Министерство Образования и Науки РФ

Новосибирский Государственный Технический Университет

Кафедра ТПИ

**Лабораторная работа №1**

по дисциплине «Низкоуровневое программирование»

Факультет: ПМИ

Группа: ПМИ-31

Студенты: Тальников В.О.

Савицкий Ю.Р.

Мазин C.C.

Преподаватель: Лисицин Д. В.

Вариант: 7

Новосибирск

2015

1. **Цель работы**

Изучить и приобрести практические навыки работы с основными командамиязыка Ассемблера, функциями ввода-вывода, регистрами и символьными данными.

1. **Задание**

Разработать программу на языке Ассемблера, решающую поставленную задачу:

* + ввод с клавиатуры 2-х чисел в восьмеричной системе счисления;
  + выполнение операции умножения над этими числами (в предположении, что размер чисел не вызывает переполнения регистров);
  + вывод результата в десятичной системе счисления.

1. **Текст программы**

.386

.MODEL FLAT, STDCALL

OPTION CASEMAP: NONE

; прототипы внешних функций (процедур) описываются директивой EXTERN, после знака @ указывается общая длина передаваемых параметров; после двоеточия указывается тип внешнего объекта – процедура

EXTERN GetStdHandle@4: PROC

EXTERN WriteConsoleA@20: PROC

EXTERN CharToOemA@8: PROC

EXTERN ReadConsoleA@20: PROC

EXTERN ExitProcess@4: PROC; функция выхода из программы

EXTERN lstrlenA@4: PROC; функция определения длины строки

.DATA; сегмент данных

STRN1 DB "Введите первое число в восьмеричной системе: ",0; выводимая строка, в конце добавлены управляющие символы: 13 – возврат каретки, 10 – переход на новую строку,

;0 – конец строки; с использованием директивы DB резервируется массив байтов

STRN2 DB "Введите второе число в восьмеричной системе: ",0

STRN3 DB "Ответ в десятичной системе: ",0

STRN4 DB "Введенный символ в числе не соответствует восьмеричной системе счисления. ", 0

DIN DD ?; дескриптор ввода; директива DD резервирует память объемом 32 бита (4 байта), знак «?» используется для неинициализированных данных

DOUT DD ?; дескриптор вывода

BUF DB 200 dup (?); буфер для вводимых/выводимых строк длиной 200 байтов

LENS DD ?; переменная для количества выведенных символов

ETH DD 8

TEN DD 10

FLAG DD 0

FLAG2 DD 0

TEMP1 DD ?

TEMP2 DD ?

.CODE; сегмент кода

MAIN PROC; начало описания процедуры с именем MAIN перекодируем строку STRN1

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Вывод строки приглашения для ввода первого числа\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

MOV EAX, OFFSET STRN1; командой MOV значение второго операнда перемещается в первый, OFFSET – операция, возвращающая адрес

PUSH EAX; параметры функции помещаются в стек командой PUSH

PUSH EAX

PUSH EAX

PUSH EAX

CALL CharToOemA@8; вызов функции получим дескриптор ввода

PUSH -10

CALL GetStdHandle@4

MOV DIN, EAX ; переместить результат из регистра EAX в ячейку памяти с именем DIN получим дескриптор вывода

PUSH -11

CALL GetStdHandle@4

MOV DOUT, EAX ; определим длину строки STRN

PUSH OFFSET STRN1; в стек помещается адрес строки

CALL lstrlenA@4; длина в EAX вызов функции WriteConsoleA для вывода строки STRN

PUSH 0; в стек помещается 5-й параметр

PUSH OFFSET LENS; 4-й параметр

PUSH EAX; 3-й параметр

PUSH OFFSET STRN1; 2-й параметр

PUSH DOUT; 1-й параметр

CALL WriteConsoleA@20; ввод строки

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Ввод первого числа\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

PUSH 0; в стек помещается 5-й параметр

PUSH OFFSET LENS; 4-й параметр

PUSH 200; 3-й параметр

PUSH OFFSET BUF; 2-й параметр

PUSH DIN; 1-й параметр

CALL ReadConsoleA@20 ; обратите внимание: LENS больше числа введенных символов на два,

;дополнительно введенные символы: 13 – возврат каретки и 10 – переход на новую строку вывод полученной строки

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Обработка первого числа\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

SUB LENS, 2

MOV ECX, LENS; счетчик цикла (строка имеет длину LENS)

MOV ESI, OFFSET BUF; начало строки хранится в переменной BUF

XOR BX, BX; обнулить регистр BX командой XOR, выполняющей побитно операцию «исключающее или»

XOR AX, AX; обнулить регистр AX

CONVERT1: ; метка начала тела цикла

MOV BL, [ESI]; поместить символ из введенной строки в регистр BL, используя косвенную адресацию

CMP BL, '-'

JE MIN1

CMP BL, ' ' ;проверка на наличие пробелов в числе

JE RR

CMP BL, '8'

JAE RR

SUB BL, '0'; вычесть из введенного символа код нуля

MUL ETH; умножить значение AX на 8, результат – в AX

ADD AX, BX; добавить к полученному в AX числу новую цифру

INC ESI; перейти на следующий символ строки

LOOP CONVERT1; перейти на следующую итерацию цикла

MOV TEMP1, EAX

JMP EXITLOOP1

MIN1: INC FLAG

INC ESI

DEC ECX; уменьшили счетчик цикла

JMP CONVERT1

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Вывод строки приглашения для ввода второго числа\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

EXITLOOP1:

MOV EAX, OFFSET STRN2; командой MOV значение второго операнда перемещается в первый, OFFSET – операция, возвращающая адрес

PUSH EAX; параметры функции помещаются в стек командой PUSH

PUSH EAX

PUSH EAX

PUSH EAX

CALL CharToOemA@8; вызов функции получим дескриптор ввода

PUSH -10

CALL GetStdHandle@4

MOV DIN, EAX ; переместить результат из регистра EAX в ячейку памяти с именем DIN получим дескриптор вывода

PUSH -11

CALL GetStdHandle@4

MOV DOUT, EAX ; определим длину строки STRN2

PUSH OFFSET STRN2; в стек помещается адрес строки

CALL lstrlenA@4; длина в EAX вызов функции WriteConsoleA для вывода строки STRN2

PUSH 0; в стек помещается 5-й параметр

PUSH OFFSET LENS; 4-й параметр

PUSH EAX; 3-й параметр

PUSH OFFSET STRN2; 2-й параметр

PUSH DOUT; 1-й параметр

CALL WriteConsoleA@20; ввод строки

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Ввод второго числа\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

PUSH 0; в стек помещается 5-й параметр

PUSH OFFSET LENS; 4-й параметр

PUSH 200; 3-й параметр

PUSH OFFSET BUF; 2-й параметр

PUSH DIN; 1-й параметр

CALL ReadConsoleA@20 ; обратите внимание: LENS больше числа введенных символов на два,

;дополнительно введенные символы: 13 – возврат каретки и 10 – переход на новую строку вывод полученной строки

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Обработка числа\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

SUB LENS, 2

MOV ECX, LENS; счетчик цикла (строка имеет длину LENS)

MOV ESI, OFFSET BUF; начало строки хранится в переменной BUF

XOR BX, BX; обнулить регистр BX командой XOR, выполняющей побитно операцию «исключающее или»

XOR AX, AX; обнулить регистр AX

CONVERT2: ; метка начала тела цикла

MOV BL, [ESI]; поместить символ из введенной строки в регистр BL, используя косвенную адресацию

CMP BL, '-'

JE MIN2

CMP BL, ' ' ;проверка на наличие пробелов в числе

JE RR

CMP BL, '8'

JAE RR

SUB BL, '0'; вычесть из введенного символа код нуля

MUL ETH; умножить значение AX на 8, результат – в AX

ADD AX, BX; добавить к полученному в AX числу новую цифру

INC ESI; перейти на следующий символ строки

LOOP CONVERT2; перейти на следующую итерацию цикла

MOV TEMP2, EAX

JMP EXITLOOP2

MIN2: INC FLAG

INC ESI

DEC ECX; уменьшили счетчик цикла

JMP CONVERT2

EXITLOOP2:

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Перемножение чисел\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

MOV EAX, TEMP1

MUL TEMP2

MOV TEMP2, EAX

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Вывод строки ответа\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

MOV EAX, OFFSET STRN3; командой MOV значение второго операнда перемещается в первый, OFFSET – операция, возвращающая адрес

PUSH EAX; параметры функции помещаются в стек командой PUSH

PUSH EAX

PUSH EAX

PUSH EAX

CALL CharToOemA@8; вызов функции получим дескриптор ввода

PUSH -10

CALL GetStdHandle@4

MOV DIN, EAX ; переместить результат из регистра EAX в ячейку памяти с именем DIN получим дескриптор вывода

PUSH -11

CALL GetStdHandle@4

MOV DOUT, EAX ; определим длину строки STRN3

PUSH OFFSET STRN3; в стек помещается адрес строки

CALL lstrlenA@4; длина в EAX вызов функции WriteConsoleA для вывода строки STRN3

PUSH 0; в стек помещается 5-й параметр

PUSH OFFSET LENS; 4-й параметр

PUSH EAX; 3-й параметр

PUSH OFFSET STRN3; 2-й параметр

PUSH DOUT; 1-й параметр

CALL WriteConsoleA@20; ввод строки

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Обработка числа ответа\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

MOV ECX, TEMP2; счетчик цикла (строка имеет длину LENS)

XOR BX, BX; обнулить регистр BX командой XOR, выполняющей побитно операцию «исключающее или»

XOR AX, AX; обнулить регистр AX

CONVERT3: ; метка начала тела цикла

CMP ECX, 10

JB LS

MOV EAX, ECX

XOR EDX, EDX

DIV TEN

ADD EDX, '0'

PUSH EDX

MOV ECX, EAX

INC FLAG2

INC ECX

LOOP CONVERT3; перейти на следующую итерацию цикла

LS:

ADD ECX, '0'

PUSH ECX

INC FLAG2

CMP FLAG, 1

JE Z

JMP NEXT

Z: PUSH '-'

INC FLAG2

NEXT:

MOV ECX, FLAG2

CONVERT4:

POP [ESI]

INC ESI

LOOP CONVERT4;

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Выводответа\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

SUB ESI, FLAG2

PUSH 0; в стек помещается 5-й параметр

PUSH OFFSET FLAG2; 4-й параметр

PUSH FLAG2; 3-й параметр

PUSH ESI; 2-й параметр

PUSH DOUT; 1-й параметр

CALL WriteConsoleA@20; ввод строки;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Обработка ошибки\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

JMP ED

RR:

MOV EAX, OFFSET STRN4; командой MOV значение второго операнда перемещается в первый, OFFSET – операция, возвращающая адрес

PUSH EAX; параметры функции помещаются в стек командой PUSH

PUSH EAX

PUSH EAX

PUSH EAX

CALL CharToOemA@8; вызов функции получим дескриптор ввода

PUSH -10

CALL GetStdHandle@4

MOV DIN, EAX ; переместить результат из регистра EAX в ячейку памяти с именем DIN получим дескриптор вывода

PUSH -11

CALL GetStdHandle@4

MOV DOUT, EAX ; определим длину строки STRN4

PUSH OFFSET STRN4; в стек помещается адрес строки

CALL lstrlenA@4; длина в EAX вызов функции WriteConsoleA для вывода строки STRN4

PUSH 0; в стек помещается 5-й параметр

PUSH OFFSET LENS; 4-й параметр

PUSH EAX; 3-й параметр

PUSH OFFSET STRN4; 2-й параметр

PUSH DOUT; 1-й параметр

CALL WriteConsoleA@20; ввод строки

ED:

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Пауза\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

PUSH 0; в стек помещается 5-й параметр

PUSH OFFSET LENS; 4-й параметр

PUSH 200; 3-й параметр

PUSH OFFSET BUF; 2-й параметр

PUSH DIN; 1-й параметр

CALL ReadConsoleA@20 ;

PUSH 0; параметр: код выхода

CALL ExitProcess@4

MAIN ENDP; завершение описания процедуры с именем MAIN

END MAIN; завершение описания модуля с указанием первой выполняемой процедуры

**Набор тестов**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Первое число | Второе число | Результат | Назначение |
| 1 | 333333 | 77777 | 3681274149 | Тест большой размерности |
| 3 | 4531 | -6523 | -8162523 | Одно из чисел отрицательно |
| 4 | -4531 | -6523 | 8162523 | Оба числа меньше нуля |
| 5 | 14567 | 4529 | Символ в числе не соответствует восьмеричной сс | Корректность входных данных |
| 6 | -752D3 | 1452 |
| 7 | 34 | 0 | 0 | Одно из чисел 0 |