Chapter 7

The Components of Intel® Active Management Technology

What we need to do is learn to work in the system, by which I mean that everybody, every team, every platform, every division, every component is there not for individual competitive profit or recognition, but for contribution to the system as a whole on a win-win basis.

—W. Edwards Deming (1900–1993)

In the previous chapters we described Intel® Active Management Technology (Intel AMT) platforms from a functional standpoint, that is, the kind of problems Intel AMT solves and the collection of features that help solve those problems. This chapter presents an overview of the architecture of Intel AMT. This includes the architecture of the hardware, firmware, and software, all of which need to come together to make Intel AMT work. At a very high level, Figure 7.1 shows how these pieces stack up.

The hardware for Intel AMT is comprised of the Intel chipset (which includes the wired Ethernet network controller), wireless communication chip, nonvolatile flash memory for code and data storage, and the communication links and buses. The firmware comprises the runtime environment, kernel, drivers, services, and firmware applications. The software is made up of OS drivers, OS/ISV agents, and console applications. Console software could reside on the same computer's OS (in the form of host-based services), or on a different computer located somewhere else on the network (network-

based console). Several vendors (such as Microsoft, Symantec, and LANDesk) provide console applications and suites that utilize Intel AMT capabilities for platform management. An up-to-date and complete list of ISV software suites that work with Intel AMT is available on Intel's Web site.

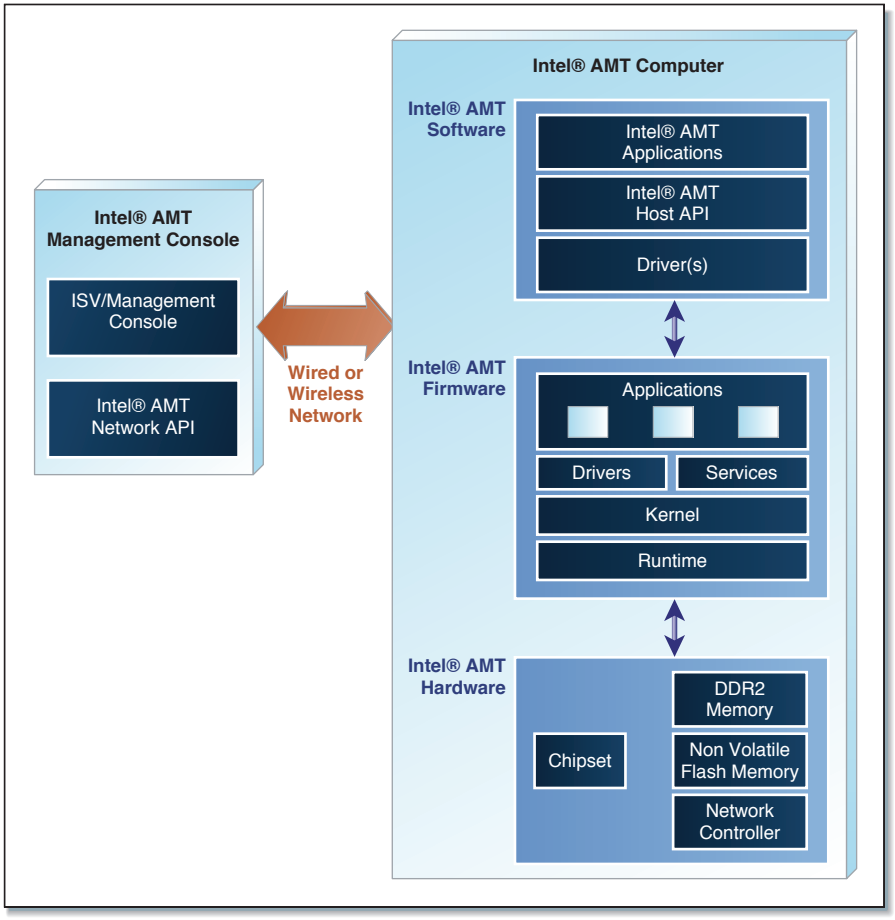

Figure 7.1 High Level View of Intel® AMT components

The next few sections go deeper into the architecture and design of the hardware, firmware, and software components that constitute Intel AMT.

One point to note here is that the architecture described in this chapter is based on the currently shipping Intel platform architecture, which includes

the CPU and the two-chip chipset—Northbridge and Southbridge (see Figure 7.2). In the new architecture, a majority of Northbridge functionality is being integrated into the CPU Core and Uncore, and the rest of the functionality is being combined with the Southbridge (see Figure 7.3). This Southbridge chip is internally called as the PCH (Platform Controller Hub) within Intel. This is a very significant change to the platform architecture, but from the standpoint of Intel AMT, the architecture of all of the components described below remains largely unchanged.

Hardware Architecture

The primary hardware pieces of Intel AMT reside in the Intel chipset (Northbridge and Southbridge). The Northbridge, also called the Graphics and Memory Controller Hub (GMCH), contains a micro controller called the Intel Manageability Engine (Intel ME), which is the heart of Intel AMT. It is the engine that runs the Intel AMT firmware services and applications. The GMCH also contains the memory controller that provides access to the system memory. A small portion of the system memory is used by the Intel ME for its runtime memory needs. This memory is separated from the memory accessed by the OS using special hardware mechanisms. The architectural mechanism that creates the separation is called Unified Memory Architecture (UMA). The Southbridge, also called the I/O Controller Hub (ICH), contains the Ethernet network controller, network filters, and non-volatile flash memory controller, among other things. The wireless LAN (Wi-Fi†) network controller is connected to the ICH over a PCI Express† bus. The network controllers, wired LAN (Ethernet) and wireless LAN (Wi-Fi), and the network filters provide out-of-band (OOB) communication access to the Intel ME. OOB communication allows the Intel ME to communicate over the network without having any dependence on the OS or the drivers that reside therein. OOB communication is capable of working even when the computer is in some states where the OS is not working or is sleeping, such as when the OS has crashed, or is in standby state or hibernate state. The flash controller in the ICH provides access to the flash memory, also called the nonvolatile memory (NVM), on the computer's motherboard. This NVM

houses the BIOS code, Intel ME code, and data, among other things. The GMCH and ICH communicate with each other using the DMI[1] bus and Controller Link (C-Link) bus. The C-Link bus is a proprietary interface that can be used even when the computer is in sleep or hibernated states, in addition to being used when the OS is operational. Figure 7.2 shows the hardware architecture of the Intel CPU and chipset based computer. We won't go into details of each and every component shown in the picture, but we will cover a good amount of detail on the components that make up Intel AMT, in the next few sections.

1 Direct Media Interface (DMI) is the chip-to-chip interconnect between the ICH and the (G)MCH. This high-speed interface ensures that the I/O subsystem (PCI Express, Intel High Definition Audio, SATA, USB, and so on) receives the bandwidth necessary for peak performance.

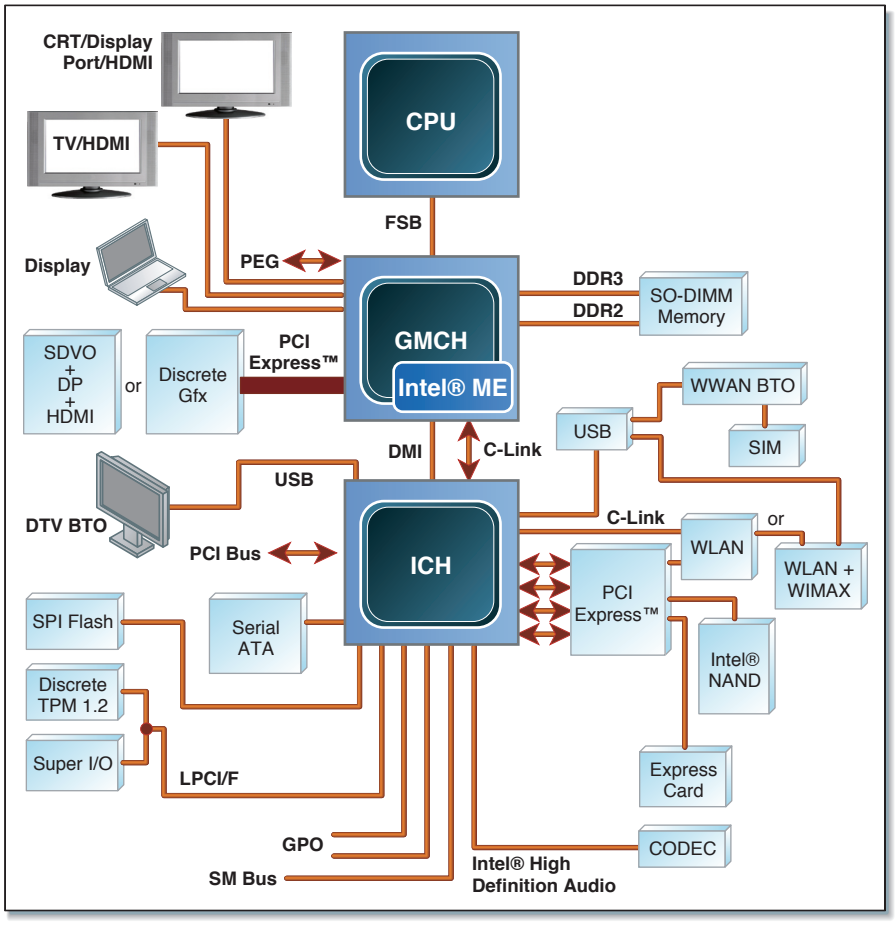

Figure 7.2 Hardware Architecture of an Intel Platform

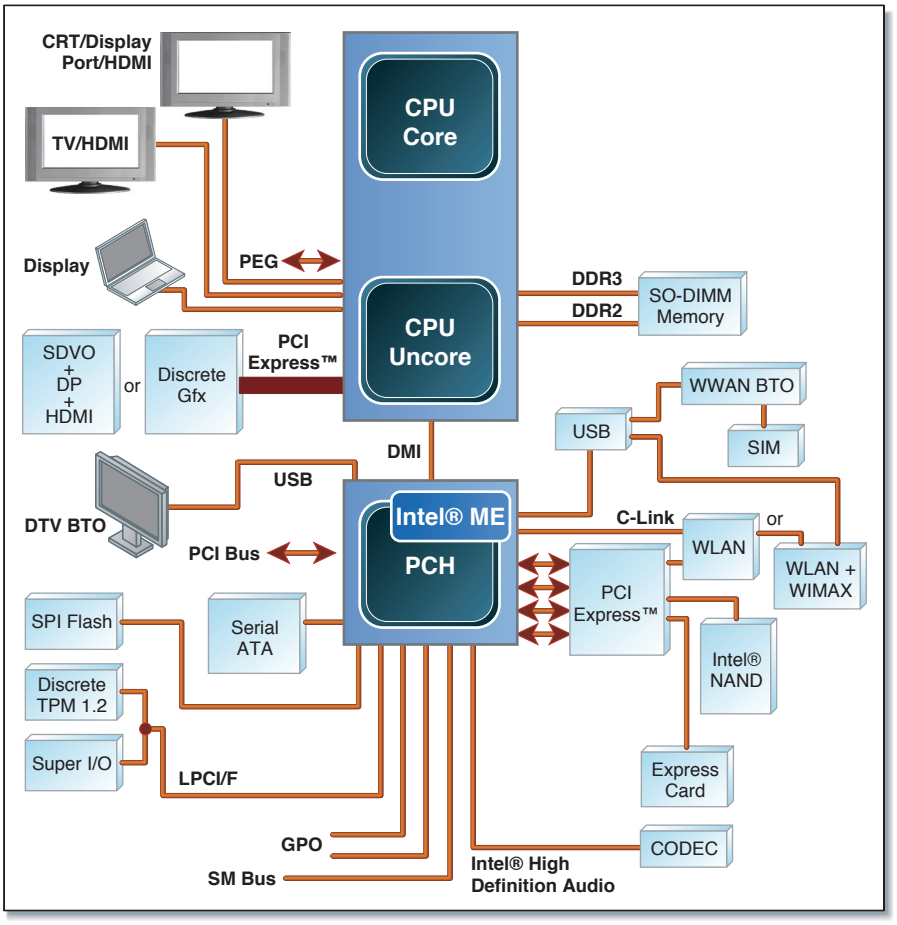

Figure 7.3 Hardware Architecture of Future Intel Platforms

Intel® Manageability Engine (Intel ME)

The Intel Manageability Engine is the core hardware component of Intel AMT. A high level picture of the Intel ME is shown in Figure 7.4.

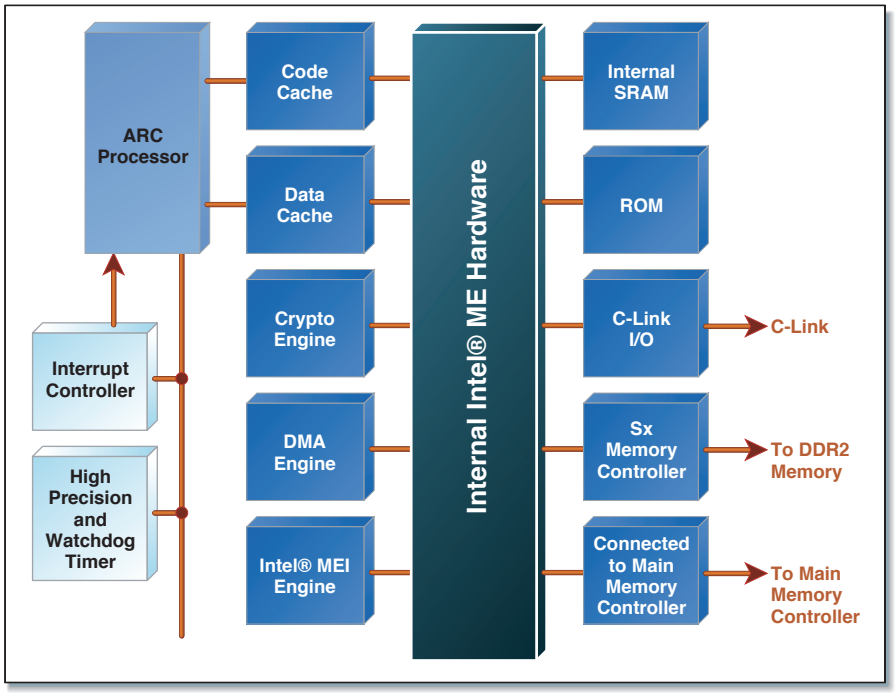

Figure 7.4 Hardware Architecture of the Intel® Manageability Engine

The Intel ME is made up of several components such as an ARC processor, code and data caches, a DMA engine, a crypto engine, ROM, C-Link interface, Intel MEI, memory controller interface, interrupt controller, and high precision and watchdog timers. These are connected together over an internal hardware bus.

The code and data caches help to accelerate Intel ME functionality by reducing the memory accesses to system memory. The DMA engine helps to move data to and from the OS memory and Intel ME UMA memory. However this DMA engine is only accessible by the Intel ME, not the OS. Also, it does not provide any generic interfaces to the OS to access this DMA engine. The crypto engine provides hardware offloads to accelerate the cryptographic operations done inside the Intel ME for secure communication protocols like wireless security, HTTP security via TLS, and so on. The initial boot code for the Intel ME is located and executed from the ROM. The C-Link

interface is used for communication between the GMCH and ICH in low power states such as sleep or hibernate. Some Intel ME–specific devices in the ICH communicate with the Intel ME exclusively over C-Link, while some devices can communicate over DMI as well as C-Link (for example, the network controller).

Memory for the Intel® ME

A small portion of the main system memory is used by the Intel ME for its runtime memory needs. This separation is done using the UMA mechanism. The Intel integrated graphics controller in the GMCH also uses the same mechanism to use a portion of the main system memory for its needs. The size of this memory is 16 MB, which is less than 1 percent of the total system RAM in a computer having 2 to 3 GB of DRAM. From the perspective of the OS the Graphics UMA memory portion will appear to be a little larger than on computers that do not have the Intel ME.

Nonvolatile Storage for the Intel® ME

The Intel ME uses a NOR flash nonvolatile memory (NVM) present on the motherboard, for persistent storage of the code, configuration data, user data, and so on. This NVM is also used to store the BIOS code and other OEM specific data. It is used for other purposes by the network controllers, which we won't discuss here. The NVM is divided into specific regions, one each for the Intel ME, the BIOS, and the network controller. The NVM contains an access control descriptor at the very beginning of the NVM (at address 0) which specifies the permissions for accessing the various regions of the NVM. The ICH hardware ensures that these permissions are enforced. The controller that the ICH uses for accessing the NVM is the based on Serial Peripheral Interface (SPI). The Intel ME region of the NVM is further divided into regions for code, recovery code, internal configuration data and variable storage, event logs, and user/ISV relevant 3PDS data. Figure 7.5 shows a high level map of the NVM showing the various regions. Chapter 14 explains more details about the protection mechanisms applied to the NVM.

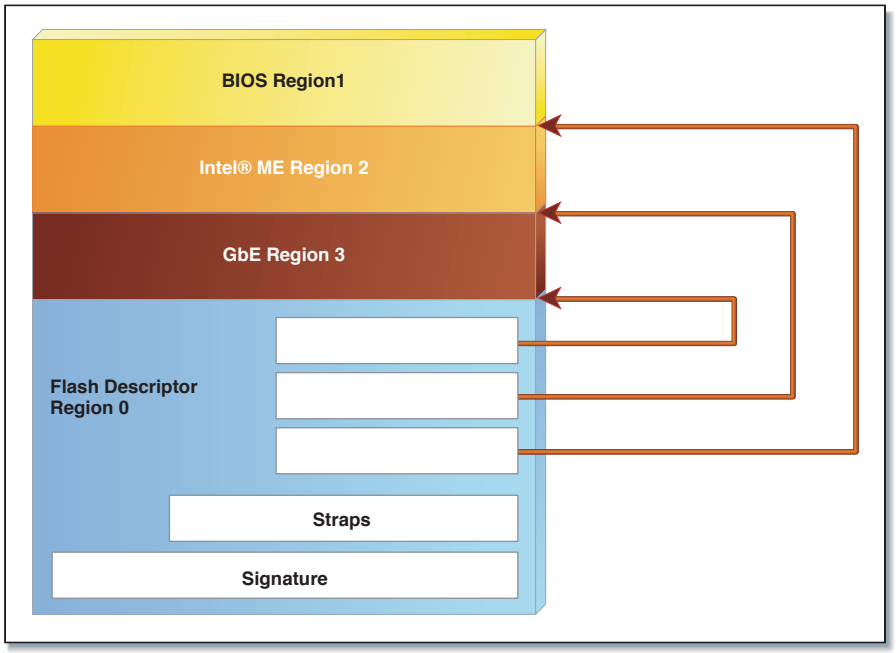

Figure 7.5 High Level Map of the Nonvolatile Memory

Network Access to Intel® ME

On desktop platforms, only the Ethernet network adapter is connected to the Intel ME. On mobile platforms, the Intel ME has access to both Ethernet and Wi-Fi network controllers and this, both when the OS is functional or not (crashed, sleeping, and so on). The Intel network controllers (Ethernet and Wi-Fi) communicate with the Intel ME using the C-Link interface. Intel ME accesses traffic from the Gigabit Ethernet controller differently from how it accesses traffic from the Wi-Fi controller. It always sends and receives traffic over the Ethernet controller directly without going via the OS. However, in the case of Wi-Fi, some design limitations require that the network controller has a single master. Therefore, when the OS is operational, the Wi-Fi traffic is routed via the Wi-Fi driver in the OS to the Intel ME. However when the OS crashes or goes to sleep, the Intel ME assumes ownership of the Wi-Fi network controller and does the communication directly.

Intel AMT provides for remote communication with computers from a management console (using the HTTP and WS-Management protocols) over these interfaces. This mechanism allows the Intel ME firmware to share a common LAN MAC, hostname, and IP address with the OS, helping to minimize the IT infrastructure cost to support functionality based on Intel AMT. The details are explained in Chapter 11.

Figure 7.6 shows the filters supported by the Ethernet network controller.

The out-of-band communications architecture supports the following filters:

- ARP: Forwards ARP packets containing a specific IP address to the host and/or the Intel ME
- DHCP: Forwards DHCP Offer and ACK packets to the host and/or the Intel ME
- IP Port Filters (HTTP and Redirection): Redirects incoming IP packets on a specific port to the Intel ME

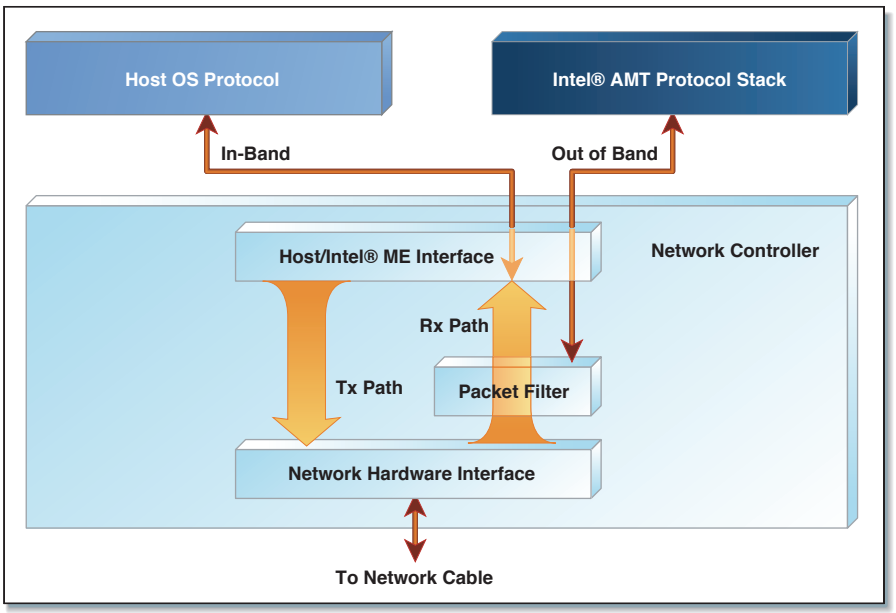

Figure 7.6 Network OOB Filters for Gigabit Ethernet

Protected Clock

The Intel ME has access to a special purpose clock on the chipset called Protected Real Time Clock (PRTC). This clock is not accessible by the OS, but only by the Intel ME. The clock is used for its time related verifications (such as certificate validations, Kerberos time stamp checks, and so on), instead of relying on the system Real Time Clock (RTC). The RTC can be changed (back-dated or moved forward) by the user or a malware in the OS, hence the Intel ME does not rely on the RTC. This PRTC is powered by the same coin battery that powers the RTC. So the PRTC maintains the time even when the computer is completely powered off.

For Experts

When Intel AMT is provisioned in Small Business mode, the protected clock is synchronized with the BIOS clock at boot time and both generally represent local time. When provisioned in Enterprise mode, the clocks are separate and it is recommended that the protected clock be set to GMT time

True Random Number Generator

The Intel ME also has access to a True Random Number Generator (TRNG), which is based on thermal noise variants. This TRNG is very helpful for assisting in cryptographic transactions such as generating random session keys, tokens, nonces, and so on. The TRNG outputs 32-bit random numbers at a time. Chapter 15 covers the TRNG in detail.

Chipset Key

The chipset has a 128-bit key for use by firmware in symmetric encryption and integrity protection operations. This key is generated during manufacturing of the chipset at the Intel manufacturing plants, by randomly blowing fuses dedicated for this purpose. The Intel ME is the only component that can access this keying material, and it provides the root of trust for several operations. No one outside the Intel ME knows the value of this key.

Firmware Architecture

This section describes the firmware architecture of the various components that constitute Intel AMT. This includes the Intel ME ROM, the kernel running on the Intel ME, common services modules (such as networking services, security services, and configuration services), and the applications in the firmware (such as 3PDS, remote control, asset inventory). A high level block diagram of the firmware is shown in Figure 7.7.

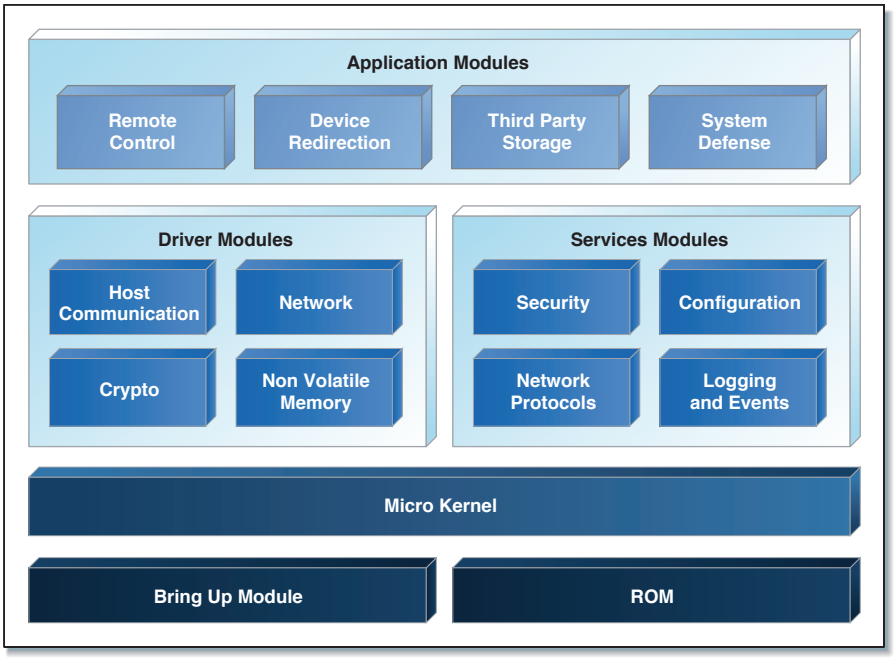

Figure 7.7 Intel® Management Engine Firmware Block Diagram

Intel® ME ROM

The Intel ME ROM is masked into the silicon of the GMCH chip. The ROM contains the reset vector (the very first set of instructions to execute after the Intel ME is reset). The ROM is only accessible by the Intel ME and not by the host or OS. Because ROM code is masked in the chip during manufacturing,

it can never be changed and hence is secure by nature. Therefore no integrity check is performed on the ROM. The size of the ROM is on the order of a few kilobytes. The main operations of the ROM include configuring the Intel ME memory areas, initializing certain critical hardware pieces, checking the integrity and signature on the firmware image on the NVM, and transferring control to the firmware image. The ROM is the root of trust of the Intel ME firmware.

Intel® ME Kernel

The Intel ME Kernel module is composed of services and drivers that provide the base functionality of the Intel ME environment. The kernel provides the basic set of services that are expected for any general purpose execution environment. These services include:

- Bootstrap and initialization
- Task and thread management
- Memory management
- Interrupt management
- Timers
- Messaging and events
- Security and cryptographic functions
- Drivers for local and network interfaces, storage, and so on
- Power management
- Interface discovery
- Firmware update

Since the Intel ME houses some very security-sensitive technologies apart from Intel AMT, such as the Intel® Trusted Platform Module (Intel TPM)2, there is a need to provide a high degree of separation and isolation right from the kernel level between the highly security-sensitive applications and others. For this reason the kernel is partitioned into privileged and non-privileged portions. The privileged portion includes the ROM, the initialization modules (such as a loader and bring-up modules), a portion of the kernel called the

2 Intel Trusted Platform Module is only available on some Intel® vPro™ technology platforms.

privileged kernel, and the Intel TPM firmware. The non-privileged portion includes the remaining portion of the kernel called the non-privileged kernel, support modules, common services modules, and other firmware applications. Firmware that executes in privileged mode has access to privileged hardware resources, such as certain memory ranges and certain hardware registers. Non-privileged firmware that attempts to access privileged resources will cause an exception or interrupt to occur. A register in the Intel ME contains the address of the code for entering and exiting out of the privileged mode. Using these addresses, the firmware code transitions between privileged mode and non-privileged mode. Figure 7.8 shows the privileged and non-privileged partitioning of the Intel ME kernel.

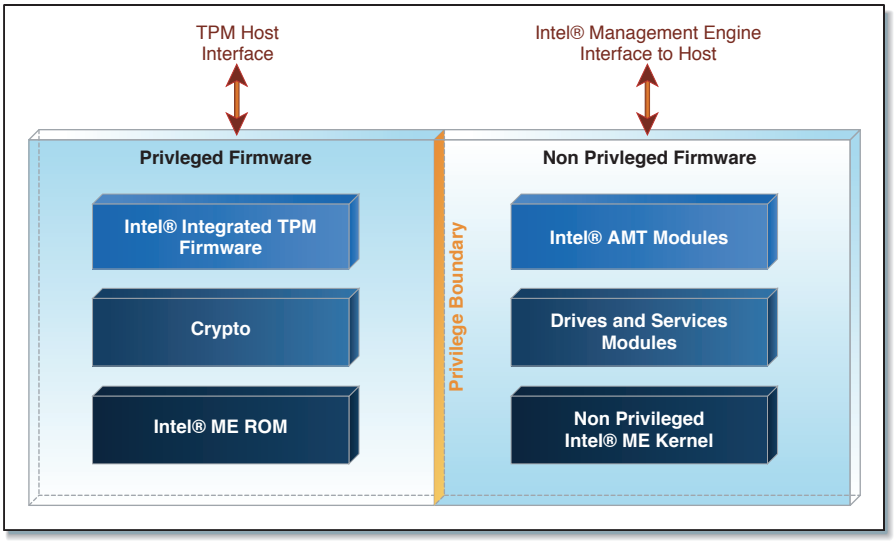

Figure 7.8 Intel® ME Kernel: Privileged and Non-Privileged Partitions

Intel® ME Common Services

The Intel ME Common Services are a set of services provided by the Intel ME firmware for different Intel ME applications such as Intel AMT applications. These services include network services, security services, and configuration services. These services are implemented on top of the Intel ME Kernel. Figure

7.9 shows the various service modules available inside the Intel ME Common Services component of the firmware.

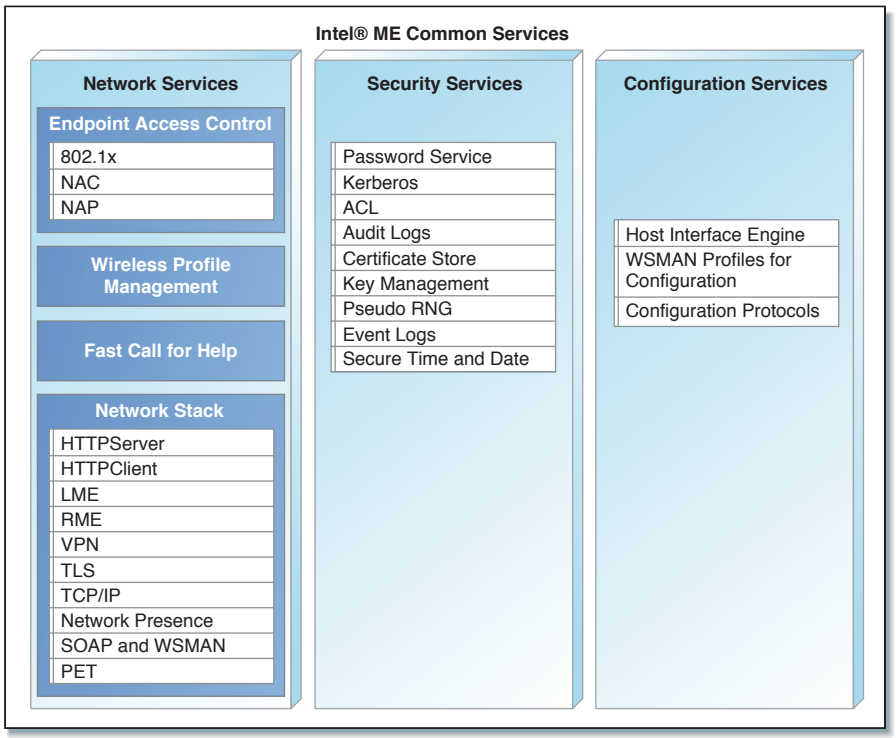

Figure 7.9 Intel® ME Common Services

Tables 7.1 through 7.3 describe briefly the role of each of the modules within the Common Services components.

Networking Services

The networking services component provides the access to various networking stacks and protocols that firmware applications use for communication, as described in Table 7.1.

Table 7.1 Networking Services

Module	Description
Network Presence	The module that establishes presence of Intel® AMT on the network by initiating DNS and DHCP transactions. DHCP helps the Intel AMT subsystem to obtain an IP address if the OS is not up, or Intel AMT and the OS are not sharing IP addresses.
TCP/IP	The TCP/IP protocol stack in firmware
TLS	The TLS protocol stack in firmware providing session encryption and integrity to keep eavesdroppers away
LME	Local Manageability Engine – this is the module that implements the firmware end of the local manageability interface with the host OS. The local manageability interface allows for communication between the local host OS and Intel AMT via a TCP/IP network connection, albeit with some constraints. We will go into this in detail in Chapter 11.
RME	Remote Manageability Engine – this is the module that implements the firmware end of the remote manageability interface. The remote manageability interface allows for communication with Intel AMT over the network from management consoles.
VPN	The VPN firmware module that allows for manageability communication through the host VPN mechanism.
Fast Call for Help (FCFH)	The FCFH firmware module that implements the FCFH capability. More details are available in Chapter 12.
HTTP Server	The HTTP server implemented in firmware to host most of the manageability applications
HTTP Client	An HTTP client module implemented in firmware for certain scenarios where Intel AMT needs to operate as a HTTP client and communicate with HTTP servers on the network.
Wireless LAN profile management	Firmware module that manages WLAN profiles and settings in Intel AMT. This allows Intel AMT to communicate using the specified wireless profiles and policies over the wireless connection.
802.1x supplicant	The 802.1x supplicant in firmware that communicates with the 802.1x enabled network infrastructure to allow Intel AMT to obtain authenticated network access.
802.1x profile management	Firmware module that manages 802.1x profiles and settings in Intel AMT.

Table 7.1 Networking Services (Continued)

Module	Description
NAC/NAP	Firmware module in firmware that communicates with the NAC/NAP enabled network infrastructure to allow Intel AMT to obtain authenticated network access. This module provides a signed posture of Intel AMT to the NAC/NAP backend infrastructure, which is evaluated for compliance with the network policies before Intel AMT can access the network.
SOAP and WSMAN	The SOAP and WSMAN protocol layer modules in firmware
PET	The PET (Platform Event Trap) alert generation module to send out PET alerts to the management console as configured by the IT administrator

Security and Utilities Services

Table 7.2 describes the security and utilities services provided that are required by most firmware applications such as authentication, access control, session encryption, auditing, and so on. It also provides some utility services such as event logs, date, and time. Most of these topics are covered in detail in Chapters 14, 15, and 16.

Table 7.2 Security and Utilities Services

Module	Description
Password Service	A password service in Intel® AMT that ensures that access to Intel AMT BIOS interfaces is allowed only after proper password authentication from the BIOS.
Kerberos	A module in the firmware that hosts a Kerberos-based service making authentication to Intel AMT integrated with the enterprise's Active Directory infrastructure in a Single Sign On manner. Using the services of this module, Intel AMT grants seamless access to privileged administrators of Intel AMT without requiring additional authentication specifically for Intel AMT.
Access Control Lists	A module that manages the permissions of authenticated administrators of the Intel AMT subsystem. Not all authenticated administrators have equal privileges in Intel AMT. This module allows for managing the permissions and enforces those permissions at the time of administrator logon and subsequent access to Intel AMT.
Security Audit Logs	A module that keeps a secure and undeniable log of the actions taken by an authenticated administrator in the Intel AMT subsystem. This module is like the video camera inside the firmware. With awareness of this capability, administrators are deterred from misusing their privileges in Intel AMT.
Certificate Store	A secure store for certificates. Certificates are used for configuration, TLS, Audit Logs, and so on. Secure storage for certificates is a necessity wherever they need to be used.
Key Management	A module that manages the various keys inside the firmware. There are various types of keys inside the Intel AMT subsystem such as TLS private key and Kerberos key.
Pseudo RNG	A pseudo–Random Number Generator module in the firmware. This takes in a true random number from the hardware based TRNG, and from there on generates pseudo-random numbers on demand. Random numbers are used by several Intel AMT and common service modules.
Event Logs	A simple logging mechanism for system events. This keeps track of various happenings in the firmware. It can be used to troubleshoot problems in the deployment or functionality of Intel AMT or for other similar purposes. This is not a secure log, and is not the same as the Security Audit Log.
Secure Time and Date	A module that manages the hardware PRTC and provides write access to the PRTC after proper authentication and authorization. It also provides read access to the PRTC to Intel AMT and common service modules.

Configuration Services

Table 7.3 describes the configuration services provided for initial configuration of Intel AMT and other firmware applications, WS-Management services, and so on.

Table 7.3 Configuration Services

Module	Description
Provisioning and Configuration	A module that manages the configuration process of Intel® AMT. This is covered in detail in Chapter 17.
WSMAN Configuration Profiles	The WSMAN (WS-Management) configuration profiles used by the WSMAN module for configuration of Intel AMT.

In this section, we have been able to cover the Intel ME common services briefly, with the intent of familiarizing the reader with the extent and breadth of the services available to firmware applications in Intel AMT. Several of these services and protocols and the mechanisms to communicate with them are explained in much greater detail in the rest of this book.

Intel® AMT Firmware Applications

The Intel AMT applications implemented in the firmware are the heart of Intel AMT. These applications implement the actual logic of the various Intel AMT features that we discussed in Chapter 5.

Software Architecture

Intel AMT software can be divided into a BIOS component, local components and remote components. The BIOS component performs certain configuration and bootstrap operations for Intel AMT. Local components consist of software drivers, services, and applications that run on the host OS of the computer that has Intel AMT. Examples are the Intel MEI driver, LMS module, UNS, and Tray Icon, as explained below. Remote components are the components that help services and applications that run on the systems that manage the Intel AMT computer over the network using management consoles. Examples are the Intel AMT SDK, Intel AMT Storage Library, and Intel AMT Redirection Library.

Intel® AMT BIOS Component

Intel AMT computers have a BIOS component integrated into the BIOS. This component is called the Intel ME BIOS Extension (Intel MEBX). It is a BIOS binary that is very similar to the other extension ROMs (also known as OPROM) commonly present in BIOS. This component provides the IT administrator the ability to configure and change the Intel ME subsystem's policies. We will come across this component at several places in this book. Some of the operations performed by the Intel MEBX component are as follows:

- Authenticate (via a password) the user before allowing any configuration changes.
- Collect the system hardware inventory information from the main BIOS and feed it into the Intel AMT subsystem so that it can be monitored by the IT administrator at a remote console. The Intel MEBX passes this information to the Intel AMT subsystem over the Intel MEI. Therefore the Intel MEBX also needs to implement an Intel MEI driver.
- Display the configuration options on the Intel MEBX screen that allows the user to view/change various Intel ME policies settings. The Intel MEBX screen is usually reached by pressing the Ctrl-P key combination at the time of system startup.

Local Software Components

Figure 7.10 shows a block diagram of the local software components of Intel AMT. The blue components depict the Intel provided components. The orange component is the host application developed by the ISV.

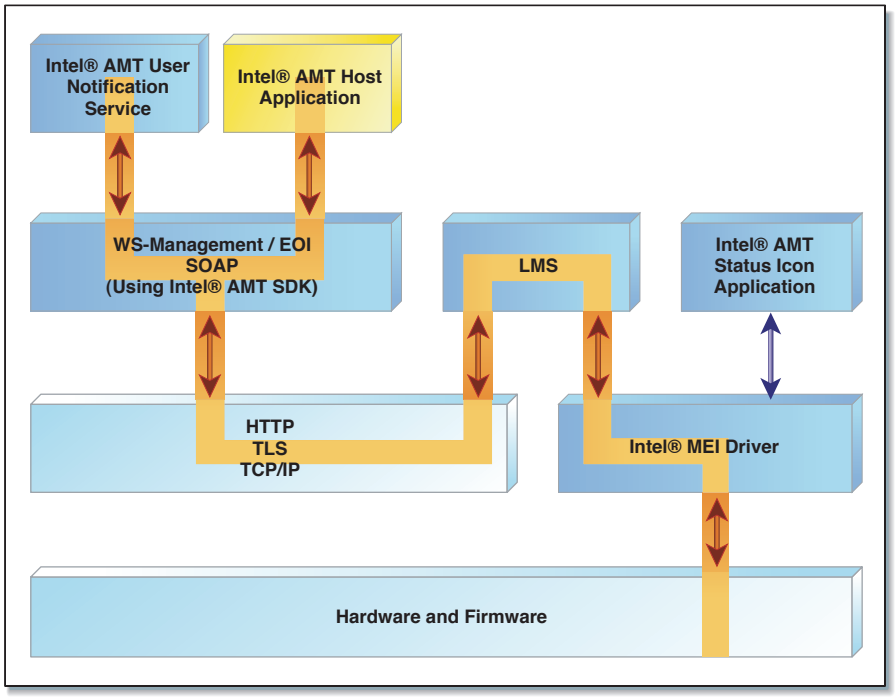

Figure 7.10 Local Software Components of Intel® AMT

The Intel Management Engine Interface (MEI) driver is the device driver for the Intel MEI that is the main communication channel between the Intel AMT firmware and the local host. In Windows, it is a WDM driver that supports Windows XP and Vista. In Linux, this is a character device driver that supports the main kernel based distributions.

The purpose of the Local Manageability Service (LMS) component is to enable Intel AMT to expose a similar interface for both local and remote applications. Basically LMS allows local applications to open a virtual "network" connection to Intel AMT firmware. From the local application point of view, it opens a network connection and uses EOI or WS-Management over SOAP. LMS redirects the network traffic over Intel MEI to the Intel AMT firmware.

Intel AMT computers have a system tray icon application that will inform the user about the existence and the enabled/disabled status of Intel AMT.

The application also points to the Intel AMT Web site so that users will be able to get detailed information about the technology and its capabilities. This is covered in more detail in Chapter 16.

User notification service (UNS) module in the Intel AMT firmware enables Intel AMT firmware to send events to the local user. The role of this software component it to listen to such events and record them in the operating system event log.

The yellow highlighted line in Figure 7.10 shows the path of communication from the Intel AMT host application to the Intel AMT subsystem in the firmware/hardware. The same path is also taken by the Intel AMT User Notification Service to communicate with the Intel AMT subsystem in the firmware/hardware.

Remote Software Components

Figure 7.11 shows a block diagram of the remote software components of Intel AMT. The blue components depict the Intel provided components. The orange component is the Management Console application developed by the ISV.

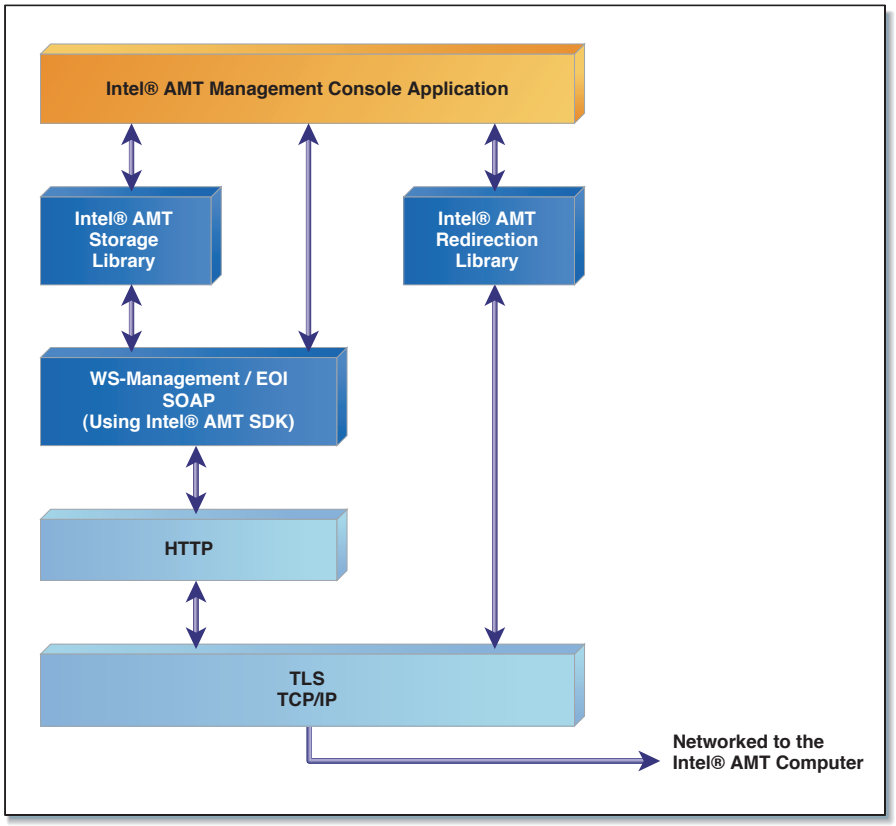

Figure 7.11 Remote Software Components of Intel® AMT

The purpose of Intel AMT SDK is to enable software developers to easily develop applications that communicate with Intel AMT and bundle Intel AMT features into their existing management consoles and other applications.

The SDK is comprised of the documentation, WSDL files for EOI/SOAP interface, and MOF files for WS-Management interface, and code samples for a majority of Intel AMT features (both EOI and WS-Management). The SDK is designed for both Windows and Linux environments, and primarily for C++ and C#, but any language that supports SOAP/WS-Management can be used (such as Java, for which a sample is included with the SDK).

The purpose of the Intel AMT Storage Library is to provide a clean programmatic interface to applications that use the third party data store feature and hide the underlying details of EOI/SOAP commands. There are Windows and Linux versions of the library. The interface to applications is in the form of a static C/C++ library.

The purpose of Intel AMT Redirection Library is to provide a clean programmatic interface to applications that use Serial over LAN or IDE Redirection features and hide the underlying details of the redirection protocols. There are Windows and Linux versions of the library. As with the Intel AMT Storage Library, the interface to applications is in the form of a static C/C++ library.

Power Management States of Intel® AMT

The Intel AMT subsystem has the capability to operate in all states when the main operating system is running and when it is sleeping (S3 state), hibernating (S4 state), or shut down (S5 state). These states were described in Chapter 5, but here is the recap. The S0 state is when the computer is working, that is, the operating system and applications are running. The S1 and S2 states are not commonly implemented on modern systems, so we will skip them. S3 is the standby state. In this state, most of the computer is powered off. Only the main memory (RAM) is powered on and in self-refresh mode, to retain its data. The resume time from S3 to S0 is therefore fairly quick, since the memory already contains the state of the operating system and applications, and execution begins in a matter of a few seconds. S4 is the hibernate state. All the components in the computer are powered off, and the operating system and application state is stored on the hard disk. Resume from S4 to S0 is relatively much longer since the state has to be restored from the disk to RAM. S5 is the state when the computer is fully powered off and the operating system has shut down, maintaining no state.

Intel AMT provides configuration options to the user (usually via the MEBx configuration screens) to let the user choose the power management policies for Intel AMT. For example, the user may choose not to keep Intel AMT operational when the computer is in S3, S4, or S5. Figure 7.12 shows a sample Intel MEBX screen where various power management options are displayed.

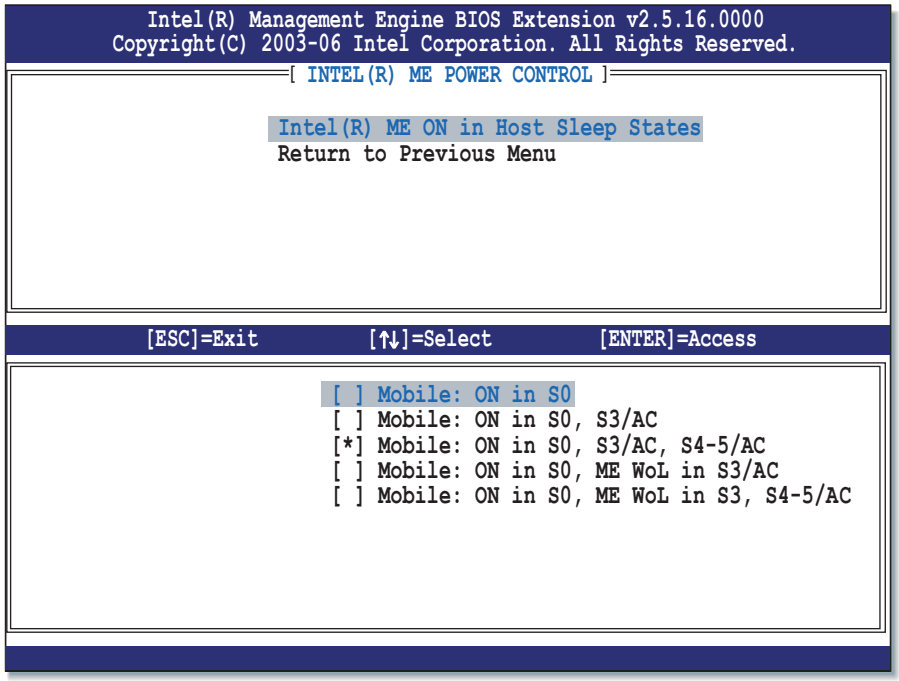

Figure 7.12 Intel® AMT Power Management Options in the Intel® MEBX

The Intel AMT subsystem defines its own power states, called *M-States*. These states are M0, M1, and M-Off. The Intel AMT subsystem is in the M0 state when the overall system is in S0. All the components of Intel AMT are powered on and operational. Figure 7.13 shows the computer in M0 state. The Intel AMT subsystem can be either in M1 or M-Off when the overall system is in S3/S4/S5, depending upon the policy configured in Intel MEBX. Intel AMT functionality is available in S3/S4/S5 states only when Intel AMT is in the M1 state, not M-Off. In the M1 state, all the components that take part in making Intel AMT work are powered on (for example, the Intel ME, some portion of RAM, NVM, networking components, and so on). The rest of the computer is off. Figure 7.14 shows the computer in M1 state. In the M-Off state, all the components are powered off. Figure 7.15 shows the computer in M-Off state.

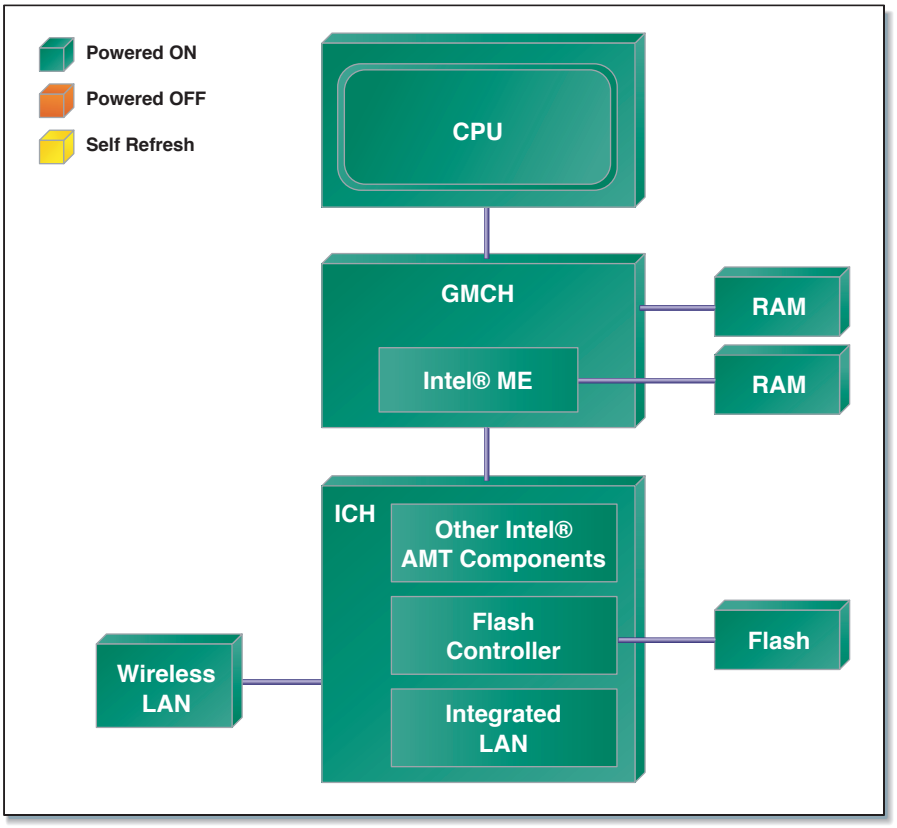

Figure 7.13 An Intel® AMT Computer in S0 and M0 State

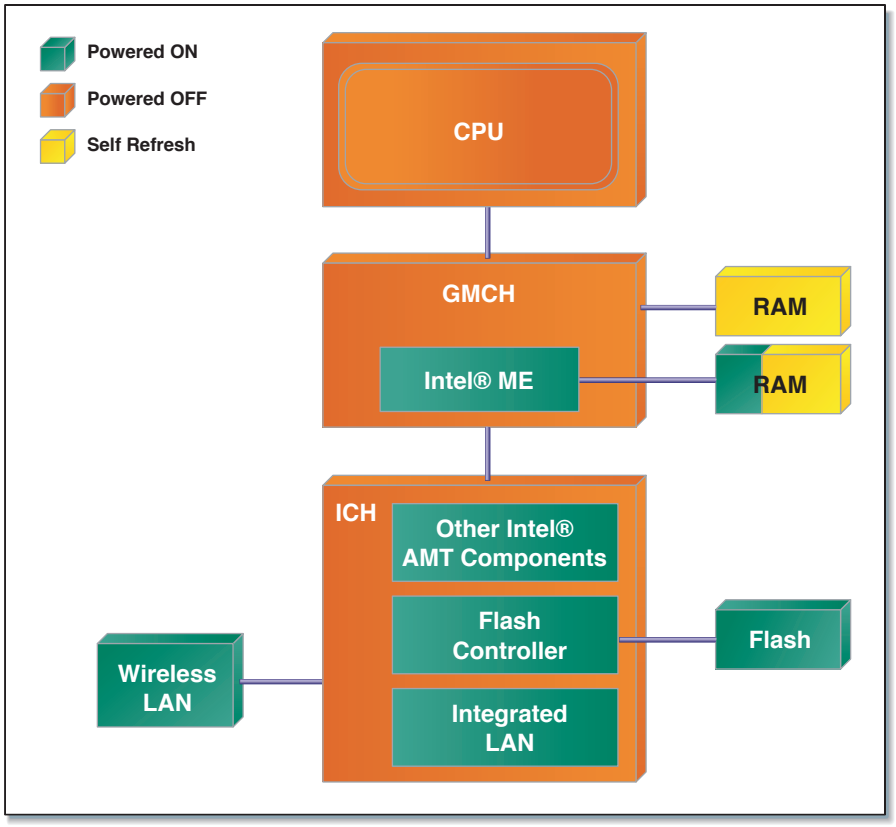

Figure 7.14 An Intel® AMT Computer in S3 and M1 State

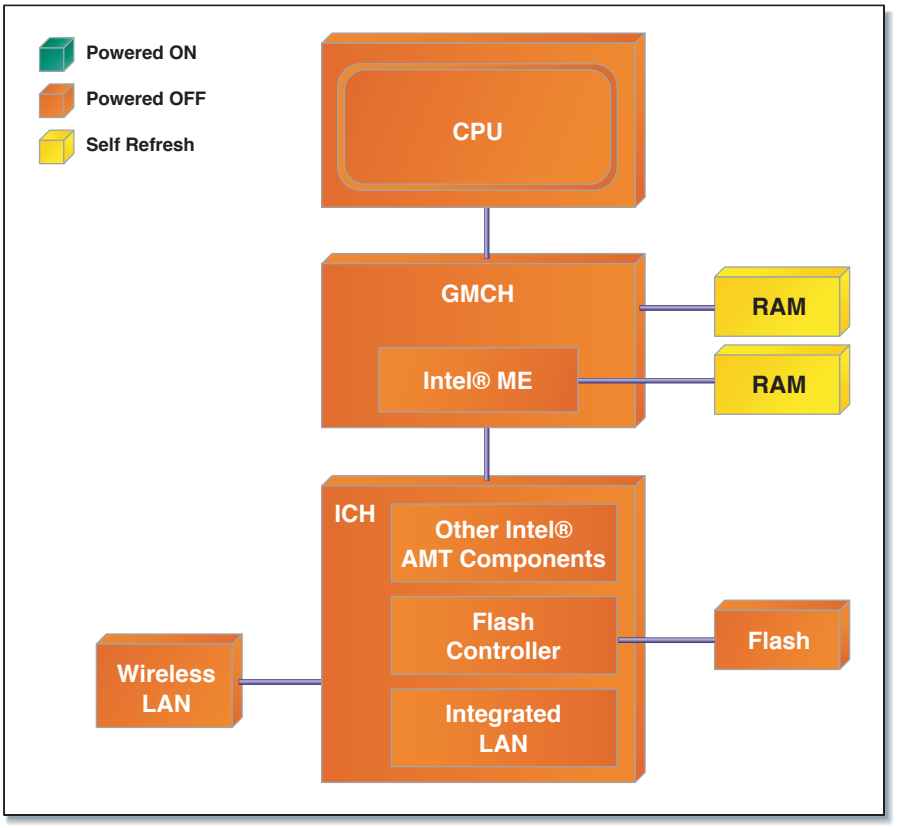

Figure 7.15 An Intel® AMT Computer in S3 and M-Off State

Summary

In this chapter we briefly covered all the important components that make up Intel AMT. This included the hardware components such as the Intel ME inside the chipset, the nonvolatile storage, memory, network controller, and so on. Then we described the firmware components such as the Intel ME kernel, common services, and firmware applications. We also discussed some details of the software components that reside on the host OS of the computer that has Intel AMT, as well as components that reside on management consoles on computers remotely located over the network. Finally we discussed the power management operations within Intel AMT and what enables Intel AMT to be operational even when the computer is in a sleep, hibernated or off state. In the next few chapters we go into more depth on the various aspects and features of Intel AMT.