ПРАКТИЧЕСКАЯ РАБОТА №14
УНИВЕРСАЛЬНЫЕ ЛОГИЧЕСКИЕ МОДУЛИ

11 Цель работы:

1. Ознакомится с концепцией LUT как универсального логического модуля.
2. Применить LUT для реализации булевых функций и комбинационных устройств.
3. Реализовать универсальный преобразователь кодов на основе шифратора и дешифратора.

12 Теоретический блок:

12.1 Реализация логических функций с использованием мультиплексоров

Помимо основного назначения, мультиплексоры могут использоваться как таблицы преобразования для выполнения логических функций. На рис. 2.1 приведён мультиплексор с тремя адресными входами, используемый для реализации булевой функции, суммирующей три входа по модулю два (1), таблица истинности приведена в табл. 2.1.

$$f = a \oplus b \oplus c \tag{1}$$

Таблица 2.1. Таблица истинности

a	b	c	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

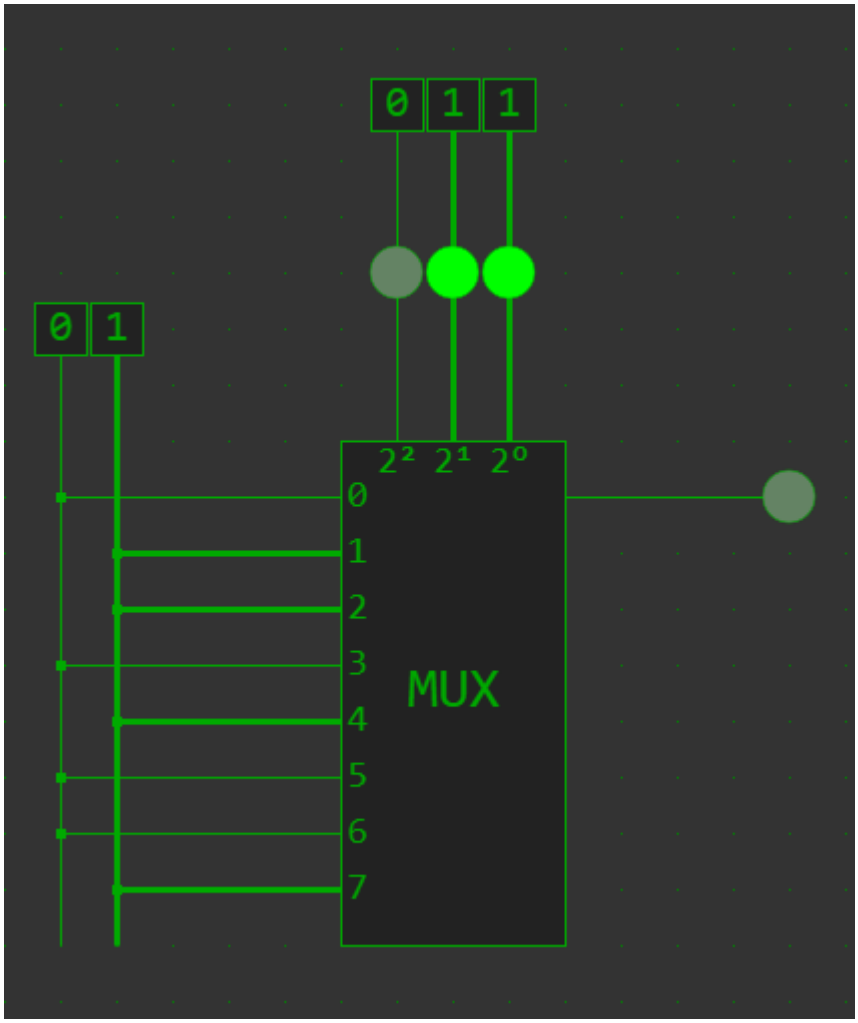

Рисунок 2.1. Схема реализации функции с использованием мультиплексора

При подаче на адресные входы мультиплексора значений переменных из таблицы истинности на выход подаётся информация с соответствующего входа, подключённого к логической единице или нулю, в соответствии с таблицей истинности интересующей нас функции.

Этот принцип нашёл широкое распространение при создании программируемой логики, в частности в ПЛИС (программируемых логических интегральных схемах). Для установки соответствующих значений на входах мультиплексора можно применять сдвиговый регистр. Пример сдвигового регистра, состоящего из трёх D-триггеров приведён на рис. 2.2. При подаче на вход первого регистра значения со входа data, и по заднему фронту синхронизирующего сигнала, информация записывается в первый регистр. То, что хранилось в первом регистре, будет записано во второй и так далее.

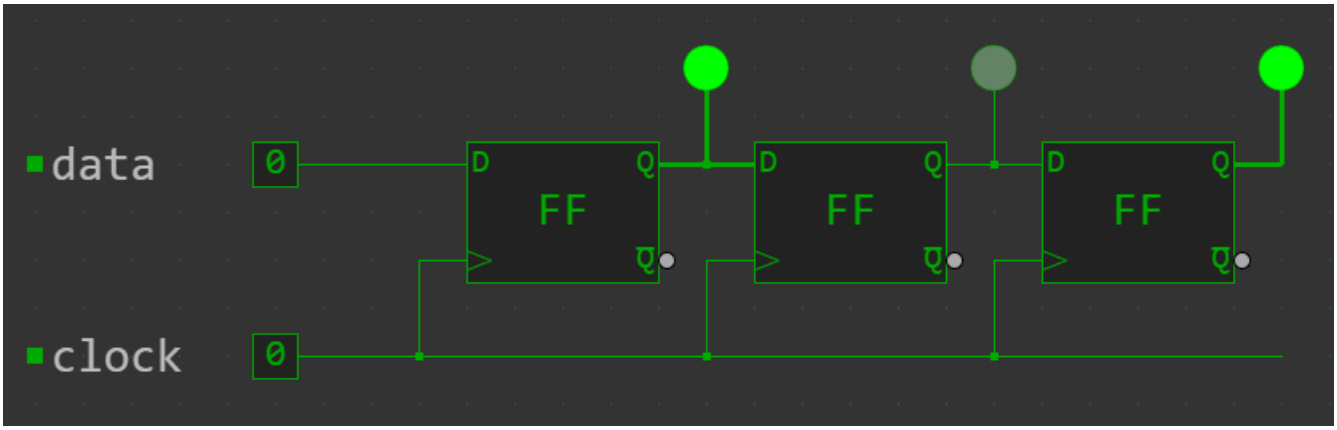

Рисунок 2.2. Пример сдвигового регистра на трёх D-триггерах

Таким образом представляется возможным программировать произвольную функцию алгебры логики. Пример комбинации сдвигового регистра и мультиплексора представлен на рис. 2.3.

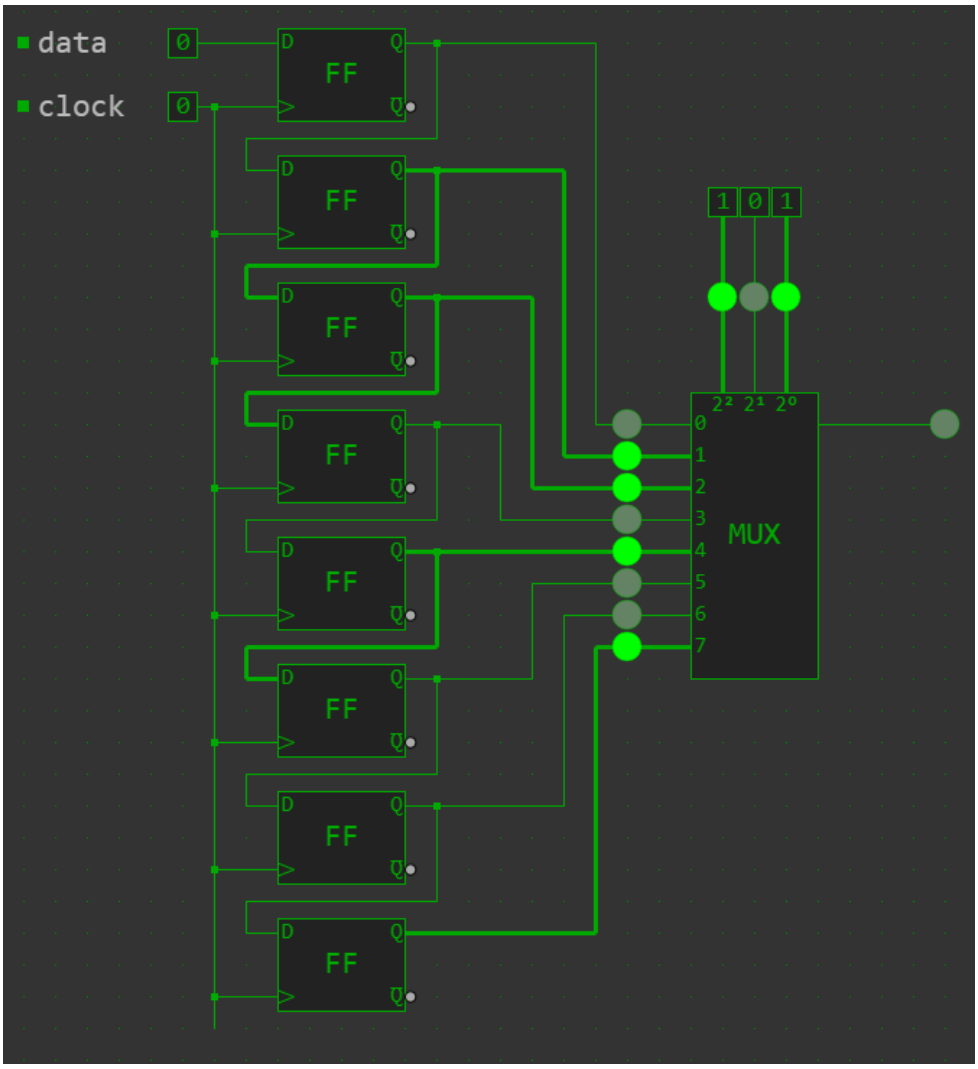

Рисунок 2.3. Пример реализации программируемой логики трёх переменных

Такая комбинация носит названия LUT (lookup table, таблица поиска). На основе этого принципа спроектированы множество ПЛИС, на рис. 2.4 приведена структурная схема ПЛИС. Элементы схемы:

● CLB (Configurable logic blocks, конфигурируемые логические блоки), являются способом реализации логических функций внутри ПЛИС, в её состав входит LUT;

● IOB (input/output blocks, блоки ввода/вывода) является интерфейсами ПЛИС с внешними схемами;

● матрица подключений необходима для соединения нескольких CLB для достижения требуемой функциональности.

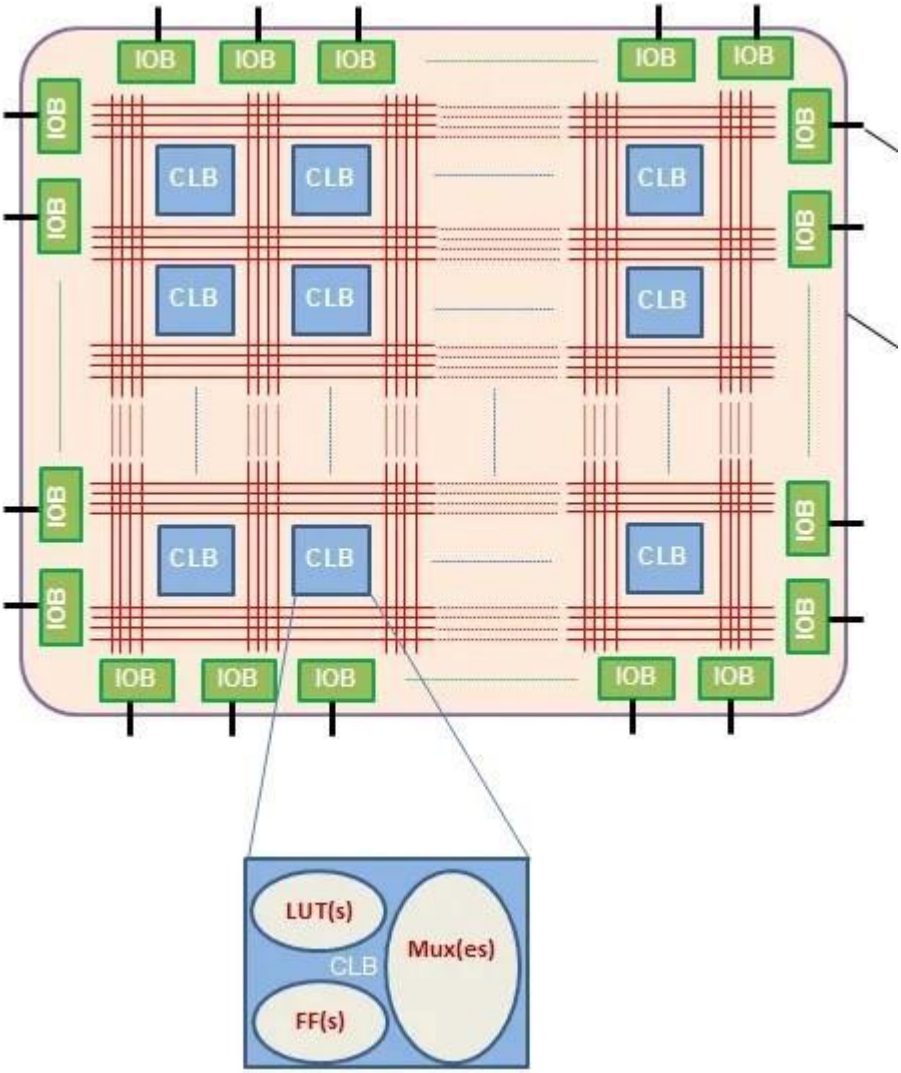

Рисунок 2.4. Пример типовой архитектуры ПЛИС

В отличие от обычных цифровых микросхем, логика работы ПЛИС, как было продемонстрировано, не определяется при изготовлении, а задаётся посредством программирования (проектирования). Для программирования используются программатор и интегрированные среды разработки, позволяющие задать желаемую структуру цифрового устройства в виде принципиальной электрической схемы или программы на специальных языках описания аппаратуры (Verilog, VHDL, AHDL и других).

12.2 Реализация преобразователя кода

При решении задач проектирования цифровых устройств встречаются задачи проектирования преобразователей кодов, например, преобразовать двоичный код в соответствующий ему код Грея. Функция (2), которая способна сделать это преобразование требует операции побитного исключающего или (^) и битового сдвига вправо (>>).

$$B = gray(A) = A \oplus (A \gg 1) \qquad (2)$$

Пример таблицы истинности этого преобразования приведён в табл. 2.2.

Таблица 2.2. Таблица истинности преобразования в код Грея

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

Для реализации этой функции можно использовать непосредственное её определение, как приведено на рис. 2.5.

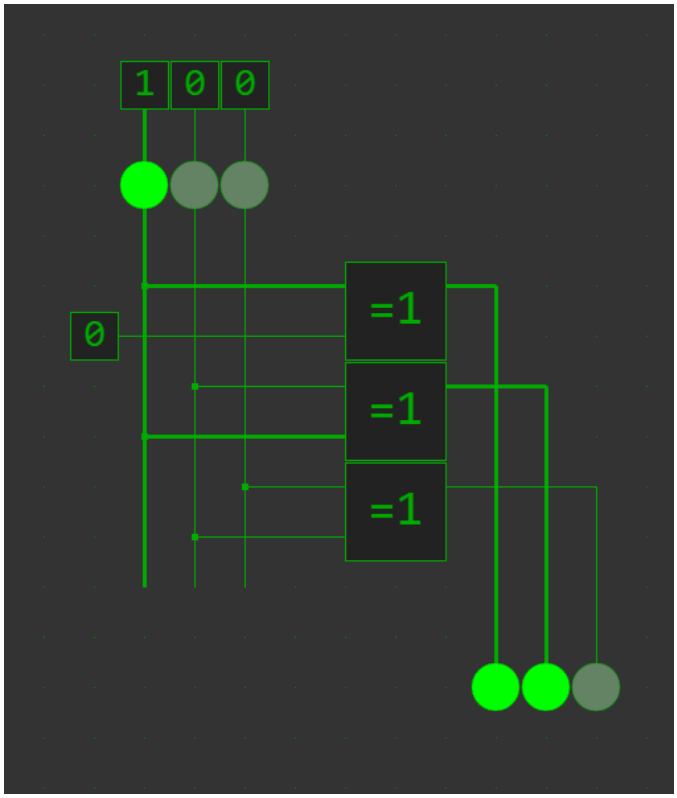

Рисунок 2.5. Пример реализации преобразования в код Грея

Однако в общем случае преобразование кодов представляет собой трудоёмкое в реализации выражение, в таких случаях целесообразно рассмотреть применение универсального преобразователя кодов.

Такие преобразователи строятся на паре, состоящей из шифратора и дешифратора, выходы шифратора, соединяются со входами дешифратора в соответствии с требуемым преобразованием. Пример реализации дешифратора трёх переменных в программной среде simulator.io приведён на рис. 2.6.

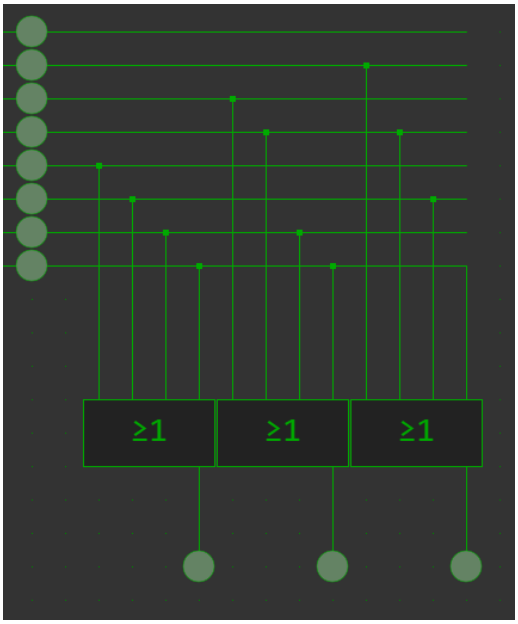

Рисунок 2.6. Пример реализации дешифратора с тремя выходами

На рис. 2.7 приведена схема реализации преобразования в код Грея на основе шифратора и дешифратора.

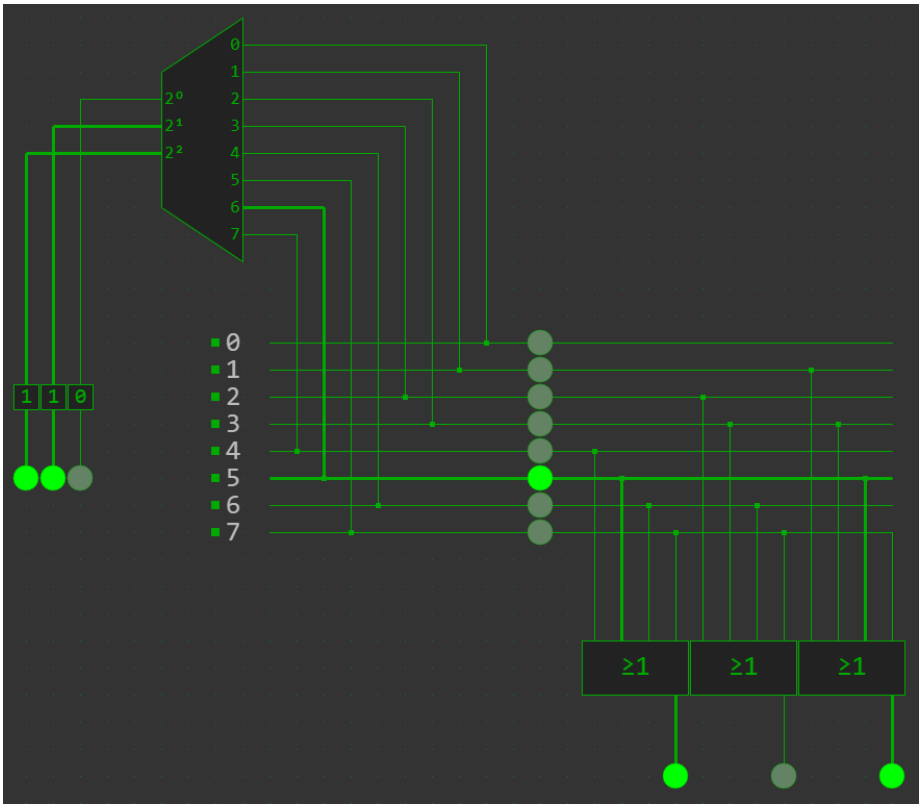

Рисунок 2.7. Пример реализации преобразователя кодов

При подаче на адресный вход шифратора значений, логическая единица передаётся на соответствующий вывод шифратора, который подключён согласно

таблице истинности (табл. 2.2) ко входу дешифратора. В итоге у нас появляется возможность реализовать таким способом любое преобразование кодов. Однако, стоит помнить, что с ростом количества бит, над которыми производится преобразование, растёт и разрядность соответствующих шифраторов и дешифраторов, что делает их нецелесообразным в некоторых приложениях. Например, при умножении двух беззнаковых 16 битовых значений, потребуется мультиплексор с 32 адресными входами и соответственно 2^32 выходами, реализация такого рода преобразования не представляется возможным посредством описываемого метода.

Можно заметить, что реализованный преобразователь — это три функции трёх переменных. На этом наблюдении можно построить программируемый преобразователя кодов, пример которого приведён на рис. 2.8. На данной схеме присутствуют три LUT элемента, регистры которых соединены последовательно, при записи последовательности, кодирующей преобразование кода, в данном случае биты заносятся в обратном порядке, сначала B_0 снизу вверх, потом B_1 и так далее.

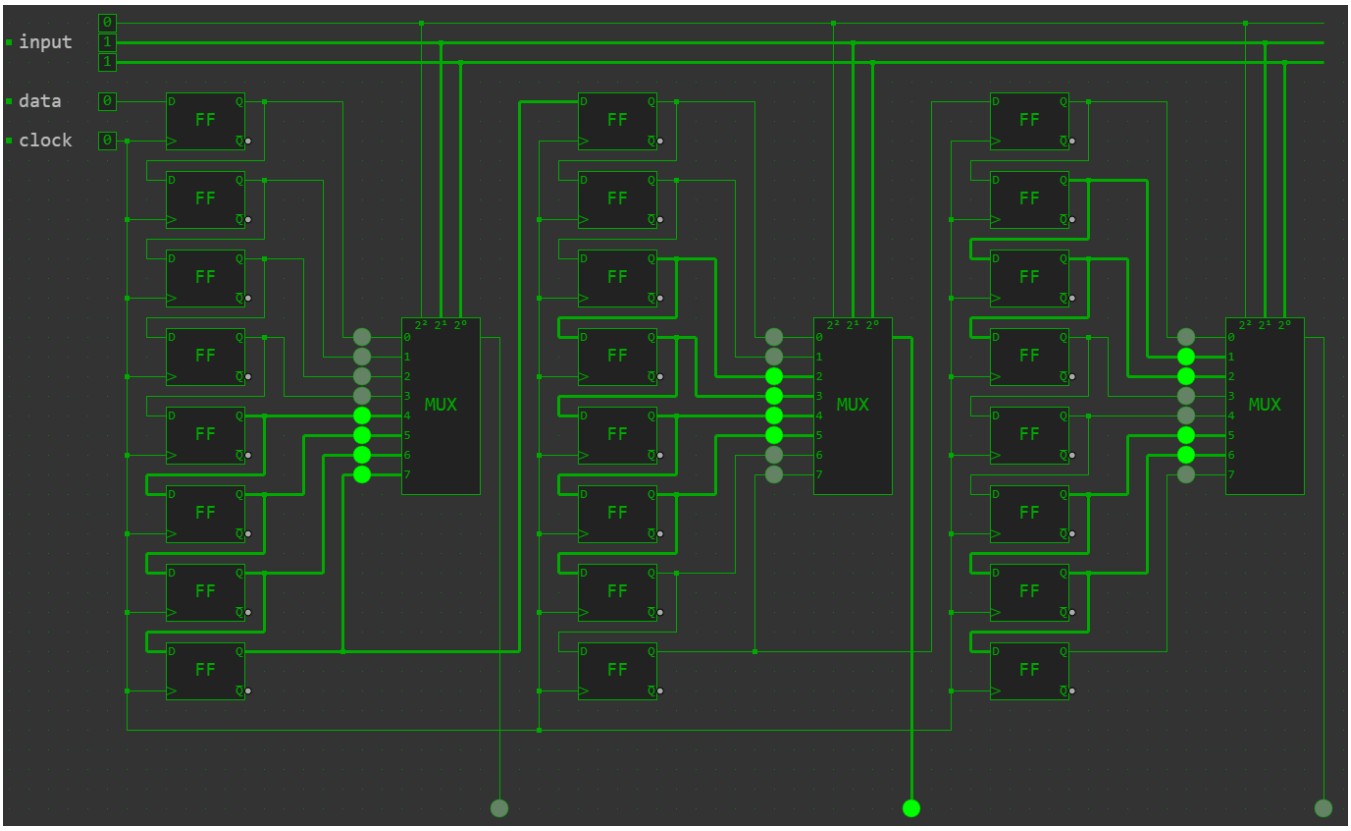

Рисунок 2.8. Пример реализации программируемого преобразователя кодов

13 Практический блок:

13.1 Задание 1

Пусть дана функция алгебры логики:

$$f = (b \wedge c) \vee (b \wedge \bar{a}) \vee (c \wedge \bar{a}) \vee (a \wedge \bar{b} \wedge \bar{c})$$

Восстановим её таблицу истинности (табл. 3.1).

Таблица 3.1. Таблица истинности функции

abc	f
000	0
001	1
010	1
011	1
100	1
101	0
110	0
111	1

Построим схему программируемой логики, согласно рис. 2.3 и запрограммируем её значениями, согласно таблице истинности (табл. 3.1)

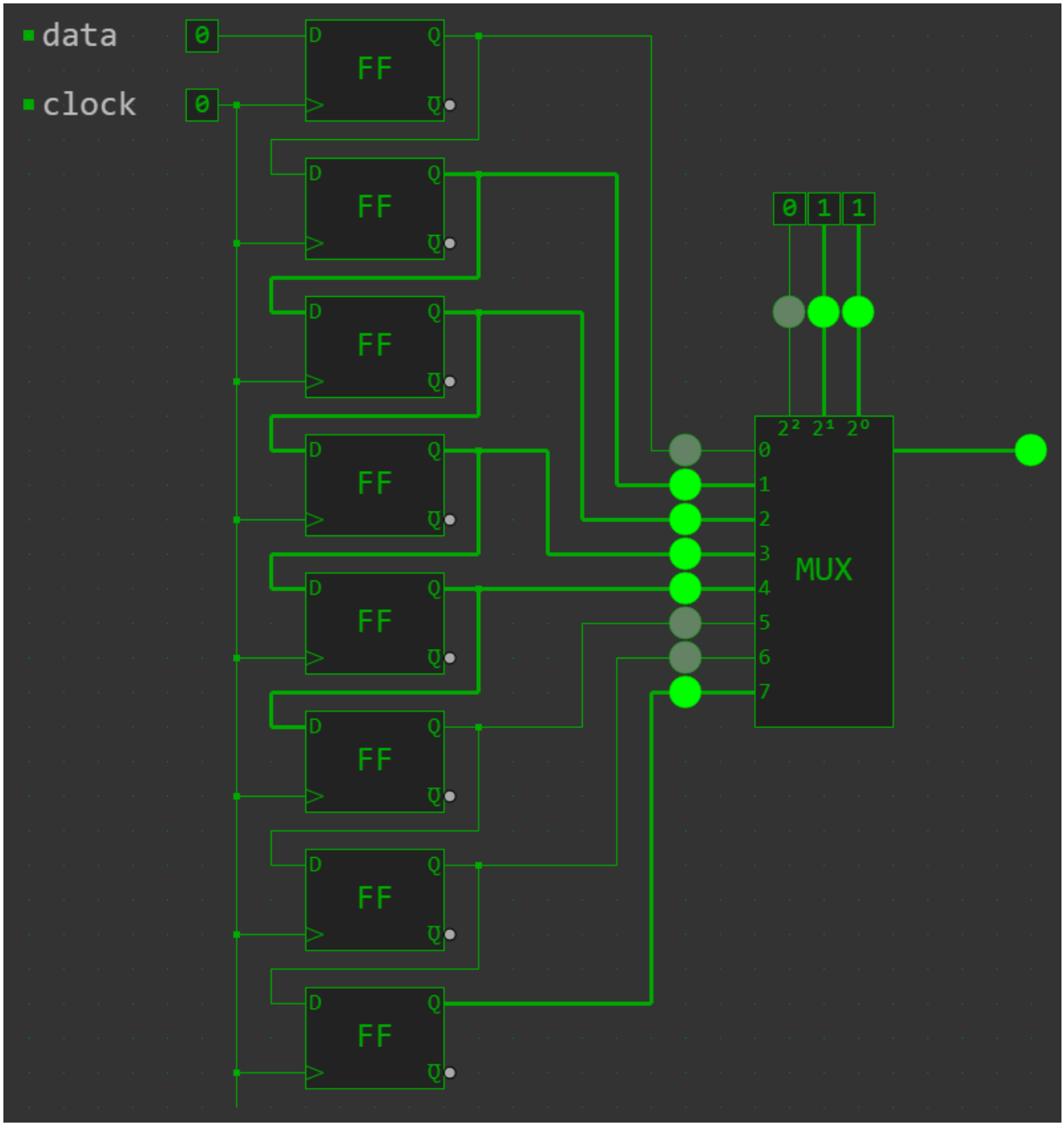

Рисунок 3.1. Реализация функции

13.2 Задание 2

Пусть дано преобразование трёхразрядных кодов в трёхразрядные коды, табл. 3.2.

Таблица 3.2. Таблица преобразования кодов, вариант XX

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	0	1
0	0	1	1	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	0	0	0
1	0	1	1	1	1
1	1	0	1	0	0
1	1	1	1	1	0

Построим схему преобразователя кодов согласно рис. 2.8 и занесём в неё управляющую последовательность согласно табл. 3.2.

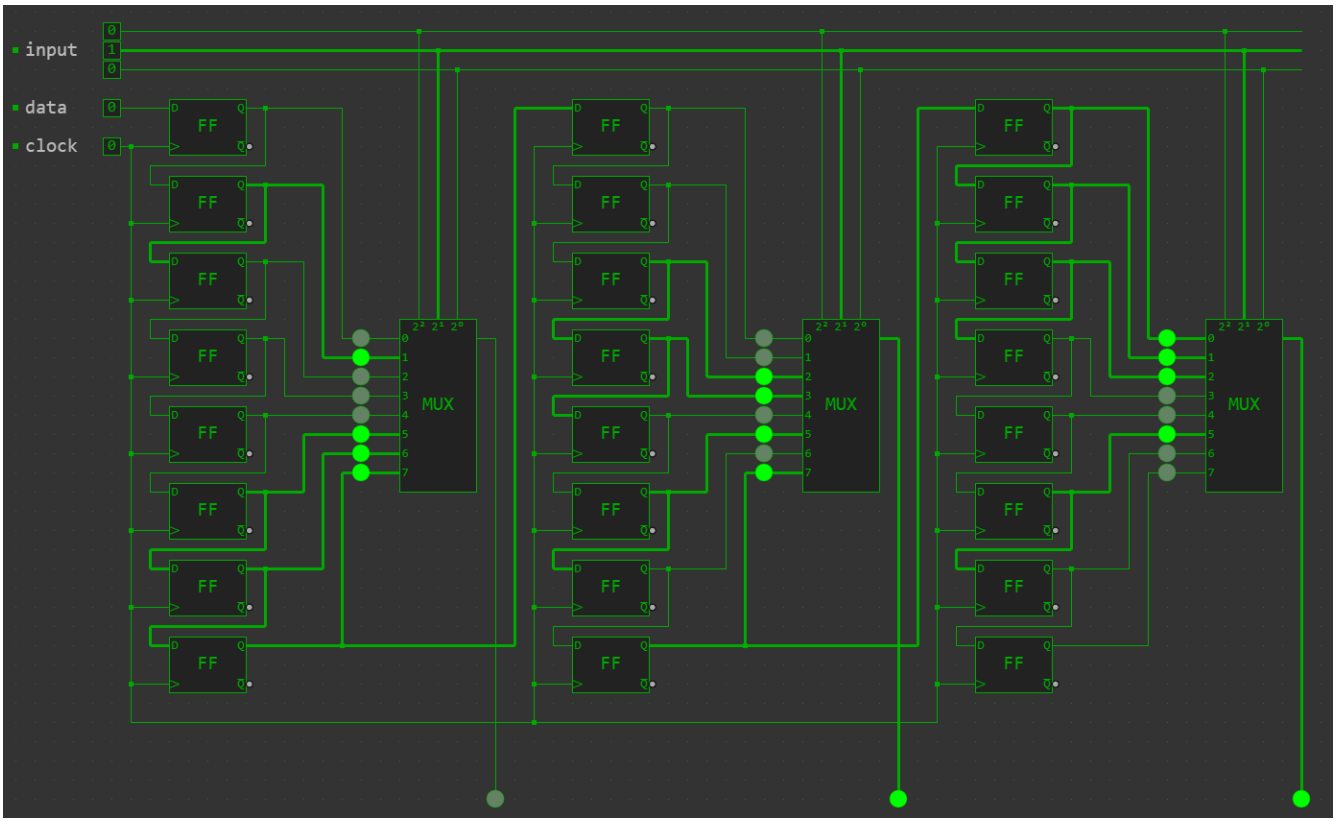

Рисунок 3.2. Реализация преобразователя кодов

14 Задание для отчёта по практической работе:

Восстановите таблицу истинности функции, заданной по вашему варианту.

Постройте схему программируемой логики на основании рис. 2.3 и занесите в неё управляющую последовательность в соответствии с вашей функцией, проверьте корректность работы схемы.

Постройте схему преобразователя кодов и занесите в неё управляющую последовательность согласно вашему варианту, протестируйте корректность её работы.

15 Варианты для самостоятельной работы:

15.1 Задание 1

1. $f = (a \wedge \bar{b}) \vee (a \wedge \bar{c}) \vee (\bar{b} \wedge \bar{c}) \vee (b \wedge c \wedge \bar{a})$
2. $f = (a \wedge b) \vee (a \wedge c) \vee (b \wedge c) \vee (\bar{a} \wedge \bar{b} \wedge \bar{c})$
3. $f = (a \wedge c) \vee (a \wedge \bar{b}) \vee (c \wedge \bar{b}) \vee (b \wedge \bar{a} \wedge \bar{c})$
4. $f = (a \wedge b \wedge c) \vee (\bar{a} \wedge \bar{b}) \vee (\bar{a} \wedge \bar{c}) \vee (\bar{b} \wedge \bar{c})$
5. $f = (a \wedge b) \vee (a \wedge c) \vee (b \wedge c) \vee (\bar{a} \wedge \bar{b} \wedge \bar{c})$
6. $f = (a \wedge c) \vee (a \wedge \bar{b}) \vee (c \wedge \bar{b}) \vee (b \wedge \bar{a} \wedge \bar{c})$
7. $f = (a \wedge c) \vee (a \wedge \bar{b}) \vee (c \wedge \bar{b}) \vee (b \wedge \bar{a} \wedge \bar{c})$
8. $f = (a \wedge b) \vee (a \wedge c) \vee (b \wedge c) \vee (\bar{a} \wedge \bar{b} \wedge \bar{c})$
9. $f = (a \wedge \bar{b}) \vee (a \wedge \bar{c}) \vee (\bar{b} \wedge \bar{c}) \vee (b \wedge c \wedge \bar{a})$
10. $f = (a \wedge b) \vee (a \wedge \bar{c}) \vee (b \wedge \bar{c}) \vee (c \wedge \bar{a} \wedge \bar{b})$
11. $f = (a \wedge b \wedge c) \vee (a \wedge \bar{b} \wedge \bar{c}) \vee (b \wedge \bar{a} \wedge \bar{c}) \vee (c \wedge \bar{a} \wedge \bar{b})$
12. $f = (a \wedge b) \vee (a \wedge c) \vee (b \wedge c) \vee (\bar{a} \wedge \bar{b} \wedge \bar{c})$
13. $f = (a \wedge b \wedge c) \vee (\bar{a} \wedge \bar{b}) \vee (\bar{a} \wedge \bar{c}) \vee (\bar{b} \wedge \bar{c})$
14. $f = (a \wedge \bar{b}) \vee (a \wedge \bar{c}) \vee (\bar{b} \wedge \bar{c}) \vee (b \wedge c \wedge \bar{a})$
15. $f = (b \wedge \bar{a}) \vee (b \wedge \bar{c}) \vee (\bar{a} \wedge \bar{c}) \vee (a \wedge c \wedge \bar{b})$
16. $f = (b \wedge c) \vee (b \wedge \bar{a}) \vee (c \wedge \bar{a}) \vee (a \wedge \bar{b} \wedge \bar{c})$
17. $f = (a \wedge \bar{b}) \vee (a \wedge \bar{c}) \vee (\bar{b} \wedge \bar{c}) \vee (b \wedge c \wedge \bar{a})$

18. $f = (a \wedge b) \vee (a \wedge \bar{c}) \vee (b \wedge \bar{c}) \vee (c \wedge \bar{a} \wedge \bar{b})$

19. $f = (c \wedge \bar{a}) \vee (c \wedge \bar{b}) \vee (\bar{a} \wedge \bar{b}) \vee (a \wedge b \wedge \bar{c})$

20. $f = (a \wedge c) \vee (a \wedge \bar{b}) \vee (c \wedge \bar{b}) \vee (b \wedge \bar{a} \wedge \bar{c})$

21. $f = (a \wedge b \wedge \bar{c}) \vee (a \wedge c \wedge \bar{b}) \vee (b \wedge c \wedge \bar{a}) \vee (\bar{a} \wedge \bar{b} \wedge \bar{c})$

22. $f = (c \wedge \bar{a}) \vee (c \wedge \bar{b}) \vee (\bar{a} \wedge \bar{b}) \vee (a \wedge b \wedge \bar{c})$

23. $f = (a \wedge b \wedge c) \vee (a \wedge \bar{b} \wedge \bar{c}) \vee (b \wedge \bar{a} \wedge \bar{c}) \vee (c \wedge \bar{a} \wedge \bar{b})$

24. $f = (c \wedge \bar{a}) \vee (c \wedge \bar{b}) \vee (\bar{a} \wedge \bar{b}) \vee (a \wedge b \wedge \bar{c})$

25. $f = (b \wedge c) \vee (b \wedge \bar{a}) \vee (c \wedge \bar{a}) \vee (a \wedge \bar{b} \wedge \bar{c})$

26. $f = (b \wedge \bar{a}) \vee (b \wedge \bar{c}) \vee (\bar{a} \wedge \bar{c}) \vee (a \wedge c \wedge \bar{b})$

27. $f = (a \wedge b \wedge c) \vee (\bar{a} \wedge \bar{b}) \vee (\bar{a} \wedge \bar{c}) \vee (\bar{b} \wedge \bar{c})$

28. $f = (b \wedge \bar{a}) \vee (b \wedge \bar{c}) \vee (\bar{a} \wedge \bar{c}) \vee (a \wedge c \wedge \bar{b})$

29. $f = (b \wedge c) \vee (b \wedge \bar{a}) \vee (c \wedge \bar{a}) \vee (a \wedge \bar{b} \wedge \bar{c})$

30. $f = (a \wedge b) \vee (a \wedge c) \vee (b \wedge c) \vee (\bar{a} \wedge \bar{b} \wedge \bar{c})$

15.2 Задание 2

Таблица 5.1. Таблица преобразования, вариант 1

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	0	0
0	0	1	1	1	0
0	1	0	1	0	0
0	1	1	0	1	0
1	0	0	0	0	1
1	0	1	0	1	1
1	1	0	1	0	1
1	1	1	1	1	1

Таблица 5.2. Таблица преобразования, вариант 2

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	1	0
0	0	1	0	0	0
0	1	0	1	0	0
0	1	1	0	1	1
1	0	0	1	1	1
1	0	1	0	1	0
1	1	0	1	0	1
1	1	1	0	0	1

Таблица 5.3. Таблица преобразования, вариант 3

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	1	1
0	0	1	1	1	0
0	1	0	1	0	0
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	0	0	0
1	1	0	0	1	0
1	1	1	1	0	1

Таблица 5.4. Таблица преобразования, вариант 4

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	1	1
0	0	1	0	1	1

A_2	A_1	A_0	B_2	B_1	B_0
0	1	0	0	0	0
0	1	1	1	1	0
1	0	0	0	1	0
1	0	1	1	0	0
1	1	0	1	0	1
1	1	1	0	0	1

Таблица 5.5. Таблица преобразования, вариант 5

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	1	0
0	0	1	1	0	1
0	1	0	1	0	0
0	1	1	0	0	1
1	0	0	0	0	0
1	0	1	1	1	1
1	1	0	0	1	1
1	1	1	0	1	0

Таблица 5.6. Таблица преобразования, вариант 6

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	0	0
0	0	1	1	0	1
0	1	0	0	0	0
0	1	1	0	1	0
1	0	0	0	1	1

A_2	A_1	A_0	B_2	B_1	B_0
1	0	1	0	0	1
1	1	0	1	1	1
1	1	1	1	1	0

Таблица 5.7. Таблица преобразования, вариант 7

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	1	0
0	0	1	1	0	1
0	1	0	0	1	0
0	1	1	1	1	1
1	0	0	0	0	0
1	0	1	0	0	1
1	1	0	0	1	1
1	1	1	1	0	0

Таблица 5.8. Таблица преобразования, вариант 8

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	1	1
0	0	1	1	0	0
0	1	0	1	1	1
0	1	1	0	0	0
1	0	0	0	1	0
1	0	1	0	0	1
1	1	0	1	1	0
1	1	1	1	0	1

Таблица 5.9. Таблица преобразования, вариант 9

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	1	0
0	0	1	0	0	1
0	1	0	1	1	1
0	1	1	0	0	0
1	0	0	1	0	1
1	0	1	1	0	0
1	1	0	0	1	1
1	1	1	1	1	0

Таблица 5.10. Таблица преобразования, вариант 10

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	1	0
0	0	1	0	1	1
0	1	0	1	0	1
0	1	1	0	0	0
1	0	0	0	1	0
1	0	1	1	0	0
1	1	0	1	1	1
1	1	1	0	0	1

Таблица 5.11. Таблица преобразования, вариант 11

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	0	1
0	0	1	1	1	0

A_2	A_1	A_0	B_2	B_1	B_0
0	1	0	1	0	0
0	1	1	0	0	1
1	0	0	0	1	0
1	0	1	1	1	1
1	1	0	0	1	1
1	1	1	0	0	0

Таблица 5.12. Таблица преобразования, вариант 12

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	0	1
0	0	1	1	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	0	0	0
1	0	1	1	0	1
1	1	0	0	1	0
1	1	1	1	1	1

Таблица 5.13. Таблица преобразования, вариант 13

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	1	0	1
0	1	1	1	1	1
1	0	0	1	1	0

A_2	A_1	A_0	B_2	B_1	B_0
1	0	1	0	1	1
1	1	0	0	1	0
1	1	1	1	0	0

Таблица 5.14. Таблица преобразования, вариант 14

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	1	1	1
0	1	1	1	1	0
1	0	0	0	1	0
1	0	1	1	0	1
1	1	0	0	1	1
1	1	1	0	0	0

Таблица 5.15. Таблица преобразования, вариант 15

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	0	1
0	0	1	0	1	1
0	1	0	1	0	1
0	1	1	0	0	0
1	0	0	1	1	1
1	0	1	1	0	0
1	1	0	0	1	0
1	1	1	1	1	0

Таблица 5.16. Таблица преобразования, вариант 16

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	1	0
0	0	1	0	0	1
0	1	0	1	0	1
0	1	1	1	1	0
1	0	0	1	1	1
1	0	1	0	0	0
1	1	0	0	1	1
1	1	1	1	0	0

Таблица 5.17. Таблица преобразования, вариант 17

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	1	1	1
0	1	1	1	0	0
1	0	0	0	0	0
1	0	1	1	1	0
1	1	0	1	0	1
1	1	1	0	1	1

Таблица 5.18. Таблица преобразования, вариант 18

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	0	1
0	0	1	1	0	0

A_2	A_1	A_0	B_2	B_1	B_0
0	1	0	1	1	1
0	1	1	1	1	0
1	0	0	0	1	1
1	0	1	0	0	0
1	1	0	0	1	0
1	1	1	0	0	1

Таблица 5.19. Таблица преобразования, вариант 19

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	0	1
0	0	1	0	0	0
0	1	0	0	1	0
0	1	1	0	0	1
1	0	0	0	1	1
1	0	1	1	0	0
1	1	0	1	1	0
1	1	1	1	1	1

Таблица 5.20. Таблица преобразования, вариант 20

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	1	1
0	0	1	1	0	1
0	1	0	0	0	1
0	1	1	0	0	0
1	0	0	1	1	1

A_2	A_1	A_0	B_2	B_1	B_0
1	0	1	1	0	0
1	1	0	0	1	0
1	1	1	1	1	0

Таблица 5.21. Таблица преобразования, вариант 21

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	1	1
0	0	1	1	0	1
0	1	0	1	1	0
0	1	1	0	1	1
1	0	0	0	0	1
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	0	0	0

Таблица 5.22. Таблица преобразования, вариант 22

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	1	0
0	0	1	0	1	1
0	1	0	0	1	0
0	1	1	1	0	0
1	0	0	0	0	1
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	1	0	1

Таблица 5.23. Таблица преобразования, вариант 23

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	1	0
0	0	1	1	0	0
0	1	0	0	1	1
0	1	1	1	0	1
1	0	0	0	1	0
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	0	0	1

Таблица 5.24. Таблица преобразования, вариант 24

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	0	0
0	0	1	1	1	0
0	1	0	1	0	1
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	0	1	1
1	1	0	1	0	0
1	1	1	0	1	0

Таблица 5.25. Таблица преобразования, вариант 25

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	0	1
0	0	1	1	1	0

A_2	A_1	A_0	B_2	B_1	B_0
0	1	0	0	0	0
0	1	1	0	1	1
1	0	0	0	0	1
1	0	1	0	1	0
1	1	0	1	1	1
1	1	1	1	0	0

Таблица 5.26. Таблица преобразования, вариант 26

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	1	1
1	1	0	0	1	1
1	1	1	0	1	0

Таблица 5.27. Таблица преобразования, вариант 27

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	0	0
0	0	1	1	1	0
0	1	0	1	0	1
0	1	1	0	1	1
1	0	0	1	0	0

A_2	A_1	A_0	B_2	B_1	B_0
1	0	1	1	1	1
1	1	0	0	1	0
1	1	1	0	0	1

Таблица 5.28. Таблица преобразования, вариант 28

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	0	1
0	0	1	0	0	0
0	1	0	1	1	0
0	1	1	1	0	0
1	0	0	1	1	1
1	0	1	0	1	0
1	1	0	1	0	1
1	1	1	0	1	1

Таблица 5.29. Таблица преобразования, вариант 29

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	1	1	0
0	0	1	0	0	0
0	1	0	0	1	1
0	1	1	1	1	1
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	1	0	1
1	1	1	0	1	0

Таблица 5.30. Таблица преобразования, вариант 30

A_2	A_1	A_0	B_2	B_1	B_0
0	0	0	0	0	1
0	0	1	1	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	0	0	0
1	0	1	1	1	1
1	1	0	1	0	0
1	1	1	1	1	0

16 Вопросы для самоконтроля

1. Что такое LUT?
2. Как реализовывается программируемая логика?
3. На основе чего можно создать преобразователь кодов?

17 Список литературы для самостоятельного изучения

1. Хеннесси, Дж. Л., Паттерсон, Д. А. Компьютерная архитектура: количественный подход. — 5-е изд. — М.: Вильямс, 2016. — 944 с.

2. Таненбаум, Э. Архитектура компьютера. Структурный подход. — 5-е изд. — СПб.: Питер, 2013. — 832 с.

3. Архитектура вычислительных систем [Электронный ресурс]: учебное пособие – Эл. изд. - Электрон. текстовые дан. (1 файл pdf: 77 с.). - Грейбо С.В., Новосёлова Т.Е., Пронькин Н.Н., Семёнычева И.Ф. 2019

4. Глушков В.М. Синтез цифровых автоматов. –М., Физматгиз, 1962г., -476с.

5. Поспелов Д.А. Логические методы анализа и синтеза схем. М.: Энергия, 1974.-228 с.