Федеральное агентство связи

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

Кафедра Вычислительных Систем

Отчет по лабораторной работе

по дисциплине «Архитектура ЭВМ»

Лабораторная работа №2

«Разработка библиотеки mySimpleComputer. Оперативная память, регистр флагов, декодирование операций»

Выполнил: студент 2 курса группы ИП-811 Мироненко К. А

Проверила: Ткачева Т.А.

Оглавление

[1. Постановка задачи 3](#_Toc30598306)

[1.1 Общие положения 3](#_Toc30598307)

[1.2 Задание в соответствии с вариантом 3](#_Toc30598308)

[2. Описание иерархии объектов и методов объектов 4](#_Toc30598309)

[3. Описание алгоритма основной программы 10](#_Toc30598310)

[4. Примеры работы программы 11](#_Toc30598311)

[5. Выводы 14](#_Toc30598312)

[6. Список используемых источников 15](#_Toc30598313)

[*Приложение* Листинг 16](#_Toc30598314)

# Постановка задачи

*Цель работы*

Изучить принципы работы оперативной памяти. Познакомиться с разрядными операциями языка Си. Разработать библиотеку mySimpleComputer, включающую функции по декодированию команд, управлению регистрами и взаимодействию с оперативной памятью.

*Задание на лабораторную работу*

1. Прочитайте главу 4 практикума по курсу «Организация ЭВМ и систем». Изучите принципы работы разрядных операций в языке Си: как можно изменить значение указанного разряда целой переменной или получить его значение. Вспомните, как сохранять информацию в файл и считывать еѐ оттуда в бинарном виде.
2. Разработайте функции по взаимодействию с оперативной памятью, управлению регистром флагов и кодированию/декодированию команд:
   1. int sc\_memoryInit () – инициализирует оперативную память Simple Computer, задавая всем еѐ ячейкам нулевые значения. В качестве «оперативной памяти» используется массив целых чисел, определенный статически в рамках библиотеки. Размер массива равен 100 элементам.
   2. int sc\_memorySet (int address, int value) – задает значение указанной ячейки памяти как value. Если адрес выходит за допустимые границы, то устанавливается флаг «выход за границы памяти» и работа функции прекращается с ошибкой;
   3. int sc\_memoryGet (int address, int \* value) – возвращает значение указанной ячейки памяти в value. Если адрес выходит за допустимые границы, то устанавливается флаг «выход за границы памяти» и работа функции прекращается с ошибкой. Значение value в этом случае не изменяется.
   4. int sc\_memorySave (char \* filename) – сохраняет содержимое памяти в файл в бинарном виде (используя функцию write или fwrite);
   5. int sc\_memoryLoad (char \* filename) – загружает из указанного файла содержимое оперативной памяти (используя функцию read или fread);
   6. int sc\_regInit (void) – инициализирует регистр флагов нулевым значением;
   7. int sc\_regSet (int register, int value) – устанавливает значение указанного регистра флагов. Для номеров регистров флагов должны использоваться маски, задаваемые макросами (#define). Если указан недопустимый номер регистра или некорректное значение, то функция завершается с ошибкой.
   8. int sc\_regGet (int register, int \* value) – возвращает значение указанного флага. Если указан недопустимый номер регистра, то функция завершается с ошибкой.
   9. int sc\_commandEncode (int command, int operand, int \* value) – кодирует команду с указанным номером и операндом и помещает результат в value. Если указаны неправильные значения для команды или операнда, то функция завершается с ошибкой. В этом случае значение value не изменяется.
   10. int sc\_commandDecode (int value, int \* command, int \* operand) – декодирует значение как команду Simple Computer. Если декодирование невозможно, то устанавливается флаг «ошибочная команда» и функция завершается с ошибкой.
3. Оформите разработанные функции как статическую библиотеку. Подготовьте заголовочный файл для неё.

*Защита лабораторной работы*

Для защиты лабораторной работы необходимо подготовить программу, демонстрирующую использование созданной библиотеки функций (сборка программы с библиотекой, использование заголовочного файла, примеры вызовов каждой функции, проверка корректности работы функций при различных входных значениях).

# Описание алгоритма основной программы

Работа программы начинается с главного меню. В главном меню пользователь может начать игру – кнопка “Начать игру”, просмотреть блок с информацией – кнопка “Инфо”, либо выйти из игры – кнопка “Выход”. Также пользователю доступен выбор сложности игры (блок в нижнем правом углу).

Перемещения по пунктам меню осуществляется с помощью мышки, нажатием лев. кн. мыши осуществляется соответствующий выбор пункта меню.

При выборе “Инфо” окно, содержащее меню, сменяется окном с информацией (InfoWindow). Выход из окна осуществляется нажатием клавиши “Esc”.

В случае когда игрок начинает игру окно сменяется окном с игрой(GameWindow). Далее, выполняется соответствующая состоянию (gameState) логика игры “Морской бой”. Выход из окна осуществляется нажатием клавиши “Esc”.

При выборе пользователем пункта главного меню “Выход” прекращается отрисовка графического окна и происходит последующее закрытие программы.

# Примеры работы программы

# *Приложение* Листинг

**main.сpp**

**.cpp**

**.hpp**

**.сpp**

**.hpp**

**.cpp**

**.hpp**

**.cpp**

**.hpp**

**.cpp**

**.hpp**

**.cpp**

**.hpp**

**.cpp**

**.hpp**

**.cpp**

**.hpp**

**.cpp**

**.hpp**