QCMApp

Overview

QCMApp is a PyQt5-based application for controlling and visualizing data from a Quartz Crystal Microbalance (QCM) device. It features real-time plotting, parameter management, and file operations for measurement data.

Dependencies

Install the following dependencies before running the application:

Python Packages

- `PyQt5`
- `pyqtgraph`
- `numpy`
- `matplotlib`

You can install them using pip:

```
pip install PyQt5 pyqtgraph numpy matplotlib
```

System Requirements

- Python 3.7+
- WSL (Windows Subsystem for Linux) if running on Windows and using USB devices

Running the Application

1. Clone the repository (if you haven't already):

```
git clone <your-repo-url>
cd QCMAppib
```

2. Install dependencies (see above).

3. Run the application:

```
sudo python3 src/main.py
```

Using USB Devices from Windows in WSL

Using USB Devices from Windows in WSL If your QCM device is connected via USB and you are running the app in WSL, you need to pass the USB device from Windows to WSL.

Steps: #### 1. Find your device in Windows:

```
- Open Device Manager and locate your QCM device under "Ports (COM & LPT)".
- Note the COM port (e.g., COM3).
```

2. Install usbipd-win on Windows:

```
- Download and install from: https://github.com/dorssel/usbipd-win
```

List USB devices in Windows:

3. List USB devices in Windows:

```
usbipd list
```

4. Attach the device to WSL:

```
usbipd wsl attach --busid <BUSID>
```

5. Check device in WSL:

```
lsusb
dmesg | tail
```

Your device should now be visible in WSL (e.g., as ttyACM0 or /dev/ttyUSB0).

6. Run the application in WSL as shown above.

Usage

1. **Start the application:**

   ```
   python main.py
   ```

2. **Connect to the sensor:**
   - Click on the connect button to open the connection window
   - Use the connection window to select the serial port for the QCM sensor and the PT100 probe.
   - Click "Connect". The software will establish the connection and verify the devices.

3. **Configure measurement parameters:**
   - Click on the settings button to open the settings window.
   - Access the settings window.
   - Set the number of measurements, frequency range, step size, harmonics to measure, waiting time between measurements, and calibration options.

4. **Start measurements:**

- Click “Start” to begin data acquisition.
- The main window displays real-time plots of amplitude and phase for calibration and measured harmonics, as well as frequency and temperature over time.

5. **View and analyze data:**
    - Use the data visualization window to plot any measured parameter on the X and Y axes.
    - Select which harmonic to display.
6. **Save data:**
    - After measurements, save the results in XML format for further analysis.

Data Format

- **Commands to the sensor:** Sent as strings with start frequency, end frequency, and step size, separated by semicolons and ending with a newline.

  ```
  500000;55990000;10000\n
  ```

- **Sensor response:** Amplitude and phase values separated by semicolons, each point on a new line. End of measurement is indicated by temperature followed by `;s`.

  ```
  2968.94;3450.61\n
  25.81;s
  ```

Signal Processing

- **Filtering:** Savitzky-Golay filter is applied to smooth the signal.
- **Interpolation:** Improves measurement precision.
- **Peak Detection:** Resonance peaks are detected by finding the maximum amplitude.
- **Dissipation Calculation:** Calculated from the resonance peak bandwidth or, if available, from the time constant.

Troubleshooting

- If connection fails, an error message is displayed. Check the serial port and device connections.
- The software uses multi-threading to keep the interface responsive during measurements.