1. **Какие логические функции выполняет дешифратор?**

Дешифратор — это комбинационное устройство, которое преобразует **n-разрядный двоичный**, **троичный** или **k-ичный код** в **k-ичный одноединичный код**. В выходном коде всегда присутствует только **один активный бит**, а остальные биты неактивны. Активный бит может быть равен 1 или 0 в зависимости от конкретной реализации дешифратора. [Дешифраторы выполняют различные **логические функции** в цифровой электронике1](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%88%D0%B8%D1%84%D1%80%D0%B0%D1%82%D0%BE%D1%80)[2](https://bing.com/search?q=). Например, они используются для преобразования кодов управления различными устройствами.

1. **Каково назначение входов управления в дешифраторе. Как влияет сигнал управления на выходные функции дешифратора?**

В дешифраторе входы управления играют важную роль. Вот как это работает:

1. **Назначение входов управления в дешифраторе**:
   * Входы управления в дешифраторе позволяют контролировать его функционирование.
   * Они определяют, когда дешифратор будет активирован или деактивирован.
   * Входы управления могут использоваться для каскадирования (увеличения разрядности) дешифраторов или при параллельной работе нескольких схем на общие выходные линии.
2. **Влияние сигнала управления на выходные функции дешифратора**:
   * Если на вход дешифратора подать двоичный код, то на выходе, соответствующем этому коду, будет сигнал высокого уровня (или на всех выходах от младшего до заданного).
   * [Если при этом на управляющий вход подать запрещающий импульс, то состояние выхода дешифратора не изменится даже при изменении сигнала на его входе1](https://otvet.mail.ru/question/34789080)[2](https://otvet.mail.ru/question/71261288).
3. **Какие дополнительные логические элементы необходимы для реализации логических функций n аргументов на основе дешифратора с прямыми выходами? А с инверсными?**

Для реализации логических функций с использованием дешифраторов с **прямыми выходами** не требуется дополнительных логических элементов. Прямые выходы дешифратора уже предоставляют активный сигнал, соответствующий входному коду.

Однако, если используются дешифраторы с **инверсными выходами**, то для получения логических функций n аргументов потребуется дополнительные инверторы (NOT-гейты). [Эти инверторы позволят инвертировать сигналы на выходах дешифратора, чтобы получить нужные логические значения1](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%88%D0%B8%D1%84%D1%80%D0%B0%D1%82%D0%BE%D1%80)[2](https://studbooks.net/2342739/tehnika/polnyy_deshifrator_pryamymi_vyhodami). Например, если выход дешифратора с инверсными выходами имеет активный уровень логического нуля, то инвертированный выход будет иметь активный уровень логической единицы и наоборот.

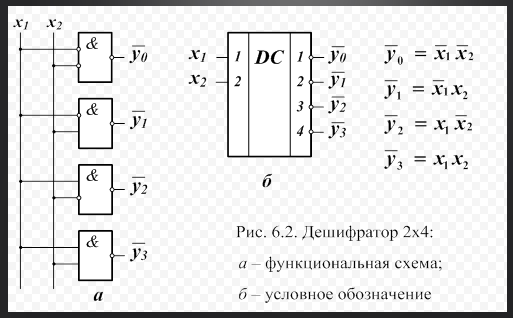
1. **Как выглядит схема дешифратора 2х4, выполненная в базисе И, ИЛИ, НЕ? Обозначьте входы дешифратора А, В, выходы Y0, Y1, Y2, Y3. Сколько элементов каждого типа для этого требуется?**

**4.1 Как выглядит схема дешифратора 2х4, выполненная в базисе И? Обозначьте входы дешифратора А, В, выходы Y0, Y1, Y2, Y3. Сколько элементов каждого типа для этого требуется?**

Схема дешифратора 2х4, выполненного в базисе И, представляет собой устройство, которое преобразует двухбитовый вход (представленный сигналами A и B) в один из четырех выходов (Y0, Y1, Y2 или Y3). Давайте рассмотрим его подробнее.

1. **Входы дешифратора**:
   * Вход A и В представляют два бита входных данных.
   * Вход A соответствует младшему биту, а В – старшему биту.
2. **Выходы дешифратора**:
   * Y0, Y1, Y2 и Y3 – это четыре возможных выхода дешифратора.
   * Каждый выход активируется при определенной комбинации входных сигналов. Например:
     + Если A=0 и B=0, то активируется Y0.
     + Если A=0 и B=1, то активируется Y1.
     + Если A=1 и B=0, то активируется Y2.
     + Если A=1 и B=1, то активируется Y3.
3. **Количество элементов**:
   * Для реализации дешифратора 2х4 в базисе И, нам потребуется **четыре логических элемента типа “4И”**.
   * Логические элементы “ИЛИ” не используются, так как каждый выход активируется только одним входом.

Таким образом, схема дешифратора 2х4 в базисе И состоит из четырех И-элементов и выполняет функцию преобразования двухбитового кода в один из четырех выходов.

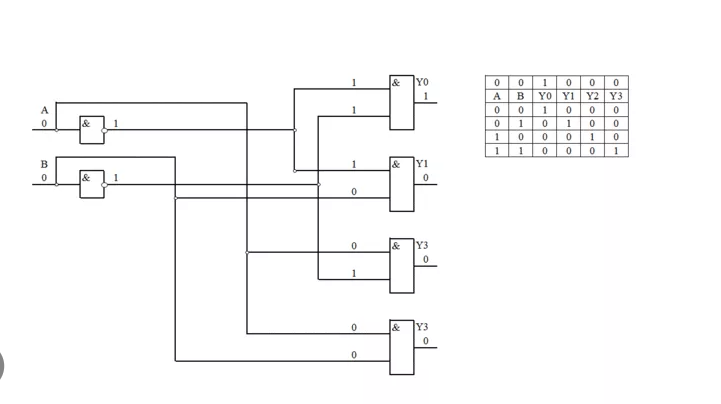


**4.2 Как выглядит схема дешифратора 2х4, выполненная в базисе ИЛИ? Обозначьте входы дешифратора А, В, выходы Y0, Y1, Y2, Y3. Сколько элементов каждого типа для этого требуется?**

Схема дешифратора 2х4, выполненного в базисе ИЛИ, представляет собой устройство, которое преобразует двухбитовый вход (представленный сигналами **A** и **B**) в один из четырех выходов (**Y0**, **Y1**, **Y2** или **Y3**). Давайте рассмотрим его подробнее:

1. **Входы дешифратора**:
   * Вход **A** и **B** представляют два бита входных данных.
   * Вход **A** соответствует младшему биту, а **B** – старшему биту.
2. **Выходы дешифратора**:
   * **Y0**, **Y1**, **Y2** и **Y3** – это четыре возможных выхода дешифратора.
   * Каждый выход активируется при определенной комбинации входных сигналов. Например:
     + Если **A=0** и **B=0**, то активируется **Y0**.
     + Если **A=0** и **B=1**, то активируется **Y1**.
     + Если **A=1** и **B=0**, то активируется **Y2**.
     + Если **A=1** и **B=1**, то активируется **Y3**.
3. **Количество элементов**:
   * Для реализации дешифратора 2х4 в базисе ИЛИ, нам потребуется **четыре логических элемента типа “4И”**.
   * Логические элементы “ИЛИ” не используются, так как каждый выход активируется только одним входом.

Таким образом, схема дешифратора 2х4 в базисе ИЛИ состоит из четырех И-элементов и выполняет функцию преобразования двухбитового кода в один из четырех выходов.

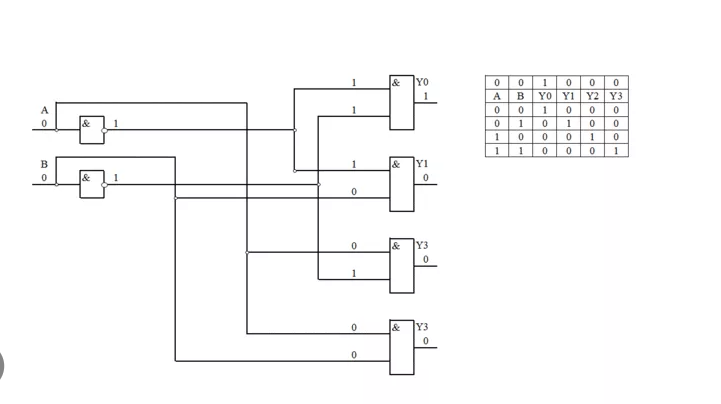


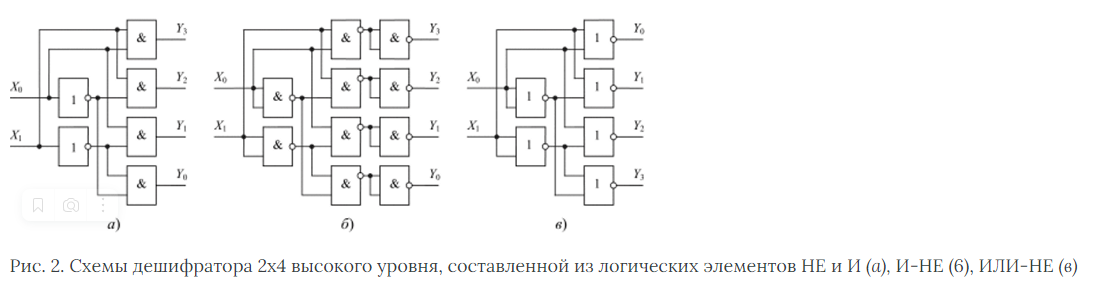
**4.3 Как выглядит схема дешифратора 2х4, выполненная в базисе НЕ? Обозначьте входы дешифратора А, В, выходы Y0, Y1, Y2, Y3. Сколько элементов каждого типа для этого требуется?**

Схема дешифратора 2х4, выполненного в базисе **НЕ**, представляет собой устройство, которое преобразует двухбитовый вход (представленный сигналами **A** и **B**) в один из четырех выходов (**Y0**, **Y1**, **Y2** или **Y3**). Давайте рассмотрим его подробнее:

1. **Входы дешифратора**:
   * Вход **A** и **B** представляют два бита входных данных.
   * Вход **A** соответствует младшему биту, а **B** – старшему биту.
2. **Выходы дешифратора**:
   * **Y0**, **Y1**, **Y2** и **Y3** – это четыре возможных выхода дешифратора.
   * Каждый выход активируется при определенной комбинации входных сигналов. Например:
     + Если **A=0** и **B=0**, то активируется **Y0**.
     + Если **A=0** и **B=1**, то активируется **Y1**.
     + Если **A=1** и **B=0**, то активируется **Y2**.
     + Если **A=1** и **B=1**, то активируется **Y3**.
3. **Количество элементов**:
   * Для реализации дешифратора 2х4 в базисе **НЕ**, нам потребуется **четыре логических элемента типа “2И-НЕ”**.
   * Логические элементы “ИЛИ” не используются, так как каждый выход активируется только одним входом.

Таким образом, схема дешифратора 2х4 в базисе **НЕ** состоит из четырех 2И-НЕ элементов и выполняет функцию преобразования двухбитового кода в один из четырех выходов.

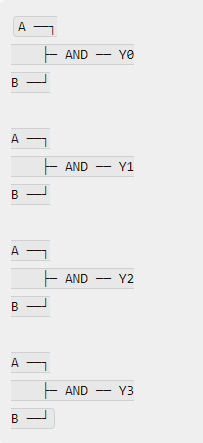




1. **Как надо видоизменить схему дешифратора 2х4 в предыдущем случае, чтобы оснастить еѐ прямым управляющим входом? Инверсным? Обозначьте входы дешифратора А, В, управляющий вход G, выходы Y0, Y1, Y2, Y3.  
   5.1 Как надо видоизменить схему дешифратора 2х4 в предыдущем случае, чтобы оснастить еѐ прямым управляющим входом? Инверсным? Обозначьте входы дешифратора А, В, управляющий вход G, выходы Y0, Y1, Y2, Y3.**

Дешифратор — это комбинационное устройство, которое преобразует входной код в некоторое десятичное число или другой код. Давайте рассмотрим схему дешифратора 2×4, выполненную в базисе И (AND).

1. **Схема дешифратора 2×4 в базисе И (AND)**:
   * Входы дешифратора обозначим как **A** и **B**.
   * Выходы дешифратора обозначим как **Y0**, **Y1**, **Y2**, и **Y3**.
   * Для преобразования десятичных чисел от 0 до 3 в двоичный код, нам понадобится 2 входа (A и B) и 4 выхода (Y0, Y1, Y2, Y3).
   * Схема дешифратора будет выглядеть следующим образом:



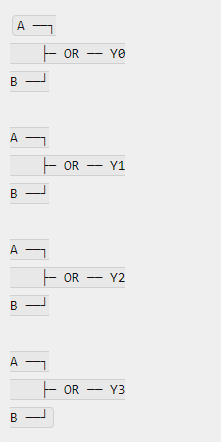
1. **Количество элементов**:
   * Для данной схемы дешифратора 2×4 в базисе И, нам потребуется **4 элемента И** (по одному для каждого выхода).
2. **Прямой управляющий вход**:
   * Чтобы оснастить дешифратор прямым управляющим входом, мы можем добавить дополнительный вход, который будет активировать или деактивировать дешифратор.
   * Обозначим этот управляющий вход как **G**.
   * Если **G** равно 1 (логическая единица), дешифратор будет активен и выполнять свою функцию.
   * Если **G** равно 0 (логический ноль), дешифратор будет отключен.
3. **Инверсный управляющий вход**:
   * Чтобы оснастить дешифратор инверсным управляющим входом, мы можем добавить дополнительный вход, который будет инвертировать сигнал.
   * Обозначим этот инверсный управляющий вход как **G̅** (читается как “G бар”).
   * Если **G̅** равно 1, дешифратор будет активен.
   * Если **G̅** равно 0, дешифратор будет отключен.

Таким образом, с управляющим входом **G** или **G̅**, мы можем контролировать работу дешифратора в зависимости от нашего выбора.

**5.2 Как надо видоизменить схему дешифратора 2х4 в предыдущем случае, чтобы оснастить еѐ прямым управляющим входом? Инверсным? Обозначьте входы дешифратора А, В, управляющий вход G, выходы Y0, Y1, Y2, Y3.**

Давайте рассмотрим схему дешифратора 2×4, выполненную в базисе ИЛИ (OR).

1. **Схема дешифратора 2×4 в базисе ИЛИ (OR)**:
   * Входы дешифратора обозначим как **A** и **B**.
   * Выходы дешифратора обозначим как **Y0**, **Y1**, **Y2** и **Y3**.
   * Для преобразования десятичных чисел от 0 до 3 в двоичный код, нам также потребуется 2 входа (A и B) и 4 выхода (Y0, Y1, Y2, Y3).
   * Схема дешифратора будет выглядеть следующим образом:



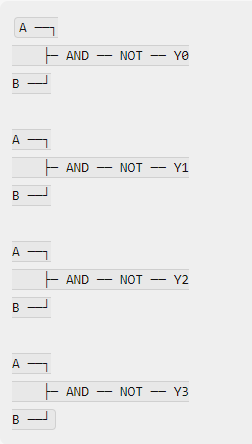
1. **Количество элементов**:
   * Для данной схемы дешифратора 2×4 в базисе ИЛИ, нам также потребуется **4 элемента ИЛИ** (по одному для каждого выхода).
2. **Прямой управляющий вход**:
   * Чтобы оснастить дешифратор прямым управляющим входом, мы можем добавить дополнительный вход, который будет активировать или деактивировать дешифратор.
   * Обозначим этот управляющий вход как **G**.
   * Если **G** равно 1 (логическая единица), дешифратор будет активен и выполнять свою функцию.
   * Если **G** равно 0 (логический ноль), дешифратор будет отключен.
3. **Инверсный управляющий вход**:
   * Чтобы оснастить дешифратор инверсным управляющим входом, мы можем добавить дополнительный вход, который будет инвертировать сигнал.
   * Обозначим этот инверсный управляющий вход как **G̅** (читается как “G бар”).
   * Если **G̅** равно 1, дешифратор будет активен.
   * Если **G̅** равно 0, дешифратор будет отключен.

Таким образом, с управляющим входом **G** или **G̅**, мы можем контролировать работу дешифратора в зависимости от нашего выбора.

**5.3 Как надо видоизменить схему дешифратора 2х4 в предыдущем случае, чтобы оснастить еѐ прямым управляющим входом? Инверсным? Обозначьте входы дешифратора А, В, управляющий вход G, выходы Y0, Y1, Y2, Y3.**

Давайте рассмотрим схему дешифратора 2×4, выполненную в базисе **И-НЕ** (AND-NOT).

1. **Схема дешифратора 2×4 в базисе И-НЕ**:
   * Входы дешифратора обозначим как **A** и **B**.
   * Выходы дешифратора обозначим как **Y0**, **Y1**, **Y2** и **Y3**.
   * Для преобразования десятичных чисел от 0 до 3 в двоичный код, нам также потребуется 2 входа (A и B) и 4 выхода (Y0, Y1, Y2, Y3).
   * Схема дешифратора будет выглядеть следующим образом:



1. **Количество элементов**:
   * Для данной схемы дешифратора 2×4 в базисе И-НЕ, нам также потребуется **4 элемента И** (по одному для каждого выхода) и **4 элемента НЕ** (по одному для каждого выхода).
2. **Прямой управляющий вход**:
   * Чтобы оснастить дешифратор прямым управляющим входом, мы можем добавить дополнительный вход, который будет активировать или деактивировать дешифратор.
   * Обозначим этот управляющий вход как **G**.
   * Если **G** равно 1 (логическая единица), дешифратор будет активен и выполнять свою функцию.
   * Если **G** равно 0 (логический ноль), дешифратор будет отключен.
3. **Инверсный управляющий вход**:
   * Чтобы оснастить дешифратор инверсным управляющим входом, мы можем добавить дополнительный вход, который будет инвертировать сигнал.
   * Обозначим этот инверсный управляющий вход как **G̅** (читается как “G бар”).
   * Если **G̅** равно 1, дешифратор будет активен.
   * Если **G̅** равно 0, дешифратор будет отключен.

Таким образом, с управляющим входом **G** или **G̅**, мы можем контролировать работу дешифратора в зависимости от нашего выбора.

1. **Как из двух дешифраторов 2х4 сделать один дешифратор 3х8?**

Для создания дешифратора 3х8 из двух дешифраторов 2х4 можно воспользоваться **каскадированием**. Вот как это можно сделать:

1. Подайте на входы обоих дешифраторов одинаковые адресные биты (например, A и B).
2. Подайте на вход разрешения (EN) первого дешифратора сигнал “0”, а на вход разрешения второго дешифратора – инвертированный сигнал “0”.
3. Выходы первого дешифратора будут представлять первые 4 выхода дешифратора 3х8, а выходы второго дешифратора – следующие 4 выхода.

[Таким образом, вы получите дешифратор 3х8, который будет иметь 8 выходов для адресов от 000 до 1111](https://www.cyberforum.ru/programmable-logic/thread1101853.html)[2](https://big-soviet.ru/remont/7/kak-iz-dvuh-deshifratorov-2h4-sdelat-odin)

1. **Как на основе нескольких дешифраторов 2х4 с управляющим входом сделать дешифратор 4х16? Сколько дешифраторов 2х4 потребуется для решения этой задачи, если не использовать другие элементы?**

Для создания дешифратора 4х16 из дешифраторов 2х4 с управляющим входом можно воспользоваться **каскадированием**. Вот как это можно сделать:

1. Сначала соедините два дешифратора 2х4 так, чтобы управляющий вход одного из них был подключен к управляющему входу другого. Это означает, что оба дешифратора будут активироваться одновременно.
2. Подайте на входы обоих дешифраторов адресные биты (например, A, B, C, D).
3. Подайте на управляющий вход первого дешифратора сигнал “0”, а на управляющий вход второго дешифратора – инвертированный сигнал “0”.
4. Выходы первого дешифратора будут представлять первые 8 выходов дешифратора 4х16, а выходы второго дешифратора – следующие 8 выходов.

Таким образом, вы получите дешифратор 4х16, который будет иметь 16 выходов для адресов от 0000 до 1111. [Для решения этой задачи потребуется **два дешифратора 2х4**](https://pandia.ru/text/80/219/32603.php)[1](https://pandia.ru/text/80/219/32603.php)[2](https://intuit.ru/studies/courses/685/541/lecture/12174).

1. **Как на основе дешифратора 2x4 сделать схему, фиксирующую совпадение двух бит (А = В = 1, А = В = 0) и реализующую функцию** 

**Дешифратор** - это логическая комбинационная схема, которая имеет **n информационных входов** и **2n выходов**. Каждой комбинации логических уровней на входах будет соответствовать активный уровень на одном из 2^n выходов. Обычно n равно 2, 3 или 4.

Давайте представим, что у нас есть дешифратор 2x4. Это означает, что у него **2 информационных входа (A и B)** и **4 выхода**. Давайте обозначим выходы как Y0, Y1, Y2 и Y3.

Теперь давайте рассмотрим следующие комбинации логических уровней на входах A и B:

1. A=0, B=0: В этом случае на выходе Y0 будет активный уровень (например, 0).
2. A=0, B=1: В этом случае на выходе Y1 будет активный уровень (например, 0).
3. A=1, B=0: В этом случае на выходе Y2 будет активный уровень (например, 0).
4. A=1, B=1: В этом случае на выходе Y3 будет активный уровень (например, 0).

Теперь давайте рассмотрим функцию **F=(A*B) + (НЕ(A*B))**:

1. **(A\*B)**: Это произведение A и B. Если оба бита равны 1, то результат будет 1, иначе 0.
2. **(НЕ(A\*B))**: Это инверсия результата (A*B). Если (A*B) равно 1, то (НЕ(A\*B)) будет 0, и наоборот.

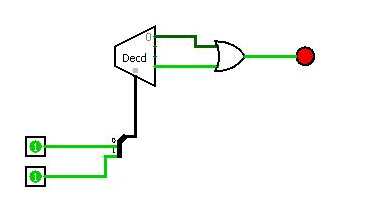
Теперь давайте объединим оба этих условия:

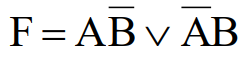
* Если **A=1 и B=1**, то (A*B) равно 1, и (НЕ(A*B)) равно 0. Таким образом, наша схема должна фиксировать совпадение двух бит, когда оба равны 1.
* Если **A=0 и B=0**, то (A*B) равно 0, и (НЕ(A*B)) равно 1. Таким образом, наша схема также должна фиксировать совпадение двух бит, когда оба равны 0.

Давайте создадим схему, которая будет выполнять оба эти условия:

1. Подключим информационные входы A и B к дешифратору 2x4.
2. Выход Y3 (который соответствует комбинации A=1, B=1) будет нашим первым выходом.
3. Выход Y0 (который соответствует комбинации A=0, B=0) будет нашим вторым выходом.

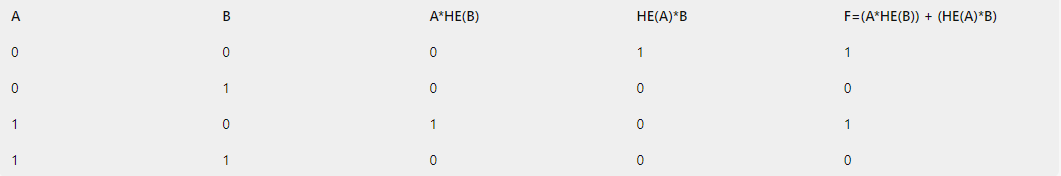
Таким образом, наша схема будет фиксировать совпадение двух бит (А=В=1 или А=В=0) и реализовывать функцию F=(A*B) + (НЕ(A*B)).



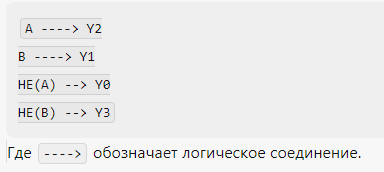
1. **Как на основе дешифратора сделать логическую схему, реализующую функцию**  ?

Для реализации функции **F=(A\*НЕ(B)) + (НЕ(A)\*B)**, мы можем воспользоваться дешифратором и другими логическими элементами. Давайте разберемся, как это можно сделать.

1. **Дешифратор** - это логическая схема, которая преобразует двоичный код в унарный, активируя только один из всех выходов. Для нашей задачи нам понадобится дешифратор с двумя информационными входами (A и B) и четырьмя выходами (Y0, Y1, Y2, Y3).
2. Давайте определим таблицу истинности для функции **F=(A\*НЕ(B)) + (НЕ(A)\*B)**:



1. Теперь давайте свяжем выходы дешифратора с логическими элементами:
   1. Выход Y0 (который соответствует комбинации A=0, B=0) будет подключен к входу НЕ(A).
   2. Выход Y1 (который соответствует комбинации A=0, B=1) будет подключен к входу B.
   3. Выход Y2 (который соответствует комбинации A=1, B=0) будет подключен к входу A.
   4. Выход Y3 (который соответствует комбинации A=1, B=1) будет подключен к входу НЕ(B).
2. Таким образом, наша логическая схема будет выглядеть следующим образом:



Эта схема реализует функцию **F=(A\*НЕ(B)) + (НЕ(A)\*B)**.

1. **Как можно использовать дешифратор в качестве демультиплексора?**

Дешифратор можно использовать в качестве **демультиплексора** (или **мультиплексора с обратным кодом**) путем подключения его в определенной конфигурации. Вот как это можно сделать:

1. **Дешифратор 2х4 как демультиплексор 1х4**:
   * Подайте адресные биты на входы дешифратора (например, A и B).
   * Подайте управляющий сигнал (EN) на вход разрешения дешифратора.
   * Выходы дешифратора будут представлять 4 различных сигнала, которые можно использовать для выбора одного из 4 входов.
2. **Дешифратор 3х8 как демультиплексор 1х8**:
   * Подайте адресные биты на входы дешифратора (например, A, B и C).
   * Подайте управляющий сигнал (EN) на вход разрешения дешифратора.
   * Выходы дешифратора будут представлять 8 различных сигналов, которые можно использовать для выбора одного из 8 входов.
3. **Дешифратор 4х16 как демультиплексор 1х16**:
   * Подайте адресные биты на входы дешифратора (например, A, B, C и D).
   * Подайте управляющий сигнал (EN) на вход разрешения дешифратора.
   * Выходы дешифратора будут представлять 16 различных сигналов, которые можно использовать для выбора одного из 16 входов.

Таким образом, дешифратор можно настроить для выбора определенного входа на основе адресных битов и управляющего сигнала. Это позволяет использовать дешифратор как демультиплексор для множества входов.