



Уральский  
федеральный  
университет

имени первого Президента  
России Б.Н.Ельцина

Институт радиоэлектроники  
и информационных  
технологий

**А. А. ШЕГАЛ**

# ПРИМЕНЕНИЕ ПРОГРАММНОГО КОМПЛЕКСА MULTISIM ДЛЯ ПРОЕКТИРОВАНИЯ УСТРОЙСТВ НА МИКРОКОНТРОЛЛЕРАХ

Лабораторный практикум

Министерство образования и науки Российской Федерации  
Уральский федеральный университет  
имени первого Президента России Б. Н. Ельцина

**ПРИМЕНЕНИЕ ПРОГРАММНОГО КОМПЛЕКСА  
MULTISIM ДЛЯ ПРОЕКТИРОВАНИЯ УСТРОЙСТВ  
НА МИКРОКОНТРОЛЛЕРАХ**

*Рекомендовано методическим советом УрФУ  
в качестве лабораторного практикума  
студентов, обучающихся по направлению 211000 – Конструирование  
и технология электронных средств*

Екатеринбург  
Издательство Уральского университета  
2014

УДК 004.42:004.415.2(075.8)  
ББК 32.973-018.2я73+32.973.26-02я73  
ШЗ8

Рецензенты:

кафедра сетевых информационных систем и компьютерных технологий обучения Российского государственного профессионально-педагогического университета (канд. пед. наук *Е. В. Чубаркова*);

д-р техн. наук, проф. *Г. В. Чирков*, главный специалист отдела энергосвязи ООО «Прософт-системы»

Научный редактор – канд. техн. наук, доц. *В. И. Иевлев*

**Шегал, А. А.**

ШЗ8 Применение программного комплекса Multisim для проектирования устройств на микроконтроллерах : лабораторный практикум / А. А. Шегал. – Екатеринбург : Изд-во Урал. ун-та, 2014. –114, [2] с.

ISBN 978-5-7996-1117-0

Изложены теория и практическое решение основных вопросов, связанных с проектированием устройств на микроконтроллерах семейства mcs-51/52 с помощью программного комплекса Multisim-10.

Предназначается для студентов обучающихся по направлению подготовки «Проектирование и технология электронно-вычислительных средств» при выполнении цикла лабораторных работ по дисциплине «Центральные и периферийные устройства ЭВС» и курсовом проектировании.

Библиогр.: 10 назв. Рис. 76. Табл. 20. Прил. 1.

УДК 004.42:004.415.2(075.8)  
ББК 32.973-018.2я73+32.973.26-02я73

ISBN 978-5-7996-1117-0

© Уральский федеральный университет, 2014

## Оглавление

<b>ПЕРЕЧЕНЬ ИСПОЛЬЗУЕМЫХ СОКРАЩЕНИЙ</b> .....	5
<b>ВВЕДЕНИЕ</b> .....	6
<b>1. МЕТОД СТРУКТУРНОГО ПРОЕКТИРОВАНИЯ МИКРОКОНТРОЛЛЕРНЫХ УСТРОЙСТВ</b> .....	8
1.1. Языки программирования микроконтроллеров .....	9
1.2. Структурное программирование МК .....	10
<b>2. ОПИСАНИЕ ЛАБОРАТОРНЫХ РАБОТ</b> .....	14
2.1. ЛАБОРАТОРНАЯ РАБОТА № 1. Основы работы с программным обеспечением Multisim .....	14
2.1.1. Интерфейс пользователя .....	14
2.1.2. Создание проекта и программного файла .....	30
2.1.3. Задания для лабораторной работы .....	33
2.1.4. Содержание отчета.....	35
2.1.5. Вопросы для самоконтроля.....	35
2.2. ЛАБОРАТОРНАЯ РАБОТА № 2. ПОДКЛЮЧЕНИЕ ВНЕШНЕЙ ПАМЯТИ И ЕЕ ТЕСТИРОВАНИЕ .....	37
2.2.1. Особенности подключения к МК внешней памяти и периферийных устройств.....	37
2.2.2. Порядок выполнения лабораторной работы .....	39
2.2.3. Задания для лабораторной работы .....	48
2.2.4. Содержание отчета.....	49
2.2.5. Вопросы для самоконтроля.....	49
2.3. ЛАБОРАТОРНАЯ РАБОТА № 3. ОРГАНИЗАЦИЯ ЗАДАНЫХ ИНТЕРВАЛОВ ВРЕМЕНИ .....	50
2.3.1. Основы настройки и использования таймеров МК-51 .....	50
2.3.2. Порядок выполнения лабораторной работы .....	53
2.3.3. Задания для лабораторной работы .....	59
2.3.4. Содержание отчета.....	60
2.3.5. Вопросы для самоконтроля.....	60
2.4. ЛАБОРАТОРНАЯ РАБОТА № 4. ОСНОВЫ ОРГАНИЗАЦИИ ПОСЛЕДОВАТЕЛЬНОГО ПОРТА .....	61
2.4.1. Основные сведения о режимах работы последовательного порта .....	61
2.4.2. Порядок выполнения лабораторной работы .....	64
2.4.3. Задания для лабораторной работы .....	67
2.4.4. Содержание отчета.....	68
2.4.5. Вопросы для самоконтроля.....	68
2.5. ЛАБОРАТОРНАЯ РАБОТА № 5. ОТОБРАЖЕНИЕ ИНФОРМАЦИИ В СИСТЕМАХ С МК-51 .....	69
2.5.1. Общие сведения о семисегментных индикаторах .....	69
2.5.2. Ход выполнения лабораторной работы .....	72

2.5.3. Задания для лабораторной работы .....	76
2.5.4. Содержание отчета.....	76
2.5.5. Вопросы для самоконтроля.....	77
<b>ЛАБОРАТОРНАЯ РАБОТА № 6. ИЗУЧЕНИЕ ПРИНЦИПОВ РАБОТЫ ЦИФРОАНАЛОГОВЫХ ПРЕОБРАЗОВАТЕЛЕЙ.....</b>	<b>78</b>
2.6.1. Общие сведения о цифроаналоговом преобразовании .....	78
2.6.2. Ход выполнения лабораторной работы .....	83
2.6.3. Задания для лабораторной работы .....	85
2.6.4. Содержание отчета.....	85
2.6.5. Вопросы для самоконтроля.....	85
<b>ЛАБОРАТОРНАЯ РАБОТА № 7. ИЗУЧЕНИЕ ПРИНЦИПОВ РАБОТЫ АНАЛОГО-ЦИФРОВЫХ ПРЕОБРАЗОВАТЕЛЕЙ.....</b>	<b>86</b>
2.7.1. Основные сведения об аналого-цифровых преобразователях .....	86
2.7.2. Ход выполнения лабораторной работы .....	93
2.7.3. Задания для лабораторной работы .....	96
2.7.4. Содержание отчета.....	97
2.7.5. Вопросы для самоконтроля.....	97
<b>ЛАБОРАТОРНАЯ РАБОТА № 8. ИССЛЕДОВАНИЕ ШИРОТНО- ИМПУЛЬСНОЙ МОДУЛЯЦИИ, РЕАЛИЗОВАННОЙ МИКРОКОНТРОЛЛЕРОМ МК-52.....</b>	<b>98</b>
2.8.1. Основы применения микроконтроллера МК-52 для получения ШИМ .....	98
2.8.2. Ход выполнения лабораторной работы .....	102
2.8.3. Задания для лабораторной работы .....	104
2.8.4. Содержание отчета.....	105
2.8.5. Вопросы для самоконтроля.....	105
<b>ЗАКЛЮЧЕНИЕ .....</b>	<b>106</b>
<b>БИБЛИОГРАФИЧЕСКИЙ СПИСОК .....</b>	<b>107</b>
<b>ПРИЛОЖЕНИЕ. НАСТРОЙКА ВИРТУАЛЬНЫХ ПРИБОРОВ: ГЕНЕРАТОРА СЛОВ, ЛОГИЧЕСКОГО ГЕНЕРАТОРА, ФУНКЦИОНАЛЬНОГО ГЕНЕРАТОРА .....</b>	<b>108</b>

## Перечень используемых сокращений

АЦП – аналого-цифровой преобразователь  
БД – база данных  
ЗМР – значение младшего разряда  
КМОП – комплементарная логика на транзисторах металл-оксид-полупроводник  
ЛКМ – левая кнопка мыши  
МК – микроконтроллеры  
МПУ – микропроцессорные устройства  
ОЗУ – оперативное запоминающее устройство  
ПЗУ – постоянное запоминающее устройство  
ПК – персональный компьютер  
ПКМ – правая кнопка мыши  
ПО – программное обеспечение  
ТТЛ – транзисторно-транзисторная логика  
ЦАП – цифро-аналоговый преобразователь  
ЭВС – электронно-вычислительные средства  
ШИМ – широтно-импульсная модуляция  
NI – National Instruments  
EWB – Electronics Workbench  
SFR – регистры специальных функций

## Введение

Современный уровень развития промышленности предъявляет повышенный спрос на специалистов в области проектирования и технологии устройств на микроконтроллерах (МК), компьютерных систем и средств связи. Очень важно, чтобы бакалавры и специалисты направления подготовки «Конструирование и технология электронных средств» могли использовать современные приемы компьютерного проектирования, которые повышают производительность и качество разработки с одновременным снижением ее стоимости.

Разработка сложного электронного устройства сопровождается физическим или математическим моделированием. Физическое моделирование часто связано с большими материальными затратами, поскольку требует изготовления макетов и их трудоемкого натурального исследования. В таких случаях, как правило, используют математическое моделирование при помощи средств и методов вычислительной техники.

Для того чтобы студенты получили практические навыки в сфере проектирования микроконтроллерных устройств, в учебный план дисциплины «Центральные и периферийные устройства ЭВС» введены лабораторно-практические занятия студентов с использованием элементов компьютерного моделирования ЭВС.

На кафедре имеется лицензионное программное обеспечение компании National Instruments (NI), поэтому подходящим продуктом является среда проектирования Multisim фирмы Electronics Workbench, которая в настоящее время входит в корпорацию NI. Она позволяет строить и анализировать любые электронные схемы: аналоговые, цифро-аналоговые и цифровые. Программа Multisim достаточно легко осваивается и удобна в работе. Важно, что последние версии ПО Multisim позволяют моделировать программируемые цифровые устройства на основе 8-разрядных МК с ядром MCS-51 и PIC-16 фирмы Microchip, поскольку в составе ПО имеются компиляторы с языка C и ассемблера указанных микроконтроллеров.

Предлагаемый цикл лабораторных работ предназначен для изучения принципов построения и функционирования электронно-вычислительных средств (ЭВС), построенных на базе микроконтроллеров семейства MCS-51 с использованием программного пакета моделирования Multisim.

Восьмиразрядные микроконтроллеры семейства MCS-51 выбраны для изучения по нескольким причинам:

- наличие программной модели микроконтроллера в базе элементов Multisim;
- простота понимания студентами 3 курса, которые имеют небольшой объем знаний по схемотехнике и программированию;
- возможность освоения основных приемов проектирования микроконтроллерных систем;
- наличие библиографических источников.

Для успешного выполнения студентами курсовой работы, которая завершает изучение дисциплины «Центральные и периферийные устройства ЭВС» в лабораторный практикум введены лабораторные занятия по решению часто встречающихся на практике задач, связанных с проектированием МК и разработкой типовых модулей микроконтроллерных устройств:

- получение требуемых интервалов времени;
- отображение информации на цифровых индикаторах;
- настройка и использование последовательного порта для приема / передачи информации;
- подключение внешней памяти к ядру МК (что является актуальным для 8-разрядных микроконтроллеров);
- цифро-аналоговое и аналого-цифровое преобразования информации, применяемые при построении систем обработки информации и управления;
- реализация широтно-импульсной модуляции, которая используется, в частности, при управлении исполнительными устройствами.

Помимо перечисленных выше тем лабораторных занятий в начале лабораторного практикума студенты знакомятся с основами работы в программной среде моделирования Multisim.

При выполнении лабораторных работ студенты должны практически освоить основы организации МК семейства MCS-51, которые рассматриваются в лекционном материале дисциплины [1], а также изложены, например, в [2].

Современным методом проектирования микроконтроллерных устройств является метод структурного проектирования, общие положения которого рассматриваются в следующем разделе.



## 1. Метод структурного проектирования микроконтроллерных устройств

Основная идея метода состоит в том, что при *использовании регламентированной последовательности действий по проектированию аппаратных средств и выборе определенной структуры программы* удастся разработать микроконтроллерную систему, отвечающую поставленным техническим требованиям.

Следует отметить, что метод структурного проектирования микроконтроллерных устройств хотя и не гарантирует обязательное успешное завершение проекта, но значительно увеличивает вероятность его успешного выполнения.

Рекомендуется использовать следующие этапы структурного проектирования [3]:

- 1) детальный анализ требований, предъявляемых к проектируемому устройству, на основе которых формулируются технические характеристики разрабатываемого устройства;
- 2) декомпозиция проектируемой системы на несколько иерархически взаимосвязанных функциональных подсистем (модулей) с определением взаимосвязей между ними. Для каждой подсистемы определяется выполняемая функция, входы и выходы, без конкретизации внутренней структуры (модель «черного ящика»);
- 3) разработка структурной схемы аппаратуры и алгоритма ее функционирования;
- 4) определение способа реализации каждого функционального модуля (выбор аппаратных и программных решений);
- 5) комплексная отладка проектируемого устройства.

В проектах разной сложности некоторые из перечисленных этапов могут быть объединены, например, 3 и 4.

В цикле лабораторных работ, выполняемом студентами, тема каждой лабораторной работы связана с проектированием типичного функционального модуля на МК. Совместная реализация определенной совокупности функциональных модулей, разработанных в лабораторном практикуме, обеспечит успешное проектирование микроконтроллерного устройства, отвечающего варианту задания курсовой работы.

## 1.1. Языки программирования микроконтроллеров

Разработчики микроконтроллерных систем пишут свои программы либо на ассемблере (машинно-ориентированном языке), либо на языках высокого уровня (ЯВУ). В настоящее время в организациях разработчиков встраиваемых устройств, как правило, применяется язык программирования C, который обеспечивает хороший доступ к аппаратным ресурсам МК.

Как любой другой язык высокого уровня, **язык C** позволяет работать такой исходный текст программы, который обладает свойством *переносимости* кода. Это означает, что программа, написанная на C для одного МК, затем может быть скомпилирована другим компилятором для МК с отличающимся процессорным ядром. Для того чтобы программа обладала свойством переносимости, синтаксис языка высокого уровня для разных компиляторов должен быть одинаков, таков, например, язык C стандарта ANSI. Основная цель стандартизации состоит в том, чтобы обеспечить разработчику возможность написания типовых функций управления (или функциональных модулей) один раз с последующим их многократным использованием в разных проектах и для разных МК.

Язык C также обеспечивает хорошую *читаемость* кода. Это свойство обозначает, что программист, не являющийся разработчиком программы, может по исходному тексту программы понять, какой алгоритм реализован и как программа работает. Еще одним преимуществом C как языка высокого уровня является наличие *библиотек математических функций* над числами, представленными в различных форматах, в том числе и в формате с плавающей точкой.

*Ассемблер*, по сравнению с языком C, имеет то важное преимущество, что *хорошо написанная на ассемблере программа исполняется за меньшее время и занимает в памяти меньший объем, чем программа на C*. Именно эти характеристики: время выполнения и размер программного кода, – являются критическими для приложений, где элементная база обладает относительно невысоким быстродействием, а память программ ограничена в объеме.

Практика совместного применения языков C и ассемблера в одном проекте показывает, что такой подход позволяет получить оптимальный результат как с точки зрения экономии времени на разработку, так и времени исполнения программы [3].

Основная часть прикладной программы, в которой производится преобразование данных, пишется на С, в то время, как критичные по времени реализации фрагменты алгоритма следует оформить на ассемблере. В некоторых ситуациях ассемблер также используют для программной поддержки внешних периферийных ИС.

В связи с вышесказанным в методических указаниях к лабораторным работам приводятся примеры реализации функциональных модулей как с использованием языка С, так и ассемблера МК.

Студенты могут выбрать язык программирования самостоятельно, однако следует отметить, что метод структурного проектирования МК, который позволяет увеличить скорость написания программ и облегчить их отладку за счет сокращения количества доступных программных конструкций, наилучшим образом отображается на программу, написанную на С.

## 1.2. Структурное программирование МК

Структурное программирование является составной частью структурного проектирования микроконтроллерного устройства. При разработке программ для МК и их отладке, как правило, используются два способа программирования: снизу вверх и сверху вниз. При написании программы **снизу вверх** приступить к ее отладке нельзя, не завершив полностью ее программирование. К тому же, каждая из невыявленных программных ошибок может привести к неработоспособности разрабатываемого устройства, может проявиться и противоположная ситуация: вследствие допущенной аппаратной ошибки не функционирует программное обеспечение.

При использовании технологии программирования **сверху вниз** на любом этапе она может быть оттранслирована и выполнена, при этом отслеживается выполнение всех фрагментов алгоритма, написанных к данному времени. Язык С наилучшим образом отвечает такой технологии программирования.

На начальном этапе разработки программа будет состоять из основной функции (main.c) – программы управления, в которой все вызываемые функции симулируются пустыми программными модулями (заглушками). По мере продвижения сверху вниз все большее число функций наполняется содержанием, и для них записывается исходный текст. Для хорошей читаемости программного кода желательно, чтобы название подпрограммы или функции отражало алгоритмическое действие. Пример такого подхода [4] показан в представленной ниже программе.

```

//Подпрограмма чтения порта
void Prochitat_Port (void)
{
//Пока это только подпрограмма заглушка!
}
//Подпрограмма включения индикатора
void Vkluchit_Indicanor (void)
{
//Пока это только подпрограмма заглушка!
}
//Основная программа
void main (void)
{
Prochitat_Port ();      //Прочитать порт
Vkluchit_Indicanor (); //Включить индикатор
}

```

Применение подпрограмм является не только средством структурирования программы, но и существенно сокращает размер машинного кода, если подпрограмма многократно используется, хотя при этом уменьшается быстродействие разрабатываемого устройства.

Основная идея структурного программирования заключается в том, что используются только четыре структурных оператора, которые позволяют получить программу любой сложности.

- 1. Линейная цепочка операторов.** Эта конструкция довольно часто используется, поскольку многие подзадачи могут быть разбиты на более простые последовательно выполняемые действия (см., например, структуру головной программы). Блок-схема конструкции линейной цепочки операторов представлена на рис. 1.1.

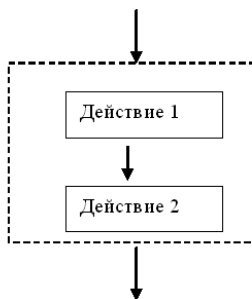


Рис. 1.1. Блок-схема конструкции управления «линейная цепочка операторов»

**2. Условное выполнение операторов.** Эта конструкция используется, когда текущее действие зависит от результата выполнения предыдущей части программы (рис. 1.2).

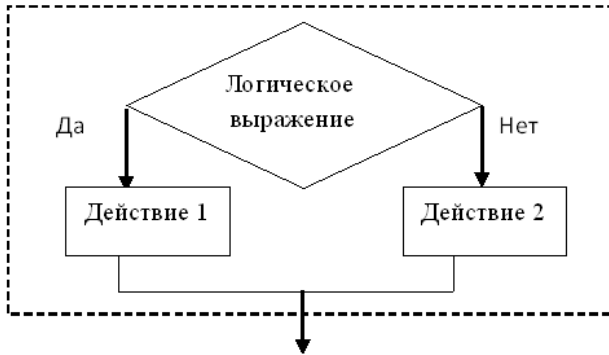


Рис. 1.2. Блок-схема конструкции управления «условное выполнение операторов»

**3. Циклическое выполнение оператора с проверкой условия после тела цикла.** Все операторы, которые должны повторяться в процессе выполнения этой конструкции, называются телом цикла. При выполнении этих операторов обычно модифицируется некоторая переменная (параметр цикла), которая влияет на завершение цикла. Такой оператор реализует конструкция языка С “do... while”. Блок-схема конструкции представлена на рис. 1.3.

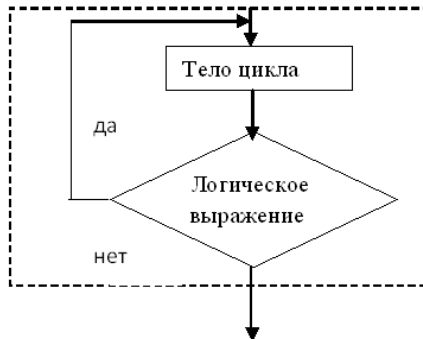


Рис. 1.3. Блок-схема конструкции управления «циклическое выполнение оператора с проверкой условия после тела цикла»

**4. Циклическое выполнение оператора с проверкой условия до тела цикла.** В отличие от предыдущей конструкции тело цикла может ни разу не выполняться (рис. 1.4).

В процессе написания программ обычно накапливаются подпрограммы и фрагменты кода, которые можно использовать в других программах. Можно копировать такие фрагменты из программы в программу, однако это приводит к неудобствам: программный код увеличивается, возникают сложности при написании и редактировании программы.

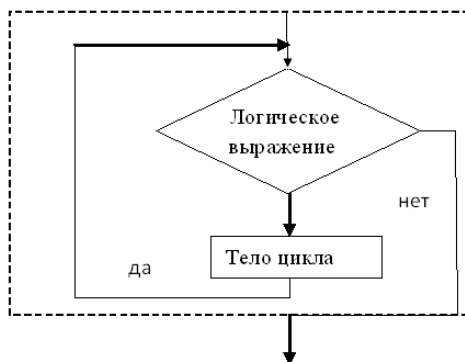


Рис. 1.4. Блок-схема конструкции управления «циклическое выполнение оператора с проверкой условия до тела цикла»

Кроме того, при изменении типов микросхем периферийных устройств, например, смене АЦП или индикатора приходится искать ранее скопированные участки кода и заменять их новыми. Это может вызвать новые ошибки и затрудняет написание и отладку программы. Поэтому проще хранить исходный текст программы в нескольких файлах, а соединение этих файлов в единую программу будет выполнять транслятор. Как правило, в один программный файл помещается участок кода, работающий с каким-то конкретным устройством или отвечающий за определенную подзадачу.

В языке C и ассемблере для этих целей используется ключевое слово (директива) *include*. Например, в языке C-51 директива включения программного файла описания регистров специальных функций МК с указанием их физических адресов запишется так: `#include <reg51.h>`. Такой файл называется заголовочным, так как обычно помещается в начале программы. В программе может быть несколько заголовочных файлов.

## 2. Описание лабораторных работ

### 2.1. Лабораторная работа № 1.

#### Основы работы с программным обеспечением Multisim

**Цель работы:** ознакомиться с интерфейсом программы Multisim, научиться создавать проект и программный файл.

#### 2.1.1. Интерфейс пользователя

##### 2.1.1.1. Основные элементы пользовательского меню

После установки программы в компьютере ее запуск осуществляется следующим образом: *Пуск – Все программы – National Instruments – Circuit Design Suite 10.0 – Multisim.*

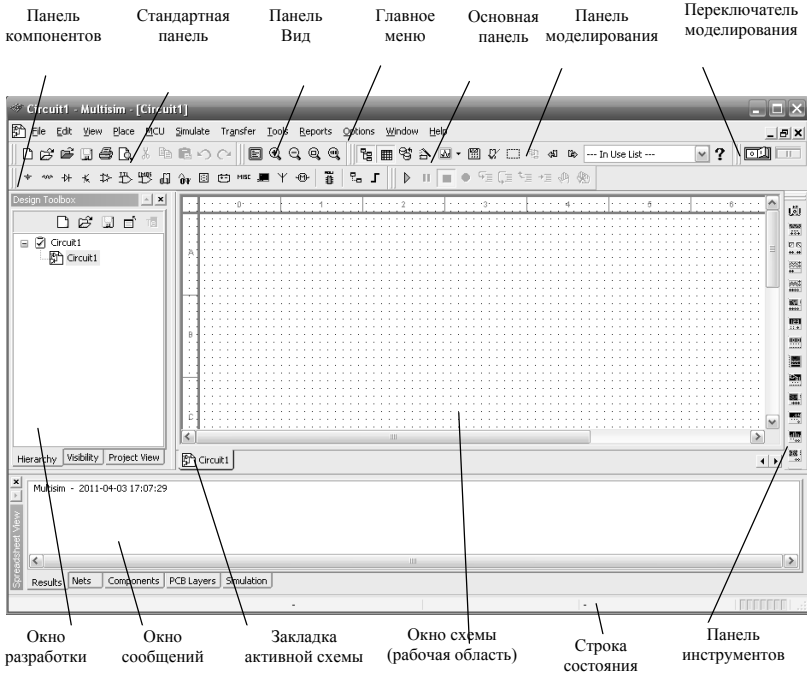



Рис. 2.1. Основные элементы пользовательского интерфейса

Открывающееся окно программы Multisim (рис. 2.1) выглядит стандартно для современного интерактивного программного про-

дукта. Основными элементами пользовательского интерфейса являются: главное меню, панель инструментов, строка состояния, полосы прокрутки и другие стандартные элементы окна программы Windows. Особенности работы с элементами меню программы Multisim 10 приводятся, например, в [5].

Для активизации окна сообщений необходимо нажать на основной панели значок  либо выбрать в пункте меню View-Spreadsheet View.

Главное меню программы Multisim обладает большим набором инструментов для подготовки схемы и проведения анализа.

*Пункты главного меню File, Edit, View* являются обычными меню с набором команд для работы с файлами и проектами, печати (File), редактирования и изменения свойств чертежа, ориентирования, удаления, выделения, перемещения элементов схем (Edit), настройки пользовательского интерфейса с возможностью изменения набора инструментальных панелей, масштабирования рабочей области (View).

***Пункт меню Place:***

Component... – предназначен для выбора и размещения компонентов схемы;

Wire – нанесение соединительных проводов;

Bus – построение информационных шин;

Junction – определение электрических узлов-соединителей;

Comment – определение комментария в схеме;

Connectors – введение соединителей;

Text – введение пояснительного текста;

Graphics – вставка элементарных графических элементов и графических изображений из внешнего файла;

Title Block – задание параметров штампа для схемы.

***Пункт меню MCU*** позволяет записать программный код для работы выбранного микропроцессора и отладить программу.

***Пункт меню Simulate*** задает типы проводимого анализа, позволяет выбрать встроенные приборы, сохранить результаты и провести их обработку. В этом меню можно изменить временной шаг моделирования при помощи пункта Interactive Simulation Settings (рис. 2.2).



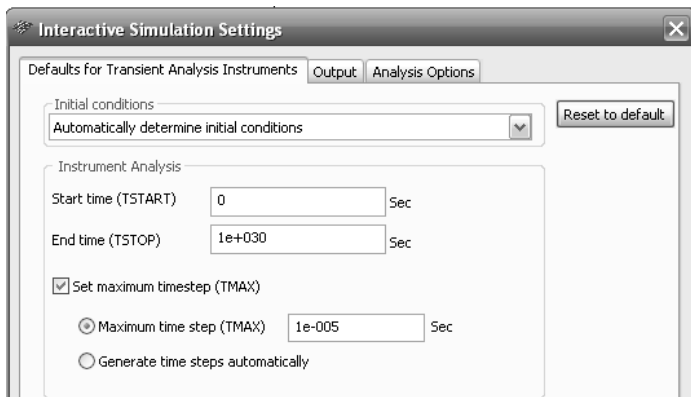


Рис. 2.2. Окно настройки временного шага моделирования

В программе по умолчанию выбрана автоматическая генерация шага моделирования ( $10^{-5}$ с). Для изменения временного шага моделирования нужно активировать строчку **Maximum time step (TMAX)** и указать необходимый шаг.

**Пункт меню Transfer** – здесь можно передать исходные данные для трассировки печатных плат в программу Ultiboard.

**Меню Tools** позволяет работать с базой данных компонентов программы, использовать возможности автоматизированного проектирования, мастера проектирования имеющихся типовых устройств на основе таймера 555 серии, полосовых фильтров, операционных усилителей и каскадов усиления на биполярных транзисторах, для которых можно выбрать требуемые выходные параметры.

В этом меню также можно подобрать исходные данные для многовариантного анализа исследуемой схемы, проверить схему на ошибки, редактировать имена электронных компонентов и даже реализовать дальнейшее сохранение полученной картинке в виде графического файла.

**Пункт меню Reports** предлагает детальный отчет о схеме: числе и типе компонентов, их параметрах, сведения об узлах схемы и многое другое. Кроме этого полученную информацию можно передать в офисные программы для дальнейшего использования.

**Пункт меню Option** задает условия работы по подготовке и сохранению схемы, определения внешнего вида схемы и условий ввода и размещения элементов при рисовании. Он содержит следующие

подменю: Global Preferences..., Sheet Properties... и Customize User Interface...

Подменю Global Preferences определяет режимы и условия работы программы как в процессе ввода схемы, так и при сохранении введенной схемы в виде файла. Так, на закладке Paths (рис. 2.3) дается путь до папок хранения файлов схем, файлов конфигурации и баз данных.

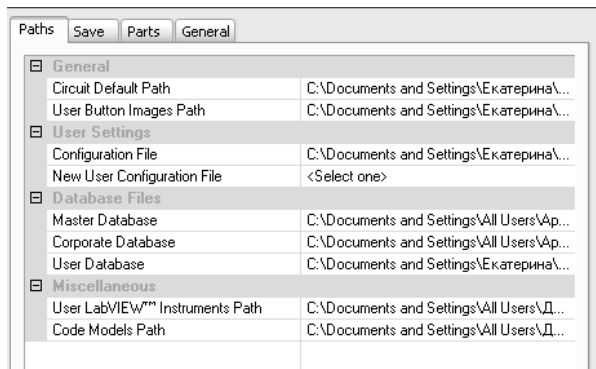


Рис. 2.3. Закладка Paths подменю Global Preferences пункта меню Option

На закладке Save (рис. 2.4) определяются режимы сохранения и размеры файла данных.

На закладке Parts (рис. 2.5) определяются действия программы при выборе и установке компонента в рабочую область схемы, выбор стандарта вида компонента, параметры автоматизации измерения результатов анализа, параметры моделирования цифровых устройств.

На закладке General (рис. 2.6) определяются действия при движении колесика мыши, задаются возможности автоматизации соединения проводников и возможности выбора выделяемых движением курсора областей.

Подменю Sheet Properties определяет внешний вид подготовленной, нарисованной схемы. На ее закладке Circuit (рис. 2.7) можно указать, какая информация будет отображаться на поле схемы возле введенного компонента.

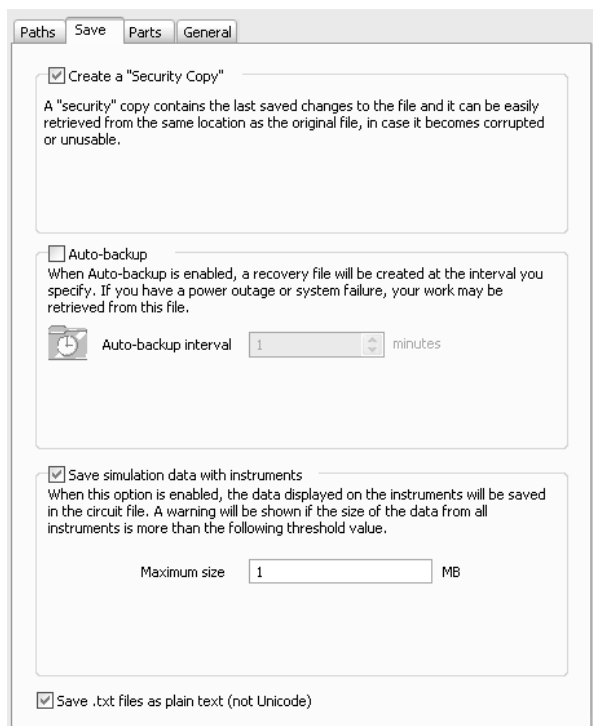


Рис. 2.4. Закладка Save подменю Global Preferences пункта меню Option

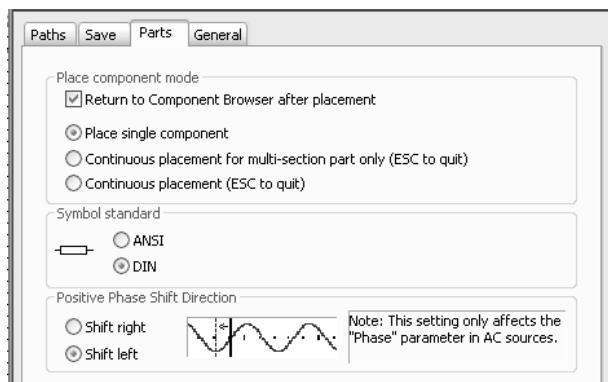


Рис. 2.5. Закладка Parts подменю Global Preferences пункта меню Option

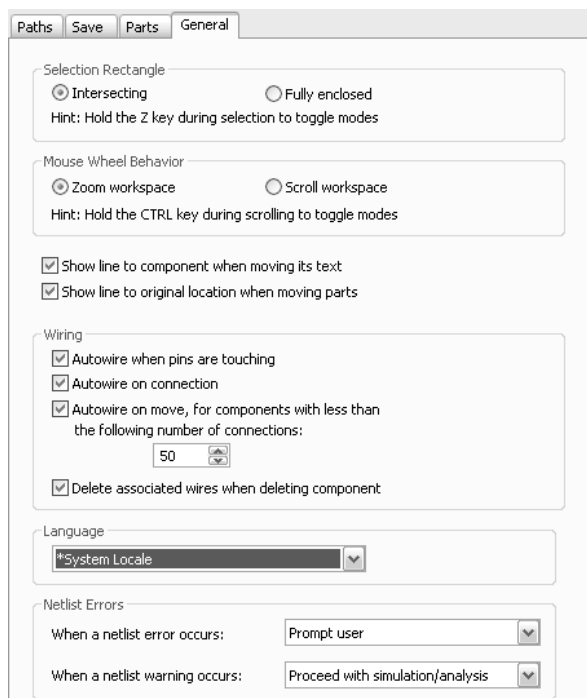


Рис. 2.6. Закладка General подменю Global Preferences пункта меню Option

Рекомендуется оставить для отображения только самую необходимую информацию: текстовые метки (Labels), позиционное обозначение (RefDes), его значение (Values). С помощью этой же закладки можно задать также цветовое решение элементов, соединительных проводов, фона схемы из стандартных наборов или же задать пользовательскую цветовую палитру (Color). Если затем нажать экранную кнопку окна ОК, то все выбранные настройки сохранятся только для текущей схемы и не будут использоваться в следующих схемах. Для сохранения настроек для следующих схем необходимо отметить флажок Save as default и уже после этого нажать ОК.

Закладка Workspace (рис. 2.8) позволяет установить видимость сетки, границы чертежа, показ границы страницы, определить размер страницы для схемы из стандартных наборов, ее ориентацию

или же определить нестандартные размеры страницы и метрику размеров.

Закладка Wiring (рис. 2.9) определяет толщину линий соединительных проводов и толщину изображения шин.

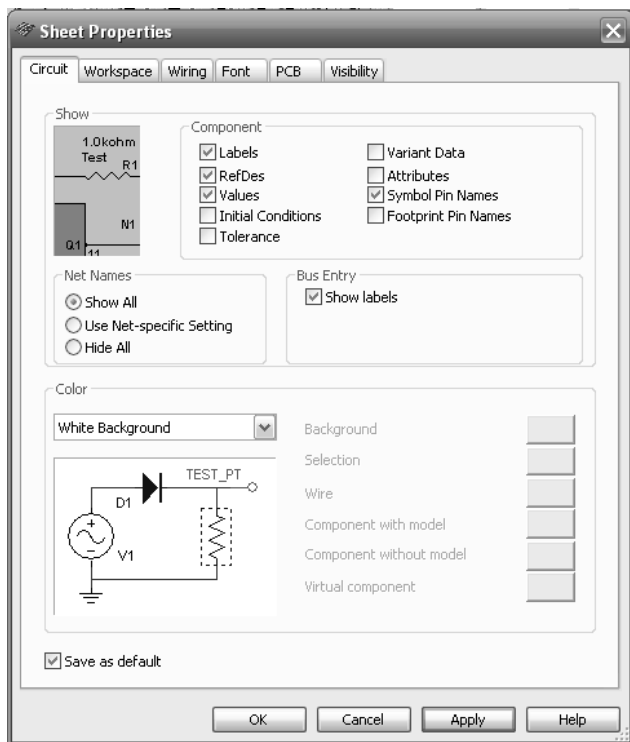


Рис. 2.7. Закладка Circuit подменю Sheet Properties пункта меню Option

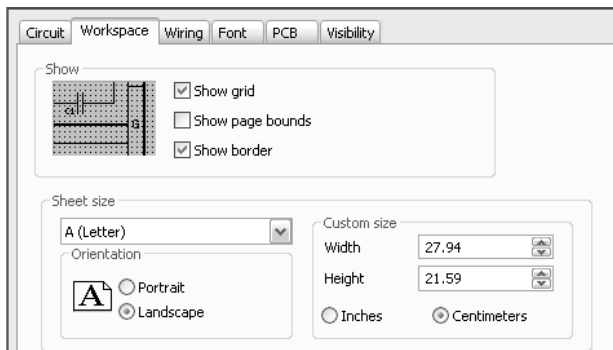


Рис. 2.8. Закладка Workspace подменю Sheet Properties пункта меню Option

Пункт меню Window – стандартный, осуществляет навигацию между внутренними окнами программы и регулирует расположение этих окон. Аналогичное можно сказать и о пункте меню Help.

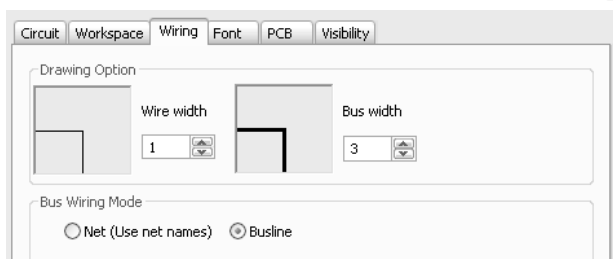


Рис. 2.9. Закладка Wiring подменю Sheet Properties пункта меню Option

### 2.1.1.2. Виртуальные инструменты

Программа Multisim содержит большое число виртуальных измерительных приборов (инструментов), которые можно использовать с целью проведения измерения или же исследования схемотехнических решений. Виртуальные измерительные приборы по своему действию соответствуют реальным приборам. С их помощью можно не только визуализировать информацию, но и сохранить ее в виде файла данных, который в дальнейшем можно будет использовать для обмена с другими программами, например **LabVIEW**.

Панель инструментов на экране может быть расположена произвольно, но, как правило, она закрепляется у границ окна. Вид панели представлен на рис. 2.10.






Рис. 2.10. Вид панели инструментов

Измерительные приборы могут иметь разный внешний вид, в зависимости от того, какую задачу ставит перед собой пользователь и где расположен сам прибор (на панели инструментов или на поле схемы), пример показан в табл. 2.1.

Таблица 2.1

Представление виртуальных приборов в Multisim

Форма представления	Описание	Внешний вид
Иконка	Представляет инструмент в панели инструментов Multisim's NI	
Символ	Представляет инструмент в цепи схемы. Для подсоединения к схеме необходимо использовать внешние выводы инструмента. Для открытия приборной панели необходимо дважды щелкнуть ЛКМ на символе инструмента	
Инструментальная панель (панель прибора)	Открывается двойным щелчком ЛКМ на символе инструмента. Позволяет пользователю взаимодействовать с инструментом – установить параметры измерения. Отображает результаты измерения	

Приборы Multisim позволяют пользователю измерять параметры моделируемой схемы, даже если он не знаком с основами языка моделирования SPICE. Если пользователь изменяет настройки прибора, тут же автоматически изменяются и параметры моделирования.

При проведении моделирования показания приборов постоянно изменяются. В одной и той же схеме может быть несколько экземпляров прибора. Атрибуты настройки прибора и соответствующие этим настройкам параметры моделирования могут быть сохранены в конфигурационном файле. Полученные при моделировании данные при использовании встроенных приборов могут быть обработаны постпроцессором и показаны в окне Grapher View. Внешний вид (размеры) инструментальной панели прибора могут быть изменены

в соответствии с требуемым разрешением экрана и способом отображения данных. Данные, полученные в результате анализа, могут быть сохранены в формате файлов TXT, LVM, и TDM.

NI Electronics Workbench Group имеет тесные партнерские связи с представителями ведущих фирм в области измерительной техники, таких Agilent® и Tektronix®, поэтому приборы, размещенные на панели инструментов Multisim, выглядят и работают абсолютно так же, как и реальные физические приборы этих производителей.

Встроенные в программу Multisim приборы могут быть сгруппированы по шести категориям (табл. 2.2–2.7).

Таблица 2.2

### Инструменты для анализа напряжения и токов

Тип прибора	Функциональные возможности	Иконка
1	2	3
Функциональные генераторы (Function generator)	Генерирование синусоидальных, трапецидальных и импульсных сигналов. Установка частоты, скважности, амплитуды сигнала	
Мультиметр (Multimeter)	Измерение постоянного и переменного тока, напряжения и потерь.	
2-канальный осциллограф (Oscilloscope)	Измерение сигнала в двух каналах. Масштабирование Y и X осей. Смещение по Y оси. Синхронизация	
4-канальный осциллограф (4 channel scilloscope)	Измерение сигнала в четырех каналах. Масштабирование Y и X осей. Смещение по Y оси. Синхронизация	
Ваттметр (Wattmeter)	Измерение мощности сигнала	
Измеритель ВАХ (IV-analysis)	Исследуются диоды, биполярные PNP и NPN-транзисторы (BJT). Канальные транзисторы (PMOS), (NMOS) и полевые. КМОП структуры (CMOS)	
Счетчики (Frequency counter)	Измеряются частота, период, фронты импульсов, АЧХ, фазовые сдвиги. Поддерживается частота измерений свыше 10 ГГц, синхронизация, развязка по постоянному току	
Построитель графика Бодэ (Bode plotter)	Исследуются частотная характеристика, фазовые сдвиги. Поддерживается частота измерений свыше 10 ГГц	
Измеритель частотных искажений (Distortion analyzer)	Измеряются интермодуляционные искажения, суммарный коэффициент гармонических искажений (коэффициент гармоник)	



Таблица 2.3

## Логические инструменты




Тип прибора	Функциональные возможности	Иконка
Логический анализатор (Logic analyzer)	Измеряются 16 каналов, история измерений. Поддерживается синхронизация. Внешняя/внутренняя опорная частота	
Логический конвертер (Logic converter)	Цифровые схемы, построенные по таблицам истинности и логическим выражениям. Таблицы истинности для цифровых схем. Логические выражения для цифровых схем. Реализуются циклы, обновление шага, сброс. HEX, DEC, Boolean, ASCII-коды	
Генератор слов (Word generator)	Реализуются HEX, DEC, Boolean, ASCII представление данных, синхронизация, временная селекция. Режимы: циклы, обновление шага, сброс	

Таблица 2.4

## Приборы радиочастотного диапазона



Тип прибора	Функциональные возможности	Иконка
Анализатор спектра (Spectrum analyzer)	Измеряются спектр, компоненты спектра (мощность, частота), непрерывный и дискретный спектр	
Прибор для анализа электрических цепей в обобщенном виде (Network analyzer)	Построение по цифровой схеме таблицы истинности или логического выражения. Обратное преобразование таблицы истинности или логического выражения в цифровую схему	

Таблица 2.5

## Инструменты, моделирующие измерительные приборы фирм-производителей измерительных устройств








Тип прибора	Функциональные возможности	Иконка
Генератор Agilent (Agilent function generator)	Тип генератора 33120A. Моделирование реального прибора	
Мультиметр DMM Agilent (Agilent multimeter)	Тип генератора 34401A. Моделирование реального прибора	
Оциллограф Agilent (Agilent oscilloscope)	Тип осциллографа 54622D. Моделирование реального прибора.	
Оциллограф Tektronix (Tektronix oscilloscope)	Тип осциллографа TDS 2024. Моделирование реального прибора	

Таблица 2.6





## Измерительные пробники

Тип прибора	Функциональные возможности	Иконка
Пробник	Измерения тока, напряжения и частоты относительно земли	
Пробник	Измерения тока, напряжения и частоты относительно другого пробника	
Пробник	Имитация поведения токовых измерителей (токовых клещей)	

В приложении рассматриваются основы настройки часто используемых виртуальных приборов: генератора слов, логического анализатора, функционального генератора.

Таблица 2.7

## Инструменты, базирующиеся на виртуальных приборах NI LabVIEW

Имя прибора	Функциональные возможности	Иконка
Микрофон	Подключение к звуковой плате компьютера. Запись звука	
Динамик	Подключение к звуковой плате компьютера	
Анализатор сигнала	Анализ сигнала во временной области. Спектр мощности	
Генератор сигнала	Гармонический, импульсный, пилообразный, треугольный сигналы	

**2.1.1.3. Организация базы данных Multisim**

Элементы схемы выбираются из базы данных и размещаются на рабочем поле тремя способами:

- 1) Через Главное меню (Place – Component...);
- 2) Через контекстное меню рабочей области (Place Component...);
- 3) Через панель компонентов (рис. 2.11).

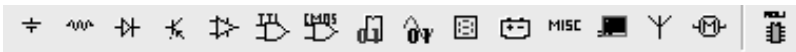


Рис. 2.11. Панель компонентов

Имеются следующие группы элементов:

 **Группа Sources – источники энергии и сигналов**

Power sources – источники питания (AC – источник питания переменного тока, DC – источник питания постоянного тока, DGND – циф-

ровая земля, GROUND – аналоговая земля, Vcc, Vdd – положительное напряжение питания, Vee, Vss – отрицательное напряжение, земля).

Аналоговое заземление используется во всех процессах моделирования за исключением моделирования цифровых устройств в реальном времени (в этом режиме, кроме задержки в логическом элементе, моделируется время фронта и время спада, выходное напряжение имеет сглаженные фронты, а сам процесс моделирования требует большего времени).

Signal voltage sources – сигнальные источники напряжения.

Signal current sources – сигнальные источники тока.

Controlled voltage sources – регулируемые источники напряжения.

Controlled current sources – регулируемые источники тока.

Control function blocks – функциональные блоки управления.



### **Группа Basic – группа с базовыми элементами**

Rpack – резистивная сборка.

Switch – переключатели (ключи, push button – кнопка, блоки с 2–10 переключателями).

Transformer – трансформатор.

Non linear transformer – нелинейный трансформатор.

Relay – реле.

Connectors – соединители, разъемы.

Sockets – сокет.

Resistor – резисторы.

Capacitor – конденсаторы.

Inductor – катушки индуктивности.

Cap electrolit – электролитические конденсаторы.

Variable capacitor – переменные конденсаторы.

Variable inductor – переменные катушки индуктивности.

Potentiometer – потенциометры.



### **Группа Diodes – диоды**

Diode – диоды.

Zener – стабилитроны (диоды Зенера).

Led – светодиоды.

FWB – диодные мосты.

Schootky diode – диоды Шоттки.

Scr – тиристоры триодные, запираемые в обратном направлении с управлением по катоду.

Diac – диоды двунаправленные.

Triac – тиристоры триодные симметричные (двунаправленные).

Varactor – варикапы (емкостные диоды).

Pin diod – pin диоды (содержат область собственной проводимости между сильнолегированными областями).



### **Группа Transistors – транзисторы**

BJT NPN – биполярные транзисторы типа NPN.

BJT PNP – биполярные транзисторы типа PNP.

BJT ARRAY – микросборки транзисторов.

DARLINGTON NPN – транзисторы Дарлингтона NPN (составные транзисторы).

DARLINGTON PNP – транзисторы Дарлингтона PNP.

DARLINGTON ARRAY – массив транзисторов Дарлингтона.

MOS 3TDN – транзистор канальный с встроенным каналом N-типа.

MOS 3TEN – транзистор канальный с изолированным затвором обогащенного типа с N-каналом, с внутренним соединением истока и подложки (индуцированный канал).

MOS 3TEP – транзистор канальный с индуцированным каналом P-типа.

JFET N – транзистор полевой с проводимостью типа N.

JFET P – транзистор полевой с проводимостью типа P.

POWER MOS N – мощный канальный транзистор с каналом N-типа.

POWER MOS P – мощный канальный транзистор с каналом P-типа.

POWER MOS COM – мощный канальный транзистор (комплементарная технология).

UJT – тиристор триодный, запираемый в обратном направлении с управлением по аноду.

THERMAL MODELS – температурные модели.



### **Группа Analog – аналоговые компоненты**

Opamp – операционные усилители.

Comparator – компараторы.

Wideband amps – широкополосные усилители.

Special function – компоненты, реализуемые специальные функции.



**Группа TTL** – элементы транзисторно-транзисторной логики 74-серии



**Группа CMOS** – комплементарная МОП-структура (комплементарные транзисторы)



**Группа MCU**

*Микроконтроллеры* – МК (805х, PIC).

*Микросхемы памяти* – RAM, ROM (HM-65642-883 (8k x 8), HM61116A120(2k x 8)), ПЗУ (27C128-12L(16k x 8), 27C256-15L (32K x8)), ППЗУ (27C64Q350-883)).



**Группа Advanced peripherals**

*Усовершенствованные периферийные устройства*, такие как виртуальная цифровая клавиатура (4x4, 4x5), LCD-дисплей, светофор.



**Группа Misc digital** – различные цифровые микросхемы

DSP – устройства DSP (цифровые сигнальные процессоры).

FPGA – устройства FPGA (программируемая пользователем вентиляционная матрица).

PLD – программируемые логические устройства.

CPLD – комплементарные программируемые логические схемы.

Microcontrollers – микроконтроллеры.

Microprocessors – микропроцессоры.

Memory – микросхемы памяти.

Line driver – линейный формирователь.

Line receiver – линейный приемник.

Line transceiver – линейные приемопередатчики.



**Группа Mixed** – устройства смешанного сигнала

Analog switch – аналоговые переключатели.

Analog switch IC – интегральная схема аналогового переключателя.

Timer – таймер.

ADC, DAC – АЦП (ADS8364Y, AD16), ЦАП (DAC7643\_FP32).

Multivibrators – мультивибраторы.



**Группа Indicators** – индикаторы

Voltmeter – вольтметры.

Ammeter – амперметры.

Probe – пробники.

Buzzer – автоматические прерыватели.

Lamp – лампы.

Hex display – дисплеи (светоиндикаторы – 15-сегментные, семи-сегментные, с общим катодом, с общим анодом, с десятичной точкой, без точки, дисплеи с «+» или «-», 7 сегментные дисплеи с двумя цифрами (с десятичной точкой, с общим анодом или с общим катодом).

Bar graph – столбцовая диаграмма.



### **Группа Power – компоненты, относящиеся к источникам питания и связанные с ними:**

Fuse – плавкие предохранители.

Voltage reference – источники опорного напряжения.

Voltage regulator – потенциометры.

Voltage suppressor – ограничительные диоды.

Power supply controller – контроллеры источников питания

Misc power – прочие источники питания.

PWM controller – широтно-импульсный модулятор.



### **Группа Misc – прочее**

Optocoupler – оптроны.

Crystal – кварцевые резонаторы.

Vacuum tube – электронные лампы.

Boost converter – усилители-преобразователи.

Lossy transmission line – линия передачи с потерями.

Lossless line type 1 – линия без потерь, тип 1.

Lossless line type 2 – линия без потерь, тип 2.

Filters – фильтры.

Mosfet driver – драйвер полевого транзистора.

Net – сеть связи.



### **Группа RF – радиочастотные устройства**

RF capacitor – радиочастотные конденсаторы.

RF inductor – радиочастотные катушки индуктивности.

RF BJT NPN – радиочастотные биполярные транзисторы типа NPN.

RF BJT PNP – радиочастотные биполярные транзисторы типа PNP.

RF MOS 3TDN – радиочастотные полевые транзисторы с встроенным каналом N-типа.

Tunnel diode – радиочастотные туннельные диоды.

Strip line – полосковые линии.



## Группа Electro Mechanical – электромеханические устройства

Sensing switches – сенсорные переключатели.

Momentary switches – мгновенные переключатели.

Supplementary contacts – дополнительные контакты.

Timed contacts – синхронизированные контакты.

Coils relays – реле.


Protection devices – элементы защиты (предохранители).

Output devices – устройства вывода.

Все рассмотренные выше компоненты являются реальными (промышленными) и имеют определенные, неизменяемые значения параметров. В схеме по умолчанию они обозначаются синим цветом. В базе данных есть также и виртуальные компоненты, имеющие в своем названии приставку VIRTUAL (неявно отображаются на схеме черным цветом). Виртуальные компоненты необходимы для исследований, так как пользователь может назначить им произвольные значения параметров.

### 2.1.2. Создание проекта и программного файла

При открытии программы Multisim автоматически создается проект схемы под названием Circuit1. Для изменения имени схемы необходимо сохранить его через пункт меню File-Save As, желательно использовать в директории английские буквы. Для создания проекта и программного файла следует выбрать микроконтроллер для программирования. В программе Multisim программируются микроконтроллеры, расположенные в БД в группе MCU, при этом МК выбирается двумя способами:

- через пункт меню Place – Component – MCU – 805x – 8051;
- через панель компонентов:  – 805x – 8051.

Устанавливаем микроконтроллер на рабочей области, появляется всплывающее окно «Мастер по созданию программного файла», которое предлагает выполнить три шага для создания проекта и программного файла.

#### *Шаг 1. Определение рабочего пространства (рис. 2.12)*

В первой строке всплывающего окна указывается путь рабочего пространства для выбранного МК. Используя кнопку «Browse», можно изменить путь рабочего пространства, предложенный программой.

В следующей строке окна предлагается ввести имя рабочего пространства.

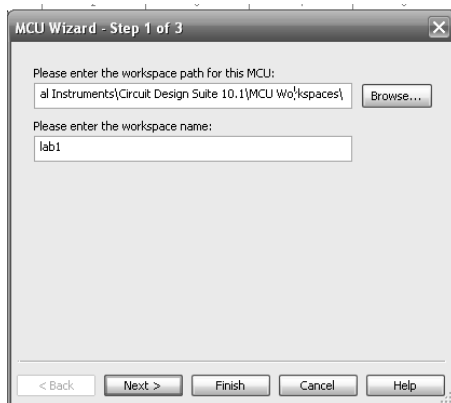


Рис. 2.12. Окно задания рабочего пространства

*Шаг 2. Создание проекта для микроконтроллера (рис. 2.13)*

В этом окне предлагается установить следующие настройки для будущего проекта:

- 1) тип проекта: Standard или Use External Hex File;
- 2) язык программирования: C, Assembly;
- 3) компилятор;
- 4) имя проекта.

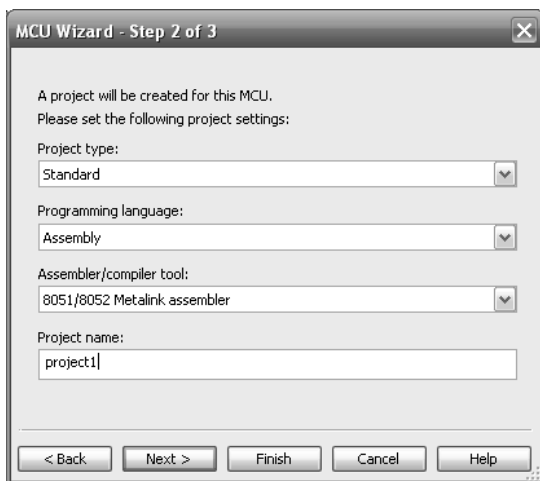


Рис. 2.13. Создание проекта для микроконтроллера



### Шаг 3. Создание программного файла (рис. 2.14)

В этом окне предлагается создать либо пустой проект, то есть без программного файла, либо добавить исходный программный файл, указав его имя. Работа с Мастером заканчивается нажатием кнопки Finish.

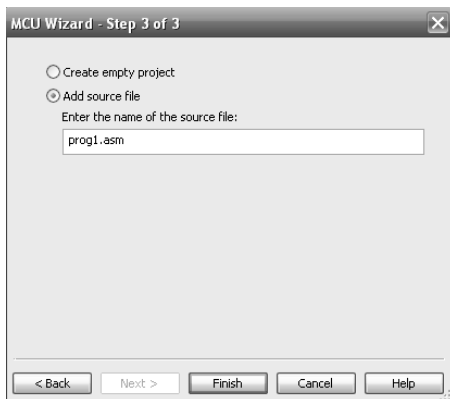


Рис. 2.14. Создание программного файла

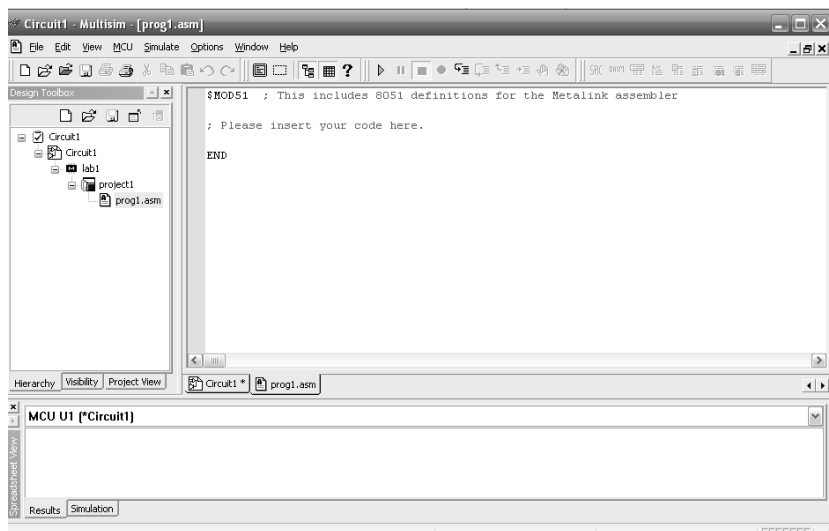


Рис. 2.15. Окно программного файла

В окне Design Toolbox (рис. 2.15) на закладке Hierarchy возможно просмотреть структуру созданного проекта. Открытие программного файла осуществляется двойным щелчком ЛКМ по его названию в окне Design Toolbox.

### **2.1.3. Задания для лабораторной работы**

Согласно варианту задания нарисовать схему с использованием МК-51 и указанных элементов (табл. 2.8). Выполнить соединения элементов (произвольно), ввести позиционные обозначения и нумерацию цепей. При выполнении задания использовать следующие стандарты: ГОСТ 2.702-75 «Правила выполнения электрических схем»; ГОСТ 2.710-81 «Правила выполнения схем». Соединения элементов с МК обозначить зеленым цветом. Также сравнить реальный и виртуальный компоненты, указанные в задании (объяснить, в чем состоит их отличие). Элементы схемы выбрать самостоятельно из базы данных Multisim.

Размещение компонентов производится через пункт меню Place или горячую клавишу Ctrl-W, которые вызывают обращение к проводнику компонентов (рис. 2.16).

В проводнике компонентов отображается текущая база данных со схемными элементами. В Multisim они организованы в группы (groups) и семейства (families). Также в проводнике компонентов отображается описание компонента (поле Function), модель и печатная плата или производитель.

Для поиска нужного элемента схемы необходимо набрать название компонента в поле Component, и проводник автоматически подберет подходящие элементы. Также требуемый элемент схемы можно найти в соответствующей ему группе (Group). При помощи подменю «Поиск» (Search) открывается расширенный поиск элементов.

Символ звездочки («\*») в названии компонента заменяет любой набор символов. Например, среди результатов запроса на элемент "74LS\*N" будут микросхемы «74LS01N» и «74LS183N».

При работе с компонентами следует иметь в виду, что любому компоненту соответствует определенная модель в БД, учитывающая различные физические характеристики компонента. Например, операционный усилитель LM358M имеет 5 внешних контактов, но в этой модели БД из них используется только 3, контакты питания не задействованы (неявно заданы). Информация об особенностях используемой модели элемента находится в поле проводника «Производи-

тель/идентификатор» (Model Manuf.\ID), для этого необходимо выделить ЛКМ «Модель» (Model).

Двойной щелчок ЛКМ по компоненту или нажатие кнопки ОК в окне проводника компонентов прикрепит его к курсору. После этого компонент помещается на схему в желаемом месте рабочего пространства при помощи ЛКМ. До установки или после установки элемента в схему его можно повернуть по/против часовой стрелки при помощи горячей клавиши Ctrl-R/Ctrl-R-Shift или выбрать в контекстном меню пункт «90 Clockwise» или «90 Counter CW» соответственно.

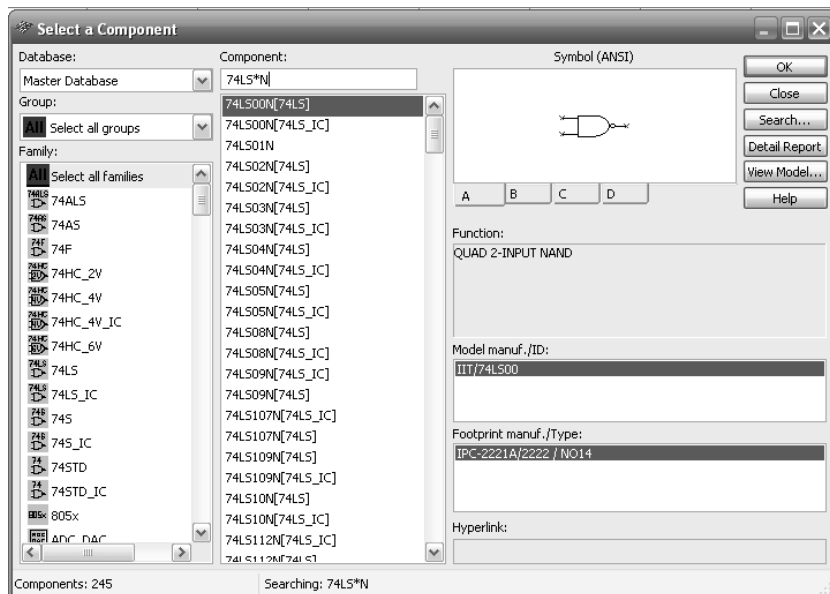


Рис. 2.16. Проводник компонентов

Чтобы выбрать компонент на схеме, необходимо щелкнуть по нему ЛК мыши. Для одновременного выбора нескольких компонентов требуется прижать ЛК мыши и перемещать ее, рисуя прямоугольник вокруг нужных компонентов. Выбранные на схеме компоненты выделяются пунктирной линией. Выделение отдельных атрибутов компонента, например значения или метки, осуществляется одинарным щелчком по соответствующему атрибуту. Клавиша Shift позволяет добавлять или снимать выделение с нескольких компонентов.

Выбранные компоненты из БД можно заменить на другие, подобные компоненты с помощью их контекстного меню, пункта Replace Components, при этом открывается окно проводника компонентов. После замены элемента Multisim восстановит соединения с остальными элементами схемы.

Таблица 2.8

Варианты задания

№	Компоненты схемы	Элемент для сравнения
1	8-разрядный регистр защелка, биполярный транзистор PNP	реле
2	JK-триггер, кнопка	конденсатор
3	EPROM 16Kx8, операционный усилитель	Биполярный транзистор NPN
4	RAM 2Kx8, конденсатор	светодиод
5	дешифратор для семисегментного индикатора источник напряжения $V_{cc}$	катушка индуктивности
6	потенциометр, RAM 8Kx8	диод Шотки
7	четыре элемента 2 И-НЕ, светодиод	транзистор биполярный PNP
8	компаратор, биполярный транзистор NPN	резистор
9	катушка индуктивности, D-триггер	операционный усилитель
10	Диодный мост, регистр сдвига	кварцевый резонатор
11	АЦП, пробник	оптрон
12	15- сегментный индикатор с общим катодом, предохранитель	ЦАП

#### 2.1.4. Содержание отчета

1. Наименование и цель работы.
2. Перечень элементов, использованных в схеме, с их краткими характеристиками.
3. Копия окна схемного файла с позиционными обозначениями и нумерацией цепей.
4. Выводы по работе.

#### 2.1.5. Вопросы для самоконтроля

1. Как изменить цветовое решение схемы?
2. Чем отличаются реальные (промышленные) компоненты от виртуальных?
3. Возможно ли изменить время моделирования процесса? Ответ пояснить.

4. Привести примеры многовентильных логических компонентов, имеющих в БД Multisim.
5. Какие виртуальные приборы используются для анализа схем по току?
6. В БД Multisim представлены цифровое и аналоговое заземления. В чем их отличие?
7. Каким образом можно отредактировать цепи?
8. Какие программируемые устройства представлены в Multisim, и какие языки программирования они используют?
9. Какие типы дешифраторов имеются в БД Multisim?
10. Какие опции ПО позволяют уменьшить количество выводимой информации для элементов схемы и исключить показ сетки?
11. Расскажите об особенностях работы генератора слов.
12. Расскажите об особенностях работы логического анализатора.

## 2.2. Лабораторная работа № 2.

### Подключение внешней памяти и ее тестирование

**Цель работы:** разработать схему подключения микроконтроллера с внешней памятью и протестировать память.

#### 2.2.1. Особенности подключения к МК внешней памяти и периферийных устройств

Микроконтроллер 8051 может работать с внешней памятью данных емкостью до 64 КБайт, построенной на одной или нескольких микросхемах статической памяти.

В БД Multisim имеются микросхемы RAM с байтовой организацией объемом 2Кх8 и 8Кх8 бит (рис. 2.17).

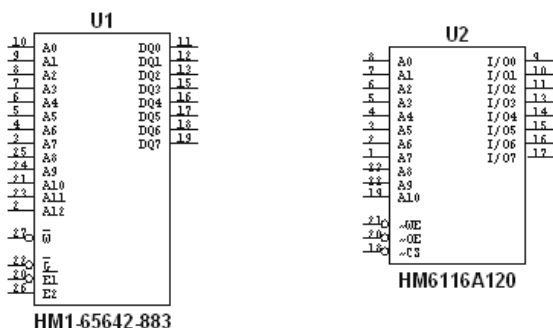


Рис. 2.17. Статическая RAM 8Кх8 и 2Кх8

Такие микросхемы имеют 8 двунаправленных выводов данных (D0–D7), 11 или 13 адресных входов (A0–A10 или A0–A12). Вход WE (W) определяет характер обращения: если на нем установлена 1, то осуществляется чтение из выбранной ячейки, при WE = 0 в ячейку будет записана информация. Вход CS (E1,E2) активизирует микросхему памяти: когда на входе CS установлена 1, она выключена, при CS = 0 допускается любое обращение к памяти. Нулевой сигнал на входе OE (G) разрешает работу выходной шины данных микросхемы.

В БД Multisim также представлены микросхемы ПЗУ (рис. 2.18): ROM (32Кх8), EPROM (8Кх8, 16Кх8). Такие микросхемы в рабочем режиме допускают только считывание информации. Выводы микросхем аналогичны микросхемам RAM, кроме вывода PGM, отвечаю-

шего за программирование. Подробно работа микросхем ОЗУ и ПЗУ рассматривается, например, в [8].

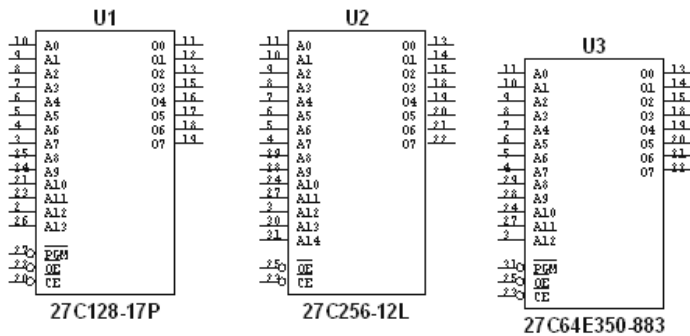


Рис. 2.18. ПЗУ EPROM 16Kx8, ROM 32Kx8 и EPROM 8Kx8

В микроконтроллерах МК51 существует 4 многофункциональных 8-битовых порта ввода/вывода P0, P1, P2, P3, предназначенных для обмена информацией с различными внешними устройствами и для выполнения специализированных функций, таких как подключение внешней памяти программ, данных, программирование внутреннего ПЗУ и др.

Каждый порт может адресоваться как побайтно, так и побитно, по конкретным физическим адресам. При подключении к МК внешней памяти через порт P0 выводится младший байт адреса, а также передается и принимается в микроконтроллер байт данных (в мультиплексированном режиме). В 1 и 2 тактах машинного цикла при обращении к внешней памяти на линиях P0 активизируется адресная информация A0–A7 при высоком уровне сигнала ALE, а затем на этих же линиях появляется сигнал D0–D7 (при низком уровне сигнала ALE). Для фиксации байта адреса в течение всего машинного цикла используются регистры-защелки, например, 74LS373N, информация в которых фиксируется по спаду сигнала на его входе ENG (рис. 2.25).

Через порт P2 выводится старший байт адреса (разряды A8–A15) внешней памяти программ и данных. Для каждого из битов порта P3 имеется ряд альтернативных функций. Сигналы стробов записи (WR#) и чтения (RD#) внешней памяти формируются на линиях P3.6 и P3.7 соответственно. Альтернативные функции всех портов реализуются только в том случае, если в соответствующий разряд фиксатора-защелки порта записана логическая «1», иначе на соответствующем выводе будет присутствовать «0».

Каждый вывод портов P0–P3 может использоваться как вход или выход независимо от других. Для настройки линии порта на ввод информации необходимо в соответствующий разряд порта записать «1», а для использования в качестве выхода – «0». При системном сбросе в регистрах защелках всех портов устанавливается значение FFh.

## 2.2.2. Порядок выполнения лабораторной работы

### 2.2.2.1. Создание схемного проекта

Открываем и сохраняем новый схемный проект Circuit2.

Размещаем на рабочем поле МК-51 (см. лабораторную работу № 1), микросхему памяти емкостью 2 Кбайта (Place – Component – MCU – RAM), регистр-защелку, например 4037BP, которую можно найти в группе CMOS в семействе CMOS\_5V, землю и питание (Place – Component – Sources – POWER\_SOURCES). Собранный электронная схема представлена на рис. 2.25.

После выбора компонентов из БД и размещения их на схеме, необходимо соединить компоненты между собой. В программе Multisim действие мышью на схеме зависит от положения курсора. Внешний вид курсора меняется в зависимости от того, на какой объект он наведен (рис. 2.19.) Для того чтобы провести соединяющий провод между элементами, необходимо кликнуть ЛКМ по выводу одного элемента, затем, не отпуская кнопку, провести соединение до вывода другого элемента. После появления соединяющего проводника между элементами Multisim автоматически присвоит ему номер. Нумерация линий увеличивается последовательно, начиная с 1.

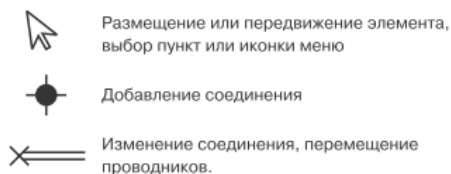


Рис. 2.19. Виды курсора

Заземляющие провода всегда имеют номер 0, это требование связано с работой скрытого эмулятора SPICE. Чтобы изменить нумерацию соединения или присвоить ему логическое имя, необходимо дважды кликнуть по соединительному проводнику (рис. 2.20).



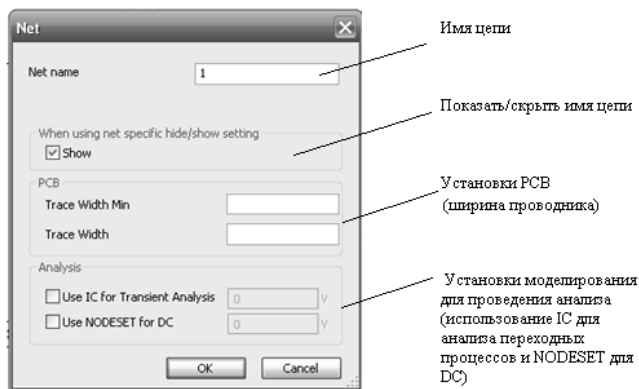



Рис. 2.20. Установки цепи

В Multisim имеется функция автоматического соединения выводов между собой и с проводниками. При добавлении компонента к существующей сети соединений необходимо, чтобы выводы компонента касались существующей сети, либо поместить компонент параллельно соединению.

Для размещения соединяющей шины в схемном проекте необходимо выбрать пункт меню Place – Bus или значок на панели компонентов  или горячую клавишу Ctrl-U.

Рассмотрим подключение требуемых компонентов к шине на примере компонента U3 (см. рис. 2.25). При помощи ЛКМ размещаем шину напротив выводов, необходимых для подключения компонентов, например, напротив выводов D0–D7 элемента U3, затем фиксируем положение шины ПКМ. Далее в контекстном меню компонента U3, соединяемого с шиной, нажимаем Bus Vector Connect (рис. 2.21).

Слева окна соединений представлены необходимые данные о компоненте (обозначение, положение выводов, выводы), справа – о шине (обозначение, шинные линии). Выбираем в подменю Pins выводы компонента 1D–8D для подключения к шине, удерживая клавишу Shift и при помощи стрелок отправляем их в нижнее окошко. Справа окна соединений выбираем в поле Name шину, в нашем случае Bus1. Далее необходимо сформировать вектора (линии шины), Multisim предлагает для этого два способа: ручной и автоматический.

**Формирование векторов вручную.** Нажимаем кнопку Add buslines (см. рис. 2.21) для задания векторов, появляется соответствующее окно (рис. 2.22).

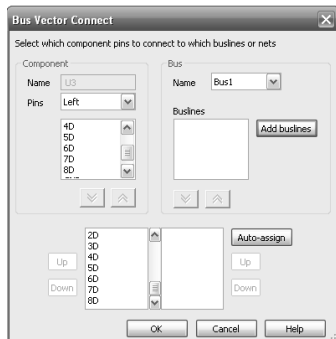


Рис. 2.21. Окно соединения компонента с шиной

Здесь указываются префикс (метка), например, *In*, начальное значение вектора, инкремент и количество векторов (в нашем случае их 8). После нажатия на кнопку ОК (рис. 2.22) полученные линии шины (вектора) отразятся в предыдущем окне. Количество линий шины должно быть эквивалентно выбранным для подключения к шине выводам. Далее выделяем полученные линии шины и нажимаем кнопку ОК (см. рис. 2.21).

**Автоматическое формирование векторов.** Вектора вводятся автоматически при нажатии кнопки Auto-assign (см. рис. 2.21).

В завершении нажимаем кнопку ОК. В таком же порядке подключаем к шине выводы МК51.

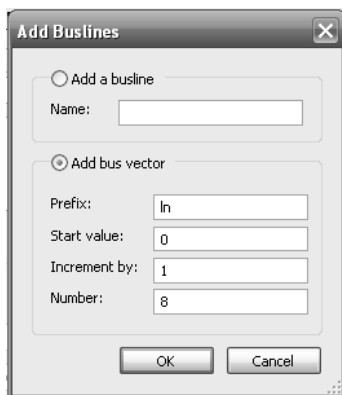


Рис. 2.22. Окно задания линий шин вручную

При подключении внешней памяти аналогичным образом рисуем шину напротив необходимых выводов элемента U2 и фиксируем ее. Эта шина автоматически будет названа Bus2. Для объединения шины Bus2 к шине Bus1 необходимо выбрать в контекстном меню Bus2 пункт Properties (рис. 2.23).

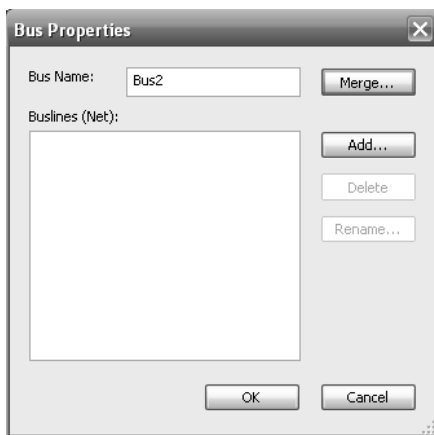


Рис. 2.23. Окно свойств шины

В открывшемся окне нажимаем кнопку Merge. В следующем открывшемся окне (рис. 2.24) слева указана первая шина, предназначенная для объединения Bus1, а справа вторая – Bus2. В окне объединения шин в поле Name второй шины выбираем Bus1, после чего в окошке Buslines появятся заданные ранее вектора Bus1. Ниже указана объединенная шина с её векторами.

Далее в окне объединения шин нажимаем кнопку Merge и в окне свойств шины кнопку ОК. Как и ранее, выбираются выводы компонента и вектора шины, только в этом случае при нажатии кнопки ОК Multisim предупреждает, что фрагмент шины, к которой необходимо произвести подключение, выбирается двойным кликом ЛКМ. Готовая схема представлена на рис. 2.25.

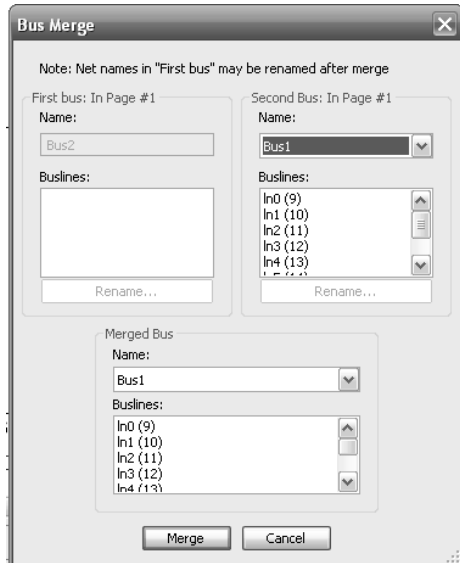


Рис. 2.24. Окно объединения шин

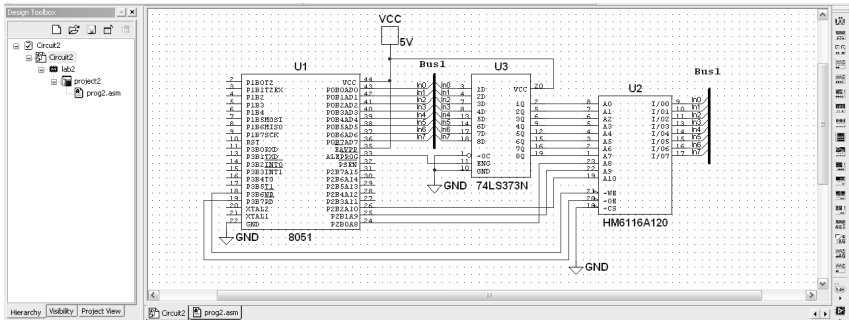


Рис. 2.25. Схема подключения внешней памяти к МК-51

В окне разработки (см. рис. 2.25) указана вся структура созданного проекта.

В заключении в свойствах МК на вкладке Value в поле Built-in External RAM необходимо указать объем подключенной внешней памяти (рис. 2.26).

## Разработка программного файла

Активируем закладку программного файла prog2.asm (prog2.c), щелкнув по ней ЛК мыши либо выбрав программный файл в окне разработки.

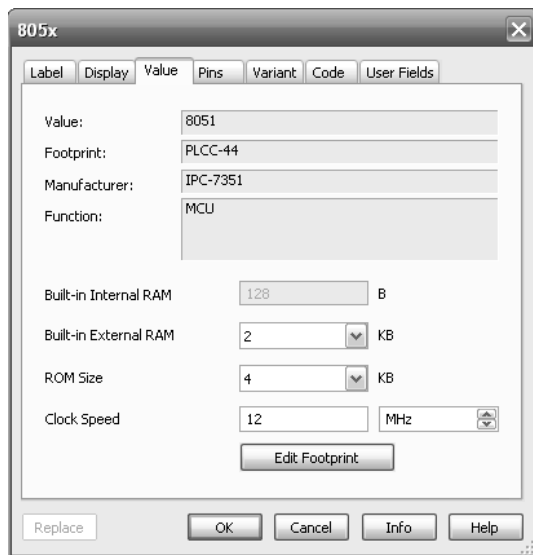


Рис. 2.26. Окно свойств МК, вкладка Value

Если имеется готовая программа, разработка которой выполнена, например, в среде PRO View [1], то вставляем полученный ассемблерный (или C) программный код и сохраняем его.

Покажем программу **тестирования внешней памяти** на следующем примере: требуется проверить 255 байт внешней памяти, начиная с ячейки 00h, используя тестовый набор 055h.

Ассемблерный файл программы:



```
$MOD51      ;подключение МК-51
org 00h     ;начинаем программу с адреса 00h
mov dptr,#00h ;загрузить в регистр-указатель число 00h
mov r2,#0ffh ;загрузить в регистр R2 число 0ffh (счетчик цикла)
mov r1,#055h ;загрузить в регистр R1 число 55h
test:
mov a,r1    ;загрузить аккумулятор ACC операндом из регистра R1
```


```


movx @dptr,a ;переслать в ячейку внешней памяти XRAM содержи-
               мое аккумулятора ACC
movx a,@dptr  ;читать в аккумулятор содержимое текущей ячейки
               внешней памяти
xrl a,#055h   ;операция XOR считанного и изначального операнда,
               если 0 в Акк, то ячейка работает нормально
jnz error     ;ошибка – выход из программы
inc dptr      ;инкремент содержимого регистра DPTR – переход
               к следующему адресу
djnz r2,test  ;вычесть 1 из содержимого регистра R2 и перейти по
               метке, если в ячейке не 0
error:
END


```


Выбираем в меню MCU – MCU 8051 U1, пункт Build, который позволяет откомпилировать и отредактировать программный файл. В окне сообщений на экране отражается информация об ошибках и предупреждениях. Если имеются ошибки, в окне сообщений указаны номера строк, в которых они находятся. Для отображения нумерации строк необходимо выбрать меню MCU – Show lines numbers.

Моделирование работы схемы производится через меню Simulate – Run, либо горячую клавишу F5 или иконку на панели Моделирование – . При этом открывается окно отладчика. Для пошагового моделирования используется пункт Step into в меню MCU, горячая клавиша F11 или иконка на панели Моделирование .

Кнопка Pause  или горячая клавиша F6 инициируют паузу моделирования, при нажатии которой возможно посмотреть, на каком этапе находится симуляция.

Кнопка Stop  останавливает выполнение моделирования.

Просмотр памяти микроконтроллера активируется через меню MCU – MCU 8051 U1 – Memory view, также можно использовать пункт MCU Windows, где во всплывающем окне можно отметить галочкой необходимые для работы окна атрибуты. При необходимости выполнения программы до определенной инструкции в Multisim предусмотрены точки останова, установить которые возможно через меню MCU – Toggle breakpoint или через панель Моделирование – .

Кнопка Remove all breakpoints  удаляет все точки останова.

После выполнения программой инструкции `djnz r2, test` посмотрим результаты записи во внешнюю память (рис. 2.27). В АСС записано число `00h`, в `R1` – `55h`, в `R2` – `FEh`, в первой ячейке внешней памяти `XRAM` – `55h`.

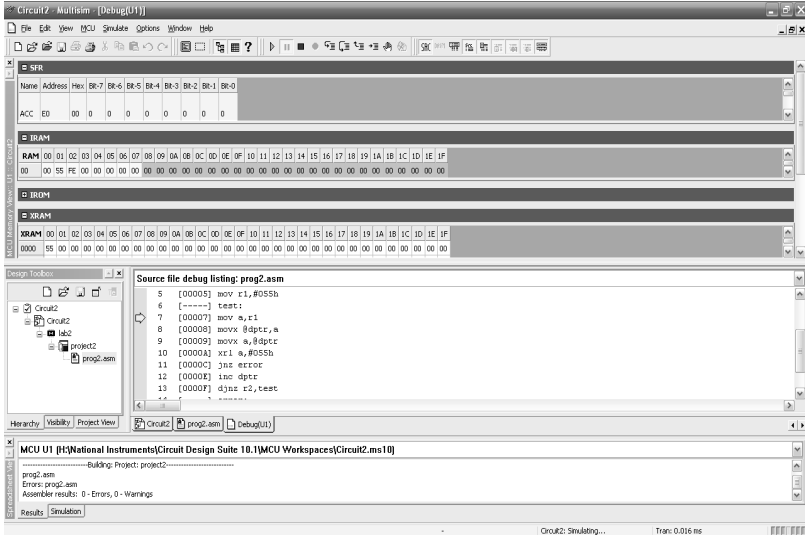


Рис. 2.27. Результаты моделирования после инструкции `djnz r2, test`

После завершения моделирования в каждой ячейке внешней памяти будет записано число `55h`, регистр `R2` будет иметь нулевое значение.

Работа с `C`-файлом аналогична. Ниже рассматривается вариант тестирования внешней памяти с адресами от `400h` до `7FFh`.

```
#include <8051.h> //подключаем заголовочные файлы
void main() //главная функция, точка входа в программу
{
    int i; //объявляем переменные
    //переменная ptr – указатель на адрес ячейки памяти внешнего ОЗУ
    char xdata *ptr;
    //символьные переменные test, nabor (байтовые переменные)
    char test, nabor;
    nabor=0x55; //тестовый набор
```

```

ptr= (char xdata *) 0x400; //начальный адрес внешней памяти
for(i=0; i<1024;i++)      //будут проверены 1024 ячейки памяти
{
//в каждую ячейку памяти посылается значение тестового набора
*ptr=nabor;
//переменная test принимает значение содержимого ячейки памяти
test=*ptr;
//если test ≠ nabor, то конец цикла (ошибка), иначе проверяется сле-
дующая ячейка памяти;
if(test!=nabor)
{
break;
}
ptr++;
}
}

```

Из сравнения программных кодов видно, что тестирование больших массивов данных (> 255 байт) легче организовать на С, чем на ассемблере, так как в ассемблерной программе потребуется организация двойных циклов.

В заключение лабораторной работы научимся пользоваться виртуальным осциллографом. Произведем подключение осциллографа к данной схеме. Выбираем на панели инструментов иконку Oscilloscope и устанавливаем на рабочей области. Символ осциллографа имеет два канала: Channel A и Channel B. Подсоединяем «+» канала А к выводу ALE/PROG микроконтроллера, «-» – к земле. На панели прибора необходимо указать деление временной шкалы, равное одному машинному циклу, то есть 1 мкс. Выбираем по оси у цену деления для канала А, примем ее равной 5 В. После запуска моделирования виден процесс генерации сигнала ALE, дважды за машинный цикл (рис. 2.28).



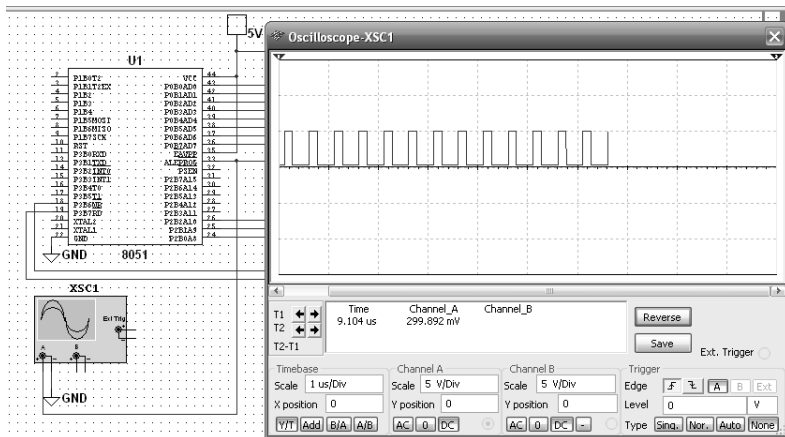


Рис. 2.28. Подключение осциллографа и его инструментальная панель во время моделирования

### 2.2.3. Задания для лабораторной работы

1. Подключить к МК внешнее ОЗУ емкостью N кбайт (табл. 2.9) и используя тестовый набор XX, произвести тестирование области памяти 1 кбайт, начиная с адреса ZZZ. Включить светоиндикатор, если записанный и считанный из ячейки памяти наборы не совпадают. Подключить осциллограф, как показано на рис. 2.28, и снять осциллограмму с вывода 0 ALE МК.

Таблица 2.9

Задание 1

Параметры	1	2	3	4	5	6
N, кбайт	8	2	8	8	8	2
XX	0AAh	55h	00h	0FFh	55h	0AAh
ZZZ	800h	100h	0C00h	1000h	1400h	400h

2. Подключить к МК внешнее ПЗУ емкостью N Кбайт (табл. 2.10). Подсчитать контрольную сумму области памяти емкостью Kбайт, начиная с адреса ZZZ и сравнить ее с константой, находящейся в резидентном ПЗУ. Если сравниваемые величины совпадают, тестирование ПЗУ прошло успешно, иначе включить светоиндикатор. Подключить осциллограф, как показано на рис. 2.28, и снять осциллограмму с вывода ALE МК.

## Задание 2

Параметры	7	8	9	10	11	12
<i>N</i> , Кбайт	8	16	32	8	16	32
<i>K</i> , байт	100	300	500	150	350	650
<i>ZZZ</i>	800h	1000h	2000	0C00h	1000h	400h

**2.2.4. Содержание отчета**

1. Наименование и цель работы.
2. Описание микросхемы памяти с назначением выводов и осциллографа, использованного в эксперименте, с его краткими характеристиками.
3. Копия схемного файла во время моделирования с указанием позиционных обозначений элементов и шин.
4. Копия программного файла (на ассемблере или на С) с подробными комментариями.
5. Полученные результаты и выводы по работе.

**2.2.5. Вопросы для самоконтроля**

1. Какие функции выполняет регистр-защелка при подключении к МК внешней памяти?
2. Назовите альтернативную функцию, выполняемую портом P2.
3. Объясните назначение сигнала PSEN (37 вывод МК-51).
4. Поясните, в чем отличия в работе процессора при обращении к внешней памяти данных по сравнению с резидентной?
5. Каково назначение вывода EA/Vpp МК?
6. Каким образом можно использовать внешнее ОЗУ для обращения и к данным и к программе?
7. Как подсчитывается контрольная сумма заданной области памяти?
8. Объясните назначение выводов CE (CS), OE микросхем памяти.
9. Каковы конструктивно-технологические отличия однократно-программируемых микросхем ПЗУ (PROM) по сравнению с репрограммируемыми ПЗУ (EPROM – с ультрафиолетовым стиранием)?
10. Поясните особенности тестирования микросхем ОЗУ по сравнению с ПЗУ.

## 2.3. Лабораторная работа № 3. Организация заданных интервалов времени

**Цель работы:** на основе встроенных таймеров МК-51 научиться реализовывать требуемые временные интервалы.

### 2.3.1. Основы настройки и использования таймеров МК-51

В состав МК-51 входят два 16-разрядных таймера/счетчика T/C0, T/C1. Состояние таймеров/счетчиков отображается в программно-доступных регистрах (TH0, TL0), (TH1, TL1), которые размещены в пространстве SFR по адресам (8Ch, 8Ah), (8Dh, 8Bh).

Таймеры/счетчики T/C0 и T/C1 могут быть запрограммированы для работы либо в качестве таймера, либо в качестве счетчика. Функция таймера состоит в счете числа машинных циклов, следующих с частотой  $F_{OSC}/12$ . Функция счетчика заключается в отслеживании числа переходов из 1 в 0 на соответствующих входах T0, T1.

Управление режимами работы устройств T/C0, T/C1 осуществляется регистром TMOD (Timer/Counter Mode), который расположен по адресу 89h, к нему возможно обращение только байтом данных (рис. 2.29).

7	6	5	4	3	2	1	0
GATE	C/T	M1	M0	GATE	C/T	M1	M0

Рис. 2.29. Структура регистра TMOD

Регистр разбит на два 4-разрядных подрегистра T0MOD и T1MOD, которые ответственны за управление T/C0 (биты 0–3) и T/C1 (биты 4–7):

TMOD.0 – M0. Младший бит поля управления режимом T/C0.

TMOD.1 – M1. Старший бит поля управления режимом T/C0.

При этом режим таймера T/C0 программируется в соответствии со значениями M0 и M1 следующим образом:

M0	M1	Режим	M0	M1	Режим
0	0	0	0	1	2
1	0	1	1	1	3

TMOD.2 – C/T. Выбор функции таймера или счетчика T/C0.

При C/T = 0 выбирается функция таймера, иначе – счетчика.

TMOD.3 – GATE. Флажок управления работой СТО. При GATE = = 1 работа разрешается, если вход INT0 = 1 и бит TR0 = 1 (см. рис. 2.30). При GATE = 0 работа счетчика зависит только от состояния TR0.

TMOD.4 – M0. То же, но для T/C1.

TMOD.5 – M1. То же, но для T/C1.

TMOD.6 – C/T. То же, но для T/C1.

TMOD.7 – GATE. То же, но для T/C1.

За управление таймерами T/C0, T/C1 также ответственен регистр TCON (Timer/Counter Control), расположенный по адресу 88h, который может управляться побитно (рис. 2.30).

F1	R1	F0	R0	E1	T1	E0	T0
----	----	----	----	----	----	----	----

Рис. 2.30. Структура регистра TCON

Младшая половина регистра используется для управления входами запроса на прерывания INT0 и INT1, старшая – для управления непосредственно таймерами T/C0, T/C1:

TCON.0 – IT0. Управление типом входа запроса на прерывание INT0. При IT0 = 1 программируется динамический по срезу тип входа, в противном случае – статический.

TCON.1 – IE0. Флажок запроса прерывания INT0 при динамическом входе. При подтверждении прерывания сбрасывается.

TCON.2 – IT1. То же, что и IT0, но для входа INT1.

TCON.3 – IE1. То же, что и IE0, но для INT1.

TCON.4 – TR0. Флажок программного запуска/останова T/C0.

TCON.5 – TF0. Флажок переполнения T/C0, который вызывает запрос прерывания. При подтверждении прерывания сбрасывается.

TCON.6 – TR1. То же, что и TR0, но для T/C1.

TCON.7 – TF1. То же, что и TF0, но для T/C1.

МК-51 имеет четыре режима работы встроенных таймеров, определяемые установкой соответствующих битов регистра TMOD. Режимы работы таймеров 0, 1, 2 совпадают для обоих таймеров. Установка в режим 3 таймера-счетчика T/C0 влияет на режим работы T/C1.

Рассмотрим режимы работы таймеров-счетчиков.

### ***Режим 0***

Таймеры / счетчики являются устройствами на базе 13-разрядных регистров, образованных соответствующими регистрами ТНх (х равно 0 или 1) и 5-ю младшими разрядами регистров ТЛх (три старших разряда ТL0 и ТL1 являются незначащими). Регистры ТЛх выполняют в таймерах функцию делителя на 32. Таймеры начинают считать при установке бита TRх регистра TCON в состояние «1». Установка в «1» бита GATEх регистра TMOD разрешает управление таймерами/счетчиками извне (по входам INT0, INT1). Установка в «0» бита TRх регистра TCON запрещает счет независимо от состояния других битов. Бит C/Tх определяет работу Т/С как таймеров («0»), так и счетчиков («1»).

### ***Режим 1***

Данный режим отличается от предыдущего только тем, что в счетчике используется 16-разрядный, а не 13-разрядный регистр.

### ***Режим 2***

В этом режиме используются 8-разрядные регистры ТL0 (в Т/С0) и ТL1 (в Т/С1). При переполнении этих регистров в процессе счета происходит перезагрузка их значением, заданным, регистрами ТН0 или ТН1. Регистры ТН0 (ТН1) загружаются программно, и процесс перезагрузки их содержимого в ТL0 (ТL1) не изменяет их значение.

### ***Режим 3***

В этом режиме работает только Т/С0. Счетчик Т/С1 заблокирован и сохраняет содержимое своих регистров ТL1 и ТН1 (как при TR1 = 0). Счетчик Т/С0 представлен двумя независимыми 8-разрядными регистрами ТН0 и ТL0. Устройство на базе ТН0 может работать только как таймер и использует некоторые управляющие биты и флаги Т/С1, например, при переполнении ТН0 происходит установка TF1, а для включения используется бит TR1. Остальные управляющие биты счетчика Т/С1 с работой таймера ТН0 не связаны. Установка Т/С0 в режим работы 3 приводит к лишению Т/С1 управляющего бита TR1. По этой причине при настройке устройства Т/С0 в 3 режим, устройство Т/С1, работающее в режимах 0, 1, 2 и при GATE1 = 0, всегда включено. При переполнении в режимах 0 и 1 таймер Т/С1 обнуляется, а во 2-м режиме – перезагружается, не устанавливая флаг TR1. Этот режим работы Т/С0 может быть использо-

ван в тех случаях, когда практически требуется работа трех независимых каналов счета.

Счетчики/таймеры T/C0, T/C1 являются внутренними источниками прерываний и используют для выработки запросов прерываний флаги переполнения TF0, TF1, представленные в регистре управления TCON. Обработка прерываний (вектора прерывания) для T/C0 и T/C1 начинается с адресов 000Bh и 001Bh соответственно.

### 2.3.2. Порядок выполнения лабораторной работы

Рассмотрим для примера следующую задачу: требуется подключить светодиоды к линиям одного из портов (P0–P3) и обеспечить их попеременное зажигание в течение заданного интервала времени. При выполнении работы проведем измерение электрических параметров светодиодов при помощи мультиметра.

Создаем схемный проект Circuit 3, размещаем на рабочем поле микроконтроллер МК-51, 8 резисторов по 270 Ом (для организации необходимого для зажигания диодов тока), 8 светодиодов, землю и питание. Можно для подключения реализовать шину Bus1. Подсоединяем 8 светодиодов к линиям порта P1 МК-51 и будем по очереди поддерживать каждый из них во включенном состоянии в течение 10 с, а затем выключать (рис. 2.31).

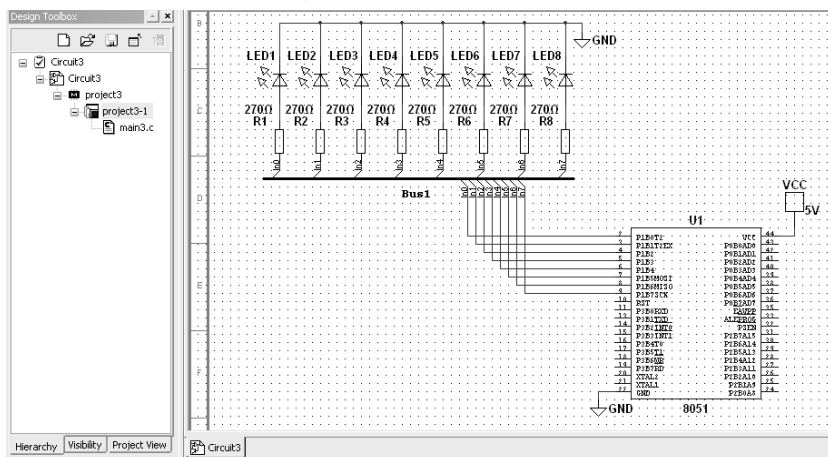


Рис. 2.31. Схема для проведения лабораторной работы № 3

В свойствах диода на вкладке Value в поле On Current (Ion) необходимо указать действующее значение тока 0,1 мА (рис. 2.32).

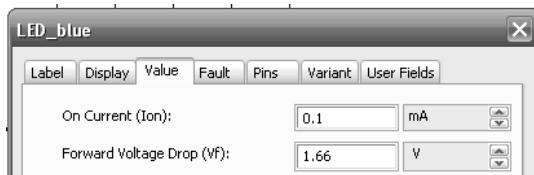


Рис. 2.32. Свойства диода, вкладка Value

Рассмотрим **ассемблерный файл программы**.

Используем первый режим работы T/C0, настроим его на счет машинных циклов (режим таймера) и организуем прерывание устройства на частоте кварца 12 МГц через каждые 50 мс (инкремент самого таймера будет происходить каждую микросекунду). Для этого таймер должен считать до 50000, переходить в режим прерывания и обнуляться.

```

$MOD51           ;подключение МК-51
jmp start        ;переход на метку start
org 0bh          ;подпрограмма прерывания по переполнению
                  ;таймера T0
clr tcon.4       ;запрещаем счет таймера
reti            ;команда возврата из прерывания
org 20h
start:           ;основная программа
clr c            ;сброс переноса
mov tmod,#01h   ;настройка T/C0 на работу в первом режиме
                  ;в качестве таймера
setb ie.7       ;общее разрешение прерывания
setb ie.1       ;разрешаем прерывание по таймеру T/C0
mov p1,#0h      ;настраиваем порт P1 на вывод информации
met1:
mov a,1h        ;заносим в аккумулятор значение 1h (для вклю-
                  ;чения диода LED1)
mov r0,#0C8h    ;r0 – счетчик цикла
;заносим число 200 в регистр r0 (200 раз по 50 мс, чтобы было 10с)
met:
mov TL0, #low(not(50000-1))
;помещаем в регистр TL0 младший байт числа 50000, считаем
от -50000 до 0
mov TH0, #high(not(50000-1))

```

```

;помещаем в регистр TH0 старший байт числа 50000, считаем
от -50000 до 0
mov p1,a          ;включаем первый светодиод
setb tcon.4       ;включаем таймер T/C0
next:
jnb tcon.5, next  ;ждем флаг переполнения TF0
djnz r0,met       ;организуем цикл 200 раз (10 с)
rlc a             ;по истечении 10 с сдвигаем содержимое Acc
                  ;и включаем следующий диод
mov r0,0C8h       ;снова загружаем счетчик цикла r0
jnc met           ;если не carry (не 8 диод горит), идем на met
clr c             ;сброс переноса
jmp met1          ;бесконечный цикл
end

```

**Рассмотрим программный код на С**, изменяя условия работы светодиодов.

Допустим, каждый диод горит по 40 с, затем после попеременного включения 8 диодов идет пауза в 20 с.

Программный код включает основную программу и подпрограмму msec, в которой реализуется получение периода времени 10 мс. Параметром подпрограммы msec является коэффициент деления требуемого временного интервала на 10 мс.

```

#include <8051.h>
//подпрограмма получения требуемого временного интервала; x – ко-
эффицент деления временного интервала на 10 мс
void msec (int x)
{
while(x-- >0)          //декремент x, до тех пор, пока x > 0
{
TH0 = (-10000)>>8     //настройка TH0 на значение старшего
                      ;байта (-10000)
TL0 = -10000;         //настройка TL0 на значение младшего
                      ;байта (-10000)
TR0 = 1;              //включить таймер
do; while(TF0==0);    //ожидание 10 мс
TF0=0;                //очистить флаг переполнения
TR0=0;                //выключить таймер

```



```

}
}
void main()                //основная программа
{
int i;
unsigned char array[9];
TMOD=0x1;                 //настройка TCO на первый режим работы
array[0]=0x0;             //значение массива для паузы
//значения массива для поочередного зажигания восьми диодов
array[1]=0x1;
array[2]=0x2;
array[3]=0x4;
array[4]=0x8;
array[5]=0x10;
array[6]=0x20;
array[7]=0x40;
array[8]=0x80;
//организовать паузу 20 мс
P1=array[0];
msec(2);
for (i=1; i<9; i++)       //организовать цикл от 1 до 8
{
P1=array[i];              //поочередно включать диоды
msec(4);                  //удерживать свечение по 40 мс
}
while (1);
}

```

Рассмотрим включение в исследуемую схему мультиметра. Подключим прибор к линиям порта P1. Для этого выбираем на поле измерительных приборов иконку мультиметра и устанавливаем на рабочую область. Изображение прибора содержит выводы, подключаемые к исследуемому компоненту схемы. Если ток «втекает» в вывод «+», то получаются положительные значения измеренного тока, при другом подключении – отрицательное значение тока (рис. 2.33).

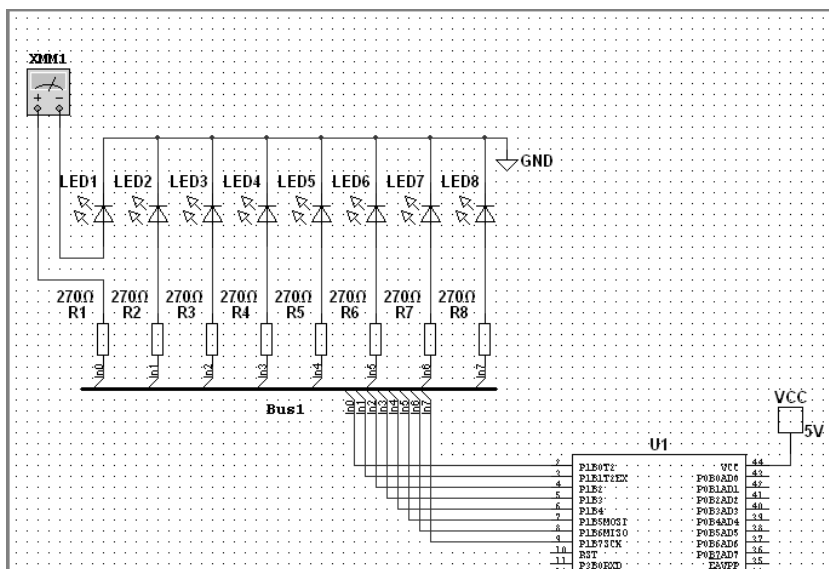


Рис. 2.33. Пример подключения мультиметра к схеме для измерения тока

С помощью мультиметра можно измерять токи, напряжения, сопротивления электронных компонентов – в полной аналогии с реальным физическим прибором, у которого на лицевой панели тоже есть такие переключатели (рис. 2.34).

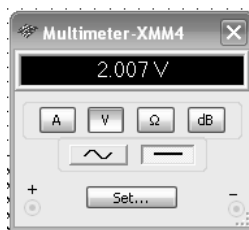


Рис. 2.34. Передняя панель мультиметра

Передняя панель мультиметра содержит еще один переключатель – режим измерения помех в децибелах, а также кнопки для переключения измерений по постоянному и переменному току.

Кнопка  на панели позволяет отобразить параметры мультиметра как прибора в отдельном окне (рис. 2.35).

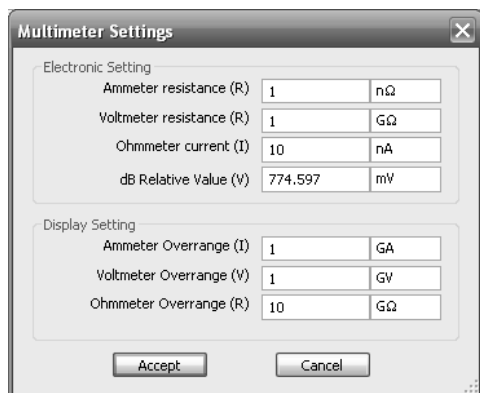


Рис. 2.35. Окно параметров мультиметра

В окне свойств прибора можно видеть следующее:

- в группе Electronic Setting показано, что он имеет внутреннее сопротивление, равное 1нОм, если он измеряет электрический ток, имеет внутреннее сопротивление 1 ГОм, если измеряет напряжение, и для измерения сопротивления прибор формирует ток 10 нА через подключенный элемент схемы, который затем используется для вычисления сопротивления;
- в группе Display Setting приводятся значения, определяющие условия индикации ошибки при измерении. Так, если, например, измеряемое напряжение превысит значение Voltmeter Over range, то программой выдается сообщение об ошибке (рис. 2.36).



Рис. 2.36. Сообщение об ошибке на передней панели мультиметра

Все указанные выше параметры могут быть изменены при настройке работы прибора в схеме: если, например, последовательное или параллельное сопротивление (измеритель тока или напряжения) будут влиять на работу схемы, то их можно изменить на новые значения.

При запуске режима моделирования МК начинает выполнять программу, а на экране мультиметра появятся значения измеряемого тока или напряжения (рис. 2.37).

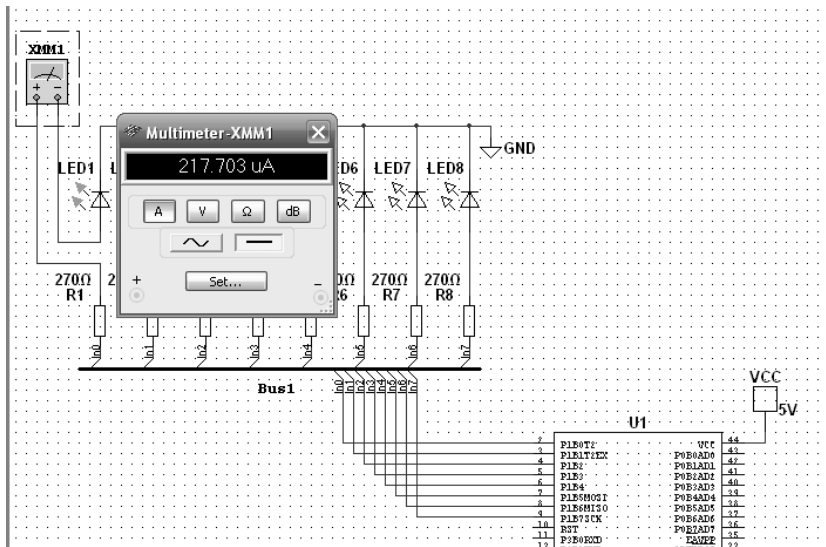


Рис. 2.37. Показания мультиметра при зажигании первого светодиода

### 2.3.3. Задания для лабораторной работы

Требуется подключить светодиод к одной из линии порта P<sub>X</sub>.Y (X – номер порта, Y– номер вывода порта) и обеспечить загорание светодиода в течение S с, затем выключение светодиода в течение K с; организовать попеременное включение/выключение светодиода в течение T мин. Произвести подключение мультиметра к светодиоду и измерить его ток и/или напряжение во время работы. Варианты заданий приведены в табл. 2.11.

Таблица 2.11

Варианты заданий

Параметры	1	2	3	4	5	6	7	8	9	10
P <sub>X</sub> .Y	P1.3	P1.7	P2.5	P2.1	P3.5	P1.6	P3.2	P1.4	P2.6	P3.3
S, с	2	3	1	0,5	4	6	0,7	5	6	7
K, с	1	2	0,5	1	3	4	0,3	4	5	6
T, мин	3	5	2	1	8	10	2	1	5	15

### **2.3.4. Содержание отчета**

1. Наименование и цель работы.
2. Описание схемных особенностей линии порта МК, используемой для подключения светодиода и мультиметра, применяемого в эксперименте, с его краткими характеристиками.
3. Описание особенностей программирования таймера МК, используемого для реализации временного интервала.
4. Копия схемного файла во время моделирования с указанием позиционных обозначений элементов.
5. Копия программного файла (на ассемблере или на С) с подробными комментариями.
6. Полученные результаты и выводы по работе.

### **2.3.5. Вопросы для самоконтроля**

1. Поясните, каким образом в базовой версии МК-51 можно организовать работу трех независимых таймеров?
2. Чем отличается работа таймера МК 51 от работы счетчика?
3. Поясните назначение разрядов регистра TMOD.
4. Как организовать включение таймеров/счетчиков, зависящее от появления определенного внешнего события?
5. Какие значения сопротивления нагрузки должны иметь измерительные приборы (вольтметры, амперметры), чтобы не исказить измеряемые величины напряжения и тока?
6. В чем отличие программно-управляемого режима работы таймера от режима работы по прерываниям?
7. Поясните назначение битов регистра TCON.
8. Поясните понятие вектора прерывания. Какие вектора прерывания имеют TC0 и TC1?
9. Поясните назначение регистра IE – маски прерываний.
10. Можно ли изменить приоритет прерывания источников запроса? Какие приоритеты (по умолчанию) имеют TC0 и TC1?

## 2.4. Лабораторная работа № 4.

### Основы организации последовательного порта

**Цель работы:** научиться использовать последовательный порт МК для различных применений.

#### 2.4.1. Основные сведения о режимах работы последовательного порта

В структуру МК-51 входит дуплексный канал последовательной связи с буферизацией, который может быть запрограммирован для работы в одном из четырех режимов:

а) режим 0 – синхронный, последовательный ввод-вывод со скоростью  $f_{\text{осц}}/12$ ;

б) режим 1 – асинхронный с 10-битовым кадром и переменной скоростью передачи, зависящей от частоты переполнения таймера / счетчика  $1 - T/C1$ ;

в) режим 2 – асинхронный с 11-битовым кадром и фиксированной скоростью передачи  $f_{\text{осц}}/32$  или  $f_{\text{осц}}/64$ ;

г) режим 3 – асинхронный с 11-битовым кадром и переменной скоростью передачи, также определяемой частотой переполнения  $T/C1$ .

Принятые входные и передаваемые выходные данные в параллельном коде хранятся в буферном регистре SBUF, который располагается в пространстве SFR по адресу 99h. Управление работой приемопередатчиков осуществляется через слово управления и состояния SCON, расположенное в регистре по адресу 98h, имеющем следующую структуру (рис. 2.38):

SM0	SM1	SM2	REN	TB8	RB8	TI	RI
-----	-----	-----	-----	-----	-----	----	----

Рис. 2.38. Структура регистра SCON

SCON.0 – RI. Флаг прерывания приемника.

SCON.1 – TI. Флаг прерывания передатчика.

SCON.2 – RB8. Восьмой бит приемника в режимах 2 и 3. В режиме 1, если  $SM2 = 0$ , то отображает стоп-бит. В режиме 0 не используется.

SCON.3 – TB8. Восьмой бит передатчика в режимах 2 и 3.

SCON.4 – REN. Разрешение приема.

SCON.5 – SM2. Запрещение приема кадров с нулевым восьмым битом данных. В режиме 0 должен быть сброшен.

SCON.6 – SM1. Младший разряд для кодирования номера режима.

SCON.7 – SM0. Старший разряд для кодирования номера режима.

Режим работы последовательного порта определяется следующим образом:

SM0	SM1	Режим	SM0	SM1	Режим
0	0	0	1	0	2
0	1	1	1	1	3

Биты SCON.0 – SCON.2 устанавливаются аппаратно, а сбрасываются программно, биты SCON.3–SCON.7 устанавливаются и сбрасываются программно.

В режиме 0 работы последовательного порта для синхронизации внешних устройств используется линия TxD (P3.0), по которой передаются синхроимпульсы, а прием и передача информации осуществляется по линии RxD (P3.1).

Скорость приема/передачи в режиме 0 определяется как

$$f_0 = f_{\text{осц}}/12,$$

где  $f_{\text{осц}}$  – частота кварцевого резонатора.

За один машинный цикл МК последовательный порт передает один бит информации.

Скорость приема/передачи последовательного порта в режиме 2 определяется как

$$f_2 = (2^{\text{SMOD}}/64)f_{\text{осц}},$$

где бит SMOD является 7 битом регистра PCON (рис. 2.39).

Регистр PCON располагается в пространстве SFR по адресу 87h и полностью реализован в микросхемах КМОП технологии для управления режимом энергопотребления.

SMOD	–	–	–	GF1	GF0	PD	IDL
------	---	---	---	-----	-----	----	-----

Рис. 2.39. Структура регистра PCON

Назначение бит регистра следующее:

PCON.0 – IDL. Бит холостого кода. При PCON.0 = 1 МК переходит в режим холостого хода.

PCON.1 – PD. Бит пониженной мощности. При PCON.1 = 1 МК переходит в режим пониженного потребления мощности.

PCON.2 – GF0. Флаг, специфицируемый пользователем.

PCON.3 – GF1. Флаг, специфицируемый пользователем.

- PCON.4. Не используется.  
 PCON.5. Не используется.  
 PCON.6. Не используется.  
 PCON.7 – SMOD. Удвоенная скорость работы последовательного порта, если SMOD = 1.

Скорость передачи в режимах 1 и 3 определяется не только битом SMOD, но и частотой переполнения таймера-счетчика T/C1. При настройке порта на эти режимы работы необходимо запретить прерывания по переполнению таймера T/C1.

$$f_{1,3} = (2^{\text{SMOD}}/32)f_{\text{OVT1}},$$

где  $f_{\text{OVT1}}$  – частота переполнения таймера-счетчика T/C1. Наиболее удобно при работе последовательного порта использовать второй режим работы T/C1 – режим 8-битного суммирующего счетчика с автоперезагрузкой. При этом частота передачи определяется выражением:

$$f_{1,3} = (2^{\text{SMOD}}/32) \cdot (f_{\text{осц}}/12)/(256 - (\text{TH1})),$$

где (TH1) – содержимое регистра TH1 таймера счетчика T/C1.

Параметры настройки T/C1 для управления частотой работы последовательного порта представлены в табл. 2.12.

Таблица 2.12

Настройки МК для управления частотой последовательного порта

Частота	$f_{\text{осц}}$ , МГц	SMOD	C/T1	Режим T/C1	TH1
Режим 0: 1 МГц	12	X*	X	X	X
Режим 2: 375 КГц	12	1	X	X	X
Режимы 1, 3: 62,5 КГц	12	1	0	2	0FFh
19,2 КГц	11,059	1	0	2	0FDh
9,6 КГц	11,059	0	0	2	0FDh
4,8 КГц	11,059	0	0	2	0FAh
2,4 КГц	11,059	0	0	2	0F4h
1,2 КГц	11,059	0	0	2	0F8h
137,5 Гц	11,059	0	0	2	1Dh
110 Гц	6	0	0	2	72h

\*Символ X обозначает безразличие в настройке соответствующего параметра.

Передача данных по последовательному порту инициируется всякий раз, когда новые данные заносятся в регистр SBUF, например



по команде *MOV SBUF, A*. Признаком окончания передачи служит установка флажка прерывания TI.

Операция приема данных активируется только при установленном бите  $REN = 1$ , когда флажок RI сброшен. Установка флажка RI свидетельствует о готовности данных для считывания из регистра SBUF, тогда может быть использована, например, команда *MOV A, SBUF*.

### 2.4.2. Порядок выполнения лабораторной работы

Создаем схемный проект Circuit4, устанавливаем на рабочей области микроконтроллер МК-51. К выводу P3.0 подключаем осциллограф (рис. 2.40) для снятия сигнала при передаче информации.

Рассмотрим, например, ассемблерную программу управления последовательным портом в режиме передачи. Допустим, необходимо передать символы А, В, С со скоростью 4800 бит/с в асинхронном режиме 11-битным кадром (3 режим работы). При написании программы будем использовать режим прерываний последовательного порта при передаче символа. Передаваемые данные размещаются в программном коде, начиная с адреса 30h.

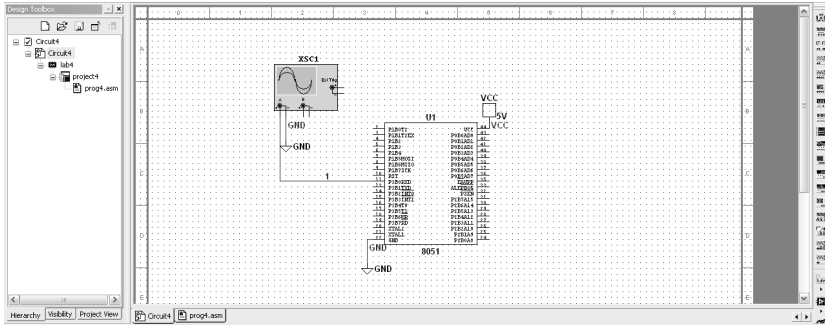


Рис. 2.40. Схема моделирования работы последовательного порта

Активируем вкладку программного файла prog4.asm и вставим следующий **ассемблерный код**:

```

$MOD51          ;подключение МК-51
org 00h
jmp start
org 23h         ;адрес начала процедуры прерываний по последовательному порту

clr TI         ;при входе в подпрограмму прерываний обнуляем флаг TI

inc r1        ;инкрементируем r1 – счетчик количества переданных символов

mov a,r1      ;пересылаем значение r1 в аккумулятор
subb a,#3h   ;проверка значения r1, чтобы не превысило трех символов

jnz metka    ;если не превышает, то идем на metka
mov r1,#00h  ;иначе обнуляем r1
metka:
reti        ;возврат из прерывания
org 30h
tabl:
db 61h      ;таблица данных для передачи
db 62h
db 63h
org 40h
start:
clr c       ;обнуляем бит carry
mov scon,#0C0h ;настраиваем порт на работу в режиме 3
mov tmod,#20h ;устанавливаем счетчик T/C1 во второй режим работы
mov IE,#90h ;разрешаем общее прерывание и прерывания от порта
setb IP.4  ;максимальный приоритет последовательного порта
mov tl1,#0FAh ;заносим в таймер число FAh, чтобы настроить порт
mov th1,#0FAh ;скорость передачи 4800 бит/с
setb TR1   ;включаем T/C1
mov r1, #0h
mov r0, #03h ;число байт, подлежащих отправке
mov dptr, #tabl ;заносим в dptr начальный адрес таблицы данных
met1:

```

```

mov a,r1
movc a, @a + dptr
mov c, PSW.0 ;установить бит четности при передаче кадра
mov TB8, c ;установить бит TB8 , равным биту четности
mov sbuf,a ;отправляем символ таблицы на передачу
met:
jnb TI,met ;ждем прерывание
djnz r0, met1 ;организация цикла передачи
END

```

Рассмотрим *пример проектирования С-программы* для передачи строки «privet iz Ekaterinburga» из последовательного порта при его работе во 2 режиме (11-битовый кадр, фиксированная скорость передачи 375 Кбит/с).

```

#include <8051.h>
//функция отправки символа по последовательному порту
void tput(unsigned char c1)
{
SBUF=c1; ;вносим символ в буфер передачи
while(!TI); ;ждем окончания передачи
TI=0; ;сбрасываем флаг окончания передачи
}
void main()
{
char z;
int i;
//объявляем и инициализируем передаваемую строку
unsigned char src[ ]={"privet iz Ekaterinburga"};
//устанавливаем бит SMOD в 1, для того чтобы скорость приема / передачи равнялась 1/32 частоты кварцевого резонатора
PCON=0x80;
for(i=0; i<24; i++)
{ ACC=src[i]; ;вносим текущий символ в аккумулятор
// число единичных бит в аккумуляторе нечетное?
if(!(PSW&&0x01))
{
//включаем асинхронный 9-ти битовый режим работы последовательного порта, бит четности сброшен
z = 0x80;

```

```

}
else
{
//включаем асинхронный 9-ти битовый режим работы последовательного порта, бит четности установлен
z = 0x88;
}
//вносим значение в регистр управления режимом приемопередатчика
SCON = z;
//передаем текущий символ в функцию отправки
trput (src[i]);
}
while(1){}          //бесконечный цикл
}

```

### 2.4.3. Задания для лабораторной работы

Допустим, необходимо принять/передать (R/T) N байт информации, настроив последовательный порт на K-режим работы со скоростью обмена S Кбит/с. Требуемые для передачи байты находятся в резидентной памяти данных, начиная с адреса XX.

В режиме приема XX – начальный адрес резидентной памяти данных, где размещаются принятые байты. При моделировании приема будем использовать два микроконтроллера МК-51, один из которых передает информацию в заданном режиме работы, а другой принимает.

Варианты заданий на лабораторную работу представлены в табл. 2.13.

Таблица 2.13

Варианты задания

Параметры	1	2	3	4	5	6	7	8	9	10
R/T	T	T	R	T	R	R	T	T	R	T
K	0	1	2	3	0	1	2	3	1	1
S, Кбит/с	1000	19,2	375	2,4	1000	4,8	187,5	62,2	9,6	1,2
XX	50h	30h	40h	50h	40h	60h	30h	40h	30h	50h
N	10	6	15	8	20	10	20	15	20	10

#### **2.4.4. Содержание отчета**

1. Наименование и цель работы.
2. Описание особенностей работы последовательного порта МК в режиме, используемом в задании.
3. Копия схемного файла во время моделирования с указанием позиционных обозначений элементов.
4. Копия программного файла (на ассемблере или на С) с подробными комментариями.
5. Полученные результаты и выводы по работе.

#### **2.4.5. Вопросы для самоконтроля**

1. Чем отличается синхронный протокол работы последовательного порта от асинхронного (старт-стопного)?
2. Может ли последовательный порт МК-51 одновременно передавать и принимать данные? Ответ обосновать.
3. Какие программируемые регистры в области памяти SFR имеет последовательный порт? Расскажите об их назначении и адресах.
4. Каково назначение бита контроля, используемого при приеме/передаче информации? В каких режимах работы последовательного порта он используется?
5. Как программно получить бит контроля при передаче информации 11-битным кадром?
6. Как проверить правильность приема символа 11-битным кадром?
7. В чем отличие программно-управляемого режима работы последовательного порта от режима прерывания?
8. Дайте понятие вектора прерывания. Какой вектор прерывания имеет последовательный порт?
9. Почему при приеме и передаче используется один и тот же вектор прерывания?
10. Какие средства имеет МК для разрешения/запрета прерываний от источников и изменения приоритета?

## 2.5. Лабораторная работа № 5. Отображение информации в системах с МК-51

**Цель работы:** научиться подключать к микроконтроллеру средства отображения информации для визуализации показаний.

В данной лабораторной работе в качестве средств визуализации рассматриваются семисегментные индикаторы. В базе данных Multi-sim также имеются 15-сегментные светоиндикаторы и многопозиционные дисплеи.

### 2.5.1. Общие сведения о семисегментных индикаторах

Семисегментные индикаторы удобны в управлении, имеют высокую яркость, широкий диапазон рабочих температур и низкую стоимость [6, 7].

Устройство состоит из семи светодиодов продолговатой формы, размещенных таким образом, чтобы, зажигая их в разных сочетаниях, можно было бы отобразить любую десятичную арабскую цифру от 0 до 9. Кроме семи основных сегментов, индикатор чаще всего дополняют восьмым маленьким сегментом, который предназначен для отображения десятичной точки (запятой). Если расположить в ряд несколько таких индикаторов, можно отображать любое десятичное число с плавающей запятой.

Внешний вид семисегментного индикатора приводится на рис. 2.41, где каждый сегмент индикатора обозначается буквой латинского алфавита. Такой индикатор обычно выполняется в виде отдельного самостоятельного компонента и имеет 9 выводов.

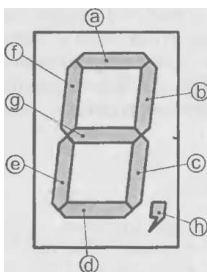


Рис. 2.41. Семисегментный цифровой индикатор

По внутренней схеме включения семисегментные индикаторы подразделяются на индикаторы с общим анодом (рис. 2.42) и индикаторы с общим катодом (рис. 2.43).

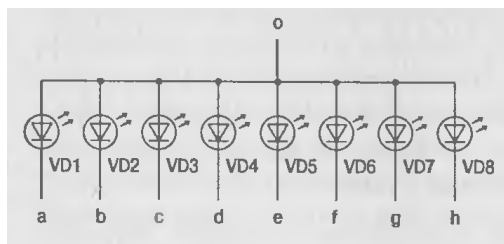


Рис. 2.42. Схема индикатора с общим анодом

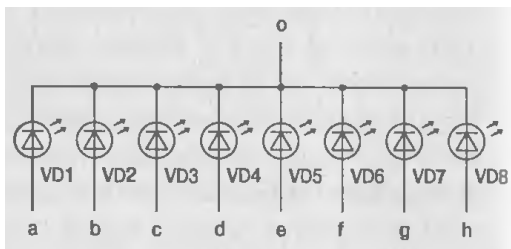


Рис. 2.43. Схема индикатора с общим катодом

В первом случае на общий вывод светодиодов подается плюс источника питания, а во втором – минус, в некоторых схемах семисегментный индикатор имеет 10 выводов (общий вывод дублируется).

Для подключения семисегментного индикатора повышенной яркости необходимо применять буферные элементы, например, регистры-защелки или преобразователь двоичного кода в код семисегментного индикатора.

Пример подключения индикатора с общим анодом к МК показан на рис. 2.44, где использовано непосредственное подключение устройства к выводам микроконтроллера.

Для преобразования цифрового кода в код семисегментного индикатора существует два способа.

1. *Программный вариант* (см. рис. 2.44). Для подключения одного индикатора используются все 8 линий порта. Необходимо программно задать таблицу включения цифровых кодов семисегментного индикатора, которая для схемы с общим анодом имеет следующий вид:

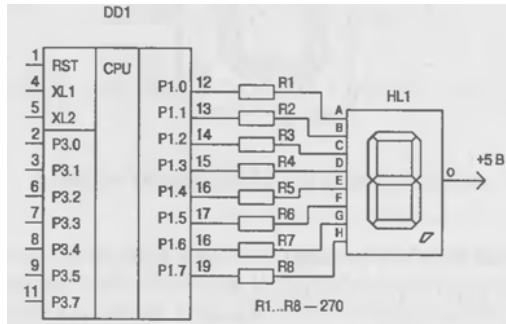


Рис. 2.44. Подключение семисегментного индикатора с общим анодом

```

db 11000000b      ; символ «0»
db 11111001b     ; символ «1»
db 10100100b     ; символ «2»
db 10110000b     ; символ «3»
db 10011001b     ; символ «4»
db 10010010b     ; символ «5»
db 10000010b     ; символ «6»
db 11111000b     ; символ «7»
db 10000000b     ; символ «8»
db 10010000b     ; символ «9»

```

Для светоиндикаторов с общим катодом коды цифр необходимо инвертировать.

2. *Аппаратный вариант* (рис. 2.45). Подключение светоиндикатора производится через преобразователь (дешифратор) двоичного кода в код семисегментного индикатора. При таком подключении используются всего 4 линии выходного порта.



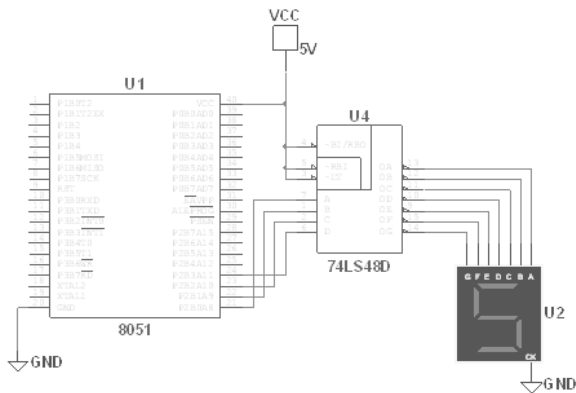


Рис. 2.45. Подключение семисегментного индикатора с общим катодом через дешифратор

### 2.5.2. Ход выполнения лабораторной работы

Рассмотрим два способа преобразования цифрового кода в код семисегментного индикатора на примере следующей задачи: требуется вывести на индикатор, подключенный к порту P1, цифры от 9 до 1.

При выполнении лабораторной работы можно воспользоваться любым из показанных ниже вариантов преобразования информации.

**Программный вариант.** Открываем и сохраняем новый схемный проект Circuit 5. Устанавливаем на рабочем поле МК-51 (создаем программный файл prog5.asm), семисегментный индикатор с общим анодом, 8 токоограничивающих резисторов по 270 Ом, землю и питание (рис. 2.46).

Активируем закладку программного файла, щелкнув по ней ЛК мыши либо выбрав программный файл в окне разработки. Затем помещаем в окно программного файла отлаженный, например, в среде PRO View ассемблерный (или Си) программный код и сохраняем его.

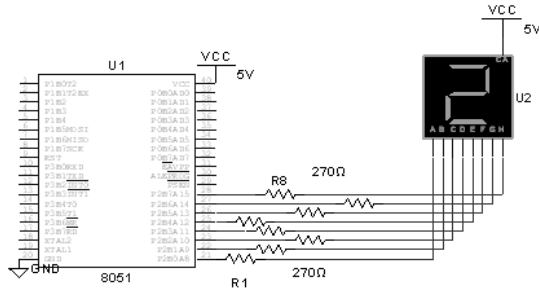


Рис. 2.46. Программный вариант преобразования информации в код индикатора

Приведем *код программы на ассемблере*:

```

$MOD51           ;подключение МК-51
org 20h          ;начинаем программу с адреса 20h
mov r2,#9       ;загрузить в регистр r2 число 9
ind:
mov a, r2       ;загрузить аккумулятор операндом из регистра r2
mov r3,#255     ;r3 – счетчик для реализации временной задерж-
ки   mov dptr,#zg ;загрузка начала таблицы знакогенератора
movc a,@a+dptr  ;считывание кода цифры в ACC
indl:
mov p2,a        ;вывод полученного кода в порт p2
nop            ;пустая операция
nop            ;пустая операция
djnz r3,indl    ;организация цикла для устойчивого отображе-
ния каждой цифры
djnz r2,ind     ;переход к отображению следующей цифры
                ;(от 9 до 0)
org 0100h       ;таблица кодов расположена с адреса 100h
zg:            ;знакогенератор
db 11000000b   ;символ «0»
db 11111001b   ;символ «1»
db 10100100b   ;символ «2»
db 10110000b   ;символ «3»
db 10011001b   ;символ «4»
db 10010010b   ;символ «5»
db 10000010b   ;символ «6»

```

```

db 11111000b      ;символ «7»
db 10000000b      ;символ «8»
db 10010000b      ;символ «9»
END

```

Ниже приводится **программа решения подобной задачи на С** для случая бесконечного вывода цифр в прямом порядке (от 0 до 9) и паузой после вывода цифры 9.

```

#include <8051.h>
void main()
{
unsigned char i,j;    // 8-битные переменные
unsigned char massiv [11]=
{
0xC0,                //массив кодов семисегментного индикатора
                        от 0 до 9
0xF9,
0xA4,
0xB0,
0x99,
0x92,
0x82,
0xF8,
0x80,
0x90,
0xFF                //код выключения
};
P1=massiv [10];      //сначала выключим индикатор
for(i=0;i<10;i++)    //затем выводим код в цикле в порт 2
{
P2=massiv[i];        //коды от 0 до 9
for(j=0;j<100;j++)
//временная задержка для устойчивого горения каждой цифры
continue;
}
P2=massiv[10];       //выключить индикатор
while(1);
}

```

Покажем *аппаратный вариант преобразования* числа в код индикатора. Открываем и сохраняем новый схемный проект Circuit 5–2. Устанавливаем на рабочем поле МК-51 (создаем программный файл prog5–2), семисегментный индикатор с общим катодом, преобразователь двоичного кода в семисегментный индикатор 74LS48D для светоиндикатора с общим катодом (данная модель дешифратора уже содержит подтягивающие сопротивления), землю и питание (см. рис. 2.45).

### Код программы на С

```
#include <8051.h>
void main()
{
    unsigned char i,j;    //8-битные переменные
    for(i=0;i<10;i++)    //затем выводим код в цикле в порт2
    {
        P2=i;           //коды от 0 до 9
        for(j=0;j<100;j++)
        //временная задержка для устойчивого горения каждой цифры
        continue;
    }
    while(1);
}
```

Для вывода байта (числа от 0 до 255) на семисегментные индикаторы может потребоваться включение в схему нескольких индикаторов. При этом необходимо использовать следующую *ассемблерную программу* преобразования байтового числа в коды BCD:

```
mov B,#100           ;загрузить в В число 100 для вычисления количества сотен в числе
div AB              ;аккумулятор содержит число сотен, т. е. старшую цифру
mov r0,A           ;пересылка в R0 старшей цифры
xch A,B            ;пересылка остатка исходного числа в аккумулятор
mov B,#10          ;загрузить в В число 10 для вычисления количества десятков в числе
div AB             ;аккумулятор содержит число десятков, В – число единиц
swap A            ;размещение числа десятков в старшей тетраде аккумулятора
add A,B           ;суммирование остатка (числа единиц), теперь аккумулятор содержит две младшие цифры
```

Для подключения нескольких семисегментных индикаторов к одному порту микроконтроллера используется прием, называемый «динамической индикацией» [6].

### 2.5.3. Задания для лабораторной работы

**Задание 1.** Байтовую переменную VAL вывести на трехразрядный семисегментный индикатор (использовать 3 индикатора, подключенных к одному из портов МК). Варианты заданий приведены в табл. 2.14.

Таблица 2.14

Варианты задания 1

Параметры	1	2	3	4	5	6
VAL	0FFh	0A7h	4Ch	210	100	150
Выходной порт	P1	P2	P3	P2	P3	P1

**Задание 2.** Сложить два числа a и b, представленных в двоично-десятичном коде (BCD). Результат вывести на двухразрядный семисегментный светоиндикатор, подключенный к одному из портов МК. Варианты заданий приведены в табл. 2.15.

Таблица 2.15

Варианты задания 2

Параметры	1	2	3	4	5	6
a	23	77	36	16	45	64
b	52	18	54	61	37	35
Выходной порт	P1	P2	P3	P3	P1	P2

### 2.5.4. Содержание отчета

1. Наименование и цель работы.
2. Описание особенностей работы семисегментного индикатора, используемого в схеме.
3. Копия схемного файла во время моделирования с указанием позиционных обозначений элементов.
4. Копия программного файла (на ассемблере или на C) с подробными комментариями.
5. Полученные результаты и выводы по работе.

### 2.5.5. Вопросы для самоконтроля

1. Расскажите о схемных особенностях использования порта P0 в качестве выходного.
2. Как настроить порт МК на прием информации?
3. Можно ли отдельные линии порта настроить на ввод, а другие на вывод?
4. В чем отличие при представлении байта кодом BCD по сравнению с двоичным кодом?
5. Каковы схемные особенности регистра-защелки информации? Покажите на примерах, имеющихся в БД Multisim.
6. Расскажите о схмотехнических отличиях, связанных с использованием программного преобразования двоичного числа в код семисегментного индикатора (десятичное число), по сравнению с аппаратным преобразованием.
7. Какие регистры МК по умолчанию используются для операций умножения и деления?
8. Как преобразовать байт в десятичные цифры, выводимые на семисегментные индикаторы?
9. Расскажите о схемных особенностях использования элементов с общим коллектором.

## Лабораторная работа № 6. Изучение принципов работы цифроаналоговых преобразователей

**Цель работы:** Ознакомление с особенностями работы интегрального цифроаналогового преобразователя.

### 2.6.1. Общие сведения о цифроаналоговом преобразовании

*Устройство, осуществляющее автоматическое преобразование входных значений, представленных числовыми кодами, в эквивалентные им значения какой-нибудь физической величины (напряжения, тока и др.), называют цифроаналоговым преобразователем (ЦАП).*

Основы классификации ЦАП представлены на рис. 2.47.

Подробная информация о ЦАП изложена, например, в [9]. Разделение на последовательные и параллельные устройства рассматривается с точки зрения параллельного или последовательного во времени преобразования сигналов.

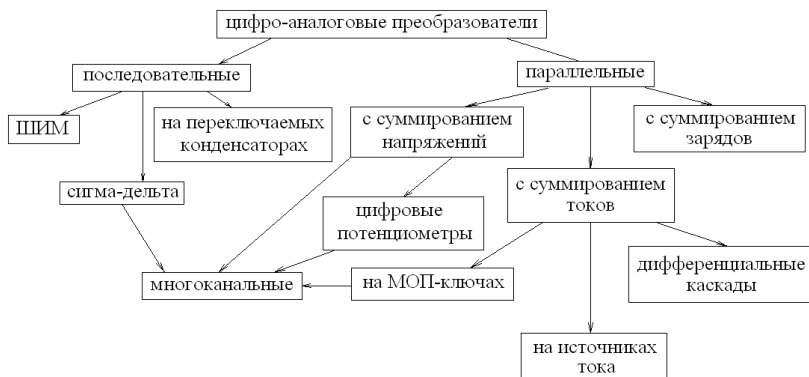


Рис. 2.48. Общая классификация ЦАП

### 2.6.1.1. Особенности работы параллельных ЦАП с суммированием токов

Существует два широко распространенных параллельных способа цифроаналогового преобразования посредством суммирования токов, показанных на рис. 2.49: с двумя номиналами сопротивлений  $R-2R$  и с весовыми двоично-взвешенными сопротивлениями.

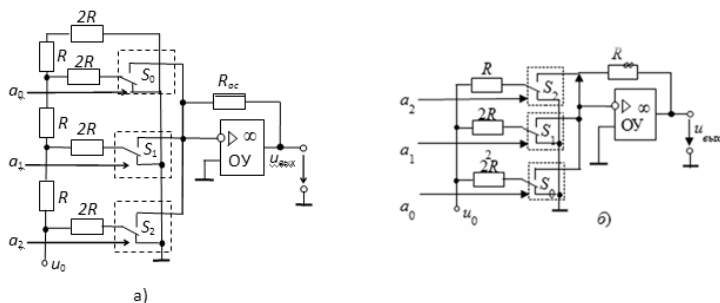


Рис. 2. 49. Параллельные 3-х разрядные ЦАП с суммированием токов на резистивных матрицах:

а) с двумя номиналами сопротивлений  $R-2R$ ;

б) с весовыми двоично-взвешенными сопротивлениями

**ЦАП с весовыми двоично-взвешенными сопротивлениями** (рис. 2.49, б) состоит: из  $n$  переключателей  $S_i$  (по одному на каждый разряд), управляемых двоичным кодом  $A_i$ ; из матрицы двоично-взвешенных резисторов с сопротивлениями  $2^{(n-1-i)} R$ ; источника опорного напряжения  $u_0$  и выходного операционного усилителя ОУ, с помощью которого суммируются токи, протекающие через резисторы, для получения аналогового выходного напряжения  $u_{\text{вых}}$ .

Каждый  $i$ -ый разряд управляет переключателем  $S_i$ , который подключается к источнику опорного напряжения  $u_0$ , когда  $a_i$  равно 1, или к общей шине, когда  $a_i$  равно 0. Сопротивления резисторов  $2^{(n-1-i)} R$  ( $n$  – разрядность ЦАП), соединенных с ключами, таковы, что обеспечивают пропорциональность в них тока двоичному весу соответствующего разряда входного кода. Следовательно, ток на входе ОУ и выходное напряжение ЦАП определяются как:

$$i = \frac{a_{n-1}u_0}{R} + \frac{a_{n-2}u_0}{2R} + \dots + \frac{a_1u_0}{2^{n-1}R} + \frac{a_0u_0}{2^n R}; \quad u_{\text{вых}} = -R_{oc}i = -u_0 \frac{R_{oc}}{2^n R} \sum_{i=0}^{n-1} a_i 2^i.$$



Напряжение на выходе ЦАП пропорционально «весу» присутствующего на входах кода, а максимальное значение имеет место, когда все разряды примут значение 1, т. е.

$$u_{max} = \left| u_0 \frac{(2^n - 1)R_{oc}}{2^n R} \right|,$$

и оно всегда меньше опорного напряжения на шаг квантования  $u_0 R_{oc} / (2^n R)$ .

Номиналы сопротивлений резисторов в младшем и старшем разрядах отличаются в  $2^{n-1}$  раз и должны быть выдержаны с высокой точностью. Например, для 12-разрядного ЦАП использование в старшем разряде резистора с сопротивлением 10 кОм потребует включения в младший разряд преобразователя резистора с сопротивлением порядка 20 МОм. Широкий набор номиналов резисторов и требования их высокой точности, в особенности при значительном числе разрядов  $n$  входного кода, создают трудности при реализации ЦАП посредством интегральной технологии.

В схеме ЦАП (см. рис. 2.49, а) с матрицей  $R-2R$  используют резисторы с двумя номиналами сопротивлений, причем резисторы с сопротивлением  $R$  включены в каждый разряд, однако в этой схеме увеличиваются значения паразитных емкостей.

Принцип работы представленной схемы основан на свойстве резистивного делителя  $R-2R$  сохранять постоянное сопротивление нагрузки для источника опорного напряжения при замыкании ключей. Вследствие этого на выводах резистора  $R$ , начиная со старшего  $n-1$  разряда, опорное напряжение последовательно делится пополам, как и входящий в каждый узел матрицы ток. При этом напряжение на выходе преобразователя с матрицей  $R-2R$ :

$$u_{вых} = -u_0 \frac{R_{oc}}{R} (a_{n-1} 2^{-1} + a_{n-2} 2^{-2} + \dots + a_1 2^{-(n-1)} + a_0 2^{-n}) = -u_0 \frac{R_{oc}}{2^n R} \sum_{i=0}^{n-1} a_i 2^i.$$

### 2.6.1.2. Основные технические характеристики

Основными параметрами ЦАП являются *число разрядов кода*, *абсолютная разрешающая способность*, *точность ЦАП*, *максимальная частота преобразования*.

**Число разрядов кода**,  $n$ , обычно составляет величину 8, ..., 24.

**Абсолютная разрешающая способность** – среднее значение минимального изменения сигнала на выходе ЦАП, обусловленное увеличением или уменьшением его кода на единицу.

Теоретически ЦАП, преобразующий  $n$ -разрядные двоичные коды, должен обеспечить  $2^n$  различных значений выходного сигнала с разрешающей способностью  $1/(2^n - 1)$ . При числе разрядов  $n = 8$ , количество независимых квантов (ступеней) выходного напряжения ЦАП равно  $2^8 - 1 = 255$ , при  $n = 12$ , количество независимых ступеней равно  $2^{12} - 1 = 4095$  и т. д.

**Абсолютное значение минимального кванта напряжения** определяется как предельным принимаемым числом  $2^n - 1$ , так и максимальным выходным напряжением ЦАП, также называемым напряжением шкалы или опорным напряжением  $u_o$ . Значение абсолютной разрешающей способности ЦАП, часто обозначается как ЗМР (значение младшего разряда).

При  $n = 8$  и опорном напряжении  $u_o = 5$  В,  $\text{ЗМР} = u_o / (2^8 - 1) = 5 / 255 \approx 0,0196$  В = 19,6 мВ.

Отличие реального значения разрешающей способности от теоретического обусловлено погрешностями и шумами входящих в ЦАП узлов.

**Точность ЦАП** определяется значением абсолютной погрешности  $\delta_a$  и нелинейностью преобразователя  $\delta_n$ .

**Абсолютная погрешность  $\delta_a$**  характеризуется отклонением максимального значения выходного напряжения  $u_{\text{max}}$  от расчетного, соответствующего конечной точке характеристики идеального преобразователя, и измеряется обычно в единицах ЗМР (рис. 2.49, линия 1).

**Нелинейность преобразователя  $\delta_n$**  характеризует отклонение действительной характеристики 2 от идеальной 1 (рис. 2.49), проведенной через центры ступенек или через нуль и точку максимального значения выходного сигнала.

Нелинейность обычно определяется в относительных единицах, но в справочных данных приводится также и в ЗМР. Для характеристики, приведенной на рис. 2.50,

$$\delta_n = \frac{\varepsilon_j}{U_{\text{пш}}} \cdot 100 \%$$

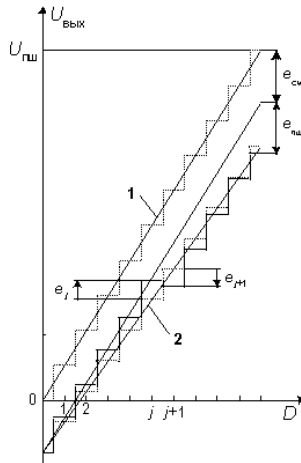


Рис. 2.49. Характеристики преобразования ЦАП:  
 1 – идеальная характеристика; 2 – реальная.  $U_{пш}$  – погрешность полной шкалы

**Дифференциальная нелинейность  $\delta_{дл}$**  – максимальное изменение (с учетом знака) отклонения реальной характеристики преобразования  $U_{вых}(D)$  от оптимальной при переходе от одного значения входного кода к другому смежному значению. Обычно определяется в относительных единицах или в ЗМР. Для характеристики, приведенной на рис. 2.50

$$\delta_{дл} = \frac{\epsilon_j + \epsilon_{j+1}}{U_{пш}} \cdot 100 \%$$

Из **динамических параметров** наиболее важным параметром является **максимальная частота преобразования  $f_{max}$**  (десятки и сотни кГц) – наибольшая частота дискретизации, при которой параметры ЦАП соответствуют заданным значениям.

Преобразование сигнала в ЦАП часто сопровождается специфическими переходными импульсами в выходном сигнале. Они возникают из-за разности времени открывания и закрывания аналоговых переключателей в ЦАП. Особенно сильное влияние переходных процессов проявляется, например, когда входной код 01...111 сменяется кодом 10...000, а переключатель старшего разряда ЦАП открывается позже, чем закрываются переключатели младших разрядов.

Следует отметить, что современные микросхемы ЦАП отличаются особенностями подключения (интерфейсом) к микроконтроллеру: параллельным (как в рассматриваемой ниже модели), так и последовательными стандартными интерфейсами типа I<sup>2</sup>C или SPI [9].

### 2.6.2. Ход выполнения лабораторной работы

**Задание 1.** Собрать схему (рис. 2.50) для испытания интегрального ЦАП [10]. В схеме использован библиотечный виртуальный 8-разрядный цифроаналоговый преобразователь A1 (из группы элементов Mixed), на входы которого могут подаваться сформированные с помощью переключателей J1, ..., J8 (группа Basic), соответствующие заданию двоичные коды (табл. 2.16). Требуется измерить с помощью вольтметра V1 (группа Indicator) или осциллографа XSC1 выходные напряжения ЦАП, отвечающие входным кодам задания (табл. 2.16). Затем полученные результаты заносятся в отчет по лабораторной работе.

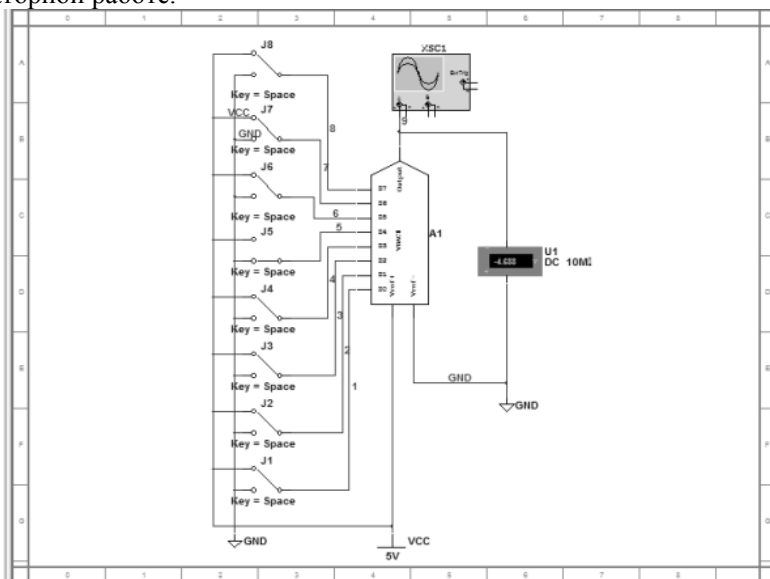


Рис. 2.50. Схема моделирования

**Задание 2.** Получить требуемые входные коды ЦАП при помощи микроконтроллера. Для этого создаем схемный проект Circuit6 (рис. 2.51), подключаем ЦАП к выводам порта P2 МК (или любого

другого порта). Полученные значения напряжения преобразования ЦАП сравниваем с результатами задания 1.

Программа совместной работы МК и ЦАП, написанная на С, показана ниже:

```
#include<8051.h>
void main()
{
  unsigned char var[3];
  //массив входных значений (согласно варианту задания);
  P2=0x00; // настройка порта P2 на выход;
  for(i=0;i<10;i++) // затем выводим код в цикле в порт P2;
  {
    P2 = var [ i]; //преобразование переменной в аналого-
                  //вое значение;
    for(j=0;j<100;j++)
      //временная задержка для устойчивого измерения результата;
      continue;
  }
  while(1);
}
```

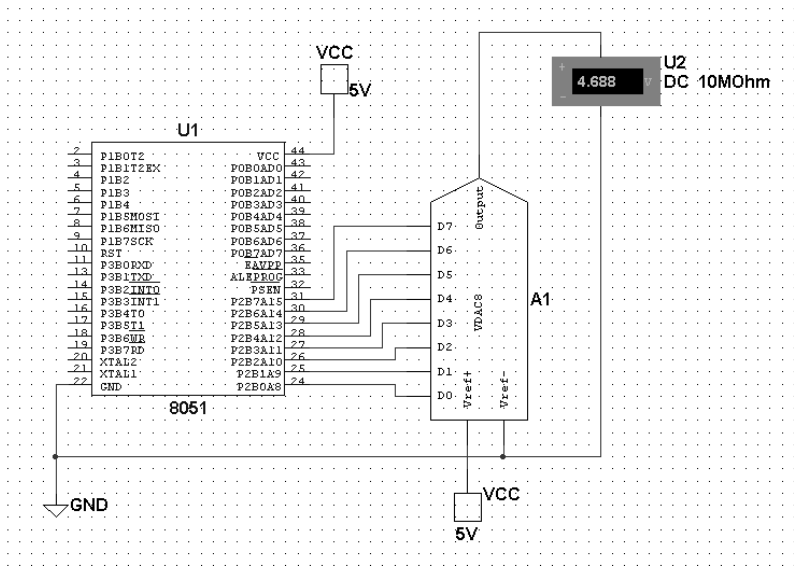


Рис. 2.51. Схема взаимодействия МК-51 с ЦАП

### 2.6.3. Задания для лабораторной работы

Требуемые значения входных 8-битных кодов ЦАП, соответствующие варианту задания (табл. 2.16), получить при помощи переключателей (см. рис. 2.50, задание 1) и при подключении ЦАП к одному из портов микроконтроллера (см. рис. 2.51, задание 2).

Таблица 2.16

Варианты задания

Параметры	Варианты задания									
	1	2	3	4	5	6	7	8	9	10
Коды	8	19	13	28	39	47	1	30	55	59
	36	101	99	125	88	165	100	145	155	144
	107	205	243	222	167	250	200	244	198	217
Порт подключения	P1	P2	P3	P1	P2	P3	P1	P2	P3	P1

### 2.6.4. Содержание отчета

1. Наименование и цель работы.
2. Копия схемного файла.
3. Копия программного файла (на ассемблере или на C) с подробными комментариями.
4. Полученные результаты и выводы по работе.

### 2.6.5. Вопросы для самоконтроля

1. Приведите примеры реализации последовательных ЦАП.
2. Назовите отличительные особенности ЦАП с двоично-взвешенными сопротивлениями от ЦАП на основе сопротивлений R-2R.
3. Назовите достоинства и недостатки ЦАП с последовательным интерфейсом.
4. Назовите достоинства и недостатки ЦАП с параллельным интерфейсом.
5. Приведите основные параметры ЦАП.
6. Какими параметрами определяется точность преобразования ЦАП?
7. Каковы схемные особенности использования порта P0 для подключения ЦАП?
8. Как определяется абсолютная разрешающая способность ЦАП при известном опорном напряжении и разрядности?
9. Почему возникают переходные процессы на выходе ЦАП?

## Лабораторная работа № 7. Изучение принципов работы аналого-цифровых преобразователей

**Цель работы:** изучение основ работы аналого-цифровых преобразователей (АЦП) и исследование особенностей их функционирования на примере виртуального 8-разрядного АЦП, представленного в базе данных Multisim.

### 2.7.1. Основные сведения об аналого-цифровых преобразователях

#### 2.7.1.1. Принципы преобразования непрерывных сигналов в цифровые коды

*Аналого-цифровой преобразователь (Analog-to-digital converter, ADC) – это устройство, которое принимает входные аналоговые сигналы и генерирует соответствующие им цифровые сигналы, пригодные для обработки микропроцессорами и другими цифровыми устройствами [9].*

Как правило, сначала различные по физической природе величины преобразуются в функционально связанные с ними электрические величины, а затем уже с помощью преобразователей напряжение-код – в цифровые.

Аналого-цифровое преобразование непрерывных сигналов, которое реализуют с помощью АЦП, представляет собой преобразование непрерывной функции времени  $U(t)$ , описывающей исходный сигнал, в последовательность чисел  $\{U_k(t_j)\}$ ,  $j = 0, 1, 2, \dots$ ,  $U_k = 0, 1, 2, \dots, N-1$ , отнесенных к некоторым фиксированным моментам времени. Такое преобразование состоит из двух независимых процедур. Первая из них называется **дискретизацией** и состоит в преобразовании непрерывной функции времени  $U(t)$  в непрерывную последовательность отсчетов  $\{U(t_j)\}$ . Вторая называется **квантованием** и состоит в преобразовании непрерывной по значению последовательности отсчетов  $\{U(t_j)\}$  в дискретную  $\{U_k(t_j)\}$ , где  $U_k = 0, 1, 2, \dots, N-1$ .

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

$$U(t) = \sum_j a_j f_j(t),$$

где  $a_j$  – некоторые коэффициенты или отсчеты, характеризующие исходный сигнал в дискретные моменты времени;  $j = 1, 2, 3, \dots$ ;

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

В основе наиболее распространенной равномерной дискретизации лежит теорема отсчетов Котельникова, для которой в качестве коэффициентов  $a_j$  следует использовать мгновенные значения сигнала  $U(t_j)$  в дискретные моменты времени  $t_j = j\Delta t$ , а период дискретизации выбирать из условия  $\Delta t = 1/2Fm$ , где  $Fm$  – максимальная частота спектра преобразуемого сигнала.

При выполнении процедуры **квантования** непрерывная по уровню последовательность отсчетов  $U(t_j)$  представляется  $\kappa$ -разрядным кодом  $U_\kappa(t_j)$ , равным  $0, 1, 2, \dots, N-1$ . Количество уровней квантования  $N$  зависит от значения  $\kappa$  – разрядности АЦП.  $N = 2^\kappa$ .

### ***Классификация и принцип действия АЦП***

В настоящее время используется большое число методов преобразования напряжение-код [9]. Эти методы отличаются потенциальной точностью, скоростью преобразования и сложностью аппаратной реализации. На рис. 2.52 представлена классификация АЦП по методам преобразования. В основе классификации АЦП используется признак, указывающий на то, как во времени разворачивается процесс преобразования аналоговой величины в цифровую. Для преобразования дискретных по времени сигналов в цифровые эквиваленты используются операции квантования и кодирования. Они могут быть осуществлены с помощью либо последовательной, либо параллельной, либо параллельно-последовательной процедур приближения цифрового эквивалента к преобразуемой величине.

Наиболее быстрыми являются параллельные АЦП, однако они самые дорогие и потребляют (рассеивают) значительную мощность. Последовательно-параллельные АЦП занимают промежуточное положение по разрешающей способности и быстродействию между параллельными АЦП и последовательными.



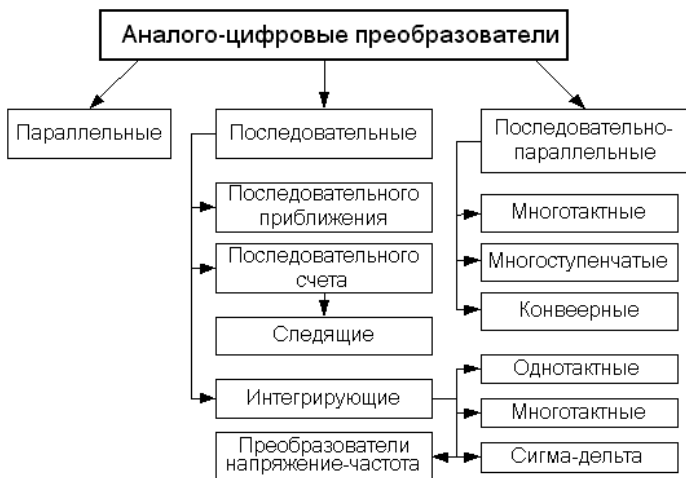


Рис. 2.52. Классификация АЦП

Особенности построения и функционирования различных типов АЦП изложены, например, в [9]. Рассматриваемые ниже АЦП последовательного счета и приближения находят широкое применение в системах управления, контроля и цифровой обработки сигналов.

### ***АЦП последовательного счета***

Этот преобразователь является типичным примером последовательных АЦП с единичными приближениями и состоит из компаратора, счетчика и ЦАП (рис. 2.53). На один из входов компаратора поступает преобразуемый аналоговый сигнал, а на другой – сигнал обратной связи с ЦАП.

Работа АЦП начинается с прихода импульса запуска, который включает счетчик, суммирующий число импульсов, поступающих от генератора тактовых импульсов – ГТИ. Выходной код счетчика подается на ЦАП, осуществляющий его преобразование в напряжение обратной связи  $U_{ос}$ . Процесс преобразования продолжается до тех пор, пока напряжение обратной связи сравнивается с входным напряжением и переключится компаратор, который своим выходным сигналом прекратит поступление тактовых импульсов на счетчик. Переход выходного значения компаратора из 1 в 0 означает завершение процесса преобразования.

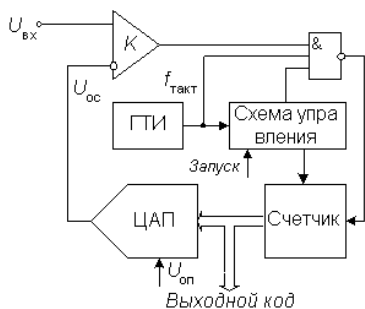


Рис. 2.53. Структурная схема АЦП последовательного счета

Полученный выходной код, пропорциональный входному напряжению в момент окончания преобразования, считывается с выхода счетчика.

АЦП данного типа без устройства выборки-хранения (УВ/Х) пригодны для работы с постоянными или медленно изменяющимися напряжениями, которые за время преобразования изменяются не более, чем на значение кванта преобразования. Таким образом, недостатком АЦП последовательного счета является небольшая частота дискретизации, достигающая нескольких килогерц.

Преимуществом АЦП данного класса является сравнительная простота построения, определяемая последовательным характером выполнения процесса преобразования.

#### ***Назначение и структура устройства выборки и хранения***

Устройство выборки и хранения (УВ/Х) предназначено для приема и хранения на время преобразования входных отсчетов, равных напряжениям входных сигналов в соответствующие дискретные моменты времени. Структурная схема УВ/Х представлена на рис. 2.54.

Необходимость хранения значения входного сигнала в течение времени его преобразования связана с неопределенностью значения этого сигнала в случае быстрого его изменения, что при отсутствии УВ/Х может привести к динамической ошибке преобразования. Кроме того, УВ/Х употребляется в многоканальных АЦП для запоминания мгновенных значений какого-либо канала и его преобразования с одновременным переключением мультиплексора на следующий канал.

По переднему фронту тактового сигнала ( $T_C$ ) устройство осуществляет отсчет (выборку) входного аналогового сигнала при замыкании ключа.



Рис. 2.54. Упрощенная структурная схема УВ/Х

Режим выборки характеризуется временем захвата (2, 4, 6, 10 и более мкс), когда конденсатор заряжается на величину  $U_{вх}(t)$ . Чем меньше время захвата (меньше емкость конденсатора  $C$ ), тем меньше соответствующая точность и увеличивается падение напряжения в процессе хранения.

По заднему фронту тактового сигнала  $ТС$  происходит размыкание ключа, и значение отсчета (напряжения) сохраняется, пока АЦП выполняет преобразование. Следует отметить, что хранимое напряжение не остается постоянным, из-за наличия утечки заряда конденсатора  $C$ . Связь УВ/Х с АЦП представлена на рис. 2.55.



Рис. 2.55. Упрощенная схема АЦП с УВ/Х

### ***АЦП последовательного приближения***

Преобразователь этого типа, также называемый АЦП с поразрядным уравниванием, является наиболее распространенным вариантом последовательных АЦП. В основе его работы лежит принцип дихотомии, т. е. последовательного сравнения измеряемой величины с  $1/2$ ,  $1/4$ ,  $1/8$  и т. д. от возможного максимального значения ее.

Рассмотрим основы работы АЦП последовательного приближения на примере классической структуры 4-разрядного преобразователя, состоящего из трех основных узлов: компаратора, регистра последовательного приближения (РПП) и ЦАП (рис. 2.57).

После подачи команды «Пуск» с приходом первого тактового импульса РПП принудительно задает на входе ЦАП код, равный по-

ловине его шкалы (для 4-разрядного ЦАП это  $1000_2 = 8_{10}$ ). Благодаря этому напряжение обратной связи на выходе ЦАП  $-U_{oc} = 2^3 h$ , где  $h$  – квант выходного напряжения ЦАП, соответствующий единице младшего разряда (ЕМР).

Данное значение  $U_{oc}$  составляет половину возможного диапазона преобразуемых сигналов. Если входное напряжение АЦП больше, чем эта величина, то на выходе компаратора устанавливается 1, если меньше, то 0. При 0 на выходе компаратора схема управления должна переключить старший разряд  $d_3$  АЦП обратно в состояние нуля. Полученный остаток  $(U_{вх} - d_3 2^3 h)$  таким же образом сравнивается с ближайшим младшим разрядом и т. д.

После четырех подобных шагов в регистре последовательного приближения оказывается двоичное число, из которого после цифро-аналогового преобразования получается напряжение, соответствующее значению  $U_{вх}$  с точностью до ЕМР. В процессе преобразования на выходе компаратора (см. рис. 2.56) формируется выходное число в виде последовательного кода старшими разрядами вперед.

Данный класс АЦП занимает промежуточное положение между последовательно-параллельными и интегрирующими АЦП по быстродействию, стоимости и разрешающей способности и находит широкое применение в системах управления, контроля и цифровой обработки сигналов.

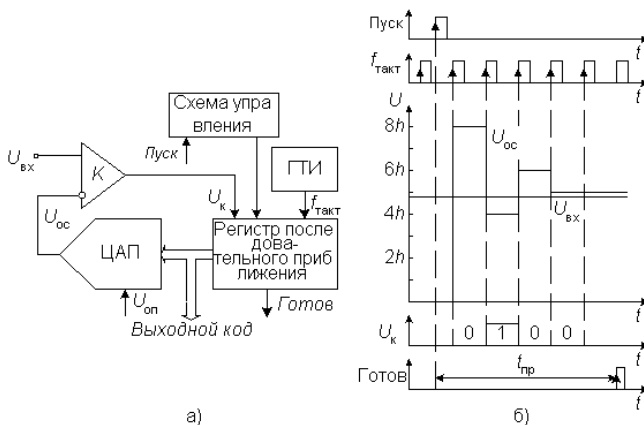


Рис. 2.56. Структурная схема АЦП последовательного приближения

### 2.7.1.3. Основные параметры АЦП

К основным параметрам АЦП относят число разрядов выходного кода, диапазон изменения входного напряжения  $u_{\text{вх. max}}$ , абсолютную разрешающую способность, абсолютную погрешность преобразования, максимальную частоту преобразования, время преобразования входного сигнала.

**Число разрядов выходного кода  $k$**  отображает исходную аналоговую величину, которая формируется на выходе АЦП. При использовании двоичного кода  $k = \log_2 N$ , где  $N$  – максимальное число кодовых комбинаций (уровней квантования) на выходе АЦП (0, 1, ...  $N-1$ ).

**Диапазон изменения входного напряжения –  $u_{\text{вх. max}}$ .** Отметим, что АЦП может обрабатывать входную информацию в виде однополярного аналогового напряжения с пределами 0, ...,  $u_{\text{вх. max}}$  и двуполярного  $\pm u_{\text{вх. max}}/2$ .

**Абсолютная разрешающая способность** – среднее значение минимального изменения входного сигнала  $u_{\text{вх}}$ , обуславливающего увеличение или уменьшение выходного кода на единицу. Разрешающая способность определяется разрядностью выходного кода и диапазоном входного напряжения.

**Абсолютная погрешность преобразования –  $\delta_l$ ,** в конечной точке шкалы есть отклонение реального максимального значения входного сигнала  $u_{\text{вх. max}}$  от максимального значения идеальной характеристики  $L$  АЦП (рис. 2.57). Обычно  $\delta_l$  измеряется в ЗМР – значении младшего разряда.

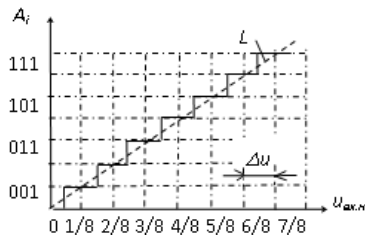


Рис. 2.57. Абсолютная погрешность преобразования АЦП

**Максимальная частота преобразования** (десятки и сотни килогерц) – максимальное значение частоты входного сигнала.

**Время преобразования входного сигнала  $t_{\text{пр. max}} \leq (1/2)\Delta t$ .**

## 2.7.2. Ход выполнения лабораторной работы

**Задание 1.** Собрать схему для исследования точности преобразования АЦП [10].

Создаем схемный проект (рис. 2.58), в котором использованы элементы: виртуальный 8-разрядный АЦП – А1 (рис. 2.60) и виртуальный ЦАП – А2 из группы элементов Mixed; источники напряжений (из группы элементов Sources): постоянного опорного напряжения V1 и V2 (подключены ко входам Vref+ и Vref-, если в варианте задания указано разнополярное опорное напряжение, либо Vref+ и земля, если используется только положительное опорное напряжение); генератор импульсного напряжения V4 для синхронизации работы АЦП (подключен к входу SOC); источник постоянного напряжения V3 для подачи входного напряжения, которое необходимо преобразовать в код; пробники X0, ..., X9 и вольтметр U1 (из группы элементов Indicators), а также инвертор U2A из группы элементов TTL, который подает перепад напряжения от 5 В до 0 на вход OE АЦП от источника V4 в противофазе с входом SOC, разрешая появления цифрового кода D0...D7 на выходах АЦП.

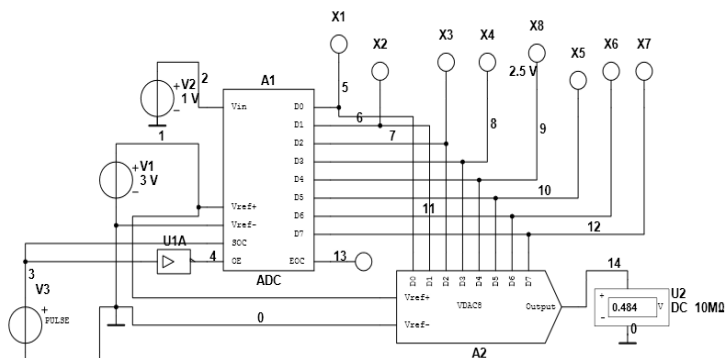


Рис. 2.59. Схема для исследования работы 8-разрядного АЦП

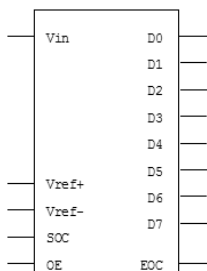


Рис. 2.60. Условное графическое обозначение УГО АЦП

Назначение выводов используемого в схеме АЦП приводится в табл. 2.17.

Таблица 2.17

Назначение основных выводов виртуального АЦП

Обозначение	Вход / выход	Назначение
$V_{in}$	I	Аналоговое входное напряжение
$V_{ref+}$	I	Опорное напряжение (+)
$V_{ref-}$	I	Опорное напряжение (-)
D0- D7	O	Цифровые выходы
SOC	I	Начать преобразование
EOC	O	Конец преобразований
OE	I	Разрешение цифрового вывода

Исследования точности АЦП проводится по следующей схеме [10].

1. Для источников  $V1$  и  $V2$  устанавливаются соответствующие заданию (табл. 2.19) значения опорного напряжения, например,  $V1 = +2,5$  В,  $V2 = -2,5$  В.
2. Поочередно на вход  $V_{in}$  АЦП подается 10 различных напряжений из диапазона изменения опорного напряжения  $V_{ref}$ , например,  $u_{вх} = 0,1; 0,2; 0,5; 1,0; 1,5; 2,0; 2,4; -0,5; -1,0; -2,0$  В (используется источник входного напряжения  $V3$ ).
3. Для каждого значения входного напряжения  $V_{in}$  запускается программа моделирования. Выходной код, определяемый по свечению пробников  $X7, \dots, X0$ , и значения напряжения  $u_{вых}$  (ЦАП) с выхода ЦАП, измеряемые вольтметром  $UI$ , заносятся в поля таблицы вида 2.18.
4. Для 8-разрядного АЦП по значению входного напряжения  $u_{вх}$  можно определить значение выходного кода –  $D_{(10)расч}$  по формуле:

$$D_{(10)расч} = 256u_{вх} / (V1 + |-V2|), \quad (2.7.1)$$

5. Производятся необходимые расчеты для заполнения всех столбцов табл. 2.18. Если значения опорного напряжения разнополярные, то код, определяемый по свечению пробников, является инверсным. При этом, если полученное значение кода больше 128, то результат положительный, а если меньше – отрицательный. Десятичные инверсные сигналы  $D_{(10)\text{инв}}$  преобразуются в прямые  $D_{(10)}$  по выражению:

$$D_{(10)} = D_{(10)\text{инв}} - 128. \quad (2.7.2)$$

6. Рассчитываются погрешности измерения напряжения по выражению:

$$\Delta U\% = 100(u_{\text{вых(ЦАП)}} - u_{\text{вх}})/u_{\text{вх}}. \quad (2.7.3)$$

Таблица 2.18

Пример исследования АЦП

$u_{\text{вх}},$ В	$u_{\text{вых(ЦАП)},}$ В	$D_{(2)}$	$D_{(16)}$	$D_{(10)\text{инв}}$	$D_{(10)}$	$D_{(10)\text{расч}}$	$\Delta U, \%$
0,1	0,09375	10000100	84	132	4	4,27	6,25
0,5	0,5156	10010101	95	149	21	21,33	3,12
1,0	0,9644	10101010	AA	170	42	42,67	3,56
2,0	2,017	11010101	D5	213	85	85,34	0,85
2,5	2,484	11101010	EA	234	106	106,67	0,64
2,9	2,906	11111011	FB	251	123	123,74	0,21
-1,0	-0,9844	01010101	55	85	-43	-42,67	3,56

**Задание 2.** Выполнить соответствующий вариант задания при подключении МК к АЦП. Получить значения выходного кода АЦП для любых трех значений изменения  $u_{\text{вх}}$  (выбирается из диапазона изменений  $V_{\text{ref}}$ , см. табл. 2.19).

Создаем схемный проект Circuit 7, устанавливаем на рабочей области МК-51, исследуемый виртуальный АЦП (8- или 16-разрядный). Подаем с источника постоянного тока  $V1$  на вход АЦП напряжение, например, 1В, к выводу опорного напряжения АЦП подключаем источник напряжения  $V2$ , например, 5В. К порту P2 МК подключаем пробники X0, ..., X7, вывод P3.6 соединяем с входом АЦП – SOC. Схема подключения MCS-51 к АЦП представлена на рис. 2.61.



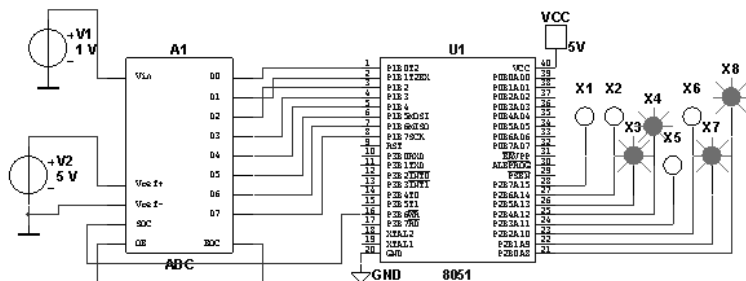


Рис. 2.61. Схема подключения МК-51 к АЦП

Активируем вкладку программного файла main.c и вставим следующий программный код:

```
#include <8051.h> //подключаем заголовочный файл
void main ()
{
    unsigned char val; //переменная содержит цифровой код, отвечающий Uвх
    P1=0xFF; //инициализируем P1 на вход
    P2=0x00; //инициализируем P2 на выход
    P36=0; //P3.6 – выход
    P36=1; //начать преобразования
    val=P1; //P1 содержит цифровой код
    P2=val; //включаем пробники
    while (1); //бесконечный цикл
}
```

Пробник показывает цифровой код, равный 51, что соответствует выражению (2.7.2).

### 2.7.3. Задания для лабораторной работы

В соответствие с вариантом задания (табл. 2.19) исследовать точность виртуального АЦП (задание 1) согласно схеме, приведенной на рис. 2.59, и заполнить таблицу вида 2.18. Порядок выполнения задания 1 приведен в п. 2.7.2. Для выполнения задания 2 собрать схему (рис. 2.61) и для любых 3 значений  $U_{вх}$  из таблицы вида 2.18 при помощи пробников снять полученный цифровой код.

Таблица 2.19

## Варианты заданий

№	1	2	3	4	5	6	7	8	9	10
Разрядность АЦП	8	16	8	16	8	16	8	16	8	16
$E_{ref1}$	+2,5	+5	+4	+5	+3	+2,5	+2	+5	+5	+4
$E_{ref2}$	-2,5	0	-1	0	-2	-2,5	-2	0	0	-1

**2.7.4. Содержание отчета**

1. Название и цель работы.
2. Перечень элементов, использованных в схеме, с их краткими характеристиками.
3. Копия окна схемного файла при моделировании.
4. Заполненная при проведении моделирования по схеме 2.59 таблица типа 2.18.
5. Снятые показания пробников при моделировании по схеме 2.61.
6. Выводы по работе.

**2.7.5. Вопросы для самоконтроля**

1. Приведите примеры использования АЦП.
2. Дайте определение разрешающей способности АЦП.
3. Как определяется частота дискретизации аналогового сигнала?
4. В каких случаях используется двуполярное опорное напряжение?
5. Как производится квантование аналогового сигнала по уровню?
6. Как определяется цифровой код АЦП при известном входном аналоговом сигнале и заданных опорных напряжениях?
7. Чем отличается принцип работы АЦП последовательного счета от АЦП последовательного приближения?
8. В чем отличие принципов работы параллельных АЦП от принципов работы последовательных АЦП?
9. Для чего в структуре АЦП необходимо использовать устройство выборки и хранения?
10. Как определяется относительная точность преобразования АЦП?

## **Лабораторная работа № 8. Исследование широтно-импульсной модуляции, реализованной микроконтроллером МК-52**

**Цель работы:** получить широтно-импульсную модуляцию (ШИМ) с требуемыми параметрами при помощи таймера T/C2, входящего в состав микроконтроллера МК-52.

### **2.8.1. Основы применения микроконтроллера МК-52 для получения ШИМ**

#### **2.8.1.1. Общие сведения о широтно-импульсной модуляции**

*Широтно-импульсной модуляцией (Pulse Width Modulation – PWM) называется импульсный сигнал постоянной частоты и переменной скважности (скважность – отношение длительности импульса к периоду его следования).*

Если научиться управлять скважностью сигнала, то можно изменять среднее напряжение на выходе ШИМ (в сущности, получать аналоговый сигнал, служащий для управления исполнительными устройствами). Например, при изменении среднего значения напряжения, подаваемого на двигатель, изменяется скорость его вращения [7] (рис. 2.62).

Скважность, обозначенная  $Q$ , на рис. 2.62–1 равна 50 %, поскольку отношение длительности импульса к периоду следования равно  $1/2$ . Если, например, используется напряжение питания 12 В, то при скважности 50 % среднее значение напряжения на двигателе будет равно 6 В.

Скважность импульсов на рис. 2.62–2 равна 75 %, а среднее напряжение на двигателе увеличивается до 9 В (при питании 12 В). Поскольку среднее напряжение, приложенное к двигателю, возрастает, следовательно, скорость вращения тоже возрастает.

Скважность импульсов на рис. 2.62–3 уменьшается до 25 %, в результате чего среднее значение напряжения за период уменьшается до 3 В, что способствует замедлению скорости вращения.

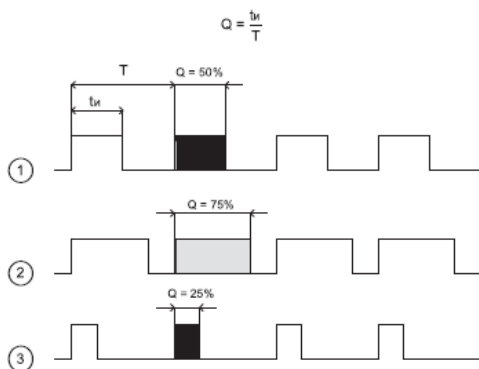


Рис. 2.62. Широтно-импульсная модуляция. Диаграммы скважности

*Основное преимущество использования ШИМ – малые потери мощности в исполнительном устройстве.*

ШИМ удобно использовать и при управлении релейными устройствами, в силу специфики их работы (включения/выключения), для них можно легко установить нужный рабочий цикл.

Пример применения ШИМ в двигателе переменного тока показан на рис. 2.63. Линейное напряжение модулируется в виде серии импульсов, что приводит к появлению синусоидальной составляющей (огibaющая) в магнитной цепи двигателя.

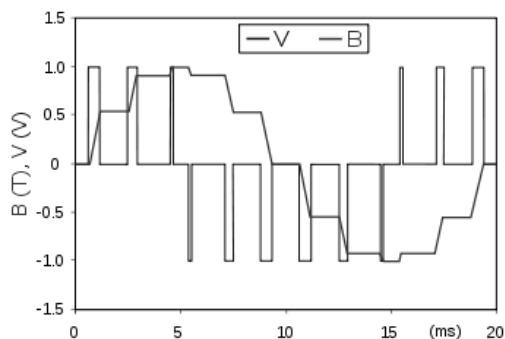


Рис. 2.63. Пример ШИМ в двигателе переменного тока

Покажем реализацию ШИМ на основе дополнительного (по сравнению с МК-51) таймера 2, встроенного в МК 8052. На одном из выводов МК формируется последовательность импульсов с посто-

янным периодом, равным периоду работы счетчика временной базы (в лабораторной работе таймер 2), а скважность зависит от переменной, определяющей длительность импульса.

### 2.8.1.2. Особенности организации микроконтроллера МК-52

МК-52 имеет дополнительный по сравнению с МК-51 встроенный таймер T/C2, который включает ряд особенностей и дополнительных возможностей по сравнению с таймерами T/C0 и T/C1 в классических МК-51. Таймер T/C2 управляется группой специальных регистров, ниже в скобках указываются адреса регистров в пространстве SFR:

- T2CON – регистр управления и контроля (0C8h). Регистр может адресоваться побитно;
- RCAP2H – старший байт, содержимое которого используется при работе в режиме автоперезагрузки (0CBh);
- RCAP2L – младший байт, содержимое которого используется при работе в режиме автоперезагрузки (0CAh);
- TH1 – старший байт таймера 2 (0CDh);
- TL1 – младший байт таймера 2 (0CCh).

Таймер 2 может работать в нескольких режимах: в качестве 16-разрядного таймера, в режиме автоперезагрузки, в режиме генератора синхронизации при последовательном обмене данными (управляет скоростью обмена последовательного порта), а также в режиме захвата – может перейти в режим прерываний при возникновении определенного события (capture mode).

Требуемый режим работы задается при программировании регистра функций T2CON, назначение битов которого проиллюстрировано на рис. 2.64 [2].

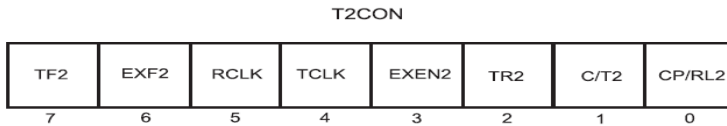


Рис. 2.64. Регистр T2CON

**TF2** – бит переполнения таймера 2. При разрешении прерывания от таймера 2, установка этого бита вызывает прерывание, программно-обработчик которого должна располагаться по адресу 02Bh в пространстве кодов.

**EXF2** – устанавливается в 1 при возникновении переполнения таймера 2 или, если сигнал на выводе T2EX (P1.1) переходит из высокого в низкий уровень (перепад 1–0). Перепад сигнала фиксируется только при установленном бите EXEN2.

**RCLK** – если данный бит установлен (равен 1), то таймер 2 используется в качестве генератора синхронизации последовательного порта при приеме данных. Если бит равен 0, то для синхронизации используется таймер 1.

**TCLK** – если данный бит установлен (равен 1), то таймер 2 используется в качестве генератора синхронизации последовательного порта при передаче данных. Если бит равен 0, то для синхронизации используется таймер 1.

**EXEN2** – при установленном бите перепад сигнала из высокого в низкий уровень на выводе T2EX (P1.1) инициирует режим захвата или вызывает переполнение таймера 2.

**TR2** – установка этого бита в 1 разрешает работу таймера 2. Установка бита в 0 останавливает работу таймера 2.

**C/T2** – при нулевом значении бита таймер 2 функционирует в режиме интервального таймера. Если данный бит установлен (равен 1), то таймер 2 инкрементируется каждый раз при перепаде сигнала из 1 в 0 на выводе T2 (P1.0) МК-52.

**CP/RL2** – при нулевом значении этого бита переполнение таймера 2 возникает при работе в режиме автоперезагрузки или при перепаде сигнала 1–0 на выводе T2EX (бит EXEN2 должен быть установлен). Если данный бит установлен (равен 1), то таймер 2 работает в режиме захвата при возникновении перепада 1–0 на выводе T2EX (бит EXEN2 должен быть установлен).

Рассмотрим основные режимы работы таймера 2.

При работе в *режиме захвата* при установке флага EXEN2 таймер может реагировать на перепад 1–0 на выводе T2EX (P1.1). В момент фиксации перепада текущие значения регистров TH2 и TL2 запоминаются в регистрах RCAP2H и RCAP2L соответственно, и устанавливается флаг EXF2, вызывающий прерывание таймера 2.

Следует иметь в виду, что даже при установленном режиме захвата установка флага TF2, сигнализирующего о переполнении таймера, также вызовет прерывание.

При работе в *режиме прямого счета с автоперезагрузкой* таймер 2 при переполнении перезагружается значениями, находящимися в регистрах RCAP2H (старшая часть) и RCAP2L (младшая часть).

Этот режим напоминает аналогичные режимы для таймеров T/C0 и T/C1 с той разницей, что там используется максимум 8 разрядов, что ограничивает диапазон 256 значениями, в то время как таймер 2 использует 16-разрядные значения перезагрузки.

### Режим задающего генератора

Если таймер T2 используется для управления работы последовательного порта, то скорость работы последовательного порта определяется формулой:

$$\text{Скорость} = F_{osc} / [32 * ((65536 - (RCAP2H.RCAP2L))]. \quad (2.8.1)$$

Значение регистров RCAP2H.RCAP2L для нужной скорости обмена можно получить по формуле:

$$RCAP2H.RCAP2L = 65536 - F_{osc} / (32 * \text{скорость}). \quad (2.8.2)$$

Таймер 2 имеет вектор прерывания, расположенный по адресу 02Bh. В регистре разрешения прерывания IE МК семейства MCS-51 [1] добавлен бит IT2 (IE.5), отвечающий за прерывания T/C2.

### 2.8.2. Ход выполнения лабораторной работы

**Задание:** получить на выводе P1.0 МК-52 широтно-импульсную модуляцию с заданной частотой сигнала и скважностью (табл. 2.20).

Создаем схемный проект Circuit 8. Для этого размещаем на схеме МК-52, осциллограф XSC-1, подключенный к порту P1.0 (рис. 2.65).

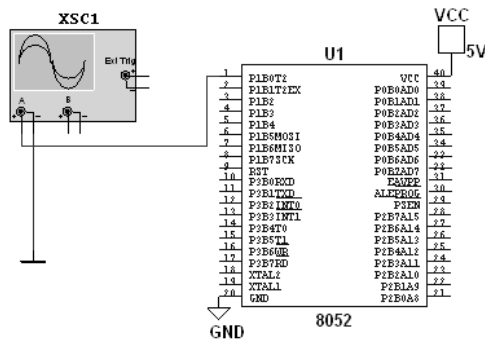


Рис. 2. 65. Схема реализации ШИМ

Используется режим *автоперезагрузки таймера 2*.

Частота ШИМ (или период следования импульсов на выводе P1.0) зависит от частоты  $F_{осц}$  и задается в свойствах МК.

Период следования импульсов –  $T$  (с) определяется содержимым регистров  $RCAP2H$  и  $RCAP2L$ . Для вычисления необходимых значений используются следующие выражения:

$$F_{\text{осц}}/12 = C \text{ (Гц)} - \text{частота следования машинного цикла МК};$$

$$T \text{ (с)} = (65536 - R)/C, \quad (2.8.3)$$

где  $R$  – значение величины перезагрузки, заносимое в регистры  $RCAP2H$  и  $RCAP2L$  (старшая и младшая часть соответственно).

Отсюда получаем:  $R = 65536 - C * T$ , полученное значение следует округлить до ближайшего целого и перевести в шестнадцатеричную систему.

Скважность определяется по формуле:

$$Q = t/T,$$

где  $t$  – длительность импульса (время высокого уровня), Скважность не зависит от частоты сигнала.

В программе скважность регулируется переменной  $tmpCnt$  и может быть определена следующим образом:

$$Q = tmpCnt / (65536 - R). \quad (2.8.4)$$

В приводимом примере используются нулевые значения перезагрузки регистров  $RCAP2H$ ,  $RCAP2L$ . При частоте кварца 11,059 МГц период переключения таймера 2 согласно (2.8.3) равен 71 мс. Чтобы уменьшить значение периода переключения ШИМ, необходимо увеличить содержимое регистров  $RCAP2H$ ,  $RCAP2L$ . Например, для реализации периода переключения в 50 мс требуется инициализировать  $RCAP2H$ ,  $RCAP2L$  значением 4925h.

Рассмотрим программу управления ШИМ, написанную на С.

```
#include <8052.H>           //подключаем заголовочный файл для МК
                           8052
#define _imkstr_(x)#x      //макрос задания обработчика прерывания
                           T/C2 для компилятора Multisim10
#define ROM_VECTOR (TIMER2, t2int_handler)
    asm("global "_imkstr_(t2int_handler));
    asm("psectvectors, ovrl");
    asm("org "_imkstr_(TIMER2));
    asm("_ljump_"_imkstr_(t2int_handler));
    asm("psecttext")
unsigned int cnt;          //переменная цикла
unsigned int tmpCnt;       //переменная, определяющая скважность
```



```

bank2                //смена банка РОН при работе подпрограммы
                    прерываний
//Функция обработчика прерывания Таймера 2
interrupt void t2int_handler(void)
{
TF2 = 0;             //очистка флага переполнения таймера 2
cnt = tmpCnt;       //переменная tmpCnt задается в main
P10 = 1;            //вкл
while (cnt != 0)cnt--;
P10 = 0;            //выкл
}
void main()         //головная программа
{
tmpCnt = 4000;      //устанавливаем скважность импульсов
ROM_VECTOR(TIMER2, t2int_handler); //задаем обработчик преры-
                               ваний
P1 = 0xFE;          //настраиваем бит P1.0 на выход
RCAP2H = 0x0;      //старший бит регистра задающего частоту
                   сигнала
RCAP2L = 0x0;      //младший бит регистра задающего частоту
                   сигнала
//Задаем режим работы таймера 2 на автоперезагрузку
T2CON &= 0xFC;
ET2 = 1;           //разрешаем прерывания от T/C2
EA = 1;           //разрешаем глобальные прерывания
T2CON |= 0x4;     //запуск таймера
while (1);
}

```

### 2.8.3. Задания для лабораторной работы

При заданной согласно варианту задания (табл. 2.20) тактовой частоте работы МК –  $F$  (МГц) настроить микроконтроллер на режим автоперезагрузки, рассчитать значения  $RCAP2H$ ,  $RCAP2L$  для получения требуемого периода следования импульсов –  $T$  (с) и реализовать на выводе P1.0 ШИМ с заданной скважностью  $Q$  %, вычисляя необходимое значение переменной  $tmpCnt$  согласно формуле (2.8.4). Вывести на осциллограф полученную на выводе P1.0 последовательность импульсов.

Таблица 2.20

Варианты задания

Параметры	1	2	3	4	5	6	7	8	9	10
$T, c$	0,07	0,07	0,05	0,06	0,02	0,04	0,07	0,05	0,06	0,03
$Q, \%$	50	50	25	75	50	50	25	75	50	50
$F, МГц$	11	12	11	13	15	13	11	12	12	15

#### 2.8.4. Содержание отчета

1. Название и цель работы.
2. Перечень элементов, использованных в схеме, с их краткими характеристиками.
3. Копия окна схемного файла при моделировании.
4. Копия программного файла с подробными комментариями.
5. Выводы по работе.

#### 2.8.5. Вопросы для самоконтроля

1. Какие отличительные особенности имеет таймер 2 по сравнению с таймерами МК51?
2. Как запрограммировать таймер 2 на режим синхронизации последовательного порта для приема?
3. Как запрограммировать таймер 2 на режим синхронизации последовательного порта для передачи?
4. Как можно использовать таймер 2 в режиме захвата?
5. В чем отличие настройки таймера 1 от таймера 2 при использовании для синхронизации последовательного обмена?
6. В чем отличие программно-управляемого режима работы таймера от режима работы по прерываниям?
7. Поясните назначение битов регистра T2CON.
8. Поясните понятие вектора прерывания. Какие вектора прерывания имеет МК-52?
9. Поясните назначение регистра IE – маски прерываний?
10. Можно ли изменить приоритет прерывания источников запроса? Какой приоритет (по умолчанию) имеет таймер 2?

## Заключение

В учебно-методическом пособии рассмотрена теория и практика решения часто встречающихся типовых задач проектирования микропроцессорных устройств, построенных на основе микроконтроллеров семейства MCS-51. Для моделирования разрабатываемых систем использована программная среда Multisim 10.0.

В лабораторный практикум включены решения следующих задач:

- основы работы с программным комплексом Multisim;
- подключение внешней памяти и ее тестирование;
- реализация требуемых значений временных интервалов;
- прием/передача последовательной информации;
- основы отображение информации в системах с МК-51;
- основы преобразования информации из аналоговой формы в цифровую и из цифровой в аналоговую;
- получение и исследование широтно-импульсной модуляции.

Описаны принципы структурного проектирования микропроцессорных устройств как базы для успешной реализации проекта.

Для каждой лабораторной работы (кроме первой) приведены типовые примеры проектирования с программированием МК 51/52. Для большинства задач даются примеры программирования МК как на ассемблере, так и на языке С, поскольку в сложных проектах именно их сочетание дает оптимальный результат.

При выполнении лабораторных работ студенты самостоятельно выбирают язык программирования проектируемого устройства (в соответствии со своими знаниями и предпочтениями).

В конце каждой лабораторной работы приводятся варианты заданий и вопросы для самоконтроля.

Автор выражает благодарность за помощь в подготовке работы к публикации и отладке ПО выпускнице 2011 г. Шишко Е. и выпускникам 2012 г. С. Михайлову, В. Пальчикову, Н. Рыбалко, А. Сибилеву.

## Библиографический список

1. Изучение архитектуры однокристального микроконтроллера базового семейства 80x51: методические указания к лабораторной работе / А. А. Шегал, О. А. Черных. Екатеринбург : УГТУ-УПИ, 2006. 44 с.
2. Бродин В. Б. Микроконтроллеры. Архитектура, программирование, интерфейс / В. Б. Бродин, И. И. Шагурин. М. : Издательство ЭКОМ, 1999. 400 с.
3. Баррет С. Ф. Встраиваемые системы. Проектирование приложений на микроконтроллерах семейства 68HC12/HCS12 с применением языка С / С. Ф. Барретт, Д. Дж. Пак. М. : ДМК-пресс, 2007. 640 с.
4. Микушин А. В. Занимательно о микроконтроллерах / А. В. Микушин. С-Пб. : БХВ-Петербург, 2006. 432 с.
5. Загидуллин Р. Ш. Multisim, LabView, Signal Express / Р. Ш. Загидуллин. М. : Горячая линия-Телеком, 2009. 366 с.
6. Белов А. В. Конструирование устройств на микроконтроллерах / А. В. Белов. СПб. : Наука и Техника, 2005. 256 с.
7. Магда Ю. С. Микроконтроллеры серии 8051: практический подход / Ю. С. Магда. М. : ДМК, 2008. 228 с.
8. Угрюмов Е. П. Цифровая схемотехника: учебное пособие / Е. П. Угрюмов. СПб. БХВ. С-Пб. : 2010. 528 с.
9. Волович Г. И. Схемотехника аналоговых и аналого-цифровых электронных устройств / Г. И. Волович. М. : Издательский дом «Додэка-XXI», 2005. 528 с.
10. Марченко А. Л. Лабораторный практикум по электротехнике и электронике в среде Multisim: учебное пособие для вузов / А. М. Марченко, С. В. Освальд. М. : ДМК Пресс, 2010. 448 с.

## Настройка виртуальных приборов: генератора слов, логического генератора, функционального генератора

### 1. Особенности работы генератора слов

Виртуальный прибор генератор слов [10]. (Word generator, рис. П.1) используется для создания цифровых сигналов. Генератор является 32-разрядным, может представлять данные в нескольких режимах: шестнадцатеричном, десятичном, двоичном, в символах ASCII.

Генератор содержит ячейки со значениями сигнала, заполняемые во вкладке Set... или вручную.

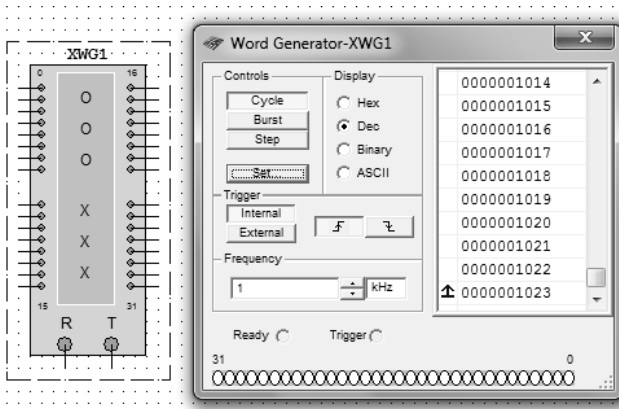


Рис. П.1. Символ и лицевая панель генератора слов

Для заполнения ячеек требуемыми значениями сигнала необходимо два раза щелкнуть ЛКМ по символу генератора слов. В открывшемся окне прибора необходимо вручную или с помощью опций меню *Set...* задать значения ячеек. Допустим, необходимо запрограммировать генератор слов на возрастание и убывание шестнадцатеричных чисел вручную от 0 до FF ( $255_{10}$ ) при шаге  $10_{16}$  ( $16_{10}$ ). Для использования другого кода (двоичного, или десятичного) в окне настроек генератора необходимо в разделе *Display* установить флажок напротив требуемого кода.

Чтобы установить начальную и конечную ячейки значений (они обозначены синими стрелками вниз и вверх соответственно), необходимо щелкнуть на требуемую ячейку правой кнопкой мыши и выбрать пункт Set Initial Position (Установить начальную позицию) или Set Final Position (Установить конечную позицию, рис. П.2.). Для введения точки останова (контрольной точки) используется опция Set Break-Point, а для обозначения текущей ячейки – Set Cursor.

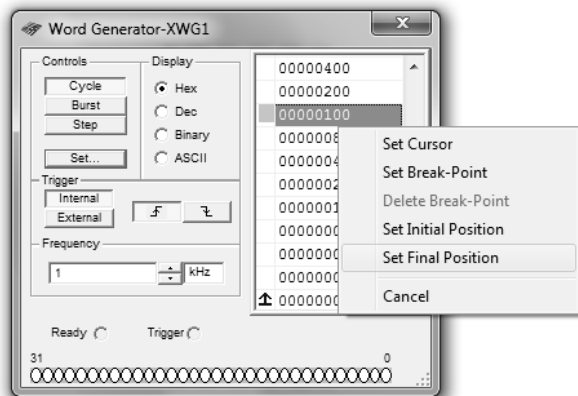


Рис. П.2. Установка конечной ячейки значений сигнала генератора

Для автоматического заполнения ячеек удобно пользоваться закладкой Settings (Настройки), вызываемой кнопкой Set... в окне настроек генератора слов (Рис. П.3).

Опции настройки Set... (Рис. П.3):

- Up Counter – установить возрастающий счет в ячейках;
- Down Counter – установить обратный счет в ячейках;
- Shift Right – правая запись (8–4–2–1 со смещением их в следующих четырех ячейках вправо);
- Shift Left – левая запись (1–2–4–8 со смещением их в следующих четырех ячейках влево);

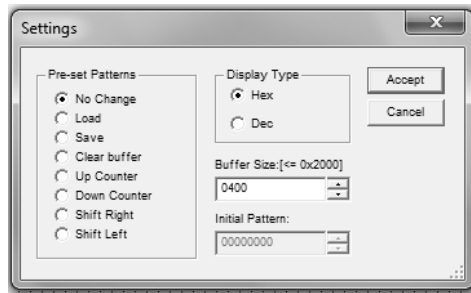


Рис. П.3. Окно закладки Settings (Настройки), содержащей различные опции настройки заполнения ячеек генератора.

- Clear buffer – очистка ячеек;
- Save, Load – сохранение и загрузка пользовательских настроек;

В закладке Settings также можно задать требуемое количество ячеек для значений (Максимальное – 8192).

Генератор слов может работать в 3 режимах, которые выбираются в разделе Controls меню настроек:

- Cycle – режим повторения циклического счета до окончания работы моделирования;
- Burst – режим выполнения счета один раз;
- Step – выводится содержимое только следующей ячейки.

Частота вывода значений сигнала из ячеек прибора для первого и второго режимов работы задается в разделе *Frequency* (рис. П.2). Рекомендуется использовать номинальное значение частоты вывода сигнала, равное 1 кГц.

В разделе Trigger можно включить срабатывание триггеров по переднему или заднему фронту сигналов, а также использовать внутреннюю или внешнюю синхронизацию. На этом программирование генератора слов заканчивается.

## 2. Настройка логического анализатора

Логический анализатор (Logic Analyzer, рис. П.4.) необходим для анализа цифровых сигналов. Анализатор предназначен для отображения на его экране 16-разрядных кодовых последовательностей, подаваемых на вход одновременно из 16 точек схемы, а также отображения значений координат сигнала в местах расположения визиров в виде шестнадцатеричных чисел в окне, расположенном внизу экрана анализатора.

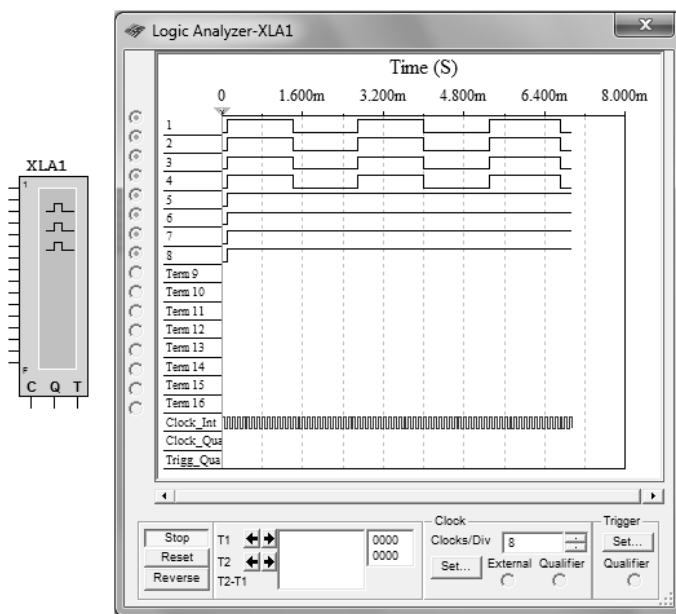


Рис. П.4. Символ и лицевая панель логического анализатора

Для настройки логического анализатора устанавливается частота дискретизации (нажимается кнопка Set... на панели настроек в разделе Clock) (рис. П.4), затем в открывшемся окне Clock Setup в разделе Clock Rate (рис. П.5.) вводится требуемая частота дискретизации. В этом же окне можно установить внешнюю либо внутреннюю синхронизацию в разделе Clock Source. При выборе внешней синхронизации к входу С необходимо подключить устройство, с которым логический анализатор будет синхронизироваться.

Для настройки триггеров необходимо воспользоваться окном Trigger Settings (Рис. П.6.), вызываемым с помощью кнопки Set... в разделе Trigger. Здесь можно установить срабатывание триггера по переднему, заднему или обоим фронтам сигналов (раздел Trigger Clock Edge, опции Positive, Negative и Both соответственно), а так же настроить шаблоны триггеров и их комбинацию (Раздел Trigger Patterns).



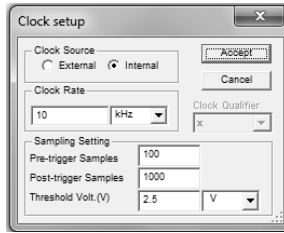


Рис. П.5. Окно настройки частоты дискретизации логического анализатора

Положением визирных линий логического анализатора, как и в осциллографе, можно управлять при помощи мыши или дискретно, кнопками с изображением стрелок на лавной панели логического анализатора.

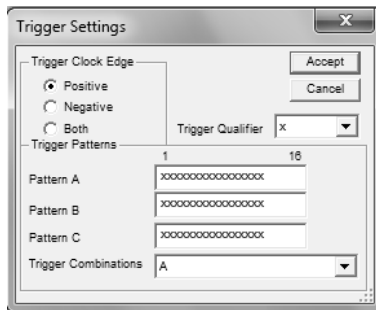


Рис. П.6. Окно настройки триггеров логического анализатора

### 3. Особенности работы функционального генератора

Прибор функциональный генератор (function generator, рис. П.7.) является источником напряжения, который может генерировать синусоидальные, пилообразные и прямоугольные импульсы. Прибор позволяет изменить форму сигнала, его частоту, амплитуду, коэффициент заполнения и постоянный сдвиг. Генератор позволяет воспроизвести сигналы с частотами от нескольких герц до аудио и радиочастотных. Он имеет три терминала-источника импульсов, общий центральный терминал определяет положение нуля.

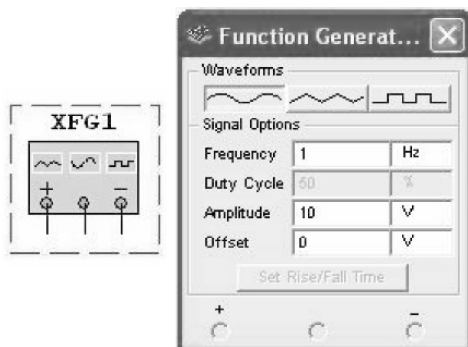


Рис. П.7. Символ и лицевая панель генератора сигналов

Настройка функционального генератора заключается в выборе следующих атрибутов: формы сигнала, его скважности, частоты и амплитуды. Для этого необходимо дважды щелкнуть мышью по символу прибора, затем в открывшемся окне настроек (рис. П.7.) выбрать форму сигнала, щелкнув на соответствующую пиктограмму в разделе Waveforms. В разделе Signal Options устанавливаются характеристики сигнала: его частота (Frequency), скважность в процентах (Duty Cycle), амплитуда в вольтах (Amplitude), смещение в вольтах (Offset).

*Учебное издание*

**Шегал Анна Айзиковна**

**ПРИМЕНЕНИЕ ПРОГРАММНОГО КОМПЛЕКСА  
MULTISIM ДЛЯ ПРОЕКТИРОВАНИЯ УСТРОЙСТВ  
НА МИКРОКОНТРОЛЛЕРАХ**

Редактор *О. В. Климова*  
Корректор *А. А. Загоруйко*  
Компьютерная верстка *Е. В. Суховой*

Подписано в печать 19.03.2014. Формат 60×90 1/16.  
Бумага писчая. Плоская печать. Усл. печ. л. 7,25.  
Уч.-изд. л. 6,4. Тираж 50 экз. Заказ № 92.

Издательство Уральского университета  
Редакционно-издательский отдел ИПЦ УрФУ  
620049, Екатеринбург, ул. С. Ковалевской, 5  
Тел.: + 7 (343) 375-48-25, 375-46-85, 374-19-41  
E-mail: [rio@urfu.ru](mailto:rio@urfu.ru)

Отпечатано в Издательско-полиграфическом центре УрФУ  
620075, Екатеринбург, ул. Тургенева, 4  
Тел. + 7 (343) 350-56-64, 350-90-13  
Факс + 7 (343) 358-93-06  
E-mail: [press-urfu@mail.ru](mailto:press-urfu@mail.ru)

*Для заметок*

