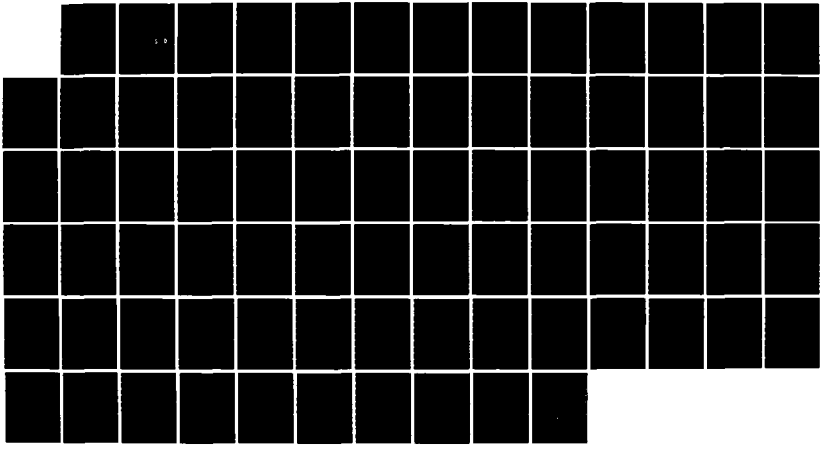
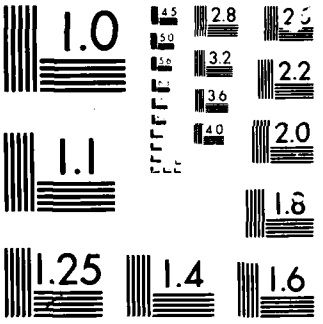
AD-A167 545 ADA (TRADE NAME) COMPILER VALIDATION SUMMARY REPORT: 1/1
VERDIX CORPORATION V.. (U) INFORMATION SYSTEMS AND
TECHNOLOGY CENTER W-P AFB OH ADA VALI.. 16 NOV 86
UNCLASSIFIED AVF-VSR-17.0286 F/G 13/2 NL





MICROCOPY CHART

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(2)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1 REPORT NUMBER / 2 GOVT ACCESSION NO.	3 RECIPIENT'S CATALOG NUMBER
4 TITLE (and Subtitle) Ada Compiler Validation Summary Report: Verdix Ada Development System, Version 5.2 for the CCI Power 6/32 under Power 6 UNIX, Release 1.11	5 TYPE OF REPORT & PERIOD COVERED 16 Nov. 1985 to 16 Nov. 1986 6. PERFORMING ORG. REPORT NUMBER AVF-VSR-17.0286
7. AUTHOR(s) Ada Validation Facility	8. CONTRACT OR GRANT NUMBER(s)
9 PERFORMING ORGANIZATION NAME AND ADDRESS Ada Validation Facility Computer Operations Division Intel Systems and Technology Center, WP AFB, OH 45433	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11 CONTROLLING OFFICE NAME AND ADDRESS Ada Joint Program Office 1211 S. Fern St., Rm. C-107 Arlington, VA 22202	12. REPORT DATE 16 November 1986 13 NUMBER OF PAGES 76
14 MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Ada Validation Facility	15 SECURITY CLASS. (of this report) UNCLASSIFIED 15a DECLASSIFICATION/DOWNGRADING SCHEDULE

16 DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

DTIC ELECTE MAY 01 1986 S D

17 DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

UNCLASSIFIED

18 SUPPLEMENTARY NOTES

DTIC FILE COPY

19 KEY WORDS (Continue on reverse side if necessary and identify by block number)

Ada Programming language, Ada Compiler Validation Summary Report, Ada Compiler Validation Capability, ACVC, Validation Testing, Ada Validation Office, AVO, Ada Validation Facility, AVF, ANSI/MIL-STD-1815A, Ada Joint Program Office, AJPO.

20 ABSTRACT (Continue on reverse side if necessary and identify by block number)

See attached.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

86 4 29 123

AD-A167 545

ABSTRACT

The Validation Summary Report presents the results and conclusions of testing performed on the Verdix Ada Development System (VADS), Version 5.2. Standardized tests serve as input to an Ada® compiler, producing results which are evaluated by the validation team. This summary briefly states the highlights of the VADS, Version 5.2, validation.

On-site testing was performed on 16 NOV 1985 at Verdix Corporation in Aloha, Oregon under the auspices of the Ada Validation Facility (AVF), according to Ada Validation Office (AVO) policies and procedures. The VADS, Version 5.2, is hosted on a CCI Power 6/32 operating under Power 6 UNIX Operating System, Release 1.11. The suite of tests known as the Ada Compiler Validation Capability (ACVC), Version 1.6, was used. The ACVC is used to validate conformance of a compiler to ANSI/MIL-STD-1815A Ada. The purpose of testing is to ensure that a compiler properly implements legal language constructs and that it identifies and rejects illegal language constructs. The testing also identifies behavior that is implementation dependent but permitted by the Ada Standard. Six classes of tests are used. These tests are designed to perform checks at compile time, at link time, or during execution.

The results of validation are summarized in the following table.

RESULT	TEST CLASS						TOTAL
	A	B	C	D	E	L	
Passed	61	777	999	17	8	3	1865
Failed	0	0	0	0	0	0	0
Inapplicable	0	4	227	0	0	0	231
Anomalous	0	0	0	0	0	0	0
Withdrawn	0	19	47	0	0	0	66
TOTAL	61	800	1273	17	8	3	2162

®Ada is a registered trademark of the United States Government (Ada Joint Program Office).

For
CRA&I
TAB
ced
ion
ion/
Availability Codes
Dist | Avail and/or Special
A-1

AVF-VSR-17.0286

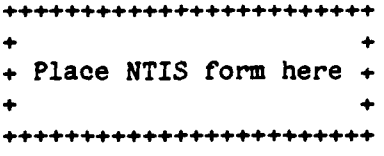
Ada® COMPILER VALIDATION SUMMARY REPORT:
Verdix Corporation
Verdix Ada Development System, Version 5.2
for the
CCI Power 6/32
under
Power 6 UNIX, Release 1.11

Completion of On-Site Validation:
16 NOV 1985

Prepared By:
Ada Validation Facility
Information Systems & Technology Center
ASD/SIOL
Wright-Patterson AFB OH 45433-6503

Prepared For:
Ada Joint Program Office
United States Department of Defense
Washington, D.C.

®Ada is a registered trademark of the United States Government (Ada Joint Program Office).


Place NTIS form here

Ada® Compiler Validation Summary Report:

Compiler Name: Verdix Ada Development System (VADS), Version 5.2

Host Computer:	Target Computer:
CCI Power 6/32	CCI Power 6/32
under	under
Power 6 UNIX, Release 1.11	Power 6 UNIX, Release 1.11

Testing Completed 16 NOV 1985 Using ACVC 1.6

This report has been reviewed and approved:

Georgeanne Chitwood

Ada Validation Facility
Georgeanne Chitwood
Information Systems & Technology Center
ASD/SIOL
Wright-Patterson AFB OH 45433-6503

John F. Kramer

Ada Validation Office (AVO)
Dr. John F. Kramer
Institute for Defense Analyses
Alexandria, VA

Virginia L. Castor

Ada Joint Program Office (AJPO)
Virginia L. Castor
Director
Washington, D.C.

®Ada is a registered trademark of the United States Government (Ada Joint Program Office).

EXECUTIVE SUMMARY

The Validation Summary Report presents the results and conclusions of testing performed on the Verdix Ada Development System (VADS), Version 5.2. Standardized tests serve as input to an Ada® compiler, producing results which are evaluated by the validation team. This summary briefly states the highlights of the VADS, Version 5.2, validation.

On-site testing was performed on 16 NOV 1985 at Verdix Corporation in Aloha, Oregon under the auspices of the Ada Validation Facility (AVF), according to Ada Validation Office (AVO) policies and procedures. The VADS, Version 5.2, is hosted on a CCI Power 6/32 operating under Power 6 UNIX Operating System, Release 1.11. The suite of tests known as the Ada Compiler Validation Capability (ACVC), Version 1.6, was used. The ACVC is used to validate conformance of a compiler to ANSI/MIL-STD-1815A Ada. The purpose of testing is to ensure that a compiler properly implements legal language constructs and that it identifies and rejects illegal language constructs. The testing also identifies behavior that is implementation dependent but permitted by the Ada Standard. Six classes of tests are used. These tests are designed to perform checks at compile time, at link time, or during execution.

The results of validation are summarized in the following table.

RESULT	TEST CLASS						TOTAL
	A	B	C	D	E	L	
Passed	61	777	999	17	8	3	1865
Failed	0	0	0	0	0	0	0
Inapplicable	0	4	227	0	0	0	231
Anomalous	0	0	0	0	0	0	0
Withdrawn	0	19	47	0	0	0	66
TOTAL	61	800	1273	17	8	3	2162

®Ada is a registered trademark of the United States Government (Ada Joint Program Office).

- The package SYSTEM is used by package TEXT_IO:

  C86001F-B.DEP

- The 'SIZE clause is supported:

  C87B62A-B.DEP

- The 'STORAGE_SIZE clause is supported:

  C87B62B-B.DEP

- The 'SMALL clause is supported:

  C87B62C-B.DEP

- Generic subroutine declarations and bodies can be compiled in separate compilation units:

  CA1012A-B.DEP

- Pragma INLINE is supported for procedures and functions:

  LA3004A-AB.ADA LA3004B-AB.ADA

- RESET and DELETE are supported for sequential and direct temporary files:

  CE2102G-B.ADA

- Dynamic creation, resetting, and deletion of files are supported for sequential, direct, and text I/O:

  CE2106A-B.ADA CE2210A-B.ADA CE3110A-B.ADA

- More than one internal file can be associated with the same external file:

  CE2107A-B.ADA CE2107B-B.ADA CE2107C-B.ADA
  CE2107D-B.ADA CE2107E-B.ADA CE2111D-B.ADA
  CE3111A-B.ADA CE3111B-B.ADA CE3111C-B.ADA
  CE3114B-B.ADA

- Instantiation of package SEQUENTIAL_IO with unconstrained array types and record types with discriminants is supported:

  CE2201D-B.DEP CE2201E-B.DEP

- Instantiation of package DIRECT_IO with unconstrained array types and unconstrained types with discriminants is supported:

  CE2401D-B.DEP

- An external file associated with more than one internal file can be reset:

  CE3115A-B.ADA

- Illegal filenames can exist:

  CE2102C-B.TST

ACVC Version 1.6 was taken on-site via magnetic tape to Verdix Corporation in Aloha, Oregon. The tape was loaded, and all tests, except the withdrawn tests and any executable tests which make use of a floating-point precision greater than SYSTEM.MAX_DIGITS, were compiled on a CCI Power 6/32. Class A, C, D, and E tests were executed on a CCI Power 6/32.

On completion of testing, all results were analyzed for failed Class A, C, D, or E programs, and all Class B and L compilation results were individually analyzed.

The ACVC, Version 1.6, contains 2162 tests of which 1865 were applicable to the VADS, Version 5.2. No anomalies were found in the testing of this compiler. Testing demonstrated that all applicable tests were passed by this compiler and conformed to the Ada Standard. The AVF concluded that the results show acceptable compliance to ANSI/MIL-STD-1815A Ada.

CHAPTER 1

INTRODUCTION

The Validation Summary Report describes how an Ada compiler conforms to the language standard. This report explains all technical terms used within it and thoroughly reports the Ada Compiler Validation Capability (ACVC) test results. Ada compilers must be written according to the language specification as given in ANSI/MIL-STD-1815A Ada. All implementation-defined features must be included for the compiler to conform to the Standard. Following the guidelines of the Standard ensures continuity between compilers. That is, the entire Standard must be implemented, and nothing can be implemented that is not in the Standard.

Even though all validated Ada compilers conform to the Standard, it must be understood that some differences do exist between implementations. ANSI/MIL-STD-1815A permits some implementation dependencies--e.g., the maximum length of identifiers, the maximum values of integer types, etc. These implementation-dependent features limit the portability of programs between compilers. Other differences between compilers are due to limitations imposed on a compiler by the operating system and by the hardware. All of these dependencies are given in the report.

Validation Summary Reports are written according to a standardized format. Compiler users can, therefore, more easily compare the reports from several compilers when selecting a compiler for a given task. The validation report can be completed mostly from the test results produced during validation testing. Additional testing information is given at the end of the report and states problems and details which are unique for a specific compiler. The format of the validation report limits variance between reports, enhances readability of the report, and accelerates report readiness.

INTRODUCTION

1.1 PURPOSE OF THIS VALIDATION SUMMARY REPORT

The Validation Summary Report documents the results of the testing performed on an Ada compiler. Testing was carried out for the following purposes:

- To identify any language constructs supported by the translator that do not conform to the Ada Standard
- To identify any unsupported language constructs required by the Ada Standard
- To describe the implementation-dependent behavior allowed by the Ada Standard

Testing of this compiler was conducted by SofTech, Inc. under the supervision of the Ada Validation Facility (AVF) according to policies and procedures established by the Ada Validation Office (AVO). Testing was conducted on 16 NOV 1985 at Verdix Corporation in Aloha, Oregon.

1.2 USE OF THIS VALIDATION SUMMARY REPORT

Consistent with the national laws of the originating country, the AVO 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 any statement or statements set forth in this report are accurate or complete, or that the subject compiler has no nonconformances to the Ada Standard other than those presented. This report is not intended for the purpose of publicizing the findings summarized herein.

Questions regarding this report or the validation tests should be directed to:

Ada Validation Office
Institute for Defense Analyses
1801 N. Beauregard
Alexandria VA 22311

and to:

Ada Validation Facility
Information Systems & Technology Center
ASD/SIOL
Wright-Patterson AFB OH 45433-6503

1.3 REFERENCES

1. Reference Manual for the Ada Programming Language, ANSI/MIL-STD-1815A, Feb 1983.

2. Ada Validation Organization: Policies and Procedures, Mitre Corporation, June 1982, PB 83-110601.

3. Ada Compiler Validation Capability Implementers' Guide, SofTech, Inc., Dec 1984.

1.4 DEFINITION OF TERMS

Anomaly	A test result that, given pre-validation analysis, is not expected during formal validation but is judged allowable under the circumstances.
ACVC	The Ada Compiler Validation Capability. A set of programs that evaluates the conformance of a compiler to the Ada language specification, ANSI/MIL-STD-1815A.
Ada Standard	ANSI/MIL-STD-1815A, February 1983.
Applicant	The agency requesting validation.
AVF	The Ada Validation Facility. In the context of this report, the AVF is responsible for conducting compiler validations according to established policies and procedures.
AVO	The Ada Validation Office. In the context of this report, the AVO is responsible for setting policies and procedures for compiler validations.
Compiler	A processor for the Ada language. In the context of this report, a compiler is any language processor, including cross-compilers, translators, and interpreters.
Failed test	A test for which the compiler generates a result that demonstrates nonconformance to the Ada Standard.
Host	The computer on which the compiler resides.
Inapplicable test	A test that uses features of the language that a compiler is not required to support or may legitimately support in a way other than the one expected by the test.
Passed test	A test for which a compiler generates the expected result.

Target	The computer for which a compiler generates code.
Test	A program that evaluates the conformance of a compiler to a language specification. In the context of this report, the term is used to designate a single ACVC test. The text of a program may be the text of one or more compilations.
Withdrawn test	A withdrawn test has an invalid test objective, fails to meet its test objective, or contains illegal use of the language.

1.5 CONFIGURATION

The candidate compilation system for this validation was tested under the configuration:

Compiler: Verdix Ada Development System (VADS), Version 5.2

Test Suite: Ada Compiler Validation Capability (ACVC), Version 1.6

Host Computer:

Machine(s):	CCI Power 6/32
Operating System:	Power 6 UNIX Operating System, Release 1.11
Memory Size:	512 Megabytes

Target Computer:

Machine(s):	CCI Power 6/32
Operating System:	Power 6 UNIX Operating System, Release 1.11
Memory Size:	512 Megabytes

CHAPTER 2

TEST RESULTS

2.1 ACVC TEST CLASSES

Conformance to ANSI/MIL-STD-1815A is measured using the ACVC. The ACVC contains both legal and illegal Ada programs structured into six test classes: A, B, C, D, E, and L. Legal programs are compiled and executed while illegal programs are just compiled. Support packages are used to report the results of the legal programs. A compiler must correctly process each of the tests in the suite and demonstrate conformance to the Ada Standard by either meeting the pass criteria given for the test or by showing that the test is inapplicable to the implementation. Tests that are found to contain errors are withdrawn from the ACVC. Detailed test results are listed in Appendix D. The results of validation testing are summarized in the following table:

RESULT	TEST CLASS						TOTAL
	A	B	C	D	E	L	
Passed	61	777	999	17	8	3	1865
Failed	0	0	0	0	0	0	0
Inapplicable	0	4	227	0	0	0	231
Anomalous	0	0	0	0	0	0	0
Withdrawn	0	19	47	0	0	0	66
TOTAL	61	800	1273	17	8	3	2162

A total of 1875 tests were processed during this validation attempt. The 66 withdrawn tests in Version 1.6 were not processed, nor were 221 Class C tests that were inapplicable because they use floating-point types having digits that exceed the maximum value for the implementation. All other tests were processed.

Some conventions are followed in the ACVC to ensure that the tests are reasonably portable without modification. For example, the tests make use of only the basic 55 character set, contain lines with a maximum length of 72 characters, use small numeric values, and place features that may not be supported in separate tests. However, some tests contain values that require the test to be customized according to implementation-specific values. The values used for this validation are listed in Appendix B.

2.1.1 Class A Tests

Class A tests check that legal Ada programs can be successfully compiled and executed. However, no checks are performed during execution to see if the test objective has been met. For example, a Class A test checks that reserved words of another language (other than those already reserved in the Ada language) are not treated as reserved words by an Ada compiler. A Class A test is passed if no errors are detected at compile time and the program executes to produce a message indicating that it has passed. If a Class A test cannot be compiled and executed because of its size, then the test is split into a set of smaller subtests that can be processed.

The following table shows that all applicable Class A tests passed:

RESULT	CHAPTER												
	2	3	4	5	6	7	8	9	10	11	12	14	TOTAL
Passed	13	6	0	5	2	12	13	3	0	0	0	7	61
Failed	0	0	0	0	0	0	0	0	0	0	0	0	0
Inapplicable	0	0	0	0	0	0	0	0	0	0	0	0	0
Anomalous	0	0	0	0	0	0	0	0	0	0	0	0	0
Withdrawn	0	0	0	0	0	0	0	0	0	0	0	0	0
TOTAL	13	6	0	5	2	12	13	3	0	0	0	7	61

2.1.2 Class B Tests

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 manually to verify that every syntax or semantic error in the test is detected. A Class B test is passed if every illegal construct that it contains is detected by the compiler. If one or more errors are not detected, then a version of the test is created that contains only the undetected errors. The resulting "split" is compiled and examined. The splitting process continues until all errors are detected by the compiler. Splits were required for 15 tests:

B24104A.ADA	B24104B.ADA	B24104C.ADA
B33004A.ADA	B37201A.ADA	B38008A-B.ADA
B44001A-B.ADA	B64001A-B.ADA	B67001B-B.ADA
B67001C-B.ADA	B67001D-B.ADA	B910ABA-B.ADA
B95001A.ADA	B97101E-AB.ADA	B97102A-AB.ADA

The following table shows that all applicable Class B tests passed:

RESULT	CHAPTER												
	2	3	4	5	6	7	8	9	10	11	12	14	TOTAL
Passed	35	72	83	111	70	55	50	92	35	8	148	18	777
Failed	0	0	0	0	0	0	0	0	0	0	0	0	0
Inapplicable	0	0	0	2	0	0	2	0	0	0	0	0	4
Anomalous	0	0	0	0	0	0	0	0	0	0	0	0	0
Withdrawn	0	1	0	0	3	2	0	0	1	0	12	0	19
TOTAL	35	73	83	113	73	57	52	92	36	8	160	18	800

2.1.3 Class C Tests

Class C tests check that legal Ada programs can be correctly compiled and executed. Each Class C test is self-checking and produces a PASS/FAIL message indicating the result when it is executed. If a Class C test cannot be compiled because it exceeds the compiler's capacity, then the test is split into smaller subtests until all are compiled and executed.

The following table shows that all applicable Class C tests passed:

RESULT	CHAPTER												
	2	3	4	5	6	7	8	9	10	11	12	14	TOTAL
Passed	22	104	168	118	70	14	96	104	36	20	55	192	999
Failed	0	0	0	0	0	0	0	0	0	0	0	0	0
Inapplicable	20	103	101	1	0	0	1	1	0	0	0	0	227
Anomalous	0	0	0	0	0	0	0	0	0	0	0	0	0
Withdrawn	0	1	27	0	4	0	0	4	7	0	0	4	47
TOTAL	42	208	296	119	74	14	97	109	43	20	55	196	1273

2.1.4 Class D Tests

Class D tests check the compilation and execution capacities of a compiler. Since there are no requirements placed on a compiler by the Ada Standard for the number of identifiers permitted in a compilation, the number of units in a library, the number of nested loops in a subprogram body, and so on, a compiler may refuse to compile a Class D test. Each Class D test is self-checking and produces a PASS/FAIL message indicating the result when it is executed. If a Class D test fails to compile because the capacity of the compiler is exceeded, then the test is classified as inapplicable.

The following table shows that all applicable Class D tests passed:

RESULT	CHAPTER												
	2	3	4	5	6	7	8	9	10	11	12	14	TOTAL
Passed	1	0	4	9	3	0	0	0	0	0	0	0	17
Failed	0	0	0	0	0	0	0	0	0	0	0	0	0
Inapplicable	0	0	0	0	0	0	0	0	0	0	0	0	0
Anomalous	0	0	0	0	0	0	0	0	0	0	0	0	0
Withdrawn	0	0	0	0	0	0	0	0	0	0	0	0	0
TOTAL	1	0	4	9	3	0	0	0	0	0	0	0	17

Capacities measured by the Class D tests are detailed in section 2.4, Implementation Characteristics.

2.1.5 Class E Tests

Class E tests provide information about the compiler in those areas in which the Ada Standard permits implementations to differ. Each Class E test is executable and produces messages that indicate how the Ada Standard is interpreted. However, in some cases the Ada Standard permits a compiler to detect a condition either at compile time or at execution time, and thus a Class E test may correctly fail to execute. A Class E test is passed if it fails to compile and appropriate error messages are issued, or if it executes properly and produces a message that it has passed. If a Class E test cannot be compiled and executed because of its size, then the test is split into a set of smaller subtests that can be processed.

The following table shows that all applicable Class E tests passed:

RESULT	CHAPTER												
	2	3	4	5	6	7	8	9	10	11	12	14	TOTAL
Passed	1	3	2	1	0	0	0	0	0	0	0	1	8
Failed	0	0	0	0	0	0	0	0	0	0	0	0	0
Inapplicable	0	0	0	0	0	0	0	0	0	0	0	0	0
Anomalous	0	0	0	0	0	0	0	0	0	0	0	0	0
Withdrawn	0	0	0	0	0	0	0	0	0	0	0	0	0
TOTAL	1	3	2	1	0	0	0	0	0	0	0	1	8

Information obtained from the Class E tests is detailed in section 2.4, Implementation Characteristics.

2.1.6 Class L Tests

Class L tests check that incomplete or illegal Ada programs involving multiple, separately compiled units are detected and not allowed to execute. Class L tests are compiled separately and execution is attempted. A Class L test passes if it is rejected at link time and the test does not execute.

The following table shows that all applicable Class L tests passed:

RESULT	CHAPTER												
	2	3	4	5	6	7	8	9	10	11	12	14	TOTAL
Passed	0	0	0	0	0	0	0	0	3	0	0	0	3
Failed	0	0	0	0	0	0	0	0	0	0	0	0	0
Inapplicable	0	0	0	0	0	0	0	0	0	0	0	0	0
Anomalous	0	0	0	0	0	0	0	0	0	0	0	0	0
Withdrawn	0	0	0	0	0	0	0	0	0	0	0	0	0
TOTAL	0	0	0	0	0	0	0	0	3	0	0	0	3

2.1.7 Support Units

Three units (REPORT package, CHECK_FILE procedure, and VAR_STRINGS package) support the self-checking features of the executable tests. The REPORT package provides the mechanism by which executable tests report results. It also provides a set of identity functions used to defeat some compiler optimization strategies that cause computations to be made by the target computer instead of by the compiler on the host computer. The CHECK_FILE procedure 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 VAR_STRINGS package defines types and subprograms for manipulating varying-length character strings. The operation of these three units is checked by a set of executable tests. These tests produce messages that are examined manually to verify that the units are operating correctly. If these units are not operating correctly, then the validation is not attempted.

All support units were compiled and were demonstrated to be operating correctly.

2.2 WITHDRAWN TESTS

Some tests are withdrawn from the ACVC because they do not conform to the Ada Standard or because they use language features whose semantics are still being clarified by the Ada Language Maintenance Committee (LMC). When testing was performed, the following 66 tests had been withdrawn for the reasons indicated below. In those cases under consideration by the LMC, a reference is given to an Ada Commentary --e.g., AI-00313.

- C35904A-B: The elaborations of the subtype declarations for SFX3 and SFX4 in this test raise NUMERIC_ERROR in some implementations. The exception is raised on the conversion of the real literals 2.0 and 5.0 to the base type of FIX.

- B38105B-AB, C48006B-B, B74207A-B, and BC3503A-B: These tests require a specific interpretation of the Reference Manual regarding whether an incomplete type can have discriminant constraints before the full type declaration; this interpretation is not fully supported by the Reference Manual or the Language Maintenance Committee.

- C45521A-B through C45521Y-B (25 tests): Cases C and I define the model interval for the result too narrowly.

- C48005C-B: Lines 38 and 63 of this test should check that the value of the designated object is null.

- C64103C-B: This test should raise CONSTRAINT_ERROR during the conversion at line 179.

- C64103D-B: This test involves an issue of whether CONSTRAINT_ERROR or NUMERIC_ERROR is to be raised. Because the issue could not be resolved quickly, the test was withdrawn from Version 1.6.

- C64105E-AB and C64105F-AB: These tests should ensure (in case E) that non-null dimensions of formal and actual parameters belong to both index subtypes (see AI-00313).

- B66001A-B: This test checks (in section G) that a function without parameters which is equivalent to an enumeration literal in the same declarative region is a redeclaration and as such is forbidden. According to section 8.3, paragraph 17 of the Reference Manual, the explicit declaration of such a function is allowed if an enumeration literal is considered to be an implicitly declared predefined operation. The Reference Manual is not clear on this point. Because the issue could not be resolved quickly, the test was withdrawn from Version 1.6.

- B67001A-B: This test is missing the "BEGIN NULL; END;" at line 414 needed to complete the block for case H beginning at line 389.

- B67004A-B: In this test, the default name for a formal generic equality function should not be allowed to be "/=" unless an expanded name is used.

- B74103F-B: This test depends on whether or not a generic formal type declaration declares a type. Because the issue could not be resolved quickly, the test was withdrawn from Version 1.6.

- C93005A-B, C93005B-B, and C93005C-B: These tests contain a declaration of an integer variable whose initialization is solely for the purpose of raising an exception. Some compilers will not raise this exception due to allowable optimizations.

- C93007B-B: This test should check for PROGRAM_ERROR rather than TASKING_ERROR (see AI-00149).

- CA1003B-AB: A compilation that contains an illegal compilation unit may now be rejected as a whole (see AI-00255/05).

- CA1011A-B: The test objective should be reversed to be consistent with AI-00199.

- CA1108A-B: A pragma ELABORATE is needed for OTHER_PKG at line 25.

- CA1108B-B: A pragma ELABORATE is needed for FIRST_PKG at line 39 and for LATER_PKG at line 49.

- BA2001E-AB: Section 10.2, paragraph 5 of the Reference Manual states, "Simple names of all subunits that have the same ancestor library unit must be distinct identifiers." This test requires that the above conditions be checked when the stub is declared. However, since the Reference Manual uses the term "subunit," it is not clear that the check must be made at the declaration or when the subunit is compiled.

- CA2009B-B and CA2009E-B: In these tests, the repetition of the main procedure after the subunit body makes the subunit body obsolete. Therefore, an attempt to execute the main procedure will fail.

- CA2009F-B: The file CA2009F2-B, which is part of this test, is missing from the test suite.

- BC1013A-B: In this test, the declaration of equality in lines 86 and 87 is illegal because the parameter type T declared in line 11 is not a limited type (see section 6.7, paragraph 4 of the Reference Manual).

- BC3204A-B, BC3204B-B, BC3204C-B, BC3204D-B, BC3205A-B, BC3205B-B, BC3205C-B, BC3205D-B, and BC3405B-B: Instantiations with types that have default discriminants are legal (see AI-00037).

- BC3220B-B: This test assumes that instantiated types may be static. Because the issue could not be resolved quickly, the test was withdrawn from Version 1.6.

- CE2107E-B: This test has a variable, TEMP_HAS_NAME, that should have been initialized to TRUE.

- CE3603A-B: In this test, the last case is inconsistent with AI-00050. If a string argument is null, no attempt to read is made and END_ERROR is not raised.

- CE3604A-B: Cases 5, 8, 9, and 11 in this test are inconsistent with AI-00050. SKIP_LINE is called only if the end of the output string has not been met.

- CE3704M-B: A superfluous SKIP_LINE causes the input and output operations to be out of synchronization.

2.3 INAPPLICABLE TESTS

Some tests do not apply to all compilers because they make use of features that a compiler is not required by the Ada Standard to support. Others may depend on the result of another test that is either inapplicable or withdrawn. For this validation attempt, 231 tests were inapplicable for the reasons indicated:

- 221 tests were not processed because SYSTEM.MAX_DIGITS was 9. Those tests were:

  C24113F-B.DEP through C24113Y-B.DEP (20 tests)
  C35705F-B.DEP through C35705Y-B.DEP (20 tests)
  C35706F-B.DEP through C35706Y-B.DEP (20 tests)
  C35707F-B.DEP through C35707Y-B.DEP (20 tests)
  C35708F-B.DEP through C35708Y-B.DEP (20 tests)
  C35802F-B.DEP through C35802Y-B.DEP (20 tests)
  C45241F-B.DEP through C45241Y-B.DEP (20 tests)
  C45321F-B.DEP through C45321Y-B.DEP (20 tests)
  C45421F-B.DEP through C45421Y-B.DEP (20 tests)
  C45424F-B.DEP through C45424Y-B.DEP (20 tests)
  C45621F-B.DEP through C45621Z-B.DEP (21 tests)

- Five tests were inapplicable because this implementation does not support LONG_INTEGER.

  C34001E-B.DEP B55B09C-AB.DEP B86001CS-AB.DEP
  B52004D-AB.DEP C55B07A-AB.DEP

- Three tests were inapplicable because this implementation does not support LONG_FLOAT.

  C34001G-B.DEP C35702B-AB.DEP B86001CQ-AB.DEP

- Test C86001F-B.DEP redefines package SYSTEM, but TEXT_IO is made obsolete by this new definition in this implementation.

- Test C96005B.TST is not applicable for this implementation because the smallest and largest values in type DURATION are equal to the smallest and largest values in DURATION's base type.

2.4 IMPLEMENTATION CHARACTERISTICS

One of the purposes for validating is to determine the behavior of a compiler in those areas of the Ada Standard that permit implementations to differ. Class D and E tests specifically check for such implementation differences. However, inapplicable tests in other classes also characterize an implementation. This compiler is characterized by the following interpretations of the Ada Standard:

- Nongraphic characters.

  Nongraphic characters are defined in the ASCII character set but are not permitted in Ada programs, even within character strings. The compiler correctly recognizes these characters as illegal in Ada compilations. The characters are not echoed in the output listing, but are printed in a graphic representation, for example "^A".

- Capacities.

  The compiler correctly processes compilations containing loop statements nested to 65 levels, block statements nested to 65 levels, procedures nested to 17 levels, and declarative parts containing 723 variables.

- Universal integer calculations.

  An implementation is allowed to reject universal integer calculations having values that exceed SYSTEM.MAX_INT. This implementation does not reject such calculations and processes them correctly.

. Predefined types.

This implementation supports the predefined types TINY_INTEGER, SHORT_INTEGER, and SHORT_FLOAT.

. Based literals.

An implementation is allowed to reject a based literal with value exceeding SYSTEM.MAX_INT during compilation or it may raise NUMERIC_ERROR during execution. This compiler raises NUMERIC_ERROR during execution.

. Array types.

An implementation is allowed to raise NUMERIC_ERROR for an array having a 'LENGTH that exceeds STANDARD.INTEGER'LAST and/or SYSTEM.MAX_INT. When an array type is declared with an index range exceeding INTEGER values and with a component that is a null BOOLEAN array, this compiler raises NUMERIC_ERROR when 'LENGTH is evaluated.

When an array type is declared with an index range exceeding SYSTEM.MAX_INT values and with a component that is a null BOOLEAN array, this compiler does not raise any exception.

A packed BOOLEAN array of length STANDARD.INTEGER'LAST+3 raises NUMERIC_ERROR when the array type is declared. A packed two-dimensional BOOLEAN array with STANDARD.INTEGER'LAST+3 components raises NUMERIC_ERROR when the array type is declared.

In assignment of one-dimensional array types, the entire expression is evaluated before CONSTRAINT_ERROR is raised when checking whether the expression's subtype is compatible with the target's subtype. In assignment of two-dimensional array types, the entire expression is not evaluated before CONSTRAINT_ERROR is raised when checking whether the expression's subtype is compatible with the target's subtype. In assignment of record types with discriminants, the entire expression is evaluated before CONSTRAINT_ERROR is raised when checking whether the expression's subtype is compatible with the target's subtype.

. Discriminated types.

An incompletely declared type with discriminants may be used in an access type definition and constrained either there or in later subtype indications.

- Aggregates.

  When evaluating the choices of a multi-dimensional aggregate, all choices are evaluated before checking against the index type.

  When evaluating an aggregate containing subaggregates, all choices are evaluated before being checked for identical bounds.

- Representation clauses.

  'SMALL length clauses are supported.

  Enumeration representation clauses are supported.

- Package CALENDAR.

  TIME_OF and SPLIT are inverses when SECONDS is a nonmodel number.

- Pragmas.

  Pragma INLINE is supported for procedures. It is also supported for functions.

- Input/Output.

  Package SEQUENTIAL_IO can be instantiated with unconstrained array types or record types with discriminants. Package DIRECT_IO can be instantiated with unconstrained array types or record types with discriminants without defaults.

  Temporary SEQUENTIAL_IO files are given a name. Temporary DIRECT_IO files are given a name. Temporary files given names are deleted when they are closed.

CHAPTER 3

COMPILER ANOMALIES AND NONCONFORMANCES

3.1 ANOMALIES

An anomaly is a test result that, given the pre-validation analysis, was not expected during formal validation but which is judged allowable by the AVF and the AVO under the circumstances of the validation. No anomalies were detected in this validation attempt.

3.2 NONCONFORMANCES

Any discrepancy between expected test results and actual test results is considered a nonconformance. No nonconformances were detected in this validation attempt.

CHAPTER 4

ADDITIONAL TESTING INFORMATION

4.1 PRE-VALIDATION

Prior to validation, a set of test results for ACVC Version 1.6 produced by the VADS, Version 5.2, was submitted to the AVF by the applicant for pre-validation review. Analysis of these results demonstrated that the compiler successfully passed all applicable tests.

4.2 TEST SITE

Tests were compiled and executed at Verdix Corporation in Aloha, Oregon.

4.3 TEST TAPE INFORMATION

A test tape containing ACVC Version 1.6 was taken on-site by the validation team. This tape contained all tests applicable to this validation as well as all tests inapplicable to this validation except for any Class C tests that require floating-point precision exceeding the maximum value supported by the implementation. Tests that were withdrawn from ACVC Version 1.6 were not written to the tape. Tests that make use of values that are specific to an implementation were customized before being written to the tape. Any split tests were also included on the test tape so that no editing of the test files was necessary when the validation team arrived on-site.

The test tape was written in ANSI standard format and loaded onto a CCI Power computer to run the tests.

4.4 TESTING LOGISTICS

Once all tests had been loaded to disk, processing was begun using command scripts provided by Verdix Corporation. The text of these scripts is given in Appendix C.

Tests were run in chapter order. One test was run at a time. A separate program library was used for each set of tests within a chapter.

4.5 TESTING DURATION

The ACVC has not been designed for use in measuring compiler performance. The information reported here thus merely describes the duration of the on-site conformity testing, and is not necessarily an indication of the subject system's performance for any particular application.

The tests required about 10 hours to run. The computer was dedicated to running the tests.

CHAPTER 5

SUMMARY AND CONCLUSIONS

The Ada Validation Facility identified 1865 tests in Version 1.6 of the Ada Compiler Validation Capability as applicable to VADS, Version 5.2. All applicable tests were passed by the compiler.

The Ada Validation Facility concludes that these results demonstrate acceptable conformance to the Ada Standard.

APPENDIX A

COMPLIANCE STATEMENT

The only allowed implementation dependencies correspond to implementation-dependent pragmas and attributes, to certain machine-dependent conventions as mentioned in chapter 13 of MIL-STD-1815A, and to certain allowed restrictions on representation clauses. The implementation-dependent characteristics of the VADS, Version 5.2, are described in the following sections which discuss topics one through eight as stated in Appendix F of the Ada Language Reference Manual (ANSI/MIL-STD-1815A).

(1) Implementation-dependent Pragmas

Pragma	Form, Placement, Effect
SHARE_BODY	The SHARE_BODY pragma takes the name of a generic instantiation or a generic unit as the first argument and one of the identifiers TRUE or FALSE as the second argument. This pragma is only allowed immediately at the place of a declarative item in a declarative part or package specification, or after a library unit in a compilation, but before any subsequent compilation unit. When the first argument is a generic unit, the pragma applies to all instantiations of that generic. When the first argument is the name of a generic instantiation, the pragma applies only to the specified instantiation or overloaded instantiations.

If the second argument is TRUE, the compiler will try to share code generated for a generic instantiation with code generated for other instantiations of the same generic. When the second argument is FALSE, each instantiation will get a unique copy of the generated code. The extent to which code is shared between instantiations depends on this pragma and the kind of generic formal parameters declared for the generic unit.

EXTERNAL_NAME — The EXTERNAL_NAME pragma takes the name of a variable defined in another language and allows it to be referenced directly in Ada. The pragma will replace all occurrences of the variable name with an external reference to the second link_argument. The pragma is allowed at the place of a declarative item in a package specification and must apply to an object declared earlier in the same package specification. The object must be declared as a scalar or an access type. The object cannot be a loop variable, a constant, an initialized variable, an array, or a record.

(2) Implementation-dependent Attributes

There are no implementation-dependent attributes.

(3) Package SYSTEM

The specification for package SYSTEM is

```
package SYSTEM is

    type ADDRESS is private;
    type NAME is (power_6_unix);

    SYSTEM_NAME  : constant NAME := power_6_unix;
    STORAGE_UNIT : constant := 8;
    MEMORY_SIZE  : constant := 3_221_225_469;

    --  System-Dependent Named Numbers:

    MIN_INT      : constant := -2_147_483_647 - 1;
    MAX_INT      : constant :=  2_147_483_647;
    MAX_DIGITS   : constant := 9;
    MAX_MANTISSA : constant := 31;
    FINE_DELTA   : constant := 2.0**(-14);
    TICK         : constant := 0.01;

    --  Other System-Dependent Declarations

    subtype PRIORITY is INTEGER range 0..7 ;

    MAX_REC_SIZE : INTEGER := 64*1024;

private

    type ADDRESS is new INTEGER;

end SYSTEM;
```

(4) Representation Clause Restrictions

Representation clauses specify how the types of the language are to be mapped onto the underlying machine. The following are restrictions on representation clauses.

Pragma PACK

Bit packing is not supported. Objects and components are packed to the nearest whole STORAGE_UNIT.

Size Specification

The size specification T'SMALL is not supported.

Record Representation Clauses

Component clauses must be aligned on STORAGE_UNIT boundaries.

Address Clauses

Address clauses are not supported.

Interrupts

Interrupts are not supported.

Change Of Representation

Change of representation is not supported for record types.

Representation Attributes

The ADDRESS attribute is not supported for the following entities:

Packages
Tasks
Labels
Entries

(5) Conventions

There are no implementation-generated names.

(6) Address Clauses

Address clauses are not supported.

(7) Unchecked Conversions

The predefined generic function UNCHECKED_CONVERSION cannot be instantiated with a target type which is an unconstrained array type or an unconstrained record type with discriminants.

(8) Input-Output Packages

The following are implementation-dependent characteristics of the input-output packages.

SEQUENTIAL_IO Package

Instantiations of SEQUENTIAL_IO use the value MAX_REC_SIZE as the record size (expressed in STORAGE_UNITS) when the size of ELEMENT_TYPE exceeds that value. For example, for unconstrained arrays such as string where ELEMENT_TYPE'SIZE is very large, MAX_REC_SIZE is used instead. MAX_REC_SIZE is defined in package SYSTEM and can be changed by a program before instantiating INTEGER_IO to provide an upper limit on the record size. SEQUENTIAL_IO imposes no limit on MAX_REC_SIZE.

DIRECT_IO Package

```
type COUNT is range 0 .. 2_147_483_647;
```

Instantiations of DIRECT_IO use the value MAX_REC_SIZE as the record size (expressed in STORAGE_UNITS) when the size of ELEMENT_TYPE exceeds that value. For example, for unconstrained arrays such as string where ELEMENT_TYPE'SIZE is very large, MAX_REC_SIZE is used instead. MAX_REC_SIZE is defined in package SYSTEM and can be changed by a program before instantiating DIRECT_IO to provide an upper limit on the record size. In any case, the maximum size supported is (1024 * 1024 * STORAGE_UNIT) bits. DIRECT_IO will raise USE_ERROR if MAX_REC_SIZE exceeds this absolute limit.

LOW_LEVEL_IO

Low-level Input/Output is not provided.

(9) Package STANDARD

```
type INTEGER is range -2_147_483_648 .. 2_147_483_647;

type SHORT_INTEGER is range -32768 .. 32767;

type TINY_INTEGER is range -128 .. 127;

type FLOAT is digits 9 range
 -2#0.11111111111111111111111111111111111111111111111111111111E127#
                          ..
  2#0.11111111111111111111111111111111111111111111111111111111E127#;

type SHORT_FLOAT is digits 6 range
    -2#0.1111111111111111111111E127# ..
     2#0.1111111111111111111111E127#;

type DURATION is delta 2#1.0E-14# range
     -2#100000000000000000.0# ..
      2#111111111111111.111111111111#;
```

(10) File Names

File names must conform to the conventions and restrictions of the host operating system.

APPENDIX B

TEST PARAMETERS

Certain tests in the ACVC make use of implementation-dependent values, such as the maximum length of an input line and invalid file names. A test that makes use of such values is identified by the extension .TST in its file name. Actual values to be substituted are identified by names that begin with a dollar sign. A value is substituted for each of these names before the test is run. The values used for this validation are given below.

Name and Meaning	Value
$MAX_IN_LEN Maximum input line length permitted by the implementation.	500 (including one for end-of-line character)
$BIG_ID1 Identifier of size MAX_IN_LEN with varying last character.	(1 .. 498 => 'A', 499 => '1')
$BIG_ID2 Identifier of size MAX_IN_LEN with varying last character.	(1 .. 498 => 'A', 499 => '2')
$BIG_ID3 Identifier of size MAX_IN_LEN with varying middle character.	(1 .. 249 => 'A', 250 = '3', 251 .. 499 => 'A')
$BIG_ID4 Identifier of size MAX_IN_LEN with varying middle character.	(1 .. 249 => 'A', 250 => '4', 251 .. 499 => 'A')

TEST PARAMETERS

Name and Meaning	Value
$NEG_BASED_INT A based integer literal whose highest order nonzero bit falls in the sign bit position of the representation for SYSTEM.MAX_INT.	16#FFFFFFFD#
$BIG_INT_LIT An integer literal of value 298 with enough leading zeroes so that it is MAX_IN_LEN characters long.	(1 .. 496 => '0', 497 .. 499 => "298")
$BIG_REAL_LIT A real literal that can be either of floating- or fixed-point type, has value 690.0, and has enough leading zeroes to be MAX_IN_LEN characters long.	(1 .. 493 => '0', 494 .. 499 => "69.0E1")
$EXTENDED_ASCII_CHARS A string literal containing all the ASCII characters with printable graphics that are not in the basic 55 Ada character set.	"abcdefghijklmnopqrstuvwxyz!$%?@[\]^'{}~"
$NON_ASCII_CHAR_TYPE An enumerated type definition for a character type whose literals are the identifier NON_NULL and all non-ASCII characters with printable graphics.	(NON_NULL)
$BLANKS Blanks of length MAX_IN_LEN - 20	(1 .. 479 => ' ')
$MAX_DIGITS Maximum digits supported for floating-point types.	9
$NAME A name of a predefined numeric type other than FLOAT, INTEGER, SHORT_FLOAT, SHORT_INTEGER, LONG_FLOAT, or LONG_INTEGER.	TINY_INTEGER

Name and Meaning	Value
$INTEGER_FIRST The universal integer literal expression whose value is INTEGER'FIRST.	-2_147_483_648
$INTEGER_LAST The universal integer literal expression whose value is INTEGER'LAST.	2_147_483_647
$LESS_THAN_DURATION A universal real value that lies between DURATION'BASE'FIRST and DURATION'FIRST or any value in the range of DURATION.	-100_000.0
$GREATER_THAN_DURATION A universal real value that lies between DURATION'BASE'LAST and DURATION'LAST or any value in the range of DURATION.	100_000.0
$LESS_THAN_DURATION_BASE_FIRST The universal real value that is less than DURATION'BASE'FIRST.	-10_000_000.0
$GREATER_THAN_DURATION_BASE_LAST The universal real value that is greater than DURATION'BASE'LAST.	10_000_000.0
$COUNT_LAST Value of COUNT'LAST in TEXT_IO package.	2_147_483_647
$FIELD_LAST Value of FIELD'LAST in TEXT_IO package.	2_147_483_647
$FILE_NAME_WITH_BAD_CHARS An illegal external file name that either contains invalid characters or is too long.	"/illegal/file_name/2{]$%2102C.DAT"

TEST PARAMETERS

Name and Meaning	Value
$FILE_NAME_WITH_WILD_CARD_CHAR An external file name that either contains a wild card character or is too long.	"/illegal/file_name/CE2102C*.DAT"
$ILLEGAL_EXTERNAL_FILE_NAME1 Illegal external file name.	"/no/such/directory/ILLEGAL_" & "EXTERNAL_FILE_NAME1"
$ILLEGAL_EXTERNAL_FILE_NAME2 Illegal external file name.	"/no/such/directory/ILLEGAL_" & "EXTERNAL_FILE_NAME2"

APPENDIX C

COMMAND SCRIPTS

The following command language scripts and C language programs were used in running ACVC Version 1.6. The compile/link/execute cycle was done with Makefiles, using the make facility, which in turn called the compile program. All other programs are parts of the VADS, Version 5.2.

```
---------- run.acvc script ----------
#! /bin/csh -f
set nonomatch
unsetenv as
set chpts = ($*)
foreach i ($*)
  cd /usr/acvc/$i
  echo "--------- start $i : `uptime | sed 's/up.*load/load/'` -------"
  make all
  rm -f .objects/*
  rm -f .nets/*
  rm -f .lines/*
end

echo "--------- finish : `uptime | sed 's/up.*load/load/'` ----------"
```

```
---------------- compile.c ----------------
#include <stdio.h>
#ifdef VMS
#include <types.h>
#include <stat.h>
#include <rms.h>
#include <descrip.h>
#include <ssdef.h>
#include <stsdef.h>
/*  file.h  6.2  83/09/23  */

#ifdef KERNEL
/*
 * Descriptor table entry.
 * One for each kernel object.
 */
struct  file {
  int  f_flag;     /* see below */
  short  f_type;     /* descriptor type */
  short  f_count;  /* reference count */
  short  f_msgcount;  /* references from message queue */
  struct  fileops {
    int  (*fo_rw)();
    int  (*fo_ioctl)();
    int  (*fo_select)();
    int  (*fo_close)();
  } *f_ops;
  caddr_t  f_data;     /* inode */
  off_t  f_offset;
};

struct  file *file, *fileNFILE;
int  nfile;
struct  file *getf();
struct  file *falloc();
#endif

/*
 * flags- also for fcntl call.
 */
#define  FOPEN     (-1)
#define  FREAD     00001     /* descriptor read/receive'able */
#define  FWRITE     00002     /* descriptor write/send'able */
#ifndef  F_DUPFD
#define  FNDELAY     00004     /* no delay */
#define  FAPPEND     00010     /* append on each write */
#endif
#define  FMARK     00020     /* mark during gc() */
#define  FDEFER     00040     /* defer for next gc pass */
#ifndef  F_DUPFD
#define  FASYNC     00100     /* signal pgrp when data ready */
#endif
#define  FSHLOCK     00200     /* shared lock present */
#define  FEXLOCK     00400     /* exclusive lock present */
```

```
/* bits to save after open */
#define  FMASK     00113
#define  FCNTLCANT  (FREAD|FWRITE|FMARK|FDEFER|FSHLOCK|FEXLOCK)

/* open only modes */
#define  FCREAT     01000     /* create if nonexistant */
#define  FTRUNC     02000     /* truncate to zero length */
#define  FEXCL     04000     /* error if already created */

#ifndef  F_DUPFD
/* fcntl(2) requests--from <fcntl.h> */
#define  F_DUPFD  0  /* Duplicate fildes */
#define  F_GETFD  1  /* Get fildes flags */
#define  F_SETFD  2  /* Set fildes flags */
#define  F_GETFL  3  /* Get file flags */
#define  F_SETFL  4  /* Set file flags */
#define  F_GETOWN 5  /* Get owner */
#define F_SETOWN 6  /* Set owner */
#endif

/*
 * User definitions.
 */

/*
 * Open call.
 */
#define  O_RDONLY  000     /* open for reading */
#define  O_WRONLY  001     /* open for writing */
#define  O_RDWR    002     /* open for read & write */
#define  O_NDELAY  FNDELAY     /* non-blocking open */
#define  O_APPEND  FAPPEND     /* append on each write */
#define  O_CREAT    FCREAT     /* open with file create */
#define  O_TRUNC    FTRUNC     /* open with truncation */
#define  O_EXCL    FEXCL     /* error on create if file exists */

/*
 * Flock call.
 */
#define  LOCK_SH    1  /* shared lock */
#define  LOCK_EX    2  /* exclusive lock */
#define  LOCK_NB    4  /* don't block when locking */
#define  LOCK_UN    8  /* unlock */

/*
 * Access call.
 */
#define  F_OK    0  /* does file exist */
#define  X_OK    1  /* is it executable by caller */
#define  W_OK    2  /* writable by caller */
#define  R_OK    4  /* readable by caller */

/*
 * Lseek call.
 */
```

```
#define  L_SET     0  /* absolute offset */
#define  L_INCR    1  /* relative to current offset */
#define  L_XTND    2  /* relative to end of file */

#ifdef KERNEL
#define  GETF(fp, fd) {   if ((unsigned)(fd) >= NOFILE ||
((fp) = u.u_ofile[fd]) == NULL) {     u.u_error = EBADF;     return;   } }
#define  DTYPE_INODE  1  /* file */
#define  DTYPE_SOCKET  2  /* communications endpoint */
#endif

#ifdef ADA
#define ADA         "ada"
#else
#define FE          "fe"
#define CG          "cg"
#define LD          "a_ld"
#endif

#define CMP         "diff"
#define RINDEX       strrchr
#define UNLINK       delete
#define OPEN_WRITE     0

#define M_OPTION     ""
#define SUCCESS       STS$K_SUCCESS
#define FATAL       STS$K_SEVERE
char *strrchr();

#else
#include <strings.h>
#include <sys/file.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/vlimit.h>

#define ADA       "/vc/install/bin/ada"
#define LD       "/vc/install/bin/a.ld"
#define ACVC     "/usr/acvc"
#define CMP         "/bin/cmp"
#define RINDEX       rindex
#define UNLINK       unlink
#define OPEN_WRITE     0666

#define M_OPTION  "-M"
#define SUCCESS       0
#define FATAL       1
#endif

#define W_MODE     "w"
#define A_MODE     "a"

#define C_CHAR       'C'
#define X_CHAR       'X'
#define L_CHAR       'L'
```

```
#define C_STR       "C"
#define X_STR       "X"
#define L_STR       "L"

#define FALSE     (0)
#define TRUE      (1)

char *self;
char *getwd();
char **command;

main(argc, argv)
register int    argc;
register char **argv;
{
  char found;
  char *mode;
  int status;
  char *file;
  char *chpt;
  char *tail;
  char *flag;
  char image[BUFSIZ];
  char cwd[BUFSIZ];
  char root[BUFSIZ];
  char il[BUFSIZ];
  char output[BUFSIZ];
  char from[BUFSIZ];
  char src_file[BUFSIZ];
  struct stat statbuf;
  int out_fd;
  int orig_stderr;
  int orig_stdout;

  if(argc < 2 || argv[1][0] != '-') {
    goto usage;
  }

  self = argv[0];
  flag = argv[1];
  file = argv[2];

#ifdef VMS
  strcpy(src_file, file);
#else
  vlimit(LIM_CPU, 500);
  vlimit(LIM_FSIZE, 1000000);
  vlimit(LIM_CORE, 0);

  strcpy(cwd, ".");
  getwd(cwd);

  chpt = RINDEX(cwd, '/');
  strcpy(src_file, ACVC);
  strcat(src_file, chpt);
```

```
    strcat(src_file, "/");
    strcat(src_file, file);
    argv[2] = src_file;
#endif

    strcpy(output, file);
    tail = RINDEX(output, '.') + 1;

    *(tail - 1) = ' ';
    strcpy(root, output);
    *(tail - 1) = '.';

    switch(flag[1]) {
    case 'c':
      printf("compile -c %s0, file);
      command = &(argv[1]);
#ifdef ADA
      command[0] = ADA;
#else
      command[0] = FE;
#endif
      mode = W_MODE;
      *tail = C_CHAR;
      break;
    case 'x':
      printf("compile -x %s0, file);
#ifdef ADA
      command = argv;
      command[0] = ADA;
      command[1] = M_OPTION;
#else
      command = &(argv[1]);
      command[0] = FE;
#endif
#ifdef VMS
      strcpy(image, root);
      strcat(image, ".exe");
#else
      strcpy(image, "a.out");
#endif
      mode = A_MODE;
      *tail = X_CHAR;
      break;
    case 'l':
      printf("compile -l %s0, file);
      command = &(argv[1]);
      command[0] = LD;
      mode = A_MODE;

#ifdef VMS
      strcpy(image, root);
      strcat(image, ".exe");
#else
      strcpy(image, "a.out");
#endif
```

```
    command[1] = root;

    *tail = ' ';
    strcpy(from, output);
    strcat(from, C_STR);
    strcat(output, L_STR);
    rename(from, output);
    break;
  default:
    goto usage;
  }

#ifdef VMS
  status = spawn_process(command, output);
#ifndef ADA
  if(flag[1] == 'x') {
    if(status == 1) {
      strcpy(il, "[.objects]");
      strcat(il, root);
      strcat(il, ".i");
      command[0] = CG;
      command[1] = il;
      command[2] = 0;
      status = spawn_process(command, "tmp.out");
      append_output("tmp.out", output);
    } else {
      return;
    }
    if(status == 1) {
      command[0] = LD;
      command[1] = root;
      command[2] = 0;
      status = spawn_process(command, "tmp.out");
      append_output("tmp.out", output);
    } else {
      return;
    }
#endif
  }
#else
  if(freopen(output, mode, stdout) == NULL) {
    perror("compile: freopen");
    exit(1);
  }
  dup2(1, 2);

  status = callsys(command[0], command);
#endif

  if(flag[1] != 'c' && stat(image, &statbuf) == 0) {
    if(statbuf.st_mode & S_IEXEC != 0) {
#ifdef VMS
      command[0] = "run";
      command[1] = image;
      command[2] = 0;
```

```
        spawn_process(command, "tmp.out");
        append_output("tmp.out", output);
#else
        command[0] = image;
        command[1] = 0;
        status = callsys(image, command);
#endif
        UNLINK(image);
      }
    }
    exit(SUCCESS);

usage:
    printf("usage: compile -clx file [ada_options]0);
    exit(1);
}

#ifdef VMS
int vms_error;

#define STRBUFSZ 16
struct dsc$descriptor strs[STRBUFSZ];
int dsc_i = 0;

struct dsc$descriptor *
new_str(str)
char *str;
{
    register struct dsc$descriptor *dsc;

    if (dsc_i >= STRBUFSZ){
      dsc_i = 0;
    }
    dsc = &strs[dsc_i++];

    dsc->dsc$w_length = strlen(str);
    dsc->dsc$b_dtype = DSC$K_DTYPE_T;
    dsc->dsc$b_class = DSC$K_CLASS_S;
    dsc->dsc$a_pointer = str;
    return dsc;
}

spawn_process(image_name, output)
char **image_name;
char *output;
{
    struct dsc$descriptor *image_str;
    int new_pid;
    int status;
    int i;
    int flags = 0x0;
    char cmd[512];

    *cmd = ' ';
    i = 0;
```

```
  while(image_name[i] != 0) {
    strcat(cmd, image_name[i++]);
    strcat(cmd, " ");
  }

  image_str = new_str(cmd);
  vms_error = lib$spawn(image_str, 0, new_str(output),
   &flags, 0, 0, &status,
    0, 0, 0, 0, 0);
  if (vms_error != SS$_NORMAL){
    vms_perror(vms_error, "lib$spawn: ");
  }
  return $VMS_STATUS_SEVERITY(status);
}

vms_perror(lcv, format, msg1, msg2, msg3, msg4)
  long lcv;
  char *format;
  int msg1;
  int msg2;
  int msg3;
  int msg4;
{
  unsigned short len;
  int status;
  int outadr;
  char buf[257];
  struct dsc$descriptor ch_str;


  if (format != 0){
    printf(format, msg1, msg2, msg3, msg4);
  }

  ch_str.dsc$w_length = 256;
  ch_str.dsc$b_dtype = DSC$K_DTYPE_T;
  ch_str.dsc$b_class = DSC$K_CLASS_S;
  ch_str.dsc$a_pointer = buf;

  status = sys$getmsg(lcv, &len, &ch_str, 0xf, &outadr);
  if (status == SS$_MSGNOTFND){
    printf("sys$getmsg: Couldn't find msg for: 0x%x0, lcv);
  }

  buf[len] = ' ';
  printf("%s0, buf);
  return;
}

rename(from, to)
char *from;
char *to;
{
  int status;
  struct dsc$descriptor_s to_d;
```

```
    struct dsc$descriptor_s from_d;

    from_d.dsc$w_length = strlen(from);
    from_d.dsc$b_dtype = DSC$K_DTYPE_T;
    from_d.dsc$b_class = DSC$K_CLASS_S;
    from_d.dsc$a_pointer = from;

    to_d.dsc$w_length = strlen(to);
    to_d.dsc$b_dtype = DSC$K_DTYPE_T;
    to_d.dsc$b_class = DSC$K_CLASS_S;
    to_d.dsc$a_pointer = to;

    status = lib$rename_file(&from_d, &to_d, 0,0,0,0,0,0,0,0,0,0);

    if(status == SS$_NORMAL) {
      return 0;
    } else {
      return status;
    }
  }

append_output(input, output)
char *input;
char *output;
{
  int out_fd;
  int in_fd;
  int status;
  char c;

  out_fd = open(output, O_WRONLY | O_APPEND, OPEN_WRITE);
  if(out_fd == -1) {
    perror("open");
    return;
  }

  in_fd = open(input, O_RDONLY, 0);
  if(in_fd == -1) {
    perror("open");
    return;
  }

  while((status = read(in_fd, &c, 1)) != 0) {
    if(status == -1) {
      perror("read");
      return;
    }
    status = write(out_fd, &c, 1);
    if(status == -1) {
      perror("write");
      return;
    }
  }

  if(close(out_fd) == -1) {
```

```
    perror("compile:close out_fd:");
  }
  UNLINK(input);
}

#else
callsys(f, v)
char *f;
char **v;
{
  int  status;
  register int  t;
  int pos1, pos2;

  t = vfork();
  if (t == -1) {
    fprintf(stderr, "No more processes.");
    fflush(stderr);
    _exit(FATAL);
    /* not reached */
  }
  if (t == 0) {
    execv(f, v);
    fprintf(stderr, "%s: Can't find %s0, self, f);
    fflush(stderr);
    _exit(FATAL);
  }
  while (t != wait((int *)&status))
    ;
  if ((t=(status&0377)) != 0 && t!=14) {
    if (t!=2)
    {
      fprintf(stderr, "%s: Fatal error in %s0, self, f);
      fflush(stderr);
    }
    exit(FATAL);
  }
  return ((status>>8) & 0377);
}

#endif
```

```
---------------- L script ----------------
#! /bin/csh
# acvc listing


set chpt = $cwd:t
set RESULTS = "/usr/acvc/certify.power/LIST_$chpt"

foreach i ($*)
  set TEST = $i:r
  set LIST = "$RESULTS/$TEST"
  set CHECK = "$TEST"
  /vc/dcn/ada/tools/a.error -l < $CHECK > $LIST
  if($status != 0) then
    a.list $i >> $LIST
  endif
  echo " " >> $LIST
  echo " " >> $LIST
  echo " " >> $LIST
  echo "<<<<<<<<<" FROM COMPILATION and EXECUTION ">>>>>>>>>" >> $LIST
  echo " " >> $LIST
  cat $CHECK >> $LIST
end
```

APPENDIX D

COMPLETE LIST OF TESTS AND RESULTS

This Appendix presents a complete list of the ACVC test files used in the validation attempt, presented in order by ACVC Implementers' Guide section and objective. Each test name indicates the class of the test and which test objective in the ACVC Implementers' Guide applies to the test.

Each test has a name that identifies the section of the Ada Standard addressed by the test objective. The name of a test is interpreted according to the table below, where the first column indicates the character position in the name and the second column, the meaning of that position:

POS	MEANING
1	Test class: A, B, C, D, E, L.
2	Implementers' Guide chapter number (in hexadecimal).
3	Implementers' Guide section number within a chapter (in Hexadecimal).
4	Implementers' Guide subsection number (in hexadecimal).
5-6	Implementers' Guide Test Objective number (in decimal).
7	Test sequence letter.
8	[Optional] Compilation sequence digit or letter.
9	[Optional] Main program designator in the case of a test having multiple compilation units.

Characters 8 and 9 are only present for tests that consist of several separately compiled units. A series of separately compiled units is counted as one test for reporting purposes. The eighth character indicates the order in which the units are to be compiled, with unit 0 being compiled first. The ninth character is only present for a file containing a main program for a test comprised of multiple files and is always M.

COMPLETE LIST OF TESTS AND RESULTS

The suffix -AB means the test was written prior to release of the ANSI Standard and is also valid for the version of Ada published in July 1980. The suffix -B means the test was written specifically for the ANSI Standard. Tests without a suffix have not yet had their names revised to -AB.

A file name ending with the extension .TST indicates that the test depends on one or more of the implementation-dependent parameters listed in Appendix B. A file name ending with .DEP indicates that the test is not necessarily applicable to all implementations because it depends upon the support of language features that a compiler may legally not implement.

The result for each file in ACVC Version 1.6 is given in the following pages, where:

P indicates Passed.

F indicates Failed.

N/A indicates Not Applicable to this implementation.

W indicates Withdrawn due to test errors.

C indicates Compiled without error.

A indicates Anomalous.

Indented names are separately compiled units (subtests) of the test under which they appear. A sequence of indented subtest names comprise one test for reporting purposes.

Support Units

VAR_STRINGS_SPEC.ADA	P
VAR_STRINGS_BODY.ADA	P
REPORT_SPEC-AB.ADA	P
REPORT_BODY-B.ADA	P
CHECK_FILE-B.ADA	P
CZ1101A-AB.ADA	P
CZ1102A-AB.ADA	P
CZ1103A-B.ADA	P
CZ1201A-AB.ADA	P
CZ1201B-AB.ADA	P
CZ1201C-AB.ADA	P
CZ1201D-AB.ADA	P

COMPLETE LIST OF TESTS AND RESULTS

Chapter 2

A21001A.ADA	P	B23002A.ADA	P	C24113C-B.DEP	P
A22002A.ADA	P	B23003D-AB.TST	P	C24113D-B.DEP	P
A26004A.TST	P	B23003E-AB.TST	P	C24113E-B.DEP	P
A29002A-B.ADA	P	B23003F-AB.TST	P	C24113F-B.DEP	N/A
A29002B-B.ADA	P	B23004A.ADA	P	C24113G-B.DEP	N/A
A29002C-B.ADA	P	B23004B.ADA	P	C24113H-B.DEP	N/A
A29002D-B.ADA	P	B24001A.ADA	P	C24113I-B.DEP	N/A
A29002E-B.ADA	P	B24001B.ADA	P	C24113J-B.DEP	N/A
A29002F-B.ADA	P	B24001C.ADA	P	C24113K-B.DEP	N/A
A29002G-B.ADA	P	B24005A.ADA	P	C24113L-B.DEP	N/A
A29002H-B.ADA	P	B24005B.ADA	P	C24113M-B.DEP	N/A
A29002I-B.ADA	P	B24104A.ADA	P	C24113N-B.DEP	N/A
A29002J-B.ADA	P	B24104B.ADA	P	C24113O-B.DEP	N/A
B22001A-AB.TST	P	B24104C.ADA	P	C24113P-B.DEP	N/A
B22001B-AB.TST	P	B26002A.ADA	P	C24113Q-B.DEP	N/A
B22001C-AB.TST	P	B26005A.ADA	P	C24113R-B.DEP	N/A
B22001D-AB.TST	P	B29001A-B.ADA	P	C24113S-B.DEP	N/A
B22001E-AB.TST	P	C23001A.ADA	P	C24113T-B.DEP	N/A
B22001F-AB.TST	P	C23003A.TST	P	C24113U-B.DEP	N/A
B22001G-AB.TST	P	C24002A.ADA	P	C24113V-B.DEP	N/A
B22001H-AB.ADA	P	C24002B.ADA	P	C24113W-B.DEP	N/A
B22001I-AB.TST	P	C24002C.ADA	P	C24113X-B.DEP	N/A
B22001J-AB.TST	P	C24003A.TST	P	C24113Y-B.DEP	N/A
B22001K-AB.TST	P	C24003B.TST	P	C26002B.ADA	P
B22001L-AB.TST	P	C24003C.TST	P	C26006A-AB.ADA	P
B22001M-AB.TST	P	C24102A.ADA	P	C26008A-AB.ADA	P
B22001N-AB.TST	P	C24102B.ADA	P	C27001A-AB.ADA	P
B22003A.ADA	P	C24102C.ADA	P	C27002A-B.ADA	P
B22004A.ADA	P	C24103A.ADA	P	D29002K-B.ADA	P
B22004B.ADA	P	C24113A-B.DEP	P	E24101A-B.TST	P
B22004C.ADA	P	C24113B-B.DEP	P		

Chapter 3

A32203B-B.ADA	P	B37004G-B.ADA	P	C34001Q-B.ADA	P
A32203C-B.ADA	P	B37101A.ADA	P	C34001R-B.ADA	P
A32203D-B.ADA	P	B37201A.ADA	P	C34001T-B.ADA	P
A34008B-B.ADA	P	B37202A.ADA	P	C34002A-B.ADA	P
A38106D-B.ADA	P	B37202B.ADA	P	C34002B-B.ADA	P
A38106E-B.ADA	P	B37203A.ADA	P	C35104A.ADA	P
B32103A-AB.ADA	P	B37204A-AB.ADA	P	C35504A-AB.ADA	P
B32106A-B.ADA	P	B37205A-AB.ADA	P	C35504B-B.ADA	P
B32201A-B.ADA	P	B37301A.ADA	P	C35505A.ADA	P
B32202A-B.ADA	P	B37301B.ADA	P	C35505B.ADA	P
B32202B-B.ADA	P	B37302A-AB.ADA	P	C35508A-AB.ADA	P
B32202C-B.ADA	P	B37303A.ADA	P	C35508B-B.ADA	P
B33001A.ADA	P	B37307B-AB.ADA	P	C35702A-AB.DEP	P
B33002A.ADA	P	B37309B-AB.ADA	P	C35702B-AB.DEP	N/A
B33003A.ADA	P	B37310B-B.ADA	P	C35703A.ADA	P
B33003B-AB.ADA	P	B37311A-AB.ADA	P	C35704A-AB.ADA	P
B33003C-AB.ADA	P	B38001A.ADA	P	C35704B-AB.ADA	P
B33004A.ADA	P	B38003A-AB.ADA	P	C35704C-AB.ADA	P
B33006A-B.ADA	P	B38008A-B.ADA	P	C35704D-AB.ADA	P
B34001S-AB.ADA	P	B38008B-AB.ADA	P	C35705A-B.DEP	P
B34008A-B.ADA	P	B38101A-B.ADA	P	C35705B-B.DEP	P
B35101A.ADA	P	B38101B-AB.ADA	P	C35705C-B.DEP	P
B35301A.ADA	P	B38103A-B.ADA	P	C35705D-B.DEP	P
B35501A.ADA	P	B38103B-B.ADA	P	C35705E-B.DEP	P
B35506A.ADA	P	B38103C-B.ADA	P	C35705F-B.DEP	N/A
B35506B.ADA	P	B38103C0	C	C35705G-B.DEP	N/A
B35701A.TST	P	B38103C1	C	C35705H-B.DEP	N/A
B35709A.ADA	P	B38103C2	C	C35705I-B.DEP	N/A
B35A03A-B.ADA	P	B38103C3M	C	C35705J-B.DEP	N/A
B36101A-AB.ADA	P	B38105A-AB.ADA	P	C35705K-B.DEP	N/A
B36102A.ADA	P	B38105B-AB.ADA	W	C35705L-B.DEP	N/A
B36103A.ADA	P	B38106A-B.ADA	P	C35705M-B.DEP	N/A
B36105A-B.ADA	P	B38106B-B.ADA	P	C35705N-B.DEP	N/A
B36171A-B.ADA	P	C32107B-B.ADA	P	C35705O-B.DEP	N/A
B36171B-B.ADA	P	C32203A-B.ADA	P	C35705P-B.DEP	N/A
B36171C-AB.ADA	P	C34001A-B.ADA	P	C35705Q-B.DEP	N/A
B36171D-AB.ADA	P	C34001B-B.ADA	P	C35705R-B.DEP	N/A
B36171E-AB.ADA	P	C34001C-B.ADA	P	C35705S-B.DEP	N/A
B36171F-AB.ADA	P	C34001D-B.DEP	P	C35705T-B.DEP	N/A
B36171G-AB.ADA	P	C34001E-B.DEP	N/A	C35705U-B.DEP	N/A
B36171H-AB.ADA	P	C34001F-B.DEP	P	C35705V-B.DEP	N/A
B36171I-AB.ADA	P	C34001G-B.DEP	N/A	C35705W-B.DEP	N/A
B36201A-B.ADA	P	C34001H-B.ADA	P	C35705X-B.DEP	N/A
B37003A-AB.ADA	P	C34001I-B.ADA	P	C35705Y-B.DEP	N/A
B37004A-B.ADA	P	C34001K-B.ADA	P	C35706A-B.DEP	P
B37004B-B.ADA	P	C34001L-B.ADA	P	C35706B-B.DEP	P
B37004C-B.ADA	P	C34001M-B.ADA	P	C35706C-B.DEP	P
B37004D-B.ADA	P	C34001N-B.ADA	P	C35706D-B.DEP	P
B37004E-B.ADA	P	C34001O-B.ADA	P	C35706E-B.DEP	P
B37004F-B.ADA	P	C34001P-B.ADA	P	C35706F-B.DEP	N/A

C35706G-B.DEP	N/A	C35708G-B.DEP	N/A	C36205B.ADA	P
C35706H-B.DEP	N/A	C35708H-B.DEP	N/A	C36205C.ADA	P
C35706I-B.DEP	N/A	C35708I-B.DEP	N/A	C36205D.ADA	P
C35706J-B.DEP	N/A	C35708J-B.DEP	N/A	C36205E.ADA	P
C35706K-B.DEP	N/A	C35708K-B.DEP	N/A	C36205F.ADA	P
C35706L-B.DEP	N/A	C35708L-B.DEP	N/A	C36205G.ADA	P
C35706M-B.DEP	N/A	C35708M-B.DEP	N/A	C36205H.ADA	P
C35706N-B.DEP	N/A	C35708N-B.DEP	N/A	C36205I.ADA	P
C35706O-B.DEP	N/A	C35708O-B.DEP	N/A	C36205J.ADA	P
C35706P-B.DEP	N/A	C35708P-B.DEP	N/A	C36205K.ADA	P
C35706Q-B.DEP	N/A	C35708Q-B.DEP	N/A	C36301A-B.ADA	P
C35706R-B.DEP	N/A	C35708R-B.DEP	N/A	C36301B-AB.ADA	P
C35706S-B.DEP	N/A	C35708S-B.DEP	N/A	C36302A.ADA	P
C35706T-B.DEP	N/A	C35708T-B.DEP	N/A	C36303A.ADA	P
C35706U-B.DEP	N/A	C35708U-B.DEP	N/A	C36304A-B.ADA	P
C35706V-B.DEP	N/A	C35708V-B.DEP	N/A	C36305A-AB.ADA	P
C35706W-B.DEP	N/A	C35708W-B.DEP	N/A	C37005A.ADA	P
C35706X-B.DEP	N/A	C35708X-B.DEP	N/A	C37007A-AB.ADA	P
C35706Y-B.DEP	N/A	C35708Y-B.DEP	N/A	C37008A-B.ADA	P
C35707A-B.DEP	P	C35711A-B.ADA	P	C37008B-B.ADA	P
C35707B-B.DEP	P	C35802A-B.DEP	P	C37011A-B.ADA	P
C35707C-B.DEP	P	C35802B-B.DEP	P	C37012A-AB.ADA	P
C35707D-B.DEP	P	C35802C-B.DEP	P	C37013A-AB.ADA	P
C35707E-B.DEP	P	C35802D-B.DEP	P	C37103A-AB.ADA	P
C35707F-B.DEP	N/A	C35802E-B.DEP	P	C37105A.ADA	P
C35707G-B.DE.	N/A	C35802F-B.DEP	N/A	C37208A-B.ADA	P
C35707H-B.DEP	N/A	C35802G-B.DEP	N/A	C37208B-AB.ADA	P
C35707I-B.DEP	N/A	C35802H-B.DEP	N/A	C37209A.ADA	P
C35707J-B.DEP	N/A	C35802I-B.DEP	N/A	C37304A-AB.ADA	P
C35707K-B.DEP	N/A	C35802J-B.DEP	N/A	C37305A.ADA	P
C35707L-B.DEP	N/A	C35802K-B.DEP	N/A	C37306A.ADA	P
C35707M-B.DEP	N/A	C35802L-B.DEP	N/A	C37307A-AB.ADA	P
C35707N-B.DEP	N/A	C35802M-B.DEP	N/A	C37309A-AB.ADA	P
C35707O-B.DEP	N/A	C35802N-B.DEP	N/A	C37310A-AB.ADA	P
C35707P-B.DEP	N/A	C35802O-B.DEP	N/A	C38004A.ADA	P
C35707Q-B.DEP	N/A	C35802P-B.DEP	N/A	C38005A-B.ADA	P
C35707R-B.DEP	N/A	C35802Q-B.DEP	N/A	C38006A-B.ADA	P
C35707S-B.DEP	N/A	C35802R-B.DEP	N/A	C38007A-B.ADA	P
C35707T-B.DEP	N/A	C35802S-B.DEP	N/A	C38102A-AB.ADA	P
C35707U-B.DEP	N/A	C35802T-B.DEP	N/A	C38102B-B.ADA	P
C35707V-B.DEP	N/A	C35802U-B.DEP	N/A	C38102C-B.ADA	P
C35707W-B.DEP	N/A	C35802V-B.DEP	N/A	E36202A-B.ADA	P
C35707X-B.DEP	N/A	C35802W-B.DEP	N/A	E36202B-B.ADA	P
C35707Y-B.DEP	N/A	C35802X-B.DEP	N/A	E38104A-B.ADA	P
C35708A-B.DEP	P	C35802Y-B.DEP	N/A		
C35708B-B.DEP	P	C35904A-B.ADA	W		
C35708C-B.DEP	P	C36172A-B.ADA	P		
C35708D-B.DEP	P	C36174A-B.ADA	P		
C35708E-B.DEP	P	C36204A-B.ADA	P		
C35708F-B.DEP	N/A	C36205A.ADA	P		

Chapter 4

B41101A-B.ADA	P	B45208A-AB.ADA	P	C41303F-B.ADA	P
B41101C-AB.ADA	P	B45208B-B.ADA	P	C41303G-B.ADA	P
B41102A-AB.ADA	P	B45208C-B.ADA	P	C41303I-B.ADA	P
B41102B-B.ADA	P	B45208G-AB.ADA	P	C41303J-B.ADA	P
B41102C-B.ADA	P	B45208H-B.ADA	P	C41303K-B.ADA	P
B41201A-B.ADA	P	B45208I-B.ADA	P	C41303M-B.ADA	P
B41201C.ADA	P	B45208M-AB.ADA	P	C41303N-B.ADA	P
B41202A-B.ADA	P	B45208N-AB.ADA	P	C41303O-B.ADA	P
B41202B-AB.ADA	P	B45208S-AB.ADA	P	C41303Q-B.ADA	P
B41202C-B.ADA	P	B45208T-AB.ADA	P	C41303R-B.ADA	P
B41202D-B.ADA	P	B45261A-AB.ADA	P	C41303S-B.ADA	P
B41302A-AB.ADA	P	B45261B-AB.ADA	P	C41303U-B.ADA	P
B41302B-AB.ADA	P	B45261C-AB.ADA	P	C41303V-B.ADA	P
B42004A-B.ADA	P	B45261D-AB.ADA	P	C41303W-B.ADA	P
B43101A-B.ADA	P	B45402A.ADA	P	C41304A-B.ADA	P
B43201A-B.ADA	P	B45522A.ADA	P	C41306A-B.ADA	P
B43201B-B.ADA	P	B45533A-AB.ADA	P	C41306B-B.ADA	P
B43201C-B.ADA	P	B48001A-B.ADA	P	C41306C-B.ADA	P
B43201D-B.ADA	P	B48001B-B.ADA	P	C42005A-B.ADA	P
B43202A-B.ADA	P	B48002A-B.ADA	P	C42006A-B.ADA	P
B43202B-B.ADA	P	B48002B-B.ADA	P	C43103A-B.ADA	P
B43202C-B.ADA	P	B48002C-B.ADA	P	C43103B-B.ADA	P
B43203A-B.ADA	P	B48002D-B.ADA	P	C43107A-B.ADA	P
B43203B-B.ADA	P	B48002E-B.ADA	P	C43205A-B.ADA	P
B44001A-B.ADA	P	B48002F-B.ADA	P	C43205B-B.ADA	P
B44002A-B.ADA	P	B48002G-B.ADA	P	C43205C-B.ADA	P
B44002B-B.ADA	P	B48003A-B.ADA	P	C43205D-B.ADA	P
B44002C.ADA	P	B48003B-B.ADA	P	C43205E-B.ADA	P
B45102A-AB.ADA	P	B48003C-B.ADA	P	C43205F-B.ADA	P
B45203A.ADA	P	B48003D-B.ADA	P	C43205G-B.ADA	P
B45203B-AB.ADA	P	B48003E-B.ADA	P	C43205H-B.ADA	P
B45205A-AB.ADA	P	B4A006A-B.ADA	P	C43205I-B.ADA	P
B45206A-AB.ADA	P	B4A016A.ADA	P	C43205J-B.ADA	P
B45206B-B.ADA	P	C41101D-B.ADA	P	C43205K-B.ADA	P
B45207A-AB.ADA	P	C41103A-B.ADA	P	C43206A-B.ADA	P
B45207B-B.ADA	P	C41103B-B.ADA	P	C43207A-B.ADA	P
B45207C-B.ADA	P	C41105A-B.ADA	P	C43207B-B.ADA	P
B45207D-B.ADA	P	C41106A-B.ADA	P	C43207C-B.ADA	P
B45207G-B.ADA	P	C41107A-AB.ADA	P	C43207D-B.ADA	P
B45207H-B.ADA	P	C41201D-B.ADA	P	C43208A-B.ADA	P
B45207I-B.ADA	P	C41203A-B.ADA	P	C43208B-B.ADA	P
B45207J-B.ADA	P	C41203B-B.ADA	P	C43210A-B.ADA	P
B45207M-AB.ADA	P	C41204A.ADA	P	C43211A-B.ADA	P
B45207N-AB.ADA	P	C41205A-B.ADA	P	C43212A-B.ADA	P
B45207O-AB.ADA	P	C41206A.ADA	P	C43212C-B.ADA	P
B45207P-B.ADA	P	C41301A-B.ADA	P	C43213A-B.ADA	P
B45207S-AB.ADA	P	C41303A-B.ADA	P	C43214A-B.ADA	P
B45207T-AB.ADA	P	C41303B-B.ADA	P	C43214B-B.ADA	P
B45207U-AB.ADA	P	C41303C-B.ADA	P	C43214C-B.ADA	P
B45207V-B.ADA	P	C41303E-B.ADA	P	C43214D-B.ADA	P

COMPLETE LIST OF TESTS AND RESULTS

C43214E-B.ADA	P	C45241X-B.DEP	N/A	C45421J-B.DEP	N/A
C43214F-B.ADA	P	C45241Y-B.DEP	N/A	C45421K-B.DEP	N/A
C43215A-B.ADA	P	C45264A-B.ADA	P	C45421L-B.DEP	N/A
C43215B-B.ADA	P	C45274A-AB.ADA	P	C45421M-B.DEP	N/A
C45101A.ADA	P	C45274B-AB.ADA	P	C45421N-B.DEP	N/A
C45101B.ADA	P	C45274C-AB.ADA	P	C45421O-B.DEP	N/A
C45101C.ADA	P	C45303A-B.ADA	P	C45421P-B.DEP	N/A
C45101E.ADA	P	C45321A-B.DEP	P	C45421Q-B.DEP	N/A
C45101G-AB.ADA	P	C45321B-B.DEP	P	C45421R-B.DEP	N/A
C45101H-AB.ADA	P	C45321C-B.DEP	P	C45421S-B.DEP	N/A
C45101I.ADA	P	C45321D-B.DEP	P	C45421T-B.DEP	N/A
C45103A-AB.ADA	P	C45321E-B.DEP	P	C45421U-B.DEP	N/A
C45103B-AB.ADA	P	C45321F-B.DEP	N/A	C45421V-B.DEP	N/A
C45103C-AB.ADA	P	C45321G-B.DEP	N/A	C45421W-B.DEP	N/A
C45104A.ADA	P	C45321H-B.DEP	N/A	C45421X-B.DEP	N/A
C45105A-AB.ADA	P	C45321I-B.DEP	N/A	C45421Y-B.DEP	N/A
C45105B-B.ADA	P	C45321J-B.DEP	N/A	C45424A-B.DEP	P
C45106A.ADA	P	C45321K-B.DEP	N/A	C45424B-B.DEP	P
C45201A.ADA	P	C45321L-B.DEP	N/A	C45424C-B.DEP	P
C45201B.ADA	P	C45321M-B.DEP	N/A	C45424D-B.DEP	P
C45202A-AB.ADA	P	C45321N-B.DEP	N/A	C45424E-B.DEP	P
C45210A.ADA	P	C45321O-B.DEP	N/A	C45424F-B.DEP	N/A
C45220A.ADA	P	C45321P-B.DEP	N/A	C45424G-B.DEP	N/A
C45220B.ADA	P	C45321Q-B.DEP	N/A	C45424H-B.DEP	N/A
C45220C.ADA	P	C45321R-B.DEP	N/A	C45424I-B.DEP	N/A
C45220D.ADA	P	C45321S-B.DEP	N/A	C45424J-B.DEP	N/A
C45220E-B.ADA	P	C45321T-B.DEP	N/A	C45424K-B.DEP	N/A
C45241A-B.DEP	P	C45321U-B.DEP	N/A	C45424L-B.DEP	N/A
C45241B-B.DEP	P	C45321V-B.DEP	N/A	C45424M-B.DEP	N/A
C45241C-B.DEP	P	C45321W-B.DEP	N/A	C45424N-B.DEP	N/A
C45241D-B.DEP	P	C45321X-B.DEP	N/A	C45424O-B.DEP	N/A
C45241E-B.DEP	P	C45321Y-B.DEP	N/A	C45424P-B.DEP	N/A
C45241F-B.DEP	N/A	C45342A-AB.ADA	P	C45424Q-B.DEP	N/A
C45241G-B.DEP	N/A	C45343A-AB.ADA	P	C45424R-B.DEP	N/A
C45241H-B.DEP	N/A	C45345A-AB.ADA	P	C45424S-B.DEP	N/A
C45241I-B.DEP	N/A	C45345B-AB.ADA	P	C45424T-B.DEP	N/A
C45241J-B.DEP	N/A	C45345C-AB.ADA	P	C45424U-B.DEP	N/A
C45241K-B.DEP	N/A	C45345D-AB.ADA	P	C45424V-B.DEP	N/A
C45241L-B.DEP	N/A	C45401A.ADA	P	C45424W-B.DEP	N/A
C45241M-B.DEP	N/A	C45401B-AB.ADA	P	C45424X-B.DEP	N/A
C45241N-B.DEP	N/A	C45413A-B.ADA	P	C45424Y-B.DEP	N/A
C45241O-B.DEP	N/A	C45421A-B.DEP	P	C45505A-B.ADA	P
C45241P-B.DEP	N/A	C45421B-B.DEP	P	C45521A-B.DEP	W
C45241Q-B.DEP	N/A	C45421C-B.DEP	P	C45521B-B.DEP	W
C45241R-B.DEP	N/A	C45421D-B.DEP	P	C45521C-B.DEP	W
C45241S-B.DEP	N/A	C45421E-B.DEP	P	C45521D-B.DEP	W
C45241T-B.DEP	N/A	C45421F-B.DEP	N/A	C45521E-B.DEP	W
C45241U-B.DEP	N/A	C45421G-B.DEP	N/A	C45521F-B.DEP	W
C45241V-B.DEP	N/A	C45421H-B.DEP	N/A	C45521G-B.DEP	W
C45241W-B.DEP	N/A	C45421I-B.DEP	N/A	C45521H-B.DEP	W

C45521I-B.DEP	W	C45621L.DEP	N/A	C48008A-B.ADA	P
C45521J-B.DEP	W	C45621M.DEP	N/A	C48008B-B.ADA	P
C45521K-B.DEP	W	C45621N.DEP	N/A	C48008C-B.ADA	P
C45521L-B.DEP	W	C45621O.DEP	N/A	C48008D-B.ADA	P
C45521M-B.DEP	W	C45621P.DEP	N/A	C48009A-B.ADA	P
C45521N-B.DEP	W	C45621Q.DEP	N/A	C48009B-B.ADA	P
C45521O-B.DEP	W	C45621R.DEP	N/A	C48009C-B.ADA	P
C45521P-B.DEP	W	C45621S.DEP	N/A	C48009D-B.ADA	P
C45521Q-B.DEP	W	C45621T.DEP	N/A	C48009E-B.ADA	P
C45521R-B.DEP	W	C45621U.DEP	N/A	C48009F-B.ADA	P
C45521S-B.DEP	W	C45621V.DEP	N/A	C48009G-B.ADA	P
C45521T-B.DEP	W	C45621W.DEP	N/A	C48009H-B.ADA	P
C45521U-B.DEP	W	C45621X.DEP	N/A	C48009I-B.ADA	P
C45521V-B.DEP	W	C45621Y.DEP	N/A	C48009J-B.ADA	P
C45521W-B.DEP	W	C45621Z.DEP	N/A	C48010A-B.ADA	P
C45521X-B.DEP	W	C48004A-B.ADA	P	C48012A-B.ADA	P
C45521Y-B.DEP	W	C48004B-B.ADA	P	C4A001A.ADA	P
C45526A-B.ADA	P	C48004C-B.ADA	P	C4A003A.ADA	P
C45621A.DEP	P	C48004D-B.ADA	P	C4A010A-B.ADA	P
C45621B.DEP	P	C48004E-B.ADA	P	C4A011A.ADA	P
C45621C.DEP	P	C48004F-B.ADA	P	C4A013A.ADA	P
C45621D.DEP	P	C48005A-B.ADA	P	D4A002A-AB.ADA	P
C45621E.DEP	P	C48005B-B.ADA	P	D4A002B.ADA	P
C45621F.DEP	N/A	C48005C-B.ADA	W	D4A004A-AB.ADA	P
C45621G.DEP	N/A	C48006A-B.ADA	P	D4A004B.ADA	P
C45621H.DEP	N/A	C48006B-B.ADA	W	E43211B-B.ADA	P
C45621I.DEP	N/A	C48007A-B.ADA	P	E43212B-B.ADA	P
C45621J.DEP	N/A	C48007B-B.ADA	P		
C45621K.DEP	N/A	C48007C-B.ADA	P		

Chapter 5

A54B01A-B.ADA	P	B54A21A-B.ADA	P	B57001A-AB.ADA	P
A54B02A-B.ADA	P	B54A25A-B.ADA	P	B57001B-B.ADA	P
A55B12A-AB.ADA	P	B54A27B-AB.ADA	P	B57001C-AB.ADA	P
A55B13A-AB.ADA	P	B54A27D-AB.ADA	P	B57001D-AB.ADA	P
A55B14A-AB.ADA	P	B54B01B-B.TST	P	B58001A-AB.ADA	P
B51001A-AB.ADA	P	B54B01C-B.ADA	P	B58002A-B.ADA	P
B51003A-AB.ADA	P	B54B02B-B.ADA	P	B58002B-AB.ADA	P
B51004B-B.ADA	P	B54B02C-B.ADA	P	B58002C-AB.ADA	P
B51004C-B.ADA	P	B54B02D-B.ADA	P	B58003A-B.ADA	P
B52002A-B.ADA	P	B54B04A-AB.ADA	P	B58003B-AB.ADA	P
B52002B-AB.ADA	P	B54B04B-AB.ADA	P	B59001A-AB.ADA	P
B52002C-AB.ADA	P	B54B05A-AB.ADA	P	B59001C-AB.ADA	P
B52002D-AB.ADA	P	B55A01A-AB.ADA	P	B59001D-AB.ADA	P
B52002E-AB.ADA	P	B55A01B-AB.ADA	P	B59001E-AB.ADA	P
B52002F-B.ADA	P	B55A01C-AB.ADA	P	B59001F-AB.ADA	P
B52002G-AB.ADA	P	B55A01D-AB.ADA	P	B59001G-AB.ADA	P
B52003A-AB.ADA	P	B55A01E-AB.ADA	P	B59001H-AB.ADA	P
B52003B-AB.ADA	P	B55A01F-AB.ADA	P	B59001I-AB.ADA	P
B52003C-AB.ADA	P	B55A01G-AB.ADA	P	C51002A-AB.ADA	P
B52004A-B.ADA	P	B55A01H-AB.ADA	P	C51004A-B.ADA	P
B52004B-AB.ADA	P	B55A01I-AB.ADA	P	C52001A-B.ADA	P
B52004C-AB.ADA	P	B55A01J-AB.ADA	P	C52001B-AB.ADA	P
B52004D-AB.DEP	N/A	B55A01K-AB.ADA	P	C52001C-AB.ADA	P
B52004E-AB.DEP	P	B55A01L-AB.ADA	P	C52005A-AB.ADA	P
B52006A-AB.ADA	P	B55A01M-AB.ADA	P	C52005B-AB.ADA	P
B53001A-AB.ADA	P	B55A01N-AB.ADA	P	C52005C-AB.ADA	P
B53001B-AB.ADA	P	B55A01O-AB.ADA	P	C52005D-AB.ADA	P
B53002A-AB.ADA	P	B55A01P-AB.ADA	P	C52005E-AB.ADA	P
B53002B-AB.ADA	P	B55A01Q-AB.ADA	P	C52005F-AB.ADA	P
B53003A-AB.ADA	P	B55A01R-AB.ADA	P	C52007A-B.ADA	P
B53004A-AB.ADA	P	B55A01S-AB.ADA	P	C52008A-AB.ADA	P
B53009A-AB.ADA	P	B55A01T-AB.ADA	P	C52008B-B.ADA	P
B53009B-AB.ADA	P	B55A01U-AB.ADA	P	C52009A-B.ADA	P
B53009C-AB.ADA	P	B55A01V-AB.ADA	P	C52009B-B.ADA	P
B54A01A-AB.ADA	P	B55B01A-AB.ADA	P	C52010A-AB.ADA	P
B54A01B-AB.ADA	P	B55B01B-AB.ADA	P	C52011A-B.ADA	P
B54A01C-AB.ADA	P	B55B09B-AB.ADA	P	C52011B-AB.ADA	P
B54A01D-AB.ADA	P	B55B09C-AB.DEP	N/A	C52012A-AB.ADA	P
B54A01E-AB.ADA	P	B55B09D-AB.DEP	P	C52012B-AB.ADA	P
B54A01F-AB.ADA	P	B55B12B-B.ADA	P	C52013A-B.ADA	P
B54A01G-AB.ADA	P	B55B12C-AB.ADA	P	C52101A-AB.ADA	P
B54A01H-AB.ADA	P	B55B14B-B.ADA	P	C52102A-AB.ADA	P
B54A01I-AB.ADA	P	B55B18A-B.ADA	P	C52102B-AB.ADA	P
B54A01J-AB.ADA	P	B56001A-AB.ADA	P	C52102C-AB.ADA	P
B54A01K-AB.ADA	P	B56001C-AB.ADA	P	C52102D-AB.ADA	P
B54A01L-AB.ADA	P	B56001D-AB.ADA	P	C52103A-AB.ADA	P
B54A05A.ADA	P	B56001E-AB.ADA	P	C52103B-AB.ADA	P
B54A05B.ADA	P	B56001F-AB.ADA	P	C52103C-AB.ADA	P
B54A08A-B.ADA	P	B56001G-AB.ADA	P	C52103F-AB.ADA	P
B54A20A.ADA	P	B56001H-AB.ADA	P	C52103G-AB.ADA	P

C52103H-AB.ADA	P	C54A07A-AB.ADA	P	C57002A-AB.ADA	P
C52103K-AB.ADA	P	C54A22A-AB.ADA	P	C57003A-AB.ADA	P
C52103L-AB.ADA	P	C54A23A-B.ADA	P	C57004A-AB.ADA	P
C52103M-AB.ADA	P	C54A24A-AB.ADA	P	C57004B-AB.ADA	P
C52103P-AB.ADA	P	C54A24B.ADA	P	C57004C-AB.ADA	P
C52103Q-AB.ADA	P	C54A26A.ADA	P	C57005A-B.ADA	P
C52103R-AB.ADA	P	C54A27A-AB.ADA	P	C58004A-AB.ADA	P
C52103S-B.ADA	P	C54A41A.ADA	P	C58004B-AB.ADA	P
C52103X-B.ADA	P	C54A42A.ADA	P	C58004C-AB.ADA	P
C52104A-AB.ADA	P	C54A42B.ADA	P	C58004D-B.ADA	P
C52104B-AB.ADA	P	C54A42C.ADA	P	C58004F-AB.ADA	P
C52104C-AB.ADA	P	C54A42D.ADA	P	C58004G-AB.ADA	P
C52104F-AB.ADA	P	C54A42E.ADA	P	C58005A-AB.ADA	P
C52104G-AB.ADA	P	C54A42F.ADA	P	C58005B-AB.ADA	P
C52104H-AB.ADA	P	C54A42G.ADA	P	C58005H-AB.ADA	P
C52104K-AB.ADA	P	C55B03A-AB.ADA	P	C58006A-AB.ADA	P
C52104L-AB.ADA	P	C55B04A-AB.ADA	P	C58006B-AB.ADA	P
C52104M-AB.ADA	P	C55B05A-AB.ADA	P	C59001B-AB.ADA	P
C52104P-AB.ADA	P	C55B06A-AB.ADA	P	C59002A-AB.ADA	P
C52104Q-AB.ADA	P	C55B06B-AB.ADA	P	C59002B-AB.ADA	P
C52104R-AB.ADA	P	C55B07A-AB.DEP	N/A	C59002C-B.ADA	P
C52104X-B.ADA	P	C55B07B-AB.DEP	P	D55A03A-AB.ADA	P
C52104Y-B.ADA	P	C55B08A-B.ADA	P	D55A03B-AB.ADA	P
C53004B-B.ADA	P	C55B09A-AB.ADA	P	D55A03C-AB.ADA	P
C53005A-AB.ADA	P	C55B15A-B.ADA	P	D55A03D-AB.ADA	P
C53005B-AB.ADA	P	C55B16A-AB.DEP	P	D55A03E-AB.ADA	P
C53006A-AB.ADA	P	C55C01A-B.ADA	P	D55A03F-AB.ADA	P
C53006B-AB.ADA	P	C55C02A-AB.ADA	P	D55A03G-AB.ADA	P
C53007A-AB.ADA	P	C55C02B-AB.ADA	P	D55A03H-AB.ADA	P
C53008A-AB.ADA	P	C55C03A-AB.ADA	P	D56001B-AB.ADA	P
C54A03A.ADA	P	C55C03B-AB.ADA	P	E52103Y-B.ADA	P
C54A04A-AB.ADA	P	C55D01A-AB.ADA	P		
C54A06A-AB.ADA	P	C56002A-AB.ADA	P		

Chapter 6

A62006D-B.ADA	P	B63102A-B.ADA	P	C64103B-B.ADA	P
A63202A-AB.ADA	P	B63103A-B.ADA	P	C64103C-B.ADA	W
B61001A-AB.ADA	P	B64001A-B.ADA	P	C64103D-B.ADA	W
B61001B-AB.ADA	P	B64002A-B.ADA	P	C64103E-B.ADA	P
B61001C-AB.ADA	P	B64002C-B.ADA	P	C64103F-B.ADA	P
B61001D-AB.ADA	P	B64003A-B.ADA	P	C64104A-AB.ADA	P
B61001E-AB.ADA	P	B64004A-B.ADA	P	C64104B-AB.ADA	P
B61001F-AB.ADA	P	B64004B-B.ADA	P	C64104C-AB.ADA	P
B61001G-AB.ADA	P	B64004C-B.ADA	P	C64104D-AB.ADA	P
B61001H-AB.ADA	P	B64004D-B.ADA	P	C64104E-AB.ADA	P
B61001I-AB.ADA	P	B64004E-B.ADA	P	C64104F-AB.ADA	P
B61001J-AB.ADA	P	B64004F-B.ADA	P	C64104G-AB.ADA	P
B61001K-AB.ADA	P	B64006A-B.ADA	P	C64104H-B.ADA	P
B61001L-AB.ADA	P	B64101A-B.ADA	P	C64104I-B.ADA	P
B61001M-AB.ADA	P	B64201A-B.ADA	P	C64104J-B.ADA	P
B61001N-AB.ADA	P	B65001A-B.ADA	P	C64104K-AB.ADA	P
B61001O-AB.ADA	P	B65002A-AB.ADA	P	C64104L-AB.ADA	P
B61001P-AB.ADA	P	B65002B-AB.ADA	P	C64104M-AB.ADA	P
B61001Q-AB.ADA	P	B66001A-B.ADA	W	C64104N-B.ADA	P
B61001R-AB.ADA	P	B66001B-B.ADA	P	C64104O-B.ADA	P
B61001S-AB.ADA	P	B66001C-B.ADA	P	C64105A-AB.ADA	P
B61001T-AB.ADA	P	B67001A-B.ADA	W	C64105B-AB.ADA	P
B61001U-AB.ADA	P	B67001B-B.ADA	P	C64105C-AB.ADA	P
B61001V-AB.ADA	P	B67001C-B.ADA	P	C64105D-AB.ADA	P
B61001W-AB.ADA	P	B67001D-B.ADA	P	C64105E-AB.ADA	W
B61003A-AB.ADA	P	B67001E-B.ADA	P	C64105F-AB.ADA	W
B61006A-B.ADA	P	B67001F-B.ADA	P	C64106A-B.ADA	P
B61011A-B.ADA	P	B67001G-B.ADA	P	C64106B-B.ADA	P
B61012A-B.ADA	P	B67004A-B.ADA	W	C64106C-B.ADA	P
B62001A-AB.ADA	P	C61003B-AB.ADA	P	C64106D-B.ADA	P
B62001B-AB.ADA	P	C61008A-B.ADA	P	C64107A-B.ADA	P
B62001C-AB.ADA	P	C61009A-B.ADA	P	C64108A-B.ADA	P
B62001D-AB.ADA	P	C61010A-AB.ADA	P	C64201B-B.ADA	P
B62006B-B.ADA	P	C62002A-B.ADA	P	C64201C-B.ADA	P
B62006C-B.ADA	P	C62003A-B.ADA	P	C64202A-B.ADA	P
B62006E-B.ADA	P	C62003B-B.ADA	P	C65003A-B.ADA	P
B62006F-B.ADA	P	C62004A-AB.ADA	P	C65003B-B.ADA	P
B63001A-AB.ADA	P	C62006A-B.ADA	P	C66002A-B.ADA	P
B63001B-AB.ADA	P	C63004A-AB.ADA	P	C66002C-AB.ADA	P
B63005A-AB.ADA	P	C64002B-B.ADA	P	C66002D-AB.ADA	P
B63005B-AB.ADA	P	C64004G-B.ADA	P	C66002E-AB.ADA	P
B63005C-AB.ADA	P	C64005A-B.ADA	P	C66002F-AB.ADA	P
B63009A-B.ADA	P	C64005B-B.ADA	P	C66002G-B.ADA	P
B63009B-B.ADA	P	C64005C-B.ADA	P	C67002A-B.ADA	P
B63009C-B.ADA	P	C64005D-B.ADA	P	C67002B-B.ADA	P
B63009C0	C	C64005DOM	C	C67002C-B.ADA	P
B63009C1	C	C64005DA	C	C67002D-B.ADA	P
B63009C2	C	C64005DB	C	C67002E-B.ADA	P
B63009C3	C	C64005DC	C	C67003A-B.ADA	P
B63010A-AB.ADA	P	C64103A-B.ADA	P	C67003B-B.ADA	P

C67003C-AB.ADA	P	D64005FOM	C	D64005GD	C
C67003D-B.ADA	P	D64005FA	C	D64005GE	C
C67003E-AB.ADA	P	D64005FB	C	D64005GF	C
C67005A-B.ADA	P	D64005FC	C	D64005GG	C
C67005B-B.ADA	P	D64005FD	C	D64005GH	C
C67005C-B.ADA	P	D64005FE	C	D64005GI	C
C67005D-B.ADA	P	D64005FF	C	D64005GJ	C
D64005E-B.ADA	P	D64005FG	C	D64005GK	C
D64005EOM	C	D64005FH	C	D64005GL	C
D64005EA	C	D64005FI	C	D64005GM	C
D64005EB	C	D64005FJ	C	D64005GN	C
D64005EC	C	D64005G-B.ADA	P	D64005GO	C
D64005ED	C	D64005GOM	C	D64005GP	C
D64005EE	C	D64005GA	C	D64005GQ	C
D64005EF	C	D64005GB	C		
D64005F-B.ADA	P	D64005GC	C		

Chapter 7

A71002A-AB.ADA	P	B71001Q-AB.ADA	P	B74105A-B.ADA	P
A71004A-AB.ADA	P	B71001R-AB.ADA	P	B74105C-B.ADA	P
A72001A-AB.ADA	P	B71001T-AB.ADA	P	B74201A-AB.ADA	P
A73001I-AB.ADA	P	B71001U-AB.ADA	P	B74205A-B.ADA	P
A73001J-AB.ADA	P	B71001V-AB.ADA	P	B74205B-B.ADA	P
A74006A-AB.ADA	P	B71001W-AB.ADA	P	B74207A-B.ADA	W
A74105B-B.ADA	P	B71002B-AB.ADA	P	B74301A-B.ADA	P
A74106A-AB.ADA	P	B73001A-AB.ADA	P	B74304A-B.ADA	P
A74106B-AB.ADA	P	B73001B-AB.ADA	P	B74304B-B.ADA	P
A74106C-AB.ADA	P	B73001C-B.ADA	P	B74304C-B.ADA	P
A74205E-B.ADA	P	B73001D-B.ADA	P	B74401A-B.ADA	P
A74205F-B.ADA	P	B73001E-AB.ADA	P	B74401B-B.ADA	P
B71001A-AB.ADA	P	B73001F-AB.ADA	P	B74409A-B.ADA	P
B71001B-AB.ADA	P	B73001G-B.ADA	P	C72001B-AB.ADA	P
B71001C-AB.ADA	P	B73001H-B.ADA	P	C73002A-B.ADA	P
B71001D-AB.ADA	P	B73006A-AB.ADA	P	C74206A-B.ADA	P
B71001E-AB.ADA	P	B74001A-AB.ADA	P	C74207B-B.ADA	P
B71001F-AB.ADA	P	B74001B-AB.ADA	P	C74209A-AB.ADA	P
B71001G-AB.ADA	P	B74003A-B.ADA	P	C74210A-AB.ADA	P
B71001H-AB.ADA	P	B74101A-B.ADA	P	C74211A-B.ADA	P
B71001I-AB.ADA	P	B74103A-B.ADA	P	C74211B-B.ADA	P
B71001J-AB.ADA	P	B74103B-B.ADA	P	C74302A-B.ADA	P
B71001K-AB.ADA	P	B74103C-B.ADA	P	C74305A-B.ADA	P
B71001L-AB.ADA	P	B74103D-B.ADA	P	C74305B-B.ADA	P
B71001M-AB.ADA	P	B74103E-B.ADA	P	C74402A-B.ADA	P
B71001N-AB.ADA	P	B74103F-B.ADA	W	C74402B-B.ADA	P
B71001O-AB.ADA	P	B74103G-B.ADA	P	C74409B-B.ADA	P
B71001P-AB.ADA	P	B74104A-B.ADA	P		

CHAPTER 8

A83A02A.ADA	P	B86001BK-B.ADA	P	C86002A1	C
A83A02B.ADA	P	B86001BL-B.ADA	P	C86002A2M	C
A83A06A-B.ADA	P	B86001BM-B.ADA	P	C86002B.ADA	P
A83C01C.ADA	P	B86001BO-B.ADA	P	C86002B1	C
A83C01D.ADA	P	B86001BU-B.ADA	P	C86002B2M	C
A83C01E.ADA	P	B86001BV-B.ADA	P	C86003A-B.ADA	P
A83C01F.ADA	P	B86001BW-B.ADA	P	C87A05A-B.ADA	P
A83C01G.ADA	P	B86001BX-B.ADA	P	C87A05B-B.ADA	P
A83C01H.ADA	P	B86001COM-AB.DEP	P	C87B02A-B.ADA	P
A83C01I.ADA	P	B86001CP-AB.DEP	P	C87B02B-B.ADA	P
A83C01J.ADA	P	B86001CQ-AB.DEP	N/A	C87B03A-B.ADA	P
A85007D-B.ADA	P	B86001CR-AB.DEP	P	C87B04A-B.ADA	P
A85013B-B.ADA	P	B86001CS-AB.DEP	N/A	C87B04B-B.ADA	P
B83A01A-AB.ADA	P	B86001DOM-AB.TST	P	C87B04C-B.ADA	P
B83A01B-B.ADA	P	B86001DT-AB.TST	P	C87B05A-B.ADA	P
B83A01C.ADA	P	B87B23B-B.ADA	P	C87B06A-B.ADA	P
B83A05A-AB.ADA	P	B87B48C-B.ADA	P	C87B07A-B.ADA	P
B83A06B-B.ADA	P	C83B02A.ADA	P	C87B07B-B.ADA	P
B83A06H-B.ADA	P	C83B02B.ADA	P	C87B07C-B.ADA	P
B83B01A-AB.ADA	P	C83C01B.ADA	P	C87B07D-B.ADA	P
B83B02C.ADA	P	C83E02A.ADA	P	C87B07E-B.ADA	P
B83C01A-AB.ADA	P	C83E02B.ADA	P	C87B08A-B.ADA	P
B83C02A.ADA	P	C83E03A.ADA	P	C87B09A-B.ADA	P
B83E02C-B.ADA	P	C83E04A.ADA	P	C87B09B-B.ADA	P
B83F02A.ADA	P	C83F01A.ADA	P	C87B09C-B.ADA	P
B83F02B.ADA	P	C83F01B.ADA	P	C87B10A-B.ADA	P
B83F04A-AB.ADA	P	C83F01C.ADA	P	C87B11A-B.ADA	P
B84001A-AB.ADA	P	C83F01C0	C	C87B11B-B.ADA	P
B84002B-B.ADA	P	C83F01C1	C	C87B13A-B.ADA	P
B84004A-B.ADA	P	C83F01C2M	C	C87B14A-B.ADA	P
B84006A-B.ADA	P	C83F01D.ADA	P	C87B14B-B.ADA	P
B85007B-B.ADA	P	C83F01DOM.ADA	C	C87B14C-B.ADA	P
B85007C-B.ADA	P	C83F01D1.ADA	C	C87B14D-B.ADA	P
B85012A-B.ADA	P	C83F03A.ADA	P	C87B15A-B.ADA	P
B85013C-B.ADA	P	C83F03B.ADA	P	C87B16A-B.ADA	P
B85015A-B.ADA	P	C83F03C.ADA	P	C87B17A-B.ADA	P
B86001A-AB.ADA	P	C83F03C0	C	C87B18A-B.ADA	P
B86001A0	C	C83F03C1	C	C87B18B-B.ADA	P
B86001A1M	C	C83F03C2M	C	C87B19A-B.ADA	P
B86001BOM-B.ADA	P	C83F03D.ADA	P	C87B23A-B.ADA	P
B86001BA-B.ADA	P	C83F03DOM	C	C87B24A-B.ADA	P
B86001BB-B.ADA	P	C83F03D1	C	C87B24B-B.ADA	P
B86001BC-B.ADA	P	C84002A-B.ADA	P	C87B26B-B.ADA	P
B86001BD-B.ADA	P	C85007A-B.ADA	P	C87B27A-B.ADA	P
B86001BE-B.ADA	P	C85007E-B.ADA	P	C87B28A-B.ADA	P
B86001BF-B.ADA	P	C85013A-B.ADA	P	C87B29A-B.ADA	P
B86001BG-B.ADA	P	C86001E-B.ADA	P	C87B30A-B.ADA	P
B86001BH-B.ADA	P	C86001F-B.DEP	N/A	C87B31A-B.ADA	P
B86001BI-B.ADA	P	C86002A.ADA	P	C87B32A-B.ADA	P
B86001BJ-B.ADA	P	C86002A0	C	C87B33A-B.ADA	P

COMPLETE LIST OF TESTS AND RESULTS

C87B34A-B.ADA	P	C87B37E-B.ADA	P	C87B45C-B.ADA	P
C87B34B-B.ADA	P	C87B37F-B.ADA	P	C87B47A-B.ADA	P
C87B34C-B.ADA	P	C87B38A-B.ADA	P	C87B48A-B.ADA	P
C87B35A-B.ADA	P	C87B39A-B.ADA	P	C87B48B-B.ADA	P
C87B35B-B.ADA	P	C87B40A-B.ADA	P	C87B54A-B.ADA	P
C87B35C-B.ADA	P	C87B41A-B.ADA	P	C87B57A-B.ADA	P
C87B37A-B.ADA	P	C87B42A-B.ADA	P	C87B62A-B.DEP	P
C87B37B-B.ADA	P	C87B43A-B.ADA	P	C87B62B-B.DEP	P
C87B37C-B.ADA	P	C87B44A-B.ADA	P	C87B62C-B.DEP	P
C87B37D-B.ADA	P	C87B45A-B.ADA	P		

Chapter 9

A91002M-B.ADA	P	B950ADA-B.ADA	P	C910BDA-B.ADA	P
A95005A.ADA	P	B950AFA-B.ADA	P	C910BDB-B.ADA	P
A97106A-AB.ADA	P	B950AHA-B.ADA	P	C910BDC-B.ADA	P
B91001A-AB.ADA	P	B950AJA-B.ADA	P	C92002A.ADA	P
B91001B-AB.ADA	P	B950BAA-B.ADA	P	C92003A.ADA	P
B91001C-AB.ADA	P	B950DHA-B.ADA	P	C920AJA-B.ADA	P
B91001D-AB.ADA	P	B96002A-B.ADA	P	C920BAA-B.ADA	P
B91001E-AB.ADA	P	B96003A-B.ADA	P	C920BBA-B.ADA	P
B91001F-AB.ADA	P	B97101A-AB.ADA	P	C920BIA-B.ADA	P
B91001G-B.ADA	P	B97101B-AB.ADA	P	C93001A-B.ADA	P
B91002A-B.ADA	P	B97101C-AB.ADA	P	C93002A-B.ADA	P
B91002B-B.ADA	P	B97101D-AB.ADA	P	C93003A-B.ADA	P
B91002C-B.ADA	P	B97101E-AB.ADA	P	C93005A-B.ADA	W
B91002D-B.ADA	P	B97102A-AB.ADA	P	C93005B-B.ADA	W
B91002E-B.ADA	P	B97102B-AB.ADA	P	C93005C-B.ADA	W
B91002F-B.ADA	P	B97102C-AB.ADA	P	C93006A-AB.ADA	P
B91002G-B.ADA	P	B97102D-AB.ADA	P	C93007B-B.ADA	W
B91002H-B.ADA	P	B97102E-AB.ADA	P	C930ABA-B.ADA	P
B91002I-B.ADA	P	B97102F-AB.ADA	P	C930AEA-B.ADA	P
B91002J-B.ADA	P	B97102G-AB.ADA	P	C930AFA-B.ADA	P
B91002K-B.ADA	P	B97102H-AB.ADA	P	C930AJA-B.ADA	P
B91002L-B.ADA	P	B97102I-AB.ADA	P	C930BAA-B.ADA	P
B91003A-AB.ADA	P	B97103A-AB.ADA	P	C94001A-B.ADA	P
B91004A-B.ADA	P	B97103B-AB.ADA	P	C94002A-B.ADA	P
B910ABA-B.ADA	P	B97103D-AB.ADA	P	C94002B-B.ADA	P
B910ACA-B.ADA	P	B97103E-AB.ADA	P	C94003A-B.ADA	P
B910AEA-B.ADA	P	B97104A-AB.ADA	P	C94004A-B.ADA	P
B910BCA-B.ADA	P	B97104B-AB.ADA	P	C94004B-B.ADA	P
B920ACA-B.ADA	P	B97104C-AB.ADA	P	C94004C-B.ADA	P
B920BDA-B.ADA	P	B97104D-AB.ADA	P	C94005A-B.ADA	P
B920BJA-B.ADA	P	B97104E-AB.ADA	P	C94005B-B.ADA	P
B95001A.ADA	P	B97104F-AB.ADA	P	C94006A-B.ADA	P
B95001B-AB.ADA	P	B97104G-AB.ADA	P	C94007A-B.ADA	P
B95002A.ADA	P	B97107A-AB.ADA	P	C94007B-B.ADA	P
B95004A-AB.ADA	P	B97108A-AB.ADA	P	C94020A-B.ADA	P
B95004B-AB.ADA	P	B97108B-AB.ADA	P	C94021A-B.ADA	P
B95006A.ADA	P	B97109A-AB.ADA	P	C940ABA-B.ADA	P
B95006B-AB.ADA	P	B97110A-AB.ADA	P	C940ACA-B.ADA	P
B95006C-AB.ADA	P	B97110B-AB.ADA	P	C940ACB-B.ADA	P
B95006D-AB.ADA	P	B97111A-AB.ADA	P	C940ADA-B.ADA	P
B95007A-AB.ADA	P	B99001A-AB.ADA	P	C940AGA-B.ADA	P
B95007B-AB.ADA	P	B99001B-B.ADA	P	C940AGB-B.ADA	P
B95020A-B.ADA	P	B99002A-B.ADA	P	C940AHA-B.ADA	P
B95020B-B.ADA	P	B99002B-B.ADA	P	C940AIA-B.ADA	P
B95020B0	C	B99002C-B.ADA	P	C940BAA-B.ADA	P
B95020B1	C	B99003A-AB.ADA	P	C940BBA-B.ADA	P
B95020B2M	C	B9A001A-AB.ADA	P	C95008A-AB.ADA	P
B950ABA-B.ADA	P	B9A001B-AB.ADA	P	C95009A-B.ADA	P
B950ABB-B.ADA	P	C900ACA-B.ADA	P	C95009B.ADA	P
B950ACA-B.ADA	P	C910AHA-B.ADA	P	C95010A.ADA	P

COMPLETE LIST OF TESTS AND RESULTS

C95011A.ADA	P	C96005A-B.ADA	P	C97203A-AB.ADA	P
C95012A-B.ADA	P	C96005B-B.TST	N/A	C97203B-AB.ADA	P
C95013A-B.ADA	P	C96005C-B.TST	P	C97204A-B.ADA	P
C95021A-B.ADA	P	C96005D-B.ADA	P	C97303A-AB.ADA	P
C95022A-B.ADA	P	C96005E-B.ADA	P	C97303B-AB.ADA	P
C95022B-B.ADA	P	C96006A-B.ADA	P	C97304A-B.ADA	P
C950ACB-B.ADA	P	C96007A-B.ADA	P	C9A003A-B.ADA	P
C950BGA-B.ADA	P	C96008A-B.ADA	P	C9A004A-B.ADA	P
C950BHA-B.ADA	P	C96008B-B.ADA	P	C9A005A-B.ADA	P
C950BJA-B.ADA	P	C97113A-B.ADA	P	C9A006A-B.ADA	P
C950CAA-B.ADA	P	C97114A-B.ADA	P	C9A007A-B.ADA	P
C950CBA-B.ADA	P	C97115A-B.ADA	P	C9A009A-B.ADA	P
C950CHA-B.ADA	P	C97201A-AB.ADA	P	C9A009B-B.ADA	P
C950CHC-B.ADA	P	C97201D-AB.ADA	P	C9A009C-B.ADA	P
C950DEA-B.ADA	P	C97201E-AB.ADA	P	C9A009D-B.ADA	P
C950DEB-B.ADA	P	C97201G-AB.ADA	P	C9A009E-B.ADA	P
C950DGA-B.ADA	P	C97201H-AB.ADA	P	C9A009F-B.ADA	P
C96001A-B.ADA	P	C97201X-AB.ADA	P	C9A009G-B.ADA	P
C96004A-B.ADA	P	C97202A-AB.ADA	P	C9A009H-B.ADA	P

Chapter 10

BA1011B-B.ADA	P	BA1101B3	C	BA3001E-AB.ADA	P
BA1011B0M	C	BA1101B4	C	BA3001E0M	C
BA1011B1	C	BA1101C-B.ADA	P	BA3001E1	C
BA1011B2	C	BA1101C0	C	BA3001E2	C
BA1011B3	C	BA1101C1	C	BA3001E3	C
BA1011B4	C	BA1101C2M	C	BA3001F-AB.ADA	P
BA1011B5	C	BA1101C3	C	BA3001F0M	C
BA1011B6	C	BA1101C4	C	BA3001F1	C
BA1011B7	C	BA1101C5	C	BA3001F2	C
BA1011B8	C	BA1101D-AB.ADA	P	BA3001F3	C
BA1011C-B.ADA	P	BA1101E-B.ADA	P	BA3006A-B.ADA	P
BA1011C0M	C	BA1101F-B.ADA	P	BA3006A0	C
BA1011C1	C	BA1101G-B.ADA	P	BA3006A1	C
BA1011C2	C	BA1101H-B.ADA	P	BA3006A2	C
BA1011C3	C	BA1101H0	C	BA3006A3	C
BA1011C4	C	BA1101H1M	C	BA3006A4	C
BA1011C5	C	BA2001A-AB.ADA	P	BA3006A5	C
BA1011C6	C	BA2001B-AB.ADA	P	BA3006A6M	C
BA1011C7	C	BA2001C-AB.ADA	P	BA3006B-B.ADA	P
BA1011C8	C	BA2001D-AB.ADA	P	BA3006B0	C
BA1020A-B.ADA	P	BA2001E-AB.ADA	W	BA3006B1	C
BA1020A0M	C	BA2001E0M	W	BA3006B2	C
BA1020A1	C	BA2001E1	W	BA3006B3	C
BA1020A2	C	BA2001E	W	BA3006B4M	C
BA1020A3	C	BA2001F-AB.ADA	P	BA3007A-B.ADA	P
BA1020A4	C	BA2001F0M	C	BA3007A0	C
BA1020A5	C	BA2001F1	C	BA3007A1	C
BA1020A6	C	BA2001F2	C	BA3007A2	C
BA1020A7	C	BA2001G-AB.ADA	P	BA3007A3	C
BA1020A8	C	BA2001G0M	C	BA3007A4	C
BA1020B-B.ADA	P	BA2001G1	C	BA3007A5M	C
BA1020B0	C	BA2003B-AB.ADA	P	BA3007B-B.ADA	P
BA1020B1	C	BA2003B0M	C	BA3007B0	C
BA1020B2	C	BA2003B1	C	BA3007B1	C
BA1020B3	C	BA2013A-B.ADA	P	BA3007B2	C
BA1020B4	C	BA2013B-B.ADA	P	BA3007B3	C
BA1020B5	C	BA3001A-AB.ADA	P	BA3007B4	C
BA1020B6M	C	BA3001A0M	C	BA3007B5	C
BA1020C-B.ADA	P	BA3001A1	C	BA3007B6	C
BA1020C0M	C	BA3001A2	C	BA3007B7	C
BA1020C1	C	BA3001A3	C	BA3007B8M	C
BA1020C2	C	BA3001B.ADA	P	BA3008A-B.ADA	P
BA1020C3	C	BA3001B0M	C	BA3008A0	C
BA1020C4	C	BA3001B1	C	BA3008A1	C
BA1020C5	C	BA3001C-AB.ADA	P	BA3008A2	C
BA1101A-AB.ADA	P	BA3001C0M	C	BA3008A3	C
BA1101B-B.ADA	P	BA3001C1	C	BA3008A4	C
BA1101B0M	C	BA3001D-AB.ADA	P	BA3008A5M	C
BA1101B1	C	BA3001D0M	C	BA3008B-B.ADA	P
BA1101B2	C	BA3001D1	C	BA3008B0	C

COMPLETE LIST OF TESTS AND RESULTS

BA3008B1	C	CA1011A6M	W	CA1108A-B.ADA	W
BA3008B2	C	CA1012A-B.DEP	P	CA1108B-B.ADA	W
BA3008B3	C	CA1012A0	C	CA2001H-B.ADA	P
BA3008B4	C	CA1012A1	C	CA2001H0	C
BA3008B5	C	CA1012A2	C	CA2001H1	C
BA3008B6M	C	CA1012A3	C	CA2001H2	C
BA3013A-B.ADA	P	CA1012A4M	C	CA2001H3M	C
BA3013A0	C	CA1012B-B.ADA	P	CA2002A-B.ADA	P
BA3013A1	C	CA1012B0	C	CA2002A0M	C
BA3013A2	C	CA1012B2	C	CA2002A1	C
BA3013A3	C	CA1012B4M	C	CA2002A2	C
BA3013A4	C	CA1013A-B.ADA	P	CA2003A-AB.ADA	P
BA3013A5	C	CA1013A0	C	CA2003A0M	C
BA3013A6	C	CA1013A1	C	CA2003A1	C
BA3013A7M	C	CA1013A2	C	CA2004A-AB.ADA	P
CA1002A-B.ADA	P	CA1013A3	C	CA2004A0M	C
CA1002A0	C	CA1013A4	C	CA2004A1	C
CA1002A1	C	CA1013A5	C	CA2004A2	C
CA1002A2	C	CA1013A6M	C	CA2004A3	C
CA1002A3M	C	CA1014A-AB.ADA	P	CA2004A4	C
CA1002A4	C	CA1014A0M	C	CA2007A-AB.ADA	P
CA1002A5	C	CA1014A1	C	CA2007A0M	C
CA1002A6	C	CA1014A2	C	CA2007A1	C
CA1002A7	C	CA1014A3	C	CA2007A2	C
CA1002A8	C	CA1022A-B.ADA	P	CA2007A3	C
CA1002A9	C	CA1022A0	C	CA2008A-B.ADA	P
CA1003A-AB.ADA	P	CA1022A1	C	CA2008A0M	C
CA1003B-AB.ADA	W	CA1022A2	C	CA2008A1	C
CA1004A-AB.ADA	P	CA1022A3	C	CA2008A2	C
CA1005A-AB.ADA	P	CA1022A4	C	CA2009A-B.DEP	P
CA1006A-AB.ADA	P	CA1022A5	C	CA2009B-B.DEP	W
CA1007A-AB.ADA	P	CA1022A6M	C	CA2009C-B.DEP	P
CA1007A0	C	CA1102A-B.ADA	P	CA2009C0M	C
CA1007A1M	C	CA1102A0	C	CA2009C1	C
CA1008A-AB.ADA	P	CA1102A1	C	CA2009D-B.DEP	P
CA1008A0	C	CA1102A2M	C	CA2009E-B.DEP	W
CA1008A1M	C	CA1105A-B.ADA	P	CA2009F-B.DEP	W
CA1009A-AB.ADA	P	CA1105A0	C	CA2009F0M	W
CA1009A0	C	CA1105A1M	C	CA2009F1	W
CA1009A1	C	CA1105B-B.ADA	P	CA3002A-B.ADA	P
CA1009A2	C	CA1105B0	C	CA3002A0	C
CA1009A3	C	CA1105B1	C	CA3002A1	C
CA1009A4M	C	CA1105B2	C	CA3002A2M	C
CA1011A-B.ADA	W	CA1105B3M	C	CA3002A3	C
CA1011A0	W	CA1105B4	C	CA3006C-B.ADA	P
CA1011A1	W	CA1105B5	C	CA3006C0	C
CA1011A2	W	CA1107A.ADA	P	CA3006C1	C
CA1011A3	W	CA1107A0	C	CA3006C2	C
CA1011A4	W	CA1107A1M	C	CA3006C3	C
CA1011A5	W	CA1107A2	C	CA3006C4	C

CA3006C5M	C	CA5002B6	C	LA3004A2	C
CA3006D-B.ADA	P	CA5002B7M	C	LA3004A3	C
CA3006D0	C	CA5003A-B.ADA	P	LA3004A4	C
CA3006D1	C	CA5003A0	C	LA3004A5	C
CA3006D2	C	CA5003A1	C	LA3004A6M	C
CA3006D3M	C	CA5003A2	C	LA3004B-B.ADA	P
CA3006E-B.ADA	P	CA5003A3	C	LA3004B0	C
CA3006E0	C	CA5003A4	C	LA3004B1	C
CA3006E1	C	CA5003A5	C	LA3004B2	C
CA3006E2	C	CA5003A6M	C	LA3004B3	C
CA3006E3	C	CA5003B-B.ADA	P	LA3004B4	C
CA3006E4	C	CA5003B0	C	LA3004B5	C
CA3006E5	C	CA5003B1	C	LA3004B6M	C
CA3006E6M	C	CA5003B2	C	LA5001A-B.ADA	P
CA5002A-B.ADA	P	CA5003B3	C	LA5001A0	C
CA5002B-B.ADA	P	CA5003B4	C	LA5001A1	C
CA5002B0	C	CA5003B5M	C	LA5001A2	C
CA5002B1	C	CA5004A-B.ADA	P	LA5001A3	C
CA5002B2	C	CA5004B-B.ADA	P	LA5001A4	C
CA5002B3	C	LA3004A-AB.ADA	P	LA5001A5	C
CA5002B4	C	LA3004A0	C	LA5001A6	C
CA5002B5	C	LA3004A1	C	LA5001A7M	C

COMPLETE LIST OF TESTS AND RESULTS

Chapter 11

BB2001A-AB.ADA	P	CB1003A-AB.ADA	P	CB4003A-AB.ADA	P
BB2002A-AB.ADA	P	CB1004A-AB.ADA	P	CB4004A-B.ADA	P
BB2003A-AB.ADA	P	CB2004A-B.ADA	P	CB4005A-AB.ADA	P
BB2003B-AB.ADA	P	CB2005A-B.ADA	P	CB4006A-B.ADA	P
BB2003C-AB.ADA	P	CB2006A-AB.ADA	P	CB4008A-AB.ADA	P
BB3001A-B.ADA	P	CB2007A-AB.ADA	P	CB4009A-AB.ADA	P
BB3002A-AB.ADA	P	CB3003A-B.ADA	P	CB5001A-B.ADA	P
BB3005A-AB.ADA	P	CB3004A-AB.ADA	P	CB5001B-B.ADA	P
CB1001A-B.ADA	P	CB4001A-AB.ADA	P		
CB1002A-B.ADA	P	CB4002A-AB.ADA	P		

Chapter 12

BC1001A-B.ADA	P	BC2001B-AB.ADA	P	BC3205D1M	W
BC1002A-B.ADA	P	BC2001C-AB.ADA	P	BC3205D2	W
BC1008A-AB.ADA	P	BC20ABA-B.ADA	P	BC3205E-B.ADA	P
BC1008B-AB.ADA	P	BC3002A-AB.ADA	P	BC3205F-B.ADA	P
BC1008C-AB.ADA	P	BC3002B-AB.ADA	P	BC3220B-B.ADA	W
BC1009A-AB.ADA	P	BC3002C-AB.ADA	P	BC32ABA-B.ADA	P
BC1011A-AB.ADA	P	BC3002D-AB.ADA	P	BC32ADA-B.ADA	P
BC1011B-AB.ADA	P	BC3002E-AB.ADA	P	BC3301A-AB.ADA	P
BC1012A-AB.ADA	P	BC3003A-AB.ADA	P	BC3301B-AB.ADA	P
BC1013A-B.ADA	W	BC3003B-AB.ADA	P	BC3302A-AB.ADA	P
BC10ABA-B.ADA	P	BC3005A-AB.ADA	P	BC3302B-AB.ADA	P
BC10ABB-B.ADA	P	BC3006A-AB.ADA	P	BC3303A-AB.ADA	P
BC10ACA-B.ADA	P	BC3009A-B.ADA	P	BC3304A-AB.ADA	P
BC10ADA-B.ADA	P	BC3009B-B.ADA	P	BC33ABA-B.ADA	P
BC10AEA-B.ADA	P	BC3009C-B.ADA	P	BC33ACA-B.ADA	P
BC10AEB-B.ADA	P	BC3011B-B.ADA	P	BC33ADA-B.ADA	P
BC10AEC-B.ADA	P	BC3011C-AB.ADA	P	BC33AEA-B.ADA	P
BC10AED-B.ADA	P	BC3013A-AB.ADA	P	BC3401A-AB.ADA	P
BC10AFA-B.ADA	P	BC3018A-B.ADA	P	BC3401B-AB.ADA	P
BC10AGA-B.ADA	P	BC30ABA-B.ADA	P	BC3402A-AB.ADA	P
BC1101A-AB.ADA	P	BC30ACA-B.ADA	P	BC3402B-AB.ADA	P
BC1102A-B.ADA	P	BC3101A-B.ADA	P	BC3403A-AB.ADA	P
BC1103A-B.ADA	P	BC3101B-B.ADA	P	BC3403B-AB.ADA	P
BC1104A-B.ADA	P	BC3102A-B.ADA	P	BC3403C-AB.ADA	P
BC1104B-B.ADA	P	BC3102B-B.ADA	P	BC3404A-AB.ADA	P
BC1106A-AB.ADA	P	BC3103A-AB.ADA	P	BC3404B-B.ADA	P
BC1107A-B.ADA	P	BC3103B-AB.ADA	P	BC3404C-AB.ADA	P
BC11ABA-B.ADA	P	BC31ABA-B.ADA	P	BC3404D-AB.ADA	P
BC11ACA-B.ADA	P	BC31ACA-B.ADA	P	BC3404E-AB.ADA	P
BC1201A-AB.ADA	P	BC31ADA-B.ADA	P	BC3404F-AB.ADA	P
BC1201B-AB.ADA	P	BC3201A-B.ADA	P	BC3405A-AB.ADA	P
BC1201C-AB.ADA	P	BC3201B-AB.ADA	P	BC3405B-B.ADA	W
BC1201D-AB.ADA	P	BC3201C-B.ADA	P	BC3405D-AB.ADA	P
BC1202A-AB.ADA	P	BC3202A-B.ADA	P	BC3405E-AB.ADA	P
BC1202B-AB.ADA	P	BC3202B-B.ADA	P	BC3405F-AB.ADA	P
BC1202C-AB.ADA	P	BC3202C-B.ADA	P	BC3501A-AB.ADA	P
BC1202D-AB.ADA	P	BC3203B-B.ADA	P	BC3501B-AB.ADA	P
BC1203A-AB.ADA	P	BC3204A-B.ADA	W	BC3501C-AB.ADA	P
BC1207A-B.ADA	P	BC3204B-B.ADA	W	BC3501D-AB.ADA	P
BC1226A-B.ADA	P	BC3204C-B.ADA	W	BC3501E-AB.ADA	P
BC12ABA-B.ADA	P	BC3204C0	W	BC3501F-AB.ADA	P
BC12ACA-B.ADA	P	BC3204C1M	W	BC3501G-AB.ADA	P
BC12ACB-B.ADA	P	BC3204C2	W	BC3501H-AB.ADA	P
BC1303A-AB.ADA	P	BC3204D-B.ADA	W	BC3501I-AB.ADA	P
BC1303B-AB.ADA	P	BC3204E-B.ADA	P	BC3501J-AB.ADA	P
BC1303C-AB.ADA	P	BC3205A-B.ADA	W	BC3501K-AB.ADA	P
BC1303D-AB.ADA	P	BC3205B-B.ADA	W	BC3502A-AB.ADA	P
BC1303E-AB.ADA	P	BC3205C-B.ADA	W	BC3502B-AB.ADA	P
BC1306A-B.ADA	P	BC3205D-B.ADA	W	BC3502C-AB.ADA	P
BC13ABA-B.ADA	P	BC3205D0	W	BC3502D-AB.ADA	P

COMPLETE LIST OF TESTS AND RESULTS

BC3502E-AB.ADA	P	CC1305B-AB.ADA	P	CC3407A-AB.ADA	P
BC3502F-AB.ADA	P	CC1307A-AB.ADA	P	CC3407B-AB.ADA	P
BC3502G-AB.ADA	P	CC1308A-AB.ADA	P	CC3407C-AB.ADA	P
BC3502H-AB.ADA	P	CC1310A-AB.ADA	P	CC3407D-AB.ADA	P
BC3502I-AB.ADA	P	CC2002A-AB.ADA	P	CC3407E-AB.ADA	P
BC3502J-AB.ADA	P	CC3004A-B.ADA	P	CC3407F-AB.ADA	P
BC3502K-AB.ADA	P	CC3007A-AB.ADA	P	CC3408A-AB.ADA	P
BC3502L-AB.ADA	P	CC3011A-B.ADA	P	CC3408B-AB.ADA	P
BC3502M-AB.ADA	P	CC3011D-B.ADA	P	CC3408C-AB.ADA	P
BC3502N-AB.ADA	P	CC3012A-AB.ADA	P	CC3408D-B.ADA	P
BC3502O-AB.ADA	P	CC3120A-AB.ADA	P	CC3504A-B.ADA	P
BC3503A-B.ADA	W	CC3120B-B.ADA	P	CC3504B-B.ADA	P
BC3503B-B.ADA	P	CC3125A-B.ADA	P	CC3504C-B.ADA	P
BC3503C-B.ADA	P	CC3203A-B.ADA	P	CC3504D-B.ADA	P
BC3503D-B.ADA	P	CC3208A-AB.ADA	P	CC3504E-B.ADA	P
BC3503F-B.ADA	P	CC3208B-AB.ADA	P	CC3504F-B.ADA	P
CC1004A-AB.ADA	P	CC3305A-AB.ADA	P	CC3504G-B.ADA	P
CC1010A-AB.ADA	P	CC3305B-AB.ADA	P	CC3504H-B.ADA	P
CC1010B-AB.ADA	P	CC3305C-AB.ADA	P	CC3504I-B.ADA	P
CC1204A-B.ADA	P	CC3305D-AB.ADA	P	CC3504J-B.ADA	P
CC1220A-B.ADA	P	CC3406A-AB.ADA	P	CC3504K-B.ADA	P
CC1301A-B.ADA	P	CC3406B-AB.ADA	P	CC3601C-AB.ADA	P
CC1302A-AB.ADA	P	CC3406C-AB.ADA	P	CC3602A-AB.ADA	P
CC1304A-AB.ADA	P	CC3406D-B.ADA	P		

Chapter 14

AE2101A-B.ADA	P	CE2111A-B.ADA	P	CE3115A-B.ADA	P
AE2101B-B.ADA	P	CE2111B-B.ADA	P	CE3201A-B.ADA	P
AE2101C-B.DEP	P	CE2111C-B.ADA	P	CE3202A-B.ADA	P
AE2101D-B.ADA	P	CE2111D-B.ADA	P	CE3203A-B.ADA	P
AE3101A-B.ADA	P	CE2201A-B.ADA	P	CE3206A-B.ADA	P
AE3702A-B.ADA	P	CE2201B-B.ADA	P	CE3208A-B.ADA	P
AE3709A-B.ADA	P	CE2201C-B.ADA	P	CE3301A-B.ADA	P
BE2101E-B.ADA	P	CE2201D-B.DEP	P	CE3301B-B.ADA	P
BE2112A-B.ADA	P	CE2201E-B.DEP	P	CE3301C-B.ADA	P
BE2112B-B.ADA	P	CE2201F-B.ADA	P	CE3302A-B.ADA	P
BE2112C-B.ADA	P	CE2202A-B.ADA	P	CE3303A-B.ADA	P
BE2114A-B.ADA	P	CE2204A-B.ADA	P	CE3305A-B.ADA	P
BE2208A-B.ADA	P	CE2204B-B.ADA	P	CE3402A-B.ADA	P
BE3001A-B.ADA	P	CE2210A-B.ADA	P	CE3402B-B.ADA	P
BE3002A-B.ADA	P	CE2401A-B.ADA	P	CE3402C-B.ADA	P
BE3002E-B.ADA	P	CE2401B-B.ADA	P	CE3402D-B.ADA	P
BE3105A-B.ADA	P	CE2401C-B.ADA	P	CE3402E-B.ADA	P
BE3205A-B.ADA	P	CE2401D-B.DEP	P	CE3403A-B.ADA	P
BE3501A-B.ADA	P	CE2401E-B.ADA	P	CE3403B-B.ADA	P
BE3606C-B.ADA	P	CE2401F-B.ADA	P	CE3403C-B.ADA	P
BE3703A-B.ADA	P	CE2402A-B.ADA	P	CE3403D-B.ADA	P
BE3802A-B.ADA	P	CE2404A-B.ADA	P	CE3403E-B.ADA	P
BE3803A-B.ADA	P	CE2405B-B.ADA	P	CE3403F-B.ADA	P
BE3902A-B.ADA	P	CE2406A-B.ADA	P	CE3404A-B.ADA	P
BE3903A-B.ADA	P	CE2407A-B.ADA	P	CE3404B-B.ADA	P
CE2102A-B.ADA	P	CE2408A-B.ADA	P	CE3404C-B.ADA	P
CE2102B-B.ADA	P	CE2409A-B.ADA	P	CE3405A-B.ADA	P
CE2102C-B.TST	P	CE2410A-B.ADA	P	CE3405B-B.ADA	P
CE2102D-B.ADA	P	CE3002B-B.TST	P	CE3405C-B.ADA	P
CE2102E-B.ADA	P	CE3002C-B.TST	P	CE3405D-B.ADA	P
CE2102F-B.ADA	P	CE3002D-B.ADA	P	CE3406A-B.ADA	P
CE2102G-B.ADA	P	CE3002F-B.ADA	P	CE3406B-B.ADA	P
CE2103A-B.TST	P	CE3102A-B.ADA	P	CE3406C-B.ADA	P
CE2103B-B.TST	P	CE3102B-B.TST	P	CE3406D-B.ADA	P
CE2104A-B.ADA	P	CE3103A-B.ADA	P	CE3407A-B.ADA	P
CE2104B-B.ADA	P	CE3104A-B.ADA	P	CE3407B-B.ADA	P
CE2105A-B.ADA	P	CE3107A-B.TST	P	CE3407C-B.ADA	P
CE2106A-B.ADA	P	CE3108A-B.ADA	P	CE3408A-B.ADA	P
CE2107A-B.ADA	P	CE3108B-B.ADA	P	CE3408B-B.ADA	P
CE2107B-B.ADA	P	CE3109A-B.ADA	P	CE3408C-B.ADA	P
CE2107C-B.ADA	P	CE3110A-B.ADA	P	CE3409A-B.ADA	P
CE2107D-B.ADA	P	CE3111A-B.ADA	P	CE3409B-B.ADA	P
CE2107E-B.ADA	W	CE3111B-B.ADA	P	CE3409C-B.ADA	P
CE2108A-B.ADA	P	CE3111C-B.ADA	P	CE3409D-B.ADA	P
CE2108B-B.ADA	P	CE3111D-B.ADA	P	CE3409E-B.ADA	P
CE2108C-B.ADA	P	CE3111E-B.ADA	P	CE3409F-B.ADA	P
CE2108D-B.ADA	P	CE3112A-B.ADA	P	CE3410A-B.ADA	P
CE2109A-B.ADA	P	CE3112B-B.ADA	P	CE3410B-B.ADA	P
CE2110A-B.ADA	P	CE3114A-B.ADA	P	CE3410C-B.ADA	P
CE2110B-B.ADA	P	CE3114B-B.ADA	P	CE3410D-B.ADA	P

COMPLETE LIST OF TESTS AND RESULTS

CE3410E-B.ADA	P	CE3704B-B.ADA	P	CE3804M-B.ADA	P
CE3410F-B.ADA	P	CE3704C-B.ADA	P	CE3805A-B.ADA	P
CE3411A-B.ADA	P	CE3704D-B.ADA	P	CE3805B-B.ADA	P
CE3411C-B.ADA	P	CE3704E-B.ADA	P	CE3806A-B.ADA	P
CE3412A-B.ADA	P	CE3704F-B.ADA	P	CE3806C-B.ADA	P
CE3412C-B.ADA	P	CE3704M-B.ADA	W	CE3806D-B.ADA	P
CE3413A-B.ADA	P	CE3704N-B.ADA	P	CE3806E-B.ADA	P
CE3413C-B.ADA	P	CE3704O-B.ADA	P	CE3809A-B.ADA	P
CE3601A-B.ADA	P	CE3706C-B.ADA	P	CE3809B-B.ADA	P
CE3602A-B.ADA	P	CE3706D-B.ADA	P	CE3810A-B.ADA	P
CE3602B-B.ADA	P	CE3706F-B.ADA	P	CE3901A-B.ADA	P
CE3602C-B.ADA	P	CE3706G-B.ADA	P	CE3905A-B.ADA	P
CE3602D-B.ADA	P	CE3707A-B.ADA	P	CE3905B-B.ADA	P
CE3603A-B.ADA	W	CE3708A-B.ADA	P	CE3905C-B.ADA	P
CE3604A-B.ADA	W	CE3801A-B.ADA	P	CE3905L-B.ADA	P
CE3605A-B.ADA	P	CE3804A-B.ADA	P	CE3906A-B.ADA	P
CE3605B-B.ADA	P	CE3804B-B.ADA	P	CE3906B-B.ADA	P
CE3605C-B.ADA	P	CE3804C-B.ADA	P	CE3906C-B.ADA	P
CE3605D-B.ADA	P	CE3804D-B.ADA	P	CE3906D-B.ADA	P
CE3605E-B.ADA	P	CE3804E-B.ADA	P	CE3906E-B.ADA	P
CE3606A-B.ADA	P	CE3804F-B.ADA	P	CE3906F-B.ADA	P
CE3606B-B.ADA	P	CE3804G-B.ADA	P	CE3907A-B.ADA	P
CE3701A-B.ADA	P	CE3804I-B.ADA	P	CE3908A-B.ADA	P
CE3704A-B.ADA	P	CE3804K-B.ADA	P	EE3102C-B.ADA	P

END

FILMED

6-86

DTIC