Справочник
комманд

ASSEMBLER

Содержание

HLT Останов.

O	D	I	T	S	Z	A	P	C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
F4	HLT	4	5	2	2	Выполняет останов.

Инструкция HLT останавливает выполнение инструкций и переводит процессор x86 в состояние останова. Разрешение прерывания (NMI) или сброс возобновит выполнение. Прерывание (включая NMI) используется для возобновления выполнения после HLT. На инструкцию после HLT указывает значение CS:IP (или CS:EIP для процессора 386).

IDIV Деление со знаком.

O	D	I	T	S	Z	A	P	C
?				?	?	?	?	?

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
F6/7	IDIV r/m8	19/20	19	17/20	101-112/107-118+EA	Деление со знаком AX на байт r/m (AL= частное, AH= остаток).
F7/7	IDIV r/m16	27/28	27	25/28	165-184/171-190+EA	Деление со знаком DX:AX на слово EA (AX= частное, DX= остаток).
F7/7	IDIV r/m32	43/44	43			Деление со знаком пары регистров EDX :EDX на двойное слово (EAX= частное, EDX= остаток).

Операция IDIV выполняет деление со знаком. Делимое, частное и остаток размещаются неявно в фиксированных регистрах. В качестве явного операнда r/m задается только делитель. Тип делителя определяет, какие регистры будут использоваться:

Размер	Частное	Делитель	Остаток	Делимое
байт	AL	r/m	AH	AH
слово	AX	r/m16	DX	DX:AX
двойное слово	EAX	r/m32	EDX	EDX:EAX (только для процессора 386)

Если полученное в результате частное слишком велико и не может разместиться в приемнике, или выполняется деление на 0, то генерируется прерывание 0. Нецелое частное усекается в сторону нуля. Остаток имеет тот же знак, что и делимое, а абсолютное значение остатка всегда меньше, чем абсолютное значение делителя.

IMUL Умножение со знаком.

O	D	I	T	S	Z	A	P	C
*				?	?	?	?	*

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
F6/5	IMUL r/m8	13-18/13-18	19-14/12-17	13/16	80-98/86-104+EA	AX <- AL* байт r/m.
F7/5	IMUL r/m16	13-26/13-26	9-22/12-25	21/24	128-154/134-160+EA	DX:AX <- AL* слово r/m.
F7/5	IMUL r/m32	12-42/13-42	9-38/12-41			EDX:EAX <- EAL * двойное слово r/m.
OF AF/r	IMUL r16, rm/16	13-26/13-26	9-22/12-25			Регистр размером в слово <- регистр размером в слово * слово r/m.
6B /r ib	IMUL r16, rm/16,imm8	13-26/13-26	9-14/12-17	21/24		Регистр размером в слово <- r/m16 * непосредственный байт, расширенный по знаку.
6B /r ib	IMUL r32, rm/32,imm8	13-42	9-14/12-17			Регистр размером в двойное слово <- r/m16 * непосредственный байт, расширенный по знаку.
6B /r ib	IMUL r16, imm8	13-26	9-14/12	21/24		Регистр размером в слово

			-17			<- регистр размером в слово * непосредственный байт, расширенный по знаку.
6B /r ib	IMUL r32, imm8	13-42	9-14/12-17			Регистр размером в двойное слово <- регистр размером в двойное слово * непосредственный байт, расширенный по знаку.
6B /r iw	IMUL r16, r/m16, imm16	13-26/13-26	9-22/12-25	21/24		Регистр размером в слово <- непосредственное слово r/m16.
69 /r id	IMUL r16, r/m16, imm16	13-26/13-26	9-22/12-25	21/24		Регистр размером в двойное слово r/m32 <- * посредственное двойное слово.
69 /r iw	IMUL r16, imm16	13-26/13-26	9-22/12-25			Регистр размером в слово <- r/m16 * не- * посредственное слово.
69 /r id	IMUL r32, imm32	13-42/13-42	9-38/12-41			Регистр размером в двойное слово <- r/m32 * посредственное двойное слово.

Инструкция IMUL выполняет умножение со знаком. Некоторые

формы этой инструкции используют неявные регистровые операнды. Сочетания операндов для всех форм инструкций показаны ниже в столбце "Описание". При следующих условиях инструкция IMUL очищает флаг переполнения и флаг переноса:

Форма инструкции	Условия очистки флагов CF (переноса) и OF (переполнения)
r/m8	AL = расширение по знаку AL до 16 бит
r/m16	AX = расширение по знаку AX до 32 бит
r/m32	EDX:EAX = расширение по знаку EAX до 32 бит
r16,r/m16	Результат умещается равно в 16 бит
r32,r/m32	Результат умещается равно в 32 бита
r16,r/m16,imm16	Результат умещается равно в 16 бит
r16,r/m32,imm32	Результат умещается равно в 32 бита

IN Ввод из порта.

O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
E4 ib	IN AL,imm8	14, pm=8/28, vm=27	12, pm=6/26	5	10	Ввод непосредственного значения из порта (байта) и запись его в регистр AL.
E5 ib	IN AX,imm8	14, pm=8/28, vm=27	12, pm=6/26	5	10	Ввод непосредственного значения из порта (слова) и запись его в регистр AX.
E5 ib	IN AX,imm8	14, pm=8/28, vm=27	12, pm=6/26			Ввод непосредственного значения из порта (двойного слова) и запись его в регистр EAX.

EC	IN AL,DX	14, pm=8/28, vm=27	12, pm=7/27	5	8	Ввод байта из порта DX и запись в регистр AL.
ED	IN AX,DX	14, pm=8/28, vm=27	12, pm=7/27	5	8	Ввод слова из порта DX и запись в регистр AX.
ED	IN EAX,DX	14, pm=8/28, vm=27	12, pm=7/27	5	8	Ввод двойного слова из порта DX и запись в регистр EAX.
8 - если CPL <= **IOPL** 28 - если CPL > IOPL или в виртуальном режиме процессора 8086						

Инструкция IN пересылает байт или слово данных из порта, номер которого указан вторым операндом в регистр (AL, AX или EAX), который задается первым операндом. Доступ к порту от 0 до 65535 можно получить, поместив номер порта в регистр DX и используя инструкцию IN с регистром DX в качестве второго параметра. Эти инструкции ввода-вывода можно сократить, используя в инструкции 8-битового порта ввода-вывода. При использовании 8-битового ввода -вывода из порта старшие 8 бит порта будут равны 0.

INC Увеличение на **1.**

O	D	I	T	S	Z	A	P	C
*				*	*	*	*	*

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
FE /0	INC r/m8	1/3	2/6	2/7	3/15+EA	Увеличивает байт r/m на **1.**
FF /0	INC **r/m16**	1/3	2/6	2/7	3/15+EA	Увеличивает слово **r/m** на **1.**
FF /6	INC r/m32	1/3				Увеличивает двойное слово **r/m** на 1.
40+rw	INC r16	1				Увеличивает регистр размером в слово на **1.**
40+rd	INC r32	1				Увеличивает регистр размером в двойное слово на 1.

Инструкция **INC** прибавляет к операнду **1.** Флаг переноса она не изменяет. Чтобы повлиять на флаг переноса, используйте инструкцию ADD со вторым операндом 1.

INS Увеличение на 1.
INSB
INSW O D I T S Z A P C
INSD

Код операции	Инструкция	Такты			Описание
		486	386	286	
6C	INS r/m8,DX	17, pm=10/32, vm=30	15, pm=9/29	5	Считывает байт из порта DX в ES:(E)DI.
6D	INS r/m16,DX	17, pm=10/32, vm=30	15, pm=9/29	5	Считывает слово из порта DX в ES:(E)DI.
6D	INS r/m32,DX	17, pm=10/32, vm=30	15, pm=9/29	5	Считывает двойное слово из порта DX в ES:(E)DI.
6C	INSB	17, pm=10/32, vm=30	15, pm=9/29	5	Считывает байт из порта DX в ES:(E)DI.
6D	INSw	17, pm=10/32, vm=30	15, pm=9/29	5	Считывает слово из порта DX в ES:(E)DI.
6D	INSD	17, pm=10/32, vm=30	15, pm=9/29	5	Считывает двойное слово из порта DX в ES:(E)DI.
10 и 9, если CPL <= IOPL 32 и 29, если CPL > IOPL или в виртуальном режиме процессора 8086.					

Инструкция INS пересылает данные из порта, номер которого указан в регистре DX, в байт или слово в памяти, индекс которых задается регистром ES:приемник. Операнд в памяти должен быть адресуемым через ES, переопределение сегмента невозможно. Если атрибут размера адреса в инструкции равен 16, то целевым регистром (приемником) является регистр DI, а если атрибут размера адреса равен 32 - то EDI.

В инструкции INS не допускается задавать номер порта в виде непосредственного операнда. К порту нужно адресоваться через значение регистра DX. Перед выполнением инструкции INS загрузите в регистр DX корректное значение.

Адрес приемника определяется содержимым целевого индексного регистра. Перед выполнением инструкции INS загрузите в индексный регистр приемника корректный индекс.

После выполнения пересылки регистры DI или EDI автоматически продвигаются. Если флаг направления равен 0 (была выполнена инструкция CLD), то регистр DI или EDI увеличивается, а если флаг направления равен 1 (была выполнена инструкция STD), то DI или EDI уменьшается. При вводе байта DI увеличивается или уменьшается на 1, при вводе слова - на 2, а при вводе двойного слова - на 4.

Инструкции INSB, INSW и INSD являются синонимами инструкции INS, работающими с байтами, словами и двойными словами соответственно. Инструкции INS может предшествовать префикс REP. При этом выполняется ввод блока из CX байт или слов. Подробности об этой операции можно узнать в описании REP.

INT Вызов процедуры обработки прерывания.
INTO

O	D	I	T	S	Z	A	P	C
		0	0					

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
CC	INT3	26	33	23	52	Прерывание 3 - перехват управления отладчиком.
CC	INT3	44	pm=59	40		Прерывание 3 - защищенный режим.
CC	INT3	77	pm=99	78		Прерывание 3 - защищенный режим.
CC	INT3	82	pm=119			Прерывание 3 - из режима V86 в

						PL0 .
CC	INT3	37+ts	ts	167		Прерывание 3 - защищенный режим.
CD ib	INTimm8	30	37	23	51	Прерывание, номер которого задается посредственным байтом.
CD ib	INTimm8	44	pm=59	40		Прерывание: защищенный режим.
CD ib	INTimm8	77	pm=99	78		Прерывание: защищенный режим.
CD ib	INTimm8	86	pm=119			Прерывание: из режима V86 в PL0.
CD ib	INTimm8	37+ts	ts	167		Прерывание: защищенный режим.
CE	INTO	Pass:28, Fail:3	Fail:3, pm=3 Pass:35	Fail:3, Pass:24	Fail:4, Pass:53	Прерывание 4, если флаг переполнения равен 1.
CE	INTO	46	pm=59	41		Прерывание 4 - защищенный режим.
CE	INTO	73	pm=99	79		Прерывание 4 - защищенный режим.
CE	INTO	84	pm=119			Прерывание 4 - из режима V86 в PL0.

CE	INTO	39+ts	ts	168		Прерывание 4 - защищенный режим.

* - добавляется 1 цикл для каждого байта следующей выполняемой инструкции.

Инструкция INT n генерирует через программное обеспечение обращение к обработчику прерываний. Непосредственный операнд (значение от 0 до 255) задает номер индекса в таблице дескрипторов прерываний (IDT) вызываемой обработки прерываний. В защищенном режиме IDT содержит массив восьмибайтовых дескрипторов. Дескриптор вызываемого прерывания должен указывать прерывание, ловушку или вентиль задачи. В реальном режиме адресации IDT представляет собой массив из четырех указателей размером в байт. В защищенном и реальном режиме адресации базовый линейный адрес IDT определяется содержимым IDTR.

Инструкции прерывания INT n идентична условной программной инструкции INTO, но номер прерывания неявно равен 4, а прерывание выполняется, если флаг переполнения процессоров 86, 286 или 386.

Первые 32 прерывания зарезервированы фирмой Intel для системных целей. Некоторые из этих прерываний используются для внутренних генерируемых исключительных ситуаций.

Инструкция INT n в общем случае ведет себя как вызов дальнего типа, но регистр флагов заносится в стек перед адресом возврата. Процедуры обработки прерывания возвращают управление через инструкцию IRET, которая извлекает из стека флаги и адрес возврата.

В реальном режиме адресации инструкция INT n заносит в стек флаги, регистр CS и адрес возврата (IP) в указанном порядке и переходит к длинному указателю, индекс которого указан номером прерывания.

INVD Запрещает использование кэш-памяти (только для процессора i486).

O D I T S Z A P C

Код операции	Инструкция	Такты	Описание
		486	
0F 08	INVD	4	Запрещает использование всей кэш-памяти.

Внутренний кэш-буфер сбрасывается, и выполняется специальная функция цикла шины, которая указывает, что внешний кэш-буфер также должен сбрасываться. Данные, содержащиеся в записываемых внешних кэш-буферах, отбрасываются.

Примечание: Данная инструкция зависит от конкретной реализации. В будущих процессорах фирмы Intel она может реализовываться по-разному.

Ответственность за возврат информации об очистке внешней кэш -памяти ложится на аппаратное обеспечение.

INVLPG Запрещает запись TBL (только для процессора i486).

O	D	I	T	S	Z	A	P	C

Код операции	Инструкция	Такты	Описание
		486	
0F 01/7	INVLPG m	12	Запрещает запись таблицы TBL.

Инструкция INVLPG используется для запрещения отдельной записи в таблице TBL. Для записи таблицы используется кэш-память. Если таблица TBL содержит допустимую запись, которая отображает адрес операнда в памяти, то запись TBL отмечается, как недопустимая.

И в защищенном режиме, и в виртуальном режиме процессора 8086 при использовании регистрового операнда генерируется недопустимый код операции.

Примечание: Данная инструкция зависит от конкретной реализации. В будущих процессорах фирмы Intel она может реализовываться по-разному.

IRET Возврат из прерывания (только для процессоров 386, i486)
IRETD

O	D	I	T	S	Z	A	P	C
*	*	*	*	*	*	*	*	*

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
CF	IRET	15	22, pm=38	17 pm=31	32	Возврат из прерывания (возврат дальнего типа и извлечение флагов).
CF	IRET	ts+32	pm=82	55		Возврат из прерывания.
CF	IRETD	ts	22, pm=38			Возврат из прерывания (возврат дальнего типа и извлечение флагов).

CF	IRETD	36	pm=82			Возврат из прерывания с меньшими привилегиями.
CF	IRETD	15	pm=60			Возврат из прерывания в режиме V86.
CF	IRETD	ts+32	ts			Возврат из прерывания.

В реальном режиме адресации инструкция IRET извлекает из стека указатель инструкций, регистр CS и регистр флагов и возобновляет выполнение прерванной подпрограммы.

В защищенном режиме действие инструкции IRET зависит от установки флага вложенной задачи (бита NT) в регистре флагов. При извлечении нового образа флага из стека бит IOPL в регистре изменяется только тогда, когда CPL = 0.

Если NT равно 0, то инструкция IRET возвращает управление из процедуры обработки прерывания без переключения задачи. Программа, куда возвращается управление, должна иметь меньшие или равные привилегии, чем процедура обработки прерывания (это показывают биты RPL селектора CS, извлеченного из стека). Если целевая программа имеет меньшие привилегии, то инструкция IRET извлекает также из стека указатель стека и регистр SS.

Если NT не равно 0, то инструкция IRET резервирует операцию CALL или INT, которая вызывает переключение задач. Измененное состояние задачи, выполняющей инструкцию IRET, сохраняется в ее сегменте состояния задачи. Если позднее вновь последует вхождение в задачу, то выполняется код, следующий за инструкцией IRET.

JCC Переход, если удовлетворяется условие.

O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
77 cb	JA rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если выше (CF = 0 и ZF = 0.
73 cb	JAE rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если выше или равно (CF = 0).
72 cb	JB rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если ниже (CF = 1).
76 cb	JBE rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если ниже или равно (CF = 1, или ZF = 1).
72 cb	JC rel8	3/1	7+m,3	7,3	16,4	Короткий переход при переносе (если CF = 1). = 1.
E3 cb	JCXZ rel8	3/1	9+m,5	8,4	18,3	Короткий переход, если регистр CX равен 0.
E3 cb	JCXZ rel8	3/1	9+m,5			Короткий переход, если регистр ECX равен 0.
74 cb	JT rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если равно (ZF = 0).
7F cb	JG rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если больше (ZF = 0 и SF = OF).

7D cb	JGE rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если больше или равно (SF = 0).
7C cb	JT rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если меньше (SF <> OF).
7E cb	JLE rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если меньше или равно (ZF = 1 и SF <> OF).
7E cb	JLE rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если не выше (CF = 1 или ZF = 1).
72 cb	JNAE rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если не выше (CF = 1).
73 cb	JNB rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если не ниже (CF = 0).
77 cb	JNBE rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если не ниже или равно CF = 0 и ZF = 0).
73 cb	JNC rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если нет переноса (CF = 0).
75 cb	JNE rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если не равно (ZF = 0).
7E cb	JNG rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если не больше (ZF = 1 или SF <> OF).

7C cb	JNGT rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если не больше или равно (SF <> OF).
7D cb	JNL rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если не меньше (SF <> OF).
7F cb	JNLE rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если не меньше или равно (SF = 0 и SF = OF).
71 cb	JNO rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если нет переполнения (OF = 0).
7B cb	JNH rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если нет четности (PF = 0).
79 cb	JNS rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если SF = 0.
75 cb	JNZ rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если ZF = 0.
70 cb	JO rel8	3/1	7+m,3	7,3	16,4	Короткий переход, если переполнение (OF = 0).
7A cb	JP rel8	3/1	7+m,3	7,3	16,4	Короткий переход по, паритету (если PF = 1).
7A cb	JPE rel8	3/1	7+m,3	7,3	16,4	Короткий переход, по четности (PF = = 1).
71 cb	JPO rel8	3/1	7+m,3	7,3	16,4	Короткий переход по нечетности (PF = 0).

78 cb	JS rel8	3/1	7+m,3	7,3	16,4	Короткий переход по знаку (SF = 0).
74 cb	JZ rel8	3/1	7+m,3	7,3	16,4	Короткий переход по нулю (ZF = 1).

Продолжение

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
OF 87 cw/cd	JA rel16/32	3/1	7+m,3			Короткий переход, если выше (CF = 0 и ZF = 1).
OF 83 cw/cd	JAE rel16/32	3/1	7+m,3			Короткий переход, если ниже (CF = 0).
OF 82 cw/cd	JB rel16/32	3/1	7+m,3			Короткий переход, если ниже (CF = 1).
OF 86 cw/cd	JBE rel16/32	3/1	7+m,3			Короткий переход, если ниже или равно (CF = 1 или ZF = 1).
OF 82 cw/cd	JC rel16/32	3/1	7+m,3			Короткий переход, если перенос (CF = 1).
OF 84 cw/cd	JE rel16/32	3/1	7+m,3			Короткий переход, если равно (ZF = 1).
OF 84 cw/cd	JZ rel16/32	3/1	7+m,3			Короткий переход, если 0 (ZF = 1).
OF 8F cw/cd	JG rel16/32	3/1	7+m,3			Короткий переход, если больше (ZF = 0 и SF = 0).
OF 8D cw/cd	JGE rel16/32	3/1	7+m,3			Короткий переход, если больше или равно (SF = OF).
OF 8C cw/cd	JL rel16/32	3/1	7+m,3			Короткий переход, если меньше (SF <> OF).

OF 8E cw/cd	JLE rel16/32	3/1	7+m,3			Короткий переход, если меньше или равно (ZF = 1 и SF <> OF).
OF 86 cw/cd	JNA rel16/32	3/1	7+m,3			Короткий переход, если не выше (CF = 1 и ZF = 1).
OF 82 cw/cd	JNAE rel16/32	3/1	7+m,3			Короткий переход, если не выше или равно (CF = 1).
OF 8F3cw/cd	JNB rel16/32	3/1	7+m,3			Короткий переход, если не ниже (CF = = 0).
OF 87 cw/cd	JNBE rel16/3	3/	7+m,			Короткий переход, если не ниже или равно (CF = 0 и ZF = 0).
OF 83 cw/cd	JNC rel16/32	3/1	7+m,3			Короткий переход, если нет переноса и ZF = 0).
OF 85 cw/cd	JNE rel16/32	3/1	7+m,3			Короткий переход, если не равно (ZF = 0).
OF 8E cw/cd	JNG rel16/32	3/1	7+m,3			Короткий переход, если не больше (ZF = 1 или SF <> OF).
OF 8C cw/cd	JNGE rel16/32	3/1	7+m,3			Короткий переход, если не больше или равно (SF <> OF).
OF 8D cw/cd	JNL rel16/32	3/1	7+m,3			Короткий переход, если не меньше (SF = OF).

OF 8F cw/cd	JNLE rel16/32	3/1	7+m,3			Короткий переход, если не меньше или равно (ZF = 0 и SF = OF).
OF 81 cw/cd	JNO rel16/32	3/1	7+m,3			Короткий переход, если не переполнение (OF = 0).
OF 8B cw/cd	JNP rel16/32	3/1	7+m,3			Короткий переход, если PF = 0.
OF 89 cw/cd	JNS rel16/32	3/1	7+m,3			Короткий переход, если SF = 0.
OF 85 cw/cd	JNZ rel16/32	3/1	7+m,3			Короткий переход, если не ноль (ZF = 0).
OF 80 cw/cd	JO rel16/32	3/1	7+m,3			Короткий переход, если переполнение OF = 0).
OF 8A cw/cd	JP rel16/32	3/1	7+m,3			Короткий переход, если PF = 1.
OF 8A cw/cd	JPE rel16/32	3/1	7+m,3			Короткий переход по четности (PF = 1).
OF 8B cw/cd	JPO rel16/32	3/1	7+m,3			Короткий переход по нечетности (PF = 1).
OF 88 cw/cd	JS rel16/32	3/1	7+m,3			Короткий переход по знаку (SF = 1).
OF 84 cw/cd	JZ rel16/32	3/1	7+m,3			Короткий переход, если нуль (ZF = 1).

* - при переходе добавьте один такт для каждого байта следующей выполняемой инструкции (только для процессора 80286).

Примечание: Первое значение тактов показано для истинного условие (осуществление перехода), второе - для ложного (переход не выполняется). Rel16/32 показывает, что имеется две форму этой инструкции. Одна из них использует 16-битовое смещение, другая - 32-битовое относительное смещение, в зависимости от атрибута размера операнда в операции.

Условные переход (кроме JCXZ/JECXZ) проверяет флаги, которые устанавливаются предыдущей инструкцией. Во всех приведенных выше описаниях в скобках показано условие для каждой мнемоники. Термины "больше" или "меньше" используются для сравнения целых чисел со знаком, а термины "выше" или "ниже" - для беззнаковых целых значений.

Если заданное условие истинно, то осуществляется переход по адресу, задаваемому операндом. Кодировка инструкции более эффективна, когда цель условного перехода находится в текущем сегменте кода и в границах от -128 до +127 байт первого байта следующей инструкции. Переход можно также выполнять по адресу от -32768 до +32767 (атрибут размера сегмента 16) или от -2 в 31 степени до 2 в 31 степени - 1 (атрибут размера сегмента 32) относительно первого байта следующей инструкции. Когда адрес условного перехода находится в другом сегменте, используйте противоположный вариант инструкции перехода (то есть JE и JNE), а затем перейдите на целевой адрес в другом сегменте с помощью дальнего безусловного перехода. Например, вы не можете записать:

```
JZ FARLABEL;
```

Вместо этого следует записать:

```
    JNZ BEYOND;
    JMP FARLABEL;
BEYOND:
```

Поскольку конкретное состояние флагов можно интерпретировать несколькими способами, для большинства кодов операций условных переходов TASM предусматривает более одной мнемоники. Например, если вы сравниваете два символа в регистре AX и ходите выполнить переход в случае их равенства, используйте инструкцию JZ, или вы выполните над регистром AX и битовой маской операцию AND и хотите перейти только в том случае, если результат равен 0, используйте инструкцию JZ, которая представляет собой синоним JE.

Инструкции JCXZ/JECXZ отличаются от других инструкций условного перехода, поскольку они проверяют не регистр флагов, а содержимое регистра CX или ECX на 0. Инструкции JCXZ/JECXZ полезно использовать в начале условного цикла, который завершается с помощью инструкции условного цикла (такой как LOOP TARGET LABEL). Инструкции JCXZ/JECXZ предотвращают вхождение в цикл при нулевом содержимом CX или ECX. В противном случае потребовалось бы выполнять цикл не 0 раз, а 32 раза или 64K.

JMP Переход.

O	D	I	T	S	Z	A	P	C

Флаги: все при переключении задачи, ни одного, если нет переключения задачи.

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
EB cb	JMP rel8	3	7+m	7	15	Короткий переход.
E9 cw	JMP rel16	3	7+m	7	15	Ближний переход.
FF /4	JMP r/m16	5/5	+m/ 0+m	7/11	11/18 +AE	Косвенный ближний переход.
EA cd	JMP ptr16:16	17pm =19	2+m, m=27+m	11, pm=23	15	Межсегментный переход, 4-байтовый непосредственный адрес.
EA cd	JMP ptr16:16	32	pm 5+m	38		Переход для вызова вентиля, те же привилегии.
EA cd	JMP ptr16:16	42+ ts	ts	175		Переход через сегмент состояния задачи.
EA cd	JMP ptr16:16	42+ ts	ts	180	24+EA	Переход через вентиль задачи.
FF /5	JMP m16:16	13, pm= 18	3+m, m=31+m	15, pm=26		Косвенный переход r/m16:16 (межсегментный).
FF /5	JMP m16:16	31	pm=49+ m	41		Переход для вызова вентиля, те же привилегии.

FF /5	JMP m16:16	41+ ts	5+ts	178		Переход через сегмент состояния задачи.
FF /5	JMP m16:16	42+ ts	5+ts	183		Переход через вентиль задачи.
E9 cd	JMP rel32	3	7+m			Ближний переход.
FF /4	JMP r/m32	5/5	7+m,10	m		Ближний переход.
EA cp	JMP ptr16:32	13pm =18	2+m, pm=27+ m			Межсегментный переход, 6-байтовый непосредственный адрес.
EA cp	JMP ptr16:32	31	pm=45+ m			Переход для вызова вентиля, те же привилегии.
EA cp	JMP ptr16:32	42+ ts	ts			Переход через сегмент состояния задачи.
FF /5	JMP ptr16:32	43+ ts	ts			Переход через вентиль задачи.
FF /5	JMP m16:32	13, pm= 18	pm=43+ m pm=27+ m			Межсегментный переход по адресу в двойном слове r/m.
FF /5	JMP m16:32	31	pm=49+ m			Переход для вызова вентиля, те же привилегии.
FF /5	JMP m16:32	41+ ts	5+ts			Переход через сегмент состояния задачи.
FF /5	JMP m16:32	42+ ts	5+ts			Переход через вентиль задачи.

Инструкция JMP передает управление в другую точку в потоке инструкций без записи информации о возврате.

Действие различных форм инструкции показано ниже.

Переходы, где цель задана операндами r/m16, r/m32, rel16 и rel32, представляют собой ближние переходы и не связаны с изменением значения сегментного регистра.

Инструкции JMP rel16 и JMP rel32 для определения целевого адреса прибавляют смещение к адресу инструкции, следующему за инструкцией JMP. Форма rel16 используется, когда атрибут размера операнда равен 16 (только при атрибуте размера сегмента 16), а форма rel32 - когда атрибут размера операнда равен 32 битам (только при атрибуте размера сегмента 32). Результат записывается в 32-битовый регистр EIP. При использовании формы rel16 старшие 16 бит регистра EIP очищаются, что дает в результате смещение, значение которого не превышает 16 бит.

Инструкции JMP r/m16 и JMP r/m32 задают регистр или адрес в памяти, из которого выбирается абсолютное смещение относительно процедуры. Смещение, выбираемое из r/m, имеет размер 32 бита при атрибуте размера операнда 32 (r/m32) или 16 бит при атрибуте размера операнда 16 (r/m16).

Формы инструкции JMP ptr16:16 и ptr16:32в качестве длинного указателя целевого адреса используют 4-байтовый или 6-байтовый операнд. Инструкции JMP ptr16:16 или ptr16:32 извлекают из указанного адреса памяти длинный указатель (косвенность). В режиме реальной или виртуальной адресации процессора 8086 длинный указатель предусматривает 16 бит для регистра CS и 16 или 32 бита для регистра EIP (в зависимости от атрибута размера операнда). В защищенном режиме обе формы с длинным указателем анализируют полномочия доступа (байт AR) в дескрипторе, индекс которого указывается селектором длинного указателя. В зависимости от значения байта AR переход будет выполнять один из следующих типов передачи управления:

- переход на сегмент кода с тем же уровнем привилегий;

- переключение задачи.

LAHF Загрузка флагов в регистр AH.

O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
9H	LAHF	3	2	2	4	Загружает в AH флаги: SF, ZF, xx, AF, xx, PF, xx, CF.

Инструкция LAHF пересылает младший байт регистра флагов (размером в слово) в AH. При этом пересылаются следующие биты (от старшего к младшему): флаг знака, нуля, неопределенный бит, дополнительного переноса, неопределенный бит, флаг четности, неопределенный бит и флаг переноса.

LAR Загрузка байта полномочий доступа (только для защищенного режима процессором 80286/386/486).

O D I T S Z A P C

Код операции	Инструкция	Такты			Описание
		486	386	286	
OF 02/r	LAR r16,r/m16	11/11	pm=15/16	14/16	r16 <-r/m16 с маской FF00.
OF 02/r	LAR r32,r/m32	11/11	pm=15/16		r32 <-r/m32 с маской 00FxFF00.

Инструкция LAR отмеченную форму второго двойного слова дескриптора для исходного селектора, если селектор, видимый в CPL (модифицирован RPL селектора) и представляет собой допустимый тип дескриптора. Целевой регистр загружается старшим двойным словом дескриптора, отмеченным 00FxFF00, а ZF устанавливается в 1 (x указывает, что четыре бита, соответствующие старшим четырем битам границы не определены в загружаемом инструкцией LAR значении). Если селектор не является видимым или имеет недопустимый тип, что флаг ZF очищается.

Если задан 32-битовый размер операнда, то в 32-битовый целе-

вой регистр загружается все 32-битовое значение. Если задан 16-битовый размер операнда, то младшие 16 бит этого значения записываются в 16-битовый целевой регистр.

Для LAR допустим есть код и дескрипторы сегмента данных. (Допустимые типы сегмента и дескриптора вентиля описываются в руководстве фирмы Intel.)

LEA Загрузка действующего смещения адреса.

O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
8D/r	LEA r16,m	1	2	3	2+EA	Записывает действующий адрес m в регистре r16.
8D/r	LEA r32,m	1	2			Записывает действующий адрес m в регистре r32.
8D/r	LEA r16,m	1	2			Записывает действующий адрес m в регистре r16.
8D/r	LEA r32,m	1	2			Записывает действующий адрес m в регистре r32.

Инструкция LEA вычисляет действующий адрес (смещение) и записывает его в заданном регистре. Атрибут размера операнда инструкции определяется выбранным регистром. Атрибут размера адреса определяется атрибутом USE сегмента, содержащего второй операнд. Атрибуты размера операнда и размера адреса на действие, выполняемое инструкцией LEA, влияют следующим образом:

Размер операнда	Размер адреса	Выполняемое действие
16	16	Вычисляется и записывается в в требуемом 16-битовом целевом регистре 16-битовый дей-

		ствующий адрес.
16	32	Вычисляется и записывается в в требуемом 16-битовом целевом регистре 32-битовый действующий адрес.
32	16	Вычисляется и записывается в в требуемом 32-битовом целевом регистре 16-битовый действующий адрес.
32	32	Вычисляется и записывается в в требуемом 32-битовом целевом регистре 32-битовый действующий адрес.

LEAVE Выход из процедуры высокого уровня
(только для процессоров 80186/286/386/486).

O D I T S Z A P C

Код операции	Инструкция	Такты			Описание
		486	386	286	
C9/r	LEAVE	5	4	5	Устанавливает SP в BP.
C9/r	LEAVE	5	4		Устанавливает ESP в EBP.

Инструкция LEAVE выполняет действие, обратное по отношению к инструкции ENTER. Путем копирования указателя границы стека в указатель стека инструкция LEAVE освобождает пространство стека, используемое процедурой для своих локальный переменных. Старый указатель стека извлекается и записывается в BP или EBP, восстанавливая границы стека вызывающей программы. Последующая инструкция RET nn удаляет все аргументы, занесенные в стек процедурой, из которой выполняется выход.

LGDT/ LIDT Загружает регистр глобальных дескрипторов/прерываний
(только защищенный режим процессоров 80286/386/486).

O D I T S Z A P C

Код операции	Инструкция	Такты			Описание
		486	386	286	
OF 01/2	LGDT m16&32	11	11	11	Загружает m в регистр таблицы глобальных дескрипторов.
OF 01/2	LIDT m16&32	11	11	12	Загружает m в регистр таблицы дескрипторов прерываний.

Инструкции LIDT и LGTD загружают линейный базовый адрес и граничное значение из шестибайтового операнда в памяти в регистры IDTR и GDTR соответственно. Если в LGTD или LIDT используется 16-

битовый операнд, то в регистр загружается 16-битовое граничное значение и 24-битовая база, а старшие 8 бит 6-байтового операнда не используются. Если используется 32-битовый операнд, то загружаются 16-битовое граничное значение и 32-битовая база, а старшие 8 бит 6-байтового операнда используются в качестве старших бит базового адреса.

Инструкции SGDT и SIDT всегда записывают все 48 бит данных 6 -байтового операнда. В процессоре 80286 старшие 8 бит после выполнения инструкции SGDT или SIDT не определены. В процессоре 80386 как для 16, так и для 32-битового операнда старшие 8 бит записываются в старшие 8 бит адреса. Если инструкции LGDT или LIDT используются с 16-битовым операндом для загрузки регистра, сохраненного по инструкции SGDT или SIDT, то старшие 8 бит записываются как нули.

Инструкции LGDT и LIDT используются в системном программном обеспечении и не применяются в прикладных программах. Это единственные инструкции, которые в защищенном режиме процессора 80386 непосредственно загружают линейный адрес (то есть не адрес относительно сегмента).

LGS Загрузка полного указателя (LGS/LSS/LFS только для
LSS процессоров 386 и i486).
LFS
LDS O D I T S Z A P C
LES

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
C5 /r	LDS r16,m16:16	6/12	7, pm=22	7, pm=21	16+EA	Загружает DS:r16 указателем из памяти.
C5 /r	LDS r32,m16:32	6/12	7, pm=22			Загружает DS:r32 указателем из памяти.
OF B2/r	LSS r16,m16:32	6/12	7, pm=22			Загружает SS:r16 указателем из памяти.
OF B2/r	LSS r32,m16:32	6/12	7, pm=22			Загружает ES:r16 указателем из памяти.

C4 /r	LES r16,m16:16	6/12	7, pm=22	7, pm=2	16 +EA	Загружает ES:r16 указателем из памяти.
C4 /r	LES r32,m16:32	6/12	7, pm=22			Загружает ES:r32 указателем из памяти.
OF B2/r	LFS r16,m16:16	6/12	7, pm=22			Загружает FS:r16 указателем из памяти.
OF B2/r	LSS r32,m16:32	6/12	7, pm=22			Загружает FS:r32 указателем из памяти.
OF B2/r	LGS r16,m16:16	6/12	7, pm=22			Загружает GS:r16 указателем из памяти.
OF B2/r	LSS r32,m16:32	6/12	7, pm=22			Загружает GS:r32 указателем из памяти.

Данные инструкции считывают из памяти полный указатель и записывают его в выбранном сегментном регистре (паре регистров). 16 бит полного указателя загружаются в сегментный регистр SS, DS, ES, FS или GS. В другой регистр загружаются 32 бита, если атрибут размера операнда равен 32, и 16 бит, если атрибут размера операнда равен 16. Другой 16 или 32-битовый регистр, который нужно загрузить, определяется тем, задан ли регистровый операнд r16 или r32.

При выполнении присваивания одному из сегментных регистров дескриптор также загружается в сегментный регистр. Данные для регистра получаются из записи таблицы дескриптора для данного селектора.

Нулевой селектор (значения 0000 - 0003) могут загружаться в регистры DS, ES, FS или GS не вызывая исключительных ситуаций по нарушению защиты. (Все последующие ссылки на сегмент, соответствующий сегментный регистр которого загружается нулевым селектором, вызывают исключительную ситуацию #GP(0). Ссылок на память в этом сегменте не происходит.)

LLDT Загрузка регистра таблицы локальных дескрипторов (только для защищенного режима процессоров 80286/386/486).

O D I T S Z A P C

Код операции	Инструкция	Такты			Описание
		486	386	286	
OF 00 /2	LLDT r/m16	11/11	20	17/19	Загружает в LDTR селектор r/m16.

Инструкция LLDT загружает регистр таблицы локальных дескрипторов (LDTR). Операнд инструкции LLDT размером в слово (память или регистр) должен содержать селектор таблицы локальных дескрипторов. Если это так, то регистр загружается из записи. Дескрипторные регистры DS, ES, SS, FS, GS и CS не затрагиваются. Поле LDT в сегменте состояния задачи не изменяется.

Операнд-селектор может быть нулевым. Если это так, то LDTR отмечается как недопустимая. Все ссылки на дескрипторы (кроме инструкций LAR, VERR, VERW или LSL) приводят с сбою #GP.

Инструкция LLDT используется в системном программном обеспечении, в прикладных программах она не применяется.

LMSW Загрузка слова состояния машины (только для защищенного режима процессоров 80286/386/486).

O D I T S Z A P C

Код операции	Инструкция	Такты			Описание
		486	386	286	
OF 01 /6	LMSW r/m16	13/13	10/13	3/6	Загружает r/m16 в слово состояния машины.

Инструкция LMSW загружает слово состояния машины (часть CR0) из операнда-источника. Данную инструкцию можно использовать для переключения в защищенный режим. В этом случае за ней должен следовать межсегментный переход для очистки очереди инструкций. Инструкция LMSW не будет выполнять обратного переключения в режим реальной адресации.

Инструкция LMSW используется только в системном программном обеспечении, в прикладных программах она не применяется.

LOCK Выдает сигнальный префикс **LOCK#**.

O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
F0	LOCK	1	0	0	2	Добавляет для следующей инструкции сигнал **LOCK#**.

Префикс **LOCK** (блокировка) вызывает при выполнении инструкции, которая за ним следует, сигнал LOCK#. В многопроцессорном окружении данный сигнал можно использовать, чтобы обеспечить ситуацию, когда процессор после выдачи LOCK# получает исключительные права на использование разделяемой памяти. В процессоре 386 для реализации проверки и установки используется обычно инструкция BTS, которая реализует обычно применяемую для этого последовательность "чтение-модификация-запись".

В процессорах 386 и i486 префикс LOCK функционирует только со следующими инструкциями:

BT, BTS, BTR, BTC	mem, reg/imm
XCHG	reg, mem
XCHG	mem, reg
ADD, OR, ADC, SBB	mem, reg/imm
NOT, NEG, INC, DEC	mem

Если префикс **LOCK** используется с другими инструкциями, отличными от перечисленных выше, то будет генерироваться перехват неопределенного кода операции.

Инструкция XCHG всегда добавляет LOCK#, независимо от наличия или отсутствия префикса LOCK.

Целостность LOCK не нарушается при выравнивании поля в памяти. Блокировка памяти соблюдается для произвольных невыровненных полей.

Блокировка доступа не обеспечивается, если другой процессор параллельно выполняет инструкцию, которая имеет одну из следующих характеристик:

- перед ней не указан префикс LOCK;

- она не является инструкцией из приведенного выше списка;

- заданный операнд в памяти не перекрывает в точности операнд-приемник (блокировка не обеспечивается для частичного перекрытия, даже если один операнд в памяти целиком содержится внутри другого).

LODS Загружает строковый операнд.
LODSB (Инструкция LODSD работает только на процессорах 386 и
LODSW i486.)
LODSD

O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
AC	LODS m18	5	5	5	2	Загружает в AL байт [(E)SI].
AD	LODS m16	5	5	5	12	Загружает в AX слово [(E)SI].
AD	LODS m32	5	5	5	12	Загружает в EAX двойное слово [(E)SI].
AC	LODSB	5	5	5	2	Загружает в AL байт DS:[(E)SI].
AD	LODSW	5	5	5	12	Загружает в AX слово DS:[(E)SI].
AD	LODSD	5	5	5	12	Загружает в EAX двойное слово DS:[(E)SI].

Инструкция LODS загружает регистр AL, AX или EAX байтом, словом или двойным словом из памяти по адресу, который указывает исходный индексный регистр. После выполнения пересылки исходный индексный регистр автоматически продвигается (увеличивается или уменьшается). Если флаг направления равен 0 (была выполнена инструкция CLD) индекс источника увеличивается, если флаг направления равен 1 (была выполнена инструкция STD), то он уменьшается. При загрузке байта индекс увеличивается или уменьшается на 1, при загрузке слова - на 2, а при загрузке двойного слова - на 4.

Если атрибут размера адреса для инструкции равен 16, то в качестве индексного регистра источника используется регистр SI. В противном случае атрибут размера адреса равен 32 битам, и используется регистр ESI. Адрес исходных данный определяется исключительно содержимым ESI/SI. Перед выполнением инструкции LODS загрузите в регистр SI корректное значение индекса. Синонимами

для инструкции LODS при работе с байтом, словом и двойным словом являются инструкции LODS, LODSW и LODSD.

Инструкции LODS может предшествовать префикс REP. Однако более часто LODS используется в конструкции LOOP, поскольку обычно требуется дальнейшая обработка данных, загруженных в регистр EAX, AX или AL.

LOOP Управление циклом с помощью счетчика CX.
LOOPусл Управление циклом с помощью счетчика CX/ECX (только для процессоров 386 и i486).

O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
E2 cb	LOOP rel8	2,6	11+m	8, noj=4	17, noj=5	Уменьшить счетчик, перейти (короткий переход), если счетчик равен 0.
E1 cb	LOOPE rel8	9,6	11+m	8, noj=4	17, noj=6	Уменьшить счетчик, перейти (короткий переход), если счетчик равен 0 и ZF=1.
E1 cb	LOOPZ rel8	9,6	11+m	8, noj=4	17, noj=6	Уменьшить счетчик, перейти (короткий переход), если счетчик равен 0 и ZF=1.
E0 cb	LOOPNE rel8	9,6	11+m	8, noj=4	19, noj=5	Уменьшить счетчик, перейти (короткий переход), если счетчик равен 0 и ZF=0.
E0 cb	LOOPNZ rel8	9,6	11+m	8, noj=4	19, noj=5	Уменьшить счетчик, перейти (короткий переход), если счетчик равен 0 и ZF=0.

Инструкция LOOP уменьшает регистр-счетчик без изменения ка-

ких-либо флагов. Затем проверяются условия в соответствии с формой используемой инструкции LOOP. Если эти условия удовлетворяются, то выполняется короткий переход на метку, заданную операндом LOOP. Если атрибут размера адреса равен 16, то в качестве регистра-счетчика используется регистр CX, в противном случае используется регистр ECX (только для процессора 386). Операнд инструкции LOOP должен лежать в диапазоне от 128 (десятичных) байт до инструкции до 127 байт после инструкции.

Инструкции LOOP обеспечивают итеративный контроль и комбинируют управление с помощью индекса цикла с переходом по условию. Используйте инструкцию LOOP, загружая беззнаковое значение счетчика итерации, а затем кодируя LOOP в конце последовательности повторяемых инструкций. Целевым адресом LOOP является метка, которая указывает на начало итерации.

LSL Загрузка границы сегмента (только для защищенного режима процессоров 80286/386 и i486).

O	D	I	T	S	Z	A	P	C
					*			

Код операции	Инструкция	Такты			Описание
		486	386	286	
OF 03 /r	LSL r16,r/m16	10/10	pm=20/21	14/16	Загрузка: r16<-граница сегмента, селектор r/m16 (с точностью до байта).
OF 03 /r	LSL r32,r/m32	10/10	pm=20/21		Загрузка: r32<-граница сегмента, граница сегмента селектор r/m32 (с точностью до байта).
OF 03 /r	LSL r16,r/m16	10/10	pm=25/26	14/16	Загрузка: r16<-граница сегмента, граница сегмента, селектор r/m16 (с точностью до страницы).
OF 03 /r	LSL r32,r/m32	10/10	pm=26/26		Загрузка: r32<-граница сегмента, селектор r/m32 (с точностью до страницы).

Инструкция LSL загружает в регистр границу сегмента и устанавливает регистр ZF в значение 1, обеспечивая доступность исходного селектора в CPL (RPL), и что тип дескриптора воспринимается LSL. В противном случае ZF сбрасывается в 0, а целевой регистр не изменяется. Граница сегмента загружается, как значение с точностью до байта. Если дескриптор имеет границу сегмента с точностью до страницы, то перед загрузкой в целевой регистр LSL будет транслировать его в байтовую границу (сдвиг влево на 12 20-битовой "необработанной" границы от дескриптора, затем операция OR с 000000FFFH).

32-битовая форма этой инструкции записывает в 16-битовом целевом регистре границу с точностью до 32 бит.

Допустимыми для LSL являются сегмент кода и сегмент данных.

LTR Загрузка регистра задачи
(только для защищенного режима процессоров 80286/386/486)

O D I T S Z A P C

Код операции	Инструкция	Такты			Описание
		486	386	286	
OF 03 /3	LTR r/m156	20/20	pm=23/27	14/16	Загрузка слова EA в регистр задачи.

Инструкция LTR загружает регистр задачи из исходного регистра или ячейки памяти, заданной операндом. Сегмент состояния загруженной задачи отмечается, как занятый. Переключения задач не происходит.

Инструкция LTR используется только в системном программном обеспечении, в прикладных программах она не применяется.

MOV Перемещение данных.

O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
88 /r	MOV r/m8,r8	1	2/2	2/3	2/9+EA	Перемещает байтовый регистр в байт r/m.
89 /r	MOV r/m16,r16	1	2/2	2/3	2/9+EA	Перемещает регистр размером в слово в байт r/m.
8A /r	MOV r/m8,r8	1	2/2	/5	2/8+EA	Перемещает байт r/m в регистр размером в байт.
8B /r	MOV r16,r/m16	1	2/2	/5	2/9+EA	Перемещает слово r/m в регистр размером в слово.

8B /r	MOV r32,r/m32	1	2/2			Перемещает двойное слово r/m в регистр размером в двойное слово.
8C /r	MOV r/m16,Sreg	3/	2/2	/3	2/9+EA	Перемещает двойное слово r/m в регистр размером в двойное слово.
8D /r	MOV Sreg,r/m16	3/	2/5 pm= 1/19	/3 m=17/	2/9+EA 9	Перемещает слово r/m в сегментный
A0	MOV AL,moffs8	1	4		10	Перемещает байт по адресу (сегмент: смещение) в AX.
A1	MOV AX,moffs16	1	4		10	Перемещает слово по адресу (сегмент: смещение) в AX.
A2	MOV EAX, moffs32	1	4		10	Перемещает двойное слово по адресу (сегмент:смещение) в регистр EAX.
A2	MOV moffs8,AL	1	4		10	Перемещает AL по адресу (сегмент: смещение).
A3	MOV moffs16,AX	1	2		10	Перемещает AX по адресу (сегмент: смещение).
A3	MOV moffs16, EAX	1	2		10	Перемещает регистр EAX по адресу (сегмент:смещение).
B0	MOV reg8,imm8	1	2		4	Перемещает непосредственный байт в регистр.

B8+rw	MOV reg16, imm16	1	2		4	Перемещает непосредственное слово в регистр.
B8+rd	MOV reg32, imm32	1	2			Перемещает непосредственное двойное слово в регистр.
C6	MOV r/m8,imm8		2/2	2	3 /10+EA	Перемещает непосредственный байт в байт r/m.
C7	MOV r/m8,imm8		2/2	2	3 /10+EA	Перемещает непосредственное слово в слово r/m.
C7	MOV r/m32, imm32		2/2			Перемещает непосредственное двойное слово в двойное слово r/m.

Инструкция MOV копирует второй операнд в первый операнд.

Если целевым операндом (приемником) является сегментный регистр (DS, ES, SS и т.д.), то в регистр также загружаются данные из дескриптора. Данные для регистра получаются из записи таблицы дескриптора для заданного селектора. Нулевой селектор (значения от 0000 до 0003) могут загружаться в регистры DS и ES не вызывая исключительной ситуации, однако, использование регистров DS или ES вызывает #GP(0), и ссылок на память не происходит.

Загрузка с помощью инструкции MOV в регистр SS предотвращает все прерывания до выполнения следующей инструкции (предпочтительнее, чтобы это была инструкция MOV в SP).

MOV Перемещение данных в специальные регистры и из них (только для процессоров 386 и i486).

O D I T S Z A P C

Код операции	Инструкция	Такты 486	Такты 386	Описание

OF 22 /r	MOV CR0,r32	16		Перемещает данные из регистра в управляющий регистр.
OF 20 /r	MOV r32,CR0/ CR2/CR3	4	6	Перемещает данные из управляюще-го регистра в регистр.
OF 22 /r	MOV CR0/CR2/ CR3,r32	4	10/ 4/5	Перемещает данные отладочного регистра в регистр.
OF 21 /r	MOV r32,DR0-	10	22	Перемещает данные отладочного регистра в регистр.
OF 21 /r	MOV r32,DR6/ DR7	10	14	Перемещает данные отладочного регистра в регистр.
OF 23 /r	MOV DR6/DR7, r32	11	22	Перемещает данные из регистра в отладочный регистр.
OF 24 /r	MOV r32,TR6/ TR7	4	12	Перемещает данные из тестово-го регистра в регистр.
OF 26 /r	MOV TR6/TR7, r32	11	22	Перемещает данные из тестово-го регистра в регистр.
OF 24 /r	MOV r32,TR3 r32		3	Перемещает данные из регистра в тестовый регистр.

Эти формы инструкции MOV используются для загрузки из регистров общего назначения следующих специальных регистров или записи из регистров общего назначения в специальные регистры:

- управляющие регистры CR0, CR2 и CR3;

- отладочные регистры DR0, DR1, DR2, DR3, DR6 и DR7;

- тестовые регистры TR3, TR4, TR5, TR6 и TR7.

С этими инструкциями всегда используются 32-битовые операнды, независимо от атрибута размера операнда.

MOVS Перемещение данных из строки в строку (MOWSD только
MOVSB для процессоров 386 и i486).
MOVSW
MOVSD O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
A4	MOVS m8,m8	7	7	5	18	Перемещает байт из [(E)SI] в ES:[(E)DI].
A5	MOVS m16,m16	7	7	5	18	Перемещает слово из [(E)SI] в ES:[(E)DI].
A5	MOVS m32,m32	7	7			Перемещает двойное слово из [(E)SI] в ES:[(E)DI].
A4	MOVSB	7	7	5	18	Перемещает байт из DS: [(E)SI]в ES:[(E)DI].
A5	MOVSW	7	7	5	18	Перемещает слово из DS: [(E)SI]в ES:[(E)DI].
A5	MOVSD	7	7	5	18	Перемещает двойное слово из DS:[(E)SI]в ES:[(E)DI].

Инструкция MOVS копирует байт слова по адресу [(E)SI] в байт или слово в ES:[(E)DI]. Операнд-приемник должен быть адресуемым через регистр ES, переопределение сегмента в целевом операнде не допускается. В операнде-источнике переопределение сегмента использовать можно. По умолчанию используется регистр DS.

Адреса исходного и целевого операнда определяются исключительно содержимым регистров E(SI) и E(DI). Перед выполнением инструкции MOVS загрузите в (E)SI и (E)DI корректные значения индекса. Синонимами инструкции MOVS для работы с байтами, словами и двойными словами являются инструкции MOVSB, MOVSW и MOVSD.

После перемещения данных выполняется автоматическое продвижение регистров (E)SI и (E)DI. Если флаг направления равен 0 (была выполнена инструкция CLD), то регистры инкрементируются (увеличиваются), если флаг направления равен 1 (была выполнена инструкция STD), то декрементируются. При загрузке байта регистры

увеличиваются или уменьшаются на 1, при загрузке слова - на 2, а при загрузке двойного слова - на 4.

Инструкции MOVS может предшествовать префикс REP. Это делается для перемещения блока из CX байт или слов. Подробности этой операции можно найти в описании инструкции REP.

MOVSX Перемещение с расширением по знаку
(только для процессора 386 и i486).

O D I T S Z A P C

Код операции	Инструкция	Такты		Описание
		486	386	
OF BE /r	MOVSX r16,r/m8	3/3	3/6	Перемещает байт в слово с расширением знака.
OF BE /r	MOVSX r32,r/m8	3/3	3/6	Перемещает байт в двойное слово.
OF BE /r	MOVSX r32,r/m16	3/3	3/6	Перемещает слово в двойное слово.

Инструкция MOVS считывает содержимое действующего адреса или регистра, как байта или слова, распространяет знак значения в соответствии с атрибутом размера операнда (16 или 32 бита) и сохраняет результат в регистре-приемнике.

MOVZX Перемещение с расширением по нулю
(только для процессора 386 и i486).

O D I T S Z A P C

Код операции	Инструкция	Такты		Описание
		486	386	
OF B6 /r	MOVZX r16,r/m8	3/3	3/6	Перемещает байт в слово с расширением по нулю.

OF B6 /r	MOVZX r32,r/m8	3/3	3/6	Перемещает байт в двойное слово.
OF B7 /r	MOVZX r32,r/m16	3/3	3/6	Перемещает слово в двойное слово.

Инструкция MOVS считывает содержимое действующего адреса или регистра, как байта или слова, расширяет значение по нулю в соответствии с атрибутом размера операнда (16 или 32 бита) и сохраняет результат в регистре-приемнике.

MUL Беззнаковое умножение AL или AX.

O	D	I	T	S	Z	A	P	C
*								

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
F8 /4	MUL r/m8	13/18, 13/18	9-14/ 12-17	13/16	70-77/ 76-83 +EA	Беззнаковое умножение (AX[(AL * r/m байт).
F7 /4	MUL r/m16	13/26, 13/26	9-22/ 12-25	21/24	118-113/ 124-139+EA	Беззнаковое умножение(DX:AX[(AX * слово r/m).
F7 /4	MUL r/m32	13/42, 13/26	9-38/ 12-41		124-139+EA	Беззнаковое умножение (EDX:EAX[(EAX * двойное слово r/m).

Инструкция MOV выполняет беззнаковое умножение. Ее действие зависит от размера операнда:

- Байтовый операнд умножается на AL, результат остается в AX. Если AH = 0, то флаги переноса и переполнения устанавливаются в 0. В противном случае они устанавливаются в 1.

- Операнд размером в слово умножается на AX, результат остается в DX:AX. Регистр DX содержит старшие 16 бит произве-

дения. Если DX = 0, то флаги переноса и переполнения устанавливаются в 0. В противном случае они устанавливаются в 1.

- Операнд размером в слово умножается на EAX, результат остается в EDX:EAX. Регистр EDX содержит старшие 16 бит произведения. Если EDX = 0, то флаги переноса и переполнения устанавливаются в 0. В противном случае они устанавливаются в 1 (только для процессора 386).

NEG Отрицание (дополнение до двух).

O	D	I	T	S	Z	A	P	C
*				*	*	*	*	*

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
F6 /3	NEG r/m8	1/3	2/6	2/7	3/16+ EA	Отрицание байта r/m (дополнение до двух).
F7 /3	NEG r/m16	1/3	2/6	2/7	3/16+ EA	Отрицание слова r/m (дополнение до двух).
F7 /3	NEG r/m32	1/3	2/6			Отрицание двойного слова r/m (дополнение до двух).

Инструкция NEG заменяет значение регистра или операнда в памяти его дополнением до двух. Операнд вычитается из нуля, а результат помещается обратно в операнд.

Если операнд не равен нулю, то флаг переноса устанавливается в 1. В противном случае флаг переноса принимает нулевое значение.

NOP Пустая операция.

O	D	I	T	S	Z	A	P	C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
90	NOP	1	3	3	3	Нет операции.

Операция **NOP** не выполняет никакой операции. **NOP** - это однобайтовая инструкция, которая занимает место, но но не влияет на содержимое машины (кроме (E)IP).

NOP - это псевдоним инструкции XCHG (E)AX, (E)AX.

NOT Отрицание (дополнение до 1).

O	D	I	T	S	Z	A	P	C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
F6 /2	NOP r/m8	1/3	2/6	2/7	3/16+EA	Изменяет значение каждого бита байта r/m.
F7 /2	NOP r/m16	1/3	2/6	2/7	3/16+EA	Изменяет значение каждого бита слова r/m.
F7 /2	NOP r/m16	1/3	2/6	2/7	3/16+EA	Изменяет значение каждого бита двойного слова r/m.

Инструкция NOT инвертирует операнд, каждое единичное значение становится нулевым и наоборот.

OR Логическая операция ИЛИ.

O	D	I	T	S	Z	A	P	C
0				*	*	?	*	0

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
0C ib	OR AL,imm8	1	2	3	4	Операция ИЛИ над непосредственным байтом и AL.
0D iw	OR AX,imm16	1	2	3	4	Операция ИЛИ над непосредственным словом и AX.
0D id	OR EAX, imm32	1	2			Операция ИЛИ над непосредственным двойным словом и регистром EAX.
80 /1 ib	OR r/m8, imm8	1/3	2/7	/7	4/ 17+ EA	Операция ИЛИ над непосредственным байтом и байтом r/m.
81 /1 iw	OR r/m16, imm16	1/3	2/7	/7	4/ 17+ EA	Операция ИЛИ над непосредственным словом и словом r/m.
81 /1 iw	OR r/m32, imm32	1/3	2/7			Операция ИЛИ над непосредственным двойным словом и двойным словом r/m.
83 /1 ib	OR r/m16, imm8	1/3	2/7			Операция ИЛИ над непосредственным байтом, расширяемым по знаку, и словом r/m.
08 /r	OR r/m8,r8	1/3	2/6	/7	3/ 16+ EA	Операция ИЛИ над байтовым регистром и байтом r/m.
09 /r	OR r/m16, r16	1/3	2/6	/7	3/ 16+ EA	Операция ИЛИ над регистром размером в слово и словом r/m.

09 /r	OR r/m32, r32	1/3	2/6			Операция ИЛИ над регистром размером в двойное слово и двойным словом r/m.
0A /r	OR r8,r/m8	1/3	2/6	/7	3/ 16+ EA	Операция ИЛИ над байтовым регистром и байтом r/m.
0B /r	OR r16, r/m16	1/3	2/6	/7	3/ 16+ EA	Операция ИЛИ над регистром размером в слово и словом r/m.
0B /r	OR r32, r/m32	1/3	2/6			Операция ИЛИ над регистром размером в двойное слово и словом r/m.

Операция OR выполняет операцию ИЛИ над двумя операндами и помещает результат в первый операнд. Бит результата равен 0, если оба соответствующих бита операндов равны 0, и 1 в противном случае.

OUT Вывод данных в порт.

O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
E6 ib	OUT imm8,AL	16, pm= 11/31, vm=29	10, pm= 4/24	3	10	Выводит байт AL в порт, номер которого задается непосредственным операндом.
E7 ib	OUT imm8,AX	16, pm= 11/31, vm=29	10, pm= 4/25	3	10	Выводит слово AX в порт, номер которого задается непосредственным операндом.
E7 ib	OUT imm8, EAX	16, pm= 11/31, vm=29	10, pm= 4/25			Выводит двойное слово в EAX в порт, номер которого задается непосредственным операндом.
EE	OUT DX,AL	16, pm= 11/31, vm=29	11, pm= 5/25		8	Выводит байт AL в в порт, номер которого задается регистром DX.
EF	OUT DX,AX	16, pm= 11/31, vm=29	11, pm= 5/25		8	Выводит слово AX в в порт, номер которого задается регистром DX.
EF	OUT DX,EAX	16, pm= 11/31, vm=29	11, pm= 5/25		8	Выводит двойное слово в EAX в порт, номер которого задается регистром DX.

11, 5 - если CPL <= IOPL.
31, 25 - если CLP > IOPL или в виртуальном режиме процессора 8086.

Инструкция OUT пересылает байт или слово данных из регистра

(AL, AX или EAX) в порт, номер которого задается первым операндом. Вывод в порт с номером от 0 до 65535 выполняется путем занесения номера порта в регистр DX и выполнения инструкции OUT с DX в качестве первого операнда. Если инструкция содержит 8-битовый идентификатор порта, то значение расширяется (нулем) до 16 бит.

OUTS Вывод строки в порт (инструкции OUTS/OUTSB/OUTSW только для процессоров 80186/286/386/486, инструкция OUTSD
OUTSB - только для процессоров 386 и i486).
OUTSW
OUTSD

O D I T S Z A P C

Код операции	Инструкция	Такты			Описание
		486	386	286	
6E	OUTS DX,r/m8	17,pm=10/32, vm=30	14,pm=8/28	5	Выводит байт [(E)SI] в порт, номер которого задан в регистре DX.
6F	OUTS DX,r/m16	17,pm=10/32, vm=30	14,pm=8/28	5	Выводит слово [(E)SI] в порт, номер которого задан в регистре DX.
6F	OUTS DX,r/m32	17,pm=10/32, vm=30	14,pm=8/28		Выводит двойное слово [(E)SI] в порт, номер которого задан в регистре DX.
6E	OUTSB	17,pm=10/32, vm=30	14,pm=8/28	5	Выводит байт DS:[(E)SI] в порт, номер которого задан в регистре DX.
6E	OUTSW	17,pm=10/32, vm=30	14,pm=8/28	5	Выводит слово DS:[(E)SI] в порт, номер которого задан в регистре DX.
6F	OUTSD	17,pm=10/32, vm=30	14,pm=8/28		Выводит двойное слово DS:[(E)SI] в порт, номер которого задан в регистре DX.

Инструкция OUTS пересылает данные из байта, слова или двойного слова в памяти в соответствии с индексным регистром источника в порт вывода, адресуемый через регистр DX. Если атрибут размера адреса для данной инструкции равен 16 битам, то в качестве индексного регистра источника используется регистр SI. В противном случае, если атрибут размера адреса равен 32, то в качестве индексного используется регистр ESI.

Инструкция OUTS не позволяет задавать номер порта, как непосредственное значение. К порту нужно адресоваться через значение в регистре DX. Перед выполнением инструкции OUTS загрузите в регистр DX корректное значение.

Адрес исходных данных определяется содержимым индексного регистра источника. Перед выполнением инструкции OUTS загрузите в регистр SI или ESI.

После выполнения передачи данных исходный индексный регистр автоматически продвигается. Если флаг направления равен 0 (была выполнена инструкция CLD), то индексный регистр источника инкрементируется (увеличивается). Если флаг направления равен 1 (была выполнена инструкция STD), то он декрементируется (уменьшается).

Инструкции OUTSB, OUTSW и OUTSD являются синонимами инструкции OUTS для работы с байтами, словами и двойными словами. Инструкции OUTS может предшествовать префикс REP. Это позволяет выводить блоки из CX байт или слов. Подробности этой операции можно найти в описании REP.

POP Извлекает слово из стека.

O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
8F /0	POP m16	6	5	5	17+EA	Извлекает слово из стека и помещает его в слово в памяти.
8F /0	POP m32	6	5			Извлекает верхний элемент из стека и помещает его в двойное слово в памяти.
58+rw	POP r16	4	4		8	Извлекает верхний элемент из стека и помещает его в регистр размером в слово.
58+rd	POP r32	4	4			Извлекает верхний элемент из стека и помещает его в регистр размером в двойное слово.
1F	POP DS	3	7,pm=21	5,pm=20	8	Извлекает верхний элемент из стека и помещает его в регистр DS.
07	POP ES	3	7,pm=21	5,pm=20	8	Извлекает верхний элемент из стека и помещает его в регистр ES.
17	POP SS	3	7,pm=21	5,pm=20	8	Извлекает верхний элемент из стека и помещает его в регистр SS.
0F A1	POP FS	3	7,pm=21			Извлекает верхний элемент из стека и помещает его в регистр FS.
0F A9	POP GS	3	7,pm=			Извлекает верхний эле-

			21			мент из стека и поме- щает его в регистр GS.

Инструкция POP заменяет предыдущее содержимое операнда в памяти, регистра или сегментного регистра словом, которое находится в вершине стека и адресуется через SS:SP (если атрибут размера адреса равен 16) или SS:ESP (если атрибут размера адреса равен 32). Указатель стека SP увеличивается на 2 для операнда размером в 16 бит и на 4 для операнда размером в 32 бита. После этого он указывает на новую вершину стека.

POP CS - это не инструкция. Извлечение из стека в регистр CS выполняется с помощью инструкции RET.

Если операндом-приемником является сегментный регистр (DS, ES, FS, GS или SS), то извлекаемое из стека значение должно быть селектором. В защищенном режиме загрузка селектора инициализирует автоматическую загрузку в скрытую часть сегментного регистра связанной с селектором информации о дескрипторе. Загрузка инициализирует также проверку допустимости информации селектора и дескриптора.

Нулевое значение (0000 - 0003) должно извлекаться и заноситься в регистры DS, ES, FS или GS не вызывая исключительной ситуации по нарушению защиты. Попытка ссылки на значение сегмента, соответствующий сегментный регистр которого загружен нулевым значением, вызывает исключительную ситуацию #GP(0). Ссылки на память не происходит. Сохраненное значение сегментного регистра равно нулю.

Инструкция POP SS запрещает все прерывания, включая NMI (немаскируемые), до выполнения следующей инструкции. Это позволяет последовательно выполнять инструкции POP SS и POP ESP без опасности получения в процессе прерывания недопустимого стека. Однако, предпочтительным методом загрузки регистров SS и ESP является использование инструкции LSS.

> Примечание: Турбо Ассемблер расширяет синтаксис инструкции POP, чтобы можно было извлекать несколько элементов последовательности. Извлекаемые элементы могут включать в себя любое допустимое значение POP, включая регистры, непосредственные значения и ячейки памяти. На генерируемый код данное средство в действительности не влияет.

POPA Извлекает из стека все общие регистры (инструкция POPA
POPAD только для процессоров 80186/286/386/486, инструкция POPAD - только для процессоров 386 и i486).

O	D	I	T	S	Z	A	P	C

Код операции	Инструкция	Такты			Описание
		486	386	286	
61	POPA	9	24	19	Извлекает из стека регистр DI.
61	POPAD	9	24	19	Извлекает из стека EDI.

Инструкция POPA извлекает из стека восемь 16-разрядных общих регистров. Однако, вместо загрузки в SP значение SP отбрасывается. Инструкция POPA изменяет действие предыдущей инструкции PUSHA на обратное, восстанавливая общие регистры в те значения, которые у них были до инструкции PUSHA. Первым извлекается регистр DI.

Инструкция POPAD извлекает из стека восемь 32-разрядных общих регистров. Однако, вместо загрузки в ESP значение ESP отбрасывается. Инструкция POPAD изменяет действие предыдущей инструкции PUSHAD на обратное, восстанавливая общие регистры в те значения, которые у них были до инструкции PUSHAD. Первым извлекается регистр EDI.

POPF Извлекает из стека регистр FLAGS (регистр флагов) или
POPFD EFLAGS (инструкция POPFD - только для процессоров 386 и i486).

O	D	I	T	S	Z	A	P	C
*	*	*	*	*	*	*	*	*

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
9D	POPF	9,pm=6	5	5	8	Извлекает из стека FLAGS.
9D	POPFD	9,pm=6	5			Извлекает из стека EFLAGS.

Инструкции POPF/POPFD извлекают слово или двойное слово из вершины стека и записывают это значение в регистре флагов. Если

атрибут размера операнда в инструкции равен 16, то слово извлекается, а значение сохраняется в регистре флагов FLAGS. Если атрибут размера операнда равен 32, то извлекается двойное слово, а значение записывается в EFLAGS.

Заметим, что биты 16 и 17 регистра EFLAGS, которые называются соответственно VM и RF инструкцией POPF или POPFD не изменяются.

Уровень привилегий ввода-вывода изменяется только при выполнении на привилегированном уровне 0. Флаг прерываний изменяется только при выполнении на уровне, привилегии которого по крайней мере не ниже, чем уровень привилегий ввода-вывода. (Режим реальной адресации эквивалентен уровню привилегий 0.) Если инструкция POPF выполняется с недостаточными привилегиями, исключительная ситуация не возникает, а биты привилегий не изменяются.

PUSH Заносит операнд в стек.

O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
FF /6	PUSH m16	4	5	5	16 +EA	Заносит в стек слово из памяти.
FF /6	PUSH m32	4	5			Заносит в стек двойное слово из памяти.
50+/r	PUSH r16	1	2	3	11	Заносит в стек слово из регистра.
50+/r	PUSH r32	1	2			Заносит в стек двойное слово из регистра.
6A	PUSH imm8	1	2	3		Заносит в стек непосредственный байт.
68	PUSH imm16	1	2	3		Заносит в стек непосредственное слово.
68	PUSH imm32	1	2			Заносит в стек непосредственное двойное слово.

0E	PUSH CS	3	2	3	10	Заносит в стек регистр CS.
16	PUSH SS	3	2	3	10	Заносит в стек регистр SS.
1E	PUSH DS	3	2	3	10	Заносит в стек регистр DS.
06	PUSH ES	3	2		10	Заносит в стек регистр ES.
0F A0	PUSH FS	3	2			Заносит в стек регистр FS.
0F A8	PUSH GS	3	2			Заносит в стек регистр GS.

Инструкция PUSH уменьшает указатель стека на 2, если атрибут размера операнда инструкции равен 16, в противном случае указатель стека уменьшается на 4. Затем инструкция PUSH помещает операнд в новую вершину стека, на которую указывает указатель стека.

Инструкция PUSH ESP процессора 386 заносит в стек значение ESP, которое существовало до выполнения инструкции. Инструкция PUSH SP процессора 80286 также заносит в стек значение регистра SP, которое существовало до выполнения инструкции. Это отличается от процессора 8086, который заносит в стек новое значение (уменьшенное на 2).

Примечание: Турбо Ассемблер расширяет синтаксис инструкции PUSH, чтобы можно было заносить в стек несколько элементов последовательности. Заносимые в стек элементы могут включать в себя любое допустимое для инструкции PUSH значение, включая регистры, непосредственные значения и ячейки памяти. На генерируемый код данное средство в действительности не влияет. Кроме того, инструкция PUSH допускает использование аргумента-константы, даже для процессора 8086. Такие инструкции заменяются в объектном коде 10-байтовой последовательностью, которая моделирует инструкцию PUSH процессора 80186/286/286 с непосредственным значением.

PUSHA
PUSHAD — Заносит в стек все общие регистры (инструкция PUSHA только для процессоров 80186/286/386/486, инструкция PUSHAD - только для процессоров 386 и i486).

O D I T S Z A P C

Код операции	Инструкция	Такты			Описание
		486	386	286	
61	PUSHA	11	18	17	Заносит в стек AX, CX, DX, BX, исходные SP, BP, SI.
61	PUSHAD	11	18		Заносит в стек EAX, ECX, EDX, EBX.

Инструкция PUSHA сохраняет в стеке восемь 16-разрядных общих регистров. Инструкция PUSHA, чтобы можно было записать 8 значений размером в слово, уменьшает указатель стека (SP) на 16. Инструкция PUSHAD уменьшает указатель (ESP) стека на 32, чтобы можно было записать 8 значений размером в двойное слово. Поскольку регистры заносятся в стек в том порядке, как они задаются, извлекаться они будут в виде 16 или 32 байт стека в обратном порядке. Последним заносится регистр DI или EDI.

PUSHF
PUSHFD

Заносит в стек регистр флагов (инструкция PUSHFD - только для процессоров 386 и i486).

O	D	I	T	S	Z	A	P	C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
9C	PUSHF	4,pm=3	4	5	10	Заносит в стек FLAGS.
9C	PUSHFD	4,pm=3	4			Заносит в стек EFLAGS.

Инструкция PUSHF уменьшает указатель стека на 2 и копирует регистр флагов FLAGS в новую вершину стека. Инструкция PUSHFD уменьшает указатель стека на 4, а регистр EFLAGS процессора 386 копируется в новую вершину стека, на которую указывает EE:ESP.

RCR
RCL
ROL
ROR

Циклический сдвиг.

O	D	I	T	S	Z	A	P	C
*								*

Код операции	Инструкция	Такты				Описание

		486	386	286	86	
D0 /2	RCL r/m8,1	3/4	9/10	2/7	2/15+EA	Циклическая перестановка 9 бит (CF, байта r/m) влево 1 раз.
D2 /2	RCL r/m8,CL	8-30 /9-31	9/10	5/8	8+4 на бит/(20 + 4 на бит)+EA	Циклическая перестановка 9 бит (CF, байта r/m) влево CL раз.
C0 /2 ib	RCL r/m8, imm8	8-30 /9-31	9/10			Циклическая перестановка 9 бит (CF, байта r/m) влево imm8 раз.
D1 /2 ib	RCL r/m16,1	3/4 /9-31	9/10	2/7	2/15+EA	Циклическая перестановка 17 бит (CF, слова r/m) влево 1 раз.
D3 /2	RCL r/m16, CL	8-30 /9-31	9/10	5/8	8+4 на бит/(20 + 4 на бит)+EA	Циклическая перестановка 17 бит (CF, слова r/m) влево CL раз.
C1 /2 ib	RCL r/m16, imm8	8-30 /9-31	9/10	5/8		Циклическая перестановка 17 бит (CF, слова r/m) влево imm8 раз.
D1 /2	RCL r/m32,1	3/4 /9-31	9/10			Циклическая перестановка 33 бит (CF, двойного слова r/m) влево 1 раз.
D3 /2	RCL r/m16, CL	8-30 /9-31	9/10			Циклическая перестановка 33 бит (CF, двойного слова r/m) влево CL раз.
C1 /2 ib	RCL r/m32, imm8	8-30 /9-31	9/10	5/8		Циклическая перестановка 33 бит (CF, двойного слова r/m)

						влево imm8 раз.
D0 /3	RCR r/m8,1	3/4	9/10	2/7	2/15+EA	Циклическая перестановка 9 бит (CF, байта r/m) вправо 1 раз.
D2 /3	RCR r/m8,CL	8-30 /9-31	9/10	5/8	8+4 на бит/(20 + 4 на бит)+EA	Циклическая перестановка 9 бит (CF, байта r/m) вправо CL раз.
C0 /3 ib	RCR r/m8, imm8	8-30 /9-31	9/10			Циклическая перестановка 9 бит (CF, байта r/m) вправо imm8 раз.
D1 /3 ib	RCR r/m16,1	3/4 /9-31	9/10	2/7	2/15+EA	Циклическая перестановка 17 бит (CF, слова r/m) вправо 1 раз.
D3 /2	RCR r/m16, CL	8-30 /9-31	9/10	5/8	8+4 на бит/(20 + 4 на бит)+EA	Циклическая перестановка 17 бит (CF, слова r/m) вправо CL раз.
C1 /2 ib	RCR r/m16, imm8	8-30 /9-31	9/10	5/8		Циклическая перестановка 17 бит (CF, слова r/m) вправо imm8 раз.
D1 /2	RCR r/m32,1	3/4 /9-31	9/10			Циклическая перестановка 33 бит (CF, двойного слова r/m) вправо 1 раз.
D3 /2	RCR r/m16, CL	8-30 /9-31	9/10			Циклическая перестановка 33 бит (CF, двойного слова r/m) вправо CL раз.
C1 /0 ib	RCR r/m32, imm8	8-30 /9-	9/10	5/8		Циклическая перестановка 33 бит (CF,

		31				двойного слова r/m) вправо imm8 раз.
D0 /0	ROL r/m8,1	3/4	9/10	2/7	2/15+EA	Перестановка 8 бит байта r/m влево 1 раз.
D2 /0	ROL r/m8,CL	8-30 /9-31	9/10	5/8	8+4 на бит/(20 + 4 на бит)+EA	Перестановка 8 бит байта r/m влево CL раз.
C0 /0 ib	ROL r/m8, imm8	8-30 /9-31	9/10			Перестановка 8 бит байта r/m влево imm8 раз.
D1 /0 ib	ROL r/m16,1	3/4 /9-31	9/10	2/7	2/15+EA	Перестановка 16 бит слова r/m влево 1 раз.
D3 /0	ROL r/m16, CL	8-30 /9-31	9/10	5/8	8+4 на бит/(20 + 4 на бит)+EA	Перестановка 16 бит слова r/m влево CL раз.
C1 /0 ib	ROL r/m16, imm8	8-30 /9-31	9/10	5/8		Перестановка 16 бит слова r/m) влево imm8 раз.
D1 /0	ROL r/m32,1	3/4 /9-13	9/10			Перестановка 32 бит двойного слова r/m влево 1 раз.
D3 /0	ROL r/m16, CL	8-30 /9-31	9/10			Перестановка 32 бит двойного слова r/m влево CL раз.
C1 /0 ib	ROL r/m32, imm8	8-30 /9-31	9/10	5/8		Перестановка 32 бит двойного слова r/m влево imm8 раз.
D0 /1	ROR r/m8,1	3/4	9/10	2/7	2/15+EA	Перестановка 9 бит байта r/m вправо 1

						раз.
D2 /1	ROR r/m8,CL	8-**30** /9-31	9/10	5/8	8+4 на бит/(20 + 4 на бит)+EA	Перестановка 9 бит байта r/m вправо CL раз.
C0 /1 ib	ROR r/m8, imm8	8-**30** /9-31	9/10			Перестановка 9 бит байта r/m вправо imm8 раз.
D1 /1 ib	ROR r/m16,1	3/4 /9-31	9/10	2/7	2/15+EA	Перестановка 16 бит слова r/m вправо 1 раз.
D3 /1	ROR r/m16, CL	8-**30** /9-31	9/10	5/8	8+4 на бит/(20 + 4 на бит)+EA	Перестановка 16 бит слова r/m вправо CL раз.
C1 /1 ib	ROR r/m16, imm8	8-**30** /9-31	9/10	5/8		Перестановка 16 бит слова r/m вправо imm8 раз.
D1 /1	ROR r/m32,1	3/4 /9-31	9/10			Перестановка 32 бит двойного слова r/m вправо 1 раз.
D3 /1	ROR r/m16, CL	8-**30** /9-31	9/10			Перестановка 32 бит двойного слова r/m вправо CL раз.
C1 /1 ib	ROR r/m32, imm8	8-**30** /9-31	9/10	5/8		Перестановка 32 бит двойного слова r/m вправо imm8 раз.
Для каждой выполненной перестановки добавьте к указанным временным значениям 1 (только для процессора 80286).						

Каждая инструкция перестановки (сдвига) сдвигает биты регистра или операнда в памяти. Инструкции сдвига влево сдвигают биты в направлении возрастания адресов, кроме верхнего бита, который возвращается "вниз". Инструкции сдвига вправо делают обрат-

ное: биты сдвигаются в направлении уменьшения адресов, а нижний бит сдвигается "вверх".

Для инструкций RCL и RCR флаг переноса является частью перемещаемой группы бит. Инструкция RCL сдвигает флаг переноса в младший (нижний) бит и сдвигает старший (верхний) бит во флаг переноса. Инструкция RCR сдвигает флаг переноса в старший бит, а младший бит - во флаг переноса. Для инструкций ROL и ROR исходное значение флага переноса не является частью результата, но во флаг переноса заносится копия бита, который был сдвинут из конца группы бит.

Операция сдвига повторяется столько раз, сколько задается вторым операндом, который может представлять собой непосредственное число или содержимое регистра CL. Чтобы уменьшить максимальное время выполнения инструкции, процессоры 80286/386 не позволяют использовать значения счетчика, превышающее 31. Если делается попытка использовать значение счетчика, превышающее 31, то используются только младшие 5 бит счетчика. В процессоре 8086 счетчики перестановки не маскируются. В процессорах 386 в виртуальном режиме 8086 выполняется маскирование счетчиком перестановки.

Флаг переполнения определен только для тех форм инструкции, где второй операнд равен 1. Во всех других случаях он не определен. Для сдвигов/циклических перестановок влево выполняется операция XOR над битом CF и старшим битом результата. Для сдвигов/циклических перестановок вправо операция XOR выполняется над старшими двумя битами результата для получения флага OF.

REP Повторения последующей строковой операции.
REPE
REPZ
REPZ
REPNE
REPNZ

O	D	I	T	S	Z	A	P	C
					*			

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
F3 6C	REP INS r/m8,DX	16+ 8(E)CX, pm=10+ 8(E)CX /30+ 8(E)CX, vm=29+ 8(E)CX	13+ 6(E)CX pm=7+ 6(E)CX /27+ 6(E)CX	5+ 4CX		Ввод (E)CX байт из порта DX в ES:[(E)DI].
F3 6D	REP INS	16+	13+	5+		Ввод (E)CX слов из пор-

	r/m16,DX	8(E)CX, pm=10+ 8(E)CX /30+ 8(E)CX, vm=29+ 8(E)CX	6(E)CX pm=7+ 6(E)CX /27+ 6(E)CX	4CX		та DX в ES:[(E)DI].
F3 6D	REP INS r/m32,DX	16+ 8(E)CX, pm=10+ 8(E)CX /30+ 8(E)CX, vm=29+ 8(E)CX	13+ 6(E)CX pm=7+ 6(E)CX /27+ 6(E)CX			Ввод (E)CX двойных слов из порта DX в ES: [(E)DI].
F3 A4	REP MOVS m8,m8	5,13, 12+3(E) CX 8(E)CX /30+ 8(E)CX, vm=29+ 8(E)CX	5+4(E) CX 6(E)CX /27+ 6(E)CX	5+ 4CX	9+ 17 CX	Ввод (E)CX байт из [(E)SI] в ES:[(E)DI].
F3 A5	REP MOVS m16,m16	5,13, 12+3(E) CX 8(E)CX /30+ 8(E)CX, vm=29+ 8(E)CX	5+4(E) CX 6(E)CX /27+ 6(E)CX	5+ 4CX	9+ 17 CX	Перемещение (E)CX слов из [(E)CX] в ES:[(E) DI].
F3 A5	REP MOVS m32,m32	5,13, 12+3(E) CX 8(E)CX /30+ 8(E)CX, vm=29+ 8(E)CX	5+4(E) CX 6(E)CX /27+ 6(E)CX			Перемещение (E)CX двойных слов из [(E) CX] в ES:[(E)DI].
F3 6E	REP OUTS DX,r/m8	17+5(E) CX,pm= 11+5(E) CX/31+	5+12(CX,pm= 6+5(E) CX/26+	5+ 4(E)CX		Вывод (E)CX байт из [E)SI] в порт DX.

		5(E)CX	5(E)CX /26+5 (E)CX			
F3 6F	REP OUTS DX,r/m16	17+5(E) CX,pm= 11+5(E) CX/31+ 5(E)CX	5+12(CX,pm= 6+5(E) CX/26+ 5(E)CX /26+5 (E)CX	5+ 4(E)CX		Вывод (E)CX слов из [E)SI] в порт DX.
F3 6F	REP OUTS DX,r/m32	17+5(E) CX,pm= 11+5(E) CX/31+ 5(E)CX	5+12(E CX,pm= 6+5(E) CX/26+ 5(E)CX /26+5 (E)CX			Вывод (E)CX двойных слов из [E)SI] в порт DX.
F3 AC	REP LODS m8	5,7+ 4(E)CX				Загрузка (E)CX байт из [(E)SI] в AL.
F3 AD	REP LODS m16	5,7+ 4(E)CX				Загрузка (E)CX слов из [(E)SI] в AX.
F3 AD	REP LODS m32	5,7+ 4(E)CX				Загрузка (E)CX двойных слов [(E)SI] в EAX.
F3 AA	REP STOS m8	5,7+ 4(E)CX	7+5(E) CX	9+ 10 CX	9+ 10 CX	Заполнение (E)CX байт ES:[(E)DI] из AL.
F3 AB	REP STOS m16	5,7+ 4(E)CX	7+5(E) CX	9+ 10 CX	9+ 10 CX	Заполнение (E)CX слов ES:[(E)DI] из AX.
F3 AB	REP STOS m32	5,7+ 4(E)CX	7+5(E) CX			Заполнение (E)CX двой- ных ES:[(E)DI] из EAX.
F3 A6	REPE CMPS m8,m8	5,7+ 7(E)CX	5+9N	5+ 9N	5+ 9N	Поиск несовпадающих байт в ES:[(E)DI] и [(E)SI].

F3 A7	REPE CMPS m16,m16	5,7+ 7(E)CX	5+9N	5+ 9N	9+ 22 N	Поиск несовпадающих слов в ES:[(E)DI] и [(E)SI].
F3 A7	REPE CMPS m32,m32	5,7+ 7(E)CX	5+9N			Поиск несовпадающих двойных слов в ES: [(E)DI] и [(E)SI].
F3 AE	REPE SCAS m8	5,7+ 7(E)CX	5+8N	5+ 9N	9+ 15 N	Поиск отличного от AL байта, начиная с ES: [(E)DI].
F3 AF	REPE SCAS m16	5,7+ 7(E)CX	5+8N	5+ 9N	9+ 15 N	Поиск отличного от AX слова, начиная с ES: [(E)DI].
F3 AF	REPE SCAS m32	5,7+ 7(E)CX	5+8N			Поиск отличного от EAX слова, начиная с ES: [(E)DI].
F2 A6	REPNE CMPS m8,m8	5,7+ 7(E)CX	5+9N	5+ 9N	9+ 22 N	Поиск совпадающих байт в ES:[(E)DI] и [(E)SI].
F2 A7	REPNE CMPS m16,m16	5,7+ 7(E)CX	5+9N	5+ 9N	9+ 15 N	Поиск совпадающих слов в ES:[(E)DI] и [(E)SI].
F2 A7	REPNE CMPS m16,m16	5,7+ 7(E)CX	5+9N	5+ 9N	9+ 15 N	Поиск совпадающих двойных слов в ES:[(E) DI] и [(E)SI].
F2 AE	REPNE CMPS m16,m16	5,7+ 7(E)CX	5+9N	5+ 9N	9+ 15 N	Поиск совпадающих двойных слов в ES:[(E) DI] и [(E)SI].
F2 AE	REPNE SCAS m8	5,7+ 7(E)CX	5+8N	5+ 9N	9+ 15 N	Поиск AL.
F2 AF	REPNE SCAS m8	5,7+ 7(E)CX	5+8N	5+ 9N	9+ 15	Поиск AX.

					N	
F2 AF	REPNE SCASm32	5,7+ 7(E)CX	5+8N			Поиск EAX.

4,3,6,9,10,22 - если CPL <= IOPL
5CX,6CX - если CPL > IOPL.
5 - если (E)CX = 0.
13 - если (E)CX = 1.
CX - если (E)CX 1.
7(E)CX - если (E)CX 0.

REP, REPE (повторять, пока равно) и REPNE (повторять, пока не равно) - это префиксы, которые применяются к строковым операциям. Каждый префикс вызывает повторение выполнения выполнения последующей строковой инструкции столько раз, сколько задается в регистре-счетчике, или до (не)выполнения определенного условия во флаге нуля (REPE, REPNE).

Синонимами инструкции REPE и REPNE являются соответственно инструкции REPZ и REPNZ.

Префикс REP применяется только к одной строковой инструкции. Для повторения блока инструкций используйте инструкцию LOOP или другую конструкцию цикла.

На каждой итерации выполняется следующее действие:

1. Если атрибут размера адреса операнда равен 16 битам, в качестве счетчика используется регистр CX. Если атрибут размера адреса равен 32, то в качестве счетчика используется регистр ECX.

2. Проверяется значение регистра CX. Если он равен нулю, то выполняется выход из итерации и переход к следующей инструкции.

3. Запрашиваются все отложенные прерывания.

4. Один раз выполняется строковая операция.

5. Значение регистра CX или ECX уменьшается на 1. Флаги при этом не модифицируются.

6. Если строковой операцией является операция SCAC или CMPS, то проверяется содержимое флага нуля. Если условие повторения не удовлетворяется, то выполняется выход из итерации и перемещение к следующей инструкции. Выход из итерации выполняется, если префиксом является REPE, и флаг ZF равен 0 (последнее сравнение не дало равенства), или пре-

фиксом является REPNE и флаг ZF равен 1 (последнее сравнение дало равенство).

7. Возврат на шаг 1 для каждой итерации.

Выйти из повторяющихся инструкций CMPS и SCAS можно, если счетчик исчерпан или условие повторения не удовлетворяется в соответствии с флагом нуля. Эти два случая можно разделить, либо используя инструкцию JCXZ, либо инструкцию условного перехода, при которой проверяется флаг нуля (JZ, JNZ и JNE).

RET Возврат из процедуры.

O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
C3	RET	5	10+m	11	16	Ближний возврат в вызывающую программу.
CB	RET	13,pm=18	18+m, pm=32+m	15, pm=25	16	Дальний возврат в вызывающую программу с теми же привилегиями.
CB	RET	13,pm=33	pm=68	55		Дальний возврат.
C2 iw	RET imm16	5	10+m	11	20	Ближний возврат.
CA iw	RET imm16	14, pm=17	18+m, pm=32+m	15, pm=25	25	Дальний возврат и извлечение 16 байт.
CA iw	RET imm16	14, pm=33	pm=68	55		Дальний возврат.

Инструкция RET передает управление по адресу возврата, который находится в стеке. Адрес обычно помещается в стек инструкцией CALL, а возврат выполняется на инструкцию, следующую за CALL.

Необязательный числовой параметр инструкции CALL задает число байт (атрибут равен 16) или слов (атрибут равен 32) в стеке, которые должны освобождаться после извлечения адреса возврата. Эти элементы используются обычно, как входные параметры вызываемой процедуры.

Для возврата ближнего типа (внутрисегментного) адрес в стеке представляет собой смещение в сегменте, которое извлекается из стека в указатель инструкций. Регистр CS не изменяется. При межсегментном (дальнем) возврате адрес в стеке представляет собой указатель дальнего типа. Сначала извлекается смещение, за ним - селектор.

В реальном режиме регистры CS и IP загружаются непосредственно. В защищенном режиме межсегментный возврат вызывает проверку процессором дескриптора, адресуемого селектором возврата. Байт AR дескриптора должен указывать на сегмент кода с равными или меньшими привилегиями (или большим либо равным числовым значением), чем текущий уровень привилегий. Возврат к более низкому уровню привилегий вызывает перезагрузку стека из значения, сохраненного перед блоком параметров.

В процессе межуровневой передачи сегментные регистры DS, ES, FS и GS могут устанавливаться инструкцией RET в нулевое значение. Если эти регистры ссылаются на сегмент, который не может использоваться на новом уровне привилегий, то, чтобы предотвратить неуполномоченный доступ из нового уровня привилегий, они устанавливаются в значение 0.

SAHF Запись регистра AH в регистр флагов.

O	D	I	T	S	Z	A	P	C
				*	*	*	*	*

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
9E	SAHF	2	3	2	4	Запись AH во флаги SF ZF xx AF xx PF xx CF.

Инструкция SAHF загружает в перечисленные выше флаги значения из регистра AH (биты 7, 6, 4, 2 и 0 соответственно).

SAL Инструкции сдвига.
SAR
SHL
SHR

O	D	I	T	S	Z	A	P	C
*				*	*	?	*	*

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
D0 /4	SAL r/m8,1	3/4	3/7	2/7	2/15+EA	Умножает байт r/m на 2.
D2 /4	SAL r/m8, CL	3/4	3/7	5/8	8+4 на бит/(20+4 на бит)+EA	Умножает байт r/m на 2 CL раз.
C0 /4 ib	SAL r/m8, imm8	2/4	3/7	5/8		Умножает байт r/m на 2.
D1 /4	SAL r/m16, 1	3/4	3/7	2/7	2/15+EA	Умножает слово r/m на 2.
D3 /4	SAL r/m16, CL	3/4	3/7	5/8	8+4 на бит/(20+4 на бит)+EA	Умножает слово r/m на 2 CL раз.
C1 /4 ib	SAL r/m16, imm8	2/4	3/7	5/8		Умножает слово r/m на 2.
D1 /4	SAL r/m32, 1	3/4	3/7			Умножает двойное слово r/m на 2.
D3 /4	SAL r/m32, CL	3/4	3/7			Умножает двойное слово r/m на 2 CL раз.
C1 /4 ib	SAL r/m32, imm8	2/4	3/7			Умножает двойное слово r/m на 2.
D0 /7	SAR r/m8,1	3/4	3/7	2/7	2/15+EA	Деление со знаком байта r/m на 2.
D2 /7	SAR r/m8, CL	3/4	3/7	2/7	8+4 на бит/(20+4 на бит)+EA	Деление со знаком байта r/m на 2.

C0 /7 ib	SAR r/m8, imm8	2/4	3/7	5/8		Деление со знаком байта r/m на 2.
D1 /7	SAR r/m16, imm8	2/4	3/7	5/8	2/15+EA	Деление со знаком слова r/m на 2.
D3 /7	SAR r/m16, CL	3/4	3/7	5/8	8+4 на бит/(20+ 4 на бит)+EA	Деление со знаком слова r/m на 2.
C1 /7 ib	SAR r/m16, imm8	2/4	3/7	5/8		Деление со знаком слова r/m на 2.
D1 /7	SAR r/m32, 1	3/4	3/7			Деление со знаком двой- ного слова r/m на 2.
D3 /7	SAR r/m32, CL	3/4	3/7			Деление со знаком двой- ного слова r/m на 2 CL раз.
C1 /7	SAR r/m32, imm8	2/4	3/7			Деление со знаком двой- ного слова r/m на 2.
D0 /4	SHL r/m8,1	3/4	3/7	2/7	2/15+EA	Умножение байта r/m на два.
D2 /4	SHL r/m8, CL	3/4	3/7	5/8	8+4 на бит/(20+ 4 на бит+EA)	Умножение байта r/m на два CL раз.
C0 /4 ib	SHL r/m8, imm8	2/4	3/7	5/8		Умножение байта r/m на два.
D1 /4	SHL r/m16, 1	2/4	3/7	2/7	2/15+EA	Умножение слова r/m на два CL раз.
D3 /4	SHL r/m16, CL	3/4	3/7	5/8	8+4 на бит/(20+	Умножение слова r/m на два CL раз.

					4 на бит)+EA	
C1 /4 ib	SHL r/m16, imm8	2/4	3/7	5/8		Умножение слова r/m на два.
D1 /4 ib	SHL r/m32, 1	2/4	3/7			Умножение двойного слова r/m на два.
C1 /4 ib	SHL r/m32, CL	2/4	3/7			Умножение двойного слова r/m на два.
C1 /4 ib	SHL r/m32, imm8	2/4	3/7	5/8		Умножение двойного слова r/m на два.
D0 /5	SHR r/m8,1	3/4	3/7	2/7	2/15+EA	Беззнаковое деление байта r/m на 2.
D2 /5	SHR r/m8, CL	3/4	3/7	2/7	8+4 на бит/(20+4 на бит)+EA	Беззнаковое деление байта r/m на 2.
C0 /5 ib	SHR r/m8, imm8	2/4	3/7	5/8		Беззнаковое деление байта r/m на 2.
D1 /5	SHR r/m16, imm8	2/4	3/7	5/8	2/15+EA	Беззнаковое деление слова r/m на 2.
D3 /5	SHR r/m16, CL	3/4	3/7	5/8	8+4 на бит/(20+4 на бит)+EA	Беззнаковое деление слова r/m на 2.
C1 /5 ib	SHR r/m16, imm8	2/4	3/7	5/8		Беззнаковое деление слова r/m на 2.
D1 /5	SHR r/m32, 1	3/4	3/7			Беззнаковое деление двойного слова r/m на 2.

D3 /5	SHR r/m32, CL	3/4	3/7			Беззнаковое деление двойного слова r/m на 2.
C1 /5	SHR r/m32, imm8	2/4	3/7			Беззнаковое деление двойного слова r/m на 2.

Инструкция SAL (или ее синоним, инструкция SHL) сдвигает биты операнда "вверх". Старший бит сдвигается во флаг переноса, а младший бит устанавливается в нулевое значение.

Сдвиг повторяется число раз, указанное вторым операндом, который представляет собой непосредственное значение или содержимое регистра CL. Чтобы уменьшить максимальное время выполнения, процессоры 80286/286 не позволяют сдвигать операнды более, чем на 31. При попытке использовать значение счетчика, превышающее 31, используются только младшие пять бит счетчика сдвига. (Процессор 8086 использует все 8 бит счетчика сдвига.)

Флаг переполнения устанавливается только при использовании форм инструкций с единичным значением счетчика. При сдвиге влево OF устанавливается в 0, если старший бит результата тот же, что и результат флага переноса (то есть старшие два бита исходного операнда совпадали). Если они различны, OF устанавливается в 1. В инструкции SAR флаг OF устанавливается в 0 для всех сдвигов на 1. В инструкции SHR OF устанавливается в значение старшего бита исходного операнда.

SBB Целочисленное вычитание с заемом.
SAR
SHL

O	D	I	T	S	Z	A	P	C
*				*	*	*	*	*

SHR

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
1C ib	SBB AL,imm8	1	2	3	4	Вычитание с заемом непосредственного байта из AL.
1D iw	SBB AX,imm16	1	2	3	4	Вычитание с заемом непосредственного слова из AX.
1D id	SBB EAX,	1	2			Вычитание с заемом не-

	imm32					посредственного двойного слова из EAX.
80 /3 ib	SBB r/m8, imm8	1/3	2/7	3/7	4/17 +EA	Вычитание с заемом непосредственного байта из байта r/m.
81 /3 iw	SBB r/m16, imm16	1/3	2/7	3/7	4/17 +EA	Вычитание с заемом непосредственного слова из слова r/m.
81 /3 id	SBB r/m32, imm32	1/3	2/7			Вычитание с заемом непосредственного двойного слова из двойного слова r/m.
83 /3 iw	SBB r/m16, imm8	1/3	2/7	3/7	4/17 +EA	Вычитание с заемом непосредственного расширенного по знаку байта из слова r/m.
83 /3 iw	SBB r/m32, imm8	1/3	2/7			Вычитание с заемом непосредственного расширенного по знаку байта из двойного слова r/m.
18 /r	SBB r/m8, r8	1/3	2/6	2/7	3/16 +EA	Вычитание с заемом байтового регистра из байта r/m.
19 /r	SBB r/m16, r16	1/3	2/6	2/7	3/16 +EA	Вычитание с заемом регистра размером в слово из слова r/m.
19 /r	SBB r/m32, r32	1/3	2/6			Вычитание с заемом регистра размером в двойное слова из двойного слова r/m.
1A /r	SBB r8,r/m8 r32	1/2	2/7	2/7	3/9 +EA	Вычитание с заемом байтового регистра из байбайта r/m.

1B /r	SBB r16, r/m16	1/2	2/7	2/7	3/9 +EA	Вычитание с заемом регистра размером в слово из слова r/m.
1B /r	SBB r32, r/m32	1/2	2/7			Вычитание с заемом регистра размером в двойное слово из двойного слова r/m.

Инструкция SBB прибавляет второй операнд (приемник) к флагу переноса CF и вычитает результат из первого операнда (источника). Результат вычитания присваивается второму операнду (приемнику). Соответствующим образом изменяются флаги.

Когда из операнда размером в слово вычитается непосредственное байтовое значение, то оно расширяется по знаку.

SCAS Сравнение строковых данных (SCASD - только для
SCASB процессоров 386 и i486).
SCASW
SCASD

O	D	I	T	S	Z	A	P	C
*				*	*	*	*	*

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
AE	SCAS m8	6	7	7	15	Сравнение байт AL - ES:[DI].
AF	SCAS m16	6	7	7	15	Сравнение слов AX - ES:[DI].
AF	SCAS m32	6	7			Сравнение двойных слов EAX - ES:[DI].
AE	SCASB	6	7	7	15	Сравнение байт AL - ES:[DI].
AF	SCASW	6	7	7	15	Сравнение слов AX - ES:[DI].
AF	SCASD	6	7			Сравнение двойных слов EAX - ES:[DI].

Инструкция SCAS вычитает байт в памяти или слово в регистре-приемнике из регистра AL, AX или EAX. Результат отбрасывается,

устанавливаются только флаги. Операнд должен быть адресуемым из регистра ES, переопределение сегмента не допускается.

Если атрибут размера адреса данной инструкции равен 16, то регистр DI используется в качестве регистра-приемника. В противном случае атрибут размера адреса равен 32 битам, и используется регистр EDI.

Адрес данных в памяти, предназначенных для сравнения, определяется исключительно содержимым регистра-приемника, а не операндом SCAS. Операнд позволяет определить адресуемость через сегмент ES и определяет тип данных. Перед выполнением инструкции SCAS загрузите в DI или в EDI корректное значение индекса.

После сравнения регистр-приемник автоматически изменяется. Если флаг направления равен 0 (была выполнена инструкция CLD), то индексный регистр источника инкрементируется (увеличивается). Если флаг направления равен 1 (была выполнена инструкция STD), то он декрементируется (уменьшается). Декрементация или инкрементация выполняется на 1 при сравнении байт, на 2 при сравнении слов, или на 4 при сравнении двойных слов.

Инструкции SCASB, SCASW и SCASD являются синонимами инструкции SCAS для работы с байтами, словами и двойными словами. Эти инструкции не требуют операндов. Писать их проще, но они не позволяют проверять тип или сегмент.

Инструкции SCAS может предшествовать префикс REP. Это позволяет выполнять поиск в блоке из CX байт или слов. Подробности этой операции можно найти в описании префикса REP.

SETcc Установка байта по условию (только для процессоров 386 и i486).

O D I T S Z A P C

Код операции	Инструкция	Такты		Описание
		486	386	
0F 97	SETA r/m8	4/3	4/5	Установка байта, если выше (CF=0 и ZF=0).
0F 93	SETAE r/m8	4/3	4/5	Установка байта, если выше или равно CF=0).
0F 92	SETB r/m8	4/3	4/5	Установка байта, если ниже (CF=0).
0F 96	SETBE r/m8	4/3	4/5	Установка байта, если ниже или равно (CF=1 или ZF=1).
0F 92	SETC r/m8	4/3	4/5	Установка байта, если перенос (CF=1).
0F 92	SETC r/m8	4/3	4/5	Установка байта, если перенос (CF=1).
0F 94	SETE r/m8	4/3	4/5	Установка байта, если равно (CF=1).
0F 9F	SETG r/m8	4/3	4/5	Установка байта, если больше (CF=0 или SF=OF).
0F 9D	SETGE r/m8	4/3	4/5	Установка байта, если больше или равно (SF=OF).
0F 9C	SETL r/m8	4/3	4/5	Установка байта, если меньше (SF=OF).
0F 9E	SETLE r/m8	4/3	4/5	Установка байта, если меньше или равно (ZF=1 и SF<>OF).
0F 96	SETNA r/m8	4/3	4/5	Установка байта, если не выше (CF=

				1).
0F 92	SETNAE r/m8	4/3	4/5	Установка байта, если не выше или равно (CF=1).
0F 93	SETNB r/m8	4/3	4/5	Установка байта, если не ниже (CF=0).
0F 97	SETNBE r/m8	4/3	4/5	Установка байта, если не ниже или равно (CF=0 и ZF=0).
0F 93	SETNC r/m8	4/3	4/5	Установка байта, если нет переноса (CF=0).
0F 95	SETNE r/m8	4/3	4/5	Установка байта, если не равно (ZF=0).
0F 9E	SETNG r/m8	4/3	4/5	Установка байта, если не больше (ZF=1 или SF<>OF).
0F 9C	SETNGE r/m8	4/3	4/5	Установка байта, если не больше или равно (SF<>OF).
0F 9D	SETNL r/m8	4/3	4/5	Установка байта, если не меньше (SF=OF).
0F 9F	SETNLE r/m8	4/3	4/5	Установка байта, если не меньше или равно (ZF=1 и SF<>OF).
0F 91	SETNO r/m8	4/3	4/5	Установка байта, если нет переполнения (OF=0).
0F 9B	SETNP r/m8	4/3	4/5	Установка байта, если нет четности (PF=0).
0F 99	SETNS r/m8	4/3	4/5	Установка байта, если нет знака (SF=0).
0F 95	SETNZ r/m8	4/3	4/5	Установка байта, если ноль (ZF=0).

0F 90	SETNO r/m8	4/3	4/5	Установка байта, если нет переполнения (OF=1).
0F 9A	SETP r/m8	4/3	4/5	Установка байта при флаге четности (PF=1).
0F 9A	SETPE r/m8	4/3	4/5	Установка байта при четности (PF=1).
0F 9B	SETPO r/m8	4/3	4/5	Установка байта при нечетности (PF=0).
0F 98	SETS r/m8	4/3	4/5	Установка байта по знаку (SF=0).
0F 94	SETZ r/m8	4/3	4/5	Установка байта по нулю (ZF=0).

Инструкция SETcc записывает в приемник, заданный действующим адресом, или в регистр байт, содержащий 1, если условие удовлетворяется, и 0, если условие не удовлетворяется.

SGDT Сохранение таблицы глобальных дескрипторов/таблицы
SIDT дескрипторов прерываний (только для защищенного режима процессоров 80186/286/386/486).

O D I T S Z A P C

Код операции	Инструкция	Такты			Описание
		486	386	286	
0F 01 /0	SGDT m	10	9	11	Запись таблицы GDTR (глобальных дескрипторов) в m.
0F 01 /1	SIDT m	10	9	11	Запись таблицы IDTR (дескрипторов прерываний) в m.

Инструкции SGDT/SIDT копируют содержимое регистра таблицы дескрипторов в шесть байт памяти, указанных операндом. Поле LIMIT регистра присваивается первому слову действующего адреса. Если атрибут размера операнда равен 32 битам, то следующим трем байтам

присваивается поле BASE (база) регистра, а в четвертый байт записывается ноль. Последний байт не определен. Если атрибут размера операнда равен 16, то следующим четырем байтам присваивается 32-битовое поле BASE регистра.

Инструкции SGDT и SIDT используются только в системном программном обеспечении. В прикладных программах они не применяются.

SHLD Сдвиг влево с двойной точностью (только для процессоров 386 и i486).

O	D	I	T	S	Z	A	P	C
?				*	*	?	*	*

Код операции	Инструкция	Такты		Описание
		486	386	
0F A4	SHLD r/m16,r16, imm8	2/3	3/7	В r/m16 записывается конкатенация SHL r/m16 и r16.
0F A4	SHLD r/m32,r32, imm8	2/3	3/7	В r/m32 записывается конкатенация SHL r/m32 и r32.
0F A5	SHLD r/m16,r16, CL	2/3	3/7	В r/m16 записывается конкатенация SHL r/m16 и r16.
0F A5	SHLD r/m32,r32, CL	2/3	3/7	В r/m32 записывается конкатенация SHL r/m32 и r32.

Инструкция SHLD выполняет сдвиг первого операнда, задаваемого полем r/m, влево на число бит, заданное операндом-счетчиком. Второй операнд задает биты, сдвигаемые справа (начиная с бита 0). Результат записывается обратно в операнд r/m. Регистр остается неизмененным.

Операнд-счетчик задается либо в виде непосредственного байта, либо в виде регистра CL. Чтобы получить величину сдвига (число от 0 до 31), эти операнды берутся по модулю 32. Поскольку величина сдвига задается указанными регистрами, эта операция полезна для выполнения сдвигов с повышенной точностью (64 бита или более). Биты SF, ZF и PF устанавливаются в соответствии со значением результата. Регистр CF устанавливается в значение последнего сдвигаемого вовне бита. Флаги OF и AF остаются неопределенными.

SHRD Сдвиг вправо с двойной точностью
(только для процессоров 386 и i486).

O	D	I	T	S	Z	A	P	C
?				*	*	?	*	*

Код операции	Инструкция	Такты		Описание
		486	386	
0F A4	SHRD r/m16,r16, imm8	2/3	3/7	В r/m16 записывается конкатенация SHR r/m16 и r16.
0F A4	SHRD r/m32,r32, imm8	2/3	3/7	В r/m32 записывается конкатенация SHR r/m32 и r32.
0F A5	SHRD r/m16,r16, CL	2/3	3/7	В r/m16 записывается конкатенация SHR r/m16 и r16.
0F A5	SHRD r/m32,r32, CL	2/3	3/7	В r/m32 записывается конкатенация SHR r/m32 и r32.

Инструкция SHRD выполняет сдвиг первого операнда, задаваемого полем r/m, вправо на число бит, заданное операндом-счетчиком. Второй операнд задает биты, сдвигаемые справа (начиная с бита 0). Результат записывается обратно в операнд r/m. Регистр остается неизмененным.

Операнд-счетчик задается либо в виде непосредственного байта, либо в виде регистра CL. Чтобы получить величину сдвига (число от 0 до 31) эти операнды берутся по модулю 32. Поскольку величина сдвига задается указанными регистрами, эта операция полезна для выполнения сдвигов с повышенной точностью (64 бита или более). Биты SF, ZF и PF устанавливаются в соответствии со значением результата. Флаг CF устанавливается в значение последнего сдвигаемого вовне бита. Флаги OF и AF остаются неопределенными.

SLDT Запись таблицы локальных дескрипторов (только для защищенного режима процессоров 80186/286/386/486).

O D I T S Z A P C

Код операции	Инструкция	Такты			Описание
		486	386	286	
0F 00 /0	SLDT r/m16	2/3	pm-=2/2	2/3	Запись таблицы LDTR (локальных дескрипторов) в слово EA.

Инструкция SLDT записывает регистр таблицы локальных дескрипторов (LDTR) в двухбайтовый регистр или ячейку памяти, заданную операндом действующего адреса. Регистр представляет собой селектор, который указывает на таблицу глобальных дескрипторов.

Инструкция SLDT используется только в системном программном обеспечении. В прикладных программах она не применяется.

SMSW Запись слова состояния машины (только для защищенного режима процессоров 80186/286/386/486).

O D I T S Z A P C

Код операции	Инструкция	Такты			Описание
		486	386	286	
0F 00 /0	SMSW r/m16	2/3	2/3, pm=2/2	2/3	Запись слова состояния машины с слово EA.

Инструкция CMSW записывает слово состояния машины (часть CR0) в двухбайтовом регистре или ячейке памяти, на которую указывает операнд действующего адреса.

STC Установка флага переноса.

O	D	I	T	S	Z	A	P	C
								1

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
F9	STC	2	2	2	2	Установка флага переноса.

Инструкция STC устанавливает флаг переноса в 1.

STD Установка флага направления.

O	D	I	T	S	Z	A	P	C
	1							

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
FD	STD	2	2	2	2	Установка флага направления, после чего будет выполняться декрементация регистров (E)SI или (E)DI.

Инструкция STD устанавливает флаг направления в 1, что вызывает во всех последующих строковых операциях декрементацию (уменьшение) индексных регистров (E)SI или (E)DI.

STI Установка флага разрешения прерывания.

O	D	I	T	S	Z	A	P	C
		1						

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
FB	STI	5	3	2	2	Установка флага разрешения прерывания.

Инструкция STI устанавливает флаг прерывания в значение 1. После этого ЦП при выполнении следующей инструкции отвечает на внешние прерывания (если при выполнении следующей инструкции прерывания останутся разрешенными). Если внешние прерывания запрещены, и вы записываете STI, RET (как в конце подпрограммы), то инструкцию RET допускается выполнять перед распознаванием внешних прерываний. Кроме того, если внешние прерывания запрещены, а вы пишете STI, CLI, то внешние прерывания не распознаются, так как при выполнении инструкции CLI флаг прерываний очищается.

STOS Запись строковых данных (инструкция STOSD - только для
STOSB процессоров 386 и i486).
STOSW
STOSD

O	D	I	T	S	Z	A	P	C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
AA	STOS m8	5	4	3	11	Запись AL в байте ES:[(E)DI].
AB	STOS m16	5	4	3	11	Запись AX в слове ES:[(E)DI].
AB	STOS m32	5	4			Запись EAX в байте ES:[(E)DI].
AA	STOSB	5	4	3	11	Запись AL в байте ES:[(E)DI].
AB	STOSW	5	4	3	11	Запись AX в слове ES:[(E)DI].
AB	STOSD	5	4			Запись EAX в байте ES:[(E)DI].

Инструкция STOS передает содержимое регистра AL, AX или EAX в байт или слово в памяти, которые задаются регистром-приемником относительно сегмента ES.

Для атрибута размера адреса, равного 16, регистром-приемником является регистр DI, а для атрибута размера адреса 32 - **EDI.**

Операнд-приемник должен быть адресуемым через регистр ES. Переопределение сегмента не допускается.

Адрес приемника определяется содержимым регистра-приемника, а не явным операндом инструкции STOS. Данный операнд используется только для определения возможности адресуемости через регистр ES и типа данных. Перед выполнением инструкции STOS загрузите в регистр-приемник корректное значение адреса.

После выполнения пересылки регистр DI автоматически обновляется. Если флаг направления равен 0 (была выполнена инструкция CLD), то регистр DI инкрементируется (увеличивается). Если флаг направления равен 1 (была выполнена инструкция STD), то он декрементируется (уменьшается). DI увеличивается или уменьшается на 1 при записи байта, на 2 при записи слова и на 4 при записи двойного слова.

Инструкции STOSB, STOSW и STOSD являются синонимами инструкции OUTS для работы с байтами, словами и двойными словами. Инструкции STOS может предшествовать префикс REP. Это позволяет заполнять блоки из CX байт или слов. Подробности этой операции можно найти в описании REP.

STR Запись регистра задачи (только для защищенного режима процессоров 80286/386/486).

O D I T S Z A P C

Код операции	Инструкция	Такты			Описание
		486	386	286	
0F 00/1	STR r/m16	2/3	pm= 23/27	2/3	Загрузка в регистр задачи слова EA.

Содержимое регистра задачи копируется в двухбайтовый регистр или ячейку памяти, указываемую операндом действующего адреса.

Инструкция STR используется только в системном программном обеспечении. В прикладных программах она не применяется.

SUB Целочисленное вычитание.

O	D	I	T	S	Z	A	P	C
*				*	*	*	*	*

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
04 ib	SUB AL,imm8	1	2	3	4	Вычитание непосредственного байта из AL.
05 iw	SUB EX,imm16	1	2	3	4	Вычитание непосредственного слова из AX
05 id	SUB EAX,imm32	1	2			Вычитание непосредственного двойного слова из EAX.
80 /0 ib	SUB r/m8,imm2	1/3	2/7	3/7	4/17+EA	Вычитание непосредственного байта из байта r/m.
81 /0 iw	SUB r/m16,imm16	1/3	2/7	3/7	4/17+EA	Вычитание непосредственного слова из слова r/m.
81 /0 id	SUB r/m16,imm8	1/3	2/7	3/7	4/17+EA	Вычитание непосредственного байта с расширением по знаку из слова r/m.
83 /0 ib	SUB r/m32,imm8	1/3	2/7			Вычитание непосредственного байта с расширением по знаку из двойного слова

						r/m.
00 /r	SUB r/m8,r8	1/3	2/7	2/7	3/16+EA	Вычитание байтового регистра из байта r/m.
01 /r	SUB r/m16,r16	1/3	2/7	2/7	3/16+EA	Вычитание регистра размером в слово из слова r/m.
01 /r	SUB r/m32,r32	1/3	2/7			Вычитание регистра размером в двойное слово из слова r/m.
02 /r	SUB r8,r/m8	1/2	2/6	2/7	3/9+EA	Вычитание байта r/m из регистра размером в байт.
03 /r	SUB r16,r/m16	1/2	2/6	2/7	3/9+EA	Вычитание байта r/m из регистра размером в слово.
03 /r	SUB r32,r/m32	1/2	2/6			Вычитание байта r/m из регистра размером в двойное слово.

Операция SUB выполняет целочисленное вычитание двух операндов (второго операнда, источника, из первого, приемника). Результат вычитания присваивается первому операнду. Соответствующим образом устанавливаются флаги.

Когда непосредственное байтовое значение вычитается из операнда размером в слово, то оно расширяется по знаку до размера операнда-приемника.

TEST Логическое сравнение.

O	D	I	T	S	Z	A	P	C
0				*	*	?	*	0

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
A8 ib	TEST AL,imm8	1	2	3	4	Операция И над непосредственным байтом и AL.
A9 iw	TEST AX,imm16	1	2	3	4	Операция И над непосредственным байтом и AX.
A9 id	TEST EAX, imm32	1	2			Операция И над непосредственным двойным словом и EAX.
F6 /0 ib	TEST r/m8, imm8	1/2	2/5	3/6	5/ 11+ EA	Операция И над непосредственным байтом и байтом r/m.
F7 /0 iw	TEST r/m16, imm16	1/2	2/5	3/6	5/ 11+ EA	Операция И над непосредственным словом и словом r/m.
F7 /0 id	TEST r/m32, imm32	1/2	2/5			Операция И над непосредственным двойным словом и двойным словом r/m.
84 /r	TEST r/m8, r8	1/2	2/5	2/6	3/9 +EA	Операция И над регистром размером в байт и байтом r/m.
85 /r	TEST r/m16, r16	1/2	2/5	2/6	3/9 +EA	Операция И над регистром размером в слово и словом r/m.
85 /r	TEST r/m32, r32	1/2	2/5			Операция И над регистром размером в двойное слово и двойным словом r/m.

Операция TEST вычисляет для двух операндом поразрядную логическую операцию И. Каждый бит результата будет равен 1, если оба соответствующих бита операндом равны 1. В противном случае бит равен 0. Результат операции отбрасывается. Модифицируются только флаги.

VERR Проверка сегмента для чтения или записи (только для
VERW защищенного режима процессоров 80286/386/486).

O	D	I	T	S	Z	A	P	C
					*			

Код операции	Инструкция	Такты			Описание
		486	386	286	
OF 00 /4	VERR r/m16	11/11	pm=10/11	14/16	Устанавливает ZF = 1, если сегмент доступен для чтения
OF 00 /5	VERW r/m16	11/11	pm=10/11	14/16	Устанавливает ZF = 1, если сегмент доступен для записи

Двухбайтовый регистр или операнд в памяти инструкции VERR или VERW содержит значение селектора. Инструкции VERR или VERW определяют, доступен ли сегмент, описанный селектором, из текущего привилегированного уровня, и можно ли для него выполнять чтение (VERR) или запись (VERW). Если сегмент доступен, то флаг нуля устанавливается в 1, а если сегмент недоступен, то флаг нуля устанавливается в 0. Чтобы установить флаг ZF, должны соблюдаться следующие условия:

- Селектор должен обозначать дескриптор в границах таблицы (GDT или LDT). Селектор должен быть "определен".

- Селектор должен описывать дескриптор сегмента кода или данных (а не быть дескриптором сегмента состояния задачи, LDT или вентиля).

- Для инструкции VERR сегмент должен быть доступен по чтению. Для инструкции VERW сегмент должен быть доступен по записи (сегмент данных).

- Если сегмент кода доступен по чтению и удовлетворяет условиям, уровень привилегий дескриптора (DPL) может иметь для инструкции VERR любое значение. В противном случае DPL должно быть больше или равно (иметь меньший или тот же

уровень привилегий), чем текущий уровень привилегий и RPL селектора.

Проверка выполняется так же, как если бы сегмент был загружен в регистры DS, ES, FS или GS и был выполнен указанный доступ (чтение или запись). Результат проверки определяет значение, которое получает флаг нуля. Значение селектора не может дать в результате исключительную ситуацию по нарушению защиты, что позволяет программному обеспечению предвидеть возможные проблемы с доступом.

WAIT Ожидание, пока разряд BUSY# будет неактивным.

O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
9B	WAIT	1-3	6	3	4+5n	Ожидание, пока разряд BUSY# будет неактивным.

Инструкция WAIT приостанавливает выполнение инструкций ЦП, пока разряд BUSI# не станет неактивным. Разряд BUSY# управляется расширением арифметического сопроцессора 80x87.

WBINVD Запись и запрещение кэш-буфера.

O D I T S Z A P C

Код операции	Инструкция	Такты	Описание
		486	
OF 09	WBINVD	5	Обратная запись и запрещение всего кэш -буфера.

Внутренний кэш-буфер сбрасывается, и выполняется специальный функциональный цикл шины, который указывает, что внешний кэш-буфер должен вывести свое содержимое обратно в оперативную память. Затем следует другой специальный функциональный цикл шины, указывающий на необходимость сброса всего кэш-буфера.

Примечание: Эта инструкция зависит от реализации. В будущих процессорах фирмы Intel ее функция может реализовываться по-разному. Ответственность за обратную запись содержимого внешнего кэш-буфера и индикацию сброса возлагается на аппаратные средства.

XADD Обмен и сложение (только для процессора i486).

O	D	I	T	S	Z	A	P	C
*				*	*	*	*	*

Код операции	Инструкция	Такты	Описание
		486	
OF C0/r	XADD r/m8,r8	3/4	Выполняет обмен содержимого регистра размером в байт и байта r/m. Загружает сумму в байт r/m.
OF C1/r	XADD r/m16,r16	3/4	Выполняет обмен содержимого регистра размером в слово и слов r/m. Загружает сумму в слово r/m.
OF C1/r	XADD r/m32,r32	3/4	Выполняет обмен содержимого регистра размером в двойное слово и двойного слова r/m. Загружает сумму в двойное слово r/m.

Инструкция XADD загружает приемник в источник, затем загружает сумму приемника и исходного значения источника в приемник.

Исключительные ситуации защищенного режима: #GP(0), если результат находится в сегменте, недоступном для записи; #GP(0) при всех недопустимых операндах действующего адреса в сегментах CS, DS, ES, FS или GS; #SS(0) при недопустимом адресе в сегменте SS; #PF (сбой) при страничном сбое; #NM, если в CR0 установлены EM или TS; #AC при невыровненной ссылке на память, если текущий уровень привилегий равен 3.

Прерывания режима реальной адресации: прерывание 13, если любая часть операнда лежит вне действующего пространства адресов (от 0 до 0FFFFh).

Прерывания виртуального режима 8086: те же исключительные ситуации, что и в режиме реальной адресации; те же исключительные ситуации #PF и #AC, что и в защищенном режиме.

XCHG Обмен содержимого памяти/регистра с регистром.

O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
86 /r	XCHG r/m8,r8	3/4	3/5	3/5	4/17+ EA	Выполняет обмен содержимого регистра размером в байт и байта EA.
86 /r	XCHG r8,r/m8	3/4	3/5	3/5	4/17+ EA	Выполняет обмен содержимого байта EA и регистра размером в байт.
87 /r	XCHG r/m16,r16	3/4	3/5	3/5	4/17+ EA	Выполняет обмен содержимого регистра размером в слово и слова EA.
87 /r	XCHG r/m32,r32	3/4	3/5			Выполняет обмен содержимого регистра размером в двойное слово и двойного слова EA.
87 /r	XCHG r32,r/m32	3/4	3/5			Выполняет обмен содержимого регистра размером в двойное слово и двойного слова EA.
90+r	XCHG AX,r16	3	3	3	3	Выполняет обмен содержимого регистра AX и слова.
90+r	XCHG r16,AX	3	3	3	3	Выполняет обмен содержимого регистра AX и слова.
90+r	XCHG EAX,r32	3	3			Выполняет обмен содержимого регистра EAX и двойного слова.

90+r	XCHG r32,EAX	3	3			Выполняет обмен содержимого регистра EAX и двойного слова.

Инструкция XCHG выполняет обмен содержимого двух операндов. Операнды могут следовать в любом порядке. Если используется операнд в памяти, то на время выполнения операции выдается BUS LOCK (блокировка шины), независимо от наличия или отсутствия префикса LOCK или значения IOPL.

XLAT Трансляция таблицы.
XLATB

O D I T S Z A P C

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
D7	XLAT m8	4	5	5	11	Устанавливает AL в байт памяти DS:[(E)BX+беззнаковое значение AL].
D7	XLATB	4	5	5	11	Устанавливает AL в байт памяти DS:[(E)BX+беззнаковое значение AL].

Инструкция XLAT изменяет значение регистра AL в соответствии со значением индекса и записью таблицы. AL должен представлять собой беззнаковый индекс с таблице, адресуемой через DS:BX (для атрибута размера адреса 16 бит) или через DS:EBX (для атрибута размера адреса 32 бита).

Операнд инструкции XLAT допускает переопределение сегмента. Инструкция XLAT использует содержимое регистра BX, если оно отличается от смещения операнда. Смещение операнда должно помещаться в BX/EBX предыдущей инструкцией.

Если таблица BX/EBX всегда будет находиться в сегменте DS, то можно использовать форму инструкции без операнда XLATB.

XOR Логическая операция "исключающее ИЛИ"

O	D	I	T	S	Z	A	P	C
0				*	*	?	*	0

Код операции	Инструкция	Такты				Описание
		486	386	286	86	
34 ib	XOR AL,imm8	1	2	3	4	Операция "исключающее ИЛИ" над непосредственным байтом и AL.
35 iw	XOR AX,imm16	1	2	3	4	Операция "исключающее ИЛИ" над непосредственным словом и AX.
80 iw	XOR EAX,imm32	1	2			Операция "исключающее ИЛИ" над непосредственным двойным словом и EAX.
80 /6 ib	XOR r/m8,imm8	1/3	2/7	3/7	7/17 EA	Операция "исключающее ИЛИ" над непосредственным байтом и байтом r/m.
81 /6 iw	XOR r/m16, imm16	1/3	2/7	3/7	7/17 EA	Операция "исключающее ИЛИ" над непосредственным словом и словом r/m.
81 /6 id	XOR r/m32, imm32	1/3	2/7			Операция "исключающее ИЛИ" над непосредственным двойным словом и двойным словом r/m.
83 /6 ib	XOR r/m16, imm8	1/3	2/7			Операция "исключающее ИЛИ" над непосредственным расширяемым по знаку словом и словом r/m.
83 /6 ib	XOR r/m32, imm8	1/3	2/7			Операция "исключающее ИЛИ" над непосредственным расширяемым по знаку словом и двойным словом r/m.

30 /r	XOR r/m8,r8	1/3	2/6	2/7	3/16 EA	Операция "исключающее ИЛИ" над регистром размером в байт и байтом r/m.
31 /r	XOR r/m16,r16	1/3	2/6	2/7	3/16 EA	Операция "исключающее ИЛИ" над регистром размером в слово и словом r/m.
31 /r	XOR r/m32,r32	1/3	2/6			Операция "исключающее ИЛИ" над регистром размером в двойное слово и двойным словом r/m.
32 /r	XOR r8,r/m8	1/2	2/7	2/7	3/9+ EA	Операция "исключающее ИЛИ" над регистром размером в байт и байтом r/m.
33 /r	XOR r16,r/m16	1/2	2/7	2/7	3/9+ EA	Операция "исключающее ИЛИ" над регистром размером в слово и словом r/m.
33 /r	XOR r32,r/m32	1/2	2/7			Операция "исключающее ИЛИ" над регистром размером в двойное слово и двойным словом r/m.

Операция XOR вычисляет операцию "исключающее ИЛИ" над двумя операндами. Каждый бит результата равен 1, если соответствующие биты операндов различны, и нулю, если соответствующие биты совпадают. Результат замещает первый операнд.

Часть 5. Инструкции сопроцессора

В этой части в алфавитном порядке описываются инструкции сопроцессора 80x87.

Запись для каждой комбинации типов операндов кодируется с помощью мнемоники.В приведенной ниже таблице поясняются идентификаторы операндов, используемые в данном руководстве.

Идентификатор	Пояснение
ST	Вершина стека; регистр, находящийся в данный момент в вершине стека.
ST(1)	Регистр в стеке с номером i (i<=0<=7), считая от вершины (i-й элемент). ST(1) - это следующий элемент в стеке, ST(2) находится ниже ST(2) и т.д.
Короткое вещественное	Короткое вещественное (32 бита) число в памяти.
Длинное вещественное	Длинное вещественное (64 бита) число в памяти.
Временное вещественное	Временное вещественное (80 бит) число памяти.
Упакованное десятичное	Упакованное десятичное целое (18 цифр, 10 байт) в памяти.
Целое размером в слово	Двоичное целое размером в слово (16 бит).
Короткое целое	Двоичное короткое целое (32 бита) в памяти.
Длинное целое	Двоичное длинное целое (64 бита) в памяти.
nn байт	Область памяти размером nn байт.

Приведем перечень возможных исключительных ситуаций, которые могут возникать в инструкциях:

- IS = недопустимый операнд из-за переполнения/потери значимости стека.
- I = операнд, недопустимый по другой причине.
- D = ненормализованный операнд.
- Z = деление на ноль.
- O = переполнение.
- U = потеря значимости.
- P = неточный результат (точность).

F2XM1 Вычисление 2^x - 1.

Исключительные ситуации: P, U, D, I, S.

F2XM1 (без операндов)

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Нет операндов		211-476	211-476	242(140-279)	2	F2XM1

FABS Абсолютное значение.

Исключительные ситуации: I.

FABS (без операндов)

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Нет операндов	10-11	10-17	22	3	2	F2XM1

FADD Целочисленное сложение.

Исключительные ситуации: I, D, O, U, P.

FADD //источник/приемник, источник

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
//ST,ST(i)/ ST(i),ST	70-100	70-100	23-34	10(8-20)	2	FADD ST,ST(4)
Короткое целое	90-120 +EA	90-120	24-32	10(8-20)	2-4	FADD AIR_TP[SI]
Длинное целое	95-125 +EA	95-125	29-37	10(8-20)	2-4	FADD [BX],MEAN

FADDP Сложение вещественных чисел и извлечение из стека.

Исключительные ситуации: I, D, O, U, P.

FADDP приемник, источник

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
ST,ST(i)	75-105	75-105	23-34	10(8-20)	2	FADDP ST(2),ST

FBLD Упакованная десятичная загрузка (BCD).

Исключительные ситуации: I.

FADDP источник

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
Упакованный десятичный	290-310	290-310	5	75(70-103)	2-4	FBLD YDT_SAL

FBSTP Запись упакованного десятичного значения (BCD) и извлечение из стека.

Исключительные ситуации: I.

FBSTP источник

Операнды	Такты выполнения			Байты кода	Пример
	87	287	387	486	
Упакованный десятичный	520-540 +EA	512-534 +EA	512-534	175(172-176)	2-4 FBSTR [BX],CAST

FCHS Изменение знака.

Исключительные ситуации: I.

FCHS (нет операндов)

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Нет операндов	10-17	10-17	24-25	6	2	FCHS

FCLEX Очистка исключительных прерываний.
FNCLEX

Исключительные ситуации: нет.

FCLEX/FNCLEX (нет операндов)

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Нет операндов	2-8	2-8	11	7		FNCLEX

FCOM Сравнение вещественных чисел.

Исключительные ситуации: I, D.

FCOM//источник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
//ST(i)	40-50	40-50	24	4		FCOM ST(1)
Короткий вещественный	60-70 +EA	60-70	26	4	2-4	FCOM [BP], UPPER_LIMIT
Длинный вещественный	65-75 +EA	65-75	31	4	2-4	FCOM LENGTH1

FCOMP Сравнение вещественных чисел и извлечение из стека.

Исключительные ситуации: I, D.

FCOMP//источник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
//ST(i)	42-52	45-52	26	4	2	FCOM ST(2)
Короткий вещественный	63-73 +EA	63-73	26	4	2-4	FCOM [BP+2], LIMIT_1
Длинный вещественный	67-77 +EA	67-77	31	4	2-4	FCOMP DENSITY

FCOMPP Сравнение вещественных чисел и извлечение из стека дважды.

Исключительные ситуации: I, D.

FCOMPP (нет операндов)

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Нет операндов	45-55	45-55	26	5	2	FCOMPP

FCOS Косинус ST(0) (только для процессоров 387 и i486).

Исключительные ситуации: IS, I, D, U, P.

FCOS

Операнды	Такты выполнения			Байты кода	Пример
	87 287	387	486		
Нет операндов		123-772*	241(193-279)	2	FCOS

* Эти временные значения соблюдаются в диапазоне /x/ /4. Для операндов, не лежащих в данном диапазоне, для уменьшение операндов может потребоваться до 76 дополнительных тактов.

FDECSTP Уменьшение указателя стека.

Исключительные ситуации: нет.

FDECSTP (нет операндов)

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Нет операндов	6-12	6-12	22	3	2	FDECSTP

FDISI Запрещение прерывания (только для сопроцессора 8087).
FNDISI

Исключительные ситуации: нет.

FDISI (нет операндов)

Операнды	Такты выполнения: Типичное значение	Диапазон	Слово операнда	Байты кода	Пример
Нет операндов	5	2-8	0	2	FDISI

FDIV Деление вещественных чисел.

Исключительные ситуации: I, D, Z, O, U, P.

FDIV //источник/приемник, источник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
//ST(i),ST	193-**203**	193-**203**	88-**91**	73	2	FDIV
Короткое вещественное	215-**225**	215-**225**	89	73	2	FDIV DISTANCE
Длинное вещественное //ST,S**T(i)**	220-**230**	220-**230**	94	73	2-**4**	FDIV ARC[DI]

FDIVP Деление вещественных чисел и извлечение из стека.

Исключительные ситуации: I, D, Z, O, U, P.

FDIVP приемник, источник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
//ST(i),ST	197-**207**	198-**209**	88-**91**	73	2	FDIVP ST(4),ST

FDIVR Деление вещественных чисел с обращением.

Исключительные ситуации: I, D, Z, O, U, P.

FDIVP //источник/приемник, источник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
//ST,ST(i) ST(i),ST	194-**204**	198-**207**	88-**91**	73	2	FDIVR ST(2),ST
Короткое целое	216-**226** +EA	215-**225**	89	73	2-**4**	FDIVR [BX],P_R
Длинное целое	221-**231** +EA	220-**230**	94	73	2-**4**	FDIVR REC,FREC

FDIVRP Деление вещественных чисел с обращением и извлечение из стека.

Исключительные ситуации: I, D, Z, O, U, P.

FDIVP приемник, источник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
ST(i),ST	198-**208**	198-**208**	88-**91**	73	2	FDIVPR ST(1),ST

FENI Разрешение прерываний (только для сопроцессора 8087).
FNENI

Исключительные ситуации: нет.

FDIVP (нет операндов)

Операнды	Такты выполнения	Байты кода	Пример
	87		
(нет операндов)	5(2-8)	2	FNENI

FFREE Освобождение регистра.

Исключительные ситуации: нет.

FFREE приемник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
ST(i)	9-16	9-16	18	3	2	FFREE ST(1)

FIADD Целочисленное сложение.

Исключительные ситуации: I, D, O, P.

FFREE источник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	86		
Целое размером в слово	102-137 +EA	102-137	71-85	2.5(19-2)	2-4	FIADD ST_TR
Короткое целое	108-143 +EA	108-143	57-72	4(20-35)	2-4	FIADD N[SI]

FICOM Целочисленное сравнение.

Исключительные ситуации: I, D.

FICOM источник

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
Целое размером в слово	72-86 +EA	72-86	71-75	18(16-20)	2-4	FICOM T.PASS
Короткое целое	108-143 +EA	108-143	57-72	24(20-35)	2-4	FICOM [BP+ 4].PARM_CNT

FICOMP Целочисленное сравнение и извлечение из стека.

Исключительные ситуации: I, D.

FICOMP источник

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
Целое размером слово	74-88 +EA	74-88	71-75	18(16-20)	2-4	FICOMP T.PASS
Короткое целое	80-93 +EA	80-93	56-63	24(20-35)	2-4	FICOMP N_SAMPL

FIDIV Деление целых чисел.

Исключительные ситуации: I, D, Z, O, U, P.

FIDIV источник

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
Целое размером в слово	224-**238** +EA	224-**238**	136-**140**	73	2-**4**	FIDIV SURV.OBS
Короткое целое	230-**243** +EA	230-**243**	120-**127**	73	2-**4**	FIDIV REL_AN[DI]

FIDIVR Деление целых чисел с обращением.

Исключительные ситуации: I, D, Z, O, U, P.

FIDIVR источник

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
Целое размером в слово	225-**239** +EA	224-**238**	135-**141**	73	2-**4**	FIDIV [BP].X_COD
Короткое целое	231-**245** +EA	230-**243**	121-**128**	73	2-**4**	FIDIV FREQUENCY

FILD Загрузка целого.

Исключительные ситуации: I.

FILD источник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Целое размером в слово	42-54 +EA	46-54	61-65	11.5(9-12)	2-4	FILD [BX].SEQ
Короткое целое	52-60 +EA	52-60	45-52	14.5(13-16)	2-4	FILD STNDOFF[DI]
Длинное целое	60-68 +EA	60-68	56-67	16.8(10-18)	2-4	FILD RESP.COUNT

FIMUL Целочисленное умножение.

Исключительные ситуации: I, D, O, P.

FIMUL источник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Целое размером в слово	124-138 +EA	124-138	76-87	8	2-4	FIMUL BEARING
Короткое целое	130-144 +EA	130-144	61-82	8	2-4	FIMUL POS.Z_R

FINCSTP Увеличение указателя стека.

Исключительные ситуации: нет.

FINCSTP (нет операндов)

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Нет операндов	6-12	6-12	21	3	2	FINCSTP

FINIT Инициализация процессора.
FNINIT

Исключительные ситуации: нет.

FINIT/FNNINIT (нет операндов)

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Нет операндов	2-8	2-8	33	17	2	FINIT

FIST Запись целого значения.

Исключительные ситуации: I, P.

FIST приемник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Целое значение размером в слово	80-90 +EA	80-95	82-95	33.4 (29-34)	2-4	FIST OBS.COUNT
Короткое целое	82-92 +EA	82-92	79-93	32.4 (28-34)	2-4	FIST [BP].F_PU

FISTP Запись целого значения и извлечение из стека.

Исключительные ситуации: I, P.

FISTP приемник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Целое значение размером в слово	82-92 +EA	82-92	82-95	33.4 (29-34)	2-4	FISTP [BX]. ALPHA_COUNT[SI]
Короткое целое	84-94 +EA	84-94	79-93	32.4 (29-34)	2-4	FISTP CORR_TIME
Длинное целое	94-105+ EA	94-105	80-97	32.4 (29-34)	2-4	FISTP PANEL.N_R

FISUB Целочисленное вычитание.

Исключительные ситуации: I, D, O, P.

FISUB приемник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Целое значение размером в слово	102-137 +EA	102-137	71-83	22.5 (19-32	2-4	FISUB BASE_FREQ
Короткое целое	108-143 +EA	108-143	57-82	24 20-35)	2-4	FISUB TR_SZ[DI]

FISUBR Целочисленное вычитание с обращением.

Исключительные ситуации: I, D, O, P.

FISUBR источник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Целое значение размером в слово	103-139 +EA	102-137	72-84	22.5 19-32	2-4	FISUBR F[BX][SI]
Короткое целое	109-144 +EA	108-143	58-83	24 20-35	2-4	FISUBR BALANCE

FLD Загрузка вещественного значения.

Исключительные ситуации: I, D.

FLD источник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
ST(i)	17-22	17-22	14	4	2	FLD ST(0)
Короткое целое	38-56 +EA	38-56	20	3	2-4	FLD READ[SI].PRS
Длинное целое	40-60 +EA	40-60	25	3	2-4	FLD [BP].TEMPR
Временное целое	53-60 +EA	53-65	44	6	2-4	FLD SAVEREAD

FLDCTW Загрузка слова управления.

Исключительные ситуации: нет.

FLDCW источник

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
2 байта	7-14+EA	7-14	19	4	2-4	FLDCW CONTROL_W

FLDENV Загрузка операционной среды.

Исключительные ситуации: нет.

FLDENV источник

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
14 байт	35-45+EA	35-45	71	44 реальный или виртуальный 34 защищенный	2-4	FLDENV [BP+6]

FLDLG2 Загрузка log10 2.

Исключительные ситуации: I.

FLDLG2 (нет операндов)

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
Нет операндов	18-24	18-24	41	8	2	FLDLG2

FLDLN2 Загрузка ln 2.

Исключительные ситуации: I.

FLDLN2 (нет операндов)

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
Нет операндов	17-23	17-23	41	8	2	FLDLN2

FLDL2E Загрузка log2 e.

Исключительные ситуации: I.

FLDL2E (нет операндов)

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
Нет операндов	15-21	15-21	40	8	2	FLDL2E

FLDL2T Загрузка log2 10.

Исключительные ситуации: I.

FLDL2T (нет операндов)

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
Нет операндов	16-22	16-22	40	8	2	FLDL2T

FLDPI Загрузка числа **Pi.**

Исключительные ситуации: I.

FLDPI (нет операндов)

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
Нет операндов	16-**22**	16-**22**	40	8	2	FLDPI

FLDZ Загрузка +0.0.

Исключительные ситуации: I.

FLDZ (нет операндов)

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
Нет операндов	11-**27**	11-**17**	20	4	2	FLDZ

FLD1 Загрузка +1.0.

Исключительные ситуации: I.

FLD1 (нет операндов)

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
Нет операндов	15-**21**	15-**21**	24	4	2	FLD1

FMUL Умножение вещественных чисел.

Исключительные ситуации: I, D, O, U, P.

FMUL //источник/приемник, источник

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
//ST(i),ST/ST, 90-105,ST(1)*	90-105	90-145	29-57	16	2	FMUL ST,ST(3)
//ST(i),ST/ST, ST,ST(1)	130-145	90-145	29-57	16	2	FMUL ST,ST(3)
Короткое вещественное	110-125 +EA	110-125	27-35	11	2-4	FMUL SPEED_FACT
Длинное вещественное*	112-126 +EA	112-168	32-57		2-4	FMUL [BP].HEIG
Длинное вещественное*	154-168 +EA	112-168	32-57		2-4	FMUL [BP].HEIG

* - когда один из операндов "короткий" (в дробной части он содержит 40 завершающих нулей, например, он был загружен из короткого вещественного операнда в памяти).

FMULP Умножение вещественных чисел и извлечение из стека.

Исключительные ситуации: I, D, O, U, P.

FMULP приемник, источник

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
ST(i),ST*	94-108	198-208	29-57		2	FMULP ST(1),ST
ST(i),ST	134-148	198-208	29-57	16	2	FMULP ST(1),ST

* - когда один из операндов "короткий" (в дробной части он содержит 40 завершающих нулей, например, он был загружен из короткого вещественного операнда в памяти).

FLNOP Нет операции.

Исключительные ситуации: нет.

FNOP нет операндов)

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
Нет операндов	10-16	10-16	12	3	2	FNOP

FPATAN Дробный арктангенс.

Исключительные ситуации: U, P (операнды не проверяются).

FPATAN (нет операндов)

Операнды	Такты выполнения			Байты кода	Пример
	87	287	387	486	
Нет операндов	250-800	250-800	314-487	5(2-17) 2	FPATAN

FPREM Дробный остаток.

Исключительные ситуации: I, D, U.

FPREM (нет операндов)

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
Нет операндов	15-190	15-190	74-155	2(2-8)	2	FPREM

FPREM1 Дробный остаток (только для процессоров 387 и i487).

Исключительные ситуации: I, D, U.

FPREM1 (нет операндов)

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Нет операндов			95-185	94.5(72-167)	2	FPREM1

FPTAN Дробный тангенс.

Исключительные ситуации: U, P (операнды не проверяются).

FPTAN (нет операндов)

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Нет операндов	30-540	30-540	191-573	244 (200-273)	2	FPTAN

FRNDINT Округление до целого.

Исключительные ситуации: I, P.

FRNDINT (нет операндов)

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
Нет операндов	16-**50**	16-**50**	66-**80**	29.1 (21-**30**)	2	FRNDINT

FRSTOR Восстановление сохраненного состояния.

Исключительные ситуации: нет.

FRSTOR источник

Операнды	Такты выполнения			Байты кода	Пример
	87	287	387	486	
94 байта	197-**207** +EA	205-**215**	308	131 2-**4** реальный или виртуальный 240 защищенный	FRSTOR[BP]

Примечание: Значение времени выполнения для данной инструкции не имеет смысла при определении полного времени выполнения. В типичном случае (процессор 80286, сопроцессор 80287) выполнение в сопроцессоре 80287 происходит параллельно с передачей операндов. Общее время выполнения инструкции определяется передачей операндов. Для процессоров 80286:80287 соотношение равно 4:8, 1:1 и 8:5, а общее время выполнения данной инструкции оценивается значениями 490, 302 и 227 тактов сопроцессора 80287 соответственно.

FSAVE Сохранение состояния.
FNSAVE

Исключительные ситуации: нет.

FSAVE/FNSAVE приемник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
94 байта	197-207 +EA	205-215	375-376		2-4	FSAVE [BP]

Примечание: Значение времени выполнения для данной инструкции не имеет смысла при определении полного времени выполнения. В типичном случае (процессор 80286, сопроцессор 80287) выполнение в сопроцессоре 80287 происходит параллельно с передачей операндов. Общее время выполнения инструкции определяется передачей операндов. Для процессоров 80286:80287 соотношение равно 4:8, 1:1 и 8:5, а общее время выполнения данной инструкции оценивается значениями 490, 302 и 227 тактов сопроцессора 80287 соответственно.

FSCALE Масштабирование.

Исключительные ситуации: I, O, U.

FSCALE (нет операндов)

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Нет операндов	32-38	32-38	67-86	31 (30-32)	2	FSCALE

FSETPM Установка защищенного режима.

Исключительные ситуации: нет.

FSETPM (нет операндов)

Операнды	Такты выполнения	Байты кода	Пример
	287		
Нет операндов	2-8		FSETPM

FSIN Синус ST(0) (только для защищенного режима процессоров 386 и i486).

Исключительные ситуации: IS, I, D, U, P.

FSIN

Операнды	Такты выполнения		Байты кода	Пример
	387	486		
Нет операндов	122-771*	241(193-279)		FSIN

* Эти временные значения соблюдаются в диапазоне /x/ /4. Для операндов, не лежащих в данном диапазоне, для уменьшение операндов может потребоваться до 76 дополнительных тактов.

FSINCOS Синус и косинус ST(0) (только для защищенного режима процессоров 386 и i486).

Исключительные ситуации: IS, I, D, U, P.

FSINCOS

Операнды	Такты выполнения		Байты кода	Пример
	387	486		
Нет операндов	194-809*	241(243-329)		FSINCOS

* Эти временные значения соблюдаются в диапазоне /x/ /4. Для операндов, не лежащих в данном диапазоне, для уменьшение операндов может потребоваться до 76 дополнительных тактов.

FSQRT Квадратный корень.

Исключительные ситуации: I, D, P.

FSQRT (нет операндов)

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Нет операндов	180-186	180-186	122-129	85.6(83-87)	2	FSQRT

FST Запись вещественного значения.

Исключительные ситуации: I, O, D, P.

FST приемник

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
ST(i)	15-22	15-22	11	3	2	FST ST(3)
Короткое вещественное	84-90 +EA	84-90	44	7	2-4	FST CORR[DI]
Длинное вещественное	96-104 +EA	84-90	45	8	2-4	FST MEAN_RD

FSTCW Запись слова управления.
FNSTCW

Исключительные ситуации: нет.

FSTCW приемник

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
2 байта	12-18 +EA	12-18	15		2-4	FSTCW SV_CON

FSTENV Сохранение операционной среды.
FNSENV

Исключительные ситуации: нет.

FSTENV приемник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
14 байт	40-**50** +EA	40-**50**	103-**104**		2-**4**	FSTENV[BP]

FSTP Сохранение вещественного значения и извлечение из стека

Исключительные ситуации: I, O, U, P.

FSTP приемник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
ST(i)	17-**24**	17-**24**	12	3	2	FSTCW ST(2)
Короткое вещественное	86-**92** +EA	86-**92**	44	7	2-**4**	FSTCW [BX].AD
Длинное вещественное	98-**106** +EA	96-**106**	45	8	2-**4**	FSTCW TOT_DOS
Временное вещественное	52-**58** +EA	52-**58**	53	8	2-**4**	FSTCW REG_SAV

FSTSW Запись слова состояния.
FNSTSW

Исключительные ситуации: нет.

FSTSW/FNSTW приемник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
2 байта	12-18 +EA	12-18	15		2-4	FSTSW SAVE_ST

FSTSW AX Запись слова состояния в регистр AX.
FNSTSW AX

Исключительные ситуации: нет.

FSTSW приемник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
AX		10-16	13	3	2	FSTSW AX

FSUB Вычитание вещественных значений.

Исключительные ситуации: I, O, U, P.

FSUB //источник/приемник, источник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
//ST,ST(i)/ ST(i),ST	70-100	70-100	26-37	7(5-17)	2	FSUB ST,ST(2)
Короткое вещественное	90-120 +EA	90-120	24-32	7(5-17)	2-4	FSUB BASE_VAL
Длинное вещественное	95-125 +EA	95-125	28-36	7(5-17)	2-4	FSUB COORD.X

FSUBP Вычитание вещественных значений и извлечение из стека.

Исключительные ситуации: I, O, U, P.

FSUBP приемник, источник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
ST(i),ST	75-105	75-105	26-37	7(5-17)	2	FSUB ST,ST(2)

FSUBR Вычитание вещественных значений с обращением.

Исключительные ситуации: I, O, U, P.

FSUB //источник/приемник, источник

Операнды	Такты выполнения			Байты кода		Пример
	87	287	387	486		
//ST,ST(i)/ ST(i),ST	70-100	70-100	26-37	7(5-17)	2	FSUBR ST,ST(1)
Короткое вещественное	90-120 +EA	90-120	24-32	7(5-17)	2-4	FSUBR VECT[SI]
Длинное вещественное	95-125 +EA	95-125	28-36	7(5-17)	2-4	FSUBR [BX].IND

FSUBRP Вычитание вещественных значений с обращением и извлечение из стека.

Исключительные ситуации: I, O, U, P.

FSUBRP приемник, источник

Операнды	Такты выполнения			Байты кода	Пример	
	87	287	387	486		
ST(i),ST	75-**105**	75-**105**	26-**37**	7(5-**17**)	2	FSUBRP ST,ST(2)

FTST Проверка вершины стека на +0.0

Исключительные ситуации: I, D.

FTST (нет операндов)

Операнды	Такты выполнения			Байты кода	Пример	
	87	287	387	486		
Нет операндов	38-**48**	38-**48**	28	4	2	FTST

FUCOM Неупорядоченное сравнение.

Исключительные ситуации: IS, I, D.

Операнды	Такты выполнения		Байты кода	Пример
	387	486		
//ST(i)	24	4	2	FUCOM ST(1)

FUCOMP Неупорядоченное сравнение (только для сопроцессоров 387 и i486).

Исключительные ситуации: IS, I, D.

Операнды	Такты выполнения		Байты кода	Пример
	387	486		
//ST(i)	26	4	2	FUCOMP ST(2)

FUCOMPP Неупорядоченное сравнение (только для сопроцессоров 387 и i486).

Исключительные ситуации: IS, I, D.

Операнды	Такты выполнения		Байты кода	Пример
	387	486		
Нет операндов	26	4	2	FUCOMPP

FWAIT Ожидание.

Исключительные ситуации: нет (инструкция ЦП).

FWAIT (нет операндов)

Операнды	Такты выполнения		Байты кода	Пример
	387	486		
Нет операндов	3+5n*	4	1	FWAIT

* - n=сколько раз ЦП проверяет линию BUSY перед завершением выполнения предыдущей инструкции.

FXAM Проверка вершины стека.

Исключительные ситуации: нет.

FXAM (нет операндов)

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Нет операндов	12-23	12-23	30-38	8	2	FXAM

FXCH Обмен содержимого регистров.

Исключительные ситуации: I.

FXCH //приемник

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
//ST(i)	10-15	10-15	18	4	2	FXCH ST(2)

FXTRACT Выделение экспоненты и значащей части.

Исключительные ситуации: I.

FXTRACT (нет операндов)

Операнды	Такты выполнения			Байты кода	Пример
	87	287	387	486		
Нет операндов	27-55	27-55	70-76	19(16-20)	2	FXTRACT

FYL2X Y * log2 X.

Исключительные ситуации: P (операнды не проверяются).

FYL2X (нет операндов)

Операнды	Такты выполнения			Байты кода	Пример
	87	287	387	486	
Нет операндов	100-1100	900-1100	120-538	311(196-329)	FYL2X

FYL2XP1 Y * log2 (X+1).

Исключительные ситуации: P (операнды не проверяются).

FYL2XP1 (нет операндов)

Операнды	Такты выполнения			Байты кода	Пример
	87	287	387	486		
Нет операндов	100-1000	700-1000	257-547	313(171-326)	2	FYL2XP1

F2XM1 2 с степени X, минус 1.

Исключительные ситуации: P (операнды не проверяются).

F2XM1 (нет операндов)

Операнды	Такты выполнения				Байты кода	Пример
	87	287	387	486		
Нет операндов	310-630	310-630	211-476	242(140-279)	2	F2XM1