CS510 Languages and Low Level Programming: Portfolio submission Topic 3: Explain how conventional operating system features (multiple address spaces, context switching, protection, etc.) motivate the desire for (and benefit from) hardware support.

Due on April 29, 2016 at 11:59pm

Mark P. Jones Spring 2016

Konstantin Macarenco

The main benefit of implementing some of the OS features in hardware is performance and security. Specialized hardware will be in most of the cases many times faster than general software. Some tasks are simply not feasible without some level or hardware support.
Software approach reduces cost and complexity of hardware centric systems. And vice versa performance of software is increased by special purpose hardware.
When some concept or a feature becomes widely adopted there is a good chance that it will be adopted in hardware, however it is not always the case, or at least not always successful (context switching).

Memory Management

Programming first computers without hardware's aid was a big challenge. What to do when a task runs out of memory? How to fit a large applications into available RAM? How to protect one task from corrupting another's task memory? Etc.
Memory Management hardware infrastructure development was driven by the concept of Virtual Memory which required to support of multiple address spaces, and need of fast dynamic translation between virtual and physical addresses. This is a hard and complex task that would be too slow and prone to security problems if implemented in pure software.
So specific hardware was created to solve Virtual Memory problems, which include but limited to

- Fast memory translation.
- Address Space Switch.
- Address Space isolation.
- Memory overflow protection (swapping).

Memory space isolation was a big problem in earlier OSs. Memory protection tries to solve problem of a process affecting other processes or the OS memory. Without hardware support this problem is impossible to solve since during execution the current process is in total control and can do anything without proper protection mode. Protection rings, segmentation and paging enable this feature. OS creates page table and changes cr3 to point to the current page table. This is a privileged instruction and can be only done in ring0.

This level of protection is impossible without proper hardware support, I see two possible software solutions (neither is as good as having hardware support):

- An OS provides virtualized environment (virtual machine that implements hardware protection). This approach is close to be as good as real hardware protection but struggles from performance problems.
- The OS can periodically scan memory for violations. This approach is more of a debugging tool, it is a lot less secure and doesn't prevent from interprocess memory access.

Context Switching

I was surprised to learn, while researching this topic, that context switch in modern OSs is a done by software. Intel task switch mechanism nowadays is used mainly to transfer from one protection level to another. Linux creates only one kernel TSS since it's required to enable protection and uses only ESP0 and SS0 to switch between security rings. Reasons behind using software - hardware context switch is appeared to be slower less flexible and not portable from x86 to x86_64. X86 during context switch saves lots of information that not all applications need, making already costly operation even worse.

Conclusion
To be adopted in hardware a concept must be widely accepted an versatile enough not to introduce any unnecessary assumptions. Memory management is a good example of such a feature, and context switching, is the opposite.