

## Course 207

# BIOS Essentials

## A 2-day Course

This course focuses on the PC's traditional firmware design – BIOS and its POST (Power-On Self Test) sequence. Using actual BIOS code sequences it shows how built-in controllers are initialized, how PCI devices and functions are discovered and configured, how expansion ROMs are detected, validated and executed. The course also describes the various data tables that are created during POST.

If you design, develop, test, or validate hardware or software platforms, ACPI, PCI, or firmware based expansion ROM capabilities that interact with BIOS this course is for you!

### You will benefit from this workshop if you

- Design, configure, or validate hardware and low-level software that depends on data structures created during the POST
- Need a detailed understanding of what happens in a PC before the operating system is loaded and given control
- Are migrating from BIOS to *The Framework (Tiano)* and UEFI based systems

### You will learn

- The components of BIOS
- Fundamentals of EFI and *The Framework (Tiano)*, Intel's approach to developing the next generation of PC firmware
- Details of each step from power-on to operating system boot loader hand-off
- What typical BIOS code looks like
- How legacy ISA devices are handled
- How PCI devices are detected and configured
- How ACPI power management is initialized
- The firmware's ongoing participation in running the PC
- How to troubleshoot the boot sequence
- The system software architecture

### Prerequisites

This course is designed for technical managers, engineers and high-level technicians who need to increase their knowledge about the PC's BIOS-based initialization and boot sequence.

Attendees are expected to have a technical background. Our *PC Architecture Overview* is a good primer for this workshop. Basic knowledge of assembly language programming, microprocessor technology, memory, and standard peripherals is expected.

### The training approach

- **Up to date information:** We update the materials before every event.
- **Straightforward explanations:** Technical concepts are explained in plain English.

## Workshop topics

### Firmware basics

- Why do we need firmware?
- What is UEFI, *Tiano*, and BIOS?
- Where is the firmware?

### Fundamental hardware concepts

- Architecture of the original PC
- What has been added
- What has been removed
- Architecture of today's PCs
- The cost of backward compatibility
- Real and protected mode memory maps
- The I/O port map
- x86 CPU registers
- Real mode capabilities
- Making sense of Segmented Memory

### The POST process

- What happens before the POST (Power-On Self Test)?
- Accessing the POST code
- Reading configuration settings from NV-RAM
- Chipset initialization
- How video is initialized
- How memory is tested
- "Walking" the PCI buses to identify and initialize installed PCI devices and functions
- Initialization of key legacy functions and devices
- USB and IEEE1394 ("Firewire") initialization
- Memory map after POST
  - The IVT (Interrupt Vector Table)
  - BDA and EBDA content and layout
- Classic hand-off to operating system
- ACPI system initialization – The RSDP and RSDT

### A new way to boot

- Why change?
- What is *UEFI* and where does it fit in?
- Major advantages over BIOS
- *Tiano* – A phased approach to creating the *UEFI*
- New driver models
- The *UEFI System Partition* layout
- The new boot process and sequence
- Legacy support

**Operating system configuration**

- The PC – A layered software architecture
- Definition of the layers
- Windows, Linux, and DOS boot sequences
- Use of information collected by POST

**Troubleshooting the boot sequence**

- Troubleshooting a PC – A three-step approach
- Common problems
- Beep codes
- Monitoring POST-codes
- Tools for development and manufacturing environments
  - ICE – In-Circuit Emulators
  - ITP – In-Target Probes
  - Logical Analyzers
  - JTAG and TAP interfaces