Freescale Semiconductor
Application Note

Document Number: AN4133
Rev. 0, 05/2010

How to Run BSP Unit Test on Windows CE Using i.MX Platforms

by *Multimedia Applications Division*
Freescale Semiconductor, Inc.
Austin, TX

The majority of time in the development process of an embedded platform is invested in testing the stability of the platform. To help this process, Windows Embedded CE 6.0 tool kit has an automated testing kit in the form of Windows Embedded CE 6.0 Test Kit (CETK). This CETK contains all the tools required to run the repetitive tests in a particular order.

This application note includes the following:

- Describes the setup process to use the CETK
- Describes how to use the CETK with the tests provided by Microsoft
- Describes how to use the custom test provided by Freescale, followed by an example of a custom test for the i.MX31 PDK platform

Contents

1 CETK Overview and Setup

This section describes the CETK overview and the CETK set up in detail.

1.1 CETK Overview

The CETK tool consists of a server application that runs on a development workstation and also a client software that runs on each target device.

The core of CETK contains three process:

- Tux Harness
- Kato Logging Engine
- Device Driver Loader

The CETK provided two different versions of Tux, Tux Test Harness and Tux.Net Test Harness. The Tux test harness is a client/server test harness that executes the test modules stored in dynamic-link library (dll). The Tux.Net is a version of the Tux written in C# and is designed to run tests that are written in managed code on Windows CE based devices.

Kato is a logging engine which is designed to provide a single logging interface to all the test applications. This logging engine can route the output of a test to multiple output devices. The process creates Kato logging objects that contain the information of the tests and using this object, the information is transmitted to a local or remote server.

The Device Driver Loader and Tux Extender (DDLX) is a process followed, to place the dll test code in an application or a Device.exe address space. If DDLX is not used, then the dll of the test is placed in the same address space of the Tux harness.

In conclusion, the Tux with the help of DDLX, runs the tests dlls on the Device. The Kato obtains the results from the dll and transmits this information to the server used. This information is typically received by the CETK application, that acts as a remote server.

1.2 CETK Setup

The CETK has multiple ways of connecting to a particular device, but this application note covers only the ones that are used in the i.MX platforms. For more information about the rest of the configuration options, refer to CETK Setup section of the CETK help.

The two types of communication layers in the i.MX platform are the KITL transport for Windows CE and the Microsoft Active Sync.

The device settings must be set prior to using the CETK application for the first time. These device settings configure the communication layer used.

The steps required to configure the device settings are as follows:

1. Within the Test Kit application, go to Connection menu option and select Start Client. This opens a Device Connection window.
2. Click on the setting to open the Windows CE platform manager configuration window.
3. The platform manager configuration window is used to add and configure devices. The default device is already set to use the KITL. The properties settings to use the KITL are Transport: KITL Transport for Windows CE and Startup Server: KITL Bootstrap Server.
4. To support the Active Sync connection, add a new device with an appropriate name such as Active Sync Device.

5. Select the new device and click on properties.
6. On the Device Properties Window, select Transport: Microsoft Active Sync and Startup Server: Microsoft Active Sync.
7. Close all the windows by clicking Ok.
8. Now, while connecting to a device, choose between the Default Device with KITL transport and the new Active Sync device.

2 Examples

This application note provides two different examples of how to use the CETK. One example shows how to run a custom test using CETK and the other shows how to run a Microsoft test using CETK.

2.1 Running Custom Freescale Test

The example describes the Freescale custom PMIC test which is provided with the i.MX31 PDK. To follow the steps, the latest SDK for the platform must be installed. The detailed instructions for the installation process of the SDK is located in the *i.MX31 PDK 1.X Windows Embedded CE 6.0* user guide. This user guide and the latest SDK can be found on the www.freescale.com\imx web site.

2.1.1 Compile Process

The previous step involves building an OS image with the desired configuration. This image should have the KITL connection enabled and the Windows Embedded CE Test Kit component and this is located in the catalog under the Device Drivers section as shown in Figure 1.

Figure 1 shows the catalog items under the device drivers section.

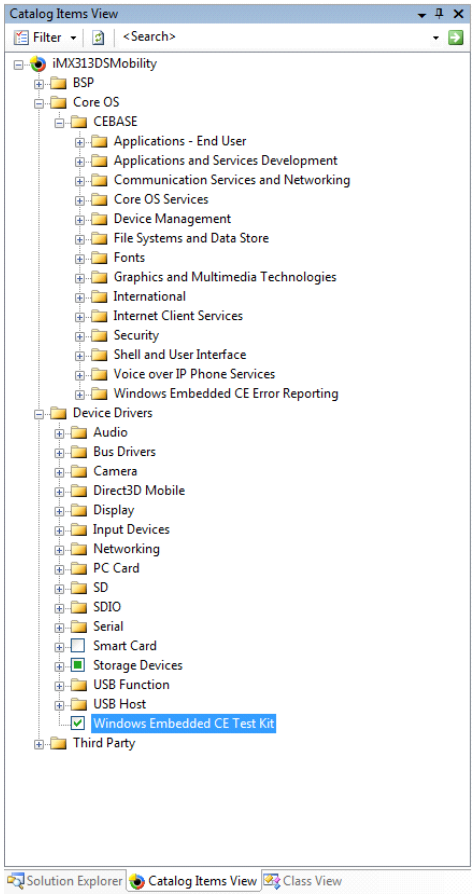

Figure 1. Catalog Items View

The steps required to compile the application are as follows:

1. Within the Platform Builder, go to Build OS menu option and select Open Release Directory menu option to open a DOS prompt.
2. Change to PMIC `TESTS` directory. (`\WINCE600\SUPPORT\MX31\TESTS\PMIC`)
3. Enter `set WINCEREL=1` on the command prompt and hit return. This copies the built DLL to the flat release directory.
4. Enter the build command - `build -c` at the command prompt and hit return. The output result should be the same as the one shown in Figure 2.

After the build is complete, the `pmictest.dll` file is located in the release directory.

Figure 2 shows the compilation of the application in the DOS prompt.

```
_WINCEDRIVE=C:
_PROJPUBLICROOT=C:\WINCE600\OSDesigns\iMX313DSMobility\WINCE600\PUBLIC
Wince ARMV4I iMX313DSMobility iMX313DS Development Environment for Enrique

C:\WINCE600\OSDesigns\iMX313DSMobility\RelDir\Freescale_i_MX31_3DS_ARMV4I_Release>cd C:\

C:\>cd WINCE600\SUPPORT\MX31\TESTS\PMIC

C:\WINCE600\SUPPORT\MX31\TESTS\PMIC>set WINCEREL=1

C:\WINCE600\SUPPORT\MX31\TESTS\PMIC>build -c
Build for Windows CE (Release 601) (Built on Aug 17 2006 15:18:52)
File names: Build.log Build.wrn Build.err Build.dat
BUILD: [Thrd:Sequence:Type   ] Message
BUILD: [00:0000000000:PROGC ] Build started with parameters: -c
BUILD: [00:0000000001:PROGC ] Build started in directory: C:\WINCE600\SUPPORT\MX31\TESTS\PMIC
BUILD: [00:0000000002:PROGC ] Checking for C:\WINCE600\sdk\bin\i386\srccheck.exe.
BUILD: [00:0000000003:PROGC ] Running passes WCEFILES0, MIDL, MC, ASN, THUNK, PRECOMPHEADER, COMPILE, LIB, LINK, MANAGED
RESX, MANAGEDMOD, MANAGEDDLL, MANAGEDEXE, MANAGEDWIN for ARM.
BUILD: [00:0000000004:PROGC ] Computing include file dependencies:
BUILD: [00:0000000005:PROGC ] Checking for SDK include directory: C:\WINCE600\sdk\CE\inc.
BUILD: [00:0000000006:PROGC ] Scan C:\WINCE600\SUPPORT\MX31\TESTS\PMIC\
BUILD: [00:0000000007:WARNS ] Directory: C:\WINCE600\platform\common\src\soc\freescale\iMX31ADS\inc\ does not exist.
BUILD: [00:0000000008:WARNS ] Directory: C:\WINCE600\platform\common\src\soc\freescale\mxarm11\inc\ does not exist.
BUILD: [00:0000000009:PROGC ] Saving C:\WINCE600\SUPPORT\MX31\TESTS\PMIC\Build.dat.
BUILD: [00:0000000018:PROGC ] Building COMPILE Pass in C:\WINCE600\SUPPORT\MX31\TESTS\PMIC\ directory.
BUILD: [01:0000000031:PROGC ] Compiling .\pmictest.cpp
BUILD: [01:0000000034:PROGC ] Compiling .\globals.cpp
BUILD: [01:0000000037:PROGC ] Compiling .\test.cpp
BUILD: [01:0000000040:PROGC ] Compiling .\adc_test.cpp
BUILD: [01:0000000043:PROGC ] Compiling .\backlighttest.cpp
BUILD: [01:0000000046:PROGC ] Compiling .\convity_test.cpp
BUILD: [01:0000000049:PROGC ] Compiling .\pmic_batterytest.cpp
BUILD: [00:0000000063:PROGC ] Building LIB Pass in C:\WINCE600\SUPPORT\MX31\TESTS\PMIC\ directory.
BUILD: [01:0000000076:PROGC ] Linking obj\ARMV4I\retail\PMICTEST.lib
BUILD: [00:0000000098:PROGC ] Building LINK Pass in C:\WINCE600\SUPPORT\MX31\TESTS\PMIC\ directory.
BUILD: [01:0000000112:PROGC ] Linking obj\ARMV4I\retail\PMICTEST.dll
BUILD: [00:0000000141:PROGC ] Saving C:\WINCE600\SUPPORT\MX31\TESTS\PMIC\Build.dat.
BUILD: [00:0000000143:PROGC ] Done.
BUILD: [00:0000000144:PROGC ]                          Files  Warnings  Errors
BUILD: [00:0000000145:PROGC ] Midl                         0         0       0
BUILD: [00:0000000146:PROGC ] Message                      0         0       0
BUILD: [00:0000000147:PROGC ] Precomp Header               0         0       0
BUILD: [00:0000000148:PROGC ] Resource                     0         0       0
BUILD: [00:0000000149:PROGC ] MASM                         0         0       0
BUILD: [00:0000000150:PROGC ] SHASM                        0         0       0
BUILD: [00:0000000151:PROGC ] ARMASM                       0         0       0
BUILD: [00:0000000152:PROGC ] MIPSASM                      0         0       0
BUILD: [00:0000000153:PROGC ] C++                          7         0       0
BUILD: [00:0000000154:PROGC ] C                            0         0       0
BUILD: [00:0000000155:PROGC ] Static Libraries             0         0       0
BUILD: [00:0000000156:PROGC ] Exe's                        0         0       0
BUILD: [00:0000000157:PROGC ] Dll's                        2         0       0
BUILD: [00:0000000158:PROGC ] Preprocess deffile           0         0       0
BUILD: [00:0000000159:PROGC ] Resx                         0         0       0
BUILD: [00:0000000160:PROGC ] CSharp Compile               0         0       0
BUILD: [00:0000000161:PROGC ] Other                        0         2       0
BUILD: [00:0000000162:PROGC ]
BUILD: [00:0000000163:PROGC ] Total                        9         2       0
BUILD: [00:0000000164:PROGC ]
BUILD: [00:0000000165:PROGC ] 2 Warnings,  0 Errors
BUILD: [00:0000000166:PROGC ] GetSystemTimes (seconds): Idle: 11    Kernel: 13    User: 3
BUILD: [00:0000000167:PROGC ] Elapsed  time  (seconds): 4

C:\WINCE600\SUPPORT\MX31\TESTS\PMIC>_
```

Figure 2. Compiling the Application at the DOS Prompt

2.1.2 Integration and Running Process

The PMIC test provided by Freescale must be run in kernel mode and to accomplish this, the tux process needs to be run on kernel mode. The kernel mode version of the tux process must be copied, as it is not copied to the release directory by default. The name of the kernel version of the tux dll is `ktux.dll`. This file is located in `\Program Files\Microsoft Platform Builder\6.00\cepb\wcetk\ddtk\armv4i` and the file must be copied to the release directory.

The test can be run in two different ways. One is by using the Target Control command window in the Visual Studio and the second option is to use the CETK application.

2.1.2.1 Use of the Target Control Window to Run Test

To use this option, the tux.exe and kato.dll files must be available in the release directory. These files are not available by default and hence need to be copied from the location `\Program Files\Microsoft Platform Builder\6.00\cepb\wcetk\ddtk\armv4i` to the release directory.

The steps required to run the PMIC test using the Target Control window are as follows:

1. Within the Platform Builder, go to Target menu option and select Target Control menu option. This opens a Windows CE Command Prompt window.
2. Run this command `s tux -o -n -d pmictest.dll` on the command prompt window, as shown in Figure 3.

The test should start and the test results can be viewed in the output panel of the Visual Studio.

Figure 3 shows the running of the command `s tux -o -n -d pmictest.dll` in the command prompt window.

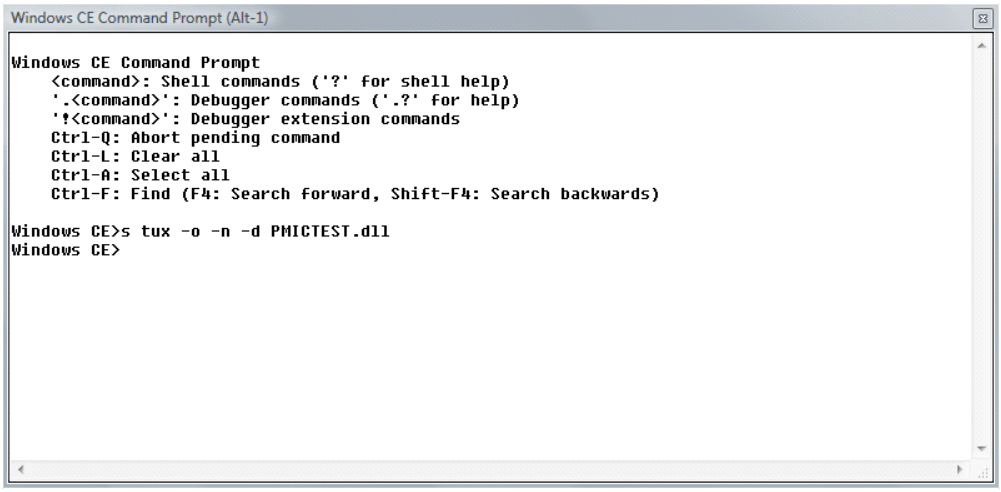

Figure 3. Running the Command s tux -o -n -d pmictest.dll

2.1.2.2 Use of the CETK Application to Run Test

The steps required to run the PMIC test using the CETK application are as follows:

1. A KITL connection needs to be established between the platform and the Visual Studio.
2. Go to windows Start Menu and open the CETK application from the Windows Embedded CE 6 folder.
3. Within the Test Kit application, go to Tests menu option and select User defined... option. This opens a User-defined test wizard.
4. In the Operation window, select Add a New Test, as shown in Figure 4.

Figure 4 shows the User-Defined Test Wizard where ‘Add a New Test’ option is selected.

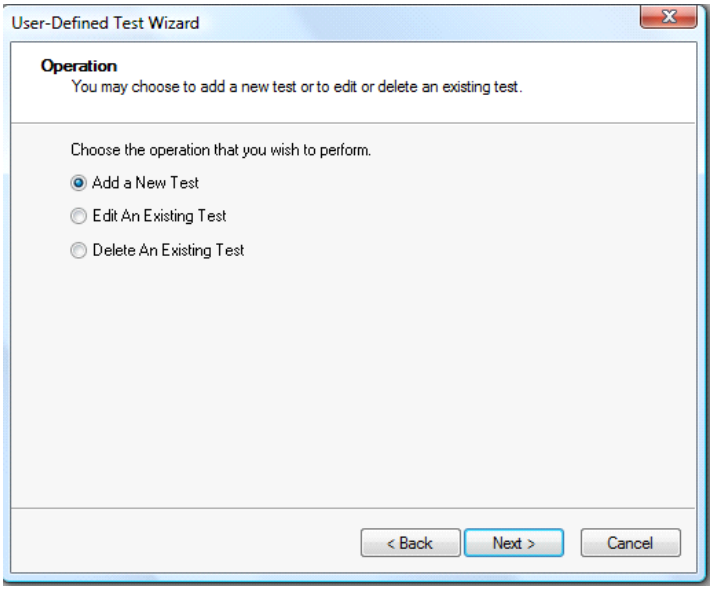

Figure 4. User-defined Test Wizard

5. In the Add window, enter the name of the test, browse for the location where the dll files are stored and select the ARM4I processor, as shown in Figure 5. The wizard sets the command and this command needs to be changed to `add -n` (kernel mode) option in the next step.

Figure 5 shows the adding of a new test in the User-Defined Test Wizard - Add window.

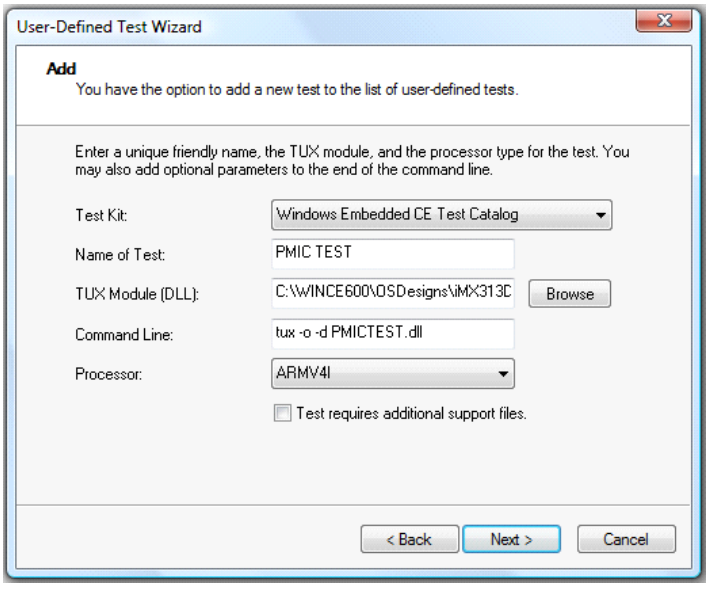

Figure 5. Adding of a New Test

6. Click Finish to complete the wizard with any of the two copy options.
7. Within the Test Kit application, go to Connection menu option and select Start Client. This opens a Device Connection window.
8. Click on Connect... and use the default device. By default, this device uses the KITL connection, and the transport layer is the one used in this example. This step establishes the connection between the device and the CETK application.
9. Within the Windows CE Test tree, there is a User Test folder available and inside this is the new PMIC TEST, as shown in Figure 6.

Figure 6 shows the Windows Embedded CE Test Kit window.

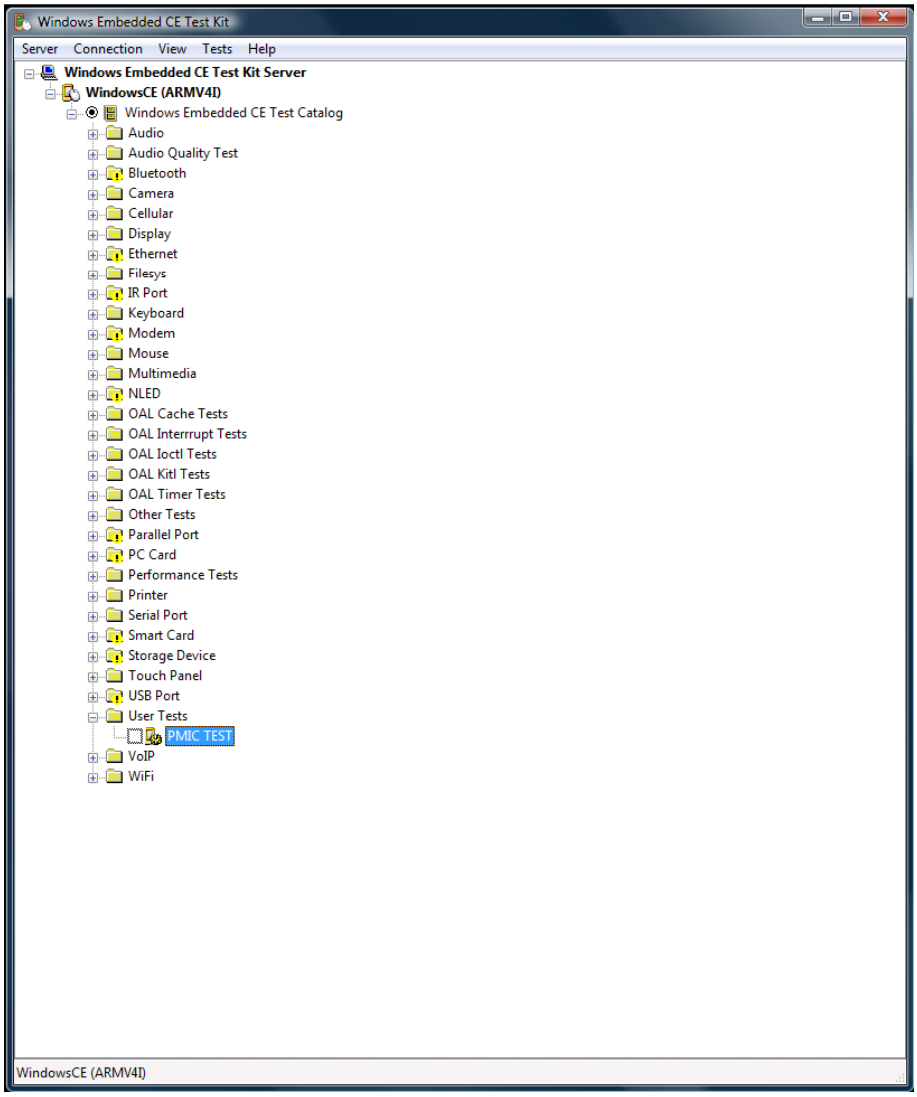

Figure 6. Windows Embedded CE Test Kit Window

10. Right click on the PMIC TEST and select edit command line to enter the correct command - `tux -o -n -d pmictest.dll` and set this change permanently.

Now to run the test, right click on the PMIC TEST and select quick start to start the test. The test results can be viewed in the output window of the Visual Studio.

The test results can also be viewed in the CETK parser. To open the result of this particular test within the Test Kit application, go to Tests menu option and select View Results... option. Then, browse for the PMIC Test results as shown in Figure 7. This step opens the CETK parser with the test results.

Figure 7 shows the steps to open the CETK parser to view the test results.

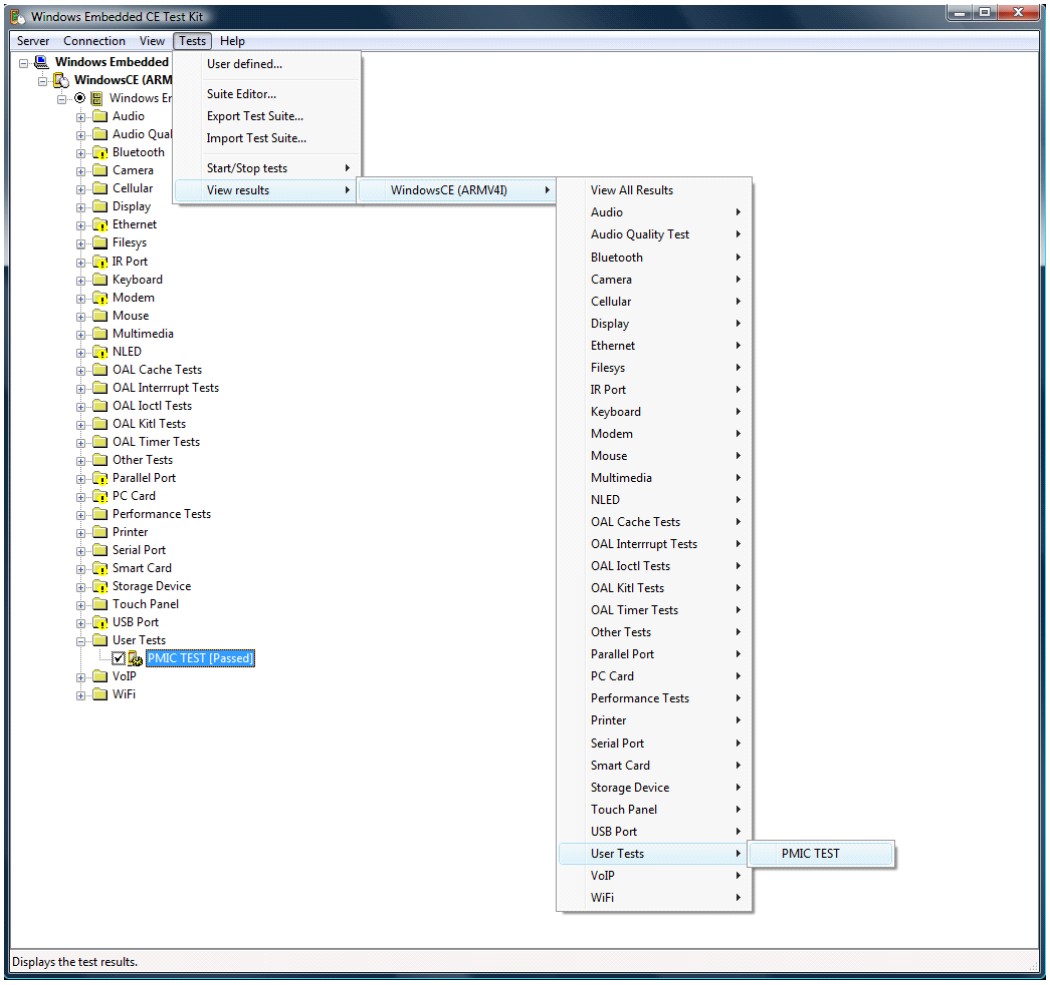

Figure 7. Steps to Open the CETK Parser to View Test Results

This steps example how to run a custom Freescale test using CETK, Freescale provided with their BSPs multiples customs test, for the particular commands needed to compile a run the test consult the Windows Embedded CE 6.0 Reference Manual for the BPS used.

2.2 Running Microsoft Test

The test provided by Microsoft can be run on any platform with a Windows Embedded CE 6.0 BSP. For this example, the platform to use is the i.MX31 PDK. To follow the steps, the latest SDK for the platform must be installed. The detailed instructions for the installation process of the SDK is located in the *i.MX31 PDK 1.X Windows Embedded CE 6.0* user guide. This user guide and the latest SDK can be found on the www.freescale.com\imx web site.

2.2.1 Running Process

The previous step involves building an OS image with the desired configuration. This image should have the KITL connection enabled and the Windows Embedded CE Test Kit component and this is located in the catalog under the Device Drivers section as shown in Figure 8.

Figure 8 shows the Windows Embedded CE Test Kit option enabled under the Device Drivers section.

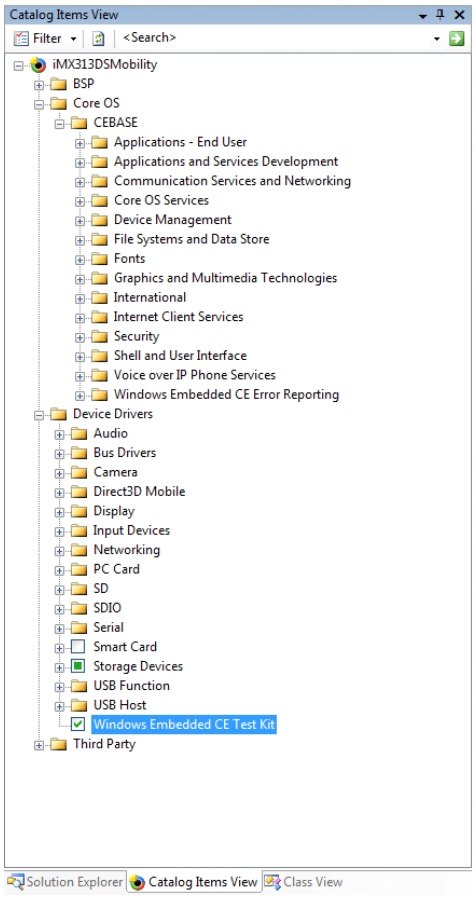

Figure 8. Windows Embedded CE Test Kit Option Enabled

In this example, the Microsoft test that is run is the DirectDraw Test.

The steps required to run this test using the CETK application are as follows:

1. A KITL connection needs to be established between the platform and the Visual Studio.
2. Within the Test Kit application, go to Connection menu option and select Start Client. This opens a Device Connection window.
3. Click on Connect... and use the default device. By default, this device uses the KITL connection, and the transport layer is the one used in this example. This step establishes the connection between the device and the CETK application.
4. Within the Windows CE Test tree, there is a folder named Display, and inside this is the DirectDraw Test, as shown in Figure 9.

Figure 9 shows the DirectDraw Test option under the Display folder.

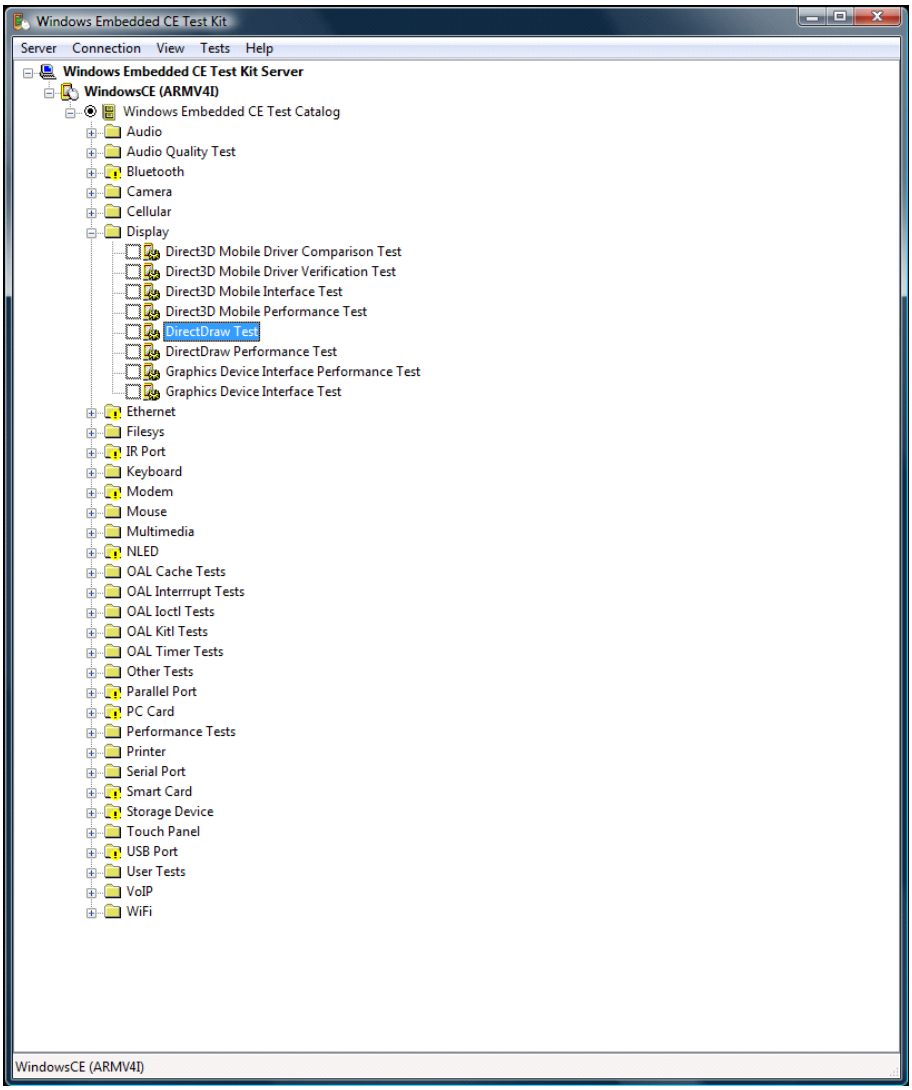

Figure 9. DirectDraw Test Option under Display Folder

Now to run the test, right click on the DirectDraw Test and select quick start to start the test. The test results can be viewed in the output window of the Visual Studio.

The test results can also be viewed in the CETK parser. To open the result of this particular test within the Test Kit application, go to Tests menu option and select View Results... option. Then, browse for the DirectDraw Test results. This step opens the CETK parser with the DirectDraw Test results.

These steps can be used to run most of the Microsoft tests, but there are few tests that need different steps or some previous setup to run. For more detailed information, seek the help of the CETK application.

3 Revision History

Table 1 provides a revision history for this application note.

Table 1. Document Revision History

Rev. Number	Date	Substantive Change(s)
0	05/2010	Initial release

THIS PAGE INTENTIONALLY LEFT BLANK

THIS PAGE INTENTIONALLY LEFT BLANK

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
1-800-521-6274 or
+1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064
Japan
0120 191014 or
+81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
1-800 441-2447 or
+1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Document Number: AN4133
Rev. 0
05/2010