МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №8
по дисциплине «Организация ЭВМ и систем»
Тема: Обработка вещественных чисел. Программирование математического сопроцессора.

Студент гр. 9382 _____ Кодуков А.В.

Преподаватель _____ Ефремов М.А .

Санкт-Петербург
2020

Задание:

Разработать подпрограмму на языке Ассемблера, обеспечивающую вычисление заданной математической функции с использованием математического сопроцессора. Подпрограмма должна вызываться из головной программы, разработанной на языке С. При этом должны быть обеспечены заданный способ вызова и обмен параметрами. Альтернативный вариант реализации: разработать на языке Ассемблера фрагмент программы, обеспечивающий вычисление заданной математической функции с использованием математического сопроцессора, который включается по принципу in-line в программу, разработанную на языке С.

Вариант 5:

Name ldexp - calculates value * 2^exp

Usage double ldexp(double value, int exp);

Prototype in math.h

Description ldexp calculates value * 2^exp

Выполнение работы:

Команды:

fld – загрузка операнда в вершину стека

fscale – масштабирование по степени 2

fstp - Сохранение вершины стека в память с выталкиванием

Тесты:

```
Enter x: 0.95
Enter pow: 4
math.h ldexp(x): 15.200000
asm Ldexp(x): 15.200000
```

```
Enter x: 0
Enter pow: 10
math.h ldexp(x): 0.000000
asm Ldexp(x): 0.000000
```

```
Enter x: 0.111111
Enter pow: 8
math.h ldexp(x): 28.444416
asm Ldexp(x): 28.444416
```

Вывод:

В ходе выполнения работы были изучены принципы работы с математическим сопроцессором и создана программа, которая умножает число с плавающей запятой на целую степень числа два.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

func.asm

```
STACKSG SEGMENT  PARA STACK 'Stack'
  DW 1024 DUP(?)
STACKSG     ENDS

DATASG  SEGMENT  PARA 'Data'; SEG DATA
  KEEP_CS DW 0; для хранения сегмента
  KEEP_IP DW 0; и смещения вектора прерывания
  GREETING  DB 'Kodukov Aleksandr 9382 $'
  crlf db 0ah, 0dh, '$'
DATASG      ENDS; ENDS DATA

CODE SEGMENT; SEG CODE
ASSUME DS:DataSG, CS:Code, SS:STACKSG

INTER_TIMER PROC FAR

  PUSH AX; сохранение изменяемых регистров
  PUSH DX

  ; действия по обработке прерывания
  MOV AH, 9; вызов того,
  INT 21H; что хранится в dx

  MOV DX, OFFSET crlf
  MOV AH, 9
  INT 21H

  POP DX; восстановление регистров
  POP AX

  MOV AL, 20H
  OUT 20H, AL

IRET

INTER_TIMER  ENDP

Main PROC FAR
  MOV AX, DATASG; ds setup
  MOV DS, AX

  MOV AH, 35H; функция получения вектора
  MOV AL, 08H; номер вектора
  INT 21H
  MOV KEEP_IP, BX; запоминание смещения
  MOV KEEP_CS, ES; и сегмента вектора прерывания

  CLI
  PUSH DS
  MOV DX, OFFSET INTER_TIMER
  MOV AX, SEG INTER_TIMER; сегмент процедуры
  MOV DS, AX; помещаем в DS
  MOV AH, 25H; функция установки вектора
  MOV AL, 08H; номер вектора
  INT 21H; меняем прерывание
  POP DS
  STI
```

```
  MOV DX, OFFSET GREETING; помещаем строку в DS
  INT 08h

  CLI
  PUSH DS
  MOV DX, KEEP_IP
  MOV AX, KEEP_CS
  MOV DS, AX
  MOV AH, 25H
  MOV AL, 08H
  INT 21H; восстанавливаем старый вектор прерывания
  POP DS
  STI

  MOV AH, 4CH
  INT 21H

Main ENDP
CODE ENDS
END Main
```