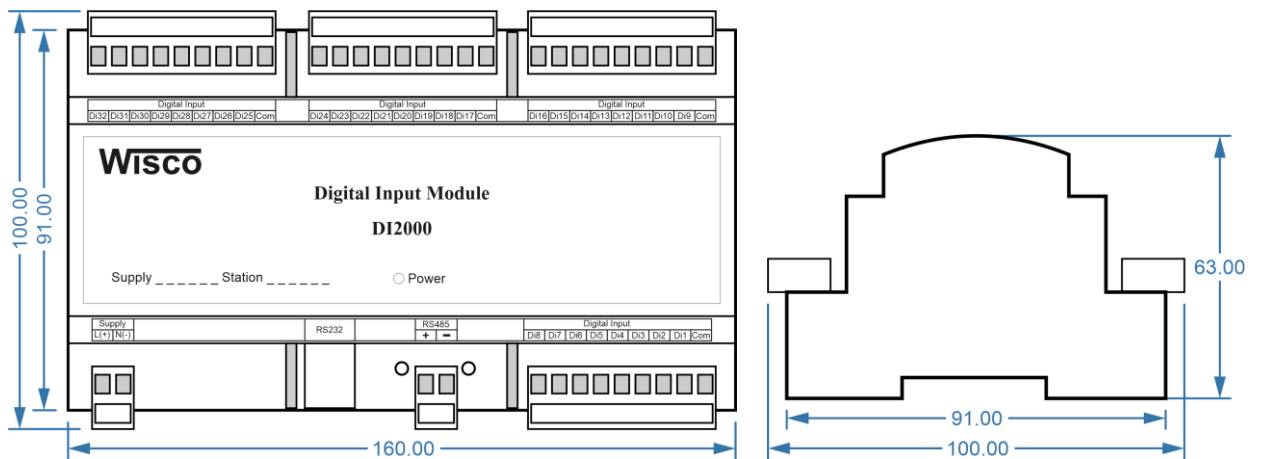


Digital Input Module

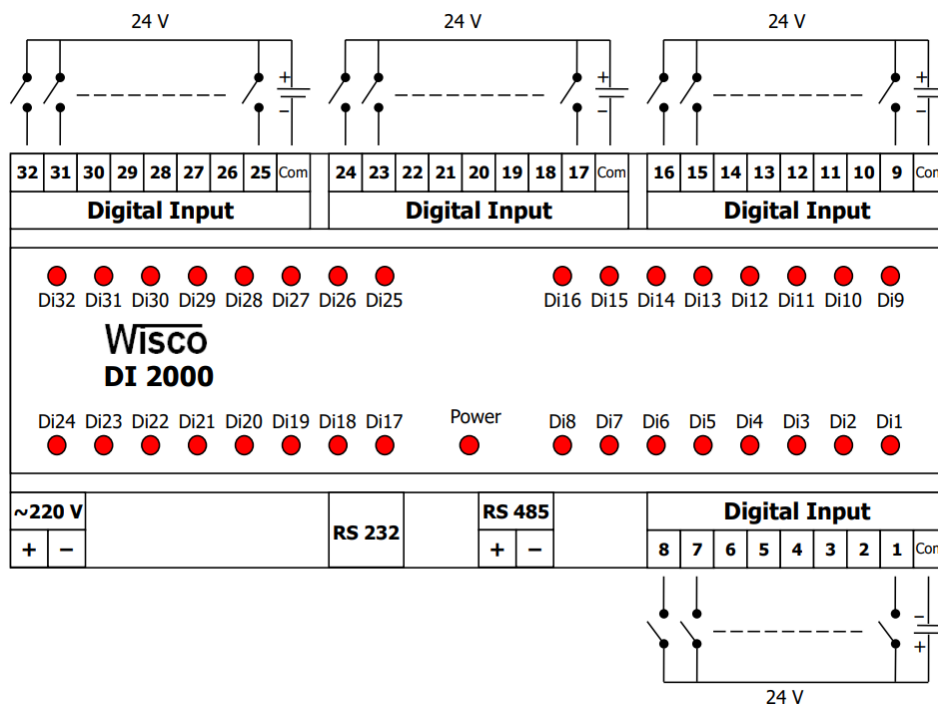
DI2000



Dimension (Unit: mm.)



Wiring



การเชื่อมต่อตัว **DI2000** สามารถเชื่อมต่อได้สองมาตรฐานคือมาตรฐาน RS-232 และ RS-485 โดยมาตรฐาน RS-232 จะเป็นการเชื่อมต่อระหว่าง **DI2000** กับ PC หนึ่งต่อหนึ่งเท่านั้น ส่วนมาตรฐาน RS-485 จะสามารถเชื่อมต่อกันได้ครั้งละหลายเครื่องโดยสามารถเชื่อมต่อ **DI2000** ได้ทั้งหมด 32 เครื่องพร้อมกันรวมกับ Computer อีก 1 เครื่อง โดยทั้งสองมาตรฐานจะใช้ข้อกำหนด (Protocol) เดียวกันในการติดต่อกับ **DI2000** โดยมีรายละเอียดดังต่อไปนี้

การติดต่อกับโมดูลโดยใช้ **Wisco Protocol**

ข้อมูลที่ใช้ในการติดต่อกับโมดูล **DI2000** จะเป็นรหัส ASCII ทั้งหมดและในคำสั่งชุดหนึ่งจะประกอบไปด้วย



ไบต์เริ่มต้น	ไบต์แรกที่บอกให้โมดูลรู้ว่าได้เริ่มต้นส่งคำสั่งแล้ว โดยจะใช้อักขระ '#' เป็นตัวเริ่มต้น
หมายเลขประจำเครื่อง	หมายเลขที่ใช้อ้างอิงตัวโมดูลสำหรับกรณีที่มีการต่อใช้งานพร้อมกันตั้งแต่ 2 ตัว ขึ้นไป โดยสามารถตั้งได้ที่ DIP Switch บนตัวโมดูล ซึ่งจะมีค่าตั้งแต่ 00h-1Fh และห้ามให้หมายเลขซ้ำกัน
คำสั่ง	คำสั่งที่ใช้กับโมดูล สำหรับ DI2000 จะมีทั้งหมด 2 คำสั่ง
ไบต์จบ	ไบต์สุดท้ายที่บอกให้โมดูลรู้ว่าสิ้นสุดการส่งคำสั่งแล้ว โดยจะใช้ [CR] (Carriage Return) ซึ่งเป็นอักขระตัวที่ 13 ในตาราง ASCII เป็นตัวปิดท้าย

Character	#	0	6	R	D	I	CR
ASCII Code	23H	30H	36H	52H	44H	49H	0DH

ตัวอย่างการใช้งานคำสั่งสำหรับ Wisco Protocol

รายละเอียดและตัวอย่างของคำสั่ง

(

--

 = 1 byte,

...

 = n bytes,

CR

 = Carriage Return)

1. คำสั่งที่ใช้อ่านค่า Digital Input

เริ่มต้นด้วย 'RDI' และจบด้วย '[CR]' เช่น อ่านค่า DI จากเครื่องหมายเลข 06 จะได้คำสั่งดังนี้ '#06RDI [CR]'

#	0	6	R	D	I	CR
---	---	---	---	---	---	----

โดยตัวโมดูลจะตอบกลับมาเป็น 'DI>' ตามด้วยค่าที่วัดได้ ทั้ง 32 ช่อง ช่องละ 1 ไบต์ รวม 32 ไบต์ (MSB -> LSB, '0' = OFF, '1' = ON) และจบด้วย '[CR]' ดังตัวอย่างนี้ 'DI>00100100000100101000000100100001[CR]'

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

2. คำสั่งที่ใช้อ่านค่า Digital Input (Hexadecimal)

คล้ายกับข้อ 1 แต่เปลี่ยนเป็นเริ่มต้นด้วย 'RDIH' และจบด้วย '[CR]' เช่น อ่านค่า DI จากเครื่องหมายเลข 0C จะได้คำสั่งดังนี้ '#0CRDIH [CR]'

#	0	C	R	D	I	H	CR
---	---	---	---	---	---	---	----

โดยตัวโมดูลจะตอบกลับมาเป็น 'DI>' ตามด้วยค่าที่วัดได้ โดยใช้รูปแบบของบิต ทั้งหมด 8 ไบต์ (MSB -> LSB, '0' = OFF, '1' = ON) และจบด้วย '[CR]' ดังตัวอย่างนี้ 'DI>24128121[CR]'

D	I	>	2	4	1	2	8	1	2	1	CR
---	---	---	---	---	---	---	---	---	---	---	----

รหัสตอบกลับมาเมื่อเกิดข้อผิดพลาดในการส่งคำสั่งไปยังตัวโมดูล DI2000

ในกรณีที่การส่งคำสั่งไปยังตัวโมดูลนั้น หากชุดคำสั่งนั้นไม่ถูกต้อง ตัวโมดูลจะไม่ทำคำสั่งชุดนั้น และรายงานความผิดพลาดที่เกิดขึ้นกลับมาเป็นรหัสต่างๆ โดยจะขึ้นต้นด้วย 'ERR=' แล้วตามด้วยตัวเลข ตั้งแต่ 1-2 ดังนี้

- | | |
|--------------------------|---|
| 1 (illegal function) | คำสั่งไม่ถูกต้อง หรือโมดูลไม่รู้จักคำสั่งนี้ |
| 2 (illegal data address) | ค่าตำแหน่งเริ่มต้น เกินช่วงตำแหน่งที่กำหนดไว้ |

สรุปคำสั่งที่ใช้กับตัวโมดูล DI2000 (Wisco Protocol)

((H) = Heximal Value, [CR] = carriage return)

Function	Command	DI2000 Response
RDI = Read Digital Input	#06RDI[CR]	DI>0010...0001[CR]
RDIH = Read Digital Input (H)	#0CRDIH[CR]	DI>24128121[CR]

การติดต่อกับโมดูลโดยใช้ MODBUS (ASCII) Protocol

โมดูล **DI2000** 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	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 **DI2000** สนับสนุนฟังก์ชันพื้นฐานของ Modbus ทั้งหมด 1 ฟังก์ชัน ดังต่อไปนี้

MODBUS ASCII

Wisco

READ DISCRETE INPUT (CODE 02)

= Read Digital Input

การอ้าง Address บนตัวโมดูลมีดังนี้

Function Code	Reference	Address
02	Digital Input	1xxxx

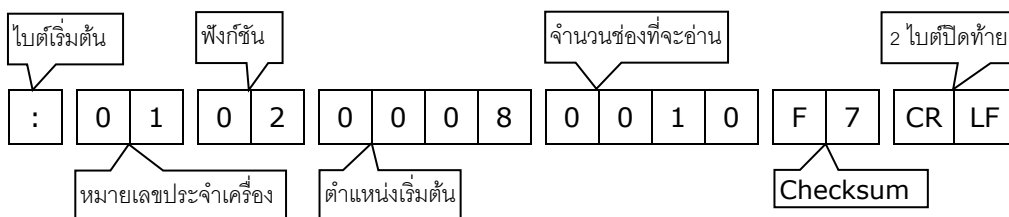
Digital Input Table

Name	Address
Digital Input Channel 1	10001
Digital Input Channel 2	10002
Digital Input Channel 3	10003
...	...
Digital Input Channel 31	10031
Digital Input Channel 32	10032

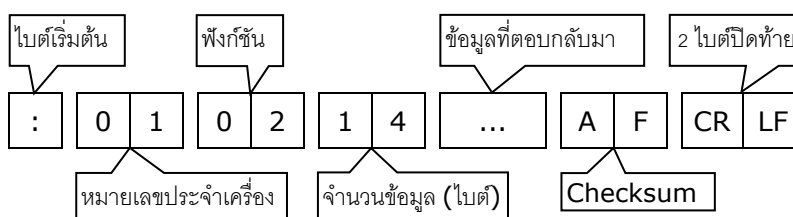
*รายละเอียดที่เหลือของ Modbus สามารถดูได้จาก 'Modbus Reference Guide' หรือที่ <http://www.modbus.org/specs.php>

ตัวอย่างฟังก์ชัน MODBUS (ASCII) PROTOCOL

Function Code 02



Response



วิธีคิด CHECK SUM สำหรับ MODBUS (ASCII) Protocol

ใน **MODBUS Protocol** จะใช้ CHECK SUM ในการตรวจสอบความถูกต้องของข้อมูลที่ส่งไปทุกคำสั่ง การคิด CHECK SUM นั้นจะใช้การบวกข้อมูลทั้งหมดเข้าด้วยกัน (บวกเฉพาะข้อมูลที่เป็นตัวเลขเท่านั้น) บวกกันครั้งละ 1 ไบต์โดยค่าที่เกิน 1 byte นั้นเราจะตัดทิ้ง จากนั้น นำค่าที่ได้ 1 byte นั้น มาทำ 1's complement และ 2's complement เป็นอันเรียบร้อย

ตัวอย่างเช่น `: 15 02 0008 0010 [CR] [LF]`

	HEXADECIMAL	BINARY
ไบต์เริ่มต้น	15H	0001 0101
	02H	0000 0010
	00H	0000 0000
	08H	0000 1000
	00H	0000 0000
ไบต์สุดท้าย	10H	0001 0000
ผลลัพธ์	2FH	0010 1111
คิดเฉพาะ 1 byte (8 bit)	2FH	0010 1111
ทำ 1's complement (invert)	D0H	1101 0000
ทำ 2' complement	D0H + 1	1101 0000 + 1
ค่า Check sum ที่ได้	D1H	1101 0001

ข้อมูลที่จะส่งจึงเป็น `: 15 02 0008 0010 D1 [CR] [LF]`

การตั้งค่าให้กับ **Dip Switch**

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

- ความยาว และ ความต้านทานของสาย
 - การรบกวนจากภายนอก
 - ถ้าติดต่อผ่านโมเด็ม ไม่ควรตั้ง Baud rate สูงมากนัก ซึ่งจะขึ้นอยู่กับคุณภาพของคู่สายโทรศัพท์
- ส่วนการกำหนด Protocol ที่ใช้ติดต่อกับโมดูล ให้เลือก Dipswitch ตำแหน่งที่ 8 ดังนี้

'0' = MODBUS RTU, '1' = MODBUS ASCII / WISCO PROTOCOL.

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

Edit: 19/04/2022