Заметки по практическому программированию на VEE

Александр СОБОЛЬ
sobol@ciam.ru

Цель предлагаемых заметок — познакомить читателя с возможностями визуальной среды программирования Agilent VEE (Visual Engineering Environment) на примерах из собственной практики. Примеры иллюстрируют некоторые проблемы, вызванные применением определенных программных решений, а также демонстрируют неожиданные эффекты.

Все упоминаемые в тексте программы разработаны в лицензионной версии VEE 5.01. Ее стоимость на момент покупки составляла около $1500. Основным результатом использования пакета VEE считается отлаженная программа; впрочем, это справедливо и для любого другого языка программирования, будь то C, BASIC, Fortran, Pascal и т.д. Все программы работоспособны и используются при проведении испытаний на стендах.

Термин «панельный драйвер» употреблен по отношению к виртуальной графической приборной панели, предоставляющей возможность интерактивного управления прибором. Такие драйверы были разработаны фирмой Hewlett-Packard (HP) специально для пакета VEE.

Немного истории

Знакомство с визуальной средой программирования VEE нужно начать с версии 3.0, тогда этот пакет поставлялся фирмой Hewlett-Packard. Поскольку HP всегда выпускала разнообразные измерительные приборы (осциллографы, генераторы, вольтметры и т. д.), то и программа VEE 3.0 была прежде всего сориентирована на работу с ними. В ее состав входили панельные драйверы для разнообразных приборов. Работать с ними одно удовольствие: нажимаешь кнопку на компьютере, а прибор, если он подключен к ПК, сразу на это действие откликается. Когда нам понадобилась программа для считывания информации с запоминающего осциллографа HP, то она была разработана за 15 минут, а затем использовалась в течение полутора лет без переделок. В составе VEE 3.0 был также софт для разработки собственных панельных драйверов. В следующих версиях эта часть пакета уже отсутствовала, и пользователи атаковали фирму вопросами — как же так, такое удобство ликвидировали! Но HP восстанавливать ничего не стала. А сами панельные драйверы существуют до сих пор, присутствуют они и в версии VEE 9.0 от 2008 г. (уже от Agilent Technologies).

Еще одна приятная особенность панельных драйверов: с ними программу можно написать и без приборов. Дело в том, что программа, работавшая на компьютере с реально подключенным прибором, может быть также запущена на компьютере, к которому этот прибор не подключен. В этом случае VEE, не теряя реальных данных о подключении, переводит прибор в режим NOT LIVE. А саму программу можно исправлять, изменять и т. д.: сообщений об ошибках подключения не будет. Когда понадобился инструмент для работы с VXI измерительной системой (VXI — **V***MEbus e***X***tention for* **I***nstrumentation*; VMEbus — **V***ersa* **M***odular* **E***urocard* **bus**), на его разработку ушло 1,5 месяца. В силу обстоятельств пришлось практически всю программу писать и отлаживать без реальной системы. Программа работает до сих пор — уже больше восьми лет, что прекрасно иллюстрирует надежность и устойчивость как пакета Agilent VEE, так и программы, разработанной на его основе.

Подключение измерительных приборов к компьютеру

Как уже упоминалось выше, фирма HP, а затем и Agilent, выпускали и выпускают до сих пор широкую линейку измерительных приборов и систем. До недавнего времени основным средством их связи с компьютером был интерфейс GPIB (**G***eneral* **P***urpose* **I***nterface* **B***us*) — универсальная шина, соответствующая стандарту IEEE 488. Этот же интерфейс известен и под именем HP-IB (**H***ewlett-***P***ackard* **I***nterface* **B***us*), а в России ему соответствует стандарт КОП (**К***анал* **О***бщего* **П***ользования*). Для многих измерительных приборов HP, Agilent и других производителей были разработаны панельные драйверы, работа с которыми осуществлялась, к сожалению, только через GPIB. Несмотря на то, что появились новые интерфейсы — USB, LXI, VXI, которые поддерживаются последними версиями Agilent VEE, GPIB по-прежнему актуален. Сегодня проблемы с интерфейсными GPIB-платами, вставляемыми в компьютер, получили современное решение. Фирмы Agilent Technologies, National Instruments (NI) и др. выпустили контроллеры GPIB-USB, представляющие собой кабель с двумя разъемами. Они позволяют подключать приборы с интерфейсом GPIB к любому современному компьютеру. В мае 2008 г. проверка такого подключения была организована представителем фирмы National Instruments. Результат оказался поразительным: прибор HP выпуска 1996 г. заработал на компьютере под управлением Windows XP без каких бы то ни было дополнительных действий. Просто подключили, и … все работает! Те же фирмы выпустили и интерфейсные GPIB-платы в формате PCI Express. Такая плата и кабель стоят одинаково (по данным с сайта NI на ноябрь 2008 г.), но USB-кабель, несомненно, удобнее, так как разъемы для подключения USB-устройств теперь уже есть практически в любом компьютере.

Теперь несколько слов о самом современном интерфейсе — LXI (**L***AN e***X***tensions for* **I***nstrumentation*). Прежде всего, раз он построен на основе сетевых подключений, то для работы с конкретным прибором требуется наличие сетевого («интернетовского») разъема. Сейчас практически все компьютеры имеют сетевую поддержку. Никаких допол-

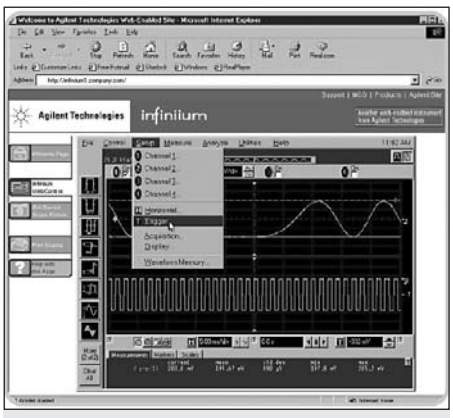

Рис. 1. Виртуальная веб-панель осциллографа Infiniium, позволяющая работать с прибором через LXI-интерфейс

нительных плат, уникальных разъемов и кабелей не требуется, и это весьма важное преимущество LXI. Далее, прибор можно подключить и отключить в любой момент. Перезагрузка компьютера после или до таких действий не требуется. Поскольку работа с подключенным прибором осуществляется по сети, то при правильной ее организации [2, 3] можно работать с прибором, используя удаленный доступ. В соответствии со спецификацией LXI-интерфейса [1] в сети должна присутствовать веб-страница с размещенной на ней виртуальной веб-панелью прибора. Работа с ней напоминает управление через панельный драйвер, о котором уже говорилось. На рис. 1 в качестве примера приведена виртуальная веб-панель осциллографа Infiniium от Agilent Technologies.

Вообще подключение и конфигурирование приборов, с которыми предполагается работать в VEE, — тема отдельной статьи. Упомянем лишь, что в последних версиях VEE этот процесс автоматизирован примерно на 80%.

Основные правила в VEE

Любая программа в VEE представляет собой блок-схему из набора библиотечных элементов. Каждый такой элемент, в том числе и панельный драйвер, — это прямоугольник с контактными точками на верхней, нижней, правой и левой сторонах. Назначение этих точек представлено на рис. 2.

Программа составляется из необходимых блоков, связанных линиями. Обычно соединяются выход со входом — как по информационным контактам, так и по управляющим. Пример простой программы показан на рис. 3. Для ее создания даже новичку потребуется буквально несколько минут, и дальше можно будет экспериментировать.

На рис. 3 видно, что линии, соединяющие блоки, разноцветные: их цвет напрямую связан с типом данных (переменных). Темно-голубой — это Integer (целое), Real (действительное), Complex (комплексное) или PComplex. Оранжевый — это String (текст). Темно-серый — нулевое значение, чаще всего на управляющем выходе. Черный — заранее неизвестный тип переменной (очень часто на выходах разных блоков). И, наконец, ярко-красный — ошибка. Требуется обязательный анализ, дабы понять, что именно вызывает ошибку, поскольку программа с таким блоком не выполняется. Если на выходе блока формируется массив, то линия широкая. Общая идеология VEE: чем меньше черных линий, тем быстрее работает программа. В программе на рис. 3 представлены почти все варианты линий. Часть блоков соединена управляющими линиями: от этого зависит последовательность выполнения процедур.

Рассматриваемая нами программа будет выполняться в следующем порядке:

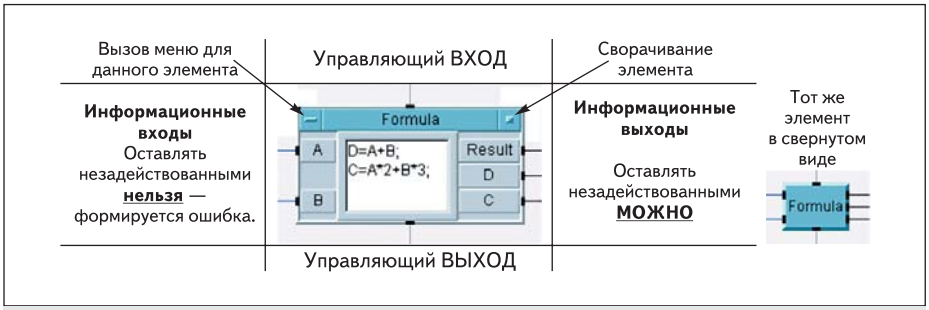

Рис. 2. Назначение контактных точек типового блока "Formula"

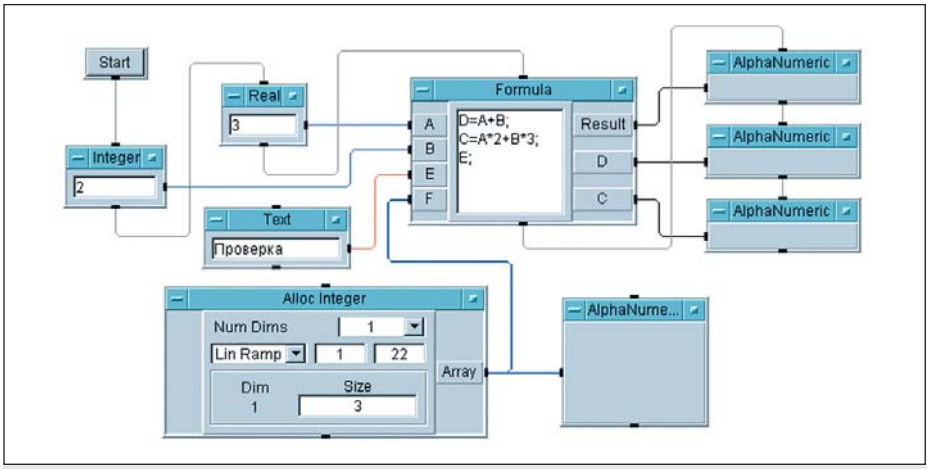

Рис. 3. Простая программа, содержащая разные типы переменных

- 1 — командная кнопка **Start**. Все блоки, не соединенные управляющими линиями.
- 2 — формирование массива **Alloc Integer**.
- 3 — окно отображения результата **AlphaNumeric** с числами 1, 12, 22.
- 4 — формирование текстовой переменной **Text**. С этого места программа выполняется в строгом соответствии с порядком, заданным управляющими линиями.
- 5 — формирование целого числа **Integer** с числом 2 и дальше по заданному порядку.

Если «беспорядочных» блоков в программе много, то отследить ход выполнения процедур и, соответственно, правильность, представляется весьма сложной задачей. Поэтому везде, где это возможно, следует использовать управляющие линии.

Теперь о блоке **Formula**. По умолчанию он имеет один входной контакт A и один выходной — **Result**. В окне с вычисляемым выражением написано **2*A+3** без каких либо знаков в конце выражения. Все остальные входные и выходные контакты добавлены по желанию программиста. Названия всех контактов, как входных, так и выходных (кроме **Result**), можно изменить по своему усмотрению. Все выражения в формульном окне должны быть написаны в обозначениях входных и выходных контактов.

Далее о результатах. На выходах **D** и **C** результат вычислений будет в формате Real: преобразование форматов в VEE производится автоматически. Если нужен вполне конкретный иной формат, преобразование придется добавлять с помощью дополнительного блока. На выход **Result** всегда попадает результат последней строки, написанной в формульном окне.

И, наконец, последнее: формульные строчки, если их несколько, обязательно заканчиваются знаком «;». Это элемент языка программирования C. Если строку нужно закомментировать, то есть не выполнять, опять же используется синтаксис языка C: в начало строки ставится «//». Хотя в VEE, в основном, строки кода писать не приходится, иногда все же необходимо и это. Так что знакомство с основными положениями языка C вовсе не лишнее. Все описанное относится практически ко всем блокам, используемым в VEE.

В [5] упоминается runtime-версия VEE, распространяемая бесплатно, в том числе и с демо-версиями пакета. Если программу, представленную на рис. 3, сохранить в варианте для runtime-версии (расширение будет ".vxe" вместо стандартного ".vee") и выполнить, то откроется пустое окно. Для получения разумного результата необходимо несколько изменить программу: добавить два блока "OK" (**Flow ==> Confirm (OK)**). Один — сразу после блока Start, а второй — в самом конце программы после блока AlphaNumeric с числом 12. У первого "OK" название оставить по умолчанию, а у второго название изменить на EXIT. Затем выделить все блоки

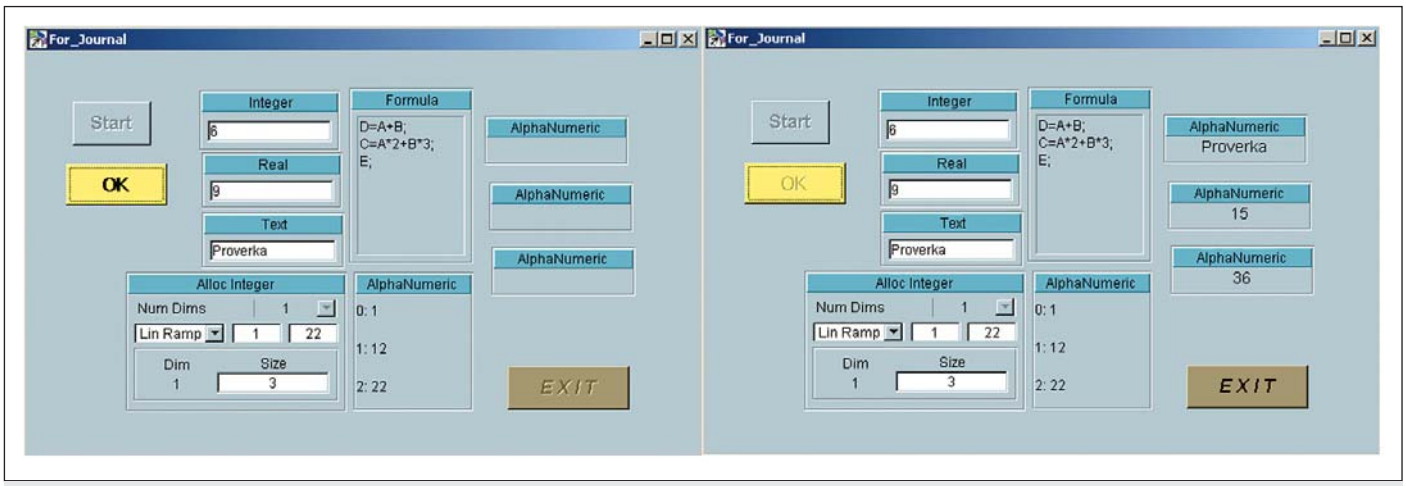

Рис. 4. Рабочее окно runtime-версии программы

программы (**Ctrl+A**) и добавить их на переднюю рабочую панель (**Edit ==> Add To Panel**), сохранить в варианте для runtime-версии и запустить на выполнение. На обоих блоках "OK" программа будет останавливаться. Для продолжения потребуется нажимать на активную кнопку.

На рис. 4 слева показан результат на первой остановке: в этом месте можно изменять данные во всех окошках. Справа — результат на второй остановке: здесь можно посмотреть и оценить конечный результат вычислений. После нажатия на EXIT программа полностью закрывается.

Если описанные выше изменения не вносить, а просто все блоки программы добавить на переднюю панель, то программа запустится, окно откроется и тут же «захлопнется».

Из практики работы с циклами

Из основного меню **Flow ==> Repeat** можно выбрать следующие блоки для создания цикла: **For Count**; **For Range**; **Until Break**; **On Cycle**, а также **Next** и **Break**, являющиеся дополнительными, но не всегда обязательными в цикле. Подробности использования каждого из упомянутых блоков можно найти в справочнике, прилагаемом к любой версии VEE, в том числе и Demo. Здесь в качестве примера, как надо и как не следует составлять циклы, приведена программа из собственной практики с использованием блока **Until Break** (он чаще всего применяется для зацикливания программ).

Назначение программы понятно из представленного на рис. 5 основного рабочего окна. Для реализации всех функций были выстроены цепочки из блоков **Until Break** и управляющей кнопки (блок "**OK**"); далее — соответствующие вычисления по варианту, изображенному на нижней левой диаграмме. При запуске одного из циклов и последующем нажатии на другую рабочую кнопку программа отрабатывает сразу две цепочки — несколько блоков из первой запущенной, затем несколько блоков из второй запущенной, и т. д. Все было бы хорошо, но по «железу» (это все-таки испытательный стенд) они стали пересекаться и явно мешать друг другу. По основной идеологии VEE в таком пересечении ничего незаконного нет, но, прежде чем реализовывать такую программу, нужно очень тщательно проработать алгоритм. В приведенном примере пришлось переделать программу таким образом, чтобы при запущенной какой-либо цепочке никакая другая не запускалась, даже если нажата соответствующая кнопка. Этот вариант отражен на рис. 5 на нижней правой диаграмме. Рабочее окно при этом осталось без изменений.

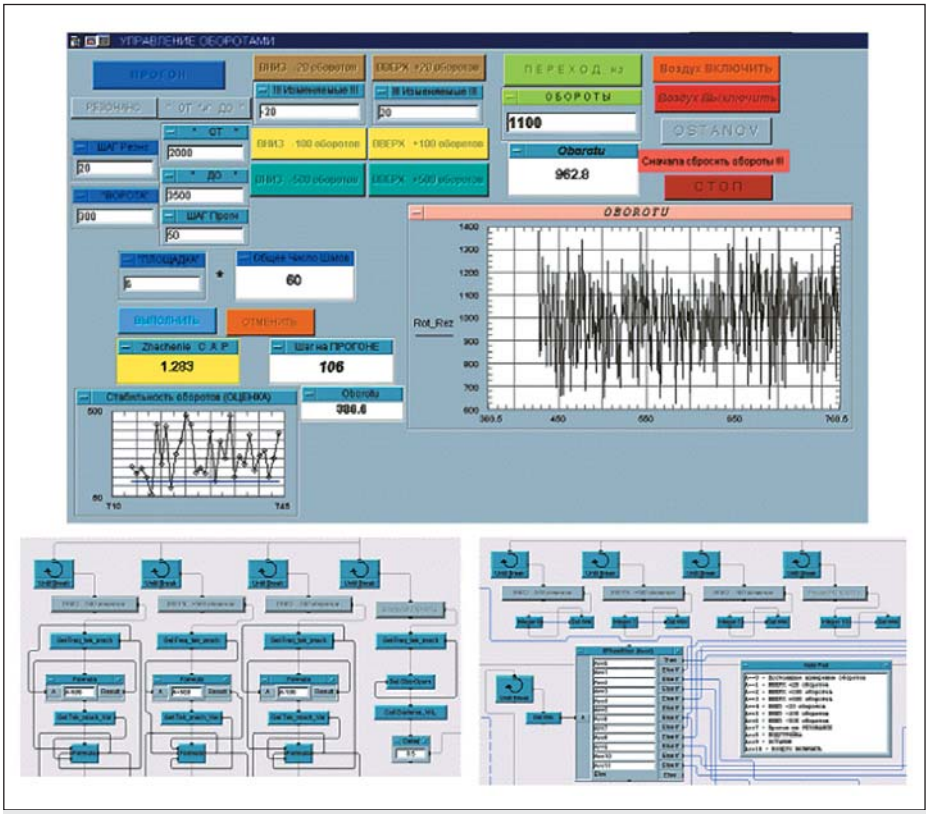

Рис. 5. Программа управления разгонным стендом для исследования характеристик компрессорных лопаток

Формирование переменных с помощью цикла

Описанная проблема была решена введением соответствующей переменной и использованием ее в качестве переключателя выполняемых цепочек. Вообще, в VEE переменные можно вводить практически в любом месте программы с помощью блока **Set Variable** и затем использовать с помощью блока **Get Variable**. Блоки **Set Variable** и **Get Variable** автоматически меняют свое название после ввода названия переменной. Тип переменной целиком и полностью будет зависеть от данных на входе блока **Set Variable**.

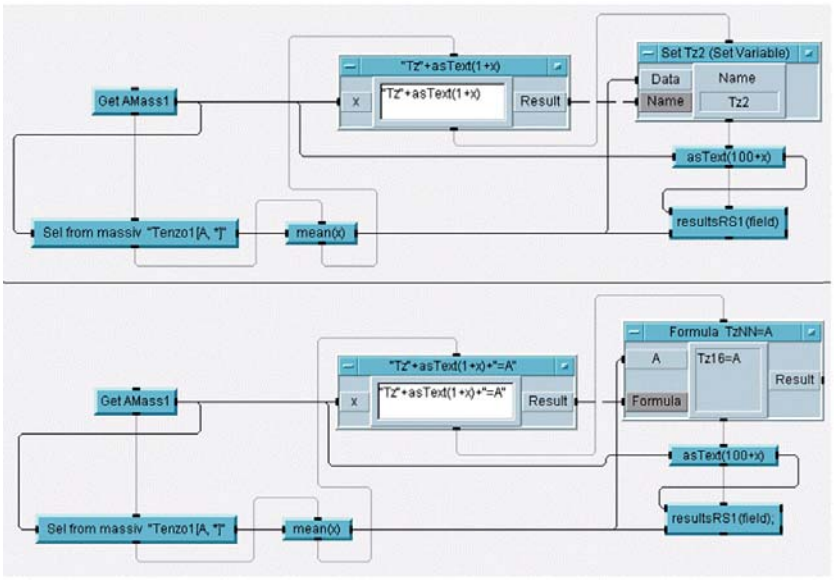

Рис. 6. Вариант динамического формирования переменных TzNN (Tz1, Tz2 и т. д.)

Более того, если в каком-либо другом месте программы произойдет переопределение переменной и данные окажутся другого типа, то и переменная изменит свой тип безо всяких предупреждений. В одной из программ понадобилось создать несколько переменных, отличающихся только номером (Tz1, Tz2, и т. д.). Для этого в цикле модифицировался номер, а в блок **Set Tz2** (Tz2 — последнее запомненное программой название) добавлен входной контакт **Name**, в который из формулы передавалось нужное название. Фрагмент этой программы представлен в верхней части рис. 6.

Программа в такой конфигурации работала нормально, за исключением одного нюанса: вся рабочая панель мигала. Причина этого визуального эффекта кроется в основной идеологии пакета VEE: эта программа компилирующая, и новая переменная для нее — это новый блок, которого не было при запуске. Соответственно, программе нужно перекомпилировать появившиеся связи. Отсюда и мигание рабочей панели. Для устранения данного эффекта требуется, если это возможно, все предполагающиеся для использования переменные объявить как глобальные — блок **Declare**. После такого изменения программы в том месте, где формировались переменные, вместо **Set Variable** нужно вставить формулу с назначением конкретного значения для конкретной переменной. Результат такой переделки показан в нижней части рис. 6. В объект "Formula TzNN=A" добавлен контакт **Formula**, куда и передается нужное выражение. И еще одно замечание: в контакте **Name** и в контакте **Formula** линии соединения прерывистые. Это признак управляющих входов (**Control Input**).

Примеры вывода информации на графики

В пакете VEE допустимы несколько вариантов построения графиков. На рис. 7 представлены некоторые из них.

Правый верхний представляет определенный интерес в связи с тем, что информация для него хранится в двух массивах. При каждом обновлении графика в окно выводятся целиком оба массива. Первый массив (темный график) формируется в начале программы и в процессе работы не изменяется. Второй в начале программы обнуляется и в процессе работы заполняется измеренными значениями. В результате на таком графике можно наблюдать выполнение программы. Нужно заметить, что программирование такого процесса — не самая простая задача.

Правый нижний график представляет собой простое поточечное отображение полученных данных. По горизонтальной оси отображены номера точек по порядку измерения. Это стандартный для VEE вариант. Однако зачастую требуется вывести на горизонтальную ось значения другого параметра. Дело в том, что данные, выводимые на график, имеют абсолютную привязку к измеренным значениям этого другого параметра. Такая подстановка довольно легко делается в Excel. В VEE, как выяснилось в результате переговоров с разработчиками, этого сделать нельзя. Проблема осталась нерешенной.

Комментарий по работе VEE с офисными приложениями

Пакет VEE поддерживает работу со многими программными пакетами. Во всех версиях присутствует большое количество примеров (папка examples) по работе с разными внешними программами. Таких примеров больше 400, так что они являются еще и своеобразными справочниками. Поэтому стандартные варианты рассматривать в данной

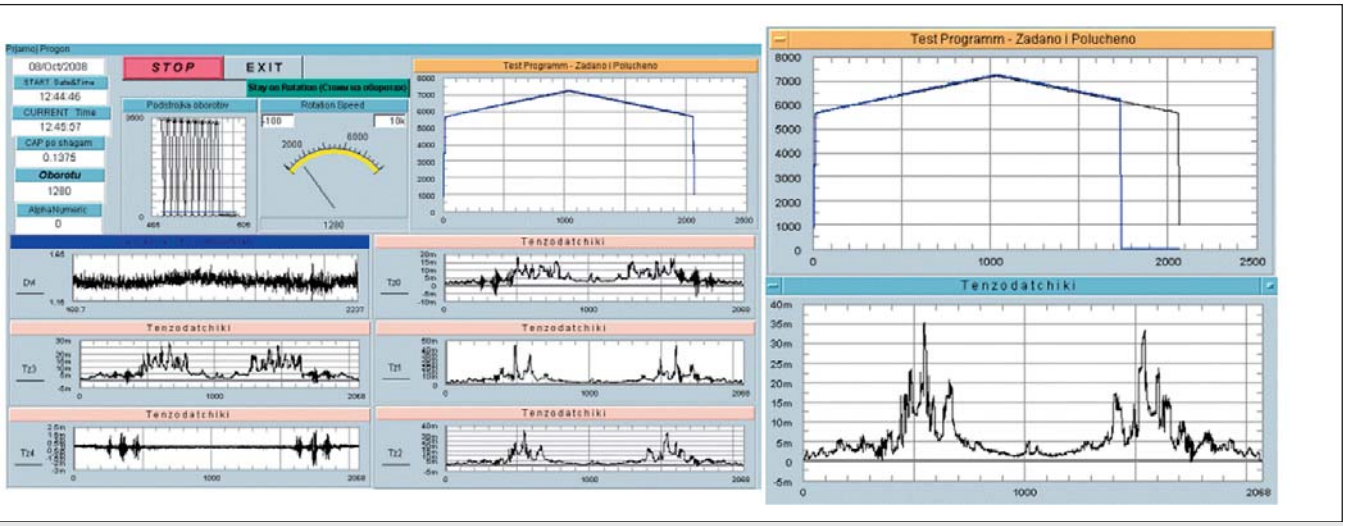

Рис. 7. Варианты графиков в VEE

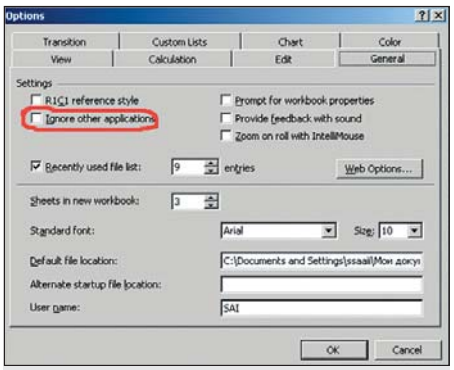

Рис. 8. Настройка Excel

статье нет никакого смысла: все можно найти в примерах. Остановимся только на «сюрпризах», выявленных на практике. Упомянутая программа для работы с VXI измерительной системой функционирует до сих пор. В ней все результаты и настроечные данные для измеряемых параметров сохраняются в таблицах Excel. В сентябре 2008 г. программа вдруг перестала работать. После проведенных проверок выяснилось, что в настройках Excel появилась галочка на строке "Ignore other application" (рис. 8, отмечено красным цветом). В разных версиях Excel этот параметр находится на том же самом месте, но может называться по-другому. По умолчанию, после установки пакета Microsoft Office, галочки нет. Как она появилась, осталось невыясненным (возможно, это результат работы какого-либо макроса). Но перед началом работы с Excel очень желательно проверить этот параметр. Ни в каких инструкциях по работе с VEE это не упомянуто. Найти этот параметр можно в Excel из меню **Tools ==> Options…**, закладка **General**.

Excel удобно использовать для запоминания результатов работы программы. Но вот в случае возникновения форс-мажорных ситуаций (например, зависание компьютера или аварийное отключение энергии) данные полностью теряются. Для надежного сохранения полученных результатов лучше использовать базу данных Microsoft Access. При любых неожиданностях в БД Access сохраняется все, вплоть до последней записанной строки. Кроме того, в такой БД можно вести непрерывное протоколирование работы программы. Бóльшая часть эксплуатируемых и упомянутых программ работает с БД Access.

Заключение

Практика работы с VEE показала, что этот программный продукт весьма удобен для проверки и настройки измерительных приборов фирм Hewlett-Packard, Agilent Technologies и др., в особенности если для них есть панельные драйверы. Простота программирования позволяет достаточно быстро создавать инструменты различной сложности. Программы сравнительно просто переносятся с компьютера на компьютер.

Данная статья затрагивает только некоторые аспекты практической работы с Agilent VEE. Основы программирования изложены в [9, 10]. Эти документы присутствуют во всех версиях VEE. Все остальное приходит по мере реальной работы с Agilent VEE.

Очень полезно познакомиться с серией статей А. Курбатова, опубликованных в журнале «Компоненты и технологии» в 2000–2001 гг. [4–7]. В них, в частности, есть краткое описание VEE. Оно актуально и для самой последней версии VEE — 9.0 от ноября 2008 г.

И последнее. Программа, представленная на рис. 5, разрабатывалась в рамках сотрудничества с индийскими специалистами. Когда их представитель увидел, что работа выполняется с помощью Agilent VEE, то был немало смущен. Выяснилось, что они сами пробовали сделать аналогичную программу на VEE, но их постигла неудача, и они отказались от этого пакета.

Литература

1. Дренкоу Г. LXI — новое поколение измерительных систем. // Электроника НТБ — научно-технический журнал. Контроль и измерения. 2006. № 6.
2. Дренкоу Г. Как создать измерительную систему стандарта LXI // Компоненты и технологии. 2007. № 2.
3. Дренкоу Г. Некоторые факты о скорости, пропускной способности и задержках в приборах стандарта LXI // Компоненты и технологии. 2007. № 1.
4. Курбатов А. Программное обеспечение для сбора и обработки данных при измерениях и испытаниях. Часть 1 // Компоненты и технологии. 2000. № 6.
5. Курбатов А. Программное обеспечение для сбора и обработки данных при измерениях и испытаниях Часть 2 // Компоненты и технологии. 2000. № 7.
6. Курбатов А. Программное обеспечение для сбора и обработки данных при измерениях и испытаниях Часть 3 // Компоненты и технологии. 2000. № 8.
7. Курбатов А. Программное обеспечение для сбора и обработки данных при измерениях и испытаниях Часть 4 // Компоненты и технологии. 2001. № 1.
8. Стетлер С. Расширенное тестирование автомобильной электроники на основе архитектуры LXI // Электронные компоненты. 2008. № 10.
9. VEE_Pro_Advanced_Techniques.pdf / Agilent Technologies.
10. VEE_Pro_Users_Guide.pdf / Agilent Technologies.