Errata

OPERATING SYSTEM CONCEPTS, FOURTH EDITION

Silberschatz and Galvin

January 3, 1995

Corrected in 2nd Printing:

page x (Preface), **Acknowledgments**: Replace first 2 lines of first paragraph with: "This book is derived from the previous editions, all of which were coauthored by James Peterson. Other people that have helped with the previous editions include..."

page x (Preface), **Acknowledgments**: Add to first line of second paragraph: "Cliff Wilkes provided technical copy-editing;"

page xii (Contents), title for Chapter 5: "CPU Processing" $\Longrightarrow$ "CPU Scheduling"

page 187, line 4 from bottom: "**while** statement S" $\Longrightarrow$ "while statement S"

page 200, line 3: seize $\Longrightarrow$ cease

page 285, Figure 8.23: Revise the arrows from the segment number, offset pair so that the arrow from "s" points to the "segment table", and the arrow from "d" points to the "less than" box.

page 295, Figure 8.28: In top half of figure, "segment discriptor" $\Longrightarrow$ "segment descriptor"

page 399, change title for Section 11.4.1 to "Linear List"

page 515, line 10: as "needed" $\Longrightarrow$ "as needed"

Corrected in 3rd Printing:

page 31, Figure 2.2 caption: revise to read "Interrupt time line for a single process doing output."

The following revisions have been made to pages 33-36:

1. Figure 2.4 is exchanged with Figure 2.3 and is recaptioned:
   Figure 2.3 Two I/O methods. (a) synchronous. (b) asynchronous.
   New Figure 2.3 appears on page 33.

2. A reference to new Figure 2.3 appears on page 33, line 18: "other system operations occur (see Figure 2.3)."

3. New "**Figure 2.4** Device-status table." appears on page 34. The reference to this figure on page 34 is revised to agree with the new figure number.

4. The last text line on page 36: "A comparison of standard interrupt-driven I/O and DMA I/O is shown in Figure 2.4" is deleted.

page 76, line 22: "Both use the same set of system calls, but they appear and act dissimilar." $\Longrightarrow$ "Both use the same set of system calls, but the calls look different and act in different ways."

page 78, Figure 3.7: "kernal interface to the hardware" $\Longrightarrow$ "kernel interface to the hardware"

page 112, line 1: "stack register" $\Longrightarrow$ "stack pointer"

page 113, Fig. 4.8: Remove the dots next to the threads in the text segment.

page 116, lines 3-5: Delete the sentence "No LWP is needed for user-level threads that do not make system calls concurrently with other threads."

page 116, lines 17-20: Delete the sentence "Further, even if all the task's LWPs are blocked, user-level threads may continue to run, as long as they do not need to contact the kernel (which they can do only through a lightweight process)."

page 172, revise 1st sentence of 2nd paragraph as follows: "The critical-section problem could be solved simply in a uniprocessor environment if we could disallow..."

page 172, revise 1st sentence of 3rd paragraph as follows: "Unfortunately, this solution is not feasible in a multiprocessor environment."

page 179, line 16: "...critical section is almost always empty," $\Longrightarrow$ "...critical section is almost never occupied,"

page 180, line 4 from bottom: $signal(S2);$ $\Longrightarrow$ $wait(S2);$

page 191, replace the second paragraph with the following:
"The monitor construct ensures that only one process at a time can be active within the monitor. Consequently, the programmer does not need to code this synchronization constraint explicitly (Figure 6.17). However, the monitor construct, as defined so far, is not sufficiently powerful for modeling some synchronization schemes. For this purpose, we need to define additional synchronization mechanisms. These mechanisms are provided by the *condition* construct. A programmer who needs to write her own tailor-made synchronization scheme can define one or more variables of type *condition*:"

page 267, line 3 from bottom: "permit the logical address space" $\Longrightarrow$ "permit the physical address space"

page 269, table midpage: change page offset from m to n

pages 278-279, replace the last paragraph on page 278, including the portion of that paragraph which extends onto page 279, and the first full paragraph on page 279 with the following:

> For a system with a 64-bit logical address space, a two-level paging scheme is no longer appropriate. To illustrate this point, suppose that the page size in such a system is 4K bytes (22). In this case, the page table will consist of up to 22 entries. If we use a two-level paging scheme, then each page in the inner page table would contain 20 4-byte entries. The outer page table will thus consist of 22 entries, or 244 bytes. Clearly, we would not want to allocate the outer page table contiguously in main memory. The obvious solution is to divide the outer page table into smaller pieces. This solution is also used on some 32-bit processors for added flexibility and efficiency.
>
> There are a number of different ways to accomplish this. We can use a three-level paging scheme, where the outer page table itself is also paged (resulting in a second-level outer page). Even with a third level of page tables, each one page in length, a 64-bit address space is daunting, since the second outer page table is still 22 bytes large.
>
> The next step would be a four-level paging scheme, where the second- level outer page table itself is also paged. The SPARC architecture (with 32- bit addressing) supports a three-level paging scheme, whereas the 32-bit Motorola 68030 architecture supports a four-level paging scheme.

page 281, Figure 8.20: page table entry marked "pid" should be entries "pid" and "p"; "f" $\Longrightarrow$ "i"

page 295, Figure 8.28: change the label "segment register" to "page directory base register"

page 394, line 6 under **Multilevel index**.: "2048-byte blocks" $\Longrightarrow$ "4096-byte blocks"

page 394, line 7 under **Multilevel index**.: "512 4-byte pointers" $\Longrightarrow$ "1024 4-byte pointers"

page 394, line 8 under **Multilevel index**.: "4,194,304 data blocks" $\Longrightarrow$ "1,048,576 data blocks"

page 394, line 9 under **Multilevel index**.: "8.5 gigabytes" $\Longrightarrow$ "4 gigabytes"

page 394, line 10 under **Multilevel index**.: "of an individual disk drive." $\Longrightarrow$ "of most individual disk drives."

page 421, line 5 from bottom: "entry i is of size $i * 16$K" $\Longrightarrow$ "entry i is of size $2^i * 16$K"

page 423, Figure 12.8: Remove the blank space between the 64K and the 128K blocks, reverse the placement of the 64K and 128K data segments, and uncross the pointers (so each entry points to the next-larger entry).

page 514, Figure 16.1: In the left column, 2nd box down, revise "part number" to read "port number".

page 716, line 12 from bottom: "(December 1969)" $\Longrightarrow$ "(December 1981)"

Corrected in 4th Printing:

page 128, Ex. 4.8: "Section 4.6" $\Longrightarrow$ "Section 4.4"

page 141, line 16 from bottom: "5.75 milliseconds." $\Longrightarrow$ "7.75 milliseconds."

page 163, line 7 from bottom: "Section 4.6" $\Longrightarrow$ "Section 4.4"

page 362, second paragraph, second line: "...*volume table* of contents." $\Longrightarrow$ ..."*volume table of contents.*"

page 523, lines 11-10 from bottom: change Artsy [1989a] to Artsy [1989b]

page 523, line 9 from bottom: change Artsy [1989b] to Artsy [1989a]

Backcover time line, entry for 1970: "kernal" $\Longrightarrow$ "kernel"

To Be Corrected:

page 80, lines 12-14: Revise sentence as follows.
"Because it, as well as I/O buffering (level 4), were placed above memory management, the device buffers were able to be placed in virtual memory."

page 124, Revise last sentence on page as follows.
"Error checking codes (such as checksums, parity, and CRC) are commonly used to detect this type of error."

page 141, line 9 from bottom: Delete the phrase "(τ): $p = 1/\tau$ "

page 191, (in 3rd and 4th printing only): the phrase "*x.wait;*" should appear as a centered display following the last line of text "The operation" and above the graphic for Figure 6.17.

page 230, Change the last sentence before Section 7.5.3 to read:
"If P_1 requests R_2, and P_2 requests R_1, then a deadlock will occur."

page 230 and 231, Figures 7.5 and 7.6: In each figure, the edge from P_2 to R_1 should be a dashed arrow.

page 264, line 20: "...is clearly best..." $\Longrightarrow$ "...is clearly better..."

page 272, line 6: "user" $\Longrightarrow$ "process"

page 278, lines 3-6: Revise paragraph as follows.
"A one-level page table for a VAX process using one segment still would be 2^{21} bits $*$ 4 bytes per entry = 8 megabytes. To further reduce main memory use, the VAX pages the user process page tables.

page 278, lines 11-13: Revise text as follows.
"...each page in the inner page table would contain 2^9 8-byte entries. (Eight bytes are needed to store the 52-bit pointer to the physical page.) The outer page table will thus consist of $2^{52}/2^9 = 2^{43}$ 8-byte entries, or 2^{46} bytes."

page 279, line 8: "...2^{32} bytes large." $\Longrightarrow$ "...2^{34} bytes large."

page 281, Figure 8.20 (in 4th printing only): The arrow from the page table to the address should point to "i", not "d"

page 285, line 7: "...segments resides..." $\Longrightarrow$ "...segment resides..."

page 344, Exercise 9.8: Add the following new second sentence to the Exercise.
"It costs an additional 1 microsecond to access a page other than the current one."

page 345, line 3: "0.1 percent" $\Longrightarrow$ "1 percent"

page 357, lines 5-10: Join second paragraph to first paragraph, and revise the text as follows.
"... structures. For instance, DEC's popular VMS operating system has a file system that does support multiple file structures. It defines three file structures."

page 357, line 7 from bottom: "executables to contain" $\Longrightarrow$ "files to contain"

page 357, last two lines: "...data that the user may not change." $\Longrightarrow$ "...data: the traditional file contents."

page 613, line 1 of Section 19.2: "UNIX as..." $\Longrightarrow$ "UNIX was..."

Backcover time line, entry for 1965: Delete the line "Control Data PDP-8, 1st commercial supercomputer"