

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ»

ПРОГРАММИРОВАНИЕ НА АССЕМБЛЕРЕ

Учебно-методическое пособие

Составители:
Г.Э. Вошинская,
Е.Е. Михайлова,
Е.М. Лещенко

Воронеж
Издательский дом ВГУ
2015

Содержание

Введение.....	4
Начальные сведения о языке ассемблера	5
Задание 1.	26
Организация процедур в ассемблере	27
Задание 2	34
Команды обработки строк.....	34
Задание 3.	38
Литература	40

Представление команд

Машинная команда занимает от 1 до 6 байт.

Формат команды

КОП [операнды]

КОП – код операции – один или два байта. В программе на языке ассемблера записывается мнемоническое обозначение КОП.

Команда может иметь от 0 до 2 операндов.

Операнд определяет, где находятся данные:

- непосредственно в команде,
- в регистре,
- в оперативной памяти.

Для команд с двумя операндами возможны следующие сочетания:

- регистр – регистр,
- регистр – память,
- регистр – непосредственный операнд,
- память – непосредственный операнд.

Тип операнда определяется форматом его записи. При этом содержимое регистра может использоваться или как сам операнд, или как составляющая его адреса.

В команде указывается имя регистра. В ассемблере закреплены следующие названия регистров.

Регистры общего назначения

32-битные регистры: EAX (аккумулятор), EBX (база), ECX (счетчик), EDX (регистр данных). Младшие 16 бит каждого из этих регистров применяются как самостоятельные регистры с именами AX, BX, CX, DX. Кроме этого, отдельные байты в 16-битных регистрах AX – DX тоже могут использоваться как 8-битные регистры и иметь свои имена. Старшие байты этих регистров называются AH, BH, CH, DH, а младшие – AL, BL, CL, DL. Еще четыре регистра общего назначения – ESI (индекс источника), EDI (индекс приемника), EBP (указатель базы), ESP (указатель стека).

Сегментные регистры

При использовании сегментированных моделей памяти для формирования любого адреса нужны два числа – адрес начала сегмента и смещение искомого байта относительно этого начала (в бессегментной модели памяти flat адреса начал всех сегментов равны). В процессорах Intel предусмотрено шесть 16-битных регистров – CS, DS, ES, FS, GS, SS, где хранятся селекторы. В отличие от DS, ES, GS, FS, которые называются регистрами сегментов данных, CS и SS отвечают за сегменты двух особенных типов – сегмента кода и сегмента стека.

Еще один регистр, используемый для организации ветвлений в программе, – регистр флагов FLAGS. Каждый его бит устанавливается в 1 при определенных условиях.

0	NT	IOPL		OF	DF	IF	TF	SF	ZF	0	AF	0	PF	1	CF
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

CF – флаг переноса. Устанавливается в 1, если результат предыдущей операции не уместился в приемнике и произошел перенос из старшего бита, или если требуется заем (при вычитании), в противном случае – в 0. Например, после сложения слова 0FFFFh и 1, если регистр, в который надо поместить результат, – слово, в него будет записано 0000h и флаг CF = 1.

PF – флаг четности. Устанавливается в 1, если младший байт результата предыдущей команды содержит четное число битов, равных 1, и в 0, если нечетное.

AF – флаг полупереноса или вспомогательного переноса. Устанавливается в 1, если в результате предыдущей операции произошел перенос (или заем) из третьего бита в четвертый. Этот флаг используется автоматически командами двоично-десятичной коррекции.

ZF – флаг нуля. Устанавливается в 1, если результат предыдущей команды – ноль.

SF – флаг знака. Он всегда равен старшему биту результата.

TF – флаг ловушки. Он был предусмотрен для работы отладчиков, не использующих защищенный режим. Установка его в 1 приводит к тому, что после выполнения каждой программной команды управление временно передается отладчику (вызывается прерывание 1).

IF – флаг прерываний. Сброс этого флага в 0 приводит к тому, что процессор перестает обрабатывать прерывания от внешних устройств. Обычно его сбрасывают на короткое время для выполнения критических участков кода.

DF – флаг направления. Он контролирует поведение команд обработки строк: когда он установлен в 1, строки обрабатываются в сторону уменьшения адресов, когда DF = 0 – наоборот.

OF – флаг переполнения. Он устанавливается в 1, если результат предыдущей арифметической операции над числами со знаком выходит за допустимые для них пределы. Например, если при сложении двух положительных чисел получается число со старшим битом, равным единице, то есть отрицательное, и наоборот.

Флаги IOPL (уровень привилегий ввода-вывода) и NT (вложенная задача) применяются в защищенном режиме.

Конструкции ветвления реализуются в языке ассемблера проверкой нужного флага.

Регистровая адресация

Операнды могут располагаться в любых регистрах общего назначения и сегментных регистрах. Значение операнда – содержимое регистра.

Некоторые команды используют определенные регистры специальным образом, при этом в команде регистр не указывается.

Пример команды с регистровой адресацией.

MOV AX,CX – пересылка содержимого регистра CX в регистр AX.

Непосредственный операнд

Некоторые команды позволяют использовать константы в качестве операнда-источника. В качестве непосредственного операнда могут быть числовые или символьные данные. Размер непосредственной константы должен быть согласован с размером второго операнда. Использование непосредственного операнда эффективнее, чем определение числовой константы в памяти.

Например, MOV AX, 500 – помещает в регистр AX число 500.

Прямая адресация

Исполнительный адрес является составной частью команды. Этот исполнительный адрес добавляется к содержимому сегментного регистра. Обычно прямая адресация применяется, если операндом служит метка.

Например, MOV AX, TABLE – пересылка значения, находящегося в оперативной памяти по адресу TABLE, в регистр AX.

Если содержимое сегмента данных:

0000		
0001	BB	TABLE
0002	AA	
0003		

Здесь значение метки TABLE равно 0001 относительно сегмента данных. Сегментный регистр по умолчанию – DS.

В результате содержимое регистра:

AX AABB

Косвенная адресация

Исполнительный адрес операнда содержится в BX, BP, SI или в DI. Начиная с процессора 80386 и старше можно использовать EAX, EBX, ECX, EDX, ESI, EDI, EBP, ESP.

Например, MOV AX,[BX] – пересылка значения, находящегося по адресу, взятому из регистра BX, в регистр AX. По умолчанию здесь в качестве сегментного регистра используется DS.

Адресация по базе

Исполнительный адрес вычисляется сложением сдвига и содержимого регистра, указанного в команде, – одного из регистров BX, BP, SI, DI, EAX, EBX, ECX, EDX, ESI, EDI, EBP, ESP.

Например, MOV AX, [BP]+4

BP 001A

Содержимое сегмента данных:

001A	
001B	
001C	
001D	
001E	BB
001F	AA

Результат команды:

AX AABB

Адресация с индексированием

Исполнительный адрес вычисляется как сумма значений сдвига и индексного регистра. В качестве 16-разрядных регистров можно использовать DI или SI, из 32-разрядных EAX, EBX, ECX, EDX, ESI, EDI.

Например, MOV AX, TABLE[DI] – пересылка значения, адрес которого вычисляется как сумма адреса TABLE и содержимого регистра DI.

DI 0004

Содержимое сегмента данных:

0001		TABLE
0002		
0003		
0004		
0005	BB	
0006	AA	

Результат команды:

AX AABB

Адресация по базе с индексированием

Исполнительный адрес вычисляется как сумма значений базового регистра, индексного регистра и, возможно, сдвига. Из 16-битных регистров можно складывать только BX+SI, BX+DI, BP+SI, BP+DI, а из 32-битных – все восемь регистров общего назначения. Например,

MOV AX, [BX][SI]+2 или MOV AX, [BX+SI]+2

Основные непривилегированные команды

MOV *приемник, источник*

Пересылка данных. Копирует содержимое источника в приемник, источник не изменяется. В качестве источника для MOV могут использоваться: число (непосредственный операнд), регистр общего назначения, сегментный регистр или переменная (то есть операнд, находящийся в памяти); в качестве приемника: регистр общего назначения, сегментный регистр (кроме CS) или переменная. Оба операнда должны быть одного и того же размера – байт, слово или двойное слово. Нельзя выполнять пересылку данных с помощью MOV из одной переменной в другую, из одного сегментного регистра в другой, и нельзя помещать в сегментный регистр непосредственный операнд – эти операции выполняют двумя командами MOV (из сегментного регистра в обычный и уже из него – в другой сегментный), или парой команд PUSH/POP.

XCHG *операнд1, операнд2*

Обмен операндов между собой. Содержимое операнда2 копируется в операнд1, а старое содержимое операнда1 – в операнд2. XCHG можно выполнять над двумя регистрами или над регистром и переменной.

PUSH *источник*

Поместить данные в стек. Помещает содержимое источника в стек. Источником может быть регистр, сегментный регистр, непосредственный операнд или переменная. Фактически эта команда уменьшает ESP на размер источника в байтах (2 или 4) и копирует содержимое источника в память по адресу SS:[ESP]. Команда PUSH почти всегда используется в паре с POP (считать данные из стека). Поэтому, чтобы скопировать содержимое одного сегментного регистра в другой (что нельзя выполнить одной командой MOV), можно использовать такую последовательность команд:

PUSH CS

POP DS ; Теперь DS указывает на тот же сегмент, что и CS.

POP *приемник*

Считать данные из стека. Помещает в приемник слово или двойное слово, находящееся в вершине стека, увеличивая ESP на 2 или 4 соответственно. POP выполняет действие, полностью обратное PUSH. Приемником может быть регистр общего назначения, сегментный регистр, кроме CS (чтобы загрузить CS из стека, надо воспользоваться командой RET), или переменная. Если в роли приемника выступает операнд, использующий ESP для косвенной адресации, команда POP вычисляет адрес операнда уже после того, как она увеличивает ESP.