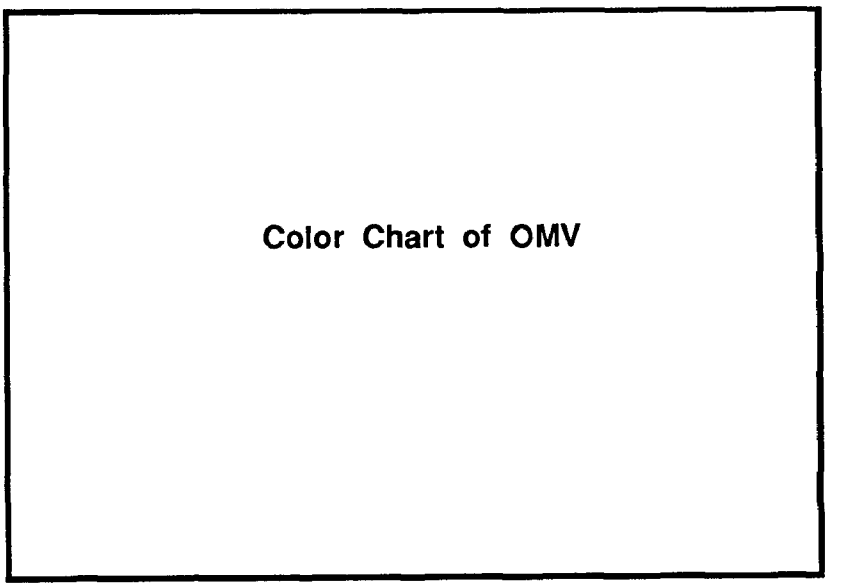
Ada in Real - Time Embedded Systems

Orbital Maneuvering Vehicle (OMV)

William T. Howle
NASA / MSFC

William T. Howle is a Computer Engineer in the Systems Software Branch of the Information and Electronics Systems Laboratory at NASA's Marshall Space Flight Center. He joined NASA in July 1987 after receiving a Bachelor's degree in Computer Engineering from Auburn University, and is currently participating in the design and development of flight software for the Orbital Maneuvering Vehicle.



The OMV is a NASA project assigned to the Marshall Space Flight Center, Huntsville, Alabama. The contract was awarded in November of 1986 to TRW Space and Technology Group, Redondo Beach, California.

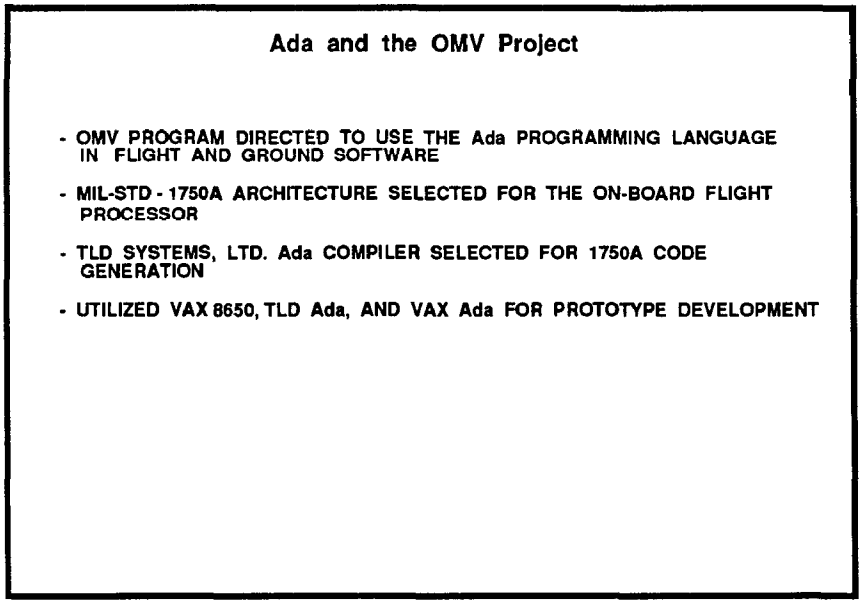
The Design, Development, Test and Evaluation (DDT&E) mission is scheduled for 1993.

OMV Requirements Summary

- PROVIDE SPACECRAFT DELIVERY, RETRIEVAL, REBOOST, DEBOOST, AND VIEWING
- SHUTTLE BASED, (ETR & WTR), GROUND CONTROLLED
- SPACE STATION BASED - GROUND AND STATION CONTROLLED
- AUTOMATIC NAVIGATION AND RENDEZVOUS
- MAN IN THE LOOP CONTROL FOR FINAL OPERATIONS
- PROVIDE LIMITED RESOURCES TO PAYLOADS
- ACCOMODATE VARIOUS MISSION KITS
- NINE MONTHS ON-ORBIT WITH SELF CONTAINED STORAGE
- CAPABLE OF ON-ORBIT MAINTENANCE
- TEN YEAR LIFE WITH REFURBISHMENT

The OMV will provide several services to various types of spacecraft including new and existing satellites. The OMV can be controlled through the shuttle's payload interface, from the Ground Control Console at JSC, or from the Space Station. It has automatic navigation and rendezvous capabilitiy and does require man in the loop control for final operations such as docking with the Space Station or payloads.

The OMV can accomodate various mission kits including a three-point docking system and a grapple arm. It has a nine month on - orbit capability with self -contained storage and is capable of on - orbit maintenance. The expected life of an OMV is ten years with refurbishment.



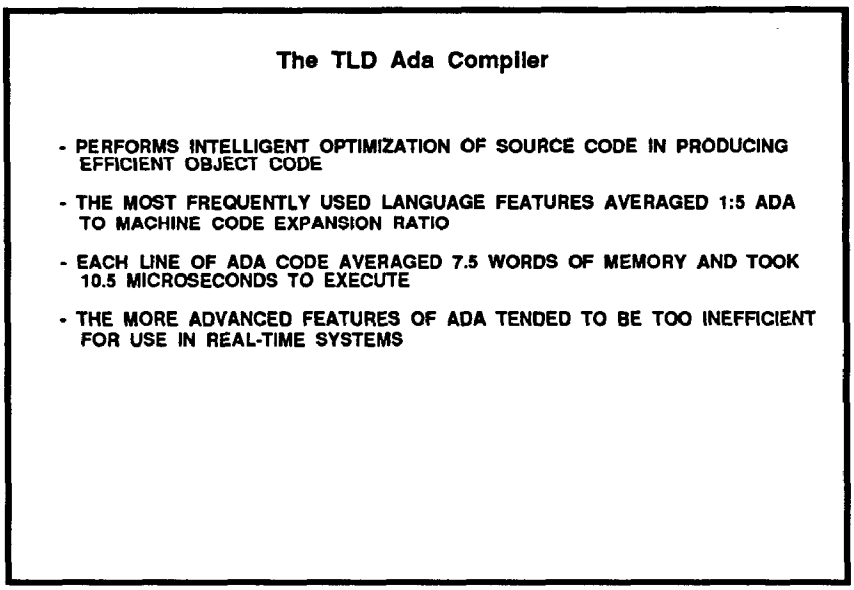
Following the DoD's commitment to Ada, NASA has begun to adopt Ada in many new space flight programs including Space Station and OMV. The OMV program originally intended to use FORTRAN, but Ada was eventually adopted through an Engineering Change Proposal (ECP).

The CDC444-RR microprocessor was chosen as the 1750A processor for the flight computer. It is a 16 - bit processor with a 250 microsecond cycle time. There will be 256 K words of memory in the OMV on-board computer.

The TLD cross compiler is hosted on a VAX 8650 and generates 1750A assembly code specific to the CDC444 processor. This code is then loaded into the on-board computer's memory before launch.

A major flight software prototyping effort took place on the VAX 8650 using VAX Ada and TLD Ada. This prototyping included models of various parts of the OMV operational flight program such as the executive model (used to evaluate Ada tasking) and a data load model (used to evaluate dynamic memory in Ada).

The prototyping provided much insight into the pros and cons of using Ada in OMV flight software.

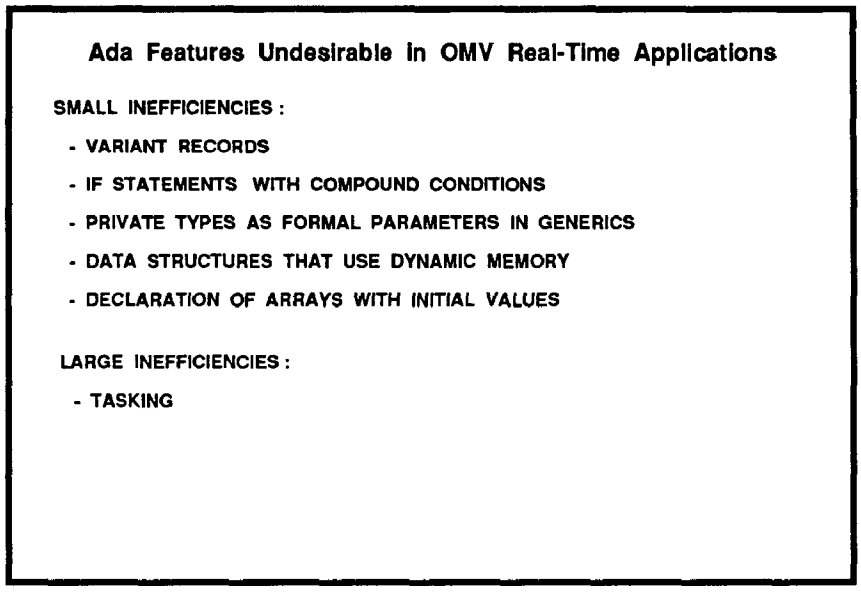


Overall, the TLD compiler does an excellent job in optimizing the Ada source code in order to produce the most efficient object code possible.

An Ada to machine code expansion ratio of 1:5 along with 7.5 words of memory and 10.5 microseconds execution time for an Ada construct proved satisfactory for OMV flight software.

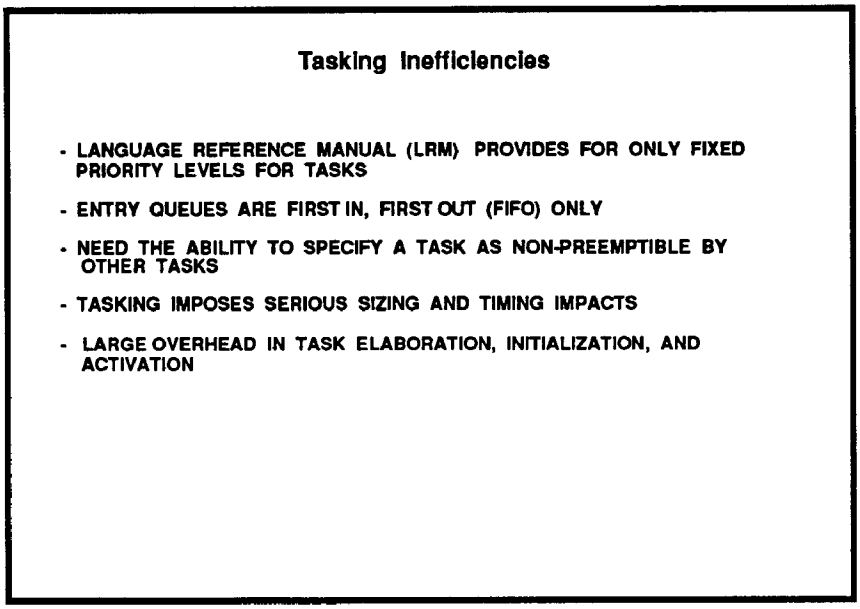
These statistics are from a group of benchmark programs designed to examine the efficiency of the TLD compiler and linker. The statistics are averaged across the Ada language and exclude language features that are inefficient or not planned for use.

Even though the results of the benchmark tests were impressive, the more advanced features of Ada can be very inefficient especially in real - time aplications such as the OMV flight program.



During benchmark testing with the TLD compiler and prototype development with VAX Ada, several features in the MIL-STD 1815A Language Reference Manual were found to be too inefficient for use in OMV flight software. These inefficiencies are specific to the TLD Ada compiler. Most of the more inefficient features of Ada are not inherently called for in real-time embedded applications such as the OMV flight program.

Variant records generate alot of control code. Private types used as formal parameters in generics generate a load / store sequence for each private object before each compiled language construct.. Dynamic memory allocation / deallocation simply requires too much overhead for a real - time system. When an array is declared with initial values, a block of literals is created and a block move is used to copy the literals into the array.

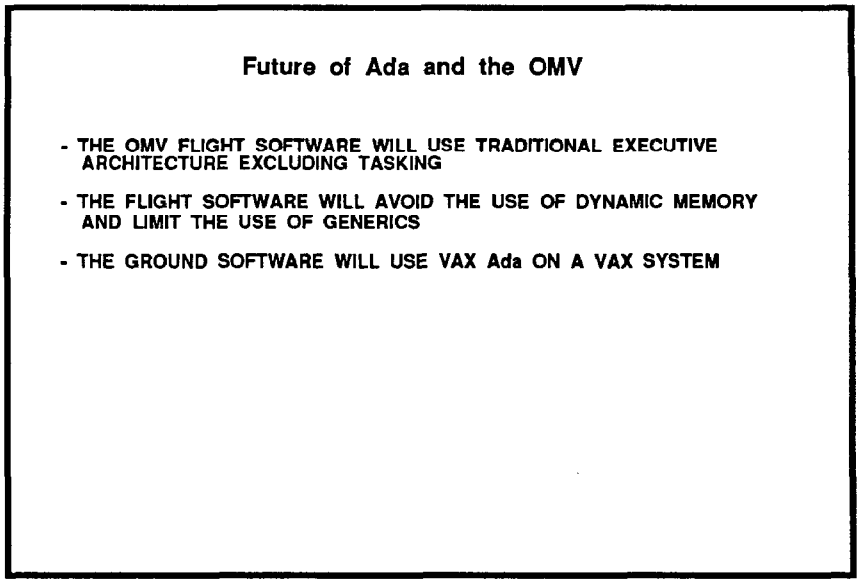


The Ada Language Reference Manual provides only fixed priority levels for tasks. There may be a need to be able to change the priority of an Ada task at run time. Most implementations allow this only at compile time, or sometimes at link time.

Entry to Ada tasks are FIFO. The Ada rendezvous requires synchronous communication. Asynchronous entry to tasks and prioritized entry queues would provide more efficient task scheduling constructs and possibly allow some Ada systems to meet real - time constraints.

May need the capability to prevent a task from being preempted by other tasks in order to meet real - time constraints.

The prototype tasking algorithms were several orders of magnitude slower than the traditional sequential real - time executive. Also, the load size was found to be approximately three times that of the traditional sequential executive.



The OMV flight software will avoid the use of tasking due to its increased overhead. The more traditional sequential executive architecture will be employed instead.

Also, the OMV flight software will avoid the use of dynamic memory and limit the use of generics.

The ground software will use VAX Ada and run on a VAX system. Many of the same issues concerning the advanced features of Ada are being reviewed .