Министерство образования и науки Российской Федерации

Владивостокский государственный университет
экономики и сервиса

Н.Н. НОМОКОНОВА
В.Ю. ГАВРИЛОВ

ЦИФРОВЫЕ УСТРОЙСТВА И МИКРОПРОЦЕССОРЫ

Практикум

Владивосток
Издательство ВГУЭС
2005

ББК 32.85
Н 81

Рецензенты: Е.И. Убанкин, канд. тех. наук,
директор Центра технологий
дистанционного обучения открытого
института ДВГУ;
Ю.А. Левашов, доцент кафедры
электроники ВГУЭС

Номоконова Н.Н., Гаврилов В.Ю.

Н 81 ЦИФРОВЫЕ УСТРОЙСТВА И МИКРОПРОЦЕССОРЫ: Практикум. – Владивосток: Изд-во ВГУЭС, 2005. – 80 с.

Настоящий практикум предназначен для проведения лабораторных занятий и обеспечения самостоятельной работы студентов по дисциплине «Цифровые устройства и микропроцессоры».

В работе содержатся методические указания по выполнению восьми лабораторных работ. Первые пять работ посвящены вопросам изучения принципов построения, функционирования и использования цифровых устройств комбинационного и последовательностного типов в современных радиоэлектронных устройствах, в том числе компьютерной технике. Здесь же приведено описание технического (аппаратного) обеспечения по проведению лабораторных работ, а именно специализированного учебного стенда.

Далее приведены три лабораторные работы по применению программаторов как основы для создания цифровых управляющих систем и блоков. При подготовке к выполнению этих лабораторных работ студенты должны заранее самостоятельно изучить функциональное предназначение программируемого узла, а также написать программу его функционирования.

Приведен список литературных источников.

Для студентов специальностей 201500 «Бытовая радиоэлектронная аппаратура» и 201700 «Средства радиоэлектронной борьбы».

ББК 32.85

Печатается по решению РИСО ВГУЭС

ЛАБОРАТОРНАЯ РАБОТА № 1
Монтаж и исследование полного трехвходового дешифратора. Исследование дешифратора-демультиплексора

Введение

Дешифратором называется логическая схема, преобразующая поступающий на ее входы код числа в другой код [1].

Работа дешифратора на три входа (N) имеет $K = 2^N = 8$ выходов и описывается таблицами состояний (табл. 1.1 и 1.2).

Таблица 1.1

Таблица состояний дешифратора 3-8 (соответствует дешифратору с прямыми выходами)

Входы			Выходы							
X2	X1	X0	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

Таблица 1.2

Таблица состояний (соответствует дешифратору с инверсными выходами)

Входы			Выходы							
X2	X1	X0	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
0	0	0	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	0	1
0	1	0	1	1	1	1	1	0	1	1
0	1	1	1	1	1	1	0	1	1	1
1	0	0	1	1	1	0	1	1	1	1
1	0	1	1	1	0	1	1	1	1	1
1	1	0	1	0	1	1	1	1	1	1
1	1	1	0	1	1	1	1	1	1	1

Исходя из табл. 1.2, можно записать логические функции, реализуемые дешифратором с инверсными выходами:

$$\overline{Y0} = \overline{X2} \cdot \overline{X1} \cdot \overline{X0}\ ,\ \overline{Y1} = \overline{X2} \cdot \overline{X1} \cdot X0\ ,\ \overline{Y2} = \overline{X2} \cdot X1 \cdot \overline{X0}\ ,$$
$$\overline{Y3} = \overline{X2} \cdot X1 \cdot X0\ ,\ \overline{Y4} = X2 \cdot \overline{X1} \cdot \overline{X0}\ ,\ \overline{Y5} = X2 \cdot \overline{X1} \cdot X0\ ,$$
$$\overline{Y6} = X2 \cdot X1 \cdot \overline{X0}\ ,\ \overline{Y7} = X2 \cdot X1 \cdot X0$$

Исходя из табл. 1.1, можно записать логические функции, реализуемые дешифратором с прямыми выходами:

$$Y0 = \overline{X2} \cdot \overline{X1} \cdot \overline{X0}\ ,\ Y1 = \overline{X2} \cdot \overline{X1} \cdot X0\ ,\ Y2 = \overline{X2} \cdot X1 \cdot \overline{X0}\ ,\ Y3 = \overline{X2} \cdot X1 \cdot X0\ ,$$
$$Y4 = X2 \cdot \overline{X1} \cdot \overline{X0}\ ,\ Y5 = X2 \cdot \overline{X1} \cdot X0\ ,\ Y6 = X2 \cdot X1 \cdot \overline{X0}\ ,$$
$$\overline{Y7} = X2 \cdot X1 \cdot X0$$

По этим логическим функциям можно построить одноступенчатые или многоступенчатые дешифраторы. Функциональная схема одноступенчатого трехвходового дешифратора с прямыми выходами представлена на рис. 1.1. Одноступенчатые дешифраторы при разрядности дешифруемого кода, равной n двоичных разрядов, требуют наличия 2^n логических схем «И», каждая из которых имеет n входов. При больших значениях n ($n > 6$) одноступенчатые дешифраторы в плане аппаратной реализации неоптимальны, поэтому вместо них применяют многоступенчатые дешифраторы, принцип построения которых состоит в следующем. Все входы группируются. Затем выходные шины первой ступени группируются и становятся входами второй ступени и так до тех пор, пока на последней ступени не будут объединены выходы всех дешифраторов предпоследней ступени. Функциональная схема, двухступенчатого трехвходового дешифратора с прямыми выходами представлена на рис. 1.2.

Сравнивая функциональные схемы одноступенчатого (рис. 1.1) и многоступенчатого (рис. 1.2) дешифраторов можно отметить, что одноступенчатые дешифраторы являются более быстродействующими.

(Техническое описание и инструкцию по эксплуатации универсального стенда к лабораторным работам № 1–5 смотрите в ПРИЛОЖЕНИИ).

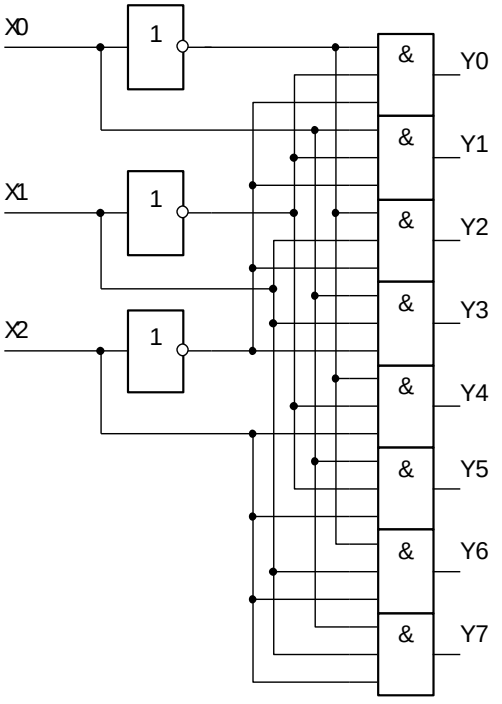


Рис. 1.1. Функциональная схема полного одноступенчатого дешифратора на 3 входа

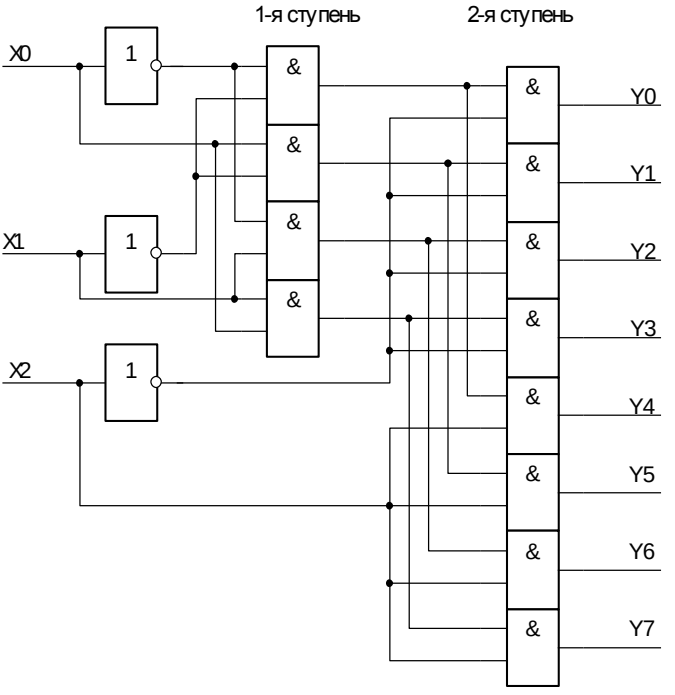


Рис. 1.2. Функциональная схема полного
двухступенчатого дешифратора на 3 входа

Задание 1
Монтаж и исследование полного трехвходового одноступенчатого дешифратора на ИС серии К155

1.1. Ознакомиться со схемой электрической принципиальной полного одноступенчатого дешифратора на 3 входа, представленного на рис. 1.3. Сравнить данную схему со схемой функциональной (рис. 1.1).

1.2. Составить таблицу соединений для схемы (рис. 1.3). Напомним рекомендации по составлению:

– начать составление таблицы с описания внешних соединений, идущих от органов управления (тумблеров, кнопок, переключателей);

– перейти к описанию выходов каждой ИС в порядке «слева направо», «сверху вниз».

1.3. Смонтировать дешифратор по таблице соединений.

1.4. Тестером проверить отсутствие короткого замыкания («КЗ») между шинами «+5В» и «земля». При отсутствии «КЗ» включить блок питания.

1.5. Проверить правильность функционирования дешифратора. Правильное функционирование описывается таблицей 1.1. Для схем на рис. 1.3 и рис. 1.4 светящийся светодиод говорит о том, что на соответствующем выходе дешифратора находится лог. «1».

Задание 2
Монтаж и исследование полного трехвходового многоступенчатого дешифратора на ИС серии К155

2.1. Ознакомиться со схемой электрической принципиальной полного многоступенчатого дешифратора на 3 входа, представленного на рис. 1.4. Сравнить данную схему со схемой функциональной (рис. 1.2).

2.2. Последовательно выполнить пункты 1.2–1.5 применительно к схеме на рис. 1.4.

2.3. Найти неисправность, внесенную в дешифратор преподавателем.

2.4. Выключить блок питания; демонтировать дешифратор.

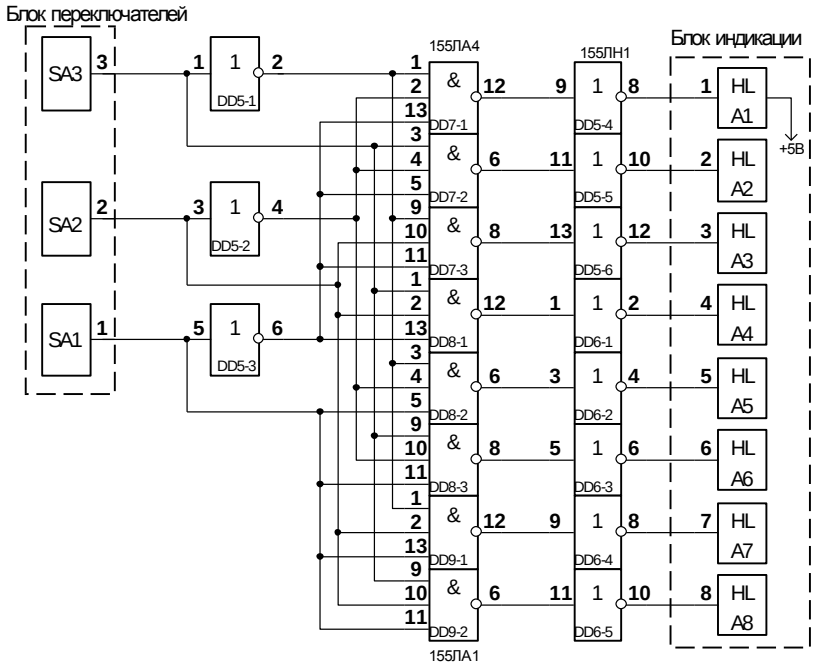


Рис. 1.3. Схема для исследования полного одноступенчатого дешифратора на 3 входа

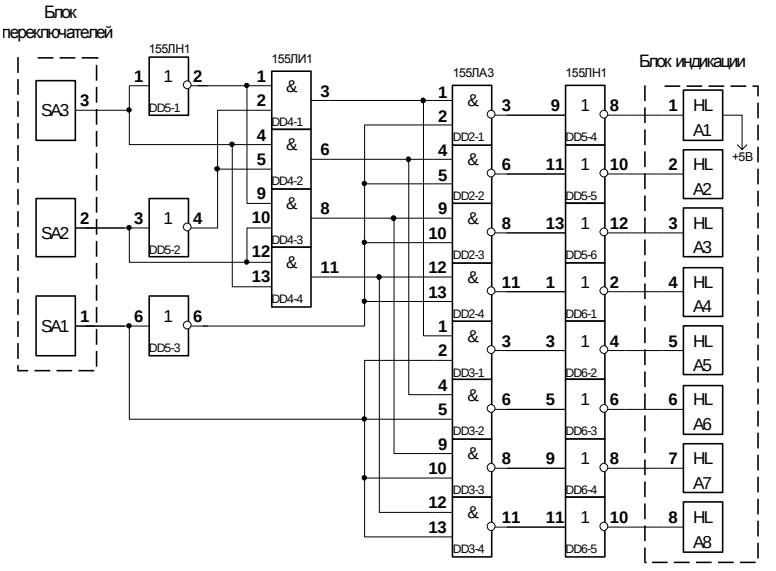


Рис. 1.4. Схема для исследования полного
двухступенчатого дешифратора на 3 входа

Задание 3
Исследование дешифратора-мультиплексора 2–4 и 3–8 на ИС К155ИД4

Рассмотренные выше дешифраторы построены на ИС малой степени интеграции. В настоящее время в различных устройствах цифровой техники используются дешифраторы на 3 или 4 входа, выполненные на одной ИС средней степени интеграции.

Исследуем работу такого дешифратора, построенного на микросхеме К155ИД4, имеющей следующее графическое представление:

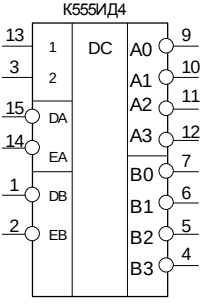


Рис. 1.5. Условное обозначение ИС К155ИД4

Микросхема К155ИД4 имеет два адресных входа: 2 и 1 (выводы 3 и 13). Один дешифратор имеет выходы А, другой – выходы В. В дешифраторе с выходами А используется отдельный стробирующий вход ЕА (вывод 14), в дешифраторе с выводами В – вход ЕВ (вывод 2). Для демультиплексирования на выходы А используется информационный выход DA (вывод 15), для демультиплексирования на выходы В используется информационный выход DB (вывод 1).

Таблица 1.3

Таблица работы дешифратора «В» *2 на 4* и демультиплексора *2 на 4* с инвертированием информации по выходу DB

Входы				Выходы			
2	1	EB	DB	B3	B2	B1	B0
X	X	1	X	1	1	1	1
0	0	0	1	1	1	1	0
0	1	0	1	1	1	0	1
1	0	0	1	1	0	1	1
1	1	0	1	0	1	1	1
X	X	X	0	1	1	1	1

Таблица 1.4

Таблица работы дешифратора «А» *2 на 4* и демультиплексора *2 на 4* с инвертированием информации по выходу DA

Входы				Выходы			
2	1	EA	DA	A3	A2	A1	A0
X	X	1	X	1	1	1	1
0	0	0	0	1	1	1	0
0	1	0	0	1	1	0	1
1	0	0	0	1	0	1	1
1	1	0	0	0	1	1	1
X	X	X	1	1	1	1	1

Если объединить входы DA (вывод 15) и DB (вывод 1), то микросхема будет работать как дешифратор 3 на 8.

Контрольные вопросы

1. Что такое дешифратор?
2. В каких случаях схемного применения вы отдадите предпочтение одноступенчатому и многоступенчатому дешифратору?
5. Что такое демультиплексирование?

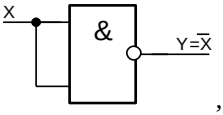
ЛАБОРАТОРНАЯ РАБОТА № 2

Исследование работы логических схем на ИС серии 155

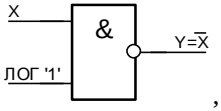
Задание 1

Начертить и смонтировать на базе ИС К155ЛА3 (2И-НЕ) схему, реализующую логическую функцию вида У=Х1VХ2. Для смонтированной схемы составить таблицу истинности [1].

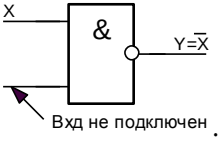
С помощью схемы 2И-НЕ реализовать функцию инверсии можно

так: 

,

или так: 

,

или так: 

.

В третьем случае учтена специфика ТТЛ – логики, а именно: интегральная схема воспринимает как лог. «1» информацию на неподключенном входе. Однако в этом случае увеличивается время задержки логического элемента.

Задание 2

Начертить и смонтировать на базе ИС К155ЛА3 схему, реализующую логическую функцию вида У=Х1*Х2*Х3

Для смонтированной схемы составить таблицу истинности.

Схема, реализующая названную логическую функцию, может быть построена так:

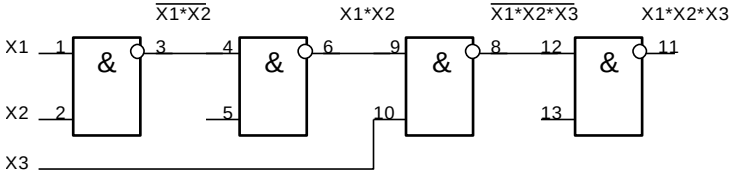
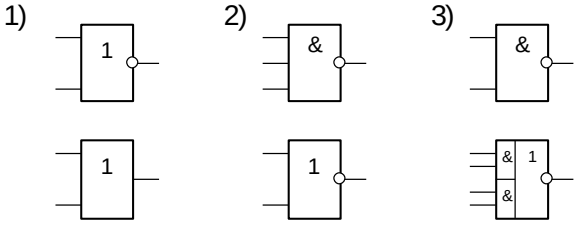


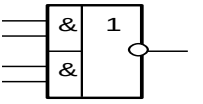
Рис. 2.1. Схема, реализующая логическую функцию У=Х1*Х2*Х3

На схемах 2И-НЕ можно реализовать любую логическую функцию. Однако аппаратное решение показанной функции может быть очень громоздким.

Контрольные вопросы

1. Какую логическую функцию реализуют следующие схемы:



2. При каких входных наборах на выходе схемы  будет лог. «0», лог. «1»?

ЛАБОРАТОРНАЯ РАБОТА № 3
Исследование RS-триггера

Триггером называется устройство с памятью, имеющее два устойчивых состояния [1].

Задание 1
Монтаж и исследование асинхронного RS-триггера на ИС серии К155

На рис. 3.1 и в табл. 3.1 приведены соответственно схема функциональная и таблица истинности асинхронного RS-триггера.

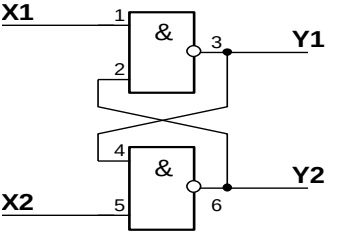


Рис. 3.1. Функциональная схема RS-триггера

Одновременная подача двух нулей на X1 и X2 запрещена, так как неизвестно, в какое состояние установится триггер после их одновременного снятия.

Таблица 3.1

Таблица истинности

X1	X2	Y1	Y2
0	1	1	0
1	0	0	1
1	1	прежнее состояние	
0	0	-	-

На схемах RS-триггер, установка которого осуществляется логическим «0», изображается следующим образом:

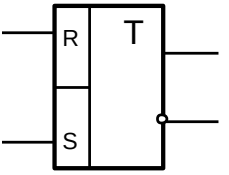


Рис. 3.2. Схема RS-триггера

Входы R и S называются установочными, а выходы – прямым и инверсным. Состояние триггера определяется по его прямому выходу. Таким образом, если на вход R (рис.3.2) поступит лог. «0», то триггер установится в «0». Если на вход S поступит лог. «0», то триггер установится в «1».

Задание

1.1. Начертить схему электрическую принципиальную, предназначенную для исследования RS-триггера на ИС К155ЛА3. Схема должна учитывать:

а) управление триггером от тумблеров или переключателей из числа SA1-SA10;

б) подключение выхода триггера, определенного как единичный, к блоку индикации 1 к контакту А1;

в) подключение выхода триггера, определенного как нулевой, к блоку индикации 1 к контакту В1.

В процессе разработки схемы ориентироваться на приложение 1 к лабораторной работе №6.

1.2. Определить номера и число используемых расширителей.

1.3. Написать таблицу соединений.

1.4. Смонтировать схему по таблице соединений.

1.5. Составить для RS-триггера таблицу истинности.

Задание 2
Монтаж и исследование синхронного RS-триггера на ИС серии К155

Рассмотрим синхронный RS-триггер.

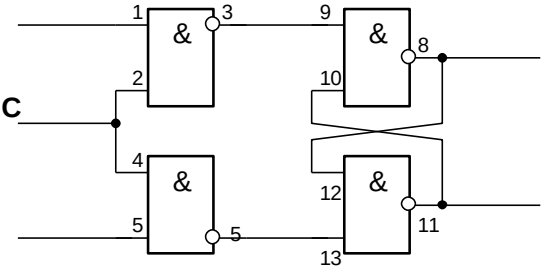


Рис. 3.3. Функциональная схема синхронного RS-триггера

Задание

2.1. Начертить схему электрическую принципиальную, предназначенную для исследования синхронного RS-триггера (рис. 3.3), построенного на ИС К155ЛА3.

Схема должна учитывать:

а) подачу синхроимпульса от блока формирователей сигналов без дребезга, от одного из контактов К1-К4, электрически связанных с кнопками или переключателями SB1-SB4 соответственно:

б) задание информации от тумблеров или переключателей SA1-SA10;

в) подключение выхода триггера, определенного как единичный, к блоку индикации 1 к контакту А1;

г) подключение выхода триггера, определенного как нулевой, к блоку индикации 1 к контакту В1.

2.2. Определить номера и число используемых расширителей.

2.3. Написать таблицу соединений.

2.4. Составить таблицу истинности.

3.8. Выключить блок питания. Демонтировать схему.

Контрольные вопросы

1. Что такое триггер?

2. Какова логическая структура асинхронного и синхронного RS-триггера?

ЛАБОРАТОРНАЯ РАБОТА № 4

Исследование работы комбинированного RS- и D-триггера. Исследование работы комбинированного RS- и JK-триггера.

Введение

Рассмотрим D-триггер. Его графическое изображение имеет следующий вид:

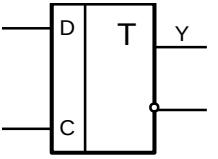


Рис. 4.1. Графическое изображение D-триггера

Таблица 4.1

Таблица истинности D-триггера

D	C	Y
x	0	старое состояние
0	⎍‾	0
x	1	старое состояние
1	⎍‾	1

Примечание. Символ «x» означает неопределенное состояние; символ «⎍‾» означает переход от низкого уровня к высокому.

Отметим две важных особенности:

1) D-триггер не имеет запрещенных входных комбинаций;

2) занесение информации в D-триггер происходит по положительному перепаду напряжения на C-входе.

Поясним сказанное следующей временной диаграммой:

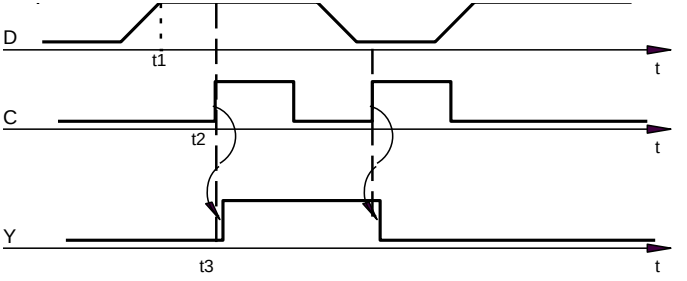


Рис. 4.2. Временная диаграмма

До момента времени t3 D-триггер находился в нулевом состоянии, т.е. на его единичном выходе Y был лог. «0». В момент времени t1 на D-входе появилась лог. «1», однако состояние триггера не изменилось (на выходе У по-прежнему – лог. «0»). В момент времени t2 на синхровходе (С-вход) появился положительный перепад. Так как на D-входе к этому времени находится лог. «1», то D-триггер устанавливается в единичное состояние.

Рассмотрим JK-триггер. Его графическое изображение имеет следующий вид:

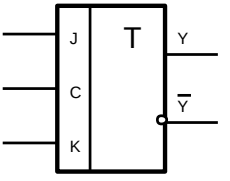


Рис. 4.3. Графическое изображение JK-триггера

Таблица 4.2

Таблица истинности для JK-триггера

J	K	C	Y
x	x	0	старое состояние
x	x	1	старое состояние
0	0	⎍	старое состояние
0	1	⎍	0
1	0	⎍	1
1	1	⎍	состояние, инверсное старому

Примечание. Символ «⎍» означает переход от высокого уровня к низкому.

Отметим две важных особенности:

1) JK-триггер не имеет запрещенных входных комбинаций;

2) занесение информации в JK-триггер происходит по отрицательному перепаду напряжения на C-входе.

Реализация сказанного представлена временной диаграммой рис. 4.4.

В реальных микросхемах имеет место комбинация RS-триггера с JK-триггером.

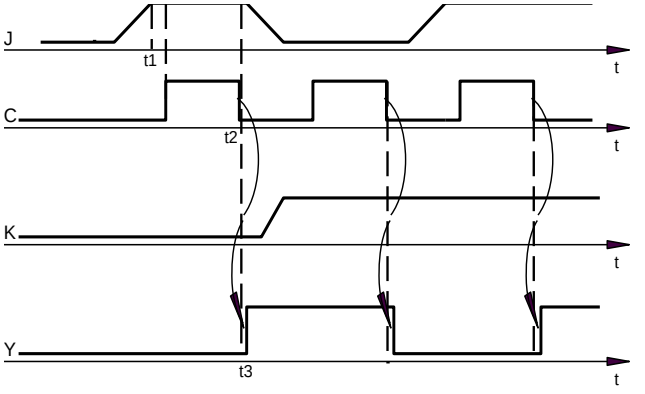


Рис. 4.4. Временная диаграмма

Задание 1
Исследование работы комбинированного RS- и D-триггера

1.1. Написать таблицу соединений.

1.2. Смонтировать схему.

1.3. Составить таблицу истинности RS-и D-триггера в виде:

S	R	D	C	Y

Примечание. Входы S и R имеют приоритет над входами D и C.

1.4. Выключить блок питания. Демонтировать схему.

Задание 2
Исследование работы комбинированного RS- и JK-триггера

2.1. Начертить схему электрическую принципиальную, предназначенную для исследования комбинированного RS- и JK-триггера на ИС К155ТБ1. Схема должна учитывать:

1) задание информации на входы S, R и J, K от тумблеров (переключателей) SA1-SA10 блока переключателей:

2) подачу синхроимпульса от одной из кнопок (переключателей) SB1-SB4 (контакты $\overline{K1}-\overline{K4}$) блока формирователей сигналов без дребезга;

3) подключение единичного выхода триггера к контакту А1 блока индикации 1;

4) подключение нулевого (инверсного) выхода триггера к контакту Б1 блока индикации 1.

2.2. Написать таблицу соединений.

2.3. Смонтировать схему.

2.4. Составить таблицу истинности для исследуемого комбинированного RS- и JK-триггера в виде:

S	R	J	K	C	Y

Примечание. Входы S и R имеют приоритет над входами J, C и K.

2.5. Выключить блок питания. Демонтировать схему.

Контрольные вопросы

1. В чем состоит особенность триггера со счетным входом?

2. Расскажите работу D-триггера по логической структуре комбинированного RS- и D-триггера.

3. В каком из двух триггеров (D или JK) осуществляется занесение информации по отрицательному перепаду на C-входе?

ЛАБОРАТОРНАЯ РАБОТА № 5
Монтаж и исследование 4-разрядного реверсивного счётчика

Введение

Счетчик – это узел, предназначенный для счета и хранения поступившего числа импульсов. Счетчики широко используются в устройствах управления ЭВМ, в различных автоматах, электронных часах и т.п. Счетчики строятся на триггерах со счетным входом. Вы уже знаете, что триггер со счетным входом (Т-триггер) можно построить на основе D-триггере, соединив нулевой его выход с D-входом. JK-триггер тоже может работать как триггер со счетным входом, так как если на входах J и K – лог. «1», то по отрицательному перепаду на С-входе триггер устанавливается в состояние, инверсное старому. То есть, триггер со счетным входом является сумматором по mod2. Действительно, если Т-триггер был установлен в «0», то с приходом синхроимпульса он устанавливается в «1». Этот факт запишется так: 0+1=1. Если Т-триггер был установлен в «1», то с приходом синхроимпульса он устанавливается в «0», т.е. 1+1=0.

Число Т-триггеров, входящих в состав счетчика и используемых непосредственно для подсчета и хранения числа импульсов, представляет разрядность счетчика. Разрядность определяет число N устойчивых состояний, в которых может находиться счетчик. Счетчики делятся на двоичные (для них $N=2^r$, где r – разрядность счетчика) и с произвольным коэффициентом пересчета ($N=2^2$). Наибольшее распространение получили двоичные и десятичные (N=10) счетчики.

По используемому принципу пересчета счетчики разделяются на асинхронные и синхронные.

Задание 1
Монтаж счетчика на ИС К155ТМ2

1.1. Ознакомиться со схемой электрической принципиальной 4-разрядного синхронного реверсивного счетчика, представленного на рис. 5.1.

1.2. Составить таблицу соединений для схемы, представленной на рис. 51.

Рекомендации по составлению:

– начать составление таблицы с: описания внешних соединений, идущих от органов управления (тумблеров, кнопок, переключателей); далее перейти к описанию выходов каждой ИС в порядке «слева направо», «сверху вниз».

1.3. Смонтировать счетчик по таблице соединений.

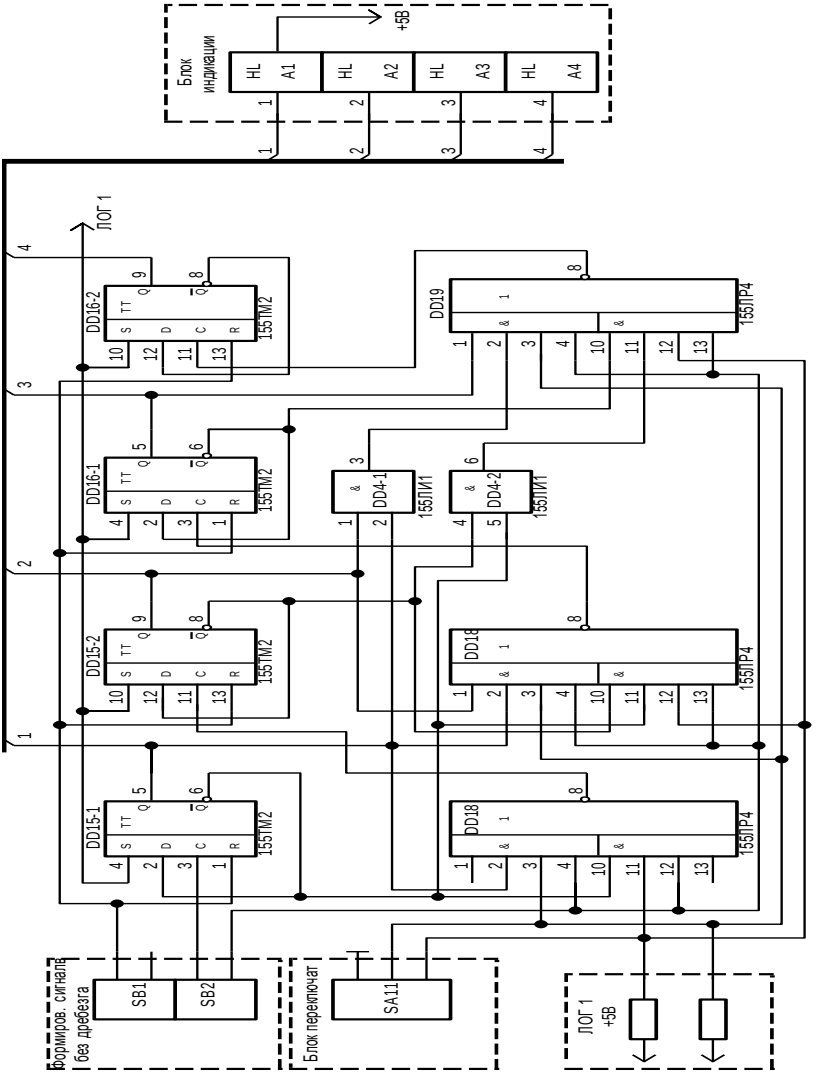


Рис. 5.1. Схема для исследования 4-разряднго реверсивного счетчика на ИС К155ТМ2

Задание 2
Исследование работы счетчика на ИС К155ТМ2

2.1. Тестером проверить отсутствие короткого замыкания («КЗ») между шинами «+5В» и «земля». При отсутствии «КЗ» включить блок питания.

2.2. Проверить правильность функционирования счетчика. Для этого надлежит выполнить следующее.

2.2.1. По R-входу установить счетчик в «0».

2.2.2. Выставить режим «суммирование».

2.2.3. Формируя тактовые импульсы, проверить по светодиодам правильность выполнения прямого счета.

2.2.4. Выставить режим «вычитание».

2.2.3. Формируя тактовые импульсы, проверить по светодиодам правильность выполнения обратного счета

2.3. Объяснить работу счетчика в режиме «суммирование» и «вычитание».

2.4. Найти неисправность, внесенную в счетчик преподавателем.

2.5. Выключить блок питания.

2.6. Оценить быстродействие счетчика в режиме «суммирование».

Так как исследуемый нами счетчик является синхронным, т.е. все его разряды переключаются практически одновременно от одного счетного импульса, то быстродействие такого счетчика определяется временем установки каждого разряда. Строго говоря, исходя из схемы рис. 5.1, время переключения разрядов 2-го, 3-го и 4-го будет больше, чем время переключения 1-го разряда. Это объясняется тем, что на триггер 1-го разряда тактовый импульс поступает непосредственно, а на триггеры 2-го, 3-го и 4-го разрядов тактовый импульс поступает через схему 2-4И-2ИЛИ-НЕ и, следовательно, задерживается на ней. Поэтому быстродействие счетчика будем оценивать по одному из разрядов: 2-му или 3-му, или 4-му. Очевидно, что быстродействие характеризуется задержкой между появлением сигнала на выходе одного из перечисленных разрядов и положительным перепадом тактового импульса.

Измерение задержки будем выполнять с помощью осциллографа. Для того, чтобы сделать видимым на осциллографе без памяти процесс пересчета, необходимо, чтобы этот процесс был циклически повторяющимся. С этой целью подключим к счетному входу счетчика выход генератора. Технически это следует выполнить так:

– вход «зап. 1» генератора соединить с лог. «1»; выход «вых. 1» генератора и вход 1 осциллографа подключить к любому свободному расширителю;

– поднять конец провода с контакта К2 блока формирователей сигналов и подключить этот конец к названному выше расширителю;

– соединить этот расширитель с входом любого инвертора на ИС К155ЛН1;

– поднять конец провода с контакта К2 блока формирователей сигналов и подключить этот конец к выходу выбранного инвертора;

– вход 2 осциллографа подключить к единичному выходу триггера 2-го или 3-го разряда счетчика.

2.7. Включить блок питания.

2.8. Постараться, по возможности, более точно измерить задержку между положительным перепадом тактового импульса и появлением сигнала на выходе выбранного разряда.

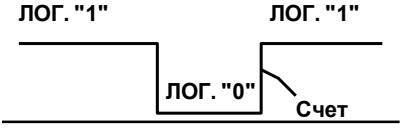
2.9. Выключить блок питания. Демонтировать схему.

Задание 3
Исследование работы счетчика на ИС К155ИЕ7

Смонтированный и исследованный нами счетчик строился на ИС малой степени интеграции.

Исследуем работу 4-х-разрядного реверсивного счетчика, выполненного на одной ИС К155ИЕ7 средней степени интеграции. Названный счетчик представлен на рис.2.

Счетчик на ИС К155ИЕ7 является синхронным, т.е. у него все триггеры переключаются одновременно от одного счетного импульса. Счетный разряд построен на основе типового JK-триггера. Направление счета определяется тем, на какой из счетных входов («+1» или «–1») будет подан импульс следующего вида:



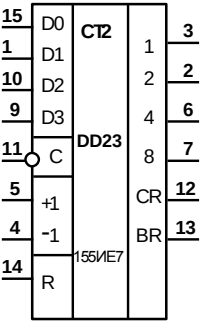
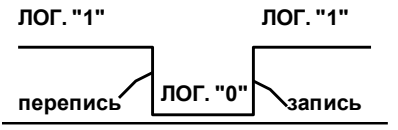


Рис. 5.2. Условное обозначение ИС К155ИЕ7

По положительному перепаду этого импульса выполняется счет. В это время на другом счетном входе должен быть высокий уровень напряжения, т.е. лог. «1».

Входы D1-D8 являются информационными. Они совместно с входом С используются для записи в счетчик начальной информации, т.е. исходного числа. Процесс записи начальной информации протекает следующим образом в зависимости от входа С:



По отрицательному перепаду исходное число с информационных входов D1-D8 передается на выходы счетчика. По положительному перепаду исходное число записывается в триггеры счетчика.

Вход R предназначен для установки счетчика в «0», т.е. в код «0000». Установка в «0» выполняется при подаче на R-вход высокого уровня напряжения (лог. «1») независимо от состояний входов D1-D8 и С.

Выходы «CR» и «BR» являются выходами прямого и обратного переноса соответственно. Они используются для построения счетчиков с разрядностью, большей четырех. При этом выход «CR» подключается ко входу прямого счета «+1» следующего каскада, а выход «BR» – ко входу обратного счета «–1» этого каскада.

Все описанные выше особенности ИС К155ИЕ7 отражены в ее графическом обозначении. Действительно, из графического обозначения следует, что:

– перед нами счетчик, так как в изображении имеется соответствующее обозначение – «СТ2»;

– стробирование (вход С), а также переносы «CR» и «BR» выполняются сигналом низкого уровня (лог. «0»), на графике о полярности сигналов говорит наличие «кружочка»;

– установка счетчика в «0» (вход R), а также задание режимов работы «+1» и «–1» осуществляется сигналом высокого уровня (лог. «1»).

Детально отразить особенности реверсивного счетчика на ИС К155ИЕ7 можно с помощью временной диаграммы, представленной на рис. 5.3.

Итак, приступим к исследованию работы 4-разрядного реверсивного счетчика на ИС К155ИЕ7.

Задание

3.1. Написать таблицу соединений для схемы (рис. 5.4), предназначенной для исследований ИС К155ИЕ7. При написании таблицы учесть, что данная схема будет модифицироваться, поэтому каждый из выходов

счетчика (выводы 3, 2, 6, 7) подключить через расширитель. Обратить внимание на графическое исполнение схемы, в частности, на изображение счетчика, на использование жгута, в котором одинаковые номера провода указывают на наличие электрического соединения (связи) между двумя или более точками схемы.

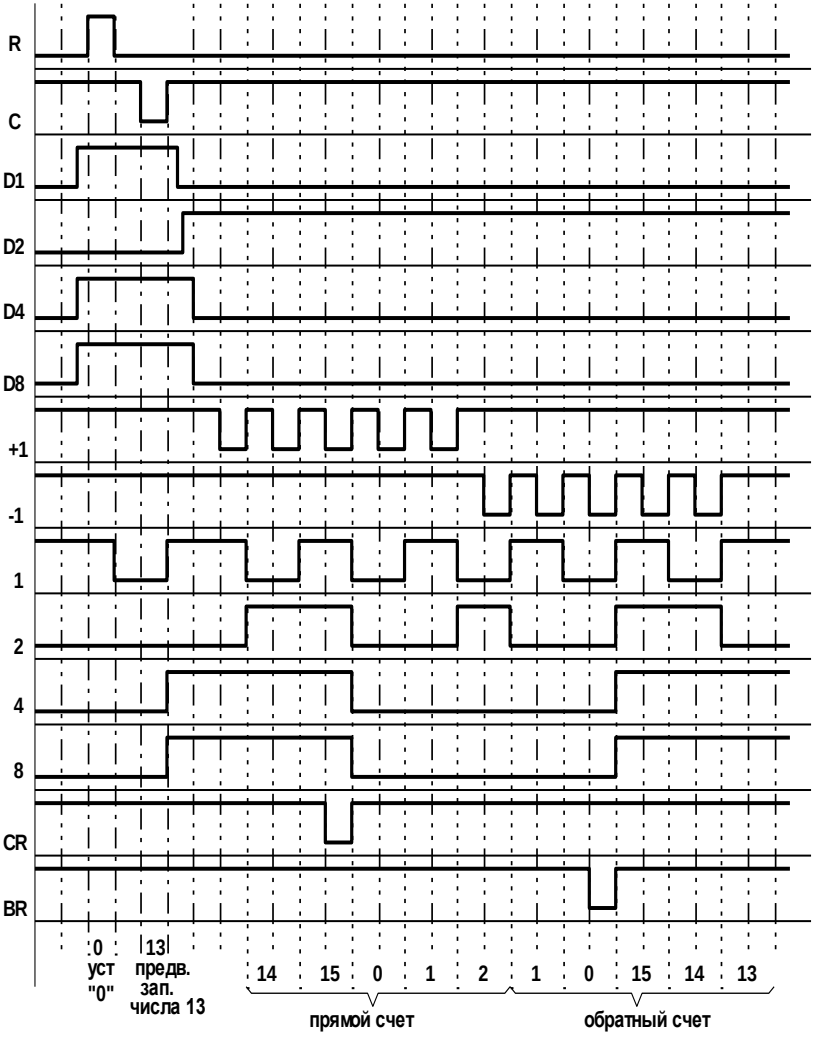


Рис. 5.3. Временная диаграмма

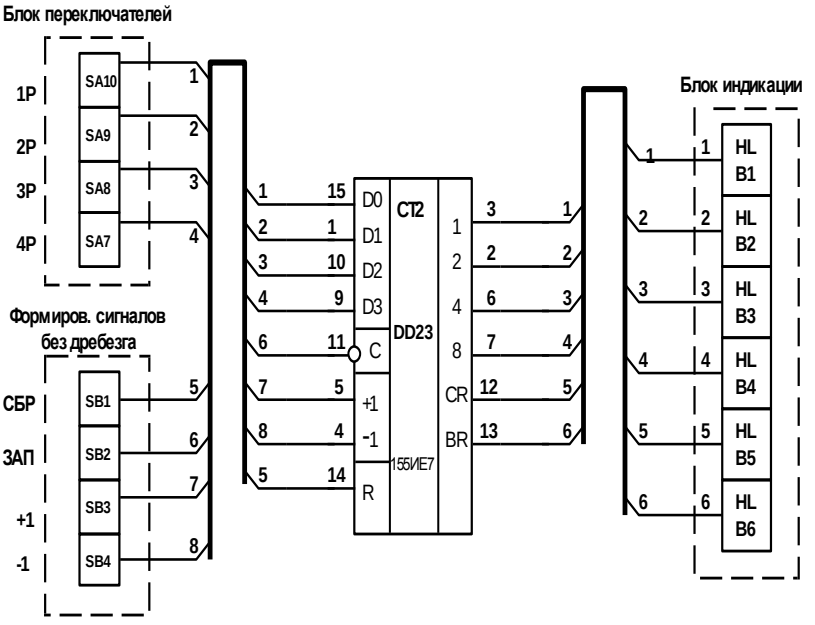


Рис. 5.4. Схема исследования ИС К155ИЕ7

3.3. Тестером проверить отсутствие короткого замыкания ("КЗ") между шинами «+5В» и «земля». При отсутствии «КЗ» включить блок питания.

3.4. По R-входу установить счетчик в «0».

3.5. По С- и D-входам записать в счетчик число «13», проверить работу счетчика по временной диаграмме (рис. 5.3).

3.6. Выключить блок питания.

3.7. Исследовать работу счетчика при суммировании в динамике, т.е. подключив его вход «+1» к выходу генератора. Технически это следует выполнить так:

– поднять конец провода с контакта КЗ и подключить его к любому свободному расширителю;

– к этому же расширителю подключить «Вых. 1» генератора и вход 1 осциллографа;

– вход «зап. 1» соединить с лог. «1».

Включить блок питания. Последовательно подключая вход 2 осциллографа к расширителям, соединенным с единичными выходами триггеров первого (1), второго (2), третьего (4) и четвертого (8) разрядов

счетчика, зарисовать осциллограммы сигналов на выходах счетчика в такой последовательности:

3.8. Выключить блок питания.

3.9. На базе данного счетчика, имеющего шестнадцать устойчивых состояний, сконструировать суммирующий десятичный счетчик. Как это сделать? Очевидно, следует по десятому тактовому импульсу установить счетчик в «0». Для этого надо распознать на счетчике число 10, представляемое в двоичной системе кодом «1010». Распознавание можно выполнить с помощью следующей функциональной схемы:

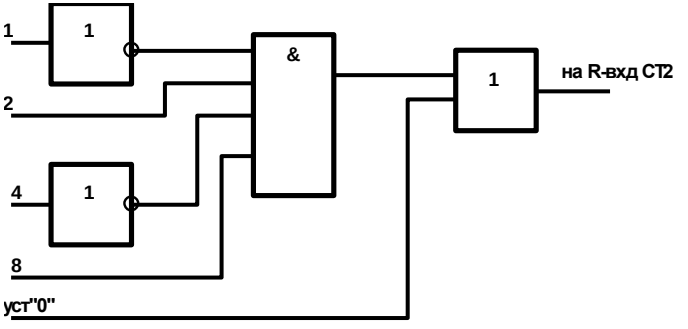
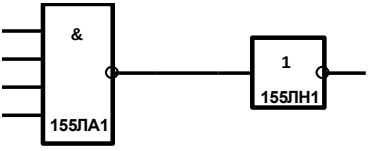


Рис. 5.5. Функциональная схема распознавания кода «1010» на счетчике

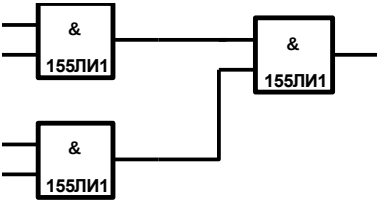
Действительно, как только на счетчике зафиксируется число 10, которому соответствует двоичный код «1010», то на выходе схемы 4И

появится сигнал, который через схему 2ИЛИ установит по входу счетчик в «0».

3.9.1. Используя элементную базу стенда, начертить схему электрическую принципиальную, реализующую логику работы исследуемой схемы (рис. 5.4). Основная альтернатива – в реализации функции 4И. В составе стенда нет ИС, непосредственно реализующей такую логику. Значит, надо комбинировать. Например, так:



Или так:



Можно выбрать любой вариант.

В схеме электрической принципиальной каждый логический элемент должен содержать номер позиции той ИС, которая реализует стандартную логическую функцию. Кроме того, выводы каждого логического элемента должны нумероваться в соответствии с номерами выводов той ИС, которая реализует заданную логическую функцию.

3.9.2. По схеме электрической принципиальной написать таблицу соединений.

3.9.3. Смонтировать схему по таблице соединений.

3.9.4. Тестером проверить отсутствие короткого замыкания («КЗ») между шинами «+5В» и «земля». При отсутствии «КЗ» включить блок питания.

3.9.5. Проверить работу суммирующего десятичного счетчика, построенного на ИС К155ИЕ7.

3.9.6. Выключить блок питания. Демонтировать схему.

Контрольные вопросы

1. Что такое счетчик? Что понимается под коэффициентом пересчета?

2. Чем отличается реверсивный счетчик от суммирующего?

3. Нарисовать схему счетчика по mod2.

ЛАБОРАТОРНАЯ РАБОТА № 6
Программаторы и системы программирования

1. Общетеоретические сведения

Как было указано во введении, целью лабораторной работы является изучение программно-аппаратного комплекса для обеспечения достоверности записи информации в большую интегральную схему, с учетом заданных условий и ограничений.

Под обеспечением достоверности записи понимается запись информации в контроллер с определенной точностью, способствующей правильному функционированию микросхемы и в будущем устройства в целом, которое реализовано на заданном контроллере.

Кроме этого, в понятие достоверности записи информации входят общепринятые задачи:

1. Обеспечение требуемого быстродействия обработки данных.
2. Выбор способа программирования и тестирования.
3. Выбор режима обмена данными между аппаратной частью и программной структурой.

Под обработкой информации подразумевается: правильный выбор системы программирования и программатора для определенного типа контроллера, выявление ошибок при обмене данными и передача в персональный компьютер содержимого того блока, в котором находится предполагаемая ошибка, или порта контроллера, в котором произошел сбой.

1.1. Классификация программаторов

В огромном многообразии изделий электронной техники семейство программируемых микросхем занимает особое место. В него входят хорошо известные программируемые постоянные запоминающие устройства (ППЗУ), однокристальные микро-ЭВМ (ОЭВМ), программируемые логические матрицы (ПЛМ), программируемые логические интегральные микросхемы (ПЛИС), микросхемы FLASH-памяти. Трудно представить современное цифровое устройство, в котором не использовались бы представители этого семейства. Так, например, для автоматизации процесса управления микропроцессорными системами используются программы, как правило, хранящиеся в ППЗУ; на базе ОЭВМ все чаще разрабатываются различные контроллеры; ПЛМ и ПЛИС вытесняют традиционные логические микросхемы малой и средней степени интеграции; ППЗУ широко применяют в персональных компьютерах: в них хранятся BIOS, шрифты знакогенераторов видеоадаптера и принтера.

Специфика программируемых микросхем состоит в том, что их необходимо определенным образом запрограммировать. Делают это с помощью специальных устройств – программаторов.

Современные профессиональные программаторы должны отвечать следующим требованиям:

- программировать возможно более разнообразные БИС в корпусах различных типов;
- обеспечивать наращивание номенклатуры программируемых БИС без существенного увеличения аппаратных средств и стоимости прибора;
- работать с входными файлами стандартных типов (*.hex, *.bin, *.jed и т. д.);
- иметь удобный интерфейс управляющей программы;
- верифицировать запрограммированную микросхему на соответствие кода входному файлу, в том числе при изменении напряжения питания микросхемы в рабочем диапазоне;
- иметь высококачественные розетки под корпуса различных типов;
- обеспечивать защиту от неправильной установки корпуса микросхемы в розетку.

Программаторы делят на автономные и неавтономные, внутренние и внешние, специализированные и универсальные. Автономный программатор может работать самостоятельно, неавтономный управляется компьютером, к которому его можно подключить через стандартный порт – последовательный или параллельный (в этом случае программатор внешний), либо, если программатор выполнен в виде платы расширения компьютера, установить в корпус системного блока (такой программатор называют внутренним). Универсальные программаторы рассчитаны на работу с микросхемами различных типов, а специализированные – только с микросхемами вполне определенного типа [6].

Определить, какой программатор необходим, можно только исходя из решаемых задач. Автономные программаторы, не имеющие связи с компьютером, можно использовать только как копировщики микросхем, и поэтому они вряд ли подойдут разработчикам электронной аппаратуры. Внутренние программаторы работают, как правило, намного быстрее, чем внешние, но их лучше использовать в специально выделенном компьютере, тогда как внешний программатор можно быстро и просто подключить к любому компьютеру. Наконец, понятно, что чем больше микросхем обслуживает программатор, тем лучше, но, во-первых, не существует программатора, который мог бы выручить во всех случаях (ведь разработчики микросхем ППЗУ постоянно пополняют список своих изделий), а во-вторых, такие программаторы существенно дороже. Поэтому, если известно, с какими микросхемами предстоит работать, вполне подойдет и специализированный программатор.

Перечень программируемых интегральных схем в мире просто гигантский – это микросхемы памяти, как с параллельным, так и с последовательным доступом информации (EPROM, EEPROM, FLASH); микроконтроллеры с внутренней памятью команд и данных; микросхемы программируемой логики (PLD). Причем, перечень таких изделий с каждым годом стремительно растет, имея тенденцию к усложнению изделий и к увеличению их гибкости. С другой стороны, как ответ на потребность использования этих микросхем, рынок наполняется большим количеством программаторов.

Рассмотрим классификацию программаторов по функциональным возможностям. Условно их можно подразделить на такие группы:

- программаторы, программирующие микросхемы памяти (EPROM, EEPROM, FLASH);
- программаторы, программирующие микросхемы памяти (EPROM, EEPROM, FLASH) и внутреннюю память микроконтроллеров;
- программаторы, программирующие микросхемы памяти (EPROM, EEPROM, FLASH), внутреннюю память микроконтроллеров, микросхемы программируемой логики (PLD);
- универсальные программаторы-тестеры.

Данную классификацию можно считать достаточно условной, жестких границ между программаторами разных групп не существует. Программаторы первой и второй групп – наиболее простые и дешевые устройства. Программаторы третьей группы, обычно, аппаратно значительно более сложны и стоимость их, соответственно, более высокая. Это объясняется, в частности, особенностью работы с устройствами программируемой логики.

Последняя функциональная группа программаторов – универсальные программаторы – наиболее сложные и дорогие устройства, но способные работать с очень большим перечнем микросхем.

Требование универсальности является важным критерием (разумеется, при соблюдении режимов программирования, указанных в документации на микросхемы). Оно связано с тем, что современные разработчики-профессионалы использует всю гамму программируемых БИС для получения максимально высоких характеристик изделия, работая в условиях сильной конкуренции, жестких требований по срокам разработки и стоимости изделия. Складывается набор функционально-топологических и программных модулей, а конкретный контроллер быстро «собирается» с использованием программируемых БИС. Заметим, что универсальность в данном случае подразумевает программирование полного набора важнейших типов программируемых микросхем, а не длинный перечень однотипных изделий, выпускаемых разными производителями. В такой набор входят:

- БИС микроконтроллеров с репрограммируемой и flash-памятью;

- микросхемы памяти с ультрафиолетовым стиранием;
- микросхемы памяти с электрическим репрограммированием;
- микросхемы flash-памяти;
- микросхемы памяти, программируемые пережиганием перемычек;
- БИС программируемой логики – ПЛИС.

Возможность наращивания функций прибора всегда полезна, но сейчас для программаторов это жизненно необходимо, поскольку в современной микроэлектронике происходит непрерывная замена версий уже существующих типов микросхем и появление новых семейств.

Структура современных программаторов вполне определилась. Большинство из них реализуется либо в виде периферийного устройства персонального компьютера, подключаемого через один из внешних интерфейсных разъемов (RS-232, Centronix), либо в виде платы расширения, размещаемой внутри компьютера, с выносом коммутирующего узла через гибкий кабель. Интерфейс с пользователем, загрузку рабочего файла, конфигурацию аппаратных средств программатора выполняет управляющая программа, функционирующая в среде операционной системы инструментального компьютера [7].

1.2. Аппаратное устройство программаторов и варианты сопряжения с компьютером

В первую очередь рассмотрим ту деталь программатора, с которой придется взаимодействовать больше всего – это колодка, куда помещается программируемая микросхема. Эта одна из самых важных деталей программатора, от качества и надежности которой зависит способность программатора выполнять свои функции. Любой программатор вне зависимости от его сложности, стоимости и функциональных возможностей обязательно должен быть снабжен специальной тестовой колодкой, обеспечивающей многократный надежный контакт с программируемой микросхемой.

Заглянем внутрь программаторов и в общих чертах попытаемся понять, чем же они отличаются друг от друга. Принципиально существует две концепции построения программаторов. Первая, и наиболее очевидная, заключается в построении программаторов на базе массива универсальных аппаратных драйверов. Универсальные драйверы подводятся к выводам тестовой колодки и должны удовлетворять ряду специфических аппаратных требований по программированию микросхем. В перечень таких требований входят: способность подавать и считывать логические уровни, способность подавать сложные тактовые последовательности, способность подводить напряжение в диапазоне 0...27 В с точностью 0,1 В и так далее. Удовлетворение всем этим требованиям приводит к колоссальным аппаратным затратам и избыточности всего устройства в целом. Количество драйверов универсального программатора должно соответствовать количеству выводов тестовой колодки,

например, 40 драйверов для колодки DIP-40 или 84 драйвера для колодки LCC-84. В результате, устройство становится очень сложным и дорогостоящим, но при этом – универсальным. Имея 40 универсальных драйверов и универсальную тестовую колодку DIP-40 можно с уверенностью сказать, что удастся поддержать все существующие, а также любые новые, микросхемы в корпусе DIP (с числом выводов до 40) без дополнительных адаптеров. Именно по такой схеме строятся дорогие универсальные программаторы.

Вторая концепция заключается в том, что аппаратура программатора оптимизируется под предполагаемый перечень поддерживаемых микросхем.

В процессе производства программаторов разработана и внедрена компромиссная концепция построения программаторов – универсальный драйвер разбивается на два функциональных блока: универсальный логический драйвер и устройство коммутации «высокого» напряжения. Такая архитектура программатора позволила в значительной степени сохранить преимущества универсального драйвера и существенно сократить аппаратные затраты.

1.2.1. Варианты подключения к компьютеру

Рассмотрим способ подключения программаторов к компьютеру. Наиболее распространенными способами подключения являются:

- подключение к принтерному порту (LPT);
- подключение к последовательному порту (COM);
- установка специальной платы в компьютер.

Каждый из этих способов имеет свои преимущества и недостатки. Использование специальных плат, устанавливаемых в компьютер, значительно упрощает схемотехнику программатора. В этом случае, как правило, удается отказаться от специального, довольно мощного источника питания, воспользовавшись источником питания компьютера, а также использовать центральный процессор компьютера в качестве управляющего процессора программатора. При способе подключения программатора к компьютеру посредством встраиваемых в компьютер плат удается достигнуть довольно значительных скоростей обмена между компьютером и программатором за счет непосредственного управления последним. Но такая реализация программатора имеет и существенные недостатки. Во-первых, значительно снижается мобильность программатора, т.е. возможность использования одного программатора на разных компьютерах, во-вторых, использование таких устройств с портативными компьютерами Notebook сопряжено с необходимостью использования специальных карт сопряжения.

Другой вариант сопряжения программатора с компьютером – последовательный канал компьютера. Это вполне допустимый вариант сопряже-

ния, допускающий работу программатора с компьютерами всех типов. К существенным недостаткам такого варианта сопряжения можно отнести невысокую пропускную способность канала. Максимальная скорость последовательного канала RS-232 ограничена значением 115 кБод, что существенно ограничивает обмен между компьютером и программатором, и, следовательно, снижает производительность последнего.

Подключение программатора к принтерному порту компьютера нам видится наиболее предпочтительным вариантом. Этот способ сочетает в себе достаточно высокую пропускную способность канала и не требует серьезных аппаратных затрат. При использовании этого способа удается воспользоваться центральным процессором компьютера в качестве управляющего процессора программатора.

1.2.2. Обновление версии работы программатора

Рассмотрим способ обновления версий программатора, способность программатора определять правильность установки микросхемы в колодке и проведение процедуры самотестирования при включении питания.

Способ обновления версии – это довольно существенный вопрос эксплуатации программаторов. Необходимость обновления версии может возникнуть по ряду причин, во-первых, при выявлении ошибки работы программатора, либо при расширении списка поддерживаемых программатором микросхем. Способ обновления версии программатора зависит от его аппаратного устройства. В одних изделиях алгоритмы программирования жестко «зашиты» в аппаратуру, в других – они являются загружаемыми. В первом случае для модификации версии требуется модификация самого устройства программатора (например, перепрограммирование ПЗУ самого программатора), а это сопряжено с рядом дополнительных неудобств. Другое дело, если обновление версии осуществляется только обновлением программного обеспечения программатора. Именно по такой схеме построены программаторы с загружаемыми алгоритмами программирования. В таких программаторах обновляется только программное обеспечение, и программатор работает уже с новой версией.

При кажущейся незначительности этой опции, мы начинаем понимать всю ее важность только после выхода из строя микросхемы при неверной установки ее в колодку. Именно для предотвращения таких ситуаций и служит эта опция. Здесь необходимо указать, что полноценная реализация такой возможности требует от разработчика больших усилий и, порой, изобретательности. Дело в том, что необходимо протестировать микросхему в колодке в самом щадящем для нее режиме, при этом ни в коем случае не допуская выхода микросхемы из строя.

И, в заключение обсуждения аппаратного устройства программаторов, необходимо упомянуть о настоятельной необходимости проведения программаторами процедуры самотестирования. Обычно эта процедура

проводится после инициализации аппаратуры программатора. Цель этой процедуры – встроенными средствами провести проверку работоспособности всего оборудования устройства и принять решение о возможности полноценной работы программатора. К сожалению, встроенными средствами не всегда удается однозначно убедиться в работоспособности всех узлов устройства, но, тем не менее, эта процедура обязательно должна проводится с целью минимизации вероятности эксплуатации неработоспособного оборудования.

1.3. Программное обеспечение программатора

Дадим общее представление о возможных способах реализации программного обеспечения (ПО) программатора. Первое, на что нужно обратить внимание, – это под управлением какой операционной системы работает программатор. Большинство программного обеспечения программаторов реализовано под управлением DOS и не предъявляет к компьютеру специфических требований. Обычно, это программы, разработанные довольно давно. В последние несколько лет стали появляться программаторы, работающие под управлением операционной системы WINDOWS. Это современные продукты, которые, как правило, выглядят гораздо элегантнее, имеют более дружественный интерфейс и обладают большим числом сервисных возможностей.

Рассмотрим архитектуру программного обеспечения программаторов. Наиболее распространенной является архитектура, в которой в качестве ядра программатора выступает промежуточный буфер данных. Все операции в программаторе выполняются с этим буфером. Для программирования микросхемы необходимо загрузить файл в буфер, запрограммировать данные из буфера в микросхему, сравнить содержимое микросхемы и буфера. При чтении данные из микросхемы записываются в буфер. Размер промежуточного буфера данных, обычно, коррелирован с размером текущего типа микросхемы. Многолетний опыт разработки и производства программаторов позволил выработать концепцию оригинальной многобуферной архитектуры ПО программаторов с неограниченным количеством буферов. Такая архитектура позволяет работать с неограниченным количеством независимых наборов данных, проводить их анализ и редактирование. Например, Вы можете воспользоваться двумя буферами для считывания в них двух разных микросхем, проведения анализа этих наборов данных и, на их базе, создания нового массива данных в третьем буфере для последующего программирования его в микросхемы памяти или сохранения на диске.

На рис. 6.1 показано взаимодействие ядра программы с модулями, подготовленными пользователем. Внутри ядра находятся основные интерфейсы, взаимодействующие с внешними (по отношению к нему) мо-

дулями и файлами данных, и другие неизменяемые части программы, обеспечивающие ее функционирование.

Модуль «Программирование» – собственно программа записи данных в микросхему, их чтения, сравнения и т.д. – реализует соответствующие временные диаграммы с учетом всевозможных параметров этих процессов.

Рис. 6.1. Взаимодействие ядра программы с модулями, подготовленными пользователем

Пользователь может разработать собственный модуль для нужной ему микросхемы, не вникая в конкретное устройство программатора и пользуясь только логическими понятиями шины данных, шины адреса, управляющих сигналов. Для этого в ядре программы имеется ряд стандартных функций, к которым можно обращаться из любого модуля.

Модуль «Редактор» служит для отображения на экране монитора содержимого буфера программирования с данными, предназначенными для занесения в ПЗУ или прочитанными из него. Чаще всего бывает достаточно поставляемым с программатором бинарного редактора для ПЗУ с линейной структурой и редактора ПЛМ для логических матриц. Но если требуется создать на экране образ ПЗУ в каком-либо необычном виде, придется написать собственный редактор. Задача эта сложная, но выполнимая. Доступен пользователю и модуль «Автоопределение» по многим причинам отделенный от модуля «Программирование». А в модуль «Подсказка» можно поместить справочные данные, относящиеся к модулям собственной разработки.

Информация, необходимая для связи всех модулей с ядром программы и относящаяся к конкретным типам программируемых микросхем, находится в конфигурационном файле, который пользователь может дополнять и редактировать. В дополнительном конфигурационном файле автоматически фиксируются данные о настройках программы, сделанных уже во время работы с ней.

Тип ПЗУ задается пользователем вручную или определяется с помощью модуля «Автоопределение». После этого программа выбирает модули «Редактор» и «Программирование», нужные для работы с ПЗУ этого типа, и передает им из конфигурационного файла необходимые параметры. «Редактор» через ядро программы выдает образ ПЗУ на экран монитора и позволяет редактировать его, пользуясь клавиатурой и «мышью». Модуль «программирование» через ядро управляет программатором, обеспечивая выполнение всех необходимых операций.

Стандартным набором функций программаторов обычно являются следующие функции: чтение, запись, сравнение, контроль на чистоту, стирание микросхем. Эти функции отображаются на приведенном рис. 6.2. Некоторые программаторы имеют функцию автоматического программирования. Эта функция позволяет осуществить часто используемую комбинацию действий для конкретного типа микросхемы. Обычно, такая комбинация состоит из такого набора: стереть микросхему, проконтролировать стертость, запрограммировать, сравнить запрограммированные данные с оригиналом, установить защиту. Удобство этой функции заключается в том, что весь набор активизируется одним нажатием.

Рис. 6.2. Структурная схема интерфейса программного обеспечения

Отдельно следует остановиться на редакторских функциях программаторов. Наиболее распространенным перечнем редакторских функций являются: редактирование данных в шестнадцатеричном формате, возможность заполнения буфера данных константой и подсчет контрольной суммы. Этого простого набора редакторских функций вполне достаточно для простых приложений. Для профессионального использования программаторов необходимы расширенные возможности редактирования. К ним можно отнести:

- возможность редактирования данных не только в шестнадцатеричном формате, но и в двоичном, восьмеричном и десятичном представлении;
- заполнение массива строкой данных;
- поиск и замена строки данных;
- инвертирование данных;
- копирование массива данных как внутри одного буфера, так и между разными буферами;
- подсчет контрольной суммы;
- конвертирование шин адреса и данных.

Еще одна особенность ПО программаторов, на которой стоит остановиться отдельно, – это пакетный режим работы. Очень в немногих программаторах такой режим реализован. А преимущество такого режима просто очевидно – это автоматизация работы. Использую пакетный режим работы, можно создавать сценарии работы с программатором, автоматизируя всю рутинную работу. Наиболее интересны устройства, где пакетный режим работы практически не имеет ограничений, в нем доступны все ресурсы программатора. В пакетном режиме можно загружать файлы, запускать программирование, манипулировать параметрами программирования, окнами на экране, выводить графические данные и т.д., и т.п. В качестве иллюстрации использования пакетного режима работы программатора можно привести задачу программирования партии микросхем, в каждой из которых должен быть запрограммирован серийный номер. На специальном языке создается сценарий работы программатора, который заключается в следующем: оператор указывает начальное значение серийного номера партии микросхем и запускает процедуру программирования, программатор программирует микросхему с текущим серийным номером и вычисляет серийный номер следующей микросхемы, помещая его в соответствующий раздел памяти, далее процедура циклически повторяется. В приведенном примере пакетный режим работы значительно облегчает работу оператора и исключает свойственные оператору ошибки.

Языки программирования

Языки программирования можно классифицировать на системные, высокого уровня и языки описания сценариев.

Системные языки появились в качестве альтернативы языкам ассемблера, позволяющим использовать в программе практически все особенности конкретной аппаратной подсистемы. Все они менее эффективно используют аппаратуру по сравнению с языками ассемблера, но позволяет быстрее создавать приложения. В результате им удалось практически полностью вытеснить языки ассемблера при создании крупных приложений.

Термин «высокоуровневый» означает следующее: многие детали обрабатываются автоматически, а программисту для создания своего приложения приходится писать меньшее количество строк. В частности:

Языки описания сценариев создавались для связывания готовых программ. Их применение подразумевает наличие достаточного ассортимента мощных компонентов, которые требуется только объединить друг с другом.

Типизация дает ряд преимуществ. Во-первых, крупные программы становятся более управляемыми. Четкость системы типов делает для программиста ясным, для чего предназначены те или иные данные; он легко может различать их между собой и соответственно использовать. Во-вторых, компиляторы используют информацию о типах для обнаружения некоторых видов ошибок, таких как попытка, использовать число с плавающей запятой в качестве указателя. В-третьих, типизация повышает производительность приложения, позволяя компиляторам генерировать более специализированный код.

В данной работе для запрограммирования микросхем на программаторе STERH была создана программа с помощью языка программирования СИ. Язык СИ – язык программирования системных и прикладных программ для профессиональных программистов.

Создание текста исходной программы для запрограммирования БИС, компиляция, редактирование, отладка и пуск программы выполнены в системе программирования СИ с использованием средств интегрированной среды и автономных компонентов операционной системы. Упрощенная схема взаимодействия компонентов приведена на рис. 6.3.

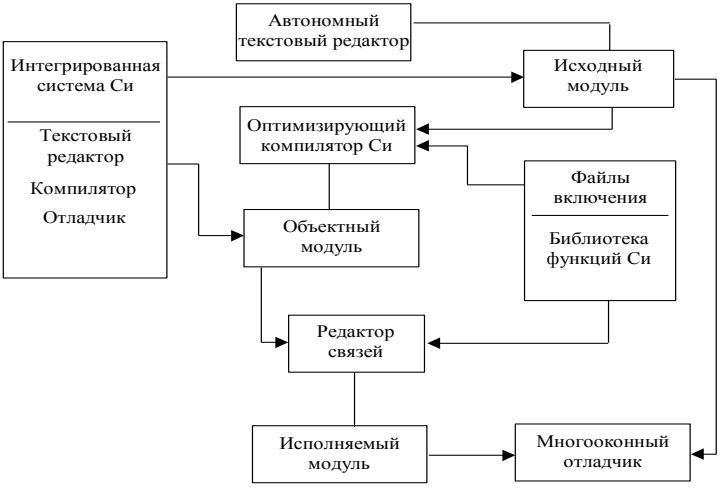


Рис. 6.3. Схема взаимодействия компонентов языка СИ

ЛАБОРАТОРНАЯ РАБОТА № 7
Методика программной и аппаратной эмуляции

1. Общетеоретические сведения

Программа для работы с программатором

Программа предназначена для работы с программатором STERH «ST-900» на компьютерах типа IBM PC XT и AT. Обмен данными между компьютером и программатором осуществляется через один из последовательных портов COM1, COM2 или параллельный порт LPT.

Создание текста исходной управляющей программы для запрограммирования БИС, компиляция, редактирование, отладка и пуск программы выполнены в системе программирования СИ и Borland C++ 5.02.

Обобщенная блок схема функционирования программы представлена на рисунке 7.1.

Работа программы на функциональном уровне:

1 – начало программы (инициализация);

2 – подключение библиотек, объявление типов, объявление констант;

3 – массивы, режимы работы программатора, указатель на файловый поток ввода/вывода;

4 – объявление вспомогательных переменных;

5 – вывод сообщения «Программатор ***»;

6 – проверка корректности параметров командной строки;

7 – параметры корректны;

8 – вызов ошибки Error message;

9 – установка режима работы программатора согласно параметрам командной строки. Инициализация файлового потока для работы со входными файлами;

10 – файловый поток проинициализирован корректно?;

11 – вызов ошибки Error message;

12 – определить адреса портов ввода/вывода;

13 – определить тип памяти (данные из команды);

14 – тип данных неизвестен?;

15 – вызов ошибки Error message;

16 – вывод «Базовый адрес LPT порта», адрес порта;

17 – программатор обнаружен?;

18 – очистить LPT порт;

19 – вывод «Программатор не обнаружен»;

20 – завершение программы;

21 – подключить программатор (задержка для надежности);

22 – разрешить программирование по трем адресам;

23 – ошибка программирования;

24 – отключить программатор, закрыть файловый поток ввода/вывода, вернуть операционной системе ошибку №0;

25- завершение программы;

26 – нажата любая клавиша;

27 – выбрать режим работы программатора.

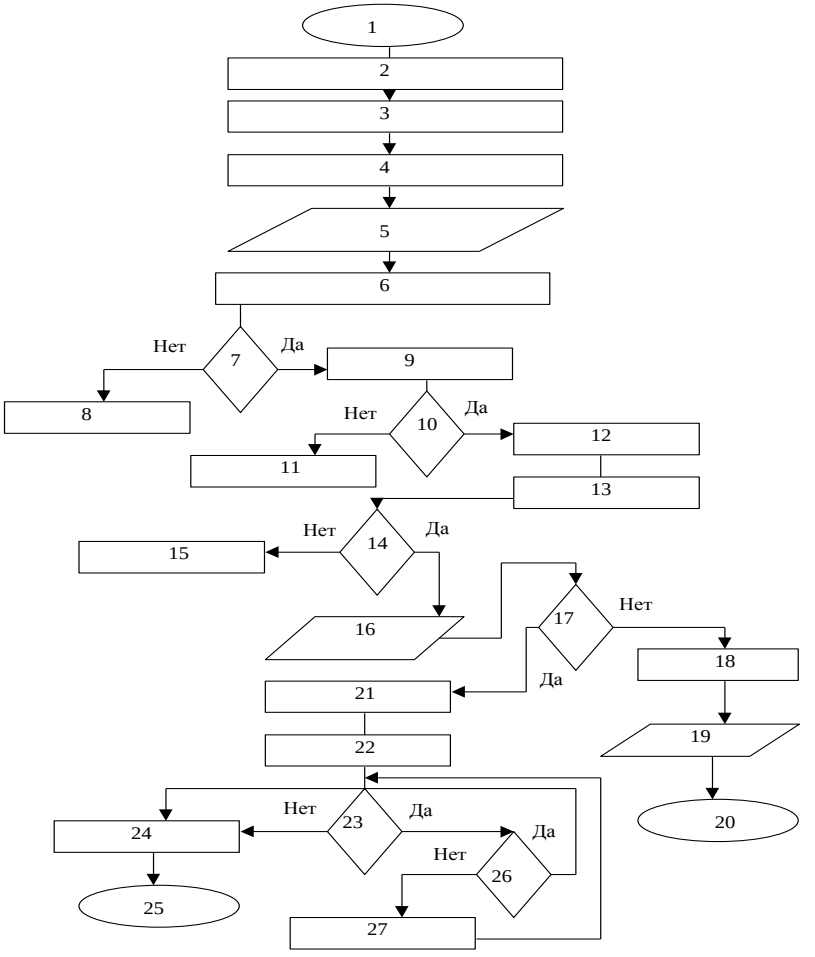


Рис. 7.1. Обобщенная блок-схема программы
для программирования БИС

Программа предоставляет пользователю следующие возможности:

– установить тип микросхемы;

– проверить микросхему на чистоту;
– проверить микросхему на возможность записи;
– прочитать микросхему в буфер;
– прочитать файл в буфер (BIN, HEX, JED);
– отредактировать данные (HEX, OCT, ASCII);
– запрограммировать содержимое буфера в микросхему;
– сохранить содержимое буфера в файле;
– сравнить содержимое микросхемы с буфером;
– прочитать идентификационную информацию из микросхемы;
– изменить алгоритмы и параметры программирования;
– сохранить параметры программирования в файле;
– восстановить параметры программирования из файла;
– вывести данные в символьном виде на принтер, консоль или в файл;
– выбрать и настроить последовательный порт для связи с программатором;
– инициировать начальные данные программы из файла ST900.CFG;
– сохранить конфигурацию;
– восстановить конфигурацию;
– открыть командный файл для пакетной работы программы;
– исполнить командный файл в пакетном режиме.

Для запуска программы с клавиатуры необходимо ввести «ST900e» («ST900r» для запуска русифицированной версии) и нажать на клавишу «ENTER». По старту программа ищет файл с именем st900.cfg в текущей директории, а если не находит, то и в директории откуда она была запущена.

Из этого файла берется информация об установленных пользователем типе микросхемы, номере порта и его скорости, и других параметрах. Если файл st900.cfg не найден, то производится установка порта COM1, скорости обмена 115200 бод, тип микросхемы 155РЕ3. Программатор настраивается на выбранную в программе скорость автоматически.

После нормального запуска программы происходит запрос версии у программатора и при правильном его подключении в правом верхнем углу экрана появляется информация.

Одновременно в центре экрана появляется информация о начальных установках программы.

Управление всеми режимами осуществляется из командной строки, которая появляется в верхней строке экрана, либо с помощью «горячих» клавиш, отображаемых в нижней строке экрана. Горячие клавиши работают из любого подменю за исключением операций, при которых происходит обмен данными с программатором. Основное поле экрана ис-

пользуется для редактирования данных и организации диалога с пользователем.

Слова в командной строке представляют собой меню, которые выпадают при выборе этих слов клавишами ALT+ первая буква команды. Попасть в командную строку также можно, нажав клавишу F10.

Перемещение по меню программы осуществляется при помощи стрелок или нажатием выделенных цветом букв. Выбор в меню производится нажатием клавиши «Enter», отказ от выбора и выход в предыдущее состояние нажатием клавиши «Esc». Выбрав определенную команду, Вы попадаете в меню, состоящее из списка элементов. Выбор элемента меню приводит к выполнению определенных действий либо к появлению нового меню. Рассмотрим все возможные меню и связанные с выбором их элементов действия.

Работа программы (редактирование, обмен с программатором и файлами) зависит от типа микросхем, поэтому в начале работы с программой необходимо выбрать нужный тип («Микросхема»).

«Файл» (File)

В этом меню возможны следующие операции:

– Сохранить файл из буфера (F2 – горячая клавиша)
– Загрузить файл в буфер (F3 – горячая клавиша)
– Загрузить ENCRYPT ARRAY из файла
– Сохранить ENCRYPT ARRAY в файл
– Загрузить SECURITY KEYS
– Сохранить SECURITY KEYS
– Открыть командный файл
– Информация
– Вызов оболочки DOS (DosShell)
– Выход из программы (AltX – горячая клавиша)

При работе с файлами команд «Чтение файла» и «Запись файла» по умолчанию используют расширения имен файлов, соответствующие выбранному типу микросхем. При чтении файла допустимо использование wildcards в имени файла, что позволяет произвести выбор файла в режиме меню. Команды для работы с ENCRYPT ARRAY (секретный массив) являются допустимыми только для микроконтроллеров серии 8Xx51. Команды для работы с SECURITY KEYS (секретные ключи) являются допустимыми только для микроконтроллеров серии 8Xx196. Для остальных типов микросхем эти строки выделены другим цветом и доступ к ним невозможен.

«Редактор» (Edit)

По этой команде происходит переход в режим редактирования данных.

Представление и редактирование данных возможно в HEX, OCT и ASCII виде.

Переход в ASCII моду и обратно происходит только из HEX режима по нажатию на клавишу «TAB». Переход HEX-OCT осуществляется по нажатию на Alt+M. Основные команды редактора можно посмотреть, нажав клавишу F1 (HELP). Для микросхем типов PLM, PAL, EPLD предусмотрены индивидуальные редакторы, учитывающие архитектуру микросхем.

«Микросхема» (Device)

В этом меню происходит выбор типа микросхемы, с которой Вы собираетесь работать. Все микросхемы разбиты на 9 подгрупп.

1. PROM – 155ре3, 556(рт4-рт7,рт7а), 556(рт11-рт20), 541рт1, 541рт2, 1608рт1, 1623рт1, 1623рт2;

   EPROM – 2716-27256, 27c64-27c080, 27c100, 573РФ2, 573РФ3, 573РФ4-РФ8, 27c1024, 27c2048, 27c4096;

   EEPROM – 2864A(1609рр2), 28c64(573рг3), 28c256, 28c010-28c040at, 28c1024, 558(рр2-рр4), 573рр2;

   SEEPROM – 24lc01-24lc65, 93c06-93c66, 85c72-85c92.

2. FLASH – 28f256-28f020, 28f001bx/t, 28f001bx/b, AM28f1024, AM28f256-AM28f020, AM29f010, AM29f040.

3. MICRO 8741(A), 8748(H), 8749H, 8048H-8050H, 1816BE48, 8755, 573РФ10;

   MCS51 – 8744H, 8751H, 8751BH, 8752BH, 87c51, 87c51(FA-FC), 8752-8758, 89c51, 1816BE51.

4. PIC16 – PIC16c54, PIC16c55, PIC16c56, PIC16c57, PIC16c58a, PIC16c61, PIC16c62, PIC16c620, PIC16c621, PIC16c622, PIC16c64, PIC16c65, PIC16c71, PIC16c73, PIC16c74, PIC16c84, PIC17c42, PIC17c43, PIC16c44.

5. PLM – 556рт1, 556рт2.

6. PAL – pal16l8, pal16r4, pal16r6, pal16r8, 1556хл8, 1556хп4, 1556хп6, 1556хп8.

7. EPLD – 85c220, 85c224, 85c060, 85c090, 85c22v10, 5ac312, 5ac324, 5ac180, 85c508, 85c960.

8. RAM – 537ру10, 537ру17.

9. Zilog – Z86e02, Z86e03, Z86e04, Z86e06, Z86e07, Z86e08, Z86e21, Z86e22, Z86e23, Z86e30, Z86e31, Z86e40.

Выбирая необходимую группу, Вы попадаете ещё в одно меню, в котором выбирается непосредственно тип микросхемы.

«Команда» (Command)

Из этого меню производится все управление блоком программатора:

– Чтение микросхемы в буфер (F5 – горячая клавиша)

– Запись микросхемы из буфера (F6 – горячая клавиша)

– Сравнение содержимого микросхемы с буфером (F7 – горячая клавиша)

– Проверка микросхемы на чистоту (F8 – горячая клавиша)

– Проверка микросхемы на возможность записи (F9 – горячая клавиша)

– Записать секретные биты

– Записать ENCRYPT ARRAY

– Стереть микросхему

– Защита записи

– Конфигурация

– Выключить питание для RAM

– Проверить тестовые вектора

– Выполнить

– Прочитать идентификатор микросхемы

«Опции» (Option)

– Выбор порта COM1, COM2, COM3, COM4 – работа с портами напрямую (макс.скорость 57600 бот)

– COM1 INT14, COM2 INT14 – работа с портами через BIOS INT14 (макс.скорость 9600 бот)

– Выбор скорости обмена 9600, 19200, 57600, 115200 бот

– Звук (вкл/откл)

– Параметры программирования

– Сохранить параметры (в файле *.par)

– Восстановить параметры (из файла *.par)

– Сохранить конфигурацию (в файле *.cfg)

– Восстановить конфигурацию (из файла *.cfg)

Программирование скорости обмена для значения выше 9600 производится напрямую для микросхемы 8250 (либо ее аналога). Поэтому для машин имеющих другие микросхемы в последовательных портах (например ЕС-1840, ROBOTRON-1910) работа со скоростью выше 9600 невозможна.

При выборе порта COMxINT14 работа будет осуществляться строго через INT 14H (BAUD <=9600).

Если при работе возникают сообщения «Ошибка связи», то возможной причиной является недостаточная скорость компьютера плюс накладные расходы BIOSa при работе через INT 14H, либо наличие резидентных программ (TSR), забирающих процессор на слишком длительное время.

Рекомендуется:

а) убрать загрузку резидентных программ при запуске компьютера;

б) понизить скорость обмена.

Наиболее важный пункт в этом меню – «Параметры программирования».

Как показывает опыт работы с ПЗУ, довольно часто бывают ситуации, когда микросхемы не программируются при напряжениях, которые

указаны в ТУ на данный тип. Поэтому в данной версии программатора практически все параметры программирования можно изменять в довольно широких пределах, что позволяет получить более высокий коэффициент выхода годных микросхем.

Существует несколько параметров, которые присутствуют у любой группы микросхем:

– Уровень защиты по напряжению питания (85,300,600,900 ма)

– Уровень защиты по напряжениям программирования (0-300ма шаг 25ма)

– Тип проверки при программировании

Все биты:

– Контроль каждого байта при записи. После записи 256 байт контроль всего блока с выключенным Vpp при напряжении питания равном VCCv1

Только программируемые биты:

– Контроль только программируемых битов без контроля блока с выключенным VPP

Без контроля:

– Запись без контроля. Подается только один импульс программирования, после чего без контроля происходит переход к следующему биту либо адресу

С контролем при двух напряжениях питания:

– Контроль каждого байта при записи. После записи 256 байт контроль всего блока с выключенным Vpp при Vcc=Vccv1 и Vcc=Vccv2

– Максимальное количество попыток программирования Npw (1-255). Для PROM и PLM микросхем реальное количество равняется Npw*256 (256-65535)

– Количество импульсов допрограммирования Ndw (1-255)

– Длительность импульса программирования Twr

Для разных групп микросхем и алгоритмов Twr меняется в разных пределах:

PROM, PAL, PLM – 4-80 мкс (шаг 4мкс);

EPROM, PLD, MICRO – 25 мкс–80мс (шаг 25мкс для QUICK алгоритма; шаг 1мс для INTEL и NORMAL алгоритма).

Программирующие напряжения зависят от типа микросхемы и изменяются индивидуально.

При старте программы значения всех параметров берётся из самой задачи ST900e.exe (ST900r.exe). Если возникла необходимость поменять какие-то параметры, и потом Вы планируете работать именно с такими значениями, то Вы можете сохранить параметры в файле «Сохранить параметры».

При очередном запуске задачи Вы выполняете операцию «Восстановить параметры», после чего Вы получите Ваши изменённые параметры.

Количество файлов с разными параметрами может быть любым, а Вы по мере необходимости работаете с любым из них.

Для работы с микросхемами К537РУ10(17) в качестве эмуляторов ПЗУ, необходимо на 12(14) и 24(28) выводы микросхем подпаять конденсатор на 10-33 мкФ., для сохранения информации при переносе микросхемы из программатора в отлаживаемую схему. Одновременно рекомендуется припаять резисторы на 4.7-10 кОм с выводов питания 24(28) на выводы CS 18(20) и WR 21(27). После завершения работы с этими типами микросхем необходимо исполнить операцию «Power OFF» для перевода программатора в исходное состояние.

«Командные файлы в программе ST900»

Каждая команда должна занимать отдельную строку.

Команда – это собственно команда и ее параметры, разделенные пробелами или табуляторами. Команды и параметры можно писать большими и маленькими буквами.

Все команды интерпретируются исходя из текущего контекста программы.

Изменения, внесенные командами установки параметров, остаются и после окончания работы командного файла.

Текст выполняемого командного файлы отображается в специальном окне по мере выполнения.

Ошибка во время исполнения любой команды останавливает выполнение командного файла.

На данный момент возможны следующие команды:

Check – проверить микросхему на чистоту

Compare – сравнить микросхему с содержимым буфера addr, size

Erase – стереть микросхему

Ident – прочитать идентификационную информацию из микросхемы

Load имя.файла – загрузка файла в буфер программы: файл загружается в буфер, начиная с адреса указанного командой «Set addr xxxx» и размером, указанным «Set size xxxx», если задан HEX формат, то читаются только данные, попавшие в окно от addr – addr+size

Load_Encr имя.файла – загружает бинарный файл в Encrypt-буфер, команда используется для микропроцессоров с Encrypt Array

Load_Key имя.файла – загружает бинарный файл в Key Буфер, используется для работы с микропроцессорами 196 серии

Protect – установить защиту

Read – прочитать микросхему, начиная с адреса addr и размером size в буфер

Save имя файла – записать буфер в файл: данные из буфера, начиная с адреса, установленного «Set addr xxxx» и размером «Set size xxx», записываются в файл

Set parameters

Parameters:

Type – установить тип микросхемы

Addr – установить стартовый адрес для операций с микросхемой. Устанавливается на начало при команде Set Type xxx

Size – установить размер в байтах для операций с микросхемой

Ask – спрашивать подтверждение для операций, связанных с программированием микросхемы

NoAsk – запретить подтверждение на операции программирования

Unprotect – снять защиту с микросхемы

Wcheck – проверить микросхему с addr размером size на возможность записи

Write [parameter [xxx]] – запрограммировать микросхему, начиная с addr размером size

Parameters:

Encrypt – запрограммировать Encrypt Array

Lock [lock_bit_number] – запрограммировать LOCK биты.

2. Программирование микросхемы

Для того, чтобы корректно запрограммировать БИС, нужно определить устройство, в котором она будет исполнять конкретную функцию, назначение микросхемы для заданного устройства, и после этого, опираясь на заданное разрабатываемое устройство, писать программу прошивки интегральной схемы.

Для написания программы прошивки микросхемы была использована интегрированная система разработки MPLAB.

2.1. MPLAB интегрированная система разработки

MPLAB – это интегрированная среда разработки (IDE) для семейства микроконтроллеров PICmicro фирмы Microchip Technology Incorporated. MPLAB позволяет писать, отлаживать и оптимизировать программы для разработок. MPLAB включает текстовый редактор, симулятор (виртуальный отладчик), менеджер проектов и поддерживает эмуляторы (внутрисхемные отладчики) MPLAB-ICE и PICMASTER, программаторы PICSTART Plus и PRO MATE II и другие средства и инструменты разработок фирмы Microchip и других фирм.

Инструментальные средства MPLAB, организованные как ниспадающие меню и определяемые быстрые клавиши, позволяют:

– ассемблировать, компилировать исходный текст;

– отлаживать логику работы, наблюдая с помощью симулятора или, в реальном времени, с эмулятором MPLAB-ICE;

– просматривать переменные в окнах просмотра;

– программировать кристаллы с помощью программаторов PICSTART Plus или PRO MATE II и многое другое.

На рис. 7.2. представлена схема общего проекта и структурные уровни иерархии в интегрированной среде MPLAB.

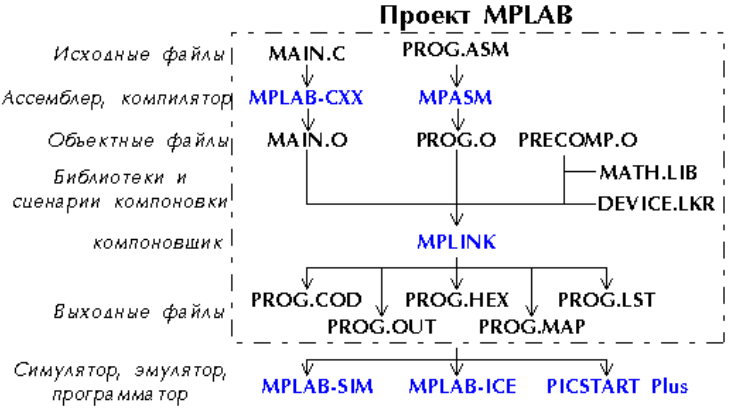


Рис. 7.2. Схема полного проекта в MPLAB

MPLAB работает под Microsoft Windows 3.1x, Windows 95, 98, NT, 2000 (начиная с версии 5.00.00). Правда, не все дополнительное оборудование, такое как внутрисхемные эмуляторы и программаторы, будет функционировать под всеми операционными системами.

2.2. MPLAB – интегрированная среда разработки (IDE)

MPLAB представляет собой законченную среду разработки, интегрируя несколько инструментальных средств:

MPLAB – менеджер проектов (project Manager). Используется для создания проектов и работы со связанными файлами.

MPLAB – редактор (Editor). Используется для создания и редактирования текстовых файлов, таких как исходные и файлы сценариев компоновки.

MPLAB-SIM – симулятор (Simulator). Позволяет моделировать выполнение команд и входные/выходные сигналы микроконтроллеров.

MPLAB-ICE – внутрисхемный эмулятор (Emulator). Позволяет, используя дополнительные аппаратные средства и компьютер, заменять

микроконтроллер в разрабатываемом устройстве в реальном времени. MPLAB-ICE – самый новый эмулятор от Microchip.

MPASM – универсальный ассемблер.

MPLINK – компоновщик (linker). Создает законченное приложение, связывая перемещаемые модули MPASM, MPLAB-C17 и MPLAB-C18.

MPLIB – управляет пользовательскими библиотеками.

MPLAB-CXX – компилятор С. Позволяет включать в проект исходные тексты, написанные на языке высокого уровня С и на ассемблере.

PRO MATE II и PICSTART Plus – программаторы.

PICMASTER и PICMASTER-CE – внутрисхемные эмуляторы.

2.3. PIC контроллеры серии 16C84

PIC16C84 относится к семейству КМОП микроконтроллеров. Отличается тем, что имеет внутреннее 1Kx14 бит EEPROM для программ, 8-битовые данные и 64 байт EEPROM памяти данных. При этом отличаются низкой стоимостью и высокой производительностью. Пользователи, которые знакомы с семейством PIC16C5X могут посмотреть подробный список отличий нового от производимых ранее контроллеров. Все команды состоят из одного слова (14 бит шириной) и исполняются за один цикл (400 нс при 10 МГц), кроме команд перехода, которые выполняются за два цикла (800 нс). PIC16C84 имеет прерывание, срабатывающее от четырех источников, и восьмиуровневый аппаратный стек.

Периферия включает в себя 8-битный таймер/счетчик с 8-битным программируемым предварительным делителем (фактически 16-битный таймер) и 13 линий двунаправленного ввода/вывода. Высокая нагрузочная способность (25 мА макс. – втекающий ток, 20 мА макс. – вытекающий ток) линий ввода/вывода упрощают внешние драйверы и, тем самым, уменьшается общая стоимость системы. Разработки на базе контроллеров PIC16C84 поддерживается ассемблером, программным симулятором, внутрисхемным эмулятором (только фирмы Microchip) и программатором.

Серия PIC16C84 подходит для широкого спектра приложений от схем высокоскоростного управления автомобильными и электрическими двигателями до экономичных удаленных приемопередатчиков, показывающих приборов и связных процессоров. Наличие ПЗУ позволяет подстраивать параметры в прикладных программах (коды передатчика, скорости двигателя, частоты приемника и т.д.).

Малые размеры корпусов, как для обычного, так и для поверхностного монтажа, делает эту серию микроконтроллеров пригодной для портативных приложений. Низкая цена, экономичность, быстродействие, простота использования и гибкость ввода/вывода делает PIC16C84 привлекательным даже в тех областях, где ранее не применялись мик-

роконтроллеры. Например, таймеры, замена жесткой логики в больших системах, сопроцессоры.

Следует добавить, что встроенный автомат программирования EEPROM кристалла PIC16C84 позволяет легко подстраивать программу и данные под конкретные требования даже после завершения ассемблирования и тестирования.

Эта возможность может быть использована как для тиражирования, так и для занесения калибровочных данных уже после окончательного тестирования.

Обзор характеристик:

– только 35 простых команд;

– все команды выполняются за один цикл (400ns), кроме команд перехода (2 цикла);

– рабочая частота 0 Гц... 10 МГц (min 400 нс цикл команды);

– 14-битовые команды;

– 8-битовые данные;

– 1024 x 14 электрически перепрограммируемой программной памяти на кристалле (EEPROM);

– 36 x 8 регистров общего использования;

– 15 специальных аппаратных регистров SFR;

– 64 x 8 электрически перепрограммируемой EEPROM памяти для данных;

– восьмиуровневый аппаратный стек;

– прямая, косвенная и относительная адресация данных и команд;

– четыре источника прерывания;

– внешний вход INT;

– переполнение таймера RTCC;

– прерывание при изменении сигналов на линиях порта B;

– по завершению записи данных в память EEPROM.

Периферия и Ввод/Вывод:

– 13 линий ввода-вывода с индивидуальной настройкой;

– втекающий/вытекающий ток для управления светодиодами;

– макс. втекающий ток – 25 мА

– макс. вытекающий ток – 20 мА

– 8-битный таймер/счетчик RTCC с 8-битным программируемым предварительным делителем;

– автоматический сброс при включении;

– таймер включения при сбросе;

– таймер запуска генератора;

– Watchdog таймер WDT с собственным встроенным генератором, обеспечивающим повышенную надежность;

– EEPROM бит секретности для защиты кода;

– экономичный режим SLEEP;

– выбираемые пользователем биты для установки режима возбуждения встроенного генератора;

– генератор: RC;

– обычный кварцевый резонатор: XT

– высокочастотный кварцевый резонатор: HS

– экономичный низкочастотный кристалл: LP

– встроенное устройство программирования EEPROM памяти программ и данных; используются только две ножки.

КМОП технология:

– экономичная высокоскоростная КМОП EPROM технология;

– статический принцип в архитектуре;

– широкий диапазон напряжений питания и температур;

– коммерческий: 2.0... 6.0 В, 0...+70С;

– промышленный: 2.0... 6.0 В, -40...+70С;

– автомобильный: 2.0... 6.0 В, -40...+125С;

– низкое потребление;

– 3 мА типично для 5В, 4МГц;

– 50 мкА типично для 2В, 32КГц;

– 26 мкА типично для SLEEP режима при 2В.

ЛАБОРАТОРНАЯ РАБОТА № 8
Обеспечение достоверности записи информации с помощью тестирования

1. Отладка микроконтроллеров с помощью эмулятора ПЗУ

Трудоемкость разработки и отладки рабочей программы электронного устройства, содержащего микропроцессор, нередко определяет стоимость его разработки в целом. В микроконтроллерах (МК), интегрировавших в себе память и некоторые периферийные устройства, это проявляется особенно сильно. Одно из средств, значительно облегчающих отладку, – эмулятор ПЗУ. Возможности отладки не ограничиваются изложенными приемами.

Сложность и трудоемкость процесса отладки программного обеспечения МК определяется следующими факторами:

– сильной взаимосвязью программной и аппаратной частей системы;

– отсутствием непосредственного доступа к внутренним ресурсам и контрольным точкам МК;

– многоразрядным характером сигналов, сложно распределенных во времени;

– непериодичностью или очень низкой частотой повторения сигналов в системе;

– большим разнообразием внешних устройств и протоколов обмена информацией с ними.

Традиционная контрольно-измерительная аппаратура (например, осциллограф) может лишь в ограниченной степени использоваться для отладки МК.

Простейший (и, одновременно, самый неэффективный) способ отладки – «метод проб и ошибок»: загрузка программы в репрограммируемое постоянное запоминающее устройство (РПЗУ), попытка ее выполнения, обнаружение и исправление ошибок в программе и аппаратуре, стирание РПЗУ, новая загрузка программы и т.д. Процессы стирания и записи данных в микросхему РПЗУ занимают много времени, а после определенного числа циклов перепрограммирования она вообще выходит из строя. Многократные установки и извлечения микросхемы снижают надежность электрических контактов в розетке РПЗУ. Возможность получения отладочной информации о системе практически отсутствует.

В настоящее время МК чаще всего отлаживают кросс-средствами на базе персонального компьютера. Это позволяет в минимальной степени отвлекать ресурсы МК. Отлаживаемое устройство, как показано на рис. 8.1, соединяют с компьютером через некоторое инструментальное средство, например, эмулятор ПЗУ. Такой комплекс позволяет загружать и редактировать программу, вводить в нее тестовые модули, получать определенную информацию о системе и многое другое.

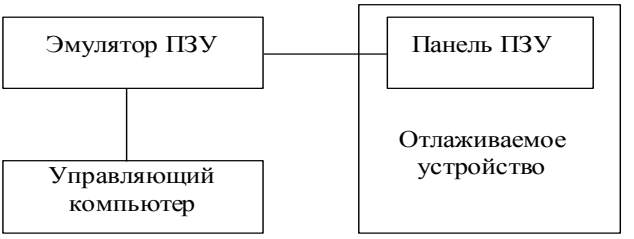


Рис. 8.1.Общая структурная схема отладки МК при помощи эмулятора

Хотя эмуляторы ПЗУ не единственное и не самое мощное отладочное средство, они по-прежнему популярны. Их «долголетие» объясняется независимостью от типа МК (необходима только возможность работы с внешней памятью программ), работой в реальном масштабе времени, невысокой ценой и доступностью широкому кругу разработчиков аппаратуры и радиолюбителей. Технологию отладки программы с помощью эмулятора ПЗУ рассмотрим на примере МК семейства PIC16C84. Приступая к отладке, необходимо проверить работоспособность аппаратной части системы и самого МК. Для этого можно воспользоваться тестами «свободного счета», которые заключаются в переборе всех возможных кодовых комбинаций на линиях портов МК. В процессе тестирования линии портов будут устанавливаться в состояние вывода, поэтому, прежде всего, убедимся по схеме отлаживаемого устройства, что они не нагружены на выходы других элементов. Временно отсоединим такие нагрузки, если они есть.

Загружаем эмулятор ПЗУ кодами команды «нет операции» (NOP) в пределах всего адресного пространства программ МК. Для PIC это код ООН. Выполняя такую «программу», МК последовательно перебирает все адреса программной памяти. Проконтролируем осциллографом сигналы ALE, PME и портов PO, P2. Осциллограммы на линиях портов должны соответствовать временным диаграммам двоичного счетчика с учетом мультиплексирования на PO младшего байта адреса и данных.

Далее проверяем порты P1 и P3, загрузив в эмулятор ПЗУ тестовую программу, приведенную в табл. 8.1. Она выводит на линии указанных портов последовательность кодов от ООН до OFFH, моделируя восьмиразрядный двоичный счетчик. Если осциллограммы соответствуют требуемым, восстановите отключенные от портов нагрузки и переходим к отладке рабочей программы МК. Помните, что вы отлаживаете одновременно программную и аппаратную части системы, и не забывайте периодически контролировать осциллографом сигналы в контрольных точках. Несоответствие вида сигналов вашим представлениям о них – повод для серьезных размышлений и дополнительных проверок.

Таблица 8.1

Тестовая программа проверки портов P1 и P3

	MOV	P1, #0;	Обнуляем P1 и P3.
	MOV	P3. #0;	
MARK:	INC	P1;	Увеличиваем на единицу
	INC	P3;	P1 и P3.
	SJMP	MARK;	Бесконечный цикл.

Для облегчения разработки и отладки следует придерживаться модульного принципа программирования, т. е. разделять программу МК на части по функциональному признаку. Это позволит легко перемещать отдельные модули, а при необходимости и применять их в других проектах. Модуль, в который передается управление после включения питания или сброса МК, называют главным или основным. Передавать управление на точку входа в программу следует командой перехода, а не вызовом подпрограммы, чтобы не засорять стек адресом возврата.

Во избежание непредсказуемых результатов работы программы необходимо присвоить начальное значение каждой из переменных до ее первого использования. В некоторых случаях может потребоваться инициализация периферийного оборудования. Блок инициализации помещают в начало основного программного модуля.

Примерный вид основного программного модуля в начале отладки приведен в табл. 8.2. Он содержит только точку входа в программу и обработчик выхода из нее. Хотя в микроконтроллерных системах такой выход используют крайне редко, его необходимо предусмотреть для корректного завершения аварийных ситуаций.

Таблица 8.2

Программный модуль вначале отладки

START:	…	…;	Точка входа в программу.
	…	…;	Команды основного
	…	…;	Программного модуля.
END:	…	…;	Команды обработчика
	…	…;	Выхода из программы.
	LJMP	END;	Бесконечный цикл.

В рассматриваемом примере после завершения работы программа «зацикливается». Из этого состояния ее выведет только перезапуск системы сигналом аппаратного сброса.

В ходе отладки к основному модулю по мере готовности добавляют другие программные модули. Порядок их подключения и отладки играет важную роль. Начинать следует с драйверов устройств вывода информации (дисплея, цифроаналогового преобразователя и т.п.), поместив их вызовы в основной модуль. Затем отлаживают драйверы остальных периферийных устройств, подпрограммы обработки данных и лишь после этого проверяют совместную работу всех программных модулей. При наличии буквенно-цифрового дисплея его драйвер отлаживают в первую очередь и используют в дальнейшем для вывода отладочной информации, например, содержимого внутренней памяти данных МК. Если используется такой эмулятор ПЗУ, что находящуюся в нем информацию способны записывать и считывать как управляющий компьютер, так и отлаживаемое устройство, МК может поместить отладочные данные в свободную область памяти эмулятора ПЗУ, а управляющий компьютер считает их и выведет на свой дисплей.

Например, чтобы вывести содержимое внутреннего ОЗУ данных МК, соедините его выход сигнала записи во внешнюю память данных (WR) со входом сигнала записи эмулятора ПЗУ и используйте подпрограмму, приведенную в табл. 8.3. Предполагается, что объем памяти программ отлаживаемого устройства не превышает 32 Кбайт, поэтому отладочная информация, помещаемая в память эмулятора, начинается с адреса 8000H. Содержимое регистров RO и R1 пересыпается отдельно, так как в дальнейшем они используются в подпрограмме для организации цикла. После вывода отладочной информации программу МК приостанавливают, считывают управляющим компьютером содержимое ячеек памяти эмулятора ПЗУ 8000H-807FH, выводят его на дисплей и анализируют. Аналогичным образом может быть выведено содержимое всех программно доступных регистров МК.

Таблица 8.3

Вывод содержимого внутреннего ОЗУ данных МК и запись во внешнюю память данных (WR) со входом сигнала записи эмулятора ПЗУ

SAVEDATA:	ANL	PSW, #0E7H;	Выбираем регистровый банк 0.
	MOV	DPTR, #8000H;	Заносим начальный адрес внешней памяти данных в DPTR
	MOV	A, R0;	Пересылаем содержимое R0 в эмулятор
	MOVX	@DPTR, A;	

	INC	DPRT;	Увеличиваем адрес внешней памяти данных
	MOV MOVX	A, R1; @DPTR, A;	Пересылаем содержимое R1 в эмулятор
	MOV	R1, #7EH;	R1 – счетчик цикла
	MOV	R0, #1H;	R0 – адрес внутреннего ОЗУ данных
LOOP:	INC	R0;	Увеличиваем адрес внутреннего ОЗУ данных
	INC	DPTR;	Увеличиваем адрес внешней памяти данных
	MOV MOVX	A, @R0; @DPTR, A;	Пересылаем содержимое внутреннего ОЗУ данных в эмулятор
	DJNZ	R1, LOOP;	Проверяем завершение цикла

2. Отладка драйверов

Приступая к отладке драйвера периферийного устройства, временно отключают от него вырабатываемые МК управляющие сигналы, чтобы избежать возможного выхода устройства из строя из-за ошибок в программе. Если процесс носит однократный характер, «зацикливаем» его и запрограммируем при необходимости сигнал синхронизации осциллографа. Отлаживаем драйвер, контролируя осциллографом формируемые МК сигналы. Убедившись в соответствии временных диаграмм управляющих сигналов требуемым, подключаем периферийное устройство и продолжаем отладку драйвера на реальной аппаратуре. В заключение удаляем из программного модуля отладочные элементы и проверяем его работу в окончательном виде.

Использование общих ресурсов МК разными модулями довольно часто приводит к тому, что отлаженная программа перестает работать при добавлении еще одной подпрограммы. Поэтому после отладки очередного модуля убедитесь, что все ранее отлаженные драйверы и подпрограммы продолжают работать правильно. Если в программе используются прерывания, не запрещайте их без крайней необходимости. Отлаженный модуль не следует удалять из программы, даже если он в данный момент не нужен.

При «зависании» МК пригодится следующий метод локализации ошибки: введите в программу контрольные точки, выводящие на дисплей последовательно возрастающие числа. После «зависания» на дисплее будет отображено число, соответствующее последней успешно пройденной контрольной точке. Если несколько таких точек попали в бесконечный цикл, числа на дисплее будут быстро сменяться. Для того, чтобы определить,

какие именно точки попали в цикл, придется искусственно замедлить смену чисел, задав при выводе каждого из них программную задержку, например, в виде холостого цикла. Если в отлаживаемой системе нет встроенного дисплея, информацию можно вывести на дисплей управляющего компьютера через свободную область памяти эмулятора ПЗУ.

Отладив все драйверы устройств, приступают к отладке прочих подпрограмм. Если какая-либо из них реализует сложный алгоритм обработки или преобразования данных, вывод на дисплей одного или нескольких промежуточных значений переменных зачастую не дает достаточной для анализа ошибок информации. Преодолеть трудности и здесь поможет запись отладочной информации необходимого объема в свободную область памяти эмулятора ПЗУ.

Добившись нормальной работы всех программных модулей, можно отлаживать их совместно. Возникающие при этом трудности делятся на две группы, К первой относятся проблемы совместного использования общих ресурсов МК: арифметико-логического устройства, памяти данных, портов ввода-вывода. Вторая связана с работой микроконтроллерных устройств в реальном масштабе времени.

Системы реального времени обычно являются многопоточными. Несколько программных задач (потоков) выполняются параллельно, взаимодействуя друг с другом и используя общие ресурсы. Но в каждый момент МК в силу своей структуры решает только одну из них, поочередно переключаясь на другие с учетом приоритета. Конфликты между задачами возникают как из-за недостатка ресурсов, так и из-за дефицита времени на обработку данных. Поэтому обращайте особое внимание на ресурсы МК, используемые в программных модулях, чаще контролируйте изменение их состояния путем вывода отладочной информации. Старайтесь уменьшить число глобальных переменных, по возможности заменяя их локальными. Следите за состоянием стека. Оценивайте время выполнения критичных участков программы, проверяйте систему при различных значениях входных сигналов, возможных на практике.

С помощью эмулятора ПЗУ можно отладить и программу, предназначенную для работы во внутренней памяти МК, если временно разместить ее во внешней памяти программ. Так как при этом порты P0 и P2 окажутся занятыми обслуживанием внешней памяти, то, если оставшихся свободными линий ввода-вывода МК недостаточно, P0 и P2 заменяют портами ввода-вывода, адресуемыми как ячейки внешней памяти данных. Их подключают по стандартным схемам, применяя для увеличения числа линий вывода триггерные регистры, а для увеличения числа линий ввода – элементы с тремя состояниями выходов.

Отладив систему с помощью эмулятора, удалите из нее отладочные элементы, запишите программу в ПЗУ (или во внутреннюю память программ МК) и проверьте работу устройства в окончательном виде.

3. Программное тестирование записи

Проверку на достоверность записи информации в микросхему возможно обеспечить не только аппаратными средствами как эмуляторы, но и встроенными программами тестерами, которые вшиты в основной программе для программирования микросхем программатором.

С помощью данных тестов мы можем определить насколько правильно и корректно запрограммировался контроллер на уровне программного исполнения.

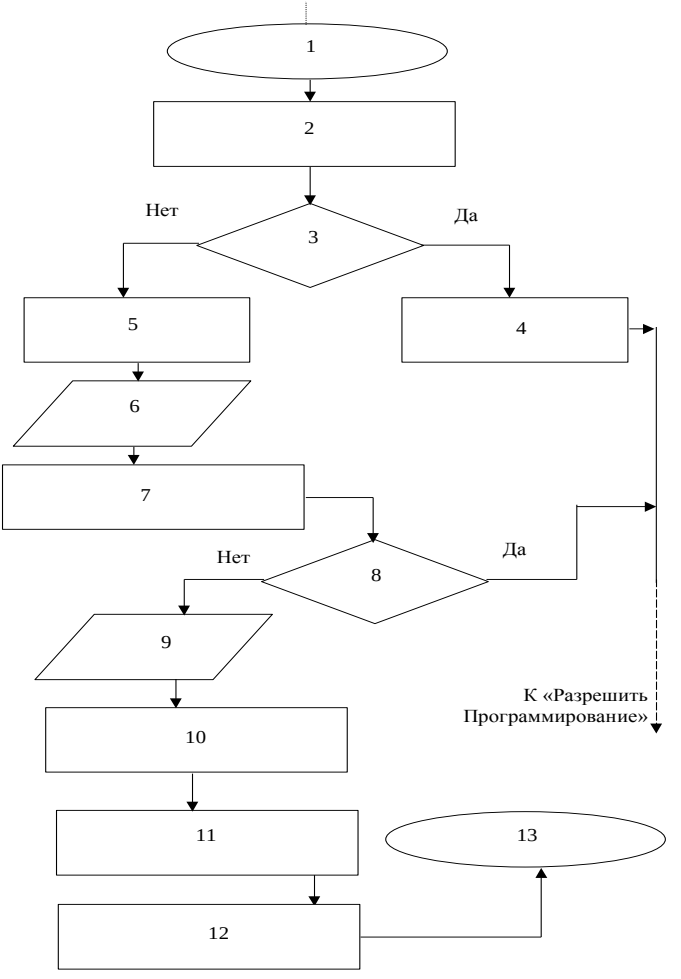


Рис. 8.3. Блок-схема тестирования контроллера

Работа блока тестирования в основной программе программирования контроллеров:

1 – инициализация блока Тестирование (Сравнение);

2 – присвоение переменной Bite очередной байт из файлового потока ввода/вывода;

3 – конец файла файлового потока ввода/вывода;

4 – возврат к «Разрешить программирование»;

5 – печатается адрес байта в память микросхемы;

6 – печатается байт;

7 – переменной Bite_Inside присваивается значение ячейки в микросхеме по адресу Adres;

8 – содержание памяти микросхемы не равно тому, что считали из файла?;

9 – вывод «Ошибка сравнения»;

10 – очистить порт;

11 – закрыть файловый поток ввода/вывода;

12 – вернуть операционной системе ошибку с № 1;

13 – завершение программы.

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

1. Номоконова Н.Н. Конспект лекций по дисциплине «Цифровые устройства и микропроцессоры». – Владивостокк: Изд-во ВГУЭС, 2003. – 50 с.

2. Попов С.В., Головкин А.Н. Программирование и отладка микросхем фирмы Motorola. – М.: Машиностроение, 1995. – 340 с.

3. Однокристальные микроЭВМ: Справочник / Бобрыкин А.В., Литвинский Г.В., Липовицкий Г.П. – М.: Бином, 1994. – 400 с.

4. Авраменко В.И. Современные PIC микроконтроллеры. – М: Радио и связь, 1997. – 240 с.

5. Мишель Ж. Программируемые контроллеры: Архитектура и применение / Пер. с фр. И.В. Федотова. – М.: Машиностроение, 1992. – 320 с.

6. Коффрон Дж., Лонг В. Промышленные программаторы / Под редакцией Нестерова П.В. – М.: Машиностроение, 1993. – 320 с.

7. Роберт Феджер. В поисках защиты для усталых глаз // Мир ПК. 1993. № 3. С. 19–21.

ПРИЛОЖЕНИЕ

СТЕНД УНИВЕРСАЛЬНЫЙ

Техническое описание и инструкция по эксплуатации

Содержание

1. Введение

Настоящее техническое описание и инструкция по эксплуатации (в дальнейшем – ТО) распространяется на стенд универсальный. ТО предназначено для описания стенда с целью обеспечения его сохранности и полного использования технических возможностей.

2. Назначение

Стенд предназначен для обучения студентов педагогических институтов, техникумов, учащихся ГПТУ основам автоматики, импульсной и вычислительной техники в процессе лабораторных занятий. Стенд может быть использован в качестве тренажера для поиска неисправностей в электронных схемах.

3. Технические данные

3.1. Стенд обеспечивает проведение цикла лабораторных работ по дисциплине «Цифровые устройства и микропроцессоры».

3.2. Стенд обеспечивает сборку электронных схем путем коммутации перемычками различных элементов, входящих в стенд. Стенд обеспечивает:

– наглядность процессов работы электронных схем;

– формирование хороших технических навыков в работе с электронными схемами;

– формирование навыков конструирования;

– формирование навыков в обнаружении неисправностей.

3.4. Основная элементная база стенда представлена интегральными схемами серии 155, 1804, 541.

3.5. Электропитание стенда осуществляется от источника напряжением +5В ± 0,25В.

3.6. Ток потребления стенда по цепи питания +58 не более З А.

3.7. Габариты стенда не более 390х260х90 мм.

3.8. Масса стенда не более 3,5 кг.

3.9. Срок службы стенда не менее 10 лет.

4. Устройство

4.1. Стенд состоит из корпуса и прикрепленной к нему печатной платы.

4.2. На печатной плате стенда смонтированы следующие функциональные узлы:

– формирователи сигналов без дребезга,

– блок переключателей,

– блок «ЛОГ. 1»,

– матрица R-2R,
– блок индикации 1,
– блок индикации 2,
– генераторы,
– одновибраторы,
– блок дискретных элементов,
– блок логических элементов,
– расширители.

4.3. Формирователи сигналов без дребезга (рис. П1) предназначены для формирования «чистых» положительных и отрицательных перепадов напряжения. Положительный перепад соответствует изменению напряжения от уровня Uн ≈ 0 В до уровня Uв ≥ +2,4 В. Отрицательный перепад соответствует изменению напряжения от уровня Uв ≥ + 2,4 В до уровня Uн ≈ 0 В, Уровень напряжения Uн ≤ 0,4 В является низким уровнем, он соответствует логическому нулю (лог. «0»). Уровень напряжения Uн ≤ 2,4 В является высоким уровнем, он соответствует логической единице (лог. «1»).

Формирователи сигналов без дребезга построены на основе R-S-триггеров и кнопок КМ-1 или переключателей П2К без фиксации. В соответствии с рис. П1 при отжатом положении кнопок SB1÷SB4 на штырях Р1÷К4 будут высокие уровни напряжения (лог. «1»), а на штырях Kl÷K4 будут низкие уровни напряжения (лог. «0»). В положении кнопок SB1÷S14 «нажато» на штырях Kl÷K4 будут низкие уровни напряжения (лог. «0»), а на штырях Kl÷K4 – высокие уровни напряжения (лог. «1»).

4.4. Блок переключателей (рис. П2) предназначен для задания уровней напряжения, соответствующих лог. «0» или лог. «1». Названный блок построен на основе тумблеров МТ1, или тумблеров ТП1-2, или переключателей П2К с фиксацией.

Для задания значений аргументов логической функции от любого элемента SAi (i<10) блока переключателей необходимо один конец перемычки подключить к штырю с номером i, другой конец – ко входу интегральной схемы (ИС).

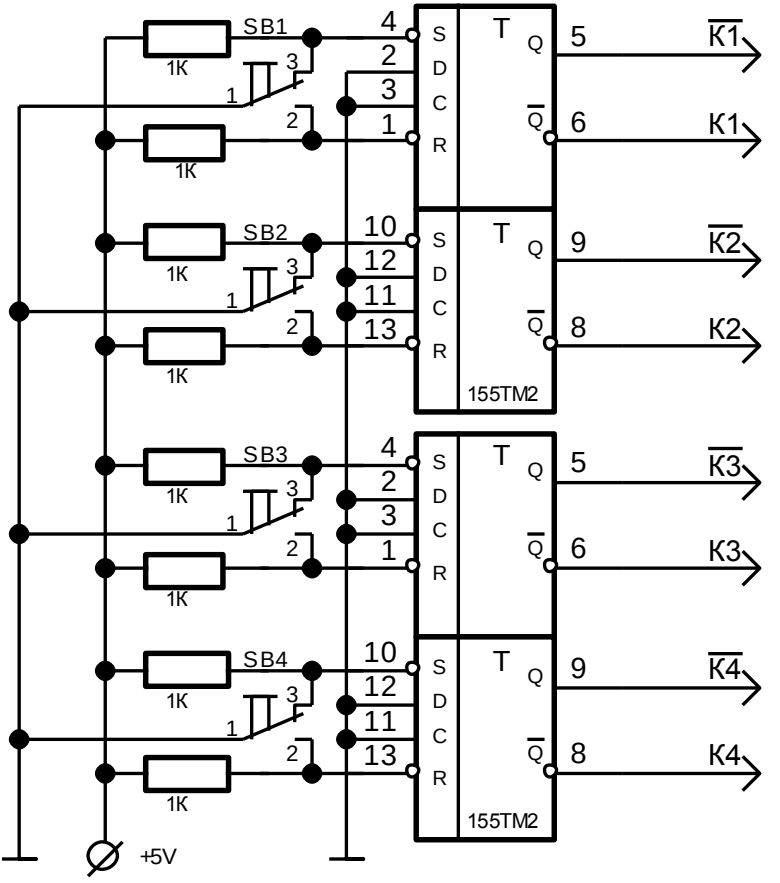


Рис. П1. Формирователь сигналов без дребезга

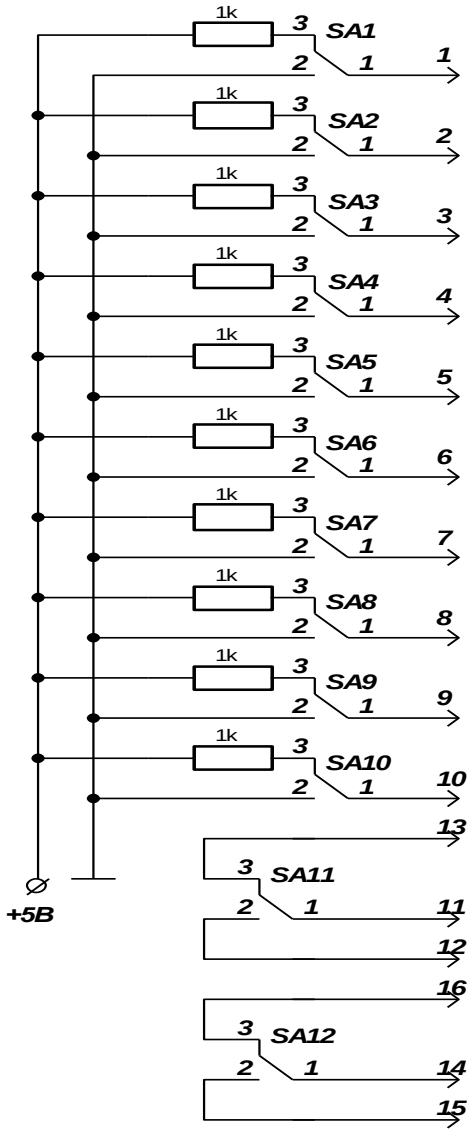


Рис. П2. Блок переключателей

Задание лог. «0» обеспечивается при положении рычажка тумблера «вниз» либо при положении переключателя П2К «отжато». Задание лог.

«1» обеспечивается при положении рычажка тумблера «вверх» либо при положении переключателя П2К «нажато».

4.5. Блок «лог.1», т.е. блок логических единиц (рис. П3), предназначен для стационарного задания уровня напряжения, соответствующего логической единице (лог. «1»).

Для стационарного задания лог. «1» на выходе ИС необходимо соединить перемычкой соответствующий вывод ИС с любым свободным штырьком блока «ЛОГ.1».

4.6. Матрица R-2R (рис. П4) имеет узко специальное назначение и используется в лабораторных работах по аналого-цифровым и цифро-аналоговым преобразователям (АЦП и ЦАП).

4.7. Блок индикации 1 (рис. П3) предназначен для визуального отображения состояний выходов логических схем. Названный блок содержит две линейки светодиодов: линейку А и линейку В. Линейка А представлена светодиодами красного свечения типа АЛ307БМ или ЗЛ341БМ Линейка В представлена светодиодами зеленого свечения типа АЛ307ВМ или ЗЛ341ВМ. Светодиоды обеих линеек управляются от трех интегральных схем К155ЛН2, каждая из которых содержит 6 инверторов с открытым коллекторным выходом. Входы этих инверторов соединены со штырьками, причем штыри и светодиод, стоящие на одной позиции, являются элементами одной электрической цепи. Таким образом, если соединить перемычкой выход исследуемой логической схемы со штырьком, стоящим на j-ой позиции линейки А, то на j-ом светодиоде названной линейки будет отображено состояние исследуемой логической схемы. В соответствии с рис. П5 светодиод будет светиться в том случае, если на входе инвертора (и, следовательно, на выходе исследуемой логической схемы) находится лог. «1». Если на входе инвертора – лог. «0», то светодиод не будет светиться.

В целях снижения нагрузки на источник питания +5В светодиоды линеек А и В записываются независимо друг от друга. Для подачи напряжения +5В на линейку А следует соединить перемычкой штырь, на котором соединены печатным проводником выводы резисторов, с одним ив двум рядом расположенных с ним штырей. Аналогично запитываются светодиоды линейки В.

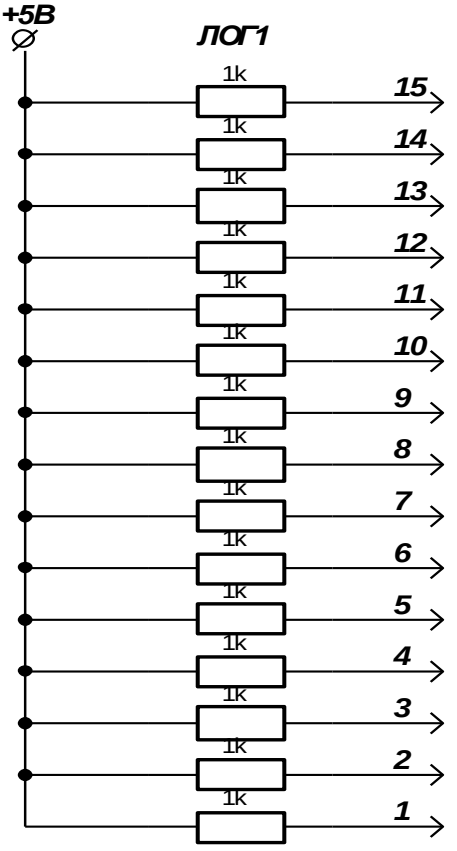


Рис. П3. блок логических единиц

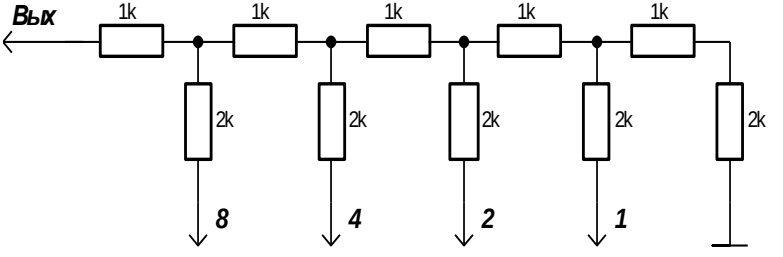


Рис. П4. Матрица R-2R

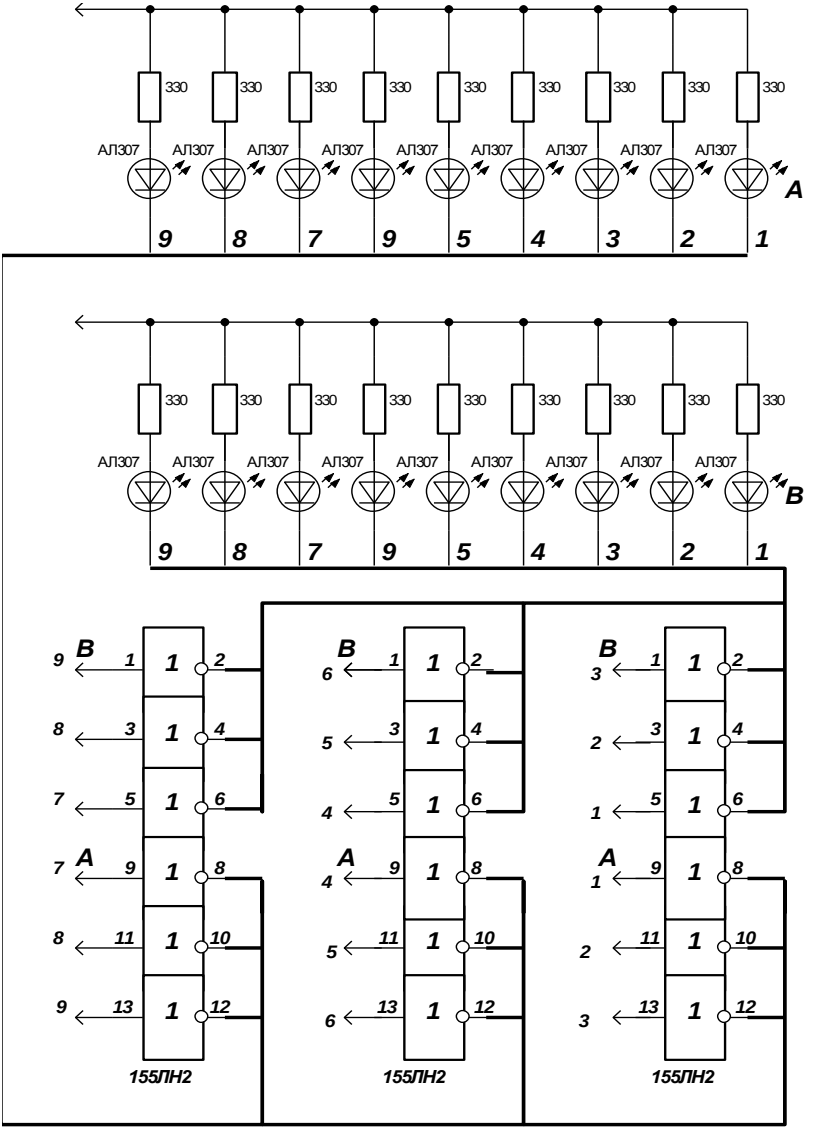


Рис. П5. Блок индикации 1

4.8. Блок индикации 2 (рис. П6) предназначен для отображения информации в виде цифр. Названный блок содержит четыре одинаковых узла, обозначенных буквами A, B, C, D. Узел включает преобразователь

двоичного кода в двоично-десятичный на ИС К155ПР7, дешифратор на ИС КР514ИД1 и полупроводниковый семисегментный индикатор АЛС324А. Информационные входы каждого узла подключены к штырям, расположенным на стенде справа от ИС К155ПР7 и пронумерованным цифрами 1, 2, 3, 4. Штырь с номером 1 соединен с выводом 5 ИС КР514ИД1. Штыри с номерами 2, 3, 4 подключены к выводам 10, 11, 12 ИС К155ПР7 соответственно. Штырь с номером 5 закоммутирован с управляющим выходом каждого узла.

Кроме перечисленных выше информационных входов каждый узел имеет управляющий вход гашения, выведенный на соответствующий штырь Г1-Г4. На стенде штыри Г1-Г4 расположены слева от ИС КР514ИД1. Подача на штыри Г1-Г4 низкого уровня напряжения (лог. «0») приводит к гашению соответствующего знакосинтезирующего индикатора АЛС324А.

Для отображения информации на знакосинтезирующих индикаторах необходимо подать на блок индикации 2 напряжение +5В. Делается это так: перемычкой соединяется штырь +5В, находящийся в правом верхнем углу блока индикации 2, со штырем, стоящим справа от ИС КР514ИД1.

В нижней части блока индикации 2 расположены две горизонтальные линейки соединенных между собой штырей, которые подключены к шине «общий» («земля»). Эти штыри используются для стационарного задания низкого уровня напряжения, соответствующего лог. «0».

4.9. Блок «генераторы» (рис. П7) включает два генератора, построенных на ИС К155ТЛ1: один (на DD-1) является высокочастотным (ВЧ) генератором (F = 4 мГц), другой (на DD-2) является низкочастотным генератором (НЧ) с регулируемой частотой $0{,}9<F<10$ Гц. Регулировка частоты последнего осуществляется переменным резистором СП4, стоящим на плате стенда на правой стороне блока «генераторы».

Для использования НЧ-генератора необходимо:

– при выключенном источнике питания +5В соединить перемычкой любой штырь из блока «ЛОГ.1» со штырем ЗАП.2 (запуск НЧ-генератора),

– подключиться к выходу НЧ-генератора (штырь ВЫХ.2),

– включить блок питания +5В.

В результате на выходе генератора должны появиться импульсы, частота следования которых может регулироваться переменным резистором.

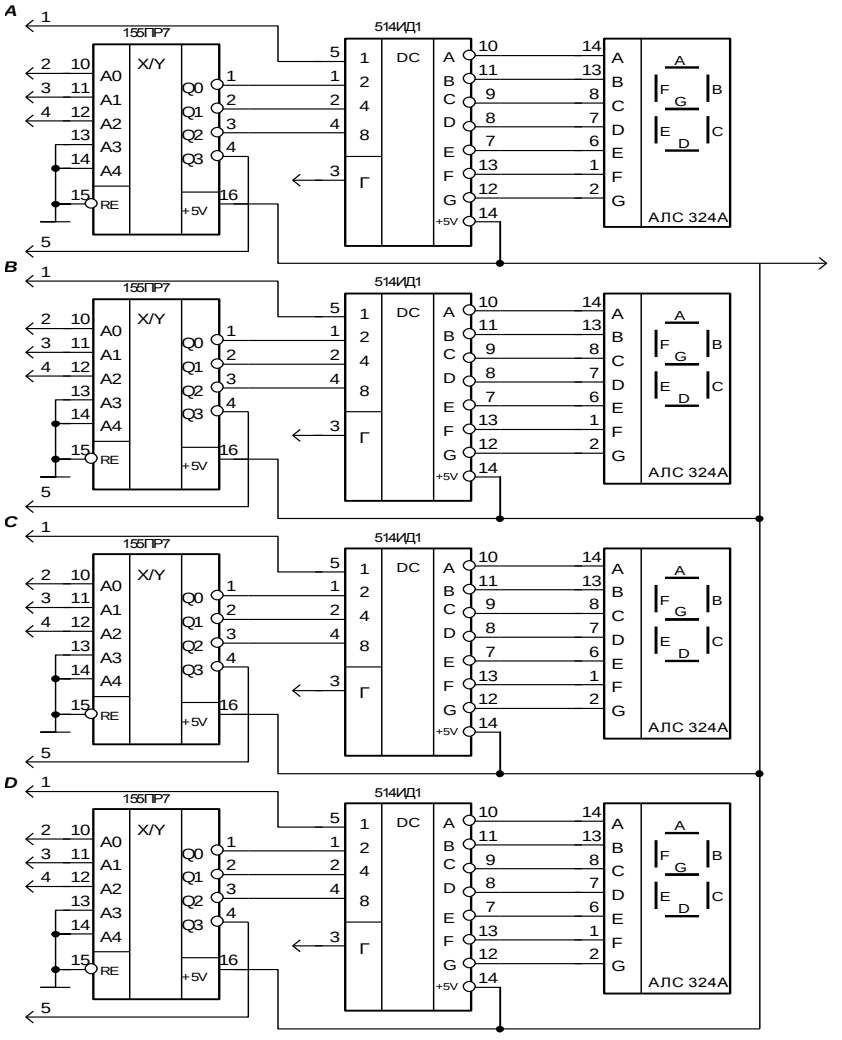


Рис. П6. Блок индикации 2

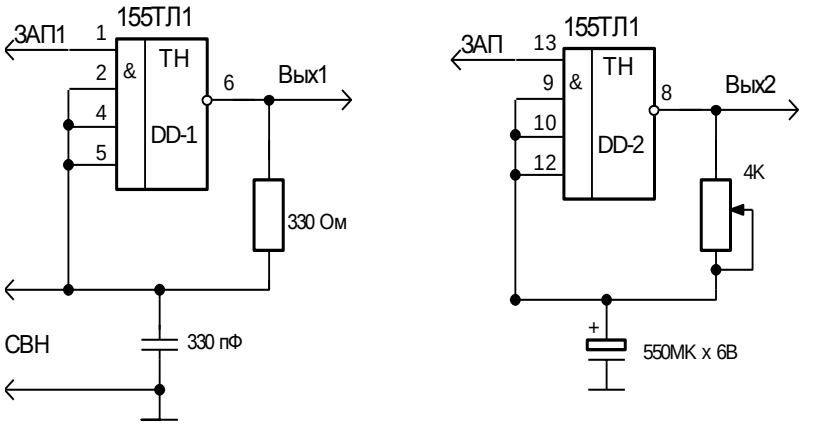


Рис. П7. Генераторы

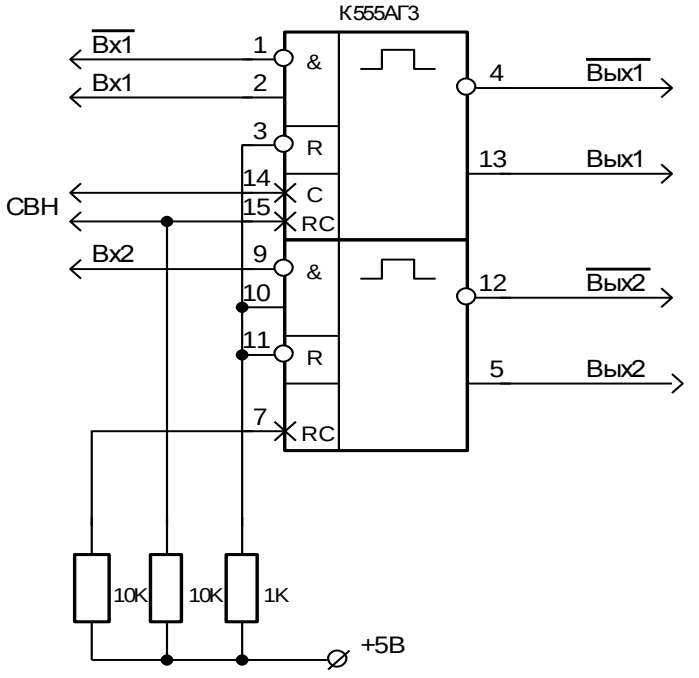


Рис. П8. Одновибраторы

Для использования ВЧ-генератора необходимо:

– при выключенном источнике питания +5В соединить перемычкой любой штырь блока «ЛОГ.1» со штырем ЗАП.1 (запуск ВЧ-генератора),

– подключиться к выходу ВЧ-генератора (штырь ВЫХ.1),

– включить блок питания +5В.

В результате на выходе генератора должны появиться импульсы длительностью по пьедесталу 100 нс (наносекунд, 1 нс=10^{-9} сек) и периодом следования Т=250 нс.

Часто в лабораторных работах возникает необходимость уменьшения частоты ВЧ-генератора. Для этого следует при отключенном источнике питания +5В параллельно конденсатору СВН генератора подключить нужный конденсатор из блока дискретных элементов.

4.10. Блок «одновибраторы» (рис. П8) включает два одновибратора, построенных на ИС К155АГ3. Первый одновибратор позволяет регулировать длительность формируемого импульса. Такая регулировка выполняется путем подключения к двум штырям СВН нужного конденсатора из блока дискретных элементов. Кроме того, первый одновибратор имеет два входа запуска $\overline{BX}.1$, *ВХ.1* и два выхода $\overline{ВЫХ}.1$, *ВЫХ.1*. Второй одновибратор имеет один вход запуска *ВХ.2* и два выхода $\overline{ВЫХ}.2$, *ВЫХ.2*.

Рассмотрим использование первого одновибратора. Если не подключать конденсатор СВН и оставить неподключенным вход *ВХ.1*, а на $\overline{BX}.1$ подать импульсы от генератора, то работа одновибратора будет описываться временной диаграммой, представленной на рис. П9.

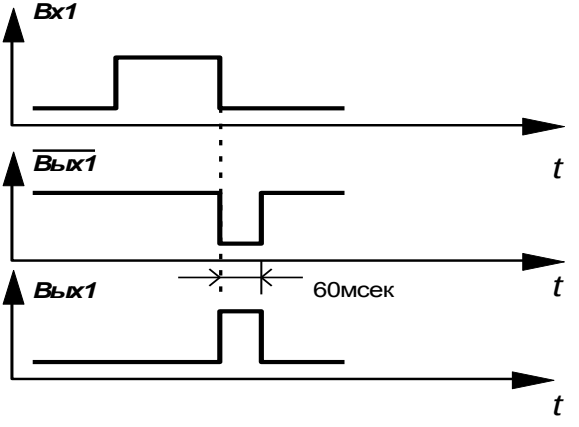


Рис. П9. Временная диаграмма работы первого одновибратора

Аналогичной временной диаграммой описывается работа второго одновибратора.

Если не подключать конденсатор СВН и оставить не подключенным $\overline{BX}.1$, а на *ВХ.1* подать импульсы от генератора, то работа первого одновибратора будет описываться временной диаграммой, представленной на рис. П10.

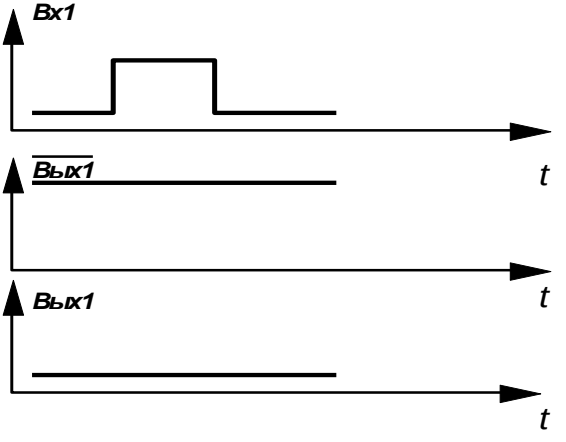


Рис. П10. Временная диаграмма работы второго одновибратора

4.11. Блок дискретных элементов (рис. П11) включает различные радиоэлементы. Каждый вывод радиоэлемента соединен с расположенным рядом с ним штырем. На средствах блока дискретных элементов можно смонтировать ряд лабораторных работ по источникам электропитания, триггерам, мультивибраторам, усилителям и т.д.

4.12. Блок логических элементов (рис. П12) представлен набором интегральных схем (ИС), который обеспечивает выполнение всего лабораторного практикума. Выводы каждой ИС соединены с расположенными рядом штырями. Исключение составляют выводы «+5В» и «общий», они не соединяются со штырями. Сделано это для того, чтобы обеспечить защиту «от дурака».

4.13. Поле расширителей предназначено для выполнения электрических соединений в тех случаях, когда вывод радиоэлемента необходимо подключить ко входам одного или нескольких радиоэлементов. Поле расширителей содержит 24–26 групп, каждая из которых включает по четыре или по пять штырей.

Примечание. В некоторых модификациях стенда расширитель 26 соединен с 16-м выводом розетки РС-16 (или Р6-16), отключенным от

шины питания +5Б. Названная розетка используется для установки ИС постоянного запоминающего устройства (ПЗУ) в процессе прожига ИС, т.е. занесения в нее информации. В этом случае на 16-ый вывод через расширитель 26 подается напряжение +15В. В режиме считывания информации из ИС ПЗУ К556РТ4 или при установке в розетку другой ИС, у которой Uпит +5В подается на 16-ый вывод, расширитель 26 подключается к штырю XS2-1(+5В).

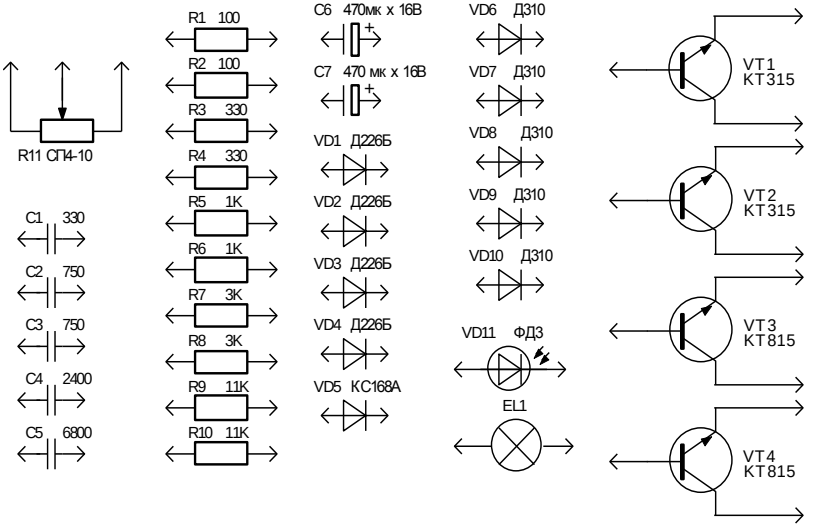


Рис. П11. Блок дискретных элементов

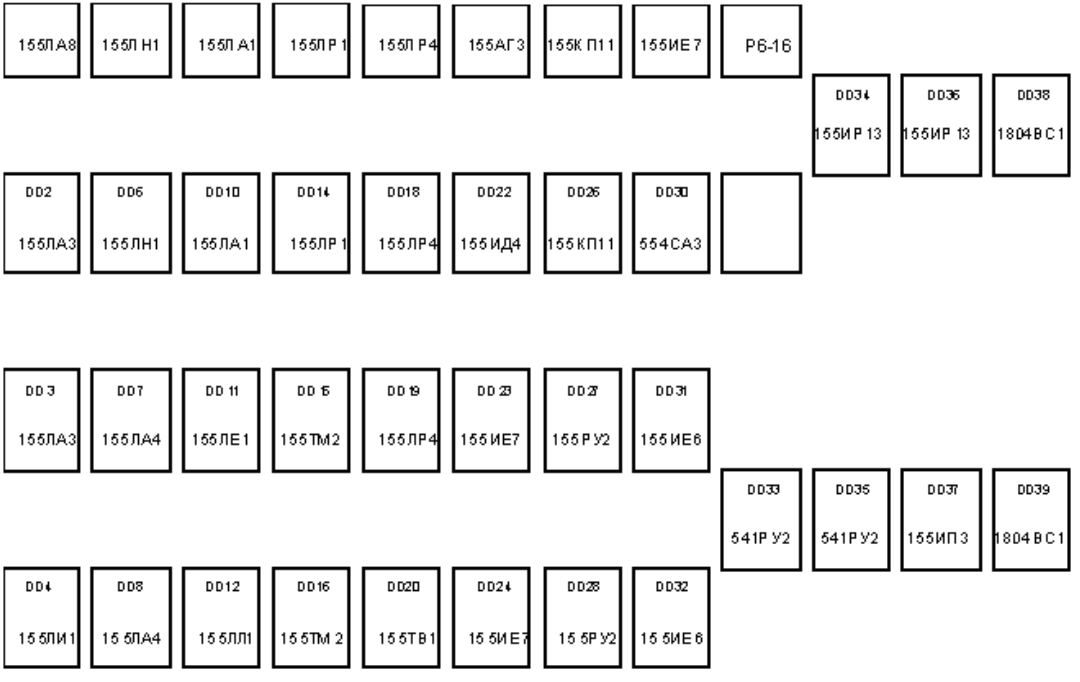


Рис. П12. Блок логических элементов

5. Указание мер безопасности

5.1. К работе со стендом допускаются лица, изучившие настоящее техническое описание, инструкцию по технике безопасности при работе с источниками электропитания, а также прошедшие местный инструктаж по безопасности труда.

5.2. При работе измерительные приборы (осциллограф, цифровой вольтметр и т.п.) и оборудование (источники питания) должны быть заземлены.

5.3. Съем и установку радиоэлементов, коммутацию перемычками производить при отключенном электропитании.

5.4. Монтажные работы на стенде производить при обязательном отсоединении проводов питания и «общий» от клемм стенда X51 и XS2. При монтаже использовать паяльник с температурой жала не более 260°С, время воздействия этой температуры на контакт микросхемы не более 3 с.

6. Подготовка к работе

6.1. После транспортировки стенда в зимнее время необходимо перед его включением визуально убедиться в том, что на печатной плате нет капелек росы. При их наличии необходимо выдержать стенд при комнатной температуре до полного его высыхания.

6.2. Внешним осмотром стенда убедиться в отсутствии механических повреждений платы, печатных проводников и радиоэлементов.

6.3. Источник электропитания, от которого будет запитываться стенд, выставить на номинальное напряжение +5В. Источник должен обеспечивать отклонение напряжения от номинала не более, чем +5% и ток нагрузки не менее 3 А.

6.4. Включить источник электропитания. Тестером убедиться в том, что на его выходе +5 В.

6.5. Выключить источник электропитания.

6.6. На выключенном источнике выход +5В подключить проводом МГШВ-0,35 красного цвета к клемме стенда XS2-1, под которой написано f53. Выход источника «общий» (⊥) подключить проводом НГШВ – 0,35 черного или белого цвета к клемме стенда XS2-2, под которой имеется знак «⊥».

6.7. Убедиться еще раз в правильности выполнения п.7.6.

ВНИМАНИЕ: не допускать переполюсовки, так как можно вывести стенд из строя.

6.8. Включить источник электропитания. Тестером убедиться в том, что на клемме XS2-1 напряжение на уровне +5В ± 0,25В относительно клеммы XS2-2.

6.9. Выключить источник электропитания. Стенд готов к работе.

7. Порядок работы

7.1. С помощью перемычек собрать схему в соответствии с выполняемой лабораторной работой.

7.2. Тестером проверить отсутствие («КЗ») между клеммами XS2-1 и XS2-2. При наличии «КЗ» найти его причину и устранить ее. При отсутствии «КЗ» перейти к п. 7.3.

7.3. Для лабораторных работ, связанных с блоком дискретных элементов, тестером проверить правильность выполненных соединений. Только после этого можно включить источник +5В. Для лабораторных работ, связанных лишь с блоком логических элементов, проверка тестером правильности выполненных соединений необязательна, поэтому после п. 2 можно включить источник +5В.

7.4. Все перекоммутации выполнять только при выключенном источнике +5В.

7.5. При необходимости выдачи сигналов на осциллограф для получения качественных осциллограмм следует внешнюю оплетку (экран) щупа осциллографа подключить к клемме стенда XS2-2 («общий» $\perp$).

7.6. Для удобства работы на стенде с измерительными приборами рекомендуется к токоведущей части щупа прибора закрепить внахлест с помощью изоленты предварительно зачищенный конец перемычки.

7.7. После окончания лабораторной работы выключить источник и только после этого демонтировать схему.

ОГЛАВЛЕНИЕ

Учебное издание

Номоконова Наталья Николаевна
Гаврилов Владимир Юрьевич

ЦИФРОВЫЕ УСТРОЙСТВА И МИКРОПРОЦЕССОРЫ

Практикум

В авторской редакции
Компьютерная верстка М.А. Портновой

Лицензия на издательскую деятельность ИД № 03816 от 22.01.2001

Подписано в печать .12.05. Формат 60×84/16.
Бумага писчая. Печать офсетная. Усл. печ. л..
Уч.-изд. л. Тираж экз. Заказ

Издательство Владивостокского государственного университета экономики и сервиса
690600, Владивосток, ул. Гоголя, 41
Отпечатано в типографии ВГУЭС
690600, Владивосток, ул. Державина, 57

80