Windows Embedded Compact Test Kit User Guide

Writers: Randy Ocheltree, John Hughes

Published: October 2011

Applies To: Windows Embedded Compact 7

Abstract

The Windows Embedded Compact Test Kit (CTK) is a tool that you can use to test the functionality and performance of device drivers and related hardware for a Windows Embedded Compact device. The test tools in the CTK provide feedback on the functionality of your drivers, which can in turn be used to further enhance the reliability of a Windows Embedded Compact device platform. This paper describes how to use the Windows Embedded Compact Test Kit.

Contents

Introduction to the Windows Embedded Compact Test Kit

The Windows Embedded Compact Test Kit (CTK) provides significant improvements to the user interface and the overall feature set when compared to the previous release known as Windows Embedded CE Test Kit (CETK).

The Windows Embedded CTK is a tool that you can use to test the functionality and performance of device drivers and related hardware for a Windows Embedded Compact powered device. The test tools in the CTK provide feedback on the functionality of your drivers, which can in turn be used to further enhance the reliability of your device. Additionally, more tests can be added to the CTK to test specific drivers.

You can use the integrated Graph Tool to convert performance test results into graphical charts. For information about the Graph Tool, see Using the Graph Tool.

The Windows Embedded Compact Test Kit (CTK) Automation Tool Solution (CATS) can be used to automate a CTK test pass. For information about the CATS Tool, see Test Automation with the CTK Automation Tool Solution (CATS). For running stress testing scenarios, the Compact Stress tool is a remote tool used for performing stress testing on a device. For information about the Compact Stress tool, see Stress Testing with the Compact Stress Tool.

Prerequisites

The Windows Embedded Compact Test Kit (CTK) requires the following:

- Microsoft Platform Builder for Windows Embedded Compact 7
- Windows Embedded Compact 7
- A hardware device to test or a virtual CEPC
- Microsoft .NET Compact Framework 2.0 SP2

The Graph Tool requires the following:

- Microsoft Office 2007 or Microsoft Office 2010 (the Graph Tool is not compatible with earlier versions of Office)

The Windows Embedded Compact Test Kit (CTK) Automation Tool Solution (CATS) tool requires the following:

- A mechanism for triggering a device power cycle without user input

What's New

The Windows Embedded Compact Test Kit (CTK) is a new application with the following key features:

- Improved user interface that resembles Visual Studio

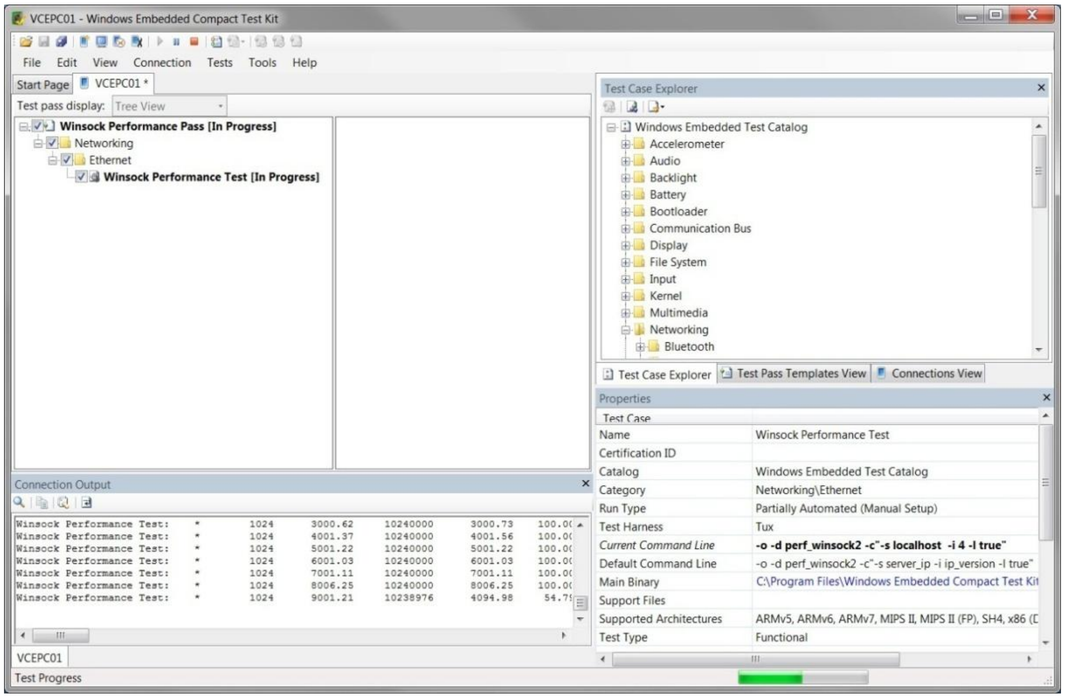
- Ability to group user-selected test cases into test passes
- Improved test results viewer lets you store and review test results per test pass
- Upon connection to a device, detection of peripherals and drivers needed for tests
- Integration with the new Graph Tool to graph performance test results
- Automated test passes using the new Windows Embedded Compact Test Kit (CTK) Automation Tool Solution (CATS) tool
- Stress testing using the new Compact Stress tool
- Users can add custom Tux test harness (TUX)-based tests to the CTK
- Support for x86, MIPSII, MIPSII_FP, ARMv5, ARMv6, ARMv7, and SH4 processors

Improved User Interface

The Windows Embedded Compact Test Kit (CTK) user interface (UI) consists of an integrated set of windows, tools, menus, toolbars, and other elements that allow you to execute, view and manage your Windows Embedded Compact tests in one place.

The user interface uses standard Windows interface functionality that resembles the Visual Studio UI. It displays multiple windows that show information and data pertinent to test cases.

Figure 1 - The Windows Embedded Compact Test Kit



```
Winsock Performance Test:     *      1024     3000.62    10240000     3000.73    100.0(
Winsock Performance Test:     *      1024     4001.37    10240000     4001.56    100.0(
Winsock Performance Test:     *      1024     5001.22    10240000     5001.22    100.0(
Winsock Performance Test:     *      1024     6001.03    10240000     6001.03    100.0(
Winsock Performance Test:     *      1024     7001.11    10240000     7001.11    100.0(
Winsock Performance Test:     *      1024     8006.25    10240000     8006.25    100.0(
Winsock Performance Test:     *      1024     9001.21    10238976     4094.98     54.7!
```

Test Case	
Name	Winsock Performance Test
Certification ID	
Catalog	Windows Embedded Test Catalog
Category	Networking\Ethernet
Run Type	Partially Automated (Manual Setup)
Test Harness	Tux
Current Command Line	**-o -d perf_winsock2 -c"-s localhost -i 4 -l true"**
Default Command Line	-o -d perf_winsock2 -c"-s server_ip -i ip_version -l true"
Main Binary	C:\Program Files\Windows Embedded Compact Test Kit
Support Files	
Supported Architectures	ARMv5, ARMv6, ARMv7, MIPS II, MIPS II (FP), SH4, x86 (D
Test Type	Functional

The Windows Embedded Compact Test Kit window, shown in **Figure 1**, is made up of an icon toolbar at the top, a menu toolbar, four primary windows, and a status bar at the bottom of the window.

The four primary windows are as follows:

1. The **Test pass display** window in the upper left displays the current test pass and is divided into two parts:
   - The left side of the window displays a test pass in either **Tree View** or a flat **List View**.
   - The right side of the window displays the Help documentation for the test case currently selected in the left side.
2. The **Test Case Explorer** window in the upper right displays the following tabs:
   - The **Test Case Explorer** tab displays the master test catalog in a tree view. It contains the Windows Embedded test catalog by default. Other catalogs may be added in the future. For more information, see Windows Embedded Test Catalog.
   - The **Test Pass Template View** tab displays the user-created and built-in test pass templates. For more information on test pass templates, see Creating a Test Pass Template.
   - The **Connection View** tab displays the device the CTK is connected to. For more information, see Connecting to a Device.
3. The **Properties** window in the lower right is similar to the **Properties** windows in Visual Studio. It displays the properties of the currently selected test case, test pass, template, or connection. For more information, see Viewing Test Properties.
4. The **Output** window in the lower left displays debug messages from the CTK and from a test case as it runs.

Test Pass Icons

The following icons are displayed next to test cases in the **Test pass display** window to describe the level of interaction needed to run the tests.

Icon	Description
▶	Fully automated
✋▶	Partially automated, requires some manual interaction
✋	Manual, requires total manual interaction

Icon	Description
🚫	The attached device does not have the required peripherals required to run the test as determined by peripheral detection

Compact Test Kit (CTK) Quick Start

This section gives a quick introduction to using the Windows Embedded Compact Test Kit (CTK). The following steps guide you through the process of connecting to a device, creating a test pass, running a simple test, and viewing the results of the test.

- Step 1 – Connect to a Device
- Step 2 – Create a Test Pass Template
- Step 3 – Create a Test Pass from the Template
- Step 4 – Run a Test
- Step 5 – View Test Results

Step 1 – Connect to a Device

A device must be running and attached to Platform Builder before you can connect to the device from the Windows Embedded CTK. If you prefer, you can create a test pass template before you connect to a device.

To connect to a device

1. In Windows Embedded Compact Test Kit, on the **Connection** menu, click **Connect to Device**.
2. Select the device you want to connect to from the **Select a Device** dialog box and then click **OK**.

   The **Connecting to a Device** dialog will appear displaying the name of the device you are trying to connect to and the progress of the connection attempt. If the connection to the device is successful, the **Connection Output** window displays the “Successfully established connection!” message. If a device has previously been connected to, the **Test pass display** window opens and displays “No test pass” or a default test pass.”

You are now ready to create a test pass template.

For more information, see Connecting to a Device.

Step 2 – Create a Test Pass Template

Before creating a test pass you must create a template to base the test pass on.

To create a test pass template

1. On the **File** menu, point to **New**, and then click **Test Pass Template**.

   The **Test Manager** window appears, displaying the **Test Case Explorer** on the left side of the window and two tabs on the right side, **Test Case Information** and **Test Pass Templates**.
2. On the **Test Pass Templates** tab, under **Edit Templates**, click the **New** button.
3. In the new template name text box that appears, type **Registry APIs** to rename the template from **New Template**.
4. In the Test Manager **Test Case Explorer** tree view display area, double-click **Windows Embedded Test Catalog** to expand the catalog tree.
5. In the catalog tree, double-click the **File System** folder, and then double-click **Registry**.
6. Click **Registry API and Functionality Test**, and then click the **Add >** button to add the selected test to the new **Registry APIs** template.
7. Click **Done** to exit **Test Manager**. Your template will be saved.

You are now ready to create a test pass from the template you just created.

For more information, see Creating a Test Pass Template.

Step 3 – Create a Test Pass from the Template

Once you have created the test pass template and connected to a device, you can create a test pass.

To create a test pass from a template

1. On the **View** menu, click **Test Pass Templates View**.
2. In the **Test Case Explorer** window **Test Pass Templates View** tab, click the **Registry APIs** template.
3. Right-click the highlighted **Registry APIs** template and then click **Use as Current Test Pass** on the pop-up menu.

   Note

   If **Use as Current Test Pass** is not available on the pop-up menu, you most likely are not connected to a device.

   Note

   If a test pass was already in the **Test Pass** display window, you'll be asked if you want to save the current test pass before a new test pass is created.
4. In the **Test Pass** display window, click the new test pass **Windows Embedded** and type **Registry API Test Pass** to rename it.

For more information, see Creating a Test Pass from a Template.

Step 4 – Run a Test

You can run the test once **Registry API and Functionality Test** has been added to a test pass.

To run a test

1. In the **Test pass display** window, click the plus sign (+) preceding **Registry API Test Pass** to expand the test pass tree.
2. Right-click on **Registry API and Functionality Test** and click **Run Test**.

When the **Test pass display** window changes from **Registry API and Functionality Test [In Progress]** to **Registry API and Functionality Test [Passed]**, the test run is complete and you are ready to view the results of the completed test.

The green status bar at the bottom of the Windows Embedded CTK main window displays the progress of the test. As the test runs, Kernel Independent Transport Layer (KITL) debug message output is displayed in the **Connection Output** window.

For more information, see Running a Test Pass.

Step 5 – View Test Results

Once the test run is complete you are ready to view the results of the test.

To view the results for a test

1. In the **Test pass display** window, select **Registry API and Functionality Test [Passed]**, right-click and then select **View Test Results**.
2. In the **Common Results View** window tab that opens, locate the completed test run in the **Test Case Results History** table, and click the folder icon in the **Log File(s)** column.
3. In the **Windows Explorer** window that opens, locate in the folder containing the log files for the test run and open the results.log file to view the test debug output in Tux test harness (Tux) format.

   Note

   The test result logs are stored using the following folder hierarchy: `C:\Users\<username>\Compact Test Kit\results\.`

For more information, see Viewing Test Results.

Connecting to a Device

Before trying to connect to a device from the Windows Embedded Compact Test Kit (CTK), your device must be running and attached to Platform Builder.

To connect to a device

1. In Windows Embedded Compact Test Kit, on the **Connection** menu, click **Connect to Device**.
2. In the **Select a Device** dialog box, select the device you want to connect to and then click **OK**.

   The **Connecting to a Device** dialog will appear displaying the name of the device you are trying to connect to and the progress of the connection attempt. If the connection to the device is successful, the **Connection Output** window displays the "Successfully established connection!" message. The **Test pass display** window opens and displays "No test pass" or a default test pass.
3. Do one of the following:
   - Create a new test pass. For more information, see Creating a Test Pass.
   - Run the default test pass. For more information, see Running a Test Pass.
   - Add to the default test pass. For more information, see Adding a Test Case to a Test Pass.

To disconnect a device

- In Windows Embedded Compact Test Kit, on the **Connection** menu, click **Remove Connection**.

To troubleshoot a device connection

- View the messages displayed in the **Connection Output** window.
- View the messages in Platform Builder in the Windows CE Debug **Output** window.
- Verify that the CPU type of the device that you are trying to connect to is supported by the CTK. Verification of the CPU type can be attained if you have access to your source code and by viewing what the CEInstructionSet is set to. If you have Windows Embedded Compact 7 installed, this information is located in the file, `C:\WINCE700\platform\<platform>\src\oal\oallib\init.c`.
- Verify that the device you are trying to connect to is running and attached to Platform Builder.

Creating a Test Pass

Before you create a test pass, you must create a template to base your test pass on. This section describes what a test pass is, how to create a test pass template, and how to create a test pass from a template.

What Is a Test Pass

The Windows Embedded Compact Test Kit uses the concept of a “test pass” as a container for a collection of test suites, which are collections of test cases. The test cases in a test pass can be run all together in one pass or run selectively.

A test pass provides a way to organize your testing according to the way you run your tests. For example, you could have one test pass called SetTopBox that contains all of the multimedia, file system, Ethernet, and display tests that you want to run on your device platform. Then you could have another test pass called SetTopBox_Multimedia that contains only the multimedia tests. You would use the SetTopBox_Multimedia test pass if you want to run only the multimedia tests and the SetTopBox test pass if you want to run the complete set of tests.

A test pass can be saved and reused at a later time. Saved test passes will be preserved during upgrades to new versions of the Windows Embedded Compact Test Kit.

To create a test pass you first create a test pass template to base your test pass on.

Creating a Test Pass Template

A test pass is created from a test pass template. Once you’ve created a test pass template you can add and remove test cases from the template.

You can organize test cases into logically related test pass groups using the Windows Embedded Compact Test Kit (CTK) to create test pass templates. You can view these groups in the **Test Case Explorer** on the **Test Pass Templates View** tab.

There are two categories of test pass templates:

1. **Microsoft Windows Embedded Templates**, which are provided by Microsoft and organized into the following groups: Basic Verification, Functional, Performance, and Windows Embedded Compact.
    - The Basic Verification, Functional, and Performance templates are subsets of all tests available in the CTK. The selection of the tests that make up each test pass template is based on relevance to the goal of that test pass.
    - The Windows Embedded Compact template contains all the tests available in the CTK.
2. **My Templates**, which are all the templates created by you.

You cannot edit the **Microsoft Windows Embedded Templates**, but you can edit the **My Templates** that you create using the Test Manager. For more information about the Test Manager, see Managing Tests.

Creating a New Test Pass Template

You use the Test Manager to create new and edit existing test pass templates. For more information about the Test Manager, see Managing Tests.

To create a new test pass template

1. In Windows Embedded Compact Test Kit, on the **File** menu, point to **New**, and then click **Test Pass Template**.
2. On the right side of the **Test Manager** window that appears, on the **Test Pass Templates** tab, under **Edit Templates**, click the **New** button.
3. In the new template that appears, type the name you want to rename the template.
4. In the **Test Case Explorer** tree view display area, double-click **Windows Embedded Test Catalog** to expand the catalog tree.
5. Click on a test case tree node or an individual test case, and then click the **Add >** button to add the selected test case to your new template.
6. Continue to add test cases from the **Windows Embedded Test Catalog** until your template is complete.
7. If a test case needs to be removed, click on a test case tree node or an individual test case in your new template to highlight the node or test case you want to remove, then click the **< Remove** button.
8. Click **Done** to exit **Test Manager**.

Copying an Existing Test Pass Template

The Windows Embedded Compact Test Kit (CTK) provides prebuilt test pass templates that you can copy to create your own templates. The provided templates are the Windows Embedded Compact template that is based on the test cases in the Windows Embedded Test catalog and a Desktop template. You can copy test pass templates that you create as well.

To copy an existing test pass template

1. In Windows Embedded Compact Test Kit, on the **File** menu, point to **New**, and then click **Test Pass Template**.

   The **Test Manager** window opens displaying the **Test Case Explorer** on the left side of the window and two tabs on the right side: **Test Case Information** and **Test Pass Templates**.
2. On the **Test Pass Templates** tab, click to highlight the template you want to copy.

   Note

   You cannot copy test cases from a template. You can only copy the template as a whole.

3. Under **Edit Templates**, click the **Copy** button.
4. In the copied template that appears, type the name you want to rename the copied template.
5. If you want to customize the template, you can add and remove test cases.
6. Click **Done** to exit **Test Manager**.

Creating a Test Pass from a Template

Once you have a created a test pass template, you can create a test pass.

To create a test pass from a template

1. In Windows Embedded Compact Test Kit, connect to a device. For more information, see Connecting to a Device.
2. On the **View** menu, click **Test Pass Templates View**.
3. In the **Test Case Explorer** window **Test Pass Templates View** tab, click to highlight the test pass template you want to base your test pass on.
4. Right-click the highlighted template and then click **Use as Current Test Pass** on the pop-up menu.

   Note

   If **Use as Current Test Pass** is not available on the pop-up menu, you most likely are not connected to a device.
5. In the **Test Pass** display window, click the new test **Windows Embedded** that is highlighted and type a new name.

   Note

   If a test pass was already in the **Test Pass** display window, you'll be asked if you want to save the current test pass before a new test pass is created.
6. Click the plus sign (+) preceding the test pass name to expand the test pass tree.

   The new test pass contains all the test cases that are in the template used to create the test pass.
7. On the **File** menu, click **Save Test Pass**.

Adding a Test Case to a Test Pass

You can add a test case directly to a test pass without adding the test case to a template.

To add a test case to a test pass

1. In Windows Embedded Compact Test Kit, connect to a device. For more information, see Connecting to a Device.
2. On the **File** menu, point to **Open**, and then click **Test Pass**.
3. In the **Open** dialog box, select a test pass file and then click **Open**.

   Note

   If a test pass was already in the **Test pass display** window, you'll be asked if you want to save the current test pass before a new test pass is created.
4. In the **Test pass display** window, on the **View** menu, click **Test Case Explorer**.
5. In the **Text Case Explorer** window, select a test case node or a test case from the **Windows Embedded Test Catalog** tree.
6. Drag and drop the selected test case or test case node onto the **Test Pass** display window. If you select a test case node, the node and all children of the node are copied to the current test pass.
7. To save the test pass, on the **File** menu, click **Save Test Pass**.

Running a Test Pass

There are several options for running a test from the currently active test pass in the **Test pass display** window.

- You can run all of the tests in the test pass, which are run sequentially. This is the default option unless you deselect specific tests.
- Choose which specific tests you want to run. To select the test to be run, select the check box next to the individual test case or test case node.
- Select and run only one test or a particular test node.

Running All Tests in a Test Pass

To run a test, you first need to create a new test pass or open an existing test pass. For information about creating a new test pass, see Creating a Test Pass.

To run all the tests in a test pass

1. In Windows Embedded Compact Test Kit, connect to a device. For more information, see Connecting to a Device.
2. On the **File** menu, point to **Open**, and then click **Test Pass**.

3. Select a test pass file in the **Open** dialog box, and then click **Open**.
4. In the **Test Pass** display window, on the **Tests** menu, click **Run Test Pass**, or press the **F5** key.

All the tests in the test pass will run, sequentially from the top down.

Running Select Test Cases

You can choose which specific tests you want to run during the test pass run.

To run multiple selected test cases in the test pass

1. In Windows Embedded Compact Test Kit, connect to a device. For more information, see Connecting to a Device.
2. On the **File** menu, point to **Open**, and then click **Test Pass**.
3. In the **Open** dialog box, select a test pass file and then click **Open**.
4. In the **Test Pass** display window, select the checkboxes of the test cases or test case nodes you want to run. Deselect the checkboxes of the test cases or test case nodes that you do not want to run.
5. On the **Test** menu, click **Run Test Pass**, or press the **F5** key.

All the selected test cases in the test pass will run, sequentially from the top down.

To run one specific test in the test pass

1. In Windows Embedded Compact Test Kit, connect to a device. For more information, see Connecting to a Device.
2. On the **File** menu, point to **Open**, and then click **Test Pass**.
3. In the **Open** dialog box, select a test pass file and then click **Open**.
4. In the **Test Pass** display window, select the individual test case or test case node you want to run.
5. Right-click the selection and click **Run Test**.

The selected test or all of the tests in the selected node will run, sequentially from the top down.

Viewing Test Progress

As each test in the test pass is run, the **Test pass display** window shows the status of each test appended to each test name in brackets. For example, if a test named **DirectDraw Performance Test** is running the name is displayed as **DirectDraw Performance Test [In Progress]**. Also, the green status bar at the bottom of the Windows Embedded Compact Test Kit main window displays the progress of the test.

Once a test has completed its run, the results of the tests are displayed in the brackets appended to the test name. For example, **DirectDraw Performance Test [Passed]** or **DirectDraw Performance Test [Failed]**. In addition, the text color of the test name is changed to green if the test passed or red if the test fails.

As the tests are run, the Kernel Independent Transport Layer (KITL) debug message output is displayed in the **Connection Output** window.

Stopping a Test

You can stop a test in progress.

To stop a test

1. On the **Test** menu, click **Stop Test Pass**.

Viewing Test Properties

Once you select a test case in the test pass, you can view the properties of the test case.

To view test properties

1. In Windows Embedded Compact Test Kit, connect to a device. For more information, see Connecting to a Device.
2. On the **File** menu, point to **Open**, and then click **Test Pass**.
3. In the **Open** dialog box, select a test pass file and then click **Open**.
4. On the test pass list, click on the test you want to view the properties for.
5. If the **Properties** window is not visible, on the **View** menu, click **Properties Window**.

   The **Properties Window** displays properties for the selected test.

Changing Command-Line Options

You can change the command-line options that a test case will use. For information on Tux test harness (TUX) command-line parameters, see Tux Command-Line Parameters (http://go.microsoft.com/fwlink/?LinkID=192154&clcid=0x409).

To change the command-line option for a test

1. In Windows Embedded Compact Test Kit, connect to a device. For more information, see Connecting to a Device.
2. On the **File** menu, point to **Open**, and then click **Test Pass**.

3. In the **Open** dialog box, select a test pass file and then click **Open**.
4. On the test pass list, click on the test case you want to change.
5. If the **Properties** window is not visible, on the **View** menu, click **Properties Window**.
6. In the **Properties** window, select the **Current Command Line** text box.
7. In the text box, type or modify the command line text to be used when you run the test.
8. On the **File** menu, click **Save Test Pass**.

Viewing Test Results

After all of the tests have completed running, the Test pass display window shows whether each test passed or failed. Test logs and results are saved for each test.

To view the overall results for a test pass

1. In the Windows Embedded Compact Test Kit toolbar, click on the **View Results for the Selected Test Pass** icon.
2. View the results for each test and a summary of the results for all the tests run in the test pass in the **Common Results View** window tab.
3. In the **Test Pass Results** table, click the folder icon in the **Log File(s)** column.
4. In the Windows Explorer window that opens, open the results.log file to view the test debug output in Tux test harness (TUX) format.

   Note

   The test result logs are stored using the following folder hierarchy:

   `C:\Users\<username>\Compact Test Kit\results\`
5. In the **Test Pass Results** table, double-click a test results row to open a **Common Results View** window tab showing the detailed history of the runs for the test case.

To view the results for a test

1. In the **Test pass display** window, select the test, right-click and then select **View Test Results**.
2. In the **Common Results View** window tab that opens, in the **Test Case Results History** table, in the **Log File(s)** column, click the folder icon.
3. In the **Windows Explorer** window that opens, open the results.log file to view the test debug output in TUX format.

   Note

   The test result logs are stored using the following folder hierarchy:

```
C:\Users\<username>\Compact Test Kit\results\
```

Managing Tests

You can add your own test cases to a test case catalog and create test pass templates using the Test Manager.

To open the Test Manager

1. In Windows Embedded Compact Test Kit, on the **Tests** menu, click **Manage Tests**.
2. In the **Test Manager** window that opens, click the **Test Case Information** tab on the right to add a test case to a test catalog or to edit the properties of an existing text case. For more information, see Adding a Test Case to a Test Catalog.
3. Click the **Test Pass Templates** tab to create a new test pass template or to modify existing templates. For more information, see Creating a Test Pass Template.
4. Click **Done** to exit the Test Manager.

Adding a Test Case to a Test Catalog

You can add your own test cases to a test case catalog using the Test Manager.

To add a test case to a catalog

1. In Windows Embedded Compact Test Kit, on the **File** menu, point to **New**, and then click **Test Case**.
2. In the **Test Manager** window that opens, on the **Test Case Information** tab on the right, click the **New** button.
3. Under the Selected test case information heading, do the following:
   a. In the **Name** box, type a name for your test case.
   b. In the **Catalog** list, select the catalog to add the test to. By default, Windows Embedded Test Catalog is selected.
   c. In the **Category** list, select a category for the test case.
   d. In the **Run type** list, select a type of run to use for the test case: **Fully Automated**, **Partially Automated**, or **Manual**.

      If you select **Automated**, in the **Test harness** list, select the test harness to use for the automated run: **Execution Engine**, **TUX**, or **Tux.net**.

   e. In the **Command line** box, type a new command if you do not want to use the default command.
   f. Under **Supported architectures**, select the check box for each CPU architecture you want your test case to support.

      Each time you select a check box, an **Add/Remove Test Files** dialog box opens. In the **Add/Remove Test Files** dialog box, select the main binary file for the selected architecture and add support files.

4. Click the **Advanced** button, in the **Advanced Test Information** dialog box, and then do the following:
   a. In the **Test type** list, select a test type for your test case.
   b. In the **Run sub-type** list, select a run sub-type.
   c. In the **Required peripherals** list, select required peripherals to use in peripheral detection.
   d. In the **Cancel command line** box, type a command line to use to stop the test.
   e. If the test is automated or semi-automated and run from the CTK UI, in the **List command line** box, type a command line to use for running the test from the Test Pass list.
   f. If the test requires a server to run on the desktop computer, in the **Server command** box, type a server application to run.
   g. (Optional) In the **Server command line** box, type a server command line command.
   h. Select the **Enable connection heartbeat** check box to enable continuous check to see if the test device is connected.
   i. In the **Description** box, type descriptive notes about the test case.
5. Click **OK**.
6. Click **Save** to add the test case to the catalog you selected.

Windows Embedded Test Catalog

The Windows Embedded test catalog is the master list of all of the test cases included in the CTK. Each test area, such as Display or Networking, contains a number of individual tests that can be selected and run. The current Windows Embedded test catalog includes the following tests:

- Accelerometer
- Audio
- Backlight
- Battery
- Bootloader
- Communication Bus
- Display

- File System
- Input
- Kernel
- Multimedia
- Networking
- NLED (Notification light-emitting diode)
- OAL (OEM Abstraction Layer)
- Power Manager
- Security
- Shell
- Smart Card
- Storage Media
- USB

Detecting Peripherals and Packages

In the Windows Embedded Compact Test Kit (CTK) you can enable or disable peripheral detection. If peripheral detection is enabled, when connecting to a device the CTK checks the device for peripherals that each test case requires.

In the **Test pass display** window a "not detected" icon (a red circle with a line through it) is displayed on each test case folder where the tests require peripherals that the attached device does not have. For more information, see Test Pass Icons.

The peripherals are specified in **Required peripherals** for a test case when using the **Test Manager** to add a new or edit an existing test case. The required peripherals for a test case can be viewed in the **Needed Peripherals** property. For more information, see Adding a Test Case to a Test Catalog.

To enable peripheral detection

1. In Windows Embedded Compact Test Kit, on the **Connection** menu, click **Peripheral Detection**.

Package detection is the same as peripheral detection, except it checks for required software packages instead of peripherals. Package detection is not currently used much in the test cases in the Windows Embedded Test Catalog. The required packages for a test case can be viewed in the **Needed Packages** property.

Using the Graph Tool

You can use the Graph Tool to graphically display the results of key performance tests such as the Winsock, USB, and Bluetooth performance tests. The Graph Tool can be run from within the Windows Embedded Compact Test Kit (CTK) or from the console command line. You can use the console version of the Graph Tool if you are running the supported performance test through Platform Builder and not the Windows Embedded Compact Test Kit (CTK). Use this method to process the perf results log and generate the graph. For more information on the console option, see Appendix B - Running the Graph Tool as a Console Application.

The Graph Tool can be run from within the Windows Embedded CTK in two different ways: integrated as part of viewing the results for a test or run directly from the **Tools** menu.

Viewing Performance Test Results Graphically

The Graph Tool can be run from within the Windows Embedded CTK as part of the process of viewing test results for the Winsock, USB, and Bluetooth performance tests.

To view performance test results graphically

1. In the Windows Embedded CTK, connect to a device. For more information, see Connecting to a Device.
2. Open a performance test, such as the Winsock Performance Test. For more information, see Appendix C - Running the Winsock Performance Test.
3. On the **File** menu, point to **Open**, and then click **Test Pass**.
4. In the **Open** dialog box, select a test pass file and then click **Open**.

   Note

   If a test pass was already in the **Test Pass** display window, you'll be asked if you want to save the current test pass before a new test pass is created.
5. In the **Test Pass** display window, select the checkboxes of the test cases or test case nodes you want to run.
6. On the **Test** menu, click **Run Test Pass**, or press the **F5** key.
7. Once the test pass has completed running, click on the **View Results for the Selected Test Pass** icon in the toolbar or right-click on the individual test and click **View Results**.
8. On the **Common Results View** window tab, in the **Test Pass Results** table, double-click the test results row for the successfully complete performance test.
9. On the **Graphical Results View** window tab, in the **Test Case Results History** table, in the **Graph View** column, click the document icon.
10. In the window that opens, view the test results data in graphical format.

Running the Graph Tool from the Tools Menu

In addition to the Graph Tool running within the Windows Embedded CTK as part of the process of viewing test results, the Graph Tool can be run directly from the **Tools** menu.

To run the Graph Tool from the Tools menu

1. In Windows Embedded Compact Test Kit, on the **Tools** menu, click **Graphical Results View**.
2. In the **Graphical Performance View** dialog box, on the **Simple** tab, do the following:
   a. In the **Test Dll Name** list, select the configuration file for the type of performance test results to graph. For example, if you are graphing the Winsock Performance Test results, select **perf_winsock2**.
   b. Click the **Open Notebook** icon to view the selected configuration file.
   c. Next to **Input File**, click the **Open Folder** icon.
   d. In the **Add File** dialog box, select the input log file to use for graphing data.
   e. Change the folder path in **Output** or keep the default path.
3. (Optional) Click the **Advanced** tab to add the following additional features to your graph:

To add this optional feature	Do the following
To graph how performance varies over time	a. Select **Trend Analysis** to graph multiple logs. b. Next to **Input File(s)** click the open folder icon. c. In the **Add File** dialog box, select an input log file to use. d. Click **Add more logs** to add more input logs for the trend analysis. You can use a maximum of 10 input log files.
To graph multiple logs using one test log as a benchmark	a. Next to **Input File(s)** click the open folder icon. b. In the **Add file** dialog box, select an input log file to use. c. Click **Add more logs** to add more input logs. d. Next to **Benchmark File** click the open folder icon. e. In the **Add file** dialog box, select an

	input log file to use as a benchmark.
To run Microsoft Excel once the graphs are drawn	Select **Launch Excel**. This option is selected by default.
To output graphs in an MIME HTML (MHTML) web page archive format	Select **mht**.
To output graphs as JPEG image files	Select **Images**.
To select a line, plotted point, or bar graph	Select **Graph Type**.
To set the font size	Select a **Font Size**.

4. Do one of the following:
   - Click the **Run** to generate graphs.
   - Click **Reset** to reset to the default setting.
   - Click **Close** to exit.

Test Automation with the CTK Automation Tool Solution (CATS)

You can use the Windows Embedded Compact Automation Tool Solution (CATS) tool to automate a Windows Embedded Compact Test Kit (CTK) test pass. CATS provides the framework to automate a broad range of test scenarios that have historically been limited to being run manually.

Features

When you use the CATS tool to automate a test pass, the following features are available:

- A wizard that you can use to create the configuration files used by a CATS test pass.
- Automated execution of CTK tests, from start to finish.
- Compatibility with test scripts (.pvt files) created manually without using the CATS Wizard.
- No system-defined limit to the number of individual tests within a CATS test pass.
- Debugging capabilities for test scripts.
- XML log files of CATS test pass results.
- Extensibility for executing custom code that resets hardware devices. For detailed instructions, see Appendix D - Creating a Power Controller Automation DLL.

Note

Microsoft Platform Builder for Windows Embedded Compact 7 deploys new system images to a device when it receives a **BOOTME** message from the device. Devices send a **BOOTME** message at the beginning of a power cycle. For CATS to deploy a new system image to a device it requires a mechanism to trigger a device power cycle without user input. That mechanism is described in the section Appendix D - Creating a Power Controller Automation DLL.

To conduct a CATS test pass, you must first prepare the test, and then run it. Both stages generate important output files.

Test Preparation and Execution

The CATS tool, installed by default with the CTK, includes a CATS Wizard. You use the CATS Wizard to enter details of the test you want to run. The wizard generates the configuration files and scripts that the CATS tool uses to execute the test.

For more information on using the CATS Wizard, see Using the CATS Wizard to Define the Automated Test Pass Parameters

The test pass executes based on the scripts and test configuration files that the CATS Wizard generates. The CATS tool uses your Platform Builder device connection to execute the tests on a remote Windows Embedded Compact device. Commercially available power reset devices can be used as part of the test preparation setup to automatically reset a target device.

For more information on running the test pass, see Running the CATS Test Pass and for more information on viewing the test results, see Viewing the CATS Test Results.

Using the CATS Wizard to Define the Automated Test Pass Parameters

The Windows Embedded Compact Test Kit (CTK) Automation Tool Solution (CATS) Wizard generates the files used to define an automated test pass. The automated test pass uses the following procedures to create the files needed to run a test pass:

1. Creation of a script file that contains the user-selected tests to run.
2. Creation of configuration files that contain device and reset information.
3. Creation of a batch file to start the CATS test pass execution.

You must be able to connect to your device from a developmental computer to complete the following procedure. You must complete the following procedure before the CATS Wizard can generate the necessary output files to launch your CATS automated test pass.

The CATS tool relies on a hardware reset device that can be automated and that you must purchase separately. Because the CATS tool cannot detect the device you are using, you are responsible for creating the library (DLL) that the CATS tool will use to communicate with your power controller hardware to initiate a power cycle on your device. For instructions on how to create this DLL, see Appendix D - Creating a Power Controller Automation DLL

Defining the automated test pass parameters

1. On the development computer, on the **Start** menu, point to **All Programs**, and then click **WindowsEmbeddedCompact7TestKit**. For information on using the CATS tool without the CTK, see Appendix E - Using CATS without the CTK.
2. On the **Tools** menu, click **CATS Wizard**.
3. On the **Inputs** tab, under the **Select source for tests to be added to Automation script file (pvt file)** section, select one of the following:

Source	Description
None (creates an empty script)	Only the basic steps are written into the script file (.pvt) and the created script does not contains any tests.
All tests in CTK catalog (.tcg file)	Uses all automated tests within the CTK test catalog (.tcg) as a source.
Tests from a CTK pass (.tps file)	Uses all of the automated tests present in the test pass file (.tps).

   For this example we will select the item **Tests from a CTK test pass (.tps file)**.
4. In the **Select source for tests to be added to Automation script file (pvt file)** text box, enter the full name and path of a test pass file (.tps).

   For information about creating a test template, see Creating a Test Pass Template.
5. Under the **Enter target device details** section, in the **Device Name:** box, enter the name of the device. Use the name that the device broadcasts (BOOTME broadcast message) when negotiating an image download in Platform Builder. Disconnect any existing CTK or Platform Builder connection to the device.
6. In the **CPU:** list, select your CPU architecture.

   If your device is an image running on a virtual PC, select **x86**.
7. In the **Image Path:** box, enter the full name and path of an image file.

   By default, this file is named nk.bin and located in the build output folder of your operating system design project.

   When the CATS tool runs your tests, it automatically loads this image onto your device.
8. If you have already configured a hardware reset device, in the **Reset Device** drop-down list, select your previously configured reset device.

If you have not previously configured a hardware reset device, you can add a reset device from the **Configuration** tab when you provide the connection DLL location for your particular device.

From the **Configuration** tab, follow the steps below to provide the location of a connection DLL that will be used to reset your device in preparation for the test pass.

a. In the **CATS Wizard**, click **Configuration tab**.

b. Under the **Please enter the details to configure a Reset Device** section, in the **Reset Device Name:** box, enter a name for your reset device.

c. In the **Connection Dll:** box, enter the location where your connection DLL is located.

d. Click **Add New**.

e. In the confirmation box that appears, click **OK**.

Note

You can edit your reset device entry by clicking **Update** or delete your reset device entry by clicking **Delete**.

9. In the **Automation run name:** box, type a name for your automated test pass.
10. In the **Configuration files location:** box, enter the location where you want the CATS Wizard to place the configuration files it generates.
11. In the **Test pass results location:** box, enter the location where you want the CATS wizard to place results for the completed test pass.
12. Click **Create**.

    The CATS Wizard generates the files required to run the tests and saves them to the location specified in the **Configuration files location:**. You can click **View Files** on the presented message box to view the configuration files or browse to the **Configuration files location:** manually.

Running the CATS Test Pass

The CATS tool creates a batch file (CATSExecution.bat) that is used to launch the CATS automated test pass. The batch file is saved to the **Configuration files location:** folder that you previously specified in Using the CATS Wizard to Define the Automated Test Pass Parameters.

To run the Windows Embedded Compact Test Kit (CTK) Automation Tool Solution (CATS) test pass

1. From the batch file location, click the CATS produced batch file and the test pass will begin. For more information about the files that are created by the CATS tool, see Appendix F - CATS File Reference.

   Note

   If you did not provide a location to save the configuration files to, the files are saved to the default location in the C:\Users\<username>\Compact Test Kit\CATS\CATS Configuration folder.

2. When the test pass has completed, a message displays to the screen stating that CATS test pass has completed. Click **OK** to exit the pass.

For information about how to view the test results, see Viewing the CATS Test Results.

Viewing the CATS Test Results

After you run a test pass, a collection of test result files are created and saved to the folder location you previously specified in the **Test pass results location:** box in the **CATS Wizard**.

Note

If you did not provide a location to save the test pass result files to, the files are saved to the default folder location in the C:\Users\<username>\Compact Test Kit\CATS\CATSResults

The results collection contains an XML file and text documents that list and report the test traces and results for the test pass. You can view the test results in the results collection by opening the files with a text editor or XML viewer of your choice.

Stress Testing with the Compact Stress Tool

You can use the Compact Stress tool, which is installed by default with the Windows Embedded Compact Test Kit (CTK), to stress-test your device. The Compact Stress tool is designed to help identify bugs that may exist in your system image. The Compact Stress tool includes a Compact Stress Configuration Editor that you use to create test-case input for the Compact Stress tool.

To complete a stress-test run, use the following procedure.

1. In the Compact Stress Configuration Editor, select the test modules to add to your stress-test mix. A stress-test mix is a collection of test modules that are run during the stress-test. For more information, see Using the Configuration Editor to Define Test Parameters.
2. Run the Compact Stress tool using your created stress-test mix as input. For more information, see Running the Compact Stress Tool.
3. View the results. For more information, see Viewing the Compact Stress Tool Results.

Features

- Allows you to run created stress-mix configurations.
- Allows you to create and edit stress-mix configurations.
- Includes a stress-mix configuration editing tool.
- Provides modular-level results.
- Provides device-side log entries.

- Allows you to diagnose device and output xml files.

Using the Configuration Editor to Define Test Parameters

The Compact Stress tool uses a stress-mix file (also called a configuration file) as input when it executes a test pass. You use the Compact Stress Configuration Editor, which is installed by default with the Windows Embedded Compact Test Kit (CTK), to create or edit this stress-mix file by entering details of the stress testing you wish to perform. The Configuration Editor gives you access to module templates and module types with which you can customize your test pass.

Creating the Configuration File

To create the stress-mix configuration file

1. In the Windows Embedded Compact Test Kit (CTK), from the **Start Page** and under **Getting Started**, click **Connect to Device**.

   Note

   Before you connect to a device from the Windows Embedded Compact Test Kit (CTK), the device must be running and must be attached to Platform Builder. For more information, see Connecting to a Device.
2. On the **Select a Device** screen, choose your device, and click **OK**.
3. On the **Tools** menu, click **Compact Stress Configuration Editor**.
4. In the **Compact Stress Configuration Editor** window, select **File**, and then select **New Mix**.
5. Under the **Configuration** folder and then the **Modules** folder, add existing modules to an existing module template grouping by right-clicking the module template and then selecting **Add Existing Module**.
6. In the **Choose Module Type** dialog box, select the test module type that you want to add to your selected module template, and then click **OK**.

   The following table describes the module types in the **Choose Module Type** dialog box that are available to add to your chosen template.

Module type	Description
Bulk	Standard stress module. Bulk modules are run concurrently to stress-test the device.
Checkpoint	Pauses the run and executes at scheduled intervals to validate stability.
Resident	Runs concurrently with the bulk module without exiting.

Initiator	Started at the beginning of a test run to initialize the environment.
Finisher	Started at the end of a test run to clean up the environment.

7. In the **Add Existing Module** dialog box, select a module, and then click **OK**.
8. To save your new Compact Stress configuration file, click the **File** menu in the Configuration Editor, select **Save**, and then save the file to a location of your choice.

Setup Options

The **Setup** folder in the Configuration Editor allows you to individually specify the options that are available for a particular module type.

You can make user-defined changes to the module types shown in the following table.

Module type	**Option**	**Description**
Bulk Manifest	Concurrency	Number of bulk modules to run concurrently.
	Module Duration	Run duration for each bulk module.
	Module Duration Unit	Unit of time used for run duration (for example, hours, minutes, or seconds).
	Module Weights	Weight distribution for modules in the run.
Chrono Manifest	Checkpoints	Include checkpoints in the run.
	Hang Duration	Duration of time after which a module is considered to have stopped responding.
	Hang Duration Unit	Unit of time used for hang duration (for example, hours, minutes, or seconds).
Health Monitoring	Break To Debugger	Determines whether to break to the debugger when health

Module type	Option	Description
		assessment fails.
	Monitored Rates	List of monitored rates and their associated thresholds.
	Monitored Threads	List of monitored thread priorities and their associated starvation thresholds.
	Health Interval	Health interval in minutes. Determines how often pass rates and storage capacity are assessed.
	Log Frequency	Number of health intervals between each logged health report. A value of 0 means that only errors are reported.
	Enable Results Logging	Determines whether to store results data to a file.
	Results Logging Path	If enabled, determines the path to save the results log file to. The $(relfsd_root) variable can be used to reference the virtual release directory.
	Minimum Ram Kb	Minimum allowed RAM amount in kilobytes.
	Minimum Storage Kb	Minimum allowed storage amount in kilobytes.
Logging Manifest	Debugger Verbosity	Filtering level to use for logger debug output.
	Desktop Logging Enabled	Specify whether to push device-side module output up to the desktop-side UI.
	Base Name	Base name to use when generating log files.
	Unsupported Features Used	Determine whether the loaded configuration file uses

Module type	Option	Description
		unsupported logging features.
Termination Manifest	Bulk Executions	Minimum number of bulk executions.
	Checkpoint Executions	Minimum number of checkpoint executions.
	Overall Iterations	Minimum number of overall test iterations.
	Runtime	Minimum runtime.
	Runtime Unit	Runtime unit of time (for example, hours, minutes, or seconds).
	Criteria	Determines whether the run should stop after any condition is met, or only after all conditions are met.

To specify options for module types

1. In the Compact Stress Configuration Editor, click the **Setup** folder, and then select a module type.
2. The available user-defined options appear in the viewing pane on the right. Highlight the option that you would like to change.
3. Edit the available value to a value of your choice.
4. When you have completed your changes, click the **File** menu, select **Save**, and then save the changed configuration file to a location of your choice.

Adding a New Module

To add a user-defined test module from the Compact Stress Configuration Editor

1. On the Compact Stress Configuration Editor **Mix** menu, click **Add New Module**.

   For more information about the Compact Stress Configuration Editor, see Using the Configuration Editor to Define Test Parameters.
2. In the **Create New Module Template** dialog box, select one of the following three options for

creating your new template, and then click **OK**.

- **Create a completely new module definition** is used to provide an empty module definition in which you create your customized module.
- **Using an existing template as a starting point** uses an existing module as a base for configuring your user-defined module.
- **Overriding a master template locally** allows you to override master template settings with local preferences.

3. For your choice, provide the necessary module information for the required fields, and then save your selection.
4. Your newly created module is now available when you add an existing module from within the Compact Stress Configuration Editor.

Running the Compact Stress Tool

In the Compact Stress tool, you use your configuration file as input to define the test modules that your stress mix will use during the test pass. You start the test pass as described in the following procedure.

To run the Compact Stress test pass

1. In the Windows Embedded Compact Test Kit (CTK), connect to a device if one is not already connected. For more information, see Connecting to a Device.
2. On the **Tools** menu, click **Compact Stress**.
3. In the **Options** dialog box, select your connected device, and then click **OK**.
4. In the **Remote Tools Shell** window, highlight the **Launch Pad** option.

   Other device and test information is available from the **Remote Tools Shell** window. The following table describes the information.

Section	**Description**
Device Log	Displays device-side entries if they were configured in the selected mix.
Run Monitor	Displays the Compact Stress status messages and module results.
Diagnostics	Reports device status and found issues in the **Report Details** window. You can also run diagnostics by selecting the **RTFx Communications** or **CeDebugX7.0**

	Diagnoser option from the **Provider** column.
Launch Pad	Provides access to the stress-mix files and the options for running a stress-mix test.

5. In the **Select the Compact Stress Mix to launch** dialog box, browse to the mix that you want to start, and then click the **Start** button. The test run begins.

   Note

   You also have the option of starting the mix at a specified time. To do this, select the **Start the run automatically (mix required)** option, and enter a start time in the provided window. The run will start at the time that you provide.
6. You can follow the status of the stress test by viewing the status messages that appear in the **Compact Stress Messages** window.
7. To stop the run at any time, click the **Stop** button. To restart the run from the beginning of your specified mix, click the **Start** button.

   Note

   There is no pause functionality. If you stop the run, clicking the **Start** button restarts it from the beginning.

Viewing the Compact Stress Tool Results

Throughout the testing process, you can view the Compact Stress tool test results in real time by using the **Output File** button, which is enabled when the stress test begins and remains enabled throughout the test.

To view the Compact Stress results

1. In the **Remote Tools Shell** window, click the **Output File** button while a test is running.

   A window, based on your default XML viewer, appears and presents a preview of the test output .xml file, which is labeled CompactStressresults_<*DeviceName*>_<*Test Run Date and Time*>.xml. This preview of the test results is a preview of the .xml output file that, when the test is finished, will contain the complete details of the test.

   Note

   The output file is constantly updated until the test pass ends. Thus, to be sure you are viewing the complete output, you should only open the actual output file after the test run has completed.
2. When the test run has completed, you can view the results by opening the output file

CompactStressresults_<*DeviceName*>_<*Test Run Date and Time*>.xml using the XML viewer of your choice. Typically, the completed output file can be found at:

C:\Users\<*UserName*>\Documents\CompactStress.

Appendixes

The appendixes contain additional information about the Windows Embedded Compact Test Kit (CTK). The following sections provide instructions for running test programs using a command-line, running a Winsock performance test, creating a power controller automation DLL, using CATS without the CTK, and file references.

- Appendix A - How to Run CTK Tests Manually Through Platform Builder
- Appendix B - Running the Graph Tool as a Console Application
- Appendix C - Running the Winsock Performance Test
- Appendix D - Creating a Power Controller Automation DLL
- Appendix E - Using CATS without the CTK
- Appendix F - CATS File Reference

Appendix A - How to Run CTK Tests Manually Through Platform Builder

The Windows Embedded Compact Test Kit (CTK) can be run using Platform Builder.

To run the Windows Embedded CTK tests manually using the command line in Platform Builder

1. To use necessary test binary files, do one of the following:
   - Copy the specific test binary files that you need to run a particular test to your OS release directory.

   a. From `%CTK_InstallRoot%\Windows Embedded Compact Test Kit\tests\target\<CPU>`, copy the Tux test harness (TUX) test DLL and any dependent binaries to your working release directory.

   b. From `%CTK_InstallRoot%\Windows Embedded Compact Test Kit\harnesses\target\<CPU>`, copy the binary files for the TUX harness.

   - In Platform Builder set the **Alternate Release Directories** to point to the directories that contain test binary files you need to run a particular test.

   a. In Platform Builder on the **Target** menu, click **Alternate Release Directories**.

   b. In **Alternate Release Directories** window, click the **Add New Directory** folder icon.

c. In the **Browse for Folder** dialog box, select the folders listed below and then click **OK**.

- `%CTK_InstallRoot%\Windows Embedded Compact Test Kit\tests\target\<CPU>`
- `%CTK_InstallRoot%\Windows Embedded Compact Test Kit\harnesses\target\<CPU>`

2. Copy the default command lines for the tests from one of the locations below:
   - The test kit help documentation TestKit.chm located at `%CTK_InstallRoot%\Program Files\Windows Embedded Compact Test Kit\Help\`.
   - The CTK **Properties** window.
3. In Platform Builder on the **Target** menu, click **Target Control**.
4. In the **Windows CE** command prompt window, paste the command line.
5. Run the test and view debug output in the Platform Builder's **Output** window.

Appendix B - Running the Graph Tool as a Console Application

In addition to running the Graph Tool from within the Windows Embedded Compact Test Kit (CTK), you can run the Graph Tool as a console application.

To run the Graph Tool from the console command line

1. Open a command prompt window.
2. Navigate to the folder where the Graph Tool is installed. The default location is:
   `C:\Program Files\Windows Embedded Compact Test Kit\GraphTool\bin\`

   Note

   By default the Graph Tool configuration files are located in:
   `C:\Program Files\Windows Embedded Compact Test Kit\GraphTool\ConfigFiles\`
3. Type the following command, replacing placeholder text in with your specific test information:

```
GraphToolConsole.exe -c <config_file> -i <input_log_file> -t
<test_dll_name>
```

Graph Tool Command-Line Parameters

This section describes the usage syntax and command-line parameters for the Graph Tool console application.

```
GraphToolConsole -c <Graph Tool Config File>  -i <Input File> -t <Test name>
```

```
    [-o <Output File(xlsx)>] [-b <Benchmark file>][-trend][-image <path>]
    [-ct <Chart Type>][-XlLaunch][-mht <Output File(mht)>][-font <Font Type>]

  -c  <Graph Tool Config File> : (Input) Path to the graph tool configuration
file(xml)
  -t  <Test name>              : (Input) Test DLL name from used tp generate logs
  -i  <Input file>             : (Input) Input log file(s)
  [-o <Output excel file>]     : (Output/Optional) Path to the output graph
file(.xlsx)
                                 (By default saves as Graphs.xlsx in base directory)
  [-b <Benchmark file>]        : (Input/Optional) Benchmark file
  [-trend]                     : (Input/Optional)  Enables trend analysis
  [-image <Path>]              : (Output/Optional) Saves graphs as JPEG images at
specified path
                                 (By default images are saved in base directory)
  [-mht  <Output mht file>]    : (Output/Optional) Saves graphs as web page(mht) at
specified path
                                 (By default saves as Graphs.mht in base directory)
  [-ct    <Chart Type>]        : (Optional) Chart type to be plotted
                                 1-Smooth curve with markers
                                 2-Smooth curve
  [-font   <font Type>]        : (Optional) Font Type
                                 1-Small font type
                                 2-Medium font type
                                 3-Large font type
  [-XlLaunch]                  : (Optional) Opens the output Microsoft Excel file
```

Note

If path for the input file(s) not specified, the base directory will be treated as the default path.

Examples:

```
GraphToolConsole -c ConfigFile.xml -t perf_winsock2 -i perf.LOG -image
GraphToolConsole -c ConfigFile.xml -o graph.xlsx -i perf.LOG -t perf_winsock2
GraphToolConsole -c ConfigFile.xml -o graph.xlsx -t perf_winsock2 -i perf.LOG
GraphToolConsole -c ConfigFile.xml -i perf.LOG -t perf_winsock2 -image -ct 2 –
XlLaunch
GraphToolConsole -t perf_winsock2 -c ConfigFile.xml -i perf.LOG -image c:\images\ -
ct 2
GraphToolConsole -c ConfigFile.xml -i perf.LOG -mht graphs.mht -t perf_winsock2 -ct
2 -font 1 –XlLaunch
```

Appendix C - Running the Winsock Performance Test

This section guides you through running the Winsock Performance Test. The Graph Tool can graph the results of this test. For more information, see Using the Graph Tool. The following steps guide you through the process of running the Winsock Performance Test.

- Executing the Winsock Performance Test Command in Platform Builder
- Creating a Winsock Performance Test Template
- Creating a Winsock Performance Test Pass
- Running the Winsock Performance Test

Executing the Winsock Performance Test Command in Platform Builder

A device must be running and attached to Platform Builder before you can connect to the device from the Windows Embedded Compact Test Kit (CTK). For more information, see Connecting to a Device.

When running the Winsock Performance Test, you must execute a command in Platform Builder to run the server part of the Winsock Performance Test before starting the Winsock Performance Test in the Windows Embedded CTK.

Before executing the command to run the server part of the Winsock Performance Test, Platform Builder must be attached to the device that you are going to connect to from the CTK to run the Winsock Performance Test.

To execute the Winsock Performance Test Command in Platform Builder

1. In Platform Builder, On the **Target** menu, click **Target Control**.
2. At the **Window CE>** command prompt, type **s perf_winsockd2 –debug** and press **Enter**.

You are now ready to run the Winsock Performance Test in the Windows Embedded CTK.

Creating a Winsock Performance Test Template

Before creating a test pass you must create a template to base the test pass on. For more information, see Creating a Test Pass Template.

To create a Winsock Performance test pass template

1. In the Windows Embedded Compact Test Kit, on the **File** menu, point to **New**, and then click **Test Pass Template**.
2. On the **Test Manager** window, on the **Test Pass Templates** tab on the right, under **Edit Templates**, click the **New** button.
3. In the new template text box that appears, type **Winsock Performance** to rename the template from **New Template**.
4. In the **Test Case Explorer** tree view display area, double-click **Windows Embedded Test Catalog** to expand the catalog tree.
5. Double-click the **Networking** folder.
6. Double-click **Ethernet**.
7. Click **Winsock Performance Test**, and then click the **Add >** button to add the selected test to the new **Winsock Performance** template.
8. Click **Done** to exit **Test Manager**.

You are now ready to create a test pass from the template you just created and saved.

Creating a Winsock Performance Test Pass

Once you have a created the test pass template, you can create a test pass.

You must first be connected to a device before you can create a test pass. Connect to the device that you have attached to Platform Builder running the **Window CE Window CE>s perf_winsockd2 – debug** command as described in Executing the Winsock Performance Test Command in Platform Builder, if you have not already done so.

To create a Winsock Performance test pass from the template

1. On the **View** menu, click **Test Pass Templates View**.
2. In the **Test Case Explorer** window, on the **Test Pass Templates View** tab, right-click the **Winsock Performance** template, and then click **Use as Current Test Pass**.

   Note

   If **Use as Current Test Pass** is not available on the pop-up menu, you most likely are not connected to a device.
3. In the **Test Pass** display window, click the new test pass **Windows Embedded** that is highlighted and type **Winsock Performance Pass** to rename it.

   Note

If a test pass was already in the **Test Pass** display window, you'll be asked if you want to save the current test pass before a new test pass is created.

Running the Winsock Performance Test

After the Winsock Performance Test has been added to a test pass, you can run the test.

To run the Winsock Performance test

1. In the **Test pass display** window, click the plus sign (+) preceding **Winsock Performance Pass** to expand the test pass tree.
2. Click on **Winsock Performance Test**.
3. If the **Properties** window is not visible, on the **View** menu, click **Properties Window**.
4. Select the **Current Command Line** text box.
5. Change **server_ip** to **localhost** and **ip_version** to **4** in the command:
   ```
   -o -d perf_winsock2 -c"-s server_ip -i ip_version -l true"
   ```
6. Right-click on **Winsock Performance Test** and click **Run Test**.
7. In the **Test pass display** window, right-click on **Winsock Performance Test** and click **Start Test**.
8. When the **Winsock Performance Test [In Progress]** changes to **Winsock Performance Test [Passed]**, the test run is complete and you can view the results.

The green status bar at the bottom of the Windows Embedded CTK main window displays the progress of the test.

For information about reviewing the test results, see Using the Graph Tool.

Appendix D - Creating a Power Controller Automation DLL

The Windows Embedded Compact Test Kit (CTK) Automation Tool Solution requires a power controller automation DLL that can automatically reset and power-cycle a device using reset hardware. The power controller automation DLL consists of a dynamically-linked library with functionality that can be used to reset a particular device. A power controller automation DLL is not reliant on any of the CATS code functionality and can be developed independently of the CATS environment.

Power Controller Automation DLL Outline

For an assembly that will be used as a power controller automation DLL, it must have one public method defined with the name `Reset`. The following example shows the method signature.

```
public bool Reset(string);
```

The string passed into the method can hold the path of a configuration file. This configuration file can contain the parameters for the power reset device that is required inside the `Reset` function. If the reset operation performed by your code is successful, you should return `true` from the method. Otherwise, return `false`.

To create a Power Controller Automation DLL using Visual Studio

1. Open Visual Studio.
2. Select **File**, click **New**, and then click **Project**.
3. In the **New Project Window**, select **Other Languages**, and then select **Visual C#** from the left-pane of the window.
4. In **Visual Studio installed templates**, select **Class library**.
5. Give your project a name, save location, and then click **OK**.
6. Implement the public method with the logic for resetting your device.
7. Select **Build**, click **Build Solution** (F7), and then save your power controller automation DLL.

Appendix E - Using CATS without the CTK

You can run the Windows Embedded Compact Test Kit (CTK) Automation Tool Solution (CATS) tool without having to launch the tool from within the CTK. You may prefer to run the CATS tool without the CTK if you do not want the system overhead of launching the CTK or need to run the CATS tool from a command line. After you provide test parameters to the CATS Wizard and then run the CATS Wizard executable (CATSWizard.exe), the CATS tool creates configuration files that allow you to complete a CATS automated test pass in standalone mode. You can manually define an automated test pass by adding the tests that you want to perform to the .pvt script file, one of the configuration files that the CATSWizard.exe creates. The following steps guide you through the process of running the CATS tool without the CTK and editing the .pvt test script file.

- Running the CATS Tool in Standalone Mode
- Editing the PVT Script File
- Running the Test Pass

Running the CATS Tool in Standalone Mode

You can choose to run the CATS tool in standalone mode, without using the Windows Embedded Compact Test Kit (CTK). The CATS tool uses a wizard, CATSWizard.exe, to create the needed configuration files that can be used to run a CATS test pass in standalone mode.

To run CATSWizard.exe in standalone mode

1. On your development computer, browse to the Windows Embedded Compact Test Kit (CTK)

installation folder and locate the CATSWizard.exe file.

By default, CATSWizard.exe is located at C:\ProgramFiles\WindowsEmbeddedCompact7TestKit\tools\TestAutomation\CATSWizard.

2. Double-click CATSWizard.exe.
3. In the CATS Wizard, on the **Inputs** tab, under **Select source for tests to be added to Automation script (pvt file)**, select **None (creates an empty script)**.

   This will create an empty .pvt script file that you can then add your custom test choices to.
4. Complete the rest of the wizard by providing details as specified in Using the CATS Wizard to Define the Automated Test Pass Parameters.
5. Click **Create**.

Editing the PVT Script File

After the CATS Wizard creates the configuration files for the test pass, the CATS tool uses this group of files in conjunction with a batch file (CATSExecution.bat) to perform an automated test pass. For more information on the files that are created, see Appendix F - CATS File Reference. The .pvt file included in this group of files contains the list of individual tests to run during the test pass. You can manually edit this .pvt file to add your custom collection of tests to run during the test pass.

To manually edit the .pvt file

1. Browse to the **configuration files location:** that you specified in the CATS Wizard to view the newly created configuration files.

   If you did not provide a location to save the configuration files to, the files are saved to the default location C:\Users\<*username*>\Compact Test Kit\CATS Configuration folder.
2. In a text editor, such as Notepad, open the .pvt script file.
3. Scroll down in the .pvt file to the following text:

   ```
   //Users can add their own tests here
   ```
4. Add your custom tests to the area below the text.

   The following code example shows an edited .pvt file section where a user has added the CTK Battery API Tests:

   ```
     //Users can add their own tests here


     TestName = "Battery API Tests";
             commandString = "s tux -o -d batapitest -x1001,1003,1004";
             commandString = wcea.StringSub.UpdateString(commandString);
             Console.WriteLine("\r\nRunOnDevice: {0}", commandString);
   ```

```
          if (wcea.RunOnDevice(commandString, 20000) == false)
          {
            Console.WriteLine("ERROR Failed to execute target control
command!\r\n");
            TestPassed = false;
            goto CLEANUP;
          }

          // Wait until tux.exe process has exited
          wcea.WaitForTuxToExit();
```

To see more examples of how to add test content to add to the .pvt file, view an existing .pvt file that was previously created using the CTK. For more information, see Test Automation with the CTK Automation Tool Solution (CATS).

5. Save the edited .pvt file by using the same naming label and location as the original file.

Running the Test Pass

After you complete the CATS Wizard and edit the .pvt test script file, you can run the automated test pass.

To run the automated test pass

1. Browse to the **Configuration files location:** that you specified in the CATS Wizard.

   If you did not provide a location to save the configuration files to, the files are saved to the default location C:\Users\<*username*>\Compact Test Kit\CATS folder.

2. Double-click CATSExecution.bat.

   A command window appears stating that the test pass has begun. When the test pass completes without errors, a message displays stating that CATS test pass has completed.

You can also run the test pass by using command-line parameters. For more information on the CATS command-line reference, see the CATS Command-Line Parameter Reference (http://go.microsoft.com/fwlink/?LinkID=228591) on the Windows Embedded website.

Viewing the test results

You can view the test results in the same way whether you ran the test within the CTK or in standalone mode. For information, see Viewing the CATS Test Results.

Appendix F - CATS File Reference

During the creation and execution of an automated test pass, the Windows Embedded Compact Test Kit (CTK) Automation Tool Solution (CATS) tool uses several types of files: input files, output configuration files, and result files. These files are described in the following tables.

For more information about the input and output files for the CATS tool, see Test Automation with the CTK Automation Tool Solution (CATS).

CATS Tool Input Files

The CATS tool uses the following input files to create the output configuration files used to run an automated test pass.

Input File	**Description**
Test pass (.tps) file	Contains all of the tests in a user-defined test pass, which the CATS tool writes to the .pvt script file. The .pvt file lists only the fully-automated tests to run during the pass. The .tps file does not contain information about whether the test is fully-automated, semi-automated or a manual test. Therefore, the CATS tool refers internally to the CTK test catalog (.tcg) file to discover if the test in the .tps file is fully-automated. For more information on creating a .tps file, see Creating a Test Pass.
Test catalog (.tcg) file	Contains all of the available tests in the CTK catalog. Only the fully-automated tests in the CTK test catalog file (.tcg file) are included in the .pvt script file.

CATS Tool Output Configuration Files

The CATS tool creates the following configuration files used for running an automated test pass.

If you do not provide a location to save the output configuration files to, the files save to the default location C:\Users\<username>\Compact Test Kit\CATS\CATS Configuration.

Output Configuration File	Description
CATS.xml	Configuration .xml file that provides device information. Required by the CATS execution batch file.
CATSExecution.bat	Batch file that the CATS tool creates and uses to start the test pass without debugging the .pvt script file.
CATSExecution_Edit.bat	Batch file that the CATS tool creates and uses to start the test pass while simultaneously debugging the .pvt script file.
CeDevice-BSP.xml	Contains the test device information, such as device name and device CPU type.
Global.xml	Contains details on locations such as the local path of the results folder and the path to the local archive of the test pass.
Testcase.pvt	Script file that contains the list of fully-automated tests to execute during the test pass.
TestStation.xml	Contains the path location of the DLL that resets the test device when needed.

CATS Tool Result Files

The CATS tool creates the following files for reviewing the results of an automated test pass.

If you do not provide a location to save the result files to, the file will save to the default location C:\Users\<username>\Compact Test Kit\CATS\CATSResults.

Result File	Description
<Your_Test_Pass_Name>.xml	XML file that contains the results of the test pass.
pbdebugoutput_trace.txt	Text file that contains the logs created by the Platform Builder debug process during the test pass.

Note

You can ignore the additional empty text files desktopprocessoutput_trace.txt and pbshelloutput_trace.txt that the CATS tool creates in the results folder.

Conclusion

The Windows Embedded Compact Test Kit (CTK) is a new application that provides significant improvements to the user interface and the overall feature set when compared to the previous release known as Windows Embedded CE Test Kit (CETK).

The Windows Embedded CTK is a tool that you can use to test the functionality and performance of device drivers and related hardware for a Windows Embedded Compact device. The Windows Embedded CTK has the following key features:

- Improved user interface that resembles Visual Studio
- Ability to group user-selected test cases into test passes
- Improved test results viewer lets you store and review test results per test pass
- Upon connection to a device, detection of peripherals and drivers needed for tests
- Integration with the new Graph Tool to graph performance test results
- Integration with the new CATS tool to automate test runs
- Integration with the new Compact Stress tool for running stress testing
- Users can add custom Tux test harness (TUX)-based tests to the CTK
- Support for x86, MIPSII, MIPSII_FP, ARMv5, ARMv6, and ARMv7 processors

The Windows Embedded Test Catalog is the master list of all of the test cases in the Windows Embedded CTK, which currently includes the following tests:

- Accelerometer
- Audio
- Backlight
- Battery
- Bootloader
- Cellular
- Communication Bus
- Display
- File System
- Input
- Kernel
- Multimedia

- Networking
- NLED (Notification light-emitting diode)
- OAL (OEM Abstraction Layer)
- Power Manager
- Security
- Shell
- Smart Card
- Storage Media
- USB

Additional Resources

- Windows Embedded website (http://go.microsoft.com/fwlink/?LinkId=183524)