МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Невинномысский технологический институт (филиал)

Методические указания по выполнению лабораторных работ по дисциплине «Компьютерная и микропроцессорная техника в электроприводе»

Направление подготовки 13.03.02 – Электроэнергетика и электротехника
Профиль подготовки – Электропривод и автоматика
Квалификация выпускника – бакалавр

Невинномысск 2019

Методические указания предназначены для проведения лабораторных работ по дисциплине «Компьютерная и микропроцессорная техника в электроприводе» для студентов направления подготовки 13.03.02 «Электроэнергетика и электротехника» и соответствуют требованиям ФГОС ВО направления подготовки бакалавров.

Составитель: Д.В. Самойленко

Содержание

1. Общие сведения об элементах микропроцессорных систем.....5
2. Комплект лабораторного оборудования по электронной технике типа К32 35
Лабораторная работа №1 37
Лабораторная работа №2 41
Лабораторная работа №3 43
Лабораторная работа №4 46
Лабораторная работа №5 52

Условные обозначения, используемые в методических указаниях:

ОЗУ – оперативное запоминающее устройство;

АЛУ – арифметико-логическое устройство;

АЦП – аналого-цифровой преобразователь;

ЦАП – цифро-аналоговый преобразователь;

ПК – программатор кодов;

ЗУ – запоминающее устройство;

БУК – блок управления комплектом;

МЗР – младший значимый разряд;

СЗР – старший значимый разряд;

УС – устройство сменное;

ИС – интегральная микросхема.

ПСИ – программатор серии импульсов;

ПК – программатор кодов;

ГПИ – генератор прямоугольных импульсов;

ОУ – операционный усилитель;

ЦОС – цепь отрицательной обратной связи;

ИОН – источник опорного напряжения.

1. ОБЩИЕ СВЕДЕНИЯ ОБ ЭЛЕМЕНТАХ МИКРОПРОЦЕССОРНЫХ СИСТЕМ

1.1. Оперативное запоминающее устройство на примере ИС типа К155РУ2.

Функциональные возможности любой ЭВМ существенно зависят от объемов и структуры ее запоминающих устройств, или, иначе говоря, от памяти машины. Память современных вычислительных машин представляет собой комбинацию вариантов запоминающих устройств: магнитных накопителей на различных носителях (лента, жесткий и мягкий диск, компакт-кассета и т.д.), полупроводниковых запоминающих устройств (ЗУ).

Цифровые ЗУ предназначены для записи, хранения и выдачи информации, представленной в виде цифрового кода.

Основными характеристиками ЗУ являются их информационная емкость, быстродействие и время хранения информации.

По выполняемым функциям различают следующие типы ЗУ: оперативные запоминающие устройства (ОЗУ, или RAM); постоянные запоминающие устройства (ПЗУ, или ROM); программируемые запоминающие устройства (ППЗУ, или PROM); репрограммируемые ПЗУ с возможностью многократного электрического перепрограммирования (РПЗУ, или EEPROM); РПЗУ с ультрафиолетовым стиранием и электрической записью информации (РПЗУ УФ или EPROM). Кроме того, существуют ассоциативные запоминающие устройства (АЗУ или CAM) – «безадресные» ЗУ, в которых поиск и выборка информации осуществляется по содержанию произвольного количества разрядов, хранящихся в АЗУ чисел, независимо от физических координат ячеек памяти.

ОЗУ – устройство памяти цифровой информации, объединенные со схемами управления, обеспечивающими режимы записи, хранения и считывания цифровой (двоичной) информации в процессе ее обработки. По способу хранения информации ЗУ делятся на статические и динамиче-

ские. В динамических ЗУ для хранения информации используют инерционные свойства реактивных элементов (например, емкости затвор-исток), что требует периодического восстановления (регенерации) состояния элементов памяти в процессе хранения информации. Элементы памяти статических ЗУ представляют собой бистабильные ячейки, что определяет потенциальный характер управляющих сигналов и возможность чтения информации без ее разрушения. Статические ЗУ бывают синхронными и асинхронными. Синхронные статические ЗУ имеют статический накопитель (матрицу элементов памяти) и динамические цепи управления, требующие синхронизации аналогично динамическим ЗУ.

Все параметры ЗУ также можно разделить на статические и динамические. Статические параметры ЗУ характеризуют его работу в установившемся режиме. Система статических параметров ЗУ представляет собой совокупность некоторых контрольных точек его вольт-амперной характеристики. Динамические параметры ЗУ определяются происходящими в нем временными процессами. Систему динамических параметров ЗУ составляет совокупность временных переходов входных и выходных сигналов, соответствующих границам правильного функционирования ЗУ. Структурная схема статического ОЗУ приведена на рисунке 1.

Рисунок 1 – Структурная схема статического ОЗУ

В данных методических указаниях рассмотрим статическое ОЗУ в интегральном исполнении серии К155РУ2. Микросхема РУ2 является

ОЗУ со схемами управления и полной дешифрацией адреса. Общий объем памяти 64 бита (организация – 16×4-разрядных слов). Логическая структура микросхемы приведена на рисунке 2.

Микросхема включает матрицу из 64 ячеек памяти, 4 усилителя записи-считывания, стробируемый дешифратор 4-разрядного двоичного кода адреса, выбирающий одно из 16 4-разрядных слов. Вход разрешения выборки несет функцию запрета при подаче на него напряжения высокого уровня.

В этом случае нельзя произвести ни запись в запоминающую ячейку, ни считывание их нее. Запись информации осуществляется прямым кодом параллельно по четырем информационным входам D1-D4 по требуемому адресу (рисунок 2). На входы ИС разрешения выборки и разрешения записи должен быть подан низкий уровень. Работа ИС описывается таблице 1.

Таблица 1 – Работа ИС К155РУ2

CS	W	D	Режим работы	C
1	1	0	Хранение	1
0	1	0	Считывание	Инверсный код выбранного числа
0	0	1	Запись 1	0
0	0	0	Запись 0	1
1	0	1	Запрет	0
1	0	0	Обращения	1

Считывание информации осуществляется при подаче на входы адреса A1-A4 прямого кода и на вход разрешения выборки ($\overline{CS}$) низкого уровня напряжения. На входе разрешения записи ($\overline{W}$) должен быть подан высокий уровень напряжения.

Записанная информация считывается параллельно по четырем выходам C1-C4 в инверсном коде. Выходные каскады усилителей считывания

выполнены по схеме с открытым коллектором, что дает возможность информационные выходы различных ИС объединять в монтажное ИЛИ.

Достоверное считывание информации после записи может осуществляться через время $t_{вос.зп}$, в течение которого происходит восстановление усилителей записи и считывания.

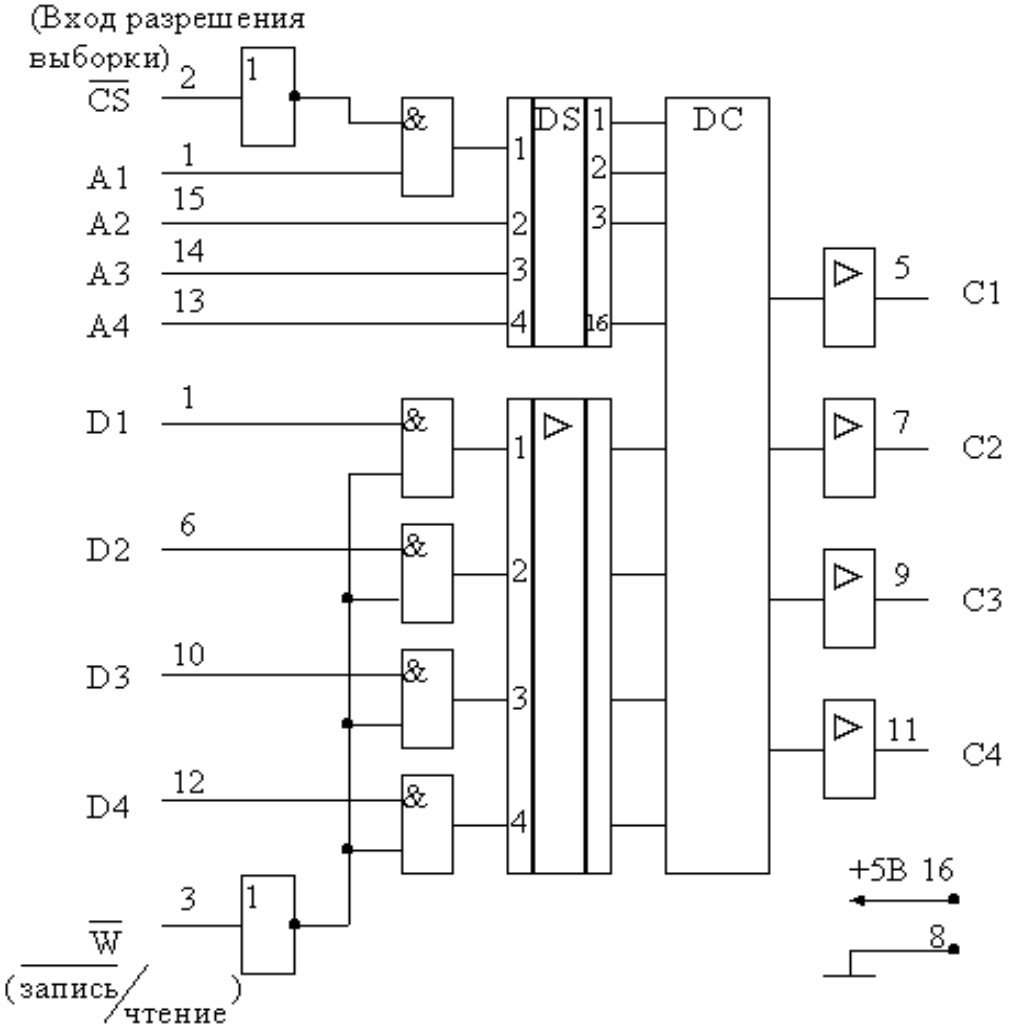


Рисунок 2 – Логическая структура микросхемы РУ2

1.2. Постоянное запоминающее устройство на примере ИС типа КР1601РР1

Постоянные запоминающие устройства (ПЗУ или ROM) – матрицы пассивных элементов памяти со схемами управления, предназначенные

для воспроизведения неизменной информации, заносимой в матрицу при изготовлении (в режиме хранения информации энергия не потребляется).

Программируемые постоянные запоминающие устройства (ППЗУ или PROM) – постоянные запоминающие устройства с возможностью однократного электрического программирования. Они отличаются от ПЗУ тем, что позволяют в процессе применения микросхемы однократно изменить состояние запоминающей матрицы электрическим путем по заданной программе.

Репрограммируемые постоянные запоминающие устройства РПЗУ (EEPROM) – постоянные запоминающие устройства с возможностью многократного электрического перепрограммирования. Они отличаются от ППЗУ тем, что допускают многократную электрическую запись информации, но число циклов записи и стирания ограничено (до 10^4 циклов).

Репрограммируемые постоянные запоминающие устройства с ультрафиолетовым стиранием и электрической записью информации (РПЗУ УФ или EPROM). Они отличаются от ПЗУ только способом стирания информации с помощью ультрафиолетового освещения, для чего в корпусе микросхемы имеется специальное окно.

ПЗУ состоит из ячеек, обратившись к которым можно вывести их содержимое. Отличие от ОЗУ состоит в том, что информация в ячейки записывается однократно, после чего в процессе эксплуатации используется лишь режим чтения.

По способу занесения информации ПЗУ делятся на два вида: ПЗУ, программируемые маской на предприятии-изготовителе, и ПЗУ, программируемые пользователем.

В ПЗУ первого типа информация заносится в процессе изготовления микросхемы с помощью соответствующего фотошаблона. Очевидно, такой способ записи пригоден в тех случаях, когда производится выпуск крупной партии ПЗУ с одной и той же записанной в них информацией. Промышленность выпускает такие ПЗУ, например, для использования в

качестве преобразователя двоичного кода в определенные двоично-десятичные коды и других преобразователей. В них входная кодовая комбинация служит адресом ячейки, а содержимое ячейки – выходной кодовой комбинацией (являющейся, например, кодовой комбинацией двоично-десятичного кода).

В ПЗУ, программируемых пользователем, запись информации производится непосредственно пользователем с помощью специальных устройств, называемых программаторами. Программатор выдает в микросхему соответствующие напряжения для записи информации, набираемой на клавиатуре, либо предварительно нанесенной путем пробивок на перфоленту. Этими напряжениями осуществляется прожигание плавких перемычек в элементах памяти. Очевидно однажды записанная в ПЗУ информация, в дальнейшем не может быть изменена. При необходимости изменить содержимое ПЗУ микросхемы с ранее записанной информацией заменяются новыми, в которые записываются новые данные.

Основная отличительная особенность микросхем РПЗУ заключается в их способности к многократному (от 100 до 10 тыс.) перепрограммированию самим пользователем. Это свойство микросхем обеспечено применением ЭП со свойствами управляемых «перемычек», функции которых выполняют транзисторы со структурой МНОП (металл Al-нитрид кремния Si_3N_4 – окисел кремния SiO_2 – полупроводник Si) и транзисторы n-МОП с плавающим затвором (ПЗ) с использованием механизма лавинной инжекции заряда ЛИЗМОП.

Всю номенклатуру выпускаемых микросхем РПЗУ можно разделить на две группы: РПЗУ с записью и стиранием электрическими сигналами (группа ЭС) и РПЗУ с записью электрическими сигналами и стиранием ультрафиолетовым излучением (группа УФ).

Микросхемы РПЗУ-ЭС содержат ЭП типа МНОП (К558, К1601) и ЛИЗМОП с двойным затвором (К573РР2, К1609РР1 и др.).

Микросхемы РПЗУ с ЭП на р-МНОП транзисторах КР558РР1, КР1601РР1, КР1601РР3 имеют сравнительно низкое быстродействие, вы-

сокое напряжение программирования (30…40 В) и требуют двух источников питания.

Для улучшения характеристик РПЗУ широко применяют технологию изготовления ЭП на n-МНОП транзисторах. Такие ЭП устроены аналогично рассмотренным, но имеют проводимость подложки р-типа, а истока и стока – n-типа. Микросхемы с ЭП на n-МНОП транзисторах КР558РР2, КР558РР3, К16111РР1 обладают втрое превосходящим быстродействием, сниженным до 22 В напряжением программирования, и работают от одного источника питания.

Устройство, принцип действия, режимы управления работой микросхем РПЗУ разных групп во многом аналогичны. Например, микросхемы К558РР2, К1609РР1, К573РР2, К573РФ2 емкостью 2Кх8 бит, относящиеся к разным группам РПЗУ по типу элемента памяти, имеют похожую структуру и одинаковую разводку выводов корпуса. Отличие между микросхемами групп ЭС и УФ состоит в способе реализации режима стирания.

Структурная схема (рисунок 3) содержит все элементы, необходимые для работы микросхемы в качестве ПЗУ: матрицу с элементами памяти, дешифраторами кода адреса строк и столбцов, селектор (ключи выбора столбцов), устройство ввода-вывода УВВ. Кроме того, в структуре предусмотрены функциональные узлы, обеспечивающие ее работу в режимах стирания и программирования (записи информации) – это коммутаторы режимов и формирователи импульсов напряжений требуемой амплитуды и длительности из напряжения программирования U_{PR}. По сравнению с микросхемами ПЗУМ и ППЗУ система управляющих сигналов дополнена сигналами программирования $\overline{PR}$ и стирания $\overline{ER}$. Накопитель с матричной организацией содержит 64 строки и 64 столбца, на пересечениях которых расположены 4096 элементов памяти. Управление накопителем осуществляют шестью старшими разрядами адресного кода, который после дешифрирования выбирает строку с 64 элементами памяти. Сигналы, считанные с элементов выбранной строки, поступают на входы селекто-

ра, назначение которого состоит в выборе из 64-разрядного кода на входах четырех разрядов, которые далее поступают через УВВ на выходы микросхемы. Селектором управляют четыре младших разряда адресного кода, которые после дешифрирования обеспечивают выборку одного четырехразрядного слова из 16 слов, содержащихся в выбранной строке. Устройство управления под воздействием сигналов на своих входах обеспечивает работу микросхемы в одном из следующих режимов: хранения, считывания, стирания, записи (программирования). Управляющие сигналы имеют следующее назначение: *CS* – выбор микросхемы; $\overline{PR}$ – разрешение на режим записи (программирования); U_{PR} – напряжение программирования; *RD* – сигнал считывания; $\overline{ER}$ – сигнал стирания информации. Входы сигналов инверсные, поэтому разрешающим значением этих сигналов является 0. Многие микросхемы группы ЭС допускают избирательное стирание по адресу.

Процесс стирания начинается с момента подачи импульса $\overline{ER}$, который должен иметь длительность 200 мс. По окончании стирания все ЭП матрицы переходят в состояние, соответствующее логической 1. В этом режиме сигналы на адресных и информационных выводах могут иметь произвольные значения.

Микросхема КР1601РР1 допускает построчное стирание. Этот режим отличается от рассмотренного значением сигнала $\overline{PR} = 0$, наличием на всех информационных выводах сигналов с уровнем 1, а на адресных входах – сигналов адреса строки A_4 – A_9, по которому следует стереть информацию из всех 64 ЭП. Время избирательного стирания то же, что и общего.

Режим хранения обеспечивают сигналом *CS* = 1, запрещающим обращение к микросхеме независимо от значений сигналов на других входах. Возможен второй вариант обеспечения режима хранения при использовании импульсного питания напряжением минус 12 В. Такой режим позволяет уменьшать потребляемую мощность. Когда в паузах между обращениями к микросхеме отключают напряжение питания, она переходит в режим хранения. Управление переключениями питания целесообразно осуществлять сигналом CS.

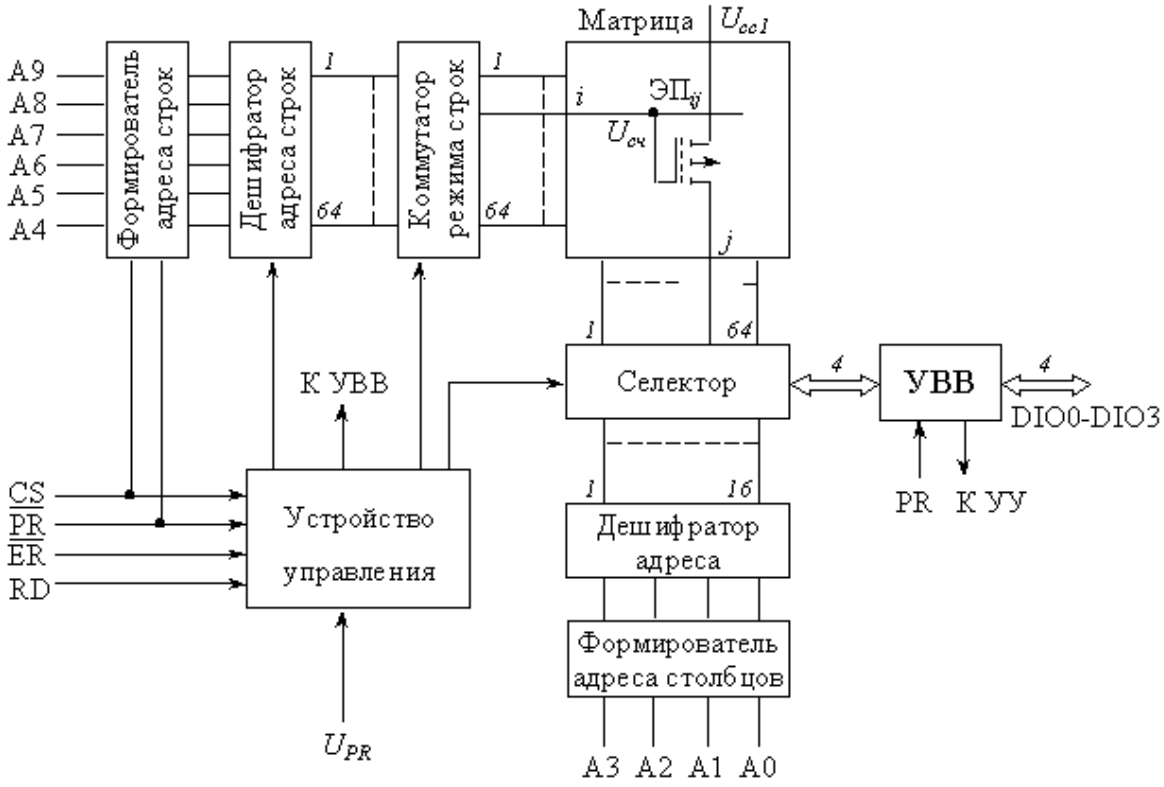


Рисунок 3 – Структурная схема РПЗ КР1601РР1

При эксплуатации микросхем РПЗУ необходимо обеспечить требуемый порядок включения и выключения напряжений питания и программирования; при включении вначале подают 5 В, затем – 12 В и последним – напряжение программирования U_{PR} = - 32 В. При выключении последовательность меняется на обратную. Можно все три напряжения включать и выключать одновременно.

Достоинством микросхем РПЗУ группы ЭС является возможность перепрограммирования без изъятия их из устройства, где они работают. Другим положительным свойством микросхем данной группы является значительное число циклов перепрограммирования, достигающее для большинства микросхем 10 тыс. Эти свойства в сочетании с энергонезависимостью позволяют широко использовать их в аппаратуре в качестве встроенных ПЗУ со сменяемой информацией. Гарантийный срок сохранения информации при отключенном питании составляет от 3 тыс. ч. до 5 лет.

1.3. Аналого-цифровой преобразователь на примере ИС типа КР572ПВ2

Аналого-цифровой преобразователь – устройство, осуществляющее автоматическое преобразование (измерение и кодирование) непрерывно изменяющихся во времени аналоговых величин в эквивалентные значения числовых кодов.

При преобразовании напряжения в цифровой код используются три независимых операции: дискретизация, квантование и кодирование. Процедура аналого-цифрового преобразования непрерывного сигнала представляет собой преобразование непрерывной функции напряжения u(t) в последовательность чисел $u(t_n)$, где n = 0, 1, 2…, отнесенных к некоторым фиксированным моментам времени. При дискретизации непрерывная функция u(t) преобразуется в последовательность ее отсчетов $u(t_n)$.

Вторая операция, называемая квантованием, состоит в том, что мгновенные значения функции u(t) ограничиваются только определенными уровнями, которые называются уровнями квантования. В результате квантования непрерывная функция u(t) принимает вид ступенчатой кривой $u_k(t)$.

Третья операция, называемая кодированием, представляет дискретные квантованные величины в виде цифрового кода, т.е. последователь-

ности цифр, подчиненных определенному закону. С помощью операции кодирования осуществляется условное представление численного значения величины.

В основе дискретизации сигналов лежит принципиальная возможность представления их в виде взвешенных сумм:

$$u(t) = \sum_n a_n f_n(t), \qquad (1.3.1)$$

где a_n – некоторые коэффициенты или отсчеты, характеризующие исходный сигнал в дискретные моменты времени,

$f_n(t)$ – набор элементарных функций, используемых при восстановлении сигнала по его отсчетам.

Дискретизация бывает равномерная и неравномерная. При равномерной дискретизации период отсчетов T остается постоянным, а при неравномерной – период может изменяться. Неравномерная дискретизация чаще всего обусловлена скоростью изменения сигнала и потому называется адаптивной.

В АЦП погрешность квантования определяется как единица младшего значащего разряда (МЗР).

Выходной величиной АЦП является цифровой код, т.е. последовательность цифр, с помощью которой представляются дискретные квантованные величины. В АЦП используют четыре основных типа кодов: натуральный двоичный, десятичный, двоично-десятичный и код Грея. Кроме этого, АЦП, предназначенные для вывода информации в десятичном коде, выдают на своем выходе специализированный код для управления семисегментными индикаторами.

Любой АЦП является сложным электронным устройством, которое может быть выполнено в виде одной интегральной микросхемы или содержать большое количество различных электронных компонентов. В связи с этим характеристики АЦП зависят не только от его построения, но и от характеристик элементов, входящих в его состав. Тем не менее

большинство АЦП оценивают по их основным метрологическим показателям, которые можно разделить на две группы: статические и динамические.

К статическим характеристикам АЦП относят: абсолютные значения и полярности входных сигналов, входное сопротивление, значения и полярности выходных сигналов, выходное сопротивление, значения напряжений и токов источников питания, количество двоичных или десятичных разрядов выходного кода, погрешности преобразования постоянного напряжения и др. К динамическим параметрам АЦП относят: время преобразования, максимальную частоту дискретизации, аппретурное время, динамическую погрешность и др.

Рассмотрим некоторые из этих параметров более подробно.

Разрешающая способность является основной характеристикой АЦП, которую принято определять величиной, обратной максимальному числу кодовых комбинаций на выходе АЦП. Разрешающую способность можно выражать в процентах, в количестве разрядов или в относительных единицах. Например, 10-разрядный АЦП имеет разрешающую способность $(1024)^{-1} \approx 10^{-3} = 0{,}1\%$. Если напряжение шкалы для такого АЦП равно 10 В, то абсолютное значение разрешающей способности будет около 10 мВ. Реальное значение разрешающей способности отличается от расчетного из-за погрешностей АЦП.

Точность АЦП определяется значениями абсолютной погрешности, дифференциальной и интегральной нелинейности.

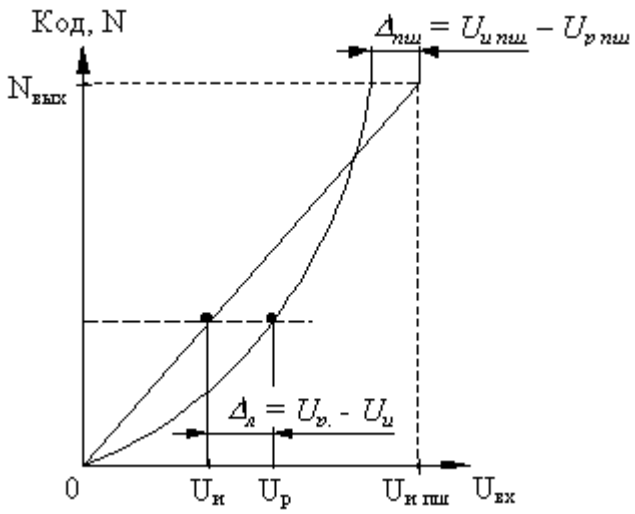
Абсолютную погрешность АЦП определяют в конечной точке характеристики преобразования, поэтому ее обычно называют погрешностью полной шкалы и измеряют в единицах младшего разряда.

Дифференциальную нелинейность (DNL) определяют через идентичность двух соседних приращений сигнала, т.е. как разность напряжений двух соседних квантов: $DNL=h_i-h_{i+1}$.

Интегральная нелинейность АЦП (INL) характеризует идентичность приращений во всем диапазоне входного сигнала. Обычно ее определяют,

как показано на рисунке 4, по максимальному отклонению сглаженной характеристики преобразования от идеальной прямой линии, т.е. INL = u’i-ui.

Рисунок 4 – Определение интегральной нелинейности и погрешности полной шкалы



Коэффициент нелинейности экспериментальной характеристики $К_н$:

$$К_н = \frac{\mathrm{tg}\alpha_{max} - \mathrm{tg}\alpha_{min}}{\mathrm{tg}\alpha_{ср}}, \qquad (1.3.2)$$

где α_{max}, α_{min}, $\alpha_{ср}$ – максимальный, минимальный и средний углы наклона касательной к экспериментальной характеристике.

Степень отклонения реальной характеристики АЦП от расчетной определяется абсолютной погрешностью полной шкалы (рисунок 4).

Время преобразования $Т_{пр}$ обычно определяет как интервал времени от начала преобразования до появления на выходе АЦП устойчивого кода входного сигнала. Для одних типов АЦП это время постоянное и не зависит от значения входного сигнала, для других АЦП это время зависит от значения входного сигнала. Если АЦП работает без устройства выборки и хранения, то время преобразования является апертурным временем.

Максимальная частота дискретизации – его частота, с которой возможно преобразование входного сигнала, при условии, что выбранный параметр (например, абсолютная погрешность) не выходит за заданные пределы. Иногда максимальную частоту преобразования принимают равной обратной величине времени преобразования. Однако это пригодно не для всех типов АЦП.

Все типы используемых АЦП можно разделить по признаку измеряемого значения напряжения на две группы: АЦП мгновенных значений и АЦП средних значений напряжения (интегрирующие АЦП). АЦП мгновенных значений можно разделить на следующие основные виды: последовательного счета, последовательного приближения, параллельные, параллельно-последовательные и с промежуточным преобразованием в интервалы времени.

АЦП средних значений напряжения (интегрирующий АЦП) можно разделить на следующие основные виды: с времяимпульсным преобразованием, с частотно-импульсным преобразованием и со статическим усреднением. Наибольшее распространение получили первые две группы АЦП.

Структурная схема интегрирующего АЦП с времяимпульсным преобразованием приведена на рисунке 5. Работу этой схемы можно разделить на три такта. В первом такте производится заряд интегратора, во втором – его разряд, а в – третьем коррекция нулевого уровня интегратора. Графики, иллюстрирующие работу АЦП, приведены на рисунок 6.

В первом такте, имеющем фиксированную длительность T_o, замкнут ключ SI, а остальные ключи разомкнуты. В этом случае входное напряжение $u_{вх}$ через замкнутый ключ SI и сопротивление R_1 заряжает емкость C_1 интегратора и выходное напряжение растет линейно во времени, как показано на рисунке 6. К концу интервала T_0 напряжение на выходе интегратора будет равно:

$$u_1(T_0) = k\int_0^{T_0} u_{вх}dt = kT_0U_{вх}\,, \qquad (1.3.3)$$

где $k^{-1} = R_1C_1$ – постоянная времени интегратора, $U_{вх}$ – среднее значение входного напряжения:

$$U_{вх} = T_0^{-1} \int_0^{T_0} u_{вх} dt . \qquad (1.3.4)$$

Во втором такте происходит разряд интегратора. При этом в зависимости от требуемой полярности источника опорного напряжения замыкается один из ключей S2 или S3. Разряд интегратора происходит с постоянной скоростью, которая не зависит от накопленного в интеграторе заряда, поэтому с увеличением накопленного заряда время разряда также увеличивается.

Конец разряда интегратора фиксируется компаратором К, после чего ключ S2 (или S3) размыкается.

Поскольку начало разряда определяет схема управления, а конец – компаратор, то длительность разряда интегратора можно определить по формуле:

$$U_{вх} = T_0^{-1} \int_0^{T_0} u_{вх} dt = 0, \qquad (1.3.5)$$

откуда

$$kT_0U_{вх} = kU_{оп}T_x \quad \text{или} \quad T_x = \frac{T_0}{U_{оп}} U_{вх},$$

что свидетельствует о пропорциональности интервала T_x среднему значению входного напряжения $U_{вх}$. Заполнение интервала T_x счетными импульсами, поступающими от схемы управления, позволяет найти числовой код $N_x = T_x f_0$. Для исследуемого АЦП $N_0 = f_0T_0 = 1000$. Тогда:

$$N_x = 10^3 U_{вх}/U_{оп}. \qquad (1.3.6)$$

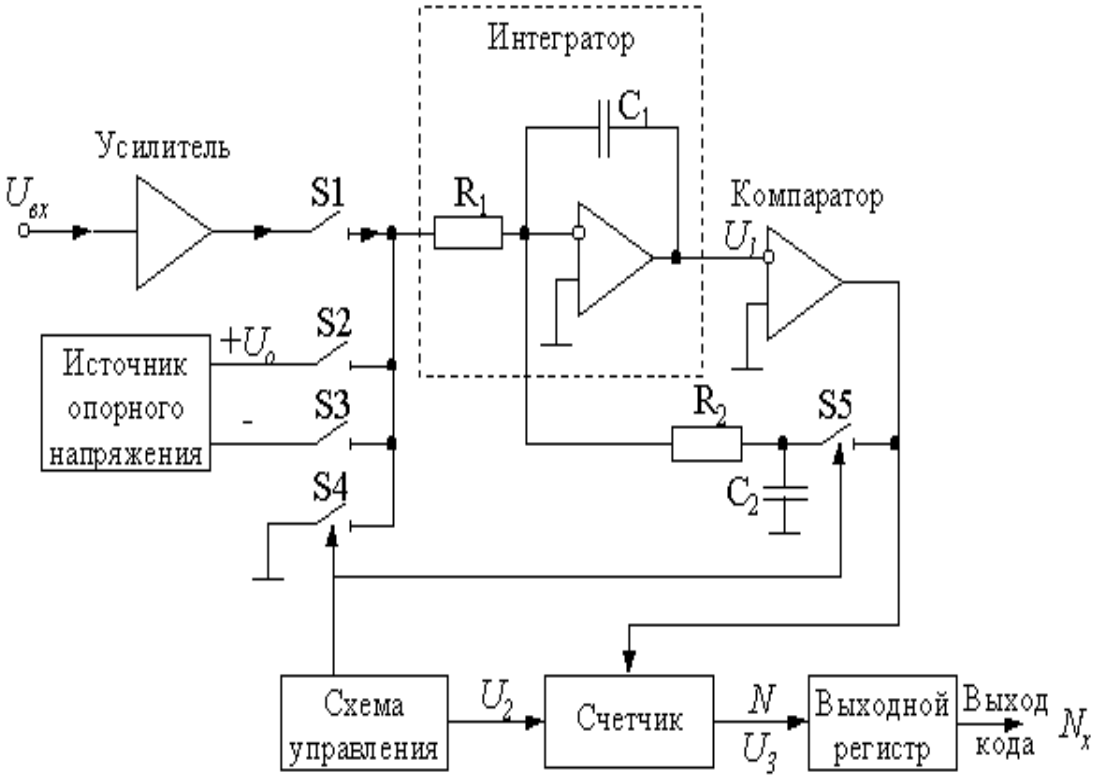


Рисунок 5 – Структурная схема интегрирующего АЦП с времяимпульсным преобразованием

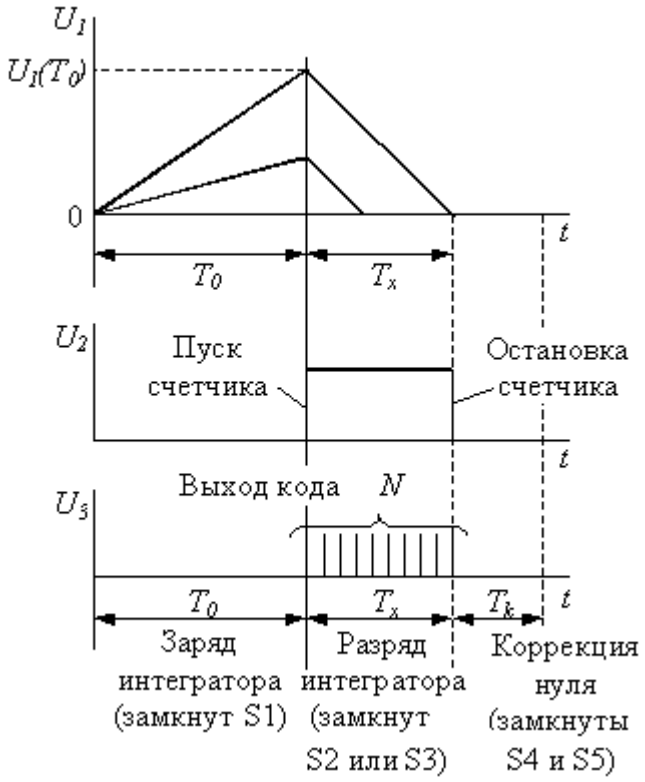


Рисунок 6 – Графики, иллюстрирующие работу АЦП

К достоинствам интегрирующих АЦП следует отнести их высокую помехозащищенность. Если на входной сигнал наложена гармоническая помеха, то при равенстве периода помехи времени заряда интегратора $T_п = T_0$ среднее значение помехи к концу интервала интегрирования будет равна нулю, как показано штриховой линией на рисунке 6. Случайные помехи и шумы также ослабляются интегрированием, хотя и в меньшей степени.

На третьем этапе производится коррекция нулевого уровня интегратора. Для этого замыкаются ключи S4 и S5, а остальные ключи размыкаются. Так как вход интегратора через сопротивление R_1 соединен с общей шиной, то конденсатор C_2 через замкнутый ключ S5 заряжается до напряжения ошибки, которое после размыкания ключей S4 и S5 вычитается из входного сигнала.

К недостаткам таких интегрирующих АЦП относится, прежде всего, сравнительно невысокое быстродействие. Кроме этого, при перегрузке АЦП большим входным сигналом происходит перезаряд интегрирующего конденсатора C_1, поэтому после снятия перегрузки в течение нескольких циклов АЦП будет работать с большой погрешностью.

Полупроводниковые БИС интегрирующего АЦП КР572 ПВ2 (А, Б, В) предназначены для применения в измерительной аппаратуре различного назначения. Совместно с ИОН, несколькими резисторами и конденсаторами они выполняют функцию АЦП двойного интегрирования с автоматической коррекцией нуля и определением полярности входного сигнала.

Классификация микросхем КР572 ПВ2 по группам А, Б, В производится по погрешности преобразования.

Нумерация и назначение выводов микросхем КР572ПВ2: 1 – напряжение питания U_{cc1}; 2 – цифровой выход d1; 3 – цифровой выход c1; 4 – цифровой выход b1; 5 – цифровой выход a1; 6 – цифровой выход f1; 7 – цифровой выход g1; 8 – цифровой выход l1; 9 – цифровой выход d10; 10 – цифровой выход c10; 11 – цифровой выход b10; 12 – цифровой выход a10; 13 – цифровой выход f10; 14 – цифровой выход l10; 15 – цифровой выход

d100; 16 – цифровой выход b100; 17 – цифровой выход f100; 18 – цифровой выход l100; 19 – цифровой выход c1000; 20 - цифровой выход g1000; 21 – цифровая земля; 22 – цифровой выход g100; 23 – цифровой выход a100; 24 – цифровой выход c100; 25 - цифровой выход g10; 26 – напряжение питания U_{cc2}; 27 – конденсатор интегратора $C_{инт}$; 28 – резистор интегратора $R_{инт}$; 29 – конденсатор автокоррекции $C_{ак}$; 30 – аналоговый вход 1, низкопотенциальный; 31 – аналоговый вход 2, высокопотенциальный; 32 – общий аналоговый вход; 33 – опорный конденсатор $C_{оп}$; 34 – опорный конденсатор $C_{оп}$; 35 – опорное напряжение $U_{оп}$; 36 – опорное напряжение $U_{оп}$; 37 – контрольный вход; 38 – конденсатор генератора тактовых импульсов (ТИ) $C_{ТИ}$; 39 – резистор генератора ТИ $R_{ТИ}$; 40 – генератор ТИ.

Цифровая информация на выходе микросхем представляется в специальном коде, предназначенном для непосредственного управления 3,5-декадным цифровым табло с 7 сегментными полупроводниковыми индикаторами. Диапазон входного сигнала определяется значением внешнего опорного напряжения и соотношения $U_{вх} = \pm 1{,}999 U_{оп}$. Текущие показания цифрового табло соответствуют $1000 \cdot (U_{вх}/U_{оп})$.

Максимальное число единиц счета преобразователя составляет ±1999. При превышении этого значения в эквиваленте входного преобразуемого сигнала три младшие цифры гаснут, а в старшем разряде остается единица.

1.4 Цифро-аналоговый преобразователь на примере ИС типа КР572 ПА1А

Цифро-аналоговым преобразователем (ЦАП) называется электронное устройство, предназначенное для преобразования цифровой информации в аналоговую. Они используются для формирования сигнала в виде напряжения или тока, функционально связанного с управляющим кодом. В большинстве случаев эта функциональная зависимость является линейной. Наиболее часто ЦАП используются для сопряжения устройств

цифровой обработки сигналов с системами, работающими с аналоговыми сигналами. Кроме этого, ЦАП используются в качестве узлов обратной связи в аналого-цифровых преобразователях и в устройствах сравнения цифровых величин с аналоговыми.

Области применения ЦАП достаточно широки. Они применяются в системах передачи данных, в измерительных приборах и испытательных установках, в синтезаторах напряжения и генераторах сложных функций, для формирования изображений на экране дисплеев и др. В связи с этим разработано и выпускается большое количество интегральных микросхем ЦАП.

Схемы ЦАП можно классифицировать по различным признакам: принципу действия, виду выходного сигнала, полярности выходного сигнала, элементной базе и др. По принципу действия наибольшее распространение получили ЦАП следующих видов: со сложением токов, с делением напряжения и со сложением напряжений. В микроэлектронном исполнении применяются только первые два типа.

По виду выходного сигнала ЦАП делят на два вида: с токовым выходом и выходом по напряжению. Для преобразования выходного тока ЦАП в напряжение обычно используются операционные усилители (ОУ). По полярности выходного сигнала ЦАП принято делить на однополярные и двухполярные.

Управляющий код, подаваемый на вход ЦАП, может быть различным: двоичным, двоично-десятичным, Грея, унитарным и др. Кроме того, различными могут быть и уровни логических сигналов на входе ЦАП.

При формировании выходного напряжения ЦАП под действием управляющего кода обычно используются источники опорного напряжения. В зависимости от вида источника опорного напряжения ЦАП делят на две группы: с постоянным опорным напряжением и с изменяющимся опорным напряжением. Кроме этого, ЦАП делят по основным характеристикам: количеству разрядов, быстродействию, точности преобразования, потребляемой мощности.

Все параметры ЦАП можно разделить на две группы: статические и динамические. К статическим параметрам ЦАП относят: разрешающую способ-

ность, погрешность преобразования, диапазон значений выходного сигнала, характеристики управляющего кода, смещение нулевого уровня и некоторые другие.

К динамическим показателям ЦАП принято относить: время установления выходного сигнала, предельную частоту преобразования, динамическую погрешность. Рассмотрим некоторые из этих параметров.

Разрешающая способность ЦАП определяется как величина, обратная максимальному количеству градаций выходного сигнала. Так, например, если разрешающая способность ЦАП составляет 10^{-5}, то это означает, что максимальное число градаций выходного сигнала равно 10^5. Иногда разрешающую способность ЦАП оценивают выходным напряжением при изменении выходного кода на единицу младшего разряда, т.е. шагом квантования:

$$\Delta_{\text{расч}} = \frac{U_{\text{оп}}}{2^n} \ , \qquad (1.4.1)$$

где Uоп – опорное напряжение, В; n – разрядность ЦАП.

Очевидно, что чем ниже разрядность ЦАП, тем выше его разрешающая способность.

Погрешность преобразования ЦАП принято делить на дифференциальную и погрешность нелинейности. С ростом кода на входе ЦАП растет и выходное напряжение, однако при увеличении напряжения могут быть отклонения от линейной зависимости.

Дифференциальной погрешностью определяется отклонением разности двух аналоговых сигналов, соответствующих соседним кодам от номинального значения МЗР (рисунок 7). Дифференциальная погрешность идеального преобразователя равна нулю. Допустимым отклонением считается $\pm\Delta_{\text{расч}}/2$ (1/2 МЗР).

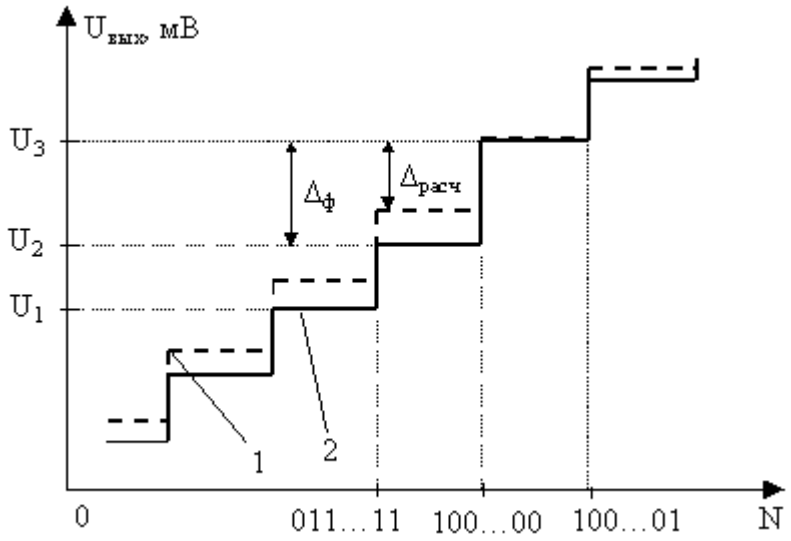


Рисунок 7 – Определение дифференциальной погрешности
1 – выходная характеристика идеального ЦАП,
2 – выходная характеристика реального ЦАП

Относительная дифференциальная погрешность определяется по формуле:

$$\delta_{\text{ндиф}} = \frac{(\Delta_{\text{ф}} - \Delta_{\text{расч}})}{\Delta_{\text{расч}}} \cdot 100\%, \qquad (1.4.2)$$

где $\Delta_{\text{ф}}$ – фактическое приращение выходного сигнала, вызванное изменением двоичного числа на 1.

Как правило, наибольшая динамическая погрешность наблюдается на так называемом главном переходе, когда увеличение двоичного числа на 1 приводит к смене предыдущего кода на код, соответствующий половине шкалы, при этом происходит выключение всех младших разрядов и включением старшего. Объясняется это тем, что при формировании выходного сигнала для кода 01111…1 происходит суммирование погрешностей всех разрядов.

Погрешностью нелинейности называют максимальное отклонение выходного напряжения от идеальной прямой во всем диапазоне преобразования (рисунок 8).

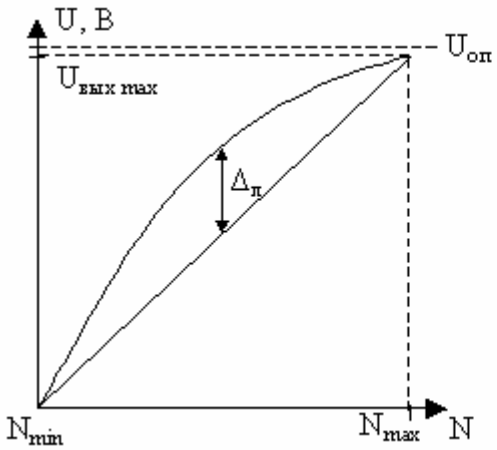


Рисунок 8 – Определение погрешности нелинейности

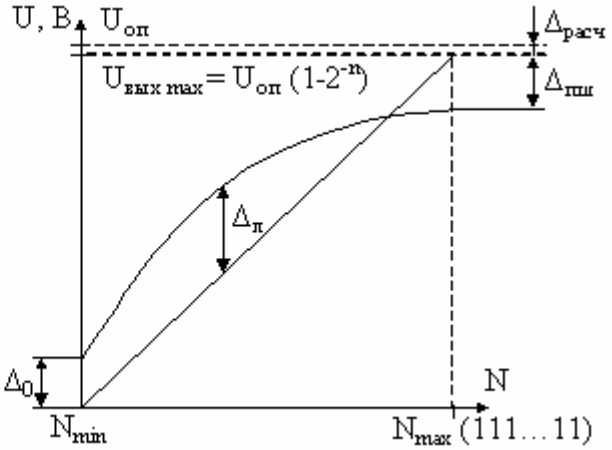


Рисунок 9 – Определение погрешностей полной шкалы, смещения нуля и нелинейности

Коэффициент нелинейности экспериментальной характеристики $К_н$:

$$К_н = \frac{tg\alpha_{max} - tg\alpha_{min}}{tg\alpha_{ср}}, \qquad (1.4.3)$$

где α_{max}, α_{min}, $\alpha_{ср}$ – максимальный, минимальный и средний углы наклона касательной к экспериментальной характеристике.

Напряжение смещения нуля определяется выходным напряжением при входном коде, соответствующем нулевому значению (рисунок 9).

Степень отклонения реальной характеристики ЦАП от расчетной определяется *абсолютной погрешностью полной шкалы* (рисунок 9).

$$\Delta_{пш} = U_{пшном} - U_{пщфакт} = (U_{оп} / 2^n)(2^n - 1) - U_{пщфакт} =$$
$$= (\Delta_{расч} - \Delta_ф^{пш})\,(2^n - 1), \qquad (1.4.4)$$

где $U_{пшном}$ – выходное напряжение полной шкалы идеального ЦАП;

$U_{пшфакт}$ – выходное напряжение полной шкалы реального ЦАП;

$\Delta_{факт}^{пш}$ – фактический шаг квантования.

Относительная погрешность полной шкалы

$$\delta_{пш} = \frac{\Delta_{пш}}{U_{пш}} = \frac{(\Delta_{расч} - \Delta_{факт}^{пш})}{\Delta_{расч}}. \qquad (1.4.5)$$

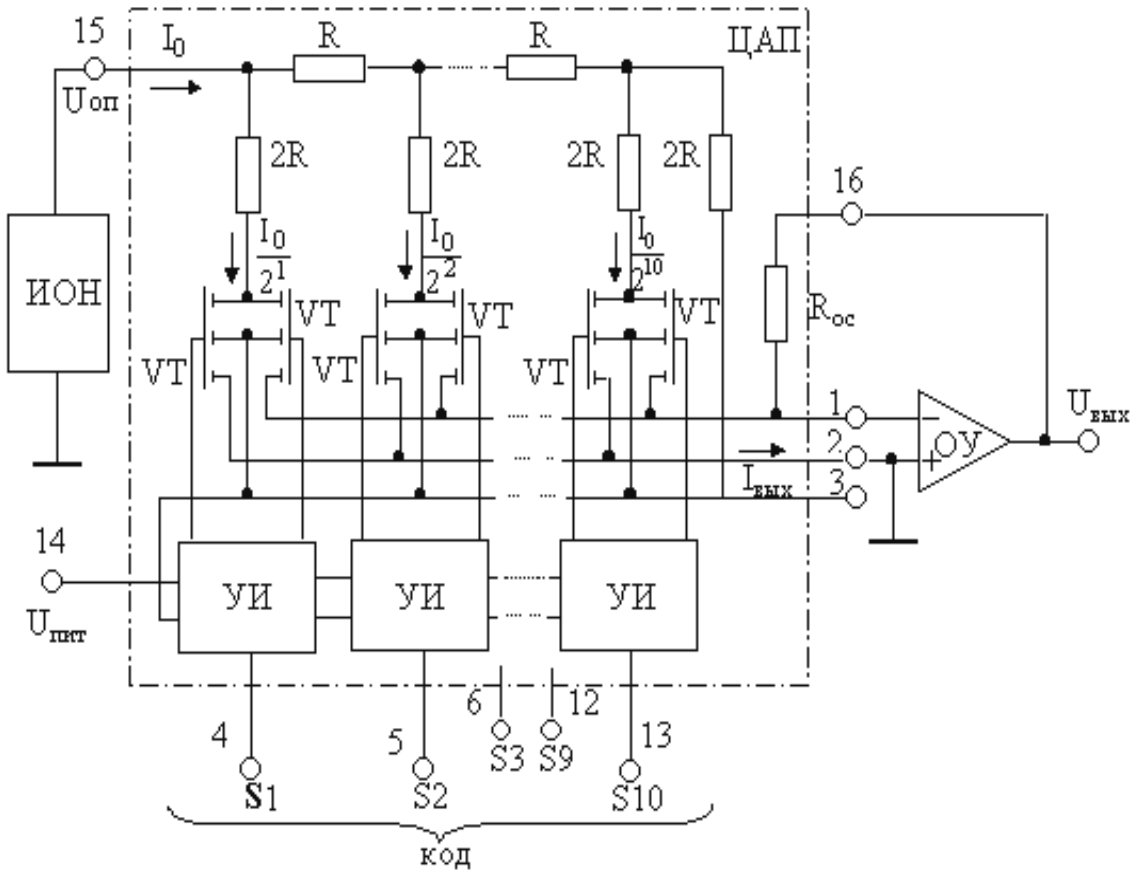


Рисунок 10 – Упрощенная функциональная электрическая схема ЦАП К572ПА1А

Время преобразования – это интервал времени от подачи входного кода до вхождения выходного сигнала в заданные пределы, определяемые погрешностью. Динамические свойства ИС характеризуются временем установления выходного напряжения (тока) на главном переходе.

При этом наблюдается наиболее длительный и сложный переходный процесс со значительными выбросами.

Максимальная частота преобразования – наибольшая частота дискретизации, при которой все параметры ЦАП соответствуют заданным значениям.

Микросхема ЦАП КР572ПА1 предназначена для преобразования 10-разрядного прямого параллельного двоичного кода на цифровых входах в ток на аналоговом выходе, который пропорционален значениям кода и опорного напряжения. Она выполнена по КМОП технологии с поликремневыми затворами.

В состав ИС ЦАП КР572ПА1 входят прецизионные поликремневая резисторная (РМ) типа R – 2R, усилители – инверторы (УИ) для управления токовыми ключами, токовые двухпозиционные ключи, выполненные на КМОП транзисторах. Для работы в режиме с выходом по напряжению к ИС ЦАП КР572ПА1 подключаются внешние источник опорного напряжения (ИОН) и операционный усилитель (ОУ) с цепью отрицательной обратной связи (ЦОС), работающей в режиме суммирования токов.

Нумерация и назначение выводов микросхемы: 1 – аналоговый выход 1; 2 – аналоговый выход 2; 3 – общий вывод; 4 – цифровой вход 1 (СЗР); 5-12 – цифровые входы 2-9; 13 – цифровой вход 10 (МЗР); 14 – напряжение источника питания; 15 – опорное напряжение; 16 – вывод резистора обратной связи.

Метод преобразования, используемый в ИС К572ПА1, предполагает суммирование в соответствии с заданным значением двоичного кода всех разрядных токов, взвешенных по двоичному закону и пропорциональных значению опорного напряжения на выводе 15. Двоичный закон распределения токов в ветвях РМ соблюдается при условии равенства потенциалов выходов 1 и 2. Это обеспечивается подключением выхода 1 к инвертирующему входу ОУ, охваченного отрицательной обратной связью. Неинвертирующий вход ОУ соединяется с выходом 2 и с шиной аналоговой земли. При этом осуществляется преобразование тока на выходе 1 в про-

порциональное ему напряжение на выходе ОУ. Резистор Roc определяет значение коэффициента преобразования и напряжения в конечной точке шкалы.

Связь между напряжением на выходе 1 схемы и двоичным кодом на цифровых входах ЦАП однозначна:

000…000	0
000…001	-2^{-10} Uоп
…	…
100…000	-2^{-1} Uоп
…	…
111…111	$-(1-2^{-10})$ Uоп

1.5 Арифметико-логическое устройство на примере ИС типа К155ИП3

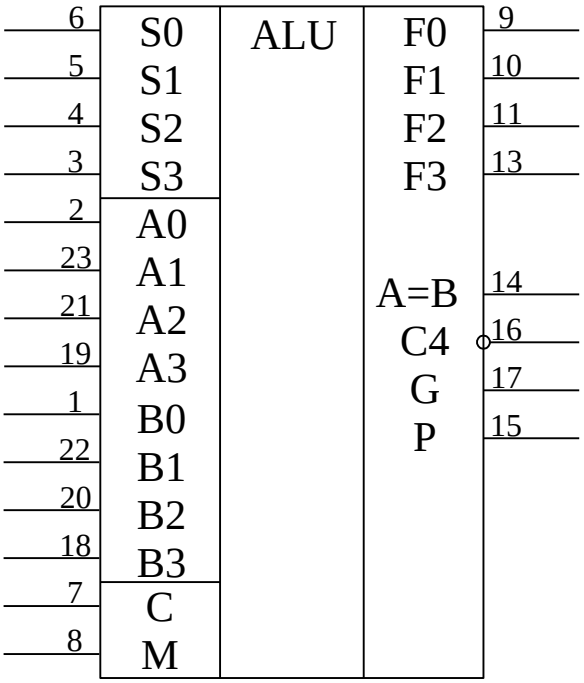


Рисунок 11 – Условное изображение ИМС К155ИП3

Выпускаются специализированные ИМС, выполняющие в соответствии с программой на входах арифметические и логические преобразования двоичной информации. Эти ИМС называют арифметико-логическими устройствами (АЛУ). По сравнению с приборами, работающими по жестокой, наперед заданной программе, АЛУ представляют собой устройства более высокого класса. В МП технике АЛУ являются базовыми элементами. Они используются в сочетании с регистрами сдвига, ОЗУ и другими узлами. АЛУ дороже простых ИМС, однако благодаря универсальны свойствам, применение их в аппа-

ратуре во многих случаях оказывается оправданным. ИМС АЛУ, принадлежащие к разным видам логик, в частности к ТТЛ-типа К155ИП3 и КМОП-типа 564ИП3 функционально во многом совпадают, в том числе и по разводке выводов.

К155ИП3 предназначена для действий с двумя 4-х разрядными двоичными словами: $A=A_3A_2A_1A_0$ и $B=B_3B_2B_1B_0$ (рисунок 11). Конкретный вид операции, выполняемой ИМС, задается 5-разрядным кодом на входах M S_3 S_2 S_1 S_0. Всего это АЛУ способно выполнить $2^5=32$ операции: 16 логических (И, И-НЕ, ИЛИ, ИЛИ-НЕ, исключающее ИЛИ и другие) и 16 арифметических и арифметико-логических(сложение, вычитание, удвоение, сравнение чисел и другое). Операции сложения и вычитания проводятся с ускоренным переносом из разряда в разряд. Кроме того, имеется вход приема сигнала переноса C.

На входах F_0, F_1, F_2 и F_3 формируются результаты логических преобразований и арифметических действий. На выходе переноса C_4 образуется сигнал старшего (пятого) разряда при выполнении арифметических операций. Дополнительные выходы – образования ускоренного переноса P-используются только при организации многоразрядных АЛУ в случае их сочетания с блоком ускоренного переноса К155ИП3 (или 564ИП4 для ИМС КМОП).

Слова A и B, подлежащие обработке, могут быть представлены в положительной либо отрицательной логике. Таблицы истинности для каждого варианта логики различны (таблица 2). Во избежание путаницы уровни сигналов обозначены в них буквенными символами. Результаты арифметических операций выражены в дополнительном коде. Как отмечалось, числа в дополнительном и обратном коде связанны простым соотношением $N_{доп}=N_{обр}+1$ или $N_{обр}=N_{доп}+1$. Поэтому в тех сторонах табли-

цы 9-13, где указана операция «минус 1», результат арифметических действий представлен в обратном коде.

Таблица 2 – Выбор функций АЛУ К155ИП3

Выходы выбора функции				Вход – выход (отрицательная логика)		Вход – выход (положительная логика)	
S3	S2	S1	S0	Логические функции (M=H)	Арифметические действия (M=L, C=L)	Логическая функция (M=H)	Арифметическое действие (M=L, C=H)
L	L	L	L	$\overline{A}$	A минус 1	$\overline{A}$	A
L	L	L	H	$\overline{A \cdot B}$	$A \cdot B$ минус 1	$\overline{A \vee B}$	$A \vee B$
L	L	H	L	$\overline{A} \vee B$	$A \cdot \overline{B}$ минус 1	$\overline{A} \cdot B$	$A \vee \overline{B}$
L	L	H	H	Логич. 1	Минус 1	Логич. 0	Минус 1
L	H	L	L	$\overline{A \vee B}$	A плюс $(A \vee \overline{B})$	$\overline{A} \cdot B$	A плюс $A \cdot \overline{B}$
L	H	L	H	$\overline{B}$	$A \cdot B$ плюс $A \vee \overline{B}$	$\overline{B}$	$(A \vee B)$ плюс $A\overline{B}$
L	H	H	L	$\overline{A \oplus B}$	A минус B минус 1	$A \oplus B$	A минус B минус 1
L	H	H	H	$A \vee \overline{B}$	$A \vee \overline{B}$	$A\overline{B}$	$A\overline{B}$ минус 1
H	L	L	L	$\overline{A}B$	A плюс $(A \vee B)$	$\overline{A} \vee B$	A плюс AB
H	L	L	H	$A \oplus B$	A плюс B	$\overline{A \oplus B}$	A плюс B
H	L	H	L	B	$A\overline{B}$ плюс $(A \vee B)$	B	$(A \vee \overline{B})$ плюс AB
H	L	H	H	$A \vee B$	$A \vee B$	AB	AB минус 1
H	H	L	L	Логич. 0	(А плюс А)*	Логич. 1	А плюс А*
H	H	L	H	$A\overline{B}$	AB плюс А	$A \vee \overline{B}$	$(A \vee B)$
H	H	H	L	AB	$A\overline{B}$ плюс А	$A \vee \overline{B}$	$(A \vee \overline{B})$ плюс А
H	H	H	H	А	А	А	А минус 1

Примечания: 1. L – низкий уровень напряжения; H – высокий уровень напряжения.

2. Высокий уровень напряжения (H) на выходе A=B имеет место при равенстве слов А и В.

* равнозначно тому, что каждый разряд сдвинут в направлении более высокого разряда.

Старший разряд кода выбора операции (вход M) определяет характер действий, выполняемых АЛУ.

Когда на этом входе сигнал высокого уровня, АЛУ производит логические операции поразрядно над каждой парой бит слов A и B. Внутренний перенос в этом режиме бездействует.

Арифметические операции выполняются, когда на входе M установлен низкий потенциал, который является также разрешающим сигналом для переноса между разрядами. Выходной результат формируется с учетом состояния входа переноса. Оба сигнала переноса – входной C и выходной C_4 – инверсны относительно сигналов на входах A и B, то есть когда слова A и B – в положительной логике, сигналу переноса отвечает низкий уровень напряжения на соответствующем выводе, а в отрицательной логике – наоборот.

Если АЛУ выполняет логико-арифметическую операцию, логическая функция реализуется поразрядно, а арифметическая с переносом. Например, вход коду M S_3 S_2 S_1 S_0=LHHLH (HBBHB) отвечает операция AB плюс A (третья снизу строка таблицы 2, отрицательная логика), где AB – логическое умножение двух слов. Если A=1010 и B=0111, то первая операция дает AB=0010 и, следовательно, 0010 плюс 1010 = 1100.

При использовании АЛУ в качестве компаратора сигнал с выхода A=B (рисунок 11). Этот выход – с открытым коллектором, и к источнику питания следует подключать через внешний резистор 1 кОм.

Режим компаратора обеспечивается при M=L и $S_3S_2S_1S_0$ =LHHL. Когда числа A и B равны, на выходе A=B формируется сигнал высокого уровня.

Одновременно сигнал на выходе C_4 характеризует соотношение между числами A и B и в случае их неравенства – согласно таблице 3.

Таблица 3 – Зависимость сигнала на выходе C_4 от соотношения входов A, B и Cn

Вид логики	Состояние выходов		Состояние выхода C_4
	Cn	A и B	
Положительная	H	A≤B	H
	L	A<B	H
	H	A>B	L
	L	A>B	L
Отрицательная	L	A≤B	L
	H	A<B	L
	L	A>B	H
	H	A≥B	H

Для арифметических действий над словами большой длины АЛУ включают последовательно. Здесь время суммирования определяется задержкой распространения сигнала переноса со входов младшего разряда до выхода с последнего АЛУ и составляет $t_{зд.р}=4\tau_{зд.р}$, где $\tau_{зд.р}$ - задержка распространения сигнала переноса в одной АЛУ.

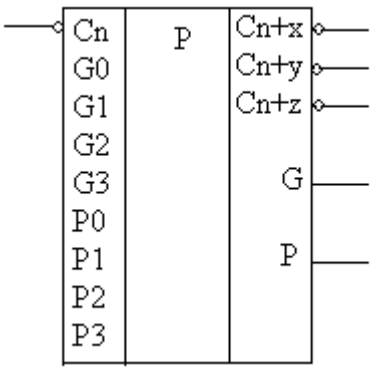


Рисунок 12 – Условное изображение К155ИП4

Уменьшить время суммирования можно применением ИМС К155ИП4 (564ИП4), специально разработанных для организации ускоренного переноса между отдельными АЛУ, а также между группами АЛУ (изображение – рисунок 12). Если при выполнении арифметических операций к быстродействию не предъявляется высоких требований, то при каскадировании АЛУ схемы ускоренного переноса не используют.

При помощи ИМС К 155ИП4 (564ИП4) можно сформировать ускоренный сквозной перенос при выполнении операции сложения группой

из четырех АЛУ (16-разрядные числа), что дает определенный выигрыш во времени сравнительно с последовательным переносом. Последовательное соединение нескольких таких ИМС, каждая из которых спарена с АЛУ, позволяет выполнять ускоренный перенос и с большим числом разрядов.

Сигналы образования группового переноса G_0 – G_3 и сигналы распространения группового переноса P_0 – P_3 с выходов АЛУ подключают с учетом разрядности к соответствующим входам ИМС ускоренного переноса (рисунок 13). Функциональные свойства этой ИМС представлены в таблице 4.

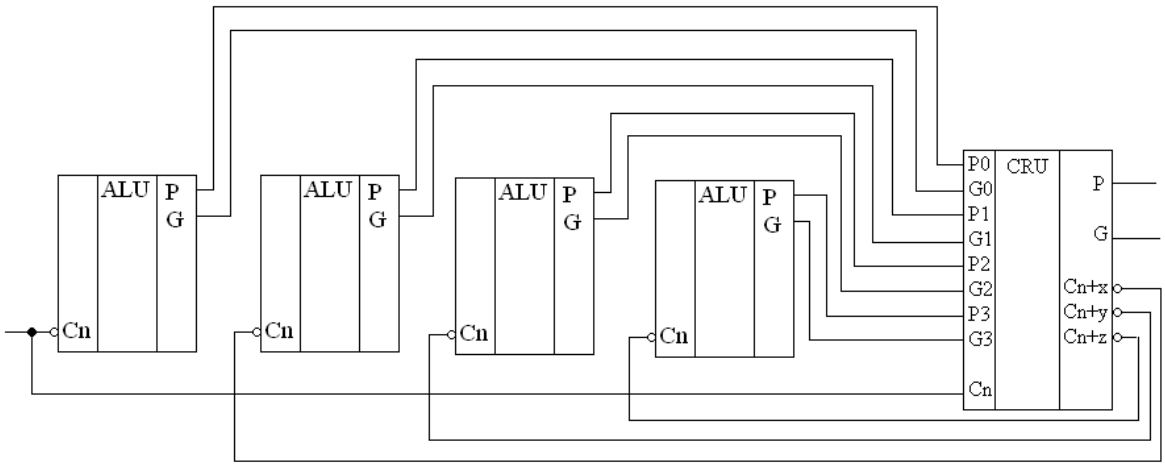


Рисунок 13 – 16-разрядное АЛУ с блоком ускоренного переноса (ИМС К155ИП4)

Таблица 4 – Фун циональные свойства ИМС К 155ИП4

Вывод	Функция (положительная логика)	
$\overline{C_{n+x}}$	$\overline{G_0} + \overline{P_0} \cdot C_n$	В случае наращивания ИМС ускоренного переноса (для чисел, число разрядов которых более 16) используются выходы P и G. С помощью 4х таких ИМС в сочетании с 15 ИМС АЛУ можно построить 64-разрядное АЛУ
$\overline{C_{n+y}}$	$\overline{G_1} + \overline{P_1}\overline{G_0} + \overline{P_1}\overline{P_0}C_n$	
$\overline{C_{n+z}}$	$\overline{G_2} + \overline{P_1}\overline{G_1} + \overline{P_2}\overline{P_1}\overline{G_0} + \overline{P_2}\overline{P_1}\overline{P_0}C_n$	
G	$\overline{G_3} + \overline{P_3}\overline{G_2} + \overline{P_3}\overline{P_2}\overline{G_0} + \overline{P_3}\overline{P_2}\overline{P_1}\overline{G_0}$	
P	$\overline{P_3}\overline{P_2}\overline{P_1}\overline{P_0}$	

2. КОМПЛЕКТ ЛАБОРАТОРНОГО ОБОРУДОВАНИЯ ПО ЭЛЕКТРОННОЙ ТЕХНИКЕ ТИПА К32

Комплект лабораторного оборудования К32 состоит из следующих составных частей (рисунок 14): блока управления комплектом (БУК);

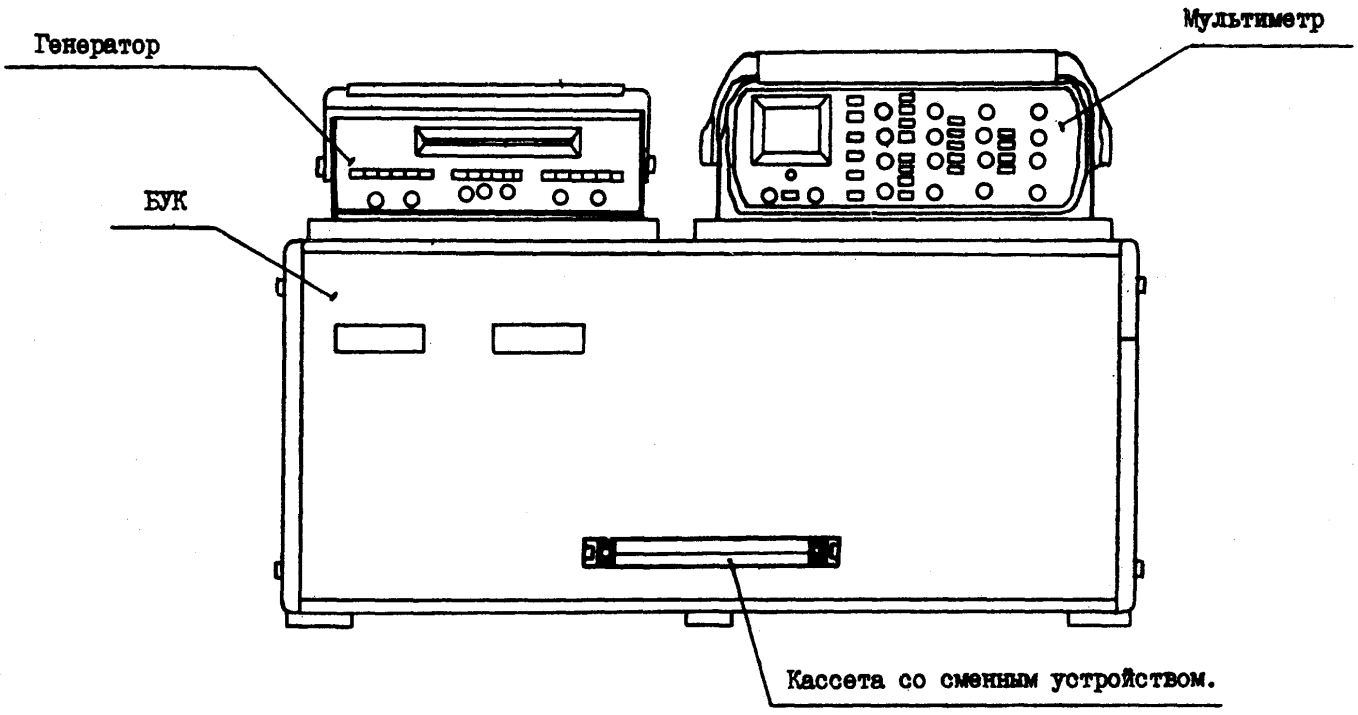


Рисунок 14 – Комплект лабораторного оборудования К32

блока мультиметра К32 (БМ К32) (далее мультиметр); устройства вспомогательного для осциллографов типа Л31 (далее генератор); кассеты и устройств сменных (УС).

БУК состоит из следующих частей (рисунок 15): передней панели (ПП); программатора серии импульсов (ПСИ); блока цифровой индикации (БЦИ); блока аналоговых сигналов (БАС); блока питания (БП).

Органы управления на ПП БУК объединены в группы согласно их функциональному назначению. Обозначение « $\overline{A}|B$ » у кнопок означает, что если кнопка не нажата, то выполняется функция А, а если нажата, то выполняется функция В.

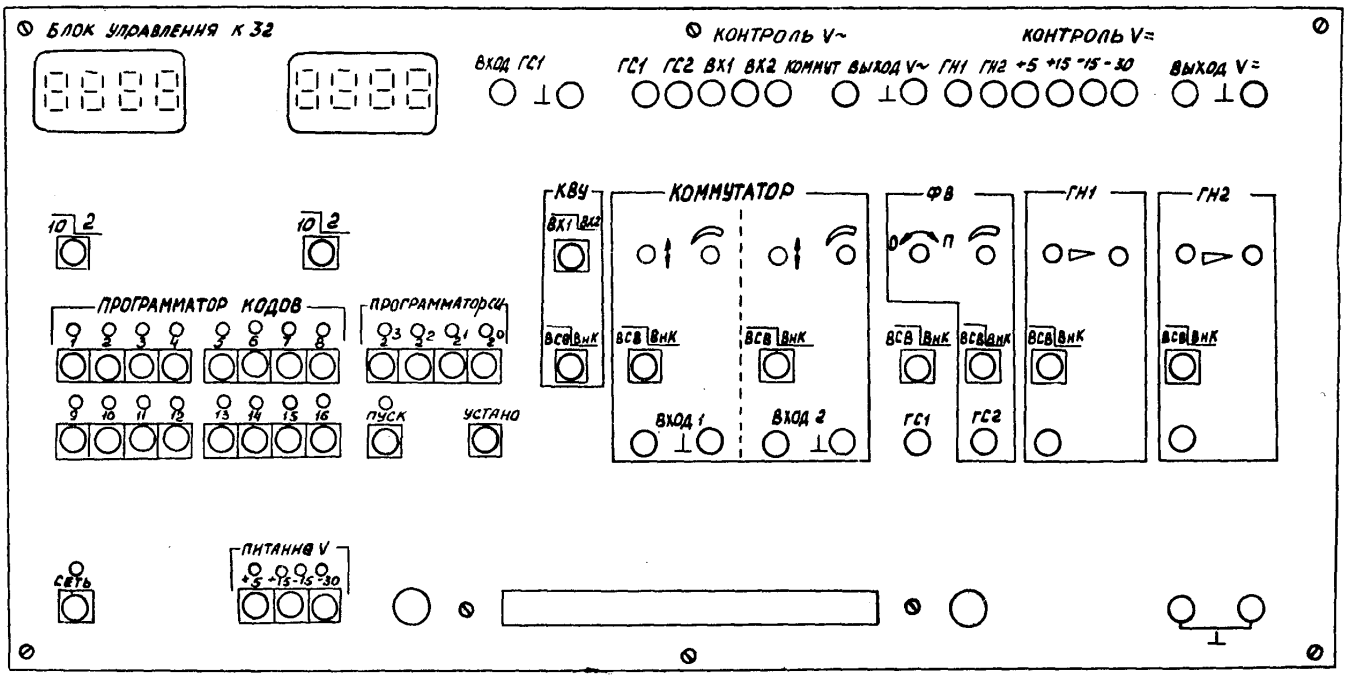


Рисунок 15 – Лицевая панель блока управления комплектом

Светодиоды над кнопками служат для индикации срабатывания кнопок – светодиоды светятся при нажатой кнопке (кроме кнопки «ПУСК») и сигнализации об исправности электрической цепи, которую коммутируют с помощью кнопочного переключателя. Кнопки под надписью «ПРОГРАММАТОР КОДОВ» служат для генерации комбинации сигналов «логический нуль» («0») – постоянного напряжения от 0 до 0,4 В – или «логическая единица» («1») – постоянного напряжения величиной от 2,4 до 5 В, генерация «1» происходит при нажатой кнопке, а «0» – при ненажатой. Кнопки « $\overline{10|2}$ » служат для выбора режима работы БЦИ. Кнопки «2^0», «2^1», «2^2», «2^3» под надписью «ПРОГРАММАТОР СИ» предназначены для набора двоичного кода количества импульсов в серии, генерируемой ГСИ, или номера канала прохождения импульсов (КПИ). Кнопка «ПУСК» служит для включения ГСИ. Кнопка «УСТАН 0» служит для установки ГСИ в исходное состояние. ПСИ предназначен для генерации пачек импульсов от одного до пятнадцати импульсов в серии с амплитудой от 2,4 до 5 В.

ЛАБОРАТОРНАЯ РАБОТА №1

Исследование работы оперативного запоминающего устройства

Цель работы

1. Исследовать основные режимы работы оперативного запоминающего устройства (ОЗУ) статического типа.
2. Получить навыки работы с программатором кодов (ПК), входящим в состав блока управления комплекта типа К32.

Программа работы

1. Исследовать функциональные возможности 4-разрядного ОЗУ типа К155РУ2 в режимах записи и считывания информации.
2. Построить временные диаграммы работы исследованной микросхемы в режимах записи и считывания данных.
3. Пояснить принцип действия исследованной микросхемы при различных режимах работы.
4. Составить и защитить отчет по результатам исследований, в котором должны быть приведены принципиальные схемы, временные диаграммы, таблицы с результатами выполнения работы и выводы с пояснением принципа действия микросхемы.

Описание лабораторной установки

Исследуемая микросхема К155РУ2 (D6) расположена на печатной плате сменного устройства (УС) №14, входящего в комплект лабораторного оборудования К32. Управляющие сигналы подаются с помощью кнопок программатора кодов (ПК), расположенных на лицевой панели БУК. Причем адресация осуществляется кнопками «1» ÷ «4» (СЗР ÷ МЗР), а набор данных – кнопками «5» ÷ «8» (СЗР ÷ МЗР). Выбор микросхемы производится с помощью кнопки «13», а запись данных – «14» (рисунок 16). Генерация «логической единицы» («1») происходит при нажатой кнопке, а «логического нуля» («0») – при ненажатой. Выходные

сигналы индуцируются на индикаторах левого цифрового табло в десятичном коде и на правом табло в двоичном коде. Микросхема DD7 – К155ЛН1.

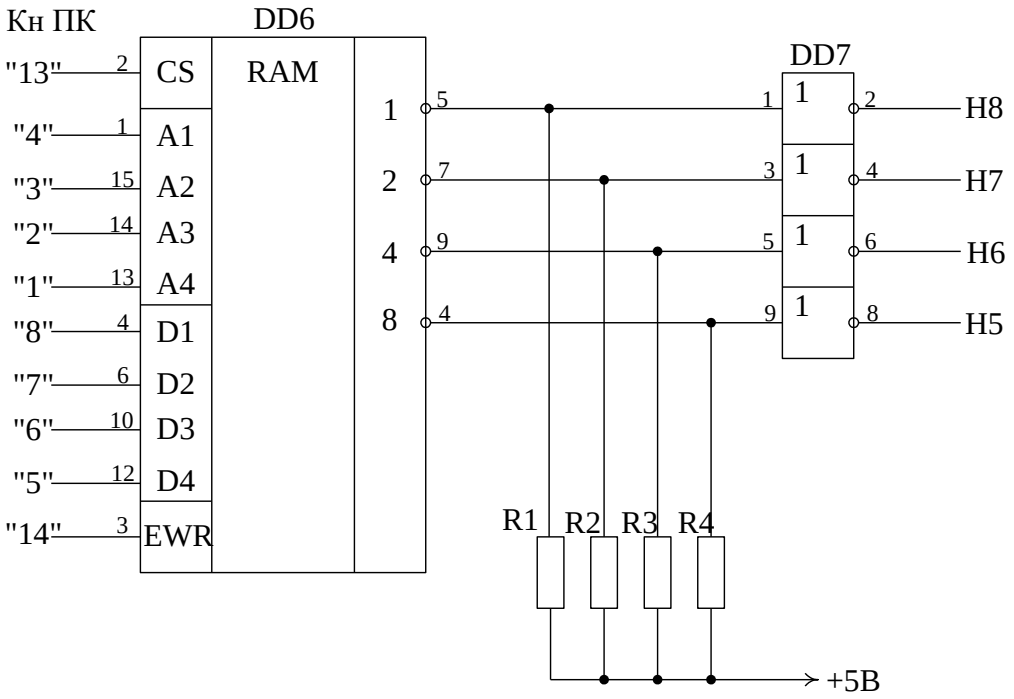


Рисунок 16 – Функциональная схема установки для исследования работы ИМС ОЗУ К155РУ2

Выполнение работы

1 Во входной разъем БУК вставить печатную плату сменного устройства №14.

2 Подключить блок управления к сети напряжения (~220В).

3 Для подачи питания на микросхему нажать кнопки «Сеть» и «+5В».

4 Исследование режимов записи и хранения.

4.1 На входы ИС выбор микросхемы и разрешение записи подать высокий уровень.

4.2 Набрать адрес строки и данные соответствующими кнопками.

4.3 Для записи данных подать на входы ИС выбора микросхемы и разрешения записи напряжение низкого уровня.

4.4 Нажатием кнопок «13» и «14» перевести микросхему в режим хране-

ния.

4.5 Заполнить все строки ЗУ в соответствии с таблицей 5.

5 Исследование режима считывания.

5.1 В режиме хранения задать адрес необходимой строки.

5.2 Отжатием кнопки «13» (выбор микросхемы) осуществить считывание записанной информации.

5.3 Считать информацию по всем адресам и заполнить таблицу 6.

5.4 Повторно считать данные по 5-6 адресам, сравнить результат с п. 3 и сделать выводы о влиянии процесса считывания на хранящуюся в ячейках информацию.

6 По завершении работы отжать кнопки «Сеть» и «+5В», вынуть УС из разъема БУК. Необходимо учитывать, что в режиме считывания выбранные ячейки памяти доступны для приема данных, поэтому логические сигналы на шинах требуется зафиксировать перед переключением уровней управляющих входов от низкого к высокому на входе выбора кристалла.

Контрольные вопросы

1. Основные типы ЗУ.
2. Основные режимы работы ОЗУ.
3. Чем характеризуются статические и динамические параметры ЗУ?

Таблица 5

Адрес				Данные			
A1	A2	A3	A4	D1	D2	D3	D4
СЗР			МЗР	СЗР			МЗР
0	0	0	0	1	1	1	1
0	0	0	1	1	1	1	0
0	0	1	0	1	1	0	1
0	0	1	1	1	1	0	0
0	1	0	0	1	0	1	1
0	1	0	1	1	0	1	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	0	0
1	0	0	0	0	1	1	1
1	0	0	1	0	1	1	0
1	0	1	0	0	1	0	1
1	0	1	1	0	1	0	0
1	1	0	0	0	0	1	1
1	1	0	1	0	0	1	0
1	1	1	0	0	0	0	1
1	1	1	1	0	0	0	0

Таблица 6

Адрес				Данные			
A1	A2	A3	A4	D1	D2	D3	D4
СЗР			МЗР	СЗР			МЗР

ЛАБОРАТОРНАЯ РАБОТА №2

Исследование работы репрограммируемого постоянного запоминающего устройства

Цель работы:

1. Исследовать основные режимы репрограммируемого постоянного запоминающего устройства (РПЗУ).

Программа работы

1. Исследование функциональных возможностей РПЗУ типа КР1601РР1 в режимах записи, хранения, стирания и считывания информации
2. Построить временные диаграммы работы исследованной микросхемы в режимах записи, хранения, стирания и считывания информации.
3. Пояснить принцип действия исследованной микросхемы при различных режимах работы.
4. Составить и защитить отчет по результатам исследований, в котором должны быть приведены принципиальные схемы, временные диаграммы, таблицы с результатами выполнения работы и выводы с пояснением принципа действия микросхемы.

Описание лабораторной установки

Исследуемая интегральная микросхема (ИС) КР1601РР1 (DD15) расположена на печатной плате сменного устройства (УС) № 19, входящего в комплект лабораторного оборудования К32. Управляющие сигналы подаются с помощью кнопок программатора кодов (ПК) и программатора серии импульсов (ПСИ). Адресация осуществляется кнопками «7» - «16» (МЗР + СЗР). Данные, предназначенные для записи, формируются на выходах асинхронного счетчика DD 4 (К155 ИЕ5), на тактовый вход которого поступают импульсы с ПСИ. Содержание счетчика отображается на левом цифровом табло. Обнуление счетчика производится нажатием кнопки «6». Считываемые данные по выбранному адресу после нажатия

кнопки «4» индицируются на правом табло. Запись информации осуществляется кнопкой «3». Кнопки «1» и «2» предназначены соответственно для общего и построчного стирания. Выбор кристалла производится нажатием кнопки «5».

Выполнение работы

1. Во входной разъем БУК вставить сменное устройство №19.
2. Подключить блок управления к сети напряжения (~220).
3. Для подачи питания на микросхемы нажать кнопки «Сеть», «+5В», «±15В» и «30В».
4. Исследование режима программирования.

   На вход ИС «выбор микросхемы» подать высокий уровень.

   Нажатием кнопки «1» произвести общее стирание информации в ячейках ЗУ (при этом во все ячейки запишутся единицы). Обнулить счетчик.

   С помощью ПСИ заполнить счетчик.

   Набрать произвольный адрес кнопками ПК.

   Осуществить запись нажатием кнопки «3».

   Обнулить счетчик.

   Произвести запись данных по 15 произвольным адресам, соблюдая последовательность в действиях согласно п.п. 4.3 - 4.6.

   Исследование режима хранения информации.

   Подать на вход «выбор микросхемы» низкий уровень.

   Отключить питание. Необходимо соблюсти следующую последовательность отключения: «30В», «±15В», «+5В».

   Подать питание на микросхему. При этом последовательность включения будет обратной последовательности п. 4.1.
5. Исследование режима считывания.

   Задать адрес необходимой строки.

   Нажать кнопку «4» и проверить содержимое ячеек.

   Согласно п.п. 6.1 - 6.2. считать записанную ранее информацию. Сде-

лать выводы.

6. Исследование режима построчного стирания.

Осуществить построчное стирание по любым 5 ранее задействованным адресам.

Убедиться, что стирание информации было произведено только по данным адресам. Сделать выводы.

7. Исследование режима общего стирания.

После нажатия кнопки «1» убедиться в том, что информация удалена.

8. Отключить питание согласно п. 5.2, отжать кнопку «Сеть». Вынуть УС из разъема БУК.

Контрольные вопросы

1. Основные типы ЗУ и их отличительные особенности.
2. Основные режимы работы исследуемого РПЗУ.
3. Почему соблюдается строгая последовательность включение/отключение питания ИС КР1601РР1.

ЛАБОРАТОРНАЯ РАБОТА №3

Исследование работы аналого-цифрового преобразователя

Цель работы

1. Исследовать работу аналого-цифрового преобразователя (АЦП) двухтактного интегрирования.

Программа работы

1. Исследования функциональных возможностей АЦП (интегральной микросхемы (ИС) типа КР572ПВ2).
2. Составить и защитить отчет по результатам исследований, в котором должны быть приведены принципиальные схемы, графики, расчеты погрешностей, таблицы с результатами выполнения работы и выводы с пояснением принципа действия микросхемы.

Описание лабораторной установки

Принципиальная электрическая схема лабораторной установки приведена на рисунке 17.

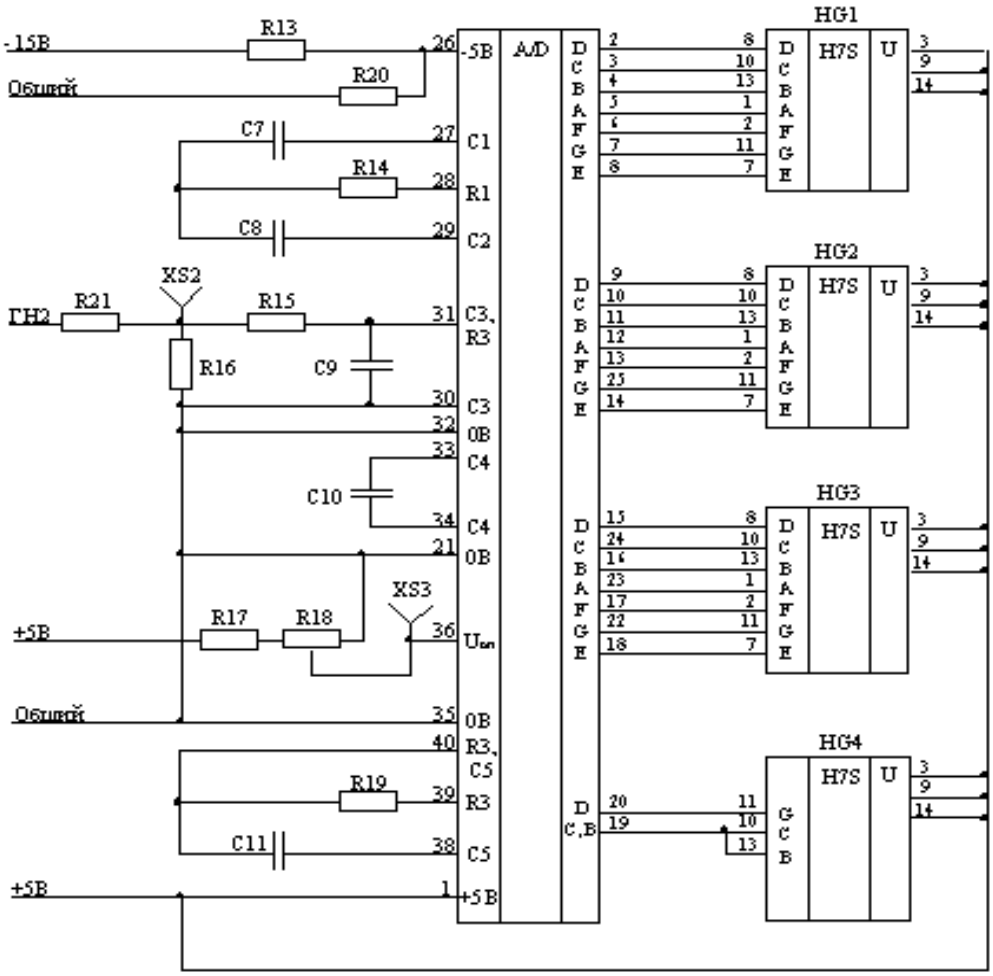


Рисунок 17 – Функциональная схема установки для исследования работы ИС АЦП КР572ПВ2

Исследуемая интегральная микросхема (ИС) КР572ПВ2 (DD6) расположена на печатной плате сменного устройства (УС) №18, входящего в комплект лабораторного оборудования К32. Аналоговый сигнал на входе АЦП регулируется с помощью генератора ГН2 (ручки расположены на лицевой панели блока управления БУК). Входное напряжение контролируется в гнезде XS2. Для контроля опорного напряжения используется гнездо XS3 (смотри рисунок 17). Выходной код отображается на индикаторах HG1 – HG4, расположенных на УС №18.

Выполнение работы

1. Во входной разъем БУК вставить печатную плату сменного устройства №18.

2. Подключить блок управления к сети напряжения (~220 В).

3. Для подачи питания на микросхемы нажать кнопки «Сеть», «+5В» и «±15В».

4. Определение разрешающей способности АЦП при Uоп = const.

Измерить опорное напряжение в гнезде XS3.

С помощью генератора напряжения ГН2 установить входное напряжение равное Uоп и снять показание индикаторов HG1 – HG4, расположенных на УС.

Повторить действия по п. 4.2, изменяя входное напряжение на величину 0,1Uоп (достаточно снять 5 значений).

Для полученных значений найти отношение приращений входного сигнала и выходного кода. Найти среднее значение разрешающей способности АЦП.

Воспользовавшись формулой (1.3.6), определить разрешающую способность АЦП и сравнить с экспериментальным значением. Сделать выводы.

5. Определение интегральной нелинейности АЦП.

Установить значение входного напряжения, выходной код при котором равен «-1999».

Зафиксировать значения выходного кода при увеличении входного сигнала с шагом 0,2Uоп. Предельное значение входного напряжения должно соответствовать коду «1999».

Построить в одних осях зависимости Nx = f(Uвх) для идеального и реального АЦП (в относительных единицах).

Определить графически погрешность интегральной нелинейности АЦП. Сделать выводы.

На основании построенных графиков рассчитать значения

$$\operatorname{tg}\alpha_{min},\ \operatorname{tg}\alpha_{ср},\ \operatorname{tg}\alpha_{max}.$$

Определить погрешность нелинейности по формуле (1.3.2) и сделать выводы.

Определение погрешности полной шкалы.

Изменяя входное напряжение, установить на цифровом табло код «1999». Замерить напряжение, соответствующее этому коду.

Найти расчетное значение напряжения полной шкалы.

Определить абсолютную и относительную погрешности полной шкалы. Сделать выводы.

6. Отключить питание «+5В» и «±15В», отжать кнопку «Сеть». Вынуть УС из разъема БУК.

Контрольные вопросы

1. Назначение аналого-цифровых преобразователей.
2. Какие виды АЦП существуют.
3. Основные параметры АЦП.
4. Критерии выбора ИС АЦП.

ЛАБОРАТОРНАЯ РАБОТА №4
Исследование работы цифро-аналогового преобразователя

Цель работы

1. Исследовать работу умножающего цифро-аналогового преобразователя (ЦАП).

Программа работы

1. Исследование функциональных возможностей ЦАП (интегральной микросхемы (ИС) типа КР572ПА1А).

2. Составить и защитить отчет по результатам исследований, в котором должны быть приведены принципиальные схемы, графики, расчеты погрешностей, таблицы с результатами выполнения работы и выводы с пояснением принципа действия микросхемы.

Описание лабораторной установки

Принципиальная электрическая схема лабораторной установки приведена на рисунке 18.

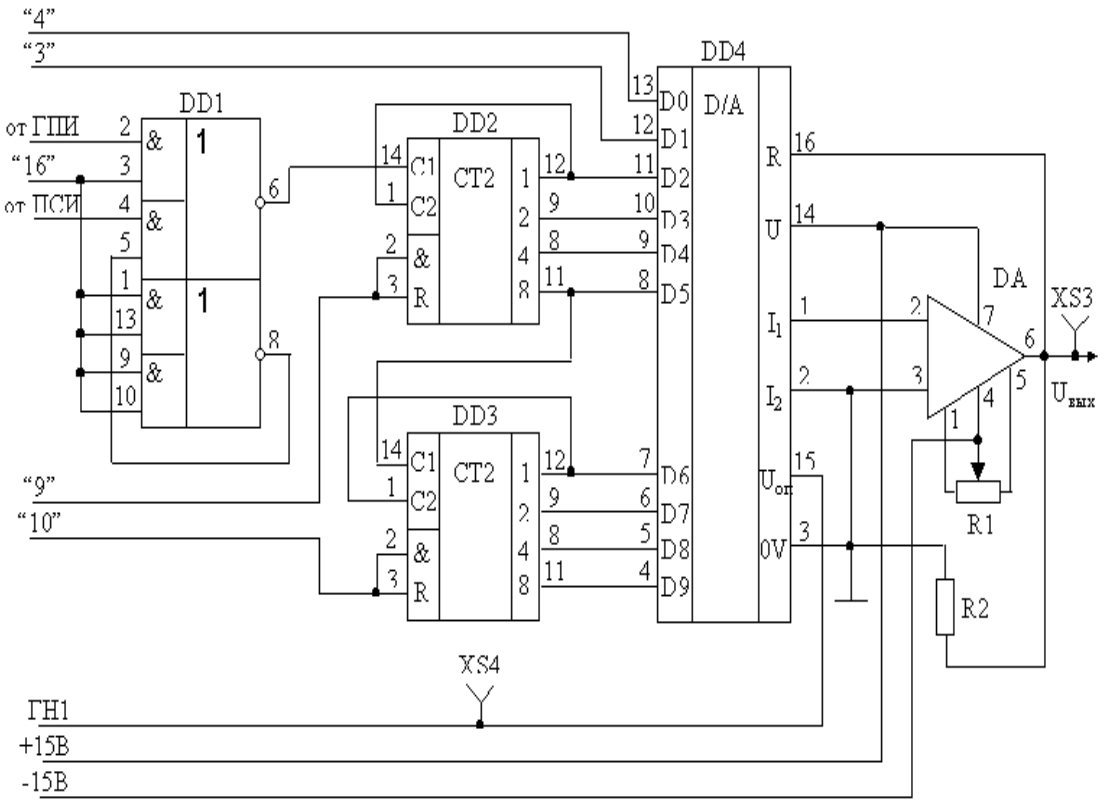


Рисунок 18 – Функциональная схема установки для исследования работы ИС ЦАП КР572ПА1А (DD4)

Исследуемая интегральная микросхема (ИС) КР572ПА1А (DD4) расположена на печатной плате сменного устройства (УС) №11, входящего в комплект лабораторного оборудования К32. Управляющие сигналы подаются с помощью кнопок "4", "3" программатора кодов (ПК), ПСИ и генератора прямоугольных импульсов (ГПИ) через буферные счётчики DD2 и DD3 (К155ИЕ5) (рисунок 18).

Импульсы, поступающие с ГПИ на тактовый вход асинхронного счётчика, приводят к изменению двоичного кода на цифровых входах ЦАП. В случае работы ПСИ количество тактовых импульсов определено набранным двоичным кодом посредством кнопок «2^0», «2^1», «2^2», «2^3», расположенных на передней панели БУК. Для генерации серии импуль-

сов необходимо нажать кнопку «ПУСК» (над ней начинает светиться светодиод). При нажатии кнопки «УСТАН 0» светодиод перестаёт светиться и ПСИ готов к передаче новой пачки импульсов. Например, требуется сгенерировать серию из 11-импульсов. Для этого необходимо осуществить следующую последовательность нажатия кнопок:

«УСТАН 0» $\rightarrow$ «2^3» + «2^1» + «2^0» =11 $\rightarrow$ «ПУСК».

Работу ПСИ можно прекратить кнопкой “16” (нажатое состояние), обеспечивая непосредственное прохождение импульсов с ГПИ на тактовый вход счётчика D6 и непрерывное изменение его содержимого, а значит информации на цифровых входах ЦАП (D2 – D9).

Сброс счётчика осуществляется либо автоматически при поступлении импульса на вход заполненного счётчика, либо нажатием кнопок “9”и “10” независимо от его состояния.

Выходной сигнал исследуется с помощью мультиметра в гнезде XS3. Величина опорного напряжения (Uоп) контролируется в гнезде XS4. Левое цифровое табло индицирует содержимое счётчиков в десятичном коде. На третьем и четвертом индикаторах правого цифрового табло отображается номер состояния двух младших разрядов ЦАП (D0 и D1) также в десятичном коде (таблица 7).

Таблица 7 – Индикация состояний, соответствующих положениям кнопок “3” и “4”, на правом цифровом табло.

№ состояния	Кн. “3”	Кн. “4”
01	0	0
02	0	1
03	1	0
04	1	1

Выполнение работы

1. Во входной разъём БУК вставить печатную плату сменного устрой-

ства №11.

2. Подключить блок управления к сети напряжения (~ 220 В).

3. Для подачи питания на микросхемы нажать кнопки “Сеть”, “+5 В” и “±15 В”.

4. Определение разрешающей способности ЦАП при $U_{оп}$ = const.

Кратковременным нажатием кнопок «9» и «10» обнулить содержимое счётчиков DD2 и DD3.

Вращением ручек генератора напряжения “ГН 1” установить величину опорного напряжения $U_{оп}$ = 10 В.

Измерить с помощью вольтметра напряжение на выходе ОУ при соответствующем двоичном коде на входах ЦАП.

Повторить действия по пп. 5.4.1 ÷ 5.4.3 для $U_{оп}$ = 7; 5; 3 В. Заполнить таблицу 5.

Определить среднее приращение выходного напряжения при изменении входного кода на единицу младшего разряда по экспериментальным данным для всех значений $U_{оп}$ (таблица 8).

Найти расчётное значение шага квантования и сравнить с экспериментальным.

Найти разрешающую способность ЦАП как величину, обратную максимальному количеству градаций выходного сигнала.

Таблица 8 – Результаты измерений выходного напряжения ЦАП

при $U_{оп}$ = 10; 7; 5; 3 В

Двоичный код на входе ЦАП				Напряжение на выходе ОУ, мВ			
D3	D2	D1	D0	$U_{оп}$ = 10В	$U_{оп}$ = 7В	$U_{оп}$ = 5В	$U_{оп}$ = 3В
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	1				
0	1	1	0				
0	1	1	1				

5. Определение дифференциальной нелинейности ЦАП.

С помощью ПСИ и кнопок “3”, “4” последовательно подать на вход ЦАП два кода: 0111111111 и 1000000000, зафиксировав соответствующие значения выходного напряжения при $U_{оп}$ = 10; 7; 5; 3 В

По формуле (1.4.2) рассчитать дифференциальную нелинейность и сравнить с паспортным значением. Сделать выводы.

6. Определение погрешности нелинейности при $U_{оп}$ = 10 В.

Заполнить таблицу 9.

Построить графики зависимостей $U_{вых}$ = f(N) идеального ЦАП и, воспользовавшись данными таблицы 6, реального ЦАП (в относительных единицах).

Определить графически погрешность нелинейности. Сделать выводы.

На основании построенных графиков рассчитать значения

$$\operatorname{tg}\alpha_{min},\ \operatorname{tg}\alpha_{ср},\ \operatorname{tg}\alpha_{max}.$$

Определить погрешность нелинейности по формуле (1.4.3) и сделать выводы.

Таблица 9 – Результаты измерения $U_{вых}$ при $U_{оп}$ = 10В

Двоичный код на входе ЦАП	Выходное напряжение $U_{вых}$, В	Двоичный код на входе ЦАП	Выходное напряжение $U_{вых}$, В
0000000000		1000010000	
0000110000		1001000000	
0001100000		1001110000	
0010010000		1010100000	
0011000000		1011010000	
0011110000		1100000000	
0100100000		1100110000	
0101010000		1101100000	
0110000000		1110010000	
0110110000		1111000000	
0111100000		1111111111	

7. Определить погрешность нуля.

8. Определение погрешности полной шкалы.

Из таблицы 6 взять значение выходного напряжения при входном коде 1111111111.

Найти расчётное значение напряжения полной шкалы $U_{пш}$.

Определить абсолютную и относительную погрешности полной шкалы. Сделать выводы.

9. Определение тактовой частоты ГПИ при $U_{оп}$ = 10 В.

Нажать кнопку “16”.

С помощью осциллографа измерить интервал времени Δt, за который выходное напряжение изменяется от нуля до своего максимального значения.

Определить тактовую частоту ГПИ по формуле:

$$f_T = \frac{2^n}{\Delta t}, \quad \text{где} \quad n = 8.$$

Контрольные вопросы

1. Назначение цифро-аналоговых преобразователей.
2. Какие виды цифро-аналоговых преобразователей существуют? Критерии классификации.
3. Основные параметры ЦАП.
4. Критерии выбора ИС ЦАП.

ЛАБОРАТОРНАЯ РАБОТА №5
Исследование работы арифметико-логического устройства

Цель работы

1. Исследовать работу арифметико-логического устройства (АЛУ).

Программа работы

1. Исследование функциональных возможностей АЛУ (интегральной микросхемы (ИС) типа К155ИП3).
2. Составить и защитить отчет по результатам исследований, в котором должны быть приведены принципиальные схемы, таблицы с результатами выполнения работы и выводы с пояснением принципа действия микросхемы.

Описание лабораторной установки

Исследуемая интегральная микросхема (ИС) К155ИП3 (DD1) расположена на печатной плате сменного устройства (УС) № 11, входящего в комплект лабораторного оборудования К32. Управляющие сигналы подаются с помощью кнопок программатора кодов (ПК), расположенных на лицевой панели БУК. Выбор функции осуществляется кнопками «12» – «9» (см. рисунок 19). Значение первого операнда А задается кнопками «4» – «1», а второго В – кнопками «8» – «5». Выбор режима работы задается кнопкой «13». На входе «Перенос» исследуемой микросхемы значение задается кнопкой «14». Выходные сигналы отображаются на левом и

правом цифровом табло в двоичном коде и на четвертом и третьем индикаторах правого цифрового табло в десятичном коде.

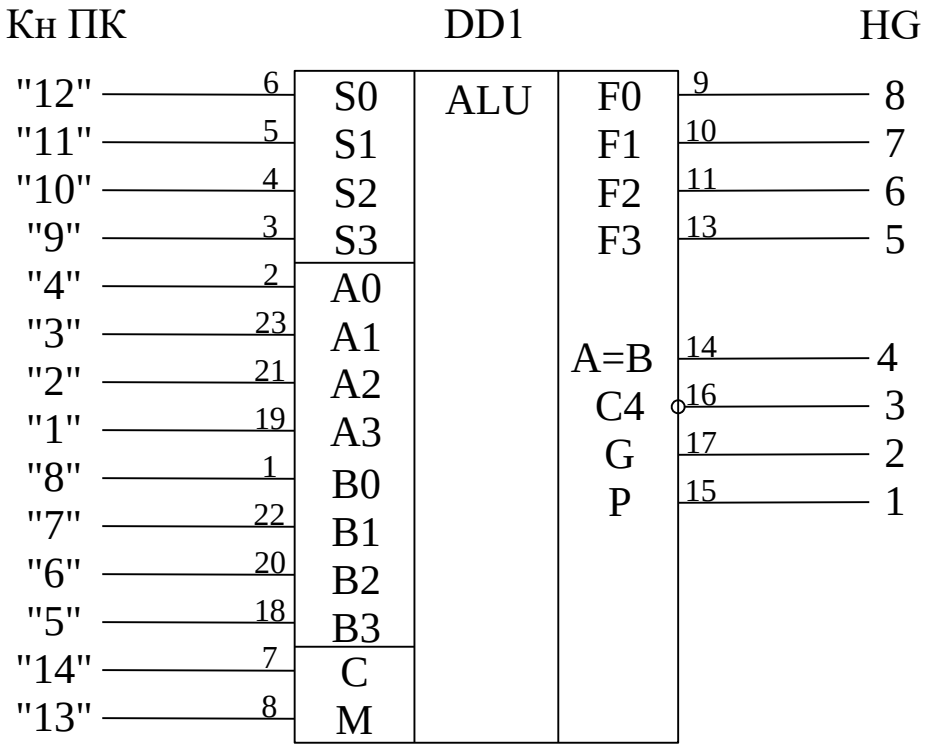


Рисунок 19 – Функциональная схема установки для исследования ИМС АЛУ К155ИП3

Выполнение работы

1. Во входной разъём БУК вставить печатную плату сменного устройства №11.
2. Подключить блок управления к сети напряжения (~ 220 В).
3. Для подачи питания на микросхемы нажать кнопки “Сеть”, “+5 В” и “±15 В”.
4. Реализовать все арифметические и логические операции над операндами, которые соответствуют варианту задания (таблица 10). Результаты вычислений представить в таблице, по форме подобной таблице 2.

Контрольные вопросы

1. Классифицируйте современные АЛУ.
2. Изобразите структуру одноразрядного АЛУ.
3. Как реализуется операция умножения в современных АЛУ?

Таблица 10 – Варианты задания для исследования АЛУ

№ №	A	B	C	№ №	A	B	C	№ №	A	B	C
1	15	10	1	14	5	14	1	27	11	11	1
2	15	12	0	15	5	15	0	28	11	12	0
3	15	13	1	16	5	16	1	29	11	13	1
4	5	4	0	17	11	1	0	30	9	4	0
5	5	5	1	18	11	2	1	31	9	5	1
6	5	6	0	19	11	3	0	32	9	6	0
7	15	7	1	20	11	14	1	33	9	7	1
8	5	8	1	21	11	5	1	34	9	8	1
9	15	9	0	22	11	6	0	35	9	9	0
10	5	10	1	23	11	7	1	36	9	10	1
11	5	11	0	24	11	8	0	37	9	11	0
12	15	12	1	25	11	9	1	38	9	12	1
13	5	13	1	26	11	10	1	39	9	13	1

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Невинномысский технологический институт (филиал)

Методические указания по выполнению практических работ по дисциплине «Компьютерная и микропроцессорная техника в электроприводе»

Направление подготовки 13.03.02 – Электроэнергетика и электротехника
Профиль подготовки – Электропривод и автоматика
Квалификация выпускника – бакалавр

Невинномысск 2019

Методические указания предназначены для выполнения практических работ по дисциплине «Компьютерная и микропроцессорная техника в электроприводе» для студентов направления подготовки 13.03.02 «Электроэнергетика и электротехника» и соответствуют требованиям ФГОС ВО направления подготовки бакалавров.

Составитель: Д.В. Самойленко

Содержание

Изучение программного пакета TASM32 4
Изучение группы арифметических команд 8
Изучение группы логических команд 11
Изучение групп команд безусловных и условных переходов и команд управления циклом 13
Изучение команд работы с подпрограммами 16
Изучение групп команд сравнения и логических сдвигов 18
Изучение группы цепочных команд 21

Практическое занятие № 1
Изучение программного пакета TASM32

Цель работы: "С помощью программного пакета TASM32 изучить программную модель микропоцессора i8086, особенности программирования на языке ассемблера и особенности выполнения группы команд пересылок, приобрести навыки отладки программ".

Краткие теоретические сведения:

Ассемблер - язык низкого уровня приближенный к аппаратным средствам компьютера и его натуральным возможностям. Следствием этого является то, что он индивидуален для каждого семейства ЭВМ.

Программы, написанные на языке ассемблера, отличаются высокой эффективностью, то есть минимальным объемом и максимальным быстродействием. Это обстоятельство обусловило широкое использование языка ассемблера в тех случаях, когда скорость работы программы или расходуемая ею память, имеют решающее значение. Некоторые классы программ (например, программы устанавливаемых драйверов устройств, отличаются жесткой структурой) требуют для своего составления обязательного использования языка ассемблера. С другой стороны, поскольку современные системы программирования позволяют объединять в одну выполнимую программу фрагменты, написанные на разных языках, широко практикуется составление комбинированных программ, в которых основная часть написана на языке высокого уровня, а наиболее критические части на ассемблере. Может использоваться и обратный метод, когда в программу на языке ассемблера вставляют фрагменты для выполнения относительно сложных логических или математических преобразований, записанных на языке высокого уровня.

Однако, кроме потребительских качеств, язык ассемблера имеет еще и значительную методическую ценность, отражая архитектурные особенности и режимы работы используемого в компьютере микропроцессора (МП). Язык ассемблера предоставляет уникальную возможность изучения машины на "низком уровне", освоение того, что и как умеет делать аппаратура компьютера и что вносит в его работу операционная система.

Программная модель

В программную модель входят узлы доступные программисту.

Регистры общего назначения

AH	AL	Аккумулятор
BH	BL	Базовый регистр
CH	CL	Счетчик
DH	DL	Регистр данных

Регистры - указатели

SI	Индекс источника
DI	Индекс приемника
BP	Указатель базы
SP	Указатель стека

Сегментные регистры

CS	Регистр сегмента команд
DS	Регистр сегмента данных
ES	Регистр дополнительного сегмента данных
SS	Регистр сегмента стека

Прочие регистры

IP	Указатель команд
FLAGS	Регистр флагов

Процессор содержит 12 шестнадцатиразрядных програмно-адресуемых регистров, которые принято объединять в три группы: регистры общего назначения (РОН), регистры-указатели и сегментные регистры. Кроме того, в состав процессора входят счетчик команд и регистр флагов.

РОН используются для временного хранения любых объектов (данных, адресов) и выполнения над ними требуемых операций. При этом регистры допускают независимое обращение к старшим (AH, BH, CH, DH) и младшим половинам (AL, BL, CL, DL).

Основное назначение индексных регистров SI и DI хранить индексы (смещения) относительно некоторой базы (т. е. начала массива) при выборке операнда из памяти.

Регистр BP, как правило, служит указателем базы при работе с данными в стековых структурах.

Регистр SP используется исключительно как указатель вершины стека.

Регистры - указатели в отличие от РОН не допускают побайтовую адресацию.

Сегментные регистры хранят начальные адреса сегментов программы и, тем самым, обеспечивают возможность обращаться к этим сегментам.

Регистр IP указывает на относительный адрес команды.

Регистр флагов содержит информацию о текущем состоянии МП.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X	X	X	X	OF	DF	IF	TF	SF	ZF	X	AF	X	PF	X	CF

CF- флаг переноса, PF- флаг паритета (четности), AF - флаг дополнительного переноса, ZF- флаг нуля, SF- флаг знака, TF - управляющий флаг трассировки, IF - управляющий флаг разрешения прерываний, DF - управляющий флаг направления, OF - флаг переполнения.

Структура программы

Программа на ассемблере представляет собой совокупность блоков памяти, называемых сегментами памяти. Программа может состоять из одного или нескольких таких блоков-сегментов. Каждый сегмент содержит совокупность предложений языка, каждое из которых занимает отдельную строку кода программы.

Предложения, составляющие программу, могут представлять собой синтаксическую конструкцию, соответствующую команде, макрокоманде, директиве или комментарию. Для того чтобы транслятор ассемблера мог распознать их, они должны формироваться по определенным синтаксическим правилам.

Предложение на языке Ассемблера может состоять из 4^x полей следующего вида:

[Метка:] Мнемокод [Операнд] [;Комментарий]

Квадратные скобки вокруг полей метки, операнда и комментария показывают, что эти поля не обязательны. Поля отделяются друг от друга хотя бы одним пробелом (или символом табуляции).

Поле метки служит для присваивания имени команде языка ассемблера. По нему на эту команду могут ссылаться другие команды программы. Метка должна заканчиваться двоеточием.

Поле мнемокода содержит имя команды микропроцессора.

Поле комментариев позволяет описать назначение команд исходной программы для обеспечения ее понимания. Перед комментарием указывается точка с запятой (;).

Группа команд передачи данных

Мнемокод	Операнд	Комментарий
MOV	dst,src	dst - приемник, а src - источник данных (src) → (dst) , прим. MOV DX,CX
XCHG	dst,src	Происх. Обмен между операндами (dst) ↔ (src)
LAHF		Загрузить в регистр AH флаги (разр 0-7)
SAHF		Загрузить флаги из регистра AH

Команда MOV осуществляет пересылки следующих видов:

- из РОН в РОН;
- из ячейки памяти в РОН;
- из РОН в ячейку памяти;
- непосредственный операнд в РОН;
- непосредственный операнд в ячейку памяти;
- из РОН в сегментный регистр;
- из сегментного регистра в РОН;
- из сегментного регистра в ячейку памяти.

По команде MOV исключаются пересылки следующих видов:

- из ячейки памяти в ячейку памяти. (MOVS);
- непосредственный операнд в сегментный регистр;
- из сегментного регистра в сегментный регистр;

Пример 1:Выполнить пересылки следующих видов: число 72Fh → (BX), (BL) → (AH), F4h → (AL), (CX) ↔ (AX), (BX) → (DI), 1FCh в ячейку памяти с адресом ds:0004h

```
code    segment                         ; начало сегмента кода
        assume      cs:code, ds:code    ; директива, кот. связывает сег-
                                        ; ментные регистры с именем сегмента
start:
        mov   bx,072fh                  ; число 72Fh → BX
        mov   ah,bl
        mov   al,0f4h
        xchg  ax,cx                     ; CX ↔ AX
        mov   di,bx
        mov   ds:0004h,1fch
code    ends                            ; конец сегмента кода
end     start                           ; конец программы с точкой входа
```

Инструкция по работе с программным пакетом TASM32

1. В текстовом редакторе создать файл с расширением asm. Например, pr.asm.
(в программном пакете FAR создать файл — Shift+F4, открыть существующий файл для редактирования — F4, открыть существующий файл для просмотра без редактирования — F3, сохранить файл — F2, выйти из редактора — Esc).
2. Откомпилировать файл, набрав в командной строке:
   tasm.exe /m /zi /l pr.asm
3. Проверить наличие ошибок, для чего убрать панели управления (Ctrl+O). Если есть ошибки, то исправить их в редакторе и повторить пункт 2.
4. Если ошибок нет, убедится в создании файла pr.obj
5. Произвести компоновку программы, набрав в командной строке:
   tlink.exe /v pr.obj
6. Запустить отладчик, набрав в командной строке:
   td.exe pr.exe
7. Войти в меню, нажав (Alt+F), выбрать директиву View, CPU.
8. Пошаговый просмотр программы осуществляется нажатием F7.

Задание на практическое занятие № 1

Выполнить пересылки следующих видов:

№ вар.	Выполнить пересылки						
1	F516H→AX	AH→BL	AX↔CX	34D1H→ ds:0006H	CX→SI	3AH→BH	F412H→DS
2	2CD3H→BX	BL→AH	D2H→BH	AX↔BX	AX→DI	ds:0003H→CX	0056H→ES
3	D206H→CX	CH→BL	ds:0023H→ DX	56H→BH	DX↔BX	DX→SI	34AAH→DS
4	A925H→DX	DH→CL	DX→BX	BX→ ds:0004H	53H→CH	BX↔CX	BX→DI
5	465AH→DX	DL→BH	D2H→BL	ds:0056H→ CX	CX→DI	BX→SI	BX↔CX
6	78D3H→CX	CL→AH	CX↔DX	DX→DI	F5H→AL	AX→ds:0003H	002FH→ES
7	C649H→BX	BH→CL	25H→CH	DX↔BX	DX→SI	ds:0002H→BX	CX→DS
8	38EFH→AX	AL→DL	DX→BX	BX→ ds:0005H	AX↔BX	BX→DI	0067H→ES
9	E36DH→AX	AH→CH	35H→CL	CL→ds:0009H	AX→DX	CX↔AX	DX→SI
10	E846H→BX	BH→DL	BX→DI	6FH→DH	DX→CX	ds:0020H→DX	DX↔CX
11	49FAH→CX	CL→AL	CX↔ DX	25H→SI	DH→BL	ds:0008H→ ds:0009H	DX→BX
12	24DBH→DX	DH→AL	25H→AH	AX↔DX	AX→DI	DL→ ds:0002H	0067H→DS
13	B531H→DX	F4H→BH	DL→BL	ds:0020H ↔ ds:0021H	DL↔AL	AX→SI	SI→CX
14	63FBH→CX	CL→AL	C3H→AH	AX→BX	CH→DH	ds:0012H↔ ds:0013H	CX→DI
15	74ACH→BX	BH→AL	AX→SI	E8H→DX	DL→CH	ds:0021H→CL	DI↔SI

Практическое занятие № 2
Изучение группы арифметических команд

Цель работы: "Изучить особенности выполнения арифметических команд, разработка алгоритма, составление и отладка программы с использованием этих команд.

Краткие теоретические сведения:

Часть 1

Мнемокод	Операнд	Комментарий
ADD	Op1,Op2	Сложение значений Op1 и Op2, результат помещается в Op1. (Op1)+(Op2) → (Op1)
ADC	Op1,Op2	Сложение значений Op1 и Op2 с учетом флага переноса CF, результат помещается в Op1. (Op1)+(Op2)+(CF) → (Op1)
SUB	Op1,Op2	Вычитание значений Op1 и Op2, результат помещается в Op1. (Op1)-(Op2) → (Op1)
SBB	Op1,Op2	Вычитание значений Op1 и Op2 с учетом заема, результат помещается в Op1. (Op1)-(Op2)-(CF) → (Op1)
INC	Op	Инкремент операнда Op . Содержимое Op увеличивается на1. (Op) + 1→ (Op)
DEC	Op	Декремент операнда Op . Содержимое Op уменьшается на 1. (Op) - 1→ (Op)
DAA		Коррекция результата сложения для представления в десятичном виде. AL
DAS		Коррекция результата вычитания для представления в десятичном виде. AL

Примечание: Операции с многобайтными числами производятся по байтам, начиная с младших. При сложении (вычитание) многобайтных чисел младшие байтных чисел младшие байты складываются (вычитаются) командой ADD (SUB) все последующие старшие байты командой ADC (SBB R).

МП может работать с двоичными и с двоично-десятичными числами. Но т.к. двоично-десятичные числа складываются (вычитаются) на двоичном сумматоре, то требуется коррекция результата, для этого используется команда DAA(DAS) - десятичная коррекция.

Часть 2

Мнемокод	Операнд	Комментарий
MUL	Op	Целочисленное умножение без учета знака (AL)*(Op8) → (AX) (AX)*(Op16) → (DX):(AX)
DIV	Op	Беззнаковое деление (AX)/(Op8) → частное а (AL), остаток в (AH) (DX)(AX)/(Op16) → частное а (AX), остаток в (DX)
AAM		Коррекция после умножения двух неупакованных BCD чисел. Преобразование двоичного числа меньшего 63h (99_{10}) в его неупакованный BCD-эквивалент.
AAD		Коррекция перед делением двух неупакованных BCD чисел. Преобразование двузначного неупакованного BCD-числа меньшего 63h (99_{10}) в двоичное представление.

Команда MUL выполняет умножение двух операндов без учета знаков. Алгоритм зависит от формата операнда команды и требует явного указания местоположения только од-

ного сомножителя, который может быть расположен в памяти или в регистре. Местоположение второго сомножителя фиксировано и зависит от размера первого сомножителя.

Для команды DIV необходимо задание двух операндов — делимого и делителя. Делимое задается неявно и размер его зависит от размера делителя, который указывается в команде. При выполнении операции деления возможно возникновение исключительной ситуации: 0 — ошибка деления. Эта ситуация возникает в одном из двух случаев: делитель равен 0 или частное слишком велико для его размещения в регистре ax/al.

Задание на практическое занятие № 2 (Часть 1)

Вариант 1

Вычислить выражение X + Y - Z, где X = 2860; Y = 88; Z =56, X,Y,Z - десятичные числа. Результат разместить в ячейках памяти, начиная с адреса ds:0005H.

Вариант 2

Найти сумму двух трехбайтных чисел A5C865H и 74D6CAH. Результат разместить в ячейках памяти, начиная с адреса ds:0010H.

Вариант 3

Найти сумму двух десятичных чисел 2357 и 9599. Результат разместить в ячейках памяти, начиная с адреса ds:0013H.

Вариант 4

Вычислить выражение X + Y - Z, где X = 4792; Y = 6888; Z =1014; X,Y,Z - десятичные числа. Результат разместить в ячейках памяти, начиная с адреса ds:0011H.

Вариант 5

Найти разность двух трехбайтных чисел D48B15H и 81A7E4H. Результат разместить в ячейках памяти, начиная с адреса ds:0003H.

Вариант 6

Найти разность двух десятичных чисел 18723 и 9569. Результат разместить в ячейках памяти, начиная с адреса ds:0004H.

Вариант 7

Найти сумму двух десятичных чисел 9092 и 2624. Результат разместить в ячейках памяти, начиная с адреса ds:0009H.

Вариант 8

Вычислить выражение X + Y - Z, где X = 6453; Y =2369; Z =38; X,Y,Z - десятичные числа, Результат разместить в ячейках памяти, начиная с адреса ds:0008H.

Вариант 9

Найти разность двух шестнадцатиричных чисел 732112H и 354634H. Результат разместить в ячейках памяти, начиная с адреса ds:0007H.

Вариант 10

Найти разность двух десятичных чисел 576623 и 8769. Результат разместить в ячейках памяти, начиная с адреса ds:0003H

Вариант 11

Найти разность двух шестнадцатиричных чисел D4568BH и 4857E4H. Результат разместить в ячейках памяти, начиная с адреса ds:0015H.

Вариант 12

Найти сумму двух шестнадцатиричных чисел F546D4H и 72D781H. Результат разместить в ячейках памяти, начиная с адреса ds:0010H.

Вариант 13

Найти сумму двух десятичных чисел 549092 и 958261. Результат разместить в ячейках памяти, начиная с адреса ds:0013H.

Вариант 14

Найти разность двух десятичных чисел 651521 и 18145. Результат разместить в ячейках памяти, начиная с адреса ds:0023H

Вариант 15

Вычислить выражение X + Y - Z, где X = 86453; Y = 12369; Z =238; X,Y,Z - десятичные числа. Результат разместить в ячейках памяти, начиная с адреса ds:0008H.

Задание на практическое занятие № 2 (Часть 2)

Составить алгоритм и программу вычисления выражения. Результат поместить в память (M)

Вариант 1

(86h*55h+2235h)/(67h*95h), M(ds:0007h)

Вариант 2

(35h*2h+28h*46h)/39h, M(ds:00013h)

Вариант 3

(99h*33h-1211h)/4325h, M(ds:0026h)

Вариант 4

8733h*25/(22h*34h+87h), M(ds:0016h)

Вариант 5

(23h*75h+25h*33h)/(78h-59h) , M(ds:0012h)

Вариант 6

(2415h*18h - 1648h)/(15h*21h), M(ds:00018h)

Вариант 7

(44h*21h+1783h)/(12h*37h), M(ds:0020h)

Вариант 8

(91h*72h-1368h)/(13h*25h), M(ds:0017h)

Вариант 9

(9124h-18h*28h)/(13h*11h), M(ds:0024h)

Вариант 10

(34h*15h-96h)/(4237h-3976h), M(ds:0032h)

Вариант 11

(86h+ 2Eh*bh)/(28eh-283h), M(ds:0058h)

Вариант 12

(91h*1eh-8bh*0fh)/13h, M(ds:0072h)

Вариант 13

(2876h*29h-256h)/(10h*1eh), M(ds:0007h)

Вариант 14

(456h*7bh+93cdh)/7d6h, M(ds:0080h)

Вариант 15

(786h*845h-29bh)/(876h-6deh), M(ds:0869h)

Практическое занятие № 3
Изучение группы логических команд

Цель работы: "Изучить особенности выполнения логических команд и программные способы маскирования данных, разработать алгоритм и программы с использованием этих команд".

Краткие теоретические сведения:

Мнемокод	Операнд	Комментарий
AND	Op1,Op2	Логическое умножение содержимого Op1 и Op2, результат помещается в Op1. $(Op1) \wedge (Op2) \rightarrow (Op1)$
OR	Op1,Op2	Логическое сложение, содержимого Op1 и Op2, результат помещается в Op1. $(Op1) \vee (Op2) \rightarrow (Op1)$
XOR	Op1,Op2	Производится отрицание равнозначности (логическое исключающее ИЛИ) содержимого Op1 и Op2, результат помещается в Op1. $(Op1) \oplus (Op2) \rightarrow (Op1)$
NOT	Op	Логическое отрицание.

AND Op1,Op2 – логическое умножение, которое очищает разряд числа, если в соответствующем разряде маски будет записан 0, и не изменяет его, если в разряде маски записано число 1:

$\wedge$ 10100101
10000111
10000101

OR Op1,Op2 – логическое сложение, которое устанавливает разряд числа 1, если в таком же разряде маски будет записана 1, и не изменяет его, если в этом же разряде маски записан 0:

$\vee$ 10100101
10000111
10100111

XOR Op1,Op2 – логическое исключающее ИЛИ, которое инвертирует содержание разряда числа, если в соответствующем разряде маски будет записана 1, и не изменяет его, если в этом же разряде маски записан 0:

$\oplus$ 10100101
10000111
00100010

Задание на практическое занятие № 3

Дано состояние 24 двоичных позиционных датчиков, сигнализирующих состояние объекта. Все датчики делятся на три группы: X,Y, Z.

При анализе состояния объекта, первоочередная информация находится в разрядах RX числа X, RY числа Y и RZ числа Z из выделенных разрядов составить число N, которое разместить в ячейку памяти с адресом ADR

№ вар.	X	Y	Z	RX	RY	RZ	ADR
1	37H	CDH	A0H	2,4	0,5,6,7	1,3	ds:2100H
2	BFH	43H	04H	1,5,7	2,3,4	0,6	ds:2102H
3	56H	E2H	2AH	0,6,7	1,4,5	2,3	ds:2200H
4	A4H	8FH	72H	1,2,4	6,7	0,3,5	ds:2202H

5	26H	DAH	1AH	0,1,5	4,2	3,6,7	ds:2101H
6	13H	AEH	35H	1,2,6	3,7	4,5	ds:2202H
7	F5H	78H	BDH	0,1,7	2,3,6	4,5	ds:3000H
8	67H	4DH	E2H	3,4,6	0,2	1,5,7	ds:3001H
9	C8H	29H	F4H	5,6	0,3,7	1,2,4	ds:3002H
10	D6H	5CH	31H	3,5,7	2,6	0,1,4	ds:3100H
11	E2H	35H	AEH	1,6	3,4	0,2	ds:2100H
12	5AH	3FH	D1H	2,3,7	0,6	1,4	ds:2200H
13	47H	61H	B4H	0,3	6,7	1,2,4	ds:2300H
14	9AH	53H	8FH	0,1,4	3,6	2,5,7	ds:3000H
15	F7H	05H	E7H	3,6,7	0,4	1,2,5	ds:3100H

Практическое занятие № 4
Изучение групп команд безусловных и условных переходов и команд управления циклом

Цель работы: "Изучить особенности выполнения команд передачи управления и управления циклом, области их применения. Разработка алгоритма, составление и отладка программы с использованием этих команд".

Краткие теоретические сведения:

При выполнении программы команды МП выполняются последовательно одна за другой. Однако во многих случаях необходимо изменять порядок следования команд, например при выполнении циклических вычислений, анализе различных признаков, влияющих на процесс выполнения программы. Такой переход может быть осуществлен безусловно или только при выполнении определенных условий. В качестве условий выступают значения признаков в регистре признаков МП. Анализируя значения признаков, команды не изменяют их.

Мнемокод	Операнд	Комментарий
JMP	метка	Производится безусловный переход на метку
JC	метка	Производится условный переход на метку, если C=1.
JNC	метка	Производится условный переход на метку, если C=0.
JZ	метка	Производится условный переход на метку, если Z=1.
JNZ	метка	Производится условный переход на метку, если Z=0.
JS	метка	Производится условный переход на метку, если S=1.
JNS	метка	Производится условный переход на метку, если S=0
JP	метка	Производится условный переход на метку, если P=1,
JNP	метка	Производится условный переход на метку, если P=0
JO	метка	Производится условный переход на метку, если O=1
JNO	метка	Производится условный переход на метку, если O=0
JCXZ	метка	Производится условный переход на метку, если CX=0
LOOP	метка	Производится •декремент регистра cx; •сравнения регистра cx с нулем: •если (cx) > 0, то управление передается на метку •если (cx) = 0, то управление передается на следующую после loop команду.
LOOPE (LOOPZ)	метка	Производится •декремент регистра cx; •сравнения регистра cx с нулем; •анализа состояния флага нуля zf: •если (cx) > 0 и zf = 1, управление передается на метку ; •если (cx) = 0 или zf = 0, управление передается на следующую после loop команду.
LOOPNE (LOOPNZ)	метка	Производится •декремента регистра cx; •сравнения регистра cx с нулем; •анализа состояния флага нуля zf: •если (cx) > 0 и zf = 0, управление передается на метку; •если (cx)=0 или zf=1, управление передается на следующую после loop команду.

Команды loope/loopz и loopne/loopnz по принципу своей работы являются взаимообратными. Они расширяют действие команды loop тем, что дополнительно анализируют флаг zf, что дает возможность организовать досрочный выход из цикла, используя этот флаг в качестве индикатора.

Задание на практическое занятие № 4

Вариант 1

Составить алгоритм и программу заполнения последовательностью констант, начиная с 1H, области памяти длиной FH байт с начальным адресом ds:0000H, если число в аккумуляторе - нечетное. В противном случае эту область заполнить константой F5H.

Вариант 2

Занести в область памяти ds:0008H - ds:0027H последовательность констант 0H - 19H и найти среди них третье по счету четное число, записать это число в регистр CL.

Вариант 3

Составить алгоритм и программу заполнения области памяти ds:0008H - ds:0057H последовательность чисел от 0H до 49H, если число в аккумуляторе не отрицательное. Вывести в ячейку памяти ds:0058H число, находящееся в аккумуляторе.

Вариант 4

Составить алгоритм и программу заполнения области памяти ds:0018H - ds:001FH константой F3H если число в аккумуляторе - четное, в противном случае константой 11H. Вывести в ячейку памяти ds:0020H число, находящееся в аккумуляторе.

Вариант 5

Составить алгоритм и программу заполнения области памяти ds:0020H - ds:0027H константой E7H если число в аккумуляторе отрицательное, в противном случае константой CEH. Вывести в ячейку памяти ds:0028H число, находящееся в аккумуляторе.

Вариант 6

Составить алгоритм и программу заполнения области памяти длиной 17H байт с начальным адресом ds:0008H константой D1H если в аккумуляторе ноль, в противном случае в эту область памяти внести последовательность чисел начиная с 1H.

Вариант 7

Занести в область памяти ds:0000H - ds:0017H последовательность констант от 70H до 87H и найти среди них второе по счету отрицательное число, записать это число в регистр BL.

Вариант 8

Составить алгоритм и программу заполнения области памяти длиной 25H байт с начальным адресом ds:0000H константу E5H если значение аккумулятора нулевое, в противном случае в эту область памяти внести последовательность чисел начиная с 1H. Вывести в ячейку памяти ds:0026H число, находящееся в аккумуляторе.

Вариант 9

Составить алгоритм и программу заполнения области памяти длиной 20H байт с начальным адресом ds:0000H последовательность констант от 0H до 20H а в область памяти ds:0020H - ds:0040H последовательность констант от 20H до 0H.

Вариант 10

Составить алгоритм и программу занесения в область памяти ds:0008H - ds:0027H последовательность констант от 70H до 8FH и найти среди них четвертое по счету отрицательное число, записать это число в ячейку памяти ds:0018H.

Вариант 11

Составить алгоритм и программу заполнения последовательностью констант 16H, 15H, 14, … области памяти длиной 8H байт с начальным адресом ds:0008H, если число в аккумуляторе - четное. В противном случае эту область заполнить константой EEH.

Вариант 12

Занести в область памяти ds:0018H - ds:0027H последовательность констант 20H - 11H и найти среди них второе по счету четное число, записать это число в регистр BH.

Вариант 13

Составить алгоритм и программу заполнения области памяти ds:0008H – ds:000FH последовательностью чисел от 0H до 7H, если число в аккумуляторе отрицательное в противном случае в эту область заполнить последовательностью чисел от 7H до 0H. Вывести в ячейку памяти ds:0058H число, находящееся в аккумуляторе.

Вариант 14

Составить алгоритм и программу заполнения области памяти ds:0018H - ds:001FH константой F3H если число в аккумуляторе - четное, в противном случае константой 11H. Вывести в ячейку памяти ds:0020H число, находящееся в аккумуляторе.

Вариант 15

Составить алгоритм и программу заполнения области памяти ds:0020H – ds:003FH константой BBH если число в аккумуляторе не отрицательное, в противном случае константой ABH. Вывести в ячейку памяти ds:0028H число, находящееся в аккумуляторе.

Практическое занятие № 5
Изучение команд работы с подпрограммами

Цель работы: "Изучить особенности выполнения команд вызова подпрограмм и возвращения из подпрограмм. Разработка алгоритма, составление и отладка программы с использованием этих команд".

Краткие теоретические сведения:

При разработке сложных программ очень часто приходится выполнять однотипные действия, но с разными значениями. Например, обрабатывать показания датчиков, преобразовывать числа из двоичной системы исчисления в десятичную и т.д. Наиболее просто это сделать с использованием подпрограмм.

Мнемокод	Операнд	Комментарий
CALL	ADR (метка)	Производится безусловный переход к подпрограмме по адресу ADR или на метку ADR → (IP), (IP) → (SP)
RET		Производится безусловный возврат из подпрограммы (SP) → (IP)

Задание на практическое занятие № 5

Вариант 1

Заполнить две области памяти ds:2100H - ds:2150H и ds:2200H - ds:2250H последовательностью констант от 0H до 50H, если число, находящееся в регистре BL четное, в противном случае эти области заполнить последовательностью констант от 50H до 0H.

Вариант 2

Если $7^{ой}$ бит аккумулятора равен 1, то заполнить константой 45H две области памяти длиной 65 байт начиная с адресов ds:2100H и ds:2200H, если $7^{ой}$ бит равен 0, то эти две области заполнить константой 55H.

Вариант 3

Если в аккумуляторе находится число 58H, заполнить константой B1H две области памяти длиной 75 байт, начиная с адресов ds:2100H и ds:2200H, в противном случае эти две области заполнить константой B2H.

Вариант 4

Заполнить две области памяти длиной 45 байт начиная с адресов ds:2100H, ds:2200H последовательностью констант начиная с 1H, если число находящееся в ячейке памяти ds:2250H отрицательное, в противном случае эти области заполнить константой A5H.

Вариант 5

Если нулевой бит аккумулятора равен 1, то заполнить константой F1H две области памяти длиной 120 байт начиная с адресов ds:2100H и ds:2200H, если нулевой бит равен 0, то эти две области заполнить константой F2H.

Вариант 6

Если число в аккумуляторе четное, заполнить константой 30H две области памяти длиной 110 байт, начиная с адресов ds:2200H и ds:2300H, в противном случае эти две области заполнить константой 90H.

Вариант 7

Если второй бит аккумулятора равен 1, то заполнить константой 55H две области памяти длиной 65 байт начиная с адресов ds:2200H и ds:2300H, если второй бит равен 0, то эти две области заполнить константой 33H.

Вариант 8

Если число в аккумуляторе четное, заполнить константой C9H две области памяти длиной 72 байт, начиная с адресов ds:2100H и ds:2200H, в противном случае эти две области заполнить последовательностью констант, начиная с 1H.

Вариант 9

Если в аккумуляторе находится число 20H, заполнить константой E5H две области памяти длиной 80 байт, начиная с адресов ds:2100H и ds:2200H, в противном случае эти две области заполнить константой D6H.

Вариант 10

Заполнить две области памяти длиной 85 байт, начиная с адресов ds:2100H, ds:2200H последовательностью констант начиная с 0H, если 5ый и 2ой бит аккумулятора равны 0, в противном случае заполнить эти области константой 50H.

Вариант 11

Если число в аккумуляторе отрицательное заполнить две области памяти длиной 15 байт начиная с адресов ds:3000H, ds:3010H последовательностью констант 01H - 0FH , в противном случае заполнить эти области памяти заполнить последовательностью констант 0FH - 01H. Вывести число, находящееся в аккумуляторе в ячейку памяти ds:3020H.

Вариант 12

Если число в регистре AL равно числу записанному в регистре BL заполнить две области памяти длиной 120 байт начиная с адресов ds:3000H, ds:3100H константой AAH, если число в регистре AL меньше числа в регистре B заполнить эти две области памяти константой ABH , если число в регистре AL больше числа в регистре B заполнить эти две области памяти константой BAH. Вывести числа находящееся в регистре AL и в регистре BL в ячейки памяти ds:3200H и ds:3201H.

Вариант 13

Если число в аккумуляторе 40H заполнить две области памяти длиной 20 байт, начиная с адресов ds:3000H, ds:3020H последовательностью констант начиная с 1H, в противном случае заполнить эти области памяти заполнить константой A5H. Вывести число находящееся в аккумуляторе в ячейку памяти ds:3040H.

Вариант 14

Если 3ий и 6ой бит аккумулятора равны 0, заполнить две области памяти длиной 20 байт, начиная с адресов ds:3000H, ds:3020H последовательностью констант начиная с 1H, в противном случае заполнить эти области памяти заполнить константой 40H. Вывести число, находящееся в аккумуляторе в ячейку памяти ds:3040H.

Вариант 15

Заполнить две области памяти ds:0008H – ds:0017H и ds:0020H – ds:002FH последовательностью констант от 0H до FH, если число, находящееся в регистре CL четное, в противном случае эти области заполнить последовательностью констант от FH до 0H.

Практическое занятие № 6
Изучение групп команд сравнения и логических сдвигов

Цель работы: Изучить особенности выполнения команд сравнения, сдвига и области их применения. Разработать алгоритм, составить и отладить программы с использованием этих команд.

Краткие теоретические сведения:

Мнемокод	Операнд	Комментарий
CMP	Оп1,Оп2	Сравнение двух операндов. Оп1–Оп2. Устанавливаются флаги. Оп1и Оп2 не изменяются.
SHL	Оп,Сч_сдв	Логический сдвиг влево. Содержимое операнда сдвигается влево на количество битов, определяемое значением Сч_сдв. Справа (в позицию младшего бита) вписываются нули;
SHR	Оп,Сч_сдв	Логический сдвиг вправо. Содержимое операнда сдвигается вправо на количество битов, определяемое значением Сч_сдв. Слева (в позицию старшего, знакового бита) вписываются нули.
ROL	Оп,Сч_сдв	Циклический сдвиг влево. Содержимое операнда сдвигается влево на количество бит, определяемое операндом Сч_сдв. Сдвигаемые влево биты записываются в тот же операнд справа.
ROR	Оп,Сч_сдв	циклический сдвиг вправо. Содержимое операнда сдвигается вправо на количество бит, определяемое операндом Сч_сдв. Сдвигаемые вправо биты записываются в тот же операнд слева.
RCL	Оп,Сч_сдв	Циклический сдвиг влево через перенос. Содержимое операнда сдвигается влево на количество бит, определяемое операндом Сч_сдв. Сдвигаемые биты поочередно становятся значением флага переноса cf.
RCR	Оп,Сч_сдв	Циклический сдвиг вправо через перенос. Содержимое операнда сдвигается вправо на количество бит, определяемое операндом Сч_сдв. Сдвигаемые биты поочередно становятся значением флага переноса cf.

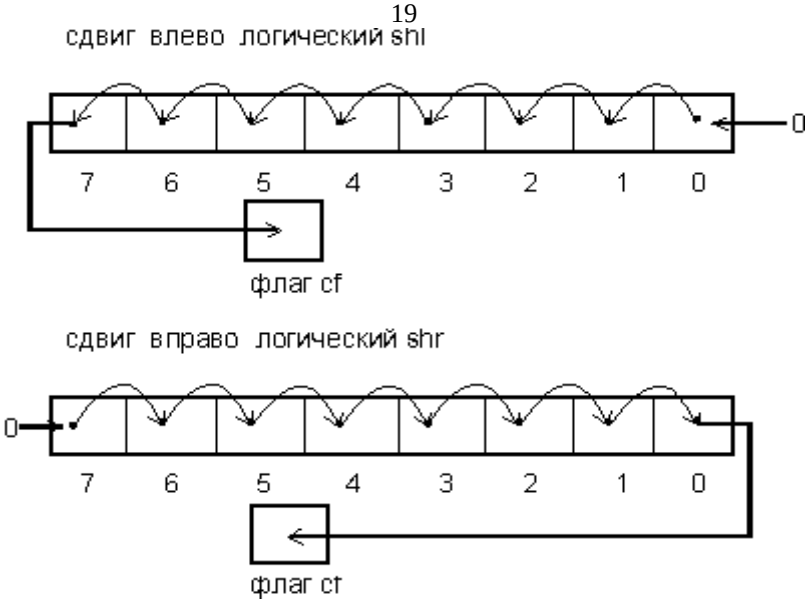
Команды линейного сдвига

К командам этого типа относятся команды, осуществляющие сдвиг по следующему алгоритму:

- очередной “выдвигаемый” бит устанавливает флаг cf;
- бит, вводимый в операнд с другого конца, имеет значение 0;
- при сдвиге очередного бита он переходит во флаг cf, при этом значение предыдущего сдвинутого бита теряется!

Команды линейного сдвига делятся на два подтипа:

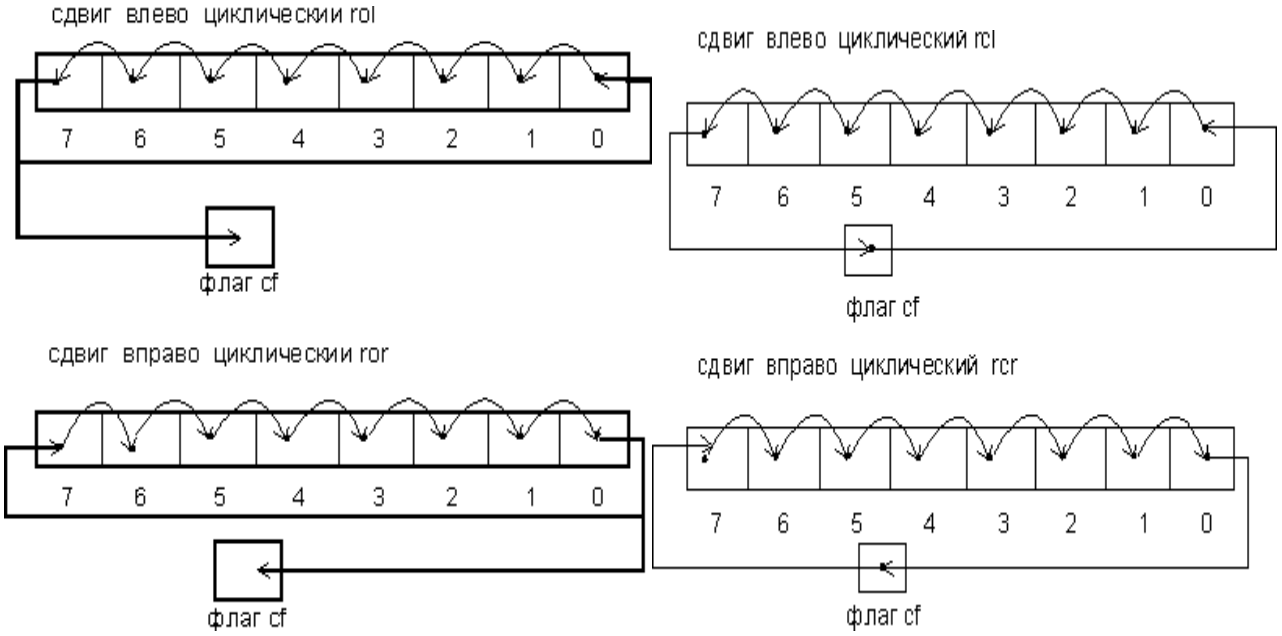
- команды логического линейного сдвига;
- команды арифметического линейного сдвига.



Команды циклического сдвига

К командам этого типа относятся команды, сохраняющие значения сдвигаемых бит. Есть два типа команд циклического сдвига:

- команды простого циклического сдвига;
- команды циклического сдвига через флаг переноса cf.



Задание на практическое занятие № 6

Вариант 1

Сравнить содержимое регистра AL с содержимым регистра DL, если содержимое AL равно содержимому регистра DL поменять полубайты числа в регистре DL, если содержимое AL меньше содержимого регистра DL вывести в ячейку памяти 0012H число 12H, если содержимое AL больше содержимого регистра DL вывести в ячейку памяти 0012H число 21H.

Вариант 2

Сравнить содержимое AL с содержимым ячейки памяти 0015H, если содержимое аккумулятора равно содержимому ячейки памяти 0015H произвести инвертирование AL и вывести полученное значение в ячейку памяти 0017H, в противном случае сдвинуть содержимое AL на 3 разряда влево и поместить в старший разряд 0, результат запомнить в ячейке памяти 0020H.

Вариант 3

Написать программу подсчета числа единиц в байте находящемся в AL в регистре CX находится счетчик цикла, результат (числа единиц в байте) поместить в регистр BL. Если число единиц в байте находящемся в AL меньше 2 поменять местами полубайты числа находящегося в регистре BL и результат записать в ячейку памяти 0015H.

Вариант 4

Сравнить содержимое AL и регистра BH, если содержимое AL равно содержимому регистра BH произвести обмен блоков памяти расположенных начиная с адресов 1200H и 1210H и содержащих по 6 байт, в противном случае сдвинуть содержимое AL на два разряда вправо.

Вариант 5

Сделать побайтное сравнение массивов данных содержащих по 10H байт и расположенных начиная с адресов 0100H и 0130H соответственно. Если в указанном массиве число кодов неравенств меньше 7 сдвинуть первое неравное число на 4 разрада влево, в противном случае на 5 разрядов. Число кодов неравенств вывести в ячейку памяти 0020H, результат сдвига первого неравного числа вывести в ячейку памяти 0021H.

Вариант 6

Сравнить содержимое AL с содержимым ячейки памяти 0022H, если содержимое AL меньше содержимого ячейки памяти 0022H сдвинуть содержимое ячейки памяти 0022H на переменное число битов, которое определяется содержимым регистра DL, вывести полученное число в ячейку памяти 0023H.

Вариант 7

Сравнить массив данных длиной 8 байт, расположенный начиная с адреса 0020H с числом F1H. Числа массива данных которые равны F1H подсчитать и их число и записать его по адресу 0035H, числа массива данных которые меньше F1H записать в область памяти, начиная с адреса 0040H, количество этих чисел записать в ячейку памяти 0036H.

Вариант 8

Сравнить массивы данных содержащих по 5 байт и расположенных начиная с адресов 0100H и 0110H соответственно. Коды неравенств поместить начиная с адреса 0120H, число кодов неравенств по адресу 0130H. Если в указанном массиве число кодов неравенств больше 3 сдвинуть первое неравное число на 2 разряда вправо, в противном случае влево на столько же разрядов. Вывести полученное число в ячейку памяти 0131H.

Вариант 9

Сравнить массивы данных содержащих по 10H байт и расположенных начиная с адресов 0300H и 0310H соответственно. Если число кодов неравенств больше 5, произвести обмен блоков памяти. В противном случае умножить константы второго блока памяти на 2 используя команды сдвигов.

Вариант 10

Сравнить содержимое AL с массивом данных длиной 6 байт расположенного начиная с адреса 0150H. Если содержимое AL больше содержимого каждой ячейки памяти данного массива, записать в ячейку памяти 0160H инвертированное значение AL, в противном случае сдвинуть содержимое AL на 4 разряда с помощью команд циклического сдвига и результат записать в ячейку памяти 0361H.

Вариант 11

Сравнить массивы данных содержащих по 7 байт и расположенных начиная с адресов 0200H и 0210H соответственно. Коды неравенств поместить начиная с адреса 0020H, число кодов неравенств по адресу 0231H. Если в указанном массиве число кодов неравенств больше 4 сдвинуть первое неравное число на 3 разряда вправо, в противном случае влево на столько же разрядов. Вывести полученное число в ячейку памяти 0232H.

Практическое занятие № 7
Изучение группы цепочных команд

Цель работы: "Изучить особенности программирования с помощью команд обработки строк символов".

Краткие теоретические сведения:

Мнемокод	Операнд	Комментарий
movs	адр.пр.,адр.ист.	Команда копирует байт, слово или двойное слово из цепочки, адресуемой операндом адрес_ист, в цепочку, адресуемую операндом адрес_пр.
movsb	адр.пр.,адр.ист.	переслать цепочку байт;
movsw	адр.пр.,адр.ист.	переслать цепочку слов
movsd	адр.пр.,адр.ист.	переслать цепочку двойных слов
cld		очистить флаг направления. Команда сбрасывает флаг направления df в 0.
std		установить флаг направления. Команда устанавливает флаг направления df в 1.
cmps	адр.пр.,адр.ист.	Сравнение цепочек
cmpsb	адр.пр.,адр.ист.	Сравнить строку байт
cmpsw	адр.пр.,адр.ист.	Сравнить строку слов
cmpsd	адр.пр.,адр.ист.	Сравнить строку двойных слов
scas	адр_приемн.	Сканирование цепочек байт
scasb	адр_приемн.	Сканирование цепочек байт
scasw	адр_приемн.	Сканирование цепочек слов
scasd	адр_приемн.	Сканирование цепочек двойных слов
lods	адр_ист.	загрузка элемента из цепочки
lodsb	адр_ист.	загрузить байт из цепочки в AL
lodsw	адр_ист.	загрузить слово из цепочки в AX
lodsd	адр_ист.	загрузить слово из цепочки в EAX
stosb	адр_приемн.	Сохранение из регистра AL в цепочке
stosw	адр_приемн.	сохранение из регистра AX в цепочке
stosd	адр_приемн.	сохранение из регистра EAX в цепочке
rep		Префикс повторения используется с командами movs и stos. Префикс rep заставляет данные команды выполняться, пока содержимое в ecx/cx не станет равным 0. При этом цепочечная команда, перед которой стоит префикс, автоматически уменьшает содержимое ecx/cx на единицу. Та же команда, но без префикса, этого не делает
repe или **repz**		Цепочная команда выполняется до тех пор, пока содержимое ecx/cx не равно нулю или флаг zf равен 1, иначе управление передается следующей команде программы. Наиболее эффективно эти префиксы можно использовать с командами cmps и scas для поиска отличающихся элементов цепочек.
Repne или **repnz**		Цепочная команда выполняется до тех пор, пока содержимое ecx/cx не равно нулю или флаг zf равен 0, иначе управление передается следующей команде программы. Данные префиксы можно использовать с командами cmps и scas для поиска совпадающих элементов цепочек.

адрес_источника — пара ds:esi/si;
адрес_приемника — пара es:edi/di.

Вариант 1

Сравнить две строки длиной 15 байт и поместить первый не совпавший байт в памяти в яч. памяти 0020H, второй в следующую и т.д. Если первый байт > второго поместить его в ячейку памяти 0041H. Вывести все данные и результаты на экран.

Вариант 2

Сравнить две строки длиной 20H байт и поместить первый не совпавший байт в памяти поместить в яч. памяти 0150H, второй в следующую и т.д. Если первый байт < второго поместить его в ячейку памяти 0149H. Вывести все данные и результаты на экран.

Вариант 3

Сравнить две строки длиной 10 байт и поместить первый совпавший байт в AL. Если первый байт четный вывести его на экран. В противном случае поменять содержимое строк местами. Вывести данные и результаты на экран.

Вариант 4

Сравнить две строки длиной 13 байт и поместить не совпавшие байты местами, если первый не совпавшие байт = 45H. Вывести все данные и результаты на экран.

Вариант 5

Сравнить две строки длиной 13 байт и поместить не совпавшие байты поместить в строку 3, если первый не совпавший байт нечетный. В случае полного совпадения переслать строку 2 в строку 3 Вывести все данные и результаты на экран.

Вариант 6

Сравнить две строки длиной 8 байт и поместить второй не совпавший байт в памяти в яч. памяти в AL. Если этот байт нечетный поменять строку 1 и строку 2 местами. Вывести все данные и результаты на экран.

Вариант 7

Сравнить две строки длиной 6 байт и поместить первый не совпавший байт в памяти поместить в яч. памяти 0150H, второй в следующую и т.д. Если первый байт > второго поместить его в ячейку памяти 0149H. Вывести все данные и результаты на экран.

Вариант 8

Сравнить две строки длиной 14 байт и поместить первый совпавший байт в AL. Если первый совпавший байт четный вывести его на экран. В противном случае поменять содержимое строк местами. Вывести данные и результаты на экран.

Вариант 9

Сравнить две строки длиной 11 байт и поместить не совпавшие байты местами, если первый не совпавшие байт четный. Вывести все данные и результаты на экран.

Вариант 10

Сравнить две строки длиной 10 байт и поместить не совпавшие байты в строку 3, если первый не совпавший байт нечетный. В случае полного совпадения обменять первые половины строк 1 и 2. Вывести все данные и результаты на экран.

Вариант 11

Сравнить две строки длиной 12 байт и поместить совпавшие байты в строку 3, если первый совпавший байт четный. В случае полного совпадения обменять вторые половины строк 1 и 2. Вывести все данные и результаты на экран.