Набор от инструкции за микропроцесор 8086

Наборът от инструкции включва инструкциите, които микропроцесорът може да изпълнява .

Инструкциите на микропроцесор 8086 позволяват обработка на 8 и 16- битови данни. Тези данни могат да бъдат :

- Двоични числа без знак;
- Двоични числа с знак;

Набор от инструкции за микропроцесор 8086

- Десетични числа в опакован или в разопакован формат;
- Символи;
- Ред от символи.

Набор от инструкции за микропроцесор 8086

Обработката на данни, представени в друг формат – например числа с плаваща запетая, се извършва с помощта на специални програми или устройство (копроцесор).

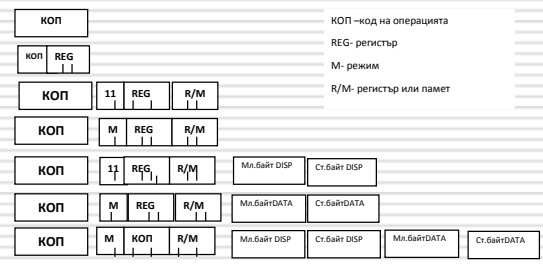
Набор от инструкции за микропроцесор 8086

Кодът на инструкциите може да заема от 1 до 8 байта. Първите един или два байта съдържат кода на операцията, а останалите – операндите, адресите на операндите и отместването. Съответно инструкциите могат да бъдат без адрес, едно и двуадресни.

Набор от инструкции за микропроцесор 8086

На фиг. 4.1. са дадени примери на формата на инструкциите на 8086. Инструкциите съдържат едно или няколко полета. Във всички инструкции задължително присъства поле с кода на операцията, която трябва да се извърши. Някои инструкции съдържат само това поле. Останалите полета определят местоположението на операндите и начина на изчисляване на адресите им (режима на адресация).

Примерен формат на кода на инструкциите на 8086



Набор от инструкции за микропроцесор 8086

Полето REG определя регистъра на микропроцесора, който участва в операцията. Полетата M и R/M определят режима на адресация на операндите. Полето DISP задава 8 или 16 – битово отместване, което трябва да се добави към регистъра, който се използва в зададения чрез M и R/M режим на адресиране. Полето DATA съдържа 8 или 16-битово число (данни), което участва непосредствено в операцията.

7

Набор от инструкции за микропроцесор 8086

Общия брой на използваните кодове на инструкции е 11986. На групите от инструкции, които изпълняват еднотипни операции на данните, са присвоени мнемонични означения, като броя на тези групи е 111. Мнемоничните означения се използват при съставяне на програмата за м икропроцесорната система.

8

Набор от инструкции за микропроцесор 8086

Инструкциите могат да се разделят на 5 класа- за обмен на данни, аритметически, логически, за управление на програмата и за управление на състоянието.

9

Набор от инструкции за микропроцесор 8086

- **Инструкции за обмен на данни**

Общият им брой е 2 695. Към тях се отнасят :

- *Еднопосочен обмен от вида регистър – регистър;*
- *Еднопосочен обмен от вида регистър-памет;*
- *Еднопосочен обмен от вида памет – памет.*

10

Набор от инструкции за микропроцесор 8086

Кодът на тези инструкции има общ вид **MOV приемник, източник,** където за източник могат да бъдат 8 или 16 – битови данни, включени в кода на инструкцията , 8 или 16- битовото съдържание на регистър или на клетка от паметта, а премник- съответно 8 или 16-битов регистър или клетка от паметта.

11

Набор от инструкции за микропроцесор 8086

Двупосочен обмен (размяна на съдържанието) между два регистъра или между регистър и клетка от паметта.

Обмен с входно- изходни устройства (обмен с интерфейса). При тези инструкции се извършва обмен на данни между входно – изходния интерфейс и регистър AL (8 бита) или AX(16 бита).

12

Набор от инструкции за микропроцесор 8086

- **Инструкции за аритметически операции**

Общият им брой е 5 198. В този клас влизат следните инструкции:

- събиране;
- събиране с пренос;
- изваждане;
- изваждане с заем;

Набор от инструкции за микропроцесор 8086

- сравнение;
- умножение;
- деление.

Набор от инструкции за микропроцесор 8086

Кодът на тези инструкции има общ вид **операция, първи операнд, втори операнд**, където за първи операнд може да бъде 8 или 16 – битовото съдържание на регистър или на клетка от паметта, а за втори операнд – 8 или 16- битови данни, включени в кода на инструкцията, или съдържанието съответно на 8 или 16 – битов регистър или клетка от паметта:

Набор от инструкции за микропроцесор 8086

- *инкремент* (увеличение с единица);
- *декремент* (намаление с единица);
- *преобразуване в допълнителен код*.

Общият им вид е **операция, операнд,** където за операнд може да бъде 8 или16 – битовото съдържание на регистър или клетка от паметта.

корекции – десетична или символна *(ASCII)*

Корекциите се извършват върху съдържанието на регистри AL или AX.

Набор от инструкции за микропроцесор 8086

- **Инструкции за логически операции**

Общият им брой е 3 904. В този клас влизат следните инструкции:

- *Побитово логическо умножение – операция логическо* **И (AND)**;
- *Побитово логическо умножение* без съхранение на резултата **(TEST);**
- *Побитово логическо обединение*- операция логическо **ИЛИ (OR);**
- *Побитово сумиране по mod2* –операция изключващо **ИЛИ (XOR).**

Набор от инструкции за микропроцесор 8086

Общият им вид е ***операция, операнд,*** където за операнд може да бъде 8 или 16 –битови данни, включени в кода на инструкцията или съдържанието на регистър или на клетка от паметта.

Набор от инструкции за микропроцесор 8086

- *Инвертиране* – логическо **НЕ** (NOT);
- Побитово изместване вляво;
- Побитово изместване надясно;
- Ротация наляво;
- Ротация надясно.

Общият им вид е операция, операнд, където за операнд може да бъде 8 или 16 –битово съдържание на регистър или на клетка от паметта.

19

Набор от инструкции за микропроцесор 8086

- **Инструкции за управление на програмата**

Общият им брой е 145. В този клас влизат следните инструкции:

- *Безусловен преход;*
- *Преход към подпрограма;*
- *Връщане на подпрограма;*
- *Условни преходи (преход при изпълнение на определено условие);*

20

Набор от инструкции за микропроцесор 8086

- *Цикли;*
- *Многократно изпълнение;*
- *Преход към подпрограмата за обработка на прекъсване;*
- *Връщане от подпрограма за обработка на прекъсване.*

21

Набор от инструкции за микропроцесор 8086

Адресът, към който трябва да се извърши прехода, може да бъде включен в кода на инструкцията като 8 или 16-битово отместване или да е записан в регистър или клетка от паметта. При извършване на условните преходи се анализира състоянието на един или няколко от флаговете на PSV.

22

Набор от инструкции за микропроцесор 8086

- **Инструкции за управление на състоянието**

Общият им брой е 44. В този клас влизат следните инструкции:

- *Индивидуално установяване на флаговете в 0 или1;*
- *Спиране на изпълнението на програмата (*HLT*);*
- NOP- *празна операция* (не се променят никакви данни);
- *За работа в многопроцесорни системи.*

23

Разработка на програми за микропроцесорна система

Физически кодът на инструкциите представлява число от 1 до 8 байта.

Този код се записва в последователни клетки на паметта в двоичен вид и се **нарича машинен код.** Това е действителният”език”, който се разбира от микропроцесора и с който той може да работи.

24

Разработка на програми за микропроцесорна система

Разработени са няколко програмни езика, в които групи от инструкции или даже цели често срещани подпрограми се заменят със словесни изрази. По такъв начин писането и проверката на програми става по по-разбираем начин.

25

Разработка на програми за микропроцесорна система

Най-близък до машинният език е асемблерния език или още както се нарича символичен език. При този език за описание на инструкциите, вместо двоичния машинен код се използват буквени съкращения на думи, които в англ. език съответстват на същността на изпълняваната инструкция.

26

Разработка на програми за микропроцесорна система

Например следната 3-байтова инструкция към микропроцесора:" запиши числото 50 000 в регистър AX", ще има следния вид:

В машинен код 10111 000 11 000011 01010000;

На асемблер MOV AX, 50 000.

27

Разработка на програми за микропроцесорна система

Очевидна е практическата невъзможност програмист да пише реална програма, използвайки машинен език. Освен замяната на машинните кодове на инструкциите със символи, в асемблерния език е възможно задаването на произволни имена за всяка клетка от паметта (или регистър) според предназначението и.

28

Разработка на програми за микропроцесорна система

Също така може да се задават и подходящи имена на адресите на преходите в програмата и на подпрограмите. Това прави процеса на програмиране много по-лесен и се допускат по-малко грешки. След като бъде написана програмата на език Асемблер, тя се прекодира в машинен език.

29

Разработка на програми за микропроцесорна система

Процесът на превеждане на асемблерска програма на машинен език се нарича ***асемблиране*** и се извършва от специална програма, наречена асемблер.

30

Разработка на програми за микропроцесорна система

Разработени са много езици от по-високо ниво, които правят програмирането още по-лесно и разбираемо. Те включват и библиотеки с готови програми и подпрограми за често срещани приложения. За микропроцесорните системи в момента най-често се използват C++, Pascal, Java,Visual Basic и други.

31

Разработка на програми за микропроцесорна система

По принцип използването на езици от високо ниво позволява да се пишат програми без да има значение за какъв точно процесор са предназначени. В процесора на прекодиране на програмата в машинен език се използват така наречените програми ***компилатори,*** предназначени за конкретен набор от инструкции на микропроцесор.

32

Програмиране на Асемблер

33

Понятие за програмния език Асемблер

Програмният език Асемблер е форма на представяне на инструкциите на микропроцесора във вид, по-близък до човешкия. Машинният език (кода на инструкциите) и езика Асемблер са еквивалентни, но програмирането на втория е неизмеримо по-лесно и понятно.

34

Понятие за програмния език Асемблер

Положително качество на Асемблера е и това, че чрез него могат да се управляват действията на микропроцесора по отделни операции и програмите могат да са максимално ефективни в сравнение с други езици от по-високо ниво. Това е една от основните причини за предпочитането на езика Асемблер в случаите, когато трябва да се постигне минимален обем на програмата и минимално време за изпълнението ѝ.

35

Структура на програмата

Текстът на програмата се пише в обикновен текстов файл. Разширението на файла трябва да бъде asm. Файлът се състои от оператори на езика Асемблер, всеки от който заема един ред. Различават се два типа оператори - команди и директиви.

36

Структура на програмата

Първите при компилиране (преобразуване в машинен код) се прекодират в двоичен код на инструкциите от машинния език. Те са предназначени за зареждане в постоянната памет на микропроцесорната система. Файлът след компилация, съдържащ двоичния код на инструкциите, се записва с разширение com или exe. Операторите от втория тип - директивите, управляват процеса на компилиране.

Формат на операторите на Асемблера

Общият вид на операторите на Асемблера има следния вид:

[Етикет[:]] Мнемоника [Операнд1 [{,Операнд2}]] [;Коментар]

Фрагментите, които са в квадратни скоби не са задължителни, а фрагментите във фигурни скоби { } могат да се използват веднъж или повече пъти. Между фрагментите трябва да има най-малко един интервал.

Формат на операторите на Асемблера

Етикетът представлява идентификатор, който е свързан с адреса на първия байт от кода на инструкцията, определен чрез мнемониката. Коментарът може да бъде произволен и предназначението му е да поясни инструкцията. Коментарът задължително трябва да започва с точка и запетая (;).

Формат на операторите на Асемблера

Етикетите се използват за посочване на началните адреси на условни и безусловни преходи и на подпрограми. Те могат да съдържат следните символи: "A" до "Z", "a" до "z","_", "@", "0" до "9". Етикетът не може да започва с цифра. Имената на етикетите не трябва да съвпадат с имената на регистри, мнемоника на инструкции или други, резервирани от Асемблера изрази. Не може да има два еднакви етикета. При компилиране всеки етикет се заменя с конкретен адрес и затова всеки етикет трябва да е уникален, различен от другите. Условен или безусловен преход към операнда с етикет (определения чрез етикета адрес) може да се извършва от много инструкции.

Формат на операторите на Асемблера

Полето на мнемониката е основно в програмния ред на Асемблера. Мнемониката на инструкциите се компилира непосредствено в двоичен код.

Операндите представляват числа или символни изрази. Те определят данните, над които ще се изпълнява операцията, заложена в кода на инструкцията.

Имена в езика Асемблер

За удобство при съставяне и разчитане на програмата в Асемблера на често срещани изрази, адреси на клетки, числа могат да се присвояват по-разбираеми имена, обвързани с предназначението им в програмата. Присвояването на имена става с директивата EQU, която има следния формат:

Име EQU израз

Името, както етикета, трябва да бъде уникално за програмата (всяко име трябва да е различно от използваните имена и изрази).

Имена в езика Асемблер

Пример: Използване на директивата EQU за задаване на стойността на тарифите в програма за таксиметров апарат:

```
TAX_den    EQU   30 ;дневна тарифа 30 ст.
TAX_ve4er EQU   40 ;нощна тарифа 40 ст.
```

коментар
число
директива
име

Имена в езика Асемблер

Използването на име (TAX_den) вместо числото 30 има най-малко две предимства (освен очевидната яснота). Първото е, че тарифата може да се използва в много инструкции и при промяна не трябва да се проверяват и променят всичките инструкции, а се променя само числото след директивата EQU. Например при промяна на дневната тарифа от 30 на 35 ст. ще се промени само един ред от програмата:

TAX_den EQU 35 ; честито нова тарифа!

Имена в езика Асемблер

Асемблерът ще промени без да сгреши всички инструкции, в които се използва тази тарифа. Втората полза от използване на имена е тази, че числото 30 може да се използва в много инструкции и при проверка и промяна на програмата не е ясно това числа за какво се отнася, докато името TAX_den е много по-разбираемо и по-рядко ще се допусне грешка.

Директиви на езика Асемблер за определяне на типа данни

Директивите за резервиране на клетки от паметта имат следния формат:

Име Директива Операнд [{, Операнд}].

Директиви на езика Асемблер за определяне на типа данни

Директивата определя размера на данните, за които се резервират клетките и може да бъде:

- DB - за резервиране на 8-битови данни (байтове);
- DW - за резервиране на 16-битови данни (думи);
- DD - за резервиране на двойни думи (4 байта);
- DQ - за резервиране на четворни думи (8 байт);
- DT - за резервиране на 10-байтови полета.

Директиви на езика Асемблер за определяне на типа данни

Примери:

Pozdrav DB 'Zdravei!';
Резервира поле в паметта, което съдържа 8-битовите кодовете на буквите от низа 'Zdravei!'

Набор от инструкции на микропроцесор Intel 8086

От основно значение за програмиста е наборът от инструкции, които микропроцесорът може да изпълнява. Дължината на инструкциите е от 1 до 6 байта.

49

Набор от инструкции на микропроцесор Intel 8086

Двоичният код на всяка инструкция се разделя на група от битове (полета) - поле на кода на операцията и едно или няколко полета за операндите. Кодът в полетата за операндите определя данните, които се използват в операцията. Те могат да бъдат непосредствено зададено число (или кодиран символ), адрес на регистър или клетка от паметта, косвен указател на адрес или данни. Микропроцесор Intel 8086 реализира следния набор от инструкции, разделени на групи в съответствие с изпълняваните операции:

50

Набор от инструкции на микропроцесор Intel 8086

- *Инструкции за прехвърляне на данни:*

IN, LAHF, LDS, LEA, LES, MOV, OUT, POP, POPF, PUSH, PUSHF, SAHF, XCHG, XLAT.

- *Инструкции за аритметически операции:*

AAA, AAD, AAM, AAS, ADC, ADD, CBW, CMP, CWD, DAA, DAS, DEC, DIV,

IDIV, IMUL, INC, MUL, NEG, SBB, SUB.

- *Инструкции за логически операции:*

AND, NOT, OR, RCL, RCR, ROL, ROR, SAL, SAR, SHL, SHR, TEST, XOR.

51

Набор от инструкции на микропроцесор Intel 8086

- *Инструкции за условен преход:*

JA - преход при по-голямо;
JAE - преход при по-голямо или равно;
JB - преход при по-малко;
JBE - преход при по-малко или равно;
JC - преход при наличие на пренос;
JCXZ - преход ако регистър CX е равен на 0;
JE - преход при равенство;
JG - преход при по-голямо;

52

Набор от инструкции на микропроцесор Intel 8086

JGE - преход при по-голямо или равно;
JL - преход при по-малко;
JLE - преход при по-малко или равно;
JNA - преход при не по-голямо;
JNAE - преход при не по-голямо и не равно;
JNB - преход при не по-малко;
JNBE - преход при не по-малко и не равно;
JNC - преход при липса на пренос;
JNE - преход при не равно;
JNG - преход при не по-голямо;

53

Набор от инструкции на микропроцесор Intel 8086

JNGE - преход при не по-голямо и не равно;
JNL - преход при не по-малко;
JNLE - преход при не по-малко и не равно;
JNO - преход при липса на препълване;
JNP - преход при нечетно;
JNS - преход при положителен резултат;
JNZ - преход при не нулев резултат;
JO - преход при препълване;
JP - преход при четно;
JPE - преход при четно;
JPO - преход при нечетно;

54

Набор от инструкции на микропроцесор Intel 8086

JS - преход при отрицателен резултат;
JZ - преход при нула;
LOOP - преход по брояч;
LOOPE - преход докато е равно;
LOOPNE - преход докато не е равно;
LOOPNZ - преход докато резултата не е нула;
LOOPZ - преход докато е нула.

- Инструкции за управление:

CALL - извикване на подпрограма;
JMP - безусловен преход;
RET - връщане от подпрограма.

Набор от инструкции на микропроцесор Intel 8086

- *Инструкции за обработка на низове:*

CMPS, CMPSB, CMPSW, LODS, LODSB, LODSW, MOVS, MOVSB, MOVSW,
REP, REPE, REPNE, REPNZ, SCAS, SCASB, SCASW, STOS, STOSB, STOSW.

- *Инструкции за прекъсване:*

INT - прекъсване;
INTO - прекъсване при препълване;
IRET - връщане след обработка на прекъсване.

- *Инструкции за управление на състоянието на микропроцесора:*

CLC, CLD, CLI, CMC, ESC, HLT, LOCK, NOP, STC, STD, STI, WAIT.

Компилиране и тестване на програмите на Асемблер

За да се преобразува текстовия файл с програмата на Асемблер във файл, който съдържа машинния код на инструкциите, се използва компилатор. В процеса на компилация се извършва логически и синтактически анализ на изходния текстов файл. Според резултатите от този анализ се генерира машинния код (кода на инструкциите).

Компилиране и тестване на програмите на Асемблер

След компилиране програмите се тестват чрез така наречените дебъгери. Те представляват специално създадени програми, които изпълняват инструкциите от тестваната програма в контрулируема от програмиста среда. Чрез тях се анализира постъпково изпълнението на инструкциите, промяната на регистрите и на флаговете, и се извежда на монитора съдържанието им.

Компилиране и тестване на програмите на Асемблер

Примери

Задача: Да се разработи програма, която да събира две числа, записани в регистри AX и CX.

Програма се състои в това, че с инструкция MOV AX,5 в регистър AX се записва първото число (първия операнд). Със следващата инструкция в регистър CX се записва вторият операнд (числото 6). Чрез инструкцията ADD AX, CX числата в двата регистъра се събират и резултатът се записва в регистър AX.

Текстът на програмата има следния вид:

```
MOV   AX,5
MOV   CX,6
ADD   AX,CX
```


Край на част 3