GRI-909/GRI-99 Simulator Usage
01-Dec-2008

COPYRIGHT NOTICE

The following copyright notice applies to the SIMH source, binary, and documentation:

This memorandum documents the GRI-909 simulator.

1 Simulator Files

sim/	scp.h
	sim_console.h
	sim_defs.h
	sim_fio.h
	sim_rev.h
	sim_sock.h
	sim_timer.h
	sim_tmxr.h
	scp.c
	sim_console.c
	sim_fio.c
	sim_sock.c
	sim_timer.c
	sim_tmxr.c
sim/gri/1	gri_defs.h
	gri_cpu.c
	gri_stddev.c
	gri_sys.c

2 GRI-909/GRI-99 Features

The GRI-909 is configured as follows:

```
device name(s)    simulates

CPU               GRI-909/GRI-99 CPU with up to 32KW of memory
HSR               S42-004 high speed reader
HSP               S42-004 high speed punch
TTI               S42-001 Teletype input
TTO               S42-002 Teletype output
RTC               real-time clock
```

The GRI-909 simulator implements the following unique stop conditions:

- An unimplemented operator is referenced, and register STOP_OPR is set
- An invalid interrupt request is made

The `LOAD` commands has an optional argument to specify the load address:

```
LOAD <filename> {<starting address>}
```

The `LOAD` command loads a paper-tape bootstrap format file at the specified address. If no address is specified, loading starts at location 200. The `DUMP` command is not supported.

2.1 CPU

The only CPU options are the presence of the extended arithmetic operator and the size of main memory.

```
SET CPU GRI909              set CPU model to GRI-909
SET CPU GRI99               set CPU model to GRI-99
SET CPU AO                  enable extended arithmetic operator
SET CPU EAO                 disable arithmetic operator
SET CPU EAO                 enable extended arithmetic operator
SET CPU NOEAO               disable extended arithmetic operator
SET CPU GPR                 enable general registers
SET CPU NOGPR               disable general registers
SET CPU BSWPK               enable byte swap/pack
SET CPU NOBSWPK             disable byte swap/pack
SET CPU 4K                  set memory size = 4K
SET CPU 8K                  set memory size = 8K
SET CPU 12K                 set memory size = 12K
SET CPU 16K                 set memory size = 16K
SET CPU 20K                 set memory size = 20K
SET CPU 24K                 set memory size = 24K
SET CPU 28K                 set memory size = 28K
SET CPU 32K                 set memory size = 32K
```

If memory size is being reduced, and the memory being truncated contains non-zero data, the simulator asks for confirmation. Data in the truncated portion of memory is lost. Initial memory size is 32K. The default configuration is a GRI-909 with AO, EAO, and GPR.

CPU registers include the visible state of the processor as well as the control registers for the interrupt system.

```
name            size        comments

SC              15          sequence counter
AX              16          arithmetic operator input register 1
AY              16          arithmetic operator input register 2
AO              16          arithmetic operator output register
TRP             16          TRP register
MSR             16          machine status register
ISR             16          interrupt status register
BSW             16          byte swapper buffer
BPK             16          byte packer buffer
GR1..GR6        16          general registers 1 to 6
XR              16          index register (GRI-99 only)
BOV             1           bus overflow (MSR<15>)
L               1           link (MSR<14>)
FOA             2           arithmetic operator function (MSR<9:8>)
AOV             1           arithmetic overflow (MSR<0>)
IR              16          instruction register (read only)
MA              16          memory address register (read only)
SWR             16          switch register
DR              16          display register
THW             6           selects operator displayed in DR
IREQ            16          interrupt requests
ION             1           interrupts enabled
INODEF          1           interrupts not deferred
BKP             1           breakpoint request
SCQ[0:63]       15          SC prior to last jump or interrupt;
                            most recent SC change first
STOP_OPR        1           stop on undefined operator
```

```
WRU             8           interrupt character
```

2.2 Programmed I/O Devices

2.2.1 S42-004 High Speed Reader (HSR)

The paper tape reader (HSR) reads data from or a disk file. The POS register specifies the number of the next data item to be read. Thus, by changing POS, the user can backspace or advance the reader.

The paper tape reader implements these registers:

```
name            size        comments

BUF             8           last data item processed
IRDY            1           device ready flag
IENB            1           device interrupt enable flag
POS             32          position in the input file
TIME            24          time from I/O initiation to interrupt
STOP_IOE        1           stop on I/O error
```

Error handling is as follows:

```
error         STOP_IOE      processed as

not attached      1         report error and stop
                  0         out of tape

end of file       1         report error and stop
                  0         out of tape

OS I/O error      x         report error and stop
```

2.2.2 S42-006 High Speed Punch (HSP)

The paper tape punch (HSP) writes data to a disk file. The POS register specifies the number of the next data item to be written. Thus, by changing POS, the user can backspace or advance the punch.

The paper tape punch implements these registers:

```
name            size        comments

BUF             8           last data item processed
ORDY            1           device ready flag
IENB            1           device interrupt enable flag
POS             32          position in the output file
TIME            24          time from I/O initiation to interrupt
STOP_IOE        1           stop on I/O error
```

Error handling is as follows:

```
error         STOP_IOE      processed as

not attached      1         report error and stop
                  0         out of tape
```

```
OS I/O error      x           report error and stop
```

2.2.3 S42-001 Teletype Input (TTI)

The Teletype interfaces (TTI, TTO) can be set to one of four modes, KSR, 7P, 7B, or 8B:

```
mode  input characters        output characters

KSR   lower case converted    lower case converted to upper case,
       to upper case,         high-order bit cleared,
      high-order bit set      non-printing characters suppressed
7P    high-order bit cleared  high-order bit cleared,
                              non-printing characters suppressed
7B    high-order bit cleared  high-order bit cleared
8B    no changes              no changes
```

The default mode is KSR.

The Teletype input (TTI) polls the console keyboard for input. It implements these registers:

```
name              size        comments

BUF               8           last data item processed
IRDY              1           device ready flag
IENB              1           device interrupt enable flag
POS               32          number of characters input
TIME              24          keyboard polling interval
```

2.2.4 S42-002 Teletype Output (TTO)

The Teletype output (TTO) writes to the simulator console window. It implements these registers:

```
name              size        comments

BUF               8           last data item processed
ORDY              1           device ready flag
IENB              1           device interrupt enable flag
POS               32          number of characters output
TIME              24          time from I/O initiation to interrupt
```

2.2.5 Real-Time Clock (RTC)

The real-time clock (CLK) implements these registers:

```
name              size        comments

RDY               1           device ready flag
IENB              1           interrupt enable flag
TIME              24          clock interval
```

The real-time clock autocalibrates; the clock interval is adjusted up or down so that the clock tracks actual elapsed time.

3 Symbolic Display and Input

The GRI-909 simulator implements symbolic display and input. Display is controlled by command line switches:

```
-a              display as ASCII character
-c              display as two packed ASCII characters
-m              display instruction mnemonics
```

Input parsing is controlled by the first character typed in or by command line switches:

```
' or -a         ASCII character
" or -c         two packed ASCII characters
alphabetic      instruction mnemonic
numeric         octal number
```

Instruction input uses modified GRI-909 basic assembler syntax. There are thirteen different instruction formats. Operators, functions, and tests may be octal or symbolic; jump conditions and bus operators are always symbolic. Addresses may be prefixed with #, indicating indexing (GRI-99 only).

```
Function out, general
Syntax:                 FO function,operator
Function symbols:       INP, IRDY, ORDY, STRT
Example:                FO ORDY,TTO

Function out, named
Syntax:                 FO{M|I|A} function
Function symbols:       M: CLL, CML, STL, HLT
                        I: ICF, ICO
                        A: ADD, AND, XOR, OR
Example:                FOA XOR

Sense function, general
Syntax:                 SF operator,{NOT} tests
Test symbols:           IRDY, ORDY
Example:                SF HSR,IRDY

Sense function, named
Syntax:                 SF{M|A} {NOT} tests
Test symbols:           M: POK, BOV, LNK
                        A: SOV, AOV
Example:                SFM NOT BOV

Register to register
Syntax:                 RR{C} src,{bus op,}dst
Bus op symbols:         P1, L1, R1
Example:                RRC AX,P1,AY

Zero to register
Syntax:                 ZR{C} {bus op,}dst
Bus op symbols:         P1, L1, R1
Example:                ZR P1,GR1

Register to self
Syntax:                 RS{C} dst{,bus op}
Bus op symbols:         P1, L1, R1
```

```
Example:                RS AX,L1

Jump unconditional or named condition
Syntax:                 J{U|O|N}{D} address
Example:                JUD 1400

Jump conditional
Syntax:                 JC{D} src,cond,address
Cond symbols:           NEVER,ALWAYS,ETZ,NEZ,LTZ,GEZ,LEZ,GTZ
Example:                JC AX,LEZ,200

Register to memory
syntax:                 RM{I|D|ID} src,{bus op,}address
Bus op symbols:         P1, L1, R1
Example:                RMD AX,P1,1315

Zero to memory
Syntax:                 ZM{I|D|ID} {bus op,}address
Bus op symbols:         P1, L1, R1
Example:                ZM P1,5502

Memory to register
Syntax:                 MR{I|D|ID} address,{bus op,}dst
Bus op symbols:         P1, L1, R1
Example:                MRI 1405,GR6

Memory to self:
Syntax:                 MS{I|D|ID} address{,bus op}
Bus op symbols:         P1, L1, R1
Example:                MS 3333,P1
```