МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)
Кафедра космического приборостроения и систем связи

УТВЕРЖДАЮ
Проректор по учебной работе
_____О.Г. Локтионова

«____» _____ 2017 г.

ПРОЕКТИРОВАНИЕ ЦИФРОВЫХ УСТРОЙСТВ

Методические указания к практическим занятиям по дисциплинам
«Проектирование цифровых устройств» и
«Вычислительная техника и информационные технологии»
для бакалавров направлений подготовки
11.03.03 «Конструирование и технология электронных средств» и
11.03.02 «Инфокоммуникационные технологии и системы связи»

Курск – 2017

Составитель О.Г. Бондарь

Рецензент
Кандидат технических наук, профессор кафедры
информационные системы и технологии В.А. Шлыков

Проектирование цифровых устройств : методические указания к практическим занятиям / Юго-Зап. гос. ун-т; сост.: О.Г. Бондарь. Курск, 2017. 31 с.: ил. 9. табл. 8. Библиогр.: с. 31.

Приведены задания для практических занятий.
Методические указания соответствуют требованиям программы, утвержденной учебно-методическим объединением по специальностям автоматики и электроники (УМО АЭ).
Предназначены для бакалавров направлений подготовки
11.03.03 «Конструирование и технология электронных средств» и
11.03.02 «Инфокоммуникационные технологии и системы связи»

Текст печатается в авторской редакции

Подписано в печать 5.07.17. Формат 60×84 1/16.
Усл. печ. л. *1,8*. Уч.- изд. л. *1,63*. Тираж 100 экз. Заказ 1340.
Бесплатно.
Юго-Западный государственный университет.
305040, г. Курск, ул. 50 лет Октября, 94

Содержание

Введение

Целью проведения практических занятий по курсу «Проектирование цифровых устройств» является закрепление материала, полученного при изучении лекций и развитие у студентов навыков и умения функционального проектирования и расчета простейших электронных устройств на базе цифровых микросхем, используемых для логического управления устройствами и процессами.

Студенты должны познакомиться с типовыми цифровыми интегральными схемами, научиться составлять комбинационные и последовательностные схемы, оптимизировать их и реализовывать их на типовых цифровых интегральных схемах.

1 Двоичное кодирование в цифровой технике

Цель занятия получить навыки кодирования чисел в системах счисления с различным основанием, преобразованием из системы в систему, представления чисел в различных форматах.

1 Представление целых чисел в двоичной, восьмеричной и шестнадцатеричной системах счисления. Правила записи чисел в позиционных системах счисления с произвольным основанием. Выполнение взаимных преобразований чисел из систем счисления с основанием 2, 8, 16. Правила преобразования чисел из систем с основанием 2^n в десятичную систему счисления.
2 Преобразование десятичных чисел в двоичную систему счисления. Преобразование целых чисел, преобразование дробных чисел.
3 Кодирование десятичных чисел двоичными. Форматы (упакованный и неупакованный).
4 Представление чисел со знаками. Прямой, обратный и дополнительный коды.
5 Представление чисел в формате с фиксированной точкой, плавающей точкой. Погрешности квантования.

Правила преобразования десятичных чисел в двоичные

Целые десятичные числа преобразуются последовательным делением на два с фиксированием дробной части на каждом шаге. Остатки, выписанные в обратном порядке и есть соответствующие двоичные числа.

Дробные числа преобразуются последовательным умножением на 2 с отбрасыванием целой части на каждом шаге. Отбрасываемые целые числа в прямом порядке образуют двоичное дробное число.

Задания

1 Преобразовать следующие числа 1110 0111, 1000 1101, 1011 1111 0001 1010, 1100 0011 0101 в восьмеричную и шестнадцатеричную систему счисления.
2 Преобразовать в десятичную систему счисления числа по п.1.
3 Выполнить преобразование в двоичную систему целых десятичных чисел 231, 141, 3125 и представить их в неупакованном и упакованном двоично-десятичном формате.
4 Записать в двоичном дополнительном коде десятичные числа 7, -11, 21, -28. При записи использовать 8-и разрядную сетку.

Контрольные вопросы

1 Сформулируйте правила записи чисел в позиционной системе счисления.
2 В чём преимущества систем счисления с основанием 8 и 16 при работе с цифровой техникой?
3 С какой целью используется обратный и дополнительный код?
4 Чем отличается упакованный и неупакованный двоично-десятичный формат представления десятичных чисел?
5 Какие другие способы кодирования десятичных цифр в двоичной системе счисления могут использоваться?

2 Минимизация логических функций

При подготовке к практической работе необходимо усвоить понятие базиса, функционально-полного базиса. Изучить методы минимизации логических функций.

Минимизация функций в аналитической форме

В процессе минимизации отыскиваются в СДНФ, или произвольной исходной аналитической форме конъюнкции, отличающиеся в одной переменной, и к ним применяется правило склеивания:

$x_1 z \vee \overline{x}_1 z = (x_1 \vee \overline{x}_1) z$

или поглощения:

$x_1 \vee x_1 z = x_1 (1 \vee z) = x_1$,

где z - отдельная переменная или логическое выражение. В результате две конъюнкции, связанные дизъюнкцией, заменяются одной, содержащей к тому же меньшее число переменных.

Пример:

$$Y=\overline{x_1 x_2 x_3} \vee \overline{x_1} x_2 x_3 \vee x_1 \overline{x_2} x_3 \vee x_1 x_2 \overline{x_3} = \overline{x_1} x_2 (x_3 \vee \overline{x_3}) \vee x_2 x_3 (\overline{x_1} \vee x_1) = \overline{x_1 x_2} \vee x_2 x_3$$

Минимизация функций графическим методом

При небольшом числе переменных удобно использовать для минимизации карты Карно, которые представляют разновидность табличного способа задания логической функции. Особенностью карт Карно является то, что конъюнкции, отличающиеся в одной переменной, оказываются рядом. И если в соседних двух клетках функция равна единице, то возможна операция склеивания по одной переменной, если четыре единицы функции расположены квадрате 2х2 или на одной линии, то возможно склеивание по двум переменным, а если восемь единиц расположены в прямоугольнике 2х4, то возможно склеивание по трем переменным. Карта Карно для трех переменных представлена на рисунке 1.

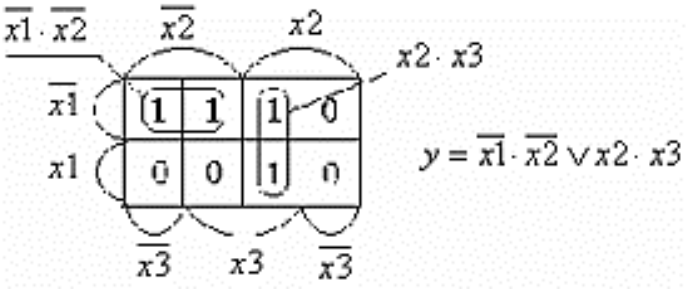


Рисунок 1 – Карта Карно

Выполняя операцию склеивания в представленном случае над двумя парами единиц так, как показано на диаграмме, можно упростить или минимизировать выражение. В результате предложенная функция запишется в следующем виде:

$$Y = x_1\overline{x_2} \vee x_2x_3$$

Задания

1 Представить функции таблицы 1 в аналитической виде в форме СДНФ и СКНФ.

2 Минимизировать полученные функции аналитически.

3 Минимизировать функции, заданные таблицами истинности графически.

Таблица 1. Логические функции

№	*x*	*y*	*z*	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}
0	0	0	0								1	1	
1	0	0	1	1	1						1		
2	0	1	0				1	1	1		1	1	
3	0	1	1	1	1			1					1
4	1	0	0		1	1		1					
5	1	0	1	1		1			1	1			1
6	1	1	0			1	1		1	1			
7	1	1	1				1			1		1	1

3 Элементы проектирования комбинационных устройств

Процесс проектирования цифрового устройства состоит в получении его формального описания в одной из возможных форм, определении размерности входных и выходных переменных, декомпозиции устройства при его высокой сложности, минимизации (при необходимости), приведении к заданному функциональному базису и построение функциональной схемы с учётом ограничений на количество входов и нагрузочную способность цифровых интегральных схем.

Формализация задания

Описание цифрового устройства может осуществляться аналитически, таблично, схемой, временной диаграммой или неформальным языком (описательно). Для проектирования устройства его описание следует привести к аналитическому или табличному виду.

В качестве примера приведём описание устройства, управляющего светом в подъезде жилого дома.

Свет в подъезде может быть включён принудительно жильцами дома, от датчика движения при наличии сигнала фотодатчика (тёмное время суток).

Чаще всего переход от описательного задания осуществляется к табличной форме представления.

К табличной форме обычно переходят и от представления устройства совмещёнными временными диаграммами. Здесь не затрагивается выбор способа кодирования логических переменных.

Декомпозиция устройства

При большом числе логических переменных задача синтеза устройства становится чрезвычайно громоздкой и необходимо прибегать к декомпозиции сложного устройства (разложению на более простые устройства). Общих правил декомпозиции не

существует. Обычно сложное устройство пробуют разложить на основе методов горизонтальной, функциональной, вертикальной декомпозиции. Хорошо известным примером может служить метод поразрядной (горизонтальной) декомпозиции сумматора.

Минимизация описания логической функции

Этот этап не всегда является необходимым, т.к. иногда бывает важнее свойство регулярности структуры, чем аппаратные затраты. Практические методы минимизации были изучены ранее.

Приведение к заданному базису

На этом этапе достигается приведение к существующей номенклатуре применяемых интегральных схем или элементов.

Построение функциональной схемы устройства

Исходным для этого этапа является аналитическое выражение функции. При составлении функциональной схемы устройства учитываются естественные ограничения на количество входов элементов интегральных схем (ИС) и их нагрузочную способность.

Задания

Привести функции, минимизированные на практическом занятии 2 к одному из указанных базисов (И-НЕ, ИЛИ-НЕ, И-ИЛИ-НЕ).

4 Характеристики и параметры цифровых элементов. Совместимость интегральных микросхем

К основным характеристикам и параметрам цифровых элементов, порождающих особенности их применения, относятся входные и нагрузочные характеристики.

Входные характеристики определяются зависимостью входного тока от напряжения на входе. Выраженные особенности имеют характеристики ДТЛ и ТТЛ элементов. Они проявляются в изменении направления входного тока и значительном изменении его величины. Существенное влияние при работе на высоких частотах вносит входная ёмкость.

Выходные характеристики представляют собой зависимость уровня логического нуля и логической единицы от величины нагрузки. Для элементов одной серии используют понятие нагрузочной способности, выражаемой количеством однотипных элементов, которое можно подключить в выходу логического элемента. При использовании в составе одного устройства различных серий элементов, целесообразно пользоваться максимальными токами логического нуля и единицы.

Цифровые элементы могут использоваться для управления индикаторами и даже электромагнитными реле.

Основными элементами индикации выступают светодиоды. Нелинейность их вольтамперной характеристики вынуждает включать резисторы последовательно со светодиодом, которые и определяют величину протекающего тока. Для современных КМОП элементов с большим выходным током (до 30 мА) способ управления светодиодом безразличен (логическим нулём или единицей). Для ТТЛ элементов втекающий ток логического нуля многократно (до 40 раз) превышает вытекающий ток логической единицы. Поэтому используется вариант подключения анода диода к положительному полюсу источника питания, а управление осуществляется по катодной цепи (рисунок 4.1).

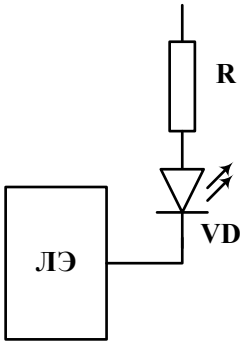
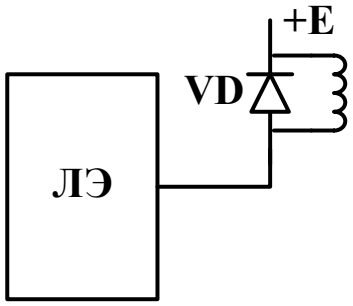


Рисунок 4.1 – Подключение светодиода к ЛЭ

К логическим элементам могут быть подключены индуктивные нагрузки, такие как высокоомные акустические преобразователи, обмотки электромагнитных реле. В этом случае для защиты логического элемента от пробоя импульсом самоиндукции параллельно индуктивному элементу включают диод (рисунок 4.2). Следует помнить, что величина выходного тока не должна превышать максимальное значение допустимого для данного элемента.



Сложные системы могут быть построены на основе различных серий цифровых ИС. При этом в состав систем могут входить микропроцессоры и микроконтроллеры с различными уровнями напряжения питания, программируемые логические интегральные схемы, логические элементы ТТЛ и КМОП типа, в том числе с разными напряжениями питания. При этом перед разработчиком стоит задача сопряжения различающихся логических уровней.

Обычно задача сопряжения решается с помощью специализированных ИС – преобразователей уровней.

В многих случаях можно обойтись без их применения. Рассмотрим несколько таких случаев.

Сопряжении КМОП и ТТЛ ИС

Полагается, что источником сигналов является КМОП ИС, а приёмником – ТТЛ. Здесь возможны два случая.

1 Напряжение питания КМОП ИС равно напряжению питания ТТЛ ИС или ниже его (3,3 В). В такой ситуации возможно непосредственное соединение выходов КМОП и ТТЛ ИС, т.к. логические уровни КМОП ИС при умеренной нагрузке мало отличаются от соответствующих уровней ТТЛ ИС. При этом

следует обращать внимание на выходные токи логического нуля КМОП ИС, так как некоторые серии элементов (К561, К564, К1564) имеют ток логического нуля при напряжении питания +5 В от 0,2 мА, что исключает их непосредственное подключение к ИС ТТЛ серий кроме К1533 (ТТЛШ) с входным током логического нуля - 0,2 мА. Серия КМОП ИС 1554 может непосредственно подключаться к любым ТТЛ и ТТЛШ ИС, т.к. её выходные токи не менее 24 мА.

2 Напряжение питания КМОП ИС выше напряжения питания ТТЛ (ТТЛШ) ИС. Максимальный выходной ток КМОП элементов растёт с ростом напряжения питания. Так для серии К561 при напряжении питания 10 В он достигает 1,3 мА (как при логическом 0, так и при логической 1 на выходе), а для серии К1561 при напряжении питания 15 В он достигает уже 2,4 мА. В отличие от предыдущего случая здесь необходимо дополнительно обеспечить согласование по уровням напряжения. При напряжении питания 10 В выходное напряжение надо уменьшить в 3 раза. При этом сопротивление делителя должно быть не менее 10В/1,3мА=8 кОм. Поскольку сопротивление нижнего плеча делителя должно быть в два раза меньше сопротивления верхнего плеча делителя, то его величина превысит 2,4 кОм, что не позволит обеспечить нормальный логический уровень нуля. Выходом из положения может стать шунтирование диодом Шоттки нижнего плеча делителей. Такое решение возможно в единичных случаях, и не является ни компактным, ни дешёвым. В этой ситуации следует прибегать к применению специализированных преобразователей уровня.

Сопряжение ТТЛ (ТТЛШ) и КМОП ИС

При любых соотношениях напряжений питаний кроме питания КМОП ИС напряжением 3,3 В, уровень логической единицы на выходе источника (ТТЛ, ТТЛШ) меньше требуемого значения на входе приёмника (КМОП). При этом уровни логического нуля соответствуют друг другу. Самый простой способ согласования

достигается применением ТТЛ (ТТЛШ) ИС с открытым коллектором. В этом случае нагрузочный резистор подключается к источнику питания КМОП ИС.

Следует иметь в виду с точки зрения работоспособности сопряжения величина резистора определяется только допустимым падением напряжения на нём от тока утечки закрытого выходного транзистора ТТЛ ИС и входного тока КМОП ИС. При этом высокоомный резистор улучшает экономичность схемы. Однако с точки зрения быстродействия следует снижать величину сопротивления, т.к. через подтягивающий резистор осуществляется перезаряд выходной ёмкости ТТЛ ИС, соединительной линии и входной ёмкости КМОП ИС. При этом увеличивается задержка срабатывания последней как при включении, так и выключении.

Сопряжение ИС со стандартным напряжением питания с низковольтными КМОП ИС

Эта ситуация часто возникает при сопряжении цифровых устройств на ИС со стандартным питанием и микроконтроллеров (МК) и ПЛИС использующих низковольтное питание. В отдельных случаях такого сопряжения не требуется так как входы МК или ПЛИС бывают **толерантны** к напряжению 5 В. При отсутствии толерантности для сопряжения применяют делитель напряжения, рассчитываемый из условий получения требуемого уровня логической единицы, не превышения им уровня питания и обеспечения требуемого быстродействия.

Задания

1 Подключить к логическому элементу ТТЛШ серии (К1533) светодиод. Выбрать тип светодиода, пользуясь справочником. Рассчитать величину последовательного резистора.
2 Выполнить сопряжение цифрового датчика с выходным каскадом на КМОП ИС серии К561 с напряжением питания +5В и

микроконтроллера, выполненного по КМОП технологии с напряжением питания 2,7 В (3,3 В). Определить задержку срабатывания, полагая что уровни срабатывания равны соответственно 0,4 Еп и 0,7 Еп, где Еп – напряжение питания МК.

Контрольные вопросы

1 Дайте определение цифрового сигнала. Сравните диапазоны напряжений, воспринимаемых логическими элементами ИС ТТЛШ и КМОП как «0», «1». Сравните диапазоны напряжений, формируемых на выходах логических элементов ИС ТТЛШ и КМОП как «0», «1». Напряжение питания ИС +5В.
2 Какие проблемы возникают при сопряжении ИС с различными напряжениями питания? Опишите способы согласования уровней, назначение и расчёт подтягивающих резисторов.
3 Опишите входные, выходные токи и их направления для логических элементов ТТЛ. Дайте понятие нагрузочной способности логического элемента.
4 Опишите входные, выходные токи и их направления для логических элементов КМОП. Дайте понятие нагрузочной способности логического элемента.
5 Какой элементной базой (КМОП или ТТЛШ) пользовались бы вы при разработке цифровой части, например, мобильного телефона? Почему?
6 С какими напряжениями питания работают ИС мобильного телефона?
7 К чему приведёт повышение напряжения от 5 до 7 В, формируемого блоком питания для ТТЛШ ИС? Дать обоснования.

5 Обнаружение рисков сбоя на основе трёхзначного моделирования

Переходные процессы в цифровых устройствах могут приводить *рисков* – сигналов на выходе схемы, не предусмотренных логикой ее работы.

Риски принято подразделять на *динамические* и *статические*. Последние могут быть как *логическими,* так и *функциональными*. *Статические риски* – кратковременные изменения выходного сигнала, который согласно логике работы схемы должен оставаться неизменным.

Если при изменении входных сигналов (при смене наборов входных переменных) сигнал на выходе схемы должен оставаться единичным, а наблюдается переход 1→0→1, то риск называется *единичным* или *1-риском* (рисунок 1,а).

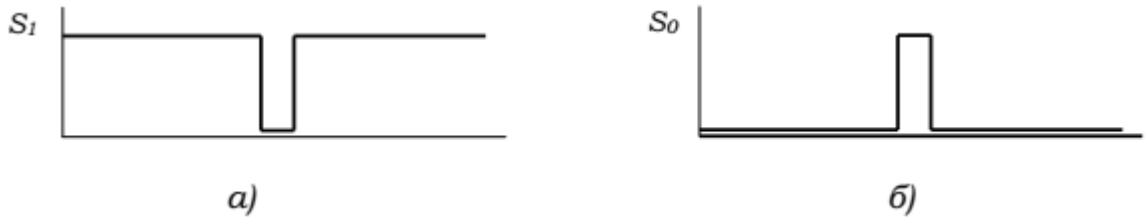


Рисунок 5.1 – Статические риски сбоя

Если при изменении входных сигналов сигнал на выходе схемы должен оставаться нулевым, а наблюдается переход 0→1→0, то риск называется *нулевым* или *0-риском* (рисунок 1,б).

Динамические риски – многократные изменения выходного сигнала, который по логике работы схемы должен измениться однократно. Риск называется динамическим *D*+ в том случае, когда при изменении входных сигналов выходной сигнал должен измениться с низкого уровня на высокий (0→1), а наблюдается переход 0→1→0 →1 (рисунок 2,а). Риск называется динамическим *D*- в том случае, когда при изменении входных сигналов выходной сигнал должен измениться с высокого уровня на низкий (1→0), а наблюдается переход 1→0 →1→0 (рисунок 2,б).

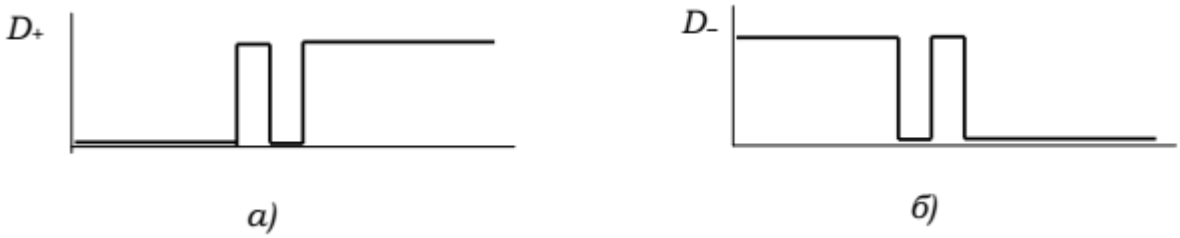


Рисунок 5.2 – Динамические риски сбоя

Логическим называется статический риск, проявляющийся при *соседней* смене наборов входных переменных (при переходе по коду Грея). Логический риск поддается устранению путем изменения логической структуры схемы, реализующей булеву функцию. *Функциональным* называется риск, проявляющийся при *много местной* смене наборов входных переменных (при не соседнем пере ходе). Такие риски характеризуют саму логическую функцию, а не способ ее реализации.

Анализ логической схемы на предмет выявления различного вида рисков является процедурой оценки функциональной устойчивости этой схемы. Статические риски считаются наиболее неблагоприятными, именно на обнаружение статических рисков направлены основные усилия разработчиков, привлекающих для решения этой задачи глубоко развитый математический аппарат.

Структурный метод

С*труктурный метод* предназначен для работы исключительно с логическими рисками. Однако он не только позволяет их определить, но и предлагает алгоритм устранения. Основан этот метод на анализе карты Карно, используемой для минимизации рассматриваемой функции (напомним, что логические риски проявляются только при соседних переходах).

Правила выявления и устранения статического логического риска: если импликанты на карте Карно образуют разобщенные контуры, схема содержит логический риск на соседних переходах между этими контурами.

Для устранения рисков на карте Карно необходимо выделить дополнительные импликанты, объединяющие разобщенные контуры. К логическому выражению, определяющему реализуемую функцию, таким образом, добавятся *термы согласования*, сохраняющие при переходе между соседними наборами переменных необходимое логическое значение. Реализация функции будет свободна от логических рисков, если все соседние клетки карты Карно покрыты хотя бы одной импликантой – то есть если все соседние переходы совершаются без пересечения общего контура импликант.

Метод трехзначного моделирования

Метод трехзначного моделирования позволяет выявить все виды статических рисков – как функциональные, так и логические. В *трехзначной* (или *троичной*) логике, помимо основных значений 0 и 1, переменные могут принимать третье, обозначаемое символом x. Значение x приписывается уровню сигнала, о котором разработчик ничего не может знать наверняка (например, сигналу во время переходного процесса), и называется *состоянием неопределенности*. Состояние x, поступая на вход логических элементов в зависимости от выполняемой ими логической операции, может порождать на их выходе либо такое же неопределенной состояние, либо одно из определенных состояний: 0 или 1. Поведение логических элементов задается при этом одним из видов троичной логики (таблица 2).

Таблица 2. Троичная логика для разных элементов

НЕ		*b*
a	**0**	1
	x	x
	1	0

ИЛИ		*b*		
		0	**x**	**1**
a	**0**	0	x	1
	x	x	x	1
	1	1	1	1

И		*b*		
		0	**x**	**1**
a	**0**	0	0	0
	x	0	x	x
	1	0	x	1

При трехзначном моделировании реакция схемы определяется как для начального и конечного наборов (N_m и $N_к$), являющихся границами перехода, так и для самого переходного вектора $N_m \rightarrow N_к$.

Пусть на вход схемы, реализующей логическую функцию n переменных $F(x_{n-1}, x_{n-2}, \ldots, x_0)$ подаются последовательно два набора переменных N_m: $a_{n-1}a_{n-2}\ldots a_0$; $N_к$: $b_{n-1}b_{n-2}\ldots b_0$. *Переходный вектор*: $N_m \rightarrow N_к$: $p_{n-1}p_{n-2}\ldots p_0$ строится следующим образом:

$$p_i = \begin{cases} \text{x} & при\ \ a_i \neq b \\ a_i & при\ \ a_i = b_i \end{cases} \qquad \forall\, i = \overline{0, n-1}. \qquad (1)$$

Правило выявления статического риска: схема содержит статический риск, если значения функции F на начальном и конечном наборах совпадают, а на переходном векторе состояние функции не определено. Метод трехзначного моделирования гарантирует выявление только статических рисков – непосредственное наличие динамических рисков этот метод не показывает. Однако в схемах, построенных на элементах И, ИЛИ, НЕ (И-НЕ, ИЛИ-НЕ), динамический риск на выходе схемы всегда является результатом присутствия статического риска в предыдущем каскаде (рисунок 5.3). Поэтому наличие динамического риска *можно предположить* в том случае, если на промежуточном этапе анализа схемы для переходов, не сохраняющих значение функции, обнаружены статические риски.

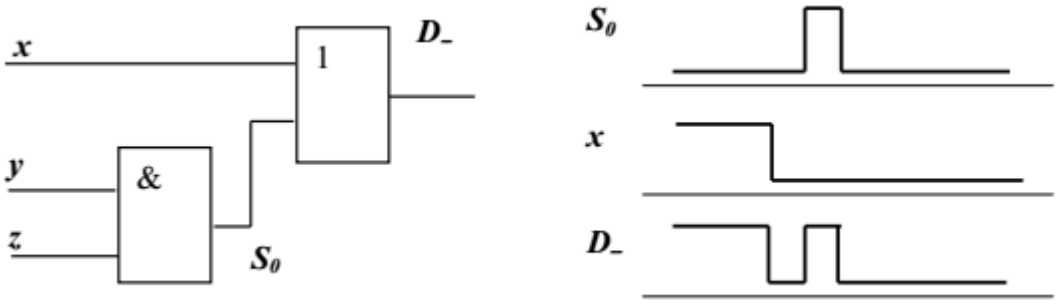


Рисунок 5.3 – Статический риск, вызывающий динамический

Задания

1 Для функции *F,* заданной таблицей 3 истинности структурным методом найти и устранить логические риски. Построить функциональные схемы исходную и с устранением риска.

Таблица 3 истинности функции F

№ набора	*a*	*b*	*c*	*d*	***F***	№ набора	*a*	*b*	*c*	*d*	***F***
0	0	0	0	0	**1**	8	1	0	0	0	**0**
1	0	0	0	1	**1**	9	1	0	0	1	**0**
2	0	0	1	0	**0**	10	1	0	1	0	**0**
3	0	0	1	1	**0**	11	1	0	1	1	**0**
4	0	1	0	0	**1**	12	1	1	0	0	**0**
5	0	1	0	1	**1**	13	1	1	0	1	**0**
6	0	1	1	0	**1**	14	1	1	1	0	**1**
7	0	1	1	1	**1**	15	1	1	1	1	**1**

2 Методом трёхзначного моделирования проанализировать функциональную устойчивость при некоторых несоседних переходах (по указанию преподавателя).

Контрольные вопросы

1 Дайте определение рисков сбоя (статических и динамических, логических и функциональных) в цифровой схеме. Объясните причины возникновения различного вида рисков.
2 Для выявления и/или устранения каких видов рисков в логических схемах предназначен структурный метод?
3 Для чего используется метод трехзначного моделирования?

6 Основы функционирования последовательностных устройств

Последовательностные устройства в соответствии с обобщённой схемой могут быть реализованы как автоматы Мура и Мили. При этом их синтез сводится к синтезу одной или двух комбинационных схем. В случае автомата Мура синтезируется лишь

комбинационная схема функций возбуждения, а функция выхода воспроизводится состояниями бистабильных ячеек. Для автомата Мили необходимо синтезировать как комбинационную схему функций возбуждения, так и функций выхода. На практике при синтезе используется преимущественно словарный метод. В соответствии с ним задаётся тип ячеек памяти (тип триггера), влияющий на функции возбуждения.

При табличном методе синтеза строится так называемая таблица переходов (эквивалент таблицы истинности для комбинационных схем). В качестве входных переменных она включает входные переменные и переменные, характеризующие текущее состояние элементов памяти. Каждому сочетанию набора входных переменных и текущему состоянию ставится в соответствие следующее состояние автомата. На основе анализа текущего и следующего состояния каждого элемента памяти определяется значение его функции перехода (сохранение 0, сохранение 1, переход к единице, переход к 0). Сопоставление функций перехода и таблиц функционирования элементов памяти выбранного типа позволяет определить функции возбуждения, т.е. значения логических переменных на выходе первой комбинационной схемы.

Входные сигналы первой комбинационной схемы являются входными сигналами и второй комбинационной схемы автомата Мили.

Задания

Синтезировать квадратурный импульсный генератор на основе динамических D-триггеров. Опорный генератор меандра с логическими уровнями, соответствующими уровням применённых триггеров считать существующим.

Контрольные вопросы

1 Изобразить временные диаграммы динамического D-триггера.

2 Построить таблицу функционирования D-триггера.
3 Построить граф-схему синтезированного автомата.

7 Автоматы Мили и Мура

Табличный способ задания автоматов Мили сводится к двум таблицам – таблице переходов и таблице выходов. Пусть автомат имеет состояния a_0, a_1, a_2, где a_0 – начальное состояние. Поведение автомата определяется совокупностью входных сигналов x_1, x_2, x_3 из множества X, а значения выходов w_1, w_2, w_3. Тогда функция переходов - δ задаётся таблицей переходов (таблица 4), а функция выходов - λ таблицей выходов (таблица 5).

Таблица 4 переходов

δ	a_0	a_1	a_2
x_1	a_0	a_2	a_1
x_2	a_2	a_1	a_0
x_3	a_1	a_0	a_2

Таблица 5 выходов

λ	a_0	a_1	a_2
x_1	w_1	w_1	w_3
x_2	w_3	w_2	w_2
x_3	w_3	w_2	w_2

Для задания автомата Мура достаточно добавить строку в таблицу переходов, так как значение функции однозначно соответствует состоянию автомата (таблица 6).

Таблица 6. Автомат Мура

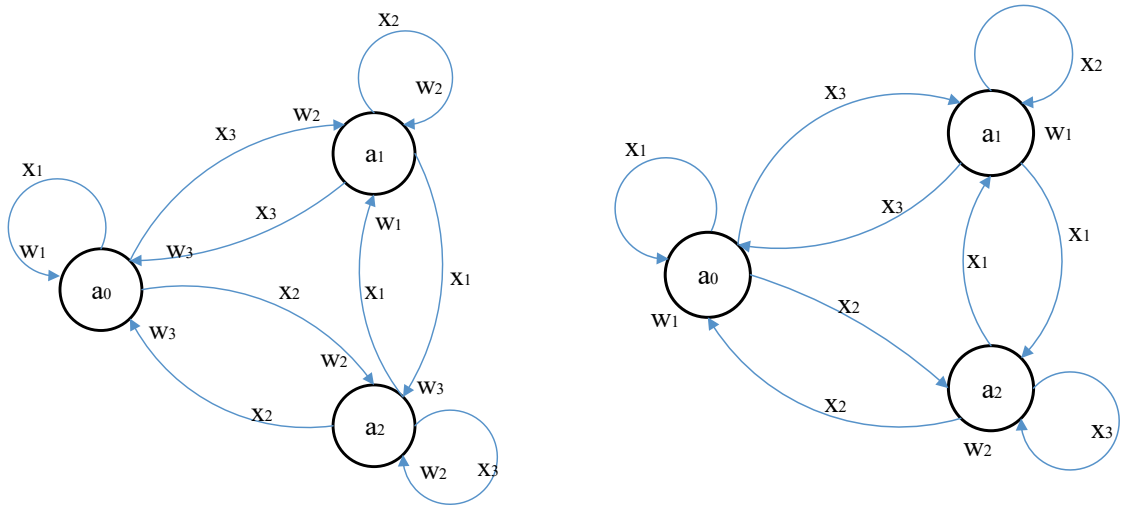
λ	w_1	w_1	w_2
δ	a_0	a_1	a_2
x_1	a_0	a_2	a_1
x_2	a_2	a_1	a_0
x_3	a_1	a_0	a_2

Автоматы можно задать с помощью граф схемы. Вершины автомата – это его состояния. Стрелки соединяющие вершины –

переходы. Над переходами пишутся значения входных переменных, вызывающих переход в следующее состояние. Значения выхода для автомат Мили указывается над стрелкой перехода, а для автомата Мура – у состояния автомата.

Задания

По приведенным на рисунках 7.а и 7.б графам автоматов Мили и Мура, соответственно, построить автоматные таблицы.



а) б)

Рисунок 7 – Графы автоматов: а) Мили, б) Мура

Контрольные вопросы

Что понимается под состоянием автомата?
Что такое функция перехода?
Что такое функция возбуждения?
Что означает – словарный метод синтеза?

8 Кодирование по паритету. Кодирование по Хеммингу

Самоконтролирующиеся коды

Коды Хемминга являются самоконтролирующимися кодами, то есть кодами, позволяющими автоматически обнаруживать ошибки при передаче данных. Для их построения достаточно приписать к каждому слову один добавочный (контрольный) двоичный разряд и выбрать цифру этого разряда так, чтобы общее количество единиц в изображении любого числа было, например, нечетным. Одиночная ошибка в каком-либо разряде передаваемого слова (в том числе, может быть, и в контрольном разряде) изменит четность общего количества единиц. Счетчики по модулю 2, подсчитывающие количество единиц, которые содержатся среди двоичных цифр числа, могут давать сигнал о наличии ошибок. При этом невозможно узнать, в каком именно разряде произошла ошибка, и, следовательно, нет возможности исправить её. Остаются незамеченными также ошибки, возникающие одновременно в двух, четырёх, и т.д. — в четном количестве разрядов. Впрочем, двойные, а тем более четырёхкратные ошибки полагаются маловероятными.

Самокорректирующиеся коды

Коды, в которых возможно автоматическое исправление ошибок, называются самокорректирующимися.

Основными характеристиками самокорректирующихся кодов являются:

1 Число разрешенных и запрещенных комбинаций. Если n - число символов в блоке, r - число проверочных символов в блоке, k - число информационных символов, то 2^n - число возможных кодовых комбинаций, 2^k - число разрешенных кодовых комбинаций, 2^n-2^k - число запрещенных комбинаций.
2 Избыточность кода. Величину r/n называют избыточностью корректирующего кода.
3 Минимальное кодовое расстояние. Минимальным кодовым расстоянием d называется минимальное число искаженных символов, необходимое для перехода одной разрешенной комбинации в другую.
4 Число обнаруживаемых и исправляемых ошибок. Если q - количество ошибок, которое код способен исправить, то необходимо и достаточно, чтобы $d \geq 2q+1$.

Для построения самокорректирующегося кода, рассчитанного на исправление одиночных ошибок, одного контрольного разряда недостаточно. Количество контрольных разрядов k должно быть выбрано так, чтобы удовлетворялось неравенство - $2^k \geq k+m+1$, или

$k \geq \log_2(k+m+1)$, где m — количество основных двоичных разрядов кодового слова. Минимальные значения k при заданных значениях m, найденные в соответствии с этим неравенством, приведены в таблице.

Таблица 7. Зависимость количества корректирующих разрядов от разрядности основного двоичного кода

Диапазон m	**k_{min}**
1	2
2-4	3
5-11	4
12-26	5
27-57	6

Код Хемминга

Построение кодов Хемминга основано на принципе проверки на четность числа единичных символов: к последовательности добавляется такой элемент, чтобы число единичных символов в получившейся последовательности было чётным.

$r_1=i_1 \oplus i_2 \oplus \ldots \oplus i_k$.

При этом можно вычислить $S= i_1 \oplus i_2 \oplus \ldots \oplus i_k \oplus r_1$. Если S=0, то ошибки нет, если S=1 – имеет место однократная ошибка. Для каждого числа проверочных символов r=3,4,5... существует классический код Хемминга с маркировкой $(n,k) =2^r-1,2^r-1-r$. Для приведенных выше *r* – (7,4), (15,11), (31,26), соответственно.

При иных значениях *k* получается так называемый усеченный код, например, международный телеграфный код МТК-2, у которого *k* =5. Для него необходим код Хемминга (9,5), который является усеченным от классического (15,11).

Получение кодового слова осуществляется следующим образом:

$$\begin{pmatrix} i_1 & i_2 & i_3 & i_4 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} = \begin{pmatrix} i_1 & i_2 & i_3 & i_4 & r_1 & r_2 & r_3 \end{pmatrix}$$

Или последовательно:

$r_1=i_1 \oplus i_2 \oplus i_3$,

$r_2=i_2 \oplus i_3 \oplus i_4$,

$r_3=i_1 \oplus i_2 \oplus i_4$.

При получении кодового слова вычисляется синдром последовательности $S=(S_1, S_2, S_3)$.

При этом:

$S_1=r_1 \oplus i_1 \oplus i_2 \oplus i_3$,

$S_2=r_2 \oplus i_2 \oplus i_3 \oplus i_4$,

$S_3=r_3 \oplus i_1 \oplus i_2 \oplus i_4$.

Получение синдрома осуществляется следующим образом:

$$\begin{pmatrix} i_1 & i_2 & i_3 & i_4 & r_1 & r_2 & r_3 \end{pmatrix} \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} S_1 & S_2 & S_3 \end{pmatrix}$$

Синдром (0,0,0) соответствует отсутствию ошибок. Каждом ненулевому синдрому соответствует определённая конфигурация ошибок. Таблица 8 устанавливает соответствие между синдромами и ошибками в символе кода.

Таблица 8. Значения синдрома и ошибки

Синдром	001	010	011	100	101	110	111
Конфигурация ошибок	0000001	0000010	0001000	0000100	1000000	0010000	0100000
Ошибка в символе	r_3	r_2	i_4	r_1	i_1	i_3	i_2

Функциональная схема получения кодового слова представлена на рисунке 8.1.

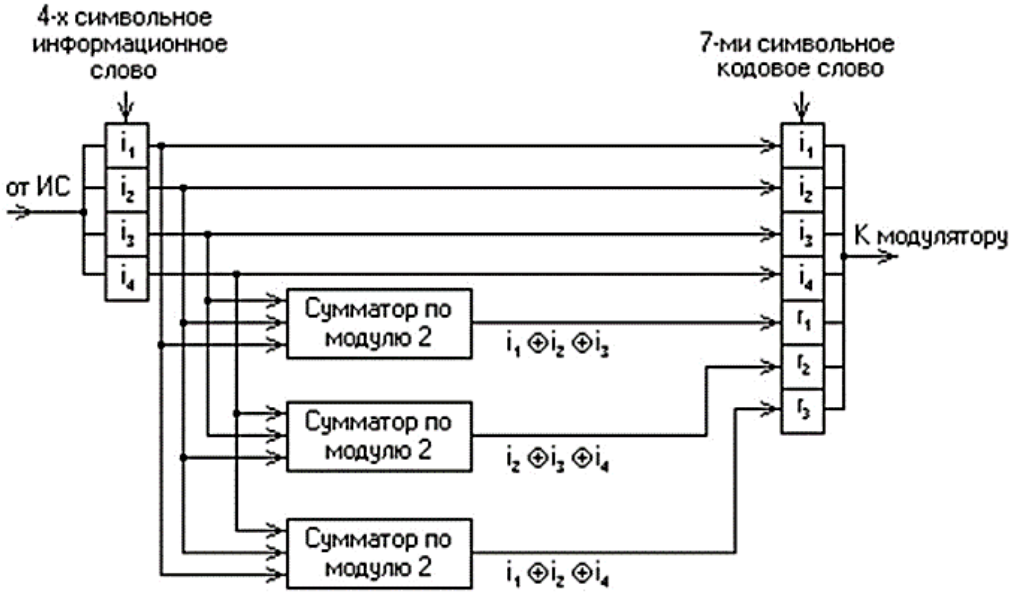


Рисунок 8.1 – Функциональная схема кодера

На рисунке 8.2 представлена функциональная схема декодера.

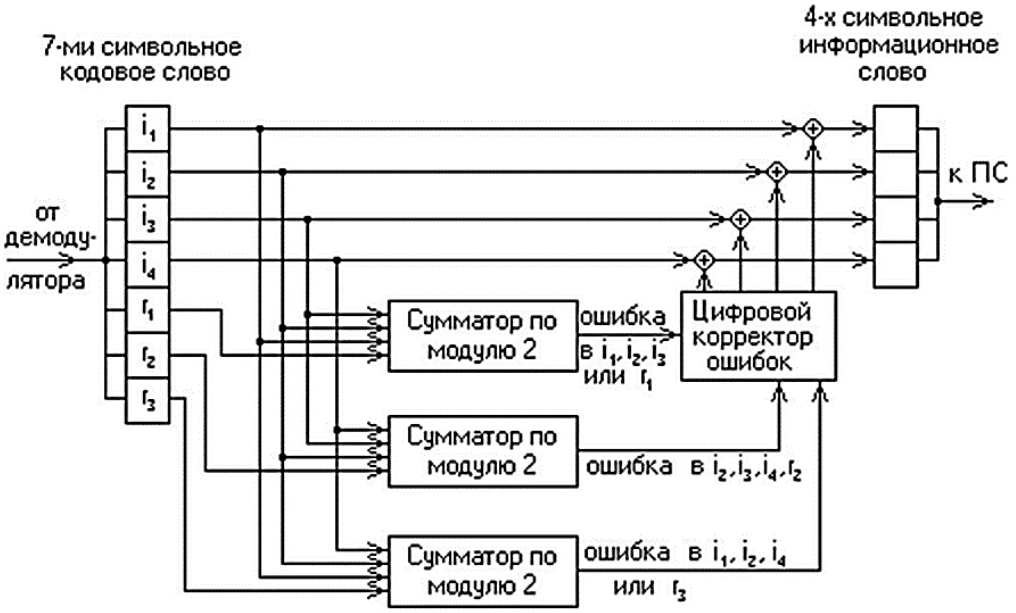


Рисунок 8.2 – Декодер

Алгоритм кодирования

Предположим, что нужно сгенерировать код Хемминга для некоторого информационного кодового слова. В качестве примера возьмём 15-битовое кодовое слово $\mathbf{x}_1 ... \mathbf{x}_{15}$, хотя алгоритм пригоден для кодовых слов любой длины. В приведённой ниже таблице в первой строке даны номера позиций в кодовом слове, во второй — условное обозначение битов, в третьей — значения битов.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
1	0	0	1	0	0	1	0	1	1	1	0	0	0	1

Вставим в информационное слово контрольные биты $\mathbf{r}_0 ... \mathbf{r}_4$ таким образом, чтобы номера их позиций представляли собой целые степени двойки: 1, 2, 4, 8, 16... Получим 20-разрядное слово с 15 информационными и 5 контрольными битами. Первоначально контрольные биты устанавливаем равными нулю. На рисунке контрольные биты выделены розовым цветом.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
r_0	r_1	x_1	r_2	x_2	x_3	x_4	r_3	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	r_4	x_{12}	x_{13}	x_{14}	x_{15}
0	0	1	0	0	0	1	0	0	0	1	0	1	1	1	0	0	0	0	1

В общем случае количество контрольных бит в кодовом слове равно двоичному логарифму числа бит кодового слова (включая контрольные биты), округлённому в большую сторону до ближайшего целого. Например, информационное слово длиной 1 или 2 бита требует двух контрольных разрядов, 3- или 4-битовое информационное слово — трёх, 5...11-битовое — четырёх, 12...26-битовое — пяти и т.д.

Добавим к таблице 5 строк (по количеству контрольных битов), в которые поместим матрицу преобразования. Каждая строка будет соответствовать одному контрольному биту (нулевой контрольный бит — верхняя строка, четвёртый — нижняя), каждый столбец — одному биту кодируемого слова.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
r_0	r_1	x_1	r_2	x_2	x_3	x_4	r_3	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	r_4	x_{12}	x_{13}	x_{14}	x_{15}	
0	0	1	0	0	0	1	0	0	0	1	0	1	1	1	0	0	0	0	1	
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	r_0
0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	r_1
0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	r_2
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	r_3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	r_4

В каждом столбце матрицы преобразования поместим двоичный номер этого столбца, причём— младший бит расположим

в верхней строке, старший — в нижней. Например, в третьем столбце матрицы будут стоять числа 11000, что соответствует двоичной записи числа три: 00011.

В правой части таблицы мы оставили пустым один столбец, в который поместим результаты вычислений контрольных битов. Вычисление контрольных битов производим следующим образом. Берём одну из строк матрицы преобразования (например, r_0) и находим её скалярное произведение с кодовым словом, то есть перемножаем соответствующие биты обеих строк и находим сумму произведений. Если произведение чётное – контрольный бит равен нулю, иначе – 1.

Если описывать этот процесс в терминах матричной алгебры, то операция представляет собой перемножение матрицы преобразования на матрицу-столбец кодового слова, в результате чего получается матрица-столбец контрольных разрядов, которые нужно взять по модулю 2.

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**		
r_0	r_1	x_1	r_2	x_2	x_3	x_4	r_3	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	r_4	x_{12}	x_{13}	x_{14}	x_{15}		
1	1	1	1	0	0	1	0	0	0	1	0	1	1	1	1	0	0	0	1		
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	r_0	1
0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	r_1	1
0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	r_2	1
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	r_3	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	r_4	1

Например, для строки r_0:

$r_0=(1\cdot0+0\cdot0+1\cdot1+0\cdot0+1\cdot0+0\cdot0+1\cdot1+0\cdot0+1\cdot0+0\cdot0+1\cdot1+0\cdot0+1\cdot1+0\cdot1+1\cdot1+0\cdot0+1\cdot0+0\cdot0+1\cdot0+0\cdot1) \bmod 2 = 5 \bmod 2 = 1.$

Полученные контрольные биты вставляем в кодовое слово вместо стоявших там ранее нулей. Кодирование по Хэммингу завершено. Полученное кодовое слово — 11110010001011110001.

Алгоритм декодирования

Алгоритм декодирования по Хэммингу абсолютно идентичен алгоритму кодирования. Матрица преобразования соответствующей размерности умножается на матрицу-столбец кодового слова и каждый элемент полученной матрицы-столбца берётся по модулю 2. Полученная матрица-столбец получила название «матрица синдромов». Легко проверить, что кодовое слово, сформированное в соответствии с алгоритмом, описанным в предыдущем разделе, всегда даёт нулевую матрицу синдромов. Матрица синдромов становится ненулевой, если в результате ошибки (например, при передаче слова по линии связи с шумами) один из битов исходного слова изменил своё значение. Предположим для примера, что в кодовом слове, полученном в предыдущем разделе, шестой бит изменил своё значение с нуля на единицу (на рисунке обозначено красным цветом). Тогда получим следующую матрицу синдромов.

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**		
r_0	**r_1**	**x_1**	**r_2**	**x_2**	**x_3**	**x_4**	**r_3**	**x_5**	**x_6**	**x_7**	**x_8**	**x_9**	**x_{10}**	**x_{11}**	**r_4**	**x_{12}**	**x_{13}**	**x_{14}**	**x_{15}**		
1	1	1	1	0	1	1	0	0	0	1	0	1	1	1	1	0	0	0	1		
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	s_0	0
0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	s_1	1
0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	s_2	1
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	s_3	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	s_4	0

Заметим, что при однократной ошибке матрица синдромов всегда представляет собой двоичную запись (младший разряд в верхней строке) номера позиции, в которой произошла ошибка. В приведённом примере матрица синдромов (01100) соответствует двоичному числу 00110 или десятичному 6, откуда следует, что ошибка произошла в шестом бите.

Задания

1 Определить коды Хемминга для следующих слов: 1100, 1001, 0101.
2 Декодировать коды Хемминга обнаружить и устранить однократную ошибку (при наличии): 0111101, 0011001, 0110101.

Контрольные вопросы

1 Что такое контроль по чётности и нечётности?
3 Как определить необходимое количество контрольных разрядов кода Хемминга, исправляющего однократные ошибки?
4 Сколько существует разрешённых кодовых слов для кода (7, 4)?

Литература

1 Цифровая схемотехника [Текст] : учебное пособие / Е. П. Угрюмов. - 2-е изд., перераб. и доп. - СПб. : БХВ-Петербург, 2005. - 800 с.
2 Батоврин В.К. LabView: Практикум по цифровым элементам вычислительной и информационно-измерительной техники [Текст] : лабораторный практикум / В.К. Батоврин, А.С. Бессонов, А.В. Мошкин. - М.: Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Московский государственный технический университет радиотехники и автоматики», 2011. - 118 с.