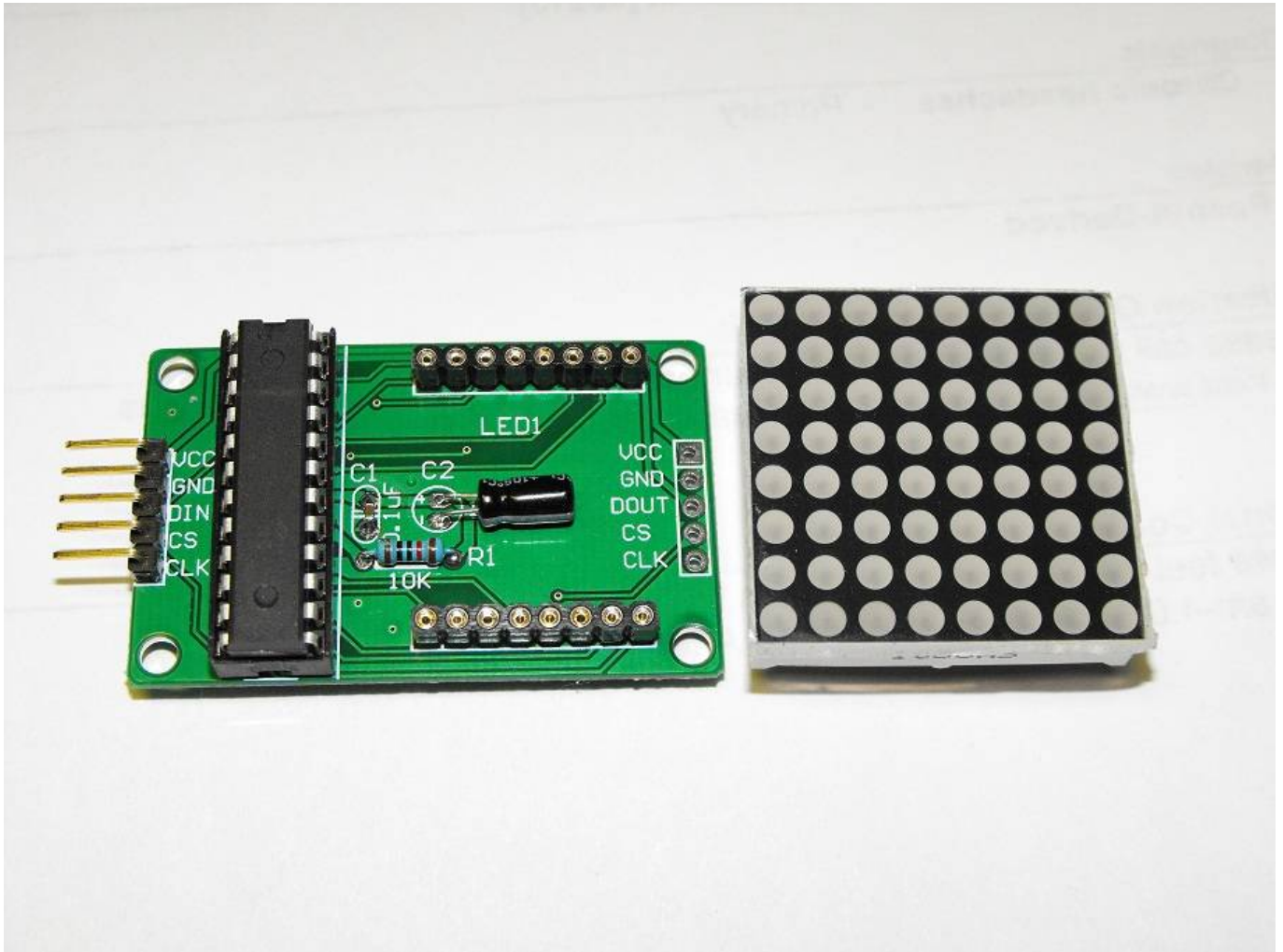


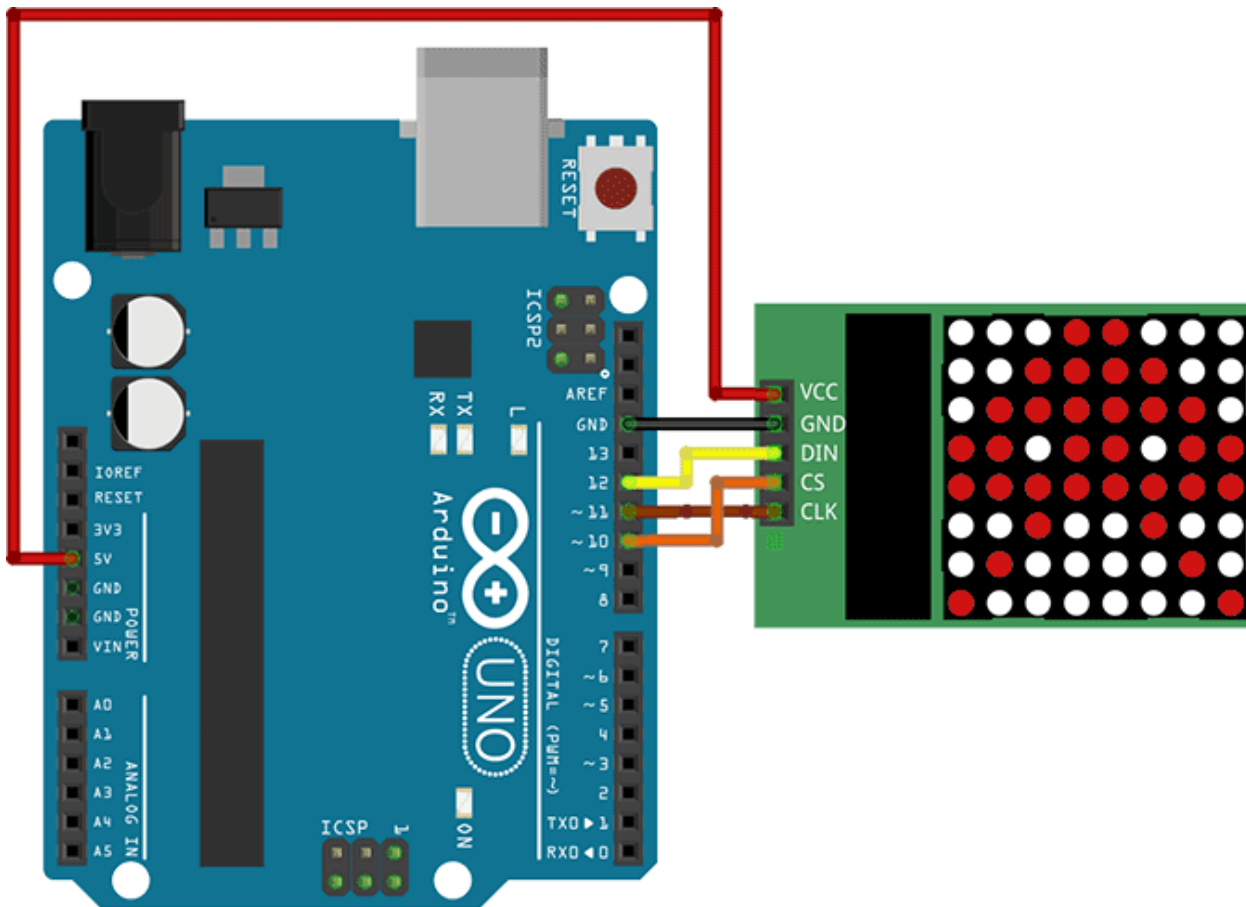


Projet électronique : Gestion d'une matrice des LED avec Arduino





Projet électronique : Gestion d'une matrice des LED avec Arduino



Sommaire

- [0.1 Objectifs et analyse de fonctionnement](#)
- [0.2 Fonctionnement de la matrice des LEDs 8x8](#)
- [0.3 Lien vers une Application gratuite de gestion d'une matrice 8x8 des LED](#)
- [0.4 Commentaires sur le programme](#)
- [0.5 Les fonctions disponibles dans la librairie LedControl.h \(MAX7219/MAX7221\)](#)
- [0.6 Exemple de programme d'utilisation de la librairie](#)
- [0.7 Fichiers projets + Librairie en C du projet électronique \(.rar\)](#)
- [0.8 Photos du projet électronique](#)
- [0.9 Rappel des cours:](#)
- [1 Cours Arduino Composants électroniques](#)
- [2 Un petit commentaire de vous, un Grand encouragement pour nous](#)

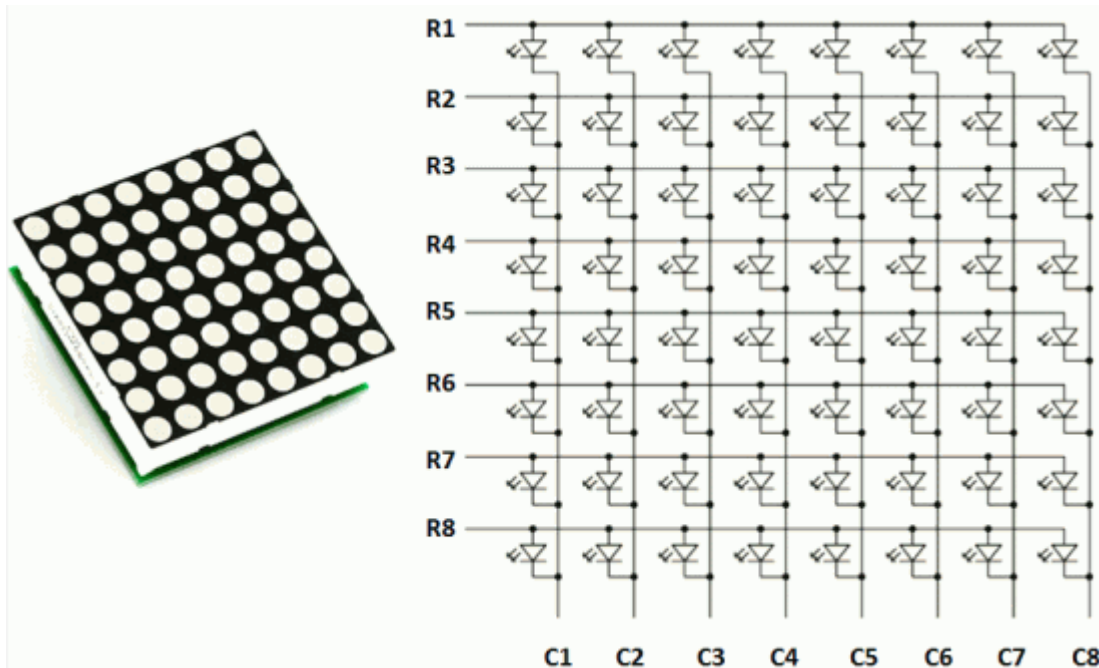


▪ 2.0.1 - Bon Courage -

Objectifs et analyse de fonctionnement

Le projet consiste la gestion d'une matrice des LED 8X8 à base d'Arduino en utilisant le circuit MAX7219CNG. Le projet électronique utilise une librairie pour la gestion du circuit MAX7219CNG avec une liaison SPI. Ce mini projet mis en évidence l'utilisation de la librairie avec d'autres fonctions secondaires (décalage, conversion, ...).

Fonctionnement de la matrice des LEDs 8×8



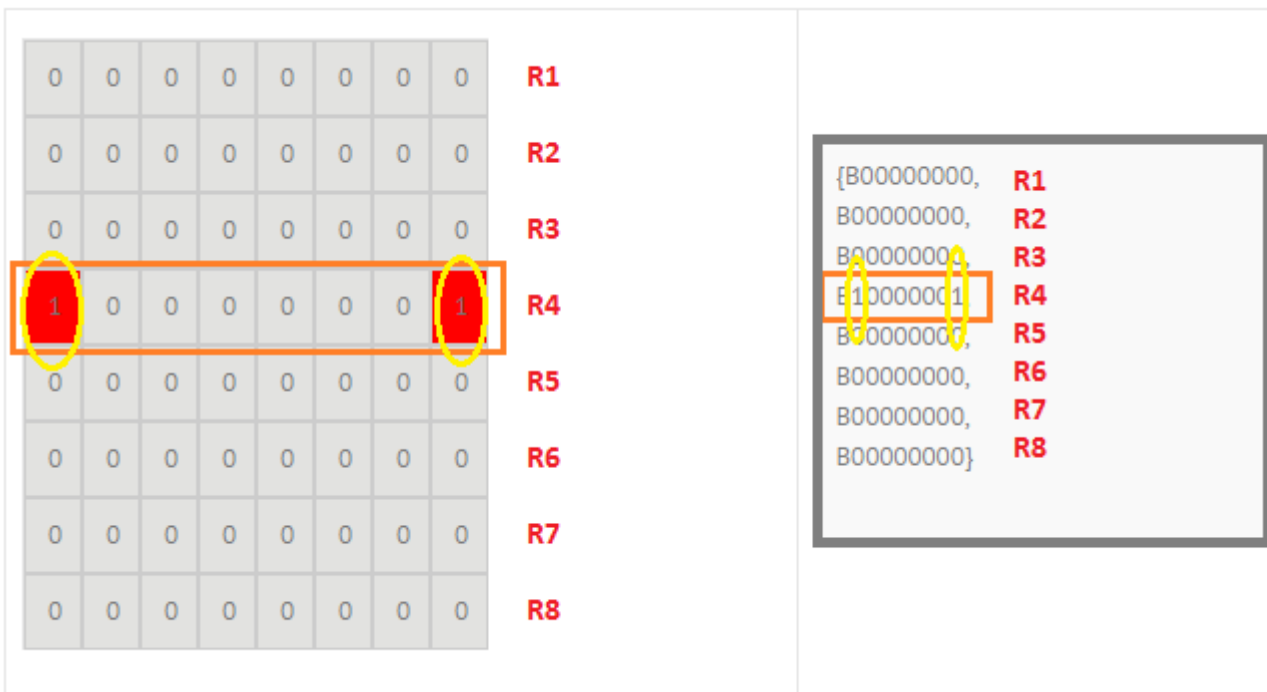
La matrice des LED est constituée de 8 lignes et 8 colonnes. On peut assimiler la matrice à un tableau 1D de 8 éléments, chaque élément du tableau est codé sur 8 bits Ex B00001111 ou 0x0F, dans la figure ci-dessous illustre le passage d'une matrice 8×8 à un tableau sur 8 éléments de type Byte ou Unsigned char. On verra dans la suite la fonction qui permet d'afficher le code dans la matrice 8×8.



Le module utilisé est basé sur MAX7219 avec une liaison série pour les données. Le bus SPI est constitué de trois fils :

- **CS** : Chip Select pour l'activation du module
- **CLK** : l'horloge synchrone pour les données. L'horloge définit la vitesse de transmission des données série
- **DIN** : La donnée série sur 8 bits

Plus des détails sur la transmission série : [SPI](#)



Lien vers une Application gratuite de



gestion d'une matrice 8×8 des LED

Commentaires sur le programme

```
void int2BitArray(unsigned char dataIn, unsigned char taille, byte *dataArray)
```

La fonction **int2BitArray** permet de convertir une variable char en un tableau sur N bits (N=8 bits)

Ex1 : pour dataIn=10, taille=8 alors la variable de sortie dataArray[8]={0,0,0,0,1,0,1,0} (équivalent à B00001010)

Ex2 : dataIn=127, taille=8 alors la variable de sortie dataArray[8]={0,1,1,1,1,1,1,1} (équivalent à B01111111)

```
void DataShiftArray(unsigned char NumShift, unsigned char taille, byte *DataIn, byte *DataOutShift)
```

La fonction **DataShiftArray** permet de décaler à droite un tableau de longueur "taille" de bits (tableau binaire) DataIn, le résultat est stocké puis retourné dans le tableau DataOutShift. Au début le tableau DataOutShift est supposé initialisé à 0.

Ex : NumShift=3, taille = 8, DataIn[8]={1,1,1,1,0,0,0,0}
alors DataOutShift[8]={**0,0,0,1,1,1,1,0**}.

L'objectif de la fonction est de décaler un objet dans la matrice 8×8 des LED.



Les fonctions disponibles dans la bibliothèque LedControl.h (MAX7219/MAX7221)

```
class LedControl {  
    private
```

```
    /* The array for shifting the data to the devices */  
    byte spidata[16];  
    /* Send out a single command to the device */  
    void spiTransfer(int addr, byte opcode, byte data);  
  
    /* We keep track of the led-status for all 8 devices in this array */  
    byte status[64];  
    /* Data is shifted out of this pin*/  
    int SPI_MOSI;  
    /* The clock is signaled on this pin */  
    int SPI_CLK;  
    /* This one is driven LOW for chip selectzion */  
    int SPI_CS;  
    /* The maximum number of devices we use */  
    int maxDevices;  
  
public:  
    /*  
    * Create a new controller  
    * Params :  
    * dataPin          pin on the Arduino where data gets shifted out  
    * clockPin         pin for the clock  
    * csPin            pin for selecting the device  
    * numDevices       maximum number of devices that can be controlled  
    */  
    LedControl(int dataPin, int clkPin, int csPin, int numDevices=1);  
  
    /*  
    * Gets the number of devices attached to this LedControl.  
    * Returns :  
    * int             the number of devices on this LedControl  
    */
```



```
int getDeviceCount();

/*
 * Set the shutdown (power saving) mode for the device
 * Params :
 * addr      The address of the display to control
 * status    If true the device goes into power-down mode. Set to false
 *           for normal operation.
 */
void shutdown(int addr, bool status);

/*
 * Set the number of digits (or rows) to be displayed.
 * See datasheet for sideeffects of the scanlimit on the brightness
 * of the display.
 * Params :
 * addr      address of the display to control
 * limit     number of digits to be displayed (1..8)
 */
void setScanLimit(int addr, int limit);

/*
 * Set the brightness of the display.
 * Params:
 * addr      the address of the display to control
 * intensity the brightness of the display. (0..15)
 */
void setIntensity(int addr, int intensity);

/*
 * Switch all Leds on the display off.
 * Params:
 * addr      address of the display to control
 */
void clearDisplay(int addr);

/*
 * Set the status of a single Led.
 * Params :
 * addr      address of the display
 * row       the row of the Led (0..7)
 * col       the column of the Led (0..7)
 * state     If true the led is switched on,
 *           if false it is switched off
 */
```



```
*/  
void setLed(int addr, int row, int col, boolean state);  
  
/*  
* Set all 8 Led's in a row to a new state  
* Params:  
* addr      address of the display  
* row       row which is to be set (0..7)  
* value     each bit set to 1 will light up the  
*           corresponding Led.  
*/  
void setRow(int addr, int row, byte value);  
  
/*  
* Set all 8 Led's in a column to a new state  
* Params:  
* addr      address of the display  
* col       column which is to be set (0..7)  
* value     each bit set to 1 will light up the  
*           corresponding Led.  
*/  
void setColumn(int addr, int col, byte value);  
  
/*  
* Display a hexadecimal digit on a 7-Segment Display  
* Params:  
* addr      address of the display  
* digit     the position of the digit on the display (0..7)  
* value     the value to be displayed. (0x00..0x0F)  
* dp        sets the decimal point.  
*/  
void setDigit(int addr, int digit, byte value, boolean dp);  
  
/*  
* Display a character on a 7-Segment display.  
* There are only a few characters that make sense here :  
*   '0','1','2','3','4','5','6','7','8','9','0',  
*   'A','b','c','d','E','F','H','L','P',  
*   '.', '-', '_', ' '  
* Params:  
* addr      address of the display  
* digit     the position of the character on the display (0..7)  
* value     the character to be displayed.  
* dp        sets the decimal point.
```




```
*/  
void setChar(int addr, int digit, char value, boolean dp);  
};
```

Exemple de programme d'utilisation de la librairie

```
//Librairie à importer !  
#include "LedControl.h"  
  
/*  
Configuration des pins (MAX72XX)  
LedControl lc=LedControl(12,11,10,1);  
Pin 12 connecté au DataIn  
Pin 11 connecté au CLK  
Pin 10 connecté au LOAD  
*/  
  
LedControl lc=LedControl(12,10,11,1);  
  
/* Temporisation de mise en service de l'afficheur */  

```



```
byte t[8]={B00000000,B11111111,B11111111,B00011000,B00011000,B00011000,B00011000,B
00011000};
byte Data[8]={0,0,0,0,0,0,0,0};
byte DataShif[8]={0,0,0,0,0,0,0,0};
byte XData[8]={B10111101,B11000011,B10011001,B10011001,B01000010,B01111110,B100110
01,B10011001};
byte EM[8]={B11110001,B11000011,B10100101,B10011001,B11111001,B10000001,B10000001,
B11110001};

/* Affichage du caractère T colonne par colonne (8écritures) */
for(int i=0;i<8;i++)
{
    // Afficher la ligne t[i] à la position de la colonne i
    lc.setColumn(0,i,t[i]);
    delay(delaytime);
}
lc.clearDisplay(0);

delay(1000);

/* Affichage du caractère T ligne par ligne (Rotation 90° par rapport
à la fonction setColumn ((8 écritures)*/
for(int i=0;i<8;i++)
{
    // Afficher la colonne t[i] à la position de la ligne i
    lc.setRow(0,i,t[i]);
    delay(delaytime);
}
lc.clearDisplay(0);
delay(1000);

/* Ecriture dans l'afficheur pixel par pixel (8x8 écritures) */
for(int k=0;k<8;k++)
{
    //Décalage à droite de valeur=k [0..7]
    DataShiftArray(k,8, EM,DataShif ) ;
    for(int i=0;i<8;i++)
    {
        // Conversion du Byte DataShif[i] en binaire sur 8 bits
        int2BitArray( DataShif[i],8 ,Data );
        for(int j=0;j<8;j++)
        {
            // Affichage le Bit Data[j] à la position (i,j) de la matrice (8x8)
            lc.setLed(0,i,j,Data[j]);
        }
    }
}
```



```
        //delay(1);
    }
}
delay(1000);
lc.clearDisplay(0);
}

}

void int2BitArray(unsigned char dataIn,unsigned char taille, byte *dataArray)
{
    int i=0;

    for(i=0;i<taille;i++)
        dataArray[taille-i-1]= (dataIn>>i)&0x01;
}

void DataShiftArray(unsigned char NumShift,unsigned char taille, byte *DataIn, byte
*DataOutShift)
{
    int i=0;

    for(i=0;i<taille;i++)
        DataOutShift[i]= (DataIn[i]>>NumShift);
}

void loop() {
    writeArduinoOnMatrix();
}
```

Fichiers projets + Librairie en C du projet
électronique (.rar)



Projet électronique : Gestion d'une matrice des LED avec Arduino

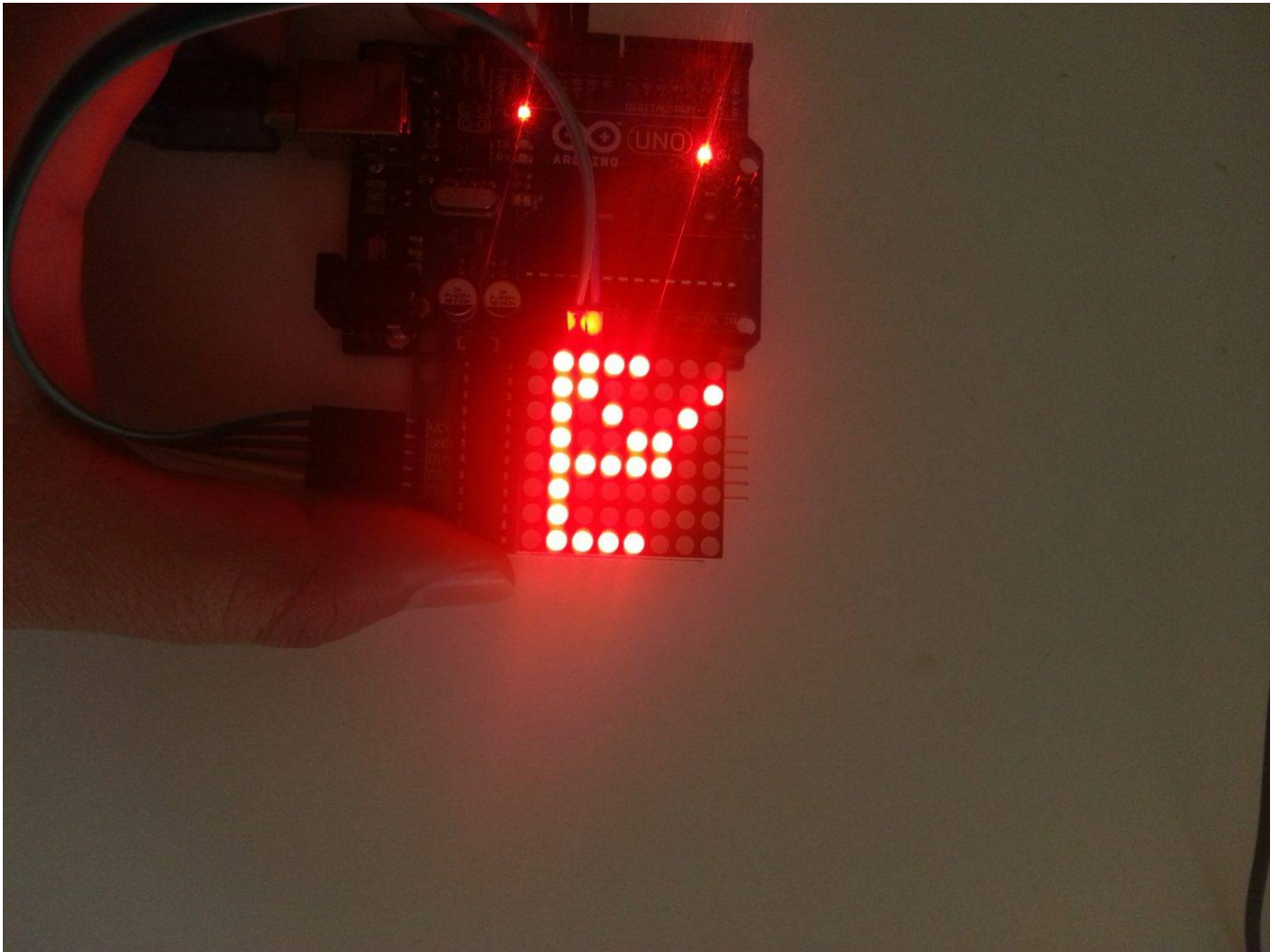
Photos du projet électronique



Projet électronique : Gestion d'une matrice des LED avec Arduino



Projet électronique : Gestion d'une matrice des LED avec Arduino





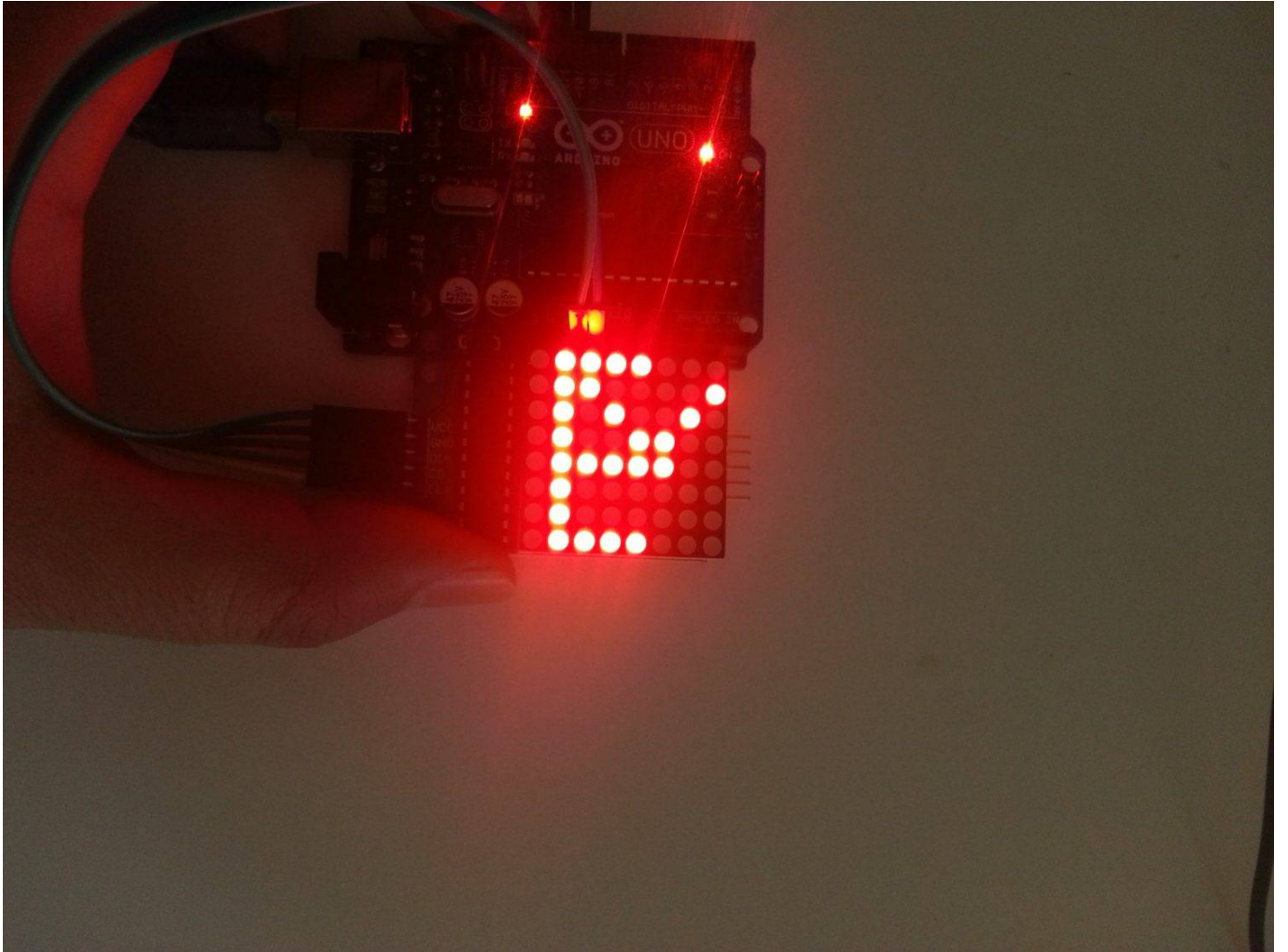
Projet électronique : Gestion d'une matrice des LED avec Arduino



Projet électronique : Gestion d'une matrice des LED avec Arduino



Projet électronique : Gestion d'une matrice des LED avec Arduino





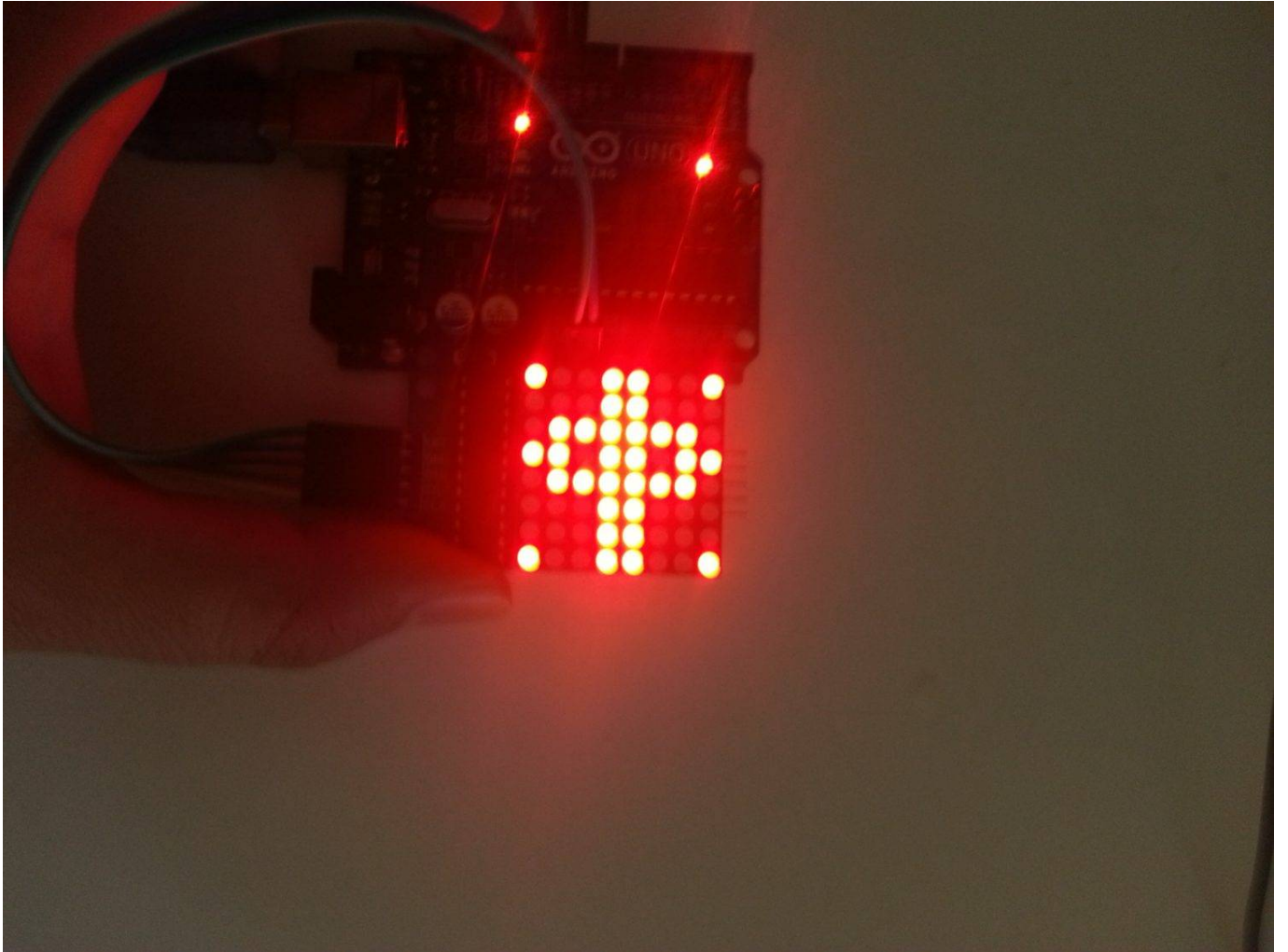
Projet électronique : Gestion d'une matrice des LED avec Arduino



Projet électronique : Gestion d'une matrice des LED avec Arduino



Projet électronique : Gestion d'une matrice des LED avec Arduino





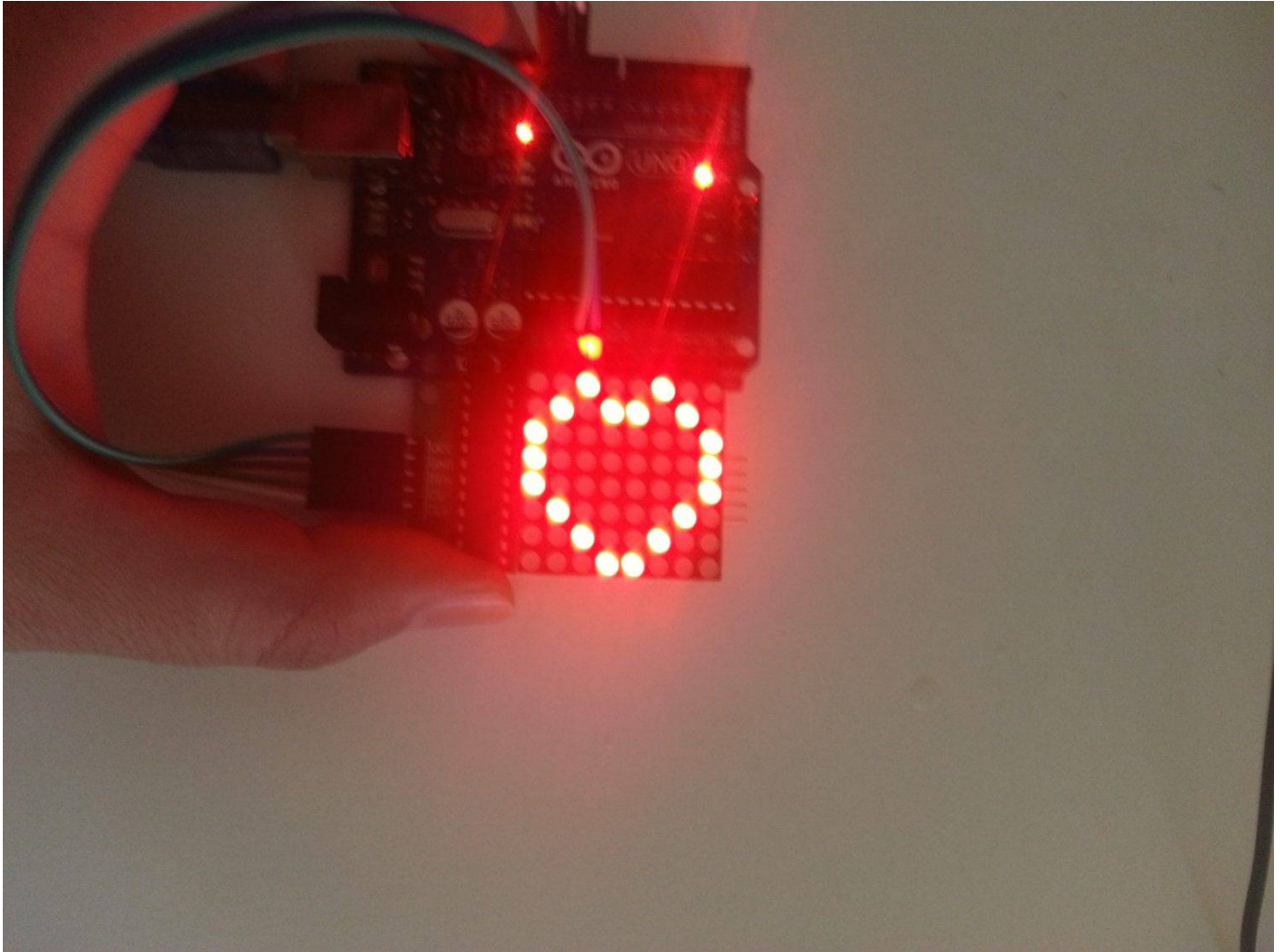
Projet électronique : Gestion d'une matrice des LED avec Arduino



Projet électronique : Gestion d'une matrice des LED avec Arduino



Projet électronique : Gestion d'une matrice des LED avec Arduino





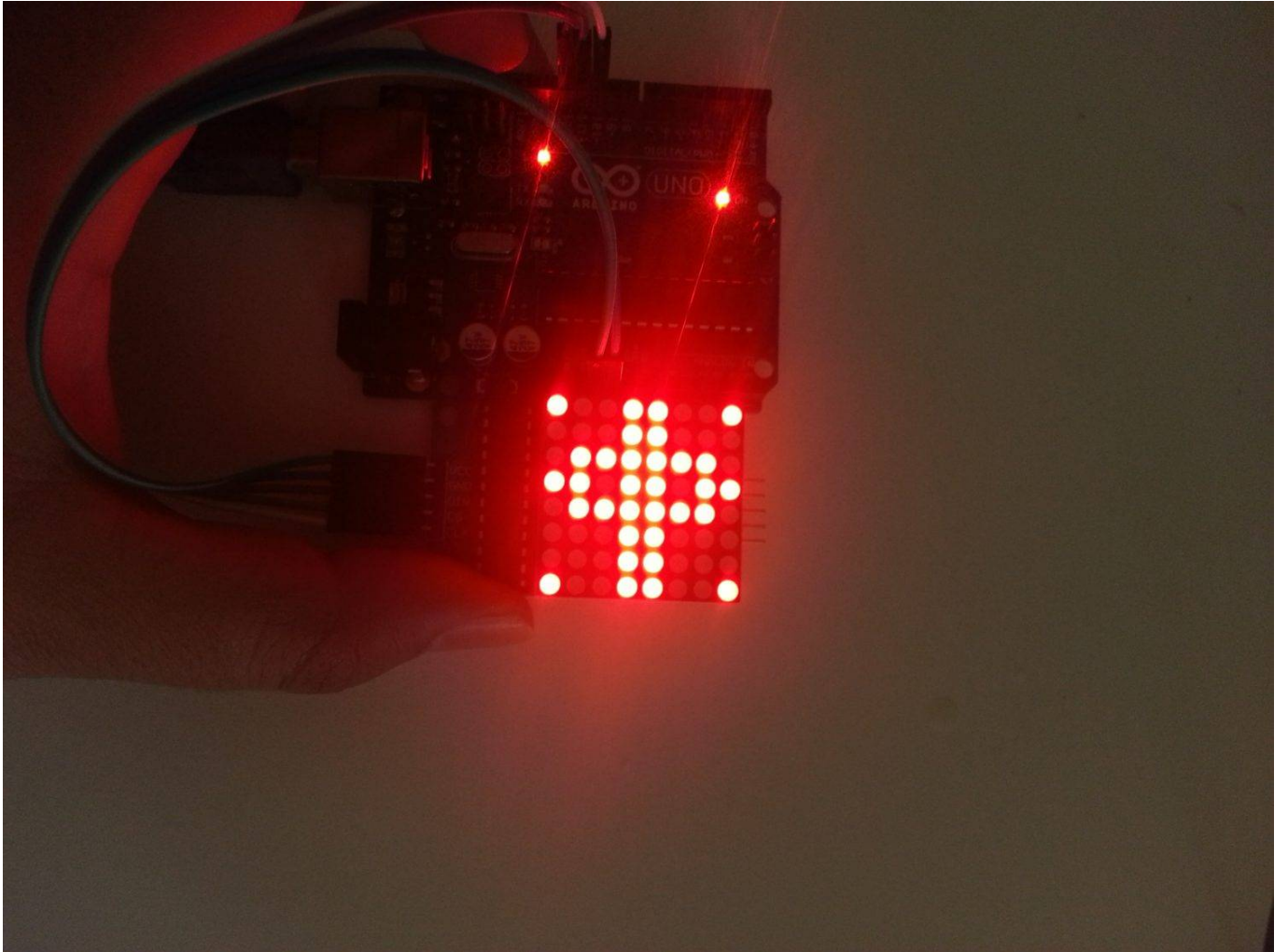
Projet électronique : Gestion d'une matrice des LED avec Arduino



Projet électronique : Gestion d'une matrice des LED avec Arduino



Projet électronique : Gestion d'une matrice des LED avec Arduino





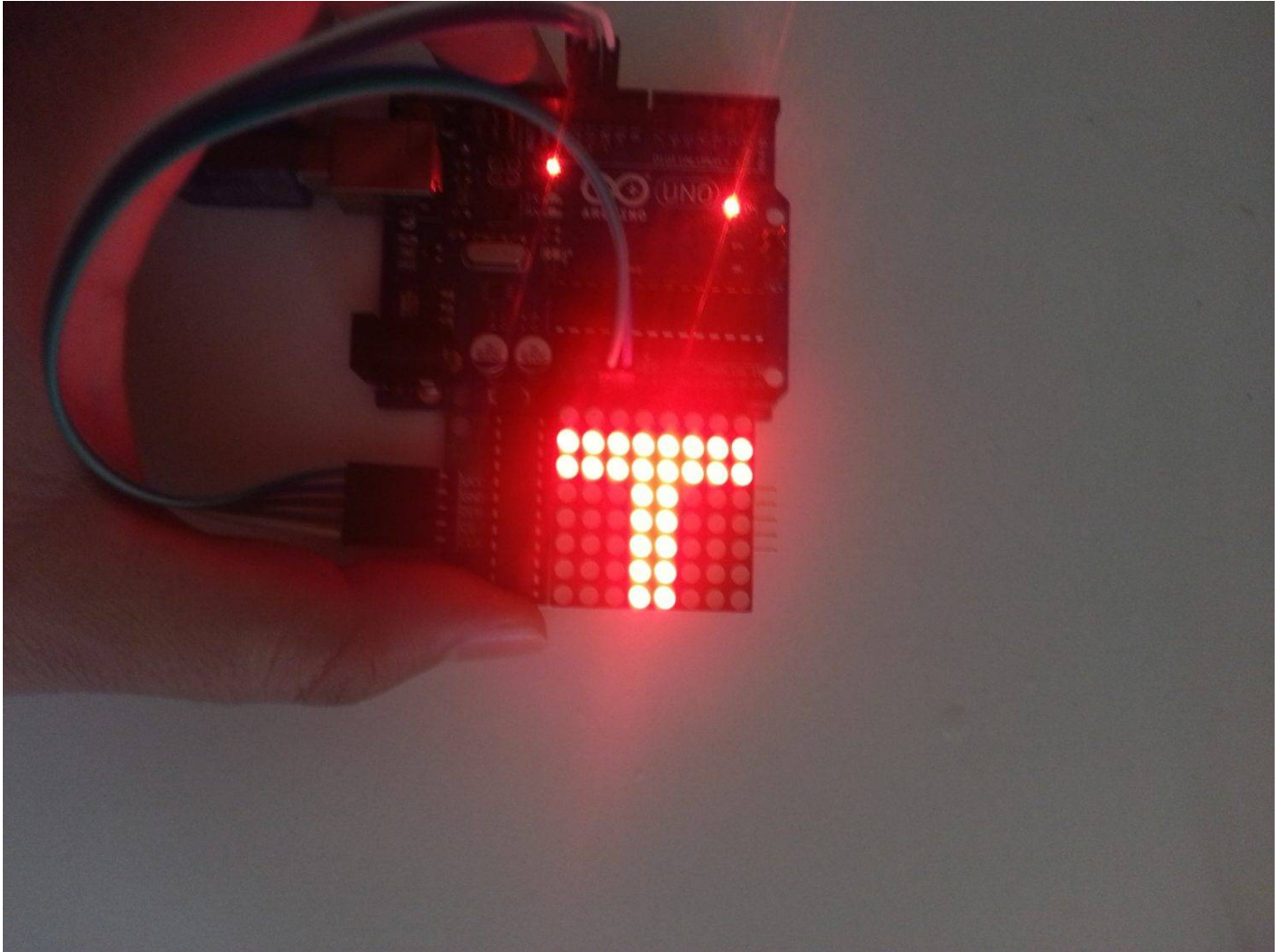
Projet électronique : Gestion d'une matrice des LED avec Arduino



Projet électronique : Gestion d'une matrice des LED avec Arduino



Projet électronique : Gestion d'une matrice des LED avec Arduino





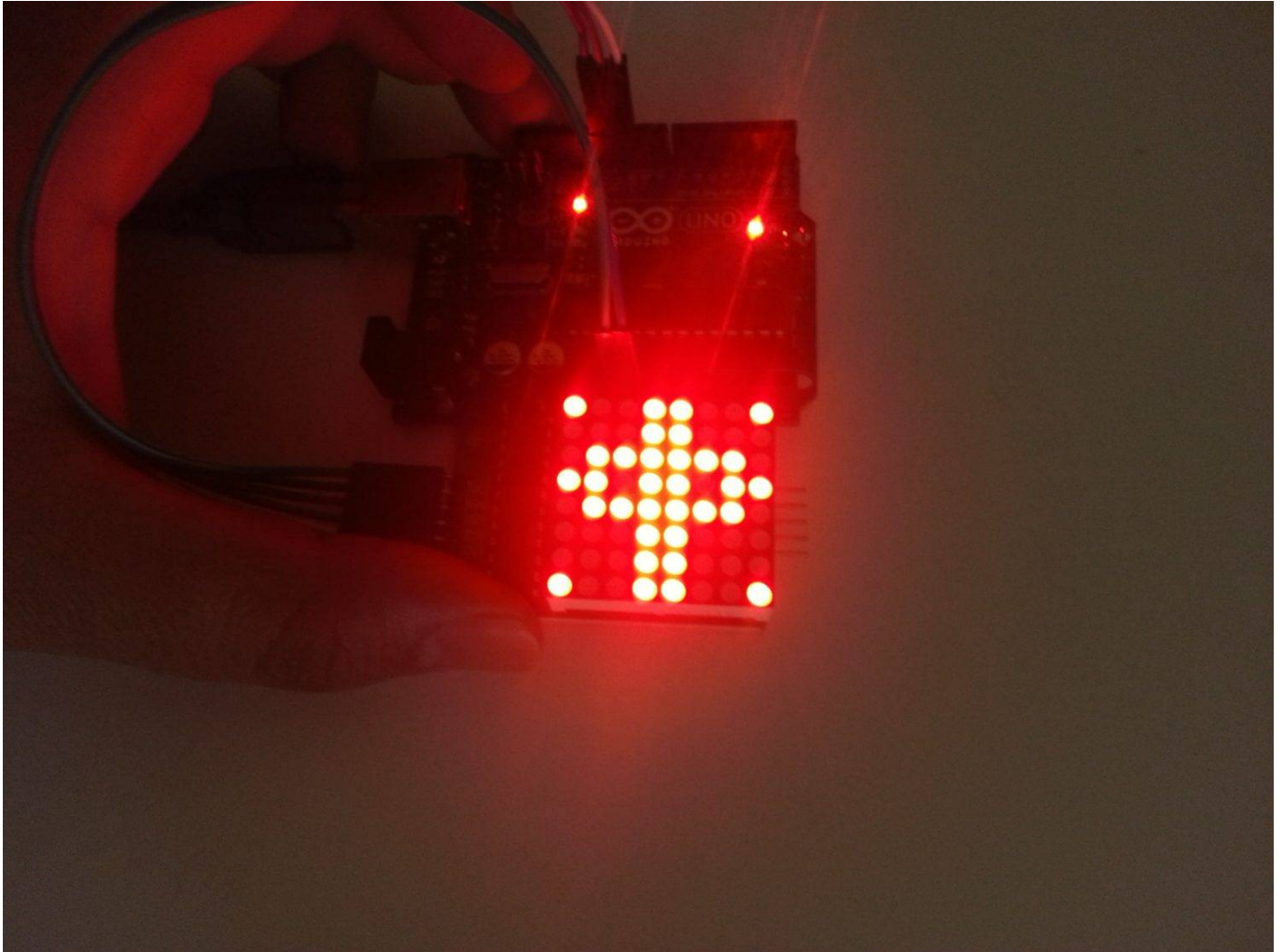
Projet électronique : Gestion d'une matrice des LED avec Arduino



Projet électronique : Gestion d'une matrice des LED avec Arduino

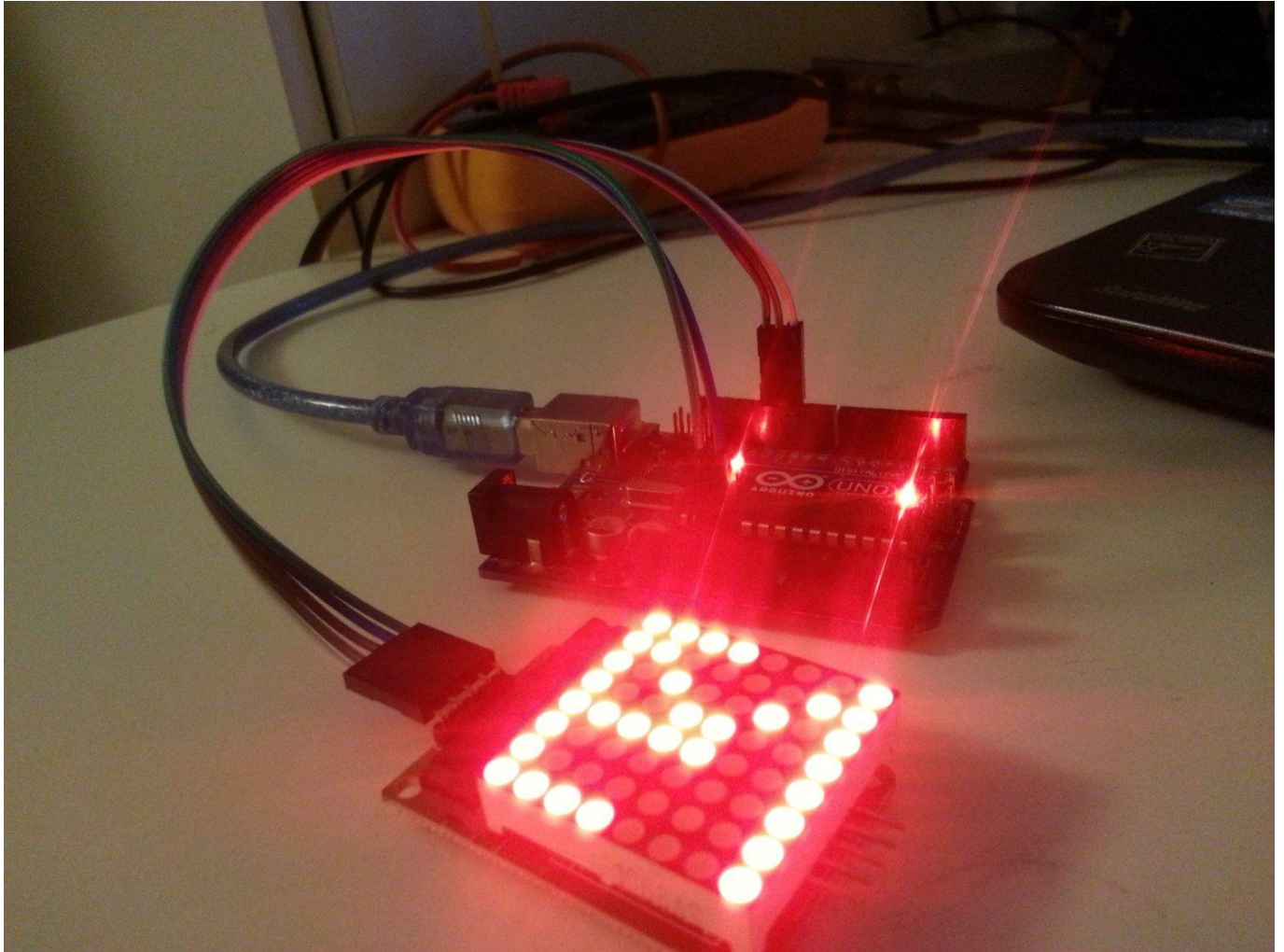


Projet électronique : Gestion d'une matrice des LED avec Arduino





Projet électronique : Gestion d'une matrice des LED avec Arduino



Rappel des cours:



Cours Arduino

Composants électroniques



Un petit commentaire de vous, un
Grand encouragement pour nous 😊

- Bon Courage -

[Total : 2 Moyenne : 5/5]