A GUIDE TO WRITING MICROCOMPUTER SIMULATIONS

David J. Fritzsche, University of Nevada-Reno
Richard V. Cotter, University of Nevada-Reno

ABSTRACT

Microcomputers offer desirable advantages over mainframe computers for writing simulations, including easier user-installation, better control of the system, an addressable screen, and increasingly widespread availability. Ways to utilize these inherent advantages are discussed. Software developers need to write completely self-contained systems of programs that can be utilized by the majority of existing microcomputer systems. MODERN BUSINESS DECISIONS (Cotter and Fritzsche, 1985) is used to illustrate the solutions to problems of program development and system utilization.

INTRODUCTION

Why Use Microcomputers

Microcomputers possess some inherent advantages which are not available on mainframe or minicomputers. Perhaps one of the major advantages of a micro is the ease of installation of programs. Those who have used mainframe simulations know the problems of reading stranger tapes, organizing and storing data files on a system different from that on which they were designed, and finally getting the simulation up and running. To install a simulation program written for a microcomputer, one has only to put the disk or disks containing the program(s) into the available disk drive(s), boot the system and perhaps copy the files to a system formatted disk. A well written program will take over from there and lead the user through a series of commands required to run the simulation program. There are some exceptions to this scenario when one is working with a general purpose program such as a spread sheet, a word processing package or data base program. These programs contain a series of commands which are used whenever needed during a program run. However, simulations are specific purpose programs and thus should contain instructions for operating the program at each point where user intervention is required.

The microcomputer allows more control over the operation of the simulation. The simulation administrator is the computer operator. The only limits on available system resources are those imposed by the limits of the computer hardware and software configuration. The microcomputer is available 24 hours per day (if one has a key to the building) and is dedicated to the user. Printer and disk use is controlled by the administrator.

The availability of a completely addressable screen provides a great deal of flexibility in program presentation which is not available on larger systems without the addition of expensive equipment. Most macros provide the capability to move the cursor to any point on the screen for data entry, provide a mechanism for highlighting areas of the screen and provide for the erasure of part of the screen without altering the remaining sections of the screen. Some computer systems also include color in the display if a color monitor is used. These capabilities allow an almost unlimited variety of screen displays to enhance communication between computers and people.

Finally, microcomputers are becoming standard equipment on the desks of many executives in the business community. By using the microcomputer for simulations, students may be provided positive experience with the same type of equipment they are likely to find in the workplace. If the simulation is tied to a data base which can be analyzed using spreadsheets, valuable additional experience is gained.

Problems Using Micros

Accompanying the advantages of a micro are a unique set of problems. While mainframe and mini computers are not totally compatible across brands, programs are easily modified to run on various systems. Micros are generally incompatible across brands and a great deal of effort is required to modify a program developed on one system to run on another system. (The exceptions include the clones or compatibles with Apple and IBM equipment.) A program stored on an Apple disk is totally unreadable to IBM, Radio Shack, Texas Instrument, etc. brand computers. The format of the disks on which the information is stored is different. The syntax of the BASIC language is different, and the format of the stored data is different.

As the simulation writer is much closer to the system when using a micro, the writer takes on added responsibility for the operation of the system. This often means that the writer must understand some of the technical details of system operation in order to make the simulation function properly. It is not uncommon if the simulation is at all complicated to include some machine language routines to perform functions which are not available in the BASIC language for a particular vendor’s on the machine or in the computer’s operating system. Fortunately a large number of machine routines are available from user groups and computer publications which, with some alteration, will perform nearly any function which the writer requires. A familiarity with the architecture of the micro and the format of the disk storage medium is quite helpful.

Program efficiency is relatively more important for microcomputers. Given that the available memory on a micro is much smaller than the memory on a typical mainframe, the simulation writer must be much more concerned with efficient memory usage. Because disk reading and writing takes a great deal more time when using floppy disks, one should attempt to minimize the volume of data transfer between disk and memory.

ISSUES TO CONSIDER

There are a number of specific issues which the simulation writer should address when designing a simulation for use on microcomputers. While probably not exhaustive, they include the following:

System of Programs

Simulations written for microcomputers should be completely self-contained. The user should be able to make desired parameter and history modifications easily using utility programs. Reruns should be allowed without requiring the user to learn system commands for file manipulation.

Thus a simulation written for a micro should include a series of programs which will permit any housekeeping chores the user desires. The major program, of course, is the simulation itself. A data entry program must be written so that the decisions may be entered onto disk (See Fritzsche, 1979). Preferably this is written in a self-instructional mode so that the student teams can enter their own decisions. In order to provide security for the decisions, the team decisions must be protected by passwords. It Is good practice to put passwords in a separate file which is read by the decision entry program so that the passwords can be changed easily by the simulation administrator. The data entry program must contain provisions for using passwords to determine whether a team is authorized to access specific decision data. This necessitates a means of protecting the data entry program from being listed so that access to the password file cannot be obtained by an inquiring competitor. The data entry program can be protected in one of several ways. The program can be released in a compiled version and thus will not be readable to humans. Some vendors provide a means of program storage that makes the stored program impossible to list. There are also a number of tricks available which will erase a file from memory if one tries to list it.

Passwords can be changed easily by providing the administrator with a program which will create and alter passwords. It is good practice to hide the password file so that it is not shown when a catalog of the disk containing the decision entry program and the decisions is listed. This can be done using tricks which are often specific to a particular brand of computer.

A number of utilities are generally required to make the simulation friendly for the administrator. As mentioned above, a program which creates and changes passwords is required. A program similar to the data entry program without the password protection should be available for the administrator to use to examine a team decisions and/or to make changes in any team’s decision deemed necessary. A program should be created which enables the administrator to change parameters easily. One or more programs are often required to initialize a series of historical decisions for the simulation. There may also be one or core machine language files which supplement the micro’s operating system and BASIC language. These files may perform such functions as chaining segments of the simulation program together and adding a print- formatting routine or other statements to the language.

Peripherals

The two peripherals of major concern are printers and storage devices. Printers range from simple, cheap $200 dot matrix devices to $2,000+ letter quality printers. Printers are connected to the micro by either a parallel or a serial board or port. Printers with a parallel port are generally much easier to connect than those with a serial port. If one assumes that a proper printer connection has been made prior to running a simulation, there should be little trouble accessing the printer. However, if the printer is not connected using the “standard slot or port for the computer, the simulation output may not print or will print in strange ways. The solution is to either reconfigure the computer using the standard~ output slot or port or include a configuration program with the simulation which will alter the program so that the nonstandard” slot or port can be used.

Most printers can print 80 column lines. However, there are a few printers which are limited to 40 columns. Others are capable of printing 132 character lines. One is generally safe in formatting the printout for 80 columns. While many 80 column printers can be altered to print 132 character lines, the configuration normally is software controlled with the specific control sequence differing from printer to printer. Thus to use the 132 character mode, one would have to provide a number of different versions, one for each printer which uses a different sequence of control characters. Page length must also be considered when the output requires more than one page.

While many micros are connected to two disk drives, there may still be a number of systems which contain only one disk drive. If the simulation is located on one disk with the simulation decisions located on another disk, it is convenient to configure the simulation [or a two drive system. However, provisions should be made to run the simulation on a one-drive system with disks being changed when necessary for reading decision input values from a second disk.

Memory

The standard memory configurations begin at 48K bytes and continue through 64K, 128K, 256K and 512K bytes. Some micros go even larger. While the cost of memory has declined significantly, there are a number of machines which are at the lower end of the continuum. The memory configuration is somewhat dependent upon the specific machine for which the simulation is designed. Eight bit processor computers are not able to address more than 64K of memory. It is possible to increase the memory above 64K, (say to 128K for the Apple) but the memory must be banked switched so that only 64K is used at one time. This generally limits the program to one bank of memory with the variable values possibly being located in the other bank of memory. If one writes programs to use this extra memory, the simulation can only be used with micros which have the extra memory. For the Apple, this would limit the simulation to Apple IIe's which contain the 80 column extended memory board. Thus for Apple II micros, one would be advised to design the simulation for 48K or 64K memory machines. It is probably safe to set up simulations for the IBM in the 128K configuration. The 16 bit processor used by the IBM and its compatibles can easily address 128K of memory and more.

Files and File Handling

There is little choice for program files other than BASIC (assuming that the simulation program is written in BASIC) or binary if the program is compiled. FORTRAN compilers are available for some micros. However, for data storage the writer has a choice of using sequential or random access text files. In all likelihood, both types of files will be used. Sequential text files are convenient to use for history files where the file will be read and/or written in its entirety. They normally take up less space and are easier to program.

Random access text files are generally better to use for parameter and decision files. Specific records in these files often need to be accessed without reading or writing the entire files. This is particularly true of decision files where all of the competing team's decisions may be included in a single file. The data entry program used by the teams to enter data should be able to access only the records pertaining to a specific team's decisions. The remaining records in the file should not be disturbed. It is often more efficient in terms of time and memory to read only part of the parameter file with the remainder, which does not pertain to the current period, being skipped. Random access files allow for these omissions. Random access parameter files also allow for changing values in specific records without disturbing other records.

Error Handling

Errors in program execution should normally be handled so that corrections can be made on the spot if possible. For errors which cannot be corrected during the current run, a message should be printed telling the simulation administrator what action should be taken to correct the error. The software systems of most micros provide error trapping routines which print standard error messages. These messages are sometimes confusing and often do not indicate the precise cause of the error. The simulation author should include programmed error traps which provide more precise messages for errors which can occur.

In some cases, errors can be corrected by the computer. If specific decision values are beyond a certain level, the values can be set to default levels. A better technique; however, is to trap these offending levels in the decision entry program where the team can alter the variable value to its desired level.

History Generation

Many simulations require startup history to be generated. This is generally accomplished by running the simulation over a specified set of periods prior to actual team competition. It is often possible to prescribe a standard startup condition with the prior pertinent data printed in the student manual. When using the standard startup, a short initialization run may be made.

When using a nonstandard startup, a series of periods of play must be run prior to beginning the simulation. These runs will generally include changes in the historical decisions which are made prior to running each historical period. The program which enables the user to generate historical data must be flexible enough to provide for both the standard and the nonstandard startup.

Manuals

A well written set of student and instructor manuals is probably as important as the simulation itself. The manual is often the simulation administrator's first contact with the simulation. The decision on whether to use or not to use the simulation is often made based upon an examination of the manuals.

The manuals should be written from the perspective of the user who is not particularly conversant with computers. The user, both the student and the instructor, should be led gently but not condescendingly by the hand through the simulation. The manuals should contain summary tables of important information located in an easy-to-find place in the manual. After the detailed explanation has been digested, the user will often refer to the summary tables as a reference.

The decision forms contained in the manual should be keyed to the screen displays in the decision entry program. Where appropriate, the units of measurement for the variables should be abbreviated on the decision forms. It is helpful to include worksheets which enable the teams to work through decision variable values

Modern Business Decisions

A discussion of a simulation recently written by the authors, Modern Business Decisions (1985), provides an illustration of the issues discussed above. The simulation comes on two disks. One disk, referred to as the decision disk, contains the data entry program and other files which enable the students to enter their decisions. The other disk, labeled program disk, contains the main simulation program and the files required to run the simulation as well as a series of utility programs for various housekeeping chores. Versions are available for the Apple II series and the IBM PC microcomputers. The system of programs and utilities included with the simulation are listed below:

Decision Disk

DECIDE -	Program for students to use to enter decisions in the decision file
PASSWORDS -	File containing student team passwords.
HELLO -	Program which executes DECIDE when disk is booted.
DEC-	File containing student team decisions.

Program Disk

CHAIN -	Utility to load program segments (Apple only).
EXAMINE -	Program used by instructor to examine team decisions.
HCREATE -	Utility (Apple only) to run simulation program for history creation.
HELLO -	Program which provides menu of alternatives when disk is booted. Program then executes selection.
INITHIST -	Program to create initial history.
MBD -	Simulation program.
PARAM -	File containing simulation parameters.
PARAMCHG -	Program used to change simulation parameters.
PUF -	Utility (Apple only) for enhancing Applesoft BASIC.
PWCREATE -	Program used to create and change team passwords.

In addition to the above files, a series of startup decision and startup history files are included.

The simulation is configured so that it will run with one or two disk drives. It will run with any printer which is compatible and successfully connected to the micro and is capable of printing 80 column lines. The printer card should be in slot one (Apple) or attached to the standard printer port (IBM).

Memory requirements for the simulation are 64K for the Apple II series and 128K for the IBM version. The program for the Apple is broken into three segments which are executed sequentially. The IBM executes the program in one segment.

Both sequential and random access files are used. The program requires two history files, both of which are sequential text files. The parameter (PARAM) file and the team decision file are random access text files. Selected records are read from the PARAM file during each simulation run. The parameter file may be altered using the PARAMCHG program which reads and writes individual records within the file. Program DECIDE reads and writes the record allocated to a team for its decisions when the appropriate password has been entered. A unique password is assigned to each record representing the team's decision.

Error handling is dealt with on three levels. First, certain entries are verified by asking the user if the entered values are correct. Second, entries in program DECIDE are checked against preset limits. If a variable value exceeds the limits, a message is issued and the team is asked to correct the value. Third, for errors which cannot be corrected in the program, an error message is issued describing the problem and the program terminates.

Prior to the beginning of play, simulation participants are presented with two years of historical data consisting of information about what happened during the eight quarterly periods of “Years 1 and 2. Simulation play then starts with Year 3, Quarter 1. The historical data may be initiated in one of two ways. The normal method is to use the preprogrammed information presented in the player's manual. This requires a short initialization run to set up history and decision files for Year 2, Quarter 4 for the appropriate numbers of student teams.
Then the simulation is ready to run.

If different historical data are desired, an alternate 8-quarter history may be generated with a longer initialization run (initiated by a choice on the menu). In this
case, the simulation administrator may choose to modify any or all of these types of preprogrammed startup data. Changes can be made to economic data (GNP, interest rates, etc.), certain program parameters and/or historical decision variables. Thus a unique and significantly different starting position is possible whenever it is desired.

Finally, both the player's and the instructor's manuals are written from the perspective that the users have had no previous computer experience. The explanation is detailed and complete. Humor has been incorporated to stimulate the reader. Summary tables and summaries at the end of the chapters provide quick reference to the details of the simulation.

Comment

Microcomputers provide some unique opportunities for simulations. Their big drawbacks, memory capacity and speed, can be dealt with through more efficient programming. These problems are also being alleviated by the trend toward larger memory and faster disk access in the new machines. A much greater problem is the lack of compatibility among machines. When writing a simulation, one must design the simulation to fit each individual type of machine on which it will be run. This is a very time consuming and frustrating job.

Simulations for the micro should be designed with consideration for the first- time user. The micro provides a unique medium for creating simulations which can be run entirely by the novice. There is no mainframe installation, no local modifications to make and no technically required setup to perform if the simulation is well designed. We now truly have the capability to build simulations which can be run by individuals who know nothing more about a computer than how to put a disk into a drive and turn on the switch. The availability of such simulations opens the door for any instructor to use simulation as a pedagogical technique. One no longer needs to be computer literate to use simulations. Simulation will be as easy to use as cases. Thus simulation is now ready for the masses. We have come of age!

REFERENCES

[1] Cotter, Richard V. and David J. Fritzsche, Modern Business Decisions, (Engle- wood Cliffs: Prentice-Hall Inc., 1985).

[2] Fritzsche, David J., “A Data Entry and Retrieval System for a Computer Simulation (DERS),” Insights Into Experiential Pedagogy, Proceedings of the Sixth Annual Conference of the Association for Business Simulation and Experiential Learning, April 18-20, 1979.