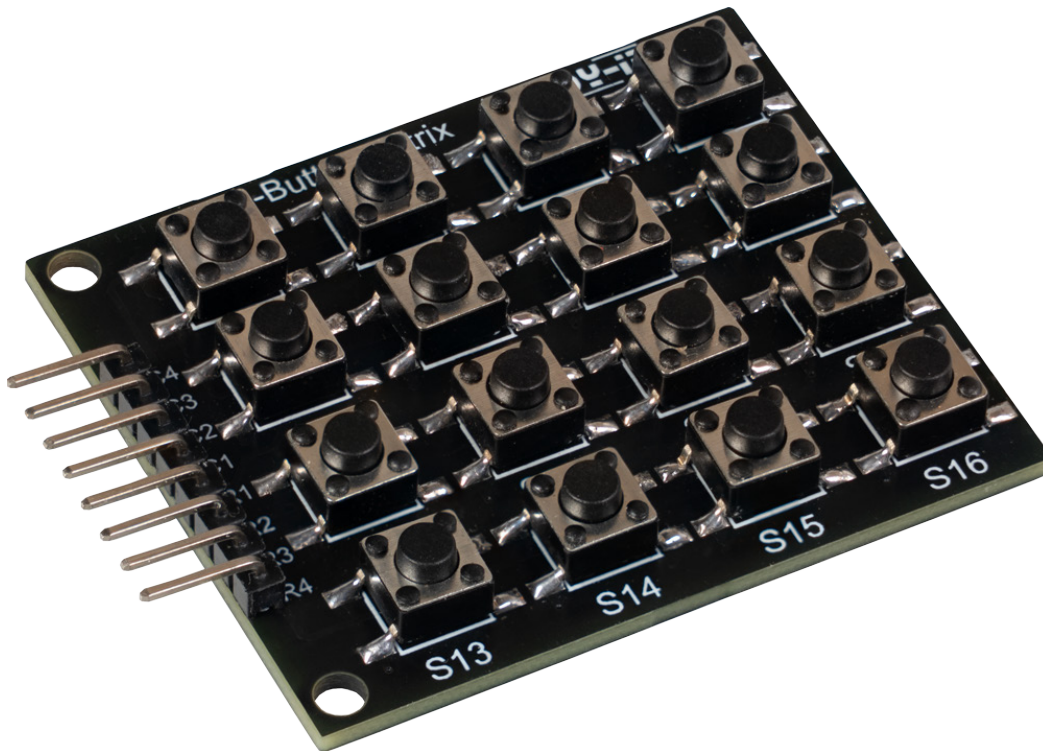


BUTTON MATRIX

4 x 4 Buttons



1. GENERAL INFORMATION

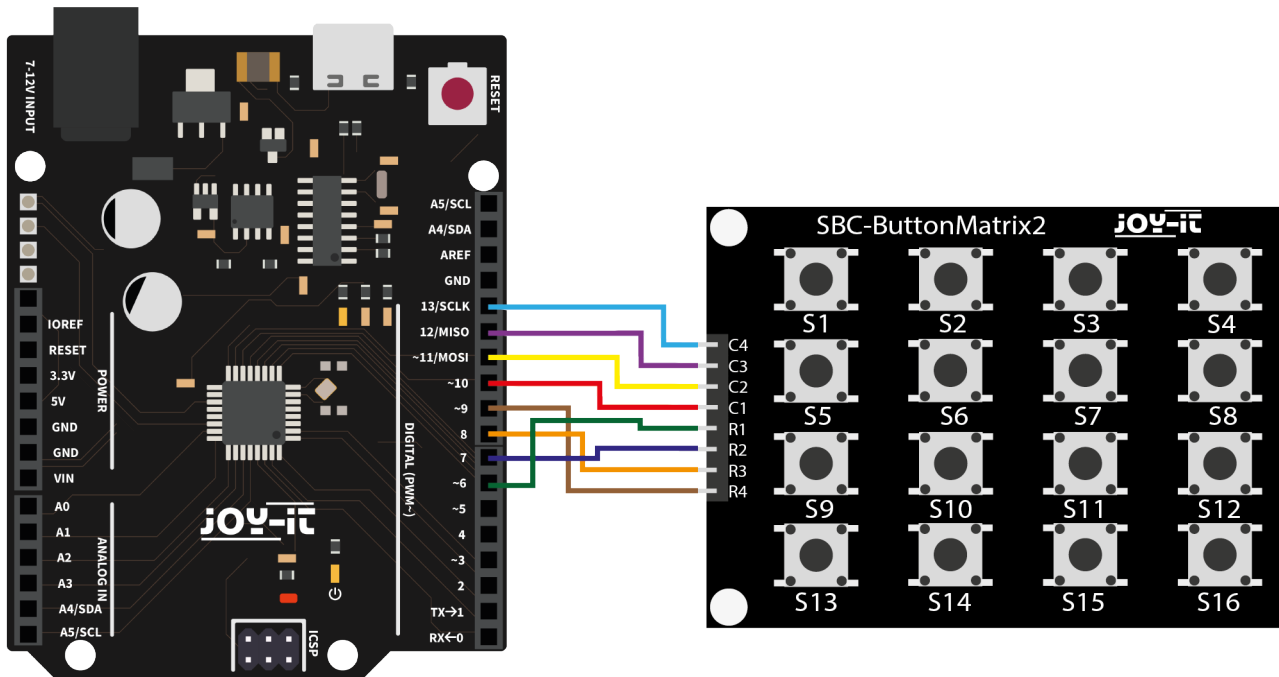
Dear customer,

thank you for choosing our product. In the following, we will show what you should note at the commissioning and during the usage.

Should you encounter any unexpected problems during use, please do not hesitate to contact us.

2. USE WITH THE ARDUINO

Connect the matrix



Arduino	Buttonmatrix
13	C4
12	C3
11	C2
10	C1
9	R4
8	R3
7	R2
6	R1

Connect the digital pins (6-13) of the Arduino with the ButtonMatrix like shown in the picture and in the chart.

Installation of the matrix

In the following, you can find a working code example which can be transferred with the Arduino.

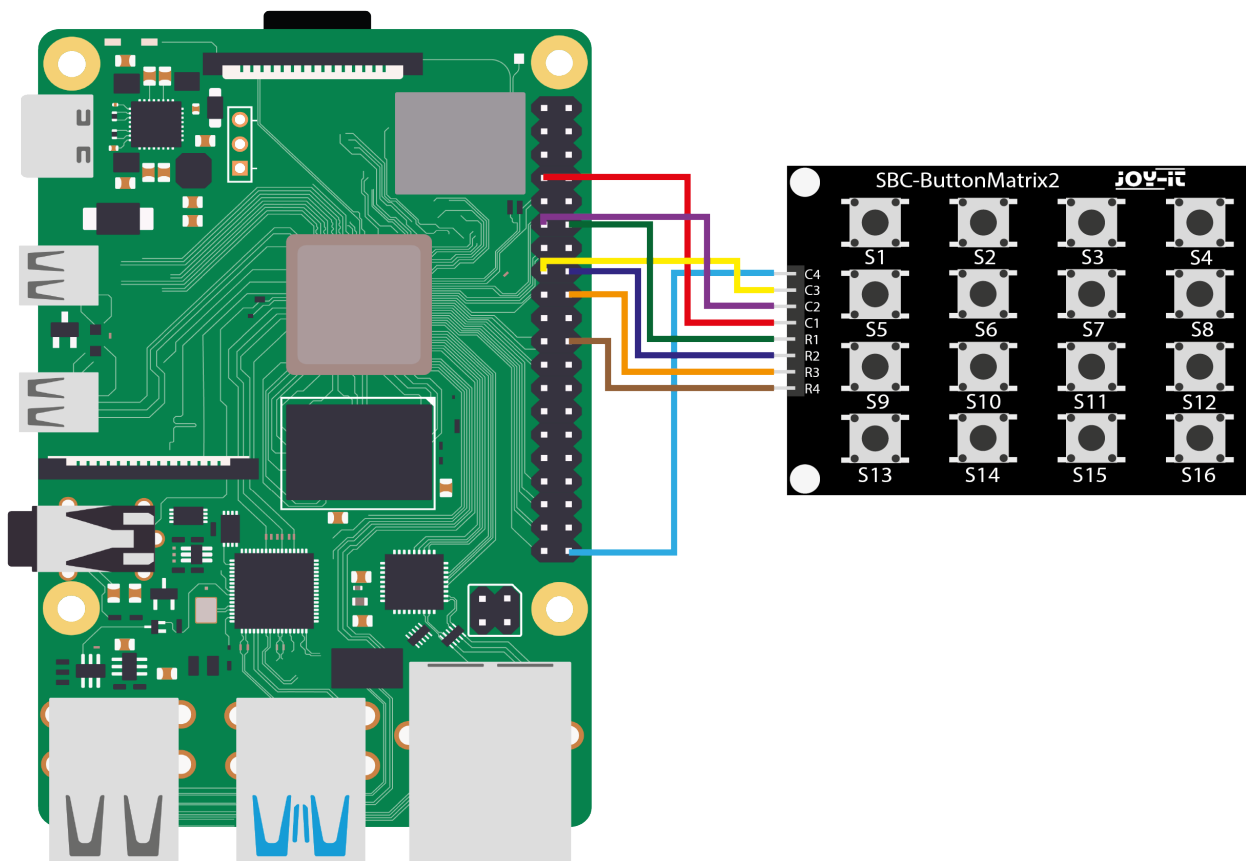
The function of the containing buttons can be expanded in the **knopfDruck**-function according to your wishes.

```
// -----  
// Pin mapping  
// -----  
const byte rowPins[4] = {6, 7, 8, 9}; // R1 -> D6, R2 -> D7, R3 ->  
D8, R4 -> D9  
const byte colPins[4] = {10, 11, 12, 13}; // C1 -> D10, C2 -> D11, C3 ->  
D12, C4 -> D13  
  
// -----  
// Key labels stored in a 2-D array  
// (strings so that "10", "11", etc. are possible)  
// -----  
const char* keys[4][4] = {  
  {"1", "2", "3", "4"},  
  {"5", "6", "7", "8"},  
  {"9", "10", "11", "12"},  
  {"13", "14", "15", "16"}  
};  
  
// -----  
// Helper array for remembering the  
// key state (pressed / not pressed)  
// -----  
bool lastState[4][4] = {  
  false, false, false, false,  
  false, false, false, false,  
  false, false, false, false,  
  false, false, false, false  
};  
  
void setup() {  
  Serial.begin(9600);  
  Serial.println("Starting 4x4 keypad scan without a library...");  
  
  // Column pins as OUTPUT, HIGH = idle  
  for (byte c = 0; c < 4; c++) {  
    pinMode(colPins[c], OUTPUT);  
    digitalWrite(colPins[c], HIGH);  
  }  
  
  // Row pins as INPUT_PULLUP  
  for (byte r = 0; r < 4; r++) {  
    pinMode(rowPins[r], INPUT_PULLUP);  
  }  
}
```

```
void loop() {  
  // Drive one column at a time  
  for (byte col = 0; col < 4; col++) {  
    // Drive the active column LOW  
    digitalWrite(colPins[col], LOW);  
  
    // Read all rows  
    for (byte row = 0; row < 4; row++) {  
      // A key is pressed when the row pin reads LOW  
      bool isPressed = (digitalRead(rowPins[row]) == LOW);  
  
      // Report only on a transition from "not pressed" to  
      "pressed"  
      if (isPressed && !lastState[row][col]) {  
        Serial.print("Key pressed: ");  
        Serial.println(keys[row][col]);  
      }  
  
      // Store current state  
      lastState[row][col] = isPressed;  
    }  
  
    // Restore column to HIGH  
    digitalWrite(colPins[col], HIGH);  
  }  
  
  // Small delay for debouncing / easing CPU load  
  delay(50);  
}
```

3. USE WITH THE RASPBERRY PI

Connect the matrix



Raspberry Pi	ButtonMatrix
PIN 40 (BCM 21)	C4
PIN 15 (BCM 22)	C3
PIN 11 (BCM 17)	C2
PIN 7 (BCM 4)	C1
PIN 12 (BCM 18)	R1
PIN 16 (BCM 23)	R2
PIN 18 (BCM 24)	R3
PIN 22 (BCM 25)	R4

Connect the ButtonMatrix with the pins of the Raspberry Pi like shown in the picture and the chart.

Installation of the matrix

First of all, you have to create a new Python-file:

```
nano matrix.py
```

Write this code completely into the editor which has opened.

```
import RPi.GPIO as GPIO
import time
class ButtonMatrix():
    def __init__(self):
        GPIO.setmode(GPIO.BCM)
        # The IDs of the buttons are defined
        self.buttonIDs = [[4,3,2,1],[8,7,6,5],[12,11,10,9],[16,15,14,13]]
        # GPIO pins for the rows are declared
        self.rowPins = [18,23,24,25]
        # GPIO pins for the columns are declared
        self.columnPins = [21,22,17,4]
        # Define four inputs with pull-up resistors
        for i in range(len(self.rowPins)):
            GPIO.setup(self.rowPins[i],GPIO.IN,pull_up_down=GPIO.PUD_UP)
        # Define four outputs and set them to high
        for j in range(len(self.columnPins)):
            GPIO.setup(self.columnPins[j], GPIO.OUT)
            GPIO.output(self.columnPins[j], 1)
    def activateButton(self, rowPin, colPin):
        # Get the button number
        btnIndex = self.buttonIDs[rowPin][colPin] - 1
        print("button " + str(btnIndex + 1) + " pressed")
        # Prevents multiple button presses in a short time
        time.sleep(.3)
    def buttonHeldDown(self,pin):
        if(GPIO.input(self.rowPins[pin]) == 0):
            return True
        return False
def main():
    # Initialization of the button matrix
    buttons = ButtonMatrix()
    try:
        while(True):
            for j in range(len(buttons.columnPins)):
                # Each output pin is set to low
                GPIO.output(buttons.columnPins[j],0)
                for i in range(len(buttons.rowPins)):
                    if GPIO.input(buttons.rowPins[i]) == 0:
                        buttons.activateButton(i,j)
                    # Do nothing while the button is held down
                    while(buttons.buttonHeldDown(i)):
                        pass
                GPIO.output(buttons.columnPins[j],1)
            except KeyboardInterrupt:
                GPIO.cleanup()
if __name__ == "__main__":
    main()
```

The file can be saved with **Strg+O** and the editor can be left with **Strg+X**. After that the file can be run with the following command and that way it will be tested.

```
python3 matrix.py
```

The file can be closed again with **Strg+C**.

4. FURTHER INFORMATION

Our information and redemption obligation according to the electro-law (ElektroG)



Symbol on electrical and electronic products :

This crossed-out bin means that electrical and electronic products do **not** belong into the household waste. You must hand over your old appliance to a registration office. Before you can hand over the old appliance, you must remove used batteries and accumulators which are not enclosed by the device.

Return options :

As the end user, you can hand over with the purchase of a new device your old appliance (which has essentially the same functions as the new one) free of charge for disposal. Small devices which do not have outer dimensions greater than 25 cm can be submitted independently of the purchase of a new product in normal household quantities.

Possibility of restitution at our company location during our opening hours :

SIMAC GmbH, Pascalstr. 8, D-47506 Neukirchen-Vluyn

Possibility of restitution nearby :

We send you a parcel stamp with which you can send us your old appliance free of charge. For this possibility, you must contact us via e-mail at service@joy-it.net or via telephone.

Information about packaging:

Please package your old appliance safe during transport. Should you not have a suitable packaging material or you do not want to use your own material, you can contact us and we will send you an appropriate package.



5. SUPPORT

If any questions remain open or problems arise after your purchase, we are available by e-mail, telephone and with a ticket support system to answer these.

E-Mail: service@joy-it.net

Ticket-System: <https://support.joy-it.net>

Telephone: +49 (0)2845 9360 – 50

For further information visit our website:s

www.joy-it.net