Одеська Політехніка
Інститут комп`ютерних систем
Кафедра інформаційних систем
Дисципліна «Операційні системи»

Тема 1: Історія операційних систем (ОС)
Лекція 3: ОС для малих комп`ютерів та персональних комп`ютерів

Олександр А. Блажко,
доцент кафедри інформаційних систем,
E-mail: blazhko@ieee.org
Telegram-канал: t.me/Operating_Systems_IS

Одеса, 20 березня 2023 року

Складність використання комп'ютера

Як скоротити трудомісткість (складність) процесів:

- споживчого використання комп'ютера (спілкування людини з комп'ютером)?
- технічного використання (обслуговування, програмування) апаратних компонент комп'ютера?

Метод наукового пізнання допоможе?

Якщо так, то який?

Абстрагування - це ...

уявне виділення (концентрація уваги) :

- будь-якого об’єкта при видаленні (лат. *abstractio*) зв'язків цього об’єкта з іншими об’єктами;
- якої-небудь властивості об’єкта при видаленні інших його властивостей;

Абстрагування та структури даних

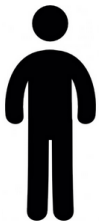
Сьогодні абстрагування - це один з 4-х принципів обєктно-орієнтованого програмування крім інкапсуляції, наслідування та поліморфізму

Абстрагування використовується ще у структурно-орієнтованому програмуванні при створенні структур даних як опис сутностів реального світу

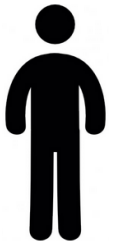
Мета створення інформаційної системи (ІС) визначає особливості абстрагування від сутності реального світу при створені її абстракної моделі

Приклад абстрагування сутності «Людина»:

- Якщо метою ІС є зменшення часу на пошук друзів, якою буде структура абстракції «друг»?
- Якщо метою ІС є зменшення часу на пошук вчителя, якою буде структура абстракції «вчитель»?



Структура «Друг» : ім’я, стать, вік, фото, захоплення

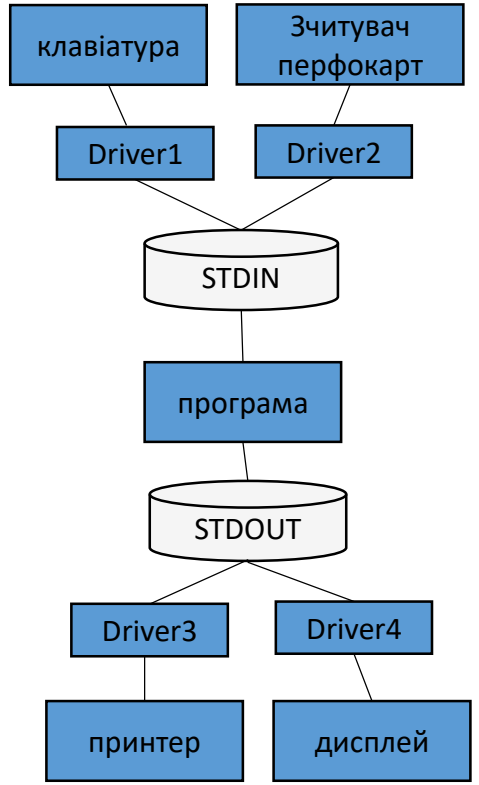


Структура «Вчитель»: ПІБ, дисципліна, досвід викладання

Застосування абстрагування в ОС. Периферійні пристрої

Абстрагування від апаратних компонент:

- драйвер (*Driver*) - програма, за допомогою якої ОС отримує доступ до пристрою;
- ОС керує «віртуальним пристроєм», який розуміє стандартний набір команд;
- драйвер переводить команди ОС в команди, які розуміє реальний пристрій.



Абстрагування роботи пристроїв *INPUT/OUTPUT*:

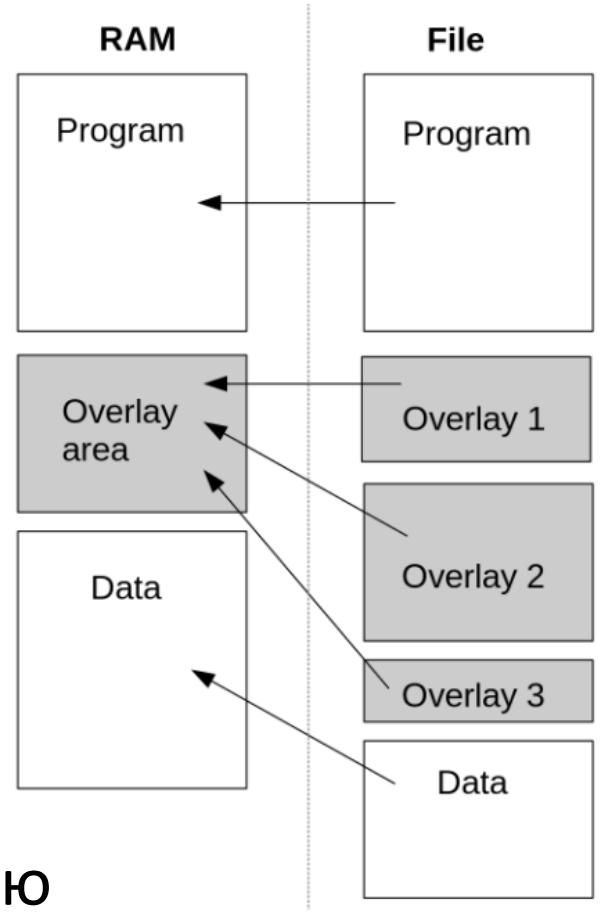
- потоки даних *INPUT/OUTPUT* – файли;
- *STDIN (Standard IN)* – файл стандартного *INPUT*-потоку
- *STDOUT (Standard OUT)* – файл стандартного *OUTPUT*-потоку
- *STDERR (Standard Error)* – файл стандартного потоку помилок

Застосування абстрагування в ОС. Керування оперативною пам`яттю (ОП)

Оперативної пам`яті завжди не вистачало для запуску великих програм

Overlay (оверлей) - метод керування роботою програми, розмір якої перевищує розмір ОП:

- програма на етапі компіляції розбивається на фрагменти - об'єктні коди меншого розміру:
  - кореневий фрагмент зі спеціальними командами;
  - підключаємі фрагменти.
- менеджер оверлеїв - програма ОС, яка за спеціальними командам завантажує з пристрою зберігання об'єктні коди-фрагменти в ОП.



Overlay-метод та модульне програмування

Ідея *Overlay*-методу може бути розвинена з урахуванням правильного процесу програмування

Модульне програмування - це організація програми у вигляді множини невеликих незалежних блоків, модулів, що спрощують тестування програми.

Об'єктний код модуля або логічної групи модулів розташовується в окремому файлі, розмір якого найчастіше менше фрагменту Overlay-методу

З'являється бібліотека готових об'єктних кодів

У програмному коді основної програми описуються посилання на бібліотеки

Динамічне зв'язування - створення в об'єктному коді посилань на об'єктні коди з бібліотек, які ОС буде динамічно завантажувати в ОП

Застосування абстрагування в ОС. Віртуальна пам'ять

Недолік Overlay-методу - програміст повинен створювати програму з урахуванням обмеження пам`яті і вручну готувати програмний код для майбутньої компіляції в об'єктні коди.

Фрагментацією програми повинна займатися ОС!

Віртуалізація пам'яті - метод управління пам`яттю для виконання програм більшого розміру ніж пам`ять шляхом переміщення частин програми між пам`яттю і пристроєм зберігання, яке виконує ОС:

- для програми пам`ять – це множина сегментів;
- підкачка (англ. swapping) – процес завантаження вмісту сегмента зі сторінок, розташованих у *swap*-файлі на пристрої постійного зберігання;
- неактивні сегменти повертаються у swap-файл

Перша реалізація накопичених ідей

MultICS (Multiplexed Information and Computing Service) – мультиплексна (складна) інформаційна та обчислювальна служба

Початок – 1964 рік за участю MIT, General Electric та Bell Labs для комп'ютерів General Electric 600-ї серії

Написана мовою високого рівня PL/1

Нововведення:

- ієрархічна файлова система;
- апаратні пристрої у вигляді файлів;
- віртуалізація пам'яті;
- динамічне зв'язування;
- *on-line reconfiguration* – зміна апаратури у процесі роботи ОС

Комп'ютерна гра і народження ОС UNIX

Більшість моделей комп’ютерів і ОС раніше досягали серйозних цілей прискорення обчислень

1969 рік, *Bell Labs* покидає проєкт ОС *MultICS*

Kenneth Thompson, тимчасово не зайнятий, створив комп’ютерну гру «Space Travel»:

- взяв з ОС *MultICS* програмний код на мові *PL/*1
- переписав на мові *Fortran* для роботи в ОС *GECOS* (*General Electric Comprehensive Operating Supervisor*) на комп’ютері *GE* 635

Denis Ritchie, його колега, полюбив гру

Але вартість 1 години роботи на *GE* 635 = 50-75 $

Thompson взяв мінікомп’ютер *PDP-7*
з низькою швидкістю, але з графічним дисплеєм, переписав гру:

- бібліотека для роботи з дійсними числами;
- графічна підсистему і *Debug*-підсистему.

ОС *UNIX* для серйозних задач

- Компанії *Bell Labs* були потрібні зручні та дешеві засоби підготовки документації … *Denis Ritchie*?!
- До своїх бібліотек *Denis Ritchie* додає файлову систему ОС *MultICS*
- В результаті отримана нова ОС, яка містить:
  - ієрархічну файлову підсистему;
  - підсистему керування процесами та пам’яттю для роботи з системою 2-х користувачів в режимі розподілу часу;
  - командний інтерпретатор *Thompson shell*;
  - декілька утиліт.
- *Brian Kernighan*, співробітник *Bell Labs*, як жарт назвав цю систему *UNICS* (*UNiplexed Information and Computing Service*) – нескладна інформаційна та обчислювальна служба (як антонім *Multiplexed*)
- Пізніше слово UNICS стало вимовлятися як *UNIX*

Ще раз про трансляцію програм. Компілятор/інтерпретатор

Компілятор – транслятор, який перетворює увесь вихідний код з мови програмування на машинну мову

Недоліки компіляції:

- великий проміжок часу на трансляцію під час *debug*-процесу
- відсутність переносності об'єктного коду на інші комп'ютери та ОС;
- відсутність жорсткого зв'язку між вихідним кодом програми та її об'єктним кодом, та подальше можливе неузгодження їх версій;
- можливі помилки під час виділення/звільнення оперативної пам'яті

Інтерпретатор – транслятор, який перетворює вихідний код з мови програмування на машинну мову порядково:

1) читання рядка вихідного коду програми;
2) синтаксичний та семантичний аналіз рядка;
3) трансляція рядка в об'єктний код;
4) виконання машинних команд об'єктного коду

Абстрагування процесору. Інтерпретація компілюючого типу

Недоліки роботи простої інтерпретації:

1) повільне виконання програми через покроковість трансляції;

2) неможливість виявити помилки у коді програми до її запуску;

3) відсутність оптимізації алгоритмів

4) неможливість програми виконуватись окремо без інтерпретатора;

Інтерпретатор компілюючого типу містить кроки:

1) компіляція вихідного коду програми у проміжне представлення коду:

- синтаксичний та семантичний аналіз коду;
- оптимізація алгоритмів;

2) інтерпретація проміжного коду на віртуальній машині (програмній моделі абстрактного процесору)

Переваги – відсутність недоліків 1-3 простої інтерпретації

BCPL (Basic Combined Programming Language) – перший інтерпретатор компілюючого типу, 1967 рік, *University of Cambridge*, *IBM* 7094, ОС *CTSS*

Абстрагування процесору та динамічна компіляція

Portable code (*P-code*) *machine* - концепція апаратно-незалежного виконання програмного коду на віртуальній машині (абстрактному процесорі)

Остання версія мінікомп'ютера PDP-11 (Programmed Data Processor-11) компанії DEC (Digital Equipment Corporation) перед Ерою персональних комп'ютерів

UC SD Pascal (*University of California San Diego*) - перша віртуальна машина для мови *Pascal* (*Pascal machine – P-machine*), 1977 рік, *Niklaus Wirth*

20 років поспіль … Компанія *Sun Microsystem …*

Java Virtual Machine та *Byte-code* – проміжне представлення програмного коду з довжиною кожного коду операції в один байт.

Динамічна компіляція або *JIT*-компіляція (*Just-in-Time compilation* - компіляція «на льоту») - одноразова компіляція проміжного коду в об'єктний код під час першого запуску програми.

Редакції (версії) ОС UNIX

1971 рік, 1-й випуск документації з ОС *UNIX* – «Перша редакція», використання мови B

Мова B – нащадок мови *BCPL* у вигляді простого інтерпретатору як нетипізований варіант майбутньої мови C

1972 рік, 2-га редакція ОС *UNIX* для мінікомп'ютера *PDP-11*, переписана на мові B;

1973 рік, 3-тя редакція зі вбудованим компілятором мови C як типізований варіант мови B;

1973 рік, 4-та редакція, частково переписана на мові C;

1974 рік, 5-та редакція:

- ОС повністю переписана на мові C;
- безкоштовне отримання вихідних програмних кодів ОС для навчальних цілей в університетах

1975 рік, 6-та редакція – комерційна версія

Остання редакція ОС *UNIX*

1976 рік, випущена 7-ма редакція:

- розробка силами *Steve Johnson* переносимого компілятора мови програмування С;
- перенос (портування) ОС на інший комп’ютер Interdata 8/32
- повноцінна оболонка інтерфейсу командного рядку *Bourne Shell* (попередник *Bash* – *Bourne again shell*):
  - назва виконуємого файлу – sh;
  - використання shell script для обробки даних при перенаправленні потоків;
  - керування порядком виконання команд та змінних;
  - керування вводом/виводом, файловими дескрипторами;
  - зняття обмежень на довжину рядків при інтерпретації скриптів.

UNIX BSD

- 1978 рік, Університет Каліфорнії в Берклі
- *BSD* (*Berkeley Software Distribution*) - система розповсюдження програм у вихідних кодах для обміну досвітом між університетами.
- ліцензія *BSD*: увесь вихідний код – власність *BSD*, всі правки – власність авторів правок.
- *Unix BSD*:
  - розвиток 6-ї (комерційної) редакції *UNIX*;
  - мова програмування *Pascal* для *UNIX*;
  - текстовий редактор *EX* – попередник *Vi*
- 1979 рік, *BSD* версії 2:
  - текстовий редактор *Vi*
  - командний інтерпретатор *C Shell*
- 1983 рік, *BSD* версія 4.2: модифікована версія реалізації мережевого протоколу *TCP/IP*

Перший персональний комп’ютер

1974 рік, мікрокомп'ютер *Altair 8800*:

- процесорний блок з *Intel* 8080, 8-бітний, 2 МГц;
- оперативна пам’ять = 256 байт;
- телетайп для:
  - считування даних з перфострічки;
  - вводу команд/даних через клавіатуру;
  - виводу результатів на принтер

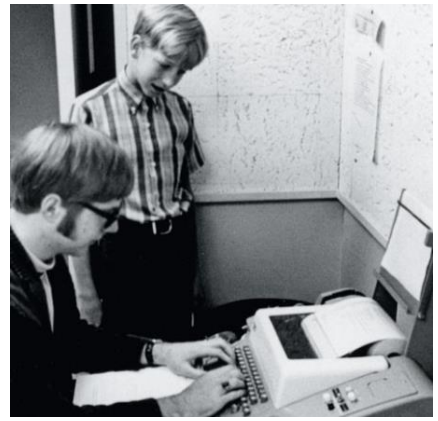
Altair 8800

Teletype Model 33

Немає системного монітору. Але…

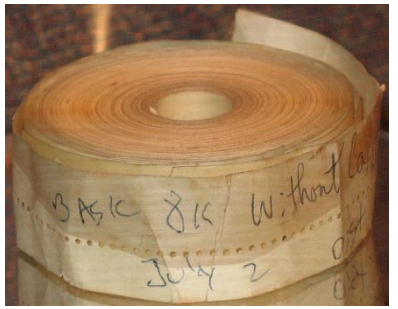
Вже популярний серед студентів *Dartmouth BASIC*

Altair BASIC – інтерпретатор мови програмування *BASIC* – перша програма компанії *Micro-Soft*, яка містила:



Біл Гейтс та Полон Ален вже 6 років програмують

- система *INPUT/OUTPUT* з телетайпом;
- редактор командного рядку.



Перфострічка з програмним кодом *Altair Basic*

Для памяті 256 байт компіляція майже неможлива

Інтерпретатор *Altair BASIC* об’ємом 4Кб записано на перфострічку

Зберігання на гнучких магнітних дисках

1971 год, IBM, діаметр – 8 дюймів

1976 – 5,25 дюйма, 1981 год – 3,5 дюйма

Доріжка диску множина секторів по 512 байт

Кластер – логічний поділ диску, який містить 1, 2, 4, 8, 16 та більше секторів (ступінь двійки)

FAT(*File Allocation Table* - таблиця розміщення файлів)

Файл може розміщатися у декількох кластерах

Комірка таблиці містить номер кластера файлу, який є наступним у ланцюжку

FAT8 може містити 2^8 = 256 комірок таблиці

FAT8, max розмір диску (файлу) = 256 х 8Кб = 2 Мб, якщо кластер = 16 секторів (розмір 8Кб)

FAT12, max розмір диску (файлу)= 2^12 х 8Кб = 32 Мб

Проблема FAT – це неекономне використання диску при зберіганні великої кількості маленьких файлів, коли розмір файлів значно менше за розмір кластеру

Обгрунтований вибір розміру кластеру

Припустимо, що використовується файлова система *FAT8*, в якій зберігається множина файлів з розмірами у кілобайтах: 1, 2, 3, 2, 1, 4, 2 Кб.

Легко підрахувати, що середній розмір файлів = 2.1 Кб

Для забезпечення максимальної економії місця на диску необхідно використати кластер мінімального розміру = 2 сектори (1 Кб). В середньому, один файл буде займати 3 комірки *FAT*-таблиці, бронюючи 3Кб на диску. На диску, в середньому, можна буде розмістити = 256/3 = 85.3 = 85 файлів із середнім значенням коефіцієнту ефективності використання місця на диску = 2.1/3 = 70%

Для підвищення економії місця у *FAT*-таблиці можна використати кластер середнього розміру = 8 секторів (4 Кб). При цьому, в середньому, один файл буде займати 1 комірку *FAT*-таблиці, бронюючи 4Кб на диску. На диску, в середньому, можна буде розмістити = 256/1 = 256 файлів із середнім значенням коефіцієнту ефективності використання місця на диску = 2.1/4 = 52%

Для забезпечення максимальної економії місця у FAT-таблиці необхідно використати кластер максимального розміру, наприклад, 16 секторів (8 Кб). При цьому, в середньому, один файл також буде займати 1 комірку FAT-таблиці, але вже бронюючи 8Кб на диску. На диску, в середньому, можна буде також розмістити = 256/1 = 256 файлів, але вже із середнім значенням коефіцієнту ефективності використання місця на диску = 2.1/8 = 26%

Перша дискова ОС – *Disk OS (DOS)*

1975 рік, популярність мікрокомп’ютерів з 8-бітними процесорами *Intel* 8080 та *Zilog* Z80

CP/M (*Control Programs for Monitor, пізніше – Control Programs for Microcomputers*) – перша «універсальна» ОС для мікрокомп'ютерів з дисковими магнітними накопичувачами (дискетами).

Створено на мові високого рівня *PL/M* (*Programming Language for Microcomputers*) без відомих *INPUT/OUTPUT*-функцій, але з прямим доступом до пам’яті та процесору, як на мові *Assembler*;

ОС складається з 3-х частин:

- базова система вводу-виводу (*BIOS - Basic Input/Output System*) - апаратно-незалежний набір програм:
  - зберігається у постійній пам’яті або на диску;
  - виконує початкове завантаження основної ОС;
- базова дискова ОС - апаратно-незалежний набір програм, зберігається на диску;
- оболонка командного рядку

Примітивна *Apple OS*

- 1976 рік, «Клуб саморобних комп’ютерів», Каліфорнія ...
- Два Стіва: Стів Джобс та Стів Возняк
- Персональні комп’ютери збираються для власних цілей тими, хто зацікавлений електронікою
- Ідея створення персонального комп’ютера для всіх
- Архітектура мікрокомп’ютера *Apple I*:
  - 8-бітний процесор *MOS* 6502 (1 МГц)
  - вбудована клавіатура;
  - підключення *INPUT/OUTPUT*-пристроїв:
    - телевізор, касетний магнітофон

- *System Monitor* – спрощена ОС *Apple OS*:
  - Керування *INPUT/OUTPUT*-пристроями
  - спрощений інтерпретатор командного рядку:
    - перегляд вмісту пам’яті

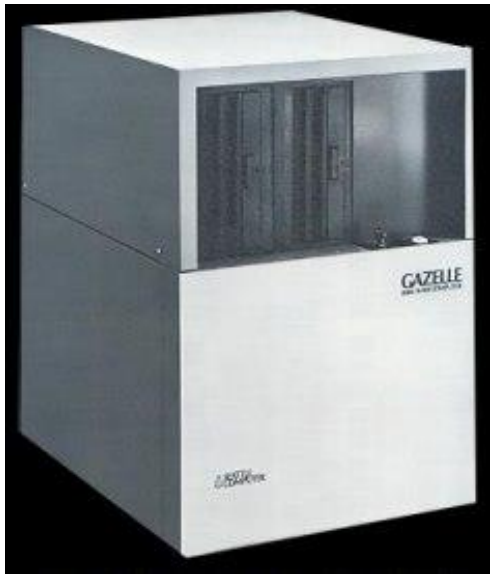
Apple DOS

1978 рік, мікрокомп’ютер *Apple II*, з підтримкою кольорової графіки

Особливості *Apple DOS*:

- інтерпретатор мови *Basic* зберігається у постійній пам’яті
- підтримка декількох *floppy*-дисків 5 дюймів
- назви файлів довжиною до 30 будь-яких символів
- жорстка типізація файлів:
  - *I* (*Integer BASIC*) – спрощений набір команд, немає графічного режиму, але швидкість виконання програми;
  - тип *A* (*Applesoft BASIC*) – розширений набір команд, графічний режим;
  - тип *B* (двійковий, об’єктний файл);
  - *T* (текстовий).
- *Apple Pascal* - інтерпретатор компілюючого типу

86-DOS. Пролог перед появою *MS-DOS…*



1979 рік, комп’ютер *Seattle Computer Products* процесором *Intel 8086*

QDOS (*Quick and Dirty Operating System* - швидка та «брудна» операційна система), в подальшому *86-DOS:*

- копія ОС *CP/M*;
- покращена логіка буферизації дискових секторів
- файлова система - *FAT12*
  - суміжні сектори диска об'єднуються у кластери
  - комірки таблиці - адреси кластерів на диску

Чому «брудна» ОС?

ОС надавало лише те, що було потрібно лише одному користувачу повільного персонального комп’ютера

ОС була суттєво обмеженою у порівнянні з ОС великих комп’ютерів

Звичайні виробничі проблеми. *IBM PC DOS, MS DOS*

Кінець 70-х років. Споживачі зацікавлені у персональних комп'ютерах

IBM створює серію *IBM 5100 Portable Computer з процесором PALM*

Але вартість – від 9 тис. $ – далека від від звичайного споживача

1980 рік, *IBM Personal Computer (IBM PC)* або модель *IBM 5150:*

- 16-бітний процесор Intel 8088, 4.77 MHz, RAM 16KB/64KB

Вже є комп'ютер, але немає ОС – це класика для компанії *IBM*

IBM звертається до компанії *Seattle Computer Products*

Компанія *Seattle Computer Products* відмовляє *IBM*

Компанія *Microsoft* викуповує у компанії ліцензію на ОС *86-DOS*

Microsoft адаптує ОС *86-DOS* під особливості архітектури *IBM PC*

Співробітники *IBM* аналізують коди ОС та виявляють 300 помилок

Microsoft виправляє помилки, але *IBM* не чекає...

IBM PC DOS (*IBM Personal Computer Disk Operating System*) – власні виправлення знайдених помилок

Microsoft одночасно випускає свою *MS DOS*

Найпопулярніший домашній комп'ютер

- 1982 рік, *Z80 ZX Spectrum,* 8-бітний домашній комп’ютер, компанія «*Sinclair Research Ltd*»
- Архітектура мікрокомп’ютера:
  - процесор *Zilog* (2,5-8 МГц)
  - оперативна пам’ять 16-48 Кб;
  - вбудована клавіатура;
  - підключення пристроїв телевізор, касетний магнітофон
- ОС *CP/M*
- Вбудований інтерпретатор мови *Sinclair BASIC*
- Кінець 80 років …, терени колишнього СРСР …, десятки «клонів» архітектури домашніх комп’ютерів за архітектурою *Z80 ZX Spectrum*

ОС для виробництва - ОС реального часу

При звичайних обчисленнях для ОС немає жорстких вимог до часу завершення операцій

Але на конвеєрних лініях виробництва є такі вимоги

ОС реального часу (*Real Time OS*) - це здатність ОС забезпечити необхідний рівень сервісу певний проміжок часу

Ідеальна ОС RT повинна мати передбачувану поведінку за всіх сценаріїв навантаження

Система жорсткого *RT* гарантує виконання операції точно в термін (або в певний період часу)

Система м'якого *RT* допускає деяке недотримання терміну будь-якої дії без негативних наслідків

RT-11 - перша *one-user OS RT*, 1970 рік, для *PDP-11*

Одеська Політехніка
Інститут комп`ютерних систем
Кафедра інформаційних систем
Дисципліна «Операційні системи»

Дякую за увагу!
Запитання?

Олександр А. Блажко,
доцент кафедри інформаційних систем,
E-mail: blazhko@ieee.org
Telegram-канал: t.me/Operating_Systems_IS

Одеса, 20 березня 2023 року