TECHNICAL REPORT

IEC TR 61926-1-1

First edition
1999-10

Design automation –

**Part 1-1:
Harmonization of ATLAS test languages**

Automatisation de la conception –

*Partie 1-1:
Harmonisation de langages d'essais ATLAS*

Reference number
IEC/TR 61926-1-1:1999(E)

Numbering

As from 1 January 1997 all IEC publications are issued with a designation in the 60000 series.

Consolidated publications

Consolidated versions of some IEC publications including amendments are available. For example, edition numbers 1.0, 1.1 and 1.2 refer, respectively, to the base publication, the base publication incorporating amendment 1 and the base publication incorporating amendments 1 and 2.

Validity of this publication

The technical content of IEC publications is kept under constant review by the IEC, thus ensuring that the content reflects current technology.

Information relating to the date of the reconfirmation of the publication is available in the IEC catalogue.

Information on the subjects under consideration and work in progress undertaken by the technical committee which has prepared this publication, as well as the list of publications issued, is to be found at the following IEC sources:

- **IEC web site***
- **Catalogue of IEC publications**
  Published yearly with regular updates
  (On-line catalogue)*
- **IEC Bulletin**
  Available both at the IEC web site* and as a printed periodical

Terminology, graphical and letter symbols

For general terminology, readers are referred to IEC 60050: *International Electrotechnical Vocabulary* (IEV).

For graphical symbols, and letter symbols and signs approved by the IEC for general use, readers are referred to publications IEC 60027: *Letter symbols to be used in electrical technology*, IEC 60417: *Graphical symbols for use on equipment. Index, survey and compilation of the single sheets* and IEC 60617: *Graphical symbols for diagrams.*

* See web site address on title page.

TECHNICAL REPORT

IEC TR 61926-1-1

First edition
1999-10

Design automation –

Part 1-1: Harmonization of ATLAS test languages

Automatisation de la conception –

Partie 1-1: Harmonisation de langages d'essais ATLAS

International Electrotechnical Commission 3, rue de Varembé Geneva, Switzerland
Telefax: +41 22 919 0300 e-mail: inmail@iec.ch IEC web site http://www.iec.ch

Commission Electrotechnique Internationale
International Electrotechnical Commission
Международная Электротехническая Комиссия

PRICE CODE X

For price, see current catalogue

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

DESIGN AUTOMATION –

Part 1-1: Harmonization of ATLAS test languages

FOREWORD

1) The IEC (International Electrotechnical Commission) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of the IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, the IEC publishes International Standards. Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. The IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of the IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested National Committees.
3) The documents produced have the form of recommendations for international use and are published in the form of standards, technical specifications, technical reports or guides and they are accepted by the National Committees in that sense.
4) In order to promote international unification, IEC National Committees undertake to apply IEC International Standards transparently to the maximum extent possible in their national and regional standards. Any divergence between the IEC Standard and the corresponding national or regional standard shall be clearly indicated in the latter.
5) The IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with one of its standards.
6) Attention is drawn to the possibility that some of the elements of this technical report may be the subject of patent rights. The IEC shall not be held responsible for identifying any or all such patent rights.

The main task of IEC technical committees is to prepare International Standards. However, a technical committee may propose the publication of a technical report when it has collected data of a different kind from that which is normally published as an International Standard, for example "state of the art".

Technical reports do not necessarily have to be reviewed until the data they provide are considered to be no longer valid or useful by the maintenance team.

IEC 61926-1-1, which is a technical report, has been prepared by IEC technical committee 93: Design automation.

The text of this technical report is based on the following documents:

Enquiry draft	Report on voting
93/93/CDV	93/102/RVC

Full information on the voting for the approval of this technical report can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 3.

This document which is purely informative is not to be regarded as an International Standard.

A bilingual version of the technical report may be issued at a later date.

OVERVIEW

A common standard test language has been of interest to the electronics testing community for many years. Such a common language offers a single communications "medium" for the description of Unit Under Test (UUT) test requirements to both humans and machines, as well as the hope for Test Program Set (TPS) cost savings through code re-use and code sharing (just as a single spoken language would benefit mankind in international communications, or a single computer programming language would allow "anyone" to read/develop/maintain computer software code). The Abbreviated Test Language for All Systems (ATLAS)TM)[1] was developed, and is being maintained, to provide this communications " medium".

This technical report presents the efforts taking place, as well as recommendations/ suggestions to harmonize two differing ATLAS test language specification "dialects", to enable a common use across two user communities

The evolution of the ATLAS language leading to the interest in harmonization of the two dominant representations of this language today, i.e. C/ATLAS 716-95 and ATLAS 626-3, took place in a technological time and context which the reader may find helpful and interesting to know when considering ATLAS today. The following material provides a brief overview of the technical history of Automatic Testing, Automatic Test Equipment and the Testing Economics which existed during the time that ATLAS was evolving. At the conclusion of this background section, a brief history of ATLAS will be included to complete the technological picture and provide the reader with a context for assessing the ATLAS issues being faced today.

The need for an automated means to perform testing followed closely on the heels of the explosive growth in complexity and functionality of the units requiring test. This explosion was driven by miniaturization. More and more capability could be packaged into a single device in the same physical envelope using less and less power and operating at faster and faster speeds.

By the early 1950s, it became clear that a methodology was required which would allow faster testing. The throughput[2] of units through a manufacturer's factory was being limited by a test bottleneck. This was due to the large number of tests required for newer units being designed and built and the limitations of the speeds at which a factory technician could perform these growing number of tests.

In addition to throughput problems, other testing problems were appearing when testing was done manually, including the consistency of test. The need to perform the same tests in the same way every time was too often found to be compromised by the mood, mental state, health and/or interest of the test technician. Additionally, there were qualitative and economic issues involved. The quality of work conditions under which a person is expected to quickly and consistently perform repetitive work with increasing rapidity was coming under question and scrutiny, as was the cost of the human test technician per unit tested.

---

1) ATLAS is a trade mark of the Institute of Electrical and Electronics Engineers.

2) Throughput – the number of units per unit time that can be processed.

A final element to this growing problem was the increasing complexity of the tests which needed to be performed. The tests and testing process reflecting the increasing complexity of the units to be tested became more difficult to perform and interpret. This further exacerbated the time and cost issues noted above by imposing a training cost to enable the technician to perform as required, plus a need for higher skilled technicians who were more costly and more difficult to find. The problems described in respect to the factory environment were being repeated in the field repair environment. Many companies in order to reduce time and shipping costs established field repair and maintenance depots. However, it was not long before these field depots were confronting very similar problems. These problems were made more difficult by the fact that the units requiring test and repair covered a broad variety of types and configurations. This meant that the test technician had less opportunity to become familiar with the traits and characteristics of a single unit. In addition, the field technician was at a remote site, not a factory. Therefore, he needed additional support documentation to compensate for the lack of access to the design engineer available at a factory for advice and guidance. The field technician had to be supported by a large number of expensive spares so that he could effect the needed repair. The expense of the repair, time and spares was at the mercy of the knowledge and diagnostic skill of the repair technician.

Suppliers and users of Automatic Test Equipment (ATE)

The users

On the user side there was a clear dichotomy in the use and application of ATE. The NATO forces were driven by the cold war and the perceived need to extract from technology its benefits in order to support their defense strategy and posture. Production rates, production costs, field maintenance and repair of what was arguably the most sophisticated of electronics were issues that were required to be addressed. Additionally, the ability of NATO to train and retain qualified field test technicians was under strain as local economies improved and increasing numbers of trained technicians left military service.

Commercially the airlines faced increasingly difficult field test and maintenance problems. Driven by concerns over safety, a far-flung set of test and maintenance repair depots and a very difficult avionics requiring test and maintenance, they too began to seek alternative test maintenance and repair approaches.

Other commercial enterprises, particularly those with broad markets and widespread field depot operations were close behind the airlines in identifying the need for a new and improved way to test.

It is safe to say that by the late 1950s all three, i.e., NATO, airlines and large commercial electronics developers were well on their way to developing test solutions, based upon automatic testing.

The suppliers

The suppliers to the three major using communities of ATE were not the same. The suppliers of ATE to the NATO communities were commercial suppliers of standard bench-top instruments configured to be controlled automatically for factory testing, and the suppliers of weapon systems for support of these systems in the field.

The suppliers of ATE to the commercial airlines in the factories were the same as those for NATO, and the suppliers of ATE at the factory test depots were the suppliers of the avionics units used in the aircraft.

The suppliers of ATE to other large commercial suppliers tended to be the suppliers of factory ATE used by NATO and commercial airlines in the factory, and alternative commercial suppliers of ATE in depots or occasionally ATE fashioned by themselves.

Automatic Test Equipment (ATE)

An ATE system has the general configuration shown in figure 1. This configuration is generally applicable from the earliest configuration of ATE to current systems.

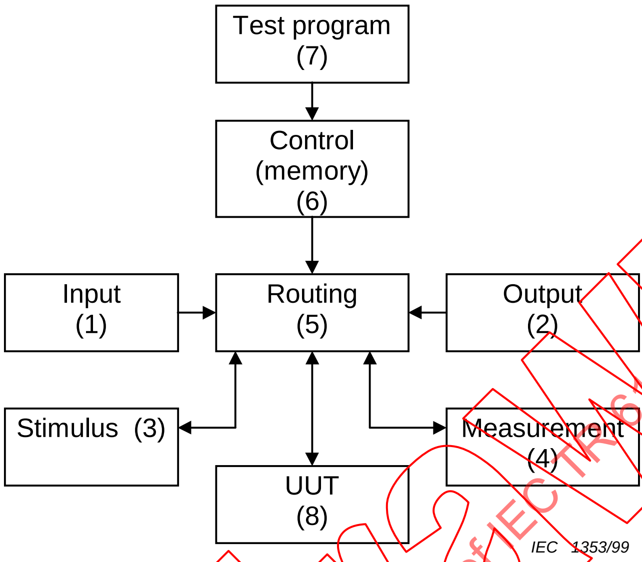


Figure 1 – General configuration of an ATE system

The eight elements of figure 1 perform as follows in any ATE system.

1 Input – the ATE input subsystem allows the ATE operator both to select the operating mode under which the system will perform (i.e. print all test results, stop after each test, print only failures) as well as provide the various media by which the operator can communicate with the system, i.e. tape, Compact Disk (CD), keyboard, floppy.

2 Output – the output subsystem provides the means by which the ATE system communicates with the ATE operator. This can include visual indicators (lights), cathode ray tubes, printers and even voice.

3 Stimulus – the stimulus subsystem consists of programmable[3] devices which can provide either power or signals to a UUT.

4 Measurement – the measurement subsystem consists of programmable devices which can assess the parametric values of power or signals from a UUT.

5 Routing – the routing subsystem consists of switching devices which by program control are capable of interconnecting the output stimulus devices or the input of measurement devices to designated locations on a UUT. The routing subsystem can also route operator inputs to designated devices and/or output information to designated devices.

[3] "Programmable" denotes the ability of having the functional capability of devices controlled by input signals without human intervention.

6 Control – the control subsystem manages the operation of the ATE. It interprets signals from the input subsystem and controls the system operation in accordance with these inputs. The control system also interprets the test instructions contained within a test program and selects the appropriate stimulus device or devices; establishes the routing configuration required to connect the stimulus as required; sends instructions to the stimulus devices to output the signal required; instructs the measurement device required to set up to make the necessary measurement; interconnects the measurement device via the routing subsystem, analyzing the resulting measurement; and selects the next test to be performed predicated upon how the measurement result compares to predetermined limits.

7 Test program – the test program is a coded set of instructions which determines the tests to be performed and the consequence of passing or failing the test.

8 UUT – the UUT or Unit Under Test is the device that is assessed by the ATE in conjunction with the test program.

Evolution of ATE systems

First generation

The first generation of ATE appeared between 1955 and 1965. They were characterized by single function stimulus devices adapted to be programmable by the ATE manufacturers.

Control of these systems was accomplished by specially designed digital devices (not general purpose computers). These devices were typically driven by a perforated tape device.

The test system software consisted of a primitive executive program, normally supported by an off-line assembler designed to process a very low level test language unique to the test system.

The component technology used in these systems consisted of discrete components and vacuum tubes. The input/output devices on these systems consisted of Nixie tubes, other lights or small printers.

These systems were quite large, used a great deal of power and were typically designed for test of a single, specific UUT.

Second generation

The second generation of ATE was found between 1962 and 1972. These ATE systems were characterized by the use of general purpose bench top instruments for stimulus and measurement adapted to be programmable by the instrument manufacturers. These instruments tended to have all their normal manual controls on their front panels even though they were automatically controlled.

Control of the second generation of ATE was accomplished by a general purpose computer having many of the characteristics of today's desktop computers although much larger and more limited in performance, speed and memory size.

The system software was more sophisticated than that found in first generation systems. This sophistication was made possible by the general purpose computer. The software was supported by an off-line compilation system and utilized some type of higher level (human readable) language specially designed by the ATE developer and proprietary to his system.

The component technology used in second generation systems consisted of discrete components and some conductor elements. These systems were still quite large but occupied a much smaller space than equivalent first generation systems. The second generation systems also utilized less power. For certain field applications they could be readily transported, when housed in a small van.

The man-machine interface for second generation systems normally was an operator switch control panel and/or keyboard. Output devices included a printer, Cathode Ray Tube (CRT) and/or other message displays.

The applicability of second generation systems expanded to include classes or families of UUT of a given type and parametric range. Examples could include Power Supplies, AM or FM transceivers or testers of Analog, Digital or Hybrid Printed Circuit boards.

Third generation ATE [6]

Between 1970 and 1980 an ATE class was introduced which relied on the use of more capable computer systems and sophisticated software to replace many of the stimulus and measurement building blocks that had characterized second generation ATE.

The third generation ATE had unique attributes: they utilized digital to analog converters to create signals synthesizing; the signal characteristics were predicated upon the mathematical definition of that signal. Fundamentally, a signal could be synthesized and shaped to form any variety of Alternating Current (AC) signals. Conversely any complex signal could be broken down by a waveform analyzer and its characteristics parameters analyzed.

The advertised benefits of the third generation system were smaller size, since it required less building blocks, and broad flexibility, since any and all test requirements were relegated to a software problem.

The third generation system architecture ushered in the use of sophisticated solid state technology, as well as very sophisticated routing systems for the interconnection of signals.

Fourth generation ATE

The fourth generation of ATE began in the late 1980s and is still seen today. The significant changes introduced by fourth generation ATE was the increasing sophistication of the computer systems, the enormous increase in memory availability and the extensive use of distributed micro-processing, hybrid arrays, and other technological innovations.

Fourth generation systems utilized an array of smart instrumentation. These stimulus measurement and switching devices took a great deal of the burden away from the central computer and executive software. They were capable of scale selection, number conversion, loss analysis and asyn-chronous processing and analysis of test results, resulting in faster and more accurate testing.

Current ATE [7]

The ATE changes today are being driven by a significant expansion of processing capability and by recognition of instrument manufacturers that programmable instruments in ATE systems required their own design attention. The use of instruments on a card unencumbered by the bulk of the bench top instruments as well as use of the asynchronous processing capabilities of these instruments on a card is increasingly common. The range, repertoire and capabilities of instruments on a card are increasingly expanding. Software systems within ATE today are capable of far more than controlling the test process. They archive data, assess trends, provide sophisticated guidance and instruction to operators and assist in management and queuing of UUTs.

Evolution of ATE software

Executive software

The software which runs and manages an ATE operation is generally called the systems executive software. The systems executive software controls the manner in which tests are processed, the selection of tests, the selection of resources to perform the tests, the evaluation of test results and the disposition of the information obtained from performing a test. These tasks are introspective to the testing operation itself. In the early days of ATE, the test executive was only capable of executing a single instruction. Today, this software is capable of compilation, interpretation and selection of the optimum execution of test program software. Over time the role of the executive software has grown to keep pace with the power of the computers which process this software and the capacity of the memory that was available to store and manipulate data. Today's executive software is no longer introspective, but rather encompasses the environment in which the testing takes place, and is capable of multi-processing, i.e. running tasks simultaneously in foreground and background modes or, for that matter, under a hierarchy of priorities.

The executive software is also capable of monitoring the health and well-being of the test system and its resources; dynamically reallocating resources as necessary to perform testing, when and if a system resource fails; maintaining a diary of test processes and test results obtained over a unit time; automatically processing test reports; tracking histories of test results on similar UUTs; tracking spares inventories for repair; and a host of functions designed to integrate the test process into the total life cycle support process.

Application software

The software containing the instructions for testing a UUT connected to the ATE is denoted application software. Normally, outside of factory production lines, ATEs in use today will be required to support many UUTs of similar type and differing configuration or of similar class and differing type, i.e. analog, digital, video, radio frequency (RF), power supply, etc. For field ATEs, the variety of types and configurations can be very broad.

It is not unusual for the cost of application software to exceed the cost of the ATE itself.

Evolution of application software

The application software written for early ATE reflected the lack of the systems and controllers upon which they were run. The test languages used were digital codes, octal codes and a variety of very specific ordered codes often selected from a code menu.

It is worthwhile to mention one other unique aspect of the ATLAS language. It is a virtual language. This means that it is divorced from the test system it is to be run on. The rules of ATLAS require that no reference to the ultimate test system upon which the ATLAS test language is to be executed be included in the program or procedure itself. Thus, an ATLAS program may have a statement such as "APPLY, 50VDC, J-1, J-2 $". It will never have a statement such as "APPLY, BB*55_ _ _$" or "APPLY S*1_ _ _ " referring to a specific resource of a specific test system. This is not to imply that the test engineer who writes the test program or procedure is unaware of the target test system. On the contrary, the test methods and test strategy utilized by the test engineer are guided by the parametric envelope provided by the test system and the specific accuracy and capabilities of the resource suite. The concept of using virtual references is predicated upon the hope and expectation that virtual reference as opposed to explicit resource references will facilitate rehostability of test programs and procedures across differing test platforms.

ATLAS has enjoyed a variety of applications besides its use as a language for test procedures and test programs. It has been used as a basis for writing test specifications and test requirements. In addition, ATLAS has been the basis for test-related specifications such as the IEEE's "Test Equipment Description Language TEDL" (IEEE Std. 993).

Over time, the sophistication of application software improved with the sophistication of the processors used. For a time, a variety of general purpose programming languages were used to design the tests for UUTs. These included FORTRAN, BASIC and C.

The application software used for the ATEs tended each to be unique to the manufacturer of the ATE. Each manufacturer used a differing form, format and language for their own ATE. Often differing ATEs built by the same manufacturer used differing languages.

In the 1960s, both the airlines and defense communities who were major users of automatic test equipment recognized the drawback of diverse languages for each ATE they used. The language differences made them difficult for the user's personnel to learn and understand. This problem was exacerbated when a variety of systems and hence languages were in use. In addition, when the user of an automatic test system desired to add new UUTs to the systems work load, the time and cost to write the program was higher because of the burden of the language. Further, the change of personnel resulting when the designer of a test program using an esoteric language changed jobs, made it difficult for a new person to pick up and maintain the test program that had been written.

At another level, both the airlines and defense communities recognized that the development of a test program for an ATE began with a test procedure by an engineer familiar with the UUT which was then converted into the language of the ATE. They reasoned that it would be of great benefit if both test procedures as well as test programs were written in an unambiguous, human readable form, using explicit testing terminology which only a test engineer could readily understand. This musing formed the genesis for the development of the ATLAS language which will be discussed next.

It should be noted that commercial ATE system suppliers, in the factory, rejected the concept of a common or standardized test language and do so today. The thinking of these vendors was a combination of belief that their unique and proprietary test languages were superior to other test languages, and an understanding that a competitive advantage was possible if customers for their ATE systems could not easily switch to an alternative ATE to support existing UUTs with their attendant test programs once a significant back log had been built up.

Evolution of ATLAS

The quest for a standard testing language began in the 1960s. Aeronautical Radio Incorporated (ARINC) started the development of a standard testing language in response to the needs of commercial airlines. (The language itself was purported to be conceived by Tom Ellison of United Airlines). The commercial airlines had a need to test and repair similar or identical avionics systems on their aircraft. They desired a means by which they could exchange test procedures they had developed in a standardized and unambiguous way, thereby precluding the need for each airline to redevelop these procedures.

The name of the language developed under the auspices of ARINC was the Abbreviated Test Language for Avionics Systems or ATLAS. The development of ATLAS was undertaken through the cooperative and supporting efforts of a large number of commercial companies interested in avionics test and support. These companies provided skilled engineering personnel familiar with the maintenance and support of avionics systems who met together and worked over time to define and develop ATLAS. Over time, the recognition of the need and benefit of a standardized testing language grew beyond the bounds of the commercial airlines.

The United States Army, Navy, Air Force and NATO services became increasingly active in the ATLAS language development efforts. Commercial companies working with these agencies as part of the defense industry also recognized the potential benefit of ATLAS for support of avionics systems. During this period, participation at ATLAS meetings swelled and became an increasingly difficult administrative burden for ARINC.

In 1976, administrative control and responsibility for ATLAS was passed from ARINC to the IEEE. At this time, the name of the language was changed to reflect the broader field of application. ATLAS became the Abbreviated Test Language for All Systems.

Within the IEEE, control of ATLAS was vested in an ad hoc committee which became Standard Coordinating Committee 20 (SCC20). However, from its beginnings under ARINC to the current time the group has been known as the ATLAS committee. This continues despite the fact that the interests of SCC20 have broadened to include other test-related standards.

The first publication of the ATLAS language standard took place in 1968 and was entitled ARINC Specification 416-1. Subsequent versions were published by ARINC when sufficient changes, upgrades, or enhancements had been processed into the language to represent significant improvement to the previously published version.

By 1976, ARINC Specification 416-13A, which was the fourteenth published version of ATLAS had been released. In 1976, IEEE Std. 416-1976 was also published. This was the first ATLAS published under the auspices of the IEEE and represented the ARINC Specification 416-13A ATLAS in IEEE format. Subsequent publications of 416 ATLAS took place through 1988. These publications represented the evolution of ATLAS from version 13A to 33.

By the time version 33 of ATLAS was published the language had grown very large. This growth reflected the sensitivity of the developers to the need for upward compatibility between versions. However, there was increasing concern and pressure to reduce the maintenance burden represented by 416 ATLAS.

In May of 1985, the IEEE published IEEE Std. 716-1985 (C/ATLAS) which represented a common subset of the 416 ATLAS. During the same year ARINC published an ATLAS subset titled ARINC Specification 626-1985. Unfortunately the 716 standard and the 626 specification were not compatible nor were they true subsets of 416 ATLAS. The IEEE ceased to publish 416 ATLAS, and formally withdrew it in 1993. IEEE C/ATLAS has been published in an updated form every three or four years since the initial publication. This is in accordance with IEEE requirements for a revision/affirmation every five years. ARINC 626 ATLAS has followed a similar publication schedule.

In 1984 the IEEE also published standard 771. This standard is a guide to the use of the ATLAS language.

Recently contacts have been made between ARINC, the sponsor and maintainer of ATLAS Specification 626, and members of a technical team associated with maintenance of C/ATLAS Std. 716-95, via the IEEE SCC20 and the use of ATLAS within the NATO community. The purpose of these contacts was to discuss the possibility of coordination and harmonization between the two communities in a variety of potential areas of automatic testing among these areas of ATLAS. The following paper will discuss explicit steps to achieve and subsequently maintain harmonization between these two dialects of ATLAS which represent the largest utilization of the language.

INTRODUCTION

Commercial airline manufacturers and carriers, United States Department of Defense (DoD) and DoD contractors, United Kingdom Ministry of Defence (MoD) and MoD contractors, as well as other NATO member countries MoDs and contractors each use implementations of a test language standard called ATLAS. The ATLAS language standard specification used by the commercial airlines is maintained and published by ARINC. The Common ATLAS (C/ATLAS) standard specification used by the defense industries is maintained and published by the IEEE on behalf of the American National Standards Institute (ANSI).

The ATLAS language specification maintained by ARINC and the C/ATLAS language standard maintained by the IEEE, initially started from a common base line, which over time, has seen the two standards diverge (see figure 2) with respect to language syntax and semantics (e.g. similar to having two dialects of the English language). The International Electrotechnical Commission (IEC) plans to publish soon the IEEE Std. 716-1995 C/ATLAS as IEC standard 61926 – ATLAS. This technical paper is the response to the IEC request that an effort be undertaken to determine how the two diverged dialects of the ATLAS definition could best be brought back into harmonization. This harmonization would allow for a common use of the ATLAS language across both the commercial airline and defense communities. This technical report presents the result of the efforts performed by the authors to first compare and contrast the two ATLAS dialects and second their conclusions and recommendations in respect to achieving harmonization. It is additionally hoped that the efforts performed can be used to point the way towards achieving harmonization between ATLAS implementations, or other test languages and thus facilitate TPS code sharing and TPS code re-use in solving test problems.

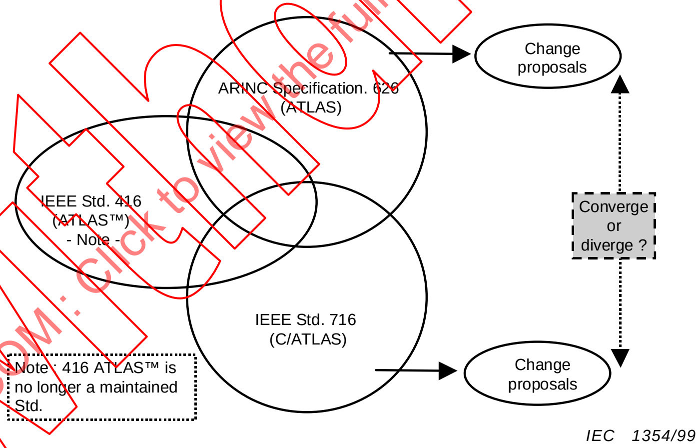


Figure 2 – ATLAS and C/ATLAS evolution

DESIGN AUTOMATION –

Part 1-1: Harmonization of ATLAS test languages

1 Scope

This technical report is applicable to the ATLAS language, the purpose of which is to define a high order language used for the writing of test programs for UUTs, so that these programs can operate on various makes and models of ATE / Automatic Test Systems (ATS).

There are at present two published language definitions of the ATLAS language; this technical report will address the differences between these and provide recommended harmonization and/or convergence of the two standard definitions.

The basis for this technical report are chapters/clauses 1 through 17 of the published IEEE Std. 716-1995 (C/ATLAS) and ARINC Specification 626-3 (ATLAS) publications. The IEEE published IEEE Std. 716-95 in March of 1995, while ARINC published ARINC Specification 626-3 in January of 1995.

2 Reference documents

ARINC Specification 626-3, *Standard ATLAS language for Modular Test* [4)]

ARINC Specification 627, *Programmers Guide for SMART® Systems using ARINC 626 ATLAS*

IEEE Standard 716-1995, *Common Abbreviated Test Language for All Systems (C/ATLAS)* [5)]

IEEE Standard 771-1990, *Guide for the use of ATLAS* [6)]

IEEE Standard 993-1997, *Test Equipment Description Language (TEDL)* [7)]

3 Definitions

For definitions related to the use of C/ATLAS, see IEEE Std. 771; for definitions related to the use of ATLAS, see ARINC Specification 627.

For the purpose of this technical report, the following definitions apply.

3.1
ATLAS
the Standard ATLAS for Modular Test as defined in ARINC Specification 626-3

3.2
C/ATLAS
the Common Abbreviated Test Language for All Systems as defined in IEEE Std. 716-1995

4) Currently being revised – see current IEEE and ARINC Working Groups for further information.

5) Copyright IEEE Standards, Piscataway, NJ.

6) Copyright IEEE Standards, Piscataway, NJ.

7) Copyright IEEE Standards, Piscataway, NJ.

4 Symbols and abbreviations

ABBET	A Broad Based Environment for Test
AMC	Airlines Maintenance Conference
ANSI	American National Standards Institute
ARINC	Aeronautical Radio Incorporated
ATE	Automatic Test Equipment
ATLAS	Abbreviated Test Language for All Systems
ATLAS-2000	Abbreviated Test Language for All Systems in the year 2000
ATS	Automatic Test System (i.e. ATE and TPS)
C/ATLAS	Common Abbreviated Test Language for All Systems
CD	Compact Disk
CRT	Cathode Ray Tube
DoD	Department of Defense
GPS	Global Positioning System
IEEE	Institute of Electrical and Electronics Engineers
IEC	International Electrotechnical Commission
MoD	Ministry of Defence
NATO	North Atlantic Treaty Organization
RF	Radio Frequency
SCC20	Standards Coordinating Committee 20
TPS	Test Program Set
TEDL	Test Equipment Description Language
UUT	Unit Under Test

5 Background information and history

The comparisons between C/ATLAS (revisions published in 1985, 1989, and 1995) and ATLAS originated at the request of the US DoD and NATO, where there were questions about migrating ATE programs from utilizing/implementing one version of C/ATLAS to a later version. This original comparison is contained in annex B.

The basis of the comparison was to evaluate the specification chapter by chapter (for the purpose of this technical report, chapters and clauses are interchangeable terms). The chapters that were compared were 1 through 17. (Formal Syntax – chapter 18 – was not included in the comparisons, but was compared to "validate" the chapter comparisons) Each chapter was "evaluated", to determine the impact of writing a test procedure or program to one or the other of the two specification versions."

Preliminary results and recommendations of the comparison and contrasting of C/ATLAS and ATLAS have previously been published and presented at AUTOTESTCON '96[1], published in PLANE TALK®[8)] [2], and presented to IEC TC 93 WG 7[3].

The results of the comparison between IEEE Std. 716-1995 (C/ATLAS) and ARINC Specification 626-3 (ATLAS) are contained in clauses 6 through 9.

8) PLANE TALK is a registered trademark of Aeronautical Radio Incorporated.

6 Relationship between C/ATLAS and ATLAS

The following relationship exists between C/ATLAS and ATLAS (see figure 3).

a) **Size** – C/ATLAS is larger than ATLAS, due primarily to the fact that defense industry applications cover a broader scope of test requirements than those that are required to support commercial airlines.

   There are 609 language elements in C/ATLAS and 507 language elements in ATLAS. Thus, C/ATLAS is approximately 20 % larger than ATLAS.

b) **Language element differences** – There are 141 language elements in C/ATLAS that are not included in ATLAS. There are 39 language elements in ATLAS that are not included in C/ATLAS.

   The reasons are explained as follows:

   – first, initially C/ATLAS and ATLAS developed incompatible subsets of the IEEE Std. 416 ATLAS syntax, and these differences have been carried forward;

   – second, each has incorporated change proposals to each respective standard. Because these have not necessarily been incorporated the same way (or at all) in the other, further differences have occurred.

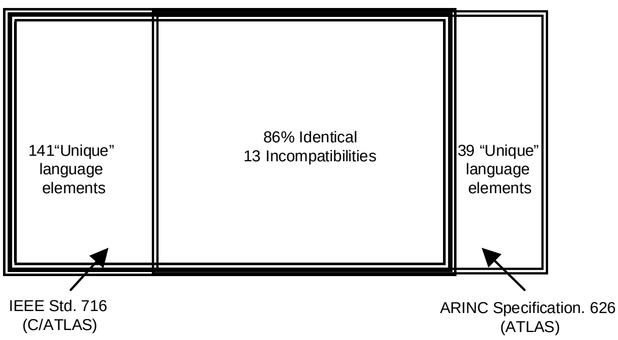
     What this means is that 77 % of the 609 language elements in C/ATLAS are identical to those in ATLAS while 92 % of the total language elements in ATLAS are identical to those in C/ATLAS. Both language dialects together are 86 % identical with respect to language element differences.

c) **Incompatibilities** – Beyond the language element differences noted in item b) above, there are 13 incompatibilities between C/ATLAS and ATLAS.

For the purpose of this technical report, an incompatibility is defined as a rule, operation or application of a programming/testing function, implemented or interpreted differently between the two language dialects.

Of the 13 incompatibilities identified, only four will present problems in regard to achieving compatibility between the two dialects. (A problem in this case represents a cost or additional technical effort in achieving compatibility / harmonization between a test program written in one dialect, being used on an ATE designed around the other dialect.)

The requirements for correction of the four problem incompatibilities, as well as the nine simpler incompatibilities, and the language differences are presented in the harmonization methods clause of this technical report (clause 9).



IEC 1355/99

Figure 3 – ATLAS and C/ATLAS relationship

7 Current events

Since the publication of the C/ATLAS and ATLAS standards in 1995, both the IEEE and ARINC ATLAS language maintenance bodies have been processing proposals for additions and/or changes to the languages.

a) Proposals to C/ATLAS are at this time unique to the C/ATLAS standard, and proposals to ATLAS are unique to the ATLAS specification. The new proposals being considered by the two language maintenance bodies, rather than fostering harmonization, could be viewed as "promoting divergence".

   Proposals being considered by the IEEE SCC20 ATLAS subcommittee to C/ATLAS are "editorial clean-up" of the standard (e.g. inconsistencies etc. are not addressed by any particular C/ATLAS change proposal) The primary inputs have come from the IEC TC 93 reviews of the IEEE Std. 716-1995 publication, and from within the SCC20 ABBET subcommittee, who, among many other efforts, are developing an Ada language binding to the IEEE Std. 716-1995 (C/ATLAS) standard. Enhancements to C/ATLAS are at present primarily being handled as requirements to the ATLAS-2000 language specification which is being developed, also within the IEEE SCC20 committee.

   NOTE ATLAS 2000 is the name given to the newly designed ATLAS language whose planned publication date is the year 2000. This newly designed language will be predicated upon the proven test language foundation and experience gained over the 30 years that ATLAS has been in existence. However, ATLAS 2000 will incorporate capabilities and technical developments in computer language design, telecommunication and computer sciences to place the proven ATLAS capabilities and advantages in a designed environment and architecture which will eliminate or mitigate many of the problems ATLAS users have contended with in the past. It will introduce a level of flexibility and capability which had heretofore not been possible.

b) At the time that this technical report was prepared, there were 11 proposals to be considered by the ARINC ATLAS working group with respect to ATLAS (the meeting to review the proposals was held 24-26 June 1997). These change proposals represent syntactical enhancements and corrections (notes, inconsistencies, etc.) in the areas of:

   1) digital bus communications;
   2) noun modifier usage;
   3) extend syntax;
   4) limit fields;
   5) evaluation fields;
   6) digital;
   7) block statements;
   8) declare statements;
   9) parameters;
   10) timers;
   11) identify statements;
   12) calculate statements;
   13) input/output statements;
   14) control;
   15) Global Positioning System (GPS);
   16) impedance;
   17) a.c. signals, d.c. signals, AM signals, waveforms, fluids, manometric, and triangular/ ramp signals.

As can be seen from the wide variety of language syntax being addressed by the proposals, and since these primarily address a modification to syntax and semantics (C/ATLAS changes are "editorial clean-up"), further divergence of the two dialects is entirely possible.

It is the plan of the authors to present this technical report containing the results of our analysis and recommendations to the IEEE, ARINC and the IEC. It will be our goal to provide a focus for movement towards bringing both languages into harmonization and avoiding further divergence. As a note pertaining to this plan, at the May 1997 Airlines Maintenance Conference (AMC), there was favorable interest shown by both the defense and airline communities, with respect toward "helping each other solve common problems". This "working together" can only help the harmonization efforts.

It is also our hope that the approaches and recommendations presented herein for ATLAS can be used to achieve harmonization between other test language dialects.

8 Implementations of ATLAS and C/ATLAS

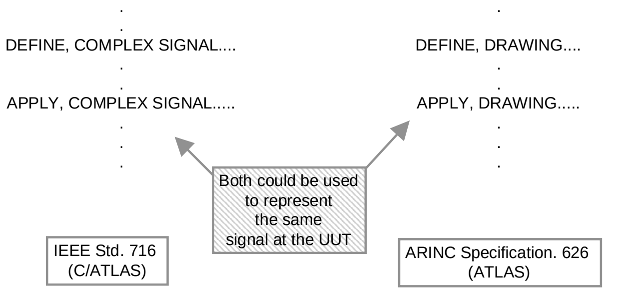
The common baseline from which C/ATLAS and ATLAS evolved is the IEEE Std. 416 ATLAS (see figure 2). The IEEE withdrew the 416-1984 standard in 1993, thus it is no longer maintained (the withdrawal was primarily due to no defense-related business requiring the use of IEEE Std. 416).

IEEE Std. 416 was an overall specification of the ATLAS language, which was not intended to be implemented in its entirety. IEEE Std. 416 contained material (not in C/ATLAS or ATLAS) with respect to the rules towards augmentation and subsets. Before C/ATLAS and ATLAS, vendors developed products to the IEEE Std. 416 definition, utilizing the augmentation and subset "capability" to a great extent. While this was valid with the IEEE Std. 416 definition, the vendors "carried" the augmentation and/or subsets to C/ATLAS products "because ATLAS allows it". What has evolved over the years are ATE unique dialects of languages that vary in their resemblance to the ATLAS language. Much of the dialect was (and is today) instrument specifics unique to the implementation. This technical report makes no attempt at addressing harmonization of the various implementations of published ATLAS and C/ATLAS standards, however the authors wish to emphasize that until procuring agencies enforce the implementation of the specifications as written, implementations of the harmonized standards cannot occur.

9 Harmonization methods

The following recommendations are being made as a means to achieve harmonization between C/ATLAS and ATLAS. The recommendations which follow are technical recommendations as opposed to administrative or organizational initiatives, which will be covered in the conclusions clause of this technical report. Both the technical and administrative/organizational initiatives will be necessary for harmonization.

It would be quite tempting to dispose of the simpler differences and incompatibilities by recommending the simple merger of the two language dialects. In this manner, C/ATLAS would have grown by only 6 % and ATLAS would have grown by 28 %. The resulting "language" would have contained C/ATLAS language elements, some of which would have been redundant in light of existing ATLAS functions, due to differing names for the same elements (see figure 4). The difficulties of the merger approach included increased overhead to implement and maintain the expanded language and the confusion caused by similar language elements having different names. In addition, this approach would not have resolved the problem of incompatibilities between the languages which could not be redressed by a merging of the two.



IEC 1356/99

Figure 4 – Merging example

The recommendations for harmonization which follow are believed to be cost effective, practical and readily implementable.

a) **Clean-up (elimination of unused language elements)** – There are elements within ATLAS whose integration into the language is not complete, i.e. they appear to exist as differences between C/ATLAS and ATLAS but in practical terms they are not used. Thus the elimination of these language elements and any other language elements in ATLAS or C/ATLAS which are basically unused would serve the goal of harmonization by disposing of an apparent if unreal difference. Additionally, the removal of these language elements would have no impact on previously written programs, since the language elements were and are unusable. The language elements in ATLAS that have been identified at this time are <*inst-val*>, <*pos-neg-edge*>, <*number*> and <*modifier-mnemonic*>. It is believed that these are a "carryover" from the IEEE Std. 416 original subset, and that as proposals were addressed that changed overall constructs, these elements of those overall constructs were omitted from the proposal through oversight. The deletion of these unused elements has been submitted by the authors to the ATLAS working group, and these will be addressed as part of the list of proposed changes as described in clause 7, item b).

   Further examination may identify other language elements of this type which should also be removed from ATLAS and/or C/ATLAS.

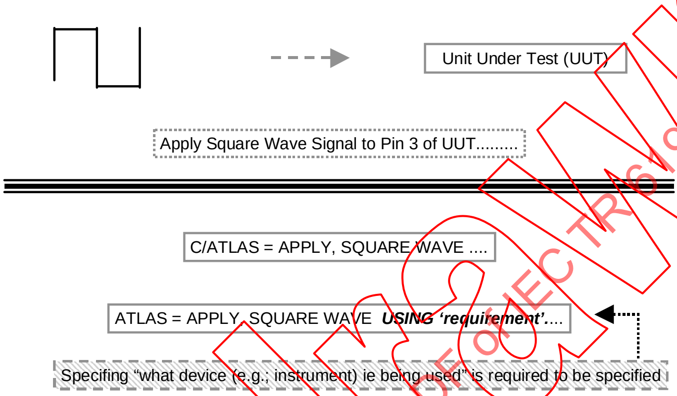
b) **Reflection (extension of language syntax)** – There are language elements in both language dialects that have an identical function, but whose name or keyword is different. The path for harmonization of these elements is to extend the syntax list in both ATLAS and C/ATLAS to accept either one or both of the elements. It will be imperative to ensure that the implementation rules are identical in both ATLAS and C/ATLAS to use this approach for harmonization.

c) **Absorption (expansion of the language element set)** – There are elements in both languages that are functionally and syntactically different from anything in the other. It was noted earlier that there are 141 language elements in C/ATLAS and 39 elements in ATLAS which are not in the other language. However, after clean-up and reflection considerations, these numbers will shrink. The next consideration in respect to absorption would be the value added to the language. Expanding functionality in a compatible manner with the existing language would be no different than the maintenance efforts currently taking place in both communities. Although the impact on maintenance and overhead on C/ATLAS would be smaller than that on ATLAS (adding 35 or less elements as opposed to adding approximately 135) the benefit to each would be an expansion in scope and capability of the language dialect. However, even if one or the other of the language maintenance groups elects not to absorb all of the different language elements not currently within their language sets, the exceptions should concern functional test categories. That is, all general purpose capabilities should be absorbed while specific generic test categories and their associated language elements (e.g. GPS) should be rejected. In this way, a user will have a very clear idea of where there is or is not compatibility between the two language dialects.

d) **Tolerance (the language rules)** – In some cases the lack of compatibility between ATLAS and C/ATLAS is a result of the manner in which the rules for language have been implemented. In some cases, the language semantics (the meaning of the language element) is impacted; in other cases the processing procedures and/or coding requirements are impacted. Some examples of the problems which are manifest as incompatibilities include:

1) In ATLAS, the use of <*requirement*> is a mandatory part of the <*noun field*> while in C/ATLAS, it is optional (see figure 5).

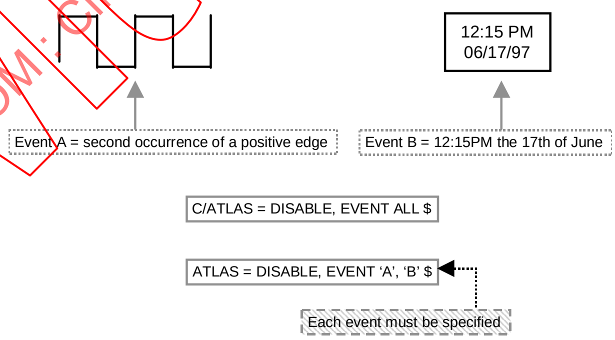
2) Both ATLAS and C/ATLAS have the label *USING*. In one case *USING* is mandatory and in the other it is optional (see figure 5).



IEC 1357/99

Figure 5 – USING label and <*requirement*>s

3) Both ATLAS and C/ATLAS have a <*disable event statement*>. The rules in C/ATLAS allow the capability for disabling all events while the rules for ATLAS require a list of all the events to be disabled (see figure 6).



IEC 1358/99

Figure 6 – Disabling events

As a general approach, all of the language rules in both language dialects must be reviewed. Each opportunity for liberalization of the rules to benefit harmonization should be taken. This approach would have no impact on programs written for a system implemented under the more stringent rules but would provide a harmonization path for programs written under rules which were less restrictive, i.e. under the alternative dialect. Once again, this would enable broader application and use of programs and would facilitate code re-use and sharing between systems designed for either of the two dialects.

e) **Additions (supplements to the existing language processing capabilities)** – After all of the easy victories are obtained and the level of closure to achieve harmonization has been narrowed with relatively direct modifications to the language element set, interpretation and rules, the user will arrive at the few remaining incompatibilities which require something extra be added to language or test systems upon which the language is being implemented. The task of the writers was to keep these to a minimum because these harmonization methods are the least palatable, most costly and involve the highest risk. In the case of ATLAS and C/ATLAS, we believe we have achieved the goal of minimizing this type of addition. The following identifies the additions we believe necessary and their purpose.

   1) C/ATLAS and ATLAS use opposite types of logic to identify PASS/FAIL or GO/NO-GO. C/ATLAS uses a high (positive logic) to denote a PASS or GO condition while ATLAS uses a low (negative logic) to denote the same thing (see figure 7). Harmonization between the two language dialects would be achieved through the use of a software runtime switch which would identify the source of the program, e.g. C/ATLAS or ATLAS, and direct the operating system implementing the program to interpret test results in accordance with the appropriate logic.

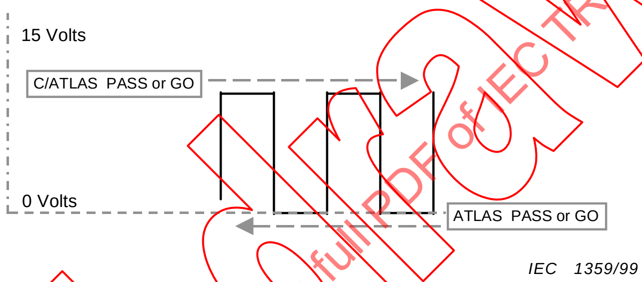
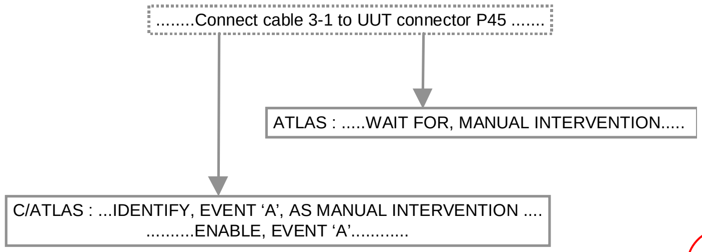


Figure 7 – Logic types

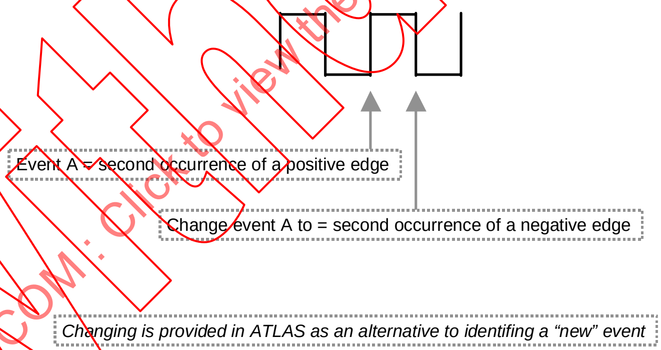
   2) Interrupt handling, particularly the way in which operator interrupts for manual intervention are handled, is significantly different in C/ATLAS and ATLAS (see figure 8). Whereas C/ATLAS treats manual intervention as simply one in a variety of possible events, ATLAS uses a unique program mechanism for this function. The differences between the two dialects are sufficiently great that the writers' only solution, at this time, is for each dialect implementation to add the manual intervention convention of the other into their repertoire. This addition in conjunction with the same software switch noted in 1) above denoting the source of the application (C/ATLAS or ATLAS) would allow interrupts to be handled consistent with the expectations of the software developer.



IEC 1360/99

Figure 8 – Manual intervention

3) In ATLAS <*change statement*>s provide for changing events by configuring an events monitor. The C/ATLAS <*change statement*> does not have this capability (see figure 9). Once again, the most straightforward and maximally upward/downward/harmonization compliant change would be to ensure that both methods for handling <*change statement*>s were present in both language dialects, with a switch available to select and identify the program source and implement the proper process.



IEC 1361/99

Figure 9 – Changing events

10 Conclusions

C/ATLAS and ATLAS are examples of test languages which evolved from a common base and evolved into two incompatible dialects designed to accomplish the same end purpose. The two languages today are 84 % identical, however their differences and incompatibilities prevent them from being used interchangeably on platforms designed for either one or the other. The impact of this is the preclusion of code re-use and sharing between the test communities that utilize each dialect. In addition, separate teams of maintenance experts have been required to evolve each of the dialects to meet evolving test and maintenance needs.

The IEEE and ARINC dialects are amenable to harmonization with relatively small effort. The areas of compatibility can be brought from 84 % to 98 % very quickly and with little cost. The remaining areas of difference and incompatibility are also subject to harmonization, although with some additional cost and effort beyond that required to achieve the greatest gains. The benefits of harmonization, it is felt, will more than pay for its cost due to reduced maintenance costs and the sharing and/or re-use of software, and thus should be seriously considered by both of the standards communities responsible for the two dialects.

The relative ease and feasibility of achieving harmonization between the C/ATLAS and ATLAS test languages implies that similar harmonization may be possible between other test languages. This is particularly true of test languages focused on narrow areas of application, for example back plane and cable test or device test for specific unit types or bare board test. Even at higher levels of complexity or broader focus, it is felt great strides could be made in test language harmonization and the consequent benefits resulting from such harmonization.

11 Recommendations

In the specific case of C/ATLAS and ATLAS the writers have four recommendations to achieve harmonization and avoid further divergence.

a) **Common forum** – Those responsible for maintenance and management of the ATLAS and C/ATLAS standards should seek a common forum which would meet on a regular basis (not less than annually). The purpose of this forum would be to present and share language issues and plans. The goal of the forum should be to preclude those changes and modifications to either of the two dialects which would preclude compatibility with the other. Methods to ensure that both language dialects would maintain harmony once it is again achieved would also fall within the purview of the group.

b) **Proposal handling** – An action which would allow common screening of proposals targeted at each language dialect would be helpful. A proposal screening board consisting of experts from both communities to review proposals to ascertain their impact on dialect compatibility and/or to agree to the sharing of proposals would help to preclude future divergence.

c) **Common maintenance team** – Consideration of a common maintenance team for both languages would be beneficial. This team would consist of experts from the communities using each dialect. It would expand the available expertise and should be sized as required but certainly should involve a number significantly smaller than the sum of both existing maintenance teams.

d) **Initial harmonization** – Both communities should consider an initial effort to implement the required changes suggested by the writers and undertake a beta site test of the resulting changes to ensure that harmonization is successfully achieved. If this is done, each should consider re-issue of their respective standards in a harmonized version.

12 Benefits

The harmonization steps suggested above, if taken, will not only provide upward compatibility across test platforms but will also carry the benefit of expanding opportunities for suppliers in each community to achieve success in the community that they had been excluded from due to language incompatibility. In addition, legacy systems supported by ATS which had become outdated would now become business opportunities for current suppliers of C/ATLAS or ATLAS based systems.

13 Other languages and/or implementations

In respect to other test languages and/or implementations, the writers suggest a serious look at their areas of difference or similarity to determine if the methods and approaches suggested in this paper are applicable. The benefits both to users of these test languages and to their suppliers are believed to be significant. This is true across differing vendor test platforms in the market place as well as across the virtual hierarchy of test platforms within a manufacturing enterprise.

Annex A
(informative)

Comparisons of 416, 716, and 626 definitions – 1984 to 1995.

The attached table is a comparison of IEEE Std. 416-1984 ATLAS; IEEE Stds. 716-1985, 716-1989, and 716-1995 C/ATLAS; and ARINC Specification 626-3 ATLAS. These standards represent the language definition (for language processor development), where the language is an implementation independent UUT test requirements specification language (not an instrument programming language).

IEEE Std. 716-1985 C/ATLAS is a subset of IEEE Std. 416-1984 ATLAS; IEEE Std. 716-1989 and IEEE Std. 716-1995 are enhancements to IEEE Std. 716-1985 to support identified new testing requirements. IEEE Std. 716-1985 to IEEE Std. 716-1989 has an estimated 5 % commonality, IEEE Std. 716-1985 to IEEE Std. 716-1995 has even less commonality. Additionally, 716 is no longer a subset of 416 (416 is no longer maintained as a standard).

The table is intended to be read from left to right, by row. Should the box be empty, this is an indication that the contents are not contained in that publication, either by not having been introduced yet, or having been deleted. The shading contained in the table is an indication of modification of that paragraph from the previous publication. It does not represent any level of complexity of change (note that references to paragraph numbers were not considered a change if they were within a syntax diagram, since paragraph numbers always seem to change).

Legend:

(1) No shade = Introduction of paragraph or no change from the box to the immediate left in that row for 416 to 716-1995. 626-3 is placed next to 716-1995, but no changes/differences are indicated where paragraphs "correlate".

(2) Shaded 10 % indicates changes from (1).

(3) Shaded 30 % indicates changes from (2).

(4) Shaded 50 % indicates changes from (3).

Notes are included at the end of the table.

Table A.1 – Comparisons of 416, 716 and 626

IEEE Std.416-1984: ATLAS (withdrawn as a standard in 1993)	IEEE Std.716-1985: C/ATLAS	IEEE Std.716-1989: C/ATLAS	IEEE Std.716-1995: C/ATLAS	ARINC Specification 626-3: Standard ATLAS Language for Modular Test
1.0 Purpose	1.0 Purpose	1.0 Purpose	1.0 Purpose	1.0 Purpose
1.1 ATLAS Characteristics				1.1 ATLAS Characteristics
1.2 Levels of ATLAS				
				1.2 Management Procedures for the use of Extensibility constructs
1.3 Rules for Creating ATLAS Subsets and ATLAS Subsets with Extensions				
1.4 ATLAS Configuration				
	1.1 Language Processors	1.1 Language Processors	1.1 Language Processors	
	1.2 Document Control	1.2 Document Control	1.2 Document Control	
2.0 Reference Material	2.0 Reference Material	2.0 Reference Material	2.0 Reference Material	2.0 Reference Material
2.1 Applicable Documents	2.1 Applicable Documents	2.1 Applicable Documents	2.1 Applicable Documents	2.1 Applicable Documents
2.2 Document Precedence	2.2 Document Precedence	2.2 Document Precedence	2.2 Document Precedence	2.2 Document Precedence
2.3 How to use this specification				
	2.3 Document Organization and Conventions	2.3 Document Organization and Conventions	2.3 Document Organization and Conventions	
	2.3.1 Document Organization			
		2.3.1 Extensibility	2.3.1 Extensibility	
	2.3.2 Organization of Syntax Specification	2.3.2 Organization of the Syntax Specification	2.3.2 Organization of the Syntax Specification	
			2.3.3 Guide to the use of the C/ATLAS Language	
2.4 Identification of Non Preferred Usage				
3.0 ATLAS Test Specification	3.0 Complete ATLAS Test Program	3.0 Complete C/ATLAS Test Program	3.0 Complete C/ATLAS Test Program	3.0 ATLAS Test Specification
3.1 ATLAS Test Specification Structure	3.1 Complete ATLAS Test Program Structure	3.1 <complete atlas test program structure>	3.1 <complete atlas test program structure>	3.1 <atlas test specification structure>
3.2 Basic Statement Elements	3.2 Basic Statement Elements	3.2 Basic Statement Elements	3.2 Basic Statement Elements	3.2 Basic Statement Elements
3.3 Manual Actions During Testing				3.3 Manual Actions During Testing

Table A.1 – Comparisons of 416, 716 and 626 *(continued)*

IEEE Std.416-1984: ATLAS (withdrawn as a standard in 1993)	**IEEE Std.716-1985: C/ATLAS**	**IEEE Std.716-1989: C/ATLAS**	**IEEE Std.716-1995: C/ATLAS**	**ARINC Specification 626-3: Standard ATLAS Language for Modular Test**
4.0 Structure Delimiter Statements	4.0 Structure Delimiter Statements	4.0 Structure Delimiter Statements	4.0 Structure Delimiter Statements	4.0 Structure Delimiter Statements
4.1 BEGIN/TERMIN-ATE Statements	4.1 BEGIN/TERMIN-ATE Statements	4.1 BEGIN/TERMIN-ATE Statements	4.1 BEGIN/TERMIN-ATE Statements	4.1 BEGIN/TERMINATE Statements
4.1.1 begin atlas program statement	4.1.1 BEGIN ATLAS PROGRAM Statement	4.1.1 BEGIN ATLAS PROGRAM Statement	4.1.1 BEGIN ATLAS PROGRAM Statement	4.1.1 BEGIN ATLAS PROGRAM Statement
4.1.2 terminate atlas program statement	4.1.2 TERMINATE ATLAS PROGRAM Statement	4.1.2 TERMINATE ATLAS PROGRAM Statement	4.1.2 TERMINATE ATLAS PROGRAM Statement	4.1.2 TERMINATE ATLAS PROGRAM Statement
4.1.3 begin atlas module statement	4.1.3 BEGIN ATLAS MODULE Statement	4.1.3 BEGIN ATLAS MODULE Statement	4.1.3 BEGIN ATLAS MODULE Statement	4.1.3 BEGIN ATLAS MODULE Statement
4.1.4 terminate atlas module statement	4.1.4 TERMINATE ATLAS MODULE Statement	4.1.4 TERMINATE ATLAS MODULE Statement	4.1.4 TERMINATE ATLAS MODULE Statement	4.1.4 TERMINATE ATLAS MODULE Statement
4.2 block definition				4.2 Block Definition
4.2.1 block structure				4.2.1 <block structure>
4.2.2 begin block statement				4.2.2 <begin block statement>
4.2.3 block body				4.2.3 <block body>
4.2.4 leave block statement				4.2.4 <leave block statement>
4.2.5 end block statement				4.2.5 <end block statement>
4.3 LEAVE/RESUME ATLAS				
4.4 commence main procedure statement				4.3 Commence Main Procedure Statement
5.0 Reserved for Future Use	5.0 Reserved for Future Use	5.0 Reserved for Future Use	5.0 Reserved for Future Use	5.0 Not Used
6.0 Preamble Statements	6.0 Preamble Statements	6.0 Preamble Statements	6.0 Preamble Statements	6.0 Preamble Statements
6.1 Main Preamble Structure	6.1 <main preamble structure>	6.1 Main Preamble Structure	6.1 Main Preamble Structure	6.1 Main Preamble Structure
6.2 local preamble structure	6.2 <local preamble structure>	6.2 <local preamble structure>	6.2 <local preamble structure>	6.2 <local preamble structure>
6.3 for UUT statement				
6.4 declare statement	6.3 DECLARE Statement	6.3 DECLARE Statement	6.3 DECLARE Statement	6.3 <declare statement>
6.5 DEFINE Statements Definition	6.4 DEFINE Statements Definition	6.4 DEFINE Statements Definition	6.4 DEFINE Statements Definition	
				6.4 <define statement structure>

Table A.1 – Comparisons of 416, 716 and 626 *(continued)*

IEEE Std.416-1984: ATLAS (withdrawn as a standard in 1993)	**IEEE Std.716-1985: C/ATLAS**	**IEEE Std.716-1989: C/ATLAS**	**IEEE Std.716-1995: C/ATLAS**	**ARINC Specification 626-3: Standard ATLAS Language for Modular Test**
6.6 define message statement	6.5 DEFINE MESSAGE Statement	6.5 Deleted		
6.7 define drawing statement				6.5 <define drawing>
	6.6 DEFINE <function> Statement			
6.8 define signal statement		6.6 DEFINE <signal> Statement	6.5 DEFINE <signal> Statement	6.6 <define signal statement>
6.9 PROCEDURE Definition	6.7 PROCEDURE Definition	6.7 PROCEDURE Definition	6.6 PROCEDURE Definition	6.7 PROCEDURE Definition
6.10 FUNCTION Definition				6.8 Function Definition
6.11 TASK Definition				
6.12 require statement	6.8 REQUIRE Statement	6.8 REQUIRE Statement	6.7 REQUIRE Statement	6.9 <require statement>
6.13 include statement	6.9 INCLUDE Statement	6.9 INCLUDE Statement	6.8 INCLUDE Statement	6.10 <include statement>
6.14 CONVERTER Definition				
6.15 PROTOCOL PROCEDURE Definition				
6.16 identify statement		6.10 IDENTIFY Statement	6.9 IDENTIFY Statements	6.12 <identify statements>
6.17 identify timer statement		6.11 IDENTIFY TIMER Statement	6.10 IDENTIFY TIMER Statement	6.13 <identify timer statement>
6.18 identify signal based event statement		6.12 IDENTIFY SIGNAL BASED EVENT Statement	6.11 IDENTIFY SIGNAL BASED EVENT Statement	6.14 <identify signal based event statement>
6.19 identify event based event statement		6.13 IDENTIFY EVENT BASED EVENT Statement	6.12 IDENTIFY EVENT BASED EVENT Statement	6.15 <identify event based event statement>
6.20 identify event interval statement		6.14 IDENTIFY EVENT INTERVAL Statement	6.13 IDENTIFY EVENT INTERVAL Statement	6.16 <identify event interval statement>
6.21 identify event indicator statement		6.15 IDENTIFY EVENT INDICATOR Statement	6.14 IDENTIFY EVENT INDICATOR Statement	6.17 <identify event indicator statement>
6.22 identify time based event statement		6.16 IDENTIFY TIME BASED EVENT Statement	6.15 IDENTIFY TIME BASED EVENT Statement	6.18 <identify time based event statement>
6.23 DIGITAL CONFIGURATION Definition		6.17 DIGITAL CONFIGURATION Definition	6.16 DIGITAL CONFIGURATION Definition	6.19 DIGITAL CONFIGURATION Definition
		6.18 EXTEND Statement	6.17 EXTEND Statement	6.21 <extend atlas statement>
		6.19 ESTABLISH PROTOCOL Statement	6.18 ESTABLISH PROTOCOL Statement	6.11 <establish protocol statement>

Table A.1 – Comparisons of 416, 716 and 626 *(continued)*

IEEE Std.416-1984: ATLAS **(withdrawn as a standard in 1993)**	**IEEE Std.716-1985: C/ATLAS**	**IEEE Std.716-1989: C/ATLAS**	**IEEE Std.716-1995: C/ATLAS**	**ARINC Specification** **626-3: Standard ATLAS Language for Modular Test**
6.24 define exchange statement		6.20 DEFINE EXCHANGE Statement	6.19 DEFINE EXCHANGE Statement	6.20 <define exchange statement>
		6.21 DEFINE DIGITAL TIMING Statement	6.20 DEFINE DIGITAL TIMING Statement	
			6.21 COMPLEX SIGNAL Definition	
			6.22 DEFINE EXCHANGE CONFIGURATION Statement	6.22 <define exchange configuration statement>
7.0 Procedural Statements	7.0 Procedural Statements	7.0 Procedural Statements	7.0 Procedural Statements	7.0 Procedural Statements
7.1 main procedural structure	7.1 <main procedural structure>	7.1 <main procedural structure>	7.1 <main procedural structure>	7.1 <main procedural structure>
7.2 main procedural statements	7.2 <main procedural statements>	7.2 <main procedural statements>	7.2 <main procedural statements>	7.2 <main procedural statements>
8.0 Data Processing Statements				8.0 Data Processing Statements
8.1 procedural statements, data processing	8.0 Procedural Statements, Data Processing	8.0 Procedural Statements, Data Processing	8.0 Procedural Statements, Data Processing	8.1 Procedural Statements, Data Processing
8.2 calculate statement	8.1 CALCULATE Statement	8.1 CALCULATE Statement	8.1 CALCULATE Statement	8.2 CALCULATE Statement
8.3 compare statement		8.2 COMPARE Statement	8.2 COMPARE Statement	8.3 COMPARE Statement
8.4 save statement				
9.0 Input/Output Statements				9.0 Input/Output Statements
9.1 procedural statements input output	9.0 Procedural Statements, Input Output	9.0 Procedural Statements, Input Output	9.0 Procedural Statements, Input Output	9.1 Procedural Statements, Input Output
9.2 fill statement	9.1 FILL Statement	9.1 Deleted		
9.3 wait for manual data statement				
9.4 indicate statement				
9.5 display statement				
9.6 print statement				
9.7 record statement				
9.8 input statement	9.2 INPUT Statement	9.2 INPUT Statement	9.1 INPUT Statement	9.2 INPUT Statement
9.9 output statement	9.3 OUTPUT Statement	9.3 OUTPUT Statement	9.2 OUTPUT Statement	9.3 OUTPUT Statement
		9.4 ENABLE FILE ACCESS Statement	9.3 ENABLE FILE ACCESS Statement	9.4 ENABLE FILE ACCESS Statement

Table A.1 – Comparisons of 416, 716 and 626 *(continued)*

IEEE Std.416-1984: ATLAS **(withdrawn as a standard in 1993)**	**IEEE Std.716-1985: C/ATLAS**	**IEEE Std.716-1989: C/ATLAS**	**IEEE Std.716-1995: C/ATLAS**	**ARINC Specification** **626-3: Standard ATLAS Language for Modular Test**
		9.5 DISABLE FILE ACCESS Statement	9.4 DISABLE FILE ACCESS Statement	9.5 DISABLE FILE ACCESS Statement
				9.6 CREATE FILE Statement
				9.7 DELETE FILE Statement
10.0 Control Statements				10.0 Control Statements
10.1 procedural statements, control	10.0 Procedural Statements, Control	10.0 Procedural Statements, Control	10.0 Procedural Statements, Control	10.1 Procedural Statements, Control
10.2 IF THEN ELSE IF Capability	10.1 IF THEN ELSE Capability	10.1 IF THEN ELSE Capability	10.1 IF THEN ELSE Capability	10.2 IF THEN ELSE Capability
10.3 WHILE THEN Capability	10.2 WHILE THEN Capability	10.2 WHILE THEN Capability	10.2 WHILE THEN Capability	10.3 WHILE THEN Capability
10.4 FOR THEN Capability	10.3 FOR THEN Capability	10.3 FOR THEN Capability	10.3 FOR THEN Capability	10.4 FOR THEN Capability
10.5 go to statement	10.4 GO TO Statement	10.4 GO TO Statement	10.4 GO TO Statement	10.5 GO TO Statement
10.6 repeat statement				
10.7 perform statement	10.5 PERFORM Statement	10.5 PERFORM Statement	10.5 PERFORM Statement	10.6 PERFORM Statement
10.8 finish statement	10.6 FINISH Statement	10.6 FINISH Statement	10.6 FINISH Statement	10.7 FINISH Statement
10.9 ENABLE DISABLE MANUAL INTERVENTION Capability				10.8 ENABLE DISABLE MANUAL INTERVENTION Capability
10.10 wait for manual intervention statement				10.9 wait for manual intervention statement
10.11 ENABLE DISABLE TASK Capability				
10.12 DECISION TABLE Structure Capability				
10.13 discard statement				
10.14 alter statement				
10.15 SUSPEND CONTINUE TASK Capability				
10.16 Task Synchronization Capability				
10.17 enable digital configuration statement		10.7 ENABLE DIGITAL CONFIGURATION Statement	10.7 ENABLE DIGITAL CONFIGURATION Statement	10.10 ENABLE DIGITAL CONFIGURATION Statement

Table A.1 – Comparisons of 416, 716 and 626 *(continued)*

IEEE Std.416-1984: ATLAS (withdrawn as a standard in 1993)	**IEEE Std.716-1985: C/ATLAS**	**IEEE Std.716-1989: C/ATLAS**	**IEEE Std.716-1995: C/ATLAS**	**ARINC Specification 626-3: Standard ATLAS Language for Modular Test**
			10.8 DISABLE DIGITAL CONFIGURATION Statement	
			10.9 Escape Mechanism	10.11 Escape Mechanism
10.18 enable escape statement				
10.19 disable escape statement				
			10.9.1 ENABLE ESCAPE TO PROCEDURE Statement	10.11.1 ENABLE ESCAPE TO PROCEDURE Statement
			10.9.2 DISABLE ESCAPE TO PROCEDURE Statement	10.11.2 DISABLE ESCAPE TO PROCEDURE Statement
	11.0 Procedural Statements Signal			
11.0 Signal Oriented Statements	11.1 Signal Oriented Statements	11.0 Signal Oriented Statements	11.0 Signal Oriented Statements	11.0 Signal Oriented Statements
11.1 procedural statements, signal		11.1 <procedural statements signal>	11.1 <procedural statements signal>	11.1 <procedural statements signal>
11.2 Single Action Verbs	11.2 Single Action Verbs	11.2 Single Action Verb	11.2 Single Action Statements	11.2 Single Action Verb Statements
11.2.1 General Description	11.2.1 General Description	11.2.1 General Description	11.2.1 General Description	11.2.1 General Description
11.2.2 setup statement	11.2.2 SETUP Statement	11.2.2 SETUP Statement	11.2.2 SETUP Statement	11.2.2 SETUP Statement
11.2.3 connect statement	11.2.3 CONNECT Statement	11.2.3 CONNECT Statement	11.2.3 CONNECT Statement	11.2.3 CONNECT Statement
11.2.4 disconnect statement	11.2.4 DISCONNECT Statement	11.2.4 DISCONNECT Statement	11.2.4 DISCONNECT Statement	11.2.4 DISCONNECT Statement
	11.2.5 CLOSE Statement			
11.2.7 arm statement		11.2.5 ARM Statement	11.2.5 ARM Statement	11.2.5 ARM Statement
	11.2.6 OPEN Statement			
11.2.8 fetch statement	11.2.8 FETCH Statement	11.2.6 FETCH Statement	11.2.6 FETCH Statement	11.2.6 FETCH Statement
	11.2.7 INITIATE Statement			
11.2.11 change statement		11.2.7 CHANGE Statement	11.2.7 CHANGE Statement	11.2.7 CHANGE Statement
11.2.12 enable event statement		11.2.8 ENABLE EVENT Statement	11.2.8 ENABLE EVENT Statement	11.2.8 ENABLE EVENT Statement
	11.2.9 COUPLE Statement			

Table A.1 – Comparisons of 416, 716 and 626 *(continued)*

IEEE Std.416-1984: ATLAS (withdrawn as a standard in 1993)	**IEEE Std.716-1985: C/ATLAS**	**IEEE Std.716-1989: C/ATLAS**	**IEEE Std.716-1995: C/ATLAS**	**ARINC Specification 626-3: Standard ATLAS Language for Modular Test**
11.2.13 disable event statement		11.2.9 DISABLE EVENT Statement	11.2.9 DISABLE EVENT Statement	11.2.9 DISABLE EVENT Statement
	11.2.10 UNCOUPLE Statement			
11.2.14 reset statement		11.2.10 RESET Statement	11.2.12 RESET Statement	11.2.10 RESET Statement
			11.2.10 ENABLE COMPLEX SIGNAL Statement	
			11.2.11 DISABLE COMPLEX SIGNAL Statement	
		11.3 Multiple Action Verb	11.3 Multiple Action Statements	
		11.3.1 General Description	11.3.1 General Description	
11.3 apply statement	11.3 APPLY Statement	11.3.2 APPLY Statement	11.3.2 APPLY Statement	11.3 APPLY Statement
11.4 remove statement	11.4 REMOVE Statement	11.3.3 REMOVE Statement	11.3.3 REMOVE Statement	11.4 REMOVE Statement
11.5 measure statement	11.5 MEASURE Statement	11.3.4 MEASURE Statement	11.3.4 MEASURE Statement	11.5 MEASURE Statement
11.6 monitor statement	11.6 MONITOR Statement	11.3.5 MONITOR Statement	11.3.5 MONITOR Statement	11.6 MONITOR Statement
11.7 verify statement	11.7 VERIFY Statement	11.3.6 VERIFY Statement	11.3.6 VERIFY Statement	11.7 VERIFY Statement
11.8 ADJUST Capability				
11.9 read statement	11.8 READ Statement	11.3.7 READ Statement	11.3.7 READ Statement	11.8 READ Statement
11.10 initiate statement		11.3.8 INITIATE Statement	11.3.8 INITIATE Statement	11.9 INITIATE Statement
11.11 Digital Verbs		11.4 Digital Verb	11.4 Digital Statements	11.10 Digital Verbs
		11.4.1 General Description	11.4.1 General Description	
11.11.1 stimulate statement		11.4.2 STIMULATE Statement	11.4.2 STIMULATE Statement	11.10.1 STIMULATE Statement
11.11.2 sense statement		11.4.3 SENSE Statement	11.4.3 SENSE Statement	11.10.2 SENSE Statement
11.11.3 prove statement		11.4.4 PROVE Statement	11.4.4 PROVE Statement	11.10.3 PROVE Statement
12.0 Timing Statements				12.0 Timing Statements
12.1 procedural statements, timing	12.0 Procedural Statements Timing	12.0 Procedural Statements Timing	12.0 Procedural Statements Timing	12.1 Procedural Statements Timing
	12.1 General	12.1 General	12.1 General	

Table A.1 – Comparisons of 416, 716 and 626 *(continued)*

IEEE Std.416-1984: ATLAS **(withdrawn as a standard in 1993)**	**IEEE Std.716-1985: C/ATLAS**	**IEEE Std.716-1989: C/ATLAS**	**IEEE Std.716-1995: C/ATLAS**	**ARINC Specification** **626-3: Standard ATLAS Language for Modular Test**
12.2 delay statement	12.2 DELAY Statement	12.2 Deleted		
12.3 wait for time statement	12.3 WAIT FOR Time Statement	12.3 Deleted		
12.4 PREPARE/ EXECUTE Capability				
12.5 ENABLE DISABLE TIME-LIMIT Capability				
12.6 wait for event statement	12.4 WAIT FOR Event Statement	12.4 Deleted		
12.7 TIME INTERVAL Measurement Capability	12.5 TIME INTERVAL Measurement Capability			12.3 TIME INTERVAL Measurement Capability
12.7.1 time interval measurement structure	12.5.1 TIME INTERVAL Measurement Structure			
12.7.2 START Statement	12.5.2 START Statement			
12.7.3 START WHEN	12.5.3 START WHEN Statement			
12.7.5 STOP Statement	12.5.4 STOP Statement			
12.7.6 STOP WHEN Statement	12.5.5 STOP WHEN Statement			
12.7.7 DISABLE TIME INTERVAL Statement	12.5.6 DISABLE TIME INTERVAL Statement			
		12.5 READ TIMER Statement	12.2 READ TIMER Statement	12.6 READ TIMER Statement
12.7.8 event based measurements				
12.8 Synchronization Capability				
12.8.1 sync statement				
12.8.2 SYNC WHEN Statement	12.6 SYNC WHEN Statement			
		12.6 WAIT FOR Statement	12.3 WAIT FOR Statement	12.2 WAIT FOR Statement
		12.7 RESET TIMER Statement	12.4 RESET TIMER Statement	12.5 RESET TIMER Statement
12.9 DO/END DO CAPABILITY		12.8 DO/END DO Capability	12.5 DO/END DO Capability	12.4 DO/END DO Capability
12.9.1 do simultaneous structure		12.8.1 <do simultaneous structure>	12.5.1 <do simultaneous structure>	12.4.1 <do simultaneous structure>
12.9.2 do simultaneous statement		12.8.2 DO SIMULTANEOUS Statement	12.5.2 DO SIMULTANEOUS Statement	12.4.2 DO SIMULTANEOUS Statement

Table A.1 – Comparisons of 416, 716 and 626 *(continued)*

IEEE Std.416-1984: ATLAS **(withdrawn as a standard in 1993)**	**IEEE Std.716-1985: C/ATLAS**	**IEEE Std.716-1989: C/ATLAS**	**IEEE Std.716-1995: C/ATLAS**	**ARINC Specification** **626-3: Standard ATLAS Language for Modular Test**
12.9.3 do simultaneous body		12.8.3 <do simultaneous body>	12.5.3 <do simultaneous body>	12.4.3 <do simultaneous body>
12.9.4 end do statement		12.8.4 END DO Statement	12.5.4 END DO Statement	12.4.4 END DO Statement
12.9.5 do timed digital structure		12.8.5 <do timed digital structure>	12.5.5 <do timed digital structure>	12.4.5 <do timed digital structure>
12.9.6 do timed digital statement		12.8.6 DO TIMED DIGITAL Statement	12.5.6 DO TIMED DIGITAL Statement	12.4.6 DO TIMED DIGITAL Statement
12.9.7 do timed digital body		12.8.7 <do timed digital body>	12.5.7 <do timed digital body>	12.4.7 <do timed digital body>
				12.4.8 <do sequential structure>
				12.4.9 DO SEQUENTIAL Statement
				12.4.10 <do sequential body>
12.10 reset timer statement				
13.0 Procedural Statements, Macros	13.0 Procedural Statements, Macros	13.0 Procedural Statements, Macro	13.0 Procedural Statements, Databus	13.0 Databus Statements
13.1 DO DIGITAL TEST	13.1 DO DIGITAL TEST			
13.2 do digital test stim only statement	13.2 The STIM ONLY Function			
13.3 do digital test resp comp statement	13.3 The RESP COMP Function			
13.4 do digital test stim resp save statement	13.4 The STIM RESP SAVE Function			
13.5 do digital test stim resp comp statement	13.5 The STIM RESP COMP Function			
13.6 do digital test stim resp match statement				
13.7 do digital test resp only statement	13.6 The RESP ONLY Function			
13.8 do digital test resp select statement	13.7 The RESP SELECT Function			
13.9 Illustrations of DO statements	13.8 Illustrations of DO Statements			
13.10 Ancillary Source and Sensor Statements	13.9 Ancillary Source and Sensor Statements			
13.11 Illustration of Test Function Sequences	13.10 Illustration of Test Function Sequences			
		13.1 <procedural statement macro>	13.1 <procedural statement databus>	

Table A.1 – Comparisons of 416, 716 and 626 *(continued)*

IEEE Std.416-1984: ATLAS **(withdrawn as a standard in 1993)**	**IEEE Std.716-1985: C/ATLAS**	**IEEE Std.716-1989: C/ATLAS**	**IEEE Std.716-1995: C/ATLAS**	**ARINC Specification** **626-3: Standard ATLAS Language for Modular Test**
13.12 do exchange statement		13.3 DO EXCHANGE Statement	13.2 DO EXCHANGE Statement	13.1 DO EXCHANGE Statement
13.13 update exchange statement		13.4 UPDATE EXCHANGE Statement		
			13.3 UPDATE EXCHANGE CONFIGURATION Statement	13.2 UPDATE EXCHANGE CONFIGURATION Statement
			13.4 FETCH EXCHANGE CONFIGURATION Statement	13.7 FETCH EXCHANGE CONFIGURATION Statement
			13.5 ENABLE EXCHANGE CONFIGURATION Statement	13.3 ENABLE EXCHANGE CONFIGURATION Statement
			13.6 CONNECT EXCHANGE CONFIGURATION Statement	13.4 CONNECT EXCHANGE CONFIGURATION Statement
			13.7 DISCONNECT EXCHANGE CONFIGURATION Statement	13.5 DISCONNECT EXCHANGE CONFIGURATION Statement
			13.8 DISABLE EXCHANGE CONFIGURATION Statement	13.6 DISABLE EXCHANGE CONFIGURATION Statement
14.0 Field and Subfield Definitions	14.0 Field and Subfield Definition	14.0 Field and Subfield Definition	14.0 Field and Subfield Definition	14.0 Field and Subfield Definition
14.1 statement characteristics	14.1 <statement characteristics>	14.1 <statement characteristics>	14.1 <statement characteristics>	14.1 <statement characteristics>
14.2 real characteristic subfield	14.2 <real characteristic subfield>	14.2 <real characteristic subfield>	14.2 <real characteristic subfield>	14.2 <real characteristic subfield>
14.3 complex characteristic subfield				
14.4 digital characteristic subfield	14.3 <digital characteristic subfield>	14.3 <digital characteristic subfield>	14.3 <digital characteristic subfield>	
14.5 sync subfield	14.4 <sync subfield>	14.4 <sync subfield>	14.4 <sync subfield>	14.3 <sync subfield>
14.6 real errlim	14.5 <real errlim>	14.5 <real errlim>	14.5 <real errlim>	14.4 <real errlim>
14.7 dim lim	14.6 <dim lim>	14.6 Deleted		14.5 <dim lim>
14.8 pc lim	14.7 <pc lim>	14.7 Deleted		14.6 <pc lim>
14.9 complex errlim				
14.10 measured characteristic	14.8 <measured characteristic>	14.8 <measured characteristic>	14.6 <measured characteristic>	14.7 <measured characteristic>
14.11 evaluation field	14.9 <evaluation field>	14.9 <evaluation field>	14.7 <evaluation field>	14.8 <evaluation field>

Table A.1 – Comparisons of 416, 716 and 626 *(continued)*

IEEE Std.416-1984: ATLAS (withdrawn as a standard in 1993)	IEEE Std.716-1985: C/ATLAS	IEEE Std.716-1989: C/ATLAS	IEEE Std.716-1995: C/ATLAS	ARINC Specification 626-3: Standard ATLAS Language for Modular Test
14.12 eval statement characteristics	14.10 <eval statement characteristics>	14.10 <eval statement characteristics>	14.8 <eval statement characteristics>	14.9 <eval statement characteristics>
14.13 slope eval statement characteristics	14.11 <slope eval statement characteristics>	14.11 Deleted		
14.14 max time	14.12 <max time>	14.12 <max time>	14.9 <max time>	14.10 <max time>
14.15 min time				
14.16 time quantity	14.13 <time quantity>	14.13 <time quantity>	14.10 <time quantity>	14.11 <time quantity>
14.17 condition	14.14 <condition>	14.14 <condition>	14.11 <condition>	14.12 <condition>
14.18 mod expression				
14.19 arithmetic expression	14.15 <arithmetic expression>	14.15 Deleted		
14.20 mathematical function	14.16 <mathematical function>	14.16 Deleted		
14.21 boolean expression	14.17 <boolean expression>	14.17 Deleted		
14.22 shifting expression				
14.23 conn	14.18 <conn>	14.18 <conn>	14.12 <conn>	14.13 <conn>
14.24 conn set	14.19 <conn set>	14.19 <conn set>	14.13 <conn set>	14.14 <conn set>
14.25 format 1	14.20 <format 1>	14.20 Deleted		
14.26 format 2	14.21 <format 2>	14.21 Deleted		
14.27 digital number descriptor	14.22 <digital number descriptor>	14.22 Deleted		
14.28 signal value	14.23 <signal value>	14.23 <signal value>	14.14 <signal value>	14.15 <signal value>
14.29 step				
14.30 value	14.24 <value>	14.24 Deleted		
14.31 variables	14.25 <variables>	14.25 Deleted		
14.32 message variables	14.26 <message variables>	14.26 Deleted		
14.33 words	14.27 <words>	14.27 Deleted		
14.34 stim interval				
14.35 resp delay	14.28 <resp delay>	14.28 Deleted		
14.36 rate/bit	14.29 <rate/bit>	14.29 Deleted		
14.37 rate/data	14.30 <rate/data>	14.30 Deleted		
14.38 rate/word	14.31 <rate/word>	14.31 Deleted		
14.39 list range	14.32 <list range>	14.32 Deleted		
14.40 info	14.33 <info>	14.33 Deleted		
14.41 real quantity	14.34 <real quantity>	14.34 <real quantity>	14.15 <real quantity>	14.16 <real quantity>
14.42 complex quantity				