NADC24 and M258 Power Measurement

Example Code Introduction for 32-bit NuMicro® Family

Document Information

Application	This example code uses NADC24 and M258 for power measurement.
BSP Version	M251_M252_M254_M256_M258_Series_BSP_CMSIS_V3.02.005
Hardware	NADC24+PSU_Evaluation_Kit_V1.0

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller and microprocessor based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

1. Overview

This example code uses the NADC24 series analog-to-digital converter and M258 series microcontroller (MCU) with voltage and current sensing circuits to quickly develop power monitoring related products and display power information such as voltage/current/power in real time.

1.1 Principle

When the PSU power signal is processed by the voltage divider resistor and current transformer, small voltage signals are generated. These signals are inputted to NADC24 in a differential mode. The NADC24 has an internal Delta-Sigma converter that converts analog voltage signals into digital ones. The M258 series is used to set NADC24 and read digital signals through the SPI interface, and then perform calculations to display line frequency, voltage RMS value, current RMS value, apparent power, active power, reactive power, and power factor and other related parameters.

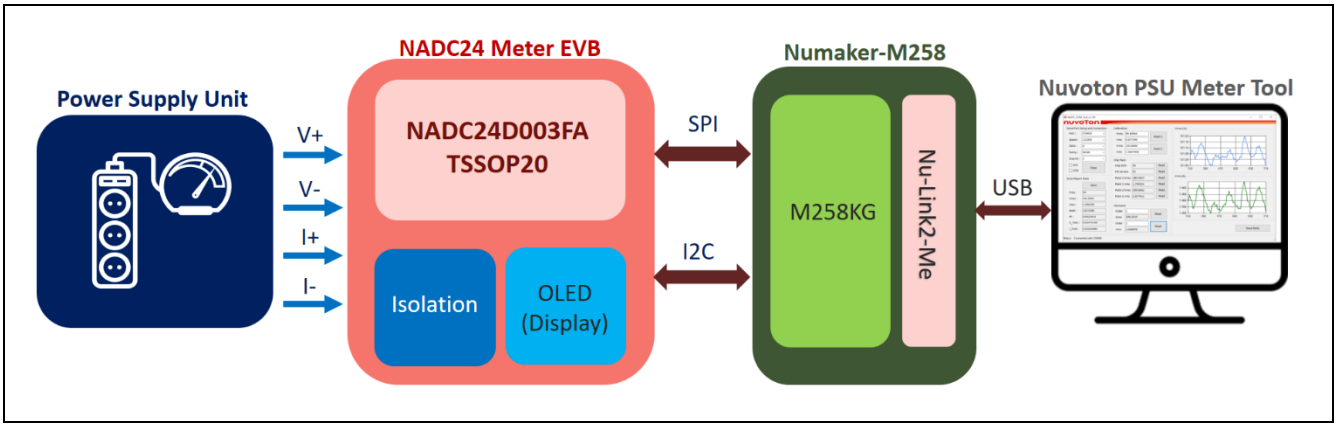

Figure 1-1 Block Diagram of Power Measurement

1.2 NADC24 Overview

The NADC24 is a high-precision 24-bit analog-to-digital converter whose main function is to read the voltage and current signals from the power supply. By setting the data output rate to 96 KSPS and the internal reference voltage to 2.4V, the MCU can provide stability and reliability in power calculations.

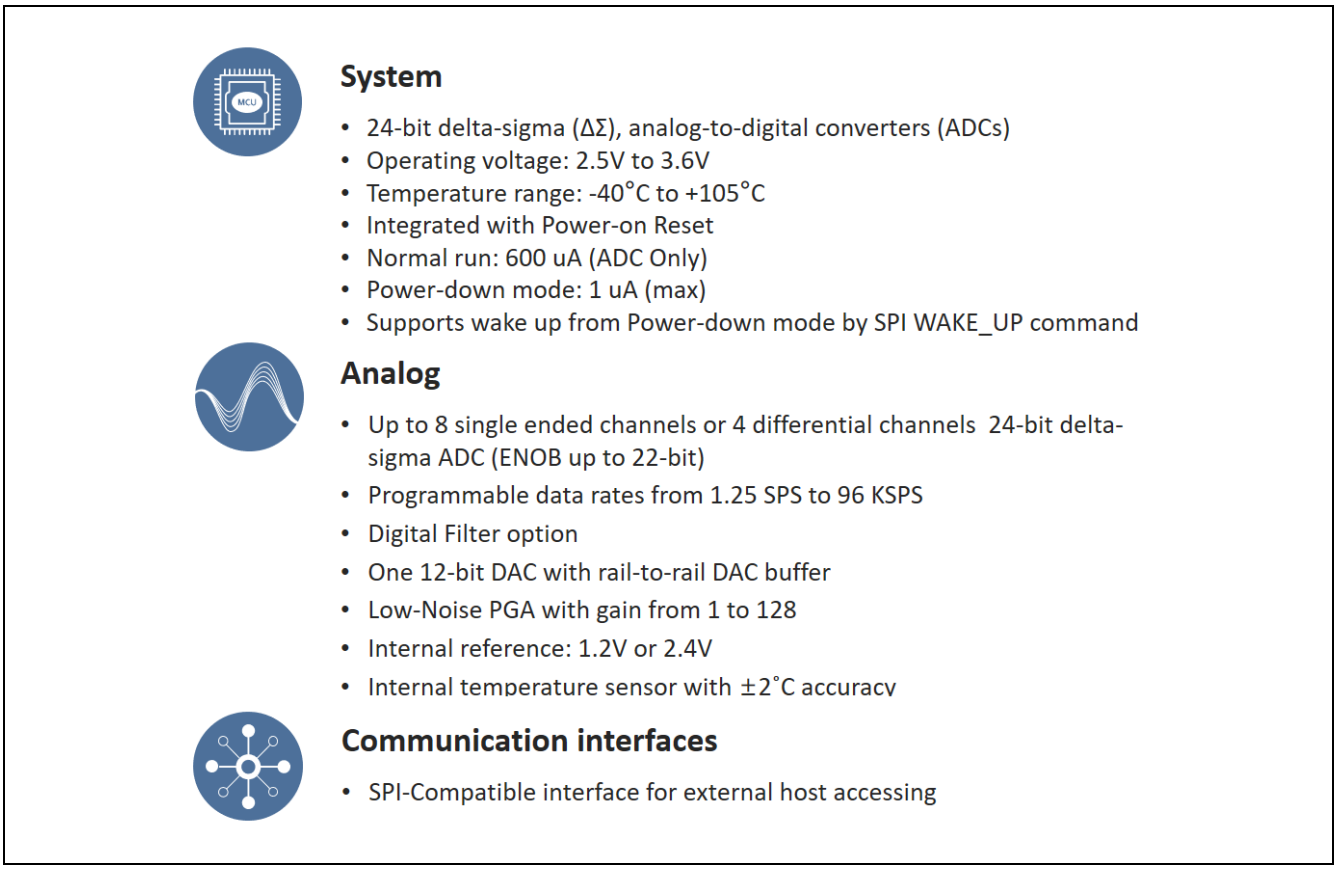

Figure 1-2 NADC24 Specification

1.3 M258 Overview

The M258 series includes the Arm Cortex-M23 core, operates at 48 MHz with Flash of 256 KB, SRAM of 32 KB, and supports interfaces for COM/SEG LCD Driver, UART, I²C, SPI, and other peripherals. The M258 reads the values converted by NADC24 through the SPI interface and then performs calculations. It can display data such as line frequency, RMS voltage, RMS current, apparent power, active power, reactive power, power factor, etc.

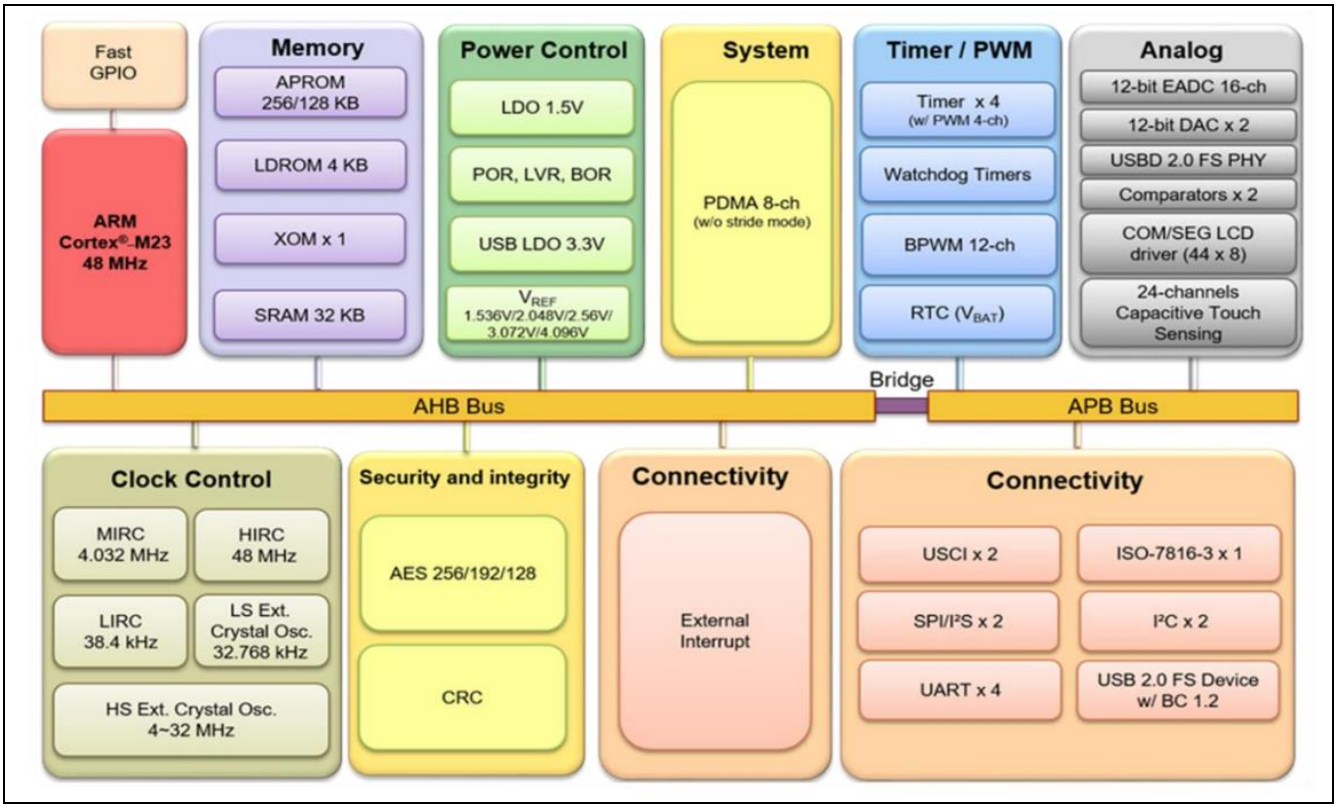

Figure 1-3 M258 Series Block Diagram

1.4 Evaluation Board Description

First, connect the NADC24+PSU_Evaluation_Kit (EVB) to the 110V mains and household appliances. Next, connect the NuMaker-M258KG to the computer via USB. The OLED screen on the EVB can be utilized to display real-time power consumption data of household appliances; or through computer tools, real-time values and curves can be presented.

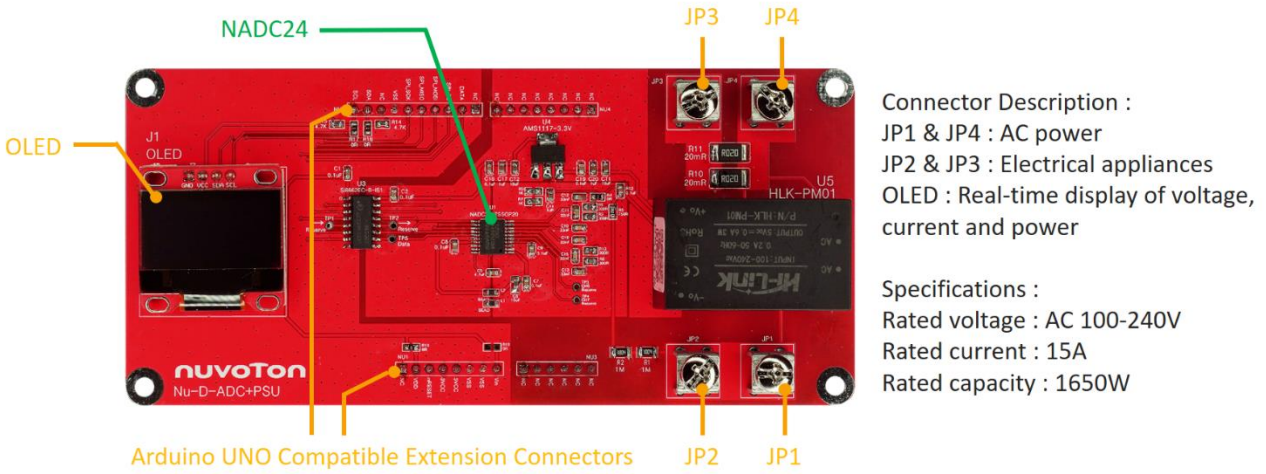

Figure 1-4 EVB Description

2. Code Description

Initialize system frequency, I2C0 for LCD and UART for debugging.

```
int main(void)
{
    uint32_t Data_Addr;
    int32_t U32_Data_1, U32_Data_2, U32_Data_3, U32_Data_4, U32_Data_5, U32_Data_6;

    /* Unlock protected registers */
    SYS_UnlockReg();

    /* Init System, IP clock and multi-function I/O. */
    SYS_Init();

    /* Configure UART0: 115200, 8-bit word, no parity bit, 1 stop bit. */
    UART_Open(UART0, 115200);

    /* Init I2C0 */
    I2C0_Init();
    LCD_Disp_Info();
```

Initialize NADC24 and set PA4 to data ready.

```
    /* Initial NADC24 */
    Reset_NADC24();
    NADC24_Calibration_and_Initial(CALI_NADC24);
    NADC24_Calibration_and_Initial(INIT_NADC24);

    /* Enable UART RDA and THRE interrupt */
    NVIC_EnableIRQ(UART0_IRQn);
    UART_EnableInt(UART0, UART_INTEN_RDAIEN_Msk);
    UART0->FIFO = ((UART0->FIFO & (~UART_FIFO_RFITL_Msk)) | UART_FIFO_RFITL_1BYTE);

    /* IO Interrupt */
    GPIO_SetMode(PA, BIT4, GPIO_MODE_INPUT);
    GPIO_EnableInt(PA, 4, GPIO_INT_FALLING);
    NVIC_EnableIRQ(GPA_IRQn);

    /* Interrupt Priority */
    NVIC_SetPriority(SPI0_IRQn, 0) ;
    NVIC_SetPriority(GPA_IRQn, 1) ;
    NVIC_SetPriority(UART0_IRQn, 2) ;
    NVIC_SetPriority(TMR0_IRQn, 3) ;
    NVIC_SetPriority(I2C0_IRQn, 3) ;
```

Initialize FFT and read the calibration data.

```
    /* FFT initial */
    arm_rfft_fast_instance_f32 S;
    arm_rfft_fast_init_f32(&S, SAMPLE_CYCLE_512);

    /* Read Calibration value to Variable */
    Read_Data_from_APROM(&Read_Cali_Value[0]);
    V1_Rms_Calibration = Read_Cali_Value[0];
    I1_Rms_Calibration = Read_Cali_Value[1];
    V2_Rms_Calibration = Read_Cali_Value[2];
    I2_Rms_Calibration = Read_Cali_Value[3];
    V1_DC_Calibration  = Read_Cali_Value[4];
    I1_DC_Calibration  = Read_Cali_Value[5];
    V2_DC_Calibration  = Read_Cali_Value[6];
    I2_DC_Calibration  = Read_Cali_Value[7];
    V_RMS_OFFSET_1     = Read_Cali_Value[8];
    I_RMS_OFFSET_1     = Read_Cali_Value[9];
    V_RMS_OFFSET_2     = Read_Cali_Value[10];
    I_RMS_OFFSET_2     = Read_Cali_Value[11];
    V_RMS_GAIN         = Read_Cali_Value[12];
    I_RMS_GAIN         = Read_Cali_Value[13];
    V_DC_OFFSET_1      = Read_Cali_Value[14];
    I_DC_OFFSET_1      = Read_Cali_Value[15];
    V_DC_OFFSET_2      = Read_Cali_Value[16];
    I_DC_OFFSET_2      = Read_Cali_Value[17];
    V_DC_GAIN          = Read_Cali_Value[18];
    I_DC_GAIN          = Read_Cali_Value[19];
```

Calculations can be performed after preparing 512 pieces of ADC data.

```
while(1)
{
    if(Read_512_Start_Flag==0)
    {
        /* Calibration */
        if((VI_Calibration_Falg)||(DC_Calibration_Falg))
        {
            ……
        }

        if(Freq_Calc_Falg)
        {
            ……
        }

        /* Re-start saving */
        Read_512_Start_Flag = 1;
    }
```

```
    if(Power_Measurement_Falg)
    {
        /* 50Hz signal */
        if(Read_50Hz_Done_Flag && Enble_50Hz_Cycle_Flag)
        {
            ……
        }

        /* 60Hz & DC signal */
        else if(Read_60Hz_Done_Flag && (Enble_60Hz_Cycle_Flag || Enble_DC_Cycle_Flag))
        {
            ……
        }
    }

    /* Command analysis */
    if(Input_Command_Flag)
    {
        ……
    }
    else
    {
        ……
    }

    /* Finish */
    Input_Command_Flag = 0;
}
```

3. Software and Hardware Requirements

3.1 Software Requirements

- BSP version
  - M251_M252_M254_M256_M258_Series_BSP_CMSIS_V3.02.005
- IDE version
  - Keil uVersion 5.36

3.2 Hardware Requirements

- Circuit components
  - NADC24+PSU_Evaluation_Kit_V1.0
  - NuMaker-M258KG
- Pin Connect
  - Two boards NADC24+PSU_EVB and NK-M258KG are connected through the Arduino interface.
  - Connect the UART0 TX (PB.13) pin to the PC UART RX to show the execution results of this example code.

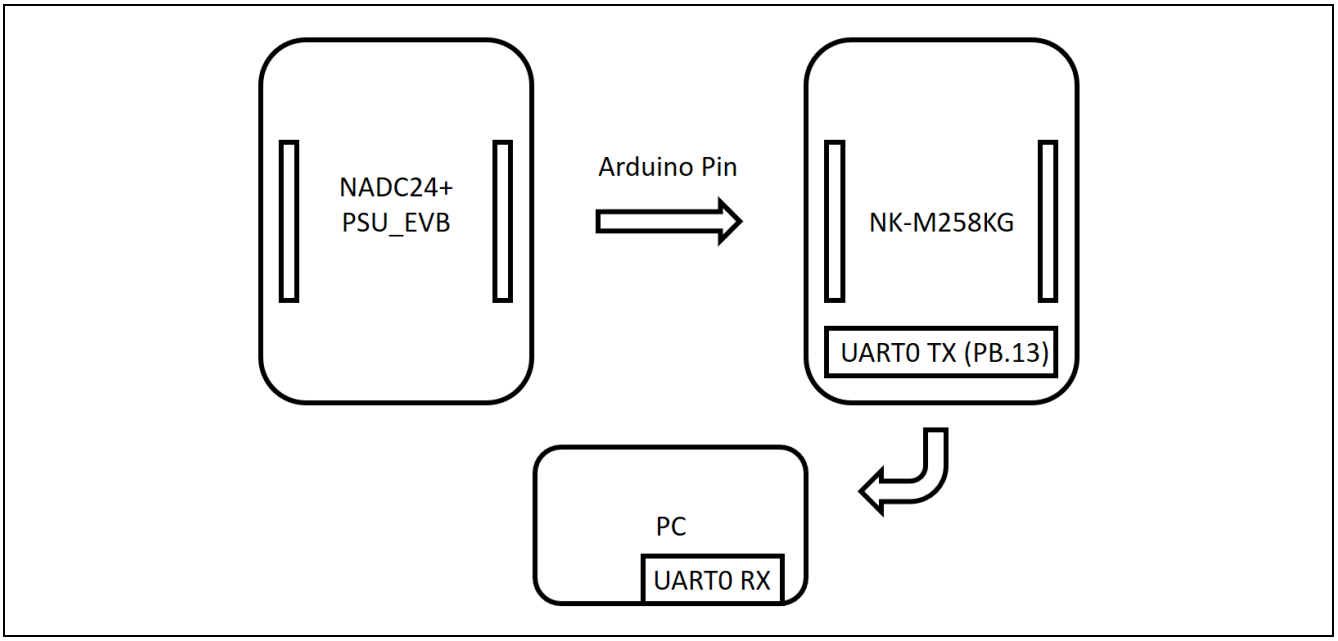

Figure 3-1 Pin Connect

4. Directory Information

The directory structure is shown below.

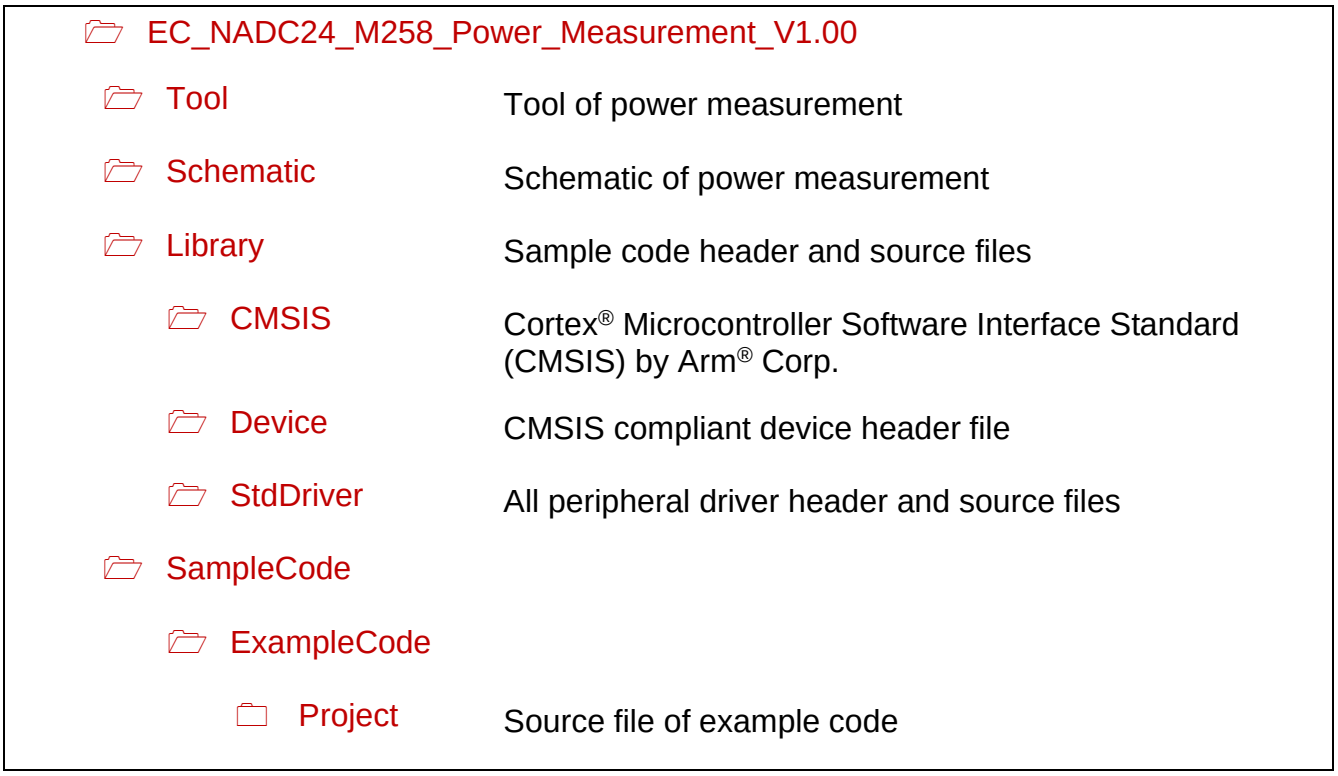

Figure 4-1 Directory Structure

5. Example Code Execution

1. Browse the sample code folder as described in the Directory Information section and double-click NADC24_M258_Power_Measurement.uvprojx.

2. Enter Keil compile mode.

   - Build
   - Download
   - Start/Stop debug session

3. Open NADC_COM_Tool_v1_00.exe and click '**Open**' to connect to the serial port.

Note: Your computer needs to have Microsoft .NET Framework 4.7.2 installed.

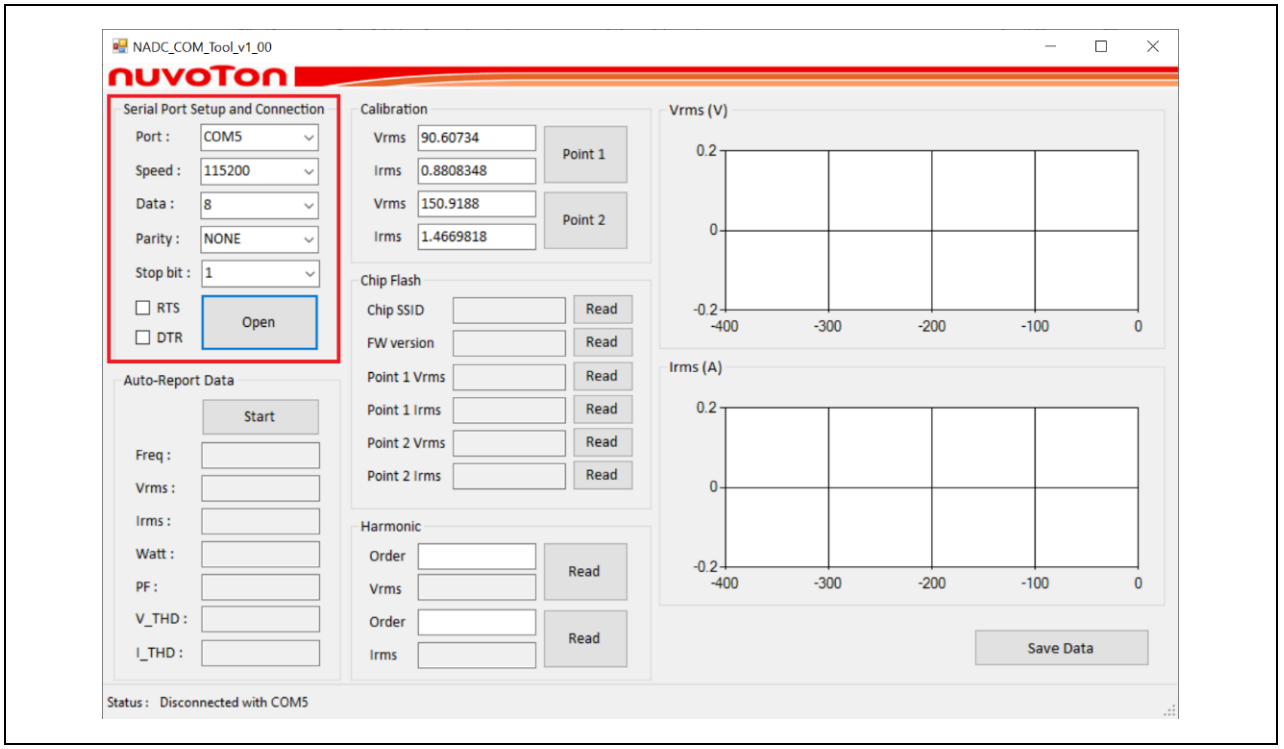

Figure 5-1 Serial Port Connection

4. Select two points for calibration based on the application. As an example, the following takes 90V and 150V. After connecting 100 Ω, high power, set 90V, press Point 1, wait for a few seconds, and the verification value will be displayed on Point 1 Vrms and Point 1 Irms. After setting 150V, press Point 2 and wait for a few seconds. The verification values will be displayed on Point 2 Vrms and Point 2 Irms to complete the calibration.

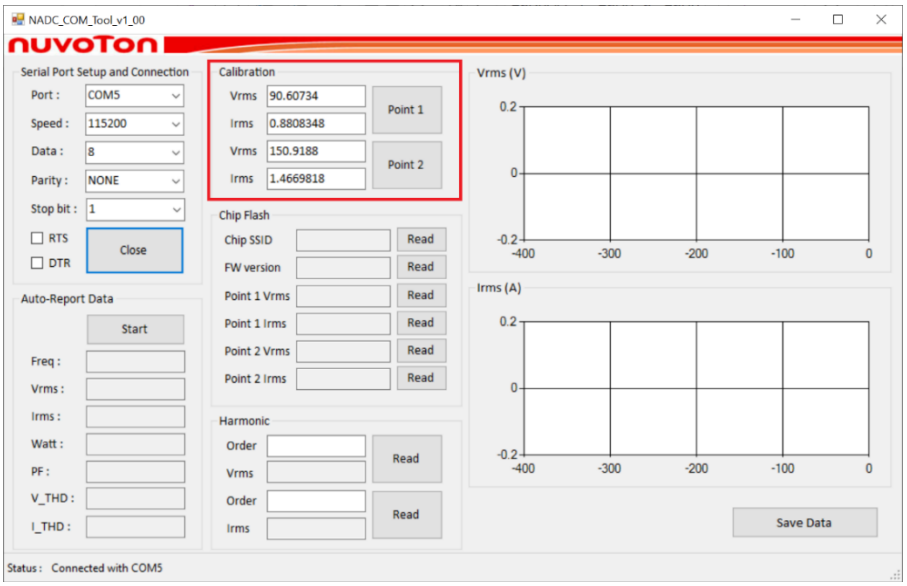

Figure 5-2 Calibration Mode

5. After clicking '**Start**', start measurement.

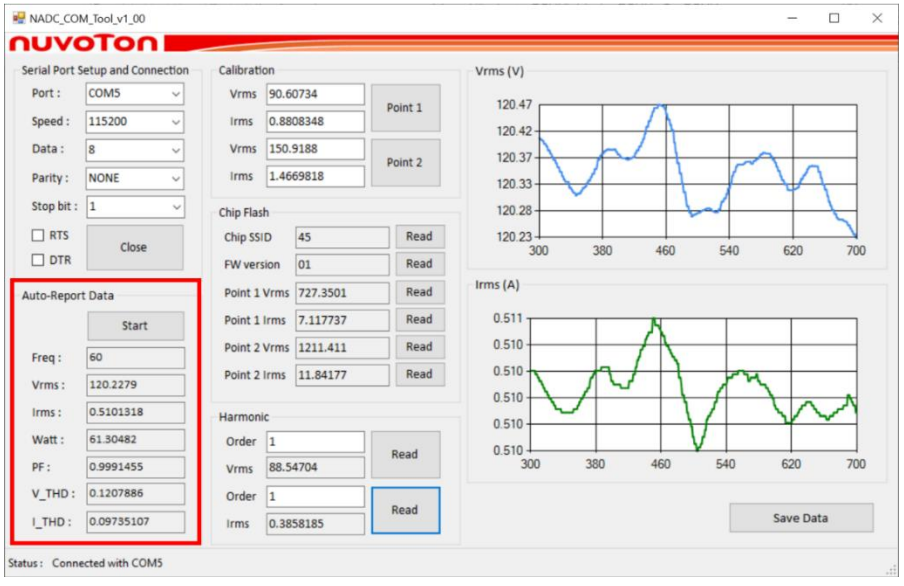

Figure 5-3 Auto-report Data

6. Check Vrms and Irms via curves and save them.

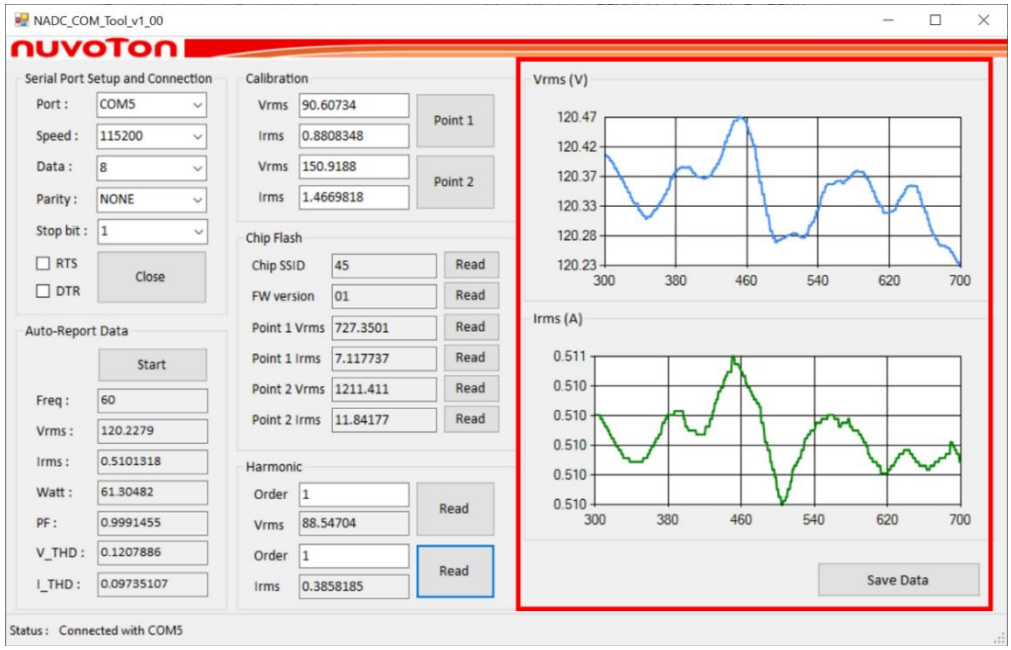

Figure 5-4 Curve of Vrms and Irms

6. Revision History

Date	Revision	Description
2023.11.06	1.00	Initial version.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.