**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «**ОРГАНИЗАЦИЯ ЭВМ и СИСТЕМ**»**

Тема: **Трансляция, отладка и выполнение программ на языке Ассемблера**

|  |  |  |
| --- | --- | --- |
| Студент гр. 8383 |  | Ларин Антон |
| Преподаватель |  |  |

Санкт-Петербург

2019

**Цель работы.**

Научится транслировать, отлаживать и выполнять программы на языке Ассемблера. Изучить основные моменты архитектуры, работу с регистрами. Разобраться в работе конкретных примеров.

**Основные теоретические положения.**

В процессоре Intel 8086 длина слова составляет 16 бит или 2 байта. Минимально адресуемой и обрабатываемой единицей информации является байт, при этом адрес слова совпадает с адресом младшего байта и является четным. При разрядности адреса 16 бит максимальная емкость прямо адресуемой памяти составляет   
216 = 64 Кбайт. Для расширения адресуемого пространства памяти используется ее разбиение на блоки – сегменты, называемое сегментированием памяти. Каждый сегмент имеет произвольную длину, не превышающую  
64 Кбайт. Адрес байта в сегментированной памяти задается двумя составляющими: сегментная часть (16 бит), определяющая адрес начала сегмента, и смещение (16 бит) байта в пределах сегмента. Обычно адрес представляется в виде пары Сегмент (Segment) : Смещение (Offset). Начало размещения сегмента выравнивается на границу блока памяти из 16 байт, называемого «параграфом». Соответственно, физический адрес памяти получается путем суммирования сегмента, сдвинутого на 4 бита влево, со значением смещения. Результатом такого суммирования является 20-битный физический адрес, чем обеспечивается адресация 1 Мбайт памяти.

# **Программно доступные регистры процессора**

Регистры общего назначения

16-битные регистры АХ (аккумулятор), ВХ (база), СХ (счетчик), DX (регистр данных) могут использоваться без ограничений для любых це­лей – временного хранения данных, аргументов или результатов различ­ных операций. Кроме этого, отдельные байты в 16-бит­ных регистрах АХ - DX тоже имеют свои имена и могут использоваться как 8-битные регистры. Старшие байты этих регистров называются АН, ВН, СН, DH, а младшие - AL, DL, CL, DL. Другие четыре регистра общего назначения (РОН) – SI (индекс источни­ка), DI (индекс приемника), BP (указатель базы), SP (указатель сте­ка) – имеют более конкретное назначение и могут применяться для хра­нения всевозможных временных переменных, только когда они не используются по назначению. Регистры SI и DI используются в стро­ковых операциях, ВР и SP используются при работе со стеком.

Смещение следующей выполняемой команды всегда хранится в специальном регистре – IP (указатель инструкции), принудительная запись в кото­рый приведет к тому, что следующей будет исполнена какая-ни­будь другая команда, а не команда, расположенная сразу за данной. В общем случае при передаче управления в другой сегмент команды безусловного перехода, вызова подпрограммы и т. п. – осуществляют запись в регистры CS и IP.

**Регистр флагов**

CF - флаг переноса (CARRY);

PF - флаг четности (PARITY);

AF - дополнительный флаг переноса (AUXILARY);

ZF - флаг нуля (ZERO);

SF - знаковый флаг (SIGN);

TF - флаг слежения, ловушка (TRAP);

IF - флаг прерываний (INTERRUPTION);

DF - флаг направления (DIRECTION);

OF - флаг переполнения (OVERFLOW).

В начальной части com-файла программы (а также и exe-файла) MS DOS размещает специальный блок – префикс сегмента программы ПСП (PSP - рrogram segment prefix), который содержит информацию для доступа программы к параметрам командной строки, к среде окружения, для реакции программы на критические ошибки и управляющие команды.

**Все команды процессора i8086 можно разделить на следующие группы:**

1. команды передачи данных;
2. команды арифметических операций над целыми числами;
3. логические команды;
4. сдвиговые команды;
5. команды обработки строк;
6. команды передачи управления;
7. команды прерываний;
8. команды управления флагами;
9. команды управления состоянием процессора;
10. команды плавающей арифметики;
11. команды мультимедийных расширений (ММХ – MultiMedia eXtension);
12. команды потокового расширения (SSE – Streaming SIMD Extension).

**Задание**

В каталоге Comput\_Org/Лабораторные работы/ содержатся 2 подкаталога: 1) MASM (с транслятором masm, компоновщиком link, отладчиком afd и библиотекой lib) и

2) Задания, в котором приведены описания заданий 9 лабораторных работ. Обязательными для получения зачета являются первые 6 работ, выполнение любой из работ 7 – 9 (по выбору студента) позволяет получить бонусы к экзамену (0.5 балла за 7 работ, 1 балл за 8 работ).

Все работы на «старых» ПК (≤ 32-бит и ОС Windows ≤ XP ) выполняются из FAR c запуском в командной строке, так как используют режим DOS.

При работе на «новых» ПК ( ≥ 64-бит и ОС Windows > XP ) необходимо работать под эмулятором DOSBOX (для личной машины скачать из Интернета). Инструкция по работе с

DOSBOX приведена в приложении 1 к данной работе.

Лабораторная работа 1 использует 2 готовых программы на ассемблере: hello1 – составлена с использованием сокращенного описания сегментов и hello2 – составлена с полным описанием сегментов и выводом строки, оформленным как процедура. Выполнение работы состоит из двух частей, по каждой из которых необходимо представить протокол с фиксацией всех выполняемых действий и полученных результатов, и подписать его у преподавателя.

Уточнение задания следует посмотреть в файле lr1\_comp.txt каталога Задания.

Часть 1

1. Просмотреть программу hello1.asm, которая формирует и выводит на экран приветствие пользователя с помощью функции ОС MSDOS, вызываемой через прерывание с номером 21H (команда Int 21h).

Выполняемые функцией действия и задаваемые ей параметры - следующие:

* обеспечивается вывод на экран строки символов, заканчивающейся знаком "$";
* требуется задание в регистре ah номера функции, равного 09h, а в регистре dx - смещения адреса выводимой строки;
* используется регистр ax и не сохраняется его содержимое.

1. Разобраться в структуре и реализации каждого сегмента программы. Непонятные фрагменты прояснить у преподавателя. Строку-приветствие преобразовать в соответствии со своими личными данными.
2. Загрузить файл hello1.asm из каталога Задания в каталог Masm.
3. Протранслировать программу с помощью строки

> masm hello1.asm

c созданием объектного файла и файла диагностических сообщений (файла листинга). Объяснить и исправить синтаксические ошибки, если они будут обнаружены транслятором. Повторить трансляцию программы до получения объектного модуля.

1. Скомпоновать загрузочный модуль с помощью строки

> link hello1.obj

с созданием карты памяти и исполняемого файла hello1.exe.

1. Выполнить программу в автоматическом режиме путем набора строки

> hello1.exe

убедиться в корректности ее работы и зафиксировать результат выполнения в протоколе.

1. Запустить выполнение программы под управлением отладчика с помощью команды

> afd hello1.exe

Записать начальное содержимое сегментных регистров CS, DS, ES и SS. Выполнить программу в пошаговом режиме с фиксацией используемых регистров и ячеек памяти до и после выполнения каждой команды. Обычные команды выполняются по F1 (Step), а вызовы обработчиков прерываний (Int) - по F2 (StepProc), чтобы не входить внутрь обработчика прерываний. Продвижение по сегментам экранной формы отладчика выполняется с помощью клавиш F7 – F10 (up, down, left, right). Перезапуск программы в отладчике выполняется клавишей F3 (Retrieve). Выход из отладчика - по команде Quit.

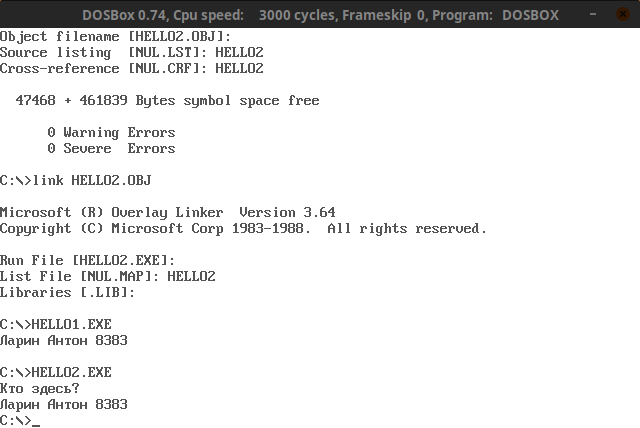
Результаты прогона программы под управлением отладчика должны быть представлены в виде, показанном на примере одной команды в табл.1, и подписаны преподавателем.

Табл.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Адрес  Команды | Символический  код команды | 16-ричный  код команды | Содержимое регистров и ячеек памяти | |
| до выполнения . | После выполнения |
| 0003 | Mov DS, AX | 8E D8 | (AX) = 2D87  (DS) = 2D75  (IP) = 0003 | (AX) = 2D87  (DS) = 2D87  (IP) = 0005 |

**Выполнение**

Результат трансляции, линковки и исполнения тестовых программ

****

Отладка HELLO1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Адрес  Команды | Символический  код команды | 16-ричный  код команды | Содержимое регистров и ячеек памяти | |
| до выполнения . | После выполнения |
| 0010 | mov ax, @data | B8AE11 | (CS) = 11AC  (DS) = 119C  (ES) = 119C  (SS) = 11B0  (AX) = 0000  (IP) = 0010 | (CS) = 11AC  (DS) = 119C  (ES) = 119C  (SS) = 11B0  (AX) = 11AE  (IP) = 0013 |
| 0013 | mov ds, ax | 8ED8 | (DS) = 119C  (IP) = 0013 | (DS) = 11AE  (IP) = 0015 |
| 0015 | mov dx, OFFSET Greeting | BA0000 | (DX) = 0000  (IP) = 0015 | (DX) = 0000  (IP) = 0018 |
| 0018 | mov ah, 9 | B409 | (AH) = 11  (AX) = 11AE  (IP) = 0018 | (AH) = 09  (AX) = 09AE  (IP) = 001A |
| 001A | int 21h | CD21 | (IP) = 001A | (IP) = 001C |
| 001C | mov ah, 4ch | B44C | (AH) = 09  (AX) = 09AE  (IP) = 001C | (AH) = 4C  (AX) = 4CAE  (IP) = 001E |
| 001E | int 21h | CD21 | (AX) = 4CAE  (CX) = 0033  (DS) = 119C  (IP) = 001E | (AX) = 0000  (CX) = 0000  (DS) = 11AE  (IP) = 0010 |

Отладка HELLO2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Адрес  Команды | Символический  код команды | 16-ричный  код команды | Содержимое регистров и ячеек памяти | |
| до выполнения . | После выполнения |
| 0005 | push DS | 1E | (CS) = 11B0  (DS) = 119C  (ES) = 119C  (SS) = 11AC  (SP) = 0018  (IP) = 0005  Stack +0 = 0000 | (CS) = 11B0  (DS) = 119C  (ES) = 119C  (SS) = 11AC  (SP) = 0016  (IP) = 0006  Stack +0 = 119C |
| 0006 | sub AX,AX | 2BC0 | (AX) = 0000  (ZF) = 0  (PF) = 0  (IP) = 0006  FLAGS = 0200 | (AX) = 0000  (ZF) = 1  (PF) = 1  (IP) = 0008  FLAGS = 0244 |
| 0008 | push AX | 50 | (SP) = 0016  (IP) = 0008  Stack +0 = 119C  Stack +2 = 0000 | (SP) = 0014  (IP) = 0009  Stack +0 = 0000  Stack +2 = 119C |
| 0009 | mov AX,DATA | B8AE11 | (AX) = 0000  (IP) = 0009 | (AX) = 11AE  (IP) = 000C |
| 000C | mov DS,AX | 8ED8 | (DS) = 119C  (IP) = 000C | (DS) = 119C  (IP) = 000E |
| 000E | mov DX, OFFSET HELLO | BA0000 | (DX) = 0000  (IP) = 000E | (DX) = 0000  (IP) = 0011 |
| 0011 | call WriteMsg | E8ECFF | (SP) = 0014  (IP) = 0011  Stack +0 = 0000  Stack +2 = 119C  Stack +4 = 0000 | (SP) = 0012  (IP) = 0000  Stack +0 = 0014  Stack +2 = 0000  Stack +4 = 119C |
| 0000 | mov AH,9 | B409 | (AH) = 11  (AX) = 11AE  (IP) = 0000 | (AH) = 09  (AX) = 09AE  (IP) = 0002 |
| 0002 | int 21h | CD21 | (IP) = 0002 | (IP) = 0004 |
| 0004 | ret | C3 | (SP) = 0012  (IP) = 0004  Stack +0 = 0014  Stack +2 = 0000  Stack +4 = 119C | (SP) = 0014  (IP) = 0014  Stack +0 = 0000  Stack +2 = 119C  Stack +4 = 0000 |
| 0014 | mov DX, OFFSET GREETING | BA0D00 | (DX) = 0000  (IP) = 0014 | (DX) = 000D  (IP) = 0017 |
| 0017 | call WriteMsg | E8E6FF | (SP) = 0014  (IP) = 0017  Stack +0 = 0000  Stack +2 = 119C  Stack +4 = 0000 | (SP) = 0012  (IP) = 0000  Stack +0 = 001A  Stack +2 = 0000  Stack +4 = 119C |
| 0000 | mov AH,9 | B409 | (AH) = 09  (IP) = 0000 | (AH) = 09  (IP) = 0002 |
| 0002 | int 21h | CD21 | (IP) = 0002 | (IP) = 0004 |
| 0004 | ret | C3 | (SP) = 0012  (IP) = 0004  Stack +0 = 001A  Stack +2 = 0000  Stack +4 = 119C | (SP) = 0014  (IP) = 001A  Stack +0 = 0000  Stack +2 = 119C  Stack +4 = 0000 |
| 001A | ret | CB | (SP) = 0014  (CS) = 11B0  (IP) = 001A  Stack +0 = 0000  Stack +2 = 119C | (SP) = 0018  (CS) = 119C  (IP) = 0000  Stack +0 = 0000  Stack +2 = 0000 |
| 0000 | INT 20 | CD20 | (AX) = 09AE  (CX) = 005B  (CS) = 119C  (DS) = 11AE  (IP) = 0000  (ZF) = 1  (PF) = 1  FLAGS = 0244 | (AX) = 0000  (CX) = 0000  (CS) = 11B0  (DS) = 119C  (IP) = 0005  (ZF) = 0  (PF) = 0  FLAGS = 0200 |

**Выводы.**

В результате работы были разобраны некоторые базовые концепции языка асемблера. Были удачно транслированы и собраны тестовые программы, разобран принцип их работ, произведена пошаговая отладка с занесением результатов в таблицу.

Приложение 1

HELLO1.ASM

; HELLO1.ASM - упрощенная версия учебной программы лаб.раб. N1

; по дисциплине "Архитектура компьютера"

; \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Назначение: Программа формирует и выводит на экран приветствие

; пользователя с помощью функции ДОС "Вывод строки"

; (номер 09 прерывание 21h), которая:

; - обеспечивает вывод на экран строки символов,

; заканчивающейся знаком "$";

; - требует задания в регистре ah номера функции=09h,

; а в регистре dx - смещения адреса выводимой

; строки;

; - использует регистр ax и не сохраняет его

; содержимое.

; \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

DOSSEG ; Задание сегментов под ДОС

.MODEL SMALL ; Модель памяти-SMALL(Малая)

.STACK 100h ; Отвести под Стек 256 байт

.DATA ; Начало сегмента данных

Greeting LABEL BYTE ; Текст приветствия

DB 'Ларин Антон 8383',13,10,'$'

.CODE ; Начало сегмента кода

mov ax, @data ; Загрузка в DS адреса начала

mov ds, ax ; сегмента данных

mov dx, OFFSET Greeting ; Загрузка в dx смещения

; адреса текста приветствия

DisplayGreeting:

mov ah, 9 ; # функции ДОС печати строки

int 21h ; вывод на экран приветствия

mov ah, 4ch ; # функции ДОС завершения программы

int 21h ; завершение программы и выход в ДОС

END

Приложение 2

HELLO2.ASM

; HELLO2 - Учебная программа N2 лаб.раб.#1 по дисциплине "Архитектура компьютера"

; дштл Программа использует процедуру для печати строки

;

; ТЕКСТ ПРОГРАММЫ

EOFLine EQU '$' ; Определение символьной константы

; "Конец строки"

; Стек программы

AStack SEGMENT STACK

DW 12 DUP(?) ; Отводится 12 слов памяти

AStack ENDS

; Данные программы

DATA SEGMENT

; Директивы описания данных

HELLO DB 'Кто здесь?', 0AH, 0DH,EOFLine

GREETING DB 'Ларин Антон 8383$'

DATA ENDS

; Код программы

CODE SEGMENT

ASSUME CS:Code ;DS:DATA SS:STACKSS

; Процедура печати строки

WriteMsg PROC NEAR

mov AH,9

int 21h ; Вызов функции DOS по прерыванию

ret

WriteMsg ENDP

; Головная процедура

Main PROC FAR

push DS ;\ Сохранение адреса начала PSP в стеке

sub AX,AX ; > для последующего восстановления по

push AX ;/ команде ret, завершающей процедуру.

mov AX,DATA ; Загрузка сегментного

mov DS,AX ; регистра данных.

mov DX, OFFSET HELLO ; Вывод на экран первой

call WriteMsg ; строки приветствия.

mov DX, OFFSET GREETING ; Вывод на экран второй

call WriteMsg ; строки приветствия.

ret ; Выход в DOS по команде,

; находящейся в 1-ом слове PSP.

Main ENDP

CODE ENDS

END Main