Reprogrammable Flash BIOS Design Using AMD's Am29F010

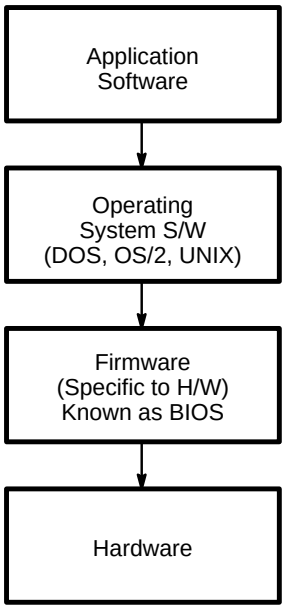
Application Note
by Kumar Prabhat

Advanced Micro Devices

This application note describes the general overview and various system level issues for a reprogrammable Flash BIOS design. Any system designer, whether notebook or desktop system, will benefit from this discussion. This application note also describes the AMD Am29F010 5.0 V-only, sector erase part and why it is an ideal choice for a reprogrammable BIOS design.

INTRODUCTION TO BIOS

Every individual computer system consists of three basic blocks—Hardware, Hardware Specific Firmware commonly named BIOS and System Software.



17078B-1

BIOS is a hardware dependent software normally stored in an EPROM, which provides an interface between specific hardware and system software. It interfaces with various hardware components like core logic chipset, graphics controller, the keyboard and disk drive. Any application software and operating system software (i.e., DOS, OS/2 and UNIX) runs above it and uses various BIOS procedures. IBM has defined various BIOS procedures to control specific peripheral functions. Some of them are mentioned as follows:

INT 13H	Disk Drive Control
INT 17H	Printer Control
INT 10H	VGA Control
INT 16H	Keyboard Control
INT 14H	Serial Communication Control

To use the BIOS procedure you load the parameters required by the procedure and then execute the $\overline{\text{INT}}$ instruction that accesses that procedure. For example, you can use the BIOS INT 10H procedure for 15 different functions related to the CRT display. Some of these functions are: set display mode, set color palette, write dot and write character to screen. You specify the function you want by loading the number for that function in the AH register before executing the INT10H instruction.

All of the 80X86 procedures boot up from the BIOS located at the very top of the 1 Mbyte memory map which is F0000H – FFFFFH. In addition 64K bytes of space lying in the address range E0000H – EFFFFH is provided for any BIOS extension.

The BIOS also contains various initialization routines to initialize system components like the serial port, DMA Controller and Interrupt Controller. During power-on it does the Power On Self Test (POST) routines. It also checks for basic system RAM functionality. If the system passes the RAM functionality, the BIOS will copy itself to the top 64K byte area of 1 Mbyte main memory. This is known as shadowing BIOS which improves the system performance since the BIOS code is run at DRAM speeds instead of slower ROM speeds.

WHY THE NEED FOR REPROGRAMMABLE BIOS

The concept of a personal computer is changing rapidly as technology progresses. Yesterday's high-end systems are becoming today's standard platforms and new technologies have brought enhanced system capabilities into the user's hands. The fastest growing segments of the computer market are in Notebooks and other portable PCs. Increasing demands for sophisticated hardware and intelligent power saving algorithms have increased the complexity of BIOS code. In the desktop computers area, the enhanced support of Ethernet/SCSI Controller on the PC motherboard also increases the need for BIOS modifications to support sophisticated peripherals. On the high-end, EISA systems need to store hardware specific configurations which are tra-

Publication# **17078** Rev. **B** Amendment **/0**
Issue Date: **March 1994**

ditionally stored in battery-backed-up SRAMs. Today's PC BIOS is no longer a standard product except for the basic 64K byte compatibility portion. The remaining 64K byte area has a significant potential to change with the addition of Power Management software and new setup utilities. To summarize, we see the potential change in BIOS code at every stage of manufacturing like design, debugging, testing and production. Code modifications with EPROMs do not provide a cost effective and timely solution since a UV Eraser and a separate EPROM programmer are required.

Flash Memories offer a superior solution for this kind of application. Code prototyping time is significantly reduced because Flash Memories can be updated with new code in a manner of seconds while still in the system. Board level diagnostics, final system test, and customer specific configuration code can all be down-loaded into the Flash memory electrically on the assembly line. Devices may be soldered directly to the system board. This reduces the cost associated with the BIOS socket and also eliminates the need to disassemble the system and replace socketed devices. Moreover, it will remove the prohibitive costs associated with a field service call. When updates to system code or system reconfiguration is necessary, these costly service calls may be replaced with remote updates or by distributing floppy disks with new data.

AMD FLASH MEMORY

This section provides a brief overview of AMD's 5.0 V-only Flash memory and in particular, AMD's Am29F010 5 V-only, Sector Erase part.

AMD's 5.0 Volt-only Flash Memory Technology

This section illustrates the fundamentals of AMD's Flash Memory Technology. AMD's Flash memory technology is very similar to that of our UV EPROM. The main difference is associated with the Fowler-Nordheim tunneling erase mechanism.

During program operations AMD's Flash memories transfer and store charge on a floating gate in a manner similar to EPROM. This provides data retention that is equivalent to that of EPROM devices. The device is programmed by raising the control gate and drain terminal to a high voltage. The source terminal is grounded. The voltage potential across the channel attracts channel electrons from the source area towards the drain. At the drain region, some of these channel electrons become "hot" electrons and are swept up through the thin oxide where they are trapped on the floating gate. The electrons stored on the floating gate create an electric field which turns off the memory transistor and represents a logic zero.

AMD's 5.0 V-only, Flash memory uses Negative Gate Erase Technology for erase operations in order to minimize the current drawn from the erase change pump. AMD's Negative Gate Erase technique actually provides the same electric fields to the Flash memory cell and uses the same erase mechanism as its 12.0 V Flash devices. A negative voltage of –10.5 V is applied to the control gate while the source terminal is at 5.0 V supplied by the system V_{CC} supply. Negative Gate Erase is used in order to reduce the current drawn from the erase charge pump. The band-to-band tunneling current (10–20 mA peak) comes directly from the system V_{CC} supply through the Array around terminal. This is the most efficient way to use an existing system V_{CC} supply. The current required from the Negative Gate Erase charge pump is less than 10 μA at –10.5 V. This significantly reduces the internal power consumption in relation to conventional 12.0 V approaches.

The Am29F010

This section describes the various features of the Am29F010.

General Description

The Am29F010 is a 1 Mbit 5.0 V-only "Flash" electrically erasable, electrically programmable read only memory organized as 128K x 8 bits. It is a 32-pin device which allows upgrades to 4 Mbit densities. The device has uniform sector architecture with 100,000 minimum endurance cycles per sector.

The Am29F010 is entirely pin and software compatible with the 5.0 V-only JEDEC standard. Commands are written to the command register using standard microprocessor write timings. Register contents serve as input to an internal state-machine which controls the erase and programming circuitry. Write cycles also internally latch addresses and data needed for the programming and erase operations. With the appropriate command sequence written to the register, standard microprocessor read timings output array data, access the auto-select codes or output data for erase and program verification. Reading data out of the device is similar to reading from 12 V Flash or EPROM devices.

Embedded Algorithms

The Am29F010 is programmed and erased using Embedded Algorithms, which completely automates the program and erase operation. The Am29F010 is programmed by executing the Program Command sequence. The Embedded Programming™ Algorithm automatically times the programming pulse width and verifies the proper cell margin. Chip erase is done by executing the erase command sequence. The Embedded Erase™ Algorithm automatically verifies if the entire array is programmed and if it is not, the algorithm will automatically pre-program it before beginning electrical erase.

The Am29F010 features $\overline{\text{Data}}$ Polling and Toggle Bit functions as a method to indicate to the host system

when the Embedded Algorithms are in progress or completed. During the Embedded Program Algorithm an attempt to read the device will produce compliment data of the data last written to DQ7. Upon completion of the Embedded Program Algorithm, an attempt to read the device will produce the true data last written to DQ7. During the Embedded Erase Algorithm, DQ7 will be "0" until the erase operation is completed. Upon completion of Embedded Erase Algorithm data at DQ7 will be "1". The device also features a "Toggle Bit" as another method to indicate to the host system when the Embedded Algorithms are in progress or completed. During an Embedded Program or Erase Operation, successive attempts to read data from the device will result in DQ6 toggling between one and zero. Once the Embedded Program or Erase Operation is completed, DQ6 will stop toggling and valid data will be read. Upon completion of the Embedded Algorithm the device returns to the read mode. The Am29F010 will also indicate through DQ5, if the program or erase time has exceeded the specified limits. Under these conditions DQ5 will produce "1" which will indicate that the program or erase cycle was not successfully completed. Then DQ4 will indicate which algorithm exceeded the limits. A "0" in DQ4 indicates a programming failure, a "1" indicates an erase failure.

The automatic nature of the Embedded Algorithms provide various benefits over standard algorithms. Embedded Algorithms increase the system level performance significantly by reducing the CPU's overhead associated with the repetitive nature of standard algorithmic commands. This frees up the CPU to execute other system level tasks.

Sector Based Architecture

The Am29F010 also features a sector erase architecture. The whole memory content of the device is divided into eight sectors of equal size. The sector architecture allows for 16K byte segments of memory to be erased and reprogrammed without affecting other sectors. The device also supports hardware sector protection. This feature will disable both program and erase operations in any number of sectors (0 through 7). Please refer to the data sheet for more details on sector protection.

Sector erase requires a six bus cycle command similar to standard E^2PROMs. There are two unlock write cycles followed by writing the "set-up" command. Two more unlock write cycles are then followed by the sector erase command. On this sixth bus cycle, sector address defined by higher address lines A16, A15 and A14 is latched on the falling edge of $\overline{WE}$, while the sector erase command (30h) is latched on the rising edge of $\overline{WE}$. Multiple sectors may be erased concurrently by writing the six bus cycle command as described above followed by a sector erase command with other sector addresses. A time-out of 100 µs from the rising edge of the $\overline{WE}$ pulse of the last sector erase command will initiate the sector erase operation. If another sector erase command is written within the 100 µs time-out window the timer is reset. Any command other than sector erase within the time-out window will reset the device to the read mode, ignoring the previous command string. The device will indicate this time-out through the DQ3 pin. If DQ3 is high the internally controlled erase cycle has begun, and attempts to write additional commands to the device will be ignored until the erase operation is completed as indicated by the $\overline{Data}$ Polling or Toggle Bit. If DQ3 is low, the sector erase timer window is still open and the device will accept additional sector erase commands. The system software should check the status of DQ3 prior to and following each sector erase command.

Am29F010—An Ideal Choice

- The Am29F010 has an access time of as fast as 45 ns which will provide true 0 wait state performance in very high speed designs without downloading the code to the Shadow RAM.
- The device incorporates Embedded Algorithms which reduces software overhead for system designer and it also increases system performance since CPU will be free to do other tasks during reprogramming operations.
- The device provides a minimum of 100,000 write endurance cycles per sector. This kind of high endurance is especially important in the emerging markets of embedded flash disks and removable cards.
- Since the Am29F010 is a 5.0 V-only device, it eliminates the need for DC to DC converter circuitry to translate the system level voltage level from 5.0 V to 12.0 V for write and erase operations. This also simplifies the hardware design, results in reduced board space and reduces the system cost by approximately $2.00 to $4.00.
- As the power consumption is proportional to the square of operation voltage, 5.0 V operation reduces the power consumption significantly during programming and erase operation.
- Sector erase architecture is another added advantage which is valued by many system designers. The Am29F010 provides a system designer eight sectors to use in their designs in order to add more functions to the system. It also eases the design and debugging process by allowing a system designer to erase a single sector during code modification. This brings the program modularity to the system.

- The device also incorporates several features to prevent inadvertent writing of the part.
  - — During the power-up and power-down, a write cycle is locked out for V_{CC} less than 3.2 V (typically 3.7 V). Under these conditions, the command register is disabled and all internal program/erase circuits are disabled.
  - — The device ignores the noise pulses or glitches of less than 5 ns (typical) on $\overline{OE}$, $\overline{CE}$ or $\overline{WE}$ and will not initiate a write cycle.
  - — During power-up of the device even with $\overline{WE}$ = $\overline{CE}$ = V_{IL} and $\overline{OE}$ = V_{IH}, the device will not accept commands on the rising edge of $\overline{WE}$. The internal state machine is automatically reset to the read mode.

Packaging Details

AMD's Am29F010 is being offered in three standard 32-pin packages: Plastic Dual In-Line Package (PDIP), Plastic Leaded Chip Carrier (PLCC), Leadless Chip Carrier (LCC) and Thin Small Outline Package (TSOP). See Figures 1 and 2 for pin-out details.

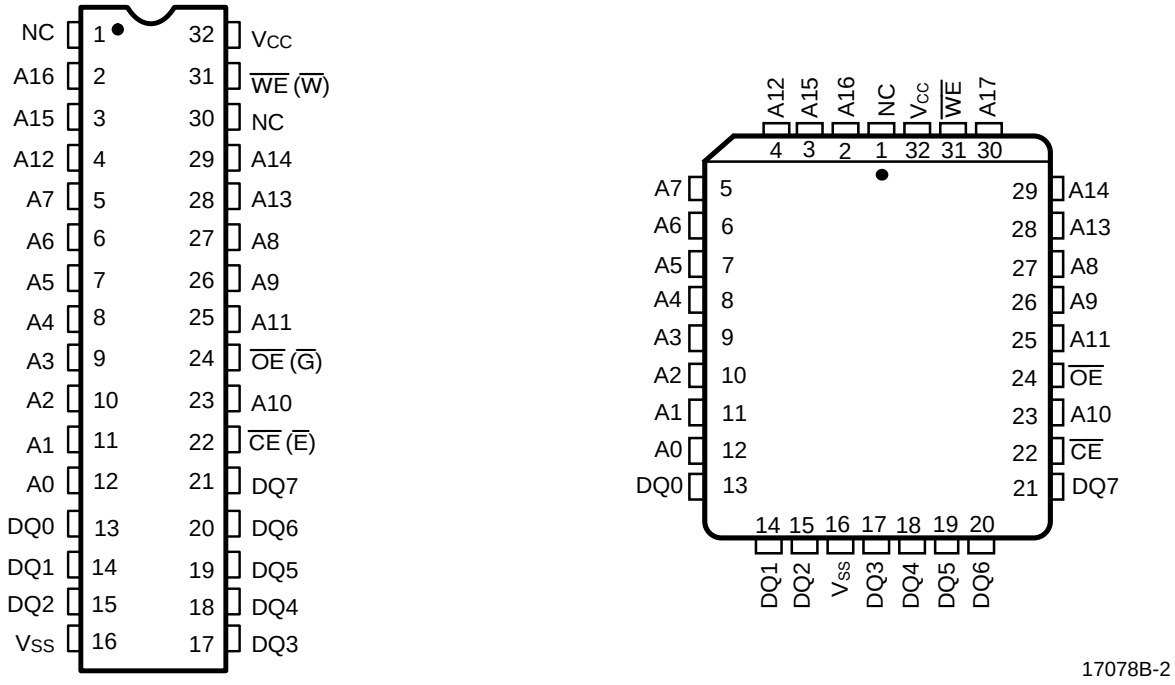


Figure 1. Am29F010 DIP, PLCC and LCC Pin-Out

29F010 Standard Pinout

29F010 Reverse Pinout

17078B-3

Figure 2. Am29F010 TSOP Pin-Out

Thin Small Outline Package

The TSOP is the industry's leading edge plastic, surface mountable memory package. System requirements for higher density and smaller, high density memory arrays are driving this packaging trend. It is becoming the standard choice for hand-held equipment and palmtop/ laptop computers as well as memory cards. This package comes in standard and reverse 32-pin options and is available in the 8 mm x 20 mm x 1.27 mm package outline. In addition to the TSOP's low height profile, maximum board space is achieved with the dual-in-line and standard/reverse pinouts. Board layers can be reduced because traces are routed under the two sides of the package that do not have leads. This allows packages to be mounted side-by-side and end-to-end. All pins except chip enable pins can be connected in parallel. This is accomplished by using standard and reverse pin-out packages in an alternating sequence as shown in Figure 3.

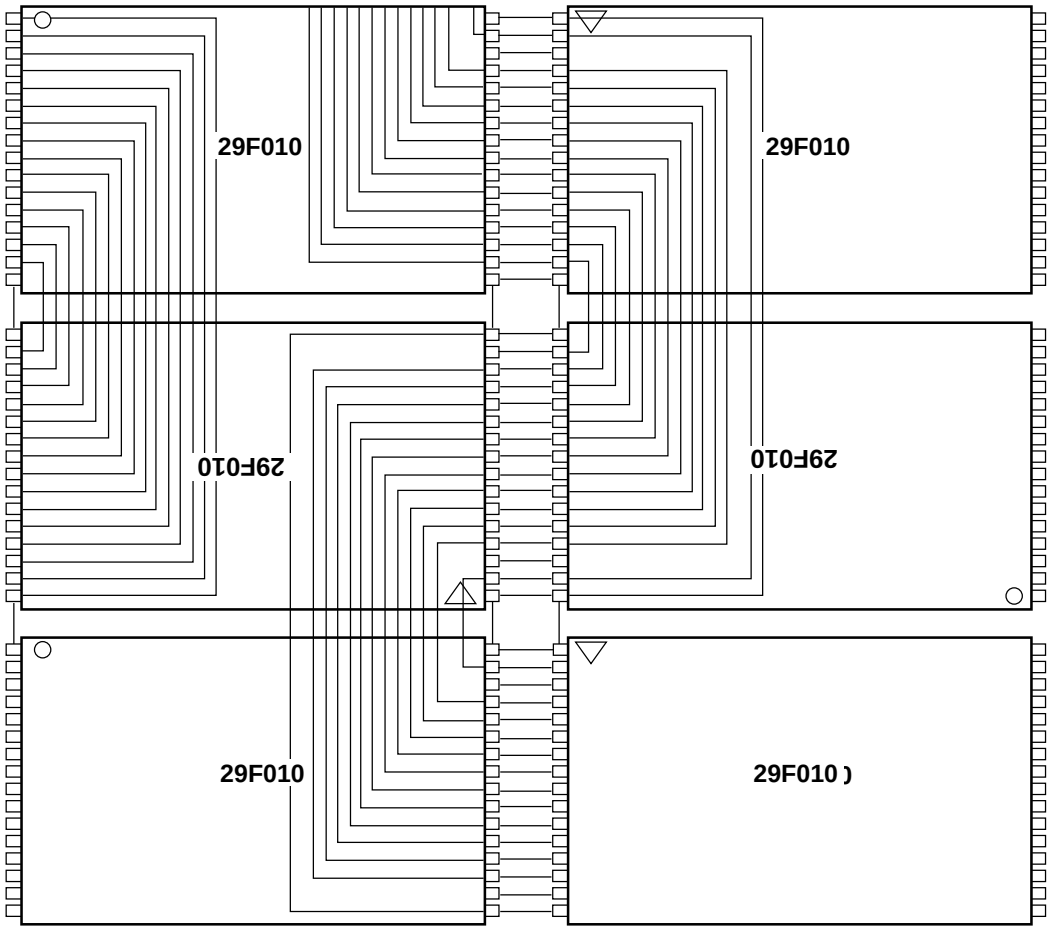


Figure 3. Optimum Board Layout with TSOP

SYSTEM LEVEL DESIGN ISSUES

This section describes various system level design considerations for the support of reprogrammable BIOS.

Hardware Design Consideration

This section describes the modifications required in the standard PC-AT motherboard to support the Flash BIOS. Below is the block level diagram of a PC-AT motherboard supporting reprogrammable Flash BIOS.

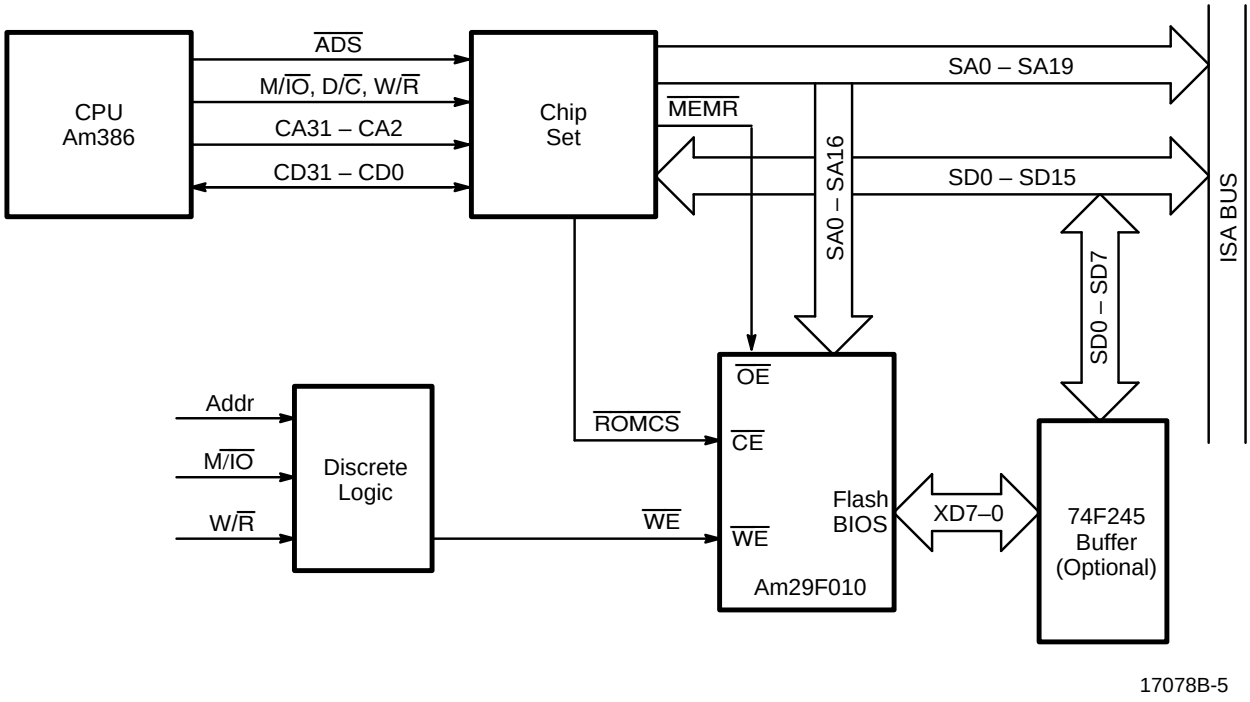


Figure 4. PC-AT Motherboard with Flash BIOS

Looking at the above block diagram we come across two main considerations for supporting the Flash based reprogrammable BIOS:

- All write accesses to EPROM address space are directed to the ISA bus and effectively discarded. Since standard PC chip sets do not generate MEMWR with ROMCS. WE for Flash EPROM must be generated externally.
- Standard PC motherboards do not support writes to the BIOS EPROM. If the chip set data buffer works only in one direction, a data buffer is required that works in both read and write directions in order to support Flash BIOS.

WE generation for Flash EPROM can be generated by using simple discrete circuitry to decode the BIOS addresses range (E0000H – FFFFFH), M/IO, and W/R. Figure 5 shows the generation of WE signal.

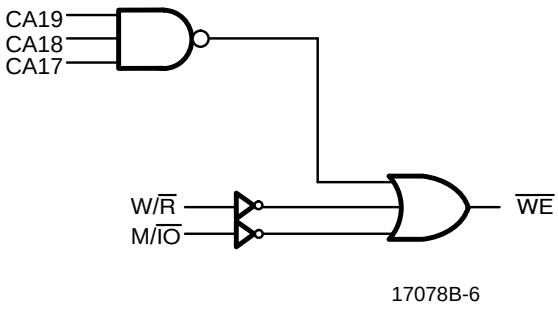


Figure 5. WE Generation Circuit

Considerations for In-System Programming

In traditional PC motherboard design, the EPROM containing BIOS, is normally socketed and disassembled from the board. Flash EPROMs eliminate the need to disassemble the system and replace the socketed device in order to update System BIOS. Flash devices may be soldered directly to a printed circuit board since they support in-system programming.

Before soldering the Flash memory on the board, the manufacturer may initially program the boot code and any other codes which have to be protected. Boot codes may be protected using external programming equipment or by AMD. To activate these modes, the programming equipment must force 12 V on the device. The particular sector will be selected using high order address lines A14, A15 and A16. Once the device is soldered into the board, the rest of the programming is done by using the local CPU. The Boot code is not meant to be changed once it is protected. However, its content may be altered by using the Sector Unprotect feature of the device and then reprogram the device with a new code. Please refer to the data sheet for more details on sector protect and unprotect.

Software Design Consideration

Let us take the BIOS design example using the AMD's Flash device Am29F010 as shown in Figure 6. BIOS Code has been divided into various modules like Boot Code, System BIOS, VGA BIOS, Power Management Code etc., and each sector may be used for individual modules.

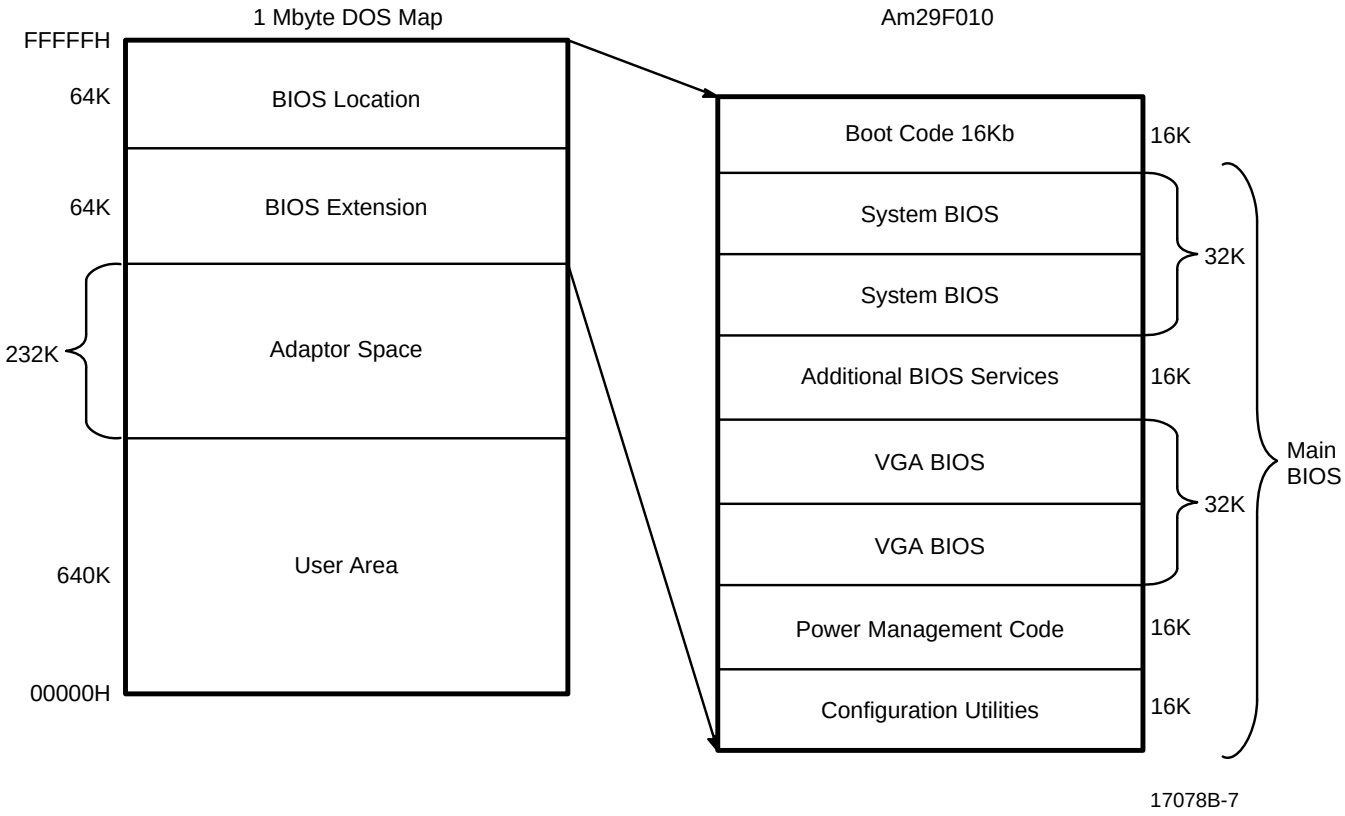


Figure 6. BIOS Design with Am29F010

Please note that the only difference between the standard BIOS and Flash BIOS is the addition of Boot Code located in a separate sector. The Boot Code resides in the system address FC000h – FFFFFh and contains the minimum code needed to boot up the system so that other blocks can be reprogrammed if required. Boot Code consists of:

- 16-byte jump vector
- BIOS check sum routine
- Recovery code

The Recovery Code contains various initialization routines and basic minimum routines for system start-up.

  - — System timer
  - — DMA/Interrupt function
  - — Keyboard
  - — Floppy drives

- During power on Boot Code takes control of the system
  - — It uses the BIOS checksum routine to check for valid main BIOS.
- If the main BIOS is valid, system RAM is checked and the main BIOS code is copied into the system DRAM memory and continue the boot operation. This feature is known as shadow memory and is used by most PC designs.
- If BIOS checksum determines an invalid BIOS, the system gives control to the recovery code for boot operation. The recovery code initializes the system RAM and floppy drive. Using basic minimum routines, it boots up the system from floppy drive and displays the message to insert the BIOS update diskette.
- The BIOS update diskette will contain:
  - — Reprogramming utility
  - — BIOS code
- The reprogramming utility is loaded into system RAM and used to reprogram the main BIOS from the diskette.

The above procedure may be also used to modify the main BIOS code.