МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**“УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”**

Факультет информационных систем и технологий

Кафедра “ Вычислительная техника”

Дисциплина “Машинно-ориентированное программирование”

**Лабораторная работа №1**

«Основы технологии программирования на ассемблере»

Вариант 3

Выполнил:

студент гр. ИВТАПбд-31

Вершинин Д. В.

Проверила:

Лылова А. В.

Ульяновск, 2019

1. **Цель работы**

Приобретение умений и навыков выполнения основных технологических операций, используемых при программировании на языке ассемблера.

1. **Техническое задание**

Использовать можно только язык TASM и ОС DOS.

Требуется добавить в программу функции ввода двух двадцатиразрядных двоичных чисел, вычисление заданного выражения и вывода результата.

Все умножения и деления на константы, кратные степени двойки, необходимо выполнять командами сдвига.

Можно использовать только машинные команды, работающие с 8 и 16-разрядными операндами.

Задание по варианту (№ 3):

Z = ? X / 2 + 2 \* Y : X \* 4 - Y; |= , = , &= ;

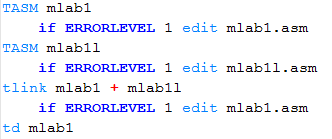
= | | | | .

Сперва необходимо произвести ввод 20-разрядных двоичных чисел X и Y, затем получить значения , где 1- 4 - это номер бита. После чего высчитать . Далее в зависимости от значения посчитанной функции вычислить число Z и изменить в нем некоторые разряды.

1. **Ход работы**

Циклический процесс “редактирование-ассемблирование-компоновка-выполнение” состоит из следующих этапов.

Изначально берутся файлы-исходники с кодом (mlab1.asm, mlab1l.asm, mlab1.inc, mlab1.mac), после чего создается bat-файл, содержащий набор команд, предназначенных для исполнения командным интерпретатором.



С помощью команды TASM ассемблируются файлы mlab1.asm и mlab1l.asm, находящиеся в папке, после чего с помощью команды TLINK происходит процесс компоновки этих файлов. После чего только что созданный файл mlab1.exe. открывается в турбо дебаггере. В случае ошибок происходит вывод соответствующего сообщения и открытие исходного кода программы в эдиторе. Таким образом реализуется циклический процесс «редактирование – ассемблирование – компоновка – выполнение».

Сравним размеры .exe и .obj файлов при ассемблировании с использованием параметров (zi – информация об идентификаторах для отладки - полная, la – генерация листинга - расширенная) и при компоновке с ключами m и v.

|  |  |
| --- | --- |
| Название файла | Размер файла (КБ) |
| Без ключей | |
| MLAB1.EXE | 2КБ |
| MLAB1.OBJ | 2КБ |
| MLAB1L.OBJ | 1КБ |
| С ключами zi, la для TASM, m, v для TLINK | |
| MLAB1.EXE | 5КБ |
| MLAB1.OBJ | 3КБ |
| MLAB1L.OBJ | 2КБ |

Видно, что при использовании ключей размеры .exe и .obj файлов увеличиваются.

Основной код программы храниться в файле mlab1.asm. Для считывания 20 разрядных чисел использовалась процедура GETS, уже написанная ранее и хранящаяся в файле MLAB1L.ASM. Функция считалась следующим образом: происходило обращение к определенному элементу строки, затем отнимая от полученного значения 30h получалось число, находящееся в данном разряде. Затем с помощью логических операторов OR и AND высчитывалось значение функции. Умножение исходных чисел проводилось с использованием сдвига всех битов операнда влево (команда SHL), а деление с использованием сдвига вправо (команда SHR). Изменение разрядов в числе Z проводилось с помощью установки определенного бита в 1 или 0 (команды BTS и BTR соответственно).

1. **Код программы на языке ассемблера**

|  |
| --- |
| ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ; MLAB1.ASM - учебный пример для выполнения  ; лабораторной работы N1 по машинно-ориентированному программированию  ; 10.09.02: Негода В.Н.  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  .MODEL SMALL  .STACK 200h  .386  ; Используются декларации констант и макросов  INCLUDE lab1\MLAB1.INC  INCLUDE lab1\MLAB1.MAC  ; Декларации данных  .DATA  SLINE DB 78 DUP (CHSEP), 0  REQ DB "Введите число: ",0FFh  EMPTYS DB 0  BUFLEN = 20  BUF DB BUFLEN  LENS DB ?  NUM1 DB BUFLEN DUP (0)  BUF2 DB BUFLEN  LENS2 DB ?  NUM2 DB BUFLEN DUP (0)  RESULT DB BUFLEN DUP (0)  FUNC DB "f = ",0FFh  RES DB "Z = ",0FFh  PAUSE DW 0, 0 ; младшее и старшее слова задержки при выводе строки  ;========================= Программа =========================  .CODE  ; Макрос заполнения строки LINE от позиции POS содержимым CNT объектов,  ; адресуемых адресом ADR при ширине поля вывода WFLD  BEGIN LABEL NEAR  ; инициализация сегментного регистра  MOV AX, @DATA  MOV DS, AX    @@L: ; циклический процесс повторения вывода заставки  PUTLS REQ ; запрос имени  LEA DX, BUF  CALL GETS    MOV AX, 03  INT 10h    PUTLS REQ ; запрос имени  LEA DX, BUF2  CALL GETS  ;MOV SI, 2  ;MOV AL, SNAME[SI]  ;CALL PUTC    PUTLSC NUM1 ; ФИО студента  PUTL EMPTYS  PUTLSC NUM2  ; разделительная черта  PUTL SLINE    XOR CX, CX  XOR AX, AX    MOV BL, [LENS]  SUB BL, 3 ;ЗДЕСЬ ПОДСТАВИТЬ НОМЕР БИТА + 1  MOV AL, NUM1[BX]  SUB AL, 30h  MOV BL, [LENS]  SUB BL, 5  MOV AH, NUM1[BX]  SUB AH, 30h  AND AH, AL    MOV CH, AH    XOR AX, AX  MOV BL, [LENS]  SUB BL, 2  MOV AL, NUM1[BX]  SUB AL, 30h  MOV BL, [LENS]  SUB BL, 4  MOV AH, NUM1[BX]  SUB AH, 30h  XOR AH, 1  AND AH, AL    OR CH, AH    XOR AX, AX  MOV BL, [LENS]  SUB BL, 2  MOV AL, NUM1[BX]  SUB AL, 30h  XOR AL, 1  MOV BL, [LENS]  SUB BL, 4  MOV AH, NUM1[BX]  SUB AH, 30h  AND AH, AL  MOV BL, [LENS]  SUB BL, 5  MOV CL, NUM1[BX]  SUB CL, 30h  XOR CL, 1  AND AH, CL  XOR CL, CL    OR CH, AH    MOV BL, [LENS]  SUB BL, 4  MOV AL, NUM1[BX]  SUB AL, 30h  MOV BL, [LENS]  SUB BL, 5  MOV AH, NUM1[BX]  SUB AH, 30h  AND AH, AL    OR CH, AH    XOR AX, AX  MOV BL, [LENS]  SUB BL, 2  MOV AL, NUM1[BX]  SUB AL, 30h  XOR AL, 1  MOV BL, [LENS]  SUB BL, 4  MOV AH, NUM1[BX]  SUB AH, 30h  AND AH, AL  MOV BL, [LENS]  SUB BL, 3  MOV CL, NUM1[BX]  SUB CL, 30h  XOR CL, 1  AND AH, CL    ;значение функции f хранится в CH  OR CH, AH      PUTLS FUNC  MOV AL, CH  ADD AL, 30h  CALL PUTC  PUTL EMPTYS  MOV BX, 0  ;XOR CX, CX  LEA SI, NUM1  JMP str2dw    str2dw:  xor ax, ax  xor dx,dx ;сумма  @lp1: lodsb ;берем cимвол  test al,al ;если это нулев байт, то заканчиваем  jz @ex  sub al,'0' ;получаем цифровое значение  cmp al,9  ja @lp1  imul edx,2 ;умножаем сумму на 2  add edx, eax ;прибавляем текущее значение  jmp @lp1  @ex: mov ebx,edx  LEA SI, NUM2  str2dw2:  xor ax, ax  xor dx,dx ;сумма  @lp2: lodsb ;берем cимвол  test al,al ;если это нулев байт, то заканчиваем  jz @ex2  sub al,'0' ;получаем цифровое значение  cmp al,9  ja @lp2  imul edx,2 ;умножаем сумму на 2  add edx, eax ;прибавляем текущее значение  jmp @lp2  @ex2: mov eax,edx    CMP CH, 1  JE @CALC1  JMP @CALC2    @CALC1:  SHR EBX, 1  SHL EAX, 1  ADD EBX, EAX  JMP @MTK  @CALC2:  SHL EBX, 2  SUB EBX, EAX    @MTK:  BT EBX, 3  JNC CF0  OR EBX, 40H  CF0:  BT EBX, 19  CMC  JNC S0  JMP S1  S1:BTS EBX, 18  JMP @M  S0:BTR EBX, 18  @M:  BT EBX, 8  JC @M1  BTR EBX, 7  @M1:  ;ВЫВОД  PUTLS RES  shl ebx, 12  mov cx, 20 ; переменная цикла равна количеству бит в слове  @@For:  mov ax, '0' ; в регистрах al=код символа '0', ah=00h  shl ebx, 1 ; выделение бита  adc al, ah ; сложение кода символа '0' со значением выделенного бита  int 29h  loop @@For  JMP CEXIT  BACK: JMP @@L  CEXIT:  CALL GETCH  CMP AL, CHESC  JE @@E  TEST AL, AL  JNE BACK  CALL GETCH  JMP @@L  ; Выход из программы  @@E: EXIT  EXTRN PUTSS: NEAR  EXTRN PUTC: NEAR  EXTRN GETCH: NEAR  EXTRN GETS: NEAR  EXTRN SLEN: NEAR  EXTRN UTOA10: NEAR  END BEGIN |

1. **Контрольные примеры работы программы**

|  |  |  |
| --- | --- | --- |
| № | Входные данные | Результат работы программы |
| 1 | Введите X: 101010  Введите Y: 11010 |  |
| 2 | Введите X: 10101100101100  Введите Y: 101011 |  |
| 3 | Введите X: 10101010111001  Введите Y: 1011110111110 |  |