

# AD-A277 981 INFORMATION PAGE

Form Approved  
OPM No.

2

P  
a  
s  
s  
e  
r  
t



average 1 hour per response, including the time for reviewing instructions, searching existing data sources gathering information. Send comments regarding this burden estimate or any other aspect of this collection of information, including accuracy, relevance, and usefulness, to the Office of Management and Budget, Paperwork Project, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302.

|                             |           |                          |
|-----------------------------|-----------|--------------------------|
| 1. AGENCY USE (Leave blank) | 2. REPORT | 3. REPORT TYPE AND DATES |
|-----------------------------|-----------|--------------------------|

|  |            |
|--|------------|
| 4. TITLE AND SUBTITLE<br>Green Valley Software, Inc., 450427 11/82<br>Compiler. C-Ada. Version 1.1 | 5. FUNDING |
|--|------------|

|  |
|--|
| 6. AUTHOR(S)<br>National Institute of Standards and Technology<br>Gaithersburg, Maryland |
|--|

|  |                            |
|--|----------------------------|
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)<br>National Institute of Standards and Technology<br>Building 255, Room A266<br>Gaithersburg, Maryland 20899<br>USA | 8. PERFORMING ORGANIZATION |
|--|----------------------------|

|  |                                  |
|--|----------------------------------|
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)<br>Ada Joint Program Office<br>The Pentagon, Rm 3E118<br>Washington, DC 20301-3080 | 10. SPONSORING/MONITORING AGENCY |
|--|----------------------------------|

**DTIC**  
**ELECTE**  
**APR 11 1994**  
**S G D**

|                         |
|-------------------------|
| 11. SUPPLEMENTARY NOTES |
|-------------------------|

|   |                   |
|---|-------------------|
| 12a. DISTRIBUTION/AVAILABILITY STATEMENT<br>Approved for Public Release; distribution unlimited | 12b. DISTRIBUTION |
|---|-------------------|

|  |
|--|
| 13. ABSTRACT<br>(Maximum 200)<br>Host/Target. ZENY 386 (under UNIX System V/386)<br>Release 3.2) |
|--|

|  |                     |
|--|---------------------|
| 14. SUBJECT TERMS<br>Ada programming language, Ada Compiler Validation Summary Report, Ada Compiler Val. Capability Val. Testing, Ada Val. Office, Ada Val. Facility<br>ANSI/MIL-STD-1815A, AJPO | 15. NUMBER OF PAGES |
|--|---------------------|

|   |   |   |  |
|---|---|---|--|
| 17. SECURITY CLASSIFICATION<br>UNCLASSIFIED | 18. SECURITY CLASSIFICATION<br>UNCLASSIFIED | 19. SECURITY CLASSIFICATION<br>UNCLASSIFIED | 20. LIMITATION OF ABSTRACT<br>UNCLASSIFIED |
|---|---|---|--|

AVF Control Number: NIST93GVS520\_1\_1.11  
 DATE COMPLETED  
 BEFORE ON-SITE: 93-08-23  
 AFTER ON-SITE: 93-09-30  
 REVISIONS:

Ada COMPILER  
 VALIDATION SUMMARY REPORT:  
 Certificate Number: 930927S1.11328  
 Green Valley Software  
 C\_Ada, Version 1.1  
 ZENY 386 => ZENY 386

|                     |                                     |
|---------------------|-------------------------------------|
| Accession For       |                                     |
| NTIS CRA&I          | <input checked="" type="checkbox"/> |
| DTIC TAB            | <input type="checkbox"/>            |
| Unannounced         | <input type="checkbox"/>            |
| Justification ..... |                                     |
| By .....            |                                     |
| Distribution/       |                                     |
| Availability Codes  |                                     |
| Dist                | Avail and/or<br>Special             |
| A-1                 |                                     |

Prepared By:  
 Software Standards Validation Group  
 Computer Systems Laboratory  
 National Institute of Standards and Technology  
 Building 225, Room A266  
 Gaithersburg, Maryland 20899  
 U.S.A.

94-10865



DTIC QUALITY INSPECTED

94 4 8 059

AVF Control Number: NIST93GVS520\_1\_1.11

Certificate Information

The following Ada implementation was tested and determined to pass ACVC 1.11. Testing was completed on September 27, 1993.

Compiler Name and Version: C\_Ada, Version 1.1

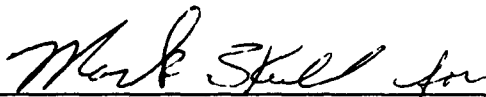
Host Computer System: ZENY 386 under UNIX System V/386, Release 3.2

Target Computer System: ZENY 386 under UNIX System V/386, Release 3.2

See section 3.1 for any additional information about the testing environment.

As a result of this validation effort, Validation Certificate 930927S1.11328 is awarded to Green Valley Software. This certificate expires 2 years after ANSI/MIL-STD-1815B is approved by ANSI.

This report has been reviewed and is approved.

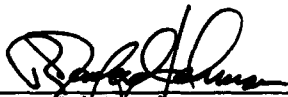



Ada Validation Facility  
Dr. David K. Jefferson  
Chief, Information Systems  
Engineering Division (ISED)

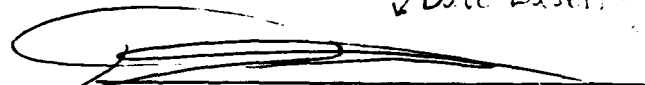


Ada Validation Facility  
Mr. L. Arnold Johnson  
Manager, Software Standards  
Validation Group

Computer Systems Laboratory (CSL)  
National Institute of Standards and Technology  
Building 225, Room A266  
Gaithersburg, Maryland 20899  
U.S.A.



 Ada Validation Organization  
Director, Computer & Software  
Engineering Division  
Institute for Defense Analyses  
Alexandria, Virginia 22311  
U.S.A.



Ada Joint Program Office  
M. Dirk Rogers, Major, USAF  
Acting Director  
Ada Joint Program Office  
Washington, DC 20301  
U.S.A.

✓ Dave East, 2/2/94  
10/27/93  
10/27/93

NIST93GVS520\_1\_1.11

DECLARATION OF CONFORMANCE

The following declaration of conformance was supplied by the customer.

Customer: Green Valley Software  
Certificate Awardee: Green Valley Software  
Ada Validation Facility: National Institute of Standards and  
Technology  
Computer Systems Laboratory (CSL)  
Software Standards Validation Group  
Building 225, Room A266  
Gaithersburg, Maryland 20899  
U.S.A.

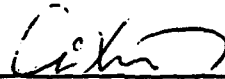
ACVC Version: 1.11

Ada Implementation:

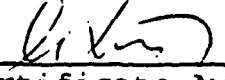
Compiler Name and Version: C\_Ada, Version 1.1  
Host Computer System: ZENY 386 under UNIX System V/386,  
Release 3.2  
Target Computer System: ZENY 386 under UNIX System V/386,  
Release 3.2

Declaration:

I, the undersigned, declare that I have no knowledge of deliberate deviations from the Ada Language Standard ANSI/MIL-STD-1815A ISO 8652-1987 in the implementation listed above.

  
\_\_\_\_\_  
Customer Signature  
Li Xin  
Company Green Valley Software  
Title: Vice President, Green Vally Software

Sept. 27, 1993  
Date

  
\_\_\_\_\_  
Certificate Awardee Signature  
Li Xin  
Company Green Valley Software  
Title: Vice President, Green Vally Software

Sept 27, 1993  
Date

## TABLE OF CONTENTS

|  |     |
|--|-----|
| CHAPTER 1.....                                 | 1-1 |
| INTRODUCTION.....                              | 1-1 |
| 1.1 USE OF THIS VALIDATION SUMMARY REPORT..... | 1-1 |
| 1.2 REFERENCES.....                            | 1-2 |
| 1.3 ACVC TEST CLASSES.....                     | 1-2 |
| 1.4 DEFINITION OF TERMS.....                   | 1-3 |
| CHAPTER 2.....                                 | 2-1 |
| IMPLEMENTATION DEPENDENCIES.....               | 2-1 |
| 2.1 WITHDRAWN TESTS.....                       | 2-1 |
| 2.2 INAPPLICABLE TESTS.....                    | 2-1 |
| 2.3 TEST MODIFICATIONS.....                    | 2-4 |
| CHAPTER 3.....                                 | 3-1 |
| PROCESSING INFORMATION.....                    | 3-1 |
| 3.1 TESTING ENVIRONMENT.....                   | 3-1 |
| 3.2 SUMMARY OF TEST RESULTS.....               | 3-1 |
| 3.3 TEST EXECUTION.....                        | 3-2 |
| APPENDIX A.....                                | A-1 |
| MACRO PARAMETERS.....                          | A-1 |
| APPENDIX B.....                                | B-1 |
| COMPILATION SYSTEM OPTIONS.....                | B-1 |
| LINKER OPTIONS.....                            | B-1 |
| APPENDIX C.....                                | C-1 |
| APPENDIX F OF THE Ada STANDARD.....            | C-1 |

## CHAPTER 1

### INTRODUCTION

The Ada implementation described above was tested according to the Ada Validation Procedures [Pro92] against the Ada Standard [Ada83] using the current Ada Compiler Validation Capability (ACVC). This Validation Summary Report (VSR) gives an account of the testing of this Ada implementation. For any technical terms used in this report, the reader is referred to [Pro92]. A detailed description of the ACVC may be found in the current ACVC User's Guide [UG89].

#### 1.1 USE OF THIS VALIDATION SUMMARY REPORT

Consistent with the national laws of the originating country, the Ada Certification Body may make full and free public disclosure of this report. In the United States, this is provided in accordance with the "Freedom of Information Act" (5 U.S.C. #552). The results of this validation apply only to the computers, operating systems, and compiler versions identified in this report.

The organizations represented on the signature page of this report do not represent or warrant that all statements set forth in this report are accurate and complete, or that the subject implementation has no nonconformities to the Ada Standard other than those presented. Copies of this report are available to the public from the AVF which performed this validation or from:

National Technical Information Service  
5285 Port Royal Road  
Springfield, Virginia 22161  
U.S.A.

Questions regarding this report or the validation test results should be directed to the AVF which performed this validation or to:

Ada Validation Organization  
Computer and Software Engineering Division  
Institute for Defense Analyses  
1801 North Beauregard Street  
Alexandria, Virginia 22311-1772  
U.S.A.

## 1.2 REFERENCES

- [Ada83] Reference Manual for the Ada Programming Language, ANSI/MIL-STD-1815A, February 1983 and ISO 8652-1987.
- [Pro92] Ada Compiler Validation Procedures, Version 3.1, Ada Joint Program Office, August 1992.
- [UG89] Ada Compiler Validation Capability User's Guide, 21 June 1989.

## 1.3 ACVC TEST CLASSES

Compliance of Ada implementations is tested by means of the ACVC. The ACVC contains a collection of test programs structured into six test classes: A, B, C, D, E, and L. The first letter of a test name identifies the class to which it belongs. Class A, C, D, and E tests are executable. Class B and class L tests are expected to produce errors at compile time and link time, respectively.

The executable tests are written in a self-checking manner and produce a PASSED, FAILED, or NOT APPLICABLE message indicating the result when they are executed. Three Ada library units, the packages REPORT and SPRT13, and the procedure CHECK\_FILE are used for this purpose. The package REPORT also provides a set of identity functions used to defeat some compiler optimizations allowed by the Ada Standard that would circumvent a test objective. The package SPRT13 is used by many tests for Chapter 13 of the Ada Standard. The procedure CHECK\_FILE is used to check the contents of text files written by some of the Class C tests for Chapter 14 of the Ada Standard. The operation of REPORT and CHECK\_FILE is checked by a set of executable tests. If these units are not operating correctly, validation testing is discontinued.

Class B tests check that a compiler detects illegal language usage. Class B tests are not executable. Each test in this class is compiled and the resulting compilation listing is examined to verify that all violations of the Ada Standard are detected. Some of the class B tests contain legal Ada code which must not be flagged illegal by the compiler. This behavior is also verified.

Class L tests check that an Ada implementation correctly detects violation of the Ada Standard involving multiple, separately compiled units. Errors are expected at link time, and execution is attempted.

In some tests of the ACVC, certain macro strings have to be replaced by implementation-specific values--for example, the

largest integer. A list of the values used for this implementation is provided in Appendix A. In addition to these anticipated test modifications, additional changes may be required to remove unforeseen conflicts between the tests and implementation-dependent characteristics. The modifications required for this implementation are described in section 2.3.

For each Ada implementation, a customized test suite is produced by the AVF. This customization consists of making the modifications described in the preceding paragraph, removing withdrawn tests (see section 2.1) and, possibly some inapplicable tests (see Section 3.2 and [UG89]).

In order to pass an ACVC an Ada implementation must process each test of the customized test suite according to the Ada Standard.

#### 1.4 DEFINITION OF TERMS

|   |   |
|---|---|
| Ada Compiler                              | The software and any needed hardware that have to be added to a given host and target computer system to allow transformation of Ada programs into executable form and execution thereof.                           |
| Ada Compiler Validation Capability (ACVC) | The means for testing compliance of Ada implementations, Validation consisting of the test suite, the support programs, the ACVC Capability User's Guide and the template for the validation summary (ACVC) report. |
| Ada Implementation                        | An Ada compiler with its host computer system and its target computer system.   |
| Ada Joint Program Office (AJPO)           | The part of the certification body which provides policy and guidance for the Ada certification Office system.  |
| Ada Validation Facility (AVF)             | The part of the certification body which carries out the procedures required to establish the compliance of an Ada implementation.  |
| Ada Validation Organization (AVO)         | The part of the certification body that provides technical guidance for operations of the Ada certification system.   |
| Compliance of an Ada Implementation       | The ability of the implementation to pass an ACVC version.  |



|                            |   |
|----------------------------|---|
| Computer System            | A functional unit, consisting of one or more computers and associated software, that uses common storage for all or part of a program and also for all or part of the data necessary for the execution of the program; executes user-written or user-designated programs; performs user-designated data manipulation, including arithmetic operations and logic operations; and that can execute programs that modify themselves during execution. A computer system may be a stand-alone unit or may consist of several inter-connected units. |
| Conformity                 | Fulfillment by a product, process, or service of all requirements specified.  |
| Customer                   | An individual or corporate entity who enters into an agreement with an AVF which specifies the terms and conditions for AVF services (of any kind) to be performed.   |
| Declaration of Conformance | A formal statement from a customer assuring that conformity is realized or attainable on the Ada implementation for which validation status is realized.  |
| Host Computer System       | A computer system where Ada source programs are transformed into executable form.   |
| Inapplicable Test          | A test that contains one or more test objectives found to be irrelevant for the given Ada implementation.   |
| ISO                        | International Organization for Standardization.   |
| LRM                        | The Ada standard, or Language Reference Manual, published as ANSI/MIL-STD-1815A-1983 and ISO 8652-1987. Citations from the LRM take the form "<section>.<subsection>:<paragraph>."  |
| Operating System           | Software that controls the execution of programs and that provides services such as resource allocation, scheduling, input/output control, and data management. Usually, operating systems are predominantly software, but partial or complete hardware implementations are possible.   |
| Target Computer System     | A computer system where the executable form of Ada programs are executed.   |

Validated Ada  
Compiler

The compiler of a validated Ada  
implementation.

Validated Ada  
Implementation

An Ada implementation that has been  
validated successfully either by AVF testing  
or by registration [Pro92].

Validation

The process of checking the conformity of an  
Ada compiler to the Ada programming language  
and of issuing a certificate for this  
implementation.

Withdrawn Test

A test found to be incorrect and not used in  
conformity testing. A test may be incorrect  
because it has an invalid test objective,  
fails to meet its test objective, or  
contains erroneous or illegal use of the Ada  
programming language.

## CHAPTER 2

### IMPLEMENTATION DEPENDENCIES

#### 2.1 WITHDRAWN TESTS

Some tests are withdrawn by the AVO from the ACVC because they do not conform to the Ada Standard. The following 95 tests had been withdrawn by the Ada Validation Organization (AVO) at the time of validation testing. The rationale for withdrawing each test is available from either the AVO or the AVF. The publication date for this list of withdrawn tests is 91-08-02.

|         |         |         |         |         |         |
|---------|---------|---------|---------|---------|---------|
| E28005C | B28006C | C32203A | C34006D | C35508I | C35508J |
| C35508M | C35508N | C35702A | C35702B | B41308B | C43004A |
| C45114A | C45346A | C45612A | C45612B | C45612C | C45651A |
| C46022A | B49008A | B49008B | A74006A | C74308A | B83022B |
| B83022H | B83025B | B83025D | B83026B | C83026A | C83041A |
| B85001L | C86001F | C94021A | C97116A | C98003B | BA2011A |
| CB7001A | CB7001B | CB7004A | CC1223A | BC1226A | CC1226B |
| BC3009B | BD1B02B | BD1B06A | AD1B08A | BD2A02A | CD2A21E |
| CD2A23E | CD2A32A | CD2A41A | CD2A41E | CD2A87A | CD2B15C |
| BD3006A | BD4008A | CD4022A | CD4022D | CD4024B | CD4024C |
| CD4024D | CD4031A | CD4051D | CD5111A | CD7004C | ED7005D |
| CD7005E | AD7006A | CD7006E | AD7201A | AD7201E | CD7204B |
| AD7206A | BD8002A | BD8004C | CD9005A | CD9005B | CDA201E |
| CE2107I | CE2117A | CE2117B | CE2119B | CE2205B | CE2405A |
| CE3111C | CE3116A | CE3118A | CE3411B | CE3412B | CE3607B |
| CE3607C | CE3607D | CE3812A | CE3814A | CE3902B |         |

#### 2.2 INAPPLICABLE TESTS

A test is inapplicable if it contains test objectives which are irrelevant for a given Ada implementation. The inapplicability criteria for some tests are explained in documents issued by ISO and the AJPO known as Ada Commentaries and commonly referenced in the format AI-ddddd. For this implementation, the following tests were determined to be inapplicable for the reasons indicated; references to Ada Commentaries are included as appropriate.

B23003D and B23003E check that an implementation imposes a limit on the length of the input line. This implementation has no such limit (see Section 2.3).

The following 285 tests have floating-point type declarations

requiring more digits than SYSTEM.MAX\_DIGITS:

|                       |                       |
|-----------------------|-----------------------|
| C24113F..Y (20 tests) | C35705F..Y (20 tests) |
| C35706F..Y (20 tests) | C35707F..Y (20 tests) |
| C35708F..Y (20 tests) | C35802F..Z (21 tests) |
| C45241F..Y (20 tests) | C45321F..Y (20 tests) |
| C45421F..Y (20 tests) | C45521F..Z (21 tests) |
| C45524F..Z (21 tests) | C45621F..Z (21 tests) |
| C45641F..Y (20 tests) | C46012F..Z (21 tests) |

The following 21 tests check for the predefined type SHORT\_INTEGER; for this implementation, there is no such type:

|         |         |         |         |         |
|---------|---------|---------|---------|---------|
| C35404B | B36105C | C45231B | C45304B | C45411B |
| C45412B | C45502B | C45503B | C45504B | C45504E |
| C45611B | C45613B | C45614B | C45631B | C45632B |
| B52004E | C55B07B | B55B09D | B86001V | C86006D |
| CD7101E |         |         |         |         |

The following 20 tests check for the predefined type LONG\_INTEGER; for this implementation, there is no such type:

|         |         |         |         |         |
|---------|---------|---------|---------|---------|
| C35404C | C45231C | C45304C | C45411C | C45412C |
| C45502C | C45503C | C45504C | C45504F | C45611C |
| C45613C | C45614C | C45631C | C45632C | B52004D |
| C55B07A | B55B09C | B86001W | C86006C | CD7101F |

C35404D, C45231D, B86001X, C86006E, and CD7101G check for a predefined integer type with a name other than INTEGER, LONG\_INTEGER, or SHORT\_INTEGER; for this implementation, there is no such type.

C35713B, C45423B, B86001T, and C86006H check for the predefined type SHORT\_FLOAT; for this implementation, there is no such type.

C35713C, B86001U, and C86006G check for the predefined type LONG\_FLOAT; for this implementation, there is no such type.

C35713D and B86001Z check for a predefined floating-point type with a name other than FLOAT, LONG\_FLOAT, or SHORT\_FLOAT; for this implementation, there is no such type.

C45531M..P and C45532M..P (8 tests) check fixed-point operations for types that require a SYSTEM.MAX\_MANTISSA of 47 or greater; for this implementation, MAX\_MANTISSA is less than 47.

C45624A..B (2 tests) check that the proper exception is raised if MACHINE\_OVERFLOW is FALSE for floating point types and the results of various floating-point operations lie outside the range of the base type; for this implementation, MACHINE\_OVERFLOW is TRUE.

C4A013B contains a static universal real expression that exceeds the range of this implementation's largest floating-point type; this expression is rejected by the compiler.

D55A03C..H ( 6 tests) use 15 or more levels of loop nesting; this level of loop nesting exceeds the capacity of the compiler.

The 23 tests below involve separate compilation of generic declarations and their corresponding proper bodies; this implementation requires that the declarations and bodies be in the same compilation (cf LRM 10.3.9).

|            |         |         |         |         |
|------------|---------|---------|---------|---------|
| B83004B    | B83004D | B83204F | B83E01F | BA1010D |
| BA1011C    | CA2009C | CA2009F | CA1012A | CA3011A |
| LA5008A..H | LA5008J | LA5008M | LA5008N | BC3204C |
| BC3205D    |         |         |         |         |

C85005C and C85006C exceed the capacity of the implementation (see Section 2.3).

B86001Y uses the name of a predefined fixed-point type other than type DURATION; for this implementation, there is no such type.

B91001H checks that an address clause for a task entry must not precede any entry; this implementation does not support interrupts (see Section 2.3).

LA3004A..B, EA3004C..D, and CA3004E..F (6 tests) check pragma INLINE for procedures and functions; this implementation does not support pragma INLINE.

CD1009C checks whether a length clause can specify a non-default size for a floating-point type; this implementation does not support such sizes.

CD2A84A, CD2A84E, CD2A84I..J (2 tests), and CD2A84O use length clauses to specify non-default sizes for access types; this implementation does not support such sizes.

BD8001A, BD8003A, BD8004A..B (2 tests), and AD8011A use machine code insertions; this implementation provides no package MACHINE\_CODE.

BD9001A, AD9001B, ED9002A, AD9004A, and BD9004B use pragma INTERFACE; this implementation does not support the pragma.

AE2101C and EE2201D..E (2 tests) use instantiations of package SEQUENTIAL\_IO with unconstrained array types and record types with discriminants without defaults; these instantiations are rejected by this compiler.

AE2101H, EE2401D, and EE2401G use instantiations of package DIRECT\_IO with unconstrained array types and record types with discriminants without defaults; these instantiations are rejected by this compiler.

The tests listed in the following table check that USE\_ERROR is raised if the given file operations are not supported for the given combination of mode and access method; this implementation supports these operations.

| Test    | File Operation | Mode       | File Access Method |
|---------|----------------|------------|--------------------|
| CE2102D | CREATE         | IN_FILE    | SEQUENTIAL_IO      |
| CE2102E | CREATE         | OUT_FILE   | SEQUENTIAL_IO      |
| CE2102F | CREATE         | INOUT_FILE | DIRECT_IO          |
| CE2102I | CREATE         | IN_FILE    | DIRECT_IO          |
| CE2102J | CREATE         | OUT_FILE   | DIRECT_IO          |
| CE2102N | OPEN           | IN_FILE    | SEQUENTIAL_IO      |
| CE2102O | RESET          | IN_FILE    | SEQUENTIAL_IO      |
| CE2102P | OPEN           | OUT_FILE   | SEQUENTIAL_IO      |
| CE2102Q | RESET          | OUT_FILE   | SEQUENTIAL_IO      |
| CE2102R | OPEN           | INOUT_FILE | DIRECT_IO          |
| CE2102S | RESET          | INOUT_FILE | DIRECT_IO          |
| CE2102T | OPEN           | IN_FILE    | DIRECT_IO          |
| CE2102U | RESET          | IN_FILE    | DIRECT_IO          |
| CE2102V | OPEN           | OUT_FILE   | DIRECT_IO          |
| CE2102W | RESET          | OUT_FILE   | DIRECT_IO          |
| CE3102E | CREATE         | IN_FILE    | TEXT_IO            |
| CE3102F | RESET          | Any Mode   | TEXT_IO            |
| CE3102G | DELETE         | -----      | TEXT_IO            |
| CE3102I | CREATE         | OUT_FILE   | TEXT_IO            |
| CE3102J | OPEN           | IN_FILE    | TEXT_IO            |
| CE3102K | OPEN           | OUT_FILE   | TEXT_IO            |

CE2203A checks that WRITE raises USE\_ERROR if the capacity of an external sequential file is exceeded; this implementation cannot restrict file capacity.

CE2403A checks that WRITE raises USE\_ERROR if the capacity of an external direct file is exceeded; this implementation cannot restrict file capacity.

CE3304A checks that SET\_LINE\_LENGTH and SET\_PAGE\_LENGTH raise USE\_ERROR if they specify an inappropriate value for the external file; there are no inappropriate values for this implementation.

### 2.3 TEST MODIFICATIONS

Modifications (see section 1.3) were required for 109 tests.

The following tests were split into two or more tests because this implementation did not report the violations of the Ada Standard in the way expected by the original tests.

|         |         |         |         |         |
|---------|---------|---------|---------|---------|
| B22003A | B22005K | B23002A | B24001A | B24001B |
| B24001C | B24005A | B24005B | B24009A | B25002A |
| B25002B | B26001A | B26005A | B27005A | B2A021A |
| B32201A | B33301A | B36201A | B37301J | B43201A |
| B44001A | B44002A | B44004A | B44004B | B44004C |
| B44004E | B45205A | B48002A | B48002D | B51003A |
| B51003B | B51003C | B51003H | B51003I | B55A01A |
| B55A01D | B55A01E | B55B17A | B56001C | B56001H |
| B61001H | B61001I | B62001D | B63001A | B63001B |
| B64003A | B66001C | B71001C | B71001F | B71001I |
| B71001L | B71001O | B71001U | B91002A | B91002B |
| B91002C | B91002D | B91002E | B91002F | B91002G |
| B91002H | B91002I | B91002J | B91002K | B91002L |
| B91003D | B95001D | B95003A | B95004A | B95030A |
| B95032A | B95061A | B95061B | B95061C | B95061D |
| B95061E | B95061F | B95061G | B95081A | B97101A |
| B97101E | B97101G | B97101H | B97103E | B97104D |
| B97104E | B97104G | BC1202E | BC2001D | BC2001E |
| BC2004A | BC3003A | BC3003B | BC3005B | BC3205C |
| BD5005D | BE2210A | BE2413A |         |         |

B23003D and B23003E were graded inapplicable by Evaluation Modification as directed by the AVO. These tests check that an implementation imposes a limit on the length of the input line, this implementation has no such limits. The AVO ruled that this behavior is acceptable.

B23003F was graded passed by Evaluation Modification as directed by the AVO. This test checks that an identifier may not exceed the limit on the input line length. Although this implementation imposes no such limit, it does limit identifiers (and literals) to 120 characters; the AVO ruled that this behavior is acceptable, and that this test thus constitutes a check that the identifier limit is correctly enforced.

B38003A and B38009A were graded passed by Test Modification as directed by the AVO. These tests check that various illegalities are detected by the compiler. This implementation required that the illegal constraint on line 74 in B38003A and line 66 in B38009A be removed in order for subsequent errors to be detected.

B83E01F and BA1011C were graded inapplicable by Evaluation Modification as directed by the AVO. These tests expect that the bodies of generic units can be compiled separately from their declarations. This implementation requires that generic declarations and bodies be in the same compilation; this restriction is allowed by LRM 10.3.9.

C85005C and C85006C were graded inapplicable by Evaluation Modification as directed by the AVO. These tests contain a combination of tasks and data structures that exceeds this implementation's capacity.

B91001H was graded inapplicable by Evaluation Modification as directed by the AVO. This test checks that an address clause for an entry cannot proceed that or any other entry of the task. This implementation does not support interrupts, and so rejects any address clause for an entry, regardless of placement.

BA3001A were graded inapplicable by Evaluation Modification as directed by the AVO. This test contains a generic subprogram declaration with no corresponding body. This implementation requires that generic declarations and bodies be in the same compilation, therefore it detected the absence of a subprogram body as an additional error. The AVO ruled that the additional error message may be ignored.



## CHAPTER 3

### PROCESSING INFORMATION

#### 3.1 TESTING ENVIRONMENT

The Ada implementation tested in this validation effort is described adequately by the information given in the initial pages of this report.

For technical information about this Ada implementation, contact:

Mr. Li Xin  
Green Valley Software  
Building 13, Wanmingyuan  
Fuchengamenwai, Beijing  
China 100037  
Telephone: 86-1-8313399-3307  
FAX: 86-1-8342707

For sales information about this Ada implementation, contact:

Ms. Li Jiangyue  
Green Valley Software  
Building 13, Wanmingyuan  
Fuchengamenwai, Beijing  
China 100037  
Telephone: 86-1-8313399-3308  
FAX: 86-1-8342707

Testing of this Ada implementation was conducted at the customer's site by a validation team from the AVF.

#### 3.2 SUMMARY OF TEST RESULTS

An Ada Implementation passes a given ACVC version if it processes each test of the customized test suite in accordance with the Ada Programming Language Standard, whether the test is applicable or inapplicable; otherwise, the Ada Implementation fails the ACVC [Pro92].

For all processed tests (inapplicable and applicable), a result was obtained that conforms to the Ada Programming Language Standard.

The list of items below gives the number of ACVC tests in various categories. All tests were processed, except those that were withdrawn because of test errors (item b; see section 2.1), those that require a floating-point precision that exceeds the

implementation's maximum precision (item e; see section 2.2), and those that depend on the support of a file system--if none is supported (item d). All tests passed, except those that are listed in sections 2.1 and 2.2 (counted in items b and f, below).

|   |      |         |
|---|------|---------|
| a) Total Number of Applicable Tests             | 3637 |         |
| b) Total Number of Withdrawn Tests              | 95   |         |
| c) Processed Inapplicable Tests                 | 438  |         |
| d) Non-Processed I/O Tests                      | 0    |         |
| e) Non-Processed Floating-Point Precision Tests | 0    |         |
| f) Total Number of Inapplicable Tests           | 438  | (c+d+e) |
| g) Total Number of Tests for ACVC 1.11          | 4170 | (a+b+f) |

### 3.3 TEST EXECUTION

A magnetic tape containing the customized test suite (see section 1.3) was taken on-site by the validation team for processing. The contents of the magnetic tape were loaded directly onto the host computer.

After the test files were loaded onto the host computer, the full set of tests was processed by the Ada implementation.

The tests were compiled, linked, executed and captured on the host/target computer.

Testing was performed using command scripts provided by the customer and reviewed by the validation team. See Appendix B for a complete listing of the processing options for this implementation. It also indicates the default options. The options invoked explicitly for validation testing during this test were:

-O name Name of executable main program file. Runada is the default value.

-t number Set size of task stack as number. 3777 is the default value.

Test output, compiler and linker listings, and job logs were captured on magnetic tape and archived at the AVF. The listings examined on-site by the validation team were also archived.

## APPENDIX A

### MACRO PARAMETERS

This appendix contains the macro parameters used for customizing the ACVC. The meaning and purpose of these parameters are explained in [UG89]. The parameter values are presented in two tables. The first table lists the values that are defined in terms of the maximum input-line length, which is the value for \$MAX\_IN\_LEN--also listed here. These values are expressed here as Ada string aggregates, where "V" represents the maximum input-line length.

| Macro Parameter              | Macro Value                                 |
|------------------------------|---|
| \$MAX_IN_LEN                 | 120            -- Value of V                |
| \$BIG_ID1                    | (1..V-1 => 'A', V => '1')                   |
| \$BIG_ID2                    | (1..V-1 => 'A', V => '2')                   |
| \$BIG_ID3                    | (1..V/2 => 'A') & '3' & (1..V-1-V/2 => 'A') |
| \$BIG_ID4                    | (1..V/2 => 'A') & '4' & (1..V-1-V/2 => 'A') |
| \$BIG_INT_LIT                | (1..V-3 => '0') & "298"                     |
| \$BIG_REAL_LIT               | (1..V-5 => '0') & "690.0"                   |
| \$BIG_STRING1                | "" & (1..V/2 => 'A') & ""                   |
| \$BIG_STRING2                | "" & (1..V-1-V/2 => 'A') & '1' & ""         |
| \$BLANKS                     | (1..V-20 => ' ')                            |
| \$MAX_LEN_INT_BASED_LITERAL  | "2:" & (1..V-5 => '0') & "11:"              |
| \$MAX_LEN_REAL_BASED_LITERAL | "16:" & (1..V-7 => '0') & "F.E:"            |
| \$MAX_STRING_LITERAL         | "" & (1..V-2 => 'A') & ""                   |

The following table contains the values for the remaining macro parameters.

| Macro Parameter                       | Macro Value                     |
|---------------------------------------|---------------------------------|
| \$ACC_SIZE                            | 32                              |
| \$ALIGNMENT                           | 4                               |
| \$COUNT_LAST                          | 2048                            |
| \$DEFAULT_MEM_SIZE                    | 5242880                         |
| \$DEFAULT_STOR_UNIT                   | 8                               |
| \$DEFAULT_SYS_NAME                    | C_Ada                           |
| \$DELTA_DOC                           | 2.0**(-31)                      |
| \$ENTRY_ADDRESS                       | 0                               |
| \$ENTRY_ADDRESS1                      | 0                               |
| \$ENTRY_ADDRESS2                      | 0                               |
| \$FIELD_LAST                          | 2147483647                      |
| \$FILE_TERMINATOR                     | STANDARD.ASCII.FS               |
| \$FIXED_NAME                          | NO_SUCH_TYPE                    |
| \$FLOAT_NAME                          | NO_SUCH_TYPE                    |
| \$FORM_STRING                         | ""                              |
| \$FORM_STRING2                        | "CANNOT RESTRICT FILE CAPACITY" |
| \$GREATER_THAN_DURATION               | 86400.01                        |
| \$GREATER_THAN_DURATION_BASE_LAST     | 1.4E05                          |
| \$GREATER_THAN_FLOAT_BASE_LAST        | 16#0.7FFF_FFFF_FFFF_FF8#E33     |
| \$GREATER_THAN_FLOAT_SAFE_LARGE       | 16#0.7FFF_FFFF_8#E32            |
| \$GREATER_THAN_SHORT_FLOAT_SAFE_LARGE |                                 |

|                                 |                                 |
|---------------------------------|---------------------------------|
|                                 | DO NOT SUPPORT SHORT_FLOAT      |
| \$HIGH_PRIORITY                 | 16                              |
| \$ILLEGAL_EXTERNAL_FILE_NAME1   | /NODIRECTORY/NONAME             |
| \$ILLEGAL_EXTERNAL_FILE_NAME2   | THIS_FILE_NAME_IS_TOO_LONG      |
| \$INAPPROPRIATE_LINE_LENGTH     | -1                              |
| \$INAPPROPRIATE_PAGE_LENGTH     | -1                              |
| \$INCLUDE_PRAGMA1               | PRAGMA INCLUDE ("A28006D1.TST") |
| \$INCLUDE_PRAGMA2               | PRAGMA INCLUDE ("B28006D1.TST") |
| \$INTEGER_FIRST                 | -2147483648                     |
| \$INTEGER_LAST                  | 2147483647                      |
| \$INTEGER_LAST_PLUS_1           | 2147483648                      |
| \$INTERFACE_LANGUAGE            | NO_LANGUAGE                     |
| \$LESS_THAN_DURATION            | -86400.01                       |
| \$LESS_THAN_DURATION_BASE_FIRST | -1.4E05                         |
| \$LINE_TERMINATOR               | STANDARD.ASCII.LF               |
| \$LOW_PRIORITY                  | 1                               |
| \$MACHINE_CODE_STATEMENT        | NULL;                           |
| \$MACHINE_CODE_TYPE             | NO_SUCH_TYPE                    |
| \$MANTISSA_DOC                  | 31                              |
| \$MAX_DIGITS                    | 9                               |
| \$MAX_INT                       | 2147483647                      |
| \$MAX_INT_PLUS_1                | 2147483648                      |
| \$MIN_INT                       | -2147483648                     |
| \$NAME                          | NO_SUCH_TYPE_AVAILABLE          |
| \$NAME_LIST                     | C_Ada                           |

|                       |                           |
|-----------------------|---------------------------|
| \$NAME_SPECIFICATION1 | X2120A                    |
| \$NAME_SPECIFICATION2 | X2120B                    |
| \$NAME_SPECIFICATION3 | X3119A                    |
| \$NEG_BASED_INT       | 16#FFFF_FFFF#             |
| \$NEW_MEM_SIZE        | 5242880                   |
| \$NEW_STOR_UNIT       | 8                         |
| \$NEW_SYS_NAME        | C_Ada                     |
| \$PAGE_TERMINATOR     | STANDARD.ASCII.FF         |
| \$RECORD_DEFINITION   | NEW_INTEGER               |
| \$RECORD_NAME         | NO_SUCH_MACHINE_CODE_TYPE |
| \$TASK_SIZE           | 32                        |
| \$TASK_STORAGE_SIZE   | 3770                      |
| \$TICK                | 0.01                      |
| \$VARIABLE_ADDRESS    | 32                        |
| \$VARIABLE_ADDRESS1   | 64                        |
| \$VARIABLE_ADDRESS2   | 128                       |
| \$YOUR_PRAGMA         | NO_SUCH_PRAGMA            |

## APPENDIX B

### COMPILATION SYSTEM OPTIONS

**-O name** Name of executable main program file. Runada is the default value.

**-t number** Set size of task stack as number. 3777 is the default value.

### LINKER OPTIONS

The linker options of this Ada implementation, as described in this Appendix, are provided by the customer. Unless specifically noted otherwise, references in this appendix are to linker documentation and not to this report.

**-n number** Maximum number of tasks in an Ada program. 18 is the default value.

**-e number** Maximum number of entries in a task. 15 is the default value.

APPENDIX C

APPENDIX F OF THE Ada STANDARD

The only allowed implementation dependencies correspond to implementation-dependent pragmas, to certain machine-dependent conventions as mentioned in Chapter 13 of the Ada Standard, and to certain allowed restrictions on representation clauses. The implementation-dependent characteristics of this Ada implementation, as described in this Appendix, are provided by the customer. Unless specifically noted otherwise, references in this Appendix are to compiler documentation and not to this report. Implementation-specific portions of the package STANDARD, which are not a part of Appendix F, are:

package STANDARD is

type INTEGER is range -2147483648 .. 2147483647;

type FLOAT is digits 9 range  
-16:0.FFFF\_FFFF:E31 .. 16:0.FFFF\_FFFF:E31;

type DURATION is delta 0.01 range -86400.00 .. 86400.00;

.....  
end STANDARD;



## Appendix F of Language Reference Manual for C\_Ada v1.1

### F.1 Implementation-Dependent Pragmas

C\_ada v1.1 does not support the predefined language pragmas: `INLINE` and `SUPPRESS`.

See Annex B for a descriptive pragma summary.

### F.2 Implementation-Dependent Attributes

C\_ada v1.1 does not have any Implementation-Dependent Attributes.

### F.3 Specification of the Package SYSTEM

package SYSTEM is

```
type NAME is (C_ADA);
```

```
SYSTEM_NAME      : constant NAME := C_ADA;
STORAGE_UNIT     : constant := 8;
MEMORY_SIZE      : constant := 5*2**20;
MAX_INT          : constant := 2**31 - 1;
MIN_INT          : constant := -(2**31);
MAX_DIGITS       : constant := 9;
MAX_MANTISSA     : constant := 31;
FINE_DELTA       : constant := 2**(-31);
TICK             : constant := 0.01;
subtype PRIORITY is INTEGER range 1 .. 16;
```

```
end SYSTEM;
```

### F.4 Restrictions on representation clause

The representation clause allowing in C\_Ada v1.1 are length, enumeration and record representation clause; C\_Ada v1.1 support address clause for object declared by an object declaration only. C\_Ada v1.1 does not support interrupt and machine code insertion.

In C\_Ada v1.1, a representation clause for a generic formal type or a type depends on a generic formal type is not allowed. In addition, a representation clause for a composite type that has a component or a subcomponent of a generic formal type or a type derived from a generic formal type is not allowed.

### F.5 Conventions For Implementation-Generated Names

C\_Ada v1.1 does not allocate implementation dependent component in record.

## F.6 Interpretation of Expression Appearing In Address Clause

It should be static.

## F.7 Restrictions On Unchecked Type Conversions

C\_Ada v1.1 supports the generic function UNCHECKED\_CONVERSION with the restriction given in section 13.10.2.

## F.8 Implementation-Dependent Characteristics of Input-Output Packages

### F.8.1 Restrictions On Input-Output Packages

C\_Ada v1.1 does not provide the package LOW\_LEVEL\_IO.

### F.8.2 Interpretation Of the FORM Parameter

Parameter FORM is used to provide access right to other users in the system.

The parameter is a string whose interpretation is a sexadecimal literal. The meaning and the syntax are the same as that in Unix. Access right is defined by 3 successive sexadecimal digits, they stand for the access right of user himself, of the users who are in the same group as the owner, and of others.

### F.8.3 Implementation-Dependent Input-Output Error Conditions

C\_Ada v1.1 has no exceptions other than that defined in package IO\_EXCEPTIONS.