УТВЕРЖДЕНО
Проректор по учебной работе
и довузовскому образованию
А. А. Воронов
09 января 2018 г.

ПРОГРАММА

по дисциплине: **Информатика. Продвинутый курс**
по направлению подготовки: 03.03.01 «Прикладные математика и физика»
физтех-школы: **ФФПК, ФАКТ, ФЭФМ**
факультеты: **ФОПФ, ФФКЭ, ФАЛТ**
кафедра: **информатики и вычислительной математики**
курс: 1
семестр: 2

Трудоёмкость:
базовая часть – 3 зачет. ед.
лекции – нет
практические (семинарские) занятия – нет
лабораторные занятия – 60 (час)

Экзамен – нет
Зачет дифф. – 2 семестр
Задания – 2
Контрольные работы – 2
Самостоятельная работа – 75 часов

ВСЕГО АУДИТОРНЫХ ЧАСОВ – 60

Программу составили: академик РАН, проф. В. П. Иванников
доцент, к.ф.-м.н. П. Н. Коротин
доцент, к.т.н. В. К. Хохлов
доцент, к.ф.-м.н. Д. С. Северов

Программа принята на заседании
кафедры информатики и вычислительной математики
14 ноября 2017 года

Заведующий кафедрой
чл.-корр. РАН, проф. И. Б. Петров

Введение. Структура ЭВМ

Уровни абстрактного представления ЭВМ, язык Ассемблера и машинные команды среди них. Элементы и контекст машинного представления информации. Трансляция и интерпретация программ и команд. Краткое описание устройств ЭВМ и схема их взаимодействия. Структура центрального процессора (ЦП). Регистры, арифметико-логическое устройство, устройство управления. Схема работы ЭВМ. Кэширование и иерархия устройств хранения. Оперативная память ЭВМ. Ячейки, адреса, машинные слова, разряды, биты. Двоичное представление информации в ЭВМ, причины выбора такого представления. Взаимодействие ЭВМ друг с другом. Одновременность и параллельность.

Представление информации в памяти ЭВМ

Двоичная система счисления. Шестнадцатеричная нотация. Слова и размеры данных. Представления целых чисел в форме с фиксированной точкой (представление беззнаковых чисел, представление знаковых чисел в прямом и дополнительном кодах). Особенности сложения и вычитания целых чисел. Флаги. Представление вещественных чисел в форме с плавающей точкой. Размещение числовых данных в памяти. Двоично-десятичные числа. Представление нечисловой информации.

Машинное представление программ

Кодирование программ. Форматы данных. Обращение к данным. Арифметические и битовые операции. Команды управления. Процедуры. Массивы. Неоднородные конструкции данных. Указатели. Использование отладчика. Некорректные ссылки и переполнение буфера. 64-битное расширение IA-32. Программы с плавающей точкой.

Архитектура процессора

Иллюстративная архитектура системы команд Y86. Способы задания операндов. Система команд как важнейшая характеристика ЭВМ. Разнообразие систем команд в реальных ЭВМ (CISC, RISC и др.). Понятие цифрового конструирования и язык управления аппаратурой. Последовательная реализация Y86. Основные принципы конвейеризации. Конвейерная реализация Y86.

Оптимизация программ

Возможности и ограничения оптимизирующих компиляторов. Измерение производительности программ. Исключение неэффективности циклов. Уменьшение количества вызовов процедур. Исключение ненужных ссылок в память. Понятие о современном процессоре. Разворачивание циклов. Увеличение степени параллелизма. Результат оптимизации кода. Ограничители производительности. Производительность памяти. Обнаружение и исключение мест потери производительности.

Иерархия памяти

Технологии хранения данных. Локальность. Иерархия видов памяти и принцип кэширования. Кэш-память. Создание кэш-ориентированных программ. Влияние кэш-памяти на производительность.

Программные сегменты

Особенности сегментирования (базирования) адресов в ПК. Префиксы сегментных регистров. Соглашения о выборе сегментных регистров по умолчанию. Описание программных сегментов, директива SEGMENT. Операторы SEG и OFFSET. Директива ASSUME и ее назначение. Начальная загрузка сегментных регистров. Директива INCLUDE. Типичная структура программы на ЯА. Модели памяти и упрощенные директивы сегментации.

Директивы описания переменных и констант

Директивы DB, DW, DD, DF, DQ, DT и EQU. Константные и адресные выражения. Операторы: TYPE, арифметические.

Использование CPU

Команды CPU. Основные приемы программирования с использованием CPU.

Динамические структуры данных

Строковые команды. Префиксы повторения. Строки переменной длины. Машинное представление списков. Основные операции над списками. Организация кучи (области для размещения списков), процедуры инициализации кучи, выделения и освобождения места в куче. Представление очередей и деревьев.

Использование FPU

Команды FPU. Основные приемы программирования с использованием FPU.

MMX технология

Регистры MMX. Типы данных MMX. Система команд MMX. Технология «одна команда много данных» (single-instruction multiple-data, SIMD). Арифметика с насыщением. Встроенные (intrinsic) средства поддержки MMX команд в языке C.

Потоковые SIMD расширения (SSE, SSE2, SSE3, SSSE3, SSE4)

Регистры XMM. Типы данных XMM. Система команд. Встроенные (intrinsic) средства поддержки потоковых SIMD расширений.

Макросредства

Предварительное преобразование текста программы; понятие макроязыка этапа макрогенерации и макрогенератора. Условное ассемблирование: назначение, IF-блоки, IF-директивы (IF, IFE, IFIDN, IFDIF, IFB, IFNB). Запись логических выражений (операторы отношений, логические операторы). Блоки повторения: назначение, REPT-, IRP- и IRPC-блоки. Макро-

операторы <>, & и !. Макросы: назначение, макроопределения, макрокоманды, макроподстановки, макрорасширения. Сравнение макросов и процедур. Директивы LOCAL и EXITM.

Многомодульные программы

Понятие модульного программирования, независимая трансляция модулей. Структура модуля. Внешние и общие имена, директивы EXTRN и PUBLIC; сегментирование внешних имен, доступ к ним. Объединение сегментов из разных модулей, параметры директивы SEGMENT. «Разноязычные» модули, соглашения о связях. Ассемблерные вставки.

Ввод-вывод. Прерывания

Машинные команды ввода-вывода (IN, OUT), порты. Использование механизма прерываний для контроля за событиями в ЭВМ, вектор прерывания INT. Примеры сервисных процедур ОС для ввода–вывода и построение на их основе операций ввода–вывода и останова, используемых в курсе.

Обработка текста программы. Задачи, решаемые современным редактором текстов программ.

Макрогенерация текста программы. Варианты взаимодействия макрогенератора с языком Ассемблера. Принципы обработки макросов, блоков повторения и директив условной трансляции.

Трансляция. Таблицы транслятора. Структура объектного модуля.

Компоновка. Соглашения компоновки. Структура загрузочного модуля.

Загрузка, запуск, выполнение и отладка. Основные действия. Управление процессом. Отладчик: основные возможности.

Контрольные вопросы и задания по базовой и вариативной части дисциплины для промежуточной аттестации по итогам освоения дисциплины

Задания

Если задача имеет варианты, то преподаватель может выбрать конкретный вариант для каждого студента. При сдаче заданий каждый студент для каждой задачи должен представить следующие материалы: (1) исходный текст программы, решающей задачу; (2) средства, достаточные для сборки исполняемого файла программы из исходного кода, если они отсутствуют на компьютере в аудитории. Также студент должен уметь объяснить исходные тексты, листинги и результаты представляемых работающих программ. Если явно не оговорено иное, то решения задач должны быть выполнены на языке программирования Си, при необходимости с ассемблерными вставками.

***Задание* 1**

(*срок сдачи 26–30 марта*)

1. Системы счисления

Напишите программы, выполняющие следующие преобразования.

1.1. Положительное десятичное целое число в двоичное.

1.2. Отрицательное десятичное целое число в двоичное (дополнительный код).

1.3. Двоичное целое число в шестнадцатеричное.

1.4. Шестнадцатеричное целое число в двоичное.

1.5. Десятичное вещественное число типа а) *float*, б) *double*, в) *long double* в двоичное представление: а) короткое, б) длинное, в) расширенное.

1.6. *Десятичное целое число в число, представленное в *BCD* кодировке: а) неупакованной, б) упакованной.

2. Директивы описания данных и сегментирования. Пересылки. Способы адресации

2.1. Напишите программу, содержащую объявления и инициализацию глобальных переменных базовых типов. Определите, во что отображаются эти действия в ассемблерном листинге. Укажите директивы сегментирования.

2.2. Напишите программу, которая в ассемблерном листинге содержит команды пересылки с непосредственной, прямой и косвенной адресацией.

3. Арифметические команды

3.1. Напишите программу, которая в ассемблерном листинге содержит команды сложения, вычитания, умножения и деления целых чисел.

3.2. Напишите программу, которая реализует вывод четверки из 0 и 1 – значений флагов *CF*, *ZF*, *SF*, *OF*, не изменяя текущего значения регистров, в том числе и регистра флагов.

3.3. Напишите программу, которая последовательно устанавливает значения флагов *CF*, *ZF*, *SF*, *OF* в 0 и 1, не изменяя текущего значения других флагов.

4. Команды переходов и цикла

4.1. Напишите программу, содержащую ветвление (*if – else*). Определите, во что отображается эта конструкция в ассемблерном листинге.

4.2. Напишите программу, содержащую множественное ветвление (*switch*). Определите, во что отображается эта конструкция в ассемблерном листинге.

4.3. Напишите программы, содержащие циклы: а) *for*, б) *while*, в) *do while*. Определите, во что отображаются эти конструкции в ассемблерном листинге.

5. Индексирование. Массивы. Структуры

5.1. Напишите программу, которая в заданном глобально одномерном массиве целых чисел находит наименьший элемент и выводит его на экран.

Определите, как производится индексирование элементов массива в ассемблерном листинге (индексная адресация).

5.2. Напишите программу, которая в заданном глобально двумерном массиве целых чисел находит наименьший элемент и выводит его на экран. Определите, как производится индексирование элементов массива в ассемблерном листинге (базово-индексная адресация).

5.3. Напишите программу, определяющую структурный тип *PERSON* (человек) со следующими тремя полями: *FAMILY* (фамилия), *NAME* (имя) и *AGE* (возраст). Объявите переменную структурного типа и произведите ее инициализацию. Определите, во что отображаются указанные выше действия в ассемблерном листинге.

6. Процедуры. Стек

6.1. На языке программирования Си напишите программу, в которой локально объявляются и инициализируются переменные базовых типов. Определите, во что отображаются эти действия в ассемблерном листинге.

6.2. На языке программирования Си напишите программу, содержащую функцию, которая производит суммирование массива целых чисел. Определите по ассемблерному листингу, как передаются параметры в эту функцию и как возвращается возвращаемое значение.

6.3. Напишите программу, содержащую функцию, которая возвращает указатель. Определите по ассемблерному листингу, как возвращается возвращаемое значение.

***Задание* 2**

(*срок сдачи 8–13 мая*)

1. Логические и сдвиговые команды

1.1. На языке программирования Си напишите программу, ассемблерный листинг которой содержит логические и сдвиговые команды.

2. Цепочечные команды

2.1. *На языке программирования Си напишите программы, ассемблерные листинги которых содержат цепочечные команды: а) пересылки цепочек (*movs*), б) сравнения цепочек (*cmps*), в) сканирования цепочки (*scas*), г) загрузки элемента из цепочки (*lods*), д) сохранения элемента в цепочке (*stos*).

3. Арифметический сопроцессор

3.1. На языке программирования Си напишите программу, содержащую функцию, которая производит суммирование массива вещественных чисел. Определите по ассемблерному листингу, как передаются параметры в эту функцию и как возвращается возвращаемое значение.

3.2. Напишите функцию возведения произвольного вещественного числа в произвольную степень x^y, используя команды сопроцессора *fyl2x* ($=y * \log_2 x$, $x \geq 0$, – в вершине стека, y – любое в регистре *ST*(1), результат

– после очистки *ST*(0) и *ST*(1) в вершине стека), *f*2*xm*1 (= $2^x - 1$, $-1 \le x \le 1$) и *fscale* (= 2^x, x – целое со знаком).

4. Многомодульные программы. Связь с языками высокого уровня

4.1. Напишите программу, состоящую из двух модулей. В головном модуле, написанном на языке Си, заданы массив целых чисел и массив вещественных чисел. Из модуля, написанного на языке Ассемблера, вызываются две функции: *sum*1, возвращающая значение суммы массива целых чисел, и функция *sum*2, возвращающая значение суммы массива вещественных чисел.

5. Макросредства

5.1. Напишите функцию, выводящую на экран значений флагов *CF*, *ZF*, *SF*, *OF*. Написать и использовать макроопределение *OUTF*, вызов которого приводит к генерации последовательности команд, реализующих вывод четверки из 0 и 1: значений флагов *CF*, *ZF*, *SF*, *OF*, и не изменяющего текущего значения регистров, в том числе и регистра флагов.

Индивидуальные задачи для программирования на языке Ассемблера

Каждому студенту выдается свой вариант задания.

Ко 2-му заданию. Обработка символьных строк

Постановка задачи

Дан текст (последовательность символов), содержащий не более 100 элементов. Признаком конца текста считается символ с кодом 0.

Требуется

- ввести текст с клавиатуры и записать его в память ЭВМ;
- определить, обладает ли этот текст заданным свойством, указанным в Вашем варианте задания;
- преобразовать текст по правилу 1 Вашего задания, если он обладает заданным свойством, и по правилу 2 в противном случае;
- вывести на экран исходный и преобразованный тексты, а также номер и формулировку примененного правила.

Варианты задания

Свойство исходного текста

1. Текст оканчивается заглавной латинской буквой, которая больше не встречается в тексте.
2. Текст начинается цифрой и оканчивается цифрой, причем эти цифры различны.
3. Текст начинается латинской буквой и оканчивается латинской буквой.
4. Текст содержит не менее трех латинских букв.
5. Текст содержит равное количество заглавных и строчных латинских букв.
6. Текст не содержит иных символов, кроме цифр и латинских букв.

Правило* 1 *преобразования текста

1. Заменить каждую заглавную латинскую букву следующей по алфавиту (*A*→*B*, *B*→*C*, …, *Z*→*A*).
2. Заменить каждую ненулевую цифру соответствующей ей по порядковому номеру строчной буквой латинского алфавита (1→*a*, 2→*b* и т.д.).
3. Заменить каждую заглавную латинскую букву цифрой, числовое значение которой равно остатку от деления порядкового номера буквы в алфавите на десять.
4. Заменить каждую строчную латинскую букву соответствующей заглавной буквой (*a*→*A*, *b*→*B*, …, *z*→*Z*).
5. Заменить каждую заглавную латинскую букву соответствующей строчной буквой (*A*→*a*, *B*→*b*, …, *Z*→*z*).
6. Заменить каждую заглавную латинскую букву «симметричной» в алфавите (*A*→*Z*, *B*→*Y*, …, *Z*→*A*).

Правило* 2 *преобразования текста

1. Перенести в начало текста все входящие в него цифры с сохранением порядка их следования.
2. Перевернуть текст, не используя дополнительную память.
3. Повторить каждый символ текста.
4. Удалить из текста все повторные вхождения его первого символа.
5. Оставить в тексте только те символы, которые входят в него ровно один раз.
6. В каждой группе следующих подряд одинаковых символов оставить только один.

Требования к программе

1. Вывод исходного текста должен быть выполнен сразу после ввода, до анализа и преобразования.
2. Вывод преобразованного текста должен быть выполнен только после завершения преобразования.
3. Алгоритмы преобразования по правилам 1 и 2 должны быть оформлены в виде подпрограмм.
4. Программа должна сохранять работоспособность при любых входных данных.

Литература

Основная

1. *Северов Д.С.* Лекции по архитектуре ЭВМ и языку Ассемблера. http://cs.mipt.ru
2. *Коротин П.Н.* Лекции по архитектуре ЭВМ и языку Ассемблера. http://cs.mipt.ru
3. *Пильщиков В.Н.* Программирование на языке Ассемблера IBM PC. – М: Диалог-МИФИ, 2000.

4. *Митницкий В.Я.* Архитектура IBM PC и язык Ассемблера: учеб. пособие. – М.: МФТИ, 2000.

Дополнительная

1. *Брайант Р., О`Халарон Д.* Компьютерные системы: архитектура и программирование. – СПб.: БХВ-Петербург, 2005.
2. *Хеннесси Дж.Л., Паттерсон Д.А.* Компьютерная архитектура. Количественный подход. Изд. 5-е. – М.: ТЕХНОСФЕРА, 2016.
3. *Ирвин К.Р.* Язык ассемблера для процессоров Intel / пер. с англ. – 4-е изд. – М.: Издательский дом «Вильямс», 2005.
4. *Корнеев В.В., Киселев А.В.* Современные микропроцессоры. – 3-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2003.
5. *Зубков С.В.* Assembler для DOS, Windows и UNIX. – М.: ДМК, 2003.
6. *Юров В.И.* Assembler: учебник для вузов. – 2-е изд. – СПб.: ПИТЕР, 2008.
7. *Юров В.И.* Assembler. Специальный справочник. – 2-е изд. – СПб.: ПИТЕР, 2004.
8. *Irvine K.R.* Assembly language for x86 processors. – 7 edition. – Pearson, 2015.

Электронные ресурсы, включая доступ к базам данных

1. http://judge.mipt.ru
2. http://cs.mipt.ru.
3. http://acm.mipt.ru.
4. Intel 64 and IA-32 Architectures Software Developer's Manual. V. 1–5. http://www.intel.com/products/processor/manuals/index.htm