**ЛАБОРАТОРНАЯ РАБОТА №6 НИЗКОУРОВНЕВАЯ РАБОТА С ПЕРИФЕРИЙНЫМИ УСТРОЙСТВАМИ**

**Цели работы:** ознакомиться с началами низкоуровневого программирования периферийных устройств на примере получения информации о доступных USB-устройствах с помощью библиотеки libusb.

**Задание к лабораторной работе:**

1. **Общее описание программы**

В процессе выполнения лабораторной работы была написана программа на языке C++ с использованием библиотеки **libusb**. За основу был взят листинг программы, прикрепленный в файле с заданием к работе. Главная задача, решаемая программой – печать в консоль код класса устройства, его номер, идентификатор производителя, идентификатор устройства, а также расшифровку производителя устройства.

(Полный листинг программы приведен в приложении 1)

1. **Описание выходных данных программы**

Для получения списка всех USB-устройств используется функция **libusb\_get\_device\_list(),** которая возвращает целое число – количество устройств.

Для воспроизведения информации об устройствах была использована функция **printdev(),** которая получает на вход указатель на устройство, и с помощью функции **libusb\_get\_string\_descriptor\_ascii()** и применения различных аргументов (iSerialNumber, iManufacturer) к ней, выводит информацию о подключенных устройствах в формате **<серийный номер // класс устройства // id производителя // id устройства // имя производителя>.**

(Полный результат работы программы приведен в приложении 2)

1. **Анализ полученных результатов**

Было обнаружено 6 USB-устройств, подключенных к компьютеру.

Устройства 1, 3, 4, 6 относятся к классу «концентраторов». Данные устройства встроены внутрь компьютера и не подключены через внешние USB-порты. Вместо имени производителя для этих устройств отображается некоторое техническое название.

Устройство 2 относится к классу «различных устройств», что не позволяет более точно определить предназначение устройства. Устройство также находится внутри компьютера и подключено не через внешний USB-порт. Однако для этого устройства уже корректно отображается имя производителя – SunPlus Inc – производитель чипов и микроконтроллеров.

Устройство 5 относится к классу «беспроводных контроллеров» и является единственным устройством из списка, подключенным через внешний USB-порт – (Bluetooth-мышь). Производитель так же, как и в 1 случае, не определен и устройству присвоено техническое название.

Перед завершением работы программы используются функции **libusb\_free\_device\_list()** для освобождения выделенной памяти и **libusb\_exit()**  для завершения работы с библиотекой libusb.

**Заключение:** в процессе выполнения лабораторной работы были освоены начала низкоуровневого программирования USB-устройств с помощью библиотеки libusb. Также была написана программа, выводящая информацию о подключенных к компьютеру USB-устройствах, результаты выполнения которой были впоследствии проанализированы.

**Приложение 1. Листинг программы**

1. #include <iostream>
2. #include <libusb-1.0/libusb.h>
3. #include <stdio.h>
5. **using namespace std;**
7. void printdev(libusb\_device \*dev);
9. int main()
10. **{**
11. libusb\_device \*\*devs; // Указатель на указатель на устройство
12. // используется для получения списка устройств
13. libusb\_context \*ctx = NULL; // контекст сессии libusb
14. int r; // для возвращаемых значений
15. **ssize\_t cnt; // число найденных USB-устройств**
16. ssize\_t i; // индексная переменная цикла перебора всех устройств
17. r = libusb\_init(&ctx);
18. if(r < 0)
19. {
20. **fprintf(stderr,**
21. "Ошибка: инициализация не выполнена, код: %d.**\n**", r);
22. return 1;
23. }
24. // задать уровень подробности отладочных сообщений
25. **libusb\_set\_debug(ctx, 3);**
26. // получить список всех найденных USB- устройств
27. cnt = libusb\_get\_device\_list(ctx, &devs);
28. if(cnt < 0)
29. {
30. **fprintf(stderr,**
31. "Ошибка: список USB устройств не получен %d.**\n**", r);
32. return 1;
33. }
35. **printf("найдено устройств: %zd\n", cnt);**
36. printf("===========================================================**\n**");
37. printf("\* serial number**\n**");
38. printf("| \* class device**\n**");
39. printf("| \* id vendor**\n**");
40. **printf("| | | \* id device \n");**
42. for(i = 0; i < cnt; i++)
43. {
44. printdev(devs[i]); // печать параметров устройства
45. **}**
46. printf("===========================================================**\n**");
47. // освободить память, выделенную функцией получения списка устройств
48. libusb\_free\_device\_list(devs, 1);
49. libusb\_exit(ctx); // завершить работу с библиотекой libusb,
50. **return 0;**
51. }
53. void printdev(libusb\_device \*dev)
54. {
55. **libusb\_device\_descriptor desc; // дескриптор устройства**
56. int r = libusb\_get\_device\_descriptor(dev, &desc);
57. if (r < 0)
58. {
59. fprintf(stderr,
60. **"Ошибка: дескриптор устройства не получен, код: %d.\n",r);**
61. return;
62. }
63. libusb\_device\_handle \*handle;
64. unsigned char data[256];
65. **libusb\_open(dev, &handle);**
66. if (handle == NULL)
67. {
68. fprintf(stderr,
69. "Ошибка: handle не открыт **\n**");
70. **return;**
71. }
72. int x = libusb\_get\_string\_descriptor\_ascii(handle, desc.iSerialNumber, data, sizeof(data));
73. if (x == LIBUSB\_ERROR\_IO)
74. {
75. **fprintf(stderr,**
76. "Ошибка: серийный номер не получен **\n**");
77. return;
78. }
79. printf("%s//", data);
81. unsigned char str\_man[256];
82. x = libusb\_get\_string\_descriptor\_ascii(handle, desc.iManufacturer, str\_man, sizeof(str\_man));
83. if (x == LIBUSB\_ERROR\_IO)
84. {
85. **fprintf(stderr,**
86. "Ошибка: серийный номер не получен **\n**");
87. return;
88. }
89. cout << hex << (int)desc.bDeviceClass << "//" << hex << desc.idVendor << "//" << hex << desc.idProduct << "//";
91. printf("%s//", str\_man);
93. cout << endl;
95. **libusb\_close(handle);**
96. }

**Приложение 2. Результат работы программы**

