EXTIO2 I2C Protocol

V1 (FW Version)

2025/3/11

REG MAP (Addr:0x45)			0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	note
MODE SETTING		0x00 W/R	IO0	IO1	IO2	IO3	IO4	IO5	IO6	IO7									Mode:0~4[1]
1	OUTPUT CTRL	0x10 W	IO0	IO1	IO2	IO3	IO4	IO5	IO6	IO7									0:LOW ; 1:HIGH
0	DIGITAL INPUT	0x20 R	IO0	IO1	IO2	IO3	IO4	IO5	IO6	IO7									0:LOW ; 1:HIGH
2	ANALOG INPUT-8Bits	0x30 R	IO0	IO1	IO2	IO3	IO4	IO5	IO6	IO7									value:0~255
	ANALOG INPUT-12Bits	0x40 R	IO0-L	IO0-H	IO1-L	IO1-H	IO2-L	IO2-H	IO3-L	IO3-H	IO4-L	IO4-H	IO5-L	IO5-H	IO6-L	IO6-H	IO7-L	IO7-H	value:0~4095[2]
3	SERVO 8Bits	0x50 W/R	IO0	IO1	IO2	IO3	IO4	IO5	IO6	IO7									value:0~180degree
	SERVO 16Bits	0x60 W/R	IO0-L	IO0-H	IO1-L	IO1-H	IO2-L	IO2-H	IO3-L	IO3-H	IO4-L	IO4-H	IO5-L	IO5-H	IO6-L	IO6-H	IO7-L	IO7-H	value:500~2500us[3]
4	RGB 24Bits	0x70 W/R	IO0-R	IO0-G	IO0-B	IO1-R	IO1-G	IO1-B	IO2-R	IO2-G	IO2-B	IO3-R	IO3-G	IO3-B	IO4-R	IO4-G	IO4-B	IO5-R	R/G/B:0~255[4]
		0x80 W/R	IO5-G	IO5-B	IO6-R	IO6-G	IO6-B	IO7-R	IO7-G	IO7-B									
I2C ADDRESS SETTING		0xF0 W/R																Addr	value: 1~127 default:0x45
Firmware version		0xF0 R															Version		Version: firmware version

[1] 0: Input, 1: Output, 2: ADC, 3: Servo, 4: NeoPixel

[2] The address for reading a 12-bit ANALOG INPUT must be 2-byte aligned, and the number of bytes read must be 2 bytes.
(1) Correct reading examples:
Read 0x40, 2 bytes; Read 0x48, 2 bytes.
(2) Incorrect reading examples:
Read 0x40, 1 byte; Read 0x41, 2 bytes; Read 0x48, 4 bytes.

[3] The address for writing a 16-bit SERVO must be 2-byte aligned, and the number of bytes written must be 2 bytes.
(1) Correct writing examples:
Write 0x60, 2 bytes; Write 0x68, 2 bytes.
(2) Incorrect writing examples:
Write 0x60, 1 byte; Write 0x61, 2 bytes; Write 0x68, 4 bytes.

[4] The address for writing a 24-bit RGB must be 3-byte aligned, and the number of bytes written must be 3 bytes.
(1) Correct writing examples:
Write 0x70, 3 bytes; Write 0x79, 3 bytes.
(2) Incorrect writing examples:
Write 0x70, 1 byte; Write 0x71, 3 bytes; Write 0x79, 6 bytes.

[5] The control of servo motors is achieved by software-driven IO toggling on the microcontroller. If computationally intensive operations such as frequent I2C read/write operations are performed while using servo, it may cause jitter in the servo.