Tech Info Library

Parity Checking: Why Apple Doesn't Use It

Revised: 6/17/92
Security: Everyone

Parity Checking: Why Apple Doesn't Use It

===

Article Created:
Article Last Reviewed: 1 June 1992
Article Last Updated: 1 June 1992

Background: Why Parity Checking Came About

Parity checking first became an issue when computer manufacturers started using early DRAM (Dynamic Random Access Memory) technologies. These chips were quite unreliable, and since they were relatively small (1KB - 4KB), vendors had to use a large number of them (increasing the odds of failure) to produce a system with a useful amount of memory. In that environment, parity checking ensured that if a soft error (one that can't be reproduced) occurred, a user would not be able to save potentially corrupted data back to disk.

Apple's Approach: Increased DRAM Reliability

Apple took a different approach and worked with its chip vendors to increase DRAM reliability. The result has been that each new generation of DRAMs seems to be twice as reliable as the previous generation. The mean time between soft errors doubles, even though the chip capacity quadruples. Thus, for a given amount of memory, each new generation of DRAMs has eight times the reliability of its predecessor.

As a practical matter, the reliability of current computer technology is not gated by the reliability of the hardware: system and application software fail (and corrupt data) several orders of magnitude more often than the hardware on which they run. There are also several good engineering reasons why Apple doesn't use parity checking:

- Cost. In addition to requiring more RAM, additional circuitry must be added to the logic board to detect parity errors.

- No Significant Reliability Improvement. The 256K DRAMs we currently use typically experience soft errors every 1,000,000 hours per device, or once every 3.5 years for a 1MB Macintosh system.

- No Real Protection. How a system reacts to a parity error is at least as important as checking for one in the first place. Most MS-DOS PCs react poorly and crash the system when they detect a parity error, threatening both the user's files and file system.

Apple is not alone in these conclusions. While early versions of IBM's 360 series of mainframes used parity checking, more recent versions have moved towards "error correcting code" to maintain system integrity.

System Reliability and System Performance

The Macintosh already checks its memory for hard failures as a part of the startup sequence. Apple could also adopt an error correction scheme similar to that used in most of today's mainframes, and totally protect the user against single bit soft errors. Essentially, this approach adds three bits to each byte so that the system can detect an error and correct it. This approach is expensive, and would require substantial changes to both our operating system and hardware.

More important, both parity checking and error correction code would impact the overall performance of future Macintosh systems. In essence, both these schemes require that the hardware detect a soft error in less time than it takes the microprocessor to execute an instruction. As Apple moves to faster microprocessors, less time is available for the hardware to test all of the memory during each instruction cycle. Given the choice between investing in faster, more reliable DRAM technology (and hence, faster systems) or investing in a parity checking algorithm that constrains system performance, most users would prefer the former. For customers who require parity checking, Apple does offer a model of the Macintosh IIci with parity checking.

Tech Info Library Article Number:2661