CN317-ABSIPSPOCZ Absolute Inductive Position Sensor

User's Manual

Rev.1.00
2022.7.15

1. What is the CN317-ABSIPSPOCZ?

The CN317-ABSIPSPOCZ Motor Encoder Solution is a proof of concept (POC) that combines two inductive position sensor IPS2200s and a high cost-efficient microcontroller RX24T to realize the function of a simple rotary encoder. RX24T is responsible for measuring the differential sine and cosine outputs from IPS2200s, converting them into an absolute position and output it. It has the advantage of flexible design, good performance, cost-effective, lighter, thinner, and total stray field immune.

The figure below is the block diagram for the system described.

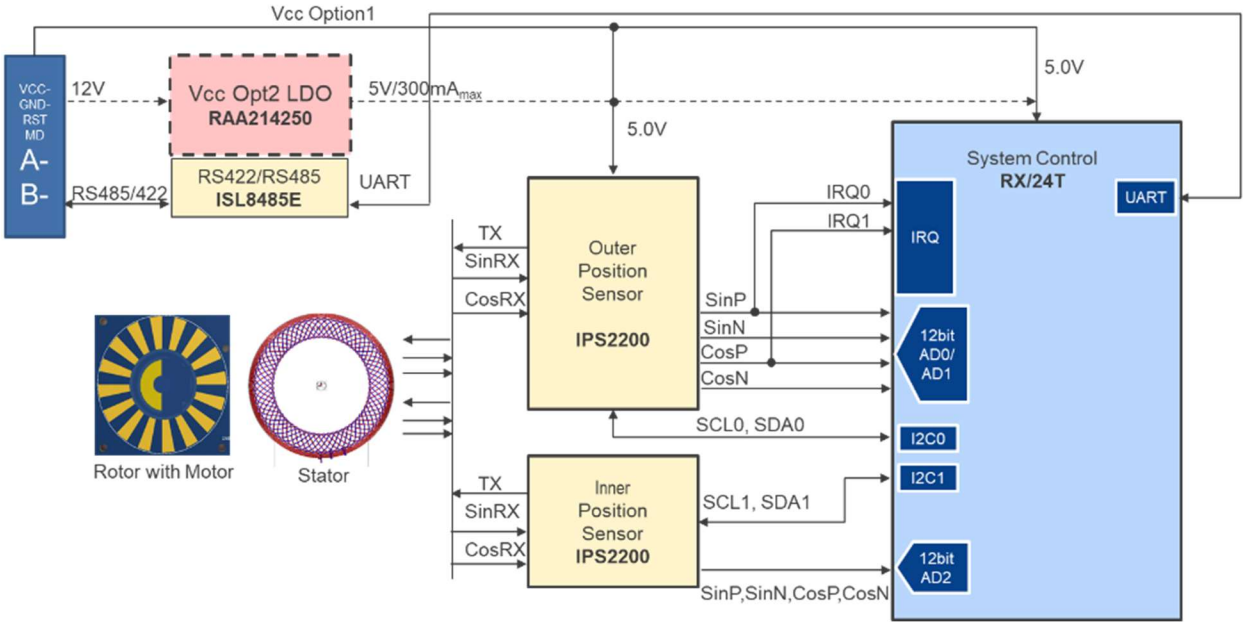


Figure 1-1: System Block Diagram

2. Table of Contents

3. Objective

The purpose of this solution is to demonstrate the performance that can be achieved by using the position sensor IPS2200 and the MCU RX24T combined as a rotary encoder, so that users can refer to it when developing their own project.

- **Resolution: 15-bit**

The resolution in this solution refers to the stable ability to obtain an position value at the same position regardless of the output rate.

- **Accuracy: 12-bit**

The accuracy is obtained by taking one actual encoder product (approximately 15-bit accuracy) on the market as a benchmark to do a simple calibration and comparing the calibrated position with it.

- **Baud rate of RS485: 5Mbps (MAX.)**

The max baud rate supported by RX/24T can reach up to 5Mbps. But the maximum baud rate supported by the USB-to-Serial converter used in the solution is 1.25Mbps, so the baud rate is set to 1.25Mbps in this sample code.

- **RS485 Output rate: 20 kHz**

The output rate depends on the output format and the baud rate, which will be explained in detail below.

- **ABZ output: 2500-line**

As the frequency of RX/24T is only 80MHz, it is not capable of handling higher lines.

4. Quick Start Up

Table 4-1 is shown the switches setting of the board.

Table 4-1 Switches Setting of Board

SW	Setting		Function
S1	1 = ON	2 = OFF	ABZ Output
	1 = OFF	2 = OFF	RS485 GUI Demo
	1 = OFF	2 = ON	RS485 Output

Table 4-2 is shown the status of LEDs.

Table 4-2 Status of LEDs

LED	Description
LED1 (White LED)	Breathing light.
LED2 (Blue LED)	ON: ABZ Output mode OFF: RS485 GUI Demo mode or RS485 Output mode
LED3 (Red LED)	Power LED

The whole demo requires two power supplies. The encoder solution is 5V power supply by micro-USB (MUSB). The motor is 12V power supply.

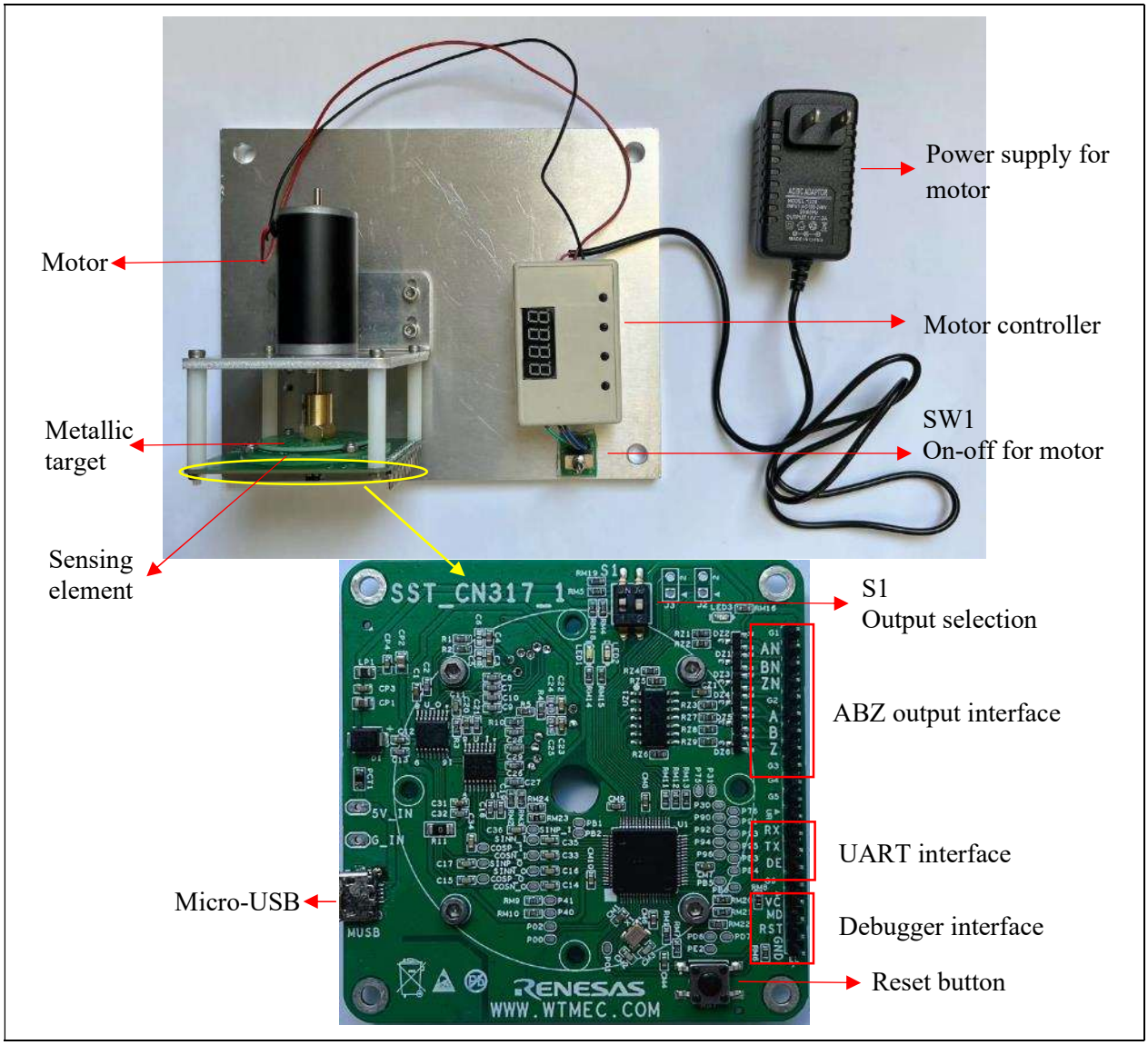


Figure 4-1: Overview of Demo

4.1. ABZ Output

Reset the board after set S1-1 to ON and S1-2 to OFF. Monitor the ABZ signals on the oscilloscope.

Turn on the motor. When the metallic target spins, we can get the figure as shown in the below.

Figure 4-2: ABZ Signals

Due to the limit of RX/24T processing speed, the maximum rotate speed supported is 4000rpm.

Figure 4-3 shows the comparison between this solution and an actual encoder product when rotating the metallic target by hand.

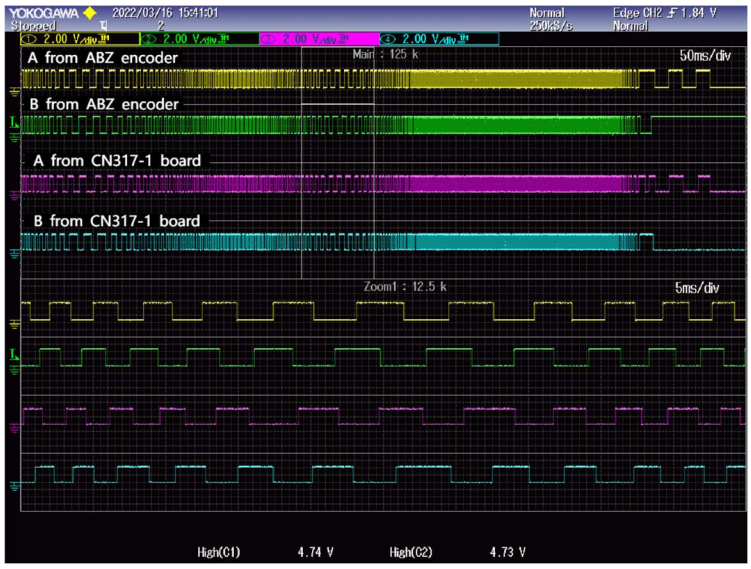


Figure 4-3: Comparison Result

4.1.1 GUI for ABZ Output Demo

If it is not convenient to use oscilloscope to monitor, we also prepared a converter board as a decoder to show the demo in another way.

The connection between encoder and decoder is as follows:

	Encoder	**Decoder**
Connection	A	PC6 (J2-2)
	B	PC7 (J2-3)
	Z	PB1 (J2-4)

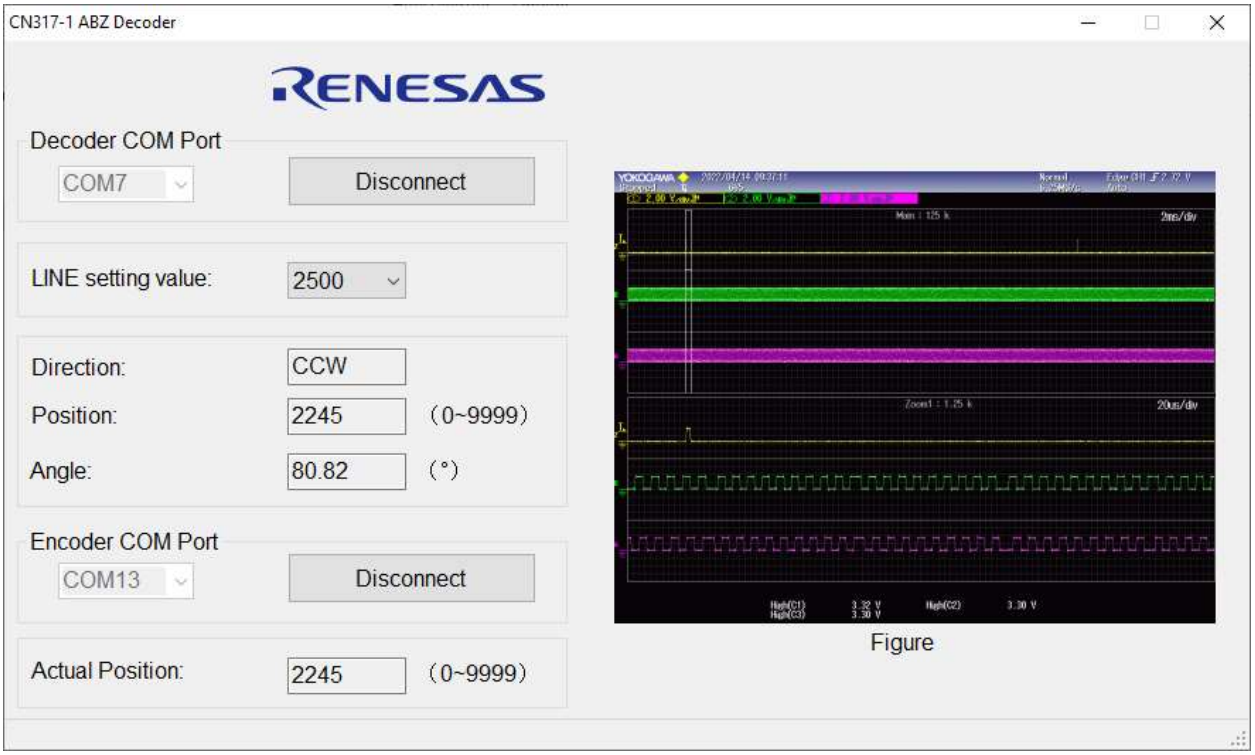
Since the decoder board is powered by 3.3V and the encoder board is powered by 5V, a level-shifter is required between the decoder and the encoder.

If the level-shifter is not used, the following problems may occur:
(1). Decoding result is not correct occasionally.
(2). The decoder board is damaged.

Connect the decoder (J6) and PC through USB cable. LED2 start blinking after Z signal is received. Open “GUI_CN317-1 ABZ Decoder.exe”. Select the COM port of this decoder. Click “Connect” button. The GUI can receive a real-time absolute position value after the metallic target has rotated past the absolute zero position.

At the same time, the encoder board is outputting the position through TX pin. Select the COM port of the encoder. Click “Connect” button. The GUI will display the actual absolute position calculated by RX/24T.

If the position value from decoder is the same as that from encoder, that indicates the ABZ signals outputted from encoder are correct.



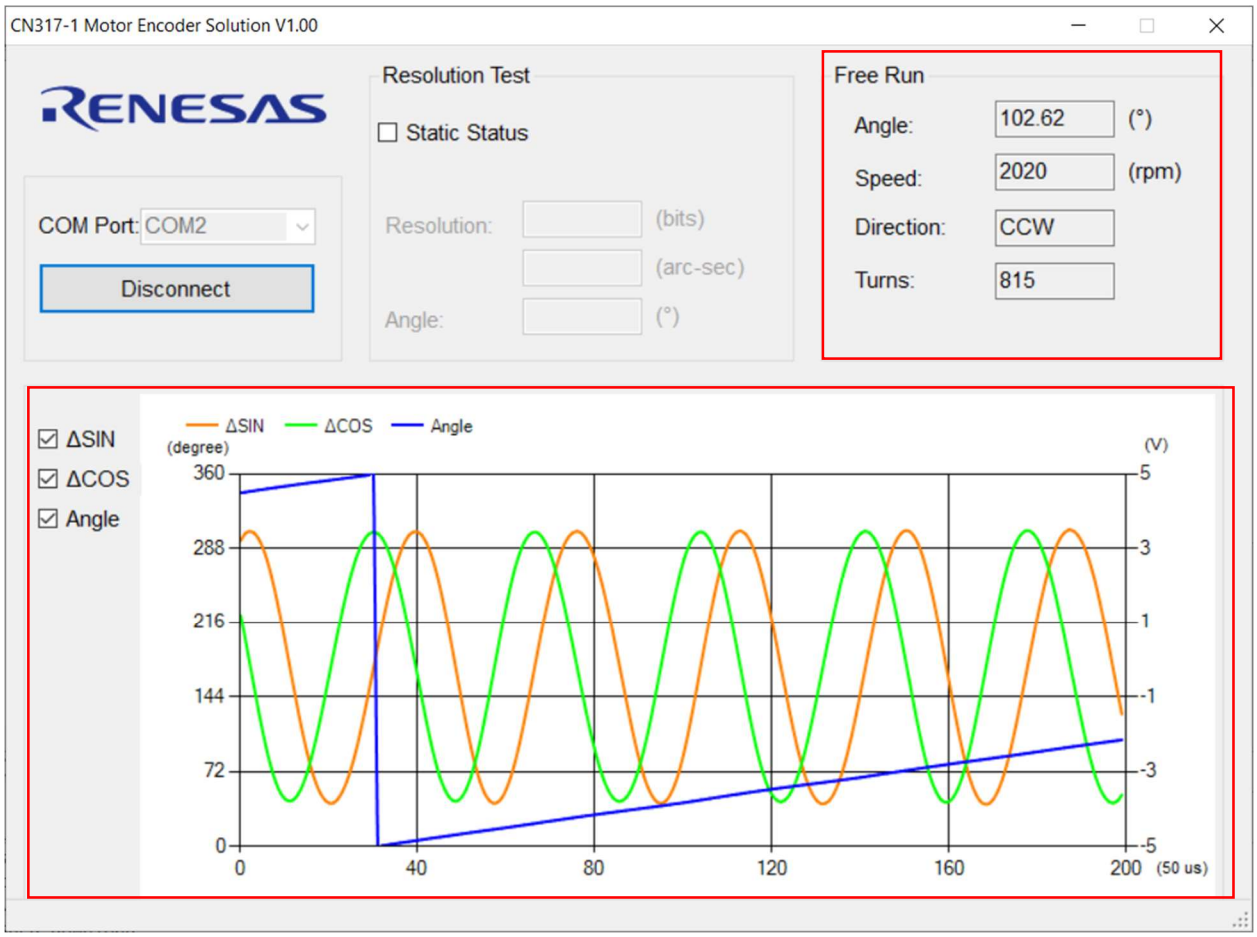
Figure

4.2. RS485 GUI Demo

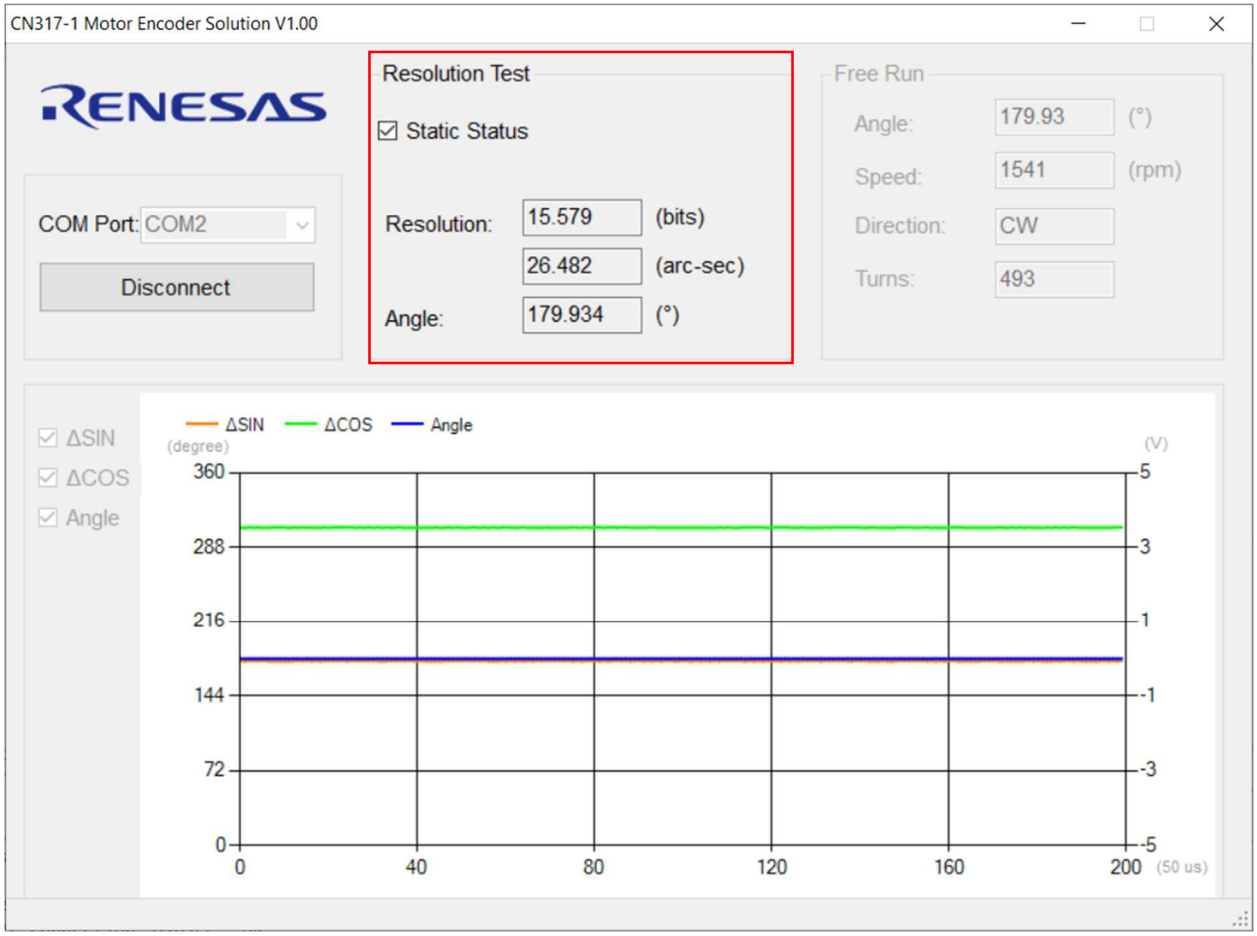
(1) Reset the board after set both of S1-1 and S1-2 to OFF.

(2) Connect to the PC through a USB-to-serial converter (TX, RX, GND).

(3) Open “GUI_CN317-1 Motor Encoder Solution.exe”. Select the COM port and click on “Connect” button. Now the system is running in Free Run mode. We can see the Angle (absolute position), the rotate speed of the motor, the direction of rotation, the number of turns, and a chart that showing the differential sine and cosine curves and the corresponding Angle. The MCU records the differential sine value, the differential cosine value, and the Angle value every 50us and sends 200 sets to PC at one time. So, the chart shows them for the last 1ms.



(4) Check the Static Status checkbox in Resolution Test after the motor stops. The system is running in Static Resolution Test mode. It takes about 40 seconds to obtain the static resolution by analyzing 1000 pieces of data.

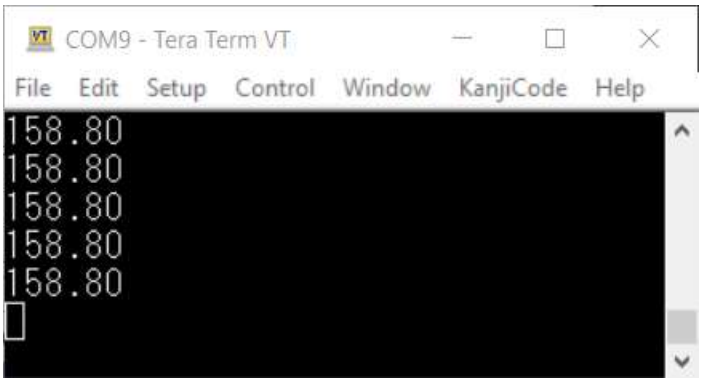


4.3. RS485 Output

Reset the board after set S1-1 to OFF and S1-2 to ON.

At present there is no RS485 chip on our board, this solution just uses UART (PD3 as TXD1, PD5 as RXD1) to output. A USB-to-serial converter is needed.

Open “teraterm” on the PC. Select the corresponding COM port and set the baud rate to 1.25Mbps. The window displays the absolute position value as the following figure shows.



Since the USB-to-serial converter used supports a maximum baud rate of 1.25Mbps, the baud rate in the solution is set to 1.25Mbps. It needs to take 64us to output 8 bytes. Take “158.80” in the figure above as an example. The output string is ‘158.80\r\n’ (‘\r’ is for ‘carriage return’, ‘\n’ is for ‘line feed’). The output bytes are ‘0x31’, ‘0x35’, ‘0x38’, ‘0x2E’, ‘0x38’, ‘0x30’, ‘0x0D’, and ‘0x0A’.

The output rate is set to 10kHz (output one position every 100us). The delay time is 50us (get a new position value every 50us).

5. System Overview

5.1. IPS2200 Setting

The registers of IPS2200 need to be initialized to fit the mechanical design. Two IPS2200s are used. The inner IPS2200 is for absolute position detection. The outer IPS2200 is for higher resolution. The function IPS2200_Register_Setting() is used to config both outer IPS2200 and inner IPS2200 sensors. It will maximize the Sine & Cosine waveform amplitude (peak-valley), adjust the offset value to align the center of the Sine & Cosine and calibrate the mechanical error (air gap/ inclination/ asymmetric circle) to keep them in appropriate phase (90-Angel differential input).

The Tx current of inner IPS2200 needs to be set to 0 to prevent the crosstalk between the two coils.

5.2. Initial Position

During the initialization, the metallic target is stationary. The differential signals of the inner IPS2200 and the outer IPS2200 are detected. On the metallic target, one sector is designed for inner coil and 16 sectors for outer coil. So, an absolute position [0-360) is calculated from the inner differential signals. It can be used to determine which sector of the outer coil the metallic target is in. Then a higher resolution absolute position can be calculated from the outer differential signals.

5.3. Position Update

After the initial position is obtained, the differential signals of the inner IPS2200 are no longer needed to be detected. The differential signals of the outer IPS2200 are detected every 50us to calculate the new position value.

6. Design Specification

6.1. Position Sensing

The IPS2200 is a magnet-free, inductive position sensor IC that can be used for high-speed absolute position sensing in industrial, medical, and consumer applications.

6.1.1 Programming Interface

In this solution, the pin (CSN_IRQN) is used to selecting the I2C slave address by hardware.

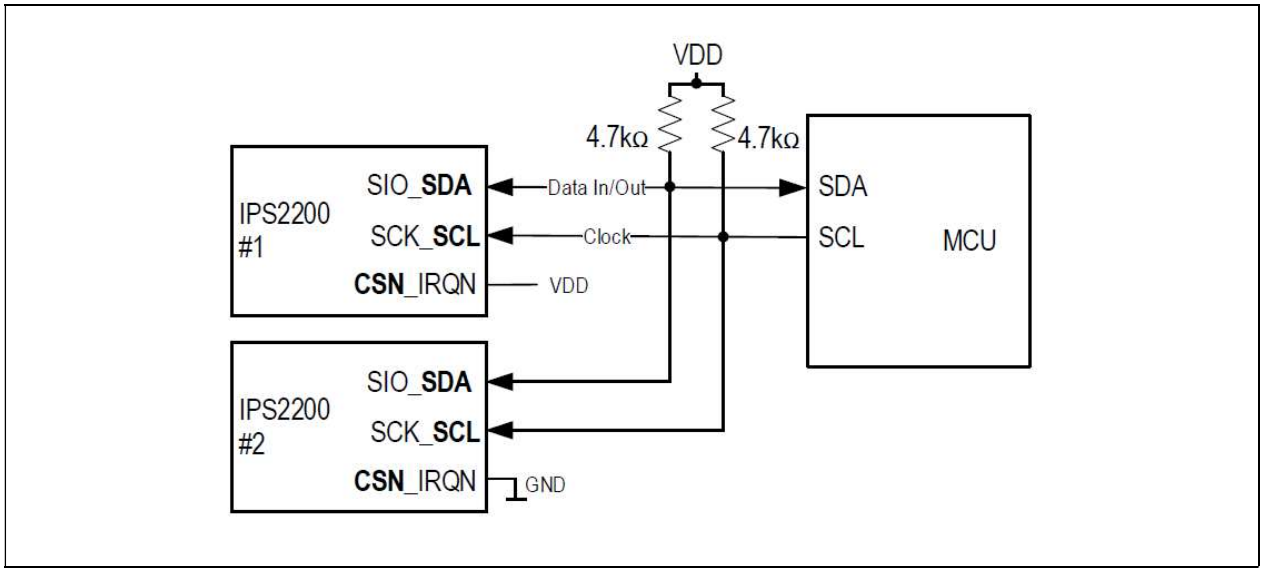


Figure 6-1: I2C Interface with Address Select

The slave address of IPS2200 #1 is 0x30. The slave address of IPS2200 #2 is 0x20.

Please refer to “4.4.1 I2C Data Format” in Programming Guide for IPS2200 (REN_IPS2200-Prog-GDE_xxxxxxxx.pdf).

6.1.2 Output Mode

The IPS2200 has three output modes: (1) Analog differential sine-cosine analog output mode. (2) Analog single ended sine-cosine analog output mode. (3) Digital incremental differential AB mode. In this solution, for higher resolution, analog differential sine-cosine analog output mode is selected.

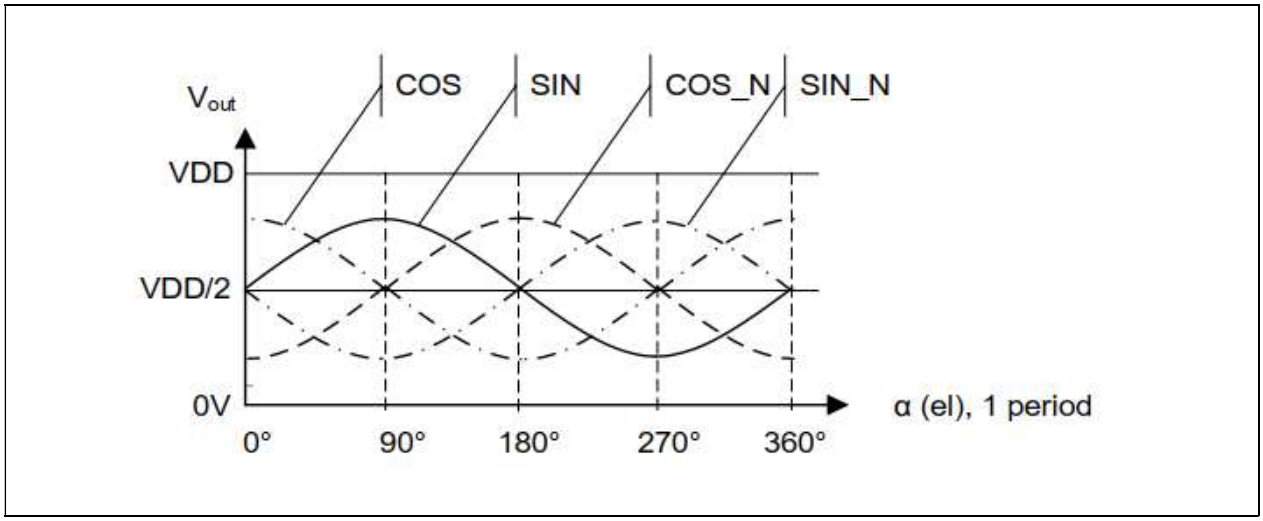


Figure 6-2: Sine-Cosine Analog Mode Output Signals

The COS, SIN, COS_N, SIN_N signals need to be sampled by four A/D channels simultaneously. The A/D of RX24T has sampling hold function and synchronous trigger, that can meet the above requirements.

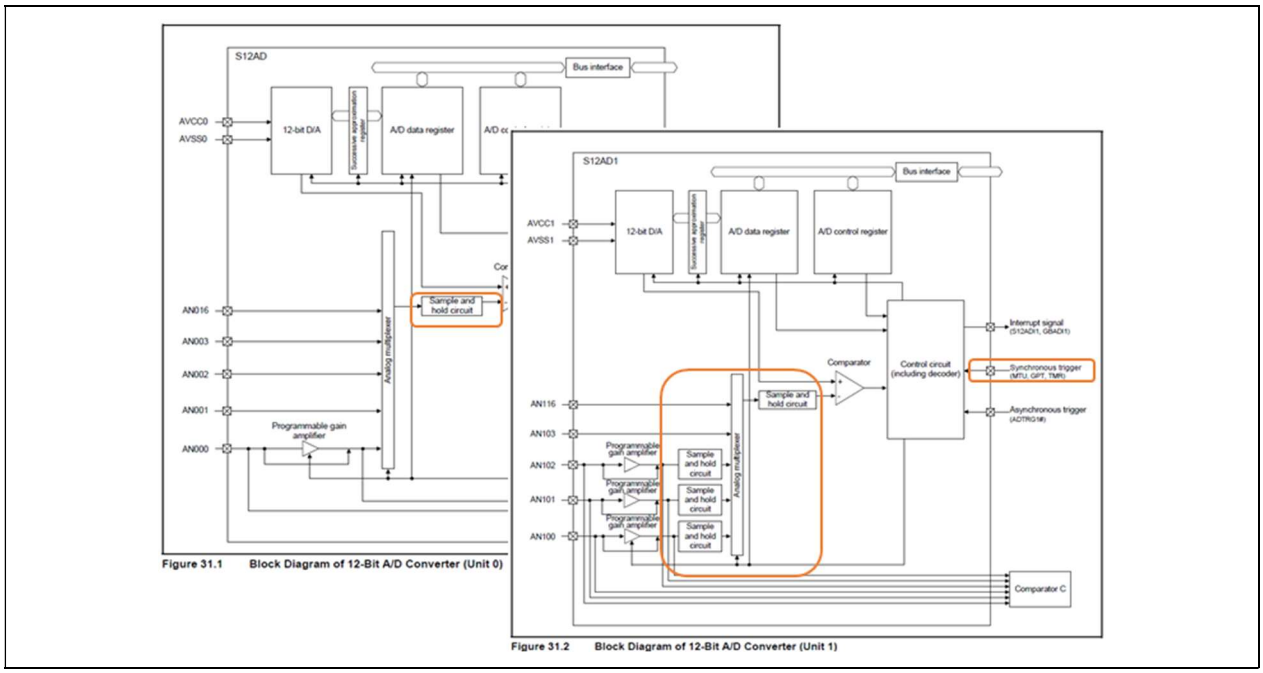


Figure 6-3: Block Diagram of 12-Bit A/D Converter

6.1.3 Sector Design

Currently, Renesas can offer designs for 1 to 32 sectors on the metallic target. The more sector, the higher the resolution, the larger the size, and the higher the processing technology requirement.

In this solution, 16 sectors are designed for outer coil and one sector is designed for inner coil. One-sector design is for the absolute position, 16-sector design is for the higher resolution.

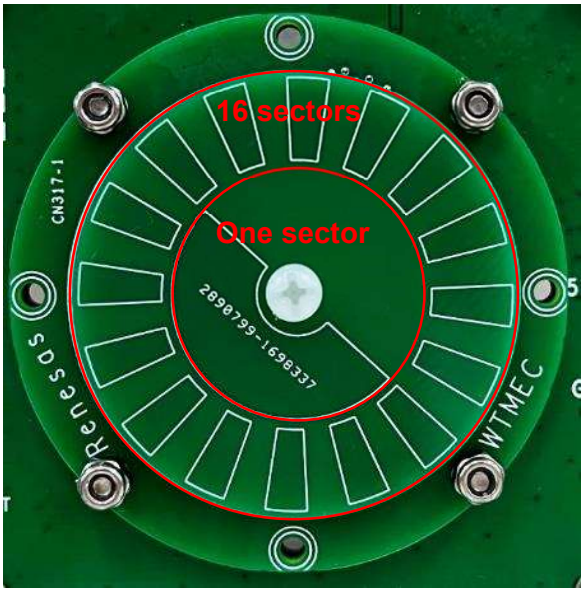


Figure 6-4: Coils

16-sector means differential sine and cosine signals repeat 16 times over one full 360-degree mechanical rotation.

6.1.4 Absolute Position Calculation

The signals can be converted into an electrical Angle value by applying an arctangent operation of delta-Vsin and delta-Vcos.

result = arctan (delta-Vsin / deltat-Vcos) * 180 / PI()

If result < 0, then

electrical Angle value = result + 360

else

electrical Angle value = result

During the initialization, the inner electrical Angle value and the outer electrical Angle value are obtained.

sector number = round ((inner electrical Angle value * 16 – outer electrical Angle value) / 360)

If sector number is equal to 16, then

sector number = 0

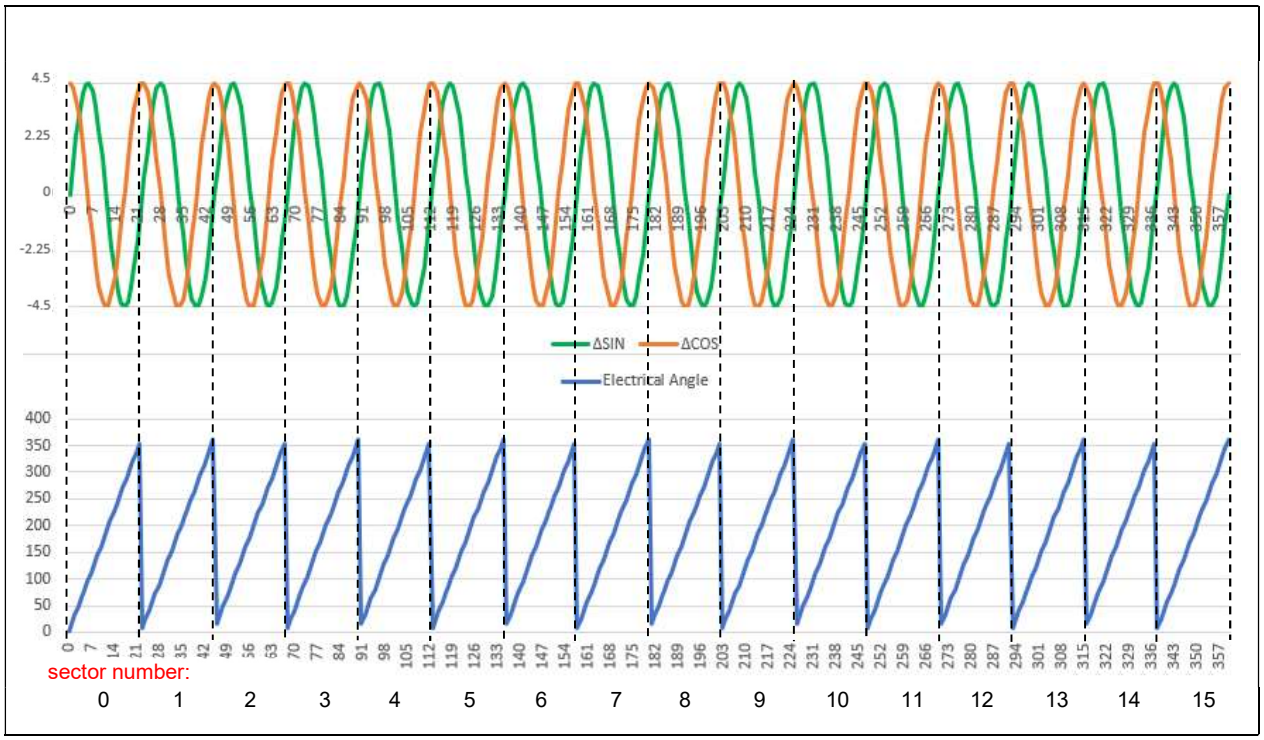


Figure 6-5: Delta Sine-Cosine Signals and Electrical Angle (Outer)

The outer electrical Angle value is obtained every 50us. After getting the electrical Angle value, the mechanical Angle value can be calculated based on the sector number.

If the outer electrical Angle value is much larger than the last one

sector number = sector number – 1

If the outer electrical Angle value is much less than the last one

sector number = sector number + 1

(The method is applicable when the motor speed is less than 75000rpm)

And the mechanical Angle value can be calculated.

mechanical Angle value = (outer electrical Angle value + 360 * sector num) / 16

6.2. Trigonometric Function

Arctangent function need to be used in this solution. Various methods exist to compute the arctangent function. There include atan2f function in Math Lib of CCRX, CORDIC algorithm, lookup table method. We tested these methods, and the result is given as follows.

Table 6-1 Trigonometric Function Test Result

Condition	Method	Accuracy (+16 sectors)	Time (MAX.)	Occupied ROM size
MCU RX24T (80MHz)	atan2f function in Math Lib of CCRX	13.5 (+4) bits	7 us	1K ROM
	CORDIC (15 iteration)	13.5 (+4) bits	9 us	<1K ROM
	CORDIC (16 iteration)	13.6 (+4) bits	9 us	<1K ROM
	CORDIC (17 iteration)	13.4 (+4) bits	10 us	<1K ROM
	Lookup table (array of 1024 elements)	11.7 (+4) bits	3 us	4K ROM
	Lookup table (array of 2048 elements)	12.5 (+4) bits	3 us	8K ROM
	Lookup table (array of 4096 elements)	13.0 (+4) bits	3 us	16K ROM

Test data: [0, 360), in 12-bit format, 360 data, 1 degree per step

Note that the test results using different MCUs are different, because of the different clock frequency and Math Libs. For example, if use RA2A1 MCU, the accuracy of CORDIC algorithm is higher, and the cost time is shorter than atan2f function in Math Lib.

Considering the accuracy, the consumption time, and the occupied ROM size, the atan2f function in Math Lib is used in this solution.

6.3. Resolution

As mentioned before, 16 sectors are designed for outer of metallic target. And we use the built-in A/D converter in RX/24T that has 12-bit resolution. Assuming that the max voltages of SINP, SINN, COSP, COSN signals are exactly equal to the reference voltage of ADC. Then the ADC value [0, 4096) corresponds to a position [0, 90) degree.

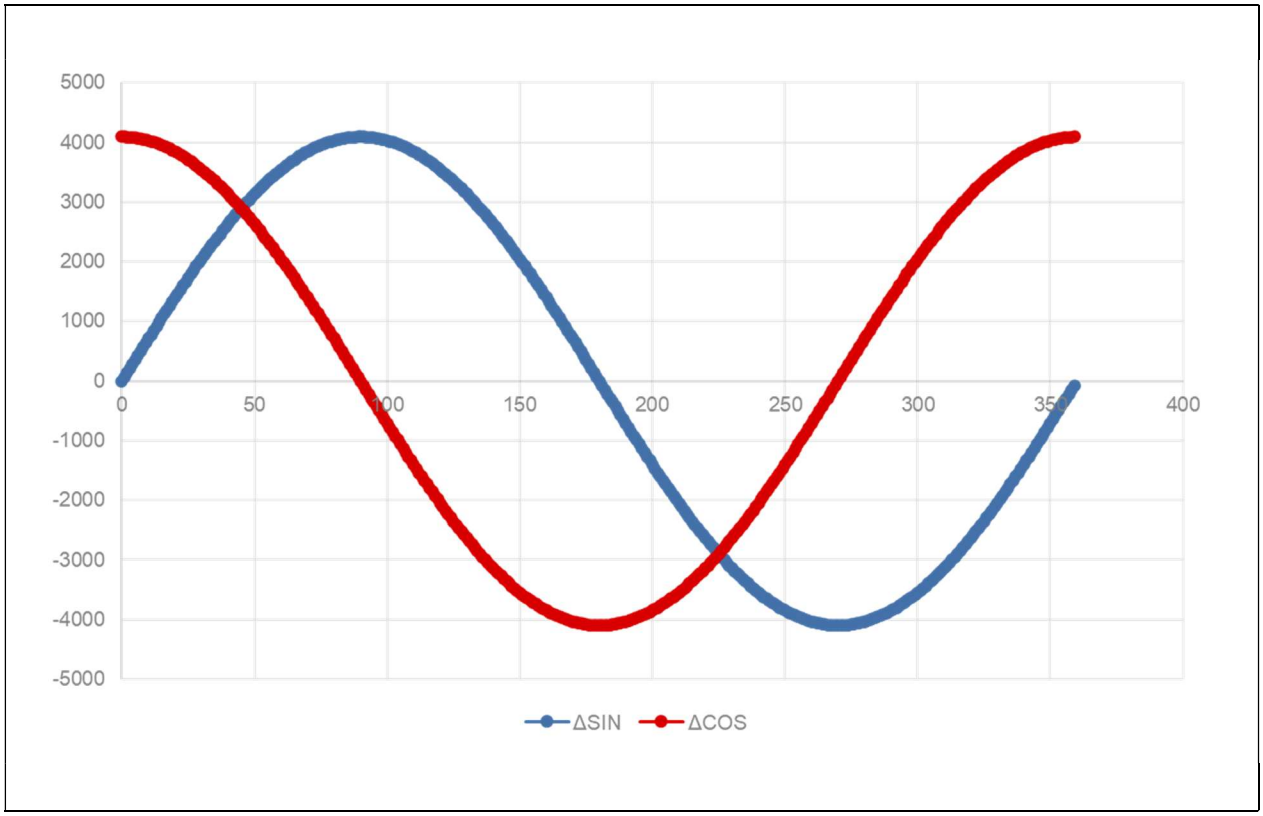


Figure 6-6: Delta Sine-Cosine Signals and A/D Converter Value

Theoretical highest resolution

= log2(16 * 2^12 * (360 / 90)) = 18-bit

Using a higher resolution A/D converter can improve the resolution of the motor encoder solution as long as the accuracy of SINP, SINN, COSP, COSN signals is higher than the resolution of ADC.

6.4. Output

In this solution, one UART interface is used for data output and communication with GUI through a USB-to-serial converter.

The setting for UART is [1250000, 8, N, 1]. In the communication protocol between RX24T and GUI, 3 bytes of data represents one absolute position value.

The cost time of each operation of the program is measured simply by a timer.

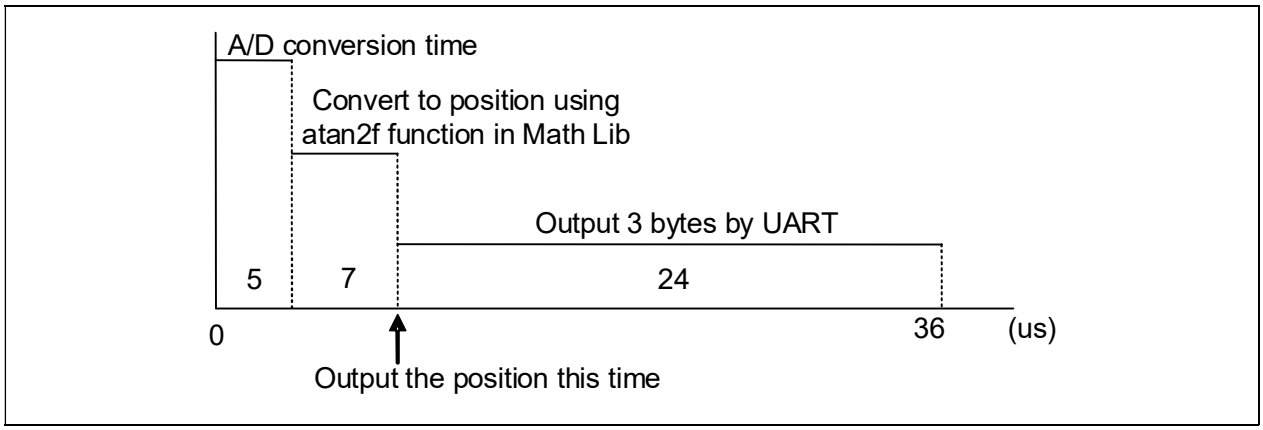


Figure 6-7: Timing without DTC

In fact, a data transfer controller (DTC) incorporated in RX24T is used to reduce the bus time taken by transmission by performing data transfers through UART ports.

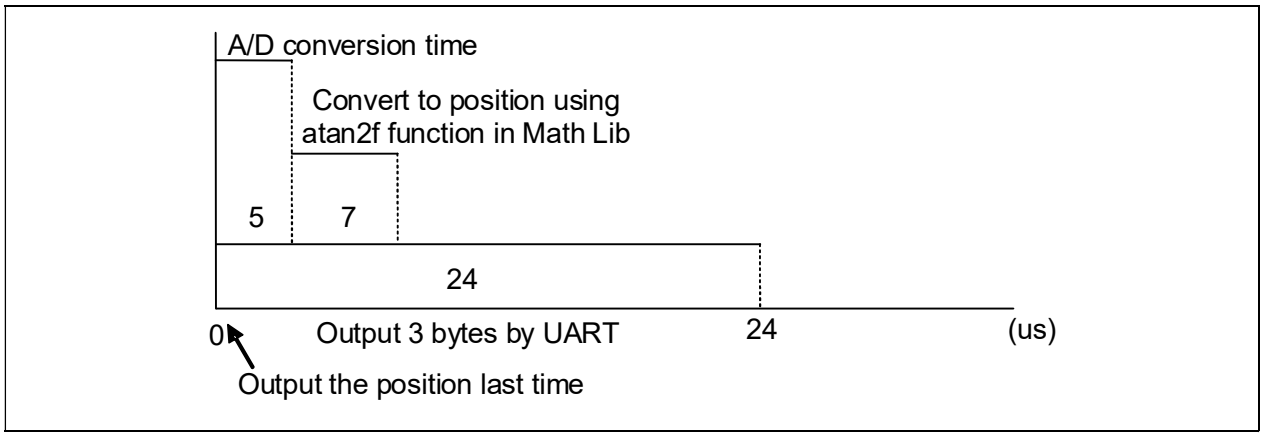


Figure 6-8: Timing with DTC

Since the timer interval of MTU0 (a timer) for triggering A/D converter is set to 50us, the output rate is 20kHz. As shown in the figure, it only takes 36us to output a position. If the timer interval is set to 40us, the output rate can be increased to 25kHz.

6.5. Accuracy in Calibration

The linear interpolation method is one option to calibrate the accuracy. The calibration parameters of each board are different under different assembly. So, this chapter only introduces the accuracy calibration method. Users can refer to this method to complete their own calibration.

We bought a rotary encoder product (absolute, 18-bit resolution, 15-bit accuracy, RS485 output) as a benchmark. In the period where sector number = 15, obtained 38 positions at random to generate a chart with the electrical Angle as X axis and the ERR (the difference between the mechanical Angle value and the reference Angle value from the encoder product) as Y axis.

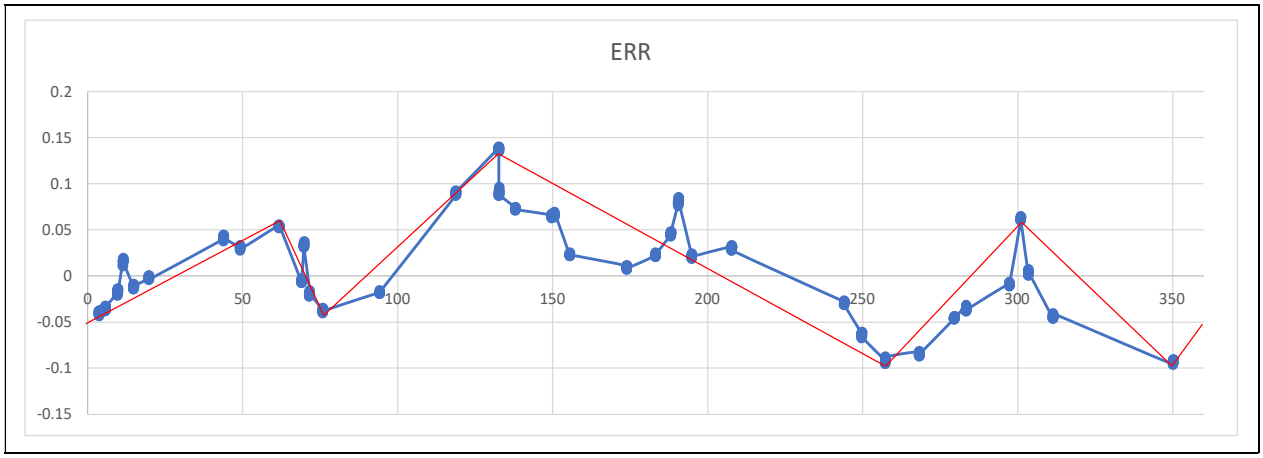


Figure 6-9: ERR Chart

As shown in the figure above, eight points of them were selected to be connected to get the red lines.

E_Angle	ERR
0	-0.05
62	0.052
76	-0.04
133	0.135
257	-0.095
301	0.061
350	-0.096
360	-0.05

Using these 8 points, we can figure out the equations of these 7 lines.

Range	**ERR = k * E_Anlge + b**	
	k	b
0-62	0.00165	-0.050
62-76	-0.00657	0.459
76-133	0.00307	-0.273
133-257	-0.00185	0.382
257-301	0.00355	-1.006
301-350	-0.0032	1.025
350-360	0.0046	-1.706

After obtaining the electrical Angle value and the mechanical Angle value, firstly, judge which range the electrical Angle is in. And then calculate the ERR according to the corresponding equation. The difference value of the mechanical Angle value minus the ERR is the calibrated mechanical Angle value.

Of course, the more position points, the more red lines, and the higher accuracy will be.

7. Hardware

The solution consists of 2 parts. The main part is the main board as an encoder. The other part is the motor and controller, which help demonstrate this solution.

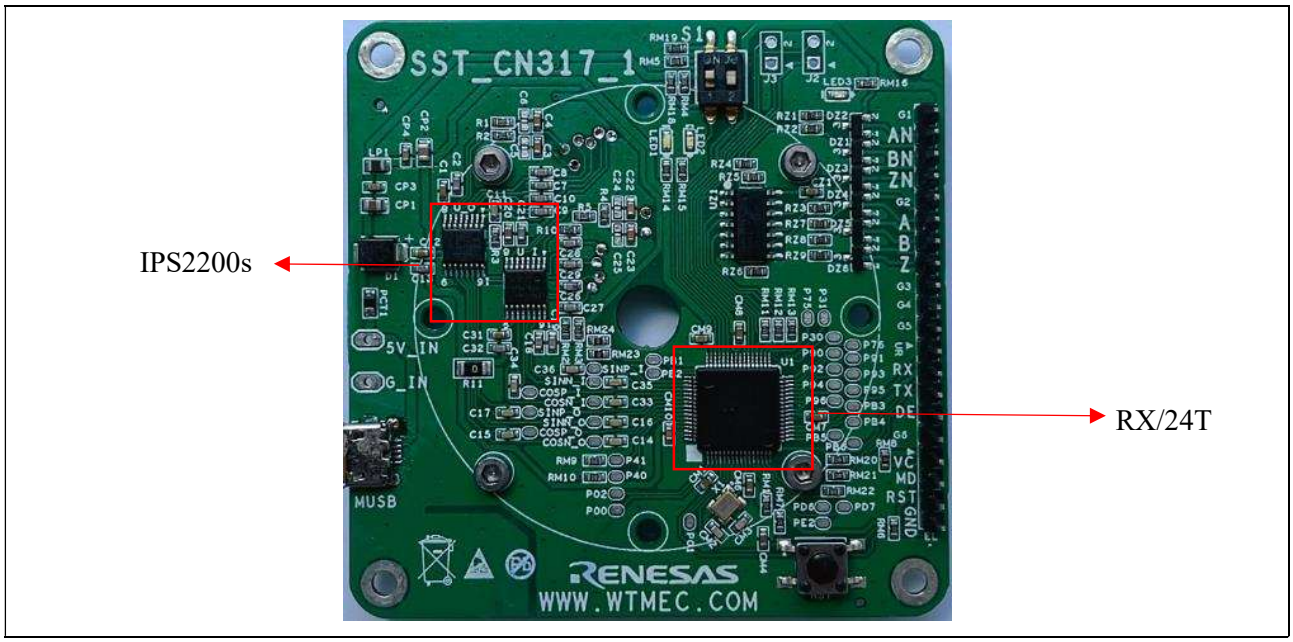


Figure 7-1: Top of Main Board

Figure 7-2: Bottom of Main Board

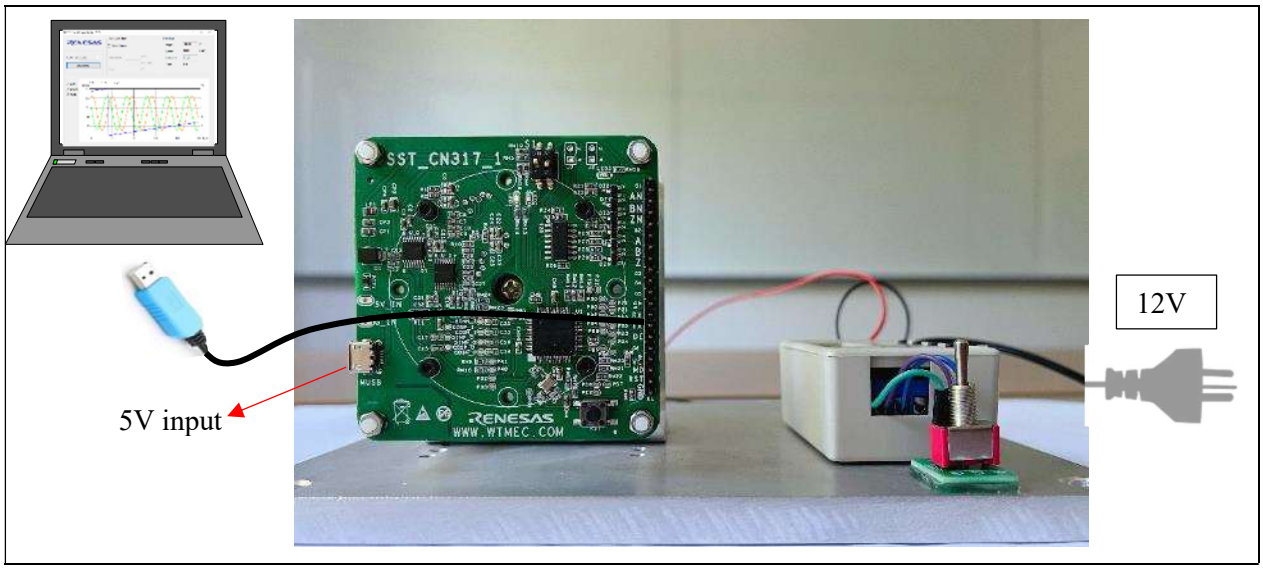


Figure 7-3: Whole system

8. Software

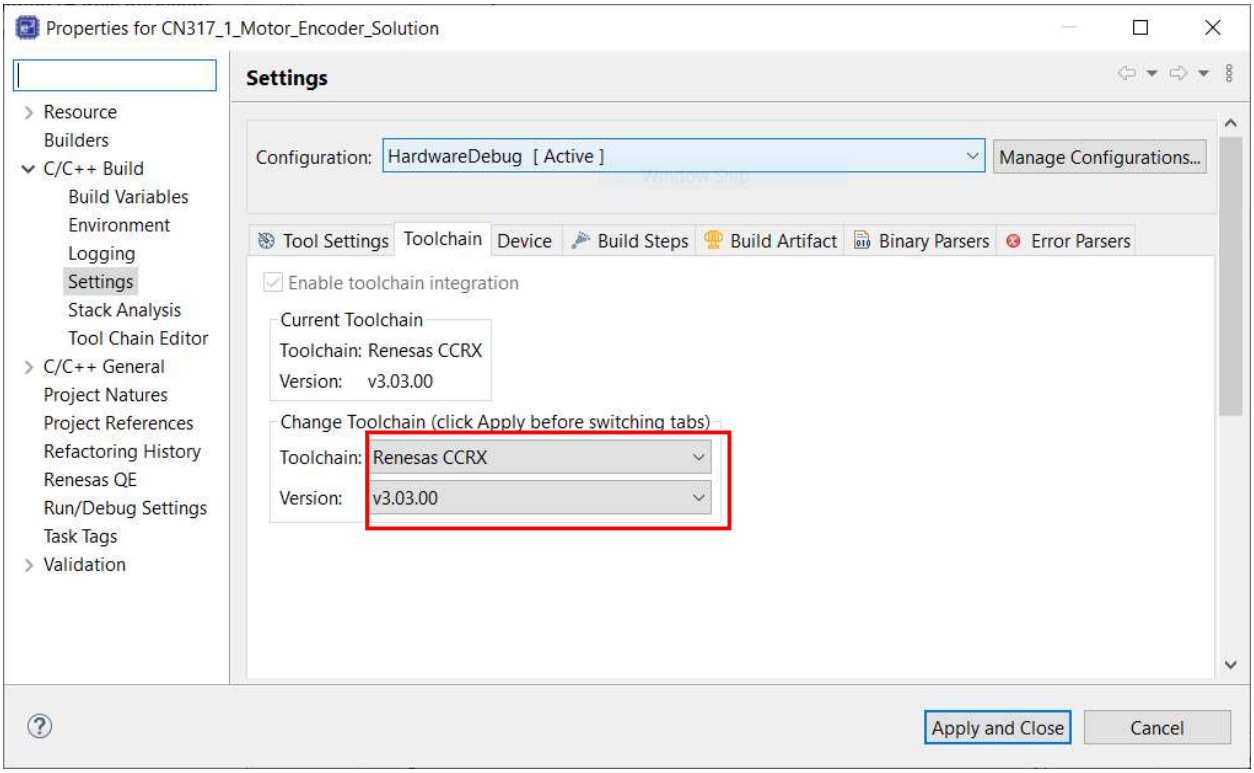
8.1. Integrated Development Environment

The sample code has been developed under the conditions listed in the table below.

Table 8-1 Operation Conditions

Item	Description
Microcontroller	RX24T (R5F524TAADFM)
Operating frequency	Resonator frequency: 20MHz FCLK: 20MHz ICLK/PCLKA: 80MHz PCLKB/PCLKD: 40MHz
Operating voltage	5.0V
Integrated development environment	e^2 studio V2021-10 from Renesas Electronics Corp.
Toolchain	CCRX V3.03.00

Note: If “CCRX V3.03.00” toolchain is not installed, the content in the red box below will be empty. There will be an error when build the project. It will be OK when rebuild the project after select the toolchain you already have installed.



8.2. MCU Functions and Pins to be Used

Table 8-2 MCU Functions and Pins to be Used

Peripheral Function	Pin	Usage	
DTC		Send/receive data to/from PC through SCI1	
S12AD0/S12AD1	AN002/P42	Measure the differential sine and cosine signals from the outer IPS2200	COS_N
	AN100/P44		COS_P
	AN101/P45		SIN_N
	AN102/P46		SIN_P
S12AD2	AN206/P50	Measure the differential sine and cosine signals from the inner IPS2200	COS_N
	AN207/P51		COS_P
	AN208/P52		SIN_N
	AN209/P53		SIN_P
PORT	P21	Drive a white LED	
	P22	Drive a blue LED	
	P23	Receive from DIP switch	
	P24	Receive from DIP switch	
	P70	Detect the level of P72 and P73 for testing	
	P71		
	PD4	Disable/Enable pin for RS485 chip	
	P74	Z signal	
MTU4 or PORT	P73	A signal	
	P72	B signal	
RIIC0	SCL0_PB1	Communication with the IPS2200 at 100kbps	
	SDA0_PB2		
SCI1	TXD1_PD3	Communication with GUI / Output the position at 1.25Mbps	
	RXD1_PD5		
MTU0		Trigger ADC once every 50us.	
MTU1		For toggling the ports (ABZ signals)	
TMR0		Rotate speed calculation	
TMR2_TMR3		2s period for measuring the SINP, SINN, COSP, COSN to calculate the IPS2200s	

8.3. Project

Table 8-3 The folder and file configurations of the project programs

Project scr folder	Sub Folder	Files	Description
src of CN317_1_Motor_Encoder_Solution	ips2200	ips2200.c	Source file of the IPS2200 NVM setting
		ips2200.h	Header file of the IPS2200 NVM setting
	smc_gen	--omitted	MCU peripherals settings by Smart Configuration
	--	rs485_demo.c	Source file of RS485 GUI demo
	--	rs485_demo.h	Header file of RS485 GUI demo
	--	main.c	main routine file
	--	userdefine.h	Header file of user define

8.3.1 Software Flowchart

Figure 8-1 shows the flowchart of ABZ output.

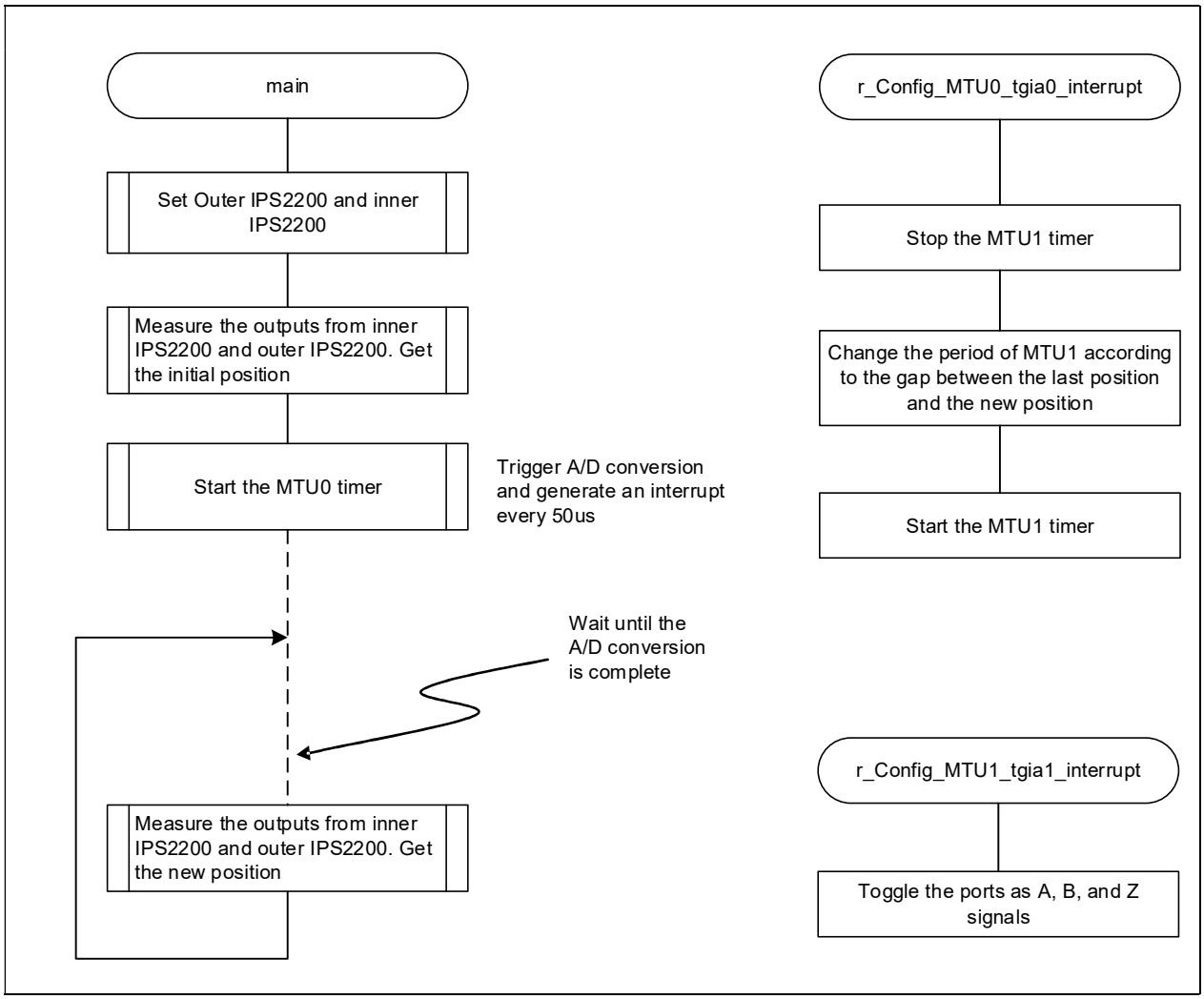


Figure 8-1: Flowchart of ABZ Output

Figure 8-2 shows the flowchart of RS485 GUI Demo.

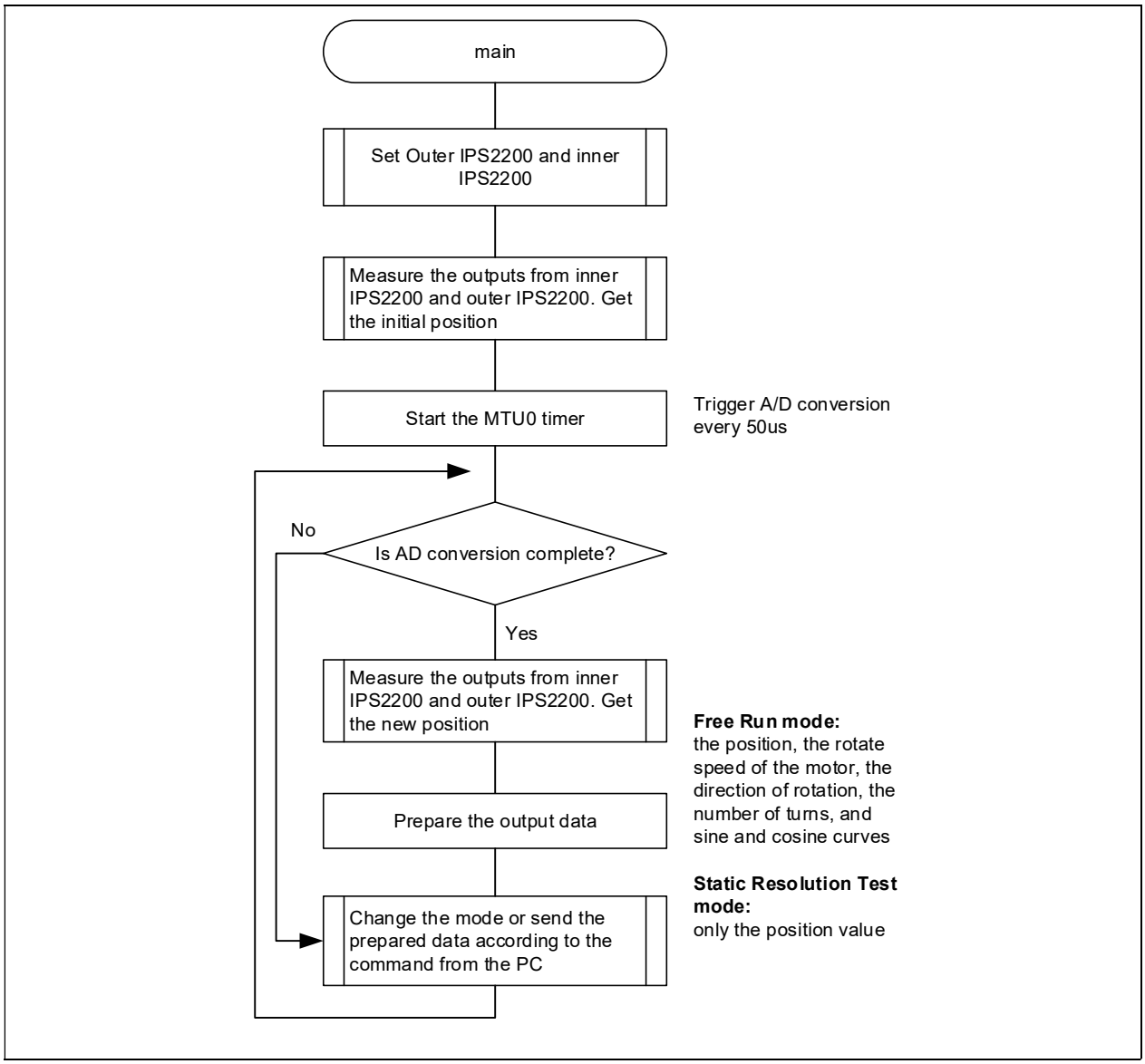


Figure 8-2: Flowchart of RS485 GUI Demo

8.4. Introduction about IPS2200 Calibration

The metallic target needs to be rotating at a constant speed while performing this step.

Standard Calibration Sequence:

1. TX Current Setup – Calculate the transmitter has bias current based on specified coil parameters. Please refer to “3.5.7 Transmitter Current Calibration” in Programming Guide for IPS2200 (REN_IPS2200-Prog-GDE_xxxxxxxx.pdf)

3.5.7. Transmitter Current Calibration

The Transmitter current bias (I_{BIAS}) influences the AC current of the LC oscillator. Increasing the AC current, increases proportionally the magnitude of the transmitter signal. The I_{BIAS} optimum setting can be calculated according to the following formula:

I_{BIAS} = VDD / (35 × L × Q × F)

Where:

VDD = Supply voltage in Volt

L = Inductance of transmit coil in Henry

F = Transmit oscillator frequency in Hz

Q = Quality factor of the Transmitter coil, it is calculated according to the following formula:

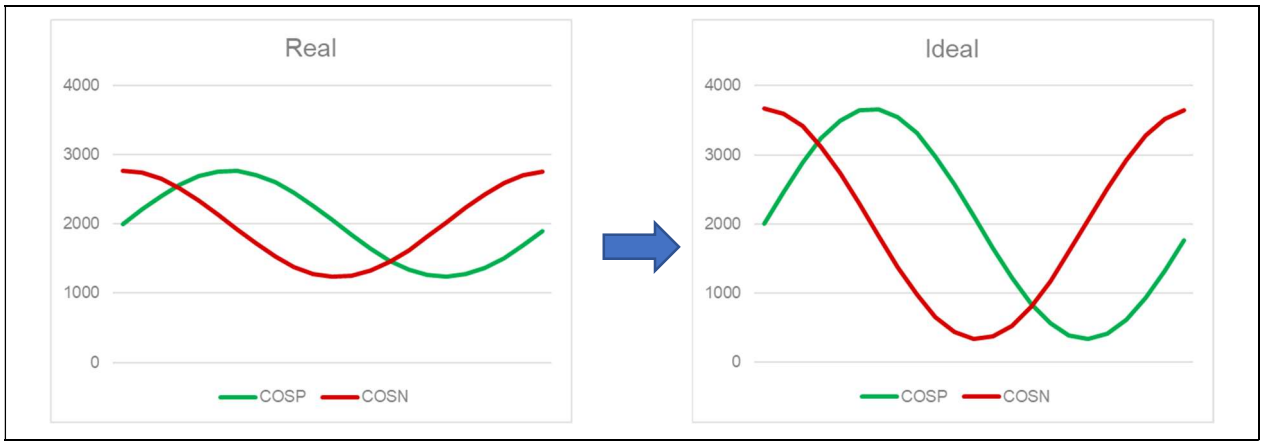
$$Q = \frac{1}{R}\sqrt{\frac{L}{C}}$$

Where:

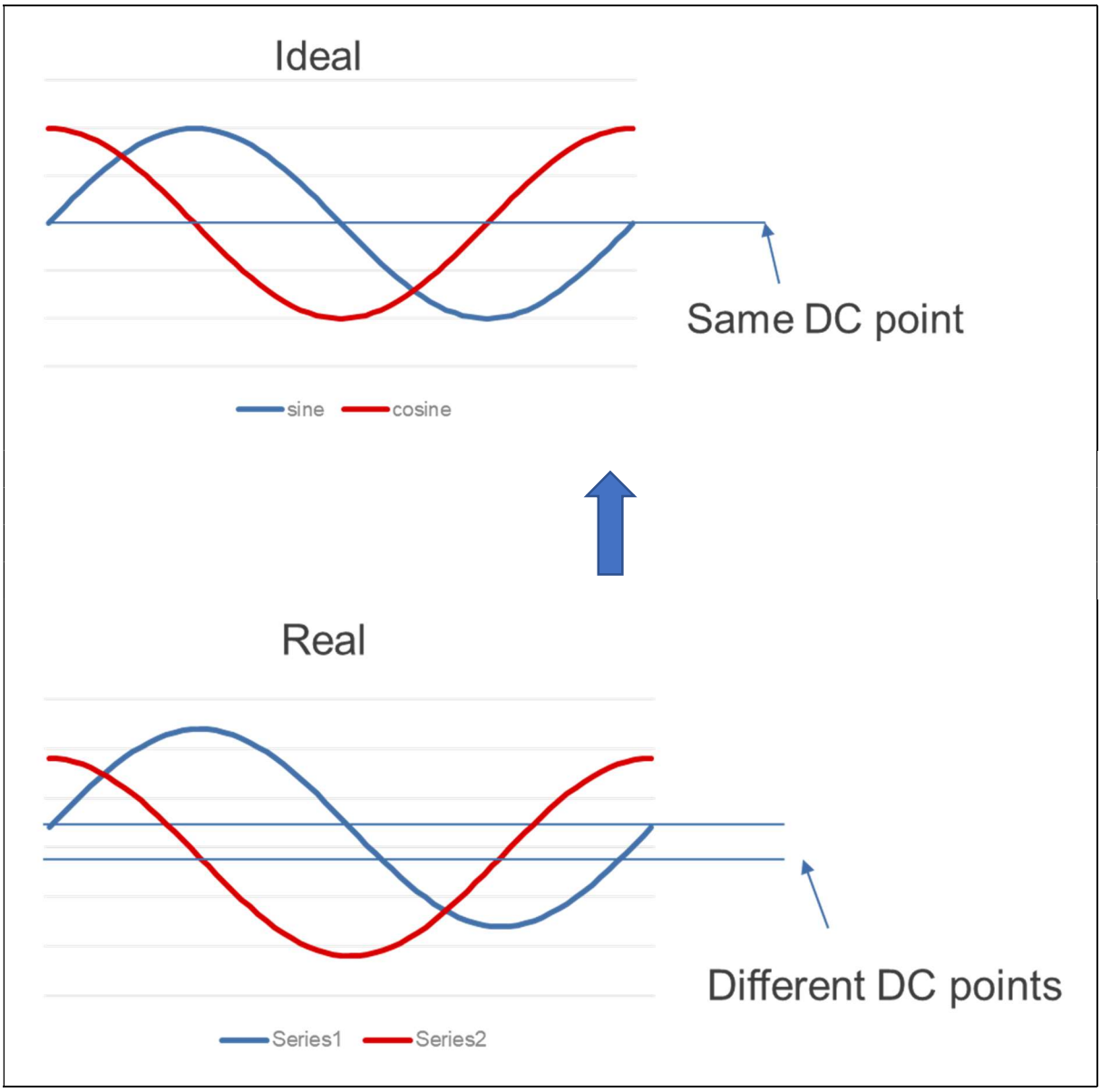
R = Resistance of the Transmitter Coil

C = Capacitance of the transmitter resonator

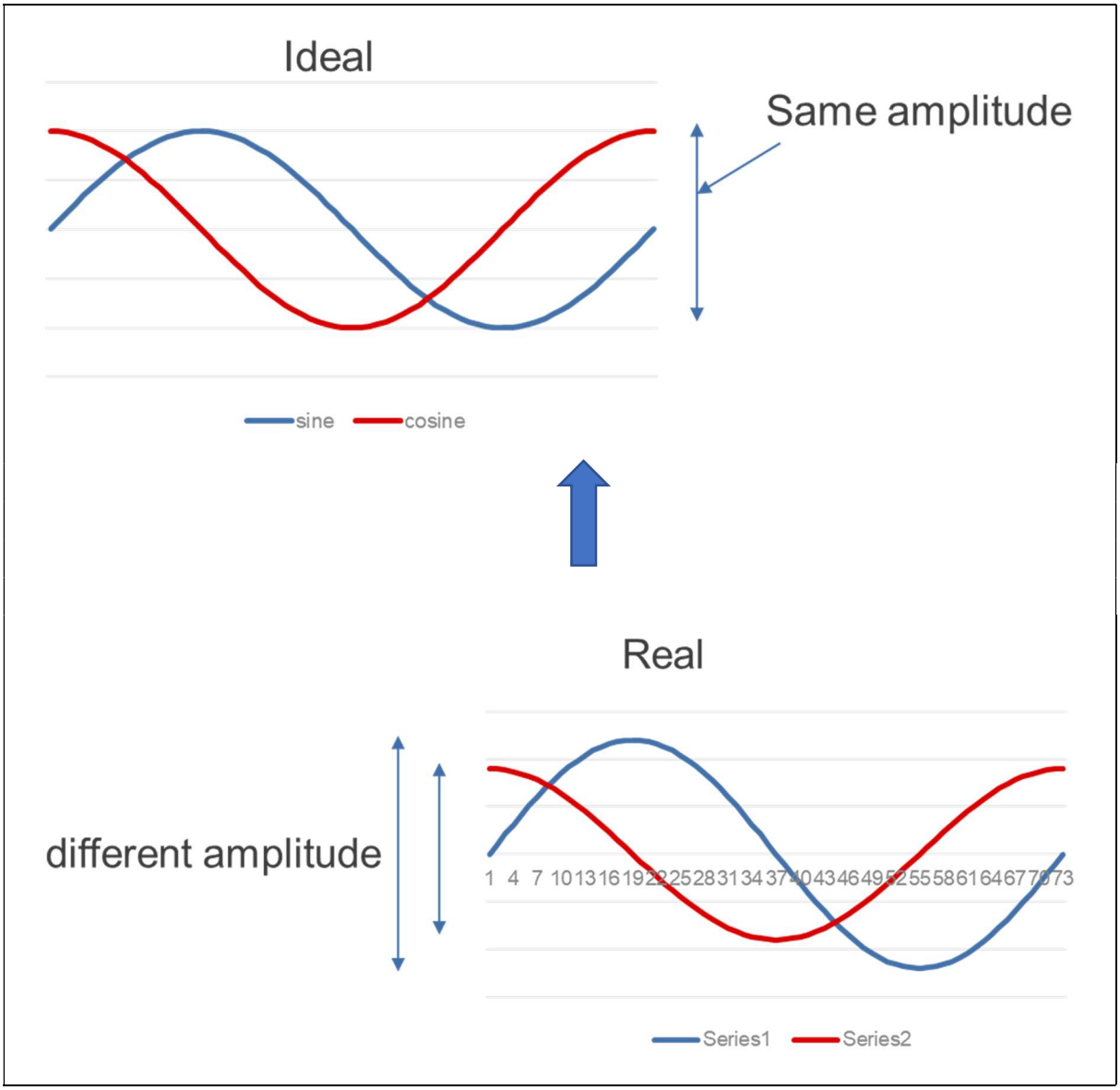
2. Integration Cycles setup – Adjust noise vs. speed.
   Please refer to “8. Sampling Rate, Resolution, Output Data Rate, and Propagation Delay” in the datasheet for IPS2200 (REN_IPS2200_DST_xxxxxxxx.pdf)
   Select the Integration Cycles based on actual application requirements.
3. Gain Stage Calibration – Calibrate the Master Gain register.
   For higher resolution, adjust the master gain register to put the output signals in an optimal signal range (ideal status) from real status as shown.



4. Offset Compensation – Calibrate the individual Coil Offset register
   Get the middle values of sine and cosine signals, adjust the R1 Coil Offset register according to the difference between the two middle values to result in the same DC point.



5. Mismatch Compensation – Calibrate the individual Fine Gain register. It equalizes the mismatch between the amplitude of the output signals.
   Get the max values of sine and cosine signals, increase or decrease the R1 Fine Gain register value according to the difference between the two max values to result in the same amplitude.



8.4.1 Execution of IPS2200 Calibration

Our sample code includes the IPS2200 calibration process. Set the macro definition “IPS2200_CALI” in “userdefine.h” file to “1”. Run this code after the metallic target is rotating at a constant speed. Calibration is complete until LED1 (While LED) flashes.

IPS2200 has NVM (nonvolatile memory), so the calibration process only needs to be performed once.

The spacing between the metallic target and the main board greatly affects stability and peak-to-peak values of signals. In the case of consistent assembly, if accuracy demand is not high, the same set of calibration parameter values can be used. Set macro definitions “IPS2200_CALI” to “0” and “IPS2200_SET” to “1” in “userdefine.h”. After calibration, the program will automatically continue to run the demo. Make sure the motor is stationary before the program runs.

If you don’t want to execute the IPS2200 calibration, set the macro definitions, both of “IPS2200_CALI” and “IPS2200_SET” to “0” in “userdefine.h”.