Software Development Guide of LCR Communication Interface Library (LCR_CIL)

Notices

1. Introduction

This document explains how to use LCR Communication Interface Library (hereinafter referred to as "LCR_CIL") DLL to interface with LCR Pro1 / PLUS devices. An example project is provided to illustrate the LCR_CIL usage and can be used as a template for customizing PC to LCR Pro1 / PLUS interface development. DLL and example projects require Visual Studio Community 2015 (or later) to run.

The LCR_CIL.dll library provides the functions available to application developers. This library using D2XX.dll to operate. This is part of LCR Data Logger Software installation package. Or, you can go to http://www.ftdichip.com/Drivers/D2XX.htm and install latest version of D2XX drivers.

The easiest way to get started using the DLL is to run the example project. Install Visual Studio Community 2017 (or 2015) from https://imagine.microsoft.com/en-us/Catalog/Product/530.

Unzip LCR_CIL folder and run LCR_CIL.sln. Connect LCR LINK1 dongle to any USB port. Connect LCR Pro1 / PLUS to LCR LINK1 dongle. Run WFA_Lcr_Cil_Test Windows forms application project from solution. Make sure it has been set as a startup project.

2. LCR_CIL Library

Any software code examples given in this document are for information only.

This section describes how to use LCR_CIL DLL's in your project. An example is provided for C# project. It contains LCR_CIL DLL source code and sample Windows forms application project.

1. You need to declare an instance of LCR_Com object in your application:
Example C#: private LCR_Com lcr_cli = new LCR_Com();

2. On you application start by calling OpenCom() method.

Example C#:
```
if (lcr_cli.OpenCom() == false) txbStatus.Text = "Fail open communication port";
else  txbStatus.Text = "Communication port OK";
```

3. You can update LCR Pro1 / PLUS setting using these 2 steps:

Step 1: Prepare all parameters in the same category you want to write to LCR Pro1 / PLUS.

Example C#: In "Measurement Setup" category you want to set primary type to C and test frequency to 10kHz:

```
lcr_cli.SetMeasurementSetup(LCR_Com.MSM_SETUP_PRIMARY_TYPE,LCR_Com.MSMP_PRIMARY_TYPE_C);
lcr_cli.SetMeasurementSetup(LCR_Com.MSM_PARAMETER_TEST_FREQ,LCR_Com.MSMP_TEST_FREQ_10KHZ);
```

Step 2: Write these changes to LCR Pro1 / PLUS.

Example C#:
```
if (lcr_cli.WriteMeasurementSetupToLCR()== false) txbStatus.Text = "Fail write to LCR Pro1 / PLUS";
else  txbStatus.Text = "OK";
```

4. You can also read current parameters from LCR Pro1 / PLUS using these 2 steps:

Step 1: Read all parameters of the same category from LCR Pro1 / PLUS.

Example C#:
```
if (lcr_cli.ReadSysteminformationFromLCR()== false) txbStatus.Text = "Fail read from LCR Pro1 / PLUS";
else  txbStatus.Text = "OK";
```

Step 2: Analyze parameters:

Example C#: If you want to know software revision

```
uint firmware_rev_major_number;
uint firmware_rev_minor_number;
firmware_rev_major_number=lcr_cli.GetSystemInformation(SYS_INFO_SW_MAJ_V);
firmware_rev_minor_number=lcr_cli.GetSystemInformation(SYS_INFO_SW_MIN_V);
```

(if we have LCR Pro1 PLUS SW Rev 2.14 then result would be - firmware_rev_major_number is 2 and firmware_rev_minor_number is 14)

5. To start measurement and read result you need to follow these steps:

Example C#:

```
float primary_value;
float secondary_value;

Switch((lcr_cli.MakeMeasurement())
{
case LCR_Com.MESUREMENT_FAIL:
    // Measurement fail – check LCR connection and re-connect
        break;
case LCR_Com. MESUREMENT_READY:
        primary_value = lcr_cli.PrimaryValue;
        secondary_value = lcr_cli. SecondaryValue;
    // Measurement finished – read results
        break;
case LCR_Com.MESUREMENT_IDLE:
    // Measurement idle – no component connected to the device tips (no component detection)
        break;
}
```

6. To start diode (LED) measurement and read result you need to follow these steps:

Example C#:

```
float forward_voltage;
float forward_current;
float reverse_voltage;
float reverse_current;

Switch((lcr_cli. MakeDiodeMeasurement())
{
case LCR_Com.MESUREMENT_FAIL:
    // Measurement fail – check LCR connection and re-connect
        break;
case LCR_Com. MESUREMENT_READY:
    // Measurement finished – read results
        forward_voltage = lcr_cli. ForwardVoltage;
        forward_current = lcr_cli. ForwardCurrent;
        reverse_voltage  = lcr_cli. ReverseVoltage;
        reverse_current = lcr_cli. ReverseCurrent;
        break;
}
```

7. To check connection and to re-connect the device you need to call
int PingLcr()

will return:
1 - all OK, connected
0 – just re-connected
4 – lost dongle connection
5 – lost LCR Pro1 / PLUS connection

3. LCR_CIL DLL Constants

Constants used as an arguments in a dll functions call. There are x categories, each use different set of parameters and values.

The 1st category: "Measurement Setup" responsible for device initial setup to define it's behavior during measurement process. In default state LCR Pro1 / PLUS will perform L-C-R measurement with automatic component type, identification and automatic test frequency and secondary parameter selection. You can update measurement setup to switch to capacitor measurement with fixed 10kHz test frequency. Another example - switch to DCR or Diode measurement.

Here are list of constants for "Measurement Setup" category

```
    public const uint MSM_SETUP_MEASUREMENT_TYPE = 0x7;
    //-----------------------------------------
    public const uint MSMS_MEASUREMENT_TYPE_LCR = 0x0;
    public const uint MSMS_MEASUREMENT_TYPE_ESR = 0x1;
    public const uint MSMS_MEASUREMENT_TYPE_DCR = 0x2;
    public const uint MSMS_MEASUREMENT_TYPE_DIODE = 0x3;
    public const uint MSMS_MEASUREMENT_TYPE_SORTING = 0x4;
    public const uint MSMS_MEASUREMENT_TYPE_RECORDING = 0x5;
    public const uint MSMS_MEASUREMENT_TYPE_CONTINUITY = 0x6;

    public const uint MSM_SETUP_PRIMARY_TYPE = 0x7 << 3;
    //-----------------------------------------
    public const uint MSMS_PRIMARY_TYPE_AUTO = 0x0 << 3;
    public const uint MSMS_PRIMARY_TYPE_R = 0x1 << 3;
    public const uint MSMS_PRIMARY_TYPE_L = 0x2 << 3;
    public const uint MSMS_PRIMARY_TYPE_C = 0x3 << 3;
    public const uint MSMS_PRIMARY_TYPE_Z = 0x4 << 3;

    public const uint MSM_SETUP_CIRCUIT_SETTING = 0x3 << 6;
    //-----------------------------------------
    public const uint MSMS_CIRCUIT_SETTING_AUTO = 0x0 << 6;
    public const uint MSMS_CIRCUIT_SETTING_SERIES = 0x1 << 6;
    public const uint MSMS_CIRCUIT_SETTING_PARALLEL = 0x2 << 6;

    public const uint MSM_SETUP_SECONDARY_SETTING = 0x1 << 8;
    //-----------------------------------------
    public const uint MSMS_SECONDARY_TYPE_RS_RP = 0x0 << 8;
    public const uint MSMS_SECONDARY_TYPE_D_Q = 0x1 << 8;

    public const uint MSM_SETUP_TEST_FREQ = 0x7 << 9;
    //-----------------------------------------
    public const uint MSMS_TEST_FREQ_AUTO = 0x0 << 9;
    public const uint MSMS_TEST_FREQ_100HZ = 0x1 << 9;
    public const uint MSMS_TEST_FREQ_120HZ = 0x2 << 9;
    public const uint MSMS_TEST_FREQ_1KHZ = 0x3 << 9;
    public const uint MSMS_TEST_FREQ_10KHZ = 0x4 << 9;
    public const uint MSMS_TEST_FREQ_100KHZ = 0x5 << 9;

    public const uint MSM_SETUP_TEST_VOLTAGE = 0x3 << 12;
    //-----------------------------------------
    public const uint MSMS_TEST_VOLTAGE_1000MV = 0x0 << 12;
    public const uint MSMS_TEST_VOLTAGE_500MV = 0x1 << 12;
    public const uint MSMS_TEST_VOLTAGE_200MV = 0x2 << 12;

    public const uint MSM_SETUP_PHASE_ANGLE = 0x7 << 14;
    //-----------------------------------------
    public const uint MSMS_PHASE_ANGLE_AUTO = 0x0 << 14;
    public const uint MSMS_PHASE_ANGLE_5 = 0x1 << 14;
    public const uint MSMS_PHASE_ANGLE_10 = 0x2 << 14;
    public const uint MSMS_PHASE_ANGLE_15 = 0x3 << 14;
    public const uint MSMS_PHASE_ANGLE_20 = 0x4 << 14;
    public const uint MSMS_PHASE_ANGLE_30 = 0x5 << 14;
    public const uint MSMS_PHASE_ANGLE_45 = 0x6 << 14;

    public const uint MSM_SETUP_TEST_SPEED = 0x7 << 17;
    //----------------------------------------
    public const uint MSMS_TEST_SPEED_AUTO = 0x0 << 17;
    public const uint MSMS_TEST_SPEED_500MS = 0x1 << 17;
    public const uint MSMS_TEST_SPEED_1S = 0x2 << 17;
    public const uint MSMS_TEST_SPEED_2S = 0x3 << 17;

    public const uint MSM_SETUP_TOLERANCE = 0x7 << 20;
    //----------------------------------------
    public const uint MSMS_TOLERANCE_0_5 = 0x0 << 20;
    public const uint MSMS_TOLERANCE_1 = 0x1 << 20;
    public const uint MSMS_TOLERANCE_5 = 0x2 << 20;
    public const uint MSMS_TOLERANCE_10 = 0x3 << 20;
    public const uint MSMS_TOLERANCE_20 = 0x4 << 20;
    public const uint MSMS_TOLERANCE_30 = 0x5 << 20;

    public const uint MSM_SETUP_CONTINUITY = 0x7 << 23;
    //----------------------------------------
    public const uint MSMS_CONTINUITY_TH_0_1 = 0x0 << 23;
    public const uint MSMS_CONTINUITY_TH_0_5 = 0x1 << 23;
    public const uint MSMS_CONTINUITY_TH_1 = 0x2 << 23;
    public const uint MSMS_CONTINUITY_TH_10 = 0x3 << 23;
    public const uint MSMS_CONTINUITY_TH_100 = 0x4 << 23;
    public const uint MSMS_CONTINUITY_TH_1K = 0x5 << 23;

    public const uint MSM_SETUP_STATE = 0x3 << 27;
    //----------------------------------------
    public const uint MSMS_STATE_CLEAR = 0x0 << 27;
    public const uint MSMS_STATE_START = 0x1 << 27;
    public const uint MSMS_STATE_UPDATE = 0x2 << 27;

    public const uint MSM_SETUP_BUZZER = 0x1 << 29;
    //----------------------------------------
    public const uint MSMS_BUZ_OFF = 0x0 << 29;
    public const uint MSMS_BUZ_BEEP = 0x1 << 29;

    public const uint MSM_SETUP_DISPLAY = 0x1 << 30;
    //----------------------------------------
    public const uint MSMS_DISPLAY_ON = 0x0 << 30;
    public const uint MSMS_DISPLAY_OFF = 0x1 << 30;
```

The 2nd category: "Measurement Parameters" used when you need to read LCR Pro1 / PLUS automatic selection of measurement parameters.

```
public const uint MSM_PARAM_MEASUREMENT_TYPE = 0x7;
//-----------------------------------------
public const uint MSMP_MEASUREMENT_TYPE_LCR = 0x0;
public const uint MSMP_MEASUREMENT_TYPE_ESR = 0x1;
public const uint MSMP_MEASUREMENT_TYPE_DCR = 0x2;
public const uint MSMP_MEASUREMENT_TYPE_DIODE = 0x3;
public const uint MSMP_MEASUREMENT_TYPE_SORTING = 0x4;

public const uint MSM_PARAMETER_PRIMARY_TYPE = 0x7 << 3;
//-----------------------------------------
public const uint MSMP_PRIMARY_TYPE_R = 0x1 << 3;
public const uint MSMP_PRIMARY_TYPE_L = 0x2 << 3;
public const uint MSMP_PRIMARY_TYPE_C = 0x3 << 3;
public const uint MSMP_PRIMARY_TYPE_Z = 0x4 << 3;

public const uint MSM_PARAMETER_CIRCUIT_SETTING = 0x3 << 6;
//-----------------------------------------
public const uint MSMP_CIRCUIT_SETTING_SERIES = 0x1 << 6;
public const uint MSMP_CIRCUIT_SETTING_PARALLEL = 0x2 << 6;

public const uint MSM_PARAMETER_SECONDARY_SETTING = 0x1 << 8;
//-----------------------------------------
public const uint MSMP_SECONDARY_TYPE_RS_RP = 0x0 << 8;
public const uint MSMP_SECONDARY_TYPE_D_Q = 0x1 << 8;

public const uint MSM_PARAMETER_TEST_FREQ = 0x7 << 9;
//-----------------------------------------
public const uint MSMP_TEST_FREQ_100HZ = 0x1 << 9;
public const uint MSMP_TEST_FREQ_120HZ = 0x2 << 9;
public const uint MSMP_TEST_FREQ_1KHZ = 0x3 << 9;
public const uint MSMP_TEST_FREQ_10KHZ = 0x4 << 9;
public const uint MSMP_TEST_FREQ_100KHZ = 0x5 << 9;

public const uint MSM_PARAMETER_TEST_VOLTAGE = 0x3 << 12;
//-----------------------------------------
public const uint MSMP_TEST_VOLTAGE_1000MV = 0x0 << 12;
public const uint MSMP_TEST_VOLTAGE_500MV = 0x1 << 12;
public const uint MSMP_TEST_VOLTAGE_200MV = 0x2 << 12;

public const uint MSM_PARAMETER_PHASE_ANGLE = 0x7 << 14;
//-----------------------------------------
public const uint MSMP_PHASE_ANGLE_5 = 0x1 << 14;
public const uint MSMP_PHASE_ANGLE_10 = 0x2 << 14;
public const uint MSMP_PHASE_ANGLE_15 = 0x3 << 14;
public const uint MSMP_PHASE_ANGLE_20 = 0x4 << 14;
public const uint MSMP_PHASE_ANGLE_30 = 0x5 << 14;
public const uint MSMP_PHASE_ANGLE_45 = 0x6 << 14;

public const uint MSM_PARAMETER_TEST_SPEED = 0x7 << 17;
//-----------------------------------------
```

```
    public const uint MSMP_TEST_SPEED_250MS = 0x1 << 17;
    public const uint MSMP_TEST_SPEED_500MS = 0x2 << 17;
    public const uint MSMP_TEST_SPEED_1S = 0x3 << 17;

    public const uint MSM_PARAMETER_TOLERANCE = 0x7 << 20;
    //----------------------------------------
    public const uint MSMP_TOLERANCE_0_5 = 0x0 << 20;
    public const uint MSMP_TOLERANCE_1 = 0x0 << 20;
    public const uint MSMP_TOLERANCE_5 = 0x0 << 20;
    public const uint MSMP_TOLERANCE_10 = 0x0 << 20;
    public const uint MSMP_TOLERANCE_20 = 0x0 << 20;
    public const uint MSMP_TOLERANCE_30 = 0x0 << 20;

    public const uint MSM_PARAMETER_DEVICE_STATUS = 0x7 << 23;
    //----------------------------------------
    public const uint MSMP_STATUS_CLEAR = 0x0 << 23;
    public const uint MSMP_STATUS_DATA_READY = 0x1 << 23;
    public const uint MSMP_STATUS_DEVICE_IDLE = 0x2 << 23;
    public const uint MSMP_STATUS_DEVICE_START = 0x4 << 23;
```

The 3rd category: "System Info" used to get or to set LCR Pro1 / PLUS user interface parameters such as contrast, sound, orientation ......

```
    public const uint SYS_INFO_DISPLAY_MODE = 0x1;
    //----------------------------------------
    public const uint SYSI_DISPLAY_MODE_RHAND = 0x0;
    public const uint SYSI_DISPLAY_MODE_LHAND = 0x1;

    public const uint SYS_INFO_CONTRAST = 0xF << 1;
    //----------------------------------------
    public const uint SYSI_CONTRAST_10 = 0x1 << 1;
    public const uint SYSI_CONTRAST_20 = 0x2 << 1;
    public const uint SYSI_CONTRAST_30 = 0x3 << 1;
    public const uint SYSI_CONTRAST_40 = 0x4 << 1;
    public const uint SYSI_CONTRAST_50 = 0x5 << 1;
    public const uint SYSI_CONTRAST_60 = 0x6 << 1;
    public const uint SYSI_CONTRAST_70 = 0x7 << 1;
    public const uint SYSI_CONTRAST_80 = 0x8 << 1;
    public const uint SYSI_CONTRAST_90 = 0x9 << 1;
    public const uint SYSI_CONTRAST_100 = 0xA << 1;

    public const uint SYS_INFO_SOUND = 0x1 << 5;
    //----------------------------------------
    public const uint SYSI_SOUND_OFF = 0x0 << 5;
    public const uint SYSI_SOUND_ON = 0x1 << 5;

    public const uint SYS_INFO_TIMEOUT = 0x7 << 6;
    //----------------------------------------
    public const uint SYSI_TIMEOUT_30S = 0x0 << 6;
    public const uint SYSI_TIMEOUT_1M = 0x1 << 6;
    public const uint SYSI_TIMEOUT_2M = 0x2 << 6;
    public const uint SYSI_TIMEOUT_4M = 0x3 << 6;
    public const uint SYSI_TIMEOUT_ALWAYS_ON = 0x4 << 6;

    public const uint SYS_INFO_HW_MAJ_V = 0x3 << 9;
    //-----------------------------------------

    public const uint SYS_INFO_HW_MIN_V = 0x7 << 11;
    //-----------------------------------------

    public const uint SYS_INFO_SW_MAJ_V = 0x7 << 14;
    //-----------------------------------------

    public const uint SYS_INFO_SW_MIN_V = 0x1F << 17;
    //-----------------------------------------

    public const uint SYS_INFO_BAT_LEVEL = 0xF << 22;
    //-----------------------------------------

    public const uint SYS_INFO_BAT_CHARGING = 0x1 << 26;
    //-----------------------------------------
    //-----------------------------------------
    public const uint SYSI_STATUS_PAR_CHANGE = 0x1 << 30;
    public const uint SYSI_STATUS_IF_BUSY = 0x1 << 30;
```

The 4th category used in self calibration process.

```
    public const uint CAL_CONFIG_SELFCAL = 0xF << 0;
    //-----------------------------------------
    public const uint CALCF_CLEAR = 0x0;
    public const uint CALCF_OPEN_TIPS = 0x1;
    public const uint CALCF_OPEN_START = 0x2;
    public const uint CALCF_OPEN_PASS = 0x3;
    public const uint CALCF_OPEN_FAIL = 0x4;
    public const uint CALCF_OPEN_RESET = 0x5;
    public const uint CALCF_SHORT_TIPS = 0x6;
    public const uint CALCF_SHORT_START = 0x7;
    public const uint CALCF_SHORT_PASS = 0x8;
    public const uint CALCF_SHORT_FAIL = 0x9;
    public const uint CALCF_SHORT_RESET = 0xA;
```

4. LCR_CIL DLL Methods / Functions

float PrimaryValue; - to hold primary measure value
float SecondaryValue; - to hold secondary measure value
public bool OpenCom(); - open communication port
public bool CloseCom(); - close communication port before your program exit
public bool WriteMeasurementSetupToLCR(); - write measurement setup to LCR
public bool ReadMeasurementSetupFromLCR(); - read measurement setup from LCR
public bool IsMeasurementSetupChanged(); - check is parameter has been changed after reading from LCR

public void SetMeasurementSetup(uint type, uint value); - set measurement setup parameter
public uint GetMeasurementSetup(uint type); - get measurement setup
public bool ReadMeasurementParametersFromLCR(); - read measurement setup from LCR(no write function !)
public bool IsMeasurementParametersChanged(); - check is parameter has been changed after reading from LCR
public uint GetMeasurementParameter(uint type); - get measurement parameter
public bool WriteSysteminformationToLCR(); - write system information setup to LCR
public bool ReadSysteminformationFromLCR(); - read system information from LCR
public void SetSystemInformation(uint type, uint value); - set system information parameter
public uint GetSystemInformation(uint type); - get system information parameter
public bool WriteCalibrationConfigToLCR(); - write calibration config setup to LCR
public bool ReadCalibrationConfigFromLCR(); - read calibration config from LCR
public void SetCalibrationConfig(uint type, uint value); - set calibration config parameter
public bool ReadPrimaryMeasurementValueFromLCR(); - read primary result from LCR
public bool ReadSecondaryMeasurementValueFromLCR(); - read secondary result from LCR