

# Spartan 821/721 SDK Documentation

V1.300

Larson Davis – PCB Piezotronics, Inc.

Confidential

## Table of Contents

Spartan 821/721 SDK Documentation .....	1
Change Log .....	3
Introduction.....	4
Using HttpLD.exe for Communication.....	4
Linux .....	4
Deployment.....	4
Commands.....	5
Live Status.....	5
Min Status.....	5
Get About Info .....	6
Request Meter Settings.....	6
Run.....	8
Stop.....	8
Pause .....	9
Store .....	9
Measurement History Count.....	9
Measurement History Interval .....	9
Get File List .....	10
Download a file.....	11
Delete a file.....	12
Delete all files .....	12
Time History Count.....	12
Time History Graph Data .....	13
Reboot Meter .....	14
Appendix A .....	15

## Change Log

Initial draft	5/16/2024
Adding additional commands	8/6/24

## Introduction

The SDK has the following parts:

1. This document which describes the functionality of the SDK and executable.
2. Sample code to talk USB Serial to 821 / 721 on Linux
3. The HttpLD.exe and supporting files that is the core of the SDK functionality.
  - a. All of the files located in the bin\Win32 folder are required to use the SDK.

## Using HttpLD.exe for Communication

The HttpLD.exe (HLD) is an executable for Windows® based OS's. It is the method of communication with the updated SoundExpert© or Spartan Model 821 (821SE or 821) and SoundExpert© or Spartan Model 721 (721SE or 721) meters. You may launch HLD from the command line to talk to a USB connected meter. This method of connection is based on standard HTTP requests and responses. Much of the data that comes from the meter is in JSON.

Most commands can be sent with the following format through HLD:

`http://<ipAddress>:<port>/sdk?func=cmd&op=send&message=<command>`

If you are not developing in a Windows environment, then the communications must be done on a raw USB Serial port, with the commands sent differently.

### Linux

The 821 / 721 meters will show up as a USB Composite device, with a USB Mass Storage interface (only for upgrades via G4) and a USB CDC serial driver. Commands can be written to the USB serial interface and read as if from a standard serial port.

### Deployment

Depending on your license agreement, you may wish to deploy an application in which you may choose to include the current HLD and associated documentation. You may add HLD (based on license agreement) to any location you choose and your application may continue to communicate through this channel even if newer HLDs are installed through other LD packages such as G4.

**Known Limitations:** Due to previous design decisions, not all commands return properly formatted JSON. This is an issue that will be rectified in the future. For now, the meter response must be filtered through the following RegEx expressions to get to a correct JSON format.

```
Regex DeletableTokens = new Regex(@"(\\"*\\"r)|(\r)|( (?<="" :-?))0*(?=([1-9]))");  
Regex WhitespacePadding = new Regex(@"(\s*:\s*)\+?";  
Regex InvalidTokens = new Regex(@"(--)|(\b(nan)|(-?inf)\b)";
```

## Commands

### Live Status:

Command: **G9\r**

Returns: JSON response with meter status, including live Leq , Lpeak, battery, and overall metrics.

Example Response:

```
{  
  "Response": {  
    "Device": "Larson Davis SoundExpert 821",  
    "Serial Number": "4027",  
    "Firmware Revision": "1.300B17",  
    "Hardware Revision": "X7",  
    "Battery Voltage": 4.2,  
    "Estimated Run Time": 57372,  
    "Free Memory": 30277,  
    "Total Memory": 30286,  
    "LAeq": 51.7,  
    "LCeq": 61.41,  
    "LZeq": 63.7,  
    "LAS": 49.8,  
    "LAF": 46.7,  
    "LAI": 60.9,  
    "LCS": 59.1,  
    "LCF": 54.4,  
    "LCI": 70.6,  
    "LZS": 62,  
    "LZF": 61.5,  
    "LZI": 71.4,  
    "LApeak": 77.7,  
    "LCpeak": 84.9,  
    "LZpeak": 84.9,  
    ... (elided for brevity)  
    "FW Date": 1719350752,  
    "FW Install Date": 1719569965,  
    "Overload Duration": 0,  
    "Overload Count": 0,  
    "Est Run Time Screen Off": 141375  
  },  
  "Result": "Success: 0 ",  
  "ResultCode": 0,  
  "ResultName": "Success"  
}
```

### Min Status:

Command: **G\r**

Returns: JSON response with a minimal meter status.

Example Response:

```
{  
    "Response": {  
        "Estimated Run Time": 57382,  
        "Free Memory": 30286,  
        "Total Memory": 30286,  
        "LCeq": 63.45,  
        "Mode": 0,  
        "Runtime": 0,  
        "Cal Delta": 0,  
        "Time": 1722942063,  
        "Error Flags": 196608,  
        "Indicators": 512,  
        "File Count": 0,  
        "Alert State": 0  
    },  
    "Result": "Success: 0 ",  
    "resultCode": 0,  
    "resultName": "Success"  
}
```

## **Get About Info:**

Command: G8\r

Returns: JSON response with meter specific info.

### Example Response:

```
{  
    "Response": {  
        "Serial Number": "4027",  
        "Firmware Revision": "1.300R17",  
        "Hardware Revision": "X7",  
        "BLE FW Ver": "0.801",  
        "Cert Date": "2022-08-23",  
        "MFG Date": "2023-01-10",  
        "Options": 277,  
        "Model": "821SE",  
        "FW Date": 1718313610,  
        "FW Install Date": 1722873844  
    },  
    "Result": "Success: 0 ",  
    "ResultCode": 0,  
    "ResultName": "Success"  
}
```

## **Request Meter Settings:**

Command: S1:215?NVj\r

Returns: JSON containing the meter settings.

### Example Response:

```
{  
  "Response": {  
    "settings": {  
      "system": {  
        "Device Name": "",  
        "Date Format": "0,  
        "Noise Level": 15,  
        "dB Reference": 0.00002,  
        "Language": 0,  
        "Decimal Separator": 0,  
        "Time Format": "12  
      }  
    }  
  }  
}
```

```
"Auto Off Time": 0,  
"BLE Auth Key": "",  
"Time": "02:00:43",  
"Date": "2024-08-05",  
"Calibration Level": 114,  
"Sensitivity": -25.95,  
"Epoch Time": 1722823243,  
"AdminPass": "",  
"AdminFlags": 7,  
"Auto Store Enable": 1,  
"Modem Enable": 1,  
"Modem Period": 30,  
"Modem Host": "20.190.42.6",  
"Modem Port": 80,  
"Backlight": 50,  
"Modem APN": "iot.truphone.com",  
"Correction": 1,  
"Preamp Serial Number": "",  
"Microphone": 0,  
"Microphone Serial Number": "",  
"Microphone Model Name": "377B02",  
"Microphone Sensitivity dB": -26,  
"Microphone Sensitivity": 50.12,  
"Device Description": "",  
"DC Output Enable": 0,  
"DAC Output Adjust": 0,  
"Ext Shutoff Volt": 10.5  
},  
"meas": {  
    "SPL Freq Wt": "A",  
    "SPL Detector": "S",  
    "Pk Freq Wt": "C",  
    "Dose1 Enable": 1,  
    "Dose1 Config Select": 5,  
    "Dose1 Mode": 0,  
    "Dose1 Title": "Custom 1",  
    "Dose1 Freq Weight": "A",  
    "Dose1 Time Weight": "S",  
    "Dose1 Peak Weight": 0,  
    "Dose1 Exch Rate": 5,  
    "Dose1 Thresh Enable": 1,  
    "Dose1 Thresh Level": 90,  
    "Dose1 Crit Level": 90,  
    "Dose1 Crit Time": 8,  
    "Dose1 Shift Time": 8,  
    "Dose2 Enable": 1,  
    "Dose2 Config Select": 5,  
    "Dose2 Mode": 0,  
    "Dose2 Title": "Custom 2",  
    "Dose2 Freq Weight": "A",  
    "Dose2 Time Weight": "S",  
    "Dose2 Peak Weight": 0,  
    "Dose2 Exch Rate": 5,  
    "Dose2 Thresh Enable": 1,  
    "Dose2 Thresh Level": 80,  
    "Dose2 Crit Level": 90,  
    "Dose2 Crit Time": 8,  
    "Dose2 Shift Time": 8,  
    "Dose3 Enable": 1,  
    "Dose3 Config Select": 5,  
    "Dose3 Mode": 0,  
    "Dose3 Title": "Custom 3",  
    "Dose3 Freq Weight": "A",  
    "Dose3 Time Weight": "S",  
    "Dose3 Peak Weight": 0,  
    "Dose3 Exch Rate": 3,  
    "Dose3 Thresh Enable": 1,  
    "Dose3 Thresh Level": 80,  
    "Dose3 Crit Level": 85,  
    "Dose3 Crit Time": 8,  
    "Dose3 Shift Time": 8,  
    "Dose4 Enable": 1,  
    "Dose4 Config Select": 5,  
    "Dose4 Mode": 1,  
    "Dose4 Title": "Custom 4",  
    "Dose4 Freq Weight": "A",  
    "Dose4 Time Weight": "S",  
    "Dose4 Peak Weight": 1,  
    "Dose4 Exch Rate": 3,  
    "Dose4 Thresh Enable": 1,  
    "Dose4 Thresh Level": 80,  
    "Dose4 Crit Level": 85,  
    "Dose4 Crit Time": 8,  
    "Dose4 Shift Time": 8,  
    "Alarm1 Enable": 0,  
    "Alarm1 LED Indicator": 0,  
    "Alarm1 Source": 0,  
    "Alarm1 Action Level": 50,  
    "Alarm1 Limit Level": 100,  
    "Alarm2 Enable": 0,  
    "Alarm2 LED Indicator": 0,  
    "Alarm2 Source": 1,  
    "Alarm2 Action Level": 50,  
    "Alarm2 Limit Level": 100,  
    "Time Hist En": 1,  
    "Time Hist Period": 1,  
    "Time Hist Metric": 0,  
    "OBA Enable": 1,  
    "OBA Freq Weight": 0,  
    "Event SR Enable": 0,  
    "Event SR Trig Src": 0,  
    "Event SR Trig Lvl": 85,  
    "Event SR Min Interval": 30,  
    "Event SR Period": 2,  
    "Event SR Pre Period": 2,  
    "SPL1": 85,  
    "SPL2": 95,  
    "Peak1": 135,  
    "Peak2": 137,  
    "Peak3": 140,  
    "Auto Enable": 1,  
    "Timer Mode": 0,  
    "Start Date": "2024-04-24",  
    "Stop Date": "2029-04-24",  
    "Timer 1 Start": "08:00:00",  
    "Timer 1 Stop": "12:00:00",  
    "Timer 2 Enable": 0,  
    "Timer 2 Start": "13:00:00",  
    "Timer 2 Stop": "17:00:00",  
    "Timer 3 Enable": 0,  
    "Timer 3 Start": "18:00:00",  
    "Timer 3 Stop": "23:00:00",  
    "Timed Stop Duration": "00:00:30",  
    "Merge Files": 0,  
    "Continuous Run Mode Interval": 0,  
    "Day Time": "07:00",  
    "Evening Time": "19:00",  
    "Night Time": "22:00",  
    "Evening Penalty": 7,  
    "Night Penalty": 11,
```

```
"Ln Enable": 1,  
"Ln Perc 1": 5,  
"Ln Perc 2": 10,  
"Ln Perc 3": 33.3,  
"Ln Perc 4": 50,  
"Ln Perc 5": 66.6,  
"Ln Perc 6": 90,  
"Ln Freq Weight": "A",  
"Ln Time Weight": "S",  
"Meas Hist Enable": 1,  
"Meas Hist Interval": 2147483647,  
"Gain Enable": 0,  
"TimeHistOBAEnable": 1,  
"Marker1": "Truck",  
"Marker2": "Automobile",  
"Marker3": "Motorcycle",  
"Marker4": "Aircraft",  
"Marker5": "Exclude",  
"Marker6": "#6",  
"Marker7": "#7",  
"Marker8": "#8",  
"Marker9": "#9",  
"Marker10": "#10",  
"TimeHistPwrEnable": 0,  
"OBA Time wt": "S",  
  
        "Time Hist Min Max Enable": 0,  
        "TH Flags 0": 2164279539,  
        "TH Flags 1": 64,  
        "TH Flags 2": 0,  
        "TH Flags 3": 0  
    },  
    "device": {  
        "Temp Comp": -0.04,  
        "Model Index": 5,  
        "Options": 277,  
        "Cert Date": "2022-08-23",  
        "MFG Date": "2023-01-10",  
        "Serial": "4027",  
        "Volt Cal": 1.023771,  
        "Range Cal": 0.997805,  
        "Model": "821SE",  
        "Certified Sensitivity dB": -9999.900391,  
        "Certified Sensitivity": -99999986991103.9  
    }  
},  
"Result": "Success: 0 ",  
"ResultCode": 0,  
"ResultName": "Success"  
}
```

**Run:** Starts a measurement.

Command: **M1\r**

Returns: “Run Pending” if successful, otherwise will indicate an error.

Example Response:

```
{  
    "Response": "Run Pending",  
    "Result": "Success: 0 ",  
    "ResultCode": 0,  
    "ResultName": "Success"  
}
```

**Stop:** Stops a measurement. If the ‘Store On Stop’ setting is On, the data will automatically be stored to a file.

Command: **M2\r**

Returns: “Stop Pending” if successful, otherwise will indicate an error.

Example Response:

```
{  
    "Response": "Stop Pending",  
    "Result": "Success: 0 ",  
    "ResultCode": 0,  
    "ResultName": "Success"  
}
```

**Pause:** Pauses a measurement.

Command: **M3\r**

Returns: “Pause Pending” if successful, otherwise will indicate an error.

Example Response:

```
{  
  "Response": "Pause Pending",  
  "Result": "Success: 0 ",  
  "resultCode": 0,  
  "ResultName": "Success"  
}
```

**Store:** Stores the measurement data to a file. Must be in “Stopped” state and ‘Store On Stop’ setting is Off.

Command: **M11\r**

Returns: “Store Pending” if successful, otherwise will indicate an error.

Example Response:

```
{  
  "Response": "Store Pending",  
  "Result": "Success: 0 ",  
  "resultCode": 0,  
  "ResultName": "Success"  
}
```

**Measurement History Count:** Gets the number of completed measurement histories in the current measurement.

Command: **R132\r**

Returns: The number of completed measurement history intervals for the current measurement.

Example Response:

```
{  
  "Response": "15",  
  "Result": "Success: 0 ",  
  "resultCode": 0,  
  "ResultName": "Success"  
}
```

**Measurement History Interval:** Gets the data for a given measurement history interval.

Command: **R133,<index>\r**

Returns: The measurement history interval for the given index. Index must be in the

range of 0 to Count – 1 (where Count is the response to R132). A special case of index = -1 can be sent to retrieve the currently active unstored interval.

Example Response:

```
{  
    "Response": {  
        "MeasHist": {  
            "Number": 7,  
            "StartTime": 1722875940,  
            "StopTime": 1722876000,  
            "RunTime": 60,  
            "PauseTime": 0,  
            "LAeq": 36.73,  
            "LAleq": 44.4,  
            "LAPeak": 53.23,  
            "LAIMax": 59.35,  
            "LAFMax": 42.88,  
            "LASMax": 46.65,  
            "LAIMin": 36.97,  
            "LAFMin": 35.86,  
            "LASMin": 36.3,  
            "LCeq": 60.92,  
            "LCleq": 64.15,  
            "LCPeak": 71.77,  
            "LCIMax": 66.73,  
            "LCFMax": 65.09,  
            "LCSMax": 62.66,  
            "LCIMin": 61.39,  
            "LCFMin": 56.53,  
            "LCSMin": 58.82,  
            "LZeq": 66.96,  
            "LZleq": 71.02,  
            "LZPeak": 79.14,  
            "LZIMax": 75,  
            "LZFMax": 72.1,  
            "LZSMax": 69.69,  
            "LZIMin": 67.16,  
            "LZFMin": 60.99,  
            "LZSMin": 63.88,  
            "OvldDur": 0,  
            "Ovld": 0,  
            ... (elided for brevity)  
            "Ln": [  
                37.6,  
                37,  
                36.8,  
                36.7,  
                36.6,  
                36.5  
            ]  
        }  
    },  
    "Result": "Success: 0 ",  
    "ResultCode": 0,  
    "ResultName": "Success"  
}
```

**Get File List:** Gets a list of files on the meter.

Command: **D2\r**

Returns: JSON array of the files on the meter or an empty array if no files exist.

Example Response:

```
{  
  "DataFiles": [  
    {  
      "name": "240618015.LD7",  
      "startTime": 1718728651,  
      "m": 34.5,  
      "t": 1,  
      "size": 19628,  
      "runTime": 58911  
    },  
    {  
      "name": "240723001.LD7",  
      "startTime": 1721734601,  
      "m": 75.5,  
      "t": 1,  
      "size": 2967988,  
      "runTime": 60766  
    },  
    {  
      "name": "240805000.LD7",  
      "startTime": 1722849337,  
      "m": 51.8,  
      "t": 1,  
      "size": 902796,  
      "runTime": 18237  
    }  
  "Status": "Success",  
  "Result": "Success: 0 ",  
  "ResultCode": 0,  
  "ResultName": "Success"  
}
```

Example Response when there are no files on the meter:

```
{  
  "Response": {  
    "DataFiles": [],  
    "Status": "Failed"  
  },  
  "Result": "Success: 0 ",  
  "ResultCode": 0,  
  "ResultName": "Success"  
}
```

## Download a file:

Command: **D5,<filename>,<offset>,<bytes requested>\r**

Returns: Packetized data archive .LD7 file with the following format, in up to 2048 byte packets:

STX	Block ID	Length	Data (up to 2048 bytes)	*	X	X	X	ETX
-----	----------	--------	----------------------------	---	---	---	---	-----

With the last packet requested adding the EOT byte at the end:

STX	Block ID	Length	Data	*	X	X	X	ETX	EOT
-----	----------	--------	------	---	---	---	---	-----	-----

\*XXX is the checksum of the data represented in ASCII with 3 characters after the \* (i.e. a checksum of 25 will show \*025). This is calculated by adding each byte in the data payload into a single unsigned byte (maximum 255).

**Delete a file:** Delete a specific file by filename. File must exist on the meter.

Command: **D8,<filename>\r**

Returns: "SUCCESS" if successful, ERR\_XXXXX if error, where XXXXX details the type of error.

Example Response:

```
{  
  "Response": "SUCCESS",  
  "Result": "Success: 0 ",  
  "resultCode": 0,  
  "resultName": "Success"  
}
```

**Delete all files:**

Command: **D9\*125\r**

Returns: "SUCCESS" if successful, ERR\_XXXXX if error, where XXXXX details the type of error.

Example Response:

```
{  
  "Response": "SUCCESS",  
  "Result": "Success: 0 ",  
  "resultCode": 0,  
  "resultName": "Success"  
}
```

**Time History Count:** Gets the number of completed time histories in the current measurement.

Command: **D11, live, 0, 0\r**

Returns: The number of completed time history records in the current measurement.

Example Response:

```
{  
  "Response": "3180",  
  "Result": "Success: 0 ",  
  "ResultCode": 0,  
  "ResultName": "Success"  
}
```

**Time History Graph Data:** Gets up to 120 consecutive time history data points, beginning at index, from the current measurement. Actual number of data points returned will be the lesser of 120 or (Count - index).

Command: **D10, live, <index>, <metric>\r**

Note: see Appendix A for a list of valid metric ids and how to specify more than one.

Returns: JSON array of Data points composed of timestamp, flags, and the value(s) of the specified metric(s).

Example Response for D10, live, 120, 1281:

```
{  
  "Response": {  
    "TimeHist": {  
      "Metrics": [  
        "Time",  
        "Flags",  
        "LAeq",  
        "LCpeak"  
      ],  
      "Data": [  
        [  
          1723034516,  
          8192,  
          39.51,  
          72.36  
        ],  
        [  
          1723034517,  
          8192,  
          39.57,  
          73.15  
        ],  
        [  
          1723034518,  
          8192,  
          38.77,  
          72.62  
        ],  
        ... (elided for brevity)  
        [  
          1723034634,  
          8192,  
          33.58,  
          ... (elided for brevity)  
        ]  
      ]  
    }  
  }  
}
```

```
    66.32
  ],
  [
    1723034635,
    8192,
    33.59,
    66.01
  ]
}
},
"Result": "Success: 0 ",
"ResultCode": 0,
"ResultName": "Success"
}
```

## Reboot Meter:

Command: **M8\*133\r**

Note: Meter must be in a “Stopped” state for command to succeed.

Returns: “Powering Off” if successful or “Stop Required” if meter is not “Stopped”.

Example Response:

```
{
  "Response": " Powering Off",
  "Result": "Success: 0 ",
  "ResultCode": 0,
  "ResultName": "Success"
}
```

## Appendix A

### Time History Graph Data

The enum defined below represents the metric ids that can be used for the <metric> parameter of the D10 command. Note that any metric with min or max in the name will not be valid unless the 'Min/Max in Time History' setting is set to Min, Max, or Both.

```
// this enum is used to provide an ID for each metric
public enum MetricId
{
    Flags = 0,
    LAeq,
    LCeq,
    LASmax,
    LAFmax,
    LCpeak,
    LCpk = LCpeak,
    LZpeak,
    LZpk = LZpeak,
    TWA3,
    TWA5,
    IntTemp,
    GaugeTemp,
    BattVolt,
    BattPercent,
    AvgCurrent,
    EstTime,
    FullOba,           // Not valid for SDK
    ThirdOba,          // Not valid for SDK
    FullObaMin,        // Not valid for SDK
    FullObaMax,        // Not valid for SDK
    ThirdObaMin,       // Not valid for SDK
    ThirdObaMax,       // Not valid for SDK
    LASmin,
    LAFmin,
    LAImin,
    LAImax,
    LApeak,
    LAPk = LApeak,
    LCSmin,
    LCSmax,
    LCFmin,
    LCFmax,
    LCImin,
```

```
LClmax,  
LZeq,  
LZSmin,  
LZSmax,  
LZFmin,  
LZFmax,  
LZImin,  
LZImax,  
ExtVolts  
}
```

It is also possible to specify up to 4 metrics at the same time by using the following formula:

Metric1 | (Metric2 << 8) | (Metric3 << 16) | (Metric4 << 24)

For example, to get both LAeq and LCpeak for each data point, the metric value would be

1 | (5 << 8) == 1281

The order of the metric ids in the formula does not matter and using higher value ids with the smaller bit shifts will result in a smaller number for the metric parameter. The following formula is equivalent to the one above.

5 | (1 << 8) == 261

Regardless of the order of the metrics used in the formula, the resulting JSON will always return values in lowest to highest id order. Also, only valid metric ids will have values returned. If none of the ids used in the formula are valid, the resulting JSON will have an empty Data array.