B/S/H/

Automated Unit-Tests with the Tcl-Test-Engine

Automatisierte Unit-Tests mit der Tcl-Test-Engine

T3000-Presentation

6/3/2015

Thorsten Klein

Overview

- Introduction: Software-Tests
- Test Procedure
- Test-Levels
- Software-Tests at BSH
- Unit-Tests at BSH
- TET vs. TEng
- BSH Test-Engine Layers
- D-BUS2 Messages
- Registration of TCL-variables and commands

Overview

- Overview: BSH-specific TCL-commands
- Handling the Test Engine manually
- Example: Read Data from Target
- Using Test-Scripts
- Creation of Test-Reports
- Automated Test (with TET)
- Problems with the current Test Engine based on TCL
- Potential for improvement
- List of Literature

Introduction: Software-Tests

- Need of correct functionality, but every software includes bugs
  - → Tests already during process of development are of advantage
  - → Localization of bugs by testing

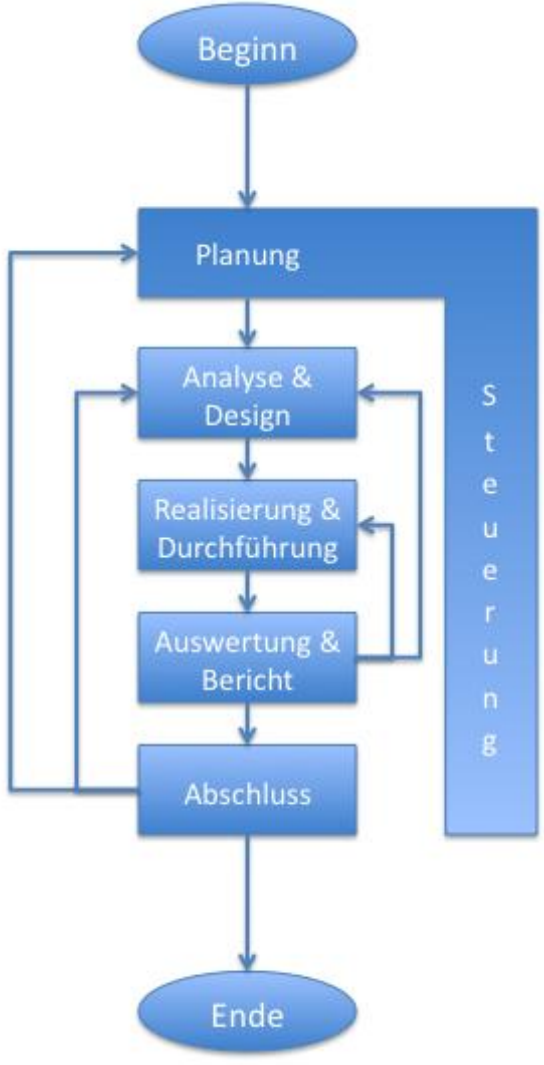
- Fulfilling of standards and norms (ISO)
  - → Validation by testing

- Processing a huge amount of data
  - → Effort: automated tests
    - →automated identification and creation of test cases
    - → automated execution of tests
    - → automated generation of test results and documentation

Test Procedure

1. Planning the test
2. Analyzing (generation of test cases)
3. Realization and Execution
4. Evaluation & Documentation
5. Finalization (certification)

} shall be automated

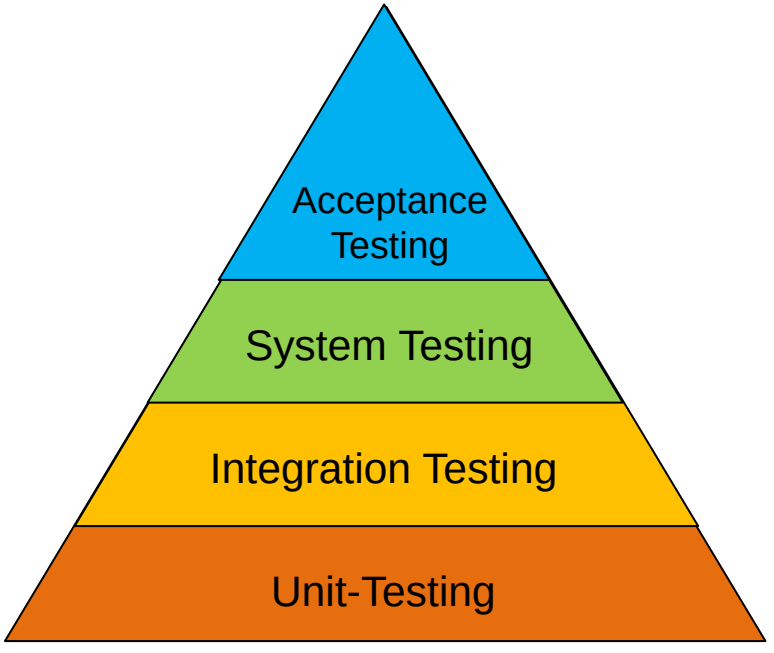


Test-Levels

- **Unit Testing**
  Testing one software module

- **Integration Testing**
  Testing several software modules connected to each other (interaction between modules)

- **SystemTesting**
  Testing the entire software with all modules in association with the target hardware



- **Component Integration Testing / Acceptance Testing**
  Testing the entire system in its final environment

➔ continuous validation of software quality during the process of development

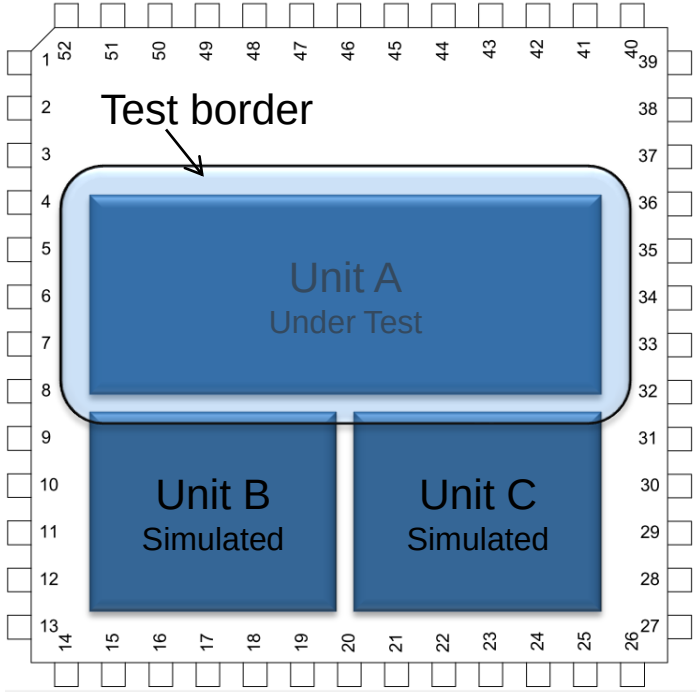
Software-Tests at BSH

Type	What is tested?	What is simulated?	Reference?
Unit-Test	SW-Unit	other SW-Units of the component	SW-module specification, design details
SW-Integration-Test	Software component (all SW-Units)	HW-components	Software specification
Component-Test	component (HW-SW)	System Household Appliance	Software specification
System test	System Household Appliance	Environment	System-specification

Unit-Tests at BSH

- A unit is the smallest piece of software that can be tested in isolation.
  - Separate and Apart from the application and all other units.
  - other units are stubbed (“unit-under-test”)
- Test focuses on the unit itself

  → not on the interfaces
- white box test
- usually performed by the developer.

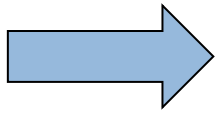


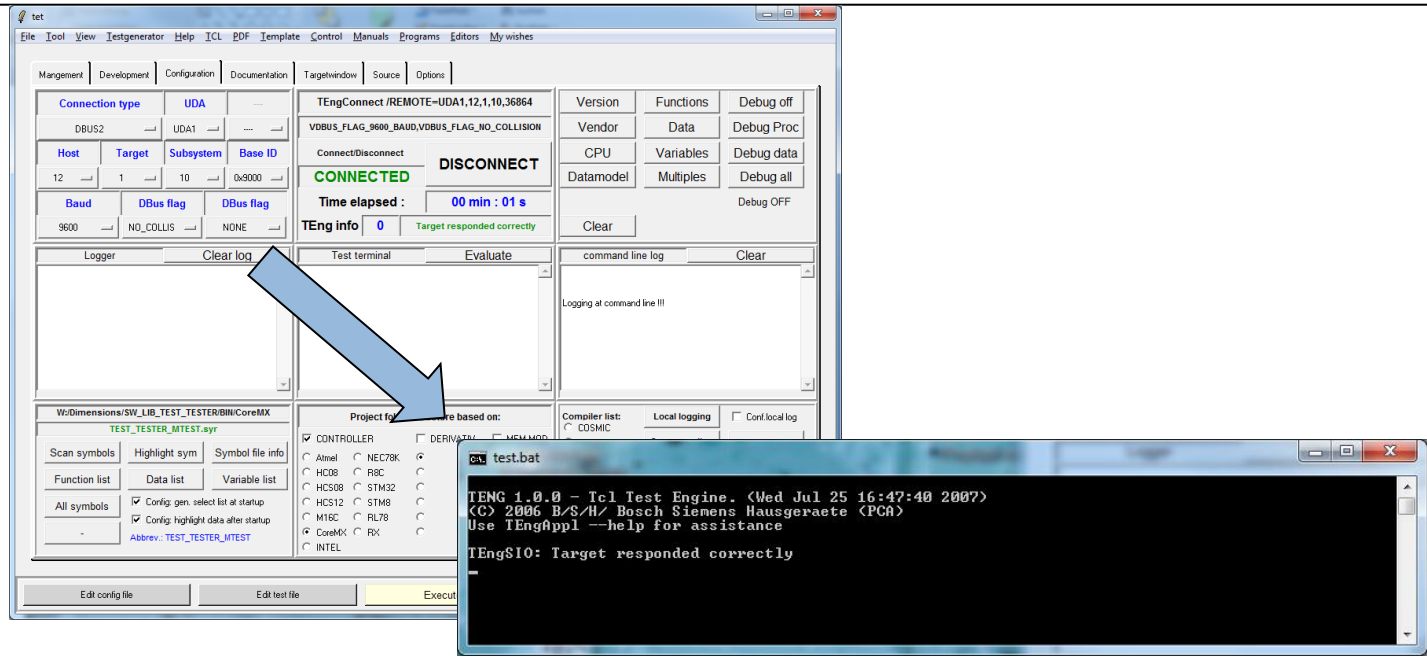
- **Steps:**
  1. Bringing all units in correct states
  2. Running the unit with given input parameters
  3. Comparison between occurring and expected behavior

TET vs. TEng

- TET:

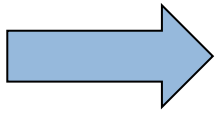
test.bat

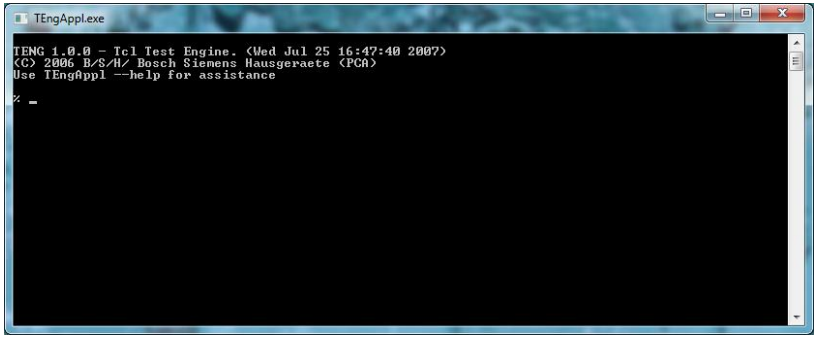




- TEng:

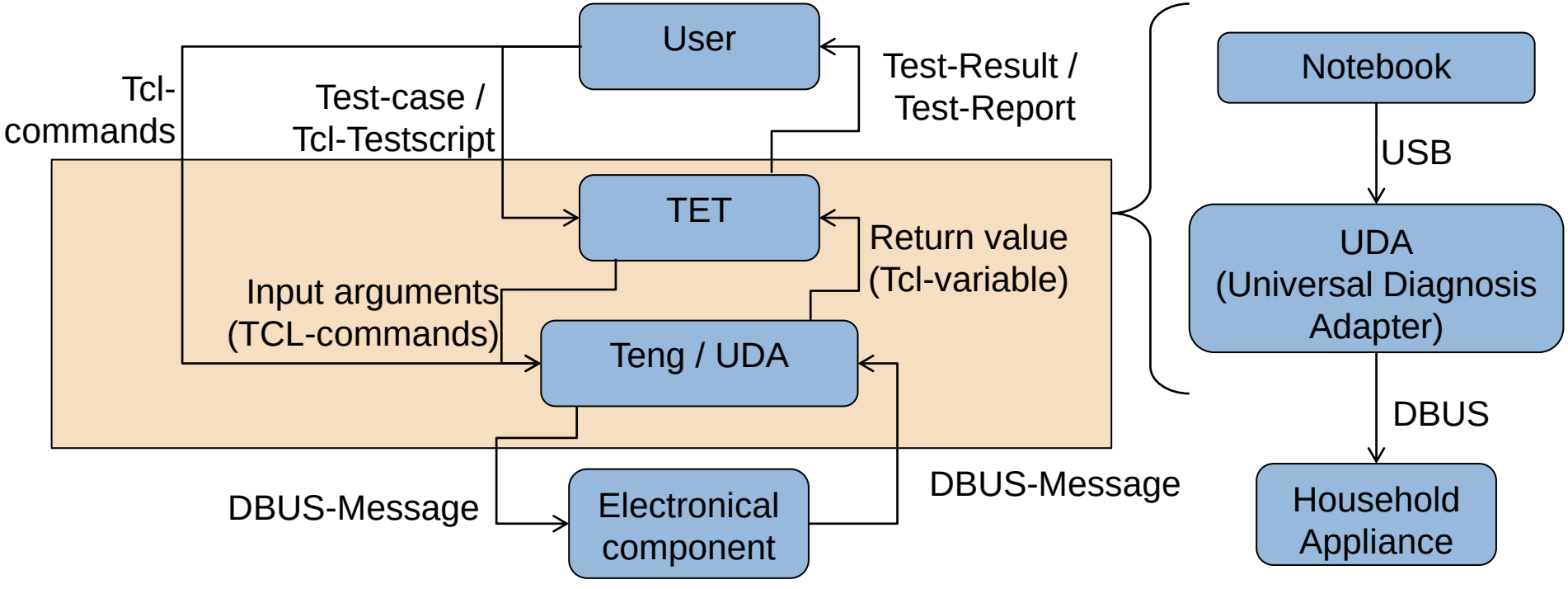
TEngAppl.exe



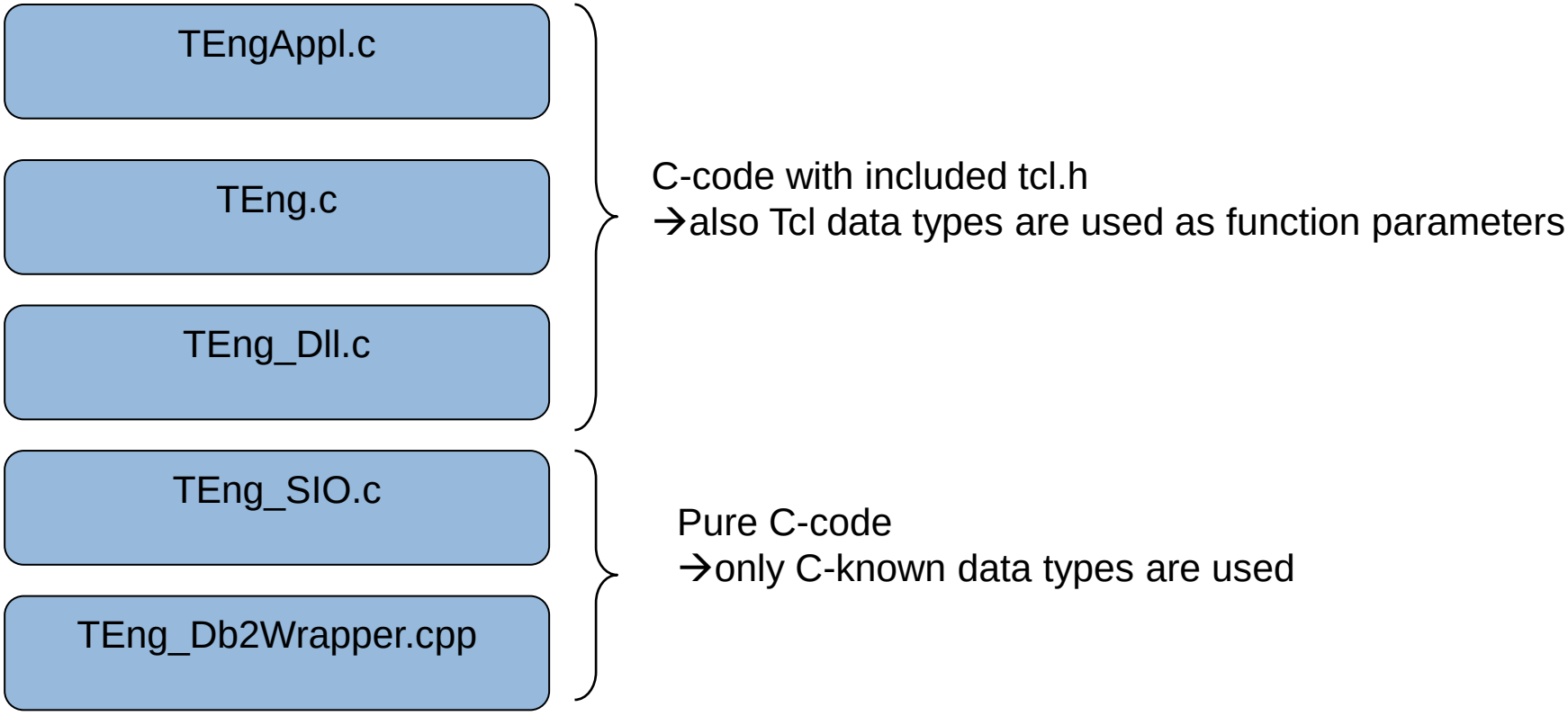


TET vs. TEng

- TEng: Test-Engine → TEngAppl.exe
- TET: TestEngineTool (GUI) → test.bat



BSH Test-Engine Layers



D-BUS2 Messages

- General D-Bus2 Message Structure:

	Frame					
<idle>	**MessageLength**	**Target Address**	**Message (Data)**	**CheckSum**	*<gap>*	***Ack***	*<idle>*
	1 Byte	1 Byte	2..n Byte	16 Bit		1 Byte	

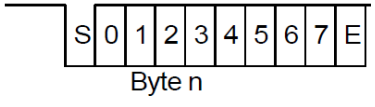
1. Message Length

	MessageLength	FrameLength
Min	2	6
Max	255-4	255

2. Target Address

Bit Number:	Target / Source Address							
	7	6	5	4	3	2	1	0
	Communication Partner				SubSystem			

3. Message



Message-identifier (16Bit)
\+ Message (0 – 249 Byte with Start- and End-Bit each Byte)

4. Checksum

Check sum - CRC	Method
16 Bit CRC	Xmodem 16bit (polynomial 1021h), Start=00

D-BUS2 Messages

- The message identifier is an uint16 value (0 – FFFFh)
- It is divided into different ranges:

F000 - FFFF	Service Messages (F800 - F9FF reserved for proprietary messages)
E000 - EFFF	NMT Messages
A000 - DFFF	Undefined
9000 - 9FFF	After Sales Service Communication objects and Test Messages (automatic tests)
8000 - 8FFF	Remote Control Messages
0000 - 7FFF	System Messages

Registration of TCL-variables

RegVar <variable type> <name> <value> <signature> <accessibility>

- a Tcl variable of type <variable type> is registered under the name <name>
  - read-only
  - protected from being changed while script execution.
    Otherwise script execution break off.
- <signature>, <accessibility> and <value> are stored for later use.
- <value> describes an offset
  → Application for members of abstract data structures or arrays

- Delete registered variable: DelVar <variable name>

Registration of TCL-Commands (Data)

RegCmd DATA <command name> <target address> <signature> <accessibility>

- command of type “DATA” is registered under the name <command name>.
- target address, signature and accessibility string are stored for later use.
- If command already existing:
  - error message
  - registration is ignored.

- Delete registered command: DelObj <name>

Registration of TCL-Commands (Procedures)

RegCmd PROC<command name> <target address> <signature>

- command of type “PROC” is registered under the name <command name>.
- target address and signature string are stored for later use.
- If command already existing:
  - error message
  - registration is ignored.

- Delete registered procedure: DelObj <name>

Overview: BSH-specific TCL-commands

- TEngGetVersion, TEngGetVendor, TEngGetCPUType, TEngGetDataModel, TEngGetBaseType
  - Show information about connected target
- GetMultipleCount, GetPROCCount, GetDATACount, GetVarCount
  - Show number of registered variables / functions
- GetCmd <name>
  - Returns the signature of the function or variable
- SizeOf <name>
  - Returns the size of the registered variable
- TEngDbg
  - Perform tracing
- TEngConnect ⇔ TEngDisconnect
  - establish or cancel connection to the target
- TEngSendDbusMsg
  - Send specific DBUS-message

Handling the Test Engine manually

- Example:

Command	Description
% TEngConnect /REMOTE=UDA1,12,1,10,36864, VDBUS_FLAG_9600_BAUD, VDBUS_FLAG_NO_COLLISION	Establish Connection
% RegCmd PROC TECCT_square_me 08002441 I1_I1 % RegCmd DATA my_global 200003F4 I1 RW	Register Variables / Procedures
% set file [open “report.txt" w]	„report.txt“ is opened as variable „file“
% puts $file "[TECCT_square_me 2] [TECCT_square_me 5]”	Return values of „TECCT_square_me” (with parameters “2” and “5”) are written into the opened file
% close $file	Close file
% TEngDisconnect	Release connection

Example: Read Data from Target

- Analyse:
  Aufrufe und Funktionsweise nachvollziehen

Using Test-Scripts

- Create test script file „testscript.syr“ e.g. →

```
source TEST_TESTER_MTEST.syr

RegCmd    PROC   TECCT_square_me 08002441  I1_I1
TECCT_square_me 2

set file [open "test.txt" w]
puts $file "Report"

RegCmd DATA test 20000049 B1 RW
RegCmd DATA test2 20000049 B1 RW

puts $file "[test] [test2]"
close $file
```

- Include / Run in Test Engine by calling

```
% source testscript.syr
```

- Alternatively (in windows command line)

```
TengAppl.exe path\testscript.syr
```

Creation of Test-Reports

- **Requirements:**
  - Information about project data, environment and target connection
  - Description of module functions, variables and struct elements
  - Test case description
  - Test execution and result analysis (actual value ⇔ reference value)
  - Result summary (PASSED / FAILED)

```
    CONNECTION INFORMATION
Type of serial link to target: DBUS2
Selected UDA port serial link: UDA1
Host address (normally PC 12): 12
Destination node addr. target: 1
Target subsystem - DBUS comm.: 10
Target base ID for DBUS comm.: 0x9000
Additional select. DBUS flags: VDBUS_FLAG_9600_BAUD,V

    SYMBOLS
Path of sourced symbol file : W:/Dimensions/SW_LIB_TEST_
Sourced symbol file for Teng: TEST_TESTER_MTEST.syr
```

TECCT_square_me	T01	T02	T03	T04
INPUT-VARIABLES	-	-	-	-
PARAMETER	-	-	-	-
param	0	1	3	4
OUTPUT-VARIABLES	-	-	-	-
FUNKTION-RETURN	-	-	-	-
RETURNVALUE	0	1	9	16
EXPECTED	0	1	9	16
PASSED/FAILED	P	P	P	P

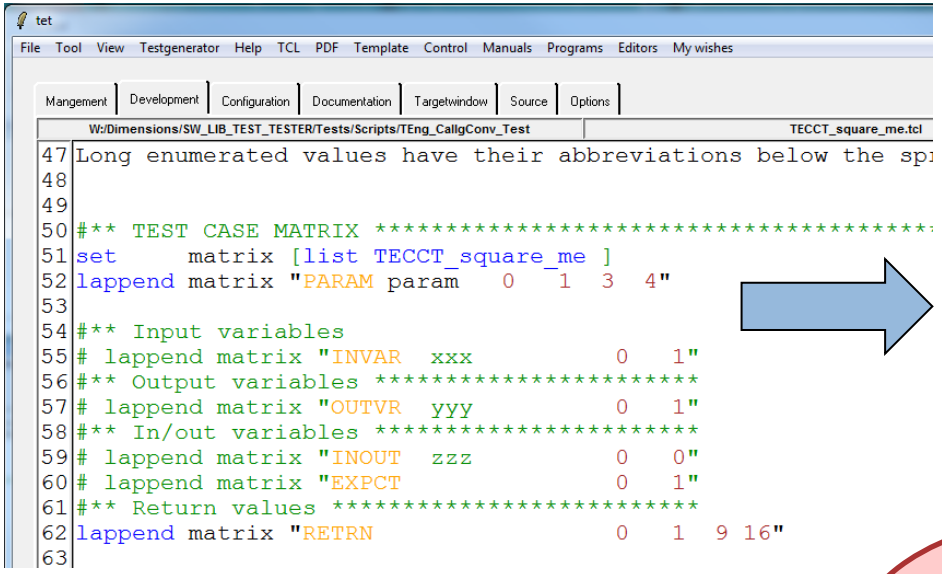
Table 3: Results of test 0

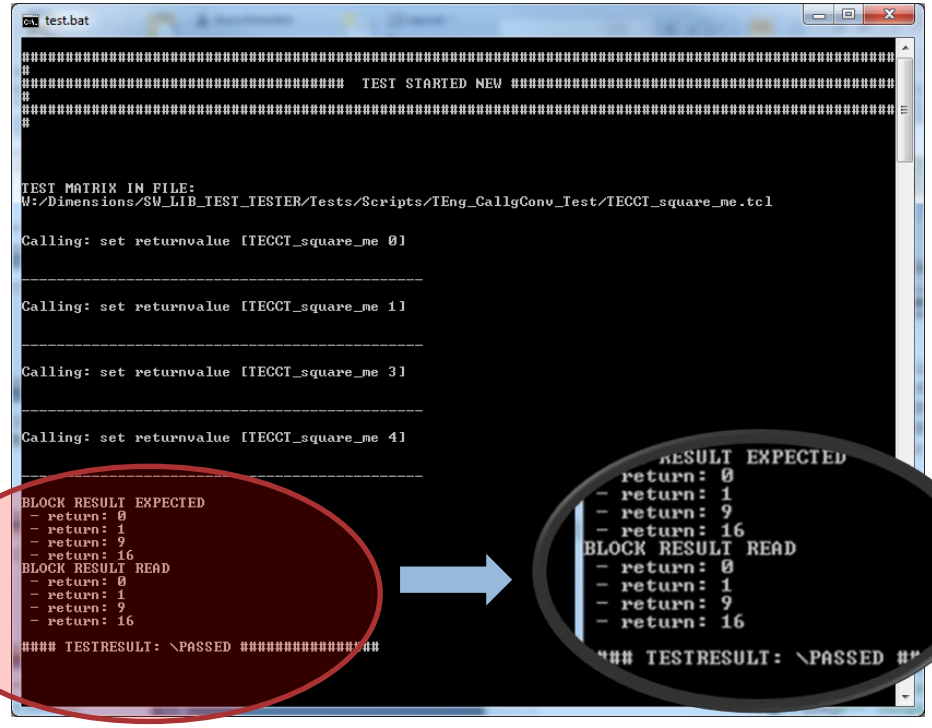
Overall result of test cases

☞ Testresult: PASSED

Automated Test (with TET)

- Definition of the Test Case Matrix in TET





→Automatic execution of all tests

→Automatic generation of a test report (also writeable to a file)

Problems with the current Test Engine based on TCL

- Based on TCL Version 8.4.12 (released 2005)
- Extent and complexity have increased step by step
  ➔ Refactoring necessary

- Disadvantages of TCL itself:
  – Slow in comparison to C
  – Not object-orientated
  – Small command set
  – Line-orientated syntax
  – Limitedly C-type compatible

Potential for improvement

- Extending the functional range of the TEng
  - → support of additional data types
  - → integration of a function to directly execute a test case with the TEng (without TET)
  - → integration of test report functionality in TEng
- Usage of an object-orientated language
- Usage of a nowadays scripting language
- Usage of an interpreting scripting language (Hardware independency)

➔ Python

List of Literature

[BSH07a] Peter Cambeis (BSH-intern). The Test Engine - Book One. BSH-Intranet, 2007.

[BSH07b] Peter Cambeis (BSH-intern). The Test Engine - Book Two. BSH-Intranet, 2007.

[BSH14] Jens Lehmann (BSH-intern). Test Levels: Workshop Automated Testing. BSHIntranet, 2014.

[Grü13] Stephan Grünfelder. Software-Test für Embedded Systems. Ein Praxishandbuch für Entwickler, Tester und technische Projektleiter. dpunkt.verlag, 2013. isbn: 978-3-86490-048-8.

[Hol12] Rune Holen. BSH General Bus Specification D-Bus-2. BSH Hausgeräte GmbH. BSH-Intranet, 2012.

List of Literature

[Köh07] Achim Köhler. Der C/C++-Projektbegleiter. C/C++ Projekte planen, dokumentieren, bauen und testen. dpunkt.verlag, 2007. isbn: 978-3-89864-470-9.

[Liu13] Huimei Liu. “Entwicklung einer Vorgehensweise zur Testautomatisierung im Rahmen des PDM am Beispiel der HELLA KGaA Hueck & Co.” MA thesis. Universität Stuttgart: Fakultät Informatik, Elektrotechnik und Informationstechnik, May 2013.

[Neu15] Gustaf Neumann. Objekt-orientiertes Design und Systementwicklung: TCL. WU Wirtschaftsuniversität Wien. May 10, 2015. url: http://wi.wu-wien.ac.at/studium/Abschnitt_2/LVA_ws99/neumann/oo/einheit3/03-9.html

List of Literature

- https://www.dpunkt.de/buecher/4003/software-test-f%C3%BCr-embedded-systems.html
  https://www.dpunkt.de/leseproben/4003/4_Unit%20Tests.pdf
- https://www.dpunkt.de/buecher/2584.html
  https://www.dpunkt.de/leseproben/2584/Leseprobe.pdf
- http://download.e-bookshelf.de/download/0000/6666/97/L-X-0000666697-0001402267.XHTML/index.xhtml
  https://www.dpunkt.de/leseproben/3427/4%20Motivation%20und%20Einleitung%20%28Kap%201%29.pdf
- http://wi.wu-wien.ac.at/studium/Abschnitt_2/LVA_ws99/neumann/oo/einheit3/03-9.html
- http://www.pst.ifi.lmu.de/Lehre/wise-12-13/sep/06-unit-testing.pdf