Multiplexeur TCA9548A

Multiplexeur TCA9548A VideoYT

►Fiche technique du TCA9548A https://cdn-shop.adafruit.com/datasheets/tca9548a.pdf

►ADA2717 Module multiplexeur I2C basé sur un TCA9548A
https://www.gotronic.fr/art-module-multiplexeur-i2c-ada2717-26051.htm

Information de chez Adafruit
https://learn.adafruit.com/adafruit-tca9548a-1-to-8-i2c-multiplexer-breakout/overview

►DFR0576 Module multiplexeur I2C basé sur un TCA9548A
https://www.gotronic.fr/art-module-multiplexeur-i2c-dfr0576-28706.htm

Fiche technique https://wiki.dfrobot.com/Gravity__Digital_1-to-8_I2C_Multiplexer_SKU_DFR0576

Librairie Arduino https://github.com/DFRobot/DFRobot_I2C_Multiplexer

► Le module générique TCA9548A Global Shuntong Store /

Module multiplexeur basé sur un TCA9548A

permettant de raccorder jusqu'à 8 modules I2C avec la même adresse sur le même bus I2C d'un microcontrôleur.

Il est possible de raccorder 8 multiplexeurs sur un seul microcontrôleur grâce à des adresses sélectionnables via pontets à souder ou via des niveaux logiques sur les broches dédiées. Un guide d'utilisation en anglais est disponible en fiche technique.

Remarque: ce module est livré avec deux connecteurs latéraux à souder soi-même pour une utilisation sur une plaque de montage rapide.

Caractéristiques: Alimentation: 3 à 5 Vcc Interface: I2C Adresses I2C sélectionnables: 0x70 à 0x77 Dimensions: 31 x 18 x 3 mm Poids: 1,8 g Référence Adafruit: ADA2717

Son utilisation est assez simple : le multiplexeur lui-même est sur l'adresse I2C 0x70 (mais peut être ajusté de 0x70 à 0x77) et vous écrivez simplement un seul octet avec le numéro de sortie multiplexé souhaité sur ce port, et bam - tous les futurs paquets I2C. sera envoyé à ce port. En théorie, vous pourriez avoir 8 de ces multiplexeurs sur chacune des adresses 0x70-0x77 afin de contrôler 64 de la même partie adressée I2C.

La puce elle-même est compatible 1,8 V - 5 V, vous pouvez donc l'utiliser avec n'importe quel niveau logique.

Broches d'alimentation :

- Vin - c'est la broche d'alimentation. Puisque la puce du capteur utilise 3 à 5 VDC. Pour alimenter la carte, donnez-lui la même puissance que le niveau logique de votre microcontrôleur - par exemple pour un micro 5V comme Arduino, utilisez 5V
- GND - terrain d'entente pour la puissance et la logique

Broches côté contrôle I2C :

- SCL - il s'agit de la broche d'horloge I2C pour la puce elle-même, connectez-vous à la ligne d'horloge I2C de vos microcontrôleurs.
- SDA - il s'agit de la broche de données I2C pour la puce elle-même, connectez-vous à la ligne de données I2C de votre microcontrôleur.
- RST - c'est la broche de réinitialisation, pour réinitialiser la puce du multiplexeur. Tiré haut par défaut, connectez-vous à la terre pour réinitialiser
- A0 A1 A2 - ce sont les broches de sélection d'adresse pour le multiplexeur . Par défaut, le multiplexeur est à l'adresse 0x70 et ces trois broches sont tirées vers le bas. Connectez-les à Vin pour définir l'adresse sur 0x71 - 0x77 .
- A0 est le bit de poids le plus faible (s'il est élevé, il augmentera l'adresse de 1).
- A1 est le deuxième bit de poids le plus faible (s'il est élevé, il augmentera l'adresse de 2).
- A2 est le troisième bit de poids le plus faible (s'il est élevé, il augmentera l'adresse de 4).

Broches côté multiplexé I2C :

- SDx et SCx : Il existe 8 jeux de broches SDx et SCx , de SD0/SC0 à SD7/SC7 . Ce sont les broches multiplexées. Chacun est un ensemble de bus I2C complètement séparé. Vous pouvez donc avoir 8 appareils I2C avec des adresses identiques, à condition qu'ils soient chacun sur un bus I2C.
- Aucun pullup n'est installé sur ces broches, donc si vous utilisez une puce ou un breakout sans pullups i2c, assurez-vous de les ajouter ! Eh bien, vous pouvez avoir Vin à 3,3 V et faire tirer ces broches jusqu'à 5 V (c'est-à-dire qu'elles sont conformes à 5 V)

Câblage et test Arduino

Le multiplexeur TCA9548A est intéressant dans la mesure où il possède une adresse I2C (0x70 par défaut) - et vous lui envoyez essentiellement une commande pour lui indiquer à quelle sortie multiplexée I2C vous souhaitez parler, vous pouvez ensuite adresser la carte à laquelle vous souhaitez adresser.

Nous vous suggérons d'utiliser ce petit assistant pour vous aider à sélectionner le port

prog001.ino

```
#définir TCAADDR 0x70
void tcaselect(uint8_t i) {
if (i > 7) return;
Wire.beginTransmission(TCAADDR);
Wire.write(1 << i);
Wire.endTransmission();
}
```

Vous pouvez ensuite appeler tcaselect(0) via tcaselect(7) pour configurer le multiplexeur.

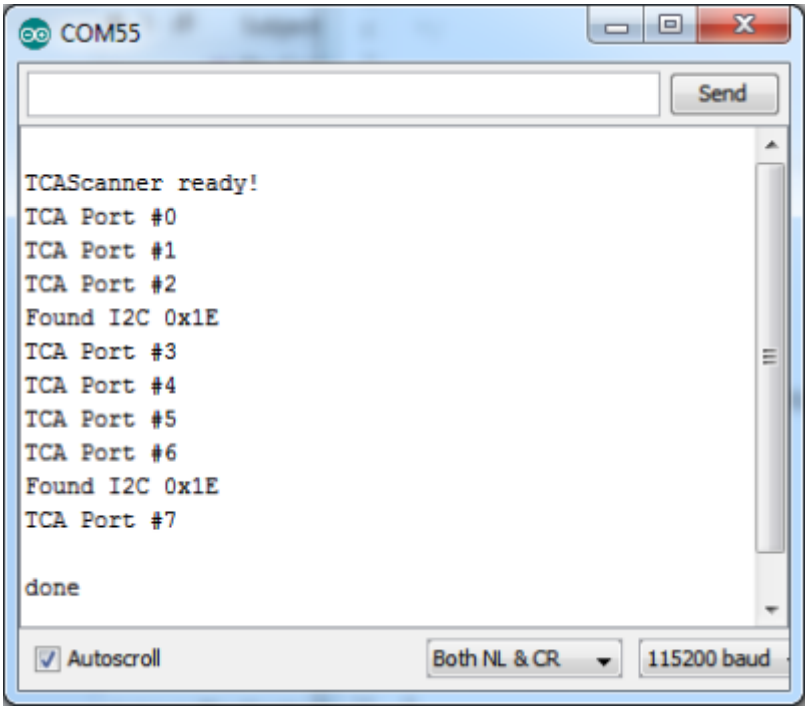
Notez que si vous possédez des appareils I2C avec l'adresse I2C 0x70, vous devrez court-circuiter l'une des broches Addr de la dérivation TCA9548 vers Vin afin d'éviter tout conflit. Étant donné que vous pouvez avoir 0x70 à 0x77,

progr002.ino

```
/**
 * TCA9548 I2CScanner.ino -- I2C bus scanner for Arduino
 *
 * Based on https://playground.arduino.cc/Main/I2cScanner/
 *
 */

#include "Wire.h"

#define TCAADDR 0x70

void tcaselect(uint8_t i) {
  if (i > 7) return;

  Wire.beginTransmission(TCAADDR);
  Wire.write(1 << i);
  Wire.endTransmission();
}


// standard Arduino setup()
void setup()
```

```
{
    while (!Serial);
    delay(1000);

    Wire.begin();

    Serial.begin(115200);
    Serial.println("\nTCAScanner ready!");

    for (uint8_t t=0; t<8; t++) {
      tcaselect(t);
      Serial.print("TCA Port #"); Serial.println(t);

      for (uint8_t addr = 0; addr<=127; addr++) {
        if (addr == TCAADDR) continue;

        Wire.beginTransmission(addr);
        if (!Wire.endTransmission()) {
          Serial.print("Found I2C 0x");  Serial.println(addr,HEX);
        }
      }
    }
    Serial.println("\ndone");
}

void loop()
{
}
```

Par exemple, l'exécuter sur la configuration ci-dessus vous donnera :



prog003.ino

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_HMC5883_U.h>

#define TCAADDR 0x70

/* Assign a unique ID to this sensor at the same time */
Adafruit_HMC5883_Unified mag1 = Adafruit_HMC5883_Unified(1);
Adafruit_HMC5883_Unified mag2 = Adafruit_HMC5883_Unified(2);

void displaySensorDetails(Adafruit_HMC5883_Unified *mag)
{
  sensor_t sensor;
  mag->getSensor(&sensor);
  Serial.println("------------------------------------");
  Serial.print  ("Sensor:       "); Serial.println(sensor.name);
  Serial.print  ("Driver Ver:   "); Serial.println(sensor.version);
  Serial.print  ("Unique ID:    "); Serial.println(sensor.sensor_id);
  Serial.print  ("Max Value:    "); Serial.print(sensor.max_value);
Serial.println(" uT");
  Serial.print  ("Min Value:    "); Serial.print(sensor.min_value);
Serial.println(" uT");
  Serial.print  ("Resolution:   "); Serial.print(sensor.resolution);
Serial.println(" uT");
  Serial.println("------------------------------------");
  Serial.println("");
  delay(500);
}

void tcaselect(uint8_t i) {
```

```
  if (i > 7) return;

  Wire.beginTransmission(TCAADDR);
  Wire.write(1 << i);
  Wire.endTransmission();
}


void setup(void)
{
  Serial.begin(9600);
  Serial.println("HMC5883 Magnetometer Test"); Serial.println("");

  /* Initialise the 1st sensor */
  tcaselect(2);
  if(!mag1.begin())
  {
    /* There was a problem detecting the HMC5883 ... check your
connections */
    Serial.println("Ooops, no HMC5883 detected ... Check your
wiring!");
    while(1);
  }

  /* Initialise the 2nd sensor */
  tcaselect(6);
  if(!mag2.begin())
  {
    /* There was a problem detecting the HMC5883 ... check your
connections */
    Serial.println("Ooops, no HMC5883 detected ... Check your
wiring!");
    while(1);
  }

  /* Display some basic information on this sensor */
  tcaselect(2);
  displaySensorDetails(&mag1);
  tcaselect(6);
  displaySensorDetails(&mag2);
}

void loop(void)
{
  /* Get a new sensor event */
  sensors_event_t event;

  tcaselect(2);
  mag1.getEvent(&event);
```

```
  /* Display the results (magnetic vector values are in micro-Tesla
(uT)) */
  Serial.print("Sensor #1 - ");
  Serial.print("X: "); Serial.print(event.magnetic.x); Serial.print("
");
  Serial.print("Y: "); Serial.print(event.magnetic.y); Serial.print("
");
  Serial.print("Z: "); Serial.print(event.magnetic.z); Serial.print("
");Serial.println("uT");

  tcaselect(6);
  mag2.getEvent(&event);
  /* Display the results (magnetic vector values are in micro-Tesla
(uT)) */
  Serial.print("Sensor #2 - ");
  Serial.print("X: "); Serial.print(event.magnetic.x); Serial.print("
");
  Serial.print("Y: "); Serial.print(event.magnetic.y); Serial.print("
");
  Serial.print("Z: "); Serial.print(event.magnetic.z); Serial.print("
");Serial.println("uT");

  delay(500);
}
```

From:
https://magenealogie.chanterie37.fr/www/fablab37110/ - **Castel'Lab le Fablab MJC de Château-Renault**

Permanent link:
https://magenealogie.chanterie37.fr/www/fablab37110/doku.php?id=start:arduino:i2c:tca9548a&rev=1713180191

Last update: **2024/04/15 13:23**