Utilisation du multiplexeur TCA9548A 1 à 8 I2C

Problème posé

De temps en temps, vous devrez peut-être utiliser plusieurs appareils avec la même adresse de bus I2C avec votre Arduino.
Comme nos mini écrans OLED pour surveiller vos cantons et la position des aiguillages autrement que par des leds :

Piste intéressante, mais il y a un souci : tous ces écrans auront la même adresse 0x3C sur le bus I2C et il ne sera pas possible de leur envoyer l'information propre au canton qu'ils renseignent.

Ce problème peut être résolu avec le multiplexeur TCA9548A 1 à 8 I2C, et je vais vous expliquer comment y parvenir facilement.

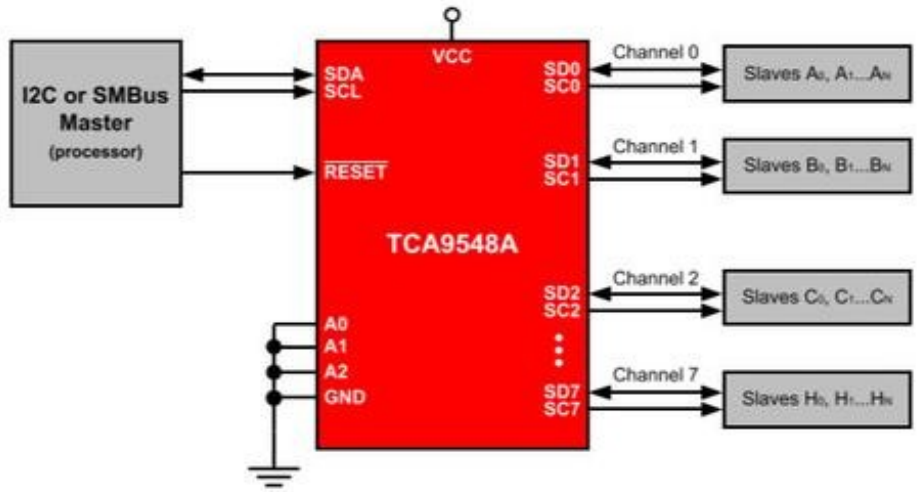
Pour acheter des TCA9548A, c'est chez Aliexpress : lien pour acheter

Ce module permet d'adresser 8 afficheurs et vous pouvez en avoir plusieurs si vous avez plus de 8 afficheurs (ou capteurs) à adresser.

Pour commencer

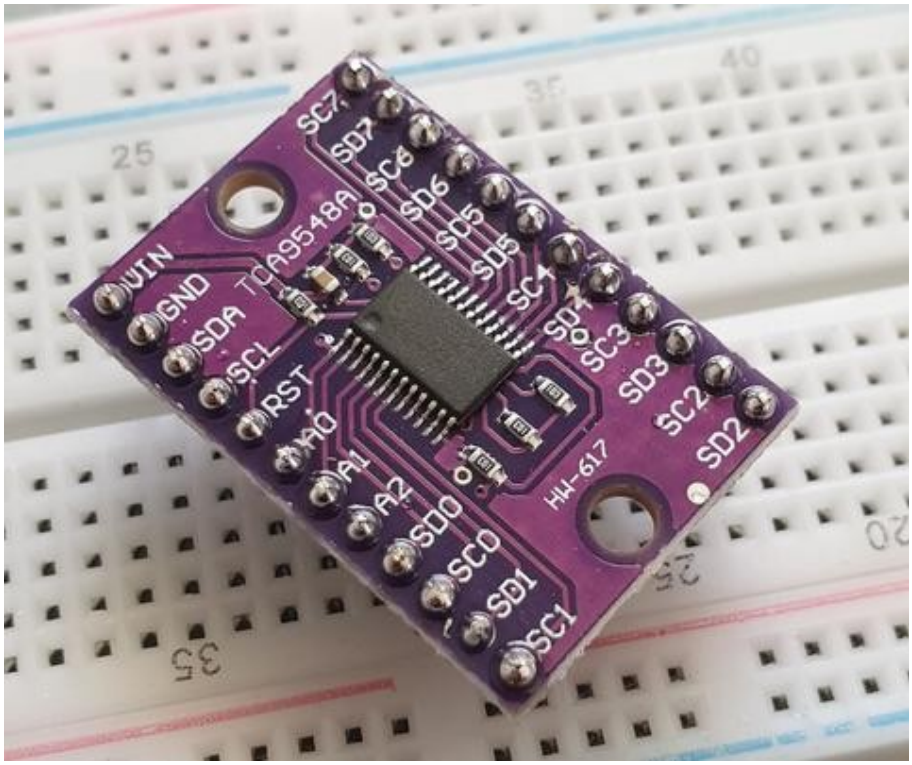
Tout d'abord, considérons le TCA9548A lui-même. C'est la passerelle entre votre Arduino et huit bus I2C distincts. Vous avez un seul bus d'un côté, connecté à votre

Arduino.
De l'autre côté du TCA9548A, vous avez huit bus I2C, et un seul d'entre eux peut être connecté à l'Arduino à la fois. Par exemple (à partir de la fiche technique) :

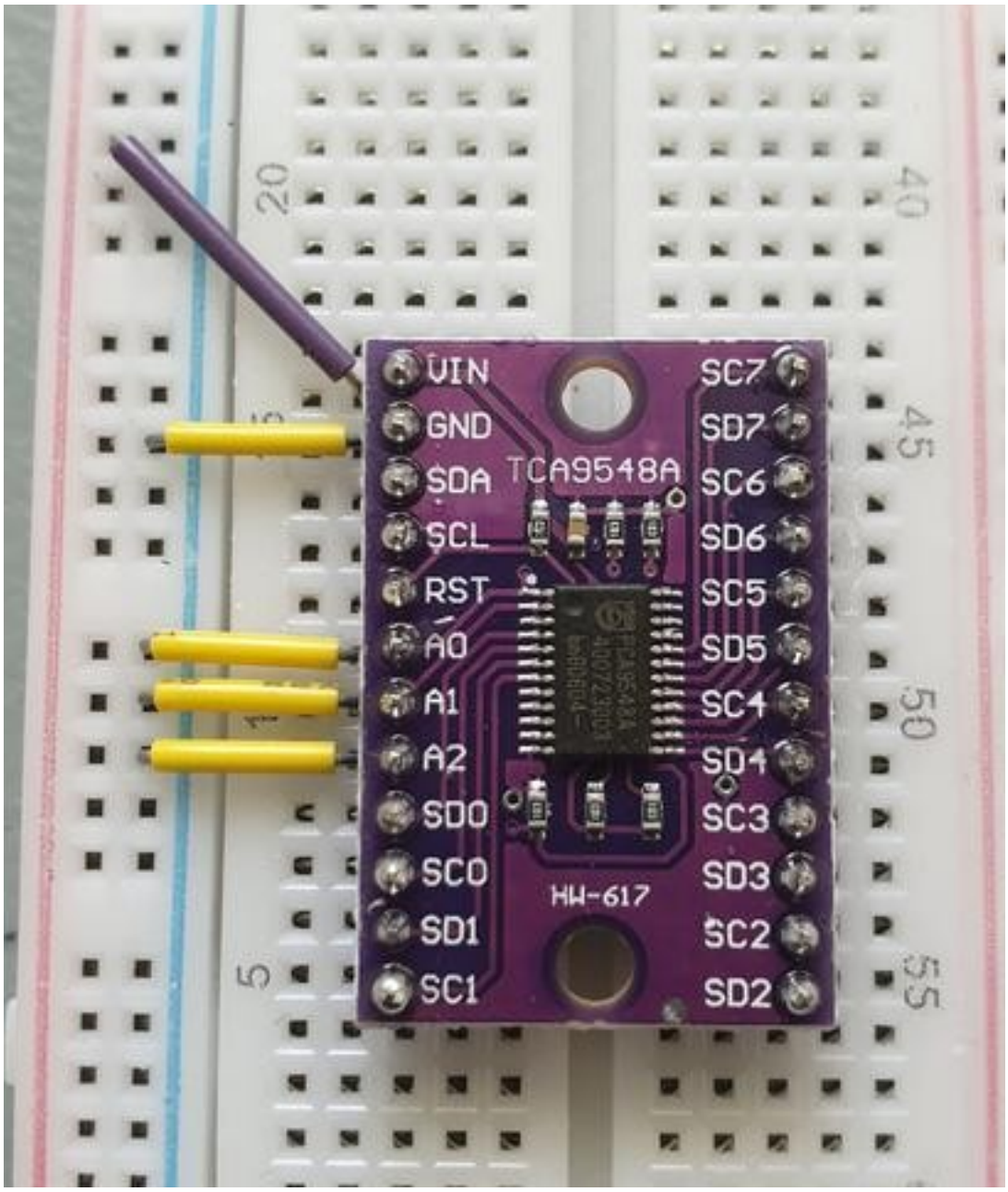


Le TCA9548 peut fonctionner sur des tensions comprises entre 1,8 et 5 V CC et fonctionner avec des appareils dont les tensions de fonctionnement sont comprises entre 1,8 et 5 V CC. C'est très pratique, car (par exemple) vous pouvez utiliser des appareils conçus pour un fonctionnement en 3,3 V avec des Arduinos 5 V, ou vice versa.

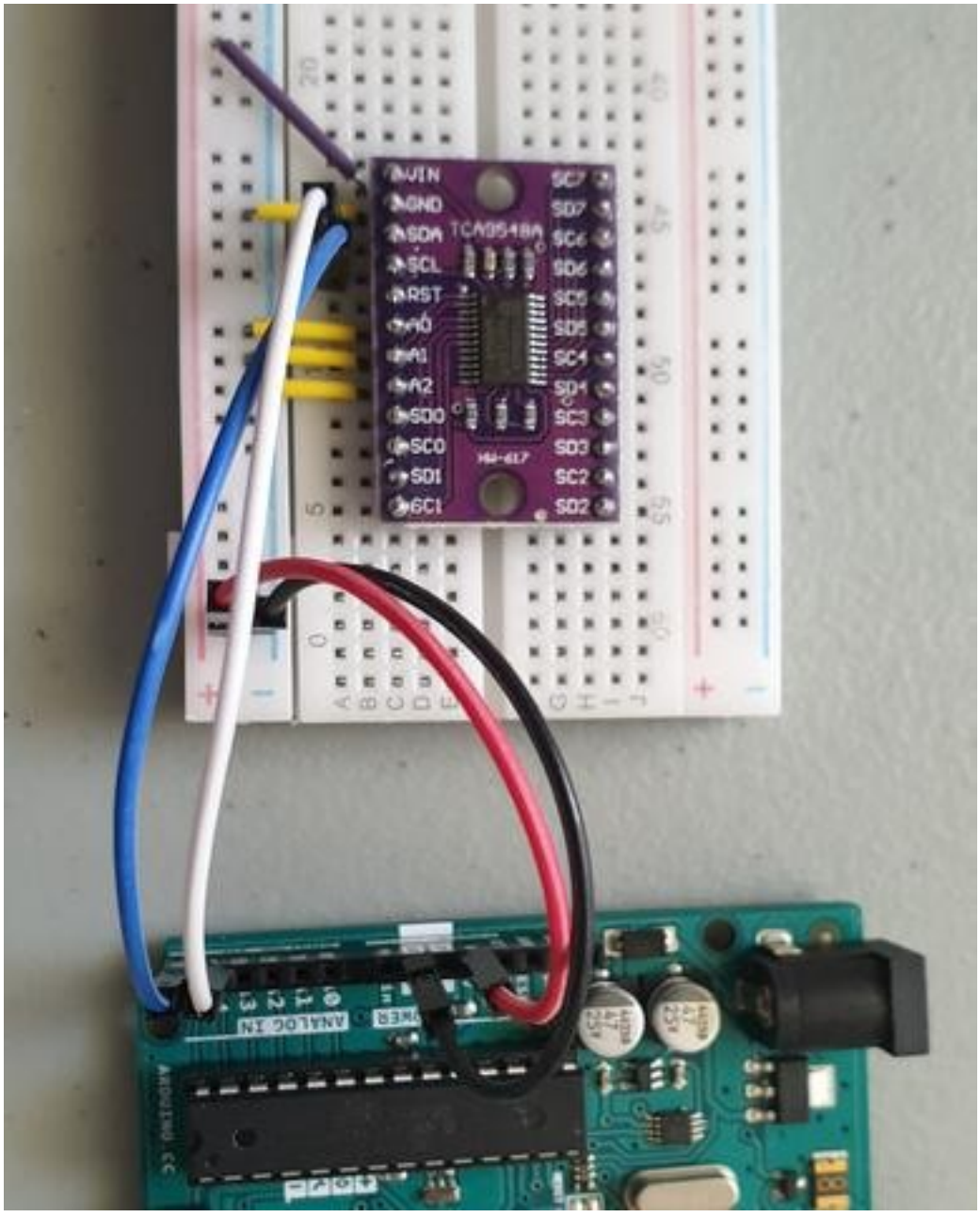
La carte de dérivation comprend des broches d'en-tête en ligne, qui ne sont pas soudées à la carte. Ce sera votre premier travail. Un moyen simple d'aligner correctement les broches consiste à le positionner sur une plaque d'essai sans soudure pour maintenir les picots droits.



Ensuite, insérez votre module sur une plaque d'essai et commencer à la câbler.



Nous utilisons les bandes verticales rouges et bleues sur la plaque d'essai comme 5V et GND respectivement. Enfin, nous connectons le 5V et le GND de l'Arduino à la platine d'expérimentation, et A4/A5 au SDA/SCL (dans le cas de l'Arduino UNO).



Les connexions électriques sont les suivantes (Module vers Arduino UNO) :

Vin à 5V
GND à GND
A0 à GND
A1 à GND
A2 à GND
SDA vers A4
SCL à A5
Ensuite, nous considérons l'adresse de bus I2C pour le TCA9548A. En utilisant la configuration de câblage indiquée ci-dessus, l'adresse est définie sur 0x70. Vous n'avez besoin de modifier cela que si l'un de vos autres appareils a également une adresse de 0x70, comme indiqué à l'étape suivante.

Modification de l'adresse I2C du TCA9548A

L'adresse de bus du TCA9548A est modifiée à l'aide des connexions aux broches A0, A1 et A2. Par défaut dans le tutoriel, nous utilisons 0x70, en connectant A0~A2 à GND (appelé LOW). En utilisant le tableau ci-dessous, vous pouvez reconfigurer à une adresse entre 0x70 et 0x77 en faisant correspondre les entrées à HIGH (5V) ou LOW (GND) :

INPUTS			I^2C BUS SLAVE ADDRESS
A2	A1	A0	
L	L	L	112 (decimal), 70 (hexadecimal)
L	L	H	113 (decimal), 71 (hexadecimal)
L	H	L	114 (decimal), 72 (hexadecimal)
L	H	H	115 (decimal), 73 (hexadecimal)
H	L	L	116 (decimal), 74 (hexadecimal)
H	L	H	117 (decimal), 75 (hexadecimal)
H	H	L	118 (decimal), 76 (hexadecimal)
H	H	H	119 (decimal), 77 (hexadecimal)

Essai du montage

Avant que nous ne soyons trop excités, le moment est venu de tester notre câblage pour nous assurer que l'Arduino peut communiquer avec le TCA9548A. Nous allons le

faire en exécutant un scan du bus I2C, qui renvoie l'adresse de bus d'un périphérique connecté.

Copiez et collez ce petit programme (il n'est pas de moi et il y en plein d'autres sur le web) dans votre IDE Arduino et téléchargez-le sur votre carte.

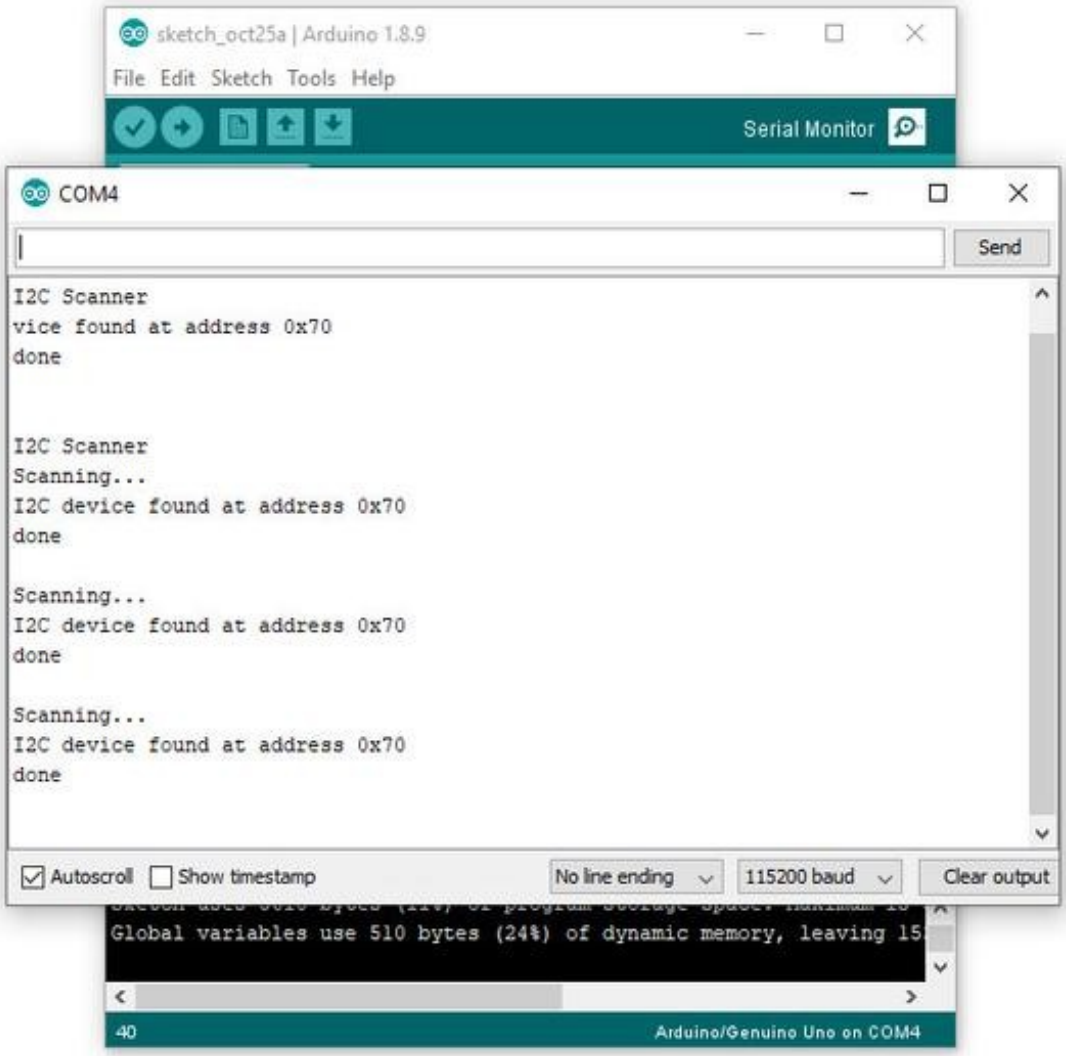
```
#include <Wire.h>

void setup() {
  Wire.begin();
  Serial.begin(115200);
  Serial.println("\nI2C Scanner");
}

void loop() {
  byte error, address;
  int nDevices;
  Serial.println("Scanning...");
  nDevices = 0;
  for(address = 1; address < 127; address++ ) {
    Wire.beginTransmission(address);
    error = Wire.endTransmission();
    if (error == 0) {
      Serial.print("I2C device found at address 0x");
      if (address<16) {
        Serial.print("0");
      }
      Serial.println(address,HEX);
      nDevices++;
    }
    else if (error==4) {
      Serial.print("Unknow error at address 0x");
      if (address<16) {
        Serial.print("0");
      }
      Serial.println(address,HEX);
    }
  }
  if (nDevices == 0) {
    Serial.println("No I2C devices found\n");
  }
```

```
  else {
    Serial.println("done\n");
  }
  delay(5000);
}
```

Ensuite, ouvrez le moniteur série et réglez le débit de données sur 115200. Vous devriez obtenir quelque chose comme ceci:



Comme vous pouvez le voir, notre scanner a renvoyé une adresse de 0x70, qui correspond au câblage décrit dans le tableau d'adresses de bus mentionné précédemment. Si vous n'avez pas réussi, débranchez l'Arduino de l'ordinateur et vérifiez votre câblage, puis réessayez.

Contrôle du sélecteur de bus

L'utilisation du TCA9548A est que votre croquis n'est pas du tout complexe, il ne nécessite qu'une seule étape avant d'utiliser votre appareil I2C normalement. Cette étape supplémentaire consiste à demander au TCA9548A d'utiliser l'un des huit bus qu'il contrôle.

Pour ce faire, nous envoyons un octet de données au registre de bus du TCA9548A qui représente lequel des huit bus nous voulons utiliser. Chaque bit de l'octet est utilisé pour activer ou désactiver le bus, avec le MSB (bit le plus significatif) pour le bus 7 et le LSB (bit le moins significatif) pour le bus 0.

Par exemple, si vous avez envoyé :

0b00000001 (en binaire) ou 0 en décimal
cela activerait le bus zéro.

Ou si vous avez envoyé :

0b00010000 (en binaire)
cela activerait le bus cinq.

Une fois que vous avez sélectionné un bus, le TCA9548A canalise toutes les données entrantes et sortantes du bus vers l'Arduino sur le bus sélectionné. Vous n'avez besoin d'envoyer les données de sélection de bus que lorsque vous souhaitez changer de bus. Nous le démontrerons plus tard.

Alors pour se simplifier la vie, on peut utiliser une petite fonction « particulière » (les habitués de l'Arduino comprendront) pour sélectionner facilement le bus souhaité (et du coup l'afficheur qu'on veut renseigner) :

```
void TCA9548A(uint8_t bus)
{
  Wire.beginTransmission(0x70);  // TCA9548A address is 0x70
  Wire.write(1 << bus);          // send byte to select bus
  Wire.endTransmission();
}
```

Cette fonction accepte un numéro de bus et place un « 1 » dans le registre de bus du TCA9548A correspondant à nos exigences. Ensuite, il vous suffit de glisser cette fonction juste avant d'avoir besoin d'accéder à un périphérique sur un bus I2C particulier. Par exemple, un équipement sur le bus 0 :

TCA9548A(0) ;
… ou un appareil sur le bus 6 :

TCA9548A(6) ;

Une note rapide sur les résistances pull-up

Vous devez toujours utiliser des résistances de rappel sur les huit bus I2C provenant du TCA9548A. Si vous utilisez un module assemblé, comme nos exemples d'appareils, ils auront les résistances, alors ce ne sera pas utile.

Sinon, consultez les fiches techniques de vos appareils pour déterminer la valeur appropriée des résistances de rappel. Si cette information n'est pas disponible, essayez des résistances 10k0.

Contrôler votre premier afficheur

Notre premier exemple d'appareil est le petit écran OLED de 0,96". Il dispose de quatre connexions, qui sont câblées comme suit (OLED — TCA9548A/Arduino) :

GND à GND
Vcc vers Arduino 3.3V
CL à TCA9548A SC0 (bus #0, broche d'horloge)
DA à TCA9548A SD1 (bus #0, broche de données)
L'OLED fonctionne à partir de 3,3 V, c'est pourquoi nous l'alimentons directement à partir de la broche 3,3 V de l'Arduino.

Maintenant, copiez et téléchargez ce programme sur votre Arduino, et après un moment, l'OLED affichera des nombres décomptés en différents valeurs :

Note : ce programme donné à titre d'exemple devra être modifié pour les écrans OLED que vous utiliserez.

```
// Affichage - https://pmdway.com/collections/oled-displays/products/0-49-64-x-32-white-graphic-oled-i2c
// Guide - https://pmdway.com/blogs/product-guides-for-arduino/tutorial-using-the-0-49-64-x-32-graphic-i2c-oled-display-with-arduino
// TCA9548A - https://pmdway.com/blogs/product-guides-for-arduino/using-the-tca9548a-1-to-8-i2c-multiplexer-breakout-with-arduino

#include <Arduino.h>
#include <U8g2lib.h> // modifier cette ligne en fonction du modèle OLED
#include <Fil.h>

annuler TCA9548A (bus uint8_t)
{
  Wire.beginTransmission (0x70); // L'adresse TCA9548A est 0x70
  Wire.write(1 << bus); // envoyer l'octet au bus sélectionné
  Wire.endTransmission();
}

U8G2_SSD1306_64X32_1F_F_HW_I2C u8g2(U8G2_R0, /* reset=*/ U8X8_PIN_NONE)

void setup()
{
  Fil.begin();
  u8g2.begin();
}

boucle vide()
{
   TCA9548A(0) ; // dire au TCA9548A que nous voulons utiliser le bus I2C numéro zéro (pour parler à l'OLED)

  // utilise l'OLED normalement
  pour (int a = 999; a >= 0; --a)
  {
    u8g2.clearBuffer(); // efface la mémoire interne
    u8g2.setFont(u8g2_font_inb24_mr ); // choisissez une police appropriée
    u8g2.setCursor(0, 24);
    u8g2.print(a);
    a = a - 47 ;
    u8g2.sendBuffer(); // transfert de la mémoire interne vers l'écran
```

```
    retard (100);
  }
}
```

Alors comment ça a marché ? Nous avons inséré la fonction de sélection de bus à la ligne 9 du croquis, puis appelé la fonction à la ligne 26 pour indiquer au TCA9548A que nous voulions utiliser le bus I2C zéro. Ensuite, le reste du croquis a utilisé l'OLED comme d'habitude.

Contrôler deux appareils

Ajoutons un autre afficheur, nous allons le connecter au bus I2C numéro sept sur le TCA5948A. Il y a quatre connexions, qui sont câblées comme suit (afficheur — TCA9548A/Arduino) :

GND à GND
Vcc vers Arduino 3.3V
CL à TCA9548A SC0 (bus #7, broche d'horloge)
DA à TCA9548A SD1 (bus #7, broche de données)