ELE 1001 : Fiche 2 d'Exercices formatif : Prise en main de la carte Arduino UNO R4 minima

Ces exercices ne seront pas notés, mais il s'agit néanmoins d'une activité obligatoire. La réalisation de ces exercices vous permettra de mieux prendre en main la troisième et dernière partie de votre projet en vous familiarisant au microcontrôleur à votre disposition. Vous devez compléter les trois exercices et enregistrer vos résultats.

Table des matières

1. Prise en main de la carte Arduino UNO

1.1. La carte Arduino Uno

1.1.1. Description de la carte

La carte Arduino est un microcontrôleur, c'est-à-dire une sorte de mini-ordinateur qui sert d'interface entre l'environnement (actions, mesures de grandeurs, etc.) et un utilisateur. La carte Arduino se programme nativement en langage C. Mais on peut utiliser d'autres langages de programmation comme Python pour la programmer. Toutefois cela nécessitera une configuration de la carte en installant sur celle-ci un Firmware (logiciel embarqué ou micrologiciel) différent pour qu'elle puisse communiquer en Python. Ainsi dans cet exercice formatif nous utiliserons les exemples codés en C++ fournis dans l'environnement Arduino. Une description de la carte Arduino est montrée sur la **Figure 1**.

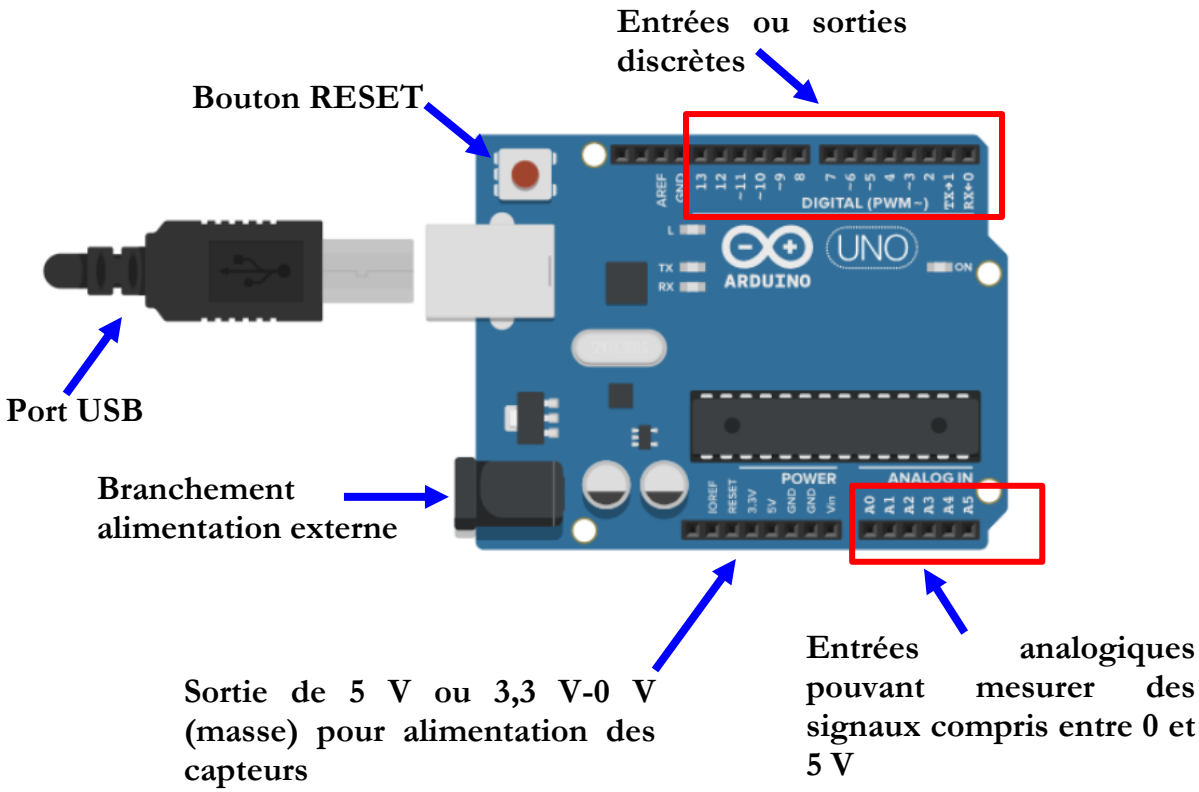


Figure 1. Description de la carte Arduino uno

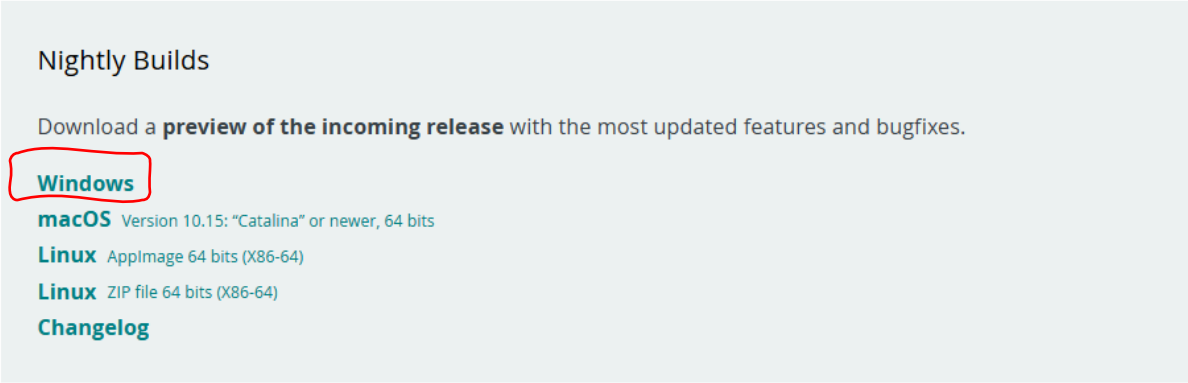
Le logiciel Arduino

En plus de ces composants matériels, vous devez installer le logiciel Arduino. Arduino a été conçu pour vous permettre de programmer les cartes avec peu de difficultés. l'IDE (environnement de développement intégré) d'Arduino est facilement installable sur votre ordinateur.

Figure 2. Icône de l'IDE Arduino

Suivre les étapes ci-dessous pour son installation.

- Cliquer sur Installer le logiciel arduino.
- Choisir Windows 10



- Au besoin, regardez la vidéo suivante : lien d'installation du logiciel arduino.

L'IDE d'Arduino est fourni avec plusieurs exemples (sketch) que vous pouvez utiliser pour apprendre les bases d'Arduino. Un **sketch** est le terme utilisé pour désigner un programme que vous pouvez télécharger sur une carte. Comme l'Arduino Uno n'a pas d'écran connecté, vous aurez besoin d'un moyen de voir la sortie physique de votre programme.

2. Exercice 1 : Clignotement de la DEL intégrée dans la carte Arduino

Pour cet exercice, vous utiliserez le code d'exemple Blink pour faire clignoter une DEL (diode électroluminescente) intégrée à la carte Arduino.

Étape 1

- Le code de l'exemple Blink sera chargé *plus tard* dans une nouvelle fenêtre de l'IDE. Pour commencer, connectez la carte Arduino à votre PC à l'aide d'un câble USB et démarrez l'IDE Arduino. Mais avant de pouvoir télécharger le sketch sur la carte, vous devez configurer l'IDE en sélectionnant votre carte et son port connecté. Pour configurer la carte, accédez au menu **Outils et Tableau (Tools and Board)**. Pour l'Arduino Uno, vous devez sélectionner Arduino UNO R4 board et Arduino Uno R4 minima comme montré sur la **Figure 3**.

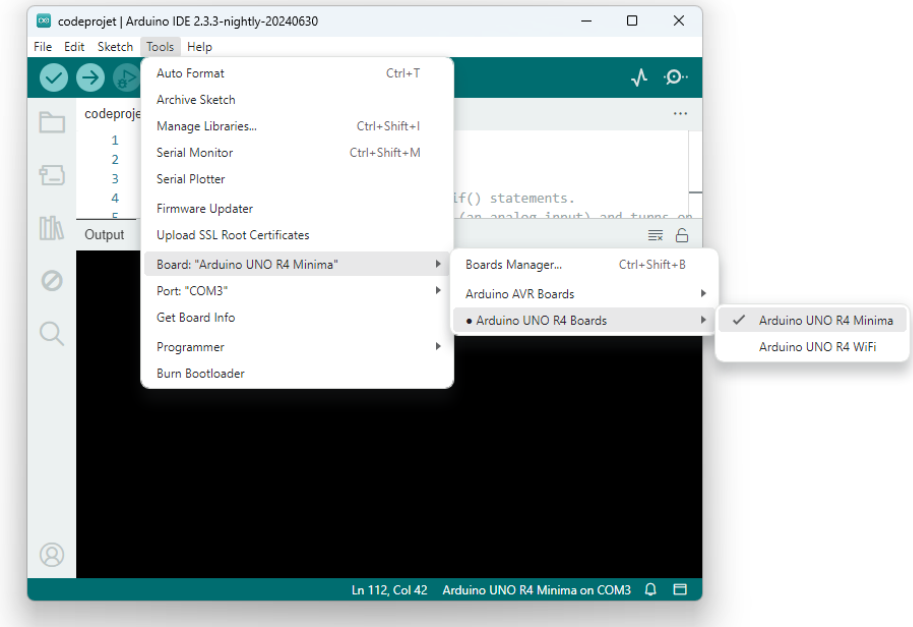


Figure 3. Sélection de la carte Arduino Uno R4 minima

Étape 2

Après avoir sélectionné la carte, vous devez définir le port approprié. Accédez à nouveau au menu Outils et sélectionnez cette fois **Port**. Sous Windows, les ports seront nommés COM3, COM4, COM5, etc., comme montré sur la **Figure 4**.

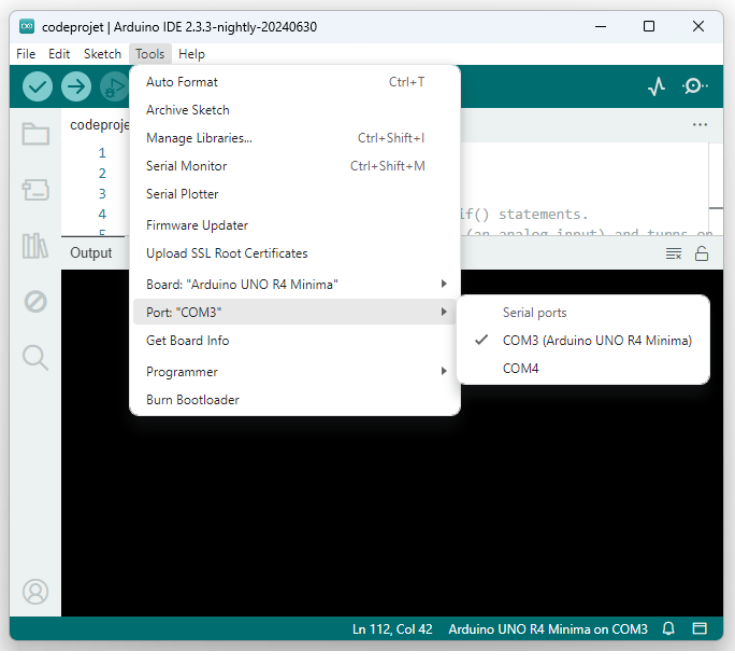


Figure 4. Sélection du port.

Étape 3

- Pour ouvrir le sketch d'exemple Blink, accédez au **menu File** et sélectionnez **Examples, puis 01.Basics et, enfin, Blink** comme montré sur la **Figure 5**.

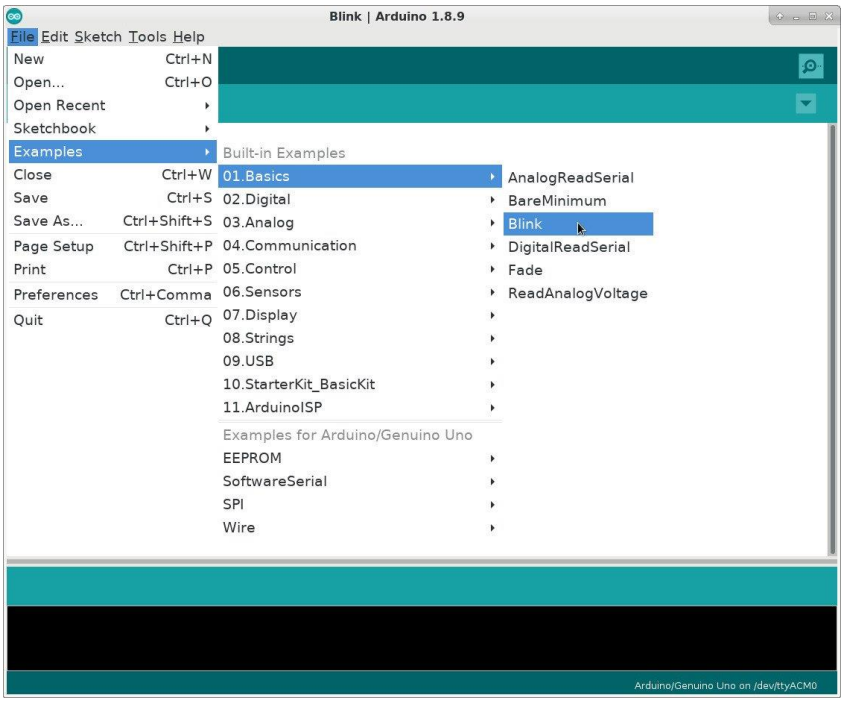


Figure 5. Ouverture du programme pour faire clignoter une DEL

- Après avoir configuré la carte et le port et ouvert l'exemple, vous êtes prêt à télécharger le sketch sur votre Arduino. Pour ce faire, il vous suffit d'appuyer sur le bouton Upload (téléversement) dans la barre d'outils de l'IDE comme montré sur la **Figure 6**.

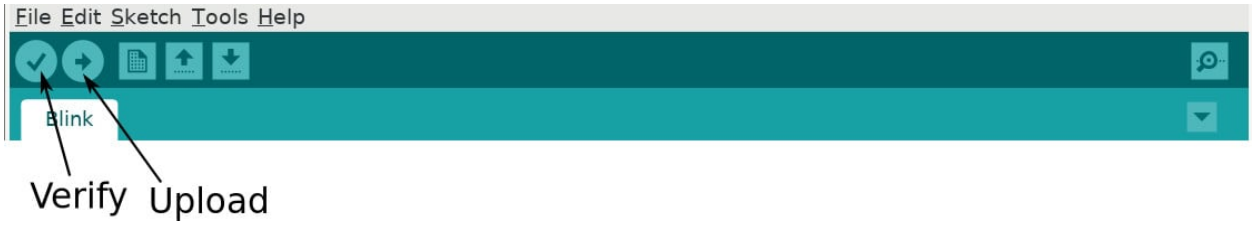


Figure 6. Téléchargement du code dans la carte Arduino.

Lorsque vous appuyez sur Upload, l'IDE compile le sketch et le télécharge sur votre carte. Si vous voulez vérifier les erreurs, vous pouvez appuyer sur "Verify" avant "Upload", ce qui compilera seulement votre esquisse.

Le câble USB fournit une connexion série pour télécharger le programme et alimenter la carte Arduino. Pendant le téléchargement, vous verrez des DEL clignoter sur la carte. Après quelques secondes, le programme téléchargé s'exécutera et vous verrez une DEL clignoter une fois par seconde.

Une fois le téléchargement terminé, le câble USB continue d'alimenter la carte Arduino. Le programme est stocké dans la mémoire flash du microcontrôleur Arduino. Vous pouvez également utiliser une batterie ou une autre source d'alimentation externe pour faire fonctionner l'application sans câble USB.

3. Exercice 2 : Lecture des touches du clavier 4 * 4

3.1. Montage à réaliser

Déconnecter de l'ordinateur la carte Arduino et faire le montage de la **Figure 7**.

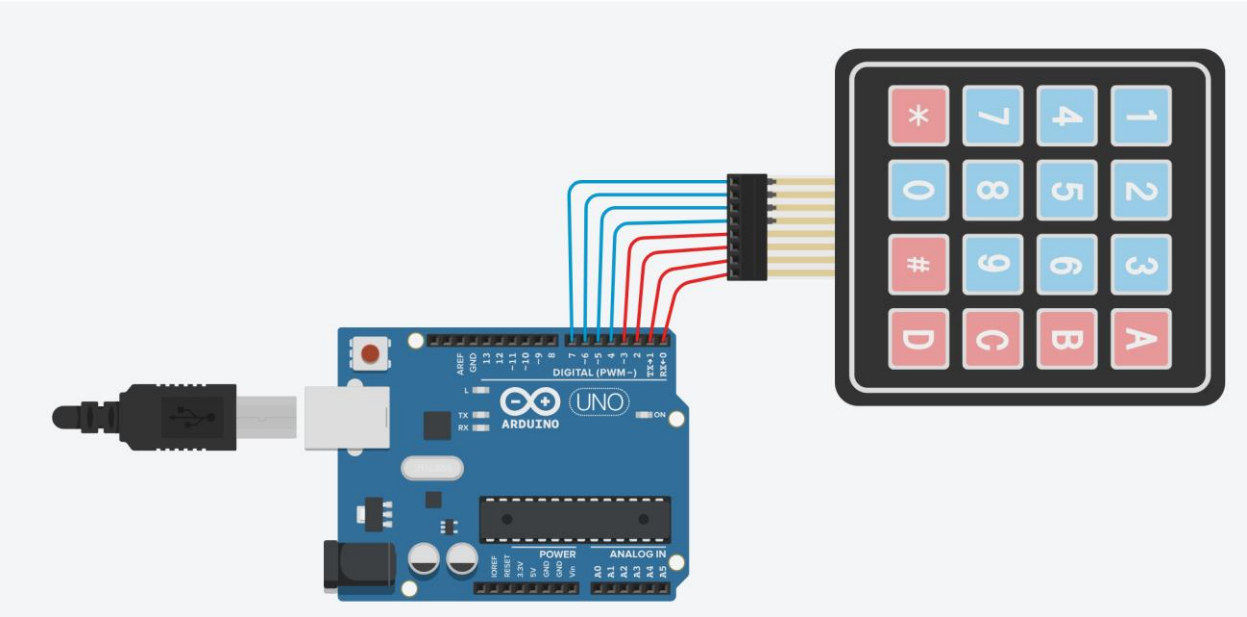


Figure 7. Montage à réaliser pour lire les touches du clavier.

3.2. Conception du code

Pour détecter une touche enfoncée, le microcontrôleur met au ground (0V) toutes les lignes en fournissant 0 aux broches sur lesquelles sont raccordées les lignes. Les colonnes quant-à-elle sont connectées au V_{CC}. Les colonnes sont ainsi continuellement scannées.

Si les données lues dans les colonnes sont égales à 1111, aucune touche n'a été enfoncée, le scan se poursuivra jusqu'à ce qu'une pression soit détectée. Après avoir identifié la ligne où la touche a été enfoncée, il devient alors facile de déterminer la valeur de la touche par ligne et par colonne.

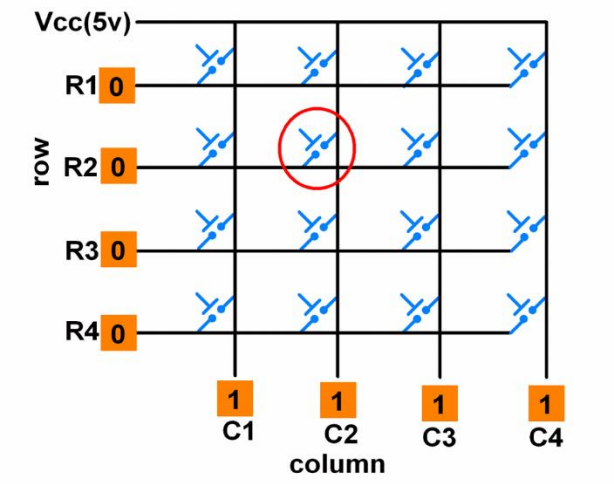


Figure 8. Image pour illustrer la conception du code. Source

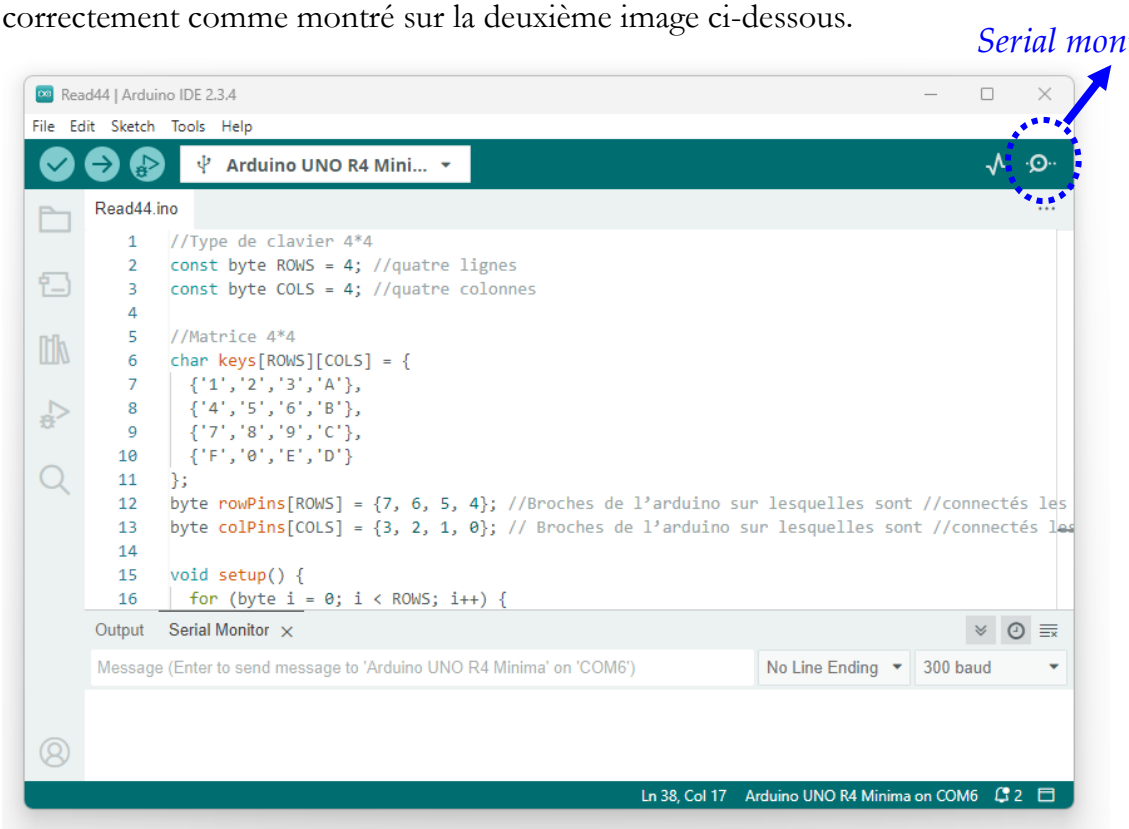
3.3. Code à utiliser (faire un copier-coller du code)

```
//Type de clavier 4*4

const byte ROWS = 4; //quatre lignes
const byte COLS = 4; //quatre colonnes

//Matrice 4*4

char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
byte rowPins[ROWS] = {7, 6, 5, 4}; //Broches de l'arduino sur lesquelles sont //connectés les lignes
byte colPins[COLS] = {3, 2, 1, 0}; // Broches de l'arduino sur lesquelles sont //connectés les colonnes

void setup() {
  for (byte i = 0; i < ROWS; i++) {
    pinMode(rowPins[i], INPUT_PULLUP);
  }
  for (byte i = 0; i < COLS; i++) {
    pinMode(colPins[i], OUTPUT);
  }
  Serial.begin(9600); //initialize serial communication
}

void loop() {
  char key = getKey();
  if (key) {
    Serial.println(key);
  }
}
```

```
char getKey() {
  for (byte c = 0; c < COLS; c++) {
    digitalWrite(colPins[c], LOW);
    for (byte r = 0; r < ROWS; r++) {
      if (digitalRead(rowPins[r]) == LOW) {
        digitalWrite(colPins[c], HIGH);
        delay(50);
        return keys[r][c];
      }
    }
    digitalWrite(colPins[c], HIGH);
  }
  return 0;
}
```

3.4. Visualisation des résultats

Après avoir vérifié et téléversé le code dans la carte arduino, vous pouvez visualiser la touche pressée du clavier sur le serial monitor. Vérifier que toutes les touches fonctionnent correctement comme montré sur la deuxième image ci-dessous.



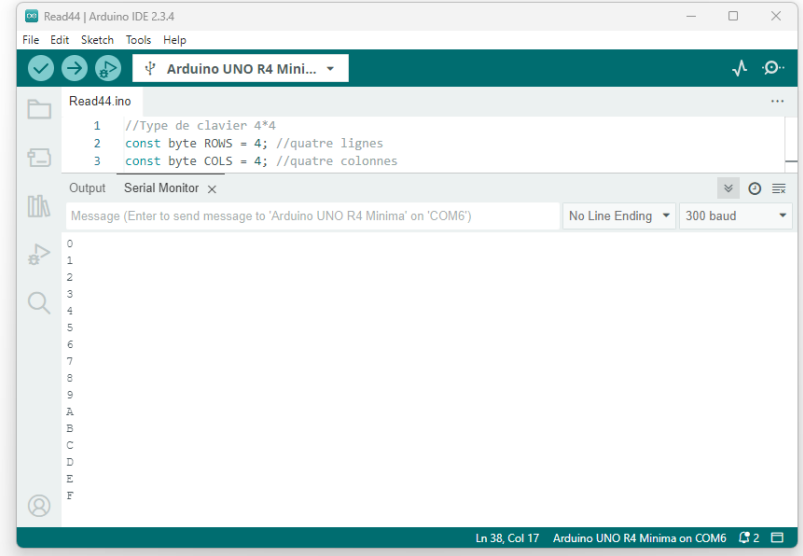


Figure 9. Montage à réaliser pour lire les touches du clavier.

4. Exercice 3 : Codage des touches du clavier

4.1. Montage à réaliser et pins à utiliser.

Pour cette partie, il sera question de concevoir quatre signaux dénommés comme suit : U3, U2, U1 et U0 traduisant le codage binaire de la touche actionnée du clavier.

⚠ Déconnecter de l'ordinateur la carte Arduino et faire le montage de la **Figure 10**.

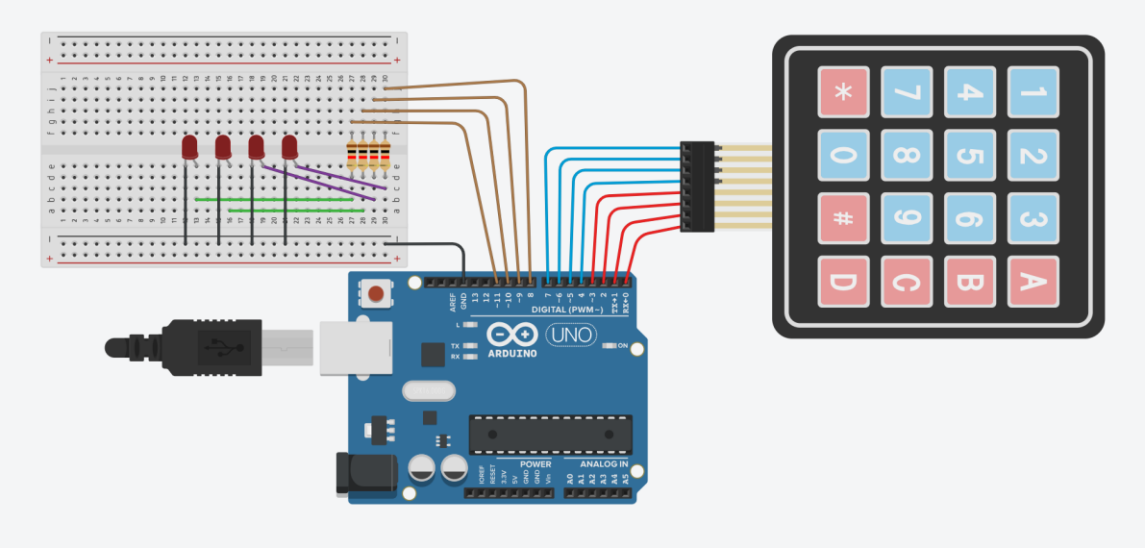


Figure 10. Montage à réaliser pour générer les signaux Z, D, U_3, U_2, U_1 et U_0.

- La **broche 11** de l'arduino fournira le signal $\mathbf{U_3}$.
- La **broche 10** de l'arduino fournira le signal $\mathbf{U_2}$.
- La **broche 9** de l'arduino fournira le signal $\mathbf{U_1}$.
- La **broche 8** fournira le signal $\mathbf{U_0}$.

Ci-dessous la table de vérité de ces signaux.

Touche du clavier	$\mathbf{U_3}$	$\mathbf{U_2}$	$\mathbf{U_1}$	$\mathbf{U_0}$
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
# (E ou14)	1	1	1	0
*(F ou 15)	1	1	1	1

À partir de cette table de vérité, on peut écrire les équations logiques suivantes :

$$\begin{cases} U_0 = 1 + 3 + 5 + 7 + 9 + F \\ U_1 = 2 + 3 + 6 + 7 + E + F \\ U_2 = 4 + 5 + 6 + 7 + E + F \\ U_3 = 8 + 9 + E + F \end{cases}$$

4.2. Code à téléverser

Téléverser le code ci-dessous dans la carte arduino et vérifier que les DEL appropriées sont à ON ou à OFF selon les exigences de la table de vérité ci-dessus.

```
//Type de clavier 4*4
const byte ROWS = 4; //quatre lignes
const byte COLS = 4; //quatre colonnes

//  constants pour les signaux
const int U3Pin = 11;      // Signal U3 sur la PIN 11
const int U2Pin = 10;      // Signal U2 sur la PIN 10
const int U1Pin = 9;      // Signal U1 sur la PIN 9
const int U0Pin = 8;      // Signal U0 sur la PIN 8

//Matrice 4*4
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'F','0','E','D'}
};
byte rowPins[ROWS] = {7, 6, 5, 4}; //Broches de l'arduino sur lesquelles sont
//connectés les lignes
byte colPins[COLS] = {3, 2, 1, 0}; // Broches de l'arduino sur lesquelles sont
//connectés les colonnes

void setup() {
  for (byte i = 0; i < ROWS; i++) {
    pinMode(rowPins[i], INPUT_PULLUP);
  }
  for (byte i = 0; i < COLS; i++) {
    pinMode(colPins[i], OUTPUT);
  }
    // initialize the LED pin as an output:
  pinMode(U3Pin, OUTPUT);
  pinMode(U2Pin, OUTPUT);
  pinMode(U1Pin, OUTPUT);
  pinMode(U0Pin, OUTPUT);
  Serial.begin(9600); //initialize serial communication
}

void loop() {
  char key = getKey();
  if ((key=='8')||(key=='9')||(key=='E')||(key=='F')) {
    Serial.println(key);
    digitalWrite(U3Pin, HIGH);
  } else {
    digitalWrite(U3Pin, LOW);
  }
```

```
  if ((key=='1')||(key=='3')||(key=='5')||(key=='7')||(key=='9')||(key=='F')) {
    Serial.println(key);
    digitalWrite(U0Pin, HIGH);
  } else {
    digitalWrite(U0Pin, LOW);
  }
  if ((key=='2')||(key=='3')||(key=='6')||(key=='7')||(key=='E')||(key=='F')) {
    Serial.println(key);
    digitalWrite(U1Pin, HIGH);
  } else {
    digitalWrite(U1Pin, LOW);
  }
  if ((key=='4')||(key=='5')||(key=='6')||(key=='7')||(key=='E')||(key=='F')) {
    Serial.println(key);
    digitalWrite(U2Pin, HIGH);
  } else {
    digitalWrite(U2Pin, LOW);
  }
}

char getKey() {
  for (byte c = 0; c < COLS; c++) {
    digitalWrite(colPins[c], LOW);
    for (byte r = 0; r < ROWS; r++) {
      if (digitalRead(rowPins[r]) == LOW) {
        digitalWrite(colPins[c], HIGH);
        delay(500);
        return keys[r][c];
      }
    }
    digitalWrite(colPins[c], HIGH);
  }
  return 0;
}
```

Fin de la fiche ici ! vous pouvez procéder à la troisième partie de votre projet