Chapitre 19 - Le bouton poussoir dans tous ses états

Nous avons vu au chapitre précédent que la façon la plus simple d'utiliser une entrée est de la relier à un bouton poussoir (B/P). Avec Arduino, on peut utiliser toute forme de B/P soit pour déclencher une action, soit comme capteur. Nous allons voir dans ce chapitre tout ce qui concerne les B/P.

Qu'est-ce qu'un B/P ?

Un B/P n'est rien d'autre que deux contacts séparés qui deviennent réunis sous l'action d'un mécanisme, ce qui permet la continuité électrique entre les deux contacts. Ce peut aussi être l'inverse ; en temps normal les deux contacts sont réunis et deviennent séparés sous l'action d'un mécanisme. Il y a donc des **B/P normalement ouverts** et des **B/P normalement fermés**. L'action d'un mécanisme est le plus souvent le doigt qui appuie, mais ce peut être aussi autre chose, comme un aimant qui va refermer les deux lames d'un ILS (Interrupteur à Lames Souples). Lorsque cette action mécanique se termine, un ressort permet de revenir à la situation normale (contacts normalement ouverts ou normalement fermés).

Bref, tout le monde pense savoir ce qu'est un B/P parce qu'on en rencontre souvent, comme celui qui permet de sonner à la porte d'une maison. Mais savez-vous exactement ce qui se passe lorsqu'on appuie sur un B/P ?

La figure 19.1 montre le signal obtenu lorsqu'on appuie sur un B/P ; la réunion des deux contacts devrait instantanément laisser passer le courant et on devrait observer un signal comme celui donné à gauche de la figure. Il n'en est rien, en fait le signal obtenu montre des rebonds, comme celui de la partie droite de la figure, qui durent généralement quelques dizaines de ms.

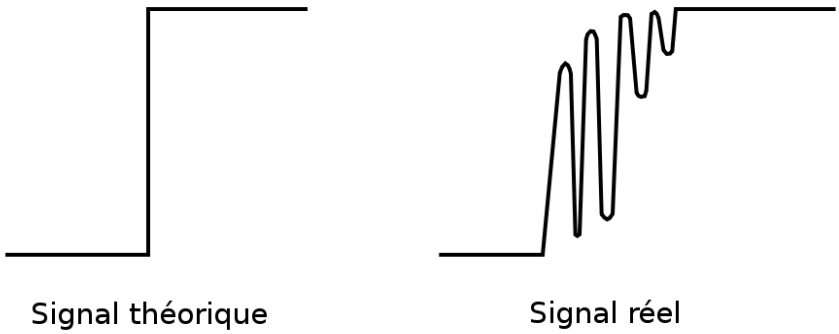

Figure 19. 1

Le microcontrôleur travaillant très vite n'est pas certain de voir le signal passer d'un état à l'autre (LOW vers HIGH ou bien HIGH vers LOW) s'il observe le poussoir au niveau des rebonds et peut ainsi louper une commande. Il y a bien sûr des solutions à cela et elles peuvent être de deux genres : électronique (un trigger de Schmitt par exemple, ce qui rajoute du matériel) ou bien informatique (il suffit d'attendre un peu que les rebonds se stabilisent pour observer s'il y a eu un changement d'état du B/P (l'attente est de l'ordre de 50 ms). Nous verrons la solution logicielle un peu plus loin mais pour l'instant, nous imaginerons que nos B/P sont parfaits et n'ont pas de rebonds.

Comment surveiller un B/P ?

Reportons nous à la figure 18.3 donnée plus haut : elle montre comment relier un B/P (mais c'est pareil pour un interrupteur) à une entrée numérique d'Arduino, grâce à des résistances de pull-up ou de pull-down. Par appui sur le B/P, le signal sur l'entrée numérique passe d'un état à l'autre. En fonction du

montage, l'appui provoquera un signal HIGH ou un signal LOW sur l'entrée numérique. Il suffit donc de surveiller cette entrée (élémentaire mon cher Watson !).

Ouvrez le programme Button donné en exemple :

Fichier > Exemples > 02.Digital > Button

Réalisez le montage donné à cette page :

https://www.arduino.cc/en/Tutorial/Button

Vous avez le plan du câblage à réaliser soit sous forme d'un schéma, soit directement sur une platine d'essai.

Comme vous le voyez, l'appui sur le B/P (ou la fermeture de l'interrupteur) entrainera un signal HIGH sur l'entrée numérique.

Commentons le programme.

Plusieurs lignes de commentaires situées entre /* et */ permettent de rappeler comment câbler Arduino. D'autres lignes de commentaires commencent par // et sont données tout au long du programme. Comme on l'a déjà dit, il est important de commenter ses programmes pour se rappeler de ce qu'on veut faire et pour faciliter la compréhension du programme pour un autre programmeur.

Ensuite, on donne un nom aux entrées-sorties : buttonPin est le nom de la broche sur laquelle est connecté le B/P, et ledPin est le nom de la broche sur laquelle est connectée la LED (ici on utilise celle du module). Le B/P est connecté sur la broche 2 et cela ne change pas au cours du programme ; il s'agit donc d'une **constante entière** d'où le const int buttonPin = 2 ;

Comme on a utilisé la LED du module, qui est reliée à la broche 13, on obtient de la même manière const int ledPin = 13 ;

Enfin, on trouve une vraie variable car la seule chose qui va changer dans ce programme est l'état du bouton (appuyé ou non) ; on appelle cette variable buttonState et on l'initialise à 0.

Ensuite arrive la fonction setup pour initialiser la broche de la LED en sortie et la broche du B/P en entrée ; cela, vous savez déjà le faire donc je ne m'y attarde pas.

La fonction loop commence par lire l'état du poussoir grâce à la fonction digitalRead qui retourne soit LOW soit HIGH. Juste une parenthèse pour dire que LOW est égal à la valeur 0 et HIGH est égal à la valeur 1 ; l'état du bouton (buttonState) est donc une variable booléenne (voir le chapitre 9). Cette valeur retournée par la fonction digitalRead est mise dans la variable buttonState qui prend donc soit la valeur LOW (ou 0), soit la valeur HIGH (ou 1).

Le programme teste ensuite la valeur de buttonState et en fonction de ce qu'il trouve, soit il allume la LED, soit il l'éteint. C'est aussi simple que cela.

La structure est if… else : après le if (qui signifie « si »), on trouve une condition (buttonState == HIGH), remarquez ici le double signe égal. Si cette condition est vérifiée, c'est que le B/P est appuyé, dans ce cas, les instructions qui suivent entre les accolades sont exécutées. Autrement on passe à la suite qui est else (qui signifie « autrement ») et on exécute les autres instructions entre les autres accolades.

Autrement dit, si buttonState est HIGH, alors digitalWrite (ledPin, HIGH) ; allumera la LED, autrement (c'est que buttonState est LOW) digitalWrite (ledPin, LOW) ; éteindra la LED.

Ce programme fonctionne car le microcontrôleur travaille très rapidement et surveille donc le B/P un grand nombre de fois par seconde. Mais si le B/P était appuyé pendant que le microcontrôleur fait autre chose, l'appui ne serait pas forcément remarqué. Pour vous en convaincre, rajouter l'instruction delay (10000) ; à la fin de la fonction loop. Cela force le programme à attendre (delay) 10000 millisecondes (donc 10 secondes) avant de regarder à nouveau l'état du poussoir. Durant ce délai où le microcontrôleur ne fait rien d'autre que d'attendre, on peut appuyer sur le B/P comme un malade sans que la LED ne s'allume !

Intérêt de l'échantillonnage

Quelle idée aussi d'avoir rajouté ce delay qui bloque tout pendant dix secondes ! Tout le monde a compris qu'on a intérêt à lire l'état du B/P le plus souvent qu'il est possible par seconde. J'ai bien peur que dans ce cas-là, on ne puisse pas faire grand-chose d'autre, un peu comme lorsqu'on surveille une casserole de lait sur le feu : c'est quand on s'en éloigne pour faire autre chose que la casserole déborde !

Ce qu'il faut réaliser, c'est une surveillance du B/P (donc effectuer une série de lecture le plus souvent possible) tout en laissant le temps au microcontrôleur pour effectuer d'autres tâches entre deux lectures. Mais sommes-nous certains pour autant qu'on ne loupera pas un appui du B/P pendant que le microcontrôleur fera autre chose ? Non, nous ne sommes pas certains à 100 %, mais cet échantillonnage est pourtant la bonne méthode, à condition de caler sa fréquence sur ce qu'il y a d'autre à effectuer.

Cette méthode de lecture d'une entrée par scrutation s'appelle **polling** en anglais ; elle doit être adaptée à l'application et notamment à la fréquence des actions sur la broche d'entrée.

Intérêt des flancs

Si on fait abstraction du phénomène de rebonds dont on a déjà parlé, l'appui sur un B/P doit provoquer un flanc montant et son relâchement un flanc descendant (si on a monté le B/P comme dans le cas b) de la figure 18.3, sinon on inverse le raisonnement). Dans ce cas, en détectant les flancs d'un signal, on peut obtenir le mouvement du B/P (appui ou relâchement).

Pour cela, on lit l'état du B/P et on le compare à son état précédent. S'il y a changement, c'est que le signal a changé donc on a détecté un flanc (montant ou descendant). Et pour s'affranchir des rebonds, on peut effectuer une nouvelle lecture après une petite attente supérieure à la durée des rebonds.

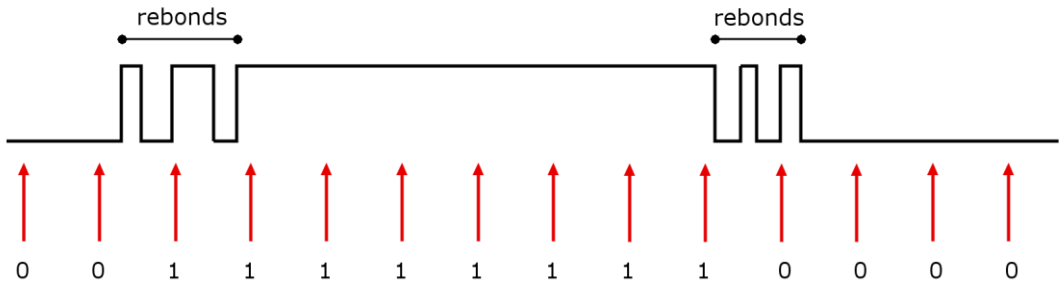

Figure 19. 2

La figure 19.2 montre un exemple d'échantillonnage ; si celui-ci tombe sur une période de rebonds, il est nécessaire d'attendre pour confirmer la présence d'un changement d'état du B/P. La fréquence d'échantillonnage ne doit pas se faire au hasard ; la théorie du signal nous dit que celle-ci doit être au moins le double de la fréquence maximum des événements d'entrée.

Boutons poussoirs et rebonds

Le phénomène de rebonds décrit plus haut est dû à l'élasticité des pièces mécaniques qui composent le bouton poussoir ; certains B/P en génèrent plus que d'autres mais il faut toujours tenir compte que les rebonds peuvent se produire lors de l'appui sur un B/P.

La solution logicielle est d'ajouter un délai dès qu'un changement d'état est détecté, pendant lequel aucune lecture n'est faite. Mais il est alors nécessaire de détecter les deux flancs.

Nous reparlerons de ces techniques plus loin en nous inspirant des exemples donnés dans l'IDE.

Différents types de boutons poussoirs

Ce que nous venons de dire pour un bouton poussoir est bien entendu valide pour un interrupteur. En effet, le poussoir garde sa position de façon temporaire (le temps de l'appui) alors qu'un interrupteur garde sa position jusqu'à ce qu'on le manipule à nouveau. Le changement d'état d'un interrupteur est donc plus facile à traiter.

L'ILS (Interrupteur à Lames Souples) peut aussi remplacer le bouton poussoir et il peut être déclenché par le passage des trains si ceux-ci sont munis d'un aimant.

Enfin, une pédale de voie dont les contacts se ferment sous le poids des roues, est également l'équivalent d'un bouton poussoir, de même que la solution d'isoler une petite portion de la voie comme nous l'avons évoqué dans le chapitre 10 lorsque nous avons décrit les triggers de Schmitt.

Dans tous les cas, le dispositif doit être monté comme le suggère la figure 18.3 de ce cours, en faisant appel à des résistances de pull-up ou de pull-down. On peut aussi faire appel aux résistances de pull-up du microcontrôleur et dans ce cas, le dispositif est simplement relié à la masse comme le montre la figure 19.3. Mais retenez que **dans tous les cas, ces résistances existent soit à l'extérieur soit à l'intérieur du microcontrôleur**.

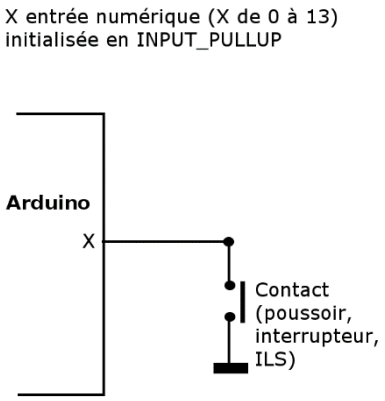

Figure 19. 3

Montage de plusieurs B/P

On aura vite épuisé les entrées numériques du microcontrôleur si on branche un B/P par entrée. Heureusement, il existe des solutions pour économiser le nombre d'entrées.

La première solution est de réaliser un montage sous forme de matrice de B/P, ce qui constitue donc un petit clavier à 12 ou 16 touches. La figure 19.4 montre un clavier à 12 touches, organisé en 4 lignes et 3 colonnes (ce genre de clavier à 12 ou 16 touches se trouve tout fait). L'appui sur une touche court-

circuite une ligne avec une colonne, ce qui permet de repérer la touche enfoncée. Il suffit d'écrire un petit logiciel basé sur le principe suivant : les lignes sont placées en sortie et génèrent un niveau logique haut. Les colonnes sont placées en entrées et vont pouvoir être lues. Le logiciel se charge de mettre successivement chaque ligne à l'état bas (et une ligne à la fois) et va lire les colonnes. Si une touche est appuyée, l'état bas de la ligne se retrouve sur la colonne et la touche peut être localisée. Bien entendu, on peut aussi inverser les états ; les lignes sont toutes à l'état bas au départ et successivement prennent un état haut qui se retrouve sur la colonne.

Un clavier à 12 touches monopolise 7 E/S du microcontrôleur, un clavier à 16 touches en monopolise 8.

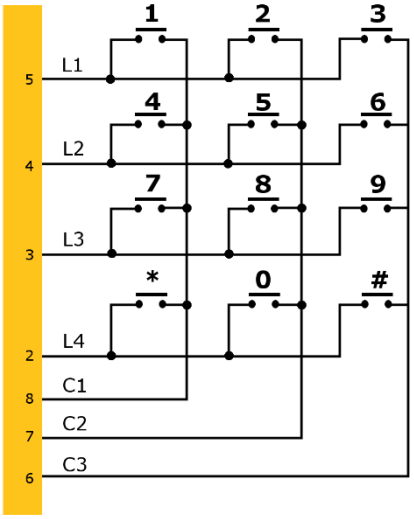

Figure 19. 4

Une autre solution est possible lorsque le nombre de B/P est limité à 4 ou 5 : c'est de faire appel à une entrée analogique et à un pont de résistances agissant comme diviseur de tension. La figure 19.5 montre le montage à réaliser, avec des ILS à la place des B/P (ce montage sert notamment à localiser la position d'un train sur un réseau où les ILS sont disposés sur la voie). Si on remplace les ILS par de simples B/P, ce montage peut servir à créer un pavé de navigation avec quatre flèches (haut, bas, gauche, droite) et un bouton de validation (OK) pour se repérer dans un menu affiché sur écran LCD (à cristaux liquides).

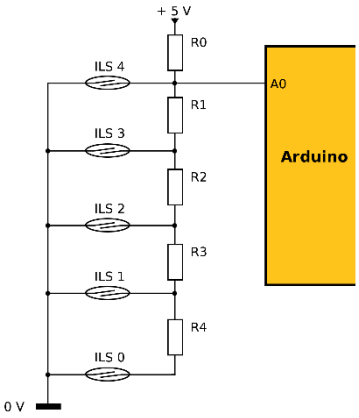

Figure 19. 5

En fonction du contact qui se ferme, la tension lue par l'entrée est différente (il suffit d'appliquer la loi d'Ohm pour faire l'étalonnage). Bien entendu, en raison de l'imprécision sur la valeur des résistances, il faut lire la valeur de l'entrée analogique et la comparer à une plage de tension (centrée sur la valeur théorique) pour déterminer la touche appuyée. Le tableau 1 montre les valeurs de résistances à choisir pour avoir une courbe de tension la plus linéaire possible :

R0	**1000**
R1	**220**
R2	**470**
R3	**800** (330 + 470) (*)
R4	**2500** (1000 + 1500) (*)

Tableau 1

(*) : Pour obtenir des valeurs proches du calcul théorique, on utilise pour les résistances R3 et R4 deux résistances en série dont les valeurs sont données entre parenthèses. Chacun fera au mieux en fonction de son stock de résistances.

Le tableau 2 montre les tensions obtenues avec les résistances du tableau 1.

ILS fermé	**Tension recherchée**	**Tension obtenue**	**Valeur numérique**
Aucun	5	5	1023
0	4	4	818
1	3	2,99	612
2	2	2,0	409
3	1	0,9	184
4	0	0	0

Tableau 2

On voit qu'avec ces valeurs de résistances, la linéarité est assez bien respectée (les tensions mesurées au voltmètre sur le montage sont à 10% près celles qu'on cherche à avoir).

Comme on l'a vu au cours de ce chapitre, les boutons poussoirs ou les contacts se fermant automatiquement sont très utilisés en électronique programmable comme périphérique d'entrée. Il faut bien sûr savoir les traiter correctement tant sur le plan du montage électrique, que sur le plan de la lecture informatique.

<u>À retenir sur les boutons-poussoirs</u> :

- Les B/P (ou contacts) sont à monter avec des résistances de pull-up ou de pull-down, ou bien utilisés avec les résistances de pull-up dont est pourvu le microcontrôleur.
- Les B/P ne font pas passer le signal instantanément d'un état à l'autre ; il se produit un phénomène de rebonds qui dure généralement moins de 50 ms.
- La lecture de l'état d'un B/P se fait par scrutation de son état, c'est-à-dire des lectures répétées correspondant à un échantillonnage.
- La fréquence de lecture doit être au moins le double de la fréquence des événements intervenant en entrée.
- Malgré cela, on n'est jamais certain de ne pas louper un événement : l'échantillonnage doit être adapté à l'application.

- La reconnaissance des flancs montants ou descendants d'un signal numérique donne les meilleurs résultats pour comprendre les changements d'état d'un B/P.
- Les rebonds sont généralement traités par double lecture avec période d'attente de l'ordre de 20 à 50 ms.
- De manière à économiser les broches d'entrée d'un microcontrôleur, plusieurs B/P sont montés en matrice, ou bien avec un diviseur de tension à résistances et en utilisant une entrée analogique.