Virtual 8086 mode

M. Sonza Reorda

Politecnico di Torino
Dipartimento di Automatica e Informatica

Motivation

It would be attractive to allow old programs developed in real mode to run in a multitasking environment in parallel with other programs running in protected mode.

Virtual machine

The CPU hardware and a proper component of the OS (named *monitor*) let the program to see an environment similar to the one it was developed for:

- Real mode for address computation
- 1 Mb of memory
- Set of virtual registers
- Accesses to system functions like if it was the only program running on the system.

Each virtual machine undergoes the task switching mechanism, thus implementing multitasking.

Address computation

In virtual mode addresses are computed as in real mode, e.g., by adding an offset and a properly multiplied segment register.

Since the offset is now stored in a 32-bit register (while in the 8086 was stored in a 16-bit register), to obtain the same behavior a check is performed, and an exception is triggered if the offset value is greater than ffffh.

Entering virtual mode

The processor switches to virtual mode when the VM bit in the EFLAG register is set.

The VM flag can only be set by

- Codes with the privilege level equal to 0
- A task switch through a TSS (in this case the OS only needs to suitably prepare the EFLAG value when setting up the TSS for the task)
- An IRET instruction (which loads the new value of EFLAG from the stack).

The processor must already be working in protected mode.

Using OS functions

The program working in virtual mode may need to call OS functions; but the OS on the machine is different, and is shared with other tasks.

Possible solutions are:

- The 8086 OS runs as part of the 8086 program
- The OS emulates the 8086 OS.