OBJECTIVES: At the end of this section, the student will be able to do the following:

⊙ Describe the organization of PC Base Memory (0-1MB).
⊙ Discuss the contents of ROM in terms of the BIOS & POST.
⊙ Describe the organization of a typical Flash Memory Chip.
⊙ Explain Recovery BIOS procedures.
⊙ Discuss the Pinout of a typical Flash Memory Chip.
⊙ Explain how to use an O’Scope to check Flash Data Pins.
# System Memory Map

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>DESCRIPTION</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000 09FFFF</td>
<td>640K SYSTEM MEMORY</td>
<td>REAL MODE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ON BOARD DRAM</td>
</tr>
<tr>
<td>0A0000 0BFFFF</td>
<td>128K VIDEO RAM</td>
<td>DISPLAY BUFFER</td>
</tr>
<tr>
<td>0C0000 0DFFFF</td>
<td>128K I/O EXPANSION ROM</td>
<td>RESERVED FOR ROM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ON I/O ADAPTERS</td>
</tr>
<tr>
<td>0E0000 0EFFFF</td>
<td>64K ROM OR DRAM*</td>
<td>BIOS EXTENSION</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OR RAM</td>
</tr>
<tr>
<td>0F0000 0FFFFFF</td>
<td>64K ROM OR DRAM*</td>
<td>ROM BIOS OR RAM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RAM</td>
</tr>
<tr>
<td>100000 10FFEF</td>
<td>HIGH MEMORY AREA</td>
<td>ON BOARD DRAM</td>
</tr>
<tr>
<td>10FFF0 FFFE0000</td>
<td>EXTENDED SYSTEM MEMORY</td>
<td>ON BOARD DRAM</td>
</tr>
<tr>
<td>FFFE0000 FFFFFFFF</td>
<td>128K ROM OR DRAM*</td>
<td>RESERVED FOR SYSTEM</td>
</tr>
</tbody>
</table>

* Reserved for system use.
RESERVED MEMORY

- The area from 0E0000H to 0FFFFFFH is reserved for BIOS.
  - Typically only the upper 64K is used (F0000-FFFFF).
- The upper 64K bytes contains the POST and BIOS code (and a character set for MDA & CGA video).
- The upper 128K bytes can be shadowed for better performance.
  - ROM memory-access times (~120ns) are usually longer than those for main system DRAM (~60ns).
  - ROM can usually only provide 8 or 16 bits at a time instead of 32 or 64 bits at a time from main system DRAM.
ROM BIOS OVERVIEW

- **Read Only Memory** retains the stored code and data when the computer is turned off.

- ROM usually contains the following:
  - **POST** (Power-On-Self-Test)
    - POST detects, checks, & initializes installed components on the system board.
  - **BIOS** (Basic Input Output System)
    - Isolates DOS and applications from the low-level hardware.
    - Low level interface to devices.
      - Low level I/O drivers and services.
    - The major advantage of using BIOS is that we do not have to be knowledgeable about the physical interface.
ROM BIOS OVERVIEW

- When the system is reset, the processor fetches and executes the first instruction from address 0FFFF0H (0FFFFFFFF0H for 386 & later CPUs) in the ROM.
  - The first instruction is typically a FAR jump to the actual POST (POWER-ON-SELF-TEST) program.
    - FAR JUMP TO F000:E05B (FE05Bp)
- The CPU Executes the Code fetched from the ROM and the POST is executed.
  - POST detects, checks, & initializes installed components on the system board.
  - **POST** writes a **CODE** to I/O **Port 80** at the **start** of each new POST test.
ROM BIOS OVERVIEW

ROM BIOS IS USED FOR:

- **POST** -- POWER-ON-SELF-TEST
- **INITIALIZING THE SYSTEM**
  - MEMORY, DMA, PICs, PIT, KYBD CONTROLLER
  - INTERRUPT VECTORS
- Configuring the System According to:
  - Built-in Set-up Program
  - Battery Backed CMOS Configuration RAM
- Disk Boot Sector Loading
  - Floppy Disk: Track 0, Sector 1, Head 0)
- Storing Characters for certain Graphics Modes
FLASH MEMORY CHIP Organization
Flash EPROMs

- Flash EPROMs are electrically erasable PROMs.
- What distinguishes Flash from EEPROMs is that the flash memory array can be erased electrically in bulk.
- Modern FLASH Memory chips have access times of ~120 nSec.
- Flash EPROMs are suited for applications that require periodic updating of code or data where EEPROMs might have been specified.
  - In many PC products, Flash Memory is commonly used on the System Board to hold the BIOS code & data.
    - The System or Boot ROM.
FLASH Memory Chips

- A 1 Mbit flash memory can be programmed in place (w/o removing the chip) within about 2 seconds.
  - Flash memories can be erased & programmed many times (~10,000).
  - The read, program, & erase processes are controlled by means of 2-byte instructions.

- In an erasure process the flash control applies an erasure pulse to the whole memory cell array so that all memory cells are erased.
  - The erase time for the whole flash memory is about 1 s.
  - In contrast, EEPROM’s are erased and programmed 1 byte at a time.
Intel 28F001BXT 1Mbit FLASH

- This FLASH is organized as **128K x 8 (128 KByte)**.
  - The FLASH device resides in system memory in **two 64KB Segments** at E0000H & F0000H
- The FLASH device can be mapped two different ways depending on the mode of operation
  - In **Normal Mode**, address line **A16 is inverted**, setting the E000H and F000H segments so that the BIOS is organized as shown.

<table>
<thead>
<tr>
<th>System Address</th>
<th>FLASH Memory Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>F0000H - FFFFFH</td>
<td>64 KB Main BIOS</td>
</tr>
<tr>
<td>EE000H - EFFFFH</td>
<td>8 KB Boot Block (Not FLASH erasable)</td>
</tr>
<tr>
<td>E0000H - EDFFFH</td>
<td>Parametric Data Storage Blocks (e.g OEM Logo)</td>
</tr>
<tr>
<td></td>
<td>&amp; Optional System BIOS extensions</td>
</tr>
</tbody>
</table>
Recovery
BIOS
Procedures
Recovery mode

- In the unlikely event that a FLASH upgrade is interrupted catastrophically, it is possible the BIOS may be left in an unusable state.
  - If power loss aborts a BIOS update, the main array block will be partially programmed/erased and the code in this block unusable.
  - The system will “hang” or not boot at all.
- Since standard BIOS code does not support boot block recovery, the BIOS software engineers must design the recovery code for the 8 KByte block.
  - NOTE: Some versions of BIOS no longer have room for the recovery code and it has been removed.
Address Inversion Configuration (**Normal**)  
**A16 Inverted**

<table>
<thead>
<tr>
<th>Sys Addr</th>
<th>Flash Addr</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFFFF</td>
<td>EFFFF</td>
<td>Intel 28F001BXT 1Mbit FLASH</td>
</tr>
<tr>
<td>F0000</td>
<td>E0000</td>
<td>64K</td>
</tr>
<tr>
<td>EFFFF</td>
<td>FFFFF</td>
<td>Main BIOS</td>
</tr>
<tr>
<td>EE000</td>
<td>FE000</td>
<td>Recovery Code 8K Boot Block (H/W Protected)</td>
</tr>
<tr>
<td>EDFFF</td>
<td>FDFFF</td>
<td>Parameter Block - Possible ESCD (Plug &amp; PLay Cfg Area)</td>
</tr>
<tr>
<td>ED000</td>
<td>FD000</td>
<td>Parameter Block - Possible OEM Logo Area</td>
</tr>
<tr>
<td>ECFFF</td>
<td>FCFFF</td>
<td></td>
</tr>
<tr>
<td>EC000</td>
<td>FC000</td>
<td></td>
</tr>
<tr>
<td>EBFFF</td>
<td>FBFFF</td>
<td>System BIOS - Reserved</td>
</tr>
<tr>
<td>E0000</td>
<td>F0000</td>
<td>Possible EISA Cfg. Info (E8-EA)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Possible Video BIOS 32KB (E0-E8)</td>
</tr>
</tbody>
</table>
Recovery Jumper

In *Normal Mode*, address line A16 is inverted, setting the E000H and F000H segments so that the BIOS is organized as shown.

**System Address** | **FLASH Memory Area**
---|---
F0000H - FFFFFH | 64 KB Main BIOS
EE000H - EFFFFH | 8 KB Boot Block (Not FLASH erasable)
Address Inversion Configuration

Inversion of address line \( A_{16} \) to the 28F001BX moves the boot block to the lower half (Exxxx) of flash memory as seen by the system (SA19:0).

\[
\begin{array}{c}
A_{16} \\
EE000H = 1110 1110 0000 0000 0000y - \text{Sys Addr} \\
FE000H = 1111 1110 0000 0000 0000y - \text{Flash Addr}
\end{array}
\]

\[
\begin{array}{ccccccc}
A_{19} & A_{18} & A_{17} & A_{16} & A_{15} & \ldots & A_{0} & \text{Address} \\
1 & 1 & 1 & 0 & xxxx & & & Exxxx \\
1 & 1 & 1 & 1 & xxxx & & & Fxxxx
\end{array}
\]
Recovery mode

- To boot from the **boot recovery block**, restore address $A_{16}$ polarity.
  - A jumper or switch on the motherboard can toggle the $A_{16}$ restore logic and “un-invert” it.
  - After reconfiguration, the processor **boots from the boot block** and executes its recovery algorithm to restore main array block contents.

- Re-inverting $A_{16}$ reinstates normal system bootup and operation.
Recovery mode removes the inversion on A16, swapping the E000H and F000H segments so that the 8 KB boot block resides at FE000H where the CPU expects the bootstrap loader.

**System Address**  **FLASH Memory Area**

FE000H - FFFFFH  8 KB Boot Block
# Address Configuration (Recovery)

## No A16 Inversion

<table>
<thead>
<tr>
<th>Sys Addr</th>
<th>Flash Addr</th>
<th>Intel 28F001BXT 1Mbit FLASH</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFFFFFF</td>
<td>FFFFFFF</td>
<td><strong>FFFFF</strong> Recovery Code</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>8K Boot Block</strong></td>
</tr>
<tr>
<td>FE000</td>
<td>FE000</td>
<td></td>
</tr>
<tr>
<td>FDFFF</td>
<td>FDFFF</td>
<td>Parameter Block - Possible</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ESCD (Plug &amp; PLay Cfg Area)</td>
</tr>
<tr>
<td>FD000</td>
<td>FD000</td>
<td></td>
</tr>
<tr>
<td>FCFFF</td>
<td>FCFFF</td>
<td>Parameter Block - Possible</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OEM Logo Area</td>
</tr>
<tr>
<td>FC000</td>
<td>FC000</td>
<td></td>
</tr>
<tr>
<td>FBFFFF</td>
<td>FBFFFF</td>
<td>System BIOS Reserved</td>
</tr>
<tr>
<td>F0000</td>
<td>F0000</td>
<td></td>
</tr>
<tr>
<td>EFFFFF</td>
<td>EFFFFF</td>
<td><strong>Main BIOS</strong></td>
</tr>
<tr>
<td>E0000</td>
<td>E0000</td>
<td><strong>64K</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sys Addr</th>
<th>Flash Addr</th>
<th>Intel 28F001BXT 1Mbit FLASH</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFFFFFF</td>
<td>FFFFFFF</td>
<td><strong>FFFFF</strong> Recovery Code</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>8K Boot Block</strong></td>
</tr>
<tr>
<td>FE000</td>
<td>FE000</td>
<td></td>
</tr>
<tr>
<td>FDFFF</td>
<td>FDFFF</td>
<td>Parameter Block - Possible</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ESCD (Plug &amp; PLay Cfg Area)</td>
</tr>
<tr>
<td>FD000</td>
<td>FD000</td>
<td></td>
</tr>
<tr>
<td>FCFFF</td>
<td>FCFFF</td>
<td>Parameter Block - Possible</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OEM Logo Area</td>
</tr>
<tr>
<td>FC000</td>
<td>FC000</td>
<td></td>
</tr>
<tr>
<td>FBFFFF</td>
<td>FBFFFF</td>
<td>System BIOS Reserved</td>
</tr>
<tr>
<td>F0000</td>
<td>F0000</td>
<td></td>
</tr>
<tr>
<td>EFFFFF</td>
<td>EFFFFF</td>
<td><strong>Main BIOS</strong></td>
</tr>
<tr>
<td>E0000</td>
<td>E0000</td>
<td><strong>64K</strong></td>
</tr>
</tbody>
</table>
Recovery Mode

- **Recovery mode** swaps the E000H and F000H segments.
  - The 8 KB boot block then resides at FE000H where the CPU expects the bootstrap loader to exist.
- This mode is only necessary in the unlikely event that a BIOS upgrade procedure is interrupted, causing the BIOS area to be left in an unusable state.
  - This mode can also be used by debug technicians to re-install a suspected corrupt BIOS.
- NOTE: The majority of the system board must be operational to use the recovery procedure which boots from the floppy drive.
Recovery Procedure

- **Boot Block Recovery requires the following steps.**
  - **NOTE:** Be sure a speaker has been attached to the board, and a floppy drive is connected as drive A:

  1. **Change Flash Recovery jumper (RCVR FLASH) to the recovery mode position.**
  2. **Install the bootable upgrade diskette into drive A:**
     - This must be a system disk
       - e.g. MSDOS.SYS, IO.SYS & COMMAND.COM
     - The recovery disk will also have an AUTOEXEC.BAT file which will call the Recovery program (e.g. FMUP.EXE)
  3. **Reboot the system**
Recovery Procedure

4. Because of the small amount of code available in the non-erasable boot block area, no video is available to direct the procedure.

- The procedure can be monitored by listening to the speaker and looking at the floppy drive LED.
  - NOTE: A POST card will show a much shorter POST sequence.

- When the system beeps and the floppy drive LED is lit, the system is copying the recovery code into FLASH device.

- As soon as the drive LED goes off, the recovery is complete.
Recovery Procedure

5. Turn the system off

6. Change the Flash Recovery jumper back to the default position

7. Continue with the original upgrade if version of Recovery BIOS installed is not the desired version.

NOTE: Technicians may want to restore CMOS defaults (Clear CMOS) if you changed the BIOS.

- If a much different version of BIOS was installed, there is a possibility that the contents of some CMOS locations will be used differently by this BIOS.
FLASH Memory Pin Description
The following diagrams show the pin descriptions for a typical FLASH Memory Chip.

Technicians often need to check the CE#, OE#, and Address & Data lines when a system cannot read the FLASH Correctly.

- The CE# (Chip Enable) pin should be asserted (active low) whenever an address is generated in the FLASH address range.
  - 0F0000H to 0FFFFFFFH physical.
- The OE# (Output Enable) pin should be asserted (active low) whenever a code fetch (memory code read) bus cycle is generated in the Flash Address Range.
### 28F001 BX Pin Description

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Type</th>
<th>Name and Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>A₀-A₁₆</td>
<td>input</td>
<td><strong>ADDRESS INPUTS:</strong> For memory addresses. Addresses are internally latched during a write cycle.</td>
</tr>
<tr>
<td>DQ₀-DQ₇</td>
<td>output/input</td>
<td><strong>DATA OUTPUTS/INPUTS:</strong> Inputs data and commands during memory write cycles. Outputs data during memory.</td>
</tr>
<tr>
<td>CE#</td>
<td>input</td>
<td><strong>CHIP ENABLE:</strong> Activates the device’s control logic, input buffers, decoders and sense amplifiers. CE# is active low.</td>
</tr>
<tr>
<td>RP#</td>
<td>input</td>
<td><strong>POWERDOWN:</strong> RP# high -&gt; normal operation. RP# =0 locks out erase or write operations, providing data protection during power transitions.</td>
</tr>
<tr>
<td>OE#</td>
<td>input</td>
<td><strong>OUTPUT ENABLE:</strong> Gates the device’s outputs through the data buffers during a read cycle. OE# is active low.</td>
</tr>
<tr>
<td>WE#</td>
<td>input</td>
<td><strong>WRITE ENABLE</strong> : Controls writes to the Command Register and array blocks. WE# is active low</td>
</tr>
<tr>
<td>Vpp</td>
<td></td>
<td><strong>ERASE/PROGRAM POWER SUPPLY :</strong> For erasing blocks of the array or programming bytes of each block</td>
</tr>
<tr>
<td>Vcc</td>
<td></td>
<td><strong>DEVICE POWER SUPPLY:</strong> (5V +/- 10%)</td>
</tr>
<tr>
<td>GND</td>
<td></td>
<td><strong>GROUND</strong></td>
</tr>
</tbody>
</table>
28F001 BX PLCC Pinout

N28F001BX
32 Lead PLCC
0.450” x 0.550”
Top View

DQ, DQ2, GND, DQ3, DQ4, DQ5, DQ6

A12, A15, A16, Vpp, Vcc, WE#, RP#

A14, A13, A8, A9, A11

CE#

OE#
28F001 BX TSOP Pinout

Standard Pinout
E28FOO 1BX
32 Lead TSOP
8MM x 20MM
Top View

TSOP Lead Configuration
Using an O’Scope to Check FLASH Data
28F001 BX FLASH Pin Description

A typical O’Scope set-up for verifying data follows.

• Use an ITP to **loop on reading 1 byte** from the FLASH
  • e.g. - Use the following ITP command:
    » While (1) {Byte 0FFFF0p}

• **CH-1: OE#** - Trigger on this channel & use it to **qualify** the data on CH-2.
  • The data is only valid during the time that **OE# = 0**.

• **CH-2: D(7:0)** - Look at each data bit (D7:0) in sequence to verify the data.
  • e.g. - EAH (1110 1010y).

If the data is incorrect, check for a correct address of 0FFFF0H on A15:0.
SUMMARY

WE HAVE DISCUSSED THE FOLLOWING:

- The organization of PC Base Memory (0-1MB).
- The contents of ROM in terms of the BIOS & POST.
- The organization of a typical Flash Memory Chip.
- Recovery BIOS procedures.
- The Pinout of a typical Flash Memory Chip.
- How to use an O’Scope to check Flash Data Pins.
Instructor Notes

- If time permits, use an O'Scope to look at FLASH DATA.
  - If an ITP is available, look at FFFF0p - “EA”
  - If an ITP is available, look at FFFF1p - “5B”
  - Assemble the following in memory for a tight O'scope loop for reading a byte from PROMs.
    
    ```asm
    asm 800p= "mov ax, 0f000", "mov ds, ax", "mov bx, 0fff0", "mov al, [bx]", "jmp $-0a"
    cs=80; ip=0; bx=0
    ```