



# REALISATION D'UNE MATRICE DE LEDS

Réalisé par Fan GAO

Polytech Lille Département Informatique Microélectronique et Automatique

Tuteurs : Xavier Redon  Alexandre Boé et Thomas Vantroys

# Remerciement

Nous tenons d'abord à exprimer notre reconnaissance et nos profonds remerciements à toute personne nous ayant aidé de près ou de loin à la réalisation de ce projet à ses différentes étapes.

Nous tenons à remercier plus particulièrement monsieur Xavier Redon monsieur Alexandre Boé et monsieur Thomas Vantroys pour la confiance qu'il nous a accordée, ainsi que sa disponibilité et le temps qu'il nous a consacré tout au long du projet.

# Sommaire

# I - Présentation du projet

## *I.1 - Description*

Il fault réaliser une matrice de LED monochrome 21x21 contrôlée par un micro-contrôleur ATMega328p.

Il y a 21 lignes et 21 colonnes, soit un total de 42 ports. Nous devons contrôler ces 42 ports séparément pour obtenir un contrôle séparé de chaque LED.

Enfin, On implante sur le micro-contrôleur une lecture de chaîne de caractères, de calculer le QR code correspondant et d'afficher le QR code sur les LEDs.

Objectifs

Les objectifs de ce projet sont de réaliser une matrice de LEDs à l'aide de pilotes de LEDs.Les pilotes sont contrôlés par un micro-contrôleur ATMega328p.

On choisit le TLC5947 à contrôler 21 colonnes, il a 24 porte, dont on peut utiliser un TLC5947 à contrôler tout les colonnes. Et utiliser des PCF8574 et des UDN2982 pour contrôler chaque ligne.

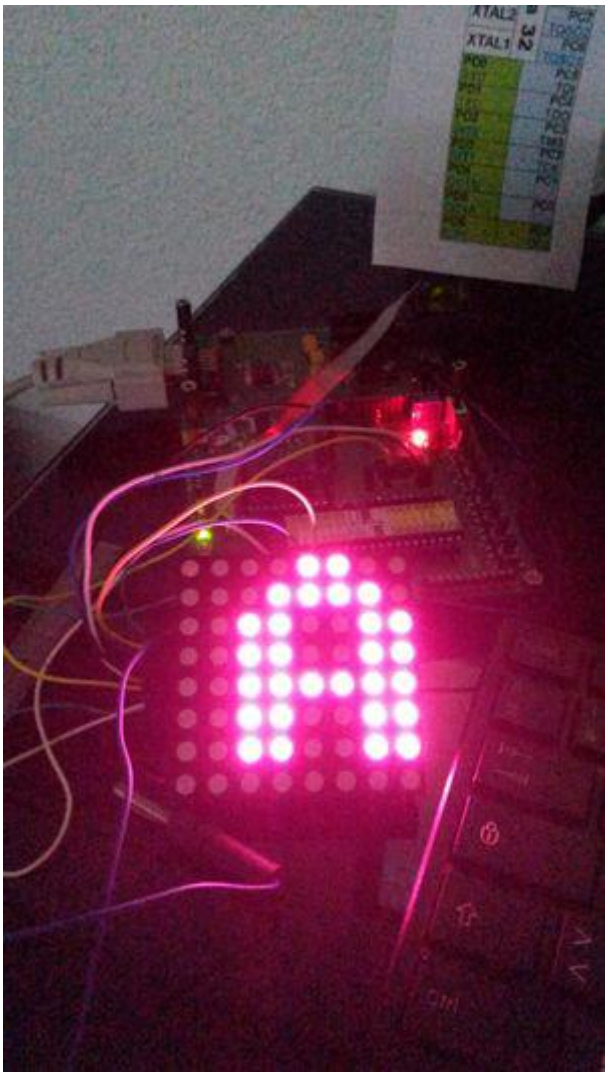
Le micro-contrôleur doit se charger, par interruptions, de balayer la matrice ligne par ligne pour allumer les LED de la ligne courante. Ce balayage doit se faire à une fréquence suffisante pour donner l'illusion d'un affichage stable.

# II - Analyse du projet

## *II.1 - Analyse du premier concurrent*

### *II.1.1 - Présentation générale*

Pilote de matrice RVB, par un instrument TLC5947 de Texas (SPI) et un PCF8574 (I2C),La matrice est chaînable.



### *II.1.2 - COMPOSANTS*

un TLC5947

Interface et CI IC / Interface d'affichage

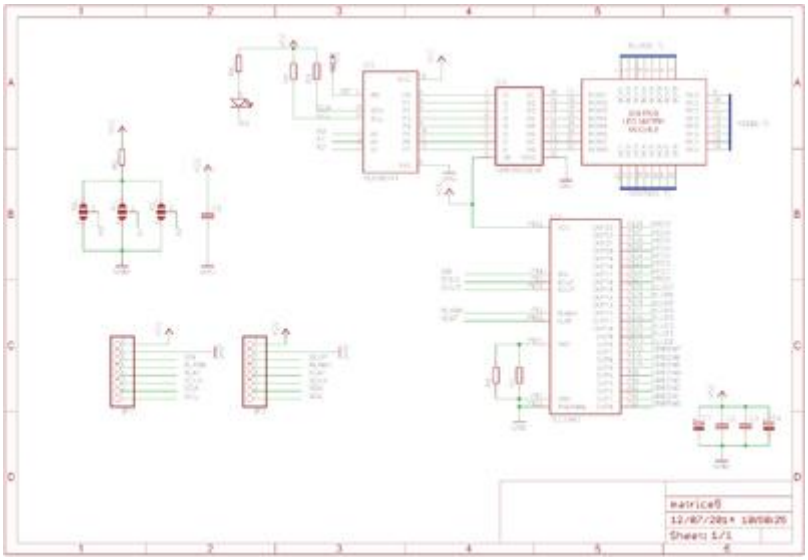
un PCF8574

Microprocessors, Microcontrollers, DSPs / Microprocessors (MPUs)

un UDN2982

CI d'interface et IO / Pilotes de périphériques et actionneurs

un RGB 8x8 LED MATRIX



## *II.2 - Analyse du second concurrent*

### *II.2.1 - Présentation générale*

Afin de mieux utiliser la technologie des écrans LED linéaires rotatifs basée sur le principe de la persistance visuelle (principe POV), une méthode basée sur le cœur TMC5947 et ARM Cortex-M3 STM32F103 a été conçue en associant la technologie tactile Qtouch à un écran LED rotatif. Contrôle de l'affichage LED haute résolution à faible coût et faible consommation.

## *II.2.2 - Conception matérielle du système*

Le STM32F103 est connecté à la LED via le TLC5947 pour contrôler l'affichage des LED sur le tableau tournant. Par exemple, le STM32F103 peut être utilisé pour contrôler la lumière LED afin d'afficher le motif d'horloge ou divers graphiques.Si les conditions le permettent, certains jeux simples peuvent être affichés. La DEL est connectée au processeur ARM et le motif d'affichage de la lampe DEL est modifié par le traitement du signal tactile par le processeur ARM.

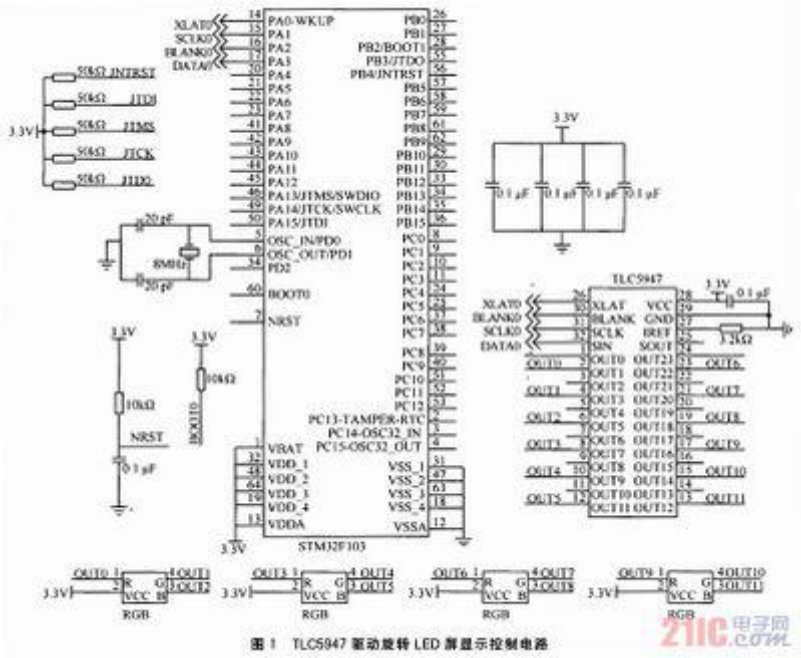


图 1 TLC5947 驱动旋转 LED 屏显示控制电路

# III - Préparation du projet

## *III.1 - Cahier des charges*

### *III.1.1 - Partie du matériel*

matrice de LEDs

Il contient un total de 441 LED dans 21x21

Circuit de contrôle

Il contient un TLC5947 et trois groupes de PCF8574 et UDN2982, il contrôle les colonnes par un TLC5947 et chaque ligne par des PCF8574 et des UDN2982

Microcontrôleur

ATMega328p

### *III.1.2 - Partie logiciel*

Une fonction qui convertit une chaîne en un QR code, sortie sous la forme d'un tableau à deux dimensions.

Une fonction d'interruption. Le micro-contrôleur doit se charger, contrôler 3 PCF8574 par interruptions (Nous assignons 3 adresses matérielles PCF8574 différentes pour le contrôle des interruptions), de balayer la matrice ligne par ligne pour allumer les LED de la ligne courante. Ce balayage doit se faire à une fréquence suffisante pour donner l'illusion d'un affichage stable.

Une fonction d'allumer. Utilisez les broches PCF8574 et TLC5947 haut et bas pour éclairer la lumière LED appropriée en fonction des informations de coordonnées en entrée.

## *III.2 - Choix techniques : matériel et logiciel*

### *logiciel*

Altium Designer

## *matériel principal*

1 x ATMega328p

1 x TLC5947

3 x PCF8574

3 x UDN2982

441 x LEDs

# IV - Réalisation du Projet

## *IV.1 - Réalisation des modules*

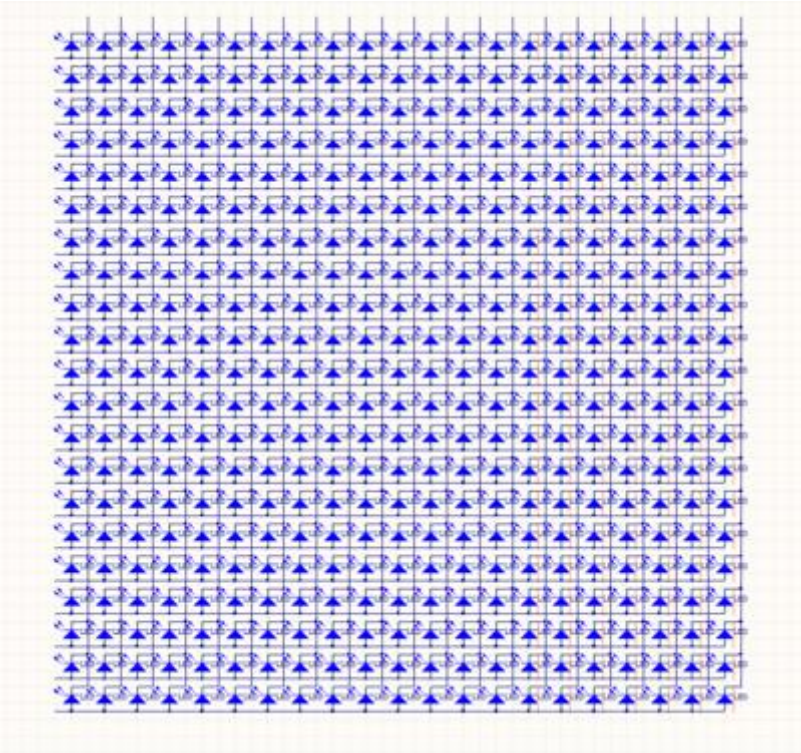
### *IV.1.1 - matrice de LEDs*

La matrice de points 21X21 est composée de 441 LED et chaque LED est placée à l'intersection des lignes et des colonnes.

S'allume, si le premier point doit être allumé, la 1ère broche est connectée au niveau haut a et le pied est connecté au niveau bas, puis le premier point est allumé;

Si la première ligne doit être allumée, la première broche du line doit être connectée au niveau haut et la broche de la colonne doit être connectée au niveau bas, puis la première ligne est allumée, si la première du colonne doit être allumée, la première ligne Lorsque la goupille est attachée basse et que la goupille sur la ligne est attachée haut, la première colonne sera allumée.

On fait le schematic en Altium Designer.

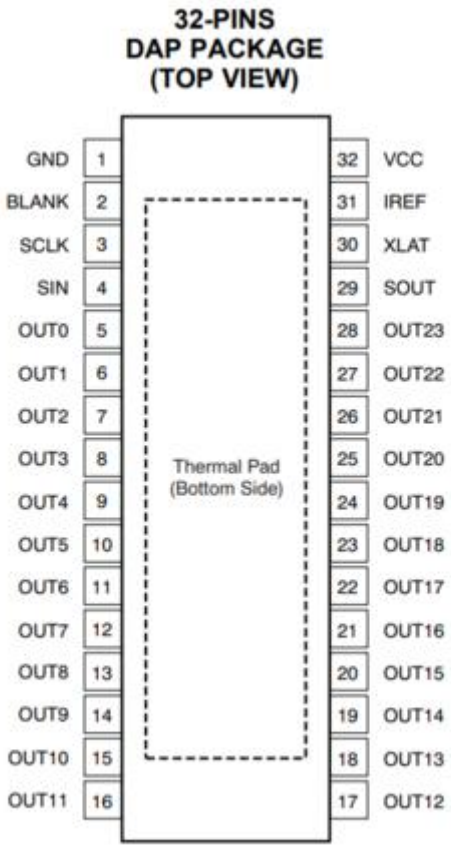


## *IV.1.2 - Circuit de contrôle*

## TLC5947

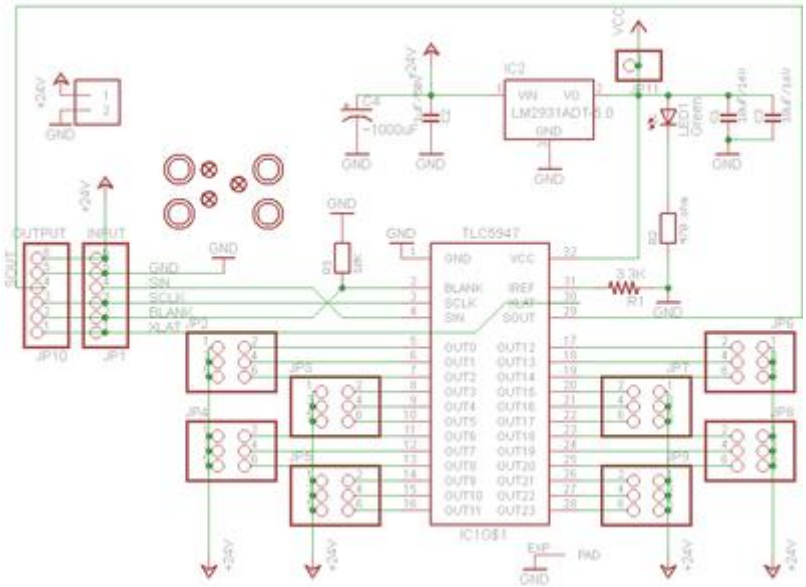
Le TLC5947 est un circuit d'attaque de LED de puits à courant constant et à 24 canaux. Chaque canal est réglable individuellement avec 4096 pas modulés en largeur d'impulsion (PWM). Le contrôle PWM est répété automatiquement avec les données en niveaux de gris (GS) programmées. Les données GS sont écrites via un port d'interface série. La valeur actuelle des 24 canaux est définie par une seule résistance externe.

Le TLC5947 est doté d'une fonction d'arrêt thermique qui désactive tous les pilotes de sortie en cas de surchauffe. Tous les pilotes de sortie redémarrent automatiquement lorsque la température revient à des conditions normales.




Le TLC5947 peut contrôler 24 canaux distincts avec une sortie PWM de 12 bits.

Afin de réaliser le TLC5947 sur mon carte, j'ai fait le schematic selon le datasheet de module TLC5947.




Connecting to the Arduino

These boards communicate using an SPI protocol. The wiring is slightly different for the two boards, so we will describe them separately. For making breadboard connections with the header pins on top of the board

Connectez-vous à l'Arduino comme suit:

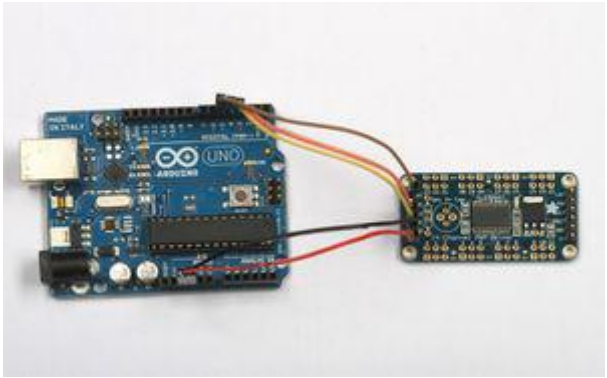
DIN -> Digital 4

CLK -> Digital 5

LAT -> Digital 6

GND -> GND

V + -> VIN

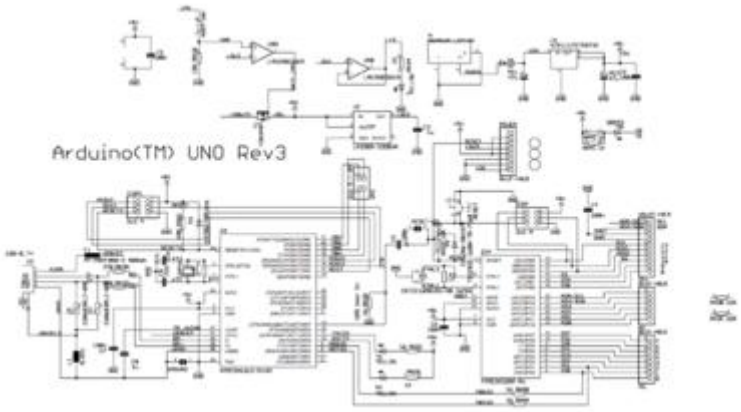


Ici, nous montrons V + connecté à la broche Arduino VIN, qui alimentera la carte opto-isolée et la LED directement à partir de l'alimentation connectée à la prise d'alimentation CC. Série de chaque canal.

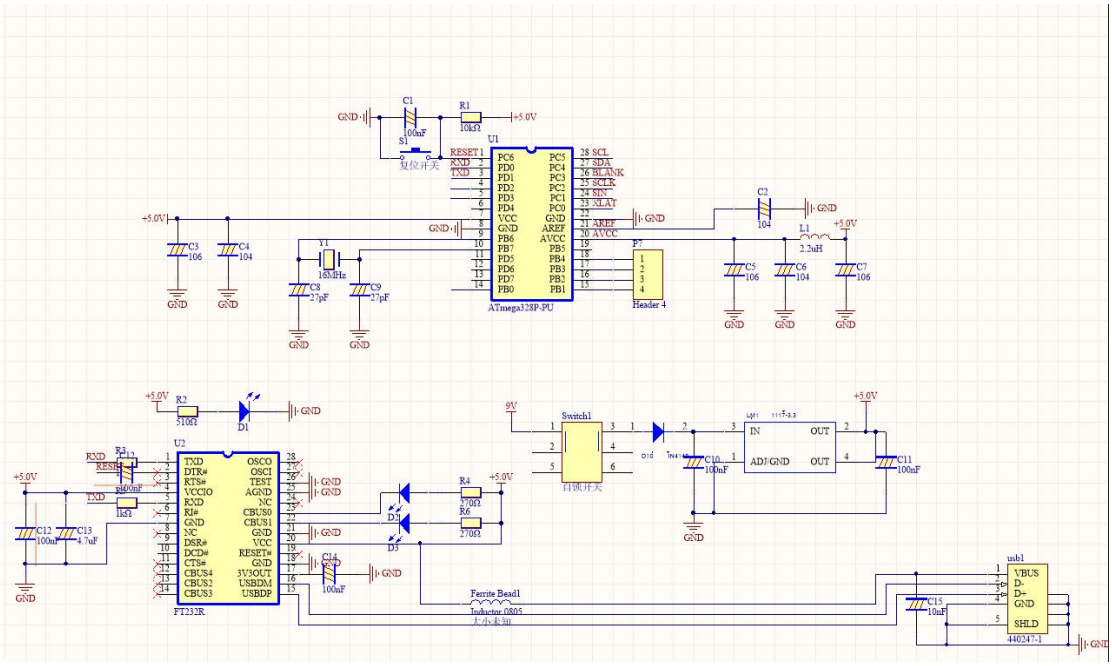
Les broches DIN / CLK / LAT peuvent être modifiées ultérieurement

## ATMega328p

Pour réaliser la partie contrôleur de circuit, on doit tout d'abord réaliser la partie complète du ATMega328p comme le cpu du notre circuit. Pour réaliser le systeme de ATMega328p,on utilise le schematic de Arduino UNO et combiné avec des documents en ligne,on fait le schematic du ATMega328p.




Afin de pouvoir se connecter au programme de gravure de l'ordinateur， fault ajouter un porte USB pour connecter avec l'ordinateur. Pour la partie connection, on choisit FT232R USB driver. Afin d'éviter un courant d'alimentation insuffisant， nous avons ajouté une batterie à l'alimentation USB. Nous pouvons choisir le mode d'alimentation via le commutateur，on a choisi le LM1117.IMA5 comme le regulateur de alimentation.



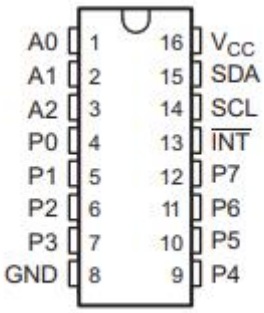

## PCF8574

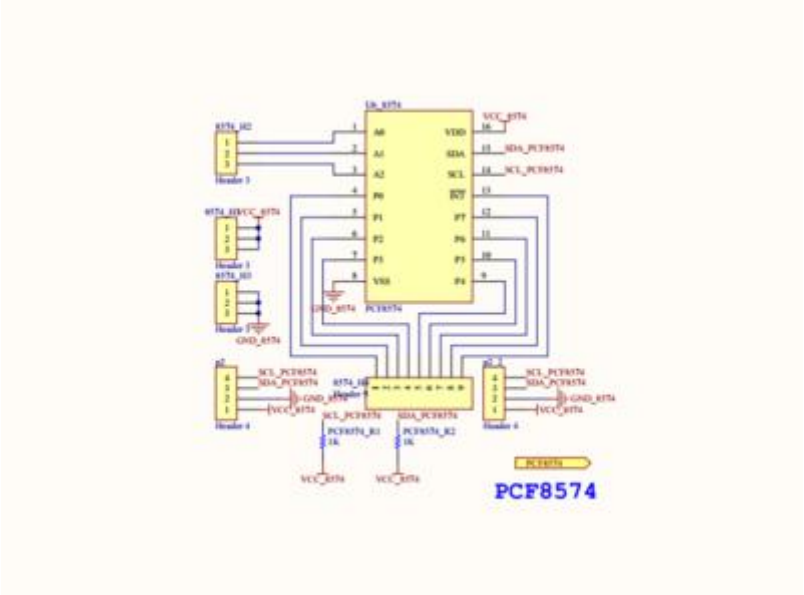
PCF8574 est un module d’extension d’entrée / sortie (E / S) à 8 bits pour le bus bidirectionnel à deux lignes (I2C) conçu pour un fonctionnement en VCC de 2,5 V à 6 V.

Le dispositif PCF8574 fournit une extension d’E / S distante à usage général pour la plupart des familles de microcontrôleurs via l’interface I 2C [horloge série (SCL), données série (SDA)].

L'appareil dispose d'un port d'E / S quasi-bidirectionnel 8 bits (P0 – P7), comprenant des sorties verrouillées avec une capacité de commande à courant élevé pour le pilotage direct de LED. Chaque E / S quasi bidirectionnelle peut être utilisée comme entrée ou sortie sans utiliser de signal de contrôle de la direction des données. À la mise sous tension, les E / S sont élevées. Dans ce mode, seule une source de courant vers VCC est active.

Afin de réaliser le PCF8574 sur mon carte, j'ai fait le schematic selon le datasheet de module PCF8574.




Dans le sujet, il fault utiliser 3 UDN2982 pour gerer l'energy, mais ça fait trop long de commander les UDN2982 en Frence, après avoir cherché des composants alternatifs, j'ai décidé de remplacer le UDN2982 par le A2982SLW. Sur le datasheet, il a dit que:

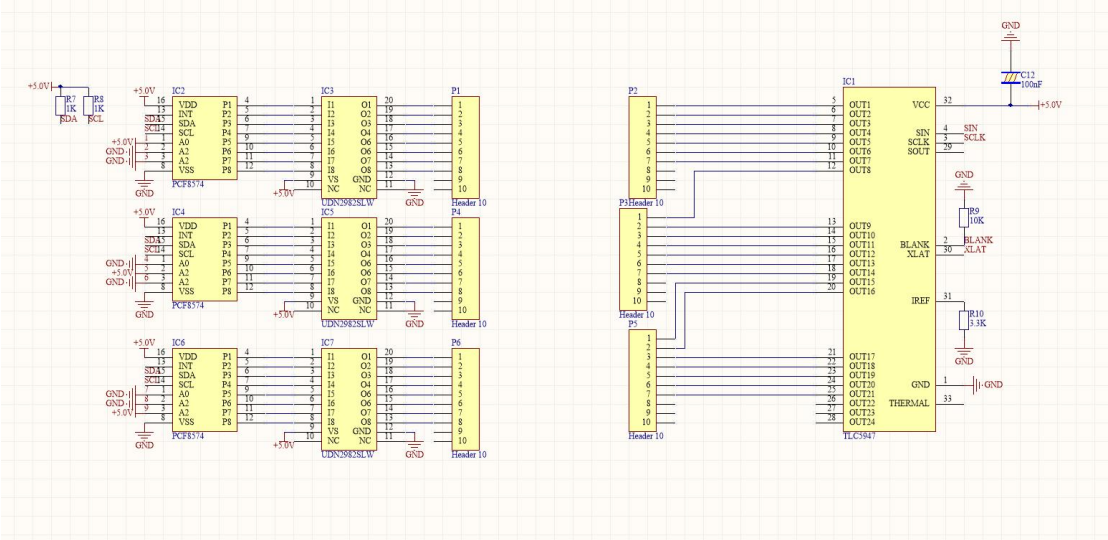
Recommandés pour les applications de commutation côté haut bénéficiant d'une logique et d'une masse de charge séparées, ces appareils incluent des tensions d'alimentation de charge jusqu'à 50 V et des courants de sortie de -500 mA. Ces pilotes sources à 8 canaux sont utiles pour l'interfaçage entre la logique de bas niveau et les charges à forte intensité. Les charges typiques comprennent les relais, les solénoïdes, les lampes, les moteurs pas à pas et / ou les servomoteurs, les marteaux d'impression et les DEL.

Les UDN2981A, UDN2982A et A2982SLW sont électriquement interchangeables, résistent à une tension maximale de sortie de 50 V et fonctionnent à un minimum de 5 V. Tous les appareils de cette série intègrent des résistances de limitation du courant d'entrée et des diodes de suppression des transitoires de sortie, et sont activés par une entrée active haute.

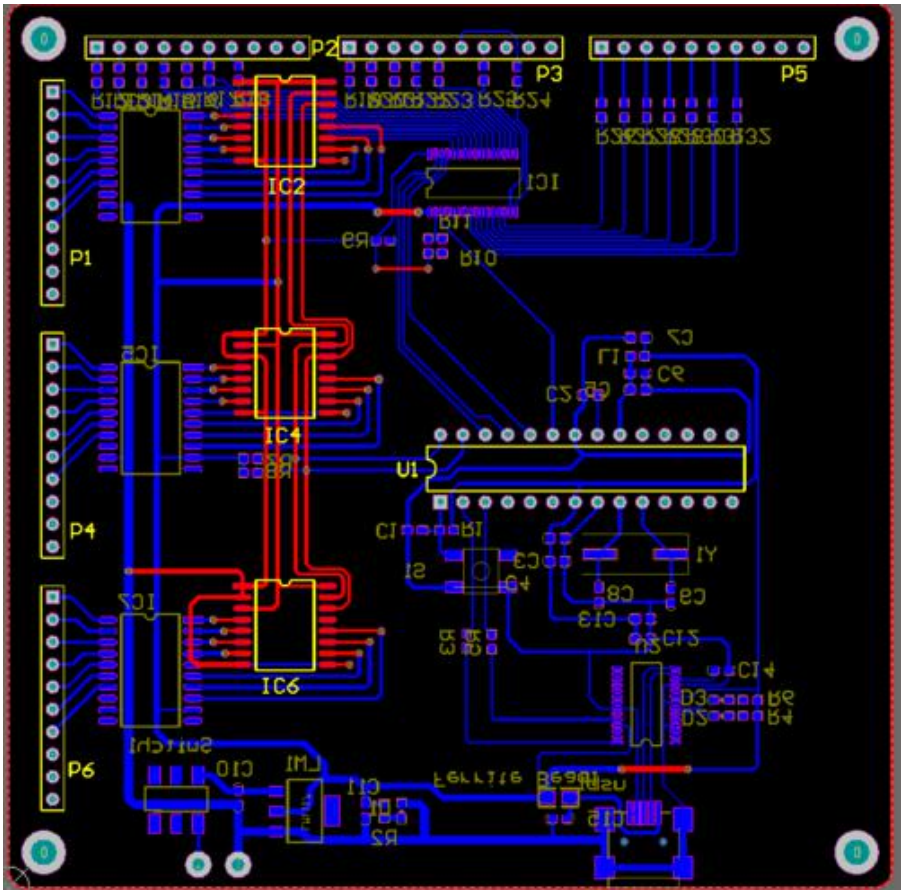
Donc je peux remplacer le UDN2982 par le A2982SLW.

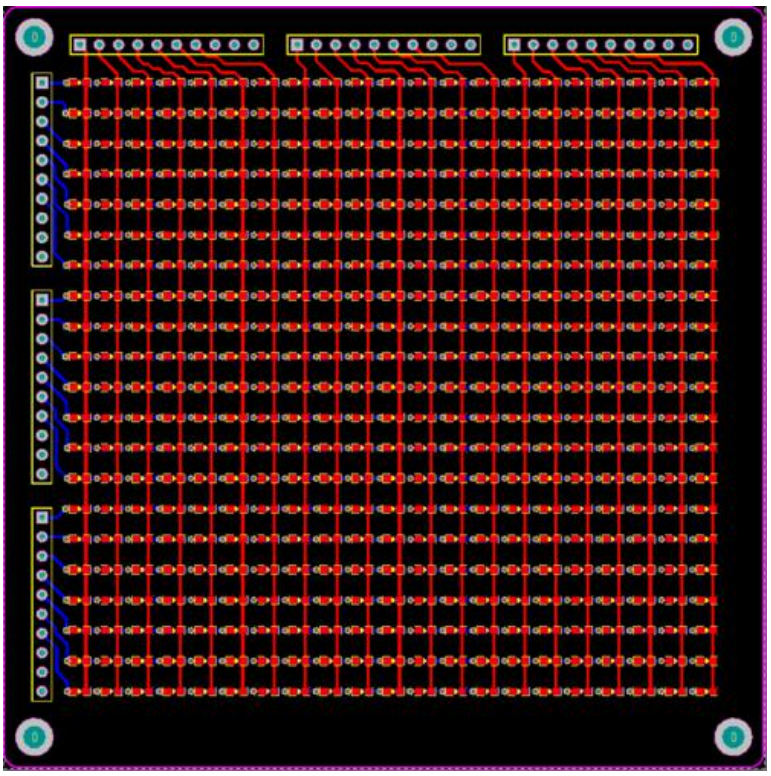
Apres, on utilise 3 groupe de PCF8574 à contrôler 21 lines (utilise P1-P7 chaque groupe et on a 21 portes pour gerer tout les lines). On utilise A0-A2 à choisir les

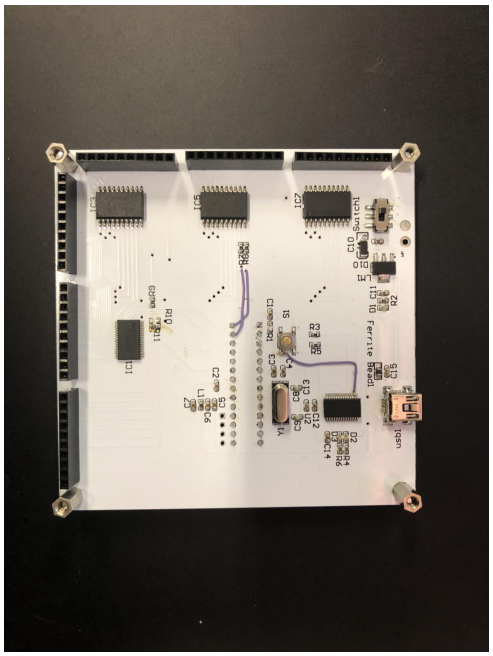
adresses des composants dont on peux les contrôler séparé avec interruption. Et on utilise TLC5947 à contrôler 21 colonnes.

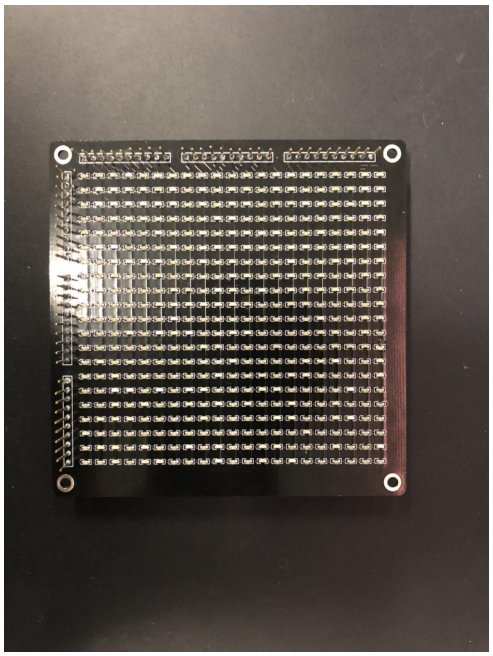


Après avoir dessiné le diagramme schématique de chaque composant et la méthode de connexion, j'ai continué à compléter la carte de circuit imprimé.









# V - Partie du programme

Pour la partie du programme,il contient principalement 2 partie.

Une fonction transfert la chaîne caractère à QR code.

Une fonction d'afficher

## *V.1 - La fonction d'afficher*

Si vous commencez par convertir la chaîne en un code à deux dimensions, le code QR est sorti sous forme de tableau à deux dimensions, qui est une matrice de 21 sur 21, où les valeurs ne sont que 0 et 1, 1 pour les taches noires, 0 pour le blanc. Bloc de couleur. Ensuite, nous traitons d'abord toutes les lignes et analysons le tableau à deux dimensions.Pour le port i de PCF8574, s'il n'ya qu'un tableau dans le tableau à deux dimensions m [i] [?], Nous allons définir le port i. Est élevé. Nous analyserons ensuite le tableau 2D une seconde fois. Pour le port l du TLC5947, si m [i] [j] = 1, le voyant de ce point doit être allumé, nous allons définir la broche TLC comme suit: Niveau bas, la DEL est allumée à ce moment-là. Si m [i] [j] = 0, c'est-à-dire si la DEL de ce point doit être éteinte, nous fixons la broche TLC à un niveau élevé.A ce stade, la DEL est électriquement égale aux deux extrémités et ne peut pas être allumée, puis la DEL s'éteint. . De cette manière, nous avons obtenu un contrôle par LED pour une seule rangée.

Ensuite, afin d'atteindre cette fonction dans la première milliseconde, nous allumons la première ligne puis nous essayons de rompre l'interruption, puis dans la seconde milliseconde nous éclairons la deuxième ligne, jusqu'à la septième milliseconde, nous sommes sur la première. Le scan de la puce PCF8574 est terminé, la commutation sur la seconde puce commence à allumer la broche 1 de la seconde puce, qui est la 8ème ligne de la matrice. Nous avons implémenté la fonction d'affichage en balayant cycliquement ces 21 lignes.

Afin d'implémenter cette fonction dans le code, j'ai utilisé la bibliothèque de fonctions fournie avec PCF8574 et TLC5947. Pour le PCF8574, nous définissons son adresse comme 0x20 et définissons une fonction pour contrôler le niveau de sortie du port et l'adresse de commutation permettant de réaliser un cycle d'un cycle par analyse et une adresse comprise entre 0x21, 0x22 et 0x24.

Pour TLC5947, nous appelons la fonction tlc.setPWM () dans la fonction de bibliothèque Après avoir déterminé le port de sortie du PCF8574, nous utilisons cette fonction pour contrôler le niveau de sortie du port TLC5947 afin de contrôler la ligne.

```
#include <Wire.h>
#include "Adafruit_TLC5947.h"

// How many boards do you have chained?
#define NUM_TLC5974 7

#define data   9
#define clock   10
#define latch   11
#define oe  -1  // set to -1 to not use the enable pin (its optional)

Adafruit_TLC5947 tlc = Adafruit_TLC5947(NUM_TLC5974, clock, data, latch);

#define EXP_ADDRESS 0x20
String comdata = "";
uint8_t qrcodeData[21*21];



//Initialiser le protocole wire
void setup() {
    Wire.begin(); // i2c
}


void loop() {
    static uint32_t databuffer[21];

    for(int row=0;row<21;row++){
        for(int col=0;col<21;col++){
            tlc.setPWM(col,(databuffer[col] ? 4095:0));
         }
        tlc.write();
        PCF8574Write(1,row);
        delay(1);
        PCF8574Write(0,row);
      }

}
void PCF8574Write(uint8_t _data,uint8_t row)
{

     uint8_t addr = 2^(2-row/7);
```

```
        Wire.beginTransmission(EXP_ADDRESS+addr);          //envoyer
l'address
        Wire.write(_data<<(row%7));                        //envoyer des
donnees
        Wire.endTransmission();                            //Fin de
transmission


    }
```

## *V.2 - La fonction de QR code*

Le code bidimensionnel enregistre les informations de symbole de données selon un motif noir et blanc réparti dans un plan (direction bidimensionnelle) par un certain motif géométrique selon un motif donné, et utilise habilement le "0" qui constitue le fondement logique interne de l'ordinateur pour la préparation du code. Le concept de flux de bits "1" utilise plusieurs formes géométriques correspondant au binaire pour représenter des informations numériques littérales, et est automatiquement lu par un dispositif de saisie d'image ou un dispositif de balayage photoélectrique afin de réaliser un traitement automatique de l'information: il présente des points communs en matière de technologie de code à barres : Chaque système de code a son propre jeu de caractères spécifique, chaque caractère occupe une certaine largeur et une certaine fonction de contrôle. En même temps, il dispose également d'une fonction de reconnaissance automatique pour différentes lignes d'informations et gère le point de changement de rotation du graphique.

### Code QR de la matrice

Il est codé dans un espace rectangulaire par différentes distributions de pixels noirs et blancs dans la matrice. Dans la position de l'élément correspondant de la matrice, l'apparition d'un point (point carré, point ou autre forme) représente un "1" binaire et le point ne semble pas représenter un "0" binaire. La disposition des points détermine le code à barres bidimensionnel de la matrice. Le sens du représentant. Le code à barres bidimensionnel de la matrice est un nouveau type de système de code de lecture et de traitement automatique de symboles graphiques basé sur la technologie de traitement d'images par ordinateur et le principe de codage combiné.

Nous avons choisi d'utiliser le port série pour réaliser la transmission de chaînes.

Le code est simple, comdata est une variable de type chaîne. Serial.available () est la quantité de données dans le pool de mémoire tampon série en cours. Serial.read () est une instruction qui lit le pool de mémoire tampon et ne peut lire qu'un octet à la fois.

En utilisant la variable de type String, il est très simple d'implémenter l'ajout de caractères à des chaînes, ainsi que la sortie de chaînes, l'affectation et d'autres problèmes gênants, afin qu'un code très simple puisse gérer des données en série.

Une attention particulière est accordée au délai (2) lorsque la lecture du port série ne peut pas être supprimée, sinon le tampon série n'est pas assez long pour accepter les données. Même si vous réduisez le délai, vous obtiendrez une erreur. Les valeurs spécifiques peuvent également être déterminées expérimentalement.

Rappelez-en un: comdata est une chaîne, également un tableau, vous pouvez utiliser comdata [0], comdata [1] pour chaque mot. . . Comdata [n]. Si nous voulons supprimer chaque octet, nous pouvons nous référer à chacun.

Effet: Quelle chaîne est entrée et quelle est la sortie.

Un code QR est composé de nombreux petits carrés, appelés modules, qui représentent des données codées, avec correction d'erreur supplémentaire (permettant de lire les codes QR partiellement endommagés).

La version d'un code QR est un nombre compris entre 1 et 40 (inclus), qui indique la taille du code QR. La largeur et la hauteur d'un code QR sont toujours égales (c'est carré) et sont égales à 4 * version + 17.

Le niveau de correction d'erreur est un nombre compris entre 0 et 3 (inclus), ou peut être l'un des noms symboliques ECC_LOW, ECC_MEDIUM, ECC_QUARTILE et ECC_HIGH. Des niveaux plus élevés de correction d'erreur sacrifient la capacité de données, mais permettent à une plus grande partie du code QR d'être endommagée ou illisible.

Le mode d'un code QR est déterminé par les données encodées. Chaque mode est codé en interne à l'aide d'une représentation compacte, de sorte que les modes inférieurs peuvent contenir plus de données.

NUMÉRIQUE: chiffres (0-9)

ALPHANUMERIC: lettres majuscules (AZ), chiffres (0-9), espace (), signe dollar ($), signe de pourcentage (%), astérisque (*), plus (+), moins (-), point décimal (.), barre oblique (/) et deux points (:).

BYTE: n'importe quel caractère

| Version | Size | Error Correction | Mode | | |
|---|---|---|---|---|---|
| | | | Numeric | Alphanumeric | Byte |
| 1 | 21 x 21 | LOW | 41 | 25 | 17 |
| | | MEDIUM | 34 | 20 | 14 |
| | | QUARTILE | 27 | 16 | 11 |
| | | HIGH | 17 | 10 | 7 |

Et on utilise le librairie QRcode pour realiser le transformation.

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

String comdata = "";
QRCode qrcode;
uint8 t qrcodeData[21*21];

//初始化串口和 wire 协议
void setup() {
    Serial.begin(115200);
    Wire.begin(); // i2c
    // Start time
    uint32_t dt = millis();

    // Create the QR code
    //qrcode_getBufferSize(1);

    qrcode_initText(&qrcode, qrcodeData, 1, 0, "HELLO WORLD");

    // Delta time
    dt = millis() - dt;
    Serial.print("QR Code Generation Time: ");
    Serial.print(dt);
    Serial.print("\n");

}

void loop() {
   static uint32_t databuffer[21];
     while (Serial.available() > 0)                    // verifier s'il
y a des données dans la mémoire tampon du port série
    {
```

```
          comdata += char(Serial.read());                  //S'il y a des
données, elles seront reçues jusqu'à ce qu'elles soient vides.        La
chaîne reçue existe dans comdata
          delay(2);
       }




     if (comdata.length() > 0)
       {
          Serial.println(comdata);
          uint16 t len=uint16 t (comdata.length());
          qrcode initBytes(&qrcode, qrcodeData, 1, 0,&comdata[0],len);
          Serial.print("\n\n\n\n");
          for (uint8_t y = 0; y < qrcode.size; y++) {

            // Left quiet zone
            Serial.print("       ");

            // Each horizontal module
            for (uint8 t x = 0; x < qrcode.size; x++) {

                //Print each module
               Serial.print(qrcode getModule(&qrcode, x, y) ? "#": " ");
            }
            Serial.print("\n");
          }

          comdata = "";
       }

   }
```

Avec le code, on a réaliser convertir une chaîne en code QR.


On utilise la fonction qrcode_getModule(&qrcode, col, row) pour lire les coordonnées de chaque point du code QR et les envoyer au TLC5947.

Si on utilise directement la valeur de la sortie de la fonction sur TLC, la fréquence de scanner sera réduite. Afin d'améliorer l'effet d'affichage, nous définissons un tableau à deux dimensions pour stocker les données du code QR et les lisons directement lors de l'affichage.

Après modification, j'ai fini le programme complet:

```
#include <Wire.h>
#include "qrcode.h"
#include "Adafruit_TLC5947.h"


#define NUM_TLC5974 1
#define  dat   4
#define  clk   5
#define  lat   6
```

```
#define oe  -1  // set to -1 to not use the enable pin (its optional)
#define EXP_ADDRESS 0x20

String comdata = "";
QRCode qrcode;
uint8_t qrcodeData[56];
uint16_t pwmData[21][21];
uint16_t *pwmbuffer = (uint16_t *)malloc(2 * 24*NUM_TLC5974);


void setup() {
    Serial.begin(115200);
    Wire.begin(); // i2c
    // Start time
     TLC5947.begin();
    uint32_t dt = millis();


    qrcode_initText(&qrcode, qrcodeData, 1, 0, "HELLO WORLD");

    // Delta time
    dt = millis() - dt;
    Serial.print("QR Code Generation Time: ");
    Serial.print(dt);
    Serial.print("\n");

}

void loop() {

     while (Serial.available() > 0)
    {
        comdata += char(Serial.read());
        delay(2);
    }
    if (comdata.length() > 0)
    {
        Serial.println(comdata);
        uint16_t len=uint16_t (comdata.length());
        qrcode_initBytes(&qrcode, qrcodeData, 1, 0,&comdata[0],len);
         for(int row=0;row<21;row++){

            for(int col=0;col<21;col++){
```

```
                    pwmData[row][col]=qrcode_getModule(&qrcode, col, row) ? 1:0;
                     Serial.print( pwmData[row][col] ? "#": " ");
                }
               Serial.print("\n");

           }

          //Serial.print("\n\n\n\n");
          comdata = "";
      }


      for(int row=0;row<21;row++){
          PCF8574Write(1,row);
          for(int col=0;col<21;col++){
               TLC5947_setPWM(col,pwmData[row][col]);
               //TLC5947_setPWM(col,qrcode_getModule(&qrcode, col, row) ? 4095:0);
           }
          // Serial.print("\n");
           TLC5947_write();
          //delay(100);
          Clear();
        }

   }
   void PCF8574Write(uint8_t _data,uint8_t row)
   {

         uint8_t addr = 1<<(row/7);
        //uint8_t addr = row/8;
        Wire.beginTransmission(EXP_ADDRESS+addr);
        Wire.write( _data<<(row%7));
        Wire.endTransmission();

   }
   void Clear()                          //清空显示

   {

     for(int i = 0;i<21;i++)
     {
       TLC5947_setPWM(i,0);
```

```
    }
     TLC5947_write();
     for(int j=0;j<3;j++){
    PCF8574Write(0,j*8);
    }
  }
  void TLC5947_init(uint16_t n, uint8_t c, uint8_t d, uint8_t l) {
    memset(pwmbuffer, 0, 2*24*n);
  }
  boolean TLC5947_begin() {
    if (!pwmbuffer) return false;

    pinMode( clk, OUTPUT);
    pinMode( dat, OUTPUT);
    pinMode( lat, OUTPUT);
    digitalWrite(_lat, LOW);

    return true;
  }
  void TLC5947_setPWM(uint16_t chan, uint16_t pwm) {
    if (pwm > 4095) pwm = 4095;
    if (chan > 24*NUM_TLC5974) return;
    pwmbuffer[chan] = pwm;
  }
  void TLC5947_write(void) {
    digitalWrite( lat, LOW);
    // 24 channels per TLC5974
    for (int16_t c=24*NUM_TLC5974 - 3; c >= 0 ; c--) {
      // 12 bits per channel, send MSB first
     for (int8_t b=11; b>=0; b--) {
       digitalWrite( clk, LOW);

  //     if (pwmbuffer[c] & (1 << b))
  //       digitalWrite( dat, HIGH);
  //     else
  //       digitalWrite( dat, LOW);
        digitalWrite(_dat, pwmbuffer[c]);

       digitalWrite( clk, HIGH);
      }
    }
    digitalWrite( clk, LOW);

    digitalWrite( lat, HIGH);
```

```
    digitalWrite( lat, LOW);
  }
```