Graham

Design Notebook

GENERAL ELECTRIC

Appendix N

May 21, 1965

DISTRIBUTION

E. Glaser, MIT (6)

J. Ossanna, Bell Telephone Lab. (3)

W. Estfan, GE (6)

W. Heffner, GE

J. Merner, GE

F. Raunikar, GE (2) for Dr. Laird at Penn State

H. Sassenfeld, GE (8)

J. Weil, GE

GENERAL ELECTRIC

April 30, 1965

To Distribution:

Attached are notes on 636 Simulation prepared by G. G. Ziegler and J. Myers. The notes describe the 636 Simulation tools that are presently being implemented.

H. M. Sassenfeld

GENERAL ELECTRIC

April 30, 1965

NOTES ON 636 SIMULATION

GENERAL

The simulation of 636 processes by a GE635 may be accomplished through either the 636A or the 636B simulator, depending on the objectives and availability of equipment.

1) The simulator 636A provides complete 636 hardware simulation, including detailed I/O. The 636A simulator will serve to check out the TSS supervisor and other modules directly dealing with I/O. The 636A itself operates in the 635 Mastermode.

2) The simulator 636B will simulate all 636 commands except the CIOC Connect I/O Channel command. This simulator will be helpful in checking out the logic of 636 procedures, especially by utilizing its built-in debugging aids. As it operates under GECOS, the 636B may be used concurrently with other 635 processes if desired.

THE SIMULATOR 636A

The simulator 636A is a 635 procedure that will interpret all 636 instructions located in lower memory. Its interpretation logic consists of the following distinct modules operated in the 635 Mastermode:

a) Instruction Analysis

Within this module the actual instruction fetch and interrogation is performed. Bit 29 status is determined and the simulated base and address registers set accordingly. The absolute or non-absolute mode of the simulated environment is examined to determine the manner in which the operand and/or indirect words will be requested. In addition the instruction class is determined for proper execution.

b) Execution

This module is partitioned into eight subdivisions to handle the following classes of instructions that require distinct treatment with respect to simulation.

1. Simulation of transfer type instructions
2. Simulation of RET and RTD instructions
3. Simulation of the XEC instruction
4. Simulation of the XED instruction, also being invoked in the course of fault processing.
5. Simulation of the RPT and the affected repeated instructions
6. Simulation of RPD and the execution of the pair of repeated instructions
7. Simulation of RPL and the execution of the repeated instruction.
8. Simulation of all other "normal" instruction that do not alter the sequence.

c) Memory Access

This module performs the address translation, the necessary segment and paging checks such as limits, access privilege, read and write permit, etc. The check includes protection of the memory portion in which the simulator resides. There is no simulation of the associative memory mechanism.

d) Fault Interpretation

This module is activated whenever a fault condition is determined and stimulates the hardware faulting process.

In the 635s on which the simulation is performed, the execute, interrupt and fault cells are always located in the first "page" of memory 0-63. For the 636 the locations of the fault and execute interrupt cells can be chosen by toggle setting on the maintenance panel. To avoid saving and restoring of the first 64 words by the simulator, it is assumed that all 636 fault and execute interrupt cells are above location 63. Their individual location can be specified by parameter cards in the assembly process of the simulator. The above arrangement makes locations 0-63 unavailable to 636 programs during simulation. This does not really constitute restriction because the location, 0-63, can later be assigned as execute interrupt cells for the second processor.

e) Input-Output Simulation

Whenever the simulator encounters a connect instruction, the instruction is executed by the 635 without change. Any of the possible returns to the fault and execute interrupt cells are recognized by the simulator which then proceeds to interpret the instruction at the location that corresponds to the experienced fault or interrupt condition. The execution of the interpreted instruction which was interrupted will be either completed or aborted, according to the hardware specifications. It must be realized that time relationship of I/O activities can not be simulated, since the I/O activities are executed by the machine at full speed, whereas the simulation execution of instructions is some 50 to 200 times slower.

RESTRICTIONS

The following instructions are not simulated and if encountered, treated as NOPs:

GTB	Gray to Binary
LBAR	Load Base Address Register
SBAR	Store Base Address Register
RMFP	Read Memory File Protect Register
SMFP	Set Memory Protect File Register

The instructions LDT and STT are simulated referring to a "clock" which is increased by a specified amount each time an access to memory takes place. This will give a fair representation of the clock with the exception of multiply and divide instructions which will be relative infrequent in the TSS system itself.

As explained in the previous paragraph, the locations 0-63 will not be available for use of 636 programs. The execute interrupt and fault cells must be specified above location 63. Also the uppermost 4K (present estimate) of memory will not be available as this is the region in which the simulator resides during execution.

OPERATION

The simulator 636A must be loaded by GELOAD. After being loaded and receiving control, the simulator enters the Mastermode by means of a special MME. It then clears the lower memory and executes a load sequence calling 1024 words* from disk and starts simulating beginning at a specified location.

The simulation can be terminated by using the ESCAPE code, a special operation code. If the simulator encounters ESCAPE, it will abandon simulation and turn control to the 635 for future execution. This could be used to re-enter GECOS. Return to the simulator from 635 code can be achieved by using the special ZERO op code. The address of the ZERO op code determines the location from which simulation is to begin again. If the 635 is used directly after ESCAPE extreme care must be taken as, depending on 636 procedures executed, memory areas of the current procedure may not be contiguous and addresses are different as they can not be stated relative to segment origins in the 635 mode. It is understood that this is extremely provisory in order to keep the machine in operating condition until a skeleton of the 636 Time Sharing System is working.

THE SIMULATOR 636B

This simulator works in the slave mode in GECOS environment. The simulation of 636 instructions is identical to the 636A simulator with the following exceptions:

The CIOC Connect I/O Channel command is treated as a NOP.

On encountering the ESCAPE code, the simulated program enters directly the normal 635 GECOS controlled environment and I/O instructions can be initiated using the IOS supervisor. Return to the simulated mode can be achieved by executing op code ZERO.

The instructions RMCM Read Memory Controller Mask Registers, SMCM Set Memory Controller Mask Registers and SMIC Set Memory Controller Interrupt Cells are treated as NOPs.

* tentative

DEBUGGING AIDS

An initial debugging tool is provided by means of an op code(SNAP) recognized by the simulator 636B. Execution of the op code will establish for the simulator, a reference to a Snap-Point Control-Table (SPCT). The SPCT, when established, is interrogated each time the simulator accesses the 636 P-counter. When an instruction fetch address matches an address that is described in the SPCT, a memory print is accomplished. The memory print is accomplished before the instruction is executed.

The print will display the 636 registers and, optionally, the 636 segments comprising the process. The memory print format will provide:

1) The location of the SNAP that initiated the print
2) The contents of the 636 registers and an optional group
3) An identification of each segment being printed
4) The location of the displayed words as a count from the segment origin
5) Each word displayed as 12 octal characters
6) Each word with bit 29 on flagged with an asterisk

The SNAP capability is controlled by the execution of the SNAP op code. The format of the instruction is:

```
SNAP   arg 1, arg 2
```

where arg 1 is the address of a SPCT and arg 2 a control flag (index) that specifies:

Arg 2 = 0 establish and utilize arg 1's SNAP Point Control Table

Arg 2 = 1 SNAP all of the process and terminate simulation of the process

Arg 2 = 2 SNAP the registers and terminate simulation of the process

The SNAP Activity is directed by the SNAP Point Control Table (SPCT). The format of the table is as follows:

```
LABEL: ZERO  arg 1, arg 2
       ZERO  arg 1, arg 2
       ZERO  arg 1, arg 2
       ZERO  777777
```

Where label is the symbolic label referred to by a SNAP, arg 1 is the address of the instruction that if accessed by an instruction fetch is to cause a print; arg 2 is the description of the extent of the print requested at this point

0	registers print only
1	registers and segments print

The argument ones of the ZERO list must have values of ascending magnitude (to facilitate the search). The last ZERO must specify arg 1 as 777777 to define the end of the table. There are a number of constraints on the use of SNAP:

1) the SNAP and its SPCT must be a part of the same segment
2) the segment containing the SNAP must be non-paged

In the usual case SNAP will be the first instruction of the segment and its SPCT the last words of the segment. The SPCT will usually contain only 1 to 4 entries.

OPERATION AND USAGE

The simulator 636B contains a loader that will load 636 code and set-up a descriptor, linkage, linkage boundary, procedure segment and stack.