Symulator poleceń Intel 8086

Autor: Bruno Talaga

Spis treści

[I. Wstęp 3](#_Toc186924546)

[II. Intel 8086 4](#_Toc186924547)

[III. Funkcjonalność symulatora 5](#_Toc186924548)

[III.I. Instrukcja MOV 5](#_Toc186924549)

[III.II. Instrukcja XCHG 9](#_Toc186924550)

[III.III. Instrukcja PUSH 11](#_Toc186924551)

[III.IV. Instrukcja POP 12](#_Toc186924552)

[III.V. Adresowanie pamięci 13](#_Toc186924553)

[III.VI. Dodatkowe funkcje 16](#_Toc186924554)

[IV. Informacje dodatkowe na temat symulatora 20](#_Toc186924555)

# I. Wstęp

Program został opracowany jako projekt zaliczeniowy z przedmiotu Architektura Systemów Komputerowych. Ma on na celu zasymulowanie niektórych poleceń dostępnych dla procesora Intel 8086 (polecenia MOV, XCHG, PUSH, POP), interakcji pomiędzy rejestrami, pamięcią oraz stosem. Symulator został napisany jako aplikacja sieciowa w języku JavaScript, jest hostowany pod adresem [brunotalaga.com/CPUsimProject/](https://www.brunotalaga.com/CPUsimProject/) a repozytorium jest dostępne pod adresem: [github.com/w4ty/CPUsimProject/](https://github.com/w4ty/CPUsimProject) gdzie zawarty jest cały kod projektu.

Autorem projektu jest Bruno Talaga.

# II. Intel 8086

Intel 8086 jest 16-bitowym mikroprocesorem wyprodukowanym przez firmę Intel w roku 1978, został zaprojektowany jako rozszerzenie 8-bitowego Intel 8080/8085.

Został on wykorzystany przez firmę IBM w komputerze IBM PC w 1981 roku, co silnie wpłynęło na popularność mikroprocesora, czego konsekwencją było rozwinięcie architektury x86, najpopularniejszej architektury procesorów w komputerach personalnych.

Prędkość zegara pierwszego modelu 8086 wynosiła 5MHz, późniejsze modele jednak miały taktowania sięgające 10MHz. Modeli 8086 było siedem, 8086, 8086-1, 8086-2, 8086-4, I8086, M8086, 80C86.

# III. Funkcjonalność symulatora

## III.I. Instrukcja MOV

Program pozwala na wpisanie wartości do rejestru w systemie heksadecymalnym za pomocą instrukcji MOV:

A screenshot of a computer

Description automatically generated

Wartość musi zawsze być liczbą heksadecymalną o długości czterech znaków, w przypadku próby wpisania wartości niepoprawnej, program zignoruje komendę i wypisze błąd wskazujący na nieprawidłowy argument:

A screenshot of a computer

Description automatically generated

Możliwe jest wypełnienie rejestrów losowymi wartościami oraz ich wyzerowanie:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Możliwe jest wpisywanie wartości jednego rejestru do drugiego:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Dozwolone jest także wpisywanie do pamięci wartości heksadecymalnej lub wartości rejestru i wypisywanie z pamięci do rejestru:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## III.II. Instrukcja XCHG

Zaimplementowana jest instrukcja XCHG, pozwalająca na zamianę wartości rejestrów i pamięci:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## III.III. Instrukcja PUSH

Program symuluje działanie stosu, co pozwala użyć instrukcji PUSH do wpisania danych z rejestru do stosu:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## III.IV. Instrukcja POP

Dzięki implementacji stosu możliwe jest także użycie instrukcji POP do wypisania wartości ze stosu (wskazywanej przez wskaźnik stosu) do dowolnego rejestru:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## III.V. Adresowanie pamięci

Program symuluje i oblicza adresy pamięci korzystając z trzech trybów adresowania pamięci: indeksowego, bazowego oraz indeksowo-bazowego:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Przykładowo chcąc wpisać wartość FFCE dla różnych trybów adresowania pamięci:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Do wyliczenia adresu pamięci można także stosować dowolne wartości heksadecymalne o długości czterech znaków dodawane operatorem „+”:

A screenshot of a computer

Description automatically generated

## III.VI. Dodatkowe funkcje

Program pozwala szybko zmienić wartości kilku rejestrów poprzez wpisanie do tabeli oczekiwanych wartości:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Pamięć może zostać wyczyszczona klikając odpowiedni przycisk:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Tak samo stos, resetuje to także wskaźnik stosu:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Log operacji też można wyczyścić:

A screenshot of a computer

Description automatically generated

Program ma wbudowaną instrukcję tłumaczącą działanie instrukcji MOV, XCHG, PUSH i POP, oraz jak można budować argumenty:

A screenshot of a computer

Description automatically generated

# IV. Informacje dodatkowe na temat symulatora

Symulator został napisany w języku JavaScript, repozytorium zawierające cały kod jest dostępne na platformie Github.

Całość programu jest oparta na interpretowaniu argumentów wpisanych w pola tekstowe w zakładce „Pole komend”, pomimo tego, że symulator zawiera przyciski pozwalające na automatyczne wpisanie wybranych rejestrów i adresów, wszystko użytkownik może wpisać samemu w odpowiednie pola tekstowe, co program po uruchomieniu komendy zinterpretuje.

Nazwy rejestrów muszą zawsze zostać wpisane dokładnie tak samo jak nazwane są w rzeczywistości, co oznacza, że przykładowo wpisanie „ax” zamiast „AX” spowoduje błąd niepoprawnego argumentu. Liczby muszą być podawane jako heksadecymalne o długości czterech znaków, w ich przypadku wielkość liter nie ma znaczenia, więc zarówno „ffff” jak i „FFFF” będą poprawne. Liczenie adresu pamięci zawsze odbywa się kwadratowych nawiasach „[ ]”, dozwolone są nazwy rejestrów (aby pobrać z nich wartość do obliczeń), liczby heksadecymalne oraz operator „+”. Wszystkie argumenty obliczeń adresu pamięci trzymają się takich samych zasad jak zwykłe argumenty.

Cały program jest lekki, pobierając trochę ponad 150kB w wersji hostowanej na stronie internetowej, z czego około 130kB to ikony używane w przyciskach (w momencie pisania sprawozdania).

Wersja hostowana na stronie [brunotalaga.com/CPUsimProject/](https://www.brunotalaga.com/CPUsimProject/) korzysta z serwerów Github Pages i jest automatycznie aktualizowana wraz z branch’em master, co oznacza, że rozwijanie projektu jest łatwe i szybkie.