Draft for approval
Published: 03/09/66

Identification

Simulator
G.G.Ziegler, G.E.

Purpose

The 645 Simulator simulates the 645 hardware operation on the process placed in memory by the 645 loader.

Error File Messages

The Simulator records its messages on the error file.

Message

.sim. normal term
The simulator encountered the ESCAPE op code with an effective address equal to zero. The C(PBR) and C(IC) indicate the location of this instruction.

.sim. not attempted
An error was detected during the loading phase prior to simulation. Simulation was not attempted.

.sim. invalid tag
An illegal address modifier was encountered during address modification. The C(PBR) and C(IC) indicate the location of the instruction on which the violation was detected.

.sim. fault i
A fault was detected for which a user supplied fault routine had not been provided. "i" is an octal value which corresponds to the fault code associated with the fault.

i	fault
1	Master Mode Entry i
2	Derail
3	Timer Runout
4	Master Mode Entry 2
5	Master Mode Entry 3
7	Master Mode Entry 4
10	Fault Tag 1
11	635 Compatibility
12	Illegal Procedure
13	Illegal Descriptor
16	Fault Tag 2

17	Fault Tag 3
20	Directed Fault 0
21	Directed Fault 1
22	Directed Fault 2
23	Directed Fault 3
24	Directed Fault 4
25	Directed Fault 5
26	Directed Fault 6
27	Directed Fault 7
30	635/645 Compatibility
31	Overflow
32	Divide Check
35	Op Not Complete
37	Trouble

.sim. time exceeded
The limit of simulation has been exceeded.

.sim. rcu error
The data supplied by an RCU instruction in an attempt to restore the control unit is invalid.

.sim. illegal repeat
The simulator detected an invalid tag modifier on an instruction to be repeated.

Usage

The simulator package consists of the following routines.

A) 645SIM - Simulator
B) 645SCR - Communication Region
C) 645DMP - Termination Interface
D) DVECTR - Initial Fault Vector
E) MCTBLE - Memory Controller Definition
F) ESCAPE - Escape Vector
G) ....TR - Trace

Each routine exists in relocatable format and is loaded (along with the 645 loader) by the General Loader (GELOAD). An interface (the escape option) is provided to permit user supplied 625/635 subprograms to reside in memory with the simulator. Escapes can then be made to the 625/635 subprograms.

A. Simulator (645SIM)

1. General

The simulator portion performs the instruction

interpretation, address modification, and fault detection. To provide an environment which best approximates actual hardware interaction, simulation is interpretive. With the exception of the associative memory, attempt has been made to simulate as much of the functions of the 645 hardware as possible.

2. Description

a. Entry

The symdef START is the entry point to 645SIM. Control should transfer to this location only after certain setup procedures within the communication region have been accomplished.

b. ESCAPE Op Code

The op code ESCAPE (octal: 001) when encountered under simulation has two special connotations.

1) An ESCAPE op code whose computed effective address equals zero is used to effect a termination of the simulation process.

2) When an ESCAPE op code is encountered whose computed effective address is not equal to zero, the simulator assumes the escape option. The option is made available as a means to interrupt the simulation process and execute 625/635 object code directly.

Under this option, the computed effective address is interpreted as an offset pointer into a vector whose origin is specified by the address field of the symdef ESCAPE. (An effective address of 1 corresponds to L (vector), 2 to L(vector +1), etc.). Recognition of an ESCAPE whose effective address is non-zero causes the contents of the vector location corresponding to the effective address to be examined. If the vector location contains zero, the ESCAPE op will be ignored and simulation will proceed with the next sequential 645 instruction. If the vector location is not zero, the simulator will execute a transfer indirect through the vector location. Index register 1 will contain the

the location to which the escape coding must return after performing its functions.

The effective address of any ESCAPE op may not exceed 2**17-1.

The simulator assumes that the vector specified by the symdef is of sufficient length to accommodate the number of unique effective addresses that may appear in an ESCAPE instruction word.

c. TRACE op-code

The op-code TRACE (octal: 002) is provided as an egression for the tracking of instruction executions. A TRACE op with an effective address of zero, initiates an instruction by instruction panel display to the standard GECOS output collector-SYSOUT. The tracing procedure is terminated whenever a TRACE op is encountered whose effective address is non-zero.

The information provided describes the condition of the processor just prior to the initial preparation of the instruction address for the instruction defined by the C(PBR) and C(IC). In addition to the normal output, the contents of the control unit is displayed whenever the instruction being executed has been obtained from the fault vector.

The normal tracing mode is off.

d. Fault Vector

The following fault vector is assumed by the simulator:

```
C(FVCTR) + (0)8     *Shutdown
         + (2)8     Master Mode Entry 1
         + (4)8     Derail
         + (6)8     Timer Runout
         + (10)8    Master Mode Entry 2
         + (12)8    Master Mode Entry 3
         + (14)8    *Connect
         + (16)8    Master Mode Entry 4
         + (20)8    Fault Tag 1
         + (22)8    635 Compatibility
         + (24)8    Illegal Procedure
         + (26)8    Illegal Descriptor
         + (30)8    *Parity
```

+ (32)8	*Illegal Memory Command
+ (34)8	Fault Tag 2
+ (36)8	Fault Tag 3
+ (40)8	Directed Fault 0
+ (42)8	Directed Fault 1
+ (44)8	Directed Fault 2
+ (46)8	Directed Fault 3
+ (50)8	Directed Fault 4
+ (52)8	Directed Fault 5
+ (54)8	DIRected Fault 6
+ (56)8	Directed Fault 7
+ (60)8	635/645 Compatibility
+ (62)8	Overflow
+ (64)8	Divide Check
+ (66)8	*Execute
+ (70)8	*Lockup
+ (72)8	Op Not Complete
+ (74)8	*Startup
+ (76)8	Trouble

*These entries pertain to faults which are not detected by the simulation process.

Notes:

1) The zero op code fault is not unique as in the 625/635. The zero op code results in an Illegal Procedure fault.

2) The simulator assumes only one processor; this processor is the control processor.

8. Communication Region (645SCR)

1. General

The communication region contains the simulated hardware registers plus other data locations pertinent to the simulation process.

2. Description

a. Initialization
The following symdef's are germane to the 645 loader in the initialization of the simulator.

Symbol: ABSM
Definition: Addressing mode of the processor
Usage: If the content of this location is non-zero, the addressing mode is assumed absolute. When in the

absolute addressing mode all instructions are accessed using the instruction counter (IC) directly. Indirect words and operand locations are referenced using the computed 18-bit effective address directly, except when bit 29 is on in an instruction word or ITS/ITB words are encountered on indirection. For these cases, address translation is invoked.

If the content of this location is zero, the addressing mode is assumed appending. With appending in effect, address translation is performed on instruction word accesses using the C(PBR) and C(IC). When computing indirect and operand word locations, the C(PBR), content of a specified base register, or the content of the address field of an ITS word is used in conjunction with the 18-bit effective address to compute an actual address.

Symbol: PBR
Definition: The Procedure Base Register
Usage: The C(PBR)0-17 is interpreted as the Procedure Base Register and specifies a location relative to some zero within a segment identified by the Descriptor Base Register. If simulation is initiated with appending invoked, this parameter must be supplied.

Symbol: IC
Definition: The Instruction Counter.
Usage: If the absolute addressing mode has been invoked, the C(IC)0-17 represents an absolute 645 memory address. If the addressing mode is not absolute, the C(IC)0-17 specifies some location relative to the beginning of a segment identified by the content of the Procedure Base Register (PBR). On initialization of the simulator,

the C(IC)0-17 locates, relative to some zero origin, the first instruction to be executed.

Symbol: ZER636
Definition: Origin of the 645 memory
Usage: The C(ZER636) 0-17 is interpreted as the 625/635 origin of the simulated 645 memory region. This value must be zero for the current version of the simulator as it is necessary that 625/635 zero and 645 zero coincide.

Symbol: TOM
Definition: Word size of the simulated 645 memory.
Usage: Each simulated memory reference causes the computed 645 memory address to be compared to the C(TOM) 0-17. If the memory address is equal to or greater than the C(TOM) 0-17, an op not complete fault is simulated.

Symbol: FVCTR
Definition: FAULT Vector Pointer
Usage: The C(FVCTR) 0-17 specifies the origin of the 645 fault vector interpreted as a 645 memory address. This value must be modulo 64. The C(FVCTR) 18-35 must be zero.

Symbol: CYCLS
Definition: The maximum number of 645 memory references that the simulator is to allow beyond which simulation is to be terminated.
Usage: Within the system, the 645 loader initializes the C(CYCLS) with a value which represents 1000's of memory requests. Entry to the simulator results in this value being converted to a 36-bit integer specifiying 2**35 minus the maximum number of references.

Each simulated 645 memory reference causes the C(CYCLS) to

be incremented by one. When the C(CYCLS) reaches a value of 2**35, simulation ceases.

Notes:

1) For normal instruction sequencing, two instructions are obtained per 645 memory request. Indirect and normally operand words are single word requests. Execute double, transfers and return may result in additional memory references depending upon the even or odd word memory location of the instruction and/or its conformant.

2) During address translation memory references required for retrieving descriptor, and page table words do not affect the C(CYCLS).

Symbol: DBR
Definition: The Descriptor Base Register
Usage: The C(DBR) 0-28 is interpreted as the Descriptor Base Register. This information specifies the absolute 645 location of the origin of the descriptor segment or the descriptor segment's page table. This parameter must be supplied if simulation is initiated with C(ABSM) =0, or if any address translation is required (e.g., bit 29) before this register can be initialized through simulation.

b. Other

The following are symdef's relevant to simulation but for which initialization is not a prerequisite.

Whenever any of the locations are not preset by an initialization procedure, zero values are assumed.

Symbol: A
Definition: The left half of the 72-bit AQ arithmetic register.

Symbol: Q
Definition: The right half of the 72-bit AQ arithmetic register.

Symbol: AQ
Definition: The 72-bit AQ arithmetic register.

Symbol: ER
Definition: The Exponent Register
Usage: The C(ER) 0-7 is interpreted as the Exponent Register.

Symbol: Xn n = 0(1)7
Definition: Index Register n.

Symbol: BRn n = 0(1)7
Definition: Base Address Register n
Usage: The C(BRn) 0-23 is interpreted as the corresponding base address register.

Symbol: IR
Definition: The Indicator Register
Usage: Content of bit positions 18-29 is interpreted as the indicator Register.

The actual value of the absolute addressing indicator is reflected by the C(ABSM). The bit position corresponding to the absolute addressing indicator may not always reflect the C(ABSM) due to requirements of the simulator. However, this will not be reflected in any 645 simulation. Instructions which store the indicators have the indicator set properly.

Symbol: TR
Defintion: The Time Register
Usage: The C(TR) represent a simulated timer register. This location is utilized as follows. Whenever the LDT instruction is encountered, the 24-bit argument is subtracted from 2**23 with the result replacing the C(TR) 0-23 and zeros replacing C(TR) 24-35.

When the instruction STT is encountered, the C(TR) is subtracted from 2**35, and the most significant 24 bits considered as the timer value.

Each simulated 645 memory reference causes the C(TR) to be incremented by one. When the C(TR) has exceeded a value of 2**35-1, and if the current instruction is not executing under Procedure Master, a timer runout fault will occur.

Notes:

1) For normal instruction sequencing, two instructions are obtained per 645 memory request. Indirect and normally operand words are single word requests. Execute double, transfers, and return may result in additional memory references depending upon the even or odd memory location of the instruction and/or its conformants.

2) During address translation memory references required for retrieving descriptor segment words and page table words do not cause C(TR) to be incremented.

Symbol: MSM
Definition: Current Procedure Mode.
Usage: If the content of this location is zero, the current instruction is executing as Procedure Slave or Procedure Execute Only. If the content of this location is non-zero, the current instruction is executing as Procedure Master.

Note: When executing in the absolute addressing mode, procedure is assumed master and therefore, the C(MSM) will be non-zero.

Symbol: CU
Definition: Control Unit Status
Usage: Whenever a fault condition is detected, the control unit status is interrogated and set into C(CU) through C(CU+5) in anticipation of an SCU instruction. When an SCU instruction is encountered, this information is moved to the area specified by the effective address of the SCU.

Symbol: IEVEN, IODD
Definition: Instruction Register
Usage: Instruction fetches are normally double precision reads. The C(IEVEN) and C(IODD) contain the active instruction pair.

Symbol: IEIO
Definition: This symbol is equivalent to IEVEN.

Symbol: MCIMR
Definition: Memory Controller Interrupt Mask Registers.
Usage: MCIMR is a vector of eight words, the contents of 0-31 of each representing a memory controller's interrupt mask register. MCIMR corresponds to controller 0, MCIMR + 1 to controller 1, etc.

Symbol: MCAMR
Definition: Memeory Controller Access Mask Register.
Usage: MCAMR is a vector of eight words, the contents of 0-7 of each representing a memory controller's access mask register. MCAMR corresponds to controller 0, MCAMR + 1 to controller 1, etc.

Symbol: MCIC
Definition: Memory Controller Interrupt Calls
Usage: MCIC is a vector of eight words, the contents of 0-31 of each representing a memory controller's interrupt cells. MCIC corresponds to controller 0, MCIC + 1 to controller 1, etc.

C. Termination Interface (645DMP)

1. General
This routine is the terminal interface for simulation within the system.

Entry is either from the simulator or the 645 loader. Entry is made from the loader only when simulation is to be avoided and results in an error message being produced on the error file, and a core image deposited on the core dump file, after which the activity is terminated. Entry from the simulator is accomplished when the simulator detects a disastrous error condition or termination of simulation is encountered. The particular entry condition results in a relevant message being produced on the error file and a core dump onto the dump file, after which the simulator activity is terminated.

An option is provided whereby an octal core dump of the 645 memory area may be produced on the standard GECOS output collector - SYSOUT. Prior to terminating the activity, the GECOS switch word for this activity is examined. If bit position 6 has been set to one, the octal dump is produced. The switch word may be set either by using the escape option and executing a MME GESETS or by placing an ON1 option on the $EXECUTE control card for the simulation activity.

2. Description

a. Entry
The following symdef's represent the entry points to 645DMP.

Symbol:	FINISH
Definition:	Terminate location for the simulator.
Usage:	Whenever the simulator detects an error condition, or when 645 object code requests termination, control is transferred to this location using the standard 625/635 call. This transfer results in an appropriate message written on the error file "ER" and the 645 registers, core memory written on the dump file "CR". After this has been accomplished, simulation terminates via a MME GEFINI.

The calling sequence assumed is

```
CALL FINISH(ARG1,ARG2)
```

where

ARG1 -specifies the location of a BCD message which is to be written to the error file, and

ARG2 -specifies the number of 6-bit BCD characters in the message.

Symbol: SUICID
Definition: Terminate location for the 645 loader.
Usage: Whenever the 645 loader determines that simulation is not possible, or undesirable, control is transferred to this location. The transfer results in a terminal message being written to the error file, "ER", and the 645 core image, together with the contents of the 645's registers, written to the dump file, "CR". After this has been accomplished, simulation terminates via a MME GEFINI.

b. Format of the Dump File
The dump file produced at termination of the simulation process contains the following information in the order listed below.

- Number of words preceding the core image
- A - Register
- Q - Register
- Exponent Register
- Indicator Register
- Timer Register
- Instruction Counter
- Index Register 0
- Index Register 1
- Index Register 2
- Index Register 3
- Index Register 4
- Index Register 5
- Index Register 6
- Index Register 7

- Descriptor Base Register
- Procedure Base Register
- Base Address Register 0
- Base Address Register 1
- Base Address Register 2
- Base Address Register 3
- Base Address Register 4
- Base Address Register 5
- Base Address Register 6
- Base Address Register 7
- Number of words assigned to the simulated 645 memory
- 625/635 address of the zeroth (bits 0-17) and top (bits 19-35) most addressable locations of the simulated 645 memory
- Control Unit Status Word 0
- Control Unit Status Word 1
- Control Unit Status Word 2
- Control Unit Status Word 3
- Control Unit Status Word 4
- Control Unit Status Word 5
- Memory Controller 0 Interrupt Mask Register
- Memory Controller 1 Interrupt Mask Register
- Memory Controller 2 Interrupt Mask Register
- Memory Controller 3 Interrupt Mask Register
- Memory Controller 4 Interrupt Mask Register
- Memory Controller 5 Interrupt Mask Register
- Memory Controller 6 Interrupt Mask Register
- Memory Controller 7 Interrupt Mask Register
- Memory Controller 0 Access Mask Register
- Memory Controller 1 Access Mask Register
- Memory Controller 2 Access Mask Register
- Memory Controller 3 Access Mask Register
- Memory Controller 4 Access Mask Register
- Memory Controller 5 Access Mask Register
- Memory Controller 6 Access Mask Register
- Memory Controller 7 Access Mask Register
- Memory Controller 0 Interrupt Cells
- Memory Controller 1 Interrupt Cells
- Memory Controller 2 Interrupt Cells
- Memory Controller 3 Interrupt Cells
- Memory Controller 4 Interrupt Cells
- Memory Controller 5 Interrupt Cells
- Memory Controller 6 Interrupt Cells
- Memory Controller 7 Interrupt Cells
- 645 Core Image*

* The core image contains only that contiguous portion of memory which contains non-zero information. Memory locations from the last addressable on down which contain zeros are not written to the core dump file.

D. Initial Fault Vector (DVECTR).

1. General
This routine contains the assumed contents of the fault vector when a fault vector is not supplied by the user.

2. Description
During its initialization tasks, the 645 loader moves this routine to the simulated fault vector origin before any user requested insertions are made. Each location of this initial vector contains the following instruction.

```
ESCAPE  -1
```

This instruction when encountered under simulation results in immediate termination of the simulation process.

E. Memory Controller Definition (MCTBLE)

1. General
This is a table which defines the interlacing of memory addresses across memory controllers.

2. Description
The origin of this 32 word table is defined by the symdef, MCTBLE. The table is consulted when simulating a SMIC, RMCM, or SMCM command to determine which memory controller the instruction is addressing. Eight memory controllers may be assumed. The address of a memory controller is determined by concatenating bits 0-2 and 15-16 of the effective address of the command and using the result as an off-set pointer into this table. The content of 0-17 of the table location is interpreted as the controller number to which the command is addressed.

The table initially supplied in the simulator package assumes that all addresses are assigned to memory controller 0.

F. Escape Vector (ESCAPE)

1. General
This is a vector which defines the 625/635 escape code options.

2. Description
This routine contains a dummy escape coding vector and the symdef ESCAPE which locates the vector. It is a requirement of the simulator that this routine be present when no user supplied escape coding exists. When escape coding is present, this routine must be replaced with a vector, and symdef locating the vector, associated with the supplied escape coding.

G. Trace (...TR)

1. General
This routine produces the instruction by instruction panel display when the tracing option is invoked.

Entry is from the simulator immediately preceding initial preparation of the address of an instruction word.

2. Description
a. Entry

Symbol:	....TR
Definition:	Entry location for panel display
Usage:	Each entry results in an octal display of the 645 register as they appear at the time of the entry. The data is written to the standard GECOS output collector-SYSOUT, using the GEFRC routine.GPRNT.
	A maximum of 1000 displays is allowed during any given simulation run.
	In addition to the normal output, the control unit status is also displayed whenever the instruction being executed has been obtained from the fault vector.

b. Format of the display

Each panel display contains the following information.

- A - Register
- Q - Register
- Exponent Register
- Indicator Register
- Timer Register*

- Instruction Counter
- Index Register 0
- Index Register 1
- Index Register 2
- INdex Register 3
- Index Register 4
- Index Register 5
- Index Register 6
- Index Register 7
- Descriptor Base Register
- Procedure Base Register
- Base Address Register 0
- Base Address Register 1
- Base Address Register 2
- Base Address Register 3
- Base Address Register 4
- Base Address Register 5
- Base Address Register 6
- Base Address Register 7
- Control Unit Status Word 0**
- Control Unit Status Word 1**
- Control Unit Status Word 2**
- Control Unit Status Word 3**
- Control Unit Status Word 4**
- Control Unit Status Word 5**
- Even active instruction
- Odd active instruction
- CYCLS*

---

* expressed in simulator format
** only when executing from the fault vector