Nova Simulator Usage
31-May-2008

COPYRIGHT NOTICE

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

This memorandum documents the Nova simulator.

1 Simulator Files

sim/	scp.h
	sim_console.h
	sim_defs.h
	sim_fio.h
	sim_rev.h
	sim_sock.h
	sim_tape.h
	sim_timer.h
	sim_tmxr.h
	scp.c
	sim_console.c
	sim_fio.c
	sim_sock.c
	sim_tape.c
	sim_timer.c
	sim_tmxr.c
sim/nova/	nova_defs.h
	nova_cpu.c
	nova_clk.c
	nova_dkp.c
	nova_dsk.c
	nova_lp.c
	nova_mta.c
	nova_plt.c
	nova_qty.c
	nova_sys.c
	nova_tt.c
	nova_tt1.c

2 Nova Features

The Nova simulator is configured as follows:

```
device names        simulates

CPU                 Nova, Nova 3, Nova 4 CPU with 32KW of memory,
                    Or Keronix CPU with 64KW of memory
-                   hardware multiply/divide
PTR,PTP             paper tape reader/punch
TTI,TTO             console terminal
TTI1,TTO1           second terminal
LPT                 line printer
PLT                 plotter
RTC                 real-time clock
DSK                 head-per-track disk controller
DKP                 moving head disk controller with four drives
MTA                 magnetic tape controller with eight drives
```

```
        QTY                 4060 multiplexer with up to 64 lines
        ALM                 4255 multiplexer with up to 64 lines
```

The Nova simulator implements these unique stop conditions:

- Reference to undefined I/O device, and STOP_DEV is set
- More than INDMAX indirect addresses are detected during an interrupt
- More than INDMAX indirect addresses are detected during memory reference address decoding

Note that indirect address loop detection does not exist on unmapped Novas. Some DG diagnostics test thousands of levels of indirect addressing. INDMAX may have to be set to 32,000 to get diagnostics to run properly.

The LOAD command supports standard binary format tapes. The DUMP command is not implemented.

All devices except TTI can be disabled or enabled, by the commands:

```
        SET <dev> DISABLED
        SET <dev> ENABLED
```

All devices except QTY are enabled by default.

2.1 CPU

The only CPU options are the presence of the optional instructions and the size of main memory.

```
        SET CPU MDV                 enable multiply/divide
        SET CPU EXT64KW             enable extended 64KW memory mode
        SET CPU NOVA3               set Nova3 CPU
        SET CPU NOVA4               set Nova4 CPU
        SET CPU KERONIX             set Keronix CPU
        SET CPU NONE                disable all optional instructions
        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
        SET CPU 36K                 set memory size = 36K
        SET CPU 40K                 set memory size = 40K
        SET CPU 44K                 set memory size = 44K
        SET CPU 48K                 set memory size = 48K
        SET CPU 52K                 set memory size = 52K
        SET CPU 56K                 set memory size = 56K
        SET CPU 60K                 set memory size = 60K
        SET CPU 64K                 set memory size = 64K

        (MDV = unsigned multiply/divide instructions)
        (Nova 3 = unsigned multiply/divide, stack, trap instructions)
        (Nova 4 = unsigned and signed multiply/divide, stack, byte,
                trap instructions)
        (Keronix = unsigned multiply/divide, extended 64KW mode)
```

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 CPU supports the `BOOT` command. `BOOT CPU` simulates the Nova hardware APL (automatic program load) feature. The switch register (SR) bits 12:17 must contain the device code of the device to be booted. If the device is a "high-speed" (channel) device, SR bit 0 should also be set.

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

```
     name              size         comments

     PC                15           program counter
     AC0..AC3          16           accumulators 0..3
     C                 1            carry
     SR                16           front panel switches
     PI                16           priority interrupt mask
     ION               1            interrupt enable
     ION_DELAY         1            interrupt enable delay for ION
     PWR               1            power fail interrupt
     INT               15           interrupt pending flags
     BUSY              15           device busy flags
     DONE              15           device done flags
     DISABLE           15           device interrupt disable flags
     STOP_DEV          1            stop on undefined IOT
     INDMAX            16           maximum number of nested indirects
     PCQ[0:63]         15           PC prior to last JMP, JMS, or interrupt;
                                    most recent PC change first
     WRU               8            interrupt character
```

The CPU can maintain a history of the most recently executed instructions. This is controlled by the `SET CPU HISTORY` and `SHOW CPU HISTORY` commands:

```
     SET CPU HISTORY              clear history buffer
     SET CPU HISTORY=0            disable history
     SET CPU HISTORY=n            enable history, length = n
     SHOW CPU HISTORY             print CPU history
     SHOW CPU HISTORY=n           print first n entries of CPU history
```

The maximum length for the history is 65536 entries.

2.2 Programmed I/O Devices

2.2.1 Paper Tape Reader (PTR)

The paper tape reader (PTR) reads data from 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
     BUSY              1            device busy flag
```

```
DONE              1           device done flag
DISABLE           1           interrupt disable flag
INT               1           interrupt pending 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 Paper Tape Punch (PTP)

The paper tape punch (PTP) 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
BUSY              1           device busy flag
DONE              1           device done flag
DISABLE           1           interrupt disable flag
INT               1           interrupt pending 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 Terminal Input (TTI)

The terminal input polls the console keyboard for input. Terminal input options include the ability to set ANSI mode or limited Dasher compatibility mode:

```
SET TTI ANSI                  normal mode
SET TTI DASHER                Dasher mode
```

Setting either TTI or TTO changes both devices. In Dasher mode, carriage return is changed to newline on input, and ^X is changed to backspace.

The terminal input implements these registers:

```
name              size          comments

BUF               8             last data item processed
BUSY              1             device busy flag
DONE              1             device done flag
DISABLE           1             interrupt disable flag
INT               1             interrupt pending flag
POS               32            number of characters input
TIME              24            keyboard polling interval
```

2.2.4 Terminal Output (TTO)

The terminal output writes to the simulator console window. Terminal output options include the ability to set ANSI mode or limited Dasher compatibility mode:

```
SET TTI ANSI                    normal mode
SET TTI DASHER                  Dasher mode
```

Setting either TTI or TTO changes both devices. In Dasher mode, carriage return is changed to newline on input, and ^X is changed to backspace.

The terminal output implements these registers:

```
name              size          comments

BUF               8             last data item processed
BUSY              1             device busy flag
DONE              1             device done flag
DISABLE           1             interrupt disable flag
INT               1             interrupt pending flag
POS               32            number of characters output
TIME              24            time from I/O initiation to interrupt
```

2.2.5 Line Printer (LPT)

The line printer (LPT) 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 printer.

The line printer implements these registers:

```
name              size          comments

BUF               8             last data item processed
BUSY              1             device busy flag
DONE              1             device done flag
DISABLE           1             interrupt disable flag
INT               1             interrupt pending flag
POS               32            number of characters output
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 paper

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

2.2.6 Real-Time Clock (RTC)

The real-time clock (RTC) line frequency can be adjusted as follows:

```
SET RTC 60HZ                    set line frequency to 60Hz
SET RTC 50HZ                    set line frequency to 50Hz
```

The default is 60Hz.

The clock implements these registers:

```
name            size            comments

SELECT          2               selected clock interval
BUSY            1               device busy flag
DONE            1               device done flag
DISABLE         1               interrupt disable flag
INT             1               interrupt pending flag
TIME0           24              clock frequency, select = 0
TIME1           24              clock frequency, select = 1
TIME2           24              clock frequency, select = 2
TIME3           24              clock frequency, select = 3
```

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

2.2.7 Plotter (PLT)

The plotter (PLT) 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 plotter.

The plotter implements these registers:

```
name            size            comments

BUF             8               last data item processed
BUSY            1               device busy flag
DONE            1               device done flag
DISABLE         1               interrupt disable flag
INT             1               interrupt pending 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
```

```
     error           STOP_IOE           processed as

     not attached    1                  report error and stop
                     0                  out of paper

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

2.2.8 Second Terminal (TTI1, TTO1)

The second terminal consists of two independent devices, TTI1 and TTO1. The additional terminal performs input and output through a Telnet session connecting into a user-specified port. The ATTACH command specifies the port to be used:

```
     ATTACH TTI1 <port>             set up listening port
```

where port is a decimal number between 1 and 65535 that is not being used for other TCP/IP activities.

Once TTI1 is attached and the simulator is running, the terminal listens for a connection on the specified port. It assumes that the incoming connection is a Telnet connection. The connection remains open until disconnected by the Telnet client, or by a DETACH TTI1 command.

The second terminal has two options, recognized on both devices, for setting limited Dasher-compatibility mode or ANSI mode:

```
     SET TTI1 ANSI                  normal mode
     SET TTI1 DASHER                Dasher mode
     SET TTO1 ANSI                  normal mode
     SET TTO1 DASHER                Dasher mode
```

Setting either TTI1 or TTO1 changes both devices. In Dasher mode, carriage return is changed to newline on input, and ^X is changed to backspace. TTO1 supports output logging. The SET TTO1 LOG command enables logging:

```
     SET TTO1 LOG=filename          log output to filename
```

The SET TTO1 NOLOG command disables logging and closes the open log file, if any.

Other special TTI1 commands:

```
     SHOW TTI1 CONNECTIONS          show current connections
     SHOW TTI1 STATISTICS           show statistics for active connections
     SET TTO1 DISCONNECT            disconnects the line.
```

The second terminal input implements these registers:

```
     name             size       comments

     BUF              8          last data item processed
     BUSY             1          device busy flag
     DONE             1          device done flag
     DISABLE          1          interrupt disable flag
     INT              1          interrupt pending flag
```

```
        TIME            24          keyboard polling interval
```

The second terminal output implements these registers:

```
        name            size        comments

        BUF             8           last data item processed
        BUSY            1           device busy flag
        DONE            1           device done flag
        DISABLE         1           interrupt disable flag
        INT             1           interrupt pending flag
        TIME            24          time from I/O initiation to interrupt
```

2.2.9 Asynchronous Multiplexers (QTY, ALM)

The QTY and ALM are terminal multiplexers with up to 64 lines. Either the QTY or ALM can be enabled, but not both; the ALM is enabled by default. The number of lines can be changed with the command

```
        SET {QTY|ALM} LINES=n       set line count to n
```

The line count maximum is 64.

The QTY and ALM support 8-bit input and output of characters. 8-bit I/O may be incompatible with certain operating systems; 7-bit is the default. The command

```
        SET {QTY|ALM} 8B
```

enables 8-bit input and output.

The terminal lines perform input and output through Telnet sessions connected to a user-specified port. The ATTACH command specifies the port to be used:

```
        ATTACH {-am} {QTY|ALM} <port> set up listening port
```

where port is a decimal number between 1 and 65535 that is not being used for other TCP/IP activities. For the ALM multiplexer, the optional switch -m turns on the multiplexer modem controls; the optional switch -a turns on active disconnects (disconnect session if computer clears Data Terminal Ready). The QTY multiplexer does not support modem control. Without modem control, the multiplexer behaves as though terminals were directly connected; disconnecting the Telnet session does not cause any operating system-visible change in line status.

Once the multiplexer is attached and the simulator is running, it listens for connections on the specified port. It assumes that the incoming connections are Telnet connections. The connection remains open until disconnected by the simulated program, the Telnet client, a SET {QTY|ALM} DISCONNECT command, or a DETACH {QTY|ALM} command.

Other special QTY/ALM commands:

```
        SHOW {QTY|ALM} CONNECTIONS    show current connections
        SHOW {QTY|ALM} STATISTICS     show statistics for active connections
        SET {QTY|ALM} DISCONNECT=n    disconnects the specified line.
```

The QTY/ALM implement these registers:

```
        name            size        comments
```

```
BUF               8           character buffer
BUSY              1           device busy flag
DONE              1           device done flag
DISABLE           1           device disable flag
INT               1           interrupt pending flag
MDMCTL            1           modem control flag
AUTODS            1           autodisconnect flag
POLLS             32          number of service polls
STOP_IOE          1           stop on I/O error
```

The multiplexers do not support save and restore. All open connections are lost when the simulator shuts down or the multiplexer is detached.

2.3 Fixed Head Disk (DSK)

Fixed head disk options include the ability to set the number of platters to a fixed value between 1 and 8, or to autosize the number of platters from the attached file:

```
SET DSK 1P                    one platter (256K)
SET DSK 2P                    two platters (512K)
SET DSK 3P                    three platters (768K)
SET DSK 4P                    four platters (1024K)
SET DSK 5P                    five platters (1280K)
SET DSK 6P                    six platters (1536K)
SET DSK 7P                    seven platters (1792K)
SET DSK 8P                    eight platters (2048K)
SET DSK AUTOSIZE              autosized on ATTACH
```

The default is 1P (minimum size). The fixed head disk controller supports the BOOT command.

The fixed head disk controller implements these registers:

```
name              size        comments

STAT              16          status
DA                16          disk address
MA                16          memory address
BUSY              1           device busy flag
DONE              1           device done flag
DISABLE           1           device disable flag
INT               1           interrupt pending flag
WLK               8           write lock switches
TIME              24          rotational delay, per sector
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           disk not ready
```

Fixed head disk data files are buffered in memory; therefore, end of file and OS I/O errors cannot occur.

2.4 Moving Head Disk (DKP)

Moving head disk options include the ability to make units write enabled or write locked, and to select the type of drive (or autosize):

```
SET DKPn LOCKED                 set unit n write locked
SET DKPn WRITEENABLED           set unit n write enabled
SET DKPn FLOPPY (or 6030)       set unit n to floppy disk
SET DKPn DSDD (or 6097)         set unit n to double density floppy
SET DKPn D31 (or 4047)          set unit n to Diablo 31
SET DKPn D44 (or 4234,6045)     set unit n to Diablo 44
SET DKPn C111 (or 4048)         set unit n to Century 111
SET DKPn C114 (or 2314,4057)    set unit n to Century 114
SET DKPn 6225                   set unit n to 6225
SET DKPn 6099                   set unit n to 6099
SET DKPn 6227                   set unit n to 6227
SET DKPn 6070                   set unit n to 6070
SET DKPn 6103                   set unit n to 6103
SET DKPn 4231 (or 3330)         set unit n to 4231
SET DKPn AUTOSIZE               set type based on file size at ATTACH
```

Units can also be set ENABLED or DISABLED. The moving head disk controller supports the BOOT command.

All drives have 256 16b words per sector. The other disk parameters are:

```
drive           cyl surf sectors size (MW)    model numbers

floppy          77   1    8        .158       6030
dsdd floppy     77   2    16       .632       6097
D31             203  2    12       1.247      4047, 4237, 4238
D44             408  4    12       5.014      4234, 6045

C111            203  10   6        3.118      4048
C114            203  20   12      12.472      2314, 4057
6225            245  2    20       2.508
6099            192  4    32       6.291
6227            245  6    20       7.526
6070            408  4    24      10.027
6103            192  8    32      12.583
4231            411  19   23      45.979      3330
```

The moving head disk controller implements these registers:

```
name             size         comments

FCCY             16           flags, command, cylinder
USSC             16           unit, surface, sector, count
STAT             16           status
MA               16           memory address
BUSY             1            device busy flag
DONE             1            device done flag
DISABLE          1            interrupt disable flag
INT              1            interrupt pending flag
DIAG             1            diagnostic mode flag
```

```
MAP                  2              map select
STIME                24             seek time, per cylinder
RTIME                24             rotational delay
```

Error handling is as follows:

```
error                processed as

not attached         disk not ready

end of file          assume rest of disk is zero

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

2.5 Magnetic Tape (MTA)

Magnetic tape options include the ability to make units write enabled or write locked.

```
SET MTAn LOCKED                  set unit n write locked
SET MTAn WRITEENABLED            set unit n write enabled
```

Magnetic tape units can be set to a specific reel capacity in MB, or to unlimited capacity:

```
SET MTAn CAPAC=m                 set unit n capacity to m MB (0 = unlimited)
SHOW MTAn CAPAC                  show unit n capacity in MB
```

Units can also be set ENABLED or DISABLED. The magnetic tape controller supports the BOOT command.

The magnetic tape controller implements these registers:

```
name                 size           comments

CU                   16             command, unit
MA                   16             memory address
WC                   16             word count
STA1                 16             status word 1
STA2                 16             status word 2
EP                   1              extended polling mode (not supported)
BUSY                 1              device busy flag
DONE                 1              device done flag
DISABLE              1              interrupt disable flag
INT                  1              interrupt pending flag
STOP_IOE             1              stop on I/O error
CTIME                24             controller delay
RTIME                24             record delay
UST[0:7]             32             unit status, units 0-7
POS[0:7]             31             position, units 0-7
```

Error handling is as follows:

```
error                processed as

not attached         tape not ready

end of file          bad tape
```

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

3 Symbolic Display and Input

The Nova 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 standard Nova assembler syntax. There are three instruction classes: memory reference, IOT, and operate.

Memory reference instructions have the format

```
memref {ac,}{@}address{,index}
```

LDA and STA require an initial register; ISZ, DSZ, JSR, and JMP do not. The syntax for addresses and indices is as follows:

```
syntax             mode  displacement       comments

0 <= n < 0400      0     n
{+/-}n >= 0400     1     {+/-}n - PC        must be in range [-200, 177]
                                            invalid on disk
.+/-n              1     {+/-}n             must be in range [-200, 177]
{+/-}n,2           2     {+/-}n             must be in range [-200, 177]
{+/-}n,3           3     {+/-}n             must be in range [-200, 177]
```

IOT instructions have one of four formats:

```
syntax              example

iot                 HALT
iot reg             INTA
iot device          SKPDN
iot reg,device      DOAS
```

Devices may be specified as mnemonics or as numbers in the range 0 - 077.

Operate instructions have the format:

```
opcode{#} reg,reg{,skip}
```

In all Nova instructions, blanks may be substituted for commas as field delimiters.