МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Кафедра информационных технологий**

**Отчет о выполнении лабораторной работы №1**

**по дисциплине «Системы реального времени»**

Работу выполнил студент 45/2 группы А.А. Козин

Руководитель

доц. каф. ИТ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.Н. Полетайкин

Краснодар

2023

**ЛАБОРАТОРНАЯ РАБОТА 1**

**ПРЕДСТАВЛЕНИЕ ЦЕЛЫХ ЧИСЕЛ В ПАМЯТИ ЭВМ**

Цель: изучение принципов представления числовой информации в памяти ЭВМ; приобретение практических навыков представления целых чисел со знаком в машинном формате.

Задание:  
 1. Представить целые числа 70 и -2504 в формате DB, DW, DD.

2. Составить и откомпилировать программу, определив число 70 в форматах DB, DW, DD, а число -2504 в форматах DW, DD.

3. Задать такие операции пересылки данных:

– загрузить регистр BX числом -2504 из сегмента данных;

– с использованием заданного варианта косвенной адресации([BP][DI]) записать содержимое BX в сегмент данных со смещением на I1 байт относительно метки I2 (число 70 предварительно загрузить в соответствующий базовый или индексный регистр, при наличии других операндов в формуле адресации определить их произвольно).

4. Проверить результаты расчетов и пересылок в дампе памяти.

5. Сделать расчет времени выполнения программы (методика расчета представлена в прил. А).

**ХОД РАБОТЫ**

Задание 1.

Для представления целых чисел 70 и -2504 в формате DB, DW, DD нужно выполнить алгоритм, состоящий из 4 для положительных и 5 операций для отрицательных.

|  |  |  |
| --- | --- | --- |
| Операции | Формат DB | Формат DW |
| 1) модуль в двоичной форме | 1000110 | 1000110 |
| 2) дописываем 0 | 0100 0110 | 0000 0000 0100 0110 |
| 3) в 16-ричном виде | 46 | 00 46 |
| 4) в дампе памяти | 46 | 46 00 |
| Формат DD | | |
| 1) модуль в двоичной форме | 1000110 | |
| 2) дописываем 0 | 0000 0000 0000 0000 0000 0000 0100 0110 | |
| 3) в 16-ричном виде | 00 00 00 46 | |
| 4) в дампе памяти | 46 00 00 00 | |

Рисунок 1 – перевод числа 70 в 16-ый код.

|  |  |
| --- | --- |
| Операции | Формат DW |
| 1) модуль в двоичной форме | 100111001000 |
| 2) дописываем 0 | 0000 1001 1100 1000 |
| 3) инвертируем и прибавляем 1 | 1111 0110 0011 1000 |
| 3) в 16-ричном виде | F6 38 |
| 4) в дампе памяти | 38 F6 |
| Формат DD | |
| 1) модуль в двоичной форме | 100111001000 |
| 2) дописываем 0 | 0000 0000 0000 0000 0000 1001 1100 1000 |
| 3) инвертируем и прибавляем 1 | 1111 1111 1111 1111 1111 0110 0011 1000 |
| 3) в 16-ричном виде | FF FF F6 38 |
| 4) в дампе памяти | 38 F6 FF FF |

Рисунок 2 – од числа -2504 в 16-й код.

Задание 2.

Используя полученные знания из курса по параллельному и низкоуровневому программированию, составим и откомпилируем программу с архитектурой ARM NASM x64.

Для компиляции программы был использован терминал iTerm.

nasm -g -felf64 asm.s

После этой команды будет создан файл asm.o.

Код программы:

section .data

i1db db 70

i1dw dw 70

i1dd dd 70

i2dw dw -2504

i2dd dd -2504

section .text

global \_start

\_start:

debug

int 0x80

Задание 3.

Для загрузки в базовый регистр BX, который используется для указания базового (начального) адреса объекта данных в памяти, была использована инструкция MOV, которая копирует данные.

Код программы:

section .data

i1db db 70

i1dw dw 70

i1dd dd 70

i2dw dw -2504

i2dd dd -2504

section .text

global \_start

\_start:

mov rbp, rsp ;debug

mov ecx, [i2dd]

mov ebx, i2dd

mov edi, [i1dd]

mov [ebp+edi], ecx

mov eax, 1

int 0x80

Задание 4

Для проверки результатов расчетов и пересылок была использована утилита hexdump и RadAsm.

hexdump — это утилита командной строки в операционных системах Unix и подобных им, которая позволяет просматривать содержимое файла в шестнадцатеричном (hexadecimal) формате, а также в других представлениях, таких как восьмеричный и символьный.

Чтобы посмотреть memory dump нужно использовать команды:

hexdump asm.o | egrep ‘ff|46|38’

A screen shot of a computer

Description automatically generated

Рисунок 3 – memory dump.

В RadAsm:

A screenshot of a computer

Description automatically generated

Рисунок 4 – memory dump before run.

A screenshot of a computer

Description automatically generated

Рисунок 5 – memory dump after run.

Задание 5.

Для подсчета времени выполнения программы использовалась методика из приложения А.

MOV регистр, регистр – 2

MOV регистр, память – 13

MOV регистр, память – 13

MOV регистр, память – 13

MOV регистр, память – 12 + 11 = 23

Время = 2 + 13 \* 3 + 23 = 64 такта.

Время работы = 64 \* 210 =13 440 (наносекунд)

**РЕЗУЛЬТАТЫ**

Для задания 1 было сделано 2 рисунка с переводом целых положительных и отрицательных чисел в формат DB, DW, DD. Для этого использовался алгоритм, предложенный в учебном пособии.

В задании 2 были изучены типы данных и базовая структура программы для компиляции.

В задании 3 был написан код внутри функции main для работы с сегментами данных и косвенной речью.

Для задания 4 использовался RadAsm dump до и после компиляции и утилита от hexdump, которая позволила в терминале узнать результаты расчетов и пересылок.

В задании 5 нужно было сделать расчет времени выполнения программы, для этого использовалось учебное пособие, а именно приложение А и его таблицы.

**ВЫВОДЫ**

В итоге, была написана программа для выполнения задания из лабораторной работы №1. Для написания программы были изучены РОН, флаги, принцип работы памяти и способ представления информации в ней.