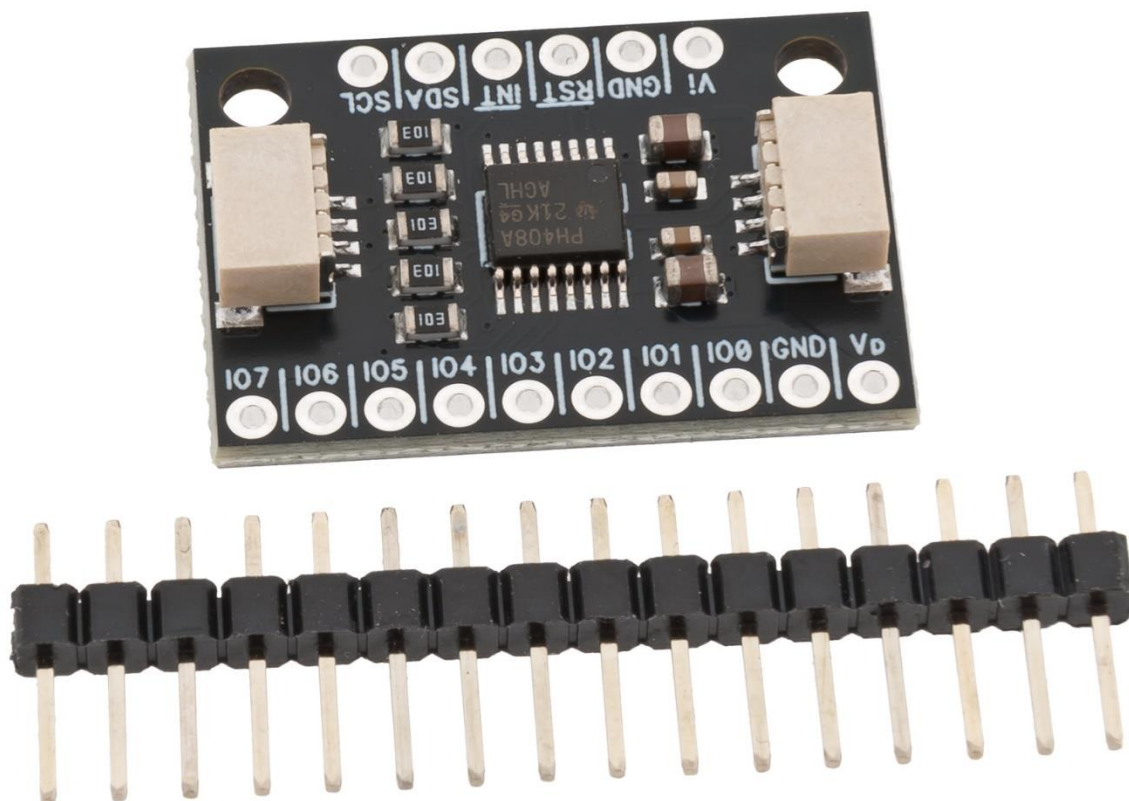


TCA6408A 8ch I2C IO Expander

Part.no. 41024442



TCA6408A is an 8-port IO expander for I2C with built-in level translator. The port expander is bidirectional, which means that each port can be configured as either an input or an output.

The chip is also capable of level translation and has separate voltage supplies for the I2C side (VCCi) and the port side (VCCp). The separate voltage supplies makes it easy to control higher or lower voltage systems without the need for external level shifters.

The outputs can handle relatively high currents to directly drive LEDs and is capable of sourcing up to 10mA or sinking up to 25mA. An interrupt pin is available for monitoring state changes when pins are set as inputs. The interrupt pin will go low momentarily any time an input changes state.

Up to two TCA6408A can be used on the same I2C bus. A solder jumper on the back sets the address.

Two QWIIC connectors are available for fast and easy wiring.

Functions

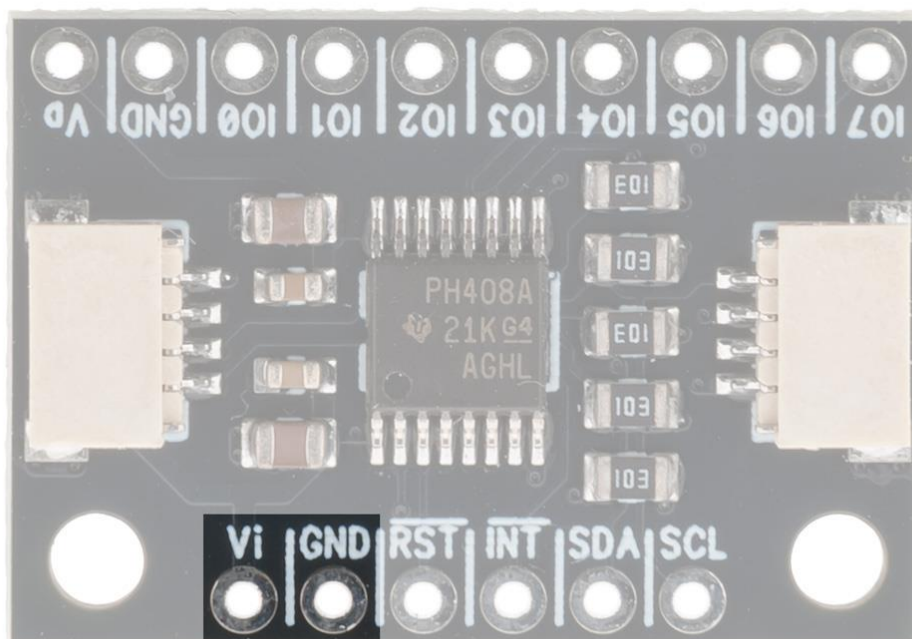
- Add 8 GPIOs via I2C
- Bidirectional functionality (in or out)
- Level translating
- High drive strength
- QWIIIC connectors for easy wiring

Specifications

- Supply voltage (VCCi): 1.65 - 5.5 VDC
- Supply voltage (VCCp): 1.65 - 5.5 VDC
- Max current (per pin): 25 mA (sinking) / 10 mA (sourcing)
- I2C bus speed: 100 kHz / 400 kHz
- I2C address: 0x20 (default) / 0x21 (solder jumper)
- Dimensions: 26 x 18 mm
- Mounting holes: c-c 21 mm / $\varnothing$ 2.5 mm

Connections

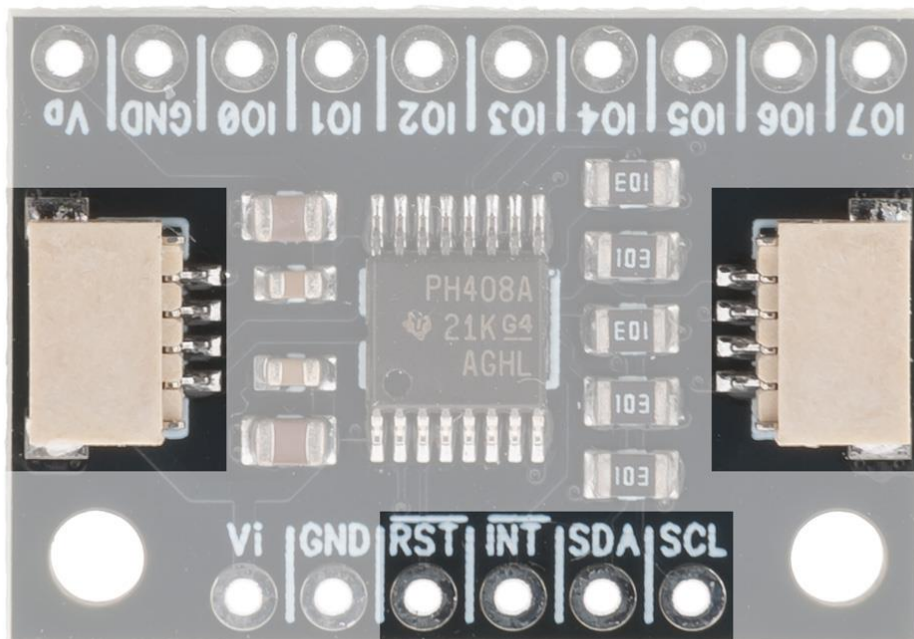
Voltage supply (VCCi):



Vi and GND are the voltage supply inputs for the I2C side of the chip and should be supplied with 1.65 - 5.5V. When using the QWIIIC connectors to wire the module, these pins can be left unused.

Note that the VCCi sets the signalling levels for the I2C communication.

I2C connections:



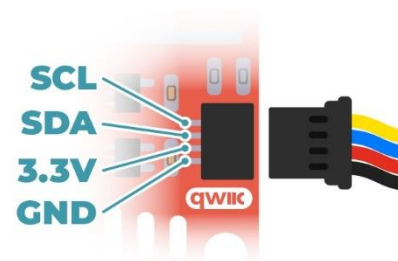
Four pins are available for communication with the port expander. SDA (serial data) and SCL (serial clock) are the two required pins, and should be wired to the corresponding pins on the microcontroller.

RST is an input and used for resetting the chip. Pull the pin to GND to reset the chip.

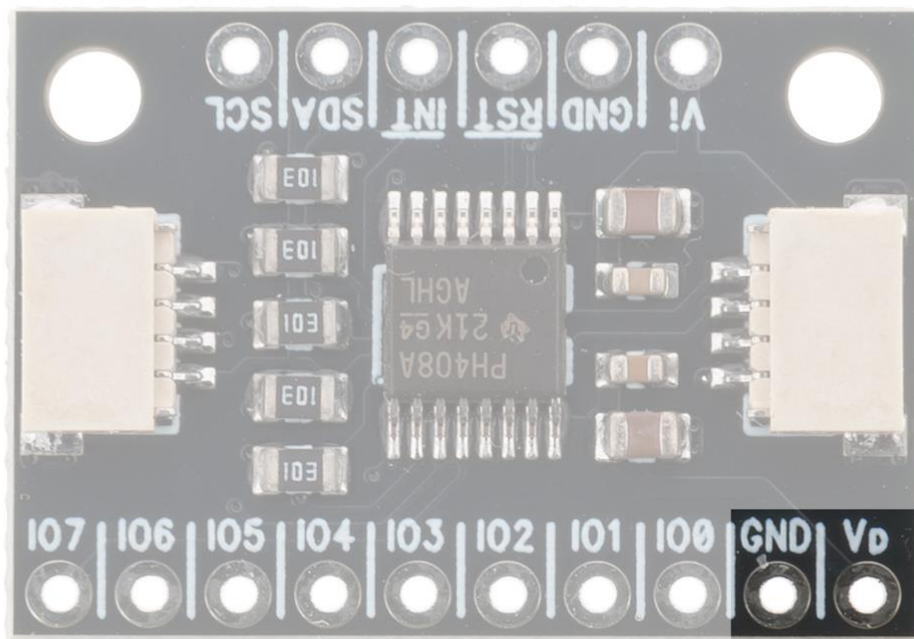
INT is an output that will momentarily go low whenever a pin configured as an input changes state relative to the stored state in the pin register. This output is normally high. The interrupt pin can be monitored by the microcontroller to take action as soon as a button is pressed or a sensor is transmitting data without the need for bidirectional communication via I2C.

For easier wiring, two QWIIC connectors on each end contains VCCi, GND, SCL and SDA, and can be used for the I2C communication with the microcontroller. The QWIIC pinout and specifications can be found at [Sparkfun](https://www.sparkfun.com/qwiic/).

The SDA and SCL lines are pulled to VCCi using 10k resistors. Solder jumpers on the rear can be cut to disable the pullups.



Voltage supply (VCCp):

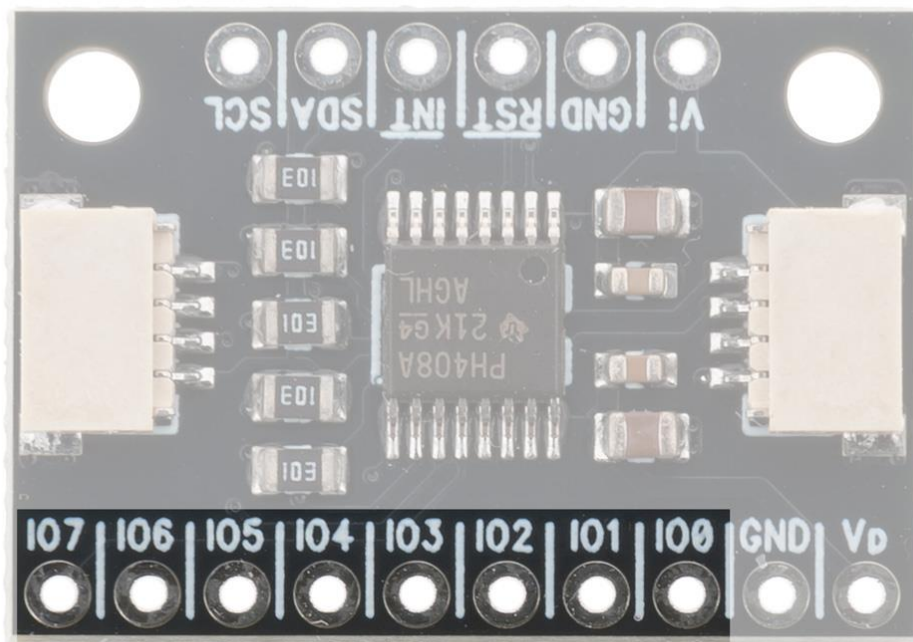


VCCp and GND are the voltage supply inputs for the port side of the chip. These pins have the same range as VCCi (1.65 - 5.5V) and sets the voltage level for the ports.

VCCp can be lower, higher or equal to VCCi. The pins must be supplied for the ports to function. If no level translation is needed, bridge VCCi to VCCp.

The GND pin is connected to the same net as the GND pin on the opposite side of the board.

Ports:

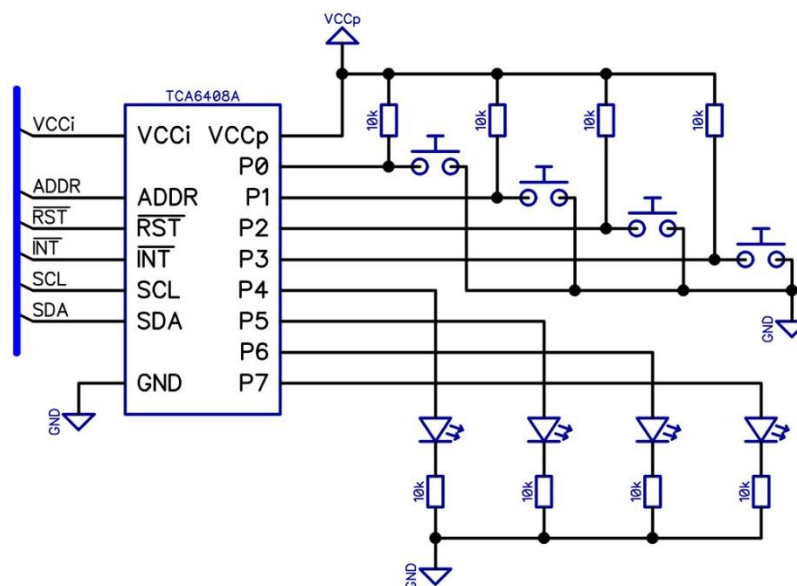
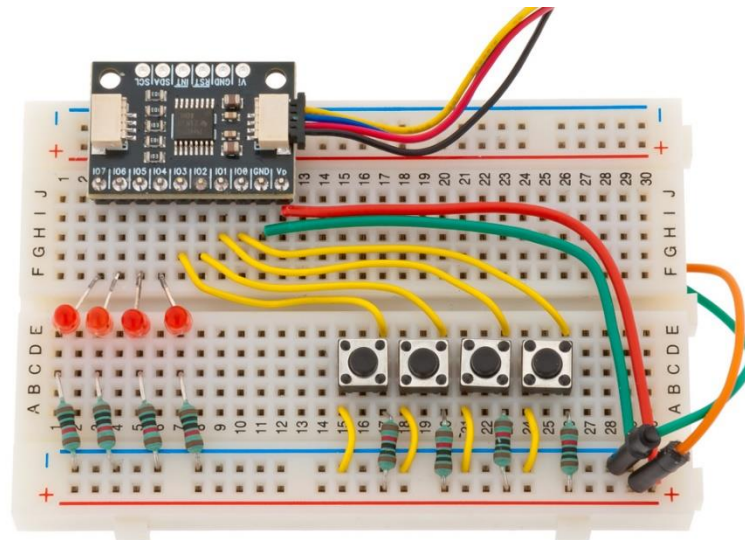


IO0 - IO7 are the eight channels available. All pins are set as inputs at startup to prevent glitches.

Maximum drive strength depends on whether the pin is used for sourcing or sinking current, but is normally sufficient for driving LEDs and other small loads.

Note that the IO pins does not have internal pullup or pulldown resistors and must be added externally if needed.

Example code and hookup



The example code and demonstration hookup configures the chip to use IO0 - IO3 as inputs with buttons and pullup resistors to VCCp and IO4 - IO7 as outputs with LEDs and current limiting resistors wired to GND.

```
/*
```

Example code for Electrokit EKM017 TCA6408A I2C IO Expander.

Wire four buttons to I00 - I03 and ~10k pullup resistors to Vp (TCA6408A does not have internal pullups).

Wire four LEDs to I04 - I07 and suitable resistors to GND.

The code will:

- Read I00 - I03
- Output the pin states to serial monitor
- Set I04 - I07 high when input goes low

```
*/
```

```
#include <Wire.h>
```

```
#define TCA6408A_ADDR 0x20
```

```
// Register addresses
```

```
#define REG_INPUT      0x00
```

```
#define REG_OUTPUT     0x01
```

```
#define REG_POLARITY   0x02
```

```
#define REG_CONFIG     0x03
```

```
void writeRegister(uint8_t reg, uint8_t value) {  
    Wire.beginTransmission(TCA6408A_ADDR);  
    Wire.write(reg);  
    Wire.write(value);  
    Wire.endTransmission();  
}
```

```
uint8_t readRegister(uint8_t reg) {  
    Wire.beginTransmission(TCA6408A_ADDR);  
    Wire.write(reg);  
    Wire.endTransmission(false);  
    Wire.requestFrom((uint8_t)TCA6408A_ADDR, (uint8_t)1);  
    return Wire.available() ? Wire.read() : 0xFF;  
}
```

```
void setup() {  
    Wire.begin();  
    Serial.begin(9600);  
  
    // Set P0-P3 as inputs, P4-P7 as outputs  
    writeRegister(REG_CONFIG, 0x0F);    // 00001111  
  
    // Invert input polarity (so pressing button = 1)  
    writeRegister(REG_POLARITY, 0x0F); // 00001111  
  
    // Initialize outputs to off (all LEDs off)  
    writeRegister(REG_OUTPUT, 0x00);  
}
```

```
void loop() {  
    // Read inputs  
    uint8_t input = readRegister(REG_INPUT) & 0x0F;  
  
    // Shift input bits to upper nibble to drive P4-P7  
    uint8_t output = input << 4;  
  
    // Write to output register  
    writeRegister(REG_OUTPUT, output);  
  
    // Print input/output states  
    Serial.print("Inputs (P0-P3): ");  
    Serial.print(input, BIN);  
    Serial.print(" | Outputs (P4-P7): ");  
    Serial.println(output >> 4, BIN);  
  
    delay(20);  
}
```

Additional resources

