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

Тема 1: Історія операційних систем (ОС)
Лекція 2: ОС для великих електронних комп`ютерів

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

Одеса, 13 березня 2023 року

Ще раз про поняття «Операційна Система»?

Операція (лат. Operatio, дія) - сукупність дій, яка:

- потрібна для досягнення якоїсь мети;
- залежить від людини;
- включає деякий набір технічних засобів.

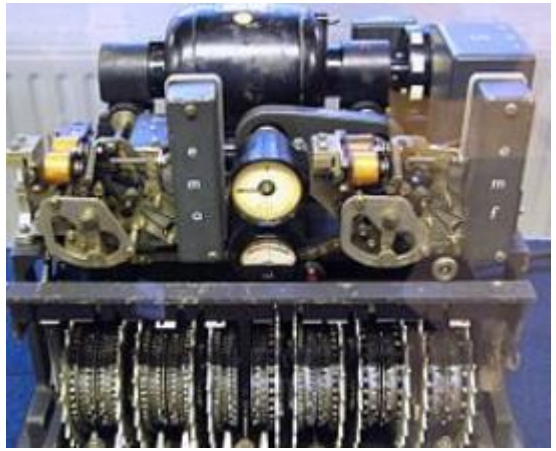
Операційна система (ОС) - це система (над-система, супер-система), метою якої є допомога іншій системі досягти свою мету.

Механічні обчислювальні пристрої 17-19 століття та першої половини 20-го сторіччя не мали механічної ОС.

ОС таких пристроїв - набір текстових інструкцій з описом алгоритмів з послідовністю кроків їх використання людиною

Завершення механічного світу обчислень. Військово-інформаційні потреби: криптоаналіз військових шифрів

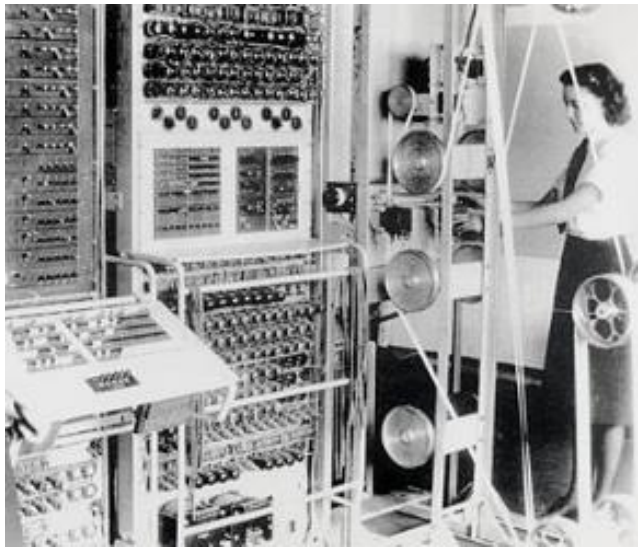
ІІ світова війна, обмін зашифрованих радіоповідомлень між «вовчими зграями» німецьких підводних човнів



Повідомлення старої системи *Enigma* успішно розшифровуються вручну британськими вченими

Повідомлення нової системи *Lorenz SZ* розшифрувати було складно: алгоритм Вернама C = M⊕ K

Невелика помилка німецького шифрувальника, який зашифрував два повідомлення одним ключем …



1943 рік, Colossus («Колос») – спеціалізований обчислювач для розшифровки повідомлень:

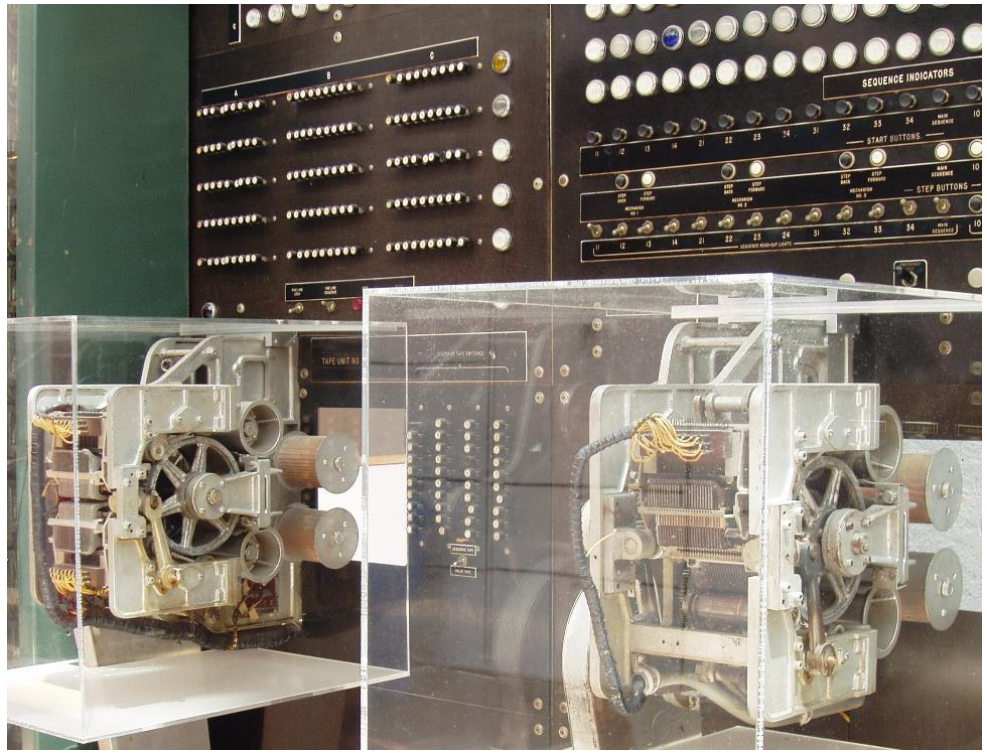
- *INPUT*-засіб для команд - механічні реле;
- *INPUT*-засіб для даних - перфострічка;
- *OUTPUT*-засіб даних результату - перфострічка

Приклад складності обчислень для війни: Розрахунок математичних таблиць

ІІ світова війна, США, складність розрахунку математичних таблиць для цілей ВМФ з ручною працею жінок-«обчислювачів» на арифмометрах

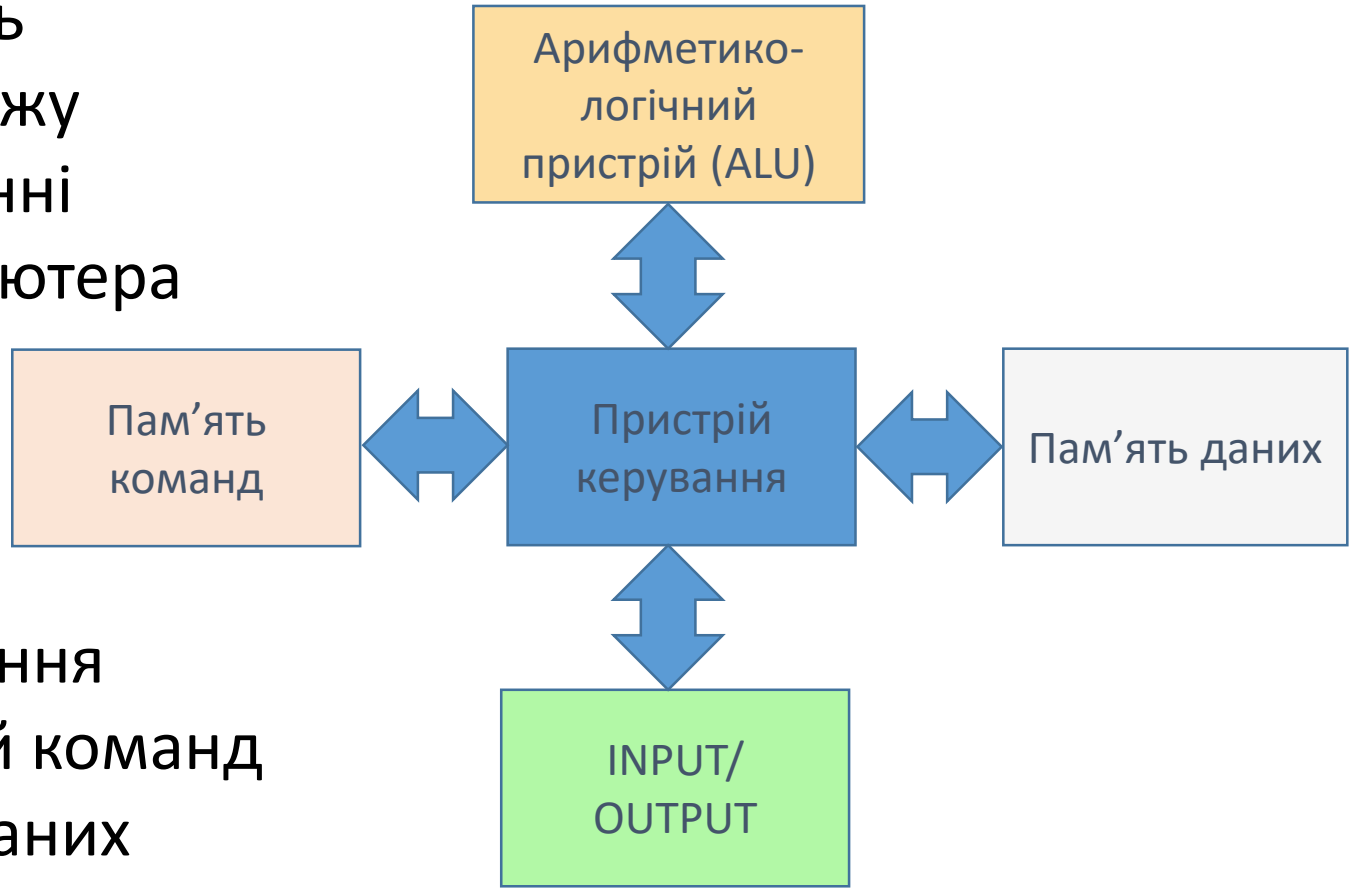
7.08.1944, компанія IBM, Гарвардський університет, «*Mark* I» *(Automatic Sequence Controlled Calculator)* – перший електромеханічний комп`ютер:

- гарвардська архітектура – принцип розподілу команд операцій і даних
- *INPUT*-потік операцій/даних – дві перфострічки;
- *OUTPUT*-потік даних результату – перфострічка;
- реалізація циклів у програмному коді – замикання початку і кінця стрічки



Гарвардська архітектура комп’ютера

Ідея належить
Чарльзу Беббіджу
при проектуванні
механічного комп’ютера



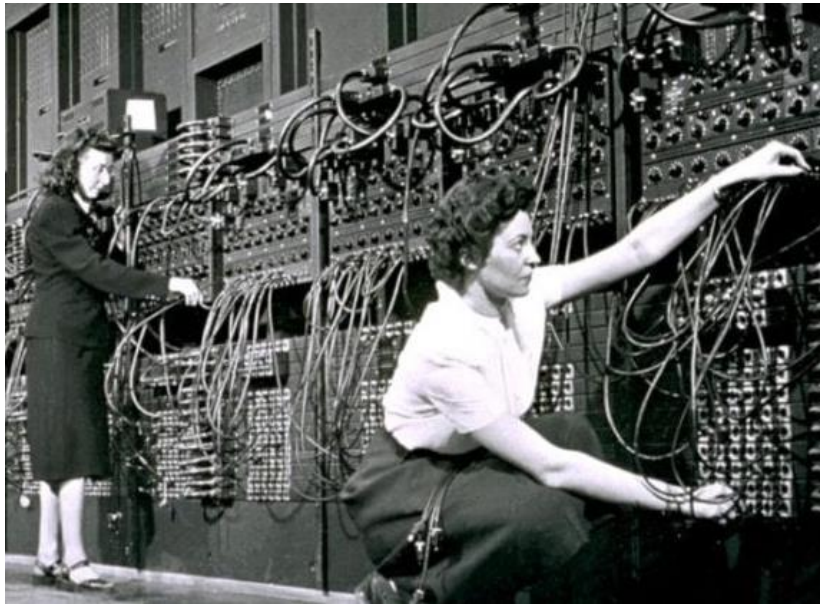
Принцип розділення
INPUT-потоку операцій команд
та *INPUT*-потоку даних

Перевага архітектури – *ALU* (процесор) одночасно може зчитувати:

- чергову команду операції;
- дані, необхідні для операції

Приклад складності обчислень для війни: Розрахунок таблиць стрільби снарядів

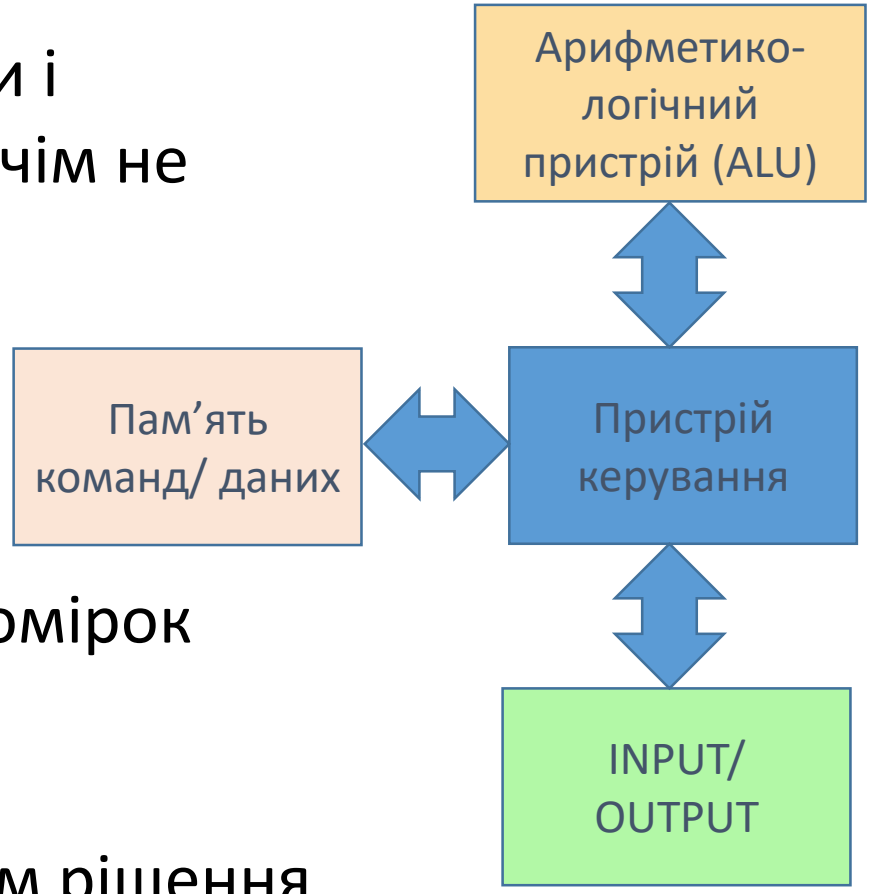
- ІІ світова війна, США, розрахунок таблиць стрільби снарядів при різних комбінаціях параметрів (піднесення ствола, номер/температура заряду, швидкість снаряда, швидкість вітру, температура і тиск повітря)
- Висока трудомісткість ручної роботи жінок- «обчислювачів» з використанням арифмометрів
- 14-15.02.1946, *ENIAC (Electronic Numerical Integrator And Computer)* – електронний цифровий суматор і обчислювач, перший електронний комп`ютер:
- INPUT-засіб для команд - механічні реле;
- INPUT-засіб для даних - перфострічка;
- перше завдання - математичне моделювання термоядерного вибуху супер-бомби



Архітектура комп'ютера «фон Неймана». Загальні принципи

1) принцип однорідності пам'яті - команди і дані розміщено в одній пам'яті і зовні нічім не відрізняються на відміну від Гарвардської архітектури

2) принцип двійкового кодування – дані і команди кодуються цифрами 0, 1



3) принцип адресності – пам’ять-масив комірок з унікальними адресами

4) принцип програмного управління :

- всі обчислення передбачені алгоритмом рішення завдання
- алгоритм - це програма як послідовність керуючих слів-команд виконання операцій обчислень.

Автоматизація роботи програміста

Комп'ютер розуміє лише машинні команди процесора

Але програмісту як людині, а не машині, складно:

- запам'ятовувати коди машинних команд;
- розраховувати та запам'ятовувати адреси переходів між командами у пам'яті комп'ютера при створенні або зміні алгоритму

Рішення проблеми - заміна кодів машинних команд на команди, які наближені до слів природної для людини мови.

Трансляція програми - перетворення програми, представленої на одній мові програмування, в програму в машинних командах

Програма-Транслятор додатково допомагає знайти помилки (семантичні і синтаксичні).

Автоматизація роботи програміста. Види трансляції

Основні види трансляції:

- компіляція;
- інтерпретація;
- інтерпретація компілюючого типу;
- динамічна компіляція.

Компілятор - транслятор, який перетворює увесь вихідний код з будь-якої мови програмування на машинну мову

Асемблер (від англ. *Assembler* – збиральник) – один з перших компіляторів, що надає:

- алфавітні коди операцій, імена регістрів;
- розмітку рядків програми для звернення до них по іменах, а не за адресами, з інших частин програми

Приклад програми на мові асемблера

Програма виведення на екран рядка «Hello, world!» :

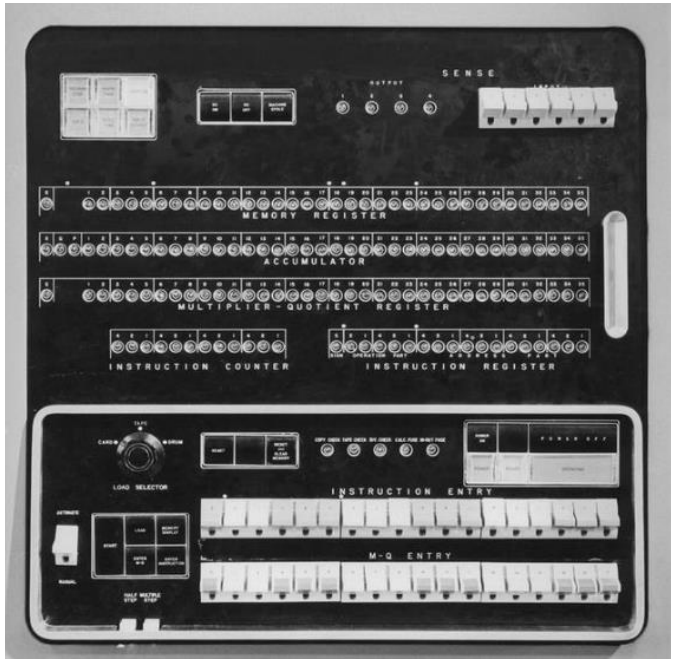
- функція ОС *MS DOS (9)* - вивід рядка;
- адреса рядка розміщена в регістрах DS: DX, виклик переривання 21h

Машинні команди

Адреса	Код операції	Мітка	Команда асемблера	Коментар
0100	1E		push DS	зберігти регістр даних у стек для подальшого використання
0101	B4 09		mov AH,09	встановити значення функції для друкування рядка на екран
0103	BA 10 01		mov DX,offset @H	встановити посилання на рядок
0106	0E		push CS	зберегти регістр коду у стек
0107	1F		pop DS	завантажити зі стеку значення регістру коду до регістру даних
0108	CD 21		int 21H	виконати переривання 21H MS DOS для друкування рядка
010A	1F		pop DS	отримати знання регыстру даних зі стеку
010B	B8 00 4C		mov AX,4C00H	встановити значення функції для завершення роботи програми
010E	CD 21		int 21H	виконати переривання 21H MS DOS для завершення роботи програми
0110	48	@H:	db "Hello,world!$"	Рядок повідомлення, $ - символ кінця рядку

Історія комп'ютера = Історія ОС

- 1952 рік, випущено комп'ютер *IBM 701* - перший комерційний комп'ютер для наукових досліджень
- *KOMPILER* - перший компілятор з мови програмування низького рівня
- Компілятор *PACT* - мінімізація використовуваної оперативної пам'яті при роботі програм



Проблема простоювання комп'ютера

З появою мов програмування організація обчислювальних робіт на комп'ютері стала включати наступні кроки:

1) Програміст пише програму на папері на мові програмування низького рівня
2) Програміст переносить код програми на перфокарти
3) Програміст приносить колоду перфокарт в кімнату введення даних і передає оператору
4) Оператор завантажує програму на комп'ютер
5) Комп'ютер передає результати роботи на принтер
6) Оператор отримує роздруковану відповідь
7) Оператор передає відповідь програмісту

Наскільки швидко може виконуватися 4-й крок, який визначається рівнем навичок людини-оператора?

Проблема простоювання комп'ютера

4-й крок «завантаження програми на комп'ютер» містить кроки:

1) завантаження колоди перфокарт в комп'ютер
2) завантаження потрібного транслятора з мови програмування;
3) запуск транслятора і отримання програми в машинних кодах;
4) зв'язування програми з бібліотечними підпрограмами;
5) завантаження програми в оперативну пам'ять;
6) запуск програми;
7) отримання результатів роботи програми на друкувальний або інший пристрій.

Лише 6-й крок – це робота комп’ютера, який коштував багато грошей

«За що ми заплати? Щоб комп’ютер когось постійно чекав?» – класичне питання керівництва

Проблема простоювання комп'ютера. Системний монітор та пакетний режим

З представлених раніше кроків тільки 6-й крок «запуск програми» безпосередньо пов'язаний з виконання програми на дорогому комп'ютері

Багато ручних дій оператора призводили до затримок у часі і простоюванні комп'ютера

З метою виключення затримки були розроблені перші спеціальні керуючі програми - монітори для автоматичного переходу між завданнями

Завдання об'єднувалися в пакети

Системний монітор - прообраз ОС, яка працювала в пакетному режимі автоматизованої обробки завдань

Історія комп'ютера = Історія ОС

Чим займається компанія *General Motors* ?

GM OS (*General Motors Operating System*)

Початок 1956 року, для потреб General Motors Research Laboratories

Перший ламповий комп'ютер *IBM 701*

Перша ОС, яка працює в пакетному режимі

Історія комп'ютера = Історія ОС

- *GM-NAA I/O (General Motors & North American Aviation Input/Output System);*
- кінець 1956 року,
- співпраця компаній *General Motors* та *American Aviation*
- комп`ютер *IBM 704*;
- *FORTRAN - «Mathematical Formula Translating System» -* перша мова програмування високого рівня

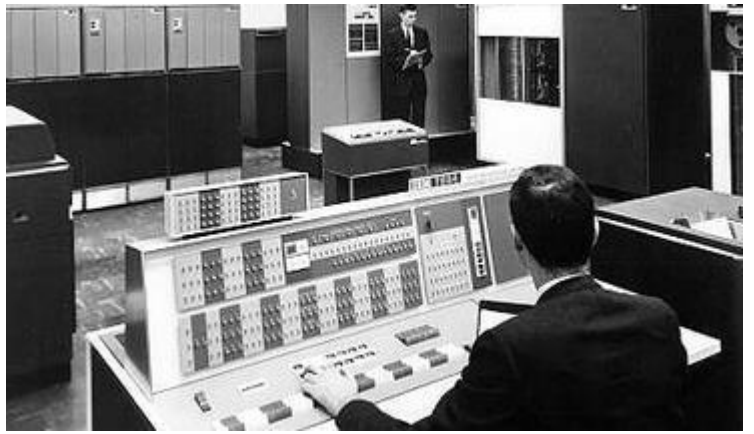
Вивід на екран рядка:
*PRINT *, “Hello, world!”*

Можливості ОС:

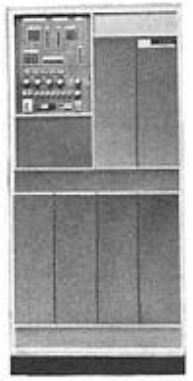
- послідовне виконання поданих на вхід програм - пакетний режим роботи;
- надання програмісту функцій для роботи з *INPUT/OUTPUT* (перфокарти, друк на принтері).

Історія комп'ютера = Історія ОС

- *BESYS (Bell Operating System),* 1957 рік, *AT&T*
- ОС для потреб обчислювального центру *Bell Labs*
- Комп'ютер *IBM 704* + комп'ютер *IBM 1401* з модулем для швидкісної обробки перфокарт, записом на магнітну стрічку і друку
- Виконання динамічно завантажуваних коротких завдань з перфокарт, які попередньо записуються на стрічку

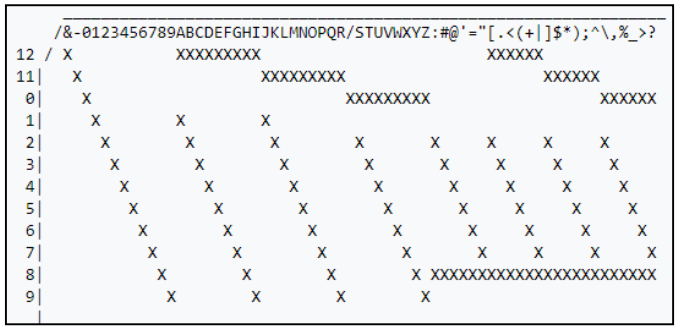


IBM 1402 Card Read-Punch



IBM 1401 Processing Unit
(1400 core-storage positions)

IBM 1403 Printer



Історія комп'ютера = Історія ОС

- *IBSYS (IB System),* 1959 рік, IBM
- Комп`ютер *IBM 7090/7094*.
- Мова програмування F*ORTRAN* і мова *COBOL (Common Business-Oriented Language)* – мова з *English-like* синтаксисом і властивостями самодокументованої мови, яка має:
  - перевагу завдяки опису програмного коду за наближеним до англійської мови синтаксисом;
  - недолік через надмірну кількість операторів, понад 300 слів.
- *FORTRAN Monitor System (FMS),* використовує магнітну стрічку для компіляції програм на мові FORTRAN.

Історія комп'ютера = Історія ОС

ОС IBSYS використовує базовий програмний моніторинг, який зчитує зображення контрольних карт між колодами програм і карт даних для завдань.

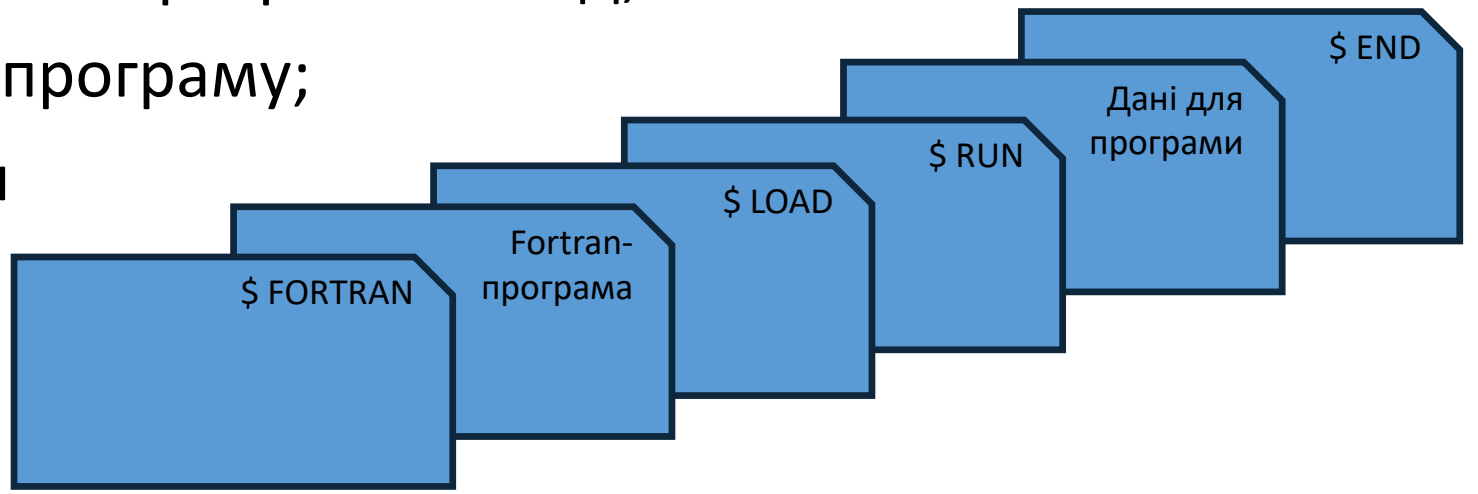
Контрольна карта починалася в 1-м стовпці з «$» і назвою команди, наприклад:

$ FORTRAN - завантажити компілятор;

$ LOAD - завантажити програмний код,

$ RUN - запустити програму;

$ END - завершити



Проблема простою комп'ютера. Компонентна багатозадачність

Швидкість пристроїв вводу/виводу даних менше швидкості процесора

Процесор часто простоює, чекаючи завершення роботи пристроїв вводу/виводу

Компонентна багатозадачність - одночасне використання для різних завдань різних компонент комп'ютера (пристроїв вводу/виводу, процесор):

- розбиття пам'яті на кілька частин - розділів, в яких виконуються окремі завдання;
- завдання №1 очікує завершення вводу/виводу;
- завдання №2 використовує процесор

Проблема простою комп'ютера. Управління буферизацією і чергами

Очікування завдання при великому обсязі даних:

- одержуваних з пристроїв вводу (перфокарти);
- переданих на пристрій виводу (принтер).

Підкачка даних або *spooling* (від англ. *SPOOL - Simultaneous Peripheral Operation On Line*) - спільна периферійна операція, яка виконується в інтерактивному режимі.

Spooler - програма управління чергою із завдань на пристрій вводу/виводу:

- при виконанні завдання з пристрою вводу данні накопичуються в спеціал. частині пам’яті - буфері;
- якщо пристрій виводу зайнято, завдання надходить в чергу і чекає звільнення пристрою;
- програма може не очікувати завершення завдань.

Невитісняюча багатозадачність

Початок 1960-х років, вже немає простою при перемиканні між двома процесором та пристроями вводу/виводу
Але програми виконуються строго послідовно
Розмір пам'яті комп'ютера став більше, дозволяючи розміщувати кілька програм
Невитісняюча багатозадачність - тип багатозадачності, при якому ОС одночасно завантажує в пам'ять декілька програм, але процесорний час надається тільки одній програмі
Контекст - інформація про поточний стан програми (значення регістрів процесора, поточна адреса виконання машинної команди)
Перемикання контексту - ОС самостійно або за запитом користувача перемикає процесор на контекст іншої програми

Дистанційна робота користувачів

Сьогодні ми господарі комп'ютерів і використовуємо їх одноосібно.

Але чи завжди ми їх використовуємо на 100% їх можливостей?

Чи можемо ми його дати в оренду іншим людям?

Чи можна віддати в оренду можливості комп'ютера, орендувати самі послуги, що надаються комп'ютером, дистанційно?

Розробники та споживачі великих комп'ютерів про це стали замислюватися ще понад 60 років тому, коли для доступу до одного дорогого комп'ютера стояла черга з тисяч споживачів.

Ідеї механічних потокових ліній та конвеєра надихали розробників

Багатозадачність на основі режиму розподілу часу

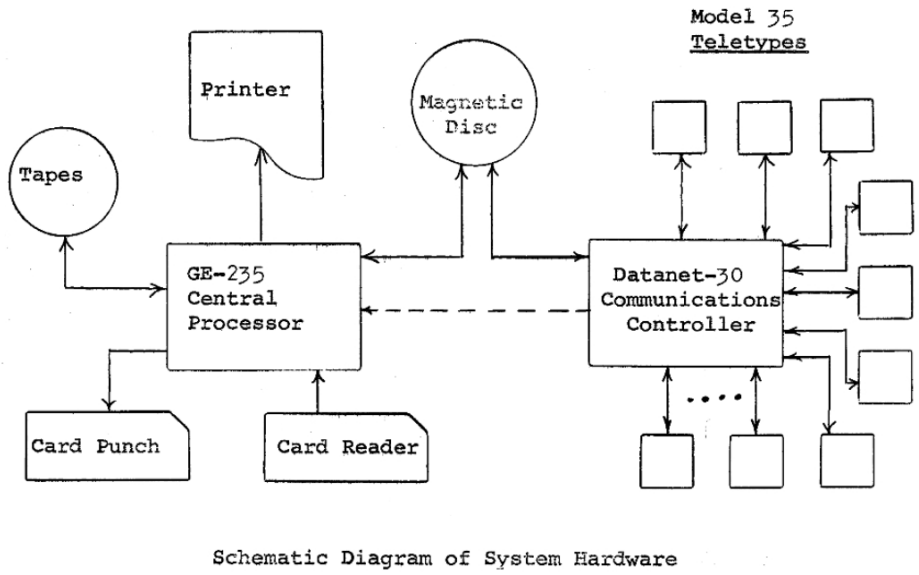
Вирішено проблему простою при перемиканні між процесором і пристроями вводу/виводу.
Але програми виконуються послідовно.
Різні типи виконуваних завдань користувачами:

- редагування програмного коду з високим часом затримки між введенням рядків;
- виконання маленької короткої програми;
- виконання великої програми.

Режим розподілу часу (*Time Sharing*) - розподіл обчислювальних ресурсів комп'ютера між багатьма користувачами.
1962 рік, комп'ютер IBM 7094
CTSS (Compatible Time Sharing System) - перша ОС з режимом розподілу часу.

Багатозадачність на основі режиму розподілу часу

Компанія General Electric, 1961 рік:
GE-235 – 20-бітний мінікомп’ютер;
DATANET-30 – комунікаційний комп’ютер для підримки до 128 телетайпів (телепринтерів), наприклад,
Teletype Model 35ASR
Да́ртмутський коледж США, 1963-1964 ...



Schematic Diagram of System Hardware

The Dartmouth Time-Sharing System (DTSS) – перша успішна великомасштабна система розподілу часу
Dartmouth BASIC (Beginners' All-purpose Symbolic Instruction Code)
в режимі компіляції програмного коду
Перше *інтегроване середовище проектування (Integrated Design Environment, IDE)* з командами: *New* – нова програма, *Save* – зберігти на диску, *Run* – скомплювати та виконати.

Завершення світу механічно-електронних INPUT/OUTPUT-засобів роботи з даними. Термінальні пристрої

Teletype як електро-механічна друкарська машина була створена ще у 19-му сторіччі.

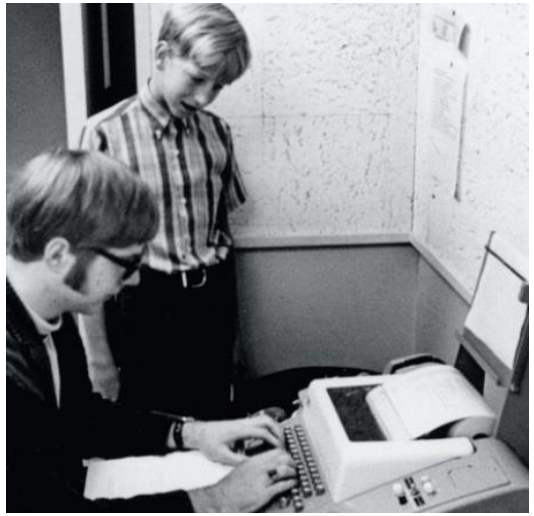
Teletype Corporation: Teletype Model 35ASR (ASR - Automatic Send and Receive), 1963 рік

В середині 20-го сторіччі телетайп (телепринтер) є основним *INPUT/OUTPUT*-засобом:

Teletype IBM 2741, 1965 рік

- *INPUT*-засіб – клавіатура, перфокарта/перфострічка;
- *OUTPUT*-засіб – друк принтером на папері;

1920-1940 – есперименти з електронно-променевою трубкою (ЕПТ) для *TeleVision*



Біл Гейтс, Полон Ален та Teletype Model 35ASR, 1968 рік

Термінальний пристрій містив:

«Скляний» термінал IBM 2260, 1965 рік

- *INPUT*-засіб – клавіатура;
- *OUTPUT*-засіб – ЕПТ;
- засіб зв'язку із комп'ютером

Початок уніфікації світу комп’ютерів. Віртуальна машина

1965 рік, випуск мейнфрейму *IBM System / 360-67* з апаратною підтримкою реалізації систем з розподілом часу (*dynamic address translation*)

Але ще немає багато-користувальницької ОС, яка працює в режимі з розподілом часу

Економне рішення проблеми від IBM:

- одно-користувальницька *ОС CMS (Cambridge Monitor System)*
- *CP/CMS (CP - Control Program)* - ОС для управління декількома копіями ОС *CMS* на одному IBM 360-67

VM/CMS (Virtual Machine / Cambridge Monitor System) – ОС для управління «віртуальними машинами» на IBM 360-67

1972 рік, випуск мейнфрейму *IBM System / 370*, який надає більше віртуалізації через гіпервизор – менеджер віртуального обладнання для запуску віртуальних ОС

80-ті роки минулого століття …
Один день із життя студента-програміста 1-го курсу

Ви вже звикли до власного персонального комп'ютера (ПК) при виконанні завдань з програмування.

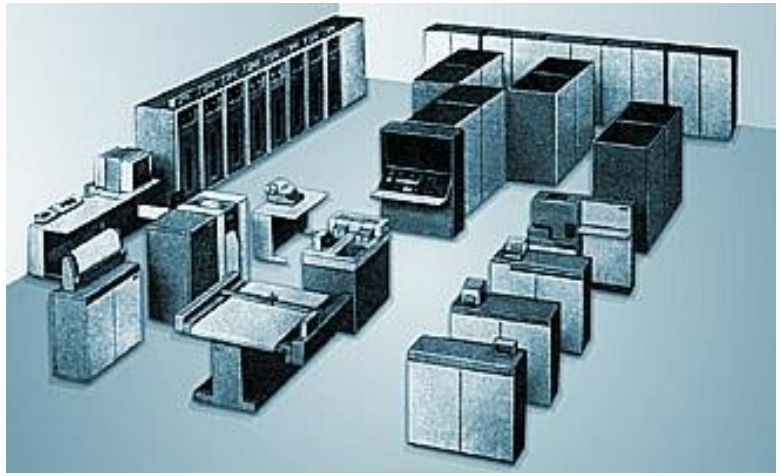
А що робили студенти-програмісти 30-50 років без ПК?

ЄС ЕОМ (Єдина Серія Електронних Обчислювальних Машин) — радянська серія комп'ютерів, сумісних з IBM System/360-370

Процес виконання лабораторної роботи, наприклад, в 1992 році:

1) на папері підготували текст
2) через термінал внесли текст
3) запустили програму
4) передали тест та результати роботи програми на принтер

Термінал роботи студента в лабораторії



Великий комп`ютер, пов’язаний з терміналами мережею

Принтер для друку тексту програм

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, 13 березня 2023 року