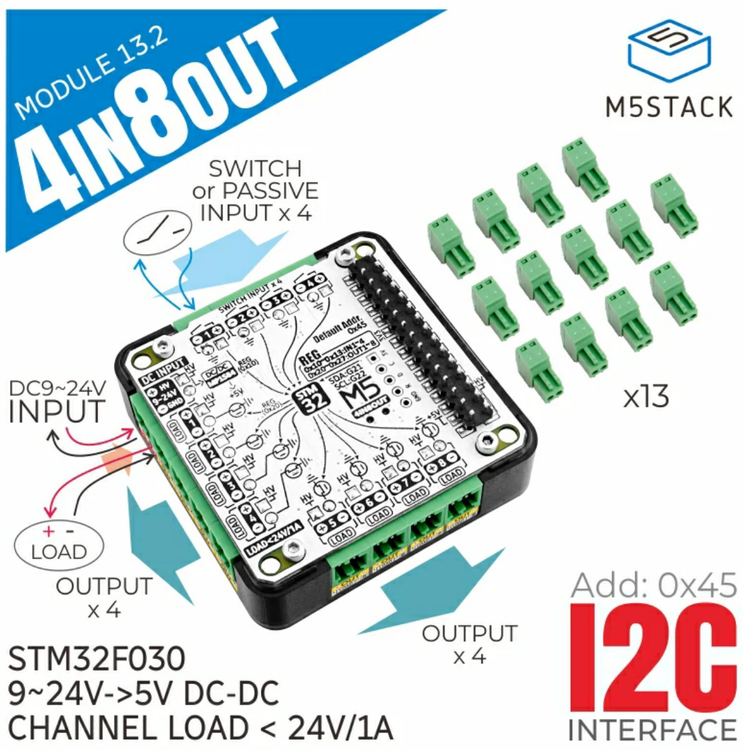
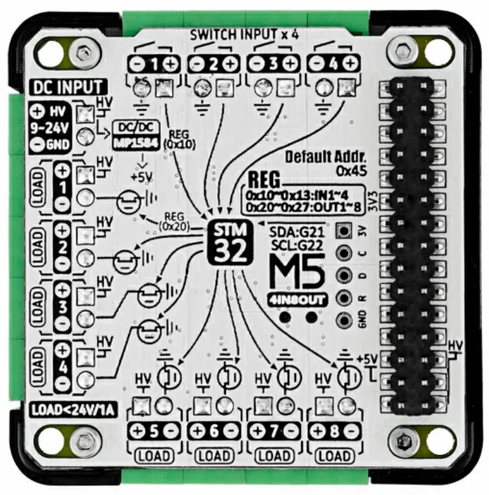
Module 13.2 4IN8OUT

SKU:M122





Description

MODULE 13.2 4IN8OUT is a '4-Channel Passive Input/Switch + 8-Channel MOS Drive Output' drive module. Adopt STM32F030 as the I2C IO chip, support 9-24V DC input, with internal DC-DC circuitry to 5V.

Product Features

- Compatible with Basic/Fire/Core2
- Adopt STM32F030 as the IO chip, I2C communication, I2C address is changeable by modifying the register
- 8-channel MOS Drive circuitry share the same VCC, channle load <1A
- 4-channel Passive Input share the same GND, no active signal or >5V signal allowed
- With internal MP1584, 9~24V -> 5V DC-DC converter

Include

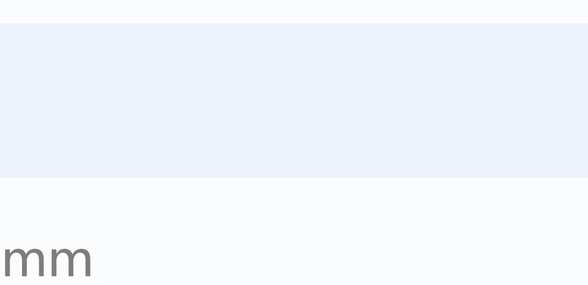
- 1x MODULE 13.2 4IN8OUT
- 13x 2P Terminal

Application

- Multi-channel load drive (Relay, Valva, Single-phase Motor, Singal LED)
- Limit Switch or Button test

Specification

Spec	Parameter
Input Voltage	9~24V
Output Channel	8
Input Channel	4
Output Load	<1A each Channel
Communication Interface	I2C
I2C Address	Default 0x45, changeable by modifying the register 0xF0
Net Weight	21.9g
Gross Weight	52.5g
Product Size	54*54*13mm
Package Size	125*68*23mm

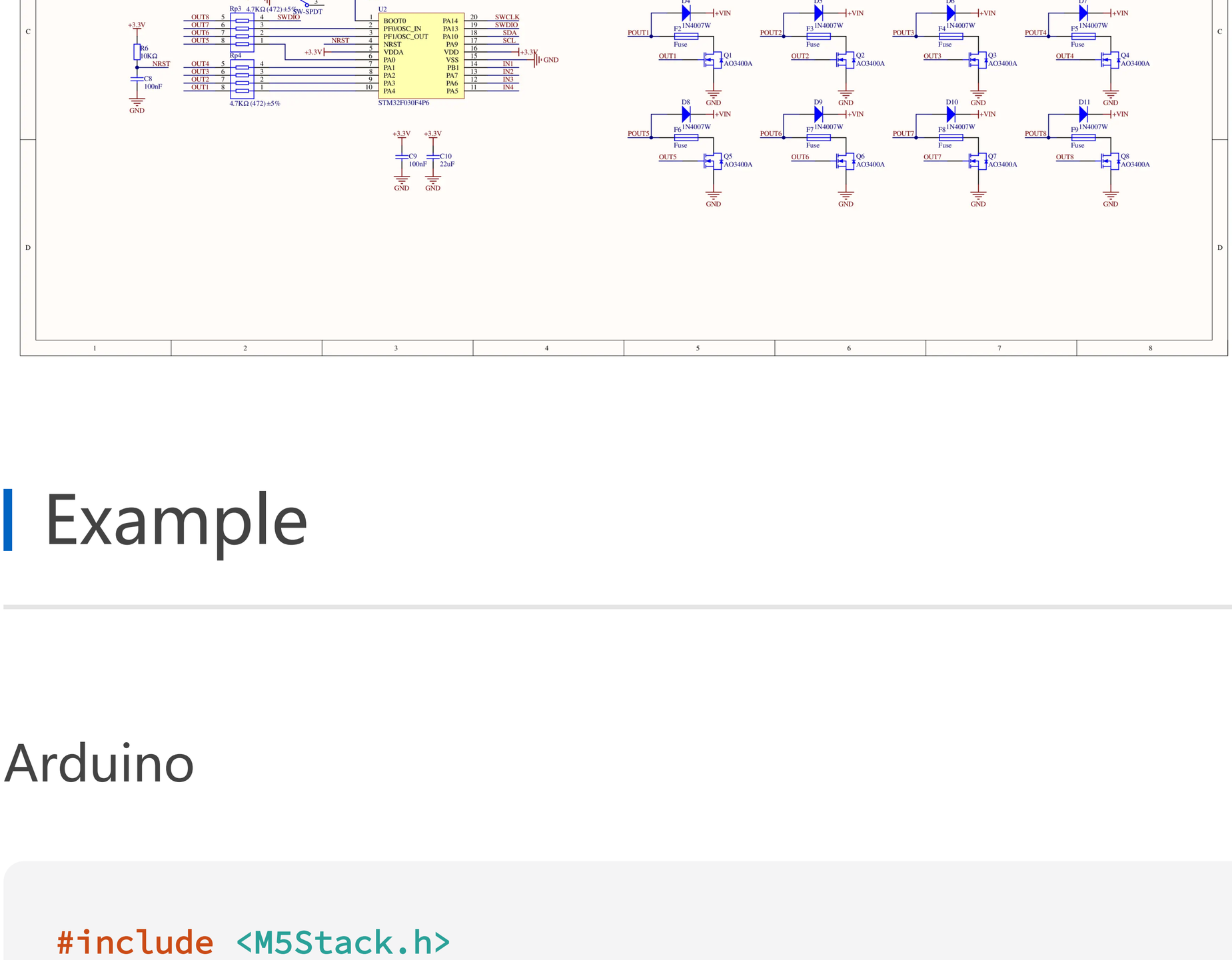


Pinmap

M5Core	GPIO22	GPIO21	5V	GND
4IN8OUT MODULE 13.2	SCL	SDA	5V	GND

REG MAP (Addr:0x45)		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	note
Switch input	0x10 R	Input0	Input1	Input2	Input3													**Input:** value of switch input, 0 or 1
Load output	0x20 R/W	Output 0	Output 1	Output 2	Output 3	Output 4	Output 5	Output 6	Output 7									**Output:** 0: Off; 1: On
Firmware version	0xF0 R															Version		**Version:** firmware version
I2C Address	0xF0 R/W																Addr	**Addr:** I2C address: 0~127

Schematic



Example

Arduino

```
#include <M5Stack.h>
#include "MODULE_4IN8OUT.h"

MODULE_4IN8OUT module;

int _I2C_dev_scan();
```

```
void setup() {
    M5.begin(1,1,1,1);  // Init M5Stack.  初始化M5Stack

    // while (1) {
    //     _I2C_dev_scan();
    //     delay(1000);
    // }


    while (!module.begin(&Wire, 21, 22, MODULE_4IN8OUT_ADDR)) {
        Serial.println("4IN8OUT INIT ERROR");
        M5.Lcd.println("4IN8OUT INIT ERROR");
        _I2C_dev_scan();
        delay(1000);
    };
    Serial.println("4IN8OUT INIT SUCCESS");
}

// void loop() {

// }

long interval = 0;
bool level    = false;

void loop() {
    for (uint8_t i = 0; i < 4; i++) {
        if (module.getInput(i) != 1) {
            // M5.Lcd.fillRect(60 + 60 * i, 0, 25, 25, TFT_BLACK);
            M5.Lcd.fillRect(60 + 60 * i, 0, 25, 25, TFT_GREEN);
        } else {
            // M5.Lcd.fillRect(60 + 60 * i, 0, 25, 25, TFT_BLACK);
            // M5.Lcd.drawRect(60 + 60 * i, 0, 25, 25, TFT_GREEN);
            M5.Lcd.fillRect(60 + 60 * i, 0, 25, 25, TFT_RED);
        }
        M5.Lcd.drawString("IN" + String(i), 40 + 60 * i, 5);
    }
    M5.Lcd.drawString("4IN8OUT MODULE", 60, 80, 4);
    // M5.Lcd.drawString("FW VERSION:" + String(module.getVersion()), 70,
120, 4);
    if (millis() - interval > 1000) {
        interval = millis();
        level    = !level;
        for (uint8_t i = 0; i < 8; i++) {
            module.setOutput(i, level);
            if (level) {
                M5.Lcd.fillRect(20 + 35 * i, 200, 25, 25, TFT_BLACK);
                M5.Lcd.fillRect(20 + 35 * i, 200, 25, 25, TFT_BLUE);
            } else {
                M5.Lcd.fillRect(20 + 35 * i, 200, 25, 25, TFT_BLACK);
```

```
                M5.Lcd.drawRect(20 + 35 * i, 200, 25, 25, TFT_BLUE);
            }
            M5.Lcd.drawString("OUT" + String(i), 18 + 35 * i, 180);
            // delay(50);
        }
    }
    // if (M5.BtnB.wasPressed()) {
    //     if (module.setDeviceAddr(0x66)) {
    //         Serial.println("Update Addr: 0x66");
    //     }
    //     // }
    //     // M5.update();

    delay(500);
}


int _I2C_dev_scan() {
    uint8_t error, address;
    int nDevices;

    Serial.println("[I2C_SCAN] device scanning...");

    nDevices = 0;
    for (address = 1; address < 127; address++ ) {
        // The i2c_scanner uses the return value of
        // the Write.endTransmisstion to see if
        // a device did acknowledge to the address.
        Wire.beginTransmission(address);
        error = Wire.endTransmission();

        if (error == 0) {
            Serial.print("[I2C_SCAN]: device found at address 0x");
            if (address < 16)
                Serial.print("0");
            Serial.print(address, HEX);
            Serial.println(" !");


            nDevices++;
        }
        else if (error == 4) {
            Serial.print("[I2C_SCAN]: unknow error at address 0x");
            if (address < 16)
                Serial.print("0");
            Serial.println(address, HEX);
        }
    }

    Serial.print("[I2C_SCAN]:");
```

```
        Serial.printf(" %d devices was found\r\n", nDevices);
        return nDevices;
    }
```

UIFlow