|  |  |  |
| --- | --- | --- |
| Министерство науки и высшего образования  Российской Федерации | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования | | |
| «Новосибирский государственный технический университет» | | |
|  | | |
|  | | |
|  | | |
| Лабораторная работа № 1 | | |
| по дисциплине «Низкоуровневое программирование» | | |
|  | | |
| **кафедра теоретической и прикладной информатики** | | |
|  | | |
|  | Факультет: | ПМИ |
| Группа: | ПМИ-12 |
|  |  |
| Студенты: | Курдюков Иван,  Омельницкая Екатерина |
| Вариант: | 6 |
|  |  |
| Преподаватели: | Сивак Мария Алексеевна |
|  |  |
|  | | |
| Новосибирск | | |

2021

1. Цель работы

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

1. Ход работы
   1. Задача: вычитание, ввод в десятичной системе счисления (не менее 3 знаков), вывод в шестнадцатеричной системе счисления
   2. Алгоритм:
      1. Перекодирование строчек с инструкциями
      2. Вывод 1 инструкции
      3. Считывание 1 строки
      4. Проверка на количество символов
      5. Перевод считанной строки в десятичное число
      6. Вывод 2 инструкции
      7. Считывание 2 строки
      8. Проверка на количество символов
      9. Перевод считанной строки в десятичное число
      10. Вычитание чисел
      11. Перевод в 16 систему
      12. Вывод результата
   3. Текст разработанной программы

.386

.MODEL FLAT, STDCALL

; 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

STRN DB "Введите числа: ",13,10,0; 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

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

BUF2 DB 200 dup (?); буфер для вводимых/выводимых строк длиной 200 байтов

LENS1 DD ?; переменная для количества выведенных символов

LENS2 DD ?; переменная для количества выведенных символов

SIGN DD ?

SIGN2 DD ?

ZNAK DD ?

ZNAK2 DD ?

ZNAK\_RES DW ?

CNTR DD ?

AHAHA DD ?

FLAG DB ?

S\_16 DD 16

ERROR\_STR DB "Произошла ошибка в входных данных", 0

ERROR\_STR2 DB "Неверный размер числа", 0

.CODE; сегмент кода

MAIN PROC; описание процедуры

MOV EAX, OFFSET ERROR\_STR

PUSH EAX

PUSH EAX

CALL CharToOemA@8

MOV EAX, OFFSET ERROR\_STR2

PUSH EAX

PUSH EAX

CALL CharToOemA@8

; перекодируем строку STRN

MOV EAX, OFFSET STRN; командой MOV значение второго операнда

; перемещается в первый, OFFSET – операция, возвращающая адрес

PUSH EAX; параметры функции помещаются в стек командой PUSH

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 STRN; 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 LENS1; 4-й параметр

PUSH EAX; 3-й параметр

PUSH OFFSET STRN; 2-й параметр

PUSH DOUT; 1-й параметр

CALL WriteConsoleA@20

; ввод строки

PUSH 0; в стек помещается 5-й параметр

PUSH OFFSET LENS1; 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

; ввод строки

PUSH 0; в стек помещается 5-й параметр

PUSH OFFSET LENS2; 4-й параметр

PUSH 200; 3-й параметр

PUSH OFFSET BUF2; 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

;конвертация строки в 10-ное число

MOV DI, 10 ; основание системы счисления

DEC LENS1

DEC LENS1

MOV ECX, LENS1 ; 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

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

CMP BL, '-'

JNE PLUS

DEC LENS1

MOV ZNAK, 1

SUB BL, '-'

INC ESI

CMP LENS1, 3

JB ERROR2

CMP LENS1, 8

JA ERROR2

DEC ECX

JMP CONVERT

PLUS:

MOV ZNAK, 0

CMP LENS1, 3

JB ERROR2

CMP LENS1, 8

JA ERROR2

CONVERT: ; метка начала тела цикла

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

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

CMP BL, 0

JB ERROR

CMP BL, 9

JA ERROR

MUL DI ; умножить значение AX на 10, результат – в AX

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

INC ESI ; перейти на следующий символ строки

LOOP CONVERT ; перейти на следующую итерацию цикла

CWDE

MOV CNTR, EAX

; В CNTR ЛЕЖИТ НАШЕ 10-НОЕ ЧИСЛО

;конвертация строки в 10-ное число

DEC LENS2

DEC LENS2

MOV ECX, LENS2 ; счетчик цикла (строка имеет длину LENS)

MOV ESI, OFFSET BUF2 ; начало строки хранится в переменной BUF2

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

XOR AX, AX ; обнулить регистр AX

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

CMP BL, '-'

JNE PLUS1

DEC LENS2

MOV ZNAK2, 1

SUB BL, '-'

INC ESI

CMP LENS2, 3

JB ERROR2

CMP LENS2, 8

JA ERROR2

DEC ECX

JMP CONVERT2

PLUS1:

MOV ZNAK2, 0

CMP LENS2, 3

JB ERROR2

CMP LENS2, 8

JA ERROR2

CONVERT2: ; метка начала тела цикла

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

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

CMP BL, 0

JB ERROR

CMP BL, 9

JA ERROR

MUL DI ; умножить значение AX на 10, 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

; перейти на следующую итерацию цикла

; B AX ЛЕЖИТ НАШЕ 10-НОЕ ЧИСЛО

MOV EBX, CNTR

MOV ECX, ZNAK

MOV EDX, ZNAK2

CMP EBX, EAX ; В EBX ПЕРВОЕ ЧИСЛО B EAX ВТОРОЕ

JA FIRST\_BIGGER

JB SEC\_BIGGER

JE EQUAL2

FIRST\_BIGGER:

CMP ECX, EDX

JA NEGATIVE11

JB NEGATIVE22

JE EQUAL123

NEGATIVE11:

CMP FLAG, 1

JE SEB

JNE SEB97

SEB:

MOV ZNAK\_RES, 0

ADD EBX, EAX

JMP EXIT000

SEB97:

MOV ZNAK\_RES, 1

ADD EBX, EAX

JMP EXIT000

NEGATIVE22:

CMP FLAG, 1

JE SEB2

JNE SEB98

SEB2:

MOV ZNAK\_RES, 1

ADD EBX, EAX

JMP EXIT000

SEB98:

MOV ZNAK\_RES, 0

ADD EBX, EAX

JMP EXIT000

EQUAL123:

CMP ECX, 1

JE METKA1

JNE METKA2

METKA1:

CMP FLAG, 1

JE SEB3

JNE SEB99

SEB3:

MOV ZNAK\_RES, 0

SUB EBX,EAX

JMP EXIT000

SEB99:

MOV ZNAK\_RES, 1

SUB EBX,EAX

JMP EXIT000

METKA2:

CMP FLAG, 1

JE SEB4

JNE SEB100

SEB4:

MOV ZNAK\_RES, 1

SUB EBX, EAX

JMP EXIT000

SEB100:

SUB EBX, EAX

MOV ZNAK\_RES, 0

JMP EXIT000

SEC\_BIGGER:

MOV FLAG, 1

MOV EDI, EAX ;МЕНЯЕМ МЕСТАМИ З

MOV EAX, EBX

MOV EBX, EDI

MOV EDI, ECX

MOV ECX, EDX

MOV EDX, EDI

JMP FIRST\_BIGGER

EQUAL2:

MOV EBX, 0

MOV ZNAK\_RES, 0

EXIT000:

XOR ESI, ESI

SIXTEEN:

MOV ESI, OFFSET BUF

;MOV AHAHA, EAX

XOR ECX, ECX

CMP ZNAK\_RES, 1

JNE FUNC

PUSH EAX

MOV EAX, '-'

MOV [ESI], EAX

POP EAX

INC ESI

FUNC:

MOV EAX, EBX

XOR EDX, EDX

XOR EDI, EDI

CONVERT\_FROM10TO16:

CMP EBX, S\_16

JAE FUNC1

JB FUNC5

FUNC1:

DIV S\_16

ADD DX, '0'

CMP DX, '9'

JA FUNC2

JBE FUNC3

FUNC2:

ADD DX, 7

FUNC3:

PUSH EDX ; кладем данные в стек, для инвертирования

ADD EDI, 1

XOR EDX, EDX

XOR EBX,EBX

MOV BX, AX

MOV ECX, 2

LOOP CONVERT\_FROM10TO16

FUNC5:

ADD AX, '0'

CMP AX, '9'

JA FUNC6

JBE FUNC7

FUNC6:

ADD AX, 7

FUNC7:

PUSH EAX ; кладем данные в стек, для инвертирования

ADD EDI, 1

MOV ECX, EDI

CONVERTS:

POP [ESI]

INC ESI

LOOP CONVERTS

PUSH OFFSET BUF

CALL lstrlenA@4

PUSH 0

PUSH OFFSET LENS1

PUSH EAX

PUSH OFFSET BUF

PUSH DOUT

CALL WriteConsoleA@20

PUSH 0

CALL ExitProcess@4

PUSH 0

CALL ExitProcess@4

ERROR:

PUSH OFFSET ERROR\_STR

CALL lstrlenA@4

PUSH 0

PUSH OFFSET LENS1

PUSH EAX

PUSH OFFSET ERROR\_STR

PUSH DOUT

CALL WriteConsoleA@20

PUSH 0

CALL ExitProcess@4

ERROR2:

PUSH OFFSET ERROR\_STR2

CALL lstrlenA@4

PUSH 0

PUSH OFFSET LENS1

PUSH EAX

PUSH OFFSET ERROR\_STR2

PUSH DOUT

CALL WriteConsoleA@20

PUSH 0

CALL ExitProcess@4

MAIN ENDP

END MAIN

* 1. Тесты

|  |  |  |
| --- | --- | --- |
| Входные данные | Выходные данные | Комментарий |
| 123  456 | -14D | Корректное вычитание положительных чисел |
| -123  456 | -243 | Корректное вычитание чисел с разными знаками |
| 1384571893475  12123 | Ошибка | Неверный размер числа |
| -111  -123 | C | Корректное вычитание отрицательных чисел |
| -123  124326 | -E621 | Результат вычитание чисел с разными знаками |
| 235/Q  -123 | Ошибка | Произошла ошибка в входных данных |