Architektura.

Tadeusz Puźniakowski

PJATK

2015

Co na dzisiaj

Plan dzisiejszego wykładu

- Dwa podejścia do architektury systemów mikroprocesorowych
- Little oraz Big Endian
- Cykl wykonania
- Adresowanie
- "Witaj w Świecie"w Asemblerze

System mikroprocesorowy

System mikroprocesorowy

System do realizacji dowolnego zadania dającego się sprowadzić do przetwarzania wektorów informacji cyfrowej. (wyjaśnię).

Składa się z na to sprzęt (hardware) i oprogramowanie (software).

System mikroprocesorowy

System mikroprocesorowy

- Procesor - CPU
- Pamięć - Memory
- Układy lub porty wejścia i wyjścia - peryferia - I/O
- Magistrale - szyny systemowe - Bus

Procesor

CPU - Central Processing Unit

Bardzo skrótowo - CPU zawiera:

- ALU
- układ sterujący z dekoderem rozkazów
- rejestry

Dwa podejścia do systemów mikroprocesorowych

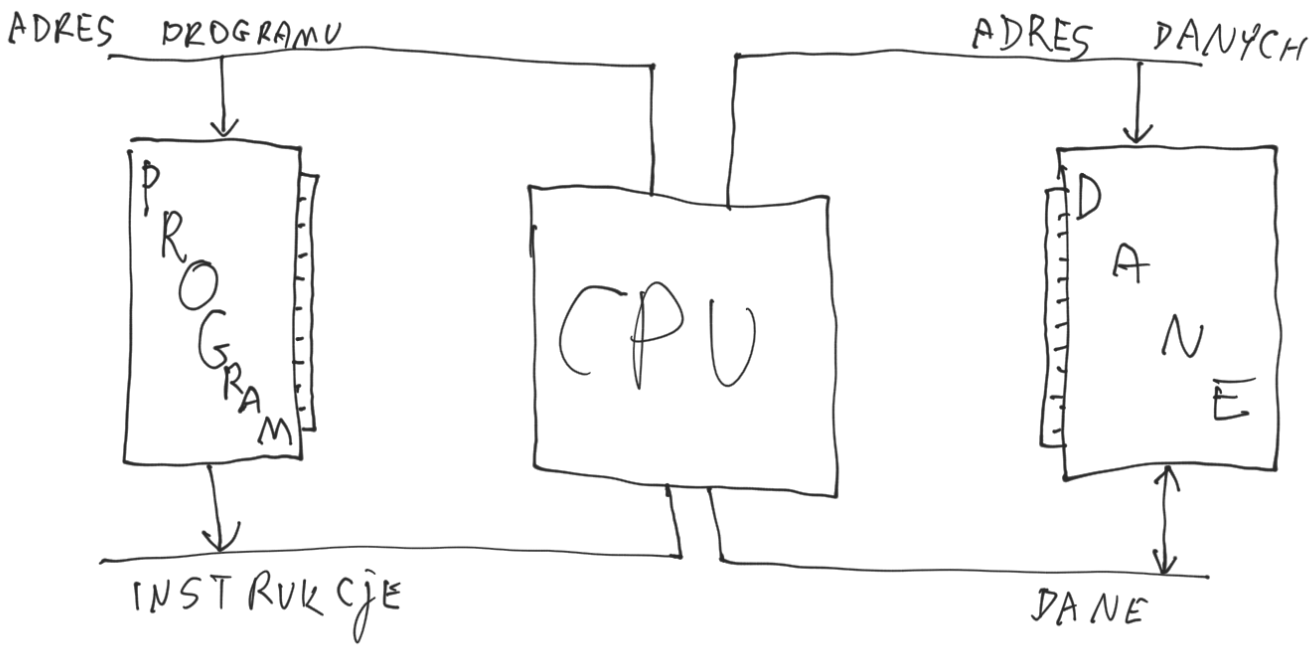
Podstawowe podejścia do architektury komputerów

- Architektura Harwardzka (oddzielnie dane/program)
- Architektura Von Neumanna (magistrala)

Architektura typu Harvardzkiego

Harvard

Nazwa wzięła się od komputera Harvard Mark I działającego na przekaźnikach (1944r).



Architektura typu Harvardzkiego

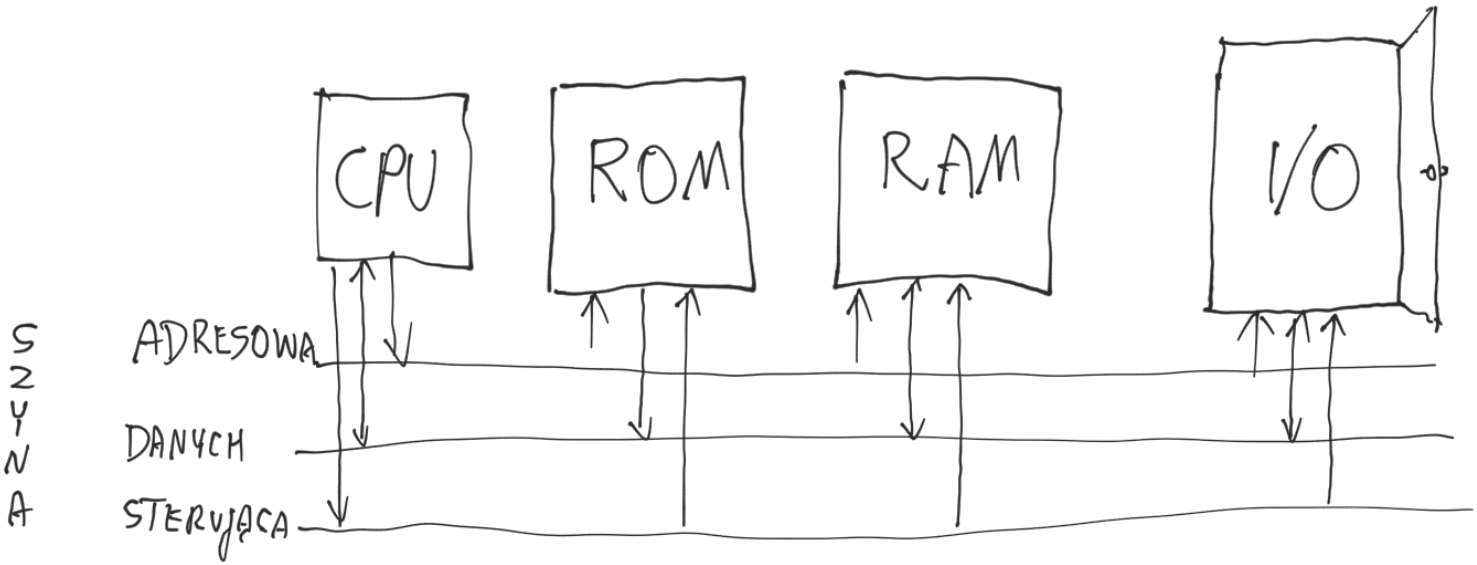
Harvard

- Niezależna pamięć dla danych i programu
- Dwie szyny adresowe
- Równoległa transmisja danych i instrukcji

Architektura typu Von Neumann

Architektura Von Neumann / Princeton

Opracowana w 1945r przez Johna von Neumanna, Johna W. Mauchly'ego oraz Johna Presper Eckerta.



Architektura typu Von Neumann

Von Neumann

- Program oraz dane znajdują się w jednej przestrzeni adresowej w pamięci RAM
- Wspólna szyna danych i programu
- Wspólna szyna adresowa
- Sekwencyjna transmisja danych i instrukcji

Zadanko

Pytanie

Czy ktoś zna procesory działające na architekturze typu Harwardzkiego?
Czy ktoś zna procesory działające na architekturze typu von Neumann?

Architektura typu Harvardzkiego

Harvard

Współcześnie używana w:

- Niektóre mikro-kontrolery
- Procesory DSP (Digital Signall Processing)
- Niektóre (ARM9) procesory ARM
- W pewnym sensie niektóre CPU z rodziny x86 (ze względu na cache)

Architektura typu Von Neumann

Harvard

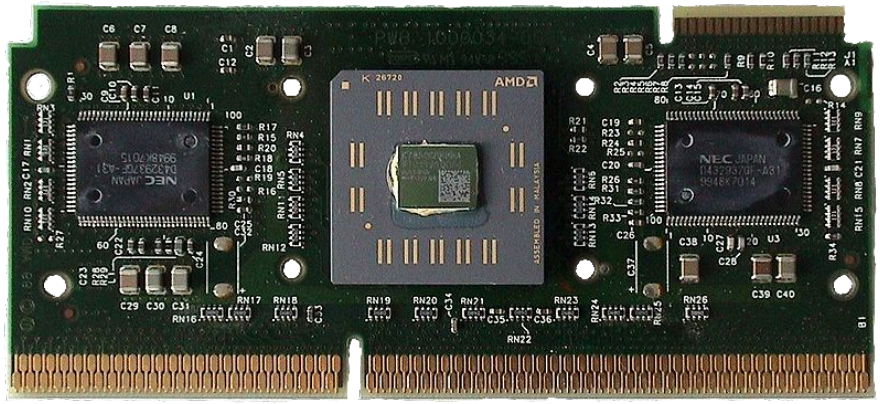
Współcześnie używana w:

- Procesory x86
- Niektóre (ARM7) procesory ARM

Porównanie

Uwaga

Aktualnie w wielu przypadkach rozróżnienie tych dwóch podejść jest płynne, na przykład procesor Athlon na slocie A. (obr. z Wikipedii)



Wykonanie instrukcji

Cykl wykonania instrukcji

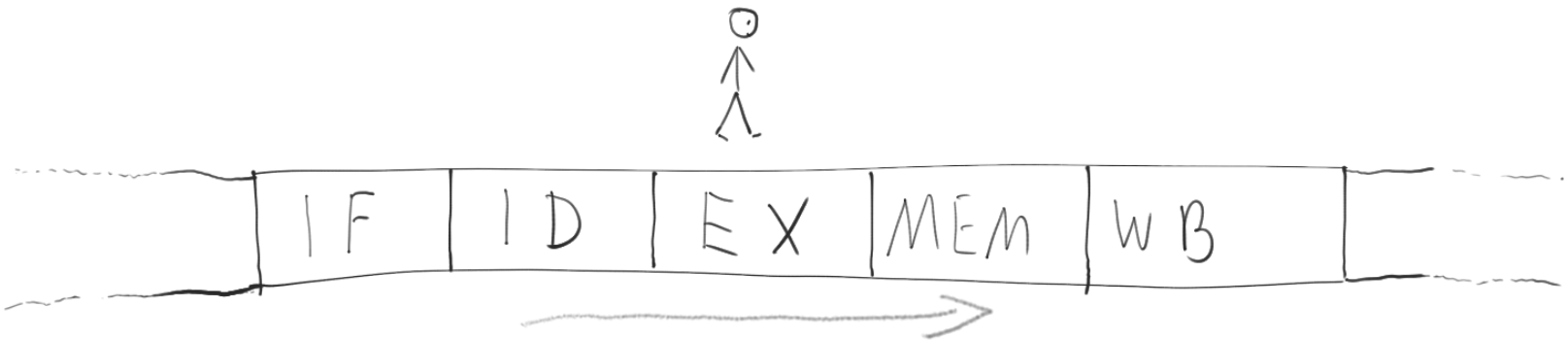
- IF — Instruction Fetch
- ID — Instruction Decode
- EX — Execute
- MEM — Memory Access
- WB — Write Back

Zobacz także
http://www.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-optimization-manual.pdf str 57

Cykl wykonania instrukcji

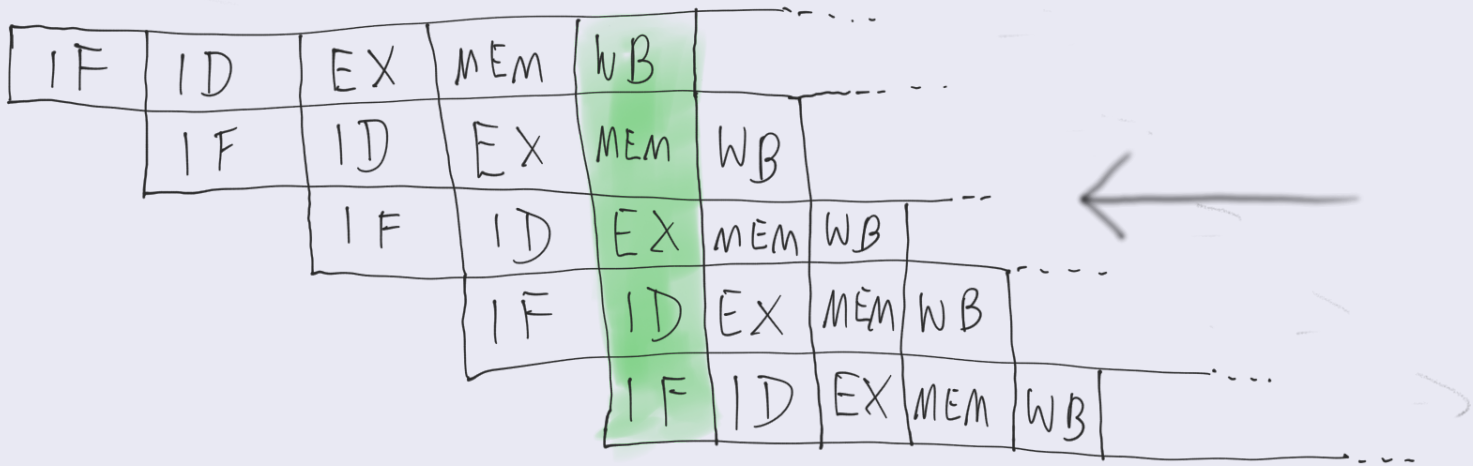
Optymalizacja

Cykl wykonawczy składa się z kilku kolejnych elementów. Może da się sprawić, aby to działało szybciej?



Cykl wykonania instrukcji

Optymalizacja - Potokowość



Kolejność bajtów w pamięci

Big Endian

Liczby nie mieszczące się w jednym bajcie zapisywane są w kolejności od bardziej znaczącego, do mniej znaczącego.

Small/Little Endian

Liczby nie mieszczące się w jednym bajcie zapisywane są w kolejności od najmniej znaczącego, do najbardziej znaczącego.

x86

Small Endian

Rejestry

Rejestr

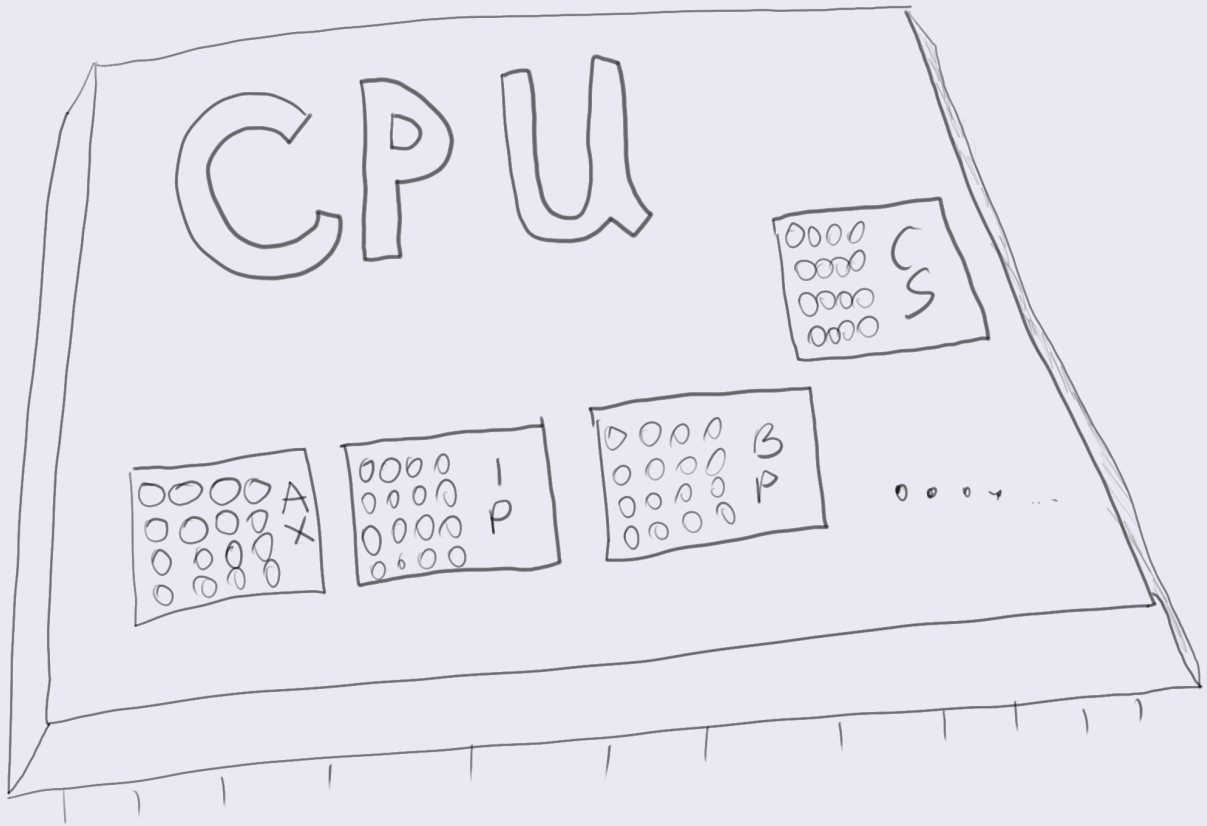
Rejestr jest to komórka pamięci wewnątrz procesora. Bardzo mała, ale działająca z taką częstotliwością jak procesor.

UWAGA

Różne rejestry mogą mieć różne przeznaczenie. Nie wszystkie rejestry można używać bezpośrednio.

Obrazek Poglądowy

Gdzie są te rejestry?



Rejestry w architekturze 8086

- Rejestry ogólnego przeznaczenia
- Rejestr stanu, albo rejestr znaczników
- Rejestry indeksowe
- Rejestry segmentowe
- Rejestr instrukcji

Rejestr instrukcji

IP - Instruction Pointer

Ten rejestr służy do przechowywania adresu aktualnie wykonywanego rozkazu. Nie można go modyfikować bezpośrednio.

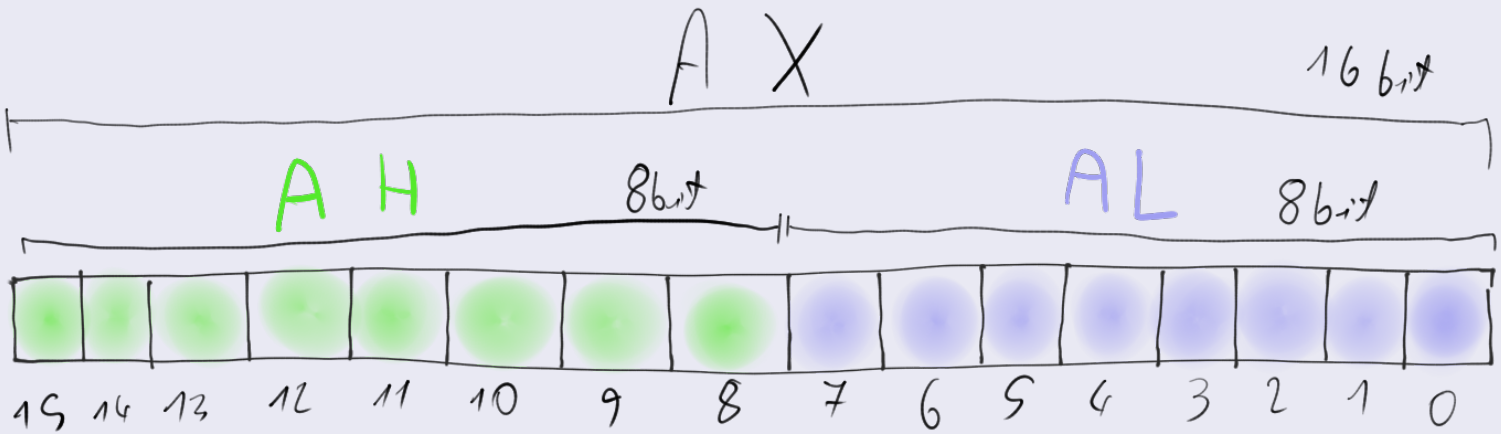
Rejestry ogólnego przeznaczenia

Służą do obliczeń oraz do przechowywania tymczasowych wyników.

- AX - składa się z AH i AL
- BX - składa się z BH i BL
- CX - składa się z CH i CL
- DX - składa się z DH i DL

Rejestry ogólnego przeznaczenia

Przykład - rejestr AX



Rejestr znaczników

Jest to 16 bitowy rejestr przechowujący stan. Nie można się do niego odnosić bezpośrednio.

Rejestry indeksowe

Wykorzystywane są przede wszystkim do określania offsetu.

- SI - Source Index
- DI - Destination Index
- BP - Base Pointer - do adresowania pamięci
- SP - Stack Pointer - wskaźnik stosu

Rejestry segmentowe

Wykorzystywane są przede do określania segmentu.

- CS - Code Segment - segment aktualnego rozkazu
- DS - Data Segment - segment danych
- ES - Extra Segment
- SS - Stack Segment - segment stosu

Procesor

UWAGA

Na większości zajęć będziemy operowali na 16 bitowym asemblerze dla systemu DOS.

Kiedy to opanujemy, to bez problemów przejdziemy na jakikolwiek inny assembler. Będziemy także w stanie spojrzeć na program z perspektywy sprzętu.

Adresy

Adres fizyczny

Jest to adres faktycznie wykorzystywany przy komunikowaniu się z pamięcią operacyjną i urządzeniami.

Adres logiczny

Adres jakim posługuje się program.

Zagadka

Pytanie

Jak zaadresować 1MB pamięci na 16bitowym procesorze?

Adresy

Adres fizyczny

Jest to adres faktycznie wykorzystywany przy komunikowaniu się z pamięcią operacyjną i urządzeniami. W trybie rzeczywistym jest szerokości 20bit.

Adres logiczny

Adres którym posługuje się program. W trybie rzeczywistym składa się z szesnastobitowych wartości - segmentu i offsetu.

Segmentacja pamięci

Segment

Pamięć jest podzielona na segmenty, w 16bitowym systemie segmenty rozpoczynają się co 16 bajtów.

Offset

Przesunięcie w bajtach względem segmentu.

Przykład adresu typu segment:offset

0A000:1234

Segmentacja pamięci

Obliczanie adresu fizycznego

Adres fizyczny = Segment $\times$ 16 + Offset

Wygodniej

Adres fizyczny = Segment $<<$ 4 + Offset (przykład na tablicy)

Segmentacja pamięci

UWAGA

Segmenty nie są rozłączne! (przykład na tablicy)

Przerwania

Przerwanie

Sygnał powodujący przerwanie aktualnie wykonywanego programu i wykonanie kodu obsługi przerwania.

Przerwanie sprzętowe

Sygnał pochodzi od jakiegoś urządzenia. Mogą być zewnętrzne i wewnętrzne (wyjątki).

Przerwanie programowe

Z programu wywoływana jest procedura obsługi przerwania.

Kod maszynowy

Zdefiniowany przez producenta układu zestaw instrukcji.

UWAGA

Dla procesora x86 nie ma różnicy między danymi a instrukcjami przechowywanymi w pamięci!

Assembler

Rozmiary danych w x86

- bit – najmniejsza porcja informacji, przyjmuje 0 (fałsz) albo 1 (prawda).
- bajt – 8 bitów. Najmniejsza porcja informacji do jakiej możemy się odnieść bezpośrednio.
- słowo – 2 bajty, 16 bitów.
- podwójne słowo – 32 bity

Assembler

Język assemblera

Niskopoziomowy język programowania, w którym zasadniczo jedna instrukcja odpowiada jednemu rozkazowi procesora. Komendy języka assemblera są zastąpione mnemonikami.

Mnemonik w języku assemblera

Pojedyncza komenda procesora zapisana w sposób czytelny dla człowieka, na przykład mnemonik "mov" z parametrami "ah, 09h" tłumaczy się na bajty 0b4h, 09h.

Assembler

Program tłumaczący komendy w języku assemblera na kod maszynowy.

Assembler

UWAGA

Nie istnieje standard języka assemblera. Na zajęciach korzystamy z NASM.

Assembler

Typy danych

Większość języków assemblera nie przechowuje typów danych i nie posiada czegoś takiego jak zmienna w klasycznym rozumieniu tego słowa.

Rozmiary danych

db - byte, dw - word, dd - double word, dq - quad word

Etykiety

W programie napisanym w języku assemblera etykieta oznacza początek czegoś. Służą do oznaczania początków danych i początków bloków programu (np. procedur).

Rozmiary danych - UWAGA

```
txt: db 'He'
```

```
txt: db 'H','e'
```

```
txt: db 72,101
```

```
txt: db 48h,65h
```

```
txt: dw 6548h
```

Przykładowy program w języku Asemblera

```
org 100h
mov dx, txt
mov ah, 9h
int 21h
mov ax, 4c00h
int 21h
txt: db "Hello", 0ah, 0dh,"$"
```

Przykładowy program w języku Asemblera

```
00000000 ba0c01      mov dx, 010ch
00000003 b409        mov ah, 9h
00000005 cd21        int 21h
00000007 b8004c      mov ax, 4c00h
0000000a cd21        int 21h
0000000c 48          db  'H'
0000000d 65          db  'e'
0000000e 6c          db  'l'
0000000f 6c          db  'l'
00000010 6f          db  'o'
00000011 0a          db  0ah
00000012 0d          db  0dh
00000013 24          db  '$'
```

Demo na żywo

Uruchomienie DosBox

Jak go ustawić, jak uruchomić

Uruchomienie przykładowego programu

Wyjaśnienie, kompilacja i uruchomienie

Demo na żywo - debug

Program debug pozwala na podstawowe debugowanie i wykonanie krok po kroku naszych programów.
O debuggerach będzie jeszcze na późniejszych zajęciach.

Uruchomienie programu w debuggerze

```
debug program.com
```

Demo na żywo - debug

Komendy programu debug

- L – ładuje program do pamięci/restartuje go
- T – wykonuje jedną instrukcję i idzie dalej
- P – wykonuje krokowo instrukcje, tak jak T, ale jak trafi na LOOP, CALL, INT, REP, wtedy wykonuje cały blok aż do końca.
- r – wyświetlenie/ustawienie wartości rejestru. np:
  - rax pokaże rejsetr ax, jeśli jeszcze poda się jakąś wartość, to zostanie ona zachowana
- D – zrzut pamięci
- Q – zamknięcie programu Debug