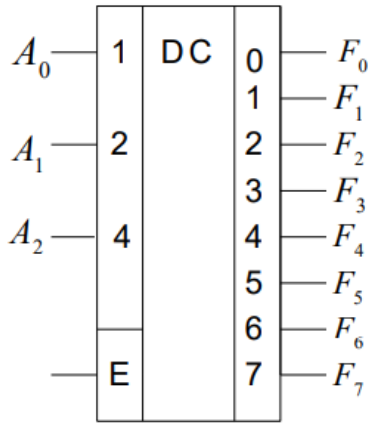
ЛР2. Дешифраторы

Калитвенцев Максим Павлович

Принцип действия линейного дешифратора



A2	A1	A0	F0	F1	F2	F3	F4	F5	F6	F7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Есть n адресных входов и N выходов. На адресные входы подается номер выхода, и на соответствующем выходе устанавливается высокий уровень сигнала (1).

К примеру, имеем 3 адресных входа, и подаем на них номер «5», т.е., 101. Тогда выход F5 = 1, все остальные выходы равны 0.

Принцип действия линейного дешифратора

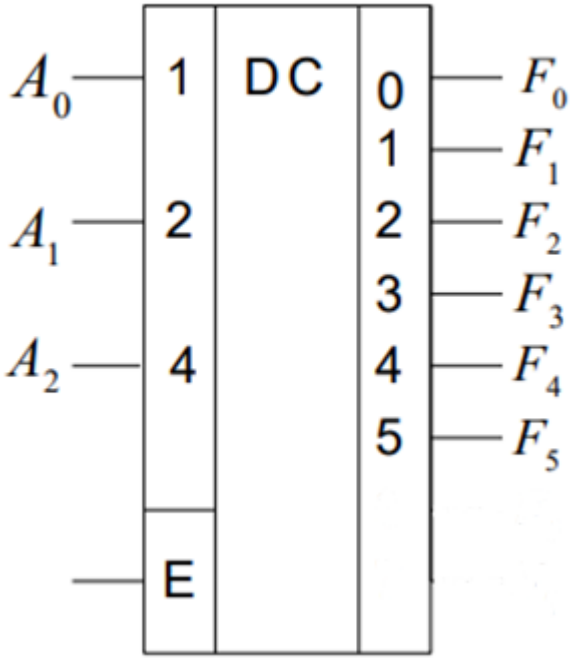
A2	A1	A0
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Адресные входы, как правило, подчиняются правилу старшего разряда, т.е., например, в трехвходовом дешифраторе вход A2 отвечает за 2-й (старший) разряд номера, A1 – за 1-й, A0 – за младший (нулевой).

Число «6», т.е., 110, на этих входах будет выглядеть так

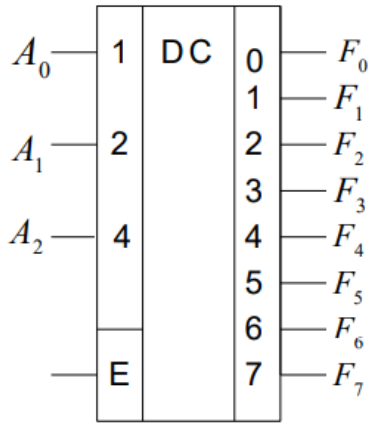
A2 = 1, A1 = 1, A0 = 0.

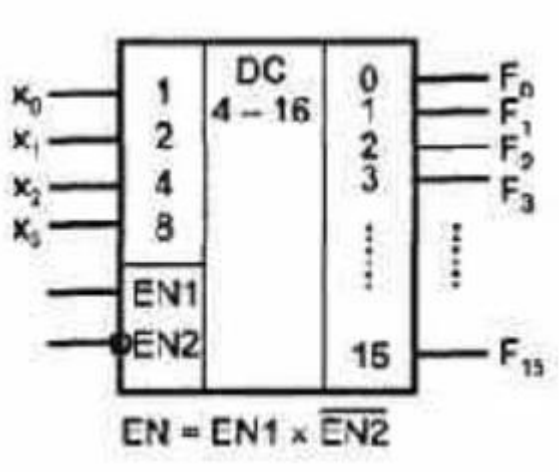
Неполный дешифратор



$N < n^2$, где N – количество выходов, n – количество адресных входов. «Лишние» адреса просто не вызывают реакции схемы.

Вход разрешения (стробирования)





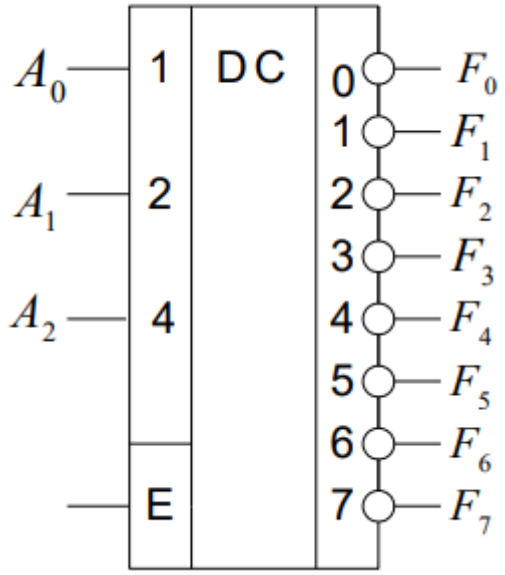
Выходы активны только пока вход EN = 1. Когда EN = 0, все выходы сбрасываются.

EN может быть инверсным, т.е., при !EN = 0, выходы активны, а при !EN = 0 нет.

EN может быть составным, и состоять из нескольких входов EN1, EN2 и т.д. Тогда EN = EN1 && EN2, т.е., чтобы выходы дешифратора начали работать, оба входа разрешения должны быть активны.

Замечание. Если EN неактивен, это не значит, что схема не работает в принципе. Внутренняя комбинационная логика продолжает дешифровку входных сигналов, результат просто не подается на выход.

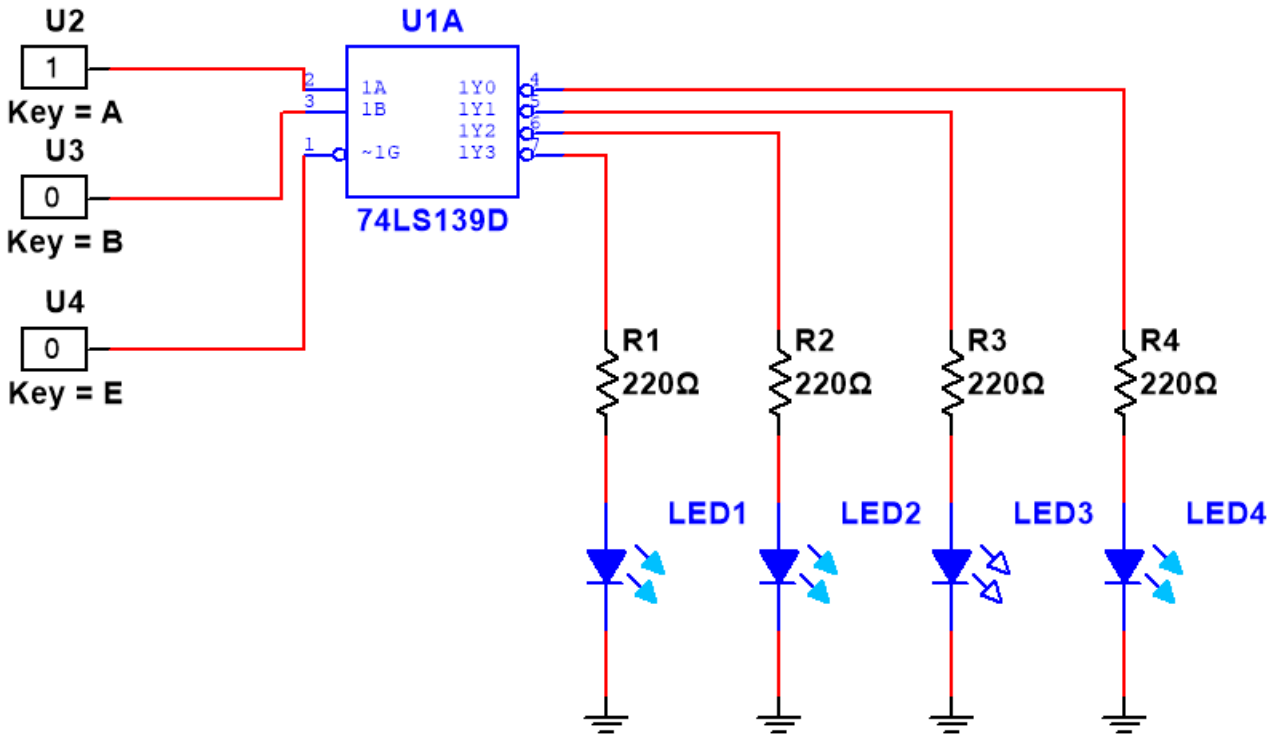
Инверсные выходы



В случае дешифратора с инверсными выходами, состояние выходов по умолчанию (неактивное) – это 1, а выход, выбранный через адресные входы, опускается в 0.

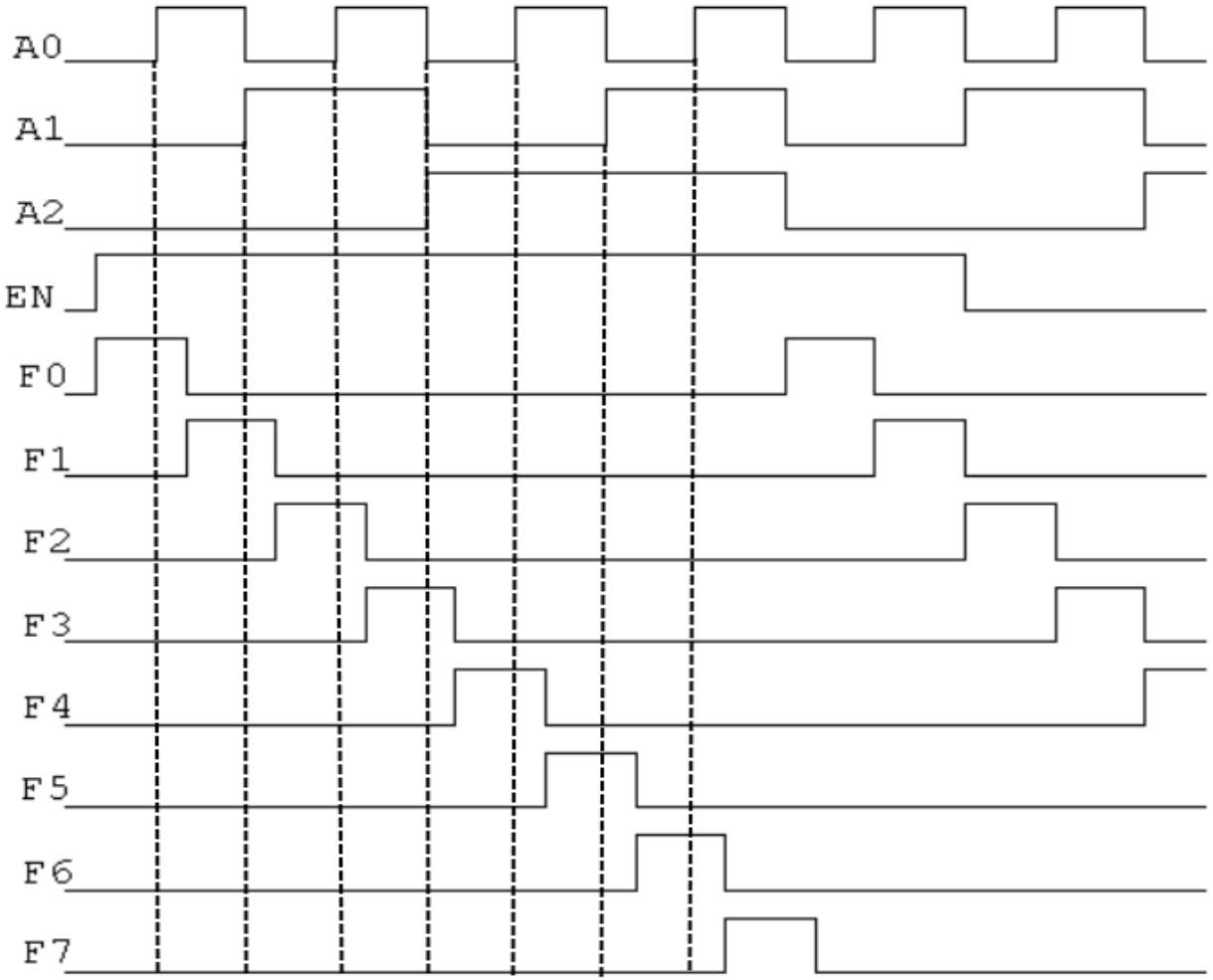
A2	A1	A0	F0	F1	F2	F3	F4	F5	F6	F7
0	0	0	0	1	1	1	1	1	1	1
0	0	1	1	0	1	1	1	1	1	1
0	1	0	1	1	0	1	1	1	1	1
0	1	1	1	1	1	0	1	1	1	1
1	0	0	1	1	1	1	0	1	1	1
1	0	1	1	1	1	1	1	0	1	1
1	1	0	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	0

Вот же круто, а зачем



- Передача сигналов к периферийным устройствам;
- Преобразование адресов ячеек памяти;
- Преобразование кодов операций в сигналы;
- И т.д.

Время задержки



Адресные и стробирующие сигналы влияют на выходы не сразу, а через некоторое время задержки, которое определяется количеством элементов И-НЕ, ИЛИ-НЕ, НЕ внутри схемы дешифратора. Каждый логический элемент на пути сигнала обеспечивает некоторую задержку по времени.

$$t_{зд.р.ср} = 2t_{зд.р.ср1} + t_{зд.р.ср2}$$

Здесь t(зд.р.ср.) – среднее время задержки сигнала от входа до выхода;

t(зд.р.ср.1) – среднее время задержки сигнала в инверторе (НЕ);

t(зд.р.ср.2) – среднее время задержки сигнала в конъюнкторе (И-НЕ).

Логические гонки

Из-за задержек сигналов и переходных процессов, на выходе схемы на какое-то время могут появиться ложные сигналы. Чтобы не допустить этого, применяется стробирование – сигнал EN не дает выходам дешифратора становиться активными, пока переходные процессы не закончатся.

В связи с этим, стробирующий сигнал EN должен подаваться с некоторой задержкой относительно адресных сигналов. Эта задержка дает переходным процессам спокойно пройти, и вычисляется по уже упомянутой формуле

$$t_{зд.р.ср} = 2t_{зд.р.ср1} + t_{зд.р.ср2}$$

Каскадные дешифраторы

Подразделяются на 2 вида – пирамидальные и ступенчатые. Пирамидальные в настоящее время не используются, т.к. обладают слишком высокими показателями задержки.

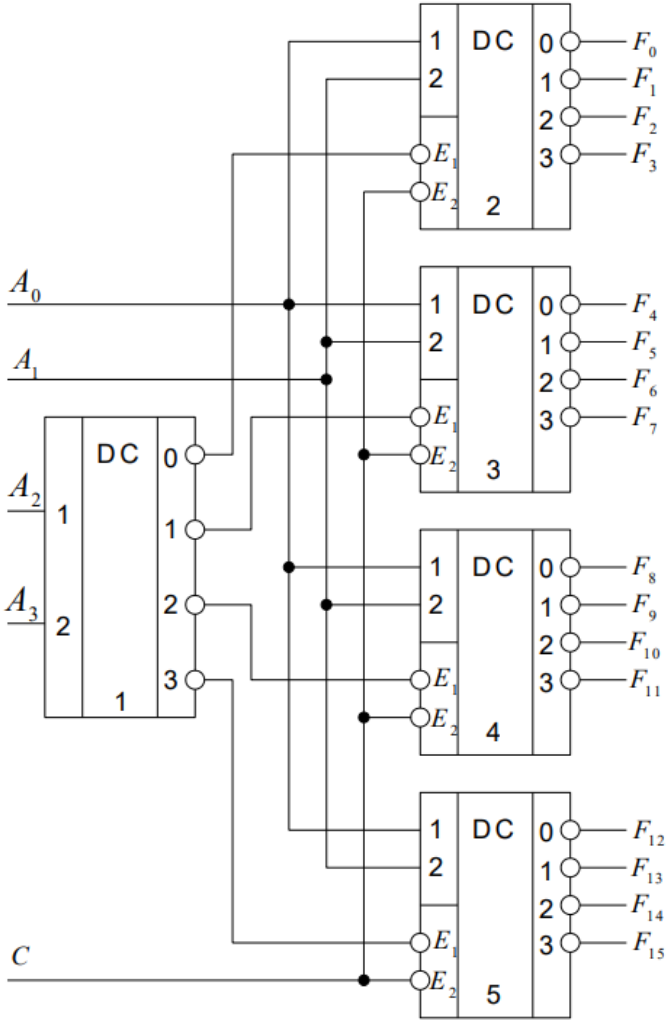
Ступенчатые получают путем сбора из нескольких линейных дешифраторов (наращивания).

Ступенчатые дешифраторы

1. Допустим, нам нужен дешифратор с n входов и N выходов, причем $\mathrm{N} = n^2$;
2. У нас есть некоторое количество дешифраторов с n1 входов и N1 выходов, причем n1 < n, N1 < N;
3. Определяем число каскадов K = $\frac{n}{n1}$;
4. Тогда количество простых дешифраторов, которое нам понадобится, равно $\frac{N}{N1}$ для выходного каскада, $\frac{N}{N1^2}$ для предвыходного и т.д., до входного каскада, где их будет $\frac{N}{N1^K}$.
   Тогда общее кол-во будет следующим
   $\frac{N}{N1} + \frac{N}{N1^2} + \ldots + \frac{N}{N1^K}$.
5. К адресным входам выходного каскада подключаются n1 младших разрядов адреса, к предвыходному – следующие n1 разрядов и т.д.
6. Выходы дешифраторов предвыходного каскада соединяются с входами стробирования дешифраторов выходного каскада, один выход – один вход (см. рисунок на следующем слайде). Точно так же предвыходной каскад принимает выходы стоящего за ним каскада и т.д.

Ступенчатые дешифраторы (пример)

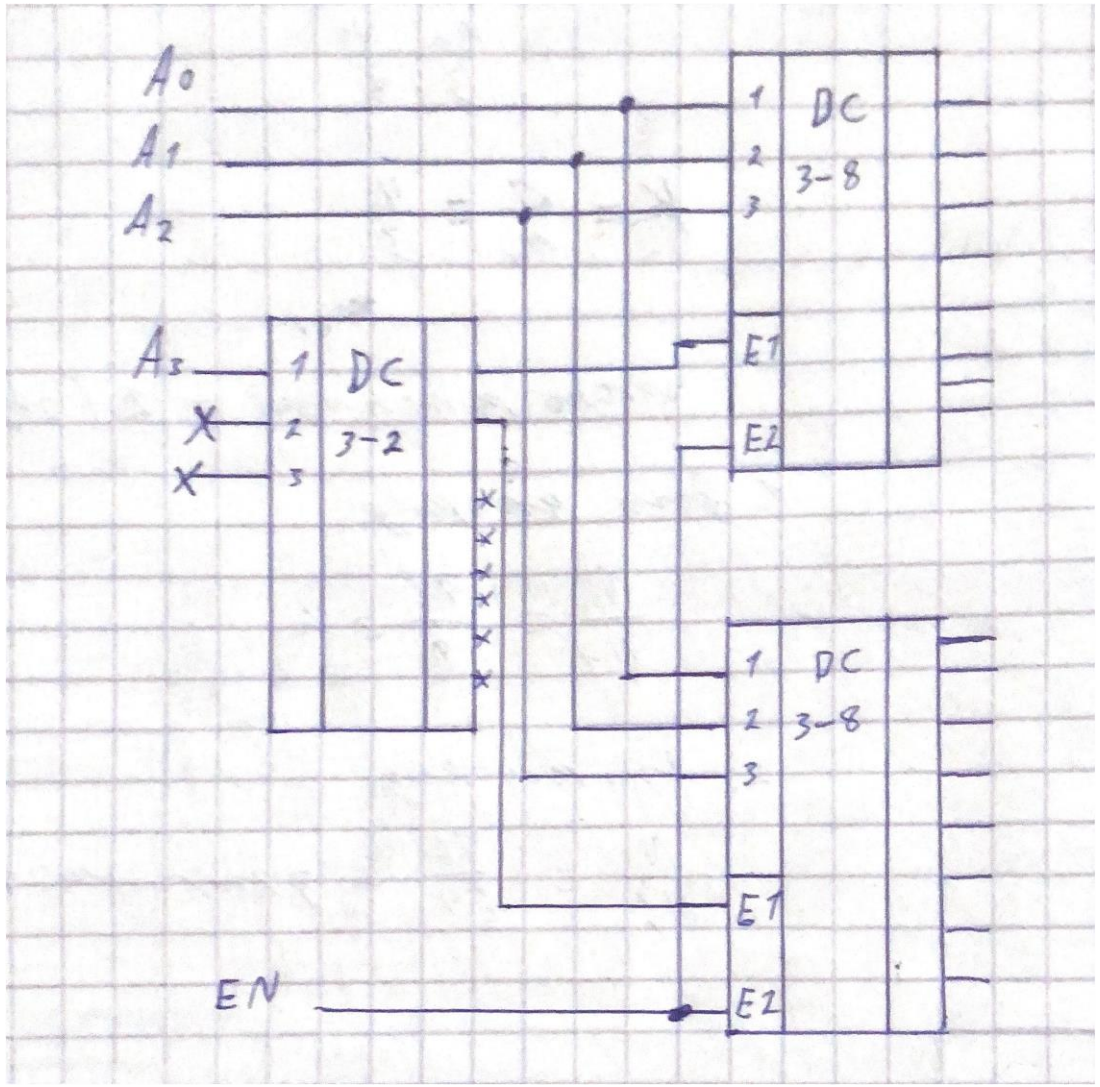
1. Нам нужен дешифратор 4-16 (4 адресных входа, 16 выходов), у нас есть несколько дешифраторов 2-4. Имеем N = 16, n = 4, N1 = 4, n1 = 2;
2. Тогда количество каскадов $\mathrm{K} = \frac{n}{n1} = \frac{4}{2} = 2$, т.е. выходной и предвыходной, здесь он же входной;
3. Для выходного каскада будет нужно $\frac{N}{N1} = \frac{16}{4} = 4$ дешифратора, для входного $\frac{N}{N1^2} = \frac{16}{4^2} = \frac{16}{16} = 1$. Итого, понадобится 5 дешифраторов 2-4;
4. 2 младших адресных входа A1, A0 подключаем к адресным входам выходного каскада, следующие 2 – во входной каскад;
5. Выходы дешифратора входного каскада подключаем к входам EN1 в выходном каскаде, один выход на один дешифратор.



Ступенчатые дешифраторы

Что делать в случае, когда К – дробное число?

1. Допустим, нам нужно собрать дешифратор 4-16 из дешифраторов 3-8;
2. Тогда число каскадов $\mathrm{K} = \frac{n}{n1} = \frac{4}{3}$. Округляем до большего, получаем 2;
3. Для выходного каскада будет нужно $\frac{N}{N1} = \frac{16}{8} = 2$ дешифратора, для входного $\frac{N}{N1^2} = \frac{16}{8^2} = \frac{16}{64}$. Округляем до большего, получаем 1. Однако, т.к. изначально получилась дробь, во входном каскаде будет использоваться неполный дешифратор;
4. Все остальное без изменений.

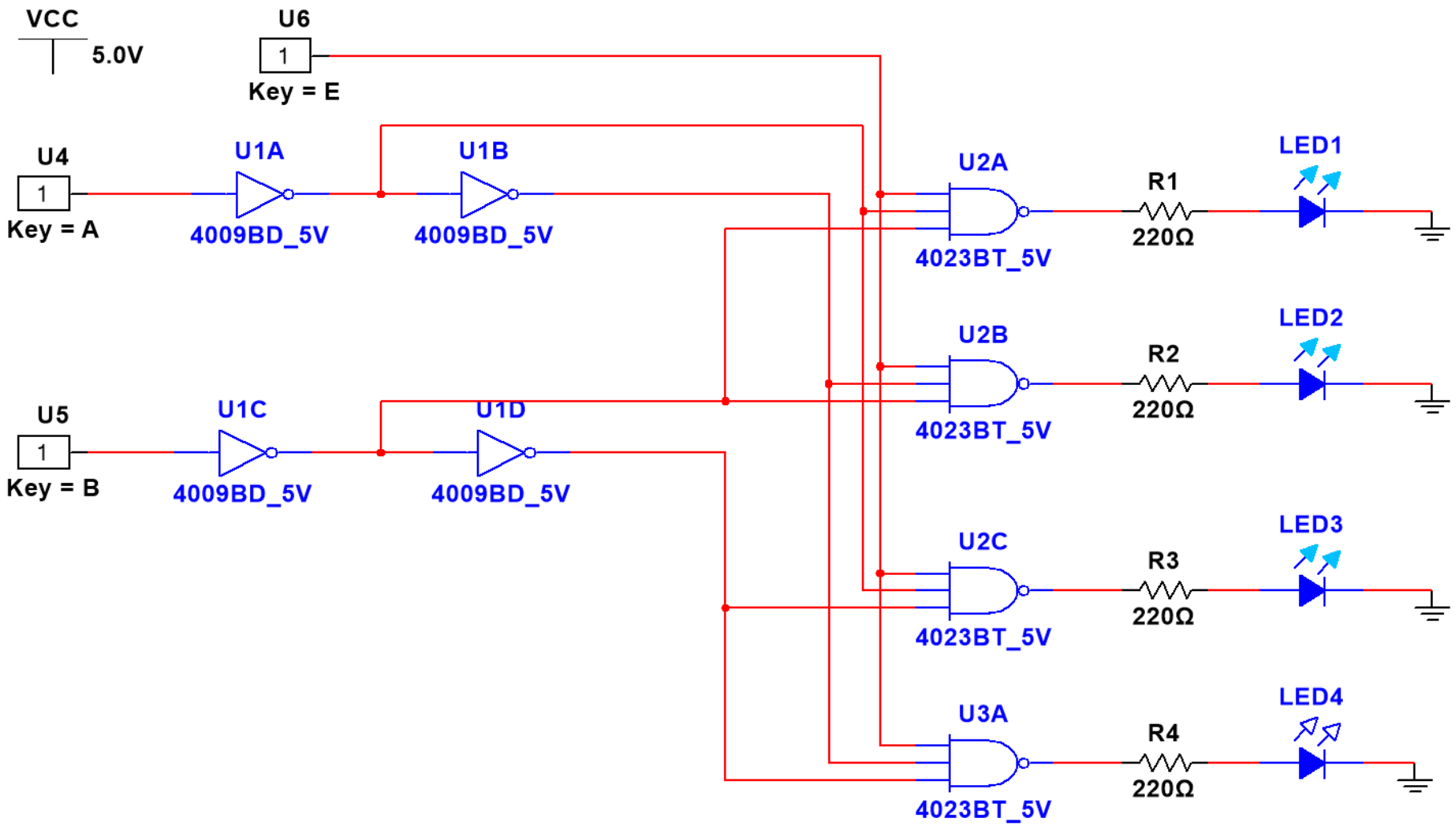


Ступенчатые дешифраторы

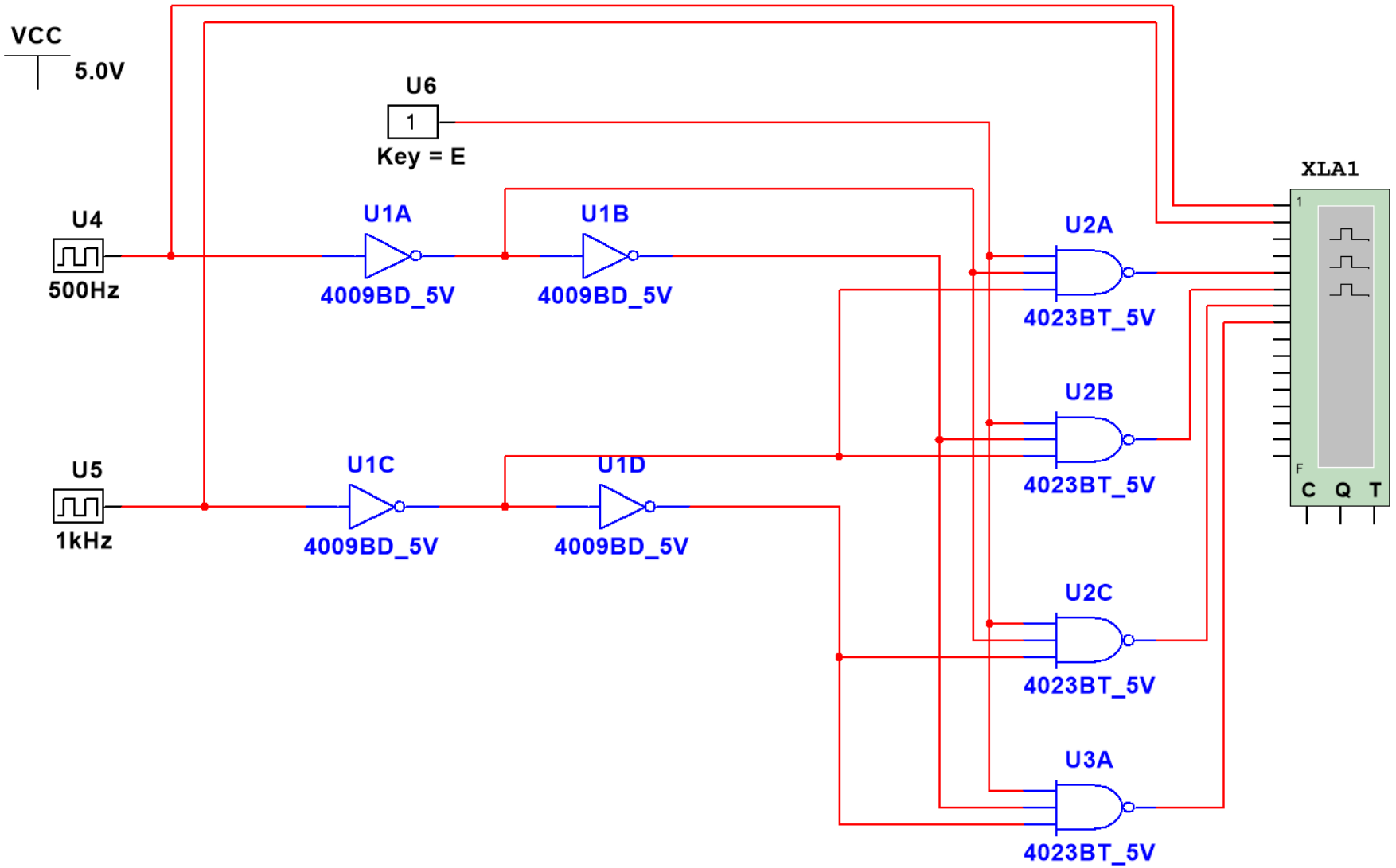
Вместе с увеличением количества каскадов, растет и задержка выходного сигнала. Время, необходимое для установления переходных процессов, растет пропорционально количеству каскадов и равно

$$t_{з} = t_{зд.р.ср.К1} + t_{зд.р.ср.К2} + \cdots + t_{зд.р.ср.КК}$$

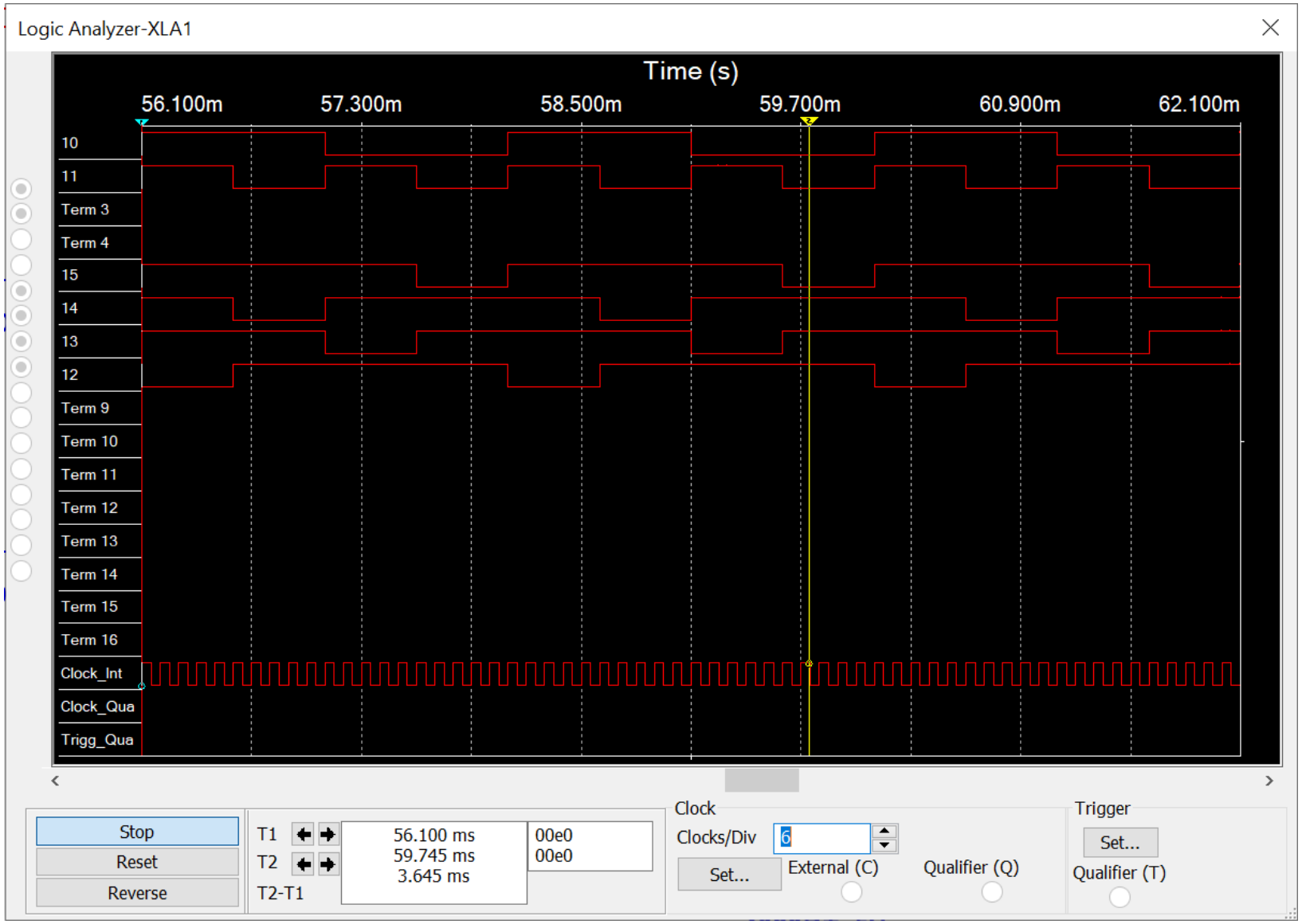
Линейный дешифратор на И-НЕ (Задание 1а, 1б)



Задание 1в, схема



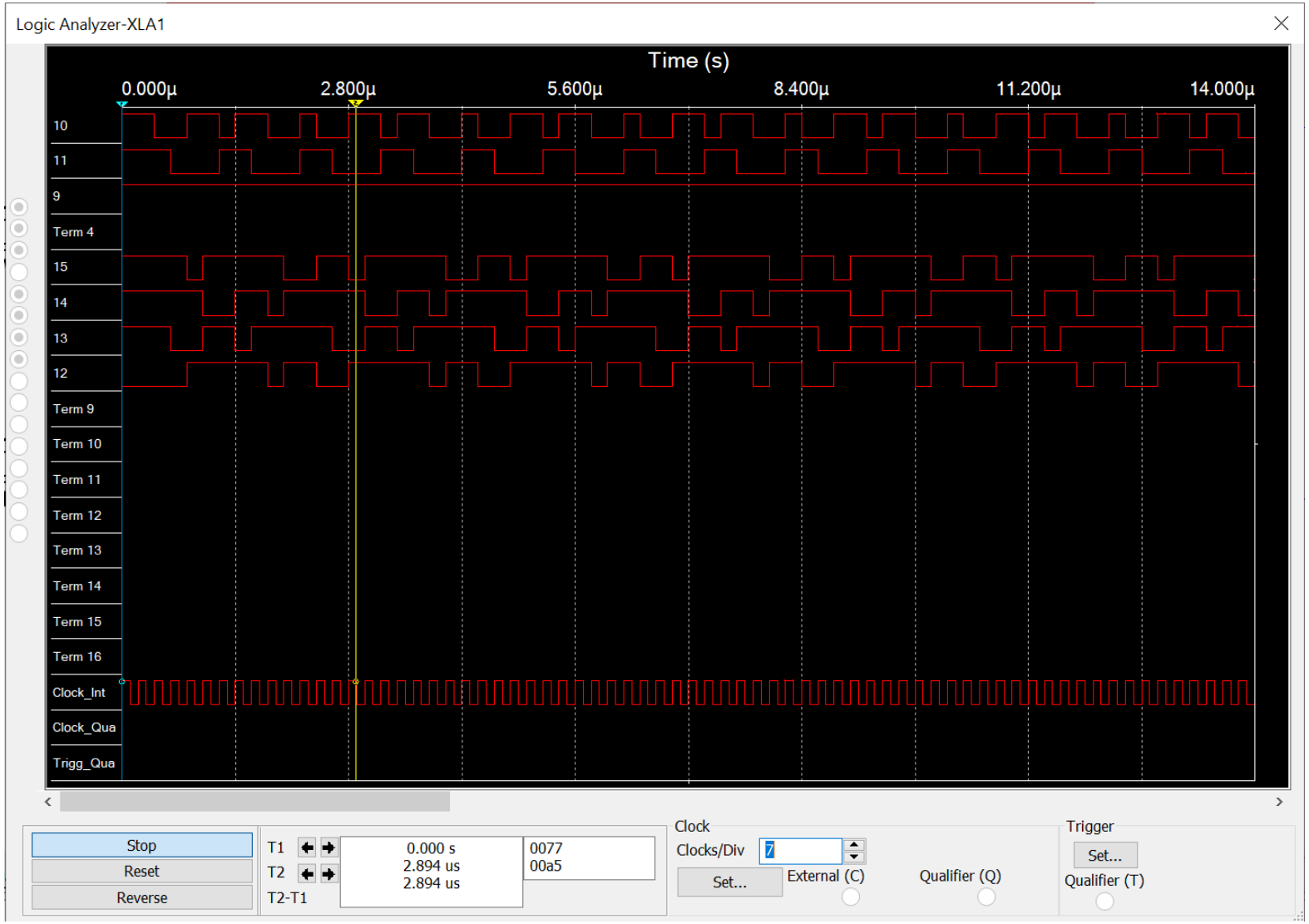
Задание 1в, временная диаграмма



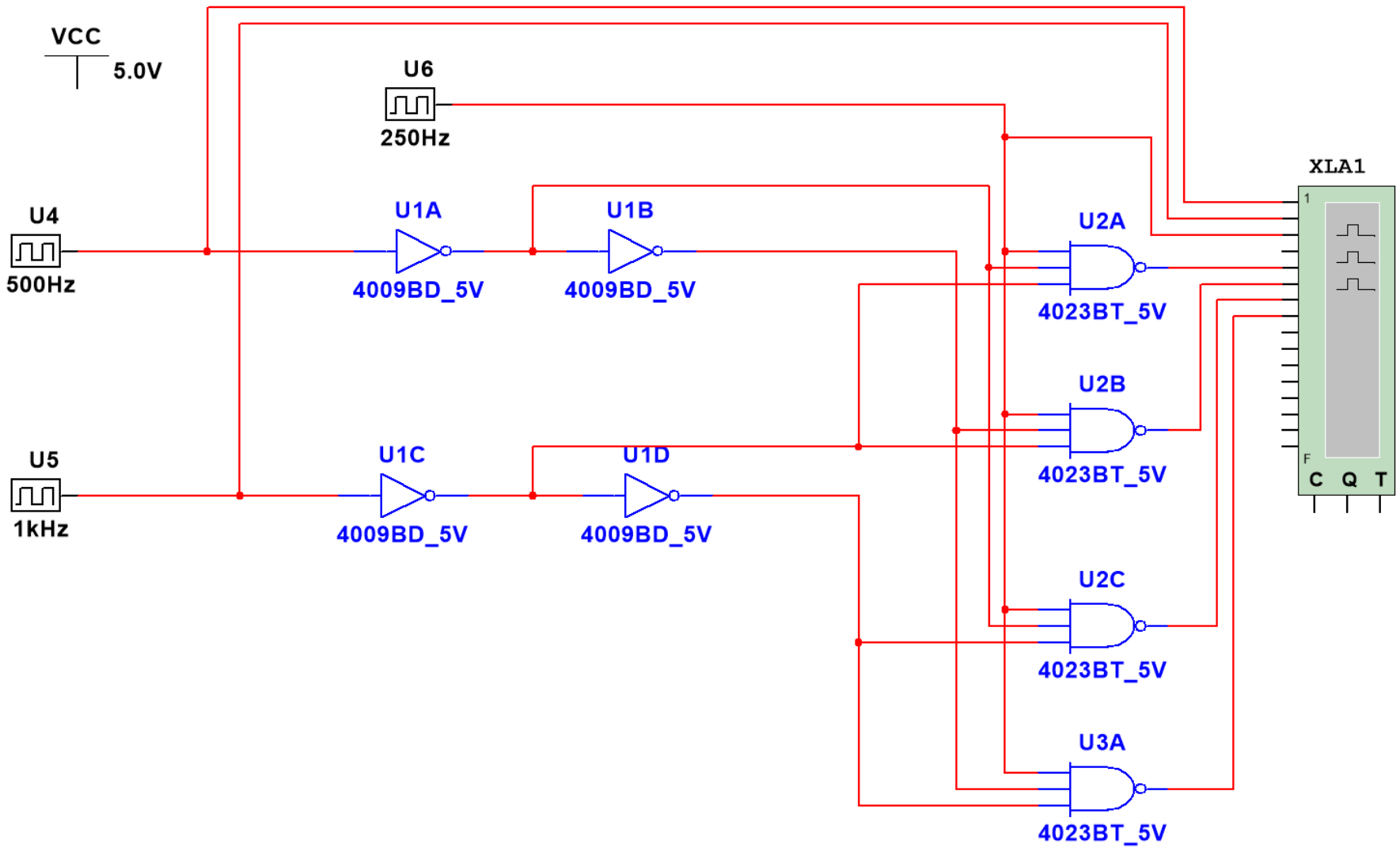
Задание 1г, определение помех

1. Используем схему из предыдущего задания;
2. Сигнал стробирования оставляем висеть в EN=1;
3. На генераторах сигналов для адресных входов выставляем высокие частоты (от 1 МГц), на двух входах выставляем разные частоты – например, на одном 1МГц, на другом 1.5 МГц;
4. Не забываем выставить собственную частоту Logic Analyzer (жмем по нему два раза, в разделе Clock жмем кнопку Set, выставляем такую частоту, чтоб была больше частоты входных сигналов, 5 МГц например), иначе ничего не отобразится;
5. Включаем симуляцию и наблюдаем бардак на выходах дешифратора (несколько входов активны одновременно, неправильная дешифрация сигналов и т.д.), фиксируем временную диаграмму.

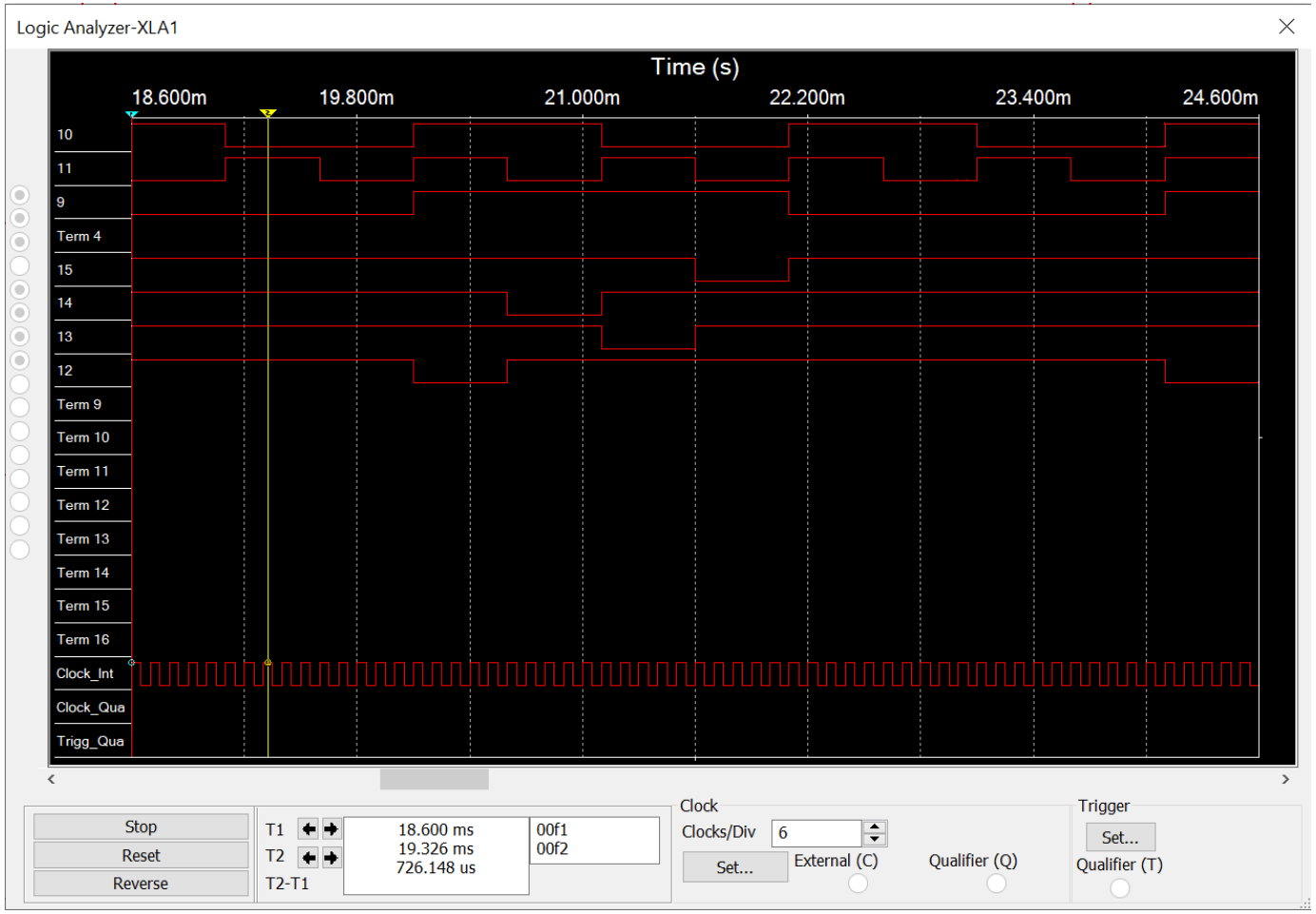
Задание 1г, временная диаграмма



Задание 1д, схема



Задание 1д, временная диаграмма



Разница с заданием 1в только в том, что сигнал стробирования EN теперь задается генератором и тоже меняется во времени. Нужно проследить, как будут вести себя выходы дешифратора.

Пока используем низкочастотные сигналы, как в 1в, чтобы понять принцип работы.

Задание 1е

Как определить время задержки сигнала стробирования.

1. Смотрим, какие элементы И-НЕ и НЕ мы используем (нельзя использовать идеальные, только реальные из секции CMOS). В примере использованы HEF4023B И-НЕ и инверторы схемы 4009;
2. Идем в интернет и ищем к этим элементам т.н. datasheet (лист данных). В гугл поиске достаточно вбить например HEF4023B datasheet;
3. Находим нужный документ, и в нем ищем информацию о задержках (табличка типа такой)

AC CHARACTERISTICS

V_{SS} = 0 V; T_{amb} = 25 °C; C_L = 50 pF; input transition times ≤ 20 ns

	V_{DD} V	SYMBOL	TYP.	MAX.		TYPICAL EXTRAPOLATION FORMULA
Propagation delays						
$I_n \rightarrow O_n$	5		65	135	ns	38 ns + (0,55 ns/pF) C_L
HIGH to LOW	10	t_{PHL}	25	50	ns	14 ns + (0,23 ns/pF) C_L
	15		15	30	ns	7 ns + (0,16 ns/pF) C_L
	5		65	130	ns	38 ns + (0,55 ns/pF) C_L
LOW to HIGH	10	t_{PLH}	30	60	ns	19 ns + (0,23 ns/pF) C_L
	15		25	45	ns	17 ns + (0,16 ns/pF) C_L
Output transition times	5		60	120	ns	10 ns + (1,0 ns/pF) C_L
HIGH to LOW	10	t_{THL}	30	60	ns	9 ns + (0,42 ns/pF) C_L
	15		20	40	ns	6 ns + (0,28 ns/pF) C_L
	5		60	120	ns	10 ns + (1,0 ns/pF) C_L
LOW to HIGH	10	t_{TLH}	30	60	ns	9 ns + (0,42 ns/pF) C_L
	15		20	40	ns	6 ns + (0,28 ns/pF) C_L

Задание 1е

4. Берем значение задержки для наших параметров (в мультисиме схема из примера используется при 5 В питания, значит, берем макс. значение при 5 В, то есть, 135 нс);

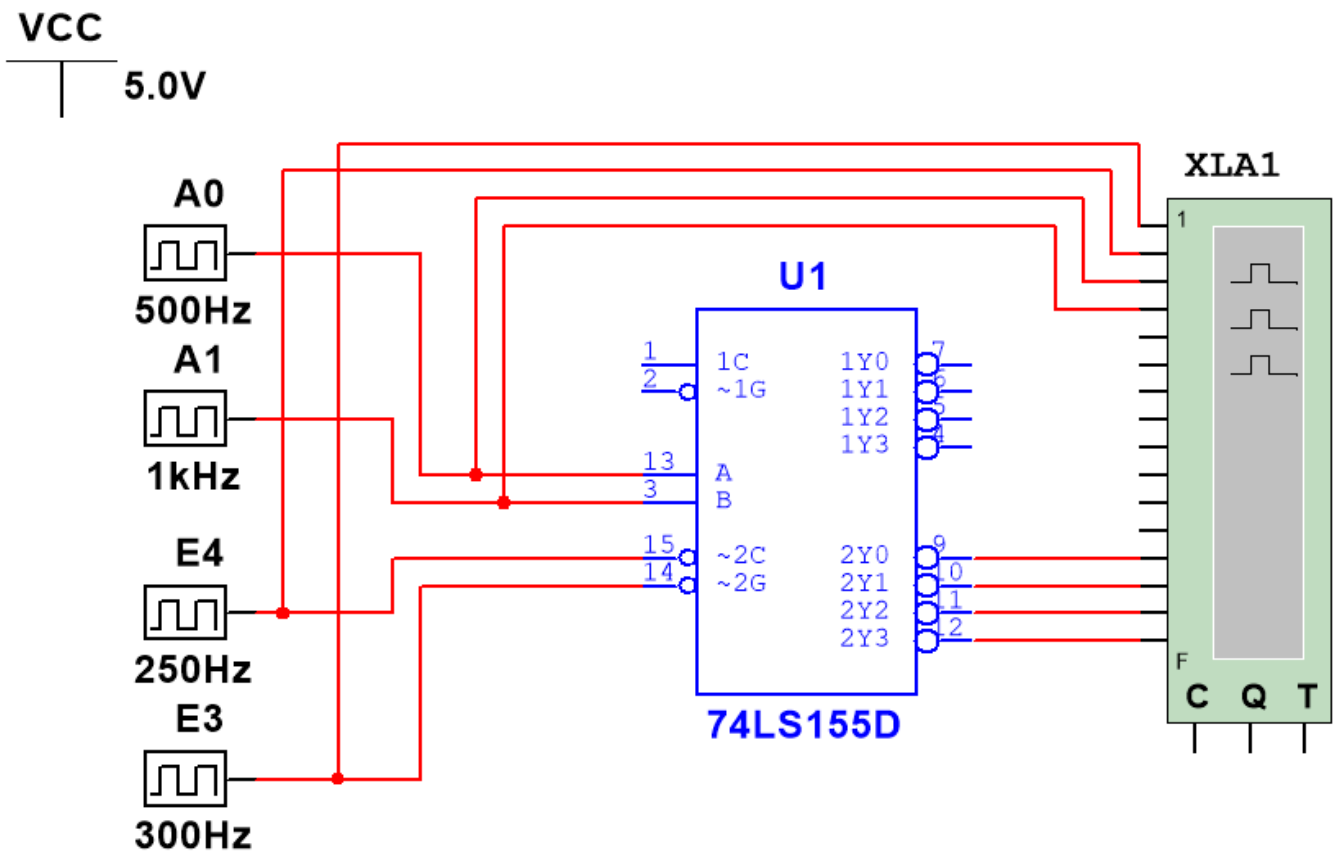
5. Аналогично находим значение для инвертора (в нашем случае – 60 нс);

6. Считаем задержку по формуле

$$t_{\text{зд.р.ср.}} = 2t_{\text{зд.р.ср.1}} + t_{\text{зд.р.ср.2}} = 2 * 60 + 135 = 255 \text{ нс}$$

7. Результаты вычислений – в отчет. МОЖНО использовать те же элементы и их листы данных, что и в примере, но за поиск и использование других можно получить пару лишних баллов.

Задание 2а



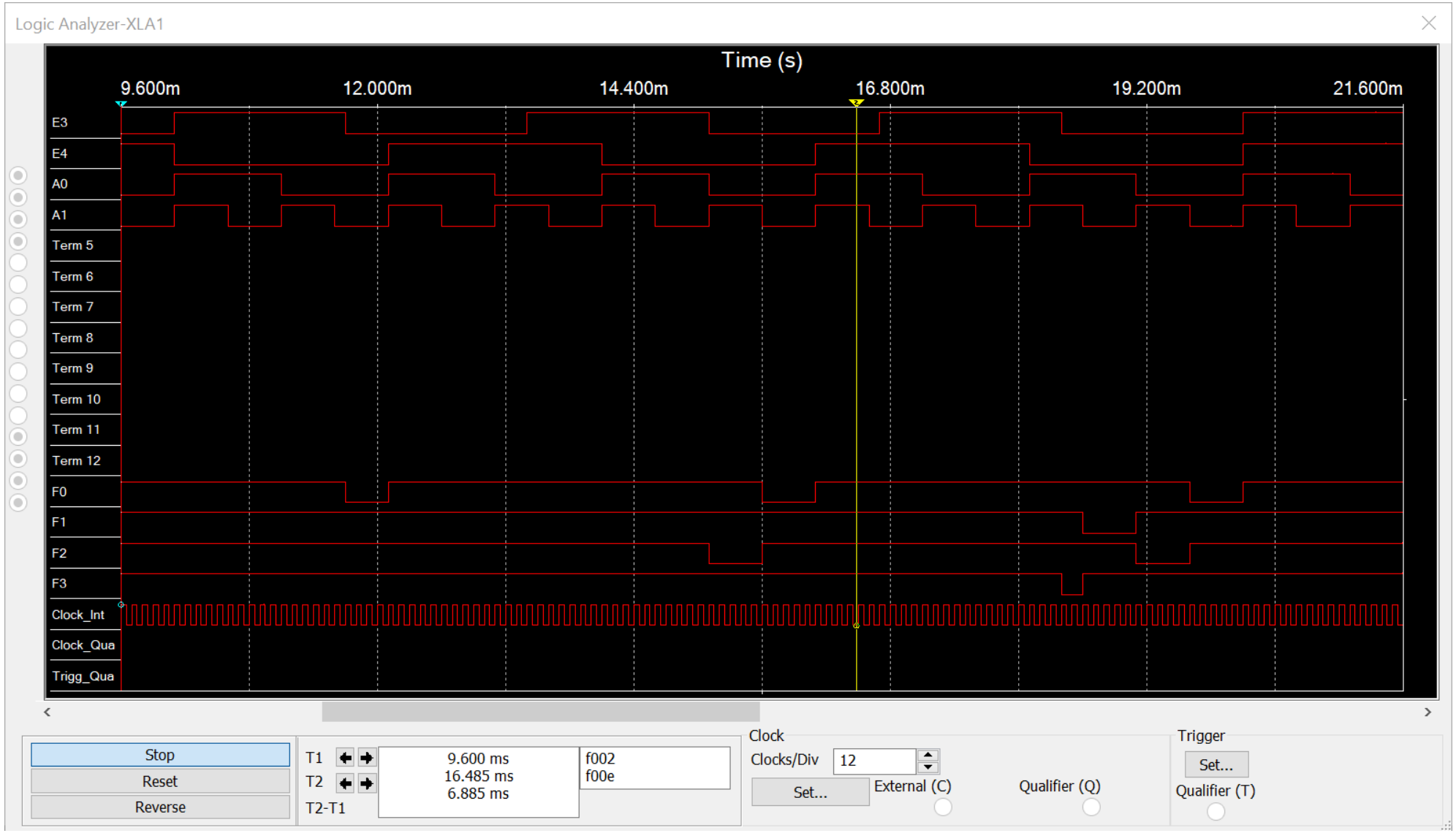
Используем низкочастотные сигналы для входов (порядка нескольких кГц и ниже), чтобы не заморачиваться с линиями задержки для стробирования.

Входы подключать ТОЛЬКО так, как на слайде.

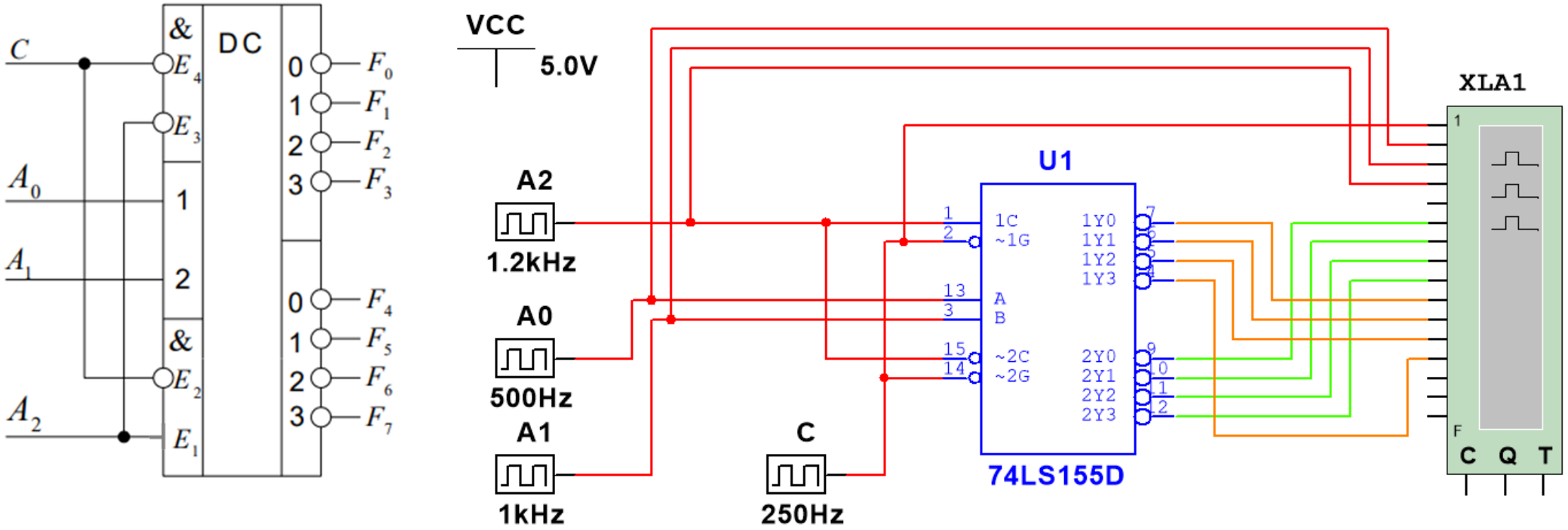
Обратите внимание, что входы стробирования – инверсные.

<u>Задание 2б делать не обязательно, для общего развития разве что.</u>

Задание 2a, временная диаграмма

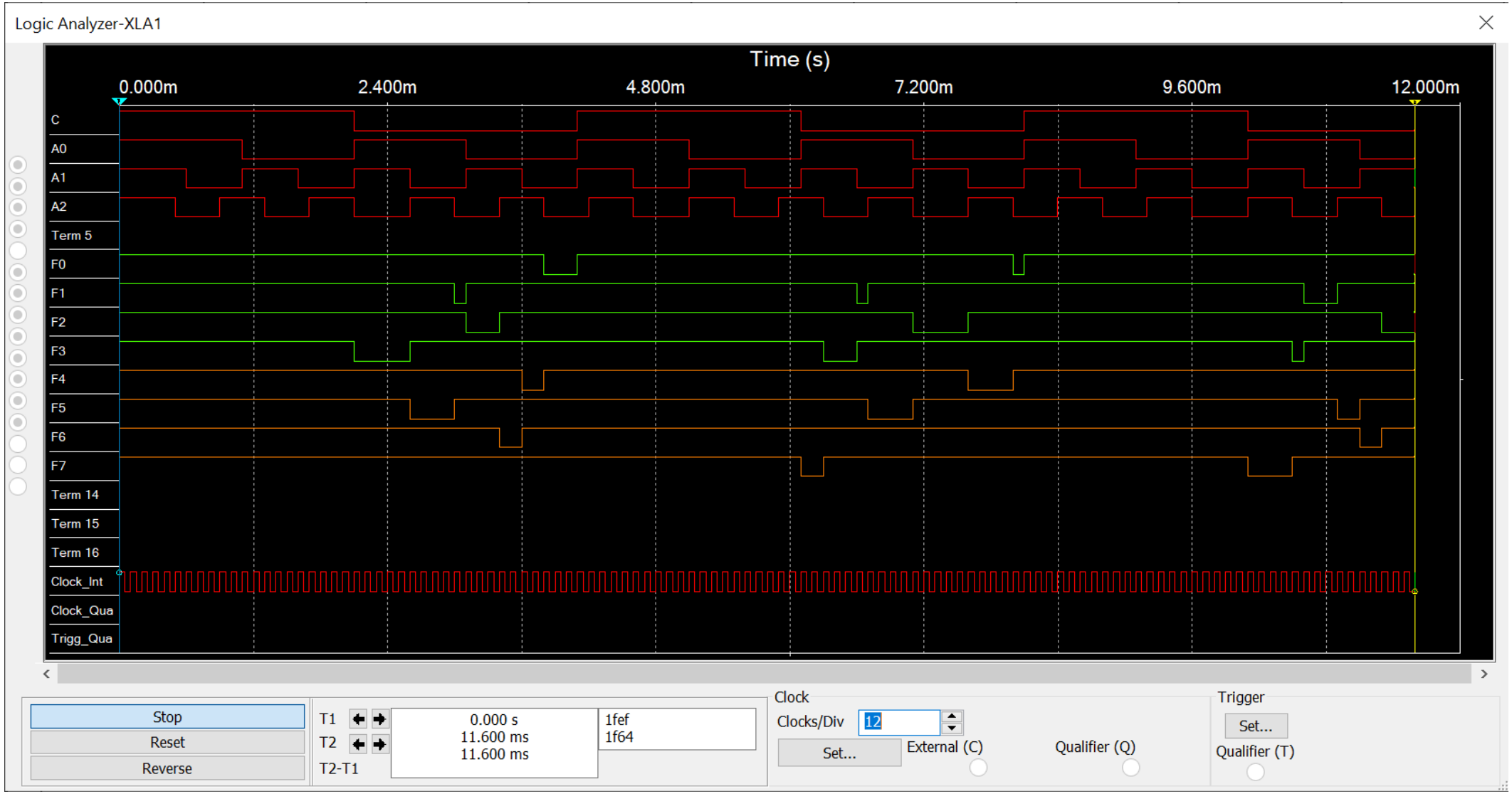


Задание 2в

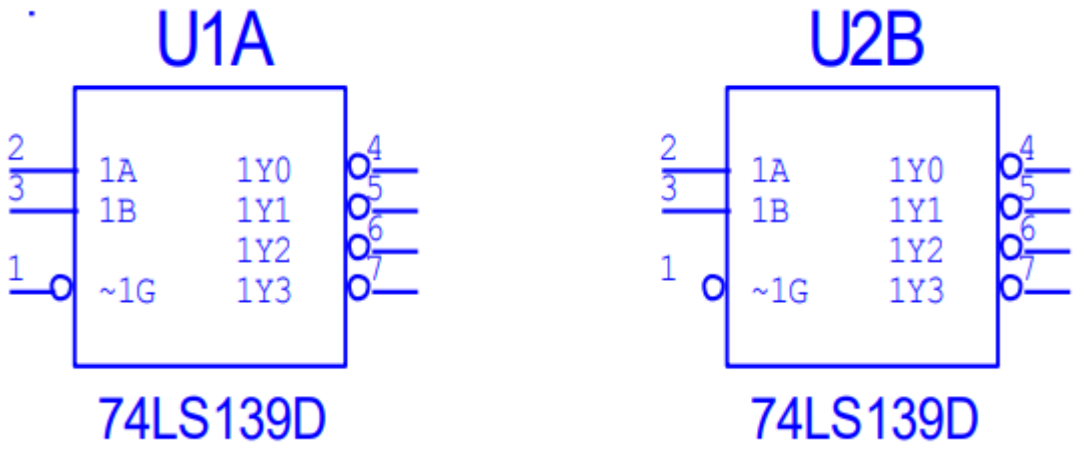


Выводы микросхемы в Multisim располагаются иначе, чем на рисунке, поэтому для корректного отображения подключаем выходы к анализатору так, как показано на слайде (старшие разряды из группы 1Y, младшие из группы 2Y). Вход A2 также подключается с другой стороны, нежели на рисунке.

Задание 2в, временная диаграмма

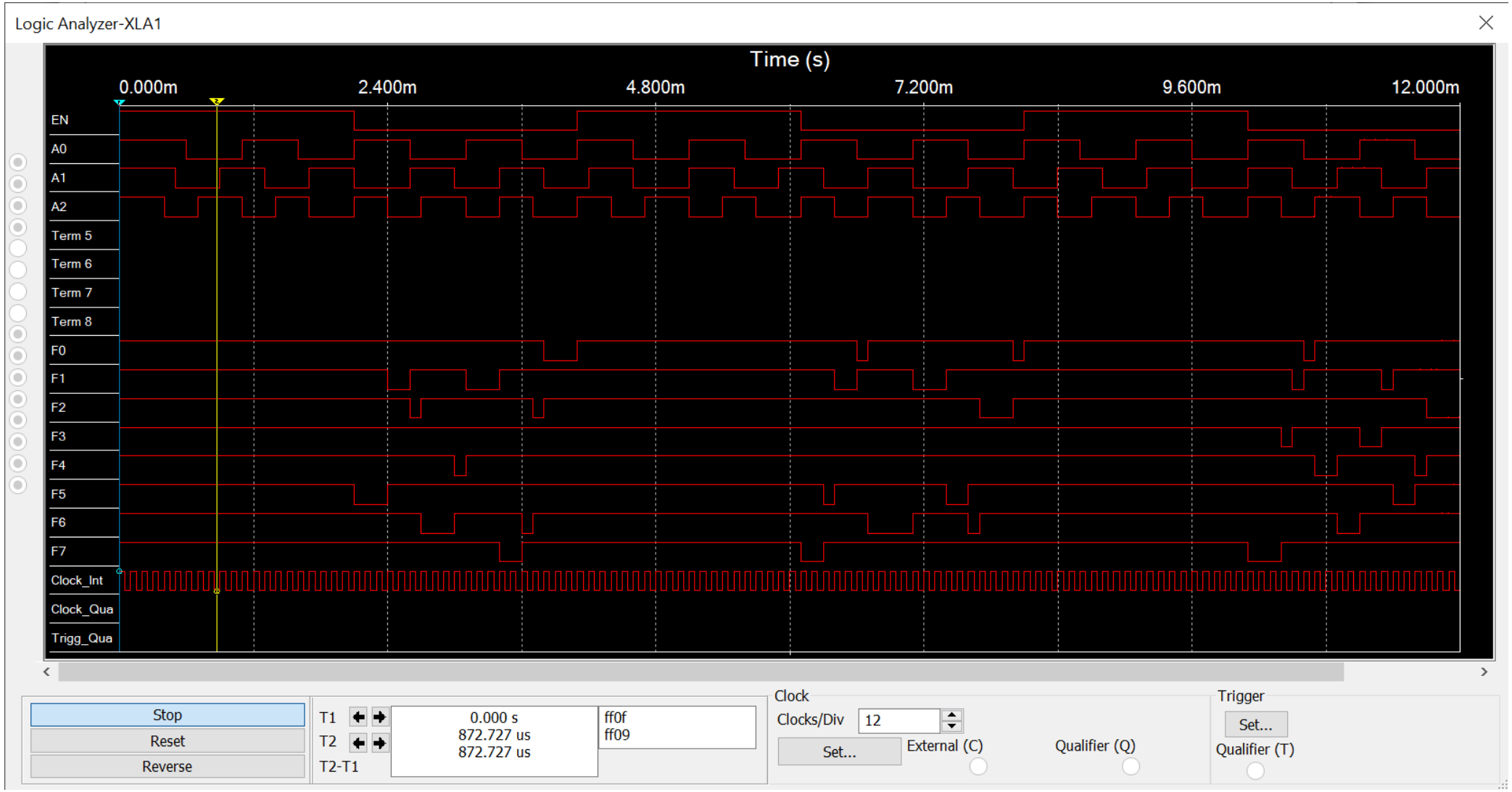


Задание 3

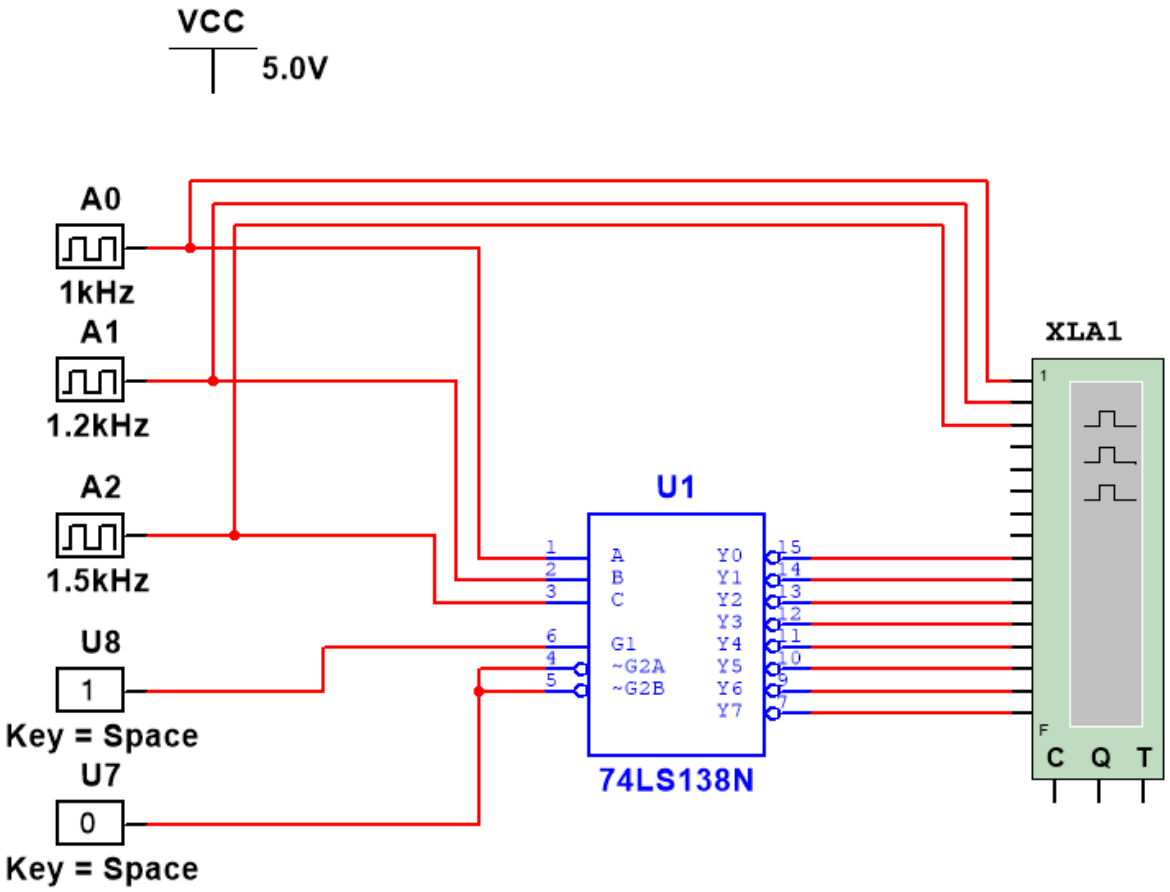


Используя микросхему 74LS139D (дешифратор 2-4), собрать дешифратор 3-8. Пользуемся правилами, обозначенными на слайдах 11-13. Входы стробирования дешифраторов выходного каскада придется расширить до 2 для каждого дешифратора – один принимает сигнал стробирования, другой служит входом для сигнала с предвыходного каскада.

Задание 3, временная диаграмма



Задание 4



4а Собираем нестробируемую схему на одном дешифраторе, снимаем временные диаграммы;

4б Из дешифраторов 3-8 микросхемы 74LS138N собираем дешифратор 5-32 по уже известным правилам. Собранную схему добавляем в отчет только в графическом виде, в Multisim ее собирать НЕ НУЖНО.