TD 2 : Commande d'un clavier matriciel

Systèmes Embarqués
ING2, EILCO
P.-A. Hébert
2023–2024

Consignes

Ce second TD a les objectifs suivants :

— utiliser les *ports d'entrées/sorties* (E/S) de l'*Atmega328p* ;
— piloter un clavier matriciel.

1 Présentation du problème proposé

1.1 Cahier des charges

On souhaite déclencher le chenillard précédemment réalisé, non plus sur une pression de bouton, mais suite à la saisie correcte d'un code à 4 chiffres sur un clavier matriciel.

1.2 Clavier matriciel

1.2.1 Rôle des broches du clavier

Le clavier matriciel est un composant passif muni d'un connecteur à 7 broches :

— 3 broches destinées à sélectionner la *colonne* du clavier à tester (1 broche par colonne, de gauche à droite) ; les colonnes doivent en effet être testées une par une (cf. annexe ci-dessous) ;
— 4 broches destinées à tester l'état des boutons sités sur la colonne sélectionnée (1 broche par *ligne*, de haut en bas).

1.2.2 Connexion du clavier matriciel à l'*Atmega328*

Afin qu'on puisse centraliser la correction, il est important de connecter le clavier à l'*Arduino* de la même manière, suivant le schéma de la figure 1 (même si ce choix est arbitraire).

— les chiffres du bas correspondent aux numéros des broches de l'*Arduino Uno* ;
— en particulier :
  — la broche à l'extrémité gauche du connecteur du clavier, doit être reliée à la broche 11 de l'*Arduino* ; c'est celle qui contrôle la ligne supérieure du clavier (caractères 1, 2 et 3) ;
  — la broche opposée du clavier doit être reliée à la broche 2 de l'*Arduino*, et elle pilote la colonne de gauche (caractères 3, 6, 9 et #).

2 Exercice

2.1 Question 1

Écrire une fonction destinée à configurer les broches de l'*Arduino Uno* connectées au clavier matriciel : les broches liées aux *colonnes* du clavier doivent être programmées en **sortie**, celles liées aux *lignes* en **entrée**.

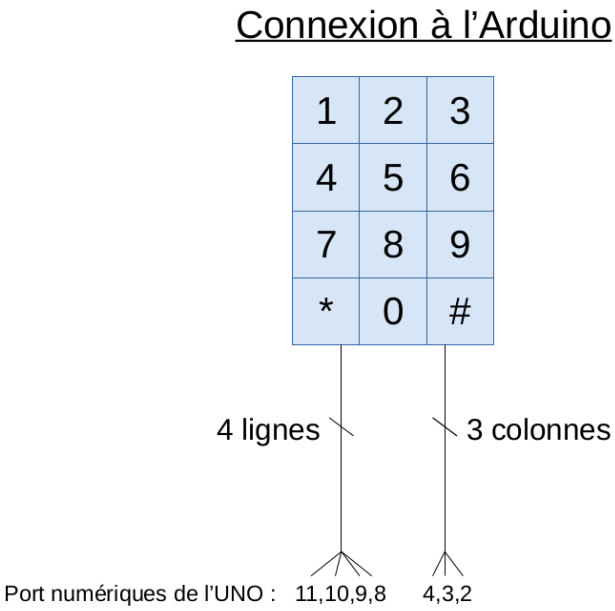

FIGURE 1 – Connexion du clavier matriciel à l'*Atmega328*

Sur les lignes en entrées, vous activerez la `résistance de tirage` (*pull-up*).

2.2 Question 2

Écrire une fonction qui teste l'état du bouton 1.

2.3 Question 3

Écrire une fonction qui détecte si une touche est pressée, renvoie son caractère si c'est le cas, et 0 sinon. *Attention, il ne s'agit pas du code ASCII du caractère « 0 » (soit '0'), mais de l'entier 0.*

Vous pourrez compléter ce début de code :

```
const byte ROWS = 4; //4 colonnes
const byte COLS = 3; //3 colonnes

const byte rowPins[ROWS] = {11,10,9,8};//entrées PORTB
const byte colPins[COLS] = {4,3,2};   //sorties PORTDã

const char keys[ROWS][COLS] = {
  {'1','2','3'},
  {'4','5','6'},
  {'7','8','9'},
  {'*','0','#'}
};
```

```
void setup() {
  Serial.begin(9600);
}

void loop() { }
```

3 Questions supplémentaires, à traiter en séance machine

3.1 Question 4

Écrire une fonction similaire, mais qui détecte seulement les « nouveaux » appuis : si une touche est pressée de manière prolongée, cette fonction ne la signalera qu'au premier appel : elle retournera 0 aux suivants, pour indiquer qu'il n'y a pas de nouvel appui. Il faudra attendre le relâchement de la touche pour qu'elle puisse à nouveau la détecter.

3.2 Question 5

Programmer l'arduino de manière à contrôler le chenillard de la manière suivante :
— touche 1 : arrêt ;
— touche 2 : marche.

3.3 Question 6

Programmer l'Arduino pour activer le chenillard sitôt qu'une combinaison prédéfinie de 4 chiffres (un « code secret ») est pressée **consécutivement**.

Exemple : la saisie de *1234* lancera le chenillard si le code est bien *1234* ; de même que la saisie de *339792834****91234*** ; mais pas ***123****763234*.

3.4 Question 7

Programmer l'Arduino pour effectuer un nombre précis de va-et-vients. Ce nombre entre 0 et 99 devra être saisi au clavier matriciel. Il devra être entré juste après le code d'activation de la question précédente.

4 ANNEXE : fonctionnement des claviers matriciels

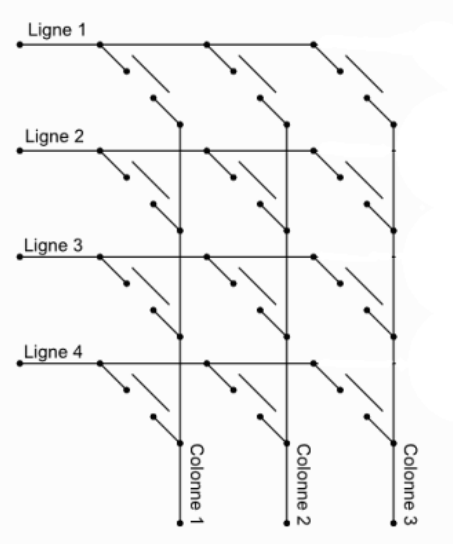

FIGURE 2 – Schéma du clavier matriciel

Le schéma de la figure 2 révèle la structure du clavier :

— il est composé de 7 lignes électriques (accessibles sur la nappe du clavier), initialement sans contact entre elles ;

— 3 correspondent au *colonnes* ; ce sont des *entrées* (du point de vue du clavier ! c'est l'inverse pour l'*Arduino*) : elles peuvent être mises à la masse (0V) - chacune leur tour - pour connaître l'état des 4 touches qui la composent ; l'activation d'une *colonne* (une seule à la fois) se fait par une mise à 0V (car les *lignes* sont par défaut sous 5V, grâce aux résistances pull-up) ;

— les 4 autres correspondent aux *lignes* ; ce sont des *sorties* du clavier : la tension d'une *ligne* rend compte de la pression du bouton situé à l'intersection de cette *ligne* et de l'unique *colonne* testée (tension 0V) : si sa tension passe de +5V (tension par défaut due à la résistance pull-up) à 0V, c'est qu'un contact a été établi (pression du bouton) entre cette *ligne* et la *colonne* active (celle à la masse) : la résistance pull-up est court-circuitée.

Autrement dit, à l'intersection de chaque *colonne* c et *ligne* l, se trouve un bouton qui permet de faire contact entre elles. Quand on le presse, la tension de la colonne c se propage jusque l. Par défaut, toutes les tensions des *lignes* sont à +5V (résistance pull-up). Donc si on mesure 0V sur une *ligne* particulière l, c'est forcément qu'un des ses 3 boutons est pressé et mis à la masse par l'une des *colonnes*. Et comme on prend soin de n'activer qu'une seule *colonne* à la fois, il est aisé d'identifier lequel des 3 boutons est pressé.

La détection d'une pression de touche s'opère donc en balayant/testant tour à tour chacune des *colonnes*, et en leur appliquant un niveau de tension 0V opposé à la fois à celui des autres *colonnes*, et au niveau de tension initial des *lignes* (qui est déterminé par des résistances de tirage : *pull-up*).