BIOS Practice: CMOS

原创 Anthony Posted on 2020-12-14 15:41:04 Read 1.7k Collection 7 Likes 1

Category Column: BIOS learning practice Article Tags: windows

CMOS, as a BIOS engineer, of course refers to CMOS in the computer field.

1.What is CMOS used for in the computer field?

In the computer field, CMOS often refers to the chip that stores basic computer startup information (such as date, time, startup settings, etc.).

2. What is the difference between CMOS and BIOS?

Sometimes people confuse CMOS and BIOS. In fact, CMOS is a readable and writable parallel or serial FLASH chip (RAM chip) on the motherboard. This chip is used to save the hardware configuration of BIOS and the user's settings of certain parameters. CMOS is used as an erasable chip. Users usually do not care about CMOS hardware problems, but only care about the information written on CMOS, that is, the BIOS settings.

3.What is the power supply for CMOS?

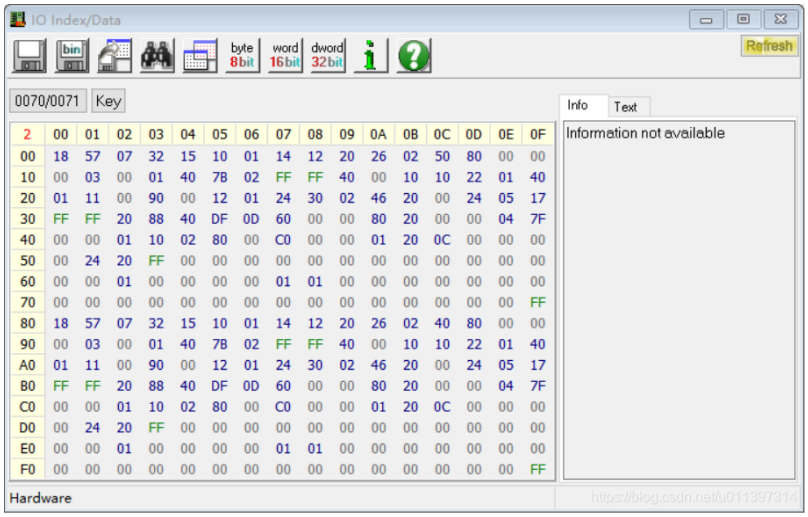
CMOS can be powered by the motherboard 's battery, so information will not be lost even if the system loses power.

4.What is the use of CMOS?

Because of its readable and writable characteristics, it is used on the computer motherboard to save the data after the BIOS sets the computer hardware parameters. The CMOS ROM itself is just a memory with only data saving function. This chip is only used to store data .

The settings of various parameters in BIOS need to be done through a special program. The BIOS setup program is generally integrated into the chip by the manufacturer. When the computer is turned on, you can enter the BIOS setup program through a specific button to conveniently set the system. Therefore, BIOS setup is sometimes also called CMOS setup.

Let's actually take a look at what the CMOS space is like. The tool uses RW and can be downloaded for free from the official website. This is a very useful tool



2	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00	18	57	07	32	15	10	01	14	12	20	26	02	50	80	00	00
10	00	03	00	01	40	7B	02	FF	FF	40	00	10	10	22	01	40
20	01	11	00	90	00	12	01	24	30	02	46	20	00	24	05	17
30	FF	FF	20	88	40	DF	0D	60	00	00	80	20	00	00	04	7F
40	00	00	01	10	02	80	00	C0	00	00	01	20	0C	00	00	00
50	00	24	20	FF	00	00	00	00	00	00	00	00	00	00	00	00
60	00	00	01	00	00	00	00	01	01	00	00	00	00	00	00	00
70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	FF
80	18	57	07	32	15	10	01	14	12	20	26	02	40	80	00	00
90	00	03	00	01	40	7B	02	FF	FF	40	00	10	10	22	01	40
A0	01	11	00	90	00	12	01	24	30	02	46	20	00	24	05	17
B0	FF	FF	20	88	40	DF	0D	60	00	00	80	20	00	00	04	7F
C0	00	00	01	10	02	80	00	C0	00	00	01	20	0C	00	00	00
D0	00	24	20	FF	00	00	00	00	00	00	00	00	00	00	00	00
E0	00	00	01	00	00	00	00	01	01	00	00	00	00	00	00	00
F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	FF

We can see that 0x50, 0x09, 0x08, 0x07, 0x04, 0x02, 0x00. The values of these addresses combined are 15:07:18 on December 14, 2020

If we don't use RW, we can also write a program to read the values of all address spaces:

```
#include "stdio.h"
#include "dos.h"
#include "conio.h"
#define  MAXSIZE   16

void main()
{
    unsigned char addr=0;
    int row,col;
    printf("\n\t\t   CMOS Read/Write Routine\n");
    printf("\t      Q or q :Quit    Enter:Flash ");
    printf("\t     00  01  02  03  04  05  06  07  08");
    printf("  09  0A  0B  0C  0D  0E  0F\n");
    for(row=0;row<9;row++)
    {
```

```
          printf("         %X0  ",row);
          for(col=0;col<MAXSIZE;col++)
          {
            outportb(0x70,addr);
            value[row][col]=inportb(0x71);
                if(value[row][col]<16){
                    printf("0%X  ",value[row][col]);
                }
                else{
                    printf("%X  ",value[row][col]);
                    addr++;
                    }
          }
          printf("\n");
    }
    addr=0;
    for(row=9;row<MAXSIZE;row++)
    {
          printf("         %X0  ",row);
          for(col=0;col<MAXSIZE;col++)
          {
            outportb(0x71,addr);
            value[row][col]=inportb(0x72);
                if(value[row][col]<16){
                    printf("0%X  ",value[row][col]);
                }
                else{
                   printf("%X  ",value[row][col]);
                   addr++;
                    }
          }
          printf("\n");
    }
    return ;
}

```


Now that you know how to read it, writing it is easy:


```
outportb(INDEX_PORT,index);
outportb(DATA_PORT,data);
```

During the debugging process of BIOS, if you want to analyze by writing values to CMOS, then write the above two sentences in the code you suspect. Of course, before writing this, you also need to add the corresponding library.