ЦИФРОВІ СИГНАЛЬНІ ПРОЦЕСОРИ

1. **Загальна характеристика й архітектурні особливості ЦСП**
2. **MAC-операції й організація пам'яті ЦСП**
3. **Інструментальні засоби розробки систем на основі ЦСП**

Цифрові сигнальні процесори (ЦСП) представляють собою спеціалізовані процесори з RISC-архітектурою, апаратно й програмно оптимізовані на рішення задач цифрової обробки сигналів (ЦОС). Вони відрізняються від звичайних мікропроцесорів своєю високою продуктивністю, обумовленої особливостями їх архітектури й системи команд.

Загальна характеристика й архітектурні особливості ЦСП

Сигнальні процесори з'явилися на початку 80-х років (однокристальна мікро-ЕОМ μPD 7720 японської корпорації NEC). Проте вже в 1982 році лідерство в цій галузі захопила американська фірма Texas Instruments, що випустила сигнальний процесор TMS32010.

Друге покоління ЦСП з'явилося в середині 80-х років. Підвищення ступеня інтеграції дозволило розширити функції ЦСП. Збільшилася швидкість виконання команд і рівень розпаралелювання обробки даних. Введено апаратну підтримку кільцевих буферів і циклів DO-UNTIL, що виключає непродуктивні витрати часу на умовні переходи. Були освоєні вітчизняні аналоги ЦСП другого покоління.

Кінець 80-х років ознаменувався переходом до третього покоління ЦСП. Процесори третього покоління виконані за субмікронною технологією й підтримують операції із плаваючою комою (точкою) без втрати у швидкості обробки.

У наш час основними виробниками ЦСП є фірми: Texas Instruments - сімейства ЦСП TMS320Cxx з фіксованою й плаваючою точкою, Motorola - сімейства DSP56xxx з фіксованою точкою й DSP960xx - із плаваючою, AT&T Microelectronics - сімейства DSP16 з фіксованою точкою й DSP32 - із плаваючою, Analog Devices - сімейства ADSP-21xx з фіксованою точкою й ADSP-21xxx - із плаваючою точкою.

Загальна характеристика й архітектурні особливості ЦСП

В основу побудови ЦСП покладені наступні принципи:

- використання гарвардської архітектури;
- скорочення тривалості командного циклу;
- застосування конвеєризації;
- застосування апаратного помножувача;
- оптимізація системи команд на задачі ЦОС;

Загальна архітектура ЦСП, що відбиває їх особливості, наведена на рис.1.

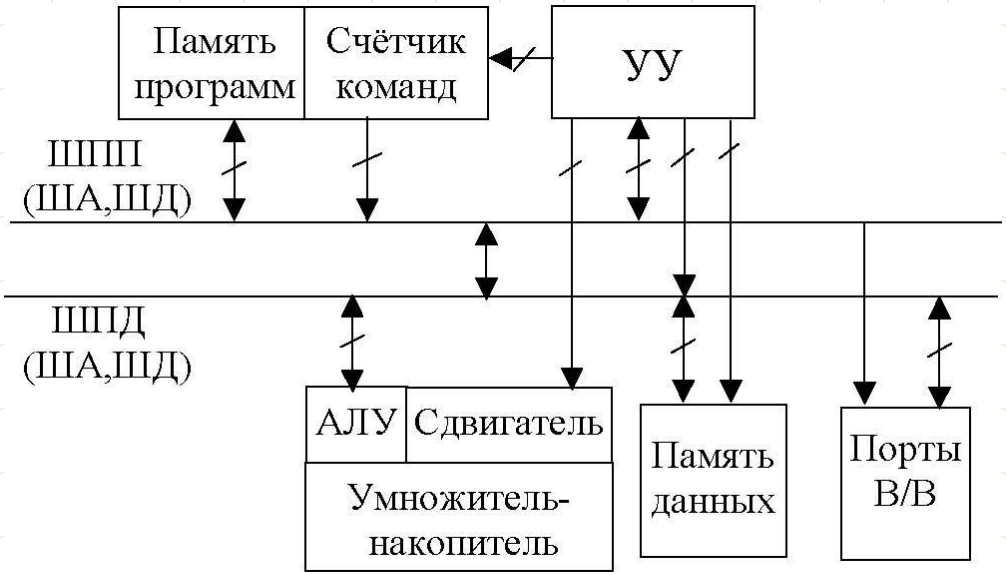


Рис.1

MAC-операції й організація пам'яті ЦСП

Більшість ЦСП здатні виконувати операцію *множення з накопиченням* за один командний цикл. Дану операцію звичайно позначають абревіатурою MAC (Multiply Accumulate). Суть MAC операції виражається оператором:

$$R \leftarrow R + a \cdot x$$

Відповідно до цього оператора спочатку обчислюється добуток значень змінних *a* й *x,* а потім отриманий добуток складається зі значенням змінної R. Результат знову запам'ятовується в змінної R.

До виконання MAC-операцій зводяться багато алгоритмів обробки сигналів: КІХ і БІХ фільтрація, обчислення кореляційних функцій, БПФ, множення векторів і матриць й ін.

Наприклад, рівняння лінійної фільтрації припускає обчислення суми добутків.

$$y(nT) = \sum_{m=1}^{M-1} a_m y(nT - mT) + \sum_{k=0}^{N-1} b_k x(nT - kT).$$

MAC-операції й організація пам'яті ЦСП

Схеми реалізації MAC -операцій утворять ядро ЦСП. Для того щоб ефективно виконувати MAC -операцію, потрібне раціональна взаємодія таких схем з пам'яттю.

Звернемося до приклада нерекурсивного фільтра, який працює за різницевим рівнянням

$$y(nT) = \sum_{k=0}^{N-1} b_k x(nT - kT)$$

Обчислення вихідної реакції фільтра зводиться до багаторазового виконання MAC-операції. При цьому в ході виконання однієї MAC-операції необхідно здійснити кілька звернень до пам'яті:

- вибрати MAC-команду з пам'яті,
- виконати зчитування аргументу X (відліки вхідного сигналу) з пам'яті,
- виконати зчитування аргументу B (коефіцієнти) з пам'яті,
- здійснити переміщення значень сигналу уздовж лінії затримки КІХ-фільтра.

MAC-операції й організація пам'яті ЦСП

Отже, процесор за один цикл команди повинен чотири рази звернутися до запам'ятовувального пристрою Таку вимогу, що висувають до ЦСП, називають *багаторазовим доступом до пам'яті.*

Класична архітектура процесора з одним банком пам'яті не дозволяє реалізувати багаторазовий доступ. Тому ЦСП будують на основі *Гарвардської архітектури* (рис.2), відповідно до якої процесорне ядро взаємодіє із двома банками пам'яті А и В.

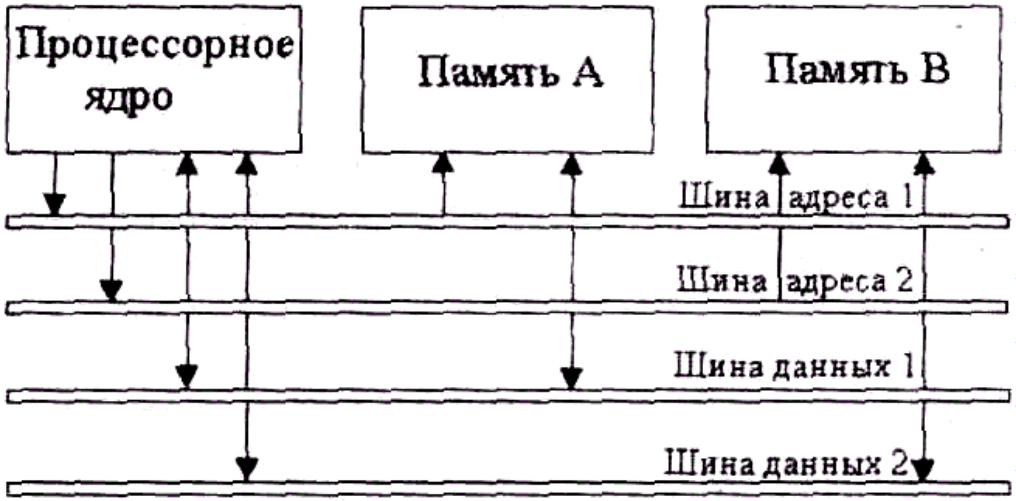


Рис.2

MAC-операції й організація пам'яті ЦСП

Звернення до кожного з банків пам'яті виконується за допомогою двох незалежних шин адреси й даних. У Гарвардській архітектурі один з банків пам'яті використовується для зберігання програм, а інший для зберігання даних.

Зазвичай використовується модифікована Гарвардська архітектура. У цьому випадку один з банків зберігає як програми, так і дані, а інший - тільки дані. Гарвардська архітектура дозволяє процесорному ядру за один цикл команди паралельно звертатися до пам'яті даних і пам'яті програм.

Зазначену архітектуру мають більшість ЦСП. Деякі ЦСП використовують три банки пам'яті із трьома незалежними парами шин адреси-даних.

MAC-операції й організація пам'яті ЦСП

Поряд з Гарвардською архітектурою, застосовують й інші прийоми, щоб забезпечити багаторазовий доступ до пам'яті Зокрема, у ЦСП застосовують швидкодіючу пам'ять, що має можливість двох послідовних доступів за один цикл команди. Два таких банки пам'яті можуть забезпечити чотири звернення за командний цикл і відповідно реалізувати вимоги до організації взаємодії з пам'яттю з боку алгоритму КІХ-фільтрації.

Інший прийом підвищення ефективності взаємодії з пам'яттю полягає в застосуванні *пам'яті з декількома портами доступу.* Така пам'ять має звичайно дві незалежних пари шин адреси й даних і забезпечує паралельний доступ по двох адресах.

MAC-операції й організація пам'яті ЦСП

Комбінація Гарвардської архітектури зі швидкодіючою й багатопортовою пам'яттю дозволяє реалізувати вимоги до ефективної організації взаємодії з пам'яттю, висунуті з боку алгоритмів ЦОС. Розглянуті можливості відносяться до пам'яті, що розташовується на кристалі. Взаємодія із зовнішньою пам'яттю звичайно виконується за допомогою однієї шини адреси й однієї шини даних. Для цього внутрішні шини ЦСП мультиплексуються.

Деякі ЦСП мають спеціальний *кеш команд,* що представляє собою пам'ять невеликого об'єму, яка входить до складу ядра процесора й призначена для зберігання команд. Попереднє завантаження команд у кеш дозволяє звільнити шини процесора для доступу до даних. Можливості кешування команд значно відрізняються в різних сімействах ЦСП.

Інструментальні засоби розробки систем на основі ЦСП

ЦСП, власне кажучи, є не що інше як мікропроцесор спеціального застосування. Як і для будь-якого іншого мікропроцесора, створення систем на основі ЦСП неможливо без інструментальних засобів підтримки розробок. Дані засоби дозволяють розробляти архітектуру апаратних засобів і здійснювати створення й налагодження відповідного програмного забезпечення. Вони застосовуються на всіх етапах розробки - від досліджень до виготовлення й тестування. Багато в чому успіх створення систем на основі ЦСП визначається наявністю розвинених інструментальних засобів підтримки розробок. У цей час практично для всіх ЦСП є такі засоби.

Інструментальні засоби розробок систем на основі ЦСП поділяються на програмні й апаратні. Загальна схема розробки системи ЦОС із використанням інструментальних засобів зображена на рис.3.

Інструментальні засоби розробки систем на основі ЦСП

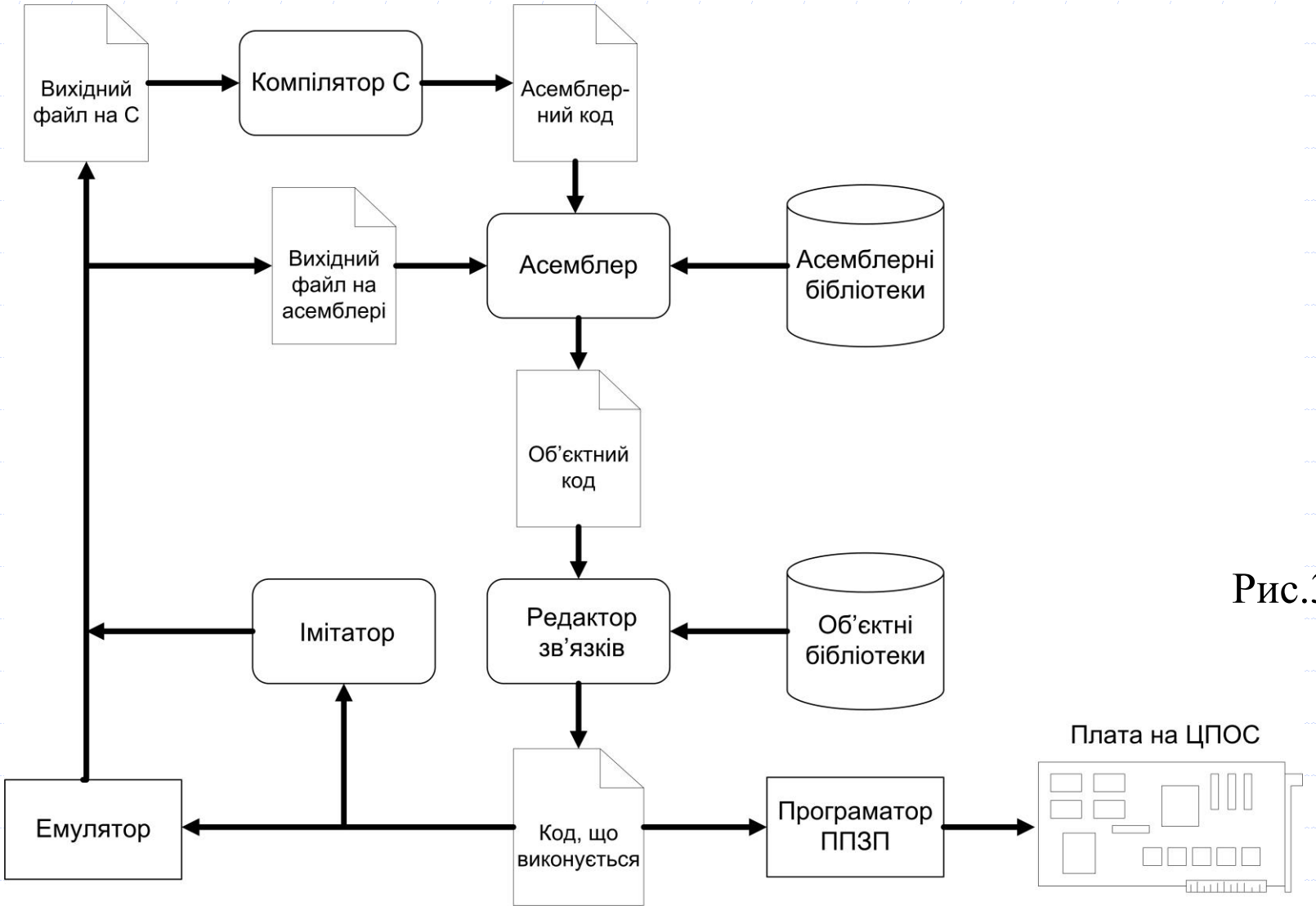


Рис.3

Інструментальні засоби розробки систем на основі ЦСП

Типовий набір *програмних* засобів розробки включає наступні програми:

- *Асемблер* транслює вихідну програму мовою асемблера в програму, що підлягає виконанню в об'єктному коді, тобто в коді, що безпосередньо сприймається ЦСП. Звичайно асемблер генерує об'єктний код, що потім розміщається в пам'яті системи ЦОС за допомогою редактора зв'язків.
- *Редактор зв'язків* дозволяє розробляти програму у вигляді автономних модулів, зв'язавши які воєдино, можна одержати повний код програми. Редактор зв'язків виконує об'єднання асембльованих об'єктних файлів і додаткових об'єктних файлів, що зберігаються в бібліотеках, у єдину програму, що виконується.
- *Асемблерні бібліотеки* містять типові арифметичні під програми й підпрограми ЦОС. Для створення бібліотек використовують спеціальні програми-бібліотекарі.

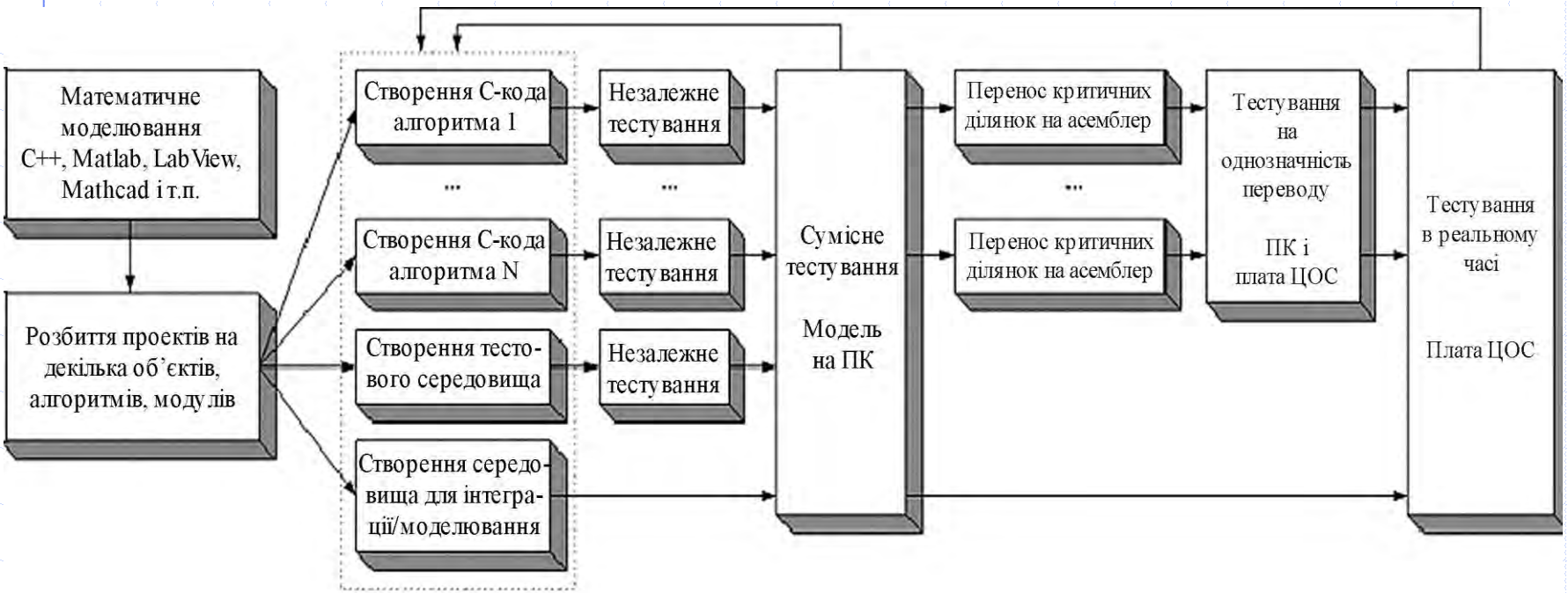
Інструментальні засоби розробки систем на основі ЦСП

- *Імітатор* програмним способом моделює ЦСП на рівні системи команд, дозволяючи тим самим перевіряти й налагоджувати програми без наявності апаратних засобів. Він використовує об'єктний код, що формується асемблером і редактором зв'язків. Імітатор моделює структуру пам'яті й можливості вводу-виводу в системі ЦОС. Для цього він використовує спеціальний файл, що визначає архітектуру розроблювальної системи. Звичайно програма-імітатор має віконний інтерфейс для взаємодії з користувачем. Імітатор представляє потужний засіб налагодження, що допускає покрокове виконання команд, імітацію переривань, установку точок програмного переривання, відображення вмісту пам'яті.
- *Компілятор* з мови Сі переводить вихідні тексти програм, написані на стандартній (ANSI) версії мови Сі в об'єктний код, або в код мовою асемблера. Компілятор звичайно допускає включення в програму, написану мовою Сі, розділів мовою асемблера. Це дозволяє програмістові самостійно оптимізувати критичні ділянки програми ЦОС.

Інструментальні засоби розробки систем на основі ЦСП

- *Налагоджувач* програм, написаних мовою Сі, забезпечує відображення змінних і значень виразів, оцінювання значень виразів, покрокове виконання інструкцій, розміщення точок програмних переривань, відображення дерева виклику функцій й ін.
- *Бібліотеки мови Сі* містять всі стандартні функції й функції, що застосовуються при обробці сигналів. Наприклад, функції цифрової фільтрації, швидкого перетворення Фур'є, матричних операцій, обробки переривань.
- *Програматор ППЗП* транслює код, призначений для виконання процесором ЦОС, в формат для програмування відповідного типу ППЗП.

ЕТАПИ РОЗРОБКИ ПРОЕКТУ СИСТЕМИ ЦИФРОВОЇ ОБРОБКИ СИГНАЛІВ НА ОСНОВІ СИГНАЛЬНИХ ПРОЦЕСОРІВ



Інструментальні засоби розробки систем на основі ЦСП

Апаратні засоби підтримки розробок базуються на використанні внутрісхемних емуляторів. *Внутрісхемний емулятор* дозволяє виконувати налагодження й оптимізацію додатка шляхом його виконання на фізичній моделі системи ЦОС. Такі емулятори складаються зі спеціальних плат, побудованих на основі відповідного ЦСП і програмного забезпечення. Програмне забезпечення завантажується в ПК й управляє платою, що представляє систему, що реконфігурується, на основі обраного ЦСП. Така система може виконувати додаток ЦОС, що завантажує в неї. Внутрісхемний емулятор дозволяє виконувати покрокове виконання програм з відображенням умісту регістрів ЦСП і пам'яті.

Інструментальні засоби розробки систем на основі ЦСП

У порівнянні із програмуванням цифрових обчислювальних пристроїв загального застосування, програмування ЦСП вимагає ефективної реалізації алгоритмів, що працюють у реальному масштабі часу. При цьому велика увага приділяється оптимізації програм, які пишуться на асемблері. Такий підхід є досить трудомістким і дорогим. Процес проектування нових систем ЦОС стає простіше, якщо є весь арсенал засобів, зазначених на рис.3.

Зокрема, розробка систем ЦОС значно спрощується, якщо виконувати програмування мовою Сі. Однак у ряді випадків це приводить до втрати швидкодії програм. Тому на практиці звичайно використовують комбіновану реалізацію алгоритмів ЦОС: одна частина алгоритмів програмується мовою Сі, а інша - мовою асемблера Програмування алгоритмів ЦОС мовою Сі не має яких не будь істотних особливостей.