Informatyka 1

Politechnika Białostocka - Wydział Elektryczny

Elektrotechnika, semestr II, studia stacjonarne I stopnia

Rok akademicki 2018/2019

Wykład nr 10 (17.05.2019)

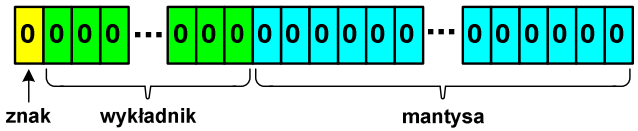
dr inż. Jarosław Forenc

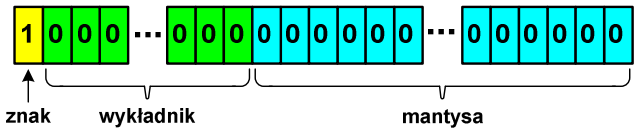
Plan wykładu nr 10

- Standard IEEE 754
  - operacje z wartościami specjalnymi
- Klasyfikacja systemów komputerowych (Flynna)
- Architektura von Neumanna i architektura harwardzka

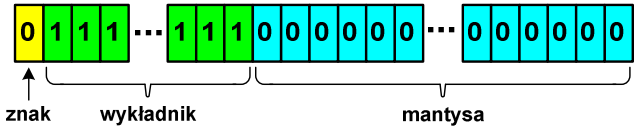
Standard IEEE 754 - wartości specjalne

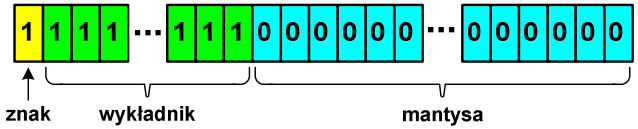
- Zero



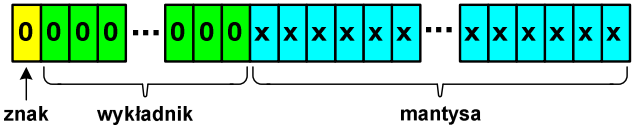


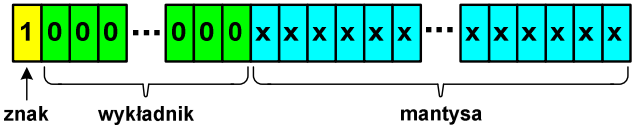
- Nieskończoność



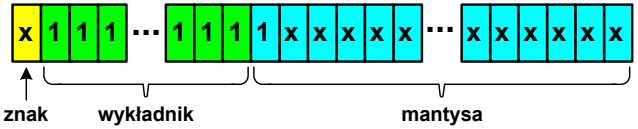


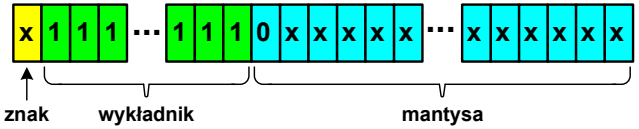
- Liczba zdenormalizowana





- Nieliczba (NaN)





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()
{
  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
```

Operacja	Wynik
$x / \pm\infty$	0
$\pm\infty \cdot \pm\infty$	$\pm\infty$
$\pm$wart_niezer / 0	$\pm\infty$
$\infty + \infty$	∞
$\pm 0 / \pm 0$	NaN
$\infty - \infty$	NaN
$\pm\infty / \pm\infty$	NaN
$\pm\infty \cdot 0$	NaN

- Środowisko: Microsoft Visual C++ 2008 Express Edition

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
Dev-C++	12 bajtów

Reprezentacja liczb zmiennoprzecinkowych w C

```
#include <stdio.h>

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

    for (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)

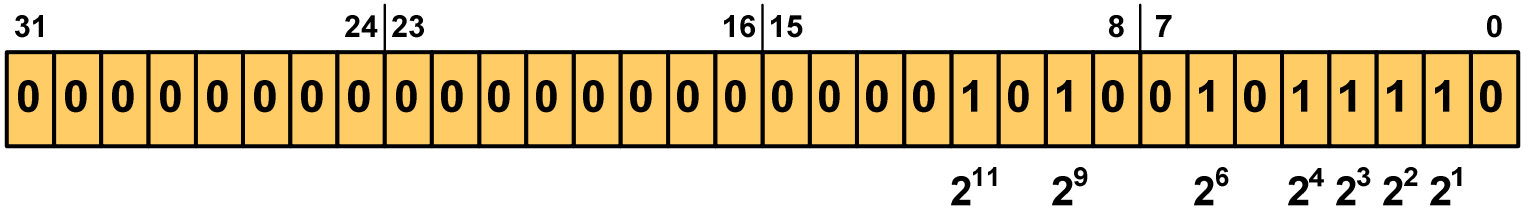
```
float:       100.00295257568359375000
double:      100.00000000001425349000
long double: 100.00000000000001388000
```

- Dev-C++ (long double - 12 bajtów)

```
float:       100.00295257568359000000
double:      100.00000000001425000000
long double: -6805647338419354100000000000000000000.0000000000000
```

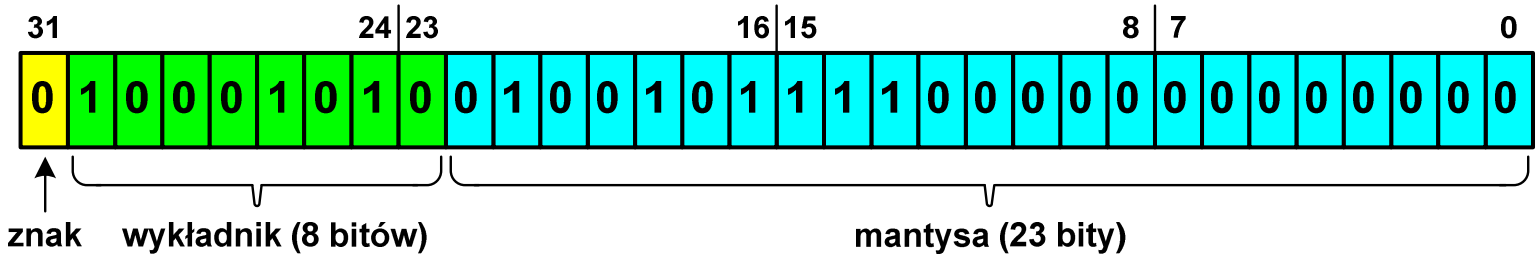
Liczba $2654_{(10)}$ jako całkowita i rzeczywista w C

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



$+ \quad 138 - 127 = 11_{(10)}$

$$1\ .0100101111_{(2)} = 1.2958984_{(10)}$$

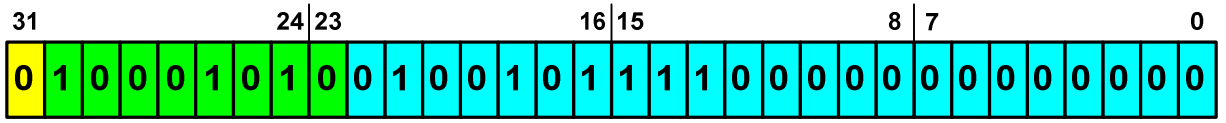
$$1.2958984 \cdot 2^{11} = 2654_{(10)}$$

Język C - nieprawidłowy specyfikator formatu

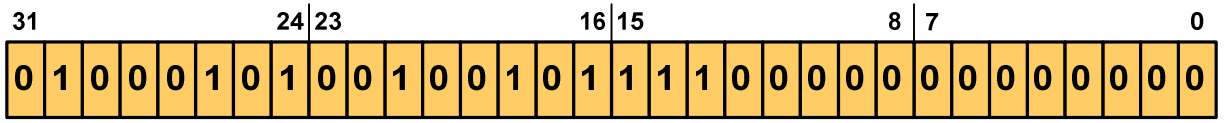
```
int x;
printf("x (%%f) = "); scanf("%f",&x);
printf("x (%%d) = %d\n",x);
printf("x (%%f) = %f\n",x);
printf("x (%%e) = %e\n",x);
```

```
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:



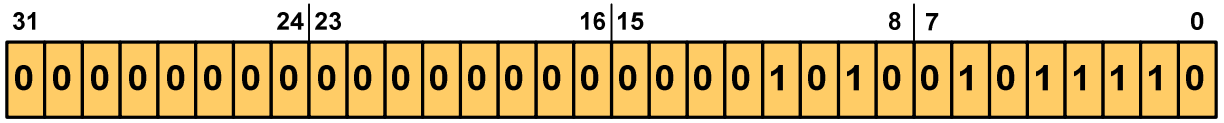
$$2^{30} + 2^{26} + 2^{24} + 2^{21} + 2^{18} + 2^{16} + 2^{15} + 2^{14} + 2^{13} = 1.160.110.080_{(10)}$$

Język C - nieprawidłowy specyfikator formatu

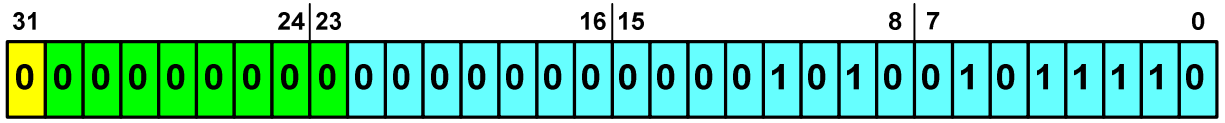
```
float x;
printf("x (%%d) = "); scanf("%d",&x);
printf("x (%%d) = %d\n",x);
printf("x (%%f) = %f\n",x);
printf("x (%%e) = %e\n",x);
```

```
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:

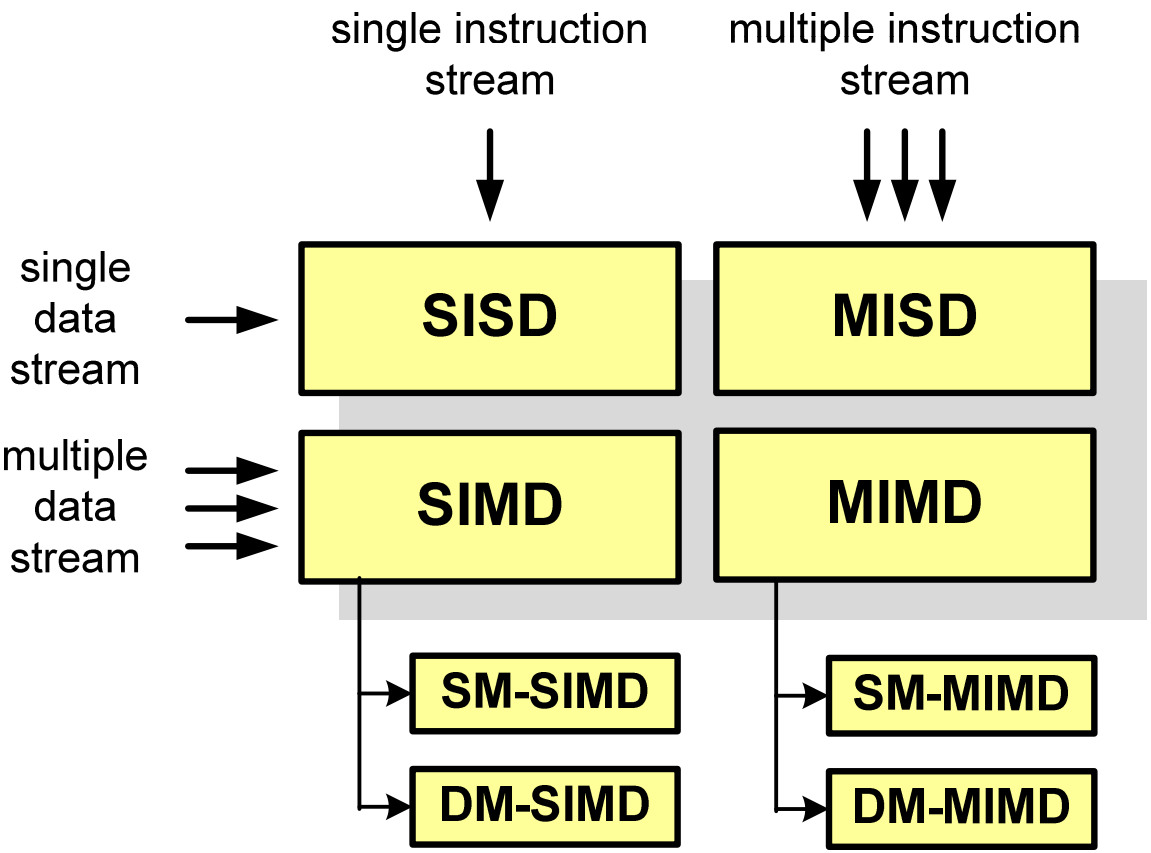


Liczba zdenormalizowana: 3,719046E-42

Klasyfikacja systemów komputerowych

- Taksonomia Flynna - pierwsza, najbardziej ogólna klasyfikacja architektur komputerowych (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



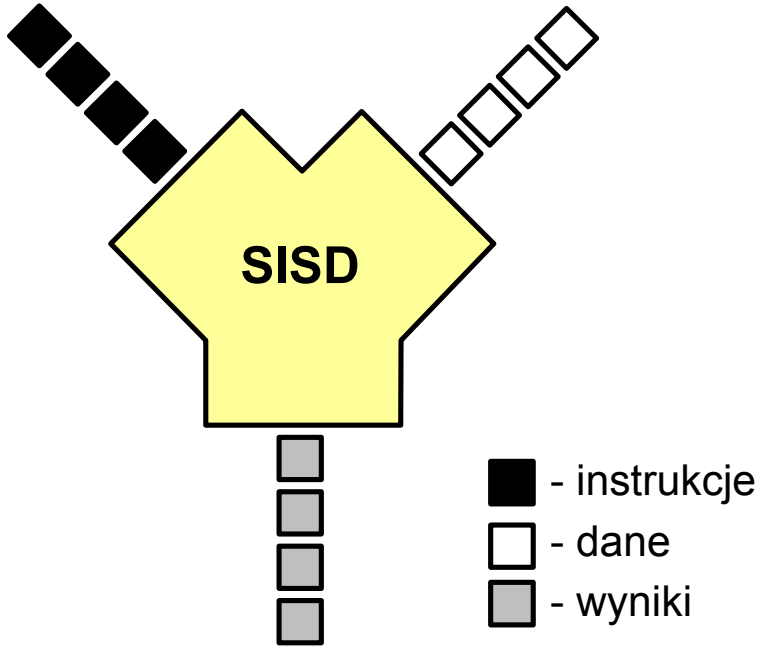
SI - Single instruction
MI - Multiple Instruction

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

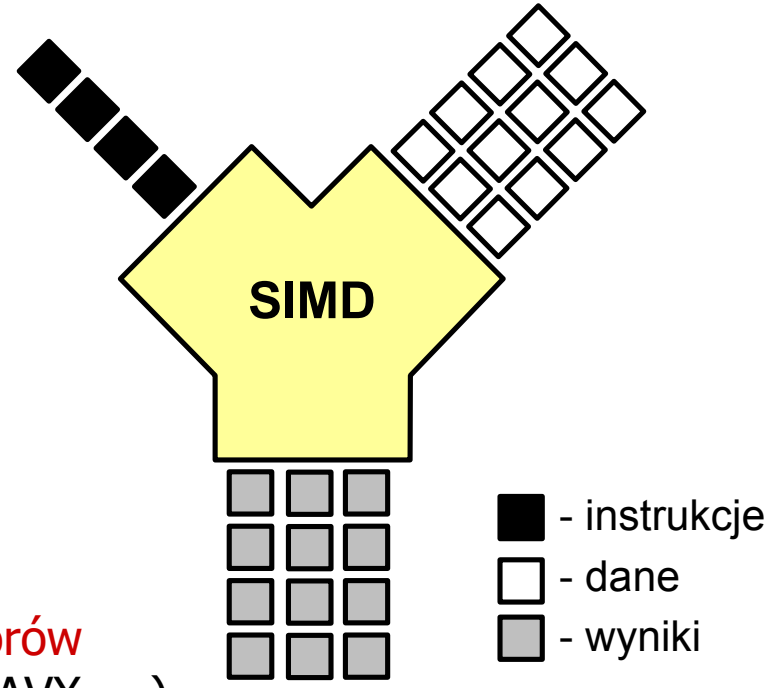
Komputer PC

Komputer PC

Laptop

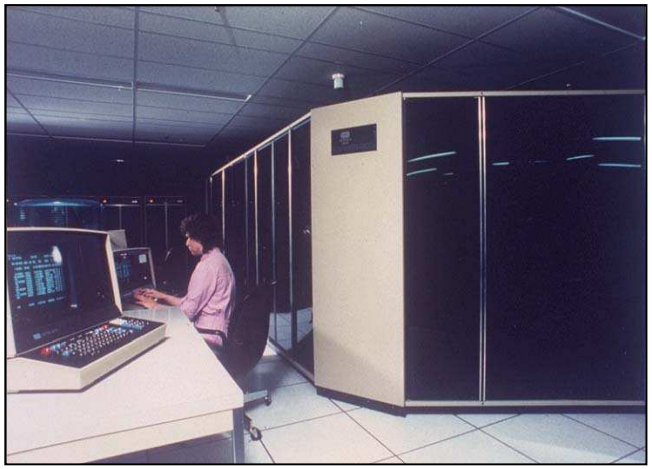
SIMD (Single Instruction, Multiple Data)

- 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)

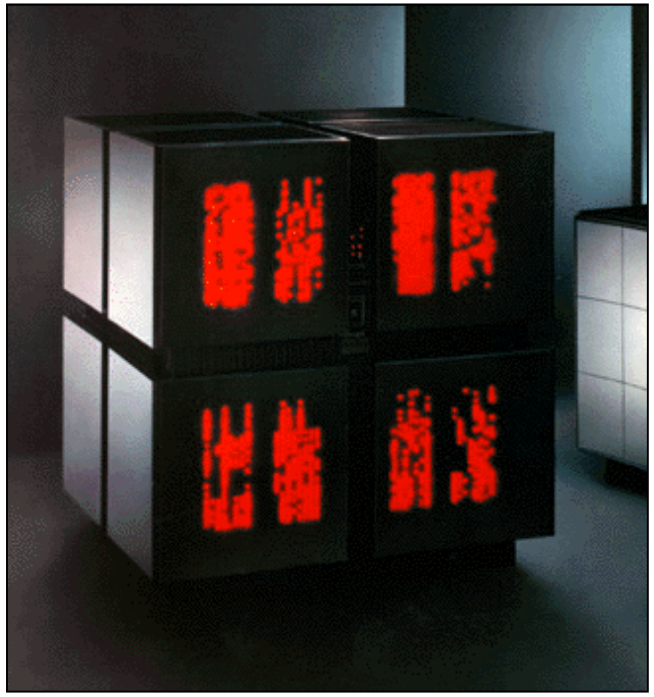
Hitachi S3600 (1994)

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
GTX Titan X

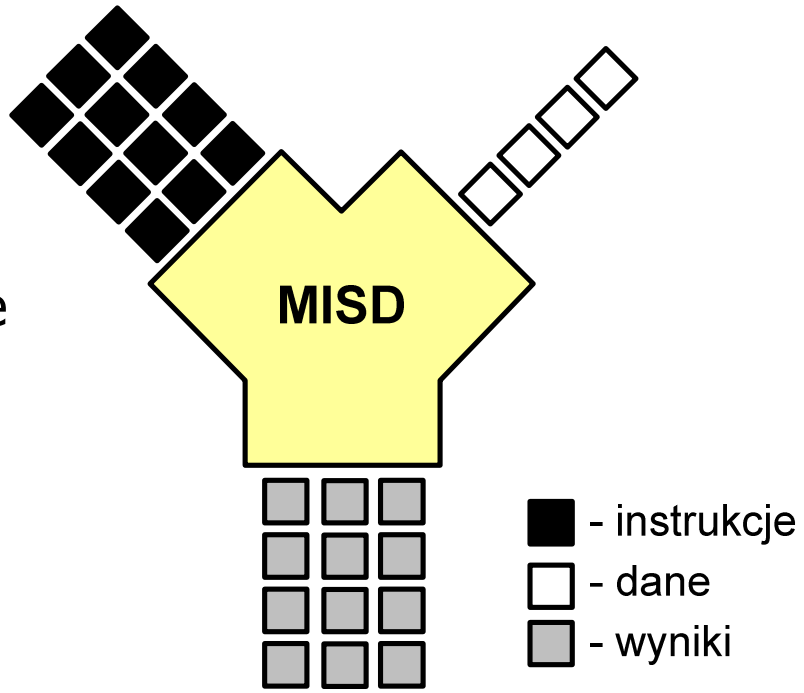
Tesla
V100

DGX-1
Volta

Tesla
D870

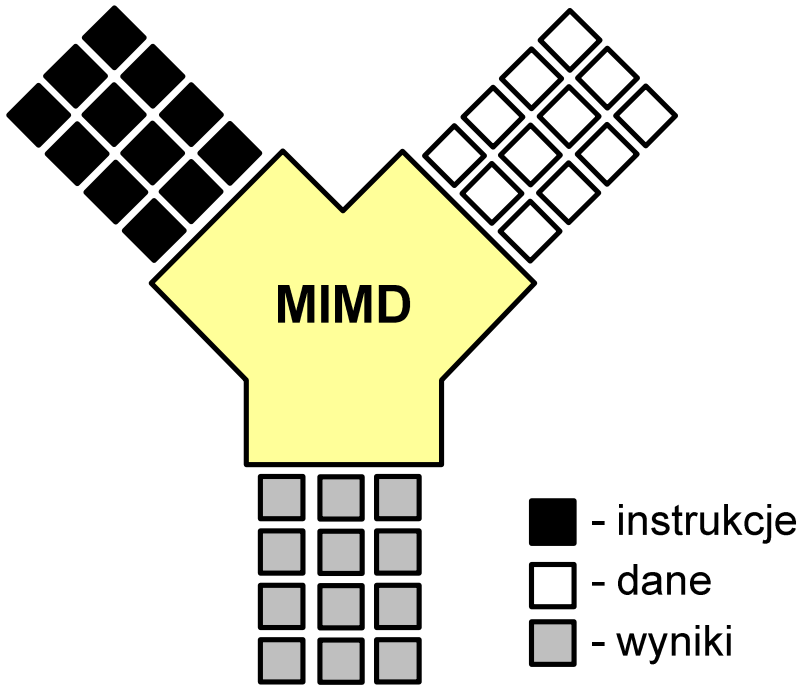
MISD (Multiple Instruction, Single Data)

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



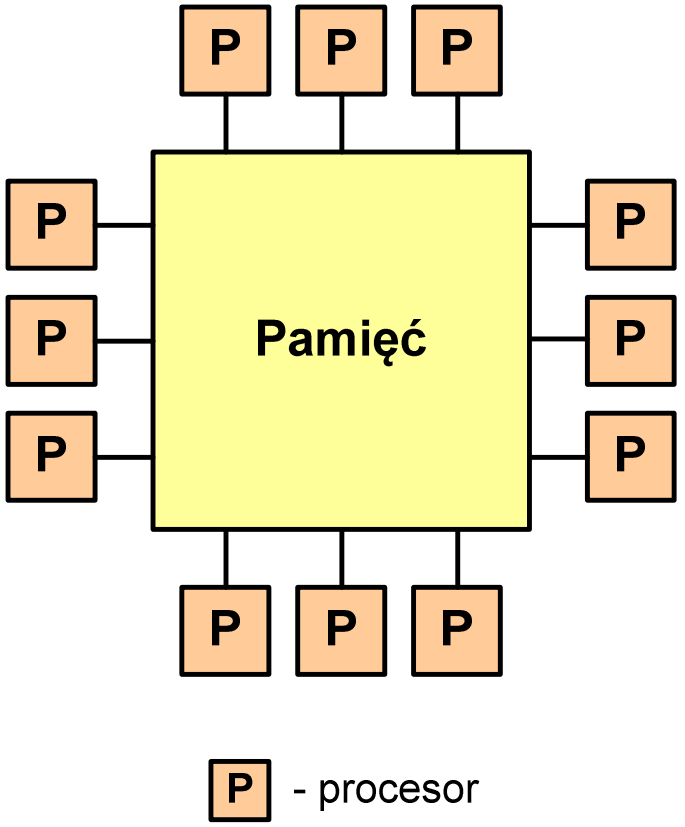
MIMD (Multiple Instruction, Multiple Data)

- 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

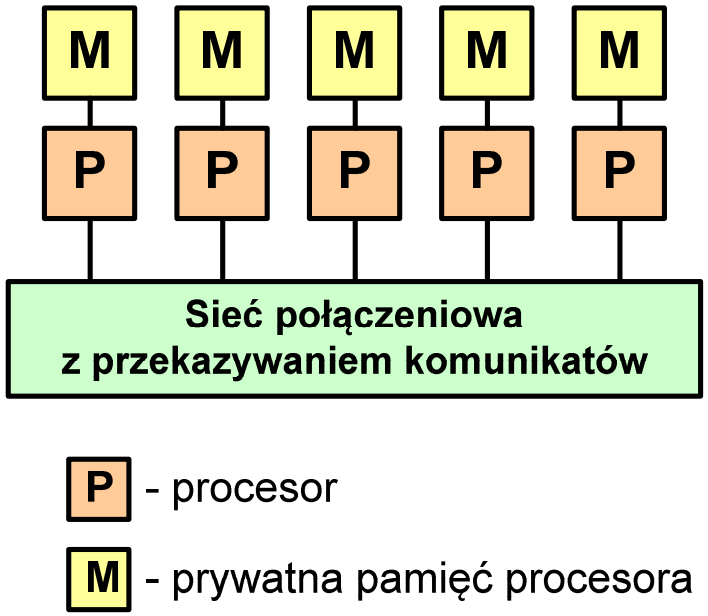
Cray YM-P
(1988)

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)



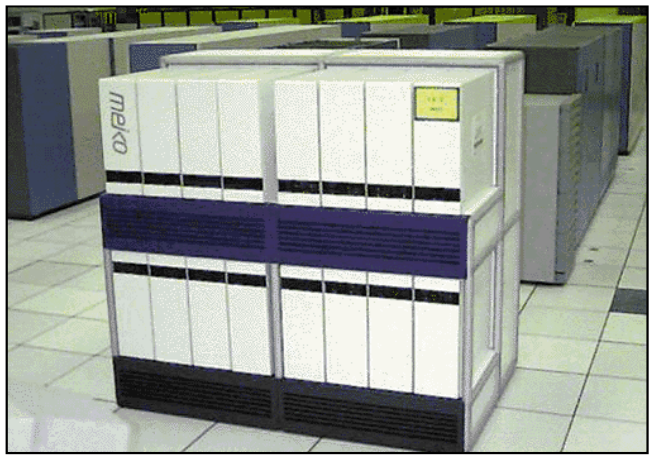
DM-MIMD - Wielokomputery

Cray T3E
(1995)

Thinking Machines CM-5
(1991)

nCube 2s
(1993)

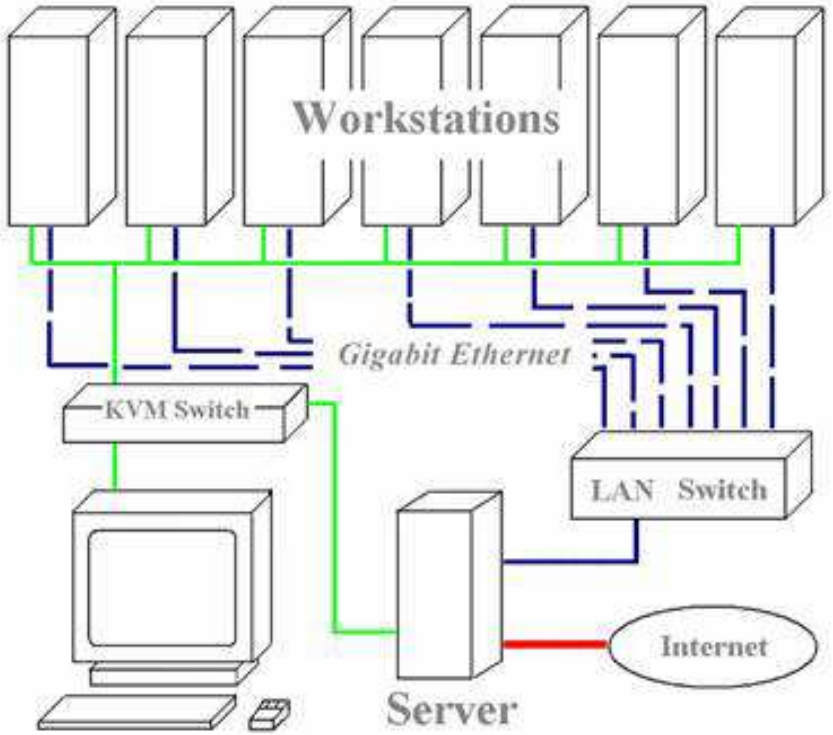
Meiko CS-2
(1993)



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

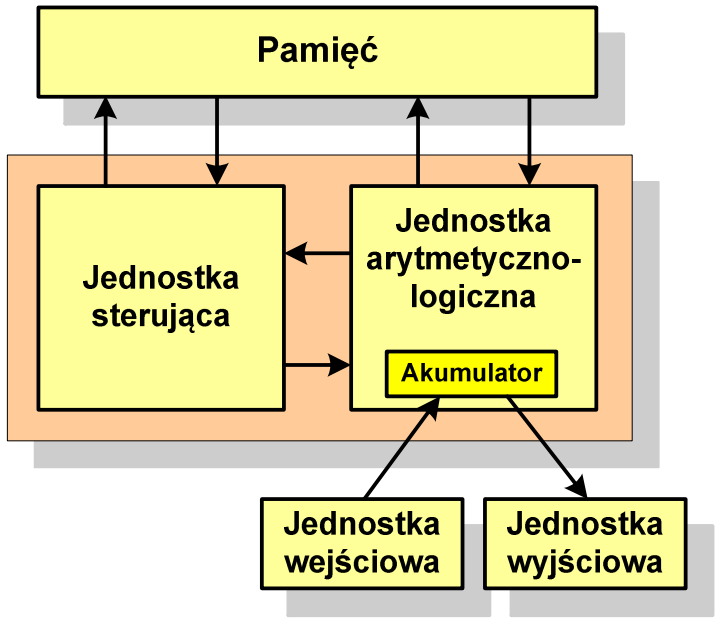
DM-MIMD - Klastry

- 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)

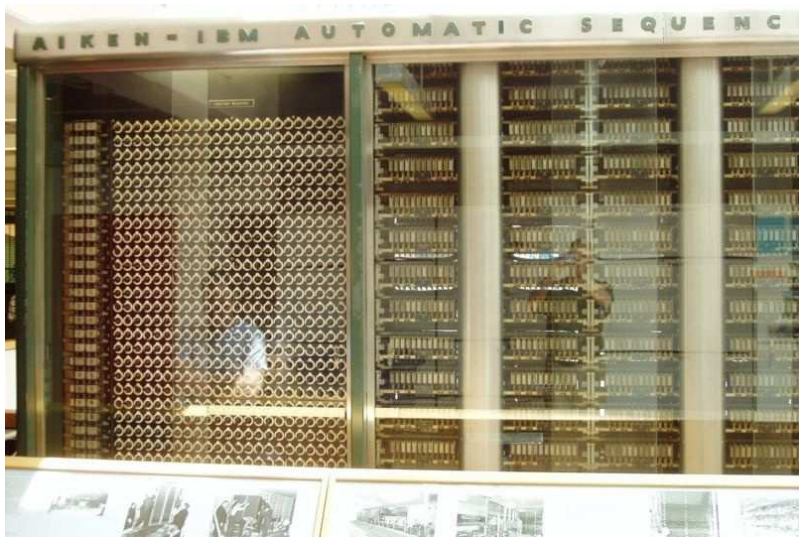


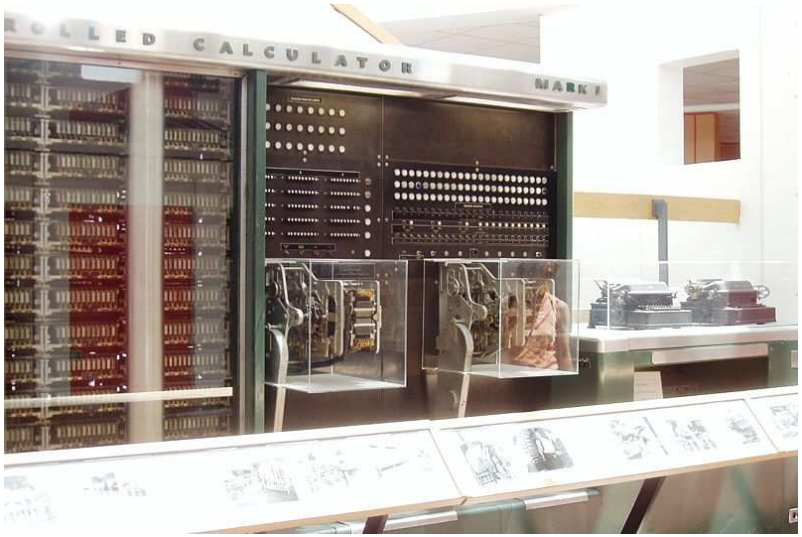
Architektura von Neumanna - podstawowe cechy

- 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



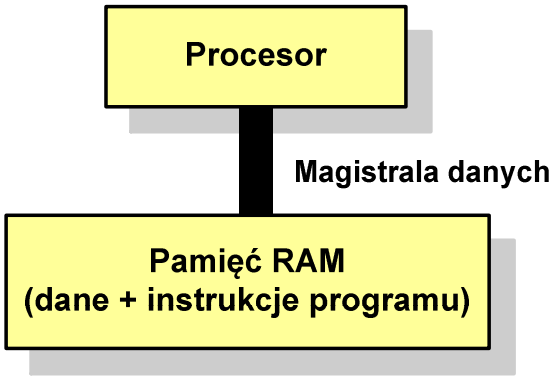


Architektura harwardzka

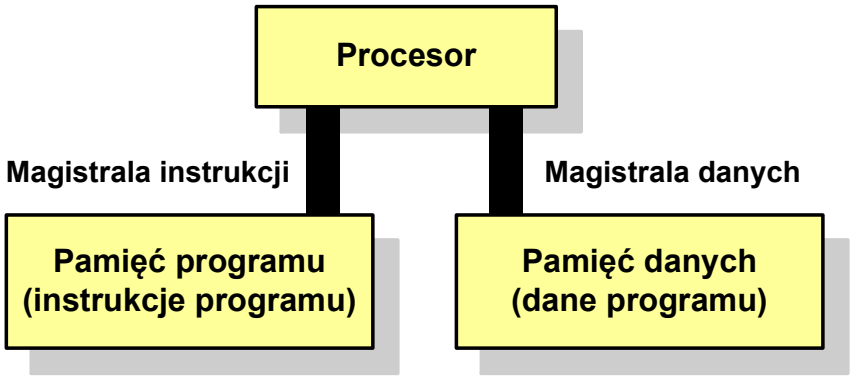
- Pamięci danych i instrukcji mogą różnić się:
  - technologią wykonania
  - strukturą adresowania
  - długością słowa
- Przykład:
  - ATmega16 - 16 kB Flash, 1 kB SRAM, 512 B EEPROM
- Procesor może w tym samym czasie czytać instrukcje oraz uzyskiwać dostęp do danych

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

Koniec wykładu nr 10

Dziękuję za uwagę!