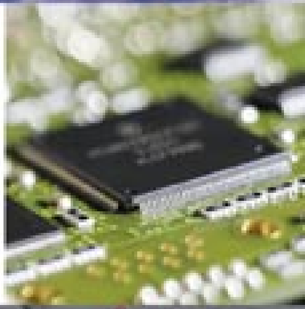
hitex
DEVELOPMENT TOOLS

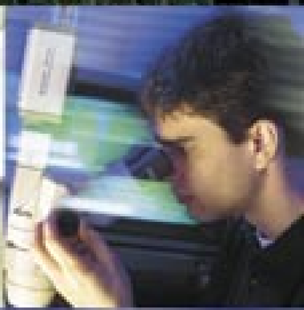
Tessy

Automated, dynamic unit/module test for embedded applications

CTE

Classification Tree Editor for test case specifications





Embedding Software Quality

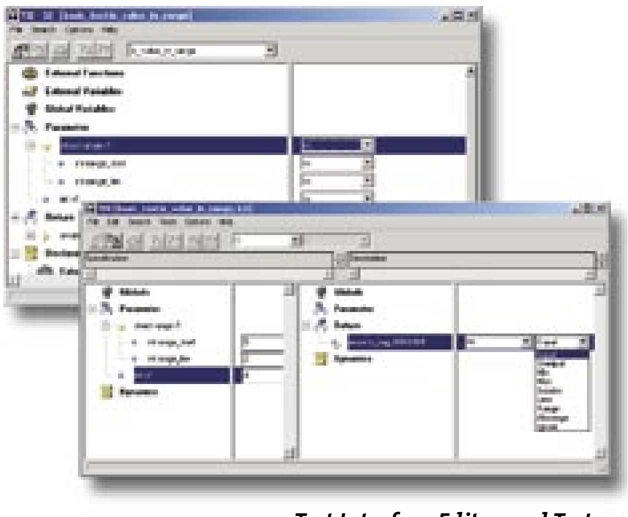
Automated Unit Testing And Debugging At Its Best

Tessy – The Invaluable Test Tool

Tessy performs automatic dynamic unit testing of embedded software and facilitates all other aspects of software testing. It saves an embedded development project tremendous amounts of time, particularly when it comes to regression testing. Regression testing itself is a key feature in achieving software quality.

Unleash It For A Test Run

Tessy starts off by analyzing the source module and then lets the user specify the function to be tested. It then identifies parameters of this function, such as inputs, outputs or both and any external or global variables associated with the function and the subroutines within it. Initial results are displayed in the Test Interface Editor (TIE), where they can be modified should this ever become necessary.



Test Interface Editor and Test Definition Editor (parts of Tessy)

State Your Case

Using the Test Definition Editor (TDE), test cases for functions are then defined by specifying values for input parameters, the expected results and how to compare the actual results with the expected results to determine if a test case has passed or failed. The test cases are saved automatically in a database.

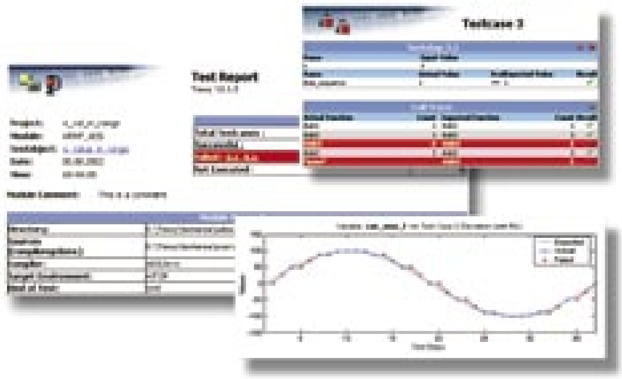
Meet Your Test Driver

Tessy now generates source code for the test driver, which calls the function under test. If this function calls subroutines that are not yet implemented, Tessy is able to create stub functions to replace them. The user may provide source code for a stub function or Tessy can optionally check the values of parameters passed to the stub function. It can also check the order of stub function calls.

Go Tessy Go

Using a suitable compiler, Tessy compiles and links the driver source code, the function under test and any stub functions, and then downloads the resulting executable to the test system. This might be a Hitex in-circuit emulator in stand-alone mode or one connected to a target system. Testing can also be performed on a host PC. Tessy executes each test case and then determines if it has succeeded or failed. Tessy may accept deviations from the exact results. A test report will then be generated in configurable levels of detail and in various formats such as html, doc, xls, or txt. Test results can even be displayed graphically, using MATLAB®.

Why Tessy Eases Testing



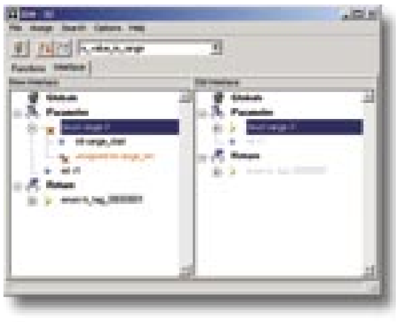
The test documentation can be generated in HTML format

Tessy And HiTOP – Dream Debugging

Using Tessy together with HiTOP, one thing's for sure: if a test case fails, an easy and efficient debugging is guaranteed. Tessy is able to re-execute the test case and have HiTOP set a breakpoint at the function's point of entry. So HiTOP is directed to the beginning of the function under test, with the parameters that caused the wrong result. HiTOP's extremely powerful features and the debug system can now be used to reveal the culprit. Tessy's built-in editor can be used to fix the bug, and the test case can be re-run.

Re-Use Test Data And Save Time

If any interface element of a tested function has been changed in the course of the development process, Tessy allows the user to reassign old interface elements to new ones. Test data from an earlier interface can thus be re-used with the new one, which considerably aids the regression testing process.



Interface Data Assign Editor IDA (part of Tessy)

Regression Testing – Did Your Modifications Cause Errors?

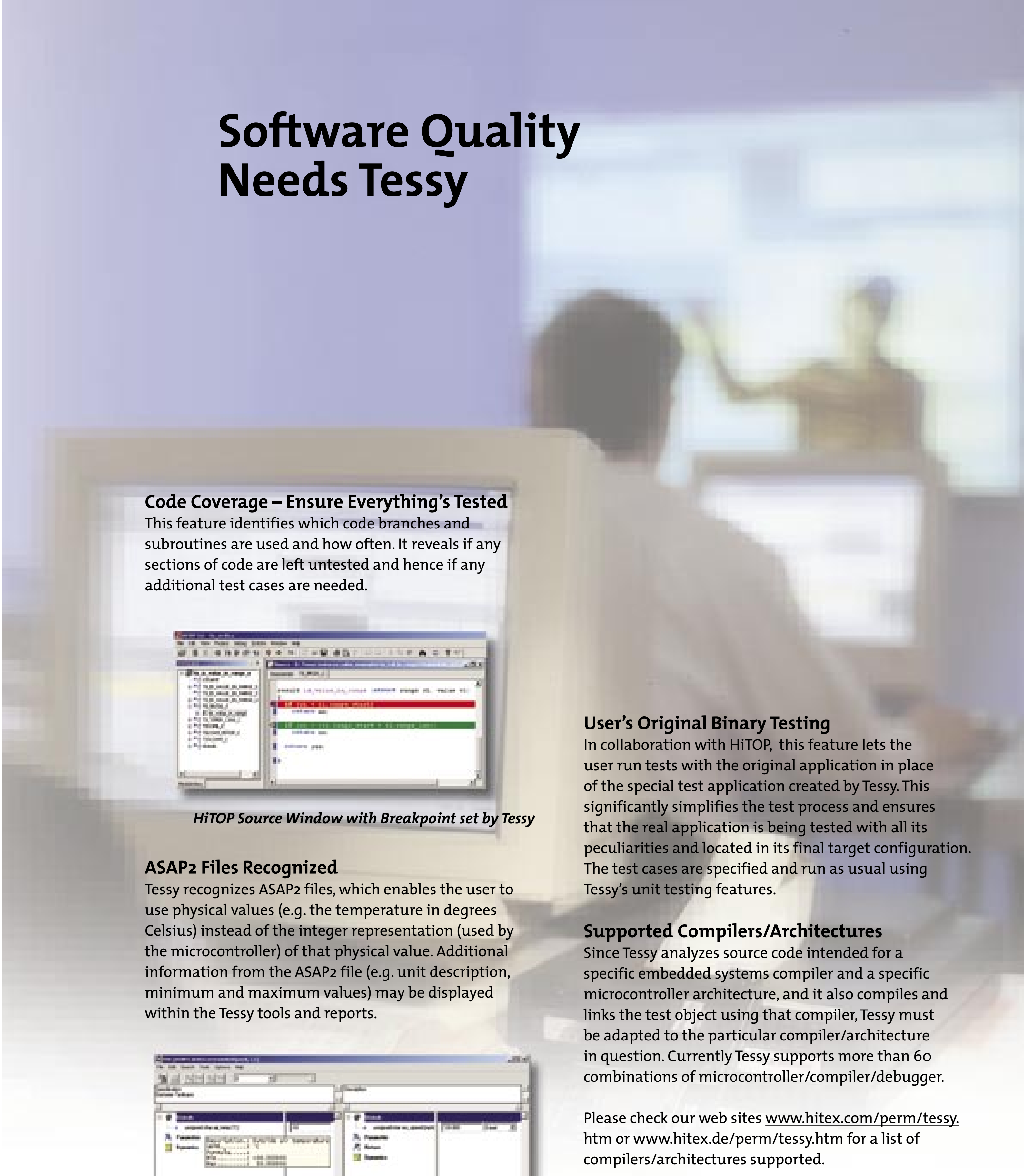
Regression testing can reveal if new errors have been introduced during further development of the application, such as bug fixes in other sections, rewriting of the tested function, switching to a new compiler version or porting the software to another microcontroller architecture. Tessy's easy-to-use regression testing ability is an extremely helpful method of checking modified software and thus ensuring software quality.

Batch Testing – Let's You Go Home

Tessy allows the user to run a selected set of test cases without any user intervention. So an extensive regression test can be run overnight and the results can be analyzed the next day.

Easy Management Of Test Data

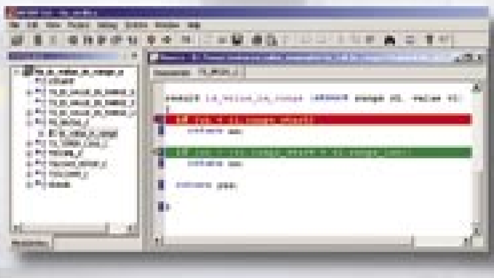
Tessy is able to export and import test data to and from other tools e.g. Excel. This includes data from CTE test case specifications. Furthermore, Tessy is capable of running a test using input parameters only, with no expected results specified. If the actual results of this test match the predicted ones, Tessy may use them as the expected output values for subsequent test runs. Tessy can generate random test data.



Software Quality Needs Tessy

Code Coverage – Ensure Everything's Tested

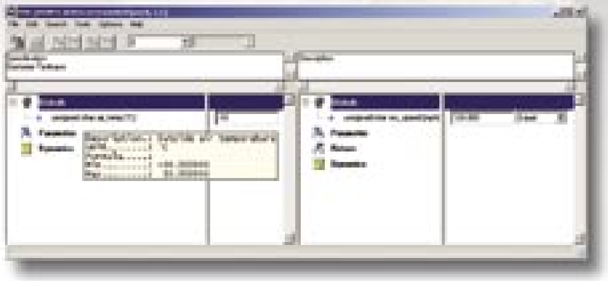
This feature identifies which code branches and subroutines are used and how often. It reveals if any sections of code are left untested and hence if any additional test cases are needed.



HiTOP Source Window with Breakpoint set by Tessy

ASAP2 Files Recognized

Tessy recognizes ASAP2 files, which enables the user to use physical values (e.g. the temperature in degrees Celsius) instead of the integer representation (used by the microcontroller) of that physical value. Additional information from the ASAP2 file (e.g. unit description, minimum and maximum values) may be displayed within the Tessy tools and reports.



ASAP support allows to use physical test values

User's Original Binary Testing

In collaboration with HiTOP, this feature lets the user run tests with the original application in place of the special test application created by Tessy. This significantly simplifies the test process and ensures that the real application is being tested with all its peculiarities and located in its final target configuration. The test cases are specified and run as usual using Tessy's unit testing features.

Supported Compilers/Architectures

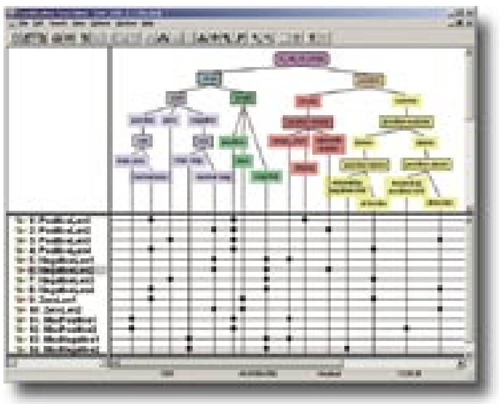
Since Tessy analyzes source code intended for a specific embedded systems compiler and a specific microcontroller architecture, and it also compiles and links the test object using that compiler, Tessy must be adapted to the particular compiler/architecture in question. Currently Tessy supports more than 60 combinations of microcontroller/compiler/debugger.

Please check our web sites www.hitex.com/perm/tessy.htm or www.hitex.de/perm/tessy.htm for a list of compilers/architectures supported.

Tessy runs on WindowsNT/2000/XP machines.

Both Tessy and the CTE originate from DaimlerChrysler's software technology laboratory in Berlin, Germany.

Design Tests For Confidence



Classification Tree Editor (CTE)

CTE And The Classification Tree Method

The Classification Tree Method supports a developer confronted with issues such as:

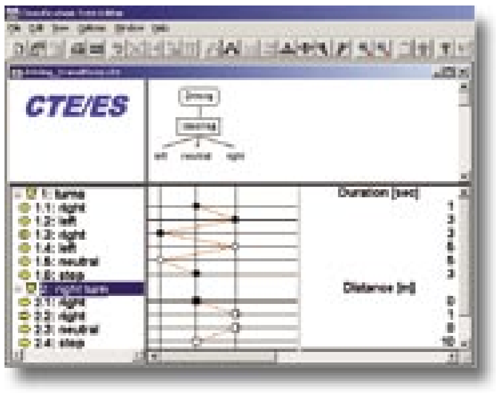
-> Finding the "right" test cases
-> Minimizing a set of test cases while assuring that none are missing
-> Estimating the amount of testing required
-> Defining criteria needed to conclude testing without risking integrity of the test process

The Classification Tree Method transforms a problem specification systematically into a set of error-sensitive, low-redundancy test cases. This method classifies test relevant aspects using the equivalence partitioning method and leads to test case specifications.

This approach is intuitive and easy to learn. It requires and encourages the developer to employ their creativity. Because thinking about the problem specification is at the very beginning, the Classification Tree Method also reveals inconsistencies or omissions in the problem specification.

The Classification Tree Editor (CTE) is a graphical tool that supports the Classification Tree Method. In the upper window, the classification tree is drawn. In the lower window, a line in the combination table specifies a test case. You can annotate information such as state descriptions, expected results, values for classes to the tree as well as to the test case specifications. Bigger trees may be split into sub-trees. The editor is able to export test case specifications to Tessy and in various file formats such as plain text or HTML and to Tessy, of course.

Although CTE is included in Tessy, its use is not only limited to embedded systems and it is therefore available as a separate product also.



Specification of test sequences in the CTE

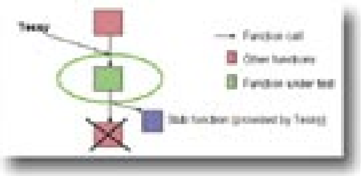
hitex
DEVELOPMENT TOOLS

Tessy/CTE

- -> **Jump start into testing**
- -> **Test data management**
- -> **Automated test execution**
- -> **Seamless integration with HiTOP**
- -> **Powerful regression testing**
- -> **Unlimited number of test cases**
- -> **Test coverage**
- -> **Test documentation**

What is Unit Testing?

During unit testing of C programs, a single C-level function is tested rigorously and in isolation from the rest of the application. *Rigorous* means that the test cases are specially made for the unit in question and that they comprise of input data that may be unexpected by the unit under test. *Isolated* means that the test result does not depend on the behavior of the other units in the application. Isolation from the rest of the application can be achieved by directly calling the unit under test and replacing calls to other units by stub functions.



What are the Benefits of Unit Testing?

Reduces Complexity of Test Case Specification
Instead of trying to create test cases that test the whole set of interacting units, the test cases for unit testing are specific to the unit under test (*divide-and-conquer*). Test cases can easily comprise of input data that is unexpected by the unit under test or by even random input test data, something which is hard to achieve if the unit under test is called by a fully-functioning unit located elsewhere in the application.

Easy Fault Isolation
If the unit under test is tested in isolation from the other units, detecting the cause of a failed test case is easy. The fault must be related to the unit under test, and not to an unit further down the calling hierarchy.

Finds Errors Early
Unit testing can be conducted as soon as the unit to be tested compiles successfully. Therefore errors inside the unit can be detected very early.

Saves Money
It is generally accepted that errors detected late in a project are more expensive to correct than errors that are detected early. Hence unit testing saves money.

Gives Confidence
Unit testing gives confidence. After the unit testing, the application will be made up of single, fully tested units. A test for the whole application will then be more likely to pass. If some tests fail, the reason will probably stem from the interaction of the units and not from an error inside a particular unit. The search for the failure can concentrate on this and need not question the internal operation of the units.

Visit us on the internet! www.hitex.com or www.hitex.de

Main Office Germany

Greschbachstraße 12	Tel.	+49-721-9628-0
D-76229 Karlsruhe	Fax	+49-721-9628-149
	E-mail	sales@hitex.de

Hitex USA

2062 Business Center	Tel.	800-45-HITEX
Drive, Suite 230	Tel.	+1-949-863-0320
Irvine, CA 92612	Fax	+1-949-863-0331
	E-mail	info@hitex.com

Hitex UK

Warwick University	Tel.	+44-24-7669-2066
Science Park	Fax	+44-24-7669-2131
GB-Coventry CV4 7EZ	E-mail	info@hitex.co.uk

Hitex Asia

25 International	Tel.	+65-6566-7919
Business Park, #04-62A	Fax	+65-6563-7539
German Centre	E-mail	
Singapore 609916		sales@hitexasia.com.sg

This brochure is intended to give overview information only. Since our policy is one of continuing development, changes and technical enhancements are possible. Trademarks of other companies used in the text refer exclusively to the products of these companies. Hitex, HiSIM, HiTOP, DProbe, JProbe, USB Agent, Tanto and Tantino are trademarks of Hitex Development Tools GmbH. Copyright ©2004.

Embedding Software Quality

B0-Tessy.indd July 2005-004