Unit EXT.IO2

SKU:U011-B

Description

Unit EXT.IO2 is an **IO expansion unit** developed based on the STM32F030 main controller, using an I2C communication interface, providing 8-channel IO expansion. Each IO channel supports independent configuration of **digital input/output**, **ADC**, **SERVO control**, and **RGB LED control** modes. It supports configuring the device's I2C address, which means users can mount multiple **Unit EXT.IO2** units on the same I2C bus to expand more IO resources. It is suitable for applications such as multi-channel digital/analog signal acquisition and light/servo control.

Features

- 8-channel input/output expansion:
  - Digital input/output
  - ADC input
  - SERVO control (PWM)
  - RGB LED control
- I2C communication interface:
  - Supports I2C address configuration

Includes

- 1 x Unit EXT.IO2
- 1 x HY2.0-4P Grove connection cable (20cm)

Applications

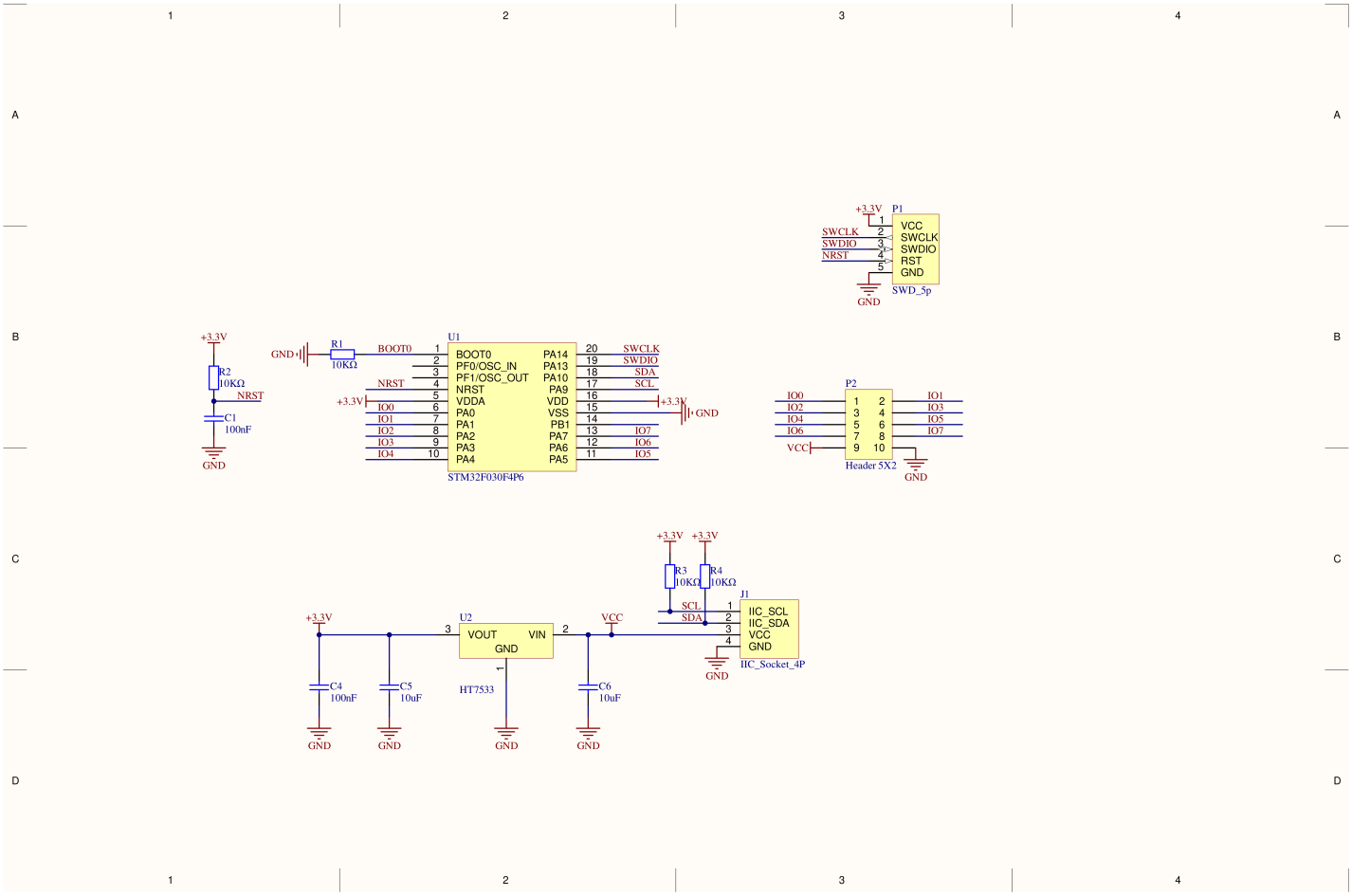
- IO expansion
- Servo control
- Multi-channel light control
- Multi-channel analog signal acquisition

Specifications

Specification	Parameter
MCU	STM32F030F4P6
Communication Interface	I2C @0x45
Number of IO Expansions	8
IO Interface Pin Spacing	2.54mm
IO Supported Modes	Digital input/output, ADC, SERVO control, RGB LED control
IO Supported Input/Output Level	3.3V
Product Size	32.0 x 24.0 x 10.2mm
Product Weight	5.0g
Package Size	138.0 x 93.0 x 11.2mm
Gross Weight	10.0g

Schematics

- Unit EXT.IO2 Schematics PDF

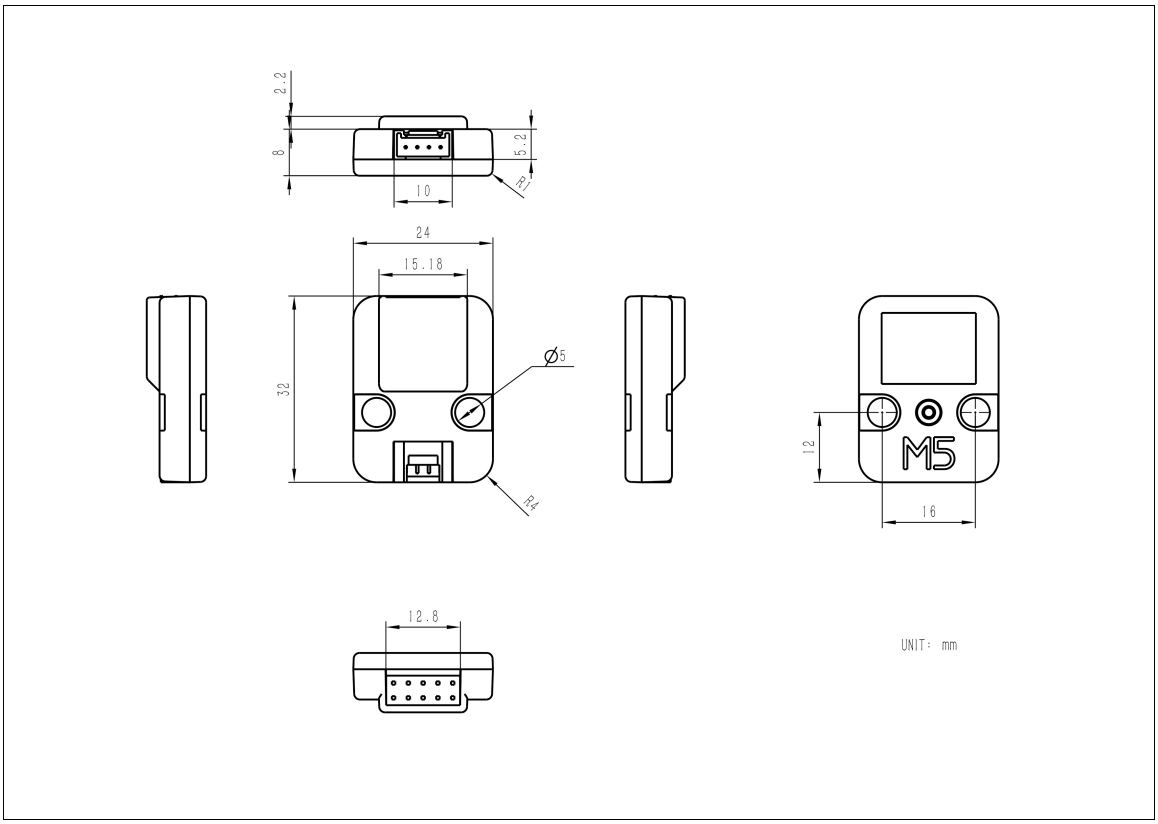


PinMap

Unit EXT.IO2

HY2.0-4P	Black	Red	Yellow	White
PORT.A	GND	5V	SDA	SCL

Model Size



Softwares

Arduino

- Unit EXT.IO2 ADC Input Example
- Unit EXT.IO2 Digital Input/Output Example
- Unit EXT.IO2 RGB LED Control Example
- Unit EXT.IO2 Servo Control Example

UiFlow1

- Unit EXT.IO2 UiFlow1 Docs

UiFlow2

- Unit EXT.IO2 UiFlow2 Docs

Internal Firmware

- Unit EXT.IO2 Internal Firmware

Firmware Version	Update	Protocol
v1	First Release Version	Unit EXT.IO2 I2C Protocol v1
v3	1. Add PWM mode 2.Add I2C IAP upgrade function	Unit EXT.IO2 I2C Protocol v3

M5 DAPLink

If you don't have the STM32 downloader tool, you can refer to the M5 DAPLink tutorial and use the Core2 or CoreS3 as a programmer to complete the firmware update for the device.

Protocol

EXTIO2 I2C Protocol																		V3 (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:0~5[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[3]
		0x80 W/R	IO5-G	IO5-B	IO6-R	IO6-G	IO6-B	IO7-R	IO7-G	IO7-B									
5	PWM DutyCycle	0x90 W/R	pwm 0	pwm1	pwm 2	pwm 3	pwm 4	pwm 5	pwm 6	pwm 7									DutyCycle:0~100
6	PWM Frequency	0xA0 W/R	frequency																0:2KHz,1:1KHz (default),2:500Hz,3:250Hz,4:125Hz
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, 5: PWM

[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 and PWM is achieved by software-driven IO toggling on the microcontroller. If servo control and PWM are used simultaneously, or if computationally intensive operations such as frequent I2C read/write operations are performed while using servo and PWM, it may cause jitter in the servo or PWM waveform.

Video