Avionics Laboratory Ada Compiler Evaluation

Marc J. Pitarys, 1st Lt, USAF

Avionics Laboratory
WRDC/AAAF-3

ABSTRACT

All Air Force weapon system software will be written in Ada. Therefore, it is essential that a quality Ada compiler be used, since it will influence the design and cost of the avionics. An evaluation must take place before selecting the project compiler. The Air Force's Avionics Laboratory has several Ada cross compilers targeted to embedded processors. This paper will present results from an evaluation of Ada cross compilers for the MIL-STD-1750A computer. Compiler performance using the ACEC, the Common Ada Missile Package (CAMP) benchmarks, and the Performance Issues Working Group (PIWG) tests will be summarized. In addition, the utility of the available test-suites for performing Ada compiler evaluations will be addressed. Issues such as ease of testing, portability, and test coverage will be examined.

INTRODUCTION

It is common knowledge that all new avionics software must be written in Ada. Unfortunately, the performance of Ada on avionics architectures is not well known. Considerable effort has been focused on establishing software engineering practices, creating Ada development tools, and promoting the benefits and cost savings of using Ada. However, little has been done in gathering quantitative data on Ada's performance. Without this data, the cost, schedule, and design of avionics systems can not accurately be determined. In an attempt to fill this void, the Air Force's Avionics Laboratory is evaluating Ada cross compilers for embedded processors. This effort is intended to identify those features of Ada that can reliably support avionics software development and maintenance, and provide data for selecting compilers for avionics applications.

BACKGROUND

To conduct a proper evaluation, an organization must have the cross compilers, test suites, embedded processors, and necessary personnel. The Avionics Laboratory's System Avionics Division (WRDC/AAA) has several Ada cross compilers available, numerous MIL-STD-1750A processors (in addition to several 32-bit commercial processors), three test-suites, and within it's Software Concepts Group (WRDC/AAAF-3) the expertise to work with Ada software.

This paper contains some of the results from testing with three test-suites. These test suites are the Performance Issues Working Group (PIWG) benchmarks, the Common Ada Missile Packages (CAMP) benchmarks, and the Ada Compiler Evaluation Capability (ACEC). Not all of the benchmarks could be compiled, linked or executed with all of the compilers. There are several reasons for this. First, all of the compilers failed on certain tests. Second, some of the benchmarks used features not supported by the compiler. And third, a great deal of time is required in collecting and analyzing the diverse information available from such a large number of tests.

The Ada/1750A cross compilers resided on several different VAX hosts. Since the object of this paper is to determine how compilers overall are performing with respect to certain features of Ada, the names of the compiler vendors will remain anonymous (DoD organization may request the vendor names by contacting the author). Two compilers, "Vendor B" and "Vendor C", were hosted on a VAX 11/785. The "Vendor A" compiler was hosted on a VAXstation III, while the "Vendor D" compiler was on a MicroVAX II. The code for each test was executed on a Sperry 1750A computer. This computer runs at approximately .4 MIPS with the DAIS mix.

PERFORMANCE ISSUES WORKING GROUP

The Performance Issues Working Group (PIWG) Benchmarks are a set of tests developed by the Association of Computing Machinery (ACM) Special Interest Group for Ada (SIGAda) PIWG sub-group. The PIWG benchmark results included in this paper tested compiler performance on several of the Ada language features. These tests are in the public domain. A description of each PIWG tests used can be found in the benchmark's source code. Table A and Table B summarize some of the PIWG time measurements collected in the evaluation effort. The tests were compiled with all the checks enabled and then with as many checks as possible suppressed.

There are several good features of the PIWG benchmarks. The tests are relatively easy to compile, link, and execute. The PIWG tests often give a rough snapshot, in minimum time, (less than 1 day), on how a cross compiler is performing with respect to another. This test suite offers three ways for extracting the timing information. This is determined by what output package body is compiled. One package body allows the output to be sent to a console screen or printer. Another package body allows the results to be sent to a file. A third package allows the time measurements to be read from memory. These results are stored as 32-bit (two words) fixed point numbers for the variables CPU_TIME and WALL_TIME. This package is especially useful when an embedded processor does not contain a console terminal.

The PIWG benchmarks are not very comprehensive when compared to the ACEC. Performance measurements for tasks, loops, exception handling, dynamic allocation, and procedures are covered. Specific tests for compiler optimizations are lacking. There are a few classical and application tests included.

In general, it is recommended that before embarking on a major compiler evaluation with multiple test suites, the PIWG performance benchmarks should be executed. The level of effort in getting the compilers to work with these benchmarks gives an indication of what can be expected throughout the entire effort. In addition, the user can obtain comparison data in a timely manner so as to filter out the poorest performing compilers from having to be tested with all of the other benchmarks used in the evaluation.

COMMON ADA MISSILE PACKAGES

The Common Ada Missile Packages (CAMP) Benchmarks are a set of tests designed to support the evaluation of compilers for several specific areas. One, they test compilers for embedded (armament) software applications. Two, they test compiler performance with respect to reusable software. And third, they test run-time performance issues such as execution time and computational accuracy [1]. The CAMP Benchmarks are available to DoD organizations and registered DoD contractors from the Data and Analysis Center for Software (DACS).

Results from using the CAMP compilation benchmarks are listed in Table C. These 31 benchmarks emphasize the Ada compiler's ability to process reusable software. "The benchmarks concentrate on the complex syntax and semantics of several Ada armament software oriented implementations using the CAMP parts. These implementations are skeletal in that do not actually implement a software subsystem but merely collect the necessary CAMP parts via generic instantiation." [2] After compilation, three benchmark drivers are linked to create an executable image.

The CAMP benchmarks are excellent for testing compilers with respect to reusable software and the implementation of generics. In addition to the compilation benchmarks, this test suite contains two other classes of benchmarks. These are the polynomial benchmarks and application benchmarks. Since these two other classes make use of many Ada programs designed to be reusable, it is recommended that these two benchmark classes only be used if there is an emphasis on evaluating compiler implementation of Ada generics, otherwise one can avoid using the CAMP benchmarks in an evaluation. As Table C indicates, cross compilers for embedded processors such as the 1750A computer have trouble with generics.

ADA COMPILER EVALUATION CAPABILITY

The Ada Compiler Evaluation Capability (ACEC) is a set of benchmarks, analysis tools, and guides for measuring the performance of Ada compilers. It is available to DoD organizations and registered DoD contractors from the Data and Analysis Center for Software (DACS).

The ACEC contains over a thousand tests that cover a wide domain. Compiler performance (speed and expansion size) is measured for compiler optimizations, Ada language features, and application specific routines.

The ACEC was designed to be portable. This is to allow many compilers on different hosts for various target processors to be tested. However, one should not assume that the ACEC tests can be compiled and executed immediately with out some effort. Fortunately, the design of the ACEC facilitates some of the modifications. Some of the areas that need modification, so a cross compiler can be tested are:

1. Command files: The command file (cmp.com) used to compile and link the tests need the compiler specific directives. When the ACEC is delivered, it contains compile and link commands for the Digital Equipment Corporation (DEC) Ada compiler. These need to be replaced with the equivalent commands the compiler under test requires.

2. Include files (Inittime.txt, Starttime.txt, and Stoptime0.txt): The code inserted by the include pragma may need to be modified. There are several reasons for this. First, the compiler may fail when trying to compile the statements that compute expansion size. Second, the TEXT_IO package may not be implemented or may only work for character, string, integer, and float types. A compiler specific I/O package may have to be substituted. In addition, the output data may have to be converted to an integer or float type.

3. Package Global: As indicated in the discussion on modifying the included files, this package may have to be modified if TEXT_IO is not fully implemented. Note that Package Global contains the code for procedure stoptime2, which outputs data.

4. Include Program: This program inserts code into the benchmark source for computing expansion size. Therefore, these statements may need to be eliminated if compiling fails on the computation of expansion size.

5. Math Package: The math package is a generic package that provides several mathematical functions. "The proper functioning of the math package in the test suite depends on access to real number representation at the bit level. Some modification is necessary for many systems. Furthermore, since compilers differ in their support of the needed Chapter 13 features such as record representation clauses and Unchecked_Conversion, modifications may be required for different compilers on the same target hardware." [3] Users of the ACEC should read the report indicated in Reference 3.

6. Benchmark Source Code: If a compiler specific I/O package is used instead of TEXT_IO, every source file will need to be changed. TEXT_IO will need to be replaced. Furthermore, the statements that instantiate FLOAT_IO and INTEGER_IO will need to be eliminated or modified.

The ACEC includes an analysis program called "median". This program takes raw timing data as input and outputs histograms and tables that allow the user to determine compiler optimizations and relative performance. Running median with data from several compilers generates considerable output. Two parts of this output that are of particular interest are the system factors and the summary of the percentages of tests that worked. Some data obtained from using the ACEC is included in Table D.

Any extensive compiler evaluation should include the ACEC. It is the most comprehensive and documented test suite available. However, because of the large number of benchmarks, using the ACEC for testing cross compilers can take from one to four person-weeks.

ADDITIONAL TESTING AREAS

Many Ada toolsets include program library services. The program library contains information on all of the program compilation units. In any large development effort, Ada library services are important in producing code efficiently and correctly. At this time, none of the test suites discussed provide explicit tests for measuring how well Ada libraries are supported by the toolsets. However, by compiling and linking the benchmarks, one can gain some insight into how well the program library performs. Table E shows the sizes of the program units in an Ada program library from compiling and linking several of the PIWG tests. A simple lesson from any benchmarking activity would be that one should never overlook deleting unneeded program units from the program library.

It would also be helpful if all of the benchmarks in each of the tests suites included documentation on the total lines of code for each test. This information would save time when computing code expansion ratios (CERs). CERs are vital when attempting to determine software costs and the needed system resources for a particular effort. In fact, automating the computation of CERs would be very helpful. A potential solution using the ACEC would be to include code that divide the number of lines in the test by the computed expansion size.

SUMMARY

This paper presented data from the Avionics Laboratory's Ada compiler evaluation. The current emphasis was on cross compilers for the MIL-STD-1750A processor. The Laboratory is currently testing cross compilers for 32-bit computers. This should help organizations planning to conduct an evaluation of Ada compilers. In addition, the inclusion of data for several compilers' performance with the benchmarks should offer some help in identifying how well Ada compilers, in general, are performing for the 1750A.

TABLE A
PIWG BENCHMARK TIMES
(MICROSEC)

PIWG TEST	A CHECKS	A	B CHECKS	B
C000001	1806	1757	1089	1072
C000002	1807	1758	1135	1106
C000003	1771	1741	1126	1106
D000001	52	23	60	34
D000002	11439	11399	12839	12795
D000003	94	60	R	R
D000004	13783	13632	R	R
E000001	20	21	50	50
E000002	54	54	64	63
E000003	56	55	38	38
E000004	69	68	33	32
E000005	211	211	290	287
F000001	3	3	11	8
F000002	4	4	8	8
H000001	121	36	20	20
H000002	983	901	2350	2347
H000003	2269	1856	R	2458
H000004	941	440	1677	746
H000005	1	0	0	0
H000006	23	33	13	23
H000007	77	82	C	C
H000008	C	C	R	R
L000001	3	4	7	7
L000002	3	4	8	9
L000003	3	4	8	8
L000004	2	3	6	6
L000005	2	3	6	6
P000001	24	13	22	10
P000002	31	21	30	19
P000003	27	13	22	8
P000004	27	13	21	8
P000005	30	16	27	14
P000006	32	18	26	13
P000007	33	18	31	18
P000010	52	41	60	47
P000011	75	61	98	83
P000012	67	54	68	56
P000013	106	93	115	102
T000001	740	738	460	446
T000002	739	737	456	450
T000003	764	739	496	469
T000004	923	912	526	511
T000005	743	732	534	523
T000006	1728	1714	892	883
T000007	740	738	462	448
T000008	1807	1772	1058	1002

C = compile error R = execution error

TABLE B
PIWG BENCHMARK TIMES

(Microsec)

PIWG TEST	C CHECKS	C	D CHECKS	D
C000001	4320	4268	R	R
C000002	4312	4260	R	R
C000003	4231	4242	R	5
D000001	348	320	34	30
D000002	13557	13303	30500	30600
D000003	386	330	4050	4025
D000004	R	R	38300	38300
E000001	54	32	40	41
E000002	69	56	72	72
E000003	20	18	76	76
E000004	6	14	94	91
E000005	203	183	356	363
F000001	5	6	7	8
F000002	5	6	7	7
H000001	9	11	4875	R
H000002	3151	2535	2300	2289
H000003	8181	1343	3050	3025
H000004	7631	722	1119	1113
H000005	0	0	70	71
H000006	136	34	234	231
H000007	C	C	84	84
H000008	C	C	R	R
L000001	6	6	8	8
L000002	7	7	9	8
L000003	6	6	7	7
L000004	6	6	7	7
L000005	6	6	7	7
P000001	39	9	30	29
P000002	97	67	34	32
P000003	56	8	30	30
P000004	56	8	29	30
P000005	60	9	32	32
P000006	64	16	38	38
P000007	73	14	39	40
P000010	100	37	57	55
P000011	144	77	81	78
P000012	107	44	78	79
P000013	152	152	127	128
T000001	969	970	725	713
T000002	969	973	712	700
T000003	1039	1007	731	719
T000004	1698	1678	R	1019
T000005	956	967	R	R
T000006	3458	3471	2320	2320
T000007	421	416	719	719
T000008	3405	3162	2113	2100

C = compile error R = execution error

TABLE C
CAMP COMPILATION BENCHMARKS

COMPILER	TESTS COMPILED	TESTS LINKED	HOST CPU TIME (SEC)
VENDOR A	28	0	304
VENDOR B	29	1	1573
VENDOR C	29	2	1047
VENDOR D	23	0	946

31 benchmarks

3 linkable benchmark drivers

TABLE D
ACEC LOOP TESTS

(Microsec)

TEST	TLD	SD
LOOP0	30100	44038
LOOP1	30100	62855
LOOP10	NO DATA	NO DATA
LOOP11	CONSTRAINT	67007
LOOP12	8010	17072
LOOP13	303003	474151
LOOP14	501	561
LOOP15	2030	4806
LOOP16	3000	3158
LOOP17	20500	41432
LOOP2	15001	27799
LOOP3	6005	10317
LOOP4A	100007	108795
LOOP4B	100040	112853
LOOP4C	50200	53042
LOOP5	4004	5540
LOOP6	402	543
LOOP7	60040	NO DATA
LOOP8	1601	NO DATA
LOOP9	2002	2969

TABLE E
LIBRARY PROGRAM UNIT SIZES

PIWG TEST	VENDOR D SIZE (BLOCKS)
C000001	798
C000002	798
C000003	792
D000001	786
D000002	801
D000003	792
D000004	798
E000001	786
E000002	792
E000003	798
E000004	798
E000005	813
F000001	780
F000002	780
H000001	822
H000002	828
H000003	822
H000004	825
H000005	789
H000006	789
H000007	789
L000001	897
L000002	897
L000003	897
L000004	786
L000005	786
P000001	777
P000002	780
P000003	786
P000004	786
P000005	786
P000006	786
P000007	786
P000010	786
P000011	789
P000012	789
P000013	789
T000001	898
T000002	804
T000003	822
T000004	813
T000005	972
T000006	882
T000007	792
T000008	858

REFERENCES

1. The CAMP Armonics Benchmark Suite
18 DEC 1987; page 1; Air Force Armament
Laboratory, Eglin AFB, FL 32542
2. Ibid. page 3
3. Ada Compiler Evaluation Capability
(ACEC) Technical Operating Report (TOR)
User's Guide; 10 AUG 1988; page 35; Air
Force Systems Command, Aeronautical
Systems Division, AFWAL/AAAF, WPAFB, OH.