

MIGRATION DU BOITIER KVB D'UN PUPITRE DE LOCOMOTIVE Le 21/03/2025

Ce document explique comment animer le boîtier KVB, d'une ancienne cabine de conduite d'une locomotive.
KVB = C(K)ommande Vitesse Balise.

Ce document se trouve ici : http://www.la-tour.info/uts/uts_page15.html

18/12/24 - Version 1.0 du logiciel et notice remaniée.

06/01/25 - Version 1.1 du logiciel : Ajout de balises '<>' sur les n° des roues codeuses. Optocoupleurs sur liaison.
Bouton [TEST] sonore, 's' = Séquence d'initialisation.

15/02/25 - Version 1.2 du logiciel : Séquences d'initialisation + "PR400", "UC512", "00 000" pour SimExpress.

21/03/25 - Version 1.2 du logiciel : Amélioration du montage d'isolation optique + Construction d'un KVB.

C'est pour animer un véritable pupitre de locomotive, utilisé avec un programme de simulateur sur ordinateur.

Si vous construisez un KVB de toutes pièces, ce document peut vous servir à en fabriquer un.



Les questions peuvent être posées sur le forum RMF. Par exemple ici :

<https://www.rmfmagazine.com/phpBB/viewtopic.php?t=203032>

Si vous réalisez des améliorations du logiciel, merci de les poster sur le forum RMF pour en faire profiter les plus grand nombre.

Si vous publiez cette réalisation avec ou sans améliorations, vous devez publier votre code source.

Ce logiciel est un logiciel libre. Exigence du concepteur : Ne pas modifier les lignes d'affichage de l'intro à la mise sous tension. A la mise sous tension, affiche "JLF xx/xx/xxxx" pendant 1 seconde.

Conserver la routine "void intro_jlf()" et son action à l'écran au démarrage de l'Arduino.

On peut modifier ce programme et le diffuser. Dans ce cas, il faut préciser l'origine et donner accès aux sources modifiées.

Définition : Un logiciel libre est un logiciel distribué avec l'intégralité de ses programmes-sources, afin que l'ensemble des utilisateurs qui l'emploient, puissent l'enrichir et le redistribuer à leur tour.

Note : Un logiciel libre n'est pas nécessairement gratuit et les droits de la chaîne des auteurs sont préservés.

Équivalent étranger : free software, open source software.

(Source : Vocabulaire de l'informatique (liste de termes, expressions et définitions adoptés), NOR: CTNX0710138K, J.O n° 93 du 20 avril 2007 page 7078, texte n° 84)

logiciel libre

Par logiciel libre on entend un logiciel qui offre la liberté aux utilisateurs d'exécuter, de copier, de distribuer, d'étudier, de modifier et d'améliorer le logiciel. Plus précisément, elle fait référence à quatre types de liberté pour les utilisateurs du logiciel :

La liberté d'exécuter le programme, pour tous les usages (liberté 0).

La liberté d'étudier comment le programme fonctionne et de l'adapter à ses besoins (liberté 1). L'accès au code source est une condition requise.

La liberté de redistribuer des copies, (liberté 2).

La liberté d'améliorer le programme et de diffuser les améliorations au public pour en faire profiter toute la communauté (liberté 3). L'accès au code source est une condition requise.

Je garde le maximum de chose d'origine, dans ce boîtier dont les afficheurs. Les afficheurs d'origine sont des :

HP5082-7620 = Hauteur digit 7,6 mm 0,3 inch - Jaune Anode (+) commune - H 19 mm x L 10 mm

HP HDSP-4600 = Hauteur digit 10,9 mm 0,3 inch - Vert Anode (+) commune - H 19 mm x L 12,5 mm

Courant nominal = 20 mA

J'aurai pu utiliser des circuits de type MAX7219 ou MAX7221, mais j'ai préféré utilisé un circuit qui gère les boutons poussoirs en même temps.

Le circuit TM1638 est nativement fait pour piloter des afficheurs à cathode commune, mais moyennant une adaptation du logiciel de l'Arduino, il peut aussi piloter des afficheurs à Anode commune.

Dans ce cas, il faut reprendre la programmation, pour inverser l'utilisation des segments et des digits.

La librairie de Ricardo Batista : <https://github.com/rjbatista/tm1638-library> est utilisable directement pour les afficheurs à Anode commune.

On garde donc les afficheurs d'origine.

Les voyants de la face avant sont des ampoules. On les remplacera par des leds.

L'éclairage des touches de la face est réalisé par des ampoules. On les remplacera par des leds blanches.

Démontage

La première étape est le démontage du KVB. On retire les grandes plaques de circuits imprimés.

On retire la face avant. On coupe aux ciseaux les nappes marron entre la plaque principale et la face avant.

On démonte le circuit imprimé de la face avant.

Ensuite, toujours avec un coup de ciseaux, on donne un coup de ciseaux entre toutes les pistes.

On retire ces fils pour avoir un circuit tout propre, avec une pompe ou une tresse à dessouder.

<https://fr.aliexpress.com/w/wholesale-pompe-%C3%A0-dessouder-%C3%A9lectrique.html>

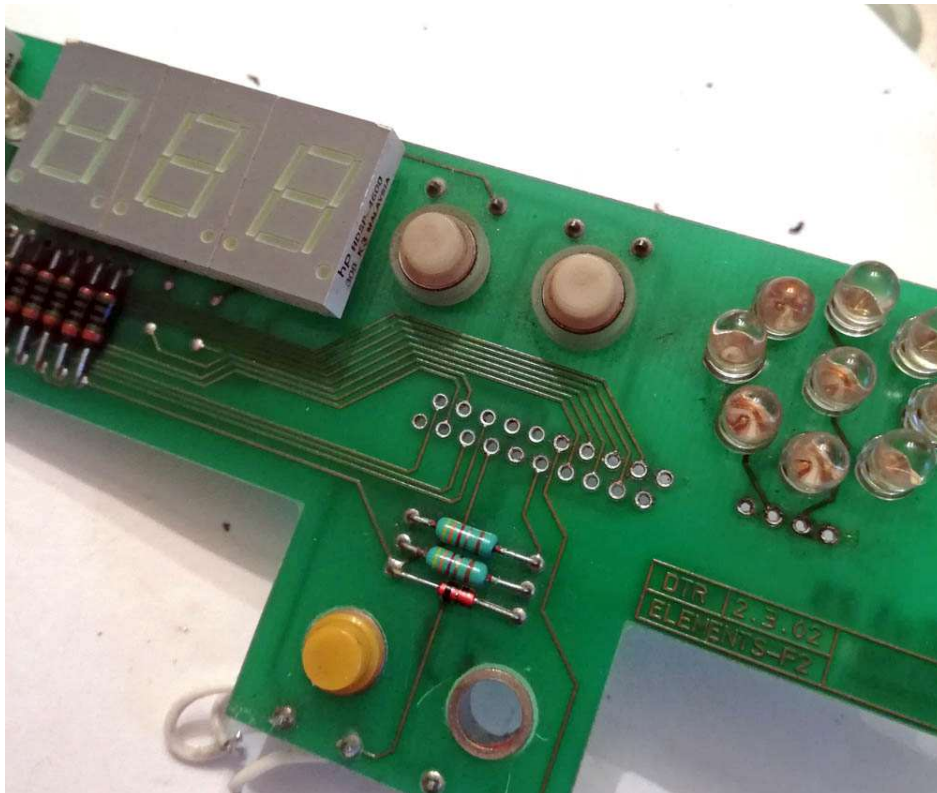
Attention, sur le modèle avec un **circuit imprimé** pour les boutons poussoir et un **circuit imprimé** pour le connecteur externe, j'ai eu beaucoup de problème.

Le circuit imprimé relie des plots entre eux, et la nappe marron fait parfois court-circuit au niveau de la découpe, les pistes des deux côtés peuvent se toucher aléatoirement.

J'ai mis un Arduino à la poubelle, alors qu'il devait être bon, car les plots RX/TX se touchaient et empêcher un téléchargement, ou un fonctionnement normal.

J'ai eu des fonctionnements aberrants. Pour bien faire, il faudrait tronçonner le circuit imprimé à la disqueuse autour des plots utilisés.

Une fois les nappes dessoudées, on a cela :



On va remplacer les ampoules à filament par des leds 3 mm, pour fortement réduire la consommation du boîtier.
Pour éclairer les boutons lumineux, on remplace les ampoules par des leds blanches au format cms 5050.
On retire les ampoules en tirant sur la languette métallique de coté.



Méthode de remplacement des lampes des boutons poussoirs par des leds

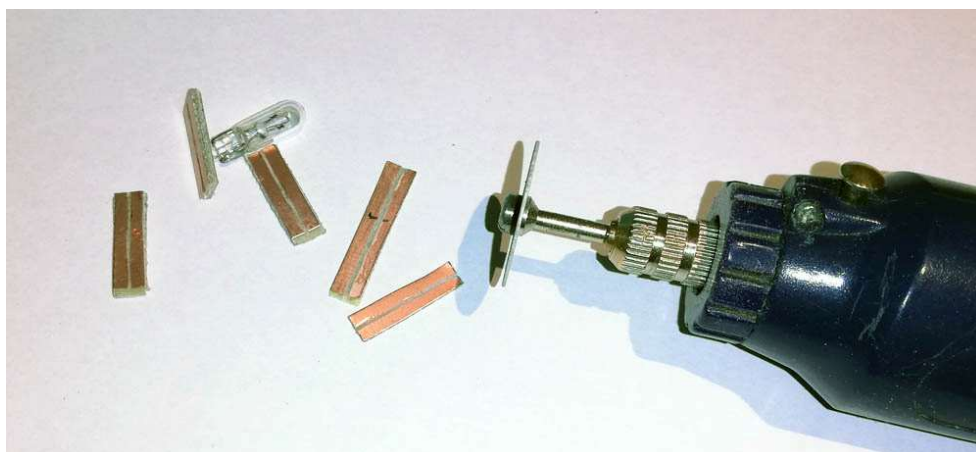
Sur cette série de KVB avec un circuit imprimé, on ne peut pas accéder à l'arrière des boutons poussoirs,. La solution est de remplacer les lampes par devant. On utilisera des leds cms 5050, blanches.



On utilisera un bout de circuit imprimé double face de 1,6 mm d'épaisseur, ou 2 fois du 0,8 mm d'épaisseur collé.

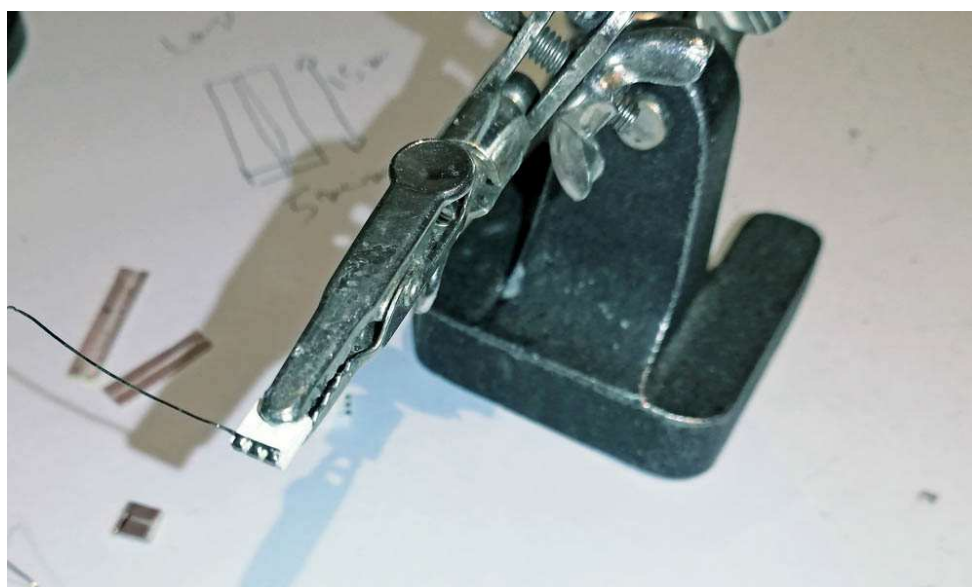
On découpe des bouts de 1,5 cm * 5 mm.

Avec un disque, on trace un sillon isolant au milieu.



On prend des leds cms 5050, blanches.

On soude un fil de chaque coté, pour relier les 3 pattes ensemble.



On pose la led, et on maintient appuyé le circuit imprimé le temps de réaliser les soudures.



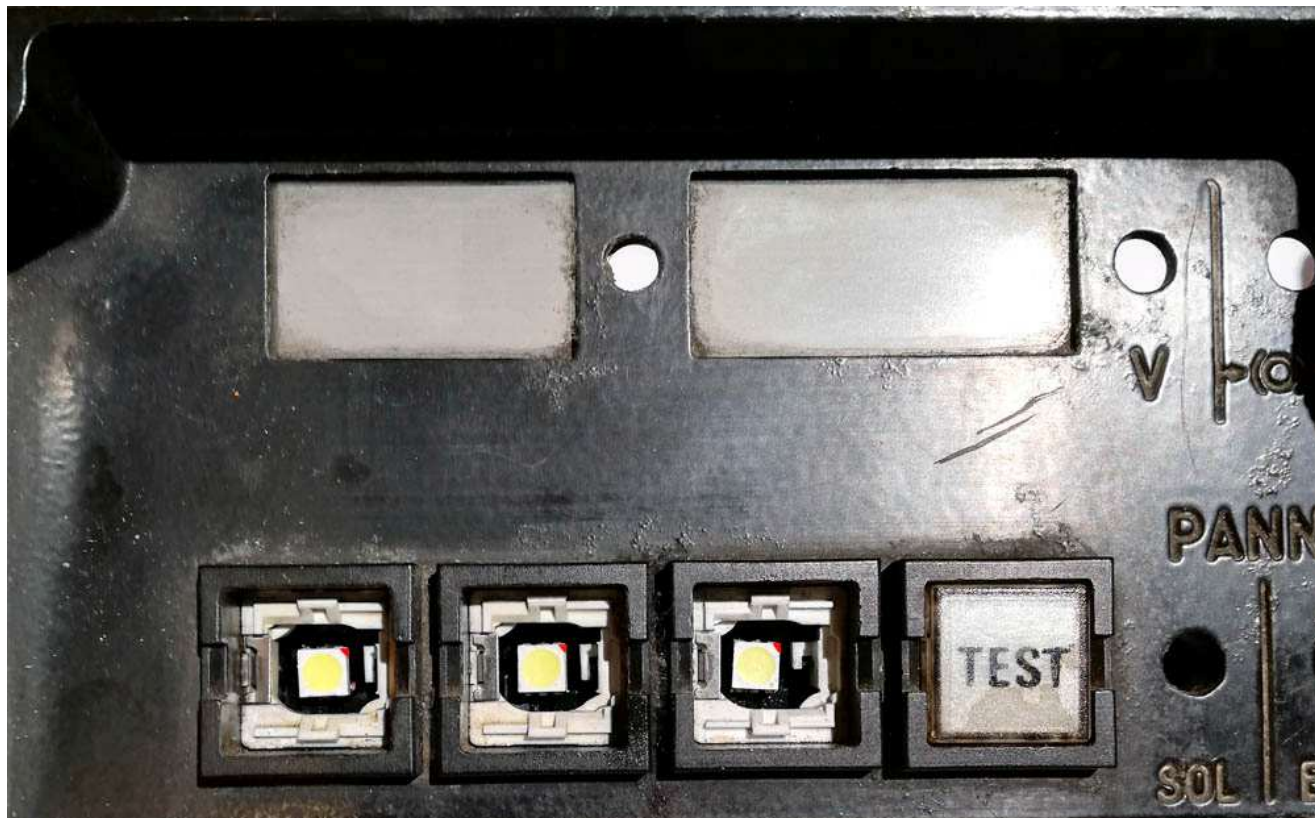
On lime en biais le bout des plaques, pour faciliter l'insertion dans son support.

Il faut étamer les pistes en cuivre, pour éviter l'oxydation, avec de la soudure puis nettoyage à l'acétone.

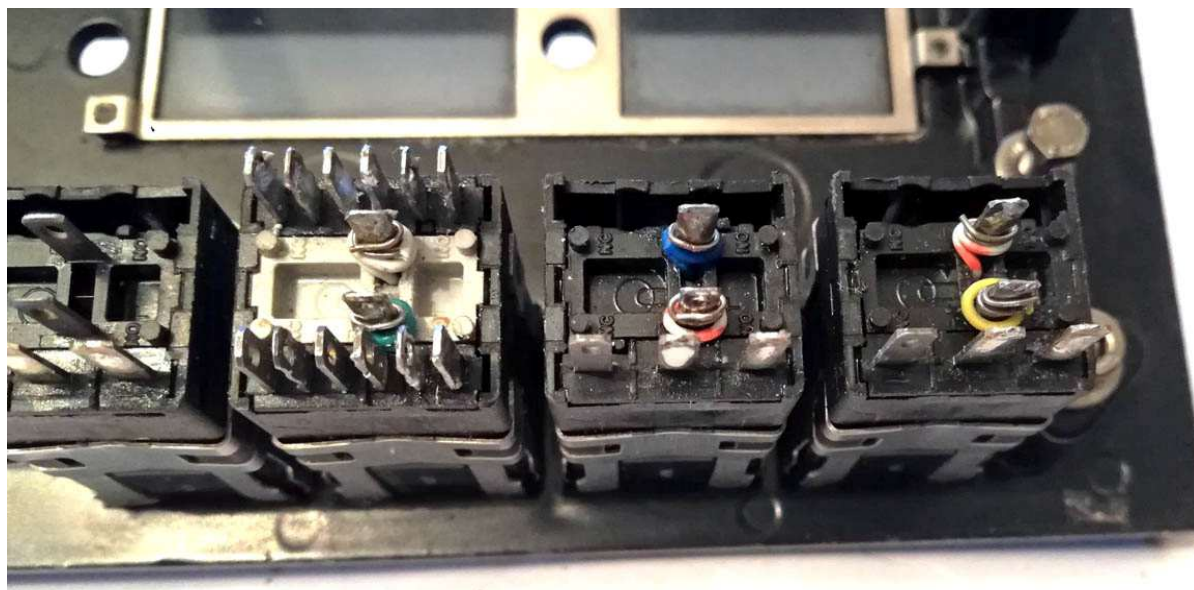
Attention, tester ces leds avant de les mettre en place, avec une résistance de 1 K Ohms en série sous 5 Volts.

On donnera un coup d'aérosol à contact, avant d'insérer ces nouvelles lampes.

Le triangle situé dans le coin des leds (*en rouge sur la photo*), doit être orienté vers le haut à droite.



Sur cette autre série de KVB sans circuit imprimé, on procède de la même manière pour remplacer les ampoules.



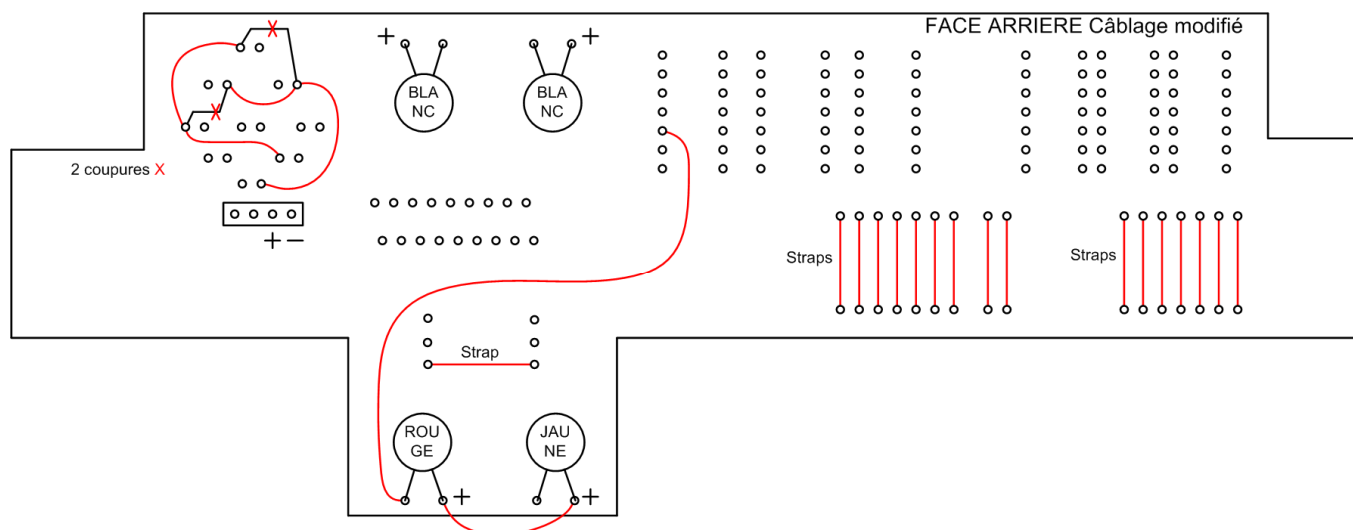
Attention pour un test, alimenter les leds avec une résistance de 1 K Ohms en série sous 5 Volts.

Câblage des 4 leds des voyants de la face avant :

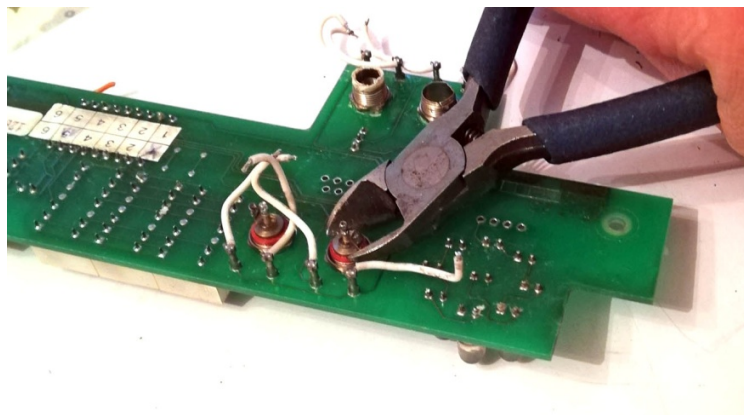
On supprime les ampoules des 4 voyants pour y placer des leds 3 mm opaque (*Pas translucide*).

Pour le gros voyant jaune LSSF d'origine, il est alimenté en 30 Volts. On change le câblage des leds pour l'alimenter en 12 Volts, sans changer les leds. Il faudra donc couper des pistes et en refaire.

Plan de coupures, installation des leds et soudure des nouveaux fils en rouge.

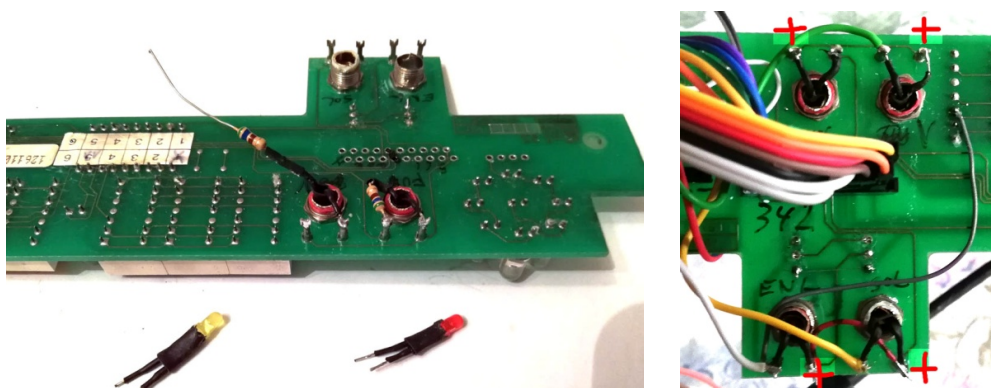


On scalpe les supports de lampes, pour y placer des leds 3 mm, au niveau de la collerette rouge :



On introduit des leds, jaune, rouge et blanches dans leurs supports. Il faudra bien isoler les pattes des leds. **Dans la version finalisée, il ne faut pas mettre de résistance comme sur la photo sur ma version provisoire.**

Souder les leds sur leurs plots de support. Attention à l'orientation des leds (+ en rouge).



Si vous montez un KVB de toute pièce, prendre des afficheurs à Anode (+) commune.

Dimensions de la face avant = 19,2 x 7,2 cm. Diamètre externe LSSF = 3,4 cm

Dimensions des afficheurs = 3 x 1,8 cm + 3,8 x 1,8 cm.

Dimensions des boutons poussoirs = 2 x 2 cm hors-tout.

Voir le schéma au paragraphe : Branchement à la carte TM1638 : La partie afficheur, voyant et led des boutons :

Les (+) des afficheurs sont reliés aux plots n° 1, 2, 3, 4, 5, 6 du TM1638.

Les segments des afficheurs sont reliés aux plots n° a, b, c, d, e, f, g du TM1638.

Le (+) des 4 leds des voyants V, FU, SOL et ENGIN doit être relié au plot n° 7.

Le (-) du voyant V doit être relié au plot n° a, FU vers n° e, SOL vers n° b, ENGIN vers n° c.

Le (+) des 3 leds d'éclairage des touches, doit être relié au plot n° 8.

Le (-) de la led d'éclairage de la touche [VAL] doit être relié au plot n° a, [MV] vers n° b, [FC] vers n° c.

Alimenter votre lampe LSSF en 5 Volts, et si besoin ajouter un transistor en sortie, pour dépasser les 5 mA.

Il ne doit pas y avoir de résistance en série avec ces leds, sauf la LSSF.

Reprise du gros voyant LSSF

D'origine, il est alimenté en 30 Volts. On change le câblage des leds pour l'alimenter en 12 Volts, sans changer les leds.

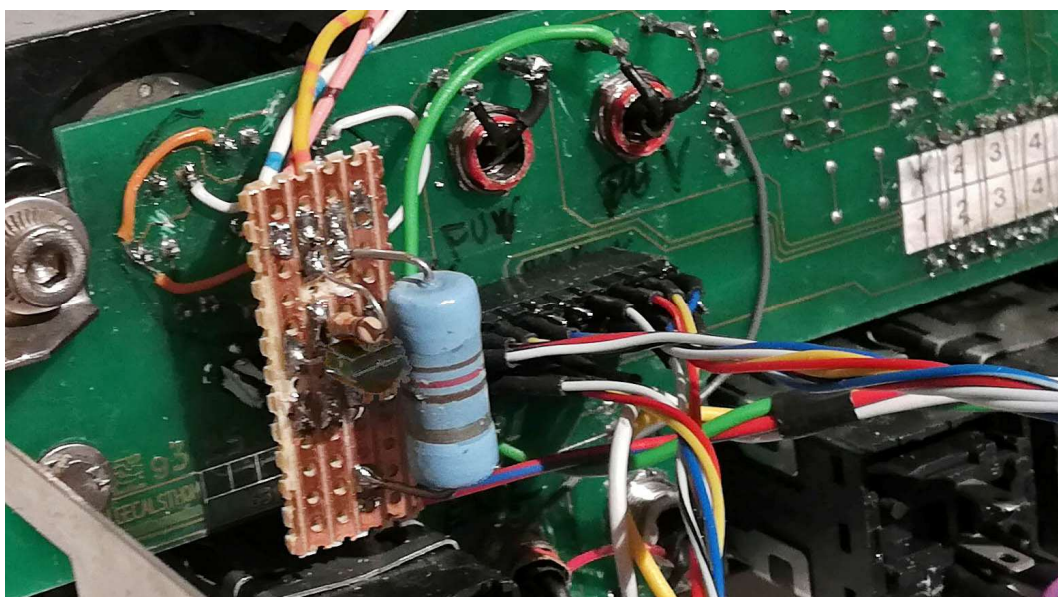
Couper les deux pistes sous les croix rouge. Souder les fils (*orange et blanc*), sur les 3 plots. Remettre correctement de la soudure sur ces plots, car le circuit imprimé est vernis. Ensuite, on pourra alimenter ce groupe de leds avec les plots '+' et '-', avec une résistance en série.



Une fois le connecteur noir à 4 broches soudé, on fabrique un mini circuit imprimé pour la commande de ce voyant 12 Volts, à partir d'une sortie de l'Arduino 0/5 Volts.

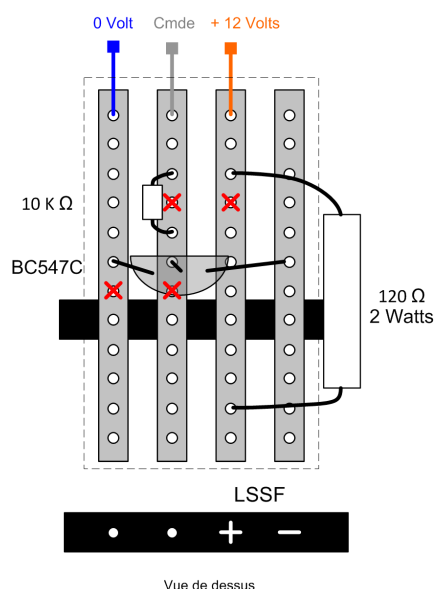
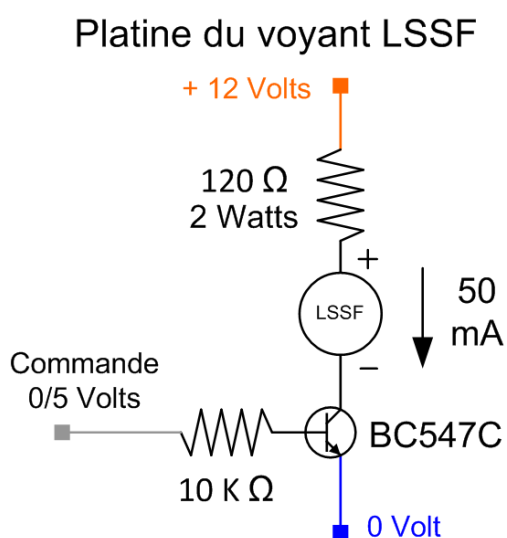
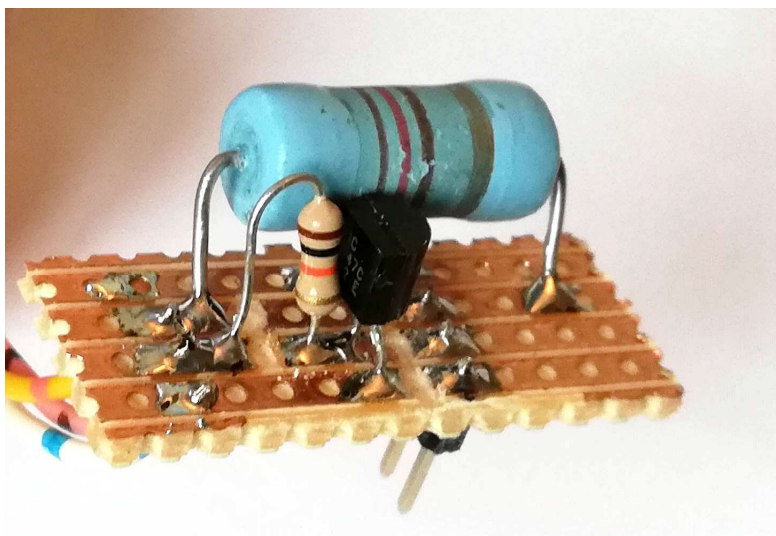
Le voyant LSSF utilise des leds jaune à faible rendement. J'ai conservé ces leds, il faut alors 50 mA pour alimenter correctement ce voyant. Elles sont soudées en 3 lignes, ce qui fait 16 mA par led.

Si le voyant n'éclaire pas assez fort, on peut augmenter la puissance de ce voyant en soudant de nouvelles leds jaune cristal, à la place des anciennes.



Il faut très attention à la réalisation de ce circuit. Il faut vérifier à l'ohmmètre, que le 12 Volts n'est pas en contact avec la piste de commande !

Vérifier l'absence de court-circuit entre les 4 plots. Idem, pour les 4 pistes en haut et en bas.



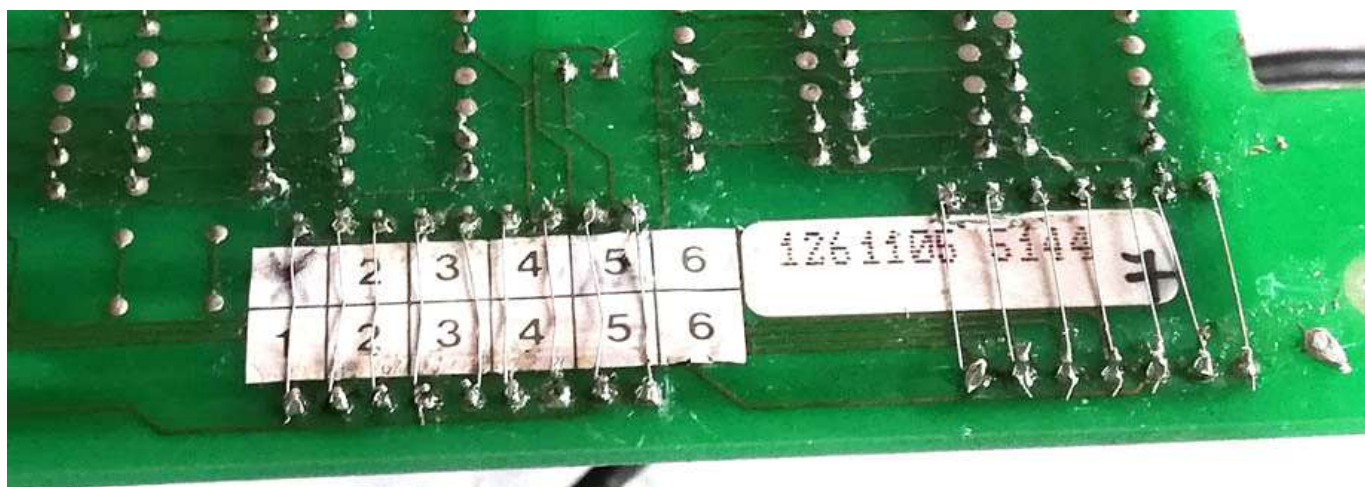
Sur un KVB, j'ai remplacé les leds jaune, par des nouvelles leds jaunes de diamètre 5 mm.

Le rendement est meilleur, et j'ai remplacé la résistance de 120 Ohms, par une de 390 ohms. La lampe LSSF ne consomme alors plus que 15 mA, au lieu de 50 mA.

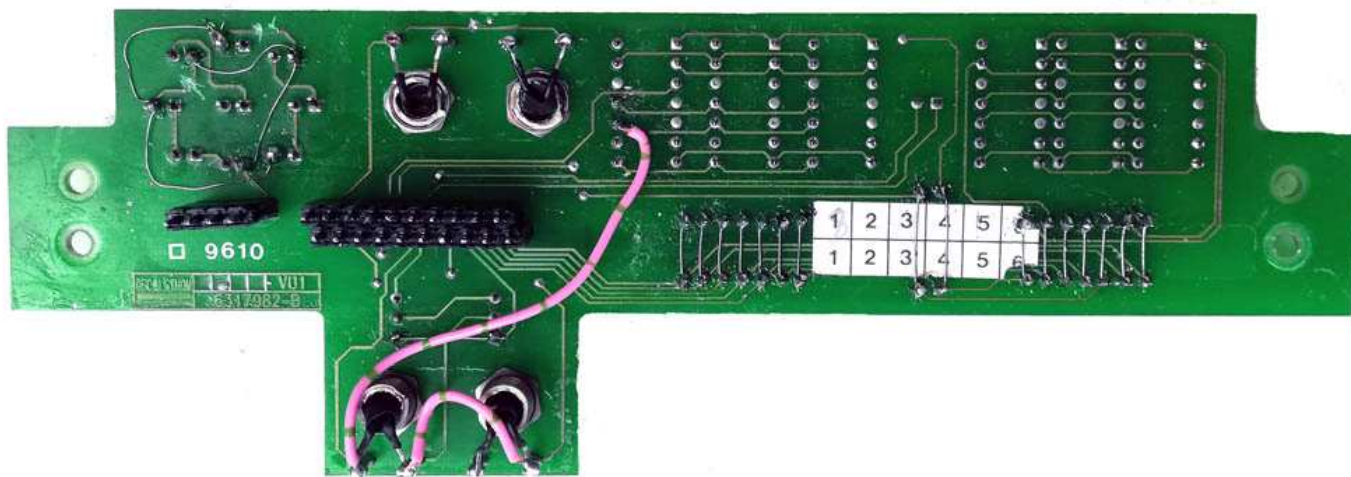
On n'a plus besoin des résistances, ni des diodes, pour limiter le courant dans les leds des afficheurs.

On soudera des fils sur les plots suivants : 9 fils à gauche et 7 fils à droite.

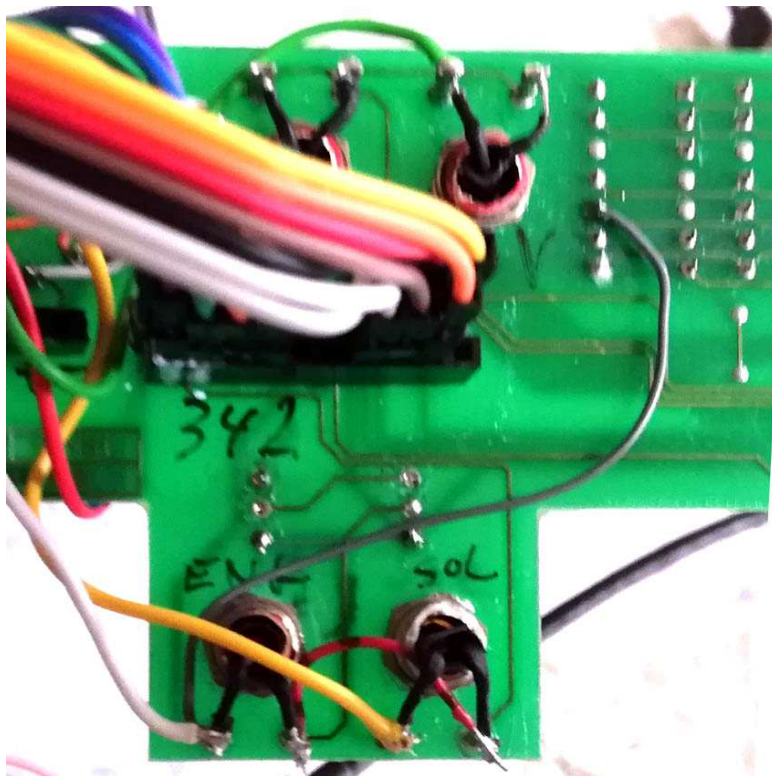
Remettre correctement de la soudure sur ces plots, car le circuit imprimé est vernis.



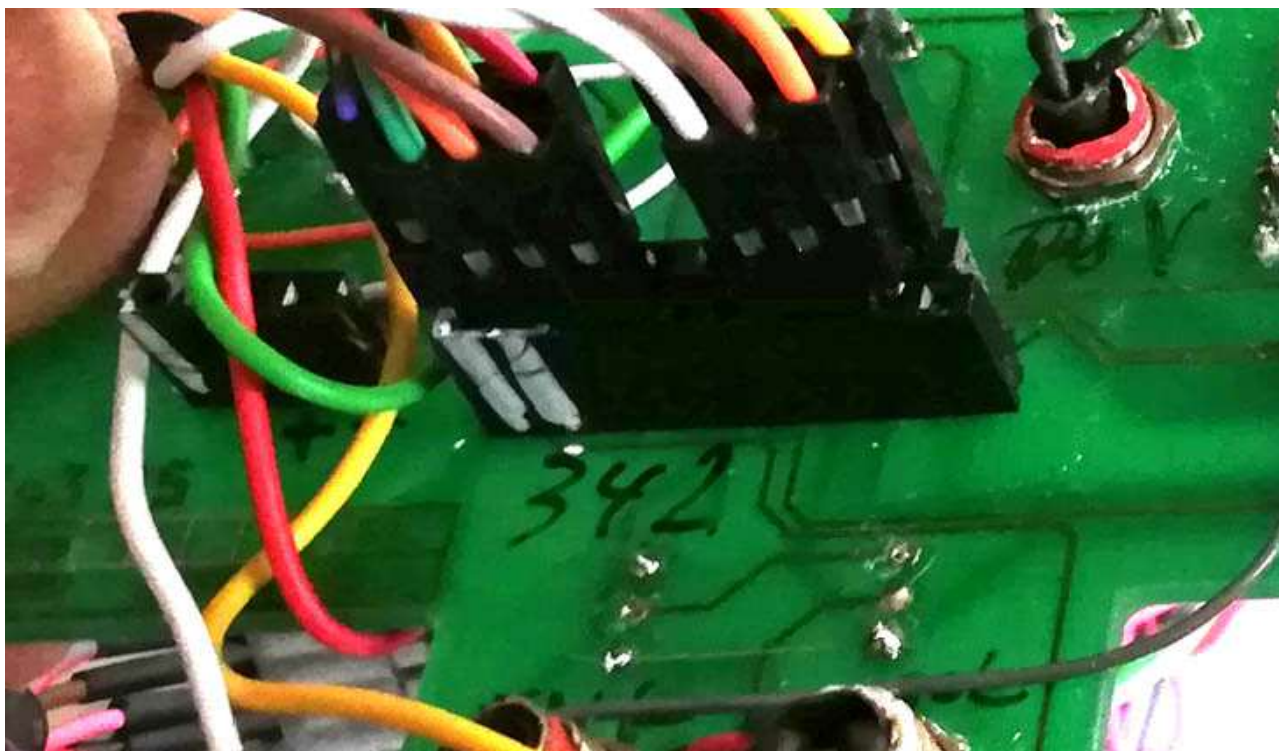
Avec une autre série de KVB, le principe est le même. Il faut aussi souder ces 16 fils verticaux.



La lampe ENGIN est actuellement câblée de façon indépendante sur ce circuit. On va la relier aux autres. Comme sur cette photo, relier les deux pattes de droite (+) des leds du bas ensemble (*fil bleu-rouge*). Relier la patte de gauche (-) de la led ENGIN au plot correspondant (*fil gris*). Souder aussi un dernier fil fin (*strap*) entre les deux plots du bas sur la photo (*Voir le plan de câblage*).



Souder deux connecteurs femelles parallèles en noir, plus un connecteur 4 plots à gauche :



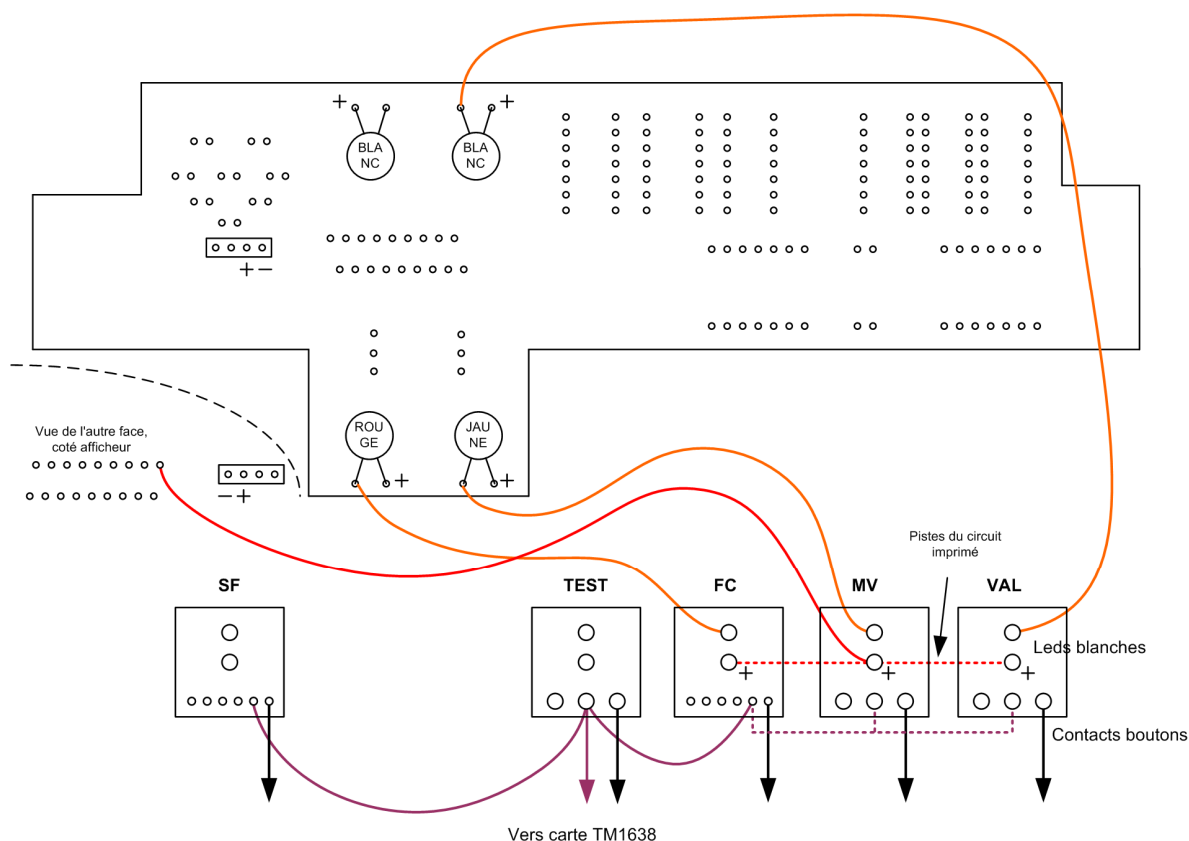
Câblage des leds d'éclairage des boutons poussoirs de la face avant :

Plan de soudure des fils. Si vous avez installé les leds différemment, le fil rouge doit être le fil commun au (+) de toutes les leds.

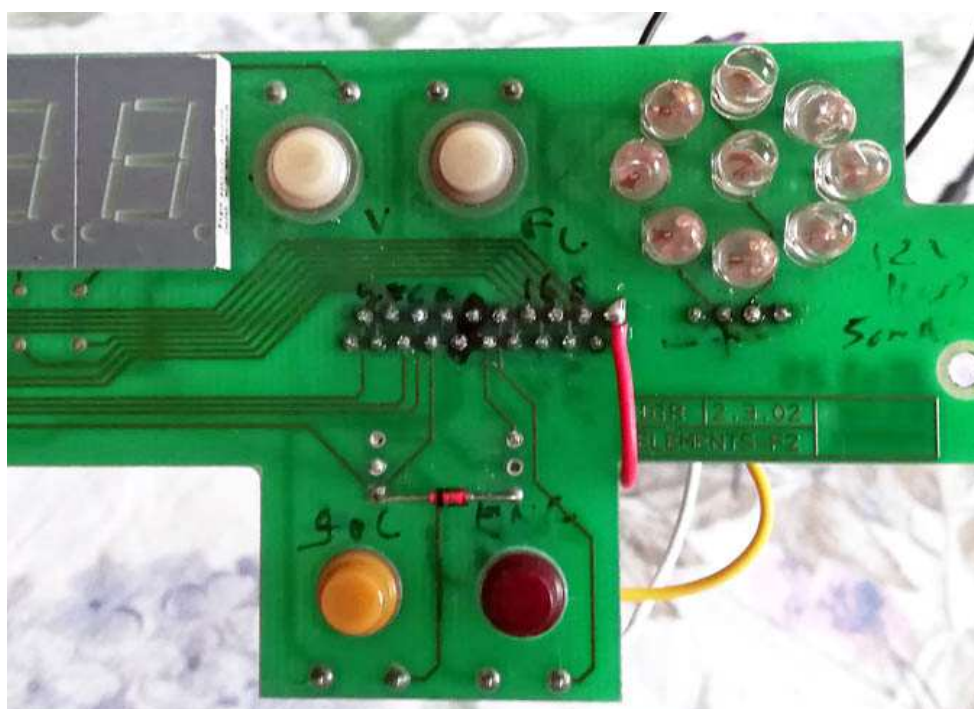
Le plan de câblage, pour une série de KVB, avec un circuit imprimé fixé aux boutons :

Respecter ce branchement.

S'il n'y a pas de circuit imprimé à l'arrière des boutons, il faut souder les 4 fils représentés en pointillé !

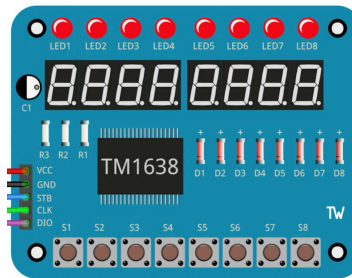


Câblage du fil commun (+) des leds des boutons poussoir. Le fil rouge arrive de l'autre coté du circuit imprimé :



Branchement à la carte TM1638 :

Acheter une carte de type : HW-154 avec un TM1638, deux afficheurs pour 8 digits, 8 leds et 8 boutons.



Dessouder les deux afficheurs, pour avoir un circuit tout propre, avec une pompe ou une tresse à dessouder.

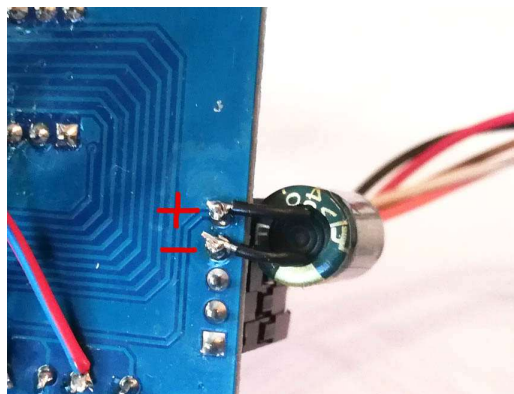
<https://fr.aliexpress.com/w/wholesale-pompe-%C3%A0-dessouder-%C3%A9lectrique.html>

Dessouder aussi toutes les leds rouges. Par contre, on garde les boutons poussoirs en place.

Souder quatre supports aux emplacements des afficheurs.

Souder impérativement, un condensateur de 470 μ F / 16 V sur les broches d'alimentation 5 Volts.

Ce circuit consomme un courant important en pointe, et plante sans ce condensateur.



Branchement des afficheurs, des voyants et des leds des boutons :

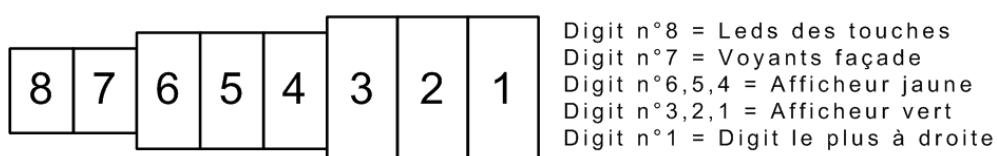
Le TM1638 commande 8 afficheurs. On utilisera 6 afficheurs pour les 6 digits.

Les voyants sont branchés en place de l'afficheur n° 7. Pour allumer les quatre voyants, on allume les segments 'a', 'b', 'c' ou 'e'

Les leds d'éclairage des boutons sont branchées en place de l'afficheur n° 8. Pour allumer les trois touches, on allume les segments 'a', 'b', ou 'c'.

.

Vue face avant de la façade



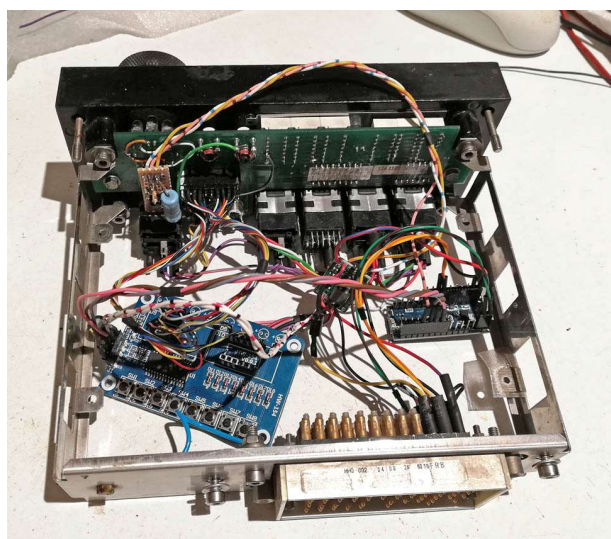
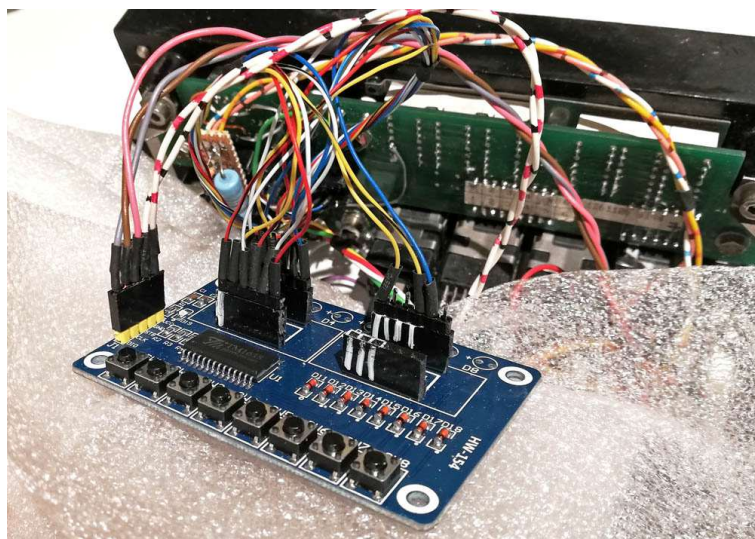
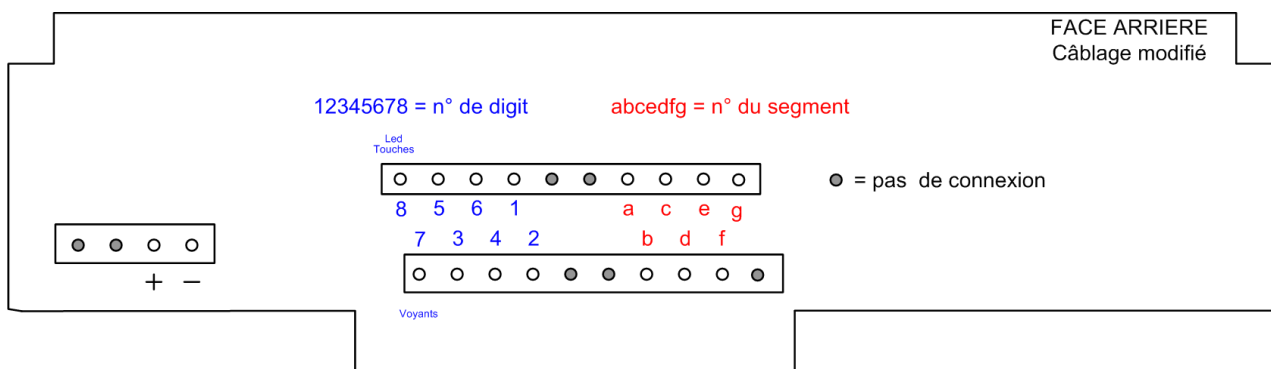
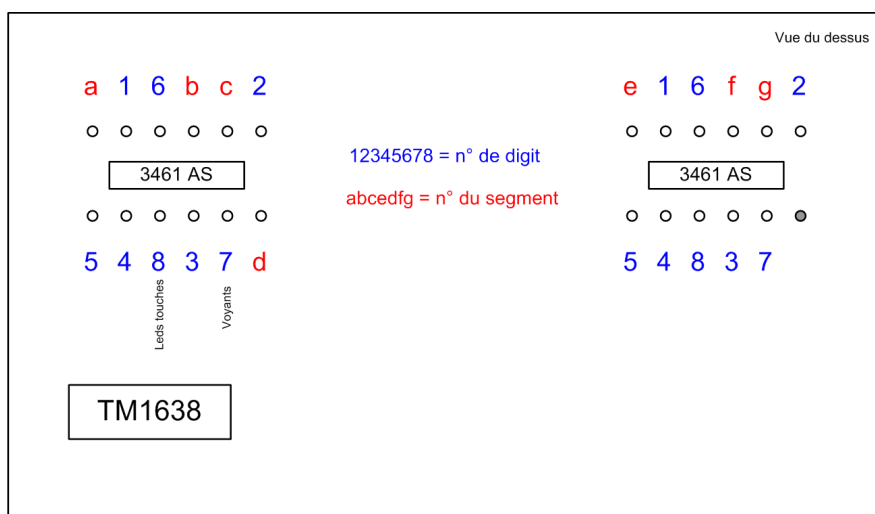
Branchement à la carte TM1638 : La partie afficheur, voyant et led des boutons :

Il faut relier les plots de la carte TM1638 au circuit imprimé du KVB.

Il faut relier ensemble, les plots de '1' à '8' et de 'a' à 'g'.

Les plots '1' à '8' sont en double, on se branche à droite ou à gauche.

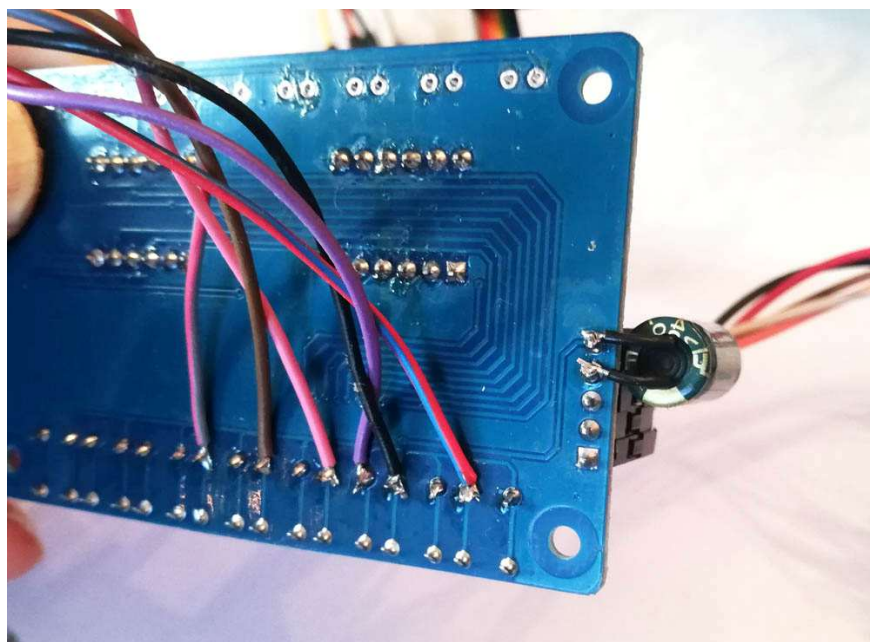
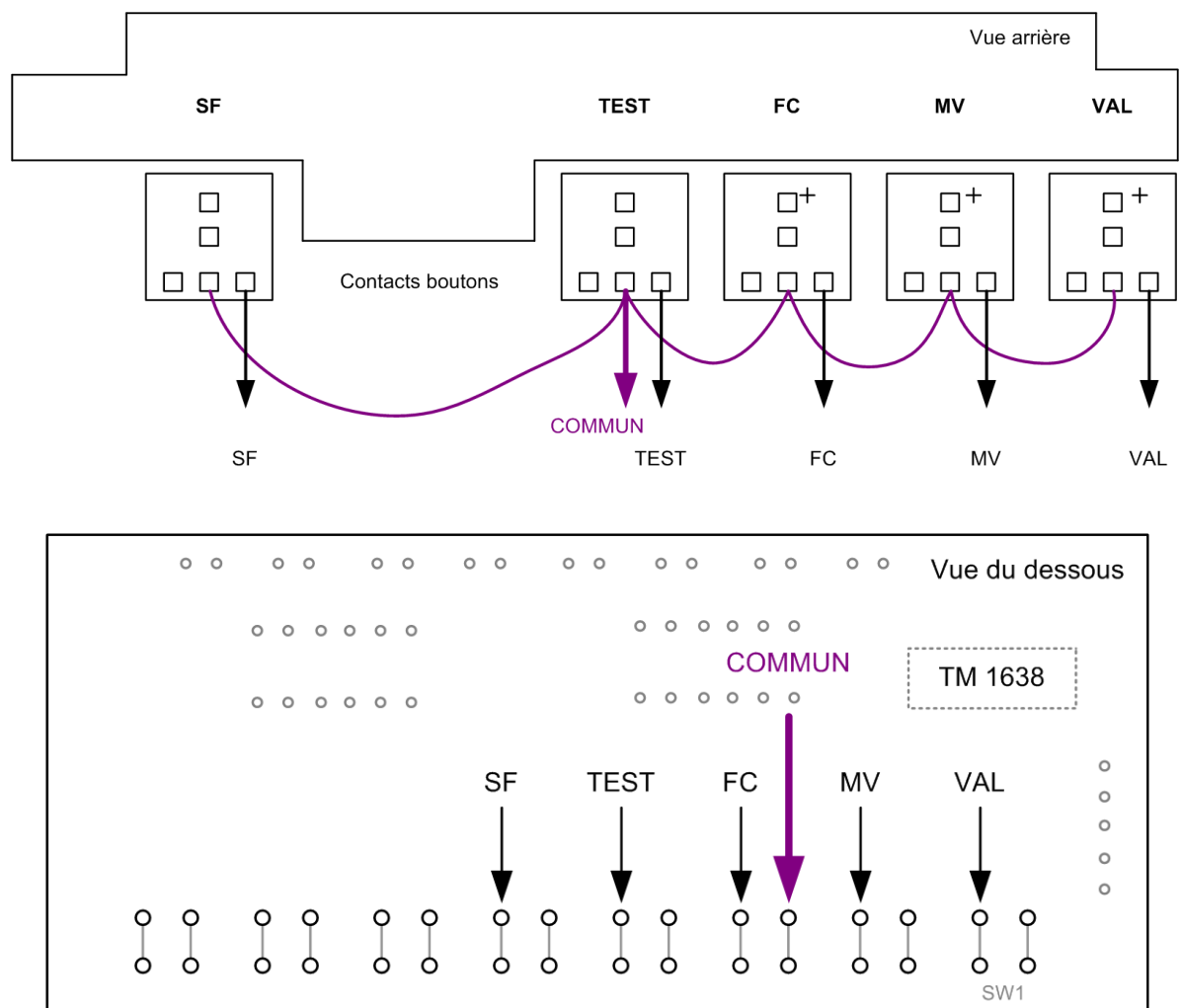
Ce câblage, c'est le jeu des 7 erreurs.



Branchement à la carte TM1638 : La partie contact des boutons :

Il faut relier les plots de la carte TM1638 aux boutons du KVB.

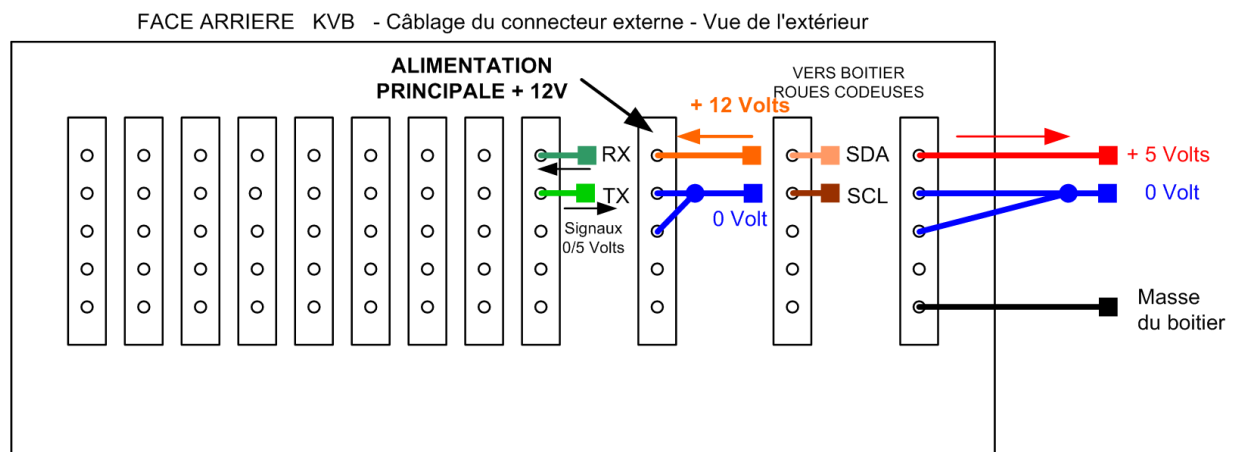
Il n'y a pas besoin de dessouder les boutons poussoirs de la carte TM1638.



La connexion externe au KVB :

On branchera le + 12 Volts.

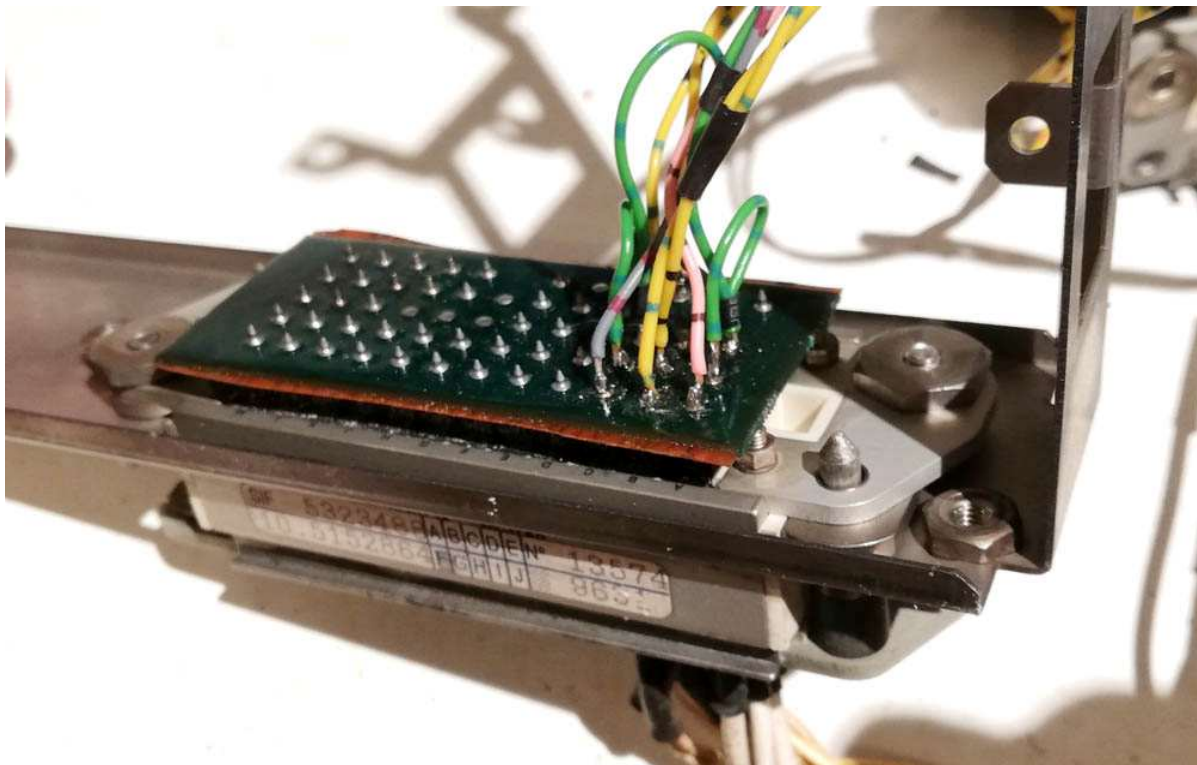
Le plot +5 Volts est une sortie pour alimenter le boîtier de roues codeuses.



Sur cette série de boîtier KVB, il faut couper la nappe marron.

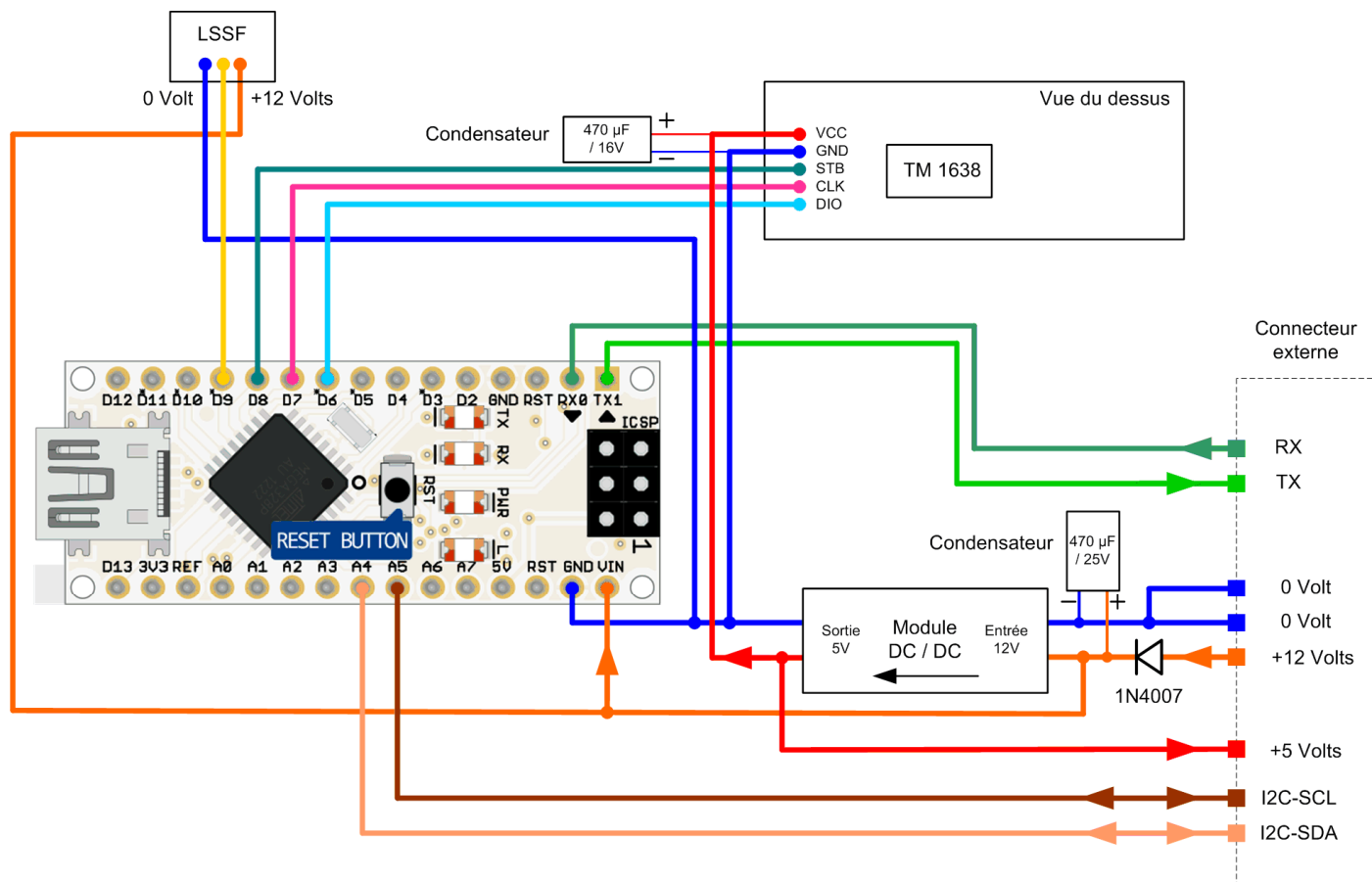
Attention, sur ce modèle j'ai eu beaucoup de problème, car le circuit imprimé relie des plots entre eux, et la nappe marron fait parfois court-circuit. J'ai eu des fonctionnements aberrants, ou l'appui de touche faisait afficher n'importe quoi.

Pour bien faire, il faudrait tronçonner le circuit imprimé à la disqueuse autour des plots utilisés.



Attention de ne pas mettre du 12 Volts sur un plot adjacent !

Le schéma de câblage interne du KVB avec l'Arduino :



Avant de tout souder, vérifier que le module DC/DC délivre bien du 5 Volts.

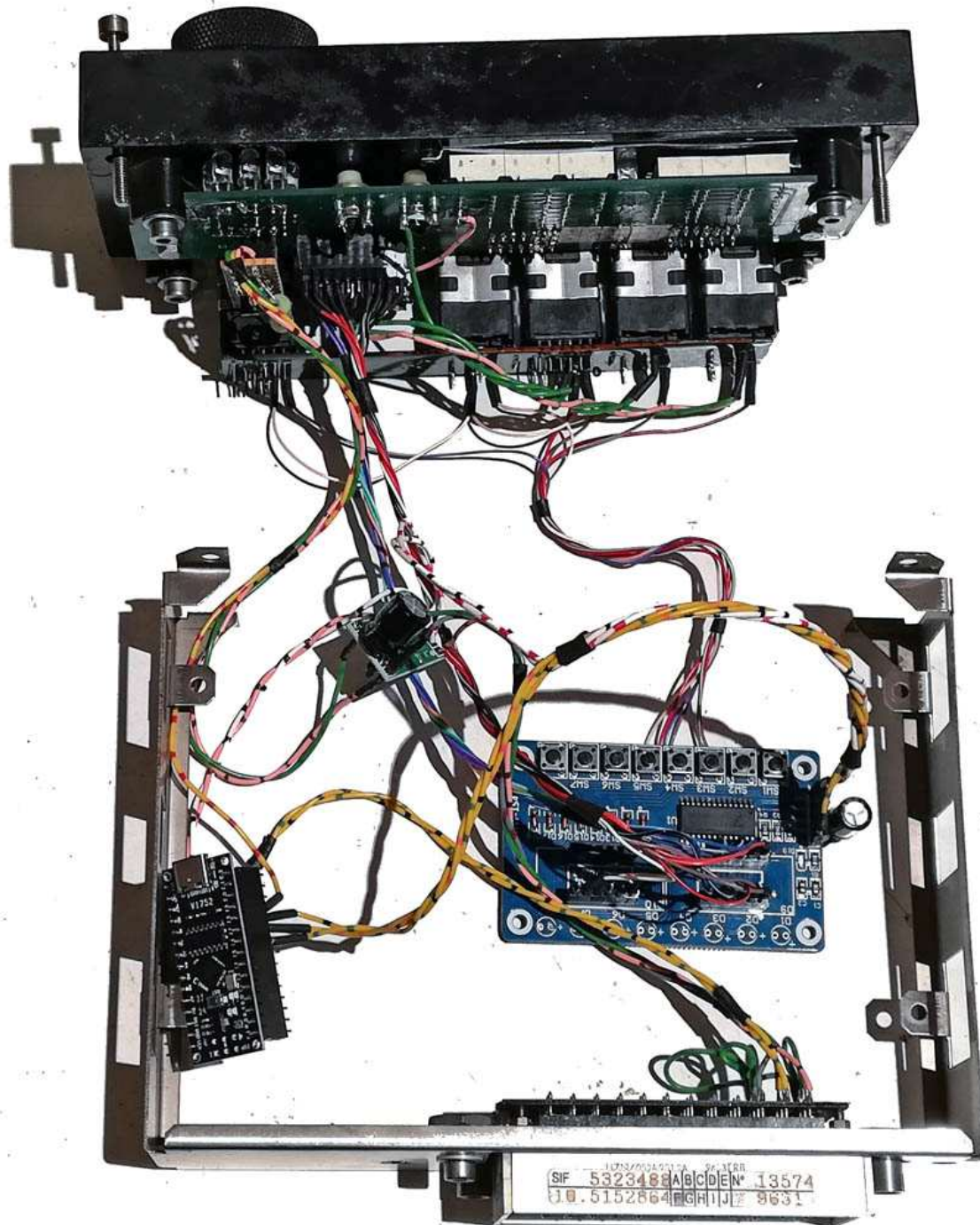
La diode 1N4007 évite les erreurs de polarité. Les condensateurs sont indispensables au bon fonctionnement. Le module DC/DC converti le +12Volts en +5 Volts, minimum 1 Amp. Consommation : 0,2 Amp sous 12 Volts.

Le montage est très compact.

Attention pendant les tests, à ce que les circuits ne se touchent pas.

On peut aussi appuyer sur les boutons du module TM1638, pour tester les touches.

En cas de problème, on peut connecter un module TM1638 non modifié, pour vérifier la communication avec l'Arduino, qui doit envoyer un 'o' sur la ligne série. Dans ce cas, l'affichage est animé, mais donne n'importe quoi.



Une fois testé, chauffer les gaines thermo rétractables restantes, et mettre rapidement tous les modules dans une mousse ou boîtier isolant.

LE BLOC DE ROUES CODEUSES

Pour configurer le KVB, on trouve à coté un boîtier équipé de roues codeuses. Le boîtier KVB fonctionne très bien sans, et le programme de l'Arduino n'a pas besoin d'être modifié, avec ou sans ce boîtier.

Le programme de l'Arduino NANO utilisera automatiquement ce boîtier, s'il est relié au boîtier KVB.



Avec ces roues codeuses en façade, on renseigne

V = Vitesse maximale de la circulation, V, en dizaines de km/h,

L = Longueur du train, L, en centaines de mètres,

D = Décélération, D, en centièmes de m/s^2 (ou cm/s^2),

Un bouton rotatif = VOyageur, MEssagerie, MArchandise

Les roues codeuses d'origine sont codées en décimal = 10 plots de sortie par roue.

Le circuit imprimé derrière les roues codeuses est câblé en multiplexage, les 10 plots des numéros sont connectés ensemble.

Il serait compliqué de lire les 60 contacts directement, on reste donc sur la solution multiplexée.

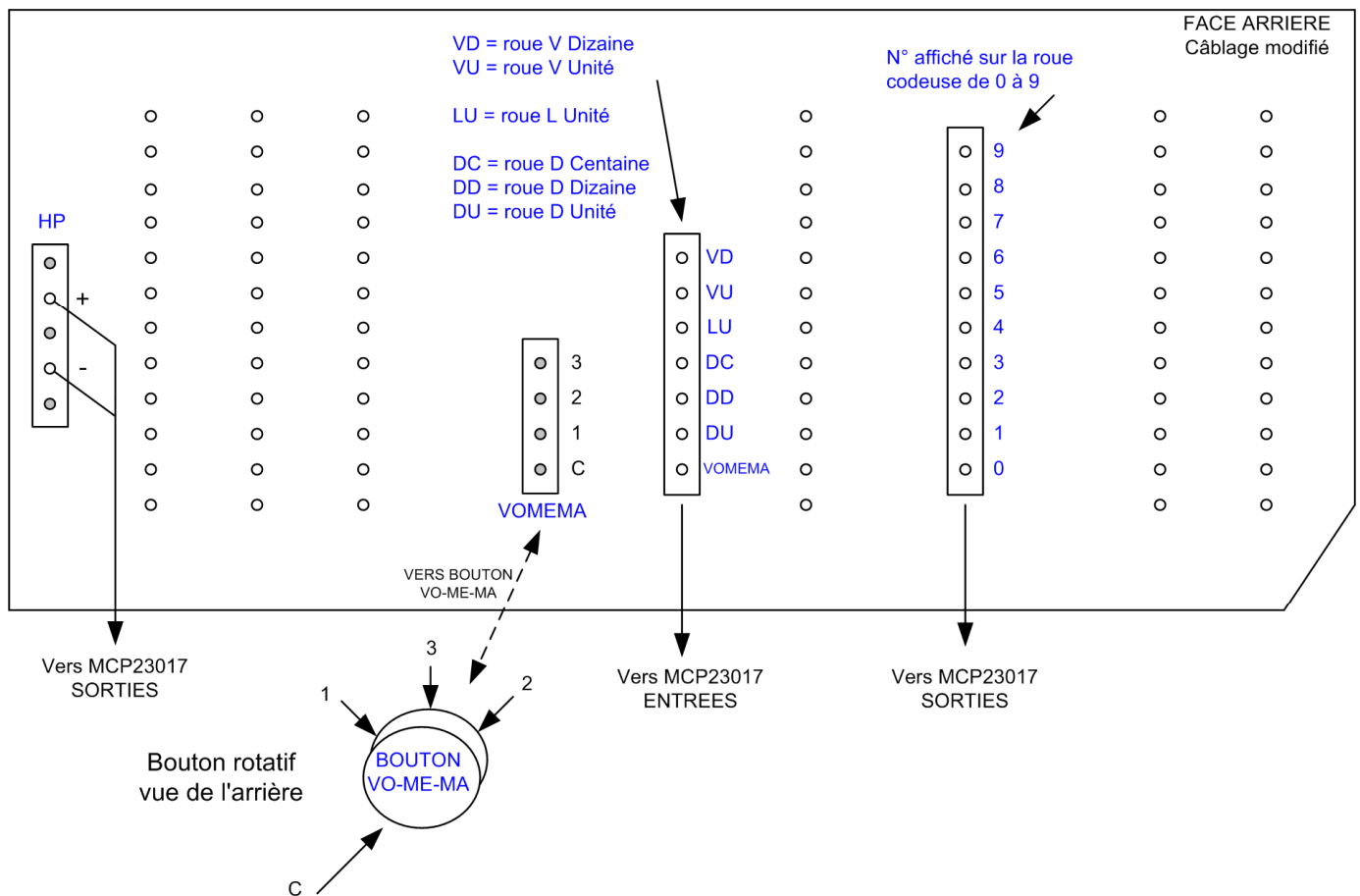
Pour limiter le nombre de fils en ce boîtier et le boîtier KVB, on va installer dans ce boîtier un périphérique I2C.

En gardant les fils d'origine, le câblage est très facile à réaliser.

Il existe au moins deux versions de ce boîtier.

Sur une autre version, les plots des n° affichés sont à droite du circuit imprimé, et les N° de roues, sont complètement à droite du circuit imprimé. L'ordre des broches est identique.

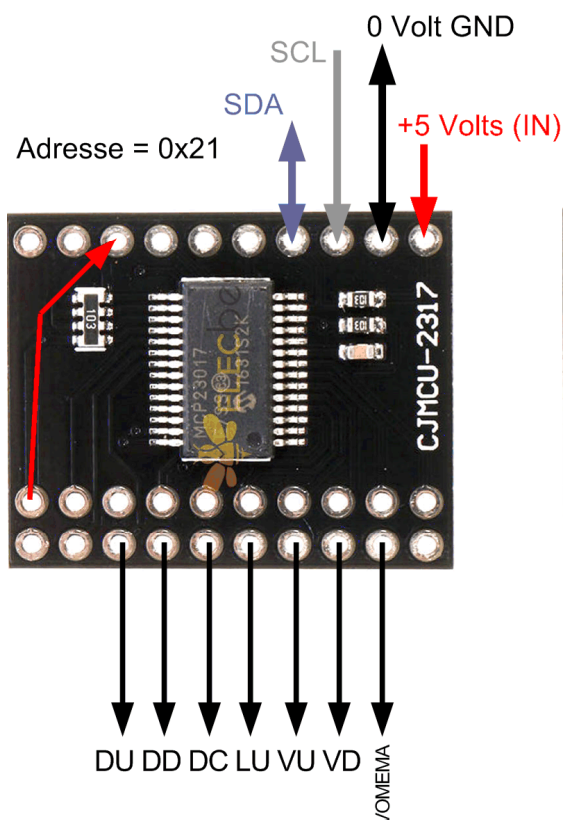
Le schéma du circuit imprimé des roues codeuses :



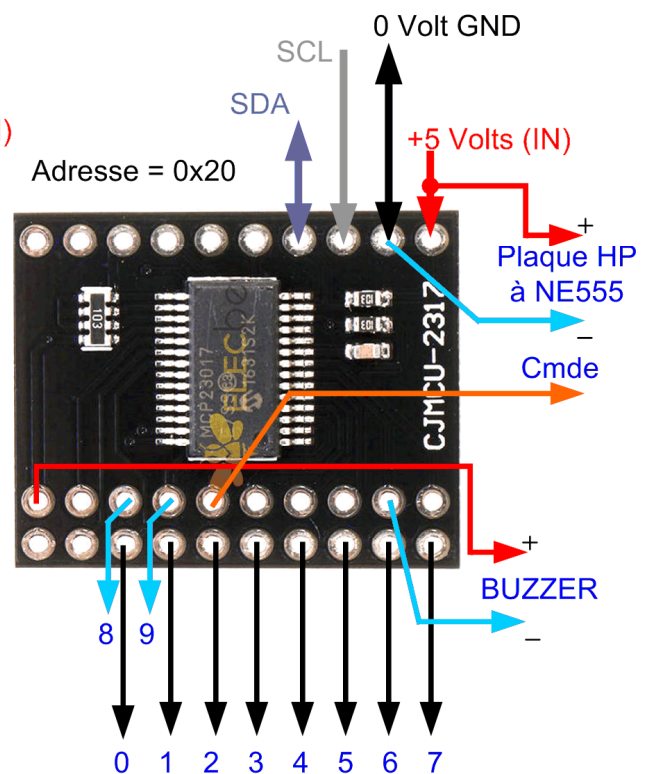
J'utilise un MCP23017 en sortie, pour alimenter successivement les 10 numéros (plots) des roues codeuses.

J'utilise un second MCP23017 en entrée, pour lire l'information de chaque roue codeuses.

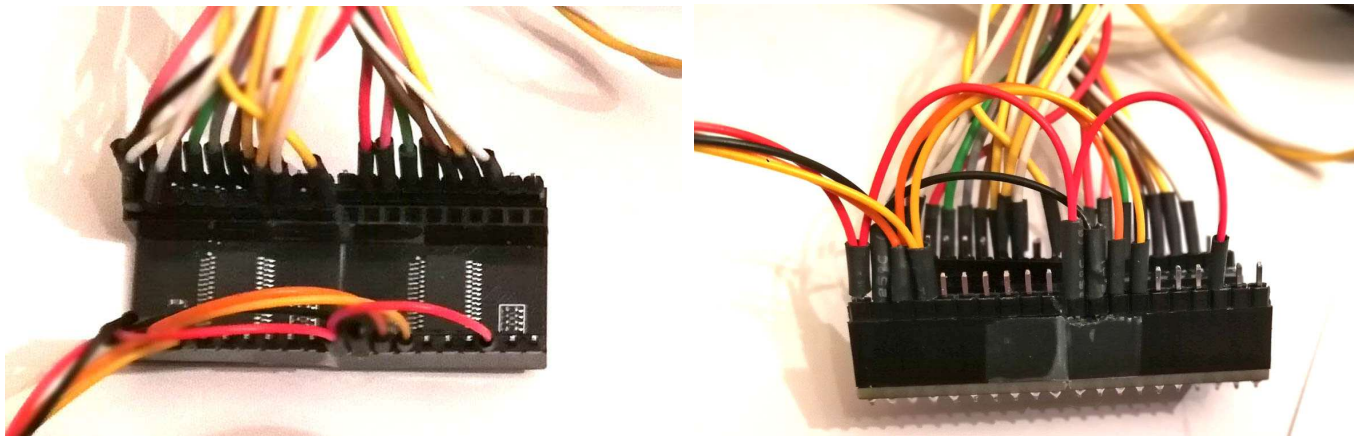
MCP23017 - ENTREES



MCP23017 - SORTIES

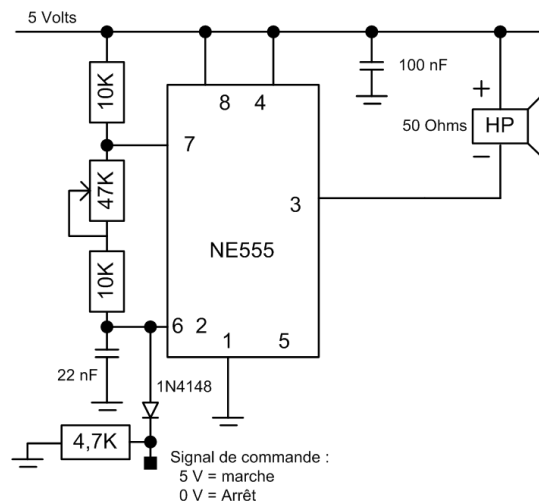


On utilise des connecteurs pour les branchements internes :



Le buzzer gris est alimenté en 5 Volts continu. Il est polarisé (*un fil + et un fil -*). Il produit un son modulé. Il est branché entre le +5 Volts et la sortie n° 15 (B6) du MCP23017.

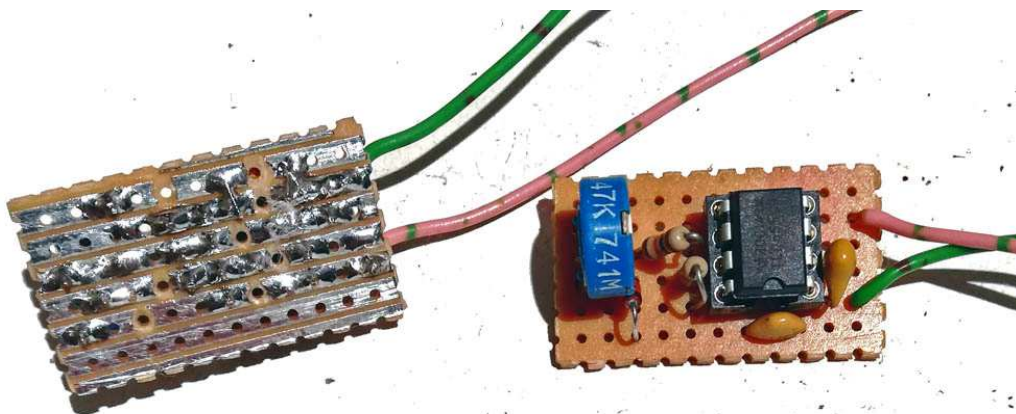
Pour le haut-parleur, construire un petit montage type sirène, commandé depuis la sortie n° 13 (B5) du MCP23017 des sorties.



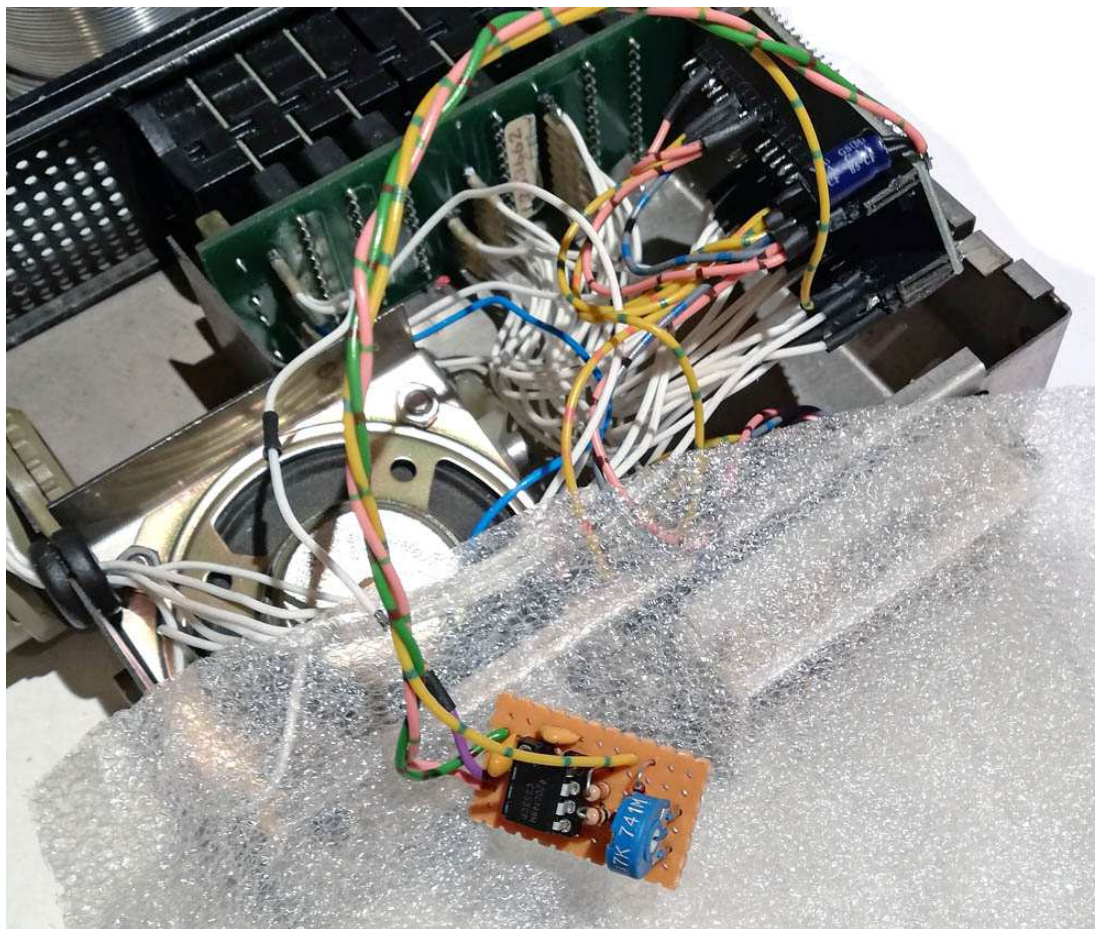
Le montage tient sur une petite plaque. Brancher les fils du haut-parleur, depuis le circuit des roues codeuses.

Le potentiomètre fait varier la fréquence du son. Le signal de commande provient du MCP23017 des sorties, broche B5.

Photos avant la soudure des fils de liaison Br2-Br6 et Br4-Br8.



Avant emballage :



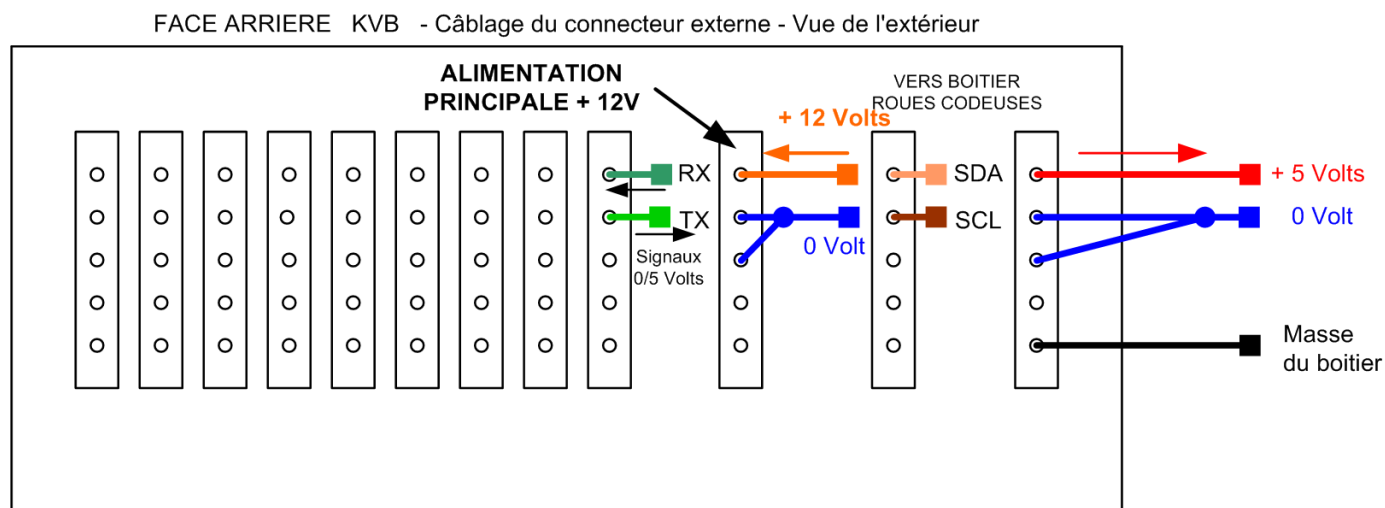
Prendre un NE555 qui fournit plus de courant et pas un TLC555.

La connexion externe :

Entre le boîtier KVB et le boîtier des roues codeuses, on a seulement quatre fils à brancher. Ci-dessous, un exemple de branchement utilisant le connecteur existant.

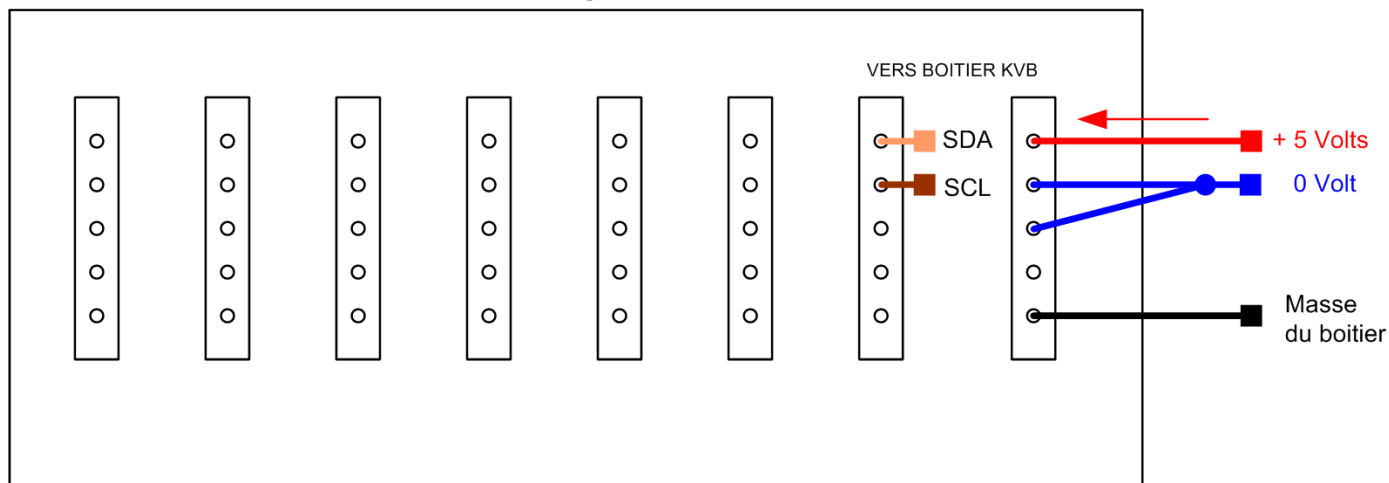
Pour la masse, utiliser deux broches pour sécuriser ce contact.

Par contre pour les signaux SDA et SCL, n'utiliser qu'une broche pour limiter les capacités parasites.



La face arrière du boîtier principal, pour réaliser les interconnexions :

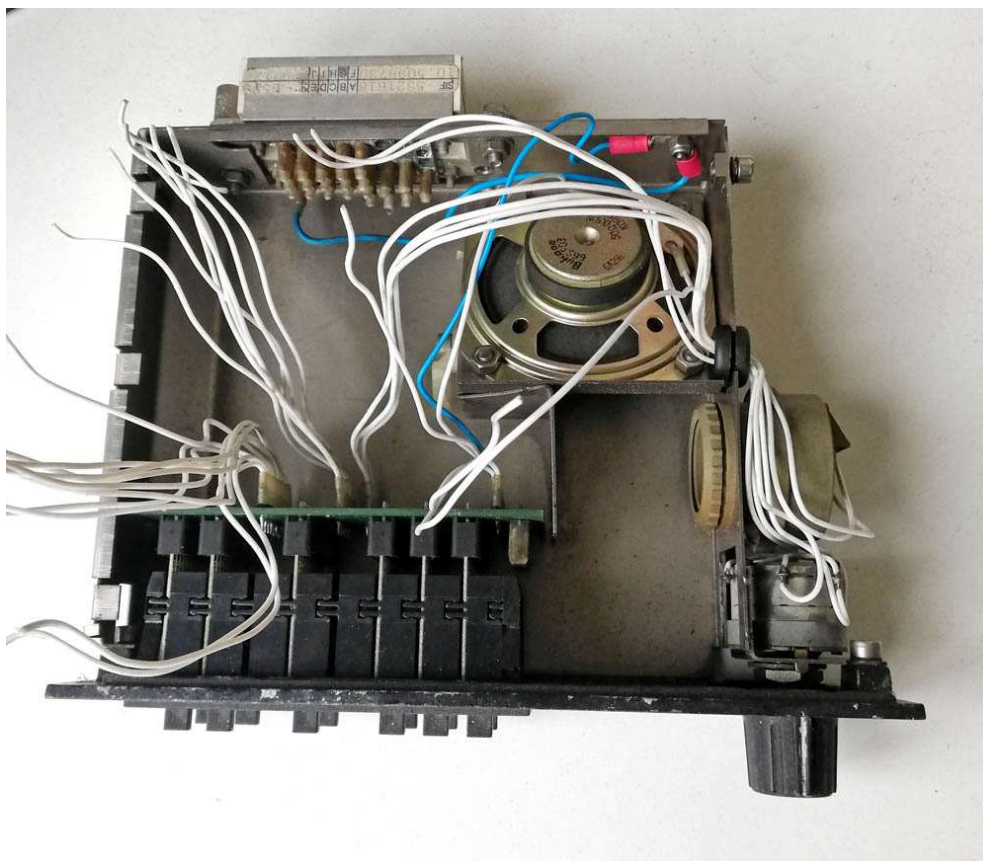
FACE ARRIERE ROUES CODEUSES - Câblage du connecteur externe - Vue de l'extérieur



Il faudra relier ces 5 broches aux broches du boîtier principal.

Mode opératoire

Couper tous les fils, sauf le **bleu**, à raz du connecteur.

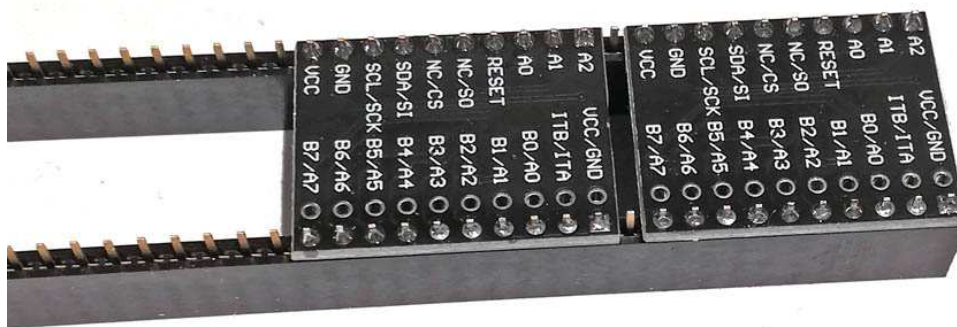


Ne pas couper le fil bleu relié à la masse du boîtier. Pour dénuder ces fils, il faut faire tourner la pince coupante, avant de tirer dessus. Ils sont en téflon et argentés, de très grande qualité.

Prendre deux modules MCP23017 et les souder sur des barrettes.

Le MCP23017 des entrées n'a pas de barrette sur le port B !

Sur la photo, la dernière barrette pour le MCP des sorties du port B, n'est pas encore en place.



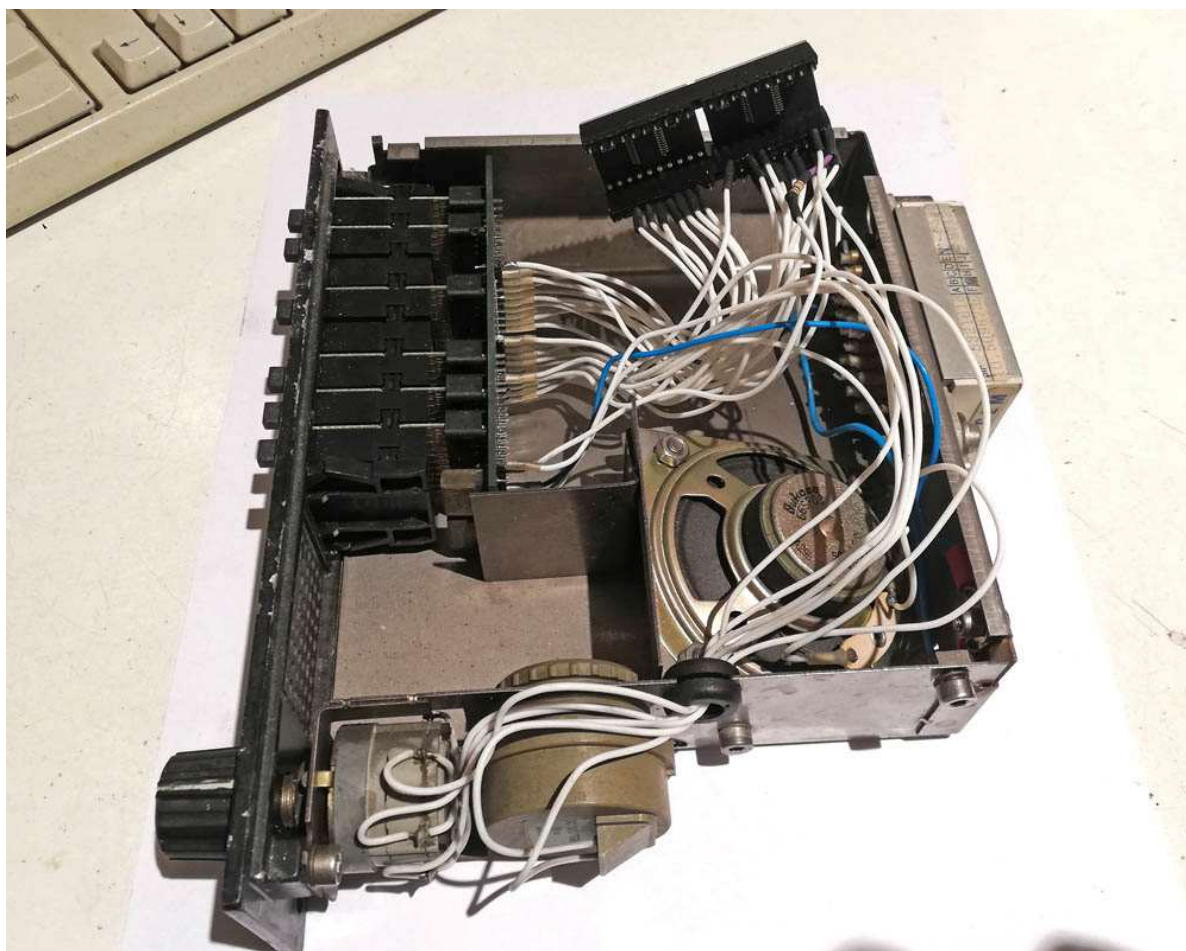
Insérer les picots de 10 broches mâles sur les barrettes.

Repérer les fils en sortie du circuit imprimé, et les souder sur les picots (*Voir schéma précédent*).

Tous les fils blanc sont directement utilisables. Les numéros, le type de roues et il y même la sortie HP (+) et (-).

Ne pas rétrécir les manchons de gaine thermo-rétractable tout de suite, on le fera une fois le boîtier testé.

Pour le bouton rotatif VO-ME-MA, il y juste besoin de souder le fil blanc 'vomema', sur le module MCP.



Attention, l'ordre des fils VD-VU-LU-DC-DD-DU-VOMEMA, n'est pas le même entre le circuit imprimé et le module MCP23017.

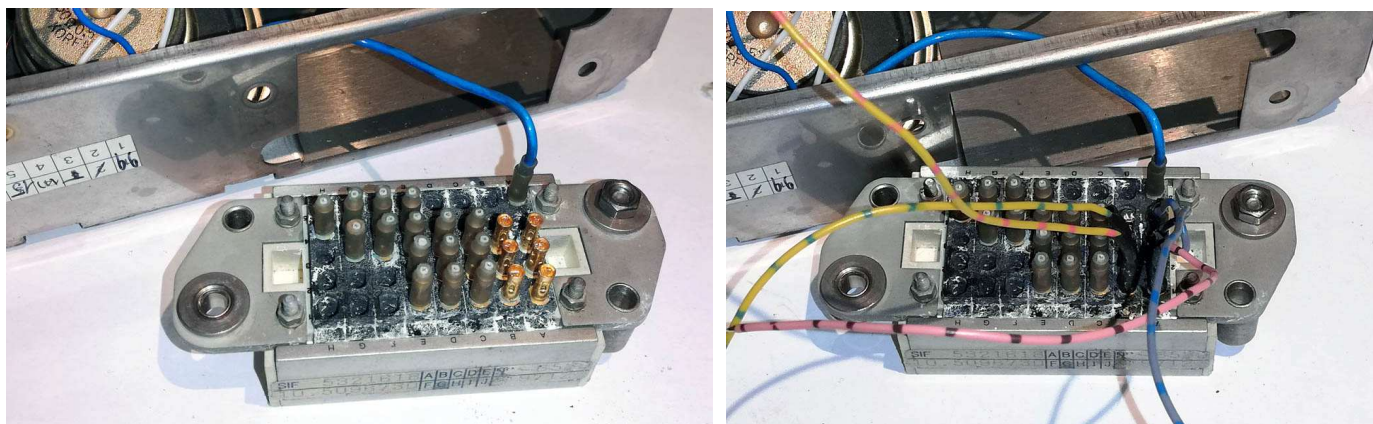
Pour le moment, on n'a pas eu besoin d'utiliser de nouveaux fils, on a utilisé ceux existants.

Repérer les fils du buzzer. Le fil (-) du buzzer va sur le circuit MCP23017.

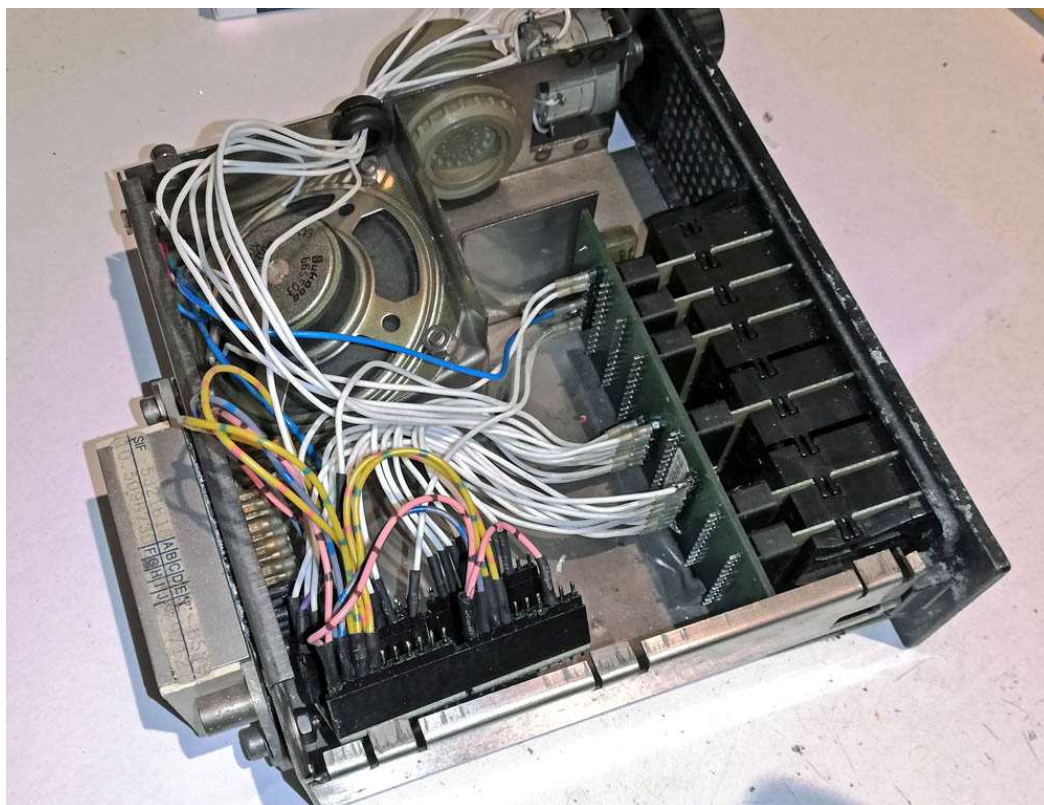
Les fils du haut-parleur proviennent du circuit imprimé des roues codeuses.

Dévisser le connecteur, retirer la gaine des 6 plots.

Souder les fils sur les plots, en prenant deux plots pour le 0 Volt.



Souder ces fils sur les modules.



Dépannage :

Vérifier le +5 Volts sur les broches VCC.

Pas de chiffres d'affiché à la mise sous tension = Inversion probable des signaux SDA et SCL.

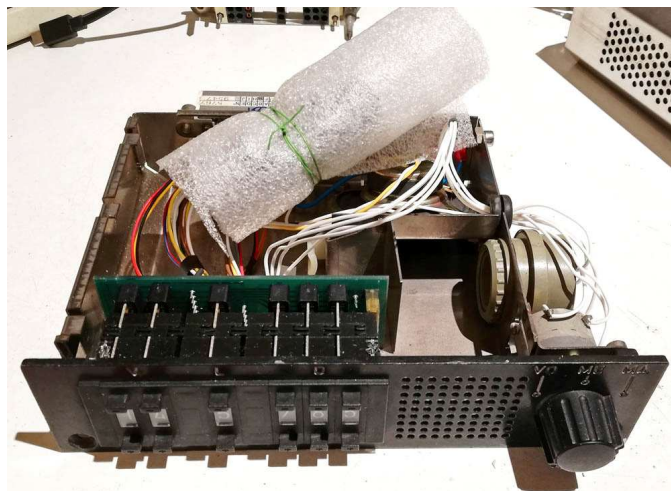
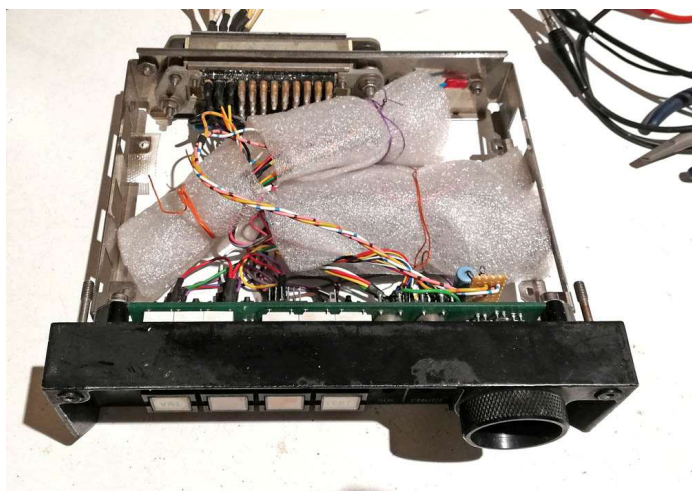
Les chiffres s'affichent mais sont incohérents = Inversion d'adresse 0x20 et 0x21 entre les deux modules.

Les autres anomalies sur les chiffres proviennent de l'ordre de câblage erroné des chiffres ou des roues.

Ne pas oublier de souder un condensateur de 220 μ F / 16 Volts, sur les plots GND (-) et VCC (+)
Ca limitera les baisses de tension, à l'activation du haut-parleur.



J'ai utilisé film plastique d'emballage mousse, pour entourer les circuits, tout en gardant un peu de ventilation.



La consommation des boîtiers KVB + RC est de 0,2 Amp maximum sous 12 Volts.
Sans rien d'affiché, la consommation est de 20 mA.
Le signal le la liaison série RT/TX en entrée/sortie du boîtier est en 0/5 Volts.

LA PARTIE LOGICIELLE V1.2

Ce document et le programme 'KVB_Arduino_JLF.ino' se trouvent ici : http://www.la-tour.info/uts/uts_page15.html

Le programme a été testé sur un Arduino NANO. Le programme simule le fonctionnement interne du boîtier KVB.

Ce logiciel est un logiciel libre. Exigence du concepteur : Ne pas modifier la routine d'intro. A la mise sous tension, affiche "JLF xx/xx/xxxx" sur écran pendant 1 secondes.

On peut modifier ce programme et le diffuser. Dans ce cas, il faut préciser l'origine et donner accès aux sources modifiées.

ATTENTION : J'ai utilisé la librairie TM1638 de Ricardo Batista : <https://github.com/rjbatista/tm1638-library>

J'ai modifié ces fichiers pour pouvoir allumer directement les afficheurs et les voyants du KVB.

Les programmes : TM1638QYF_JLF.h, TM1638QYF_JLF.cpp, TM163XXFonts_JLF.h, TM18XX_JLF.h et TM16XX_JLF.cpp sont modifiés. Ne pas les mettre à jour.

Cette librairie est l'une des rares prévues pour des afficheurs à Anode commune, comme celle du KVB original.

Je l'ai développé pour qu'un caractère ascii reçu corresponde à un ordre à exécuté sur le KVB.

De même, l'appui d'une touche du KVB, envoie un caractère ascii sur la liaison série.

Ce fonctionnement est suffisant, lorsque le KVB sera relié à un autre ARDUINO, lui-même relié au simulateur de conduite.

Si l'on utilise un logiciel d'interface comme TSClasicInterface,

<https://www.univers-simu.com/wp-content/uploads/2024/02/Demarrage-Rapide-FR.pdf>

<https://forums.dovetailgames.com/threads/ts-classic-raildriver-and-joystick-interface.72488/>

on peut reprendre la partie d'envoi et de réception sur la liaison série, pour s'interfacer avec le KVB.

Si l'on veut relier ce KVB directement à un ordinateur, il faudrait peut être prendre un autre type d'Arduino simulant un clavier.

Ce programme fonctionne pour un boîtier KVB seul, mais aussi pour un boîtier KVB et un boîtier de roues codeuses.

Il n'y a pas de modification à apporter au programme, il détecte tout seul la présence du boîtier des roues codeuses.

Ce programme peut aussi servir dans le cas d'un nouveau boîtier KVB, créé de A à Z.

La vitesse de la liaison série est de 9600 bps, et peut être modifiée dans le code.

A la mise sous tension, si le module TM1638 n'est pas détecté, on ne peut rien afficher. Dans ce cas la lampe LSSF clignote rapidement.

A la mise sous tension, le KVB affiche une intro, puis les 6 chiffres des roues codeuses si le boîtier des roues codeuses est présent.

L'appui de la touche [VAL] pendant plus de 3 secondes, permet de modifier la luminosité des afficheurs et des voyants. C'est aussi un moyen de faire un reset du KVB, mais le reset ne fonctionne pas avec tous les Arduino NANO.

La valeur de la luminosité est sauvegardée, mais il faut ensuite remettre l'Arduino hors + sous tension.

Fonctions de base

Le KVB affiche les pictogrammes suivants à la réception d'un caractère reçu sur la ligne série RX0 :

```
r => [JLF][ ] Reset logicielle Arduino
R => [ ][ ] Eteint tous les afficheurs, voyants, touches, HP et buzzer
b => [ ][ b ]
c => [ ][ ] Clear : Efface que l'afficheur [ ][ ]
e => [ ][ ==] Egale : 3 tirets
F => [ ][ F ] Faux
l => [ L ][ ]
L => [ ][ L ]
m => [---][---] Moins Fixe
M => [---][---] Moins Clignotant
p => [ p ][ ]
P => [ ][ P ]
t => [888][888] Affiche 888 888
U => [ ][ FU] fU
y => [ 00][ ] 2 Zéro
Y => [000][ ] 3 Zéro
x => [---][ 00] Moins + 2 Zéro Fixe
X => [ 00][000] 2 Zéro + 3 Zéro Fixe
z => [ ][ 00] 2 Zéro Clignotant
Z => [ ][ 00] 2 Zéro Fixe
n => [ PA][400] Affichage = PA400 + Buzzer. Spécial pour BB7200 de SimExpress
N => { UC}[512] + Lampes (SOL) et (ENGIN) allumées + HP. Spécial pour SimExpress
o => Lampes (SOL) et (ENGIN) allumées + Buzzer + HP. Spécial SimExpress
O => [ ][ ] + Lampes (SOL) et (ENGIN) éteintes + Stop Buz+HP. Spécial SimExpress
s => Séquence d'initialisation du KVB, à la mise sous tension du pupitre
T => Séquence de test complet des voyants, afficheurs, alarmes sonores pendant 3 secs
```

Ces ordres ne sont pas cumulables, un ordre concerne l'affichage complet des deux parties des afficheurs.

Exemple, la réception de 'l' puis 'L' donne [][L].

On peut demander si le KVB est prêt. Il renvoie alors un caractère.

? => Demande si le KVB est prêt ? Réponse = 'o' pour ok.

Réponse = 'n' pour nok, si la carte TM1638 n'est pas détectée.

On peut demander au KVB, les numéros affichés sur les roues codeuses.

! => Demande le n° des roues codeuses + position du bouton VO-ME-MA

En retour renvoie 7 chiffres entre les balises '<>'. Exemple '<1234561>' ou '<0000000>' si pas de boîtier de RC.

Attention, ce n'est pas le dernier nombre validé par le bouton [VAL], mais la position actuelle des RC.

Le dernier chiffre = 1:VO, 2:ME, 3:MA.

On peut demander la version du KVB.

v => Demande la version du KVB

En retour, renvoie JLF_Vx.y_jj/mm/aaaa

Les voyants à la réception d'un caractère reçu sur la ligne série RX0 :

```
0 => (V)   Eteint      1 => (V)   Allumée
2 => (FU)  Eteint      3 => (FU)  Allumée
4 => (SOL) Eteint      5 => (SOL) Allumée
6 => (ENG) Eteint      7 => (ENG) Allumée
8 => Eteint toutes les lampes et LSSF, mais pas les touches
g => [VAL] Eteint      G => [VAL] Allumée      A => [VAL] Clignotant
h => [MV]  Eteint      H => [MV] Allumée
i => [FC]  Eteint      I => [FC] Allumée
j => (LSSF) Eteint     J => (LSSF) Fixe        C => (LSSF) Clignotant
9 => Eteint toutes les touches, mais pas les voyants ni LSSF
```

Le haut-parleur et le buzzer du boîtier des roues codeuses, sont gérés indépendamment :

```
k => Arrêt klaxon      K => Klaxon actif sur le haut-parleur
q => Arrêt buzzer      Q => Buzzer actif
```

La fonction TEST :

```
T => Séquence de test complet des voyants, afficheurs, alarmes sonores pendant 3 secs
```

Code des touches envoyées sur la liaison série. On envoie un caractère par touche enfoncée ou relâchée :

```
[VAL] => 'A' enfoncée, 'a' relâchée
[MV]  => 'B' enfoncée, 'b' relâchée
[FC]  => 'C' enfoncée, 'c' relâchée
[TEST] => 'D' enfoncée, 'd' relâchée
[SF]  => 'E' enfoncée, 'e' relâchée
```

Si besoin, on peut supprimer le code de touche relâchée, ou ignorer ces codes de touche en minuscule.

Fonctions supplémentaires

A la mise sous tension, si le module TM1638 n'est pas détecté, on ne peut rien afficher. Dans ce cas la lampe LSSF clignote très rapidement. De toute façon le KVB sera inutilisable.

A la mise sous tension, le KVB affiche une intro, puis les 6 chiffres des roues codeuses si le boîtier des roues codeuses est présent.

A la mise sous tension, quand le KVB est prêt, il envoie un caractère.

```
KVB pret => 'o' pour ok.
           'n' pour nok, si la carte TM1638 n'est pas détectée et donc pas d'affichage possible.
```

L'appui de la touche [TEST] allume toute la face avant (voyants, touches et afficheurs) et active les alarmes sonores pendant 3 secondes. A la fin de ce délai, le KVB réaffiche l'état précédent.

L'appui de la touche [VAL] pendant plus de 3 secondes, permet de modifier la luminosité des afficheurs et des voyants (*Sauf le voyant LSSF*). C'est aussi un moyen de faire un reset du KVB, si l'on ne modifie pas la luminosité.

Dans ce cas, les trois touches [VAL] [MV] [FC] sont éclairées.

[MV] = Diminue la luminosité.

[FC] = Augmente la luminosité.

[VAL] = Validation de la luminosité, puis fait un reset logiciel. Cette valeur est sauvegardée en eeprom.

L'appui sur les touches [MV] ou [FC] allume ou éteint localement la touche. Dépend de l'activation de la fonction "FONCTION_VOY_MV_FC".

Dans ce cas, on envoie sur la ligne série, le code du voyant de la touche = 'h', 'H', 'i' ou 'I'.

Panneau des roues codeuses.

Quand on bouge une des roues codeuses, le voyant [VAL] clignote.

Quand on appuie sur ce voyant [VAL], il s'éteint. Le KVB envoie alors les 6 chiffres correspondant aux roues codeuses et le code du bouton VO-ME-MA (vo=1, me=2, ma=3) sur la liaison série, entre les balises '<>'. Exemple : A<1234561>a ('Aa' étant les codes de la touche [VAL]).

Le programme ne vérifie pas la cohérence des données saisies, et l'appui sur la touche [VAL] est toujours validé.

Dans le code source, on peut modifier les variables :

```
const long duree_antirebond = 300; Durée anti rebond des boutons = 300 msec par défaut.  
const long duree_timer_clignotant = 300; Durée clignotant du LSSF et de [000] = 300 msec par défaut.  
const long duree_timer_test_local = 3000; Durée du test local du KVB = 3 sec par défaut.  
const long duree_timer_luminosite = 3000; Temps d'appui sur la touche [VAL] pour passer en mode  
réglage luminosité = 3 sec par défaut.
```

Dans le code source, on peut désactiver les fonctions suivantes, en plaçant le sigle '//' en début de ligne.

Cela permet de définir si le boîtier KVB est en partie autonome, ou dépend entièrement du simulateur pour fonctionner.

```
#define FONCTION_TEST // Permet de lancer la fonction test avec l'appui de la touche [TEST].  
#define FONCTION_AFF_ROUES // Permet d'afficher le n° des roues codeuses au démarrage, pendant 1 sec.  
#define FONCTION_VAL_ROUES // Permet de faire clignoter le bouton [VAL] si l'on modifie les roues codeuses.  
#define FONCTION_ENV_ROUES // Permet d'envoyer les 7 chiffres des roues codeuses sur la liaison série, une fois  
validé par le bouton [VAL].  
#define FONCTION_OK_MARCHE // Envoie 'o' ou 'n' sur la liaison série, une fois le KVB en ordre de marche.  
#define FONCTION_VOY_MV_FC // Allume les touches [MV] et [FC] en cas d'appui dessus. Dans ce cas, on envoie  
aussi sur la ligne série, le code du voyant de la touche = 'h', 'H', 'i' ou 'I'.  
#define FONCTION_SF_LSSF // Un appui sur le bouton [SF] éteint le voyant LSSF.  
#define FONCTION_VAL_VAL // Un appui sur le bouton [VAL] éteint le bouton [VAL].
```

Remarque :

Si vous modifiez le code, ne pas oublier de mettre tous les 'Serial.print()' en commentaire dans la version finale!

J'ai mis 300 msec pour l'anti rebond des touches. C'est assez long, et si l'on appuie rapidement sur une touche, un seul appui sera pris en compte. A mon avis, cette durée importante est nécessaire, vu les mauvais contacts de cet ancien équipement.

Les questions peuvent être posées sur le forum RMF. Par exemple ici :

<https://www.rmf-magazine.com/phpBB/viewtopic.php?t=203032>

Si vous réaliser des améliorations matériels ou logicielles, merci de les poster sur le forum RMF pour en faire profiter les plus grand nombre.

Si vous publier cette réalisation avec ou sans améliorations, vous devez publier votre code source.

Ce logiciel est un logiciel libre. Exigence du concepteur : Ne pas modifier les lignes d'affichage de l'intro à la mise sous tension. A la mise sous tension, affiche "JLF jj/mm/aaaa" pendant 1 seconde.

Conserver la routine "void intro_jlf()" et son action à l'écran au démarrage de l'Arduino.

On peut modifier ce programme et le diffuser. Dans ce cas, il faut préciser l'origine et donner accès aux sources modifiées.

Définition : Un logiciel libre est un logiciel distribué avec l'intégralité de ses programmes-sources, afin que l'ensemble des utilisateurs qui l'emploient, puissent l'enrichir et le redistribuer à leur tour.

Note : Un logiciel libre n'est pas nécessairement gratuit et les droits de la chaîne des auteurs sont préservés.

Équivalent étranger : free software, open source software.

(Source : Vocabulaire de l'informatique (liste de termes, expressions et définitions adoptés), NOR: CTNX0710138K, J.O n° 93 du 20 avril 2007 page 7078, texte n° 84)

logiciel libre

Par logiciel libre on entend un logiciel qui offre la liberté aux utilisateurs d'exécuter, de copier, de distribuer, d'étudier, de modifier et d'améliorer le logiciel. Plus précisément, elle fait référence à quatre types de liberté pour les utilisateurs du logiciel :

La liberté d'exécuter le programme, pour tous les usages (liberté 0).

La liberté d'étudier comment le programme fonctionne et de l'adapter à ses besoins (liberté 1). L'accès au code source est une condition requise.

La liberté de redistribuer des copies, (liberté 2).

La liberté d'améliorer le programme et de diffuser les améliorations au public pour en faire profiter toute la communauté (liberté 3). L'accès au code source est une condition requise.

A+

Avec une alimentation 12 Volts peut-être séparée, avec une longueur de câble assez longue et un environnement perturbé, il vaut mieux protéger cette liaison. En plus, l'Arduino DUE risque de griller, si une de ses pattes dépasse la tension de 3,3 Volts.

Avec cette installation, il faudra peut-être débrancher l'Arduino NANO des ses broches, pour mettre à jour le programme ! La liaison série via le port COMx par la prise USB, ne fonctionnera plus très bien !

Ce montage à base d'optocoupleur EL817C, n'est pas très performant, et le débit sera limité à 9600 bps.

Si vous voulez un montage plus performant, utiliser des circuits de type : TLP2962.

Il y a déjà une résistance de 1 K en entrée RX sur l'Arduino NANO. Ajouter une résistance de 2,2 K suffit en entrée RX d'un Arduino NANO. Remplacer la 680 Ohms par une 2,2 K.

Schéma pour un branchement sur un Arduino DUE (3,3 Volts) (Résistance 680 à remplacer par une 2,2K sur un Arduino NANO) :

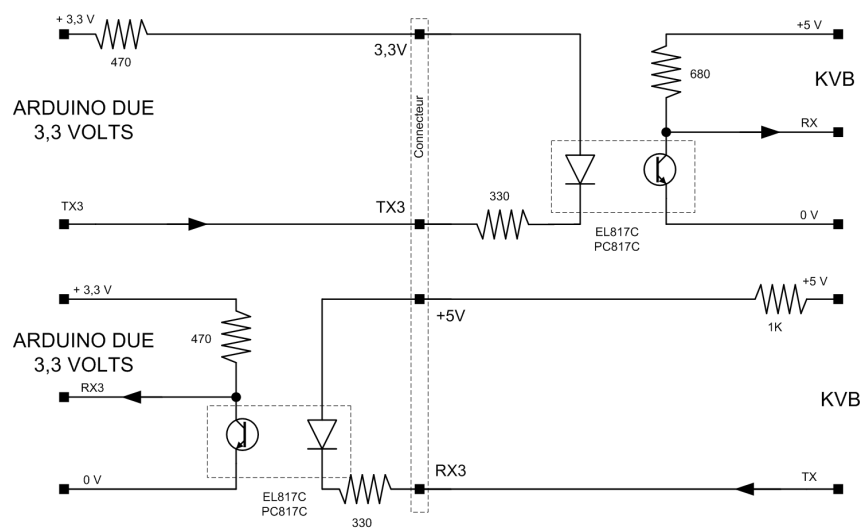
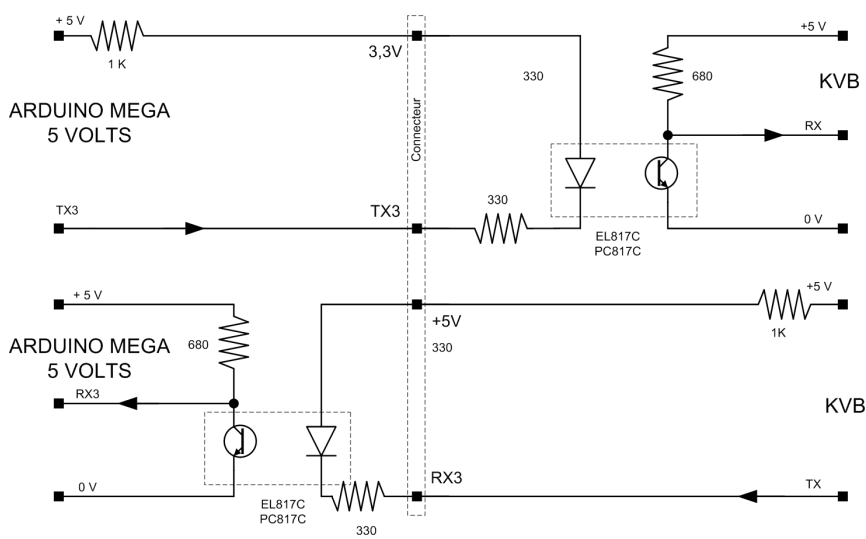
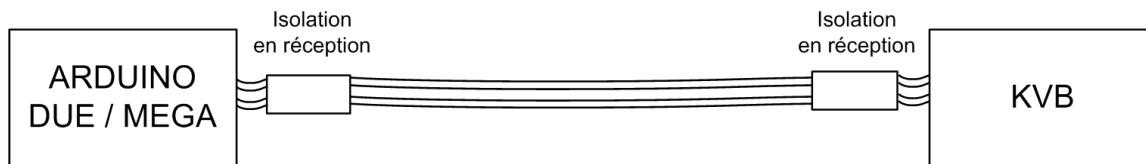


Schéma pour un branchement sur un Arduino MEGA (5 Volts) (Résistance 680 à remplacer par une 2,2K sur un Arduino NANO):

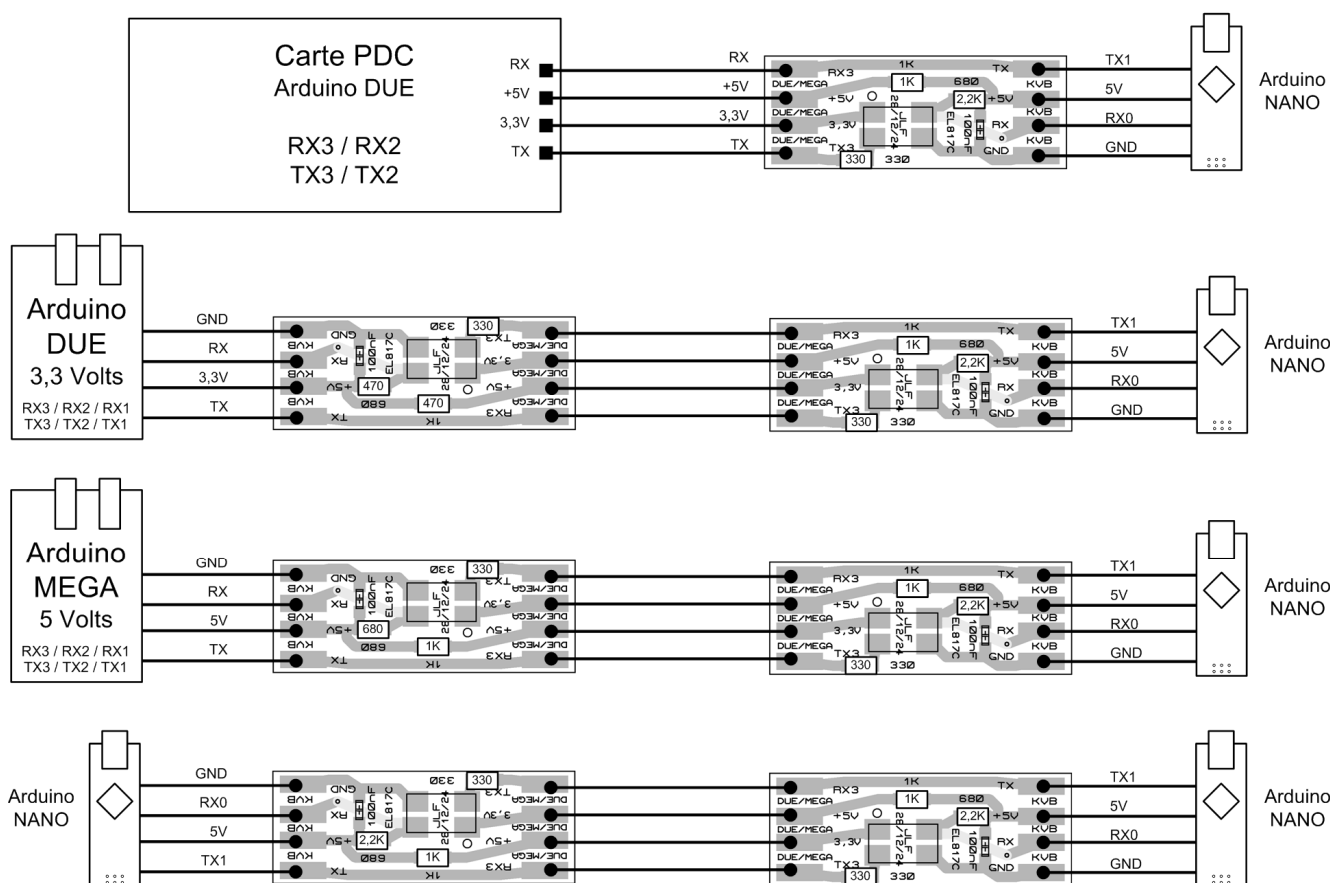


On placera les montages d'isolation de telle sorte que le phototransistor (*Coté RX*) soit relié avec un câble court.



Le fichier pour faire fabriquer ces circuits imprimés sont fournis. Il suffit d'envoyer le fichier 'PDC_Arduino_JLF_OPTO - CADCAM.ZIP', à un site comme 'JLPCB'. <https://jlpcb.com>

JLF 22/02/2025



L'utilisation d'un EL817c ou PC817c, n'est pas optimum, mais fonctionne.

Une valeur correcte des résistances est important pour obtenir un bon fonctionnement, mais suivant la version d'optocoupleur, il faudra peut-être les modifier.

Pour la liaison, un connecteur à 4 plots est prévu coté Arduino Due / Mega.

En cas de problème, il faut vérifier si les signaux en sortie des optocoupleurs vont bien de 0 à 3,3 Volts ou 5 Volts.

Si besoin, ajuster les valeurs des résistances.

Interface de ce KVB avec les locomotive BB7200 de SimExpress sur TRAIN SIMULATOR CLASSIC de DTG.

J'ai réalisé un montage à base d'ARDUINO qui s'interface avec TRAIN SIMULATOR CLASSIC de DTG, ici : http://www.la-tour.info/uts/uts_page15.html#conduite

Avec le programme "TSClassic Raildriver and Joystick Interface" qui envoie l'état du KVB du simulateur à mon montage, je peux animer le vrai KVB avec les lignes de code suivantes :

Dans "buffer_rx[1]", on a le nom de la variable envoyée par le simulateur.

Dans "buffer_rx_valeur", on a la valeur envoyée par le simulateur.

```
// <KVB:XXXXXX:1> ==> Envoi d'un ordre vers le KVB, sur la liaison série n°3.

if(!strcmp(buffer_rx[0],"KVB")) {
    buffer_rx_valeur = atoi(buffer_rx[2]); // Valeur = 0 ou 1.
    if (!strcmp(buffer_rx[1],"KVB_BP_VAL_lumiere_control")) {if(buffer_rx_valeur == 1) KVB_car_vers_KVB = 'G'; else KVB_car_vers_KVB = 'g'; }
    else if(!strcmp(buffer_rx[1],"KVB_BP_CAR_lumiere_control")) {if(buffer_rx_valeur == 1) KVB_car_vers_KVB = 'I'; else KVB_car_vers_KVB = 'i'; }
    else if(!strcmp(buffer_rx[1],"KVB_BP_VIO_lumiere_control")) {if(buffer_rx_valeur == 1) KVB_car_vers_KVB = 'H'; else KVB_car_vers_KVB = 'h'; }
    else if(!strcmp(buffer_rx[1],"Autotest_KVB_control")) {if(buffer_rx_valeur == 0) KVB_car_vers_KVB = 'O'; // [ ][ ] + lampes SOL et ENG éteintes.
    else if(buffer_rx_valeur == 1) KVB_car_vers_KVB = 'n'; // [ PA][400]
    else if(buffer_rx_valeur == 2) KVB_car_vers_KVB = 'N'; // [ UC][512] + lampes SOL et ENG allumées.
    else if(buffer_rx_valeur == 3) KVB_car_vers_KVB = 'o'; } // lampes SOL et ENG allumées.

    else if(!strcmp(buffer_rx[1],"KVB_LS_SF_control")) {if(buffer_rx_valeur == 1) KVB_car_vers_KVB = 'J'; else KVB_car_vers_KVB = 'j'; }
    else if(!strcmp(buffer_rx[1],"KVB_LS_V_control")) {if(buffer_rx_valeur == 1) KVB_car_vers_KVB = 'l'; else KVB_car_vers_KVB = '0'; }
    else if(!strcmp(buffer_rx[1],"KVB_LS_FU_control")) {if(buffer_rx_valeur == 1) KVB_car_vers_KVB = '3'; else KVB_car_vers_KVB = '2'; }
    else if(!strcmp(buffer_rx[1],"bip_V_control")) {if(buffer_rx_valeur == 1) KVB_car_vers_KVB = 'Q'; else KVB_car_vers_KVB = 'q'; }
    else if(!strcmp(buffer_rx[1],"KVB_visu_control")) {if(buffer_rx_valeur == 0) KVB_car_vers_KVB = 'c'; // [ ][ ]
    else if(buffer_rx_valeur == 1) KVB_car_vers_KVB = 'm'; // [---][---]
    else if(buffer_rx_valeur == 3) KVB_car_vers_KVB = 'T'; // [888][888] Séquence Test.
    else if(buffer_rx_valeur == 8) KVB_car_vers_KVB = 't'; // [888][888]
    else if(buffer_rx_valeur == 9) KVB_car_vers_KVB = 'F'; } // [ ][ F ]

}
}
```

Et je peux envoyer le code des touches, correspondant à l'appui des boutons [VAL] et [SF] du KVB du pupitre physique, vers le simulateur.

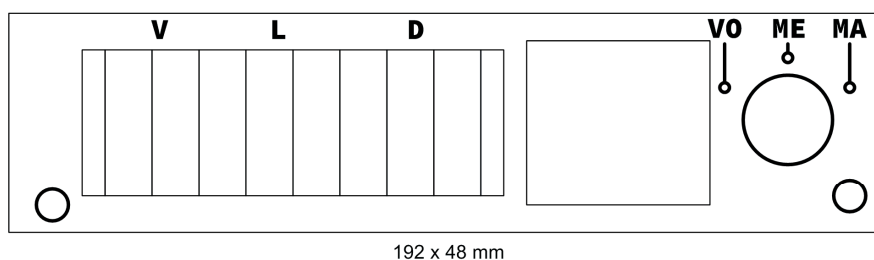
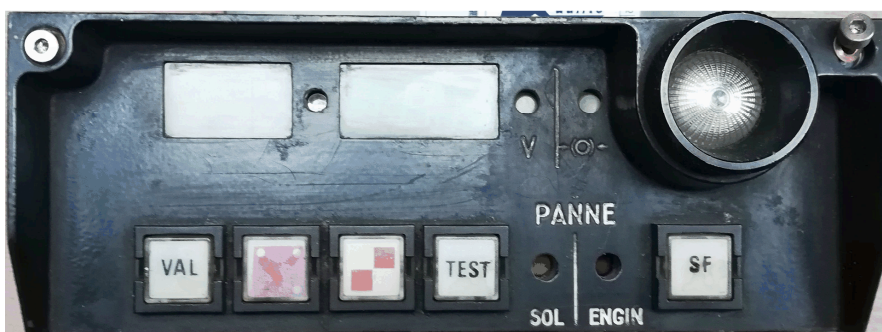
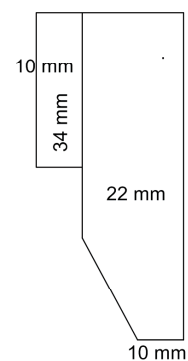
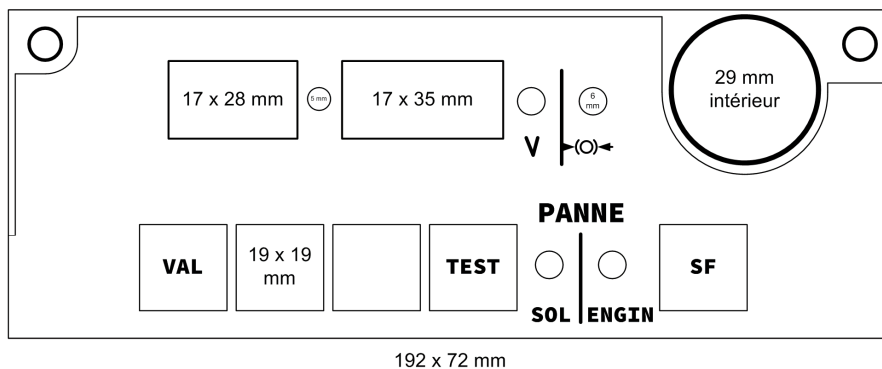
```
/* KVB : On envoie, si l'on a reçu un appui de touche du KVB, une commande au PC.
Code des touches reçues sur la liaison série : On envoie un caractère par touche.
[VAL] => 'A' enfoncée, 'a' relâchée
[MV] => 'B' enfoncée, 'b' relâchée
[FC] => 'C' enfoncée, 'c' relâchée
[TEST] => 'D' enfoncée, 'd' relâchée
[SF] => 'E' enfoncée, 'e' relâchée */

//
if (Serial3.available() > 0) {
    char car_lu = Serial3.read();
    if (car_lu == 'A') { tch_env(KEY_LEFT_SHIFT+KEY_LEFT_CTRL, 'v', duree_touche_200); } // Appui sur le bouton [VAL] du KVB. 200 msec obligatoire
    if (car_lu == 'E') { tch_env(NORMAL, KEY_KP_ENTER, duree_touche_200); } // Appui sur le bouton [SF] du KVB. 200 msec obligatoire
/*
    else if (car_lu == '<') { roues_codeusescpt = 1; roues_codeuses[0] = '<'; }
    else if ((car_lu >= '0') && (car_lu <= '9') && (roues_codeusescpt >= 1)) {
        roues_codeuses[roues_codeusescpt] = car_lu;
        roues_codeusescpt++;
        if (roues_codeusescpt >= 9) roues_codeusescpt = 0;
    }
    else if ((car_lu == '>') && (roues_codeusescpt == 8)) {
        roues_codeusescpt = 0; // Le tableau roues_codeuses[] = <1234561> =
        roues_codeuses[8] = '>'; // [VD VU] [VU] [DC DD DU] Centaines, Dizaines, Unités + Bouton [VO-ME-MA] 1:VO, 2:ME, 3:MA
        roues_codeuses[9] = '\0'; // peut être envoyé au PC.
        Serial.write(roues_codeuses,9);
    } */
}
}
```

Construction d'un KVB

Il est aussi possible de construire une réplique du KVB pour votre animer votre pupitre.

Les plans sont fournis avec la documentation :



Il faut trois afficheurs jaunes :

D'origine HP5082-7620 = Hauteur digit 7,6 mm 0,3 inch - Jaune Anode (+) commune - H 19 mm x L 10 mm
ou plus récent : OPD-S3010LA-BW chez TME

Il faut trois afficheurs verts :

D'origine HP HDSP-4600 = Hauteur digit 10,9 mm 0,3 inch - Vert Anode (+) commune - H 19 mm x L 12,5 mm
ou plus récent : HDSP-F501 chez TME

Les nouveaux afficheurs n'ont pas les mêmes dimensions. Il faudra ajouter un cache noir autour des afficheurs.

On peut tester des afficheurs jaune en 0,28 pouces (7,2 mm) et vert en 0,36 pouces (9,1 mm), par groupe de trois en anode commune :

<https://fr.aliexpress.com/item/1005006944959612.html> ou <https://fr.aliexpress.com/item/1005006944967883.html>

On peut trouver des boutons poussoirs carrés : LAS2F-11/24/W chez TME.

En espérant que l'on puisse changer la led ou la résistance pour passer son éclairage en 5 Volts.

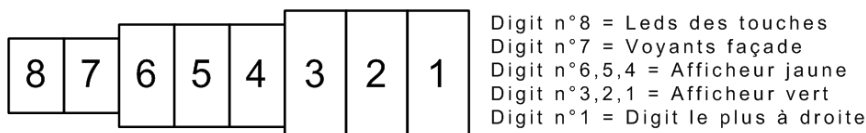
Ou chez AliExpress : <https://fr.aliexpress.com/w/wholesale-bouton-poussoir-carr%C3%A9.html>

Les liens sont donnés sans aucune garanti. A vous de vérifier.

En prenant des afficheurs à anode commune comme ceux d'origine, on peut reprendre le même schéma de câblage que pour l'original.

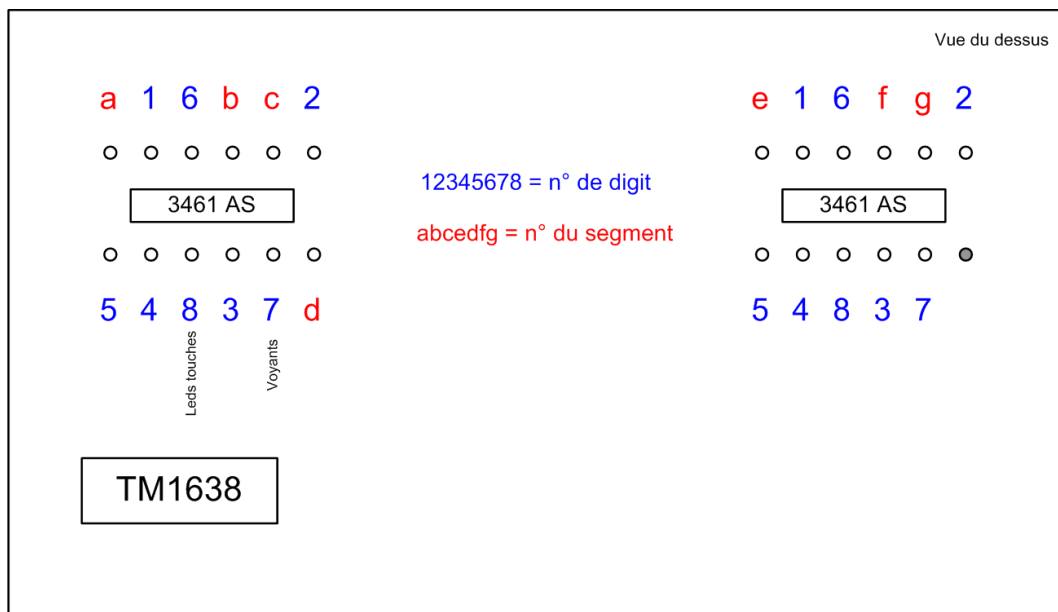
Pour le branchement des 6 afficheurs à anode commune :

Vue face avant de la façade



Ci-dessous : Broche n° 1 = Fil (+) de l'afficheur n° 1...

Broche 'a' = Fil (-) pour les 6 afficheurs (segment a) et (-) d'un voyant et (-) d'une led de bouton poussoir...



Pour le branchement des 4 voyants en face avant :

Fil n° 7 = (+) commun des 4 leds
Fil 'a' = (-) led (V)
Fil 'b' = (-) led (SOL)
Fil 'c' = (-) led (ENG)
Fil 'd' = (-) led (FU)

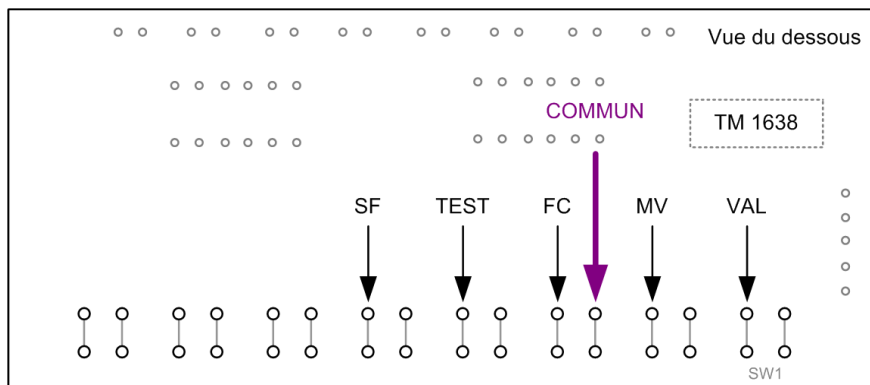
Pour le branchement des 3 leds des boutons poussoir éclairés :

Fil n° 8 = (+) commun des 3 leds
Fil 'a' = (-) led (VAL)
Fil 'b' = (-) led (MV)
Fil 'c' = (-) led (FC)

Pour le branchement des boutons poussoir :

Un fil commun à tous les contacts des boutons poussoir.

Ensuite, les autres fils sont soudés sous le circuit imprimé.



A+