Лекція 04
[4.1]
Складові компоненти мікропроцесорів [2.2]
Цифрові інформаційні пристрої

Лекція 4

Судаков О.О, Радченко С.П.

«Сучасна мікропроцесорна техніка»

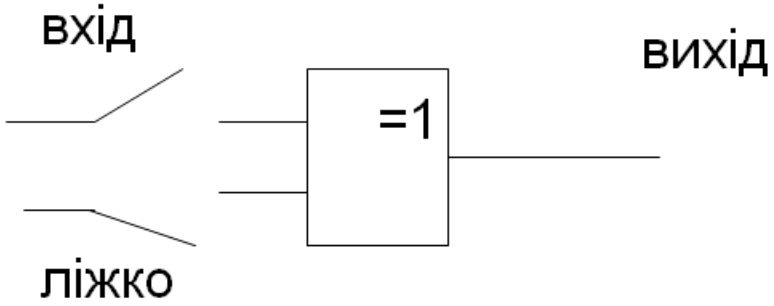
[4.2]
Комбінаційні та послідовносні схеми

- Комбінаційна –результат на виході залежить від поточного значення параметрів на входах
  - комутатори
  - логічні схеми
  - Генератори
  - Шифратори -дешифратори
- Послідовносна – результат на виході залежить від входів, часу, стану
  - тригери
  - лічильники
  - регістри
  - Пам'ять
  - процесори

[3.3]
Комбінаційна схема (вимикач світла)

- В кімнаті 2 вимикачі
  - 1-й біля входу
  - 2-й біля ліжка
- Зробити схему яка вмикає і вимикає світло в кімнаті
  - Вмикаєш при вході
  - Вимикаєш з ліжка
  - Вмикаєш з ліжка
  - Вимикаєш при вході

Вхід	Ліжко	Вихід
0	0	0
1	0	1
0	1	1
1	1	0

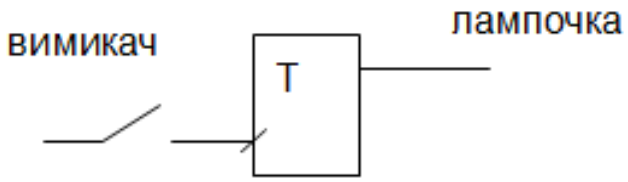


Вихід= (вхід AND NOT ліжко) OR
(NOT вхід AND ліжко)=вхід XOR ліжко

[4.4]

Послідовнісна схема

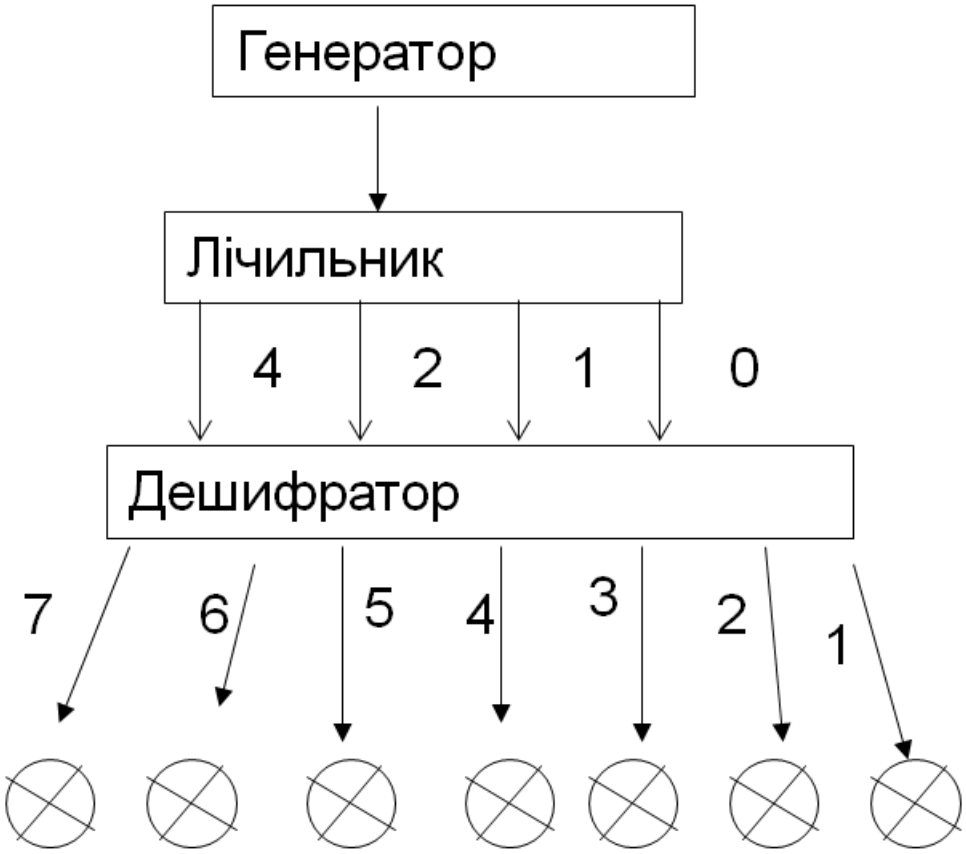
- Один вимикач і одна лампочка
  - Вимикання вимакача нічого не міняє
  - Вмикання вимикача вимикає лампочку, коли вона горить і вмикає – коли не горить



[4.5]

Біжучі вогні

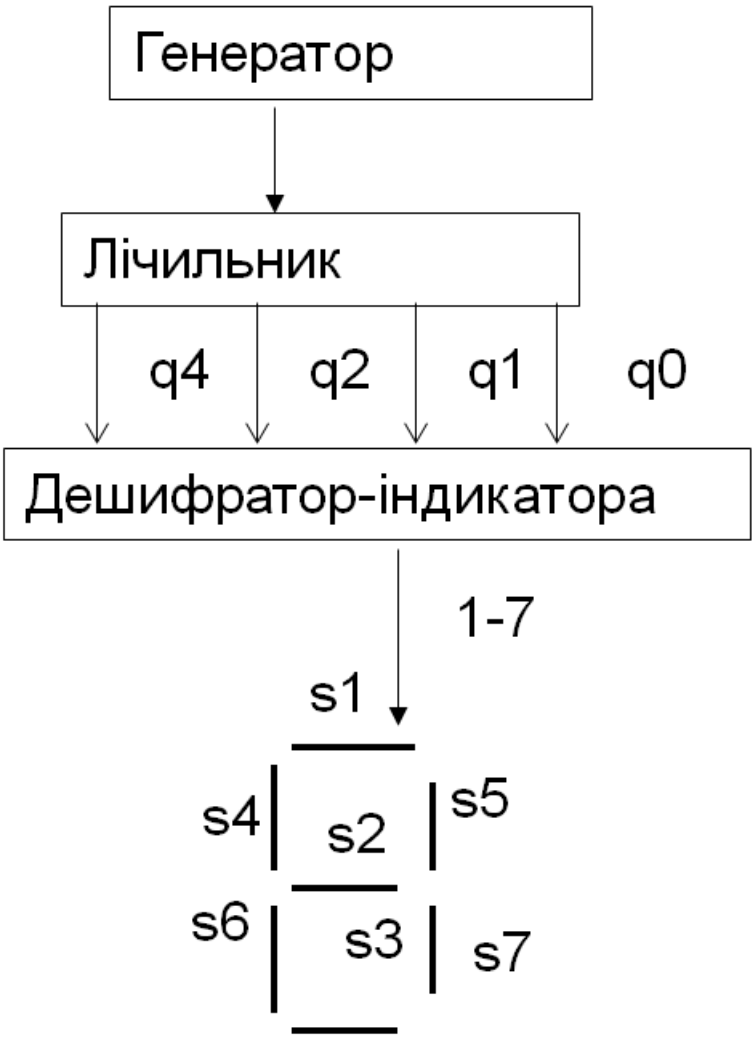
- **Система з послідовним виконанням**
- **Примітивний процесор**
  - Лічильник рахує імпульси генератора
  - Видає на виході двійковий код
  - Дешифратор запалює лампу з відповідним номером
- **Компоненти**
  - Генератор – системний годинник (SC)
  - Лічильник-програмний лічильник (PC)
  - Дешифратор – пам’ять команд
  - Лампочки- виконувальний пристрій
- **Вихід лічильника**
  - Шина адрес
  - Адреса команди



[4.6]

Дешифратор

- Дешифратор –комбінаційна схема
  - Кожному вхідному коду співставляється вихідний код
  - Аналог програми і пам’яті
- Кожній адресі можна співставити
  - Пристрій
  - Команду
  - Дані
- Постійна пам’ять

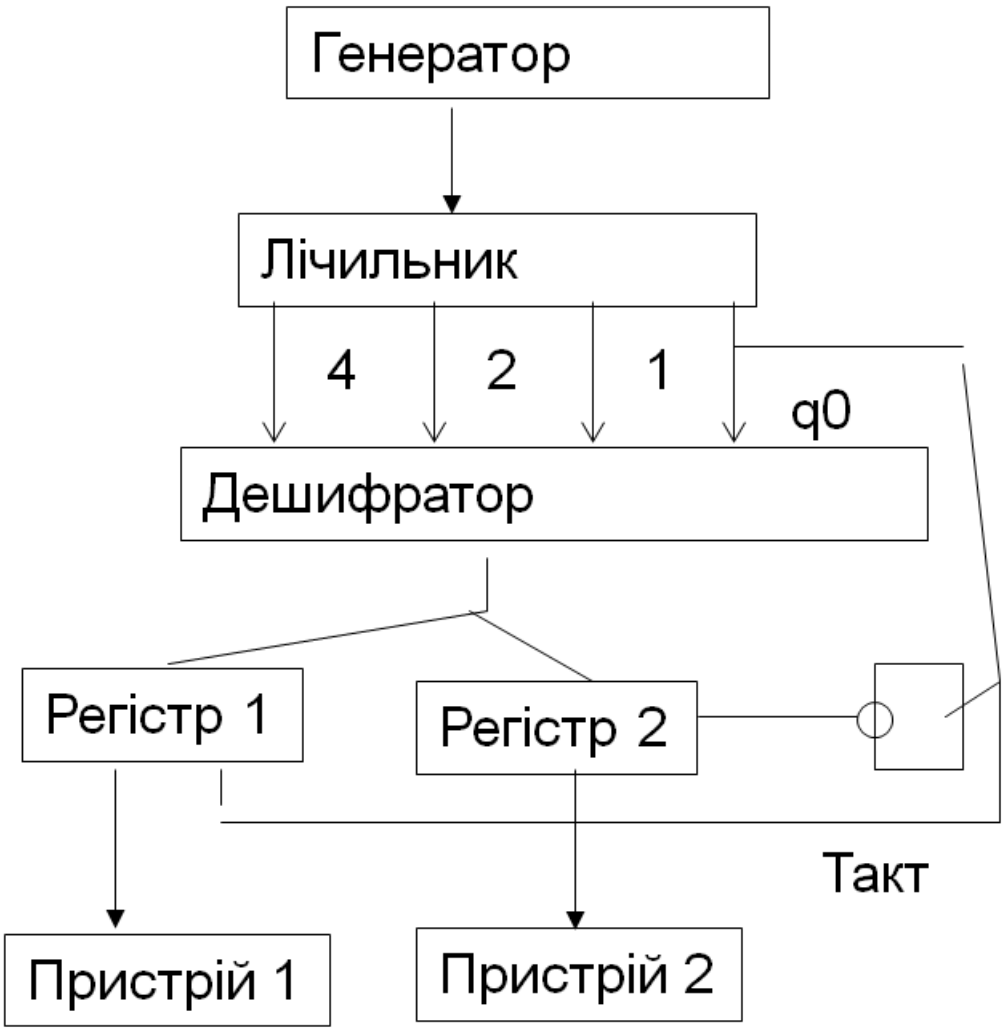


0:s1 & s4 & s6 & s3 & s7 & s5
S1= (NOT q0 & NOT q1 & NOT q2 & NOT q4) or ..
S2=

[4.7]

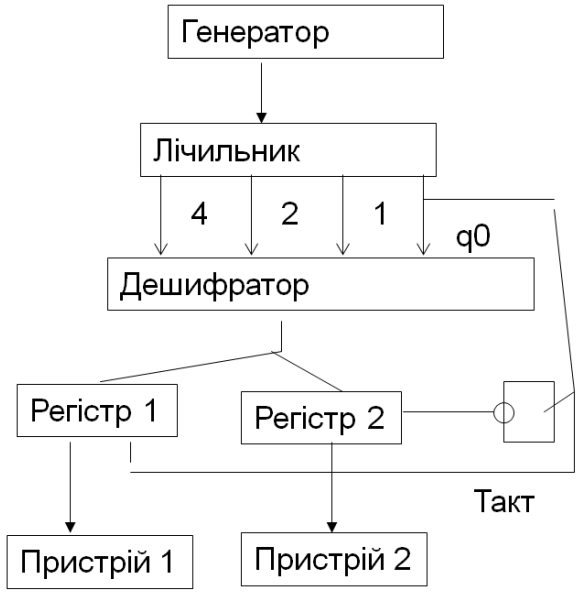
Керування декількома пристроями

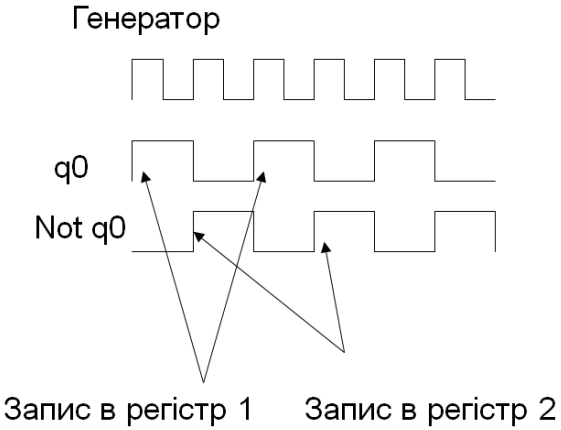
- Задача
  - Пристрій 1 – кожна парна команда
  - Пристрій 2 – кожна непарна команда
- Реалізація
  - Дешифратор видає на вихід команди
  - Регістр 1 запам’ятовує непарні команди
  - Регустр 2 запам’ятовує парны команди



[4.8]

Робота схеми





Операція з пристроєм виконується за 2 такта – вдвічі повільніше

[4.9]

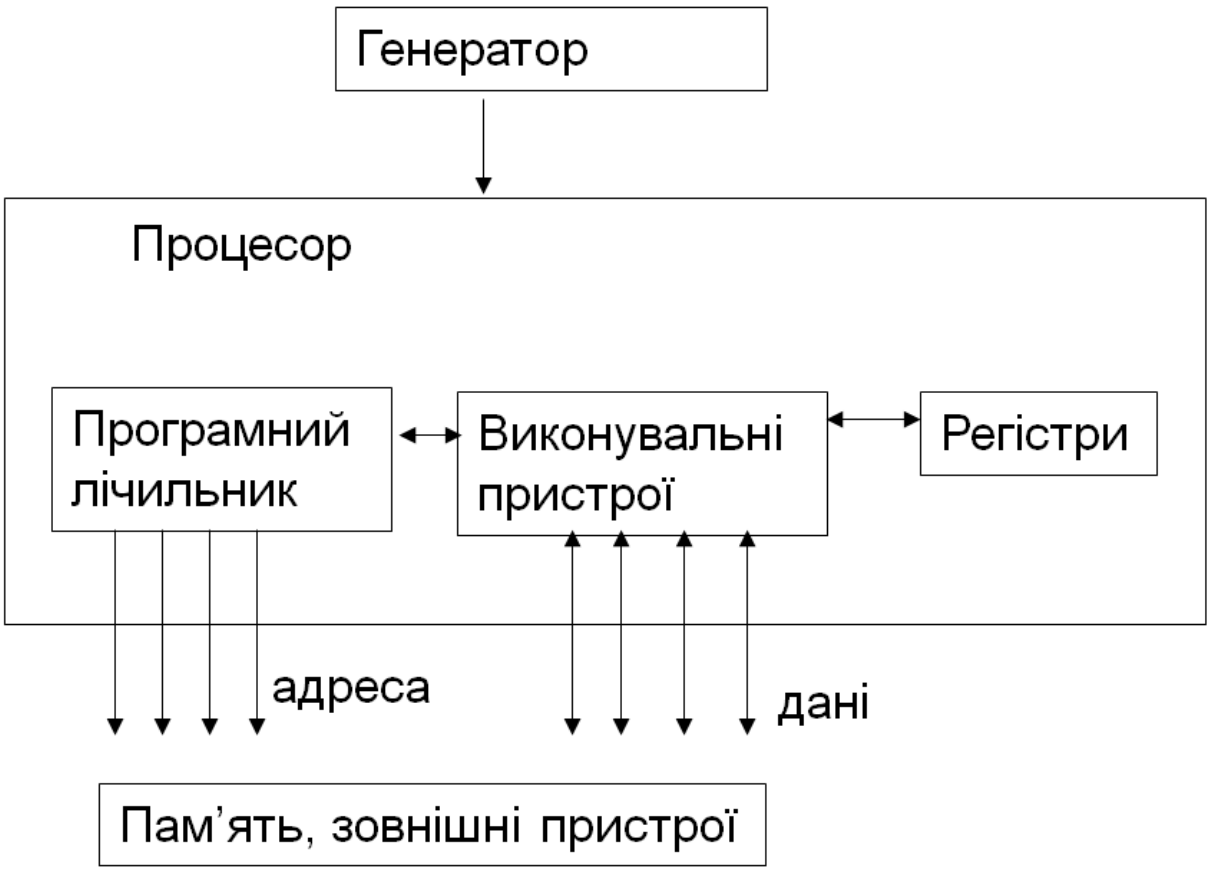
Застосування регістрів

- Передача даних різним пристроям по одним проводам
  - Такт1 – видача даних і запис в регістр пристрою 1
  - Такт2 – видача даних і запис в регістр пристрою 2
- Виконання математичних операцій з декількома порціями даних
  - Такт1 – видача даних1 і запис в регістр1
  - Такт2 - видача даних2 і запис в регістр 2
  - Такт3 дані1+дані2 і запис в регістр 3
  - Виконання команди – 3 такти
- Оперативна пам'ять
  - Такт 1 - По проводам адреси – дозвіл запису-читання в певний регістр
  - Такт 1 По проводам керування – керування записом-читанням
  - Такт 1 По проводам даних – передача, або прийом даних
  - Такт 2 По проводам тактування – запис або читання
- Збільшення кількості функцій за рахунок зменшення швидкості

[4.10]

Схема процесора

- Процесор
  - Програмний лічильник
  - Виконувальні пристрої (АЛП)
  - Регістри
- Як працює
  - На шину адреси видається адреса команди
  - Пам'ять видає команду чи дані за цією адресою
  - Виконувальний пристрій виконує роботу
  - Результати записуються регістри або в пам'ять



[4.11]

Арифметико-логічний пристрій (АЛУ, ALU, пристрій керування, функціональний пристрій)

- Керування роботою і виконання функцій
  - Дешифратор команд
  - Буферні регістри (для тимчасового зберігання команд і даних)
  - Арифметичні пристрої (додавання, віднімання, множення)

- Розрядність АЛУ
  - Максимальна кількість бітів які можуть бути оброблені за один такт
  - 2,4,8,16,32,64

- Система команд
  - Які команди може виконати процесор
  - ADD, MUL, AND, XOR, NOR…

[4.12]

Регістри процесора

- **Програмний лічильник** (регістр адреси, вказівник інструкції, Instruction Pointer, IP, Program Counter, PC)
  - Адреса наступної комірки пам'яті

- **Регістри загального призначення** (GPR, РЗП)
  - A,B,C,D - аргументи і результати виконання команд
- **Акумулятори** (A)
  - Регістри, які по замовчанню використовуються для операції
  - Команда **A=A ADD B** виглядає простіше **ADD B**

- **Регістр стану** (Status Register, SR, регість прапорців, ознак)
  - Чи було перенесення, чи був результат операції 0, …

- **Індексні регістри**
  - Спрощення адресації пам'яті

- **Регістри стеку** (магазинної пам'яті, BP,SP)
  - Спрощення адресації пам'яті

[4.13]

Система команд процесора

- Система команд (набір інструкцій)
  - Які дії може виконувати процесор
    - Арифметичні операції
    - Логічні операції
    - Запис-зчитування даних з пам'яті чи інших пристроїв
    - Виклик підпрограм …
  - Набір команд жорстко "зашитий" в дешифраторі команд процесора
  - Кожна команда має машинний код (унікальну бітову комбінацію)
  - Команди зберігається в зовнішній пам'яті і завантажуються в процесор
  - Деякі команди мають аргументи, тоді треба завантажити команду і аргументи
  - Мова асемблера – кожна команда процесора – одна окманда мови

- Різні команди виконуються впродовж різного часу
  - **Командний цикл** – послідовність дій виконання команди
  - **Мікрокоманда** – етап виконання команди
  - Регістр-регістр – швидкі команди (мало обміну із зовнішніми пристроями)
  - Регістр-пам'ять/ пристрій – повільніші команди (більше обміну із зовнішніми пристроями)
  - Команди без аргументів – швидші
  - Чим більше аргументів (1,2,3…) тим повільшіші

[4.14]

Шина (BUS)

- **Шина** - Набір проводів спільного призначення
  - Шина адреси – передача адреси команди чи даних в пам'яті
  - Шина даних – передача самої команди чи даних
  - Шина керування – керування пристроями
  - Внутрішні шини процесора

- До шин приєднано багато пристроїв
  - Внутрішні присрої процесора
  - Пам'ять
  - Зовнішні пристрої

- Розрядність шини – кількість проводів (бітів)

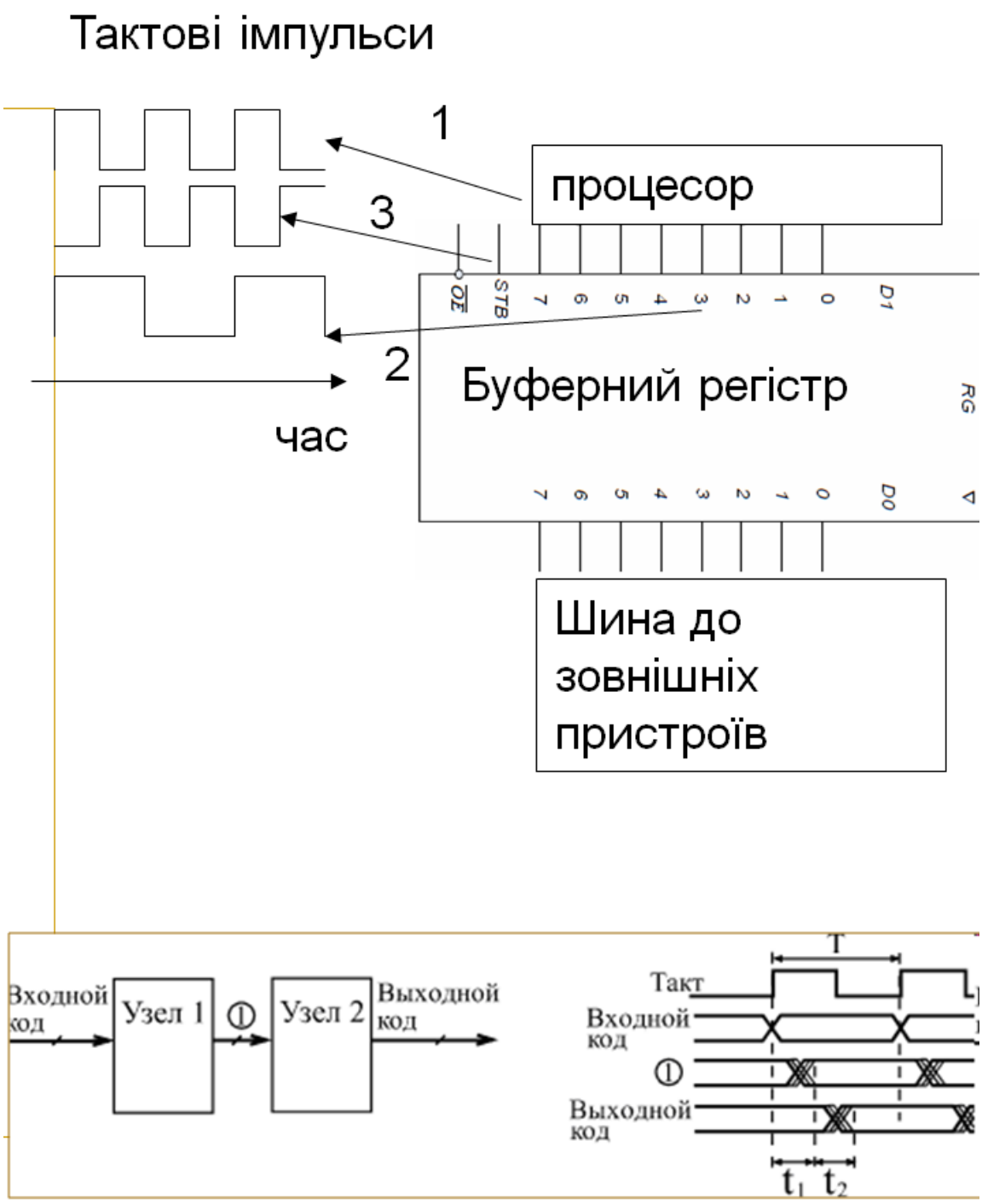
- Контролер шини – пристрій який керує обміном по шині
  - Буферний регістр – регістр запису-зчитування даних на шині
  - Шинний формувач – порт з високою навантажувальнорю здатністю

- Шини мають свої протоколи передачі інформації
  - Протокол – набір правил, якими керують пристрої для правильного обміну (1 передає – інші приймають тощо)

[4.15]

Контролер шини, буферні регістри, формувачі

- Багато пристроїв підключені до шини
  - Процесор не витримає такий струм
  - Можуть бути затримки через різні швидкість
- Буферний регістр
  - Висока навантажувальна спроможність
- Запис-зчитування даних
  - Тактовий імпульс 1 – виставлення даних процесором на шину
  - Дані доступні впродовж такту 2
  - Тактовий імпульс 3 (інверсний через півперіода) – запис в регістр і на шину
  - Дані передаються з затримкою на 1 такт
  - Відключення - переведення в стан високого вихідного опору (hold, утримання)
- Є двонаправлені, програмовані



[4.16]

Розрядність

- Шина адреси – найбільша розрядність
  - 8,16,32,64,128
- Шина даних
  - 8,16,32,64
- Шина керування
  - Часто суміщена з даними чи адресою
- Розрядність процесора
  - Розрядність шини даних 8,16,32
  - Розмір машинного слова – порція даних з якою ефективно може працювати процесор 8,16,32,64
  - Максимальна адресована область – розрядність лічильника комад 2^8,2^16,2^32,2^64
  - Максимальний об’єм пам’яті 2^8,2^16,2^32,2^64,2^128
  - Байт – одиниця передачі, адресації і збереження даних

[4.17]

Приклади команд процесора

- Без аргументів
  - HLT –зупинити роботу
  - MOV A,B пересилка з регістра B в A (A=B)
  - ADD B додати до акумулятора вміст регістра B
- З одним агрументом
  - MVI A,<B> пересилка значення <B> в регістр A (A=<B>)
- З одним аргументом великої довжини
  - LDA AB завантажити акумулятор значенням в області пам'яті з адресою AB
  - CALL AB викликати підпрограму за адресою області пам'яті AB
- Кожна вимагає певну кількість машинних тактів і циклів

[4.18]

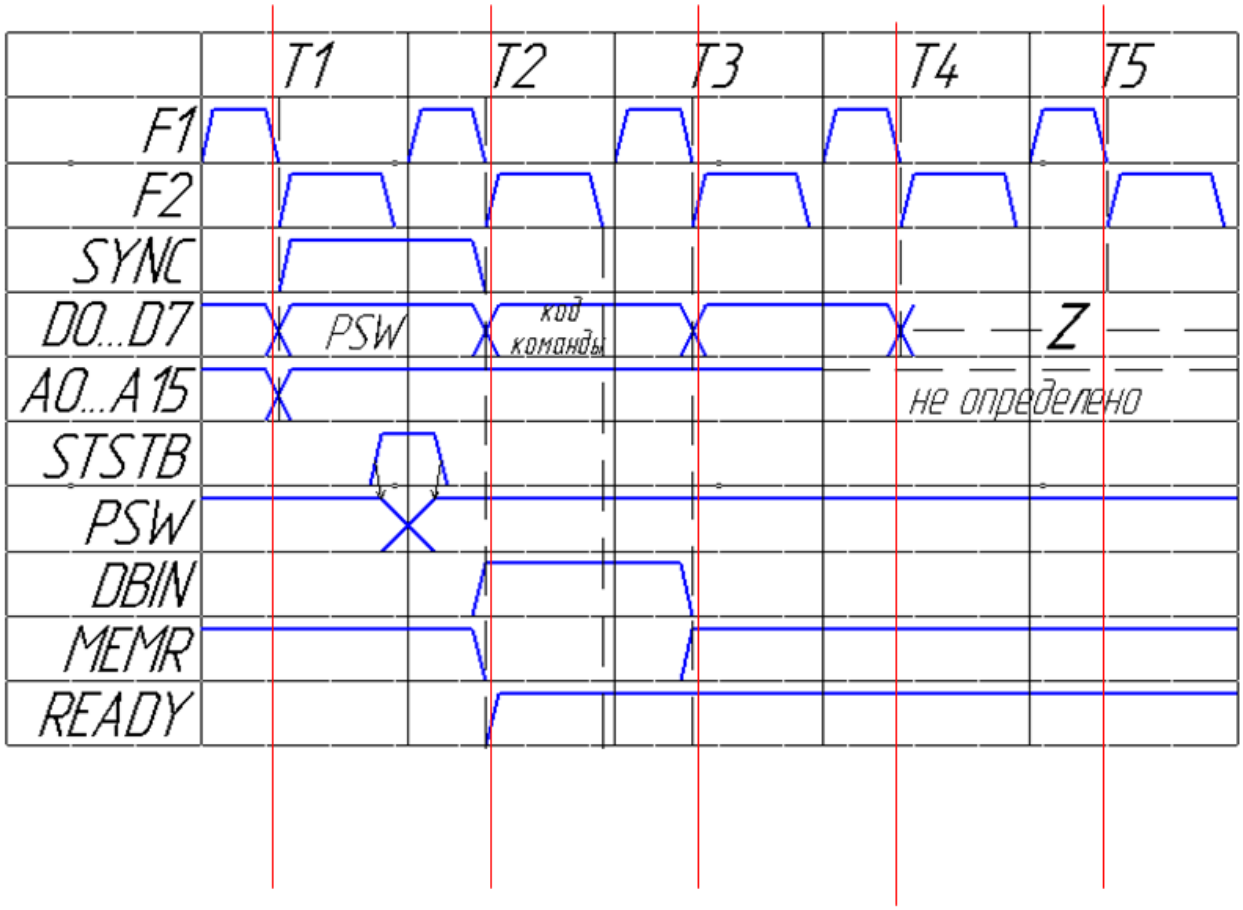
Такт, командний цикл, машинний цикл, мікрокоманда

- Час виконання команди – командний цикл
- Кожна пересилка даних по шині – машинний цикл
  - Командний цикл 1-7..10 машинних циклів
  - Пересилка по шині даних 1-2 байти
- Машинний цикл
  - Декілька мікрокоманд
- Мікрокоманда – дії за один такт
  - Такт 1. Видача на шину даних байта стану, на шину адреси – адреси команди, на шину керування сигналу синхронізації
  - Такт 2. Очікування готовності зовнішнього пристрою чи пам'яті
  - Такт 3 Отримання команди
  - ……
  - Такт N Виконання

[4.19]

Приклад машинного цикла

- Такт1 –видача адреси команди і слова стану
  - Інформація про те, що процесор "хоче" робити
- Такт 2 – перевірка готовності зовнішній пристроїв
  - Перевіряє поки не буде сигналу READY
- Такт 3 – читає команду по шині адреси
- Такт 4 – виконання команди
- Може бути багато циклів і тактів в залежності від команди
  - 1-7 циклів 3-25 тактів



В сучасних процесорах
за такт може виконуватись декілька операцій

[4.20]

Особливості роботи з пам'яттю

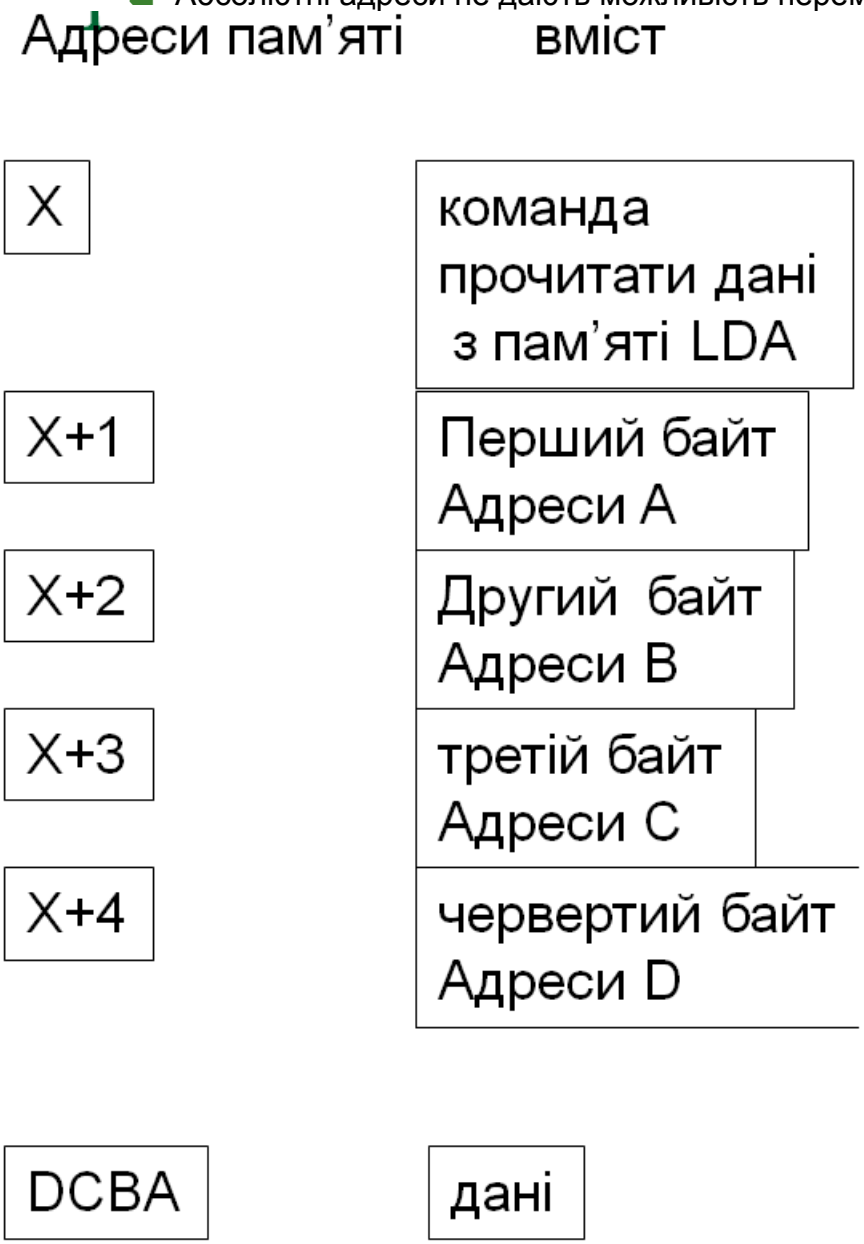
- Кожен елемент даних (байт, слово) має адресу в пам'яті
  - При виставлення адреси на шину адреси на шині даних контролер пам'яті видає дані з цієї комірки
- Пряма адресація
  - Адрес пам'яті є параметром команди
  - Повільна передача по шині
- Непряма адресація
  - Адреса пам'яті знаходиться в регістрі
  - Швидка передача по шині
- Індексна адресація
  - Адреса пам'яті знаходится в регістрі
  - В іншому регістрі знаходиться зміщення
  - Ефективно оптимізується
- Стек
  - В регістрі знаходиться адреса
  - При записі до адреси дані пересилаються і адреса зменшується на 1
  - При зчитуванні дані пересилаються і адреса збільшується на 1
  - Ефективна для викликів підпрограм

[4.21]

Пряма (безпосередня) адресація

- Адреса вказується як параметр команди
- LDA DCBA – завантажити в аккумулятор дані, які зберігаються в пам'яті за адресою DCBA
- Застосовується рідко

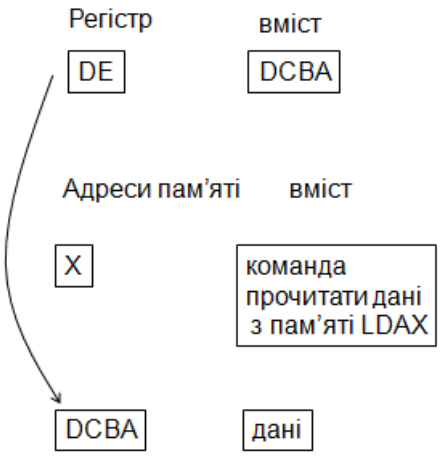
- 5 циклів читання команди
- 1 цикл читання даних
- Дуже повільно (~20 тактів)
- Абсолютні адреси не дають можливість переміщувати дані в пам'яті



[4.22]

Непряма (регістрова) адресація

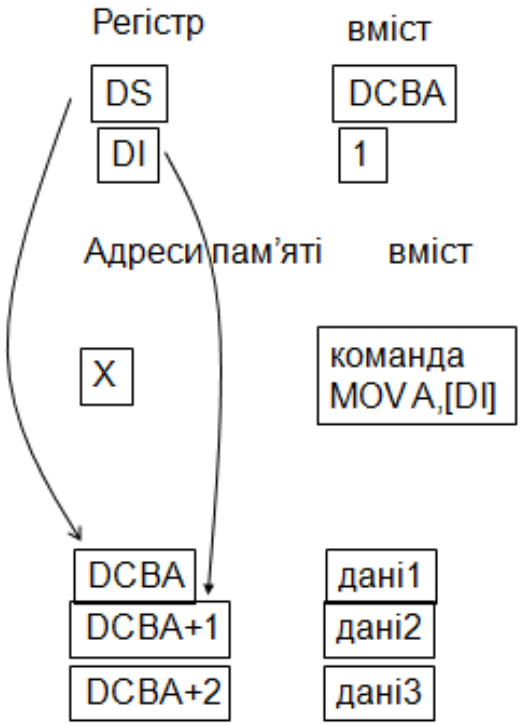
- Адреса зберігається в регістрі
- LDAX завантажити в аккумулятор дані, які зберігаються за адресою пам'яті, яка записана в регістрі DE
- Застосовується часто
  - 1 цикл читання команди
  - 1 цикл читання даних
  - Швидко (~5 тактів)
  - Легко переміщувати дані в пам'яті



[4.23]

Індексна адресація – спеціальна форма непрямої

- Адреса зберігається в одному регістрі, зміщення – в іншому
- MOV A,[DI] завантажити дані, які містяться в пам'яті за адресою, яка рівна сумі значень в регістрі DS та DI
- Широко застосовується
  - 1 цикл читання команди
  - 1 цикл читання даних
  - Швидко (~5 тактів)
  - Ефективно працювати з масивами (кожна наступна команда 3 такти)

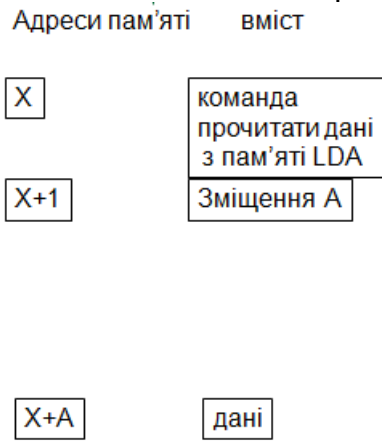


[4.24]

Пряма відносна адресація – спеціальна форма прямої

- В параметрі команди вказується зміщення відносно адреси команди
- Часто застосовується
  - 2 цикли читання команди
  - 1 цикл читання даних

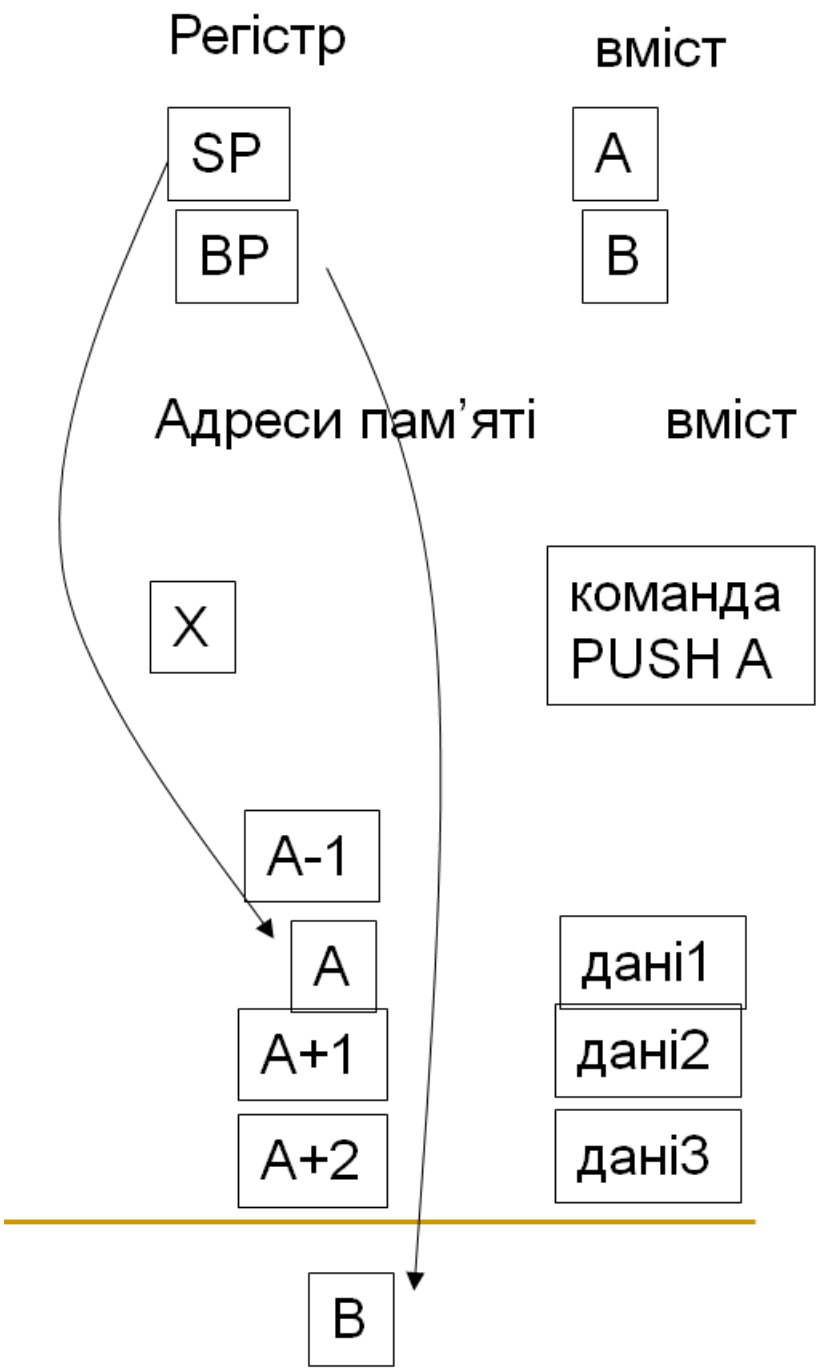
- Можн апереміщувати програму в пам’яті



[4.25]

Стек - спеціальна форма непрямої адресації

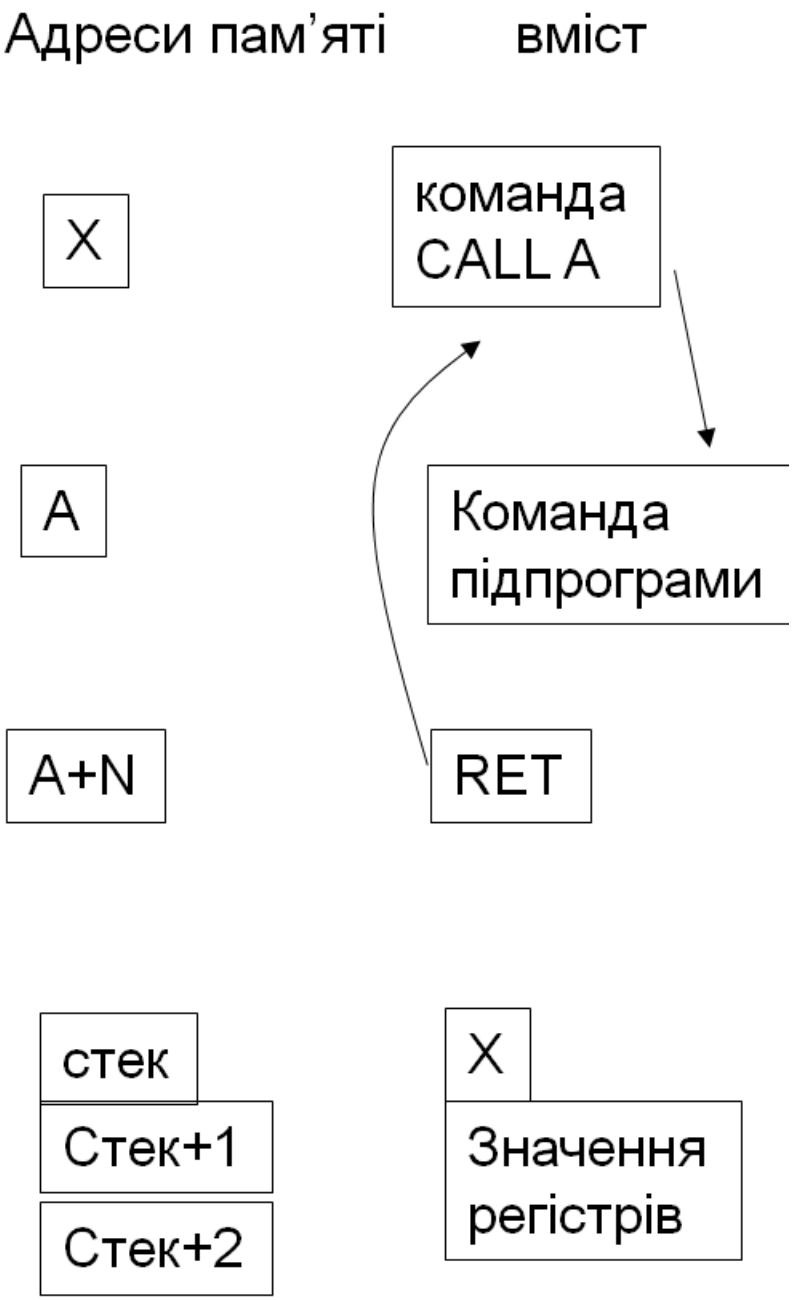
- Останнім зайшов-першим вийшов
- В регістрі SP (stack pointer) зберігається адреса пам’яті де знаходяться останні записані дані
- Запис даних PUSH A
  - Значення SP зменшується на1
  - Дані з A записуються за новим значенням SP (у вершину стека)
- Читання даних POP A
  - Дані за адресою SP зчитуються в регістр A
  - Значення SP збільшується на1
- Регістр BP-основа стека ([SP]>=[BP])



[4.26]

Виклик підпрограм

- Підпрограма – частина програми, яка може викликатись з різних частин програми
- Команда виконання підпрограми CALL ADDR
  - Запис в стек поточного значення командного лічильника
  - Запис в стек регістра стану і інших регістрів
  - Перехід за адресою ADDR
- Команда повернення з підпрограми RET
  - Відновлення із стеку регістра стану і інших регістрів
  - Відновлення командного лічильника



[4.27]

Області пам'яті

- Часто область пам'яті ділять на функціональні частини
- Сегмент коду (Code segment, CS)
  - Адреси пам'яті в яких зберігаються команди процесора
- Сегмент даних (Data Segment DS)
  - Адреси пам'яті в яких зберігаються дані
- Стек (Stack Segment, SS)
  - Адреси пам'яті в яких знаходиться стек
- Куча
  - Адреси пам'яті, які використовуються для тимчассового виділення пам'яті

[4.28]

Архітектури ЕОМ – концепція побудови

- фон-Нейманівська архітектура
  - Процесор відділений від пам’яті
  - Система команд не міняється
  - Одна й та є сама пам’ять може використовуватись для команд, даних, стеку, кучі, тощо
  - Пам’ять адресується послідовно
  - Програма виконується послідовно
  - Надзвичайно поширена і універсальний підхід
- Гарвардська архітектура
  - Команди і дані зберігаються в різних областях пам’яті і передаються по різним каналам
  - Модифікована гарвардська архітектура – в середині і ззовні процесора різні шини адрес і даних
  - Переваги: пам’ять програм може бути більшою і дешевшою ніж даних, можлива паралельна передача команд і даних
  - Недоліки – не універсальна
- Гібридні архітектури
  - Різний кеш команд і даних, але спільна оперативна пам’ять