МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ЛЕСОТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ИМЕНИ Г.Ф. МОРОЗОВА»

Ю.С. Сербулов
Е.А. Аникеев

ЦИФРОВЫЕ АВТОМАТЫ

Учебное пособие

Воронеж 2017

УДК 519.713

Печатается по решению учебно-методического совета
ФГБОУ ВО «ВГЛТУ» (протокол № ... от ..... г.)

Рецензенты: кафедра информационных технологий
и автоматизированного проектирования в строительстве
ФГБОУ ВО «ВГТУ»;
профессор кафедры информатики и графики
ФГБОУ ВО «ВГТУ» к-т физ-матем. наук,
доц. А.Д. Кононов

Сербулов, Ю. С.
Б44? Цифровые автоматы [Текст] : учебное пособие /
Ю. С. Сербулов, Е.А. Аникеев ; М-во образования и науки РФ, ФГБОУ ВО «ВГЛТУ». –
Воронеж, 2017. – 124 с.
ISBN .....

В учебном пособии рассматриваются основы цифровых автоматов, способы создания комбинационных и последовательностных схем.

Учебное пособие предназначено для студентов 2 курса механического факультета по направлению подготовки 09.03.02 – Информационные системы и технологии.

УДК 519.713

ISBN .....

Оглавление

Введение 4
Глава 1 Основы цифровой техники 5
1.1 Системы счисления 5
1.2 Перевод чисел из одной системы счисления в другую 8
1.3 Законы алгебры логики 13
1.4 Переключательные функции 16
1.5 Минимизация логических функций 21
1.6 Реализация логических функций в различных базисах 25
Контрольные вопросы к главе 1 31
Глава 2 Комбинационные цифровые схемы 33
2.1 Формы представления цифровых автоматов без памяти 33
2.2 Дешифраторы 37
2.3 Шифраторы 42
2.4 Мультиплексоры 44
2.5 Компараторы 47
2.6 Сумматоры 53
Контрольные вопросы к главе 2 62
Глава 3 Цифровые схемы последовательностного типа 64
3.1 Формы представления цифровых автоматов с памятью 64
3.2 Элементарные конечные автоматы 70
3.3 Синтез функций активации и выходов цифрового автомата с использованием RS-триггеров 87
3.4 Синтез конечного автомата с использованием Т-триггера 100
3.5 Счётчики 107
3.6 Регистры 112
Контрольные вопросы к главе 3 119
Список литературы 123

Введение

Учебное пособие «Цифровые автоматы» предназначено для студентов для студентов 2 курса механического факультета по направлению подготовки 09.03.02 – Информационные системы и технологии и используется при изучении дисциплины «Цифровые автоматы».

В пособии изложены основы прикладной теории автоматов применительно созданию узлов вычислительной техники: общие сведения об системах счисления и алгебре логики, способы описания и синтеза комбинационных и последовательностных автоматов.

Глава 1 Основы цифровой техники

1.1 Системы счисления

В современных вычислительных системах вся информация представлена в виде кодов чисел с использованием двоичной системы счисления. Изучение систем счисления необходимо для понимания основ функционирования вычислительных машин.

Система счисления – это способ изображения и именования чисел с помощью символов, имеющих количественное значение.

Существуют позиционные и непозиционные системы счисления.

В непозиционных системах значение каждого символа в числе не зависит от его местоположения. Например, в римской системе XXX – тридцать. Эти системы не получили широкого применения для расчётов, так как способы чтения этих чисел и арифметических операций над ними довольно сложны.

В позиционных системах счисления значение каждого символа в числе зависит от его местоположения в этом числе. Например: 111 – сто одиннадцать. Это число состоит из трёх единиц, количественное значение которого зависит от занимаемого места. Первая единица имеет количественное значение «сто», вторая – «десять», третья – «один».

В позиционной системе счисления любое число можно представить в виде:

$$A = \pm a_{m-1}\ a_{m-2}\ \mathbf{K}\ a_1\ a_0\,, a_1\ a_2\ \mathbf{K}\, a_k\,.$$

Позиции, пронумерованные $m-1, m, \mathbf{K}\, 0$ и $1, 2, \mathbf{K}\, k$ называются разрядами числа. Дробные разряды отделяются от целых запятой. Общее число разрядов составляет $m+k$.

Каждая цифра разрядов числа может принимать одно из значений от 0 до $N-1$. При этом число N называется основанием системы счисления. Оно численно равно количеству цифр или других символов, используемых в изображении чисел в данной системе счисления. Кроме того, основание системы счисления показывает, во сколько раз единица некоторого разряда числа больше единицы предыдущего разряда числа.

Учитывая это, любое число в позиционной системе счисления можно представить в виде:

$$A = \pm\left[a_{m-1}N^{m-1} + a_{m-2}N^{m-2} + \mathbf{K} + a_1N^1 + a_0N^0 + a_{-1}N^{-1} + \mathbf{K} + a_{-k}N^{-k}\right].$$

Например:

$1995{,}21 = 1 \cdot 10^3 + 9 \cdot 10^2 + 9 \cdot 10^1 + 5 \cdot 10^0 + 2 \cdot 10^{-1} + 1 \cdot 10^{-2}$.

Такая форма записи числа называется канонической. Основание системы счисления здесь 10. Название системы счисления определяется её основанием. Например, система, включающая в себя десять цифр $0,1,2,3,4,5,6,7,8,9$ – десятичная.

В аппаратной части современных вычислительных машин все операции выполняются в двоичной системе счисления, включающей в себя две цифры: 0 и 1. Использование такой системы имеет два преимущества. Для кодирования чисел можно использовать достаточно простые и надёжные элементы, имеющие два устойчивых состояния, одно из которых кодируется нулём, другое – единицей. Выполнение арифметических операций в двоичной системе является наиболее простым.

Кроме двоичной системы, в программировании вычислительных машин используются восьмеричная и шестнадцатеричная системы счисления, содержащие соответственно 8 и 16 символов. Так как используемая обычно десятичная система содержит только десять символов, в шестнадцатеричной системе в качестве цифр используются буквы A, B, C, D, E, F (таблица 1.1).

Арифметические операции с числами в любой позиционной системе счисления производятся поразрядно при помощи определения соответствующих элементарных операций.

Элементарные операции сложения в двоичной системе:

$0 + 0 = 0$

$1 + 0 = 0 + 1 = 1$

$1 + 1 = 10$

Таблица 1.1 Таблица соответствий чисел систем счисления

Десятичная	Двоичная	Восьмеричная	Шестнадцатеричная
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10

Пример сложения чисел в двоичной системе:

$$\begin{array}{rr} & 11011,101 \\ + & 1110,001 \\ \hline & 101001,110 \end{array}$$

Элементарные операции умножения в двоичной системе:

$0 \cdot 0 = 0$

$1 \cdot 0 = 0 \cdot 1 = 0$

$1 \cdot 1 = 1$

Умножение производится, начиная с младших или старших разрядов множителей, так как при умножении любых цифр в двоичной системе счисления не возникает единиц переноса в соседний старший разряд.

Пример умножения чисел в двоичной системе:

×							1	1	1	0	1,	1	0	1
									1	0	0,	1	1	1
							1	1	1	0	1	1	0	1
						1	1	1	0	1	1	0	1	
					1	1	1	0	1	1	0	1		
				0	0	0	0	0	0	0	0			
			0	0	0	0	0	0	0	0				
		1	1	1	0	1	1	0	1					
	1	0	0	1	0	0	0	0,	0	1	1	0	1	1

1.2 Перевод чисел из одной системы счисления в другую

Одним из распространённых методов перевода чисел является универсальный алгоритм. Кроме него, применяют перевод при помощи весов разрядов, схему Горнера и другие алгоритмы.

При использовании универсального алгоритма отдельно переводятся целая и дробная части. Для перевода целой части необходимо последовательно делить целую часть числа и образующиеся при делении целые частные, записанные в исходной системе счисления, на основание новой системы счисления, записанное в исходной системе счисления. Образующиеся на каждом шагу деления остатки представляют собой цифры разрядов числа в новой системе счисления, записанные цифрами исходной системы счисления. При этом первый остаток – это цифра младшего разряда числа, а последний остаток – цифра старшего разряда числа

Для перевода дробной части числа необходимо последовательно умножать дробную часть числа и дробные части образующихся произведений, записанные в исходной системе счисления, на основание новой системы счисления, записанное в исходной системе счисления. При этом целые части образующихся на каждом шагу умножения произведений представляют собой цифры разрядов дробной части числа в новой системе счисления, записанные в исходной системе счисления. При этом целая часть нового произведения – это старший разряд дробной части числа, стоящего справа от запятой.

Рассмотрим пример.

$A_{10} = 30{,}6$. $A_2 = ?$

30	2				
30	15	2			
0	14	7	2		
	1	6	3	2	
		1	2	1	2
			1	0	0
				1	

1	1	1	1	0	

$\times$

0,	6
	2
1,	2
	2
0,	4
	2
0,	8
	2
1,	6

1	0	0	1		

В результате $A_2 = 11110{,}1001$.

Перевод чисел с использованием весов разрядов удобно использовать для перевода из двоичной системы счисления в десятичную. Вес разряда – это число, показывающее, во сколько раз единица данного разряда больше или меньше единицы младшего разряда целой части числа. Если в формуле канонической записи числа все цифры записать в новой системе счисления и выполнить указанные в формуле арифметические действия, то получим число в новой системе счисления.

Пример:

1	0	0	0	1	0	1	- число
64	32	16	8	4	2	1	- веса разрядов

Просуммировав ненулевые разряды, получим: 64+4+1 = 69

Перевод чисел по схеме Горнера представляет собой вычисления по преобразованной формуле канонической формы числа, например:

$$A_N = a_3 N^3 + a_2 N^2 + a_1 N^1 + a_0 N^0 = \left(\left(a_3 N + a_2\right) N + a_1\right) N + a_0 .$$

Для перевода числа из одной системы счисления в другую нужно умножить цифру старшего разряда числа, записанную в новой системе счисления, на основание исходной системы счисления, записанное в новой системе счисления. Затем к полученному результату прибавить цифру следующего разряда числа, записанного в новой системе счисления, и полученную сумму снова умножить на основание исходной системы счисления, и так далее до последнего, младшего разряда. После прибавления младшего разряда умножение на основание системы счисления не производится.

Примеры:

$A_8 = 312$. $A_{10} = (3 \cdot 8 + 1) \cdot 8 + 2 = 202$.

$A_{16} = 2EF$. $A_{10} = (2 \cdot 16 + 14) \cdot 16 + 15 = 751$.

Рассмотрим теперь некоторые особенности перевода из восьмеричной и шестнадцатеричной системы в двоичную.

Восьмеричная система счисления используется при указании порядковых номеров команд в кодах операций и адресов команд во многих языках низкого уровня. Основание этой системы равно 2^3, что обуславливает простой перевод из A_8 в A_2 и наоборот. Для такого перевода надо каждую цифру восьмеричного числа заменить соответствующим ей трехразрядным двоичным числом или триадой.

A_8 = 1035

A_2 = 001 000 011 101

Для перевода двоичного числа в восьмеричное нужно разбить его влево и вправо от запятой на группы по три двоичных разряда. Если в крайних группах число разрядов меньше трех, то нужно дополнить их до трёх нулями, а затем каждую триаду заменить соответствующим восьмеричным числом.

A_2 = 001 101 , 111 010

A_8 = 15,72

При байтовой организации ЭВМ каждый байт удобно представлять двухразрядным шестнадцатеричным числом.

Для перевода чисел из шестнадцатеричной системы счисления в двоичную нужно каждую цифру шестнадцатеричного числа заменить соответствующим ей четырехразрядным двоичным числом, или тетрадой.

A_{16} = B1,5

A_2 = 1011 0001 , 0101

Для перевода чисел из двоичной системы счисления в шестнадцатеричную нужно разбить его влево и вправо от запятой на группы по четыре двоичных разряда. Если в крайних группах число разрядов меньше четырёх, то нужно дополнить их до четырёх нулями, а затем каждую тетраду заменить соответствующим шестнадцатеричным числом.

A_2 = 0010 1010 , 1111 1000

A_{16} = 2A,F8

Рассмотренные ранее методы имеют особенности, которые в ряде случаев не позволяют использовать их для непосредственного перевода чисел из одной системы в другую. Универсальный алгоритм предполагает, что все арифметические операции выполняются в исходной системе счисления. Например, в случае перевода $A_3 \to A_{10}$ это может быть трудным. Схема Горнера и метод весов разрядов предполагают знание правил арифметических действий в новой системе счисления. В случае $A_{10} \to A_7$ это также может быть непростым. Кроме того, существуют такие случаи, как $A_5 \to A_3$. Здесь удобно воспользоваться промежуточной системой счисления, выбрав её так, что мы знаем правила операций в ней. Перевод проводится в два этапа:

1. Число переводится в десятичную систему счисления по схеме Горнера.
2. Число переводится в новую систему счисления с использованием универсального алгоритма.

Пример.

A_5 = 103 A_3 = ?

$A_{10} = (1 \cdot 5 + 0) \cdot 5 + 3 = 28$.

Вычислим теперь A_3.

28/3 = 9, остаток 1.
9/3 = 3, остаток 0.
3/3 = 1, остаток 0.
1/3 = 0, остаток 1.

Запишем остатки в обратном порядке, снизу вверх.

A_3 = 1001

Этим методом можно пользоваться при переводе чисел $A_{10} \to A_2$. В качестве промежуточной системы счисления используется восьмеричная.

A_{10} = 17 A_2 = ?

17/8 = 2, остаток 1.
2/8 = 0, остаток 2.
Итак, A_8 = 21.
Теперь представим A_8 в виде двоичных триад:

A_2 = 010 001 = 10001.

Наиболее часто употребляемыми в настоящее время позиционными системами являются:

2 – двоичная (применяется в дискретной математике, информатике, программировании);

3 – троичная (при разработке устройств с троичную логикой);

8 – восьмеричная (в информатике, программировании);

10 – десятичная (используется повсеместно);

12 – двенадцатеричная (счёт дюжинами);

16 – шестнадцатеричная (используется в программировании, информатике);

20 – двадцатеричная (использовалась в древности, следы её сохранились во многих языках мира как основание для образования числительных);

60 – шестидесятеричная (единицы измерения времени, измерение углов и, в частности, координат, долготы и широты).

В позиционных системах чем больше основание системы, тем меньшее количество разрядов (цифр) требуется при записи числа.

1.3 Законы алгебры логики

Все элементы современных вычислительных машин представляют собой цифровые автоматы. Для описания функционирования и синтеза этих устройств используется двоичная логика. Она определена на области, состоящей из двух элементов: 0 и 1. Для них определены логические операции (таблица 1.2)

Таблица 1.2 Операции в алгебре логики

Название	Обозначение
Эквивалентность	$x_1 = x_2$
Дизъюнкция (логическое сложение)	$x_1 \vee x_2$
Конъюнкция (логическое умножение)	$x_1 \wedge x_2$ или $x_1 \cdot x_2$ или $x_1 x_2$
Отрицание (логическая инверсия)	$\neg x_1$ или $\overline{x_1}$

Существует множество алгебр логики, в зависимости от того, каким образом определяются эти отношения. Для используемой в вычислительных машинах алгебры логики постулируются следующие отношения при выполнении операций:

Дизъюнкция:

$0 \vee 0 = 0, \quad 0 \vee 1 = 1 \vee 0 = 1, \quad 1 \vee 1 = 1.$

Конъюнкция:

$0 \cdot 0 = 0, \quad 0 \cdot 1 = 1 \cdot 0 = 0, \quad 1 \cdot 1 = 1.$

Инверсия:

$\overline{0} = 1, \quad \overline{1} = 0.$

На основании этих постулатов можно вывести законы алгебры логики:

Закон единичных элементов:

$x \vee 0 = x, \quad x \cdot 0 = 0, \quad x \vee 1 = 1, \quad x \cdot 1 = x.$

Законы отрицания:

Двойное отрицание:

$\overline{\overline{x}} = x$

Дополнение:

$x \vee \overline{x} = 1, \quad x \cdot \overline{x} = 0.$

Двойственность (закон де Моргана)

$\overline{x_1 \vee x_2} = \overline{x}_1 \cdot \overline{x}_2,$

$\overline{x_1 \cdot x_2} = \overline{x}_1 \vee \overline{x}_2.$

Законы комбинационные:

Тавтология:

$x \cdot x \cdot x \cdot \mathbf{K} = x, \quad x \vee x \vee x \vee \mathbf{K} = x.$

Переместительный:

$x_1 \vee x_2 = x_2 \vee x_1, \quad x_1 \cdot x_2 = x_2 \cdot x_1.$

Сочетательный:

$x_1 \vee (x_2 \vee x_3) = (x_1 \vee x_2) \vee x_3$,

$x_1 \cdot (x_2 \cdot x_3) = (x_1 \cdot x_2) \cdot x_3$.

Распределительный:

$x_1 \cdot (x_2 \vee x_3) = x_1 \cdot x_2 \vee x_1 \cdot x_3$,

$x_1 \vee x_2 \cdot x_3 = (x_1 \vee x_2) \cdot (x_1 \vee x_3)$.

Склеивание:

$x_1 \cdot x_2 \vee \overline{x_1} \cdot x_2 = x_2 \cdot (x_1 \vee \overline{x_1}) = x_2$.

Поглощение:

$x_1 \vee x_1 \cdot x_2 = x_1 \cdot (1 \vee x_2) = x_1$.

При выполнении логических операций необходимо соблюдать их приоритетность. Наивысшим приоритетом обладает инверсия. Затем выполняется конъюнкция (логическое умножение), затем дизъюнкция (логическое сложение). Этот порядок может быть изменён при использовании скобок. При этом внутри скобок действия выполняются в порядке изложенного выше приоритета. Если выражение содержит множество скобок, то выполнение операций начинается с внутренних скобок.

Закон де Моргана часто применяется для преобразования логических функций с целью их упрощения. В обобщённом виде этот закон можно записать так:

$\overline{f}(x_1, x_2, \mathbf{K}\, x_n\, ; \vee ; \cdot) = f(\overline{x_1}, \overline{x_2}, \mathbf{K}\, \overline{x_n}\, ; \cdot ; \vee)$.

Применение операций инвертирования произвольной комбинации двоичных переменных эквивалентно замене в ней исходных значений логических переменных их инверсными значениями при одновременной смене знака дизъюнкции на конъюнкцию, а конъюнкции на дизъюнкцию. Например, преобразуем логической выражение, применив к нему двойное отрицание и обобщённый закон де Моргана:

$(x_1 \vee \overline{x_2}) \cdot (\overline{x_3} \vee \overline{x_4}) = \overline{\overline{(x_1 \vee \overline{x_2}) \cdot (\overline{x_3} \vee \overline{x_4})}} = \overline{\overline{x_1} \vee \overline{\overline{x_2}} \vee \overline{\overline{x_3}} \vee \overline{\overline{x_4}}} = \overline{\overline{x_1} \cdot x_2 \vee x_3 \cdot x_4}$.

Проверим правильность преобразования. Для этого составим таблицу истинности этих выражений, подставляя в них все сочетания входящих в них переменных, и сравним значения функций.

Таблица 1.3 Таблица истинности

№	x_1	x_2	x_3	x_4	$\left(x_1 \vee \overline{x_2}\right)\cdot\left(\overline{x_3} \vee \overline{x_4}\right)$	$\overline{\overline{x_1}\cdot x_2 \vee x_3 \cdot x_4}$
0	0	0	0	0	1	1
1	0	0	0	1	1	1
2	0	0	1	0	1	1
3	0	0	1	1	0	0
4	0	1	0	0	0	0
5	0	1	0	1	0	0
6	0	1	1	0	0	0
7	0	1	1	1	0	0
8	1	0	0	0	1	1
9	1	0	0	1	1	1
10	1	0	1	0	1	1
11	1	0	1	1	0	0
12	1	1	0	0	1	1
13	1	1	0	1	1	1
14	1	1	1	0	1	1
15	1	1	1	1	0	0

Как видно из таблицы, преобразования проведены правильно.

1.4 Переключательные функции

Переключательной (двоичной) функцией называется двоичная переменная, значение которой зависит от значений других двоичных переменных, называемых аргументами этой функции:

$p = p(x_1, x_2, \mathbf{K}\, x_n)$.

Двоичная функция считается заданной, если каждому набору значений аргументов поставлено в соответствие определённое значение этой функции. Переключательные функции называются разными, если их значения

отличаются, по крайней мере, для одного набора значений аргументов. Если имеется n аргументов логической функции, то полное число наборов значений N этой функции составит:

$N = 2^n$.

Поскольку на каждом наборе аргументов логические функции могут принимать два значения, то общее число F функций n аргументов составит:

$F = 2^{2^N}$.

Простейшим способом задания переключательной функции является составление таблицы истинности. Например, нужно задать все функции одной переменной.

Число наборов для функции с двумя аргументами составляет:

$N = 2^n = 2^1 = 2$.

Число всех возможных функций для одной переменной:

$F = 2^N = 2^2 = 4$.

Таблица 1.4 Функции одной переменной

x	f_1	f_2	f_3	f_4
0	0	0	1	1
1	0	1	0	1

Эти функции можно задать в виде логических формул:

$f_1 = 0$ – константа ноль.

$f_4 = 1$ – константа единица.

$f_3 = \bar{x}$ – функция инверсии.

$f_2 = x$ – функция тождественности.

Рассмотрим формулы и таблицы истинности основных отношений в логических функциях.

Дизъюнкция:

$p = x_1 \vee x_2$ (ИЛИ).

Таблица 1.5 Значения дизъюнкции двух переменных

x_1	x_2	p
0	0	0
0	1	1
1	0	1
1	1	1

Конъюнкция:

$p = x_1 \cdot x_2$ (И).

Таблица 1.6 Значения конъюнкции двух переменных

x_1	x_2	p
0	0	0
0	1	0
1	0	0
1	1	1

Исключающее или (сложение по модулю 2):

$p = \overline{x_1} \cdot x_2 \vee x_1 \cdot \overline{x_2} = x_1 \oplus x_2$.

Таблица 1.7 Значения исключающего или двух переменных

x_1	x_2	p
0	0	0
0	1	1
1	0	1
1	1	0

Функция Шеффера (И-НЕ):

$p = \overline{x_1 \cdot x_2}$

Таблица 1.8 Значения функции Шеффера двух переменных

x_1	x_2	p
0	0	1
0	1	1
1	0	1
1	1	0

Функция Пирса (ИЛИ-НЕ):

$p = \overline{x_1 \vee x_2}$

Таблица 1.9 Значения функции Пирса двух переменных

x_1	x_2	p
0	0	1
0	1	0
1	0	0
1	1	0

Эти функции широко применяются при синтезе цифровых автоматов.

Рассмотрим задание произвольной логической функции при помощи таблицы истинности и минимизацию этой функции.

В таблице истинности записываются все возможные наборы аргументов логической функции, а затем для каждого набора значений аргументов проставляются значения функции. Рассмотрим функцию трёх аргументов. Число наборов для неё составит: $N = 2^n = 2^3 = 8$.

Таблица 1.10 Таблица истинности функции трёх аргументов

№	x_1	x_2	x_3	p
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

Можно использовать такой способ записи логической функции, заданной в таблице 1.10:

$p(x_1, x_2, x_3) = 1(1,3,5,6).$

$p(x_1, x_2, x_3) = 0(0,2,4,7).$

При проведении минимизации табличную функцию представляют в одной из двух канонических форм: совершенной нормальной дизъюнктивной форме (СДНФ) и совершенной нормальной конъюнктивной форме (СКНФ). Они образуются при помощи двух вспомогательных функций: минтерма и макстерма.

Минтерм – это функция, принимающая значение 1 только на одном наборе аргументов. Минтерм выражается через конъюнкцию аргументов, причём аргумент берётся без знака отрицания, если он равен 1, и со знаком отрицания, если он равен 0.

Макстерм – это функция, принимающая значение 0 только на одном наборе аргументов. Макстерм выражается через дизъюнкцию аргументов, причём аргумент берётся со знаком отрицания, если он равен 1, и без знака отрицания, если он равен 0.

Например, для наборов №1 и №3 получаются минтермы:

$\overline{x_1} \cdot \overline{x_2} \cdot x_3$ и $\overline{x_1} \cdot x_2 \cdot x_3$.

Для наборов №0 и №2 получаются макстермы:

$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3}$ и $x_1 \cdot \overline{x_2} \cdot x_3$.

Совершенная нормальная дизъюнктивная форма (СДНФ) – это дизъюнкция минтермов, равных 1 на тех же наборах, что и заданная логическая функция.

Совершенная нормальная конъюнктивная форма (СКНФ) – это конъюнкция макстермов, равных 0 на тех же наборах, что и заданная логическая функция.

Также СДНФ называют записью по единицам, а СКНФ – записью по нулям.

Функция p равна единице на наборах 1, 3, 5, 6. Запишем СДНФ:

$p = \overline{x_1} \cdot \overline{x_2} \cdot x_3 \vee \overline{x_1} \cdot x_2 \cdot x_3 \vee x_1 \cdot \overline{x_2} \cdot x_3 \vee x_1 \cdot x_2 \cdot \overline{x_3}$.

Функция p равна нулю на наборах 0, 2, 4, 7. Запишем СКНФ:

$p = (x_1 \vee x_2 \vee x_3) \cdot (x_1 \vee \overline{x_2} \vee x_3) \cdot (\overline{x_1} \vee x_2 \vee x_3) \cdot (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$.

Составляющие этих функций, например $\overline{x_1} \cdot \overline{x_2} \cdot x_3$ или $(x_1 \vee x_2 \vee x_3)$, называют простыми импликантами.

1.5 Минимизация логических функций

Минимизацией логической функции называется представление функции в аналитической форме с минимальным количеством переменных и логических операций.

Одним из широко используемых методов минимизации является применение карт Карно. Этот способ минимизации функций был предложен в 1952 Эдвардом В. Вейчем и усовершенствован в 1953 Морисом Карно. В карту Карно двоичные переменные передаются из таблицы истинности и упорядочиваются с помощью кода Грея, в котором каждое следующее число отличается от предыдущего только одним разрядом.

Основным методом минимизации логических функций, представленных в виде СДНФ или СКНФ, является операция попарного неполного склеивания и элементарного поглощения. Операция попарного склеивания осуществляется между двумя импликантами, содержащими одинаковые переменные, вхождения которых (прямые и инверсные) совпадают для всех переменных, кроме одной. В этом случае все переменные, кроме одной, можно вынести за скобки, а оставшиеся в скобках прямое и инверсное вхождение одной переменной подвергнуть склейке.

Исходной информацией для работы с картой Карно является таблица истинности минимизируемой функции. Фактически, карта Карно – это представление таблицы истинности в двумерном виде. Рассмотрим пример для функции трёх переменных (таблица 1.11).

Таблица 1.11 Таблица истинности функции трёх аргументов

№	x_1	x_2	x_3	p
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	0
4	1	0	0	1
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

Преобразуем таблицу истинности функции трёх переменных в карту Карно (рисунок 1.1).

		$x_1\ x_2$			
		00	01	11	10
x_3	0	1	0	1	1
	1	0	0	0	1

Рисунок 1.1 – Карта Карно для функции трёх переменных

Склейку клеток карты Карно можно осуществлять по единицам, если необходимо получить дизъюнктивную форму (ДНФ), или по нулям (если требуется КНФ). Область, которая подвергается склейке, должна содержать только единицы (нули). Склеивать можно только прямоугольные области с числом единиц (или нулей) 2^n, где n – целое число.

Крайние клетки каждой горизонтали и каждой вертикали считаются смежными и могут объединяться. Все единицы (нули) должны попасть в какую-либо область. Одна ячейка карты может входить сразу в несколько областей (это следует из закона тавтологии).

Проведём склейку ячеек по единицам для получения ДНФ. Объединяем соседние ячейки $x_1\ x_2 = 00,\ x_3 = 0$ и $x_1\ x_2 = 10,\ x_3 = 0$. В этом случае переменные x_2 и x_3 не меняют своего значения, а x_1 принимает значения 0 и 1. При этом $x_2 = 0$ и $x_3 = 0$. Запишем импликант: $\overline{x_2}\ \overline{x_3}$. При склеивании соседних ячеек $x_1\ x_2 = 11,\ x_3 = 0$ и $x_1\ x_2 = 10,\ x_3 = 0$, получаем импликант $x_1\overline{x_3}$.

Аналогично, при объединении $x_1\ x_2 = 10,\ x_3 = 0$ и $x_1\ x_2 = 10,\ x_3 = 1$ получим $x_1\overline{x_2}$. Таким образом, минимальная дизъюнктивная форма функции:

$p = \overline{x_2}\ \overline{x_3} \vee x_1\overline{x_3} \vee x_1\overline{x_2}$.

Минимальная форма логической функции, получаемая при помощи карты Карно, не является единственной, в отличие от СКНФ и СДНФ. Возможно несколько эквивалентных друг другу ДНФ (КНФ), которые соответствуют разным способам покрытия карты Карно прямоугольными областями.

Для получения конъюнктивной формы произведём объединение ячеек по нулям. При этом рассматриваем клетки с нулями, неменяющиеся переменные в пределах одной области объединяем в дизъюнкции (инверсии проставляем над единичными переменными), а дизъюнкции областей объединяем в конъюнкцию.:

$x_1\ x_2 = 01,\ x_3 = 0$ и $x_1\ x_2 = 01,\ x_3 = 1$,

$x_1\ x_2 = 00,\ x_3 = 1$ и $x_1\ x_2 = 01,\ x_3 = 1$,

$x_1\ x_2 = 01,\ x_3 = 1$ и $x_1\ x_2 = 11,\ x_3 = 1$.

Получаем:

$p = \left(x_1 \vee \overline{x_2}\right) \cdot \left(x_1 \vee \overline{x_3}\right) \cdot \left(\overline{x_2} \vee \overline{x_3}\right)$.

Рассмотрим пример минимизации функции для четырёх переменных при помощи карты Карно.

		$x_1\ x_2$			
		00	01	11	10
$x_3\ x_4$	00	0	1	1	0
	01	0	0	0	0
	11	1	0	0	1
	10	0	1	1	0

Рисунок 1.2 – Карта Карно для функции четырёх переменных

Для получения минимальной ДНФ объединим ячейки по единицам.

$x_1\ x_2 = 01,\ x_1\ x_2 = 11,\ x_3\ x_4 = 00,\ x_3\ x_4 = 10.$

$x_1\ x_2 = 00,\ x_1\ x_2 = 10,\ x_3\ x_4 = 11.$

В результате:

$p = x_2\overline{x_4} \vee \overline{x_2}x_3x_4$.

Получим минимальную КНФ, для чего произведём объединение ячеек по нулям.

$x_1\ x_2 = 01,\ x_1\ x_2 = 11,\ x_3\ x_4 = 01,\ x_3\ x_4 = 11.$

$x_1\ x_2 = 00,\ x_1\ x_2 = 10,\ x_3\ x_4 = 00,\ x_3\ x_4 = 01.$

$x_1\ x_2 = 00,\ x_1\ x_2 = 10,\ x_3\ x_4 = 10.$

В результате:

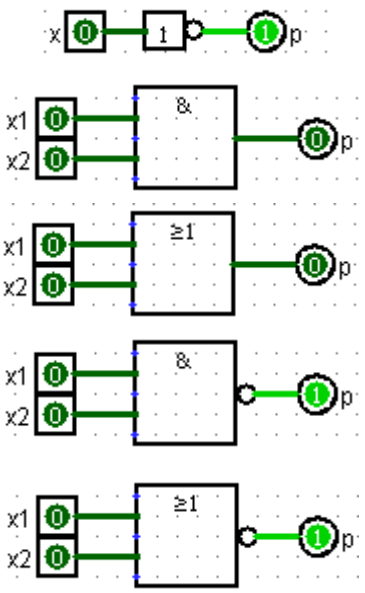
$p = \left(\overline{x_2} \vee \overline{x_4}\right) \cdot (x_2 \vee x_3) \cdot \left(x_2 \vee \overline{x_3} \vee x_4\right).$

Для карт Карно с числом переменных более четырёх могут получаться более сложные области.

1.6 Реализация логических функций в различных базисах

Для изображения логических схем применяются условные изображения логических элементов.

Входы у всех элементов располагаются слева, выходы – справа. В верхней части прямоугольника расположен символ операции, выполняемой элементом. Входы и выходы могут быть прямыми и инверсными. Если вход инверсный, то элемент выполняет свою логическую операцию над инверсным значением переменной, переданной на этот вход. Если выход элемента инверсный, результатом работы элемента является инверсное значение выполняемой элементом логической операции. Существуют различные условные изображения логических элементов. В данном курсе используется программа моделирования цифровых электрических схем Logisim (http://www.cburch.com/logisim/).



$\bar{x}$	Инвертор: НЕ
$x_1 \vee x_2$	Дизъюнкция: ИЛИ
$x_1 x_2$	Конъюнкция: И
$\overline{x_1 x_2}$	Функция Шеффера И-НЕ
$\overline{x_1 \vee x_2}$	Функция Пирса: ИЛИ-НЕ

Рисунок 1.3 – Условные обозначения элементов в Logisim

Все эти элементы имеют статические входы, то есть реагируют на уровень входного сигнала (с 0 на 1). У более сложных логических элементов могут использоваться динамические входы, которые реагируют не на уровень входного сигнала, а на переход входного сигнала с одного уровня на другой. Эти элементы выполняют свою функцию только в те моменты, когда входной сигнал меняет своё значение.

При помощи условных обозначений изображается реализация функций в различных базисах.

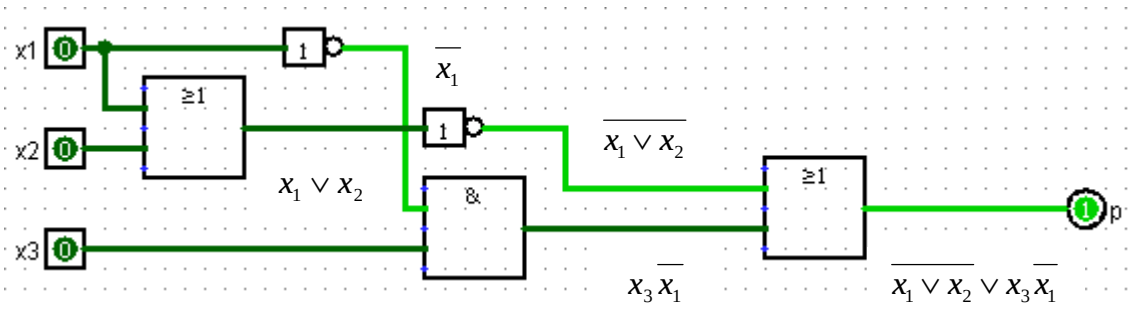
Функционально полная система – это такой набор логических функций, с помощью которых можно выразить любую логическую функцию. Такой набор функций также называют базисом.

Система логических элементов, реализующих функционально полную систему логических функций, называют функционально полной системой логических элементов. Обычно используют три базиса. Основной базис включает в себя операции НЕ, И, ИЛИ. Базис Шеффера (И-НЕ) содержит операцию И-НЕ. Базис Пирса (ИЛИ-НЕ) содержит операцию ИЛИ-НЕ.

Рассмотрим пример реализации логической функции в основном базисе.

$p = \overline{x_1 \vee x_2} \vee x_3 \overline{x_1}$.

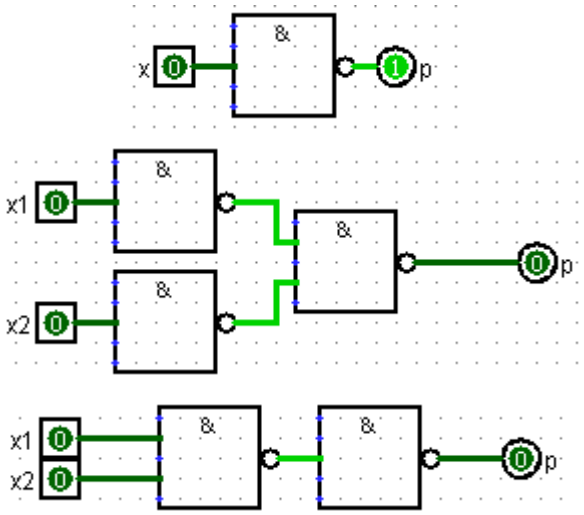
Построим эту схему в основном базисе (рисунок 1.4). Кроме того, воспользуемся функцией Logisim и построим таблицу истинности рассматриваемого выражения.



x1	x2	x3	p
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Рисунок 1.4 – Реализация функции в основном базисе с таблицей истинности

Для реализации функций в базисе И-НЕ соответствующие элементы получают, используя закон де Моргана (рисунок 1.5).



$\bar{x}$ — Инвертор: НЕ

$x_1 \vee x_2 = \overline{\overline{x_1 \vee x_2}} = \overline{\bar{x}_1 \cdot \bar{x}_2}$ — Дизъюнкция: ИЛИ

$x_1 x_2 = \overline{\overline{x_1 x_2}}$ — Конъюнкция: И

Рисунок 1.5 – Логические функции НЕ, ИЛИ, И в базисе И-НЕ

Кроме непосредственной замены элементов, по закону де Моргана можно преобразовать сразу выражение в целом.

Для преобразования выражения $p = \overline{x_1 \vee x_2} \vee x_3 \bar{x}_1$ в базис И-НЕ произведём замены:

$z_1 = \overline{x_1 \vee x_2}$,

$z_2 = x_3 \bar{x}_1$.

Таким образом,

$p = z_1 \vee z_2$.

Проведём преобразования:

$z_1 = \overline{x_1 \vee x_2} = \overline{\bar{x}_1 \cdot \bar{x}_2}$,

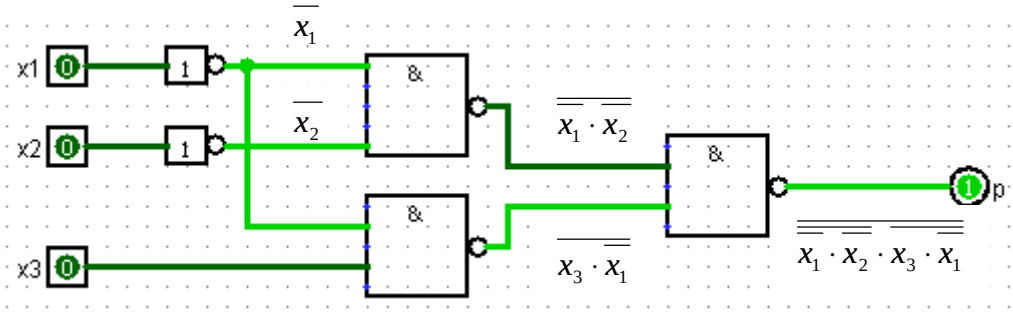
$z_2 = x_3 \bar{x}_1 = \overline{\overline{x_3 x_1}}$,

$p = z_1 \vee z_2 = \overline{\bar{z}_1 \cdot \bar{z}_1}$.

Произведём подстановку преобразованных z_1 и z_2, а также сократим чётное число инверсий:

$$p = z_1 \vee z_2 = \overline{\overline{x_1} \cdot x_2} \cdot \overline{\overline{x_3} \cdot x_1} = \overline{\overline{x_1} \cdot x_2} \cdot \overline{\overline{x_3} \cdot x_1}\ .$$

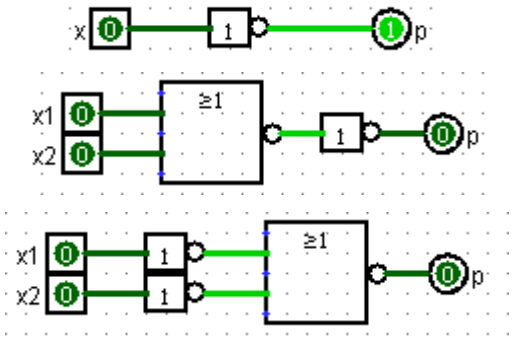
Эта функция содержит только операции И-НЕ. Построим эту схему, а также таблицу истинности, воспользовавшись функцией Logisim.



x1	x2	x3	p
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Рисунок 1.6 – Реализация функции в базисе И-НЕ с таблицей истинности

Рассмотрим ту же функцию в базисе ИЛИ-НЕ. Для реализации функций в этом базисе соответствующие элементы получают, используя закон де Моргана.



$\overline{x}$ — Инвертор: НЕ

$x_1 \vee x_2 = \overline{\overline{x_1 \vee x_2}}$ — Дизъюнкция: ИЛИ

$x_1 x_2 = \overline{\overline{x_1 x_2}} = \overline{\overline{x_1} \vee \overline{x_2}}$ — Конъюнкция: И

Рисунок 1.7 – Логические функции НЕ, ИЛИ, И в базисе ИЛИ-НЕ

Для преобразования выражения $p = \overline{x_1 \vee x_2} \vee x_3 \overline{x_1}$ в базис ИЛИ-НЕ произведём замены:

$z_1 = \overline{x_1 \vee x_2}$,

$z_2 = x_3 \overline{x_1}$.

Таким образом,

$p = z_1 \vee z_2$.

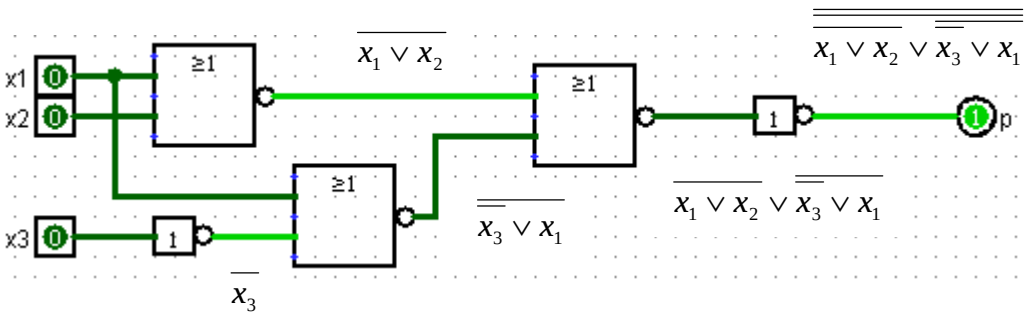
Функция z_1 уже представлена в нужном базисе. Преобразуем z_2:

$z_2 = x_3 \overline{x_1} = \overline{\overline{x_3 x_1}} = \overline{\overline{x_3} \vee \overline{x_1}} = \overline{\overline{x_3} \vee x_1}$.

В результате:

$p = \overline{x_1 \vee x_2} \vee \overline{\overline{x_3} \vee x_1} = \overline{\overline{x_1 \vee x_2} \vee \overline{x_3} \vee x_1}$.

Эта функция содержит только операции ИЛИ-НЕ. Построим эту схему, а также таблицу истинности, воспользовавшись функцией Logisim.



x1	x2	x3	p
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Рисунок 1.8 – Реализация функции в базисе И-НЕ с таблицей истинности

Как видно таблиц истинности на рисунках 1.4, 1.6 и 1.8, все преобразования эквивалентны друг другу.

Если логическая функция задана таблицей истинности, то её можно реализовать в любом базисе. Запись СДНФ или СКНФ является реализацией функции в основном базисе. Для того, чтобы получить ту же функцию в базисе И-НЕ, надо записать СДНФ этой функции, а затем взять от всего выражения двойное отрицание и применить обобщённый закон де Моргана. Если же нужно получить реализацию функции в базисе ИЛИ-НЕ, то надо записать СКНФ, а затем взять двойное отрицание и применить закон де Моргана.

Рассмотрим пример. Пусть логическая функция задана таблицей истинности:

Таблица 1.12 Таблица истинности функции трёх аргументов

№	x_1	x_2	x_3	p
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

Для её реализации в базисе И-НЕ запишем СДНФ:

$$p = \overline{x_1} \cdot \overline{x_2} \cdot x_3 \vee \overline{x_1} \cdot x_2 \cdot x_3 \vee x_1 \cdot \overline{x_2} \cdot x_3 \vee x_1 \cdot x_2 \cdot \overline{x_3}\,.$$

Применим двойное отрицание и закон де Моргана:

$$p = \overline{\overline{x_1} \cdot \overline{x_2} \cdot x_3 \vee \overline{x_1} \cdot x_2 \cdot x_3 \vee x_1 \cdot \overline{x_2} \cdot x_3 \vee x_1 \cdot x_2 \cdot \overline{x_3}} = \overline{\overline{x_1} \cdot \overline{x_2} \cdot x_3} \cdot \overline{\overline{x_1} \cdot x_2 \cdot x_3} \cdot \overline{x_1 \cdot \overline{x_2} \cdot x_3} \cdot \overline{x_1 \cdot x_2 \cdot \overline{x_3}}\,.$$

Для реализации этой функции в базисе ИЛИ-НЕ запишем СКНФ:

$$p = (x_1 \vee x_2 \vee x_3) \cdot (x_1 \vee \overline{x_2} \vee x_3) \cdot (\overline{x_1} \vee x_2 \vee x_3) \cdot (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3}).$$

Применим двойное отрицание и закон де Моргана:

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

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

Построение этих функций и их таблиц истинности показывает эквивалентность проведённых преобразований.

Контрольные вопросы к главе 1

1. Что такое система счисления?
2. Какие существуют системы счисления?
3. Как формируются числа в непозиционной системе?
4. Как можно представить число в позиционной системе?
5. Как выглядит каноническая форма записи числа?
6. Каковы преимущества использования двоичной системы в аппаратной части вычислительной техники?
7. Какие ещё системы счисления используются, какие в них используются цифры?
8. Как выполняется перевод целой части числа по универсальному алгоритму?
9. Что необходимо сделать для перевода дробной части числа по универсальному алгоритму?
10. Как при переводе чисел используется схема Горнера?
11. Как выполняется перевод чисел при помощи весов разрядов?
12. Какие существуют отношения между элементами алгебры логики, как они обозначаются?
13. Напишите формулы логического сложения, умножения и инверсии.
14. В чём заключаются закон единичных элементов, двойное отрицание и дополнение?
15. Сформулируйте закон де Моргана.
16. Напишите формулы для закона тавтологии, а также переместительного и сочетательного законов.
17. Какова формулировка распределительного закона, а также склеивания и поглощения?

18. Что называется переключательной (двоичной) функцией, какая функция считается заданной, чем они отличаются?
19. Как определить число наборов для логической функции и число логических функций?
20. Опишите все возможные функции одной переменной в табличном и символьном виде.
21. Запишите в символьном и табличном виде функции дизъюнкции, конъюнкции и сложения по модулю 2.
22. Запишите в символьном и табличном виде функции Шеффера и Пирса.
23. Составьте таблицу истинности для функции от трёх аргументов.
24. Какие существуют канонические формы записи логической функции, что такое минтерм и макстерм?
25. Что такое СДНФ и СКНФ?
26. Что понимают под минимизацией логической функции?
27. Какая существует связь между таблицей истинности и картой Карно?
28. Как получить дизъюнктивную минимальную форму логической функции?
29. Как получить конъюнктивную минимальную форму логической функции?
30. Каковы общие принципы изображения логических элементов?
31. Как изображаются элементы инверсии, дизъюнкции, конъюнкции, Шеффера и Пирса?
32. Какие типы входов существуют у логических элементов, как они изображаются?
33. Что такое базис, какие базисы существуют, какие операции они содержат?
34. Как осуществляется переход между базисами?

Глава 2 Комбинационные цифровые схемы

2.1 Формы представления цифровых автоматов без памяти

Цифровой автомат (ЦА) – это формальная модель любого информационного устройства дискретного действия. Модель предназначена для представления устройства на логическом уровне, т.е. в ней не рассматриваются физические характеристики процессов, происходящих при работе реального устройства.

Любой дискретный узел, входящий в состав вычислительной техники, представляет собой цифровой автомат, осуществляющий преобразование исходных последовательностей цифровых кодов в результирующие последовательности по определённому алгоритму.

Каждый цифровой автомат характеризуется количеством состояний, в которых он может находиться в процессе работы. Обычно это количество конечно, поэтому цифровые автоматы такого типа называют конечными автоматами. Бесконечное число состояний для цифрового автомата означает его бесконечную сложность, так как иначе нельзя обеспечить отличие одного состояния от другого. Можно представить себе бесконечный ЦА, как автомат с неограниченно наращиваемой памятью состояний, и такой автомат представляет определенный теоретический интерес, но в данном курсе рассматриваются только конечные автоматы. Термины «конечный автомат» и «цифровой автомат» часто используются как синонимы.

Конечные автоматы бывают двух типов: комбинационные (автоматы Мура, автоматы без памяти) и последовательностые (автоматы Мили, автоматы с памятью).

Для комбинационных автоматов каждой комбинации входных сигналов соответствует одна комбинация выходных сигналов. Такие автоматы называют также комбинационными схемами. В них отсутствуют элементы памяти. Типичными автоматами без памяти являются цифровые комбинационные схемы: сумматоры, дешифраторы, схемы сравнения, мультиплексоры и т.п.

Для синтеза автомата Мура можно использовать различные способы: табличный, матричный, графический, а также табличную форму матрицы переходов.

Рассмотрим синтез при помощи таблицы переходов. Так как в комбинационном автомате выходные состояния однозначно соответствуют внутренним, то в таблице имеется строка, отражающая это соответствие. В качестве примера рассмотрим автомат Мура с входными состояниями x_1, x_2, x_3, внутренними q_1, q_2, q_3, q_4 и выходными u_1, u_2. Зададим таблицу переходов:

Таблица 2.1 Таблица переходов автомата Мура

q(t) / x(t)	u_1	u_1	u_2	u_1
	q_1	q_2	q_3	q_4
x_1	q_1	q_3	q_4	q_2
x_2	q_4	q_2	q_1	q_3
x_3	q_2	q_1	q_4	q_2

Матричное представление автомата Мура имеет один дополнительный столбец, с помощью которого задаётся соответствие между внутренними состояниями и выходными сигналами автомата.

Таблица 2.2 Матричное представление автомата Мура

q_{t+1} / q_t	q_1	q_2	q_3	q_4	u_t
q_1	x_1	x_3	–	x_2	u_1
q_2	x_3	x_2	x_1	–	u_1
q_3	x_2	–	–	$x_1 \vee x_3$	u_2
q_4	–	$x_1 \vee x_3$	x_2	–	u_1

Другим способом представления конечных автоматов является задание их при помощи графов. Для обоих типов автоматов их состояние представляется вершинами графа, а переходы из одного состояния в другое изображаются при помощи направленных дуг. Состояния, вызывающие переходы, приписываются соответствующим дугам графа. Для автомата Мура (без памяти) выходной сигнал ставится в соответствие вершине графа.

Если переход автомата из одного состояния в другое может происходить под воздействием двух различных комбинаций входных сигналов, то дуги, соответствующие этим переходам, можно заменить одной дугой, которой соответствует дизъюнкция этих состояний. Построим граф автомата Мура, заданного матричным представлением выше.

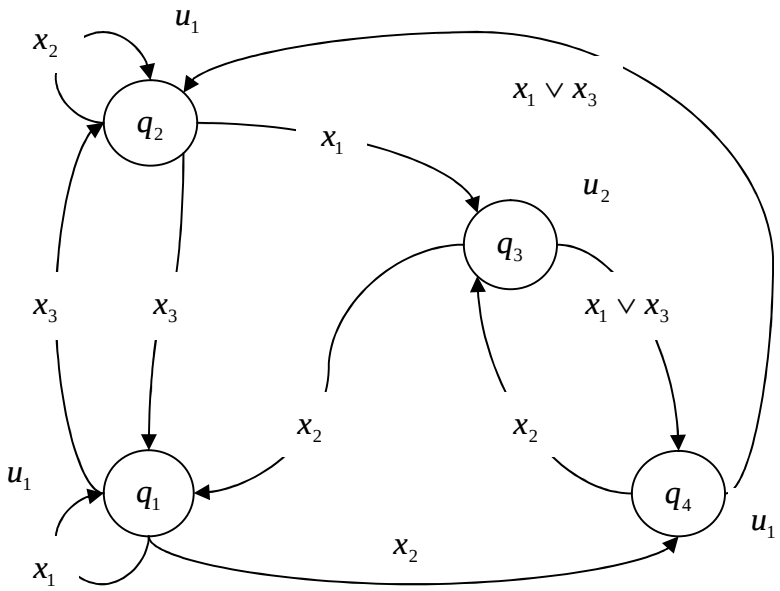


Рисунок 2.1 – Представление автомата Мура при помощи графа

Построим теперь табличную форму матрицы переходов для этого автомата Мура. В этом случае добавляется столбец, отражающий соответствия переходов внутренних состояний автомата смене выходных состояний u_t / u_{t+1}.

Количество строк в табличной форме матрицы переходов определяется числом внутренних состояний q автомата Мура и составляет m^2, где m – число внутренних состояний.

Таблица 2.3 Табличная форма матрицы переходов автомата Мура

q_t	q_{t+1}	Условие перехода	u_t / u_{t+1}
q_1	q_1	x_1	u_1 / u_1
q_1	q_2	x_3	u_1 / u_1
q_1	q_3	–	u_1 / u_2
q_1	q_4	x_2	u_1 / u_1
q_2	q_1	x_3	u_1 / u_1
q_2	q_2	x_2	u_1 / u_1
q_2	q_3	x_1	u_1 / u_2
q_2	q_4	–	u_1 / u_1
q_3	q_1	x_2	u_2 / u_1
q_3	q_2	–	u_2 / u_1
q_3	q_3	–	u_2 / u_2
q_3	q_4	$x_1 \vee x_3$	u_2 / u_1
q_4	q_1	–	u_1 / u_1
q_4	q_2	$x_1 \vee x_3$	u_1 / u_1
q_4	q_3	x_2	u_1 / u_2
q_4	q_4	–	u_1 / u_1

Синтез произвольных комбинационных схем можно осуществлять при помощи таблиц истинности, минимизируя получаемые логические функции при помощи карт Карно. Наиболее известными комбинационными устройствами являются дешифратор, шифратор, мультиплексор, демультиплексор, компаратор, полусумматор и сумматор.

2.2 Дешифраторы

Преобразование данных из одного кода в другой – одна из часто встречающихся операций в технике построения логических схем. Существуют различные виду таких устройств. Например, это шифраторы и дешифраторы. Кроме того, если при помощи такого устройства коммутируется большое количество входных сигналов на небольшое число выходов, такие устройства называют коммутаторами или мультиплексорами. Если информация с небольшого числа входов распределяется на большое число выходов, то устройства называется демультиплексором, или распределителем.

Дешифратор – это комбинационный автомат, обеспечивающий для каждой комбинации входных сигналов появление логической единицы на одном определённом выходе. При этом остальные выходы имеют противоположное состояние. Любой дешифратор имеет n входов и 2^n выходов.

В цифровых устройствах дешифраторы используются для подачи сигналов в различные цепи управления, в зависимости от значения входного кода. Кроме того, их можно использовать в аналого-цифровых и обратных преобразователях для подключения к ним входов и выходов в заданной последовательности.

Дешифратор описывается системой уравнений выходов:

$$p_0 = \bar{x}_{n-1}\bar{x}_{n-2}\,\mathbf{K}\,\bar{x}_1\bar{x}_0,$$

$$p_1 = \bar{x}_{n-1}\bar{x}_{n-2}\,\mathbf{K}\,\bar{x}_1 x_0,$$

$$p_2 = \bar{x}_{n-1}\bar{x}_{n-2}\,\mathbf{K}\,x_1\bar{x}_0,$$

$$p_3 = \bar{x}_{n-1}\bar{x}_{n-2}\,\mathbf{K}\,x_1 x_0,$$

$$\mathbf{KKKK}$$

$$p_{k-1} = x_{n-1}x_{n-2}\,\mathbf{K}\,x_1 x_0,$$

где $x_{n-1}x_{n-2}\,\mathbf{K}\,x_1x_0$ – входные сигналы (переменные) дешифратора,

$p_0, p_1, \mathbf{K}\, p_{k-1}$ – функции выхода дешифратора.

Если в правой части уравнений поставить соответственно нули и единицы и интерпретировать уравнения как двоичные числа, то индекс функции выхода будет равен десятичному эквиваленту соответствующего этой функции двоичного числа.

Из уравнений выхода дешифратора видно, что для его реализации нужно иметь k элементов И на n входов каждый, и для реализации каждой функции выхода использовать отдельный элемент И. Такие схемы называются линейными дешифраторами. Если число адресных входов дешифратора n связано с числом его выходов k соотношением $k = 2^n$, то дешифратор называют полным.

Построим полный дешифратор на три входа. Число выходов: $k = 2^3 = 8$. Входные переменные: x_1, x_2, x_3. Функции выходов:

$p_0 = \bar{x}_2 \bar{x}_1 \bar{x}_0$,

$p_1 = \bar{x}_2 \bar{x}_1 x_0$,

$p_2 = \bar{x}_2 x_1 \bar{x}_0$,

$p_3 = \bar{x}_2 x_1 x_0$,

$p_4 = x_2 \bar{x}_1 \bar{x}_0$,

$p_5 = x_2 \bar{x}_1 x_0$,

$p_6 = x_2 x_1 \bar{x}_0$,

$p_7 = x_2 x_1 x_0$.

Построим эту схему в программе Logisim (рисунок 2.2) и проверим её работу. Действительно, при подаче на вход комбинации $x_1, x_2, x_3 = 101$ на выходе появляется $p_5 = 1$.

При большом количестве входов применяют ступенчатые дешифраторы. Любой дешифратор на n входов и $k = 2^n$ выходов может быть построен в виде двухступенчатого дешифратора, состоящего из двух дешифраторов на n_1 и n_2 входов и, соответственно, на $k_1 = 2^{n_1}$ $k_2 = 2^{n_2}$ выходов. При этом $n = n_1 + n_2$.

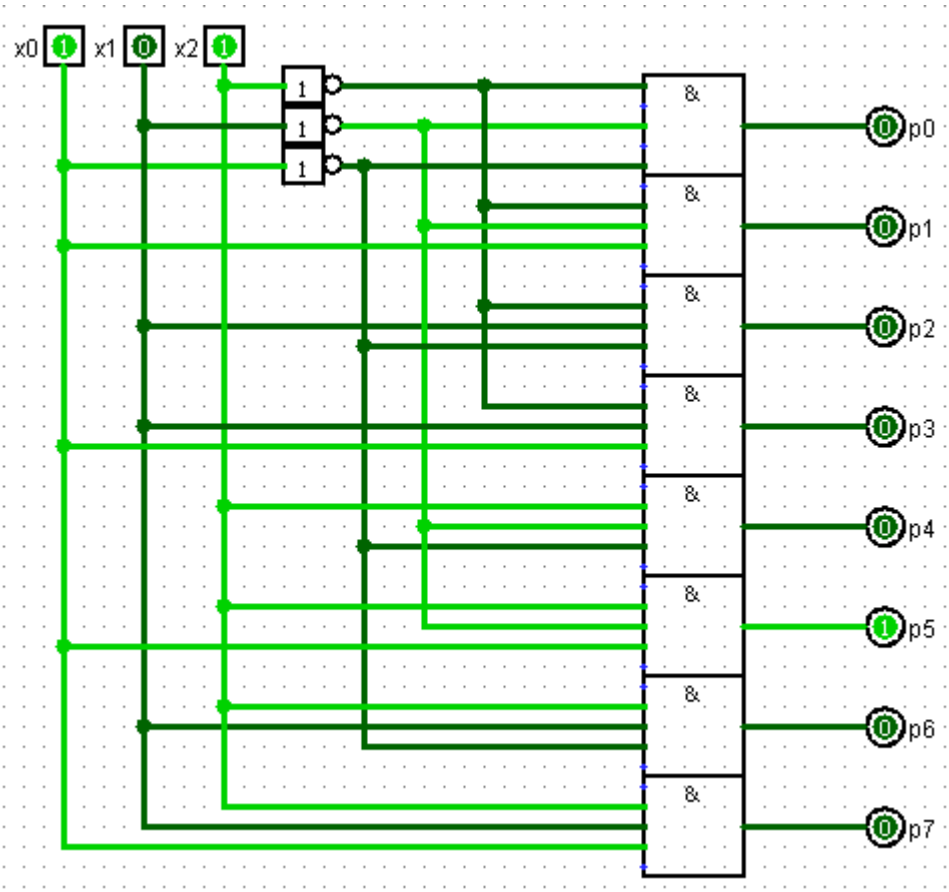


Рисунок 2.2 – Полный дешифратор на три входа

Построим ступенчатый дешифратор на 4 входа. Он содержит $k = 2^n = 16$ выходов. При построении линейного дешифратора понадобится 16 элементов И с четырьмя входами каждый. Применив ступенчатый дешифратор, мы понизим количество входов каждого элемента, но увеличим количество самих элементов.

Входные переменные: x_3, x_2, x_1, x_0.

Функции выхода:

$p_0 = \bar{x}_3\bar{x}_2\bar{x}_1\bar{x}_0$,

$p_1 = \bar{x}_3\bar{x}_2\bar{x}_1 x_0$,

$p_2 = \bar{x}_3\bar{x}_2 x_1\bar{x}_0$,

$p_3 = \bar{x}_3\bar{x}_2 x_1 x_0$,

$p_4 = \bar{x}_3 x_2\bar{x}_1\bar{x}_0$,

$p_5 = \bar{x}_3 x_2\bar{x}_1 x_0$,

$p_6 = \bar{x}_3 x_2\bar{x}_1\bar{x}_0$,

$p_7 = \bar{x}_3 x_2 x_1 x_0$,

$p_8 = x_3 \bar{x}_2 \bar{x}_1 \bar{x}_0$,

$p_9 = x_3 \bar{x}_2 \bar{x}_1 x_0$,

$p_{10} = x_3 \bar{x}_2 x_1 x_0$,

$p_{11} = x_3 \bar{x}_2 x_1 x_0$,

$p_{12} = x_3 x_2 \bar{x}_1 \bar{x}_0$,

$p_{13} = x_3 x_2 \bar{x}_1 x_0$,

$p_{14} = x_3 x_2 x_1 \bar{x}_0$,

$p_{15} = x_3 x_2 x_1 x_0$.

Произведём замены:

$A_0 = \bar{x}_1 \bar{x}_0$,

$A_1 = \bar{x}_1 x_0$,

$A_2 = x_1 \bar{x}_0$,

$A_3 = x_1 x_0$.

Для получения A_i надо иметь дешифратор на два входа. Его входными переменными будут $x_1 x_0$.

$B_0 = \bar{x}_3 \bar{x}_2$,

$B_1 = \bar{x}_3 x_2$,

$B_2 = x_3 \bar{x}_2$,

$B_3 = x_3 x_2$.

Для получения B_i надо иметь дешифратор на два входа, его входные переменные $x_1 x_0$.

После замены получим:

$p_0 = B_0 A_0$,

$p_1 = B_0 A_1$,

$p_2 = B_0 A_2$,

$p_3 = B_0 A_3$,

$p_4 = B_1 A_0$,

$p_5 = B_1 A_1$,

$p_6 = B_1 A_2$,

$p_7 = B_1 A_3$,

$p_8 = B_2 A_0$,

$p_9 = B_2 A_1$,

$p_{10} = B_2 A_2$,

$p_{11} = B_2 A_3$,

$p_{12} = B_3 A_0$,

$p_{13} = B_3 A_1$,

$p_{14} = B_3 A_2$,

$p_{15} = B_3 A_3$.

Построим схему этого дешифратора (рисунок 2.3).

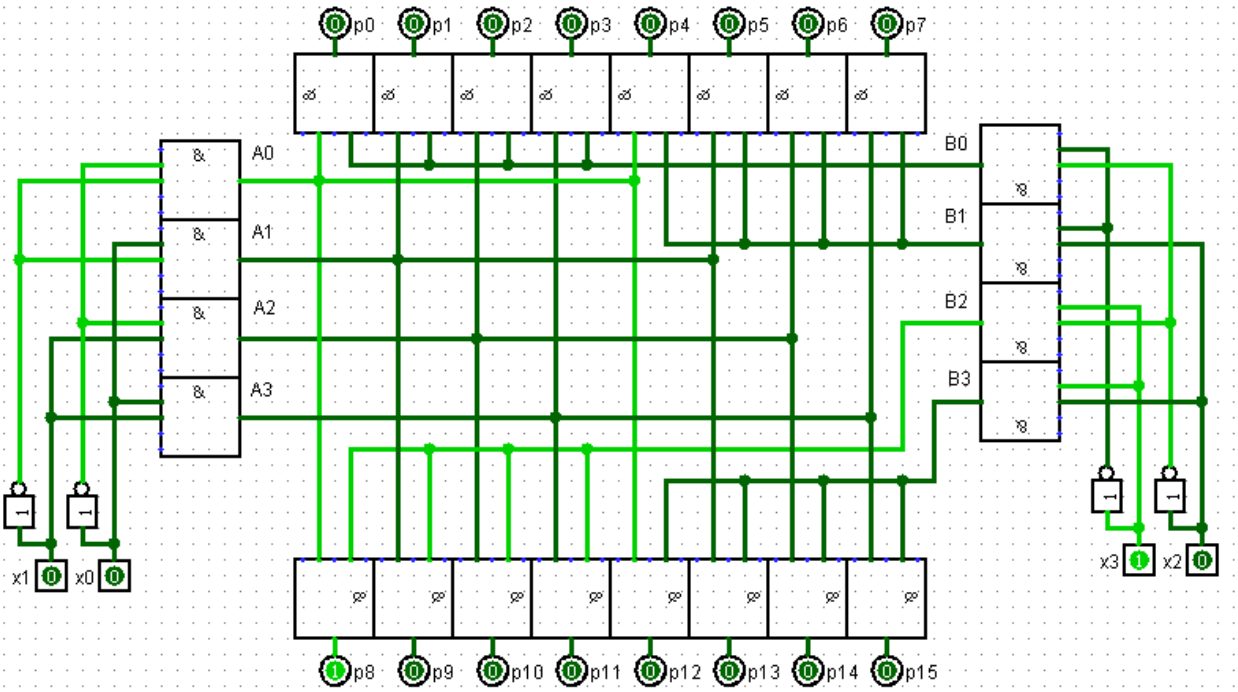


Рисунок 2.3 – Полный двухступенчатый дешифратор на четыре входа

Проверим его работу. Действительно, при подаче на вход $x_3, x_2, x_1, x_0 = 1000$ на выходе появляется $p_8 = 1$.

Дешифраторы применяются обычно в интегральном исполнении, то есть в виде одной микросхемы (рисунок 2.4).

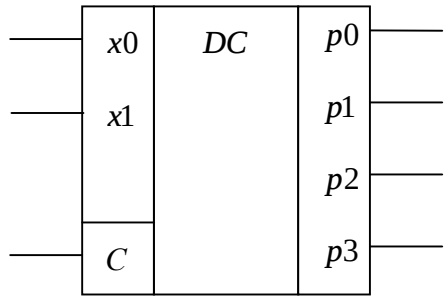


Рисунок 2.4 – Условное обозначение дешифратора

Такие устройства имеют ещё один вход – стробирующий (синхронизации). Он может быть прямым или инверсным. В последнем случае нулевой сигнал на синхронизирующем входе разрешит работу дешифратора, а единичный сигнал запретит её.

2.3 Шифраторы

В разработке цифровых схем часто встречается задача, обратная решаемой дешифраторами. Это преобразование одного кода в другой – например, восьмеричного в двоичный.

Шифратор – комбинационное логическое устройство, выполняющее преобразование позиционного кода в n-разрядный двоичный код.

При подаче сигнала на один из n входов (обязательно на один, не более) на выходе появляется двоичный код номера активного входа.

Если в шифраторе используются все возможные комбинации сигналов на выходе, то такой шифратор называется полным, если не все, то неполным. Число входов и выходов в полном шифраторе связано соотношением:

$n = 2^m$,

где

n – число входов,

m – число выходных двоичных разрядов.

Построим схему шифратора на $n = 8$ входов. Число выходных двоичных разрядов составит: $m = [\log_2 n]_{ббц} == [\log_2 8]_{ббц} = 3$. При этом примем, что сигнал логического нуля на выходах шифратора будет появляться при подаче нулей на все входы одновременно. Таким образом, можно обойтись семью входами, без входа «нуль».

Синтез таких устройств производится по таблице истинности (табл. 2.4).

Таблица 2.4 Таблица истинности шифратора на семь входов

№	Входы							Выходы		
	1	2	3	4	5	6	7	4	2	1
0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	1
2	0	1	0	0	0	0	0	0	1	0
3	0	0	1	0	0	0	0	0	1	1
4	0	0	0	1	0	0	0	1	0	0
5	0	0	0	0	1	0	0	1	0	1
6	0	0	0	0	0	1	0	1	1	0
7	0	0	0	0	0	0	1	1	1	1

Построим СДНФ для функций выходов:

$$f_4 = \bar{x}_1\bar{x}_2\bar{x}_3x_4\bar{x}_5\bar{x}_6\bar{x}_7 \vee \bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4x_5\bar{x}_6\bar{x}_7 \vee \bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4\bar{x}_5x_6\bar{x}_7 \vee \bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4\bar{x}_5\bar{x}_6x_7,$$

$$f_2 = \bar{x}_1x_2\bar{x}_3\bar{x}_4\bar{x}_5\bar{x}_6\bar{x}_7 \vee \bar{x}_1\bar{x}_2x_3\bar{x}_4\bar{x}_5\bar{x}_6\bar{x}_7 \vee \bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4\bar{x}_5x_6\bar{x}_7 \vee \bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4\bar{x}_5\bar{x}_6x_7,$$

$$f_1 = x_1\bar{x}_2\bar{x}_3\bar{x}_4\bar{x}_5\bar{x}_6\bar{x}_7 \vee \bar{x}_1\bar{x}_2x_3\bar{x}_4\bar{x}_5\bar{x}_6\bar{x}_7 \vee \bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4x_5\bar{x}_6\bar{x}_7 \vee \bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4\bar{x}_5\bar{x}_6x_7.$$

В работе шифратора может возникнуть ситуация, в которой на его вход будет подано более одной логической единицы. Для таких случаев используют приоритетные шифраторы. На вход такого устройства может быть подан произвольный двоичный код, т.е. код, который может содержать произвольное число единиц, расположенных в любом порядке. На выходе приоритетного шифратора формируется натуральный двоичный код, определяющий номер позиции приоритетной единицы, т.е. единицы, стоящей в самом старшем разряде.

Такие устройства широко применяются в случаях, когда нескольким периферийным устройствам с различным уровнем приоритета нужно связаться с микропроцессором. Приоритетные шифраторы также могут использоваться в качестве обычных шифраторов.

2.4 Мультиплексоры

Задача передачи на выход сигнала с определённого входа часто встречается при разработке цифровых схем.

Мультиплексор – устройство, имеющее несколько сигнальных входов, один или более управляющих входов и один выход. Мультиплексор позволяет передавать сигнал с одного из входов на выход; при этом выбор желаемого входа осуществляется подачей соответствующей комбинации управляющих сигналов.

Аналоговые и цифровые мультиплексоры значительно различаются по принципу работы. Первые электрически соединяют выбранный вход с выходом (при этом сопротивление между ними невелико – порядка единиц/десятков ом). Вторые же не образуют прямого электрического соединения между выбранным входом и выходом, а лишь «копируют» на выход логический уровень (0 или 1) с выбранного входа. Аналоговые мультиплексоры иногда называют ключами или коммутаторами.

Мультиплексор можно представить в виде коммутатора, обеспечивающего подключение одного из нескольких информационных входов к одному выходу устройства. Коммутатор обслуживает управляющая схема, в которой имеются адресные входы, а также стробирующие.

Сигналы на адресных входах определяют, какой информационный вход подключен к выходу. Число информационных входов полного мультиплексора n связано с числом адресных m соотношением $n = 2^m$. Иначе мультиплексор называют неполным.

Стробирующие входы используются для наращивания разрядности мультиплексора, а также синхронизации его работы с работой других узлов. Сигналы на разрешающих входах могут разрешать или запрещать подключение определенного входа к выходу, то есть могут блокировать действие всего устройства. В качестве управляющей схемы обычно используется дешифратор.

Построим схему мультиплексора на четыре информационных входа x_0, x_1, x_2, x_3. Число управляющих входов $N = [\log_2 4]_{ббц} = 2$. Обозначим их y_0 и y_1. Элементы $D0 - D3$ представляют собой дешифратор с выходами $p_0 - p_3$.

Функция мультиплексора:

$F = C(x_0 p_0 \vee x_1 p_1 \vee x_2 p_2 \vee x_3 p_3)$.

С подстановкой функций дешифратора:

$F = C(x_0 \bar{y}_0 \bar{y}_1 \vee x_1 \bar{y}_0 y_1 \vee x_2 y_0 \bar{y}_1 \vee x_3 y_0 y_1)$.

Реализуем её в программе Logisim.

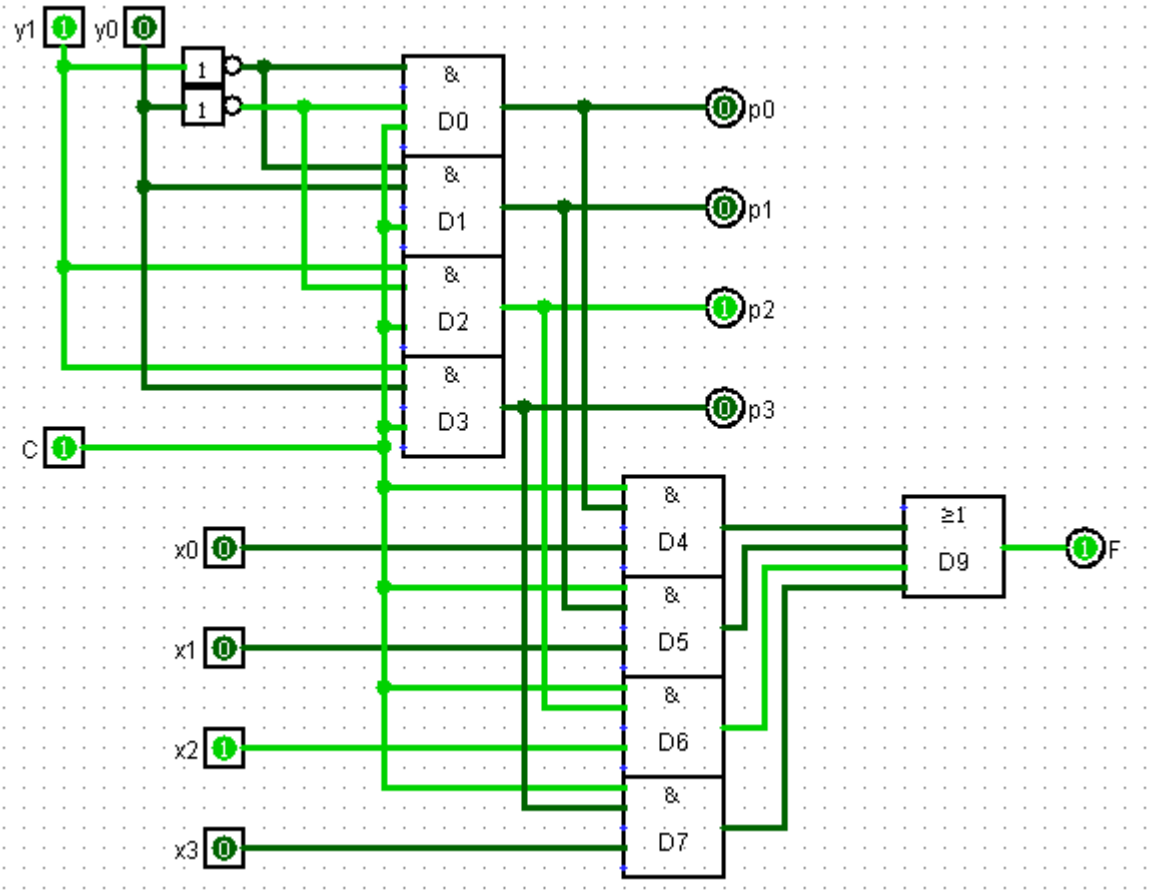


Рисунок 2.5 – Мультиплексор на четыре информационных входа

Проверим работу схемы. Действительно, при подаче на управляющие входы кода $y_1 y_0 = 10$ активируется выход дешифратора $p2$, и при подаче логической единицы на информационный вход x_2 появляется единица на выходе F мультиплексора.

Мультиплексоры могут использоваться в делителях частоты, триггерных устройствах, сдвигающих устройствах, а также для преобразования параллельного двоичного кода в последовательный. В интегральном исполнении мультиплексор обозначается MC или MUX .

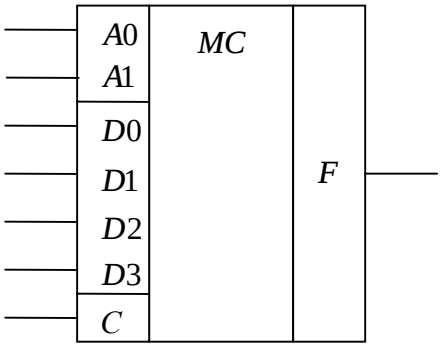


Рисунок 2.6 – Условное изображение мультиплексора

Демультиплексор – это логическое устройство, предназначенное для переключения сигнала с одного информационного входа на один из информационных выходов. В функциональном отношении он противоположен мультиплексору. На схемах демультиплексоры обозначают как DMX или DMS.

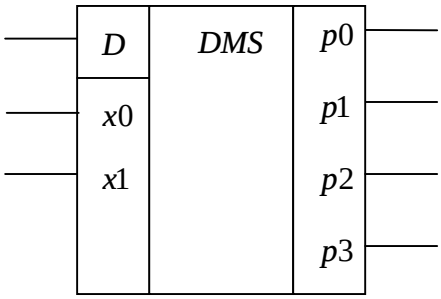


Рисунок 2.7 – Условное изображение демультиплексора

Это устройство имеет один информационный вход (D), n управляющих входов (x) и не более $m = 2^n$ информационных выходов (p). Демультиплексоры используют для подключения одного источника, например линии связи или шины данных, к нескольким приёмникам. В качестве демультиплексора обычно используют дешифратор со стробирующим входом.

2.5 Компараторы

Для решения задачи сравнения чисел в схемотехнике применяются компараторы.

Цифровой компаратор (компаратор кодов) – это логическое устройство, предназначенной для сравнения кодов, поступающих на информационные входы. Обычно имеет два информационных входа. На них подаются два разных двоичных слова, которые имеют равную длину в битах. Компаратор имеет три выхода, на которые выдаётся признак сравнения входных слов – больше, меньше или равно. Часто цифровые компараторы не имеют выходов «больше», «меньше», а только выход «равно».

Компараторы широко используются в вычислительной технике, измерительной технике, радио- и проводной связи, бытовых приборах. Например, цифровые часы с будильником содержат цифровой компаратор, при совпадении текущего времени с заданным, подается звуковой сигнал.

Построим схему простейшего компаратора для сравнения двух однобитных цифр. Входные переменные обозначим x_0 и x_1, выходная переменная y будет иметь два значения: 1 «равно» и 0 «не равно». Для решения задачи построим таблицу истинности функции y (таблица 2.5).

Таблица 2.5 Таблица истинности функции компаратора y

x_0	x_1	y
0	0	1
0	1	0
1	0	0
1	1	1

Для минимизации функции составим карту Карно (рисунок 2.8).

x_0 / x_1	0	1
0	1	0
1	0	1

Рисунок 2.8 – Карта Карно функции y

Минимизируем функцию:

$y = \bar{x}_1\bar{x}_2 \vee x_1x_2$.

Построим эту схему в Logisim в основном базисе.

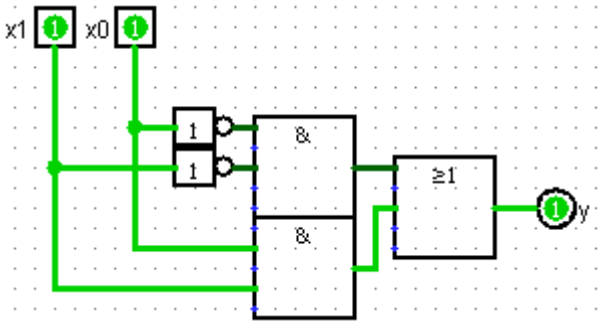


Рисунок 2.9 – Функция компаратора y в основном базисе

Проверим её работу. при подаче на вход кода $x_0x_1 = 11$ на выходе y появляется сигнал логической единицы.

Теперь построим схему сравнения на равенство двух чисел, каждое из которых включает в себя два разряда. Она также может быть получена при помощи карты Карно.

Таблица 2.6 Таблица истинности компаратора двухразрядных чисел

x_0	x_1	x_2	x_3	y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Определим входные переменные для первого числа $a = x_1 x_0$, и $b = x_3 x_2$ для второго числа. Выходная переменная y будет равна единице, если $a = b$.

Используя таблицу 2.6, составим карту Карно.

		$x_0\ x_1$			
		00	01	11	10
$x_2\ x_3$	00	1	0	0	0
	01	0	1	0	0
	11	0	0	1	0

Рисунок 2.10 – Карта Карно для минимизации функции компаратора двухразрядных чисел

Как видно из рисунка 2.10, функция не минимизируется, поэтому можно было использовать СДНФ. Любым из этих способов получим искомую логическую функцию.

$$y = \bar{x}_0 \bar{x}_1 \bar{x}_2 \bar{x}_3 \vee \bar{x}_0 x_1 \bar{x}_2 x_3 \vee x_0 x_1 x_2 x_3 \vee x_0 \bar{x}_1 x_2 \bar{x}_3 .$$

Построим схему функции y.

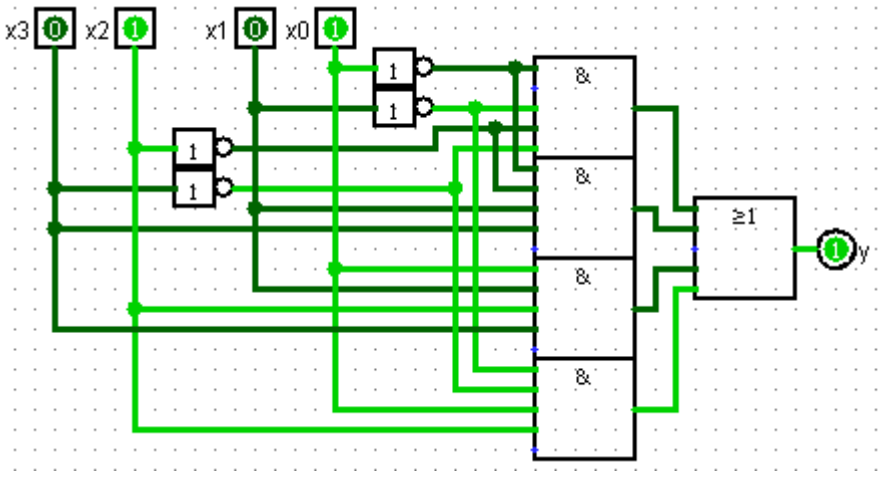


Рисунок 2.11 – Схема компаратора двухразрядных чисел в основном базисе

Проверим её работу. Действительно, если на входы подать код $x_1 x_0 = 01$ и $x_3 x_2 = 01$, то $y = 1$.

Теперь построим компаратор для сравнения двух двухрязрядных чисел с тремя выходами. Определим входные переменные для первого числа $a = x_1 x_0$, и для второго числа $b = x_3 x_2$. Выходная переменная y_0 будет равна единице, если $a = b$, выходная переменная y_1 будет равна единице, если $a > b$, и выходная переменная y_2 будет равна единице, если $a < b$.

Таблица 2.7 Условия работы компаратора с тремя выходами

Условие	Битовое условие	Равно 1
$a = b$	$x_1 x_0 = x_3 x_2$	y_0
$a > b$	$x_1 x_0 > x_3 x_2$	y_1
$a < b$	$x_1 x_0 < x_3 x_2$	y_2

Построим таблицу истинности этих функций.

Таблица 2.8 Таблица истинности компаратора с тремя выходами

b		a		$a = b$	$a > b$	$a < b$
x_3	x_2	x_1	x_0	y_0	y_1	y_2
0	0	0	0	1	0	0
0	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	0	1	0
0	1	0	0	0	0	1
0	1	0	1	1	0	0
0	1	1	0	0	1	0
0	1	1	1	0	1	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	1	0	0
1	0	1	1	0	1	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	1	0	0

Теперь составим карты Карно для этих трёх функций.

		$x_1\ x_0$			
		00	01	11	10
$x_3\ x_2$	00	1	0	0	0
	01	0	1	0	0
	11	0	0	1	0

Рисунок 2.12 – Карта Карно функции y_0

Минимизируем функцию y_0.

$y_0 = \bar{x}_0\bar{x}_1\bar{x}_2\bar{x}_3 \vee \bar{x}_0x_1\bar{x}_2x_3 \vee x_0x_1x_2x_3 \vee x_0\bar{x}_1x_2\bar{x}_3$

		$x_1\ x_0$			
		00	01	11	10
$x_3\ x_2$	00	0	1	1	1
	01	0	0	1	1
	11	0	0	0	0
	10	0	0	1	0

Рисунок 2.13 – Карта Карно функции y_1

Минимизируем функцию y_1.

$y_1 = x_0\bar{x}_2\bar{x}_3 \vee x_1\bar{x}_3 \vee x_0x_1\bar{x}_2$.

		$x_1\ x_0$			
		00	01	11	10
$x_3\ x_2$	00	0	0	0	0
	01	1	0	0	0
	11	1	1	0	1

Рисунок 2.14 – Карта Карно функции y_2

Минимизируем функцию y_2.

$y_2 = \bar{x}_0\bar{x}_1x_2 \vee \bar{x}_1x_3 \vee \bar{x}_0x_2x_3$.

Построим эту схему функций y_0, y_1 и y_2 в Logisim.

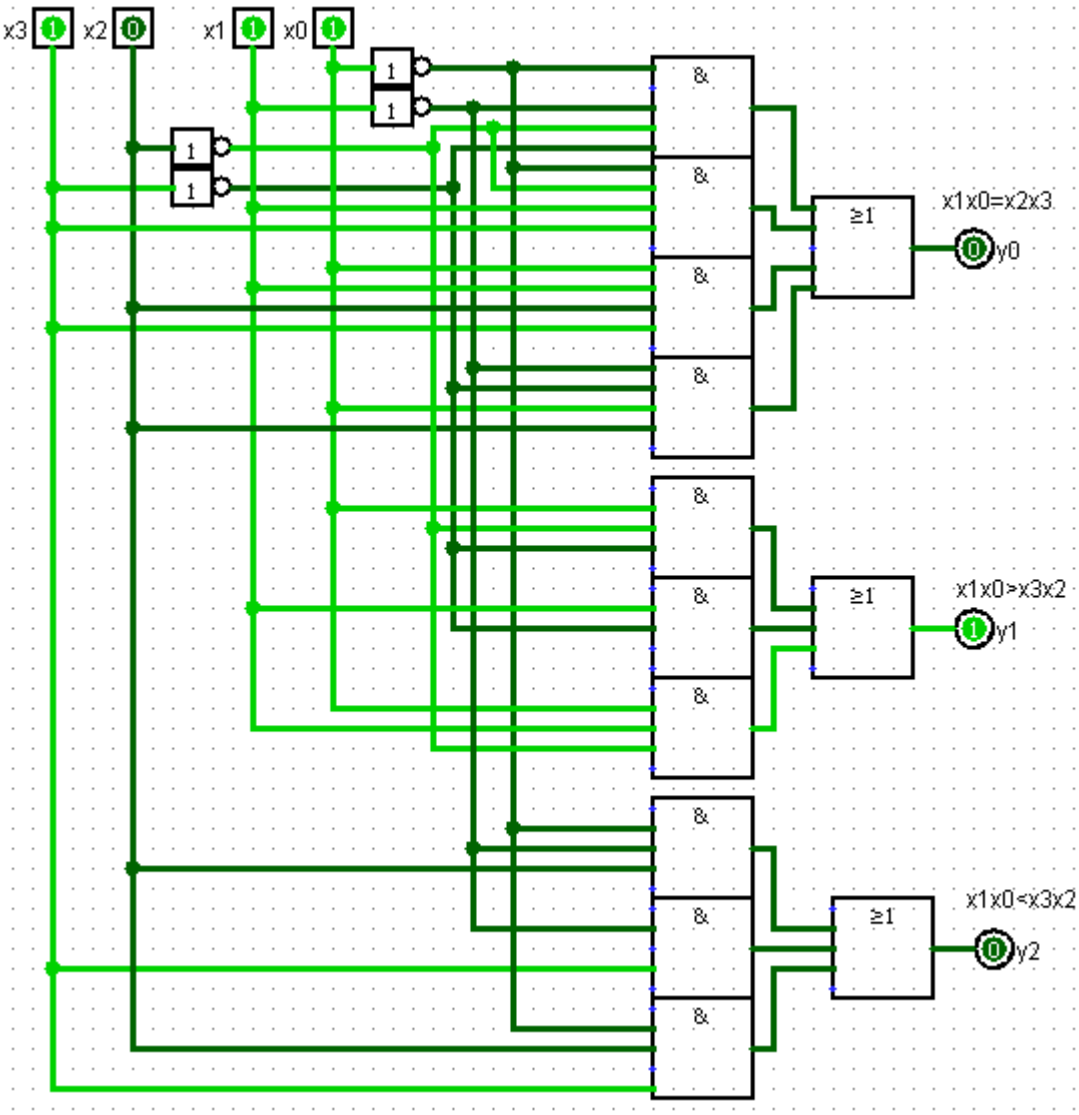


Рисунок 2.15 – Схема компаратора с тремя выходами

Проверим её работу. Действительно, при подаче на входы кодов $x_3x_2 = 10$ и $x_1x_0 = 11$ на выходе y_1 появляется единица.

2.6 Сумматоры

Для выполнения арифметических действий в процессорах используется арифметико-логическое устройство (АЛУ). Важной частью этого устройства являются сумматоры, выполняющие арифметическое сложение чисел.

Сумматор – это цифровое устройство, предназначенное для арифметического сложения чисел, представленных в виде двоичных кодов.

Существует много вариантов конструкций сумматоров с различными параметрами сложности и быстродействия.

Рассмотрим следующие типы сумматоров:

1. Одноразрядный;
2. Многоразрядный для последовательных операндов;
3. Многоразрядные для параллельных операндов:
   - с последовательным переносом;
   - с параллельным переносом.

Существуют и другие типы сумматоров. Наряду с сумматорами могут быть реализованы вычитатели. На практике их используют редко, поскольку вычитание удобнее выполняется через сложение с применением дополнительных либо обратных кодов.

По архитектуре различают полусумматоры и полные сумматоры.

Полусумматор производит сложение двух операндов по модулю с разрядом переноса. Имеют два входа, на которые подаются одноимённые разряды двух чисел, и два выхода. На одном реализуется арифметическая сумма по модулю в данном разряде, а на другом – перенос в следующий (старший) разряд.

Таблица 2.9 Таблица истинности одноразрядного полусумматора

Набор	Слагаемые		Результат	
	A	B	S	P
1	0	0	0	0
2	1	0	1	0
3	0	1	1	0
4	1	1	0	1

Запишем логические функции для суммы S и переноса P.

$S = \overline{A}B \vee \overline{B}A$,

$P = AB$.

Построим схему одноразрядного полусумматора.

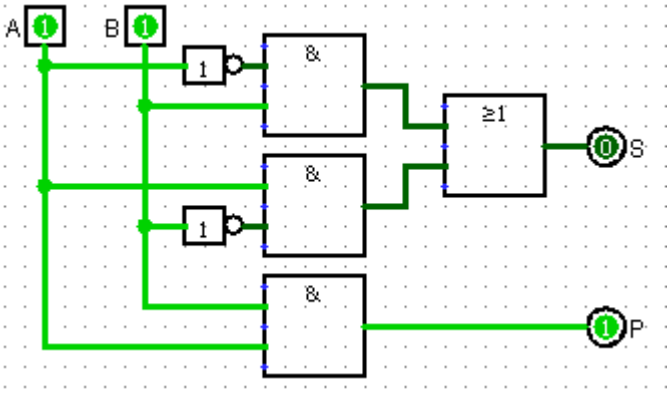


Рисунок 2.16 – Схема одноразрядного полусумматора в основном базисе

На схемах полусумматоры обозначают как HS.

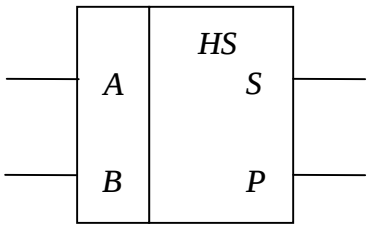


Рисунок 2.17 – Условное обозначение полусумматора

Устройство имеет два информационных входа (A и B), а также выход суммы S и переноса P.

Полные сумматоры производят сложение трёх операндов по модулю с разрядом переноса. Имеют три входа, на которые подаются одноимённые разряды двух чисел и перенос из предыдущего (младшего) разряда, и два выхода. На одном реализуется арифметическая сумма по модулю в данном разряде, а на другом – перенос в следующий (старший) разряд.

Таблица 2.10 Таблица истинности одноразрядного полного сумматора

Набор	Слагаемые			Результат	
	A	*B*	*P* –	*S*	*P* +
1	0	0	0	0	0
2	0	0	1	1	
3	0	1	0	1	0
4	0	1	1	0	1
5	1	0	0	1	0
6	1	0	1	0	1
7	1	1	0	0	1
8	1	1	1	1	1

Реализуем схему полного одноразрядного сумматора.

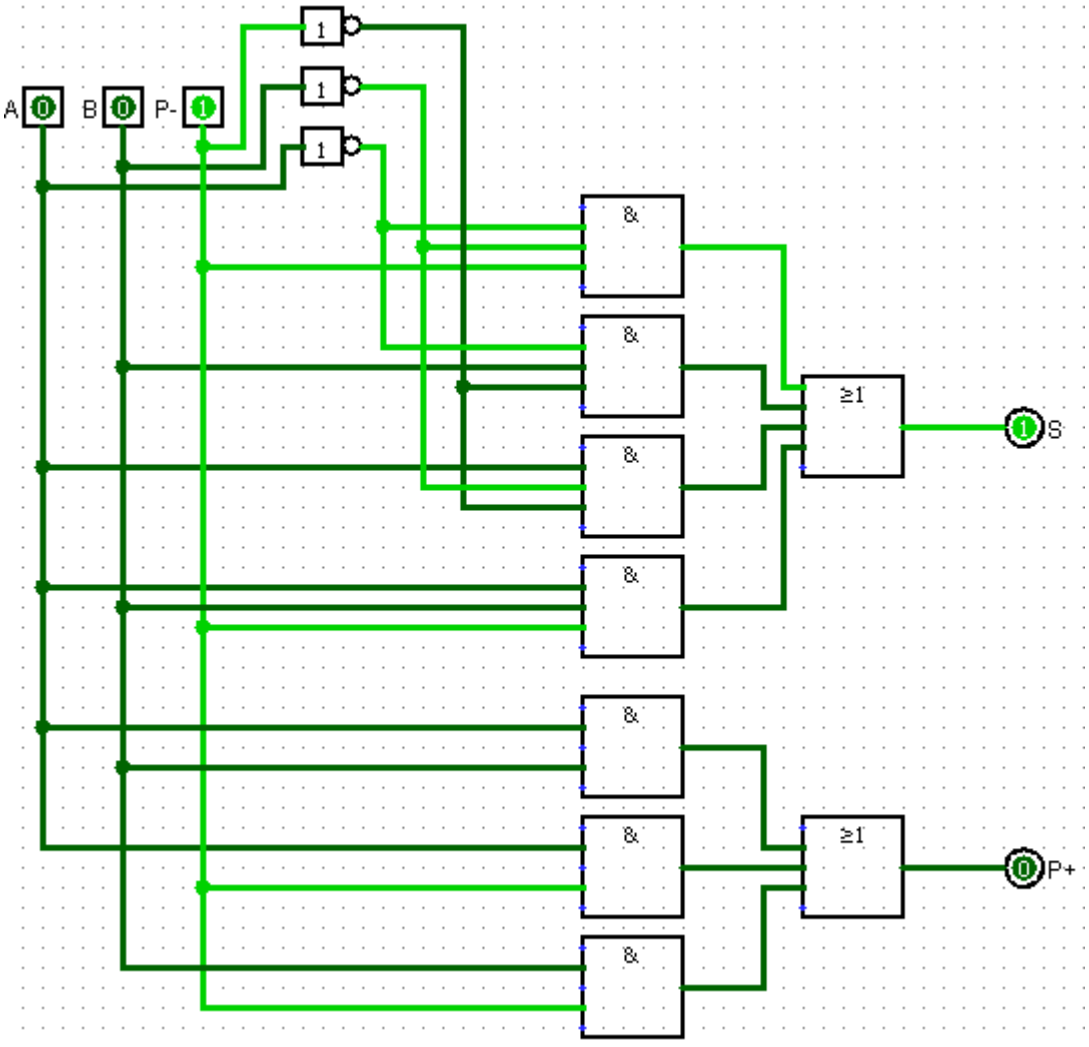


Рисунок 2.18 – Схема полного одноразрядного сумматора

На схемах полные сумматоры обозначают как SM .

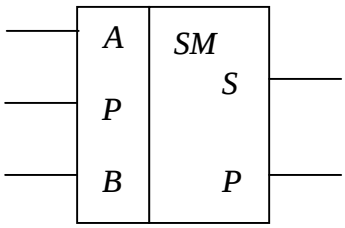


Рисунок 2.19 – Условное обозначение полного сумматора

Это устройство имеет те же входы и выходы, что и полусумматор, а также дополнительно вход P переноса из предыдущего разряда.

Обычно многоразрядный сумматор представляет собой комбинацию одноразрядных сумматоров (рисунок 2.20). При сложении двух чисел в каждом разряде производится сложение трех цифр: цифры первого слагаемого, цифры второго слагаемого и цифры переноса из младшего разряда. В результате суммирования на выходных шинах получается сумма и перенос в старший разряд.

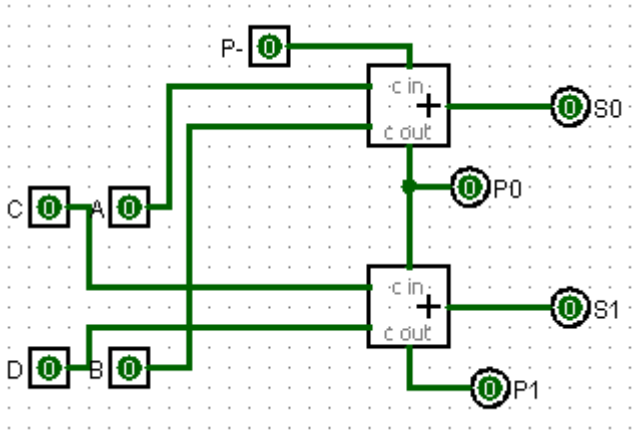


Рисунок 2.20 – Двухразрядный сумматор

В сумматоре для последовательных операндов один одноразрядный сумматор, обрабатывает разряды поочередно, начиная с младшего (рисунок 2.21). Сложив младшие разряды A_0 и B_0, сумматор вырабатывает сумму S_0 для младшего разряда результата, а также перенос, который запоминается D-триггером на один такт. В следующем такте складываются вновь поступившие разряды слагаемых A_1 и B_1 с переносом C_0 из младшего разряда и получается следующий разряд суммы S_1 и перенос из него и т. д.

Схема сумматора, помимо одноразрядного сумматора SM, содержит сдвигающие регистры слагаемых и суммы, а также триггер запоминания переноса. Регистры и триггер тактируются синхроимпульсами.

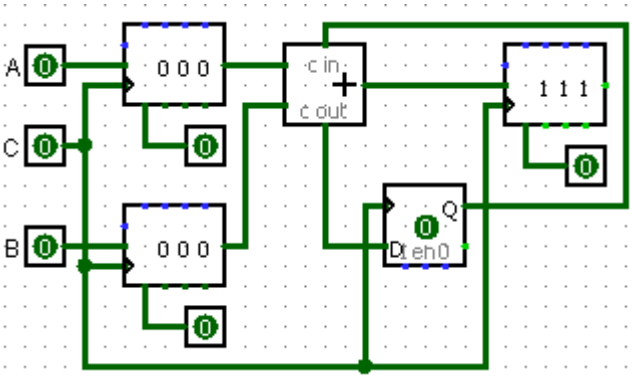


Рисунок 2.21 – Сумматор для последовательных операндов

Записав последовательно в регистры, например, A=100 и B=011, получим в выходном регистре S=111 или, в десятичном виде, 4+3=7.

Сумматор с поразрядным последовательным переносом наиболее прост с точки зрения схемной реализации, однако имеет низкое быстродействие. Оно ограничено временем, за которое сигнал переноса последовательно распространяется через все разряды сумматора. Для повышения быстродействия используются сумматоры с параллельным переносом.

Многоразрядный сумматор для параллельных операндов с последовательным переносом представляет собой группу одноразрядных сумматоров, соединённых последовательно по цепям переноса.

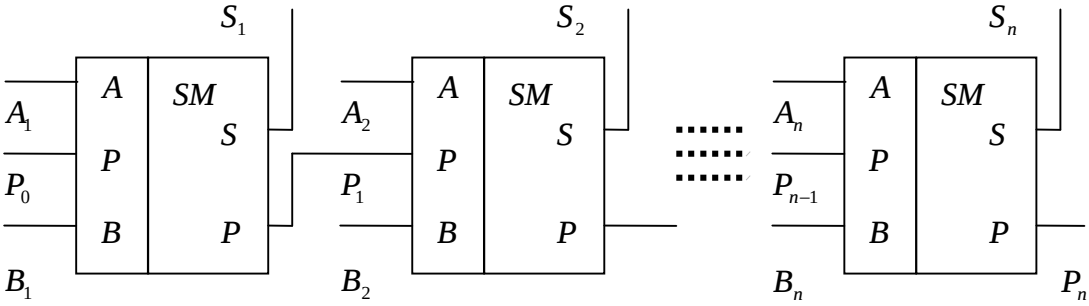


Рисунок 2.22 – Многоразрядный сумматор для параллельных операндов с последовательным переносом

Как и в других схемах с последовательным распространением сигналов от разряда к разряду, здесь время суммирования при достаточно больших n практически пропорционально разрядности сумматора.

Сумматоры для параллельных операндов с параллельным переносом разработаны для получения максимального быстродействия (рисунок 2.22). Сумматоры с параллельным переносом не имеют последовательного распространения переноса вдоль разрядной сетки. Во всех разрядах результаты вырабатываются одновременно, параллельно во времени. Сигналы переноса для данного разряда формируются специальными схемами, на входы которых поступают все переменные, необходимые для выработки переноса.

Перенос в следующий разряд состоит из входного переноса и значения всех разрядов слагаемых, младших относительно данного. Схема строится на одноразрядных сумматорах. В них выход переноса не используется, так как перенос формируется специальными схемами.

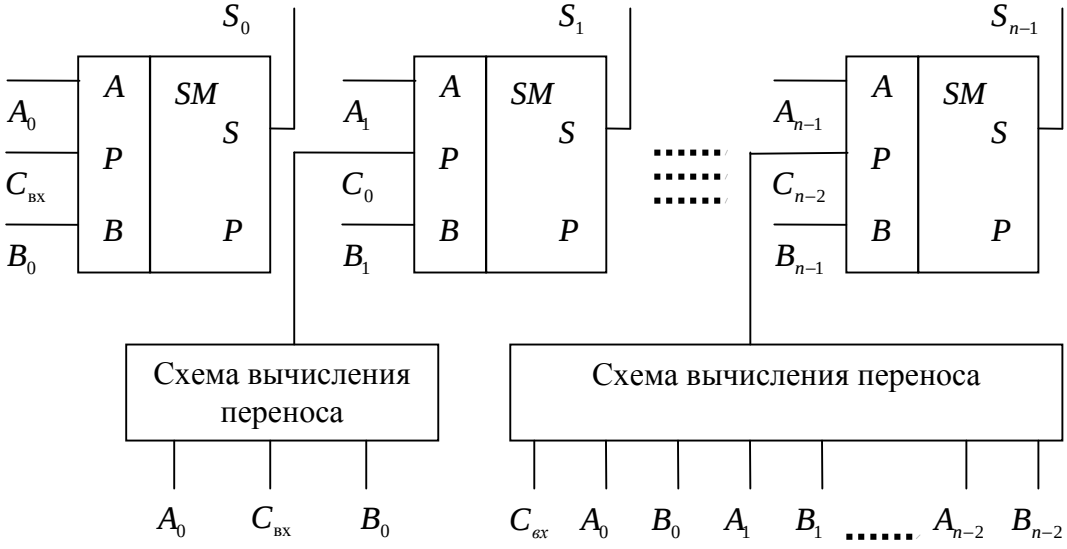


Рисунок 2.23 – Многоразрядный сумматор для параллельных операндов с параллельным переносом

Для получения схемы сумматора используются две вспомогательные функции.

Функция генерации принимает единичное значение, если перенос на выходе данного разряда появляется независимо от наличия или отсутствия входного переноса. Очевидно, что эта функция $g_i = a_i b_i$.

Функция прозрачности принимает единичное значение, если перенос на выходе данного разряда появляется только при наличии входного переноса. То есть, функция прозрачности должна выражаться как «исключающее ИЛИ», т. е. $h_i = a_i\bar{b}_i \vee \bar{a}_i b_i$. При этом разница между функциями ИЛИ и «исключающее ИЛИ» проявляется только при $a_i = b_i = 1$, т. е. в ситуации, где перенос все равно формируется из-за g_i = 1, то есть для упрощения схемы допустимо заменить эту функцию на $h_i = a_i \vee b_i$.

Таким образом, перенос на выходе разряда либо генерируется им, либо проходит от младшего разряда при прозрачности данного.

$C_i = g_i \vee h_i C_{i-1}$.

Таким образом, для младшего разряда сумматора можно записать:

$C_0 = g_0 \vee h_0 C_{вх}$.

Аналогично, на выходе следующего разряда:

$C_1 = g_1 \vee h_1 C_0$.

Подставим в эту формулу соотношение для переноса предыдущего разряда, получим:

$C_1 = g_1 \vee h_1 g_0 \vee h_1 h_0 C_{вх}$.

Для следующего разряда:

$C_2 = g_2 \vee h_2 C_1 = g_2 \vee h_2 g_1 \vee h_2 h_1 g_0 \vee h_2 h_1 h_0 C_{вх}$.

Из формул видно, что перенос на выходе разряда формируется или в нём непосредственно, или приходит от предыдущего, младшего разряда. Для произвольного разряда можно записать:

$C_i = g_i \vee g_{i-1} h_i \vee g_{i-2} h_i h_{i-1} \vee ... \vee g_0 h_i h_{i-1} ... h_1 \vee h_i h_{i-1} ... h_1 C_{вх}$.

Построим сумматор параллельных операндов с параллельным переносом для четырёхразрядных чисел с входным переносом в нулевой разряд. Функции переноса в первый, второй и третий разряды получены выше. Произведём подстановку $g_i = a_i b_i$ и преобразуем выражения в базис И-НЕ при помощи закона Де Моргана.

$h_0 = a_0 \vee b_0 = \overline{\bar{a}_0 \cdot \bar{b}_0}$,

$h_0 = a_1 \vee b_1 = \overline{\bar{a}_1 \cdot \bar{b}_1}$,

$h_2 = a_2 \vee b_2 = \overline{\bar{a}_2 \cdot \bar{b}_2}$,

$C_0 = g_0 \vee h_0 C_{вх} = a_0 b_0 \vee h_0 C_{вх} = \overline{\overline{a_0 b_0} \cdot \overline{h_0 C_{вх}}}$,

$$C_1 = g_1 \vee h_1 g_0 \vee h_1 h_0 C_{вх} = \overline{\overline{a_1 b_1} \cdot \overline{h_1 a_0 b_0} \cdot \overline{h_1 h_0 C_{вх}}},$$

$$C_2 = g_2 \vee h_2 C_1 = g_2 \vee h_2 g_1 \vee h_2 h_1 g_0 \vee h_2 h_1 h_0 C_{вх} =$$

$$= a_2 b_2 \vee h_2 a_1 b_1 \vee h_2 h_1 a_0 b_0 \vee h_2 h_1 h_0 C_{вх} = \overline{\overline{a_2 b_2} \vee \overline{h_2 a_1 b_1} \vee \overline{h_2 h_1 a_0 b_0} \vee \overline{h_2 h_1 h_0 C_{вх}}}.$$

Построим эту схему в Logisim.

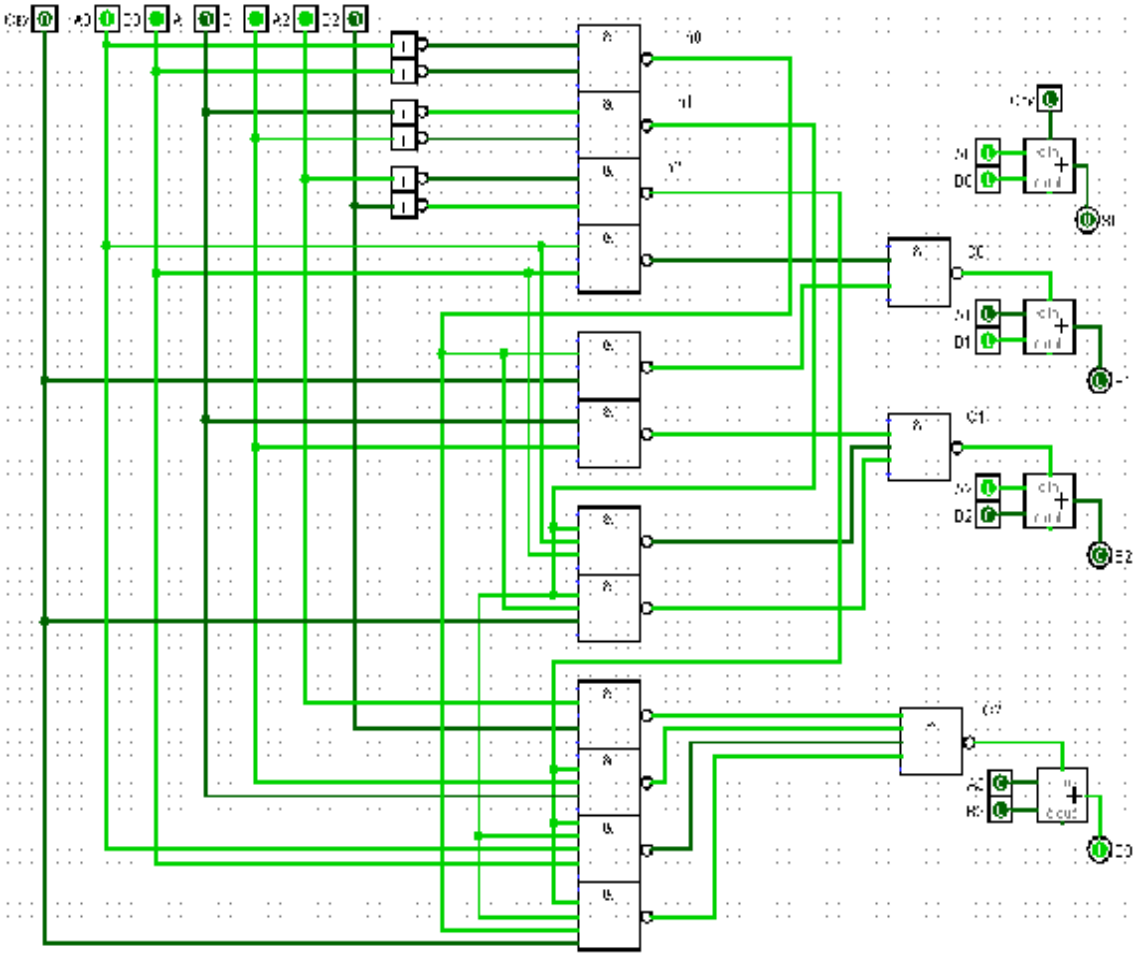


Рисунок 2.24 – Сумматор параллельных операндов с параллельным переносом для четырёхразрядных чисел с входным переносом в нулевой разряд

Длительность суммирования рассматриваемой логической схемы теоретически не зависит от разрядности n сумматора, что характерно для сумматоров с параллельным переносом. На практике это не совсем так. С ростом разрядности сумматора увеличивается нагрузка на элементы схемы, что увеличивает их задержки. Поэму рост разрядности всё же замедляет процесс суммирования, хотя и в гораздо меньшей степени, чем в схемах с последовательным переносом.

Такие сумматоры имеют преимущество для схем до $n = 4$, так как их схемы более просты. После этого до $n = 8$ проявляются преимущества схем с параллельным переносом. Затем в них появляются элементы с большим числом входов, замедляющих работу сумматора.

В схемотехнике используются и другие виды сумматоров для параллельных операндов, отличающиеся быстродействием.

Сумматоры с групповой структурой используют различные виды переносов в одной схеме.

Также применяются сумматоры с условным переносом. В них младшие разряды суммируются одновременно со старшими. При этом старшие суммируются одновременно при условии наличия переноса из младших и при условии его отсутствия. К тому моменту, когда становится известен факт наличия переноса из младших разрядов, старшие уже просуммированы для обоих случаев, и при помощи мультиплексора выбирается нужный результат.

Накапливающий сумматор представляет собой сочетание комбинационного сумматора и сдвигового регистра.

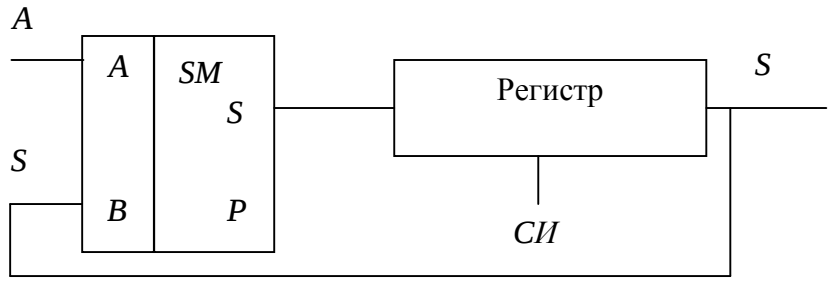


Рисунок 2.25 – Накапливающий сумматор

Устройство работает по формуле $S = S + A$, согласно которой к содержимому сумматора добавляется очередное слагаемое, и результат замещает прежнее значение суммы. Очередное прибавление слагаемого тактируется синхроимпульсами. Учитывая особенности функционирования, накапливающие сумматоры называют иногда аккумуляторами.

Контрольные вопросы к главе 2

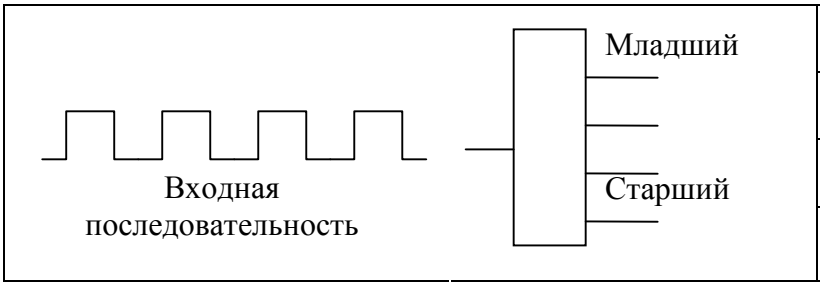
1. Что такое цифровой автомат?
2. Чем характеризуется ЦА, что такое конечный автомат?
3. Что называют комбинационным автоматом?
4. Что называют последовательностным автоматом?
5. Как составляется таблица переходов для синтеза автомата Мура?
6. Как задать автомат Мура при помощи графа?
7. Опишите матричное представление автомата Мура.
8. Какие столбцы содержит табличная форма матрицы перехода автомата Мура и сколько в ней строк?
9. Что такое дешифратор, для чего он применяется?
10. Как описывается дешифратор?
11. Постройте дешифратор на три входа.
12. Постройте ступенчатый дешифратор на четыре входа.
13. Каково условное обозначение дешифратора?
14. Что такое мультиплексор, для чего он применяется?
15. Опишите работу мультиплексора.
16. Постройте мультиплексор на четыре информационных входа.
17. Каково условное обозначение мультиплексора?
18. Опишите работу демультиплексора и его условное обозначение.
19. Что такое шифратор и для чего он применяется?
20. Постройте шифратор на восемь входов.
21. Что такое компараторы, для чего они применяются?
22. Постройте схему компаратора для двух однобитных цифр.
23. Постройте схему компаратора для двух двухбитных чисел.
24. Постройте компаратор двух двухбитных чисел для сравнения на превышение и на равенство.
25. Что такое сумматор и для чего он применяется, какие виды сумматоров существуют?
26. Постройте полусумматор.
27. Постройте схему полного сумматора. Каково его условное обозначение?
28. Постройте двухразрядный последовательный сумматор на базе двух полных сумматоров.

29. Как работает сумматор для последовательных операндов? Постройте его схему.
30. Постройте схему последовательного сумматора для параллельных операндов для $n-1$ разрядов и расскажите об её работе.
31. В чём недостаток сумматоров с последовательным переносом, для чего разработаны сумматоры с параллельным переносом?
32. Постройте общую схему сумматора с параллельным переносом для $n-1$ разрядов и расскажите об её работе.
33. Как получить вспомогательные функции, описывающие сумматор с параллельным переносом для параллельных операндов?
34. Какова формула формирования переноса для любого числа разрядов параллельного сумматора?
35. Постройте сумматор параллельных операндов с параллельным переносом для четырёхразрядных чисел с входным переносом в нулевой разряд.
36. От чего зависит скорость суммирования в сумматорах с параллельным переносом, для какого числа разрядов они наиболее оптимальны? Где лучше применить последовательные сумматоры?
37. Расскажите о работе накапливающего сумматора, постройте его схему.

Глава 3 Цифровые схемы последовательностного типа

3.1 Формы представления цифровых автоматов с памятью

Последовательностые автоматы обладают элементами памяти, которые хранят предысторию автомата. Поэтому выходные сигналы такого автомата зависят не только от того, какие входные сигналы были поданы на вход автомата, но и от того, какая последовательность сигналов была подана на вход до этого. То есть, в отличие от комбинационных, автоматы с памятью могут неоднозначно реагировать на одну и ту же последовательность входных сигналов. Примером автомата с памятью является двоичный четырёхразрядный счётчик.



0	1	0	1
0	0	1	1
0	0	0	0
0	0	0	0

Рисунок 3.1 – Процесс функционирования четырёхразрядного счётчика

Методы синтеза и минимизации комбинационных автоматов были рассмотрены в ранее. Последовательностные автоматы являются более сложными устройствами, чем комбинационные, но в конечном счёте их синтез сводится к синтезу комбинационных схем, выбору элементов памяти и соединению их определённым образом.

Работа ЦА осуществляется в автоматном времени, определяемом числом периодов поступления входных сигналов. При анализе и синтезе автоматов с памятью предполагается, что входные и выходные сигналы появляются не непрерывно, а только в некоторые фиксированные моменты времени t. Эти моменты обычно обозначаются цифрами натурального ряда (0, 1, 2, …). При этом поведение автомата в промежутке между соседними моментами времени является несущественным и не рассматривается.

Существует два способа задания автоматного времени. По этому признаку выделяют два типа автоматов с памятью: асинхронные и синхронные.

В синхронных автоматное время задаётся при помощи источника тактирующих сигналов, который является внешним по отношению к схеме автомата. У асинхронных автоматов моменты автоматного времени будут являться моментами изменения сигнала на выходе. У синхронного автомата очередной символ входной последовательности будет воспринят только в том случае, если в тот же момент подан тактирующий сигнал. Поэтому моменты автоматного времени для синхронного автомата являются моментами подачи тактирующего сигнала.

Существуют различные модели автоматов с памятью. Одной из наиболее употребительных является модель Клини.

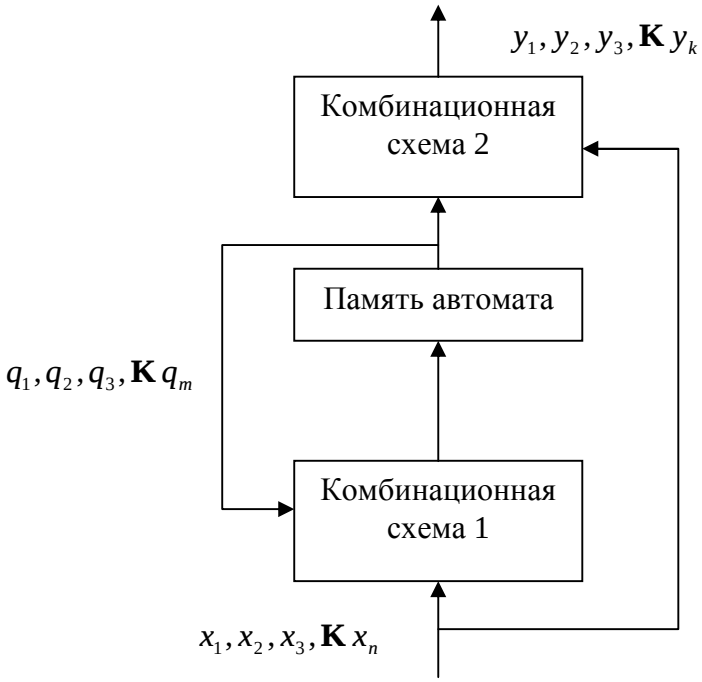


Рисунок 3.2 – Модель Клини автомата с памятью

В этой модели $x_1, x_2, x_3, \mathbf{K}\, x_n$ входные сигналы (переменные), $q_1, q_2, q_3, \mathbf{K}\, q_m$ - внутренние сигналы (переменные), $y_1, y_2, y_3, \mathbf{K}\, y_k$ - выходные сигналы (переменные). Все возможные наборы входных, внутренних и выходных переменных образуют входной X, внутренний Q и выходной Y алфавиты автомата. Конечный автомат задаётся этими тремя алфавитами.

$A = \{X, Q, Y\}$.

Если числа n, m, k конечны, то автомат является абстрактным конечным. На практике применяются именно конечные автоматы. Они осуществляют преобразование входного алфавита в выходной. Это преобразование определяется характеристическими функциями переходов и выходов.

Функция переходов определяет состояние автомата в момент времени $t+1$ в зависимости от его входного и внутреннего состояний в момент времени t.

$q(t+1) = d(q(t), x(t))$.

Функция выходов определяет значение выходного сигнала $y(t)$. Эта функция может быть задана двумя способами.

Если выходное состояние автомата в момент времени t однозначно определятся его внутренним состоянием в тот же момент (автомат без памяти):

$y(t) = l(q(t))$.

Если выходное состояние автомата определяется внутренним и входным состоянием автомата в тот же момент времени (автомат с памятью):

$y(t) = l(q(t), x(t))$.

Кроме внутреннего и внешнего состояния, на выходной сигнал влияет также и начальное состояние q_0 автомата.

Существует несколько способов задания автоматов Мили (с памятью) и Мура (без памяти). В данной работе будут рассмотрены способы:

- при помощи таблиц переходов и выходов,
- с использованием графов,
- матрица переходов и выходов,
- табличная форма матрицы.

Все эти способы являются формами представления модели абстрактного конечного автомата Мили или Мура и однозначно связаны между собой.

Таблицы переходов и выходов автомата Мили содержит столько строк. Сколько входных состояний у автомата, и столько столбцов, сколько у него внутренних состояний. Таким образом, задаются все возможные пары входных x_i и внутренних q_j состояний в момент времени t. На пересечении i-й строки и j-го столбца записывается то внутреннее состояние q_{ij}, в которое автомат переходит в момент времени $t+1$, если в предыдущий момент времени t он находился во входном состоянии x_i и во внутреннем состоянии q_j.

Рассмотрим в качестве примера абстрактный конечный автомат с памятью, имеющий три входных состояния x_1, x_2, x_3 и три внутренних состояния q_1, q_2, q_3, и заданный таблицей переходов.

Таблица 3.1 Переходы автомата с памятью

	q_1	q_2	q_3
x_1	q_1	q_3	q_2
x_2	q_3	q_1	q_2
x_3	q_1	q_2	q_1

Эта таблица показывает, например, что при если автомат находится в состоянии q_1, то при подаче на вход автомата состояния x_2, автомат перейдёт в состояние q_3.

Таблица выходов автомата Мили имеет столько же строк и столбцов, сколько таблица переходов. Но на пересечениях проставляются выходные сигналы y_i автомата, которые появляются на выходе автомата для данного перехода.

Таблица 3.2 Выходы автомата с памятью

	q_1	q_2	q_3
x_1	y_1	y_1	y_2
x_2	y_1	y_2	y_1
x_3	y_2	y_2	y_1

Часто эти две таблицы объединяют в одну, называемую таблицей переходов и выходов.

Таблица 3.3 Переходы и выходы автомата с памятью

	q_1	q_2	q_3
x_1	q_1 / y_1	q_3 / y_1	q_2 / y_2
x_2	q_3 / y_1	q_1 / y_2	q_2 / y_1
x_3	q_1 / y_2	q_2 / y_2	q_1 / y_1

Если переходы автомата определены не полностью, то он называется не полностью определённым или частичным конечным автоматом. В клетках таблицы переходов, соответствующих неопределённым переходам, ставятся прочерки.

Другим способом представления конечных автоматов является задание их при помощи графов. Для обоих типов автоматов их состояние представляется вершинами графа, а переходы из одного состояния в другое изображаются при помощи направленных дуг. Состояния, вызывающие переходы, приписываются соответствующим дугам графа. Для автомата Мили (с памятью) выходной сигнал ставится в соответствие дуге графа. Если переход автомата из одного состояния в другое может происходить под воздействием двух различных комбинаций входных сигналов, то дуги, соответствующие этим переходам, можно заменить одной дугой, которой соответствует дизъюнкция этих состояний. Ниже приведёны фрагменты автомата Мили (а, б).

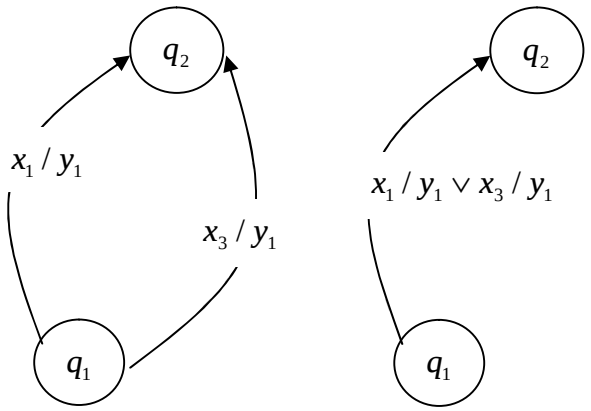


Рисунок 3.3 – Представление конечных автоматов при помощи графов

Матричное представление абстрактного автомата является таблицей, отражающей его структурные особенности. Матрица автомата Мили представляет собой квадратную таблицу, строки которой соответствуют текущим внутренним состояниям автомата, т.е. состояниям во время t, а столбцы соответствуют внутренним состояниям автомата в следующий момент времени, т.е. в момент $t+1$. На пересечениях строк и столбцов указываются условия, вызывающие этот переход. Эти условия представляют собой входное состояние с записанным через косую черту выходным состоянием.

При этом некоторые пересечения могут быть пустыми. Это означает, что соответствующий переход отсутствует. В клетках матрицы переходов содержатся записи, относящиеся к дугам графа.

Рассмотрим в качестве примера автомат Мили, имеющий три входных состояния x_1, x_2, x_3 и три внутренних состояния q_1, q_2, q_3, и заданный следующей матрицей переходов и выходов.

Таблица 3.4 Матрица переходов и выходов автомата

q_{t+1} / q_t	q_1	q_2	q_3
q_1	–	–	x_2 / y_1
q_2	x_2 / y_2	x_3 / y_2	x_1 / y_1
q_3	x_3 / y_1	$x_1 / y_2 \vee x_2 / y_1$	–

Из матрицы следует, что из состояния q_1 невозможно сразу перейти в состояние q_2.

Построим граф этого автомата.

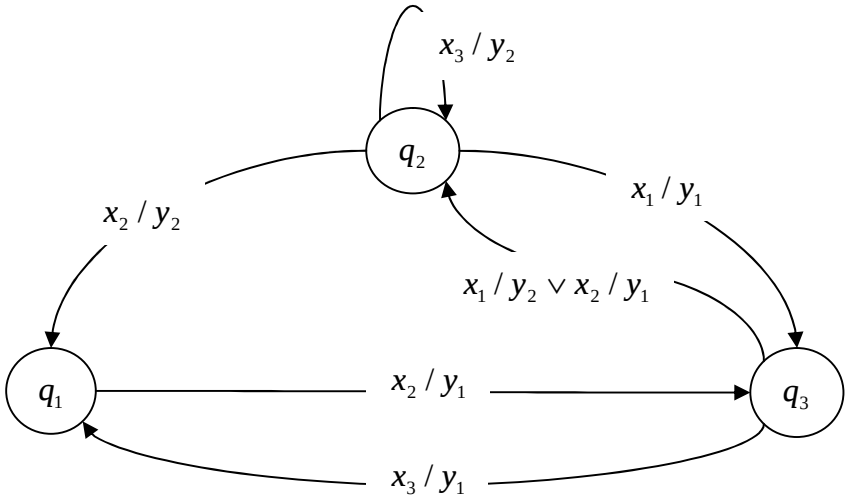


Рисунок 3.4 – Представление автомата Мили при помощи графа

Кроме представления в виде таблицы переходов и выходов, графа или матрицы модель цифрового автомата Мили или Мура можно задать в виде табличной формы матрицы переходов.

В случае автомата с памятью она состоит из трёх столбцов: состояние в текущий момент времени q_t, состояние в следующий момент времени q_{t+1} и условие перехода из одного состояния в другое.

Представим рассмотренный выше автомат Мили в такой форме.

Таблица 3.5 Табличная форма матрицы переходов автомата

q_t	q_{t+1}	Условие перехода
q_1	q_1	–
q_1	q_2	–
q_1	q_3	x_2 / y_1
q_2	q_1	x_2 / y_2
q_2	q_2	x_3 / y_2
q_2	q_3	x_1 / y_1
q_3	q_1	x_3 / y_1
q_3	q_2	$x_1 / y_2 \vee x_2 / y_1$
q_3	q_3	–

Количество строк в табличной форме матрицы переходов определяется числом внутренних состояний q автомата Мили и составляет m^2, где m – число внутренних состояний. Таким образом, для большого числа внутренних состояний такая форма представления модели будет слишком большой по сравнению с матричным представлением.

3.2 Элементарные конечные автоматы

3.2.1 Однотактный RS-триггер

Под синтезом комбинационных схем понимают процесс создания цифровых схем, реализующих функции заданных абстрактных цифровых автоматов. Процесс синтеза заключается в выборе типов запоминающих устройств и синтезе связанных с ними комбинационных схем. Ниже будет рассмотрено функционирование элементов памяти – элементарных конечных автоматов, или триггеров.

Триггеры предназначены для запоминания двоичной информации. Использование триггеров позволяет реализовывать устройства оперативной памяти. Также триггеры могут использоваться и для построения цифровых устройств с памятью, таких как счётчики, преобразователи последовательного кода в параллельный и т.д. Триггеры являются автоматами с двумя устойчивыми состояниями, обладающие полнотой переходов и выходов. Полнота переходов определяется наличием хотя бы одного входного сигнала, который переводит автомат из одного состояния в другое. Полнота выхода означает, что двум различным внутренним состояниям соответствуют два различных выходных сигнала.

При синтезе конечного автомата с памятью в качестве элементарных автоматов используются триггеры, схемы которых отличаются друг от друга как числом входов, так и логикой функционирования. При этом все триггеры являются автоматами Мура (без памяти), а автоматы, использующие триггеры – автоматами Мили (с памятью).

Рассмотрим некоторые особенности описания триггеров.

Внутреннее состояние триггера обозначается буквой Q (или q). При Q=1 триггер находится в единичном состоянии, при Q=0 – в нулевом состоянии.

Выходной сигнал полностью повторяет внутреннее состояние триггера и также обозначается Q.

Триггер имеет два выхода, сигналы на которых взаимно инверсны: Q и $\overline{Q}$.

Если триггер является синхронным, то импульсы синхронизации подаются на специальный синхронизирующий вход и обозначаются C.

Все триггеры, как правило, имеют два типа входов: рабочие и установочные. Вход установки триггера в единичное состояние обозначается S (*Set*), вход установки триггера в нулевое состояние обозначается R (*Reset*). Установочные входы всегда асинхронны. Условное графическое обозначение триггера:

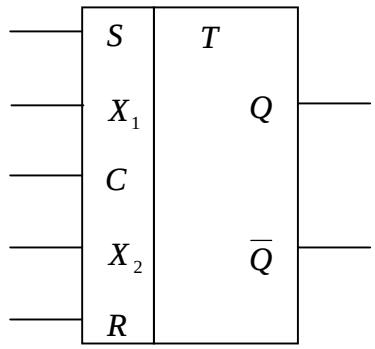


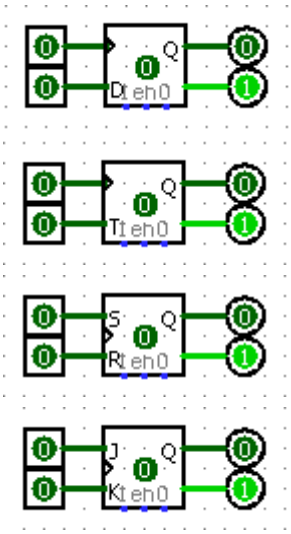
Рисунок 3.5 – Условное обозначение триггера

Детали этого обозначения могут меняться. Тип триггера, обозначенный на данном рисунке как T, обозначает одноступенчатый (однотактный) триггер. Двухступенчатое (двухтактное) устройство такого типа обозначается TT.

Рабочие входы обозначены X_1 и X_2. Эти обозначения меняются в зависимости от типа триггера. Наиболее распространёнными типами являются D (*delay*, триггер задержки), T (счётный) RS (*ResetSet*, он же SR) и JK (*JabKeep*, универсальный).

Входы S и R являются установочными, вход C - синхронизирующим. Эти обозначения, как правило, не меняются.

В программе моделирования логических схем Logisim используются немного другие обозначения элементов.



D – триггер

T – триггер

RS – триггер

JK – триггер

Рисунок 3.6 – Элементы D, T, RS и JK-триггеры в Logisim

На этих обозначениях установочные входы расположены снизу.

Основным триггером, на котором базируются все остальные, является RS -триггер. Он имеет два логических входа: R - установка 0, и S - установка 1. Также RS -триггер имеет два выхода: Q - прямой, $\overline{Q}$ - обратный (инверсный). Состояние триггера определяется состоянием прямого выхода. Простейший асинхронный RS-триггер состоит из двух логических элементов, связанных перекрёстной положительной обратной связью:

Рассмотрим работу однотактного асинхронного триггера. Пусть рабочие входы триггера находятся в состоянии $R=0$, $S=1$. Нижний двухвходовой логический элемент выполняет логическую функцию И-НЕ, т.е. единица на любом его входе приводит к тому, что на его выходе будет логическая единица $Q2=1$. Но в то же время на оба входа верхнего элемента И-НЕ поданы нули, что приводит к появлению единицы на его выходе $Q1=1$. В то же время эта единица приходит на вход нижнего элемента, поэтому на его выходе получаем $Q2=0$. Триггер находится в единичном состоянии $Q1=1$, $Q2=0$.

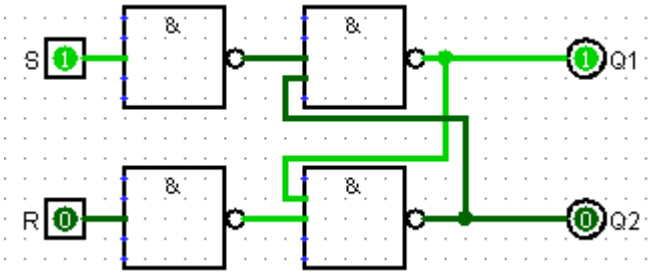


Рисунок 3.7 – Запись единицы в RS -триггер

Если теперь на входы подать $S=0$ и $R=0$, то на выходе ситуация не изменится, т.к. несмотря на то, что на нижний вход нижнего логического элемента будет поступать 0, на его верхний вход поступает 1 с выхода верхнего логического элемента. Триггер будет находиться в единичном состоянии, пока на вход R не поступит сигнал сброса $R=1$.

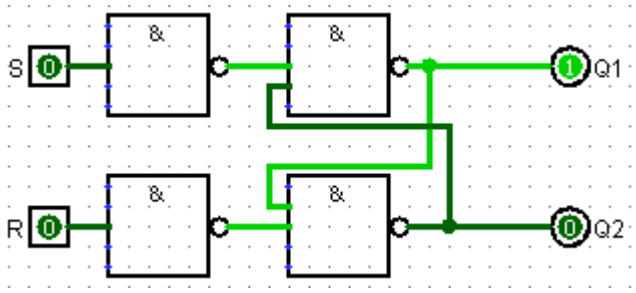


Рисунок 3.8 – Хранение единицы в RS -триггере

Пусть теперь $R=1$, $S=0$. Тогда $Q1=0$, а $Q2=1$. Триггер переключился в нулевое состояние.

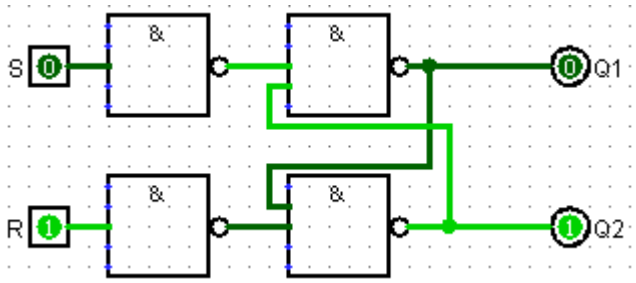


Рисунок 3.9 – Запись нуля в RS -триггер

Если после этого убрать сигнал сброса ($R=0$, $S=0$), то триггер не изменит своего состояния.

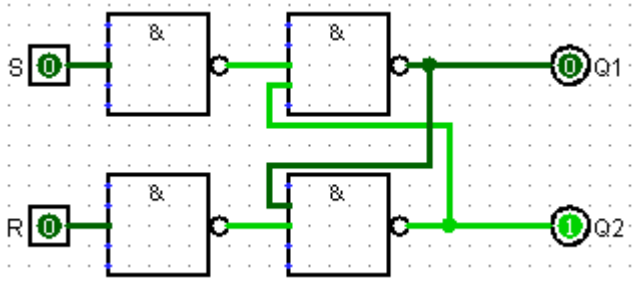


Рисунок 3.10 – Хранение нуля в RS -триггере

Для описания работы триггера используют таблицу состояний (переходов). Обозначим:

$Q(t)$ – состояние триггера до поступления управляющих сигналов (изменения на входах R и S);

$Q(t+1)$ – состояние триггера после изменения на входах R и S.

Таблица 3.6 Переходы RS-триггера

R	S	$Q(t)$	$Q(t+1)$	Режимы
0	0	0	0	Хранения информации
0	0	1	1	
0	1	0	1	Режим установки единицы: $S=1$
0	1	1	1	
1	0	0	0	Режим установки нуля: $R=1$
1	0	1	0	
1	1	0	*	Запрещённый: $R=1$, $S=1$
1	1	1	*	

На основе таблицы переходов получим граф RS -триггера.

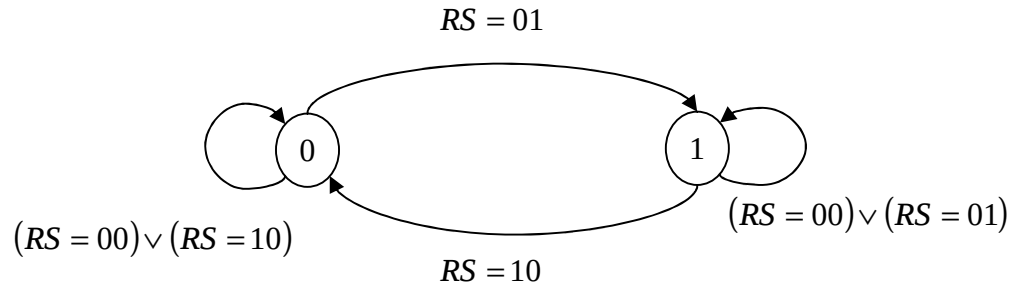


Рисунок 3.11 – Граф RS -триггера

Представим таблицу переходов RS -триггера в виде карты Карно.

$Q(t)$		0	1
$R(t)$	$S(t)$		
0	0	0	1
0	1	1	1
1	1	*	*
1	0	0	0

Рисунок 3.12 – Карта Карно RS -триггера

Так как комбинация 11 является запрещённой, можно поставить в эти клетки нули или единицы. Поставим единицы и получим выражение для функции переходов RS -триггера:

$$Q(t+1) = S(t) \vee \overline{R(t)} \cdot Q(t).$$

Применив закон Де Моргана, можно получить выражения для реализации схемы триггера в различных базисах.

Используя граф триггера, можно получить табличную форму матрицы переходов. При этом можно упростить условия переходов, используя законы алгебры логики, например: $\overline{R} \cdot \overline{S} \vee R \cdot \overline{S} = \overline{S} \cdot \left(\overline{R} \vee R\right) = \overline{S}$.

Таблица 3.7 Таблица условий переходов RS-триггера

$Q(t)$	$Q(t+1)$	Условие перехода	Упрощённое условие
0	0	$\overline{R}\cdot\overline{S}\vee R\cdot\overline{S}$	$-\overline{S}$
0	1	$\overline{R}\cdot S$	$\overline{R}\cdot S$
1	0	$R\cdot\overline{S}$	$R\cdot\overline{S}$
1	1	$\overline{R}\cdot\overline{S}\vee\overline{R}\cdot S$	$\overline{R}-$

Запись $-\overline{S}$ означает, что на вход S подаётся сигнал 0, а на вход R – безразлично.

Схема RS-триггера позволяет запоминать состояние логической схемы. При изменении входных сигналов может возникать переходный процесс (этот процесс называется «гонки сигналов»). Поэтому запоминать состояния логической схемы нужно только в определённые моменты времени, когда все переходные процессы закончены, и сигнал на выходе комбинационной схемы соответствует выполняемой ею функции. Поэтому большинство цифровых схем требуют сигнала синхронизации (тактового сигнала). Все переходные процессы в комбинационной логической схеме должны закончиться за время периода синхросигнала, подаваемого на входы триггеров. Триггеры, запоминающие входные сигналы только в момент времени, определяемый сигналом синхронизации, называются синхронными. Принципиальная схема однотактного синхронного RS-триггера приведена на рисунке.

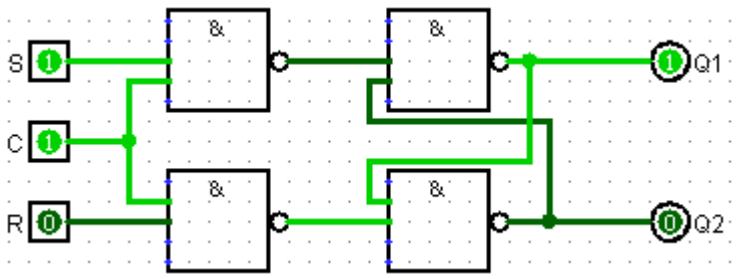


Рисунок 3.13 – Схема синхронного RS-триггера в базисе И-НЕ

На вход C подаётся синхросигнал. Состояние триггера меняется только в моменты его подачи. Без синхроимпульса триггер сохраняет своё состояние неизменным.

Таблица 3.8 Таблица переходов синхронного RS-триггера

R	S	C	$Q(t)$	$Q(t+1)$	Режимы
0	0	1	0	0	Хранения информации
0	0	1	1	1	
0	1	1	0	1	Установки единицы: $S=1$
0	1	1	1	1	
1	0	1	0	0	Установки нуля: $R=1$
1	0	1	1	0	
1	1	1	0	*	Запрещённый: $R=1,\ S=1$
1	1	1	1	*	

Для предварительной установки триггера в определённое состояние применяют установочные входы ($R0$ и $S0$ на рисунке).

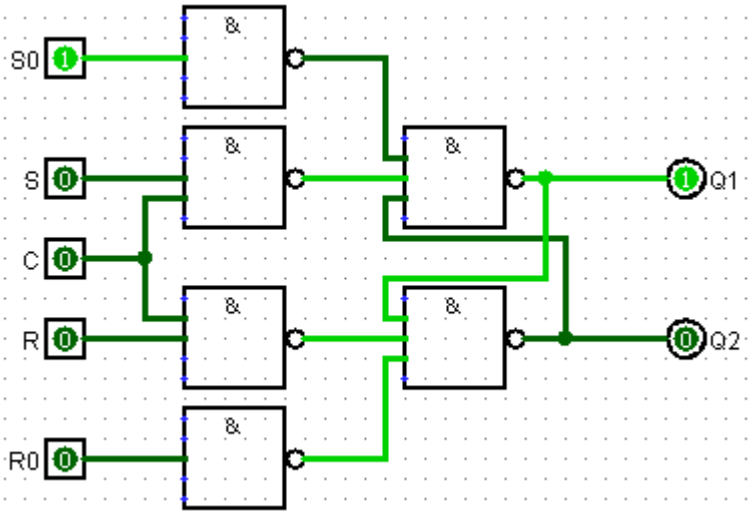


Рисунок 3.14 – Схема синхронного RS -триггера с установочными входами

Установочные входы всегда асинхронные.

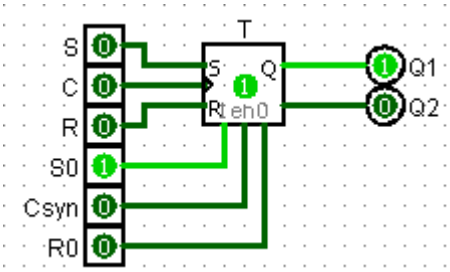


Рисунок 3.15 – Элемент RS -триггер в Logisim

Триггеру можно присвоить обозначение (в данном случае T, рисунок 3.15). Входы S и R являются рабочими входами триггера. Значение, установленное на этих входах, становится значением внутреннего состояния триггера в момент срабатывания тактирующего входа C.

На вход C (на символе устройства обозначен треугольником) подаётся тактовый сигнал. В момент, когда значение на этом входе меняется с 0 на 1 (передний фронт), значение триггера будет обновлено в соответствии входами S и R. Пока значение на этом входе остаётся 0 или 1, входы S и R не имеют эффекта.

Входы $R0$ (на символе устройства обозначен «1») и $S0$ (на символе устройства обозначен «0») являются установочными. Пока $S0$=1, состояние триггера равно на 1. Это происходит асинхронно, то есть вне зависимости от текущего значения на тактовом входе. Пока на этом входе 1, другие входы не имеют эффекта, за исключением входа $R0$, который имеет приоритет. Пока $R0$=1, значение триггера фиксировано на 0.

Вход *Csyn* (на символе устройства обозначен «eh») является синхронизатором тактового входа. Когда на этом входе 0, срабатывания тактового входа игнорируются. Текущий бит по-прежнему поступает на выход. Срабатывания тактового входа включаются, когда значение этого входа 1 или не определено. Прямой выход триггера на символе устройства обозначен Q.

3.2.2 Двухтактный RS-триггер

Во многих схемах, например, в регистрах сдвига, устойчивая работа триггера возможна только в том случае, если занесение в него новой информации осуществляется после передачи информации о его состоянии в следующий триггер. В этих случаях используют двухтактные триггеры. Кроме этого, на двухтактных триггерах строятся некоторые другие типы триггеров.

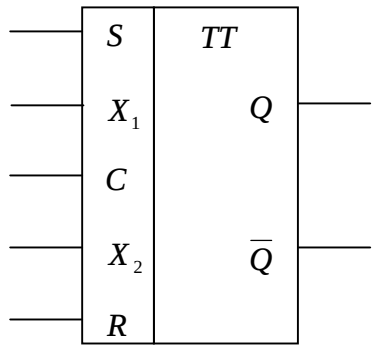


Рисунок 3.16 – Условное обозначение двухтактного триггера

Условное обозначение двухтактного триггера содержит буквы *TT* в поле обозначения триггера. Символы и функции остальных входов аналогичны однотактному.

Двухтактный *RS* -триггер содержит два однотактных триггера и инвертор в цепи синхронизации. При подаче синхросигнала (C=1) входная информация заносится на первый триггер, а во втором триггере еще сохраняется старая информация, гарантируя ее передачу на следующий триггер. После окончания активного C=1, становится активным сигнал синхронизации с выхода инвертора, который записывает входную информацию (с задержкой на время действия C=1) на второй триггер, который и является элементом хранения.

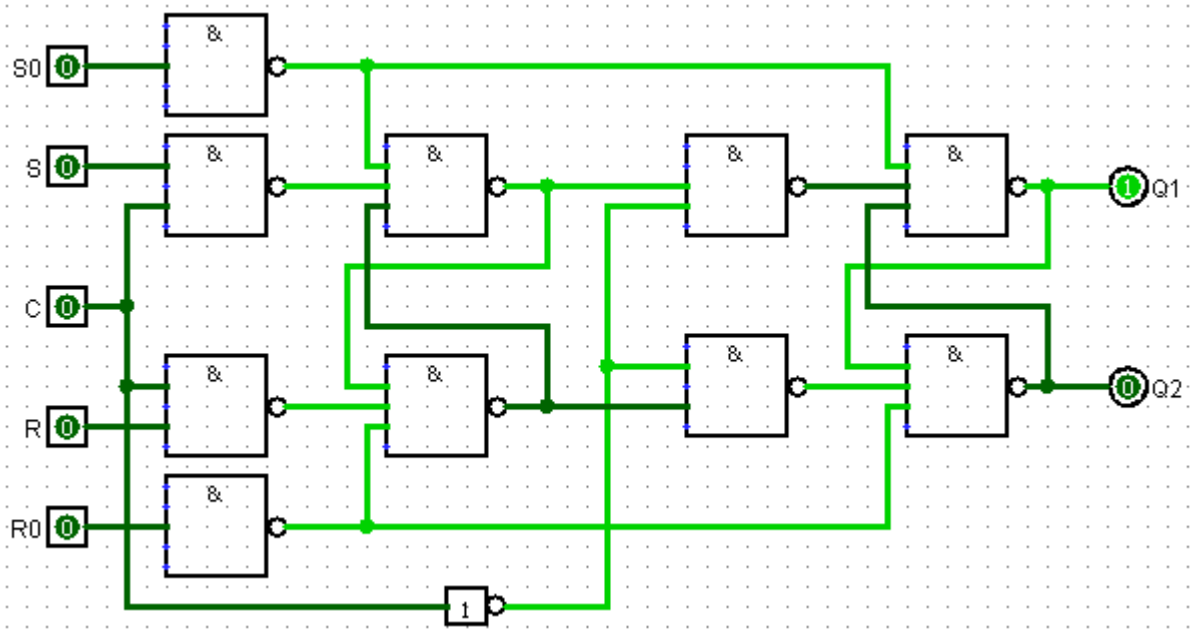


Рисунок 3.17 – Схема двухтактного *RS* -триггера с установочными входами в базисе И-НЕ

В Logisim нет отдельного элемента для двухтактного триггера, но его можно построить на двух однотактных.

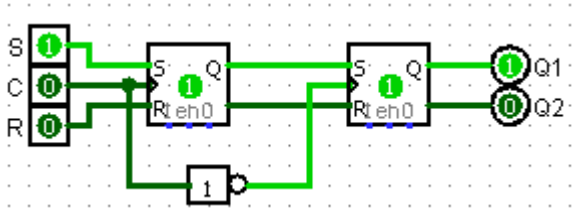


Рисунок 3.18 – Схема двухтактного RS -триггера на двух однотактных

3.2.3 D-триггер

Триггер задержки (D-триггер) можно построить на базе синхронного RS -триггера.

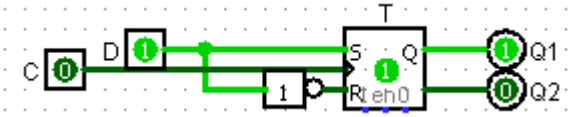


Рисунок 3.19 – D-триггер в на основе синхронного RS -триггера

Этот триггер имеет один информационный вход (D-вход). Существуют только синхронные D-триггеры. Состояние информационного входа передаётся на выход под действием синхроимпульса (вход C). Если на входе D=1, то по приходу синхроимпульса C=1 состояние выхода $Q1$=1. Если D=0, то по приходу синхроимпульса C=1 состояние выхода $Q1$=0.

Для описания работы триггера используем таблицу состояний (переходов).

Таблица 3.9 Таблица переходов D-триггера

C	D	$Q(t)$	$Q(t+1)$	Режимы
0	*	0	0	Хранения информации
0	*	1	1	
1	0	*	0	Записи информации
1	1	*	1	

Граф D- триггера:

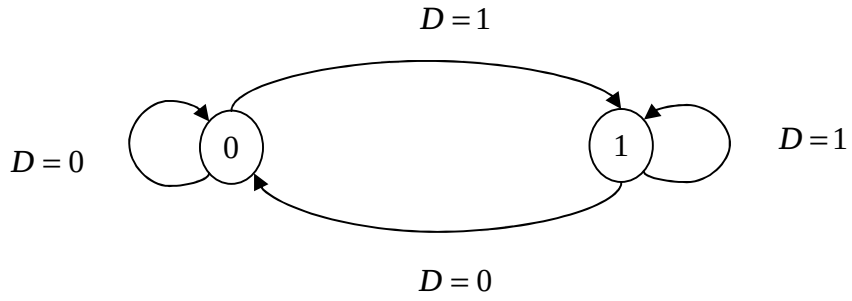


Рисунок 3.20 – Граф D-триггера

Используя граф триггера, можно получить табличную форму матрицы переходов D-триггера Для S и R.

Таблица 3.10 Табличная форма матрицы переходов D-триггера для входа S

$Q(t)$ / $D(t)$	0	1
0	0	0
1	1	1

Таблица 3.11 Табличная форма матрицы переходов D-триггера для входа R

$Q(t)$ / $D(t)$	0	1
0	*	1
1	0	0

Интерпретировав таблицу переходов как карту Карно, запишем минимизированное выражение для функций перехода D-триггера.

$S(t) = D(t)$,

$R(t) = \overline{D}(t)$.

Для описания работы триггера используем матрицу переходов.

Таблица 3.12 Таблица условий переходов D-триггера

$Q(t)$	$Q(t+1)$	Условие перехода
0	0	$\overline{D}$
0	1	D
1	0	$\overline{D}$
1	1	D

Обозначение D-триггера в программе Logisim приведено на рисунке 3.21.

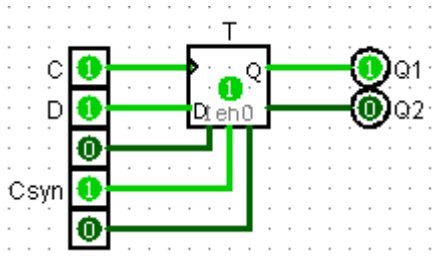


Рисунок 3.21 – Элемент D-триггер в Logisim

Этот триггер иногда называют «защёлкой» ($lock$) и используют для хранения поданного на его вход бита.

3.2.4 T-триггер

Этот триггер, называемый также счётным, бывает асинхронным и синхронным (англ. *Toggle* - переключатель). Он является простейшим счётчиком до 2.

Асинхронный T- триггер можно построить на базе D- триггера, если его вход D соединить с инверсным выходом $Q2$.

Этот T-триггер имеет один счётный информационный вход T. Если в момент t на его вход поступает сигнал логической единицы, то в момент времени $t+1$ триггер переключается в противоположное состояние. Также этот триггер можно построить на базе RS-триггера.

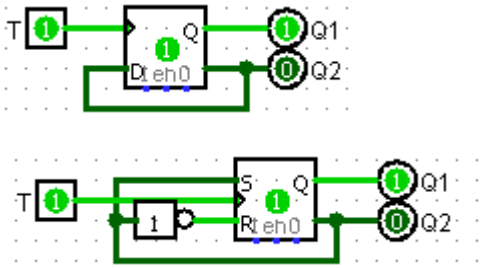


Рисунок 3.22 – T -триггер на основе D -триггера и RS -триггера

Этот триггер предназначен для подсчёта входных сигналов в двоичной системе счисления.

Для описания работы триггера используем таблицу состояний (переходов).

Таблица 3.13 Таблица переходов Т-триггера

T	$Q(t)$	$Q(t+1)$	Режимы
0	0	0	Хранения информации
1	1	1	
0	0	1	Записи информации
1	1	0	

Граф T -триггера:

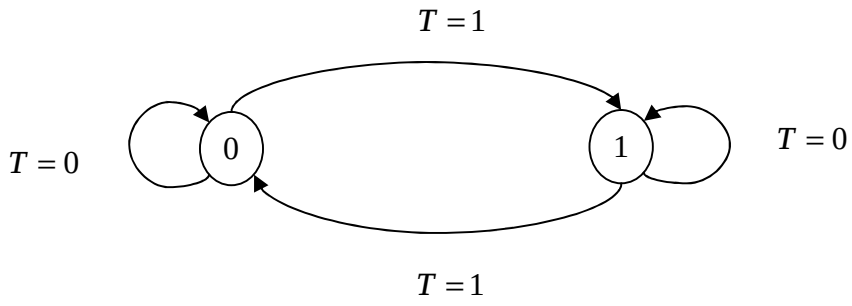


Рисунок 3.23 – Граф T -триггера

Используя граф триггера, можно получить табличную форму матрицы переходов T -триггера:

$R(t)$ \ $Q(t)$	0	1
0	0	1
1	1	0

Рисунок 3.24 – Карта Карно D-триггера

Интерпретировав таблицу переходов как карту Карно, запишем минимизированное выражение для функции переходов D-триггера.

$$Q(t+1)=\overline{T(t)}\cdot Q(t)\vee T(t)\cdot\overline{Q(t)}.$$

Для описания работы триггера используем матрицу переходов.

Таблица 3.14 Таблица условий переходов Т-триггера

$Q(t)$	$Q(t+1)$	Условие перехода
0	0	$\overline{T}$
0	1	T
1	0	T
1	1	$\overline{T}$

Синхронный *T*-триггер, при единице на входе *T*, по каждому такту на входе С изменяет своё логическое состояние на противоположное, и не изменяет выходное состояние при нуле на входе *T*. Этот триггер можно построить на *JK*-триггере, на двухступенчатом *D*-триггере, а также на двух одноступенчатых *D*-триггерах и инверторе.

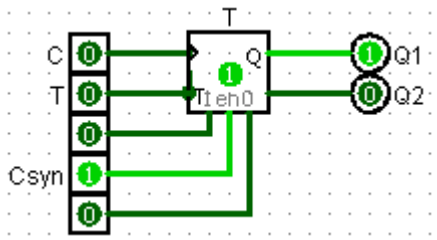


Рисунок 3.25 – Элемент T-триггер в Logisim

В программе Logisim представлен синхронный T-триггер. Когда его тактовый вход C срабатывает, значение, хранящееся в триггере, меняется или остаётся прежним в зависимости от того, какое значение на входе T, 1 или 0. Обозначение T-триггера в программе Logisim приведено на рисунке 3.25.

3.2.5 JK-триггер

Универсальный триггер (JK-триггер) применяется для построения любых типов триггеров. Он является двухтактным и строится на основе RS-триггера, от которого отличается тем, что комбинация $SR=11$ не является для него запрещённой. Такой триггер имеет информационные входы J и K, которые по своему влиянию аналогичны входам S и R синхронного RS-триггера.

Рассмотрим его работу. При $JK=10$ триггер по тактовому импульсу устанавливается в состояние $Q=1$. Если $JK=01$ - переключается в состояние $Q=0$. При $JK=00$ триггер хранит ранее принятую информацию. При комбинации $JK=11$, запрещённой для RS-триггера, переводит JK-триггер в противоположное состояние.

Таблица 3.15 Таблица переходов JK-триггера

K	J	C	$Q(t)$	$Q(t+1)$	Режимы
0	0	1	0	0	Хранения информации
0	0	1	1	1	
0	1	1	0	1	Установки единицы: $J=1$
0	1	1	1	1	
1	0	1	0	0	Установки нуля: $K=1$
1	0	1	1	0	
1	1	1	0	1	Инверсии:
1	1	1	1	0	$R=1,\ S=1$

Граф JK-триггера:

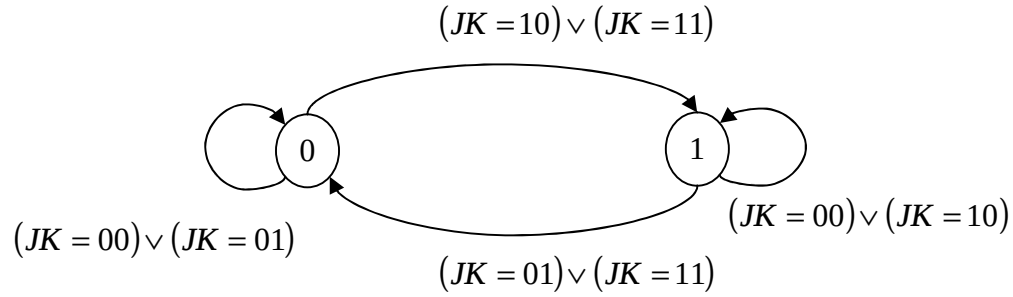


Рисунок 3.26 – Граф *JK* -триггера

Используя граф триггера, можно получить табличную форму матрицы переходов. При этом можно упростить условия переходов, используя законы алгебры логики.

Таблица 3.16 Таблица условий переходов JK-триггера

$Q(t)$	$Q(t+1)$	Условие перехода	Упрощённое условие
0	0	$\overline{J}\cdot\overline{K}\vee\overline{J}\cdot K$	$\overline{J}$ –
0	1	$J\cdot\overline{K}\vee J\cdot K$	J –
1	0	$\overline{J}\cdot K\vee J\cdot K$	– K
1	1	$\overline{J}\cdot\overline{K}\vee J\cdot\overline{K}$	– $\overline{K}$

Запись $\overline{J}$ – означает, что на вход J подаётся сигнал 0, а на вход K – безразлично.

Представим таблицу переходов *JK* -триггера в виде карты Карно:

$Q(t)$		0	1
$J(t)$	$K(t)$		
0	0	0	1
0	1	0	0
1	1	1	0
1	0	1	1

Рисунок 3.27 – Карта Карно *JK* -триггера

Получим выражение для функции переходов *JK* -триггера:

$Q(t+1)=J(t)\cdot\overline{Q(t)}\vee\overline{K(t)}\cdot Q(t)$.

Схематически JK - триггер строится на базе двухтактного RS -триггера за счёт обратной связи.

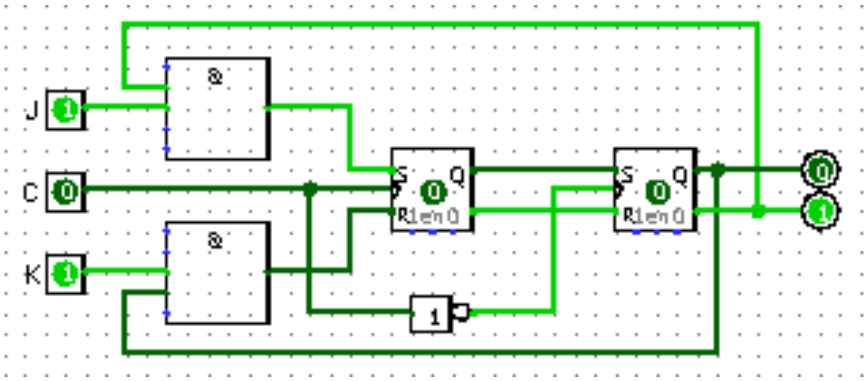


Рисунок 3.28 – Схема JK -триггера

Обозначение в программе Logisim:

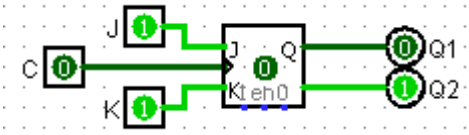


Рисунок 3.29 – Элемент JK -триггер в Logisim

Кроме рассмотренных типов триггеров, существуют и другие: S, R, E - триггеры. Они отличаются от базового RS реакцией на комбинацию 11. При подаче такой комбинации сигналов на входы S и R, S -триггер переходит в состояние Q=1, R -триггер переходит в состояние Q=0, а E -триггер сохраняет своё предыдущее состояние.

3.3 Синтез функций активации и выходов цифрового автомата с использованием RS-триггеров

Кодирование внутренних состояний абстрактного конечного автомата заключается в установлении соответствия между внутренними состояниями автомата и комбинациями состояний элементарных комбинационных схем (автоматов Мура), входящих в синтезируемый автомат с памятью (автомат Мили). В общем случае кодирование может быть произвольным, но на практике выбор того или иного варианта кодирования влияет на сложность функций активации его элементарных автоматов.

Функцией активации (возбуждения) элементарного конечного автомата называется логическое выражение, описывающее зависимость входных сигналов элементарного автомата от состояний элементарного автомата и входных сигналов синтезируемого конечного автомата.

Предполагается, что для всех типов триггеров, используемых при синтезе, функции активации определяются функциями внешних переходов этих автоматов.

Функция внешних переходов элементарного конечного автомата – это логическое выражение, определяющее зависимость состояния элементарного автомата в момент $t+1$ от комбинаций состояний всех элементарных автоматов и входных сигналов синтезируемого конечного автомата в момент времени t.

Конечный автомат может быть задан несколькими способами. Пусть в данном случае он задан таблицей состояний (переходов), а также кодами этих состояний и входных воздействий.

Таблица 3.17 Состояния цифрового автомата

q / x	1	2	3
1	2	3	1
2	2	3	2
3	1	1	2
4	3	1	1

В развёрнутом виде эта таблица выглядит так:

Таблица 3.18 Развёрнутые состояния цифрового автомата

Внутренние состояния q / Внешние воздействия x	q_1	q_2	q_3
x_1	q_2	q_3	q_1
x_2	q_2	q_3	q_2
x_3	q_1	q_1	q_2
x_4	q_3	q_1	q_1

Вначале необходимо определить число N триггеров синтезируемого конечного автомата по формуле:

$N = [\log_2 m]_{ббц}$,

где ббц - ближайшее большее целое,

m – число внутренних переменных абстрактного конечного автомата,

N – число триггеров.

В рассматриваемом случае имеется три внутренних переменных q:

$N = [\log_2 3]_{ббц} = 2$

Обозначим триггеры Q_1 и Q_2.

Теперь установим соответствия между внутренними состояниями конечного автомата и комбинациями состояний триггеров, входящих в синтезируемый автомат.

Определим выходные состояния элементарных автоматов (триггеров Q_1 и Q_2) с учётом того, что у каждого из них имеется прямой и инверсный вход:

$q_1 = \overline{Q_1} \cdot \overline{Q_2} = 00$,

$q_2 = \overline{Q_1} \cdot Q_2 = 01$,

$q_3 = Q_1 \cdot Q_2 = 11$,

$q_4 = Q_1 \cdot \overline{Q_2} = 10$.

В процессе кодирования получилось четыре состояния. Но в заданной выше таблице используется только три внутренних состояния q синтезируемого автомата. Следовательно, из четвёртого состояния автомат будет переходить в неопределённое.

Найдём число входных переменных синтезируемого автомата: $N = [\log_2 4]_{ббц} = 2$.

Обозначим их z_1 и z_2. Закодируем эти состояния:

$x_1 = \overline{z_1} \cdot \overline{z_2} = 00$,

$x_2 = \overline{z_1} \cdot z_2 = 01$,

$x_3 = z_1 \cdot z_2 = 11$,

$x_4 = z_1 \cdot \overline{z_2} = 10$.

Заменим в заданной выше таблице переходов синтезируемого автомата обозначения состояний q и входных воздействий x их кодами. Например, $q_1 = 00$, $x_2 = 01$ и так далее.

Таблица 3.19 Коды состояний цифрового автомата

x \ q	00	01	11	10
00	01	11	00	–
01	01	11	01	–
11	00	00	01	–
10	11	00	00	–

Разделим эту таблицу на две. В первой оставим только те цифры внутренних состояний, которые соответствуют триггеру Q_1 (левые), а во второй – те, которые соответствуют триггеру Q_2 (правые).

Для триггера Q_1:

Таблица 3.20 Коды внутренних состояний для триггера Q_1

x \ q	00	01	11	10
00	0	1	0	–
01	0	1	0	–
11	0	0	0	–
10	1	0	0	–

Для триггера Q_2:

Таблица 3.21 Коды внутренних состояний для триггера Q_2

x \ q	00	01	11	10
00	1	1	0	–
01	1	1	1	–
11	0	0	1	–
10	1	0	0	–

Зададимся типом триггера, который будет использован при синтезе конечного автомата. Пусть это RS-триггер. Рассмотрим приведённую выше закодированную таблицу переходов, предназначенную для определения функций внешних переходов входящих в автомат триггеров. Если в её клетках проставить не состояния элементарного автомата, а условия, вызывающие переход в эти состояния, то по такой таблице можно определить функции активации соответствующего триггера. Условия, вызывающие переход, берутся из табличной формы матрицы переходов выбранного типа триггера. В рассматриваемом случае следует воспользоваться соответствующей таблицей RS-триггера:

Таблица 3.22 Таблица условий переходов RS-триггера

$Q(t)$	$Q(t+1)$	Упрощённое условие
0	0	$-\overline{S}$
0	1	$\overline{R}\cdot S$
1	0	$R\cdot\overline{S}$
1	1	$\overline{R}-$

Состояние $Q(t)$ отмечено в строке «Внутренние состояния», а состояние $Q(t+1)$ - в клетках таблицы.

Для триггера Q_1:

Таблица 3.23 Таблица условий переходов состояний для Q_1

x \ q	00	01	11	10
00	$-\overline{S_1}$	$\overline{R_1}\cdot S_1$	$R_1\cdot\overline{S_1}$	–
01	$-\overline{S_1}$	$\overline{R_1}\cdot S_1$	$R_1\cdot\overline{S_1}$	–
11	$-\overline{S_1}$	$-\overline{S_1}$	$R_1\cdot\overline{S_1}$	–
10	$\overline{R_1}\cdot S_1$	$-\overline{S_1}$	$R_1\cdot\overline{S_1}$	–

Найдём выражение функции активации для входа R_1 триггера Q_1. Для этого поставим в эту таблицу в выражения состояний $Q(t+1)$ значения $R_1 = 1$ и $S_1 = 0$:

$-\overline{S_1} = -\cdot\bar{0} = -\cdot 1 = -$ (неопределённость),

$\overline{R_1}\cdot S_1 = \bar{1}\cdot 0 = 0$,

$R_1 \cdot \overline{S_1} = 1\cdot\bar{0} = 1\cdot 1 = 1$.

Результат запишем в виде карты Карно.

q / x	00	01	11	10
00	–	0	1	–
01	–	0	1	–
11	–	–	1	–
10	0	–	1	–

Рисунок 3.30 – Карта Карно для функции активации входа R_1

Получим логическое выражение функции активации для входа R_1 триггера Q_1:

$R_1 = Q_1$.

Совершим те же операции S_1 триггера Q_1, положив $R_1 = 0$ и $S_1 = 1$.

$-\overline{S_1} = -\cdot\bar{1} = 0$,

$\overline{R_1}\cdot S_1 = \bar{0}\cdot 1 = 1$,

$R_1 \cdot \overline{S_1} = 0\cdot\bar{1} = 0$.

Используем получившуюся таблицу как карту Карно.

q / x	00	01	11	10
00	0	1	0	–
01	0	1	0	–
11	0	0	0	–
10	1	0	0	–

Рисунок 3.31 – Карта Карно для функции активации входа S_1

Запишем логическое выражение для активации входа S_1.

$S_1 = \overline{z_1}\,\overline{Q_1}Q_2 \vee z_1\overline{z_2}\,\overline{Q_2}$.

Для триггера Q_2 используем правые части кодов исходной таблицы, подставив условия перехода для RS -триггера:

Таблица 3.24 Условия переходов состояний для Q_2

q / x	00	01	11	10
00	$\overline{R}\cdot S$	$\overline{R}-$	$R\cdot\overline{S}$	–
01	$\overline{R}\cdot S$	$\overline{R}-$	$\overline{R}-$	–
11	$-\overline{S}$	$R\cdot\overline{S}$	$\overline{R}-$	–
10	$\overline{R}\cdot S$	$R\cdot\overline{S}$	$R\cdot\overline{S}$	–

Рассмотрим карту Карно для R_2 ($R_2 = 1$ и $S_2 = 0$).

q / x	00	01	11	10
00	0	0	1	–
01	0	0	0	–
11	–	1	0	–
10	0	1	1	–

Рисунок 3.32 – Карта Карно для функции активации входа R_2

Запишем логическое выражение для активации входа R_2.

$R_2 = \overline{z_2}Q_1 \vee z_1\overline{Q_1}Q_2$.

Рассмотрим карту Карно для S_2 ($R_2 = 0$ и $S_2 = 1$).

q / x	00	01	11	10
00	1	–	0	–
01	1	–	–	–
11	0	0	–	–
10	1	0	0	–

Рисунок 3.33 – Карта Карно для функции активации входа S_2

Запишем логическое выражение для активации входа S_2.

$S_2 = \overline{z_1}\,\overline{Q_1} \vee \overline{z_2}\,\overline{Q_2}$.

Итак, мы имеем выражения для активации входов триггеров Q_1 и Q_2:

$R_1 = Q_1$,

$S_1 = \overline{z_1}\,\overline{Q_1} Q_2 \vee z_1 \overline{z_2}\,\overline{Q_2}$,

$R_2 = \overline{z_2} Q_1 \vee z_1 \overline{Q_1} Q_2$

$S_2 = \overline{z_1}\,\overline{Q_1} \vee \overline{z_2}\,\overline{Q_2}$.

Теперь необходимо построить эту схему в Logisim и проверить её работу. Так как триггеры в Logisim синхронные, добавим вход синхронизации (вход C внизу на схеме).

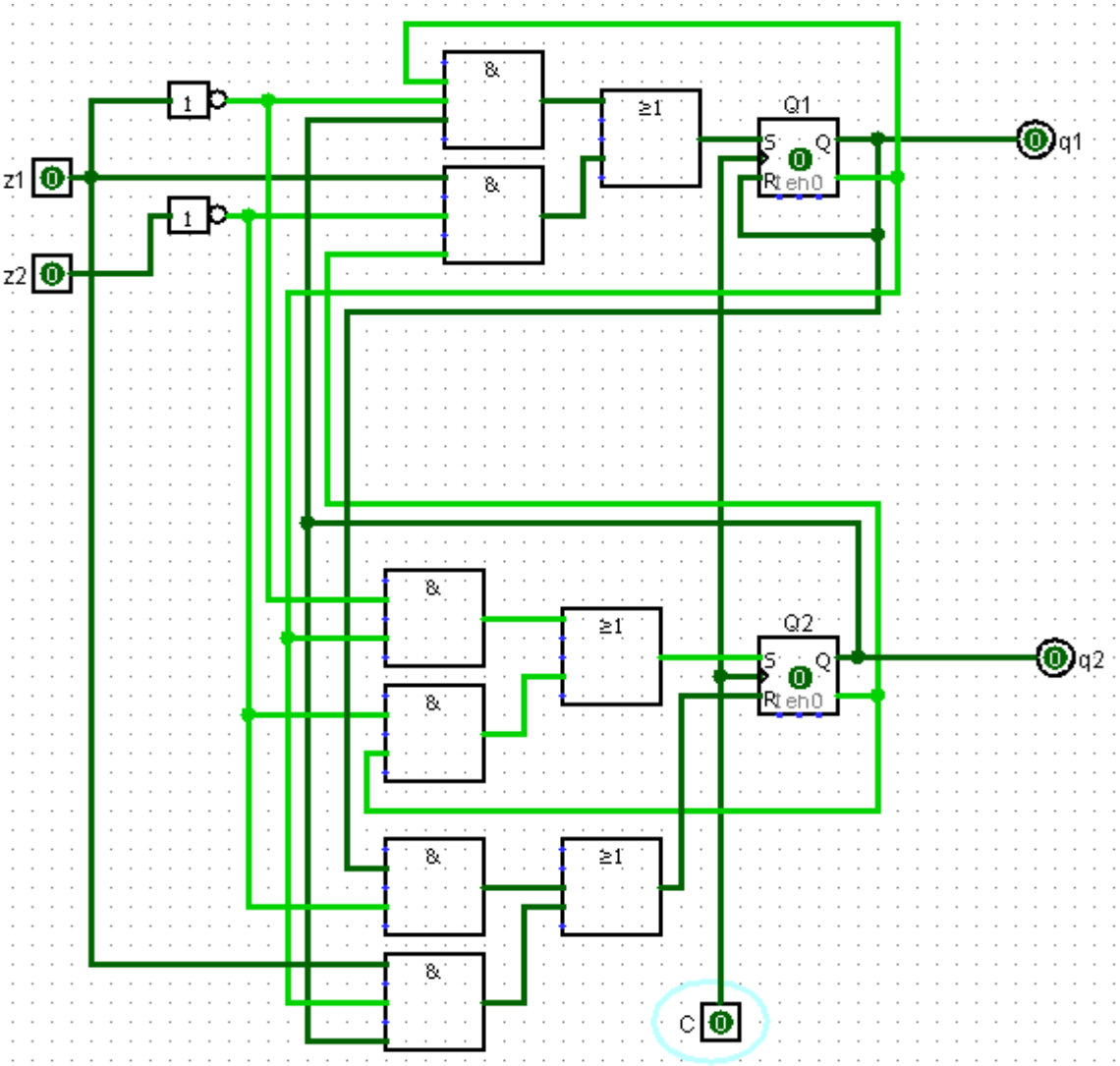


Рисунок 3.34 – Схема проектируемого цифрового автомата с функциями активации

Состояние триггеров будет меняться только при подаче единицы на вход синхронизации. Исходное состояние триггера можно задавать щелчком мыши по его изображению. Для проверки используем исходную таблицу состояний (переходов) и коды состояний.

Согласно таблице переходов, если $Q_1Q_2 = 00$, то при воздействии $z_1z_2 = 00$ автомат должен перейти в состояние $Q_1Q_2 = 01$ (логический ноль). Установив триггеры в состояние $Q_1Q_2 = 00$ и входы в состояние $z_1z_2 = 00$ (рисунок выше), подадим сигнал синхронизации $C = 1$. Получим переход автомата в состояние $Q_1Q_2 = 01$:

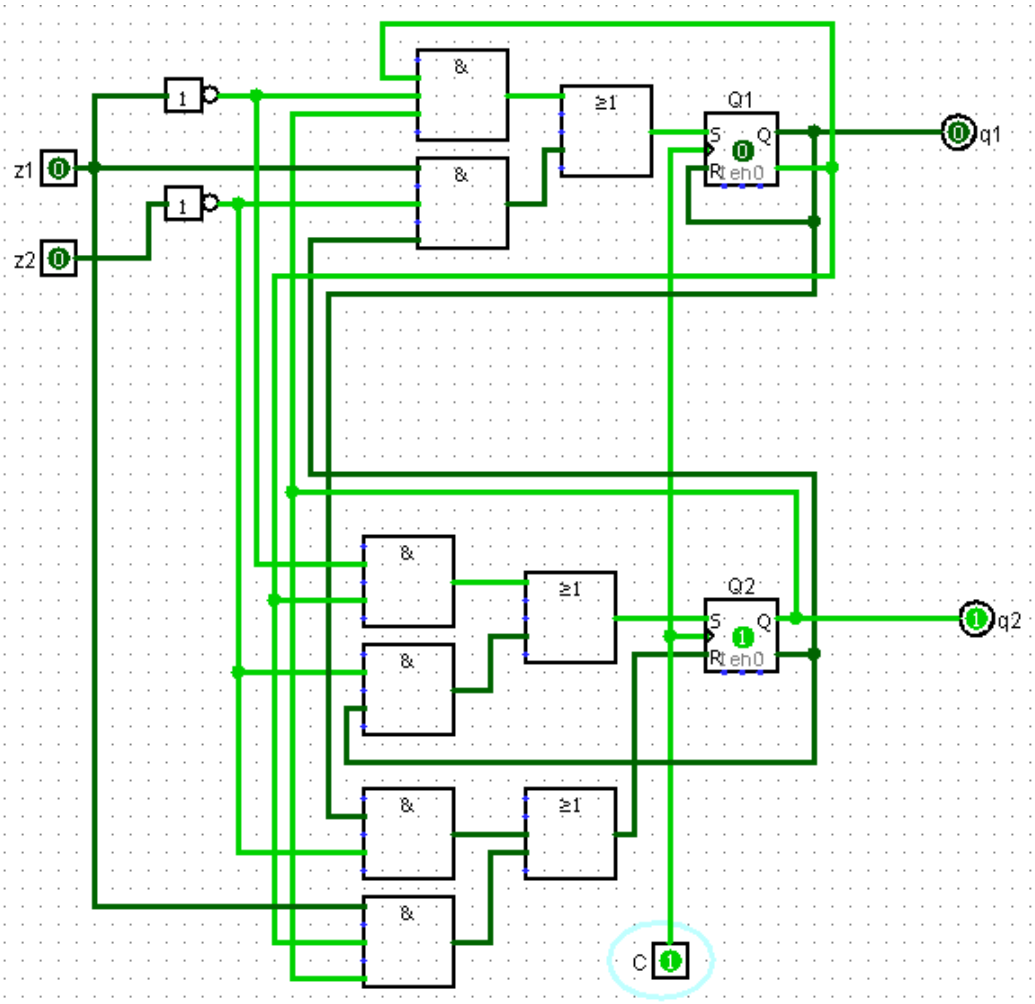


Рисунок 3.35 – Переход автомата в состояние $Q_1Q_2 = 01$ из $Q_1Q_2 = 00$ при подаче $z_1z_2 = 00$ и $C = 1$

Аналогично проверяются другие состояния.

Теперь рассмотрим построение функций выхода конечного автомата. В качестве исходных данных используется заранее заданная таблица выходов конечного автомата. Пусть для автомата определены четыре выходных состояния y_1, y_2, y_3, y_4., а таблица выходов имеет следующий вид:

Таблица 3.25 Выходные состояния автомата

y / x	1	2	3
1	1	2	1
2	1	2	3
3	3	1	4
4	1	4	2

В развёрнутом виде эта таблица выглядит так:

Таблица 3.26 Развёрнутые выходные состояния автомата

Выходные состояния y / Входные воздействия x	q_1	q_2	q_3
x_1	y_1	y_2	y_1
x_2	y_1	y_2	y_3
x_3	y_3	y_1	y_4
x_4	y_1	y_4	y_2

Количество переменных для заданного числа выходных состояний:

$N = [\log_2 k]_{\text{ббц}}$,

где ббц - ближайшее большее целое,

k – число выходных состояний абстрактного конечного автомата,

N – число переменных.

В рассматриваемом случае имеется четыре выходных состояния y:

$N = [\log_2 4]_{\text{ббц}} = 2$.

Таким образом, достаточно двух переменных (и двух выходов) для описания четырёх состояний Обозначим их Q_1 и Q_2. Закодируем эти состояния:

$y_1 = \overline{Q_1} \cdot \overline{Q_2} = 00$,

$y_2 = \overline{Q_1} \cdot Q_2 = 01$,

$y_3 = Q_1 \cdot Q_2 = 11$,

$y_4 = Q_1 \cdot \overline{Q_2} = 10$.

Перепишем таблицу выхода автомата в закодированном виде:

Таблица 3.27 Коды состояний выходов автомата

$Q_1\ Q_2$ / $z_1\ z_2$	00	01	11	10
00	00	01	00	–
01	00	01	11	–
11	11	00	10	–
10	00	10	01	–

Разделим эту таблицу на две, для Q_1 и Q_2.

Для Q_1:

$Q_1\ Q_2$ / $z_1\ z_2$	00	01	11	10
00	0	0	0	–
01	0	0	1	–
11	1	0	1	–
10	0	1	0	–

Рисунок 3.36 – Карта Карно для функции выхода Q_1

Используем таблицу для Q_1 как карту Карно:

$y_1 = Q_1 z_2 \vee \overline{Q}_2 z_1 z_2 \vee \overline{Q}_1 Q_2 z_1 \overline{z}_2$.

Для Q_2:

$Q_1\ Q_2$ / $z_1\ z_2$	00	01	11	10
00	0	1	0	–
01	0	1	1	–
11	1	0	0	–
10	0	0	1	–

Рисунок 3.37 – Карта Карно для функции выхода Q_2

$$y_2 = \overline{Q}_1 Q_2 \bar{z}_1 \vee Q_2 \bar{z}_1 z_2 \vee \overline{Q}_2 z_1 z_2 \vee Q_1 z_1 \bar{z}_2 .$$

Итак, получены формулы для логических функций активации триггеров и выходов конечного автомата:

Функции активации:

$$R_1 = Q_1,$$
$$S_1 = \overline{z_1}\,\overline{Q_1} Q_2 \vee z_1 \overline{z_2}\,\overline{Q_2},$$
$$R_2 = \overline{z_2} Q_1 \vee z_1 \overline{Q_1} Q_2,$$
$$S_2 = \overline{z_1}\,\overline{Q_1} \vee \overline{z_2}\,\overline{Q_2} .$$

Функции выхода:

$$y_1 = Q_1 z_2 \vee \overline{Q}_2 z_1 z_2 \vee \overline{Q}_1 Q_2 z_1 \bar{z}_2,$$
$$y_2 = \overline{Q}_1 Q_2 \bar{z}_1 \vee Q_2 \bar{z}_1 z_2 \vee \overline{Q}_2 z_1 z_2 \vee Q_1 z_1 \bar{z}_2 .$$

Теперь необходимо построить эту схему в Logisim и проверить её работу.

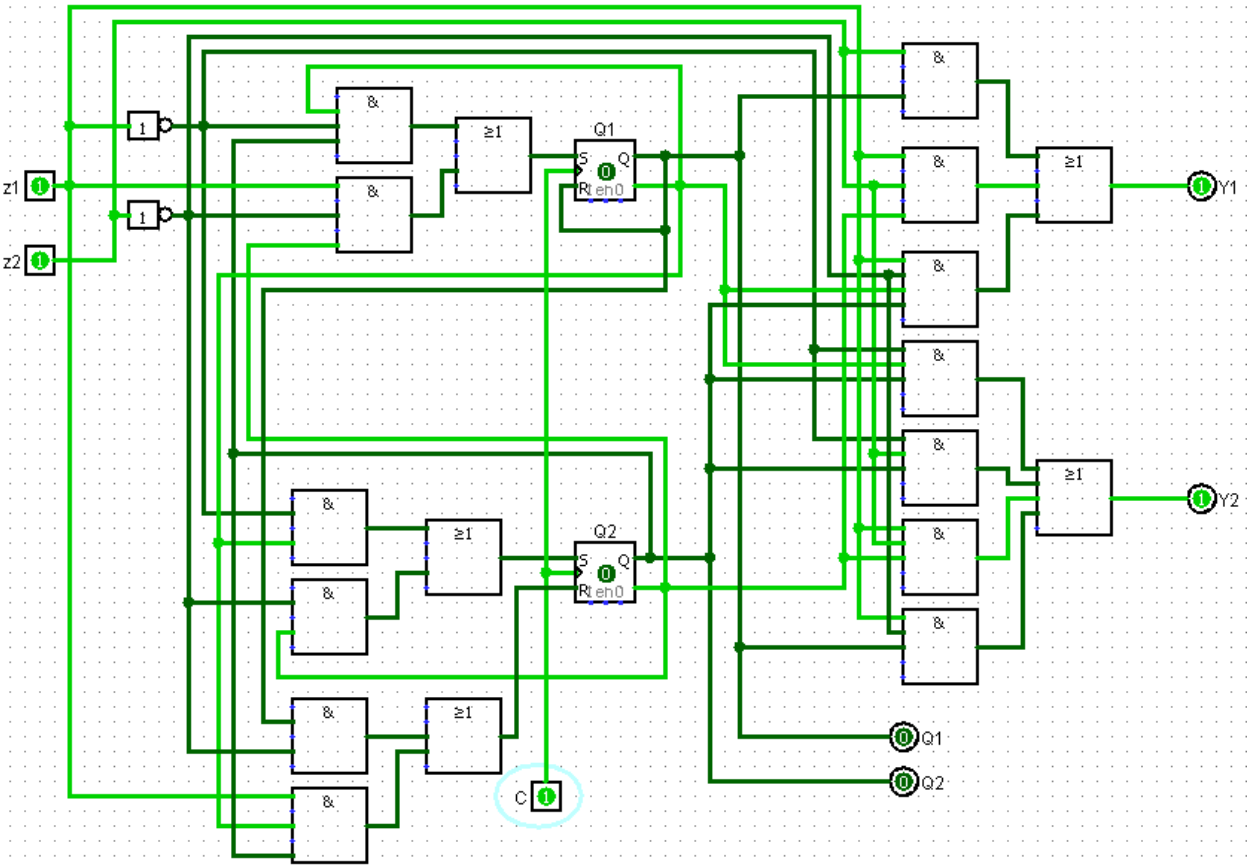


Рисунок 3.38 – Проектируемый цифровой автомат с функциями активации и выходов

Например, пусть триггеры находятся в состоянии q_1, то есть $Q_1 = 0$ и $Q_2 = 0$. При подаче на вход схемы состояния x_3, то есть $z_1 = 1$ и $z_2 = 1$, триггеры перейдут в состояние q_1 (таблица переходов автомата). Найдём столбец с этим состоянием в таблице выходов. На его пересечении со строкой входного воздействия x_3 увидим, что автомат перейдёт в состояние y_1, то есть $Y_1 = 1$ и $Y_2 = 1$ (таблица выходов автомата).

3.4 Синтез конечного автомата с использованием Т-триггера

Рассмотрим синтез конечного автомата на основе *T*- триггеров.

Зададимся таблицей внутренних состояний автомата.

Таблица 3.28 Внутренние состояния автомата

q / x	1	2	3
1	1	1	2
2	2	1	3
3	2	1	3
4	2	1	3

В развёрнутом виде эта таблица выглядит так:

Таблица 3.29 Развёрнутые внутренние состояния автомата

Внутренние состояния q / Внешние воздействия x	q_1	q_2	q_3
x_1	q_1	q_1	q_2
x_2	q_2	q_1	q_3
x_3	q_2	q_1	q_3
x_4	q_2	q_1	q_3

Число триггеров синтезируемого конечного автомата:

$N = [\log_2 3]_{\text{ббц}} = 2$

Обозначим триггеры Q_1 и Q_2.

Теперь установим соответствия между внутренними состояниями конечного автомата и комбинациями состояний триггеров, входящих в синтезируемый автомат. Определим выходные состояния элементарных автоматов (триггеров Q_1 и Q_2):

$q_1 = \overline{Q_1} \cdot \overline{Q_2} = 00$,

$q_2 = \overline{Q_1} \cdot Q_2 = 01$,

$q_3 = Q_1 \cdot Q_2 = 11$,

$q_4 = Q_1 \cdot \overline{Q_2} = 10$.

В процессе кодирования получилось четыре состояния. Но в заданной выше таблице используется только три внутренних состояния q синтезируемого автомата. Следовательно, из четвёртого состояния автомат будет переходить в неопределённое.

Найдём число входных переменных синтезируемого автомата: $N = [\log_2 4]_{ббц} = 2$.

Обозначим их z_1 и z_2. Закодируем эти состояния:

$x_1 = \overline{z_1} \cdot \overline{z_2} = 00$,

$x_2 = \overline{z_1} \cdot z_2 = 01$,

$x_3 = z_1 \cdot z_2 = 11$,

$x_4 = z_1 \cdot \overline{z_2} = 10$.

Заменим в заданной выше таблице переходов синтезируемого автомата обозначения состояний q и входных воздействий x их кодами.

Таблица 3.30 Коды внутренних состояний автомата

Q_1Q_2 / z_1z_2	00	01	11	10
00	00	00	01	–
01	01	00	11	–
11	01	00	11	–
10	01	00	11	–

Разделим эту таблицу на две. В первой оставим только те цифры внутренних состояний, которые соответствуют триггеру Q_1 (левые), а во второй – те, которые соответствуют триггеру Q_2 (правые).

Для триггера Q_1:

Таблица 3.31 Коды внутренних состояний для триггера Q_1

z_1z_2 \ Q_1Q_2	00	01	11	10
00	0	0	0	–
01	0	0	1	–
11	0	0	1	–
10	0	0	1	–

Для триггера Q_2:

Таблица 3.32 Коды внутренних состояний для триггера Q_2

x \ Q_1Q_2	00	01	11	10
00	0	0	1	–
01	1	0	1	–
11	1	0	1	–
10	1	0	1	–

Поставим в клетках таблицы выше условия, вызывающие переход T-триггера в указанное в клетке состояние. Для этого воспользуемся таблицей переходов T-триггера.

Таблица 3.33 Условия переходов Т-триггера

$Q(t)$	$Q(t+1)$	Условие перехода
0	0	$\overline{T}$
0	1	T
1	0	T
1	1	$\overline{T}$

Для триггера Q_1:

Таблица 3.34 Условия переходов состояний для Q_1

z_1z_2 \ Q_1Q_2	00	01	11	10
00	$\overline{T}$	$\overline{T}$	T	–
01	$\overline{T}$	$\overline{T}$	$\overline{T}$	–
11	$\overline{T}$	$\overline{T}$	$\overline{T}$	–
10	$\overline{T}$	$\overline{T}$	$\overline{T}$	–

Найдём выражение функции активации для входа T_1 триггера Q_1. Для этого поставим в эту таблицу в выражения состояний $Q(t+1)$ значения $T_1 = 1$:

$\overline{T_1} = \overline{1} = 0$,

$T_1 = 1$.

Получим для триггера Q_1:

z_1z_2 \ Q_1Q_2	00	01	11	10
00	0	0	1	–
01	0	0	0	–
11	0	0	0	–
10	0	0	0	–

Рисунок 3.39 – Карта Карно для минимизации функции T_1

Используя эту таблицу как карту Карно, получим логическое выражение функции активации для входа T_1 триггера Q_1:

$T_1 = \bar{z}_1\bar{z}_2Q_1$.

Для триггера Q_2 используем правые части кодов исходной таблицы, подставив условия перехода для T - триггера.

Таблица 3.35 Условия переходов состояний для Q_2

z_1z_2 \ Q_1Q_2	00	01	11	10
00	$\overline{T}$	T	$\overline{T}$	–
01	T	T	$\overline{T}$	–
11	T	T	$\overline{T}$	–
10	T	T	$\overline{T}$	–

Найдём выражение функции активации для входа T_2 триггера Q_2. Для этого поставим в эту таблицу в выражения состояний $Q(t+1)$ значения $T_2 = 1$:

$\overline{T_2} = \overline{1} = 0$,

$T_2 = 1$.

z_1z_2 \ Q_1Q_2	00	01	11	10
00	0	1	0	–
01	1	1	0	–
11	1	1	0	–
10	1	1	0	–

Рисунок 3.40 – Карта Карно для минимизации функции T_2

Используя эту таблицу как карту Карно, получим логическое выражение функции активации для входа T_2 триггера Q_2:

$T_2 = \overline{Q_1}Q_2 \vee z_2\overline{Q_1} \vee z_1\overline{Q_1}$.

Итак, мы имеем выражения для активации входов триггеров Q_1 и Q_2:

$T_1 = \bar{z}_1\bar{z}_2Q_1$,

$T_2 = \overline{Q_1}Q_2 \vee z_2\overline{Q_1} \vee z_1\overline{Q_1}$.

В качестве исходных данных для получения функций выхода используется заранее заданная таблица выходов конечного автомата. Пусть для автомата определены четыре выходных состояния y_1, y_2, y_3, y_4., а таблица выходов имеет следующий вид:

Таблица 3.36 Выходные состояния автомата

x \ y	1	2	3
1	3	2	1
2	1	3	3
3	3	1	4
4	2	1	1

Количество переменных для заданного числа выходных состояний:
$N = [\log_2 4]_{\text{ббц}} = 2$.

Таким образом, достаточно двух переменных (и двух выходов) для описания четырёх состояний Обозначим их Q_1 и Q_2. Закодируем эти состояния:

$y_1 = \overline{Q_1} \cdot \overline{Q_2} = 00$,

$y_2 = \overline{Q_1} \cdot Q_2 = 01$,

$y_3 = Q_1 \cdot Q_2 = 11$,

$y_4 = Q_1 \cdot \overline{Q_2} = 10$.

Перепишем таблицу выхода автомата в закодированном виде.

Таблица 3.37 Коды выходных состояний автомата

$z_1\ z_2$ \ $Q_1\ Q_2$	00	01	11	10
00	11	01	00	–
01	00	11	11	–
11	11	00	10	–
10	01	00	00	–

Разделим эту таблицу на две, для Q_1 и Q_2.

$z_1\ z_2$ \ $Q_1\ Q_2$	00	01	11	10
00	1	0	0	–
01	0	1	1	–
11	1	0	1	–
10	0	0	0	–

Рисунок 3.41 – Карта Карно для минимизации функции y_1

Используем таблицу для Q_1 как карту Карно.

$y_1 = \bar{z}_1\bar{z}_2\overline{Q}_2 \vee \bar{z}_1 z_2 Q_2 \vee z_1 z_2 Q_1 \vee z_1 z_2 \overline{Q}_2$.

$z_1\ z_2$ \ $Q_1\ Q_2$	00	01	11	10
00	1	1	0	–
01	0	1	1	–
11	1	0	0	–
10	1	0	0	–

Рисунок 3.42 – Карта Карно для минимизации функции y_2

$y_2 = \bar{z}_1\bar{z}_2\overline{Q}_1 \vee \bar{z}_1 z_2 Q_2 \vee z_1\overline{Q}_1\overline{Q}_2$.

Итак, получены формулы для логических функций активации триггеров и выходов конечного автомата:

Функции активации:

$T_1 = \bar{z}_1\bar{z}_2 Q_1$,

$T_2 = \overline{Q}_1 Q_2 \vee z_2\overline{Q}_1 \vee z_1\overline{Q}_1$.

Функции выхода:

$y_1 = \bar{z}_1\bar{z}_2\overline{Q}_2 \vee \bar{z}_1 z_2 Q_2 \vee z_1 z_2 Q_1 \vee z_1 z_2 \overline{Q}_2$,

$y_2 = \bar{z}_1\bar{z}_2\overline{Q}_1 \vee \bar{z}_1 z_2 Q_2 \vee z_1\overline{Q}_1\overline{Q}_2$.

Таким образом, получены логические выражения для функций активации и выходов цифрового автомата.

3.5 Счётчики

К счетчикам относят цифровые автоматы с памятью, которые под действием входных импульсов переходят из одного состояния в другое, фиксируя тем самым число поступивших на их вход импульсов в том или ином коде.

Специфичной для счетчиков операцией является изменение их содержимого на единицу, которая может быть и условной. Прибавление такой единицы соответствует операции инкрементации, вычитание – операции декрементации. Обычно счетчиками выполняются также и другие операции – сброс, установка, параллельная загрузка и др.

Счетчик характеризуется модулем счета М (емкостью), т. е. числом возможных состояний счетчика. После поступления на счетчик М входных сигналов начинается новый цикл, повторяющий предыдущий.

По направлению счёта счётчики делятся на суммирующие, или прямого счёта, и вычитающие, или обратного счёта, а также реверсивные, или с изменением направления счёта. По способу кодирования различают – двоичные, с кодом Грея и др. Соответственно организации межразрядных связей выделяют счётчики с последовательным, параллельным и комбинированным переносами. По одновременности срабатывания различают синхронные и асинхронные счётчики.

Счётчики обычно используют двумя способами: для регистрация числа поступивших на счетчик сигналов и как делитель частоты.

В первом случае результатом работы является содержимое счетчика. Во втором случае выходными сигналами являются импульсы переполнения счетчика.

Как и любой автомат, счетчик можно строить на триггерах любого типа, но удобнее всего использовать для этого триггеры типа Т (счетные) и JK, имеющие при $J = K = 1$ счетный режим.

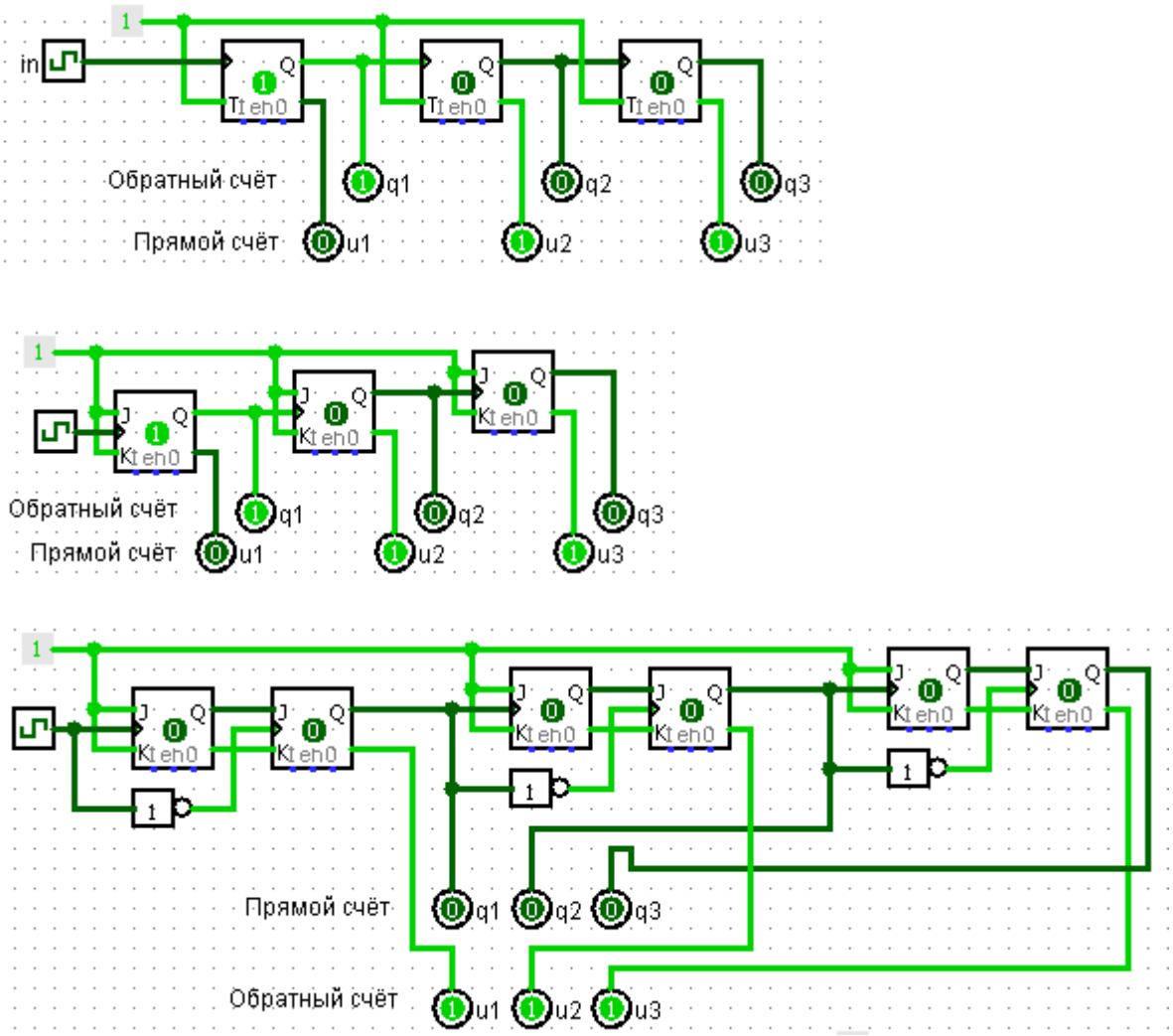


Рисунок 3.43 – Асинхронный трёхразрядный суммирующий счётчик на T-триггерах, JK-триггерах и двухтактных JK-триггерах

Быстродействие счетчика характеризуется временем установления в нем нового состояния (для первого режима), а также максимальной частотой входных сигналов f_{max}

Рассмотрим схему асинхронного суммирующего трёхразрядного счётчика на T-триггерах (рисунок 3.42).

При подаче тактовых сигналов на вход in на выходах $q_1q_2q_3$ формируются сигналы обратного счёта, а на выходах $u_1u_2u_3$ – прямого. Разряды u_3 и q_3 являются старшими, u_1 и q_1 – младшими.

Так как каждый триггер переключается в новое состояние с некоторой задержкой, то быстродействие этих счётчиков обратно пропорционально разрядности. Чем больше разрядов, тем медленнее он работает.

Общую схему работы таких счётчиков можно представить в виде цепочки последовательно включённых счётных триггеров:

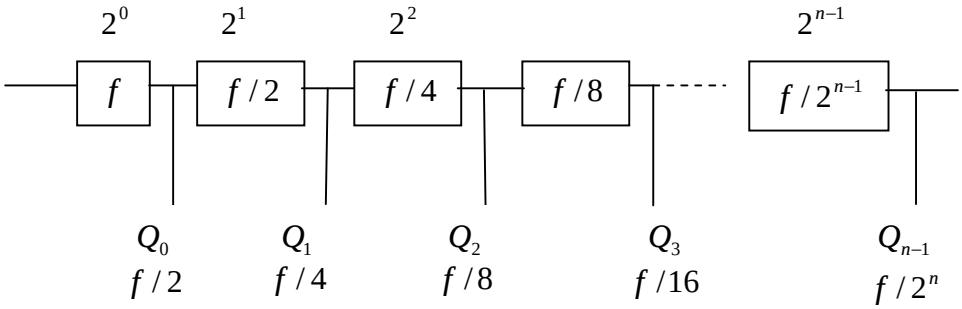


Рисунок 3.44 – Общая схема работы суммирующих асинхронных счётчиков

Представление счетчика цепочкой Т-триггеров справедливо как для суммирующего, так и для вычитающего вариантов, поскольку закономерность по соотношению частот переключения разрядов сохраняется как при просмотре таблицы сверху вниз (прямой счет), так и снизу вверх (обратный счет). Различия при этом состоят в направлении переключения предыдущего разряда, вызывающего переключение следующего.

Полученные структуры относятся к асинхронным счетчикам, т. к. в них каждый триггер переключается выходным сигналом предыдущего, и эти переключения происходят не одновременно. Переключение одного триггера за другим есть не что иное, как распространение переноса по разрядам числа при изменении содержимого счетчика.

В худшем случае перенос распространяется по всей разрядной сетке от младшего разряда к старшему, т. е. для установления нового состояния должны переключиться последовательно все триггеры. Отсюда видно, что время установления кода в асинхронном счетчике составит величину $t_{\text{уст}} = nt_{\text{тг}}$, где n – число триггеров.

Максимальная частота входных импульсов в режиме деления частоты ограничивается возможностями триггера младшего разряда, т. к. все последующие разряды переключаются с более низкими частотами. Особенностью последовательных счетчиков является возникновение в переходных процессах ложных состояний из-за задержек переключения триггеров. Опасность воздействия коротких ложных импульсов на ЦУ заставляет прибегать при необходимости к стробированию выхода счетчика.

Более быстродействующими являются синхронные счётчики. Существует два вида таких устройств: с последовательным сквозным переносом и с параллельным переносом.

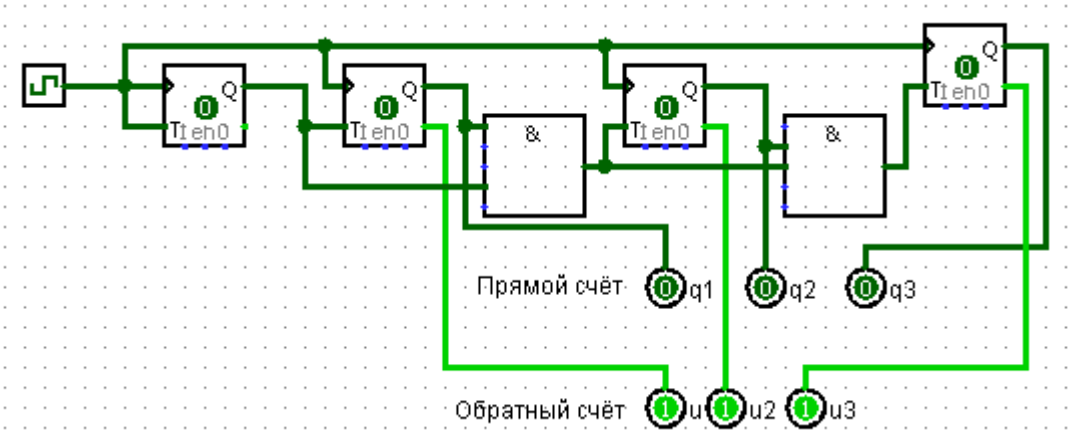


Рисунок 3.45 – Схема синхронного счётчика с последовательным сквозным переносом

Задержка в счётчиках с последовательным сквозным переносом составляет величину $t_{уст} = nt_{и}$, где n – число элементов И. Таким образом, задержка такого счётчика всё же зависит от его разрядности.

Счётчики с параллельным переносом лишены этого недостатка и поэтому являются наиболее быстродействующими.

Недостатком таких устройств являются многовходовые элементы И.

Счётчик характеризуется коэффициентом пересчёта M. Он численно равен количеству импульсов, которые, будучи поданы на вход счётчика, установленного в исходное состояние, снова устанавливают его в исходное состояние. Для двоичных счётчиков $M = 2^n$, где n – число разрядов счётчика.

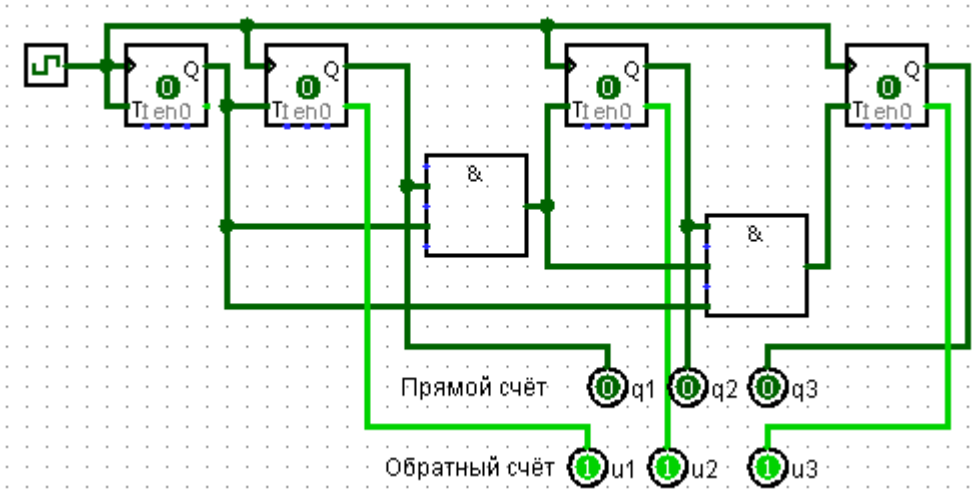


Рисунок 3.46 – Синхронный счётчик с параллельным переносом

Частота сигналов на выходе двоичного счётчика уменьшается в два раза по отношению к частоте входа. В общем случае, двоичный счётчик осуществляет деление входной частоты на коэффициент пересчёта: $f_{вых} = f_{вх} / M$. Это позволяет использовать счётчики в качестве делителей частоты входного сигнала. Сформированные частоты могут использоваться как тактовые, для синхронизации различных устройств, либо в качестве генераторов опорных частот в радиопередатчиках и приёмниках.

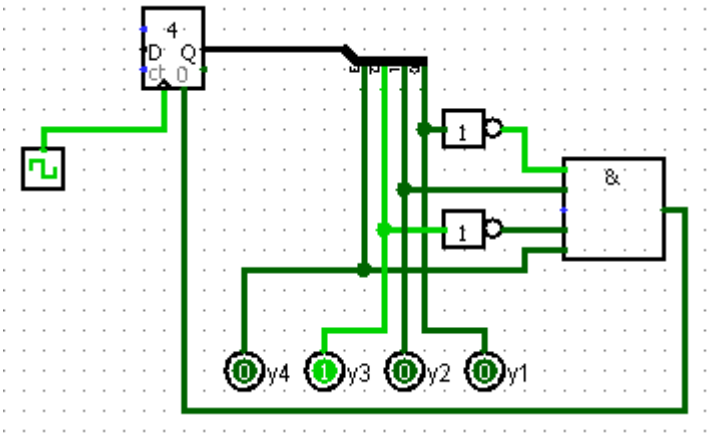


Рисунок 3.47 – Счётчик с коэффициентом пересчёта $M_{10} = 10$

На практике часто нужны счётчики с коэффициентами пересчёта, отличающимися от степеней двойки. Это счётчики с произвольными M. Построить такой счётчик можно за счёт исключения лишних значений. Эта операция осуществляется за счёт обратной связи, подключённой на вход R

сброса состояния счётчика. Для исключения лишних состояний используется дешифратор, который определяет нужное состояние счётчика и сбрасывает его в исходное состояние. В качестве примера рассмотрим счетчик с коэффициентом пересчёта $M = 10_{10} = 1010_2$. Это устройство должно считать от 0 до 9 и сбрасываться в исходное состояние. Для этого к выходам счётчика подключим дешифратор, который при состоянии выходов $y_4 y_3 y_2 y_1 = 1010$ подаст сигнал сброса на счётчик (рисунок 3.47).

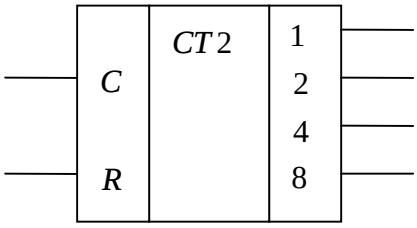


Рисунок 3.48 – Условное обозначение счётчика

Условное графическое обозначение суммирующего двоичного четырёхразрядного счётчика содержит выходы 1-8, счётный вход C, и вход сброса R, который позволяет записать во все триггеры счётчика нулевые значения. Это состояние называют исходным состоянием счётчика.

3.6 Регистры

Регистр – цифровое устройство, используемое для хранения двоичных данных и выполнения преобразований над ними. Эти устройства являются самыми распространёнными узлами цифровых устройств. Регистры являются частью процессорных систем. Они хранят операнды, признаки и промежуточные результаты арифметических операций, индексы элементов массивов, указатели на области памяти, адреса и селекторы сегментов памяти и многое другое.

Эти устройства оперируют со связанными битовыми переменными, составляющими слово. В регистре над словами выполняется ряд операций:

- приём слова в регистр (установка состояния);
- передача слова из регистра;
- сдвиг слова влево или вправо на заданное число разрядов в сдвиговых регистрах;

– преобразование последовательного кода слова в параллельный и обратно;
– установка регистра в начальное состояние (сброс).

Регистры состоят из разрядных схем, в которые входят триггеры, а также часто и логические элементы для управления регистром. По числу линий передачи сигналов регистры делятся на однофазные и парафазные, по системе синхронизации на однотактные, двухтактные и многотактные. Основным классификационным признаком является способ приема и выдачи данных. По этому признаку различают параллельные (статические) регистры, последовательные (сдвигающие) и параллельно-последовательные.

Последовательно-параллельные регистры имеют входы-выходы последовательного и параллельного типа. Имеются варианты с последовательным входом и параллельным выходом (SIPO, Serial Input – Parallel Output), параллельным входом и последовательным выходом (PISO, Parallel Input – Serial Output), а также варианты с возможностью любого сочетания способов приема и выдачи слов. Общими для разрядов обычно являются цепи тактирования, сброса и установки, разрешения выхода или приема данных, т.е. цепи управления.

В параллельных регистрах прием и выдача слов производятся по всем разрядам одновременно. В последовательных регистрах слова принимаются и выдаются разряд за разрядом. Их называют сдвигающими, т. к. тактирующие сигналы при вводе и выводе слов перемещают их в разрядной сетке. Сдвигающий регистр может быть нереверсивным (с однонаправленным сдвигом) или реверсивным (с возможностью сдвига в обоих направлениях).

Схематически регистр представляет собой параллельное или последовательное соединение триггеров. Обычно это D-триггеры. Количество триггеров в регистре определяет его разрядность. В качестве отдельных микросхем существуют четырёх- и восьмиразрядные триггеры.

В параллельных регистрах входы и выходы всех триггеров используются независимо. Данные в них записываются одновременно. Разрядность регистра определяется числом его триггеров.

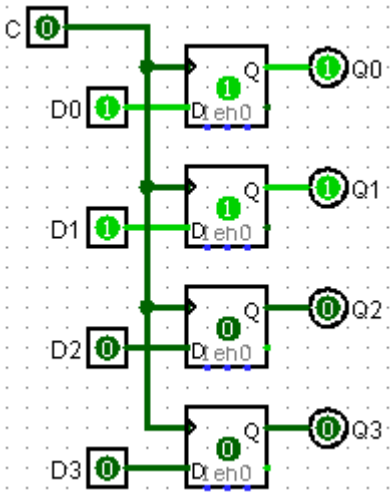


Рисунок 3.49 – Схема четырёхразрядного параллельного регистра

Условно-графическое обозначение регистра приведено на рисунке 3.50.

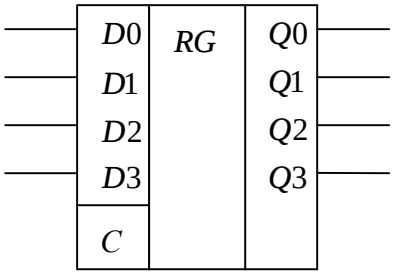


Рисунок 3.50 – Условное обозначение регистра

В условном обозначении не показаны инверсные выходы триггеров. В микросхемах они обычно не выводятся наружу для экономии выводов корпуса. При записи данных в параллельный регистр все биты записываются одновременно, поэтому все тактовые входы триггеров, входящих в его состав, объединены. Назначение разрядов в параллельном регистре является условным. Её можно свободно менять. При этом нумерацию входов регистра нужно менять соответственно. Разрядность регистров можно наращивать, используя стандартные блоки. Например, шестнадцатиразрядный параллельный регистр может быть реализован на четырёх регистрах разрядности четыре.

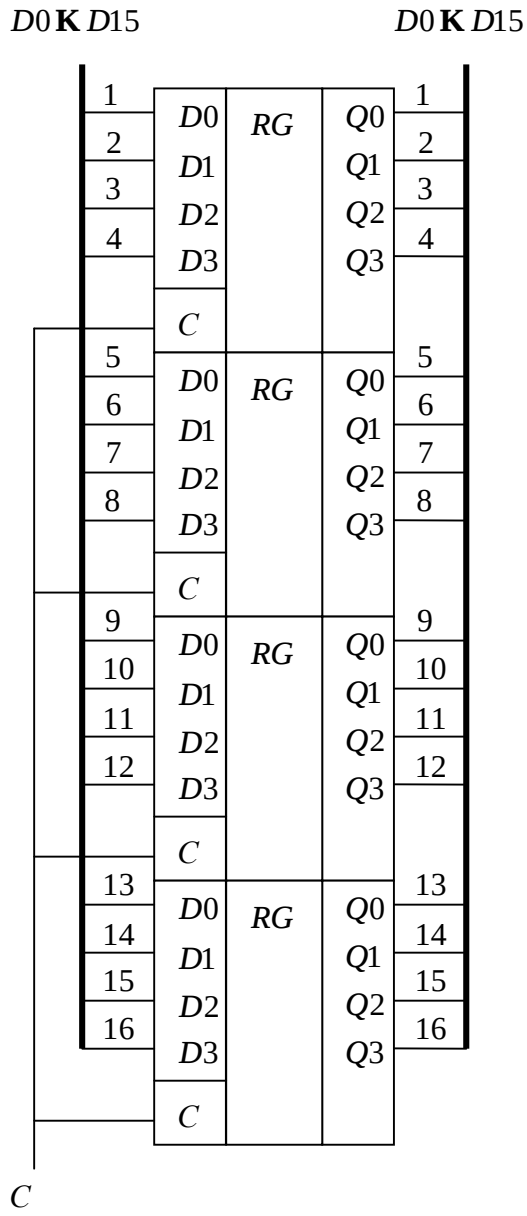


Рисунок 3.51 – Схема шестнадцатиразрядного параллельного регистра

Кроме параллельного соединения триггеров, для построения регистров используется последовательное их соединение. Последовательный регистр, или регистр сдвига, обычно применяется для преобразования последовательного кода в параллельный и наоборот.

В принципиальной схеме четырёхразрядного последовательного регистра выход первого триггера соединён со входом второго. Выход второго – со входом третьего и т.д.

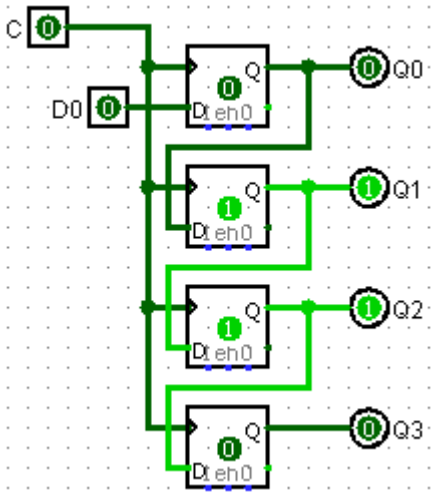


Рисунок 3.52 – Четырёхразрядный последовательный регистр

Входы синхронизации отдельных триггеров объединяются. Это обеспечивает одновременность смены внутреннего состояния всех триггеров. В сдвигающих регистрах, не имеющих логических элементов в межразрядных связях, нельзя применять одноступенчатые триггеры, управляемые уровнем, поскольку некоторые триггеры могут за время действия разрешающего уровня синхросигнала переключиться неоднократно, что недопустимо.

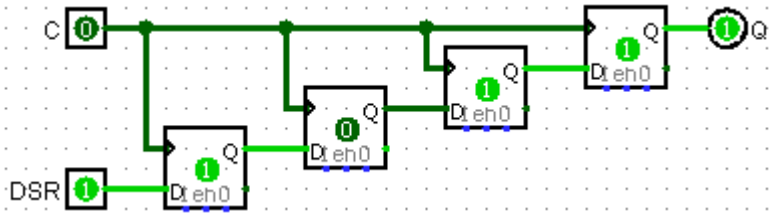


Рисунок 3.53 – Четырёхразрядный регистр со сдвигом вправо

В однотактных регистрах со сдвигом вправо и влево слово сдвигается при поступлении синхросигнала «Сдвиг». Вход и выход последовательные (DSR – Data Serial Right, DSL – Data Serial Left). В реверсивном регистре имеются связи триггеров с обоими соседними разрядами, но соответствующими сигналами разрешается работа только одной группы этих связей, «влево» или «вправо».

Направление сдвига в регистрах не является геометрическим понятием, имеется в виду лишь направление сдвига в сторону старших или младших разрядов.

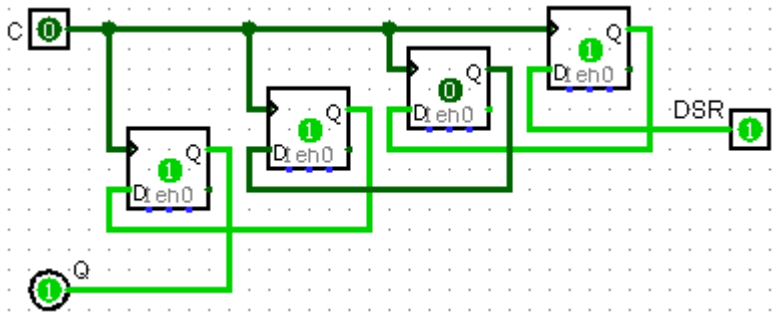


Рисунок 3.54 – Четырёхразрядный регистр со сдвигом влево

В зависимости от варианта записи слова в регистр одна и та же схема может быть как регистром со сдвигом влево, так и регистром со сдвигом вправо. Поэтому схемотехнически существуют фактически два типа последовательных регистров: сдвигающие и реверсивные.

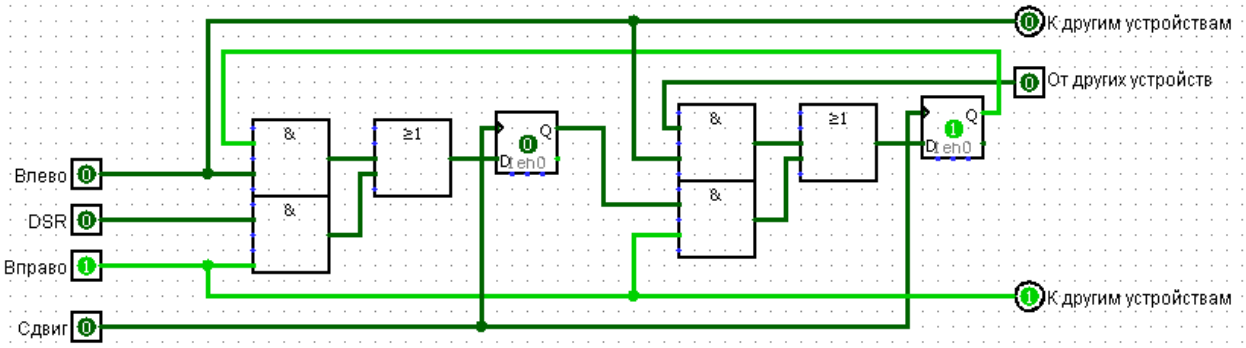


Рисунок 3.55 – Четырёхразрядный реверсивный регистр

Регистры сдвига обычно выполняют как универсальные последовательно-параллельные схемы. Это связано с необходимостью записи в регистр параллельного двоичного кода при преобразовании параллельного кода в последовательный. Переключение регистра из параллельного режима работы в последовательный осуществляется при помощи мультиплексора. Он подключает выходы триггеров регистра либо к внешним выводам микросхемы, либо к выходу предыдущего триггера.

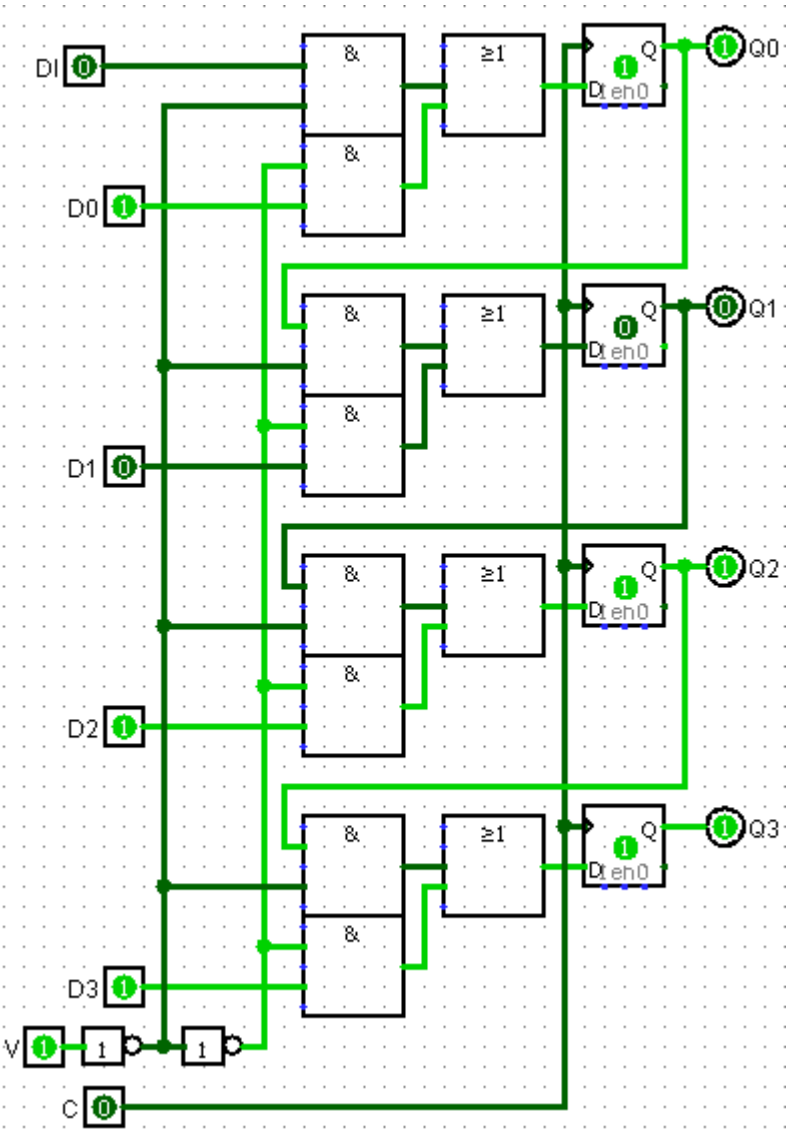


Рисунок 3.56 – Универсальный регистр

В схеме универсального регистра для переключения режима работы используется вход V. Подача на этот вход логической единицы переключает схемы у параллельный режим. При этом на входы ключей, подключённых к информационным входам D, подаётся логическая единица. Это приводит к тому, что сигналы с входов параллельной записи данных поступают на входы логических элементов ИЛИ. При этом на входы ключей, подключённых к выходам предыдущих триггеров, подаются нулевые потенциалы.

Подача на вход V нулевого потенциала приводит к отключению входов параллельных данных от входов триггеров. Сигналы с входа предыдущего триггера проходят через верхние логические элементы И на вход следующего триггера.

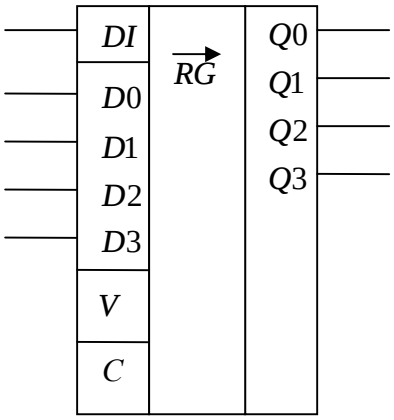


Рисунок 3.57 – Условное обозначение универсального регистра

На условном графическом изображении универсального регистра вход последовательного ввода данных обозначен как DI, вход управления – как V, и вход синхронизации – как C.

Контрольные вопросы к главе 3

1. Что понимают под синтезом комбинационных схем?
2. Для чего предназначены триггеры?
3. Какими цифровыми автоматами являются триггеры, что такое полнота переходов и выходов?
4. Какие бывают состояния триггера, как они обозначаются?
5. Какие типы входов есть у триггера?
6. Каково условное графическое обозначение триггера?
7. Какие существуют типы триггеров по выполняемой функции?
8. Нарисуйте схему однотактного асинхронного RS-триггера в базисе И-НЕ и расскажите о её работе.
9. В каких режимах может работать однотактный асинхронный RS-триггер?
10. Нарисуйте таблицу переходов однотактного асинхронного RS-триггера.
11. Нарисуйте граф однотактного асинхронного RS-триггера.
12. Выведите выражение для функции переходов RS-триггера.
13. Нарисуйте табличную форму матрицы переходов RS-триггера.

14. Нарисуйте схему однотактного синхронного RS-триггера в базисе И-НЕ и расскажите о её работе.
15. Нарисуйте таблицу переходов однотактного синхронного RS-триггера.
16. Нарисуйте схему однотактного синхронного RS-триггера с установочными входами в базисе И-НЕ и расскажите о её работе.
17. Для чего применяют двухтактные триггеры?
18. Нарисуйте схему двухтактного RS-триггера в базисе И-НЕ и расскажите о её работе.
19. Нарисуйте схему двухтактного RS-триггера на базе двух однотактных и расскажите о её работе.
20. Как построить D-триггер на базе синхронного RS-триггера? Расскажите о его работе.
21. Нарисуйте таблицу состояний (переходов) D-триггера.
22. Нарисуйте граф D-триггера.
23. Нарисуйте табличную форму матрицы переходов D-триггера и получите из неё выражение для функции переходов.
24. Нарисуйте матрицу переходов D-триггера.
25. Как построить асинхронный T-триггер на базе D-триггера и RS-триггера? Расскажите о его работе.
26. Нарисуйте таблицу состояний (переходов) T-триггера.
27. Нарисуйте граф T-триггера.
28. Нарисуйте табличную форму матрицы переходов T-триггера и получите из неё выражение для функции переходов.
29. Нарисуйте матрицу переходов T-триггера.
30. Для чего применяется JK-триггер, чем он отличается от RS-триггера?
31. Нарисуйте таблицу состояний (переходов) JK-триггера.
32. Нарисуйте граф JK-триггера.
33. Нарисуйте табличную форму матрицы переходов JK-триггера с условиями переходов.
34. Нарисуйте табличную форму матрицы переходов JK-триггера и получите из неё выражение для функции переходов.

35. Постройте JK-триггер на базе двухтактного RS-триггера за счёт использования обратной связи.
36. Расскажите об S, R, E-триггерах.
37. В чём заключается кодирование внутренних состояний автомата?
38. Что такое функция активации?
39. Что такое функция внешних переходов?
40. Как определяется число триггеров в синтезируемом автомате?
41. Как кодируется внутренние состояния автомата?
42. Как определяется число внешних переменных автомата и как кодируются их состояния?
43. Как составляется закодированная таблица переходов для автомата и для каждого триггера?
44. Как получить таблицы активации триггеров синтезируемого автомата?
45. Как из таблицы активаций получить функции активации для различных входов триггера?
46. Что такое счётчик, какие операции он выполняет, чем он характеризуется?
47. Какова классификация счётчиков?
48. Для чего используются счётчики, чем определяется его быстродействие?
49. Постройте схему асинхронного суммирующего трёхразрядного счётчика на T-триггерах и расскажите об её работе.
50. Выполните ту же схему, что в предыдущем вопросе, использую JK-триггеры.
51. Постройте схему асинхронного суммирующего счётчика для $n-1$ разрядов.
52. Каково время установления кода в асинхронном счётчике, чем в них применяют стробирование?
53. Какие существуют виды синхронных счётчиков? Постройте схему такого устройства для трёх разрядов. Чем определяется его задержка?
54. Постройте схему трёхразрядного синхронного счётчика с параллельным переносом и расскажите об её работе.

55. Как построить счётчик с коэффициентом пересчёта M, отличном от степени двойки?
56. Постройте синхронный четырёхразрядный счётчик с $M = 10_{10}$.
57. Каково условное обозначение счётчика?
58. Что такое регистр, какие операции выполняет это устройство?
59. Как классифицируются регистры?
60. Расскажите о работе параллельного и сдвигающего регистров.
61. Постройте схему параллельного регистра и расскажите об её работе.
62. Как наращивать разрядность регистра? Постройте шестнадцатиразрядный регистр на основе четырёхразрядных.
63. Как строятся регистры сдвига? Постройте такую схему для четырёх разрядов и расскажите об её работе.
64. Постройте схемы регистров влево и вправо и расскажите об их работе.
65. В чём состоит смысл понятия сдвига в регистре? Постройте схему регистра на два разряда со сдвигом влево и вправо и расскажите об её работе.
66. Как работает универсальная последовательно-параллельная схема регистра? Постройте схему такого устройства для четырёх разрядов. Каково условное обозначение такого регистра?

Список литературы

1. Микушин, А.В. Цифровые устройства и микропроцессоры: учеб. пособие / А.В. Микушин, А.М. Сажнев, В.И. Сединин. – СПб.: БХВ-Петербург, 2010. – 832 с.
2. Таненбаум, Э. Архитектура компьютера: пер. с англ. / Э. Таненбаум, Т. Остин. – 6-е изд. СПб. Питер, 2013. 816 с.
3. Харрис, Д. Цифровая схемотехника и архитектура компьютера: пер. с англ./ Д. Харрис, С. Харрис – 2-е изд. – Уолтэм: Morgan Kaufman, 2013. – 1621 с.
4. Угрюмов Е.П. Цифровая схемотехника: учеб. пособие для вузов / Е.П. Угрюмов. – 3-е изд. СПб.: БХВ-Петербург, 2010. – 816 с.
5. Цилькер, Б. Я. Организация ЭВМ и систем: учебник для вузов. / Б.Я. Цилькер, С.А. Орлов. – СПб. : Питер, 2006. – 668 с.

Учебное издание

Юрий Стефанович **Сербулов**
Евгений Александрович **Аникеев**

ЦИФРОВЫЕ АВТОМАТЫ

Учебное пособие

Редактор Е.А. Богданова

Подписано в печать . Формат 60×90 /16.
Усл. печ. л. . Уч.-изд. л. . Тираж экз. Заказ
ФГБОУ ВО «Воронежский государственный лесотехнический университет имени Г.Ф. Морозова»
РИО ФГБОУ ВО «ВГЛТУ». 394087, г. Воронеж, ул. Тимирязева, 8
Отпечатано в УОП ФГБОУ ВО «ВГЛТУ»
394087, г. Воронеж, ул. Докучаева, 10