

# EdgeDebounceLite

## TUTORIEL

### UN INTERRUPTEUR NE PEUT ÊTRE QUE FERMÉ OU OUVERT, NON?

Oui et non. Un interrupteur est conçu pour interrompre le courant ou lui laisser le passage.

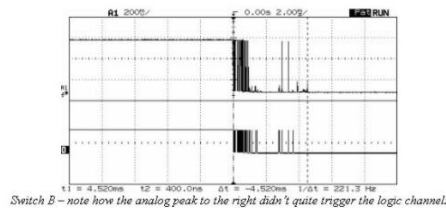
Habituellement, l'interrupteur est composé de deux (ou plus) pièces d'un élément conducteur. Quand les deux pièces sont en contact, l'interrupteur est fermé et laisse passer le courant entre les deux bornes.

Quand les deux pièces ne sont plus en contact, l'interrupteur est ouvert et le courant ne passe plus.



Malheureusement, il y a deux moments où l'interrupteur laisse passer le courant d'une manière aléatoire. C'est quand on l'ouvre ou quand on le ferme. Quand les contacts sont tout près l'un de l'autre, mais que l'interrupteur n'est pas tout à fait ouvert ou fermé. Les électrons peuvent migrer d'un contact vers l'autre de manière sporadique. On pourrait le comparer à des éclairs dans un orage. Plus les contacts sont près l'un de l'autre, plus il y aura d'éclairs laissant passer le courant de l'un à l'autre.

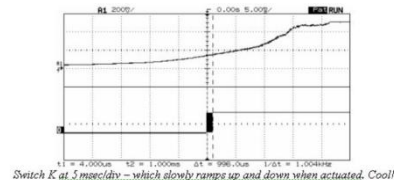
Voici un exemple, à l'oscilloscope, d'un interrupteur que l'on est en train de fermer<sup>1</sup> :



Dans le haut de l'image, on voit le voltage tel que mesuré. Dans le bas on voit le voltage retourné par un convertisseur analogique-digital. On peut voir les « éclairs », très fréquents au début, puis qui s'espacent jusqu'à ce que le convertisseur se stabilise à 0 V. Il se passe 2 millisecondes entre le moment où l'interrupteur commence à être relâché et celui où il est enfin stable. Si on met en boucle un `digitalRead()` pendant cette période, il retournera

environ 650 lectures et on ne peut être certain d'aucune d'elles.

Voici un autre exemple :



Cet interrupteur ne se comporte pas comme l'autre. Le voltage augmente progressivement. Si on regarde l'image du bas, il se passe quelque chose d'étrange. Pendant un certain temps le convertisseur analogique-digital est stable à 0 V. Puis, il devient très instable, passant de 5 V à 0 V très rapidement. Il se stabilise enfin à 5V, quand l'interrupteur est effectivement fermé. Cela tient aux spécifications

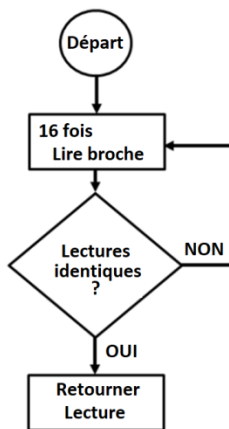
du convertisseur analogique-digital. S'il mesure entre 0 et 0,8 V sur la broche, il retourne LOW. S'il mesure entre 2,2 et 5 V, il retourne HIGH. Entre 0,8 et 2,2 V, il retourne au hasard HIGH ou LOW, ne nous laissant pas savoir ce qu'il fait.

<sup>1</sup> Ces exemples sont tirés de : <http://www.ganssle.com/debouncing.htm>

## COMMENT SAVOIR SI L'INTERRUPTEUR EST RÉELLEMENT FERMÉ OU OUVERT?

Comme nous venons de le voir, le comportement erratique ne se produit que lorsque l'interrupteur est en voie de se fermer ou de s'ouvrir. L'approche en sera une de patience. Au lieu de ne prendre qu'une lecture de la broche, nous en prendrons plusieurs successivement. Supposons que l'on décide de prendre 16 lectures successives aussi rapprochées que possible (le tout se passe en environ 90 millièmes de secondes). On vérifie ensuite que toutes les lectures sont identiques (toutes HIGH ou toutes LOW). Si c'est le cas, les probabilités que le résultat soit valide est grand. Si, par contre, il y a des lectures HIGH parmi des LOW, on a la certitude que l'on est en phase de transition. Comme on l'a dit précédemment, il suffit d'être patient et d'attendre que la tempête cesse. On va reprendre des séries de 16 lectures jusqu'à ce qu'elles concordent toutes (16 HIGH ou 16 LOW).

### L'ALGORITHME<sup>2</sup>



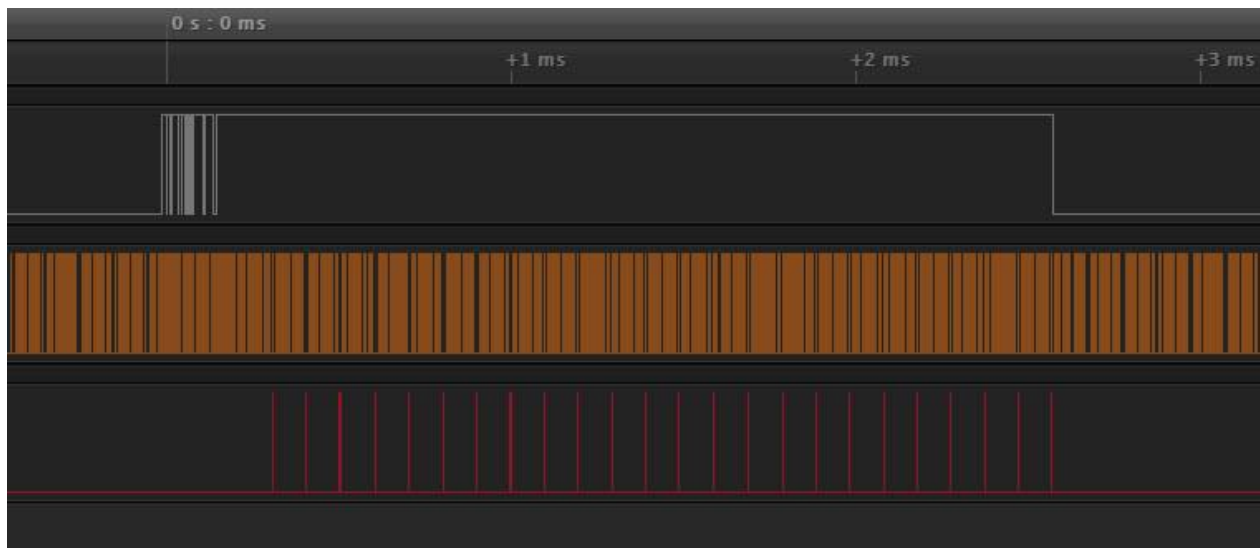
1. Lire 16 fois la broche
2. Si toutes les lectures sont identiques, retourner la lecture
3. Si non, reprendre à l'étape 1

Par défaut, les broches sont lues en rafales de 16. Il est toutefois possible de changer la taille de ces rafales entre 1 (aucune stabilisation du signal) et 32 (nombre maximal de lectures faites en rafale  $\approx 180 \mu s$ ).

Pour la plupart des interrupteurs, 16 lectures par rafale sont suffisantes pour stabiliser le signal. Selon le type d'interrupteur ou de l'application qu'on désire en faire, il est possible d'en changer la valeur. Par exemple, j'ai plusieurs encodeurs et j'obtiens de meilleurs résultats avec ceux-ci avec des rafales de 8 lectures.

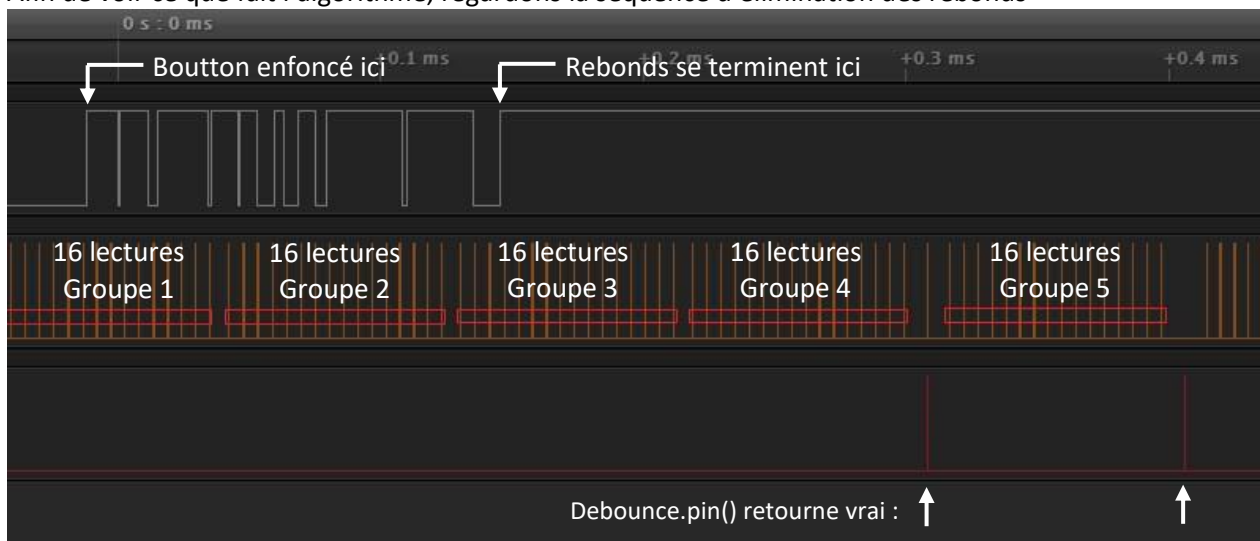
<sup>2</sup> Cet algorithme peut être retrouvé ici : <http://www.ganssle.com/debouncing-pt2.htm>

Quand j'ai mentionné cette Librairie sur le forum d'Arduino, **larryd** m'a envoyé des images du déroulement de l'algorithme tel que vus à l'aide d'un oscilloscope.  
(<https://forum.arduino.cc/index.php?topic=489925.0> à la page 2)



Le signal du haut (en gris) est celui de l'interrupteur. Le passage au mode fermé montre des rebonds.  
Le signal du milieu (en orange) montre les séries de lectures de la broche  
Le signal du bas (en rouge) montre que bouton.closed() ne retourne vrai que si l'interrupteur est ouvert et qu'il n'y a plus de rebonds.

Afin de voir ce que fait l'algorithme, regardons la séquence d'élimination des rebonds



L'appel à `debounce.pin()` est fait à la gauche de l'image.  
Dans le premier groupe de lectures, on détecte des fluctuations. On doit lire un nouveau groupe.  
Dans le second groupe de lectures, on détecte encore des fluctuations. On doit lire un nouveau groupe.  
Dans le troisième groupe de lectures, on détecte encore des fluctuations. On doit lire un nouveau groupe.  
Dans le quatrième groupe de lectures, le signal est stable. `debounce.pin()` retourne **vrai**.  
Un deuxième appel à `debounce.pin()` est fait immédiatement après.  
Dans le cinquième groupe de lectures, le signal est stable. `debounce.pin()` retourne immédiatement **vrai**.

## LA LIBRAIRE

Pour pouvoir utiliser cette librairie, il faut d'abord l'ajouter à l'IDE de l'Arduino. Ce site explique bien comment faire : <https://knowledge.parcours-performance.com/librairies-arduino-installer/>

Dans le sketch, on doit mentionner que l'on désire l'utiliser. La ligne suivante doit apparaître au début du sketch, avant la section setup() :

```
#include <EdgeDebounceLite.h>
```

Par la suite nous devons instancier l'objet :

```
EdgeDebounceLite debounce;
```

Par la suite, vous pouvez donner un nom à la broche à laquelle est attaché l'interrupteur. (Il n'est pas nécessaire de le faire, mais cela rend le code plus facile à lire par la suite et cela permet de n'avoir qu'un seul changement à faire au sketch si on change de broche en chemin.)

```
#define BROCHE_BOUTON 2;
```

On ne doit pas oublier, comme on le fait normalement, d'invoquer pinMode :

```
pinMode(BROCHE_BOUTON, INPUT_PULLUP);
```

Pour lire la broche, il suffit d'utiliser debounce.pin() au lieu de digitalRead() :

```
byte uneVariable = debounce.pin(BROCHE_BOUTON);
```

au lieu de

```
byte uneVariable = digitalRead(BROCHE_BOUTON);
```

Une dernière chose. Il est possible d'ajuster la sensibilité du stabilisateur (debouncer) en modifiant la taille des rafales de lectures entre 1 et 32.

Dans la section setup() de notre sketch, nous mettrons :

```
debounce.setSensitivity(8);
```

On peut aussi obtenir quelle est la sensibilité actuelle avec :

```
int laSensibilite = debounce.getSensitivity();
```

## EXEMPLE

```
#include <EdgeDebounceLite.h>
EdgeDebounceLite debounce;

#define BROCHE_BOUTON 2
#define BROCHE_DEL 13

void setup() {
    pinMode(BROCHE_BOUTON, INPUT);
    pinMode(BROCHE_DEL, OUTPUT);
}

void loop() {
    if (debounce.pin(BROCHE_BOUTON)) {
        digitalWrite(BROCHE_DEL, HIGH);
    }
    else {
        digitalWrite(BROCHE_DEL, LOW);
    }
}
```

**J'espère sincèrement que la Librairie EdgeDebounceLite vous sera utile dans vos projets**  
Jacques Bellavance