Лекция 3 - 4

ОПЕРАЦИОННИ СИСТЕМИ

гл. ас. Р. Начева

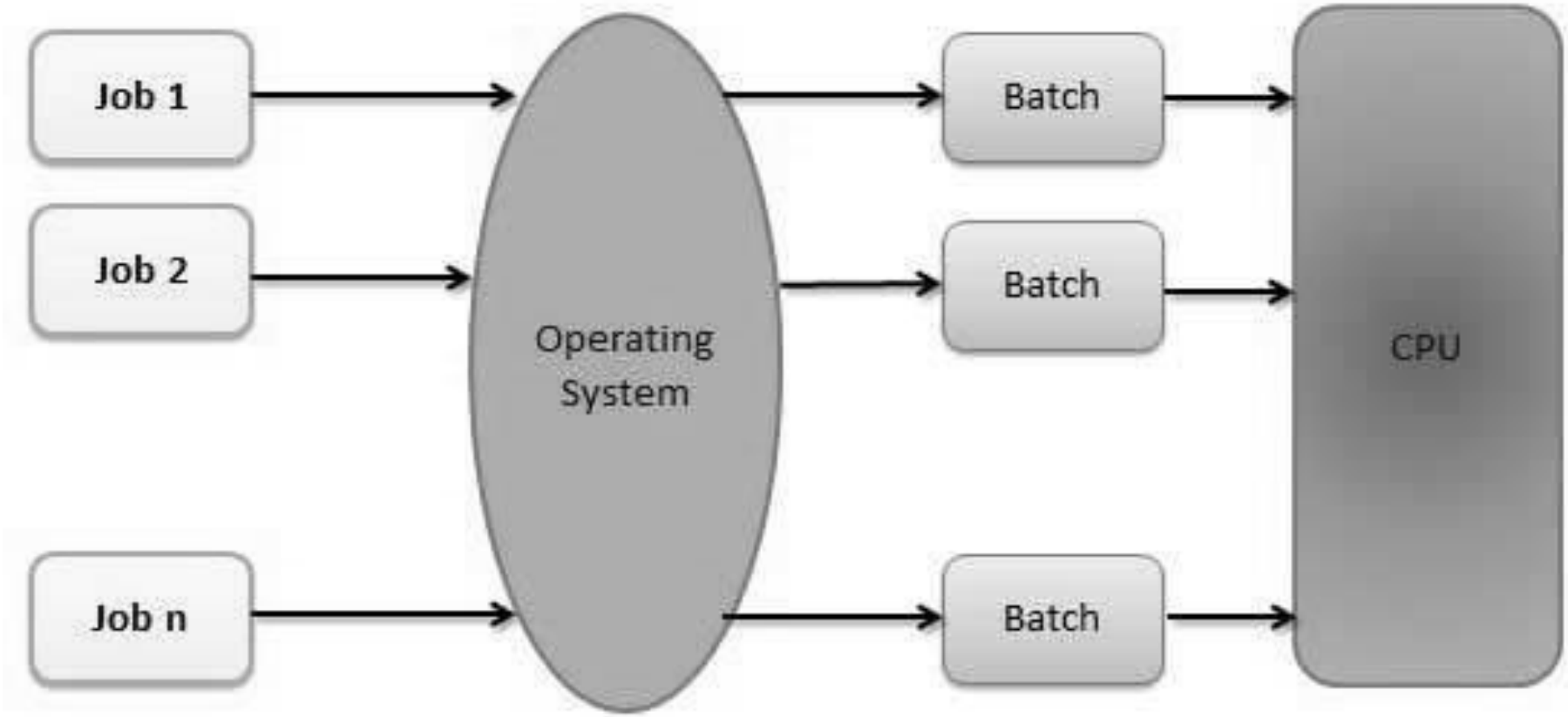
Катедра “Информатика”, ИУ - Варна

Съдържание на лекцията

- Основни характеристики на ОС
- Базови механизми на ОС
- Управление на процеси и нишки – същност, механизми за разпределяне на ресурсите

Основни характеристики на ОС

Пакетна обработка



Основни характеристики на ОС

Пакетна обработка

Техника, при която ОС обединява програмите и данните в пакети, преди да започне обработката. Операционната система извършва следните дейности, свързани с пакетна обработка:

- определя задача (job), която съдържа предварително дефинирана последователност от команди, програми и данни, обединени в едно цяло;
- поддържа определен брой задачи в паметта;
- задачите се обработват по реда на постъпване, т.е. когато дадена задача завърши изпълнението си, оперативната памет се освобождава и резултатът се запазва в spool за последващ печат или обработка.

SPOOL (simultaneous peripheral operations on-line) - механизъм или процес, при който данните временно се поставят в буфер за използване и изпълнение от устройство, програма или система. Данните се изпращат до и се съхраняват в паметта или друго запаметяващо устройство, докато програмата или компютърът не го поиска за изпълнение.

Основни характеристики на ОС

Пакетна обработка

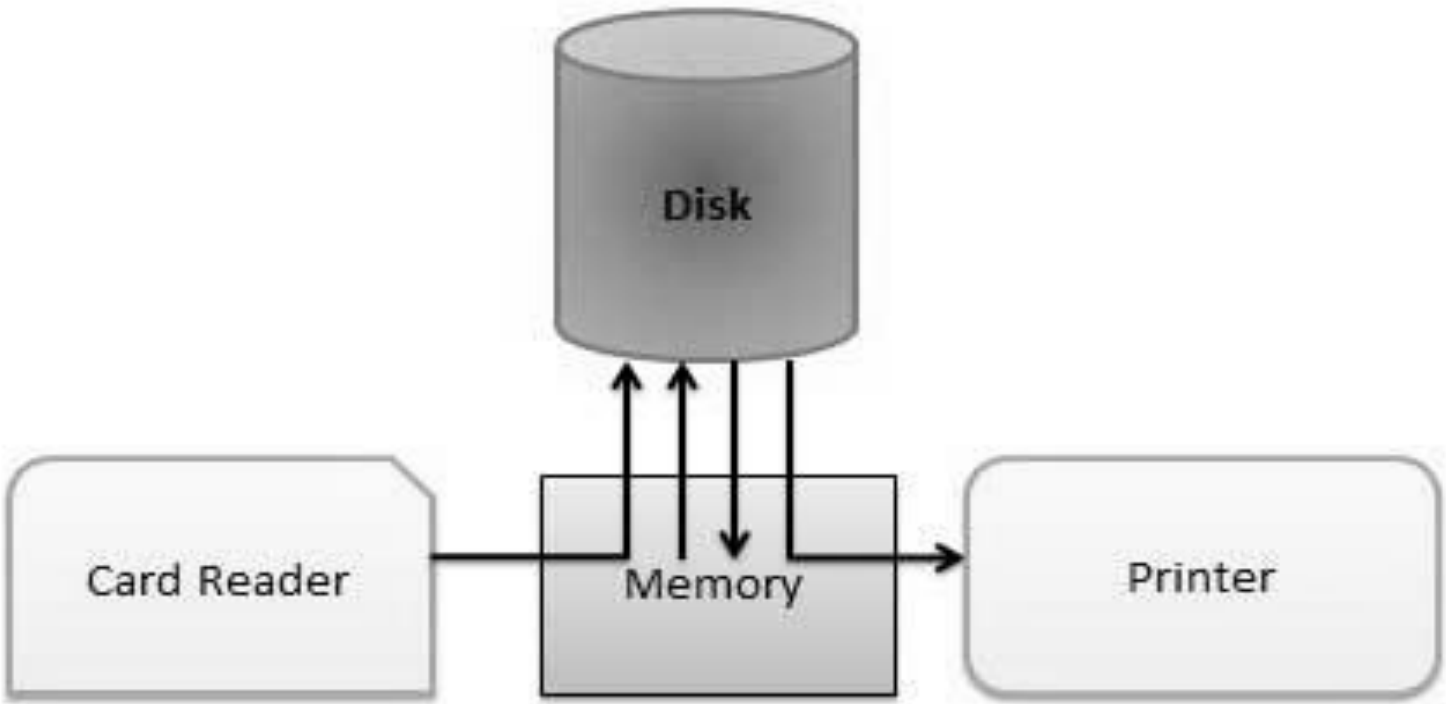
Спулингът е вид опашка за заявки, в която данните, инструкциите и процесите от множество източници се натрупват за по-късно изпълнение. Обикновено се поддържа за физическата памет на компютъра (RAM - Random Access Memory), буферите или прекъсванията за В/И устройства. Spool-ът се обработва по FIFO метод, т.е. първата инструкция в опашката ще бъде и първата, която ще бъде изведена и изпълнена.

Чести приложения на Spooling:

- при обработване на документи за принтиране;
- при обработване на задачи веднага след като се освободят системни ресурси;
- при изпращане на имейли и др.

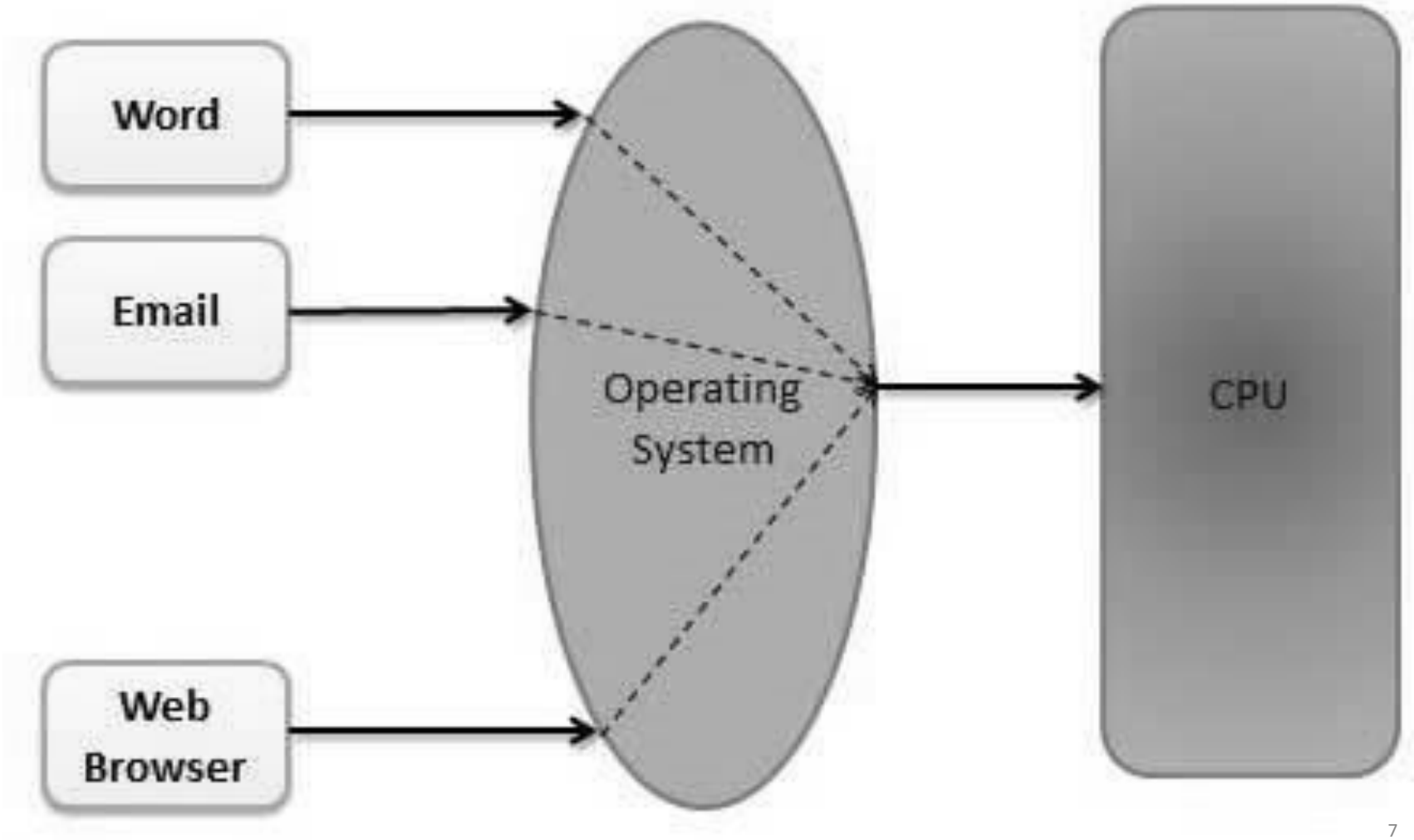
Основни характеристики на ОС

Spooling



Основни характеристики на ОС

Многозадачност



Основни характеристики на ОС

Многозадачност

Няколко задачи се изпълняват едновременно от процесора чрез превключване между тях. Превключването се извършва толкова често, че потребителите могат да взаимодействат с всяка програма, докато тя работи.

Специфики на многозадачната работа на ОС:

- потребителят дава инструкции на операционната система или на програмата директно и получава незабавен отговор;
- ОС обработва няколко операции / изпълнява няколко програми едновременно;
- многозадачните ОС са известни също като time-sharing systems;
- осигурява се интерактивно използване на компютърна система;
- използва се концепцията за планиране на работата на процесора (CPU scheduling) и мултипрограмирането (multiprogramming), за да се осигури на всеки потребител малка част от общото процесорно време;
- всеки потребител има поне една програма в паметта.

Основни характеристики на ОС

Многозадачност

Специфики на многозадачната работа на ОС:

- програма, която е заредена в паметта и се изпълнява, обикновено се нарича процес;
- входно-изходните устройства обикновено работят по-бавно, през това време процесорът може да бъде използван от друг процес;
- ОС дава възможност за многопотребителска работа на един компютър – за всяко действие или команда в системата се заема малко процесорно време.
- разрешава се проблема за планиране на задачите, които се изпълняват от процесора – извършва се превключване от една задача към друга чрез реализиране на прекъсвания (interruptions) – сигнали, които „поставят“ в буфер текущата активност (задачи) и се връща изпълнението на друга задача, която е била изпълнявана по-рано. Това е т. нар. „превключване на контекста“ (context switch).

Основни характеристики на ОС

Многозадачност

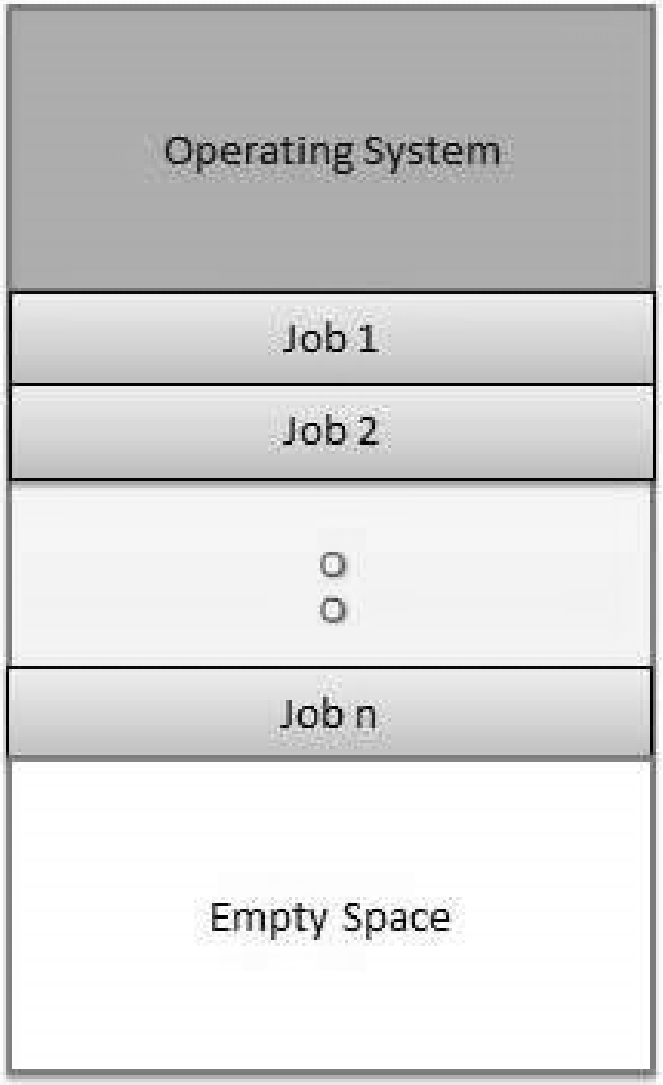
Типове:

- Кооперативна - след като дадена задача заеме процесора, той не може да й бъде отнет;
- Приемптивна (preemptive) - ОС може да отнеме контрола на процесора от дадена задача без нейното „съгласие“ и да го предаде на друга задача.

Основни характеристики на ОС

Мултипрограмиране

Споделянето на процесора от две или повече програми, заредени в паметта по едно и също време, се нарича мултипрограмиране. Мултипрограмирането предполага споделяне на процесорното време. То подобрява организацията на задачите на ОС, така че процесорът винаги да изпълнява една от тях.



Основни характеристики на ОС

Мултипрограмиране

Основни характеристики:

- ОС запазва няколко задачи в паметта едновременно;
- този набор от задачи е подмножество на задачите, съхранявани в опашката за задачи (job pool);
- ОС избира и започва да изпълнява една от задачите в паметта;
- наблюдава се състоянието на всички активни програми и системни ресурси, използвайки програми за управление на паметта, за да гарантират, че процесорът никога не е неактивен, освен ако няма задачи за обработка;
- ефективно използване на процесора, изискващо планиране.

Основни характеристики на ОС

Многозадачност и Мултипрограмиране

Основни разлики:

Многозадачност	Мултипрограмиране
Програмата се разделя на фиксиран брой страници.	Цялата програма се зарежда в паметта.
Превключването на контекста се извършва след определен интервал от време.	Не е фиксиран интервал от време за превключване.
Поддържа се в съвременните ОС.	Остаряла концепция, използвана като база за реализиране на многозадачност.

Основни характеристики на ОС

Транслатори

«Превеждат» инструкциите на програмите на машинен език, съставен от серия от битове (0 и 1), и обратно. Могат да са: асемблери, компилатори и интепретатори.

- **Асемблер**: преобразува асемблерен в изпълним от процесора код (асемблиране на кода). «Асемблерен език е език за програмиране от ниско ниво, предназначен за компютри или други програмируеми устройства, при които има много строго (обикновено едно към едно) съответствие между езика и машинните кодови инструкции на процесорната архитектура. Всяка процесорна архитектура има свой собствен специфичен асемблерен език, за разлика от повечето езици за програмиране от високо ниво, които са съвместими с различни процесорни архитектури, но изискват превеждане или компилиране.» [1]

[1] Източник: https://bg.wikipedia.org/wiki/Асемблерен_език

Основни характеристики на ОС

Транслатори

- **Компилатор**: преобразува изходния код на програма, написана на език на високо ниво в език на по-ниско ниво (асемблерен език), предназначен за «конкретен процесор или процесорна фамилия, както и междинен език за конкретна виртуална машина или друг език от високо ниво. Когато се компилира до машинен език, крайният продукт е изпълнима програма или обектен код.» [1]

[1] Източник: https://bg.wikipedia.org/wiki/Компилатор

Основни характеристики на ОС

Транслатори

- **Интерпретатор**: «извършва последователен анализ на командите от изходния код, непосредствено ги превежда на машинен език и изпълнява. По своята същност интерпретаторът е транслатор. Програмите обикновено се пишат на език от високо ниво, който трябва да бъде превърнат в машинен език, за да могат да бъдат изпълнени от процесора. Това превръщане се извършва от компилатор или от интерпретатор. За да може програмата да се изпълнява компилаторът превежда изходния код само веднъж, докато интерпретаторът извършва превода при всяко нейно стартиране.» [1] Той на «разглежда» цялата програма, преди да може да започне изпълнението на кода.

[1] Източник: https://bg.wikipedia.org/wiki/Интерпретатор_(софтуер)

Базови механизми на ОС

Управление на прихващанията

- **Събития**: Прекъсване (interruption) и Изключение (exception)
- Те показват, че съществува условие някъде в ОС, хардуера или в рамките на изпълняваната в момента задача, което изисква заемане на процесорно време. Събитията обикновено водят до принудително прехвърляне на изпълнение от текущо изпълняваната задача към специална софтуерна програма, която се занимава с обработване на прекъсванията и съответно на изключенията (interrupt handler и съответно exception handler).
- Действието, предприето от процесора в отговор на прекъсване или изключение се нарича обслужване или обработка на прекъсването или изключението.

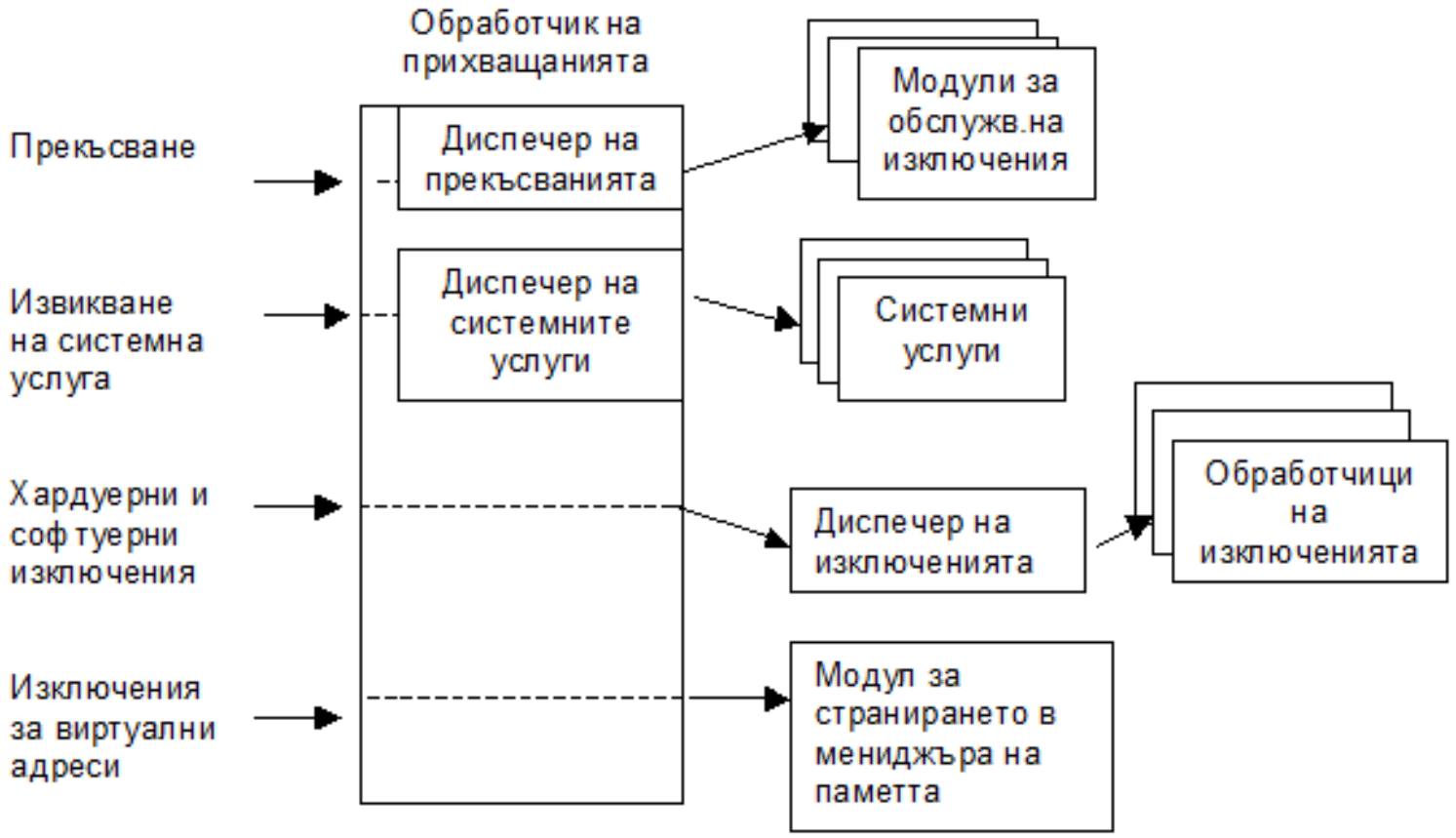
Базови механизми на ОС

Управление на прихващанията

- При настъпване на прекъсване или на изключение, текущо изпълняваната задача се спира, докато процесорът изпълнява обработчикът на прекъсвания или на изключения. Когато приключи изпълнението на обработчика, процесорът възобновява изпълнението на прекъсната задача. Това се случва без да се прекъсне програмата, освен ако възстановяването от изключението или прекъсването не е било възможно и е довело до спиране на текущо изпълняваната програма.
- Прекъсванията и изключенията настъпват на софтуерно и хардуерно ниво.

Базови механизми на ОС

Управление на прихващанията



Източник: лекционен материал, доц. д-р Надежда Филипова

Базови механизми на ОС

Управление на прихващанията

- При възникване на изключение или прекъсване, се прихваща преходът на изпълнение от потребителски към системен режим, където се обработва изключението или прекъсването. В тази ситуация изпълняващият се в момента процес трябва първо да бъде запазен в паметта. След това ядрото вече е готово да обработи изключението / прекъсването в следния ред:
  - определя причината за изключението / прекъсването;
  - обработва изключението / прекъсването.
- При обработване на изключението / прекъсването ядрото изпълнява следните стъпки:
  - избира процес за възстановяване и възобновяване;
  - възстановява контекста (състоянието) на избрания процес;
  - възобновява изпълнението на избрания процес.

Базови механизми на ОС

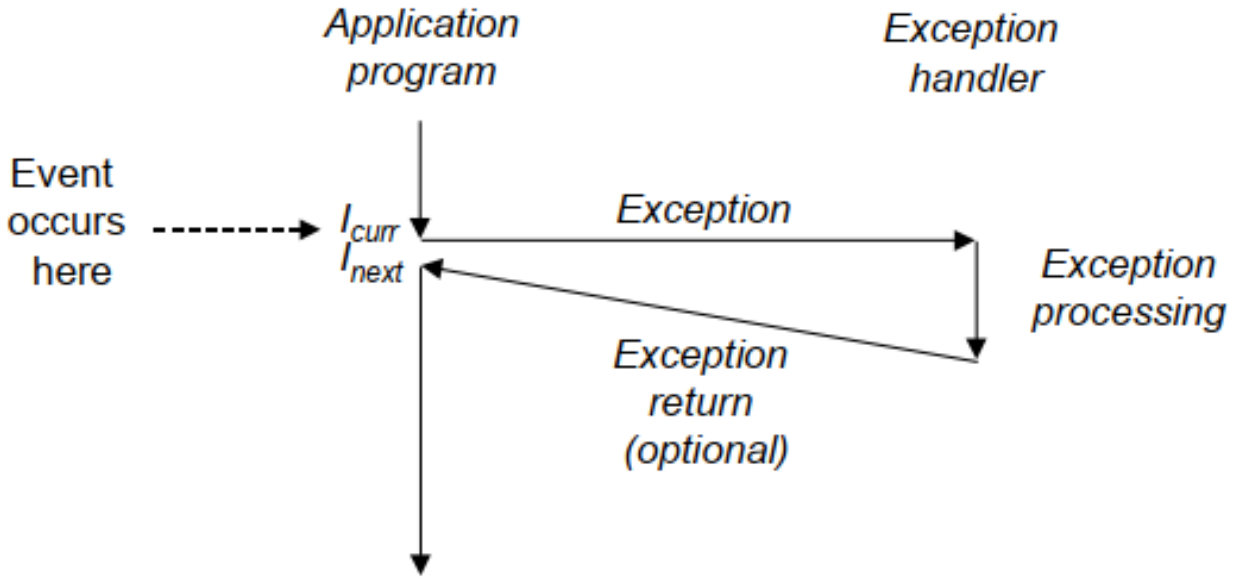
Управление на прихващанията

- **Изключения**: Изключенията настъпват, когато процесорът открие грешка при изпълнение на инструкция на програма, като например разделяне на нула. Също така те могат да бъдат причинени от неизправности на хардуера. Други примери за грешки са свързаните с виртуалната памет, препълване в следствие на извършваните аритметичните операции (arithmetic overflow), проблеми с входно-изходните устройства и т.н.
- На всяко изключение в системата се присвоява уникално неотрицателно цяло число (ID). Някои от тези номера са зададени от производителя на процесора. Други номера се задават от разработчиците на ядрото на операционната система.
- При зареждане ОС инициализира таблицата на изключенията (exception table), която съдържа адреса на обработчика на изключенията.

Базови механизми на ОС

Управление на прихващанията

Изключения: Промяната в състоянието на процесора (събитие) води до внезапно прехвърляне на управлението (изключение) от приложната програма към обработчика на изключения. След като приключи обработката, обработчикът или връща контрола на прекъснатата програма, или прекратява изпълнението ѝ.



Базови механизми на ОС

Управление на прихващанията

Видове изключения: прихващане (trap), грешка (fault), прекратяване (abort).

Прихващане: изключение, което се докладва веднага след изпълнението на инструкцията за прихващане. Този тип изключения дават възможност изпълнението на дадена задача да продължи без тя да бъде прекъсната. Настъпва например при невалиден адрес в паметта. Обикновено води до превключване към системен режим, при което операционната система извършва набор от действия, преди да върне управлението на прихванатия процес. Прихващане на системен процес е по-сериозен проблем в сравнение с прихващане на потребителски процес, а в някои ОС е фатално.

Базови механизми на ОС

Управление на прихващанията

Видове изключения: прихващане (trap), грешка (fault), прекратяване (abort).

Грешка: изключение, което обикновено може да бъде коригирано и след това позволява на програмата да се рестартира и да възстанови нормалната си работа. При възникване на грешка процесорът възстановява състоянието на машината до това, в което е била преди началото на изпълнението на грешната инструкция. Обработчикът на изключенията поддържа адрес (уникален код) на грешната инструкция. Ако обработчикът може да коригира грешката, той връща контрола към програмата. В противен случай обработчикът прекратява програмата, която е причинила грешката.

Базови механизми на ОС

Управление на прихващанията

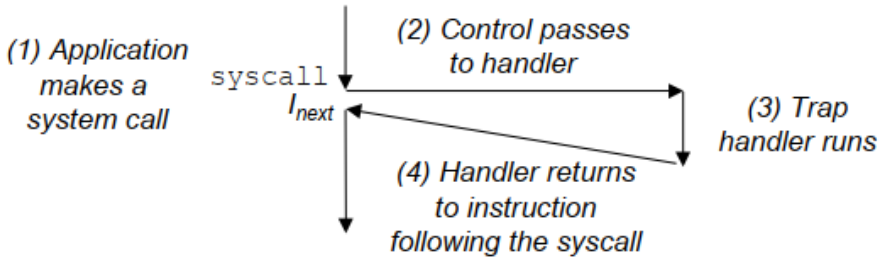
Видове изключения: прихващане (trap), грешка (fault), прекратяване (abort).

Прекратяване: изключение, което не винаги отчита точното местоположение на грешната инструкция и не позволява рестартиране на програмата или задачата, която е причинила изключението. Използва се при възникване на сериозни грешки в системен режим на ОС или при хардуерни грешки.

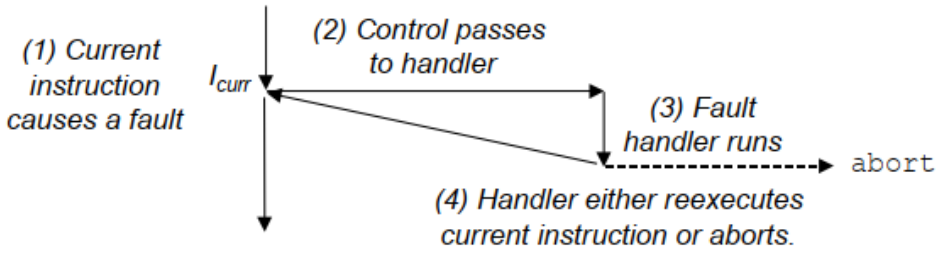
Базови механизми на ОС

Управление на прихващанията

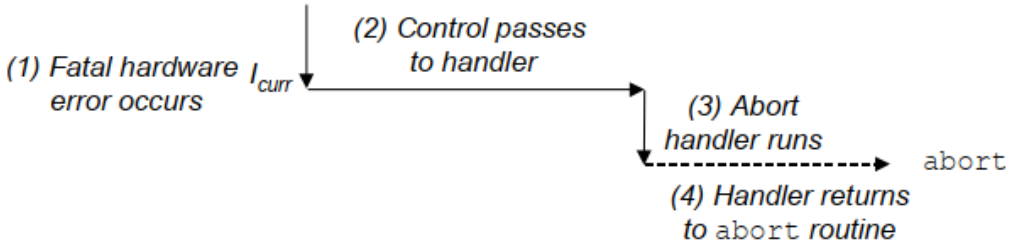
Прихващане (trap)



Грешка (fault)



Прекратяване (abort)



Базови механизми на ОС

Синхронизация

- Отнася се до синхронизиране на процесите, което е необходимо при поделяне на едни и същи ресурси с цел намаляване на конкуренцията за ресурса между тях. Потребности от синхронизацията са свързани с:
  - Разделяне (fork) и обединение (join): Когато дадена задача достигне определена точка от изпълнението си, тя се разделя на N подзадачи, които след това се обслужват от n задачи. Всяка подзадача изчаква докато всички други подзадачи се обработват. След това отново се обединяват.
  - Производител-потребител (Producer-Consumer): процесът на потребителя зависи от процеса на производителя, докато се произведат необходимите данни.

Базови механизми на ОС

Синхронизация

- Примери за синхронизация в ОС Windows:
  - **Маскиране на прекъсвания (interrupt masks)** - защитават достъпа до глобални ресурси (критична секция) на еднопроцесорни системи; маскира се хардуерното прекъсване чрез код в поддържан за целта специален регистър (interrupt mask register - IMR);
  - **Spinlocks** – при многопроцесорни системи се предотвратява отнемането на контрола на дадена нишка, като се блокира временно изпълнението ѝ. Обикновено това става в системен режим;
  - **Диспечери (dispatchers)** – мутекс, семафори, събития, таймери. *Семофорът* е променлив или абстрактен тип данни, използван за контрол на достъпа до общ ресурс чрез множество процеси в многозадачна ОС. Това е променлива, която се използва за решаване на проблеми с критичните секциите (участък от кода, в който се работи с общите за всички процеси данни). *Мутексът* (mutual exclusion) – взаимно изключване, е механизъм на контрола на конкурентността на нишките за общ ресурс. Нишката, която се изпълнява, никога не влиза в критичната си секция по едно и също време с друга нишка.

Управление на процеси и нишки

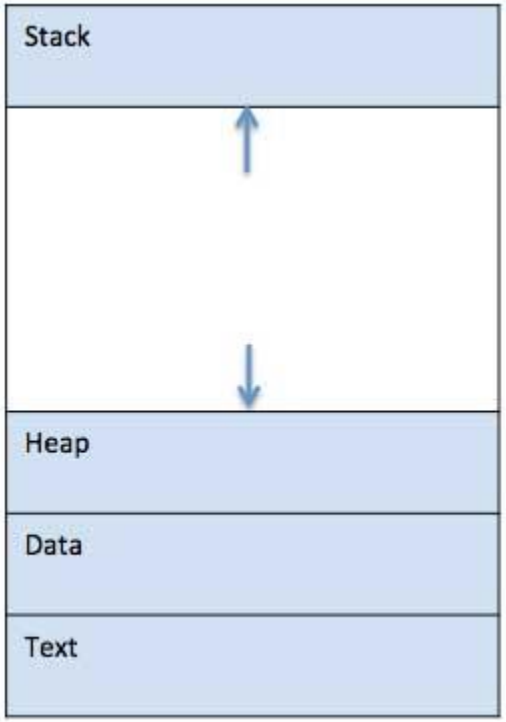
Същност

- **Неформални дефиниции**: програма, заредена в оперативната памет, в процес на изпълнение; програмна единица. Синоними: задача, единица работа.
- **Определение**: „Процесът е двойката „процесор-програма“ при изпълнение, който включва стек, текущите стойности на брояча на команди и други регистри на процесора, структури от данни от ОС за управление на изпълнението му (блок за управление, опашки, таблици и др.)“[1].
- Процесът е „активна“ същност, докато програмата се счита за „пасивна“. Атрибутите на процеса включват хардуерно състояние, памет, CPU и т.н. Програмата е последователност от краен брой команди (инструкции).

[1]**Източник**: Николов, Л. (2009). Операционни системи. Сиела, с. 39.

Управление на процеси и нишки

Същност

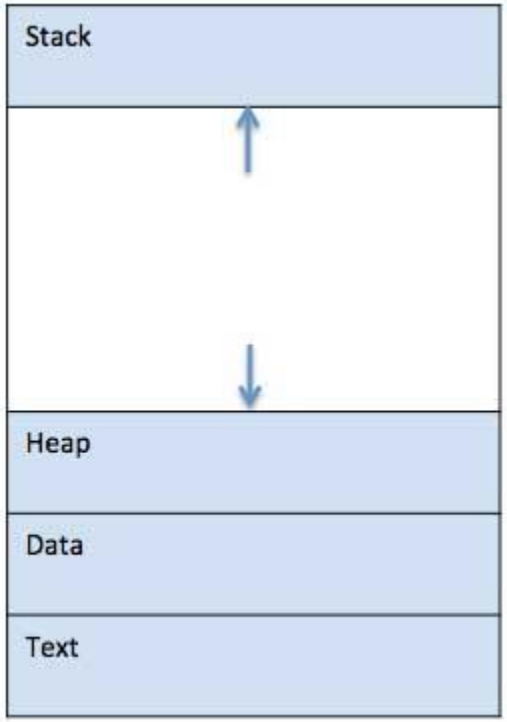


Паметта, заета от процеса, е разделена на:

- **Стек (Stack)** - съдържа временни данни като параметри на метод / функция, адрес за връщане и локални променливи. Всеки път, когато дадена функция декларира нова променлива, тя се поставя в стека. След това всеки път, когато дадена функция приключи, всичките ѝ променливи, които са в стека, се освобождават (т.е. те се изтриват). След освобождаването на променлива от стека, тази област на паметта става достъпна за други променливи на стека. Потребителят не заделя «ръчно» памет, а се управлява от ОС. Чрез стека се органицира оперативната памет за извършваните обработки от процесора. Променливите в стека са локални.

Управление на процеси и нишки

Същност



Паметта, заета от процеса, е разделена на:

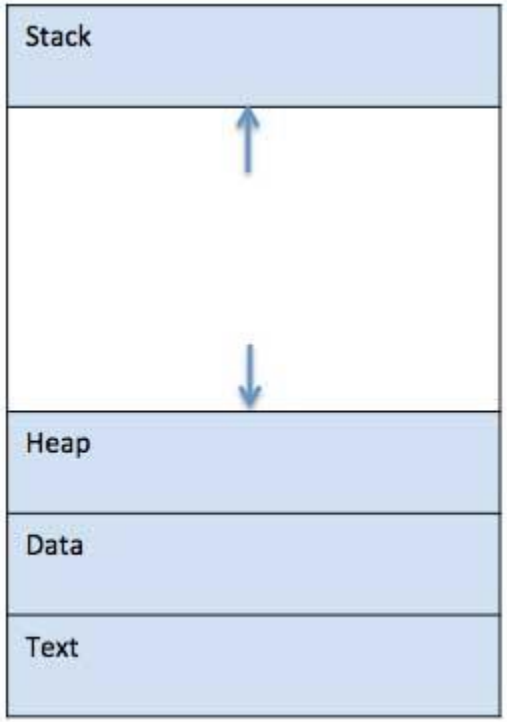
- **Динамична памет (Heap)** – управлява се чрез извиквания (calls), т.е. управлението не е автоматично. Променливите се достъпват само глобално, достъпът до heap е по-бавен в сравнение със стека. Не се гарантира ефективното използване на адресното пространство в паметта. Паметта се разделя на блокове, които се освобождават посредством вградените в езика на високо ниво функции. Т.е. програмистите са отговарни за разпределянето и освобождаването на променливите в динамичната памет.

Управление на процеси и нишки

Същност

Паметта, заета от процеса, е разделена на:

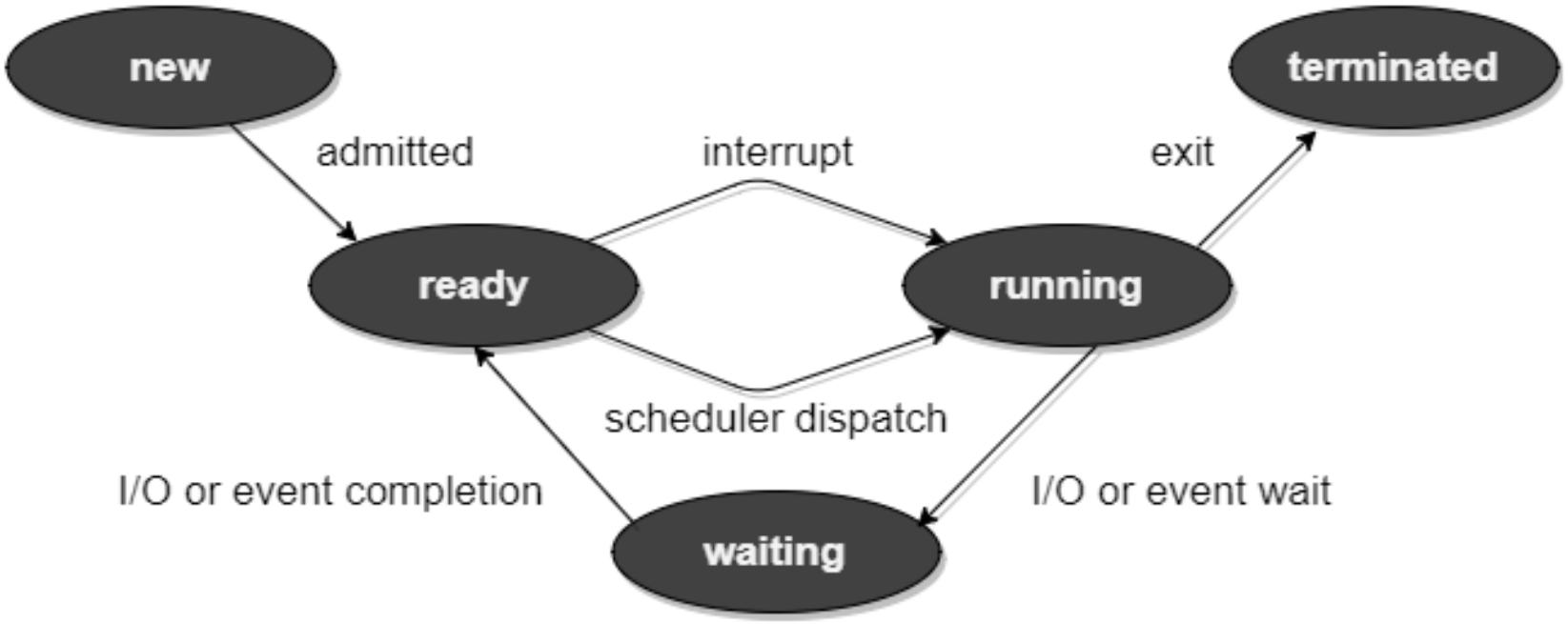
- **Данни (Data section)** - състои се от глобални и статични променливи, разпределени и инициализирани преди изпълнението на основната.
- **Текст (Text section)** - съставена е от компилирания програмен код, който се чете от енергонезависима памет при стартиране на програмата. Съдържа регистри на процесора (processor registers) и брояч на програми (Program Counter).



Управление на процеси и нишки

Същност

Процесът преминава през различни състояния по време на неговото изпълнение или това са операциите, които се извършват върху него:



Управление на процеси и нишки

Състояния на процеса

№	Състояние и описание
1	**Създаване на процес (START)** Първоначалното състояние, зареждане на процеса в паметта.
2	**Готовност за изпълнение (READY)** Процесът чака да заемането на процесорно време, за да се изпълнят зададените инструкции. Процесът може да е в това състояние след създаването си или при изпълнението си, ако е прекъснат за заемане на CPU от друг процес.
3	**Изпълнение (RUNNING)** Процесът е подаден към процесора от планировчика на ОС (scheduler) за изпълнение на инструкциите на процеса.
4	**Прекъсване на изпълнението (WAITING)** Процесът се премества в състояние на изчакване, ако трябва да изчака заемането на ресурс. Например изчакване на вход от потребителя или на достъп до файл.
5	**Унищожаване на процес (TERMINATED)** След като процесът завърши изпълнението си или се прекрати от операционната система, той се премества в крайното състояние, където изчаква да бъде премахнат от оперативната памет.

Управление на процеси и нишки

Основни елементи на всеки процес

- Блок за управление на процеса (PCB – Process Control Block);
- Идентификатор на процеса (process ID);
- Идентификатор на родителския процес;
- Квота за памет;
- Дескриптор на виртуалното адресно пространство;
- Информация за виртуалната памет;
- Блок за обкръжението на процеса (PEB – process environment block).

Управление на процеси и нишки

Блок за управление на процеса (PCB – Process Control Block)

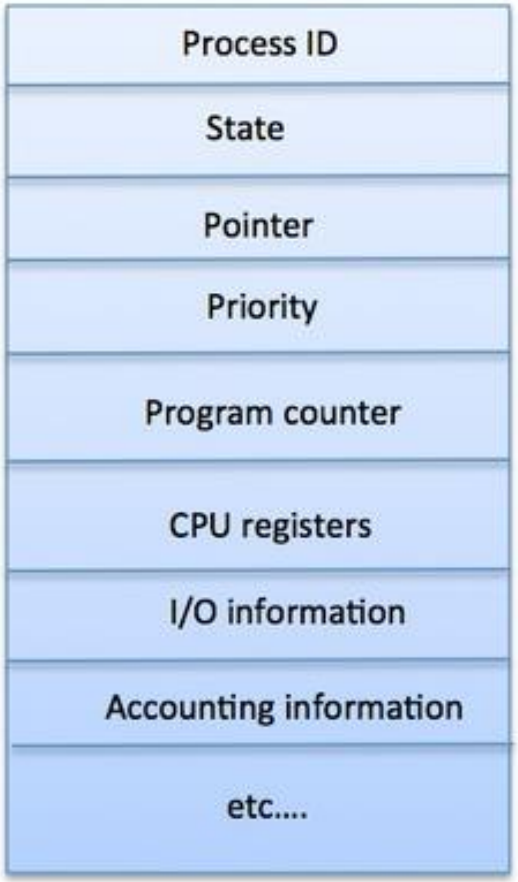
PCB е структура от данни, поддържана от операционната система за всеки процес. Тя се идентифицира с цяло число (ID на процеса - PID) и съхранява цялата информация, необходима за проследяване на процеса:

- **Състояние на процеса (Process State):** създаване, изпълнение, унищожаване и т.н.
- **Номер на процес (Process ID):** уникален идентификатор;
- **Регистри на процесора и брояч на програмите (CPU registers / Program Counter):** броячът съдържа адреса на следващата инструкция, която трябва да се изпълни за този процес;
- **Информация за разпределянето на процесора (CPU Scheduling information):** приоритет и указатели в паметта за процесите;
- **Управление на паметта (Memory Management information):** частите (страниците), на които е разделен процеса в оперативната памет;
- **Статистическа информация:** заемано процесорно време в потребителски и системен режим, например.
- **Информация за В/И:** например използвани В/И устройства.

Управление на процеси и нишки

Блок за управление на процеса (PCB – Process Control Block)

Структурата на Блока за управление зависи от архитектурата на ОС. Следната схема е обобщаваща, но може да варира според ОС. Той поддържа от ОС, докато процесът е активен. След унищожаването му се изтрива.



Управление на процеси и нишки

Механизми за разпределяне на ресурсите

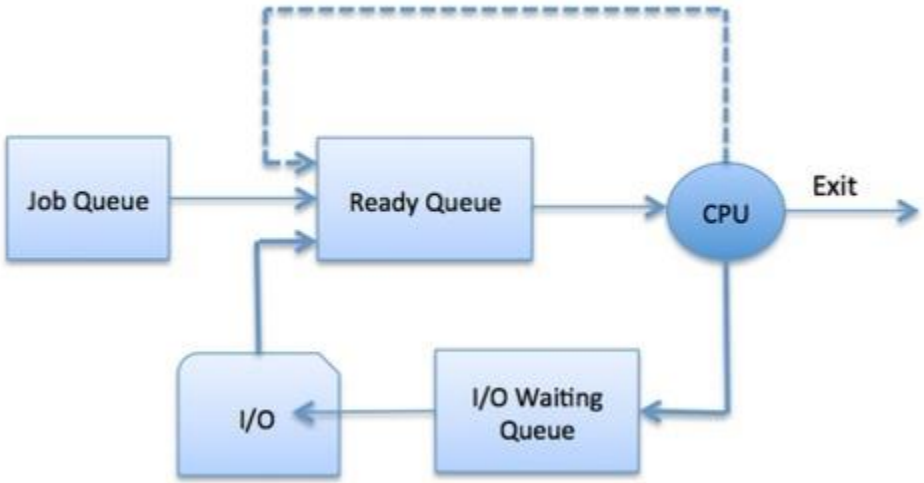
- **Мениджърът на процесите (Системата за управление на процеси)** е компонентът на ОС, който е отговорен за създаване и унищожаване на процеси и нишки, поддържане на тяхната работа, промяна на приоритета им. ОС поддържат различни механизми за разпределяне на ресурсите (Process scheduling). Основната цел на планирането на процесите е поддържане на процесора зает през цялото време и осигуряване на минимално време за изпълнение на всички програми.
- ОС поддържа няколко опашки (буфера), които се използват за планиране на заеманото процесорно време. Поддържа се и отделна опашка за всяко от състоянията на процеса. PCB на всички процеси в едно и също състояние се поставят в една и съща опашка. Когато състоянието на процеса бъде променено, неговият PCB се премества в нова й опашка.

Управление на процеси и нишки

Механизми за разпределяне на ресурсите

Видове опашки за планиране на процесите:

- Опашка на задачите (Job queue) - запазва всички процеси в системата.
- Опашка на изпълнените задачи (Ready queue) - поддържа набор от всички процеси, които се намират в основната памет, готови и чакащи за изпълнение. В тази опашка винаги се поставят новите процес.
- Опашки на устройства (Device queues) – поддържа процесите, които са блокирани поради липса на В/И устройство.



Управление на процеси и нишки

Механизми за разпределяне на ресурсите

Поддържат се следните основни типове:

- Първи постъпил, първи обслужен (First-Come, First-Served - FCFS)
- Най-кратката задача е първа (Shortest-Job-Next - SJN)
- Базиран на приоритета (Priority-based)
- Най-малко оставащо време (Shortest Remaining Time)
- Round Robin (RR)
- Базиран на йерархични нива на опашките (Multiple-Level Queues)

Управление на процеси и нишки

Механизми за разпределяне на ресурсите

Първи постъпил, първи обслужен

- Задачите се изпълняват на принципа „първи постъпил, първи обслужен“, който се базира на прицина FIFO - данните, добавени първо към опашката, са тези, които я напускат (се обработват) първо.
- Това е преемптивен алгоритъм за планиране на ресурсите.
- Лесен за разбиране и прилагане.
- Производителността от гл. т. на средното време на изчакване е ниска.

Управление на процеси и нишки

Механизми за разпределяне на ресурсите

Най-кратката задача е първа

- Това е преемптивен алгоритъм за планиране на ресурсите.
- Използва се от системите за пакетна обработка, при които времето, отделено от CPU за обработка е известно предварително.
- Най-добър подход за минимизиране на времето за изчакване.
- Процесорът трябва предварително да знае колко време ще отнеме изпълнението на процеса.

Управление на процеси и нишки

Механизми за разпределяне на ресурсите

Механизъм, базиран на приоритета

- Широко използван механизъм, при който на всеки процес се присвоява приоритет. Първо се изпълнява процесът с най-висок приоритет и т.н.
- Процеси с еднакъв приоритет се изпълняват на принципа "първи постъпил, първи обслужен".
- Приоритет може да бъде определен въз основа на изискванията за памет, изисквания за време за изпълнение или други изисквания към системните ресурси.

Управление на процеси и нишки

Механизми за разпределяне на ресурсите:

Най-малко оставащо време

- Процесорът е зает от дадена задача, която приключва след кратък период от време. Контролът може да бъде прехвърлен към друга задача, която следва да приключи по-скоро.
- Невъзможно е да се приложи в интерактивни системи, където необходимото CPU време не е известно.
- Често се използва в пакетни ОС, където времето за изпълнението на задачите е кратко.

Управление на процеси и нишки

Механизми за разпределяне на ресурсите:

Round Robin

- На всеки процес се предоставя с фиксирано време за изпълнение, който се нарича квант.
- След като процесът се изпълни за даден период от време, той се прекъсва и друг процес се изпълнява за определено време.
- Певключването на контекста се използва за запазване на състоянията на изпълняваните процеси.

Управление на процеси и нишки

Механизми за разпределяне на ресурсите

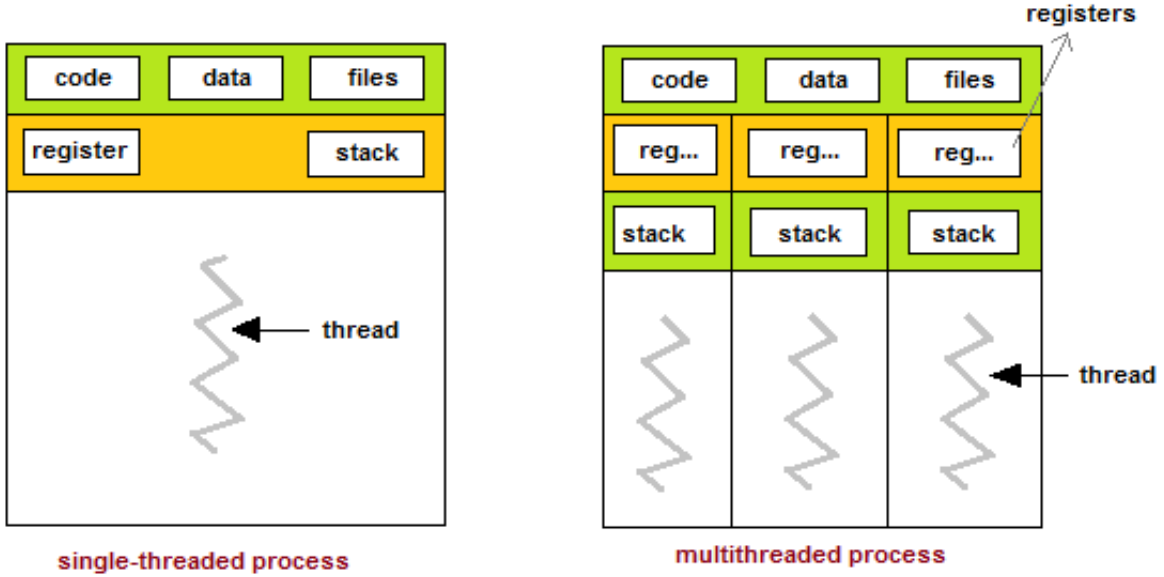
Базиран на йерархични нива на опашките

- Не е независим алгоритъм за разпределяне на ресурсите. Използват се други съществуващи алгоритми за групиране и планиране на задачите с общи характеристики.
- Поддържат се множество опашки за процеси с общи характеристики.
- Всяка опашка може да има свои собствени алгоритми за планиране.
- Задават се приоритети на всяка опашка.

Управление на процеси и нишки

Нишки – същност и структура

- Нишката е единица за изпълнение, която се състои от собствен програмен брояч, стек и набор от регистри. Те са начин за изпълнение на процесите чрез т. нар. паралелизъм. Процесорът превключва бързо между нишките, създавайки илюзия, че те се изпълняват паралелно.
- Процесите се разделят на множество нишки за ефективно реализиране на многозадачната работа на ОС. Приоритета на нишките се определя на база на приоритета на процеса и на нейния (относителен) приоритет.



Управление на процеси и нишки

Нишки – същност и структура

Има два вида нишки:

- **Изпълнявани в потребителски режим (User Threads)** – това са нишките, на които се разделят приложните програми, изпълнявани в ring3 (user mode).
- **Изпълнявани в системен решим (Kernel Threads)** – поддържат се от ядрото на операционната система. Всички модерни операционни системи поддържат нишки в системен режим, като позволяват на ядрото да изпълнява едновременнf няколко задачи и / или да обслужва едновременно множество системни извиквания.

Управление на процеси и нишки

Нишки – същност и структура

Сравнение между двата типа нишки:

№	Нишки в потребителски режим	Нишки в системен режим
1	По-бързо се създават и управляват.	По-бавни за създаване и управление.
2	Реализацията е чрез библиотека с нишки на потребителско ниво (от програмите).	Поддържат се от операционната система.
3	Може да се стартира от всяка една ОС.	Специфични са за ОС.

Управление на процеси и нишки

Нишки – същност и структура

Предимства на многонишковия подход:

- Минимизира се времето за превключване на контекста;
- Използването на нишки осигурява едновременност (concurrency) на процеса;
- Ефективна комуникация;
- Дават възможност за ефективното използване на многопроцесорни архитектури, поддържани за мащабни дейности, като например в научната сфера.

Управление на процеси и нишки

Нишки – същност и структура

Основни елементи на нишките:

- Блок за управление;
- Идентификатор на процеса, на който принадлежи нишката;
- Време на създаване и прекъсване на изпълнението на нишката;
- Начален адрес на нишката;
- Блок за обкръжението на нишката.

Допълнителна литература

- OS Sim (OS Concepts Simulator)
- CPUlator Computer System Simulator
- Operating System Concepts