OPERATING NOTICE

B1471-92003

THIS PAGE MAY BE DISCARDED AFTER YOU PLACE THE DOCUMENT UNDER IT INTO THE BINDER.

Operating Notice
Improving Simulated I/O Performance

With HP 64700 system firmware version 4.0 or greater, much of the simulated I/O functionality has been moved to the emulation card cage. This improves simulated I/O performance. However, you may have to change your environment dependent compiler libraries so that simulated I/O operates the same as it did with pre-4.0 firmware.

Also, for the best simulated I/O performance, make sure the simulated I/O control address is placed in dual-port emulation memory.

This notice shows you how to:

- Open stdin with ndelay turned OFF.
- Place the simulated I/O control address is in dual-port emulation memory.

Note

With HP 64700 firmware version 4.0 or greater, some simulated I/O operations complete without host intervention or notification (for example, reading from a file descriptor that has not been opened). When simulated I/O status messages are enabled (by answering "yes" to the "Enable simio status messages?" configuration question), there can be some simulated I/O commands that are not reported.

In addition, the simulated I/O write operation is now completed in the card cage, and any errors from write operations will not be reported on the write command that caused the error. Eventually, the host will report to the card cage that a write has failed, and any write operations after this will receive the error.

Opening stdin With Ndelay Turned OFF

With HP 64700 system firmware version 4.0 or greater, ndelay is now implemented for simio_keyboard.

Current and past versions of the HP Advanced Cross Language System compiler libraries open stdin with ndelay turned ON. This causes simio to do non-blocking reads of the keyboard (reads will return 0 anytime there is no data available in the emulator box).

To turn OFF non-blocking reads of the keyboard, you must modify the file /usr/hp64000/env/hp64XXX/src/startup.c (for example, /usr/hp64000/env/hp64747/startup.c for the 64747 emulator) and build a new environment dependent library. The line:

```
stdin->_file = open_file(STDIN, O_READ|O_NDELAY);
```

should be changed to:

```
stdin->_file = open_file(STDIN, O_READ);
```

The procedure for modifying and building a new environment dependent library is described in the *User's Guide* for each compiler in the section "Modifying the Environment Dependent Libraries" in Chapter 1.

Another way of modifying your code to turn OFF non-blocking reads of the keyboard is to close STDIN and reopen it without the O_NDELAY flag. This can be done at the beginning of your main routine or in some initilization routine before any reads from stdin. The following code segment can be used:

```
#include "stdio.h"
main()
{
   int fd;

   close(0);
   if ((fd = open("stdin", O_READ)) != 0)
   {
      fprintf(stderr, "cannot open stdin, descriptor returned = %d\n"
         fd);
      exit(1);
   }

   /* continue with your code here */
}
```

Note that you cannot open any other files between the close(0) and the open of "stdin". The libraries assume that stdin is file descriptor 0 and will not work correctly if stdin is a different descriptor. After closing file descriptor 0, the next open command will return descriptor 0 (the test is just for saftey).

Placing the Control Address in Dual-Port Emulation Memory

To get the most performance out of simulated I/O, the control address must be located in dual-port emulation memory. This is true for any version of HP 64700 system firmware.

For most emulators, all emulation memory is dual-port; however, for the 68020, 68EC030, 68030, 68340, and other emulators, only a small portion of emulation memory is dual-port. Note that dual-port memory for these emulators is only available when using the background monitor. If you are using a foreground monitor, you cannot get this performance improvement.

To place the control address into dual-port memory, the file /usr/hp64000/env/hp64XXX/iolinkcom.k must be modified to separate the section "envdata" from the other sections so that it can be put in a specific location in memory. Also, the emulation configuration file must be modified to place the specified locations in dual-port memory instead of normal emulation memory.

The following lines in the file iolinkcom.k should be changed from:

```
ORDER envdata,data,idata,udata,libdata,libcdata,mondata,heap
SECT  envdata=$FFFE8000         ; Load address for data sections
```

to:

```
ORDER data,idata,udata,libdata,libcdata,mondata,heap
SECT  envdata=$FFFE8000         ; Load address for data sections
SECT  data=$FFFE9000            ; Load address for data sections
```

Also, change the following lines in the file ioconfig.EA (or modify the configuration) from:

```
#-- Map 96k bytes for all data sections and heap
0FFFE8000H thru 0FFFFFFFFH emulation ram
```

to:

```
#-- Map 96k bytes for all data sections and heap
0FFFE8000H thru 0FFFE8FFFH emulation ram dp
0FFFE9000H thru 0FFFFFFFFH emulation ram
```

This puts the segment envdata containing the simulated I/O control address in dual-port memory at 0FFFE8000H and puts the remaining data addresses starting at 0FFFE9000H.

If you are using your own linker command file instead of the supplied linker command file, you need to make similar changes to seperate the section envdata and locate it in a region of memory mapped to dual-port memory.