X-Cite®
Fluorescence Illumination • In Control

Software Development Kit

Updated: January 3rd, 2012

For Products:

X-Cite XLED1

Table of Contents

1.0 Overview

This software development kit describes the information required to communicate with the X-Cite devices. This document will describe the communication protocols for the various X-Cite devices. All X-Cite devices are either communicated with via the COM port or can be communicated with via a virtual COM port. As there are numerous ways to program a serial port, they will not all be described in this document. Microsoft has a number of examples in C++, Basic and .NET on their website. A basic outline of serial communications within Windows is found in the last section of this document.

1.1 Important Notes

It is important to note that when the virtual COM port is assigned a port number, this port number remains constant for the serial number of the device attached. For example, when an X-Cite XLED1 is connected, it may be assigned COM port 4. The COM port 4 will always be assigned to the X-Cite XLED1 with the same serial number. If another X-Cite XLED1 is connected with a different serial number, a new COM port number will be assigned.

2.0 X-Cite XLED1

The X-Cite XLED1 is connected to the computer via a USB cable. The drivers that are included with the XLED1 include a virtual COM port. The virtual or actual COM Port, as the case dictates, is to be set at 19,200 baud, 8 data bits, no parity and 1 stop bit. The X-Cite XLED1 has no hardware handshaking.

2.1 RS-232 Commands

Many of the commands discussed in this section result in an acknowledgement being sent from the X-Cite *XLED1* when the command has been received successfully by the X-Cite XLED1. Otherwise, an error message is sent by the X-Cite *XLED1*.

An acknowledgement has a packet structure as defined in the following table.

Byte	Alphanumeric Value	ASCII Value
0	'\r'	0x0D

An error message has a packet structure as defined in the following table.

Byte	Alphanumeric Value	ASCII Value
0	'e'	0x65
1	'\r'	0x0D

2.2 X-Cite XLED1 Command Set

Description	Command	Response
Connect	"co\r"	"\r" means that it has connected. "e\r" means that the computer is already connected or a partial command was received.

DisConnect	“dc\r”	“\r” means that it has disconnected. “e\r” means that it the computer is already disconnected.
Get LED Hours	“lh?\r”	“xxxxx,xxxxx,xxxxx,xxxxx\r” where xxxxx is the number of hours for the LED in the corresponding position. To determine which wavelength is in which position, use the LED type command.
Get Software Version	“sv?\r”	“x.y.z/x.y.z/x.y.z\r” where x is the major version, y is the minor version and z is the bug fix release level. The first set is XLED40 Controller, second is PWM, and third is XLEDCP.
Get Unit Status	“us?\r”	“xxx,xxx,xxx,xxx,yyyyy\r” where xxx is a number from 0 to 255 for the head status in the corresponding position. To determine which wavelength is in which position, use the LED type command. The status bits are: Bit 7 – Over Temperature Bit 6 – LED is present Bit 5 – LED was present at power on. Bit 4 – Current Alarm Bit 3 – Type Mismatch Bit 2 – Under Temperature Bit 1 – NVM Error Bit 0 – LED On/Off The “yyyyy” is the system status, it is a number from 0 to 65535. The system status bits are: Bit 15 – Reserved Bit 14 – Reserved Bit 13 – Reserved Bit 12 – System Performance Error Bit 11 – A/D Error Bit 10 – NVM Error Bit 9 – Touch Screen Present Bit 8 – PWM Module Present Bit 7 – Reserved Bit 6 – Touch Screen lock Bit 5 – Reserved Bit 4 – Single Shot/Continuous, 1 = single shot, 0 = continuous Bit 3 – 1 = one or more heads on, 0 = all off. Bit 2 – Reserved Bit 1 – Light guide sensor Bit 0 – Alarm on/off
Turn LED Off	“of?\r”	“w,x,y,z\r” where each of the letters is either a 0 or a 1, 0 means the LED is off, 1 means the LED is on. To determine which wavelength is on or off, use the LED type command.

	"of=w,x,y,z\r" Where w,x,y,z are the head numbers to turn off. The off command works by reading the LED number to turn off, separated by commas. Send only the head numbers you wish to turn off. E.g. Send "of=4,2\r" to turn off LED's 2 and 4. To determine which wavelength is LED 2 and 4 see the get LED type command.	"\r" if the command was accepted, "e\r" if the command was rejected.
Turn all LED's Off	"of=a\r"	"\r" if the command was accepted, "e\r" if the command was rejected.
Turn LED On	"on?\r"	"w,x,y,z\r" where each of the letters is either a 0 or a 1, 0 means the LED is off, 1 means the LED is on. To determine which wavelength is on or off, use the LED type command.
	"on=w,x,y,z\r" Where w,x,y,z are the head numbers to turn on. The on command works by reading the LED number to turn on, separated by commas. Send only the head numbers you wish to turn on. E.g. Send "of=1,3\r" to turn on LED's 1 and 3. To determine which wavelength is LED 1 and 3 see the get LED type command.	"\r" if the command was accepted, "e\r" if the command was rejected.
Turn all LED's On	"on=a\r"	"\r" if the command was accepted, "e\r" if the command was rejected.
Clear Alarm	"ca\r"	"\r" when the alarm is cleared. "e\r" on an invalid command returned.
Lock Front Panel	"lo?\r"	"x\r" where x is the lock status. A 1 means the front panel is locked, 0 means the front panel is not locked.
	"lo\r" will lock the front panel.	"\r" to indicate the command has been discontinued.
UnLock Front Panel	"ul?\r"	"x\r" where x is the lock status. A 1 means the front panel is locked, 0 means the front panel is not locked.
	"ul\r" will unlock the front panel.	"\r" to indicate the command has been accepted. To be discontinued, will be implemented with lock command in future versions.
Get Unit Serial Number	"sn?\r"	"xxxxx\r" where xxxxx is the serial number of the unit.
Get or Set Intensity	"ip?\r"	"wwww,xxxx,yyyy,zzzz\r" where each four letter case is the intensity percentage from 0000 (0.0%) to 1000 (100.0%). To determine the percentage for each wavelength, use the LED type command.

	“ip=wwww,xxxx,yyyy,zzzz\r” where each four letters represents the intensity to be set in 0.1% increments. To determine which letter is for which wavelength, use the LED type command. To set the intensity for one LED to 25.5%, say the third one, send “ip=,,255\r”. The minimum intensity level is 5.0% or 50. A level of 0% will also be accepted.	“\r” if the command was accepted.
Internal Signal Generator, delay time.	“dt?\r”	“wwwww,xxxxx,yyyyy,zzzzz\r” where each five letters represents the signal delay time from 0 to 65535.
	“dt=wwwww,xxxxx,yyyyy,zzzzz\r” where each five letters represents the signal delay time. To determine which letter is for which wavelength, use the LED type command. To set the delay time for one LED, say the second one, send “dt=,390\r”.	“\r” if the command was accepted.
Internal Signal Generator, on time.	“ot?\r”	“wwwww,xxxxx,yyyyy,zzzzz\r” where each five letters represents the signal on time from 0 to 65535.
	“ot=wwwww,xxxxx,yyyyy,zzzzz\r” where each five letters represents the signal on time. To determine which letter is for which wavelength, use the LED type command or LED get name command. To set the on time for one LED, say the third one, send “ot=,,80\r”.	“\r” if the command was accepted.
Internal Signal Generator, off time.	“ft?\r”	“wwwww,xxxxx,yyyyy,zzzzz\r” where each five letters represents the signal off time from 0 to 65535.
	“ft=wwwww,xxxxx,yyyyy,zzzzz\r” where each five letters represents the signal off time. To determine which letter is for which wavelength, use the LED type command or LED get name command. To set the off time for one LED, say the first one, send “ft=5\r”.	“\r” if the command was accepted.
Internal Signal Generator, trigger advance time.	“tt?\r”	“wwwww,xxxxx,yyyyy,zzzzz\r” where each five letters represents the signal delay time from -32767 to 32767.
	“tt=wwwww,xxxxx,yyyyy,zzzzz\r” where each five letters represents the trigger delay time. To determine which letter is for which wavelength, use the LED type command or LED. To set the trigger time for one LED, say the fourth one, send “tt=,,,1000\r”.	“\r” if the command was accepted.

Internal PWM Start/Stop and Status	“is\r”	“x\r” where if x is 0 it means internal PWM is not running. If x is 1, internal PWM is running.
	“is=x\r” sets the internal PWM status. If x is 0 it means internal PWM is not running. If x is 1, internal PWM is running.	“\r” if the command is accepted.
Single Shot or Continuous	“sc?\r”	“x\r” where x is a 0 or 1. A 0 means the internal PWM generator is in continuous mode. A 1 means the internal PWM generator is in single shot mode.
	“sc=x\r” where x is the desired setting. If x is 0, the internal PWM will run continuously. If x is 1, the internal PWM will do a single shot.	“\r” if the number is accepted.
Set PWM units	“su?\r”	“w,x,y,z\r” where each letter represents an LED. Each letter can be a 0, 1 or 2. A 0 is uS, a 1 is mS, a 2 is S. NOTE: If the units are in microseconds, the returned numbers for delay, on, off and trigger times are in units of 10 uS.
	“su=w,x,y,z\r” where each letter represents an LED. Each letter can be a 0, 1 or 2. A 0 is uS, a 1 is mS, a 2 is S. NOTE: If the units are in microseconds, the returned numbers for delay, on, off and trigger times are in units of 10 uS.	“\r” if the units are accepted.
Internal/External Generator and pulse mode	“pm?\r”	“x\r” where x is the mode of the LED pulse control. The modes are as follows: 0: No Pulse control. 1: Pulse mode, internal generator 2: Pulse mode, external generator 3: Pulse mode, global external generator
	“pm=x\r” where x is the desired pulse generation mode. The modes are as follows: 0: No Pulse control. 1: Pulse mode, internal generator 2: Pulse mode, external generator 3: Pulse mode, global external generator	“\r” if the command is accepted.
LCD Screen	“ss?\r”	“xx\r” where x is a number from 2 to 13.
LCD Brightness	“lb?\r”	“xxx\r” where xxx is the LCD brightness from 0 to 100%. The number returned will be from 0 to 255 indicating a 0 to 100% range.
	“lb=xxx\r” where xxx is the desired LCD brightness from 0 to 100%. Send a number from 0 to 255 which covers the 0 to 100% range.	“\r” if the command is accepted.
Get LED Temperature	“gt?\r”	“www,xxx,yyy,zzz\r” where each three letter character set is the temperature value, this temperature is in ℃.

Get LED Serial Number	“ls?\r”	“wwwww,xxxxx,yyyyy,zzzzz\r” where each five letter character set is a serial number. To determine which serial number is to which LED, use the get LED type command.
Get LED Type	“lt?\r”	“www,xxx,yyy,zzz\r” where each three characters is the type of LED installed in the system. For example, to get the type for LED 1, call this command. If www is 001, the LED type is 1. Refer to LED type numbers and definitions in the manual.
Get LED WaveLength	“lw?\r”	“www,xxx,yyy,zzz\r” where each three characters is the wavelength for the LED’s installed in the system. The order will be the same as to all commands that referenced this one. For example, to get the intensity for the 365nm head, call this command. If www is 365, the intensity command will return the 365nm intensity before the first comma.
Get LED Full Width Half Maximum Value	“lf?\r”	“www,xxx,yyy,zzz\r” where each three characters is the FWHM of LED installed in the system. For example, to get the FWHM for LED 1, call this command. If www is 001, the LED FWHM is 1nm.
Get LED Maximum temperatures	“mt?\r”	“ww,xx,yy,zz\r” where each two characters is the maximum allowed temperature of the LED installed in the system. In degrees C.
Get LED Minimum temperatures	“nt?\r”	“ww,xx,yy,zz\r” where each two characters is the minimum allowed temperature of the LED installed in the system. In degrees C.
Screen Saver	“st?\r”	“xxxxx\r” where xxxxx is the screen saver timeout. Setting this value to 0 will disable the LCD timeout.
	“st=xxxxx\r” Where xxxxx is the desired screen saver timeout in seconds. A zero timeout means the LCD is always on.	“\r” if the command is accepted.
LED Mfg Date	“md?\r”	“mm/dd/yy,mm/dd/yy,mm/dd/yy,mm/dd/yy\r” where each date is in order of the LED’s in the system. Year is a two digit year, insert 20 before the yy to get the current year.
LED Temperature Hysteresis	“th?\r”	“www,xxx,yyy,zzz\r” where each three letters is the temperature hysteresis for the corresponding LED in degrees Celsius.
Get LED Name	“ln?\r”	“<name1>,<name2>,<name3>,<name4>\r” where each name is up to eight characters for each of the current LED names.
Speaker Volume	“vo?\r”	“xxx\r” where the x’s are the speaker volume from 0 to 255 which is scaled from 0 to 100% volume.
	“vo=xxx\r” where xxx is the desired speaker volume from 0 to 255. The 0 to 255 is scaled from 0 to 100% volume.	“\r” if the command is accepted.
Minimum LED Pulse Width	“mw?\r”	“www,xxx,yyy,zzz\r” where each three letters is the corresponding minimum LED pulse width in units of 10 uS.
Commands only supported in XLED40 Software Version 1.2 or later		

Set LED High Speed Mode	"hs=w,x,y,z\r" where each character is the high speed enabled setting. Set LED to 1 for it to respond to high speed USB commands, 0 for not to respond to USB high speed commands.	"\r" if the command is accepted. NOTE: LED's must be off for it to be put into USB high speed command mode.
Get LED High speed Mode setting	"hs?\r"	"w,x,y,z\r" where each letter represents the high speed mode for each LED. A 1 means the LED is in high speed mode, a 0 means it is not in USB high speed mode.
Turn off or on LEDs	This is a bitwise command separated by nibble. All examples will be in hexadecimal format. Send a 0x8Y to set the LED on/off status where Y represents the on/off state. So a 0x81 will turn on LED 1 and off LED 2,3,4. Sending a 0x8F will turn on all LED's, sending a 0x80 will turn off all LEDs. The byte is formatted as: <b7> Extended Command <b6> 0 for on off command <b5> Set to 0 <b4> Set to 0 <b3> LED 4 on/off <b2> LED 3 on/off <b1> LED 2 on/off <b0> LED 1 on/off	Responds with the same byte that was sent if accepted. If bit 4 is a 1 in the reply packet, an error occurred changing the LED state.
Set LED intensity at high speed.	This is a bitwise command separated by nibble. All examples will be in hexadecimal format. Send a 0xCY to set the LED that will get the new intensity. So a 0xC1 will set the intensity for LED 1. Sending a 0xCF will set the intensity for all LED's. The byte is formatted as: <b7> Extended Command <b6> 1 for intensity command <b5> Set to 0 <b4> Set to 0 <b3> LED 4 on/off <b2> LED 3 on/off <b1> LED 2 on/off <b0> LED 1 on/off Second byte is the intensity in 1% increments with the MSB set, add 0x80 to the value. To set the LEDs to 50%, send two bytes as <0xCF><0xB2>.	Responds with the same bytes that were sent.

3.0 Windows Serial Port Programming Overview

Programming serial ports in Windows is done through the file interface. The examples listed below are using Microsoft Visual C++. To open a serial port in windows it is the same as creating a file. The filename however is the system path plus the COM port name you wish to open. For example:

```
sprintf(szComPort,"\\\\.\\COM%d",cpPort);
```

Where cpPort is the port number you wish to open.

Then to open the communications port, open a file, for example:

```
if((ComPort=CreateFile(szComPort,GENERIC_READ|GENERIC_WRITE,0,NULL,OPEN_EXIST
ING, 0, 0)) == INVALID_HANDLE_VALUE) {
      return(FALSE);
} else {
      return(TRUE);
}
```

Then, set the communication timeouts, use the COMMTIMEOUTS data structure and call the SetCommTimeouts function. Once the timeouts are set, set the communication timing parameters using the BuildCommDCB function. For example:

```
sprintf(szBuf,"COM%d: baud=%d parity=%c data=%d stop=%d to=on xon=off dsr=off
octs=off dtr=off rts=off idsr=off",cpPort,nBaud,cParity,nDataBit,nStopBit);

if(!BuildCommDCB(szBuf, &dcb)) {
      return(FALSE);
}
```

Once the communication parameters have been set, you may want to set the communication state and event triggers using the SetCommState and SetCommMask functions. These functions are not required; however since X-Cite devices use CR at the end of all communication strings you may want to trigger a function on this byte. One other useful function is PurgeComm, to clear out the transmit and receiving buffers.
To transmit data, it is the same as writing to a file, just call the WriteFile function, example:

```
WriteFile(ComPort,&szBuf,nLength,&dwCount,NULL);
```

To receive data, first check to see how much data has been received, you can use the ClearCommError function to obtain the current status and how many bytes have been received. For example:

```
ClearCommError(ComPort,&dwEvent,&ComStat);
```

The ComStat structure has a cbInQue member that contains the number of received bytes. You can then use the ReadFile function to obtain the data, for example:

```
ReadFile(ComPort,&c, 1, &dwCount, NULL);
```

Where the created pointer to c is a character buffer and the 1 specifies the length of c. You may want to do this for byte by byte processing or you can specify a longer buffer and read more of the data in one function call.

When you are done with communicating, it is best to close the serial port by closing the handle, for example:

```
CloseHandle(ComPort);
```

The examples provided are enough to perform basic windows serial port communication. There are numerous methods for using these functions and implementing a serial port class. It is best to use the

Microsoft developer network resources for serial port programming and to obtain specific examples for the language you choose to use.