Use Free Space in AMI BIOS on M6117D

2003-10-24

As one of the leading companies in the embedded field, we developed our 386 BIOS with three special features: quick boot-up, with spare space in CMOS RAM for customer programming and with spare space in BIOS for customer programming. This page is composed to guide you how to use the free space in a BIOS chip.

We reserve approximate 2K byte of free space in BIOS. This free space is located at the very early stage during POST (Power On Self Test). A user can use this space to implement specific software for customized display or to control I/O devices before boot-up. This technology has been broadly applied in POS (custom display) and the machine control (I/O setting), in which the users need to pre-set or pre-format the external devices before the system is booted up.

Please note this spare space is check-free by Checksum.

Remove the BIOS

First of all, the BIOS chip must be removed from the 386 single board computer. Please use EPROM WRITER to read the binary codes of the BIOS and then set a file name to save it. (Or you can simply ask our sales representatives to send you the file).

For example: Remove the BIOS chip from the 6012 tiny module and then use EPROM WRITER to read the binary codes and save it as the file name, 6012A4.ROM.

Check the location of the spare space

The next step is to use the DOS utility, DEBUG.COM, to check the starting address of this free space inside the BIOS.

```
C:\>debug [enter]
-n 6012a4.rom
-l 2000:0
-u 2000:0
2000:0000 B00A        MOV  AL,0A
2000:0002 E670        OUT  70,AL
2000:0004 E471        IN   AL,71
2000:0006 2470        AND  AL,70
2000:0008 3C20        CMP  AL,20
2000:000A 7564        JNZ  0070
2000:000C B00A        MOV  AL,0A
2000:000E E670        OUT  70,AL
```

```
2000:0010 E471          IN   AL,71
2000:0012 A880          TEST AL,80
2000:0014 75F6          JNZ  000C
2000:0016 B000          MOV  AL,00
2000:0018 E670          OUT  70,AL
2000:001A E471          IN   AL,71
2000:001C 3C59          CMP  AL,59
2000:001E 7750          JA   0070
-u
2000:0020 B00A          MOV  AL,0A
2000:0022 E670          OUT  70,AL
2000:0024 E471          IN   AL,71
2000:0026 A880          TEST AL,80
2000:0028 75F6          JNZ  0020
2000:002A B002          MOV  AL,02
2000:002C E670          OUT  70,AL
2000:002E E471          IN   AL,71
2000:0030 3C59          CMP  AL,59
2000:0032 773C          JA   0070
2000:0034 B00A          MOV  AL,0A
2000:0036 E670          OUT  70,AL
2000:0038 E471          IN   AL,71
2000:003A A880          TEST AL,80
2000:003C 75F6          JNZ  0034
2000:003E B004          MOV  AL,04
-u
2000:0040 E670          OUT  70,AL
2000:0042 E471          IN   AL,71
2000:0044 3C23          CMP  AL,23
2000:0046 7728          JA   0070
2000:0048 B00A          MOV  AL,0A
2000:004A E670          OUT  70,AL
2000:004C E471          IN   AL,71
2000:004E A880          TEST AL,80
2000:0050 75F6          JNZ  0048
2000:0052 B006          MOV  AL,06
2000:0054 E670          OUT  70,AL
2000:0056 E471          IN   AL,71
2000:0058 3C31          CMP  AL,31
2000:005A 7714          JA   0070
2000:005C B00A          MOV  AL,0A
```

```
2000:005E E670          OUT  70,AL
-u
2000:0060 E471          IN   AL,71
2000:0062 A880          TEST AL,80
2000:0064 75F6          JNZ  005C
2000:0066 B008          MOV  AL,08
2000:0068 E670          OUT  70,AL
2000:006A E471          IN   AL,71
2000:006C 3C12          CMP  AL,12
2000:006E 7616          JBE  0086
2000:0070 B409          MOV  AH,09
2000:0072 8AC4          MOV  AL,AH
2000:0074 E670          OUT  70,AL
2000:0076 B001          MOV  AL,01
2000:0078 E671          OUT  71,AL
2000:007A FECC          DEC  AH
2000:007C 75F4          JNZ  0072
2000:007E B00A          MOV  AL,0A
-u

2000:0080 E670          OUT 70,AL
2000:0082 B024          MOV AL,24
2000:0084 E671          OUT 71,AL
2000:0086 FFE5          JMP BP
2000:0088 0000          ADD [BX+SI],AL
2000:008A 0000          ADD [BX+SI],AL
2000:008C 0000          ADD [BX+SI],AL
2000:008E 0000          ADD [BX+SI],AL
2000:0090 0000          ADD [BX+SI],AL
2000:0092 0000          ADD [BX+SI],AL
2000:0094 0000          ADD [BX+SI],AL
2000:0096 0000          ADD [BX+SI],AL
2000:0098 0000          ADD [BX+SI],AL
2000:009A 0000          ADD [BX+SI],AL
2000:009C 0000          ADD [BX+SI],AL
2000:009E 0000          ADD [BX+SI],AL
-u
```

In the debug mode, we will suggest the users to read the BIOS data and to allocate it in any spare SEGMENT with the OFFSET starting from 0000. At the above illustration, we put it in the SEGMENT 2000 Hex with OFFSET 0000.

After the setting the user will find we already uses the space from OFFSET 0000 to 0087 Hex. The data after SEGMENT:OFFSET (2000:0088) are all zero (0), and it means the free space for customer application.

Relocate the "JMP BP" command

Remove the line of program of "JMP BP" , which locates in SEGMENT:OFFSET (2000:0086). Relocate this line into the last line of this free space.

Therefore, the user must put the customized program into the space from SEGMENT:OFFSET (2000:0086) and add the "JMP BP" into the last line of the user program.

Note

The users **MUST NOT** use any commands, such as "PUSH" or "POP", of the x86 assembly language because this free space is located at a very early stage of POST at which stage the CPU does not initial the DRAM (main memory) yet.

After finishing editing the program, save it under a filename to update your BIOS and it is done.

Technical Support

For more technical support, please visit http://www.dmp.com.tw/tech or mail to tech@dmp.com.tw.