Библиотека LPOS_VFD_LCD.dll предназначена для настройки дисплея покупателя LPOS-LCD и LPOS-VFD.

Библиотека LPOS_VFD_LCD.dll является динамически подключаемой библиотекой (dll), предназначенной для работы под управлением ОС Windows95, “98, ME, NT4, Windows2000, Windows XP.

Описание библиотеки для настройки дисплея покупателя

Дисплеи покупателя LPOS- LCD и LPOS- VFD

Rev. 1-03/08

Содержание

1. ОБЩИЕ СВЕДЕНИЯ О БИБЛИОТЕКЕ

Библиотека **LPOS_VFD_LCD.dll** предназначена для обслуживания дисплеев покупателя LPOS-VFD-2029D и LPOS-LCD.

LPOS_VFD_LCD.dll – является динамически подключаемой библиотекой (dll), предназначенной для работы под управлением ОС Windows 98SE/ Me/ 2000/ XP.

Библиотека предоставляет такие сервисы:

- динамическая загрузка/выгрузка библиотеки;
- отправка данных на дисплей покупателя;
- установка интернационального шрифтового набора;
- установка режимов работы дисплея;
- установка курсора в указанную позицию и установка режима его отображения;
- установка/чтение битовой матрицы пользовательского символа;
- разрешение/запрет использования пользовательских шрифтов;
- отмечивание/разотмечивание отдельных символов из пользовательской таблицы символов.

2. ОПИСАНИЕ ПРОЦЕДУР И ФУНКЦИЙ БИБЛИОТЕКИ

Ниже приведены следующие процедуры и функции, которые экспортируются из библиотеки. При работе с библиотекой, её инициализировать не нужно. Процесс инициализации осуществляется **автоматически**.

2.1. ОТПРАВКА СИЛВОЛА НА ДИСПЛЕЙ

Для вывода символа на дисплей нужно использовать функцию *send_char*. За один раз выводится один символ. Экспортируемая функция send_char имеет такой формат:

Для C:

long WINAPI send_char (char data);

Для DELPHI:

Function send_char (data: byte):integer; StdCall;

где:
data – символ, который будет выведен на дисплей.

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.2. ОТПРАВКА HEX-КОДА СИМВОЛА НА ДИСПЛЕЙ

Для вывода hex-кода символа на дисплей нужно использовать функцию *send_hex*. За один раз выводится один символ. Экспортируемая функция send_hex имеет такой формат:

Для C:

long WINAPI send_hex (**char hex_data**);

Для DELPHI:

Function send_hex (hex_data:byte):integer; StdCall;

где:
hex_data – hex-код символа, который будет выведен на дисплей.

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.3. ВЫВОД СТРОКИ СИМВОЛОВ НА ДИСПЛЕЙ

Для вывода строки символов на дисплей нужно использовать функцию *send_str*. Экспортируемая функция *send_str* имеет следующий формат:

Для C:

long WINAPI send_str (void* str, char row, len);

Для DELPHI:

Function send_str(str: Pointer; row, len: byte):integer; StdCall;

где:
str – указатель на строку символов, которая будет выведена на дисплей,
row – выбор строки дисплея. 0 - первая строка, 1 - вторая,
len – длинна строки.

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.4. УСТАНОВКА КУРСОРА В УКАЗАННУЮ ПОЗИЦИЮ

Установка курсора в нужную позицию производится с помощью функции *moveXY*. Она перемещает курсор в положение с координатами [Row, Column], где Column – позиция столбца, Row – строки. Экспортируемая функция moveXY имеет такой формат:

Для C:

long WINAPI moveXY (**char Offs_X, char Offs_Y**);

Для DELPHI:

Function moveXY (Offs_X, Offs_Y: byte):integer; StdCall;

где:
Offs_X – позиция столбца, 0< **=Offs_X<=19**;
Offs_Y – позиция строки, 0< **=Offs_Y<=1**.

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.5. УСТАНОВКА КУРСОРА В НАЧАЛО ТЕКУЩЕЙ СТРОКИ

Перемещение курсора в крайнюю левую позицию текущей строки производится с помощью функции *home_str*. Экспортируемая функция home_str имеет такой формат:

Для C:

long WINAPI home_str (**void**);

Для DELPHI:

Function home_str ():integer; StdCall;

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.6. УСТАНОВКА КУРСОРА В КОНЕЦ ТЕКУЩЕЙ СТРОКИ

Перемещение курсора в крайнюю правую позицию текущей строки производится с помощью функции *end_str*. Экспортируемая функция end_str имеет такой формат:

Для C:

long WINAPI end_str(**void**);

Для DELPHI:

Function end_str ():integer; StdCall;

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.7. УСТАНОВКА КУРСОРА В НАЧАЛЬНУЮ ПОЗИЦИЮ

Перемещение курсора в верхнюю крайнюю левую позицию дисплея производится с помощью функции *home_all*. Экспортируемая функция home_all имеет такой формат:

Для C:

long WINAPI home_all (**void**);

Для DELPHI:

Function home_all ():integer; StdCall;

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.8. УСТАНОВКА КУРСОРА В КОНЕЧНУЮ ПОЗИЦИЮ

Перемещение курсора в нижнюю крайнюю правую позицию дисплея производится с помощью функции *end_all*. Экспортируемая функция end_all имеет такой формат:

Для C:

long WINAPI end_all (**void**);

Для DELPHI:

Function end_al ():integer; StdCall;

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.9. ОТОБРАЖАТЬ КУРСОР

Включение курсора производится с помощью функции *cursor_en*. Тип курсора – знак подчеркивания; режим - мигающий. Экспортируемая функция cursor_en имеет такой формат:

Для C:

long WINAPI cursor_en (**void**);

Для DELPHI:

Function cursor_en ():integer; StdCall;

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.10. СКРЫТЬ КУРСОР

Выключение курсора производится с помощью функции *cursor_dis*. Экспортируемая функция cursor_dis имеет такой формат:

Для C:

long WINAPI cursor_dis (**void**);

Для DELPHI:

Function cursor_dis ():integer; StdCall;

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.11. РЕЖИМ ПЕРЕЗАПИСИ

Функция *owerwrite* устанавливает дисплей в режим перезаписи. В этом режиме курсор будет перемещаться вправо, начиная с верхней крайней левой позиции. Когда курсор достигнет конца верхней строки, он переместится на нижнюю крайнюю левую позицию. Дойдя до конца нижней строки, курсор переместится в верхнюю крайнюю левую позицию, и будет перезаписывать предыдущие символы. Экспортируемая функция owerwrite имеет такой формат:

Для C:

long WINAPI owerwrite (**void**);

Для DELPHI:

Function owerwrite ():integer; StdCall;

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.12. РЕЖИМ ГОРИЗОНТАЛЬНОЙ ПРОКРУТКИ

Функция *horizontal* устанавливает дисплей в режим горизонтальной прокрутки. В этом режиме продолжительность перемещения курсора зависит от заданного предела, ограниченного для верхней строки. Когда курсор достигнет конца предела, символ, поступающий далее, будет подвигать уже отображённые символы на позицию влево. Экспортируемая функция horizontal имеет такой формат:

Для C:

long WINAPI horizontal (**void**);

Для DELPHI:

Function horizontal ():integer; StdCall;

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.13. РЕЖИМ ВЕРТИКАЛЬНОЙ ПРОКРУТКИ

Функция *vertical* устанавливает дисплей в режим вертикальной прокрутки. В этом режиме курсор будет перемещаться вправо, начиная с верхней крайней левой позиции. Когда курсор достигнет конца верхней строки, он переместится на нижнюю крайнюю левую позицию и будет двигаться до конца нижней строки. По достижении курсора конечной позиции, нижняя строка переместится на верхнюю позицию, а поле нижней строки очистится. Далее курсор будет передвигаться, начиная с первой позиции нижней строки. Экспортируемая функция vertical имеет такой формат:

Для C:

long WINAPI vertical (**void**);

Для DELPHI:

Function vertical ():integer; StdCall;

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.14. ОЧИСТКА ДИСПЛЕЯ

Удаление всех отображенных символов производится с помощью функции *clear_disp*. Экспортируемая функция clear_disp имеет такой формат:

Для C:

long WINAPI clear_disp (**void**);

Для DELPHI:

Function clear_disp ():integer; StdCall;

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.15. СЧИТЫВАНИЕ ДАННЫХ БИТОВОЙ МАТРИЦЫ СИМВОЛА

Считывание битовой матрицы символа производится функцией *get_sym*. Каждый байт представляет собой шрифтовой шаблон одного рядка, начиная с верхней части символа, причем младший бит младшего байта содержит пиксель верхнего правого угла символа. Экспортируемая функция get_sym имеет такой формат:

Для C:

long WINAPI get_sym (**void* bit_f, char font_val**);

Для DELPHI:

Function get_sym (bit_f: Pointer; font_val: byte):integer; StdCall;

где:

bit_f – указатель на буфер, куда будут помещены данные битовой матрицы символа (буфер должен иметь размер 7 байт);

font_val – символа, битовая матрица которого будет считана.

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.16. ЗАПИСЬ ДАННЫХ БИТОВОЙ МАТРИЦЫ СИМВОЛА

Определение заданного пользователем символа производится функцией *load_sym*. Каждый байт представляет собой шрифтовой шаблон одной строки, начиная с верхней части символа, причем младший бит младшего байта содержит пиксель верхнего правого угла символа. Экспортируемая функция load_sym имеет такой формат:

Для C:

long WINAPI load_sym (void* bit_f, char font_val);

Для DELPHI:

Function load_sym (bit_f: Pointer; font_val: byte):integer; StdCall;

где:
bit_f – указатель на буфер с данными битовой матрицы символа (буфер должен иметь размер 7 байт);
font_val – код символа, который будет перезаписан.

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.17. РАЗРЕШЕНИЯ ИСПОЛЬЗОВАНИЯ ВСЕХ СИМВОЛОВ ИЗ ПОЛЬЗОВАТЕЛЬСКОЙ ТАБЛИЦЫ СИМВОЛОВ

Для разрешения использования всех символов из пользовательской таблицы символов используется функции *en_user_char*. Экспортируемая функция en_user_char имеет такой формат:

Для C:

long WINAPI en_user_char (**void**);

Для DELPHI:

Function en_user_char ():integer; StdCall;

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.18. ЗАПРЕТ ИСПОЛЬЗОВАНИЯ ВСЕХ СИМВОЛОВ ИЗ ПОЛЬЗОВАТЕЛЬСКОЙ ТАБЛИЦЫ СИМВОЛОВ

Для запрета использования всех символов из пользовательской таблицы символов – используется функции *dis_user_char*. Экспортируемая функция dis_user_char имеет такой формат:

Для C:

long WINAPI dis_user_char (**void**);

Для DELPHI:

Function dis_user_char ():integer; StdCall;

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.19. РАЗРЕШЕНИЯ ИСПОЛЬЗОВАНИЯ СИМВОЛА ИЗ ПОЛЬЗОВАТЕЛЬСКОЙ ТАБЛИЦЫ СИМВОЛОВ

Для разрешения использования символа из пользовательской таблицы символов используется функции *en_sym*. Экспортируемая функция en_sym имеет такой формат:

Для C:

long WINAPI en_sym (char **font_val**);

Для DELPHI:

Function en_sym (font_val: byte):integer; StdCall;

где:
font_val – код символа, который будет разрешен.

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.20. ЗАПРЕТ ИСПОЛЬЗОВАНИЯ СИМВОЛА ИЗ ПОЛЬЗОВАТЕЛЬСКОЙ ТАБЛИЦЫ СИМВОЛОВ

Для запрета использования символа из пользовательской таблицы символов используется функции *dis_sym*. Экспортируемая функция dis_sym имеет такой формат:

Для C:

long WINAPI dis_sym (char **font_val**);

Для DELPHI:

Function dis_sym(font_val: byte):integer; StdCall;

где:
font_val – код символа, который будет запрещен.

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

2.21. УСТАНОВКА ИНТЕРНАЦИОНАЛЬНОГО ШРИФТА

Функция *font_set* предназначена для установки значений интернационального шрифта и интернациональной таблицы символов. Устройство позволяет установить постоянные значения (по умолчанию используются сохраненные параметры) и временные.

Для включённого дисплея можно задать иные значения для интернационального шрифта и интернациональной таблицы символов. Их можно будет применять до первого выключения дисплея; такие параметры являются временными. Чтоб использовать заданные значения постоянно – их нужно сохранить.

Параметры могут сохраняться в оперативной памяти, либо же в EEPROM устройства (electrically erasable/programmable read-only memory). Устройство запрограммировано таким образом, что если флаг записи в EEPROM не установлен, то при повторном включении устройства интернациональный шрифт и интернациональная таблица символов будут установлены в значения, сохраненные в EEPROM. Это происходит на аппаратном уровне.

Экспортируемая из библиотеки функция font_set имеет такой формат:

Для C:

long WINAPI font_set (char **font_lo**, char **font_hi**, char **EEPROM_en**);

Для DELPHI:

Function font_set(font_lo, font_hi, EEPROM_en: byte):integer; StdCall;

где:
font_lo – номер интернационального шрифтового набора для символов с кодами 20h- 7Fh;

font_hi – номер интернационального шрифтового набора для символов с кодами 80h- FFh;
EEPROM_en – флаг записи в EEPROM. Если он установлен, то номера таблиц символов сохраняются в EEPROM.

Интернациональный шрифтовой набор (20h - 7Fh):

n	Интернациональный шрифтовой набор	*n*	Интернациональный шрифтовой набор
0	АНГЛИЙСКИЙ (U.S.A)	8	ЯПОНСКИЙ
1	ФРАНЦУЗСКИЙ	9	НОРВЕЖСКИЙ
2	НЕМЕЦКИЙ	A	ДАТСКИЙ II
3	АНГЛИЙСКИЙ (U.K)	B	СЛАВЯНСКАЯ ГРУППА
4	ДАТСКИЙ I	C	РУССКИЙ
5	ШВЕДСКИЙ	D	Зарезервирован
6	ИТАЛЬЯНСКИЙ	E	Зарезервирован
7	ИСПАНСКИЙ	F	Зарезервирован

Интернациональный шрифтовой набор (80h - FFh):

n	**Интернациональный шрифтовой набор (80h - FFh)**
0	Page 0: PC437: английский (U.S.A), европейский стандарт
1	Page 1: Katakana: японский
2	Page 2: PC858: многоязычный
3	Page 3: PC860: португальский
4	Page 4: PC863: канадский, французский
5	Page 5: PC865: скандинавский
6	Page 6: русский
7	Page 7: славянская группа

Возвращаемые коды ошибок:

Код ошибки	Описание
ERROR_NO_ERROR	Завершение прошло успешно.
ERROR_FAILED	Ошибка связи с устройством.

3. ДЕМОНСТРАЦИОННАЯ ПРОГРАММА

Демонстрационная программа написана на языке Delphi7, демонстрирует работу всех возможностей библиотеки LPOS_VFD_LCD.dll.

ПРИЛОЖЕНИЕ 1. ФАЙЛ-ОПРЕДЕЛЕНИЙ ДЛЯ БИБЛИОТЕКИ

```
unit lpos_vfd_lcd;

// Описание функций экспортируемых из LPOS_VFD_LCD.dll

interface
Uses Windows;

Const
  ERROR_NO_ERROR         = 0;
  ERROR_FAILED       = 1;


  Function home_str():integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function end_str():integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function home_all():integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function end_all():integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function cursor_en():integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function cursor_dis():integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function moveXY(Offs_X, Offs_Y: byte):integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function owerwrite():integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function horizontal():integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function vertical():integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function send_hex(hex_data:byte):integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function clear_disp():integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function send_char(char: byte):integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function get_sym(bit_f: Pointer; font_val: byte):integer; StdCall; External
'LPOS_VFD_LCD.dll';
  Function load_sym(bit_f: Pointer; font_val: byte):integer; StdCall; External
'LPOS_VFD_LCD.dll';
  Function en_user_char():integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function dis_user_char():integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function en_sym(font_val: byte):integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function dis_sym(font_val: byte):integer; StdCall; External 'LPOS_VFD_LCD.dll';
  Function font_set(font_lo, font_hi, EEPROM_en: byte):integer; StdCall; External
'LPOS_VFD_LCD.dll';
  Function send_str(str: Pointer; row, len: byte):integer; StdCall; External
'LPOS_VFD_LCD.dll';
implementation

end.
```