Informatyka 1 (EZ1E2008)

Politechnika Białostocka - Wydział Elektryczny

Elektrotechnika, semestr II, studia niestacjonarne I stopnia

Rok akademicki 2020/2021

Wykład nr 6 (23.04.2021)

dr inż. Jarosław Forenc

Plan wykładu nr 6

- Język C - pętle
  - pętle while i do...while
- Struktura i funkcjonowanie komputera
  - procesor, rozkazy, przerwania, magistrala
  - pamięć komputerowa, hierarchia pamięci
  - pamięć podręczna

Przykład: pierwiastek kwadratowy

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

int main(void)
{
    float x, y;

    printf("Podaj liczbe: ");
    scanf("%f",&x);

    if (x>=0)
    {
        y = sqrt(x);
        printf("Pierwiastek liczby: %f\n",y);
    }
    else
        printf("Blad! Liczba ujemna\n");

    return 0;
}
```

```
Podaj liczbe: -3
Blad! Liczba ujemna
```

```
Podaj liczbe: 3
Pierwiastek liczby: 1.732051
```


Przykład: pierwiastek kwadratowy (pętla while)

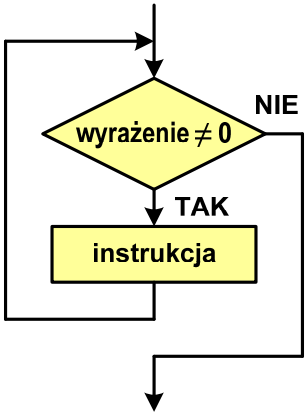
```
#include <stdio.h>
#include <math.h>

int main(void)
{
    float x, y;

    printf("Podaj liczbe: ");
    scanf("%f",&x);
    while (x<0)
    {
        printf("Blad! Liczba ujemna\n\n");
        printf("Podaj liczbe: ");
        scanf("%f",&x);
    }
    y = sqrt(x);
    printf("Pierwiastek liczby: %f\n",y);

    return 0;
}
```

```
Podaj liczbe: -3
Blad! Liczba ujemna

Podaj liczbe: -5
Blad! Liczba ujemna

Podaj liczbe: 3
Pierwiastek liczby: 1.732051
```

Język C - pętla while

```
while (wyrażenie)
    instrukcja
```

- „dopóki wyrażenie w nawiasach jest prawdziwe wykonuj instrukcję”



- Wyrażenie w nawiasach:
  - prawdziwe - gdy jego wartość jest różna od zera
  - fałszywe - gdy jego wartość jest równa zero
- Jako wyrażenie najczęściej stosowane jest wyrażenie logiczne

Język C - pętla while

```
while (wyrażenie)
    instrukcja
```

- Instrukcja:
  - prosta - jedna instrukcja zakończona średnikiem
  - złożona - jedna lub kilka instrukcji objętych nawiasami klamrowymi

```
int x = 10;
while (x>0)
    x = x - 1;
```

```
int x = 10;
while (x>0)
{
    printf("%d\n",x);
    x = x - 1;
}
```


Przykład: suma liczb dodatnich

```
#include <stdio.h>

int main(void)
{
    int x, suma = 0;

    printf("Podaj liczbe: ");
    scanf("%d",&x);

    while(x>0)
    {
        suma = suma + x;
        printf("Podaj liczbe: ");
        scanf("%d",&x);
    }
    printf("Suma liczb: %d\n",suma);

    return 0;
}
```

```
Podaj liczbe: 4
Podaj liczbe: 8
Podaj liczbe: 2
Podaj liczbe: 3
Podaj liczbe: 5
Podaj liczbe: -2
Suma liczb: 22
```


Język C - pętla while

- Program pokazany na poprzednim slajdzie zawiera typowy schemat przetwarzania danych z wykorzystaniem pętli while

```
printf("Podaj liczbe: ");
scanf("%d",&x);

while(x>0)
{
    suma = suma + x;

    printf("Podaj liczbe: ");
    scanf("%d",&x);
}
```

wczytanie danych

operacje na danych

wczytanie danych

- Dane mogą być wczytywane z klawiatury, pliku, itp.

Język C - pętla while (break, continue)

- break i continue są to instrukcje skoku

```
int x=0;
while (x<10)
{
    x++;
    if (x%2==0)
        continue;
    if (x%5==0)
        break;
    printf("%d\n",x);
}
```

- continue przerywa bieżącą iteracje
- break przerywa wykonywanie pętli

Język C - pętla while (najczęstsze błędy)

- Postawienie średnika po wyrażeniu w nawiasach powoduje powstanie pętli nieskończonej - program zatrzymuje się na pętli

```
int x = 10;
while (x>0);
    printf("%d ",x--);
```

- Brak aktualizacji zmiennej powoduje także powstanie pętli nieskończonej - program wyświetla wielokrotnie tę samą wartość

```
int x = 10;
while (x>0)
    printf("%d ",x);
```

```
10 10 10 10 10 ...
```


Język C - pętla while (pętla nieskończona)

- W pewnych sytuacjach celowo stosuje się pętlę nieskończoną (np. w mikrokontrolerach)

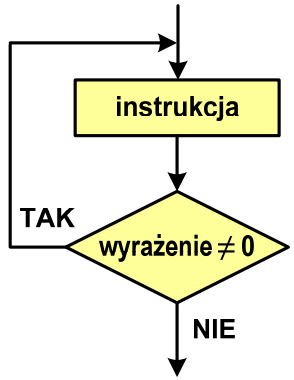
```
while (1)
{
    instrukcja
    instrukcja
    ...
}
```

- W układach mikroprocesorowych program działa aż do wyłączenia zasilania

Język C - pętla do … while

```
do
    instrukcja
while (wyrażenie);
```

- „wykonuj instrukcję dopóki wyrażenie w nawiasach jest prawdziwe”



- Wyrażenie w nawiasach:
  - prawdziwe - gdy jego wartość jest różna od zera
  - fałszywe - gdy jego wartość jest równa zero

Język C - pętla do … while

```
do
    instrukcja
while (wyrażenie);
```

- Instrukcja:
  - prosta - jedna instrukcja zakończona średnikiem
  - złożona - jedna lub kilka instrukcji objętych nawiasami klamrowymi

```
int x = 10;
do
    x = x - 1;
while (x>0);
```

```
int x = 10;
do
{
    printf("%d\n",x);
    x = x - 1;
}
while (x>0);
```


Język C - pętla do … while (break, continue)

- break i continue są to instrukcje skoku

```
int x=0;

do
{
    x++;
    if (x%5==0)
        break;
    if (x%2==0)
        continue;
    printf("%d\n",x);
}
while (i<10);
```

- break przerywa wykonywanie pętli
- continue przerywa bieżącą iterację

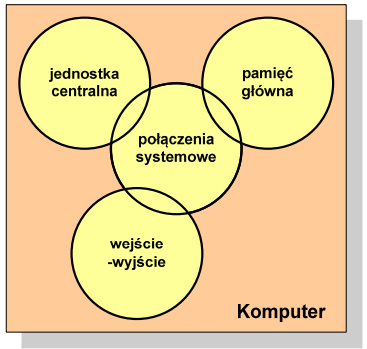
Przykład: suma liczb < 100

```
#include <stdio.h>

int main(void)
{
    int x, suma = 0;

    do
    {
        printf("Podaj liczbe: ");
        scanf("%d",&x);
        suma = suma + x;
    }
    while(suma<100);

    printf("Suma liczb: %d\n",suma);

    return 0;
}
```

```
Podaj liczbe: 34
Podaj liczbe: 9
Podaj liczbe: 26
Podaj liczbe: -8
Podaj liczbe: 67
Suma liczb: 128
```

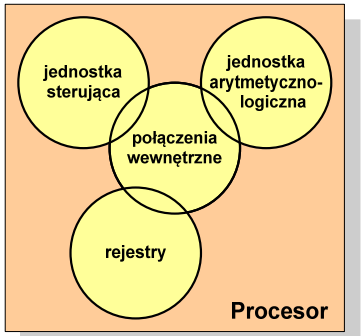

Ogólna struktura systemu komputerowego

- Komputer tworzą cztery główne składniki:
  - procesor (jednostka centralna, CPU) - steruje działaniem komputera i realizuje przetwarzanie danych
  - pamięć główna - przechowuje dane
  - wejście-wyjście - przenosi dane między komputerem a jego otoczeniem zewnętrznym
  - połączenia systemu - mechanizmy zapewniające komunikację między składnikami systemu



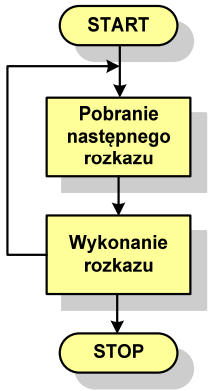
Ogólna struktura procesora

- Główne składniki strukturalne procesora to:
  - jednostka sterująca - steruje działaniem procesora i pośrednio całego komputera
  - jednostka arytmetyczno-logiczna (ALU) - realizuje przetwarzanie danych przez komputer
  - rejestry - realizują wewnętrzne przechowywanie danych w procesorze
  - połączenia procesora - wszystkie mechanizmy zapewniające komunikację między jednostką sterującą, ALU i rejestrami.



Działanie komputera

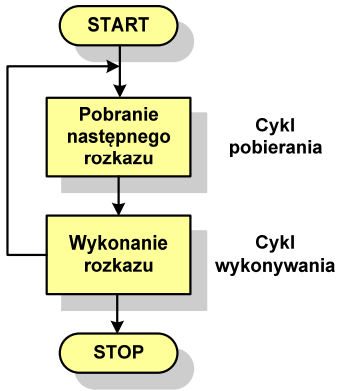
- Podstawowe zadanie komputera to wykonywanie programu
- Program składa się z rozkazów przechowywanych w pamięci
- Rozkazy są przetwarzane w dwu krokach:



- Cykl pobierania (ang. fetch):
  - odczytanie rozkazu z pamięci
  - licznik rozkazów (PC) lub wskaźnik instrukcji (IP) określa, który rozkaz ma być pobrany
  - jeśli procesor nie otrzyma innego polecenia, to inkrementuje licznik PC po każdym pobraniu rozkazu.

Działanie komputera

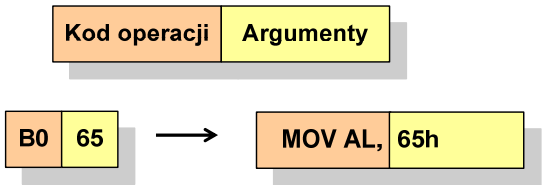
- Podstawowe zadanie komputera to wykonywanie programu
- Program składa się z rozkazów przechowywanych w pamięci
- Rozkazy są przetwarzane w dwu krokach:



- Cykl wykonywania (ang. execution):
  - pobrany rozkaz jest umieszczany w rejestrze rozkazu (IR)
  - rozkaz określa działania, które ma podjąć procesor
  - procesor interpretuje rozkaz i przeprowadza wymagane operacje.

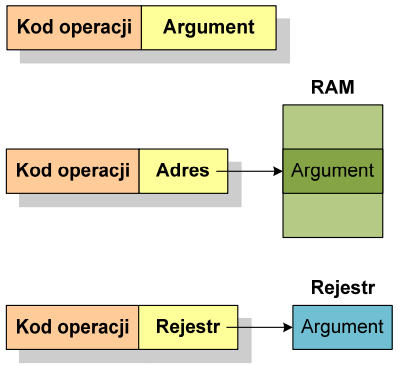
Działanie komputera

- Rozkaz:
  - przechowywany jest w postaci binarnej
  - ma określony format
  - używa określonego trybu adresowania
- Format - sposób rozmieszczenia informacji w kodzie rozkazu
- Rozkaz zawiera:
  - kod operacji (rodzaj wykonywanej operacji)
  - argumenty (lub adresy argumentów) wykonywanych operacji



Działanie komputera

- Tryb adresowania - sposób określania miejsca przechowywania argumentów rozkazu (operandów)
- Przykładowe rodzaje adresowania:
  - natychmiastowe - argument znajduje się w kodzie rozkazu
  - bezpośrednie - kod rozkazu zawiera adres komórki pamięci, w której znajduje się argument
  - rejestrowe - kod rozkazu zawiera oznaczenie rejestru, w którym znajduje się argument



Program w asemblerze

```
.model SMALL
.286
.stack 100h
.code
   start:
      jmp begin

   handler:
      pusha
      push ds
      pop ds
      popa
      iret

   begin:
      mov ax,0000h
      mov ds,ax
      mov di,0070h
      lea ax,handler
```

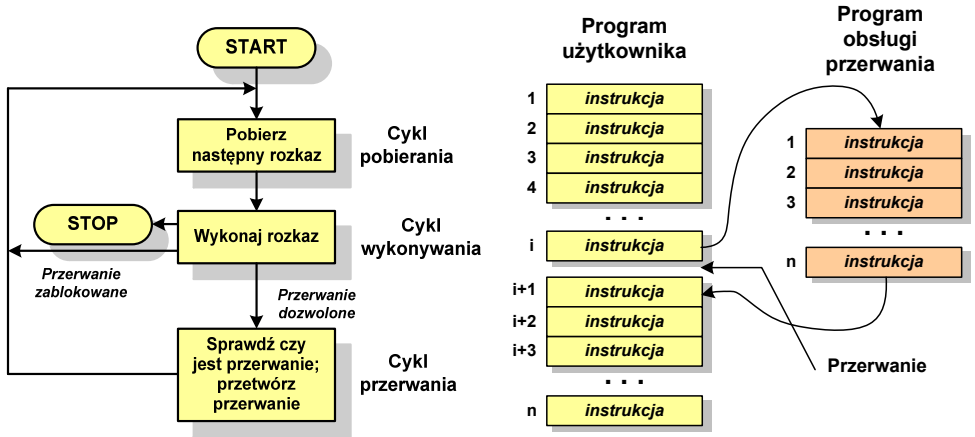
```
cli
mov [di],ax
mov [di+2],cs
sti
mov ax,3100h
mov dx,(offset begin - offset handler)
inc dx
int 21h
end
   start
```


Działanie komputera - przerwania

- Wykonywanie kolejnych rozkazów przez procesor może zostać przerwane poprzez wystąpienie tzw. przerwania (interrupt)
- Przerwanie jest to sygnał pochodzący od sprzętu lub oprogramowania informujący procesor o wystąpieniu jakiegoś zdarzenia (np. wciśnięcie klawisza na klawiaturze)
- Bez przerwań procesor musiałby ciągle kontrolować wszystkie urządzenia zewnętrzne, np. klawiatura, port szeregowy
- Każde przerwanie posiada procedurę obsługi przerwania, która jest wykonywana w momencie jego wystąpienia
- Adresy procedur obsługi przerwań zapisane są w tablicy wektorów przerwań

Działanie komputera - przerwania

- Implementacja przerwań wymaga dodania cyklu przerwania do cyklu rozkazu

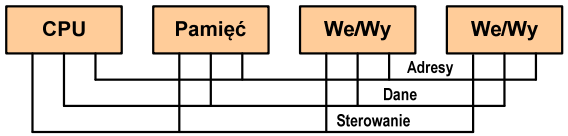


Rodzaje przerwań

- Sprzętowe
  - zewnętrzne - sygnały pochodzące z urządzeń zewnętrznych i służące do komunikacji z nimi, np. 08H - zegar, 09h - klawiatura
  - wewnętrzne - wywoływane przez procesor w celu zasygnalizowania sytuacji wyjątkowych (faults, traps, aborts)
- Programowe
  - instrukcje programu wywołują przerwanie - tym samym wykonywana jest procedura obsługi przerwania
  - służą głownie do komunikacji z systemem operacyjnym (DOS - 21h, Windows - 2h, Linux - 80h)

Magistrala

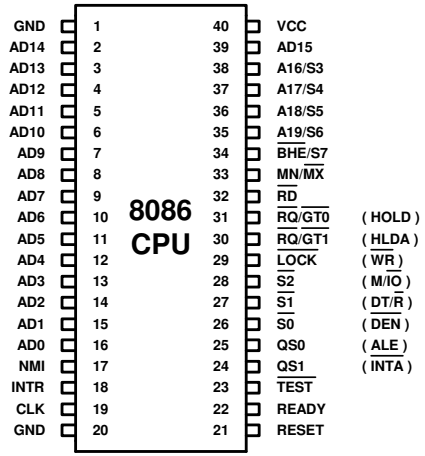
- Najczęściej stosowana struktura połączeń to magistrala, składająca się z wielu linii komunikacyjnych, którym przypisane jest określone znaczenie i określona funkcja

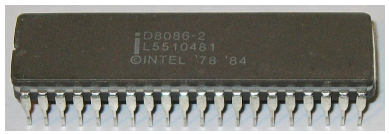


  - linie danych (szyna danych) - przenoszą dane między modułami systemu, liczba linii określa szerokość szyny danych (8, 16, 32, 64 bity)
  - linie adresowe - służą do określania źródła i miejsca przeznaczenia danych przesyłanych magistralą; liczba linii adresowych określa maksymalną możliwą pojemność pamięci systemu
  - linie sterowania - służą do sterowania dostępem do linii danych i linii adresowych

Intel 8086

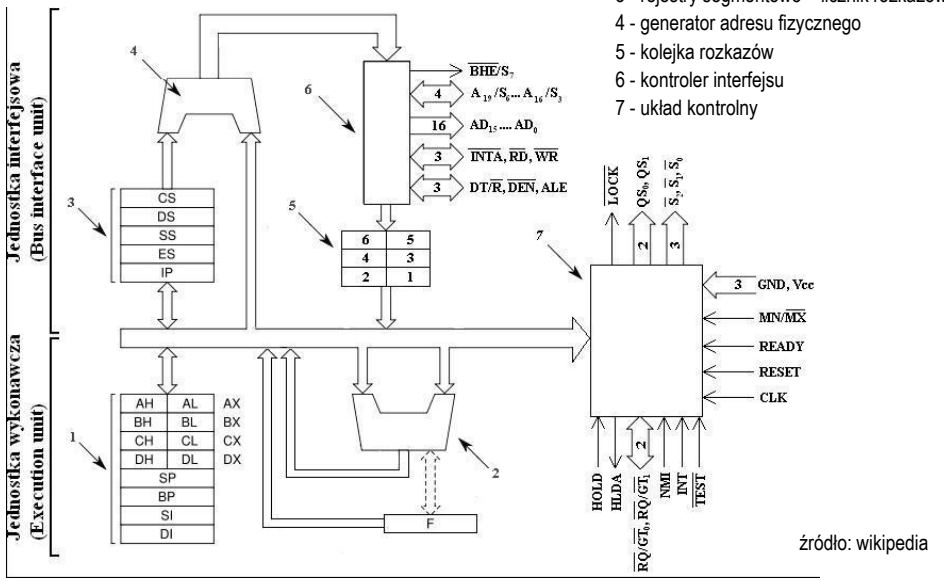
- 1978 rok
- Procesor 16-bitowy
- 16-bitowa magistrala danych
- 20-bitowa magistrala adresowa
- Adresowanie do 1 MB pamięci
- Częstotliwość: 10 MHz
- Multipleksowane magistrale: danych i adresowa
- Litografia: 3 μm





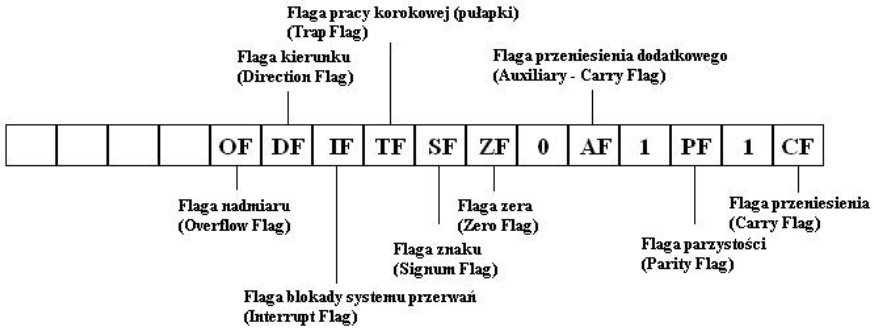
Intel 8086

1 - rejestry ogólnego przeznaczenia
2 - ALU + rejestr znaczników (flag)
3 - rejestry segmentowe + licznik rozkazów
4 - generator adresu fizycznego
5 - kolejka rozkazów
6 - kontroler interfejsu
7 - układ kontrolny



źródło: wikipedia

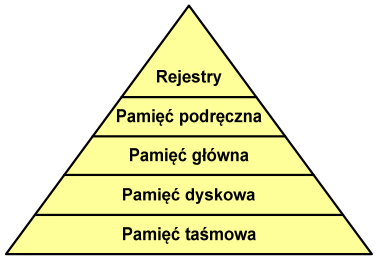
Intel 8086 - Rejestr flag



źródło: wikipedia

Systemy pamięci komputerowych

- W systemach komputerowych nie stosuje się jednego typu pamięci, ale hierarchię pamięci



- Rozpatrując hierarchię od góry do dołu obserwujemy zjawiska:
  - malejący koszt na bit
  - rosnącą pojemność
  - rosnący czas dostępu
  - malejącą częstotliwość dostępu do pamięci przez procesor

Półprzewodnikowa pamięć główna

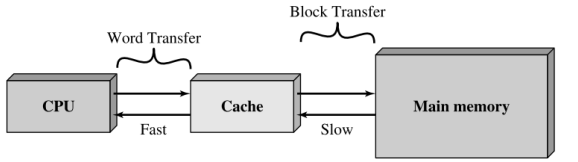
- RAM (Random Access Memory) - pamięć o dostępie swobodnym
  - odczyt i zapis następuje za pomocą sygnałów elektrycznych
  - pamięć ulotna - po odłączeniu zasilania dane są tracone
  - DRAM - pamięć dynamiczna:
    - przechowuje dane podobnie jak kondensator ładunek elektryczny
    - wymaga operacji odświeżania
    - jest mniejsza, gęściej upakowana i tańsza niż pamięć statyczna
    - stosowana jest do budowy głównej pamięci operacyjnej komputera
  - SRAM - pamięć statyczna:
    - przechowuje dane za pomocą przerzutnikowych konfiguracji bramek logicznych
    - nie wymaga operacji odświeżania
    - jest szybsza i droższa od pamięci dynamicznej
    - stosowana jest do budowy pamięci podręcznej

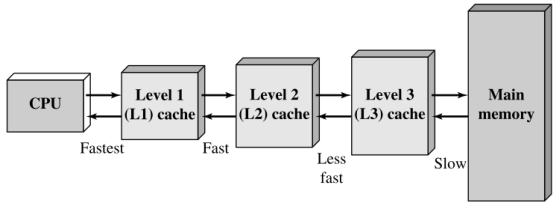
Półprzewodnikowa pamięć główna

- ROM (ang. Read-Only Memory) - pamięć stała
  - pamięć o dostępie swobodnym przeznaczona tylko do odczytu
  - dane są zapisywane podczas procesu wytwarzania, pamięć nieulotna
- PROM (ang. Programmable ROM) - programowalna pamięć ROM
  - pamięć nieulotna, może być zapisywana tylko jeden raz
  - zapis jest realizowany elektrycznie po wyprodukowaniu
- EPROM - pamięć wielokrotnie programowalna, kasowanie następuje przez naświetlanie promieniami UV
- EEPROM - pamięć kasowana i programowana na drodze elektrycznej
- Flash - rozwinięcie koncepcji pamięci EEPROM, możliwe kasowanie i programowanie bez wymontowywania pamięci z urządzenia

Pamięć podręczna (cache)

- Dodatkowa, szybka pamięć (SRAM) umieszczana pomiędzy procesorem a pamięcią główną
- Zastosowanie pamięci podręcznej ma na celu przyspieszenie dostępu procesora do pamięci głównej





źródło: W. Stallings, Computer Organization and Architecture

Koniec wykładu nr 6

Dziękuję za uwagę!