Informatyka 1 (ES1F1002)

Politechnika Białostocka - Wydział Elektryczny

Elektrotechnika, semestr I, studia stacjonarne I stopnia

Rok akademicki 2024/2025

Wykład nr 4 (25.10.2024)

dr inż. Jarosław Forenc

Plan wykładu nr 4

- Standard IEEE 754
  - liczby 32-bitowe, liczby 64-bitowe
  - zakres i precyzja liczb
  - wartości specjalne, operacje z wartościami specjalnymi
  - reprezentacja liczb zmiennoprzecinkowych w języku C
- Klasyfikacja systemów komputerowych (Flynna)
- Architektura von Neumanna i architektura harwardzka
- Budowa komputera
  - jednostka centralna
  - płyta główna

Standard IEEE 754

- IEEE Std. 754-2008 - IEEE Standard for Floating-Point Arithmetic
- Standard definiuje następujące klasy liczb zmiennoprzecinkowych:

Precyzja	Długość słowa [bity]	Znak [bity]	Wykładnik		Mantysa	
			Długość [bity]	Zakres	Długość [bity]	Cyfry znaczące
Pojedyncza (Single Precision, binary32)	32	1	8	$2^{\pm 127} \approx 10^{\pm 38}$	23	7
Pojedyncza rozszerzona (Single Extended)	≥ 43	1	≥ 11	$\geq 2^{\pm 1023} \approx 10^{\pm 308}$	≥ 31	≥ 10
Podwójna (Double Precision, binary64)	64	1	11	$2^{\pm 1023} \approx 10^{\pm 308}$	52	16
Podwójna rozszerzona (Double Extended)	≥ 79	1	≥ 15	$\geq 2^{\pm 16383} \approx 10^{\pm 4932}$	≥ 63	≥ 19

źródło: Gryś S.: „Arytmetyka komputerów w praktyce”. PWN, Warszawa, 2007 (str. 116).

Standard IEEE 754

- W przypadku liczb:
  - pojedynczej rozszerzonej precyzji (ang. Single Precision)
  - podwójnej rozszerzonej precyzji (ang. Double Precision)

  standard podaje jedynie minimalną liczbę bitów pozostawiając szczegóły implementacji producentom procesorów i kompilatorów
- Stosowany był 80-bitowy format podwójnej rozszerzonej precyzji (Extended Precision) wprowadzony przez firmę Intel
- W 80-bitowym formacie Intela:
  - długość słowa: 80 bitów
  - znak: 1 bit
  - wykładnik: 15 bitów (zakres: $2^{\pm 16383} \approx 10^{\pm 4932}$)
  - mantysa: 63 bity (cyfry znaczące: 19)

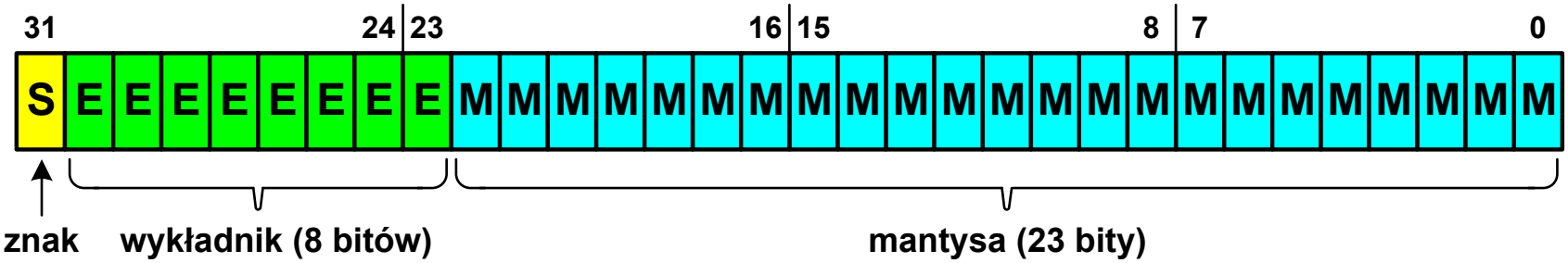
Standard IEEE 754

- Standard IEEE 754 definiuje dziesiętne typy zmiennoprzecinkowe (operujące na cyfrach dziesiętnych):
  - decimal32 (32 bity, 7 cyfr dziesiętnych)
  - decimal64 (64 bity, 16 cyfr dziesiętnych)
  - decimal128 (128 bitów, 34 cyfry dziesiętnych)

- Standard IEEE 754 definiuje:
  - sposób reprezentacji specjalnych wartości, np. nieskończoności, zera
  - sposób wykonywania działań na liczbach zmiennoprzecinkowych
  - sposób zaokrąglania liczb

Standard IEEE 754 - liczby 32-bitowe

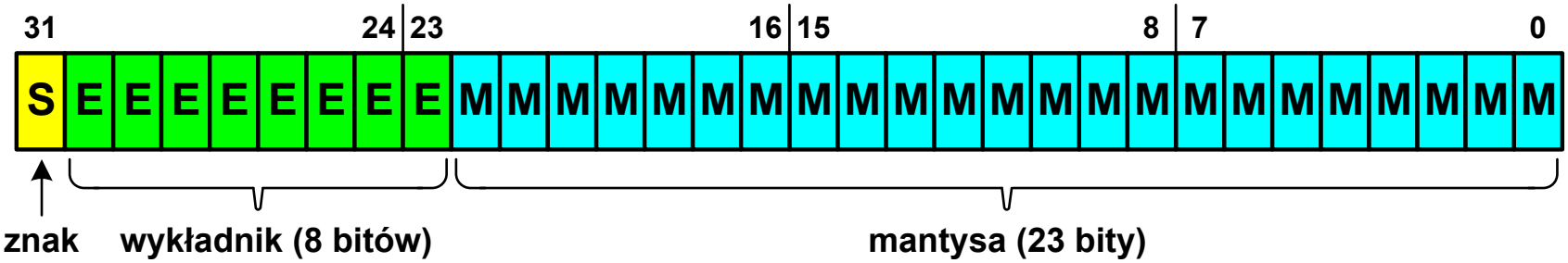
- Liczba pojedynczej precyzji przechowywana jest na 32 bitach:



- Pierwszy bit w zapisie (bit nr 31) jest bitem znaku (0 - liczba dodatnia, 1 - liczba ujemna)
- Wykładnik zapisywany jest na 8 bitach (bity nr 30-23) z nadmiarem o wartości 127
- Wykładnik może przyjmować wartości od -127 (wszystkie bity wyzerowane) do 128 (wszystkie bity ustawione na 1)

Standard IEEE 754 - liczby 32-bitowe

- Liczba pojedynczej precyzji przechowywana jest na 32 bitach:



- Mantysa w większości przypadków jest znormalizowana
- Wartość mantysy zawiera się pomiędzy 1 a 2, a zatem w zapisie liczby pierwszy bit jest zawsze równy 1
- Powyższy bit nie jest zapamiętywany, natomiast jest automatycznie uwzględniany podczas wykonywania obliczeń
- Dzięki pominięciu tego bitu zyskujemy dodatkowy bit mantysy (zamiast 23 bitów mamy 24 bity)

Standard IEEE 754 - liczby 32-bitowe

- Przykład:
  - obliczmy wartość dziesiętną liczby zmiennoprzecinkowej

$$01000010110010000000000000000000_{(\mathrm{IEEE}754)} = ?_{(10)}$$

  - dzielimy liczbę na części

$$\underbrace{0}_{\mathrm{S-bit\ znaku}} \quad \underbrace{10000101}_{\mathrm{E-wykladnik}} \quad \underbrace{10010000000000000000000}_{\mathrm{M-mantysa\ (tylko\ czesc\ ulamkowa)}}$$

  - określamy znak liczby

$$\mathrm{S} = 0 \quad -\mathrm{liczba\ dodatnia}$$

  - obliczamy wykładnik (nadmiar: 127)

$$10000101_{(2)} = 128+4+1 = 133 \quad \Rightarrow \quad \mathrm{E} = 133 - \underbrace{127}_{nadmiar} = 6_{(10)}$$

Standard IEEE 754 - liczby 32-bitowe

- Przykład (cd.):
  - wyznaczamy mantysę dopisując na początku 1, (część całkowita)

$$M = 1{,}10010000000000000000000 =$$

$$= 1 \cdot 2^{0} + 1 \cdot 2^{-1} + 1 \cdot 2^{-4} = 1 + 0{,}5 + 0{,}0625 = 1{,}5625_{(10)}$$

  - wzór na wartość dziesiętną liczby zmiennoprzecinkowej:

$$L = (-1)^{S} \cdot M \cdot 2^{E}$$

  - podstawiając otrzymujemy:

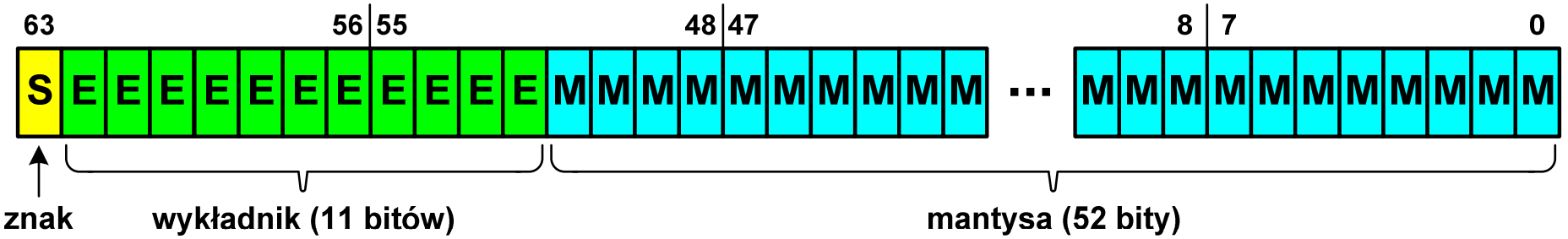
$$S = 0, \qquad E = 6_{(10)}, \qquad M = 1{,}5625_{(10)}$$

$$L = (-1)^{0} \cdot 1{,}5625 \cdot 2^{6} = 100_{(10)}$$

$$01000010110010000000000000000000_{(\mathrm{IEEE}\,754)} = 100_{(10)}$$

Standard IEEE 754 - liczby 64-bitowe

- Liczba podwójnej precyzji przechowywana jest na 64 bitach:



- Pierwszy bit w zapisie (bit nr 63) jest bitem znaku (0 - liczba dodatnia, 1 - liczba ujemna)
- Wykładnik zapisywany jest na 11 bitach (bity nr 62-52) z nadmiarem o wartości 1023
- Wykładnik może przyjmować wartości od -1023 (wszystkie bity wyzerowane) do 1024 (wszystkie bity ustawione na 1)
- Mantysa zapisywana jest na 52 bitach (pierwszy bit mantysy, zawsze równy 1, nie jest zapamiętywany)

Standard IEEE 754 - zakres liczb

- Pojedyncza precyzja:
  - największa wartość: $\approx 3{,}4 \cdot 10^{38}$
  - najmniejsza wartość: $\approx 1{,}4 \cdot 10^{-45}$
  - zakres liczb: $\langle -3{,}4 \cdot 10^{38} \ldots -1{,}4 \cdot 10^{-45} \rangle \cup \{0\} \cup \langle 1{,}4 \cdot 10^{-45} \ldots 3{,}4 \cdot 10^{38} \rangle$
- Podwójna precyzja:
  - największa wartość: $\approx 1{,}8 \cdot 10^{308}$
  - najmniejsza wartość: $\approx 4{,}9 \cdot 10^{-324}$
  - zakres liczb: $\langle -1{,}8 \cdot 10^{308} \ldots -4{,}9 \cdot 10^{-324} \rangle \cup \{0\} \cup \langle 4{,}9 \cdot 10^{-324} \ldots 1{,}8 \cdot 10^{308} \rangle$
- Podwójna rozszerzona precyzja:
  - największa wartość: $\approx 1{,}2 \cdot 10^{4932}$
  - najmniejsza wartość: $\approx 3{,}6 \cdot 10^{-4951}$
  - zakres liczb: $\langle -1{,}2 \cdot 10^{4932} \ldots -3{,}6 \cdot 10^{-4951} \rangle \cup \{0\} \cup \langle 3{,}6 \cdot 10^{-4951} \ldots 1{,}2 \cdot 10^{4932} \rangle$

Standard IEEE 754 - precyzja liczb

- Precyzja - liczba zapamiętywanych cyfr znaczących w systemie (10)

  4,86452137846 → 4,864521 - 7 cyfr znaczących

- Precyzja liczby zależy od liczby bitów mantysy
- Liczba bitów potrzebnych do zakodowania 1 cyfry dziesiętnej:

$$10^1 = 2^n \quad \rightarrow \quad n = \log_2(10) \approx 3{,}321928$$

- Liczba cyfr dziesiętnych (d) możliwa do zakodowania na m bitach:

  $\log_2(10)$ bitów - 1 cyfra dziesiętna

  m bitów - d cyfr dziesiętnych

$$d = \frac{m}{\log_2(10)}$$

Standard IEEE 754 - precyzja liczb

- Dla formatu pojedynczej precyzji:
  - mantysa: 23 + 1 = 24 bity
  - cyfry znaczące: 7

$$d = \frac{24}{\log_2(10)} = \frac{24}{3{,}321928} = 7{,}2247 \approx 7$$

- Dla formatu podwójnej precyzji:
  - mantysa: 52 + 1 = 53 bity
  - cyfry znaczące: 16

$$d = \frac{53}{\log_2(10)} = \frac{53}{3{,}321928} = 15{,}9546 \approx 16$$

- Dla formatu podwójnej rozszerzonej precyzji:
  - mantysa: 63 + 1 = 64 bity
  - cyfry znaczące: 19

$$d = \frac{64}{\log_2(10)} = \frac{64}{3{,}321928} = 19{,}2659 \approx 19$$

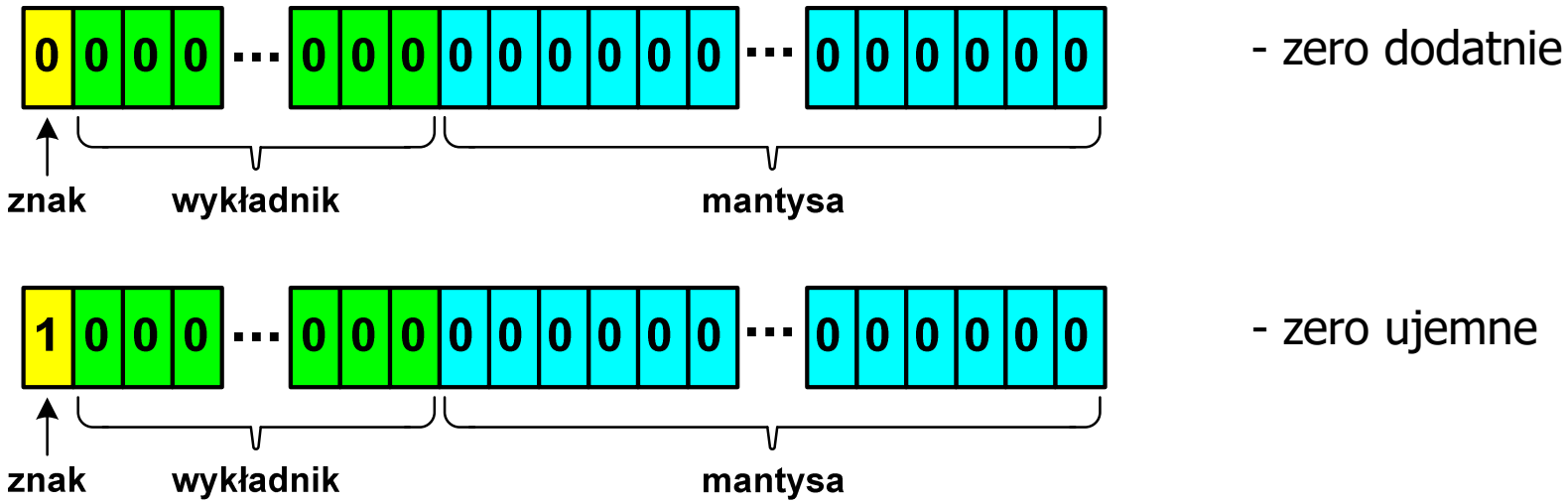
Standard IEEE 754 - precyzja liczb

```
#include <stdio.h>

int main()
{
    float x;
    double y;

    x = 1234567890.0;    /* 1.234.567.890 */
    y = 1234567890.0;    /* 1.234.567.890 */
    printf("float  -> %f\n",x);
    printf("double -> %f\n\n",y);

    y = 12345678901234567890.0;
    printf("double -> %f\n",y);

    return 0;
}
```

```
float  -> 1234567936.000000
double -> 1234567890.000000

double -> 12345678901234567000.000000
```

Standard IEEE 754 - wartości specjalne

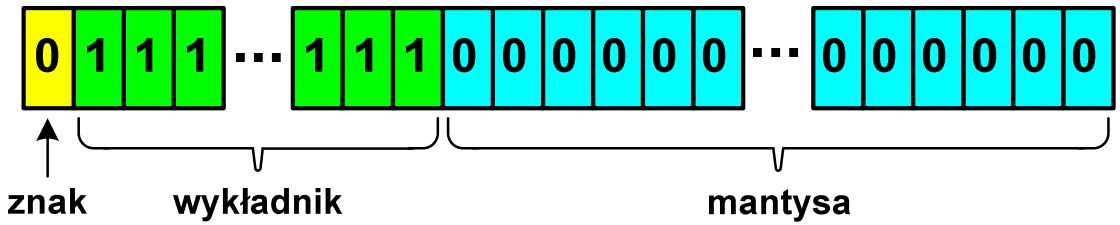
- Zero:



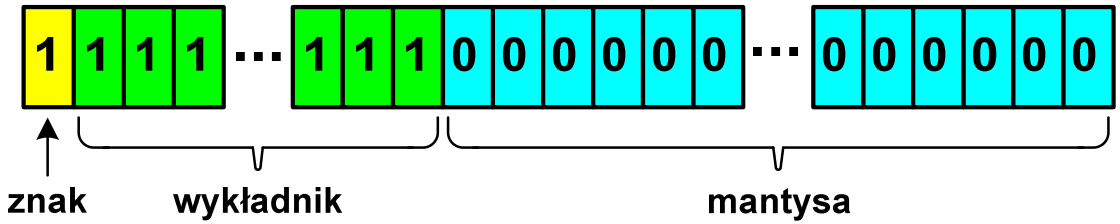
- Podczas porównań zero dodatnie i ujemne są traktowane jako równe sobie

Standard IEEE 754 - wartości specjalne

- Nieskończoność:



- nieskończoność dodatnia

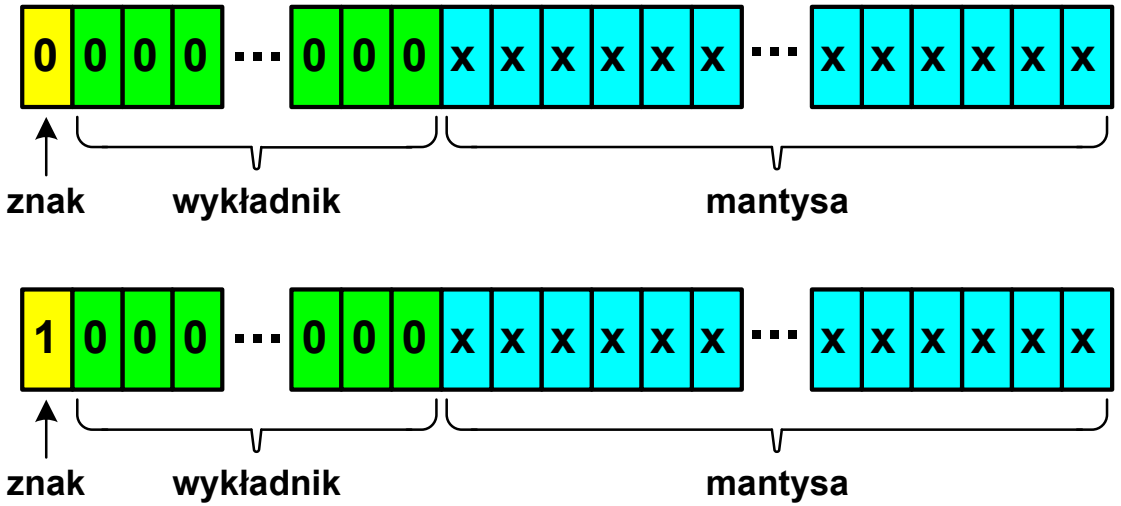


- nieskończoność ujemna

- Nieskończoność występuje w przypadku wystąpienia nadmiaru (przepełnienia) oraz przy dzieleniu przez zero

Standard IEEE 754 - wartości specjalne

- Liczba zdenormalizowana:

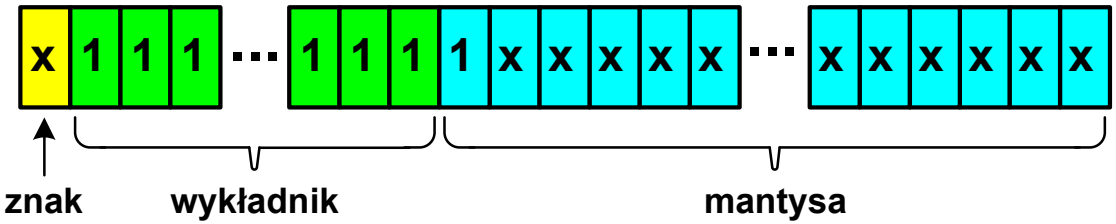


- Pojawia się, gdy występuje niedomiar (ang. underflow), ale wynik operacji można jeszcze zapisać denormalizując mantysę
- Mantysa nie posiada domyślnej części całkowitej równej 1, tzn. reprezentuje liczbę o postaci 0,xxx…xxx, a nie 1,xxx…xxx

Standard IEEE 754 - wartości specjalne

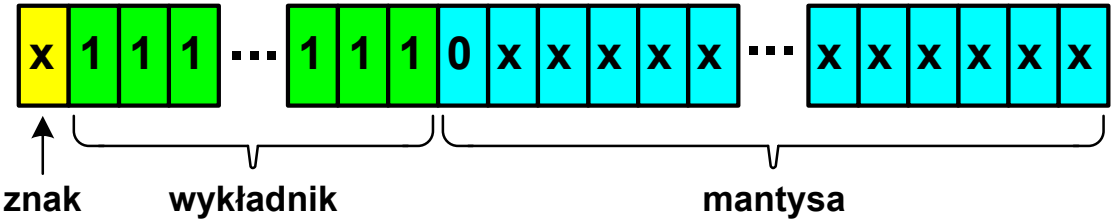
- Nieliczby - NaN (Not A Number) - nie reprezentują wartości liczbowej
- Powstają w wyniku wykonania niedozwolonej operacji

- QNaN (ang. Quiet NaN) - ciche nieliczby



- „przechodzą” przez działania arytmetyczne (brak przerwania wykonywania programu)

- SNaN (ang. Signaling NaN) - sygnalizujące, istotne, głośne nieliczby



- zgłoszenie wyjątku (przerwanie wykonywania programu)

Standard IEEE 754 - wartości specjalne

- Standard IEEE 754 definiuje dokładnie wyniki operacji, w których występują specjalne argumenty

Operacja	Wynik
x / ±∞	0
±∞ · ±∞	±∞
±wart_niezer / 0	±∞
∞ + ∞	∞
±0 / ±0	NaN
∞ - ∞	NaN
±∞ / ±∞	NaN
±∞ · 0	NaN

Język C - operacje z wartościami specjalnymi

```
#include <stdio.h>
#include <math.h>

int main(void)
{
  float x = 0.0;
  printf("1.0/0.0     = %f\n",1.0/x);
  printf("-1.0/0.0    = %f\n",-1.0/x);
  printf("0.0/0.0     = %f\n",0.0/x);
  printf("sqrt(-1.0)  = %f\n",sqrt(-1.0));
  printf("1.0/INF     = %f\n",1.0/(1.0/x));
  printf("0*INF       = %f\n",0.0*(1.0/x));

  return 0;
}
```

```
1.0/0.0     =  1.#INF00
-1.0/0.0    = -1.#INF00
0.0/0.0     = -1.#IND00
sqrt(-1.0)  = -1.#IND00
1.0/INF     =  0.000000
0*INF       = -1.#IND00
```

- Środowisko: Microsoft Visual C++ 2008 Express Edition

Operacja	Wynik
x / ±∞	0
±∞ · ±∞	±∞
±wart_niezer / 0	±∞
∞ + ∞	∞
±0 / ±0	NaN
∞ - ∞	NaN
±∞ / ±∞	NaN
±∞ · 0	NaN

Język C - operacje z wartościami specjalnymi

```c
#include <stdio.h>
#include <math.h>

int main(void)
{
  printf("1.0/0.0     = %f\n",1.0/0.0);
  printf("-1.0/0.0    = %f\n",-1.0/0.0);
  printf("0.0/0.0     = %f\n",0.0/0.0);
  printf("sqrt(-1.0)  = %f\n",sqrt(-1.0));
  printf("1.0/INF     = %f\n",1.0/(1.0/0.0));
  printf("0*INF       = %f\n",0.0*(1.0/0.0));

  return 0;
}
```

```
1.0/0.0     =  1.#INF00
-1.0/0.0    = -1.#INF00
0.0/0.0     = -1.#IND00
sqrt(-1.0)  = -1.#IND00
1.0/INF     =  0.000000
0*INF       = -1.#IND00
```

- Środowisko: Code::Blocks 20.03

Operacja	Wynik
x / ±∞	0
±∞ · ±∞	±∞
±wart_niezer / 0	±∞
∞ + ∞	∞
±0 / ±0	NaN
∞ - ∞	NaN
±∞ / ±∞	NaN
±∞ · 0	NaN

Reprezentacja liczb zmiennoprzecinkowych w C

- Typy zmiennoprzecinkowe w języku C:

Nazwa typu	Rozmiar (bajty)	Zakres wartości	Cyfry znaczące
float	4 bajty	$-3{,}4\cdot10^{38}$ … $3{,}4\cdot10^{38}$	7-8
double	8 bajtów	$-1{,}8\cdot10^{308}$ … $1{,}8\cdot10^{308}$	15-16
long double	10 bajtów	$-1{,}2\cdot10^{4932}$ … $1{,}2\cdot10^{4932}$	19-20

- Typ long double może mieć także inny rozmiar:

Środowisko	Rozmiar (bajty)
MS Visual C++ 2008 EE	8 bajtów
Borland Turbo C++ Explorer	10 bajtów
Code:Blocks 20.03	16 bajtów (*)
Dev-C++ 5.11	16 bajtów (*)

Reprezentacja liczb zmiennoprzecinkowych w C

```
#include <stdio.h>

int main(void)
{
    float       sf  = 0.0f;
    double      sd  = 0.0;
    long double slg = 0.0L;

    for (int i=0; i<10000; i++)
    {
        sf  = sf  + 0.01f;
        sd  = sd  + 0.01;
        slg = slg + 0.01L;
    }

    printf("float:       %.20f\n",sf);
    printf("double:      %.20f\n",sd);
    printf("long double: %.20Lf\n",slg);

    return 0;
}
```

Reprezentacja liczb zmiennoprzecinkowych w C

- Microsoft Visual C++ 2008 Express Edition (long double - 8 bajtów)

```
float:       100.00295257568359000000
double:      100.00000000001425000000
long double: 100.00000000001425000000
```

- Borland Turbo C++ Explorer (long double - 10 bajtów)

```
float:       100.00295257568359375000
double:      100.00000000001425349000
long double: 100.00000000000001388000
```

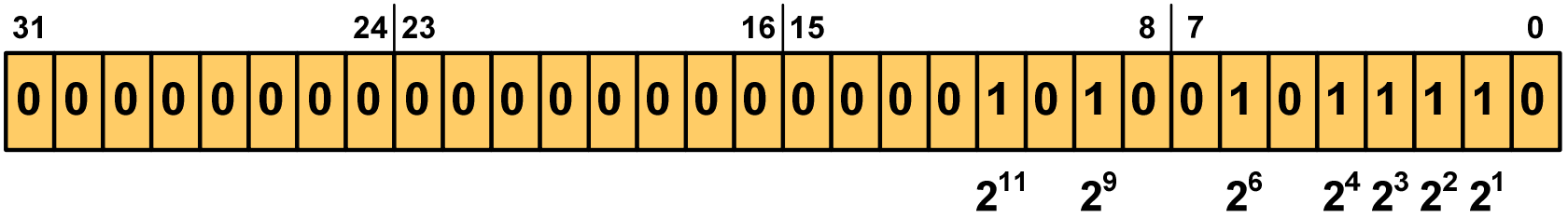
- Code::Blocks 20.03 (long double - 16 bajtów)

```
float:       100.00295257568359000000
double:      100.00000000001425000000
long double: 0.00000000000000000000
```

```
warning: unknown conversion
type character 'L' in format
[-Wformat=]
```

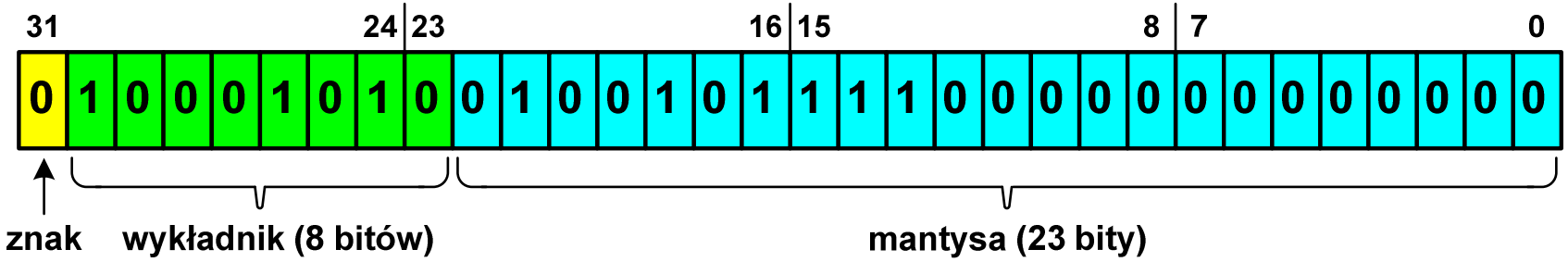
Liczba $2654_{(10)}$ jako całkowita i rzeczywista w C

- int (4 bajty): $2654_{(10)} = 00\ 00\ 0A\ 5E_{(16)}$



$$2^{11} + 2^9 + 2^6 + 2^4 + 2^3 + 2^2 + 2^1 = 2048 + 512 + 64 + 16 + 8 + 4 + 2 = 2654_{(10)}$$

- float (4 bajty): $2654_{(10)} = 45\ 25\ E0\ 00_{(IEEE\ 754)}$



$+ \quad 138 - 127 = 11_{(10)}$ $\qquad 1\ .0100101111_{(2)} = 1.2958984_{(10)}$

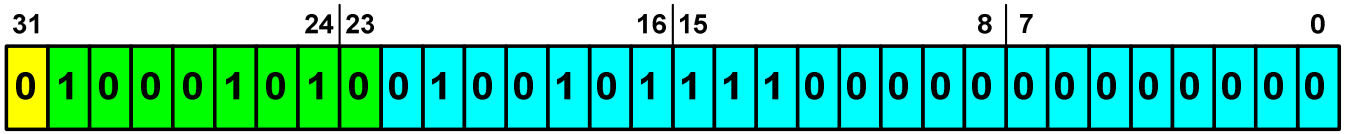
$$1.2958984 \cdot 2^{11} = 2654_{(10)}$$

Język C - nieprawidłowy specyfikator formatu

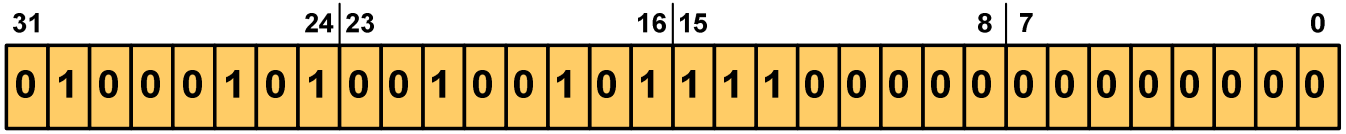
```
int x;
printf("x (%%f) = "); scanf("%f",&x);
printf("x (%%d) = %d\n",x);
printf("x (%%f) = %f\n",x);
printf("x (%%e) = %e\n",x);
```

```
x (%f) = 2654
x (%d) = 1160110080
x (%f) = 0.000000
x (%e) = 5.731705e-315
```

- Zgodnie ze standardem języka C wynik jest niezdefiniowany
- Zapamiętana wartość:



- Wyświetlona wartość przy wykorzystaniu %d:



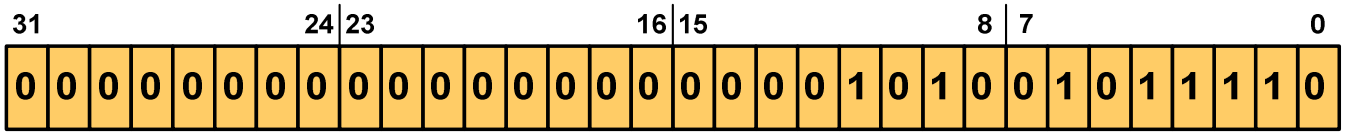
$$2^{30} + 2^{26} + 2^{24} + 2^{21} + 2^{18} + 2^{16} + 2^{15} + 2^{14} + 2^{13} = 1.160.110.080_{(10)}$$

Język C - nieprawidłowy specyfikator formatu

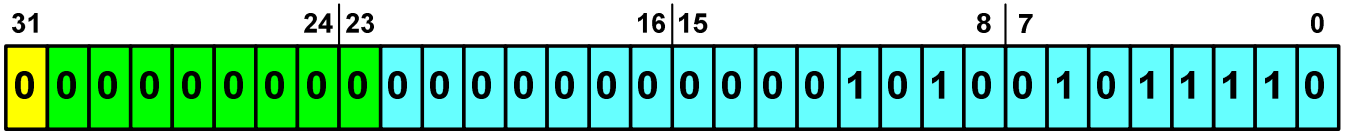
```
float x;
printf("x (%%d) = "); scanf("%d",&x);
printf("x (%%d) = %d\n",x);
printf("x (%%f) = %f\n",x);
printf("x (%%e) = %e\n",x);
```

```
x (%d) = 2654
x (%d) = 0
x (%f) = 0.000000
x (%e) = 3.719046e-042
```

- Zgodnie ze standardem języka C wynik jest niezdefiniowany
- Zapamiętana wartość:



- Wyświetlona wartość przy wykorzystaniu %e:

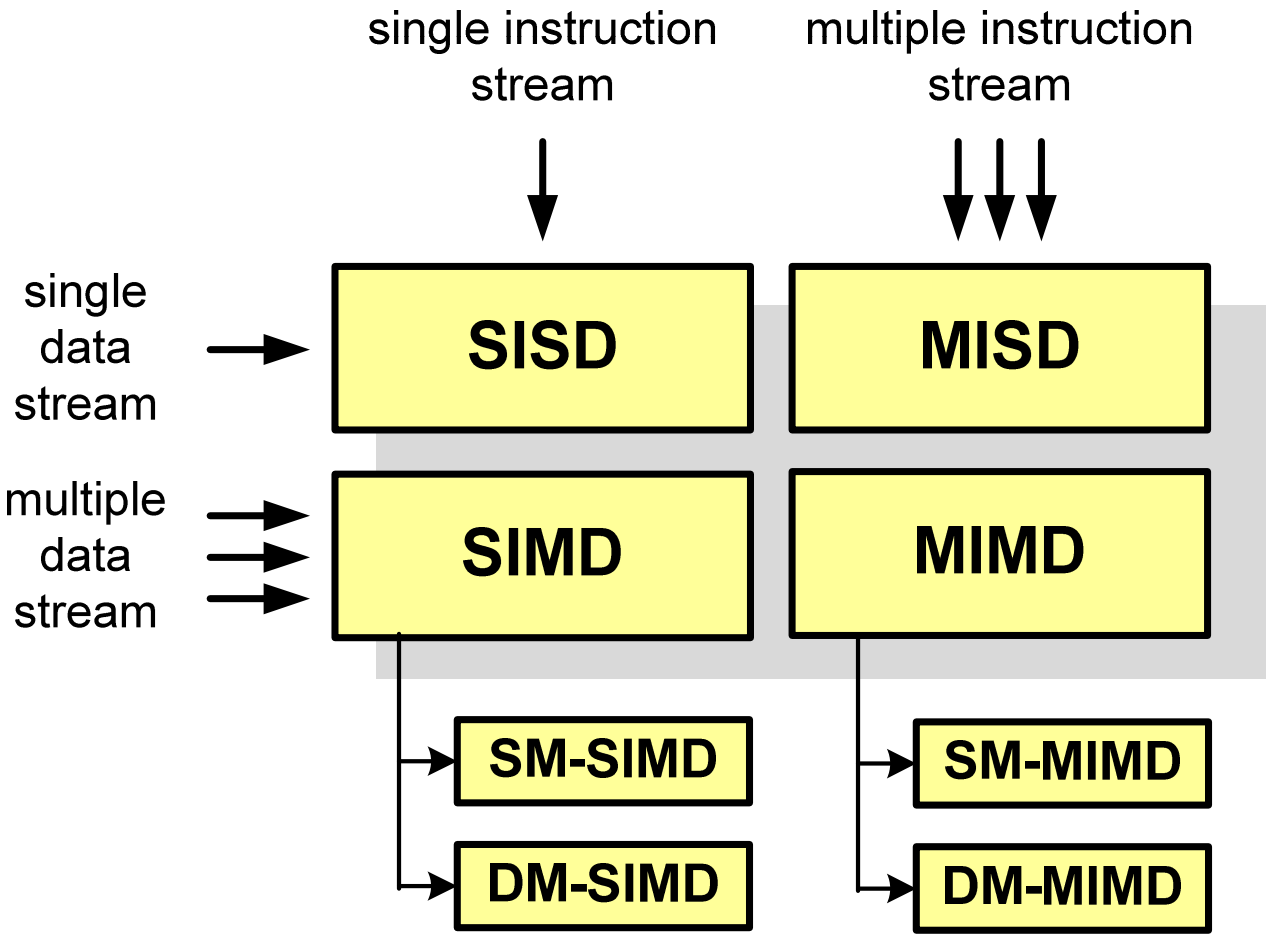


Liczba zdenormalizowana: 3,719046E-42

Klasyfikacja systemów komputerowych

- Taksonomia Flynna - pierwsza, najbardziej ogólna klasyfikacja architektur komputerowych (1966, 1972):
  - Flynn M.J.: „Some Computer Organizations and Their Effectiveness”, IEEE Transactions on Computers, Vol. C-21, No 9, 1972.
- Opiera się na liczbie przetwarzanych strumieni rozkazów i strumieni danych:
  - strumień rozkazów (Instruction Stream) - odpowiednik licznika rozkazów; system złożony z n procesorów posiada n liczników rozkazów, a więc n strumieni rozkazów
  - strumień danych (Data Stream) - zbiór operandów, np. system rejestrujący temperaturę mierzoną przez n czujników posiada n strumieni danych

Taksonomia Flynna



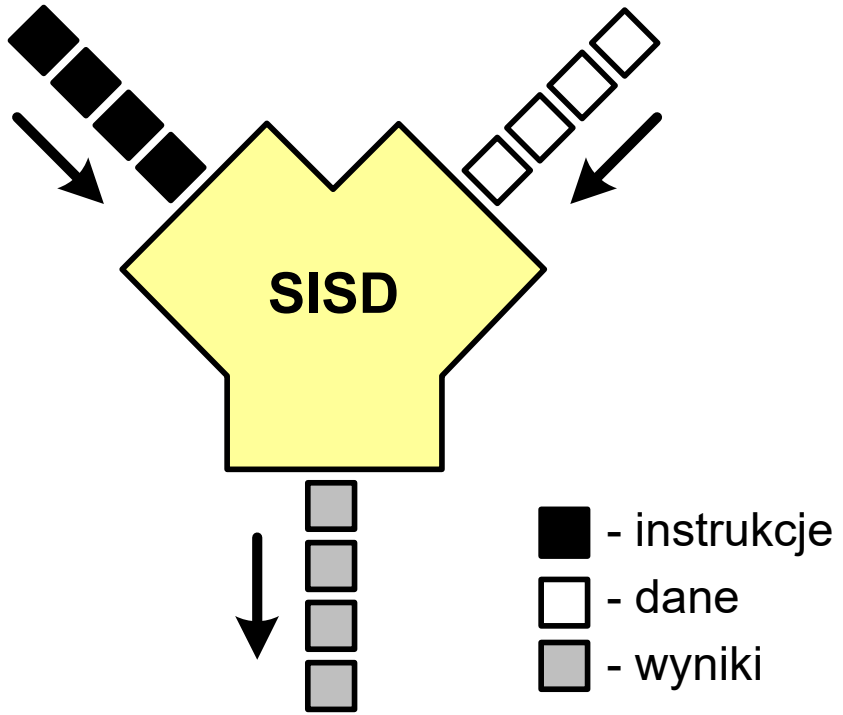
SI - Single instruction
MI - Multiple Instruction

SD - Single Data
MD - Multiple Data

SM - Shared Memory
DM - Distributed Memory

SISD (Single Instruction, Single Data)

- Jeden wykonywany program przetwarza jeden strumień danych
- Klasyczne komputery zbudowane według architektury von Neumanna
- Zawierają:
  - jeden procesor
  - jeden blok pamięci operacyjnej zawierający wykonywany program.



SISD (Single Instruction, Single Data)

Komputer IBM PC/AT

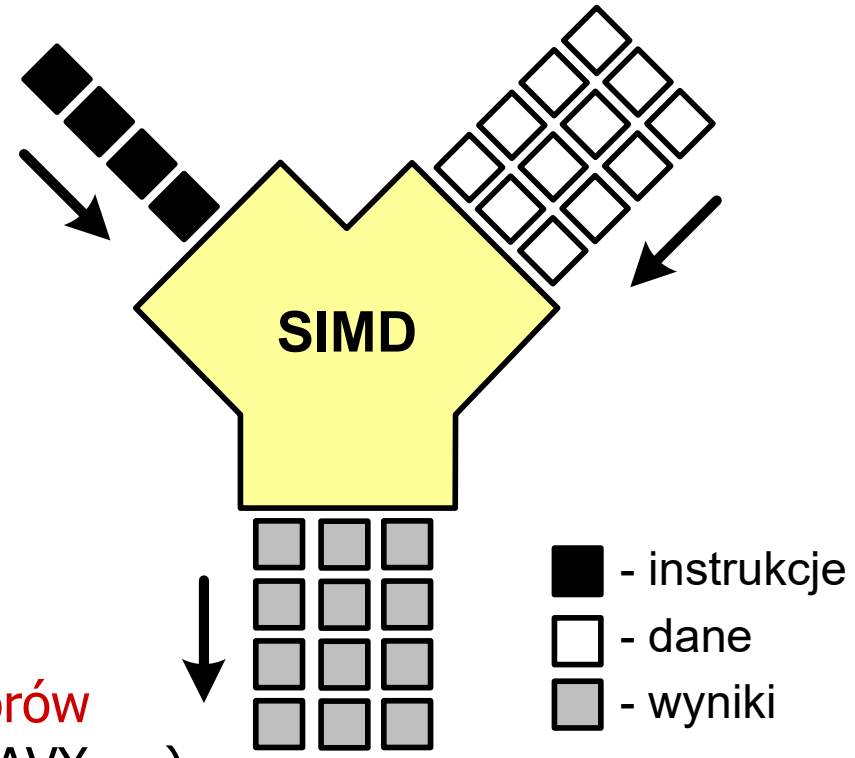
Komputer PC

Komputer PC

Laptop

SIMD (Single Instruction, Multiple Data)

- Jeden wykonywany program przetwarza wiele strumieni danych
- Te same operacje wykonywane są na różnych danych
- Podział:
  - SM-SIMD (Shared Memory SIMD):
    - komputery wektorowe
    - rozszerzenia strumieniowe procesorów (MMX, 3DNow!, SSE, SSE2, SSE3, AVX, ...)
  - DM-SIMD (Distributed Memory SIMD):
    - tablice procesorów
    - procesory kart graficznych (GPGPU)



SM-SIMD - Komputery wektorowe

CDC Cyber 205 (1981)

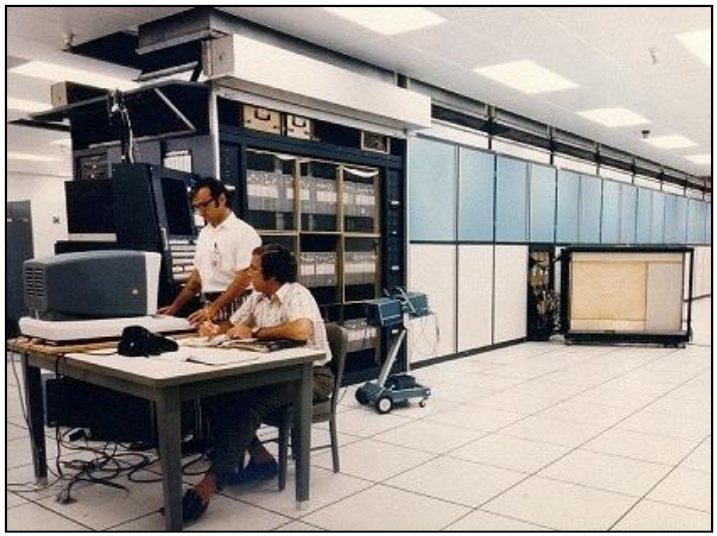
Cray-1 (1976)

Cray-2 (1985)

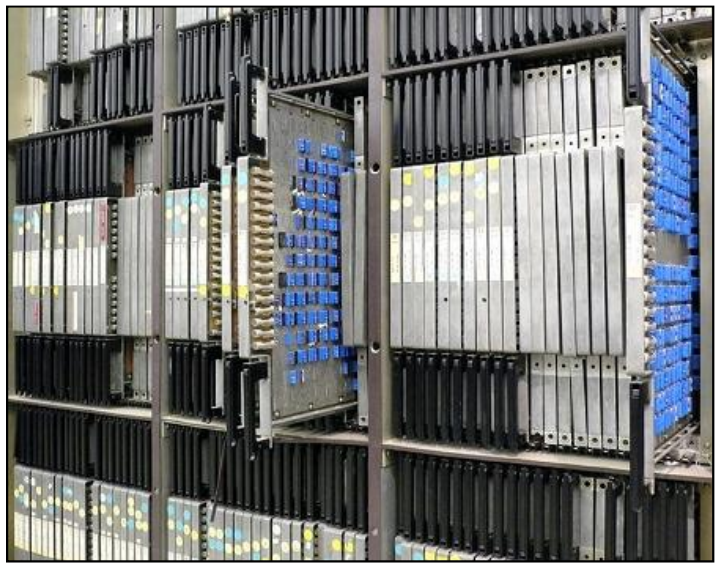
Hitachi S3600 (1994)

DM-SIMD - Tablice procesorów

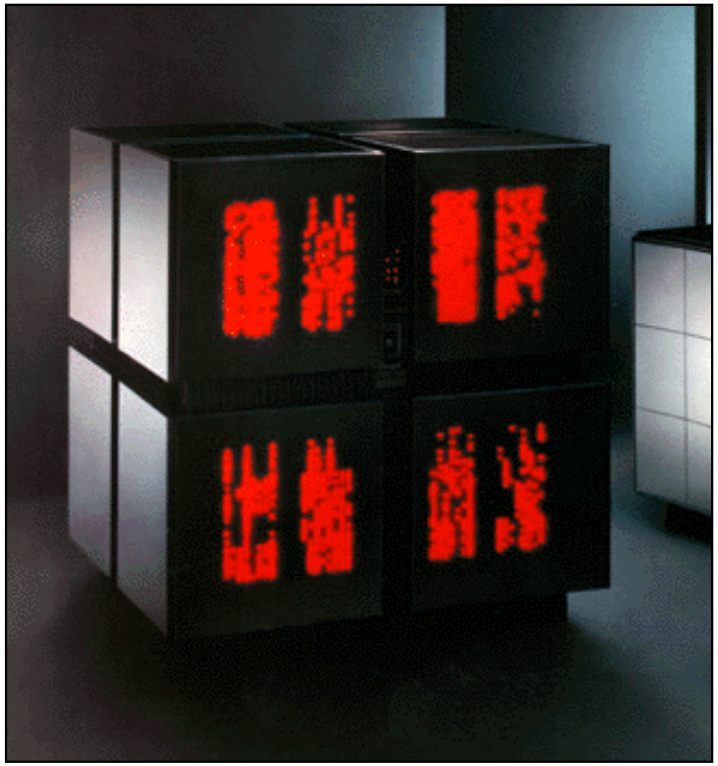
Illiac IV
(1976)



Illiac IV
(1976)



MasPar
MP-1/MP-2
(1990)



Thinking
Machines
CM-2
(1987)

DM-SIMD - Procesory graficzne (GPU)

GeForce RTX 4090

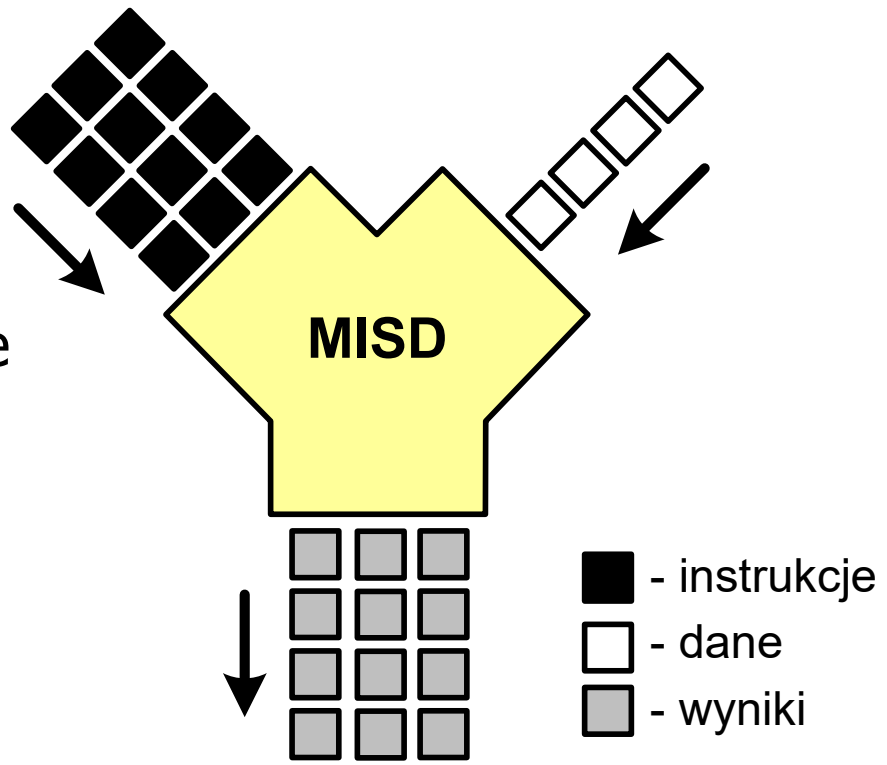
Tesla V100

DGX-1 Volta

Tesla D870

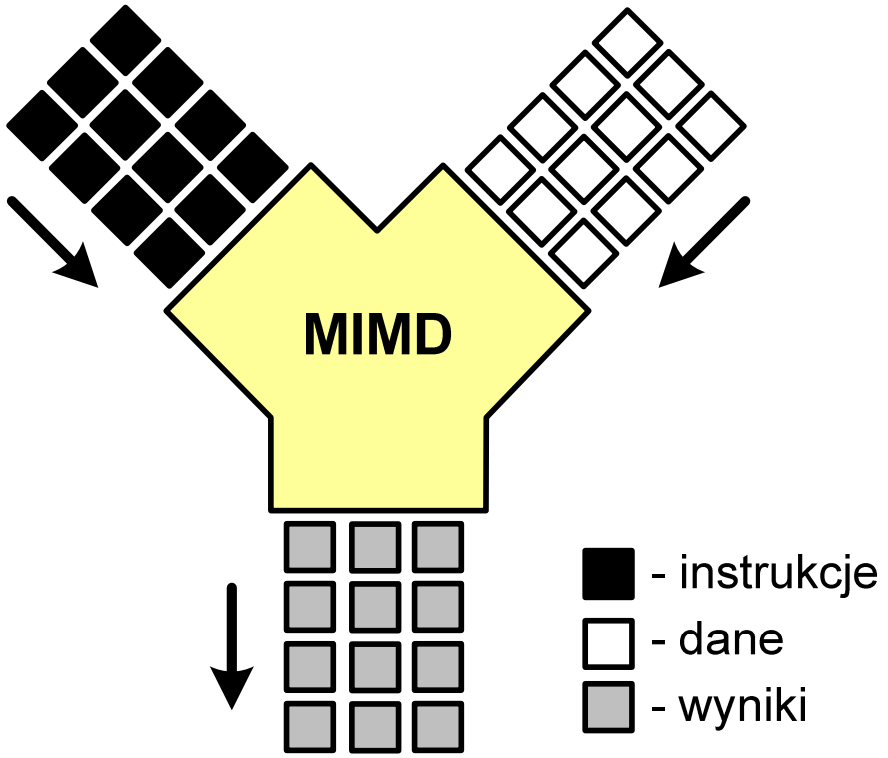
MISD (Multiple Instruction, Single Data)

- Wiele równolegle wykonywanych programów przetwarza jednocześnie jeden wspólny strumień danych
- Systemy tego typu nie są spotykane



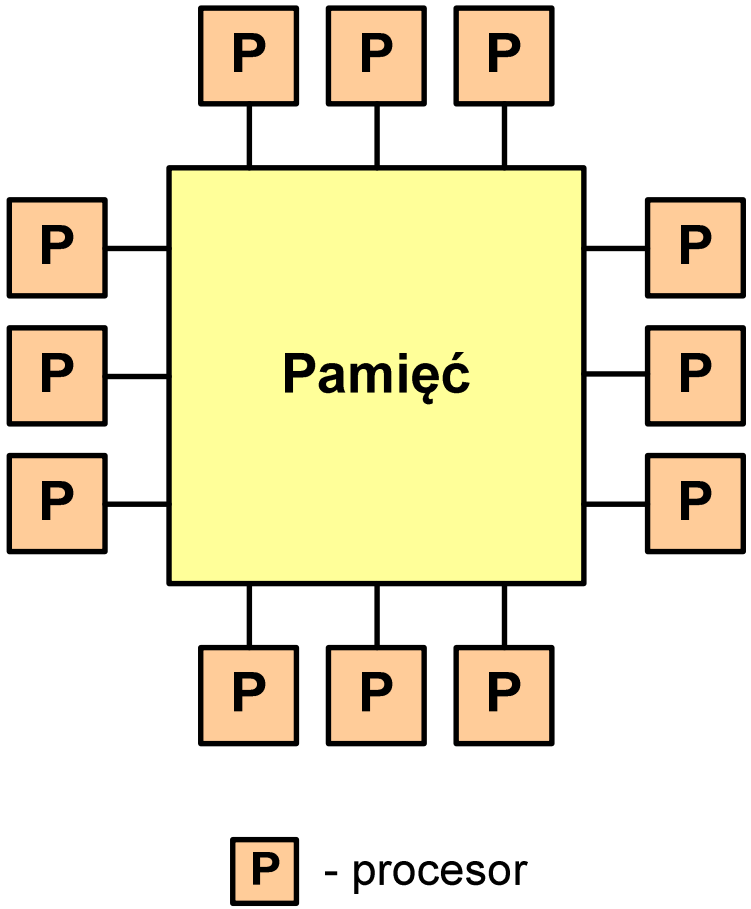
MIMD (Multiple Instruction, Multiple Data)

- Równolegle wykonywanych jest wiele programów, z których każdy przetwarza własne strumienie danych
- Podział:
  - SM-MIMD (Shared Memory):
    - wieloprocesory
  - DM-MIMD (Distributed Memory):
    - wielokomputery
    - klastry
    - gridy



SM-MIMD - Wieloprocesory

- Systemy z niezbyt dużą liczbą działających niezależnie procesorów
- Każdy procesor ma dostęp do wspólnej przestrzeni adresowej pamięci
- Komunikacja procesorów poprzez uzgodniony obszar wspólnej pamięci
- Do SM-MIMD należą komputery z procesorami wielordzeniowymi



SM-MIMD - Wieloprocesory

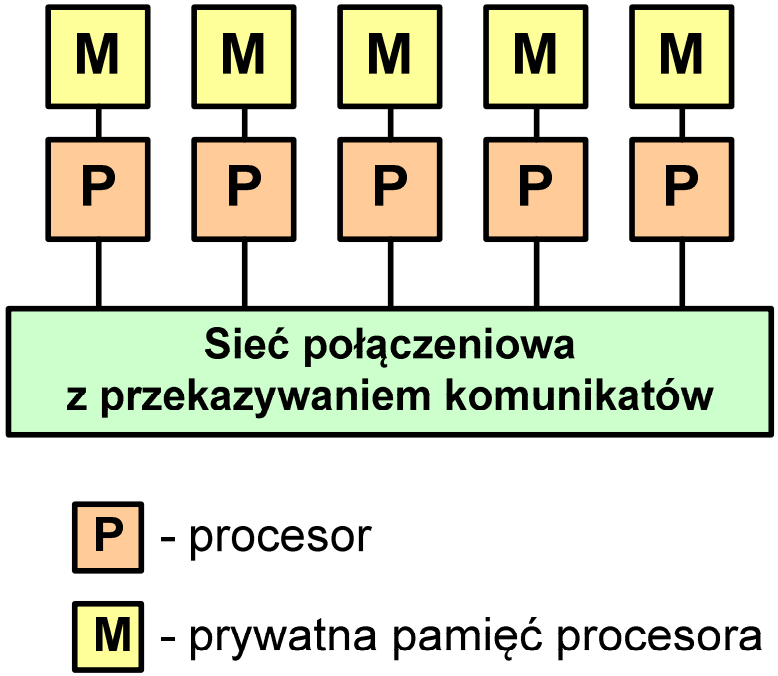
Cray YM-P
(1988)

Cray
CS6400
(1993)

Cray J90
(1994)

DM-MIMD - Wielokomputery

- Każdy procesor wyposażony jest we własną pamięć operacyjną, niedostępną dla innych procesorów
- Komunikacja między procesorami odbywa się za pomocą sieci poprzez przesyłanie komunikatów
- Biblioteki komunikacyjne:
  - MPI (Message Passing Interface)
  - PVM (Parallel Virtual Machine)



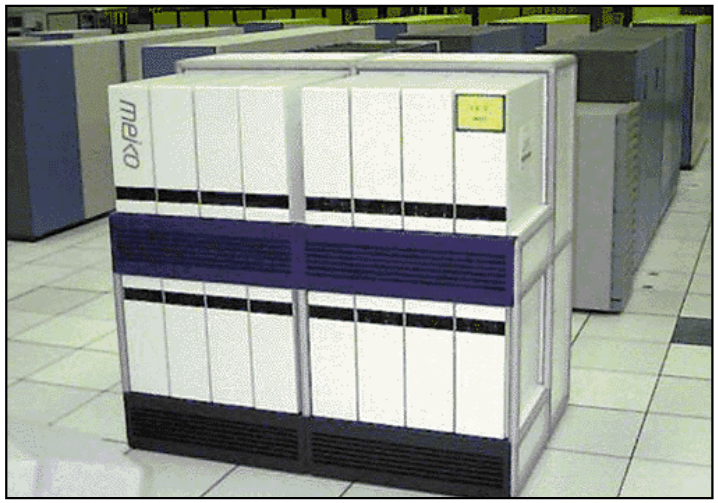
DM-MIMD - Wielokomputery

Cray T3E
(1995)

Thinking Machines CM-5
(1991)

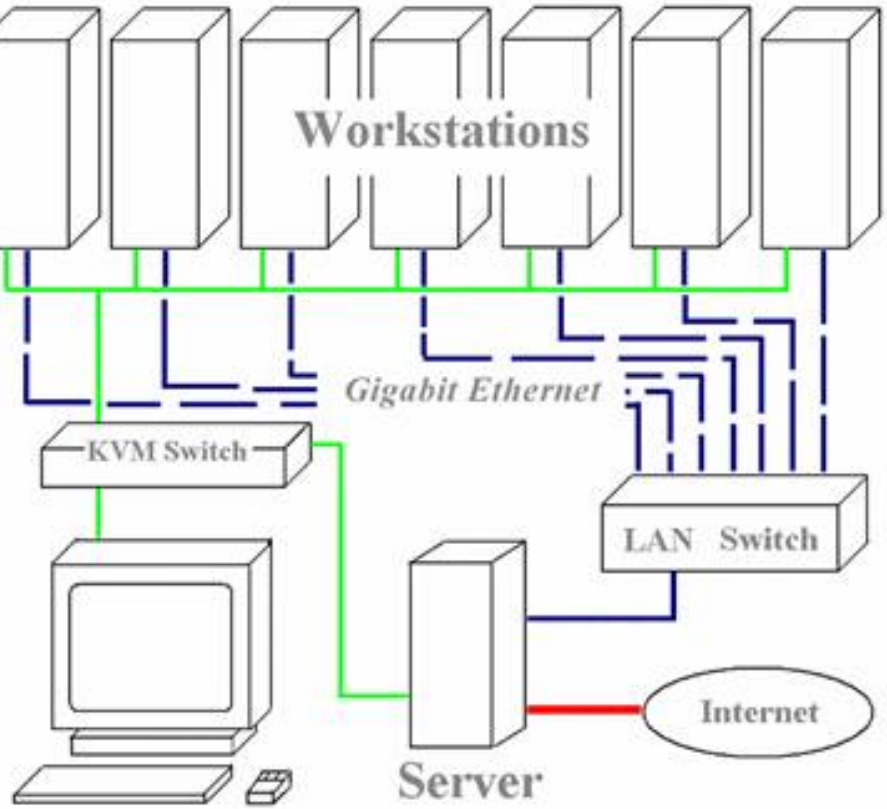
nCube 2s
(1993)

Meiko CS-2
(1993)



DM-MIMD - Klastry

- Klaster (cluster):
  - równoległy lub rozproszonego system składający się z komputerów
  - komputery połączone są siecią
  - używany jest jako pojedynczy, zintegrowany zespół obliczeniowy
- Węzeł (node) - pojedynczy komputer przyłączony do klastra i wykonujący zadania obliczeniowe



źródło:
http://leda.elfak.ni.ac.rs/projects/SeeGrid/see_grid.htm

KVM - Keyboard, Video, Mouse

DM-MIMD - Klastry

- Klastry Beowulf budowane były ze zwykłych komputerów PC

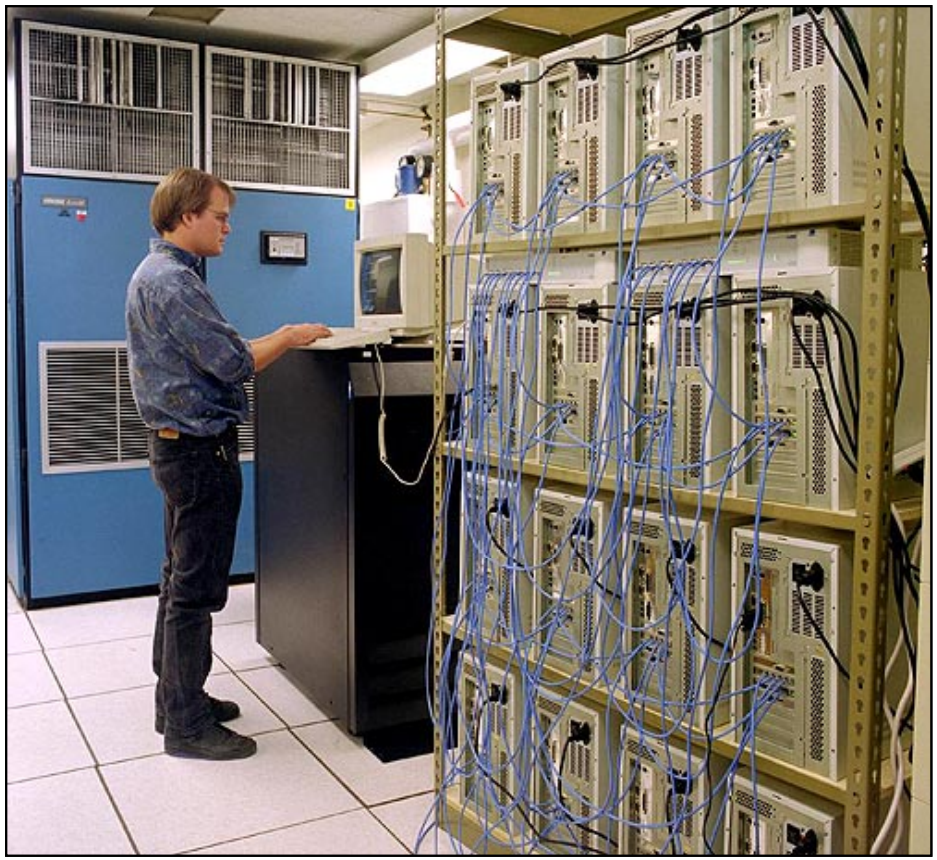
Odin II Beowulf Cluster Layout, University of Chicago, USA

DM-MIMD - Klastry

- Klastry Beowulf budowane były ze zwykłych komputerów PC

NASA 128-processor Beowulf cluster: A cluster built from 64 ordinary PC's

DM-MIMD - Klastry



Early Aspen Systems Beowulf Cluster With RAID

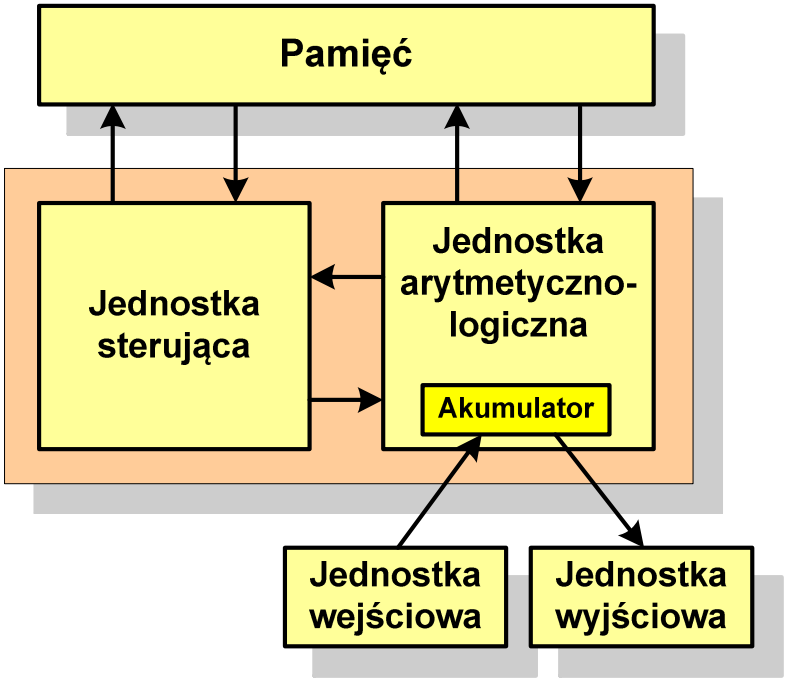
DM-MIMD - Klastry

- Obecnie klastry też są bardzo popularnym typem systemów

SuperMUC-NG, Leibniz Rechenzentrum, Germany

Architektura von Neumanna

- Rodzaj architektury komputera, opisanej w 1945 roku przez matematyka Johna von Neumanna
- Inne spotykane nazwy: architektura z Princeton, store-program computer (koncepcja przechowywanego programu)
- Zakłada podział komputera na kilka części:
  - jednostka sterująca (CU - Control Unit)
  - jednostka arytmetyczno-logiczna (ALU - Arithmetic Logic Unit)
  - pamięć główna (memory)
  - urządzenia wejścia-wyjścia (input/output)

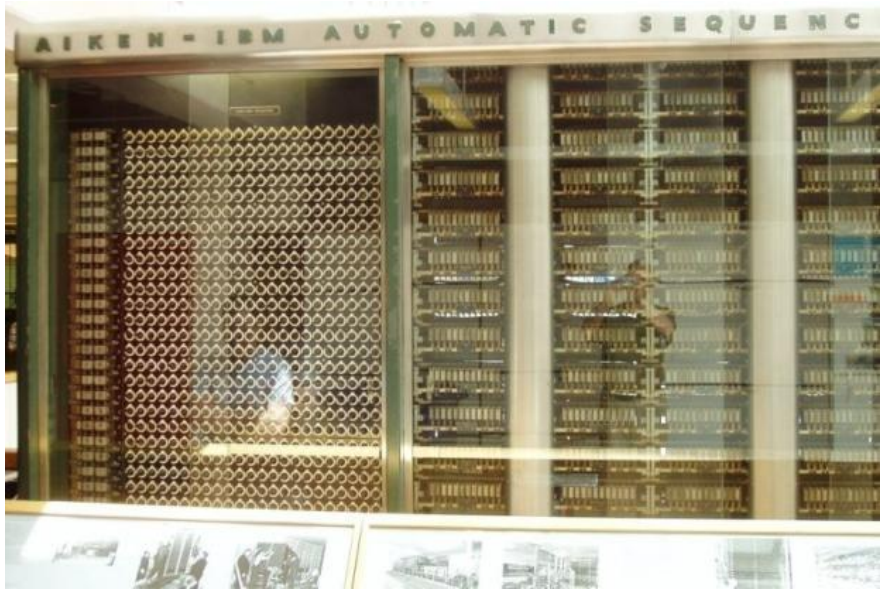


Architektura von Neumanna - podstawowe cechy

- Informacje przechowywane są w komórkach pamięci (cell) o jednakowym rozmiarze, każda komórka ma numer - adres
- Dane oraz instrukcje programu (rozkazy) zakodowane są za pomocą liczb i przechowywane w tej samej pamięci
- Praca komputera to sekwencyjne odczytywanie instrukcji z pamięci komputera i ich wykonywanie w procesorze
- Wykonanie rozkazu:
  - pobranie z pamięci słowa będącego kodem instrukcji
  - pobranie z pamięci danych
  - wykonanie instrukcji
  - zapisanie wyników do pamięci
- Dane i instrukcje czytane są przy wykorzystaniu tej samej magistrali

Architektura harwardzka

- Architektura komputera, w której pamięć danych jest oddzielona od pamięci instrukcji
- Nazwa architektury pochodzi komputera Harward Mark I:
  - zaprojektowany przez Howarda Aikena
  - pamięć instrukcji - taśma dziurkowana,
    pamięć danych - elektromechaniczne liczniki

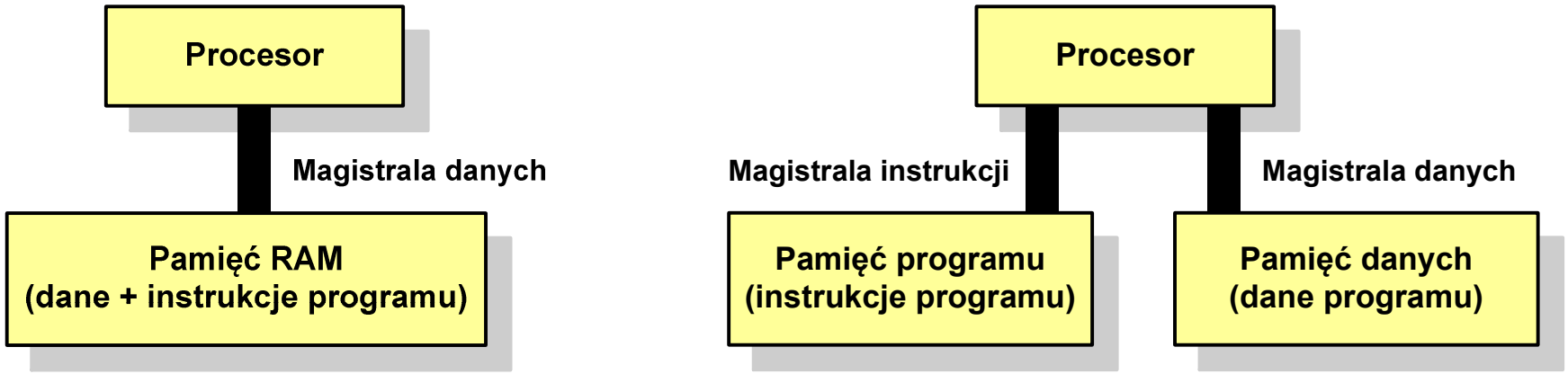


Architektura harwardzka

- Pamięci danych i instrukcji mogą różnić się:
  - technologią wykonania
  - strukturą adresowania
  - długością słowa
- Przykład:
  - ATmega16 - 16 kB Flash, 1 kB SRAM, 512 B EEPROM
- Procesor może w tym samym czasie czytać instrukcje oraz uzyskiwać dostęp do danych

Architektura harwardzka i von Neumanna

- W architekturze harwardzkiej pamięć instrukcji i pamięć danych:
  - zajmują różne przestrzenie adresowe
  - mają oddzielne szyny (magistrale) do procesora
  - zaimplementowane są w inny sposób

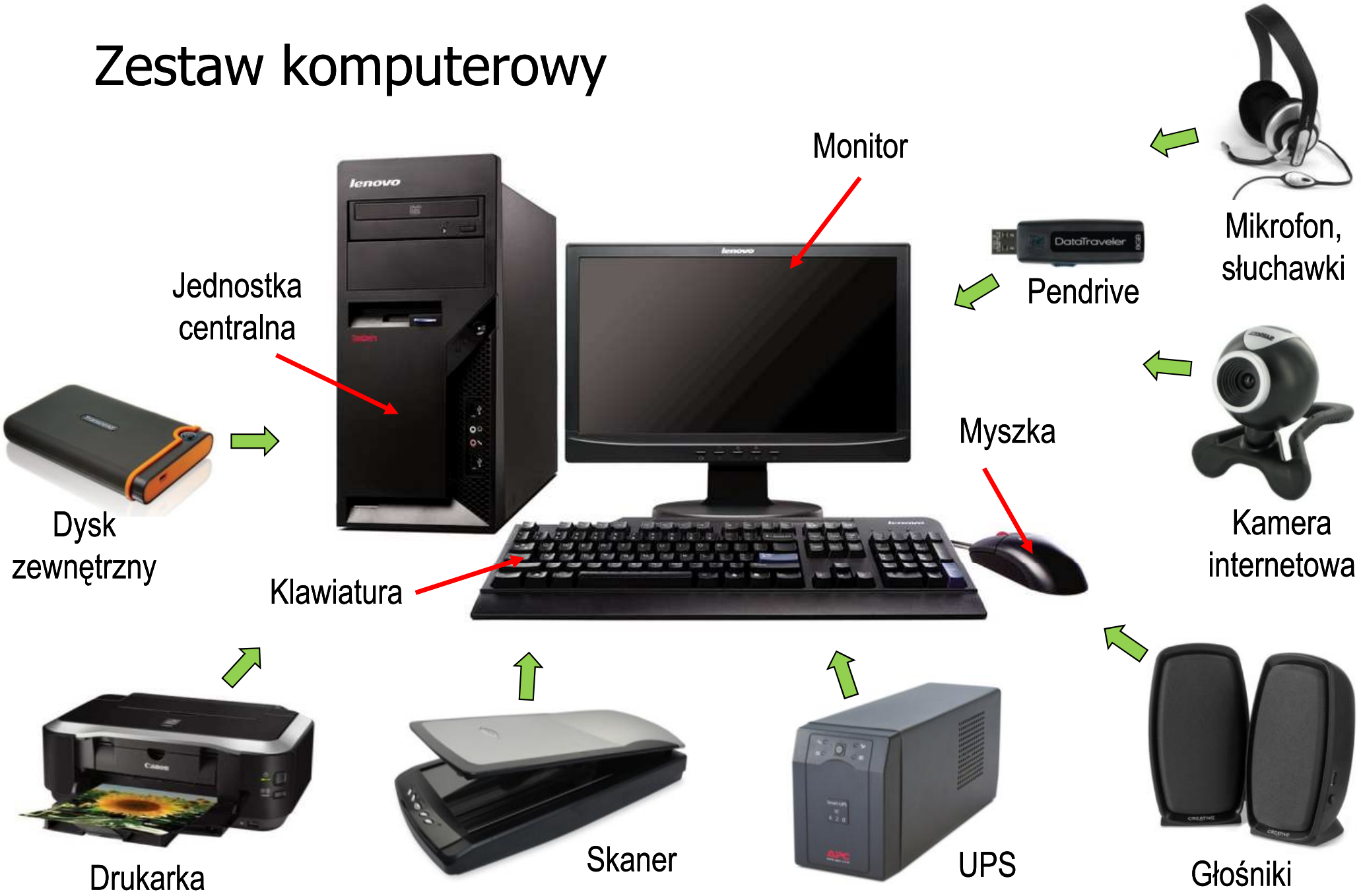


Architektura von Neumanna

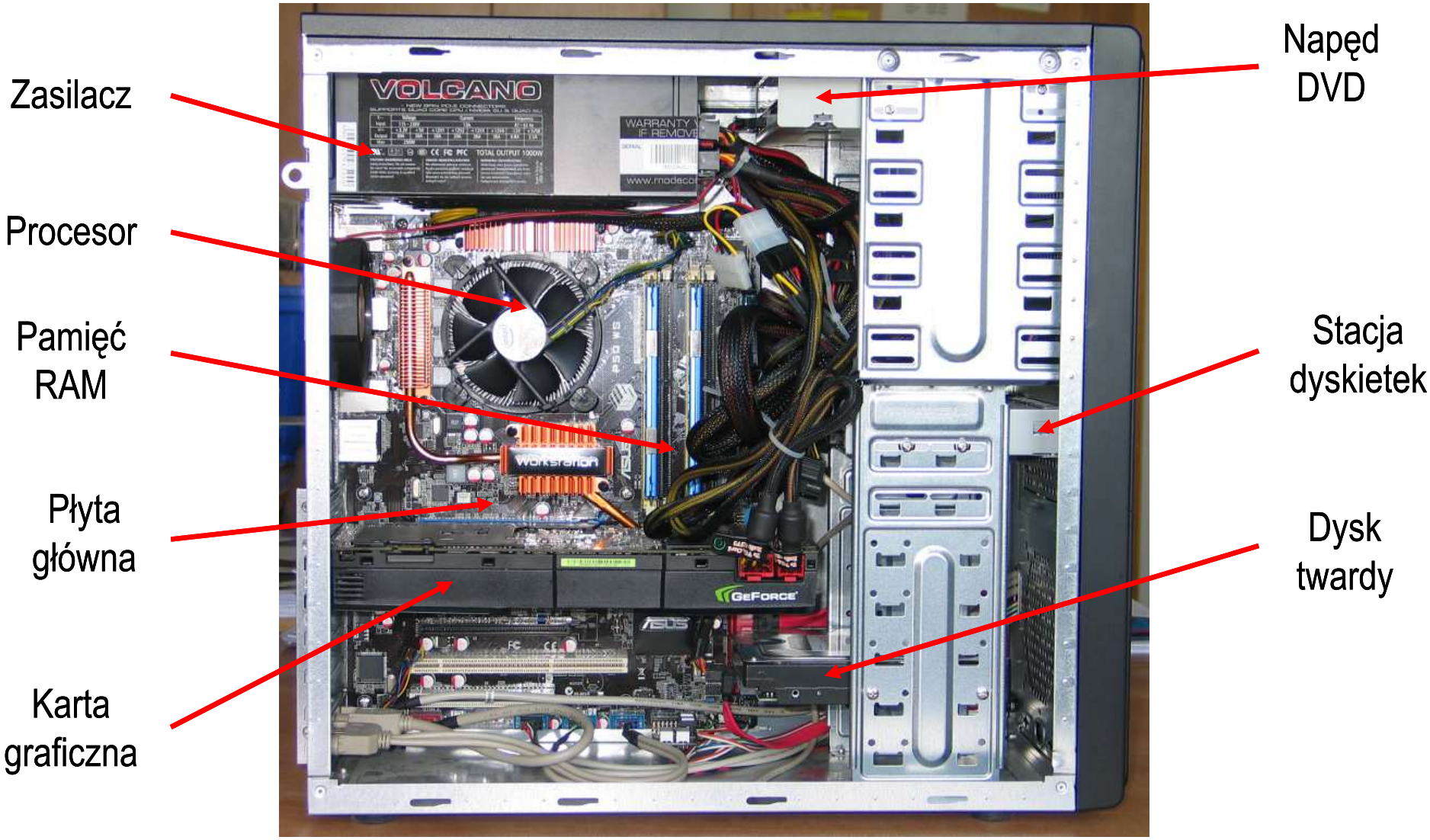
Architektura harwardzka

- Zmodyfikowana architektura harwardzka:
  - oddzielone pamięci danych i rozkazów, lecz wykorzystujące wspólną magistralę

Zestaw komputerowy



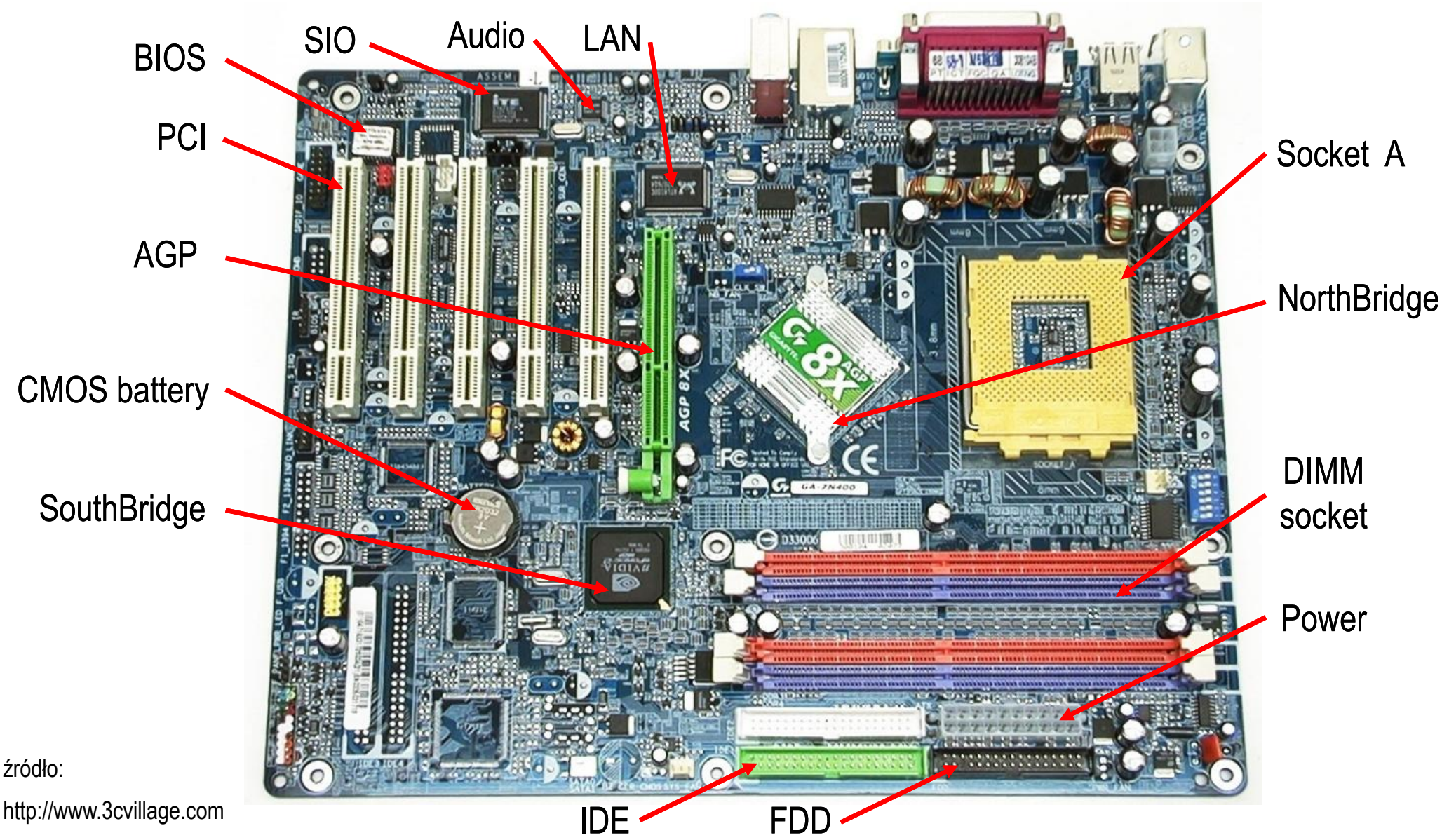
Jednostka centralna



Płyta główna (motherboard) - przykłady

Model	Gigabyte GA-7N400-L	Gigabyte GA-X58A-UD5	Gigabyte G1-Assassin 2
Rok	2003	2009	2011
Gniazdo	Socket A	Socket 1366	Socket 2011
Procesor	AMD Athlon, Athlon XP	Intel Core i7	Intel Core i7
Northbridge	nVIDIA nForce 2 Ultra 400	Intel X58 Express Chipset	Intel X79
Southbridge	nVIDIA nForce 2 MCP	Intel ICH10R	
Pamięć	4 x 184-pin DDR DIMM sockets, max. 3 GB	6 x 1.5V DDR3 DIMM sockets, max. 24 GB	4 x 1.5V DDR3 DIMM sockets, max. 32 GB
Format	ATX	ATX	ATX
Inne	AGP, 5 × PCI, 2 × IDE, FDD, LPT, 2 × COM, 6 × USB, IrDA, RJ45, 2 × PS/2	4 × PCIe x16, 2 × PCIe x1, PCI, 8 × SATA II 3 Gb/s, 2 × SATA II 6 Gb/s, 2 × eSATA, IDE, FDD, 2 × RJ45, 10 × USB 2.0, 2 × USB 3.0, 2 × PS/2	3 × PCIe x16, 2 × PCIe x1, PCI, 4 × SATA II 3 Gb/s, 4 × SATA III 6 Gb/s, 2 × eSATA, RJ45, 9 × USB 2.0, 3 × USB 3.0, PS/2

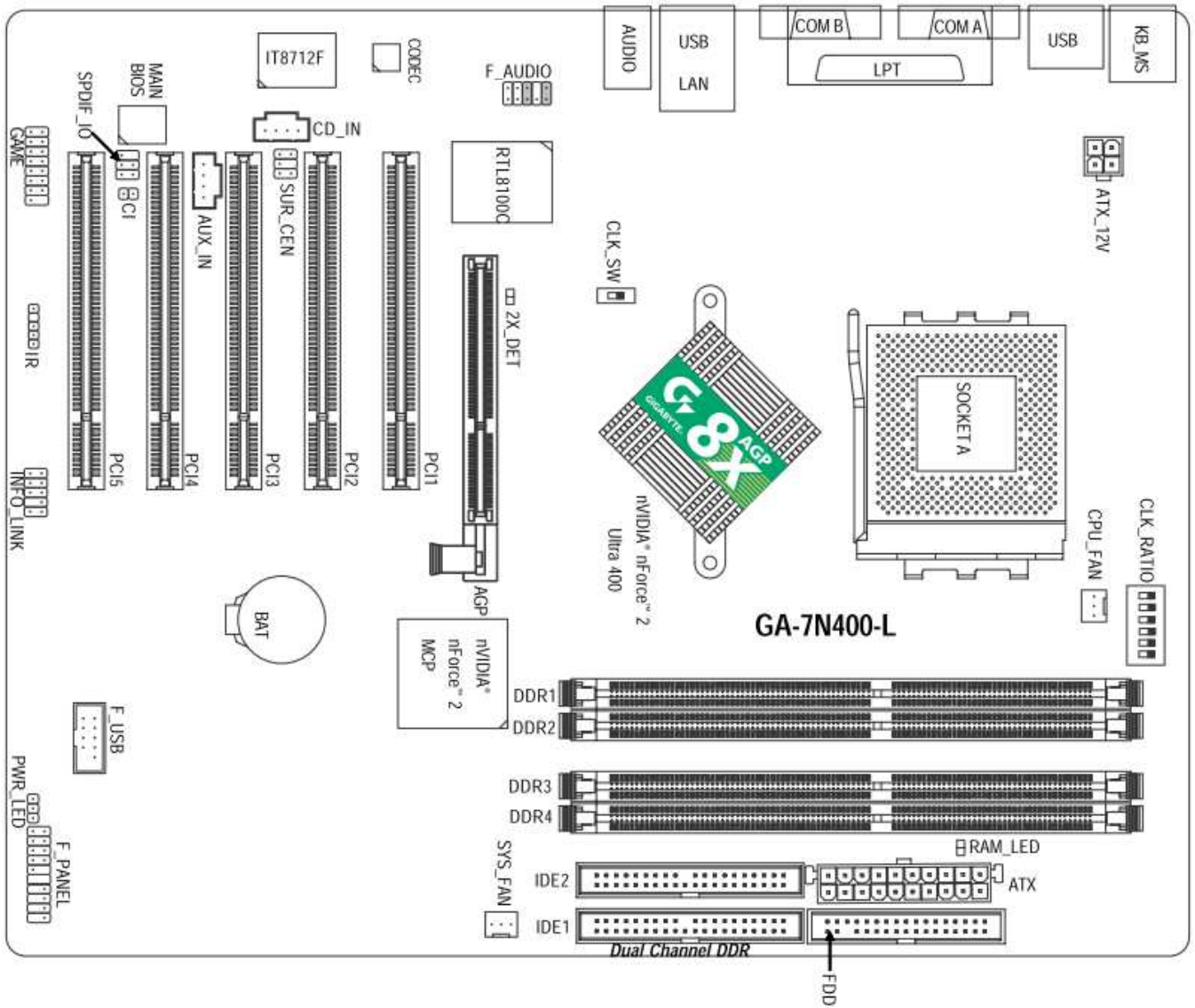
Gigabyte GA-7N400-L



źródło:

http://www.3cvillage.com

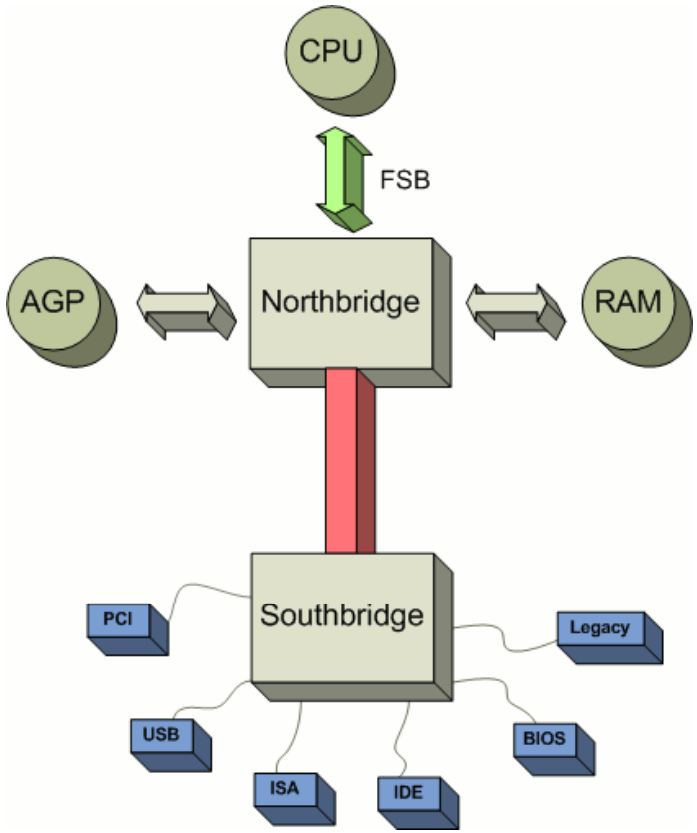
Gigabyte GA-7N400-L

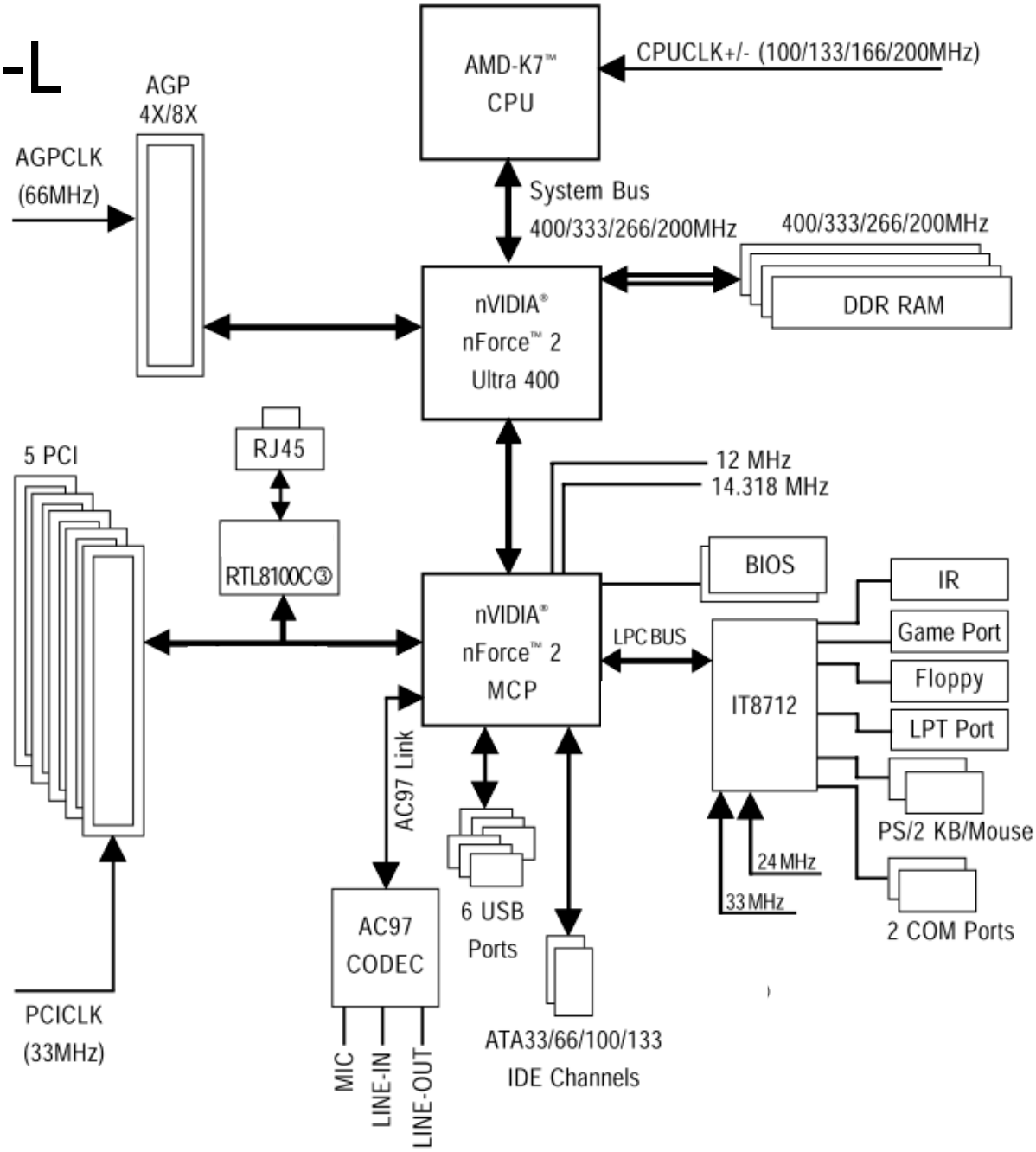


źródło:

GA-7N400 Pro2 / GA-7N400 /
GA-7N400-L
AMD Socket A
Processor Motherboard
User's Manual

Gigabyte GA-7N400-L

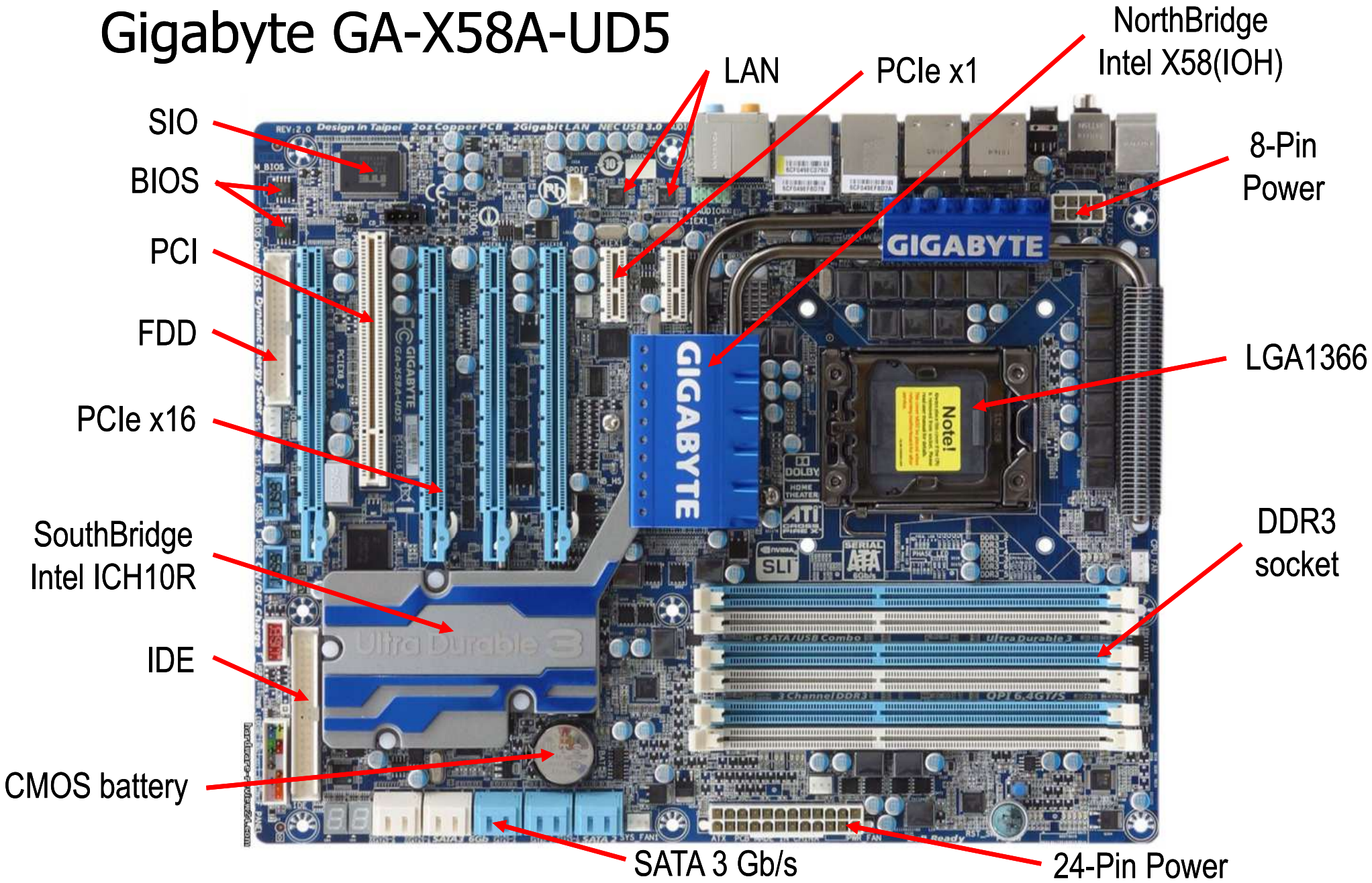




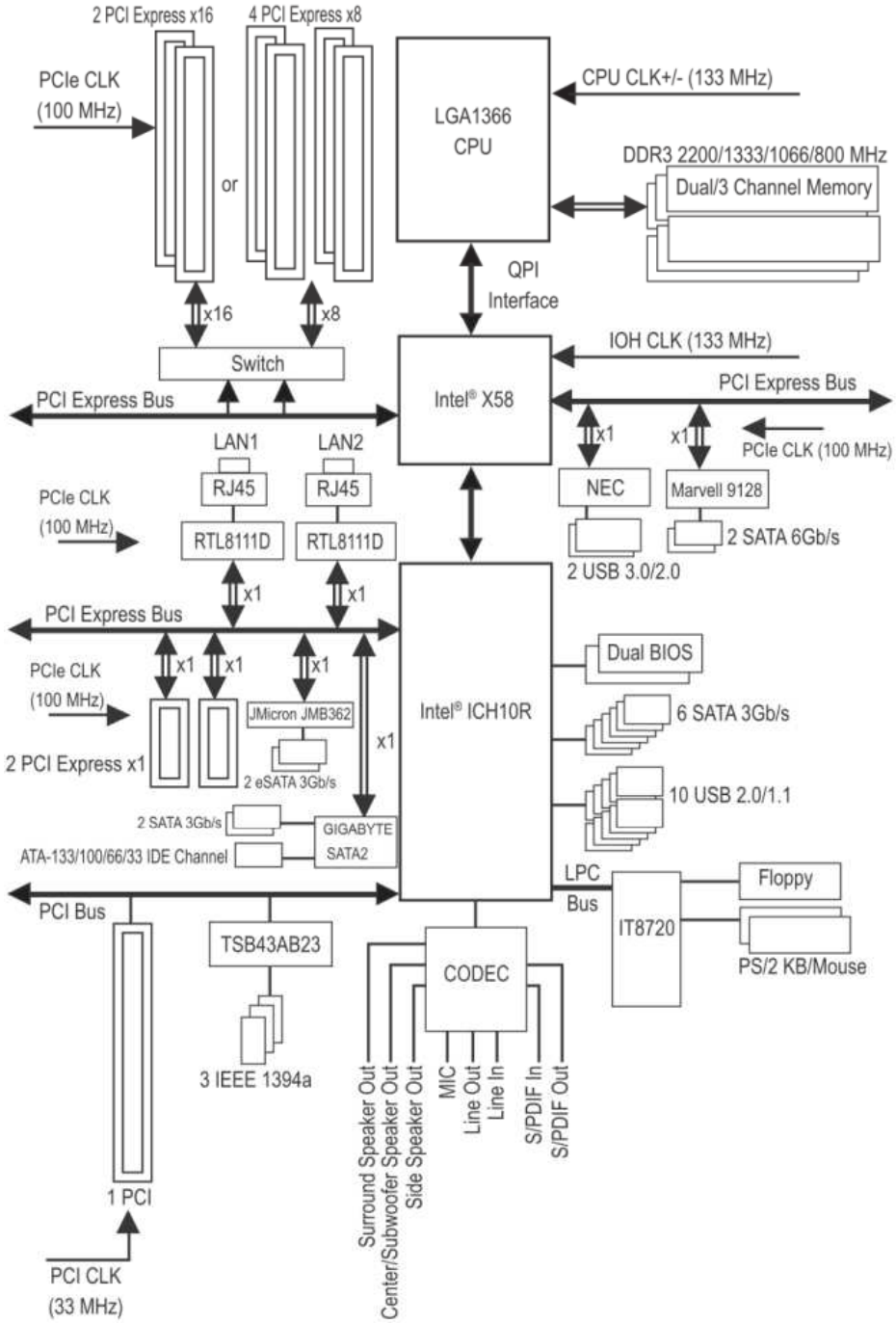
źródło:

GA-7N400 Pro2 / GA-7N400 / GA-7N400-L
AMD Socket A Processor Motherboard
User's Manual

Gigabyte GA-X58A-UD5



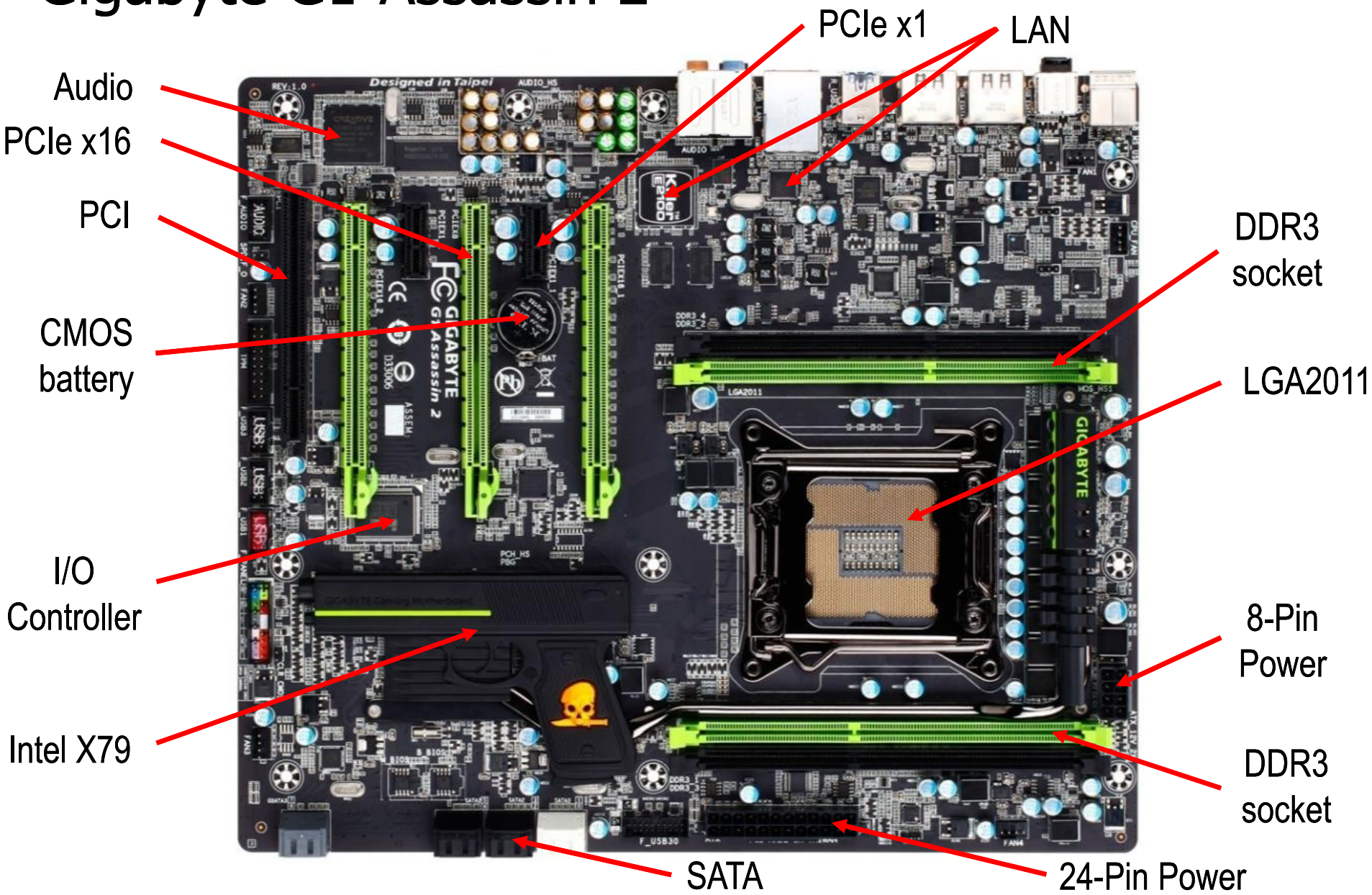
Gigabyte GA-X58A-UD5



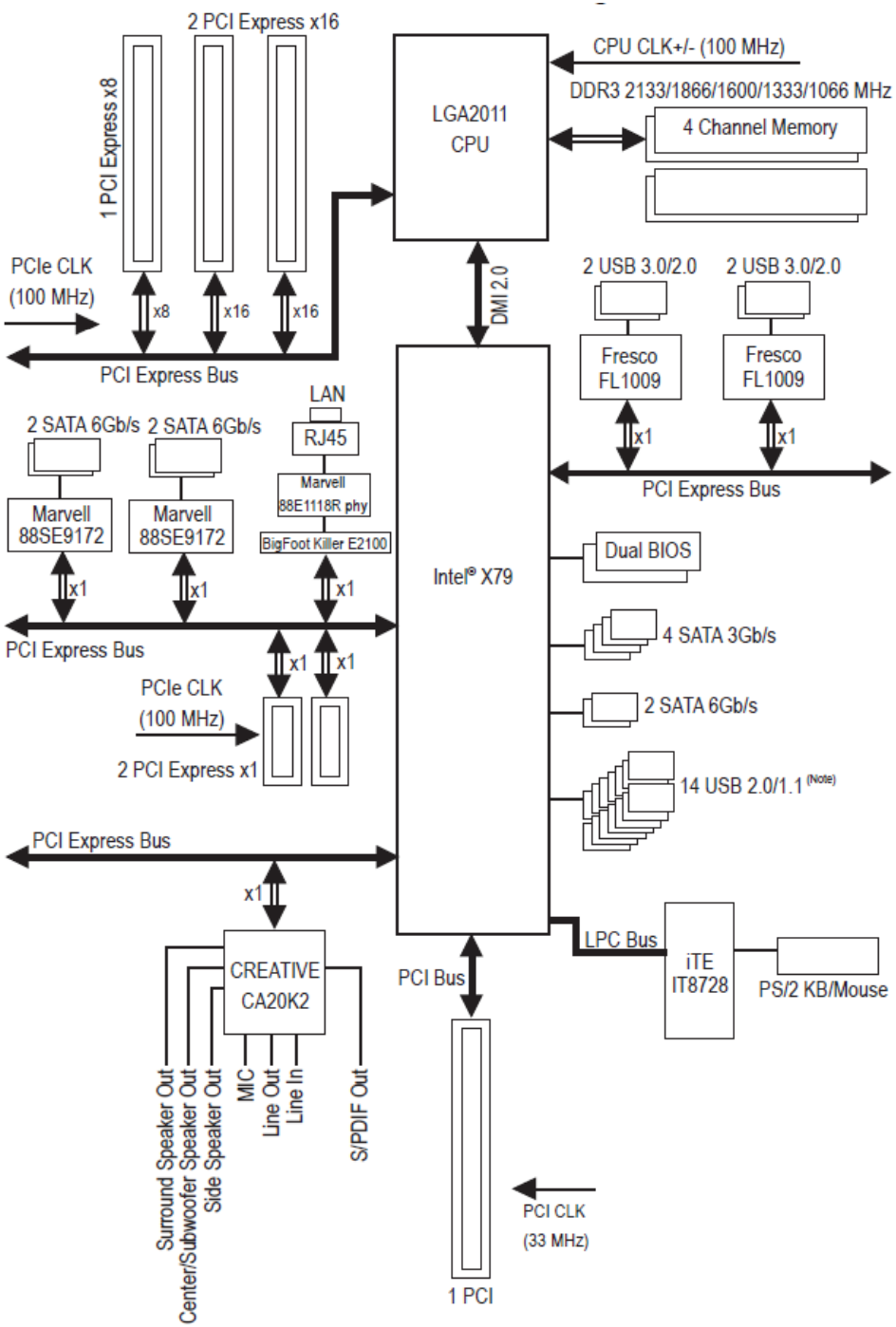
źródło:

GA-X58A-UD5
LGA1366 socket motherboard for Intel ® Core ™ i7 processor family
User's Manual

Gigabyte G1-Assassin 2



Gigabyte G1-Assassin 2



źródło:

Gigabyte G1.Assassin 2, User's Manual, Rev. 1001

Płyty główne - standardy

Standard	Rok	Wymiary
AT	1984 (IBM)	12 × 11–13 in 305 × 279–330 mm
Baby-AT	1985 (IBM)	8.5 × 10–13 in 216 × 254–330 mm
ATX	1996 (Intel)	12 × 9.6 in 305 × 244 mm
Micro-ATX	1996	9.6 × 9.6 in 244 × 244 mm
Mini-ITX	2001 (VIA)	6.7 × 6.7 in 170 × 170 mm max.
Nano-ITX	2003 (VIA)	4.7 × 4.7 in 120 × 120 mm
Pico-ITX	2007 (VIA)	100 × 72 mm max.

źródło: http://en.wikipedia.org

Płyty główne - standardy

Standard-ATX

Micro-ATX

Mini-ITX

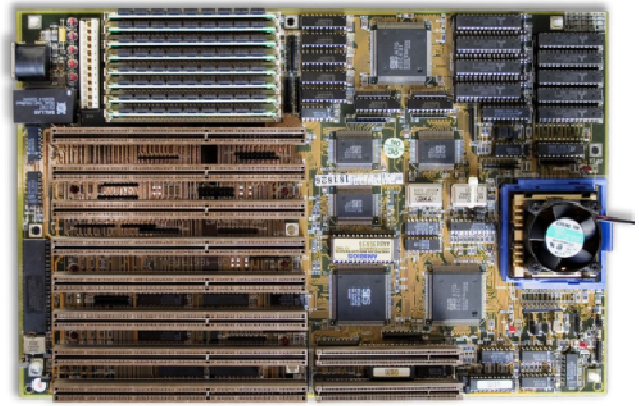
Nano-ITX

Pico-ITX

ATX (Advanced Technology Extended)

źródło:

http://en.wikipedia.org



Baby-AT

AT (Advanced Technology)

Koniec wykładu nr 4

Dziękuję za uwagę!