OAK RIDGE NATIONAL LABORATORY
operated by
UNION CARBIDE CORPORATION
for the
U.S. ATOMIC ENERGY COMMISSION

UNION CARBIDE

ORNL - TM - 404

COPY NO. - 92

DATE - October 18, 1962

An ALGOL Compiler for the Control Data Corporation 1604

Progress Report No. 2

L. L. Bumgarner
A. A. Grau
M. P. Lietzke
R. G. Stueland*
K. A. Wolf*

Abstract

This report summarizes the progress to date on the ALGOL compiler for the Control Data Corporation 1604 Computer, based on machine-independent specifications developed at ORNL. Included are: (1) a brief description of the structure of the compiler, (2) language restrictions, (3) the hardware language representation, and (4) the required equipment configuration. An example illustrates the object program instructions generated.

*Control Data Corporation

An Algol Compiler for the Control Data Corporation 1604

Progress Report No. 2

October 18, 1962

L. L. Bumgarner
A. A. Grau
M. P. Lietzke
R. G. Stueland*
K. A. Wolf*

This report supersedes the previous progress report [7].

Compiler structure

There are actually two separate compilers, each with its own supervisory program under the monitor master control system. One is a fast load-and-go compiler and the other produces input to the assembly program. The latter also generates library subroutines. The two compilers are identical in most respects, however, and we shall use the term "compiler" to refer to both.

The compiler makes two passes through the source program, but there is no intermediate tape use except on very large programs. Processing is terminated at the end of the first pass if errors are discovered. Since the first progress report, the two passes have been somewhat more closely integrated.

The compiler is designed to produce an efficient target program. Unnecessary use of temporary storage is avoided, and considerable effort has been applied to the optimization of loops. This has not included recursive address calculation by means of index registers [1] or replacement of common sub-expressions. See example near end of this report.

---

*Control Data Corporation

Language restrictions

The following list of Algol 60 features which are not presently implemented or are handled only in a modified form is not to be considered final. This is the current list only, and some of these restrictions will be lifted.

1. Recursive procedures.
2. Integer labels.
3. Own arrays.
4. Call of arrays by value.
5. The following restrictions on types (arising from the fact that integer quantities are handled by fixed-point and real quantities by floating-point):
   a. Minor redefinition of the type of exponentiation expressions (to prevent the type from depending upon the value of the exponent).
   b. Types of actual parameters must match those of corresponding formal parameters, where the latter are assumed to have type real unless specified integer (of course this refers only to arithmetic quantities).
   c. The simple arithmetic expressions occurring as constituents of conditional arithmetic expressions must have the same type.
6. GO TO statements involving undefined switch designators are undefined.
7. An identifier cannot be used as both a label and the name of a variable in the same block.

Input-Output

At present facilities exist for making use of the Fortran input-output routines in the monitor. Format and list declarations are defined as adjuncts to the language.

In addition, a simple input routine has been written which requires no format information and accepts data in any of the forms permitted in Algol, making any type conversions necessary. It is felt desirable to have some output routines of the same type.

Work still remains on the design of binary read-write routines.

Hardware Algol Language

Algol word delimiters are recognized by the fact that they must be enclosed in "escape symbols". The escape symbol consists of a key punch character represented as an 8-4 punch on cards and printed on available equipment as an apostrophe or a dash. The present hardware representation was made necessary by the very limited character set available on existing key punch machines.

The philosophy behind the hardware language was based on the experience gained in the use of Algol on the Oracle at Oak Ridge [3]. This philosophy has been supported by a report from users of the KDF9 computer of the English Electric Company, Ltd. [4]. When the reference language symbol is available, it is used; otherwise, a substitution must be made. The rules of transliteration are so simple that they may be applied by the key punch operator: "If a symbol in the manuscript exists on the keyboard, use it; if not, refer to the card containing the rules and make the appropriate substitution"[4].

One exception to the rule seems to be desirable. The assignment symbol := is rendered by the equality sign =, while the equality relation operator becomes the word delimiter EQ (with escape symbols). This is justified by the very frequent occurrence of the assignment symbol and the desirability of having all the relational operators follow the same pattern. (Discussions with members of the ALCOR group have indicated it may be well to tolerate any of the symbols =, .= and ..= as the assignment symbol in the final version.)

The word delimiters includes all of those delimiters designated in the reference language by bold face type such as BEGIN and END. In addition, it is necessary to represent the relational and logical operators and the separator, $_{10}$, by word delimiters:

Reference Algol	Hardware Algol
$<$	'LS'
$\leq$	'LQ'
$=$	'EQ'
$\geq$	'GQ'
$>$	'GR'
$\neq$	'NQ'
$\neg$	'NOT'
$\wedge$	'AND'
$\vee$	'OR'
$\supset$	'IMP'
$\equiv$	'EQV'
$_{10}$	'E'

In some cases where the reference language delimiter cannot be represented by the same character in hardware language, a substitution of one character or a combination of two characters was made:

Reference Algol	Hardware Algol
×	*
↑	**
÷	//
:	..
;	$
:=	=
[	(/
]	/)
↑(string quote)	(*
↓(string quote)	*)

Equipment Configuration

This system is designed to operate under the CO-OP monitor. Requirements for input-output are:

1. master tape
2. standard input tape
3. standard output tape
4. punch tape
5. load-and-go tape
6. scratch 1
7. scratch 2

The standard input tape can be replaced by an on-line card reader, the standard output tape can be replaced by an on-line printer and the standard punch tape can be an on-line card punch. The punch tape, load-and-go tape and one scratch tape are not used by the load-and-go compiler.

Present Status

The coding is largely complete. Further coding will be limited to additional input-output facilities, removal of language restrictions, and minor streamlining modifications.

Debugging has been limited to a total of seven or eight weeks, so considerable testing effort remains. Progress thus far has been very satisfactory. Some 20 or 30 programs have been compiled and/or executed, a few under the monitor.

It is anticipated that the compiler will be released in early 1963.

Statistics

The compiler consists of about 5200 words of instructions, constants and temporary storage, exclusive of the supervisory program.

Some preliminary compilation speeds were recently obtained for the load-and-go compiler. They are of the order of 10,000 target instructions per minute. Compilation through the assembly language will of course be slower.

Sample Compilation

The following sample loop is from an actual test case run on the compiler. However, a variable will not appear on the actual output as shown here, but will be represented by a V followed by a serial number.

For example, N will appear on the listing as V2. The correspondence will be given by a mapping in the output listing. Temporary locations are designated by a T and generated labels by a GL.

```
FOR I := 1 STEP 1 UNTIL N DO
BEGIN
     R:= A[I]*T;
     F:= F*Q+R;
     G:= -G*Q+R;
     T:= T*P
END
```

All variables are of type integer.

The target code is:

```
            LDA      (1)
            STA       I
GL1         SUB       N
            INA      -1
            AJP P     GL2
            LDA       I
            ADD       A
            STA       T1
            LDA       T
            MUI 7     T1
            STA       R
            LDA       Q
            MUI       F
            ADD       R
```

```
        STA     F
        LDA     Q
        MUI     G
        SCM     ALLØNES
        ADD     R
        STA     G
        LDA     P
        MUI     T
        STA     T
        RAØ     I
        SLJ     GL1
GL2
```

For Use On The 3600 Computer

The compiler should operate satisfactorily on the Control Data 3600 computer. In the light of machine-independent specifications developed at ORNL, some thought has been given, however, to the question of modifications which would improve the performance on the more powerful machine. A significant improvement can be made in the efficiency of subroutine linkage in target programs. The implementation of double precision also appears attractive. Of course the coding of the compiler itself can be improved to a limited degree through use of some of the more powerful instructions.

These improvements could be made by personnel familiar with the compiler in about three man-months.

Documentation

A programming manual will be issued when the compiler is released. The specifications for the compiler will eventually be

documented as an ORNL report. Also, machine-independent specifications developed prior to and along with these will be made an ORNL report.

References and Bibliography

[1] F. L. Bauer and K. Samelson, "Sequential Formula Translation," Communic. Assoc. Comp. Mach. 3 (1960), no. 2, pp. 76-82.

[2] A. A. Grau, The Structure of an Algol Translator, ORNL-3054 (1961).

[3] ORACLE ALGOL Translator 1 (Preliminary Report). Multilithed Mathematics Panel Report, ORNL, Sept. 1960.

[4] F. G. Duncan, "ALGOL Translation for KDF9," Automatic Programming Information Bulletin, May 1961, no. 7, pp. 31-32.

[5] H. H. Bottenbruch, "Structure and Use of Algol 60," ORNL-3148 (1961), and Jour. Assoc. Comp. Mach. 9 (1962), no. 2, pp. 161-221.

[6] A Simple Introduction to Algol Programming for KDF9, The English Electric Company, Limited, Data Processing and Control Systems Division, Dec. 1961.

[7] L. L. Bumgarner, et al, An Algol Processor for the Control Data Corporation 1604 - Progress Report, ORNL-TM-190 (April 2, 1962).

Distribution

1. F. S. Acton, Princeton University
2. G. J. Atta
3-22. L. L. Bumgarner
23. D. B. J. Bridges, Jr., Lockheed Missiles and Space Company
24. W. R. Busing
25. H. P. Carter
26. Robert Church, Control Data Corporation
27. A. C. Downing
28. D. Drake, National Bureau of Standards
29. T. H. Elrod, Control Data Corporation
30. Hugh Everett, Institute for Defense Analyses
31. M. Feliciano
32. Rex Franciotti, International Business Machines
33. Wallace Givens, Argonne National Laboratory
34. A. A. Grau
35. R. T. Gregory, University of Texas
36. J. C. Gysbers, Rocketdyne
37. M. I. Halpern, Lockheed Missiles and Space Company
38. L. Hendricks, Sandia Corporation
39. A. Hirsch, General Electric Company
40. A. S. Householder
41. E. T. Irons, Institute for Defense Analyses
42. W. H. Jordan
43. R. S. Lehman, University of California
44. R. C. Lesser, Cornell University
45. M. P. Lietzke
46. C. L. Perry, University of California
47. I. Pyne, Princeton University
48. R. M. Price, Control Data Corporation
49. I. Rabinowitz, Princeton University
50. R. Ruble, Michigan State University
51. R. L. Shively, Swarthmore College
52. R. G. Stueland, Control Data Corporation
53. J. G. Sullivan
54. Bernie Tiefa, Control Data Corporation
55. M. Troost, General Dynamics Company
56. W. H. Whitaker, Control Data Corporation
57-66. K. A. Wolf, Control Data Corporation
67. D. M. Young, Jr., University of Texas
68. R. A. Zemlin, Control Data Corporation
69-70. Central Research Library
71. Document Reference Section
72-91. Laboratory Records
92. Laboratory Records - Record Copy
93-107. Division of Technical Information Extension
108. Research and Development Division