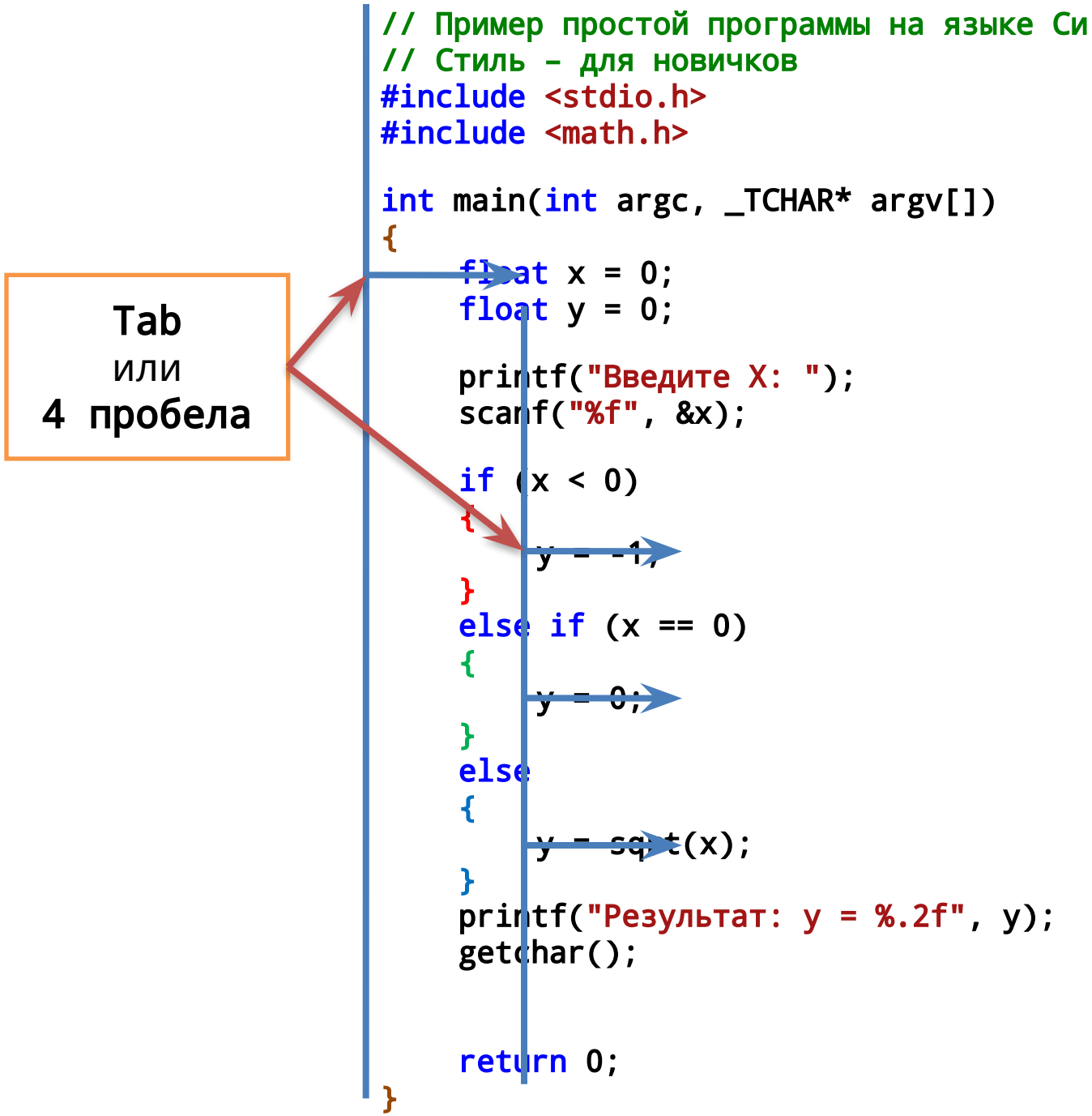
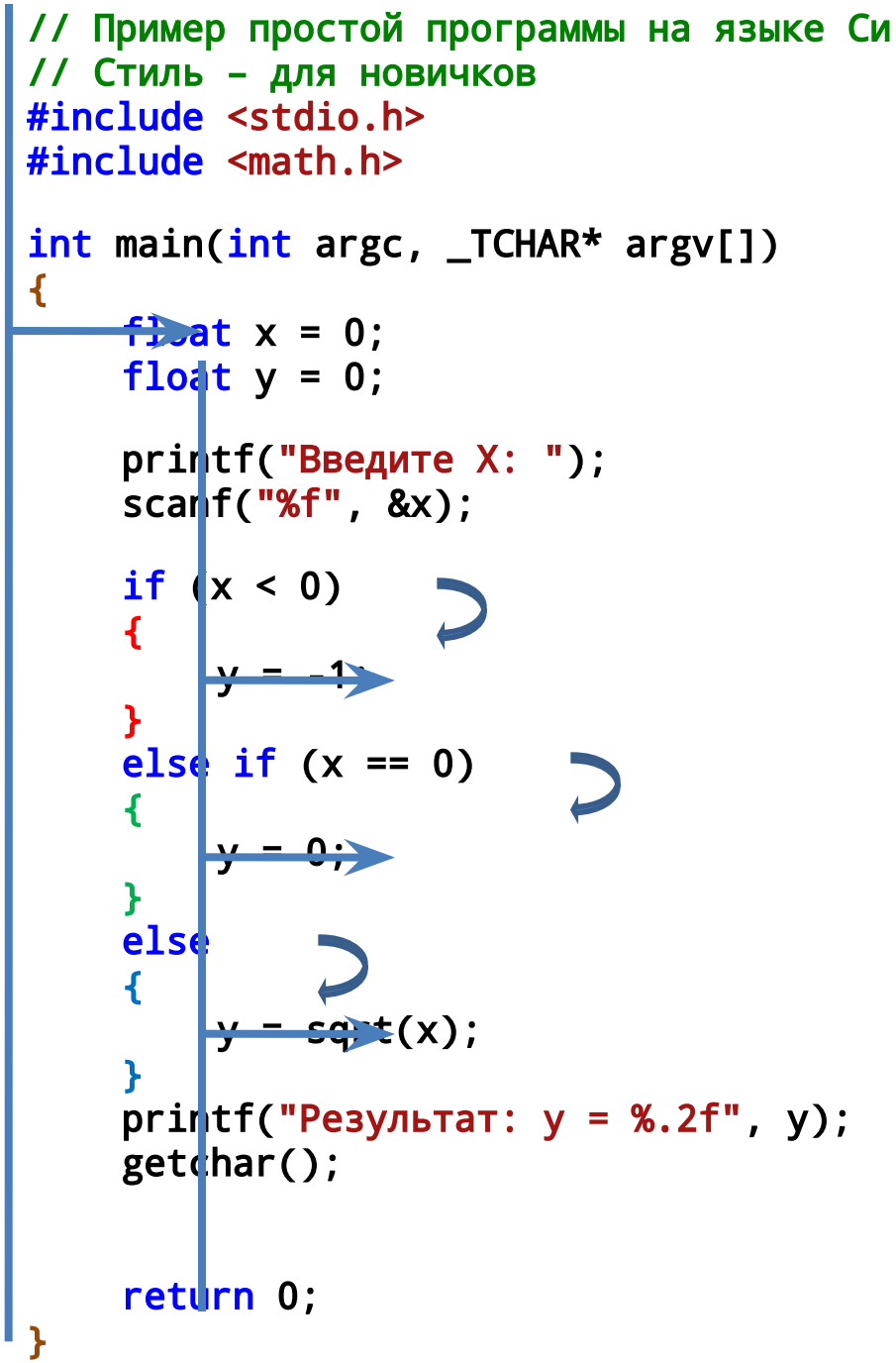
Стиль программирования

Утверждения по поводу стиля

1. Для любого языка существует один распространённый **стиль программирования**, или число таких стилей невелико.
2. Не существует и никогда не будет существовать стиля программирования, который обладал бы очевидными преимуществами по сравнению с любым из стандартных стилей.
3. В любом нетривиальном проекте желательно применять стандартный стиль программирования.
4. Каждый член команды должен писать в

```c
// Пример простой программы на языке Си
// Стиль – для новичков
#include <stdio.h>
#include <math.h>

int main(int argc, _TCHAR* argv[])
{
    float x = 0;
    float y = 0;

    printf("Введите X: ");
    scanf("%f", &x);

    if (x < 0)
    {
        y = -1;
    }
    else if (x == 0)
    {
        y = 0;
    }
    else
    {
        y = sqrt(x);
    }
    printf("Результат: y = %.2f", y);
    getchar();


    return 0;
}
```


Tab
или
4 пробела
// Пример простой программы на языке Си
// Стиль – для новичков
#include <stdio.h>
#include <math.h>
int main(int argc, _TCHAR* argv[])
{
float x = 0;
float y = 0;
printf("Введите X: ");
scanf("%f", &x);
if (x < 0)
{
y = -1;
}
else if (x == 0)
{
y = 0;
}
else
{
y = sqrt(x);
}
printf("Результат: y = %.2f", y);
getchar();
return 0;
}


// Пример простой программы на языке Си
// Стиль – для новичков
#include <stdio.h>
#include <math.h>
int main(int argc, _TCHAR* argv[])
{
float x = 0;
float y = 0;
printf("Введите X: ");
scanf("%f", &x);
if (x < 0)
{
y = -1;
}
else if (x == 0)
{
y = 0;
}
else
{
y = sqrt(x);
}
printf("Результат: y = %.2f", y);
getchar();
return 0;
}

```c
// Пример простой программы на языке Си
// Стиль – для новичков
#include <stdio.h>
#include <math.h>

int main(int argc, _TCHAR* argv[])
{
    float x = 0;
    float y = 0;

    printf("Введите X: ");
    scanf("%f", &x);

    if (x < 0)
    {
        y = -1;
    }
    else if (x == 0)
    {
        y = 0;
    }
    else
    {
        y = sqrt(x);
    }
    printf("Результат: y = %.2f", y);
    getchar();


    return 0;
}
```

Вертикальный «пробел», используется для деления кода на логические блоки

```c
// Пример простой программы на языке Си
// Стиль – эконом, логический блок одной строкой
#include <stdio.h>
#include <math.h>

int main()
{
    float x = 0, y = 0;

    // %f - float, %c - char, * == ignore
    printf("Введите X: "); scanf("%f%*c", &x);

    y = (x < 0) ? -1 : sqrt(x);

    printf("Результат: y = %.2f", y); getchar();

    return 0;
}
```

Пробелы

```
float x = 0;

scanf("%f", &x);

if (x < 0)
{
   y = -1;
}
else if (0 == x)
{
   y = 0;
}

printf("Результат: при x = %d, y = %.2f", x, y);
```

Пробелы в условиях

Пробелы между if, for, while и скобкой:

Пробелы в операторах присваивания и математических выражениях:

Пробелы между параметрами функций

```c
//Пример “профи”-стиля
#include <stdio.h>
#include <math.h>

int main()
{
    float x = 0, y = 0;

    printf("Введите X: ");
    scanf("%f", &x);

    if      (x <  0)  y = -1;
    else if (x == 0)  y = 0;
    else              y = sqrt(x);

     printf("Результат: y = %.2f", y);
     getchar();

     return 0;
}
```

```
//Пример отсутствия стиля программирования

#include                        "stdafx.h"
#include    <stdio.h>
#include  <math.h>

int main(int argc, _TCHAR* argv[]){
float x=0;float y=0;
{ printf("Введите X: "); }

scanf("%f",&x);
if(x<0)
{y=-1;}
else if(x==0)y=0;
   else
{
      y=sqrt(x);}

printf("Результат: y = %.2f",y);
   getchar();                   getchar();
         return 0;
                        }
```

Аппаратное обеспечение
(aka **Hardware**, **железо**)

Представляет собой фундамент (платформу) на котором строится **программное обеспечение**

Любое электронное или механическое устройство (**device**) как в сборке так и по частям можно называть аппаратным обеспечением

Modern hardware types of computer

- Калькулятор

- **Персональный компьютер** (ПК, Personal Computer, PC)
  - **Настольный компьютер** (Desktop PC)
  - **Ноутбук** (Notebook, Лэптоп, Mobile PC)
    - **Нетбук** (Netbook)
  - **Игровая приставка** (Игровая консоль)
  - **Карманный компьютер** (КПК, PDA)
  - **Коммуникатор** (*Communicator, PDA Phone)*
  - **Смартфон** (Smartphone)
  - **Планшетный ПК** (Tablet PC)
- **Рабочая станция** (workstation)
- **Сервер (**server**)**
- **Суперкомпьютер**

Архитектура фон Неймана

1. Принцип **двоичности** - для представления **данных** и **команд** используется **двоичная система счисления**.

2. Принцип **последовательного программного управления** - **программа** состоит из набора **команд**, которые **выполняются процессором** последовательно, одна после завершения другой.

3. Принцип **однородности памяти** - Как **программы** (**команды**), так и **данные** хранятся в одной и той же **памяти**. Над **командами** можно **выполнять** такие же **действия**, как и над **данными**.

4. Принцип **адресуемости памяти** - структурно **основная память** состоит из пронумерованных **ячеек**; **процессору** в произвольный момент времени **доступна** любая **ячейка**.

5.Принцип **условного перехода** - возможно присутствие в **программе команд**, которые изменяют последовательность **выполнения программы**

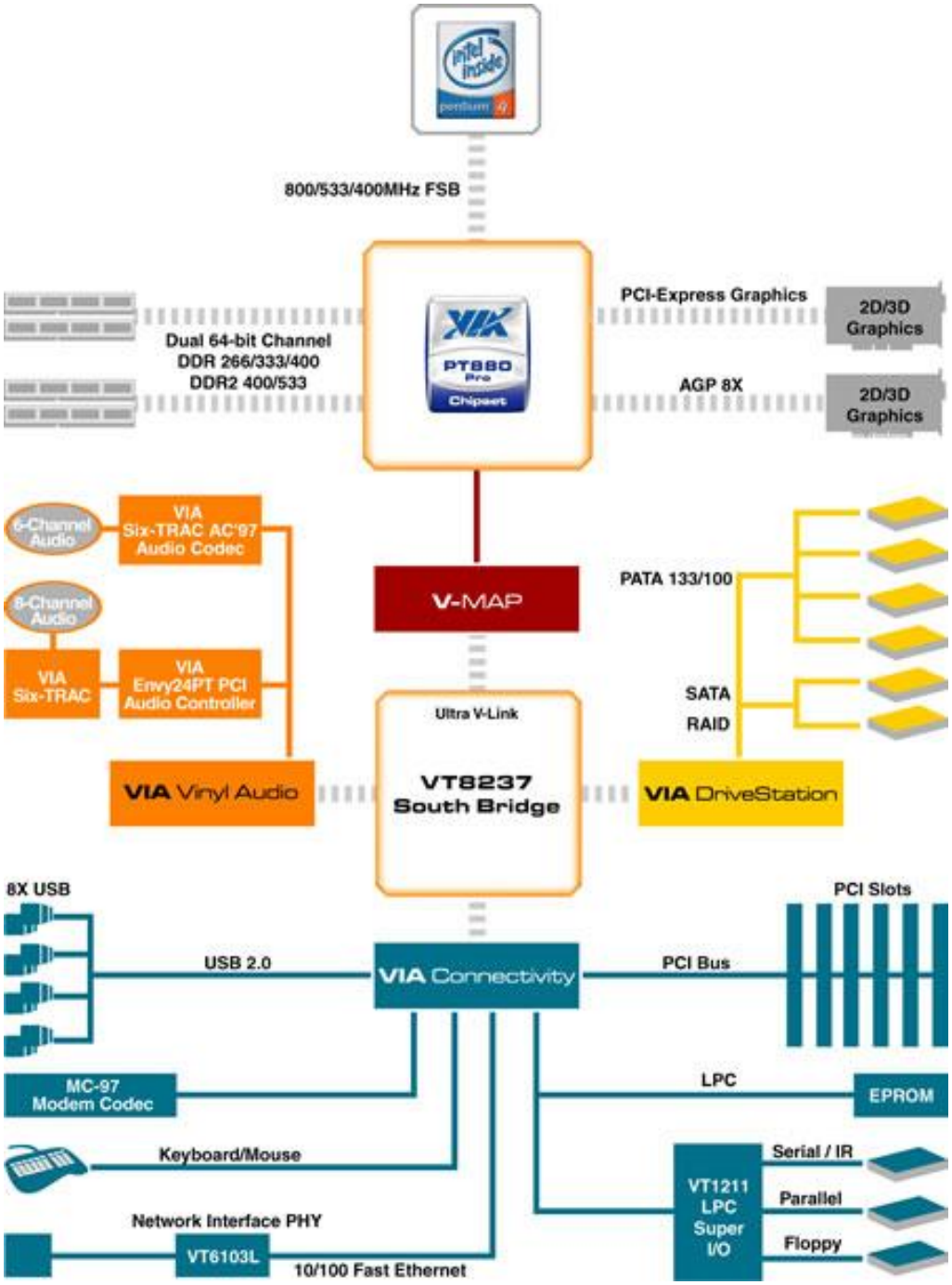
Главные компоненты компьютера

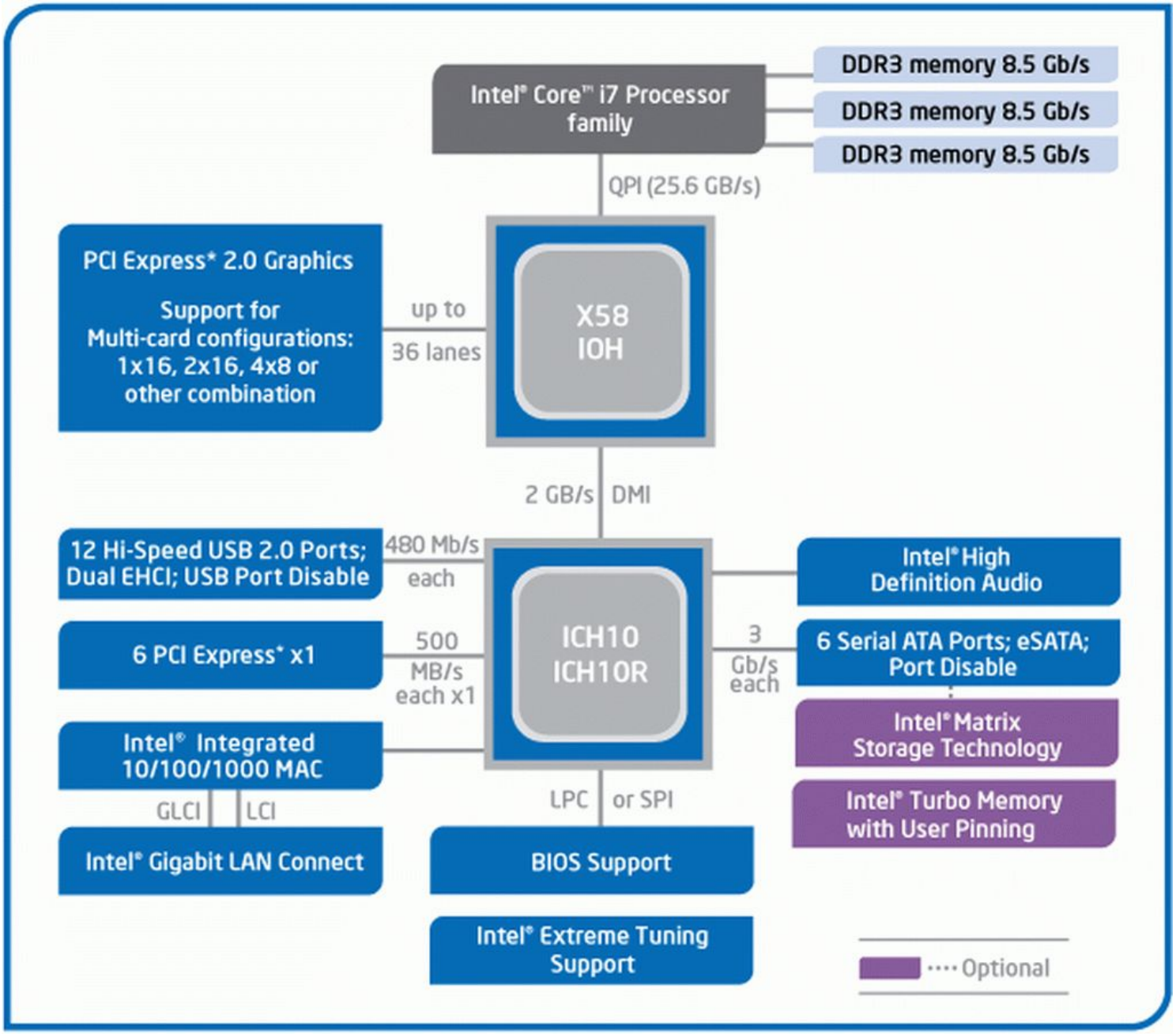
Процессор (Central Processing Unit) – выполняет команды, производит вычисления

Оперативная память (Random Access Memory) предназначена для хранения программных команд и результатов вычислений.

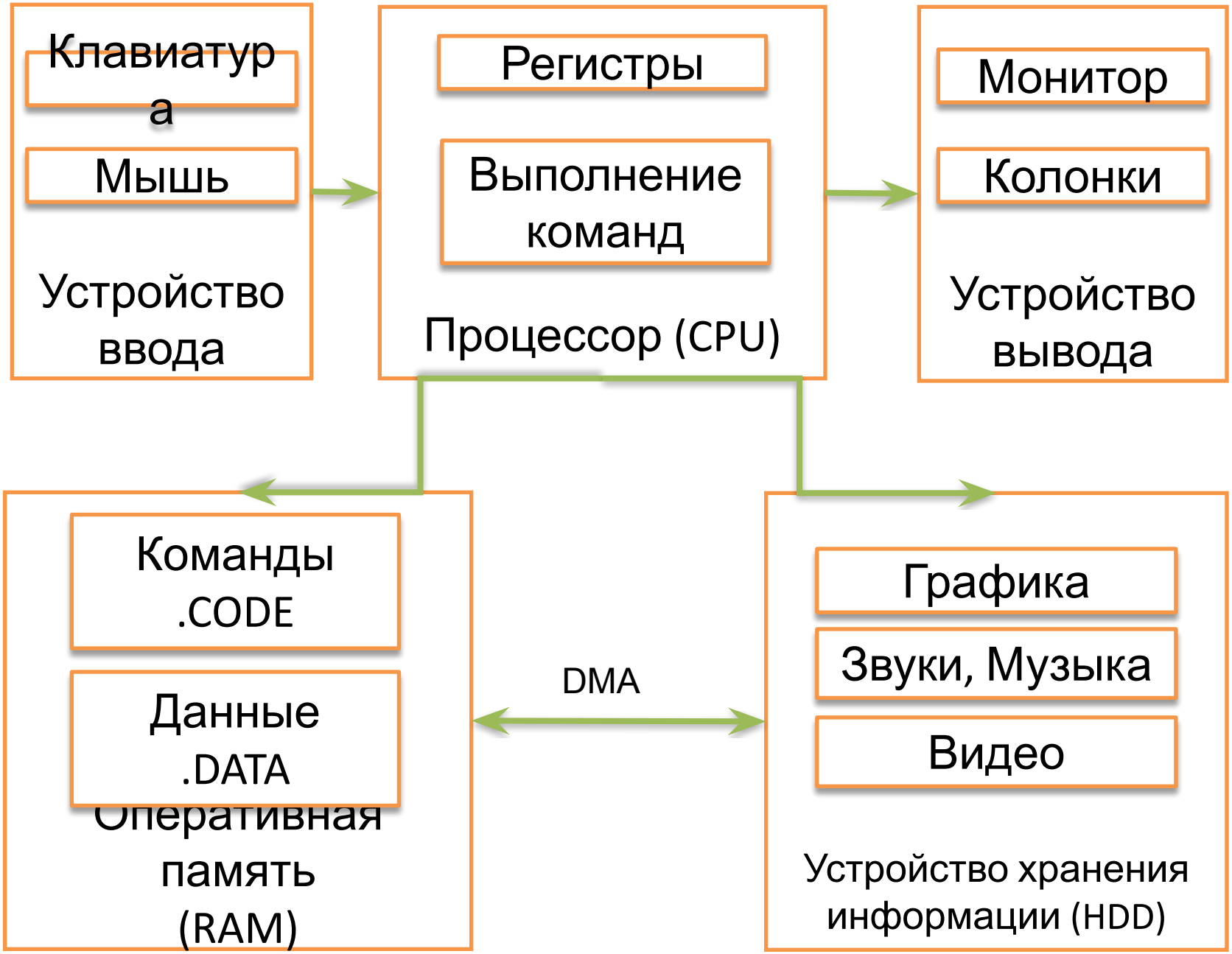
Устройством ввода (Input Device) называется любое устройство, позволяющее человеку передать информацию в компьютер.

Устройством вывода (***Output Device***) называется любое устройство, с помощью которого компьютер выдаёт информацию человеку.

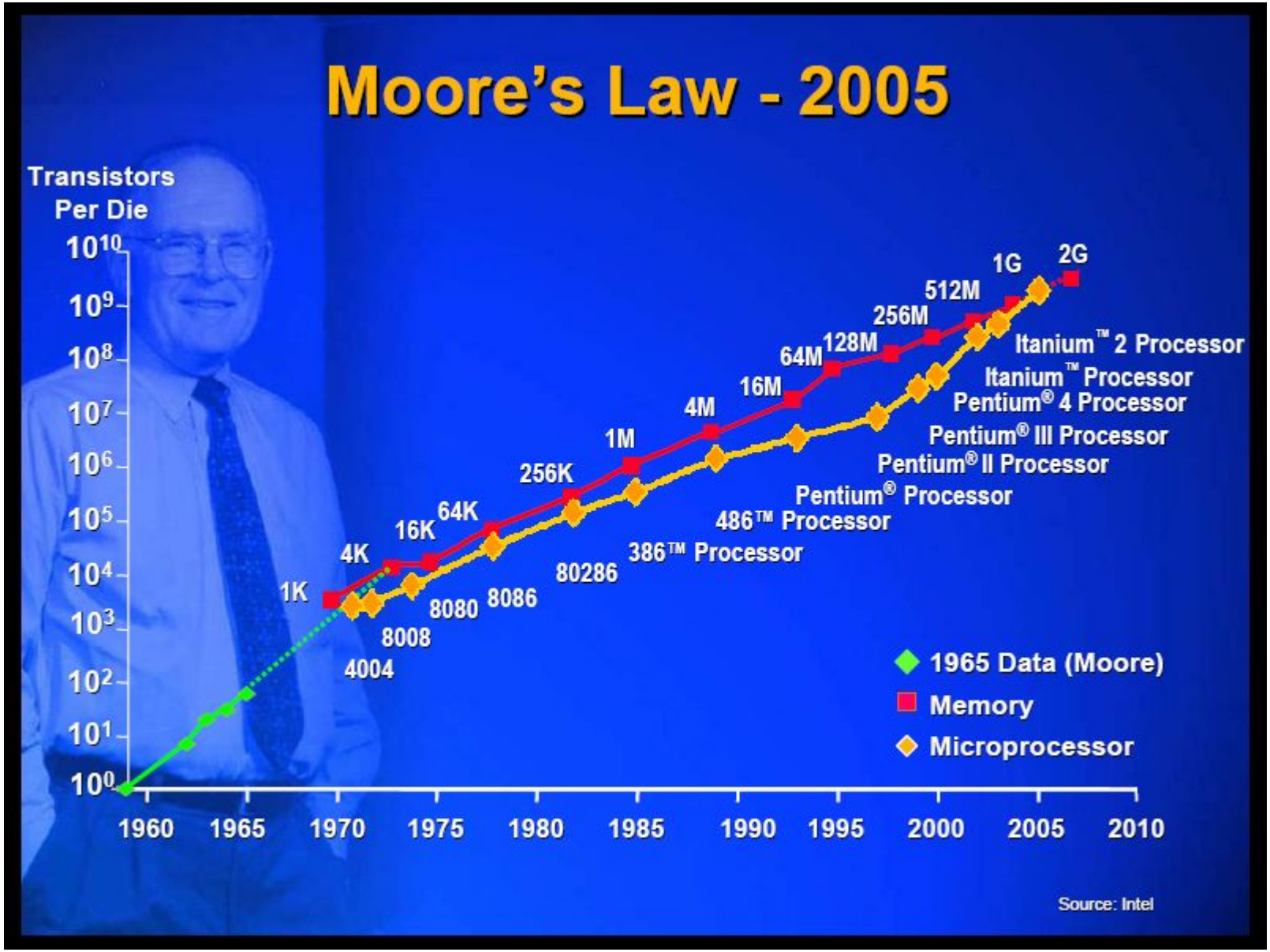

800/533/400MHz FSB
Dual 64-bit Channel
DDR 266/333/400
DDR2 400/533
PT880 Pro
Chipset
PCI-Express Graphics
2D/3D Graphics
AGP 8X
2D/3D Graphics
6-Channel Audio
VIA Six-TRAC AC'97 Audio Codec
8-Channel Audio
VIA Six-TRAC
VIA Envy24PT PCI Audio Controller
VIA Vinyl Audio
V-MAP
PATA 133/100
SATA
RAID
Ultra V-Link
VT8237 South Bridge
VIA DriveStation
8X USB
USB 2.0
VIA Connectivity
PCI Bus
PCI Slots
MC-97 Modem Codec
LPC
EPROM
Keyboard/Mouse
Serial / IR
Parallel
Floppy
VT1211 LPC Super I/O
Network Interface PHY
VT6103L
10/100 Fast Ethernet

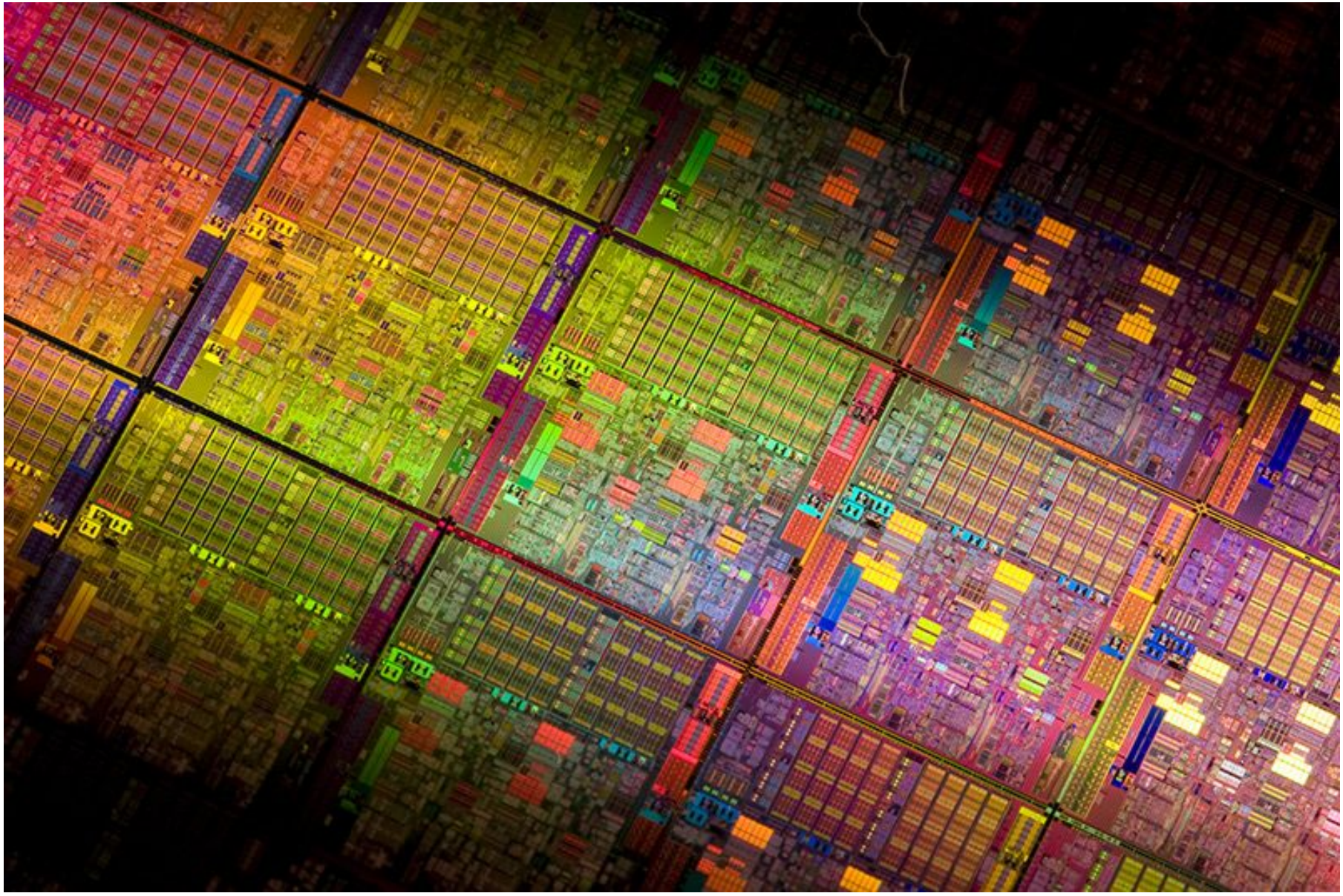

Intel® Core™ i7 Processor family
DDR3 memory 8.5 Gb/s
DDR3 memory 8.5 Gb/s
DDR3 memory 8.5 Gb/s
QPI (25.6 GB/s)
PCI Express* 2.0 Graphics
Support for Multi-card configurations: 1x16, 2x16, 4x8 or other combination
up to 36 lanes
X58 IOH
2 GB/s
DMI
12 Hi-Speed USB 2.0 Ports; Dual EHCI; USB Port Disable
480 Mb/s each
6 PCI Express* x1
500 MB/s each x1
Intel® Integrated 10/100/1000 MAC
GLCI
LCI
Intel® Gigabit LAN Connect
ICH10 ICH10R
3 Gb/s each
Intel® High Definition Audio
6 Serial ATA Ports; eSATA; Port Disable
Intel® Matrix Storage Technology
Intel® Turbo Memory with User Pinning
LPC
or SPI
BIOS Support
Intel® Extreme Tuning Support
Optional

Упрощённая **Архитектура** компьютера



Закон Мура - число **транзисторов** на **кристалле** будет **удваиваться** каждые 24 месяца





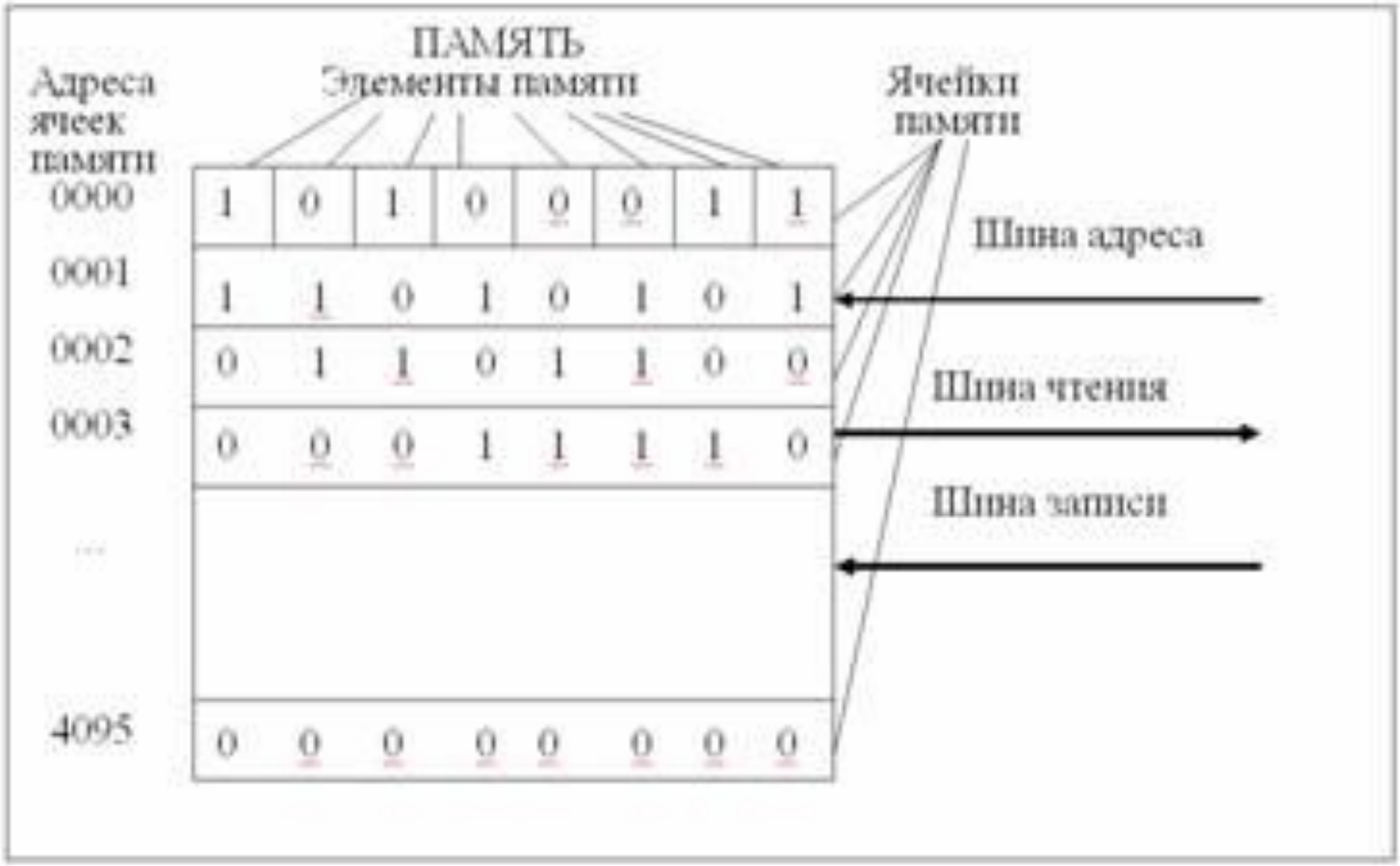
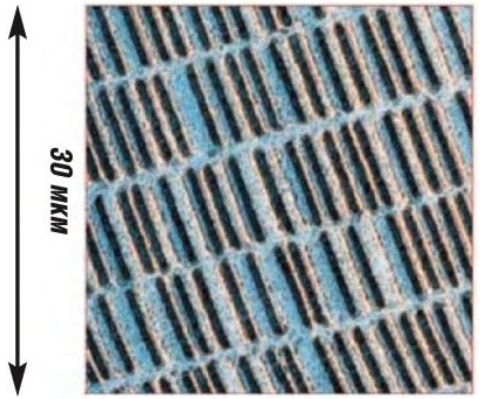
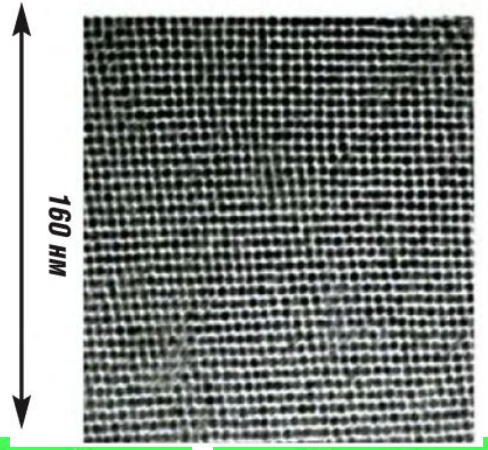
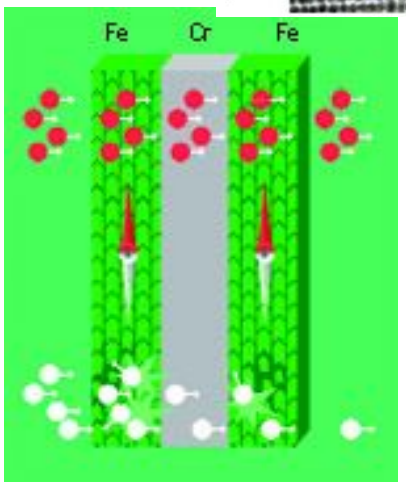


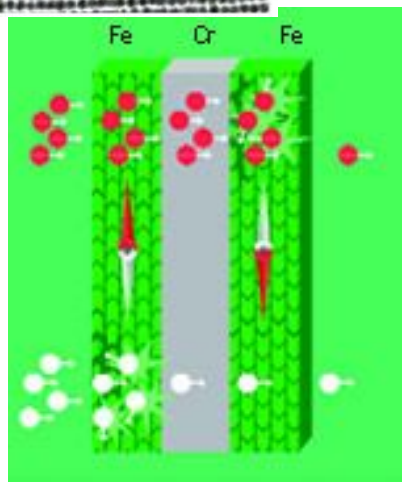
Рис 4.2 Организация оперативной памяти ЭВМ

DEKSI Hard Disk Manager
PROFESSIONAL
V2.90
(3586)
DEK
Software International


30 мкм


160 нм


Fe
Cr
Fe


Fe
Cr
Fe

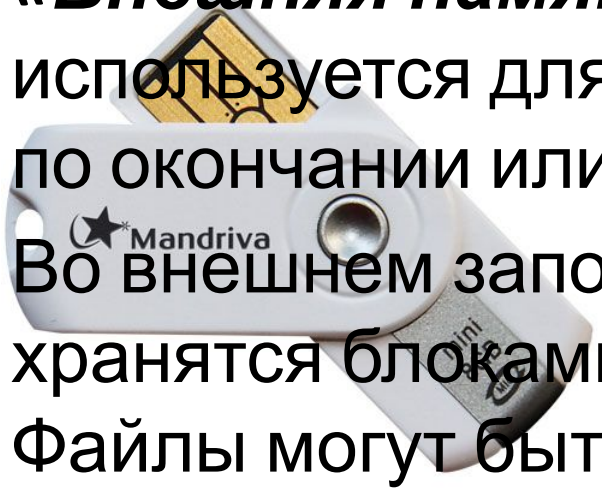
Намагниченная игла

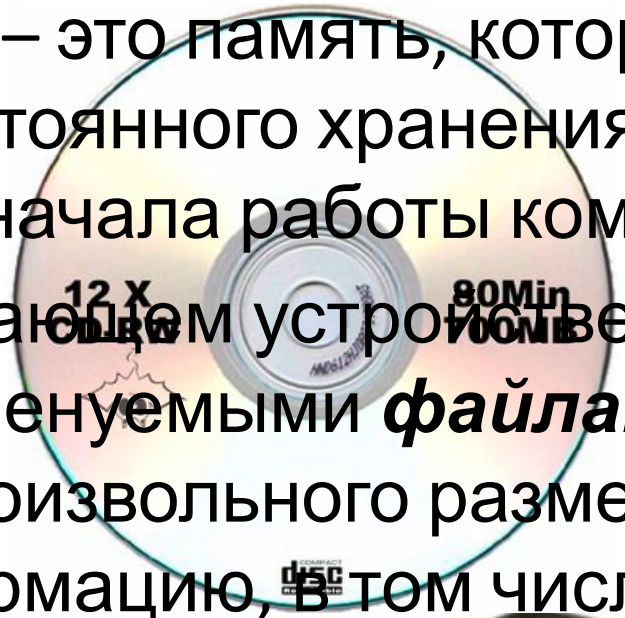
Рис. 1

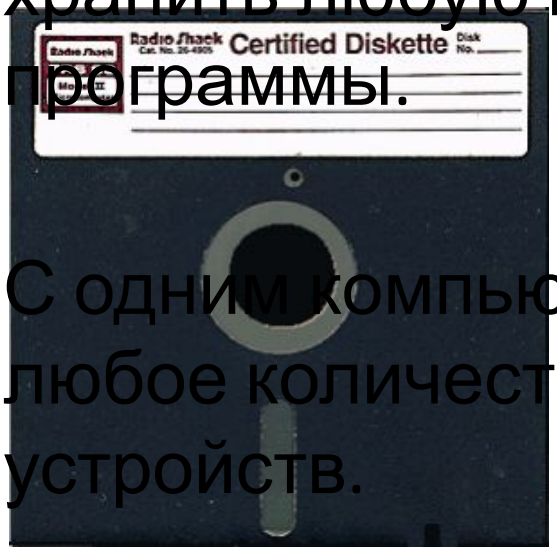
Устройства хранения информации

«Внешняя память» – это память, которая используется для постоянного хранения данных по окончании или до начала работы компьютера.

Во внешнем запоминающем устройстве данные хранятся блоками, именуемыми ***файлами***. Файлы могут быть произвольного размера и хранить любую информацию, в том числе и программы.







С одним компьютером может быть соединено любое количество внешних запоминающих устройств.