

# **CTC User's Guide**

# Table of Contents

Foreword	0
<b>Part I CTC</b>	<b>7</b>
1 Overview .....	7
2 Tool requirements.....	8
<b>Part II Getting Started</b>	<b>11</b>
1 Setting up for testing.....	11
2 Creating a test package.....	11
3 Creating a test configuration.....	12
4 Running tests.....	12
5 Usage Example.....	13
<b>Part III Basic Concepts</b>	<b>20</b>
1 Startup Wizard.....	20
2 ConfigurationWizard.....	22
3 Auto-Configuration.....	23
4 Test Package.....	23
5 Configuration Template.....	24
6 Configuration Tree.....	25
7 Sequence Tree.....	25
8 Module .....	26
9 Device .....	26
10 Test .....	27
11 Parameter .....	27
12 Parameter Array.....	28
13 Parameter Group.....	28
14 Flow Group.....	28
15 Message Types.....	28
<b>Part IV Document File Types</b>	<b>30</b>
1 Package File.....	30
2 Snippet File.....	30
3 Summary File.....	31
4 Message Redirection Files.....	31
5 XML DTD File.....	31
6 Language Localization.....	34

<b>Part V CTC User Interface</b>	<b>35</b>
<b>1 Title Bar</b>	<b>36</b>
Application Control	36
Status Bar	36
<b>2 Menu Bar</b>	<b>37</b>
<b>File menu</b>	<b>37</b>
New	38
Open	38
Recent	39
Save	40
Save As	40
Exit	40
<b>Edit menu</b>	<b>41</b>
Undo	42
Redo	42
Cut	42
Copy	42
Paste	43
Switch Configuration	43
Clone Configuration	43
Remove Configuration	43
Switch Flow	43
New Flow	44
Clone Flow	44
Remove Flow	44
Node Properties	44
Package Properties	44
<b>Insert menu</b>	<b>45</b>
Add Device	45
Add Test	45
Add Parameter	46
Add Parameter Array	46
Add Parameter Group	46
Add Test Group	46
<b>Action menu</b>	<b>46</b>
Run Flow	47
Stop Tests	47
Clear Results	47
Configure Tests	47
Import Tests	48
ReConfigure Tests	48
ReCalc Tests	50
Run This	51
Log Options	52
<b>Tools menu</b>	<b>53</b>
Options	53
Help Options	53
Font Options	55
Message Header Colors	56
Application Colors	57
License Options	57
<b>Help menu</b>	<b>58</b>

	About .....	59
	Help Topics.....	59
<b>3</b>	<b>Toolbar .....</b>	<b>59</b>
<b>4</b>	<b>Configuration View.....</b>	<b>60</b>
	Import... ..	62
	Configure... ..	62
	Reconfigure... ..	65
	Recalc... ..	67
	Node Help .....	68
	Node Edit Control .....	69
	Icon Definitions .....	69
<b>5</b>	<b>Sequence View.....</b>	<b>71</b>
	Node Help .....	72
	Edit Control .....	72
<b>6</b>	<b>Icon Definitions.....</b>	<b>73</b>
<b>7</b>	<b>Test Message View.....</b>	<b>74</b>
	Maximize Message Window .....	75
	Save Messages Button .....	76
	Search Messages Button .....	77
	Error Message .....	78
	Advisory Message .....	78
	Status Messages .....	78
	Statistic Message .....	78
	Verbose Debug Messages .....	79
	Log Options .....	79
	Summary File.....	81
	Verbose Redirection.....	81
	Status Redirection.....	81
	Advisory Redirection.....	82
	Info Redirection.....	82
	Statistic Redirection.....	82
	Prompt Redirection.....	83
	Error Redirection.....	83
	Archive Files.....	83
<b>8</b>	<b>Property Sheets.....</b>	<b>84</b>
	Configuration Property Sheet .....	84
	Test Property Sheet .....	85
	Device Property Sheet .....	88
	Test Group Property Sheet .....	90
	Parameter Group Property Sheet .....	92
	Parameter Property Sheet .....	93
	Parameter Array Property Sheet .....	93
	<b>Part VI Return Codes .....</b>	<b>95</b>
	<b>Part VII Invocation Parameters .....</b>	<b>95</b>
<b>1</b>	<b>/ARCV .....</b>	<b>96</b>
<b>2</b>	<b>/AUTO .....</b>	<b>96</b>
<b>3</b>	<b>/CFG .....</b>	<b>97</b>
<b>4</b>	<b>/DLL .....</b>	<b>97</b>



5	/FD .....	97
6	/FLOW .....	98
7	/FOREVER .....	98
8	/HE .....	98
9	/LF .....	98
10	/LOOPS .....	98
11	/MAX .....	99
12	/MINUTES .....	99
13	/N .....	99
14	/PASSFILE .....	99
15	/PC .....	99
16	/PKG .....	99
17	/PORT .....	100
18	/Q .....	100
19	/QUIT .....	100
20	/QUITPASS.....	100
21	/RECONFIG.....	100
22	/RMTLOG .....	100
23	/RUN .....	101
24	/SELF .....	101
25	/SUMMARY.....	101
26	/W .....	101
27	/V .....	101
28	Message Redirection.....	102

## Part VIII Reserved Parameters 102

1	\$AUTOCONFIG.....	102
2	\$EXEFILE .....	102
3	\$HELPPFILE.....	103
4	\$TESTNUMBER.....	103
5	\$VERBOSEMASK.....	103

## Part IX Remote Support 104

1	Remote client service.....	104
2	Remote pkx setup.....	111
3	Remote pkx setup for multi nodes.....	114
4	Remote log files.....	116
5	Remote error & exception handling.....	117
6	Remote probing over IP range.....	119

**Index****122**

# 1 CTC

Cross-Platform Test Control (CTC) is a Graphical User Interface (GUI) test executive of the Intel® Modular Test Architecture(MTA).

## Disclaimer and Legal Information

**INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.**

**UNLESS OTHERWISE AGREED IN WRITING BY INTEL, THE INTEL PRODUCTS ARE NOT DESIGNED NOR INTENDED FOR ANY APPLICATION IN WHICH THE FAILURE OF THE INTEL PRODUCT COULD CREATE A SITUATION WHERE PERSONAL INJURY OR DEATH MAY OCCUR.**

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

This document and the software described in it are furnished under license and may only be used or copied in accordance with the terms of the license. The information in this document is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Intel Corporation. Any software source code reprinted in this document is furnished under a software license and may only be used or copied in accordance with the terms of that license. Intel Corporation assumes no responsibility or liability for any errors or inaccuracies that may appear in this document or any software that may be provided in association with this document. Except as permitted by such license, no part of this document may be reproduced, stored in a retrieval system, or transmitted in any form or by any means without the express written consent of Intel Corporation.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or by visiting [Intel's Web Site](#).

Copyright © 2008-2017 Intel Corporation. All rights reserved.

† Third-party names, brands and trademarks are the property of their respective owners.

## 1.1 Overview

### Overview:

Intel® Modular Test Architecture is a software framework for creating and running functional tests.

It includes the following:

- Cross-platform Test Control (CTC) - Graphical User Interface (GUI)

- Test Control (T) - Command Line test control
- Help files
- Executable files: ctc, t

**OS Supported:**

Cross-platform Test Control GUI is cross compilable across multiple OS platforms. Currently CTC is supported on Windows®, Linux™ and Mac OS-X.

**Customized Tests:**

Using this tool, test developers can build a customized test package out of numerous tests and test modules--already designed and verified for a wide range of features and devices in today's PCs.

**Easily extensible:**

By adding new test modules, Cross-platform Test Control - Intel® Modular Test Architecture can be easily updated to accommodate new hardware as it is developed.

**High Coverage Level:**

Test packages created with this tool are able to catch both static and intermittent failures by running tests in parallel and at an accelerated rate, thereby exercising more system operations and stressing more silicon functional blocks simultaneously during testing. Also, testing the OS and drivers destined for the customer provides a higher correlation with the user's experience.

Note: Intel® Modular Test Architecture is used interchangeably with CTC throughout this documentation.

## 1.2 Tool requirements

**License:**

All software components used in the Intel® Modular Test Architecture (MTA) require a license key to operate. The license key is stored in a file, 'license.key', and needs to be located in the same directory as the test executive, executive libraries, and modules.

**Supported operating systems:**

- Windows 2000 32-bit
- Windows XP 32-bit
- Windows Server 2003 SP1 32-bit
- Windows Server 2008 SP2 32-bit
- Windows Server 2008 SP2 x64
- Windows 7 x64
- Windows 8.1 x64
- Windows 10
- Windows Server 2012
- Windows Server 2016
- Red Hat Linux Advanced Server 4.x (includes Updates 1-8), 5.x (includes Updates 1-4), 6.x (all Updates) and 7.x (all Updates)
- SuSE Linux Enterprise Server 9.3, 9.4, 10, 10.1, 10.2, 10.3, 10.4, 11 and 12

- Ubuntu 16
- OS-X

**Windows "vcredist" requirement:** CTC is compiled with Microsoft .NET support. Many of the Microsoft OS versions do not fully support .NET binaries unless a patch, called "vcredist", is installed. There are multiple versions of this patch, one for each architecture: vcredist\_x86.exe, vcredist\_x64.exe, and vcredist\_IA64.exe. This patch, which is included with this CTC package, must be installed before running any CTC components. If CTC was installed with the BitRock(TM) Installer, vcredist is automatically installed. Otherwise, to install manually, simply double-click on the executable name, then follow the instructions.

**Windows "Data Execution Prevention" (DEP) requirement:** Once vcredist has been installed, Data Execution Prevention must be disabled. This must be done manually. Access the System Properties from the Control Panel, and choose the "Advanced" tab. Click on the "Settings" button inside the "Performance" box, then click on the "Data Execution Prevention" tab. It is suggested to choose the selection to "Turn on DEP for essential Windows programs and services only". Alternatively, the option may be chosen to "Turn on DEP for all programs and services except those I select". If the second option is selected, each binary and test module executable must be found and added to the list on this page manually. This method is not suggested, as with each new module addition or name change the tab must be updated, it is easy to miss one or more entries, and the error given when DEP is triggered can be time consuming to debug. NOTE: The order of vcredist and DEP settings are important. If vcredist is installed, it may overwrite current DEP settings - so first install vcredist, then change the DEP setting.

**RHEL 6 Linux requirement:** CTC is compiled with 32 bit Linux libraries. New version RHEL 6 does not install 32 bit libraries by default for x86\_64 system. It is suggested to install all the 32 bit Linux libraries post installation. This must be done manually. To install 32 bit Linux libraries on x86\_64 system, follow below steps:

- Insert RHEL 6 CD
- cd /media/RHEL\_6.0 x86\_64 Disc 1/Packages
- rpm -ivh --force --nodeps \*.i686.rpm (If fails or doesn't work try)
- rpm -ivh --force --nodeps \*.i386.rpm
- Or
- rpm -ivh --force --nodeps \*.i586.rpm

This takes approximately 25 minutes.

This will install all of the 32-bit compatibility and emulation libraries needed to run any 32-bit application on the Redhat x86\_64 system.

If this procedure needs to be repeated the following procedure describes how to install the minimum number of 32-bit rpms necessary to get ctc execute.

1. Install the operating system in this case Redhat
2. Install CTC using installer code or through ilvss.

3. Make sure the OS installation DVD is loaded.
4. Open up an Xterm and open a file called Redhat\_32bit\_rpms or pick any name which tickles your fancy. Add a hash, #, as first character at the top of file and on the next line enter the command.
  - a. `cd /media/ RHEL_6.0 x86_64 Disc 1/packages`  
This file is used to create a script which can be used if one needs to reinstall.
5. Open up an Xterm and execute CTC. An error message complaining about not being able to find a library will appear, but if CTC executes close Redhat-32bit\_rpms and the process is DONE.
6. Open up another Xterm and execute a variant of the following command:
  - a. `cd /media/Redhat ...../packages`  
(The path to the packages directory will probably be different from OS release to release. Packages is the name of the directory which has all of the rpms both 64-bit and 32-bit.  
The 32-bit rpms are typically name with an i686.rpm suffix ( I have seen cases where i386 or even i586 have been used. So in the following you may have to change the name dependencies.)
  - b. The missing library name, call it missinglib, in step 5 is used to find the rpm which needs to be loaded. Execute the following:
    - i. `ls *missinglib*i686*.rpm`
  - c. There may be more than one rpm found. Use the rpm which has the closes match to missinglib found in step 4. Execute the following:
    - i. `rpm -i rpm` (where rpm is the rpm you want to install.)
  - d. If this command works add the command `rpm -I rpm` to Redhat\_32bit\_rpms and go back to step 5 and execute ctc.
  - e. If step 6.c.i failed it will be due to dependencies of this rpm on other rpms which are not loaded. The failure message should list these rpms.  
Install these missing rpms using the same command as step 6.c.i where rpm is now the missing rpm.
    - i. `rpm -i missing_rpm`
  - f. If this fails repeat step 6.e.i with the missing rpm(s) until the missing rpm(s) install without error.
  - g. Add the command `rpm -I missing_rpm` to Redhat\_32bit\_rpms and go back and try to reinstall the rpm which failed.
  - h. Once all of the rpm(s) are installed successfully go back to step 5 and try to execute CTC again and repeat this process.

#### Minimum software set:

- One of the supporting operating systems
- Drivers for all hardware in the system
- Intel MTA CTC Test Executive and libraries supported for the given OS
- CTC Test Modules that support the hardware under test

#### Optional Test hardware:

Actual hardware needed for full feature coverage will vary. This list represents the recommended set of Intel® custom-designed test hardware and other add-ons used to test most Intel systems:

- COM loop-back
- Parallel port loop-back
- Audio loop-back

## 2 Getting Started

Welcome to Cross-platform Test Control implementation of the Intel® Modular Test Architecture! The following topics are designed to get you started creating your first test:

- Setting up for testing
- Startup Wizard
- Configuration Wizard
- Auto-Configuration
- Creating a test configuration
- Running the test configuration

### 2.1 Setting up for testing

#### Set up the hardware.

Build a "known good" or "gold" system that is identical to the systems that will be tested. Be sure to plug in all Intel® custom or other test hardware devices needed to perform the tests. Examples:

- LAN loop-back
- COM loop-back
- parallel port loop-back
- audio MUX board
- optocoupler board (ActiveWire)

This "gold" system will be used to autoconfigure the test package.

#### Install the software.


The following software is the minimum required:

- One of the officially supported Operating Systems
- Drivers for all hardware in the system
- Cross-platform Test Control for Intel® Modular Test Architecture

Creating a test package

### 2.2 Creating a test package

**Creating a test package** by using the Startup Wizard or manually by following a these simple steps:

1. **Start CTC**  Double-Click the program icon.

2. **If the Startup Wizard** pops up, select 'New Package' and click 'OK'.
3. **Click the 'Import...' button** or Click F9 or use the menu command under Actions - Import Tests...
4. **Choose which test modules** you would like in your package by selecting their snippet files, i.e. CPU.snix and click 'OK'.
5. **Configure Tests:** You now have test devices in the template section of your test package. See Creating a Test Configuration for more details on configuring tests. This action is commonly referred to as Auto-Configure.

You are now ready to run tests.

## 2.3 Creating a test configuration

### Create the test configuration.

- The Startup Wizard can be used to create a test configuration.
- Steps to manually create a configuration:
  - From the File Menu create a new package file (Ctl-N) or open an existing one (Ctl-O).
    - In a new package you will have to import (F9) Test Modules into your empty configuration template.
    - In an existing package file there should already exist devices in the configuration template. You can import updated or add new test modules at any time in the same manner as a new package file.
  - In the Action Menu select Switch Configuration (Ctl-W) or use the Configuration Tool Bar to select the Template Configuration.
  - In the Action menu select Configure Tests (F8) or use the Configuration Tool Bar to start Auto-Configure. The tool will automatically detect hardware on the system and configure the test accordingly.
    - A series of dialog boxes may open requesting additional information about the hardware.
    - Respond to each dialog box until Auto Configure has completed structuring a test flow based on the detected devices and your configuration instructions.
  - The newly created Test Configuration can be renamed in the Configuration Tool Bar.
  - Save the test package to a different name, filename.pkx, in the Configuration Tool Bar.

You are now ready to run tests.

## 2.4 Running tests

### On each system to be tested:

- Repeat the steps in Setting up for testing on each system under test.
- Onto the local drive of the system, copy the test module files and the filename.pkx file created in Creating a test. Run CTC and open the .pkx file or



use the /PKG command line flag to automatically load the file.

**From the Sequence toolbar, press the Run button to start test flow:**

The menu command under Actions - Run Flow (F5) or the Run button on the Sequence View toolbar will run all tests in the test flow tree of the currently selected configuration. You can observe the progress of your tests by checking the icons and status in the Sequence View:

- Blue icons: Tests that have not yet run.
- Green icons: Tests have run and never failed.
- Red icons: Tests have run and failed at least once.
- Active icon: Test is currently executing.

The test sequence view will display the summary of the tree's currently selected node and as well as its children.

Individual tests or test groups can be run by right-clicking on the test or group and selecting 'Run This'. You can also select a test or group and use the menu command under Actions - Run This (Ctl-U). The sequence in which tests are run can also be changed by manipulating the Test Flow tree.

**Analyze the test results:**

Diagnostic messages will appear in the test messages view for failed tests. Determine if failures were caused by bad hardware or by incorrectly configured tests. If an incorrectly configured test was the source of the failure, use the Edit window to correct. Examples:

- Verify/correct test parameters.
- Delete unsupported/faulty tests by deleting the node from the Test Flow tree.
- Fix the hardware configuration or the bad hardware device.

Be sure to save corrected test packages!

**Rerun the test package and fix issues until all failures are all understood and mitigated.**

**Usage Example**

## 2.5 Usage Example

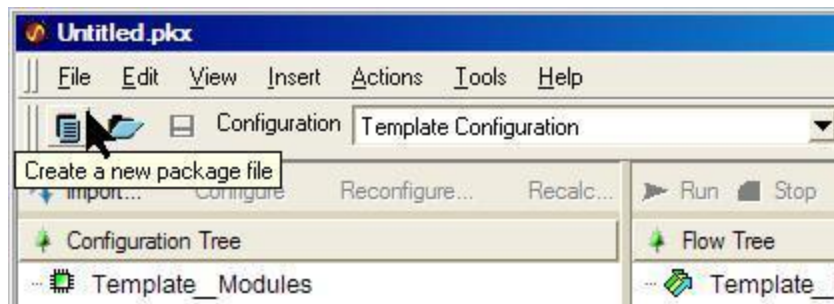
**Usage Objective** – Create a Test Package that will verify a system has four 2.2Ghz processors.

**5 Step Process**

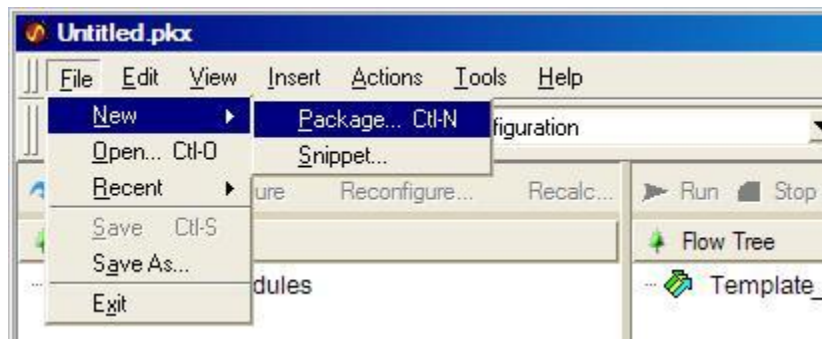
1. Create a new test package
2. Import the CPU module
3. Configure the module
4. Edit the test flow
5. Run the test flow
6. Verify the test results

## Create a new test package

Create a new test package by selecting File Menu -> New -> Package (Ctl-N).

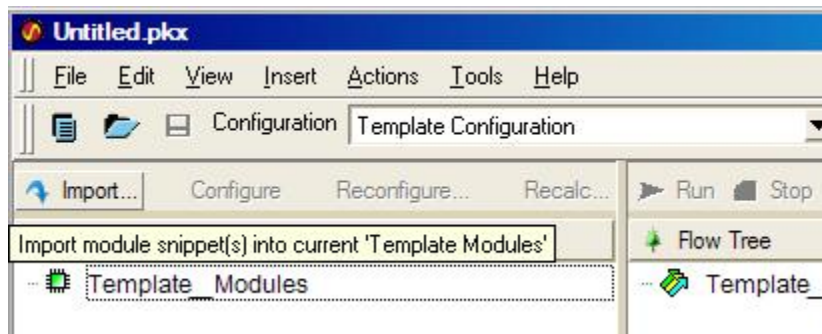


**-OR-**

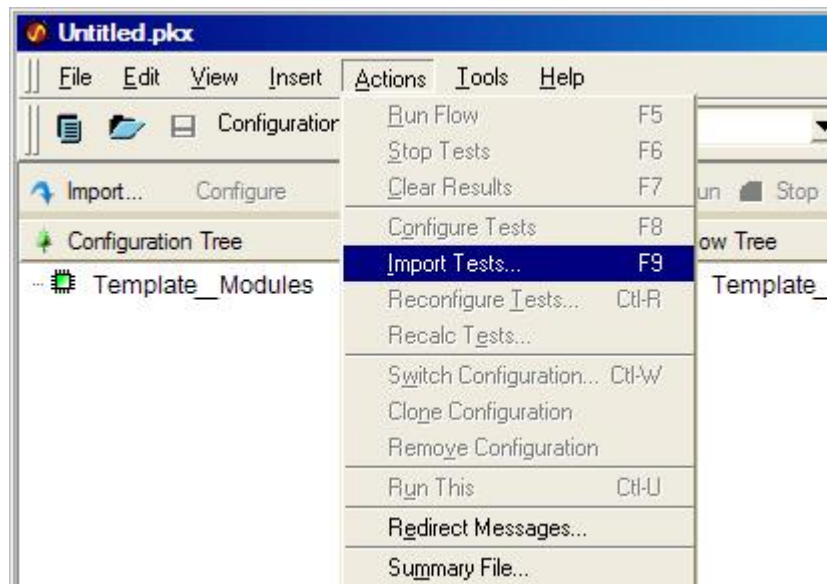


## Import the CPU module

Import the CPU.snx file into your recently created package file. This file contains all of the tests and parameters the CPU module supports.

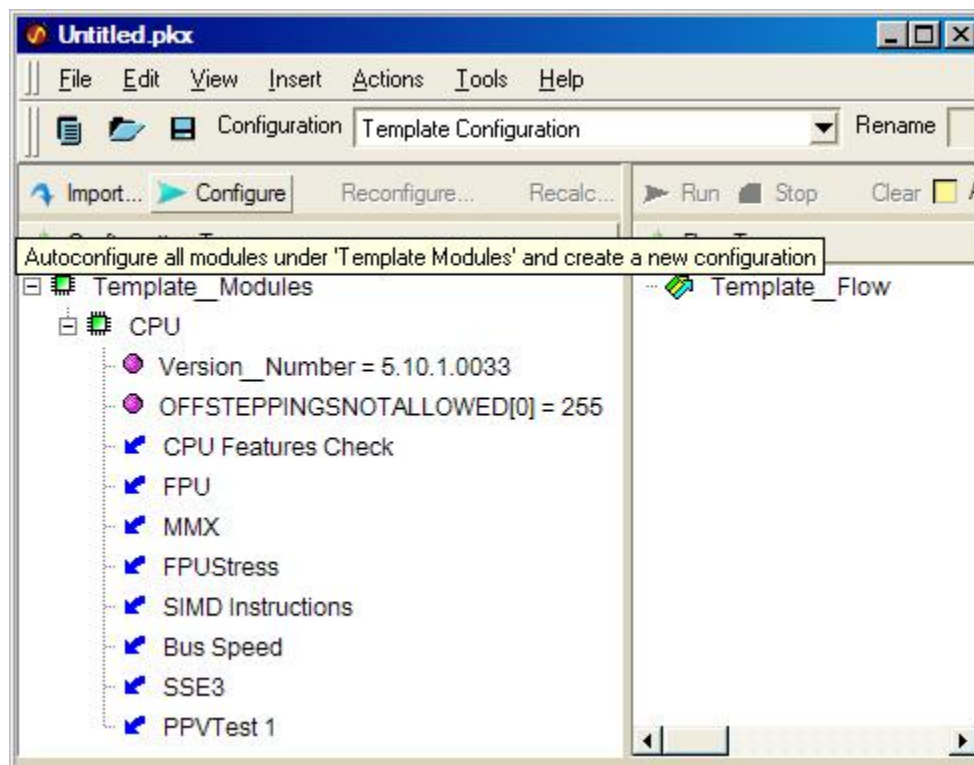


**-OR-**

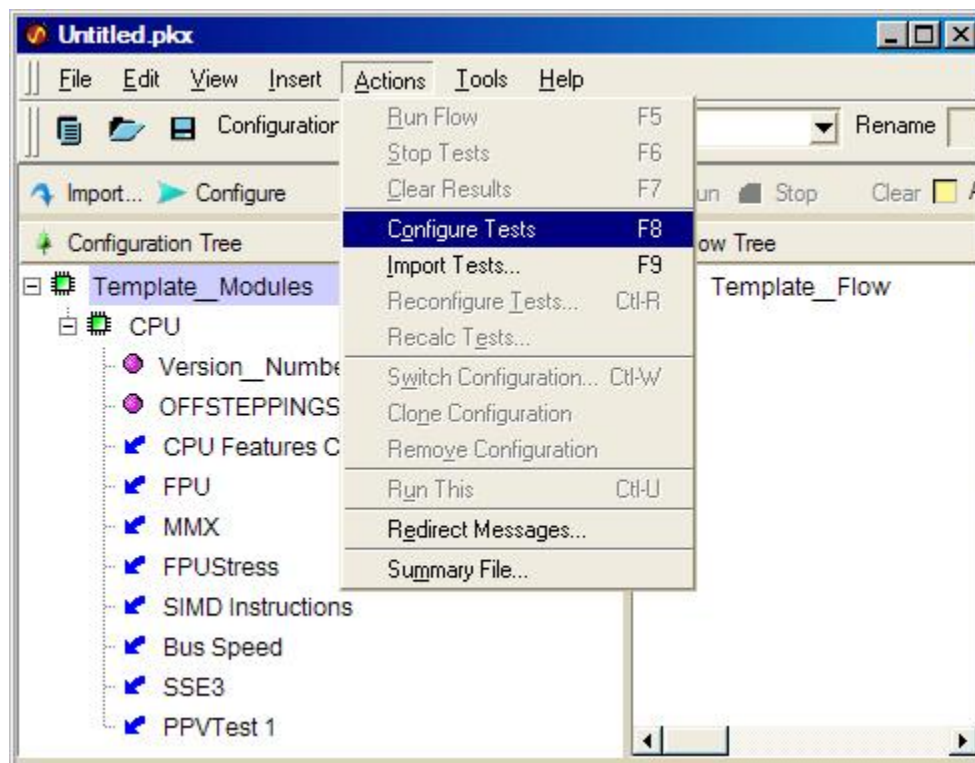


## Configure the module

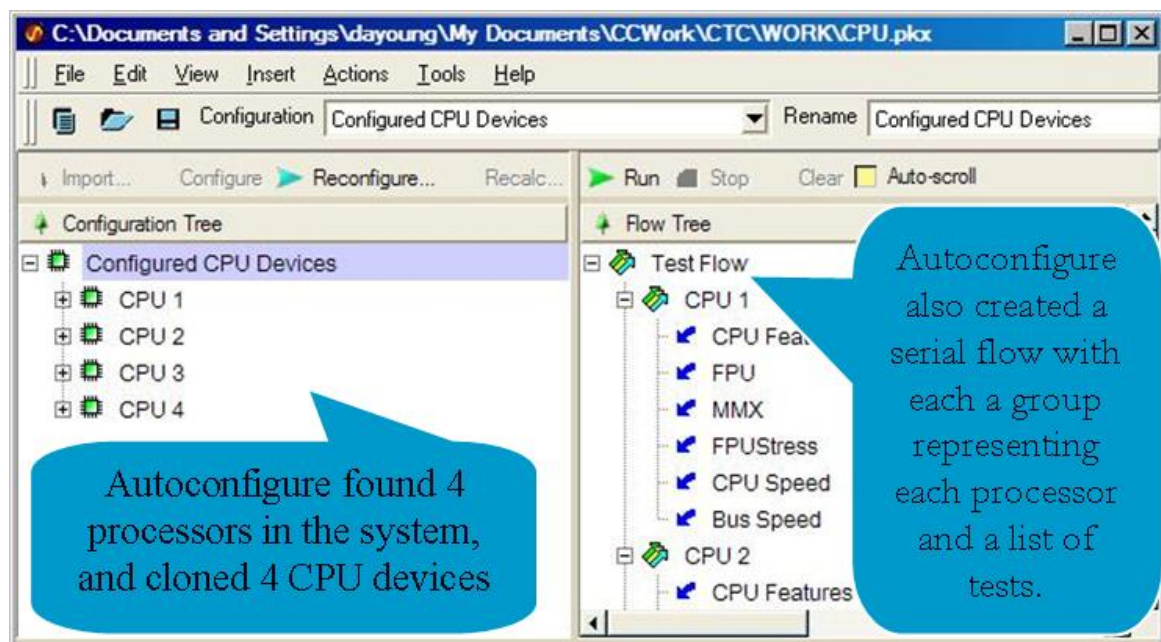
We have just imported the CPU module into our template configuration tree (Template\_Modules - CPU). Remember that the template tree contains unconfigured tests and default parameters.

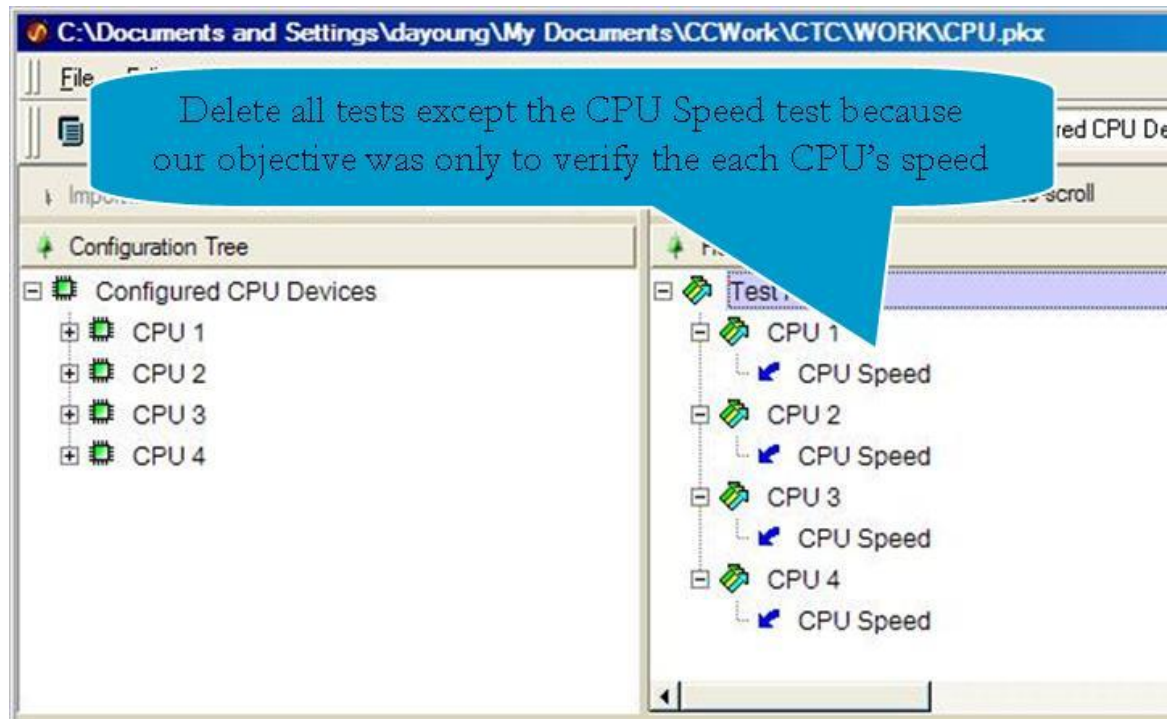


To create a configured test tree we need to configure the tests and parameters to match the hardware configuration. The requirement for this step is to auto-Configure on a "Gold" system. This means a system that has four 2.2Ghz processors installed and configured correctly.



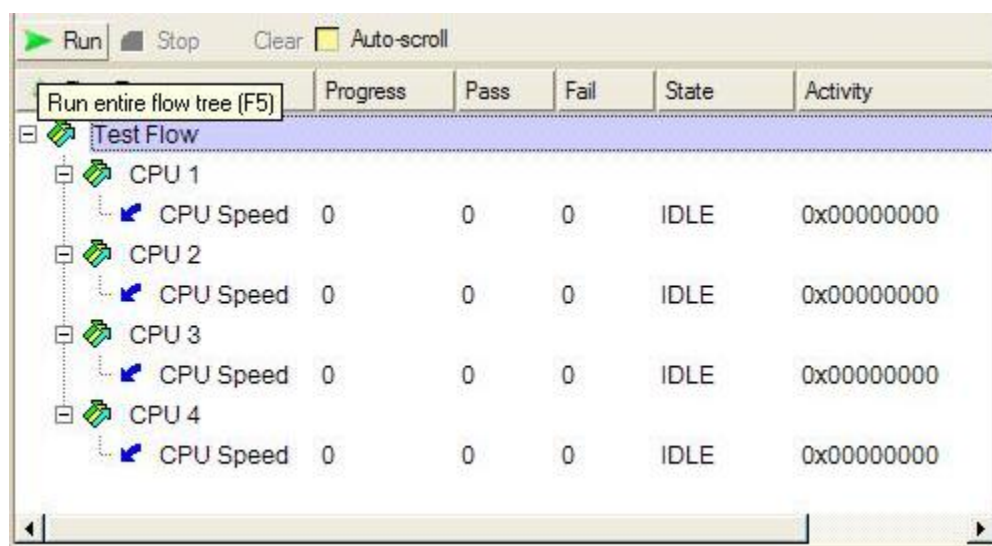
## Edit the test flow





## Run the test flow

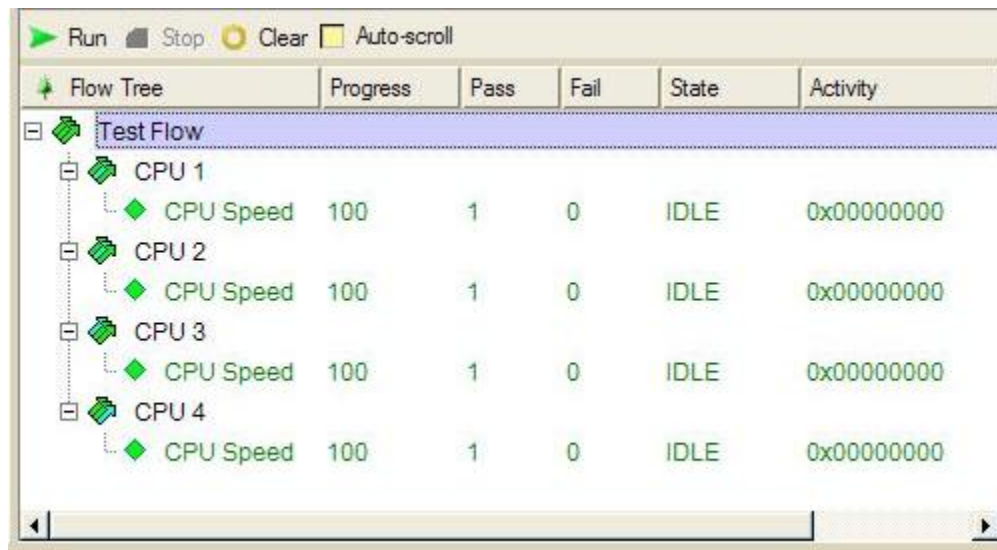
Click the run button in the test flow's tool bar to start the test running. The entire flow tree should run in the order indicated by the serial test groups.





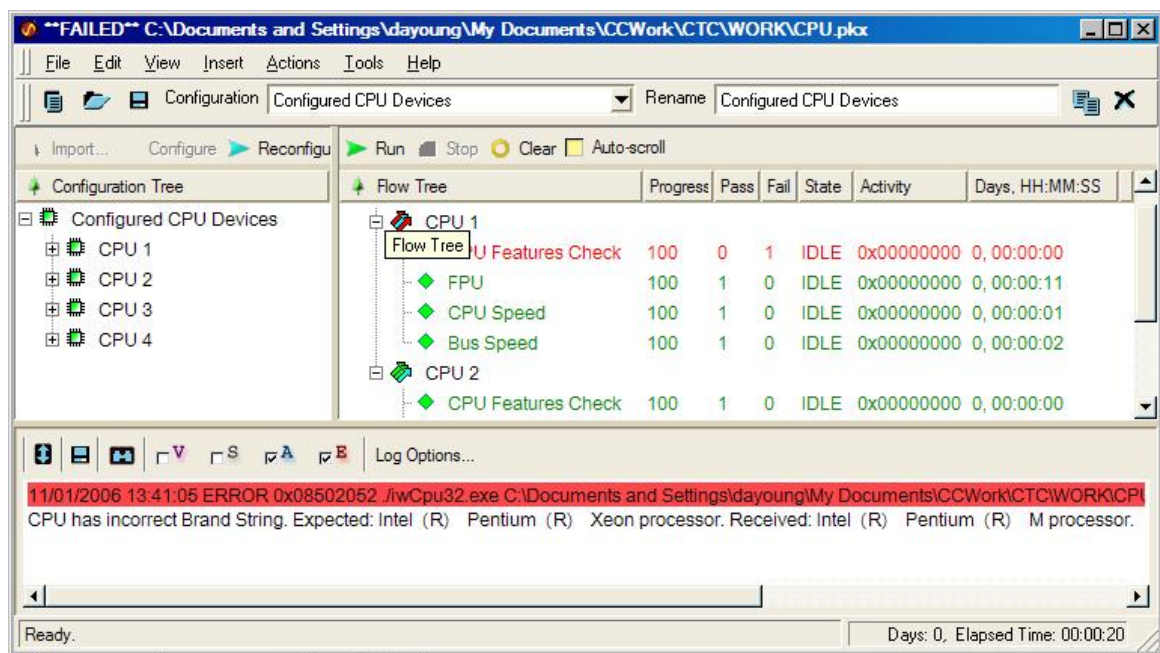
## Verify the test results

The test flow should finish, indicated by Progress being 100% and the icons no longer active. All the tests should be green indicating they passed. The Pass column should indicate each test has run 1 time.



	Progress	Pass	Fail	State	Activity
<b>Test Flow</b>					
CPU 1					
CPU Speed	100	1	0	IDLE	0x00000000
CPU 2					
CPU Speed	100	1	0	IDLE	0x00000000
CPU 3					
CPU Speed	100	1	0	IDLE	0x00000000
CPU 4					
CPU Speed	100	1	0	IDLE	0x00000000

If any test has failed the test node will turn red and the Fail column will be incremented once. There will also be an error message in the Test Message View at the bottom indicating the cause of the failure. Error will not be displayed if the Error Message display has been disabled.



	Progress	Pass	Fail	State	Activity	Days, HH:MM:SS
<b>Configuration Tree</b>						
Configured CPU Devices						
CPU 1						
CPU 2						
CPU 3						
CPU 4						
<b>Flow Tree</b>						
CPU 1						
CPU Features Check	100	0	1	IDLE	0x00000000	0, 00:00:00
FPU	100	1	0	IDLE	0x00000000	0, 00:00:11
CPU Speed	100	1	0	IDLE	0x00000000	0, 00:00:01
Bus Speed	100	1	0	IDLE	0x00000000	0, 00:00:02
CPU 2						
CPU Features Check	100	1	0	IDLE	0x00000000	0, 00:00:00

11/01/2006 13:41:05 ERROR 0x08502052 /wCpu32.exe C:\Documents and Settings\dayoung\My Documents\CCWork\CTC\WORK\CPU.plx CPU has incorrect Brand String. Expected: Intel (R) Pentium (R) Xeon processor. Received: Intel (R) Pentium (R) M processor.

## 3 Basic Concepts

This section describes all the basic components and features to manipulate a test package.

- Startup Wizard
- Configuration Wizard
- Auto-Configuration
- Test Package
- Configuration Template
- Configuration Tree
- Sequence Tree
- Module
- Device
- Test
- Parameter
- Parameter Array
- Parameter Group
- Test Group
- Message Types

### 3.1 Startup Wizard

The startup wizard is a dialog box that will pop up whenever CTC is executed. It provides an efficient method of selecting a usage option which can automate several initial steps. Each of these usage options can be performed manually or there are command line options that provide the same features. This feature can be disabled via the "Don't Show Again" check box and CTC will default to bringing up a new package file on the next execution.





### Dialog Logo

Logo contains the standard CTC dialog logo title. It also indicates the Licensing Info which describes who the license is registered to and the current expiration date.

### New Package

This option will create a new package file with an empty 'Template Configuration'. In order to run tests, modules (snippet files) will need to be imported into the 'Template Configuration' and configured, before any tests can be run.

### Quick Start

This option will create a new package file and allow you to choose which modules to initially import into the 'Template Configuration'. The default **Run Options** is to automatically configure the newly imported modules and immediately start running the configured module's tests. This is a great way to quickly start running tests.

### Open Existing Package

This option will allow you to open an existing package file with the option to configure and run the tests.

### New Snippet

This option will create a new snippet file. Test developers will need to use this option when developing new test modules.

### Existing Snippet

This option will allow a test developer to open an existing snippet file. Test developers will need to use this option to modify existing snippets when modifying tests and parameters in existing modules.

*Warning: The module owner should be the only one to edit existing snippet files. Altering the snippet file could lead to unexpected failures during autoconfigure and testing.*

### Configure Tests

Configure modules for the current system. This option is only available when using *Quick Test* where it is optional or *Open Existing Package* where it is default enabled.

### Run Tests After Configuring

This option runs all the tests in the flow in the newly created configuration. This option is only available when using *Quick Test* where it is optional when *Configure Tests* is enabled or *Open Existing Package* where it is default enabled.

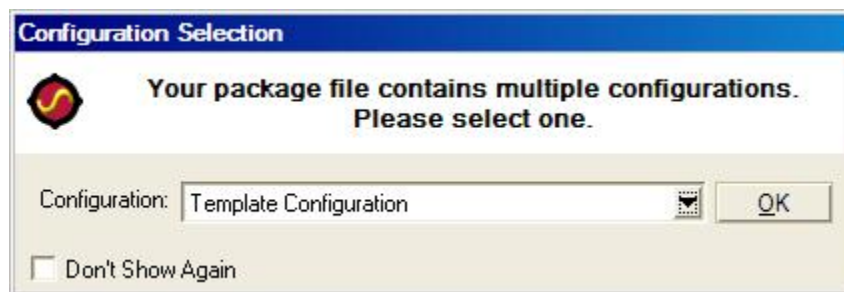
### Don't Show Again

This option disables the Start Up Wizard from running when opening the CTC GUI. This also can be turned off and back on again through the Tools Menu via the Startup Wizard radio button.

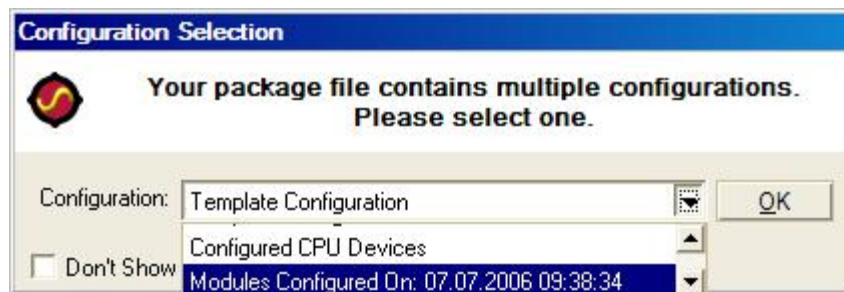
## 3.2 ConfigurationWizard

The configuration wizard is a dialog box that will pop up when opening a package file that supports multiple configurations. It allows you to select which configuration you want to select to begin with.

### Configuration Selection Dialog Box



### Configuration Drop Down Selection



**Don't Show Again**

This option disables the Configuration Wizard from running when opening a package file. CTC will default to the Template Configuration if this feature is disabled. This feature can be turned off and back on again through the Tools Menu via the Configuration Wizard radio button.

### 3.3 Auto-Configuration

**Why Auto-configure?**

- Fast & Efficient method to develop a configuration specific to a given systems hardware device's and configuration.
- When you auto-configure, it's taking a "snap-shot" of the current system's hardware configuration.
- Assuming the system you configured is a "gold" system, you can then take your configured package to another identical system and verify it truly is identical and fully functional.

**Great for Manufacturing:**

- For manufacturing we would have test modules that support each device we want to test in the board/system located in the template tree and then auto-configure on a known "gold" board/system.
- After auto-configuring we know what our systems and boards are supposed to look like on the manufacturing floor.
- We run the tests on the newly built boards and verify they are functioning exactly like the "gold" system we auto-configured on.

**Great for Validation & Stress Testing:**

- Our modules are not only developed to catch configuration mismatches, but to stress hardware components.
- Modules like the lxFMemory, lxFDiskDrv, lxFNIC, and lxFCPU all have tests designed for stressing those subsystems.
- Auto-configure is still needed to define the exact hardware configuration and envelope, like available memory, disk sizes, and number of devices available to test.
- Once auto-configure runs, the stress tests know the hardware boundaries and can test & stress within those boundaries.
- Tests can also be run in serial or parallel as well as looped for a specified number of cycles or a specified time. This enables the stressing of the system under real-world loads, instead of concentrating on individual subsystems.

### 3.4 Test Package

A test package is a configuration file CTC supports that stores a collection of configuration trees and test flow trees. A special configuration tree, called the configuration template, consists of one or more modules and is used during the auto-configuration process to generate new, customized configuration trees.

A configuration tree consists of devices and tests. A test is the smallest execution unit that can produce a pass/fail result. A test group can represent a module or a hardware device or function.

## 3.5 Configuration Template

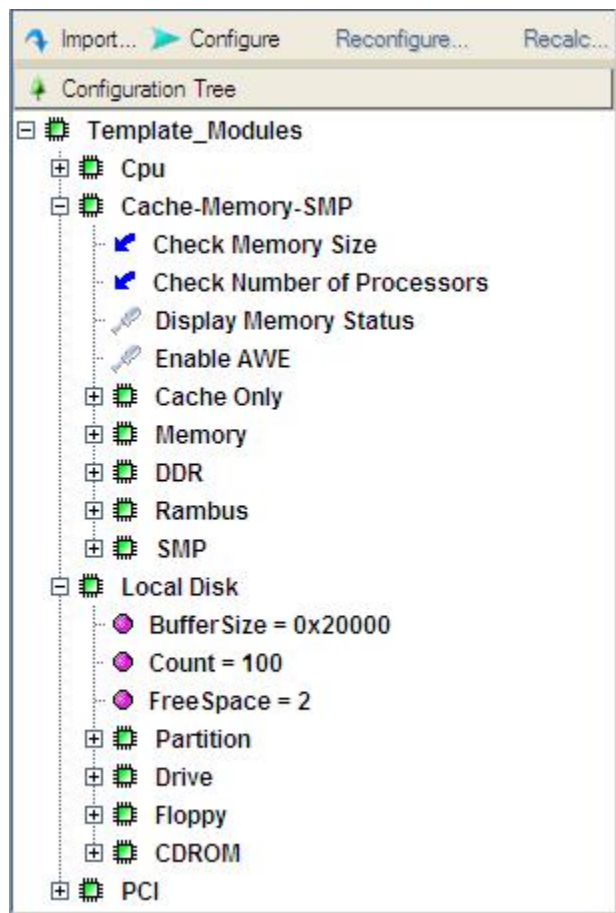
The "Configuration Template" is actually one of the configuration trees. It has a unique purpose: it is used as a template during the autoconfig process. This means that Autoconfig will invoke the autoconfig subtest of each module in the template. The output from the autoconfig subtests is used to build a new configuration. Autoconfig will not run unless the Configuration Template tree is selected.

A new test package has an empty "Autoconfig Template" by default.

### Empty Configuration Template



### Configuration Template with Module Snippets



### 3.6 Configuration Tree

A configuration tree is a collection of tests and parameters that have been adjusted for a specific hardware configuration. A package file will always contain a Configuration Template, even if it is empty. It can also contain any number of additional configurations.

See the Tool Bar for description on selecting between configurations.

### 3.7 Sequence Tree

A Sequence Tree is a hierarchy of test flow groups that controls the order of test execution. The flow groups contain tests (or other flow groups) and specify how the tests are run. The groups can be configured to run tests in parallel or serial, forever, halt on error, etc.

To add a test to the flow tree it is dragged from a configuration tree and dropped in the desired group. CTC creates a link between the new test icon in the Sequence Tree and

the test definition in the Configuration Tree. Most of the test configuration definition exists in the Configuration Tree, but can be overridden in the Sequence Tree as well. This allows the same test to be run multiple places in the test flow, as well as enables the different test instances in the test flow to be changed so they focus on different things.

An example of this would be to run several tests in serial like the CPU, Memory & Disk Drive each completely consuming the bandwidth on each device. Then running all those tests in parallel to stress the entire system, but modifying the tests so they do not consume the entire bandwidth of those devices, especially the memory test.

See the Sequence Tree View for more detailed definition.

### 3.8 Module

A module is a collection of tests encapsulated in a single binary executable that complies with the Intel® Modular Test Architecture. Modules tend to target a device subsystem, hardware device, or a class of devices.

For example, a module called IWCPU32 might have a number of tests that target the CPU as opposed to memory or chip sets. The definition is a little vague because some modules may have multiple test groups which target different hardware functions. This can be implemented by independent tests for different features or leveraging the same tests but configuring them to target specific subsystem via parameters and test patterns. An example of this would be a IWMem32 module that exercises memory, cache and symmetric multi-processing.

A module consists of the following collection of files:


module.exe	The executable file containing the test code.
module.chm	A Compiled HTML file describing all the tests, parameters, and error messages.
module.snx	A configuration tree branch defining the devices, test groups, tests and parameters.
*.dll or lib*	Any dll's or libraries required by the module binary
*.sys/vxd	Any required (custom) drivers

The name of the devices in the configuration tree need not agree with the file name.

### 3.9 Device

A Module can support one or more devices. Fundamentally a device just represents a group of tests. Generally a device represents a specific hardware device, type of device, subsystem or feature to be tested. The module snippet file should contain all of the devices it supports


The configuration template contains all the devices that the package file supports, configurations contain all the devices supported by that particular configuration which can be all or a subset of the ones in the configuration template.

Devices can contain other devices (generally considered sub devices or subsystems) and one or more tests. The icon  represents a device node.

The Device Property Sheet defines the device node properties.

### 3.10 Test


A test is the smallest executable unit of a module that returns a pass/fail result. A module represents an executable file. When the executable file is run, a parameter is passed to it to select the desired test by test number.

A test is represented by a node in the configuration tree. The Test Property Sheet defines the test node properties, some of which are only configurable via the snippet file of the individual modules. The test node is represented by an icon  and is followed by the test name. The icon can change to represent the properties of the test node as shown below.



See the Icon Property Definitions for more details on test icon properties.

### 3.11 Parameter

A parameter is an input to a test and usually describes a variable characteristic of hardware, such as the size of memory. Parameters can also be used to control the test algorithm in other ways, such as representing a testing pattern or the number of bytes to test. The icon  represents a parameter node.

A parameter has a name and a value. Parameters may be optional or required. Parameters may be numeric or string valued. The only way to know these things is to have read the modules help file.

Currently CTC supports two special parameters: the parameter[X] that represents a list or one dimensional array of parameters and the multi-dimensional array parameters [X,Y]. (X and Y represent consecutive integers from 0-n)


A parameter may be defined in one of four ways, default value in executable(documented in module help file), pre-defined in help snippet file, auto-configure may modify any parameter based on hardware, or manually over-riden by editing the parameter property value.

Parameters are inherited and can be over-riden. In other words, a test will be passed


all parameters in all of its parent groups and, if a parameter occurs more than once, only the parameter value located closest to the test (leaf node) in the tree is passed. Also parameters in the Test Flow Tree will over ride parameters in the Configuration Tree.

The Parameter Property Sheet defines the parameter node properties.

### 3.12 Parameter Array



The Parameter Array Property Sheet defines the parameter array node properties. The icon  represents a parameter array node.

### 3.13 Parameter Group

Parameter groups may contain only parameters and other parameter groups. A developer might use parameter groups to help organize parameters when they become numerous. The icon  represents a parameter group node.

The Parameter Group Property Sheet defines the parameter group node properties.

### 3.14 Flow Group

Flow Groups are used to specify how a group of tests will be executed. The icons for flow groups represent serial  and parallel  test flows. Flow groups can contain one or more flow groups or test nodes. The flow group type and a number of other parameters such as number of loops and time limit can be specified and changed using the Add or Edit Flow Group Property Sheet.

The Flow Group Property Sheet defines the test group node properties.

#### Shortcuts:

Mouse: Double-click the left mouse button on a flow group to edit its parameters.

### 3.15 Message Types

Various message types can be sent to CTC from Intel MTA test modules. Each message type can be individually filtered in the test messages tab window. Each type can also be redirected separately to a file. Each message consists of a message header and the message content.

Type	Screen Color	Use
Advisory	Cyan	Important messages to the user that should be noted, but do not cause a test failure.
Error	Red	Messages reporting test failure.
Status	Gray	General messages to the user such as test progress.
Prompt	Yellow	Messages asking for user input.



Info	White	Messages to the user that cannot be suppressed by the /Q command line switch. This type of message is used to display text for tests like "Display CMOS Map" where suppressing the output would not make sense.
Statistic	Sky Blue	Statistic messages are used for collecting statistical data from the tests. Typically these messages are Key = Value Strings.
Verbose	Magenta	Debug messages. These are not displayed unless the /V command line switch is used.

## Message Format

The message header is color code and consists of the following space delimited default information:

<Date> <Time> <Message Type> <PKG Path/Test Node Path>

<Date> - format of mm/dd/yyyy

<Time> - hh:mm:ss

<Message Type> - ERROR, STATUS, INFO, ADVISORY, STATISTIC, VERBOSE

<PKG Path> - Full file name and path

<Test Node Path> - /Configuration Name/Flow Name/Groups/.../Test Name

Error Messages also include an error code between Message Type and paths.

<0x08502009> - 0xMMMTTEEE - MMM:Module Code - TT:Test Number - EEE:Error Code

Below are coded message types:

07/11/2006 00:19:35 ERROR 0x08502009 C:\package.pfx/Test Cfg/Flow Sequence/Group 1/Test Node

Error messages are sent when a failure condition is detected by the test or the test executive. If this message is sent, the test will indicate a failure in the test status window.

07/11/2006 00:19:35 INFO C:\package.pfx/Test Cfg/Flow Sequence/Group 1/Test Node  
Informative messages generated by tests with names like "Display CPU". Also, debugging messages are tagged with this icon.

07/11/2006 00:19:35 ADVISORY C:\package.pfx/Test Cfg/Flow Sequence/Group 1/Test Node

Advisory messages are sent to indicate conditions that you should be aware of during testing, but are not failures. For example, the test executive could display an advisory if a test is sending keep-alive messages at intervals close to the test time-out period. This would indicate that the test is running slowly, which is possibly due to running too many tests in parallel.

07/11/2006 00:19:35 STATUS C:\package.pfx/Test Cfg/Flow Sequence/Group 1/Test Node

Status messages are sent to indicate progress through the test.

07/11/2006 00:19:35 PROMPT C:\package.pfx/Test Cfg/Flow Sequence/Group 1/Test Node

Prompt messages log the message and answer after a user-interactive dialog box is displayed by a test.

07/11/2006 00:19:35 STATISTIC C:\package.pfx/Test Cfg/Flow Sequence/Group 1/Test

### Node

Statistic message is a piece of data you may want to collect in the data collection system, such as the exact value of an analog measurement.

07/11/2006 00:19:35 VERBOSE C:\package.pbx/Test Cfg/Flow Sequence/Group 1/Test Node

Verbose messages are debug messages and are usually only make sense to the module developer.

### See Also:

Test Message View  
Message Log Options

## 4 Document File Types

There are several types of files that CTC can use as a document file.

<u>File Type</u>	<u>Extension</u>	<u>Description</u>
package	.pkx	an XML representation of a test package
snippet	.snx	an XML representation of a branch of a configuration tree that supports a module
summary	.csv	Test Summary File
redirect	.txt	Message log options files
language	.txt	Language Localization file which controls the GUI text via UTF-8 translation strings

### 4.1 Package File

A package file (extension .PKX) is an XML representation of a Test Package. The package file must conform to the XML template file xmlpkg.dtd.

### 4.2 Snippet File

This snippet file (extension .SNX) is a subset of the package file that supports a specific test module. The purpose of the snippet file is to define the default definition of the test module's supported devices, subtests and parameters. It contains a branch of a Configuration Template for each device the test module supports and does not contain a test flow tree.

The snippet file can be opened directly with the File Open menu. The snippet file can be inserted into a package file via the Import... button in the Configuration Tree window or thru the Actions Menu -> Import Tests command.

The snippet file must conform to the XML template file xmlpkg.dtd.

## 4.3 Summary File

The summary file is a text file that records the cumulative test results for all invocations of the Test Executive. In the file is a line for each test that summarizes the number of passes, failures, and errors that each test has reported. The format of the file (extension CSV) is as follows:

The summary file is updated once per second during testing. By default CTC will update a file named "ctc\_summary.csv", but you can change the name in the Message Redirection Tab or by using the /SUMMARY invocation parameter.

```

--- Test --- Pkg Path & Flow Path ----- State Pass Fail Activity Progress Start
Time End Time:
testname pkgpath/configname/flowpath/testname IDLE ##### 0x### ##%
mm/dd/yyyy hh:mm:ss

```

Note that the file is cumulative across Test Executive invocations. Summary can be cleared by deleting the file.

### Sample Output

Test	Test Path	State	Pass	Fail	Activity	Progress	Start Time	End Time
CPU Features	CPU.pkx/Cfg1/Testing Sequence/CPU 0/CPU Features Check	IDLE	0	1	0x00000000	100%	'07/11/2006 20:50:47'	'07/11/2006 20:50:47'
FPU	CPU.pkx/Cfg1/Testing Sequence/CPU 0/FPU	IDLE	1	0	0x00000000	100%	'07/11/2006 20:50:47'	'07/11/2006 20:50:57'
MMX	CPU.pkx/Cfg1/Testing Sequence/CPU 0/MMX	IDLE	1	0	0x00000000	100%	'07/11/2006 20:50:57'	'07/11/2006 20:51:07'
FPUStress	CPU.pkx/Cfg1/Testing Sequence/CPU 0/FPUStress	IDLE	1	0	0x00000000	100%	'07/11/2006 20:51:07'	'07/11/2006 20:51:10'
SIMD Instrs	CPU.pkx/Cfg1/Testing Sequence/CPU 0/SIMD Instructions	IDLE	1	0	0x00000000	100%	'07/11/2006 20:51:10'	'07/11/2006 20:51:11'
Bus Speed	CPU.pkx/Cfg1/Testing Sequence/CPU 0/Bus Speed	IDLE	1	0	0x00000000	100%	'07/11/2006 20:51:11'	'07/11/2006 20:51:13'

## 4.4 Message Redirection Files

Message redirection files are text files that CTC redirects test messages into. All test message types can be individually redirected to a text file or they can all be redirected to a single file. The messages are redirected into the file in the same format as they appear in the Test Messages window, minus the color coding. The message redirection can be configured in the Message Log Options or by using command line flags.

## 4.5 XML DTD File

A DTD is a formal description in XML Declaration Syntax of a particular type of document. The below DTD is required to be in the same directory as the PKX or SNX file. Its filename should be "xmlpkg.dtd".

```

<!ELEMENT package (version, modules, flows?, configurations?)>
<!-- package
    name CDATA #IMPLIED
    version CDATA #FIXED "0.5"
-->

<!-- version (#PCDATA)>
<!-- modules (device)*>
<!-- modules
    expanded CDATA #IMPLIED
-->

```

```

<!ELEMENT flows (flow | group)*>
<!ATTLIST flows
    expanded CDATA #IMPLIED
>

<!ELEMENT configurations (cfg)*>
<!ATTLIST configurations
    expanded CDATA #IMPLIED
>

<!ELEMENT device (param*, paramarray*, test*, paramgroup*,device*)*>
<!ATTLIST device
    name CDATA #REQUIRED
    binary CDATA #IMPLIED
    timeout CDATA #IMPLIED
    version CDATA #IMPLIED
    expanded CDATA #IMPLIED
    cloneof CDATA #IMPLIED
    configurable CDATA #IMPLIED
>

<!ELEMENT paramgroup (param | paramgroup | paramarray)*>
<!ATTLIST paramgroup
    name CDATA #REQUIRED
    expanded CDATA #IMPLIED
>

<!ELEMENT paramarray (paramarraycolhead | paramarraycell+)*>
<!ATTLIST paramarray
    name CDATA #REQUIRED
    x_size CDATA #REQUIRED
    y_size CDATA #REQUIRED
    expanded CDATA #IMPLIED
>

<!ELEMENT paramarraycell EMPTY>
<!ATTLIST paramarraycell
    value CDATA #REQUIRED
    x CDATA #REQUIRED
    y CDATA #REQUIRED
    expanded CDATA #IMPLIED
>

<!ELEMENT paramarraycolhead EMPTY>
<!ATTLIST paramarraycolhead
    value CDATA #REQUIRED
    col CDATA #REQUIRED
    size CDATA #IMPLIED
>

<!ELEMENT param EMPTY>
<!ATTLIST param

```

```
    name CDATA #REQUIRED
    value CDATA #REQUIRED
    type CDATA #IMPLIED
    expanded CDATA #IMPLIED
>

<!ELEMENT test (param | paramgroup | paramarray)*>
<!ATTLIST test
    name CDATA #IMPLIED
    number CDATA #IMPLIED
    ref CDATA #IMPLIED
    type (generic | utility) "generic"
    destructive (yes | no) "no"
    interactive (yes | no) "no"
    disabled CDATA #IMPLIED
    timeout CDATA #IMPLIED
    binary CDATA #IMPLIED
    expanded CDATA #IMPLIED
    requires CDATA #IMPLIED
    configurable CDATA #IMPLIED
>

<!ELEMENT group (device | test | group | param | paramarray)*>
<!ATTLIST group
    name CDATA #REQUIRED
    type (serial | parallel | parallel_exact) "serial"
    onerror (halt | continue | pause | trigger) "continue"
    loop (forever | time | number) "number"
    loopt CDATA #IMPLIED
    loopn CDATA #IMPLIED
    timeout CDATA #IMPLIED
    expanded CDATA #IMPLIED
>

<!ELEMENT flow (device | test | group | param | paramarray)*>
<!ATTLIST flow
    name CDATA #REQUIRED
    type (serial | parallel | parallel_exact) "serial"
    onerror (halt | continue | pause | trigger) "continue"
    loop (forever | time | number) "number"
    loopt CDATA #IMPLIED
    loopn CDATA #IMPLIED
    timeout CDATA #IMPLIED
    expanded CDATA #IMPLIED
>

<!ELEMENT cfg (device | flow)+>
<!ATTLIST cfg
    name CDATA #IMPLIED
    timeout CDATA #IMPLIED
    expanded CDATA #IMPLIED
>
```

## 4.6 Language Localization

CTC supports language localization of the GUI interface text only. This support is controlled via a UTF8 formatted file named "*GUI\_selection.txt*". When the file exists in the same directory as the CTC binary the GUI toolbar displays a list control that allows the selection of any language supported in that file. This feature does not control the configuration and flow trees which are controlled via the SNX/PKX files, as well as the test messages are still in the native language supported by the test module binary.

The file must be a UTF8 formatted file, and has a basic ini format file structure. The first section [languages] indicates the different languages supported and is what populates the selection list. Each string used in the GUI has a [parameter] section that contains a key=value for each language supported. The key must be the exact language name used in the [languages] section and the value is the subsequent translated text in that language. The GUI will default to the hard coded English text if there is not a translation line for the selected language found.

A language localization file is released with the Test Executive and has the full compliment of supported text sections. This file can be modified and additional translation languages can be added as needed.

---

```
[languages]
lang1=english
lang2=chinese
lang3=japanese

[Linux]
english=Linux(tm)
chinese=Linux(tm)

[Windows]
english=Windows(R)
chinese=Windows(R)

[IMTA]
english=Intel(R) Modular Test Architecture
chinese=Intel(R) ??????????

[run button]
english=Run
chinese=??

[run button->tip]
english=Run entire flow tree <F5>
chinese=???? <F5>

[run this]
english=Run This
chinese=????????

[stop button]
```

english=Stop  
chinese=??

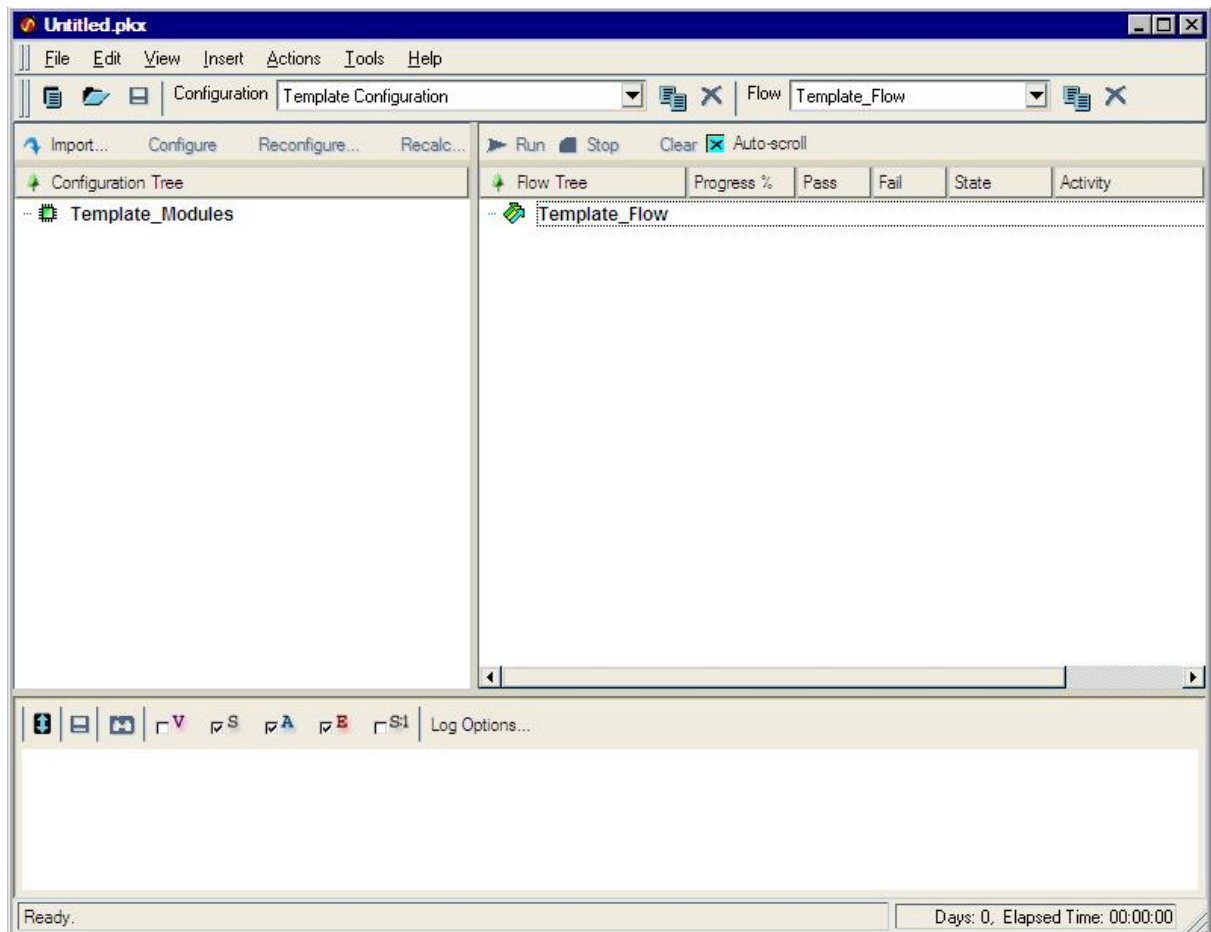
[stop button->tip]  
english=Signal all tests to abort <F6>  
chinese=??????? <F6>

[clear button]  
english=Clear  
chinese=??

[clear button->tip]  
english=Clear the Test Summary and Test Message window <F7>  
chinese=??????????? <F7>

## 5 CTC User Interface

The User Interface consists of a window split into three sections the Configuration View pane, Sequence View pane and the Message View pane. It also contains a file menu bar, a tool bar and a tooltip bar.










## 5.1 Title Bar


The title bar is located along the top of a window. It contains the name of the application and document. To move the window, drag the title bar. Note: You can also move dialog boxes by dragging their title bars.

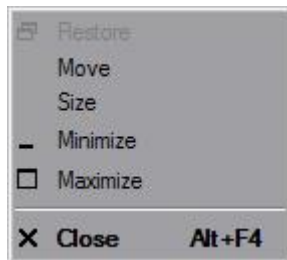


The title bar contains the following elements:

-  Application Control - menu button
-  **\*\*PASSING\*\*** Passing test indicator, exists when test are executed
-  **\*\*FAILED\*\*** Failed test indicator, exists when test are executed
-  **C:\Program Files\CTC\CPU.plx** Name & path of the document
-  Minimize button
-  Maximize button (or Restore, if already maximized)
-  Exit button

### 5.1.1 Application Control

Use the  Application Control menu button control the CTC window. CTC presents the standard application control dialog box to modify the window.

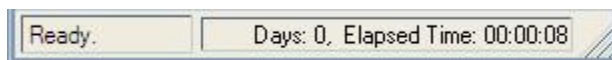


**See the following for command details:**

<b>Restore:</b>	Restore the window to the last size when maximized.
<b>Move:</b>	Sets the window and mouse icon is drag mode.
<b>Size:</b>	Sets the window and mouse icon is resize mode.
<b>Minimize:</b>	Minimizes the window to an icon.
<b>Maximize:</b>	Maximized the window to full screen.
<b>Close:</b>	Exits the CTC application.

### 5.1.2 Status Bar

The status bar is displayed at the bottom of the CTC window.



The left text area of the status bar is the tool tip control which describes actions of menu



items as you use the arrow keys to navigate through menus. This area similarly shows messages that describe the actions of toolbar buttons as you depress them, before releasing them. If after viewing the description of the toolbar button command you wish not to execute the command, then release the mouse button while the pointer is off the toolbar button.

The right text area of the status bar timestamp counter for the last executed or currently executing test run:

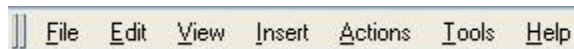
Field	Description
Days: 0	The number of days executed
00:00:20	Elapsed time in hours:minutes:seconds



This button allows the CTC window to be resized when pressed & dragged.

## 5.2 Menu Bar

The Menu Bar is a free floating toolbar. It is used to edit or configure the CTC User Interface and test package.

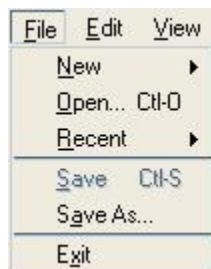


The Menu bar has the following Menus.

- File Menu
- Edit Menu
- View menu
- Insert Menu
- Actions Menu
- Tools Menu
- Help Menu

### 5.2.1 File menu

The File menu offers the following commands:



**See the following for command details:**

New	Opens a new blank test package file.
Open...	Opens an browser dialog to open an existing package or snippet

	file.
Recent	Opens a recently edited package or snippet file.
Save	Saves the current file to existing file name.
Save As...	Opens a browser dialog to save the existing file to selected location and file name.
Exit	Exits CTC executive.

### 5.2.1.1 New

Use this command to create a new PKX or SNX package in CTC.

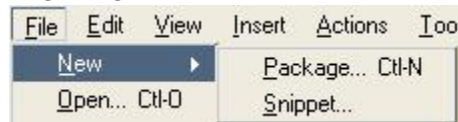
A new test package will have one empty configuration template tree, and one empty test flow template tree.

A new module snippet package will have one empty configuration template tree, and will have no sequence tree window since it the sequence tree is not supported in snippet files.

You can open an existing test package with Open.

#### Shortcuts

File->New:



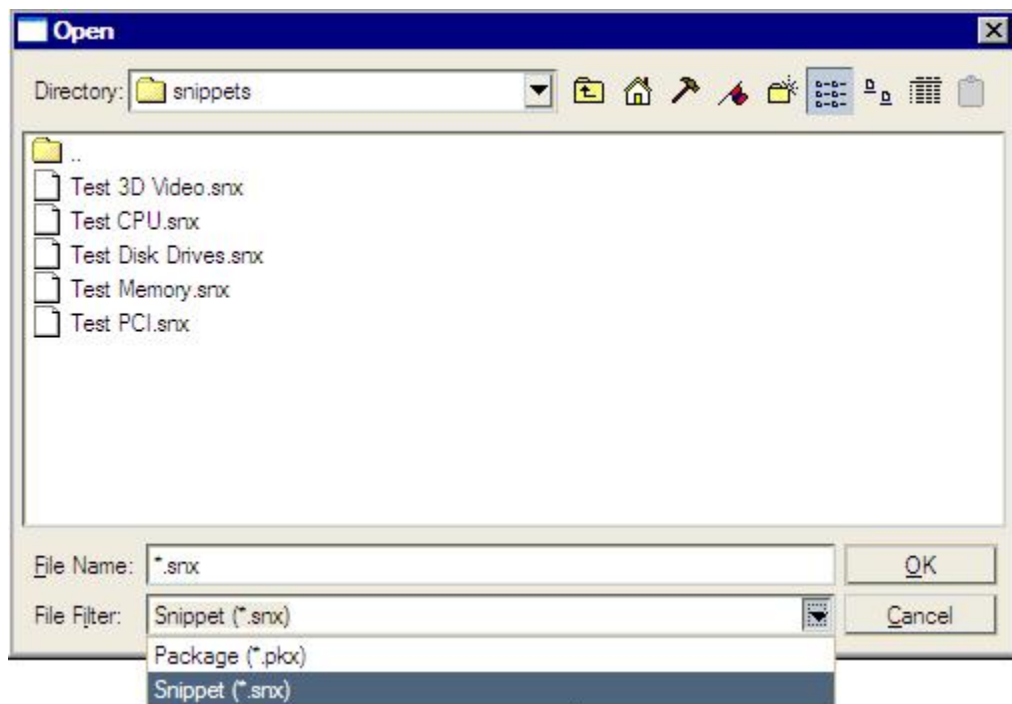
Toolbar:



Keys: Ctrl+N

### 5.2.1.2 Open

Use this command to open existing files. CTC presents the standard file open dialog box so you can browse for the file of your choosing. There are only two file extensions that are supported by this dialog box ,the PKX and SNX.



**Note:** Use this command to open existing documents. To create new documents use New.

### Shortcuts

Toolbar:



Keys: Ctrl+O

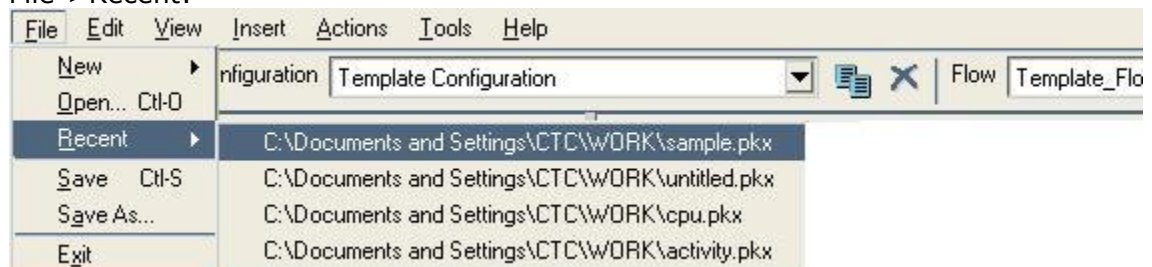
### 5.2.1.3 Recent

Use this command to open one of the last four recently used files.

You can open any existing file with Open.

### Shortcuts

File->Recent:



#### 5.2.1.4 Save

Use this command to save the current test package to its current name and directory. When you save a test package for the first time, CTC displays the Save As dialog box so you can name your document. If you want to change the name and directory of an existing document before you save it, use Save As.

##### Shortcuts

Toolbar:

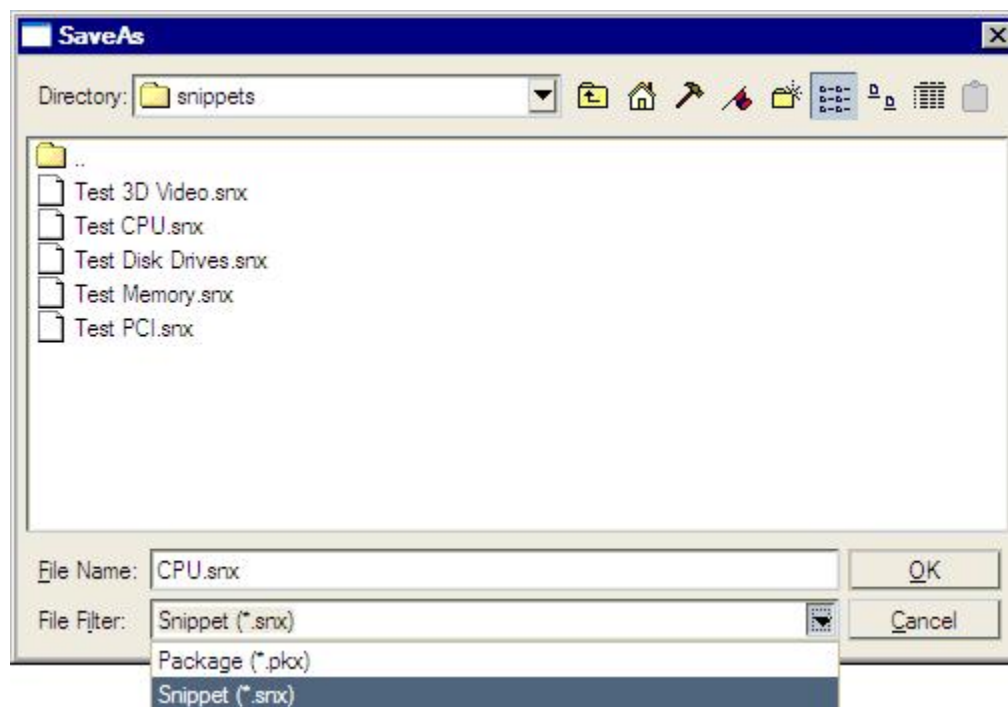


Keys:

Ctrl+S

#### 5.2.1.5 Save As

Use this command to save the existing file to a different destination or filename. CTC presents the standard file save dialog box so you can name your test package file. Similar to the Open dialog this dialog only support PKX and SNX file extensions.



This command may be used to save the file as a test package (PKX), or a snippet file (SNX).

To save a document without changing its name or directory, use the Save command.

#### 5.2.1.6 Exit

Use this command to end your CTC session. CTC prompts you to save documents with unsaved changes.

##### Shortcuts

Mouse: Double-click the application's icon in the upper left of the application window, or single-click the X in the upper right.



## 5.2.2 Edit menu

The Edit menu offers the following commands:

Edit	View	Insert	Actions	I
Undo			Ctrl-Z	
Redo			Ctrl-Y	
Cut			Ctrl-X	
Copy			Ctrl-C	
Paste			Ctrl-V	
Switch Configuration...			Ctrl-W	
Clone Configuration				
Remove Configuration				
Switch Flow			Ctrl-F	
New Flow				
Clone Flow				
Remove Flow				
Node Properties...			Ctrl-Shift-P	
Package Properties...				

### See the following for command details:

Undo	Reverses the previous change to the test package. Note that "Undo" may need to be pressed several times if the previous action affected several items. For example, if a group is deleted from the configuration, "Undo" must be pressed multiple times until each test in the corresponding flow for that group has been added back.
Redo	Re-applies the last change reversed by Undo.
Cut	Deletes data from the document and moves it to the clipboard.
Copy	Copies data from the document to the clipboard.
Paste	Pastes data from the clipboard into the document.
Switch Configuration	Switch to the next configuration.
Clone Configuration	Create a new copy of the current configuration.
Remove Configuration	Delete the current configuration from the Package or Snippet file.
Switch Flow	Switch to the next flow.
New Flow	Create a new flow.
Clone Flow	Create a new copy of the current flow.
Remove Flow	Delete the current flow from the Package or Snippet file.
Node Properties...	Edits the currently selected node (device, test, group,

Package Properties... parameter, etc)  
Edits the package properties.

#### 5.2.2.1 Undo

Use this command to reverse the last editing action, if possible. The last four editing operations to the test package can be reversed. The command is disabled if there is no more edits to undo.

##### Shortcuts

Keys:      Ctl+Z

##### See Also:

Redo

#### 5.2.2.2 Redo

Use this command to re-apply the last editing action previously reversed by Undo. The last four undo operations can be re-applied. The Redo command is disabled in the menu if there are no more undo actions to redo.

##### Shortcuts

Keys:      Ctl+Y

##### See Also:

Undo

#### 5.2.2.3 Cut

Use this command to remove the currently selected branch from the configuration tree or sequence tree and put it on the clipboard.

Cutting data to the clipboard replaces the contents previously stored there.

##### Shortcuts

Keys:      Ctl+X

#### 5.2.2.4 Copy

Use this command to copy the currently selected branch from the configuration tree or flow tree to the clipboard.

Copying data to the clipboard replaces the contents previously stored there.

##### Shortcuts

Keys:      Ctl+C

#### 5.2.2.5 Paste

Use this command to insert a copy of the clipboard contents at the insertion point. This command is disabled if the clipboard is empty.

##### Shortcuts

Keys: Ctl+V

#### 5.2.2.6 Switch Configuration...

Use this command to switch configurations. This command will rotate sequentially thru the defined configurations in the package file, the next configuration will be displayed in the Configuration View. This command and selection control are disabled when tests are running.

##### Shortcuts

Toolbar:

Keys: Ctl-W



#### 5.2.2.7 Clone Configuration

Use this command to copy the current configuration to a new configuration. The new configuration will be the one displayed in the Configuration View. The new configuration will be renamed with '\_clone\_#' added to the original configuration name. The '#' will default to 0 or if there are already clones of that specific configuration CTC will add one to the largest one that exists. This command & button are disabled when tests are running.

##### Shortcuts

Toolbar:



#### 5.2.2.8 Remove Configuration

Use this command to delete the current configuration from the package file. This command & button are disabled when tests are running, and when editing the template configuration.

##### Shortcuts

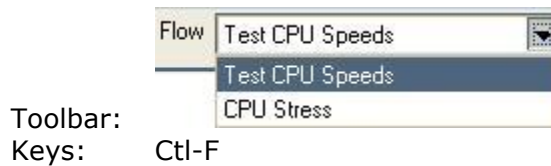
Toolbar:



#### 5.2.2.9 Switch Flow

Use this command to switch to the next flow in the current configuration. This command will rotate sequentially thru the defined flows in the package file, the next flow will be displayed in the Sequence View. This command and selection control are disabled when tests are running or the current configuration only has one flow defined.

##### Shortcuts



#### 5.2.2.10 New Flow

Use this command to create a new flow under the current configuration. The new empty flow will be displayed in the Sequence View window and named "New Flow". This command and selection control are disabled when tests are running.

#### 5.2.2.11 Clone Flow

Use this command to make a copy of the current flow. The new flow will be the one displayed in the Sequence View. The new flow will be renamed with '#' added to the original configuration name. This command & button are disabled when tests are running.


##### Shortcuts

Toolbar: 

#### 5.2.2.12 Remove Flow

Use this command to delete the current flow from the package file. This command & button are disabled when tests are running, and when there is only one flow in the configuration.

##### Shortcuts

Toolbar: 

#### 5.2.2.13 Node Properties...

Use this command to edit the properties of the currently selected node. This command will display the Property Sheet for the node selected.

See the individual node property sheets for further details.

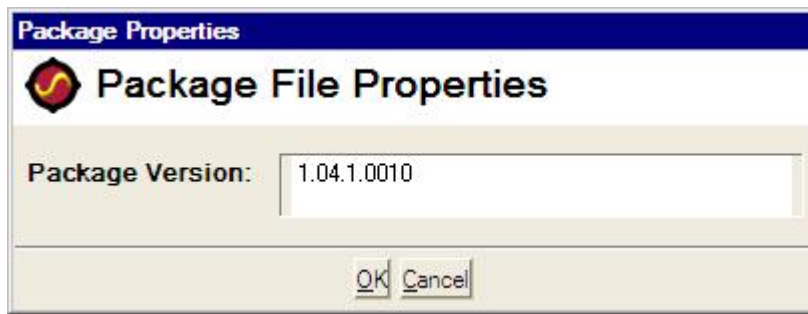
##### Shortcuts

Mouse: Double-click on a node or Right-click and select Edit Node from the shortcut menu.  
Keys: Enter

#### 5.2.2.14 Package Properties...

Use this command to edit the package files version. CTC displays the package file version in the Help About menu.



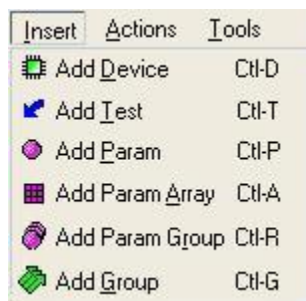


### Properties:

Package Version      Use this property to edit the package files version.

## 5.2.3 Insert menu

The insert menu offers the following commands



**See the following for command details:**

- Add Device
- Add Test
- Add Param
- Add Param Array
- Add Param Group
- Add Group

### 5.2.3.1 Add Device

Use this command to add a Device Node to the configuration tree. The new device is added inside the currently selected group. If a group is not selected, this command is unavailable.

The Device Property Sheet defines the device node properties.

### 5.2.3.2 Add Test

Use this command to add a test to a device or test group. The new test is added inside the currently selected device or group node. If a device or group node is not selected, this command is unavailable.

The Test Property Sheet defines the test node properties.

### 5.2.3.3 Add Parameter

Use this command to add a parameter to a device, test or test group.

The Parameter Property Sheet defines the parameter node properties.

### 5.2.3.4 Add Parameter Array

Use this command to add a parameter array to a device, test or test group.

The Parameter Array Property Sheet defines the parameter array node properties.

### 5.2.3.5 Add Parameter Group

Use this command to add a parameter group to a device, test or test group node. The current selection must be a test or group to make this command available.

The Parameter Group Property Sheet defines the parameter group node properties.

### 5.2.3.6 Add Test Group

Use this command to add a test group to a device or test group. The new test is added inside the currently selected device or test group node. If a device or test group node is not selected, this command is unavailable.

The Test Group Property Sheet defines the test group node properties.

## 5.2.4 Action menu

The Test Flow menu offers the following commands

A	T	H
Run Flow		F5
Stop Tests		F6
Clear Results		F7
Configure Tests		F8
Import Tests...		F9
Reconfigure Tests...	Ctrl	R
Recalc Tests...		
Run This	Ctrl	U
Log Options...		

**See the following for command details:**


Run Flow

Stop Tests  
Clear Results  
  
Configure Tests  
Import Tests...  
Reconfigure Tests...  
Recalc Tests...  
  
Run This  
  
Log Options...

#### 5.2.4.1 Run Flow

Use this command to start the execution of the test flow displayed in the Sequence View. The Sequence View will keep track of the run data in the tree view. The Test Message View will display any messages from tests, test messages are filtered so may not all be displayed. The command & button are disabled when tests are running.


##### Shortcuts

Toolbar:   
Keys: F5

#### 5.2.4.2 Stop Tests

Use this command to stop the execution of the test flow displayed in the Sequence View. The Sequence View will mark all currently running test with an ABORTED status. This command and button are disabled when no tests are running.

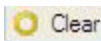
##### Shortcuts

Toolbar:   
Keys: F6

#### 5.2.4.3 Clear Results

Use this command to clear the results in the test flow tree displayed in the Sequence View. This command will also clear all the messages out of the Test Message View. Command & Button disabled when tests are running.

##### Shortcuts

Toolbar:   
Keys: F7

#### 5.2.4.4 Configure Tests...

Use this command to auto-configure a new configuration in the Configuration View. This command & button only available when the configuration template is selected.

See Configure for more details.

### Shortcuts

Toolbar:   
Keys: F8

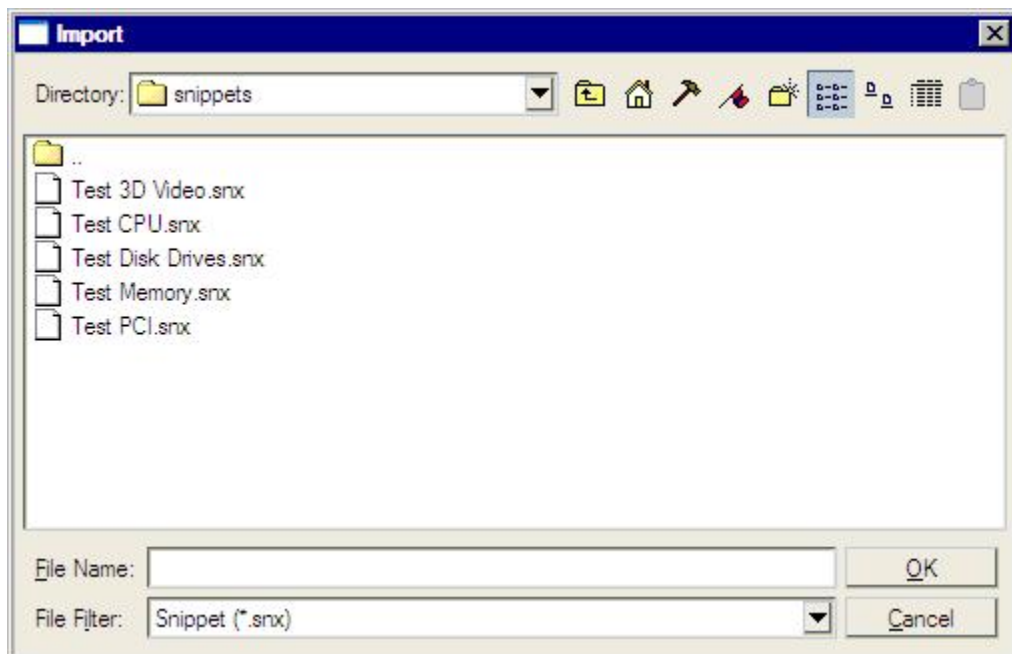
#### 5.2.4.5 Import Tests...

Use this command to import test module snippets into the Configuration Template. CTC presents the standard open file dialog box so you can search and select one or more snippet files from a directory. This command & button only available when the configuration template is selected.

### Shortcuts

Toolbar:   
Keys: **F9**

### Import Dialog



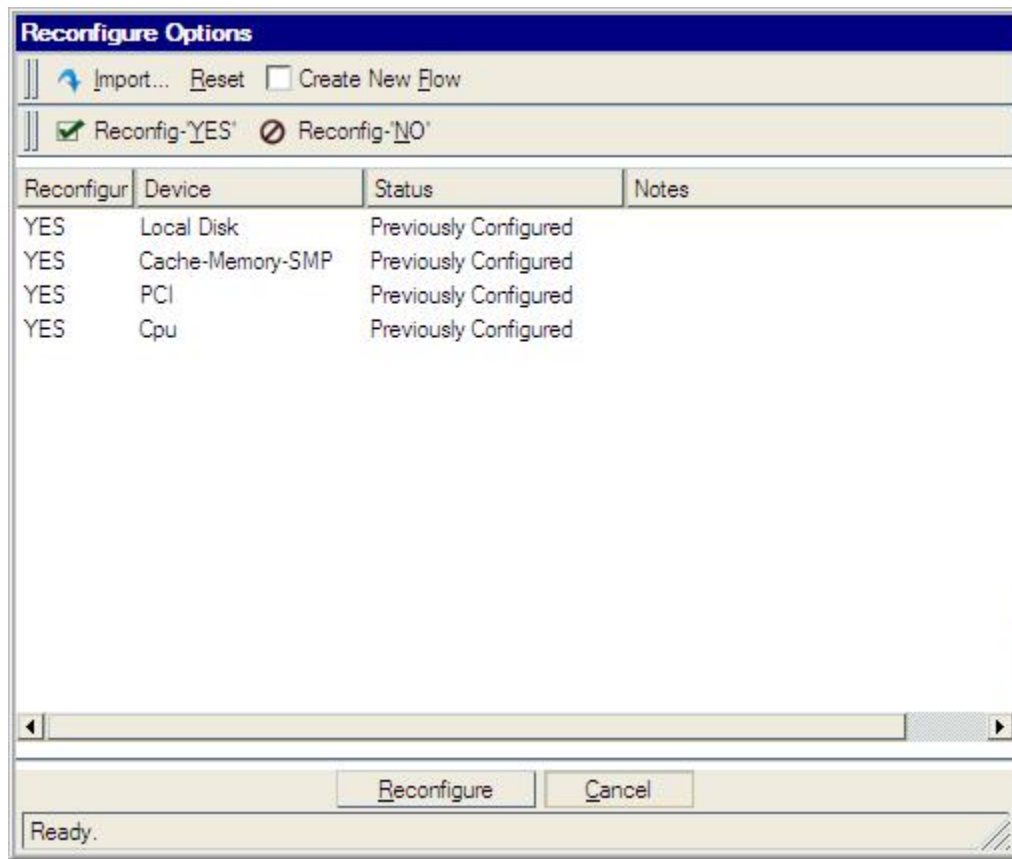
#### 5.2.4.6 ReConfigure Tests...

Use this command to re-configure the tests in the current configuration in the Configuration View. This command & button not available in the configuration template, this is due to the fact that it is not configured.

### Shortcuts

Toolbar: 

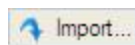
### Reconfigure Dialog Box



Re-Configure is handy when you have a configuration with only a subset of the devices from the template configuration tree, or if your configuration's flow tree differs from the template flow. It is also useful when importing a new or updating a test module into your test package.

**Autoconfigure** copies all the devices from the template's configuration tree into a new configuration and runs the auto configuration process on them, it also copies the template's flow tree into the new configuration. **Re-Configure** only runs autoconfigure on selected devices within the currently selected configuration tree. It does not create a new configuration, it makes the changes to the current one. The following dialog is displayed, which allows the user to select the devices to reconfigure. It also allows the user to import new devices from snippet files on disk.

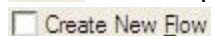
### Control Property Descriptions:



Import Button: Brings up the Import Dialog box to add new devices to the list.



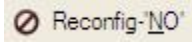





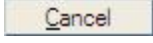
Reset Button: Resets the list back to original state.



New Flow: Check box to create a new flow with reconfiguration changes.



Reconfig-Yes: Changes the Reconfig Column of the list item(s) selected to YES.

	Reconfig-No:	Changes the Reconfig Column of the list item(s) selected to NO.
	Reconfig Column:	Indicates whether that device will be reconfigured. This field can be toggled between 'YES' & 'NO' via the associated buttons.
	Device Column:	Indicates the device to be reconfigured.
	Status Column:	Indicates the status of the device. Currently the only two states supported are 'New' & 'Previously Configured'.
	Notes Column:	Indicates additional info about the device. New devices are required to go through reconfiguration to be imported into the configuration.
	Reconfig Button:	Starts the reconfiguration execution on all selected devices.
	Cancel Button:	Exists without reconfiguring.

#### 5.2.4.7 ReCalc Tests...

Use this command to re-calculate the parameters for a specific device in the current configuration in the Configuration View. This feature requires specific support in the modules auto configuration routine and is not supported by all test modules. The command & button is disabled if there are no devices in the current configuration that support this feature. The modules snippet file must enable this feature through the Device Property Dialog if the module supports this feature.

#### Shortcuts

Toolbar: 

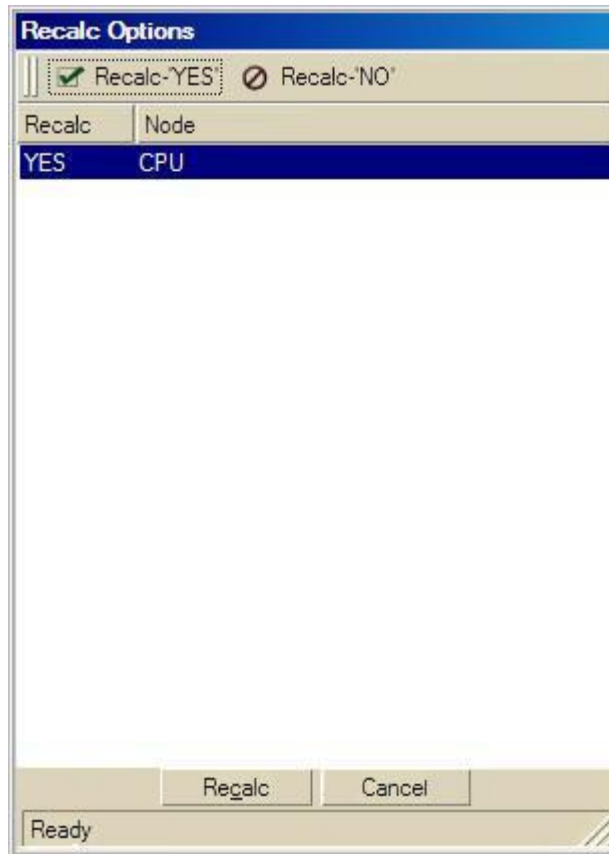
#### Recalc Options Dialog Box

Re-calculate is handy when you have a device that has a lot of parameters that need configured a lot to support various systems or SKUs. With this feature the user can update &/or reconfigure his system or SKU and automatically reconfigure the devices parameters without changing the current configuration or flow.

This recalculate function calls autoconfigure on the selected device with an additional parameter. So the module code needs to check for this parameter during autoconfig, thus it is up to the module whether it support this feature or not. Recalculate ONLY changes parameters &/or parameter groups, it should not change how the devices or tests are configured in either the Configuration or Sequence tree. If the package needs to support multiple SKUs or configurations and the operator want to keep the old configuration, then the operator should clone the configuration and then run Recalc... on the new configuration. **Re-Configure** only runs on the selected devices which also support the feature within the currently selected configuration tree. It does not create a

new configuration, it makes the changes to the current one. The following dialog is displayed, which allows the user to select the devices to recalculate parameters on.

### Dialog Control:



### Control Property Descriptions:



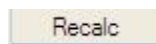
Reconfig-YES

Reconfig-YES: Changes the Reconfig Column of the Node(s) selected to YES.

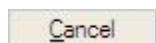


Reconfig-NO

Reconfig-NO: Changes the Reconfig Column of the Node(s) selected to NO.



Recalc Button: Starts the recalc execution on the selected device nodes.



Cancel Button: Exist without recalculating any device nodes.

#### 5.2.4.8 Run This

Use this command to start execution of the currently selected node in the Sequence View . The Sequence View will keep track of the run data in the tree view. The Test Message View will display any messages from tests, test messages are filtered so may not all be

displayed. The command & button are enabled only when a valid test flow node is selected and no tests are currently running.

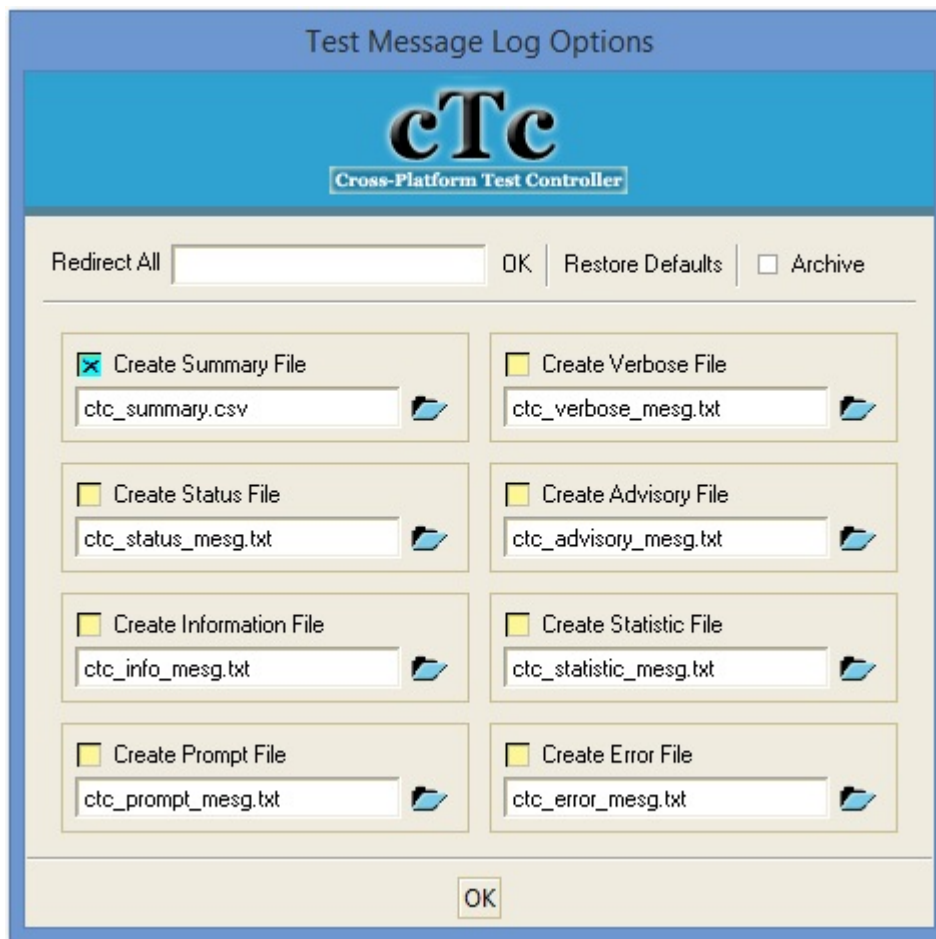
### Shortcuts

Keys: Ctl-U

#### 5.2.4.9 Log Options

The Message Redirection dialog displays controls for setting the message redirection options. It allows redirection of all the different message types to a specified file. They can be redirected to the same file, separate files or any combination. It also allows the summary file to be specified. By default, only the summary file redirection is enabled.

### Test Message Log Options Dialog



### Message Redirection Toolbar:

- Redirect All:** This edit control allows a redirection file name to be specified for all redirection files.
- OK:** Press this button to apply the redirection file name to all redirection files. Does not change the summary file name.
- Restore Defaults:** Press this button to apply default settings to all controls.

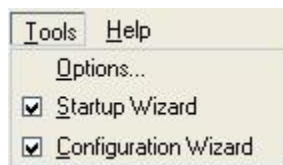


**Message Redirection Properties:**

- Create Summary File
- Redirect VERBOSE Messages
- Redirect STATUS Messages
- Redirect ADVISORY Messages
- Redirect INFO Messages
- Redirect STATISTIC Messages
- Redirect PROMPT Messages
- Redirect ERROR Messages
- Archive Files

**5.2.5 Tools menu**

The tools menu offers the following commands

**See the following for command details:**

Startup Wizard - This radio button allows the Startup Wizard dialog to be enabled or disabled when executing the CTC GUI.

Configuration Wizard - This radio button allows the Configuration Wizard to be enabled or disabled when opening a package file.

Options...

**5.2.5.1 Options**

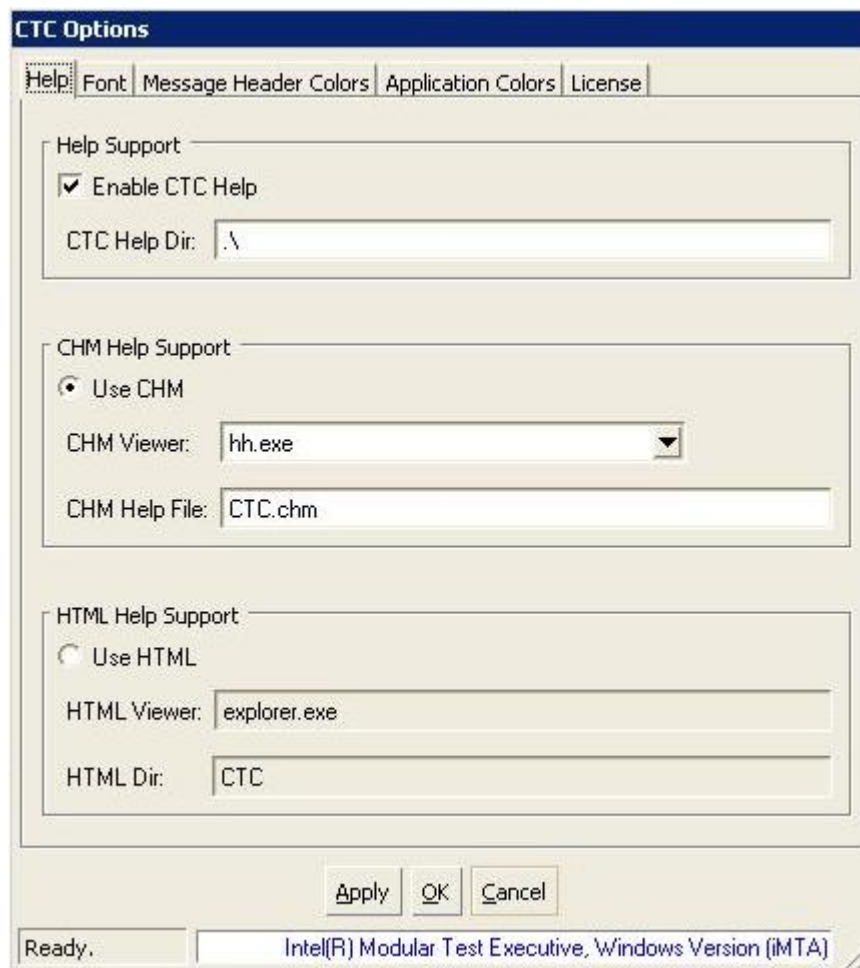
Use this command to configure CTC option properties. CTC presents a tab dialog box that contain various CTC GUI options.

**See the following for tab controls details:**

- Help
- Font
- Message Header Colors
- Application Colors

**5.2.5.1.1 Help Options**

Use this Options Tab to edit the help file properties. CTC presents the following tab view in the options dialog box.



### Property Definitions:

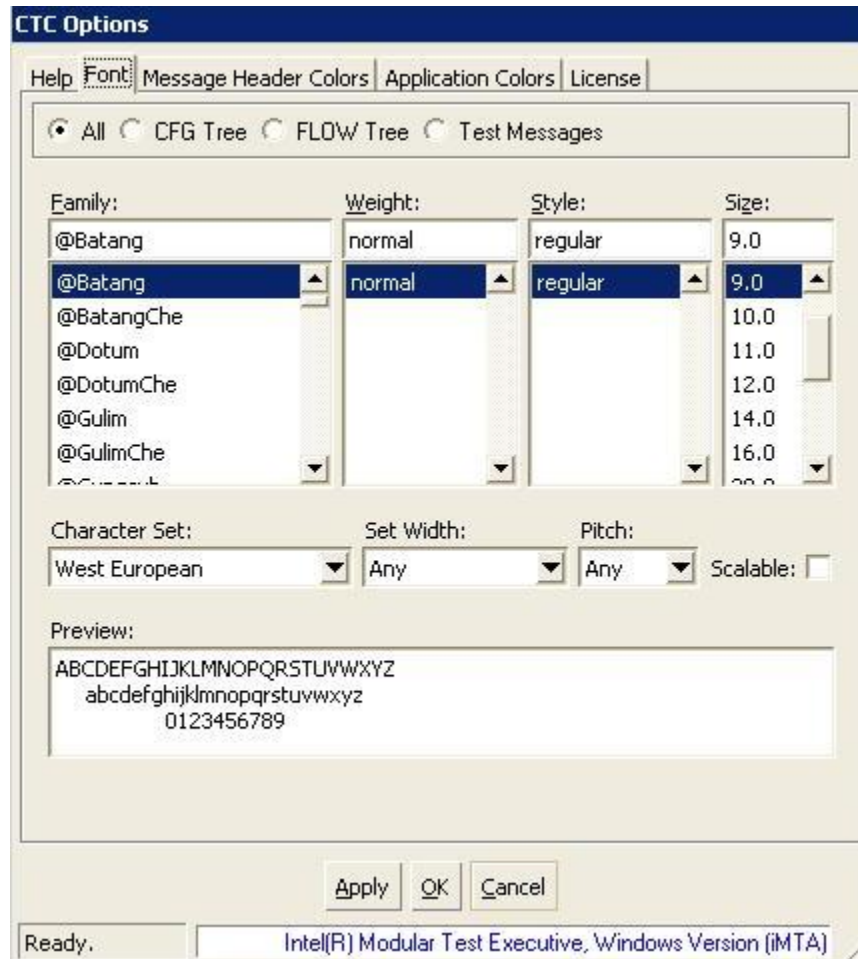
- CTC Help Dir:** Defines the global help file directory, all help file paths are relative to this one.
- Use CHM:** Selects CHM Help File Support, unselect's HTML support. Both SNX and PKX configuration files must support calling out the Test Module CHM file in the device's '\$HLPFILE' parameter. See the Help Control in the Device Property Sheet for more details.
- CHM Viewer:** Identifies the CHM Viewer to use. Windows defaults to use hh.exe and Linux defaults to use the xchm viewer. Viewers must be in the path for CTC to execute them.
- CHM Help File:** Identifies the CHM Help file for CTC, path must be relative to CTC Help Dir.
- Use HTML:** Selects HTML Help File Support, unselect's CHM support. Both SNX and PKX configuration files must support calling out the Test Module HLP directory containing the help htm files in the device's '\$HLPFILE' parameter. See the Help Control in the Device Property Sheet for more details.
- HTML Viewer:** Identifies the HTML Viewer to use. Windows defaults to using explorer and Linux defaults to using mozilla. The viewer can be

changed, but it must support the htm file to load as a command line parameter.

**HTML CTC Dir:** Identifies the HTML Help file directory containing the CTC htm help files, path must be relative to CTC Help Dir.

#### 5.2.5.1.2 Font Options

Use this Options Tab to edit the font properties. CTC presents the following tab view in the options dialog box.



#### Property Definitions:

**All:** Selects setting the font style for all window panes.  
**CFG Tree:** Selects setting the font style for the configuration view only.  
**FLOW Tree:** Selects setting the font style for the sequence/flow view only.  
**Test Messages:** Selects setting the font style for the Test Message View only.

**Family:** Selects Font Family style.  
**Weight:** Selects normal/bold font style.  
**Style:** Selects font style if applicable.  
**Size:** Selects font size.

<b>Character Set:</b>	Selects font character set style.
<b>Set Width:</b>	Selects font width
<b>Pitch:</b>	Selects font pitch
<b>Scalable:</b>	Sets scalable or non-scalable font type
<b>Preview:</b>	Displays a preview of the font style selected.

#### 5.2.5.1.3 Message Header Colors

Use this Options Tab to edit the Message Header Color properties. CTC presents the following tab view in the options dialog box.

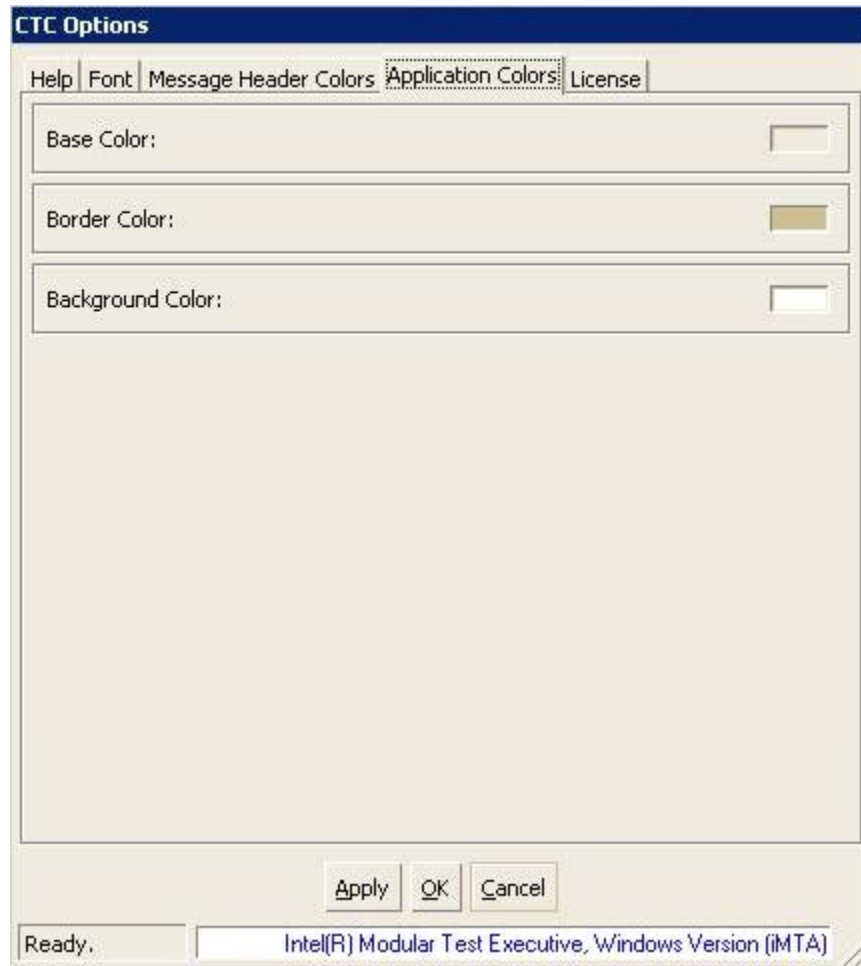


#### Property Definitions:

<b>Advisory Header:</b>	Selects color style for Advisor Messages.
<b>Status Header:</b>	Selects color style for Status Messages.
<b>Error Header:</b>	Selects color style for Error Messages.
<b>Verbose Header:</b>	Selects color style for Verbose Messages.
<b>Statistic Header:</b>	Selects color style for Statistic Messages.
<b>Info Header:</b>	Selects color style for Info Messages.

## 5.2.5.1.4 Application Colors

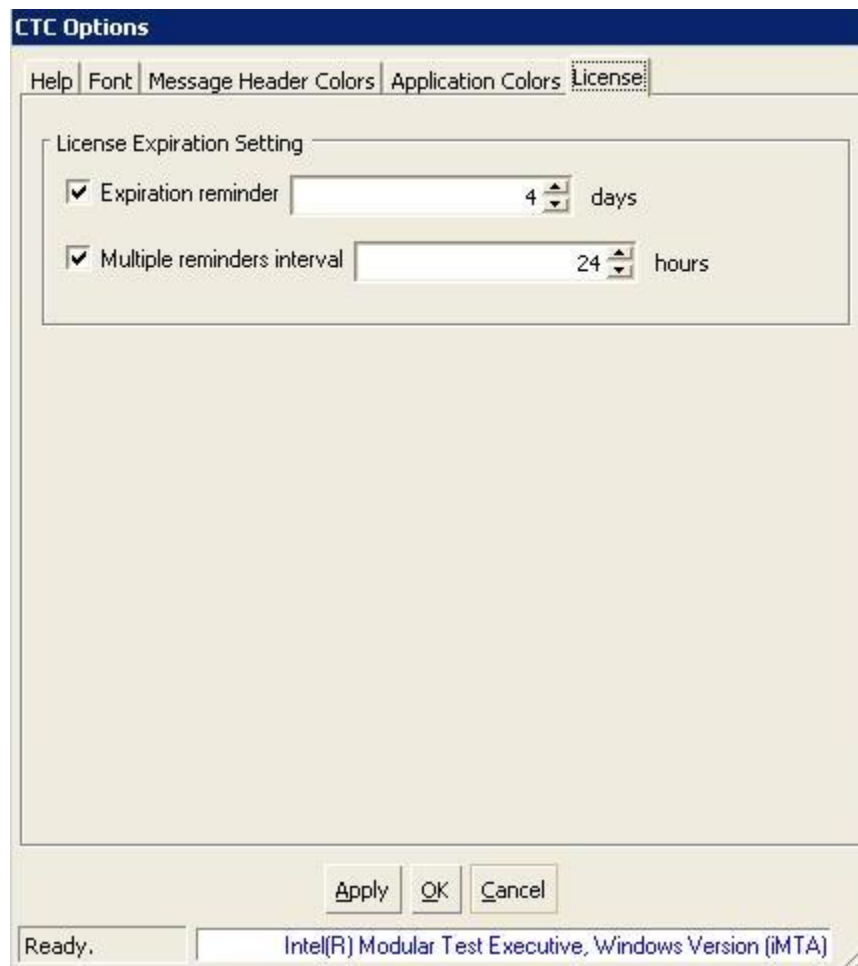
Use this Options Tab to edit the Application Color properties. CTC presents the following tab view in the options dialog box.

**Property Definitions:**

**Base Color:** Selects color style for the base window.  
**Border Color:** Selects color style for the window borders.  
**Background Color:** Selects color style for the window background.

## 5.2.5.1.5 License Options

Use this Options Tab to edit the License setting properties. CTC presents the following tab view in the options dialog box.

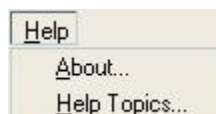


### Property Definitions:

<b>Expiration reminder:</b>	Enables Expiration reminder dialog message.
<b>Multiple Reminder interval:</b>	Enable multiple reminder.

## 5.2.6 Help menu

The Help menu offers the following commands

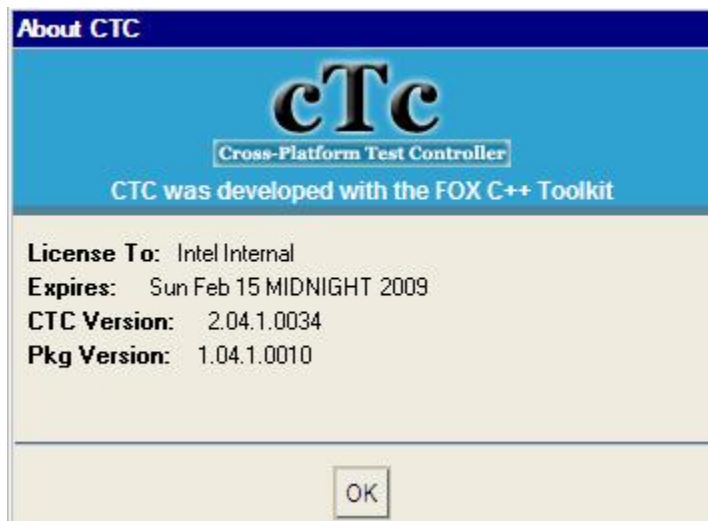


### See the following for command details:

About  
Help Topics

### 5.2.6.1 About

Presents the following information dialog box about CTC.



### 5.2.6.2 Help Topics

Use this command to view help file for CTC. The CTC Help file name and location is specified in the Help Options dialog under the tools menu.

## 5.3 Toolbar

The toolbar is displayed across the top of the CTC Window, below the menu bar. It provides quick mouse access common configuration functions.



### Buttons



Open a new document file.



Open an existing document. CTC displays the Open dialog box, in which you can locate and open the desired file.




Save the active document or template with its current name. If you have not named the document, CTC displays the Save As dialog box.




Clone (copy) the selected configuration or flow to a new configuration or flow.

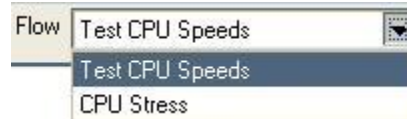


Delete the selected configuration or flow from the package file.

**Configuration Selection** is the first drop-down from the left. It controls the configuration selection of the Configuration Tree. CTC can support multiple configurations, so to select a configuration click the drop down  button to see the drop-down list, then click on the desired configuration name. The new configuration will appear in the Configuration View.



**Flow Selection** is the second drop-down from the left. It controls the flow selection of the Sequence Tree. CTC can support multiple flows per configuration, so to select a flow click the drop down  button to see the drop-down list, then click on the desired flow name. The new flow will appear in the Sequence Tree.



**Language Localization Selection** is the last drop-down. It only exists if there is a language localization file "gui\_translations.txt" in the same directory as CTC. It controls what language CTC displays in the GUI. This features only controls the CTC GUI text; this means that the configuration and flow trees which are controlled via the SNX/PKX files as well as the test messages are still in their native language.

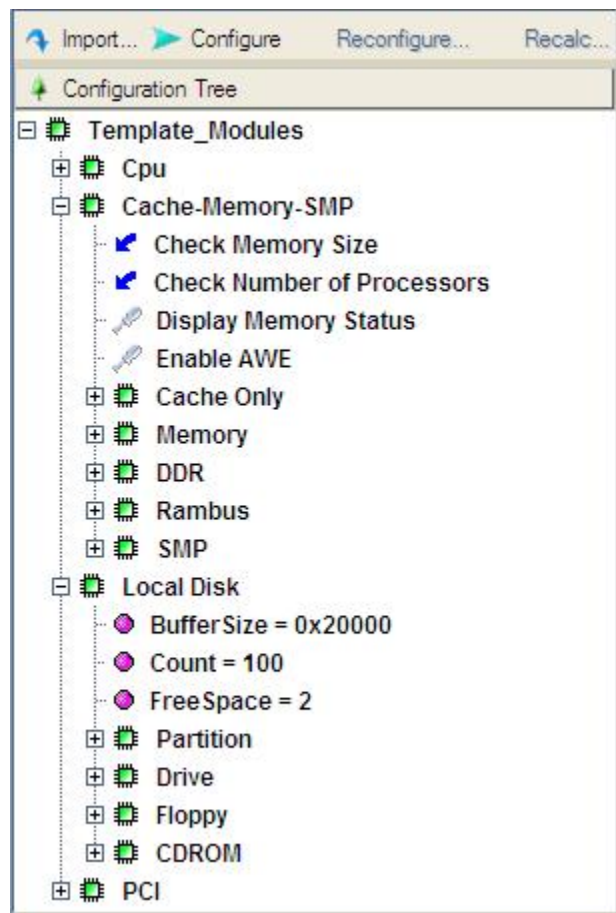


## 5.4 Configuration View









The Configuration View is the left pane of the CTC Window. It contains a hierarchical list of all available devices and tests. This view provides the capability to add, delete and edit devices, tests, and parameters in the currently selected configuration.

### Configuration Template View





### Icons Definitions

	Device a group of tests and parameters targeted at a specific subsystem or feature.
	Generic test, represents a subtest of a module
	Destructive test
	Interactive test
	Diagnostic tool (usually displays information)
	Group of parameters (no tests)
	Parameter, usually passed to a test
	Parameter Array, represents a two dimensional data array.

### Configuration View Tool Bar

Import Button...	Inserts new Test Module Devices from snippet files in the configuration template.
Configure Button	Configures the devices in the configuration template, disabled in all other configurations.
Reconfigure Button...	Brings up a Reconfigure Dialog to reconfigure devices in the configuration tree. Disabled in the configuration template.
Recalc Button...	Brings up a Recalculate Dialog to recalculate the device parameters in the configuration tree. Feature is only supported

Edit Control

on the devices that indicate they support it. This feature is enabled/disabled in the Device Property dialog control.  
<right-click> on any flow tree node brings up the Edit control dialog.

#### See Also:

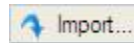
Edit menu  
Insert menu  
Action menu  
Configuration Tree  
Sequence View  
Icon Definitions

### 5.4.1 Import...

Use this command to import test module snippets into the Configuration Template. CTC presents the standard open file dialog box so you can search and select one or more snippet files from a directory. This command & button only available when the configuration template is selected.

#### Shortcuts

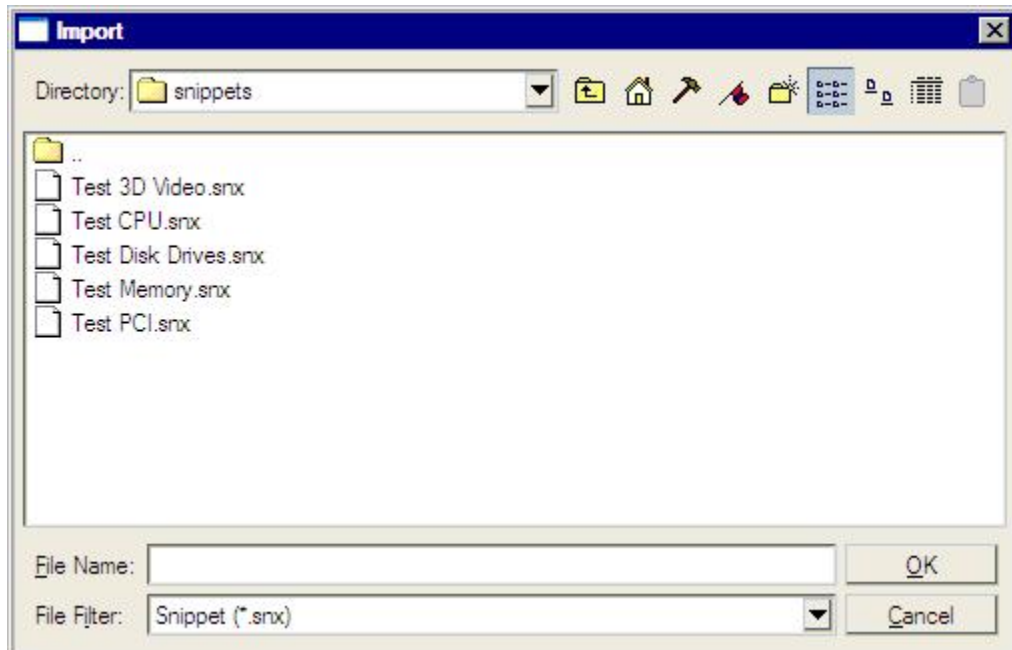
Toolbar:



Keys:

**F9**

#### Import Dialog



### 5.4.2 Configure...

Use this command to auto-configure a new configuration in the Configuration View. This command & button only available when the configuration template is selected.

## Shortcuts

Toolbar:  Configure  
 Keys: F8

## Configuration Tree Updates

Once autoconfigure is complete you can expect a new configuration with devices in the new configuration tree ready to test the current platform. In order to do this, modules are given the ability to programmatically alter the config. tree during the autoconfigure process. Below are the supported autoconfigure actions.

### Disable Tests

In certain situations a device may need to disable a test. For instance, the disk drive module disables all physical write tests because they are destructive. Tests which are disabled can be easily re-enabled by double clicking the test and clearing the disable check box in the test node property sheet.



The faded icons represent a disabled tests.

### Remove Tests

In some situations, tests need to be completely removed from the configuration tree because there is no way for the test to possibly run. For example, if the USB module detects the current system does not support USB2.0, it will remove all USB2.0 specific tests from the configuration tree.

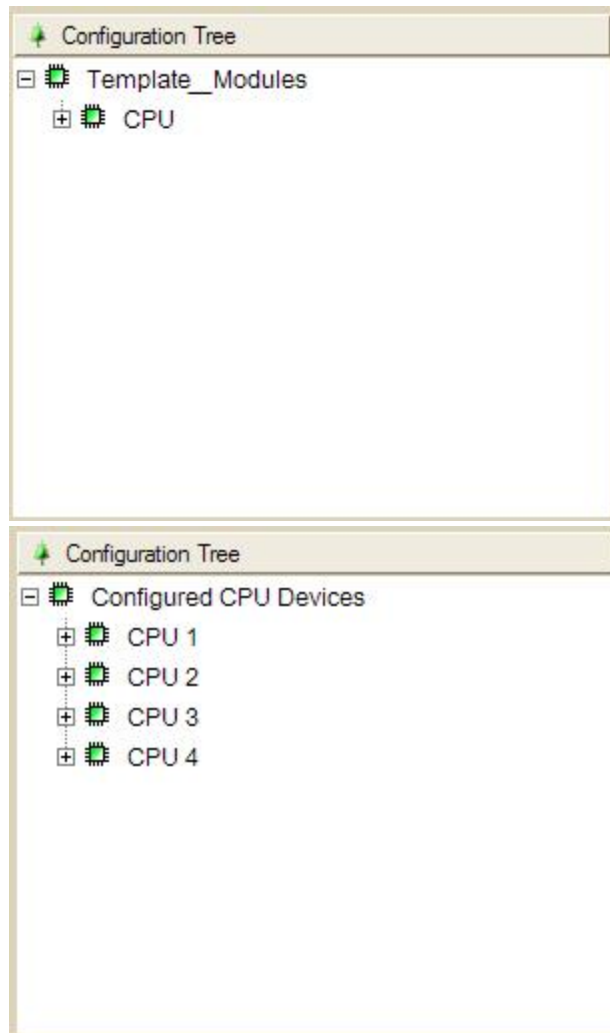
### Remove Devices

When a module's entire suite of tests are incapable of running, the entire device should be removed. For example, when the USB module detects that a system has no USB controllers, it removes itself completely from the configuration tree.

### Clone Devices

Modules that support an undetermined number of devices, like CPUs & hard drives, have the ability to clone and rename itself within the configuration. The first configuration represents the default Configuration Template. The second represents

the newly configured configuration tree autoconfigure on a system with four processors.



### Change Parameter Values

Devices within the template configuration usually have default parameter values. When a module probes the actual hardware, it will usually update the parameter values to accurately reflect the state of the hardware.

### Add New Parameters

New parameters may also show up in the configuration tree.

### Autoconfigure Messaging

Devices may display any of the normal test messages during autoconfigure, except Error Messages. It is recommended not to Fail or Error during auto configuration, the feature tests or device should be removed if probing fails. There are exceptions when something catastrophic occurs or the code detects a hardware device failure and wants to alert the

operator.

**Autoconfigure also generates a summary of the Devices and Tests that were changed during configuration. These include the following.**

**Subtests Removed During Autoconfigure (Name, Device, Number, Reason Removed)**

-->Hard Disk LED, Drive, 23, Requires IO Card  
-->AW SCSI LED, Drive, 25, Requires IO Card  
-->AW SCSI LED, Drive, 25, Requires IO Card  
-->IOC Disk Change, Floppy, 22, Requires IO Card

**Subtests Disabled During Autoconfigure (Name, Device, Number, Reason Disabled)**

-->Destructive Physical Rnd WRC, Drive 0 [IDE], 12, destructive

**Modules Removed During Autoconfigure (Name, Reason Removed)**

-->Partition, Clone Source  
-->Drive, Clone Source  
-->Floppy, No Floppy device detected.  
-->CDROM, Clone Source.

**Modules Removed Unexpectedly During Autoconfigure**

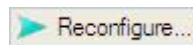
None

### 5.4.3 Reconfigure...

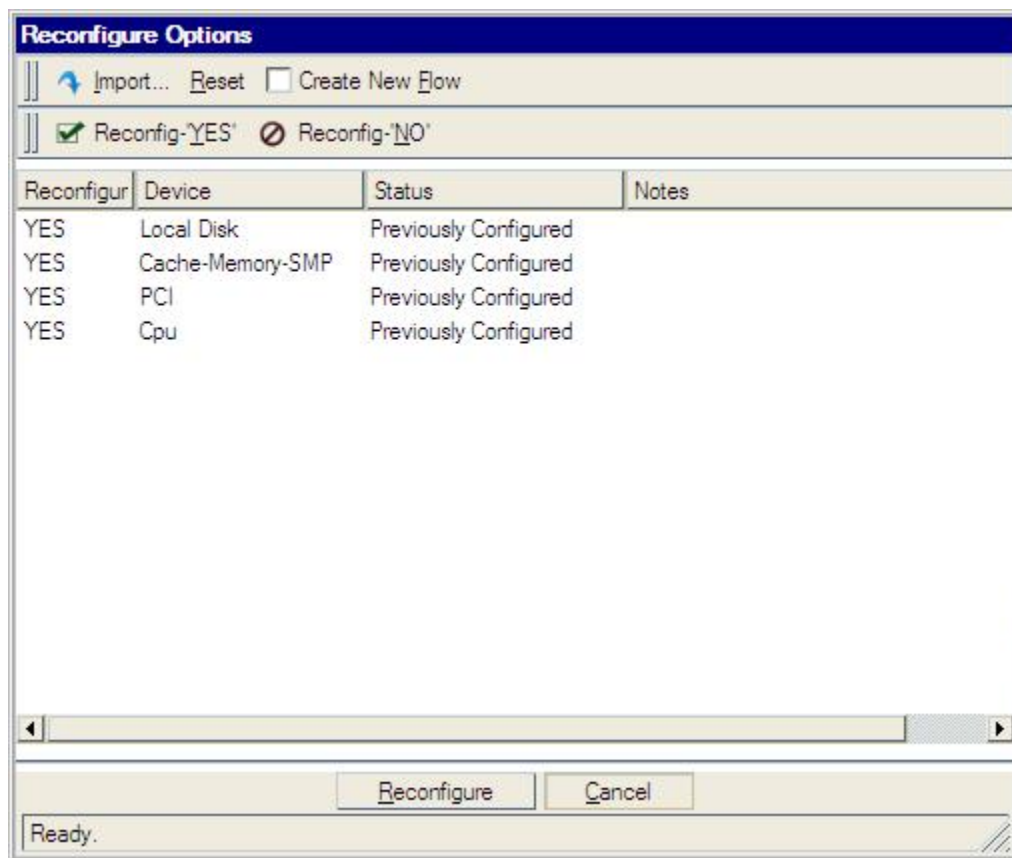
Use this command to re-configure the tests in the current configuration in the Configuration View. This command & button not available in the configuration template, this is due to the fact that it is not configured.

**Shortcuts**

Toolbar:



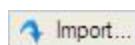
**Reconfigure Dialog Box**



Re-Configure is handy when you have a configuration with only a subset of the devices from the template configuration tree, or if your configuration's flow tree differs from the template flow. It is also useful when importing a new or updating a test module into your test package.

**Autoconfigure** copies all the devices from the template's configuration tree into a new configuration and runs the auto configuration process on them, it also copies the template's flow tree into the new configuration. **Re-Configure** only runs autoconfigure on selected devices within the currently selected configuration tree. It does not create a new configuration, it makes the changes to the current one. The following dialog is displayed, which allows the user to select the devices to reconfigure. It also allows the user to import new devices from snippet files on disk.

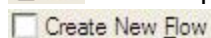
#### Control Property Descriptions:



Import Button: Brings up the Import Dialog box to add new devices to the list.



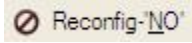





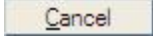
Reset Button: Resets the list back to original state.



New Flow: Check box to create a new flow with reconfiguration changes.



Reconfig-Yes: Changes the Reconfig Column of the list item(s) selected to YES.

	Reconfig-No:	Changes the Reconfig Column of the list item(s) selected to NO.
	Reconfig Column:	Indicates whether that device will be reconfigured. This field can be toggled between 'YES' & 'NO' via the associated buttons.
	Device Column:	Indicates the device to be reconfigured.
	Status Column:	Indicates the status of the device. Currently the only two states supported are 'New' & 'Previously Configured'.
	Notes Column:	Indicates additional info about the device. New devices are required to go through reconfiguration to be imported into the configuration.
	Reconfig Button:	Starts the reconfiguration execution on all selected devices.
	Cancel Button:	Exists without reconfiguring.

#### 5.4.4 Recalc...

Use this command to re-calculate the parameters for a specific device in the current configuration in the Configuration View. This feature requires specific support in the modules auto configuration routine and is not supported by all test modules. The command & button is disabled if there are no devices in the current configuration that support this feature. The modules snippet file must enable this feature through the Device Property Dialog if the module supports this feature.

##### Shortcuts

Toolbar: 

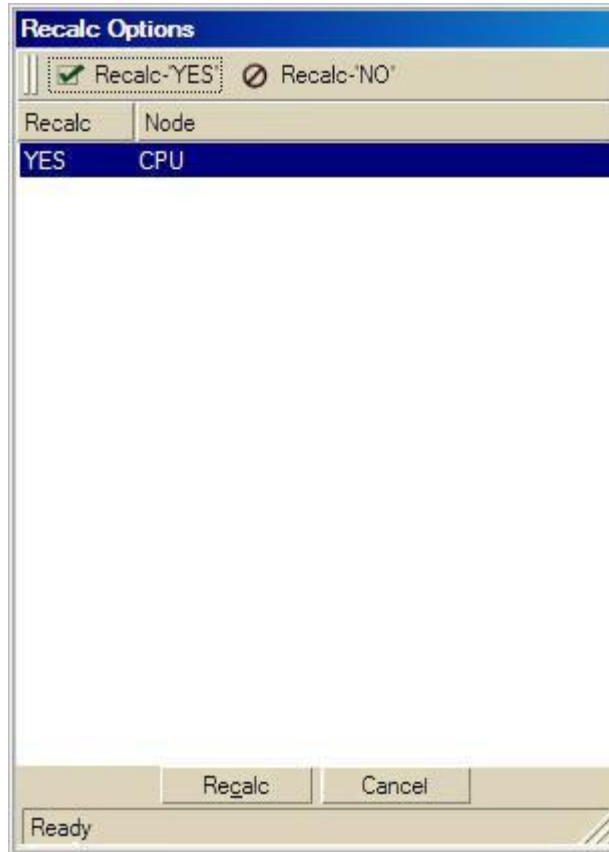
##### Recalc Options Dialog Box

Re-calculate is handy when you have a device that has a lot of parameters that need configured a lot to support various systems or SKUs. With this feature the user can update &/or reconfigure his system or SKU and automatically reconfigure the devices parameters without changing the current configuration or flow.

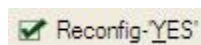
This recalculate function calls autoconfigure on the selected device with an additional parameter. So the module code needs to check for this parameter during autoconfig, thus it is up to the module whether it support this feature or not. Recalculate ONLY changes parameters &/or parameter groups, it should not change how the devices or tests are configured in either the Configuration or Sequence tree. If the package needs to support multiple SKUs or configurations and the operator want to keep the old configuration, then the operator should clone the configuration and then run Recalc... on the new configuration. **Re-Configure** only runs on the selected devices which also support the feature within the currently selected configuration tree. It does not create a

new configuration, it makes the changes to the current one. The following dialog is displayed, which allows the user to select the devices to recalculate parameters on.

#### Dialog Control:



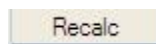
#### Control Property Descriptions:



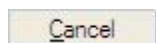
Reconfig-YES: Changes the Reconfig Column of the Node(s) selected to YES.



Reconfig-NO: Changes the Reconfig Column of the Node(s) selected to NO.



Recalc Button: Starts the recalc execution on the selected device nodes.



Cancel Button: Exist without recalculating any device nodes.

### 5.4.5 Node Help

Use this Edit Control (right-click) command to view help file for Test Module of the selected node. The help file name can be viewed & changed in the parent Device's property sheet.



## Shortcuts

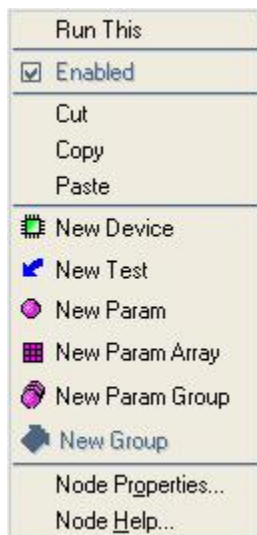
Keys: F1

## See Also

Help Options  
\$HELPPFILE

### 5.4.6 Node Edit Control

The Edit Control can be accessed by <right-click> on any node in either the Configuration View or the Sequence View.













**See the following for command details:**

Run This  
Enabled  
Cut  
Copy  
Paste  
New Device  
New Test  
New Param  
New Param Array  
New Param Group  
New Group  
Node Properties...  
Node Help...







### 5.4.7 Icon Definitions

Flow Icons indicate the current or last execution status of flow groups and tests. While tests are running flow groups and tests rotate through a sequence of icons to show that the tests are running.









### Idle Icons

	Device, group of tests and parameters
	Serial group of flow groups, tests and parameters.
	Parallel group of flow groups, tests and parameters.
	Generic test, represents a subtest of a module
	Destructive test
	Interactive test
	Diagnostic tool (usually displays information)
	Group of parameters (no tests)
	Parameter, usually passed to a test
	Parameter Array, represents a two dimensional data array.









### Running Icons

	Sequence of a running serial group that is passing.
	Sequence of a running parallel group that is passing.
	Sequence of a running test that is passing.
	Sequence of a running serial group that has failed.
	Sequence of a running parallel group that has failed.
	Sequence of a running test that is has failed.

### Enabled Test Icons



	Generic test
	Destructive test
	Interactive test
	Destructive Interactive test
	Diagnostic tool or utility
	Destructive Diagnostic tool or utility
	Interactive Diagnostic tool or utility
	Destructive Interactive Diagnostic tool or utility

### Disabled Test Icons





	Disabled Generic test
	Disabled Destructive test
	Disabled Interactive test
	Disabled Destructive Interactive test
	Disabled Diagnostic tool or utility
	Disabled Destructive Diagnostic tool or utility
	Disabled Interactive Diagnostic tool or utility
	Disabled Destructive Interactive Diagnostic tool or utility

## 5.5 Sequence View










The Sequence View is the upper right pane of the CTC window. It contains a hierarchical tree of flow groups, tests and parameters.

The icons for flow groups represent serial  and parallel  test flows. Flow groups are created by using Add Group from the Edit menu or New Group from the Edit Control. Tests are added to flow groups by dragging them from the Configuration View and dropping them on a flow group.

### Sequence View

 Run  Stop  Clear  Auto-scroll						
Flow Tree	Progress %	Pass	Fail	State	Activity	
Test Flow						
Cpu						
Merom Instructions	100	1	0	IDLE	0x00000003	
CPU Features Check	100	0	1	IDLE	0x00000003	
FPU	100	1	0	IDLE	0x00000003	
MMX	100	1	0	IDLE	0x0000000C	
FPUStress	100	1	0	IDLE	0x0000000D	
SIMD Instructions	100	1	0	IDLE	0x0000000C	
CPU Speed	100	0	1	IDLE	0x00000004	
Willamette Features	100	1	0	IDLE	0x00000003	
Bus Speed	0	0	0	RUNNING	0x00000000	
SSE3	0	0	0	IDLE	0x00000000	
PPVTest 1	0	0	0	IDLE	0x00000000	
PPVTest 2	0	0	0	IDLE	0x00000000	
PPVTest 3	0	0	0	IDLE	0x00000000	
PPVTest 4	0	0	0	IDLE	0x00000000	

### Icons

	Serial group of flow groups, tests and parameters.
	Parallel group of flow groups, tests and parameters.
	Generic test, represents a subtest of a module
	Destructive test
	Interactive test
	Diagnostic tool (usually displays information)
	Group of parameters (no tests)
	Parameter, usually passed to a test
	Parameter Array, represents a two dimensional data array.

## Sequence View Tool Bar

<b>Run Button</b>	Start test execution from the root of the flow tree
<b>Stop Button</b>	Stop all test execution, all running test set to ABORTED state.
<b>Clear Button</b>	Clears all the test execution status, states and messages.
<b>Edit Control</b>	<right-click> on any flow tree node brings up the Edit control dialog.
<b>Auto-Scroll</b>	Sets the Sequence View to auto-scroll to keep currently executing tests inside the viewable window pane.

## Sequence List Properties

<b>Sequence Tree:</b>	Displays all the flow groups, tests and parameters in the Flow Tree. Also displays the activity status of executed groups and tests by icon color states.
<b>Progress:</b>	Displays the current progress percentage (0%-100%) of the test
<b>Pass:</b>	Displays the number of executions that the test passed.
<b>Fail:</b>	Displays the number of executions that the test failed.
<b>State:</b>	Displays the current execution state of the test, IDLE, RUNNING, ABORTED.
<b>Activity:</b>	Displays the last activity state the test sent. The first word is the number of test library calls made. The second word is up to the module developer.
<b>Timestamp:</b>	Displays the last or current execution time of a test. (0, 00:00:00) Days, Hours:Minutes:Seconds

### See Also:

Edit menu  
 Insert menu  
 Action menu  
 Configuration Tree  
 Configuration View  
 Icon Definitions

### 5.5.1 Node Help

Use this Edit Control (right-click) command to view help file for Test Module of the selected node. The help file name can be viewed & changed in the parent Device's property sheet.

#### Shortcuts

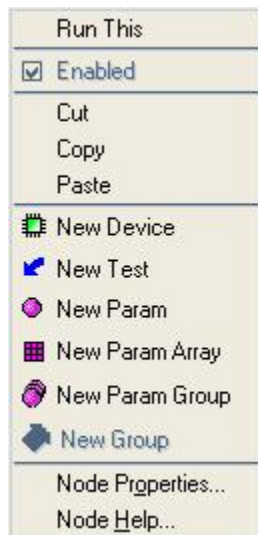
Keys: F1

#### See Also

Help Options  
 \$HELPPFILE

### 5.5.2 Edit Control

The Edit Control can be accessed by <right-click> on any node in either the Configuration View or the Sequence View.












**See the following for command details:**

Run This  
 Enabled  
 Cut  
 Copy  
 Paste  
 New Device  
 New Test  
 New Param  
 New Param Array  
 New Param Group  
 New Group  
 Node Properties...  
 Node Help...

## 5.6 Icon Definitions

Flow Icons indicate the current or last execution status of flow groups and tests. While tests are running flow groups and tests rotate through a sequence of icons to show that the tests are running.

### Idle Icons

-  Device, group of tests and parameters
-  Serial group of flow groups, tests and parameters.
-  Parallel group of flow groups, tests and parameters.
-  Generic test, represents a subtest of a module
-  Destructive test
-  Interactive test
-  Diagnostic tool (usually displays information)
-  Group of parameters (no tests)
-  Parameter, usually passed to a test



Parameter Array, represents a two dimensional data array.

### Running Icons



Sequence of a running serial group that is passing.



Sequence of a running parallel group that is passing.



Sequence of a running test that is passing.



Sequence of a running serial group that has failed.



Sequence of a running parallel group that has failed.



Sequence of a running test that is has failed.

### Enabled Test Icons



Generic test



Destructive test



Interactive test



Destructive Interactive test



Diagnostic tool or utility



Destructive Diagnostic tool or utility



Interactive Diagnostic tool or utility



Destructive Interactive Diagnostic tool or utility

### Disabled Test Icons



Disabled Generic test



Disabled Destructive test



Disabled Interactive test



Disabled Destructive Interactive test



Disabled Diagnostic tool or utility



Disabled Destructive Diagnostic tool or utility



Disabled Interactive Diagnostic tool or utility

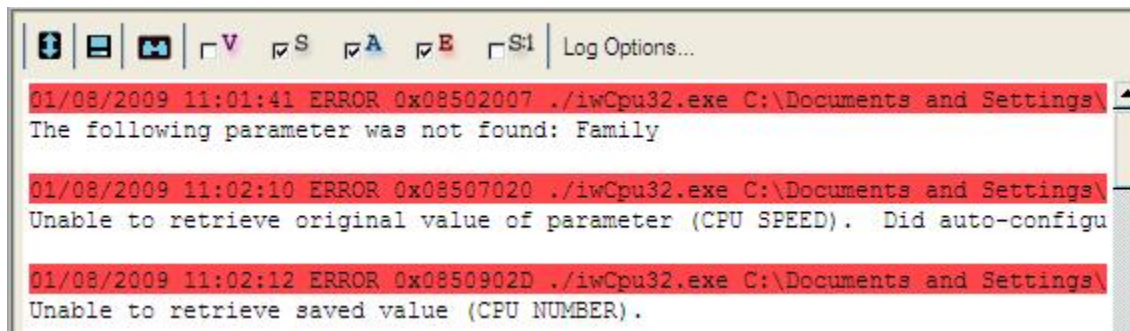


Disabled Destructive Interactive Diagnostic tool or utility

## 5.7 Test Message View

The Test Messages View is the bottom pane of the CTC Window. It displays the test messages being sent by the running tests. By default, it displays all info, prompt, & error messages sent from tests. This default can be changed by selecting the desired filter from the View Menu on the menu bar, or using the buttons on the Toolbar.

### Test Messages View



### Test Messages View Tool Bar

	Maximize Button	Maximize or Minimize the Test Message View.
	Save Msg Button	Save all the test messages in the Test Message Tab to a file.
	Search Button	Search all the test messages for a specified string.
	Verbose Filter	Verbose message filter, displays verbose message when checked.
	Status Filter	Status message filter, displays verbose message when checked.
	Advisory Filter	Advisory message filter, displays verbose message when checked.
	Error Filter	Error message filter, displays verbose message when checked.
	Statistic Filter	Statistic message filter, displays statistic message when checked.
	Log Options...	Brings up the Log Options Dialog Control. Enables the redirection of test messages to log files.

### Shortcuts

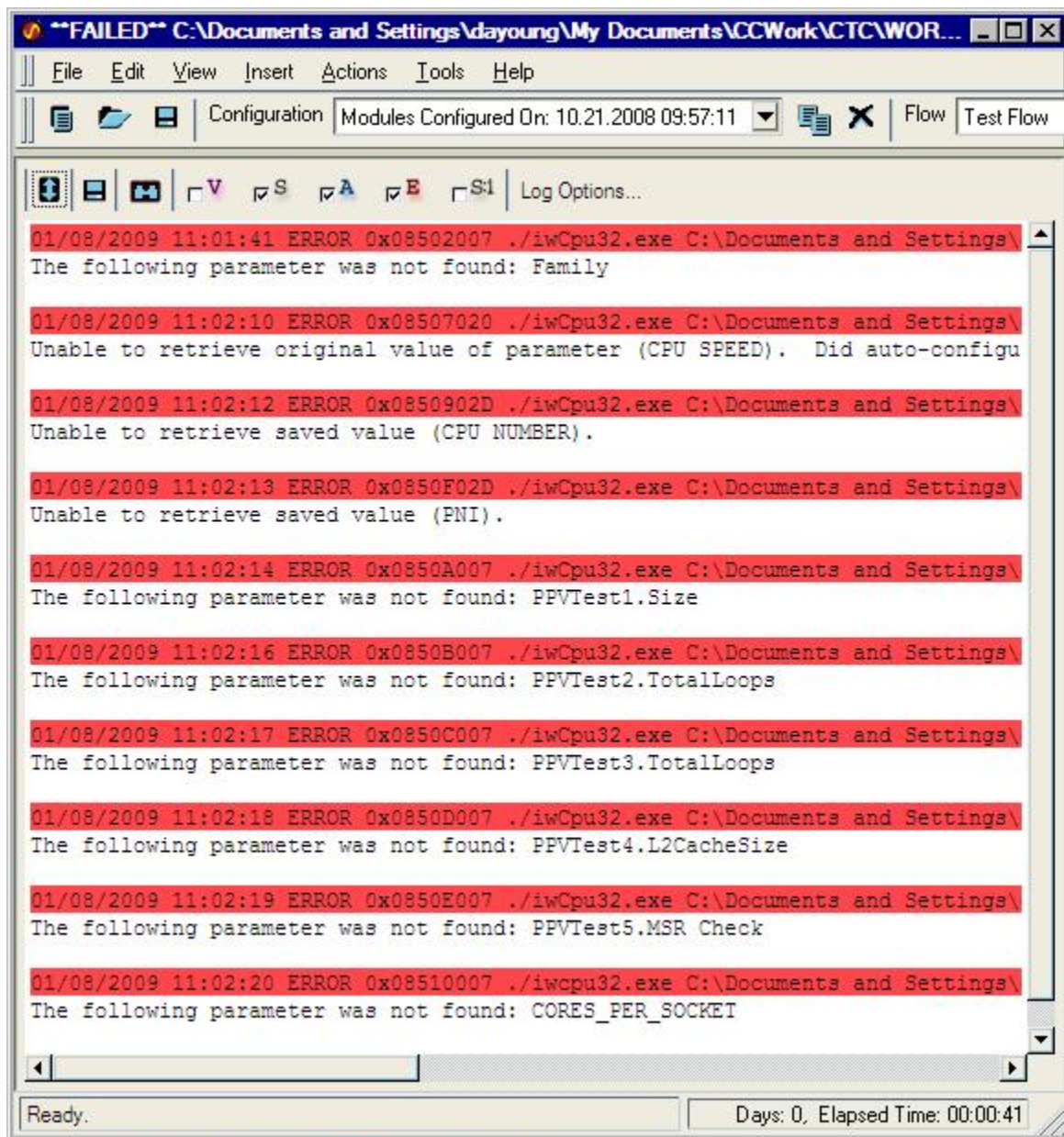
Keys:      Ctrl+C   Copies text selected to the clipboard.

## 5.7.1 Maximize Message Window

The Maximize Message Button will maximize or minimize the Test Message View window. The button icon is located on the Test Message View toolbar.


### Maximized Test Message Tab





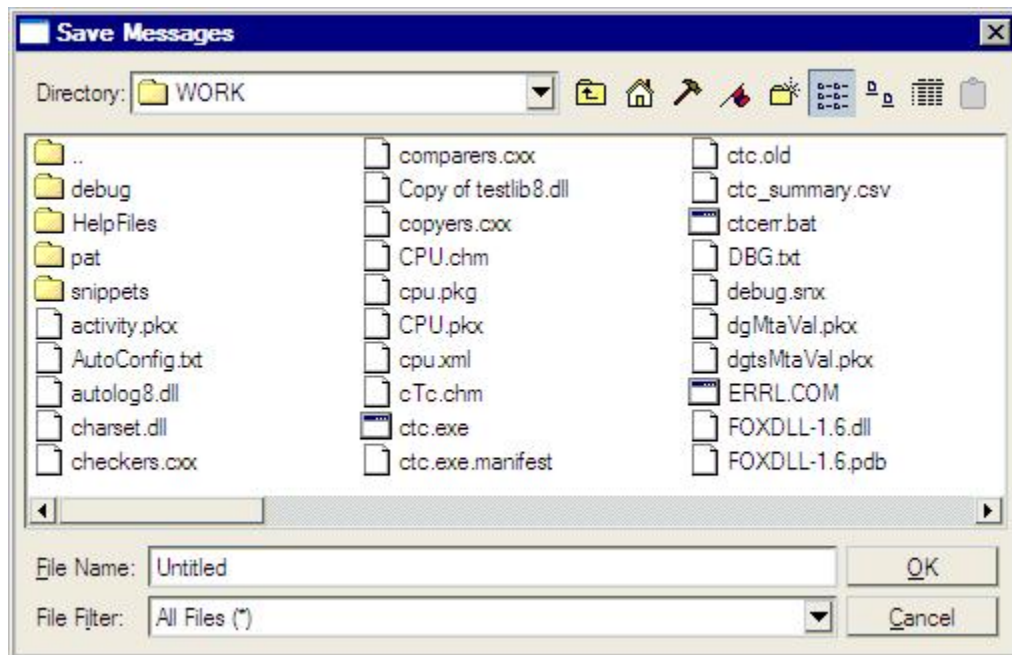
### 5.7.2 Save Messages Button

The Save Message Button will bring up the standard file save dialog box so you can select or name a new file to save your messages to.

The button icon  is located on the Test Message View toolbar.


#### Save Messages Dialog



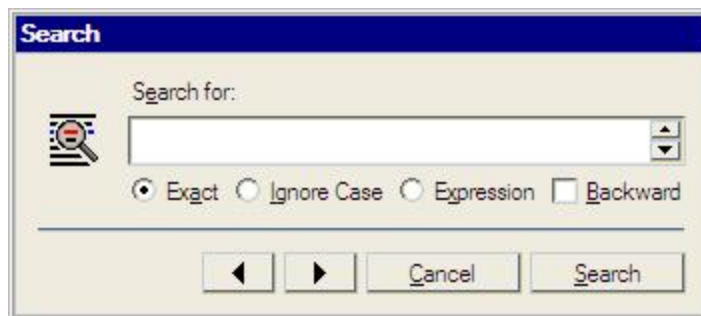


### 5.7.3 Search Messages Button



The Search Message Button will bring up search dialog box so you can select a string to search for, as well as specify the search parameters.

The button icon  is located on the Test Message View toolbar.

### Search Messages Dialog



### Search Messages Properties

<b>Search for:</b>	Edit Control where you can type a new search string or select from a list of previous search strings.
<b>Exact:</b>	Select this radio button if you want to search for the exact word.
<b>Ignore Case:</b>	Select this radio button if you want to ignore Upper and Lower case characters in the search string.
<b>Expression:</b>	Select this radio button if the search string is an expression.
<b>Backward:</b>	Select this radio button if you want to search backward through the messages. Default direction is forward when unselected.
	Search backward through the messages for the next match.
	Search forward through the messages for the next match.

**Cancel:** Exit the dialog without further searching.  
**Search:** Begin searching through the messages for the specified string.

#### 5.7.4 Error Message

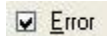
A check mark next to this item indicates that error messages will be displayed in the Test Messages View. The check mark is toggled by selecting the item or selecting the same check box control in the View menu.

##### Shortcuts

Toolbar:



View Menu:



#### 5.7.5 Advisory Message

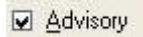
A check mark next to this item indicates that advisory messages will be displayed in the Test Messages View. The check mark is toggled by selecting the item or selecting the same check box control in the View menu.

##### Shortcuts

Toolbar:



View Menu:



#### 5.7.6 Status Messages

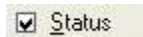
A check mark next to this item indicates that status messages will be displayed in the Test Messages View. The check mark is toggled by selecting the item or selecting the same check box control in the View menu.

##### Shortcuts

Toolbar:



View Menu:



#### 5.7.7 Statistic Message

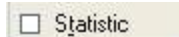
A check mark next to this item indicates that statistic messages will be displayed in the Test Messages View. The check mark is toggled by selecting the item or selecting the same check box control in the View menu.

##### Shortcuts

Toolbar:



View Menu:



### 5.7.8 Verbose Debug Messages

A check mark next to this item indicates that status messages will be displayed in the Test Messages View. The check mark is toggled by selecting the item or selecting the same check box control in the View menu.

#### Shortcuts

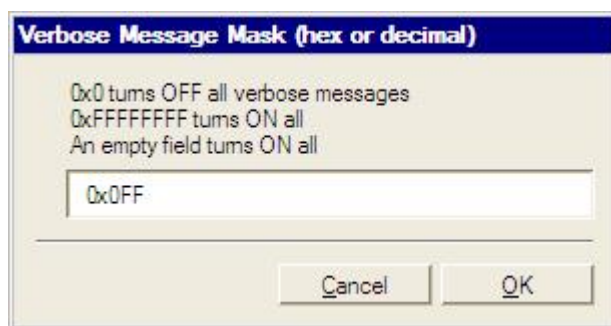
Toolbar:



View Menu:



When the item is selected from the menu a dialog box is activated to allow the setting of the "verbose mask".



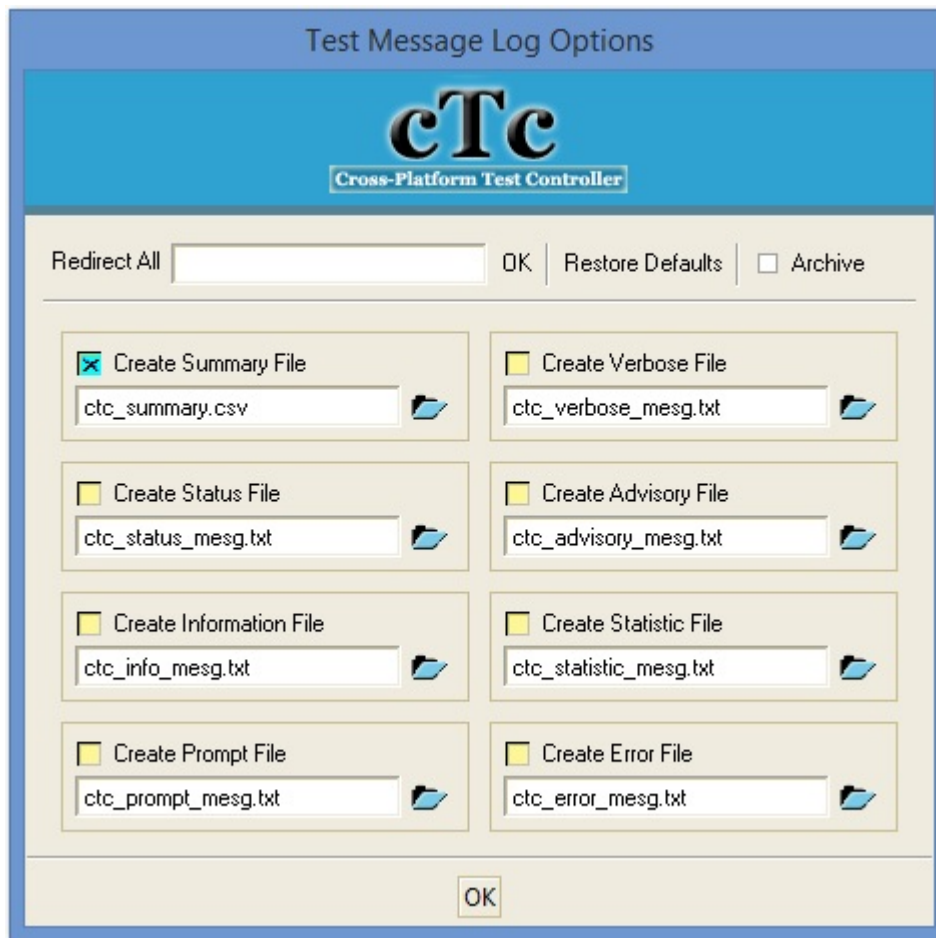
The verbose mask is used to filter the debug messages for a test. A verbose mask of 0 indicates no messages, hence this is the default. You must refer to the documentation of a specific module to learn what debugging messages are available for other settings of the mask value.

The idea of the mask is that each bit in the mask enables a different message or level of messaging. This allows a trouble shooter to select exactly which messages are desired and hence filter out any unnecessary messages.

### 5.7.9 Log Options

The Message Redirection dialog displays controls for setting the message redirection options. It allows redirection of all the different message types to a specified file. They can be redirected to the same file, separate files or any combination. It also allows the summary file to be specified. By default, only the summary file redirection is enabled.

#### Test Message Log Options Dialog



### Message Redirection Toolbar:

- Redirect All:** This edit control allows a redirection file name to be specified for all redirection files.
- OK:** Press this button to apply the redirection file name to all redirection files. Does not change the summary file name.
- Restore Defaults:** Press this button to apply default settings to all controls.

### Message Redirection Properties:

Create Summary File  
 Redirect VERBOSE Messages  
 Redirect STATUS Messages  
 Redirect ADVISORY Messages  
 Redirect INFO Messages  
 Redirect STATISTIC Messages  
 Redirect PROMPT Messages  
 Redirect ERROR Messages  
 Archive Files

### 5.7.9.1 Summary File

The Summary File check box and edit control are located in the Test Message Log Options. The check box enable/disables summary file redirection and the edit control selects the filename.

☒ Create Summary File:



This control toggles the Enable/Disable of the summary file. Summary file is enabled by default.

Edit control to specify the filename of the summary file.

 File Button:

This button brings up the standard file selection dialog box, it allows each file name to be selected from the directory structure.

### 5.7.9.2 Verbose Redirection

The Verbose Redirection check box and edit control are located in the Test Message Log Options. The check box enable/disables verbose redirection and the edit control selects the filename.

☐ Create VERBOSE File:



This control toggles the Enable/Disable redirection of the verbose messages.

Edit control to specify the filename of the verbose file.




This button brings up the standard file selection dialog box, it allows each file name to be selected from the directory structure.

### 5.7.9.3 Status Redirection

The Status Redirection check box and edit control are located in the Test Message Log Options. The check box enable/disables status redirection and the edit control selects the filename.

☐ Create Status File:



This control toggles the Enable/Disable redirection of the status messages.


Edit control to specify the filename of the status file.



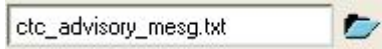
This button brings up the standard file selection dialog box, it allows each file name to be selected from the directory structure.

#### 5.7.9.4 Advisory Redirection

The Advisory Redirection check box and edit control are located in the Test Message Log Options. The check box enable/disables advisory redirection and the edit control selects the filename.

 Create Advisory File:

This control toggles the Enable/Disable redirection of the advisory messages.




Edit control to specify the filename of the advisory file.



This button brings up the standard file selection dialog box, it allows each file name to be selected from the directory structure.

#### 5.7.9.5 Info Redirection

The Info Redirection check box and edit control are located in the Test Message Log Options. The check box enable/disables info redirection and the edit control selects the filename.

 Create INFO File:

This control toggles the Enable/Disable redirection of the info messages.



Edit control to specify the filename of the info file.



This button brings up the standard file selection dialog box, it allows each file name to be selected from the directory structure.

#### 5.7.9.6 Statistic Redirection

The Statistic Redirection check box and edit control are located in the Test Message Log Options. The check box enable/disables statistic redirection and the edit control selects the filename.

 Create STATISTIC File:

This control toggles the Enable/Disable redirection of the statistic messages.



Edit control to specify the filename of the statistic file.



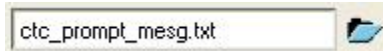
This button brings up the standard file selection dialog box, it allows each file name to be selected from the directory structure.

#### 5.7.9.7 Prompt Redirection

The Prompt Redirection check box and edit control are located in the Test Message Log Options. The check box enable/disables prompt redirection and the edit control selects the filename.

 Create Prompt File:

This control toggles the Enable/Disable redirection of the prompt messages.



Edit control to specify the filename of the prompt file.



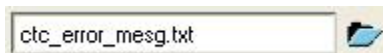
This button brings up the standard file selection dialog box, it allows each file name to be selected from the directory structure.

#### 5.7.9.8 Error Redirection

The Error Redirection check box and edit control are located in the Test Message Log Options. The check box enable/disables error file redirection and the edit control selects the filename.

 Create ERROR File:

This control toggles the Enable/Disable redirection of the error messages.



Edit control to specify the filename of the error file.




This button brings up the standard file selection dialog box, it allows each file name to be selected from the directory structure.


#### 5.7.9.9 Archive Files

The Archive check box is located in the Test Message Log Options. The check box enable/disables the archive files option.

If the archive files option is enabled, a directory with timestamp will be created automatically each time the CTC "RUN" button is hit. All test messages redirection files will be generated inside this timestamp folder.

 Archive

This control toggles the Enable/Disable of the archive files option.

 09\_30\_2015\_16\_34\_00

The timestamp directory will be generated automatically.

All test messages redirection files will be generated inside the timestamp directory.

Release 09_30_2015_16_34_00				
Name	Date modified	Type	Size	
ctc_advisory_mesg.txt	9/30/2015 16:34	TXT File	1 KB	
ctc_error_mesg.txt	9/30/2015 16:34	TXT File	1 KB	
ctc_info_mesg.txt	9/30/2015 16:34	TXT File	1 KB	
ctc_prompt_mesg.txt	9/30/2015 16:34	TXT File	1 KB	
ctc_statistic_mesg.txt	9/30/2015 16:34	TXT File	1 KB	
ctc_status_mesg.txt	9/30/2015 16:34	TXT File	13 KB	
ctc_verbose_mesg.txt	9/30/2015 16:34	TXT File	7 KB	
sum_20150930163106.txt	9/30/2015 16:34	TXT File	14 KB	

## 5.8 Property Sheets

CTC supports a property sheet for each type of node found in the trees. Each of the sheets will provide viewing access to the properties of the nodes, but edit access is restricted in some of the trees.

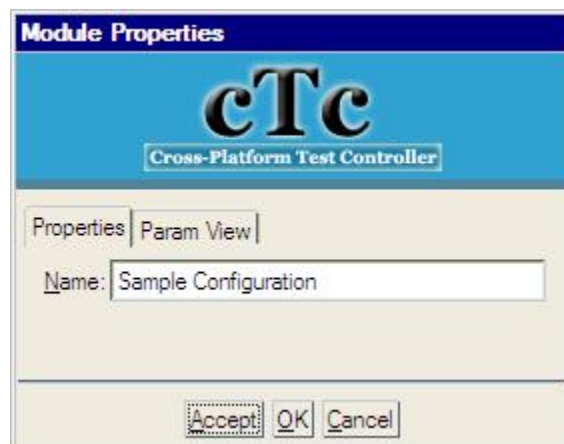
- Configuration Property Sheet
- Device Property Sheet
- Test Property Sheet
- Test Group Property Sheet
- Parameter Group Property Sheet
- Parameter Property Sheet
- Parameter Array Property Sheet

### 5.8.1 Configuration Property Sheet

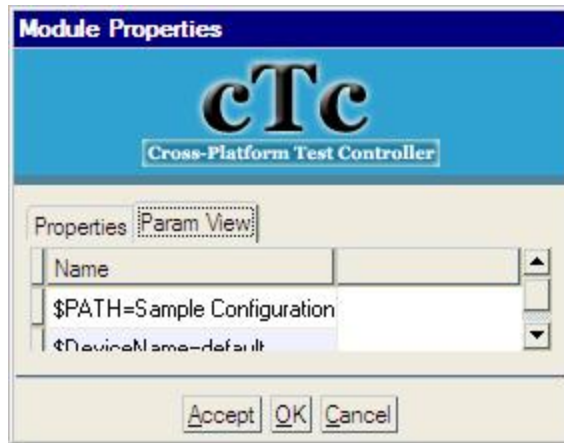
The configuration property sheet displays the name of the Configuration Tree. This property sheet allows editing of the configuration name.

The property sheet is modeless and will dynamically update when the focus changes in the one of the view panes.

#### Configuration Properties:





**Configuration Parameter Properties:****5.8.2 Test Property Sheet**

The test property sheet displays all the information relevant to a Test. This property sheet is also used to edit and add tests in a snippet file. When viewing the properties of a test in a package file some of the edit controls are disabled; **Destructive, Interactive, Utility, Autoconfigure & Recalc Parameters**. These are pre-defined capabilities based on the test & its features and should be defined via the snippet file.

The property sheet is modeless and will dynamically update when the focus changes in the one of the view panes.

**Snippet Properties Tab:**

**Test Properties**

**cTc**  
Cross-Platform Test Controller

Properties: Param View

Name: CPU Features Check

Binary: ./iwCpu32.exe

Number: 2

Timeout: 300

☐ Destructive    ☐ Interactive    ☐ Utility

☐ Disabled    ☐ Requires:

☒ Supports Autoconfigure

☐ Supports Autocalc Parameters

Accept OK Cancel

### Property Definitions:

- Name:** Name of the Test.
- Binary:** Name of the Test Module binary file to execute.
- Number:** Test Number passed to the Test Module on execution.
- Timeout:** Indicates the number of seconds CTC will wait in between heart beats from the test before determining the test is hung and terminating the test with an error.
- Destructive:** Indicates the test is a destructive test.
- Interactive:** Indicates the test is interactive and will prompt the operator to perform some task before continuing execution.
- Utility:** Indicates the test is a utility (display, viewer, configuration or some other function)
- Disabled:** Indicates the test is disabled, must be enabled before test can execute. Test can generally be disabled by default in the snippet, manually or by auto-configuration.
- Requires:** Indicates the test requires something in order to execute properly, generally it is some sort of test hardware.
- Edit Control:** This text control enables the snippet to define the Required hardware that is required.
- Autoconfigure:** Indicates that this test is auto configurable.
- Autocalc Parameters:** Indicates that this test supports recalculating test parameters.

*Note: several test properties will effect the test node icon displayed in the tree.*

### Package Properties Tab:

**Test Properties**

**cTc**  
Cross-Platform Test Controller

Properties | Param View

Name: CPU Features Check

Binary: ./iwCpu32.exe

Number: 2

Timeout: 300

☐ Destructive    ☐ Interactive    ☐ Utility

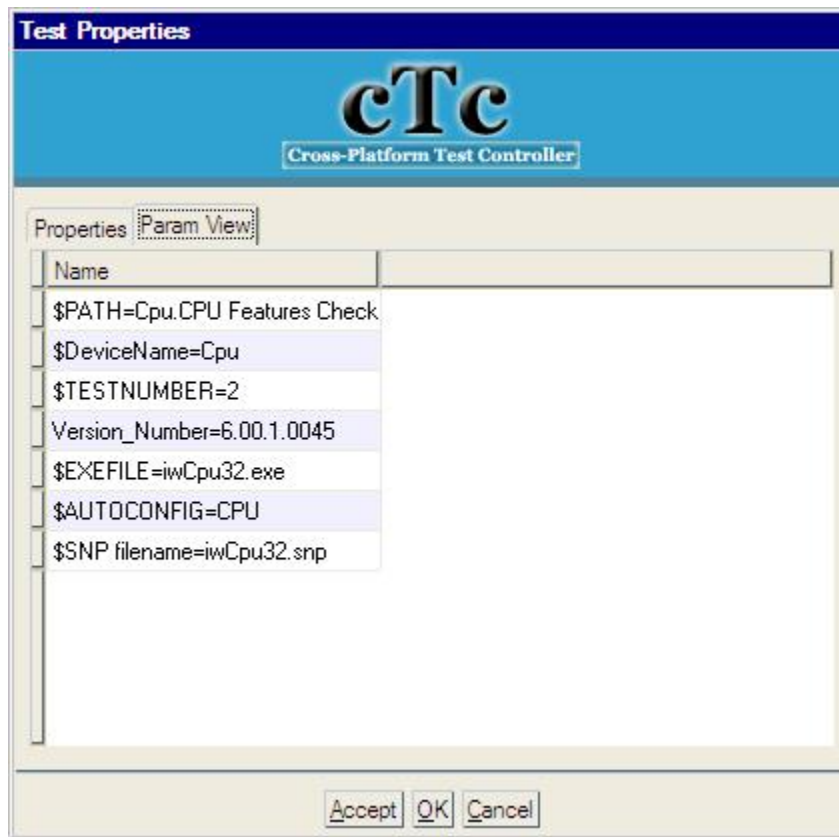
☐ Disabled    ☐ Requires:

☐ Supports Autoconfigure

☐ Supports Autocalc Parameters

Accept OK Cancel

### Parameters Tab:



### 5.8.3 Device Property Sheet

The device property sheet displays all the information relevant to a device node. This property sheet is also used to edit and add devices in a snippet file. When viewing the properties of a device in a package file the some of the edit controls are disabled;

**Version Auto configurable & Recalc Parameters.**

The property sheet is modeless and will dynamically update when the focus changes in the one of the view panes.

**Snippet Properties Tab:**

**Package Properties Tab:**

The image displays two screenshots of the 'Device Properties' dialog box from the 'cTc Cross-Platform Test Controller'. Both screenshots show the 'Param View' tab. The fields are as follows:

- Name:** CPU (top) / Cpu (bottom)
- Binary:** ./iwCpu32.exe
- Help:** CPU.chm (top) / (empty) (bottom)
- Version:** Not Supported
- Timeout:** 300
- Supports Autoconfigure:** ☐ (top) / ☒ (bottom)
- Supports Recalc Parameters:** ☒ (top) / ☐ (bottom)

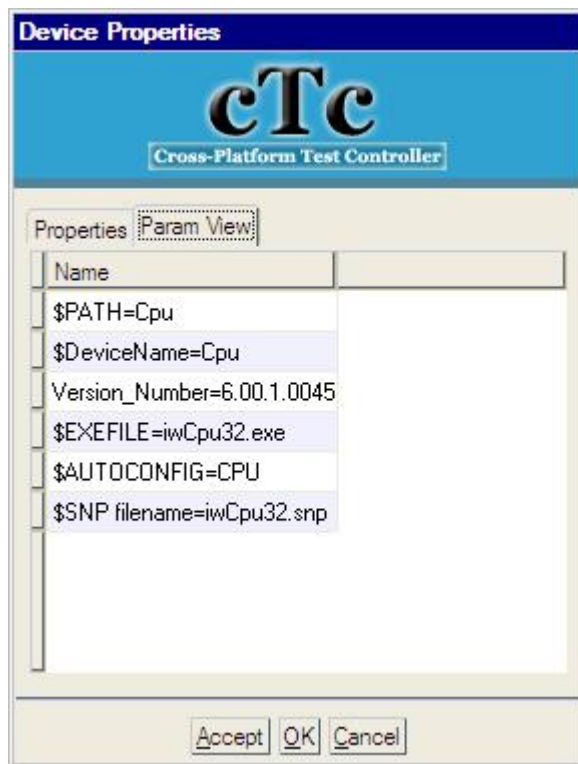
Buttons at the bottom: Accept, OK, Cancel.

**Property Definitions:**

**Name:** Name of the Test.

**Binary:** Name of the Test Module binary file to execute.  
**Help:** Name of the Help File associated with Test Module of this Device. The help file name is stored in the SNX or PKX file in a '\$HLPFILE' parameter at the device node. See Help Options for more details.  
**Version:** Indicates the version of the Test Module's binary file.  
**Timeout:** Indicates the number of seconds CTC will wait in between heart beats from the tests of this Device before determining the test is hung and terminating the test with an error.  
**Autoconfigurable:** Indicates that this device is auto configurable.  
**Autocalc Parameters:** Indicates that this device supports recalculating parameters.

#### Parameters Tab:

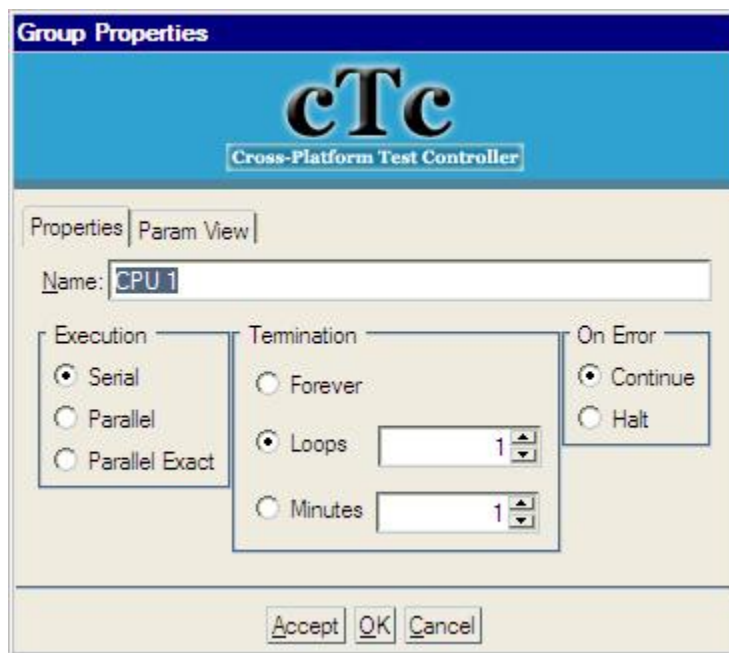


### 5.8.4 Test Group Property Sheet

The Test Group property sheet displays all the information relevant to a Test Group. This property sheet is used to edit and add groups also. This property sheet is only available in a package file. A test group can only be in the Sequence Tree and the snippet file does not support a Sequence Tree.

The property sheet is mode-less and will dynamically update when the focus changes in the one of the view panes.

#### Package Properties Tab:



### Property Definitions:

**Name:** Name of the Test Group Node.

#### **Execution**

**Serial:** Runs each test and test group sequentially in the order displayed in the tree.

**Parallel:** Runs all the tests and test groups recursively in this node branch in parallel.

**Loop Ex:** First test runs 2 times in a sec and second test runs 4 times a sec, parallel with loops equals 4 means that both test will run until the first test reaches 4 loops, approx. in 2 second. Second test will have run approx. 8 times

**Minutes Ex:** Same tests run in parallel with minutes set to 1 means the first test will have run 120 times and the second test will have run 240 times.

**Parallel Exact:** Runs all the tests and test groups in this node branch in parallel for an equal number of times. So all tests are started and then CTC waits until all tests complete then starts them all again.

**Loop Ex:** Same tests as parallel example, run in parallel exact with loops equals 4 means that both tests will run 4 times, at the 1 second mark both tests will only have run 2 times.

**Minutes Ex:** Same tests run in parallel exact with minutes set to 3 means both tests will run for three minutes, first test will have looped 6 times and the second test will have run 12 times.

#### **Termination When**

**Forever:** The tests in the node will loop forever when executed.

**Loops:** The tests in the node will loop for specified number of loops when executed.

**Minutes:** The tests in the node will loop for the specified number of minutes

when executed.

**On Error**

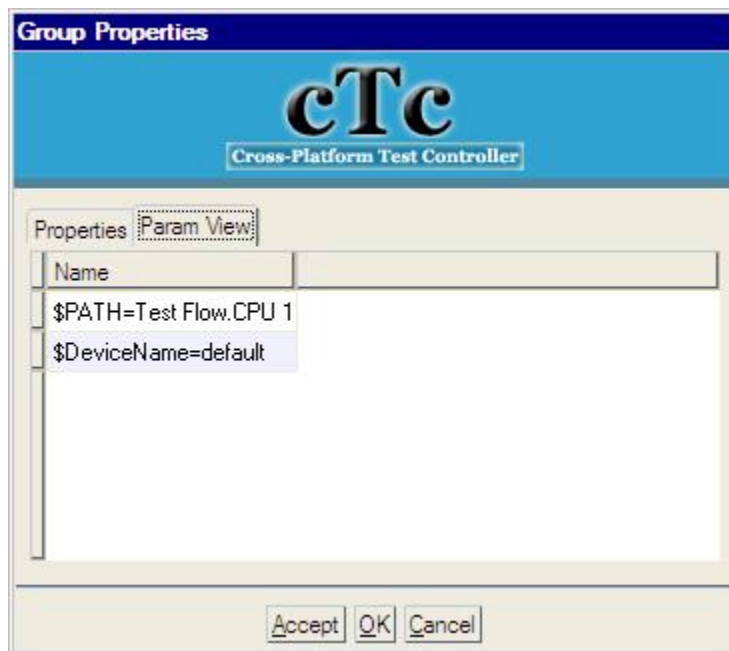
**Continue:**

The tests will continue to execute when an error occurs.

**Halt:**

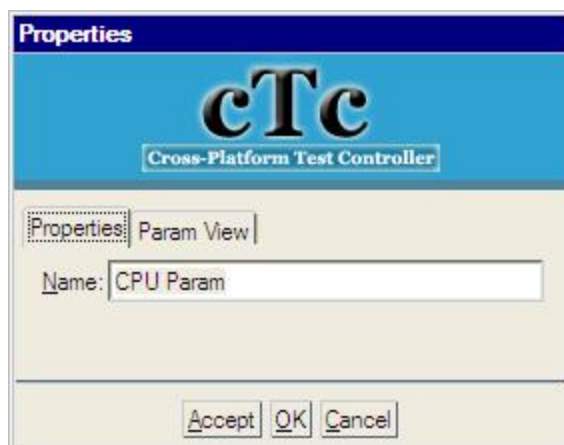
All test execution in this node branch will halt on any error in this branch.

**Parameters Tab:**



### 5.8.5 Parameter Group Property Sheet

The Parameter Group property sheet displays all the information relevant to a Parameter Group node. This property sheet is used to edit and add parameter groups.



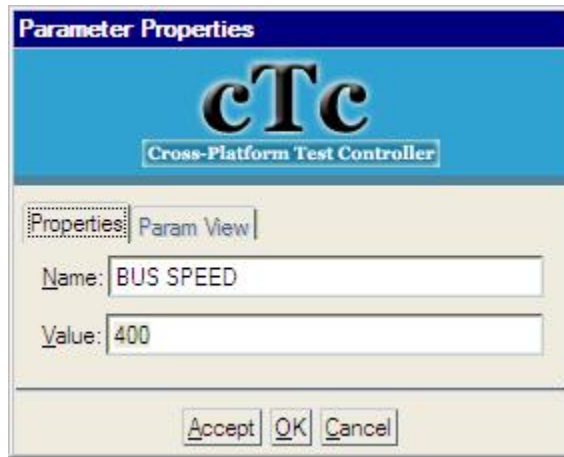
**Property Definitions:**

**Name:** Name of the Parameter Node.



### 5.8.6 Parameter Property Sheet

The Parameter property sheet displays & edits all the information relevant to a Parameter



#### Property Definitions:

**Name:** Name of the Parameter Node.  
**Value:** Parameter Value represented as a string.

### 5.8.7 Parameter Array Property Sheet

The Parameter Array property sheet displays all the information relevant to a Parameter Array. This property sheet is used to edit and add parameter arrays.

C:\Documents and Settings\dayoung\My Documents\CCWork\CTC\WORK\CPU.plx

**cTc**  
Cross-Platform Test Controller

Properties | Param View

Name:

Dimensions: 3 x 3

Array Actions				
<input type="button" value="Insert Row Above"/>	0			
<input type="button" value="Insert Row Below"/>	1			
<input type="button" value="Insert Col Before"/>	2			
<input type="button" value="Insert Col After"/>				
<input type="button" value="Delete Row(s)"/>				
<input type="button" value="Delete Column(s)"/>				
<input type="button" value="Clear Selection"/>				
<input type="button" value="Undo Ctl-Z"/>				
<input type="button" value="Redo Ctl-Y"/>				
<input type="button" value="Up"/>				
<input type="button" value="Left"/> Move Selection <input type="button" value="Right"/>				
<input type="button" value="Down"/>				

### Property Definitions:

**Name:** Name of the Parameter Array.

**Dimensions:** The current parameter array's X, Y dimension.

**Cells:** The window has a grid of cells matching the array dimensions. Each cell is an edit control allowing you to change the parameter value of that cell.

**Cell Headers** Each cell column had a header that is an edit control, which can be used to define a column label

### Property Controls (*Table Actions*):

**Insert Row Above:** Inserts a new row above the currently selected cell(s).

**Insert Row Below:** Inserts a new row below the currently selected cell(s).

**Insert Col Before:** Inserts a new column before the currently selected cell(s).

**Insert Col After:** Inserts a new column after the currently selected cell(s).

**Delete Row(s):** Deletes the row(s) of the currently selected cell(s).

**Delete Col(s):** Deletes the column(s) of the currently selected cell(s).

**Clear Selection:** Clears the contents of the currently selected cell(s).

**Undo:** Un-does the last edit, (Ctl-Z) hot key.

**Redo:** Re-does the last undo, (Ctl-Y) hot key.  
**Move Selection:** These controls move the currently selected cells in the directions indicated; **Up, Down, Left, & Right.**

## 6 Return Codes

The CTC GUI is intended to be fully scriptable, it supports the following return codes:

- 0 - PASS condition, tests have never been manually aborted and have never failed.
- 1 - ABORT condition, indicates the test have been manually aborted. This return code takes precedence over all other return codes.
- 2 - FAIL condition, at least one error occurred and the tests were never manually aborted.
- 3 - INVALID condition, at least one error occurred and the tests were never started.

See also, Invocation Parameters for scriptable and execution options.

## 7 Invocation Parameters

These parameters can be entered on the command line invocation of CTC to control various features. All command line switches may start with either / or - and the actual argument is case in-sensitive.

Example:

```
./ctc -auto
./ctc /AUTO
```

Parameters following a switch are required if surrounded by brackets []. Optional parameters are surrounded by angle brackets <>.

/AUTO	Run autoconfig first.
/CFG	Use specified configuration tree on startup.
/DLL	Custom DLL to load with CTC.
/FD	Generates a PASS/FAIL screen on test completion.
/FLOW	Use specified flow tree associated with given configuration specified with the /CFG flag
/FOREVER	Runs the test flow forever.
/HE	Set top flow node to Halt on error.
/LOOPS	Specifies the number of loops to run.
/MAX	Forces the CTC GUI into full screen mode.
/MINUTES	Stop testing after number of minutes.
/N	Stop testing after number of loops.
/PASSFILE	Generates file with specified filename only if automated tests pass (useful for scripting)
/PC	Same as /CFG
/PKG	Initial document file opened by CTC.
/PORT	Indicates Executive to Run engine communication socket port.
/Q	Quite mode for test messages
/QUIT	Exit CTC when tests complete.
/QUITPASS	Exit CTC when tests have completed with no errors.
/RUN	Run tests without user input.

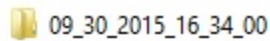
/SELF	Same as /AUTO
/SUMMARY	Creates a summary file of test results.
/W	Returns the default response for all interactive messages.
/V	Turns on verbose and supplies a mask.

### Message Redirections:

/RE	Redirect errors to filename.
/RA	Redirect advisory to filename.
/RS	Redirect status to filename.
/RP	Redirect pause to filename.
/RT	Redirect statistics to filename.
/RR	Redirect all to filename.
/RV	Redirect verbose to filename.

## 7.1 /ARCV

This flag will turn on the archive files option. If the archive files option is enabled, a directory with timestamp will be created automatically each time the CTC "RUN" button is hit. All test messages redirection files will be generated inside this timestamp folder.



The timestamp directory will be generated automatically.

All test messages redirection files will be generated inside the timestamp directory.

Name	Date modified	Type	Size
ctc_advisory_mesg.txt	9/30/2015 16:34	TXT File	1 KB
ctc_error_mesg.txt	9/30/2015 16:34	TXT File	1 KB
ctc_info_mesg.txt	9/30/2015 16:34	TXT File	1 KB
ctc_prompt_mesg.txt	9/30/2015 16:34	TXT File	1 KB
ctc_statistic_mesg.txt	9/30/2015 16:34	TXT File	1 KB
ctc_status_mesg.txt	9/30/2015 16:34	TXT File	13 KB
ctc_verbose_mesg.txt	9/30/2015 16:34	TXT File	7 KB
sum_20150930163106.txt	9/30/2015 16:34	TXT File	14 KB

## 7.2 /AUTO

This parameter will cause auto-configure to be run when CTC starts.

**Command Line:**     /AUTO <configname>

<configname> - can be any valid string – see 4.6.13 for valid configuration names.

The auto switch executes auto-configure and uses the configuration name if specified,

otherwise it uses a unique configuration name. Default name is '*Modules Configured On: <today's date&time>*' This is the same as the /SELF switch, except that it saves the package file when auto-configure has completed.

If this parameter is used in combination with the /RUN parameter, auto-configure will always occur before tests are run.

This flag cannot be used in combination with /CFG & /FLOW as auto-configure will create a new configuration & new flow. It will not update the configuration & flow tree specified by the /CFG & /FLOW flags.

### 7.3 /CFG

The pc switch is used in combination with the run switch and determines which configuration to run. The /PC and /CFG switches are the same.

**Command Line:**     /CFG name

### 7.4 /DLL

Causes CTC to dynamically load the specified list of libraries.

**Command Line:**     /DLL [ *filename1*[*p1*, .. *pN*] ]

Where *filenameX* is the full name of the DLL and *p1*,..*pN* are any parameters required by the DLL. The filename must be an absolute path only if the file does not reside in the same directory as CTC. More than one filename may be specified if each

The libraries specified must conform to the CTC Custom DLL API. This API allows a test integrator to customize how CTC responds to list of events:

- All testing started
- All testing stopped
- Each test started
- Each test stopped
- Error message sent
- ... (other messages)

The syntax of the command allow more than one DLL to be loaded at a time and the parameter field allows parameters to be passed to each DLL.

### 7.5 /FD

When used will display a "PASSED" dialog box at the end of the test run if no errors occurred. A "FAILED" dialog box will be displayed on any error.

**Command Line:**     /FD

## 7.6 /FLOW

The /FLOW switch is used in combination with the /CFG switch to determine which supported flow to run. The configurations can support multiple flow trees, this enables the selection of which one is used from the command line. Must use quotes around filenames that contain white space.

**Command Line:** /FLOW "flow name"

## 7.7 /FOREVER

The forever switch overrides the package file parameters which determine how long a flow runs for. This switch is used in combination with the /run switch.

**Command Line:** /FOREVER

## 7.8 /HE

Forces testing to stop after an error occurs. When tests are stopped a signal is sent to all running tests. Each test is responsible to check the signal, stop testing, clean up and execute. The amount of time it takes to stop is non-deterministic when tests are running in parallel.

**Command Line:** /HE

## 7.9 /LF

The LF flag causes the auto-creation of a log file containing all Error, Status, Info and Advisory messages. If no file name is specified, the log file will be named <package\_filename>\_<timestamp>.txt. If a file name is specified, the log file created will have that name. No change occurs to other redirection or verbose/quiet message settings.

**Command Line:** /LF [filename]

## 7.10 /LOOPS

The loops switch overrides the package file parameters which determine how many loops a flow does. This switch is used in combination with the /RUN switch. The /N and /LOOPS switches are the same.

**Command Line:** /LOOPS [*number*]

[*number*] - 1 ... 2<sup>31</sup> (unsigned long)

The parameter *number* indicates how many minutes to allow testing to take place. This parameter is required. If the parameter is omitted the command is ignored. When *number* value is 0 the testing will loop forever.

## 7.11 /MAX

Forces the CTC GUI into full screen mode.

**Command Line:**     /MAX

## 7.12 /MINUTES

Forces testing to stop after specified number of minutes. This switch is used in combination with the /run switch.

**Command Line:**     /MINUTES [*number*]

[*number*] - 1 ... 2<sup>31</sup> (unsigned long)

The parameter *number* indicates how many minutes to allow testing to take place. This parameter is required. **If the parameter is omitted the command is ignored.**

## 7.13 /N

Forces testing to stop after specified number of loops. By default testing stops after 1 loop, so this parameter is only necessary when *number* is greater than 1. The /N and /LOOPS switches are the same.

**Command Line:**     /LOOPS [*number*]

[*number*] - 0 ... 2<sup>31</sup>

The parameter *number* indicates how many minutes to allow testing to take place. This parameter is required. If the parameter is omitted the command is ignored. When *number* value is 0 the testing will loop forever.

## 7.14 /PASSFILE

Write an empty file to the specified path name when testing is complete and all tests have passed. This is most useful when attempting to monitor CTC from a batch file.

**Command Line:**     /PASSFILE filename

## 7.15 /PC

The pc switch is used in combination with the run switch and determines which configuration to run. The /PC and /CFG switches are the same.

**Command Line:**     /PC name

## 7.16 /PKG

Use the specified filename as the test package file to open. The default is "T.PKX".

**Command Line:**     /PKG <*pkgname*>

## 7.17 /PORT

This switch specifies the port to use for communication between the Test Executive and Run Engine. It is recommended to use a port above 2000.

**Command Line:**     /PORT [*port#*]

[*port#*] - 1024 – 65535

## 7.18 /Q

Suppress most output to screen. Does not suppress Info and Prompt messages, since they are intended to always be viewed.

**Command Line:**     /Q

## 7.19 /QUIT

Forces CTC to exit after tests have been run. This command is most commonly used in combination with the /RUN or /AUTO command.

**Command Line:**     /QUIT

## 7.20 /QUITPASS

The quitpass switch exits CTC after all other tasks have completed without error.

**Command Line:**     /QUITPASS

## 7.21 /RECONFIG

This flag will trigger reconfiguration on the pkg under specific configuration & flow tree. It can be used together with /CFG & /FLOW flags. Configuration tree is a must to run this flag while flow is optional. If the flow tree is not specified, the default flow tree will be reconfigure.

If this parameter is used in combination with the /RUN parameter, tests will run after the reconfiguration.

**Command Line:**     /RECONFIG /CFG <cfgtree> /FLOW <flowtree>  
                          (OPTIONAL)

## 7.22 /RMTLOG

This flag will enable the remote log files under remote CTC execution. Three log files will be generated: *librmthostlog.txt*, *librmthost\_recv\_ts.txt* & *librmthost\_recv\_nts.txt*.



*librmthostlog.txt* - captured all the transaction activities, including parameters, commands & data received from the remote service.  
*librmthost\_recv\_ts.txt* & *librmthost\_recv\_nts.txt* - only captured the data receiving from the remote service.

## 7.23 /RUN

The run switch executes a test flow. This switch can be used in combination with the /CFG, /AUTO, /SELF.

Example: `./t /PKG t.pkg /AUTO cfg1 /RUN /CFG cfg1`

This will open the t.pkg file, run autoconfigure and name it cfg1, and run the flow associated with cfg1.

**Command Line:**     /RUN

## 7.24 /SELF

Self sense the hardware and run tests. Similar to /AUTO but does not save the configuration to the package file.

**Command Line:**     /SELF

## 7.25 /SUMMARY

The summary switch creates a comma separated summary file.

**Command Line:**     /SUMMARY

## 7.26 /W

Do not wait for user response. Instead, use module's default value for any prompts. Intended for scripting.

**Command Line:**     /W

## 7.27 /V

This switch sets the verbose filter to the verbose mask if specified, otherwise it uses a verbose mask of -1 (turn on all verbose messages).

**Command Line:**     /V <mask>

<mask> format handles both decimal and hexadecimal.

## 7.28 Message Redirection

The following parameters can be used to redirect test message types to files. Any combination of the parameters is allowed.

### Command Line:

/RE [filename]	Redirect errors to <i>filename</i> .
/RA [filename]	Redirect advisory to <i>filename</i> .
/RS [filename]	Redirect status to <i>filename</i> .
/RP [filename]	Redirect pause to <i>filename</i> .
/RT [filename]	Redirect statistics to <i>filename</i> .
/RI [filename]	Redirect info and debug to <i>filename</i> .
/RR [filename]	Redirect all to <i>filename</i> .
/RV [filename]	Redirect verbose to <i>filename</i> .
/SF [filename]	Redirect summary info to <i>filename</i> .

## 8 Reserved Parameters

These parameters are used by CTC, not by the test module. **They are hidden by CTC.**

```
$AUTOCONFIG
$EXEFILE
$HELPPFILE
$TESTNUMBER
$TESTTIMEOUT
$VERBOSEMASK
```

### 8.1 \$AUTOCONFIG

CTC requires the \$AUTOCONFIG parameter to be defined in each module which participates in automatic configuration. The value of this parameter is used by the module to determine which type of test group to auto configure. This may seem odd until you consider that modules may support more than one kind of hardware. For instance a disk module could support floppies, hard-drives and CDRoms. Each module defines its own set of values for this parameter so see the module help file for specifics.

#### Examples:

```
$AUTOCONFIG = floppy
$AUTOCONFIG = auto
```

#### See Also:

CTC Parameters

### 8.2 \$EXEFILE

CTC requires the \$EXEFILE parameter to be defined for each test that it executes. It should specify the name of the executable module that contains the test. If the full path to the executable is not specified, CTC searches the current directory. The \$EXEFILE

parameter is specified in the device property control, by <double-click> on the device node or <right-click> and select Node Properties in the edit control. The binary is only editable in the snippet file.

**Examples:**

```
$EXEFILE=c:\diskdrv\debug\diskdrv.exe  
$EXEFILE=diskdrv.exe
```

**See Also:**

CTC Parameters

## 8.3 \$HELPPFILE

CTC requires the \$HELPPFILE parameter to be defined for each device. It should specify the name of the help file for that device. If the full path to the file is not specified, CTC searches the current directory. The \$HELPPFILE parameter can be defined in the device property control, by <double-click> on the device node or <right-click> and select Node Properties in the edit control.

**Examples:**

```
$HELPPFILE=c:\program files\CTC\hlp\diskdrv.chm  
$HELPPFILE=./diskdrv.chm
```

**See Also:**

Node Help

## 8.4 \$TESTNUMBER

CTC requires the \$TESTNUMBER parameter to be defined for each test that it executes. It should be an integer representing the test index in the test module defined by the \$EXEFILE parameter.

**Example:**

```
$TESTNUMBER=1
```

**See Also:**

CTC Parameters

## 8.5 \$VERBOSEMASK

This optional parameter is a 32-bit value where each bit represents a different group of extra messages that can be sent by a test module. It is up to the test developer what messages are displayed by each bit. This parameter is normally only used for debugging purposes.

**Examples:**

\$VERBOSEMASK=0xFFFFFFFF ;displays all extra messages  
\$VERBOSEMASK=0 ;default, displays no extra messages  
\$VERBOSEMASK=0x40 ;display messages represented by bit 7 in this mask.

**See Also:**

CTC Parameters

## 9 Remote Support

CTC is now supporting remote testing. Remote CTC allows user to open & run the test package (.pkx) from the host system while the test modules are executed at the remote client system. All the test commands & messages are communicating through TCP/IP packets. The following topics are designed to get you started with the CTC remote testing:

- Setup Remote Client
- Configure Remote Pkx
- Multiple Nodes Support
- Remote Log Files
- Error & Exception Handling
- Remote Probing Over IP Range

### 9.1 Remote client service

#### Windows

**Background:**

Remote CTC client service is consist of two binaries:

- 1.) RmtLibMtaSvc.exe - Win32 service routine, calling the spawned RmtLibMta.exe.
- 2.) RmtLibMta.exe - Service main, always listening to the designated socket port, sending/receiving test commands/messages and executing the test modules.

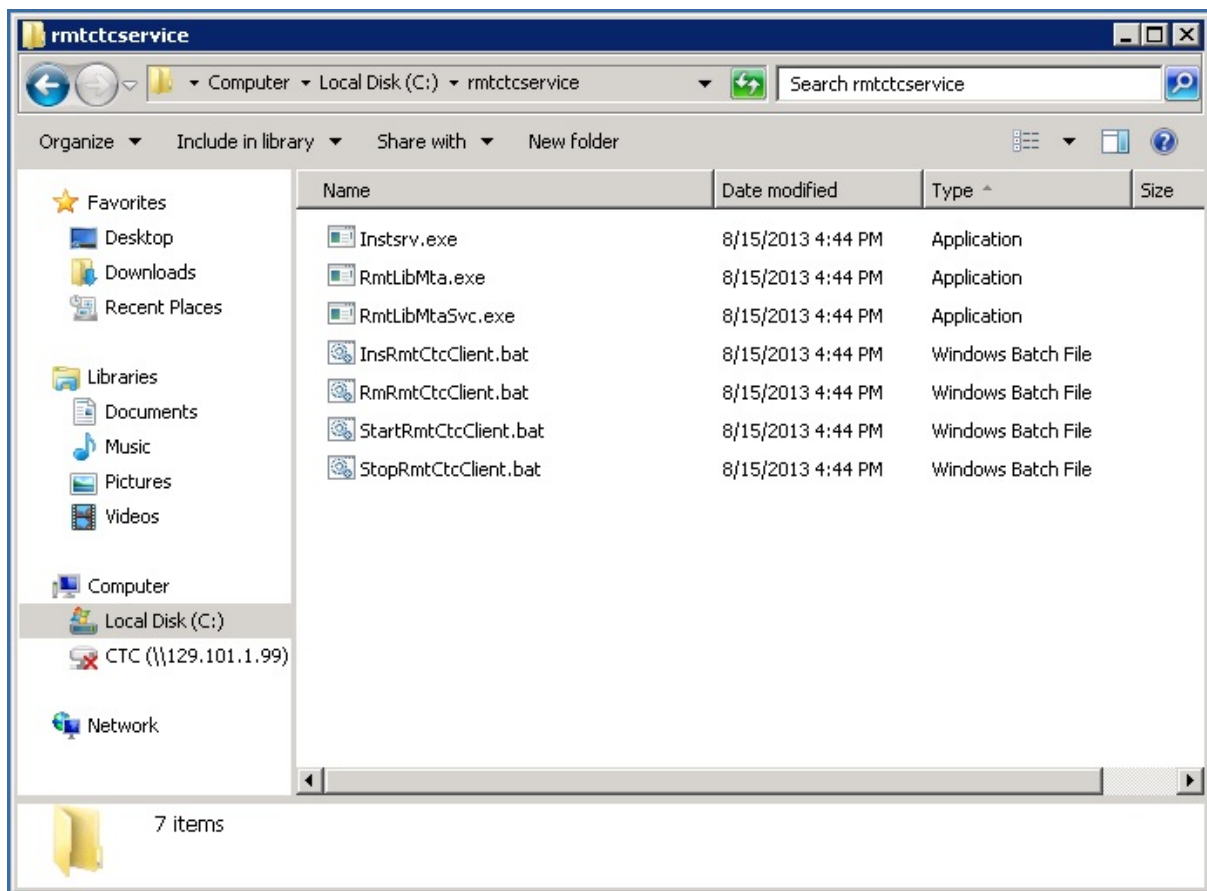
**Installing the service:**

Installing the remote client service can be done easily by clicking at the *InstallRemoteCTCClient* batch file. *InstallRemoteCTCClient.bat* can be found inside the CTC installation directory once you've installed the CTC through the CTC installer. Here are what the batch file will do:

- 1.) Creating a new *rmtctcservice* directory under *C:\*.
- 2.) Copying the two service exe files to the new created *C:\rmtctcservice* directory.
- 3.) Copying all the files inside the *rmtctcservice* folder under CTC installation directory to the new created *C:\rmtctcservice* directory.
- 4.) The *rmtctcservice* folder under the CTC installation directory contains all the

files that needed to install, start, stop & remove the service.

5.) Installing the service.



```

C:\CTC_2.12>InstallRemoteCTCClient.bat

"Creating rmtctcservice dir..."

"Copying files to C:\rmtctcservice..."
    1 file(s) copied.
    1 file(s) copied.
.\rmtctcservice\InsRmtCtcClient.bat
.\rmtctcservice\Instsrv.exe
.\rmtctcservice\RmRmtCtcClient.bat
.\rmtctcservice\StartRmtCtcClient.bat
.\rmtctcservice\StopRmtCtcClient.bat
    5 file(s) copied.

"Installing Remote CTC client service..."

***** WARNING WARNING WARNING *****
This file adds the RemoteCtcClient service to the operationg system.
This file should only be run once during the life of a system.
If you are not sure this file has been run before check
Control Panel - Services to see if RemoteCtcClient is installed.
If it is, do NOT run this batch file without first removing it.
If you don't know what you are doing hit CTL C NOW!
Press any key to continue . . .

CreateService: RmtCtcClient SUCCESS
Description: Remote CTC client service

c:\rmtctcservice>_

```

After the installation, you should be able to see the *RmtCtcClient* service in the service console.

Remote Registry	Enables remote users to modify registry settings ...	Started	Automatic	Local Service
Resultant Set of Policy Provider	Provides a network service that processes requ...		Manual	Local System
<b>RmtCtcClient</b>	<b>Remote CTC client service</b>		<b>Automatic</b>	<b>Local System</b>
Routing and Remote Access	Offers routing services to businesses in local are...		Disabled	Local System
RPC Endpoint Mapper	Resolves RPC interfaces identifiers to transport ...	Started	Automatic	Network Service

### Configuring the service:

The remote client service supports 6 parameters to allow user to customize the settings. Default value will be used if there is no input from user.

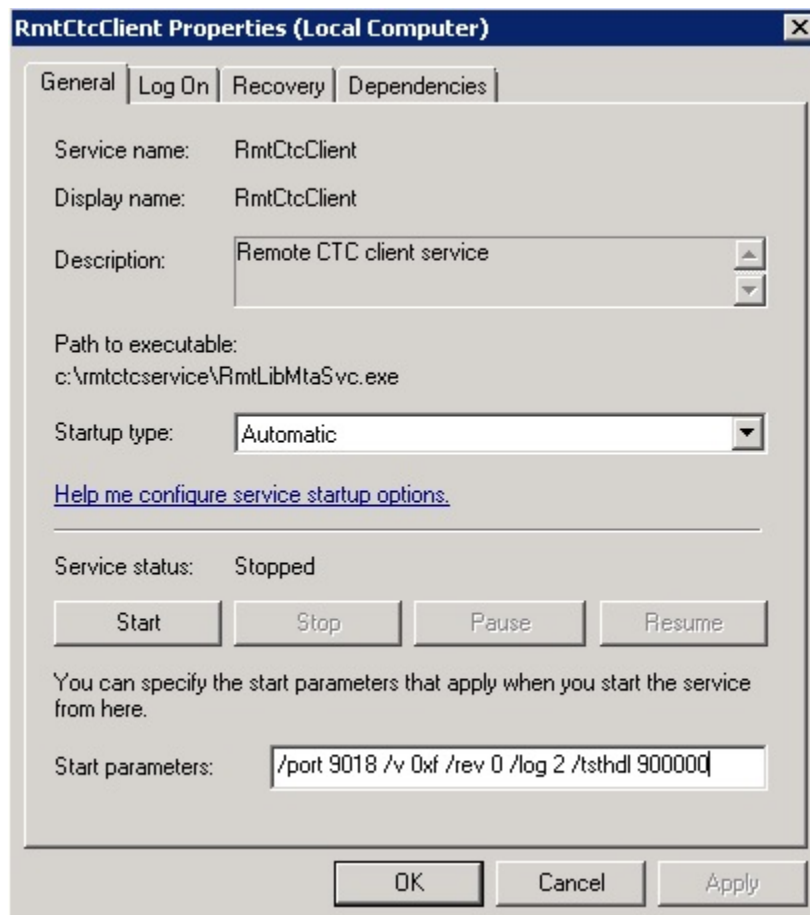
- 1.) /port - Allows user to change the remote port number. Default = 8080.
- 2.) /v - Allows user to set the verbose messages level. Default = 0 (off).
- 3.) /rev - Allows user to set the internal verbose messages level. Default = 0 (off).
- 4.) /log - Allows user to set the log file output level. Default = 0 (no log file).
- 5.) /tsthdl\* - Allows user to set the initial test handle number. Default = 800000.
- 6.) /path - Allows user to specify the remote CTC service binary path. Example: /path [\remote]. Default = "C:\rmtctcservice".

\*The test handle number will keep increasing depends on the total number of tests we have in the ptx. Each test handle number is representing a single test node in the ptx;

Therefore, they have to be identical to prevent any confusion at the CTC GUI. For example, the starting test handle number = 100000 and we have 100 tests inside the pkx; the first test will get the test handle number 100000 while the last test will get the test handle number 100100. It is recommended to reserve a comfortable range of initial test handle number within the remote clients to avoid overlapping. For example: client 1 = 100000, client 2 = 300000, client 1 are able to support 200000 tests before it overlap with client 2. Test handle number will be reset each time we restart the service.

To configure the service:

- 1.) Go to service console and double click on the *RmtCtcClient* service.
- 2.) *RmtCtcClient* properties dialog box will be popped up, stop the service (if it is started) by clicking at the *Stop* button.
- 3.) Input the customize parameters at the *Start parameters* input pane.
- 4.) Click *Start* button to start the service, *OK* to close the dialog box.

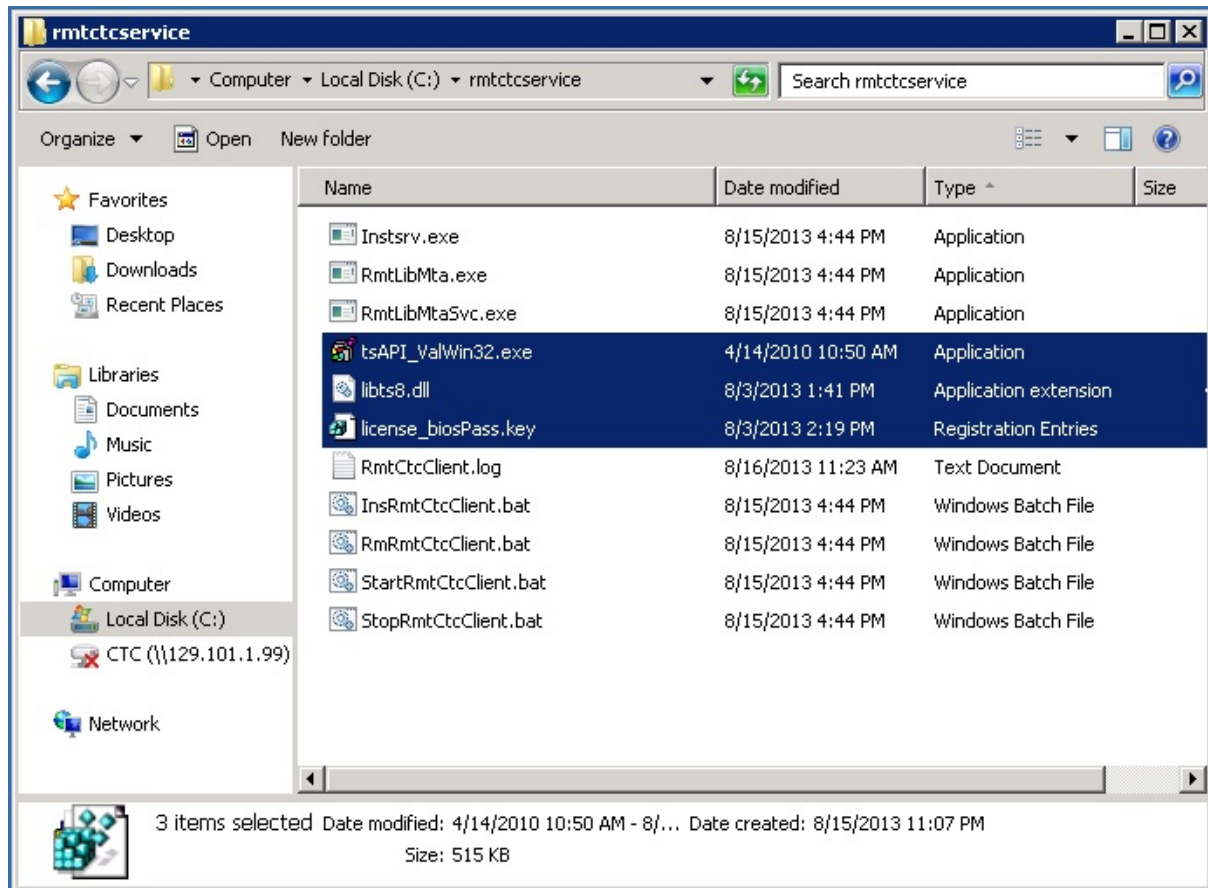


- 5.) A "Started" string indicates that the service is started successfully.

Remote Registry	Enables remote users to modify registry settings ...	Started	Automatic	Local Service
Resultant Set of Policy Provider	Provides a network service that processes requ...		Manual	Local System
RmtCtcClient	Remote CTC client service	Started	Automatic	Local System
Routing and Remote Access	Offers routing services to businesses in local are...		Disabled	Local System
RPC Endpoint Mapper	Resolves RPC interfaces identifiers to transport ...	Started	Automatic	Network Service

### Preparing for test:

Copy the test module binary, license key & all the required .dll files (*libts8.dll*, *testlib8.dll*, *wmi.dll* & etc) to the *C:\rmtctcservice* directory. The client is ready to support remote testing!



## Linux

### Background:

Remote CTC client service is compiled as a Daemon service in Linux. *RmtLibMtaSvc* is the service binary.

### Installing & Configuring the service:

Installing the remote client service can be done easily by running the *InsRmtSvc* bash file. *InsRmtSvc* bash file can be found inside the CTC installation directory once you've installed the CTC through the CTC installer.

The remote client service supports 5 parameters to allow user to customize the settings. Default value will be used if there is no input from user.

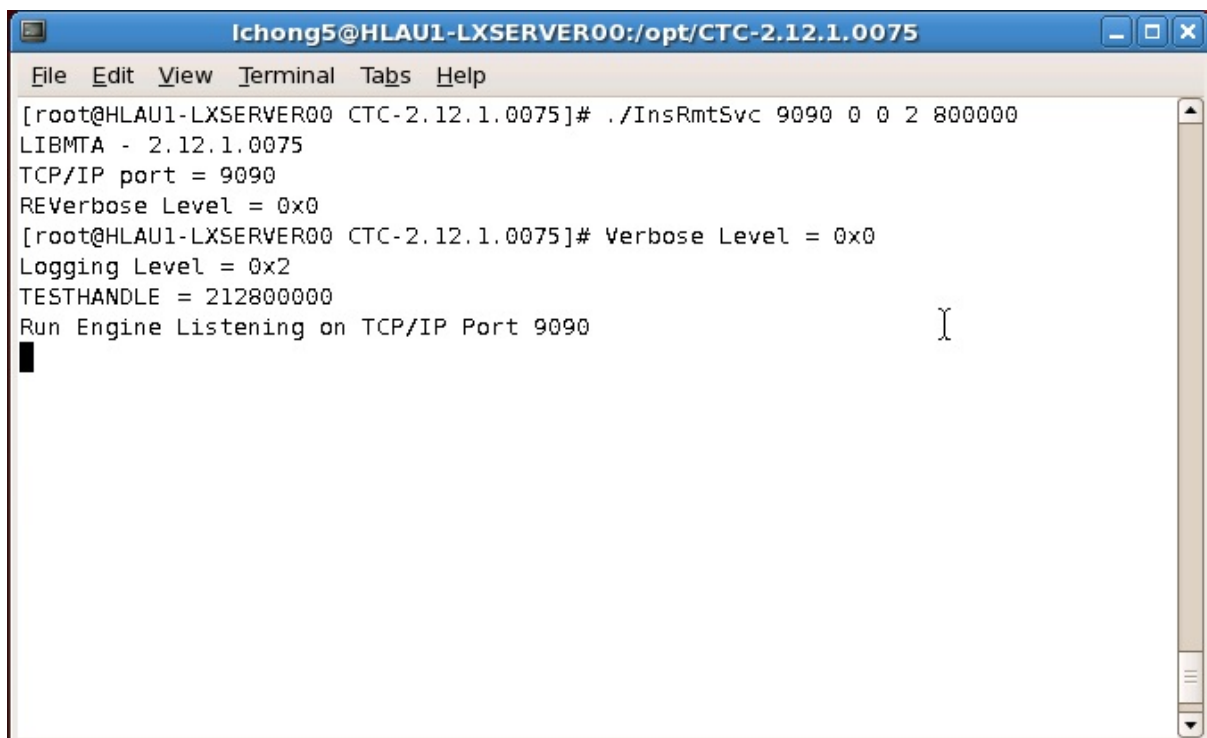


- 1.) /port - Allows user to change the remote port number. Default = 8080.
- 2.) /v - Allows user to set the verbose messages level. Default = 0 (off).
- 3.) /rev - Allows user to set the internal verbose messages level. Default = 0 (off).
- 4.) /log - Allows user to set the log file output level. Default = 0 (no log file).
- 5.) /tsthdl\* - Allows user to set the initial test handle number. Default = 800000.

\*The test handle number will keep increasing depends on the total number of tests we have in the pxx. Each test handle number is representing a single test node in the pxx; Therefore, they have to be identical to prevent any confusion at the CTC GUI. For example, the starting test handle number = 100000 and we have 100 tests inside the pxx; the first test will get the test handle number 100000 while the last test will get the test handle number 100100. It is recommended to reserve a comfortable range of initial test handle number within the remote clients to avoid overlapping. For example: client 1 = 100000, client 2 = 300000, client 1 are able to support 200000 tests before it overlap with client 2. Test handle number will be reset each time we restart the service.

To install & configure the service:

- 1.) Open a terminal and go to the CTC installation directory.
- 2.) Execute the *InsRmtSvc* bash file with the desired input parameters.



The screenshot shows a terminal window titled "Ichong5@HLAU1-LXSERVER00:/opt/CTC-2.12.1.0075". The terminal displays the following output after running the script:

```
[root@HLAU1-LXSERVER00 CTC-2.12.1.0075]# ./InsRmtSvc 9090 0 0 2 800000
LIBMTA - 2.12.1.0075
TCP/IP port = 9090
REVerbose Level = 0x0
[root@HLAU1-LXSERVER00 CTC-2.12.1.0075]# Verbose Level = 0x0
Logging Level = 0x2
TESTHANDLE = 212800000
Run Engine Listening on TCP/IP Port 9090
```

3.) Once the installation is completed successfully, use the *ps -ux* command to make sure the *RmtLibMtaSvc* service is started successfully.

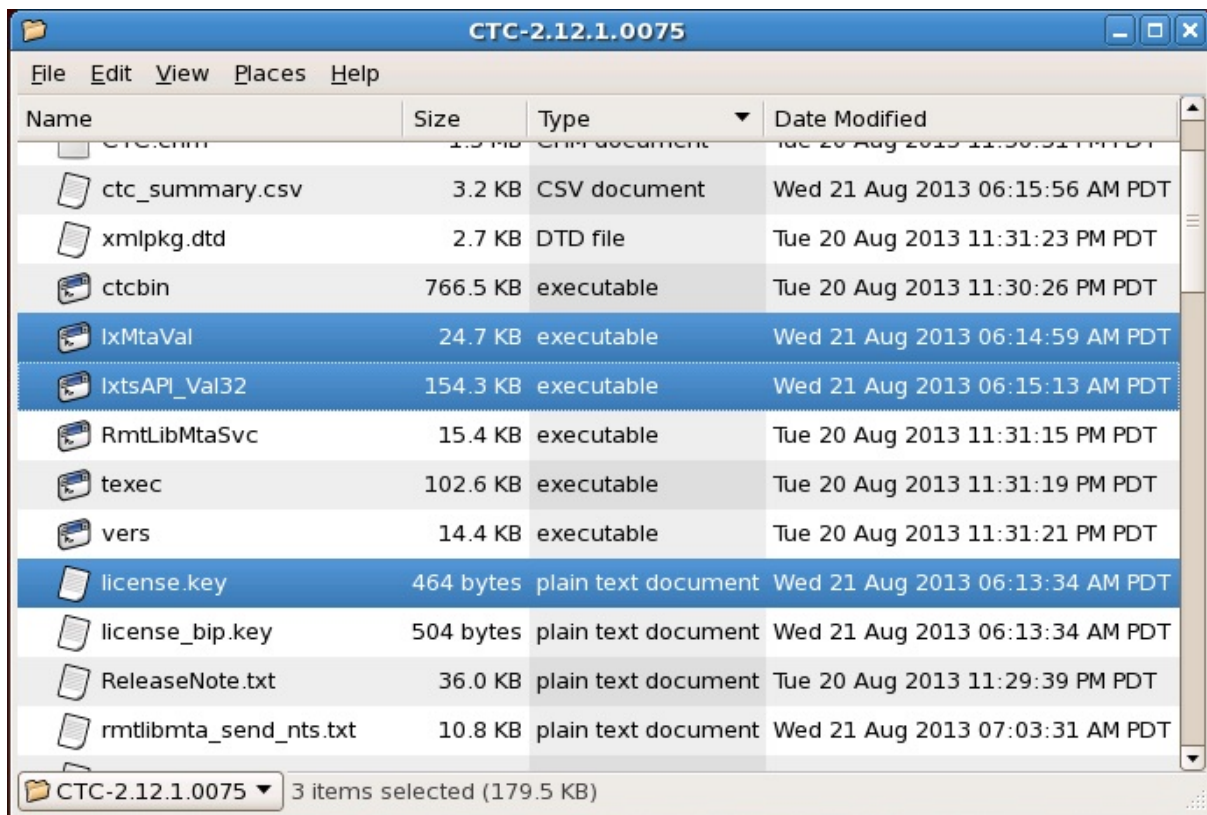
```

lchong5@HLAU1-LXSERVER00:~$ ps -aux
File Edit View Terminal Tabs Help
lchong5 25288 0.0 0.6 24516 12988 ? Ss 05:51 0:00 /usr/bin/python
lchong5 25294 0.0 0.4 73324 10140 ? Ss 05:51 0:00 nm-applet --sm-
lchong5 25297 0.0 0.2 15112 4248 ? Ss 05:51 0:00 pam-panel-icon
lchong5 25303 0.0 0.2 73132 6188 ? Ss 05:51 0:00 gnome-power-man
lchong5 25306 0.0 0.1 17400 2268 ? Sl 05:51 0:00 ./escd --key_In
lchong5 25320 0.0 0.0 2532 1144 ? S 05:51 0:00 /usr/libexec/ga
lchong5 25322 0.2 0.5 75768 11396 ? S 05:51 0:10 /usr/libexec/wn
lchong5 25324 0.0 0.4 105132 10248 ? S 05:51 0:00 /usr/libexec/tr
lchong5 25328 0.0 1.0 41540 22316 ? S 05:51 0:00 /usr/bin/python
lchong5 25337 0.0 0.3 22668 6348 ? S 05:51 0:00 /usr/libexec/no
lchong5 25339 0.0 0.4 27568 10076 ? S 05:51 0:00 /usr/libexec/cl
lchong5 25341 0.0 0.4 75708 9976 ? S 05:51 0:00 /usr/libexec/mi
lchong5 25344 0.0 0.0 2452 832 ? S 05:51 0:00 /usr/libexec/ma
lchong5 25350 0.0 0.0 15584 2060 ? Ss 05:52 0:01 gnome-screensav
lchong5 27152 0.2 0.6 39520 13316 ? Sl 06:23 0:04 gnome-terminal
lchong5 27155 0.0 0.0 2452 672 ? S 06:23 0:00 gnome-pty-helpe
lchong5 27156 0.0 0.0 4632 1444 pts/0 Ss 06:23 0:00 bash
lchong5 27422 0.0 0.0 4636 1456 pts/1 Ss 06:45 0:00 bash
lchong5 27579 0.0 0.0 33860 800 ? Ssl 06:57 0:00 ./RmtLibMtaSvc
lchong5 27584 0.0 0.0 4228 928 pts/1 R+ 06:57 0:00 ps -ux
lchong5@HLAU1-LXSERVER00 ~]$

```

### Preparing for test:

Copy the test module binaries & license key to the CTC installation directory. The client is ready to support remote testing!



## 9.2 Remote ptx setup

Parameter *Rmt\_IP\_Address* is required to turn on the remote testing for CTC. Users have to manually set this parameter with the targeted remote client IP & port number. Port number can be set by putting a ':' following the IP address, refer to the examples below for more details. If port number is not defined, default port number of 8080 will be used.

CTC will loop through the ptx file; When the *Rmt\_IP\_Address* parameter is detected, it will launch the remote testing process. The *Rmt\_IP\_Address* parameter is just like other CTC parameter where all parameter's attributes are inherited. If it is put on top of the test node tree in the ptx, the entire ptx will run remotely; However, if it is put under specific test node, only that particular test will run remotely. Please refer to some of the examples below for more details.

The *Rmt\_Connect\_Retry* optional parameter allows user to define the number of retries to connect to the remote client. If the parameter is not set, default value of 20 retries will be used.

The *Rmt\_Packet\_Size* optional parameter allows user to define the number of bytes to send in a single TCP packet. Smaller size packets will improve the stability of the remote tests but will also increase the test time as more "sends" are needed to complete a single test. If the parameter is not set, the default packet size of 2k bytes will be used. If the parameter size is set > 8k bytes, the packet size will be limit to 8k bytes to incorporate with the common TCP packet size limit that set at maximum 8k bytes.

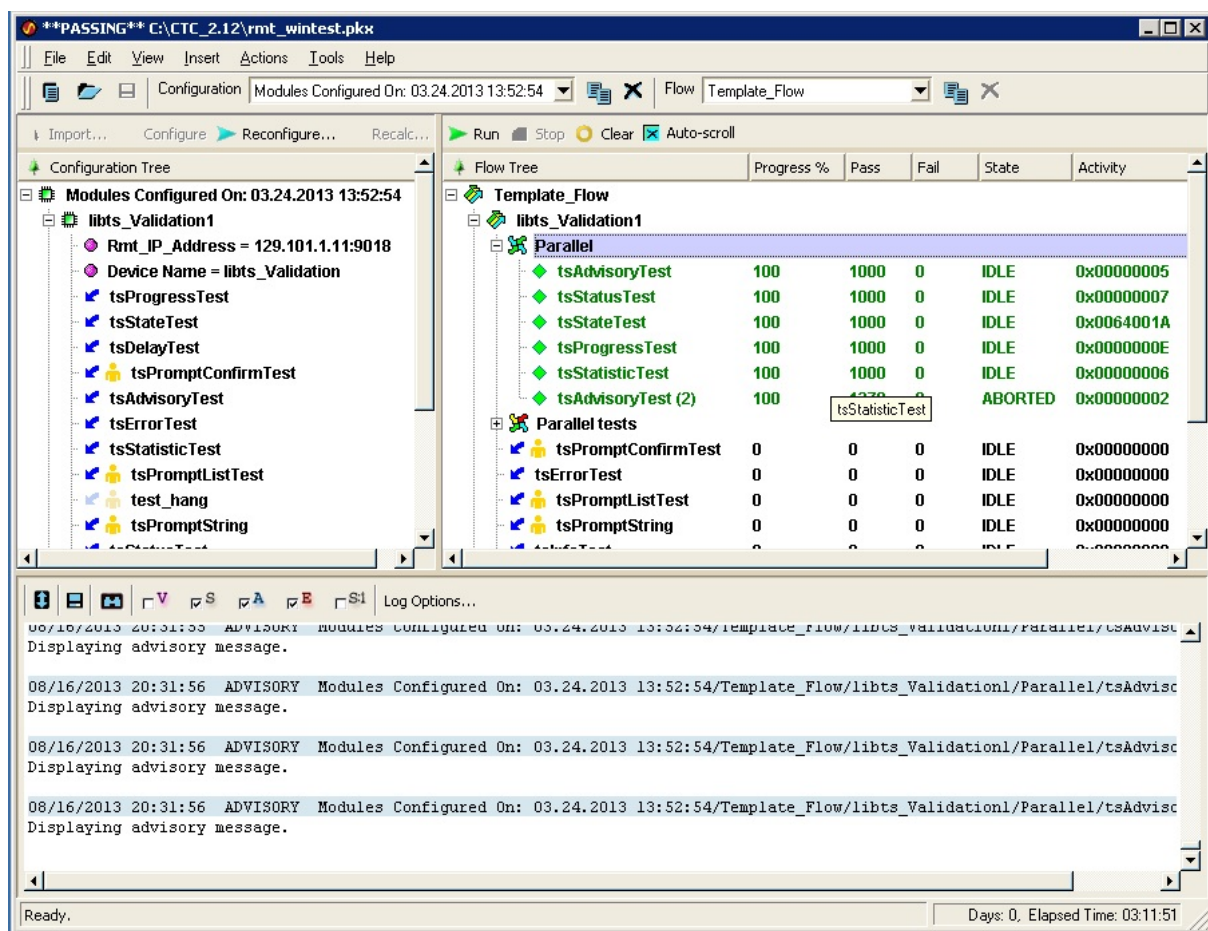
The *Rmt\_Test\_Delays* optional parameter allows user to set a delay in milliseconds within the remote test cycle, providing more time to Remote CTC service to cleanup the resource before another test kick started. This parameter helps to improve the overnight looping test stability. The default will be no delay.

### Examples:

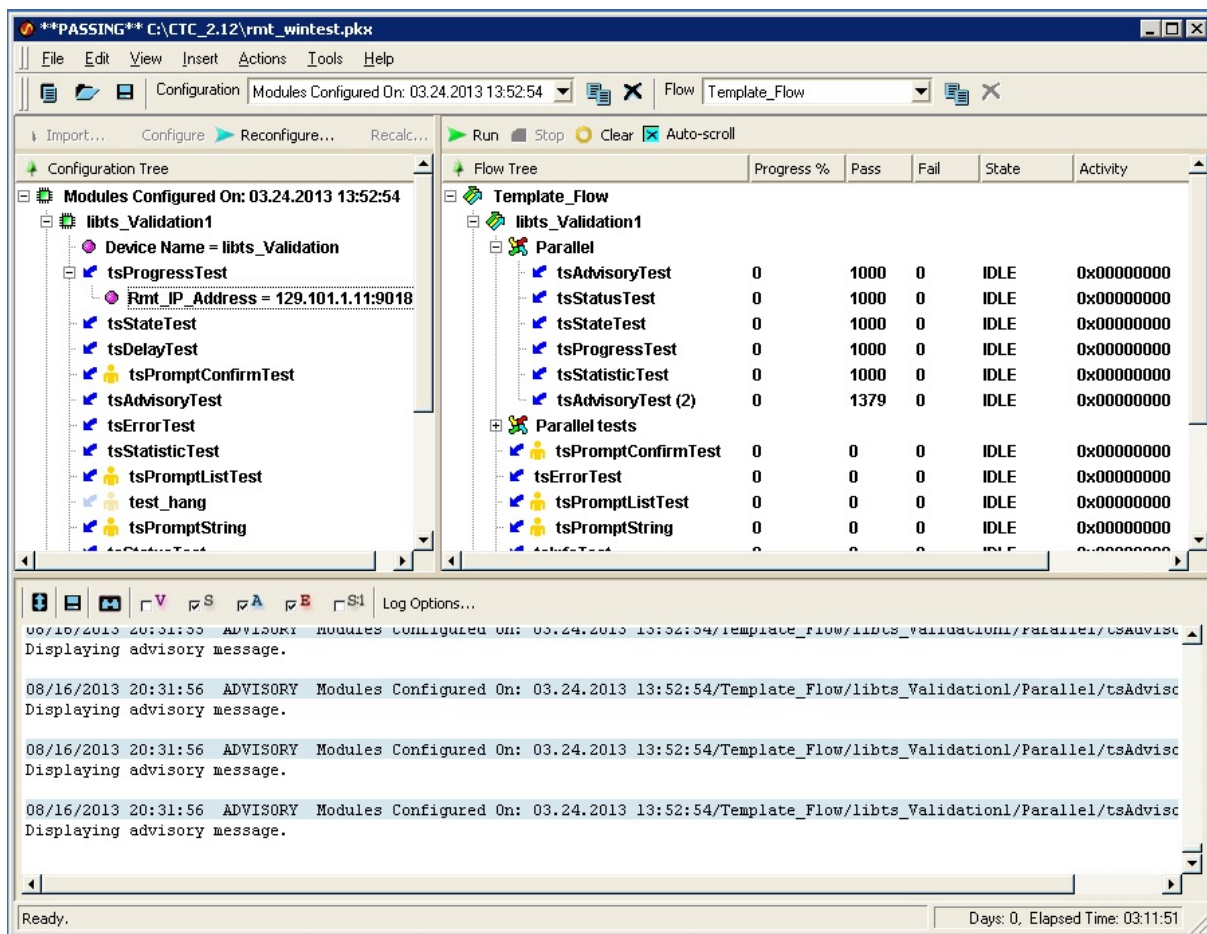
Rmt\_IP\_Address=172.21.133.88  
 Rmt\_IP\_Address=172.21.133.88:9090  
 Rmt\_Connect\_Retry=50  
 client. (optional)  
 Rmt\_Packet\_Size=800  
 packet. (optional)  
 Rmt\_Test\_Delays=200  
 (optional)

- default port number 8080 will be used.
- port number 9090 will be used.
- number of attempts to connect to remote
- number of bytes will be sent in a single TCP
- delay in milliseconds within the test cycle.

Entire PKX will run remotely:

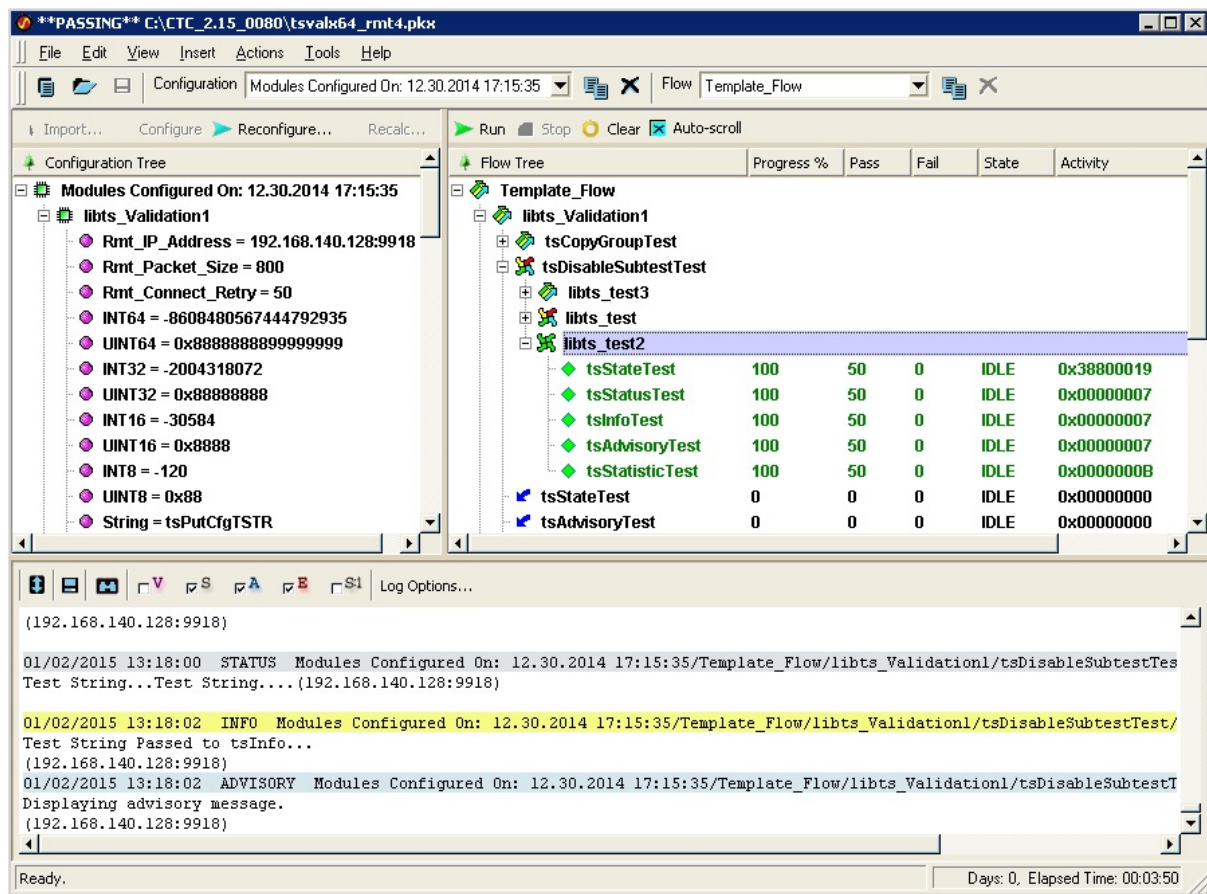


Only tsProgressTest will run remotely:



Customized packet size:





## 9.3 Remote pkx setup for multi nodes

Remote CTC do support multi nodes / systems in a single pkx file. The multi nodes pkx setup example is shown as below:

- 1.) Set the *Rmt\_IP\_Address* parameter with the particular node's IP & port number (node 1 in picture 1 & node 2 in picture 2).
- 2.) Create a different groups at the flow tree for each nodes / systems, the groups can be any type of serial, parallel or parallel exact.
- 3.) Put all the sub-tests that will be executed on the particular node / system to the test groups that created earlier.
- 4.) Local node tests can be executed simultaneously with other remote nodes tests.

File Edit View Insert Actions Tools Remote Help

Configuration Modules Configured On: 02.14.2015 01:01:57 Flow Template\_Flow

Import... Configure Reconfigure... Recalc... Run Stop Clear Auto-scroll

Configuration Tree

- Modules Configured On: 02.14.2015 01:01:57
  - Local System
    - Remote System 1
      - Rmt\_Connect\_Retry = 10
      - Rmt\_IP\_Address = 172.21.133.72:32688
      - INT64 = -8608480567444792935
      - UINT64 = 0x8888888889999999
      - INT32 = -2004318072
      - UINT32 = 0x88888888
      - INT16 = -30584
      - UINT16 = 0x8888
      - INT8 = -120
      - UINT8 = 0x88
      - String = tsPutCfgTSTR
      - INT8\_ARRAY [2,2]
      - INT8\_SUCCESS = 11
      - Device Name = libts\_Validation
      - Version = 6.01.1.0001
      - StressTest = no
      - tsEnableTest
      - TestHang
      - tsCopyGroupTest

Flow Tree

	Progress %	Pass	Fail	State	Activity
Template_Flow					
Local_System					
tsAdvisoryTest	100	436	0	IDLE	0x00000007
tsInfoTest	100	436	0	IDLE	0x00000007
tsStateTest	0	435	0	RUNNING	0x38800019
tsStatisticTest	0	435	0	RUNNING	0x00000008
tsStatusTest	0	435	0	RUNNING	0x00000007
Remote_System_1					
tsAdvisoryTest	100	69	0	RUNNING	0x00000007
tsProgressTest	100	94	0	IDLE	0x00000013
tsStateTest	100	52	0	RUNNING	0x38800019
tsStatusTest	0	68	0	RUNNING	0x00000007
tsInfoTest	100	71	0	RUNNING	0x00000007
Remote_System_2					
tsAdvisoryTest	0	66	0	RUNNING	0x00000007
tsStateTest	0	51	0	RUNNING	0x38800019
tsStatusTest	100	64	0	RUNNING	0x00000007
tsInfoTest	0	65	0	RUNNING	0x00000007
tsProgressTest	0	89	0	RUNNING	0x00000013

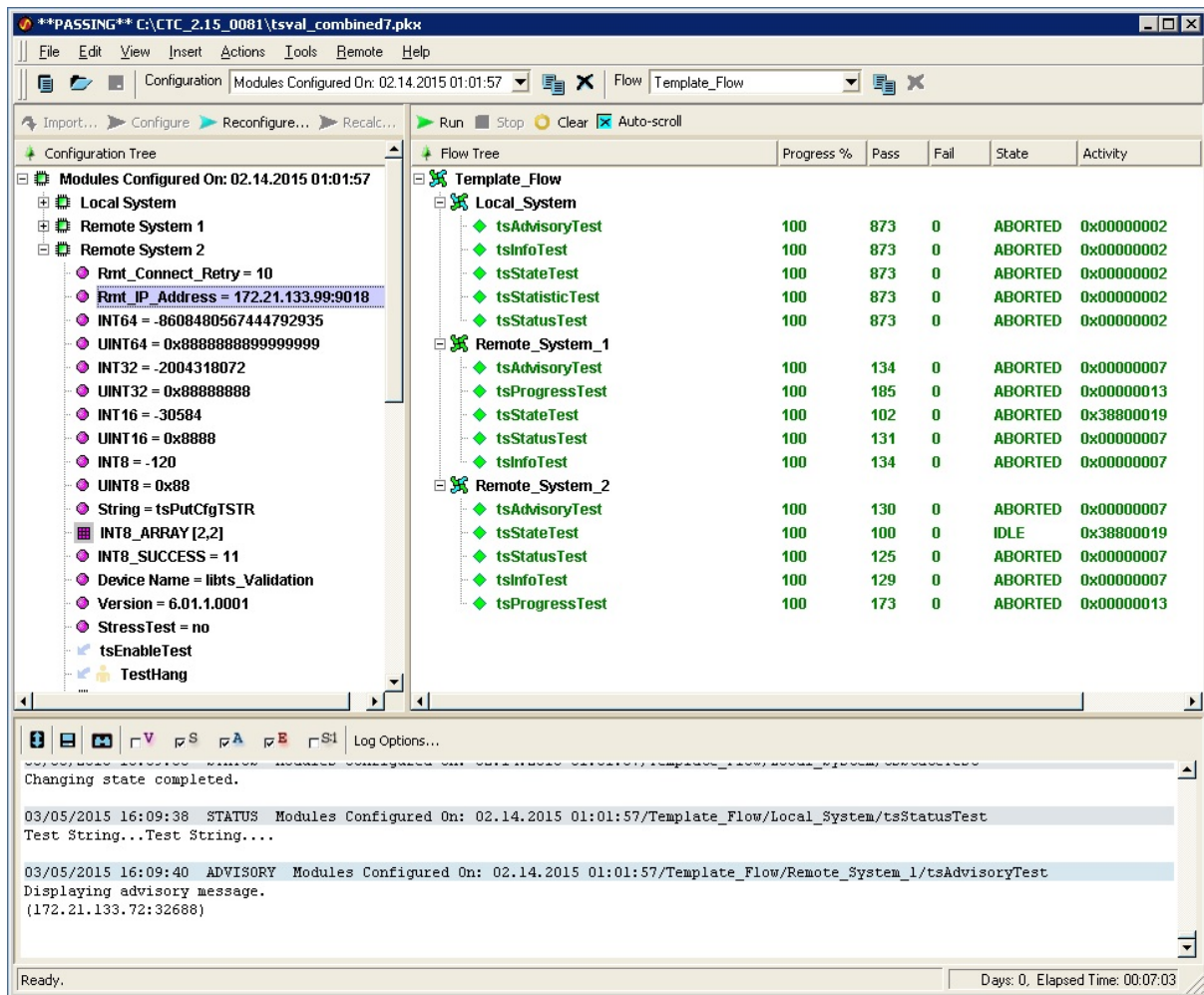
Log Options...

03/05/2015 16:06:14 INFO Modules Configured On: 02.14.2015 01:01:57/Template\_Flow/Local\_System/tsInfoTest  
Test String Passed to tsInfo...

03/05/2015 16:06:14 STATUS Modules Configured On: 02.14.2015 01:01:57/Template\_Flow/Local\_System/tsStateTest  
Changing state completed.

03/05/2015 16:06:14 STATUS Modules Configured On: 02.14.2015 01:01:57/Template\_Flow/Local\_System/tsStatusTest  
Test String...Test String....

Ready. Days: 0, Elapsed Time: 00:03:35



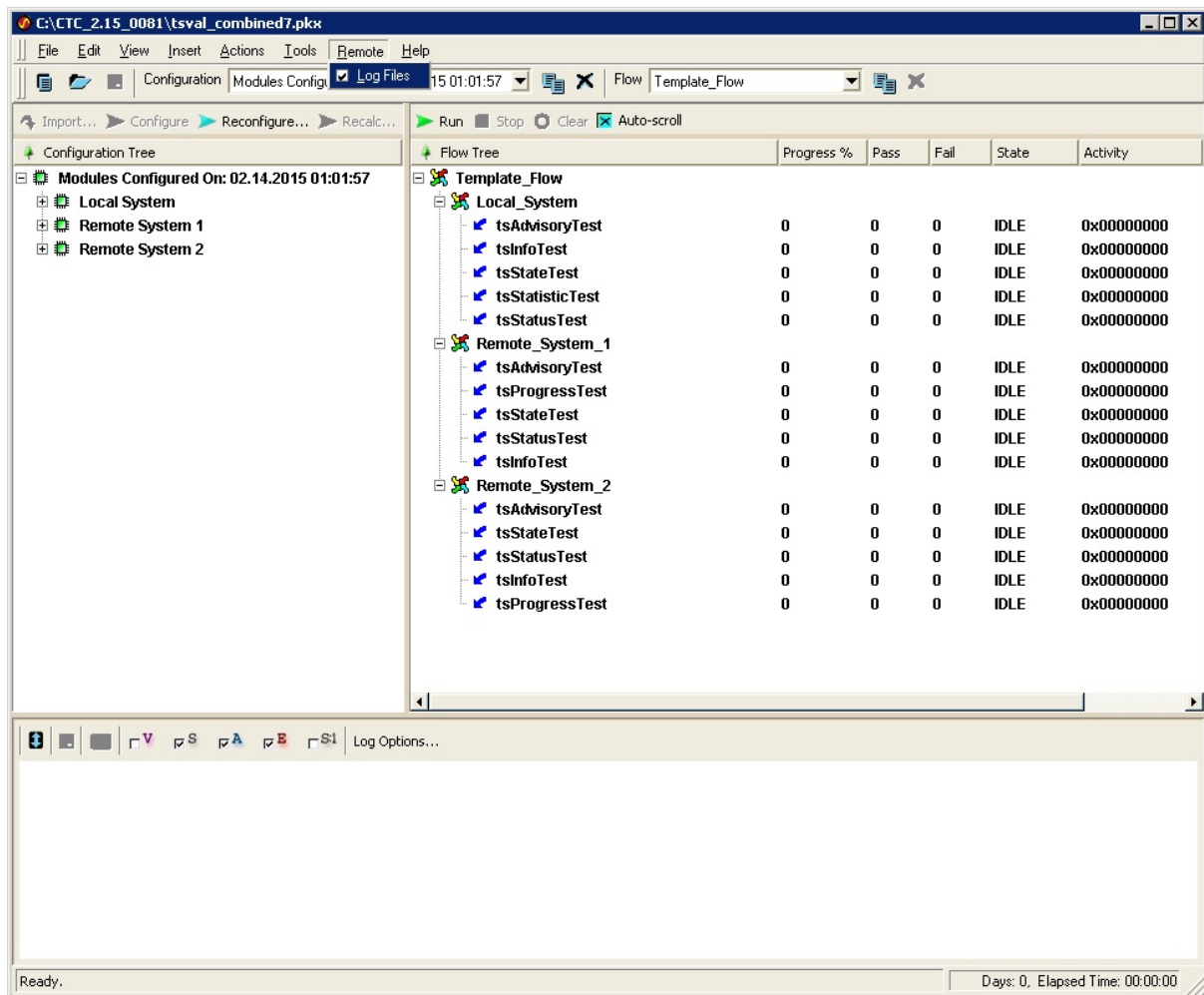
## 9.4 Remote log files

Remote log files can be enabled by checking the *Log Files* checkbox under *Remote* menu pane. Three log files will be generated: *librmthostlog.txt*, *librmthost\_recv\_ts.txt* & *librmthost\_recv\_nts.txt*.

*librmthostlog.txt* - captured all the transaction activities, including parameters, commands & data received from the remote service; for debug used.

*librmthost\_recv\_ts.txt* & *librmthost\_recv\_nts.txt* - only captured the data receiving from the remote service; for validation used.

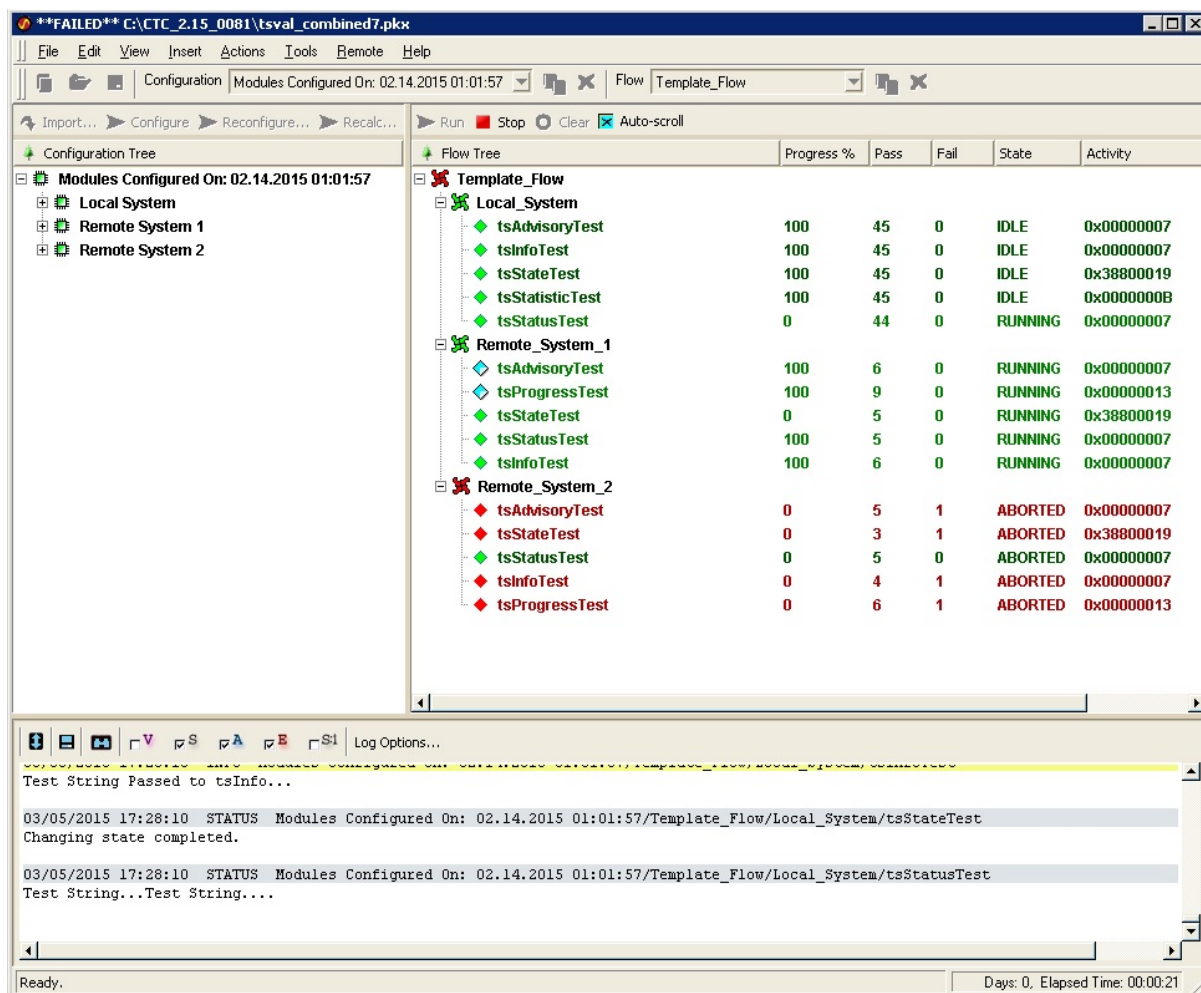




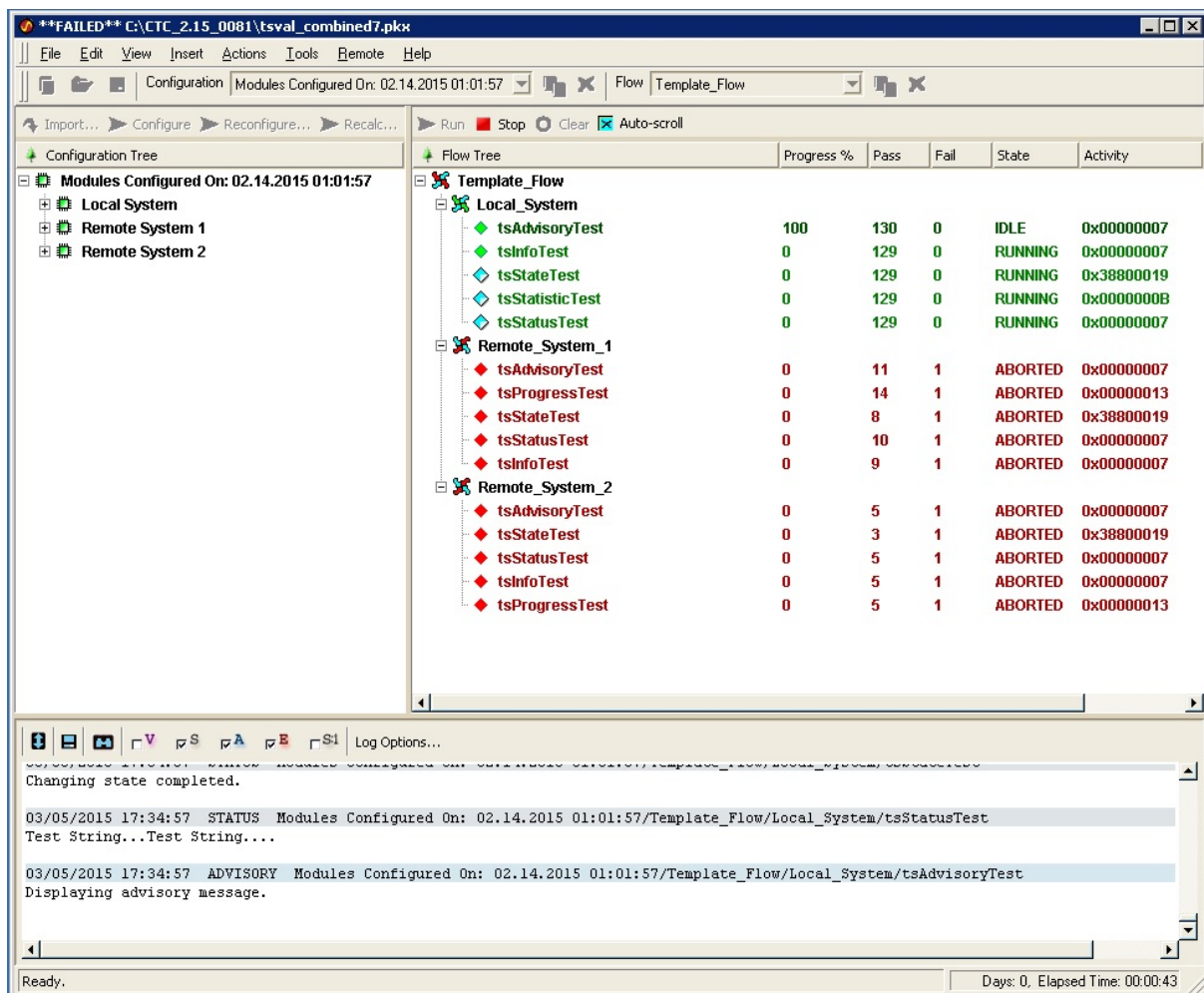
## 9.5 Remote error & exception handling

Error & exception occurred when the network connection is lost in the middle of the remote testing. CTC will identify the failed IP and put it into a failed IP list and aborted all the remote tests tied to this IP. The remaining nodes / systems that are not impacted by the failed IP will continue testing as usual. If more than one remote nodes are losing connection, all the failed IP will be inserted into the failed IP list and these nodes will be aborted. The remaining nodes / systems that are not impacted by the failed IPs will continue testing as usual.

Example 1 - Remote system 2 lost connection, all tests under this node are aborted but others keep running as usual:



Example 2 - Both remote system 1 & 2 lost connection, all tests under these nodes are aborted, only local node is running as usual:



## 9.6 Remote probing over IP range

Remote CTC supports the remote probing feature to automatically discover, clone and configure the remote clients / nodes over the remote IP addresses range defined by the users. Parameter *Remote\_IP\_Range* is required to turn on the remote probing feature for CTC. Users have to manually set this parameter with the targeted remote client IP addresses range & port number. The IP addresses range parameter accepts ' - ' & ' , ' hot keys, use ' - ' to specify the IP addresses range in same subnet and use ' , ' to specify the IP addresses range from different subnet. Port number can be set by putting a ':' following the IP address range, the defined port number will be used across all clients / nodes that discovered from the IP range. If port number is not defined, default port number of 8080 will be used. The examples are shown below.

### Examples:

- |                                    |                                       |
|------------------------------------|---------------------------------------|
| Remote_IP_Range=172.21.133.88      | - single IP with default port number. |
| Remote_IP_Range=172.21.133.88:9090 | - single IP with port number 9090.    |
| Remote_IP_Range=172.21.133.88-100  | - IP range from 88 to 100.            |

100 with same subnet with default port number.

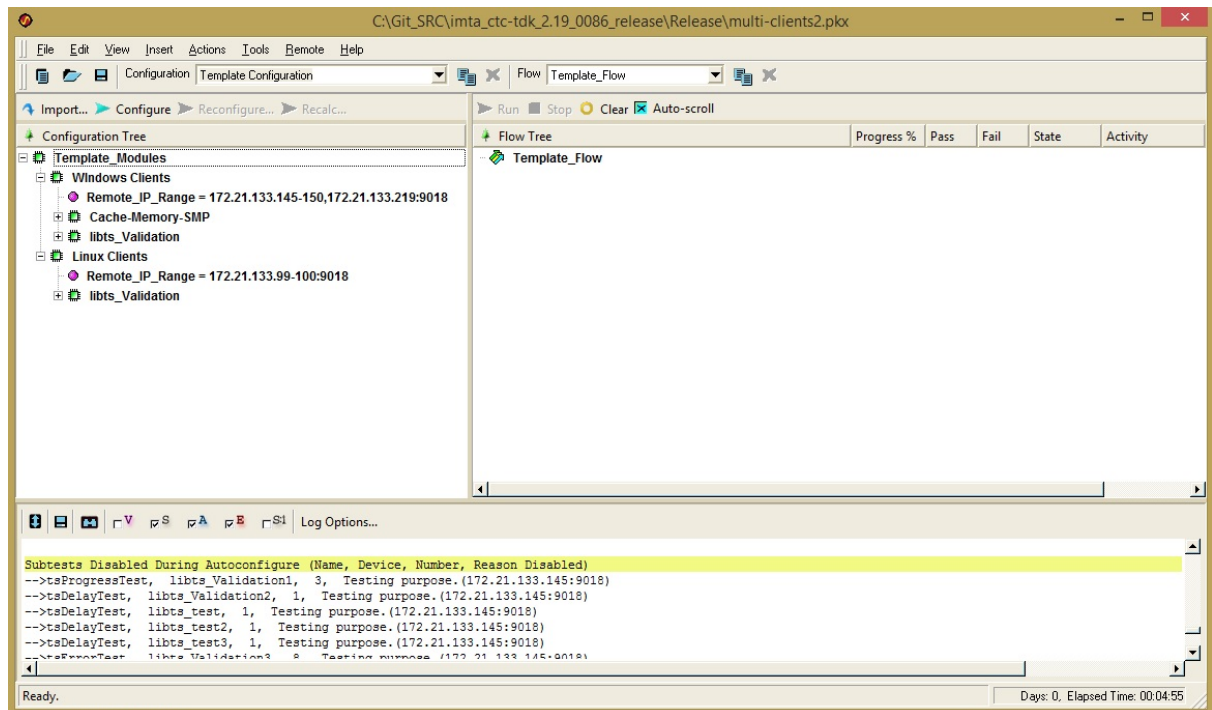
Remote\_IP\_Range=172.21.133.88-100,172.21.131.88-100 - IP range with different subnet with default port number.

Remote\_IP\_Range=172.21.133.88-100,172.21.131.88-100:9090 - IP range with different subnet with port number 9090.

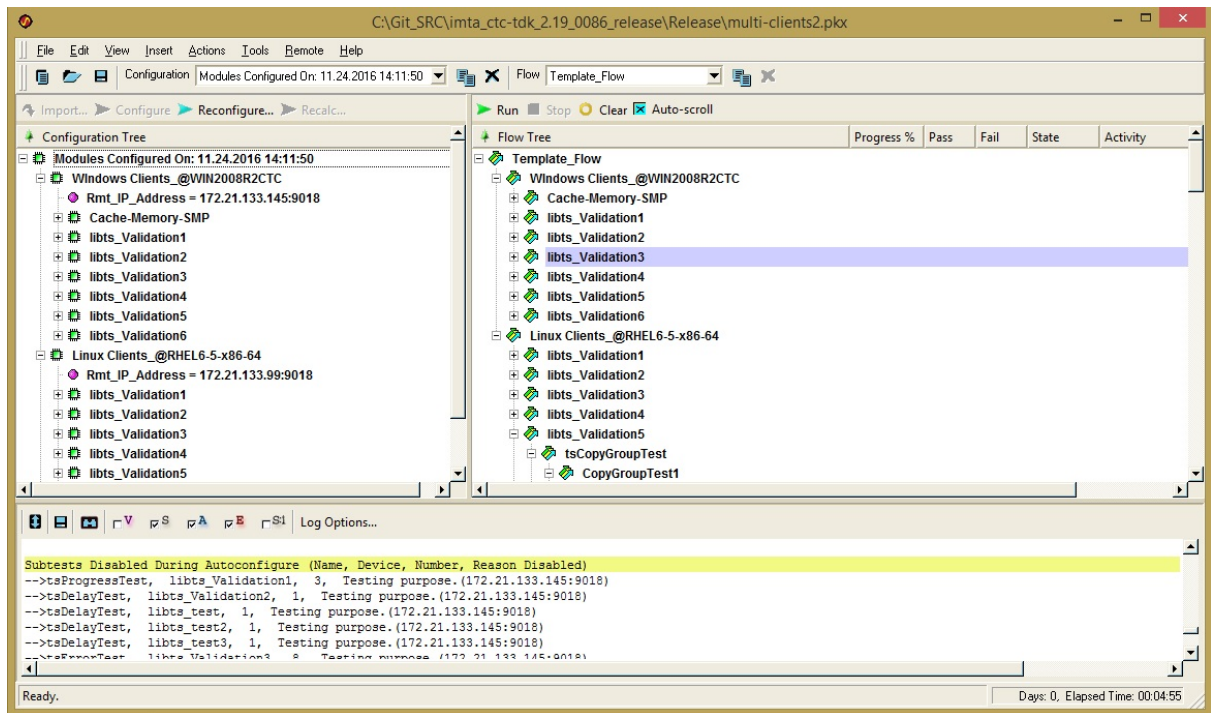
Remote\_IP\_Range=172.21.133.88,172.21.131.88,172.23.133.88 - Multiple IP addresses from different subnet with default port number.

Each clients that being discovered from the IP range will be configured and cloned automatically with client's computer name appended as the new node's name (refer to picture 2 below for more details). Users are recommended to create a new device as the client's node, insert the *Remote\_IP\_Range* parameter and all the test modules into the client's node to make the pkx looks more organized as the cloned will be happened at the client's node layer (refer to picture 1 & 2 below for more details). The remote probing pkx example is shown as below:

Before Auto Configure:



After Auto Configure:



# Index

## - A -

About 59  
 Action Menu 46  
 add  
   device 45  
   parameter 46  
   parameter array 46  
   parameter group 46  
   test 45  
   test group 46  
 Autoconfigure 23

## - B -

Basic concepts 20  
 button  
   advisory messages 78  
   auto-scroll 71  
   clear 71  
   clone configuration 59  
   clone flow 59  
   configure 62  
   configure buttons 60  
   delete configuration 59  
   delete flow 59  
   error messages 78  
   import 48, 62  
   maximize messages 75  
   message buttons 74  
   messages 77  
   new file 59  
   open 59  
   recalc 50, 67  
   reconfigure 48, 65  
   run 71  
   save 59  
   save messages 76  
   search messages 77  
   sequence buttons 71  
   statistic messages 78  
   status messages 78  
   stop 71

verbose messages 79

## - C -

### Cmd Line

/AUTO 96  
 /CFG 97  
 /DLL 97  
 /FD 97  
 /FLOW 98  
 /FOREVER 98  
 /HE 98  
 /LF 98  
 /LOOPS 98  
 /MAX 99  
 /MINUTES 99  
 /N 99  
 /PASSFILE 99  
 /PC 99  
 /PKG 99  
 /PORT 100  
 /Q 100  
 /QUIT 100  
 /QUITPASS 100  
 /RA 102  
 /RE 102  
 /RI 102  
 /RP 102  
 /RR 102  
 /RS 102  
 /RT 102  
 /RUN 101  
 /RV 102  
 /SELF 101  
 /SF 102  
 /SUMMARY 101  
 /V 101  
 /W 101  
 Autoconfiguration 96  
 Bypass Interactive Tests 101  
 Configuration 97  
 Configuration Flow Trees 98  
 Create File if Tests Pass 99  
 Exit after testing 100  
 Exit after tests pass 100  
 Flow Trees 98  
 Halt on Error 98  
 Load Custom DLL 97

Cmd Line  
 Log File Message Redirection 98  
 Max Window Size 99  
 Message Redirection Flags 102  
 Pass Dialog 97  
 Quiet Mode 100  
 Run Forever 98  
 Run Number of Loops 98  
 Run Number of Minutes 99  
 Run package file 99  
 Run Product Code 99  
 Run tests 101  
 Select Comm Socket 100  
 Self Configuration 101  
 Summary File 101  
 Verbose Mask 101

Cmd Line Parameters 95

configuration  
 clone 43  
 configure 62  
 import 48, 62  
 recalc 50, 67  
 reconfigure 48, 65  
 remove 43  
 select 59  
 switch 43

Configuration File 30  
 Configuration Property Sheet 84  
 Configuration Template 24  
 Configuration Tree 25  
 Configuration Wizard 22  
 content 11  
 CTC 59

## - D -

Device 26  
 Device Node 45  
 Device Property Sheet 88  
 Dialog  
 message log options 52, 79  
 Document Files 30

## - E -

edit  
 configuration property 84

device property 88  
 flow group property 90  
 parameter array property 93  
 parameter group property 92  
 parameter property 93  
 test group property 90  
 test property 85  
 edit control 69, 72  
 Edit Menu 41  
 Execute Tests 47  
 Executive Parameters 102, 104  
 \$AUTOCONFIG 102, 104  
 \$EXEFILE 102, 111  
 \$HELPPFILE 103  
 \$TESTNUMBER 103  
 \$VERBOSEMASK 103  
 Exiting 40

## - F -

File Menu 37  
 files  
 CSV 31  
 DTD 31  
 language localization 34  
 managing 40  
 message redirection 31  
 new 38  
 open 38  
 package 30  
 PKX 30  
 recent 39  
 save 40  
 saving 40  
 snippet 30  
 SNX 30  
 summary 31

Flow  
 clone 44  
 new 44  
 remove 44  
 switch 43  
 flow control 28  
 Flow group 28  
 Flow Group Node 46  
 Flow Group Property Sheet 90  
 flow selection 59  
 Font Options 55

## - G -

groups 28  
GUI 35

## - H -

help  
    about 59  
    node (F1) 68, 72  
    node help 69, 72  
    options 53  
    test 69, 72  
Help Menu 58  
Help Options 53

## - I -

Icon Definitions 69, 73  
Insert Menu 45  
Inserting 45

## - L -

language localization 59  
Language Localization File 34  
language selection 59  
license  
    options 57  
License Options 57  
Log Options 52, 79

## - M -

Maximize Message View 75  
Menu  
    action 46  
    edit 41  
    file 37  
    help 58  
    insert 45  
    toolbar 37  
    tools 53  
menu bar 37  
message

types 28  
message filter  
    advisory 78  
    error 78  
    statistic 78  
    status 78  
    verbose 79  
Message Redirection 52, 79  
Message Redirection File 31  
Message Types 28  
Module 26

## - N -

New File 38  
node  
    configuration property 84  
    device 45  
    device property 88  
    flow group property 90  
    parameter 46  
    parameter array 46  
    parameter array property 93  
    parameter group 46  
    parameter group property 92  
    parameter property 93  
    test 45  
    test group 46  
    test group property 90  
    test property 85  
nodes  
    copying 42  
    cutting 42  
    deleting 42  
    duplicating 42  
    managing 42, 44  
    pasting 43  
    properties 44  
    redoing 42  
    removing 42  
    reversing 42  
    undoing 42

## - O -

Open File 38



## - P -

- Package File 30
- Package Property 44
- Package Version 44
- Parameter 27
- Parameter Array 28
- Parameter Array Node 46
- Parameter Array Property Sheet 93
- Parameter Group 28
- Parameter Group Node 46
- Parameter Group Property Sheet 92
- Parameter Node 46
- Parameter Property Sheet 93
- PKG 30
- PKX 30
- Property Sheets 84
  - Configuration 84
  - Device 88
  - Flow Group 90
  - package 44
  - Parameter 93
  - Parameter Array 93
  - Parameter Group 92
  - Test 85
  - Test Group 90

## - R -

- Recent Files 39
- Return Codes 95
- Run Tests 47

## - S -

- save
  - messages 76
- Save As 40
- Save File 40
- select
  - configuration 59
  - flow 59
  - language localization 59
- Sequence Tree 25
- Snippet File 30
- SNP 30

- SNX 30
- Start 47
- Startup 20
- Startup Wizard 20
- status bar 36
- summary 31
- switch
  - configuration 43
  - flow 43

## - T -

- Template
  - configuration 24
- Test 27
- Test Group Node 46
- Test Group Property Sheet 90
- Test Node 45
- Test Property Sheet 85
- tests
  - starting 47
- title bar 36
- tool bar 59
- Tools Menu 53
- Tree
  - configuration 25
  - sequence 25

## - U -

- User Interface 35

## - V -

- version
  - package 44
- views
  - configuration 60
  - flow 71
  - message 74
  - sequence 71

## - W -

- window
  - application control 36
- windows

## windows

configuration 60  
flow 71  
message 74  
sequence 71

**- X -**

## XML

DTD 31  
package 30  
PKX 30  
snippet 30  
SNX 30  
XML DTD File 31