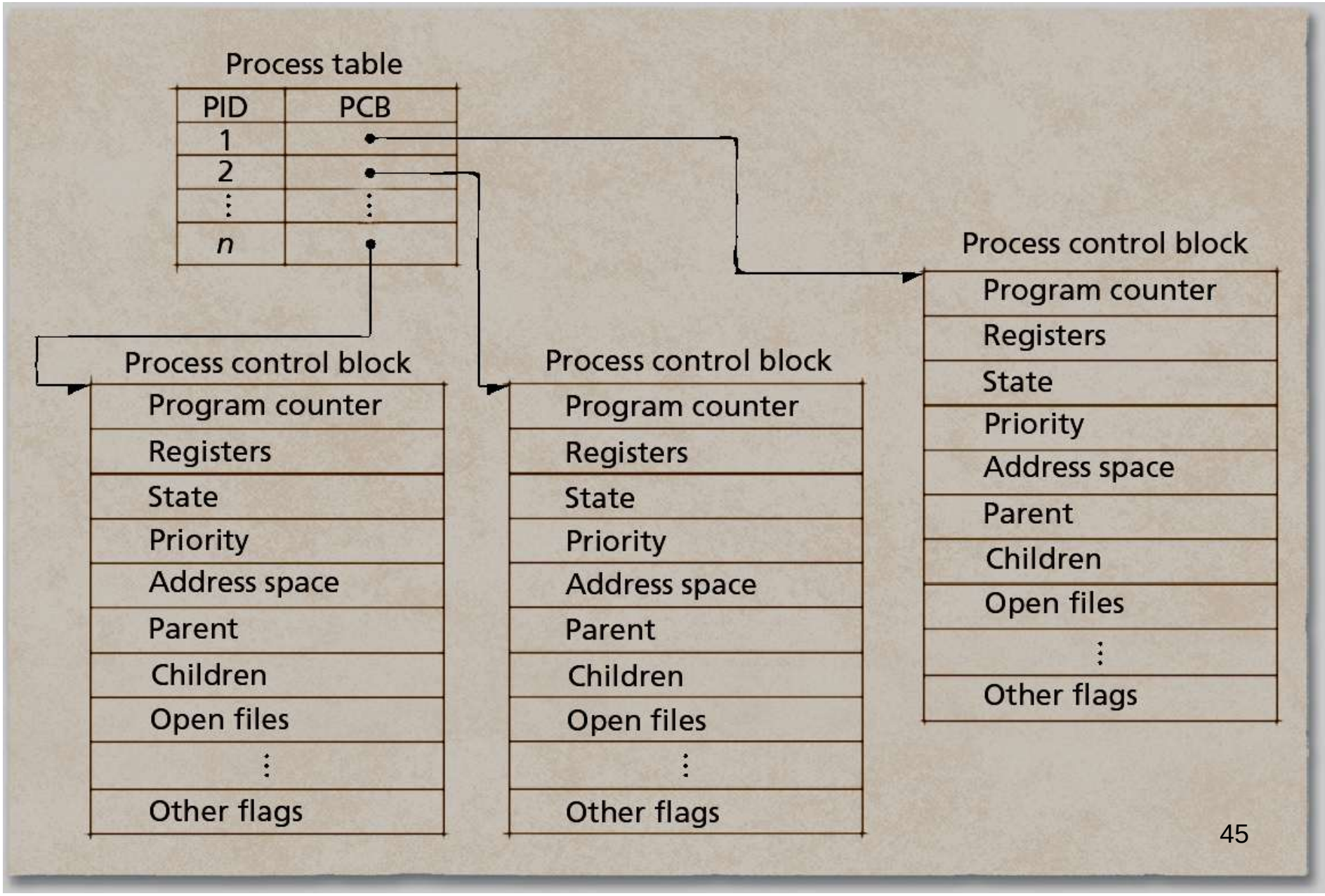
Блок управления процессом

Опр. Блок управления процессом (Process Control Block, PCB) – структура данных, содержащая информацию о процессе (состояние, адресное пространство и пр.). Другое название – дескриптор процесса.

Обычное содержимое PCB

- Программный счетчик
- Контекст выполнения
- Состояние процесса
- Приоритет процесса
- Указатели на адресное пространство
- Указатель на родительский процесс
- Указатели на дочерние процессы
- Указатели на открытые файлы
- …

Таблица процессов и блоки управления процессами



Программный счетчик

Опр. Программный счетчик (program counter) – указатель на следующую инструкцию процесса, которая должна быть выполнена процессором. После выполнения этой инструкции, значение программного счетчика корректируется таким образом, чтобы тот ссылался на следующую инструкцию процесса.

Контекст выполнения

Опр. Контекст выполнения (registers, execution context) – содержимое регистров общего назначения и некоторых управляющих регистров перед выходом процесса из состояния выполнения. Контекст выполнения зависит от архитектуры системы. Благодаря этим данным операционная система может восстанавливать значения регистров в случае возвращения процесса в состояние выполнения.

Состояние процесса

Опр. Состояние процесса (process state) – статус процесса (выполняется, готов, блокирован).

Приоритет процесса

Опр. Приоритет процесса (process priority) – значение, определяющее важность данного процесса по сравнению с другими процессами.

Адресное пространство

Опр. Адресное пространство (address space) – области памяти (команд, данных и стека), с которыми может работать процесс.

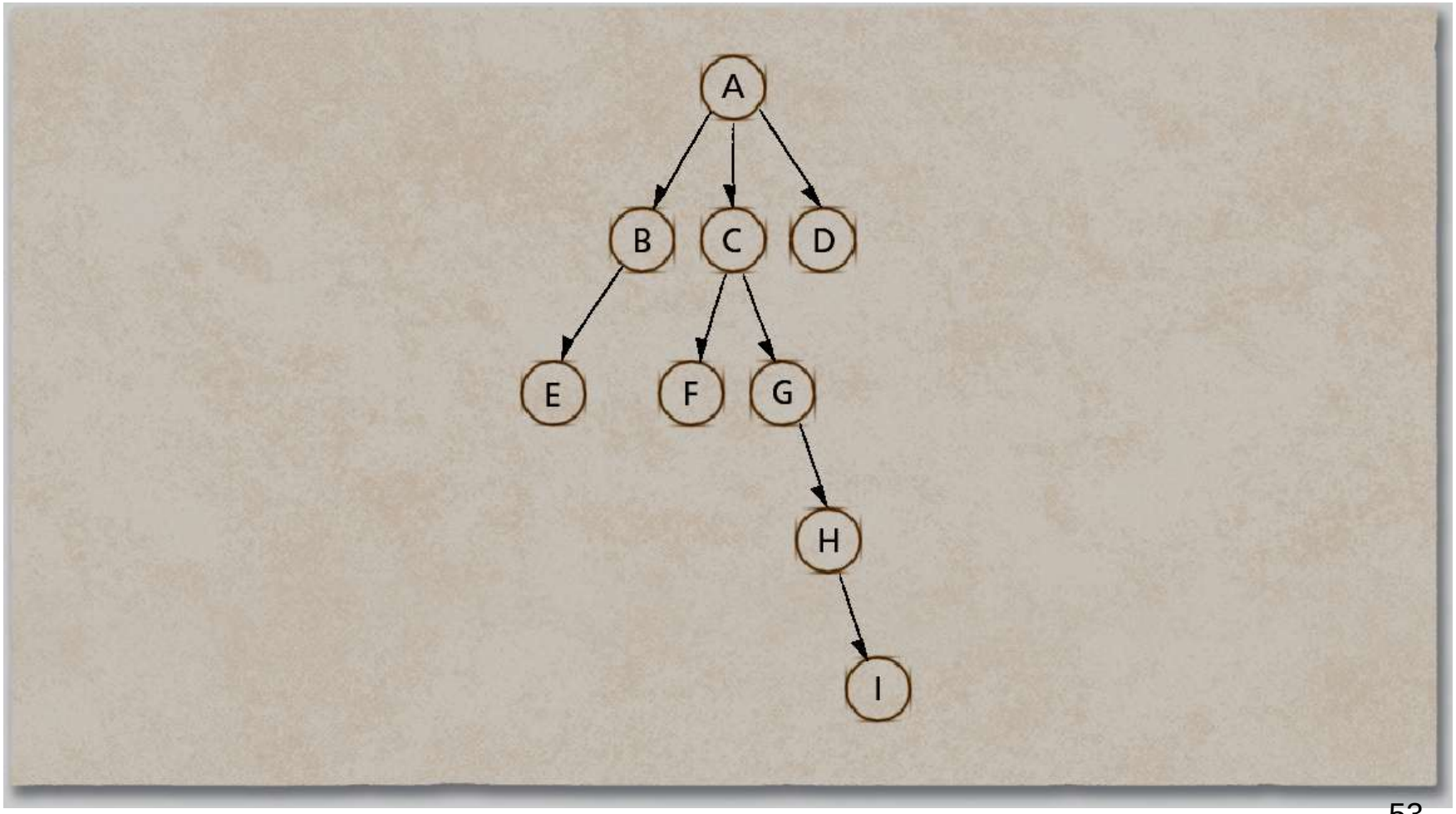
Родительский процесс

Опр. Родительский процесс (parent process) – процесс, породивший один или несколько дочерних процессов.

Дочерний процесс

Опр. Дочерний процесс (child process) – новый процесс, созданный родительским процессом. Дочерний процесс располагается на один уровень ниже родительского в иерархии создания процессов.

Иерархия создания процессов



Вопрос для самопроверки

- Правда ли, что таблица процессов нужна для их упорядочивания? (Да/Нет)

Вопрос для самопроверки

- Правда ли, что таблица процессов нужна для их упорядочивания? (Да/Нет)
- Нет. Таблица процессов позволяет находить блоки управления процессами.

Вопрос для самопроверки

- Правда ли, что структура блока управления процессом зависит от реализации операционной системы? (Да/Нет)

Вопрос для самопроверки

- Правда ли, что структура блока управления процессом зависит от реализации операционной системы? (Да/Нет)
- Да. Структура блока управления процессом определяется архитектурой вычислительной системы и существенно зависит от реализации операционной системы.

Вопрос для самопроверки

- Правда ли, что процесс может вообще не иметь родительского процесса? (Да/Нет)

Вопрос для самопроверки

- Правда ли, что процесс может вообще не иметь родительского процесса? (Да/Нет)
- Да. Первый создаваемый операционной системой процесс, например, init в Unix не имеет родительского процесса.

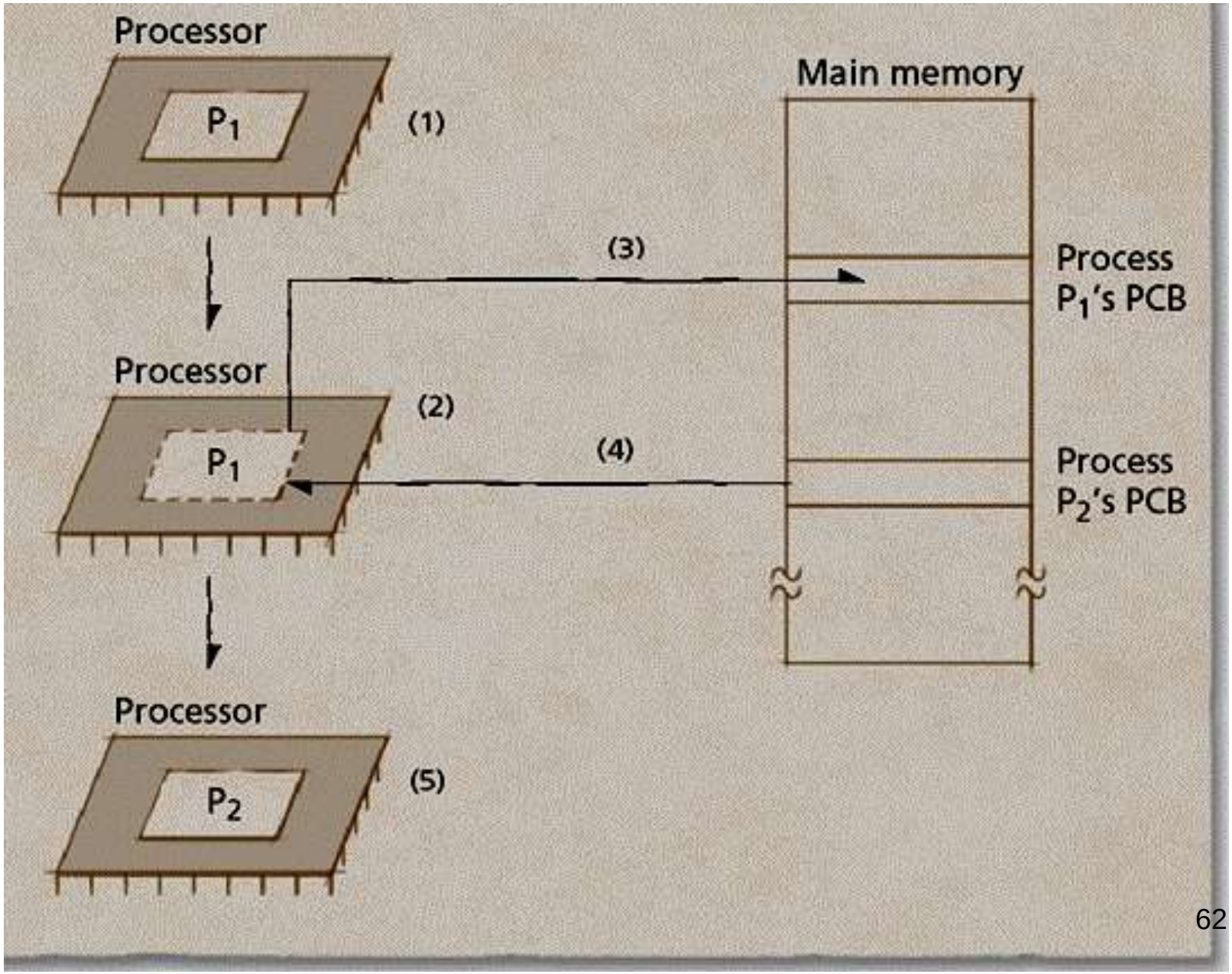
Концепции процесса

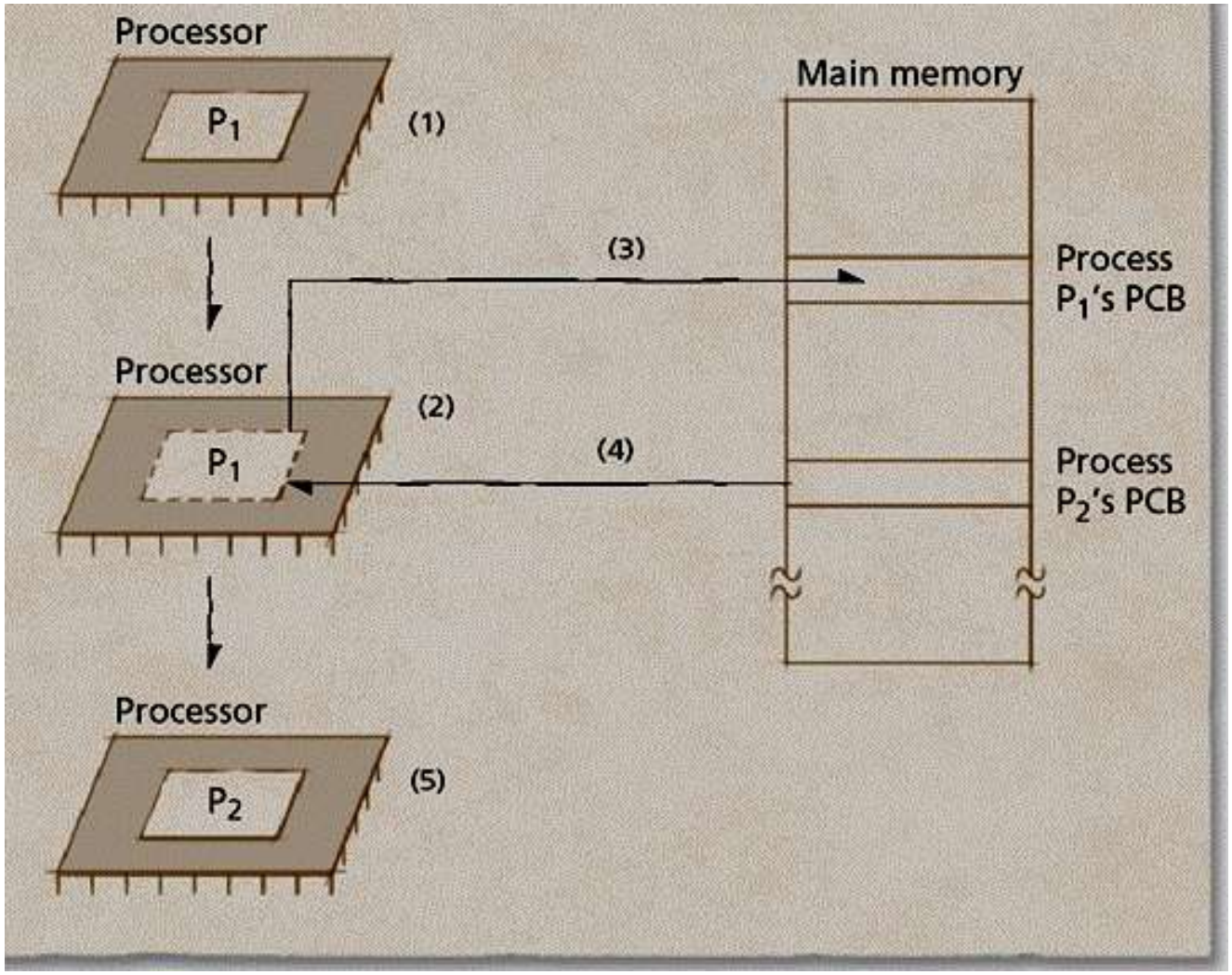
Переключение контекста

Переключение контекста

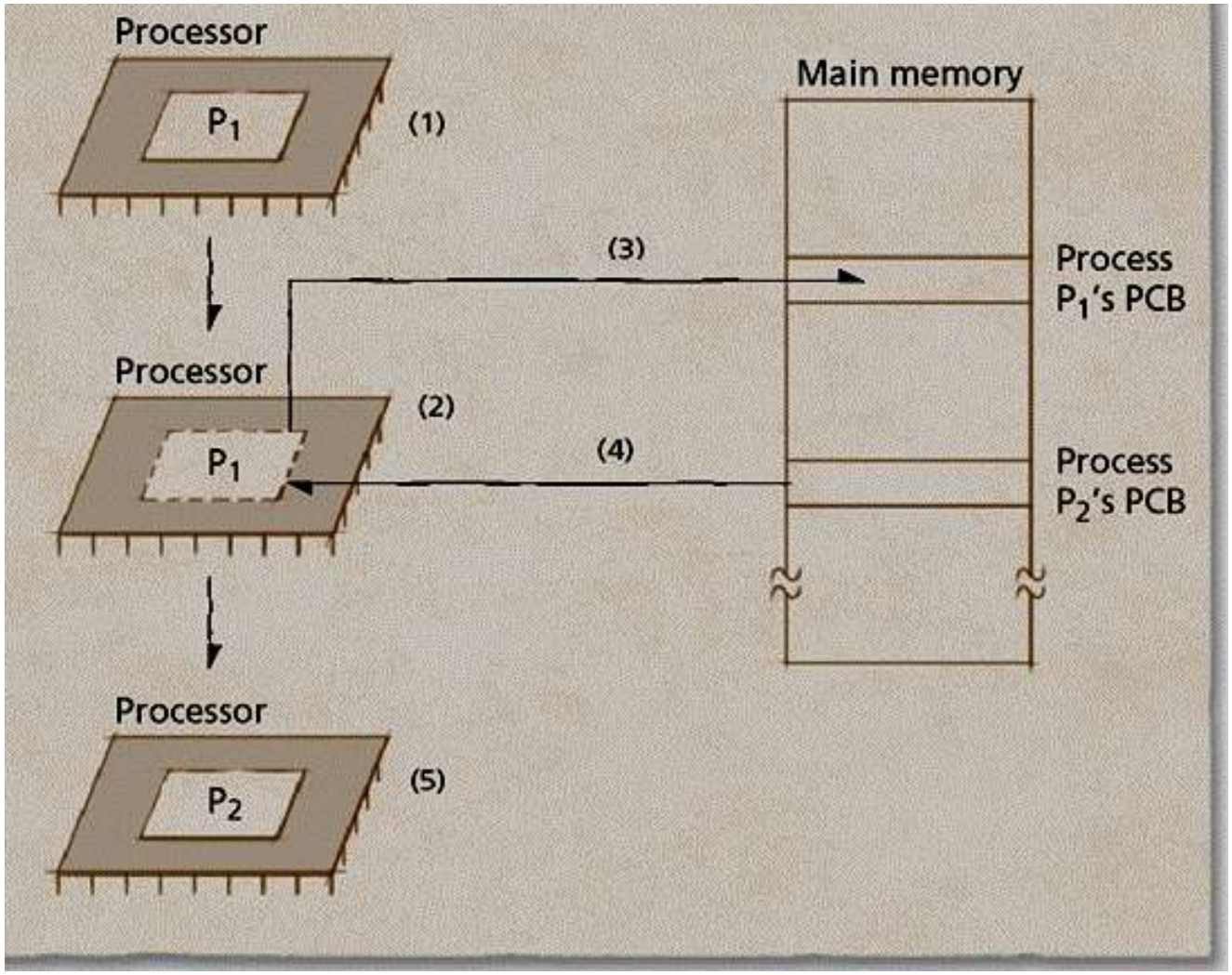
Опр. Переключение контекста (context switching) – выполняемая операционной системой операция отбора процессора у одного процесса и его выделения другому процессу. При этом ОС обязана сохранить состояние прерываемого процесса. Аналогично, система должна восстановить состояние процесса, выбираемого для запуска из списка готовых к выполнению.

Переключение контекста

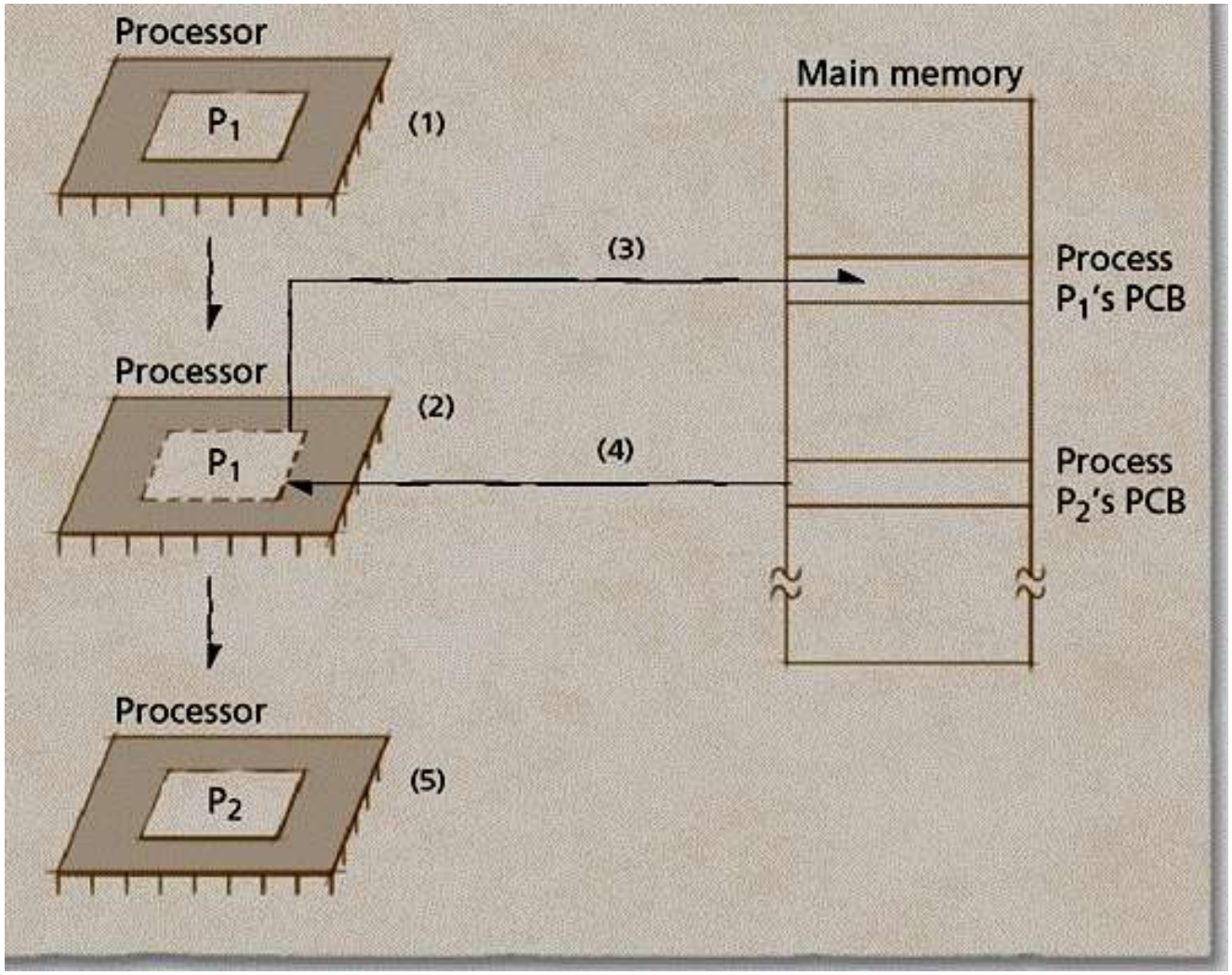




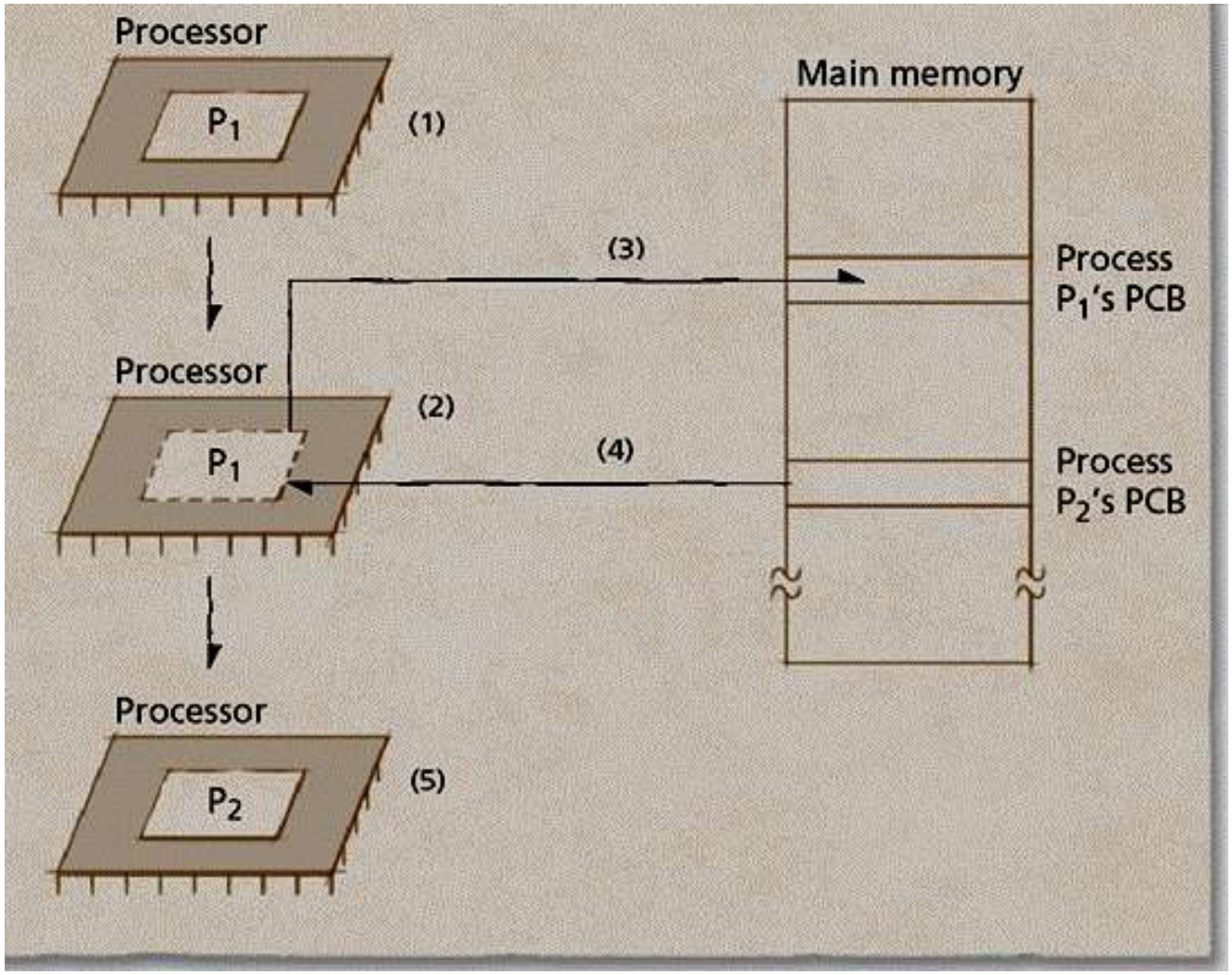
1. До начала переключения контекста процесс P_1 выполняется на процессоре



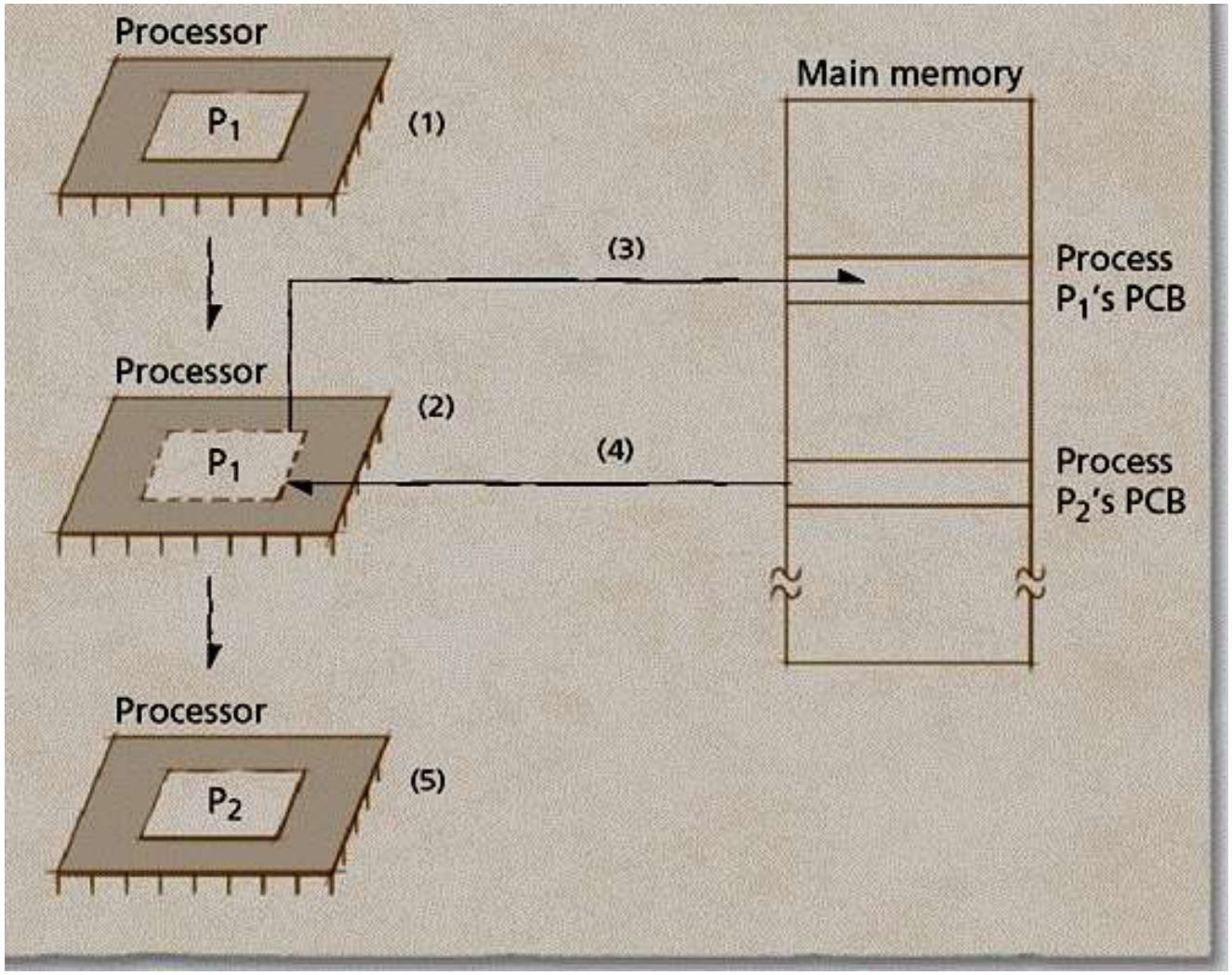
2. После поступления сигнала прерывания ядро принимает решение о выборе нового процесса и инициирует переключение контекста



3. Ядро сохраняет контекст выполнения процесса P_1 в его PCB в оперативной памяти



4. Ядро загружает контекст выполнения процесса P_2 из его PCB из оперативной памяти



5. После переключения контекста процесс P_2 выполняется на процессоре

Вопрос для самопроверки

- Верно ли, что во время переключения контекста процессор не может выполнять инструкции от имени процессов? (Да/Нет)

Вопрос для самопроверки

- Верно ли, что во время переключения контекста процессор не может выполнять инструкции от имени процессов? (Да/Нет)
- Да. Во время переключения контекста процессор не может выполнять «полезные вычисления». Поэтому операционная система должна минимизировать время, затрачиваемое на переключение контекста.

Вопрос для самопроверки

- Верно ли, что операционная система сохраняет контекст прерванного процесса в таблице процессов? (Да/Нет)

Вопрос для самопроверки

- Верно ли, что операционная система сохраняет контекст прерванного процесса в таблице процессов? (Да/Нет)
- Нет. Во время переключения контекста ОС сохраняет контекст прерванного процесса в его блоке управления процессом (PCB).

Концепции процесса

Прерывания

Прерывание

Опр. Прерывание (interrupt) – аппаратный сигнал, сообщающий о наступлении определенного события. Прерывания заставляют процессор выполнять последовательность программных инструкций, называемых обработчиком прерываний.

Обработчик прерываний

Опр. Обработчик прерываний (interrupt handler) – код ядра, выполняемый в ответ на прерывание.

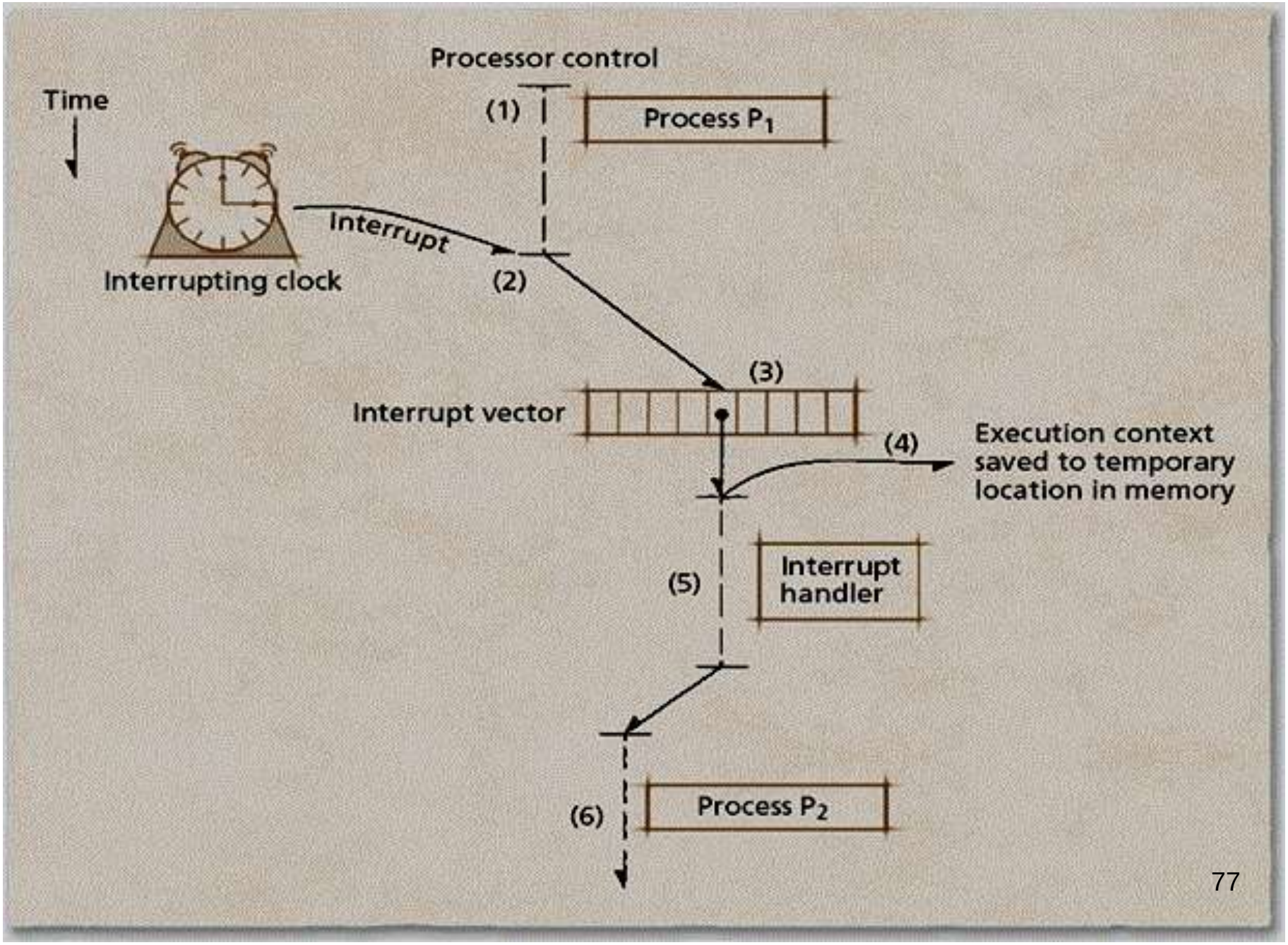
Вектор прерываний

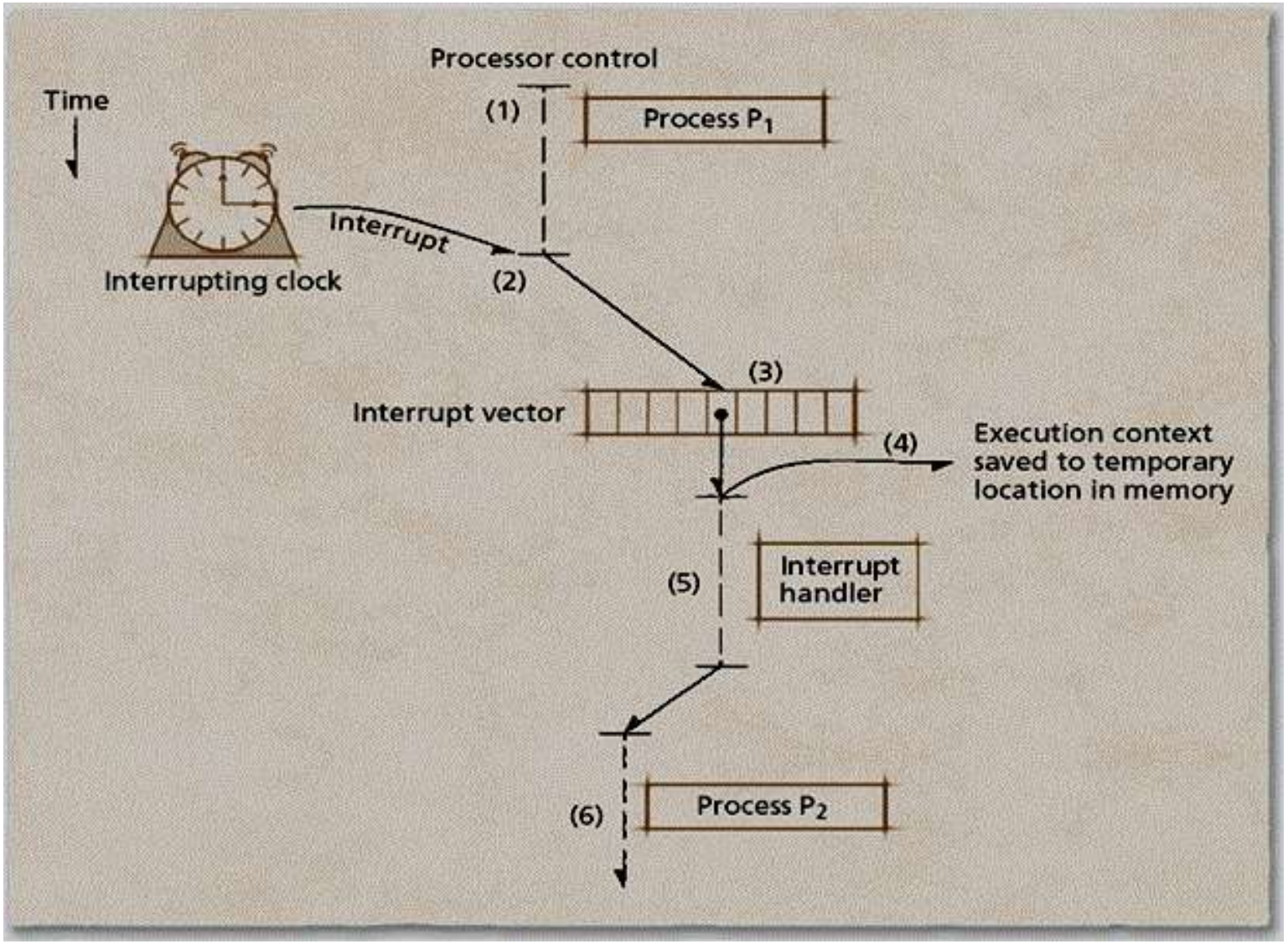
Опр. Вектор прерываний (interrupt vector) – защищенный массив в памяти, в котором хранятся указатели на местоположение обработчиков прерываний.

Контроллер прерываний

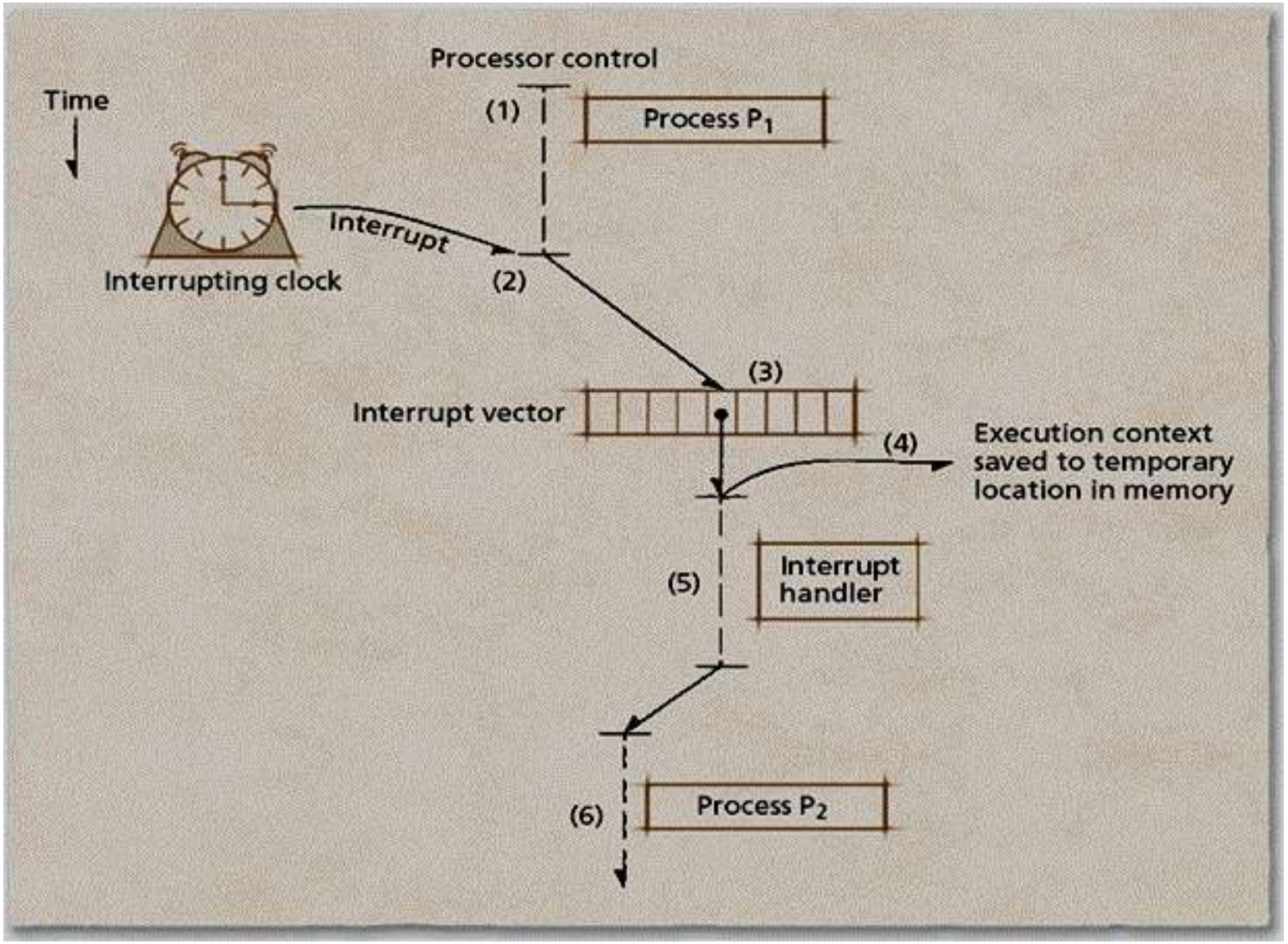
Опр. Контроллер прерываний (interrupt controller) – микросхема на материнской плате, либо аппаратный компонент процессора, сортирующий прерывания в соответствии с их приоритетом, обеспечивая первоочередную обработку высокоприоритетных прерываний.

Обработка прерывания таймера прерываний

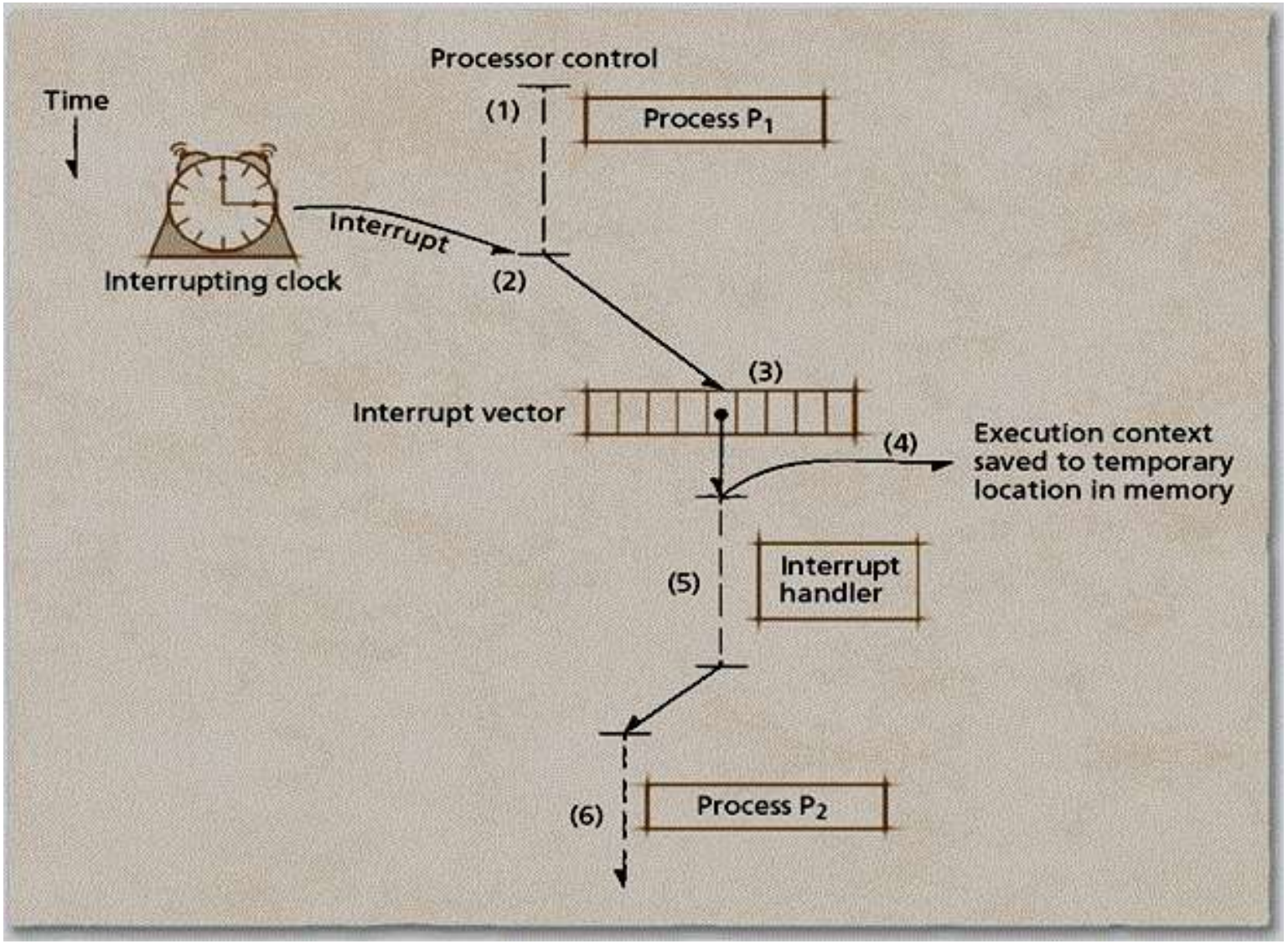




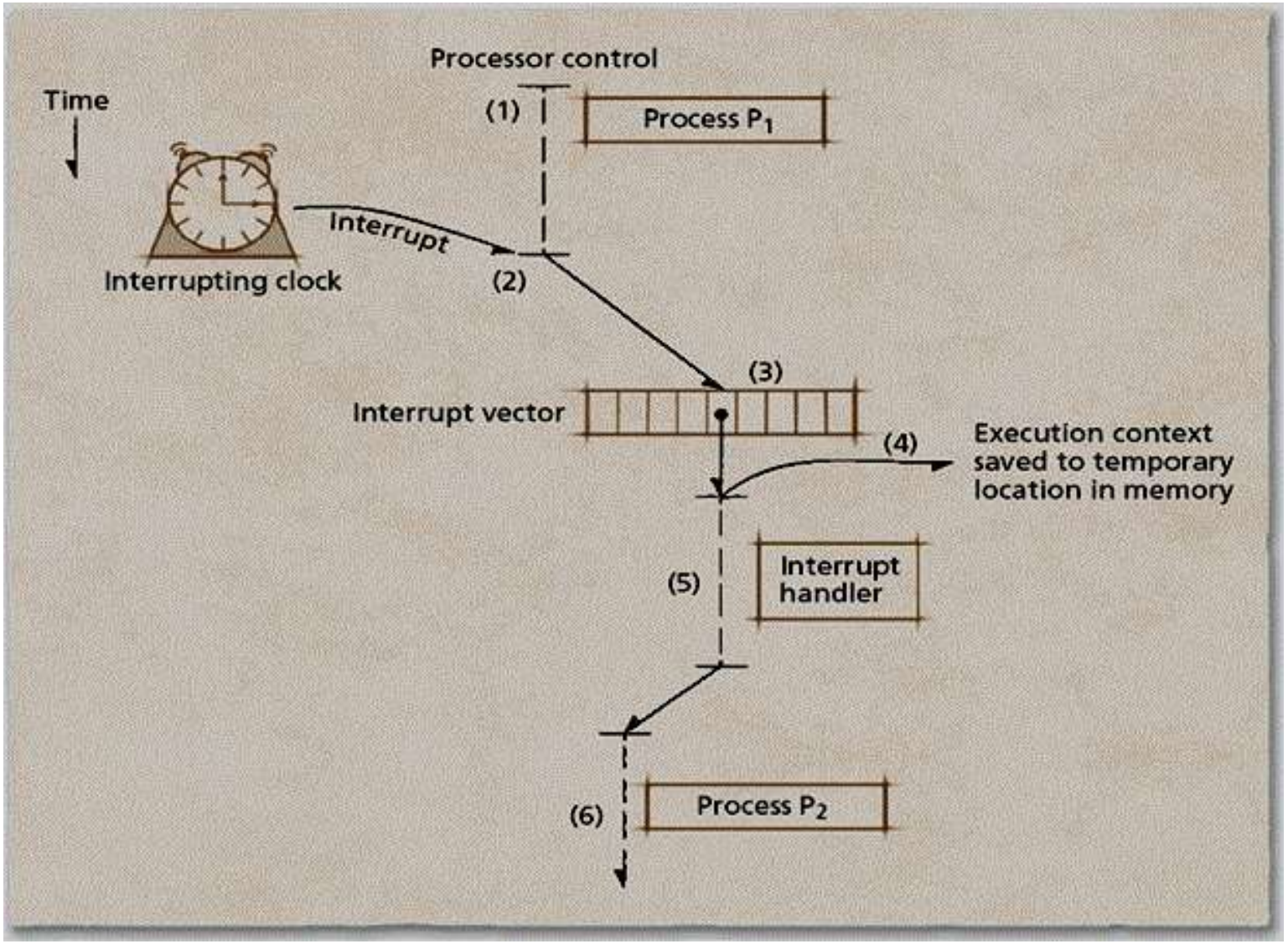
1. Процесс P_1 выполняется на процессоре до поступления сигнала прерывания



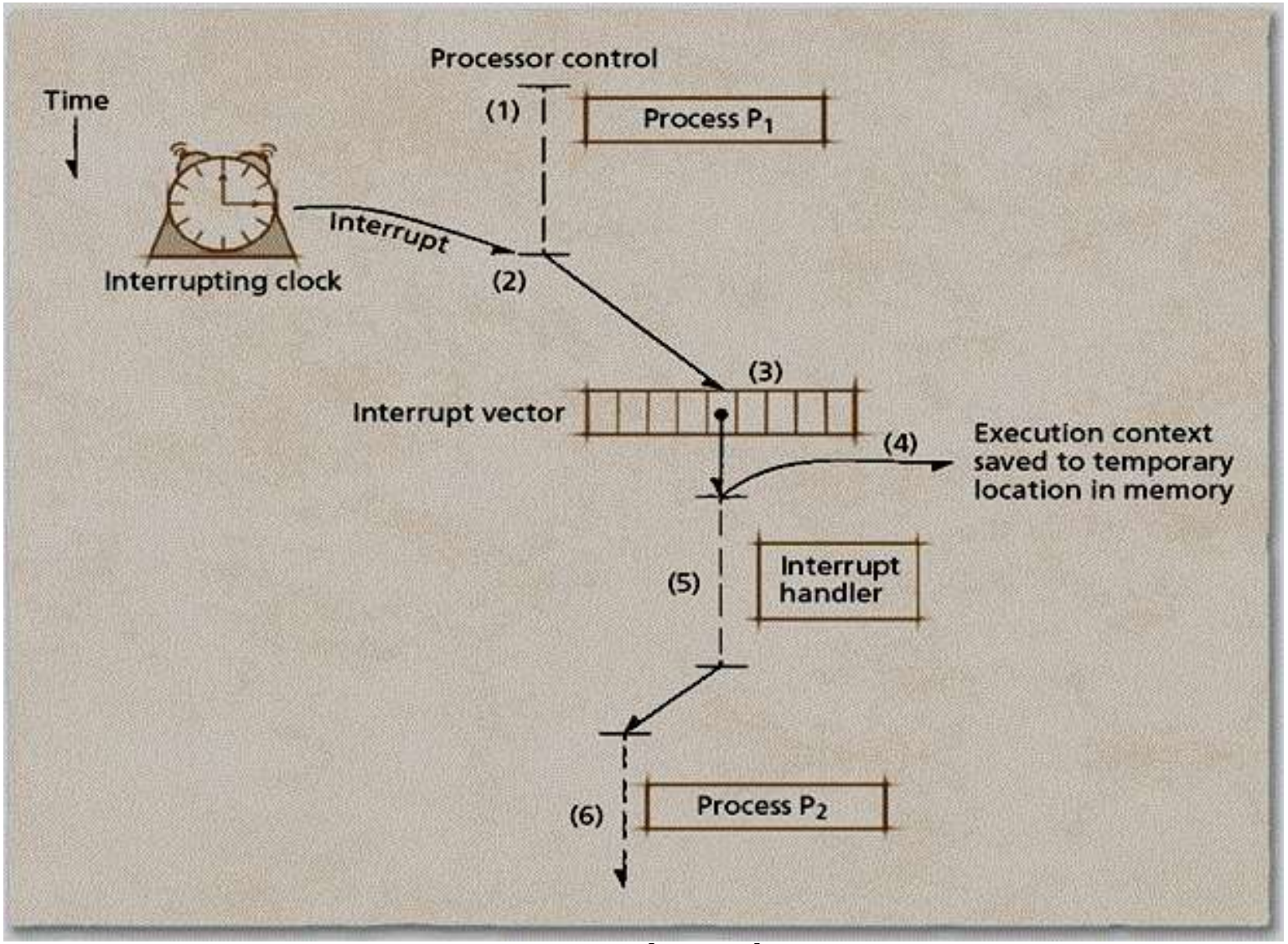
2. Таймер прерываний по истечении кванта времени генерирует прерывание, чтобы исключить захват процессора процессом P_1



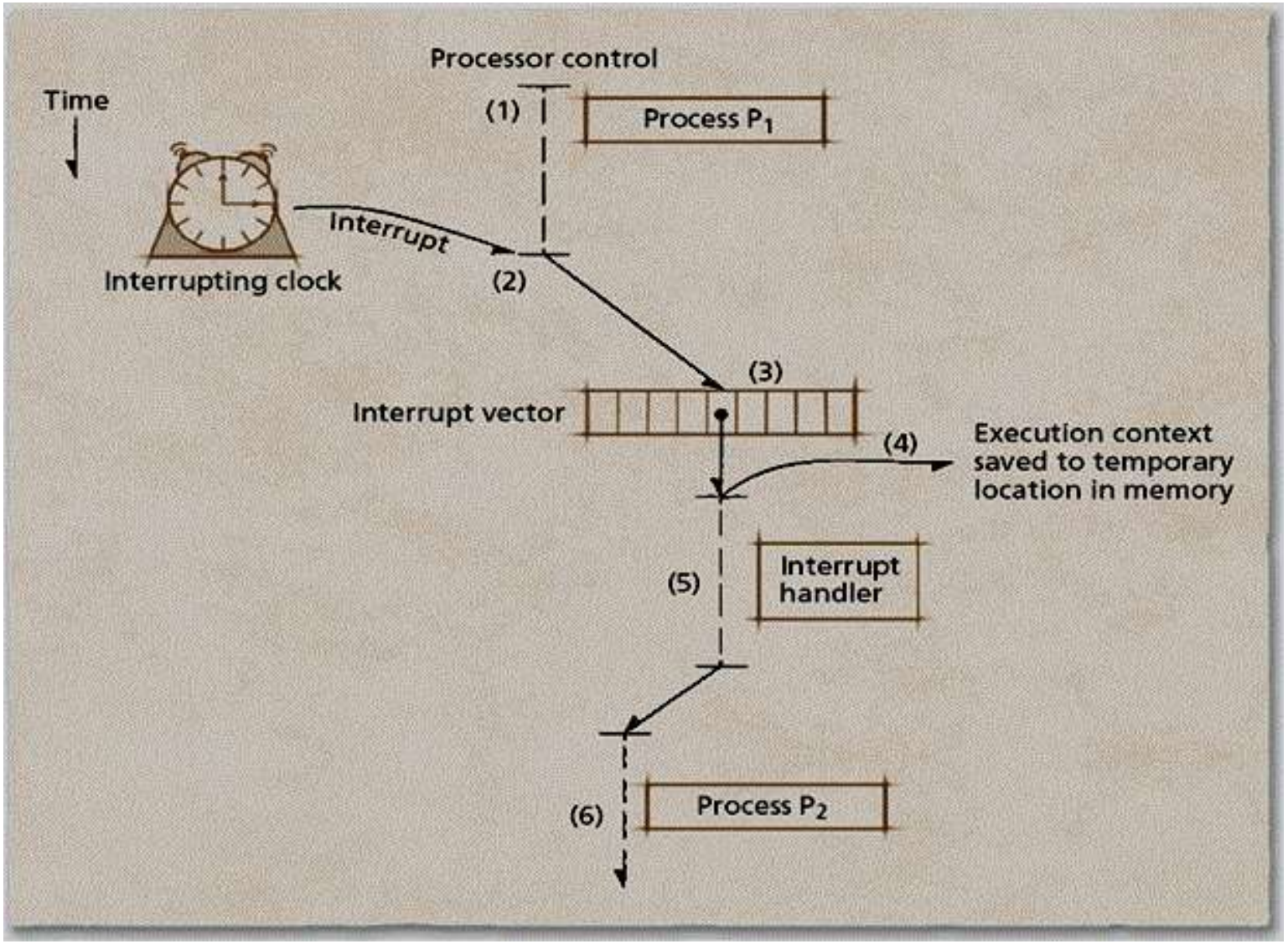
3. Процессор обращается к записи вектора прерываний, соответствующей прерываниям таймера прерываний



4. Процессор сохраняет контекст выполнения процесса P_1 в его PCB в оперативной памяти



5. Процессор выполняет обработчик прерываний, тот вызывает планировщик процессов, который загружает контекст выполнения процесса P_2 из его PCB из оперативной памяти



6. Процесс P_2 выполняется на процессоре после загрузки его контекста выполнения

Вопрос для самопроверки

- Верно ли, что при запуске обработчика прерываний контекст выполнения прерванного процесса сохраняется в памяти? (Да/Нет)

Вопрос для самопроверки

- Верно ли, что при запуске обработчика прерываний контекст выполнения прерванного процесса сохраняется в памяти? (Да/Нет)
- Да. Если не сохранить в памяти контекст выполнения процесса, обработчик прерываний может перезаписать значения регистров, используемых процессом.

Вопрос для самопроверки

- Может ли прерванный процесс продолжить свое выполнение сразу после обработки прерывания таймера прерываний? (Да/Нет)

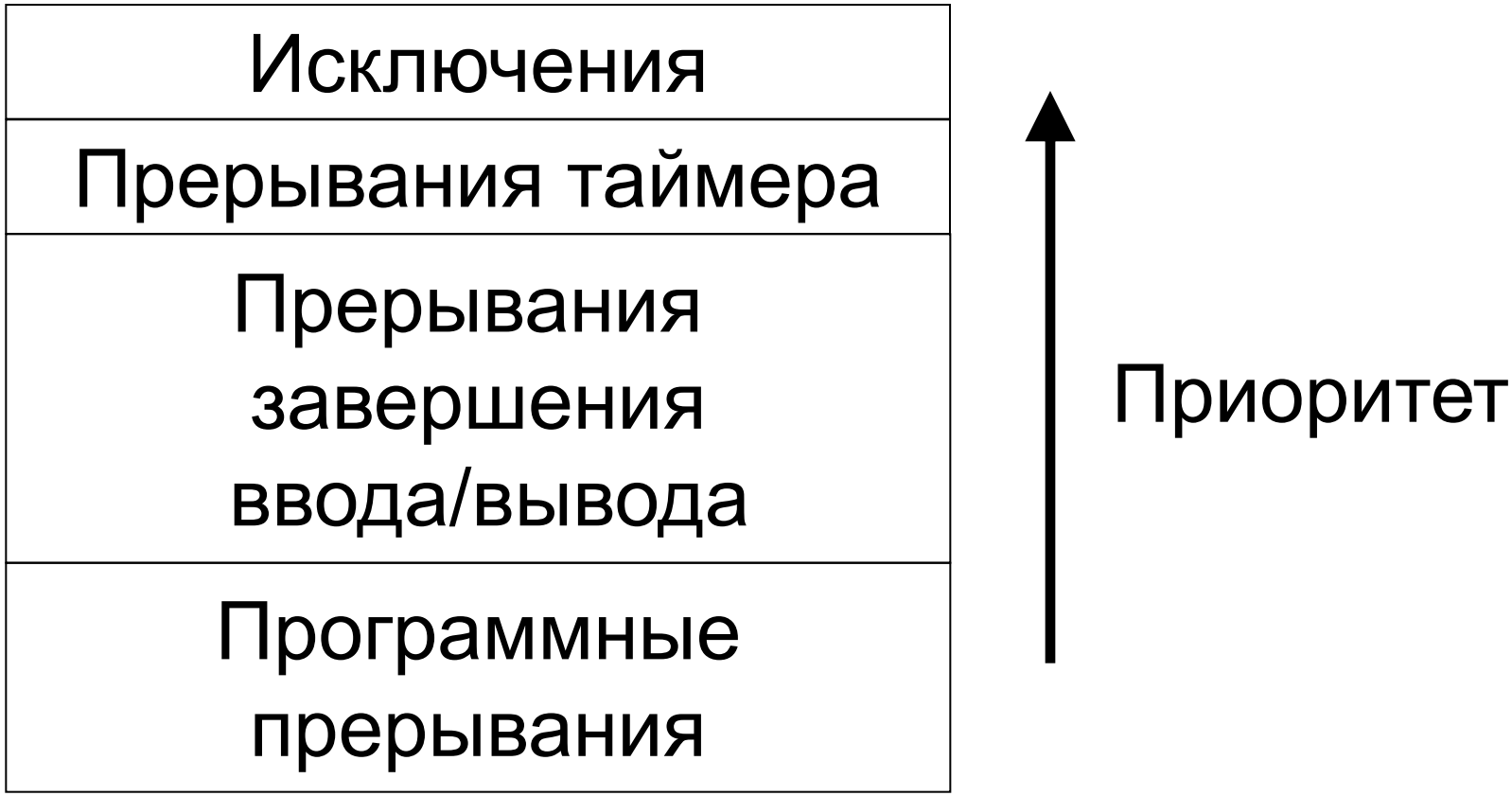
Вопрос для самопроверки

- Может ли прерванный процесс продолжить свое выполнение сразу после обработки прерывания таймера прерываний? (Да/Нет)
- Да. Планировщик процессов может загрузить контекст выполнения прерванного процесса, и он продолжит свое выполнение, если в списке готовых к выполнению процессов он окажется первым.

Концепции процесса

Классы прерываний

Классы прерываний



Исключение

Опр. Исключение (exception) – аппаратный сигнал, вызванный ошибкой. Исключения делятся на

- Аварийные завершения
- Ловушки
- Промахи

Аварийное завершение

Опр. Аварийное завершение (abort) – операция досрочного прекращения выполнения процесса, например, в результате аппаратного сбоя. Невосстановимая ошибка процесса. Имеет наивысший приоритет.

Ловушка

Опр. Ловушка (trap) – исключение, возникающее, например, в результате ошибок переполнения (когда значение, которое должно быть сохранено в регистре, превышает его разрядность). Ловушка перезапускает процесс, начиная с инструкции следующей за той, которая вызвала исключение.

Промах

Опр. Промах (fault) – вид исключения, вызванного, например, запрещенной операцией доступа к памяти. Некоторые промахи могут быть исправлены с помощью соответствующих обработчиков исключений (например, страничный промах). Тогда возобновление работы процесса начинается с инструкции, вызвавшей промах.

Прерывания таймера прерываний

Опр. Прерывания таймера прерываний (interval timer interrupt) – сигнал, вызываемый таймером прерываний по истечении выделенного процессу кванта времени.

Прерывания завершения ввода/вывода

Опр. Прерывания завершения ввода/вывода (I/O completion interrupt) – сигнал, выдаваемый устройством по окончании обслуживания запроса ввода/вывода.

Программные прерывания

Опр. Программные прерывания (software-generated interrupt) – командные инструкции, использующиеся процессами пользователя для вызова системных функций. Переключают процессор из пользовательского режима в режим ядра.

Маскирование прерываний

Опр. Маскирование прерываний (mask interrupt) – при маскировании определенного типа прерываний, прерывания указанного типа не доставляются процессу, замаскировавшему их. В этом случае прерывания либо устанавливаются в очередь для последующей доставки, либо удаляются процессором.

Вопрос для самопроверки

- Верно ли, что программные прерывания имеют самый низкий приоритет? (Да/Нет)

Вопрос для самопроверки

- Верно ли, что программные прерывания имеют самый низкий приоритет? (Да/Нет)
- Да. Сигналы, поступающие от оборудования, приоритетнее вызова процессами пользователя системных функций, так как если оборудование занято, или работает не правильно, эти вызовы могут быть не обслужены.

Вопрос для самопроверки

- Верно ли, что ядро системы всегда обязано реагировать на прерывания, даже если сильно перегружено? (Да/Нет)

Вопрос для самопроверки

- Верно ли, что ядро системы всегда обязано реагировать на прерывания, даже если сильно перегружено? (Да/Нет)
- Нет. Большинство процессоров позволяют ядру маскировать прерывания определенных типов. Тогда процессор просто игнорирует прерывания данного типа либо сохраняет их в очереди отложенных прерываний.

Концепции процесса

Взаимодействие процессов сигналами

Сигналы

Опр. Сигналы (signals) – программные прерывания, уведомляющие процесс о наступлении определенного события, либо возникновении ошибки.

Сигналы

- Не позволяют передавать данные
- При поступлении сигнала операционная система определяет
  - Кому предназначен данный сигнал
  - Как этот процесс должен на него реагировать

Варианты реакции процесса на сигнал

- Перехват
- Игнорирование
- Маскирование

Перехват

Опр. Перехват (catch) – процесс определяет процедуру, вызываемую операционной системой в случае поступления сигнала.

Игнорирование

Опр. Игнорирование (ignore) – процесс перекладывает ответственность за выполнение действия по умолчанию по обработке сигнала на операционную систему. Чаще всего – это аварийное завершение процесса, либо приостановка его выполнения.

Маскирование

Опр. Маскирование (masking) – процесс маскирует сигнал определенного типа (например, сигнал приостановки) и операционная система блокирует сигналы данного типа до тех пор, пока маскирование не будет отключено.

Вопрос для самопроверки

- Позволяют ли сигналы передавать данные между процессами? (Да/Нет)

Вопрос для самопроверки

- Позволяют ли сигналы передавать данные между процессами? (Да/Нет)
- Нет. Это основной недостаток использования сигналов при взаимодействии процессов.

Вопрос для самопроверки

- Верно ли, что процесс может отреагировать на сигнал одним из трех возможных способов? (Да/Нет)

Вопрос для самопроверки

- Верно ли, что процесс может отреагировать на сигнал одним из трех возможных способов? (Да/Нет)
- Да. Процесс может перехватывать, игнорировать, либо маскировать сигналы определенного типа.

Концепции процесса

Взаимодействие процессов путем передачи сообщений

Передача сообщений

Опр. Передача сообщений (massage passing) – механизм, позволяющий процессам общаться путем обмена данными. Прием и отправка сообщений обычно реализуется в виде вызова системных функций вида

`send(reciverProcess, message)`

`receive(senderProcess, message)`

Передача сообщений

- Блокирующая
- Неблокирующая

Блокирующая передача

Опр. Блокирующая передача (blocking send) – процесс вынужден ожидать до тех пор, пока сообщение не будет доставлено получателю, требуя подтверждения приема.

Неблокирующая передача

Опр. Неблокирующая передача (nonblocking send) – процесс отправитель может продолжить выполнение других операций даже если сообщение еще не было доставлено получателю.

Пр. Отсылка процессом данных на занятый сервер печати.

Канал

Опр. Канал (pipe) – механизм передачи сообщений, формирующий прямой поток данных между двумя процессами. Реализуется в виде защищенной операционной системой области памяти, которая выступает в качестве буфера.

Вопрос для самопроверки

- Верно ли, что в распределенных системах вместо сигналов обычно используется технология передачи сообщений? (Да/Нет)

Вопрос для самопроверки

- Верно ли, что в распределенных системах вместо сигналов обычно используется технология передачи сообщений? (Да/Нет)
- Да. Как правило, типы передаваемых сигналов зависят от архитектуры системы, что чревато несовместимостью сигналов разных компьютеров. Кроме того, сигналы не позволяют компьютерам обмениваться данными, без чего не может обойтись большинство распределенных систем.

Вопрос для самопроверки

- Может ли процесс, выполнивший блокирующую передачу, не получить подтверждения приема сообщения? (Да/Нет)

Вопрос для самопроверки

- Может ли процесс, выполнивший блокирующую передачу, не получить подтверждения приема сообщения? (Да/Нет)
- Да. Он может не получить подтверждения о доставке, что приведет к бесконечному блокированию процесса. Для решения этой проблемы используется механизм тайм-аута – если подтверждение не поступит через определенный промежуток времени, передача сообщения будет повторена.