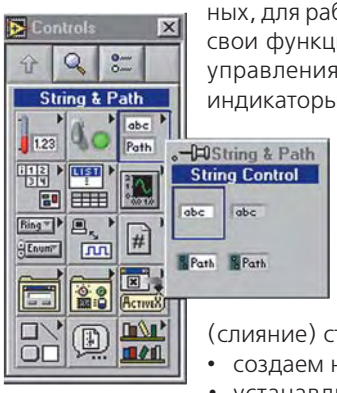
Уроки по LabVIEW

На этом уроке Вы научитесь:

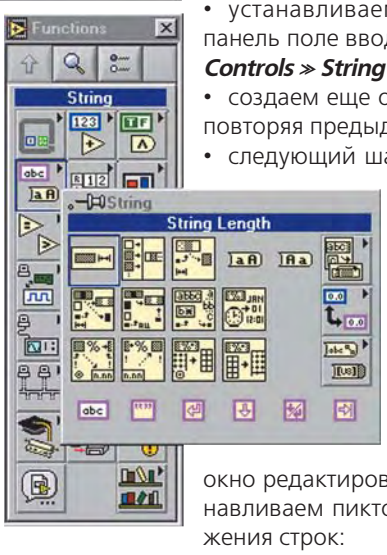
- ***работать со строковыми переменными и использовать функции преобразования;***
- ***освоите технологию изменения свойств элементов управления и индикаторов;***
- ***познакомитесь с технологией изменения свойств элементов управления и индикаторов;***
- ***создадите приложения, в которых будут применены функции генерации сигналов, массивы, строки, а также операции визуализации данных, сохраненных в файле.***

Как и во всех языках программирования высокого уровня, в LabVIEW также реализована работа со строками. Строки в LabVIEW - это еще один тип данных, для работы с которыми существуют свои функции, индикаторы и элементы управления. Элементы управления и индикаторы размещены на панели управления ***Controls ≫ String & Path***. А функциональные элементы, соответственно, в ***Functions ≫ String***.

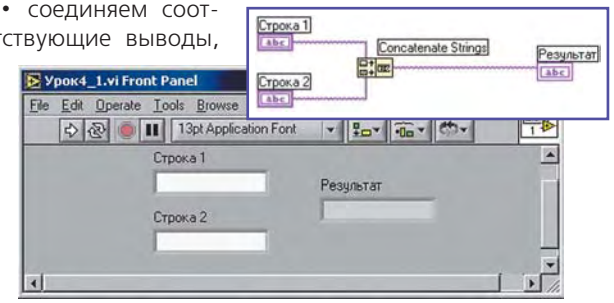


"Напишем" элементарную программу, которая реализует конкатенацию (слияние) строк:

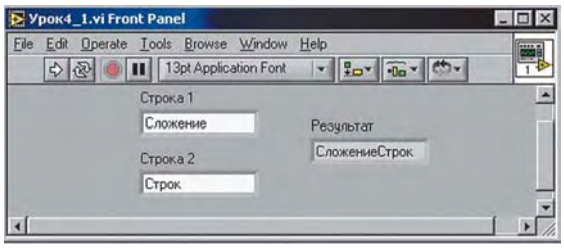
- создаем новое приложение;
- устанавливаем на интерфейсную панель поле ввода строки (строк): ***Controls ≫ String & Path ≫ String Control***;
- создаем еще один такой же элемент, повторяя предыдущий шаг;
- следующий шаг - текстовый индикатор для вывода результата. Устанавливаем его на панель: ***Controls ≫ String & Path ≫ String Indicator***.



- переименуем все установленные на панели элементы, как показано ниже;
- переключаемся в окно редактирования диаграмм и устанавливаем пиктограмму-функцию сложения строк: ***Functions ≫ String ≫ Concatenate Strings***.
- соединяем соответствующие выводы, как показано на диаграмме, и программа готова.



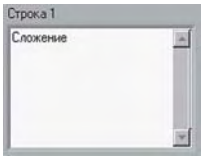
Выполним тестирование написанной программы. Заполняем первую и вторую строку ввода произвольной последовательностью символов, и запускаем программу на выполнение:



Вы уже привыкли к тому, что изменение свойств объектов, рассмотренных в предыдущих уроках, сводилось к нажатию правой клавишей мыши на объекте и выбором необходимой опции. Строки не являются исключением. Если Вы хотите вводить не одну строку, а несколько, то необходимо просто изменить размеры элемента ввода или индикатора.

Добавим теперь полосу прокрутки для области ввода текста. Для этого нажимаем правой клавишей мыши на выбранном объекте и выбираем в появившемся меню свойство:

Visible Items ≫ Scroll Bar.



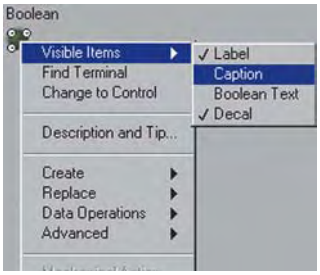
Теперь можно вводить и изменять введенный текст, как в простом текстовом редакторе. Нажимая клавишу мыши на строке ввода, Вы непременно должны были обратить внимание на опцию ***Password Display***. Нетрудно догадаться, что эта опция переключает поле ввода в режим, предназначенный для ввода пароля. При этом вместо вводимых символов, поле будет заполняться звездочками.

Давайте напишем простую программу, которая при правильном или неправильном вводе пароля информирует об этом пользователя путем изменения цвета овальной "лампочки-индикатора":

- установим поле ввода на интерфейсную панель: ***Controls ≫ String & Path ≫ String Control***;

• изменим свойства этого поля. Нажав правой клавишей мыши на объекте, в выпадающем меню выберем опцию ***Password Display***. Заодно и переименуем объект;

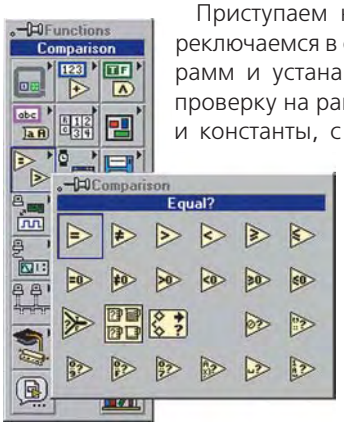
• установим овальный (получается из круглого) индикатор на панель ***Controls » Boolean » Round Led***. Во всплывающем меню, в пункте ***Visible Items*** необходимо убрать галочку с метки ***Label***, а на ***Boolean Text***, наоборот, ее установить;



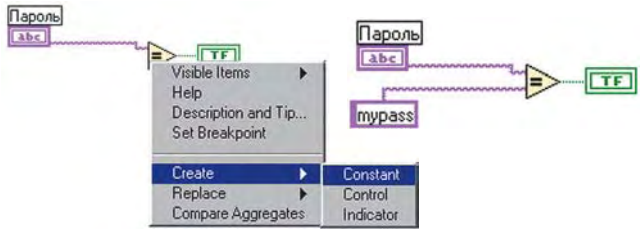
• изменяем размер индикатора, текст надписи - ***Caption***, пассивный и активный цвета. Пассивный цвет сделаем малиновым, а активный - салатовым. Для изменения цвета необходимо подвести указатель мыши к палитре инструментов и нажать на верхнем квадрате. Далее следует выбрать желаемый цвет;

• ту же операцию нужно проделать и для нижнего квадрата. Потом подвести курсор в виде кисти к индикатору и нажать левую клавишу мыши. Объект изменит свой цвет. Чтобы задать цвет для другого режима индикатора, необходимо в панели инструментов выбрать указатель в виде пальца. Затем нажать им на индикаторе. Индикатор "переключится". Меняем текст надписи и цвет.

Приступаем к программированию. Переключаемся в окно редактирования диаграмм и устанавливаем знак сравнения - проверку на равенство введенного пароля и константы, с которой он сравнивается: ***Functions»Comparison»Equal?***

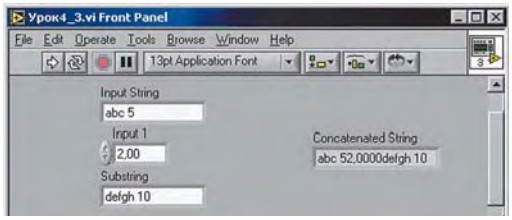


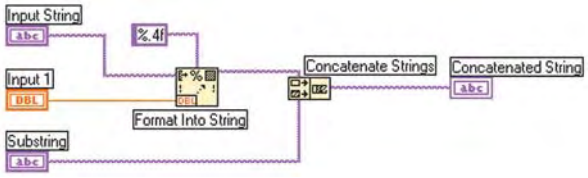
Соединяем строку ввода и индикатор с соответствующими выводами. Для создания константы, с которой будет сравниваться вводимая строка, необходимо подвести указатель мыши в виде катушки ко второму выводу функции сравнения и нажать правую клавишу мыши. Теперь можно создать константу:



Далее необходимо ввести значение константы-пароля, например "mypass". Все, можно проверить программу на работоспособность. Для этого, необходимо переключиться на интерфейсную панель. Сначала введите верный пароль и запустите на исполнение, а затем - неверный. Проанализируйте получившийся результат. Попробуйте модифицировать эту программу.

А теперь "пощупаем" функции преобразования чисел в строку. Составьте программу, как показано ниже.





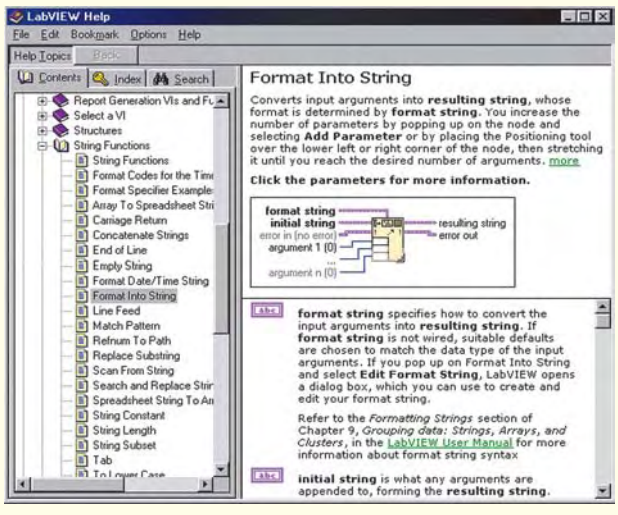
Причем для сложения строк используйте уже известную Вам функцию конкатенации, а для преобразования числа в строку - новую функцию из того же функционального меню:

Functions»String»Format Into String.

Проведите все необходимые связи, создайте константы, и введите необходимые значения. Запускаем программу.

Следует обратить особое внимание на константу, которая определяет формат числа. Так, запись %.4f означает, что вводимое число будет преобразовано в строку, где 4 - это количество знаков после запятой.

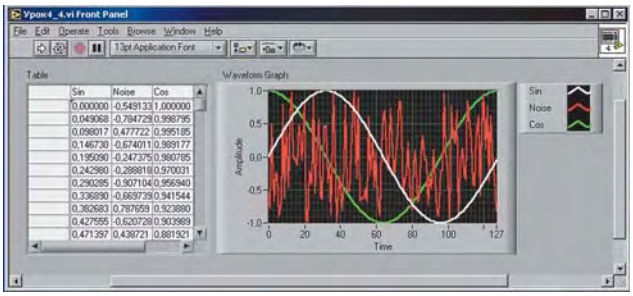
Совет: За более подробной информацией о той или иной функции следует обращаться в Справку. Нажав на правую кнопку мыши в области произвольной функции в окне редактирования диаграмм, и выбрав в выпадающем меню Help, Вы немедленно получите подробную информацию об используемой функции или блоке.



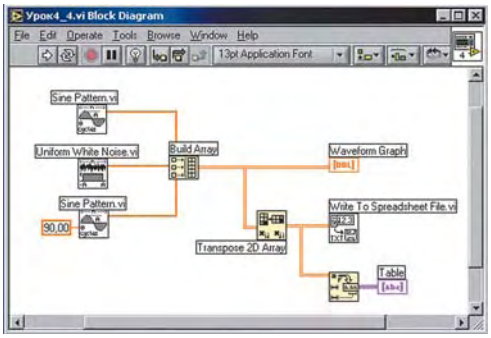
Создадим программу, которая будет генерировать 2-х мерный массив из 128 строк и 3-х столбцов. Первый столбец будет содержать данные синусоидальной волны, второй - шумовую волну, а третий - косинусоидальную волну. Кроме этого, результат формирования волн будет отображаться в виде графиков одной области и в табличном виде.

Вызов таблицы – *Controls»List & Table*.

Чтобы таблицу преобразовать из элемента управления в индикатор необходимо при нажатии правой клавиши мыши на объекте выбрать пункт ***Change to Indicator.*** Программа может иметь, например, такой внешний вид:



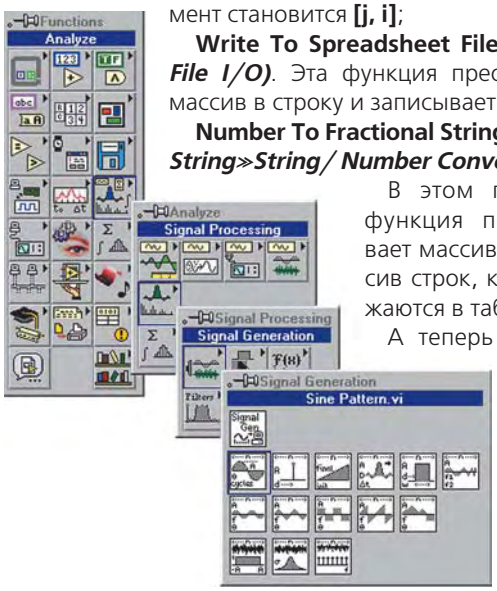
При этом "листинг" ее должен быть следующим:



Самостоятельно реализуйте рассмотренный пример, используя для генерации сигналов функции, которые можно найти на функциональной панели:

Build Array *(Functions»Array)*. В этом упражнении эта функция создает 2-мерный массив из 3-х зависимостей: синуса, шума и косинуса;

Transpose 2D Array *(Functions»Array)*. Эта функция перегруппировывает массив так, что каждый **[i, j]** элемент становится **[j, i]**;



Write To Spreadsheet File *(Functions» File I/O)*. Эта функция преобразовывает массив в строку и записывает ее в файл;

Number To Fractional String *(Functions» String»String/ Number Conversation)*.

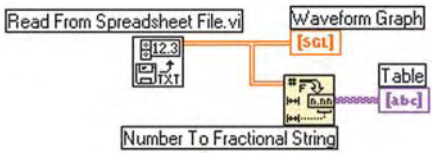
В этом примере эта функция преобразовывает массив чисел в массив строк, которые отражаются в таблице.

А теперь рассмотрим пример чтения данных из файла и представления их в виде графиков и таблицы. Последовательность уже знакомых действий будет следующая:

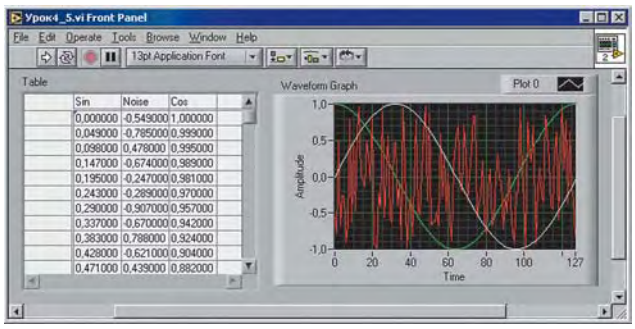
- создаем новое приложение;
- устанавливаем на интерфейсную панель таблицу. Преобразовываем ее из элемента управления в индикатор и оформим ее внешний вид, как показано ниже;
- устанавливаем элемент ***Waveform Graph***;
- в окне редактирования диаграмм устанавливаем следующие компоненты

Number To Fractional String,
Read From SpreadSheet File.vi;

- выполняем необходимые соединения и запускаем программу;

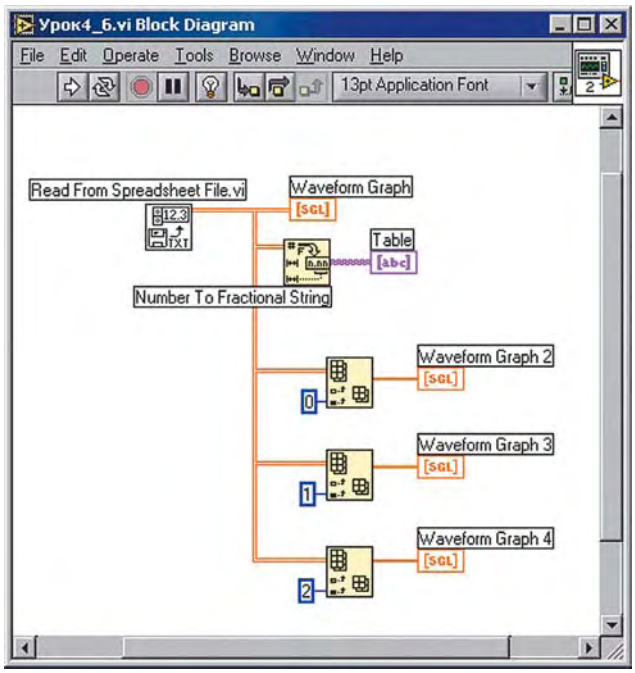


- после запуска, в появившемся окне необходимо выбрать файл данных (созданный в предыдущем примере).



Немного модифицируем созданную программу - разделим двумерный массив на три одномерных и построим графики каждого сигнала отдельно. Для этого воспользуемся уже известной функцией работы с массивами ***Functions » Array » Index Array***. С ее помощью будем выделять столбцы массива.

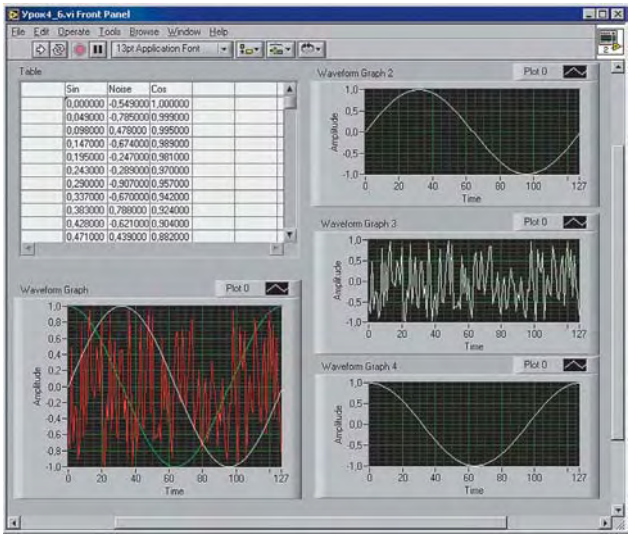
- добавляем еще три **Waveform Graph**;



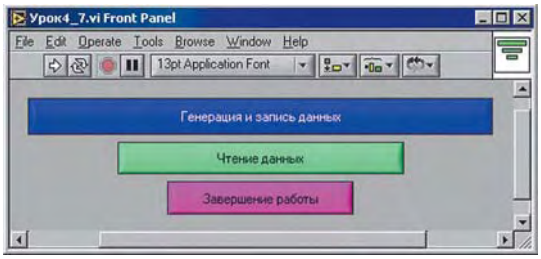
• в окно редактирования диаграмм устанавливаем три **Index Array**, соединяем с исходным массивом и создаем константы (0, 1, 2), определяющие номер столбца массива, а значит и тип зависимости;

• корректируем соединения и запускаем программу.

В результате получаем новую программу:



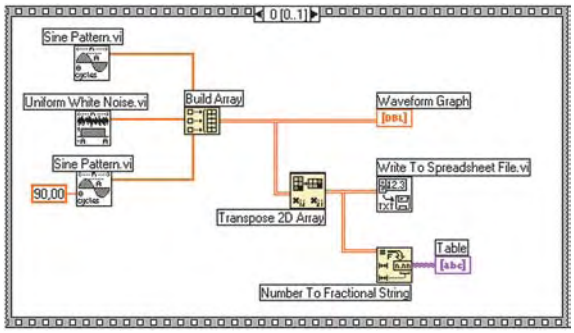
Если Вы успешно выполнили все рассмотренные примеры, то будем считать - Вы готовы создать приложение, которое позволит записывать данные в файл, считывать и визуализировать их, а также корректно завершать работу с созданным приложением. Причем каждую из функций реализуем в виде отдельных виртуальных инструментов, которые будут "вызываться" главной программой.



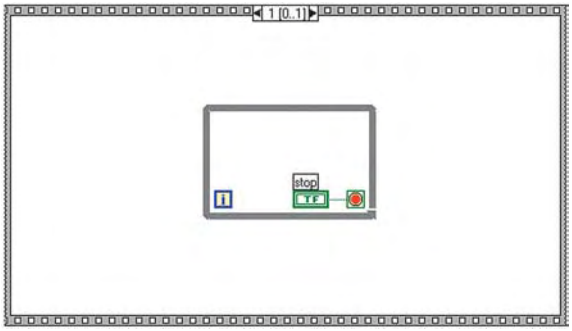
В качестве основных блоков будем использовать уже созданные в предыдущих примерах виртуальные инструменты, которые необходимо только чуть-чуть модифицировать. Займемся первой подпрограммой:

• установим в окно редактирования диаграмм структуру "последовательность";

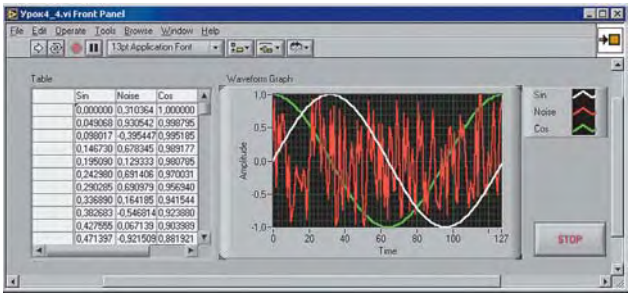
• в нулевой кадр помещаем уже существующую программу генерации сигналов, записи в файл и представления их в таблице и на графике;



• создаем первый кадр, а в нем - пустой цикл **WhileLoop**, который будет "удерживать" программу активной (в запущенном состоянии), пока не будет нажата клавиша **"Завершение работы"** ***(STOP)*** на интерфейсной панели.



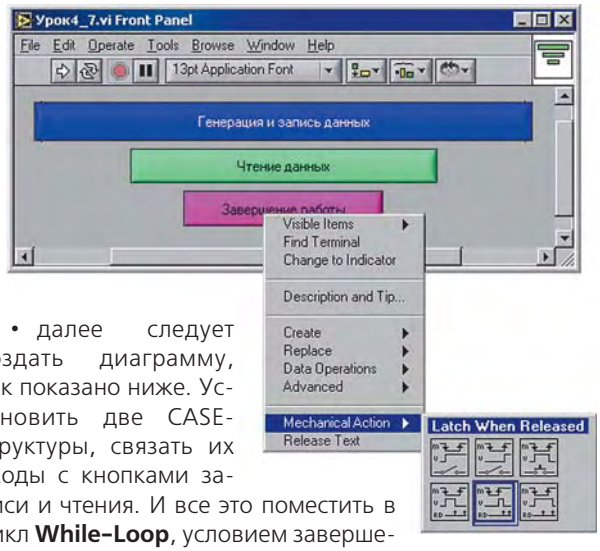
• модифицируем также иконку программы, дважды щелкнув левой клавишей мыши на иконке в правом верхнем углу;



• аналогично модифицируем и подпрограмму, которая считывает и визуализирует данные;

Можно переходить к написанию главной программы:

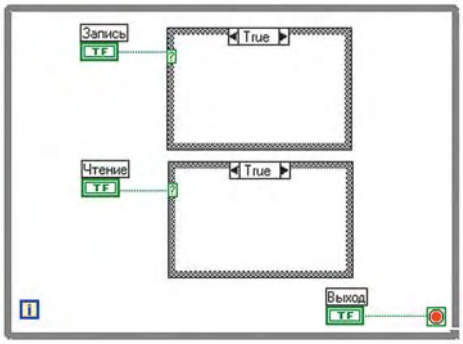
• для начала, создадим графический интерфейс, который состоит из трех кнопок с соответствующими надписями, для которых установлена опция программной реакции по отпусканию клавиши;



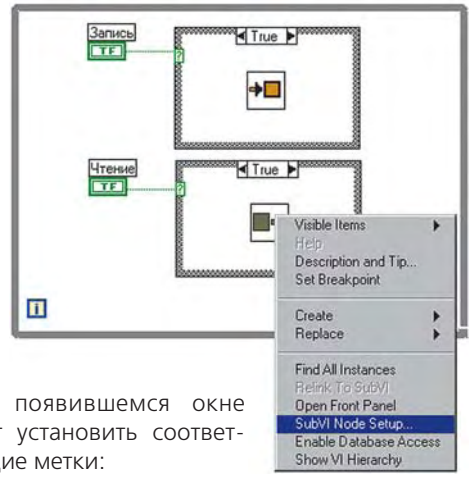
• далее следует создать диаграмму, как показано ниже. Установить две CASE-структуры, связать их входы с кнопками записи и чтения. И все это поместить в цикл **While-Loop**, условием завершения работы которого будет нажатие на соответствующую кнопку;

• выполним установку подпрограмм в окно редактирования диаграмм ***Functions » Select VI...***;

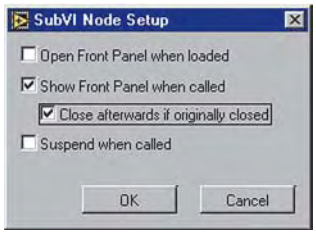
• теперь следует сконфигурировать работу подпрограмм так, чтобы при вызове (нажатии на соответствующую клавишу) открывалась передняя панель подпрограммы, а по завершению работы с ней - исчезала. Для



этой цели необходимо нажать правую клавишу мыши на выбранной подпрограмме, и в выпадающем меню выбрать опцию конфигурирования подпрограммы ***(SubVI Node Setup)***.



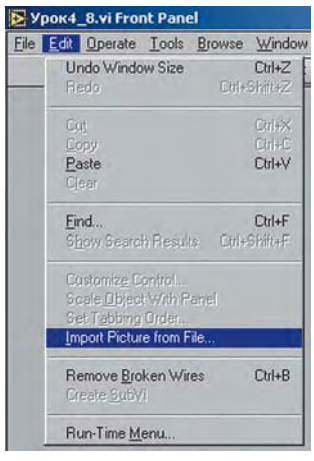
В появившемся окне следует установить соответствующие метки:



Остается только запустить программу на выполнение и убедиться в ее работоспособности.

А в завершение урока - еще один незатейливый пример. Ваши проекты в LabVIEW должны иметь привлекательный вид. Рассмотрим как это можно сделать, используя графические файлы.

- Создадим новое приложение;
- в качестве фонового рисунка, будем использовать файл формата ***bmp*** с изображением автомобиля. Для того чтобы поместить картинку на интерфейсную панель или в окно редактирования диаграмм, необходимо в меню ***Edit*** выбрать пункт ***Import Picture from File***;



- выбираем желаемый bmp-файл (в нашем случае ***redchevy.bmp***);
- далее, в том же меню ***(Edit)***, следует выбрать опцию вставки ***(Paste)***.

Пишем программу, которая "включает" правый или левый "поворот" автомобиля в зависимости от положения переключателя.

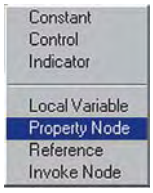
- первым пунктом будет помещение графических объектов на интерфейсную панель, таких как:

(Controls) » Boolean » Round LED (два объекта).
(Controls) » Boolean » Horizontal Toggle Switch.

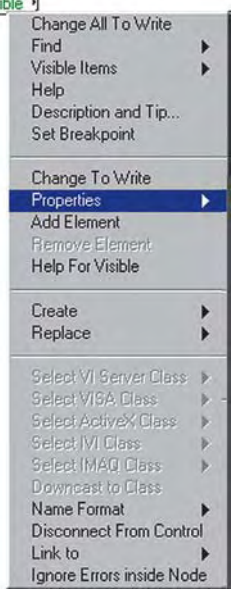
- помещаем круглые лампочки (Round LED) на место фар поворотов автомобиля, а переключатель - в произвольное место;
- изменяем размер, активный и пассивный цвет этих индикаторов:

Перейдем в окно редактирования диаграмм и реализуем заданный алгоритм:

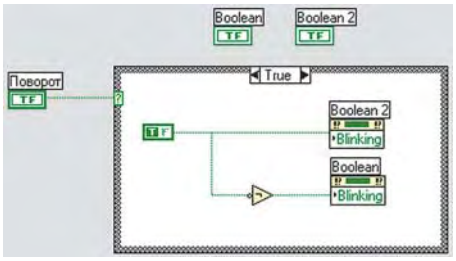
- создаем элемент управления свойством объекта, в нашем случае - лампочки левого поворота. Для этого необходимо нажать правой клавишей мыши на желаемой иконке и в выпадающем меню выбрать: ***Create » Property Node***.



- появившаяся иконка определяет свойство видимости по умолчанию. Нам же нужно изменить свойство видимости ***(Visible)*** на свойство "мигания" ***(Blinking)***;

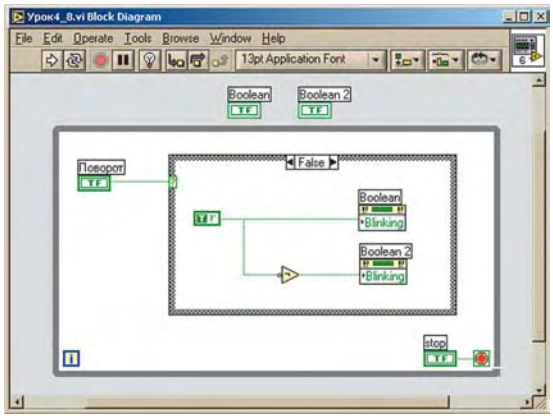


- и последний шаг - изменить свойство чтения на свойство записи - ***Change To Write.***



• аналогичным способом создаем элемент управления свойством "мигания" для второй лампочки-индикатора.

• создаем CASE-структуру, управляющим элементом которой будет переключатель поворотов.



• помещаем CASE-структуру в цикл ***(Functions)*** ≫ ***Structures*** ≫ ***While Loop***, условием выхода из которого будет нажатие на кнопку **"STOP"**.

Обратите внимание, что функция инвертирования используется для того, чтобы во время "мигания" одного поворота, второй был отключен, и наоборот. Запускаем программу на выполнение. Переключая указатель поворотов, убеждаемся, как мигают соответствующие индикаторы. Создав эту программу, Вы, прежде всего, использовали свойства объекта, изменяя их. У объектов есть общие свойства, например, свойство видимости объекта (Visible). Но каждый объект имеет и особые индивидуальные свойства, использование которых позволит Вам создавать программы с уникальным интерфейсом оператора. Пусть это и будет Вашим домашним заданием. Дерзайте!

Возможности LabVIEW практически неограниченны. А это значит описать все - невозможно. Изложенный в 4-х уроках материал охватывает базовый курс "теоретической" подготовки. Для тех, кто успешно освоил его ("ПИКАД" №№ 1-2, 3-4 2003г. и №№ 1, 2, 3 2004г.) следующий шаг - как работать с "железом".

Уроки по LabVIEW

На следующем уроке:

Какой Soft для работы в LabVIEW должен поставлять уважающий себя производитель аппаратных средств.
Ввода/вывод сигналов с популярных плат АЦП/ЦАП/ЦВВ для PC.
Как из DLL сделать драйвер для LabVIEW.
Работа с COM-портом. Модули семейств tetraCON, i-7000/i-8000.
Локальные переменные.