UDC 681. 322-181. 4:681. 3. 06

SOFTWARE SUPPORT OF FUJI MICRO CONTROL SYSTEM

Tsuneyoshi Ohboshi
Masaki Nonaka
Jun'ichi Hiramatsu

I. INTRODUCTION

Products employing micro processor are being developed in large number these days, but debugging and coding programs which lock simple at a glance are surprisingly becoming more time and labor consuming matters due to imperfection of program development use hardware and software system.

The software support system can easily make program development use software with regard to any micro processor and aims at supporting construction of applied products at high efficiency.

Configuration of the system is shown in *Fig. 1*.

Phase I comprises procedure necessary for construction of development software (assembler, simulator).

Phase II represents procedure of making applied products programs.

II. CONFIGURATION OF HARDWARE

An example of standard configuration of hardware related to host computer of support system is shown in *Fig. 2*.

PROM program and ROM simulator are necessary when inserting constructed program into applied products system.

III. CONFIGURATION OF SOFTWARE

The support system consists of the programs shown

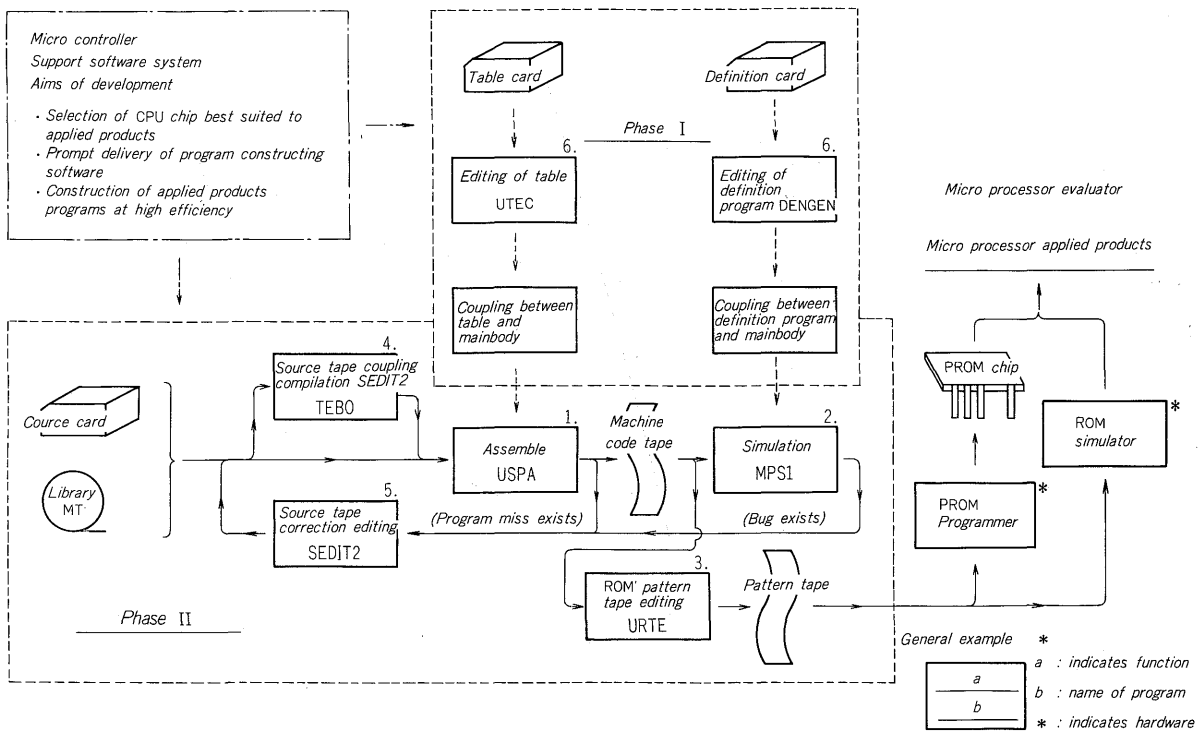

Fig. 1 Configuration of micro controller software support system

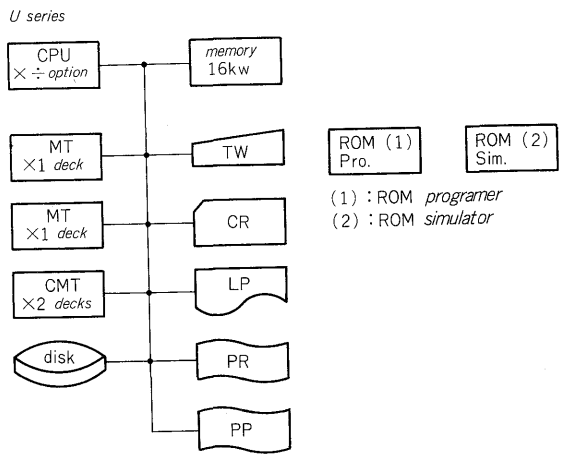

Fig. 2 Construction of hardware system

Table 1 Construction of software system

Control program (UMOS)
- 1) assembler (USPA)
- 2) simulator (USPA)
- 3) ROM tape editing program (URTE)
- 4) source tape coupling program (TEBO)
- 5) source tape correction program (SEDIT)
- 6) tape editing program (UTEC, DENGN)*
- 7) others (FASP, PRBL, FLINK etc.)*

* programs marked (*) are necessary only in phase I.

in *Table 1*. These are described in the following.

1. Meta Assembler

Processing content of assembler remains the same no matter what the architecture of micro processor is, if held to basic content. That is, source text is read, it is checked whether the text is correctly written or not, coded numbers are combined with machine codes, address is attached, and object tape is finalized. Here, format of writing instructions, construction of machine codes, etc. which differ with micro processor shold be edited in the form of a table for future reference.

The meta assembler (named USPA) offers means of easily constructing assembler with regard to any micro processor.

First, lets describe about main points of the type of assembler language of USPA.

(1) Statement can be written in free form from column 1 to column 72 of 80-column card. Statement consists of the following 3 fields.
[Label] : [Instruction] ; [Comment]
[Label] : and [Comment] are written if necessary. Method of writing [Instruction] is defined for every micro processor.

(2) Character set consists of 36 English characters and the following 25 special characters.
_ * , ¥ . + /) (= # ' % " > < ; : ¬ $? @ & ! |

(3) Constants are written as decimal, binary, octal, sexadecimal numbers. As regards 2,8,16 notation, numbers, series of numbers is enclosed with ' and B, O, X, C is attached to the head respectively.
For example, B'01'.

(4) As regards calculation system, addition, subtraction and multiplication are permitted and are indicated by +, –, *.

(5) Label is indicated by English number series up to 8 characters with the head as English character.
Next, main features of USPA are described.
(1) Machine code can have any number of bits. Moreover, construction of machine code can be of any type. By this, it is possible to assemble even in complicated cases like micro code.
(2) For every micro processor, assembler language form can be decided for the ease of reading.
(3) Address of program list and printing form of machine code can be decided as desired.
(4) Related tables and table writing language for construction and compiler program as shown are shown in 1) ~ 3) of *Table 1*.
An example of writing table is given below.

```
BEGIN  ;
UNT    1,8 ; /* BIT/WORD */
ADR    4 ,4 ,4 ,4 ;
FMT    1 =(4 ,4 ) ,2 =( 4 ,4 ,' ',
       4 ,4 ) ;/* FORMAT */
RSV    1 =( HLT , ' 0076') ,
       2 =( RLC , ' 0007') ,
          .
          .
          .
SYN    1 =( 6 / 1 , 1 / 1 · 1 , 3 / 2 ),
       /* OPCODE TYPE 01 */
```

2. Mobile Simulator

In CPU board combined with micro processor application products, since debug panel mechanism (for example, setting and indication of register content, instruction fetch, address, stop, etc.) is not attached at all in almost all the cases, debugging of program, if necessary, is accomplished by playing tricks with program, putting out signal, and operation is verified and investigated by monitoring by oscilloscope or LED. However, since this method takes a long time and much labor and is inefficient, it is necessary to replace it with the method using simulator having plenty of debug functions. By this, efficient debugging can be expected including trace of register content, confirmation of content of input/output data, calculation of number of execution steps, etc. The mobile simulator (called MPSI) offers this highly efficient debug with regard to any micro processor. First, let's describe about main commands (provided from any one of TW, CR, PR) of MPSI.

(1) /TRACE X_1, X_2
If instructions from address X_1 to address X_2 are executed, result of execution (content of register) for each instruction is put out to the line printer. Combinations X_1 and X_2 can be set up to 8 as required.

(2) /FSTOP X_1 [, n]
Instruction of X_1 address is executed n-1 times and nth execution is held. Here, n is less than 255. When abridged, n is considered as 1. Combinations of X_1 and n can be set up to 8.

(3) /SSTOP [$\begin{matrix}I\\D\end{matrix}$,] X_1
When there is writing in X_1 address, the instruction is executed, pertaining address and content and value of program counter are printed out and it waits for the next command.
I, D indicate classification of bus. X_1 can be set up to 8 for each bus.

(4) /RUN [X_1]
Simulation is started from X_1 address. When X_1 is abridged, value of the program counter at that time becomes the initial address.

(5) /CANCL [x [, $\begin{matrix}I\\D\end{matrix}$]]
When operand is abridged, all the previous specifications of TRACE, ESTOP, SSTOP become ineffective. If any one of T, F, S (head latter of command) is written in x, only the specification of the associated command is effective.
I, D are required only for S.

(6) /RDPLY [X_1 , X_2 , ... X_n]
Prints content of register. Register is indicated by mnemonic code X_i . When operand is abridged, contents of all the registers are printed.

(7) /RALT X_1 . Y_1 , X_2/Y_2, ... X_n/Y_n
Modifies content of register. Register is indicated by mnemonic code X_i and modification content is written by sexadecimal number Y_i.

(8) /DTIME
Displays contents of memory reference frequency counter and steps counter.

(9) /ROM [$\begin{matrix}I\\D\end{matrix}$,] X_1 , X_2
From X_1 address to X_2 address is ROM (Read Only Memory) area. If there is writting in this area, its content is printed. I, D indicate classification of BUS and combination of X_1 and X_2 can be set up to 8 for each BUS.

(10) Moreover, commands of LOAD, SINT, INRT, DPLAY, CALT, WATCH, DUMP, etc. are prepared and a very efficient debug is supported.
Next we describe characteristics of MPSI mainly occuring in phase I.

(1) There can be any number of bits of register, memory.
(2) Since standard arithmetic logic calculation, flag, calculation routines are available, imitation of instruction can be easily defined.
(3) Editing program and for table writing program for easily constructing tables, etc. which MPSI refers to are available.
An example of table writing is given below.

```
/ FRMAT
 FMT = 1
 OPE = 0 / 3 , 5 / 3          OP · FIELD
 OP 1 = 2 , 6 / 1             OP · CODE
        .
        .
        .
/ OPSQ
 OSQ = 1
 MOV   ( , = 1 ( 8 ) , A )
 ADD   ( A, B, A )  FLG = ( C 1 = C )
 JMP   OSQ = 1  STEP = 2
 SEND
```

3. ROM Tape Editor

In most of the cases, it is not necessary to rewrite program of micro processor for making it exclusive and since cost is also cheap, it can be housed in ROM or PROM (Programmable Read Only Memory).

Since construction of ROM chip is 4 x 512 bits or 8 x 256 bits, for example, if program of micro processor of 16 bit instruction bit length is housed in these chips, since bit length is adjusted, chips are used in paralle, 4 bits into four or 8 bits into two.

For this reason, the zero-one pattern to be written in ROM chip is required to be collected in the tape following the input type of ROM program after editing the binary tape put out from the assembler.

The ROM tape editor (called URTE) edits the pattern tapes necessary for writing in this ROM chip.

Functions possessed by URTE are described below.

(1) Patterns to be written in ROM are edited from binary tape and put out to paper tape.
At this time, content of tape is printed in the line printer also.

(2) Pattern tape can be edited according to the composition of ROM chip (for example, 4 x 512, 8 x 256, etc.).

(3) Pattern tape can be made according to the input type (for example, BNPF, sexadecimal, binary, etc.).

(4) When editing pattern tape from binary tape, conversion from positive logic to negative logic can be specified.
An example of edit command is given below.

```
URTE KO S = MT, P & BNPF, U = 0/256,
F = 0/8 ;
```

Here, from 0 address to 255 address of the program called KO is edited in the BNPF type pattern tape. In this case, composition of ROM chip is 8 x 256.

4. Source Text Bonder

Program of micro processor is mostly constructed by gathering small groups of routine of a few tens of steps at the most.

As regards these routine groups, while developing a number of micro processor applied products, a number of

such groups is accumulated and a library is made.

Therefore, for making programs at high efficiency, it is necessary to make a program by taking out most usuable routines from the library.

The source text bonder (called TEBO) is used for coupling editing of source texts of these routines.

Functions possessed by TEBO are described below.

(1) Paragraph of the text to be coupled is specified by any one of the combinations of file name, line number, identification code and paragraph name.

Here, the source text specified by paragraph name is confined to,

PART name BEGIN ; and
PART name END ;

English letter series written in name comes before the name of paragraph.

(2) As regards all the English letter series written in the text to be coupled, specification for changing to arbitrary letter series is possible.

Here, it is also possible to change either of the upper, lower part of the English letter series (for example, ABCxxx → XYZxxx).

These are used for change of instruction and change of label and are convenient for rewriting program.

Next, we give an example of coupling statement.

XYZ = ABC (10/20) + BCD (SEC1) ;

Here, coupling is done by extracting paragraph called SEC1 from 10th line to 20th line from the programs called ABC and BCD respectively, and is changed to the program called XYZ. Medium of the source tape is magnetic tape.

5. Source Tape Corresting Program

Correction when logic miss or writing miss of program is discovered is performed by SEDIT.

As regards the method of correcting, there are changing of statement, inserting, erasing, etc. Medium of the source tape is magnetic tape.

6. Table Editing Program

Tables necessary for USPA and MPSI are edited by UTEC and DFNGN respectively.

After this, in order to couple with the mainbody program, some processing is necessary like assemble, linkage edit, etc.

IV. CONCLUSION

An outline of support system is given. Use of micro processor is expected to increase day by day. At that time it will be appreciable if this system is effectively used.