NOSY: A CORE-SAVING OPERATING SYSTEM

Ivan L. Finkle
H. L. Scantlin

July 1965

P-3168

PREFACE

This Paper describes an operating system designed to provide more core storage for the programmer than is available when using the IBM system, IBSYS. This was achieved by sacrificing efficiency of input/output and, in the RAND system, at the expense of decoupling two machines and using only one. Therefore, it should be used only for programs which cannot run under IBSYS because of core storage requirements.

NOSY: A CORE-SAVING OPERATING SYSTEM

Ivan L. Finkle*
H. L. Scantlin

The RAND Corporation, Santa Monica, California

1. INTRODUCTION

Converting codes from FORTRAN II to FORTRAN IV causes loss of core storage, because

1) The IBSYS nucleus requires about 12700_8 words of storage (this includes IOCS);
2) The FORTRAN IV compiler in most cases produces more instruction words than FORTRAN II.

The combined effect of these losses is that many large codes "EXCEED AVAILABLE STORAGE". (A diagnostic message now familiar to programmers everywhere.)

Breaking these large codes into chain links and using the chain feature solves most of the problems. A few exist, however, whose largest chain link still "EXCEEDS AVAILABLE STORAGE". One RAND code in particular (a two-dimensional hydrodynamics model) continued to defy the chaining method and stubbornly refused to fit into available storage. We decided then to eliminate the nucleus (during execution only), simplify input/output, and pass the core storage savings on to the programmer.

---

*Any views expressed in this paper are those of the authors. They should not be interpreted as reflecting the views of The RAND Corporation or the official opinion or policy of any of its governmental or private research sponsors. Papers are reproduced by The RAND Corporation as a courtesy to members of its staff.

RAND's computing system is an IBM 7040 coupled to an IBM 7044. The 7040 is the I/O processor, while the 7044 is the job processor. In order to minimize our "non-system," the two machines are uncoupled during the execution of a "non-system" application.

We call the resulting system, or non-system if you prefer, NOSY. NOSY runs in IBSYS; that is, it can be run as an IBJOB application. Most of IBSYS is still available to the user: FORTRAN IV, MAP, the loader, the subroutine library, the chain feature, the reload feature, and most IBSYS control cards. The IBSYS I/O system, IOBS, is not available. The reload feature is an integral part of the system, as an absolute binary deck is automatically prepared and written on tape. This tape can be used repeatedly. In the example of the two-dimensional hydrodynamics code, $13{,}000_{10}$ words of core were returned to the program, which now runs without chaining, as an IBJOB application.

Section 2 describes the "non-system" system and discusses its restrictions and limitations.

2. NOSY SYSTEM DESCRIPTION

NOSY is a minimal system designed for programs with small amounts of input and output, but requiring a great deal of storage space. It has its own I/O system, EIEIO, which allows input and output of type 1* binary tapes with 128-word physical records, and output of BCD information. BCD input is not permitted. However, the user may write an additional program which reads BCD input and prepares

*IBM Form C28-6309 (IBM 7040/7044 Operating System, 16/32K, Input/Output Control System), Record Formats.

a binary tape for later processing. This program would precede the user's main program. Additional restrictions are that all BCD output must be on the standard output tape, and no provision for variable format statements exists. EIEIO does not overlap I/O operations, so programs with significant I/O requirements should not use this system. Jobs run under NOSY load at 100_8. During execution of the user's program, NOSY requires about 2000_8 words.

NOSY operates as follows: The user's job is compiled under IBSYS. The $IBJOB card must specify NONSYS, COPY=U10, and NOGO; it may specify other standard options as well.* Specifying NONSYS is similar to specifying a level of IOCS (e.g., IOOP1 or IOCS). This causes the program to have a loading address of 100_8. IBLDR was modified so that "NONSYS" replaces "IOOP1" as an IBJOB option and the absolute origin of 100_8 results. The COPY=U10 option results in an absolute binary tape being prepared for loading later. NOGO must be specified on the IBJOB card, because the job is to be run under NOSY rather than IBSYS. EIEIO, the user's deck, and special versions of POSTX, FPT, and chaining routines, are all loaded together.

In addition to their usual functions, the special versions of POSTX and FPT contain trapping routines for an illegal instruction, interval timer overflow, and STR trap, and print error messages when these or floating-point traps occur. These error messages plus I/O error messages printed

---

*IBM Form C28-6318 (IBM 7040/7044 Operating System, 16/32K, Programmer's Guide), Control Cards.

by EIEIO give the programmer almost the same diagnostics as the full IBSYS system.

NOSY is run as the second IBJOB in the job. It inhibits channel traps and writes itself and IBSYS out onto tape. It then zeros all of storage except itself, loads the user's code from the COPY(U10) tape, zeros itself, and turns control over to the user's code. When EXIT is called, the special version of POSTX reloads NOSY which in turn reloads IBSYS, from tape, and returns control to it.

Each output statement in the user's code causes EIEIO to write two logical binary records. These are used later, in the output phase, to produce BCD output. The instructions into which the format statement was compiled are written as the first record. They are altered only in that all TSX's, TSL's, and TRA's to conversion routines are replaced by the name in Hollerith of the routine referenced. All the words in the list are written in the next record. This avoids having the I/O conversion routines in core storage during execution. A third IBJOB procedure, AEIOU, then converts this binary tape to BCD output. It reads the format information first, reconstructs the format statement, then reads the list and writes it in BCD according to the reconstructed format statement. This third IBJOB is a normal IBSYS job.

3. SUMMARY

NOSY is a four-phase system:

1) Conversion of BCD input tape to binary tape;

2) Loading and preparation of absolute binary tape;

3) Execution;

4) Output (conversion of binary tape to BCD output).

IBSYS is purged from core during the execution phase only. This is, of course, the main purpose of the system. Also during the execution phase, the 7040 and 7044 are virtually disconnected, so that the 7040 will eventually hang up, waiting for the 7044 to assign the next 7040 task. This signal cannot be sent until the output phase, when IBSYS is back in core.

At an installation where machines are not coupled and where IBSYS consumes too much core, this type of three-phase system should be considered. At least such a system can be included and run optionally under IBSYS. The effect on running time is difficult to assess. I/O operations are slower; however, additional core storage often reduces running time.

4. PROGRAMMING INSTRUCTIONS

NOSY

NOSY is normally the second IBJOB. It writes itself and the system on tape, then loads the user's job. It is recalled by NSPSTX and returns control to the system. A typical job setup might be:

```
1       8          16
$JOB               2090,NOSY,S1540,3,100,100,C
$CLOSE             S.SU10,REWIND
$IBJOB             NONSYS,COPY=U10,NOGO
(User's deck)
EIEIO deck  \
NSPSTX deck  |
NSFPT. deck  } NOSY SYSTEM PACKAGE
IOU deck     |
NSBOBK deck  |
NSCH deck   /
$ENTRY
$IBJOB NOSY
  (NOSY deck)
$ENTRY             NOSY
$IBJOB AEIOU
  (AEIOU deck)
$ENTRY             AEIOU
$IBSYS             ENDJOB
```

An IBJOB application may be inserted immediately after the $JOB card to produce a binary tape for input. Since the binary tape must be type 1 with 128-word records to enable EIEIO to read it, a file card is necessary. The insertion looks like:

```
1       8          16
$IBJOB
$FILE              'FTCxx.',Uxx,*,TYPE1,BLOCK=128,LRL=127,SINGLE,
$ETC               ERR=RERRx.,EOR=REORx.,EOF=REOFx.,REEL
   (User's code for converting BCD input to a binary
    tape)
$ENTRY
```

Use tape C4, C5, or C7 for this purpose as these are not used by IBSYS or NOSY.

RESTRICTIONS

1) All BCD output statements to be printed by AEIOU must use FN 6 or be a PRINT statement;

2) Variable formats are not permitted because the routines necessary to compile them are not in core;

3) No BCD output list may exceed 23,040 words;

4) Double precision is not available.

The tapes are assigned as follows:

FN logical	TAPE	
0	C0	Used to keep NOSY and IBSYS on. NOT AVAILABLE TO USER.
1	C1	Used by IBSYS, available during execution.
2	C2	
3	C3	
4	C4	
5	C5	
6	C6	Standard BCD output tape.
7	C7	
8	C8	
9	C9	User's code tape.

NSFPT

NSFPT initializes the following trap cells:

a) Floating-point traps,

b) Illegal instruction traps,

c) STR traps,

d) Interval timer overflows.

The floating trap routine is the same as the RAND FPT routine, and is not suitable for double precision. The user can get control when an interval timer overflow occurs by using RAND routine RSW035.

Error conditions are treated as in IBSYS,* the following message being printed:

ERROR NO xx AT LOC xxxxx.

The error numbers are the same as for FORTRAN IV and the same action is taken. The following error conditions have been added:

100 Illegal instruction trap,
101 STR trap,
102 Interval timer overflow.

5. OPERATING INSTRUCTIONS

When NOSY is executed, the message "THIS IS A NONSYS JOB" will be typed on the on-line typewriter. The following procedure covers difficulties between this point and the end-of-job indication.

1) Try transferring to 63_8,
2) If that fails, the user has clobbered core. In this case, enter the following instructions into storage and transfer to 74_8.

into	73_8	load	377777000100	IORD	100_8,,-1
"	74_8	"	076200003232	RDS	CO
"	75_8	"	054100000073	RCHC	73_8
"	76_8	"	006200000076	TCOC	*
"	77_8	"	002000000101	TRA	101_8

*IBM Form C28-6339 (IBM 7040/7044 Operating System, 16/32K, System Programmer's Guide), Error Exits.