

LR-USB USB Communication
Sample Program
(Excel Macro VBA64-bit Version)

Description

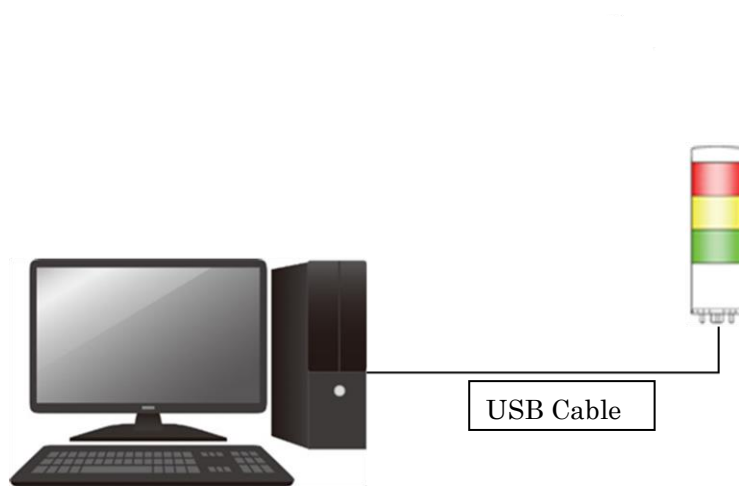
LR-USB USB Communication Sample Program (Excel Macro VBA 64bit).....	1
1. Overview	3
1.1. System Overview	3
2. Development environment.....	3
2.1. Windows Requirements	3
3. Sample Source Overview	4
3.1. Screen Operation	4
3.1.1. Command List	4
3.1.2. LED Unit Control	5
3.1.3. Control Several LED Units	5
3.1.4. Alarm Controlled by Alarm Pattern.....	5
3.1.5. Alarm Control by Alarm Pattern and Scale	6
3.1.6. Reset	7

1. Overview

This is an outline of sample programming to control LR-USB via USB communication. The programs are intended to control the unit using Excel Macro VBA without the use of DLLs provided by PATLITE.

1.1. System Overview

The system configuration diagram of this program is as follows. The sample program controls one LR-USB by USB communication



2. Development environment

The development environment of the sample program is shown below.

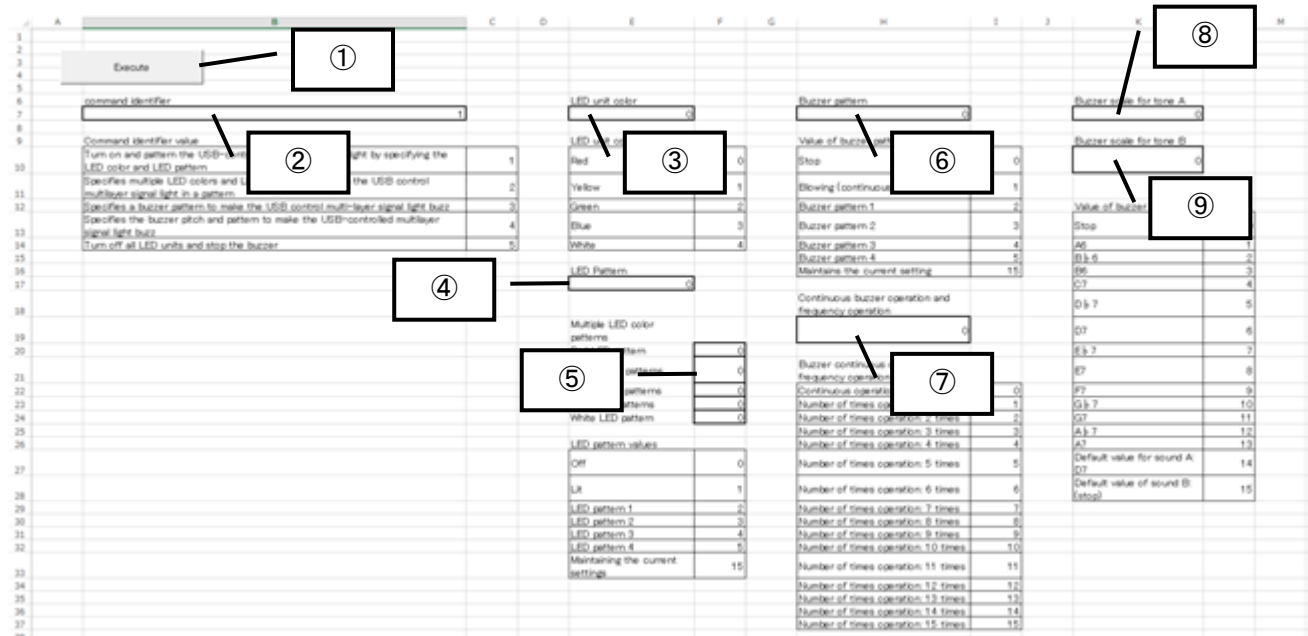
2.1. Windows Requirements

Development Environment		Remarks
Development OS	Windows10	
Development Language	VBA	Excel 2010 (64-bit version)

3. Sample Sorce Overview

3.1. Screen Operation

In Excel, the command for each action is executed by clicking the Execute button with the identifier of the command to be executed and the parameters to be used are specified.



No.	Item Name	Description
①	Execute Button	Executes a command with the specified command identifier
②	Command Identifier	Select the command identifier to execute
③	LED Unit Color	Set the LED color to be controlled
④	LED Pattern	Set Continuous On Patterns for LEDs
⑤	Multiple LED Color Patterns	Set Continuous On Pattern for Red, Amber, Green, Blue, and White LEDs
⑥	Alarm Pattern	Set the alarm sounding pattern
⑦	Alarm Continuous Operation and Number of Cycles	Set the number of times the alarm sounds
⑧	Sound A Alarm Scale	Set Alarm scale for Sound A
⑨	Sound B Alarm Scale	Set Alarm scale for Sound B

3.1.1. Command List

Command Name	Description
Controls LED Unit	Set LED color and LED pattern to display and activate it.
Controls Several LED Units	Set multiple LED colors and LED patterns to display and activate them.

Alarm Controlled by Alarm Pattern	Set alarm pattern and activate it.
Alarm Controlled by Alarm Pattern and Scale	Set alarm scale and pattern and activate it.
Reset	Turn off all LED units and stop the alarm.

3.1.2. LED Unit Control

Set the following values in Excel and then press the Execute button to execute the command.

No.	設定パラメータ	値
1	Command ID	1
2	LED Unit Color	Red: 0 Amber: 1 Green: 2 Blue: 3 White: 4
3	LED Pattern	Off: 0 Continuous On: 1 LED Pattern1: 2 LED Pattern2: 3 LED Pattern3: 4 LED Pattern4: 5 No change: 15

3.1.3. Control Several LED Units

Set the following values in Excel and then press the Execute button to execute the command.

No.	Configuration Parameters	Value
1	Command ID	2
2	Red LED Pattern	Off: 0
3	Amber LED Pattern	Continuous On: 1
4	Green LED Pattern	LED Pattern1: 2
5	Blue LED Pattern	LED Pattern2: 3
6	White LED Pattern	LED Pattern3: 4 LED Pattern4: 5 No change: 15

3.1.4. Alarm Controlled by Alarm Pattern

Set the following values in Excel and then press the Execute button to execute the command.

No.	Configuration Parameters	Value
1	Command ID	3

2	Alarm Pattern	Stop: 0 Sounding (Continuous): 1 Alarm Pattern 1: 2 Alarm r Pattern 2: 3 Alarm r Pattern 3: 4 Alarm Pattern 4: 5 No change: 15
3	Alarm Continuous Operation and Number of Cycles	Continuous operation: 0 Number of cycles: 1 to 15

3.1.5. Alarm Control by Alarm Pattern and Scale

Set the following values in Excel and then press the Execute button to execute the command.

No.	Configuration Parameters	Value
1	Command ID	4
2	Alarm Pattern	Stop: 0 Sounding (Continuous): 1 Alarm Pattern 1: 2 Alarm r Pattern 2: 3 Alarm r Pattern 3: 4 Alarm Pattern 4: 5 No change: 15
3	Alarm Continuous Operation and Number of Cycles	Continuous operation: 0 Number of cycles: 1 to 15
4	Sound A Alarm Scale	Stop: 0
5	Sound B Alarm Scale	A6: 1 Bb6: 2 B6: 3 C7: 4 Db7: 5 D7: 6 Eb7: 7 E7: 8 F7: 9 Gb7: 10 G7: 11 Ab7: 12 A7: 13 Default value of sound A: D7: 14

		Default value of sound B: (Stop): 15
--	--	--------------------------------------

3.1.6. Reset

Set the following values in Excel and then press the Execute button to execute the command.

No.	Configuration Parameters	Value
1	Command ID	5

3.2. 関数定義

3.2.1. 関数一覧

関数名	説明
UsbOpen	USB 制御積層信号灯へ USB 通信で接続する
UsbClose	USB 制御積層信号灯との USB 通信を終了するコマンドを送信
SendCommand	コマンドを送信する
SetLight	LED 色と LED パターンを指定して USB 制御積層信号灯を点灯、パターン点灯させる
SetTower	複数の LED 色と LED パターンを指定して USB 制御積層信号灯をパターン点灯させる
SetBuz	ブザーのパターンを指定して USB 制御積層信号灯をブザー吹鳴させる
SetBuzEx	ブザーの音階とパターンを指定して USB 制御積層信号灯をブザー吹鳴させる
Reset	LED ユニットのすべてを消灯し、ブザーを停止させる

3.2.2. UsbOpen 関数

関数名	Private Function UsbOpen() As Long	
パラメータ	なし	
戻り値	Long	成功:0、失敗:0 以外
説明	ベンダーID が「0x191A」とデバイスが「0x8003」のデバイスをオープンする	
関数の使用方法	<pre>' メイン関数 Sub Run_Click() ' USB 制御積層信号灯へ USB 通信で接続する Dim dwRet As Long dwRet = UsbOpen() If dwRet <> 0 Then Exit Sub End Sub</pre>	
備考	関数のプログラムの概要は「4.1LR-USB に接続」を参照	

3.2.3. UsbClose 関数

関数名	Private Sub UsbClose()	
パラメータ	なし	
戻り値	なし	
説明	LR-USB との USB 通信を終了する ※プログラム終了時に自動的に開放されるため、実装なし	
関数の使用方法	<pre>' メイン関数 Sub Run_Click() ' USB 制御積層信号灯へ USB 通信で接続する Dim dwRet As Long dwRet = UsbOpen() If dwRet <> 0 Then Exit Sub ' USB 制御積層信号灯との USB 通信を終了する UsbClose() End Sub</pre>	
備考	関数のプログラムの概要は「4.2LR-USB との切断」を参照	

3.2.4. SendCommand 関数

関数名	Private Function SendCommand(ByRef sendData() As Byte, ByVal sendLength As Long) As Long	
パラメータ	ByRef sendData() As Byte	送信データ
	ByVal sendLength As Long	送信データサイズ
戻り値	Long	成功:0、失敗:0 以外
説明	接続した LR-USB にデータを送信する	
関数の使用方法	<pre> ' メイン関数 Sub Run_Click() ' USB 制御積層信号灯へ USB 通信で接続する Dim dwRet As Long dwRet = UsbOpen() If dwRet <> 0 Then Exit Sub ' 送信データを作成 Dim sendData(SEND_BUFFER_SIZE - 1) As Byte sendData(0) = &H0 sendData(1) = &H0 sendData(2) = &H0 sendData(3) = &H0 sendData(4) = &H0 sendData(5) = &H0 sendData(6) = &H0 Dim ret As Long ret = SendCommand(sendData, UBound(sendData) + 1) If ret <> 0 Then Debug.Print ("failed to send data") Exit Function End If ' USB 制御積層信号灯との USB 通信を終了する UsbClose() End Sub </pre>	
備考	関数のプログラムの概要は「4.3 コマンドを送信」を参照	

3.2.5. SetLight 関数

関数名	Private Function SetLight(ByVal color As Byte, ByVal state As Byte) As Long	
パラメータ	ByVal color As Byte	制御する LED 色(赤:0、黄:1、緑:2、青:3、白:4)
	ByVal state As Byte	LED パターン(消灯:0、点灯:0x1、LED パターン 1:0x2、LED パターン 2:0x3、LED パターン 3:0x4、LED パターン 4:0x5、現状の設定を維持:0x6~0xF)
戻り値	Long	成功:0、失敗:0 以外
説明	LED 色と LED パターンを指定して USB 制御積層信号灯を点灯、パターン点灯させるブザーおよび、指定された LED 色以外の LED ユニットは現在の状態を維持する	
関数の使用方法	<pre> ' メイン関数 Sub Run_Click(): ' USB 制御積層信号灯へ USB 通信で接続する Dim dwRet As Long dwRet = UsbOpen() If dwRet <> 0 Then Exit Sub ' 赤色のユニットを点灯 dwRet = SetLight(LED_COLOR_RED, LED_ON) ' USB 制御積層信号灯との USB 通信を終了する UsbClose() End Sub </pre>	
備考	関数のプログラムの概要は「4.4LED 色と LED パターンを指定したコマンドの送信」を参照	

3.2.6. SetTower 関数

関数名	Private Function SetTower(ByVal red As Byte, ByVal yellow As Byte, ByVal green As Byte, ByVal blue As Byte, ByVal white) As Long	
パラメータ	ByVal red As Byte	赤の LED パターン(消灯:0、点灯:0x1、LED パターン 1:0x2、LED パターン 2:0x3、LED パターン 3:0x4、LED パターン 4:0x5、現状の設定を維持:0x6~0xF)
	ByVal yellow As Byte	黄の LED パターン(消灯:0、点灯:0x1、LED パターン 1:0x2、LED パターン 2:0x3、LED パターン 3:0x4、LED パターン 4:0x5、現状の設定を維持:0x6~0xF)
	ByVal green As Byte	緑の LED パターン(消灯:0、点灯:0x1、LED パターン 1:0x2、LED パターン 2:0x3、LED パターン 3:0x4、LED パターン 4:0x5、現状の設定を維持:0x6~0xF)
	ByVal blue As Byte	青の LED パターン(消灯:0、点灯:0x1、LED パターン 1:0x2、LED パターン 2:0x3、LED パターン 3:0x4、LED パターン 4:0x5、現状の設定を維持:0x6~0xF)
	ByVal white As Byte	白の LED パターン(消灯:0、点灯:0x1、LED パターン 1:0x2、LED パターン 2:0x3、LED パターン 3:0x4、LED パターン 4:0x5、現状の設定を維持:0x6~0xF)
戻り値	Long	成功:0、失敗:0 以外
説明	複数の LED 色と LED パターンを指定して USB 制御積層信号灯をパターン点灯させる	
関数の使用方法	<pre> ' メイン関数 Sub Run_Click(): ' USB 制御積層信号灯へ USB 通信で接続する Dim dwRet As Long dwRet = UsbOpen() If dwRet <> 0 Then Exit Sub ' 赤色のユニット:パターン 1 ' 黄色のユニット:パターン 3 ' 緑色のユニット:パターン 4 ' 青色のユニット:点灯 ' 白色のユニット:消灯 dwRet = SetTower(LED_PATTERN1, LED_PATTERN3, LED_PATTERN4, LED_ON, LED_OFF) ' USB 制御積層信号灯との USB 通信を終了する UsbClose() End Sub </pre>	
備考	関数のプログラムの概要は「4.5 複数の LED 色と LED パターンを指定したコマンドの送信」を参照	

3.2.7. SetBuz 関数

関数名	Private Function SetBuz(ByVal buz_state As Byte, ByVal limit As Byte) As Long	
パラメータ	ByVal buz_state As Byte	ブザーパターン(停止:0x0、吹鳴(連続):0x1、ブザーパターン 1:0x2、ブザーパターン 2:0x3、ブザーパターン 3:0x4、ブザーパターン 4:0x5、現状の設定を維持:0xF)
	ByVal limit As Byte	連続動作:0、回数動作:1~15
戻り値	Long	成功:0、失敗:0 以外
説明	ブザーのパターンを指定して USB 制御積層信号灯をブザー吹鳴させる LED ユニットは現在の状態を維持する。音階はデフォルト値で動作する	
関数の使用方法	<pre> ' メイン関数 Sub Run_Click(): ' USB 制御積層信号灯へ USB 通信で接続する Dim dwRet As Long dwRet = UsbOpen() If dwRet <> 0 Then Exit Sub ' ブザーをパターン 1 で、2 秒吹鳴 dwRet = SetBuz(BUZZER_PATTERN1, 2) ' USB 制御積層信号灯との USB 通信を終了する UsbClose() End Sub </pre>	
備考	関数のプログラムの概要は「4.6 ブザーのパターン指定したコマンドの送信」を参照	

3.2.8. SetBuzEx 関数

関数名	Private Function SetBuzEx(ByVal buz_state As Byte, ByVal limit As Byte, ByVal pitch1 As Byte, ByVal pitch2 As Byte) As Long	
パラメータ	ByVal buz_state As Byte	ブザーパターン(停止:0x0、吹鳴(連続):0x1、ブザーパターン 1:0x2、ブザーパターン 2:0x3、ブザーパターン 3:0x4、ブザーパターン 4:0x5、現状の設定を維持:0xF)
	ByVal limit As Byte	連続動作:0、回数動作:1~15
	ByVal pitch1 As Byte	音 A のブザー音階(停止:0x0、A6:0x1、B♭6:0x2、B6:0x3、C7:0x4、D♭7:0x5、D7:0x6、E♭7:0x7、E7:0x8、F7:0x9、G♭7:0xA、G7:0xB、A♭7:0xC、A7:0xD、音 A のデフォルト値:D7:0xE、音 B のデフォルト値:(停止):0xF)
	ByVal pitch2 As Byte	音 B のブザー音階(停止:0x0、A6:0x1、B♭6:0x2、B6:0x3、C7:0x4、D♭7:0x5、D7:0x6、E♭7:0x7、E7:0x8、F7:0x9、G♭7:0xA、G7:0xB、A♭7:0xC、A7:0xD、音 A のデフォルト値:D7:0xE、音 B のデフォルト値:(停止):0xF)
戻り値	Long	成功:0、失敗:0 以外
説明	ブザーの音階とパターンを指定して USB 制御積層信号灯をブザー吹鳴させる	
関数の使用方法	<pre> ' メイン関数 Sub Run_Click(): ' USB 制御積層信号灯へ USB 通信で接続する Dim dwRet As Long dwRet = UsbOpen() If dwRet <> 0 Then Exit Sub ' ブザーをパターン 1 で、2 秒吹鳴 ' 音 A:B♭6 ' 音 B:E7 dwRet = SetBuzEx(BUZZER_PATTERN1, 2, BUZZER_PITCH2, BUZZER_PITCH8) ' USB 制御積層信号灯との USB 通信を終了する UsbClose() End Sub </pre>	
備考	関数のプログラムの概要は「4.7 ブザーのパターンと音階を指定したコマンドの送信」を参照	

3.2.9. Reset 関数

関数名	Private Function Reset() As Long	
パラメータ	なし	
戻り値	Long	成功:0、失敗:0 以外
説明	LED ユニットをすべて消灯し、ブザーを停止させる	
関数の使用方法	<pre>' メイン関数 Sub Run_Click(): ' USB 制御積層信号灯へ USB 通信で接続する Dim dwRet As Long dwRet = UsbOpen() If dwRet <> 0 Then Exit Sub ' LR-USB の状態をリセット dwRet = Reset() ' USB 制御積層信号灯との USB 通信を終了する UsbClose() End Sub</pre>	
備考	関数のプログラムの概要は「4.8 リセットコマンドの送信」を参照	

3.3. 定数定義

3.3.1. ベンダーID

定数名	値	説明
VENDOR_ID	0x191A	LR-USB のベンダーID

3.3.2. デバイスID

定数名	値	説明
DEVICE_ID	0x8003	LR-USB のデバイス

3.3.3. コマンドバージョン

定数名	値	説明
COMMAND_VERSION	0x00	LR-USB へコマンドを送信する時のコマンドバージョン

3.3.4. コマンド ID

定数名	値	説明
COMMAND_ID	0x00	LR-USB へコマンドを送信する時のコマンド ID

3.3.5. ホスト→USB 制御積層信号灯に送信するためのエンドポイントアドレス

定数名	値	説明
ENDPOINT_ADDRESS	1	PC から LR-USB へ送信するためのエンドポイント

3.3.6. コマンド送信時のタイムアウト時間

定数名	値	説明
SEND_TIMEOUT	1000	コマンドを送信する時のタイムアウト時間 単位はミリ秒

3.3.7. プロトコルデータ領域サイズ

定数名	値	説明
SEND_BUFFER_SIZE	8	送信するデータのバッファサイズ

3.3.8. LED ユニット色

定数名	値	説明
LED_COLOR_RED	0	赤
LED_COLOR_YELLOW	1	黄
LED_COLOR_GREEN	2	緑

LED_COLOR_BLUE	3	青
LED_COLOR_WHITE	4	白

3.3.9. LED パターン

定数名	値	説明
LED_OFF	0x0	消灯
LED_ON	0x1	点灯
LED_PATTERN1	0x2	LED パターン 1
LED_PATTERN2	0x3	LED パターン 2
LED_PATTERN3	0x4	LED パターン 3
LED_PATTERN4	0x5	LED パターン 4
LED_KEEP	0xF	現状の設定を維持

3.3.10. ブザーパターン

定数名	値	説明
BUZZER_OFF	0x0	停止
BUZZER_ON	0x1	吹鳴(連続)
BUZZER_PATTERN1	0x2	ブザーパターン 1
BUZZER_PATTERN2	0x3	ブザーパターン 2
BUZZER_PATTERN3	0x4	ブザーパターン 3
BUZZER_PATTERN4	0x5	ブザーパターン 4
BUZZER_KEEP	0xF	現状の設定を維持

3.3.11. ブザー音階

定数名	値	説明
BUZZER_PITCH_OFF	0x0	停止
BUZZER_PITCH1	0x1	A6
BUZZER_PITCH2	0x2	B ♭ 6
BUZZER_PITCH3	0x3	B6
BUZZER_PITCH4	0x4	C7
BUZZER_PITCH5	0x5	D ♭ 7
BUZZER_PITCH6	0x6	D7
BUZZER_PITCH7	0x7	E ♭ 7
BUZZER_PITCH8	0x8	E7
BUZZER_PITCH9	0x9	F7
BUZZER_PITCH10	0xA	G ♭ 7
BUZZER_PITCH11	0xB	G7
BUZZER_PITCH12	0xC	A ♭ 7

BUZZER_PITCH13	0xD	A7
BUZZER_PITCH_DFLT_A	0xE	音 A のデフォルト値:D7
BUZZER_PITCH_DFLT_B	0xF	音 B のデフォルト値:(停止)

4. プログラム概要

起動後のプログラムの動作を要点のみ記載する。

4.1. LR-USB に接続

プログラム	説明
<pre> LR6-USB_Sample_VBA_64.xlsm UsbOpen() ret = HidD_GetHidGuid(guidHid) ' Get a list of currently connected HID class devices hDevInfo = SetupDiGetClassDevs(guidHid, vbNullString, 0, dwIndex = 0 sDeviceInterfaceData.cbSize = LenB(sDeviceInterfaceData) Do While (SetupDiEnumDeviceInterfaces(hDevInfo, 0, guidHi dwIndex = dwIndex + 1 ' Get the memory size of the device interface detaile ret = SetupDiGetDeviceInterfaceDetail(hDevInfo, sDevi Dim DetailDataBuffer() As Byte UetailData = Needed sDeviceInterfaceDetailData.cbSize = LenB(sDeviceInter ReDim DetailDataBuffer(Needed) MoveMemory VarPtr(DetailDataBuffer(0)), VarPtr(sDevic ' Read detailed information about the device interfac ret = SetupDiGetDeviceInterfaceDetail(hDevInfo, sDevi If ret Then Dim pBuff() As Byte ReDim pBuff(Needed - 6) Dim i As Integer For i = 0 To Needed - 6 pBuff(i) = DetailDataBuffer(i + 4) Next strDevicePath = pBuff ' Get a file handle for the device hDevice = CreateFile(strDevicePath, GENERIC_READ If hDevice <> INVALID_HANDLE_VALUE Then Dim sHidAttributes As HID_ATTRIBUTES ret = HidD_GetAttributes(hDevice, sHidAttrib ' Identify LR6-USB from Vendor ID and Device If ret And sHidAttributes.VendorID = nVendor </pre>	<p>→HID デバイスの識別子を取得</p> <p>→PC に接続中の HID クラス・デバイスのリストを取得</p> <p>→HID クラス・デバイスのリストから LR-USB のデバイスを探す</p> <p>→デバイスインターフェース詳細情報のメモリサイズを取得</p> <p>→デバイスインターフェースの詳細情報を読込</p> <p>→デバイスのファイルハンドルを取得して、有効なデバイスかどうか確認</p> <p>→ベンダーID, デバイス ID を比較して、LR6-USB かどうか確認</p>

```
ret = HidD_GetPreparedData(hDevice, Prep
If ret Then
    ret = HidP_GetCaps(PreparedData, sHi
    If ret Then
        g_nInputSize = sHidpCaps.InputRep
        g_nOutputSize = sHidpCaps.OutputR
    End If
    HidD_FreePreparedData PreparedData

    If ret Then
        g_hDevice = hDevice
        SetupDiDestroyDeviceInfoList hDev
        Exit Function
    End If
End If
End If
CloseHandle hDevice
g_hDevice = INVALID_HANDLE_VALUE
End If
Loop
```

→デバイス情報が取得して、データ送信時に必要なデータサイズを取得

→データ送信時に必要なデバイスハンドラを保持

4.2. LR-USB との切断

プログラム	説明
<pre> LR6-USB_Sample_VBA_64.xlsm UsbClose() Private Sub UsbClose() If g_hDevice <> INVALID_HANDLE_VALUE And g_hDevice <> CloseHandle g_hDevice End If g_hDevice = INVALID_HANDLE_VALUE g_nInputSize = 0 g_nOutputSize = 0 End Sub </pre>	→デバイスをクローズする

4.3. コマンドを送信

各コマンドの送信データフォーマットの送信データを LR-USB にコマンドデータを送信する。

各コマンドの送信データフォーマットの作成は「4.4LED 色と LED パターンを指定したコマンドの送信」以降を参照

プログラム	説明
<pre> LR6-USB_Sample_VBA_64.xlsm SendCommand() If g_hDevice = INVALID_HANDLE_VALUE Or g_hDevice = 0 Ther SendCommand = -1 Exit Function End If Dim pBuff() As Byte ReDim pBuff(g_nOutputSize) As Byte ' The first byte is fixed at 0x00, and the data to be ser MoveMemory VarPtr(pBuff(1)), VarPtr(sendData(0)), sendLen Dim sOverlapped As OVERLAPPED Dim dwWritten As Long Dim ret As Boolean sOverlapped.hEvent = CreateEvent(0, False, True, "") ret = WriteFile(g_hDevice, pBuff(0), g_nOutputSize, dwWri Dim dwWait As Long dwWait = WaitForSingleObject(sOverlapped.hEvent, SEND_TIM If dwWait = WAIT_OBJECT_0 Then ret = GetOverlappedResult(g_hDevice, sOverlapped, dwW If ret <> True Then SendCommand = -1 Exit Function End If Debug.Print ("Successfully sent") End If </pre>	<p>→LR-USB のデバイスハンドルの有無の確認</p> <p>→1 バイト目は 0x00 固定で、2 バイト目から送信するデータをコピーする</p> <p>→LR-USB に送信データを送信する</p> <p>→送信が完了するまで待つ</p> <p>→送信結果を取得する</p>

4.4. LED 色と LED パターンを指定したコマンドの送信

プログラム	説明
<pre> LR6-USB_Sample_VBA_64.xlsm SetLight() Dim sendData(SEND_BUFFER_SIZE - 1) As Byte ' command version (0x00: fixed) sendData(0) = COMMAND_VERSION ' Command ID (0x00: fixed) sendData(1) = COMMAND_ID ' Buzzer control (keep current) sendData(2) = BUZZER_KEEP ' buzzer scale sendData(3) = 0 ' LED control sendData(4) = (LED_KEEP * (2 ^ 4) Or LED_KEEP) sendData(5) = (LED_KEEP * (2 ^ 4) Or LED_KEEP) sendData(6) = LED_KEEP * (2 ^ 4) If color = LED_COLOR_RED Then ' red sendData(4) = (state * (2 ^ 4) Or LED_KEEP) ElseIf color = LED_COLOR_YELLOW Then ' yellow sendData(4) = (LED_KEEP * (2 ^ 4) Or state) ElseIf color = LED_COLOR_GREEN Then ' green sendData(5) = (state * (2 ^ 4) Or LED_KEEP) ElseIf color = LED_COLOR_BLUE Then ' blue sendData(5) = (LED_KEEP * (2 ^ 4) Or state) ElseIf color = LED_COLOR_WHITE Then ' white sendData(6) = state * (2 ^ 4) Else Debug.Print ("out of range color") SetLight = -1 Exit Function End If ' Empty ((0x00: Fixed)) sendData(7) = 0 ' send command Dim ret As Long ret = SendCommand(sendData, UBound(sendData) + 1) If ret <> 0 Then Debug.Print ("failed to send data") SetLight = -1 Exit Function End If </pre>	<p>以下の順で送信データを作成</p> <ul style="list-style-type: none"> →1 バイト目: コマンドバージョン (0x00) →2 バイト目: コマンド ID (0x00) →3 バイト目: ブザー制御 <p>ブザー制御は BUZZER_KEEP (0x0F) で設定を維持させる</p> <ul style="list-style-type: none"> →4 バイト目: ブザー音階 (0x00) →5~7 バイト目: LED 制御 <p>5 バイト目の 4bit~7bit: 赤の LED パターン 5 バイト目の 0bit~3bit: 黄の LED パターン 6 バイト目の 0bit~3bit: 緑の LED パターン 6 バイト目の 0bit~3bit: 青の LED パターン 7 バイト目の 0bit~3bit: 白の LED パターン 7 バイト目の 0bit~3bit: 0x0 固定</p> <p>引数で指定した色、LED パターンを設定する LED パターンは「3.3.9LED パターン」を参照</p> <ul style="list-style-type: none"> →8 バイト目: 空き (0x00) <p>→「4.3 コマンドを送信」を呼び出し、機器にデータを送信</p>

4.5. 複数の LED 色と LED パターンを指定したコマンドの送信

プログラム	説明
<pre> LR6-USB_Sample_VBA_64.xlsm SetTower() Dim sendData(SEND_BUFFER_SIZE - 1) As Byte ' command version (0x00: fixed) sendData(0) = COMMAND_VERSION ' Command ID (0x00: fixed) sendData(1) = COMMAND_ID ' Buzzer control (keep current) sendData(2) = BUZZER_KEEP ' buzzer scale sendData(3) = 0 ' LED control sendData(4) = (red * (2 ^ 4) Or yellow) sendData(5) = (green * (2 ^ 4) Or blue) sendData(6) = white * (2 ^ 4) ' Empty ((0x00: Fixed)) sendData(7) = 0 ' send command Dim ret As Long ret = SendCommand(sendData, UBound(sendData) + 1) If ret <> 0 Then Debug.Print ("failed to send data") SetTower = -1 Exit Function End If </pre>	<p>以下の順で送信データを作成</p> <ul style="list-style-type: none"> →1 バイト目: コマンドバージョン (0x00) →2 バイト目: コマンド ID (0x00) →3 バイト目: ブザー制御 <p>ブザー制御は BUZZER_KEEP (0x0F) で設定を維持させる</p> <ul style="list-style-type: none"> →4 バイト目: ブザー音階 (0x00) →5~7 バイト目: LED 制御 <p>5 バイト目の 4bit~7bit: 赤の LED パターン 5 バイト目の 0bit~3bit: 黄の LED パターン 6 バイト目の 0bit~3bit: 緑の LED パターン 6 バイト目の 0bit~3bit: 青の LED パターン 7 バイト目の 0bit~3bit: 白の LED パターン 7 バイト目の 0bit~3bit: 0x0 固定</p> <p>引数で指定して赤、黄、緑、青、白の LED パターンを設定する</p> <p>LED パターンは「3.3.9LED パターン」を参照</p> <ul style="list-style-type: none"> →8 バイト目: 空き (0x00) <p>→「4.3 コマンドを送信」を呼び出し、機器にデータを送信</p>

4.6. ブザーのパターン指定したコマンドの送信

プログラム	説明
<pre> LR6-USB_Sample_VBA_64.xlsm SetBuz() Dim sendData(SEND_BUFFER_SIZE - 1) As Byte ' command version (0x00: fixed) sendData(0) = COMMAND_VERSION ' Command ID (0x00: fixed) sendData(1) = COMMAND_ID ' Buzzer control sendData(2) = (limit * (2 ^ 4) Or buz_state) ' buzzer scale sendData(3) = (BUZZER_PITCH_DFLT_A * (2 ^ 4) Or BUZZER_P: ' LED control sendData(4) = (LED_KEEP * (2 ^ 4) Or LED_KEEP) sendData(5) = (LED_KEEP * (2 ^ 4) Or LED_KEEP) sendData(6) = LED_KEEP * (2 ^ 4) ' Empty ((0x00: Fixed)) sendData(7) = 0 ' send command Dim ret As Long ret = SendCommand(sendData, UBound(sendData) + 1) If ret <> 0 Then Debug.Print ("failed to send data") SetBuz = -1 Exit Function End If </pre>	<p>以下の順で送信データを作成</p> <ul style="list-style-type: none"> →1 バイト目: コマンドバージョン (0x00) →2 バイト目: コマンド ID (0x00) →3 バイト目: ブザー制御 <ul style="list-style-type: none"> 4bit～7bit: 連続動作/回数動作 0bit～3bit: ブザーパターン 引数で指定したブザーパターンと吹鳴させる回数(連続:0、回数:1～15)を設定する ブザーパターンは「3.3.10 ブザーパターン」を参照 →4 バイト目: ブザー音階 <ul style="list-style-type: none"> 4bit～7bit: 音 A で選択する音階 0bit～3bit: 音 B で選択する音階 音階はデフォルト(音 A は 0xE、音 B は 0xF)を設定 音階は「3.3.11 ブザー音階」を参照 →5～7 バイト目: LED 制御 LED 制御は LED_KEEP (0x0F) で設定を維持させる →8 バイト目: 空き (0x00) <p>→「4.3 コマンドを送信」を呼び出し、機器にデータを送信</p>

4.7. ブザーのパターンと音階を指定したコマンドの送信

プログラム	説明
<pre> LR6-USB_Sample_VBA_64.xlsm SetBuzEx() Dim sendData(SEND_BUFFER_SIZE - 1) As Byte ' command version (0x00: fixed) sendData(0) = COMMAND_VERSION ' Command ID (0x00: fixed) sendData(1) = COMMAND_ID ' Buzzer control sendData(2) = (limit * (2 ^ 4) Or buz_state) ' buzzer scale sendData(3) = (pitch1 * (2 ^ 4) Or pitch2) ' LED control sendData(4) = (LED_KEEP * (2 ^ 4) Or LED_KEEP) sendData(5) = (LED_KEEP * (2 ^ 4) Or LED_KEEP) sendData(6) = LED_KEEP * (2 ^ 4) ' Empty ((0x00: Fixed)) sendData(7) = 0 ' send command Dim ret As Long ret = SendCommand(sendData, UBound(sendData) + 1) If ret <> 0 Then Debug.Print ("failed to send data") SetBuzEx = -1 Exit Function End If </pre>	<p>以下の順で送信データを作成</p> <ul style="list-style-type: none"> →1 バイト目: コマンドバージョン (0x00) →2 バイト目: コマンド ID (0x00) →3 バイト目: ブザー制御 <ul style="list-style-type: none"> 4bit～7bit: 連続動作/回数動作 0bit～3bit: ブザーパターン 引数で指定したブザーパターンと吹鳴させる回数(連続:0、回数:1～15)を設定する ブザーパターンは「3.3.10 ブザーパターン」を参照 →4 バイト目: ブザー音階 <ul style="list-style-type: none"> 4bit～7bit: 音 A で選択する音階 0bit～3bit: 音 B で選択する音階 引数で指定した音A、Bの音階を設定 音階は「3.3.11 ブザー音階」を参照 →5～7 バイト目: LED 制御 LED 制御は LED_KEEP (0x0F) で設定を維持させる →8 バイト目: 空き (0x00) →「4.3 コマンドを送信」を呼び出し、機器にデータを送信

4.8. リセットコマンドの送信

プログラム	説明
<pre> LR6-USB_Sample_VBA_64.xlsm Reset() Dim sendData(SEND_BUFFER_SIZE - 1) As Byte ' command version (0x00: fixed) sendData(0) = COMMAND_VERSION ' Command ID (0x00: fixed) sendData(1) = COMMAND_ID ' Buzzer control sendData(2) = BUZZER_OFF ' buzzer scale sendData(3) = BUZZER_PITCH_OFF ' LED control sendData(4) = LED_OFF sendData(5) = LED_OFF sendData(6) = LED_OFF ' Empty ((0x00: Fixed)) sendData(7) = 0 ' send command Dim ret As Long ret = SendCommand(sendData, UBound(sendData) + 1) If ret <> 0 Then Debug.Print ("failed to send data") Reset = -1 Exit Function End If </pre>	<p>以下の順で送信データを作成</p> <ul style="list-style-type: none"> →1 バイト目: コマンドバージョン (0x00) →2 バイト目: コマンド ID (0x00) →3 バイト目: ブザー制御 停止 (0x0) を設定 →4 バイト目: ブザー音階 停止 (0x0) を設定 →5~7 バイト目: LED 制御 消灯 (0x0) を設定 →8 バイト目: 空き (0x00) <p>→「4.3 コマンドを送信」を呼び出し、機器にデータを送信</p>