On a Compiler for a Process Computer

AKIRA TANAKA* AND KIYOSHI SUZUKI**

Regarding to the automatic programming systems for the process control computers, several works have been announced up to date. IBM, TRW (now, Bunker-Ramo) or Honeywell are their suppliers, for instance.

These systems are constructed to facilitate the treatment of the FORTRAN II language in their arithmetic computations.

Specific features for process control in handling the input/output signals have not yet been characterized in detail.

The authors planned to develop a system program for a process control computer HOC-300 in 1961.

A process computer application with a memory of 4∼32 K words is available for such purposes as,

(1) data gathering,
(2) data analysis, or
(3) optimal control.

Apart from such an application as a *data logger,* an operational program of a process computer is neccessarily under reprogramming after several thousand hours of operation.

The ratio of the instruction areas (including various constant tables) to the data areas generally lies in a range from 1.5 to 3.0. The ratio of the input/output routines to the inner computational routines lies between 0.78 and 1.46. The former value indicates the case of data gathering application to a blast furnace of Fuji Iron & Steel Co., and the latter of real-time control application to a basic oxygen steelmaking process of Nippon Kokan K.K.

Each program of those applications was achieved with some six thousand machine words. A highly experienced coder usually takes four or five hundreds hours in pure coding of those kind of applications. His labor is rather devoted to coding the input/output procedures because of complexity of the input/output signal specifications, the time relations among parts of process operations, the outer interruption procedures and their nestings, the man/machine interface characteristics and the programmed reliabilities. He is apt to make errors in the codings or allocations in the programming of these parts.

The development of an automatic programming system peculiar to process control

---

This paper first appeared in Japanese in Joho Shori (The Journal of the Information Processing Society of Japan), Vol. 5, No. 5 (1964), pp. 253-261.

* Nippon Electric Co. Ltd., Tokyo; formerly in Hokushin Electric Works Co., Tokyo.
** Toyo Kogyo Co. Ltd., Hiroshima; formerly in Hokushin Electric Works Co., Tokyo.

is, therefore, requested with special considerations in its input/output handlings. The purposes of this requirement are divided into;

(1) cut-down of programming cost,
(2) unneccessity of trained programmers in the process company,
(3) cut-down of turn-around time and out-of-service time.

The authors' project was planned and performed as shown below. The related computer was the process computer HOC–300 of Hokushin Electric Works, and the automatic programming system completed for HOC–300 was named HODRAL (Hokushin Data Reduction Algorithmic Language).

1961	Dec.	Start of the project.
1962	Jan.∼Feb.	Planning for development.
	Mar.∼Apr.	System design.
	May	Flow charting.
1963	Mar.	Completion of assembler I.
	Jun.	Completion of compiler.
	Jul.∼Sep.	Partial modification.
	Oct.∼	Completion of HODRAL compiler.
1964	Jan.	On-line debugging.

In the period from March to December of 1962, the authors and their cooperators were faced to four standardization tasks;

(1) the input/output hardware system of HOC–300,
(2) the machine code programming techniques,
(3) standardized design of library subroutines, flow charts, allocation maps, and input/output specification sheets, and
(4) their filing method.

HODRAL was oriented to an ALGOL-like language. The reason of this orientation was based on the less definiteness on its input/output representations of ALGOL. The function of inner computations of HODRAL, however, was subject to some restrictions in comparison with standard ALGOL. They were, instead, constructed with such abilities as;

(1) concurrent use with the assembler codes,
(2) program linkage, and
(3) input/output signal handling

Fig. 1 shows the processing flow of this program system, and Table 1 shows the statements of HODRAL.

Four statements, BEGIN, END, CANCEL and HALT are used for program linkage. The essential points in program linkage are illustrated below.

(1) A variable name is to be used independently in the other program. Or, a specified variable name is to be used relatively in multiple programs.
(2) Two or more source programs should be linked into a single object tape if neccessary. Or, they should be linked as specified.
(3) The storage of the object program is to be optionally assigned.

POINT and SHIFT are two statements useful to normalize the numbers manipulated in the arithmetic operations. The HOC–300 computer has no floating arithmetics, but has a rather long word of 36 bits.

Five statements, SAMPLE, CONVERT, LIMIT, MONITOR and EXTRACT are mainly used in the input signal manipulations. In manipulations of the process input variables, the followings are important.

Table 1 Statements of HODRAL.

arithmetic statement

V: =E;
V: =……V: =E;
V: =SHIFT (V, N, N);
V: =F (V);

control statement

IF (R)⇨L;
IF (R)⇨L, L;
FOR (SV: =C, C, C); S;……S;↓
GOTO (L);
HALT JUMP (L);

input/output statement

READIN (V);
TYPEOUT (V);
SAMPLE (V, N);
CONTROL (X, Y, N);

where, V or ∼ is written in the place of X, and one of ST, AT, ON, OFF, D, or B for Y.

MONITOR (V, C, L);
IDCTE (V);
LAMP (X, Y);

where, ON or OFF is written in the place of X, and one of N, SCAN or ADC for Y.

miscellanea

BEGIN;
BEGIN (A);
END;
HALT;
HALT (A);
CANCEL (V,……,V);
CONSTARRAY (V[C]……, V[C]);
DATASARRAY (V[C],……, V[C]);
LIMIT (V, V)·
FORMAT (C, C);
READER (C);
TYPER (C);
IDCTOR (C);
CONVERT (V, L);
EXTRACT (V, C, L);
MASK; UNMASK;

In the above, the meanings of variables are as follows.

V: variable name
(SV: simple variable name)
E: arithmetic expression
C: constant (N: integer)
R: relation
L: label
S: statement
A: absolute address

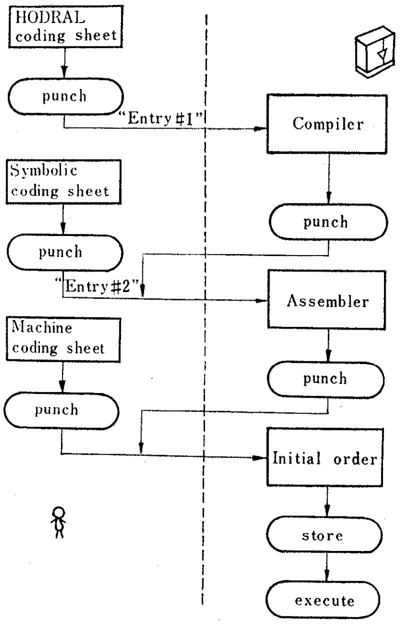

Fig. 1 Processing flow of the system.

(1) Variety of sampling methods: priority interruption, random selection, and sequential addressing.
(2) Time-shared operation of input devices: scanner, pre-amplifier, and analog/digital converter.
(3) Mode conversion of the input variables into the HOC-300 inner mode. This implies the linearization of a non-linear input and data compensation.
(4) Bit manipulation of the input variables to separate and examine an individual bit or a bit group which implies a contact status, a flag input, or a variable identification code.

SAMPLE statement is applied to the functions (1) and (2), CONVERT to (3), and EXTRACT to (4). Especially, in place of L (label) of the CONVERT statement (Table 1), names of the thermo-couples and thermo-resistance bulbs such as pt 50, CA or PR can be directly used.

LIMIT and MONITOR statements are generally used in pair to watch the process variables.

The concurrent use of the assembler codes with the HODRAL language aims at two effects.

(1) To get an efficient object program in case of running time critical. For instance, routines of the iterative readings of the flow variables, or routines of the iterative pulse outputs to actuate the control valves, should be partially written by the assembler codes to keep the operation cycles in some cases.
(2) In order not to deadlock the programming even when this trial—HODRAL—is recognized to be practically incomplete in its input/output signal treatments. There had not been reported such an example as this programming system

at the date of this project.

Four statements, IDCTOR, IDCTE, TYPER and TYPEOUT are used in man/machine interfaces such as operational guide indication or plant efficiency or alarm logging. As the control functions of a typewriter, CRLF, SPACE, RED, BLACK and TAB statements can be applied as they are.

CONTROL statement can be used in both cases of set point control and of direct digital control. Variations of the usage of this statement are

(1) CONTROL (X, ST, N); or
CONTROL (X, AT, N);

(2) CONTROL (∼, ON, N); or
CONTROL (∼, OFF, N);

(3) CONTROL (X, D, N); or
CONTROL (X, B, N);

where, X denotes a variable name of value o∼999, N an address of the distributor, D a conversion mode into binary coded decimal form, B no conversion mode. ST implies that the control value X is the value to be adjusted of the set point. AT implies, on the other hand, that X is the offset of the set point concerned. In case of velocity modulation, the lable B is used, and in case of time modulation an absolute address 63 are simultaneously used with B as

CONTROL (X, B, 63)

The address 63 denotes one of the output clock registers. This causes the clock register be set an interval X, and X is lowered by the basic clock pulse until it reaches to zero, when an interruption occurs.

For priority interruption, two statements MASK and UNMASK are supplied.

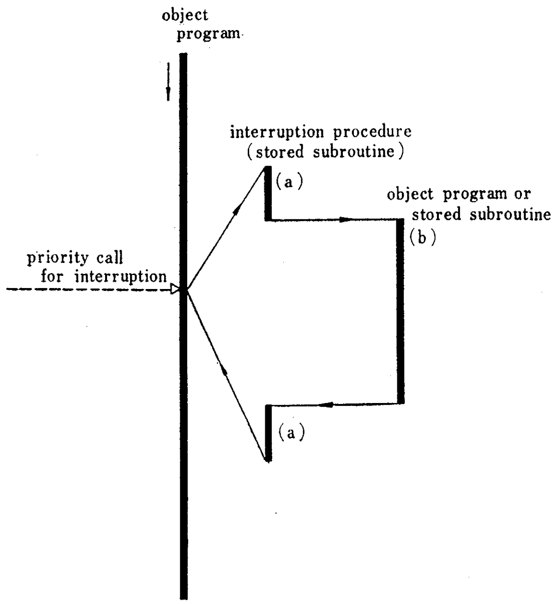

Fig. 2 Interruption procedure.

The interruption procedure is performed by a stored subroutine (a) as illustrated in Fig. 2. The subroutine (a), however, doesn't retire the contents of the working addresses of an other stored subroutine (b) which is just running when the interruption occurs to the undestroyable storage. Further, even in the event that the same subroutine (b) is to be called by two or more levels of the interruptions, in general only one subroutine (b) is stored for common use to reduce the required computing capacity. In these cases, MASK statement is of great use not to disturb the working addresses in the subroutine (b).

In the HODRAL compiler, variables are automatically stored in the data areas. No *DATAS* declaration is neccessary. Memory protection is automatically assured for every 1028 words, if required, in specific 2048 words for every 64 words. Whole memory capacity of HOC-300 is 8192 words.

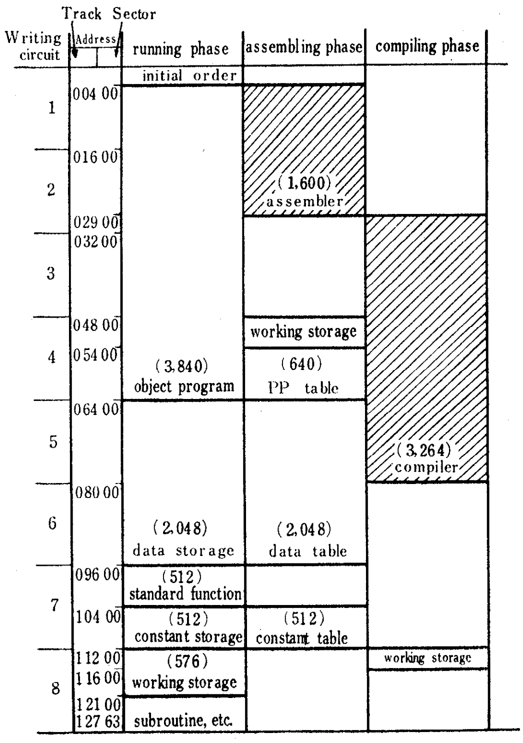

Fig. 3 Assigned blocks of the system.

Optimum allocation faculty of the 1+1 address computer HOC-300 is performed for every 128 instructions. The priority of the considered addresses in optimization of an instruction is

(1) its own operand address,
(2) the operand address of the preceding instruction,
(3) the stored address of the preceding instruction.

References

[1] BROOKS, M.E., Problems in programming control computers. *Automation, 10,* 2 (Feb. 1963), 78–82.
[2] TANAKA, A., et al., Computer control of basic oxygen steelmaking process and problems in its programming. *Information Processing in Japan, 4* (1964), 49–61.