Министерство образования Республики Беларусь

Учреждение образования

БелорусскиЙ государственный университет

информатики и радиоэлектроники

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Отчет по лабораторной работе № 3

по предмету

Аппаратное обеспечение компьютерной техники на тему:

Исследование работы комбинационных и последовательных цифровых устройств

Вариант 5

|  |  |  |
| --- | --- | --- |
| Выполнил  Студент гр. 951007 |  | М.А. Воривода |
| Проверила |  | О. Н. Образцова |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Минск, 2020

**Задание**

Прочитать регистры автоконфигурации и вывести следующую информацию:

1. Адрес устройства (номер шины, номер устройства и номер функции);
2. 16-разрядный код производителя (в hex-формате);
3. 16-разрядный код устройства (в hex-формате);
4. Производитель и название устройства;
5. Если устройство не мост (0-бит поля Header Type =0) вывести и расшифровать значение поля Сlass Code;
6. Если устройство не мост (0-бит поля Header Type =0) вывести и расшифровать значение поля Interrupt Pin;
7. Если устройство мост (0-бит поля Header Type =1) вывести и расшифровать значение полей Primary Buss Number, Secondary Buss Number, Subordinate Buss Number.

*Реализовано на Linux Debian 10.6.*

**Адрес устройства**



Для перебора всех возможных адресов необходимо три цикла:

1. Первый цикл – перебор шин. Всего возможно 256 шин, номер шины необходимо сдвинуть 16 раз влево.
2. Второй цикл – перебор устройств. Всего возможно 32 устройств, номер устройства необходимо сдвинуть 11 раз влево.
3. Третий цикл – перебор функций. Всего возможно 8 функций, номер функций необходимо сдвинуть 2 раза влево.

**Коды производителя и устройства. Производитель и название устройства.**

0х00 – адрес регистра, хранящего коды производителя (Vendor ID) и устройства (Device ID).



Взятие Device ID – сдвинуть полученное число 16 раз вправо.

Взятие Vendor ID – наложение маски 0xFFFF.

Найти Device ID и Vendor ID в структурах PciDevTable и PciVenTable.

**Class code**



0х08 – адрес регистра, хранящего код класса (старшие 24 бита).

Взятие Base class – сдвинуть полученное из регистра число 24 раза вправо (если равно 6, то это мост).

Взятие Subclass – сдвинуть 16 раз вправо и наложить маску 0xFF.

Взятие Interface – сдвинуть 8 раз вправо и наложить маску 0xFF.

**Bus number**



0x18 – адрес регистра, хранящего номер шины.

Взятие Subordinate bus number – сдвинуть полученное из регистра число 16 раза вправо и наложить маску 0xFF.

Взятие Secondary bus number – сдвинуть 8 раз вправо и наложить маску 0xFF.

Взятие Primary bus number – наложить маску 0xFF.

**Interrupt pin**

0x3C – адрес регистра, хранящего контакт прерывания.

Взятие Interrupt pin – сдвинуть полученное число 8 раз и наложить маску 0xFF.



**Код программы**

#include <stdio.h>  
#include <sys/io.h>  
#include <unistd.h>  
#include "pci.h"  
  
const unsigned int PCI\_ENABLE\_BIT = 0x80000000;  
const unsigned int PCI\_CONFIG\_ADDRESS = 0xCF8;  
const unsigned int PCI\_CONFIG\_DATA = 0xCFC;  
  
FILE \* output;  
  
unsigned int try\_device(unsigned char bus, unsigned char device, unsigned char func, unsigned char pcireg)  
{  
  
 unsigned int tested\_address = PCI\_ENABLE\_BIT | (bus << 16) | (device << 11) | (func << 8) | (pcireg << 2);  
  
 outl(tested\_address, PCI\_CONFIG\_ADDRESS);  
 unsigned int ret = inl(PCI\_CONFIG\_DATA);  
  
 return ret;  
}  
  
void printDev(unsigned int data)  
{  
 unsigned int venId = data & 0xFFFF;  
 unsigned int devId = data >> 16;  
  
 printf("\nVendorID = 0x%08x\nDeviceID = 0x%08x\n\n", venId, devId);  
 fprintf(output, "\nVendorID = 0x%08x\nDeviceID = 0x%08x\n\n", venId, devId);  
 for(int i = 0; i < PCI\_VENTABLE\_LEN; i++)  
 {  
  
 if(venId == PciVenTable[i].VendorId)  
 {  
 printf("Vendor: %s\n", PciVenTable[i].VendorName);  
 fprintf(output, "Vendor: %s\n", PciVenTable[i].VendorName);  
 }  
 }  
  
 for(int i = 0; i < PCI\_DEVTABLE\_LEN; i++)  
 {  
 if(devId == PciDevTable[i].DeviceId && venId == PciDevTable[i].VendorId)  
 {  
 printf("Device: %s\n", PciDevTable[i].DeviceName);  
 fprintf(output, "Device: %s\n", PciDevTable[i].DeviceName);  
 }  
 }  
}  
  
char \* interruptPinType(unsigned char pin)  
{  
 switch(pin)  
 {  
 case 0:  
 return "Not used";  
 break;  
 case 1:  
 return "INTA#";  
 break;  
 case 2:  
 return "INTB#";  
 break;  
 case 3:  
 return "INTC#";  
 break;  
 case 4:  
 return "INTD#";  
 break;  
 case 5:  
 return "Reserved";  
 break;  
 }  
}  
  
void checkDevice(unsigned char bus, unsigned char device, unsigned char func)  
{  
 unsigned int data = try\_device(bus, device, func, 0x08);  
 unsigned char baseClass = data >> 24;  
 unsigned char subClass = (data >> 16) & 0xFF;  
 unsigned char interface = (data >> 8) & 0xFF;  
  
 if(baseClass != 0x06)  
 {  
 printf("+----------------------------------------------+\n");  
 fprintf(output, "+----------------------------------------------+\n");  
 char isFound = 0;  
 for(int i = 0; i < PCI\_CLASSCODETABLE\_LEN; i++)  
 {  
 if(baseClass == PciClassCodeTable[i].BaseClass  
 && subClass == PciClassCodeTable[i].SubClass  
 && interface == PciClassCodeTable[i].ProgIf)  
 {  
 printf("Class code: \n\tBase class: %s \n\tSubclass: %s \n\tInterface: %s\n", PciClassCodeTable[i].BaseDesc, PciClassCodeTable[i].SubDesc, PciClassCodeTable[i].ProgDesc);  
 fprintf(output, "Class code: \n\tBase class: %s \n\tSubclass: %s \n\tInterface: %s\n", PciClassCodeTable[i].BaseDesc, PciClassCodeTable[i].SubDesc, PciClassCodeTable[i].ProgDesc);  
 isFound = 1;  
 }  
 }  
  
 if(!isFound)  
 {  
 printf("Class code: [NOT FOUND]\n");  
 fprintf(output, "Class code: [NOT FOUND]\n");  
 }  
 data = try\_device(bus, device, func, 0x3C);  
 unsigned char interruptPin = (data >> 8) & 0xFF;  
  
 printf("+----------------------------------------------+\n");  
 printf("Interrupt pin: %s (0x%08x)\n", interruptPinType(interruptPin), interruptPin);  
 fprintf(output, "+----------------------------------------------+\n");  
 fprintf(output, "Interrupt pin: %s (0x%08x)\n", interruptPinType(interruptPin), interruptPin);  
 }  
 else  
 {  
 unsigned int data = try\_device(bus, device, func, 0x18);  
 unsigned char subordinateBus = (data >> 16) & 0xFF;  
 unsigned char secondaryBus = (data >> 8) & 0xFF;  
 unsigned char primaryBus = data & 0xFF;  
  
 printf("+----------------------------------------------+\n");  
 printf("Bus: \n\tSubordinate bus: 0x%08x \n\tSecondary bus: 0x%08x \n\tPrimary bus: 0x%08x\n", subordinateBus, secondaryBus, primaryBus);  
 fprintf(output, "+----------------------------------------------+\n");  
 fprintf(output, "Bus: \n\tSubordinate bus: 0x%08x \n\tSecondary bus: 0x%08x \n\tPrimary bus: 0x%08x\n", subordinateBus, secondaryBus, primaryBus);  
 }  
}  
  
int find()  
{  
 unsigned char bus, device, func;  
 unsigned int data;  
  
 for (bus = 0; bus < 255; bus++)  
 {  
 for (device = 0; device < 32; device++)  
 {  
 for (func = 0; func < 8; func++)  
 {  
 data = try\_device(bus, device, func, 0);  
 if (data != 0xffffffff)  
 {  
 printf("bus %d, device %d, func %d: vendor=0x%08x\n", bus, device, func, data);  
 fprintf(output, "bus %d, device %d, func %d: vendor=0x%08x\n", bus, device, func, data);  
  
 printDev(data);  
 checkDevice(bus, device, func);  
  
 printf("===================================================================\n");  
 fprintf(output, "===================================================================\n");  
 }  
 }  
 }  
 }  
 return 0;  
}  
  
int main()  
{  
  
 if (geteuid() != 0)  
 {  
 printf("please run under root\n");  
 }  
 else  
 {  
 output = fopen("out.txt", "w");  
 iopl(3);  
 find();  
 }  
  
 return 0;  
}

**Вывод программы**

bus 0, device 0, func 0: vendor=0x12378086

VendorID = 0x00008086

DeviceID = 0x00001237

Vendor: Intel Corporation

Device: 440FX - 82441FX PMC [Natoma]

+----------------------------------------------+

Class code:

Base class: Pre-2.0 PCI Specification Device

Subclass: Non-VGA

Interface:

+----------------------------------------------+

Interrupt pin: Not used (0x00000000)

===================================================================

bus 0, device 1, func 0: vendor=0x70008086

VendorID = 0x00008086

DeviceID = 0x00007000

Vendor: Intel Corporation

Device: 82371SB PIIX3 ISA [Natoma/Triton II]

+----------------------------------------------+

Class code:

Base class: Pre-2.0 PCI Specification Device

Subclass: Non-VGA

Interface:

+----------------------------------------------+

Interrupt pin: Not used (0x00000000)

===================================================================

bus 0, device 1, func 1: vendor=0x71118086

VendorID = 0x00008086

DeviceID = 0x00007111

Vendor: Intel Corporation

Device: 82371AB/EB/MB PIIX4 IDE

+----------------------------------------------+

Class code: [NOT FOUND]

+----------------------------------------------+

Interrupt pin: Not used (0x00000000)

===================================================================

bus 0, device 2, func 0: vendor=0x040515ad

VendorID = 0x000015ad

DeviceID = 0x00000405

Vendor: VMware

Device: SVGA II Adapter

+----------------------------------------------+

Class code:

Base class: Pre-2.0 PCI Specification Device

Subclass: Non-VGA

Interface:

+----------------------------------------------+

Interrupt pin: Not used (0x00000000)

===================================================================

bus 0, device 3, func 0: vendor=0x100e8086

VendorID = 0x00008086

DeviceID = 0x0000100e

Vendor: Intel Corporation

Device: 82540EM Gigabit Ethernet Controller

+----------------------------------------------+

Class code:

Base class: Pre-2.0 PCI Specification Device

Subclass: Non-VGA

Interface:

+----------------------------------------------+

Interrupt pin: Not used (0x00000000)

===================================================================

bus 0, device 4, func 0: vendor=0xcafe80ee

VendorID = 0x000080ee

DeviceID = 0x0000cafe

Vendor: InnoTek Systemberatung GmbH

Device: VirtualBox Guest Service

+----------------------------------------------+

Class code:

Base class: Pre-2.0 PCI Specification Device

Subclass: Non-VGA

Interface:

+----------------------------------------------+

Interrupt pin: Not used (0x00000000)

===================================================================

bus 0, device 5, func 0: vendor=0x24158086

VendorID = 0x00008086

DeviceID = 0x00002415

Vendor: Intel Corporation

Device: 82801AA AC'97 Audio Controller

+----------------------------------------------+

Class code:

Base class: Pre-2.0 PCI Specification Device

Subclass: Non-VGA

Interface:

+----------------------------------------------+

Interrupt pin: Not used (0x00000000)

===================================================================

bus 0, device 6, func 0: vendor=0x003f106b

VendorID = 0x0000106b

DeviceID = 0x0000003f

Vendor: Apple Inc.

Device: KeyLargo/Intrepid USB

+----------------------------------------------+

Class code:

Base class: Pre-2.0 PCI Specification Device

Subclass: Non-VGA

Interface:

+----------------------------------------------+

Interrupt pin: Not used (0x00000000)

===================================================================

bus 0, device 7, func 0: vendor=0x71138086

VendorID = 0x00008086

DeviceID = 0x00007113

Vendor: Intel Corporation

Device: 82371AB/EB/MB PIIX4 ACPI

+----------------------------------------------+

Class code:

Base class: Pre-2.0 PCI Specification Device

Subclass: Non-VGA

Interface:

+----------------------------------------------+

Interrupt pin: Not used (0x00000000)

===================================================================

bus 0, device 13, func 0: vendor=0x28298086

VendorID = 0x00008086

DeviceID = 0x00002829

Vendor: Intel Corporation

Device: 82801HM/HEM (ICH8M/ICH8M-E) SATA Controller [AHCI mode]

+----------------------------------------------+

Class code: [NOT FOUND]

+----------------------------------------------+

Interrupt pin: Not used (0x00000000)

===================================================================