Software Product Description

PRODUCT NAME: VAX Ada Version 2.2 **SPD 26.60.10**

DESCRIPTION

VAX Ada is Digital Equipment Corporation's validated implementation of the full ANSI/MIL-STD-1815A-1983 Ada Language. The VAX Ada compiler runs under the VMS Operating System and generates optimized, shareable, and position-independent code.

As a native mode VMS language, VAX Ada is integrated into the VMS common language environment. All VMS system services and utilities are thus available to programs written in VAX Ada. VAX Ada supports the VAX Record Management Services (RMS) sequential, relative, and indexed file organizations and associated access methods. VAX Ada programs can invoke modules written in other VMS languages. Additionally, programs written in other languages can invoke VAX Ada modules.

Ada is a powerful, general-purpose language that supports many modern programming practices. The language was designed as the result of a competition sponsored by the United States Department of Defense. The purpose of the competition was to define a language suitable for programming embedded computer systems. Among the requirements for the language were features that would reduce software costs by increasing maintainability, evolvability, reliability, and portability.

Ada meets these requirements by providing a modular structure for programs and allowing separate compilation of program units, as well as providing strong typing, exception handling, standard language features that must be supported across implementations. Ada provides a number of other features that make it suitable for a variety of applications including general systems and computational applications. Because it includes language features that support multitasking, Ada is particularly well suited for real-time applications.

Ada Language Features

- Strong typing - An object (variable) of a given type may take on only those values that are appropriate to that type, and only certain predefined operations may be performed to data of that type. Because type checking is done at compile time, strong typing ensures that any errors associated with incorrect data types are detected at compile time.
- Data Abstraction - Also known as information hiding, data abstraction hides implementation details while providing users with mechanisms for using the implementation. Abstraction allows the user to focus on important characteristics while ignoring underlying details. Ada provides various levels of abstraction through features such as private data types and packages.
- Concurrent Processing - For many applications, it is important that a program be conceived of as a number of parallel, rather than serial, activities. Most high-order languages provide little or no support for handling such parallel or concurrent activities; they rely on facilities of the host operating system. Ada uses tasks to allow parallel activities to be programmed directly within the language.
- Separate compilation - Ada's separate compilation feature allows a programmer to divide a large program into compilation units that may be compiled at different times. When a unit is compiled, the Ada program library manager records information about that unit and other related units. This feature is unlike separate compilation features in other languages, where very little information about separately compiled modules is maintained.
- Generic Definitions - A generic unit is a template from which specific instances can be made at compile time. In many cases, the logic of an algorithm or a set of operations is independent of the specific type of the values being manipulated. However, in a strongly typed language such as Ada, all types must be defined at compile time. Generic definitions allow the user to define a general algorithm or set of related operations, and then create a specific instance of that algorithm or set of operations for each type to which the algorithm or operations must apply.
- Exception Handling - In many operations, especially embedded computer systems, it is critical that a system be able to recover quickly and efficiently from error conditions. Ada provides the ability to raise

and handle exceptions. It includes predefined exceptions and also permits the user to define exceptions. When an exception occurs in an Ada program, normal processing is abandoned and control passes to the exception handler.

VAX Ada Components and Special System-Related Features:

- An Ada compiler fully conforming to ANSI/MIL-STD-1815A-1983 ("Reference Manual for the Ada Programming Language") and validated by the Ada Joint Program Office.
- An Ada program library manager that provides support for programming teams through:
  - — Program libraries that can be shared by many programmers
  - — Program sublibraries that can be used to isolate, develop, and test Ada units in the context of parent program libraries
  - — Automatic recompilation of obsolete compilation units
  - — The ability to share compiled Ada code either by reference or copy
- High-level, fully symbolic debugging capability through the VMS debugger, including support for:
  - — Mixed Ada and non-Ada code
  - — Packages
  - — Tasking programs
- Integration with VMS Operating System including:
  - — Conformance to the VAX Calling Standard, which allows Ada code to call and be called by code written in other languages, as well as to call VMS system services and the VMS run-time library
  - — The ability to call VAX Record Management Services (RMS) routines directly
  - — Full access to relative and indexed file capabilities
  - — The ability to handle exceptions from non-Ada code and generate exceptions to be handled by non-Ada code
  - — The ability to handle VMS asynchronous system traps (ASTs)
  - — The ability to link with shared images and use shared global sections
  - — The ability to share data with non-Ada code through global variables and psects (common blocks)
- System-dependent facilities - Different systems vary in such characteristics as the size of storage units, memory size, and the smallest and largest integer values supported. Ada provides the predefined package SYSTEM to define system-related constants and to represent system-dependent information.
- Ada also provides representation clauses, which allow the user to tailor the representation of data to suit a particular system.
- VAX Ada provides:
  - — Length clauses, which specify the amount of storage associated with a type.
  - — Enumeration representation clauses, which specify the internal codes for the literals of enumeration types.
  - — Record representation clauses, which specify the layout of a record type, such as the order, position, and size of record components.
  - — Address clauses, which specify required addresses in storage for particular entities (in VAX Ada, only for constants and variables).
- VAX Ada provides a number of pragmas (compiler directives) that allow various system-related parameters to be set and changed, control mixed-language programming.
- Comprehensive diagnostic messages, including automatic syntax error correction, directed at helping the new Ada user.

HARDWARE REQUIREMENTS

VAX, MicroVAX, VAXstation or VAXserver configuration as specified in the System Support Addendum (SSA 26.60.10-x).

SOFTWARE REQUIREMENTS*

For Systems Using Terminals: (No DECwindows interface)

- VMS Operating System

For Workstations Running VWS:

- VMS Operating System
- VMS Workstation Software

For Workstations Running DECwindows:

- VMS Operating System (and necessary components of VMS DECwindows)

VAX Ada supports a DECwindows compiler interface for compile qualifier selection on workstations. This interface is accessible from FileView.

* Refer to the System Support Addendum (SSA 26.60.10-x) for availability and required versions of prerequisite/optional software and for information regarding components of VMS DECwindows.

ORDERING INFORMATION

Software Licenses: QL-056A*-**
Software Media: QA-056A*-**
Software Documentation: QA-056AA-GZ
Software Product Services: QT-056A*-**

* Denotes variant fields. For additional information on available licenses, services and media, refer to the appropriate price book.

SOFTWARE LICENSING

This software is furnished under the licensing provisions of Digital Equipment Corporation's Standard Terms and Conditions.

For more information about Digital's licensing terms and policies, contact your local Digital office.

LICENSE MANAGEMENT FACILITY SUPPORT

This layered product supports the VMS License Management Facility. License units for this product are allocated on a CPU-capacity basis.

For more information on the License Management Facility, refer to the VMS Operating System Product Description (25.01.xx) or the VMS Operating System documentation set.

For more information on Digital's licensing policies, contact your local Digital office.

SOFTWARE PRODUCT SERVICES

A variety of service options are available. For more information contact your local Digital office.

SOFTWARE WARRANTY

Warranty for this software product is provided by Digital with the purchase of a license for the product as defined in the Software Warranty Addendum of this SPD.

™ The DIGITAL Logo, VMS, DECwindows, VAX, MicroVAX, VAXstation, VAXserver and VAXset are trademarks of Digital Equipment Corporation.