

# Tessy

## Frequently Asked Questions (FAQs)

### Contents

#### General

- Q1** What is the main objective of Tessy?
- Q2** What is a unit for Tessy?
- Q3** What is unit testing?
- Q4** Can Tessy be used for module testing according to IEC 61508 / DIN EN 50128?
- Q5** Can Tessy be used for testing according to DO178B?
- Q6** Can I have the complete testing done as a service?
- Q7** What about test coverage by Tessy?
- Q8** Can Tessy test state machines and filter functions?
- Q9** Can Tessy test RTOS applications?
- Q10** Can Tessy be used for integration testing?
- Q11** Where does Tessy execute the tests?
- Q12** Can Tessy be used without execution environment for a microcontroller?
- Q13** Can Tessy be used with simulators?
- Q14** Can Tessy export / import test data?
- Q15** Can Tessy be used for regression testing?
- Q16** Which test report formats are available?
- Q17** What operating systems are supported?

#### Licensing

- Q18** What license models are available?
- Q19** What is included in a license?
- Q20** Can I check out a license?

#### After Purchase

- Q21** Is a maintenance contract available and what does it provide?
- Q22** What is the Tessy User Day?
- Q23** Is training available?
- Q24** Is consulting available?

#### Technique

- Q25** What happens if the function under test itself calls subroutines?
- Q26** Can two functions under test use different stub functions?
- Q27** Are stub functions possible for called subroutines inside the same module?
- Q28** Can a stub function provide different return values?
- Q29** Which data types are supported for input/output variables?
- Q30** Which storage classes are supported for input/output values?
- Q31** How does Tessy find include files?
- Q32** How does the test application interact with peripheral registers (SFRs)?
- Q33** Can Tessy test static functions?
- Q34** Can Tessy generate random test data?
- Q35** Is full specification of the test data required?
- Q36** What relations are available for the evaluation of the output data?
- Q37** Why are test cases further subdivided into test steps?
- Q38** Can test cases be copied?
- Q39** Does Tessy allow the insertion of user supplied code in the test application?
- Q40** Can Tessy test assembler code?
- Q41** Are test results logged continuously?
- Q42** Is it necessary to adapt Tessy to embedded compilers?
- Q43** Does Tessy support C++?

#### Test Case Specification

- Q44** What is the Classification Tree Editor (CTE)?
- Q45** What is the Classification Tree Method?
- Q46** What is the main benefit of the Classification Tree Method?
- Q47** Why is the CTE offered as a separate product?
- Q48** How are CTE and Tessy connected?

## General

**Q1** What is the main objective of Tessy?

**A1** Tessy automates the unit testing of embedded software. Tessy provides the input parameters, executes the tests and evaluates the results. In addition, Tessy manages the test data and documents the test results. Furthermore, Tessy states the test coverage.

**Q2** What is a unit for Tessy?

**A2** Tessy considers functions in the sense of the C programming language as units to be tested.

**Q3** What is unit testing?

**A3** During unit testing, the unit under test is tested intensively in isolation from the rest of the application. Unit testing is also called module testing, and sometimes also component testing.

**Q4** Can Tessy be used for module testing according to IEC 61508 / DIN EN 50128?

**A4** Yes. If the programming language is C, a function in the sense of C can be considered as a module according to IEC 61508 / DIN EN 50128. Tessy is already used successfully to achieve certifications according to the above mentioned standards.

**Q5** Can Tessy be used for testing according to DO178B?

**A5** Yes. Tessy is already used successfully in aerospace applications. A Tessy Tool Qualification Document is available on request.

**Q6** Can I have the complete testing done as a service?

**A6** Yes. Hitex provides testing by Tessy as a service.

**Q7** What about test coverage by Tessy?

**A7** Tessy provides test coverage by determining the C1 code coverage of the tests. C1 code coverage is also called branch coverage or decision coverage.

Starting with Tessy V2.5, Tessy also provides C2 coverage: Multiple Condition Coverage and Modified Condition / Decision Coverage (MC/DC).

Tessy instruments the test application to get the coverage information. The code coverage done by Hitex in-circuit emulators is completely independent of the coverage analysis done by Tessy.

**Q8** Can Tessy test state machines and filter functions?

**A8** Yes. State machines and filter functions are examples for functions maintaining an internal state. Tessy is well-suited to test such functions.

**Q9** Can Tessy test RTOS applications?

**A9** Yes. Tessy can replace calls to the real-time operating system (RTOS) by stub functions.

**Q10 Can Tessy be used for integration testing?**

**A10** Tessy can be used to some extent for integration testing. This is automatically the case, if a called subroutine of the function under test is not replaced by a stub function.

**Q11 Where does Tessy execute the tests?**

**A11** Tessy can execute the tests on the host or on the actual target hardware. Normally, a debugger acts as interface to the execution environment.

**Q12 Can Tessy be used without execution environment for a microcontroller?**

**A12** Yes. Tessy can also run the tests completely on the host (Windows PC). In this case, the test application is compiled using the GNU tool set instead of a cross compiler.

**Q13 Can Tessy be used with simulators?**

**A13** Yes. However, Tessy must be adapted to the debugger the simulator is integrated with.

**Q14 Can Tessy export / import test data?**

**A14** Yes. Tessy can import / export test data using pure text files (ASCII) or from / to Microsoft Excel. This allows for using user supplied test data.

Tessy can also import test data from the Classification Tree Editor (CTE).

**Q15 Can Tessy be used for regression testing?**

**A15** Yes, the so-called batch test feature can re-test either a selection of or all test cases of any test object fully automated. It is also possible to run automated tests from the operating system's command line, i.e. without interactive usage of Tessy. This is useful for running regression tests over night.

**Q16 Which test report formats are available?**

**A16** Tessy can produce test reports in XML, Word, Excel, HTML, TXT, and Windows help file format.

**Q17 What operating systems are supported?**

**A17** Tessy runs on Windows NT, 2000, and XP. Future versions will support Vista.

## Licensing

**Q18 What license models are available?**

**A18** Tessy is available with a node-locked license or with a floating license.

The node-locked license restricts the use of Tessy to the PC the license is issued for. The node-locked license is not transferable to another PC. It is not a dongle license.

One or more floating licenses are administrated by a Floating License Server (FLS). At the same time, you may use Tessy as many times as the amount of licenses the FLS disposes over.

**Q19** What is included in a license?**A19** Both node-locked and floating license of Tessy include:

- A license for the Classification Tree Editor CTE.
- Maintenance for 6 months after purchase.
- A license to use all combinations of microcontroller / cross compiler / debugger supported by your Tessy version.

**Q20** Can I check out a license?**A20** Yes. From the Floating License Server, a license can be transferred to another PC for a maximum of 30 days. This allows Tessy to be used on a PC not connected to any network. For the license transfer, it is not necessary for the remote PC to have access to the Floating License Server.

## After Purchase

**Q21** Is a maintenance contract available and what does it provide?**A21** A maintenance contract is available. It provides support (by phone and e-mail) and updates (including updates requiring a new license).

A maintenance contract entitles in the participation of the Tessy User Day.

**Q22** What is the Tessy User Day?**A22** The Tessy User Day is a meeting of Tessy Users. Its intention is to share Tessy expertise, to learn about new Tessy features, and to collect new requirements for Tessy. It is organized by Hitex every 3 to 4 months and usually takes place in Karlsruhe, but also in other locations in Germany. It is held in German language. To find out about the date of the next Tessy User Day, see <http://www.hitex.de/training.html>.**Q23** Is training available?**A23** Training for Tessy is provided by Hitex. Training usually takes place at the customer's site. Training is customized to your requirements. Normally, Training takes between 1 and 3 days. For details, please contact [training@hitex.de](mailto:training@hitex.de).**Q24** Is consulting available?**A24** Consulting for Tessy is provided by Hitex. For details, please contact [training@hitex.de](mailto:training@hitex.de).

## Technique

**Q25** What happens if the function under test itself calls subroutines?**A25** Principally, there are two possibilities: Either the user provides the called subroutine, or Tessy does. If Tessy provides the called subroutine, Tessy generates a so-called "stub" function. Tessy provides two kinds of stub functions: stub functions, in which the user may insert C code; and stub functions, where Tessy checks the value of passed parameters and provides input to the function under test.

If the user provides the called subroutine, it can be done either in source form or in binary form. If the user provides the called subroutine, this is the first step away from module / unit testing towards integration testing.

**Q26 Can two functions under test use different stub functions?**

**A26** Yes. Let two different functions under test both call the same subroutine. Tessy allows having the first function under test use a different stub function than the second function under test. This refers not only to the behavior of the stub function, but also to the kind of stub function (user code vs. parameter check).

**Q27 Are stub functions possible for called subroutines inside the same module?**

**A27** Yes. If a function under test calls a subroutine implemented in the same C source module than the function under test, normally, the linker automatically links that subroutine. However, special handling by Tessy allows having a stub function for the called subroutine in the case at hand.

**Q28 Can a stub function provide different return values?**

**A28** Yes. The behavior of the stub function can be differentiated dependant e.g. on the test case number. If the same stub function is called several times during a test case execution, the stub function can return different values on each call. An application note exists on the latter topic.

**Q29 Which data types are supported for input/output variables?**

**A29** All ANSI-C types like structures, arrays, unions, pointers etc. can be input or output (or both). Of course, the types can be nested, e.g. structures containing pointers pointing to other pointers, linked lists, etc. The more complex a variable is, the more benefit arises from using Tessy. Additionally, non-ANSI-C data types like bit, which are often found in embedded programs, are supported.

**Q30 Which storage classes are supported for input/output values?**

**A30** The parameters of the function and global variables (both variables global to the module and variables global to the program; both external and non-external variables) may be input or output or both. If a pointer points to a non-allocated memory area, Tessy can allocate memory for it. Naturally, the return value of the function is output in every case.

**Q31 How does Tessy find include files?**

**A31** Tessy lets the user specify the options for the embedded compiler used to compile the test application. The user may also specify a separate list of include paths where included header files of the test application can be found.

**Q32 How does the test application interact with peripheral registers (SFRs)?**

**A32** If the test is executed on the target hardware, the normal interaction with the periphery is maintained, i.e. the microcontroller-internal and -external hardware is included in the tests.

Tessy can also “simulate” the periphery by checking values written to and supplying values read from the peripheral registers (SFR). However, this feature depends on the capability of the target platform in use.

**Q33 Can Tessy test static functions?**

**A33** Yes. Normally, functions in C preceded by the keyword `static` are only visible in the C source module where they are defined. This hinders calling static functions from the outside, what is normally required for testing purposes. However, special handling by Tessy enables unit testing for static functions.

**Q34 Can Tessy generate random test data?**

**A34** Yes. The Test Data Editor is able to set variables to random values.

It is also possible to import random test data from an external source (e.g. an Excel sheet).

However, Tessy provides the Classification Tree Method, which is intended to determine systematically a minimal set of error-sensitive test case specifications. The main idea behind the method is to carefully select test cases and test case data, what is opposite to using randomly generated test data.

**Q35 Is full specification of the test data required?**

**A35** No.

Regarding output values: It is possible and it makes sense to do an (initial) test run without having specified the expected results. After the first run, you may check the results and take the resulting values as expected values for consecutive test runs. This is useful if you test e.g. a sorting algorithm, where you can check the sorted result at a glance instead of manually determining the result and then entering it manually.

Regarding the input values: It would make no sense to run a test case with undetermined input values. However, Tessy is able to treat any input variable as "irrelevant" and you can run the test without specifying a value for it.

**Q36 What relations are available for the evaluation of the output data?**

**A36** The actual output data is compared to the expected data. Not only equality, but also inequality, ranges and tests for greater than, less than, etc. can be specified. It is also possible to specify a deviation, either in absolute format (e.g. -0.2) or in relative format (e.g. -3%). Both absolute and relative deviations can be allowed either in positive direction (+), or in negative direction (-), or in both directions (w/o sign). The evaluation can also be suppressed, i.e. it yields o.k. for any output value.

**Q37 Why are test cases further subdivided into test steps?**

**A37** Test cases group test steps, so that several tests are conducted in a certain order. This is most useful, if internal states of a test object shall be maintained between the test steps. Hence, if a test object needs an initialization, or if a test object resembles a state machine, the test step feature provides a solution.

**Q38 Can test cases be copied?**

**A38** Yes. Both test cases and test steps can be copied.

**Q39 Does Tessy allow the insertion of user supplied code in the test application?**

**A39** Yes. Such code can be inserted at various places in the test application, e.g. in the prolog of the test object execution, before each test step, etc. Such code can be used to `#include` files, to `#define` macros/constants or to supply data for characteristic curves, etc.

**Q40 Can Tessy test assembler code?**

**A40** Tessy needs a C interface to the code to test. Tessy cannot recognize variables that are defined using assembler instructions (e.g. ".DB"). Therefore, these variables will not show up in the interface, and therefore, a test case will not include values for these variables. To test pure assembler code with Tessy, a work-around exists: You may write wrapper functions (in C), that hide the assembler code from Tessy. Normally, inline assembler code (of C functions) can be tested without problems, because inline assembler code seldom defines variables.

**Q41 Are test results logged continuously?**

**A41** Yes, the test report is extended automatically at the end of each single test run.

**Q42 Is it necessary to adapt Tessy to embedded compilers?**

**A42** Yes, for two reasons:

- (1) Tessy parses the source code module and must therefore understand compiler and architecture specific keywords like "bit" or "interrupt".
- (2) Tessy compiles and links using an embedded compiler. Therefore, Tessy must know about the compiler and linker (command line options, linker command files, etc.) and the appropriate microcontroller startup code.

A list of supported compilers and architectures is available at <http://www.hitex.de/perm/tessy.htm>.

**Q43 Does Tessy support C++?**

**A43** No. Tessy only supports C at the moment. Support for C++ is planned.

## Test Case Specification

**Q44 What is the Classification Tree Editor (CTE)?**

**A44** A graphical tool supporting the Classification Tree Method.

**Q45 What is the Classification Tree Method?**

**A45** A method to transform a functional problem specification into a minimal and sufficient set of test case specifications.

**Q46 What is the main benefit of the Classification Tree Method?**

**A46** The Classification Tree Method visualizes testing ideas. This is very useful for test reviews and certifications.

**Q47 Why is the CTE offered as a separate product?**

**A47** Because the usability of the Classification Tree Method (and hence the CTE) is not limited to embedded software, but to general (functional) problems. This method can be even applied to problems of everyday live.

**Q48** How are CTE and Tessy connected?

**A48** Test case specifications and test data can be defined in CTE and exported to Tessy. After Tessy has run the tests, the test results are exported from Tessy to CTE.