Clé de sélection des pantographes

16/12/2025

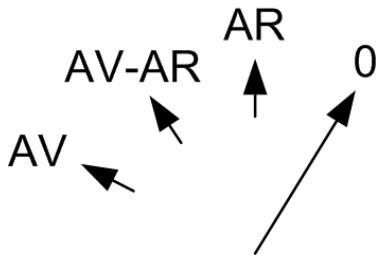
Ce document indique comment câbler une clé de pantographe de BB7200 ou BB7600, pour mon programme Arduino de poste de conduite.

La description du poste de conduite se trouve ici : https://www.la-tour.info/uts/uts_page15.html

Clé de pantographe de BB7600 :

La clé de sélection des pantographes, permet de choisir quatre positions pour les pantographes.

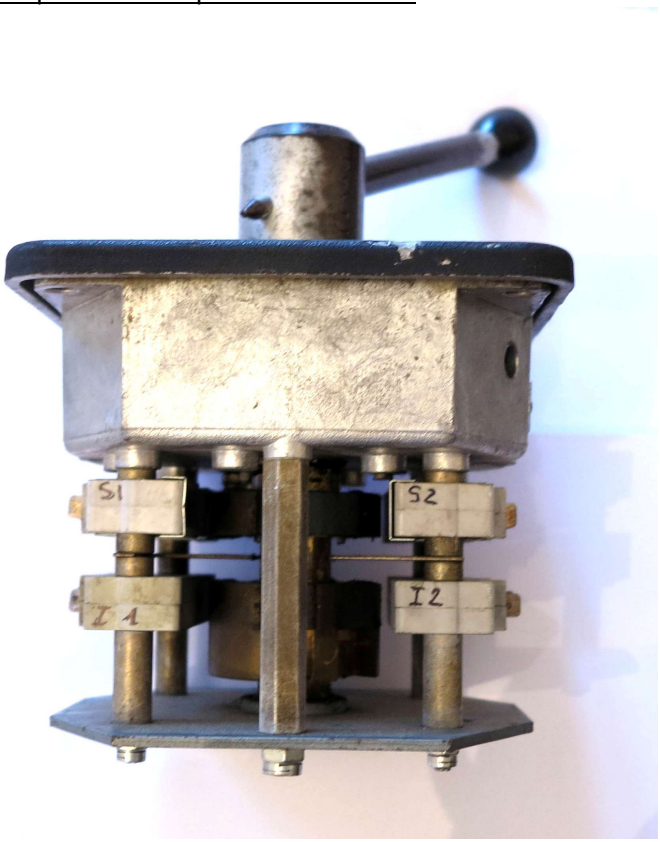
On peut avoir les deux pantographes baissés, l'arrière levé, les deux levés ou l'avant levé.

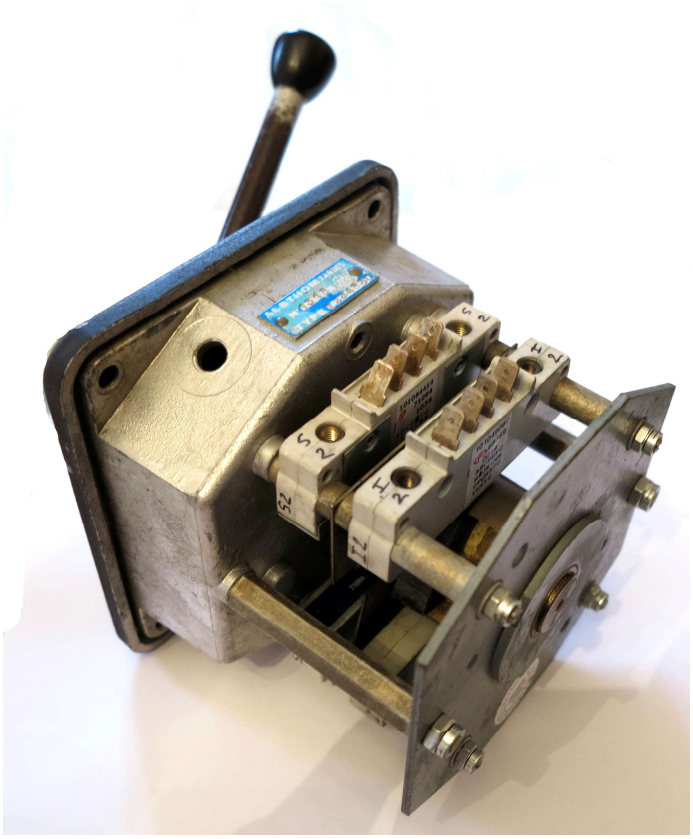


Sous ce gros bloc, on trouve 4 interrupteurs à 4 plots à cosses (S1, S2, I1 et I2).

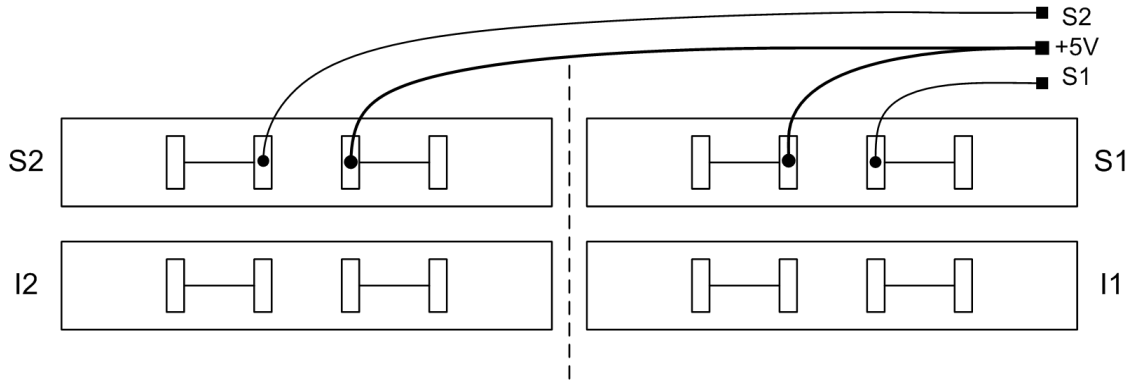
Sur les quatre cosses, les plots à cosses de droite et de gauche, sont relies entre eux, sur tous les interrupteurs. On a donc un seul interrupteur par bloc.

<u>Les quatre interrupteurs sous la clé :</u>





<u>On va les câbler de la manière suivante :</u>



<u>Pour les quatre positions de la clé, on a les contacts suivants établis :</u>

	S1	S2	I1 / I2
0	—		
AR	+		+
AV-AR	+	+	+
AV		+	+

Les contacts fonctionnent de la même façon pour les deux interrupteurs I1 et I2.

En utilisant uniquement les contacts S1 et S2, on peut en déduire la position de la clé.

Variable x = "Etat S1" + 2*"Etat S2"

On a donc x = 0 : Position 0, x = 1: Position AR, x = 3 : Position AV+AR, x = 2 : Position AV.

Entre deux états, on n'a qu'une entrée qui change de valeur. C'est donc compatible avec la logique de mon programme, qui est séquentiel. On traite les changements d'état dés qu'ils arrivent.

Pour concevoir le code Arduino, on utilise une variable "cle_selection_panto" qui est la valeur de 'x' avant mouvement de la clé.

Le code fonctionne pour le programme sur PC : "Train Simulator Classic" et la locomotive BB4400KW de SimExpress.

La variable 'i' est la valeur "Etat S1" + 2*"Etat S2" de la position de la clé.

Dans le code, dans la gestion des entrées, on en déduit si la clé a tourné à gauche ou à droite, pour envoyer la touche 'p' ou 'P' au simulateur.

Sur ma carte Arduino, l'interrupteur S1 est câblé sur l'entrée 12, et S2 sur 13.

```
else if ((x == 12) || (x == 13)) {
   i = bitRead(i2c_in3_val, 12) + 2 * bitRead(i2c_in3_val, 13); // i : 0 = 0, 1 = AR, 3 = AV+AR, 2 = AV.
   if      (((cle_selection_panto == 0) && (i == 1)) || ((cle_selection_panto == 0) && (i == 3)) ||
((cle_selection_panto == 1) && (i == 3)) || ((cle_selection_panto != 2) && (i == 2))) {
        tch_env(NORMAL,         'p', duree_touche_200); bitWrite(i2c_in3_val_ref, x, bitRead(i2c_in3_val,
x)); } // p  Clé sélection panto vers la gauche.
   else if (((cle_selection_panto == 2) && (i == 3)) || ((cle_selection_panto == 2) && (i == 1)) ||
((cle_selection_panto == 3) && (i == 1)) || ((cle_selection_panto != 0) && (i == 0))) {
        tch_env(KEY_LEFT_SHIFT, 'p', duree_touche_200); bitWrite(i2c_in3_val_ref, x, bitRead(i2c_in3_val,
x)); } // P  Clé sélection panto vers la droite.
   cle_selection_panto = i;
}
```

Dans le code, dans la gestion des informations reçus par l'Arduino,

```
// <   :cle_panto_continu_control:1> ==> Interrupteur à 4 positions - Pantographe.
//          S2    S1
//   i2c_in3_val(13)   i2c_in3_val(12)    Position pantographes    valeur cle_panto_continu_control
//          0     0     =    0           0.000
//          0     1     =   AR           0,333
//          1     1     = AV + AR        0,666
//          1     0     =   AV           1.000

    else if (!strcmp(buffer_rx[1],"cle_panto_continu_control")) {       //
      buffer_rx_fvaleur = atof(buffer_rx[2]);   // Obligé de passer par un nombre flottant.
      if       (buffer_rx_fvaleur > 0.98)        { bitWrite(i2c_in3_val_sim, 13, 1);
bitWrite(i2c_in3_val_sim, 12, 0);
   bitWrite(i2c_in3_maj_sim, 13, 1); bitWrite(i2c_in3_maj_sim, 12, 1);  // La comparaison et demande de mise
à jour, se fera automatiquement toutes les xxx msec.
      }
      else if  (buffer_rx_fvaleur < 0.02)        { bitWrite(i2c_in3_val_sim, 13, 0);
bitWrite(i2c_in3_val_sim, 12, 0);
   bitWrite(i2c_in3_maj_sim, 13, 1); bitWrite(i2c_in3_maj_sim, 12, 1); // La comparaison et demande de mise à
jour, se fera automatiquement toutes les xxx msec.
      }
      else if ((buffer_rx_fvaleur > 0.31) && (buffer_rx_fvaleur < 0.35)) { bitWrite(i2c_in3_val_sim, 13, 0);
bitWrite(i2c_in3_val_sim, 12, 1);
   bitWrite(i2c_in3_maj_sim, 13, 1); bitWrite(i2c_in3_maj_sim, 12, 1);  // La comparaison et demande de mise
à jour, se fera automatiquement toutes les xxx msec.
      }
      else if ((buffer_rx_fvaleur > 0.64) && (buffer_rx_fvaleur < 0.68)) { bitWrite(i2c_in3_val_sim, 13, 1);
bitWrite(i2c_in3_val_sim, 12, 1);
   bitWrite(i2c_in3_maj_sim, 13, 1); bitWrite(i2c_in3_maj_sim, 12, 1); // La comparaison et demande de mise à
jour, se fera automatiquement toutes les xxx msec.
```

```
      }
    }
```

…

Dans le code, dans la gestion des informations reçus par l'Arduino, on vérifié périodiquement si la clé est en bonne position sur le simulateur.

Si la clé n'est pas dans la même position, on indique à l'Arduino la position actuelle dans le simulateur :

```
cle_selection_panto = bitRead(i2c_in3_val_sim, 12) + 2 * bitRead(i2c_in3_val_sim, 13); // i : 0 = 0, 1 = AR, 2 = AV+AR, 3 = AV.
```

Pour plus de détail sur le code, voir la description du poste de conduite de BB7200 qui se trouve ici : https://www.la-tour.info/uts/uts_page15.html

A+