

# Integration

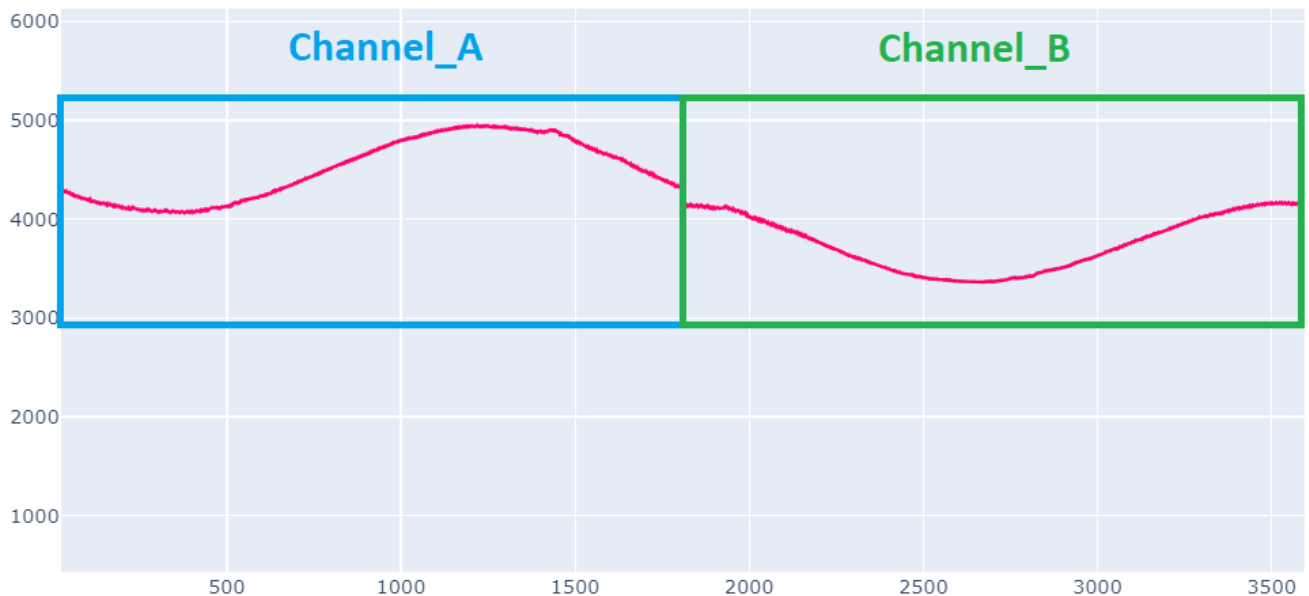
- [Analysis Data Waveform and Orbit Theory](#)
- [Modbus Register Map](#)

# Analysis Data Waveform and Orbit Theory

```
In [22]: print(dynamic_data)
```

executed in 20ms, finished 08:54:32 2022-10-25

42660,	6,	3600,	54064,	0,	65535	0,	9987,	4313,	4318,	4308,	4300,	4309,	4313,	4303,	4298,	4305,	4303,	4293,	4296,	4306,	4298,
4286,	4291,	4300,	4293,	4286,	4291,	4292,	4280,	4279,	4291,	4290,	4276,	4274,	4282,	4279,	4273,	4276,	4280,	4268,	4261,	4271,	
4276,	4266,	4258,	4265,	4265,	4255,	4256,	4265,	4259,	4247,	4250,	4258,	4253,	4245,	4250,	4252,	4241,	4237,	4247,	4248,	4237,	
4234,	4242,	4240,	4231,	4235,	4242,	4235,	4224,	4229,	4238,	4230,	4223,	4229,	4231,	4220,	4217,	4228,	4226,	4214,	4213,	4221,	
4218,	4209,	4213,	4219,	4210,	4203,	4210,	4215,	4205,	4201,	4209,	4209,	4198,	4199,	4208,	4205,	4194,	4197,	4204,	4198,	4192,	
4197,	4201,	4191,	4185,	4193,	4197,	4186,	4182,	4190,	4187,	4178,	4180,	4190,	4183,	4172,	4176,	4183,	4176,	4171,	4177,	4179,	
4168,	4165,	4174,	4176,	4166,	4164,	4171,	4167,	4159,	4164,	4172,	4164,	4154,	4160,	4166,	4158,	4154,	4161,	4160,	4149,	4149,	
4159,	4157,	4147,	4149,	4155,	4148,	4142,	4151,	4155,	4145,	4138,	4145,	4148,	4140,	4141,	4148,	4144,	4133,	4137,	4146,	4142,	
4133,	4137,	4143,	4134,	4129,	4138,	4141,	4131,	4127,	4135,	4136,	4126,	4127,	4134,	4130,	4120,	4124,	4133,	4127,	4118,	4124,	
4129,	4123,	4116,	4125,	4128,	4118,	4115,	4124,	4123,	4113,	4114,	4123,	4119,	4111,	4115,	4123,	4116,	4108,	4114,	4120,	4111,	
4106,	4114,	4117,	4107,	4106,	4115,	4114,	4104,	4106,	4115,	4111,	4102,	4107,	4114,	4108,	4101,	4109,	4114,	4106,	4101,	4110,	
4113,	4103,	4102,	4111,	4110,	4101,	4103,	4112,	4107,	4098,	4102,	4108,	4101,	4094,	4102,	4105,	4096,	4092,	4100,	4102,	4093,	
4093,	4102,	4098,	4089,	4093,	4102,	4096,	4089,	4094,	4099,	4091,	4087,	4097,	4098,	4088,	4086,	4095,	4096,	4087,	4089,	4096,	
4089,	4081,	4088,	4095,	4087,	4080,	4087,	4089,	4081,	4081,	4090,	4088,	4076,	4078,	4086,	4083,	4076,	4083,	4087,	4079,	4074,	
4083,	4087,	4078,	4077,	4084,	4081,	4073,	4077,	4086,	4080,	4073,	4078,	4082,	4074,	4072,	4081,	4081,	4072,	4072,	4082,	4079,	
4072,	4078,	4084,	4077,	4071,	4080,	4084,	4076,	4074,	4082,	4081,	4073,	4077,	4085,	4078,	4070,	4077,	4081,	4072,	4070,	4079,	
4077,	4067,	4070,	4078,	4073,	4067,	4074,	4079,	4070,	4067,	4076,	4078,	4069,	4071,	4079,	4074,	4067,	4073,	4078,	4071,	4067,	
4075,	4075,	4066,	4067,	4076,	4073,	4064,	4069,	4075,	4067,	4064,	4072,	4073,	4062,	4063,	4073,	4070,	4062,	4068,	4073,	4064,	
4061,	4070,	4072,	4064,	4065,	4073,	4068,	4061,	4069,	4075,	4067,	4064,	4072,	4073,	4064,	4068,	4076,	4071,	4065,	4071,	4075,	
4067,	4069,	4073,	4075,	4066,	4073,	4070,	4063,	4070,	4070,	4060,	4070,	4075,	4064,	4070,	4075,	4063,	4064,	4070,	4070,	4066,	



Header structure is the following:

- First register contains signature
- Second register contains type
- Third register contains data array size
- Fourth register contains data array CRC
- Fifth register contains groove array size
- Sixth register contains groove array CRC
- Seventh register is reserved
- Eight register contains CRC of the header

In your example:

4660 = 0x1234 is the correct signature

6 is type

3600 is the size of the data array

54064 is CRC of the 3600-sample data array

0 is the size of the groove indexes array (this indicates that you did not have any notches)

65535 is CRC of the groove array( which you do not need to calculate in your case because you did not have any notches)

0 is just a reserved member of header, we do not use it for now. We allocated it for later in case if we ever need to place something else in header.

9987 is crc of the header.

## Orbit Plots: [DataClipSamples.xlsx](#)

Above is a computer-generated signal in excel.

I used it to check whether our ideas with notch functionality would work.

This describes the process of getting from two sinewaves to an orbit signal.

I replaced 1800 Ch A data clip samples with 1800 samples of perfect sin wave.

Likewise, 1800 Ch B data clip samples with 1800 samples of perfect cos wave.

The computer-generated signal created perfect looking orbit, so we could test our notch functionality.

As for combining two data charts into orbit, you simply set MODBUS CHANNEL NUMBER REGISTER 40033 to MODBUS\_CHANNELS\_AB command.

This will capture data from both channels.

Data array size will be 3600 samples.

First 1800 samples are ch A samples.

Next 1800 samples are ch B samples.

So, it is easy to separate them.

Then you just plot channel A against channel B and you will get your orbit.

# Modbus Register Map

## About these Registers:

DEVICE\_ID / REMOTE\_TERMINAL\_UNIT (RTU) / SLAVE\_ID:

Each sensor on a single multi-drop bus line must have a unique DEVICE\_ID / RTU / SLAVE\_ID:  
By Default the DEVICE\_ID / RTU / SLAVE\_ID is the **LAST 2 DIGITS OF THE SENSORS SERIAL NUMBER**

The serial number (and therefore, the RTU number) can be found on the side of the TwinProx on the white label.

INDEXING:

Note that **the listed registers** below are considered **0-Indexed (the first value starts at 0)**

**Some Modbus masters will need to shift all the values up by one** value if their master recognized the first Modbus value at 1 (known as 1-indexed).

SERIAL COMMUNICATION SETTINGS:

**Baudrate:** 115200

**Parity:** None

**Handshakes:** None

**Data Bits:** 8

**Stop Bits:** 1

FUNCTION CODES:

The function codes supported by TwinProx Sensor are:

03 - (0x03) READ MULTIPLE **HOLDING** REGISTERS

16 - (0x10) WRITE MULTIPLE **HOLDING** REGISTERS

--- If you want to read or write to just a single register, you can do this by setting the length/offset/number of registers to 1 ---

## Endianness:

The TwinProx sensor uses the **Big Endian** memory allocation paradigm.

In computing, **endianness** is the order or sequence of bytes of a word of digital data in computer memory. Endianness is primarily expressed as **big-endian (BE)** or **little-endian (LE)**. A big-endian system stores the most significant byte of a word at the smallest memory address and the least significant byte at the largest. A little-endian system, in contrast, stores the least-significant byte at the smallest address.

## Modbus Register Map

Register Address	Number of Registers	Register Contents Description	Range	Default Value	Scale	Unit	Data Type	Read / Write	Notes
40176	1	Channel A Gap / Distance	0 to 105	n/a	value/100	mils	16-bit Unsigned Integer	R	
40177	1	Channel B Gap / Distance	0 to 105	n/a	value/100	mils	16-bit Unsigned Integer	R	
40178	1	Channel A Displacement	0 to 105	n/a	value/100	mils Pk-Pk	16-bit Unsigned Integer	R	
40179	1	Channel B Displacement	0 to 105	n/a	value/100	mils Pk-Pk	16-bit Unsigned Integer	R	

40049	1	DDC_START_SAMPLE	n/a		sample index	16-bit Unsigned Integer	R/W	<div>This register will start by reporting a 0 when read (indicating the 0th sample is in register 50, ready to be read). After successfully reading the data clip sample in register 171, this register should read 122 (indicating the 122nd sample is in register 50, ready to be read).</div>
-------	---	------------------	-----	--	--------------	-------------------------	-----	---

40050-40171	122	DDC_Samples	n/a	n/a			16-bit Unsigned Integer	R	Block reads of register 49 - 171 repeated until SAMP collec equal 1800.
40171	1	Auto_Reload	DDC_Chunk	n/a			16-bit Unsigned Integer	R	Each time register 171 is successfully read by a Modbus Master Register 49 is updated to reflect the index of the sample in register 50 and the next set of DDC Sample is loaded into register 50 - 171.



