Control Vortex86SX/DX CPU Speed

2008-01-24

Internally the Vortex86SX is using the PLL technology (Phase-Locked Loop), the CPU clock can be adjust by changing the register value of the North Bridge Offset register A0h.

The CPU speed could be divided from 1 to 8 by default CPU clock. For example, if the default CPU clock is 300MHz, and you choice the "CPU speed divide by 5", the CPU speed will be 300 / 5 = 60 MHz. And please be noticed the CPU speed could not lower then PCI speed which is 33MHz.

To change CPU clock, find PCI register A0H in north bridge (Vendor ID: 17F3, Device: 6021). Bit 7-3 is reserved and bit 2-0 is CPU speed divided control:

Bit[2-0]	Description
000	Divide 1
001	Divide 2
010	Divide 3
011	Divide 4
100	Divide 5
101	Divide 6
110	Divide 7
111	Divide 8

For example: if CPU clock is 300MHz and set speed divided control to "001", CPU clock will be 150MHz. Here is assembler example:

```
mov dx,  cf8h ; PCI address port set = north bridge offset register a0h
mov eax, 800000a0h
out dx,  eax

mov dx,  cfch ; PCI data port read / write
in  eax, dx
                    ; if CPU clock is 300MHz
or  eax, 00000001h ; set CPU clock to 150MHz
or  eax, 00000004h ; set CPU clock to  60MHz
out dx,  eax
```

DOS Example

If you are using DOS that can not use 32-bit instructions, refer to this example:

```
#include <stdio.h>
#include <conio.h>

#define PCI_BIOS_INTERRUPT          0x1A
#define PCI_BIOS_FUNCTION_ID        0xB1
#define PCI_BIOS_PRESENT            0x01
#define PCI_BIOS_FIND_DEVICE        0x02
#define PCI_BIOS_READ_CONFIG_BYTE   0x08
#define PCI_BIOS_READ_CONFIG_WORD   0x09
#define PCI_BIOS_WRITE_CONFIG_BYTE  0x0B
```

```
#define PCI_BIOS_WRITE_CONFIG_WORD  0x0C

unsigned char _cBusNum;
unsigned char _cDeviceNum;

char          IsPciBiosPresent();
unsigned char PciBios_FindDevice(unsigned nVenderID, unsigned nDeviceID, int nIndex, unsigned char *pcBusNum, unsigned
char *pcDeviceNum);
unsigned char PciBios_ReadByte (char cBusNum, char cDeviceNum, int nOffset);
unsigned char PciBios_WriteByte(char cBusNum, char cDeviceNum, int nOffset, unsigned char cValue);

void main()
{
  unsigned char cBusNum, cDeviceNum;
  unsigned char c;

  /* Check PCI BIOS */
  if(!IsPciBiosPresent())
  {
    printf("Unable to find PCI BIOS.\n");
    return;
  }

  /* Find bus and device number */
  PciBios_FindDevice(0x17F3, 0x6021, 0, &cBusNum, &cDeviceNum);

  /* Read A0H in north bridge (Vendor ID: 17F3, Device: 6021).
     Bit 7-3 is reserved and bit 2-0 is CPU speed divided control. */
  c = PciBios_ReadByte(_cBusNum, _cDeviceNum, 0xA0);
  c &= 0x07;  /* Clear bit 2-0 */

  /* Set clock: bit[2-0]
     000 -> Divide 1
     001 -> Divide 2
     010 -> Divide 3
     011 -> Divide 4
     100 -> Divide 5
     101 -> Divide 6
     110 -> Divide 7
     111 -> Divide 8 */
  c |= 0x01;  /* If CPU is 300MHz, set clock to 150MHz */
}

char IsPciBiosPresent()
{
    char cRet;
    asm {
        mov ah, PCI_BIOS_FUNCTION_ID
        mov al, PCI_BIOS_PRESENT
        int PCI_BIOS_INTERRUPT
        mov cRet, ah
    }
    return !cRet;
}

unsigned char PciBios_FindDevice(unsigned nVenderID, unsigned nDeviceID, int nIndex,
                        unsigned char *pcBusNum, unsigned char *pcDeviceNum)
{
    unsigned char cRet, cBus, cDevice;
    asm {
        mov ah, PCI_BIOS_FUNCTION_ID
        mov al, PCI_BIOS_FIND_DEVICE
        mov cx, nDeviceID
        mov dx, nVenderID
        mov si, nIndex
        int PCI_BIOS_INTERRUPT
        mov cRet, ah
        mov cBus, bh
        mov cDevice, bl
    }
    *pcBusNum    = cBus;
    *pcDeviceNum = cDevice;
    return !cRet;
}

unsigned char PciBios_ReadByte(char cBusNum, char cDeviceNum, int nOffset)
{
```

```
    unsigned char data, cRet;
    asm{
        mov ah, PCI_BIOS_FUNCTION_ID
        mov al, PCI_BIOS_READ_CONFIG_BYTE
        mov bh, cBusNum
        mov bl, cDeviceNum
        mov di, nOffset
        int PCI_BIOS_INTERRUPT
        mov cRet, ah
        mov data, cl
    }
    if (cRet)
        return -1;
    return data;
}

unsigned char PciBios_WriteByte(char cBusNum, char cDeviceNum, int nOffset, unsigned char cValue)
{
    unsigned char cRet;
    asm {
        mov ah, PCI_BIOS_FUNCTION_ID
        mov al, PCI_BIOS_WRITE_CONFIG_BYTE
        mov bh, cBusNum
        mov bl, cDeviceNum
        mov di, nOffset
        mov cl, cValue
        int PCI_BIOS_INTERRUPT
        mov cRet, ah
    }
    return !cRet;
}
```

Technical Support

For more technical support, please visit http://www.dmp.com.tw/tech/vortex86sx or mail to tech@dmp.com.tw.