

MIGRATION DU MODULE RST D'UN PUPITRE DE LOCOMOTIVE Le - 17/11/2024

Ce document explique comment animer le bloc RST, récupéré dans une ancienne cabine de conduite d'une locomotive de la SNCF. RST = Radio Sol Train.

Se sera pour animer un vrai pupitre de locomotive, utilisé avec un programme de simulation sur ordinateur.

Ce bloc est traité indépendamment du simulateur, il n'est pas relié à l'ordinateur et est 100 % autonome.

Ce document a été validé en modifiant un second bloc RST avec succès.



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 des améliorations, vous devez publier votre code source.

Ce logiciel est un logiciel libre. Exigence du concepteur : Ne pas modifier les lignes d'affichage à la mise sous tension. A la mise sous tension, affiche "JLF 10/11/2024" sur écran fluo pendant 5 secondes.

```
strncpy(case_g, "JLF", 3);
print_g();
strncpy(case_d, "10/11/2024", 10);
print_d();
```

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.

1 / CONNEXION A L'AFFICHEUR ALPHANUMERIQUE DE 16 SEGMENTS FLUORESCENT.

L'afficheur est piloté par le circuit intégré 10937p de Rockwell.

Plutôt que de refaire toute la partie électronique de l'afficheur, on utilisera ce circuit, ce qui permet d'utiliser la partie haute tension -24 Volts, et la tension alternative générées sur la carte d'origine.

On envoie les données à afficher avec un signal de données "DATA", et un signal d'horloge "SCLK".

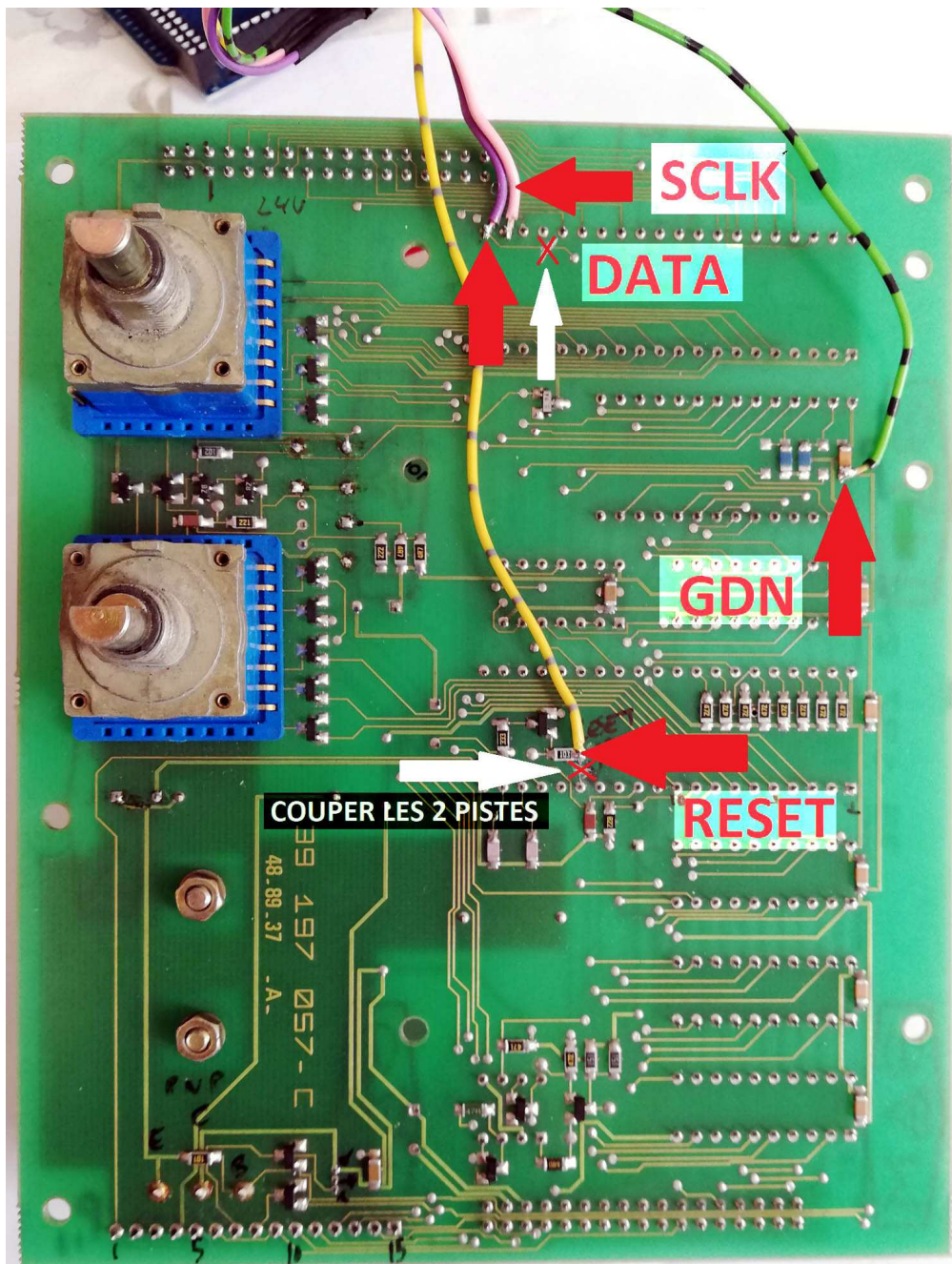
Sur le 10937p, "DATA" = patte n° 21, "SCLK" = patte n° 22. Par contre la patte "RESET" ne doit pas être utilisée directement, mais par l'intermédiaire d'un transistor. Ce transistor est déjà présent sur la carte RST, et on l'utilisera.

On va relier 4 fils (DATA, SCLK et RESET) et la masse entre un ARDUINO et la carte RST.

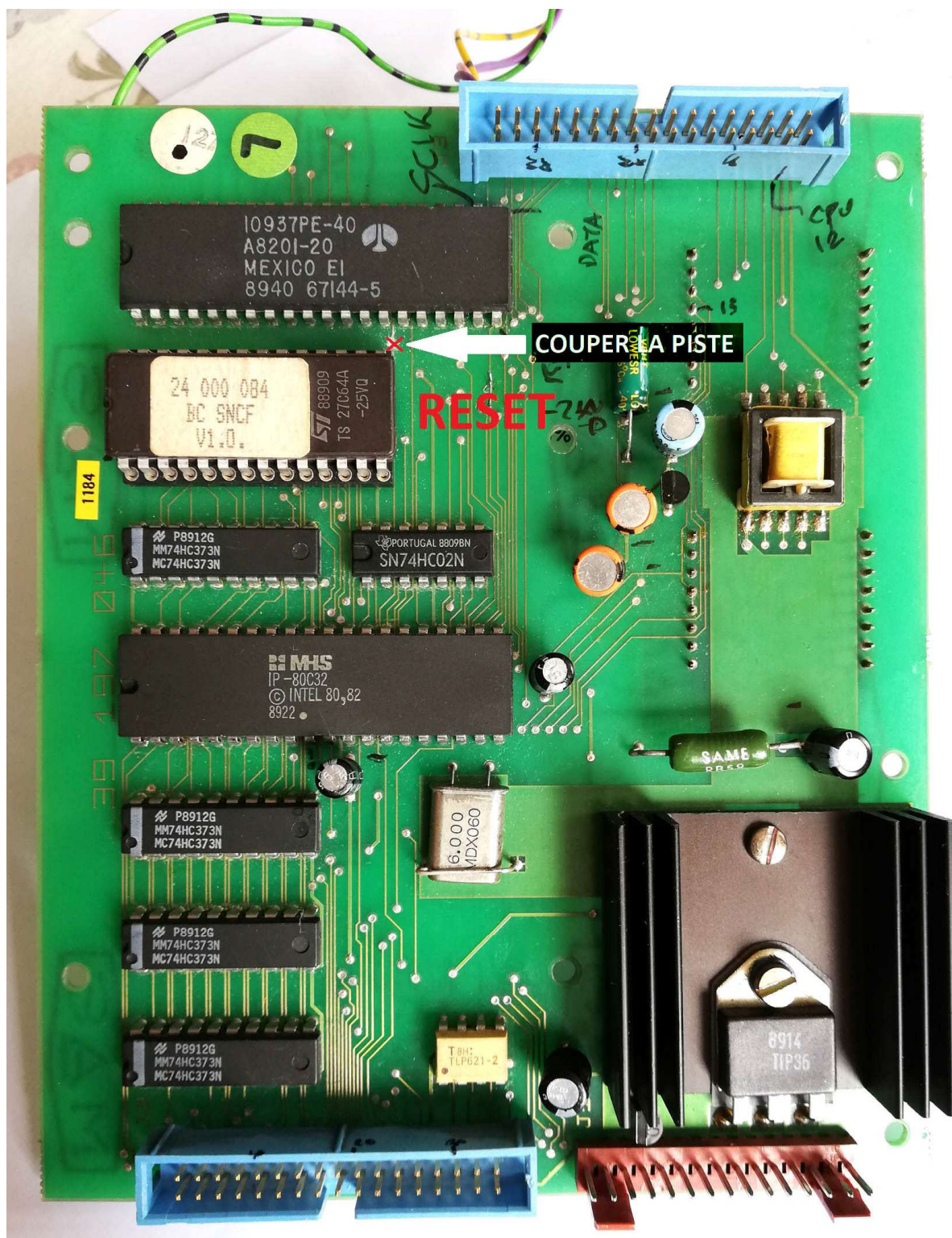
Attention, comme sur la photo, le fil de masse (ici le fil vert/noir) ne doit pas être relié à une patte du 10937p. Il n'y a pas de patte relié au 0 Volt sur le circuit 10937p !!!

Le fil vert de masse, est donc relié au 0 Volt de l'eprom.

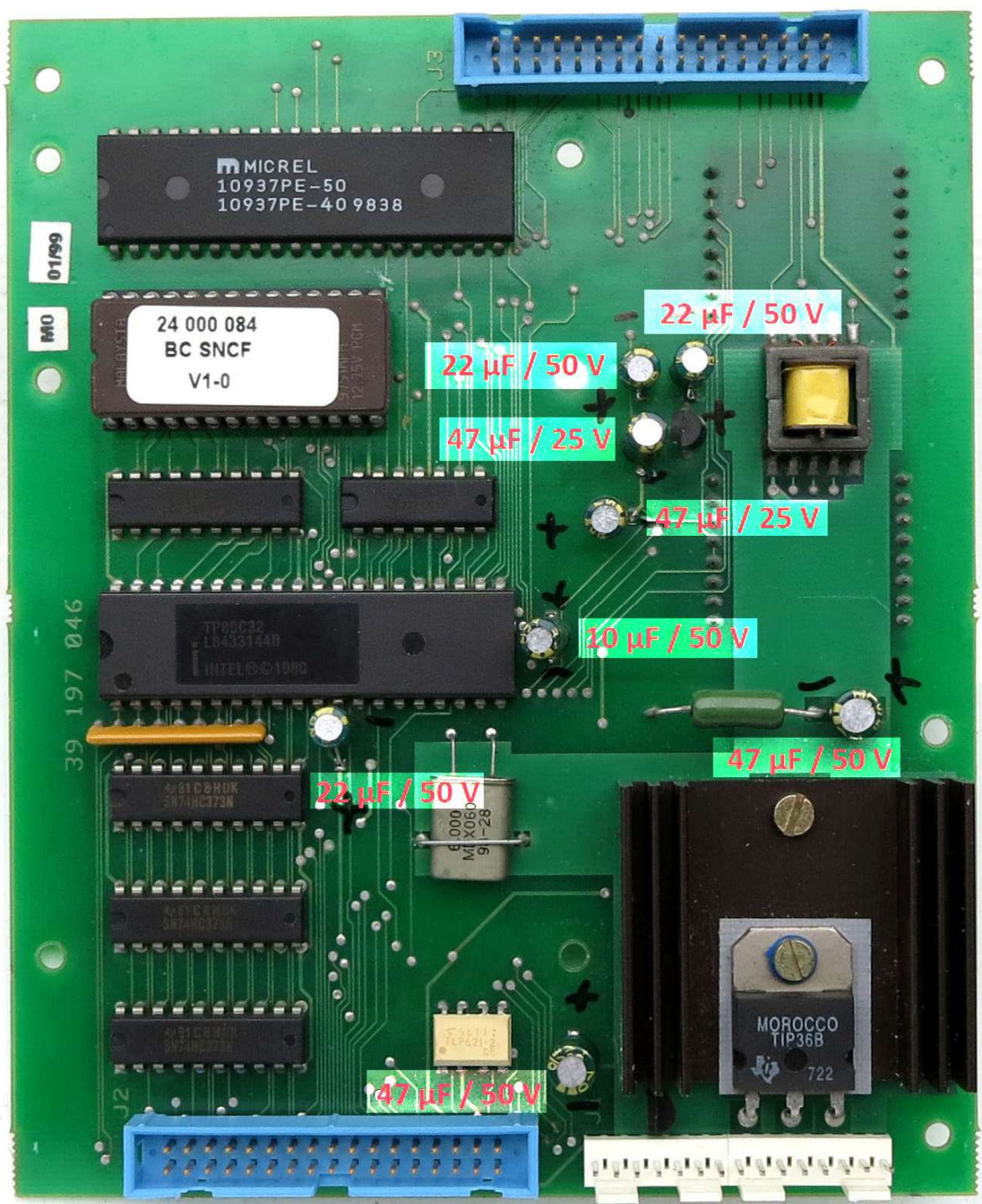
Couper aussi deux pistes de ce côté. Le fil DATA est rose, et le fil SCLK est violet.



De ce coté, il y a une piste à couper.



Remplacer tous les condensateurs chimiques.



R

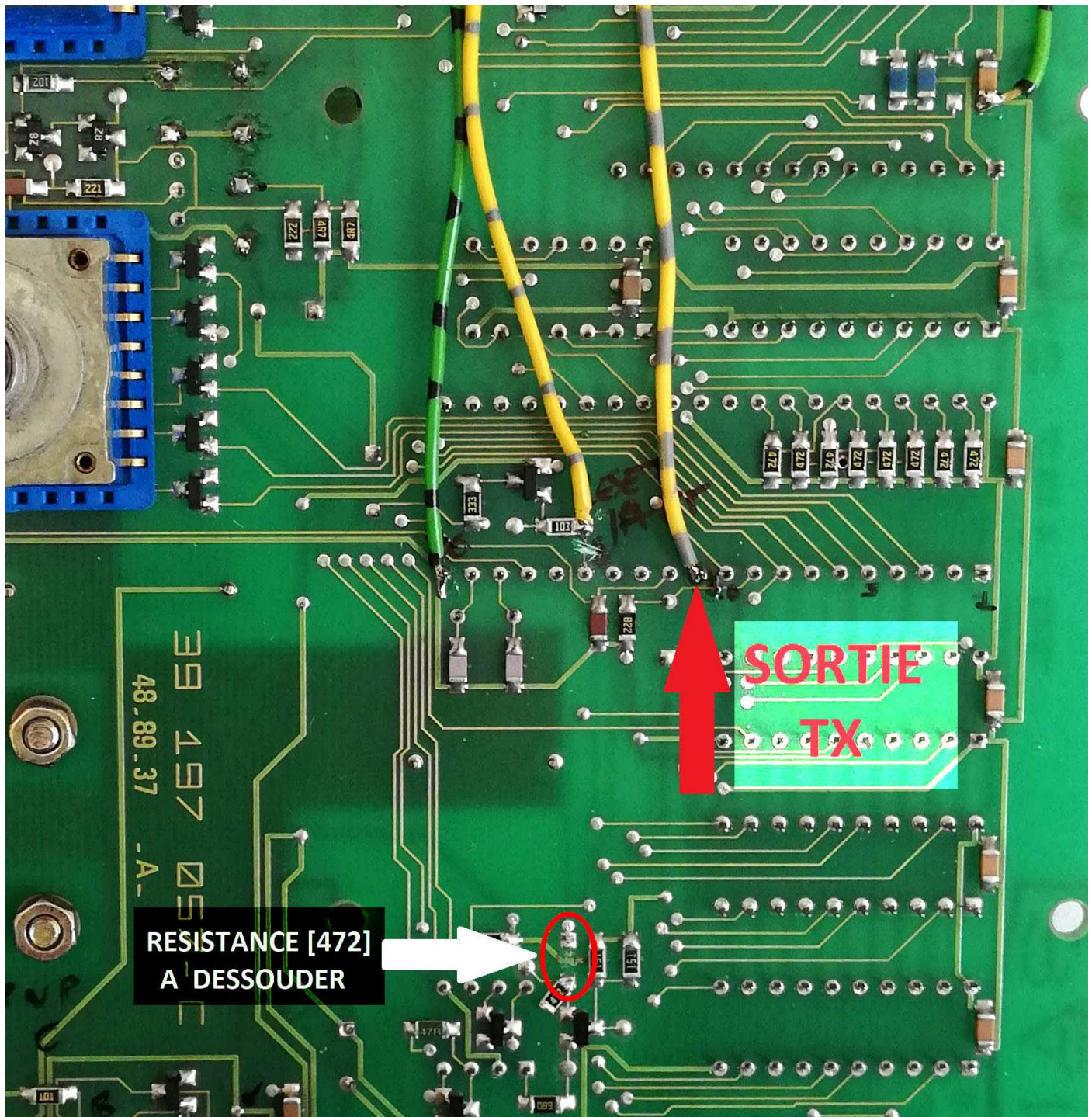
2 / CONNEXION A LA SORTIE TX DU CPU 80C32

On va se brancher directement sur la sortie TX du cpu 80C32 en patte n°11.

On soude un fil pour aller vers l'entrée RX1 de l'ARDUINO.

On soude un fil de masse (Ici en vert/noir) sur la patte n°20.

On dessoudera la résistance de 4,7K marquée [472].



Ici, j'ai juste déplacé la résistance [472], pour pouvoir la remettre en place si besoin.

3 / L'ALIMENTATION EXTERNE

On alimentera la platine RST depuis le connecteur arrière.

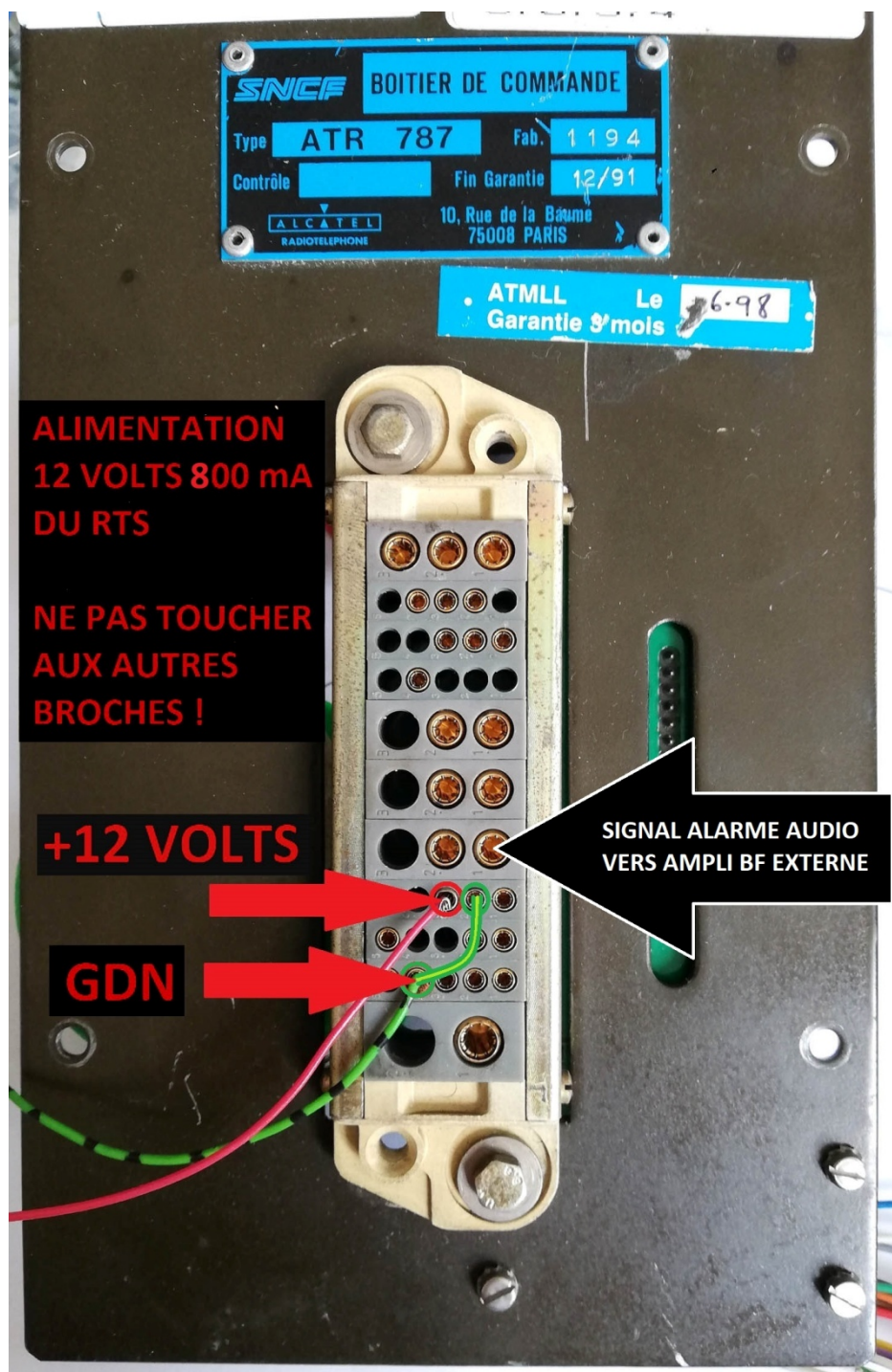
Attention, sur ce connecteur, sont branchés directement des pattes du cpu 80C32 !

Si le +12 Volts touche une autre broche que celle prévue sur la photo, le 80C32 risque de griller.

Il sera peut être à remplacer (*Si une série de touche ne fonctionne plus par exemple*) !

Par précaution, il faudra cacher cette prise par du ruban adhésif.

Il faut fournir une tension de 12 Volts, sous 800 mA maximum quand toutes les lampes du clavier sont allumées.
Sinon, sans allumer le clavier, c'est environ 200 mA.



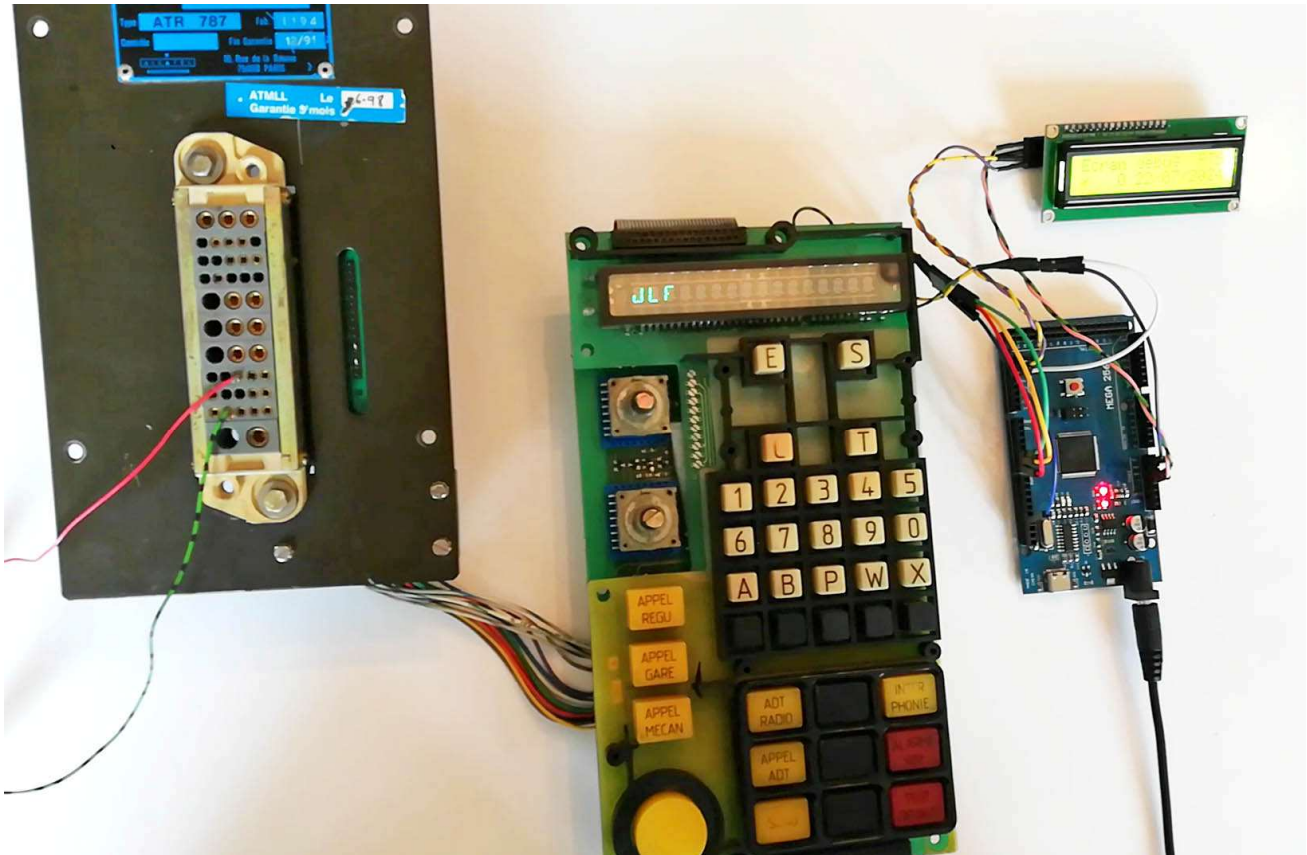
L'alimentation 12Volts, avec en série une diode 1N5404 insérée dans le fil rouge du +12 Volts.
En soudant les fils, cela ne se défait plus.



4 / LA PARTIE LOGICIEL

J'ai utilisé provisoirement, une carte Arduino MEGA 2560 que j'avais sous la main. Au final, une carte NANO sera installée directement dans le boîtier.

Le montage :



L'écran LCD 1602 sert uniquement, lors de la phase de mise au point. Cet écran ne sera pas installé lors de l'intégration de l'Arduino dans le boîtier.

Sur cette carte provisoire ARDUINO MEGA :

L'entrée RX1(19) sera reliée à la sortie TX du 80C32.

Les sorties (*Déclarées dans le code*) :

SDA(20), SLC(21) (et GND, +5V) reliées l'écran LCD I2c 1602 et la carte d'extension des sorties.

(5) reliée à la patte DATA du 10937p

(7) reliée à la patte SLCK du 10937p

(6) reliée à la résistance [103] à la place de la patte n° 15 du cpu, pour activer le RESET du 10937p

L'afficheur fluorescent 16 segments est câblé de façon particulière sur le RST.

Les segments 11, 12 et 13 ne sont pas utilisés, car ils sont sous le cache.

Sur l'affichage, les 10 segments de droite, sont bien câblés sur les sorties 1 à 10 du 10937p.

Par contre, les 3 segments de gauche, sont branchés sur les sorties 11, 12 et 13 du 10937p.

Les sorties 14, 15 et 16 du 10937p ne sont pas utilisées.

Il faudra donc dans le programme de l'Arduino, configurer le 10937p avec 13 segments !

Le code ARDUINO se compose de :

- RST_Arduino_JLF.ino
- Samsung_16LF01_VFD.cpp (De DnaX Daniele Napolitano : https://github.com/DnaX/Samsung_16LF01_VFD)
- Samsung_16LF01_VFD.h (Timing du code ajusté pour le 10937p)
- MCP23017.cpp (De Rob Tillaart : https://github.com/RobTillaart/MCP23017_RT)
- MCP23017.h
- MCP23x17_registers.h
- LiquidCrystal_I2C.cpp (<https://www.arduino.cc/reference/en/libraries/liquidcrystal-i2c>)
- LiquidCrystal_I2C.h

La librairie `https " Samsung_16LF01_VFD "` vient du site `://github.com/DnaX/Samsung_16LF01_VFD`

5 / FONCTIONNALITES

Attention !!! Ne pas laisser ce boîtier avec toutes les lampes allumées. Ce n'est pas un fonctionnement normal, et le boîtier va finir par chauffer.

A la mise sous tension, il faut attendre l'extinction du message de bienvenue (10 sec), avant d'appuyer sur une touche. Cela permet au RST d'envoyer ses trames série à la mise sous tension. Pendant ces 10 secondes, l'Arduino ignore ces trames.

La led bleue, à gauche de l'afficheur, s'allume quand le RST est en cours d'envoi d'une trame série. Il faut attendre que la led bleue s'éteigne pour appuyer sur une nouvelle touche. Le RST bufférisé jusqu'à 5 appuis de touches. La led bleue restera alors allumée pendant plus longtemps.

- Allumage du RST, bouton sur "M1" : Affiche [00][-----].
- Appui sur la touche [C] : On saisie le numéro du canal qui vient se mettre sur l'afficheur CANAL et le garde en mémoire --> La validation se fait avec la touche [E](Enter) : 00 -> [C] -> : -> :2 -> :23 -> [E] -> 23
- Appui sur la touche [T] : On saisie le numéro du train maxi 6 chiffres, on valide en appuyant sur [E]. : ----- -> [T] -> : -> :2 -> :23 -> :234567 -> [E] -> [ENTREE] -> [234567]
- Il n'y a pas d'ordre de saisie. On peut très bien saisir en premier le numéro de train ou le canal et inversement.
- Appui sur la touche "TEST DEFAULT" : Si l'on n'est pas dans une séquence de saisie de n° de train ou de canal, activation d'une alarme sonore, éclairage de toutes les touches, et affichage de [TEST] sur les segments de droite, pendant 3,5 secondes. A la fin, arrête l'alarme sonore et l'éclairage de touches.
- Appui sur le bouton rond rouge "ALERTE" : Activation d'une alarme sonore, et éclairage de la touche. L'arrêt de la sonnerie et de l'éclairage se fait en ré-appuyant dessus.
- Lors de l'appui sur les boutons [ADT RADIO]/[APPEL ADT]/[SONO]/[INTER PHONIE]/[ALARME VOY]/[APPEL REGU]/[APPEL GARE]/[APPEL MECAN] : Ils doivent simplement s'éclairer et on les éteint en ré-appuyant dessus.
- Si l'on met le bouton sur "A", le RST s'éteint.

6 / L'ECLAIRAGE DES TOUCHES

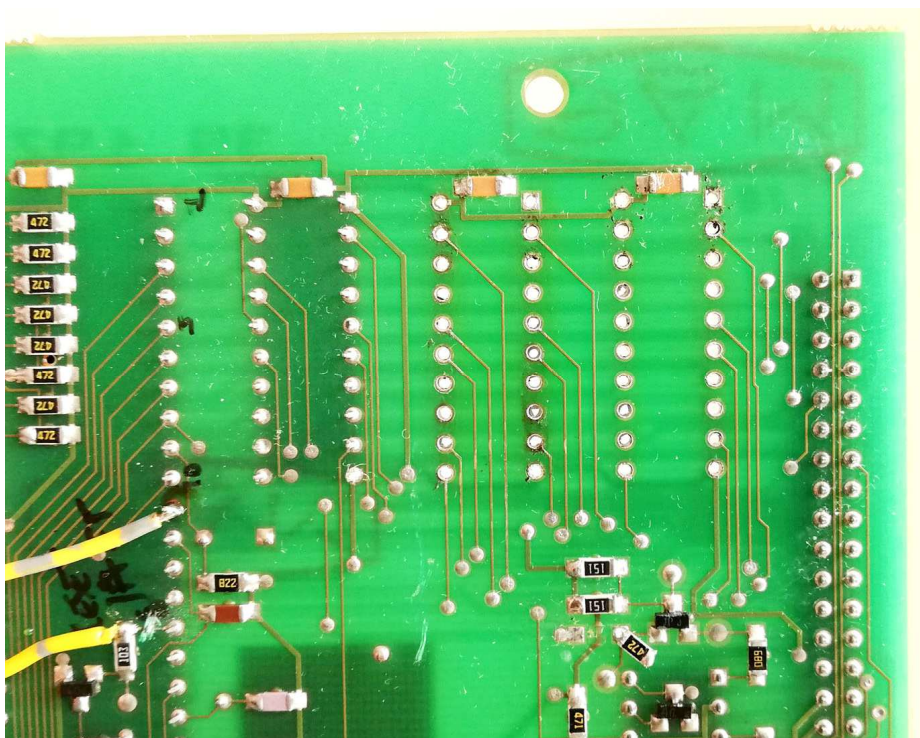
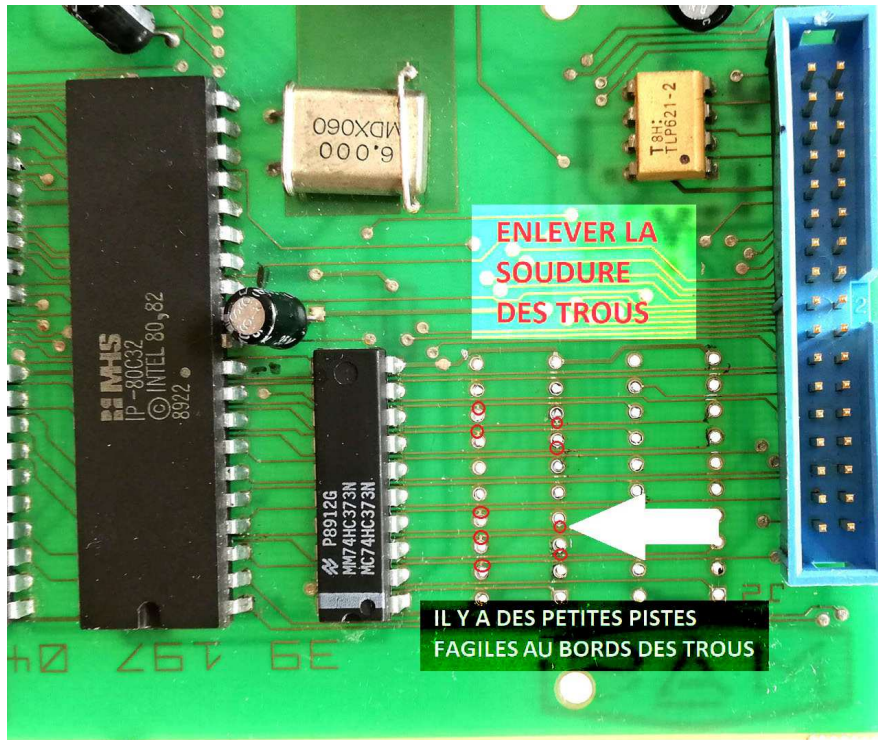
Dessouder avec précaution les deux 74HC373 situés à droite. Les pistes sont très fines, et il ne faut pas les arracher. On peut retirer ces circuits, ils ne servent qu'à commander l'allumage des lampes du clavier. La platine continuera à fonctionner normalement.

Pour faciliter la chose, couper les pattes des circuits à la pince coupante fine, puis retirer les pattes avec le fer à souder.

Ne pas toucher au 74HC373 de gauche !

Bien nettoyer les trous (Pompe ou tresse à dessouder).

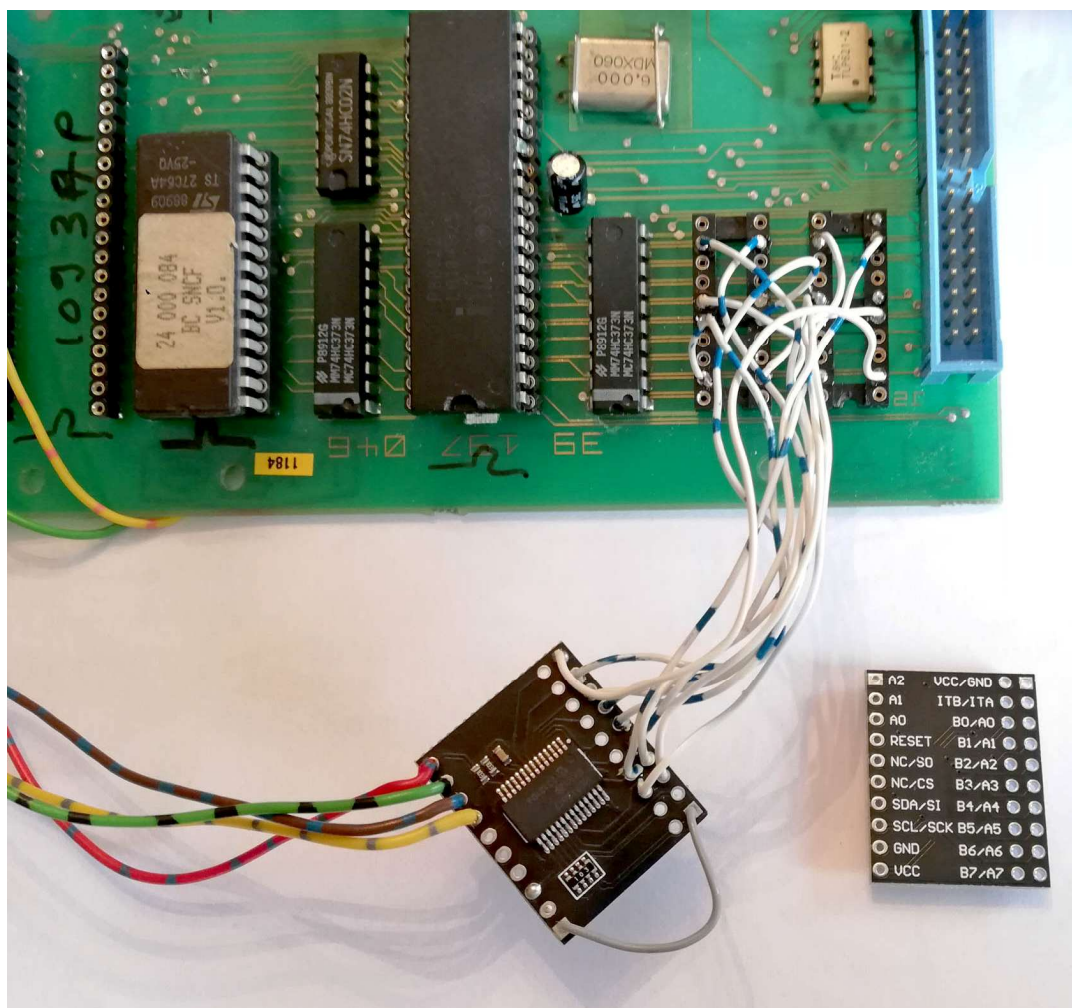
Les minuscules pistes au bord des trous, relient bien les pistes qui passent à côté. Ce n'est pas un reste de soudure.



On peut souder des supports de CI à ces emplacements. Cela évitera d'arracher une piste par la suite. Ce n'est pas obligatoire, et je ne l'ai pas fait pour la seconde radio RST traitée.

Souder des fils de liaison sur les supports de CI, suivant le plan de câblage, page suivante.

Pour commander les lampes, il faudra mettre ces fils au +5 Volts. La consommation est de 2 mA par fil.



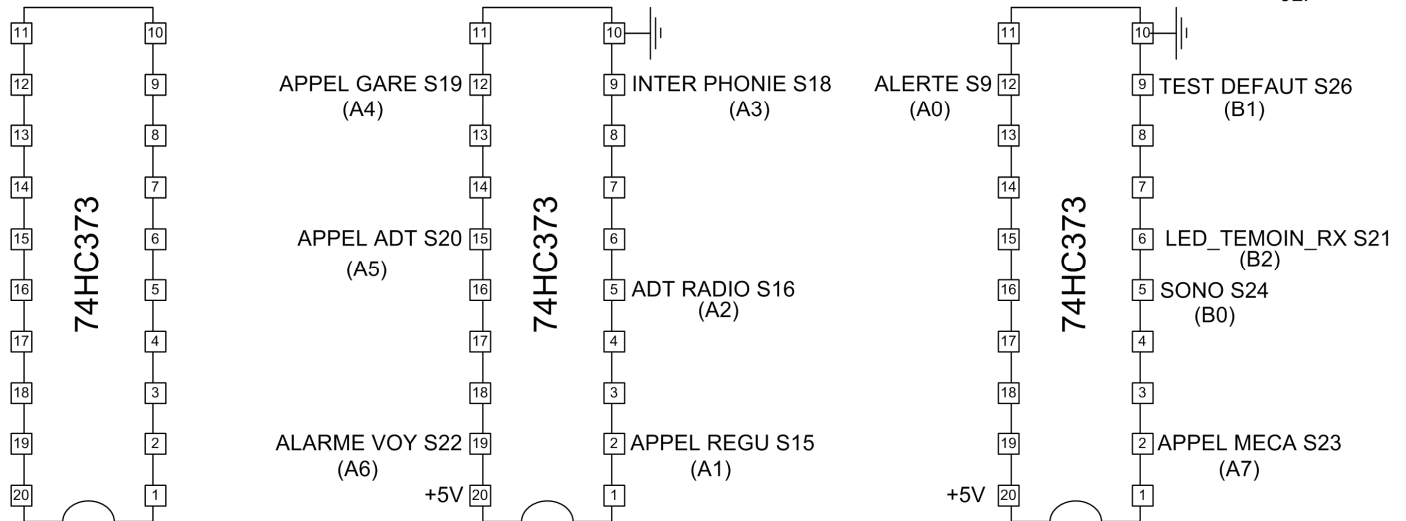
Pour commander les 10 lampes plus la led RX, il faut un module avec un circuit MCP23017, pouvant piloter 16 sorties, avec un courant d'au moins 5 mA pour une sortie à l'état "haut".

Ce module est piloté par une liaison série I2C.

On soude 11 fils entre ce module et les deux supports de CI.

Le 29/08/2024

JLF



CI utilisé par le cpu

Ces broches sont connectées à une résistance de 2,2K qui commandes des transistors, pour alimenter les lampes.

+5 Volts = lampe allumée.

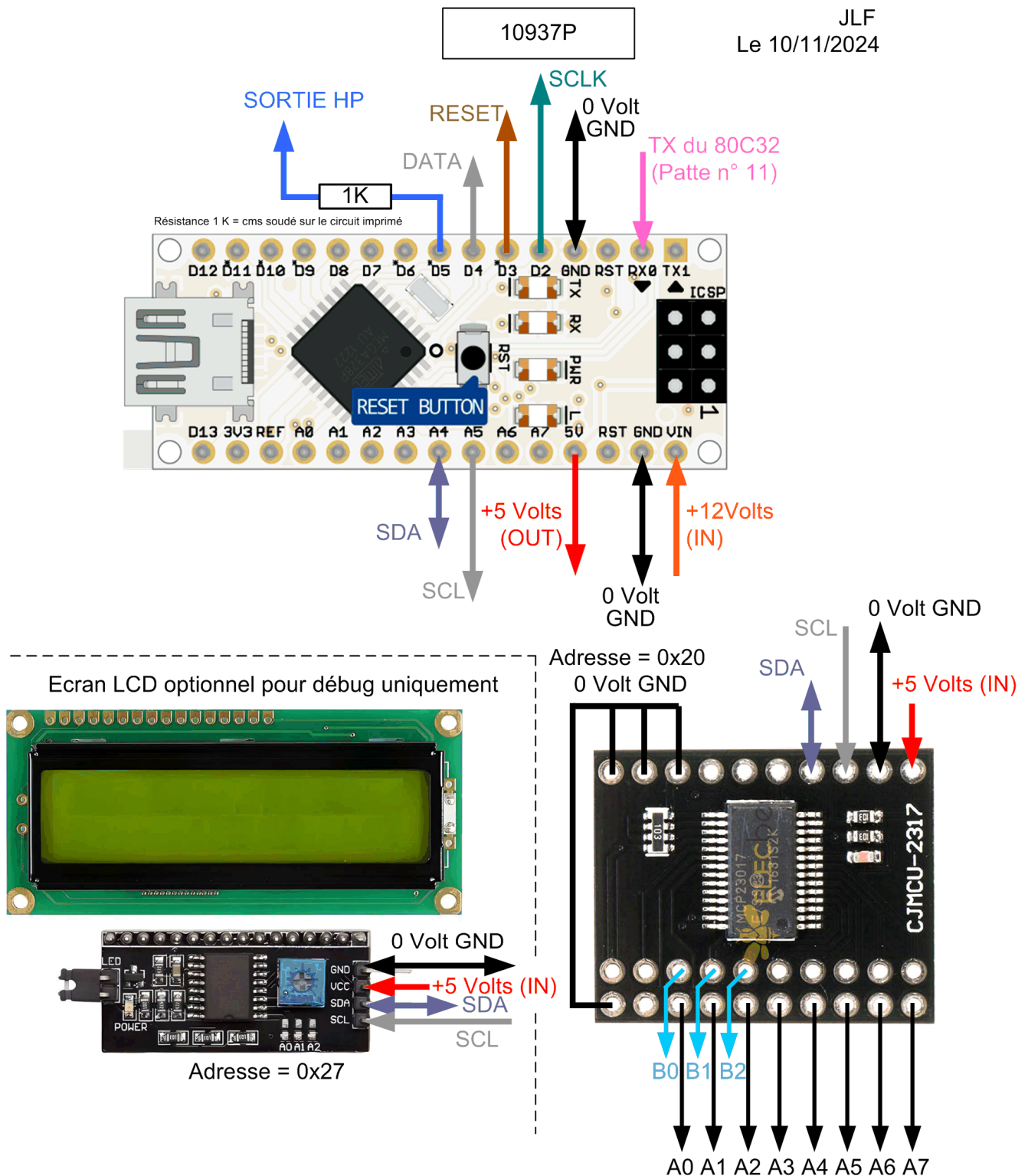
S26 = Numéro de broche du connecteur 34 broches.
(A5) = Broche de sortie du MCP23017

On soude les fils suivants :

VOYANTS	N° de sortie du module MCP23017
VOY_ALERTE	A0
VOY_APPEL_REGU	A1
VOY_ADT_RADIO	A2
VOY_INTER_PHONIE	A3
VOY_APPEL_GARE	A4
VOY_APPEL_ADT	A5
VOY_ALARME_VOY	A6
VOY_APPEL_MECA	A7
VOY_SONO	B0
VOY_TEST_DEFAULT	B1
VOY_TEMOIN_RX	B2

On soude les quatre fils venant de l'Arduino : SCL, SDA, GDN et VCC.

JLF
Le 10/11/2024



L'écran LCD ne sert que pour la phase de mise au point, lors de modification du code. En utilisation normale, pas besoin de l'installer ni de le câbler.

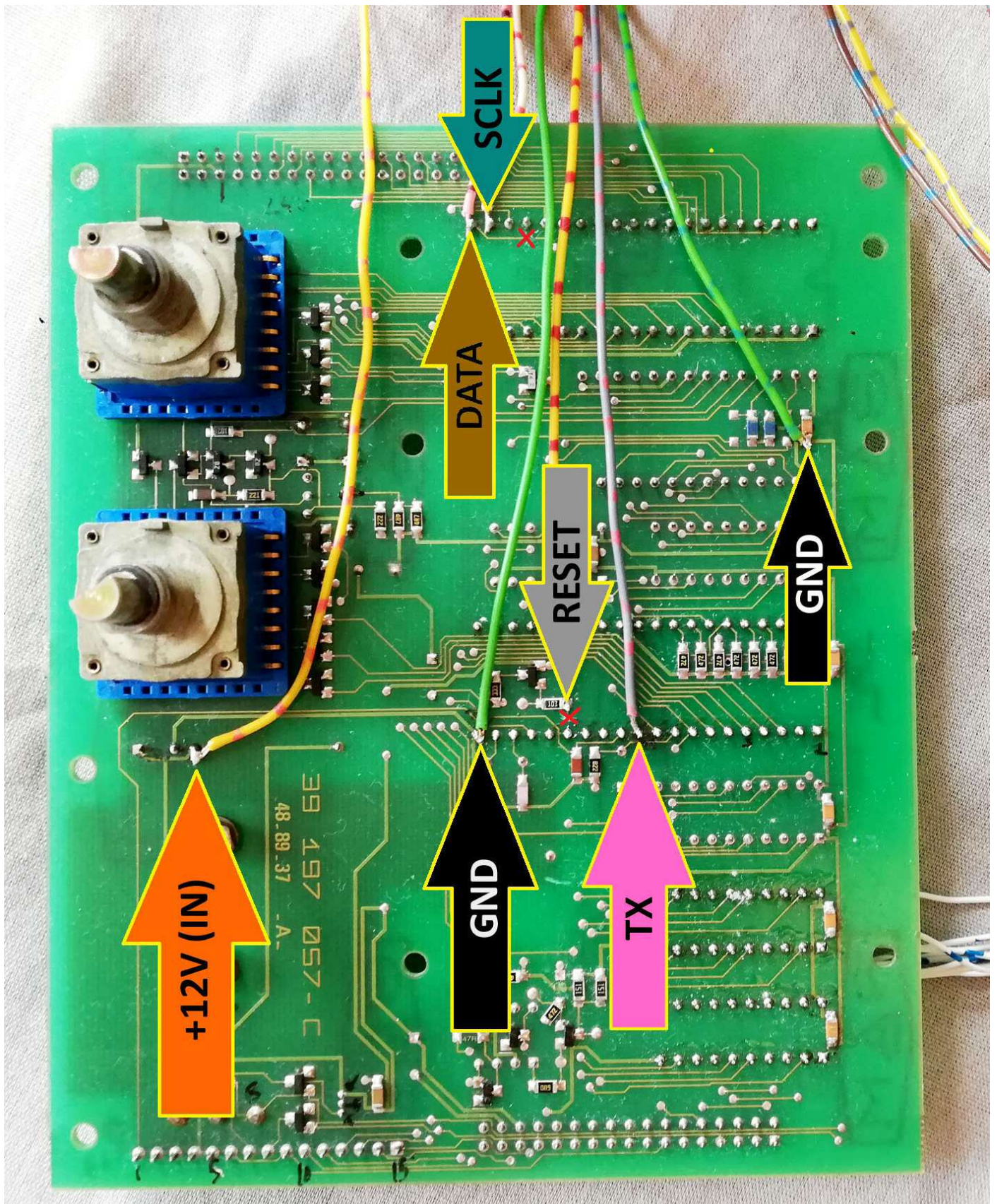
Sur le module MCP23017, on modifie l'adresse par défaut, pour configurer 0x20 comme adresse du module. Il faut mettre les broches A0, A1 et A2 à la masse (GND).

Relier les broches GND entre elles. Les deux fils soudés sur le circuit imprimé, sont branchés sur l'Arduino.

Relier les broches +5 Volts entre elles.

Relier les broches SDA entre elles.

Relier les broches SCL entre elles.



Il y a deux coupures de piste de ce coté, marqué [X] et une coupure de piste de l'autre coté.

8 / AJOUT DE LA LED BLEUE EN FACE AVANT.

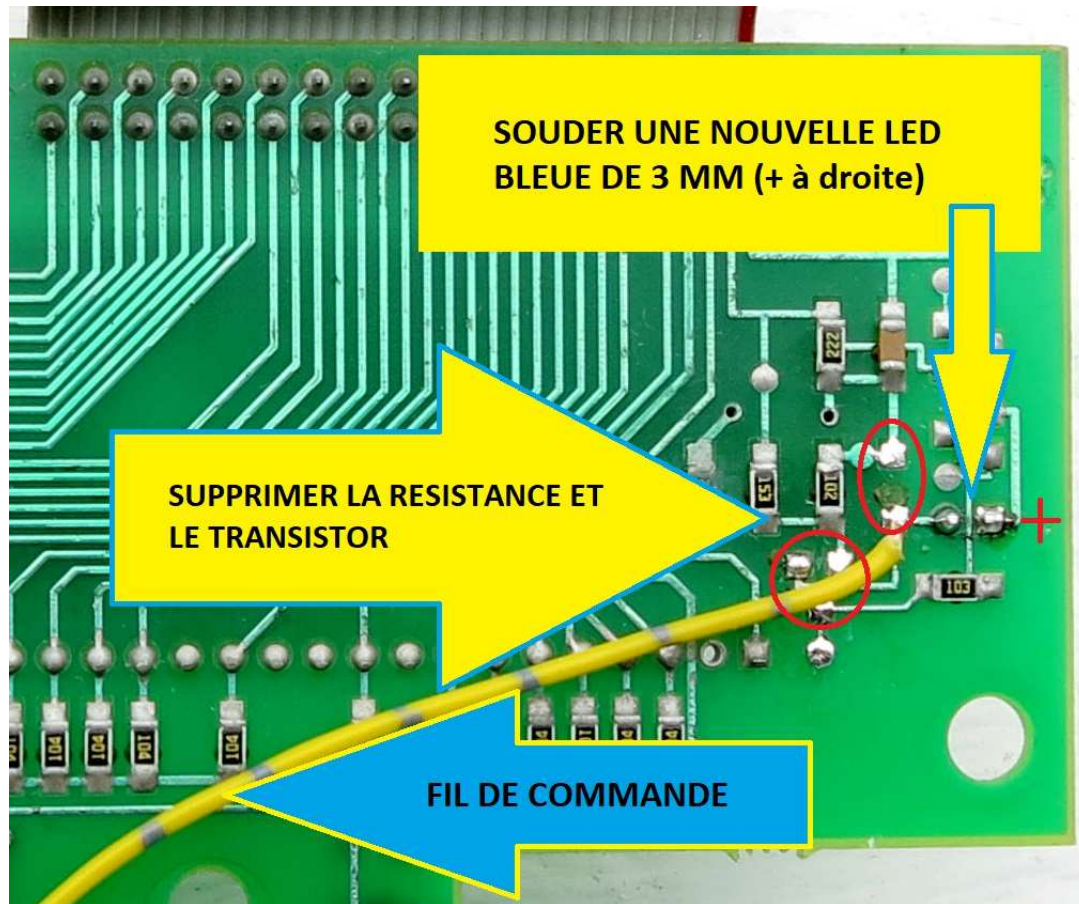
C'est un ajout optionnel.

La led bleue s'allume quand le RST est en cours d'envoi d'une trame série à 2400 bps. Il faut attendre que la led bleue s'éteigne pour appuyer sur une nouvelle touche. En fait, le système bufférise 3 à 4 touches d'avance.

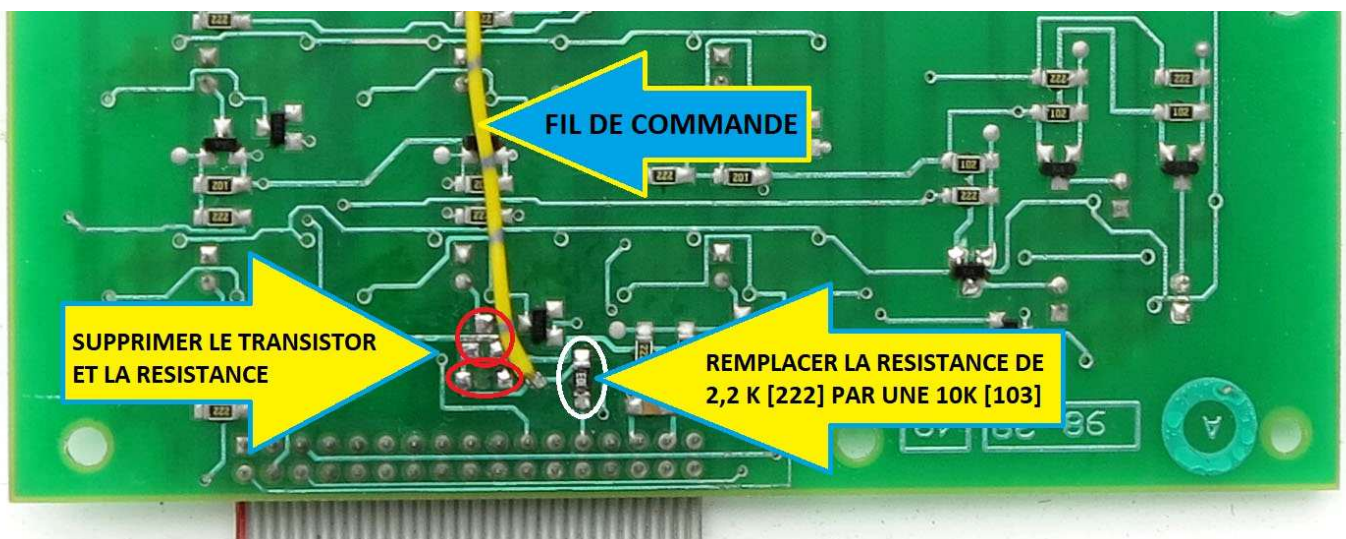
La led est reliée à la sortie B2 du MCP23017.

On soude en série une résistance de 10 K [103] pour limiter le courant à 1 mA.

Le fil de commande de cette led bleue arrive sur le 74HC373 sur la broche "LED_TEMOIN_RX". Elle sera reliée à la sortie B2 du module MCP23017. On peut déplacer des composants ou plus simplement les retirer.



Déplacer la résistance marquée 2,2K [222] ou 1 K [102].



9 / AJOUT DE LA SORTIE AUDIO POUR L'ALARME SONORE.

Une alarme sonore est produite pendant 3,5 secondes, lors de l'appui sur la touche "TEST DEFAULT", en même temps que l'éclairage des touches du clavier.

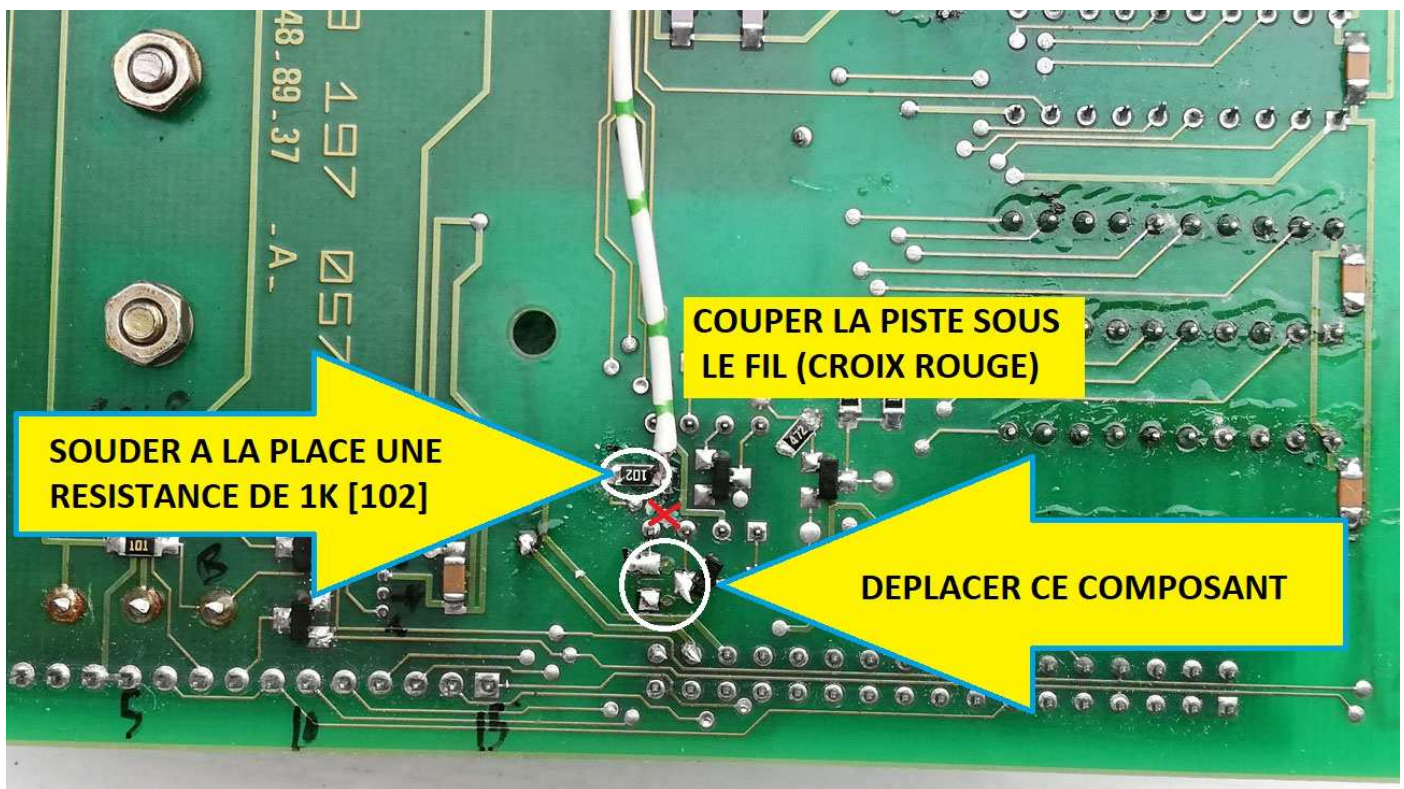
Une alarme sonore est aussi produite, quand l'on appuie sur la touche "ALERTE". Il faut ré-appuyer sur cette touche pour arrêter l'alarme.

La sortie du signal BF se fait sur un plot du connecteur de la face arrière. Voir la photo du connecteur.

On ne peut pas directement brancher un haut-parleur en sortie. Il faut relier ce fil à un ampli BF externe.

L'amplitude du signal audio est de 5 Volts. On protège la sortie de l'Arduino, en limitant le courant à 5 mA, avec une résistance de 1K soudée sur le circuit imprimé.

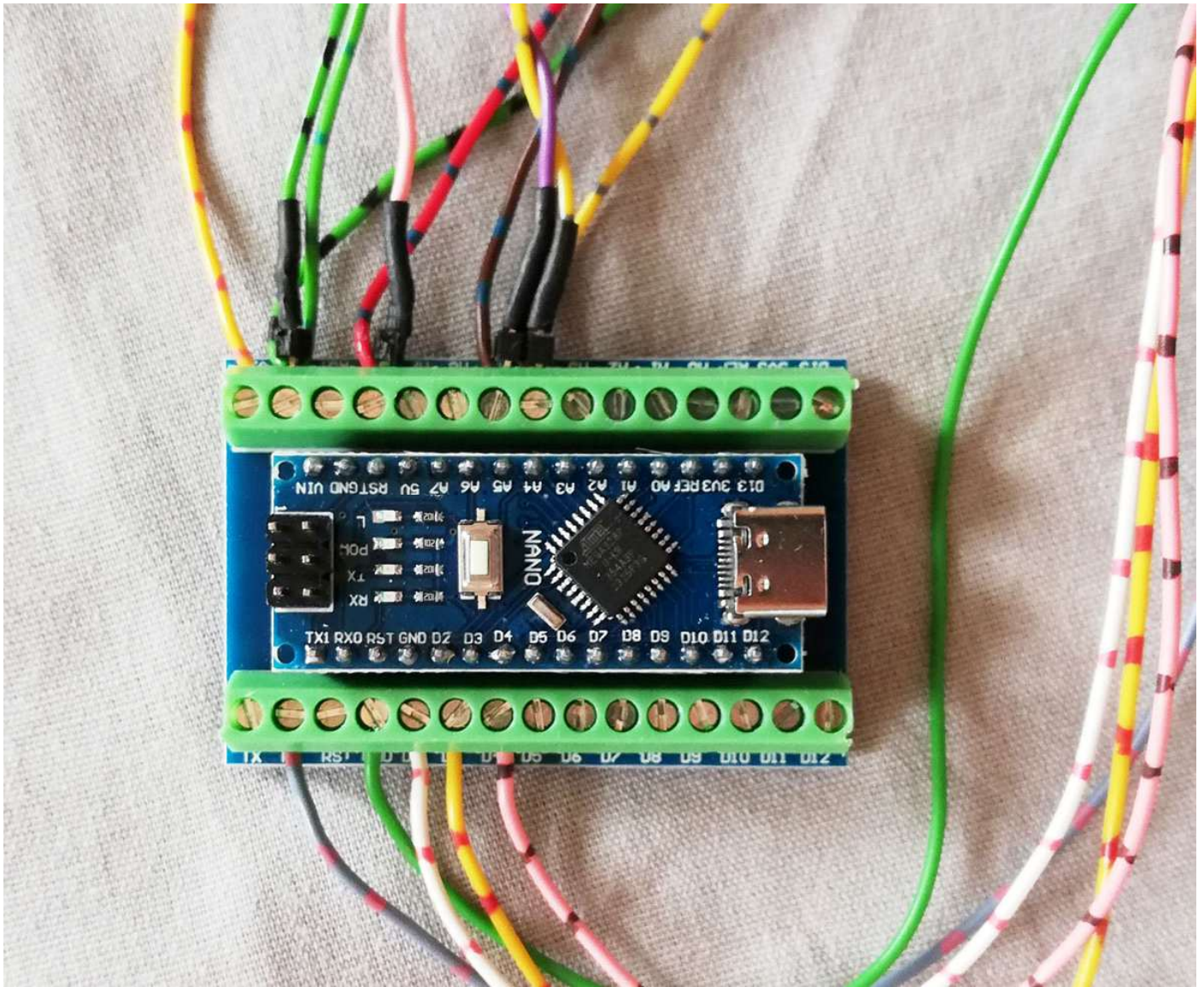
La sortie audio de l'Arduino est la broche "D5" (*Fil blanc/vert*).



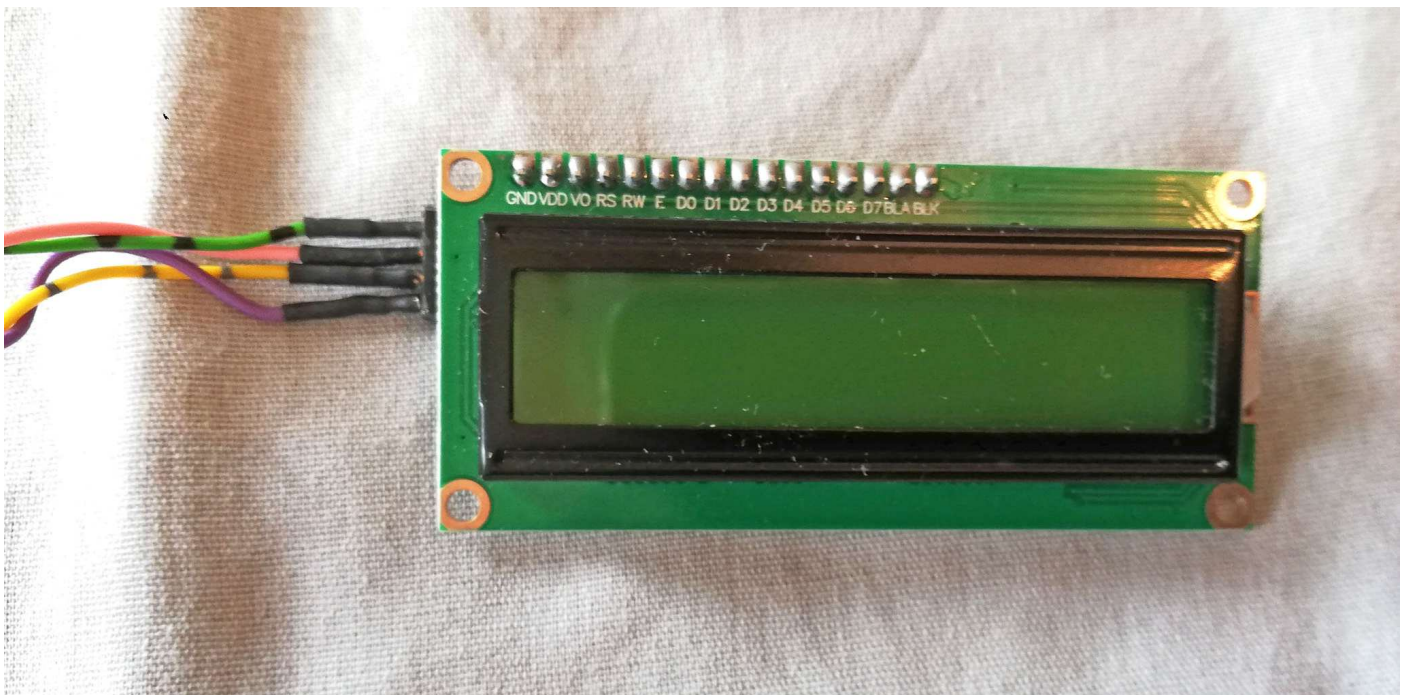
Il faut supprimer ou déplacer le transistor cms.

Il faut remplacer la résistance cms. Si l'on n'a pas de résistance cms, il faudra alors ajouter une résistance en série de 1 KOhm sur ce fil.

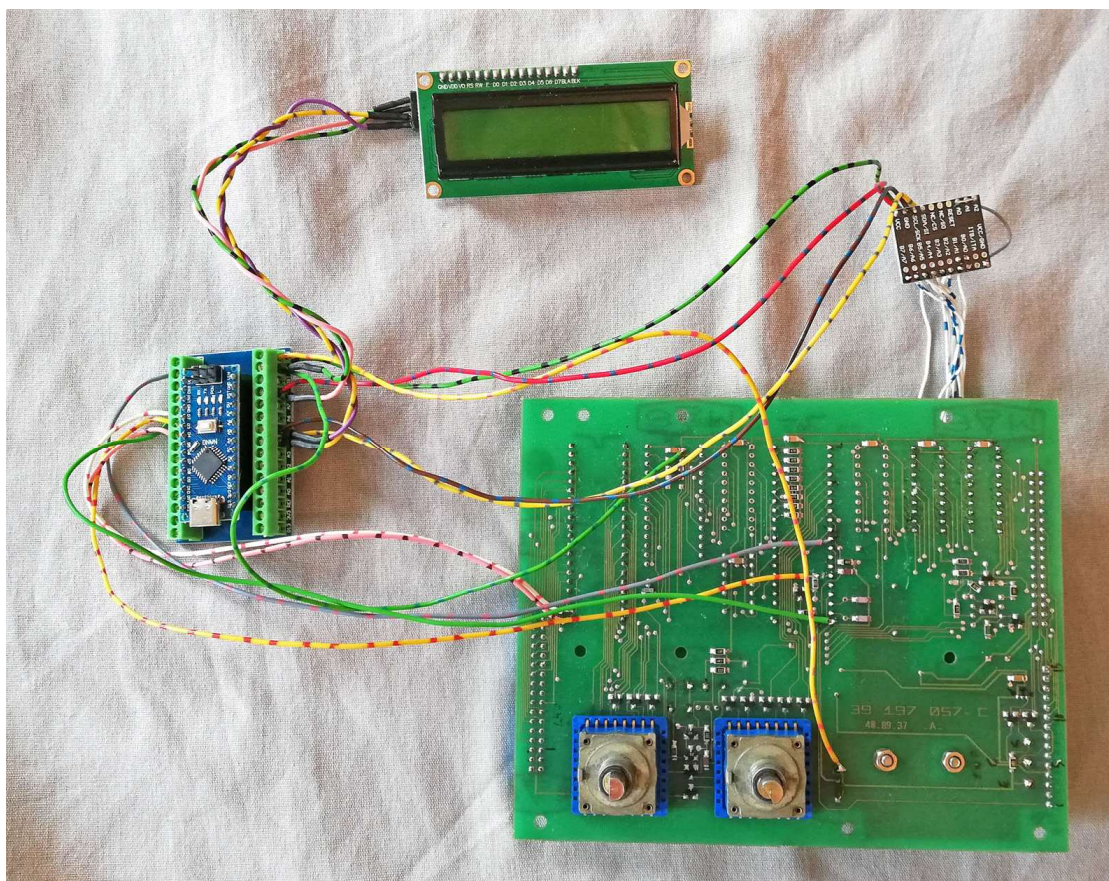
Le branchement de la carte Arduino Nano (Sans le fil de la sortie HP en D5)



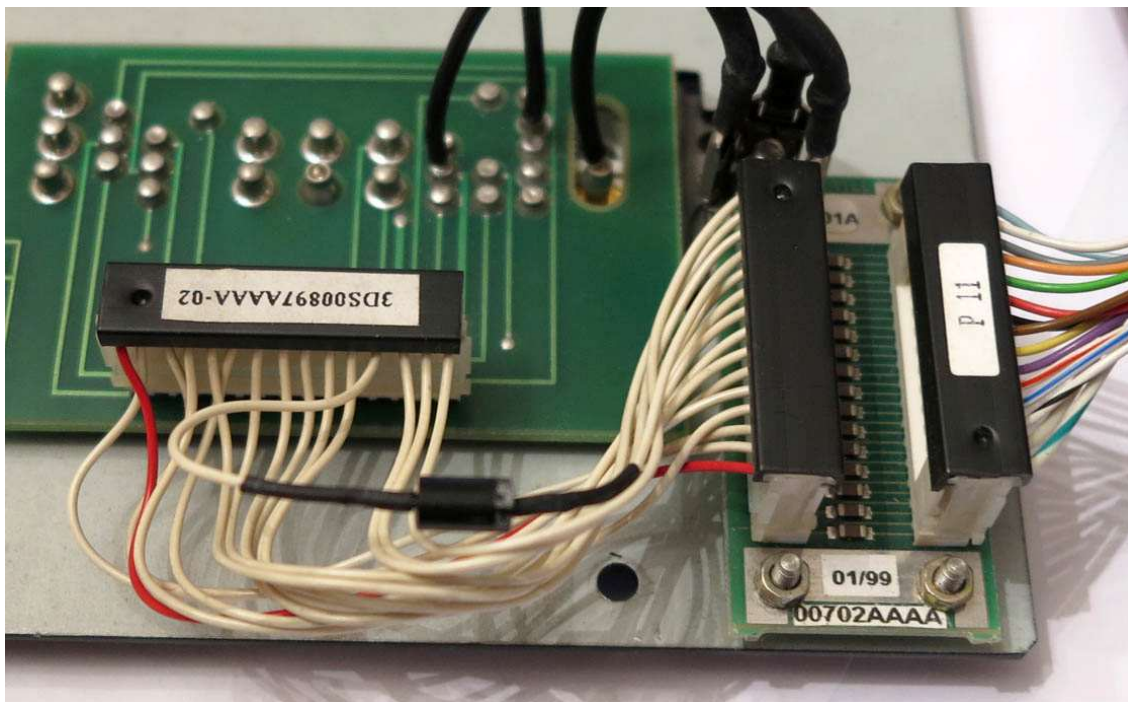
Le branchement de l'écran LCD I2C, 2x16 caractères, pour la phase de mise au point du programme seulement.



Le montage général (Sans les fils de commande de la led bleue et la sortie HP).



On peut placer la diode de protection 1N5404 dans le boîtier, sur le fil n° 4 en partant de la droite. Ça protège de l'inversion des fils d'alimentation. C'est plus pratique que de souder cette diode sur le gros connecteur doré.



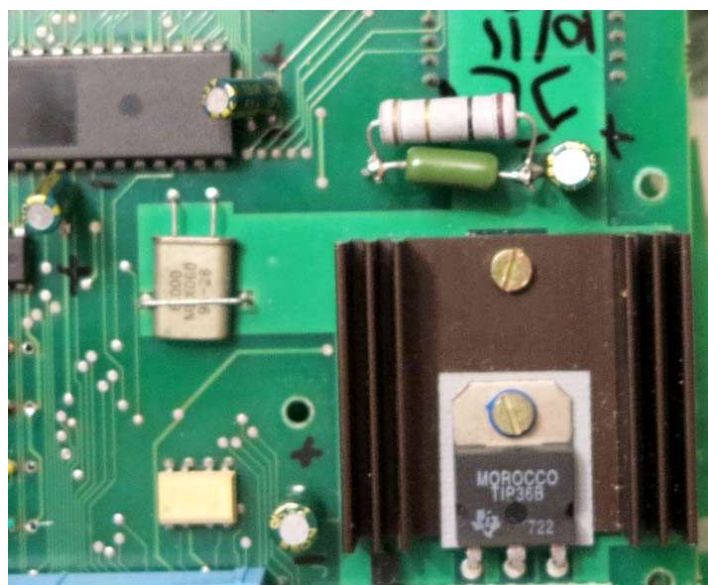
10 / DEPANNAGE

Le montage alimenté en 12 Volts, on doit avoir 5 Volts sur la broche n° 20 des 74HC373.

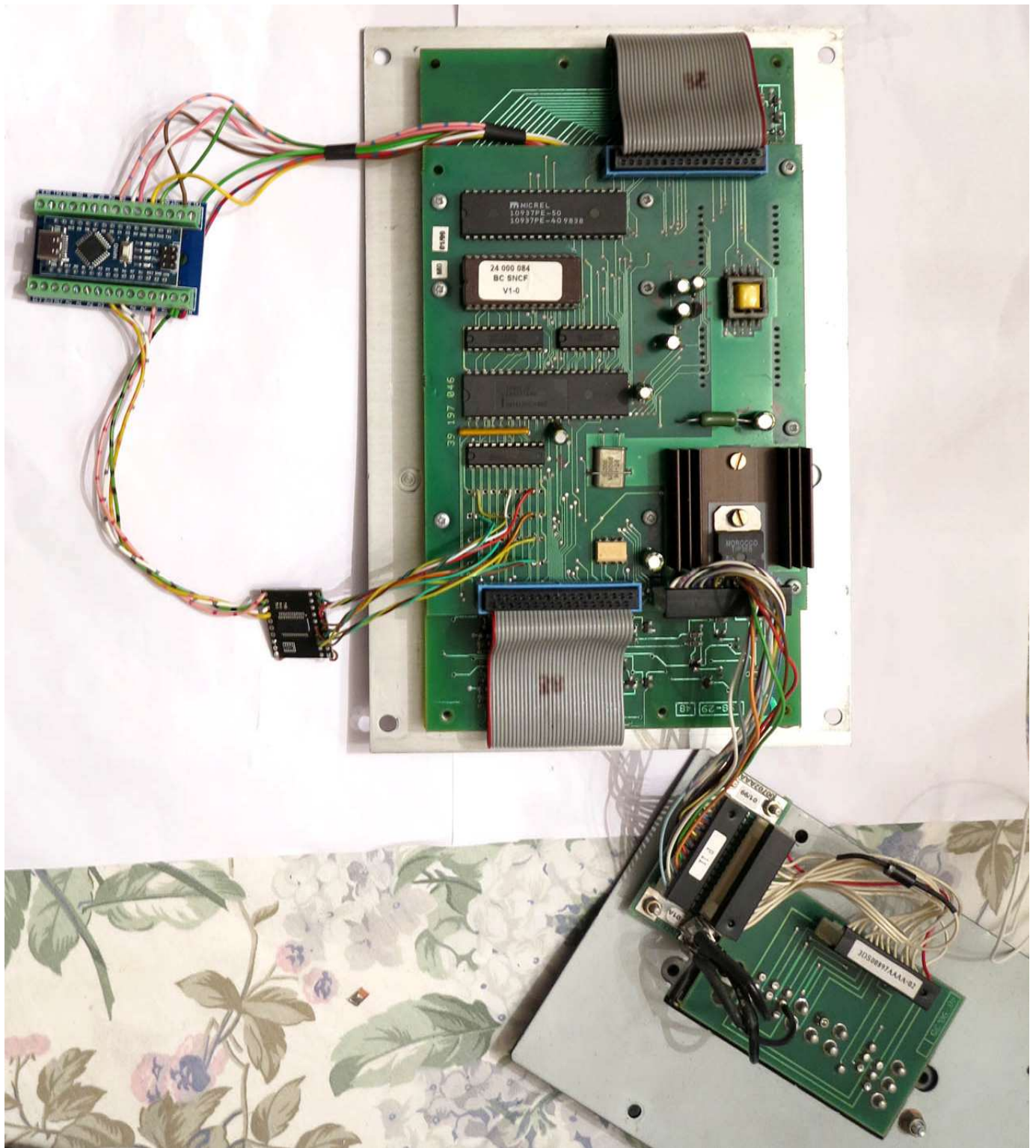
Si on n'a pas d'affichage sur l'écran bleu, remplacer les condensateurs. Vérifier aussi, si il n'y a pas eu d'inversion des signaux RESET, SCLK et DATA.

Si vous griller les circuits intégrés 10937p ou 80C32, on en trouve sur Aliexpress.

Afficheur bloqué. Si après avoir appuyé sur le bouton [Test Défaut], l'affichage reste bloqué, c'est dû à un appel de courant trop important. Dans ce cas, si c'est alimenté en 12 Volts, on peut souder une résistance de 1 Ohm, 3 Watts, ici en gris sur la photo, sur la grosse résistance verte.



11 / VOICI LA MODIFICATION D'UN SECOND BOITIER



A+

ANALYSE D'UN BLOC RST EXISTANT

Pour cette réalisation, j'ai commencé par analyser ce boîtier RST. Pour savoir si ce terminal était utilisable tel que, dans un premier temps, j'ai construit virtuellement le montage du RST.

C'est une carte composée d'un cpu 80C32 de la série de 80C51, d'une eeprom 27C64 contenant le programme, de circuits 74LS373 pour gérer le clavier et l'allumage des touches, et d'un circuit d'affichage 10937P de Rockwell pour piloter l'afficheur fluorescent alphanumérique 16 segments.

J'ai récupéré le contenu de l'eprom 27C64, sur un lecteur d'eprom.

J'ai désassemblé le fichier binaire, pour obtenir un code assembleur 80C32 exploitable.

J'ai utilisé le programme "D52 8052 Disassembler Version 3.3.6",

et construit le fichier "RST SNCF - 27C64.ctl" pour que le programme isole les données dans le fichier, pour ne pas les désassembler.

Contenu du fichier "RST SNCF - 27C64.ctf":

```

; D52 configuration file for RST SNCF - 27C64.bin
; Generated by D52 V3.3.6 on 07/10/24 21:48
b 0022-0022
b 0789-0796
b 07f9-07ff
b 1357-13cf
b 14e0-14e3

```

On désassemble par une ligne de commande DOS : `d52 -db "RST SNCF - 27C64.bin"`

Une fois le programme assembleur refait, on peut l'utiliser avec un simulateur de montage électronique.

Je constate que le RST envoie une trame série à chaque appui sur une touche du clavier, ou une manipulation d'un interrupteur rotatif.

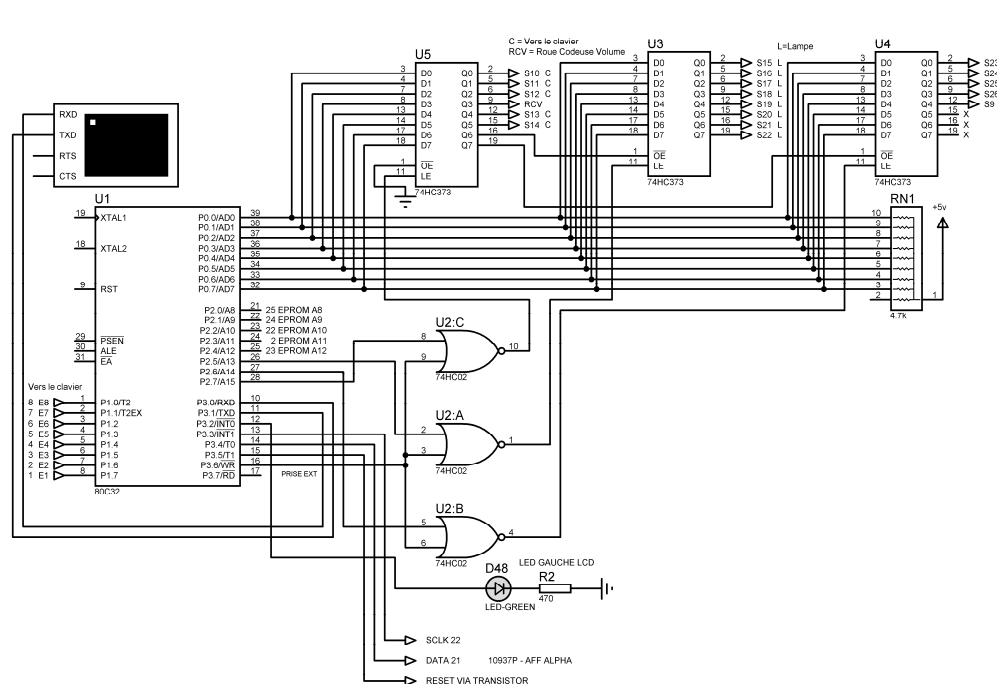
En fait, les échanges de trames sur la liaison série, est complexe. On a des trames de 12 ou 14 octets. Si l'on retrouve facilement ce qu'envoie le RST, retrouver ce qu'attend le RST pour faire fonctionner l'afficheur ou les lampes est trop ardu.

Néanmoins, l'étude du fonctionnement du RST n'est pas perdue et permettra de concevoir un pilotage par une carte Arduino.

Etude préalable et schémas créés pour la simulation de la carte sur ordinateur

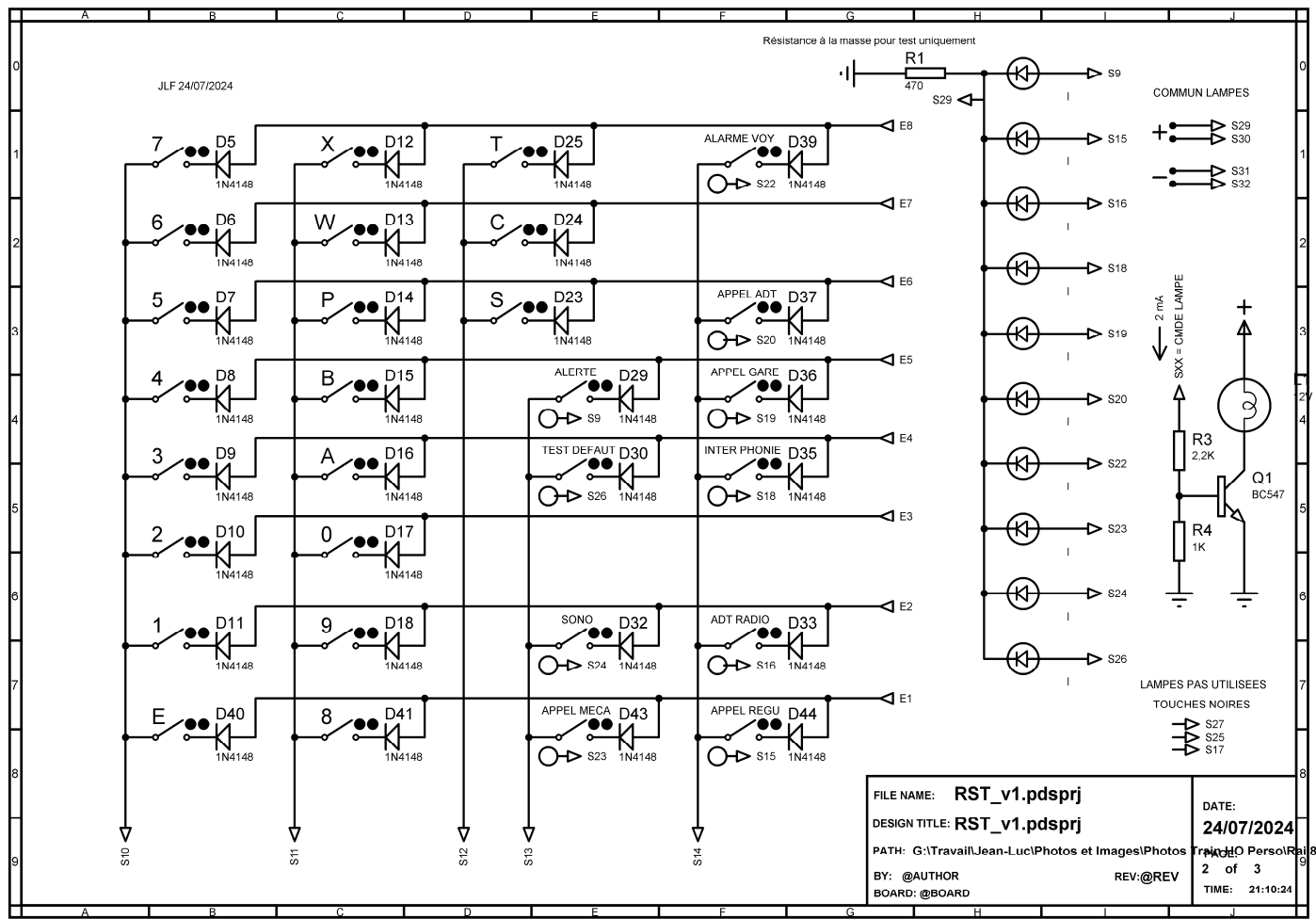
L'écran alphanumérique de 16 segments, et son CI de pilotage le 10937p, ne sont pas représentés.

CPU

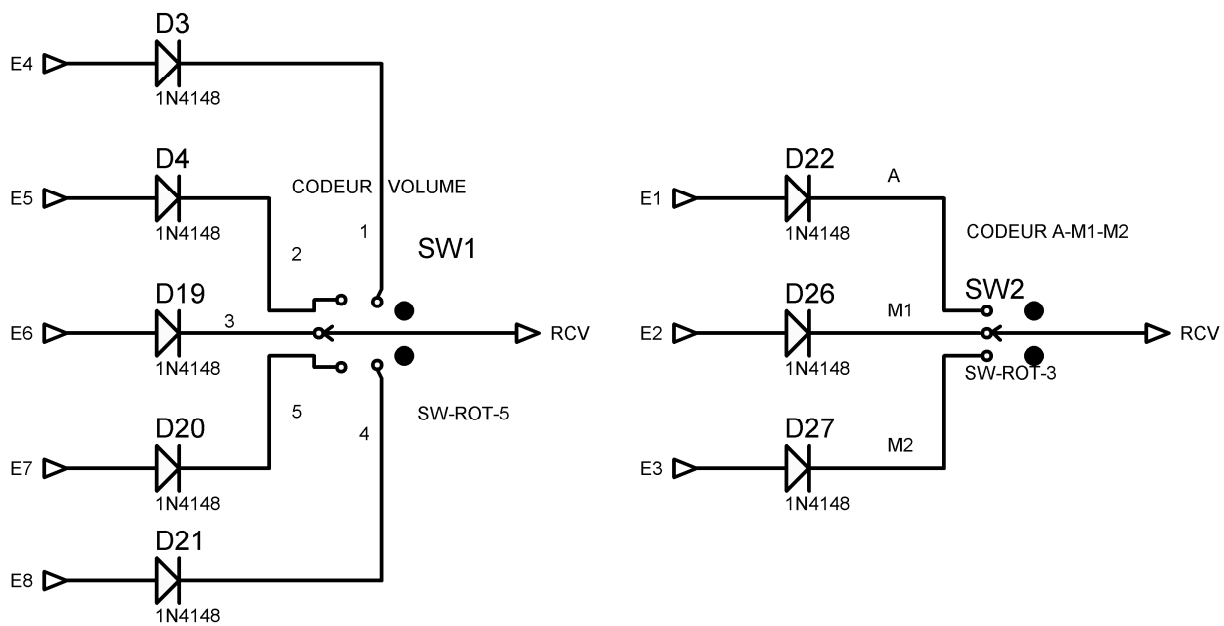


CLAVIER

La numérotation des broches est celle du connecteur.



LES DEUX ROUES CODEUSES



L'ANALYSE DES TRAMES ENVOYÉES PAR LE RST

A chaque action sur la platine RST, le cpu envoie des trames sur la liaison série.

La liaison série est au format 2400 bps, pas de parité, 1 bit de stop.

Les 8 bits de chaque octet, sont de parité paire (*Par logiciel*).

A la mise sous tension, la platine RST envoie une longue série de trame. Il faut donc attendre un moment (*10 secondes*) avant d'utiliser le clavier.

A chaque action sur la platine, elle envoie 4 trames. 3 trames (*De 12 ou 14 octets*) identiques, avec juste l'octet 2 qui change à chaque fois, et au final une 4ème trame de 14 octets.

Toutes ces trames de 12 ou 14 octets, commencent par 0x82 et finissent par 0x8D

J'ai décodé toutes les actions.

Seules les trames envoyées pour l'action sur le bouton volume, ne permettent pas de savoir dans quelle position est le bouton. On sait juste qu'il a été manipulé.

```
82 = Octet d'entête
30, B1, B2, 33, 30, B1,,, = Valeur cyclique identique pour une série de trames envoyées
B1, B2, 33, B4 = Numéro incrémental de trame dans la série
B2 = Trame de 12 octets, B4 = Trame de 14 octets
B1 = Trame intermédiaire, B4 = dernière trame de la série
30
XX XX XX = Numéro de touche enfoncée
30
XX
8D = Octet de fin si 12 octets
30 42 8D = Octets de fin si 14 octets

(-----Trame répétée 3 fois-----)
0 1 2 3 4 5 6 7 8 9 10 11 En rouge la séquence qui permet de retrouver l'action.
[Mise tension] = 82 30 B1 B2 B1 30 35 C6 C6 30 39 8D // 82 30 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[3] = 82 30 B1 B2 B1 30 B1 30 33 30 39 8D // 82 30 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[5] = 82 30 B1 B2 B1 30 B1 30 35 30 39 8D // 82 30 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[6] = 82 B1 B1 B2 B1 30 B1 30 36 30 39 8D // 82 B1 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[9] = 82 33 33 B2 B1 30 B1 30 39 30 39 8D // 82 33 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[0] = 82 30 B1 B2 B1 30 B1 30 41 30 39 8D // 82 30 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[A] = 82 B1 B1 B2 B1 30 B1 30 42 30 39 8D // 82 B1 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[P] = 82 33 33 B2 B1 30 B1 30 44 30 39 8D // 82 33 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[1] = 82 33 B1 B2 B1 30 B1 30 B1 30 39 8D // 82 33 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[2] = 82 B2 B1 B2 B1 30 B1 30 B2 30 39 8D // 82 B2 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[4] = 82 33 B1 B2 B1 30 B1 30 B4 30 39 8D // 82 33 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[7] = 82 B2 B1 B2 B1 30 B1 30 B7 30 39 8D // 82 B2 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[8] = 82 B2 33 B2 B1 30 B1 30 B8 30 39 8D // 82 B2 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[B] = 82 B2 33 B2 B1 30 B1 30 C3 30 39 8D // 82 B2 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[W] = 82 30 B1 B2 B1 30 B1 30 C5 30 39 8D // 82 30 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[X] = 82 B1 33 B2 B1 30 B1 30 C6 30 39 8D // 82 B1 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[INTER PHONIE] = 82 B2 B1 B2 B1 30 B1 33 33 30 39 8D // 82 B2 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[APPEL ADT] = 82 30 B1 B2 B1 30 B1 33 35 30 39 8D // 82 30 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[SONO] = 82 33 B1 B2 B1 30 B1 33 39 30 39 8D // 82 33 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[ADT RADIO] = 82 B1 B1 B2 B1 30 B1 33 B1 30 39 8D // 82 B1 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[APPEL GARE] = 82 33 B1 B2 B1 30 B1 33 B4 30 39 8D // 82 33 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[ALARME VOY] = 82 B1 B1 B2 B1 30 B1 33 B7 30 39 8D // 82 B1 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[APPEL MECA] = 82 B2 B1 B2 B1 30 B1 33 B8 30 39 8D // 82 B2 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[ALERTE] = 82 B2 B1 B2 B1 30 B1 33 C3 30 39 8D // 82 B2 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[S] = 82 30 33 B2 B1 30 B1 B2 B1 30 39 8D // 82 30 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[T] = 82 30 B1 B2 B1 30 B1 B2 33 30 39 8D // 82 30 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[C] = 82 B1 B1 B2 B1 30 B1 B2 B2 30 39 8D // 82 B1 B4 B4 B4 30 35 30 30 30 42 30 42 8D

[E] = 82 B2 B1 B4 B1 30 B2 30 35 30 B1 30 42 8D // 82 B2 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[TEST DEFAULT] = 82 30 B1 B4 B1 30 B2 30 B4 30 B1 30 42 8D // 82 30 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[APPEL REGU] = 82 33 B1 B4 B1 30 B2 30 36 30 B1 30 42 8D // 82 33 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[Roue code A] = 82 B2 B1 B4 B1 30 B2 30 30 30 30 42 8D // 82 B2 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[Roue code M1] = 82 33 B1 B4 B1 30 B2 30 30 30 B1 30 42 8D // 82 33 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[Roue code M2] = 82 30 B1 B4 B1 30 B2 30 30 30 B2 30 42 8D // 82 30 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[Volume 1] = 82 30 B1 B4 B1 30 B2 30 B1 30 35 30 42 8D // 82 30 B4 B4 B4 30 35 30 30 30 42 30 42 8D (Trames V1 à V5 identiques !)
[Volume 2] = 82 B1 B1 B4 B1 30 B2 30 B1 30 35 30 42 8D // 82 B1 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[Volume 3] = 82 B2 B1 B4 B1 30 B2 30 B1 30 35 30 42 8D // 82 B2 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[Volume 4] = 82 33 B1 B4 B1 30 B2 30 B1 30 35 30 42 8D // 82 33 B4 B4 B4 30 35 30 30 30 42 30 42 8D
[Volume 5] = 82 30 B1 B4 B1 30 B2 30 B1 30 35 30 42 8D // 82 30 B4 B4 B4 30 35 30 30 30 42 30 42 8D
```

Le RST envoie une trame série à chaque appui sur une touche du clavier, ou une manipulation des interrupteurs rotatifs.

On va donc pouvoir exploiter ces trames série, pour savoir quel bouton a été manipulé.

Par contre, comme on ne sait pas envoyer des trames série à la platine RST, on va envoyer directement les caractères au 10937P pour l'affichage.

Pour faire fonctionner ce RST, on va utiliser une carte Arduino.

Il nous faut une carte Aduino avec :

- Un port série TX/RX
- Un port I2c pour piloter un afficheur LCD 1602 pour la phase debug.
- Au moins 10 sorties pour allumer les lampes sous les touches du clavier
- 3 sorties pour y brancher le 10937P

A+