Лабораторная работа № 4
Использование инструкции CPUID для определения функциональных возможностей CPU

Цель работы: познакомиться с базовыми принципами определения вычислительных возможностей и аппаратной конфигурации CPU.

Современные CPU семейства x86 прошли долгий эволюционный путь, начиная от процессоров Intel i4004 (1971 г.) и Intel 8086 (1978 г.) и заканчивая современными процессорами семейств Intel Core (рынок настольных компьютеров, на середину 2026 г. последним является 15 поколение), Intel Xeon (рынок серверов и рабочих станций) и AMD Ryzen (на середину 2026 г. последним является Zen 5) и EPIC. Внутренняя организация современных CPU является в достаточной степени сложной и во многом делает их похожими на системы на чипе (SoC) с такими структурными элементами как:

- поддержка различных специализированных сопроцессоров и акселераторов, физически реализованных в составе процессора (FPU, TPU/NPU и т.д.);
- разрядность (4-, 8-, 16-, 32- или 64-битные CPU), число и размер регистров общего назначения, объем адресуемой физической/виртуальной памяти;
- поддержка различных векторных расширений системы команд (MMX, SSEx, AVXx, FMAx);
- поддержка дополнительных расширений системы команд (AES, SHA, GFNI, APX и т.д.);
- поддержка матричных расширений системы команд (AMX);
- поддержка многоядерности (в соответствии с моделью SMP или NUMA), число физических и логических ядер процессора (технологии Hyper-Threading или SMT);
- поддержка кэш-памяти с различной организацией (уровни кэшей от L1i/L1d, L2, L3 до L0 и L4 в некоторых CPU различного объема с различной латентностью обращения и числом портов чтения/записи; инклюзивная/эксклюзивная организация кэшей; полно- или наборно-ассоциативная организация кэшей с различной степенью ассоциативности (англ. n-way); наличие HBM RAM памяти, интегрированной на кристалл);
- наличие и организация TLB буферов (число записей, ассоциативность);

Понимание внутренней организации CPU необходимо для разработки эффективных параллельных программ, поддерживающих наиболее продвинутые версии функциональных возможностей, предоставляемых процессором.

Для определения состава реализованных функций начиная с процессора Intel i486 (1989 г.) в состав системы команд CPU была включена ассемблерная команда CPUID. Логика ее работы сводится к помещению аргумента данной команды в регистр EAX, вызова команды CPUID, которая возвращает результат своей работы в регистрах EAX, EBX, ECX, EDX (т.н. leaf, лист). В зависимости от начального значения аргумента в регистре EAX результат работы команды CPUID будет различным на различных процессорах. Перечисление полного перечня возможностей команды CPUID можно найти в справочной документации для различных процессоров, с выходом новых поколений процессоров ее функционал расширяется, в рамках выполнения данной работы ограничимся рассмотрением лишь основных функций, имеющих отношение к разработке параллельных программ.

Перед использованием команды CPUID вообще говоря необходимо удостовериться в том, что процессор ее поддерживает, для чего необходимо попытаться изменить 21-й бит в регистре флагов EFLAGS. Удачное изменение значения данного бита сигнализирует о поддержке процессором команды CPUID:

```
PUSHFD              // Помещение регистра EFLAGS в стек
POP     EAX         // Извлечение значения EFLAGS из стека в регистр EAX
MOV     EBX, EAX    // Пересылка значения в регистр EBX
XOR     EAX, 200000h // Инверсия 21-го бита
PUSH    EAX         // Помещение нового значения в стек
POPFD               // Извлечение нового значения в регистр EFLAGS
PUSHFD              // Повторное помещение регистра EFLAGS в стек
POP     EAX         // Извлечение значения регистра EFLAGS в регистр EAX
XOR     EAX, EBX    // Проверка изменения 21-го бита
JE      no_cpuid    // Если он не изменился, CPUID не поддерживается
```

В современных процессорах старше Intel i486 данная команда поддерживается всеми процессорами без исключения, поэтому проведение подобной проверки можно опустить.

Первым шагом при использовании команды CPUID является ее вызов с нулевым значением регистра EAX (сокращенно CPUID EAX=0). При этом в регистре EAX будет возвращено максимально допустимое значение параметра инструкции CPUID, поддерживаемое данным процессором, а в регистрах EBX, ECX, EDX – сокращенное наименование производителя процессора (см. табл. 1):

```
XOR     EAX, EAX    // EAX := 0
CPUID
MOV     [CpuInfo.NumberOfBasicFunctions], EAX
MOV     DWORD PTR [CpuInfo.Vendor[0]], EBX
MOV     DWORD PTR [CpuInfo.Vendor[4]], EDX
MOV     DWORD PTR [CpuInfo.Vendor[8]], ECX
```

Таблица 1

Пример наименования производителей процессоров для CPUID EAX=1

Производитель	ASCII-строка	16-ричное представление (EBX:ECX:EDX)
Intel	"GenuineIntel"	756E6547:49656E69:6C65746E
AMD	"AuthenticAMD"	68747541:69746E65:444D4163
Cyrix	"CyrixInstead"	69727943:736E4978:64616574
МЦСТ	"E2K MACHINE "	204B3245:4843414D:20454E49

Вместо использования ассмеблерного кода в программах допускается вызов intrinsic-версий ассемблерной команды CPUID с аналогичным функционалом:

```
void __cpuid(
   int cpuInfo[4],
   int function_id
);

void __cpuidex(
   int cpuInfo[4],
   int function_id,
   int subfunction_id
);
```

Значение основной функции (регистр EAX) передается в параметре `function_id`, значение дополнительной функции (при необходимости, регистр ECX, т.н. subleaf, подлист) – в параметре `subfunction_id`, результирующие значения регистров EAX, EBX, ECX и EDX после выполнения команды CPUID возвращаются в параметре `cpuInfo[4]`. Для использования данных функций необходимо подключение заголовочного файла `intrin.h`.

Далее команда CPUID вызывается с различными значениями в регистре EAX (и в регистре ECX при необходимости) для детального выяснения состава функций, поддерживаемых процессором. Ниже приведен перечень наиболее интересных функций, востребованных с точки зрения параллельного программирования.

Функция CPUID EAX=1 – подробная информация о версии процессора.

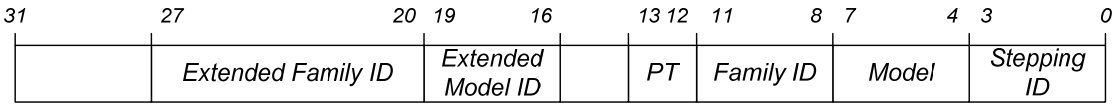

Рис. 1. Содержимое регистра EAX (PT – Processor Type)

Значение в регистре EAX однозначно идентифицирует процессор, на котором производился запуск программы (например, значение 306C3h соответствует процессору Intel Core i7 4770). Его можно трактовать целиком, а можно в виде отдельных полей:

Stepping ID = 3h
Model = Ch
Family = 6h
Processor type = 0h
Extended model = 3h
Extended family = 0h

Для доступа к выбранному биту удобно использовать обращение вроде

```
bool some_bit_value = _eax & (1 << bit_num) != 0;
```

а для доступа к выбранному полю:

```
unsigned int some_field = (_eax >> lower_bit) & field_size;
```

Значения в регистрах EBX, ECX и EDX представляют собой комбинацию битовых флагов, установка каждого из которых свидетельствует о поддержке одного из расширений системы команд. Наиболее интересные флаги и поля, имеющие отношение к разработке параллельных программ, приведены ниже.

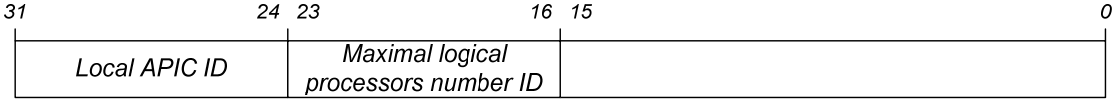

Рис. 2. Содержимое регистра EBX

Maximal logical processors number ID – максимальное значение идентификатора числа логических процессоров в составе одного физического (ранее в процессоре Pentium 4 при поддержке технологии Hyper-Threading/SMT равно 2, иначе 1, для ускорителей Intel Xeon Phi могло иметь большее значение).

Local APIC ID – идентификатор логического процессора, на котором в настоящее время производится выполнение текущего потока программного кода.

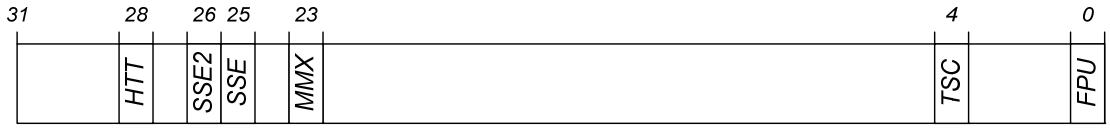

Рис. 3. Содержимое регистра EDX

FPU – наличие арифметического сопроцессора в составе процессора
TSC – наличие счетчика Time Stamp Counter (TSC) и поддержка команды RDTSC
MMX – поддержка технологии MultiMedia Extensions
SSE – поддержка технологии Streamed SIMD Extensions
SSE2 – поддержка технологии Streamed SIMD Extensions 2
HTT – поддержка технологии Hyper-Threading

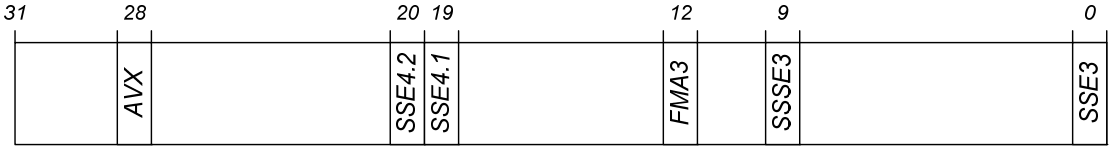

Рис. 4. Содержимое регистра ECX

SSE3 – поддержка технологии Streamed SIMD Extensions 3
SSSE3 – поддержка технологии Supplemental SSE 3
FMA3 – поддержка технологии Fused Multiply and Add с трехоперандными командами
SSE4.1 – поддержка технологии Streamed SIMD Extensions 4.1
SSE4.2 – поддержка технологии Streamed SIMD Extensions 4.2
AVX – поддержка технологии Advanced Vector Extensions

Функция CPUID EAX=4 – определение параметров и способа организации подсистемы кэш-памяти процессоров Intel (для процессоров AMD вместо EAX=4 необходимо использовать значение EAX=8000001Dh). В качестве дополнительного параметра в регистре ECX передается 0, 1, 2, … до тех пор, пока в регистре EAX не будет возвращено нулевое значение.

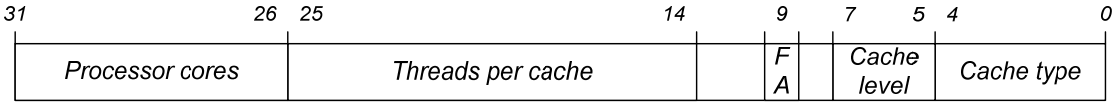

Рис. 5. Содержимое регистра EAX

Cache type – типа кэша (1 – кэш данных, 2 – кэш команд, 3 – кэш общего назначения)
Cache level – уровень кэша
FA (Fully Associative) – признак полностью ассоциативного кэша
Threads per cache – число ядер/потоков, разделяющих данный кэш (к значению поля необходимо добавить +1)
Processor cores – общее число ядер процессора (можно использовать данное значение однократно для ECX=0)

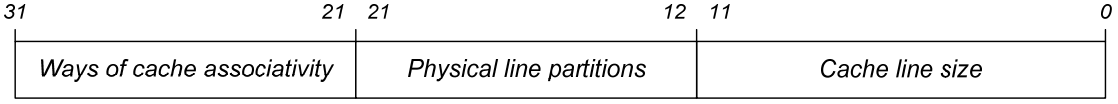

Рис. 6. Содержимое регистра EBX

Cache line size – размер кэш линии в байтах (к значению поля необходимо добавить +1)
Physical line partitions – число кэш линий, которые могут совмещать один и тот же тэг (к значению поля необходимо добавить +1)
Ways of cache associativity – степень ассоциативности для наборно-ассоциативного кэша (к значению поля необходимо добавить +1)

Содержимое регистра ECX определяет число наборов (англ. sets) для наборно-ассоциативного кэша (к значению поля необходимо добавить +1).

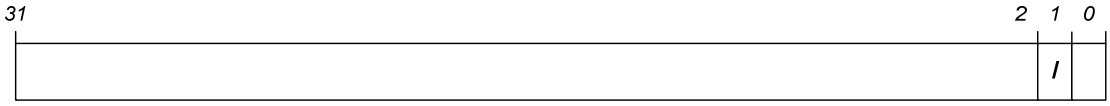

Рис. 7. Содержимое регистра EDX

I (Inclusiveness) – инклюзивная (=1)/эксклюзивная (=0) организация кэша

Исходя из приведенных выше данных, объем кэша может быть определен как

$$Cache\ size = Cache\ line\ size \times Physical\ line\ partitions \times Ways\ of\ cache\ associativity \times \times Sets\ number$$

Например, для процессора Intel Core i7 4770 параметры организации подсистемы кэш-памяти выглядят следующим образом:

ECX=0: 1C004121:01C0003F:0000003F:00000000 (значения регистров EAX:EBX:ECX:EDX в 16-ричном виде)
Data cache
Cache level = 1
Threads per cache = 2
Cache line size = 64 Bytes
Number of cache lines that share a cache address tag = 1
Ways of cache associativity = 8
Exclusive
Number of sets = 64
Cache size = 32 KB

ECX=1: 1C004122:01C0003F:0000003F:00000000
Instruction cache
Cache level = 1
Threads per cache = 2
Cache line size = 64 Bytes
Number of cache lines that share a cache address tag = 1
Ways of cache associativity = 8
Exclusive
Number of sets = 64
Cache size = 32 KB

ECX=2: 1C004143:01C0003F:000001FF:00000000

Unified cache
Cache level = 2
Threads per cache = 2
Cache line size = 64 Bytes
Number of cache lines that share a cache address tag = 1
Ways of cache associativity = 8
Exclusive
Number of sets = 512
Cache size = 256 KB

ECX=3: 1C03C163:03C0003F:00001FFF:00000006
Unified cache
Cache level = 3
Threads per cache = 8 (на самом деле процессор возвращает некорректное значение 16)
Cache line size = 64 Bytes
Number of cache lines that share a cache address tag = 1
Ways of cache associativity = 16
Inclusive
Number of sets = 8192
Cache size = 8192 KB

Функция CPUID EAX=7, ECX=0 – определение состава поддерживаемых расширений системы команд процессора. В регистре EAX возвращается максимальное значение для номера подфункции, передаваемого в регистре ECX. Регистры EBX, ECX и EDX содержат битовые флаги поддерживаемых расширений.

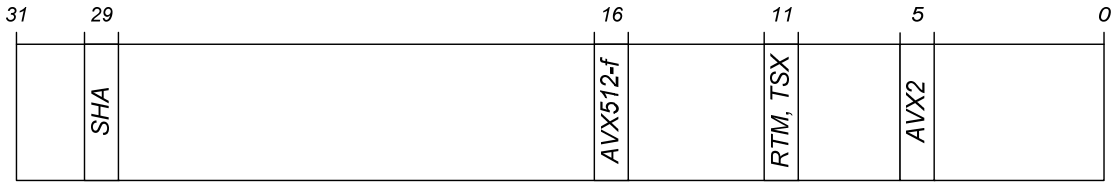

Рис. 8. Содержимое регистра EBX

AVX2 – поддержка технологии Advanced Vector Extensions 2
RTM (TSX) – поддержка Restricted Transactional Memory (RTM) и Transactional Synchronization Extensions (TSX) на ее базе
AVX512-f – поддержка технологии Advanced Vector Extensions 512 (базовый набор команд)
SHA – поддержка технологии Secure Hash Algorithm

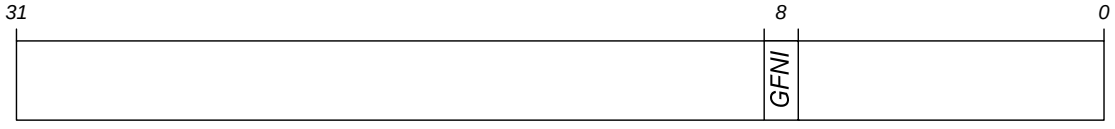

Рис. 9. Содержимое регистра ECX

GFNI – поддержка технологии Galua Field New Instructions

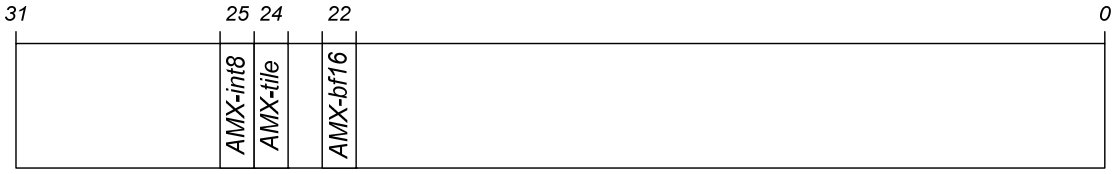

Рис. 10. Содержимое регистра EDX

AMX-x – поддержка различных команд в рамках технологии Advanced Matrix Extensions

Функция CPUID EAX=7, ECX=1 – определение состава поддерживаемых расширений системы команд процессора.

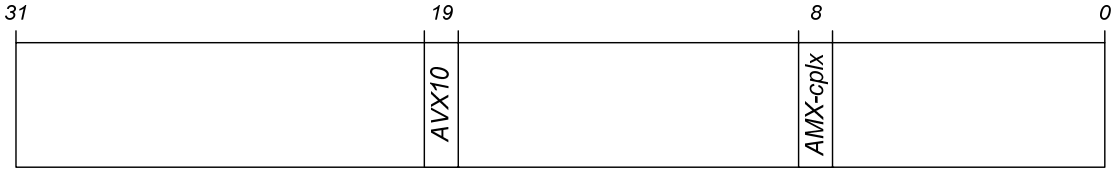

Рис. 11. Содержимое регистра EDX

AMX-cplx – поддержка команд умножения матриц с комплексными коэффициентами в рамках технологии Advanced Matrix Extensions
AVX10 – поддержка совокупности различных команд в рамках технологии Advanced Vector Extensions

Функция CPUID EAX=16h – определение базовой тактовой частоты процессора (результат в регистре EAX), максимальной тактовой частоты процессора в boost-режиме (результат в регистре EBX) и тактовой частоты системной шины (результат в регистре ECX), используются только 16 младших бит, значения представлены в МГц.

Начиная со значения EAX=80000000h процессор выводит информацию в виде ряда расширенных функций.

Функция CPUID EAX=80000000h – возвращает в регистре EAX число поддерживаемых расширенных функций (максимальное значение аргумента в регистре EAX).

Функция CPUID EAX=80000001h – определение состава поддерживаемых расширений системы команд процессора.

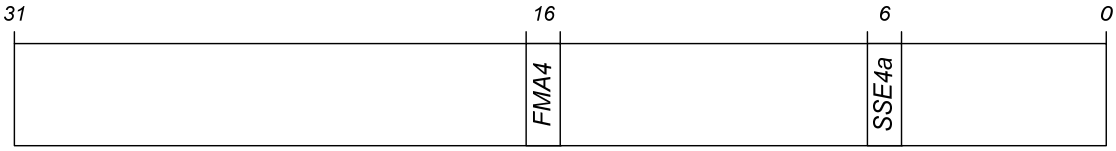

Рис. 12. Содержимое регистра ECX

SSE4a – поддержка расширения Streamed SIMD Extension, версия a
FMA4 – поддержка расширения Fused Multiply and Add с четырехоперандами командами (поддерживаются некоторыми процессорами AMD)

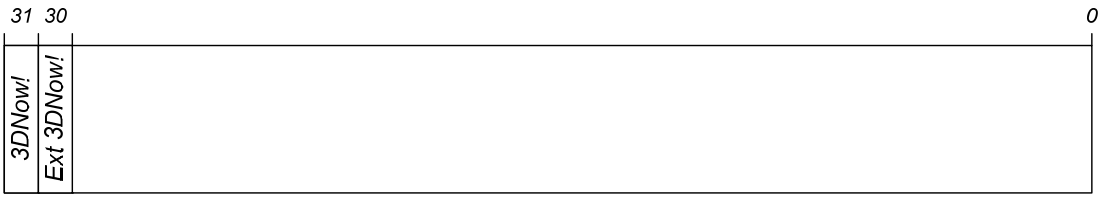

Рис. 13. Содержимое регистра EDX

3DNow! – поддержка технологии 3DNow!
Ext 3DNow! – поддержка технологии Extended 3DNow!
(поддерживаются некоторыми процессорами AMD)

Функции CPUID EAX=80000002h – 80000004h – возвращают название процессора (англ. CPU brand string) в регистрах EAX:EBX:ECX:EDX при последовательном вызове функций (например, «Intel(R) Core(TM) i7-4770 CPU @ 3.40GHz»).

Замечание. Некоторая информация, возвращаемая процессором для некоторых вызываемых функций может быть неточной, некорректной или отсутствовать вовсе (регистры имеют нулевые значения). Некоторые функции являются недокументированными.

Вывод результатов работы программы может быть представлен в следующем виде (см. рис. 14).

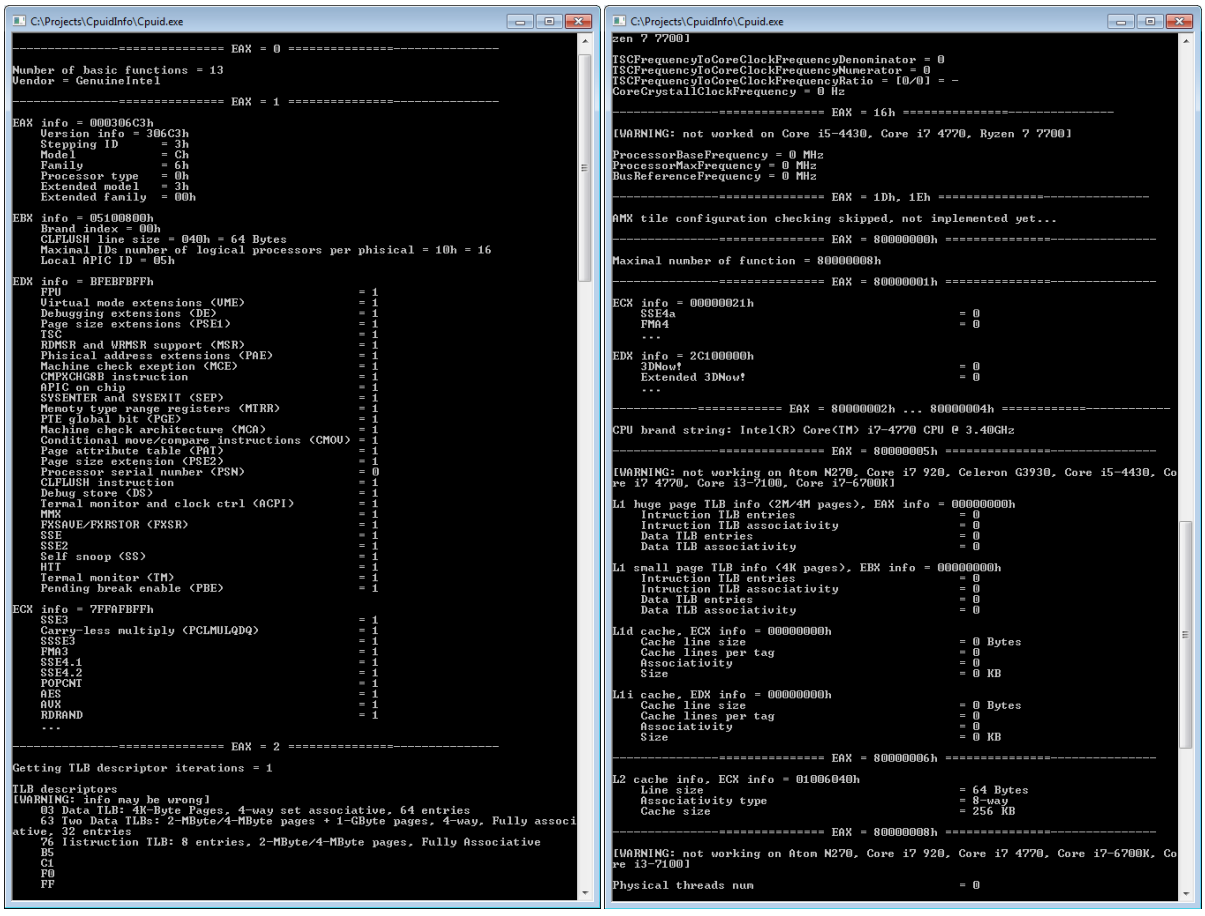

Рис. 14. Пример вывода программы

С примером вывода всей информации можно ознакомиться здесь: http://evatutin.narod.ru/evatutin_cpuid.7z. Из документации к процессорам можно узнать ряд дополнительной информации про поддержку таких технологий как мониторинг температур и управление питанием (EIST, C-states), Intel Key Locker, Trust Domain Execution (TDX), работу гипервизоров и виртуальных машин под их управлением и т.д., что выходит за рамки данного курса.

Задание

Разработать программу для опроса процессора с использованием команды CPUID и вывода полученной информации в удобной для пользователя форме (оконное приложение, консольное приложение с выводом на экран или в файл).

Рекомендуемая литература

1. https://en.wikipedia.org/wiki/CPUID.
2. Intel Processor Identification and the CPUID Instruction (application note 485), last version.
3. Intel 64 and IA-32 Architectures Software Developer's Manual, last version.
4. AMD CPUID specification, last version.
5. AMD64 Architecture Programmer's Manual Volume 3: General-Purpose and System Instructions, last version.