Software Product: AML-include TempK_90

Preface

This software validation method, described in the document "Nordtest Method of Software Validation", is basically developed to assist accredited laboratories in validation of software for calibration and testing. The actual report is provided via a Word 2000 template "Nordtest Software Validation Report.dot" which is organized in accordance with the life cycle model used in the validation method. There are two main tasks associated with each life cycle phase:

- *Preliminary work*. To specify/summarize the requirements (forward/reverse engineering for prospective/retrospective validation), to manage the design and development process, make the validation test plan, document precautions (if any), prepare the installation procedure, and to plan the service and maintenance phase.
- *Peer review and test*. To review all documents and papers concerning the validation process and conduct and approve the planned tests and installation procedures.

The report template contains 5 sections:

1. *Objectives and scope of application.* Tables to describe the software product, to list the involved persons, and to specify the type of software in order to determine the extent of the validation.
2. *Software life cycle overview.* Tables to specify date and signature for the tasks of preliminary work and the peer reviews assigned to each life cycle phase as described above.
3. *Software life cycle activities.* Tables to specify information that is relevant for the validation. It is the intention that having all topics outlined, it should be easier to write the report.
4. *Conclusion.* Table for the persons responsible to conclude and sign the validation report.
5. *References and annexes.* Table of references and annexes.

Even if possible, it is recommended not to delete irrelevant topics but instead mark them as excluded from the validation by a "not relevant" or "not applicable" (n/a) note – preferably with an argument – so it is evident that they are not forgotten but are deliberately skipped.

It is the intention that the validation report shall be a "dynamic" document, which is used to keep track on all changes and all additional information that currently may become relevant for the software product and its validation. Such current updating can, however, make the document more difficult to read, but never mind – it is the *contents*, not the *format*, which is important.

Table of contents

1 Objectives and scope of application

This section describes the software product in general terms. It includes objectives and scope of application and, if relevant, overall requirements to be met (such as standards and regulations).

All persons who are involved in the validation process and are authorized to sign parts of this report should be listed in the Role / Responsibility table. The report could hereafter be signed electronically with date and initials of those persons at suitable stages of the validation process.

The type of the software is outlined in order to determine the extent of validation and testing.

1.1 Objectives and scope of application	
General description	AML-include TempK_90
Scope of application	Include-module for AML/ACCS for calculation between temperature and voltage and reverse for type K (ITS 90).
Product information	F:\ics\accs97\include\temp\tempK_90.inc
Overall requirements	AML-compiler (AREPA Measurement Language). ACCS-Calibration-program (AREPA Calibration Control System).

1.2 Role / Responsibility	*Title and Name*	*Initials*
System owner	Head of Electrical Laboratory K.K.Hansen	KKH
System administrator	Head of IT department P.Madsen	PM
Application administrator	Program Developer J.M.Thomsen	JMT
System user	All Technicians in Electrical Laboratory	
Quality responsible	Calibration Responsibel. O.Kristensen	OK
Requirements team...	Technicians and developer. S.Gadegaard J.M.Thomsen	 SG JMT
Development team...	Program Developer. J.M.Thomsen	JMT
Peer review team...	Technician S.Gadegaard Program Developer J.M.Thomsen Head of IT Department P.Madsen	SG JMT PM
Testing team...	Technicians S.Gadegaard C.Madsen	 SG CM

1.3 Type of software	
Purchased Software:	Self-developed software:
☐ Configurable software package	☐ Compiled executable program (e.g. C/C++)
☐ Commercial off-the-shelf software	☑ Spreadsheet (macro code, Add-In, etc.)
☐ Tool to assist in the software development	☐ Simple spreadsheet (no macro code)
☐ Subcontracted software development	☐ Tool to assist in development or testing
☐ Source code available and known	☐ Includes purchased software components
☐ Only partial validation	☐ Subcontracted software validation
Comments:	Comments: General Include Module.

2 Software life cycle overview

This section outlines the activities related to the phases in the life cycle model used in the validation process. The numbers refer to the corresponding subsections in section 3. Each activity contains a field for the preliminary task to be performed, a field for the validation method, and fields to specify the date and signature when the work is done.

Activity	**2.1 Requirements and system acceptance test specification**	*Date / Initials*
Task	3.1.1 Requirements specification	18-09-2002 *SG*
Method	3.1.1 Peer review	19-09-2002 *JMT*
Check	3.1.1 Requirements specification approved	19-09-2002 *SG*
Task	3.1.2 System acceptance test specification	19-09-2002 *SG*
Method	3.1.2 Peer review	20-09-2002 *JMT*
Check	3.1.2 System acceptance test specification approved	20-09-2002 *SG*

Activity	**2.2 Design and implementation process**	*Date / Initials*
Task	3.2.1 Design and development planning	*N/A*
Method	3.2.1 Peer review	*N/A*
Task	3.2.2 Design input	*N/A*
Method	3.2.2 Peer review	*N/A*
Task	3.2.3 Design output	23-09-2002 *JMT*
Method	3.2.3 Peer review	23-09-2002 *PM*
Task	3.2.4 Design verification	23-09-2002 *.MT*
Method	3.2.4 Peer review	23-09-2002 *PM*
Task	3.2.5 Design changes 1. Description: 2. Description: 3. ...	N/A

Activity	**2.2 Design and implementation process**	*Date / Initials*
Method	3.2.5 Peer review 1. Action: 2. Action: 3. ...	N/A

Activity	**2.3 Inspection and testing**	*Date / Initials*
Task	3.3.1 Inspection plan	23-09-2002 *SG*
Method	3.3.1 Inspection	23-09-2002 *JMT*
Check	3.3.1 Inspection approved	24-09-2002 *SG*
Task	3.3.2 Test plan	23-09-2002 *SG*
Method	3.3.2 Test performance	23-09-2002 *JMT*
Check	3.3.2 Test approved	24-09-2002 *SG*

Activity	**2.4 Precautions**	*Date / Initials*
Task	3.4.1 Registered anomalies	N/A
Method	3.4.1 Peer review	N/A
Task	3.4.2 Precautionary steps taken	N/A
Method	3.4.2 Verification of measures	N/A

Activity	**2.5 Installation and system acceptance test**	*Date / Initials*
Task	3.5.1 Installation summary	24-09-2002 *SG*
Method	3.5.1 Peer review	24-09-2002 *JMT*
Task	3.5.2 Installation procedure	25-09-2002 *SG*
Method	3.5.2 Verification and test of installation	25-09-2002 *SG*
Task	3.5.3 System acceptance test preparation	25-09-2002 *SG*
Method	3.5.3 System acceptance test	25-09-2002 *JMT*
Check	3.5.3 System acceptance test approved	25-09-2002 *SG*

Activity	**2.6 Performance, servicing, maintenance, and phase out**	*Date / Initials*
Task	3.6.1 Performance and maintenance	N/A
Method	3.6.1 Peer review	N/A
Task	3.6.2 New versions 1. Version: 2. Version: 3. ...	N/A

Activity	**2.6 Performance, servicing, maintenance, and phase out**	*Date / Initials*
Method	3.6.2 Peer review 1. Action: 2. Action: 3. ...	N/A
Task	3.6.3 Phase out	N/A
Method	3.6.3 Peer review	N/A

3 Software life cycle activities

This section contains tables for documentation of the software validation activities. Each subsection is numbered in accordance with the overview scheme above. The tables are filled in with information about the tasks to be performed, methods to be used, criteria for acceptance, input and output required for each task, required documentation, the persons that are responsible for the validation, and any other information relevant for the validation process. Topics excluded from being validated are explicitly marked as such.

3.1 Requirements and system acceptance test specification

The requirements describe and specify the software product completely and are basis for the development and validation process. A set of requirements can always be specified. In case of retrospective validation (where the development phase is irrelevant) it can at least be specified what the software is purported to do based on actual and historical facts. The requirements should encompass everything concerning the use of the software.

Topics	**3.1.1 Requirements specification**
Objectives *Description of the software product to the extent needed for design, implementation, testing, and validation.*	2 sub-routines: **ConvFromTempK_90(temp)** Calculation of the voltage (µV) from temperature (°C). **ConvToTempK_90(input, result)** Calculation of the temperature (°C) from voltage (µV) with the accuracy of the last digit of result.
Version of requirements *Version of, and changes applied to, the requirements specification.*	Versions are controlled with VCSLITE (Freeware), where the dates of changes are stored with all the old versions. If any changes is made the Software Developer has to put the new version into VCSLITE with a comment of the changes.
Input *All inputs the software product will receive. Includes ranges, limits, defaults, response to illegal inputs, etc.*	Inputs are described in Objectives.

Topics	**3.1.1 Requirements specification**
Output *All outputs the software product will produce. Includes data formats, screen presentations, data storage media, printouts, automated generation of documents, etc.*	Outputs are described in Objectives.
Functionality *All functions the software product will provide. Includes performance requirements, such as data throughput, reliability, timing, user interface features, etc.*	Functions are described in Objectives.
Traceability *Measures taken to ensure that critical user events are recorded and traceable (when, where, whom, why).*	When a critical user event is found, the user has to inform a Software Developer that takes actions to make a registration of the event, and find out what to do about it.
Hardware control *All device interfaces and equipments to be supported.*	N/A
Limitations *All acceptable and stated limitations in the software product.*	Limits are described in Objectives.
Safety *All precautions taken to prevent overflow and malfunction due to incorrect input or use.*	No scaling is made if the prefixes are not as described in Objectives. The accuracy of the output is up to 12 digits.
Default settings *All settings applied after power-up such as default input values, default instrument or program control settings, and options selected by default. Includes information on how to manage and maintain the default settings.*	N/A

Topics	**3.1.1 Requirements specification**
Version control *How to identify different versions of the software product and to distinguish output from the individual versions.*	Versions are controlled with VCSLITE (Freeware), where the dates of changes are stored with all the old versions. If any changes is made the Software Developer has to put the new version into VCSLITE with a comment of the changes. When a certificate is made we can trace, which version of the program was used and which version of the include-module was active at the calibration time.
Dedicated platform *The hardware and software operating environment in which to use the software product. E.g. laboratory or office computer, the actual operating system, network, third-party executables such as Microsoft® Excel and Word, the actual version of the platform, etc.*	The include-module is compiled with AML, and the compiled program is used with ACCS. The software is running on Windows 98 – PC's in the laboratory, Windows Terminal Server (Windows 2000), and in some offices.
Installation *Installation requirements, e.g. installation kit, support, media, uninstall options, etc.*	The AML-compiler is using the include-module, placed on the network.
How to upgrade *How to upgrade to new versions of e.g. service packs, Microsoft® Excel and Word, etc...*	If a new version of the AML-compiler and/or the ACCS-calibration-program is used the Acceptance Test must be used again.
Special requirements *Requirements the laboratory is committed to, security, confidentiality, change control and back-up of records, protection of code and data, precautions, risks in case of errors in the software product, etc.*	Backup of the include-module is done according to AREPA's Quality Manual.
Documentation *Description of the modes of operation and other relevant information about the soft-ware product.*	No documentation, except this document.
User manual *User instructions on how to use the software product.*	No manual, except this document.

Topics	**3.1.1 Requirements specification**
On-line help *On-line Help provided by Windows programs.*	No On-Line Help for the include-module.
Validation report *Additional documentation stating that the software product has been validated to the extent required for its application.*	No Validation Report, except this document.
Service and maintenance *Documentation of service and support concerning maintenance, future updates, problem solutions, requested modifications, etc.*	Any service and maintenance documentation will be reported in this document.
Special agreements *Agreements between the supplier and the end-user concerning the software product where such agreements may influence the software product development and use. E.g. special editions, special analysis, extended validation, etc.*	N/A
Phase out *Documentation on how (and when) to discontinue the use of the software product, how to avoid impact on existing systems and data, and how to recover data.*	The include-module has to be saved, as long time the laboratory has to recover the data, where the include-module is used.
Errors and alarms *How to handle errors and alarms.*	When a critical user event is found, the user has to inform a Software Developer that takes actions to make a registration of the event, and find out what to do about it.

The system acceptance test specification contains objective criteria on how the software product should be tested to ensure that the requirements are fulfilled and that the software product performs as required in the environment in which it will be used. The system acceptance test is performed after the software product has been properly installed and thus is ready for the final acceptance test and approval for use.

Topics	**3.1.2 System acceptance test specification**
Objectives *Description of the operating environment(s) in which the software product will be tested and used.*	The include-module is compiled with AML, and the compiled program is used with ACCS (Version 2.0.17). The software is running on Windows 98 – PC's in the laboratory, Windows Terminal Server (Windows 2000), and in some offices.
Scope *Scope of the acceptance test. E.g. installation and version, startup and shutdown, common, selected, and critical requirements, and areas not tested.*	The Testprogram "F:\ics\accs97\worksrc\SW_Validering\TempK_90.aml" is copied to "jmt.aml" and compiled. The testprogram with AREPA-number 7022 is started in ACCS to run the test.
Input *Selected inputs the software product must receive and handle as specified.*	**ConvFromTempK_90(temp)** Input °C: Table µV: -250 -6404 -110 -3852 0 0 50 2023 310 12624 520 21497 840 34908 1000 41276 1160 47367 1370 54819 The error can be up to 0.5 µV, the table is only with a resolution of 1 µV. **ConvToTempK_90(input, result)** Input µV: Tabel °C: -6404 -250 -2243 -60 0 0 2851 70 6540 160 20218 490 31628 760 40494 980 48838 1200 54819 1370 The error can be up to 0.1 ° at –250 °C, at others less than 0.05 °C, the table is only with a resolution of 1 µV.
Output *Selected outputs the software product must produce as specified.*	The expected Output is described in the Input Section.

Topics	**3.1.2 System acceptance test specification**
Functionality *Selected functions the software product must perform as specified.*	The Functions is described in the Input Section.
Personnel *Description of operations the actual user(s) shall perform in order to make evident that the software product can be operated correctly as specified and documented.*	Run the program as described in Objectives.
Errors and alarms *How to handle errors and alarms.*	When a critical user event is found, the user has to inform a Software Developer that takes actions to make a registration of the event, and find out what to do about it.

3.2 Design and implementation process

The design and implementation process is relevant when developing new software and when handling changes subjected to existing software. The output from this life cycle phase is a program approved and accepted for the subsequent inspection and testing phase. Anomalies found and circumvented in the design and implementation process should be described in section 3.4, Precautions.

Topics	**3.2.1 Design and development planning**
Objectives *Expected design outcome, time schedule, milestones, special considerations, etc.*	N/A – The include-module was made in December 1998, and has been used since.
Design plan *Description of the software product e.g. in form of flow-charts, diagrams, notes, etc.*	N/A – The include-module was made in December 1998, and has been used since.
Development plan *Development tools, manpower, and methods.*	N/A – The include-module was made in December 1998, and has been used since.
Review and acceptance *How to review, test, and approve the design plan.*	N/A – The include-module was made in December 1998, and has been used since.

The design input phase establishes that the requirements can be implemented. Incomplete, ambiguous, or conflicting requirements are resolved with those responsible for imposing these requirements. The input design may be presented as a detailed specification, e.g. by means of flow charts, diagrams, module definitions etc.

Topics	**3.2.2 Design input**
Requirements analysis *Examinations done to ensure that the requirements can be implemented.*	N/A – The include-module was made in December 1998, and has been used since.
Software modules *Description of the software modules to be implemented.*	N/A – The include-module was made in December 1998, and has been used since.
Review and acceptance *How to review, test, and approve the Design Input section.*	N/A – The include-module was made in December 1998, and has been used since.

The design output must meet the design input requirements, contain or make references to acceptance criteria, and identify those characteristics of the design that are crucial to the safe and proper functioning of the product. The design output should be validated prior to releasing the software product for final inspection and testing.

Topics	**3.2.3 Design output**	
Implementation (coding and compilation) *Development tools used to implement the software, notes on anomalies, plan for module and integration test, etc.*	N/A – The include-module was made in December 1998, and has been used since.	
Version identification *How to identify versions on screen, printouts, etc. Example "Version 1.0.0".*	Versions are controlled with VCSLITE (Freeware), where the dates of changes are stored with all the old versions. If any changes is made the Software Developer has to put the new version into VCSLITE with a comment of the changes.	
Good programming practice *Efforts made to meet the recommendations for good programming practice...*	Source code is... ☑ Modulized ☐ Encapsulated ☑ Functionally divided ☐ Strictly compiled ☐ Fail-safe (handling errors)	Source code contains... ☐ Revision notes ☐ Comments ☐ Meaningfull names ☐ Readable source code ☑ Printable source code
Windows programming *If implementing Windows applications...*	☐ Interface implemented using standard Windows elements ☐ Interface implemented using self-developed Windows elements ☐ Application manages single/multiple running instances Comments: N/A - Only a sub-routine.	

Topics	**3.2.3 Design output**
Dynamic testing *Step-by-step testing made dynamically during the implementation...*	☑ All statements have been executed at least once ☑ All functions have been executed at least once ☑ All case segments have been executed at least once ☐ All loops have been executed to their boundaries ☐ Some parts were not subject to dynamic test Comments:
Utilities for validation and testing *Utilities implemented to assist in validation and testing and specification of the test environment.*	N/A
Inactive code *Inactive (dead) code left for special purposes.*	N/A
Documentation *Documentation provided as output from the Design Output section.*	N/A
Review and acceptance *How to review, test, and approve the Design Output section.*	N/A

At appropriate stages of design, formal documented reviews and/or verifications of the design should take place before proceeding with the next step of the development process. The main purpose of such actions is to ensure that the design process proceeds as planned.

Topics	**3.2.4 Design verification**
Review *Review current development stage according to the design and development plan.*	N/A
Change of plans *Steps taken to adjust the development process.*	N/A

The Design Change section serves as an entry for all changes applied to the software product, also software products being subjected to retrospective validation. Minor corrections, updates, and enhancements that do not impact other modules of the program are regarded as changes that do not re-

quire an entire revalidation. Major changes are reviewed in order to decide the degree of necessary revalidation or updating of the requirements and system acceptance test specification.

Topics	**3.2.5 Design changes**	*Date / Initials*
Justification *Documentation and justification of the change.*	1. Description: 2. Description: 3. ...	N/A
Evaluation *Evaluation of the consequences of the change.*	1. Description: 2. Description: 3. ...	N/A
Review and approving *Review and approving the change.*	1. Description: 2. Description: 3. ...	N/A
Implementing *Implementing and verifying the change.*	1. Action: 2. Action: 3. ...	N/A
Validation *The degree of revalidation or updating of requirements.*	1. Action: 2. Action: 3. ...	N/A

3.3 Inspection and testing

The inspection and testing of the software product is planned and documented in a test plan. The extent of the testing is in compliance with the requirements, the system acceptance test specification, the approach, complexity, risks, and the intended and expected use of the software product.

Topics	**3.3.1 Inspection plan and performance**	*Date / Initials*
Design output *Results from the Design Output section inspected...*	☑ Program coding structure and source code ☑ Evidence of good programming practice ☐ Design verification and documented reviews ☐ Change-control reviews and reports Comments:	27-09-2002 *SG*
Documentation *Documentation inspected...*	☐ Program documentation, flow charts, etc. ☑ Test results ☐ User manuals, On-line help, Notes, etc. ☐ Contents of user manuals approved Comments:	27-09-2002 *SG*

Topics	**3.3.1 Inspection plan and performance**	*Date / Initials*
Software development environment *Environment elements inspected...*	☑ Data integrety ☑ File storage ☑ Access rights ☐ Code protection ☐ Installation kit, replication and distribution Comments:	27-09-2002 *SG*
Result of inspection *Approval of inspection.*	☑ Inspection approved Comments:	27-09-2002 *SG*

The test plan is created during the development or reverse engineering phase and identify all elements that are about to be tested. The test plan should explicitly describe what to test, what to expect, and how to do the testing. Subsequently it should be confirmed what was done, what was the result, and if the result was approved.

Topics	**3.3.2 Test plan and performance**	*Date / Initials*
Test objectives *Description of the test in terms of what, why, and how.*	Tested as described in 3.1.2.	27-09-2002 *SG*
Relevancy of tests *Relative to objectives and required operational use.*	Samples of the possible inputs are tested for the 3 sub-routines.	27-09-2002 *SG*
Scope of tests *In terms of coverage, volumes, and system complexity.*	N/A	27-09-2002 *SG*
Levels of tests *Module test, integration test, and system acceptance test.*	Module Test – Samples of the possible inputs are tested for the 2 sub-routines.	27-09-2002 *SG*
Types of tests *E.g. input, functionality, boundaries, performance, and usability.*	Tested as described in 3.1.2.	27-09-2002 *SG*
Sequence of tests *Test cases, test procedures, test data and expected results.*	Tested as described in 3.1.2.	27-09-2002 *SG*
Configuration tests *Platform, network, and integration with other systems.*	Tested on Windows 98 PC's, and Windows Terminal Server (Windows 2000).	27-09-2002 *SG*

Topics	**3.3.2 Test plan and performance**	*Date / Initials*
Calculation tests *To confirm that known inputs lead to specified outputs.*	Tested as described in 3.1.2.	27-09-2002 *SG*
Regression tests *To ensure that changes do not cause new errors.*	N/A	27-09-2002 *SG*
Traceability tests *To ensure that critical events during use are recorded and traceable as required.*	When a critical event is found, the test-user has to inform a Software Developer that takes actions to make a registration of the event, and find out what to do about it.	27-09-2002 *SG*
Special concerns *Testability, analysis, stress, reproducibility, and safety.*	N/A	27-09-2002 *SG*
Acceptance criteria *When the testing is completed and accepted.*	The test gives the expected results. See the Screen print in Section 5.	27-09-2002 *SG*
Action if errors *What to do if errors are observed.*	When a critical user event is found, the user has to inform a Software Developer that takes actions to make a registration of the event, and find out what to do about it. Make some changes in the sub-routine that makes it work right.	27-09-2002 *SG*
Follow-up of tests *How to follow-up the testing.*		27-09-2002 *SG*
Result of testing *Approval of performed tests.*	☑ Testing approved Comments: The test gives the expected results. See the Screen print in Section 5.	27-09-2002 *SG*

3.4 Precautions

When operating in a third-party software environment, such as Microsoft® Windows and Office, some undesirable, inappropriate, or anomalous operating conditions may exist. A discrepancy between the description of the way an instrument should operate, and the way it actually does, may be regarded as an anomaly as well. Minor errors in a software product may sometimes be acceptable if they are documented and/or properly circumvented.

Topics	**3.4.1 Registered anomalies**
Operative system *Anomalous operating conditions in e.g. Windows.*	N/A

Topics	**3.4.1 Registered anomalies**
Spreadsheet *Anomalous operating conditions in e.g. Excel.*	N/A
Instruments *Anomalous operating conditions in the used instruments.*	N/A
General precautions *Anomalous operating conditions associated with the software product itself.*	N/A

The steps taken to workaround anomalous, inappropriate, or undesired operating conditions are verified and tested.

Topics	**3.4.2 Precautionary steps taken**	*Date / Initials*
Operative system *Precautionary steps taken in e.g. Windows settings.*	N/A	N/A
Spreadsheet *Precautionary steps taken to workaround problems using e.g. Excel.*	N/A	N/A
Instruments *Precautionary steps taken to workaround problems with the used instruments.*	N/A	N/A
General precautions *Precautionary steps taken to workaround problems with the software product itself.*	N/A	N/A

3.5 Installation and system acceptance test

The validation of the installation process ensures that all software elements are properly installed on the host computer and that the user obtains a safe copy of the software product.

Topics	**3.5.1 Installation summary**
Installation method *Automatic or manual installation...*	☐ Automatic - installation kit located on the installation media ☐ Manual - Copy & Paste from the installation media Comments: The include-module is compiled with the calibration-programs.
Installation media *Media containing the installation files...*	☐ Diskette(s) ☐ CD-ROM ☑ Source disk folder (PC or network) ☐ Download from the Internet Comments:
Input files *List of (relevant) files on the installation media.*	N/A
Installed files *List of (relevant) installed files, e.g. EXE- and DLL-files, spreadsheet Add-ins and Templates, On-line Help, etc.*	N/A
Supplementary files *Readme files, License agreements, examples, etc.*	N/A

The program is tested after installation to the extent depending on the use of the product and the actual requirements, e.g. an adequate test following the validation test plan. Sometimes it is recommendable to carry out the installation testing in a copy of the true environment in order to protect original data from possible fatal errors due to using a new program.

Topics	**3.5.2 Installation procedure**	*Date / Initials*
Authorization *Approval of installation in actual environment.*	Person responsible: SG	27-09-2002 *SG*
Installation test *The following installations have been performed and approved...*	☐ Tested and approved in a test environment ☐ Tested and approved in actual environment ☑ Completely tested according to test plan ☐ Partly tested (known extent of update) Comments:	27-09-2002 *SG*

The system acceptance test is carried out in accordance with the system acceptance test specifications after installation. The software product may subsequently be approved for use.

Topics	**3.5.3 System acceptance test**	*Date / Initials*
Test environment *The environment in which the system acceptance test has been performed...*	☑ The actual operating environment (site test) ☐ A true copy of the actual environment ☐ External environment (supplier factory test) Comments:	27-09-2002 *SG*
Test performance *Areas, which have been tested and approved...*	☐ Installation and version ☐ Startup and shutdown ☐ Selected or critical requirements ☑ Selected inputs ☑ Selected outputs ☑ Selected functionality ☐ Performance vs. user instructions Comments:	27-09-2002 *SG*
User level test *Test if users of various skills can use the software product...*	☑ Tested on beginner user level ☐ Tested on experienced user level ☐ Tested on professional user level Comments:	27-09-2002 *SG*
Result of testing *Approval for use.*	☑ Testing approved Comments:	27-09-2002 *SG*

3.6 Performance, servicing, maintenance, and phase out

In this phase the software product is in use and subject to the requirements for service, maintenance, performance, and support. This phase is where all activities during performance reside and where decisions about changes, upgrades, revalidation, and phase out are made.

Topics	**3.6.1 Performance and maintenance**	*Date / Initials*
Problem / solution *Detection of software problems causing operating troubles. A first step could be to suggest or set up a well-documented temporary solution or workaround.*	1. Problem / solution: 2. Problem / solution: 3. ...	N/A

Topics	**3.6.1 Performance and maintenance**	*Date / Initials*
Functional maintenance *E.g. if the software product is based on international standards, and these standards are changed, the software product, or the way it is used, should be updated accordingly.*	1. Function / action: 2. Function / action: 3. ...	N/A
Functional expansion and performance improvement *List of suggestions and requests, which can improve the performance of the software product.*		N/A

When a new version of the software product is taken into use, the effect on the existing system is carefully analyzed and the degree of revalidation decided. Special attention is paid to the effect on old spreadsheets when upgrading the spreadsheet package.

Topics	**3.6.2 New versions**	*Date / Initials*
Description *Description of the new version to the extent needed to decide whether or not to upgrade.*	1. Version: 2. Version: 3. ...	N/A
Action *Action to be taken if upgrade is decided. See also the Design Changes section.*	1. Action: 2. Action: 3. ...	N/A

It is taken into consideration how (and when) to discontinue the use of the software product. The potential impact on existing systems and data are examined prior to withdrawal.

Topics	**3.6.3 Phase out**	*Date / Initials*
How and when *To discontinue the use of the software product.*	When new software is bought or developed to replace the sub-routine, or the full AML and ACCS package.	N/A
Consequences *Assumed impact on existing systems and data and how to avoid or reduce the harm.*	The include-module has to be saved, as long time the laboratory has to recover the data, where the include-module is used.	N/A

4 Conclusion

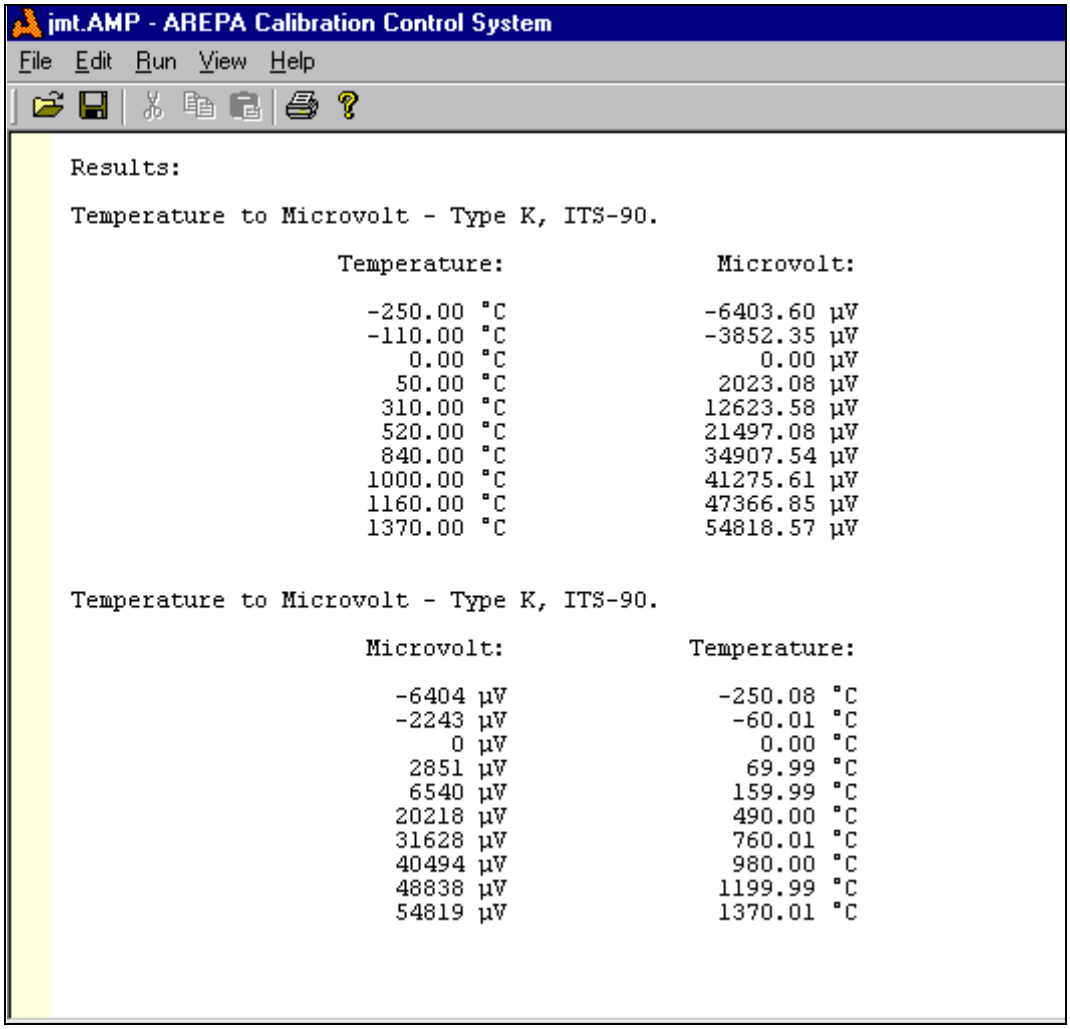
By the subsequent signatures it becomes evident that all validation activities are documented and approved.

Final approval for use	
Laboratory Identification:	Electrical Laboratory, Arepa Silkeborg
Responsible for validation:	S.Gadegaard
Remarks:	
Date: 30-09-2002	Signature: *S.Gadegaard*

Conclusion	
☑ All check boxes are locked for editing (to avoid inadvertent change of settings)	
Comments: The module is working as planned.	
Date: 30-09-2002	Signature: *J.M.Thomsen*

5 References and annexes

All external documents (if any) must be dated and signed.



Screen print of test results