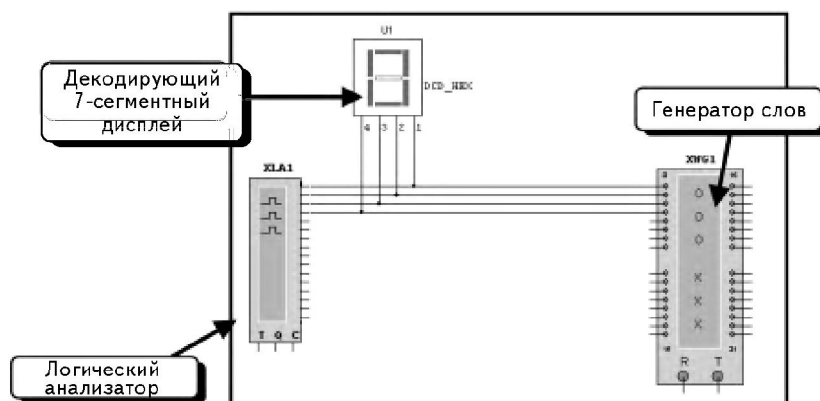
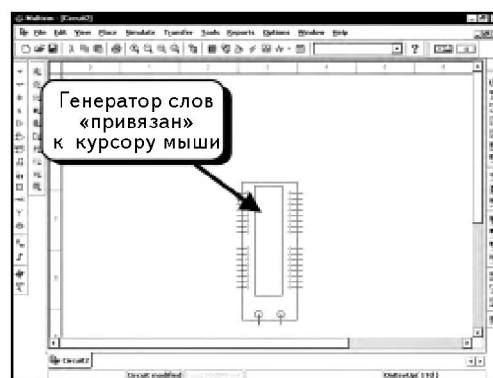
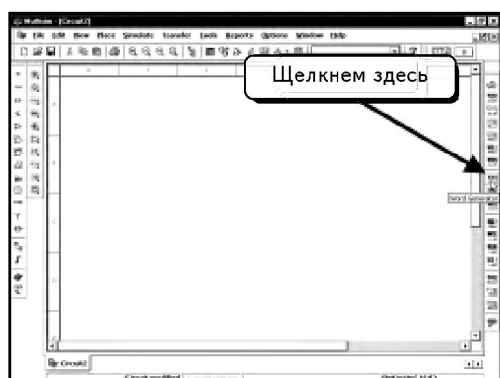


7.1.1. Инструменты Word Generator и Logic Analyzer

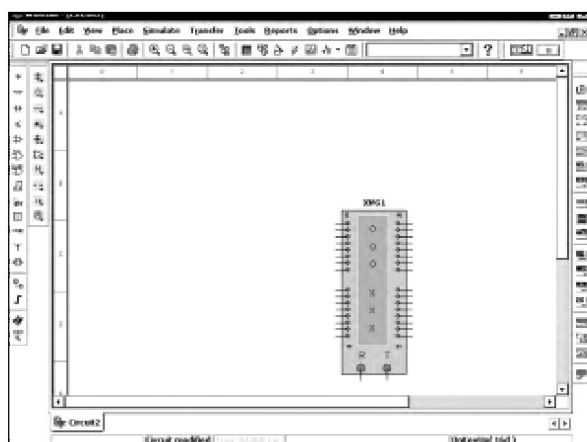
Сначала сформируем часть данной схемы:



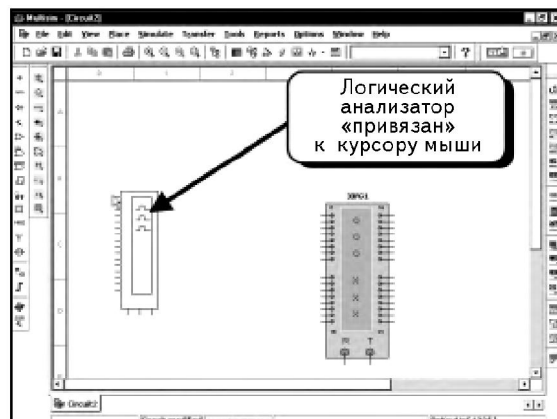
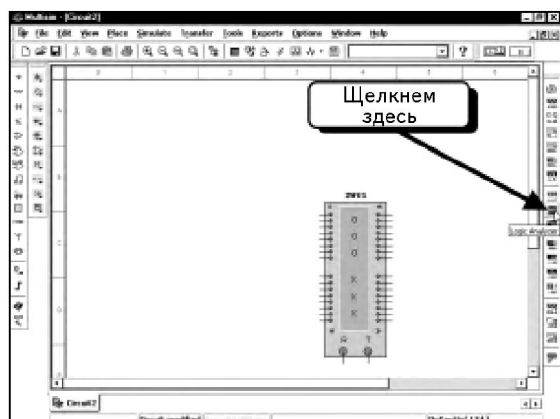
Инструменты **Word Generator** и **Logic Analyzer** находятся на панели инструментов **Instruments**. Чтобы добавить инструмент **Word Generator**, щелкнем кнопку **Word Generator** . Инструмент будет «привязан» к курсору мыши:



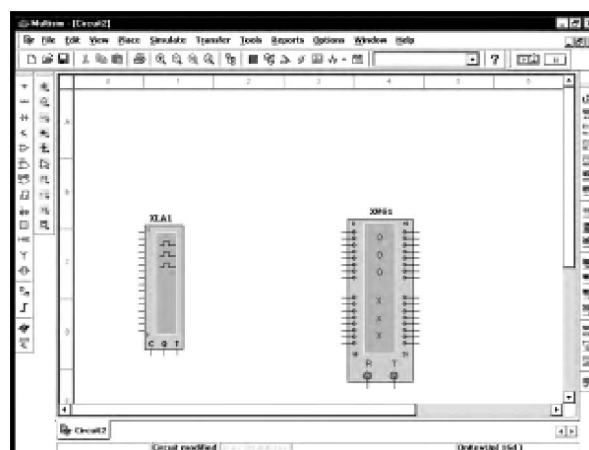
Переместим курсор мыши в нужное положение и щелкнем **ЛЕВОЙ** кнопкой мыши, добавив компонент в схему:



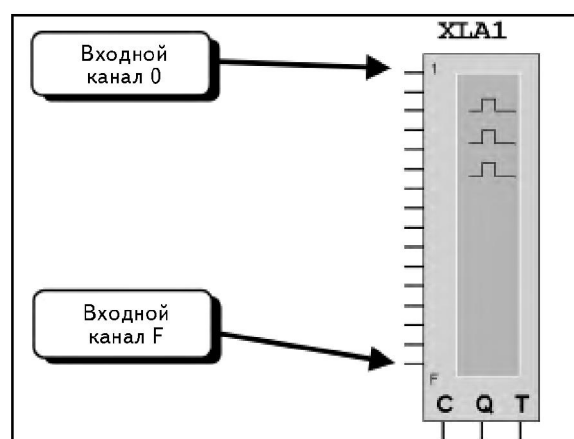
Чтобы добавить инструмент **Logic Analyzer**, щелкнем кнопку **Logic Analyzer** . Инструмент также будет «привязан» к курсору мыши:



Добавим компонент в схему:

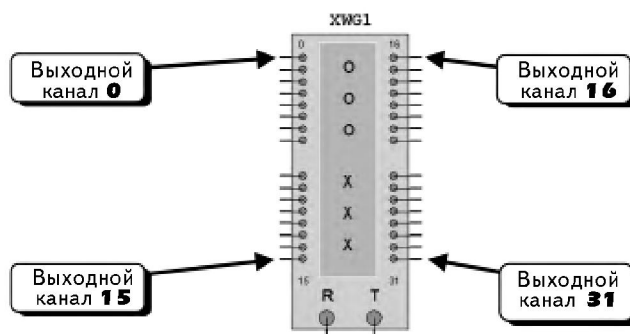


Рассмотрим более внимательно условные изображения инструментов **Logic Analyzer** и **Word Generator**. Изображение инструмента **Logic Analyzer** показано ниже:



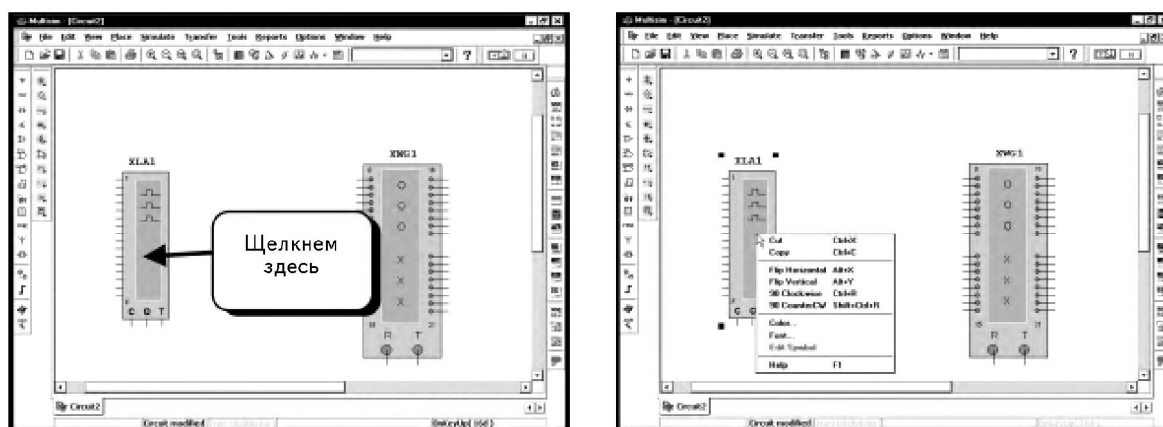
Он имеет 16 входов, помеченных цифрами от **1** до **F** (а не от 0 до F). При работе с логическими схемами учтите, что F — это шестнадцатеричное число, которое соответствует числу 15 в десятичной системе исчисления или числу 1111 в двоичной системе. Можно одновременно отобразить все 16 входов инструмента **Logic Analyzer**. С помощью ярлыков **1** и **F** определим, какой вход является первым, а какой — последним.

Условное изображение инструмента **Word Generator** показано ниже:

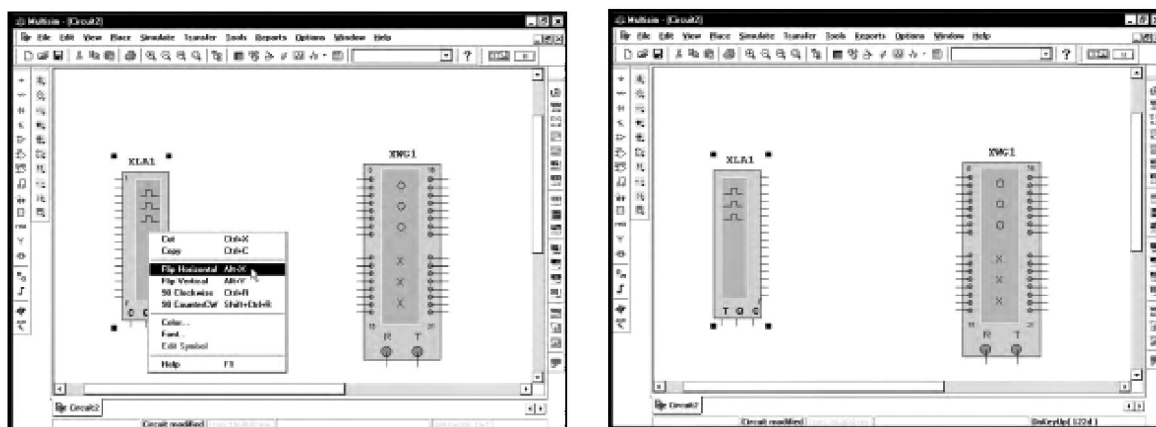


Он имеет 32 выхода, помеченных от 0 до 31. Каждый выход создает логический сигнал, который может использоваться как входной для логической схемы. Выход любого канала является независимым от других каналов.

В данном примере будут использоваться первые 4 выхода инструментов **Word Generator** и **Logic Analyzer**. Чтобы упростить работу с инструментами, надо перевернуть **Logic Analyzer** по горизонтали, направив входные контакты в сторону инструмента **Word Generator**. Щелчком правой кнопки мыши по иконке **Logic Analyzer**. Появится меню:

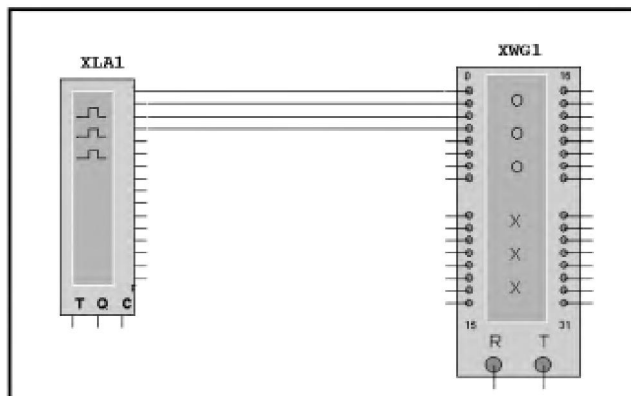


Выберем, как показано, опцию **Flip Horizontal** (Перевернуть по горизонтали):

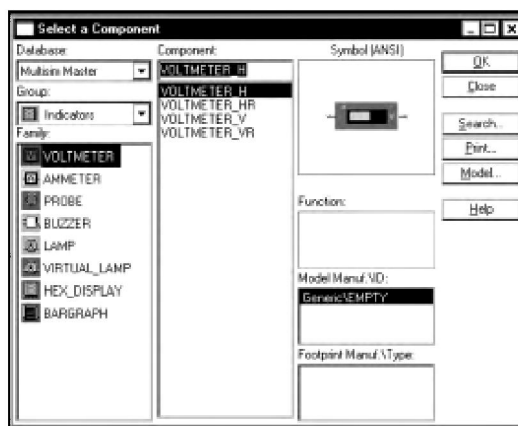
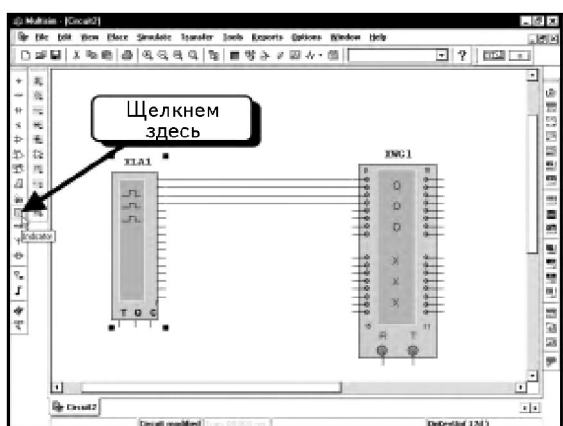


Теперь входные контакты инструмента **Logic Analyzer** находятся справа.

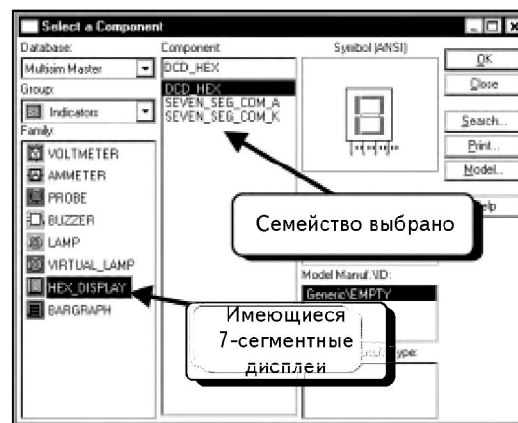
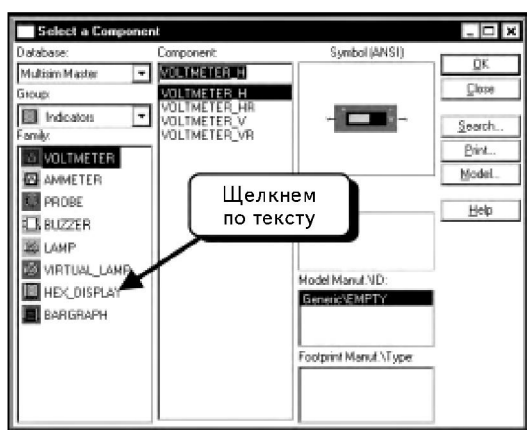
Далее соединим инструменты **Logic Analyzer** и **Word Generator**. (Если вы не знаете, как соединять компоненты, обратитесь к разделу 1.4). Возможно, понадобится переместить один из инструментов вверх или вниз:



Теперь введем в схему декодирующий 7-сегментный дисплей. Этот компонент находится в группе **Indicator** (Индикатор). Нажмем кнопку **Indicator**  :

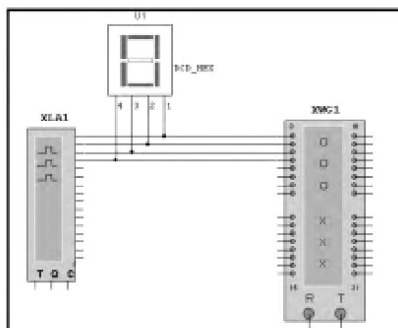


Выберем семейство **HEX_DISPLAY** :

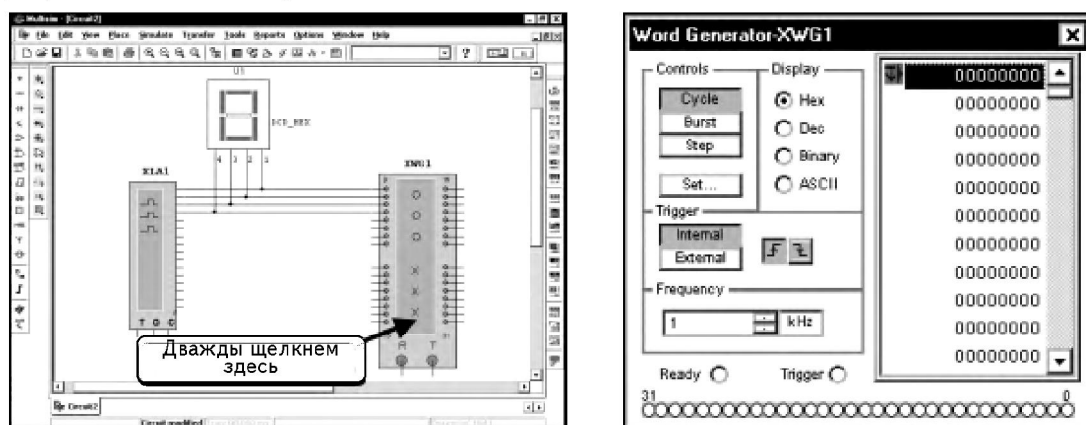


Компонент **DCD_HEX** представляет собой индикатор с 7 сегментами. Это позволяет подать на 4 входа двоичный код от 0000 до 1111 и на буквенно-цифровом дисплее получить его шестнадцатеричный эквивалент от 0 до F. Система преобразования входных двоичных сигналов встроена в компонент. Два других компонента имеют входы, каждый из которых управляет непосредственно 1 сегментом. Для управления придется добавить схему преобразования сигналов. Если вами освоены индикаторы, скорее всего, вы воспользуетесь компонентом **Seven_Seg_Com_K** или **Seven_Seg_Com_A**, т. к. схема уже включает все необходимые элементы. Если необходимо отобразить состояние 4 двоичных входов, выберите компонент **DCD_HEX**, тогда достаточно добавить его в схему и подключить четыре входа.

Компонент **DCD_Hex** выделен по умолчанию; нажмем кнопку **OK**, чтобы добавить его в схему. Подключим компонент так, как показано:

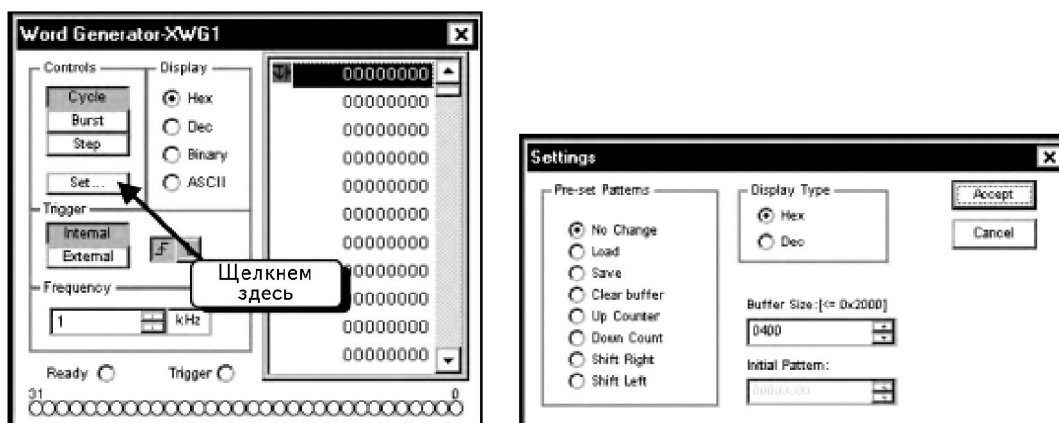


Последнее, что следует сделать до моделирования, — это запрограммировать инструмент **Word Generator**. Дважды щелкнем по компоненту **Word Generator**:

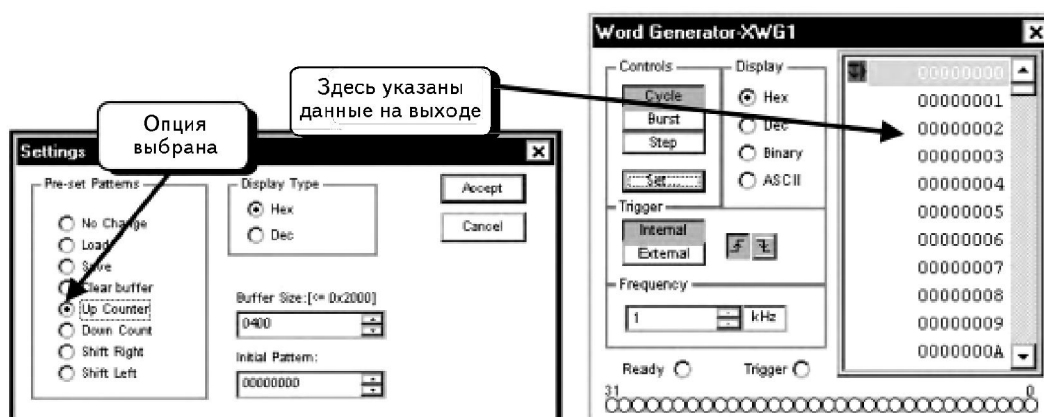


Чтобы инструмент **Word Generator** выполнял счет от 0 до 15 (F) и обратно, нам потребуется 30 (1E) значений, которые займут 30 ячеек памяти инструмента **Word Generator**. Необходимы только 30 ячеек, поскольку придется возвращаться к ячейке, в которой записан 0.

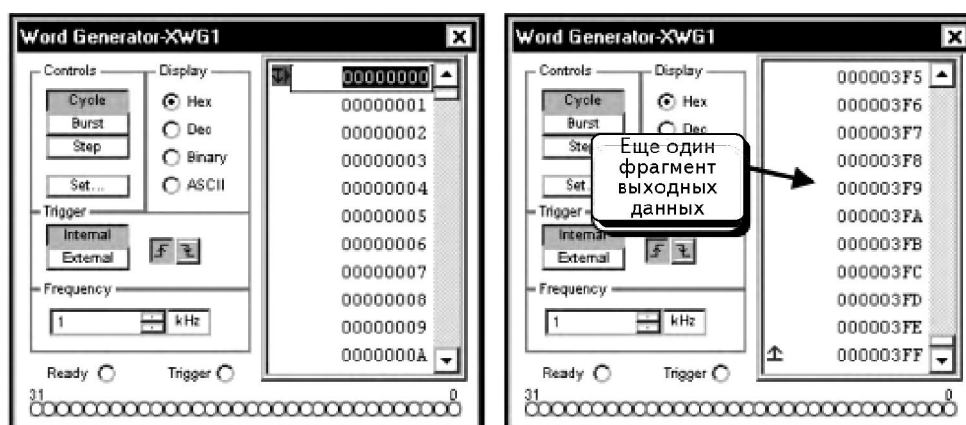
Можно автоматически создать возрастающие значения. Нажмем кнопку **Set** (Настроить):



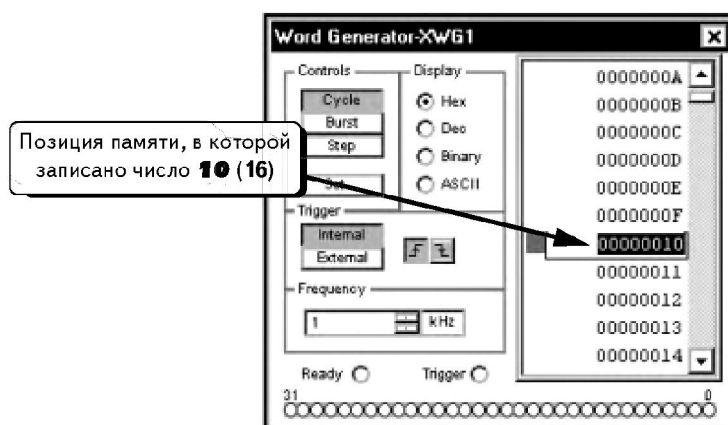
Это диалоговое окно позволяет загрузить и сохранить данные в файл, а также создать стандартные системы счета. Выберем опцию **Up Counter** (Возрастающий счет) и нажмем кнопку **Accept** (Принять), как показано ниже:



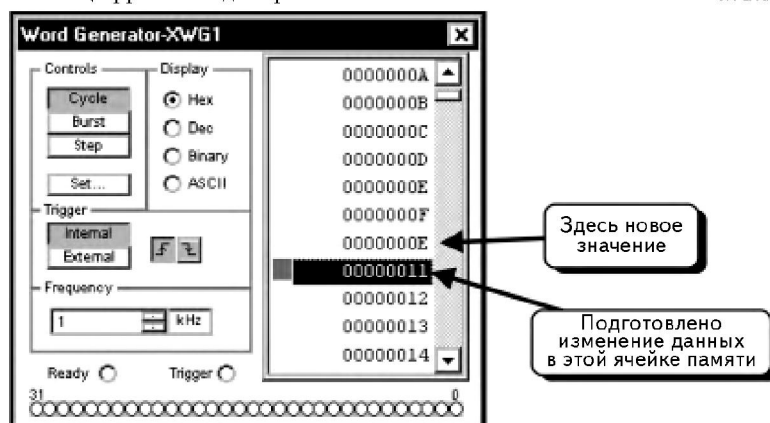
Если посмотреть на данные выхода, можно заметить, что счет выполняется в шестнадцатеричной системе от 0 до 3FF:



Это значит, что можно создать возрастающую систему счета. Нисходящую систему счета необходимо сформировать вручную. Найдем адрес 10 (шестнадцатеричное число) и выделим его. Для этого щелкнем по ячейке:

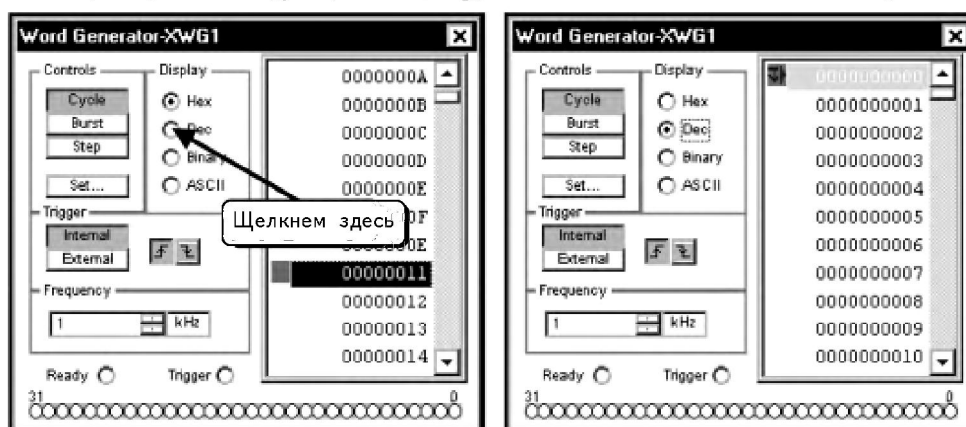


Новое значение можно ввести как двоичное или шестнадцатеричное число или как код ASCII (для нашего примера он не подходит). Далее необходимо записать в эту ячейку **Word Generator** шестнадцатеричный код E. Введем «E» и нажмем клавишу **ENTER**. Курсор переместится в следующую ячейку памяти, появится возможность изменить ее содержимое:

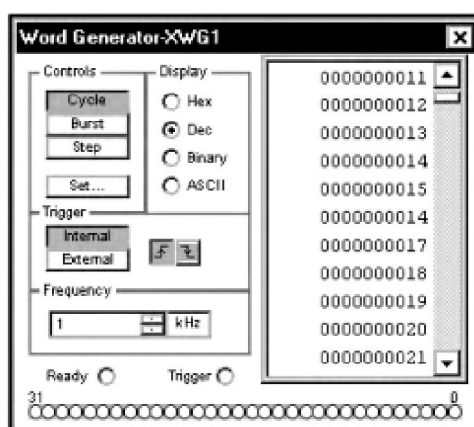


Можно изменить и следующий адрес памяти.

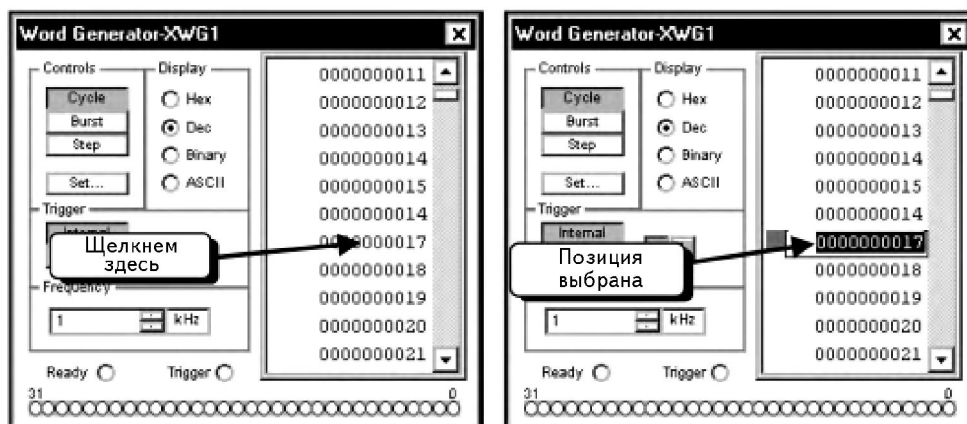
Чтобы показать, как работать с функцией **Display**, изменим значение на **Decimal**. Выберем опцию **Dec**:



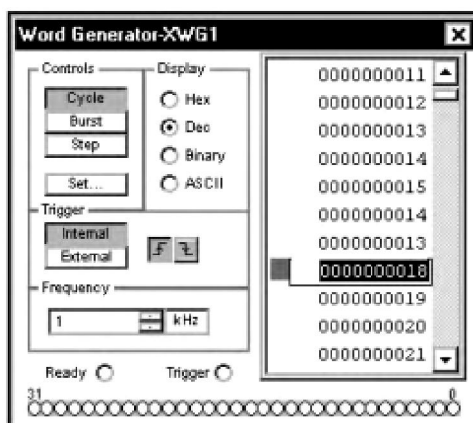
Адрес данных был обнулен. Если посмотреть данные, то видно, что теперь информация отображена в десятичном коде:



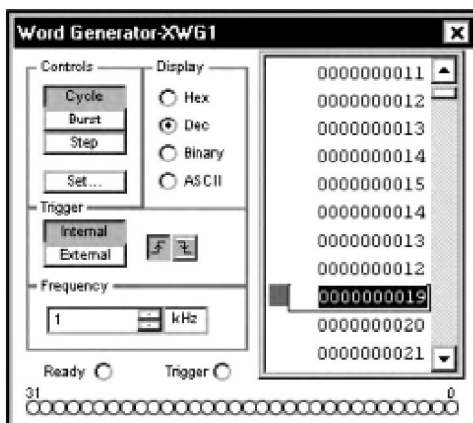
Видно, что сейчас система выполняет счет от **0** до **15**, потом возвращается к **14** (т. к. уже было записано одно число для отсчета в нисходящем порядке), переходит к **17** и продолжает считать в восходящем порядке. Щелкнем по значению **17**, чтобы изменить данные:



Если ячейка памяти выбрана, можно ввести в нее новое значение. Чтобы продолжить отсчет в нисходящем порядке, введем число **13** и нажмем клавишу **ENTER**:

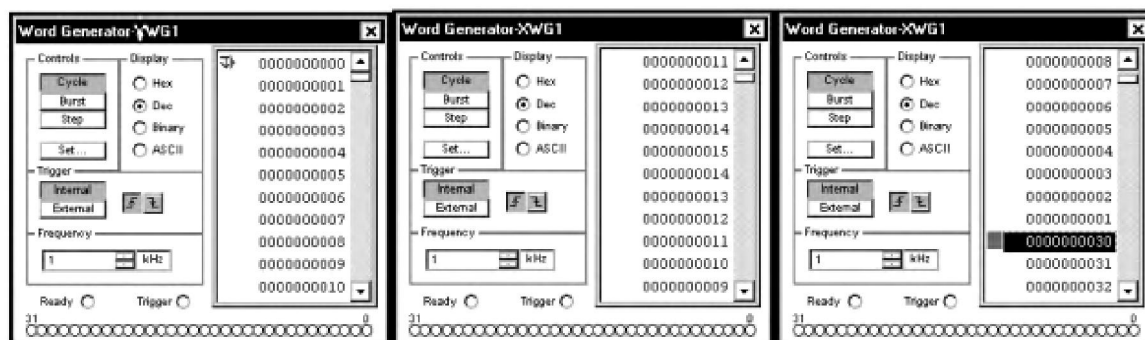


Теперь можно перейти к следующей ячейке памяти. Введем значение **12** и нажмем клавишу **ENTER**:

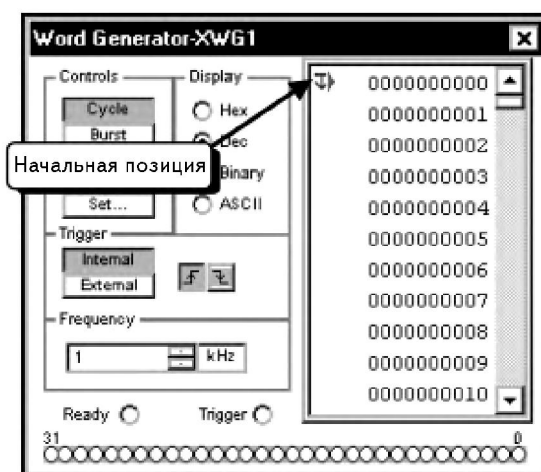


Будем действовать подобным образом, пока не вернемся к значению 1. Значения были заданы в десятичном коде **Decimal (Десятичный)**. Можем отобразить значения в шестнадцатеричном, десятичном, двоичном или ASCII коде.

Когда введем все данные, окно инструмента **Word Generator** должно выглядеть так, как показано ниже:

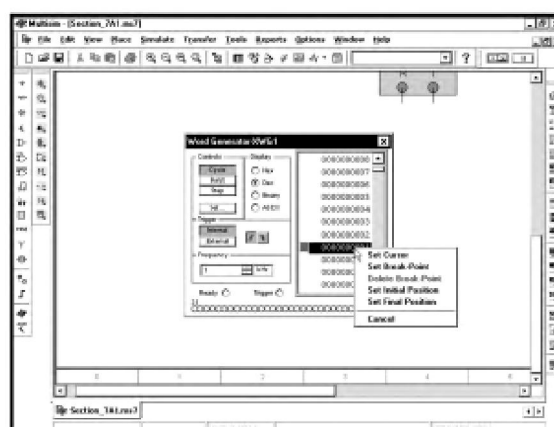
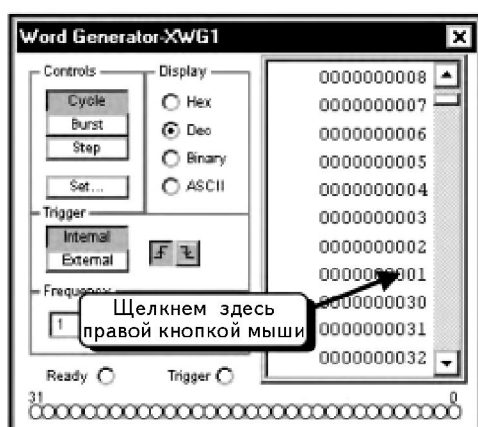


Далее сообщим инструменту **Word Generator**, где находятся начальная и конечная ячейки данных. Моделирование начинается в начальной ячейке данных и завершается в конечной. По умолчанию начальная ячейка имеет значение 0 и обозначается синей стрелкой, направленной вниз  :

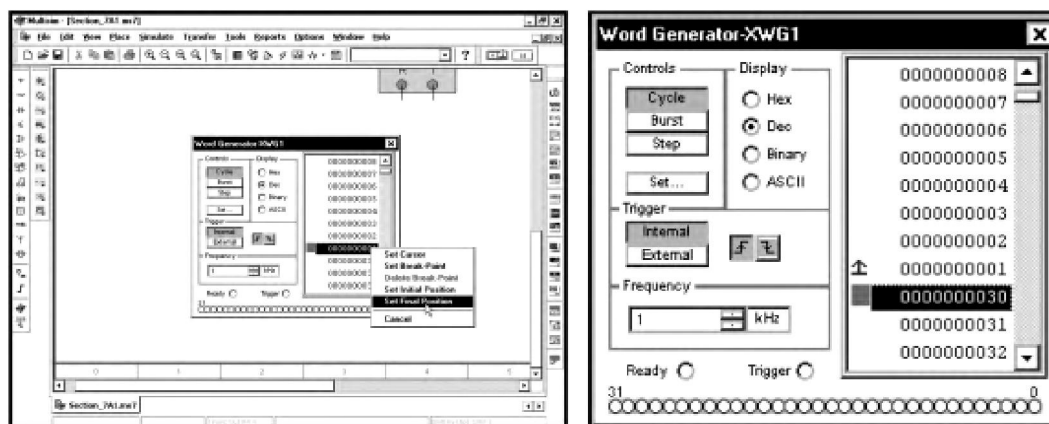


Новый цикл моделирования начнется в ячейке, которая содержит индикатор начального положения.

По умолчанию конечная ячейка данных располагается после всех 1024 ячеек. В данном примере используются только 30 ячеек данных (с адресами от 0 до 29), поэтому необходимо переместить индикатор конечной ячейки. Пойдем последнюю ячейку в последовательности и щелкнем по ней правой кнопкой мыши:

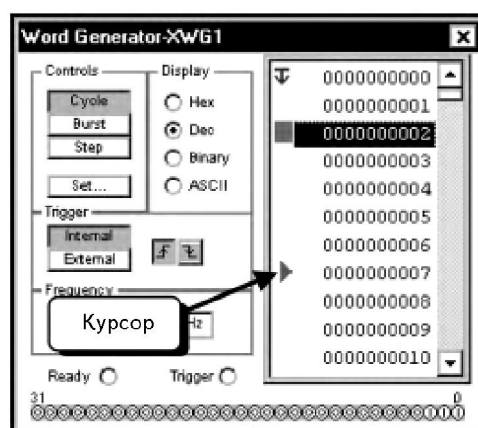


Выберем опцию **Set Final Position** (Настроить конечное положение), как показано:



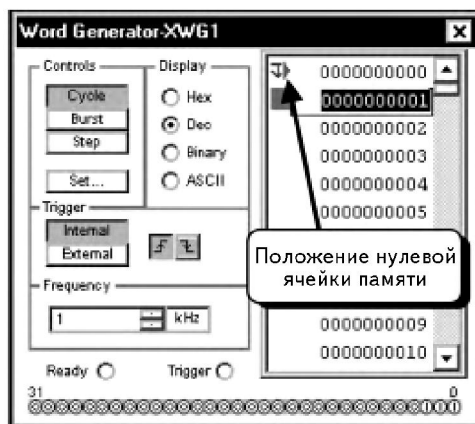
Видно, что индикатор конечного положения  появился рядом с последней запрограммированной ячейкой памяти. При выполнении моделирования инструмент **Word Generator** будет проходить цикл только для данных, расположенных между индикаторами начального и конечного положения.

Запрограммировав **Word Generator**, надо выбрать режим его работы. Режим **Cycle** (Цикл) обозначает, что инструмент **Word Generator** последовательно выводит содержимое каждой ячейки памяти от начальной до конечной. Когда инструмент достигает конечной ячейки, он возвращается к начальной и повторяет цикл. В режиме **Burst** (Просмотр) цикл будет повторяться до тех пор, пока не будет остановлено моделирование. В режиме **Step** (Шаг) цикл будет повторяться только один раз. В режиме **Step** (Шаг) будет выведено содержимое лишь одной ячейки. При щелчке по кнопке **Step** инструмент **Word Generator** будет переходить к следующей ячейке, выводя ее содержимое. Индикатор будет указывать на данную ячейку, пока вновь не будет нажата кнопка **Step**. Этот режим полезен при отладке схем. Если во время программирования инструмента **Word Generator** щелкнуть по ячейке памяти правой кнопкой мыши и выбрать опцию **Set Break-Point** (Настроить точку прерывания), данная ячейка будет настроена как точка прерывания. **Word Generator** остановится в точке прерывания в режиме **Cycle** или **Burst**. Процесс перемещения по ячейкам не возобновится до тех пор, пока не будет нажата кнопка **Cycle**, **Burst** или **Step**. Чтобы удалить точку прерывания, щелкнем по ячейке памяти и выберем команду **Remove Break-Point** (Удалить точку прерывания).

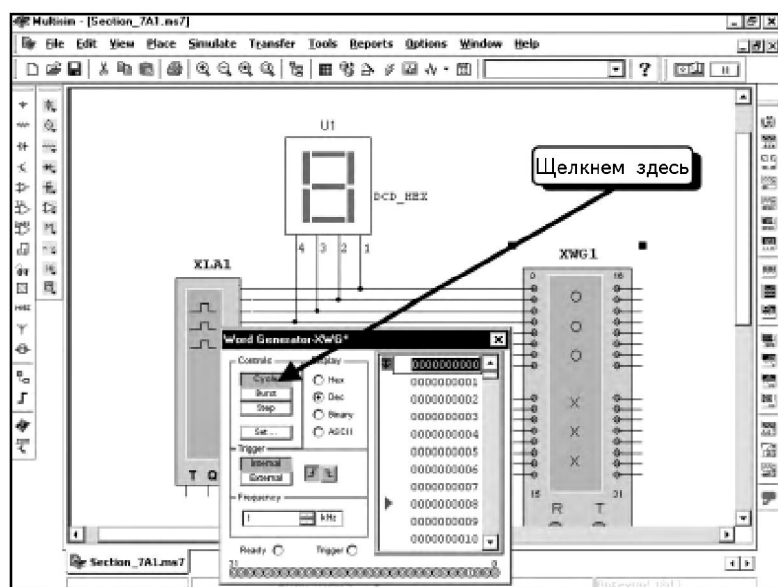


Во время моделирования курсор  в окне **Word Generator** указывает на текущую ячейку: В начале моделирования инструмент **Word Generator** начнет вывод данных с текущей ячейки. В процессе моделирования курсор будет перемещаться по ячейкам, которые обрабатываются инструментом **Word Generator**. Остановив моделирование, можно щелкнуть по ячейке и выбрать команду **Set Cursor** (Настроить курсор), чтобы изменить положение курсора.

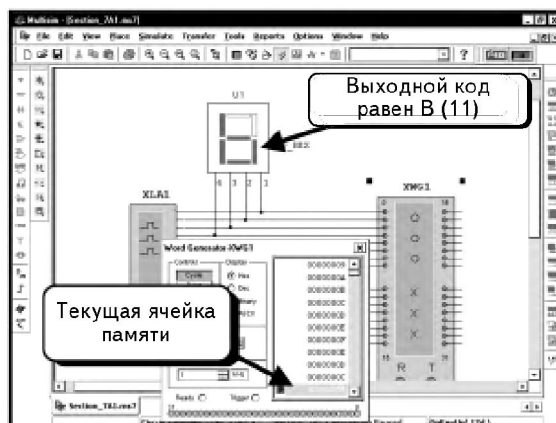
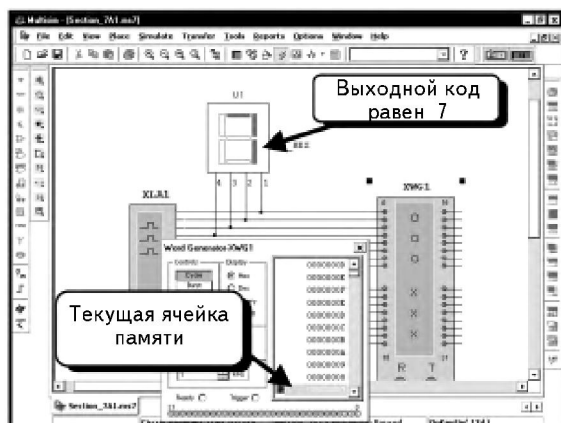
Поместим курсор в ячейку памяти 0:



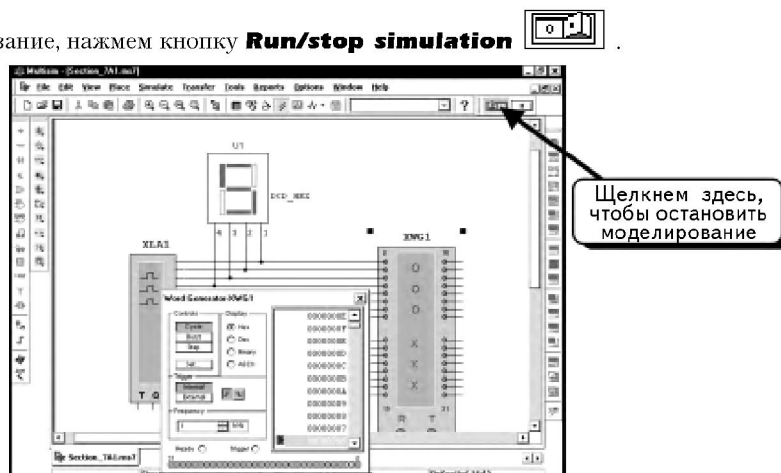
Все готово к моделированию. Убедимся в том, что окно **Word Generator** не закрывает цифробуквенный дисплей U1, и нажмем кнопку **Cycle**:



При нажатии этой кнопки начнется моделирование. Будет видно, как дисплей изменит значение от 0 до F, а потом — обратно до 0. Будет подсвечиваться также текущая ячейка памяти.

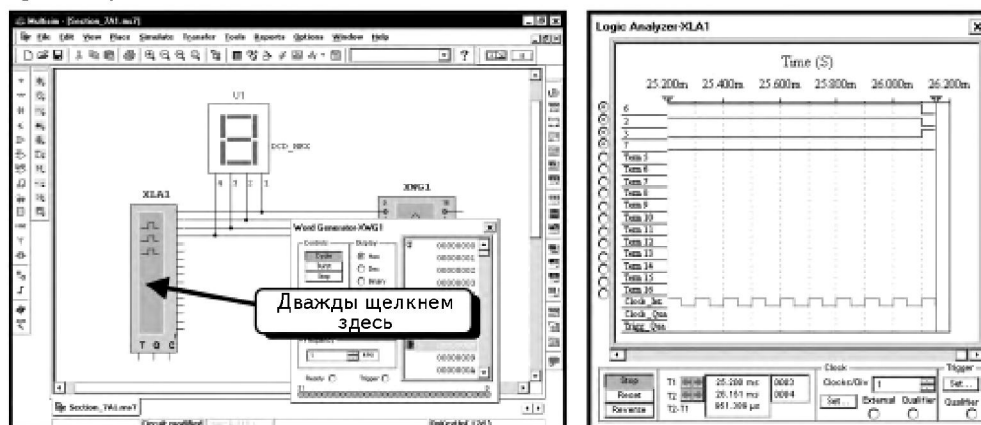


Чтобы остановить моделирование, нажмем кнопку **Run/stop simulation** .

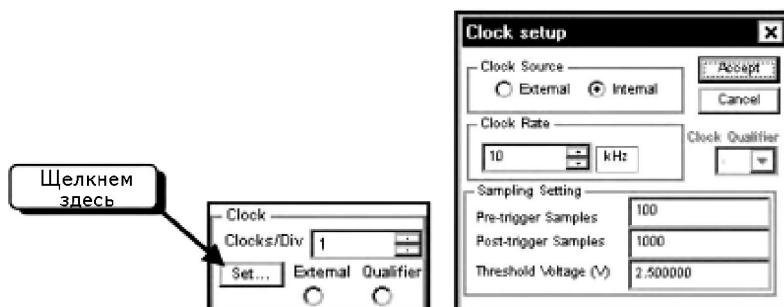


Пользуясь кнопками **Burst** и **Step**, можно изучить работу инструмента **Word Generator**. Помните, что в режиме **Step** для перехода к новой ячейке памяти надо каждый раз нажимать кнопку **Step**.

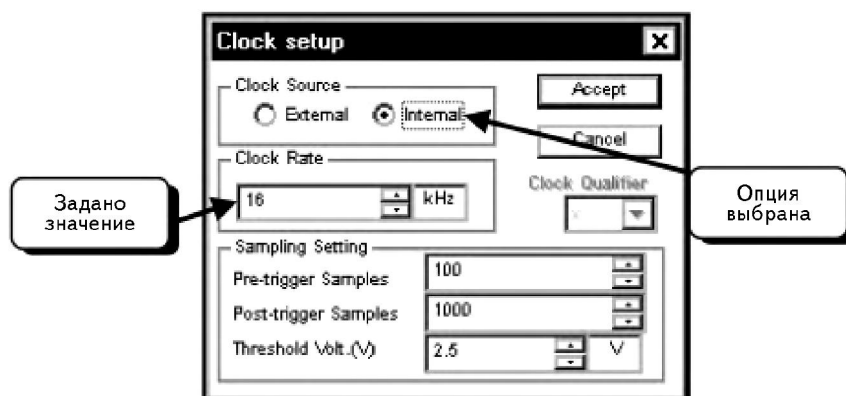
С помощью функции **Logic Analyzer** будем изучать диаграммы логических сигналов. Дважды щелкнем по иконке **Logic Analyzer**, чтобы открыть окно, показанное ниже:



В окне некоторые диаграммы отображаются неверно, поскольку инструмент **Logic Analyzer** пока не настроен. Щелкнем по кнопке **Set** в разделе **Clock (Таймер)**, чтобы изменить частоту выборки:



Таймер задает частоту выборки (sample rate) инструмента. В каждом цикле таймера **Logic Analyzer** анализирует состояние входных каналов и отображает его на экране. Даже если входные значения изменятся, значения на экране не сохранятся до следующего цикла. Это значит, что частота выборки должна намного превышать частоту изменения входного сигнала. Инструмент **Word Generator** был настроен на частоту 1 кГц, следовательно, параметр **Clock Rate (Частота таймера)** инструмента **Logic Analyzer** должен иметь более высокое значение. Выберем частоту, которая превышает значение 1 кГц в 2^n раз. Почему так надо сделать, будет ясно позднее. Выберем значение 16 кГц:

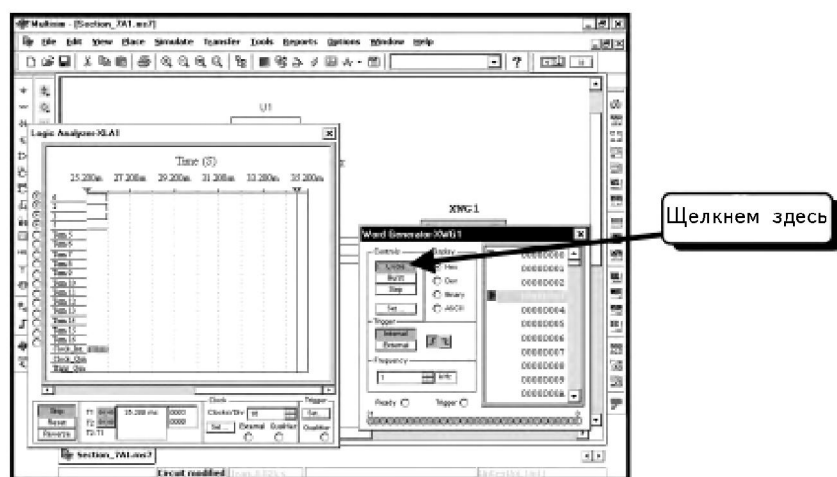


Убедимся, что параметр **Clock Source** (Источник синхронизации) настроен на режим **Internal** (Внутренний). Мы не обеспечиваем синхронизацию, поэтому **Logic Analyzer** самостоятельно сформирует необходимый сигнал. После этого нажмем кнопку **Accept**.

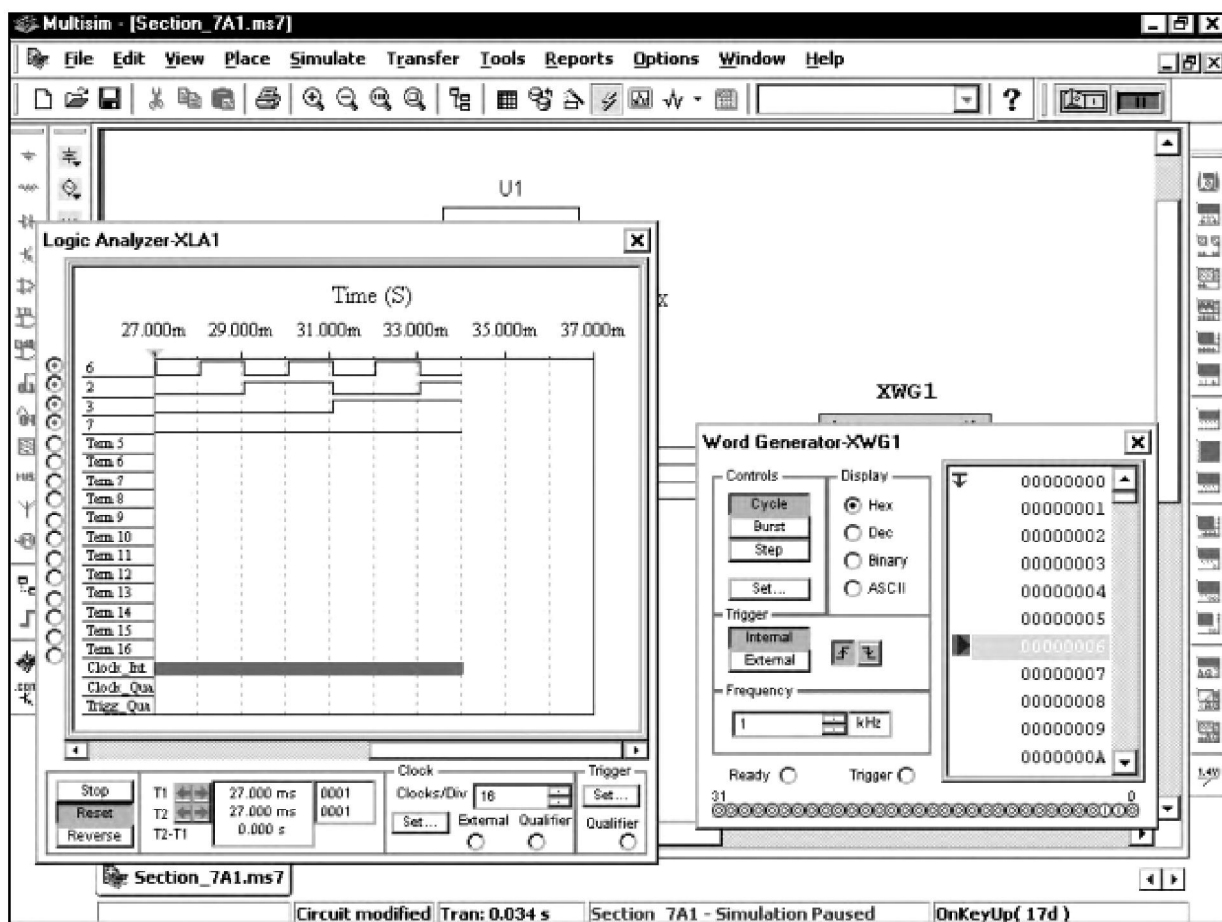
Настроим дисплей анализатора. На дисплее имеются деления, следовательно, необходимо указать количество импульсов таймера, приходящихся на деление (**Clocks/Div**). Можно выбрать значение 1, 2, 4, 8, 16 и так далее — до 128. Именно по этой причине выбрана частота таймера, превосходящая в 2^n раз частоту **Word Generator**. Для параметра **Clocks/Div** мы выберем значение 16. Так как частота таймера **Logic Analyzer** составляет 16 кГц, деление по времени будет равно 1 мс, что соответствует периоду частоты **Word Generator**:



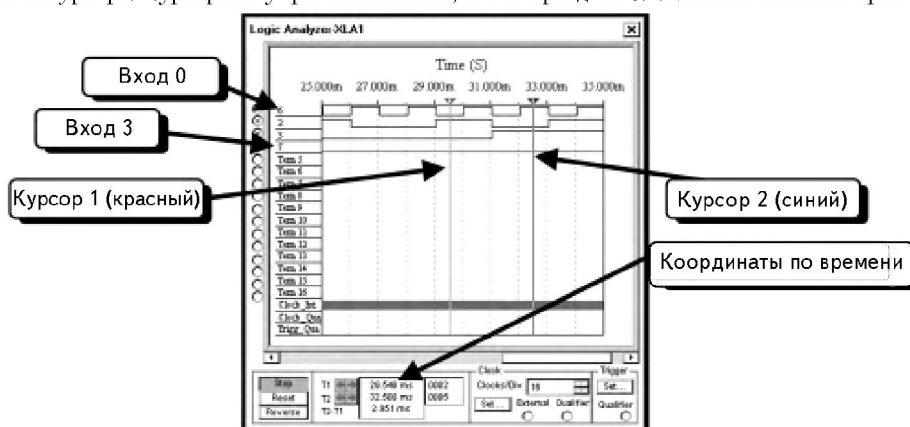
Все готово к моделированию. Нажмем кнопку **Cycle**:



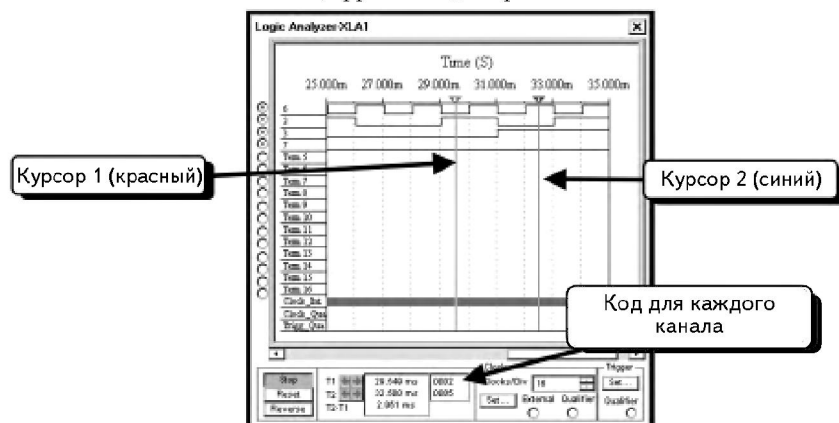
Начнется моделирование, и на экране **Logic Analyzer** появятся данные:



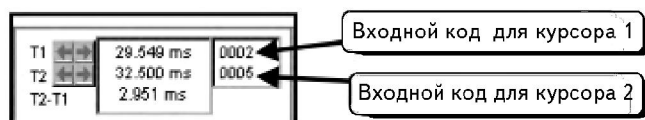
Если остановить моделирование, нажав кнопку **Run/Deactivate** , то можно измерить значения на экране с помощью курсоров. Курсор 1 — красный, а курсор 2 — синий. Чтобы переместить курсор, переместим символ в виде треугольника на курсоре. Курсорами управляем так же, как и в разделе 6.1.4. Ниже показан экран с курсорами:



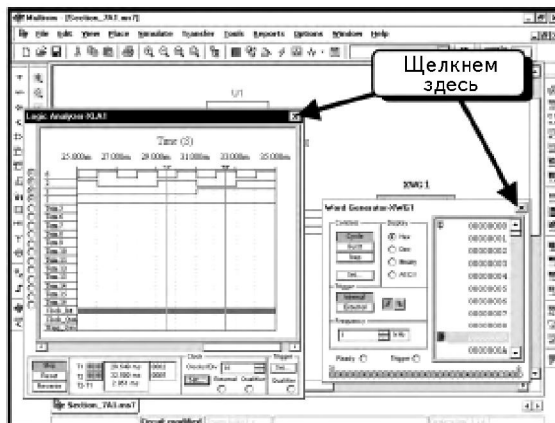
Курсоры показывают координаты по времени, а также временной сдвиг между курсорами. Кроме того, с помощью курсоров можно получить входные значения. Если вход 0 будем считать самым младшим разрядом, а вход 15 — самым старшим, то состояние входов **Logic Analyzer** может быть представлено 16-разрядным двоичным кодом. Соответствующий ему шестнадцатеричный код отображен в окне курсора:



Если посмотреть на курсор 1, то видно, что входы 0, 2 и 3 находятся на низком логическом уровне, а вход 1 — на высоком. Все прочие входы не используются, и диаграммы для них не выводятся. Таким образом, состояние входов в момент, соответствующий положению курсора 1, отображается числом 0010, то есть 2 (в шестнадцатеричном коде). Если посмотрим на курсор 2, то заметим, что входы 1 и 3 находятся на высоком уровне, а входы 2 и 4 — на низком. Таким образом, состояние входов, соответствующее положению курсора 2, отображается двоичным числом 0101, то есть 5 в шестнадцатеричном коде. Эта информация и показана в окне курсора:



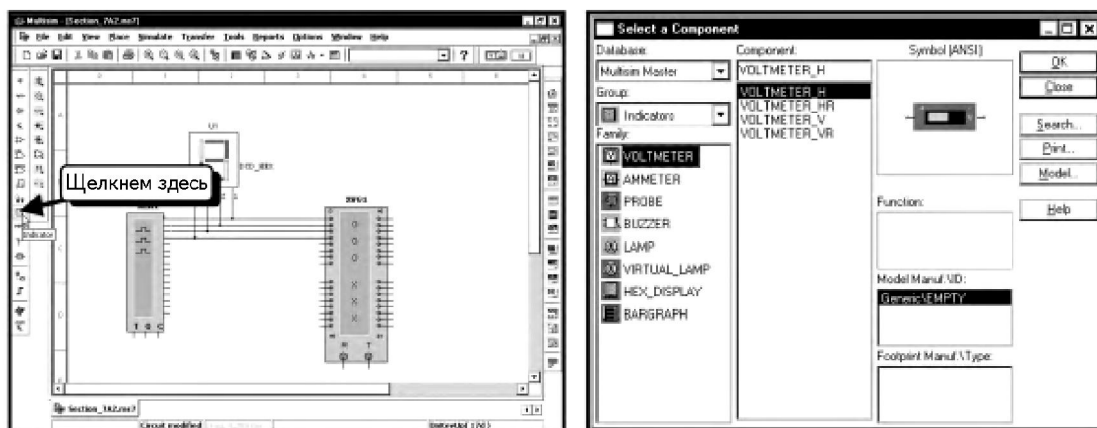
Измерения завершены и можно закрыть окна. Нажмем кнопку с символом  :



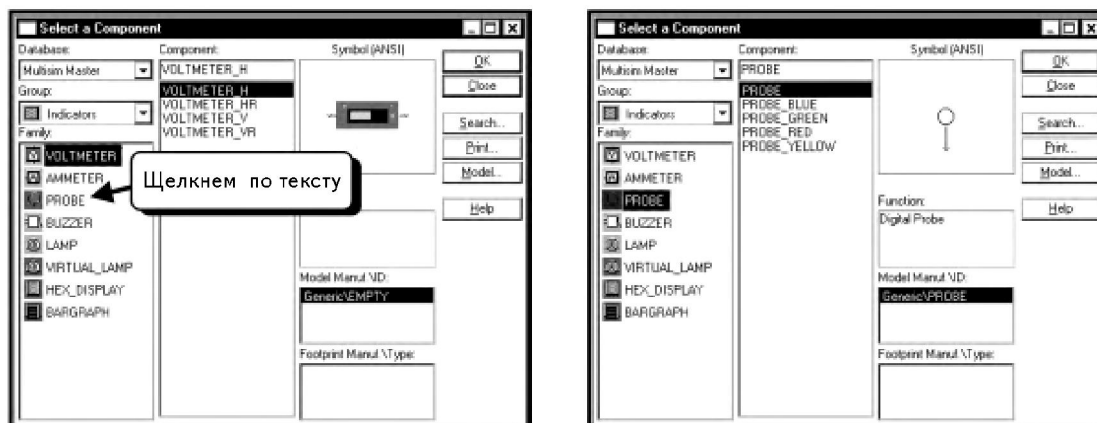
Работа с этой схемой будет продолжена в следующем разделе.

7.1.2. Цифровой пробник и недешифрующие столбиковые индикаторы

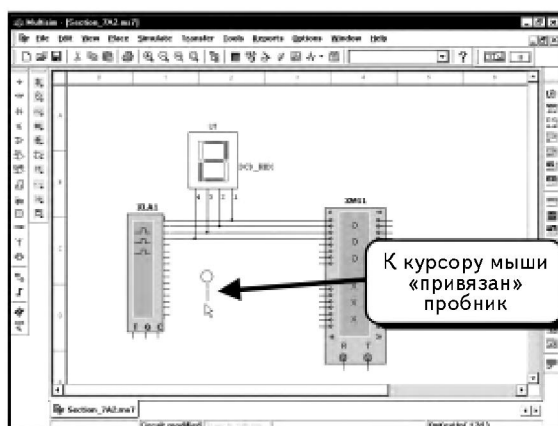
Далее рассмотрим два индикатора, которые могут использоваться для просмотра состояния одного бита. Логический пробник (digital probe) напрямую подключается к проводу и показывает его уровень. Недешифрующий столбиковый индикатор (undecoded bar graph) представляет собой набор из десяти диодов, аналогичный набору светодиодов. Он подключается так же, как и набор светодиодов, но отличается от него по ряду параметров. Диоды обладают внутренним сопротивлением и требуют тока 5 мА. Позднее об этом будет рассказано подробно. Логический пробник расположен в группе Indicators. Щелкнем кнопку **Indicator** :



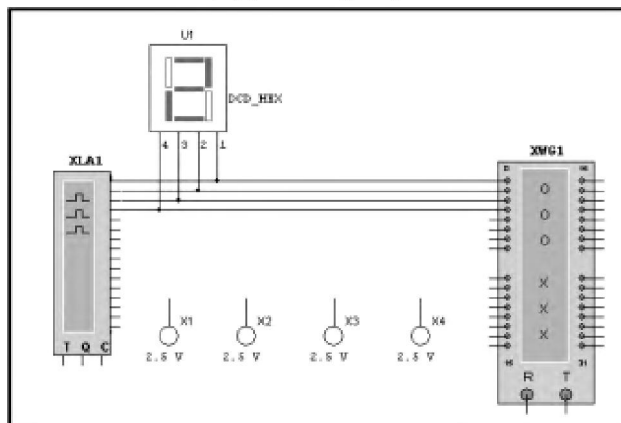
Щелкнем по объекту **PROBE** (Пробник), чтобы выбрать семейство:



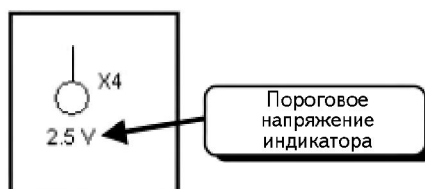
Нам доступны пять пробников. Один из них серый, другие — цветные. Выберем пробник и нажмем кнопку **OK**. Условное обозначение пробника будет «привязано» к курсору мыши:



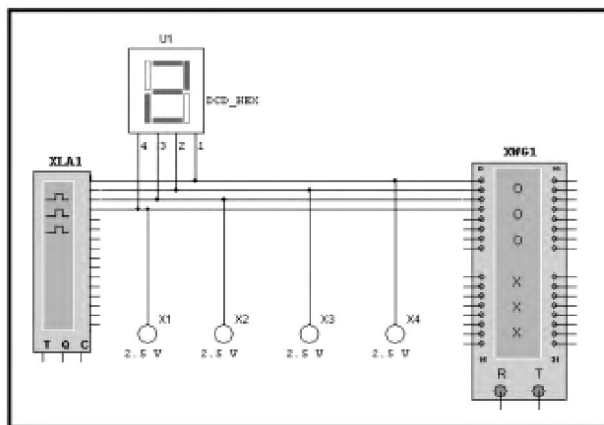
Введем в схему четыре пробника. Чтобы повернуть компонент, нажмем клавиши **CTRL+R**. Добавим в схему четыре разноцветных пробника:



Обратите внимание на запись **2.5 V** около условного обозначения пробника:

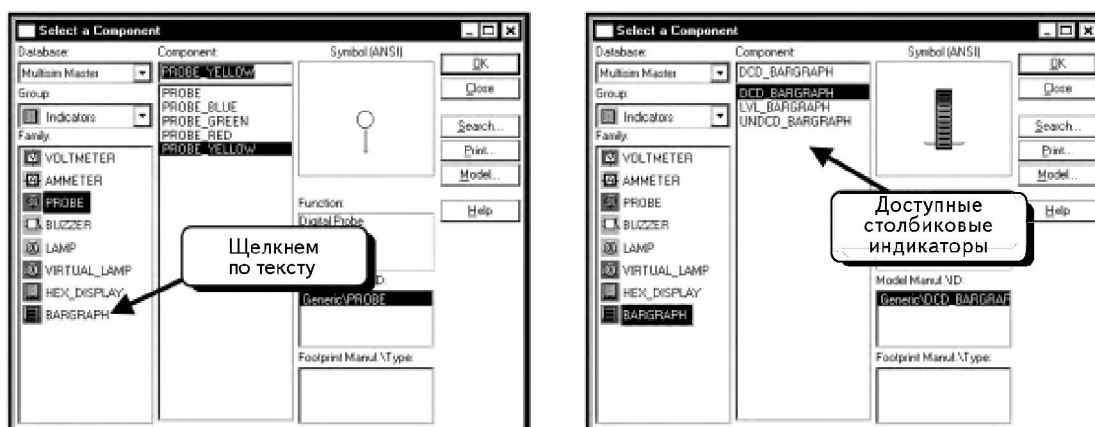


Это — пороговое значение напряжения для пробника. Он не будет светиться , если напряжение на нем меньше порогового, и засветится  при достижении напряжением порогового значения, которое будет показано на схеме, что упрощает настройку. Для нашего примера пороговое значение **2,5 В** является идеальным, поэтому не будем его изменять. Пробник не потребляет тока, следовательно, не нужно беспокоиться о его воздействии на схему. Подключим пробники, как показано ниже:

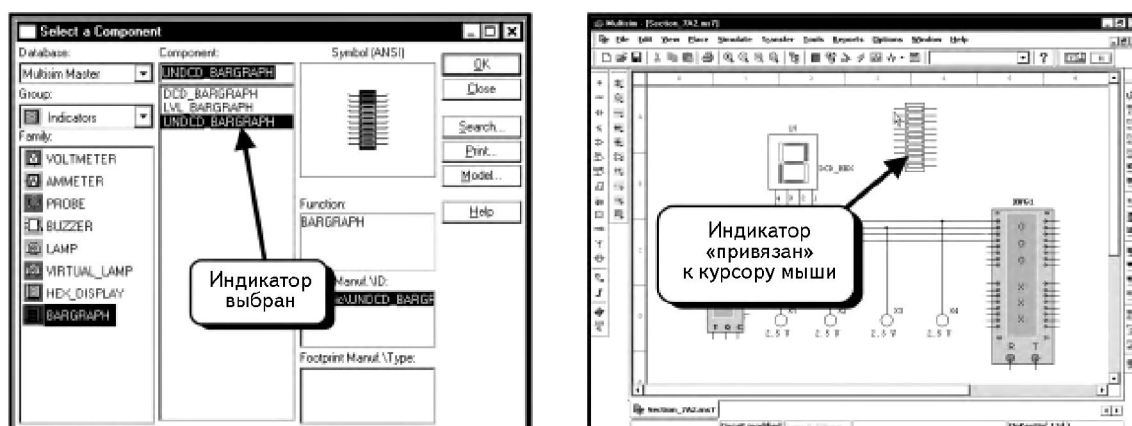


Теперь пробники отобразят состояние каждого входа инструмента **Word Generator**. Самый правый пробник (**X4**) — это самый младший бит, а самый левый (**X1**) — самый старший бит.

Добавим теперь в схему столбиковый индикатор. Этот индикатор тоже находится в группе **Indicators**. Щелкнем по объекту **BARGRAPH**, чтобы выбрать необходимое семейство:

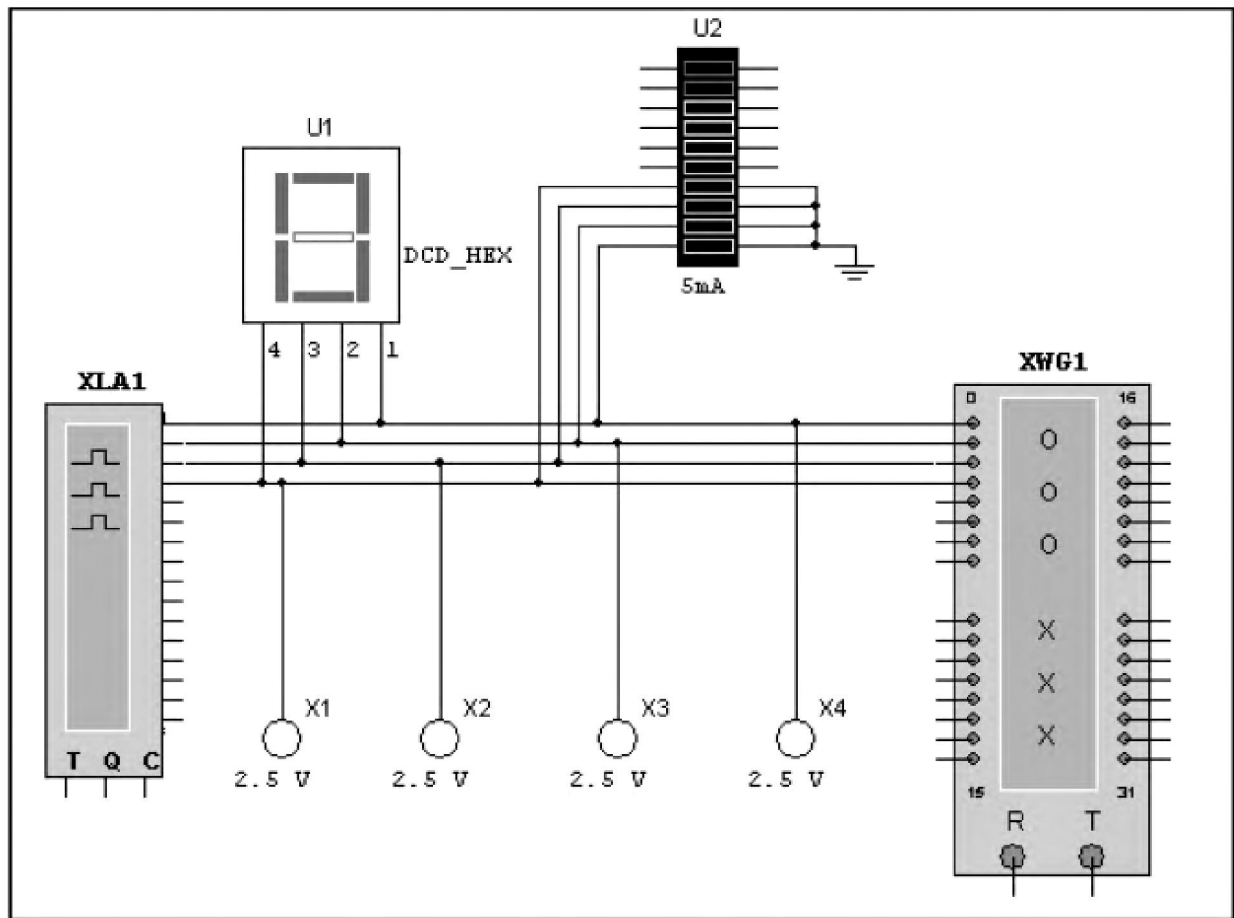


Диалоговое окно отобразит список всех доступных индикаторов. Нам необходимо использовать индикатор **UND CD_Bargraph**. Выделим его и нажмем кнопку **OK**. Компонент будет «привязан» к курсору мыши:

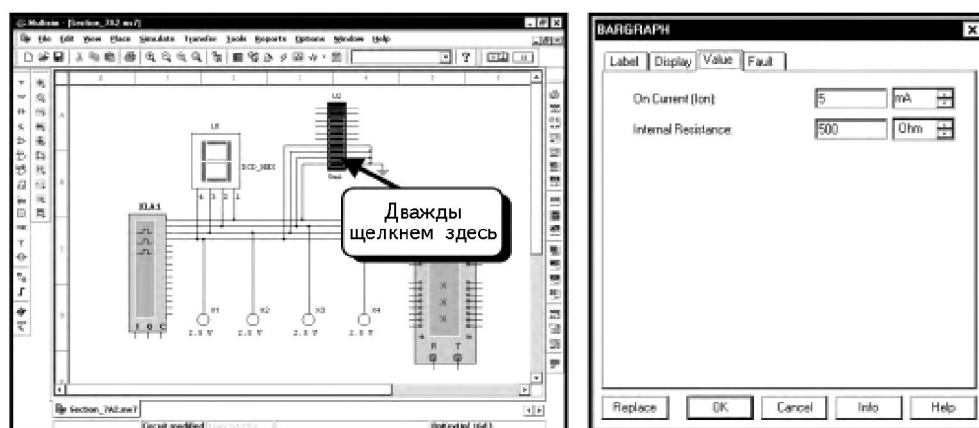


Он представляет собой набор из десяти диодов. Однако они не являются светодиодами. Ток может проходить через них в обоих направлениях, причем они не имеют прямого падения в 1,5 В, характерного для светодиодов. Вместо этого, каждый диод имеет внутреннее сопротивление 500 Ом и характеризуется определенным пороговым значением тока. Как показано выше, когда ток, не превышающий 5 мА, входит в левый контакт диода и выходит из правого контакта (то есть движется слева направо), диод не будет светиться. Эта функция аналогична функции светодиода, хоть и не полностью. Можно изменять как пороговое значение, так и сопротивление компонента. Если перевернуть компонент, то контакты также перевернутся. При начальном размещении компонента расположение контактов известно. Следите за ним при вращении компонента.

Добавим индикатор в схему и подключим его, как показано ниже. Нам не понадобятся дополнительные последовательные резисторы (ограничивающие ток через светодиоды), так как компонент обладает внутренним сопротивлением:

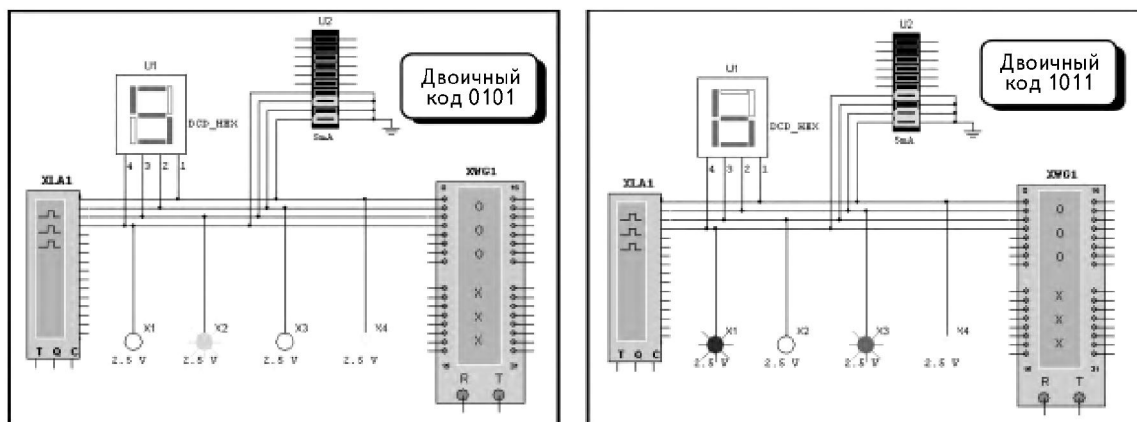


Нет необходимости использовать управляющие схемы, потому что инструмент **Word Generator** может создавать ток, достаточный для свечения диодов. Дважды щелкнув по индикатору, можно увидеть, что пороговое значение составляет 5 мА, а внутреннее сопротивление равно 500 Ом:



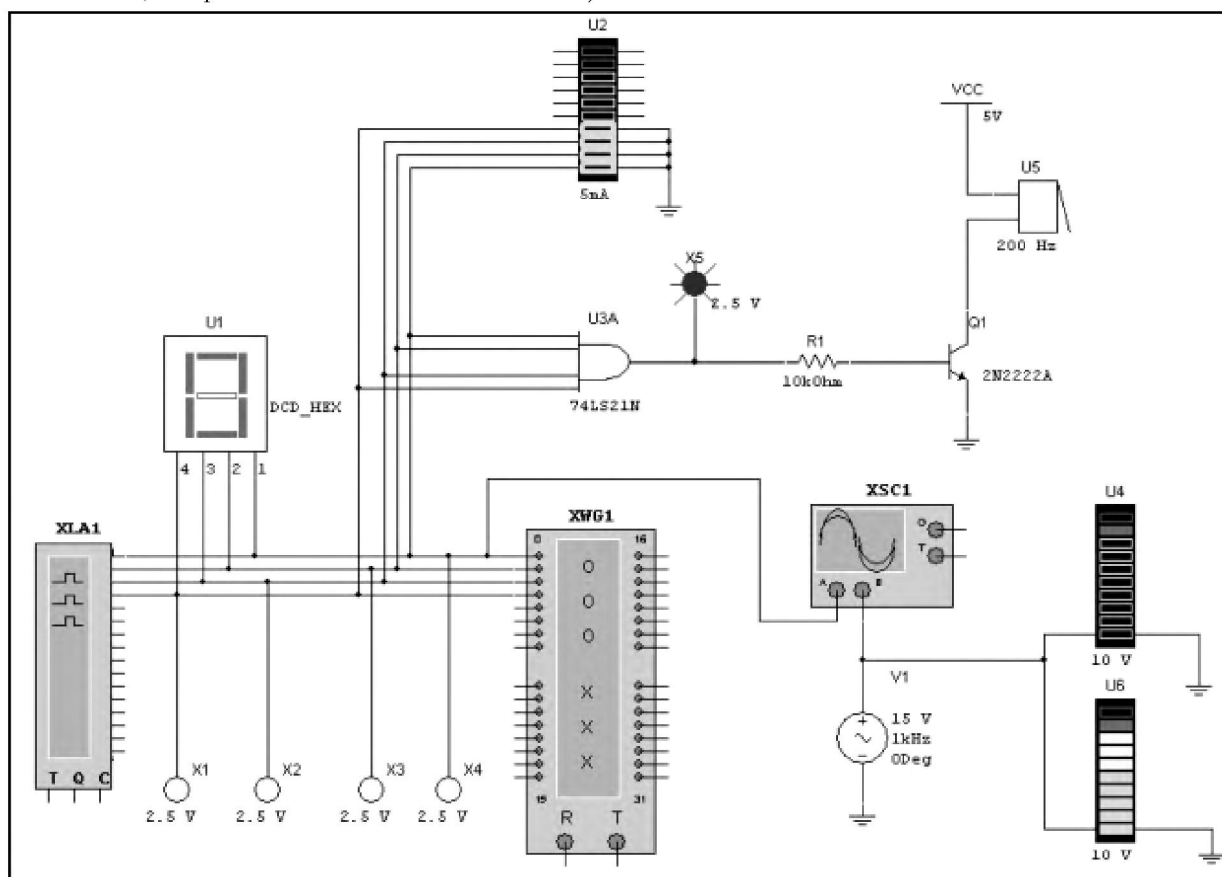
Нажмем кнопку **OK**, чтобы вернуться к схеме:

Теперь все готово к моделированию. Нажмем кнопку **Cycle**, чтобы начать его. Индикатор и пробники отображат двоичные коды, которые соответствуют данным в окне с информацией:

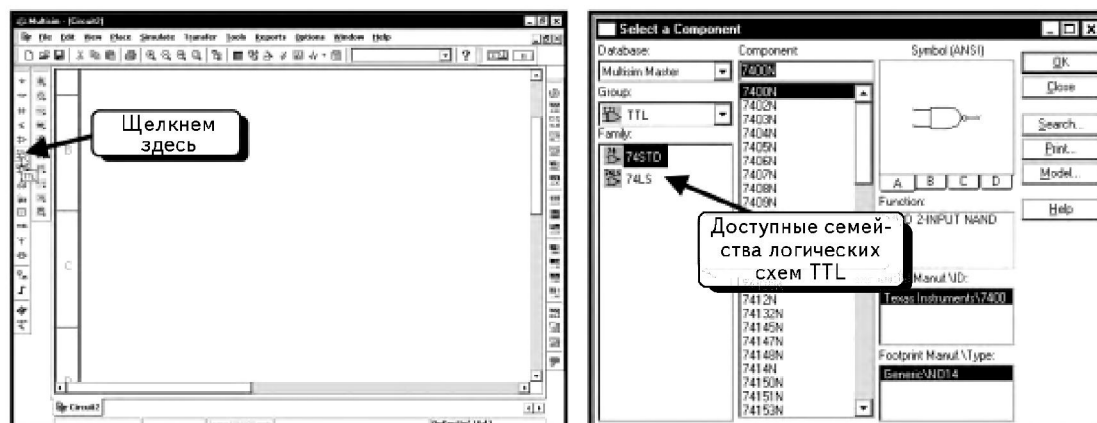


7.1.3. Различные индикаторы сигналов

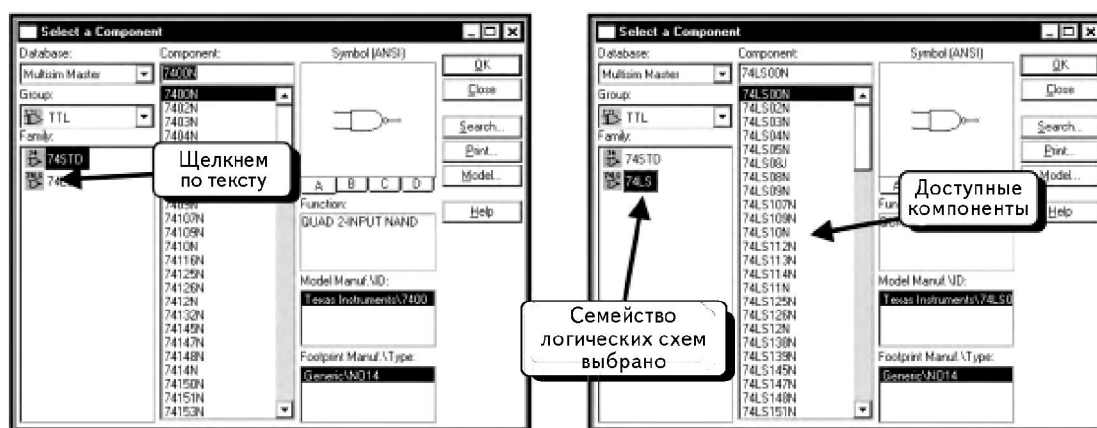
Добавим еще в схему зуммер и проверим работу индикаторов. Воспользуемся также осциллографом, чтобы получить две осциллограммы. Создадим следующую схему (В этой главе использованы некоторые аналоговые компоненты, которые были описаны в 3–6-й главах.):



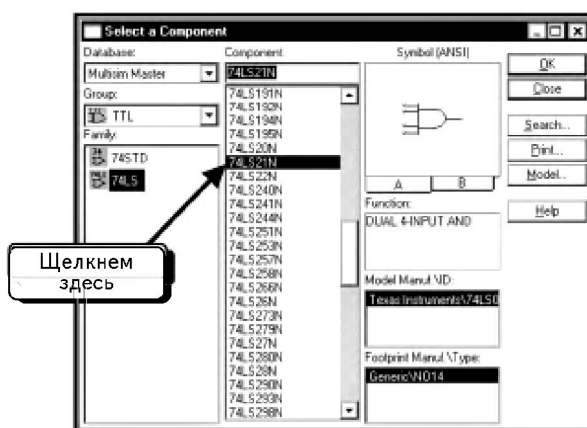
Зуммер находится в группе **Indicators**; добавим два столбиковых индикатора с помощью методики, которая использовалась при добавлении индикаторов в предыдущем разделе. Компоненты U4 и U6 называются **LVL_Bargraph** и **DCD_Bargraph** и находятся в семействе **Bargraph** группы **Indicators**. Компонент **AND** расположен в группе **TTL**. Чтобы добавить компонент **AND** (схему И), нажмем кнопку **TTL** :



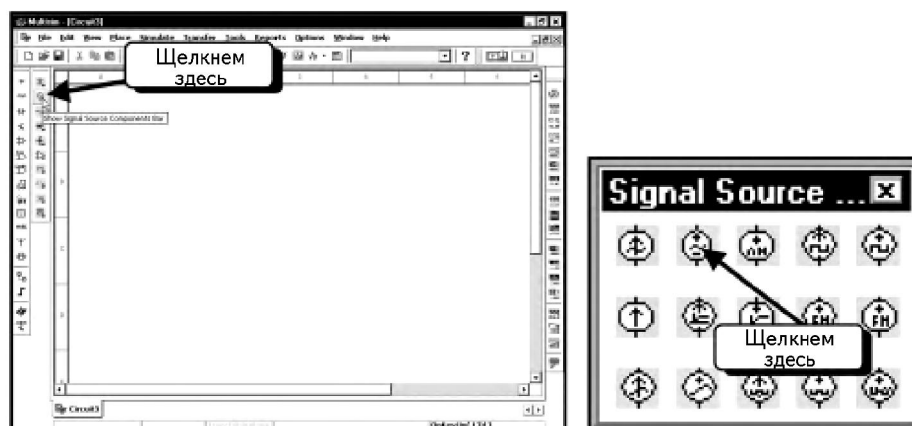
На левой панели показаны доступные семейства компонентов TTL. Видно, что доступны семейства TTL и Schottky TTL. Будем использовать компоненты Шоттки; выберем микросхему **74LS**:



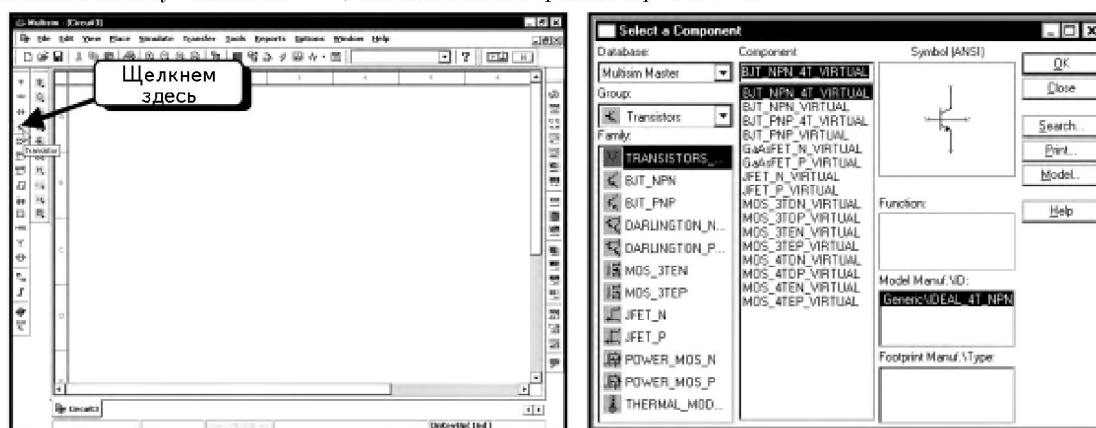
В диалоговом окне отображены все доступные компоненты Шоттки. Найдем компонент 74LS21N и выделим его:



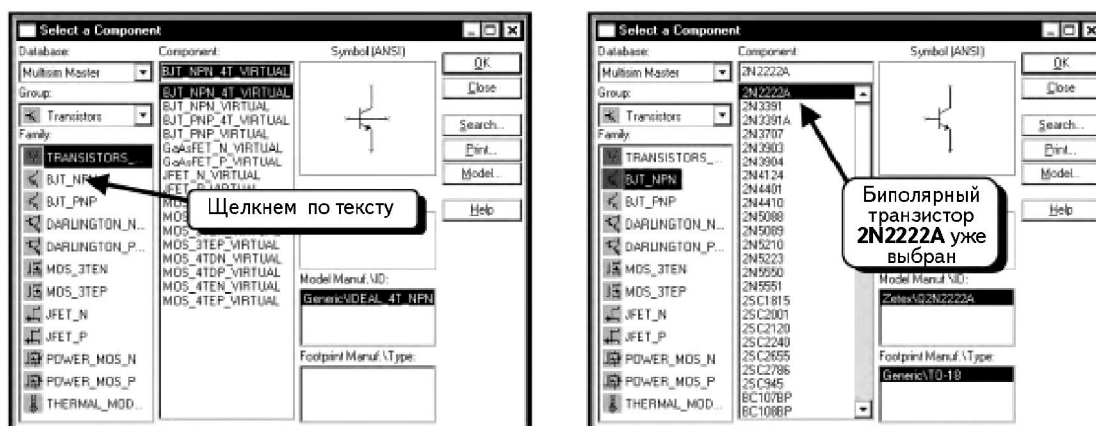
Нажмем кнопку **OK**, чтобы добавить компонент. Добавим также источник переменного напряжения, нажав кнопки **Show Signal Source Components Bar**  и затем **AC VOLTAGE SOURCE**  :



Нажмем кнопку **Transistor** , чтобы добавить транзистор 2N2222A:

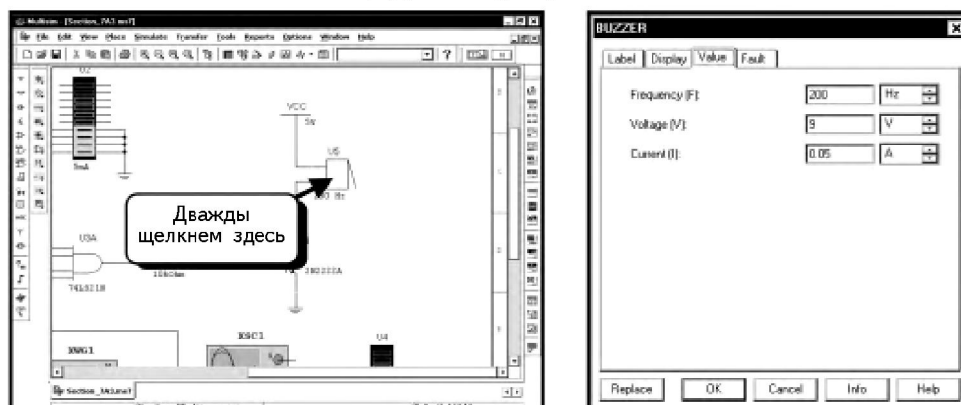


Транзистор 2N2222A входит в семейство NPN биполярных транзисторов; выберем семейство **BJT_NPN**:



В этом диалоговом окне показан список всех доступных транзисторов NPN. Выделенный транзистор — это именно тот компонент, который нужен. Нажмем кнопку **ОК**, чтобы добавить его в схему.

Теперь соединим все компоненты. Рассмотрим некоторые из них подробнее. Изменим настройки зуммера. Дважды щелкнем по его изображению, чтобы изменить параметры:



Частота зуммера равна 200 Гц, рабочее напряжение — 9 В при токе 50 мА. Изначально схема была настроена на напряжение 4 В; изменим его значение:



При желании можно изменить частоту и ток зуммера. Однако слишком большое значение тока может вызвать проблемы при работе с транзистором 2N2222A. Нажмем кнопку **OK**, чтобы принять изменения. Если инструмент **Word Generator** создаст код 1111, схема **И** определит это, и зуммер издаст сигнал.

Компоненты **LVL_BARGRAPH** и **DCD_BARGRAPH** являются аналоговыми индикаторами. Это индикаторы с одним аналоговым входом, которые преобразуют входной сигнал в уровень столбика на дисплее. В них встроены логические схемы, позволяющие преобразовать аналоговый сигнал в десять цифровых сигналов, а также дополнительные схемы, отображающие логические сигналы на дисплее. Необходимо лишь поместить компоненты в схему и подключить входы к заземлению и измеряемому сигналу. В нашей схеме индикаторы подключены к синусоидальному источнику напряжения 15 В, поэтому показания будут изменяться в соответствии с синусоидой напряжения.

Перейдем теперь к моделированию. Нажмем кнопку **Cycle**. Каждый раз при считывании кода 1111 зуммер должен издавать звук. Высота столбиков в индикаторах повторяет изменения в синусоиде напряжения. Изображения, помещенные ниже, отображают этот процесс:

