## Architektury systemów koputerowych

## Stanislav Podmazov

## Nr albumu: 14221

## Projekt zaliczeniowy: Symulacja rozkazu procesora Intel8086

## Na ocenę: 3,5

Mikroprocesory 8086 składają się z dwóch części pracujących w dużej mierze niezależnie . - układu wykonawczego EU - układu sterowania magistral BIU. Jego głównym zadaniem jest dekodowanie oraz wykonanie kolejnych rozkazów, wprowadzonych przez układ sterowania magistral do kolejki rozkazów. Wykonanie rozkazów odbywa się za pomocą jednostki arytmetyczno-logicznej ALU związanego z nią rejestru znaczników, rejestrów arytmetycznych ogólnego przeznaczenia orazprogramowoniedostępnychrejestrówroboczych.Układwykonawczykomunikuje się z otoczeniem zewnętrznych wyłącznie za pośrednictwem układu sterowania magistral. Układ ten realizuje wszystkie operacje dotyczące zewnętrznych magistral procesora, takie jak komunikacja z pamięcią operacyjną, bądź wymiana danych z urządzeniami zewnętrznymi. Układ sterowania magistral zawiera sumator oraz rejestry służące do obliczania 20-bitowego adresu fizycznego. Wymiana danych odbywa się magistra]ą 16-bitową w procesorze 8086 i 8-bitową w procesorze 8088. Mówi się czasem, że procesor 8088 jest mikroprocesorem 8-bitowym o rozbudowanej liście rozkazów. Kolejka rozkazów, a ściślej kolejka następujących po sobie bajtów rozkazów, ma długość 6 bajtów w procesorze 8086 i 4 bajtów w 8088. Magistrala adresowa jest 20-bitowa, co pozwala na zaadresowanie pamięci operacyjnej o pojemności do 1 MB. Konstrukcja mikroprocesora 8086 umożliwia jego zastosowanie w systemach mikrokomputerówych bardziej lub mniej rozbudowanych, o różnych mocach obliczeniowych. Mikroprocesor 8086 może pracować w dwóch trybach - minimalnym i maksymalnym. Do wyboru trybu pracy służy specjale wejście.

Adresowanie

Format rozkazu wewnętrznego komputera określa podział słowa rozkazu na pola i sposób dekodowania poszczególnych pól. Najprostszy podział słowa rozkazu wyróżnia pole kodu operacji i pole adresowe. Pole kodu operacji może składać się z wielu pól, z których każde określa jeden element kodu operacji rozkazu: samą operację. Pole adresowe określa pochodzenie argumentów operacji, przeznaczenie wyniku. Pole adresu zawiera albo adres pamięci operacyjnej, albo numer rejestru albo tez dana umieszczona w rozkazie. Bardzo często zawartość pola adresowego rozkazu nie jest bezpośrednio użyta jako adres danej lub rozkazu lecz procesor wykonuje na nim tzw. operacje adresujące (ang. addressing operations), które przekształcają zawartość pola adresowego zanim zostanie wykorzystana jako ostateczny adres argumentu rozkazu. Rodzaj operacji, którą procesor wykonuje na zawartości pola adresowego rozkazu jest nazywany trybem adresowania (ang. addressing mode). Tryb adresowania dla rozkazu jest zwykle określony w jednym lub kilku polach należących do pola kodu operacji rozkazu. W niektórych przypadkach rozkazów, tryb adresowania wynika bezpośrednio z kodu operacji rozkazu.

Typy adresowania:

Adresowanie bezpośriednie jest najbardziej podstawowym trybem adresowania. W tym trybie zawartość pola adresowego stanowi już finalny adres argumentu rozkazu w pamięci operacyjnej i nie podlega przekształceniu. Ten tryb stosujemy, gdy nie zależy nam na tym, aby nasz program był przesuwalny w pamięci operacyjnej, lecz jest przeznaczony do wykonania przy zapisie w ściśle określone miejsce w pamięci.

Adresowanie indeksowe jest inaczej nazywane modyfikacją adresu przez indeksowanie. W tym trybie wykorzystuje się specjalne rejestry procesora tzw. rejestry indeksowe (ang. index registers), które zawierają przesunięcie, który trzeba dodać do adresu istniejącego w rozkazie aby wyliczyć adres finalny operandu. Ten tryb adresowania pozwala przesunąć adres zawarty w rozkazie o wartość rejestru indeksowego. Używając tego trybu we wszystkich rozkazach programu, można osiągnąć możliwość wykonania programu przy załadowaniu w dowolne miejsce pamięci.

Adresowanie rejestrowe stosuje się, gdy dana dla rozkazu jest przechowywana w rejestrze. Część adresowa rejestru zawiera wtedy jedno lub więcej pól, w których znajdują się identyfikatory rejestrów. Adresowanie rejestrowe jest często stosowane w tym samym rozkazie razem z innymi trybami adresowania dotyczącymi pamięci operacyjnej. Wówczas rozkaz dotyczy zarówno pamięci operacyjnej jak i rejestrów procesora.

Adresowanie względne polega na modyfikacji adresu zawartego w rozkazie przez aktualną zawartość licznika rozkazów. Ten tryb adresowania dostarcza innego sposobu osiągnięcia dynamicznej przesuwalności adresów dostępu do danych, tj. gdy nie chcemy lub nie możemy znać przesunięcia całości programu w stosunku do adresu zerowego.

Przy adresowaniu pośrednim rozkaz zawiera adres komórki pamięci operacyjnej, w której zawarty jest finalny adres operandu rozkazu. W tym przypadku komórka pamięci wskazana przez adres rozkazu pośredniczy w określeniu finalnego adresu. Ten tryb stosujemy, gdy chcemy, aby finalny adres operandu rozkazu mógł być dynamicznie wstawiony do komórki pośredniczącej w adresowaniu w czasie wykonywania programu. Może tak być, gdy ten adres zależy od jakichś testów na wyniku operacji poprzedzającego rozkazu.

Rejestry arytmetyczne

* ***AX*** – Akumulator (*Accumulator*)  
  Rejestr ten bezpośrednio współpracuje z jednostką arytmetyczno-logiczną.

Niektóre operacje, których argumenty znajdują się w akumulatorze, wykonywane są szybciej niż ich odpowiedniki wykorzystujące inne rejestry. Takie rozkazy jak: mnożenie, dzielenie i operacje wejścia/wyjścia wymagają użycia akumulatora do przechowywania argumentu bądź też zapisu wyniku.

* ***BX*** – Baza (*Base*)  
  Rejestr ten może być używany do adresowania argumentu, znajdującego się w pamięci, stanowiąc bazę do obliczania adresu.
* ***CX*** – Licznik (*Counter*)  
  Rejestr ten jest używany jako licznik w operacjach łańcuchowych oraz pętlach. Po każdej iteracji jego zawartość jest automatycznie dekrementowana. W rozkazach przesunięć, rejestr CL (mniej znaczący bajt rejestru CX), wykorzystywany jest jako licznik bitów.
* ***DX*** – Dane (*Data*)  
  Rejestr ten jest wykorzystywany w niektórych operacjach arytmetycznych do przechowywania części argumentu lub wyniku operacji (mnożenie i dzielenie 16-bitowe). Zawiera on także adres urządzenia w operacjach wejścia/wyjścia.

Rozkazy przesłań

* MOV- przesłanie bajtu lub słowa
* PUSH- przesłanie słowa na stos
* POP- odczyt słowa ze stosu
* XCHG- zamiana miejscami bajtu lub słowa
* XLAT- pobranie bajtu z tablicy