# ЛАБОРАТОРНАЯ РАБОТА №7

# ИССЛЕДОВАНИЕ АРХИТЕКТУРЫ 32-РАЗРЯДНЫХ ПРОЦЕССОРОВ С ПОМОЩЬЮ ОТЛАДЧИКА OLLYDBG

Цель работы

Исследование архитектуры 32-разрядных процессоров и возможностей отладчика OllyDbg при разработке и отладке ассемблерных программ, приобретение практических навыков по анализу и отладке программ на языке ассемблера.

Задачи

1. Повторить теоретический материал, касающийся структуры 32-разрядных микропроцессоров (включая суперскалярные), программно доступных регистров и системы команд языка ассемблера;
2. Создать приложение типа Win32 с графическим интерфейсом, вычисляющее сумму и разность двух чисел: первое число – номер в группе, второе – число, противоположное номеру первой буквы фамилии в алфавите;
3. Загрузить разработанное приложение в отладчик OllyDbg и исследовать его работу;
4. Проанализировать, какие изменения происходят в различных регистрах процессора:
   * изменяйте значения регистров EAX и EBX таким образом, чтобы произошли изменения состояний флагов O, P, Z, S, C;
   * поместите значение в стек с помощью инструкции PUSH;
   * извлеките значение из стека с помощью инструкции POP;
   * изучите принцип работы инструкции MOV c 8-ми, 16-ти и 32-х разрядными регистрами;
   * изучите работу математических инструкций: INC, DEC, ADD, SUB, MUL, IMUL, DIV, NEG;
   * измените значение регистра ESP;
   * измените значение регистра EIP;
   * найдите инструкцию PUSH 0 и замените ее на NOP;

Ход работы

Номер студента в группе - 8. Номер первой буквы фамилии студента – 12 (М).

Была написана консольная программа на языке C, которая находит и выводит сумму и разность этих двух чисел (Рисунок 1).

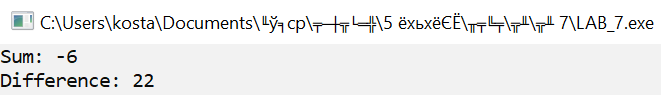


Рисунок 1 – Результат выполнения программы

Программа была скомпилирована в 32-битном формате и загружена в отладчик OllyDbg (Рисунок 2). В ходе её пошагового выполнения можно наблюдать изменение значений в памяти, стеке и регистрах 32-разрядного процессора.

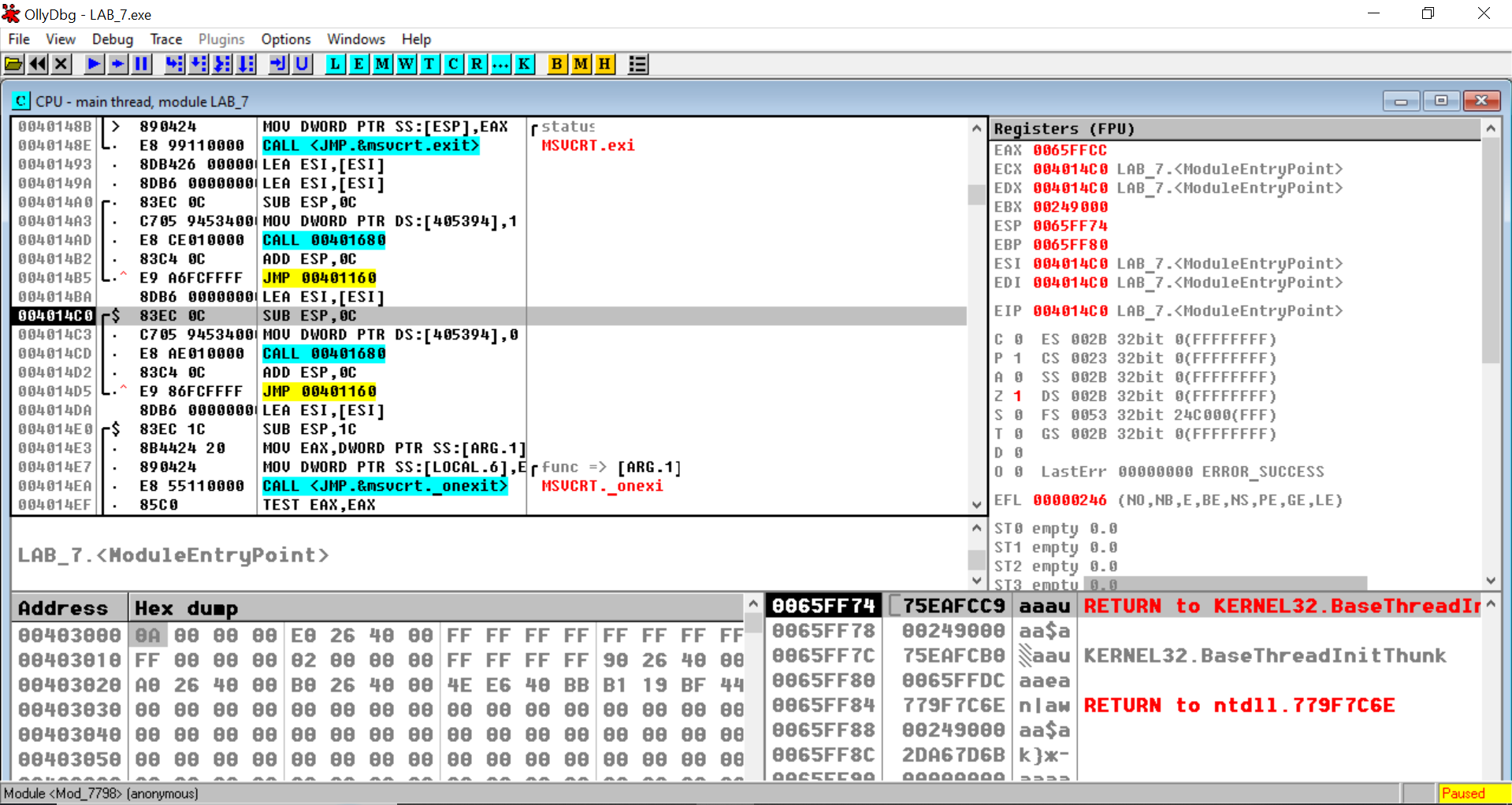


Рисунок 2 – Рассмотрение программы в OllyDbg

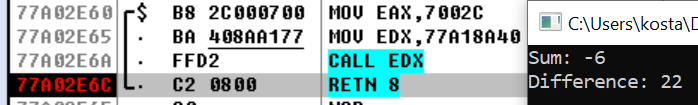


Рисунок 3 – Выполнение программы в отладчике

O – флаг переполнения. Установлен в случае, если операция привела к займу в старший разряд. Для его изменения прибавим к 7FFFFFFFh единицу. Поскольку первый бит отвечает за отрицательность, 80000000h – первое отрицательное число, следовательно при операции сложения происходит переполнение, O=1 (Рисунок 4).

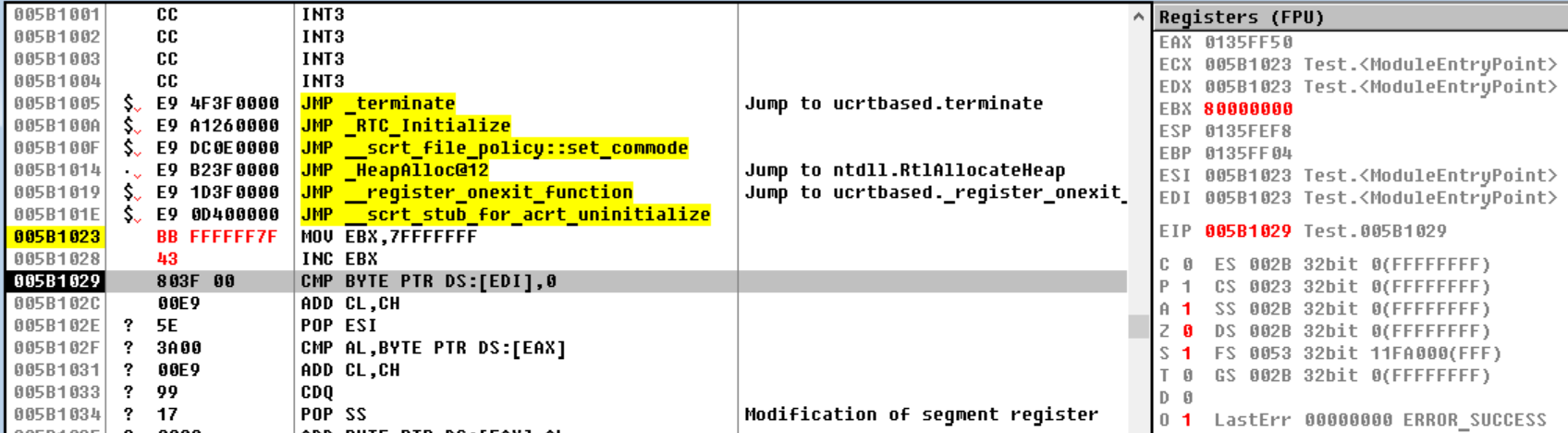


Рисунок 4 – Изменение флага переполнения O

P – флаг чётности, установлен если в младших 8 бит результата чётное число единиц. Изначально флаг равен 1, для его сброса к EAX (младший байт равен 0) было прибавлено 3 (112).

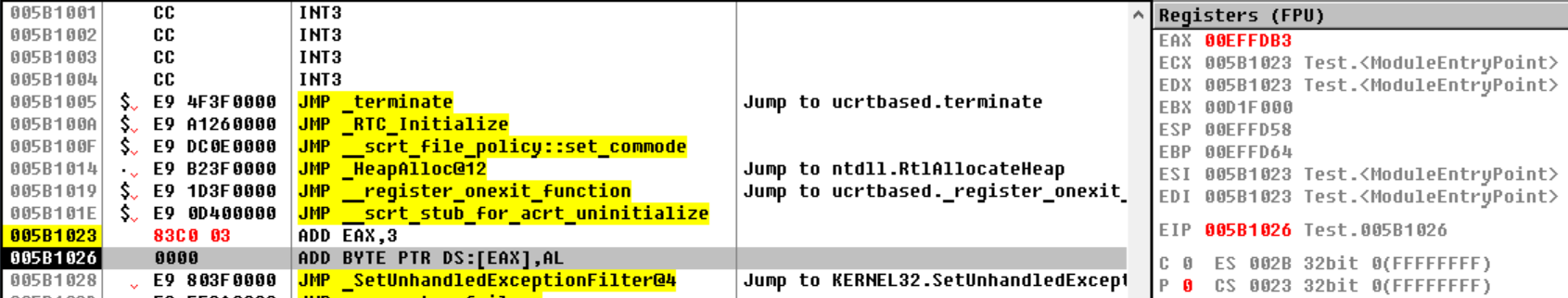


Рисунок 5 – Изменение флага чётности P

Z – флаг нуля, установлен если результат операции равен 0. Для его установки из регистра EAX было убавлено его изначальное значение – 10FFF3C (Рисунок 6).

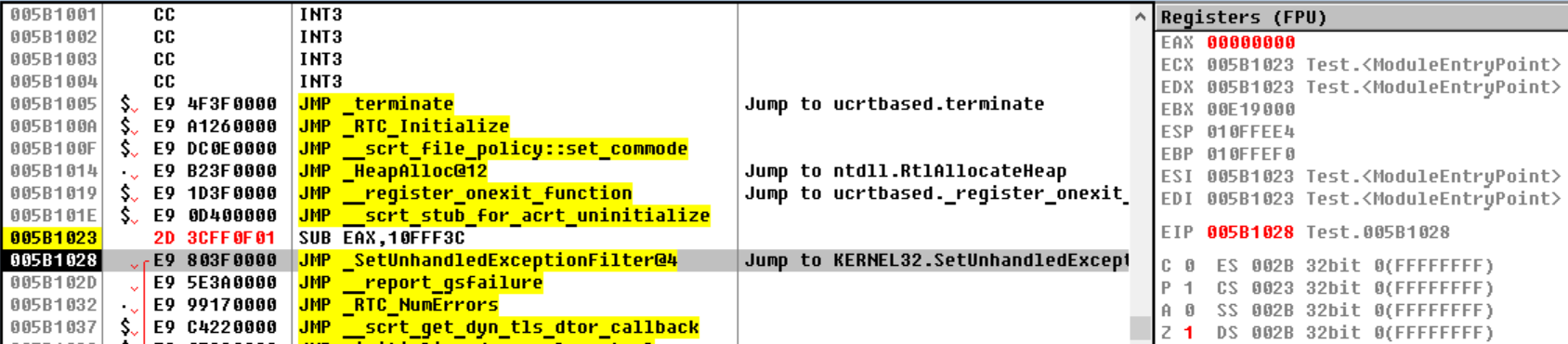


Рисунок 6 – Изменение регистра нуля Z

C – флаг переноса, установлен если в результате операции произошёл займ из старшего разряда (при сложении) или займ для старшего бита (при вычитании). Для его изменения из A было вычтено FFFFFFF (займ для вычитания).

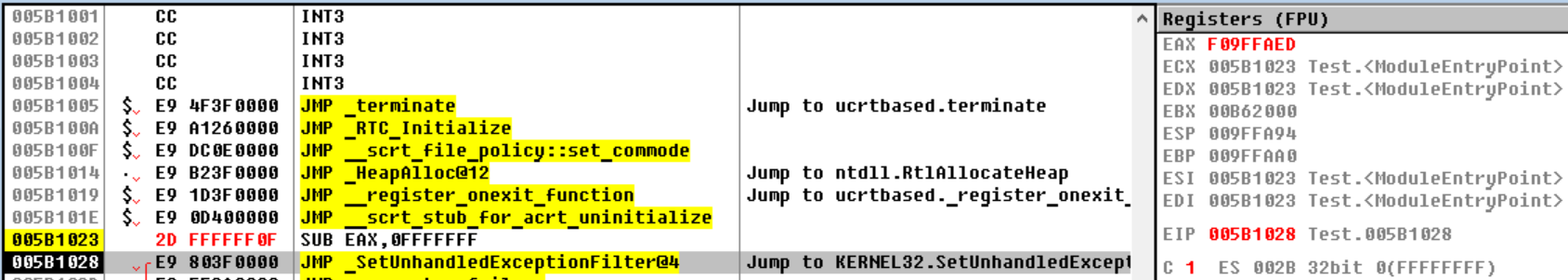


Рисунок 7 – Изменение флага переполнения C

Для тестирования работы со стеком были прописаны ассемблерные команды PUSH 7777777 и POP EAX. Первая помещает в стек значение 7777777h (Рисунок 8), вторая снимает его с вершины стека и заносит в регистр EAX (Рисунок 9). Также в ходе операций можно заметить изменение указателя стека SP.

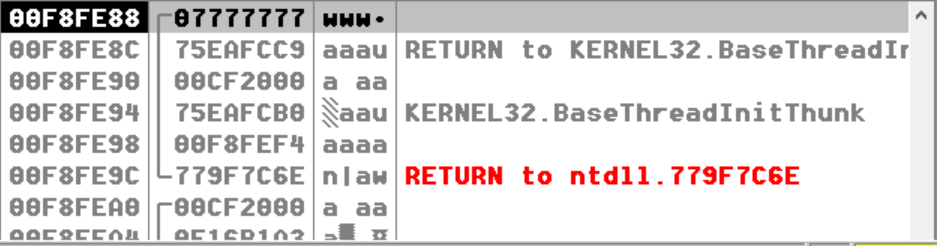


Рисунок 8 – Занесение значения в стек

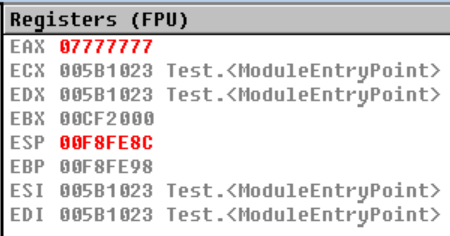


Рисунок 9 – Снятие значения из стека

Для примера работы с 32-, 16- и 8-разрядными регистрами в рамках одного процессора были прописаны следующие команды:

MOV EAX, 11111111

MOV AX, 0

MOV AL, 11

В результате в регистре EAX сохранилось значение 11110011.

Все регистры 32-разрядного процессора соответственно 32-разрядные. К ним обращаются так: EAX, EBX, ECX, EDX и др. Однако для РОН возможен доступ к 16 младшим разрядам, а в них к младшему и старшему байтам. Обращения те же, что и для 16-разрядного процессора: AX (AH/AL), BX (BH/BL) и др.

Работа с математическими инструкциями была продемонстрирована ранее. Вот их описание:

* INC – инкремент, прибавляет к операнду 1;
* DEC – декремент, убавляет от операнда единицу;
* ADD – прибавляет к первому операнду второй;
* SUB – вычитание из первого операнда второго;
* MUL – беззнаковое умножение EAX на операнд;
* IMUL – беззнаковое умножение EAX на операнд;
* DIV – беззнаковое деление EAX на операнд;
* NEG – изменение знака числа (операнда) на противоположный;

ESP – регистр указателя стека, обозначает смещение верхнего элемента стека в его сегменте. При его изменении в фрейме стека меняется указатель на текущий элемент. При выборе некорректного смещения программа становится неработоспособна.

EIP – регистр для смещения в сегменте кода, является указателем на текущую исполняемую строчку кода. Его нельзя изменить вручную, только через последовательное выполнение команд и инструкции перехода (JMP, CALL).

При замене операции PUSH 0 на NOP в программу были вставлены две операции NOP, следовательно, по размеру операции не эквивалентны.



Рисунок 10 – Замена PUSH 0 на NOP

Код программы

Листинг 1 – Сложение и вычитание 32-разрядных чисел

#include <stdio.h>

#include <stdint.h>

int main()

{

int32\_t a = 8;

int32\_t b = -14;

int32\_t summ = a + b;

printf("Sum: %d\n", summ);

int32\_t diff = a - b;

printf("Difference: %d\n", diff);

return 0;

}

Вывод

В ходе работы была исследована архитектура 32-разрядного процессора 80386 и возможности отладчика OllyDbg при разработке и отладке 32-битных приложений. Были приобретены навыки написания кода под регистры 32-разрядного процессора на языке ассемблер, а также изучения деассемблированного машинного кода.