Тема 8. Система команд процесора

8.1 Основні групи команд процесора
8.2 Особливості виконання різних команд
8.3 Методи організації підпрограм.

8.1 Основні групи команд процесора

У загальному випадку система команд процесора включає в себе наступні чотири основні групи команд:

• команди пересилки даних;
• арифметичні команди;
• логічні команди;
• команди переходів.

Команди пересилання даних не вимагають виконання ніяких операцій над операндами. Операнди просто пересилаються (точніше, копіюються) з джерела (Source) в приймач (Destination). Джерелом і приймачем можуть бути внутрішні регістри процесора, комірки пам'яті або пристрою вводу / виводу. АЛУ в даному випадку не використовується.

Арифметичні команди виконують операції додавання, віднімання, множення, ділення, збільшення на одиницю (інкрементування), зменшення на одиницю (декрементування) і т.д. Цим командам потрібно один або два вхідних операнда. Формує команди один вихідний операнд.

Логічні команди виробляють над операндами логічні операції, наприклад, логічне І, логічне АБО, що виключає АБО, очищення, різноманітні зрушення (вправо, вліво, арифметичні зрушення, циклічний зсув). Цим командам, як і арифметичним, потрібно один або два вхідних операнда, і формують вони один вихідний операнд.

Нарешті, команди переходів призначені для зміни звичайного порядку послідовного виконання команд. З їх допомогою організовуються переходи на підпрограми і повернення з них, всілякі цикли, розгалуження програм, пропуски фрагментів програм і т.д. Команди переходів завжди змінюють вміст лічильника команд. Переходи можуть бути умовними і безумовними. Саме ці команди дозволяють будувати складні алгоритми обробки інформації.

Відповідно до результату кожної виконаної команди встановлюються або очищаються біти регістра стану процесора (PSW). Але треба пам'ятати, що не всі команди змінюють всі наявні в PSW прапори. Це визначається особливостями кожного конкретного процесора.

У різних процесорів системи команд істотно розрізняються, але в основі своїй вони дуже схожі. Кількість команд у процесорів також по-різному. Наприклад, у згадуваного вже процесора MC68000 всього 61 команда, а у процесора 8086 - 133 команди. У сучасних потужних процесорів кількість команд досягає декількох сотень. У той же час існують процесори з скороченим набором команд (так звані RISC-процесори), в яких за рахунок максимального скорочення кількості команд досягається збільшення ефективності і швидкості їх виконання.

Розглянемо тепер особливості чотирьох виділених груп команд процесора більш докладно.

Команди пересилання даних займають дуже важливе місце в системі команд будь-якого процесора. Вони виконують такі найважливіші функції:

• завантаження (запис) вмісту у внутрішні регістри процесора;

• збереження в пам'яті вмісту внутрішніх регістрів процесора;

• копіювання вмісту з однієї області пам'яті в іншу;

• запис в пристрої введення / виводу і читання з пристроїв введення / виводу.

У деяких процесорах (наприклад, T-11) всі ці функції виконуються однією єдиною командою MOV (для байтових пересилань - MOVB) але з різними методами адресації операндів.

В інших процесорах крім команди MOV є ще кілька команд для виконання перерахованих функцій. Наприклад, для завантаження регістрів можуть використовуватися команди завантаження, причому для різних регістрів - різні команди (їх позначення зазвичай будуються з використанням слова LOAD - завантаження). Часто виділяються спеціальні команди для збереження в стеці і для вилучення з стека (PUSH - зберегти в стеці, POP - витягти з стека). Ці команди виконують пересилання з Автоінкрементний і з автодекрементной адресацією (навіть якщо ці режими адресації не зраджу-дивіться в процесорі в явному вигляді).

Іноді в систему команд вводиться спеціальна команда MOVS для малої (або ланцюгової) передачі даних (наприклад, в процесорі 8086). Ця команда передає не одне слово або байт, а задану кількість слів або байтів (MOVSB), тобто ініціює не один цикл обміну по магістралі, а кілька. При цьому адреса пам'яті, з якою відбувається взаємодія, збільшується на 1 або на 2 після кожного звернення або ж зменшується на 1 або на 2 після кожного звернення. Тобто в неявному вигляді застосовується автоінкрементна або автодекрементна адресація.

У деяких процесорах (наприклад, в процесорі 8086) спеціально виділяються функції обміну з пристроями введення / виводу. Команда IN

використовується для введення (читання) інформації з пристрою вводу / виводу, а команда OUT використовується для виведення (записи) в пристрій вводу / виводу. Обмін інформацією в цьому випадку проводиться між регістром - акумулятором і пристроєм вводу / виводу. У більш просунутих процесорах цього ж сімейства (починаючи з процесора 80286) додані команди сатиричного (ланцюгового) введення (команда INS) і сатиричного виведення (команда OUTS). Ці команди дозволяють пересилати цілий масив (рядок) даних з пам'яті в пристрій вводу / виводу (OUTS) або з пристрою вводу / виводу в пам'ять (INS). Адреса пам'яті після кожного звернення збільшується або зменшується (як і у випадку з командою MOVS).

Також до команд пересилання даних відносяться команди обміну інформацією (їх позначення будується на основі слова Exchange). Може бути передбачений обмін інформацією між внутрішніми регістрами, між двома половинами одного регістра (SWAP) або між регістром і осередком пам'яті.

Арифметичні команди розглядають коди операндів як числові виконавчі або двійково-десяткові коди. Ці команди можуть бути розділені на п'ять основних груп:

• команди операцій з фіксованою комою (додавання, віднімання, множення, ділення);

• команди операцій з плаваючою комою (додавання, віднімання, множення, ділення);

• команди очищення;

• команди інкремента і декремента;

• команда порівняння.

Команди операцій з фіксованою комою працюють з кодами в регістрах процесора або в пам'яті як зі звичайними двійковими кодами. Команда складання (ADD) обчислює суму двох кодів. Команда віднімання (SUB) обчислює різницю двох кодів. Команда множення (MUL) обчислює добуток двох кодів (розрядність результату вдвічі більше розрядності співмножників). Команда ділення (DIV) обчислює частку від ділення одного коду на інший. Причому всі ці команди можуть працювати як з числами зі знаком, так і з числами без знаку.

Команди операцій з плаваючою комою (крапкою) використовують формат уявлення чисел з порядком і мантисою (зазвичай ці числа займають дві послідовні комірки пам'яті). В сучасних потужних процесорах набір команд з плаваючою комою не обмежується тільки чотирма арифметичними діями, а містить і безліч інших більш складних команд, наприклад, обчислення тригонометричних функцій,

логарифмічних функцій, а також складних функцій, необхідних при обробці звуку і зображення.

Команди очищення (CLR) призначені для запису нульового коду в регістр або елемент пам'яті. Ці команди можуть бути замінені командами пересилання нульового коду, але спеціальні команди очищення зазвичай виконуються швидше, ніж команди пересилання. Команди очищення іноді відносять до групи логічних команд, але суть їх від цього не змінюється.

Команди інкремента (збільшення на одиницю, INC) і декремента (зменшення на одиницю, DEC) також бувають дуже зручні. Їх можна в принципі замінити командами підсумовування з одиницею або віднімання одиниці, але інкремент і декремент виконуються швидше, ніж підсумовування і віднімання. Ці команди вимагають одного вхідного операнда, який одночасно є і вихідним операндом.

Нарешті, команда порівняння (позначається CMP) призначена для порівняння двох вхідних операндів. По суті, вона обчислює різницю цих двох операндів, але вихідного операнда не формує, а всього лише змінює біти в регістрі стану процесора (PSW) за результатом цього вирахування. Наступна за командою порівняння команда (зазвичай це команда переходу) буде аналізувати біти в регістрі стану процесора і виконувати дії в залежності від їх значень . У деяких процесорах передбачені команди ланцюгового порівняння двох послідовностей операндів, що містяться в пам'яті (наприклад, в процесорі 8086 і сумісних з ним).

Логічні команди виконують над операндами логічні (побітові) операції, тобто вони розглядають коди операндів не як єдине число, а як набір окремих бітів. Цим вони відрізняються від арифметичних команд. Логічні команди виконують такі основні операції:

• логічне І, логічне АБО, додавання по модулю 2 (виключає Або);

• логічні, арифметичні та циклічні зрушення;

• перевірка бітів і операндів;

• установка і очищення бітів (прапорів) регістра стану процесора (PSW).

Команди логічних операцій дозволяють побитно обчислювати основні логічні функції від двох вхідних операндів. Крім того, операція І (AND) використовується для примусового очищення заданих бітів (в якості одного з операндів при цьому використовується код маски, в якому розряди, що вимагають очищення, встановлені в нуль). Операція АБО (OR) застосовується для примусової установки заданих бітів (в якості одного з операндів при цьому використовується код маски, в якому розряди, що вимагають установки в одиницю, дорівнюють одиниці).

Операція «Що виключає Або» (XOR) використовується для інверсії заданих бітів (в якості одного з операндів при цьому застосовується код маски, в якому біти, що підлягають інверсії, встановлені в одиницю). Команди вимагають двох вхідних операндів і формують один вихідний операнд.

Команди зрушень дозволяють побитно зрушувати код операнда вправо (в сторону молодших розрядів) або вліво (в сторону старших розрядів). Тип зсуву (логічний, арифметичний або циклічний) визначає, яким буде нове значення старшого біта (при зсуві вправо) або молодшого біта (при зсуві вліво), а також визначає, чи буде десь збережено колишнє значення старшого біта (при зсуві вліво) або молодшого біта (при зсуві вправо). Наприклад, при логічному зсуві вправо в старшому розряді коду операнда встановлюється нуль, а молодший розряд записується як прапор перенесення в регістр стану процесора. А при арифметичному зсуві вправо значення старшого розряду зберігається колишнім (нулем або одиницею), молодший розряд також описується як прапор перенесення.

Циклічні зрушення дозволяють зрушувати біти коду операнда по колу (за годинниковою стрілкою при зсуві вправо або проти годинникової стрілки при зсуві вліво). При цьому в кільце зсуву може входити чи не входити прапор переносу. У біт прапора перенесення (якщо він використовується) записується значення старшого біта при циклічному зсуві вліво і молодшого біта при циклічному зсуві вправо. Відповідно, значення біта прапора перенесення буде переписуватися в молодший розряд при циклічному зсуві вліво і в старший розряд при циклічному зсуві вправо.

Для прикладу на рис. 8.1 показані дії, що виконуються командами зрушень вправо.

Команди перевірки бітів і операндів призначені для установки або очищення бітів регістра стану процесора в залежності від значення обраних бітів або всього операнда в цілому. Вихідного операнда команди не формують. Команда перевірки операнда (TST) перевіряє весь код операнда в цілому на рівність нулю і на знак (на значення старшого пита), вона вимагає тільки одного вхідного операнда. Команда перевірки біта (BIT) перевіряє тільки окремі біти, для вибору яких в якості другого операнда використовується код маски.

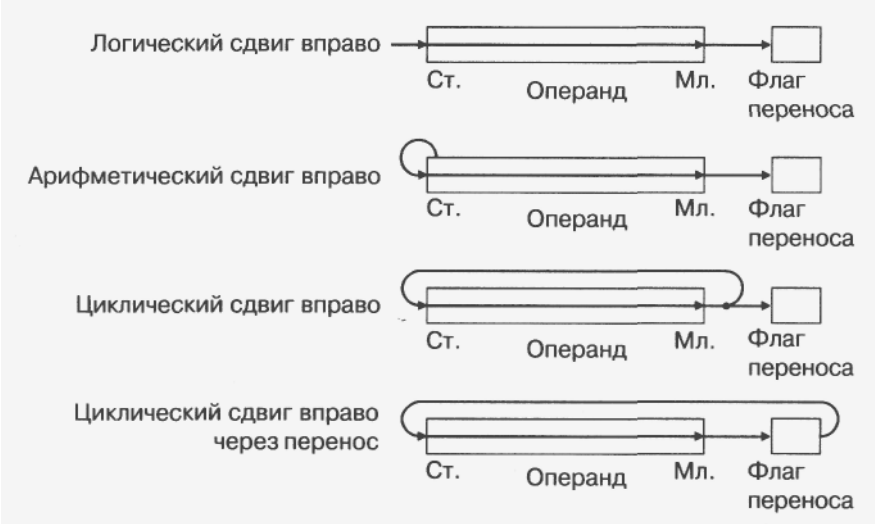

Рис. 8.1. Команди зрушень вправо.

Нарешті, команди установки і очищення бітів регістра стану процесора (тобто прапорів) дозволяють встановити або очистити будь-який прапор, що буває дуже зручно. Кожному прапору зазвичай відповідають дві команди, одна з яких встановлює його в одиницю, а інша скидає в нуль. Наприклад, прапору переносу C (від Carry) будуть відповідати команди CLC (очищення) і SEC або STC (установка).

Команди переходів призначені для організації всіляких циклів, розгалужень, викликів підпрограм і т.д., тобто вони порушують післяпослідовно хід виконання програми. Ці команди записують в регістр-лічильник команд нове значення і тим самим викликають перехід процесора не до наступної по порядку команди, а до будь-якої іншої команди в пам'яті програм. Деякі команди переходів передбачають надалі повернення назад, в точку, з якої був зроблений перехід, інші не передбачають цього. Якщо повернення передбачено, то поточні параметри процесора зберігаються в стеку. Якщо повернення не передбачене, то поточні параметри процесора не зберігаються.

Команди переходів без повернення діляться на дві групи:

• команди безумовних переходів;

• команди умовних переходів.

У позначеннях цих команд використовуються слова Branch (розгалуження) і Jump (стрибок).

Команди безумовних переходів викликають перехід в нову адресу незалежно ні від чого. Вони можуть викликати перехід на зазначену величину зсуву (вперед або назад) або ж на вказану адресу пам'яті. Величина зміщення або нове значення адреси вказуються в якості вхідного операнда.

Команди умовних переходів викликають перехід не завжди, а тільки при виконанні заданих умов. В якості таких умов зазвичай виступають

значення прапорів в регістрі стану процесора (PSW). Тобто умовою переходу є результат попередньої операції, яка змінює значення прапорів. Всього таких умов переходу може бути від 4 до 16. Кілька прикладів команд умовних переходів:

- перехід, якщо дорівнює нулю;
- перехід, якщо не дорівнює нулю;
- перехід, якщо є переповнення;
- перехід, якщо немає переповнення;
- перехід, якщо більше нуля;
- перехід, якщо менше або дорівнює нулю.

Якщо умова переходу виконується, то проводиться завантаження в регістр-лічильник команд нового значення. Якщо ж умова переходу не виконується, лічильник команд просто нарощується, і процесор вибирає і виконує наступну по порядку команду.

Спеціально для перевірки умов переходу застосовується команда порівняння (CMP), що передує команді умовного переходу (або навіть декільком командам умовних переходів). Але прапори можуть встановлюватися і будь-якою іншою командою, наприклад командою пересилання даних, будь-якою арифметичною або логічною командою. Відзначимо, що самі команди переходів прапори не змінюють, що якраз і дозволяє ставити кілька команд переходів одну за одною.

Спільне використання декількох команд умовних і безумовних переходів дозволяє процесору виконувати розгалужені алгоритми будь-якої складності. Для прикладу на рис. 8.2 показано розгалуження програми на дві гілки з подальшим з'єднанням, а на рис. 8.3 - розгалуження на три гілки з подальшим з'єднанням.

Команди переходів з подальшим поверненням в точку, з якої був зроблений перехід, застосовуються для виконання підпрограм, тобто допоміжних програм. Ці команди називаються також командами виклику підпрограм (поширена назва - CALL). Використання підпрограм дозволяє спростити структуру основної програми, зробити її більш логічною, гнучкою, легкою для написання і налагодження. У той же час треба враховувати, що широке використання підпрограм, як правило, збільшує час виконання програми.

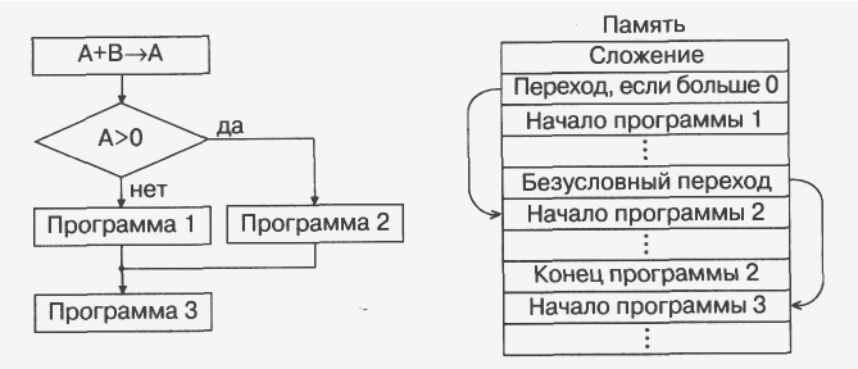

Рис. 8.2. Реалізація розгалуження на дві гілки

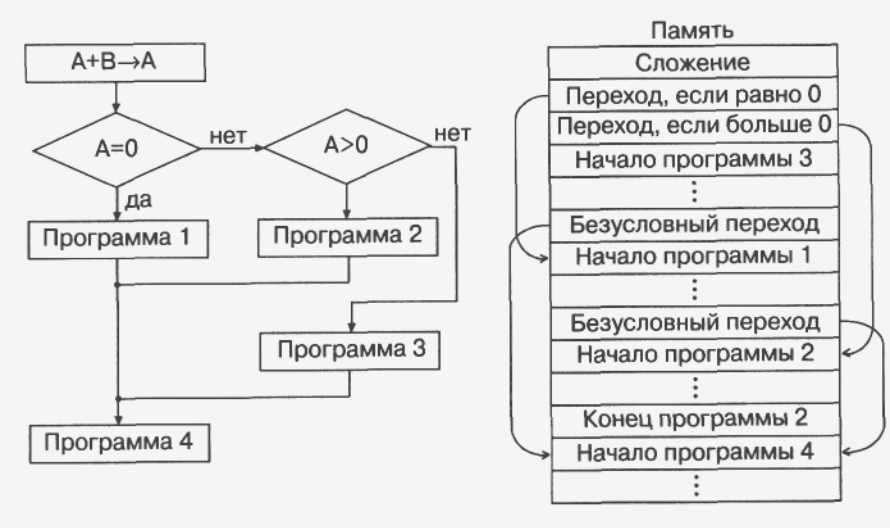

Рис. 8.3. Реалізація розгалуження на три гілки.

Всі команди переходів з поверненням припускають безумовний перехід (вони не перевіряють ніяких прапорів). При цьому вони вимагають одного вхідного операнда, який може вказувати як абсолютне значення нової адреси, так і зміщення, складатися з поточним значенням адреси. Поточне значення лічильника команд (поточна адреса) зберігається перед виконанням переходу в стеці.

Для зворотного повернення в точку виклику підпрограми (точку переходу) використовується спеціальна команда повернення (RET або RTS). Ця команда через залучання зі стека значення адреси команди переходу записує його в регістр-лічильник команд.

Особливе місце серед команд переходу з поверненням займають команди переривань (поширена назва - INT). Ці команди в якості вхідного операнда вимагають номер переривання (адреса вектора). Обслуговування таких переходів здійснюється так само, як і апаратних переривань. Тобто для виконання даного переходу процесор звертається до таблиці векторів переривань і отримує з неї за номером переривання адресу пам'яті, в яку йому необхідно перейти. Адреса виклику переривання і вміст регістра стану процесора (PSW)

зберігаються в стеку. Збереження PSW - важлива відмінність команд переривання від команд переходів з поверненням.

Команди переривань у багатьох випадках виявляються зручніше, ніж звичайні команди переходів з поверненням. Сформувати таблицю векторів переривань можна один раз, а потім вже звертатися до неї в міру необхідності. Номер переривання відповідає номеру підпрограми, тобто номеру функції, виконуваної підпрограмою. Тому команди переривання набагато частіше включаються в системи команд процесорів, ніж звичайні команди переходів з поверненням.

Для повернення з підпрограми, викликаної командою переривання, використовується команда повернення з переривання (IRET або RTI). Ця команда витягує із стека збережене там значення лічильника команд і регістра стану процесора (PSW).

Відзначимо, що у деяких процесорів передбачені також команди умовних переривань, наприклад, команда переривання при переповненні.

Звичайно, в даному розділі ми розглянули тільки основні команди, які найчастіше зустрічаються в процесорах. У конкретних процесорів можуть бути і багато інших команд, що не належать до перерахованих груп команд. Але вивчати їх треба вже після того, як обраний тип процесора, відповідний для завдання, розв'язуваної даної мікропроцесорної системою.

Швидкодія процесора - це одна з найважливіших його характеристик, що визначає ефективність роботи всієї мікропроцесорної системи в цілому. Швидкодія процесора залежить від безлічі факторів, що ускладнює порівняння швидкодії навіть різних процесорів усередині одного сімейства, не кажучи вже про процесорах різних фірм і різного призначення.

Виділимо найважливіші фактори, що впливають на швидкодію процесора.

Перш за все, швидкодія залежить від тактової частоти процесора. Всі операції всередині процесора виконуються синхронно, тактуються єдиним тактовим сигналом. Зрозуміло, що чим більше тактова частота, тим швидше працює процесор, причому, наприклад, дворазове збільшення тактової частоти якогось процесора знижує вдвічі час виконання команд цим процесором.

Однак треба враховувати, що різні процесори виконують однакові команди за різну кількість тактів, причому кількість тактів, затрачених на команду, може змінюватися від одного такту до десятків або навіть сотень. У деяких процесорах за рахунок розпаралелювання мікрооперацій на команду витрачається навіть менше одного такту.

Кількість тактів, що витрачаються на виконання команди, залежить від складності цієї команди і від методів адресації операндів. Наприклад, швидше за все (за менше число тактів) виконуються команди пересилання данних між внутрішніми регістрами процесора. Найповільніше (за велике число тактів) виконуються складні арифметичні команди з плаваючою комою, операнди яких зберігаються в пам'яті.

Спочатку для кількісної оцінки продуктивності процесорів застосовувалася одиниця виміру MIPS (Mega Instruction Per Second), яка відповідала кількості мільйонів виконуваних інструкцій (команд) за секунду. Природно, виробники мікропроцесорів намагалися орієнтуватися на найшвидші команди. Зрозуміло, що подібний показник не дуже вдалий. Для вимірювання продуктивності при виконанні обчислень з плаваючою комою (крапкою) трохи пізніше була запропонована одиниця FLOPS (Floating point Operations Per Second), але вона за визначенням вузькоспеціальна, так як в деяких системах операції з плаваючою комою просто не використовуються.

Інший аналогічний показник швидкодії процесора - час виконання коротких (швидких) операцій. Для прикладу в таблиці 3.1 представлені показники швидкодії декількох 8-розрядних і 16-розрядних процесорів. В даний час цей показник практично не використовується, як і MIPS.

Час виконання команд - важливий, але далеко не єдиний фактор, що визначає швидкодію. Велике значення має також структура системи команд процесора. Наприклад, деяким процесорам для виконання якоїсь операції знадобиться одна команда, а іншим процесорам - кілька команд. Якісь процесори мають систему команд, що дозволяє швидко вирішувати завдання одного типу, а якісь - завдання іншого типу. Важливими є й методи адресації, дозволені в даному процесорі, і наявність сегментування пам'яті, і способи взаємодії процесора з пристроями введення / виводу і т.д.

Суттєво впливає на швидкодію системи в цілому і те, як процесор «спілкується» з пам'яттю команд і пам'яттю даних, чи застосовується суміщення вибірки команд з пам'яті з виконанням раніше обраних команд.

Швидкодія системи в цілому визначається також і розрядністю процесора. Наприклад, 8-розрядний процесор буде повільніше пересилати і обробляти великі масиви даних, ніж 16-розрядний процесор. Так само як 16-розрядний процесор буде значно повільніше працювати з великими числами (більшими, ніж 65536), ніж 32-розрядний процесор.

Табл. 8.1.

Параметри деяких процесорів

Процессор	8085	6800	68000	8086
Фирма	Intel	Motorola	Motorola	Intel
Разрядность	8	8	16	16
Количество команд	80	72	61	133
Тактовая частота, МГц	3	1	8	5
Время выполнения коротких операций, мкс	1,3	2	0,5	0,4

При високій складності вирішуваних завдань швидкодія системи залежить і від загального обсягу системної пам'яті. Адже якщо системної пам'яті мало, системі доводиться зберігати дані в зовнішній пам'яті (наприклад, на магнітному диску), а це дуже сильно (на кілька порядків) сповільнює роботу. Так що розрядність шини адреси процесора теж важлива.

Тому кількісні показники продуктивності процесорів дуже умовні, вони лише побічно характеризують швидкодію системи на базі цього процесора. Проте, деякі виробники пропонують кількісні показники для своїх процесорів, які характеризують час виконання спеціально складених тестових програм, що містять найрізноманітніші команди в тих чи інших співвідношеннях.

Так, для порівняння продуктивності 32-розрядних процесорів фірма Intel, яка виробляє процесори для персональних комп'ютерів, в 1992 році запропонувала свою одиницю виміру iCOMP Index (Intel Comparative Microprocessor Performance). Для обчислення цього показника використовується суміш 16- і 32-бітних цілочисельних команд, команд з плаваючою точкою, команд обробки графіки і відео. В якості базового узятий процесор i486SX-25, чий індекс прийнятий рівним 100. У Таблиці 3.2 наведені індекси iCOMP для деяких процесорів фірми Intel. Як видно з таблиці, за рахунок більш розвиненою архітектури процесори сімейства 486 завжди швидше процесорів сімейства 386, а будь-який Pentium швидше будь-якого процесора з сімейства 486. Тактова частота (вказана в таблиці через рисочку) визначає продуктивність тільки в межах одного сімейства. У 1996 році розробниками Intel був запропонований інший показник - iCOMP Index 2.0, для обчислення якого не використовуються 16-розрядні команди, зате введений мультимедійний тест, а за базу взято Pentium-120, чий індекс прийнятий рівним 100. У таблиці 3.3 представлені ці показники для деяких типів процесорів Intel.

При цьому треба враховувати, що вимірювання проводяться в складі системи, налаштованої на максимальну швидкодію саме даних процесорів, і тільки самою фірмою Intel.

Цінність цих показників і всіх їм подібних не надто велика. Для конкретного комп'ютера і різних процесорів величина показника може надати цілком об'єктивні дані, що дозволяють оцінити, наприклад, доцільність заміни процесора на більш потужний. Але посередність показників iCOMP не дозволяє точно сказати, як буде себе вести процесор в різних завданнях, які орієнтовані на переважне використання різних типів команд.

Табл. 8.2.

Індекси продуктивності iCOMP.

i486SX-25	100	i486DX4-100	435
i386DX-33	56	Pentium-60	510
i486SX-33	136	Pentium-100	815
i486DX2-66	297	Pentium-133	1110

Табл. 8.3.

Індекси продуктивності iCOMP Index 2.0.

Pentium-100	90	Pentium MMX-166	160
Pentium-120	100	Pentium MMX-233	203
Pentium-150	114	Pentium Pro-200	220
Pentium-200	142	Pentium II-266	303

Точна оцінка швидкодії процесора можлива тільки в складі конкретної системи при вирішенні певного завдання. Але всі перераховані тут фактори можна і потрібно враховувати при виборі процесора. А кількісні показники допомагають зробити вибір.