SSD1306 affichage OLED i2c _ C Arduino _ version 2022

petit écran Oled, type SSD1306, largeur 128c x longueur 64c, i2c, noir et blanc permettant d'afficher des textes, des images..

Évènements

- FabLab Crepp : 24/01/2023
- FabLab Crepp : 31/01/2023
- CAO-3D Atelier : 04/02/2023
- FabLab Crepp : 07/02/2023
- Arduino Atelier : 11/02/2023

EcoWatt

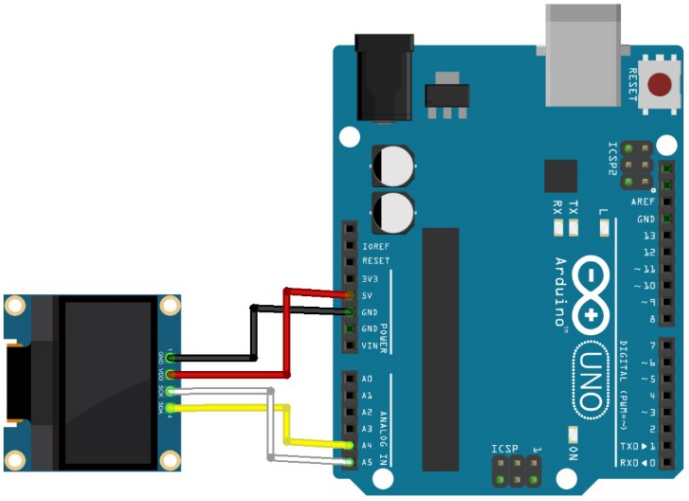
On peut utliser l'Oled soit en mode **SPI**, soit en mode I2C.

nous utiliserons le mode I2C et cela complètera la partie micropython vue en novembre 2018.

4 broches:

- GND
- VCC 3,3v / 5v
- SCL –> A5 communication i2c
- SDA–> A4 communication i2c

- Nano: SDA (A4); SCL (A5);
- MEGA: SDA (20); SCL (21);
- Leonardo: SDA (20); SCL (21);

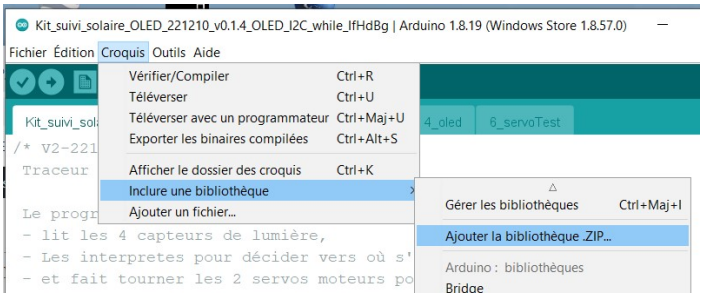


plusieurs bibliothèques sont disponibles pour utiliser l'oled SSD1306 dont adafruit_SSD1306.h.

Pour changer, j'utiliserai la bibliothèque de rinky-dink-electronics: OLED_I2C.zip bien documentée et simple d'utilisation:

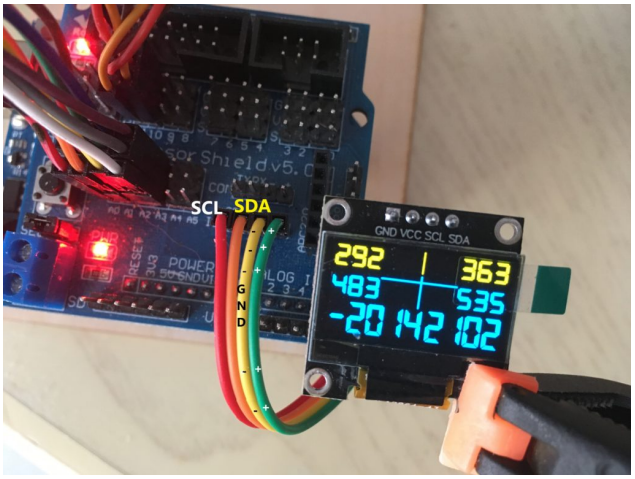
Pour installer la bibliothèque **OLED_I2C.zip** dans l'IDE Arduino:

Articles récents



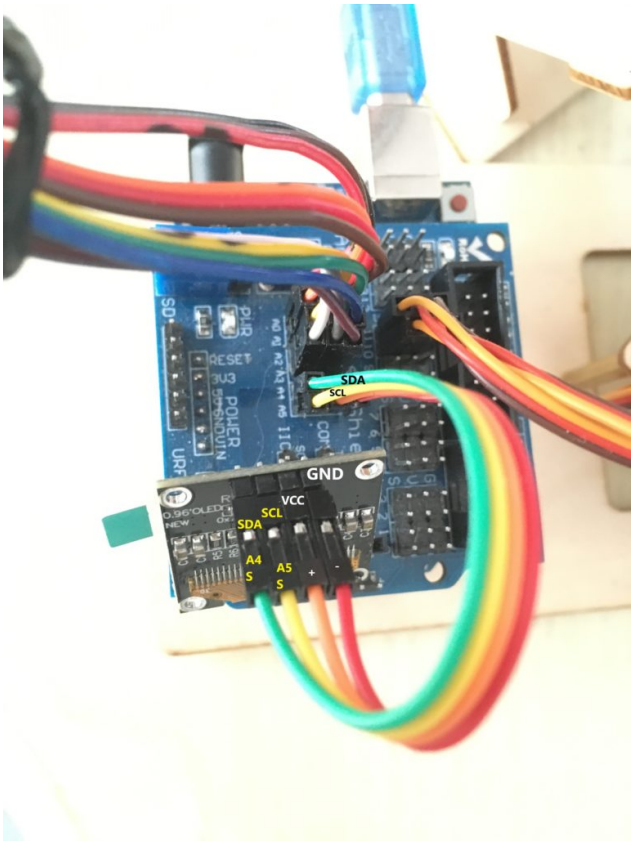
Il y a 3 possibilités de cablage de l'oled sur le shield Arduino mais dans les 3 cas **SDA–>A4 et SCL–>A5**:

1. utilisation des broches I2C du ShieldAduino

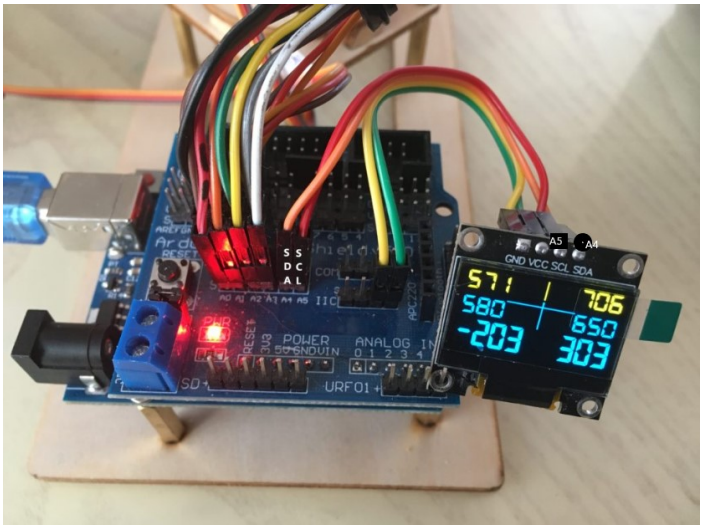


2. ou on complète les broches A4 et A5 du shield Arduino

Agenda



3. enfin, un mixte montrant l'utilisation des broches A4 et A5



dans le programme suivi solaire nous rajouterons:

1. **avant la partie setup,** pour définir l'environnement Oled et la création de l'objet myOLED; il n'y a aucune définition des broches A4 et A5 pour l'utilisation de l'I2C.

```
// ----------ppr début --------------
// écran OLED
```

```
// ---------------------------------
//
#include <OLED_I2C.h>
OLED  myOLED(SDA, SCL, 8);
extern uint8_t SmallFont[];
extern uint8_t MediumNumbers[];
extern uint8_t BigNumbers[];
// ----------ppr fin   -------------
//
```

2. **dans la partie setup**, démarrage de l'écran OLED

```
//
----------ppr début -----------------
// écran OLED démarrage
// ---------------------------------
//
myOLED.begin();
myOLED.setFont(SmallFont);
// ----------ppr fin ---------------
//
```

3. **enfin dans la partie loop,** après *// Affichage les consignes sur le moniteur série*, j'insère le code opérationnel affichant les informations sur l'écran Oled

```
  // ----------ppr début ------------
  // écran OLED affichage
  // ------------------------------
  //myOLED.drawLine(0,0,127,63);
  // Draw a line from the upper left
  // to the lower right corner
    myOLED.drawLine(0,20,127,20);  //
    myOLED.drawLine(65,0,65,40);   //
    myOLED.setFont(MediumNumbers);

    myOLED.printNumI(HG, LEFT, 0);
    myOLED.printNumI(HD, RIGHT, 0);
    myOLED.printNumI(BG, LEFT, 20);
    myOLED.printNumI(BD, RIGHT, 20);

    myOLED.setFont(BigNumbers);
    myOLED.printNumI(consigneInclinai
    myOLED.printNumI(consigneRotatior
    myOLED.update();
  // ----------ppr fin -------------
  //
```

Attention: pour les besoins du test manuel de l'écran Oled,
j'ai dévalidé les 2 servomoteurs en les reliant à des broches non connnectées 8 et 9:

```
int brocheServoRotation = 8; // 10
```

opérationnel

int brocheServoInclinaison = 9; // 11

opérationnel

Pour refaire fonctionner l'Arduino solaire avec affichage Oled, il faut redéfinir les broches des 2 servomoteurs:

int brocheServoRotation = 10; // 10

opérationnel, 8 test Oled

int brocheServoInclinaison = 11; // 11

opérationnel, 9 test Oled

ce qui donne au final:

Kit_suivi_solaire_OLED_221210_v0.1.2_OLED_I2Czip

```
/* V2-221118 bis: ajout OLED ppr
 Traceur solaire 4 capteurs de lumière.

 Le programme:
 - lit les 4 capteurs de lumière.
 - Les interpretes pour décider vers où s'ori
 - et fait tourner les 2 servos moteurs pour

 Créé le 1 Sep 2022, par Xavier Lesot, basé s
 Modifié le 4 Nov 2022, par Xavier Lesot : A
 Modifié le 18 Nov 2022, par Xavier Lesot : A
 Modifié le 2 Déc 2022, pas Patrick Pastor: a
*/

// ----------------------------------------
// Câblage des broches
// Connecte la broche physique avec sa référe
// ----------------------------------------
// Entrées
int brocheHG = A0;
int brocheHD = A1;
int brocheBG = A2;
int brocheBD = A3;
int brocheJovX = A4;
int brocheJoyY = A5;

// Sorties
//* ****** PPR: test OLED: on désactive
//*  les 2 servomoteurs en débranchant
//*  10 --> 8.  11--> 9
int brocheServoRotation = 8;     // 10 opérat
int brocheServoInclinaison = 9;  // 11 opérat
int brocheLed = 13;

// ----------------------------------------
// Créer des Variables
```

```
// Définit des cases dans la mémoire RAM pour
// int = 16 bits = -32.768 to 32.767
// ----------------------------------------
int HG; // stockera la valeur du capteur Haut
int HD; // stockera la valeur du capteur Droi
int BG; // stockera la valeur du capteur Bas
int BD; // stockera la valeur du capteur Bas
int JoyX; // Option contrôle par joystick
int JoyY;

// Réglages paramétrable:
int frequenceActualisation = 25; // Réglage d
int consigneRotation = 90; // Position initia
int consigneInclinaison = 10; // Position ini
bool modeJoystick = false; // false: Mode cap

// limites logiciels
// Butées logiciels pour empêcher le mecanism
// Ces réglages doivent être adjustés lorsque
int LimiteRotationMax = 180;
int LimiteRotationMin = 0;
int LimiteInclinaisonMax = 180;  // 100 ppr v
int LimiteInclinaisonMin = 10;

// ----------------------------------------
// Servo
// ----------------------------------------
// créer un objet servo pour contrôler un ser
// douze objets servo peuvent être créés sur
#include <Servo.h> // La bibliotèque d'instruc
Servo servoRotation; // crée un objet servo r
Servo servoInclinaison; // crée un objet serv

// ----------ppr début ----------------------
// écran OLED
// ----------------------------------------
//
#include <OLED_I2C.h>
OLED  myOLED(SDA, SCL, 8);
extern uint8_t SmallFont[];
extern uint8_t MediumNumbers[];
extern uint8_t BigNumbers[];
// ----------ppr fin   ----------------------
//


// ========================================
// Setup
// ========================================
void setup() { // mettez votre code d'install

  // Servo
  servoRotation.attach(brocheServoRotation);
  servoInclinaison.attach(brocheServoInclinai

  // broche à configurer en sortie
  pinMode(brocheLed, OUTPUT); // dit à l'ardu

  // Liaison Série
  // Démarre la communication sur la liaison
  // (bauds: bit per second: bps. Représente
  // (9600 bauds: +-1ko/s. Durée d'un bit: 1.
  Serial.begin(115200); // Démarre la liaison
  Serial.println("Hello World !"); // « Hello

  // ----------ppr début ----------------------
  // écran OLED démarrage
```

```
  // ----------------------------------------
  //
  myOLED.begin();
  myOLED.setFont(SmallFont);
  // ----------ppr fin   --------------------
  //

}

// ========================================
// Loop
// ========================================
void loop() { //mettez votre code principal i

  // ----------------------------------------
  // Lecture des entrées analogiques.
  // ----------------------------------------
  // Lectures des valeurs des photoresistance
  HG = analogRead(brocheHG); // lit l'entrée
  HD = analogRead(brocheHD);
  BG = analogRead(brocheBG);
  BD = analogRead(brocheBD);

  if (modeJoystick){ // si mode Joystick alor
    JoyX = analogRead(A4); // lit aussi les j
    JoyY = analogRead(A5);
  }

  // A propos des photorésistances:
  // https://fr.wikipedia.org/wiki/Photor%C3%
  // Exposé à la lumière affiche grands chiff
  // Caché de la lumière affiche petits chiff

  // Affichage des résultats sur le moniteur
  // (L'arduino envoi sur la liaison série. I
  Serial.println(); // Commence sur une nouve

  Serial.print("HG: "); // envoi un texte qui
  Serial.print(HG); // envoi la valeur du car
  Serial.print(", "); // envoi virgule et esp

  Serial.print("HD: ");
  Serial.print(HD);
  Serial.print(", ");

  Serial.print("BG: ");
  Serial.print(BG);
  Serial.print(", ");

  Serial.print("BD: ");
  Serial.print(BD);
  Serial.print(", ");

  if (modeJoystick){
    Serial.print("JoyX: "); Serial.print(JoyX
    Serial.print("JoyY: "); Serial.print(JoyY
  }
  // ----------------------------------------
  // Interprétation des capteurs:
  // Selon la provenance de la lumière,
  // change la consigne à l'axe du servomoteu
  // ----------------------------------------
  Serial.print("| "); // mise en forme

  // Rotation
  if (HG > HD) { // Si +plus de lumière à Gau
    consigneRotation -= 1; // incrémente 1° d
```

```
  Serial.print("R- "); // Affiche le result
}
if (BG > BD) { // Si +plus de lumière à Gau
  consigneRotation -= 1; // incrémente 1° d
  Serial.print("R- ");
}
if (HG < HD) { // Si +plus de lumière à Dro
  consigneRotation += 1; // incrémente 1° d
  Serial.print("R+ ");
}
if (BG < BD) { // Si +plus de lumière à Dro
  consigneInclinaison += 1; // incrémente i
  Serial.print(" Ih+:");
  Serial.print(HG-BG);
}
if (HD > BD){ // Si +plus de lumière en Hau
  consigneInclinaison += 1; // incrémente i
  Serial.print(" Ih+:");
  Serial.print(HD-BD);
}
if (HG < BG){ // Si +plus de lumière en Bas
  consigneInclinaison -= 1; // incrémente i
  Serial.print(" Ib-:");
  Serial.print(BG-HG);
}
if (HD < BD){  // Si +plus de lumière en Ba
  consigneInclinaison -= 1; // incrémente i
  Serial.print(" Ib-:");
  Serial.print(BD-HD);
}
Serial.print("| "); // mise en forme
// ----------------------------------------
// interprétation des capteurs terminé
// ----------------------------------------

// ----------------------------------------
// Fin de course logiciel.
// Limite les consignes minimums et maximum
// la limite max des servomoteur c'est 0...
// (modification des réglages des limites t
// ----------------------------------------


// Rotation Minimum
if (consigneRotation < LimiteRotationMin) {
  consigneRotation = LimiteRotationMin; //
}
// Rotation Maximum
if (consigneRotation > LimiteRotationMax) {
  consigneRotation = LimiteRotationMax; //
}
// Inclinaison Minimum
if (consigneInclinaison < LimiteInclinaison
  consigneInclinaison = LimiteInclinaisonMi
}
// Inclinaison Maximum
if (consigneInclinaison > LimiteInclinaison
  consigneInclinaison = LimiteInclinaisonMa
}

// ----------------------------------------
  // Joystick
  // Si bouton mode joystick alors branche
  // ----------------------------------------

if (modeJoystick){
  consigneRotation = map(JoyX, 0, 1023, 0,
```

```
    consigneInclinaison = map(JoyY, 0, 1023,
  }

  // ---------------------------------------
    // Ecriture de la consigne vers les moteu
    // ---------------------------------------
  // Affichage les consignes sur le moniteur
  Serial.print("Rotation: ");
  Serial.print(consigneRotation);
  Serial.print(", ");
  Serial.print("Inclinaison: ");
  Serial.print(consigneInclinaison);
  Serial.print(", ");


  // ----------ppr début ----------------------
  // écran OLED affichage
  // ---------------------------------------
  //myOLED.drawLine(0,0,127,63);
  // Draw a line from the upper left
  // to the lower right corner

    myOLED.drawLine(0,20,127,20);  // ligne h
    myOLED.drawLine(65,0,65,40);   // ligne v
    myOLED.setFont(MediumNumbers);

    myOLED.printNumI(HG, LEFT, 0);
    myOLED.printNumI(HD, RIGHT, 0);
    myOLED.printNumI(BG, LEFT, 20);
    myOLED.printNumI(BD, RIGHT, 20);

    myOLED.setFont(BigNumbers);
    myOLED.printNumI(consigneInclinaison, LEF
    myOLED.printNumI(consigneRotation, RIGHT,
    myOLED.update();
  // ----------ppr fin ----------------------
  //

  servoRotation.write(consigneRotation); // e
  servoInclinaison.write(consigneInclinaison)


  // fait clignoter la led à chaque actualisa
  if (digitalRead(brocheLed) == LOW){
    digitalWrite(brocheLed, HIGH);
  }else{
    digitalWrite(brocheLed, LOW);
  }

  // ---------------------------------------
  // Fréquence d'actualisation.
  // C'est le temps avant de relire la boucle
  // C'est donc le Temps avant de refaire tou
  // C'est donc la vitesse du traceur solaire
    // ---------------------------------------
  delay(frequenceActualisation); // en millis
} // Fin.
```

documentation:

OLED_I2C

Multi-platform library for I²C 128x32 and 128x64 pixel SSD1306 OLEDs

Manual

Introduction:

This library has been made to make it easy to use 128x32 and 128x64 pixel OLED displays based on the SSD1306 controller chip with an Arduino, a chipKit, an ESP8266 or an ESP32.

This library will default to I²C Fast Mode (400 KHz) when using the hardware I²C interface.

Arduino and chipKit:
The library has not been tested in combination with the Wire library and I have no idea if they can share pins. Do not send me any questions about this. If you experience problems with pin-sharing you can move the displays SDA and SCL pins to any available pins on your development board. This library will in this case fall back to a software-based, TWI-/I²C-like protocol which will require exclusive access to the pins used.

ESP8266 and ESP32:
The library makes use of the Wire library and should be able to share pins with other devices that make use of the Wire library. This has not been tested and I cannot provide any support if it does not work.
Please note that ESP8266 support has only been tested on the Wemos/Lolin D1 Mini, and ESP32 support on the Heltec WiFi Kit 32.

If you are using a chipKit Uno32 or uC32 and you want to use the hardware I²C interface you must remember to set the JP6 and JP8 jumpers to the I²C position (closest to the analog pins).

IMPORTANT *if upgrading from v1.xx of the library:*
The way the library allocates its display buffer memory has changed from v1.xx of the library. Due to this change it is highly recommended that you verify that the library was able to properly allocate the memory it needs for the display buffer.

begin() will return a boolean value indicating if it was a success or not.
You should make sure you do not use the display if the allocation fails. Allocation may fail if there isn't enough free RAM when calling begin().

The display buffer will require 512 bytes for 128x32 displays and 1024 bytes for 128x64 displays to be available. Make sure you leave enough unused RAM for the buffer in your sketch.

You can always find the latest version of the library at **http://www.RinkyDinkElectronics.com/**

For version information, please refer to **version.txt**.

Defined Literals:

Display size

For use with begin()

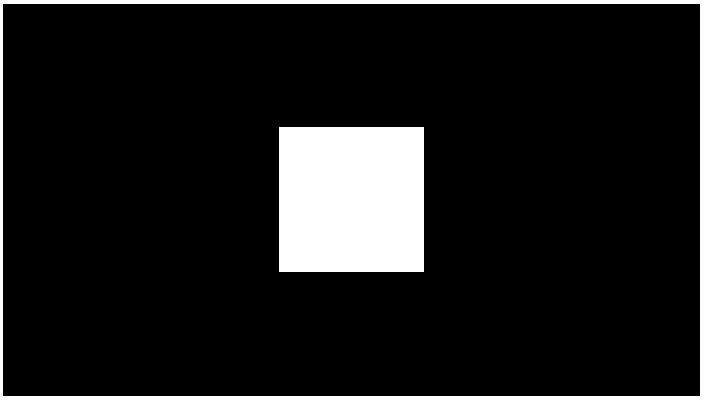
```
SSD1306_128X32:  32
SSD1306_128X64:  64
```

Alignment

et

…

c’est fini pour 2022 !



00:00 00:06