Ассемблер Intel 8086

Команды пересылки данных

Базовый набор команд для пересылки данных между регистрами, памятью и устройствами ввода-вывода:

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

Ассемблер Intel 8086

Команды пересылки данных: mov

Общий формат: mov Operand1, Operand2
Действие: Operand1 := Operand2
Описание: команда предназначена для передачи значения второго операнда первому. В зависимости от описания операндов пересылается слово или байт. Если операнды описаны по-разному или нельзя однозначно определить размер операнда, используется один из атрибутных операторов: **byte ptr** или **word ptr.**
Особенность: запрещены пересылки из ячейки памяти в ячейку памяти.
Примеры:

```
mov AX, BX
mov AH, Mem_DB
mov Mem_DW, CX
mov DS, AX
mov DX, ES
mov Value_SS, SS
mov CH, 200
mov Mem_DB, 200
mov word ptr [bx], 1       ; mov byte ptr [bx], 1
```

Ассемблер Intel 8086

Команды пересылки данных: push, pop

Общий формат: push Operand
Действие: занесение операнда в стек.
Описание: содержимое указателя стека SP уменьшается на 2, после чего по адресу SS:SP заносится двухбайтовый операнд.

Общий формат: pop Operand
Действие: извлечение слова из стека.
Описание: слово, содержащееся по адресу SS:SP, пересылается по адресу операнда-приёмника, после чего содержимое регистра SP увеличивается на 2.

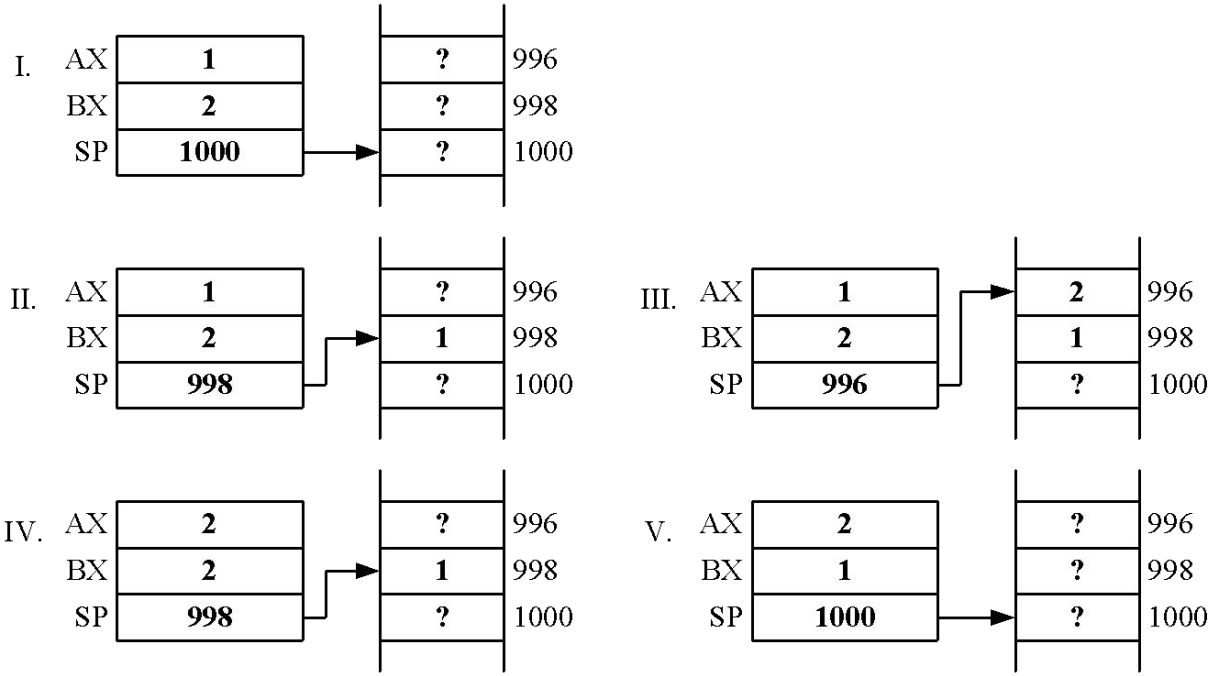
Особенность команд push и pop: операндами могут быть только 16-разрядные регистры (кроме CS) или ячейки памяти.

Ассемблер Intel 8086

Команды пересылки данных: push, pop

Пример: поменять содержимое двух регистров (AX, BX) местами, используя стек.

```
Push AX      ; поместить AX в стек
Push BX      ; поместить BX в стек
Pop AX       ; извлечь информацию …
Pop BX       ; … в нужном для решения задачи порядке
```



Ассемблер Intel 8086

Команды пересылки данных: xchg

Общий формат: xchg Operand1, Operand2
Действие: Operand1 и Operand2 обмениваются значениями.
Особенность: запрещён обмен значениями ячейки памяти с ячейкой памяти.
Примеры:

```
Xchg AX, Mem_DW
Xchg AX, BX
```

Ассемблер Intel 8086

Команды пересылки данных: lea, lds, les

Общий формат: lea Reg16, Mem
Действие: загрузка в регистр Reg16 смещения ячейки памяти Mem.
Особенность: Reg16 не может быть сегментным регистром.

Общий формат: lds Reg16, Mem32
Действие: загрузка далёкого адреса из ячейки памяти Mem32 в регистры DS:Reg16, при этом младшая часть ячейки записывается в Reg16, а старшая – в регистр DS.
Особенность: Reg16 не может быть сегментным регистром.

Общий формат: les Reg16, Mem32
Действие: загрузка далёкого адреса из ячейки памяти Mem32 в регистры ES:Reg16, при этом младшая часть ячейки записывается в Reg16, а старшая – в регистр ES.
Особенность: Reg16 не может быть сегментным регистром.

Ассемблер Intel 8086

Арифметические операции

Базовый набор арифметических операций для работы с целыми числами:

AAA	**AAD**	**AAM**	**AAS**
ADC	**ADD**	**CBW**	**CMP**
CWD	**DAA**	**DAS**	**DEC**
DIV	**IDIV**	**IMUL**	**INC**
MUL	**NEG**	**SBB**	**SUB**

Ассемблер Intel 8086

Арифметические операции: add, adc

Общий формат: add Operand1, Operand2
Действие: Operand1 := Operand1 + Operand2
Описание: сложение двух целых чисел со знаком или без знака.
Особенность: Operand1 и Operand2 не могут быть ячейками памяти одновременно.

Общий формат: adc Operand1, Operand2
Действие: Operand1 := Operand1 + Operand2 + CF.
Описание: сложение двух целых чисел со знаком или без знака, при этом в операции принимает участие и флаг переноса из регистра флагов.
Особенность: Operand1 и Operand2 не могут быть ячейками памяти одновременно.

Пример: сложение двух 32-разрядных чисел (операнд1 – BX:AX, операнд2 – DX:CX)

```
Add AX, CX
Adc BX, DX
```

Ассемблер Intel 8086

Арифметические операции: sub, sbb

Общий формат: sub Operand1, Operand2
Действие: Operand1 := Operand1 – Operand2
Описание: вычитание двух целых чисел со знаком или без знака.
Особенность: Operand1 и Operand2 не могут быть ячейками памяти одновременно.

Общий формат: sbb Operand1, Operand2
Действие: Operand1 := Operand1 – Operand2 – CF.
Описание: вычитание двух целых чисел со знаком или без знака, при этом в операции принимает участие и флаг переноса из регистра флагов.
Особенность: Operand1 и Operand2 не могут быть ячейками памяти одновременно.

Ассемблер Intel 8086

Арифметические операции: mul, imul

Общий формат: mul Operand
Описание: умножение двух целых чисел без знака. Первый операнд хранится в регистре AL (при умножении байтов) или AX (при умножении слов), второй операнд задаётся в команде. Результат помещается в регистр AX или в регистры DX:AX соответственно.
Особенность: разрядность операции задаётся разрядностью Operand.

Общий формат: imul Operand
Описание: умножение двух целых чисел со знаком. Первый операнд хранится в регистре AL (при умножении байтов) или AX (при умножении слов), второй операнд задаётся в команде. Результат помещается в регистр AX или в регистры DX:AX соответственно.
Особенность: разрядность операции задаётся разрядностью Operand.

Пример:

```
Mov AX, 100h
Mul AX          ; = AX*AX, результат – DX:AX
```

Ассемблер Intel 8086

Арифметические операции: div, idiv

Общий формат: div Operand
Описание: деление двух целых чисел без знака.
Особенность: разрядность операции задаётся разрядностью делителя – Operand.

Общий формат: idiv Operand
Описание: деление двух целых чисел со знаком.
Особенность: разрядность операции задаётся разрядностью делителя – Operand.

Делимое	Делитель	Частное	Остаток
AX	8 разрядов	AL	AH
DX:AX	16 разрядов	AX	DX

Ассемблер Intel 8086

Арифметические операции: cmp

Общий формат: cmp Operand1, Operand2
Описание: выполняется сравнение двух операндов путём вычитания второго операнда из первого. Результат операции не записывается, вместо этого устанавливаются значения флагов в регистре флагов процессора.

Ассемблер Intel 8086

Команды передачи управления

Базовый набор команд передачи управления:

CALL JMP RET JA
JAE JB JBE JC
JCXZ JE JG JGE
JL JLE JNA JNAE
JNB JNBE JNC JNE
JNG JNGE JNL JNLE
JNO JNP JNS JNZ
JO JP JPE JPO
JS JZ LOOP LOOPE
LOOPNE LOOPNZ LOOPZ

Ассемблер Intel 8086

Команды передачи управления: jmp

Общий формат: jmp Target
Описание: выполняется передача управления по адресу, заданному параметром команды Target. Адрес может задаваться как напрямую (с помощью метки), так и с помощью регистров и ячеек памяти.
Особенность: переход внутри одного сегмента задаётся только смещением, переход между сегментами задаётся полным адресом.
Виды безусловных переходов:
1) прямой короткий (пример: jmp short Point1);
2) прямой ближний (пример: jmp near ptr Point2);
3) прямой дальний (пример: jmp far ptr Point3);
4) косвенный ближний (пример: jmp word ptr [SI+2]);
5) косвенный дальний (пример: jmp dword ptr [DX]).

Другие примеры:
jmp BX
jmp SkipAdd
jmp dword ptr AddrTable[SI+2]

Ассемблер Intel 8086

Команды передачи управления: j?*

Общий формат: j?* Target
Описание: при выполнении некоторого условия выполняется передача управления по адресу, заданному параметром команды Target. Адрес может задаваться как напрямую (с помощью метки), так и с помощью регистров и ячеек памяти.
Особенность: выполняется только короткий переход, т.е. адрес, заданный операндом Target, должен быть расположен в диапазоне от -128 до +127 байтов от адреса, хранимого в регистре IP.

При написании команд удобно пользоваться сокращениями:

A – above (выше)
B – below (ниже)
C – carry (перенос)
E – equal (равно)
G – greater (больше)
L – less (меньше)
N – not (не)
O – overflow (переполнение)
P – parity (паритет)
S – sign (знак)
Z – zero (ноль)

Ассемблер Intel 8086

Команды передачи управления: j?*

Примеры конструирования команд условного перехода:
ja, jnbe – переход, если выше/переход, если не ниже и не равно
jg, jnle – переход, если больше/переход, если не меньше и не равно
jc – переход, если установлен флаг переноса
jz, je – переход, если ноль/переход, если равно
…

Дополнительная команда условного перехода:
jcxz – переход, если CX=0

Ассемблер Intel 8086

Команды передачи управления: loop

Общий формат: loop LoopLabel
Описание: команда предназначена для организации циклических вычислений. Количество повторений задаётся в регистре CX. Выход из цикла происходит, когда при очередной проверке CX оказывается равным нулю.
Алгоритм:

```
CX := CX – 1;
if CX <> 0 then jmp LoopLabel;
```

Особенность: выполняется только короткий переход, т.е. адрес, заданный операндом LoopLabel, должен быть расположен в диапазоне от -128 до +127 байтов от адреса, хранимого в регистре IP.

Пример:

```
    mov AX, 0
    mov CX, 100
AXIncLoop:
    add AX, 1
    loop AXIncLoop
```

Ассемблер Intel 8086

Команды передачи управления: loope, loopz

Общий формат: loope LoopLabel
Описание: команда предназначена для организации циклических вычислений. Количество повторений задаётся в регистре CX. Кроме значения регистра CX данная команда анализирует содержимое флага Z регистра флагов. Цикл выполняется, пока содержимое регистра CX не равно нулю и флаг ZF равен 1.
Алгоритм:

CX := CX – 1;
if (CX <> 0) and (ZF = 1) then jmp LoopLabel;

Особенность: выполняется только короткий переход, т.е. адрес, заданный операндом LoopLabel, должен быть расположен в диапазоне от -128 до +127 байтов от адреса, хранимого в регистре IP.

loopz – синоним команды loope.

Ассемблер Intel 8086

Команды передачи управления: loopne, loopnz

Общий формат: loopne LoopLabel
Описание: команда предназначена для организации циклических вычислений. Количество повторений задаётся в регистре CX. Кроме значения регистра CX данная команда анализирует содержимое флага Z регистра флагов. Цикл выполняется, пока содержимое регистра CX не равно нулю и флаг ZF равен нулю.
Алгоритм:

CX := CX – 1;
if (CX <> 0) and (ZF = 0) then jmp LoopLabel;

Особенность: выполняется только короткий переход, т.е. адрес, заданный операндом LoopLabel, должен быть расположен в диапазоне от -128 до +127 байтов от адреса, хранимого в регистре IP.

loopnz – синоним команды loopne.

Ассемблер Intel 8086

Команды передачи управления: call

Общий формат: call ProcName
Описание: команда вызова подпрограммы.
Алгоритм:

```
if FAR CALL then
  begin
      PUSH CS;
      CS := dest_seg
  end;
PUSH IP;
IP := dest_offset;
```

Особенность: подпрограмма может быть вызвана как напрямую (с использованием метки – имени подпрограммы), так и косвенно (адрес находится в регистре или ячейке памяти).

Ассемблер Intel 8086

Команды передачи управления: ret

Общий формат: ret [PopBytes]
Описание: команда возврата из подпрограммы. Необязательный параметр PopBytes указывает, сколько байтов необходимо освободить из стека при выходе из подпрограммы (например, освобождение стека от входных параметров, переданных подпрограмме).
Алгоритм:

```
POP IP
if FAR RETURN then POP CS;
SP := SP + PopBytes;
```

Особенность: тип возврата из подпрограммы (дальний или близкий) задаётся программистом при описании подпрограммы.

Ассемблер Intel 8086

Логические операции

Базовый набор команд, реализующих логические операции:

AND NOT OR XOR

X	Y	X and Y	not X	X or Y	X xor Y
0	0	0	1	0	0
0	1	0	1	1	1
1	0	0	0	1	1
1	1	1	0	1	0

Ассемблер Intel 8086

Логические операции

Общий формат: and Destination, Source
Описание: Destination := Destination AND Source

Общий формат: or Destination, Source
Описание: Destination := Destination OR Source

Общий формат: xor Destination, Source
Описание: Destination := Destination XOR Source

Общий формат: not Destination
Описание: Destination := NOT(Destination)

Ассемблер Intel 8086

Команды работы с битами

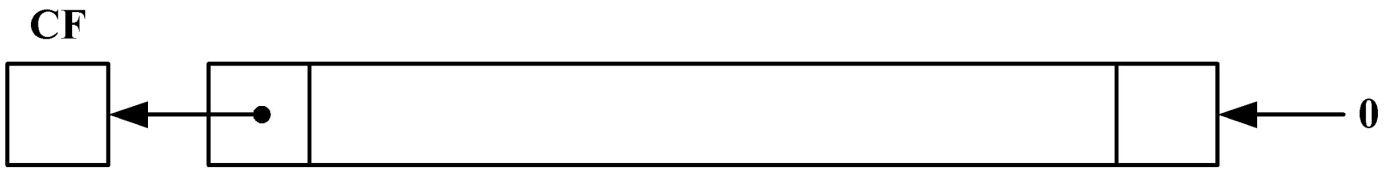
Базовый набор команд, реализующих операции над отдельными битами:

RCL	**RCR**	**ROL**	**ROR**
SAL	**SAR**	**SHL**	**SHR**
TEST			

Ассемблер Intel 8086

Команды работы с битами: sal, shl

Общий формат: sal/shl Operand, ShiftCount
Описание: сдвиг арифметический (sal) или логический (shl) влево на заданное количество битов. В качестве операнда может быть слово или байт.
Особенности: при сдвиге выпадающий бит попадает в CF – флаг переноса. Если сдвиг был более, чем на один бит, то во флаге CF хранится последний выпавший бит. В качестве операнда ShiftCount обычно выступает регистр CL.

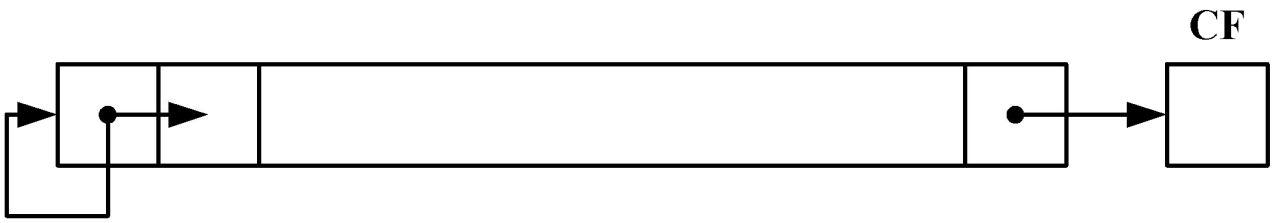


Примеры:
sal AX, 1
shl WordVar, CL

Ассемблер Intel 8086

Команды работы с битами: sar

Общий формат: sar Operand, ShiftCount
Описание: сдвиг арифметический вправо на заданное количество битов. В качестве операнда может быть слово или байт.
Особенности: при сдвиге выпадающий бит попадает в CF – флаг переноса. Если сдвиг был более, чем на один бит, то во флаге CF хранится последний выпавший бит. В качестве операнда ShiftCount обычно выступает регистр CL. При арифметическом сдвиге знак операнда сохраняется.

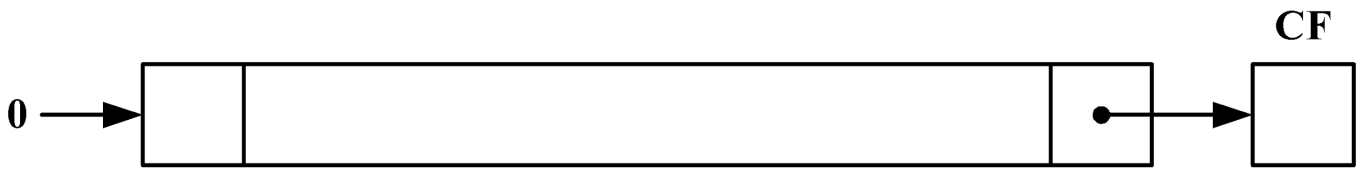


Примеры:
sar AX, 1
sar ByteVar, CL

Ассемблер Intel 8086

Команды работы с битами: shr

Общий формат: shr Operand, ShiftCount
Описание: сдвиг логический вправо на заданное количество битов. В качестве операнда может быть слово или байт.
Особенности: при сдвиге выпадающий бит попадает в CF – флаг переноса. Если сдвиг был более, чем на один бит, то во флаге CF хранится последний выпавший бит. В качестве операнда ShiftCount обычно выступает регистр CL. При логическом сдвиге в операции участвует весь операнд.

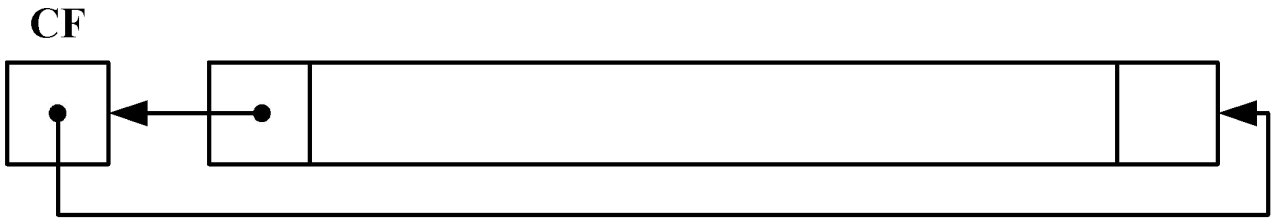


Примеры:
shr AH, 1
shr ByteVar, CL

Ассемблер Intel 8086

Команды работы с битами: rcl

Общий формат: rcl Operand, ShiftCount
Описание: циклический сдвиг влево через флаг переноса на заданное количество битов. В качестве операнда может быть слово или байт.
Особенности: при сдвиге выпадающий бит попадает в CF, а старое значение этого флага передаётся в освободившийся после сдвига правый бит операнда. В качестве операнда ShiftCount обычно выступает регистр CL.

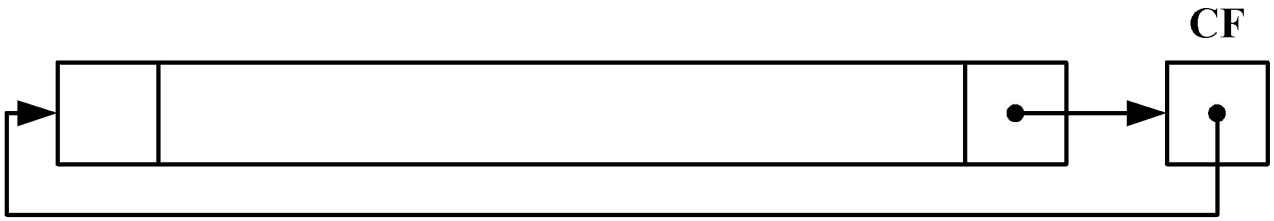


Примеры:
rcl AH, 1
rcl ByteVar, CL

Ассемблер Intel 8086

Команды работы с битами: rcr

Общий формат: rcr Operand, ShiftCount
Описание: циклический сдвиг вправо через флаг переноса на заданное количество битов. В качестве операнда может быть слово или байт.
Особенности: при сдвиге выпадающий бит попадает в CF, а старое значение этого флага передаётся в освободившийся после сдвига левый бит операнда. В качестве операнда ShiftCount обычно выступает регистр CL.

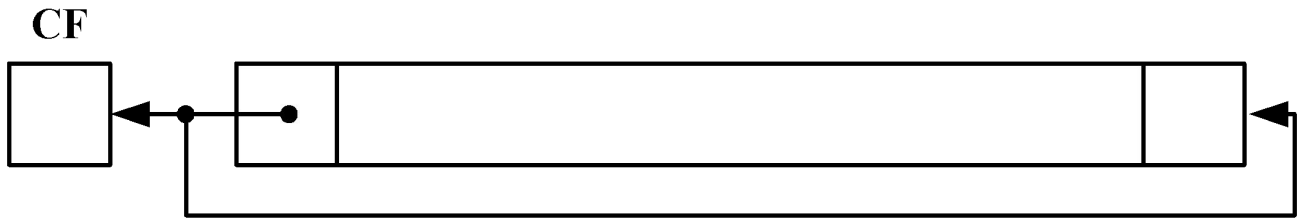


Примеры:
rcr CX, 1
rcr ByteVar, CL

Ассемблер Intel 8086

Команды работы с битами: rol

Общий формат: rol Operand, ShiftCount
Описание: циклический сдвиг влево на заданное количество битов. В качестве операнда может быть слово или байт.
Особенности: при сдвиге выпадающий бит попадает в CF и одновременно передаётся в освободившийся после сдвига правый бит операнда. В качестве операнда ShiftCount обычно выступает регистр CL.

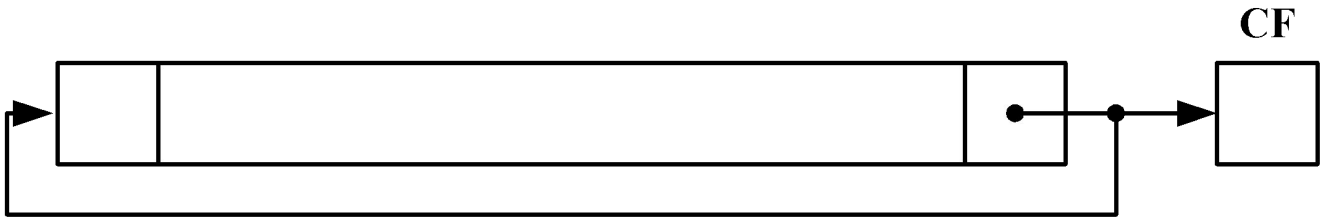


Примеры:
rol DI, 1
rol ByteVar, CL

Ассемблер Intel 8086

Команды работы с битами: ror

Общий формат: ror Operand, ShiftCount
Описание: циклический сдвиг вправо на заданное количество битов. В качестве операнда может быть слово или байт.
Особенности: при сдвиге выпадающий бит попадает в CF и одновременно передаётся в освободившийся после сдвига левый бит операнда. В качестве операнда ShiftCount обычно выступает регистр CL.



Примеры:
ror BH, 1
ror WordVar, CL

Ассемблер Intel 8086

Команды обработки строк

Базовый набор команд, реализующих операции со строками:

CMPS CMPSB CMPSW LODS LODSB LODSW MOVS MOVSB MOVSW SCAS SCASB SCASW STOS STOSB STOSW

Ассемблер Intel 8086

Команды обработки строк: сравнение строк

Общий формат: cmpsb
Описание: сравнение байтов по адресам DS:SI и ES:DI с установкой соответствующих флагов. Фактически производится операция вычитания. После выполнения сравнения индексные регистры изменяют своё значение на единицу:
SI:=SI+1 и DI:=DI+1, если флаг DF=0;
SI:=SI-1 и DI:=DI-1, если флаг DF=1.

Общий формат: cmpsw
Описание: сравнение слов по адресам DS:SI и ES:DI с установкой соответствующих флагов. После выполнения сравнения индексные регистры изменяют своё значение на два:
SI:=SI+2 и DI:=DI+2, если флаг DF=0;
SI:=SI-2 и DI:=DI-2, если флаг DF=1.

Ассемблер Intel 8086

Команды обработки строк: загрузка

Общий формат: lodsb
Описание: загрузка в регистр AL байта, находящегося по адресу DS:SI, после чего индексный регистр изменяет своё значение на единицу:
SI:=SI+1, если флаг DF=0;
SI:=SI-1, если флаг DF=1.

Общий формат: lodsw
Описание: загрузка в регистр AX слова, находящегося по адресу DS:SI, после чего индексный регистр изменяет своё значение:
SI:=SI+2, если флаг DF=0;
SI:=SI-2, если флаг DF=1.

Ассемблер Intel 8086

Команды обработки строк: пересылка строк

Общий формат: movsb
Описание: содержимое байта с адресом DS:SI пересылается в ячейку памяти с адресом ES:DI, после чего индексные регистры изменяют свои значения на единицу:
SI:=SI+1 и DI:=DI+1, если флаг DF=0;
SI:=SI-1 и DI:=DI-1, если флаг DF=1.

Общий формат: movsw
Описание: содержимое слова с адресом DS:SI пересылается в ячейку памяти с адресом ES:DI, после чего индексные регистры изменяют свои значения:
SI:=SI+2 и DI:=DI+2, если флаг DF=0;
SI:=SI-2 и DI:=DI-2, если флаг DF=1.

Ассемблер Intel 8086

Команды обработки строк: сканирование

Общий формат: scasb
Описание: содержимое регистра AL сравнивается с байтом, находящимся по адресу ES:DI, после чего индексный регистр изменяет своё значение на единицу:
DI:=DI+1, если флаг DF=0;
DI:=DI-1, если флаг DF=1.

Общий формат: scasw
Описание: содержимое регистра AX сравнивается со словом, находящимся по адресу ES:DI, после чего индексный регистр изменяет своё значение на два:
DI:=DI+2, если флаг DF=0;
DI:=DI-2, если флаг DF=1.

Особенность: команда предназначена только для установки флагов, операнды своих значений не меняют.

Ассемблер Intel 8086

Команды обработки строк: сохранение

Общий формат: stosb
Описание: содержимое регистра AL записывается в память по адресу ES:DI, после чего индексный регистр изменяет своё значение на единицу:
DI:=DI+1, если флаг DF=0;
DI:=DI-1, если флаг DF=1.

Общий формат: stosw
Описание: содержимое регистра AX записывается в память по адресу ES:DI, после чего индексный регистр изменяет своё значение на два:
DI:=DI+2, если флаг DF=0;
DI:=DI-2, если флаг DF=1.

Ассемблер Intel 8086

Команды обработки строк: команды с параметрами

Общий формат: cmps String1, String2
Описание: сравнение очередных байтов строк String1 и String2, находящихся по адресам DS:SI и ES:DI с установкой соответствующих флагов.
Общий формат: lods String1
Описание: загрузка слова (байта), находящегося по адресу DS:SI, в регистр AX (AL).
Общий формат: movs String1, String2
Описание: пересылка информации из строки String2 в строку String1, которые расположены по адресам DS:SI и ES:DI соответственно.
Общий формат: scas String1
Описание: сравнение слова (байта), находящегося по адресу ES:DI, с содержимым регистра AX (AL).
Общий формат: stos String1
Описание: запись в очередную позицию строки (по адресу ES:DI) содержимого регистра AX или AL.
Особенность: указанные команды позволяют задать параметры, которые несут информацию **только о типе обрабатываемых данных** (слово или байт). Для корректной работы **необходимо** задавать значения соответствующих регистров: DS, ES, SI, DI.

Ассемблер Intel 8086

Команды обработки строк: префиксы команд

Все команды обработки строк предназначены для однократного выполнения задания (т.е. будет обработан один байт или одно слово). Для повторения команд обработки строк можно организовать цикл, но целесообразнее использовать префиксы.

Общий формат: <код префикса> <команда обработки строк>

Возможные префиксы:

1) **REP** – выполнить команду столько раз, сколько указано в регистре CX. Условие прекращения выполнения команды: CX=0;
2) **REPE/REPZ** – повторять выполнение команды, пока равно/пока ноль. Условие прекращения выполнения команды: CX=0 или ZF=0;
3) **REPNE/REPNZ** – повторять выполнение команды, пока не равно/пока не ноль. Условие прекращения выполнения команды: CX=0 или ZF=1.

Ассемблер Intel 8086

Команды изменения состояния процессора

Базовый набор команд для управления состоянием процессора:

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

Ассемблер Intel 8086

Команды изменения состояния процессора: изменение флагов

CLC – очистка флага переноса (CF:=0).
CMC – инверсия флага переноса (CF:=not(CF)).
STC – установка флага переноса (CF:=1).

CLD – очистка флага направления (DF:=0).
STD – установка флага направления (DF:=1).

CLI – очистка флага прерываний (IF:=0, запрет некоторых прерываний).
STI – установка флага прерываний (IF:=1).

Ассемблер Intel 8086

Команды изменения состояния процессора: изменение флагов

CLC – очистка флага переноса (CF:=0).
CMC – инверсия флага переноса (CF:=not(CF)).
STC – установка флага переноса (CF:=1).

CLD – очистка флага направления (DF:=0).
STD – установка флага направления (DF:=1).

CLI – очистка флага прерываний (IF:=0, запрет некоторых прерываний).
STI – установка флага прерываний (IF:=1).

Ассемблер Intel 8086

Команды для работы с прерываниями

Набор команд для реализации работы с прерываниями:

INT **INTO** **IRET**

Ассемблер Intel 8086

Команды для работы с прерываниями: int

Общий формат: int IntNumber

Описание: команда инициирует в процессоре процедуру прерывания, в результате чего управление передаётся на программу обработки прерывания с номером IntNumber.

Алгоритм:

```
PUSHF; //сохранение регистра флагов в стек
TF := 0; //сброс флага трассировки
IF := 0; //сброс флага прерываний – запрет прерываний
CALL FAR (IntNumber*4); //дальний вызов обработчика прерываний
```

Особенность: при выполнении дальнего вызова в регистр IP записывается слово, расположенное по адресу (IntNumber*4), а в регистр CS – слово, расположенное по адресу (IntNumber*4+2).

Ассемблер Intel 8086

Команды для работы с прерываниями: into

Общий формат: into

Описание: команда активизирует прерывание с номером 4, если флаг OF равен 1; если OF = 0, то эта команда не выполняет никаких действий. Если OF = 1, то прерывание выполняется аналогично команде INT 4.

Алгоритм:

```
PUSHF; //сохранение регистра флагов в стек
TF := 0; //сброс флага трассировки
IF := 0; //сброс флага прерываний – запрет прерываний
CALL FAR (4*4); //дальний вызов обработчика прерываний
```

Ассемблер Intel 8086

Команды для работы с прерываниями: iret

Общий формат: iret
Описание: команда передаёт управление из обработчика прерывания в место возникновения прерывания, восстанавливая при этом значения регистров IP, CS и регистра флагов, которые были записаны в стек при вызове обработчика прерывания.
Алгоритм:

```
POP  IP; //восстановление значения регистра IP
POP  CS; //восстановление значения сегментного регистра
POPF; //восстановление регистра флагов на момент прерывания
```

Особенность: при выполнении дальнего вызова в регистр IP записывается слово, расположенное по адресу (IntNumber*4), а в регистр CS – слово, расположенное по адресу (IntNumber*4+2).