Програмування на Асемблері

Вступ

Викладач: Порєв Віктор Миколайович

Що таке Асемблер?

Асемблер – це мова програмування, яка максимально наближена до програмування у машинних кодах

Вимоги до знань програмістів відповідно рівням мов програмування

Від програміста мов високого рівня **не вимагається** (у залежності від конкретної мови програмування):

- або повного знання архітектури комп'ютера
- або загального розуміння устрою комп'ютера
- або взагалі будь-яких знань (окрім мови) ?

Програміст на Асемблері **повинен знати**:

- архітектуру комп'ютера та його функціонування на апаратному та програмному рівнях
- устрій та роботу процесора, його команди, особливості виконання команд різними процесорами
- організацію пам'яті, внутрішнє представлення даних
- багато чого ще

Питання про машинні коди:

- що повинен знати програміст на машинних кодах?
- хто так програмує і наскільки зараз популярне програмування на цьому рівні?

Для чого потрібен Асемблер?

1. Для **вивчення та розуміння того, як працює комп'ютер** та комп'ютерні програми. Доки комп'ютер є цифровим засобом, заснованим на принципі **програмного керування від команд**, які записуються у пам'ять, як і дані, доти Асемблер буде потрібен.

2. Для створення **швидкодіючих** програм.

3. Для створення **системних** програм.

4. Для створення таких програм, **які не можна** запрограмувати іншою мовою.

5. Для створення програмного забезпечення для особливих процесорів, наприклад, для деяких **мікроконтролерів**.

Основне джерело знань – документація від розробників процесорів. Це є інформацією “від перших рук”. Тільки розробники процесорів знають, як насправді працює процесор. Усі решта можуть лише здогадуватися та фантазувати кожен на свій розсуд.

Порада: якщо Ви хочете чомусь навчитися, обов’язково вивчайте першоджерела. До того ж там і цікавіше та зрозуміліше…

У цьому курсі ми будемо вивчати програмування для **процесорів архітектури Intel x86**. Тому базовим джерелом знань буде:

Intel® 64 and IA-32 Architectures. Software Developer’s Manual. [Електронний ресурс]. – Режим доступу: http://www.intel.com/content/www/us/en/processors/architectures-software-developer-manuals.html

Вигляд титульного
аркушу першого тому
Керівництва

Intel® 64 and IA-32 Architectures Software Developer's Manual

Volume 1: Basic Architecture

NOTE: The *Intel® 64 and IA-32 Architectures Software Developer's Manual* consists of seven volumes: *Basic Architecture*, Order Number 253665; *Instruction Set Reference A-M*, Order Number 253666; *Instruction Set Reference N-Z*, Order Number 253667; *Instruction Set Reference*, Order Number 326018; *System Programming Guide, Part 1*, Order Number 253668; *System Programming Guide, Part 2*, Order Number 253669; *System Programming Guide, Part 3*, Order Number 326019. Refer to all seven volumes when evaluating your design needs.

Література, для бажаючих навчитися на Асемблері

Для нас корисними можуть бути також такі видання:

Microsoft Software Developers Library (MSDN). [Електронний ресурс]. – Режим доступу: http://msdn.microsoft.com
(для бажаючих навчитися програмувати будь-що для Windows – це першоджерело від розробників Windows)

Зубков С. В. Assembler для DOS, Windows и UNIX. - М. : ДМК Пресс ; СПб.: Питер, 2004. - 608 с.
(мабуть, це найкращий російськомовний підручник по Асемблеру)

Рудольф Марек. Ассемблер на примерах. Базовый курс. – СПб.: Наука и техника, 2005. 240 с.
(простий та зрозумілий стиль викладення – для початківців)

та багато ще інших видань …

Ми з вами будемо вивчати такі середовища розробки програм:

1. MASM32

Цей пакет являє собою вільне (freeware) програмне забезпечення для використання, у тому числі для навчальних цілей. Знайти інсталяційні файли можна у мережі Інтернет за електронною адресою http://www.masm32.com

2. Microsoft Visual Studio

Середовище для професійної розробки програмного забезпечення. Підтримує такі мови програмування, як C++, C#, Visual Basic, Асемблер. Можна скачати на сайті microsoft.com

Тема:

Базове середовище виконання програм

Будь-яка програма, яка виконується процесором, отримає набір ресурсів для виконання команд та зберігання коду, даних та інформації про стан. Такі ресурси утворюють **базове середовище виконання програм**.

Базове середовище виконання програм використовується сумісно як прикладними, так і системними програмами.

Базове середовище виконання програм тут розглядається з точки зору програмістів на асемблері, які створюють програми для комп'ютерів на основі процесорів сімейства Intel x86.

На сьогоднішній день найбільш розповсюдженими з цього сімейства є процесори з архітектурою **IA-32** та **Intel 64**. Нижче надані основні відомості, які стосуються саме цих типів архітектури. Відомості взяті з документів фірми Intel.

Режими роботи процесора архітектури IA-32

Архітектура **IA-32** підтримує три основні режими роботи:

- захищений режим;
- режим реальної адресації;
- режим системного керування.

З документації Intel:

3.1 MODES OF OPERATION

The IA-32 architecture supports three basic operating modes: protected mode, real-address mode, and system management mode. The operating mode determines which instructions and architectural features are accessible:

- **Protected mode** — This mode is the native state of the processor. Among the capabilities of protected mode is the ability to directly execute "real-address mode" 8086 software in a protected, multi-tasking environment. This feature is called **virtual-8086 mode**, although it is not actually a processor mode. Virtual-8086 mode is actually a protected mode attribute that can be enabled for any task.
- **Real-address mode** — This mode implements the programming environment of the Intel 8086 processor with extensions (such as the ability to switch to protected or system management mode). The processor is placed in real-address mode following power-up or a reset.
- **System management mode (SMM)** — This mode provides an operating system or executive with a transparent mechanism for implementing platform-specific functions such as power management and system security. The processor enters SMM when the external SMM interrupt pin (SMI#) is activated or an SMI is received from the advanced programmable interrupt controller (APIC).

  In SMM, the processor switches to a separate address space while saving the basic context of the currently running program or task. SMM-specific code may then be executed transparently. Upon returning from SMM, the processor is placed back into its state prior to the system management interrupt. SMM was introduced with the Intel386™ SL and Intel486™ SL processors and became a standard IA-32 feature with the Pentium processor family.

Режими роботи процесора архітектури IA-32

Захищений режим (Protected mode) – цей режим є природнім для процесора. Однією з можливістю цього режиму є здатність безпосередньо працювати з програмами, які розроблені для реального режиму адресації у захищеному багатозадачному середовищі. Така можливість зветься **віртуальним-8086 режимом** (virtual-8086 mode), хоча це не є окремим режимом роботи процесора. Віртуальний-8086 режим насправді є атрибутом захищеного режиму, який може бути використаний для будь-якої відповідної завантаженої програми.

Режими роботи процесора архітектури IA-32

Режим реальної адресації (Real-address mode) – часто звуть просто **реальним режимом**. Цей режим втілює програмне середовище процесора Intel 8086 з деякими розширеннями (такими як можливість переключатися у захищений режим або режим системного керування). Процесор починає своє функціонування після вмикання живлення або після перезавантаження саме у режимі реальної адресації.

Режими роботи процесора архітектури IA-32

Режим системного керування (System Management Mode – SMM). Цей режим надає операційній системі або програмі механізм реалізації керування живленням або системної безпеки. Процесор входить у цей режим після активації зовнішнього переривання SMM interrupt по зовнішньому сигналу (SMI#) або від контролера переривань APIC (Advanced Programmable Interrupt Controller). У режимі SMM процесор вмикає розділений адресний простір для запам'ятовування стану поточної виконуємої програми або задачі. Після повернення з режиму SMM процесор відновлює роботу у тому стані, який був до того.

Режими роботи процесора архітектури Intel 64

3.1.1 Intel® 64 Architecture

Intel 64 architecture adds IA-32e mode. IA-32e mode has two sub-modes.
These are:

- **Compatibility mode (sub-mode of IA-32e mode)** — Compatibility mode permits most legacy 16-bit and 32-bit applications to run without re-compilation under a 64-bit operating system. For brevity, the compatibility sub-mode is referred to as compatibility mode in IA-32 architecture. The execution environment of compatibility mode is the same as described in Section 3.2. Compatibility mode also supports all of the privilege levels that are supported in 64-bit and protected modes. Legacy applications that run in Virtual 8086 mode or use hardware task management will not work in this mode.

  Compatibility mode is enabled by the operating system (OS) on a code segment basis. This means that a single 64-bit OS can support 64-bit applications running in 64-bit mode and support legacy 32-bit applications (not recompiled for 64-bits) running in compatibility mode.

  Compatibility mode is similar to 32-bit protected mode. Applications access only the first 4 GByte of linear-address space. Compatibility mode uses 16-bit and 32-bit address and operand sizes. Like protected mode, this mode allows applications to access physical memory greater than 4 GByte using PAE (Physical Address Extensions).

- **64-bit mode (sub-mode of IA-32e mode)** — This mode enables a 64-bit operating system to run applications written to access 64-bit linear address space. For brevity, the 64-bit sub-mode is referred to as 64-bit mode in IA-32 architecture.

  64-bit mode extends the number of general purpose registers and SIMD extension registers from 8 to 16. General purpose registers are widened to 64 bits. The mode also introduces a new opcode prefix (REX) to access the register extensions. See Section 3.2.1 for a detailed description.

  64-bit mode is enabled by the operating system on a code-segment basis. Its default address size is 64 bits and its default operand size is 32 bits. The default operand size can be overridden on an instruction-by-instruction basis using a REX opcode prefix in conjunction with an operand size override prefix.

  REX prefixes allow a 64-bit operand to be specified when operating in 64-bit mode. By using this mechanism, many existing instructions have been promoted to allow the use of 64-bit registers and 64-bit addresses.

(З документації Intel)

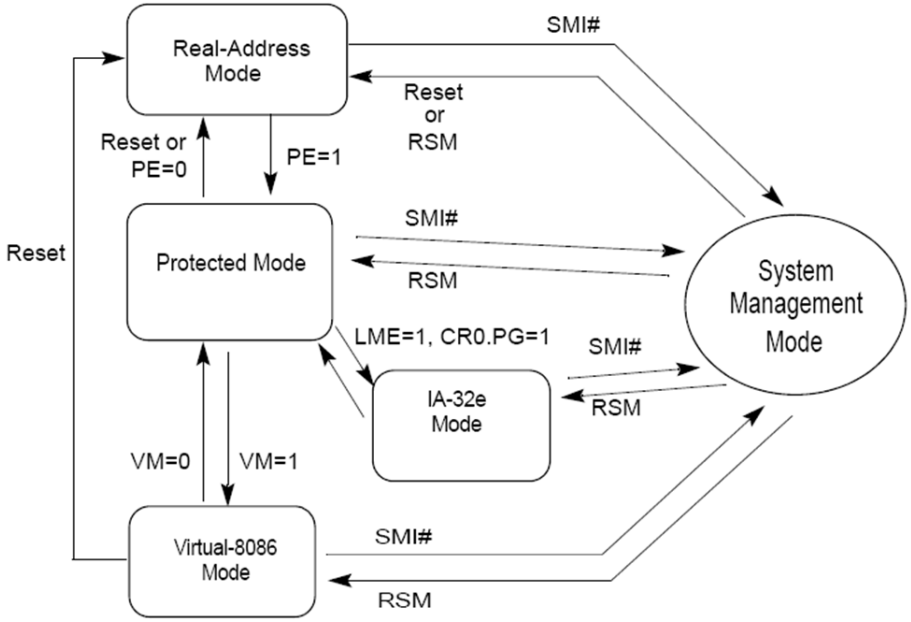
Режими роботи процесора архітектури Intel 64

Процесори архітектури **Intel 64** підтримують вказані вище режими плюс мають ще один режим – режим IA-32e. Режим IA-32e має два різновиди:

Режим сумісності (Compatibility mode – sub-mode of IA-32e mode). Режим сумісності дозволяє виконувати без перекомпіляції 16- та 32-бітні програми у середовищі 64-бітної операційної системи. Цей режим подібний захищеному 32-бітному режиму архітектури IA-32 – програми 32-бітні можуть використовувати лише перші 4ГБ лінійного адресного простору, а при використанні механізму PAE (Physical Address Extensions) – більше 4ГБ.

64-бітний режим (64-bit mode – sub-mode of IA-32e mode). Цей режим дозволяє 64-бітній операційній системі виконувати спеціально написані програми, які отримують доступ до 64-бітного лінійного адресного простору. Крім того, у цьому режимі доступно більше регістрів процесора, а самі регістри вже 64-бітні.

Переходи, перемикання режимів роботи



(З документації Intel)

Основні елементи середовища

Ресурси, які може використати програміст, визначаються:

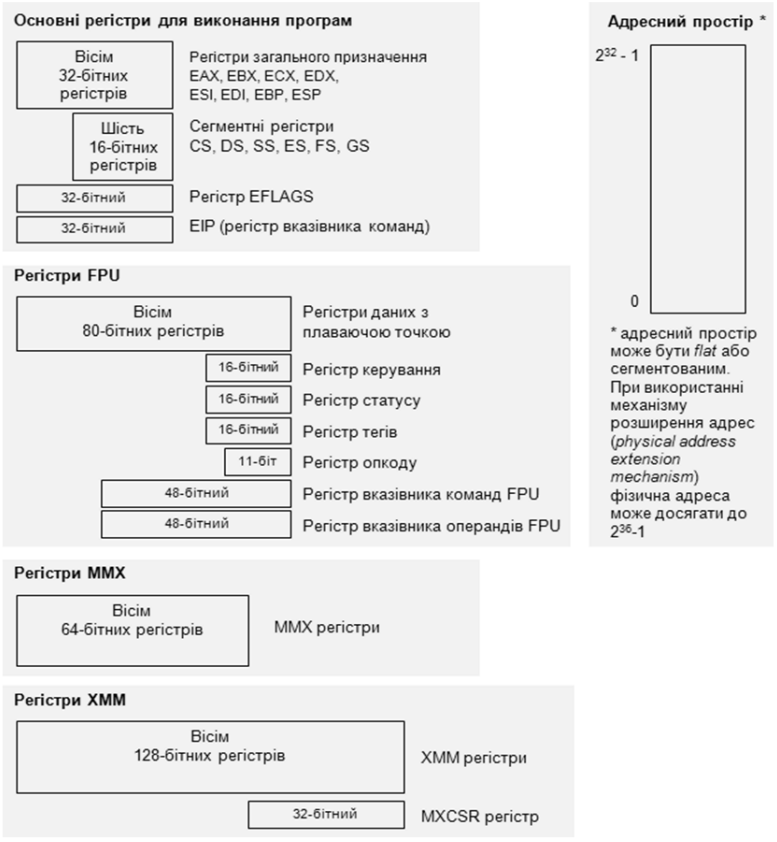
1. Архітектурою процесора

2. Режимом роботи процесора

Елементи середовища, які надає архітектура IA-32

- **Адресний простір –** будь-яка програма, яка виконується на процесорі IA-32 може використовувати лінійний адресний простір до 4 Гбайт (2^{32} байтів) та фізичний адресний простір до 64 Гбайт (2^{36} байтів).
- **Основні регістри** для виконання програм – 8 регістрів загального призначення, 6 сегментних регістрів, регістр EFLAGS, та EIP є основними регістрами для підтримки виконання команд загального призначення. Такі команди виконують арифметичні операції над цілими числами, керують загальною послідовністю виконання програми, обробляють рядки байтів та бітів та адресують пам'ять.
- **Регістри x87 FPU –** 8 регістрів даних FPU, регістр керування FPU, регістр статусу FPU, регістр вказівника команд FPU, регістр вказівника операндів FPU, регістр тегу FPU, та регістр опкоду FPU утворюють середовище для обробки чисел у форматах з плаваючою точкою та інших форматах.
- **MMX регістри –** 8 регістрів підтримують виконання SIMD-команд, якими обробляються набори цілих чисел упакованих у 64-бітові дані.
- **XMM регістри –** 8 регістрів даних та регістр MXCSR підтримують виконання SIMD-команд, якими обробляються набори чисел з плаваючою точкою, а також цілих, упакованих у 128-бітові дані.
- **Регістри, які з'являються у процесі подальшої еволюції** процесорів, наприклад, регістри YMM.

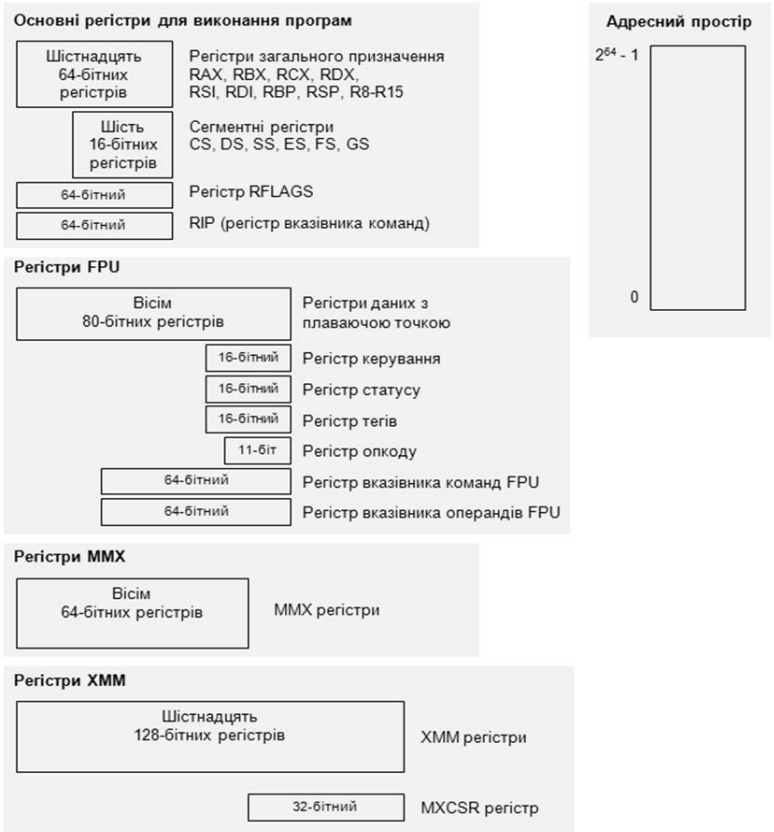
Основні елементи середовища IA-32



На наведеному вище рисунку не відображений такий важливий ресурс для кожної програми, як **стек**. Стек розташовується у пам'яті і використовується для підтримки підпрограм (процедур).

Ресурси, які використовуються системними програмами:
- **порти вводу-виводу** (I/O ports)
- **регістри керування** (control registers). П'ять регістрів (CR0 - CR4) визначають режим процесора і характеристики поточної виконуємої задачі (програми)
- **регістри керування пам'яттю** (memory management registers) – GDTR, IDTR, LDTR, TR (task register) визначають розташування структур даних, які використовуються для керування пам'яттю у захищеному режимі
- **регістри налагодження** (debug registers). Вісім регістрів (DR0 – DR7) для підтримки операцій налагодження
- **регістри діапазону пам'яті** (memory type range registers, MTRRs). Використовуються для виділення блоків пам'яті
- **машинно-специфічні регістри** (machine specific registers, MSRs). Використовуються для контролю та продуктивності
- **регістри контролю помилок** (machine check registers). Для керування, контролю статусу та помилок апаратури
- **лічильники продуктивності** (performance monitoring counters). Для моніторингу продуктивності процесора.

Основні елементи середовища Intel 64



Основні відмінності 64-бітового середовища від 32-бітового:

- Адресний простір – для 64-бітних програм доступний лінійний адресний простір до 2^{64} байтів та фізичний адресний простір у теперішній час до 2^{46} байтів (залежіть від моделі процесора).

- Основні регістри для виконання програм – 16 64-бітових регістрів загального призначення, 6 сегментних регістрів, 64-бітовий регістр RFLAGS, та 64-бітовий регістр вказівника команд RIP.

- XMM регістри – вже 16 регістрів даних для SIMD операцій SSE.

- Стек – розрядність елементів стеку 64 біт, розрядність вказівника стеку – також 64 біт.

Та інші відмінності.