SAS on 32-Bit Mini-Computers
Richard A. Usanis, SAS Institute Inc.

Introduction
Good morning and welcome to this session on the SAS system on 32-bit mini-computers. There is a handout containing major differences between the Portable SAS system and the 1982 version of the SAS system. The differences are changes to the SAS language, most of which will be implemented in the ISM version in 1984.

I am speaking to you today as Director of the Portable Systems Division at SAS Institute. In order to understand my role in the development of the Portable SAS system, I’ll mention my background. Prior to coming to the Institute in September 1982, I was the Director of the Computing Center at North Carolina State University involved in software, hardware, and user interface. Prior to 1974, I was a private consultant in statistics and computing with research companies in the Research Triangle Park. I received my Ph.D. in quantitative genetics and forestry from N. C. State University in 1972 where I was working as the Statistical Analysis System. My first introduction to the SAS system was discussions with its founders as far back as 1968 when I was a programmer with the Genetics Department. I have been closely tied with SAS software and SAS users for many years and hope to use this experience to provide you with another good SAS product.

Also here from the Portable Systems Division of SAS to answer any technical questions during the demonstrations are: Carl Thorne, Manager of the Digital Equipment Corporation VAX Systems; Jeff Poizin, Manager of the Data General MV Systems; and Mark Watson, Systems Programmer in the Compiler Writing Group. The Portable Systems Division includes eight other staff members in Cary.

People here in New Orleans have asked me three questions almost constantly:

1. When is SAS going to be available on mini-computers? Followed up by:
2. What will it look like and how will it perform?
3. When will it be available on micro-computers?

The first I will address in this talk, and I think you will see that there is only one SAS software system. With regard to a version for micro-computers, I can say that the current SAS system at its present level of complexity is not adaptable to the smaller systems.

In the next 25 minutes, I am going to review the history, objectives, requirements, and constraints that we have had to work with in developing the system; give you an overview of system components and provide a rather detailed list of differences between the present SAS system and the Portable SAS system. Finally, I will present a schedule we hope to meet over the next few months and discuss a little about the demonstration you can see here at SUGI.

Portable means different things to different people. To Webster, it means: “capable of being carried; easily moved or carried; and, endurable, supportable.” To the computer world, it usually means that the equipment itself is movable or carriable. During the last five years, when we came into the micro-computer era, portable has come to mean that software can be moved across different computer hardware systems and different operating systems. The Portable SAS system is the SAS system that can be moved across different makes of computers and operating systems, in particular, several major 32-bit mini-computers.

As an aside, let me say that mini-computers are really a lot of fun to work with. They are of different architectures and have different problems from main-frame computers, but they are also somewhat friendlier. You can get into a much more intimate relationship with them. However, they are also not as powerful as their manufacturers would like to believe; they have many non-standard, non-compatible features among them that cause considerable headache in transporting code from one to another, even code that each says they will support. This has become evident during development of the Portable SAS system when we manipulate large amounts of data and analyze and perform many calculations with them. After this brief digression, I think it best to present the history of the project and tell you more about what we have accomplished.

History of the Project
This project began because future-oriented users over the years like you have requested that we put the SAS system on mini-computers. It has become the number one item on the SASware Ballot. With that in mind, the Institute last year committed the necessary resources to develop the SAS system for 32-bit mini-computers.

The first person was hired in February of 1982 and soon additional staff members were hired. During the first quarter, the people at the Institute and those hired for this project developed the grammar and procedure conversion specifications for the new system.
In the second quarter of 1982, the Portable SAS system was designed and more staff members were hired to begin coding the components of the system. In addition, the conversion process for the procedures presently running under the IBM version was designed, and the developers of the procedures began going through this process.

The third quarter brought new procedure interface routines for the IBM system. These routines allowed the development of a single set of procedures for both portable and present systems. This minimized maintenance and maximized the independence of the PROC and DATA steps. After the interface routines stabilized, the procedures were changed to use the IBM system. At the same time, a primitive DATA step was developed and demonstrated on the Data General MV 8000 System that we have at the Institute.

During the fourth quarter, the DATA step was demonstrated on the in-house Digital VAX 11/780 System running under VMS and by the end of December was almost 90% complete. In addition, 25 procedures were working on this hardware.

Overall, the Portable systems staff has spent approximately eight man-years on this project and has brought it a long way toward its completion. This is, I believe, a great tribute to the Portable systems staff and their ability to handle this significant project well.

Project Objectives
Three major objectives of the Portable SAS system project are:

1. Keep the system independent of hardware architecture. To this end, 90% of the code is now hardware-independent, and we do not foresee that more than 20% of the code will ever be hardware-dependent.

2. Keep the system independent of operating systems. We are now almost totally independent of operating systems. The system is written in PL/I subset G; it should run under most systems that have this compiler.

3. Have only one set of procedures for present and future systems. This has been accomplished to such a degree that when procedures compile on the IBM system, they will also compile on any of the mini-computer systems.

Thus, we have gone to great lengths to minimize the maintenance and maximize the independence of the Portable SAS system. This independence and maintainability should provide you with improved ability to use SAS no matter what equipment you have available.

The Base Requirements
Our overriding requirement in developing the Portable SAS system was to maintain the features available within the present SAS system. In addition, we wanted it to look the same as the current SAS system to the end user. It was designed for a 32-bit mini-computer with one megabyte of real memory and a virtual operating system. Although the initial product may not perform well in this environment after it is tuned for both performance and size, we believe you will be pleased with the system. It has been written in the AN/SI PL/I subset G, which makes it transferable to other systems.

The key to Portable SAS system development was to make it as portable to other machines as possible while maintaining the features and appearance of present SAS software. We have essentially accomplished this.

Overview of System Flow
In its simplest form, the Portable SAS system has three components: the grammar section, the compiler section, and the interpreter section.

The grammar is the formal definition of the SAS language. There are separate grammars for the DATA step, the PROC step, and the global statements. There are several programs that process the grammar and produce input files for the compiler. It is written in a manner similar to the Backus-Naur Form and contains information about the syntax and semantics of the language. The compiler, the major component of the second phase, uses these files, director sets, and parsing tables to read the SAS input and produces pseudo-code, or "S-code." The compiler itself is two-pass, non-backtracking, and deterministic in its mode of operating, which has several advantages and some disadvantages in the development of the product. The third component of the system, the interpreter, is a system of programs which obtains the S-code from the compiler and interprets each instruction as it receives it, producing SAS output.

Figure 1 presents a more complete picture of the Portable SAS system. Input from the user comes into the operating system through the host interface to the Portable SAS supervisor. The tokenizer obtains it and directs statements to the syntaxing section of the compiler, the PROC interfaces, or the global interfaces. The main compiler takes the tokenized version of the source code, analyzes the syntax, and executes semantic actions. This produces S-code instructions, which are passed to the first phase of the runtime interpreter, that decodes the instruction. After decoding, the instruction is executed; in other words, it does a mathematical calculation, performs a branch or I/O function, or does one of several other operations. After execution, the results are stored in the appropriate place for handling by the supervisor. All of the components on the top half of the figure are interfaced to the Portable SAS supervisor, a general interface to the host system. The host system interface contains the major portion of the non-PL/I code that we have written. These segments are primarily for
Overall system flow is shown in a different form in Figure 2. The compiler is a "traffic cop" for the source code. Using the output from the grammar subsystem, it directs source statements to either the PROC division or the S-code generator. In both cases, they are interpreted and outputted through the SAS supervisor.

I end this brief description with a few statistics (Figure 3) about the Portable SAS system. These figures represent completion of approximately 80% of the DATA step, 50% of the PROC step, and 50% of the SAS/GRAPH product. On the VAX, the present working set size is approximately 650 kilobytes of memory. The system contains a tremendous amount of code, and we expect these figures to increase. The final product with all procedures in the base product and SAS/GRAPH will be approximately 325,000 lines of source code and generate a sharable executable image that will require three megabytes of disk space.

Figure 3: System Size Statistics on January 11, 1983

<table>
<thead>
<tr>
<th>Disk</th>
<th>Source Lines</th>
<th>Source Modules</th>
<th>Image Kb</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA</td>
<td>7317</td>
<td>146,340</td>
<td>722</td>
</tr>
<tr>
<td>PROCs</td>
<td>3501</td>
<td>70,021</td>
<td>78</td>
</tr>
<tr>
<td>GRAPH</td>
<td>541</td>
<td>10,613</td>
<td>123</td>
</tr>
<tr>
<td>Total</td>
<td>11359</td>
<td>226,974</td>
<td>923</td>
</tr>
</tbody>
</table>

Machines and Operating Systems
In-house we have a Data General MV 8000 running AOS/VS, a Digital Equipment Corporation VAX 11/780 running VMS, and a Prime 550 computer running Primos. We plan to support all the computers in these series for these manufacturers. In addition, we are reviewing the possibility of supporting other manufacturers, including WANG, Hewlett-Packard, and UNIVAC, and operating systems such as UNIX. We will also be looking for a PL/I compiler for our use to develop SAS for almost any computer system in the market place. We are also looking at hard-code generation of the portable system to make it much more efficient for your use. This is a longer term project and will probably not be completed during the coming year, but you can look forward to significant improvements as we go through the next year.

Language Differences between Current and Portable SAS Systems
During the design and development of the Portable SAS system, we have found it advisable to make many changes and a significant number of additions to the SAS system to make it a more useful and more powerful programming language and data analysis system. I will present these changes to you in some detail; and if you have any questions about them, please feel free to ask about them after this presentation. I will first present the additions we have made to make your life easier, then the deletions from the language followed by the changes that have been made to statements. The last are mostly enhancements to the language, many of which you requested.

Recall that your number one request in the SASware Ballot this year was "Explicit Indexing of Arrays." This will be provided in the Portable SAS system. We will also have two-dimensional arrays in our production version. Other items that are on the SASware Ballot that will be found in the Portable SAS system are: the ability to use the DROP= and KEEP= data set options along with other options; having format justifiers; and, as we will now see, an ATTRIBUTE statement that allows specifying all the information about a variable in one DATA step statement.

Additions to the Language

1. ATTRIB STATEMENT: The ATTRIB statement allows you to specify a variable's format, informat, label, and length in one DATA step statement.

2. SELECT GROUPS: The SELECT statement allows one of several blocks of code to be executed depending on the value of a control variable. SELECT groups can be nested. It is a DATA step statement of the general form:

   SELECT (var);
   WHEN (var) SAS statements; END;
   OTHERWISE SAS statements; END;
   END;

   For example:

   SELECT (X);
   WHEN (1) DO; SAS statements; END;
   WHEN (2) DO; SAS statements; END;
   OTHERWISE DO; SAS statements; END;
   END;

3. FORMAT JUSTIFIERS: You can have a format specification followed by a justifier—a hyphen and a single letter—to control where the formatted output is to be placed within the field: R, right-justified, L, left-justified, or C, centered.

4. RUN CANCEL and RUN QUIT: Two options on the RUN statement allow you to immediately end compilation of the current DATA or PROC step: "RUN CANCEL;" and "RUN QUIT;". The step will not be executed.

5. NON-INTEGER VALUES IN OPTIONS: Many options and the ABORT statement require an integer argument in the current SAS system. In the Portable SAS system, these arguments are accepted but truncated to an integer.

6. ERFC: The new function ERFC(argument) returns 1-ERF(argument).
7. ENCRYPTION: The system will include data set encryption but probably not in the early 1983 release.

8. COMPLEX DO GROUPS: In the portable system, a DO group can contain a WHILE or UNTIL clause after an iterative specification. For example: "DO YEAR = 1900 TO 1980 BY 10 WHILE (GNP(YEAR) < 5.0E+11);".

9. BY before TO in DO GROUPS: The BY-specification can precede or follow the TO-specification in DO groups. For example: "DO I = 1 BY 2 TO 99;" is equivalent to "DO I = 1 TO 99 BY 2;".

10. SINGLE-VALUE DO GROUPS: "DO I = value;" is allowed. It executes once, and I equals the value during and after the execution of the group. "DO I = value BY expression;" without a TO-specification is allowed. It executes indefinitely incrementing I by the by-expression each time until stopped by a WHILE or UNTIL clause, a GOTO out of the loop, or other means.

11. LIBNAME; LIBSEARCH: The new LIBNAME and LIBSEARCH commands allow you to equate SAS data library names (called DDnames on IBM Machines) to minicomputer path names.

12. HOST COMMAND: The HOST command passes a command in the host operating system's command language to the operating system and then returns the operating system's return code.

13. OPEN and CLOSE STATEMENTS: You can explicitly open and close files referenced in FILE and INFILE statements with the OPEN and CLOSE statements.

14. ALTERNATE SPELLINGS: In the Portable SAS system more alternate spellings are acceptable. For example, the keywords COLOR, CENTER, and DL1 can be spelled COLOUR, CENTRE, and Dll. The keyword CHARACTER in aggregate variable lists like X-CHARACTER-Y (meaning all character variables from X through Y) can be spelled CHAR.

15. COMMAS in FORMAT LISTS: You can use commas between items in parenthesized format lists. This allows the SAS system to distinguish between ($ 5.), which it interprets as the single format $5., and ($,5.), which it interprets as two separate formats, $ and 5.

16. SVARYING, FORMAT with INTEGER: The format SVARYING can be followed by an integer length value or a length variable. For example: "PUT STRING SVARYING30, 20;". The current SAS system only allows a variable.

Deletions from the Language

1. READ PASSWORD. The Portable SAS system does not support data set read passwords; however, it accepts these passwords and then treats them as comments.

2. EXTERNAL CALLS. The Portable SAS system does not support calls to external subroutines. Unless there is significant market demand and a satisfactory solution to implementation problems, external calls will not be supported.

3. COMMENT KEYWORD. Asterisk comments and comments of the form /* */ are allowed, but the keyword COMMENT statement is not allowed.

4. ERROR STATEMENT. The ERROR statement has been deleted from the language. The same result can be accomplished by using the two statements "_ERROR_ = 1; PUT ...;" instead.

5. REDUNDANT OPERATORS. The Portable SAS system does not support the following comparison operators: Instead of: NG use LE, NL use GE, @> use <=, @< use >=, <= use <=, => use >=.

6. STATEMENT LABELS on CARDS or CARDS4 STATEMENTS. Not allowed.

Changes

Arrays

1. EXPLICIT INDEXING: To reference an element of an array in the system, the index value for the element is enclosed in brackets or braces after the array name. Braces or brackets are used instead of parentheses to eliminate ambiguity with function references. This enhances readability of program listings and improves performance at compile time. For example, X(5) is the fifth element of the array X; LIST(A+B) is the A+B'th element of the array LIST.

2. NO INDEX VARIABLE: Arrays are declared without an index variable; for example, "ARRAY A A1-A99;" instead of "ARRAY A(INDEX) A1-A99;". Explicit indexing eliminates the need for this feature.

3. N-DIMENSIONAL ARRAYS: Arrays in the Portable SAS system can now have one or two dimensions. Declare a five-by-twenty element two-dimensional array: "ARRAY X(5,20) X1-X100;". The forty-fifth element of that array would be referenced: X(3,5). N-dimensional arrays are planned for implementation in mid-year.
4. **DO OVER:** The DO OVER statement does not exist in the portable SAS system. The practice of using "DO OVER arrayname;" and referring to array elements within the group by the array name alone is replaced by coding a regular DO group and referring to array elements within the group by subscripted array names (like "X(Q)").

5. **DIM FUNCTIONS:** The DIM (or DIM1) function returns the upper bound of the first dimension of an array. The DIM2 function returns the upper bound of the second dimension. For example: to print the value of each element of a two-dimensional array, code "DO I = 1 TO DIM(ARR); DO J = 1 TO DIM2(ARR); X = ARR(I,J); PUT X; END; END;".

Others

1. **QUOTED CHARACTER STRINGS:** Character strings are enclosed in single or double quotes in TITLE, FOOTNOTE, LABEL, and NOTE statements; and in the options LABEL=, PROMPTCHARS=, PARM=, MISSING=, MODECHARS=, SORTLIB= and SYSPARM= as well as READPW= on the INFILE statement. Previously, quotes were allowed but not required.

2. **DROP= and KEEP= OPTIONS:** If a DROP= or KEEP= data set option is used along with other options on the same data set, it must be the last one specified. Only DROP= or KEEP= may be used on a data set, not both, and only one of each.

3. **X COMMAND:** The X command, which sends a command to the host operating system, requires two operands in the Portable SAS system. The format of the command is "X osname 'command';" where osname is the name of the operating system. If the osname is not the same as the operating system on which the program is being executed, the X command is treated as a comment.

4. **DECIMAL FORMAT SPECIFICATIONS:** In the Portable SAS system, a decimal point must precede decimal format specifications. For example, "INPUT X 1-5 .3;" instead of "INPUT X 1-5 3;".

5. **SAS/GRAPH OPTIONS:** SAS/GRAPH options like ".C=RED" are not preceded by a period in the Portable version. The end of file is indicated by a control character sequence or by "/" as it is on the IBM system. In general, you will find that mini-computers use control character sequences for many operations that are done on the IBM systems with program functions and other types of job control statements. Some hardware systems require different missing value representation which may affect some of your output processing. However, we hope to resolve these difficulties before shipping out the test versions.

In addition, we will not be supporting the G and H floating point hardware on the Digital VAX, although this will not hurt executing of the system. Also, on the Digital VAX the absolute range of numbers is only 2.9E-39 to 1.7E+38, considerably smaller than that available on the IBM equipment and many other mini-computers.

**Schedules**

Voltaire once said that "doubt is a difficult state of mind, but certainty is ridiculous"; so I am not going to talk about certainty when I talk about schedules. I will, however, give you our best estimates. We hope to begin testing the product both at the alpha and beta level during the first quarter of 1983. When the system is completed and we believe the DATA step has become stable and its performance reasonable, we will thoroughly test the system in-house. For testing in-house, we are using Portable systems staff, Institute consultants, and marketing representatives to run programs they feel will thoroughly test the system. Once this has been completed and everyone agrees, we will provide a small number of alpha test sites with a copy for about a month. They will provide detailed feedback on the ability and performance of the product. These alpha test sites have been
selected, and you should know who you are. We hope to do this by the first of February. We will provide tapes to anyone desiring a test tape toward the end of the first quarter of 1983. The components of this test will include most of the procedures in the 1982 BASICS and STATISTICS manuals as well as most of the SAS/GRAPH procedures.

We plan to begin distributing the product during the second quarter of 1983. Next we will be providing SAS/FSP, the macro facility, IML (Interactive Matrix Language), and MATRIX language, and each product as it is converted to the Portable SAS system.

Some Delayed Features
There are always a few clouds on the horizon, and to provide the product as soon as we have, we have had to delay some features. Some of these, as you have seen, are some of the products available from the Institute for IBM systems such as the macro processor. In addition, we will not have named input or the format modifiers. We do not plan to have abbreviated variable lists working until some time after the product is out.

Probably the most significant feature for many users is that we will not initially support user-written procedures or external CALL statements. At the present time, it is difficult to both provide these and to protect our code. Also, it will be more difficult for users to develop procedures under the Portable SAS system. This, in turn, requires that a considerable educational support effort by the Institute before releasing this facility. I hasten to add that this feature will be maintained on the IBM system for the foreseeable future. The question is how it may be provided on the Portable SAS system. Finally, data set encryption will probably not be available until the end of the year.

Demonstration
There is a demonstration of the Portable SAS system on a Digital VAX 11/730 in La Galeries 3. This shows many of the DATA step features and many procedures. This demonstration will be going on throughout the conference beginning right after this talk. This evening at 6:00 p.m. there will be a "Birds of a Feather" in Mardi Gras D to answer any questions and concerns that you may have that I have not answered this morning.

Summary
In summary, the Portable SAS system is a very faithful implementation of the SAS system on 32-bit virtual system computers.

The Portable SAS system is written in PL/I subset G and will be initially distributed as one executable module. It requires a .7 mb memory image and will have an overall executable image size of less than 3 megabytes. The disk source will be somewhere around 15 megabytes. The initial release is planned to be available on the Data General MV series running AOS/VS, the Digital Equipment Corporation VAX series running VMS, and Prime 50 series running PRIMOS.

The major deletions from the SAS language are the DO OVER statement, the ERROR statement, and the COMMENT keyword. The major additions or changes to the language include explicit array subscripting, multiple dimensions for arrays, the addition of ATTRIBUTE, SELECT, OPEN and CLOSE statements, and format justifiers. A delayed feature is data set encryption.

For the future, we are currently looking at providing SAS under UNIX and operating systems from many other manufacturers both with and without the PL/I language and 32-bit processors.

Thank you for your patience and attention. If you have any questions, see me now or come to the demonstration or the "Birds of a Feather." I will be around during the entire conference to discuss the Portable SAS system with you.
Overview of the System

Source
Grammar → Compiler → PROCs
S-code
Interpreter
Supervisor
Output

Figure 1: Portable SAS System Components

Figure 2: Portable SAS System Flow