DECUS
PROGRAM LIBRARY

DECUS NO.	8-636
TITLE	BEST - Binary to Symbolic Traductor
AUTHOR	Michel Morel and Francoise Landre
COMPANY	Submitted by: J. A. Gaudron E.N.S.E.E.C. Caen, France
DATE	June 22, 1973
SOURCE LANGUAGE	PAL III

BEST - BINARY TO SYMBOLIC TRADUCTOR

DECUS Program Library Write-up DECUS NO. 8-636

Abstract:

The Binary to Symbolic Traductor accepts a paper tape in a binary format, and produces either a printed listing or a paper tape in ASCII format, acceptable to the PAL III Assembler. It can disassemble 8K programs, with interrupts and FPP instructions. It sorts out instructions from constants, and automatically produces tags at the referenced addresses, which helps the operator to understand quickly any program. The operator can converse with BEST, indicate various starting addresses, and options for outputs (Automatic paging, Pass 3 listing).

I N T R O D U C T I O N

Binary to Symbolic Traductor (BEST) is a general utility program. It will provide a symbolic listing /or punch from a binary tape. PAL III mnemonics (but for EAE instructions) and a symbolic name for the referenced addresses are typed out. It is also conversationnal, the operator giving it indications about the program, resulting from previous and partial disassembling. It is possible to ask BEST to list only small parts of the program, or to search a word within the limits of this program.

Its great advantage over other disassemblers is that it can sort out instructions from constants in a program (which we call the object program), and type out tags for every referenced address , without being given any symbol table. Nevertheless, the operator can give BEST some indications for a more accurate disassembling. The output is such that every translated instruction is actually executed when the object program runs, and the comprehension of this program is made far more easy.

In order to sort out instructions from constants, BEST follows the running of the object program: it looks down the object program from the starting address,(which the operator has to give him) until it finds a JMS, JMP or Skip code. We call the starting address of this routine or subroutine a Branching. BEST "opens" this Branch and goes on looking down (in case of Skips or JMS), or jumps to the referenced address (for JMPs). This "opened" Branch will be explored when the disassembler has ended the present Branch, or has reached a part already explored. When the object program is simple, BEST can find every actual branch, and never mistake a constant for an actually executed instruction.

In a more complicated program, where branchings could be calculated, depending on the result of a test or of some computing:

```
          TAD NBR
          TAD JPICOM
          DCA .+1
          Ø
JPICOM,   JMP I .+1
          BR1
          BR2
           .
           .
           .
```

NBR may be the result of some computation ; if NBR = Ø, an effective JMP BR1 is executed ; if NBR = 1, an effective JMP BR2 is executed.

BEST cannot guess what values NBR could take, so it does not open any branching, but issues a message warning the operator that a program modification has occured, and at what location. The operator, after examining this small part of program (BEST will list it when requested), will give BEST the actual Branchings.

When BEST has correctly sorted out all instructions from constants, it gives names to the referenced addresses, and is ready to type or punch out a complete and easy to read listing, which enables the operator to draw the flowchart and understand the object program as easily as if it had been the real symbolic listing of this program.

=-=-=-=-=-=-=

I - REQUIREMENTS

1/ Equipment

PDP-8 with extended memory (at least Field 1), ASR 33 Teletype, and High-Speed Reader.

2/ Storage

BEST requires the whole Field Ø for itself. The object program is stored in Field 1 by the Disassembler.

The Binary Loader should be placed in Field 1. If ever the object program had overlapped the Bin Loader, replace the latter in core memory, using the input routine of the Disassembler, at the end of processing.

II - LOADING PROCEDURE

1- Place the BEST tape in the reader.
2- Set the Switch Register to 7777, and press LOAD ADD.
3- If using the High-speed reader, set Bit Ø = Ø.
4- Press START.

III - USING THE PROGRAM

In short, BEST loads the object program, asks for indications, gives information about what it found in the program, gets ready to list all or part of the translation; after investigation, the operator can restart BEST and give it further indications to improve that translation.

1/ Starting procedure

The starting address of BEST is Ø2ØØ (Field Ø).

a- Place the object program tape in the High-speed reader.
b- Set Ø2ØØ on the S.R.; press LOAD ADD.

c- The object program may be a 8K program. The Disassembler will consider separately what had to be loaded in Field Ø, and in Field 1, and will load that in its real locations .

Set Bit 11 to 1 or Ø, according to the part you want to be loaded.

d- Press START

If no tape is in the HSR, BEST will disassemble what has been read in a previous pass, and jump to the initial dialogue.

The tape is now read. If the program consists in several tapes, place the next tape in the reader and press CONTINUE. When the object program is read, remove all tape from the reader, assume that content of AC = Ø (if not, start operation again from the begginning).

e- If the object program is in 8K, and the part of the program that has not been loaded has previously been disassembled, change the value of Bit 11. Thus, if you change Bit 11, BEST will disassemble the parts that are called from the other field, when the object program runs.

If some floating point instructions may occur in the program, set Bit 3 to 1, to have these instructions translated.

(Be careful: some versions of PAL III do not accept FPP mnemonics such as SQUARE, SQROOT, etc; you will have to define them with SQUARE = 1, for instance, before any assembling of the paper tape provided by BEST)

f- Press CONTINUE.

2/ Initial dialogue

BEST will then issue the four following questions on ASR 33: Branching, Instructions, Data, Arguments, which you shall answer this way:

BRANCHING

Give the starting address of the branches (on first disassembling, give only the starting address of the object program), each followed by a RETURN. BEST echoes a Carriage Return/Line Feed. When all are given, press the SPACE key,

BEST will then ask:

INSTRUCTIONS

Some constants are instructions bound to be used for modifying the program. If you want them to be turned into symbolic mnemonics to ease the comprehension of the object program, give their addresses the same way as for the branchings.

This does not affect correct disassembling.

DATA

For some subroutines which want a variable number of arguments, it is important that BEST should not mistake these arguments for instructions, as placed after a JMS. You then have to place the address of the first of these arguments in the DATA list, the same way as for the branchings. You should be careful as to "reopen" the location corresponding to the actual return from the subroutine by placing it in BRANCHING.

ARGUMENTS

If a subroutine has a fixed number of arguments, you enter first the starting address of the subroutine, then a RETURN (BEST echoes a CR/LF), then the number of arguments. If other argumented subroutines are to be found, press RETURN (echo CR/LF) and enter the next. Then, press the SPACE key.

If any mistype has occured, press SHIFT/O, a back arrow (←) will be echoed (no CR/LF), and the number ignored.

3/ Messages

Messages are issued by BEST when the object program is likely to have been modified in a way BEST could not follow.
The operator will have to look down at the list of messages, and interpret them in the order of output: if BEST made a mistake, a message will keep track of this mistake, that can have caused a misinterpretation of constants as instructions, which in turn can cause some other messages to be issued. These are of course nonsensical, and need not be considered.

Thus, the operator must look up first the first messages, which are likely to be right.

After all the messages are issued, the operator waits for BEST to be in command mode (bell rings), and examines the disassembled object program (See Commands). After that, he can restart BEST and give it indications, in the initial dialogue, according to the messages.

IR (Illegal Reference)

BEST has translated a 4--- (JMS) or 5--- (JMP) code, and the referenced branch is not within the limits of the object program.

For instance, if the object program is located in pages Ø, 1, 2, and BEST finds:

```
*235
          5767      /JMP I 367
*367
          7402
```

the following message will be issued:

IR 7402 AT 235

There obviously is a mistake, for there is not any instruction of the object program in location 7402. BEST does not consider this false branching.

What to do

There are two reasons why BEST can issue the IR message:

- The object program was punched on several (two or more) paper tapes and you forgot to load one.
(N.B.: if Bit 3 is set to 1, the JMS I 7 entering FPP will not be taken as an IR, even if FPP is not loaded.)

- More likely, there has been an argumented subroutine (i.e.: the content of the location following the JMS SUBR is a constant used in the subroutine, taken by a TAD I SUBR), and you did not warn BEST, which understood the argument as an instruction. In this case, there must be a SI message matching this one (see below).

Ask BEST to type out a small part of the program before

the faulty location, and try to understand what misleaded BEST. It is probable that you will have to put the faulty location in DATA, when restarting BEST.

SI (Suspect Incrementation)

BEST found an ISZ, the operand of which was the return address of a subroutine.

```
        .
        .
*23Ø
        432Ø      /JMS 32Ø
        .
        .
*32Ø
        Ø
        .
        .
        232Ø      /ISZ 32Ø
        .
        .
        572Ø      /JMP I 32Ø
```

This shows you that the return address of the subroutine is modified, then the content of the location following the JMS is likely to be an argument.

What to do

Examine carefully the subroutine in order to know how many times this return address is incremented.
If the number of arguments is fixed, place the address of the subroutine and the number of arguments in Arguments, the next time you restart BEST. You need not reopen the actual return address of the subroutine.
If the number of arguments is not fixed, see DATA.

MA (Modified Address)

BEST found a DCA code, the operand of which was among the instructions (i.e. in a branch). The program is likely to have been modified (See example in the Introduction).

What to do

If an instruction is deposited in the program, for instance a JMP, BEST will recognise it and open a new branching. But the

operand of this JMP might have been the result of a complex computation, which BEST could not understand. The operator will have to give BEST the branchings resulting from that computation. If he wants the deposited instruction to be translated, he will have to place its address in Instructions, when restarting BEST.

TF (Table Full)

When sorting instructions from constants, and giving symbolic names to the referenced addresses, BEST builds internal tables. One of these tables might overflow, then BEST will issue the following message:

TF nnnn

nnnn being the starting address of a table, which has been overlapped by the precedent.

Here are the starting addresses of the various tables:

42ØØ (ARG) Subroutine addresses and number of arguments (filled by operator)

43ØØ (SAVECH) Branchings caused by external field calling (internal table)

44ØØ (OUVCH) Branching in the other field (internal table)

45ØØ (BRANCH) Limits of branches (internal table)

52ØØ (OUV) Branchings (filled partly by the operator and BEST)

5¢ØØ (BUFFER) Interrupt output buffer (internal table)

From 5¢4Ø to 7776, BEST builds tables, one after the other, without a fixed starting address. They contain the symbolic names of the referenced addresses. A TF 7776 may occur if there are too many referenced addresses.

What to do

It is highly improbable that a table might overflow. If you have to disassemble a very intricate program, we suggest that, if a table overflows, you should separate the object program in two relatively independent parts, the links between which you know well, and disassemble them separately.

4/ Switch Register Options

Bit Ø	Ø	Normal Output
	1	BEST issues a listing similar to PAL III Pass 3 listing.
Bit 1	Ø	No effect
	1	Automatic Paging, with a line of hyphens separating each page, a full peal of bells warns operator that the page is ended, and BEST waits 2 seconds for the operator to tear out the page.
Bit 2	Ø	No effect
	1	Stops output, BEST is ready to take new commands.

5/ Commands

BEST will echoe a CR/LF after each command letter. After a command has been executed, wait for the bell to ring (command mode) before giving another command.

L (List)

L: Lists the entire object program.

nL: BEST lists content of address n, with a tag if the address is referenced.

Ex: 235L

```
BR235,  CLA CLL          Ø235  73ØØ  BR235,  CLA CLL
(Bit Ø = Ø)                        (Bit Ø = 1)
```

n,mL: BEST lists the contents of addresses between n and m, included.

P (Punch)

The limits are given the same way as for L command. BEST issues symbolic tape (Bit Ø and Bit 1 = Ø, if the tape is to be assembled), with Leader-Trailer, PAUSE and CTRL/Form at the end.

Give the limits of what you want to be punched out, press P, and immediately turn ON the punch.

/ (slash) has the value of the upper location used by the object program. It can be used in any command using an address number.

RETURN

Causes BEST to list or punch out the content of next address.

<

Lists content of previous address.

W (Word Search)

nW: Best gives the number of the addresses the content of which is n.

G (Go)

After listing part of the program, if the operator wants to give further information to BEST (open new branchings, for instance), he can restart BEST by typing G, Which causes an effective JMP 2ØØ.

If any mistype occurs, press RUBOUT, a back arrow (←) and a CR/LF will be echoed, and the number ignored.

6/ Symbolic names

BEST gives symbolic names to the referenced addresses ,this name is composed of a two letter code, and at most four numerals giving the location of the referenced address. BEST choses the names according to what instruction referenced the address, with that order of priority:

instruction	letter code	signification
JMS	SB	Subroutine
JMP	BR	Branching
JMS I	SL	Subroutine link
JMP I	BL	Branching link
TAD n	MD	Modification

(if n was placed in Instructions)

DCA m	CH	Change of program

(if m is the address of an instruction recognized by BEST)

All the other referenced addresses are called, according to their content:

CT (constant) if content is ≠ Ø

VR (variable) if content is = Ø

whenever a JMS I SL or a JMP I BL occur, the effective SB or BR is given:

```
JMS I SL5330      /SB5420
JMP I BL320       /BR5453
```

For any indirectly referenced address, the effective address, with its symbolic name, is given after a slash.

For the constants, the octal value of the constant id typed out after a slash. The value of FPP constants is given by the octal value of the three locations of the constant: exponent, high and low order mantissa.

7/ Execution Time

Computing time (search of branches, attribution of symbolic names) depends on the complexity of the object program. The maximum computing time we have experienced is relative to the FORTRAN Compiler (more than 4000 instructions). It took 35 mn on a PDP-8/S, and would take 3 mn 30'' on a PDP-8/I, after giving BEST several indications.

The output time is limited by the teletype, to 10 characters/ second.

8/ Restrictions

Restrictions could be made on account of the size and degree of intrication of the object program, that could cause a table to overflow.

We have experienced disassembling of several programs, the biggest of which was the FORTRAN Compiler, and no table did ever overflow.

9/ Additionnal facilities

- Some users of BEST have pointed out to us a drawback in our method for giving indications: the operator has to type all of them, every time he restarts BEST.

In order to remedy that, the operator can give all the indications with the Low-Speed Punch ON. (Turn it ON before starting or restarting BEST). When Best is restarted again, put the tape in the LSR, wait for BEST to ask for BRANCHING, type the new ones, and read the old ones from the LSR. When BEST asks for DATA, turn the reader OFF, give the new ones and restart the reader, and so on until all indications are given.

If you want to erase one indication, correct the indication tape with the Symbolic Tape Editor, which you can place in Field 1. You will then have to read again the object program.

- When disassembling is through, and you want to punch the symbolic tape out on a High-Speed Punch, Stop the computer, load the High-Speed Punch Patch with the Binary Loader and restart BEST in 2ØØ. BEST will then be in command mode and you can ask for a punch out.

(N.B.: Commands will have no echo on the Teletype.)

1Ø/ Example

```
                *2ØØ
Ø2ØØ  6Ø46                    TLS
Ø2Ø1  4216                    JMS CRLF
Ø2Ø2  1213                    TAD M1Ø
Ø2Ø3  3214                    DCA CNTR
Ø2Ø4  4615                    JMS I PRINTL
Ø2Ø5  Ø255                    255
Ø2Ø6  2214                    ISZ CNTR
Ø2Ø7  52Ø4                    JMP .-3
Ø21Ø  4216                    JMS CRLF
Ø211  74Ø2                    HLT
Ø212  5211                    JMP .-1
Ø213  777Ø    M1Ø,            -10
Ø214  ØØØØ    CNTR,           Ø
Ø215  Ø4ØØ    PRINTL,         PRINT
Ø216  ØØØØ    CRLF,           Ø
Ø217  4615                    JMS I PRINTL
Ø22Ø  Ø215                    215
Ø221  4615                    JMS I PRINTL
Ø222  Ø212                    212
Ø223  5616                    JMP I CRLF
                *4ØØ
Ø4ØØ  ØØØØ    PRINT,          Ø
Ø4Ø1  16ØØ                    TAD I PRINT
Ø4Ø2  22ØØ                    ISZ PRINT
Ø4Ø3  6Ø41                    TSF
Ø4Ø4  52Ø3                    JMP .-1
Ø4Ø5  6Ø46                    TLS
Ø4Ø6  72ØØ                    CLA
Ø4Ø7  56ØØ                    JMP I PRINT
```

This program causes a CR/LF, ten hyphens, and a CR/LF to be typed out on the Teletype. It shows an example of argumented subroutine.

2ØØ was put in BRANCHING. Here is what BEST issued:

```
SI Ø4ØØ  AT Ø4Ø2

L

*2ØØ
            TLS
            JMS SB216
            TAD CT213       /777Ø
            DCA VR214
BR2Ø4,      JMS I SL215     /SB4ØØ
            AND 255
            ISZ VR214
            JMP BR2Ø4
            JMS SB216
            HLT
CT212,      JMP .-1
CT213,      777Ø
VR214,      Ø
SL215,      SB4Ø0
SB216,      Ø
            JMS I SL215     /SB4ØØ
            AND SL215       /4ØØ
            JMS I SL215     /SB4ØØ
            AND CT212       /5211
            JMP I SB216

*4Ø0
SB4ØØ,      Ø
            TAD I .-1
            ISZ SB4ØØ
            TSF
            JMP .-1
            TLS
            CLA
            JMP I SB4ØØ
```

There is a suspect incrementation at 4Ø2. The subroutine uses one and only one argument. The operator will then put 4ØØ, and 1, in ARGUMENTS.

Here is the final issue, in the Pass 3 listing form. One can compare it with the original symbolic text, to realize how clear it is

ARGUMENTS:

```
4ØØ
I

SI Ø4ØØ AT Ø4Ø2

L

                    *2ØØ
Ø2ØØ    6Ø46                TLS
Ø2Ø1    4216                JMS SB216
Ø2Ø2    1213                TAD CT213       /7777Ø
Ø2Ø3    3214                DCA VR214
Ø2Ø4    4615        BR2Ø4,  JMS I SL215     /SB4ØØ
Ø2Ø5    Ø255                255
Ø2Ø6    2214                ISZ VR214
Ø2Ø7    52Ø4                JMP BR2Ø4
Ø21Ø    4216                JMS SB216
Ø211    74Ø2                HLT
Ø212    5211                JMP .-1
Ø213    7777Ø       CT213,  7777Ø
Ø214    ØØØØ        VR214,  Ø
Ø215    Ø4ØØ        SL215,  SB4ØØ
Ø216    ØØØØ        SB216,  Ø
Ø217    4615                JMS I SL215     /SB4ØØ
Ø22Ø    Ø215                215
Ø221    4615                JMS I SL215     /SB4ØØ
Ø222    Ø212                212
Ø223    5616                JMP I SB216

                    *4ØØ
Ø4ØØ    ØØØØ        SB4ØØ,  Ø
Ø4Ø1    16ØØ                TAD I .-1
Ø4Ø2    22ØØ                ISZ SB4ØØ
Ø4Ø3    6Ø41                TSF
Ø4Ø4    52Ø3                JMP .-1
Ø4Ø5    6Ø46                TLS
Ø4Ø6    72ØØ                CLA
Ø4Ø7    56ØØ                JMP I SB4ØØ
```