

# Data Logger

## DL2100



### ข้อมูลทางเทคนิคสำหรับโมดูล DL2100 (Technical Data)

#### 1. คุณสมบัติพิเศษ (Features)

- 8 ช่อง Analog Input เลือกชนิดสัญญาณได้ (Programmable Analog Input) 13 ชนิด ต่อ 1 channel
- แยกสัญญาณเข้า (Isolation) Analog ด้วย Relay, Digital ด้วย OPTO ELECTRONICS
- สัญญาณการสื่อสาร ตามมาตรฐาน RS-232/RS-485 (มีตัว RS-232 to RS-485 Converter ภายใน)
- 2 ข้อกำหนดในการสื่อสาร (Communicate Protocol) ได้แก่ MODBUS (ASCII) Protocol Compatible และ Wisco Protocol
- มีอัตราเร็วในการรับ/ส่งข้อมูลให้เลือก 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
Power Consumption		ไม่เกิน 220 W (กินกระแสไม่เกิน 1 A)

**ข้อมูลเฉพาะ (Specifications)****Acquisition Time**                      สูงสุด 0.6 วินาที (600 ms)**Analog Input**

Resolution	คูตารางในหน้าที่ 5
Accuracy	คูตารางในหน้าที่ 5
Thermocouple	R, S, K, E, J, T, B
Cold Junction Temperature Compensation Accuracy	$\pm 2^{\circ}\text{C}$
RTD	Pt100 3-wire type
Amperage	$\approx 0.25\text{ mA}$
Resistance	สูงสุด 5 $\Omega$ ต่อ 1 สาย (5 $\Omega$ Max./Wire)
Voltage	0-10 V, 0-5 V, 0-100 mV
Input Impedance	$\approx 200\text{ k}\Omega$
Current	0-20 mA, 0-40 mA (ต้องต่อความต้านทานภายนอกด้วย)
External Resistor	$\approx 100\text{-}250\ \Omega$

**Digital Input**

Input Range	0-24 V (Low < 5 V, High > 9 V)
Input Impedance	$\approx 1\text{ k}\Omega$

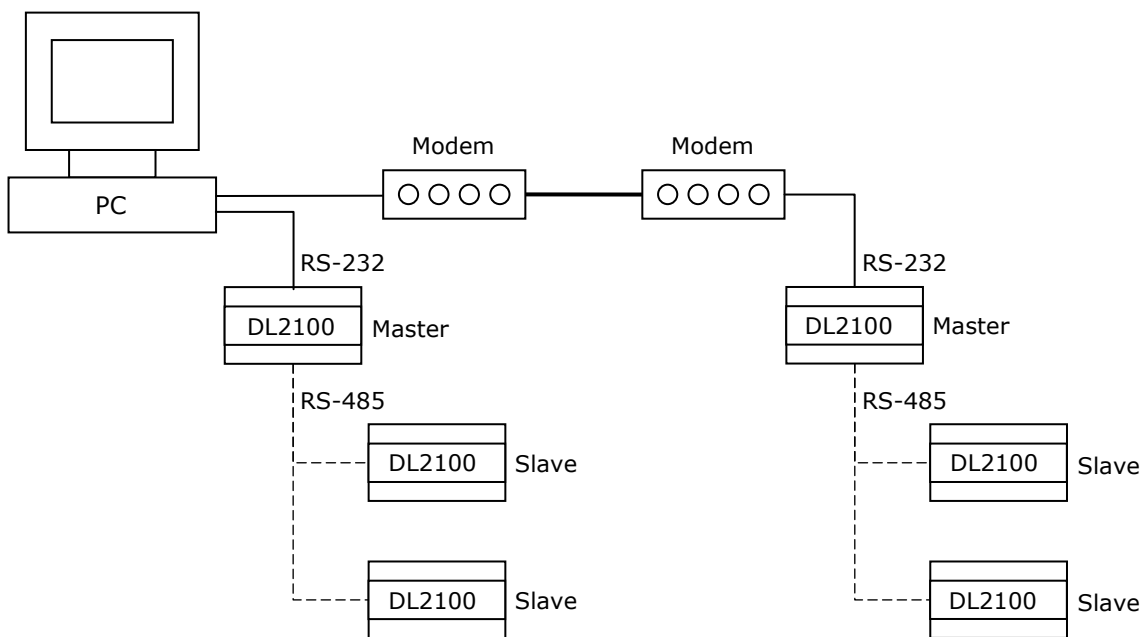
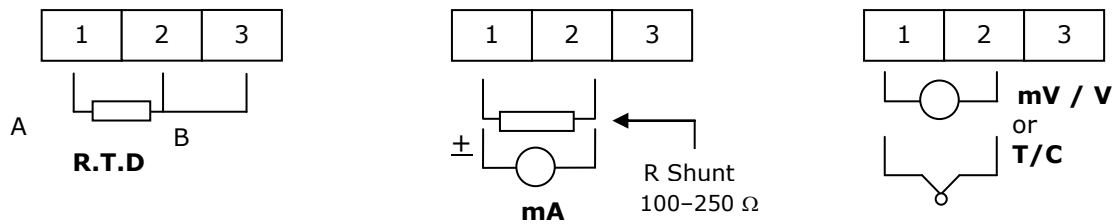
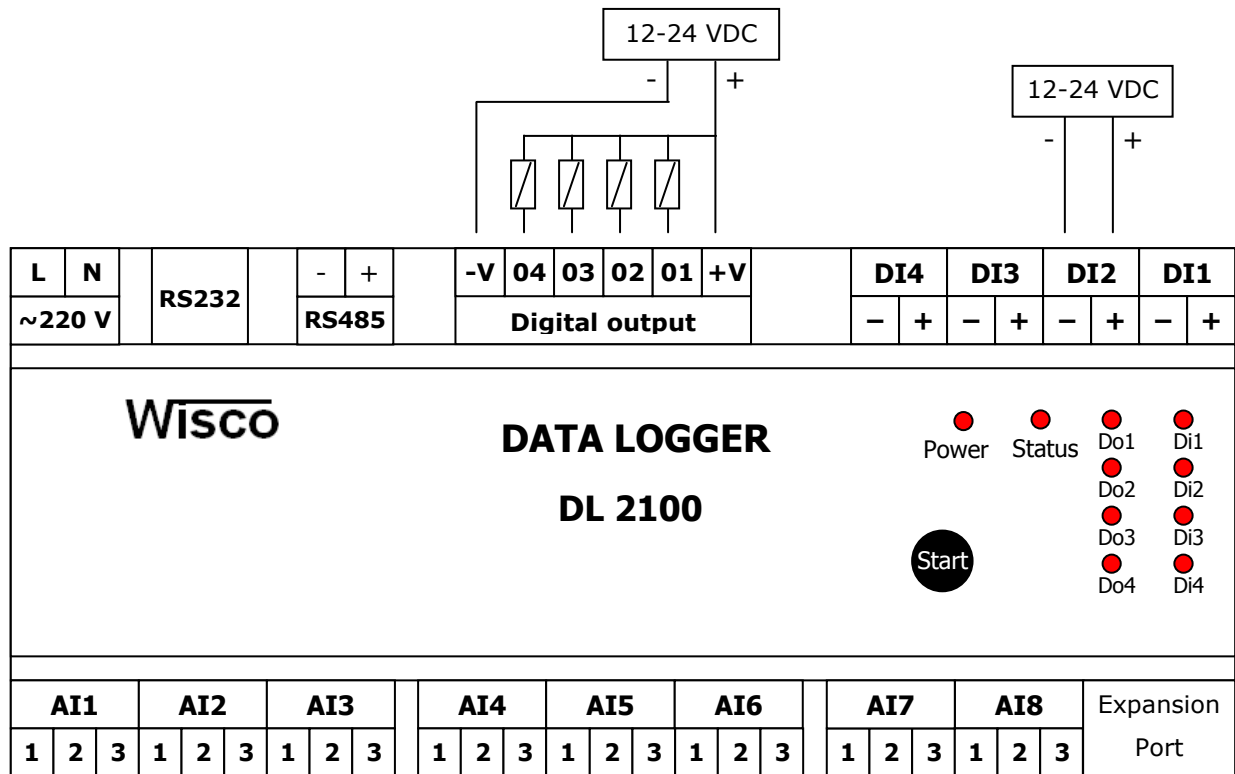
**Digital Output**

Output Type	Open Collector Transistor 50 V, 500 mA
-------------	--

**Communication**

Signal Level	EIA Standards, RS-232, and RS-485
Data Format	8 data bit, 1 stop bit, none parity
Station Number	0-31 (00H-1FH)
Baud Rate	4800, 9600, 19200, and 57600 bps
Distance	RS-232 max. 20 m, RS-485 max. 1 km
Number of Devices	Max. 32 stations

**การต่อสาย (Wiring Diagram)**



รหัสและย่านการวัดของช่องสัญญาณ **Analog** แต่ละชนิด

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

การเชื่อมต่อตัว **DL2100** สามารถเชื่อมต่อได้สองมาตรฐานคือมาตรฐาน RS-232 และ RS-485 โดยมาตรฐาน RS-232 จะเป็นการเชื่อมต่อระหว่าง **DL2100** กับ PC หนึ่งต่อหนึ่งเท่านั้น ส่วนมาตรฐาน RS-485 จะสามารถเชื่อมต่อกันได้ครั้งละหลายเครื่องโดยสามารถเชื่อมต่อ **DL2100** ได้ทั้งหมด 32 เครื่องพร้อมกันรวมกับ Computer อีก 1 เครื่อง โดยทั้งสองมาตรฐานจะใช้ข้อกำหนด (Protocol) เดียวกันในการติดต่อกับ **DL2100** โดยมีรายละเอียดดังต่อไปนี้

### การติดต่อกับโมดูลโดยใช้ **Wisco Protocol**

ข้อมูลที่ใช้ในการติดต่อกับโมดูล **DL2100** จะเป็นรหัส ASCII ทั้งหมดและในคำสั่งชุดหนึ่งจะประกอบไปด้วย



ไบต์เริ่มต้น

ไบต์แรกทีบอกให้โมดูลรู้ว่าได้เริ่มต้นของชุดคำสั่ง โดยจะใช้อักขระ '#' เป็นตัวเริ่มต้น

หมายเลขประจำเครื่อง

หมายเลขที่ใช้อ้างอิงตัวโมดูลสำหรับกรณีที่มีการต่อใช้งานพร้อมกันตั้งแต่ 2 ตัว ขึ้นไป โดยสามารถตั้งได้ที่ DIP Switch บนตัวโมดูล ซึ่งจะมีค่าตั้งแต่ 00-1F (เลขฐาน16) และห้ามให้หมายเลขซ้ำกัน

คำสั่ง

คำสั่งที่ใช้กับโมดูล สำหรับ **DL2100** จะมีทั้งหมด 19 คำสั่ง

ไบต์จบ

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

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 Protocol**

## รายละเอียดและตัวอย่างของคำสั่ง **Wisco Protocol**

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

### 1. คำสั่งที่ใช้อ่านค่า *Analog Input*

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

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

โดยตัวโมดูลจะตอบกลับมาเป็น 'AI>' ตามด้วยค่าที่วัดได้เป็นเลขฐาน 16 โดยแต่ละช่องจะถูกคั่นด้วย ',' และจบด้วย '[CR]' ดังตัวอย่างนี้ '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	0	R	A	I	CR
---	---	---	---	---	---	----

ทั้งนี้ค่าที่ได้ต้องนำมาแปลงก่อนจึงจะได้ค่าที่ถูกต้อง โดยสามารถดูรายละเอียดได้ที่ ตารางการแปลงข้อมูล *Analog* ชนิด *Sign Integer* ที่อยู่บนหน้าสุดท้าย

### 2. คำสั่งที่ใช้อ่านค่า *Analog Input (Floating Point)*

คล้ายกับข้อ 1 แต่เปลี่ยนเป็นเริ่มต้นด้วย 'RAIF' ตามด้วยช่องสัญญาณที่จะอ่าน และจบด้วย '[CR]' เช่น อ่านค่า AI จากเครื่องหมายเลข 01 ช่องที่ 1, 3, 5, 7 จะได้คำสั่งดังนี้ '#01RAIF1357 [CR]'

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

โดยตัวโมดูลจะตอบกลับมาเป็น 'AI>' ตามด้วยค่าที่วัดได้เป็นเลขทศนิยม ทำให้สามารถนำค่าที่ได้ไปใช้ต่อได้ทันที โดยช่องจะถูกรคั่นด้วย ',' และจบด้วย '[CR]' ดังตัวอย่างนี้ '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
---	---	---	---	---	---	---	----

### 3. คำสั่งที่ใช้อ่านค่า Analog Input (Expansion Module)

เหมือนกับข้อ 1 แต่ใช้อ่านข้อมูลจากโมดูล **DL2100** ที่ต่อกับโมดูลเสริม **EX24** ทำให้สามารถอ่านค่าได้ตั้งแต่ช่อง 1-24 ขึ้นต้นคำสั่งด้วย 'RAIX' ตามด้วยช่องสัญญาณที่จะอ่าน โดยใช้รูปแบบของ bit ทั้งหมด 6 ไบต์ (MSB -> LSB, '0' = 'ไม่อ่าน', '1' = 'อ่าน') และจบด้วย '[CR]' เช่น อ่านค่า AI จากเครื่องหมายเลข 02 ช่องที่ 24, 22, 20, 17, 16, 15, 10, 7, 4, 3, 2, 1 จะได้คำสั่งดังนี้ '#02RAIXA9C24F [CR]'

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

โดยตัวโมดูลจะตอบกลับมาเป็น 'AI>' ตามด้วยค่าที่วัดได้เป็นเลขฐาน 16 โดยแต่ละช่องจะถูกคั่นด้วย ',' และจบด้วย '[CR]' ดังตัวอย่างนี้ '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	2	R	A	I	X	F	F	F	F	F	F	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	---	------

### 4. คำสั่งที่ใช้อ่านค่า Analog Input (Floating Point - Expansion Module)

คล้ายกับข้อ 3 แต่เปลี่ยนเป็นขึ้นต้นด้วย 'RAIFX' ตามด้วยช่องสัญญาณที่จะอ่าน โดยใช้รูปแบบของ bit ทั้งหมด 6 ไบต์ (MSB -> LSB, '0' = 'ไม่อ่าน', '1' = 'อ่าน') และจบด้วย '[CR]' เช่น อ่านค่า AI จากเครื่องหมายเลข 03 ช่องที่ 24, 23, 22, 18, 13, 10, 9, 5 จะได้คำสั่งดังนี้ '#03RAIFXE21310 [CR]'

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

โดยตัวโมดูลจะตอบกลับมาเป็น 'AI>' ตามด้วยค่าที่วัดได้เป็นเลขทศนิยม ทำให้สามารถนำค่าที่ได้ไปใช้ต่อได้ทันที โดยแต่ละช่องจะถูกคั่นด้วย ',' และจบด้วย '[CR]' ดังตัวอย่างนี้ '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	3	R	A	I	F	X	F	F	F	F	F	F	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	---	---	------

### 5. คำสั่งที่ใช้อ่านค่า Digital Input

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

#	0	4	R	D	I	CR
---	---	---	---	---	---	----

โดยตัวโมดูลจะตอบกลับมาเป็น 'DI>' ตามด้วยค่าที่วัดได้ ('0' = OFF, '1' = ON) ช่องละ 1 ไบต์ และจบด้วย '[CR]' ดังตัวอย่างนี้ 'DI>0010[CR]'

D	I	>	0	0	1	0	CR
---	---	---	---	---	---	---	----

### 6. คำสั่งที่ใช้อ่านค่า Digital Output

คล้ายกับข้อ 5 แต่เปลี่ยนเป็นขึ้นต้นด้วย 'RDO' และจบด้วย '[CR]' ซึ่งจะทำการอ่านค่า DO ทั้ง 4 ช่อง ดังนี้

#	0	5	R	D	O	CR
---	---	---	---	---	---	----

โดยตัวโมดูลจะตอบกลับมาเป็น 'DO>' ตามด้วยค่าที่วัดได้ ('0' = OFF, '1' = ON) ช่องละ 1 ไบต์ และจบด้วย '[CR]' ดังตัวอย่างนี้ 'DO>0101[CR]'

D	O	>	0	1	0	1	CR
---	---	---	---	---	---	---	----

### 7. คำสั่งที่ใช้อ่านค่า Input/Output ทั้งหมด

เพื่อความสะดวกในการอ่านค่า AI, DI และ DO ทุกช่องพร้อมกัน ให้ใช้คำสั่งที่ขึ้นต้นด้วย 'RADIO' และจบด้วย '[CR]' ซึ่งจะได้คำสั่งดังนี้ '#07RADIO[CR]'

#	0	7	R	A	D	I	O	CR
---	---	---	---	---	---	---	---	----

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

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



### 8. คำสั่งที่ใช้อ่านค่า Input/Output ทั้งหมด (Analog Floating Point)

คล้ายกับข้อ 9 แต่จะใช้คำสั่งที่ขึ้นต้นด้วย 'RADIOF' และจบด้วย '[CR]' ซึ่งจะได้คำสั่งดังนี้ '#08RADIOF [CR]'

#	0	8	R	A	D	I	O	F	[CR]
---	---	---	---	---	---	---	---	---	------

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

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

### 9. คำสั่งที่ใช้อ่านค่า Input/Output ทั้งหมด (Analog Expansion Module)

เหมือนกับข้อ 9 แต่ใช้อ่านข้อมูลจากโมดูล DL2100 ที่ต่อกับโมดูลเสริม EX24 ขึ้นต้นคำสั่งด้วย 'RADIOX' และจบด้วย '[CR]' ซึ่งจะได้คำสั่งดังนี้ '#09RADIOX [CR]'

#	0	9	R	A	D	I	O	X	[CR]
---	---	---	---	---	---	---	---	---	------

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

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

### 10. คำสั่งที่ใช้อ่านค่า Input/Output ทั้งหมด (Analog Floating Point - Expansion Module).

คล้ายกับข้อ 3 แต่เปลี่ยนเป็นขึ้นต้นด้วย 'RADIOFX' และจบด้วย '[CR]' ซึ่งจะได้คำสั่งดังนี้ '#0ARADIOFX [CR]'

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

โดยตัวโมดูลจะตอบกลับมาเป็น 'AI>' ตามด้วยค่า AI เป็นเลขทศนิยม ทั้ง 24 ช่อง โดยแต่ละช่องจะถูกคั่นด้วย ',' ตามด้วยค่า DI ทั้ง 4 ช่อง คั่นด้วย ',' ตามด้วยค่า DO ทั้ง 4 ช่อง และจบด้วย '[CR]' ดังตัวอย่างนี้ 'AI>15.2,-9.83,..., 0110, 0011 [CR]'

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

### 11. คำสั่งที่ใช้อ่านค่า R Shunt

ขึ้นต้นด้วย 'RRI' แล้วตามด้วยช่องที่จะอ่านค่า R Shunt ช่องละ 1 ไบต์ และจบด้วย '[CR]' เช่น อ่านค่า R Shunt จากเครื่องหมายเลข 0C ช่องที่ 2, 6, 8 จะได้คำสั่งดังนี้ '#0CRRI268 [CR]'

#	0	C	R	R	I	2	3	8	[CR]
---	---	---	---	---	---	---	---	---	------

โดยตัวโมดูลจะตอบกลับมาเป็น 'RIN>' ตามด้วยค่า R Shunt เป็นเลขทศนิยม โดยแต่ละช่องจะถูกคั่นด้วย ',' และจบด้วย '[CR]' ดังตัวอย่างนี้ '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	C	R	R	I	[CR]
---	---	---	---	---	---	------

### 12. คำสั่งที่ใช้อ่านค่า R Shunt (Expansion Module)

เหมือนกับข้อ 13 แต่ใช้อ่านค่า R Shunt จากโมดูล **DL2100** ที่ต่อกับโมดูลเสริม **EX24** ทำให้สามารถอ่านค่าได้ตั้งแต่ช่อง 1-24 ขึ้นต้นคำสั่งด้วย 'RRIX' ตามด้วยช่อง R Shunt ที่จะอ่าน โดยใช้รูปแบบของ bit ทั้งหมด 6 ไบต์ (MSB -> LSB, '0' = 'ไม่อ่าน, '1' = 'อ่าน) และจบด้วย '[CR]' เช่น อ่านค่า R Shunt จากเครื่องหมายเลข 0D ช่องที่ 23, 22, 17, 14, 10, 9, 8, 7, 6, 4, 3 จะได้คำสั่งดังนี้ '#0DRRIX6123EC[CR]'

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

โดยตัวโมดูลจะตอบกลับมาในรูปแบบเดียวกับข้อ 13

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

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

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

### 13. คำสั่งที่ใช้อ่านค่าจากหน่วยความจำของ *Real Time Clock (RTC DS1307)*

ขึ้นต้นด้วย 'RRTC' ตามด้วยตำแหน่งเริ่มต้น 2 ไบต์ ตามด้วยจำนวนไบต์ที่จะอ่าน 2 ไบต์ และจบด้วย '[CR]' เช่น อ่านค่า RTC จากเครื่องหมายเลข 0E โดยเริ่มจากตำแหน่ง 08H จำนวน 22 Address (16H) จะได้คำสั่งดังนี้ '#0ERRTC0816 [CR]'

#	0	E	R	R	T	C	0	8	1	6	[CR]
---	---	---	---	---	---	---	---	---	---	---	------

โดยตัวโมดูลจะตอบกลับมาเป็น 'RTC>' ตามด้วยค่าที่อยู่ใน RTC เป็นเลขฐาน 16 ตามด้วยค่า Checksum อีก 2 ไบต์ (ดูวิธีคำนวณในหัวข้อ วิธีคิด Checksum สำหรับ Wisco Protocol) และจบด้วย '[CR]' ดังตัวอย่างนี้ 'RTC>0251D7A8...9630xx [CR]'

R	T	C	>	0	2	5	1	D	7	A	8	...	9	6	3	0	x	x	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	-----	---	---	---	---	---	---	------

### 14. คำสั่งที่ใช้อ่านค่า *Input Type (Analog Input)*

ขึ้นต้นด้วย 'RTY' แล้วตามด้วยช่องที่จะอ่านชนิดของ AI ช่องละ 1 ไบต์ และจบด้วย '[CR]' เช่น อ่านค่าชนิดของ AI จากเครื่องหมายเลข 0F ช่องที่ 1, 4, 5, 7 จะได้คำสั่งดังนี้ '#0FRTY1457 [CR]'

#	0	F	R	T	Y	1	4	5	7	[CR]
---	---	---	---	---	---	---	---	---	---	------

โดยตัวโมดูลจะตอบกลับมาเป็น 'TYPE>' ตามด้วยค่าชนิดของ AI แต่ละช่องเป็นเลขฐาน 10 โดยแต่ละช่องจะถูกคั่นด้วย ',' 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	F	R	T	Y	[CR]
---	---	---	---	---	---	------

### 15. คำสั่งที่ใช้อ่านค่า *Input Type (Analog Input – Expansion Module)*

เหมือนกับข้อ 15 แต่ใช้อ่านค่าชนิดของ AI จากโมดูล **DL2100** ที่ต่อกับโมดูลเสริม **EX24** ทำให้สามารถอ่านค่าได้ตั้งแต่ช่อง 1-24 ขึ้นต้นคำสั่งด้วย 'RTYX' ตามด้วยช่องที่จะอ่านชนิดของ AI โดยใช้รูปแบบของ bit ทั้งหมด 6 ไบต์ (MSB -> LSB, '0' = ไม่อ่าน, '1' = อ่าน) และจบด้วย '[CR]' เช่น อ่านค่าชนิดของ AI จากเครื่องหมายเลข 10 ช่องที่ 23, 19, 17, 11, 7, 5, 3, 2, 1 จะได้คำสั่งดังนี้ '#10RTYX450457[CR]'

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

โดยตัวโมดูลจะตอบกลับมาในรูปแบบเดียวกับข้อ 15

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

กรณีที่ต้องการอ่านชนิดของ AI ทั้ง 24 ช่อง ให้ใช้ 'RTYXFFFFFF' แล้วจบด้วย '[CR]' ได้เลย

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

### 16. คำสั่งที่ใช้เขียนค่า *Digital Output*

ขึ้นต้นด้วย 'WDO' ตามด้วยช่องสัญญาณที่จะเขียน คั่นด้วย ',' ตามด้วยค่าที่ต้องการจะเขียน ('0' = OFF, '1' = ON) และจบด้วย '[CR]' เช่น เขียนค่า DO ไปที่เครื่องหมายเลข 12 ช่องที่ 1=OFF, 2=ON, 4=OFF จะได้คำสั่งดังนี้ '#12WDO124, 010 [CR]'

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

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

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

### 17. คำสั่งที่ใช้เขียนค่า R Shunt

ขึ้นต้นด้วย 'WRI' ตามด้วยช่อง R Shunt ที่จะเขียน คั่นด้วย '=' ตามด้วยค่าที่ต้องการจะเขียนเป็นเลขทศนิยม และจบด้วย '[CR]' เช่น เขียนค่า R Shunt ให้กับเครื่องหมายเลข 14 ช่องที่ 5 = 245.75 จะได้คำสั่งดังนี้ '#14WRI5=247.5[CR]'

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

โดยตัวโมดูลจะตอบกลับมาเป็น 'RIN(' ตามด้วยช่อง R Shunt ที่เขียน ตามด้วย '>OK' และจบด้วย '[CR]' ดังตัวอย่างนี้ 'RIN(5)>OK[CR]'

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

ทั้งนี้ คำสั่งนี้ใช้เขียนค่าให้ R Shunt ได้ครั้งละ 1 ช่องเท่านั้น

### 18. คำสั่งที่ใช้เขียนค่าไปที่หน่วยความจำชนิดของ Real Time Clock

ขึ้นต้นด้วย 'WRTC' ตามด้วยตำแหน่งเริ่มต้น 2 ไบต์ ตามด้วยจำนวนไบต์ที่จะเขียน 2 ไบต์ ตามด้วยข้อมูลที่จะเขียน ตามด้วย Checksum (ดูวิธีคำนวณในหัวข้อ วิธีคิด Checksum สำหรับ Wisco Protocol) อีก 2 ไบต์ และจบด้วย '[CR]' เช่น เขียนค่า RTC ไปที่เครื่องหมายเลข 15 โดยเริ่มจากตำแหน่ง 10H จำนวน 2 ไบต์ (FE DC) จะได้คำสั่งดังตัวอย่างนี้ '#15WRTC0102FEDC14[CR]' (14 = checksum)

#	1	5	W	R	T	C	1	0	0	2	F	E	D	C	1	4	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	------

โดยตัวโมดูลจะตอบกลับมาเป็น 'RTC>OK' และจบด้วย '[CR]' ดังนี้

R	T	C	>	O	K	[CR]
---	---	---	---	---	---	------

### 19. คำสั่งที่ใช้กำหนดค่า Input Type (Analog Input)

ขึ้นต้นด้วย 'WTY' ตามด้วยชุดคำสั่งที่มี ช่องที่จะกำหนดชนิดของ AI ตามด้วย '=' ตามด้วยค่าที่ต้องการจะเขียนเป็นเลขฐาน 10 โดยแต่ละช่องจะคั่นด้วย ',' และจบด้วย '[CR]' เช่น กำหนดชนิดของ AI ให้กับเครื่องหมายเลข 16 ช่องที่ 1=1, 8=12, 21=9 จะได้คำสั่งดังตัวอย่างนี้ '#16WTY1=1, 8=12, 21=10 [CR]'

#	1	6	W	T	Y	1	=	1	,	8	=	1	2	,	2	1	=	9	[CR]
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	------

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

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

## รหัสตอบกลับมาเมื่อเกิดข้อผิดพลาดในการส่งคำสั่งไปยังตัวโมดูล DL2100

ในกรณีที่การส่งคำสั่งไปยังตัวโมดูลนั้น หากชุดคำสั่งนั้นไม่ถูกต้อง ตัวโมดูลจะไม่ทำคำสั่งชุดนั้น และรายงานความผิดพลาดที่เกิดขึ้นกลับมาเป็นรหัสต่างๆ โดยจะขึ้นต้นด้วย 'ERR=' แล้วตามด้วยตัวเลข ตั้งแต่ 1-6 ดังนี้

- |                            |   |
|----------------------------|---|
| 1 (illegal function)       | คำสั่งไม่ถูกต้อง หรือโมดูลไม่รู้จักคำสั่งนี้  |
| 2 (illegal data address)   | ค่าตำแหน่งเริ่มต้น เกินช่วงตำแหน่งที่กำหนดไว้   |
| 3 (illegal data value)     | ค่าของข้อมูลที่ใช้ในชุดคำสั่งไม่ถูกต้อง<br>เช่น ค่าของ DO ที่จะอ่าน ไม่ถูกต้อง                                  |
| 4 (invalid data frame)     | รูปแบบของชุดคำสั่งไม่ตรงตามข้อกำหนด<br>เช่น เขียนค่า DO โดยไม่มี ',' คั่นระหว่างหมายเลขช่อง<br>กับค่าที่จะเขียน |
| 5 (check sum error)        | ค่า check sum ไม่ถูกต้อง (อาจเกิดจากความ<br>ผิดพลาดระหว่างส่งข้อมูล)  |
| 6 (invalid number of byte) | จำนวนข้อมูลที่ได้รับมาไม่ครบตามจำนวนที่แจ้งไว้  |

## วิธีคิด CHECK SUM สำหรับ Wisco Protocol

ใน **DL2100** จะใช้ CHECK SUM ในการตรวจสอบความถูกต้องของข้อมูล ที่ส่งไปสำหรับ Read หรือ Write กับ EEPROM การคิด CHECK SUM นั้นจะใช้การบวกข้อมูลทั้งหมดเข้าด้วยกัน (บวกเฉพาะข้อมูลที่เป็นตัวเลขเท่านั้น) บวกกันครั้งละ 1 ไบต์โดยค่าที่เกิน 1 byte นั้นเราจะตัดทิ้งจากนั้น นำค่าที่ได้ 1 byte นั้นมาทำ 1's complement และ 2's complement เป็นอันเรียบร้อย

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	<u>FBH + 1</u>	<u>1111 1011 + 1</u>
ค่า Check sum ที่ได้	<u>FCH</u>	<u>1111 1100</u>

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

**สรุปคำสั่งที่ใช้กับตัวโมดูล DL2100 (Wisco Protocol)**

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

Function	Command	DL2100 Response
RAI = Read Analog Input (H)	#00RAI12458[CR]	AI>0FD1,05A3,...,072E[CR]
RAIF = Read Analog Input (D)	#01RAIF1357[CR]	AI>12.1,470,...,-0.5[CR]
RAIX = Read Analog Input (E,H)	#02RAIXA9C24F[CR]	AI>0FD1,05A3,...,072E[CR]
RAIFX = Read Analog Input (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]
RRTC = Read Real Time Clock	#0ERRTC0816[CR]	RTC>0251D7A8...9630xx[CR]
RTY = Read AI Type	#0FRTY1457[CR]	TYPE>1,1,3,12[CR]
RTYX = Read AI Type (E)	#10RTYX450457[CR]	TYPE>11,12,...,6[CR]
WDO = Write Digital Output	#12WDO13,11[CR]	DO>OK[CR]
WEE = Write EEPROM	#13WEE00100021234B7[CR]	EE>OK[CR]
WRI = Write R Shunt	#14WRI5=247.5[CR]	RIN(5)>OK[CR]
WRTC = Write Real Time Clock	#15WRTC0102FEDCB7[CR]	RTC>OK[CR]
WTY = Write AI Type	#16WTY1=1,8=12,21=9[CR]	TYPE>OK[CR]



## การติดต่อกับโมดูลโดยใช้ MODBUS (ASCII) Protocol

โมดูล **DL2100** สามารถใช้ Protocol MODBUS ในการติดต่อได้เช่นกัน โดยจะมีรูปแบบของคำสั่งดังต่อไปนี้ (CHAR = Character; 1 CHAR ประกอบไปด้วย 7 Data Bits, 1 Start Bit, 1 or 2 Stop Bits, และ 1 (optional) Parity Bit)

ADDR	FUNCTION	DATA	ERROR CHECK	EOF	READY TO REC RESP
2-CHAR 16-BITS	2-CHAR 16-BITS	N x 4-CHAR N x 16-BITS	2-CHAR 16-BITS	CR	LF

โมดูล **DL2100** สนับสนุนฟังก์ชันพื้นฐานของ Modbus ทั้งหมด 8 ฟังก์ชัน ดังต่อไปนี้

### **MODBUS ASCII**

READ OUTPUT STATUS (CODE 01)  
 READ INPUT STATUS (CODE 02)  
 READ OUTPUT REGISTERS (CODE 03)  
 READ INPUT REGISTERS (CODE 04)  
 FORCE SINGLE COIL (CODE 05)  
 PRESET SINGLE REGISTER (CODE 06)  
 FORCE MULTIPLE COILS (CODE 15)  
 PRESET MULTIPLE REGISTERS (CODE 16)

### **Wisco**

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

การอ้าง Address บนตัวโมดูลมีดังนี้

Function Code	Reference	Address
01, 05, 15	Digital Output	0xxxx
02	Digital Input	1xxxx
04	Analog Input	3xxxx
03, 06, 16	EEPROM	4xxxx

โดยค่า xxxx หมายถึงตัวเลขที่ระบุช่องสัญญาณนั้นๆ ดังต่อไปนี้

Module Channel	MODBUS Address (xxxx)
Digital Output 1-4	0000-0003
Digital Input 1-4	0000-0003
Analog Input 1-8	0000-0007
EEPROM	0000-03FF

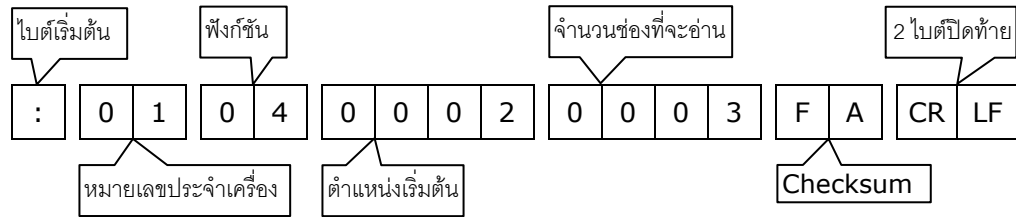
Address ของ EEPROM ถูกแบ่งใช้งานดังนี้

0000-0017	เก็บค่าชนิดของสัญญาณ (Analog Input) ช่องที่ 1-24 ตามลำดับ
0018-03FF	เก็บค่า Configuration ที่สำคัญเกี่ยวกับการทำงานของโมดูล <b>DL2100</b> เช่น ค่าสอบเทียบสัญญาณมาตรฐาน, ชื่อช่องสัญญาณวัดแต่ละช่อง ฯลฯ

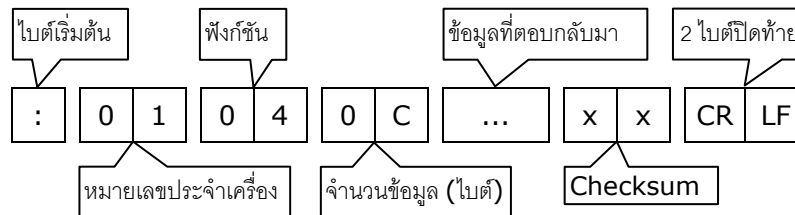
\* รายละเอียดที่เหลือของ Modbus สามารถดูได้จาก 'Modbus Reference Guide' หรือที่ <http://www.modbus.org/specs.php>

## ตัวอย่างฟังก์ชัน MODBUS (ASCII) PROTOCOL

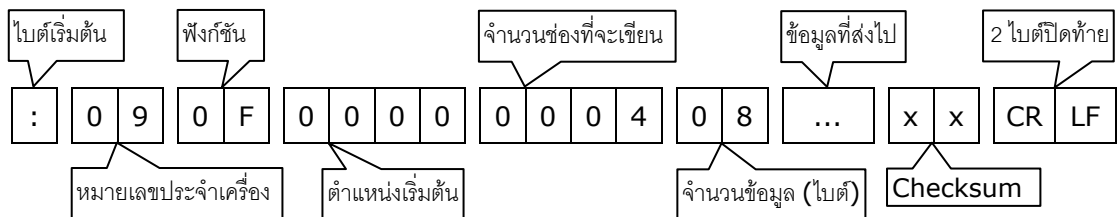
### Function Code 04



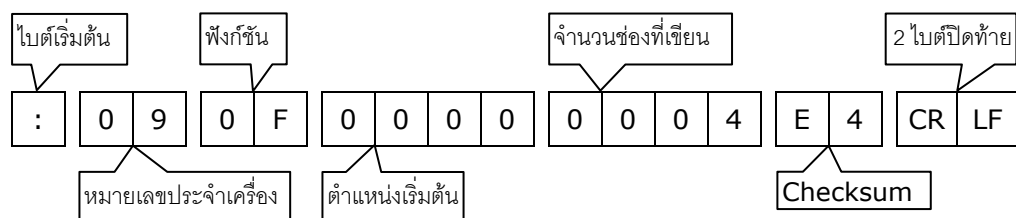
### Response



### Function Code 15



### Response



## วิธีคิด CHECK SUM สำหรับ MODBUS (ASCII) Protocol

ใน **DL2100** จะใช้ CHECK SUM ในการตรวจสอบความถูกต้องของข้อมูล ที่ส่งไปทุกคำสั่ง การคิด CHECK SUM นั้นจะใช้การบวกข้อมูลทั้งหมดเข้าด้วยกัน (บวกเฉพาะข้อมูลที่เป็นตัวเลขเท่านั้น) บวกกันครั้งละ 1 ไบต์โดยค่าที่เกิน 1 byte นั้นเราจะตัดทิ้ง จากนั้น นำค่าที่ได้ 1 byte นั้นมา ทำ 1's complement และ 2's complement เป็นอันเรียบร้อย

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	<u>C8H + 1</u>	<u>1100 1000 + 1</u>
ค่า Check sum ที่ได้	<u>C9H</u>	<u>1100 1001</u>

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

## การตั้งค่าให้กับ Dip Switch

เมื่อแกะฝาด้านบนของโมดูลออก จะพบ Dipswitch ที่ใช้เลือก Station (ตำแหน่งที่ 1-5) และ Baud rate (ตำแหน่งที่ 6-7) ตามต้องการ และ ควรเลือกให้เหมาะสมกับการใช้งาน ซึ่งมีข้อควรพิจารณาดังนี้

- ความยาว และ ความต้านทานของสาย
- การรบกวนจากภายนอก
- ถ้ามีการติดต่อผ่านโมเด็มไม่ควรตั้ง Baud rate สูงมากนัก ซึ่งจะขึ้นอยู่กับคุณภาพของคู่สายโทรศัพท์

ส่วนการกำหนด Protocol ที่ใช้ติดต่อกับโมดูล ให้เลือก Dipswitch ตำแหน่งที่ 8 ดังนี้  
 '0' = MODBUS RTU, '1' = MODBUS ASCII / WISCO PROTOCOL

**การเลือกค่า Baud rate และ Protocol**

1-5	6	7	8		Baud rate
			ASCII	RTU	
x	0	0	1	0	4800
x	1	0	1	0	9600
x	0	1	1	0	19200
x	1	1	1	0	57600

**การเลือก Station**

1	2	3	4	5	Station
0	0	0	0	0	0
1	0	0	0	0	1
0	1	0	0	0	2
1	1	0	0	0	3
0	0	1	0	0	4
1	0	1	0	0	5
0	1	1	0	0	6
1	1	1	0	0	7
0	0	0	1	0	8
1	0	0	1	0	9
0	1	0	1	0	10
1	1	0	1	0	11
0	0	1	1	0	12
1	0	1	1	0	13
0	1	1	1	0	14
1	1	1	1	0	15

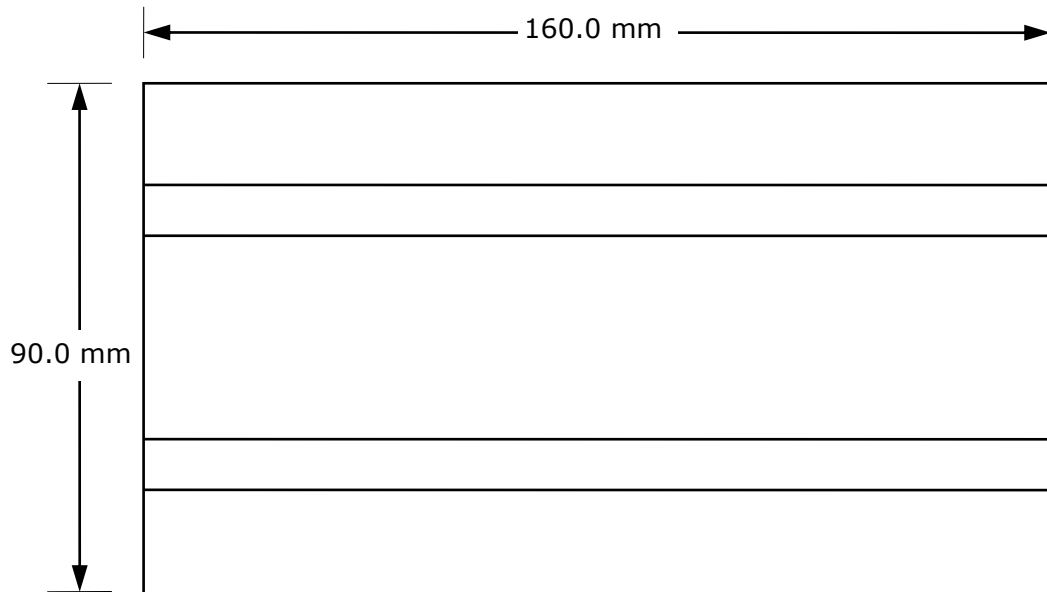
1	2	3	4	5	Station
0	0	0	0	1	16
1	0	0	0	1	17
0	1	0	0	1	18
1	1	0	0	1	19
0	0	1	0	1	20
1	0	1	0	1	21
0	1	1	0	1	22
1	1	1	0	1	23
0	0	0	1	1	24
1	0	0	1	1	25
0	1	0	1	1	26
1	1	0	1	1	27
0	0	1	1	1	28
1	0	1	1	1	29
0	1	1	1	1	30
1	1	1	1	1	31

## การแปลงข้อมูล Analog ชนิด Sign Integer

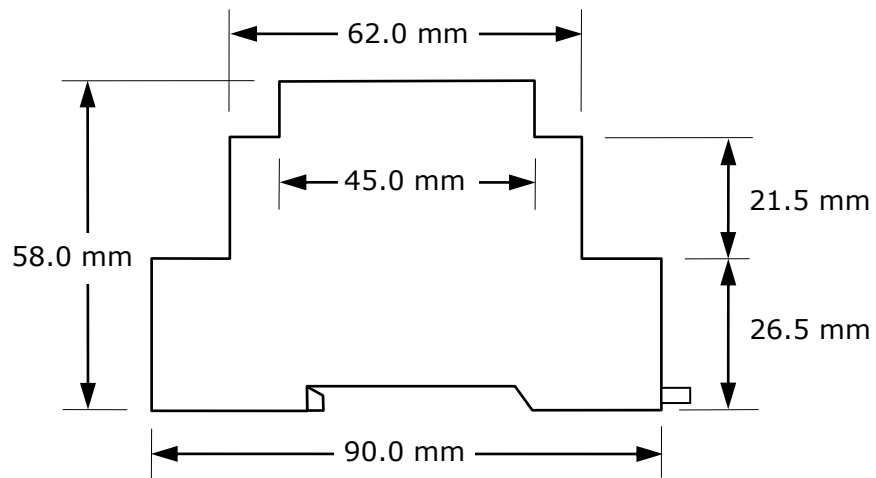
ข้อมูลชนิด sign integer ของโมดูล **DL2100** นั้นได้จากการนำค่าวัดที่ได้ซึ่งเป็นค่าทศนิยม (IEEE741 Floating Point) มาทำการคูณเลื่อนจุดทศนิยม เพื่อให้ข้อมูลเป็นจำนวนเต็มสามารถเก็บในตัวแปร sign integer ขนาด 2 ไบต์ได้เพื่อลดขนาดข้อมูลลง ทำให้การส่งข้อมูลเร็วขึ้นและข้อมูลสามารถส่งผ่านข้อกำหนดของ MODBUS ได้ ดังนั้น ข้อมูลชนิด sign integer ที่ได้จึงต้องทำการเลื่อนจุดทศนิยมเข้าที่เดิมเสียก่อนโดยการหารด้วยค่าคงที่ตามตาราง ซึ่งค่าแต่ละชนิดจะไม่เท่ากันดังตารางข้างล่าง คอลัมน์สุดท้ายเป็นตัวหารสำหรับเลื่อนตำแหน่งทศนิยมเข้าที่เดิม

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

**ขนาดกล่อง (External Dimensions)**



Top View



Side View