



Wisco AI210 Protocol



I. คุณสมบัติพิเศษ (Features)	1
II. การต่อสาย (Wiring Diagram)	2
III. รหัสและย่านการวัดของช่องสัญญาณ Analog Input แต่ละชนิด	3
IV. การตั้งค่า Dip Switch	4
1. การติดต่อกับโมดูลโดยใช้ Wisco ASCII Protocol	5
2. รายละเอียดและตัวอย่างของคำสั่ง Wisco Protocol	6
2.1 คำสั่งที่ใช้อ่านค่า Analog Input (Integer)	6
2.2 คำสั่งที่ใช้อ่านค่า Analog Input (Floating Point)	6
2.3 คำสั่งที่ใช้อ่านค่า Analog Input (Integer - Expansion Module)	7
2.4 คำสั่งที่ใช้อ่านค่า Analog Input (Floating Point - Expansion Module)	7
2.5 คำสั่งที่ใช้อ่านค่า Digital Input	8
2.6 คำสั่งที่ใช้อ่านค่า Digital Output	8
2.7 คำสั่งที่ใช้อ่านค่า Input/Output ทั้งหมด (Analog Integer)	8
2.8 คำสั่งที่ใช้อ่านค่า Input/Output ทั้งหมด (Analog Floating Point)	8
2.9 คำสั่งที่ใช้อ่านค่า Input/Output ทั้งหมด (Analog Integer - Expansion Module)	9
2.10 คำสั่งที่ใช้อ่านค่า Input/Output ทั้งหมด (Analog Floating Point - Expansion Module)	9
2.11 คำสั่งที่ใช้เขียนค่า Digital Output	9
2.12 คำสั่งที่ใช้อ่านค่า Analog Input Type	10
2.13 คำสั่งที่ใช้อ่านค่า Analog Input Type (Expansion Module)	10

2.14	คำสั่งที่ใช้กำหนดค่า Analog Input Type	11
2.15	คำสั่งที่ใช้อ่านค่า R Shunt	11
2.16	คำสั่งที่ใช้อ่านค่า R Shunt (Expansion Module)	11
2.17	คำสั่งที่ใช้เขียนค่า R Shunt	12
2.18	คำสั่งที่ใช้อ่านค่าจากหน่วยความจำชนิด EEPROM	12
2.19	คำสั่งที่ใช้เขียนค่าไปที่หน่วยความจำชนิด EEPROM	12
3.	รหัสที่ตอบกลับมาเมื่อเกิดข้อผิดพลาดในการส่งคำสั่งไปยังโมดูล AI210	13
4.	วิธีคิด CHECK SUM สำหรับ Wisco ASCII Protocol	14
5.	สรุปคำสั่งที่ใช้กับโมดูล AI210 (Wisco ASCII Protocol)	15
6.	การติดต่อกับโมดูลโดยใช้ MODBUS (ASCII) Protocol	16
7.	วิธีคิด CHECK SUM สำหรับ MODBUS (ASCII) Protocol	19
8.	การแปลงข้อมูล Analog ชนิด Sign Integer	20

Data Logger AI210



I. คุณสมบัติพิเศษ (Features)

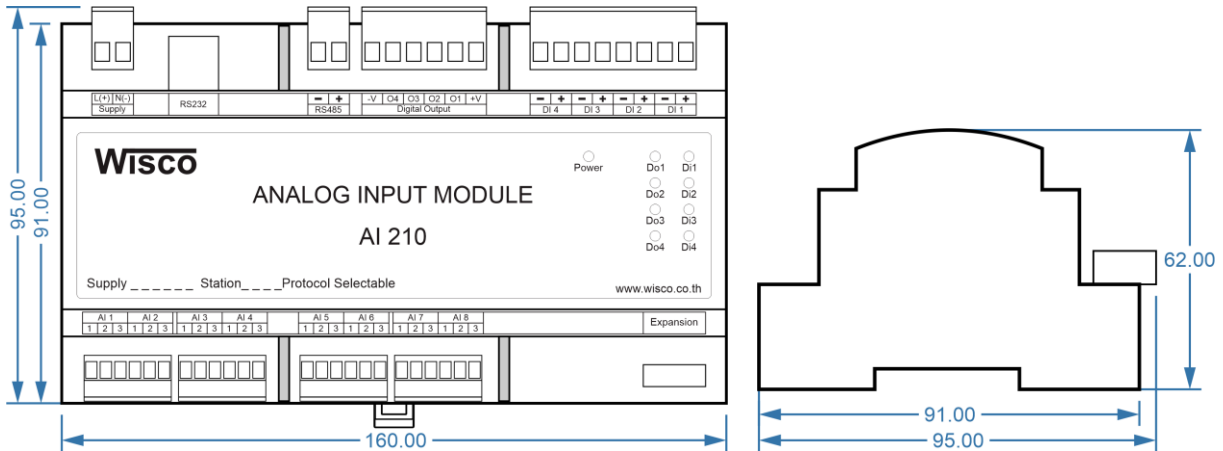
- ❖ 8 Channels Analog Input เลือกชนิดของสัญญาณได้ (Programmable Analog Input) 13 ชนิด ต่อ 1 channel
- ❖ แยกสัญญาณเข้า (Isolation) ของ Analog ด้วย Relay และแยกสัญญาณเข้าของ Digital ด้วย OPTO ELECTRONICS
- ❖ การสื่อสารตามมาตรฐาน RS-232 และ RS-485
- ❖ 2 ข้อกำหนดในการสื่อสาร (Communicate Protocol) ได้แก่ MODBUS RTU, MODBUS ASCII Protocol Compatible และ Wisco ASCII Protocol
- ❖ มีอัตราความเร็วในการรับ/ส่ง ข้อมูลเลือกได้ 4 แบบ คือ 4800, 9600, 19200, 57600

อัตราการใช้งานสูงสุด (Absolute Maximum Ratings)

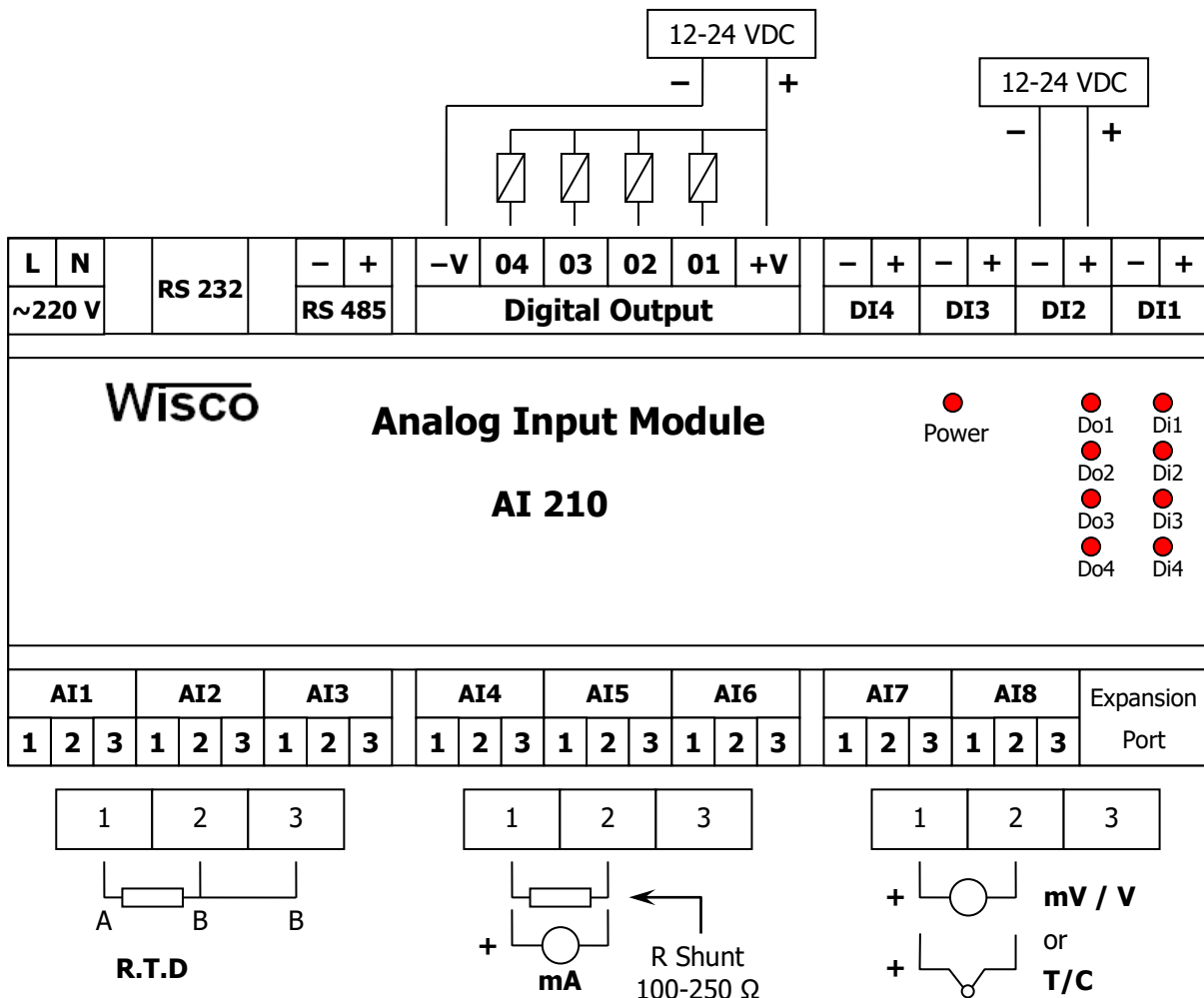
Analog Input	ขั้ว 1 ถึง 2	(-) 0.3 – 15 V
Digital Input	ขั้ว + ถึง -	(-) 0.3 – 30 V
Digital Output	ขั้ว +V ถึง -V	50 V, 500 mA
Operating Temperature		5-50 °C

II. การต่อสาย (Wiring Diagram)

Dimension (Unit: mm.)



Wiring



III. รหัสและย่านการวัดของช่องสัญญาณ Analog Input แต่ละชนิด

Code	Input Type	Measuring Range	Resolution	Accuracy (%FS) @25 °C	
00	Not Use	–	–	–	
01	Thermocouple	R	0 – 1700 °C	1 °C	± 0.2% (3.4 °C)
02		S	0 – 1700 °C	1 °C	± 0.2% (3.4 °C)
03		K	(-)250.0 – 1300.0 °C	0.1 °C	± 0.2% (2.6 °C)
04		E	0.0 – 1000.0 °C	0.1 °C	± 0.2% (2.0 °C)
05		J	(-)200.0 – 700.0 °C	0.1 °C	± 0.2% (1.4 °C)
06		T	(-)250.0 – 400.0 °C	0.1 °C	± 0.2% (0.8 °C)
07		B	0 – 1800 °C	1 °C	± 0.2% (3.6 °C)
08	R.T.D. Pt100	(-)200.0 – 800.0 °C	0.1 °C	± 0.2% (1.6 °C)	
09	Voltage(mV) 0 – 100	0.00 – 100.00 mV	0.01 mV	± 0.02% (0.02 mV)	
10	Voltage (V)	0 – 5	0.000 – 5.000 V	0.001 V	± 0.04% (0.002 V)
11		0 – 10	0.000 – 10.000 V	0.001 V	± 0.02% (0.002 V)
12	Current (mA)	0 – 20	0.00 – 20.00 mA	0.01 mA	± 0.1% (0.02 mA)
13		0 – 40	0.00 – 40.00 mA	0.01 mA	± 0.05% (0.02 mA)

IV. การตั้งค่า Dip Switch

เมื่อแกะฝาด้านบนของโมดูลออกจะพบ Dipswitch ที่ใช้สำหรับเลือก Station (ตำแหน่งที่ 1-5), Baud Rate (ตำแหน่งที่ 6-7) และ Protocol (ตำแหน่งที่ 8) มีรายละเอียดดังนี้

ตารางการตั้งค่า Dip Switch

1	2	3	4	5	Station
0	0	0	0	0	0 (00h)
1	0	0	0	0	1 (01h)
0	1	0	0	0	2 (02h)
1	1	0	0	0	3 (03h)
0	0	1	0	0	4 (04h)
1	0	1	0	0	5 (05h)
0	1	1	0	0	6 (06h)
1	1	1	0	0	7 (07h)
0	0	0	1	0	8 (08h)
1	0	0	1	0	9 (09h)
0	1	0	1	0	10 (0Ah)

1	2	3	4	5	Station
1	1	0	1	0	11 (0Bh)
0	0	1	1	0	12 (0Ch)
1	0	1	1	0	13 (0Dh)
0	1	1	1	0	14 (0Eh)
1	1	1	1	0	15 (0Fh)
0	0	0	0	1	16 (10h)
1	0	0	0	1	17 (11h)
0	1	0	0	1	18 (12h)
1	1	0	0	1	19 (13h)
0	0	1	0	1	20 (14h)
1	0	1	0	1	21 (15h)

1	2	3	4	5	Station
0	1	1	0	1	22 (16h)
1	1	1	0	1	23 (17h)
0	0	0	1	1	24 (18h)
1	0	0	1	1	25 (19h)
0	1	0	1	1	26 (1Ah)
1	1	0	1	1	27 (1Bh)
0	0	1	1	1	28 (1Ch)
1	0	1	1	1	29 (1Dh)
0	1	1	1	1	30 (1Eh)
1	1	1	1	1	31 (1Fh)

6	7	Baud rate
0	0	4800
1	0	9600
0	1	19200
1	1	57600

8	Protocol
0	MODBUS RTU
1	MODBUS ASCII / WISCO

การเชื่อมต่อ AI210 สามารถเชื่อมต่อได้สองมาตรฐานคือมาตรฐาน RS-232 และ RS-485 โดยมาตรฐาน RS-232 จะเป็นการเชื่อมต่อระหว่าง AI210 กับ PC หนึ่งต่อหนึ่งเท่านั้น ส่วนมาตรฐาน RS-485 จะสามารถเชื่อมต่อกันได้ครั้งละหลายเครื่องโดยสามารถเชื่อมต่อ AI210 ได้ทั้งหมด 32 เครื่อง พร้อมกันรวมกับเครื่องคอมพิวเตอร์อีก 1 เครื่อง โดยทั้งสองมาตรฐานจะใช้ข้อกำหนด Protocol เดียวกันในการติดต่อกับ AI210 มีรายละเอียดดังนี้

1. การติดต่อกับโมดูลโดยใช้ Wisco ASCII Protocol

ข้อมูลที่ใช้ในการติดต่อกับโมดูล AI210 จะเป็นรหัส ASCII ทั้งหมดและในคำสั่งชุดหนึ่งจะประกอบไปด้วย



ไบต์เริ่มต้น

ไบต์แรกที่บอกให้โมดูลรู้ว่าได้เริ่มต้นของชุดคำสั่ง โดยจะใช้อักขระ '#' เป็นตัวเริ่มต้น

หมายเลขประจำเครื่อง

หมายเลขประจำเครื่องที่ใช้อ้างอิงโมดูลสำหรับกรณีที่มีการต่อใช้งานพร้อมกันตั้งแต่ 2 เครื่องขึ้นไป โดยสามารถกำหนดได้ที่ DIP Switch ภายในโมดูล ซึ่งจะมีค่าตั้งแต่ 00h-1Fh และห้ามให้หมายเลขประจำเครื่องซ้ำกัน

คำสั่ง

คำสั่งที่ใช้กับโมดูล AI210 จะมีทั้งหมด 19 คำสั่ง

ไบต์จบ

ไบต์สุดท้ายที่บอกให้โมดูลรู้ว่าสิ้นสุดของชุดคำสั่ง โดยจะใช้ [CR] (Carriage Return) ซึ่งเป็นอักขระตัวที่ 13 ในตาราง ASCII ปิดท้าย

Character	#	0	0	R	A	I	1	2	4	5	8	CR
ASCII Code	23H	30H	30H	52H	41H	2AH	31H	32H	34H	35H	38H	0DH

ตัวอย่างการใช้งานคำสั่งสำหรับ Wisco ASCII Protocol

2. รายละเอียดและตัวอย่างของคำสั่ง **Wisco Protocol**

(= 1 byte, ... = n bytes, CR = Carriage Return)

2.1 คำสั่งที่ใช้อ่านค่า **Analog Input (Integer)**

เริ่มต้นด้วย 'RAI' ตามด้วยช่องสัญญาณที่จะอ่าน และจบด้วย '[CR]' เช่น อ่านค่า AI จาก
เครื่องหมายเลข 01 ช่องที่ 1, 2, 4, 5, 8 จะได้คำสั่งดังนี้ '#00RAI12458 [CR]'

#	0	1	R	A	I	1	2	4	5	8	CR
---	---	---	---	---	---	---	---	---	---	---	----

โดยโมดูลจะตอบกลับมาเป็น 'AI>' ตามด้วยค่าที่วัดได้เป็นเลขฐาน 16 โดยแต่ละช่องจะถูกคั่น
ด้วย ',' และจบด้วย '[CR]' ตัวอย่างดังนี้ 'AI>0FD1, 05A3, ..., 072E [CR]'

A	I	>	0	F	D	1	,	0	5	A	3	,	...	,	0	7	2	E	CR
---	---	---	---	---	---	---	---	---	---	---	---	---	-----	---	---	---	---	---	----

กรณีที่ต้องการอ่านค่า AI ทั้ง 8 ช่อง ให้ใช้ 'RAI' แล้วจบด้วย '[CR]' ได้เลย

#	0	1	R	A	I	CR
---	---	---	---	---	---	----

ทั้งนี้ค่าที่ได้ต้องนำมาแปลงก่อนจึงจะได้ค่าที่ถูกต้อง โดยสามารถดูรายละเอียดได้ที่ "หัวข้อที่ 8

ตารางการแปลงข้อมูล Analog ชนิด Sign Integer"

2.2 คำสั่งที่ใช้อ่านค่า **Analog Input (Floating Point)**

เริ่มต้นด้วย 'RAIF' ตามด้วยช่องสัญญาณที่จะอ่าน และจบด้วย '[CR]' เช่น อ่านค่า AI จาก
เครื่องหมายเลข 01 ช่องที่ 1, 3, 5, 7 จะได้คำสั่งดังนี้ '#01RAIF1357 [CR]'

#	0	1	R	A	I	F	1	3	5	7	CR
---	---	---	---	---	---	---	---	---	---	---	----

โดยโมดูลจะตอบกลับมาเป็น 'AI>' ตามด้วยค่าที่วัดได้เป็นเลขทศนิยม โดยแต่ละช่องจะถูกคั่น
ด้วย ',' และจบด้วย '[CR]' ตัวอย่างดังนี้ 'AI>12.1, 470, ..., -0.5 [CR]'

A	I	>	1	2	.	1	,	4	7	0	,	...	,	-	0	.	5	CR
---	---	---	---	---	---	---	---	---	---	---	---	-----	---	---	---	---	---	----

กรณีที่ต้องการอ่านค่า AI ทั้ง 8 ช่อง ให้ใช้ 'RAIF' แล้วจบด้วย '[CR]' ได้เลย

#	0	1	R	A	I	F	CR
---	---	---	---	---	---	---	----

2.3 คำสั่งที่ใช้อ่านค่า Analog Input (Integer - Expansion Module)

ใช้สำหรับอ่านข้อมูลจากโมดูล AI210 ที่เชื่อมต่อกับโมดูลเสริม EX24 ทำให้สามารถอ่านค่าได้ตั้งแต่ช่องที่ 1-24 โดยขึ้นต้นคำสั่งด้วย 'RAIX' ตามด้วยช่องสัญญาณที่จะอ่าน โดยใช้รูปแบบของ bit ทั้งหมด 6 ไบต์ (MSB -> LSB, '0' = ไม่อ่าน, '1' = อ่าน) และจบด้วย '[CR]' เช่น อ่านค่า AI จากเครื่องหมายเลข 01 ช่องที่ 24, 22, 20, 17, 16, 15, 10, 7, 4, 3, 2, 1 จะได้คำสั่งดังนี้ '#02RAIXA9C24F [CR]'

#	0	1	R	A	I	X	A	9	C	2	4	F	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	---	------

โดยโมดูลจะตอบกลับมาเป็น 'AI>' ตามด้วยค่าที่วัดได้เป็นเลขฐาน 16 โดยแต่ละช่องจะถูกคั่นด้วย ',' และจบด้วย '[CR]' ตัวอย่างดังนี้ 'AI>0FD1,05A3,...,072E[CR]'

A	I	>	0	F	D	1	,	0	5	A	3	,	...	,	0	7	2	E	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	---	-----	---	---	---	---	---	------

กรณีที่ต้องการอ่านค่า AI ทั้ง 24 ช่อง ให้ใช้ 'RAIXFFFFFF' แล้วจบด้วย '[CR]' ได้เลย

#	0	1	R	A	I	X	F	F	F	F	F	F	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	---	------

2.4 คำสั่งที่ใช้อ่านค่า Analog Input (Floating Point - Expansion Module)

ใช้สำหรับอ่านข้อมูลจากโมดูล AI210 ที่เชื่อมต่อกับโมดูลเสริม EX24 ทำให้สามารถอ่านค่าได้ตั้งแต่ช่องที่ 1-24 โดยขึ้นต้นคำสั่งด้วย 'RAIFX' ตามด้วยช่องสัญญาณที่จะอ่าน โดยใช้รูปแบบของ bit ทั้งหมด 6 ไบต์ (MSB -> LSB, '0' = ไม่อ่าน, '1' = อ่าน) และจบด้วย '[CR]' เช่น อ่านค่า AI จากเครื่องหมายเลข 01 ช่องที่ 24, 23, 22, 18, 13, 10, 9, 5 จะได้คำสั่งดังนี้ '#01RAIFXE21310 [CR]'

#	0	1	R	A	I	F	X	E	2	1	3	1	0	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	---	---	------

โดยโมดูลจะตอบกลับมาเป็น 'AI>' ตามด้วยค่าที่วัดได้เป็นเลขทศนิยม โดยแต่ละช่องจะถูกคั่นด้วย ',' และจบด้วย '[CR]' ตัวอย่างดังนี้ 'AI>12.1, 470,...,-0.5[CR]'

A	I	>	1	2	.	1	,	4	7	0	,	...	,	-	0	.	5	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	-----	---	---	---	---	---	------

กรณีที่ต้องการอ่านค่า AI ทั้ง 24 ช่อง ให้ใช้ 'RAIFXFFFFFF' แล้วจบด้วย '[CR]' ได้เลย

#	0	1	R	A	I	F	X	F	F	F	F	F	F	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	---	---	------

2.5 คำสั่งที่ใช้อ่านค่า Digital Input

เริ่มต้นด้วย 'RDI' และจบด้วย '[CR]' ซึ่งจะทำการอ่านค่า DI ทั้ง 4 ช่อง ดังนี้

#	0	1	R	D	I	[CR]
---	---	---	---	---	---	------

โดยโมดูลจะตอบกลับมาเป็น 'DI>' ตามด้วยค่าที่วัดได้ ('0' = OFF, '1' = ON) ช่องละ

1 ไบต์ และจบด้วย '[CR]' ตัวอย่างดังนี้ 'DI>0010[CR]'

D	I	>	0	0	1	0	[CR]
---	---	---	---	---	---	---	------

2.6 คำสั่งที่ใช้อ่านค่า Digital Output

เริ่มต้นด้วย 'RDO' และจบด้วย '[CR]' ซึ่งจะทำการอ่านค่า DO ทั้ง 4 ช่อง ดังนี้

#	0	1	R	D	O	[CR]
---	---	---	---	---	---	------

โดยโมดูลจะตอบกลับมาเป็น 'DO>' ตามด้วยค่าที่วัดได้ ('0' = OFF, '1' = ON) ช่องละ

1 ไบต์ และจบด้วย '[CR]' ตัวอย่างดังนี้ 'DO>0101[CR]'

D	O	>	0	1	0	1	[CR]
---	---	---	---	---	---	---	------

2.7 คำสั่งที่ใช้อ่านค่า Input/Output ทั้งหมด (Analog Integer)

สามารถอ่านค่า AI, DI และ DO ทุกช่องพร้อมกัน โดยใช้คำสั่งที่เริ่มต้นด้วย 'RADIO' และจบด้วย '[CR]' ซึ่งจะได้คำสั่งดังนี้ '#01RADIO[CR]'

#	0	1	R	A	D	I	O	[CR]
---	---	---	---	---	---	---	---	------

โดยโมดูลจะตอบกลับมาเป็น 'AI>' ตามด้วยค่า AI เป็นเลขฐาน 16 ทั้ง 8 ช่อง โดยแต่ละช่องจะถูกคั่นด้วย ',' ตามด้วยค่า DI ทั้ง 4 ช่อง คั่นด้วย ',' ตามด้วยค่า DO ทั้ง 4 ช่อง และจบด้วย '[CR]' ตัวอย่างดังนี้ 'AI>0FD1, 05A3,..., 0110, 0011 [CR]'

A	I	>	0	F	D	1	,	...	,	0	1	1	0	,	0	0	1	1	[CR]
---	---	---	---	---	---	---	---	-----	---	---	---	---	---	---	---	---	---	---	------

2.8 คำสั่งที่ใช้อ่านค่า Input/Output ทั้งหมด (Analog Floating Point)

สามารถอ่านค่า AI, DI และ DO ทุกช่องพร้อมกัน โดยใช้คำสั่งที่เริ่มต้นด้วย 'RADIOF' และจบด้วย '[CR]' ซึ่งจะได้คำสั่งดังนี้ '#01RADIOF [CR]'

#	0	1	R	A	D	I	O	F	[CR]
---	---	---	---	---	---	---	---	---	------

โดยโมดูลจะตอบกลับมาเป็น 'AI>' ตามด้วย AI ค่าที่วัดได้เป็นเลขทศนิยม ทั้ง 8 ช่อง โดยแต่ละช่องจะถูกคั่นด้วย ',' ตามด้วยค่า DI ทั้ง 4 ช่อง คั่นด้วย ',' ตามด้วยค่า DO ทั้ง 4 ช่อง และจบด้วย '[CR]' ตัวอย่างดังนี้ 'AI>15.2,-9.83,...,0110,0011 [CR]'

A	I	>	1	5	.	2	,	...	,	0	1	1	0	,	0	0	1	1	[CR]
---	---	---	---	---	---	---	---	-----	---	---	---	---	---	---	---	---	---	---	------

2.9 คำสั่งที่ใช้อ่านค่า Input/Output ทั้งหมด (Analog Integer - Expansion Module)

ใช้สำหรับอ่านข้อมูลจากโมดูล AI210 ที่เชื่อมต่อกับโมดูลเสริม EX24 โดยขึ้นต้นคำสั่งด้วย 'RADIOX' และจบด้วย '[CR]' ซึ่งจะได้คำสั่งดังนี้ '#01RADIOX [CR]'

#	0	1	R	A	D	I	O	X	CR
---	---	---	---	---	---	---	---	---	----

โดยโมดูลจะตอบกลับมาเป็น 'AI>' ตามด้วยค่า AI เป็นเลขฐาน 16 ทั้ง 24 ช่อง โดยแต่ละช่องจะถูกคั่นด้วย ',' ตามด้วยค่า DI ทั้ง 4 ช่อง คั่นด้วย ',' ตามด้วยค่า DO ทั้ง 4 ช่อง และจบด้วย '[CR]' ตัวอย่างดังนี้ 'AI>15.2,-9.83,...,0110,0011 [CR]'

A	I	>	0	F	D	1	,	...	,	0	1	1	0	,	0	0	1	1	CR
---	---	---	---	---	---	---	---	-----	---	---	---	---	---	---	---	---	---	---	----

2.10 คำสั่งที่ใช้อ่านค่า Input/Output ทั้งหมด (Analog Floating Point - Expansion Module)

ใช้สำหรับอ่านข้อมูลจากโมดูล AI210 ที่เชื่อมต่อกับโมดูลเสริม EX24 โดยขึ้นต้นคำสั่งด้วย 'RADIOFX' และจบด้วย '[CR]' ซึ่งจะได้คำสั่งดังนี้ '#01RADIOFX[CR]'

#	0	1	R	A	D	I	O	F	X	CR
---	---	---	---	---	---	---	---	---	---	----

โดยโมดูลจะตอบกลับมาเป็น 'AI>' ตามด้วยค่า AI เป็นเลขทศนิยมทั้ง 24 ช่อง โดยแต่ละช่องจะถูกคั่นด้วย ',' ตามด้วยค่า DI ทั้ง 4 ช่อง คั่นด้วย ',' ตามด้วยค่า DO ทั้ง 4 ช่อง และจบด้วย '[CR]' ตัวอย่างดังนี้ 'AI>15.2,-9.83,...,0110,0011 [CR]'

A	I	>	1	5	.	2	,	...	,	0	1	1	0	,	0	0	1	1	CR
---	---	---	---	---	---	---	---	-----	---	---	---	---	---	---	---	---	---	---	----

2.11 คำสั่งที่ใช้เขียนค่า Digital Output

ขึ้นต้นด้วย 'WDO' ตามด้วยช่องสัญญาณที่จะเขียน คั่นด้วย ',' ตามด้วยค่าที่ต้องการจะเขียนของช่องที่ต้องการ ('0' = OFF, '1' = ON) และจบด้วย '[CR]' เช่น เขียนค่า DO ไปที่เครื่องหมายเลข 11 ช่องที่ 1=OFF, 2=ON, 4=OFF จะได้คำสั่งดังนี้ '#01WDO124,010[CR]'

#	0	1	W	D	O	1	2	4	,	0	1	0	CR
---	---	---	---	---	---	---	---	---	---	---	---	---	----

โดยโมดูลจะตอบกลับมาเป็น 'DO>OK' และจบด้วย '[CR]' ดังนี้

D	O	>	O	K	CR
---	---	---	---	---	----

2.12 คำสั่งที่ใช้อ่านค่า Analog Input Type

เริ่มต้นด้วย 'RTY' แล้วตามด้วยช่องที่จะอ่านชนิดของ AI ช่องละ 1 ไบต์ และจบด้วย '[CR]'
 เช่น อ่านค่าชนิดของ AI จากเครื่องหมายเลข 01 ช่องที่ 1, 4, 5, 7 จะได้คำสั่งดังนี้
 '#01RTY1457[CR]'

#	0	1	R	T	Y	1	4	5	7	CR
---	---	---	---	---	---	---	---	---	---	----

โดยโมดูลจะตอบกลับมาเป็น 'TYPE>' ตามด้วยค่าชนิดของ AI แต่ละช่องเป็นเลขฐาน 10 โดย
 แต่ละช่องจะถูกคั่นด้วย ',' และจบด้วย '[CR]' ดังตัวอย่างนี้ 'EE>1,1,3,12[CR]'

T	Y	P	E	>	1	,	1	,	3	,	1	2	CR
---	---	---	---	---	---	---	---	---	---	---	---	---	----

กรณีที่ต้องการอ่านชนิดของ AI ทั้ง 8 ช่อง ให้ใช้ 'RTY' แล้วจบด้วย '[CR]' ได้เลย

#	0	1	R	T	Y	CR
---	---	---	---	---	---	----

2.13 คำสั่งที่ใช้อ่านค่า Analog Input Type (Expansion Module)

ใช้สำหรับอ่านค่าชนิดของ Analog Input จากโมดูล AI210 ที่เชื่อมต่อกับโมดูลเสริม EX24
 ทำให้สามารถอ่านค่าได้ตั้งแต่ช่องที่ 1-24 โดยเริ่มต้นคำสั่งด้วย 'RTYX' ตามด้วยช่องที่จะอ่านชนิด
 ของ AI โดยใช้รูปแบบของ bit ทั้งหมด 6 ไบต์ (MSB -> LSB, '0' = ไม่อ่าน, '1' = อ่าน) และจบ
 ด้วย '[CR]' เช่น อ่านค่าชนิดของ AI จากเครื่องหมายเลข 01 ช่องที่ 23, 19, 17, 11, 7, 5, 3,
 2, 1 จะได้คำสั่งดังนี้ '#01RTYX450457 [CR]'

#	0	1	R	T	Y	X	4	5	0	4	5	7	CR
---	---	---	---	---	---	---	---	---	---	---	---	---	----

โดยโมดูลจะตอบกลับมาในรูปแบบเดียวกับหัวข้อที่ 2.12

T	Y	P	E	>	1	1	,	1	2	,	...	,	6	CR
---	---	---	---	---	---	---	---	---	---	---	-----	---	---	----

ถ้าต้องการอ่านชนิดของ AI ทั้ง 24 ช่อง ให้ใช้ 'RTYXXXXXXX' แล้วจบด้วย '[CR]' ได้เลย

#	0	1	R	T	Y	X	F	F	F	F	F	F	CR
---	---	---	---	---	---	---	---	---	---	---	---	---	----

2.14 คำสั่งที่ใช้กำหนดค่า Analog Input Type

เริ่มต้นด้วย 'WTY' ตามด้วยชุดคำสั่งที่มี ช่องที่จะกำหนดชนิดของ AI ตามด้วย '=' ตามด้วยค่าที่ต้องการจะเขียนเป็นเลขฐาน 10 โดยแต่ละช่องจะคั่นด้วย ',' และจบด้วย '[CR]' เช่น กำหนดชนิดของ AI ให้กับเครื่องหมายเลข 01 ช่องที่ 1=1, 8=12, 21=9 จะได้คำสั่งดังตัวอย่างนี้ '#01WTY1=1, 8=12, 21=10[CR]'

#	0	1	W	T	Y	1	=	1	,	8	=	1	2	,	2	1	=	9	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	------

โดยโมดูลจะตอบกลับมาเป็น 'TYPE>OK' และจบด้วย '[CR]' ดังนี้

T	Y	P	E	>	O	K	[CR]
---	---	---	---	---	---	---	------

2.15 คำสั่งที่ใช้อ่านค่า R Shunt

เริ่มต้นด้วย 'RRI' แล้วตามด้วยช่องที่จะอ่านค่า R Shunt ช่องละ 1 ไบต์ และจบด้วย '[CR]' เช่น อ่านค่า R Shunt จากเครื่องหมายเลข 01 ช่องที่ 2, 6, 8 จะได้คำสั่งดังนี้ '#01RRI268 [CR]'

#	0	1	R	R	I	2	6	8	[CR]
---	---	---	---	---	---	---	---	---	------

โดยโมดูลจะตอบกลับมาเป็น 'RIN>' ตามด้วยค่า R Shunt เป็นเลขทศนิยม โดยแต่ละช่องจะถูกคั่นด้วย ',' และจบด้วย '[CR]' ตัวอย่างดังนี้ 'RIN>15.4, 205, 9.73 [CR]'

R	I	N	>	1	5	.	4	,	2	0	5	,	9	.	7	3	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	------

กรณีที่ต้องการอ่านค่า R Shunt ทั้ง 8 ช่อง ให้ใช้ 'RRI' แล้วจบด้วย '[CR]' ได้เลย

#	0	1	R	R	I	[CR]
---	---	---	---	---	---	------

2.16 คำสั่งที่ใช้อ่านค่า R Shunt (Expansion Module)

ใช้สำหรับอ่านค่า R Shunt จากโมดูล AI210 ที่เชื่อมต่อกับโมดูลเสริม EX24 ทำให้สามารถอ่านค่าได้ตั้งแต่ช่องที่ 1-24 ด้วยการขึ้นต้นคำสั่งด้วย 'RRIX' ตามด้วยช่อง R Shunt ที่จะอ่าน โดยใช้รูปแบบของบิตทั้งหมด 6 ไบต์ (MSB -> LSB, '0' = ไม่อ่าน, '1' = อ่าน) และจบด้วย '[CR]' เช่น อ่านค่า R Shunt จากเครื่องหมายเลข 01 ช่องที่ 23, 22, 17, 14, 10, 9, 8, 7, 6, 4, 3 จะได้คำสั่งดังนี้ '#01RRIX6123EC [CR]'

#	0	1	R	R	I	X	6	1	2	3	E	C	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	---	------

โดยโมดูลจะตอบกลับมาในรูปแบบเดียวกับหัวข้อที่ 2.15

R	I	N	>	3	9	.	6	,	3	.	5	,	...	,	4	.	4	8	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	---	-----	---	---	---	---	---	------

ถ้าต้องการอ่านค่า R Shunt ทั้ง 24 ช่อง ให้ใช้ 'RRIXFFFFFF' แล้วจบด้วย '[CR]' ได้เลย

#	0	1	R	R	I	X	F	F	F	F	F	F	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	---	------

2.17 คำสั่งที่ใช้เขียนค่า R Shunt

เริ่มต้นด้วย 'WRI' ตามด้วยช่อง R Shunt ที่จะเขียน คั่นด้วย '=' ตามด้วยค่าที่ต้องการจะเขียนเป็นเลขทศนิยม และจบด้วย '[CR]' เช่น เขียนค่า R Shunt ให้กับเครื่องหมายเลข 01 ช่องที่ 5 = 245.75 จะได้คำสั่งดังนี้ '#01WRI5=247.5[CR]'

#	0	1	W	R	I	5	=	2	4	7	.	5	CR
---	---	---	---	---	---	---	---	---	---	---	---	---	----

โดยโมดูลจะตอบกลับมาเป็น 'RIN(' ตามด้วยช่อง R Shunt ที่จะเขียน ตามด้วย ')>OK' และจบด้วย '[CR]' ตัวอย่างดังนี้ 'RIN(5)>OK[CR]'

R	I	N	(5)	>	O	K	CR
---	---	---	---	---	---	---	---	---	----

Note คำสั่งนี้ใช้เขียนค่าให้ R Shunt ได้ครั้งละ 1 ช่องเท่านั้น

2.18 คำสั่งที่ใช้อ่านค่าจากหน่วยความจำชนิด EEPROM

เริ่มต้นด้วย 'REE' ตามด้วยหมายเลขของ EEPROM ที่จะอ่าน 1 ไบต์ (คำสั่งจะนับตัว EEPROM โดยเริ่มนับจาก 0) ตามด้วยตำแหน่งเริ่มต้น 4 ไบต์ ตามด้วยจำนวนไบต์ที่จะอ่าน 4 ไบต์ และจบด้วย '[CR]' เช่น อ่านค่า EEPROM จากเครื่องหมายเลข 01 โดยเริ่มจากตำแหน่ง 200H จำนวน 500 Address (01F4H) จะได้คำสั่งดังนี้ '#01REE0020001F4 [CR]'

#	0	1	R	E	E	0	0	2	0	0	0	1	F	4	CR
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	----

โดยโมดูลจะตอบกลับมาเป็น 'EE>' ตามด้วยค่าที่อยู่ใน EEPROM เป็นเลขฐาน 16 ตามด้วยค่า Checksum อีก 2 ไบต์ (ดูวิธีคำนวณในหัวข้อ วิธีคิด Checksum สำหรับ Wisco Protocol) และจบด้วย '[CR]' ตัวอย่างดังนี้ 'EE>0320FF45...A79Dxx[CR]'

E	E	>	0	3	2	0	F	F	4	5	...	A	7	9	D	x	x	CR
---	---	---	---	---	---	---	---	---	---	---	-----	---	---	---	---	---	---	----

2.19 คำสั่งที่ใช้เขียนค่าไปที่หน่วยความจำชนิด EEPROM

เริ่มต้นด้วย 'WEE' ตามด้วยหมายเลขของ EEPROM ที่จะเขียน 1 ไบต์ (คำสั่งจะนับ EEPROM โดยเริ่มนับจาก 0) ตามด้วยตำแหน่งเริ่มต้น 4 ไบต์ ตามด้วยจำนวนไบต์ที่จะเขียน 2 ไบต์ ตามด้วยข้อมูลที่จะเขียน ตามด้วย Checksum (ดูวิธีคำนวณในหัวข้อ วิธีคิด Checksum สำหรับ Wisco Protocol) อีก 2 ไบต์ และจบด้วย '[CR]' เช่น เขียนค่า EEPROM ไปที่เครื่องหมายเลข 01 โดยเริ่มจากตำแหน่ง 100H จำนวน 2 ไบต์ (12 34) จะได้ คำสั่ง ดังตัวอย่างนี้ '#01WEE00100031234B7 [CR]' (B7 = checksum)

#	0	1	W	E	E	0	0	1	0	0	0	2	1	2	3	4	B	7	CR
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	----

โดยโมดูลจะตอบกลับมาเป็น 'EE>OK' และจบด้วย '[CR]' ดังนี้

E	E	>	O	K	CR
---	---	---	---	---	----

3. รหัสที่ตอบกลับมาเมื่อเกิดข้อผิดพลาดในการส่งคำสั่งไปยังโมดูล AI210

ในกรณีที่ส่งคำสั่งไปยังโมดูลนั้น หากชุดคำสั่งนั้นไม่ถูกต้อง โมดูลจะไม่ทำคำสั่งชุดนั้น และรายงานความผิดพลาดที่เกิดขึ้นกลับมาเป็นรหัสต่างๆ โดยจะขึ้นต้นด้วย 'ERR=' แล้วตามด้วยตัวเลขตั้งแต่ 1-6 มีรายละเอียดดังนี้

- | | |
|----------------------------|---|
| 1 (illegal function) | คำสั่งไม่ถูกต้อง หรือโมดูลไม่รู้จักคำสั่งนี้ |
| 2 (illegal data address) | ค่าตำแหน่งเริ่มต้น เกินช่วงตำแหน่งที่กำหนดไว้ |
| 3 (illegal data value) | ค่าของข้อมูลที่ใช้ในชุดคำสั่งไม่ถูกต้อง
เช่น ค่าของ DO ที่จะอ่าน ไม่ถูกต้อง |
| 4 (invalid data frame) | รูปแบบของชุดคำสั่งไม่ตรงตามข้อกำหนด
เช่น เขียนค่า DO โดยไม่มี ',' คั่นระหว่างหมายเลขช่องกับ
ค่าที่จะเขียน |
| 5 (check sum error) | ค่า check sum ไม่ถูกต้อง (อาจเกิดจากความผิดพลาด
ระหว่างส่งข้อมูล) |
| 6 (invalid number of byte) | จำนวนข้อมูลที่ได้รับมาไม่ครบตามจำนวนที่แจ้งไว้ |

4. วิธีคิด CHECK SUM สำหรับ Wisco ASCII Protocol

AI210 จะใช้ CHECK SUM ในการตรวจสอบความถูกต้องของข้อมูลที่ส่งไปสำหรับ Read หรือ Write กับ EEPROM การคิด CHECK SUM นั้นจะใช้การบวกข้อมูลทั้งหมดเข้าด้วยกัน (บวกเฉพาะข้อมูลที่เป็นตัวเลขเท่านั้น) บวกกันครั้งละ 1 ไบต์โดยค่าที่เกิน 1 byte นั้นจะตัดทิ้ง จากนั้น นำค่าที่ได้ 1 ไบต์ นั้นมาทำ 1's complement และ 2's complement

ตัวอย่างเช่น '# 1A WEE 0 0000 05 11 22 33 44 55 [CR]'

	HEXADECIMAL		BINARY
ไบต์เริ่มต้น	00H	}	0000 0000
	00H		0000 0000
	05H		0000 0000
	11H		0001 0001
	22H		0010 0010
	33H		0011 0011
	44H		0100 0100
ไบต์สุดท้าย	55H		0101 0101
ผลลัพธ์	104H		1 0000 0100
คิดเฉพาะ 1 byte (8 bit)	04H		0000 0100
ทำ 1's complement (invert)	FBH		1111 1011
ทำ 2' complement	FBH + 1		1111 1011+ 1
ค่า Check sum ที่ได้	FCH		1111 1100

ข้อมูลที่จะส่งจึงเป็น '# 1A WEE 0 0000 05 11 22 33 44 55 FC [CR]'

5. สรุปคำสั่งที่ใช้กับโมดูล AI210 (Wisco ASCII Protocol)

((H) = Heximal Value, (D) = Decimal Value, (E) = Extension Module,
xx = check sum, [CR] = carriage return)

Function	Command	AI210 Response
RAI = Read AI Value (H)	#00RAI12458[CR]	AI>0FD1,05A3,...,072E[CR]
RAIF = Read AI Value (D)	#01RAIF1357[CR]	AI>12.1,470,...,-0.5[CR]
RAIX = Read AI Value (E,H)	#02RAIXA9C24F[CR]	AI>0FD1,05A3,...,072E[CR]
RAIFX = Read AI Value (E,D)	#03RAIFXE21310[CR]	AI>12.1,470,...,-0.5[CR]
RDI = Read Digital Input	#04RDI234[CR]	DI>010[CR]
RDO = Read Digital Output	#05RDO[CR]	DO>1001[CR]
RADIO = Read All I/O	#07RADIO[CR]	AI>0FD1,...,0110,0011[CR]
RADIOF = Read All I/O (H)	#08RADIOF[CR]	AI>15.2,...,0110,0011[CR]
RADIOX = Read All I/O (E)	#09RADIOX[CR]	AI>0FD1,...,0110,0011[CR]
RADIOFX = Read All I/O (E,H)	#0ARADIOFX[CR]	AI>15.2,...,0110,0011[CR]
REE = Read EEPROM	#0BREE0020001F4[CR]	EE>0320FF45...A79Dxx[CR]
RRI = Read R Shunt	#0CRRI2368[CR]	RIN>15.4,205,9.73[CR]
RRIX = Read R Shunt (E)	#0DRRIX6123EC[CR]	RIN>39.6,3.5,...,4.48[CR]
RTY = Read AI Type	#0ERTY1457[CR]	TYPE>1,1,3,12[CR]
RTYX = Read AI Type (E)	#0FRTYX450457[CR]	TYPE>11,12,...,6[CR]
WDO = Write Digital Output	#11WDO13,11[CR]	DO>OK[CR]
WEE = Write EEPROM	#12WEE00100021234B7[CR]	EE>OK[CR]
WRI = Write R Shunt	#13WRI5=247.5[CR]	RIN(5)>OK[CR]
WTY = Write AI Type	#14WTY1=1,8=12,21=9[CR]	TYPE>OK[CR]

6. การติดต่อกับโมดูลโดยใช้ MODBUS (ASCII) Protocol

AI210 สามารถใช้ Protocol MODBUS ในการติดต่อได้เช่นกัน โดยจะมีรูปแบบของคำสั่งดังต่อไปนี้ (CHAR = Character; 1 CHAR ประกอบไปด้วย 8 Data Bits, 1 Start Bit, และ 1 Stop Bit)

ADDR	FUNCTION	DATA	ERROR CHECK
2-CHAR 16-BITS	2-CHAR 16-BITS	N x 4-CHAR N x 16-BITS	2-CHAR 16-BITS

AI210 สนับสนุนฟังก์ชันพื้นฐานของ Modbus ทั้งหมด 5 ฟังก์ชัน ดังต่อไปนี้

MODBUS ASCII

READ OUTPUT STATUS (CODE 01)
 READ INPUT STATUS (CODE 02)
 READ INPUT REGISTERS (CODE 04)
 FORCE SINGLE COIL (CODE 05)
 FORCE MULTIPLE COILS (CODE 15)

Wisco

= Read Digital Output
 = Read Digital Input
 = Read Analog Input
 = Write Digital Output
 = Write Digital Output

การอ้าง Address ของโมดูลมีดังนี้

Function Code	Reference	Address
01, 05, 15	Digital Output	0xxxx
02	Digital Input	1xxxx
04	Analog Input	3xxxx

Digital Output Table

Name	Address
Digital Output Channel 1	00001
Digital Output Channel 2	00002
Digital Output Channel 3	00003
Digital Output Channel 4	00004

Digital Input Table

Name	Address
Digital Input Channel 1	10001
Digital Input Channel 2	10002
Digital Input Channel 3	10003
Digital Input Channel 4	10004

Analog Input Table (Floating Point)

Name	Address
Analog Input Channel 1	30001 - 30002
Analog Input Channel 2	30003 - 30004
Analog Input Channel 3	30005 - 30006
Analog Input Channel 4	30007 - 30008
Analog Input Channel 5	30009 - 30010
Analog Input Channel 6	30011 - 30012
Analog Input Channel 7	30013 - 30014
Analog Input Channel 8	30015 - 30016
.	.
.	.
.	.
Analog Input Channel 24	30047 - 30048

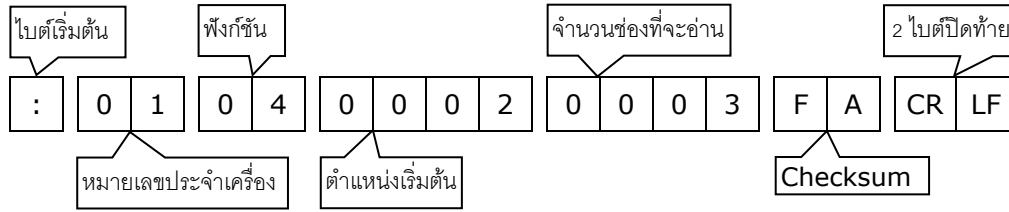
Analog Input Table (Integer)

Name	Address
Analog Input Channel 1	30101
Analog Input Channel 2	30102
Analog Input Channel 3	30103
Analog Input Channel 4	30104
Analog Input Channel 5	30105
Analog Input Channel 6	30106
Analog Input Channel 7	30107
Analog Input Channel 8	30108
.	.
.	.
Analog Input Channel 24	30124

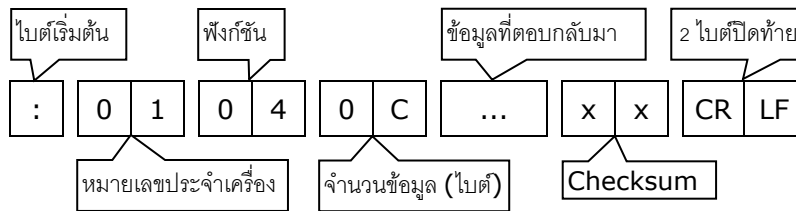
*** รายละเอียดที่เหลือของ Modbus สามารถดูได้จาก 'Modbus Reference Guide' หรือที่ <http://www.modbus.org/specs.php>

ตัวอย่างฟังก์ชัน MODBUS (ASCII) PROTOCOL

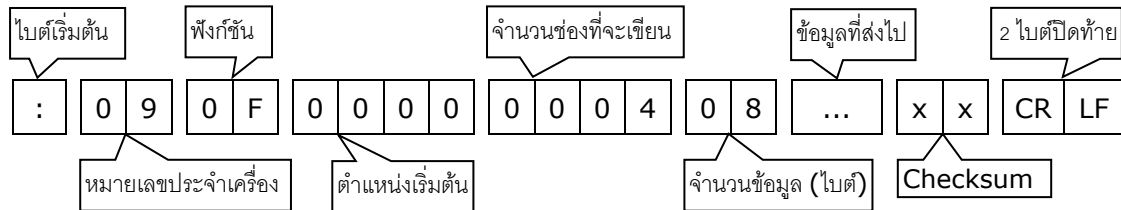
Function Code 04



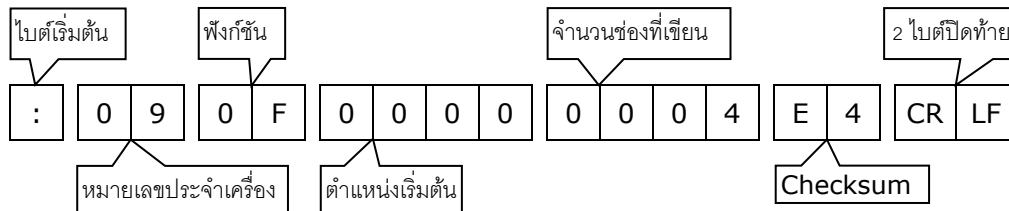
Response



Function Code 15



Response



7. วิธีคิด CHECK SUM สำหรับ MODBUS (ASCII) Protocol

MODBUS Protocol จะใช้ CHECK SUM ในการตรวจสอบความถูกต้องของข้อมูลที่ส่งไปทุกคำสั่ง การคิด CHECK SUM นั้นจะใช้การบวกข้อมูลทั้งหมดเข้าด้วยกัน (บวกเฉพาะข้อมูลที่เป็นตัวเลขเท่านั้น) บวกกันครั้งละ 1 ไบต์โดยค่าที่เกิน 1 byte นั้นเราจะตัดทิ้ง จากนั้น นำค่าที่ได้ 1 byte นั้นมาทำ 1's complement และ 2's complement

ตัวอย่างเช่น `: 0F 04 0001 0023 [CR] [LF]`

	HEXADECIMAL	BINARY
ไบต์เริ่มต้น	0FH	0000 1111
	04H	0000 0100
	00H	0000 0000
	01H	0000 0001
	00H	0000 0000
ไบต์สุดท้าย	23H	0010 0011
ผลลัพธ์	37H	0011 0111
คิดเฉพาะ 1 byte (8 bit)	37H	0011 0111
ทำ 1's complement (invert)	C8H	1100 1000
ทำ 2' complement	C8H + 1	1100 1000 + 1
ค่า Check sum ที่ได้	C9H	1100 1001

ข้อมูลที่จะส่งจึงเป็น `: 0F 04 0001 0023 C9 [CR] [LF]`

8. การแปลงข้อมูล Analog ชนิด Sign Integer

ข้อมูลชนิด sign integer ของโมดูล AI210 ได้จากการนำค่าวัดที่ได้ซึ่งเป็นค่าทศนิยม (IEEE741 Floating Point) มาทำการคูณเลื่อนจุดทศนิยม เพื่อให้ข้อมูลเป็นจำนวนเต็มสามารถเก็บในตัวแปร sign integer ขนาด 2 ไบต์ได้เพื่อลดขนาดข้อมูลลง ทำให้การส่งข้อมูลเร็วขึ้นและข้อมูลสามารถส่งผ่านข้อกำหนดของ MODBUS ได้ ดังนั้น ข้อมูลชนิด sign integer ที่ได้จึงต้องทำการเลื่อนจุดทศนิยมเข้าที่เดิมเสียก่อนโดยการหารด้วยค่าคงที่ตามตาราง ซึ่งค่าแต่ละชนิดจะไม่เท่ากันดังตารางข้างล่าง คอลัมน์สุดท้ายเป็นตัวหารสำหรับเลื่อนตำแหน่งทศนิยมเข้าที่เดิม

รหัส	ชนิดของสัญญาณ		ค่าวัด (Floating Point)	ค่าที่อ่านจากโมดูล	ตัวหารกลับ
00	Not Use		-	-	-
01	Thermocouple	R	0 - 1700 °C	0-1700	1
02		S	0 - 1700 °C	0-1700	1
03		K	(-)250.0 - 1300.0 °C	(-)2500-13000	10
04		E	0.0 - 1000.0 °C	0-10000	10
05		J	(-)200.0 - 700.0 °C	(-)2000-7000	10
06		T	(-)250.0 - 400.0 °C	(-)2500-4000	10
07		B	0 - 1800 °C	0-1800	1
08		R.T.D. Pt100		(-)200.0 - 800.0 °C	(-)2000-8000
09	Voltage(mV) 0 - 100		0.00 - 100.00 mV	0-10000	100
10	Voltage (V)	0 - 5	0.000 - 5.000 V	0-5000	1000
11		0 - 10	0.000 - 10.000 V	0-10000	1000
12	Current (mA)	0 - 20	0.00 - 20.00 mA	0-2000	100
13		0 - 40	0.00 - 40.00 mA	0-4000	100

Edit: 01/04/2022