Vacuum Fluorescent Display
Dynamic Link Library (DLL)
Exported Function
(Version 1.09)

www.sam4s.com

Revision A (Feb. 10, 2014)

Contents

VFD control DLL (VFD_DLL.dll)

You can control character type VFD module (2x20) from this.

In VFD_DLL.dll, setting values for VFD are like below metrics and they are fixed. User can select only serial port when using VFD module.

Default serial port for using VFD module	
QBIG series SPT-5000 series	**COM4**
SPT-3000 series	**COM5**
SPT-4000 series	**COM2**
SPT-4500 series	**COM5**
SPT-3700 series SPT-4700 series SPT-4740 series SPT-4800 series SPT-4840 series SPT-7000 series	**COM6**

Initial setting value	
Baud Rate	**9600 bps**
Data Bit	**8**
Parity Bit	**1**
Stop Bit	**1**

You can load or de-load dll library dynamically, and refer the way how to as below.

```
//how to load dll
hDll = LoadLibrary("VFD_DLL.dll");

//how to release dll
FreeLibrary(hDll);
```

Exported functions of DLL

VFD_DLL.dll has several interfaces. ALL OF THEM CAN BE CALLED IN THE CONDITION OF DLL LOADED.

PortOpen

long PortOpen(**char** PortName[]);

Return Value

TRUE

Open port is succeeded.

FALSE

Open port is failed. Port cannot be opened or is already opened.

Parameters

PortName

Serial port name for using VFD (COM1, COM2.. etc)

Remarks

We can set port open with this method. VFD can be set features; text clear (ClearText), overwriting mode (setOverwriteMode), and cursor hiding (setCursorOnOff) in PortOpen.

Example (Pseudo codes)

```
//how to define function pointer type
typedef long (*PortOpenFunc)();

//how to get function pointer pointed to function of dll
PortOpenFunc lpPortOpenFunc;
lpPortOpenFunc = (PortOpenFunc)GetProcAddress(hDll,"PortOpen");

//how to call function of dll
lpPortOpenFunc();
```

PortClose

long PortClose();

Return Value

TRUE

Close Port is succeeded.

FALSE

Close port is failed. Port cannot be closed or is not yet opened.

Parameters

None

Remarks

We can set port close with this method.

Example (Pseudo codes)

```
//how to define function pointer type
typedef long (*PortCloseFunc)();

//how to get function pointer pointed to function of dll
PortCloseFunc lpPortCloseFunc;
lpPortCloseFunc = (PortCloseFunc)GetProcAddress(hDll,"PortClose");

//how to call function of dll
lpPortCloseFunc();
```

ClearText

long ClearText();

Return Value

TRUE

Clear text is succeeded.

FALSE

Clear text is failed. This is occurred when port is not opened.

Parameters

None

Remarks

ClearText clears the current window to blanks, sets CursorRow and CursorColumn to zero, and resynchronizes the beginning of the window with the start of the viewport.

Example (Pseudo codes)

```
//how to define function pointer type
typedef long (*clearTextFunc)();

//how to get function pointer pointed to function of dll
clearTextFunc lpclearTextFunc;
lpclearTextFunc = (clearTextFunc)GetProcAddress(hDll,"clearText");

//how to call function of dll
lpclearTextFunc();
```

DisplayText

long DisplayText(**char** str[]);

Return Value

TRUE

Display text is succeeded.

FALSE

Display text is failed. This is occurred when port is not opened.

Parameters

str

The string of characters to display.

Remarks

The characters in str are processed beginning at the location specified by CursorRow and CursorColumn, and continue in succeeding character positions. Any previous data in a character position is overwritten, including character.

Example (Pseudo codes)

```
//how to define function pointer type
typedef long (*displayTextFunc)(char str[]);

//how to get function pointer pointed to function of dll
displayTextFunc lpdisplayTextFunc;
lpdisplayTextFunc = (displayTextFunc)GetProcAddress(hDll,"displayText");

//how to call function of dll
char* Temp = (char*)(const char*)str;
lpdisplayTextFunc(Temp);
```

DisplayTextAt

long displayTextAt(**long** posX, **long** posY, **char** str[]);

Return Value

TRUE

Function call is succeeded.

FALSE

Function call is failed. This is occurred when port is not opened.

Parameters

posX

The start row for the text. (Range 1~20)

posY

The start column for the text. (Range 1~2)

str

The string of characters to display.

Remarks

The characters in str are processed beginning at the window location specified by the posX and posY parameters, and continuing in succeeding columns. The line indicated by posY parameters will be cleared before displayTextAt method. The operational characteristics of the displayTextAt method are the same as the displayText method.

Example (Pseudo codes)

```
//how to define function pointer type
typedef long (*displayTextAtFunc)(long X , long Y, char str[]);

//how to get function pointer pointed to function of dll
displayTextAtFunc lpdisplayTextAtFunc;
lpdisplayTextAtFunc = (displayTextAtFunc)GetProcAddress(hDll,"displayTextAt");
```

```
//how to call function of dll
int Temp1 = _ttoi(numX);
int Temp2 = _ttoi(numY);
char* Temp3 = (char*)(const char*)str;
lpdisplayTextAtFunc(Temp1, Temp2, Temp3);
```

AddTextAt

long AddTextAt(long posX, **long** posY, **char** str[]);

Return Value

TRUE Function call is succeeded.

FALSE Function call is failed. This is occurred when port is not opened.

Parameters

posX The start row for the text. (Range 1~20)

posy The start column for the text. (Range 1~2)

str The string of characters to display.

Remarks

The characters in str are added beginning at the window location specified by the posX and posY parameters, and continuing in succeeding columns. The line indicated by posY parameters will not be cleared before displayTextAt method.

Example (Pseudo codes)

```
//how to define function pointer type
typedef long (*AddTextAtFunc)(long X , long Y, char str[]);

//how to get function pointer pointed to function of dll
AddTextAtFunc lpAddTextAtFunc;
lpAddTextAtFunc = (AddTextAtFunc)GetProcAddress(hDll,"AddTextAt");

//how to call function of dll
int Temp1 = _ttoi(numX);
int Temp2 = _ttoi(numY);
char* Temp3 = (char*)(const char*)str;
lpAddTextAtFunc(Temp1, Temp2, Temp3);
```

ClearDescriptors

long clearDescriptors();

Return Value

TRUE

Function call is succeeded.

FALSE

Function call is failed. This is occurred when port is not opened.

Parameters

None

Remarks

This function is used for turning off all descriptors.

Example (Pseudo codes)

```
//how to define function pointer type
typedef long (*ClsDescriptorFunc)();

//how to get function pointer pointed to function of dll
ClsDescriptorFunc lpClsDescriptorFunc;
lpClsDescriptorFunc = (ClsDescriptorFunc)GetProcAddress(hDll,"clearDescriptors");

//how to call function of dll
lpClsDescriptorFunc();
```

SetDescriptors

long setDescriptors(**long** data, **long** attribute);

Return Value

TRUE

Function call is succeeded.

FALSE

Function call is failed. This is occurred when port is not opened.

Parameters

Data

The row of the descriptor. The data parameter indicates which descriptor to change. (range 1~20)

Attribute

Function can turn the descriptor on/off (range 0~1) with attributes.

If attribute parameter is 0, the descriptor is off.

If attribute parameter is 1, the descriptor is on.

Remarks

Set the state of one of the descriptors, which are small indicators with a fixed label.

Example (Pseudo codes)

```
//how to define function pointer type
typedef long (*SetDescriptorFunc)(long data,long attribute);

//how to get function pointer pointed to function of dll
SetDescriptorFunc lpSetDescriptorFunc;
lpSetDescriptorFunc = (SetDescriptorFunc)GetProcAddress(hDll,"setDescriptor");

//how to call function of dll
int Temp1 = _ttoi(num);
lpSetDescriptorFunc(Temp1,DISP_SD_ON);
```

SetCursorOnOff

long setCursorOnOff(**long** attribute);

Return Value

TRUE

Function call is succeeded.

FALSE

Function call is failed. This is occurred when port is not opened.

Parameters

Attribute

If attribute parameter is 0, the cursor is off.

If attribute parameter is 1, the cursor is on.

Remarks

Set the state of cursor.

Example (Pseudo codes)

```
//how to define function pointer type
typedef long (*SetcuronFunc)(long attribute);

//how to get function pointer pointed to function of dll
SetcuronFunc lpSetcuronFunc;
lpSetcuronFunc = (SetcuronFunc)GetProcAddress(hDll,"setCursorOnOff");

//how to call function of dll
lpSetcuronFunc(1);
```

SetOverwriteMode

long setOverwriteMode();

Return Value

TRUE

Function call is succeeded.

FALSE

Function call is failed. This is occurred when port is not opened. Overwriting mode is set currently.

Parameters

None

Remarks

Selects overwrite mode as the screen display mode. In overwrite mode, entering a character code moves the cursor to the left end of the lower line when the cursor is at the right end of the upper line, and to the left end of the upper line when the cursor is at the right end of the lower line.

This mode is selected when the power is turned on. Selecting overwrite mode cancels horizontal or vertical scroll mode. Except when the cursor is at the right end, entering a character code moves the cursor one character to the right after displaying the character.

Example (Pseudo codes)

```
//how to define function pointer type
typedef long (*OverwritemodeFunc)();

//how to get function pointer pointed to function of dll
OverwritemodeFunc lpOverwritemodeFunc;
lpOverwritemodeFunc = (OverwritemodeFunc)GetProcAddress(hDll,"setOverwriteMode");

//how to call function of dll
lpOverwritemodeFunc();
```

SetVertiScrlMode

long setVertiScrlMode();

Return Value

TRUE

Function call is succeeded.

FALSE

Function call is failed. This is occurred when port is not opened. Current mode is Vertical scroll.

Parameters

None

Remarks

Selects vertical scroll mode as the screen display mode. In vertical scroll mode, entering a character code moves the cursor to the left end of the lower line when the cursor is at the right end of the upper line, scrolls the characters displayed on the lower line to the upper line, and clears the lower line when the cursor is at the right end of the lower line. At this time, the cursor is moved to the left end of the lower line.
Selecting vertical scroll mode cancels overwrite or horizontal scroll mode. Except when the cursor is at the right end, entering a character code moves the cursor one character to the right after displaying the character.

Example (Pseudo codes)

```
//how to define function pointer type
typedef long (*VertiscrlmodeFunc)();

//how to get function pointer pointed to function of dll
VertiscrlmodeFunc lpVertiscrlmodeFunc;
lpVertiscrlmodeFunc = (VertiscrlmodeFunc)GetProcAddress(hDll,"setVertiScrlMode");

//how to call function of dll
lpVertiscrlmodeFunc();
```

SetHoriScrlMode

long setHoriScrlMode();

Return Value

TRUE

Function call is succeeded.

FALSE

Function call is failed. This is occurred when port is not opened. Current mode is Horizontal scroll.

Parameters

None

Remarks

Selects horizontal scroll mode as the screen display mode. In horizontal scroll mode, entering a character code scrolls all displayed characters (including commas and periods) one character to the left, then displays the new character at the right end (when the cursor is at the right end of either line.)
Selecting horizontal scroll mode cancels overwrite or vertical scroll mode. Except when the cursor is at the right end, entering a character code moves the cursor one character to the right after displaying the character.

Example (Pseudo codes)

```
//how to define function pointer type
typedef long (*HoriscrlmodeFunc)();

//how to get function pointer pointed to function of dll
HoriscrlmodeFunc lpHoriscrlmodeFunc;
lpHoriscrlmodeFunc = (HoriscrlmodeFunc)GetProcAddress(hDll,"setHoriScrlMode");

//how to call function of dll
lpHoriscrlmodeFunc();
```