# Лабораторная работа 10. Идентификация процессоров cемейства x86-64

## 10.1 Цель работы

Изучение способов идентификации микропроцессоров, совместимых с архитектурой семейства Intel x86-64; получение практических навыков по разработке низкоуровневых программ, определяющих тип установленного центрального процессора.

## 10.2 Самостоятельная работа студентов

Выполнение работы базируется на знании базовой архитектуры прикладного и системного уровней микропроцессоров семейства x86-610.

С целью получения практических навыков по разработке и выполнению программ, определяющих тип CPU, необходимо самостоятельно составить программу в соответствии с вариантом задания.

Следует знать, что не существует единственного универсального метода идентификации всех микропроцессоров архитектурного семейства x86-610. Поэтому для определения типа любого из процессоров указанного семейства необходимо совместное использование нескольких способов и средств.

### 10.2.1 Идентификация по регистру флагов

Одним из простых способов идентификации является контроль за изменением разрядов в регистре флагов (E)FLAGS ([приложение А](file:///G:\Users\Dima\0501-KI%D0%B7%20-%2013%2014%20%D0%90%D0%9A\2020%20%D0%BE%D1%81%D0%B5%D0%BD%D1%8C\CA_appendix.pdf#%5B%7B%22num%22%3A15%2C%22gen%22%3A0%7D%2C%7B%22name%22%3A%22XYZ%22%7D%2C54%2C790%2C0%5D)). При этом используются следующие биты указанного регистра: ID, VIP и VIF, которые присутствуют в регистре флагов процессоров, начиная с процессора Pentium (P5); AC (i486); VM, RF (i386); NT, поле IOPL (i286).

Для идентификации 16-битных процессоров анализируют значение разрядов 12-15 регистра флагов после попыток его модификации:

* у процессоров 8086/88 они всегда установлены, и попытка сбросить их не удается;
* у процессоров i286 в реальном режиме они всегда сброшены.

Для идентификации 32-битных процессоров анализируется содержимое EFLAGS:

* биты 12-15:
  1. в реальном режиме бит 15 сброшен всегда, биты 12-14 хранят последнее загруженное в них значение;
  2. в защищенном режиме бит 15 сброшен всегда, бит 14 хранит последнее в него загруженное значение; биты 12, 13 можно изменить только при IOPL=0;
* бит 18 определен у процессоров, начиная с i486, поэтому в i386 его невозможно изменить;
* бит 21 определяет возможность использования инструкции CPUID; признаком доступности инструкции является возможность программного изменения его значения.

Для обращения к регистрам FLAGS и EFLAGS используются команды PUSHF/POPF и PUSHFD/POPFD, соответственно. Фрагмент программы, определяющей тип процессора 8086/8088 по регистру флагов приведен в [примере 10.3.1](file:///G:\Users\Dima\0501-KI%D0%B7%20-%2013%2014%20%D0%90%D0%9A\2020%20%D0%BE%D1%81%D0%B5%D0%BD%D1%8C\Lab_10\index.html#p4_3_1).

### 10.2.2 Сигнатура идентификации CPU

BIOS может получить информацию о типе 32-разрядного процессора в виде сигнатуры идентификации, которая формируется в регистре EDX непосредственно после сигнала сброса процессора (рис. 10.1). Следует обратить внимание на некоторые отличия сигнатуры современных микропроцессоров от i386.



а)



б)

Рисунок 10.1 – Сигнатура идентификации процессора:  
а) в регистре EDX процессора i386 после его сброса;  
б) в регистре EAX процессоров i486+ после выполнения CPUID с EAX = 1

Поле "Extended family" (биты 20-27) используется совместно с "Family code" (биты 8-11) для обозначения принадлежности процессора семействам i386, i486, Pentium, Pentium Pro или Pentium 10. Семейство P6 включает все процессоры, основанные на микроархитектуре Pentium Pro, имеет значение поля "Extended family" равное 00h и значение поля "Family code" равное 06h. Семейство Pentium 4 включает все процессоры, основанные на микроархитектуре Intel NetBurst, имеет значение поля "Extended family" равное 00h, значение поля "Family code" равное 0Fh.

Поле "Extended model" (биты 16-19) используется совместно с полем "Model number" (биты 4-7) для обозначения модели процессора в рамках семейства.

Поле "Stepping ID" (биты 0-3) обозначает модификацию (ревизию) конкретной модели.

Поле "Type" (биты 12 и 13) может обозначать, что процессор является оригинальным OEM-процессором (00), OverDrive-процессором (01) или процессором, который может быть использован в двухпроцессорных системах (10).

Возможность получения сигнатуры процессора в любое время без необходимости сброса присутствует в процессорах, поддерживающих команду CPUID (см. [10.2.3](file:///G:\Users\Dima\0501-KI%D0%B7%20-%2013%2014%20%D0%90%D0%9A\2020%20%D0%BE%D1%81%D0%B5%D0%BD%D1%8C\Lab_10\index.html#p4_2_3)). Такая поддержка была реализована, начиная с поздних версий процессоров i486.

### 10.2.3 Команда CPUID

Команда CPUID предоставляет программному обеспечению подробную информацию о процессоре, на котором она выполнена. Ее функция определяется содержимым регистра EAX в момент вызова. Все множество поддерживаемых функций разделено на два подмножества: функции первого и второго подмножеств возвращают, соответственно, базовую и расширенную информацию о процессоре.

В табл. 10.1 показаны входные и выходные параметры нескольких базовых функций команды CPUID.

Таблица 10.1 – Параметры команды CPUID

|  |  |
| --- | --- |
| **Входной параметр** | **Выходной параметр** |
| EAX = 0 | EAX = max – наивысшее значение входного параметра, воспринимаемое командой CPUID |
| EBX:EDX:ECX – строка идентификации производителя |
| EAX = 1 | EAX – сигнатура идентификации процессора (рис. 10.1) |
| ECX:EDX – флаги свойств ([таблицы Г.1, Г.2](file:///G:\\Users\\Dima\\0501-KI%D0%B7%20-%2013%2014%20%D0%90%D0%9A\\2020%20%D0%BE%D1%81%D0%B5%D0%BD%D1%8C\\CA_appendix.pdf" \l "%5B%7B%22num%22%3A98%2C%22gen%22%3A0%7D%2C%7B%22name%22%3A%22XYZ%22%7D%2C68%2C790%2C0%5D" \t "_blank)) |
| EBX – дополнительная информация (рис. 10.2) |
| EAX = 2 | EAX, EBX, ECX, EDX – параметры конфигурации процессора |
| . . . | . . . |
| EAX > max | EAX, EBX, ECX, EDX не определены (не используются) |

Для того, чтобы определить максимально допустимое значение входного параметра базовых функций команды CPUID, необходимо загрузить в регистр EAX значение 0, а затем выполнить команду CPUID:

mov EAX, 0

cpuid

После такого вызова команды CPUID регистр EAX будет содержать наивысшее значение входного параметра базовых функций. Кроме того, будет возвращена текстовая строка идентификации производителя ("Vendor ID"), состоящая из трех частей по четыре символа (в регистре EBX первые четыре ASCII-кода символа, в EDX – следующие, в ECX – последние).

Выполнение команды CPUID со значением EAX = 1 позволяет получить сигнатуру идентификации CPU, которая возвращается в регистре EAX.

Помимо сигнатуры, будет предоставлена также некоторая дополнительная информация в регистре EBX:

* поле "Brand ID" (биты 0-7) – значение индекса элемента таблицы, содержащей текстовые строки брендов ("Brand strings") процессоров IA-32; нулевое значение поля указывает на отсутствие поддержки таблицы брендов;
* поле "Chunks" (биты 8-15) – значение размера строки кэша (в 8-байтных блоках) flushed посредством команды CLFLUSH;
* поле "Count" (биты 16-23) – максимальное количество логических процессоров в одном модуле; для процессоров, поддерживающих технологию Multithreading ([приложение Г](file:///G:\\Users\\Dima\\0501-KI%D0%B7%20-%2013%2014%20%D0%90%D0%9A\\2020%20%D0%BE%D1%81%D0%B5%D0%BD%D1%8C\\CA_appendix.pdf" \l "%5B%7B%22num%22%3A98%2C%22gen%22%3A0%7D%2C%7B%22name%22%3A%22XYZ%22%7D%2C68%2C790%2C0%5D" \t "_blank)) – в одном физическом корпусе;
* поле "APIC ID" (биты 24-31) – 8-битный идентификатор, назначенный локальному модулю APIC процессора при включении питания.



Рисунок 10.2 – Содержимое регистра EBX после выполнения команды CPUID с EAX = 1

Кроме того, в регистрах EDX и ECX будут расположены флаги свойств ("Feature flags", [таблицы Г.1, Г.2](file:///G:\Users\Dima\0501-KI%D0%B7%20-%2013%2014%20%D0%90%D0%9A\2020%20%D0%BE%D1%81%D0%B5%D0%BD%D1%8C\CA_appendix.pdf#%5B%7B%22num%22%3A98%2C%22gen%22%3A0%7D%2C%7B%22name%22%3A%22XYZ%22%7D%2C68%2C790%2C0%5D)). Единичные значения разрядов означают поддержку процессором соотвествующих свойств. Используя данную информацию, программное обеспечение может выявить и предотвратить возможные несовместимости с различными свойствами процессора, добавляемыми или удаляемыми производителями.

### 10.2.3 Прочие способы идентификации

По аналогии с методом идентификации по регистру флагов процессор можно идентифицировать по выполнению команд, поддерживаемых им (или процессором выше). При выполнении этих команд на процессорах ниже предполагаемого возникает особый случай – недействительный код операции (прерывание 6). Используемые с указанной целью команды:

cmpxchg8b ; (Pentium+)

xadd DX, DX ; (i486+)

mov EDX, CR0 ; (i386+)

smsw DX ; (i286)

shl DX, 5 ; (80186).

Идентифицировать процессор также возможно, выполнив на нем определенное количество команд и сравнив время их выполнения с табличным значением. При этом помимо типа процессора можно определить и его тактовую частоту. Однако данный способ не всегда дает верный тип процессора, который был «разогнан».

Для идентификации специфических процессоров 8086/88 фирмы NEC, различных DLC и SLC фирмы СYRIX и т.п. существуют определенные библиотеки программ, а также описания, предоставляемые производителями данных процессоров.

## 10.3 Примеры выполнения практических заданий

### 10.3.1 Определение процессора 8086/8088 по регистру флагов

; фрагмент ассемблерной вставки:

pushf ; сохранить копию р-ра флагов

pop BX ; восстановить в BX

mov AX,0FFFh ; очистить биты 12-15 в AX

and AX,BX

push AX ; сохранить новое значение флагов

popf

pushf ; установить регистр флагов

pop AX ; сохранить копию нового р-ра флагов в AX

mov CX,0F000h

and AX,CX ; если биты 12-15 установлены,

cmp AX,CX ; это процессор 8086/88

jne non\_8086 ; переход, если не 8086/88

jmp cputype\_8086; печать типа процессора 8086/88

non\_8086: mov AX, 0 ; это не 8086/88 процессор

cputype\_8086: mov AX, 1 ; это 8086/88 процессор

## 10.4 Порядок выполнения работы

Выполнение работы состоит в разработке, отладке и выполнении ассемблерных программ для идентификации CPU в среде разработки Visual C++. Основные этапы работы в среде Visual C++ рассмотрены в [лабораторной работе 8 (п. 8.4)](file:///G:\Users\Dima\0501-KI%D0%B7%20-%2013%2014%20%D0%90%D0%9A\2020%20%D0%BE%D1%81%D0%B5%D0%BD%D1%8C\Lab_2\index.html#p2_4).

1. Используя регистр флагов, необходимо убедиться в наличии 32-разрядного процессора в системе.
2. Убедиться в поддержке команды CPUID (посредством бита ID регистра EFLAGS) и определить максимальное значение параметра ее вызова.
3. Получить строку идентификации производителя процессора и сохранить ее в памяти.
4. Получить сигнатуру процессора и определить его модель, семейство и т.п. Выполнить анализ дополнительной информации о процессоре.
5. Получить флаги свойств. Составить список поддерживаемых процессором свойств.

## 10.5 Содержание отчета

Отчет о лабораторной работе должен содержать:

* цель работы;
* краткое описание теоретических основ;
* постановку задачи;
* экспериментально-практическую часть (включая анализ результатов);
* выводы по работе.

В экспериментально-практической части необходимо привести исходные тексты разработанных программ (на языке C++ с ассемблерными вставками) и результаты их выполнения, полученные в регистрах CPU и (или) ячейках памяти.

## 10.7 Контрольные вопросы

1. Какие существуют способы идентификации типа процессора? Назовите их достоинства и недостатки.
2. Перечислите биты регистра флагов и укажите их назначение.
3. Каким образом можно идентифицировать процессоры 8088 и 8086?
4. Что можно определить по степпингу и семейству процессора?
5. Как можно использовать BIOS и команду CPUID при идентификации процессора?
6. Что представляют собой флаги свойств?
7. В чем заключается отличие в назначении полей "Vendor ID" и "Brand ID"?