WinVCL User's Guide

MANUAL

VEHICLE CONTROL LANGUAGE

WinVCL User's Guide

WinVCL User's Guide, p/n 37312
Rev. A: September 2005

CURTIS INSTRUMENTS, INC.
200 Kisco Avenue
Mt. Kisco, New York 10509 USA
Tel. 914.666.2971
Fax 914.666.2188

www.curtisinstruments.com

CONTENTS

OVERVIEW

This manual describes how to use WinVCL, the graphical user interface to Curtis's **V**ehicle **C**ontrol **L**anguage—which is used to build VCL programs and programmable parameter sets for VCL-enabled systems. WinVCL:

- provides a convenient graphical interface to the VCL compiler
- allows you to quickly generate and modify parameter sets for your Curtis controller
- uses a "project" concept that allows you to organize your code efficiently
- provides convenient access to help files (such as SysInfo).

After reading this manual, you will be able to install the WinVCL interface and use it to build and distribute programs.

You will find this manual useful if you are directly responsible for writing programs in VCL, or if you are simply evaluating the suitability of VCL for your project, or if you are responsible for supervising a project that uses VCL. You should have some programming experience. To use VCL effectively, you should also have some appreciation of the problems involved in making a program respond in real-time.

Organization of the Manual

This manual begins with instructions for installing WinVCL on your computer and connecting a controller to the computer (Section 1), and a brief description of the WinVCL workspace (Section 2). A more thorough description is presented next, walking through WinVCL's displays button-by-button: Main Display (Section 3), Monitor Display (Section 4), and Management Display (Section 5), concluding with an explanation of how to determine program timing (Section 6).

1 INSTALLATION

WinVCL is packaged in one self-extracting file (for example, WinVCL_6_2_1.exe). In addition, you will need one or more controller-specific operating system files (for example, os1236_040716_1_0.os). This section describes how to install WinVCL on your PC and how to connect a controller to the PC.

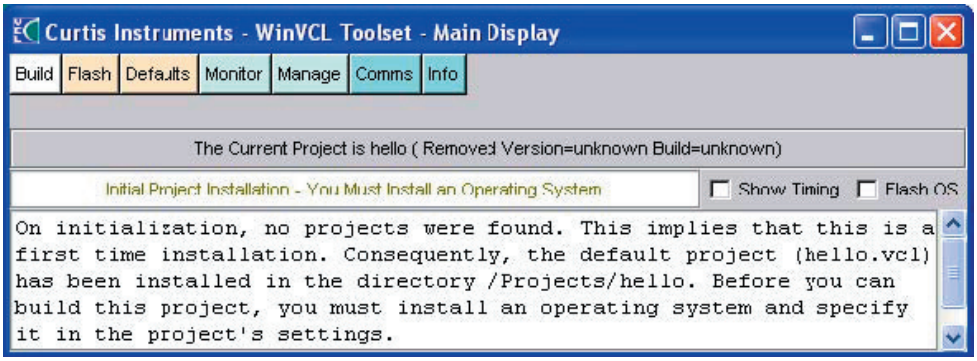
1.1 SYSTEM REQUIREMENTS

WinVCL has been tested on these Windows systems: Win95, Win98, WinNT, and Win2000. Its hardware requirements are quite modest to run; however, to get the best performance, you should have at least:

- Pentium II processor
- 256 KB of RAM
- 5 MB of hard disk space
- serial port to download programs.

1.2 INSTALLATION / UPGRADE

WinVCL is distributed as a standard Windows setup program. If this is the first time you have installed WinVCL, simply execute WinVCL_#_#_# and follow the instructions on the screen. You will notice that the Hello project is automatically installed.

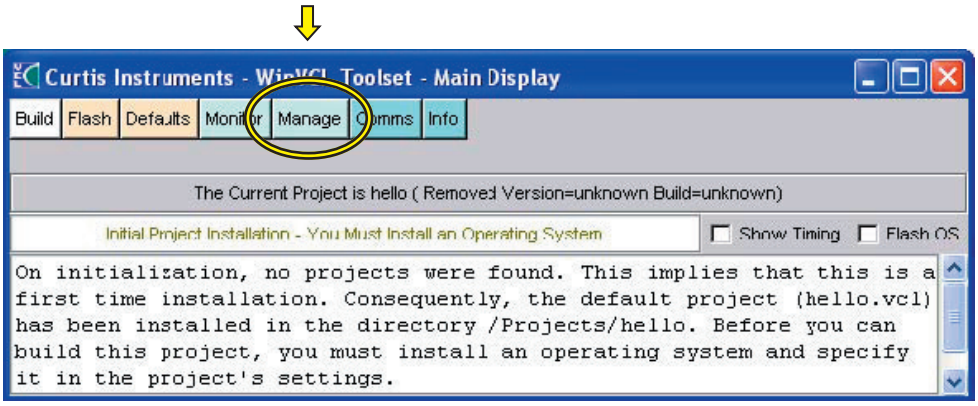


If you have an earlier version of WinVCL, you should uninstall it before installing the new version. Your previous operating system(s), project list(s), and user preferences will be transferred automatically from your previous installation to the new version of WinVCL when you uninstall the old version.

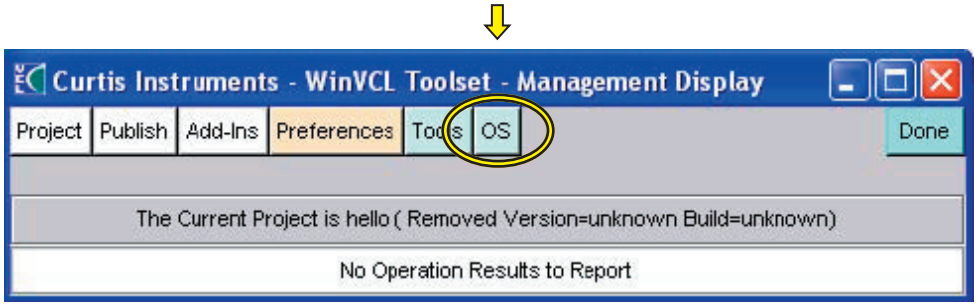
1.3 ADDING AN OPERATING SYSTEM

WinVCL requires that there be at least one installed operating system. WinVCL allows multiple operating systems and even multiple versions of a given operating system to coexist. If this is the first time you have installed WinVCL, you will have to add an operating system.

To add an operating system, click on the Manage button on the main display:



This will bring up the management display. Click on the OS button on the management display:



In the dropdown list that appears, select Install a new OS, and in the file dialog window that appears move to the directory containing the operating system file you want to install and open that OS file (operating systems always have an extension of *.os*).

Click on the Done button on the management display to return to the main display.

1.4 ADDITIONAL STEPS FOR FIRST-TIME INSTALLATIONS

If this is a first-time installation, there are two additional steps. First, click on the management display's Preferences button, and then click on "HTML Viewer" and locate your internet browser; this is necessary for you to be able to view the SysInfo file properly.

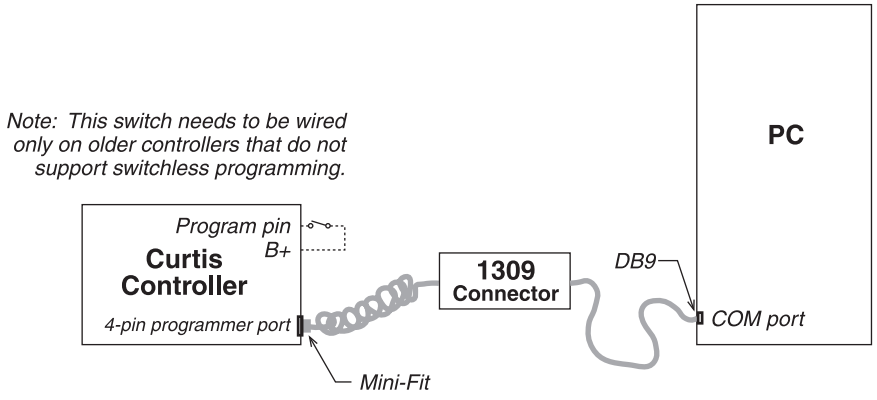
Second, confirm the communications settings. On the main display, click on the Comms button, which will allow you to select the appropriate baud rate, serial port, and switchless strategy. The default settings for these three items will probably work for you. However, if you have an older controller, you may want to change the switchless strategy setting; see Section 3.3, page 8, for more information.

1.5 CONNECTING A CONTROLLER TO YOUR PC

In order to program the flash memory or to monitor variables in the target controller, you must make a physical connection between the controller's serial port and one of your PC's serial COM ports.

The Curtis 1309 adapter is typically used to do this. It provides the correct physical connection, and also provides electrical isolation. The DB9 end of the 1309 connects to one of the PC's serial COM ports. The Mini-Fit end of the 1309 connects to the controller's 4-pin programmer port.

All Curtis controllers now support switchless programming; see Section 3.3, page 8. Older Curtis controllers require a switch to enter programming mode, wired as shown below. If you want, you can can use a switch with the newer controllers even though they do not require one.



2

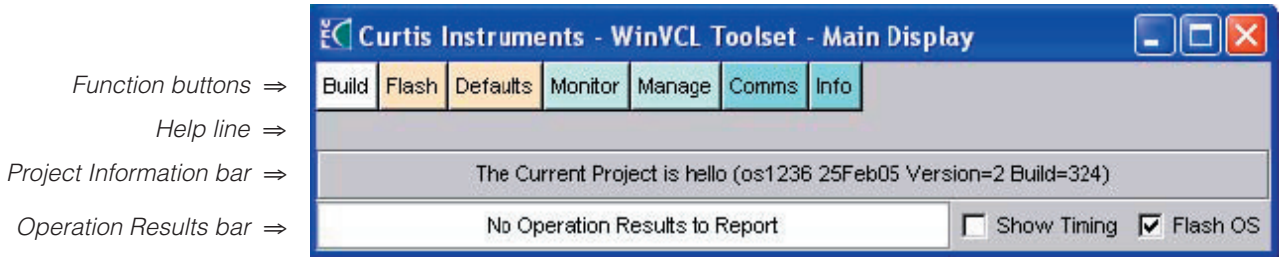
THE WINVCL WORKSPACE

At the heart of the WinVCL workspace are three similar windows:

- Main Display
- Monitor Display
- Management Display

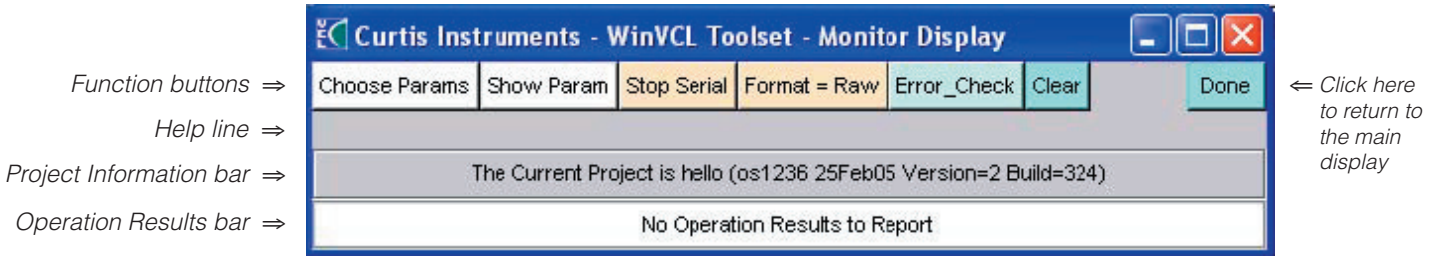
Here is what they look like:

MAIN DISPLAY



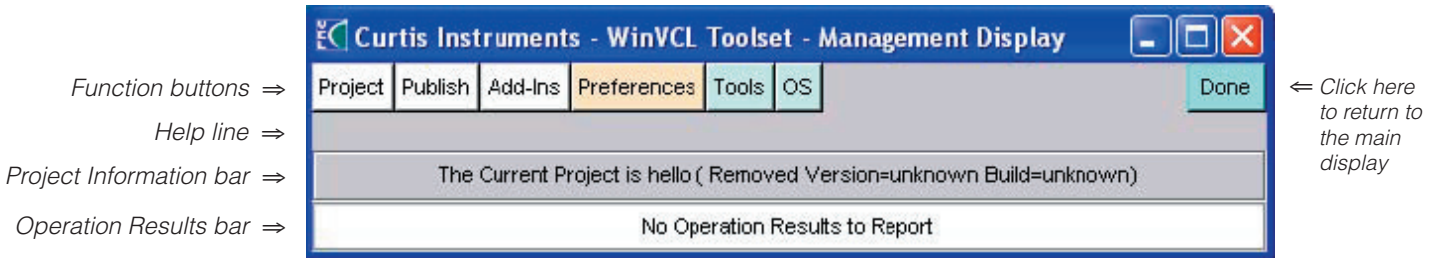
Clicking on the main display's Monitor button brings up the monitor display. This is where you will monitor your system parameters and your controller's serial output.

MONITOR DISPLAY



Clicking on the main display's Management button brings up the management display. This is where you will organize your project files and do other general management tasks.

MANAGEMENT DISPLAY



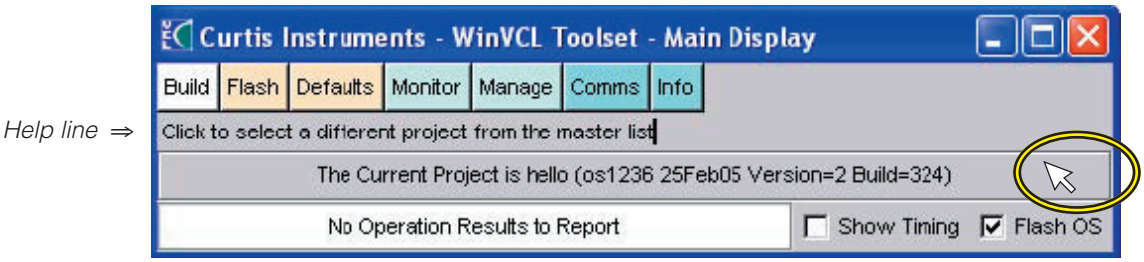
As you can see, although the specific function buttons vary, these three windows share a common structure.

The text in the Operation Results bar shows the progress and results of the current operation, concluding with a message "- Operation Finished". This message is in green if the operation was successful, and in red if there were errors.

To provide additional workspace, WinVCL employs a variety of subwindows, dropdown lists, dialog boxes, and edit windows. These are typically activated by clicking a function button or the Operation Results bar.

The subwindow added at the bottom of the display when you click the Operation Results bar is called the log window. In the case of errors, however, the log window appears automatically, without the Operation Results bar being clicked. Clicking the Operation Results bar while the log window is displayed erases the contents of the log window and closes it.

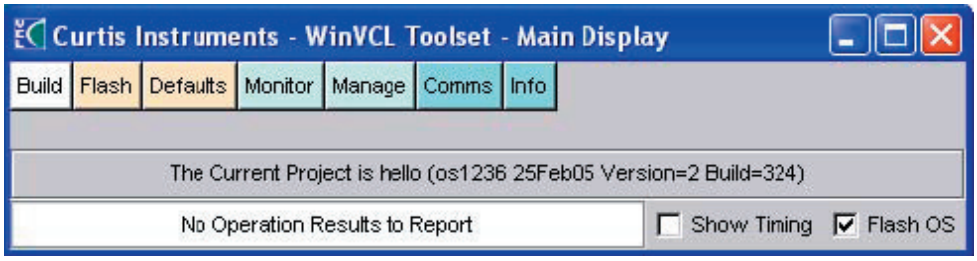
Contextual help is available if you get lost in a display. Hold the cursor on an item (without clicking), and a brief explanation will appear in the Help line. In this example, the cursor is on the Project Information bar, and the Help line is displaying an explanation of this bar:



The best way to become familiar with the workspace is to use it. We urge you to take a few minutes just to click your way around, before continuing to read this manual.

3 MAIN DISPLAY

The main display has seven function buttons and two checkboxes:



Function buttons ⇒

⇑ *Checkboxes* ⇑

Function buttons

Build .....Compiles your source code into an executable image called the Application Package.

Flash .....Downloads the Application Package to the target controller.

Defaults .....Forces all default values to be reinstated.

Monitor .....Brings up the monitor display; see Section 4.

Manage .....Brings up the management display; see Section 5.

Comms .....Allows you to adjust the communications settings.

Information .....Displays revision and copyright information.

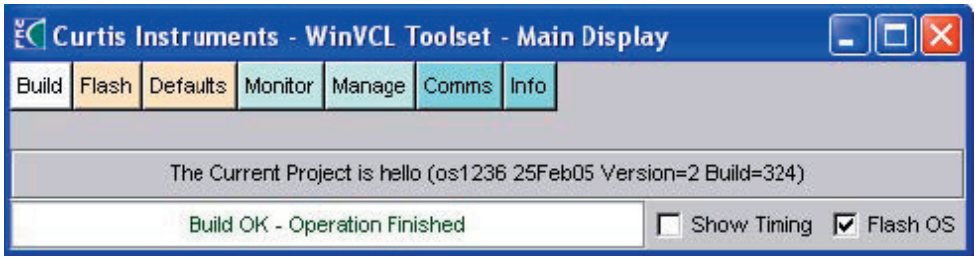
Checkboxes

Show Timing ...Causes the VCL compiler to issue timing information; see Section 6.

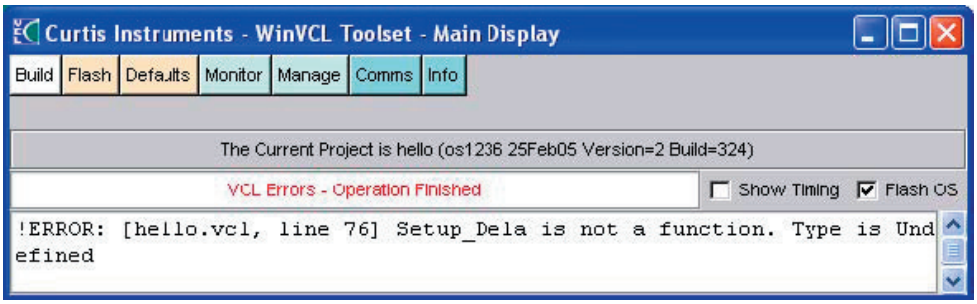
Flash OS .....Includes the current operating system with the Application Package download.

3.1 BUILD BUTTON

The most common series of operations starts with clicking on the main screen's Build button, thus causing an executable image of your program to be generated. If all the parts of the build process are completed successfully, the Operation Results bar announces "Build OK - Operation Finished" in green, as here:



When you click on the Build button, you will see that it causes a number of subprograms to run (see Appendix A for a description of all the subprograms). The Operation Results bar displays the status of each subprogram as it runs. If there are no errors, the final message will be displayed in green (the default color of success) and will end with "Operation Finished". If there are any warnings, they will be announced in yellow (the default warning color). If there are any errors, the build will be terminated, the errors will be announced in red (the default error color), and the bottom of the window will expand to show additional information.

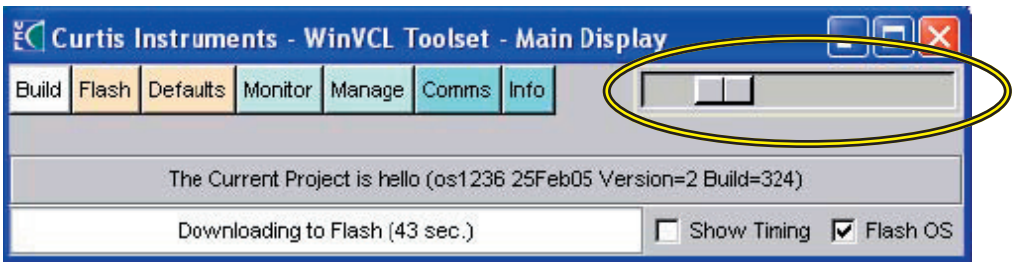


To create this example, we removed the *y* from *Delay* on line 76 of the default project file Hello (the line "Call Delay" became "Call Dela"). You can clear the error text (and make this bottom part of the screen disappear) by clicking the Operation Results bar.

3.2 FLASH AND DEFAULT BUTTONS; OS CHECKBOX

These buttons send information to the controller over the serial communications line. You must set the switchless strategy correctly (see Section 3.3). When you do that, you should also set the baud rate to the highest possible rate so the download takes the least amount of time.

When you click on the Flash button, an estimate of the download time is displayed on the Operation Results bar (in this case, 43 sec). In addition, a progress bar is displayed in the upper right corner of the main display:



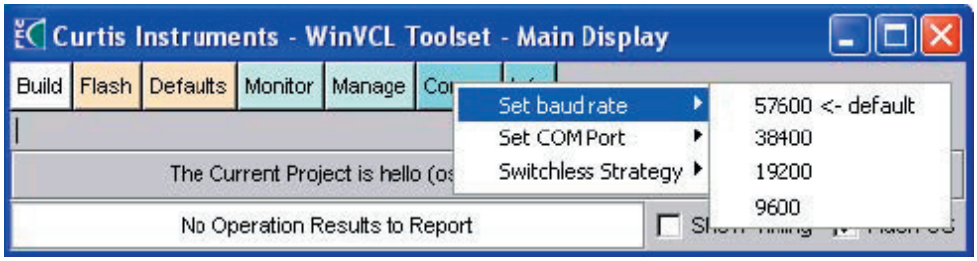
As the download progresses, the progress bar slider moves to the right. When the download finishes, the progress bar disappears. During the download both of the controller's Status LEDs will be lit solid. Before attempting any further operations on the controller (e.g., trying to monitor parameters), make sure that its yellow Status LED has resumed its normal blinking pattern, and that the red Status LED is off.

When Flash OS is checked, the operating system file will be transferred into the target controller's memory along with the application package when the Flash button is clicked. By unchecking this box, you can shorten the flash download time; however, you should take care after installing a new operating system to rebuild your project and to check the Flash OS checkbox at least the first time you download the re-compiled application package. Because this is so important, the Flash OS checkbox is automatically set when you install an operating system.

The Defaults button sends a signal to the controller to refresh its default values. All default values will be restored to their initial state. Defaults are automatically updated when you install a new operating system; however, you will need to click this button when you change your user-defined parameters.

3.3 COMMS BUTTON

The Communications button allows you to set three serial communications parameters: the flash programming baud rate, the COM port, and the switchless programming strategy. Clicking on the Comms button shows a dropdown list with these three parameters:



This example also shows the sub-list for baud rates. Similar sub-lists appear when Set COM Port (a list of available COM ports) or Switchless Strategy (a list of available strategies) is selected. Select the appropriate options for your system.

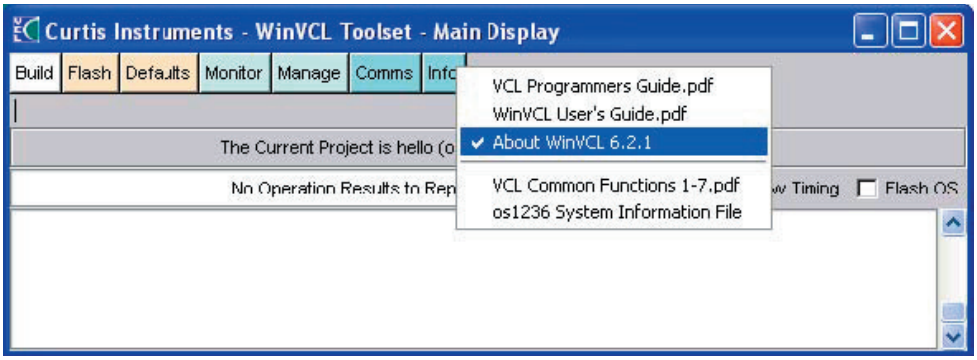
There are three Switchless Strategy options: "Enabled" (the default option), which is the present strategy, now used on all new programmable Curtis controllers; "Legacy_1", which was the original switchless strategy, and may be appropriate for some older controllers; and "Disabled," which is for controllers that use a physical switch to enter programming mode (see page 3).

As you will see in Section 5.1.2, page 19, each individual project can also have its own baud, COM, and switchless settings that override the master settings set via the main display's Comms button. This can be particularly useful, for example, in the rare case where you have two or more controllers in your system, each attached to a different serial port.

Be aware that any changes made using the Comms button also change the project-specific settings; you will need to re-set any project-specific settings that you want to have differ from this master setting.

3.4 INFO BUTTON

The Information button gives you quick access to documentation. Clicking this button produces a dropdown list, from which you can make your selection.

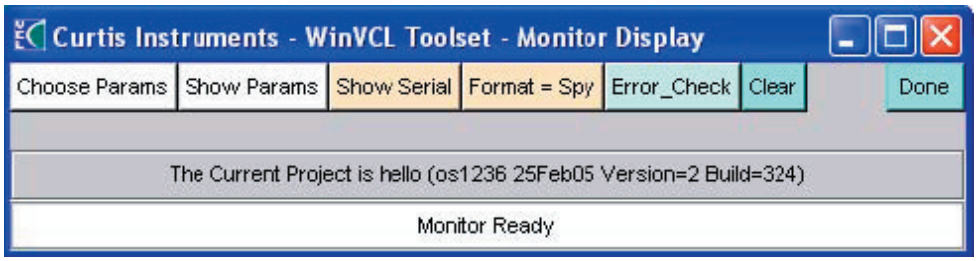


4 MONITOR DISPLAY

Clicking on the Monitor button in the main display brings up the monitor display. To return to the main display, click the Done button.

The monitor display has six function buttons:

Function buttons ⇒



Function buttons

Choose Params ...Allows you to select which parameters to monitor.

Show Params .....Displays values for the selected parameters.

Serial .....Displays the output of the serial port; the button toggles between "Show Serial" and "Stop Serial".

Format.....Determines the format of the Spyglass output; the button toggles between "Format - Raw" and "Format - Spy".

Error Check .....Checks for VCL run-time errors.

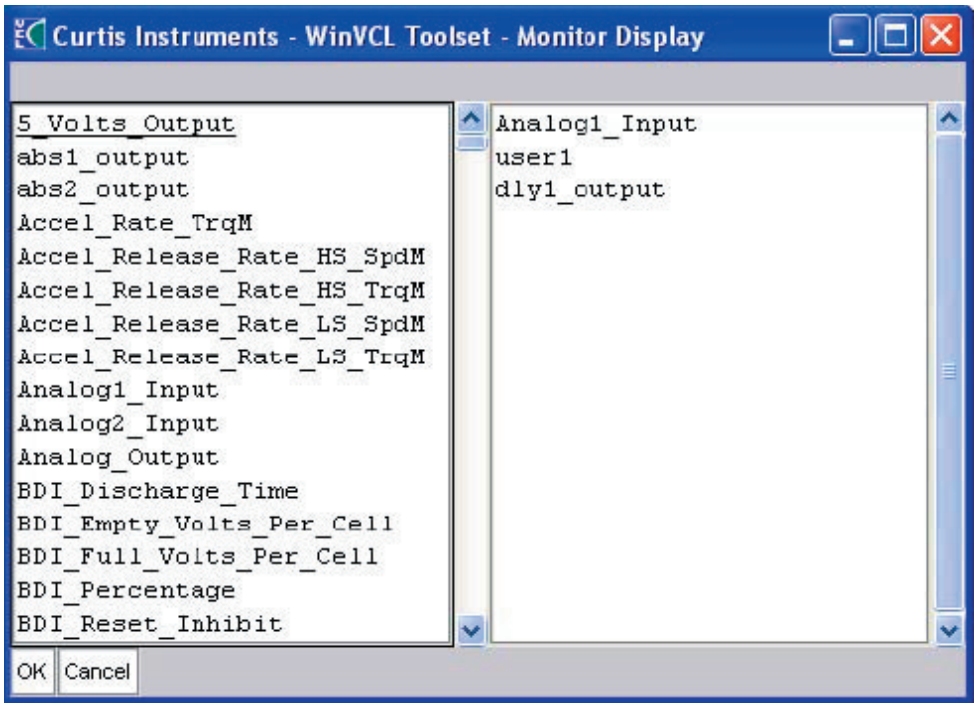
Clear .....Clears the info displayed in the Operation Results bar.

4.1 ACCESSING PARAMETER VALUES

Assuming you have a parameter set installed, the monitor display allows you to read and modify parameter values. If you have not installed a parameter set, the Choose Params and Show Params buttons will not appear; for instructions on how to install a parameter set, see Section 5.1.2: Edit this project.

4.1.1 Choose Params button

Before you can monitor a value, you must select it by clicking on the Choose Params button. This provides you with a secondary window:



The left-hand pane holds an alphabetized list of all the system variables. To find variables in the left-hand pane, use the scroll bar. Typing the first letter of a variable brings you to that portion of the alphabetized list.

The right-hand pane holds a list of variables that are currently displayed. This pane holds up to 16 entries at a time. If you try to add more, you will get a warning message reminding you of the limit.

To add a variable from the left-hand pane (all variables) to the right-hand pane (currently displayed variables), click on it. To remove a variable from the right-hand pane, click on it.

When you have finished selecting the parameters you want to access, click OK to return to the standard monitor display.

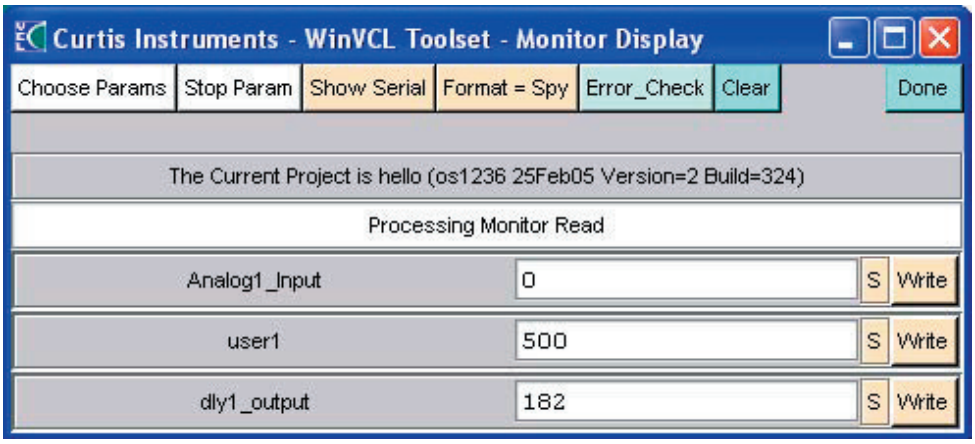
4.1.2 Show Params button

To display values for the parameters you selected, click on the Show Params button. Clicking the button a second time (it will now be labeled Stop Params) will stop the parameter updates.

By default, parameter values are displayed as signed decimals. You can change the format by clicking on the small button to the right of the displayed value. The single letter on this button indicates the current display format; clicking the button changes the format in this sequence:

S Signed
U Unsigned
H Hexadecimal
B Binary.

Here is the Show Params display for the three parameters selected in the Choose Params example:



4.1.3 Modifying parameter values

To modify the value of a parameter, click on its displayed value; this will stop the parameter updates. The background will change slightly, indicating that the value can now be altered. Replace the existing value with the new value. You can then click on the Write button, which will write the new parameter value but not restart the parameter display. Alternatively, you can click the Enter key on your keyboard to both write the new parameter value and restart the parameter display.

You can enter parameters as decimal integers, or as hex values with the prefix "0x" (zero followed by the letter *x* in either upper or lower case).

4.1.4 Using parameters to debug code

The traditional method of debugging a program is to insert print statements in your code and then assess the displayed values. You can certainly do this in VCL by using the Spyglass functions; however, it is generally much more useful to directly monitor parameter values. For example, instead of using a text message to print out "GotHere", you can put a value into a user variable, which you then monitor.

4.2 SEEING THE SERIAL OUTPUT

All VCL enabled systems have a serial I/O connection, which is used for three purposes: (1) the Flash programming utility uses the serial port to download new programs into the controller's flash memory, (2) the Spyglass output goes out the serial port, during normal operation, and (3) any device that uses the Extended Serial (ES) protocol (e.g., the 1311 handheld programmer and this monitor program) uses the serial port.

4.2.1 Show Serial button

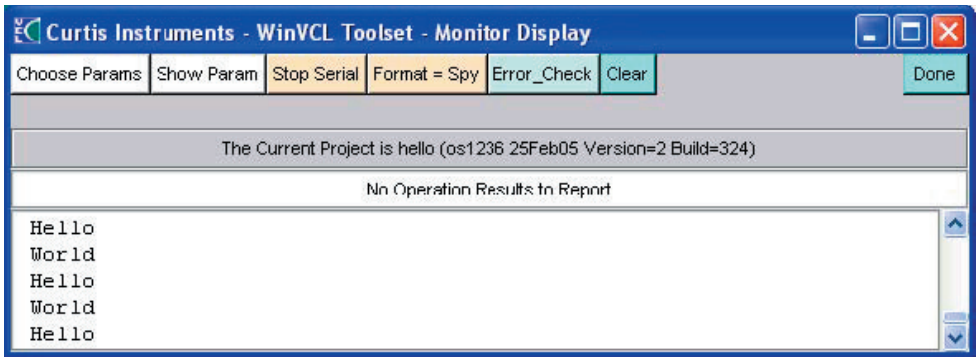
Clicking on the Show Serial button expands the window and displays the output of the serial port. Clicking the button a second time (it will now be labeled "Stop Serial") stops the serial display.

The Show Spyglass Output checkbox determines the display mode: Spyglass mode or raw mode.

4.2.2 Format button

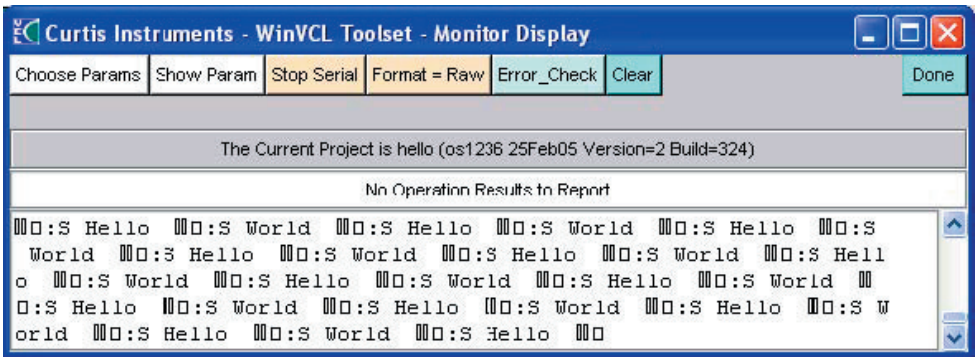
Clicking on the Format button toggles it between "Format - Spy" (the default mode) and "Format - Raw".

Spyglass mode is the system default. In this mode, Spyglass messages are processed to give you a more readable display by removing the leading ":S" and the trailing two bytes of LED display information and the checksum.



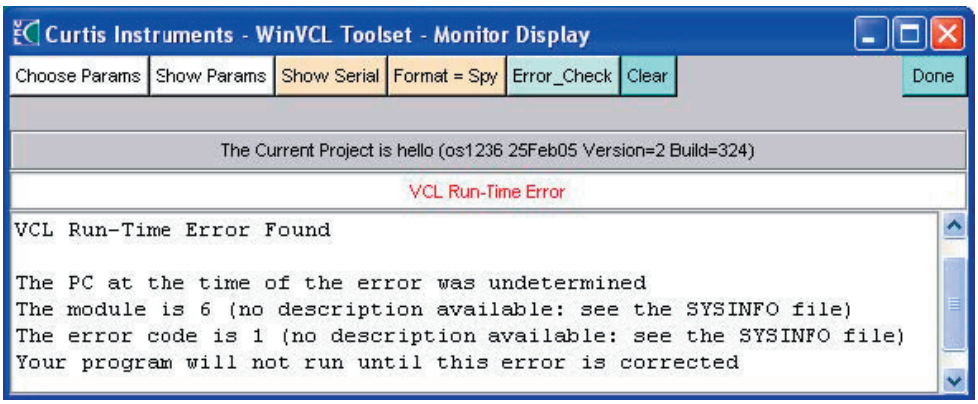
Occasionally, a partial message will be received due to the processing within WinVCL. In this case, you will see the message "Partial message (part-that-was-received)".

Raw mode allows you to see the serial output without any formatting. All output is displayed as if it were text. This means that some values (such as a Spyglass checksum) may display as non-readable characters.



4.3 ERROR CHECKING

Clicking the Error_Check button causes VCL to check for run-time errors. If an error is found, the VCL PC, the Module ID, and the Error Code are displayed, as in this example:



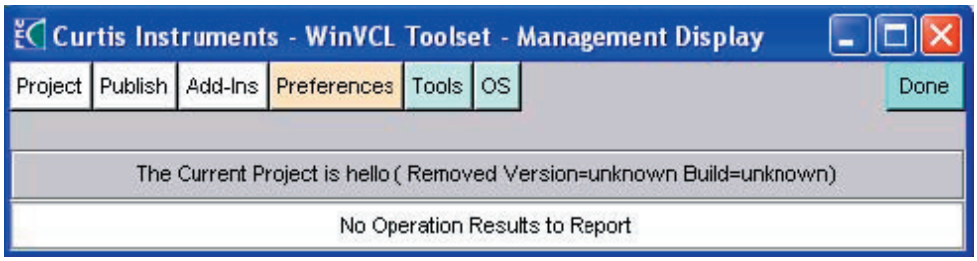
The version of the OS used in generating this example did not have the ability to explicitly report the module and error code descriptions. In this case, you would have to consult the SysInfo file to determine where the error occurred (the error module) and the nature of the error (the error code); the error modules and codes appear at the end of the SysInfo file.

5 MANAGEMENT DISPLAY

Clicking on the Manage button in the main display brings up the management display. To return to the main display, click the Done button.

The management display has six function buttons:

Function buttons ⇒



Function buttons

Project .....Allows you to create, modify, delete, and restore project settings.

Publish .....Allows you to distribute your project to users or to other developers.

Add-Ins .....Allows you to add additional files to a project distribution.

Preferences .....Sets general options and preferences.

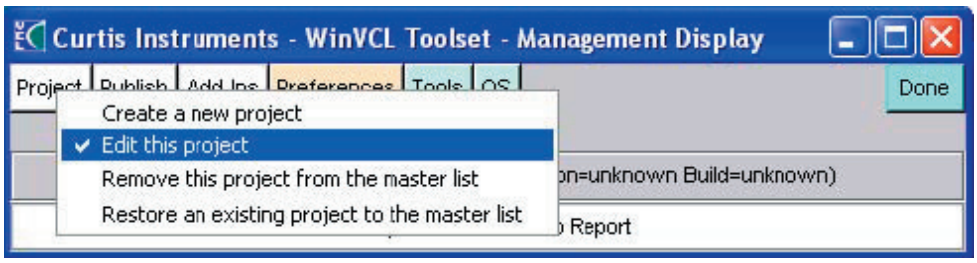
Tools .....Provides access to other software tools.

OS .....Installs, reinstalls, and deletes operating systems.

Note: The Project button gives you access to project maintenance functions; the remaining buttons control universal settings independent of project.

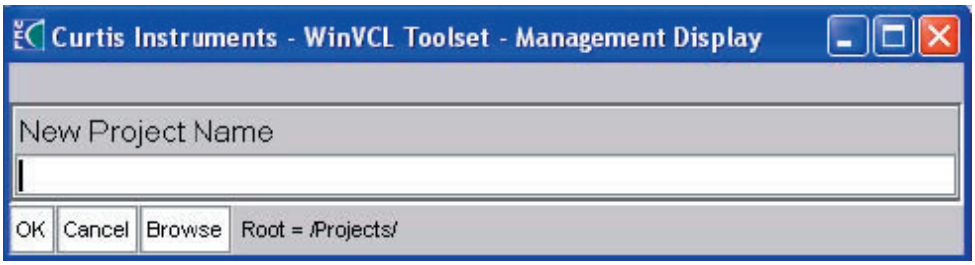
5.1 PROJECT BUTTON

The collection of files, paths, and filenames used to create an application is called a *project*. The Project button is used to manage this collection. Clicking on the Project button causes a dropdown list to appear:



5.1.1 Create a new project

Clicking on "Create a new project" brings up this dialog box:



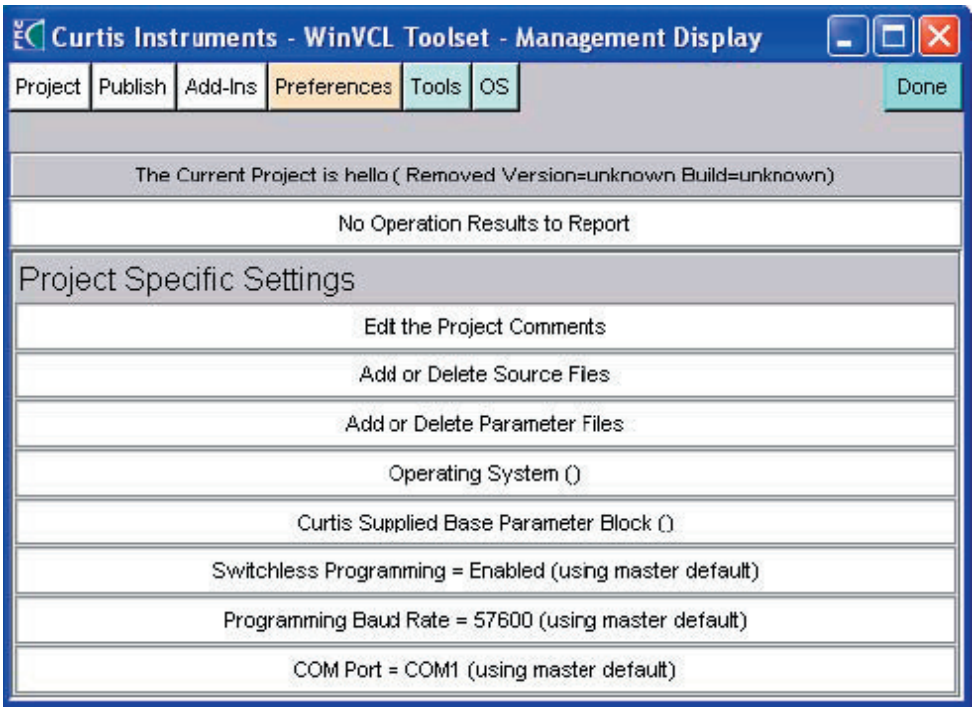
Type in a name for your new project. This name will be used to create a project subdirectory and to create an INI file (project information file).The assumption is that the project's VCL source file will have this same name, although you can select a different VCL source file name if you wish; see Section 5.1.3.

You must click the OK button to create the new subdirectory and project INI file. You can click Cancel to cancel the operation without creating any files.

If you look to the right of the Browse button, you will see "Root = /Projects/". This tells you where the new subdirectory for your project will be located. If you want, you can use the Browse button to select a different project root. In the Browse button's file dialog window, click on the directory underneath which you want to create your new project, or click on Cancel to go back to the New Project Name dialog. (The default root directory is set in Preferences; see Section 5.3.)

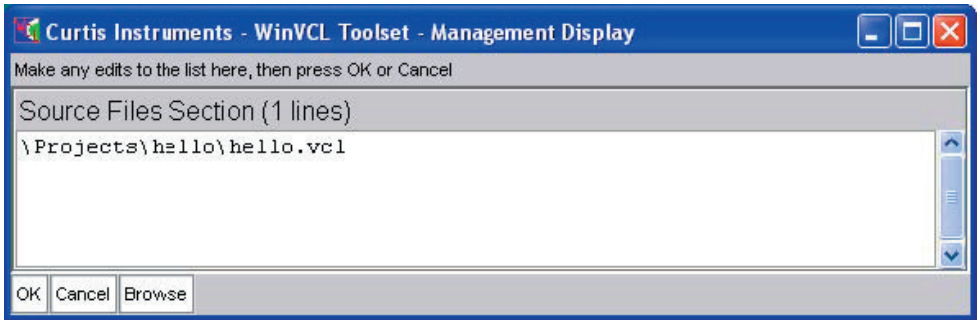
5.1.2 Edit this project

If you click on the "Edit this project" member of the list, a number of project-related clickable bars will appear at the bottom of the window:



Edit the Project Comments brings up an entry window where you can modify the project comments. When a project is created, its name (full name, including path) and creation time are automatically entered. These comments are for your benefit and are not used by the system. You can type anything you want to here. A typical use of this section is to keep a running history of major changes to the project. You must click on OK or Cancel to return to the project editing screen. This ensures that your changes are explicitly saved or discarded, depending on which button you click.

Add or Delete Source Files brings up an entry window containing a list of your project's source files. These are the files that will be processed by the VCL compiler.



VCL allows your program source code to reside in more than one file. If you have more than one file, there are two methods you can use to inform the compiler of the names and locations of the files: you can use an include-statement, or you can list the files in the Source Files section. We have a prejudice for using the include-statement, as it makes the code more self-documenting.

When you create a project, the system assumes there is a VCL source file, with the same name as the project. This source file is automatically entered in the Source Files section for you. Be aware—if your source file has a different name than the project (and we do not recommend this!)—you will need to edit the Source Files section to use the new name. Also, if you have a complex project with multiple VCL source files, you will need to add them to the Source Files section; the system will not automatically do this for you. (The other alternative, of course, is to use include-statements.)

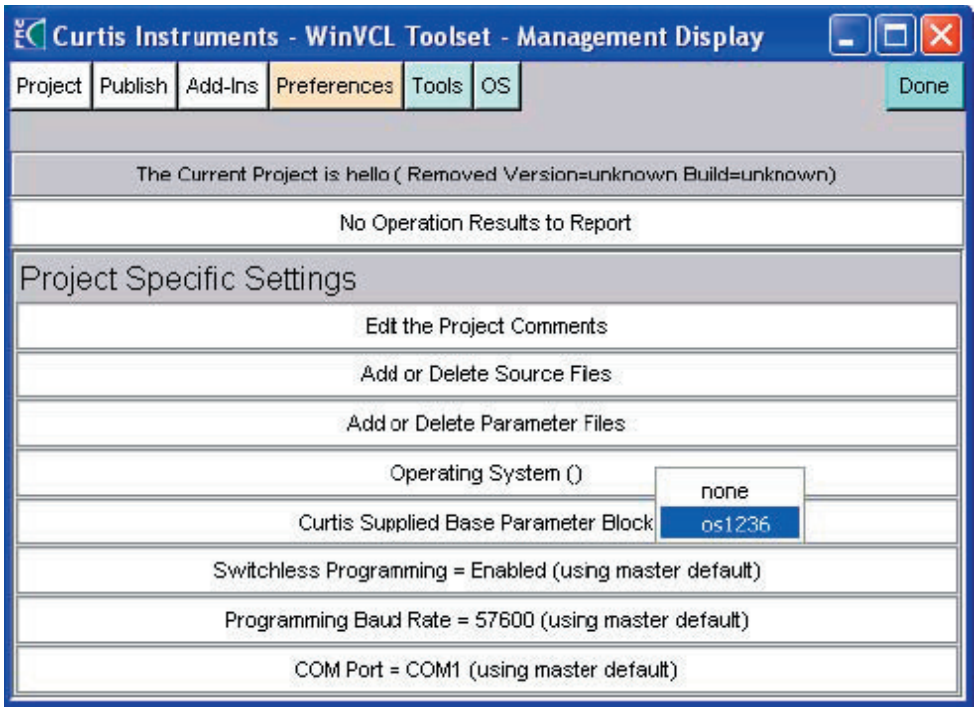
If your source file has include-statements, you do not need to enter these additional files into the Source Files section. The VCL compiler will automatically include them as it runs.

Add or Delete Parameter Files brings up an entry window with a list of the files containing 1311 parameter definitions for your project.

When you create a project, the system automatically makes an entry into the Parameter Files section using the same assumption it does for the Source Files

section, i.e., that the name of the primary file is the same as the project's name. If your source file name is different, you will have to edit this section. Also, if you have put your parameters into a different file, you will have to edit this section. If your project does not use any parameters, you may safely delete all files from this section.

Operating System Before you can compile a system, you must select an operating system. To do this, click on the Operating System bar. This will bring up a list of operating systems. If no operating systems are installed, the only item on the list will be None.



As you move the cursor over each operating system name, a list of specific versions appears to the right. (Yes, you can have multiple versions of an operating system installed.)

Curtis Supplied Base Parameter Block presents a dropdown list of available Curtis-supplied parameter blocks. The list of parameter blocks that you can load depends on the operating system you have installed; this means **you must specify an operating system before you can select a parameter block**.

If you have an application that uses only local parameters (i.e., you do not use a Curtis-supplied parameter block), you should select None from the drop-down list. The system will then automatically delete the parameter entry in the Application Package section (the PRM file with the same name as the application). You will need to edit the Application Package section to reinstate your own parameter block.

If you select one of the installed Curtis-supplied parameter blocks, the system will automatically add a parameter entry to the Application Package section. If you have created any parameters in your project, they will be combined ("stitched") with the Curtis-supplied parameter block to form the final comprehensive parameter block in the Application Package section.

Switchless Programming allows you to select a project-specific switchless strategy. The strategy you select here will override the master setting (set via the main display's Comms button, page 12) when this project is active. The dropdown list of strategies will indicate which strategy is currently in use by this project and which is the default master strategy. Any time you change the master setting, the new setting will apply to individual projects as well. If you want a different project-specific setting, you will have to re-set it via the management display's Project button > Switchless Programming.

Programming Baud Rate is used to select a project-specific programming baud rate. The rate you select here will override the master setting (set via the main display's Comms button) when this project is active. Any time you change the master setting, the new setting will apply to individual projects as well. If you want a different project-specific setting, you will have to re-set it via the management display's Project button > Programming Baud Rate.

COM Port is used to select a project-specific COM port. The port you select here will override the master setting (set via the main display's Comms button) when this project is active. Any time you change the master setting, the new setting will apply to individual projects as well. If you want a different project-specific setting, you will have to re-set it via the management display's Project button > COM Port.

5.1.3 Remove this project from the master list

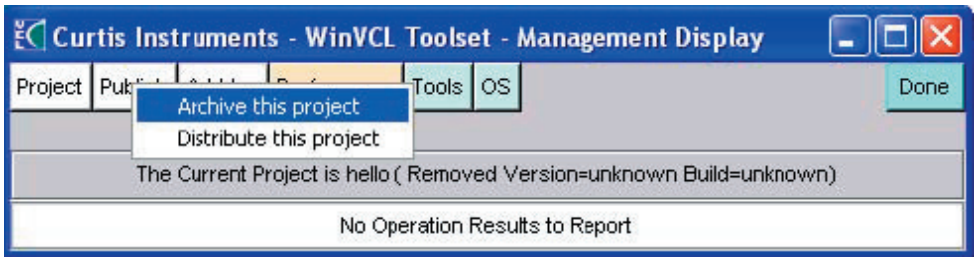
This selection from the Project button's dropdown list allows you to remove a project from the master project list (the master project list is what you see when you click on the Project Indicator bar). Removing a project from the master project list does not destroy the project. You may wish to remove projects that are no longer active to reduce screen clutter.

5.1.4 Restore an existing project to the master list

This selection allows you to restore a previously defined project to the master list. You may wish to do this to restore a project that you have previously removed from the master project list. You may also use this option to install a project created by someone else. **Always check the project settings to make sure that they are still correct** (for example, the operating system version might have changed, or the Curtis-supplied parameter block may no longer be valid).

5.2 PUBLISH BUTTON

The Publish button allows you to choose whether to archive or distribute your project, via a this dropdown list of two elements:

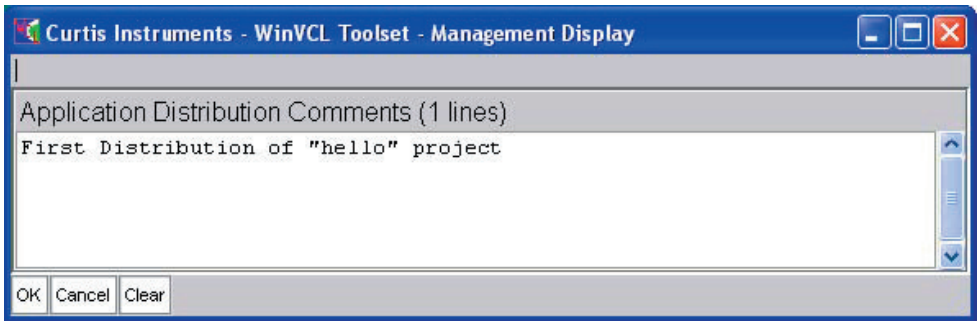


Archiving is generally used to provide a copy of your project to another VCL developer. Distributing is used to send your project to an end-user in a self-executable file.

5.2.1 Archive this project

There are several reasons why you might want to archive your project. You may want to save your work-in-progress, especially before making any large changes to your source code. You may want to share your code with other developers. Or, this may be the format in which you distribute your code.

Clicking on "Archive this project" causes a comment panel to appear:



The Clear button at the bottom of the display allows you to clear any previous comments; comments are saved between sessions with the assumption that most often you will append your comments to an existing list. Once you have typed in your comments, click the OK button to proceed. You can click the Cancel button to stop at this point, with no archive created.

The new archive will be placed in the "archive\app_source" subdirectory of your project's directory. For example, if your project was "C:\Projects\hello", the archive will be placed in C:\Projects\hello\archive\app_source.

For a complete description of the naming conventions used for application distribution files, see Appendix D.

The archive will contain the following files:

BSL.ZIP: This is a ZIP of all Binary Segment Load files (files with a BSL extension). These files are used to download the application over the CAN bus. This file will only be included if all of the subfiles exist; you must have successfully built the project to create these files.

PROJECT_NAME.HEX: This is the hex image of the application package. This hex file is downloaded over the serial port. This file is only included if the project has been successfully built.

PROJECT_NAME.INI: This project information file identifies all the files needed for this project, plus definitions of all project settings.

PROJECT_NAME.TXT: This is a standard read-me file (see Appendix C), which also holds all of your comments.

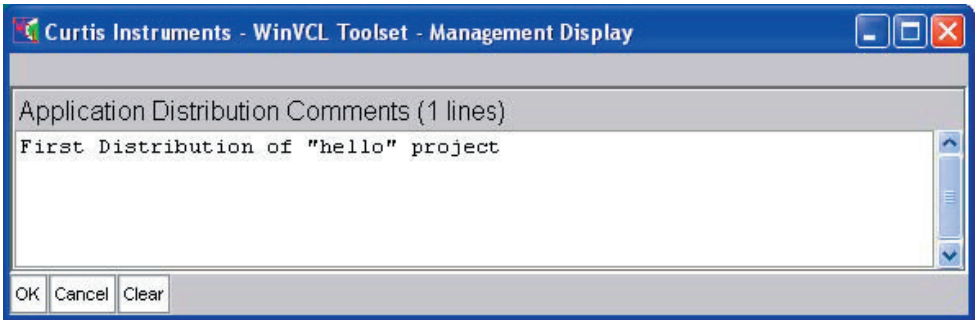
***.VCL:** These are the files that are listed in the source files section of your project (Project → Edit this project → Add or Delete Source Files).

OS_NAME.OS: This is a copy of the OS distribution file used by this project.

To use this archive, unzip it into its own directory. Then use the project "Restore an existing project to the master list" selection (see Section 5.1.4) to add the project to the current project list.

5.2.2 Distribute this project

To distribute your project you must have a copy of WinZip's self-extractor program installed on your system. With WinZip's self-extractor installed, clicking on "Distribute this project" brings up an edit window where you can make notes or comments about this distribution:



The text you type here will be saved in the DISTRIB subdirectory of your project, in a read-me file with the same base name as the project, but with a *.txt* extension (see Appendix C).

When you click the OK button, a single executable file is created in the DISTRIB subdirectory. An archive of all the files needed to recreate this executable file will

be created in a project subdirectory called “archive\app_sources” (see Section 5.2.1: “Archive this project”). The executable file needs no other files in order to run on any Windows system, Win95 and later.

The VCL compiler also generates BSL files that can be used to download your project over the CAN bus. See Curtis’s *Generic CANopen Implementation Guide* for more information on BSL files.

Project distribution directory entries

Before you read further, you should take a look at Appendix D, which describes how file names are automatically generated. You might also take a quick look ahead in Section 5.4.2, page 25, at the behavior of “Archive Root Directory”, which allows you to specify a public directory path and a user name.

When you create a distribution, the *.exe* file is placed in a subdirectory of your project named “Distrib”. All your project files (the *.vcl* files and the *.ini* file) are also saved. In addition, both the *.exe* and the project files are saved in a public directory, if you’ve defined one using the Archive Root Directory button.

Below is an example of how files are created and named. These are the assumptions in this example:

- The project name is “Test”.
- The project root directory is “\Projects\Test”.
- The public archive path is “\Public_Archive” with the user name given as Ralph.
- The OS that Ralph used is Release 2, build 0.
- The starting date for this example was 21 July 2004.

Local Directories

```
\Projects\Test\Distrib\Test_040721_2_0_1_Ralph.exe
\Projects\Test\Distrib\Test_040721_2_0_2_Ralph.exe
\Projects\Test\Distrib\Test_040722_2_0_1_Ralph.exe
\Projects\Test\Archive\App_Sources\Test_040721_2_0_1_Ralph.zip
\Projects\Test\Archive\App_Sources\Test_040721_2_0_2_Ralph.zip
\Projects\Test\Archive\App_Sources\Test_040722_2_0_1_Ralph.zip
```

Public Directories

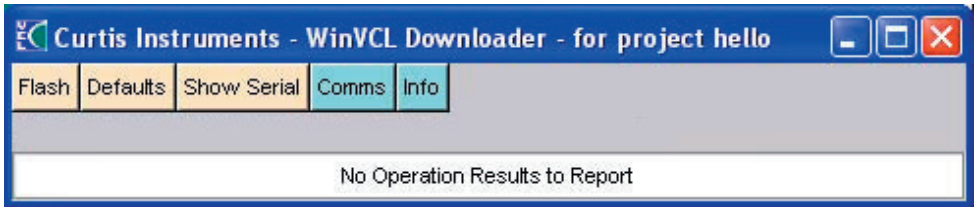
```
\Public_Archive\Test\App_Distribution\Test_040721_2_0_1_Ralph.exe
\Public_Archive\Test\App_Distribution\Test_040721_2_0_2_Ralph.exe
\Public_Archive\Test\App_Distribution\Test_040722_2_0_1_Ralph.exe
\Public_Archive\Test\App_Sources\Test_040721_2_0_1_Ralph.zip
\Public_Archive\Test\App_Sources\Test_040721_2_0_2_Ralph.zip
\Public_Archive\Test\App_Sources\Test_040722_2_0_1_Ralph.zip
```

These file sets imply the following sequence of events:

1. Ralph made an application distribution (sequence 1).
2. Ralph made another application distribution later that day (sequence 2).
3. The next day, Ralph made another application distribution (sequence 1).

Executing the distribution

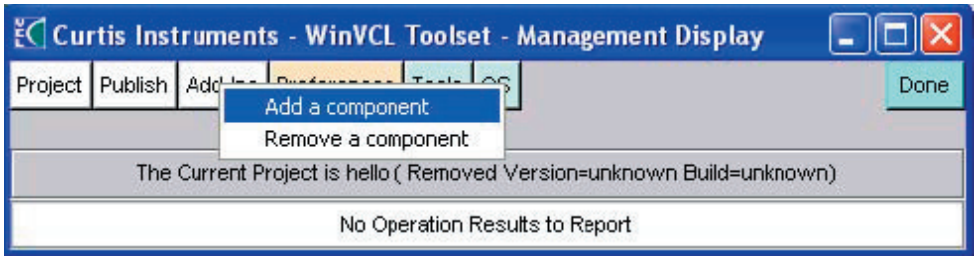
When the end-user runs the executable, this secondary window appears:



The user may need to adjust the baud rate and COM port settings to values appropriate for their system. Once this has been done, they can click on the Flash button to update the controller. If parameter values have changed, they will need to click on the Defaults button as well.

5.3 ADD-INS BUTTON

The Add-Ins button allows you to add additional files to your archive files. This function is particularly useful for adding documentation files to your archives—additional release notes, specifications, etc. Clicking on the Add-Ins button provides a dropdown list of two elements:

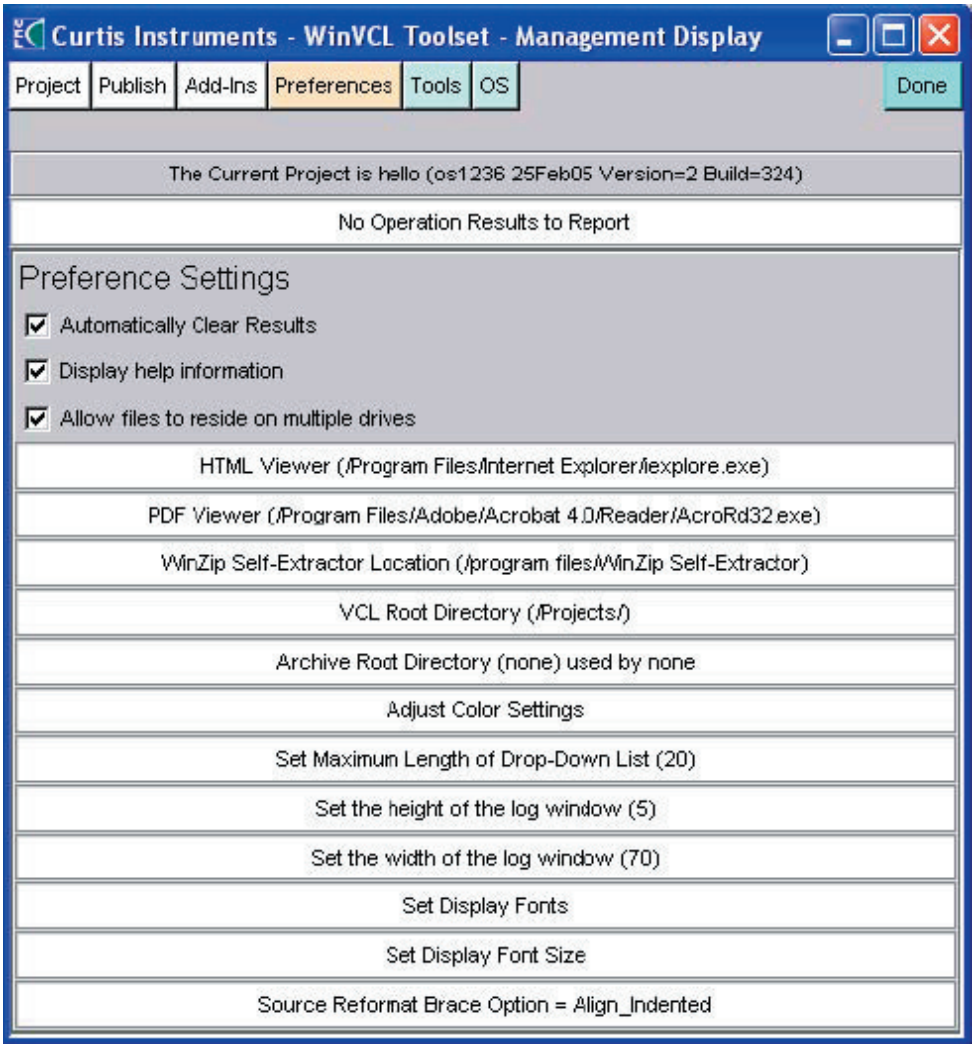


Clicking on “Add a component” brings up a dialog box that lets you select any file of your choice. These files are included in the project archive that is created whenever you explicitly publish an archive or in the archive that is automatically created when you publish a distribution.

Clicking on “Remove a component” brings up a list of previously added components. Simply click on any add-in components you wish to remove.

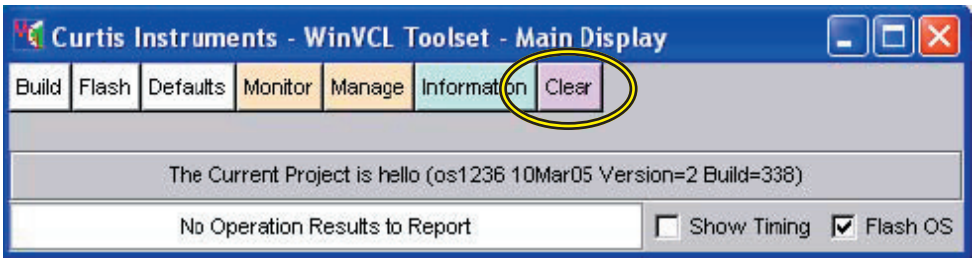
5.4 PREFERENCES BUTTON

These settings allow you to customize the system to your own personal preferences and needs. Clicking the Preferences button expands the management display to show three checkboxes, under the heading "Preference Settings". By default, all three checkboxes are checked. Below these are twelve full-width bars.



5.4.1 Preferences checkboxes

Unchecking Automatically Clear Results causes a Clear button to appear at the end of the row of function buttons in both the main and management displays:



As results are no longer automatically cleared, the Clear button provides a means to clear them manually.

Normally, operation results are automatically cleared each time you initiate a new operation. Unchecking this box allows you to leave the last results in the Operation Status bar, perhaps for purposes of comparison.

Unchecking Display help information stops the system from displaying the help text that normally appears in the help line.

Unchecking Allow files to reside on multiple drives causes the system to automatically remove the drive specification for all path names. This can be useful when you want to share projects between users, each of whom may have a different root drive.

5.4.2 Preferences bars

Clicking on HTML Viewer brings up a file dialog allowing you to select the program you'll use to view HTML files. The HTML viewer (internet browser) is used to display the SysInfo file for your current operating system.

Clicking on PDF Viewer brings up a file dialog allowing you to select the program you'll use to view PDF files. The PDF viewer is used to display the online system documentation (exclusive of the SysInfo file).

Clicking on WinZip Self-Extractor Location brings up a file dialog allowing you to set the location of the WinZip self-extractor program, if it is installed on your system. If this program is not installed on your system, the Distribute_App button on the Management display will not be displayed.

The VCL Root Directory determines where VCL will place new projects (new projects are created using the Project button option "Create a new project"). Clicking on this bar brings up a standard file dialog that allows you to reposition the VCL root directory. This should only be done when absolutely necessary as it may complicate sharing and distributing projects within your organization.

The Archive Root Directory allows you to specify a directory to hold archive distributions. Recall that the system always creates an archive of the current application's source files in a directory under the root of your project called Archive. The Archive Root Directory is an additional directory that can reside on a public directory. Clicking on this bar provides a directory selection dialog, and then asks for a user name. The user name will be suffixed to the application distribution names.

The remaining bars allow you to customize the appearance of WinVCL: the colors, fonts, font sizes, log window size, etc.

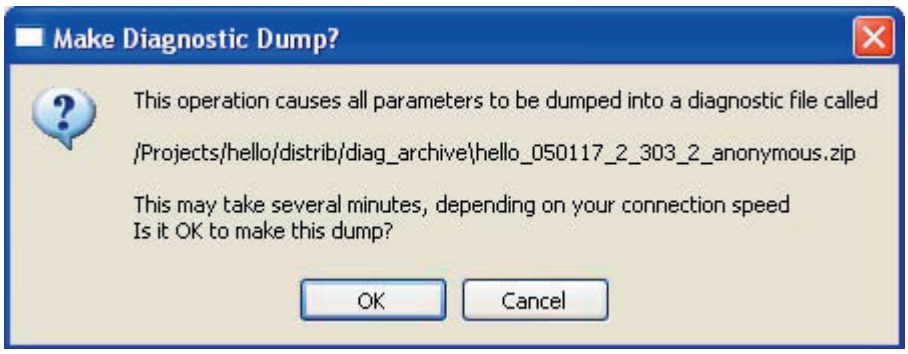
5.5 TOOLS BUTTON

The Tools button provides access to two functions that are not part of the standard workflow: “Make a diagnostic dump” and “Reformat VCL source code appearance”.

5.5.1 Make a diagnostic dump

This selection is used to create diagnostic dump file, which can be used by our development engineers to identify and remedy certain kinds of problems. This file is only useful to Curtis engineers, who will explicitly request that you supply them with this file.

The process will take several minutes, assuming a high baud rate connection, and much longer at lower baud rates. When you click on “Make a diagnostic dump” you will see a confirmation screen:



This screen tells you the name of the file that will be created and its location. It also warns you that a diagnostic dump operation can take a considerable amount of time to complete. Finally, it gives you the opportunity to cancel the operation in case you clicked this selection by mistake.

As the confirmation screen reminds you, the diagnostic dump will be placed in the /distrib/diag_archive subdirectory of your project directory.

5.5.2 Reformat VCL source code appearance

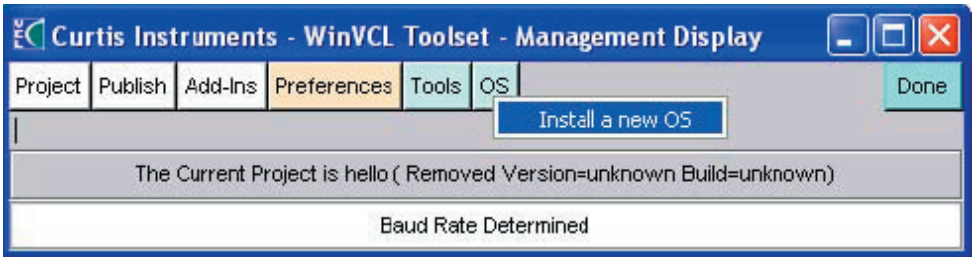
This selection causes a “pretty printer” program to run, which will make your code’s appearance uniform. You can use the Preference button’s last bar (“Source reformat brace option”, see Section 4.4.2) to set the way braces are placed.

5.6 OS BUTTON

The OS button allows you to install a new operating system (OS), restore a previously installed operating system, or remove a previously installed operating system. Clicking on this button causes a dropdown list of these three options to appear.

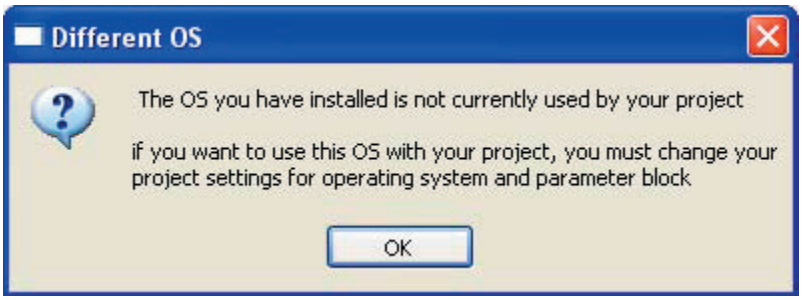
5.6.1 Installation of a New OS

The first item on the dropdown list is always "Install a new OS"; if you have not yet installed an operating system, this will be the only item on the list:

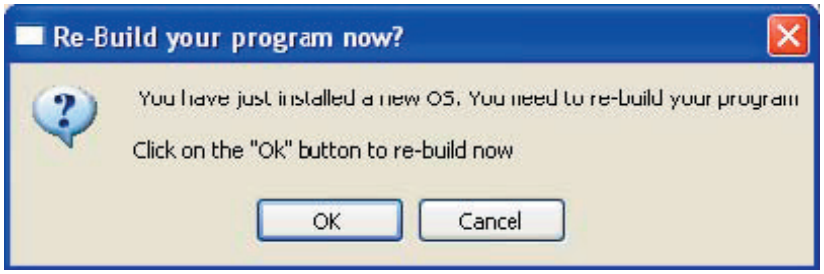


Selecting "Install a new OS" provides a list of directories and OS files (operating systems have an extension of *.os*). Move to the directory that contains the new operating system you wish to install and open that OS file.

If you install an OS that is not the one currently selected for the project, you will see this dialog box:



If you install an OS that is the one currently selected for the project (e.g., os1236), you will see this dialog box, which gives you the opportunity to immediately rebuild your project:



In either case, if you open the log window (by clicking on the Operation Results bar) you will see the distribution identification details and release notes for the operating system you've just installed.

5.4.2 Reinstallation / Removal of a previously installed OS

Installation of a new operating system does not destroy the current operating system (if any). Each time an operating system is installed, a copy is put into WinVCL OS archives. This allows you to reinstall a previous OS, should the need arise.

The Reinstall option allows you to change easily between operating systems. The selection “Reinstall a previously installed OS” provides a list of one or more operating system names. Move the cursor over an OS name to see a dropdown list of all the previously installed versions of that operating system. Click on the version you wish to reinstall.

The Remove option is used to reduce clutter by removing operating systems that are no longer used. The selection “Remove a previously installed OS” provides a list of one or more operating system names. Move the cursor over an OS name to see a dropdown list of all the previously installed versions of that operating system. Click on the version you wish to remove.

6

PROGRAM TIMING

The VCL compiler can generate timing information. If you check the box on the main display labeled Show Timing, the next time you click on the Build button, timing information will appear in the log window.

The VCL compiler translates your program into an equivalent set of pseudo-codes (p-codes). Each instruction results in one or more p-codes being generated. It is this set of p-codes that will actually be executed by the run-time system.

Clearly, to determine program timing, you need to know how many p-codes there are in your program. Although this information is necessary, it is still not sufficient to determine your program's timing because all but the most trivial programs have conditional statements that alter the execution sequence. Rather than knowing how many total p-codes there are in your program, it would be much more useful to know how many p-codes there are between points where the program can alter its path of execution. As you will see, this is exactly how VCL presents timing information.

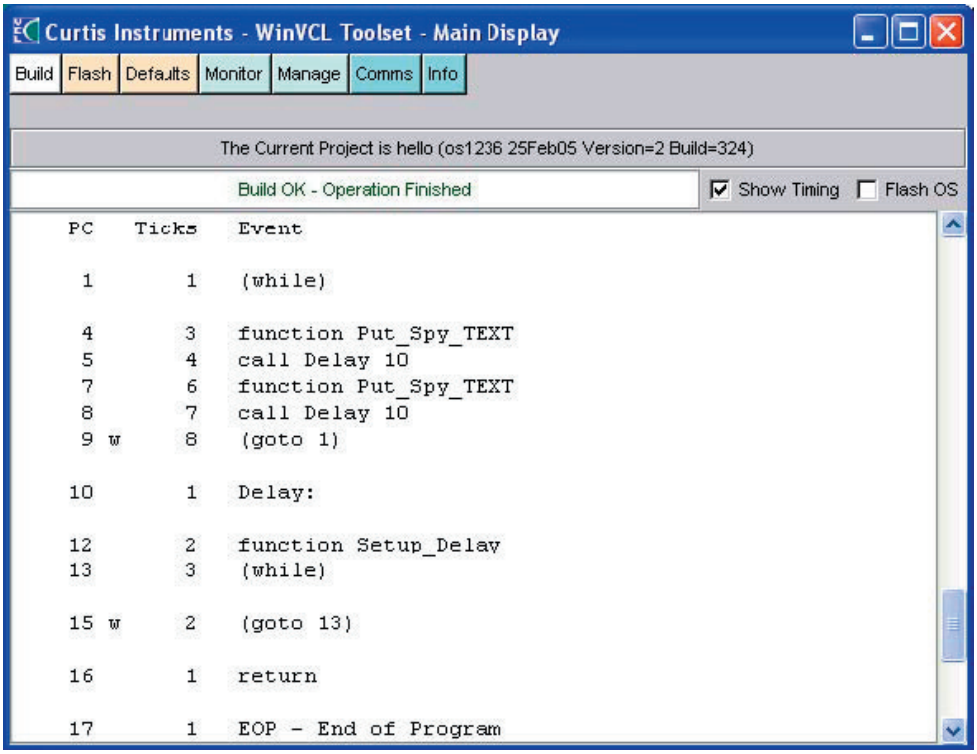
Here is a slightly altered version of the Hello program. The delays have been extracted to a common subroutine so that you can see the effect in the timing information.

```
;-------------------------------------------------
; Main :: Main Line of the Hello World Program
;-----------------------------------------
;
while (1=1)
    {
    put_spy_text(" Hello ")
    call Delay
    put_spy_text(" World ")
    call Delay
    }


;-------------------------------------------------
; Delay :: Run the Fixed Delay
;----------------------------
;
Delay:
    setup_delay(DLY1,500)
    while (DLY1_Output <> 0) {}
    return
```

As it turns out, this program has a total of 16 p-codes executed. As you can imagine, that will not help you very much in answering questions about program timing. For example, how much time is spent in the main loop as opposed to the delay routine? In this example, you could make a reasonably good guess (i.e., a lot of time in the delay routine and not very much at all in the main loop); however, in most real-world programs, the answer will not be nearly so clear.

Compiling this program with the Show Timing button checked yields these results:



As you can see, there are three columns: PC, Ticks, and Event.

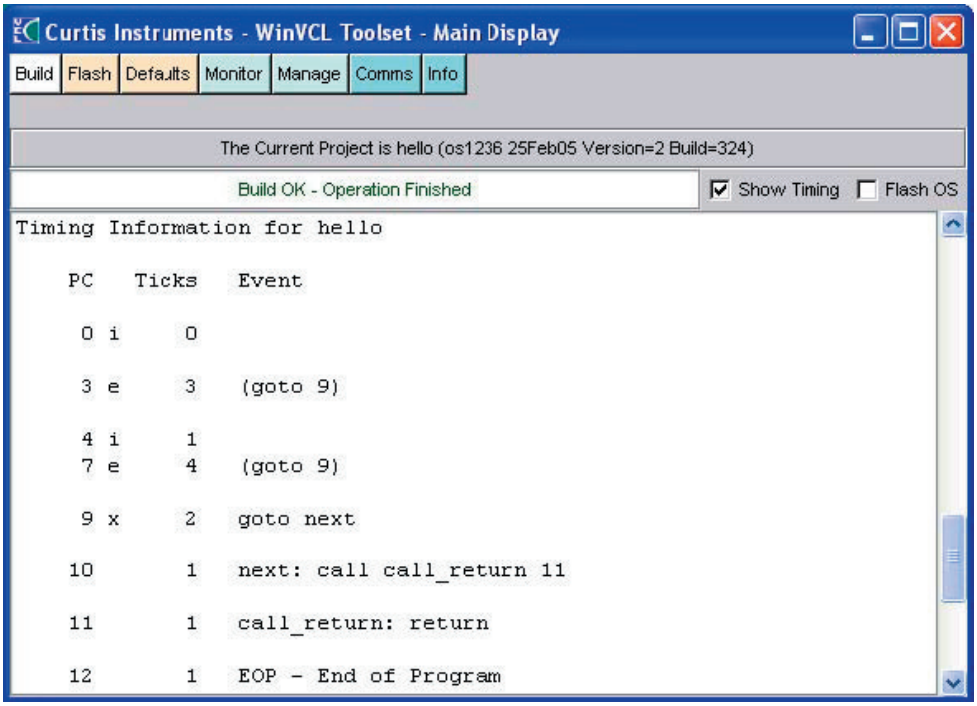
The PC column shows the Program Counter's value. The PC column increments by one for each p-code. From this listing, you can see that there are 16 p-codes total. (The 17 at the bottom of the PC column does not count, as this line is simply a flag to indicate the end of the program has been reached.)

The Ticks column shows the number of p-codes executed since the last potential break in program flow was encountered. The Ticks column will continue incrementing until a *while*, *goto*, *call*, or *return* statement is encountered. At that point, the current number of ticks is printed and Ticks is reset to 1. If you add up the value of Ticks before each space, you will find it equals the total number of p-codes (i.e., the final value of the PC). In this case, the final value of the PC is 16, and the total number of Ticks is 1+8+1+3+2+1, which is also 16.

An "event" in the Event column is something that might change the timing of your program, plus any VCL functions (this last is helpful in relating the timing output to your source code). Specifically, any *goto*, *call*, *return*, *enter*, or *exit* is a point at which your program may (or must) change its program counter. Notice that when a label is encountered, it is counted as an event. This is so you can insert labels specifically to found out the number of ticks between them.

You may have noticed that sometimes letters appear just to the right of the PC column. These letters indicate instructions that may implicitly change the timing of the program. Specifically, *if-else_if-else* statements cause *goto* statements to be

automatically inserted into your program. These are marked by the letters *i*, *e*, and *x*. An *i* is inserted when an *if* statement is encountered. An *e* is inserted when an *else* statement is encountered. (An *e* and an *i* are both inserted if an *else if* statement is encountered.) An *x* appears to mark the first instruction following a compound *if-then-else* chain. For example, this display is the result of processing the program fragment presented below it.




```
variable1 alias user1
variable2 alias user2
variable3 alias user3
        ; If statement
        if (variable1 = variable2)
                {
                variable1 = 1
                }
        else if (variable1 = variable3)
                {
                variable1 = 2
                }
        else
                {
                variable1 = 3
                }

        ; Goto Statement
        goto next
next:
        ; Call Statement
        call call_return

call_return:
        return
```

Keep in mind that the timing display is a static analysis tool. The timing information is generated by an analysis of the compiled sequence of p-codes, not by running them. You will need to use your knowledge of the program and the system in which it is to run to decide whether the program will take a given branch.

APPENDIX A
WINVCL SUBPROGRAMS

WinVCL has a number of subprograms, or tools. When you click on a button, you will often invoke one or more of these subprograms. As each subprogram runs, its name is announced on the Operation Status bar. If there are any errors, the subprogram's name will appear in the error message. The following brief subprogram descriptions are provided to help you more effectively locate and remedy errors.

Filecopy

Filecopy copies files from one location to another.

Flash

Flash is used to program the micro-controller's flash memory. It works in conjunction with a DLL (ST10Flasher.dll) supplied by STMicroelectronics. Flash is used to estimate the amount of time it will take to program the controller and also to download the program.

GetUdir

GetUdir is used to find the **u**ser's current **dir**ectory.

Make_CBF

Make_CBF concatenates the two BSL files (OS and Application Package) into a single file (**C**ombined **B**SL **F**ile) with an identifying header.

Monitor

Monitor is used to communicate with the controller while you are using the Monitor screen.

Package

Package is responsible for building the final loadable hex application package. Package accepts one or more properly wrapped files (see Wrapper) and combines them into a single application package, which can be downloaded into the controller using Flash.

PIR

PIR (**P**arameter **I**nformation **R**eader) reads through one or more files, looking for parameter generation keywords. It outputs a set of files that are used by POUT (see below). PIR will only be run if you have specified parameter files in your project definition using the Add or Delete Parameter Files button.

POUT

POUT (**P**arameter **OUT**put) accepts a set of files generated by PIR and outputs a properly formed parameter block. POUT will only be run if you have specified parameter files in your project definition using the Add or Delete Parameter Files button.

PP

PP reformats your VCL source code on demand. The reformat command is one of the choices in the Tools button's dropdown list; see Section 5.5.2, page 26.

Stitch

Stitch is used to combine your user parameter set with the selected base parameter block. Stitch will only be run if you have specified parameter files in your project definition using the Add or Delete Parameter Files button and you have specified a base parameter block using the Curtis Supplied Base Parameter Block button. In this case, a separate parameter block will be generated which Stitch will combine with the base parameter block. The output of Stitch must be sent through Wrapper before it is included in the final application package.

VCL

VCL is the compiler that translates your source code into pseudo-code (p-code) that is interpreted by the target system. VCL accepts one or more source files and outputs two binary files for each one. The binary files have the same base name as the input file, but with different extensions. The *.cmd* file holds the p-codes generated as a result of the compilation. The *.str* file holds any strings found in the source code. These files are properly formatted (i.e., pre-wrapped) for inclusion in the final application package.

Wrapper

Wrapper is responsible for generating and adding the header and checksum information required by Package. Wrapper can be used with any file to generate a file that can be added to the application package. (Note: Some programs—such as VCL—create output files that are already properly wrapped.)

APPENDIX B
COM PORT POWER MANAGEMENT

Sometimes the flash download program fails to connect, issuing the message "Error: Could not communicate with controller - is it connected?" or, less often, "Bad Acknowledge value". If flash fails to connect, you may need to disable power management for the COM port you are using.

Background

Portable computers are typically configured to try to conserve battery life by turning off some of their subsystems. In recent years, desktop computers have also been enabled to save power by turning off subsystems. In a Windows environment, the power control system is called Advanced Power Management, or APM, which is based on an industry standard called Advanced Configuration & Power Interface, or ACPI.

Although APM is a good idea, it does cause problems for some devices. If the COM port used for flash downloading has been turned off, its idle electrical state will be the opposite of what the controller connected to it would expect. This prevents the controller from properly responding to the "wake up" message.

You can avoid this problem by creating a registry entry that disables power management for the COM port you are using. You should backup your registry before making any changes. You will use the registry editor that comes with your distribution of Windows (REGEDIT.EXE in the WINDOWS or WINNT directory).

Windows 2000 (service pack 3 and higher)

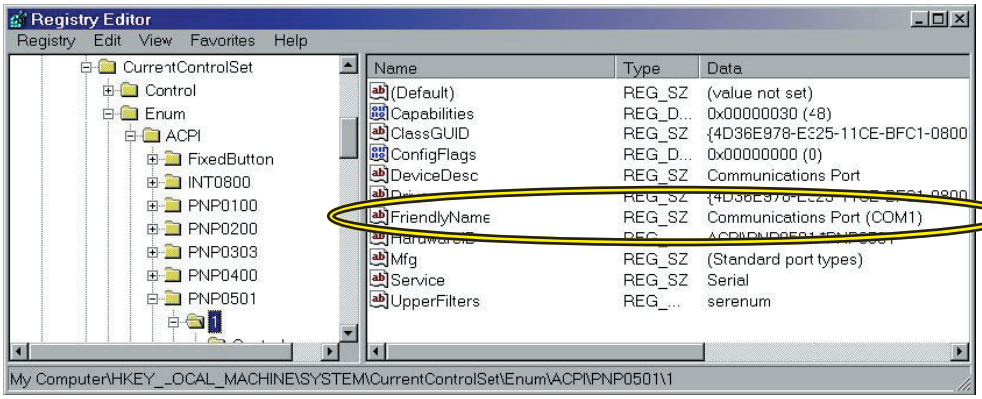
Open Registry Editor and use it to find the key:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Serial\Enum\ACPI
```

Using Registry Editor, search for the subkey that contains the name of your COM port. This will be a string listed for a key called FriendlyName. For example,

```
FriendlyName REG_SZ Communications Port (COM1)
```

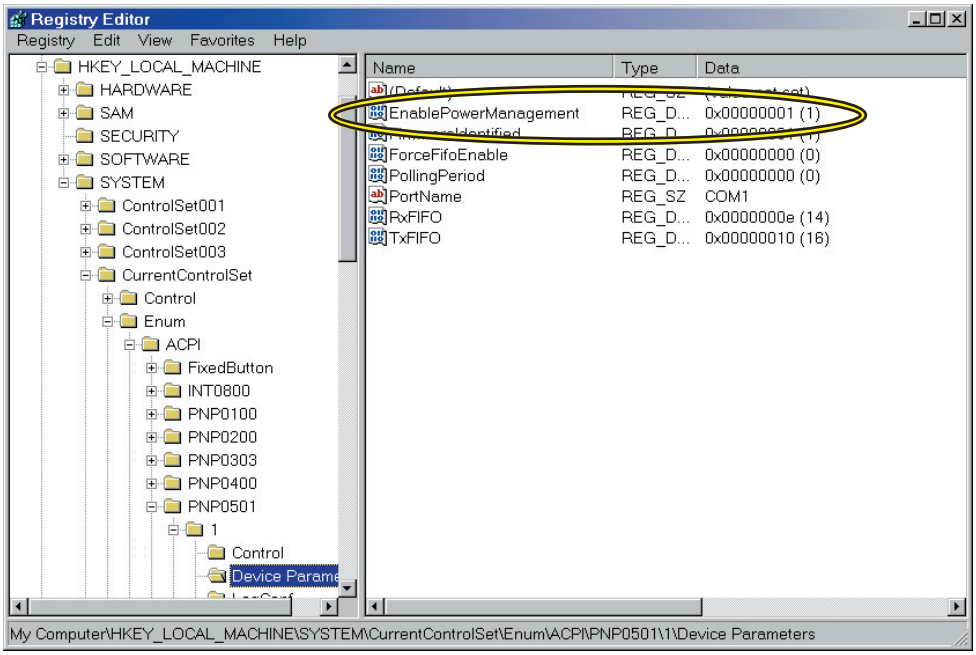
Here is an example of what the registry entry looks like.



Now go one level deeper to view the Device_Parameters subkey.

If there is no DWORD entry called EnablePowerManagement in the Device_Parameters subkey, you will have to create one. Right-click on Device_Parameters to see a dropdown menu. Click on New, and then select DWORD value. Change the name of the newly created DWORD value from New Value #1 to EnablePowerManagement. The case of the characters is important.

Next, double-click on EnablePowerManagement to open up the Edit DWORD Value dialog. Change "Value data" from 0 to 1 and then click on OK (yes, "1" *disables* power management). Here is an example of the EnablePowerManagement key once it has been defined and its value set.



Finally, close Registry Editor and restart the computer.

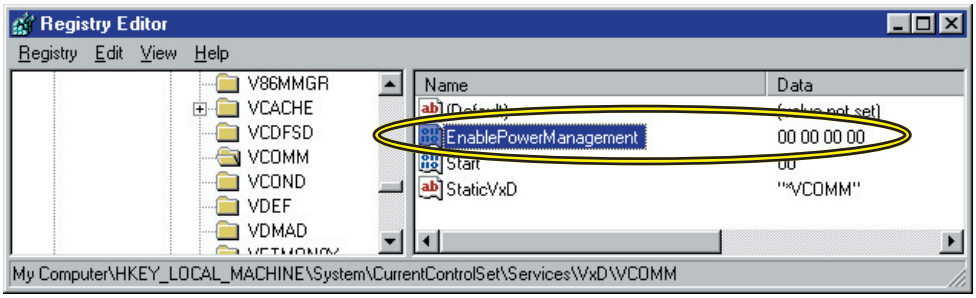
Windows 95/98

Open Registry Editor and use it to find the key:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\VxD\VCOMM
```

You will see a key called EnablePowerManagement. The data for this key is a hex string with a value of 01 00 00 00. You will need to change the first hex value from 01 to 00.

Double-click on EnablePowerManagement to open up the key editor. Press the Delete key on your keyboard to delete the first entry (the 01), and then press the 0 key (the number 0 not the letter O) to enter a value of 00 for the first entry. Make sure the hex string still has four values, 00 00 00 00; changing the length of the string will cause problems. Once you have changed the value, click on the OK to enter it. Here is an example of the key once its value has been altered.



Finally, close Registry Editor and restart the computer.

References

These references are all Microsoft Knowledge Base Articles:

- Article 256986 “Description of the Microsoft Windows Registry”
- Article 275042 “Modem Powers Off When It Is Not in Use”
- Article 199209 “How to Disable Advanced Power Management for PC Card Modems”

APPENDIX C
READ-ME FILE STRUCTURE

A “read-me” file is saved each time you distribute an application in the source file archive. The read-me file contains the comments you typed in when the distribution was created. The comments are preceded by information about the system that was used to create the distribution.

The first seven lines of the read-me file contain common system information. Your comments start on line 8. Here is the format:

1. Copyright message, the distribution type (OS or Application), and the date of distribution with the name suffixed to it with an underscore.
2. Operating system name.
3. Operating system version.
4. WinVCL version.
5. Parameter block name and version.
6. VCL compiler level.
7. Processor type, number of 64k blocks for VCL application package, Start Manager Present flag (yes or no), and Start Manager file name.
8. Comments.

Here is an example of a read-me file. The line numbers do not appear in the read-me file; they are added in this example to make it easier to correlate each line with the preceding description.

1. `Copyright (c) 2002-2005 - Curtis Instruments, Inc. - Application Distribution, 19Jul04_hello`
2. `os1236`
3. `os1236_050228_2_324_unreleased`
4. `WinVCL 6.1.5.a`
5. `os1236 100`
6. `Current_Version`
7. `ST269 1 yes StartManager .H86`
8. `First distribution of "hello" project.`

APPENDIX D
OPERATING SYSTEM AND APPLICATION NAMING CONVENTIONS

Operating systems and application distributions have a stylized method of naming, as follows.

Operating System Names

Operating system names consist of four parts, separated by underscores:

1. The name of the operating system always appears first (e.g., os1236).
2. The date on which the operating system was created, in strict numeric form. The year comes first, followed by the month and the day, with two digits per element (040719 = 19 July 2004).
3. The third part contains the version number and build number, separated by an underscore (e.g., 1_0).
4. The last part, or suffix, is generally the model number followed by the parameter block version number (e.g., 12365302_101). You may receive an interim distribution, in which case the suffix will be the developer's name (e.g., anonymous).

Examples of operating system names:

os1236_040719_1_0_12365302_101
os1236_040719_0_3_anonymous.

Application Distribution Names

Application distribution names are very similar to operating system names. They also consist of four parts:

1. The name of the project always appears first (e.g., hello).
2. The date on which the distribution was created, in strict numeric form. The year comes first, followed by the month and the day, with two digits per element (040719 = 19 July 2004).
3. The third part contains the two-part versioning information from the operating system combined with a one or two digit (1 to 99) "sequence" number (e.g., 1_0_12). The sequence number starts at one and increments up by one for each distribution that you create. The sequence number is reset to one at the beginning of the day.
4. The last part, or suffix, is either the name "anonymous" or the name you provided when you defined your public archive directory (see section 5.4.2: Preferences Button, page 25).

Examples of application distribution names:

hello_040719_0_3_anonymous
hello_040719_1_0_Ralph.

APPENDIX E
COPYRIGHTS AND ACKNOWLEDGEMENTS

Some of the software included in WinVCL is the result of the effort of various generous individuals and groups.

ST10FLASHER.dll

Provided by:

T.P.A. Division
DMD Group
STMicroelectronics
60, rue Lavoisier
38330 Montbonnot St. Martin
France

Tclkit-WIN32.exe

License: Tclkit is a combination of Tcl, Tk, IncrTcl, TclVFS, Zlib, and Metakit. Metakit is open source, using an X/MIT-style license. Tcl/Tk, IncrTcl, TclVFS, and Zlib have their own BSD-ish open source licenses, so in my understanding this code can be used freely, also commercially.

Acknowledgements: The original release of Tclkit relied heavily on Matt Newman's Tcl implementation of VFS, a "Virtual File System" layer for Tcl. As of March 2002, Tclkit uses the next generation C implementation by Vince Darley, which has become part of the standard Tcl core. The contributions and feedback of both are very gratefully acknowledged.

ZIP.exe, ZIPNote.exe, and UNZIP.exe

For the purposes of this copyright and license, "Info-ZIP" is defined as the following set of individuals:

> Mark Adler, John Bush, Karl Davis, Harald Denker, Jean-Michel Dubois, Jean-loup Gailly, Hunter Goatley, Ian Gorman, Chris Herborth, Dirk Haase, Greg Hartwig, Robert Heath, Jonathan Hudson, Paul Kienitz, David Kirschbaum, Johnny Lee, Onno van der Linden, Igor Mandrichenko, Steve P. Miller, Sergio Monesi, Keith Owens, George Petrov, Greg Roelofs, Kai Uwe Rommel, Steve Salisbury, Dave Smith, Christian Spieler, Antoine Verheijen, Paul von Behren, Rich Wales, Mike White.

This software is provided "as is," without warranty of any kind, express or implied. In no event shall Info-ZIP or its contributors be held liable for any direct, indirect, incidental, special, or consequential damages arising out of the use of or inability to use this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. Redistributions of source code must retain the above copyright notice, definition, disclaimer, and this list of conditions.

2. Redistributions in binary form (compiled executables) must reproduce the above copyright notice, definition, disclaimer, and this list of conditions in documentation and/or other materials provided with the distribution. The sole exception to this condition is redistribution of a standard UnZipSFX binary as part of a self-extracting archive; that is permitted without inclusion of this license, as long as the normal UnZipSFX banner has not been removed from the binary or disabled.

3. Altered versions—including, but not limited to, ports to new operating systems, existing ports with new graphical interfaces, and dynamic, shared, or static library versions—must be plainly marked as such and must not be misrepresented as being the original source. Such altered versions also must not be misrepresented as being Info-ZIP releases—including, but not limited to, labeling of the altered versions with the names "Info-ZIP" (or any variation thereof, including, but not limited to, different capitalizations), "Pocket UnZip," "WiZ" or "MacZip" without the explicit permission of Info-ZIP. Such altered versions are further prohibited from misrepresentative use of the Zip-Bugs or Info-ZIP e-mail addresses or of the Info-ZIP URL(s).

4. Info-ZIP retains the right to use the names "Info-ZIP," "Zip," "UnZip," "UnZipSFX," "WiZ," "Pocket UnZip," "Pocket Zip," and "MacZip" for its own source and binary releases.

The definitive version of the license document should be available indefinitely at ftp://ftp.info-zip.org/pub/infozip/license.html.

All Other Programs

Produced and copyrighted by:

Curtis Instruments, Inc.
200 Kisco Avenue
Mt. Kisco, NY 10549 USA