

Réplique matérielle d'une machine ENIGMA avec Arduino.

Commencé par Nulentout le Lundi 25 Mars 2024.

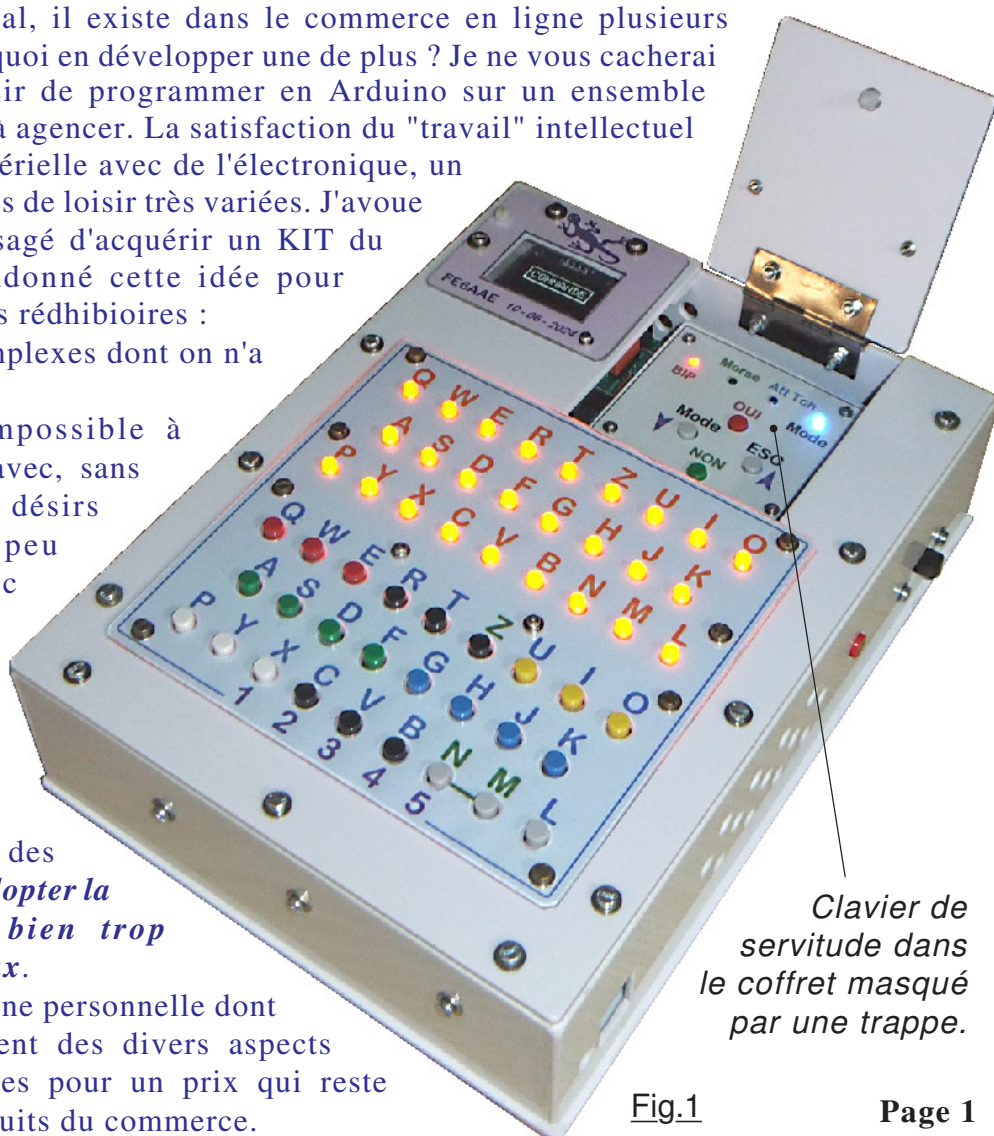
Terminé le jeudi 27 Juin 2024.

C'était promis dans le didacticiel sur la réalisation d'un tout petit boîtier avec une carte Arduino NANO pour émuler une chiffreuse ÉNIGMA en utilisant le Moniteur de l'IDE pour l'interface de dialogue avec l'utilisateur. Ce petit projet décrivait en détails le fonctionnement interne de cette codeuse et par une kyrielle de démonstrateurs émule virtuellement les organes internes de la machine. Il n'est plus question dans ce projet un peu fou de reprendre toutes ces études. On supposera donc que le fonctionnement d'ÉNIGMA vous est parfaitement connu. Si ce n'est pas le cas, le didacticiel qui sert de préambule à ce nouveau projet est disponible sur :

<https://www.robot-maker.com/ouvrages/00-realiser-minuscule-chiffreuse-enigma/>

L'idée n'a rien d'original, il existe dans le commerce en ligne plusieurs répliques. Alors pourquoi en développer une de plus ? Je ne vous cacherais pas que c'est pour le plaisir de programmer en Arduino sur un ensemble particulièrement complexe à agencer. La satisfaction du "travail" intellectuel associé à de la création matérielle avec de l'électronique, un coffret etc. Bref, des activités de loisir très variées. J'avoue que j'ai sérieusement envisagé d'acquérir un KIT du commerce, mais j'ai abandonné cette idée pour plusieurs raisons à mon sens rédhibitoires :

- Des circuits imprimés complexes dont on n'a pas le schéma.
- Un microcontrôleur impossible à reprogrammer. On fait avec, sans pouvoir adapter à des désirs personnels. C'est un peu dommage vu qu'avec Arduino on est largement autonome dans ce domaine.
- Ils n'ont pas toujours un vrai tableau de FICHES croisées avec des cordons à insérer dans des fiches : *À regret je vais adopter la même stratégie, car bien trop compliqué et trop coûteux.*
- On peut se faire une machine personnelle dont on domine cent pour cent des divers aspects matériels et informatiques pour un prix qui reste inférieur à celui des produits du commerce.



Clavier de servitude dans le coffret masqué par une trappe.

Fig.1

01) Étude préliminaire.

Sorte d'habitude, lorsque je développe un nouveau projet, je pars du principe qu'il est valorisant de le partager. Une telle activité prend un temps extrêmement long et s'étend sur plusieurs mois. Aussi, rédiger le didacticiel quand tout est terminé et parfaitement au point n'est pas envisageable, car au fur et à mesure de l'avancement "des travaux", j'oublie les détails techniques, les astuces, les points importants. Aussi, il est préférable pour moi de rédiger les divers documents au fur et à mesure de l'avancement du projet. Cette approche a l'avantage de dévoiler l'intégralité du cheminement, *avec, c'est inévitable, des remises en cause et des changements en cours de route.* L'inconvénient, c'est qu'il ne s'agit pas d'un *tutoriel linéaire* qui part d'un point A pour aller en ligne droite à un point B. En revanche, au point de vue expérience, c'est à mon sens beaucoup plus riche.

➤ Les critères fondamentaux.

Sans que ce soit forcément réalisable, il me semble incontournable d'établir initialement des objectifs (*Très ou trop*) ambitieux, quitte lors du développement à faire des concessions et à adopter des compromis. Voici un résumé de ce à quoi je souhaiterais aboutir :

- Un ensemble dont l'apparence ressemblera le plus possible à celle de l'ENIGMA historique.
- Minimiser le nombre de circuits intégrés ajoutés à la carte Arduino NANO.
- Outre le module des "ampoules électriques, et celui du clavier, limiter au maximum les circuits imprimés typiques à réaliser. (*Pas de FICHES croisées matérielles pour simplifier.*)
- Si facilite la conception, utiliser au maximum des modules du commerce avec leurs bibliothèques.
- Associer un petit afficher OLED pour l'affichage des positions des **Rotors**.
- Trouver des solutions simples utilisables par n'importe quel internaute. Ne nécessiter qu'un outillage banal pour tout "bricoleur en électronique".
- N'utiliser que des composants faciles à approvisionner dans le commerce en ligne.
- Machine autonome en énergie sous forme d'accumulateurs rechargeables.
- Minimiser le coût de l'ensemble.

➤ L'ampleur du défi.

Bien que nous ne soyons jamais à l'abri de mauvaises surprises en programmation, je ne crois pas que ce sera le logiciel qui contribuera à la plus grande difficulté. Normalement, les routines de traitement sont déjà rédigées et bien au point dans le petit projet désigné en page de garde. Les procédures ont fait leurs preuves et il sera probablement possible de les réemployer pratiquement sans les modifier. Comme on va rapidement le voir, la prouesse se situe dans la gestion des Entrées/Sorties. En effet, une estimation préalable conduit à une structure "informatique" qui va se contenter d'une seule carte Arduino NANO pour gérer intégralement le traitement des données. Ce n'est déjà pas élémentaire. ***La difficulté majeure réside dans le nombre des Entrées/Sorties qu'il va falloir gérer.*** En effet, avec une cartes Arduino NANO on dispose de :

- 11 broches binaires + 8 broches analogiques soit 19 E/S configurables.
 - On dispose donc de 19 broches configurables en Entrées ou en Sorties.
- Faisons la liste de ce que l'on désire gérer avec ces interfaces :
- 26 LEDs pour la planche qui simules les ampoules à incandescence de la machine réelle.
 - 26 touches pour le clavier de la codeuse.
 - Un petit afficheur graphique OLED.
 - 6 LEDs pour l'initialisation de la machine. (*6 ou moins ... on verra plus tard.*)
 - Trois ou quatre boutons poussoir pour les servitudes d'exploitation.
 - Un petit bruiteur pour générer du morse ou pour le dialogue Homme machine.
 - Deux multiplexeurs de type PCA9685.

RÉSUMÉ de la situation :

- >>> On dispose potentiellement de 19 broches d'interfaçage.
- >>> On doit gérer environ 67 liaisons extérieures.

Trouvez l'erreur !

02) Gérer environ 67 liaisons extérieures avec 19 broches d'E/S.

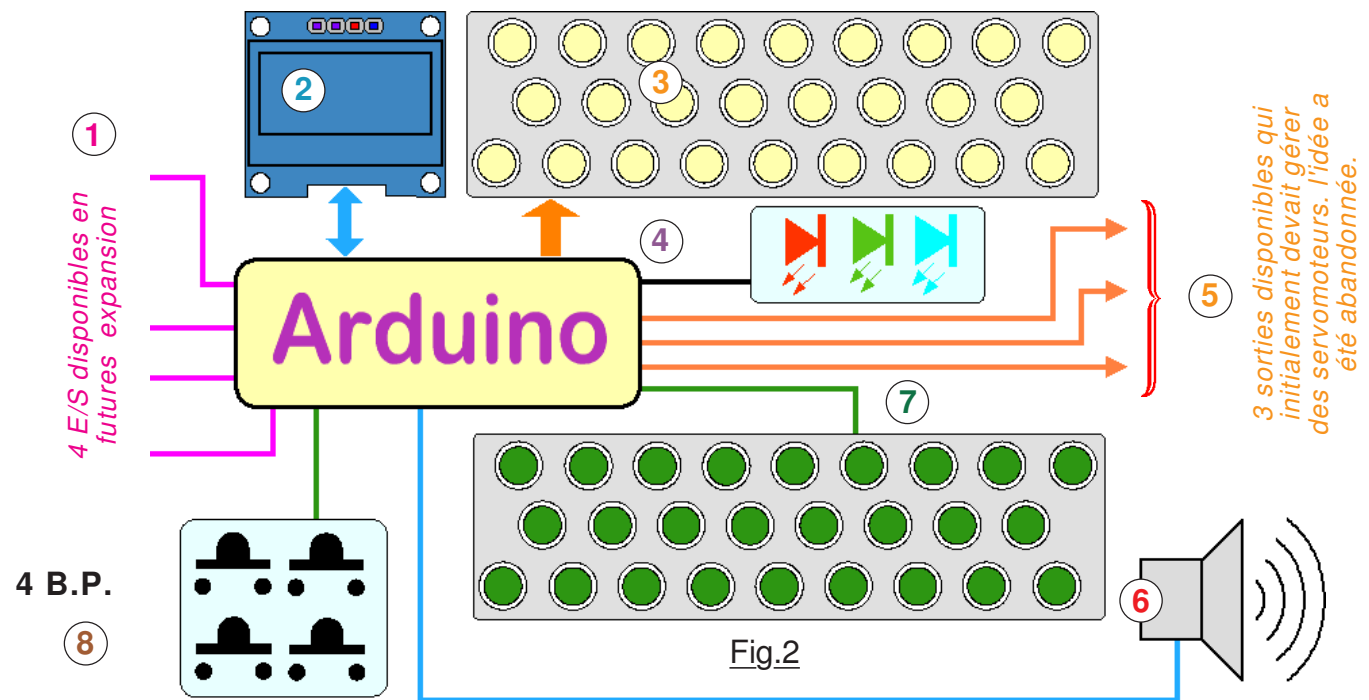
Lister une famille de critères à respecter c'est bien, mais encore faut-il avant de s'engager réellement dans le projet, d'en vérifier la faisabilité. Si vraiment les premières études tendent à prouver que ce n'est pas réaliste, il vaut infiniment mieux abandonner l'idée, que de foncer tête baissée dans le guidon, investir des sommes et surtout des heures et des heures de loisir pour en fin de compte aboutir à un échec. Les premières études "griffonnées sur du papier" laissent entrevoir que la pierre d'achoppement réside dans la gestion des nombreuses Entrées/Sorties avec une seule carte Arduino NANO. (*Gérer des servomoteurs pour faire tourner des rotors 3D a été abandonnée par exemple car générerait trop de parasites.*) Il est donc impératif de s'assurer que les choix effectués sont viables.

Pour éviter d'alourdir le didacticiel par de trop nombreuses parenthèses techniques, ce tutoriel est accompagné d'un grand nombre de fiches au format A5 que je vous invite à imprimer en Recto / Verso et disponibles dans le document FICHES A5.pdf.

Commencer par consulter la [Fiche n°11](#) et la [Fiche n°12](#) pour faire connaissance avec Arduino NANO.

➤ **Architecture globale envisagée.**

Véritable chef d'orchestre, le processeur **Arduino** se charge de l'intégralité des traitements. Outre l'afficheur OLED en **2** il assure la lecture du clavier à vingt-six touches en **7** plus quatre boutons poussoir de servitude en **8** dont la fonction reste à définir en cours de développement. En **3** il faut également gérer les vingt-six LEDs du tableau des ampoules auquel on peut ajouter les



trois témoins en **4** gérant une LED tricolore. Le petit **bruiteur actif** placé en **6** sera piloté par une sortie binaire. Il reste à prouver que tout cela est viable et réaliste. (*Trois servomoteurs en **5** et trois capteurs micro-Switch en **1**. ont été abandonnés.*)

➤ **Répartition et justification des Entrées/Sorties.**

Avant de vérifier le bienfondé de la structure adoptée en Fig.2 qui résulte directement de la répartition des broches, passons en revue les critères techniques qui ont conduit à la distribution des lignes d'interfaçage indiquée en Fig.1 de la [Fiche n°1](#).

C'est volontairement que les deux broches analogiques **A6** et **A7** sont **Non Utilisées** pour que l'on puisse librement utiliser une carte Arduino UNO si cette dernière emporte votre préférence car un exemplaire disponible "traîne" dans vos tiroirs. Initialement le **BUZZER actif** était branché sur **A13**. Toutefois, cette broche est sur la LED d'Arduino qui s'illumine une fraction de seconde lors d'un RESET, ce qui engendrait un BIT intempestif. Du coup l'ensemble du multiplexage du clavier a été décalé "vers le haut". Les broches de lecture du multiplexage en entrées ont été choisies contigües de **D3** à **D8** car généralement cette approche peut simplifier la gestion informatique du clavier. Même critères pour **D9** à **D13** qui en sorties assurent le multiplexage en

balayage ligne. Toutes les broches binaires sont affectées, puisque **D1** et **D2** sont réservées pour le dialogue série avec le **Moniteur de l'IDE** et le téléversement des démonstrateurs. Initialement prévues pour la lecture des trois capteurs d'orientation des **Rotors** les entrées analogique **A0** à **A2** sont non utilisées et restent disponibles en Entrées ou Sorties. Pour la gestion de la ligne I2C qui pilote l'afficheur OLED et les deux multiplexeurs PCA9685 nous n'avons pas le choix. Les bibliothèques de ces périphériques imposent les deux broches **A4** et **A5**. Au final, il reste encore la broche analogique **A3** qui peut aussi bien servir en entrée qu'en sortie binaire pour un éventuel complément non encore prévu et trois sorties sur le multiplexeur n°2.

ATTENTION : Il existe de nombreux clones de la carte Arduino NANO disponibles en ligne. Toutes les références ne sont pas forcément compatibles avec l'**IDE** de base et peuvent nécessiter l'installation d'un pilote spécifique pour être compatibles. (*En particulier certains clones importés de chine.*) Aussi assurez-vous de la compatibilité des références que vous allez approvisionner. Par exemple j'ai commandé un groupe de cinq parfaitement compatibles sur : https://www.amazon.fr/gp/product/B078S8BJ8T/ref=ppx_yo_dt_b_asin_title_o01_s00?ie=UTF8&psc=1
Naturellement il n'est pas du tout obligatoire d'en approvisionner cinq d'un coup, mais comme j'en utilise souvent sur de multiples petites applications, j'ai pris de l'avance !

03) Étude de faisabilité.

Multiplexer à outrance constitue la clef de voûte de notre entreprise. C'est la solution classique qui permet de gérer infiniment plus d'éléments indépendants que l'on ne dispose de broches d'E/S sur le(s) processeur(s) inclus dans l'organisation matérielle d'un projet. On peut multiplexer par le truchement d'une ligne I2C ou par la technique "matrice lignes/Colonnes". Dans cette réalisation les deux techniques vont cohabiter. Pour évaluer la viabilité des solutions envisagées, nous allons passer en revue chacune des facettes particulières de ce projet.

➤ **La ligne de multiplexage I2C.**

Avant de pouvoir mettre en œuvre le démonstrateur **P01** il nous faut impérativement faire connaissance avec l'indispensable technique **I2C** qui permet de piloter un très grand nombre de périphériques avec seulement deux fils et une masse commune **GND**. Dans ce but, il nous faut absolument consulter la **Fiche n°3** et la **Fiche n°4**. Nous y apprenons que pour traiter une telle ligne on va se contenter d'utiliser la bibliothèque **<Wire.h>** qui gère les protocoles **I2C** sur les deux broches analogiques **A4** et **A5**. Il importe de noter que **<Wire.h>** n'est qu'une "sous-couche" logicielle pour mettre en œuvre les deux sorties filaires **A4** et **A5**. À elle seule elle ne sert à rien. **Il faut lui adjoindre une bibliothèque spécifique pour chaque module du commerce** qui dialoguera en **I2C**.

➤ **Présentation du multiplexeur PCA9685.**

Concrètement, ce module du commerce particulièrement bien pensé et très compact a été sélectionné, car à lui seul il peut fournir 16 sorties **PWM** indépendantes. Consultons la **Fiche n°9** qui montre qu'avec deux de ces modules on va pouvoir gérer jusqu'à 32 LEDs ... et avec seulement les deux broches **A4** et **A5**. Nous ne pouvons pas faire l'économie de la **Fiche n°5** et de la **Fiche n°6** qui décrivent la bibliothèque **Adafruit-PWM-Servo-Driver-Library-master** qui accompagne le module PCA9685. C'est précisément cette couche logicielle spécifique au module électronique qui se servira de **<Wire.h>** pour dialoguer avec la carte Arduino NANO. Notons au passage que "**Servo-Driver**" fait référence à la spécificité de ces modules qui peuvent gérer des servomoteurs par des signaux PWM. Tous les servomoteurs prévus pour de la petite robotique sont pilotés par un signal de type **PWM** (**Pulse Width Modulation : Impulsions à modulation de largeur.**) dont la fréquence de répétition est de 50Hz. Mais ici on ne va piloter que des LEDs, donc la fréquence peut être bien plus élevée. Par exemple 1000Hz éliminant tout phénomène de scintillement même si la LED clignote à cadence rapide.

Préalable à l'utilisation de **P01** nous devons placer "en cascade" deux modules PCA9685. Cette obligation impose de modifier l'adressage d'au moins l'un des deux circuits imprimés. Pour simplifier on conserve **x40** pour le premier, et l'on adresse le deuxième en **x41** ce qui impose de faire une petite soudure. La **Fiche n°7** décrit comment changer l'adressage de l'un des

modules alors que la [Fiche n°8](#) précise comment les chaîner électriquement. Sur [Image 01.JPG](#) on voit en "macro" la soudure à effectuer, alors que sur [Image 02.JPG](#) on observe le deuxième connecteur à ajouter au module [x40](#) pour chaîner le deuxième module électronique.

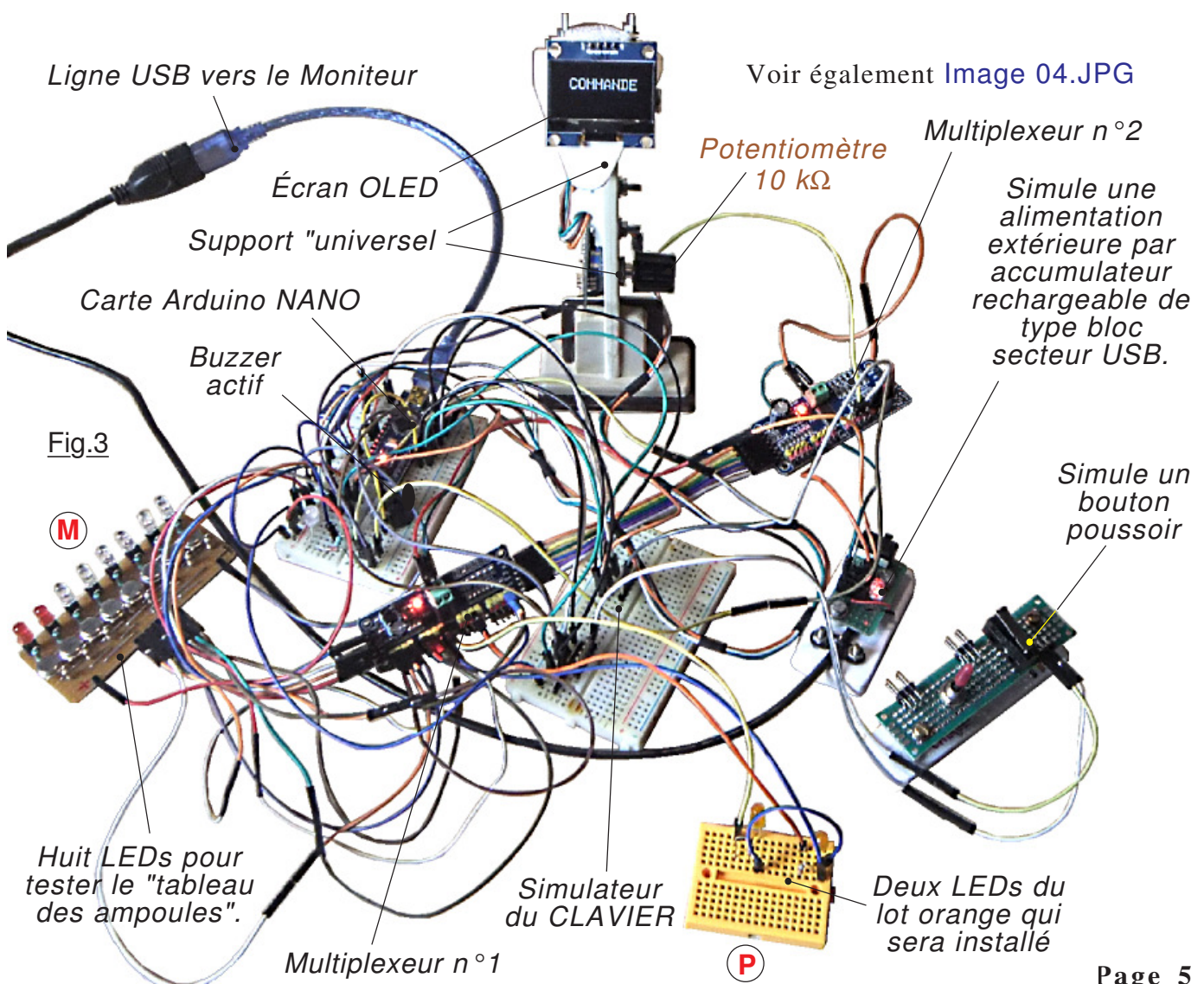
04) Confirmation du bienfondé de la répartition des interfaces.

C'est souvent lors de "l'intégration des systèmes" que l'on découvre des incompatibilités ou des interactions sources inépuisables de complexité. À partir du moment où l'on fait cohabiter des entités de technologies différentes, on peut rencontrer des perturbations de proximité et ce d'autant plus que l'on est dans un domaine qui "vibre" avec des signaux électriques binaires à des fréquences élevées, générant par nature des harmoniques à l'infini. Nous allons dans ce chapitre nous assurer que l'ensemble de la répartition des E/S du schéma de la Fig.1 sur la [Fiche n°14](#) est viable.

➤ **Compatibilité de l'afficheur OLED avec les deux circuits PCA9685.**

Partageant la ligne **I2C** avec les deux multiplexeurs, il faut impérativement s'assurer que l'on n'aura pas de problème d'adressage. On commence par s'instruire avec la [Fiche n°2](#) qui nous apprend que la couche logicielle qui va gérer ce composant est constituée de la bibliothèque spécifique **<U8glib.h>** disponible dans le dossier **<Documents>** vous évitant ainsi une recherche laborieuse sur Internet. C'est celle que j'utilise et qui ne m'a jamais posé de problème. Dans le même dossier je vous propose un petit livret personnel qui résume les méthodes de cette bibliothèque nommé **Bibliothèque U8glib.pdf** et formaté pour en faire un petit livret au format A5 bien commode. La technique d'assemblage du livret est précisée dans **Réaliser un petit livret.pdf**.

Avec l'utilitaire **P00_Test_ligne_I2C.ino** disponible dans **<OUTILS>** on confirme l'adresse **I2C 0x3C** de la documentation qui en hexadécimal donne **0x78** qui ne devrait pas interférer avec les multiplexeurs PCA9685 ce que semble confirmer le démonstrateur **P01_Programme_de_base.ino**.



NOTE : Une foule de polices de caractères est disponible. Le livret explique comment déclarer celle qui correspond à votre préférence. Pour lister les caractéristiques des ces polices disponibles aller sur : <https://nodemcu-build.com/u8g-fonts.php#collapseOne>

➤ OLED et PCA9685 font bon ménage.

Avec le démonstrateur **P01** on entre dans le vif du sujet, c'est à dire que l'on compile un logiciel qui va servir de fondation aux autres démonstrateurs qui vont suivre, aboutissant à l'étape ultime du programme d'exploitation de notre réplique d'Énigma. C'est à dire qu'à partir d'ici, certaines séquences de traitement seront "définitives" et surtout on va systématiquement adopter l'architecture électronique qui résulte des affectations de la **Fiche n°1**. Le démonstrateur que l'on téléverse dans l'ATmega328 est conçu pour vérifier l'intégralité des périphériques gérés par la ligne **I2C**. Sur RESET le module OLED et les deux modules PCA9685 sont initialisés. L'écran affiche "**BONJOUR**" prouvant ainsi la compatibilité entre les trois circuits électroniques. Les protocoles d'utilisation de **P01** sont listés dans la **Fiche n°15**. Il importe toutefois de se reporter à la **Fiche n°10** car la répartition des lettres affectées sur les sorties des multiplexeurs correspond à la géométrie du tableau simulé des ampoules à incandescence. Il faudra un peu jongler avec le "fouillis" Fig.3 des branchements provisoires entre les modules et les plaquettes d'essais. La LED triple est directement branchée sur le multiplexeur n°2. Pour les autres LEDs, je ne disposais que du petit module **M** auquel j'ai ajouté la plaquette **P** pour y placer deux LEDs oranges du lots approvisionné et ainsi déterminer leur valeur de **1kΩ** pour la limitation du courant qui les traverse.

Éventuellement, voici le lien ou j'ai approvisionné ces composants :

https://www.amazon.fr/dp/B0CBX3JQFV?psc=1&ref=ppx_yo2ov_dt_b_product_details

L'offre précise un courant nominal de 20mA. Leur rendement est très bon, et les 2,7mA qui les traversent sont largement suffisants pour obtenir une luminosité tout à fait appropriée.

Outre la vérification de l'intégralité des 32 LEDs, la sortie pour le BUZZER est également testée. Toutefois, il faudra au préalable valider le bruiteur avec la commande [ESPACE], l'état de cette option étant précisé dans la fenêtre du **Moniteur**.

Lorsque toutes les commandes propres à **P02** auront été testées, mis à part le CLAVIER l'intégralité des périphériques aura été validée. Valider le CLAVIER est précisément l'objet du chapitre qui suit.

➤ Multiplexage matriciel.

Classique dans le domaine de la programmation, la technique consiste à organiser l'exploration d'un clavier en l'organisant sous forme d'une matrice de **L** lignes et de **C** colonnes. Le nombre de touches que l'on pourra ainsi différencier sera égal à **L** fois **C**. **ATTENTION : Ce type de structure électronique n'est possible que pour des touches à contact travail et repos isolé.** Si aucune des touches n'est cliquée, l'intégralité des broches d'interfaçage doivent être isolées les unes des autres. Du coup, cette technique n'était pas utilisable pour explorer le tableau des fiches croisées par exemple, raison pour laquelle sa matérialisation a été abandonnée. Le petit tableau de la Fig.4 résume la combinatoire, pour **L** et **C** compris entre 2 et 6. La combinaison **2 x 2** ne justifie pas du multiplexage puisqu'elle utilise le même nombre de broches que pour une sélection directe. Comme le CLAVIER présente 26 lettres, on doit choisir dans ce tableau une combinaison d'au moins cette valeur. Pour optimiser le matériel, la plus proche est soit **5 x 6** soit **6 x 5**. Si mathématiquement ces deux solutions sont strictement équivalentes, dans leur mise en œuvre elle ne l'est pas. En effet, en consultant la **Fiche n°17** on observe que chaque colonne **C** de la matrice sera réunie par une résistance de **10kΩ** à **GND** pour forcer sur cette dernière un état "**0**". Du coup, pour simplifier au maximum le matériel, moins il y a de colonne meilleure est la solution. La combinaison **6 x 5** serait donc préférable. Pourtant c'est **5 x 6** qui a été retenue. Tout simplement, lors du développement, je n'ai pas pensé à ce petit détail. Aussi, si informatiquement reprendre le programme serait assez élémentaire, je n'ai pas le courage de refaire la documentation, et en particulier la **Fiche n°1** et la **Fiche n°13**. Aussi, par paresse, je n'en suis pas à économiser un résistor. Puisque l'on aborde l'aspect logiciel, je vous invite à consulter la **Fiche n°14**

Fig.4

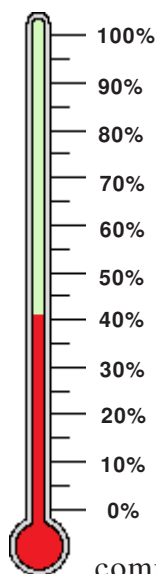
	2	3	4	5	6
2	4	6	8	10	12
3	6	9	12	18	24
4	8	12	16	20	24
5	10	15	20	25	30
6	12	18	24	30	36

qui traite de la façon dont est géré le CLAVIER à 30 boutons poussoir par le démonstrateur [P02_Validation_du_CLAVIER.ino](#) que l'on doit téléverser dans l'ATmega328. La [Fiche n°16](#) pour sa part donne la liste des commandes qui cette fois sont imposées par le CLAVIER simulé sur [Image 03.JPG](#). On commence à s'approcher de la structure définitive, car l'intégralité du matériel est pratiquement validée. Comme la grille laisse libre quatre touches, ces dernières serviront à gérer les servitudes d'exploitation de notre Énigma. Leur fonction sera définitivement affectée lors du développement qui générera forcément des besoins à satisfaire.

Comme le clavier "retourne" les 26 lettres de l'alphabet, il fallait attribuer quatre caractères à ces touches qui restent disponibles. Par simplification ce sont les chiffres '1', '2', '3' et '4' qui ont été choisis. Arbitrairement, si au moment du sondage du clavier par la procédure [Tester_le_Clavier\(\)](#) aucune touche n'est active, le caractère retourné sera arbitrairement le '@'.

Fig.5

➤ Un premier bilan sur l'évolution du programme.



Normalement, vous avez été voir le didacticiel précisé dans la page de garde de ce document, ou tout au moins vous l'avez "survolé". Du coup, vous savez que j'affectionne l'artifice qui consiste à représenter la place occupée par le programme en cours de développement sous forme d'un thermomètre. Avec la Fig.5 je ne résiste pas à ce petit plaisir pour poursuivre cet amusement dans ce tutoriel. *(Du reste cette technique me sert de statistiques en cours de développement pour savoir si l'on conserve la facilité de placer les textes affichés directement dans le programme, ou s'il faut impérativement en loger en EEPROM pour dégager de la place car le code OBJET approche la saturation de la zone dédiée.)*

Nous n'avons rédigé que quelques procédures pour commencer à valider l'architecture matérielle et déjà 41% de l'espace réservé au programme est consommé. Pourtant, à ce stade du développement il n'y a pas vraiment de raison de s'inquiéter. En effet, le codage fait encore moins de 200 lignes, du coup nous avons l'impression qu'il ne fait pas grand chose. Détrompez-vous. Quelques instructions "discrètes" se goinfrent en OCTETs comme la gestion de la ligne série USB du Moniteur ou [Wire.h](#) pour traiter l'I2C. Par ailleurs, on a ajouté la bibliothèque [Adafruit-PWM-Servo-Driver-Library-master](#) qui accompagne le module PCA9685. Elle est très performante mais gourmande en code objet. Enfin, et ce n'est pas la plus économe, [U8glib.h](#) particulièrement aisée à utiliser, mais qui discrètement amène en mémoire une table de génération de la police de caractères. *Bref, l'essentiel est en place.*

05) Commencer à réaliser les circuits imprimés supportant l'électronique.

Durant les études qui précèdent, nous avons évalué ce qui est possible et ce qui a été abandonné. Par exemple des rotors 3D animés par des servomoteurs a été définitivement écarté, car leur fonctionnement n'était pas stable et surtout ces éléments généraient des perturbations incompatibles avec la fiabilité du programme. Le reste du [schéma électronique](#) a été validé et *ne changera* probablement *plus*. Continuer avec le méli-mélo de la Fig.3 ne serait pas sérieux, il est temps de commencer à façonner les cartes électroniques de notre réplique d'ÉNIGMA.

➤ La carte principale qui supporte le module Arduino NANO.

Véritable chef d'orchestre, c'est l'ATmega328 qui par ses E/S et le truchement de la ligne I2C sur [A4](#) et [A5](#) traite intégralement les 67 liaisons d'interfaçage avec les éléments périphériques. Aussi, il est naturel de commencer par le circuit qui supportera la carte Arduino NANO et les divers connecteurs HE14 de liaison avec son environnement. Gouverner c'est prévoir ! Aussi, divers détails envisagent des compléments éventuels pour "le futur". Par exemple un connecteur à deux broches permettrait de brancher un bouton de RESET extérieur. *(Peut s'avérer utile lors d'une reprogrammation sur site lorsque tout sera intégré dans un coffret.)* Surtout deux petits HE14 à trois broches sont prévus pour pouvoir croiser la polarité d'alimentation de l'afficheur OLED. Ainsi le circuit imprimé restera inchangé quel que soit le modèle approvisionné ou un de rechange qui serait différent. Toutes les broches d'interfaçage non utilisées [A3](#), [A6](#) et [A7](#) ainsi que [A0](#) à [A2](#) sont disponibles sur des connecteurs répartis à la périphérie du circuit imprimé. Autant dire que coté possibilités potentielles on reste "dans le luxe". Enfin, sans qu'elle ne soit

spécifiquement étendue, une zone libre permet si nécessaire d'ajouter quelques composants sur le circuit si nécessité s'en faisait sentir. Il est probable que ces possibilités ne seront pas exploitées, ceci dit, "ça ne mange pas de pain ... et qui peut le plus peut le moins".

La **Fiche n° 17** fournit le dessin du circuit imprimé vu coté composants et vu coté pastilles. Les six résistances de polarisation des entrées de gestion du clavier sont placées en épingles sous la carte Arduino NANO. **Il importe de les couder court et les souder au raz du circuit imprimé** pour ne pas qu'elles talonnent avec le module situé au dessus. Vérifier également que le condensateur **C** de 0,47µF soit suffisamment petit pour les mêmes raisons. La **Fiche n° 18** plus épurée indique les branchements à effectuer avec les modules extérieurs à cette carte électronique. Sur **Image 05.JPG** on commence par implanter les connecteurs HE14 et les six résistances soudées en épingles. Elles sont plus visibles sur **Image 06.JPG** saisie en "macro". Sur **Image 07.JPG**, **Image 08.JPG** et **Image 09.JPG** le petit circuit imprimé est achevé. Noter sur **Image 08.JPG** la présence de ponts isolés.

V alider le circuit imprimé reste relativement élémentaire. **La carte Arduino n'étant pas sur le support HE14** on commence par vérifier visuellement avec un "compte fil" qu'il n'y a aucune liaison intempestive entre deux pastilles voisines aux soudures des connecteurs. Puis on vérifie qu'il n'y a aucun contact entre des lignes voisines avec un contrôleur de continuité. *(Ce dernier peut sonner s'il est sensible et détecter des résistances relativement importantes. Par exemple c'est le cas entre le +Vcc et GND généré par la présence du condensateur de 470µF.)* On branche une alimentation USB sur le petit connecteur d'alimentation à deux broches et on vérifie la propagation de **GND** et du **+5Vcc** partout où leur présence est implicite. On force du **+5Vcc** à la place de la broche **D2**, le Buzzer doit sonner. On position des strap pour l'alimentation de l'afficheur OLED et on vérifie la polarité sur le connecteur. On coupe l'alimentation et on relie la ligne de l'afficheur graphique. On installe la carte Arduino NANO programmée avec **P2**. On met sous tension en alimentant cette dernière par sa prise mini USB. OLED doit afficher **COMMANDE**. On coupe l'alimentation. On relie le connecteur dédié aux multiplexeurs PCA9685. On relie provisoirement la LED tricolore placée sur un support de développement. À la mise sous tension elle doit clignoter en bleu. Toujours provisoirement on effectue quelques pontages sur le connecteur du clavier et on observe le résultat avec le **Moniteur de l'IDE**. On a ponté deux ou trois LEDs orange sur le multiplexeur. Le comportement observé étant celui attendu, on peut affirmer que le circuit est bon pour le service.

➤ Réalisation du CLAVIER à 26 touches.

C' est la suite logique pour pouvoir continuer à développer de façon commode sans se torturer avec un artifice tel que celui d'**Image 03.JPG** où tester une lettre particulière impose une étude des branchements des connecteurs en liaison avec le schéma de la **Fiche n° 13** ... une galère inacceptable sur le long terme. L'étude du circuit imprimé est un vrai "chassé/croisé". En effet, en consultant la **Fiche n° 13** on observe que la matrice Lignes/Colonnes est organisée dans une répartition verticale en ordre alphabétique. Ce n'est pas le fruit du hasard. **Cette organisation permet un traitement informatique particulièrement économe en instructions :**

```
Caractere = (Ligne + (5 * Colonne)) - 6;  
// Enlever 6 pour avoir une valeur qui va de 0 à 29.  
switch (Caractere) {  
    case 26 : {TOUCHE = '1'; break;}  
    case 27 : {TOUCHE = '2'; break;}  
    case 28 : {TOUCHE = '3'; break;}  
    case 29 : {TOUCHE = '4'; break;}  
    default : TOUCHE = char(Caractere + 65);}  
// Ci-dessus le + 65 pour aller de 'A' à 'Z'.
```

Cet algorithme est organisé pour que la variable Caractère contienne en sortie de procédure le code ASCII de la touche activée. C'est impératif car ce codage universel est employé partout en informatique et en particulier par la bibliothèque de l'afficheur OLED.

La difficulté, c'est que les touches du clavier sont organisées comme celles du tableau des ampoules électriques d'Énigma qui lui est réparti comme montré sur la **Fiche n° 10**. Du coup il faut réunir les touches "dans l'ordre alphabétique" en correspondance avec les lignes **L1** à **L5** et les colonnes **C1** à **C6** du schéma de la **Fiche n° 13**. Cette adaptation "géométrique" impose les nombreuses lignes électriques croisées du circuit imprimé du clavier.

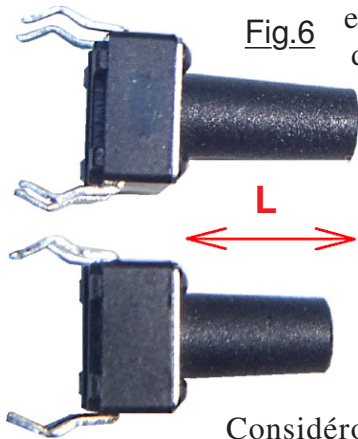
➤ Un joyeux "chassé / croisé".

Sans aller jusqu'à dire que la réalisation de ce circuit imprimé est délicate, elle présente toutefois quelques pièges à éviter et impose de travailler avec méthode. Personnellement j'ai approvisionné mes touches en ligne sur :

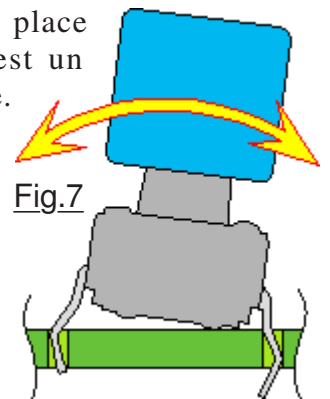
https://www.amazon.fr/dp/B0CB3FCDX4?psc=1&ref=ppx_yo2ov_dt_b_product_details

Les éléments fournis sont relativement petits et parfaits pour réaliser notre clavier. La première étape va consister à trier les 26 boutons poussoir que l'on va installer sur le circuit imprimé décrit

en **Fiche n°19** à **Fiche n°21**. **Attention**, dans ce lot il y a deux sortes d'éléments. Le dépassement (*Voir la Fig.6*) des tétons coniques sur lesquels s'emmanchent les cabochons de couleur présentent deux longueurs **L** différentes. Si l'on désire que le dépassement de tous les boutons poussoir soit homogène il faut prendre soit les courts, soit les longs. **Pour ma part j'ai opté pour les plus longs**. Ces composants ont des broches pliées en "griffes" et se positionnent parfaitement dans une grille de trous au pas standard de 2,54mm. Quand on les insère dans les trous de traversée, ils se positionnent très facilement et restent en place quand on retourne le circuit imprimé ce qui est un avantage incontestable pour l'opération de soudage.



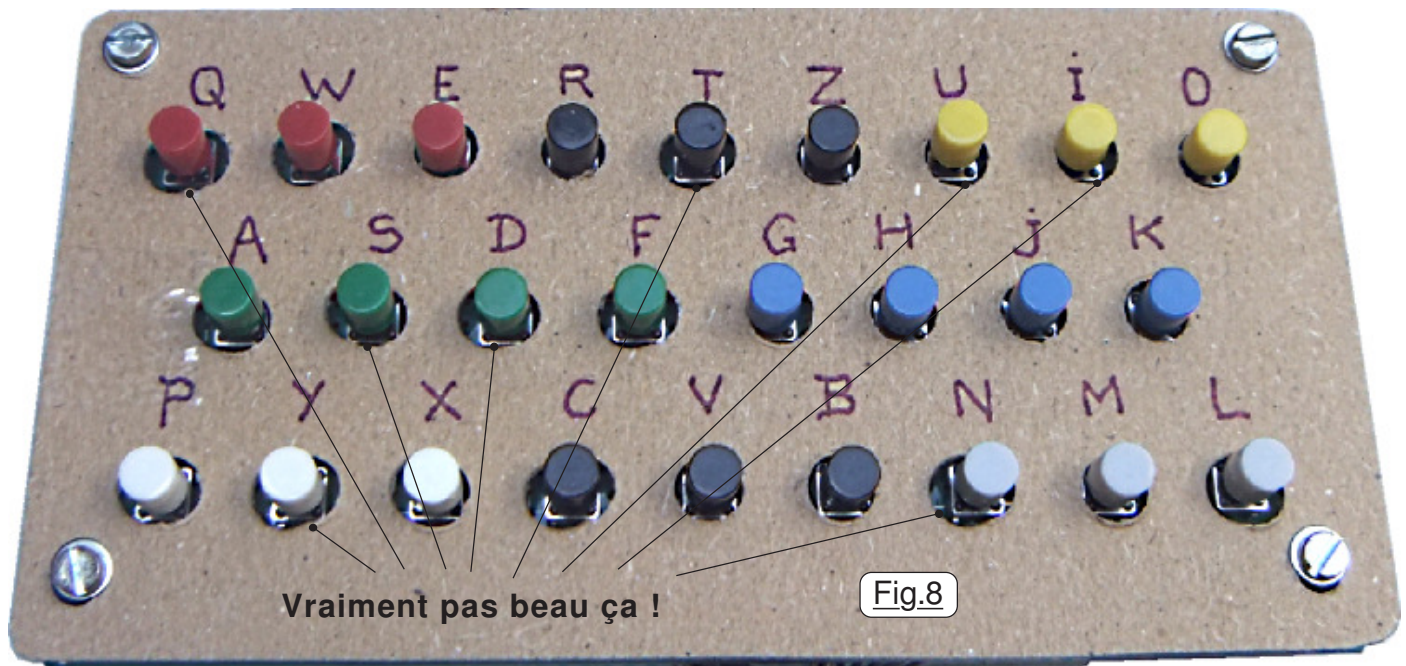
Considérons la Fig.7 : Il est vital, de les orienter parfaitement à la verticale pour des raisons esthétiques et surtout pour faciliter le perçage de la plaque du dessus qu'ils doivent traverser. D'un jeu minimal dépend la beauté de notre coffret. Si toutes les touches sont parfaitement verticales, la "grille" de perçage sera régulière et facile à tracer. Il faudra absolument éviter un "massacre" tel que celui de la Fig.8 où le carton provisoire pour repérer les lettres a été réalisé "à l'arrache". Pour le coffret il faudra absolument faire mieux. Revenons à la réalisation du circuit imprimé. Si on observe **Image 10.JPG** on constate que lorsque les touches sont insérées dans les orifices du circuit imprimé, les griffes dépassent à peine coté pastilles cuivrées. Pour arriver à un alignement tel que celui d'**Image 11.JPG** on doit impérativement agir dans un ordre très strict :



- 1) On insère les 26 touches sur le circuit imprimé.
- 2) On aligne bien verticalement tous les boutons poussoir.
- 3) On retourne la plaque que l'on pose sur les tétons car les cabochons ne sont pas encore placés. (*Avec le léger serrage des griffes dans les trous de traversée les touches ne bougent pas.*)
- 4) Sur les 26 éléments on soude une seule broche, par exemple celle en haut à gauche.
- 5) On retourne le circuit imprimé. On effectue un dernier alignement vertical sur tous les B.P.
- 6) Plaque retournée on soude les trois autres broches. On aboutit au résultat d'**Image 12.JPG**.
- 7) Sur le dessus et sur le dessous on soude les petits fils rigides qui remplacent les pistes cuivrées pour arriver au résultat d'**Image 13.JPG** et celui d'**Image 14.JPG**. Noter sur **Image 15.JPG** que si la liaison est courte, pas besoin de mettre un fil de cuivre, un simple "paquet" de soudure fait l'affaire. **Pour les lignes longues**, on voit sur **Image 16.JPG** qu'un pont réalisé avec un petit bout de fil rigide soudé coté cuivre sert à les maintenir bien en place.

Attention : On passe son temps à travailler dessus, dessous et à décrypter les dessins des circuits imprimés donnés en **Fiche n°19** et **Fiche n°20**. Il faut être très attentif, les risques de se tromper sont considérables. Par exemple sur **Image 13.JPG** et **Image 14.JPG** qui ont été saisiés avant la vérification du circuit, il y a quatre erreurs à corriger. Il faut reconnaître que j'ai imprimé les dessins sur papier pour les opérations de soudage. Par moment, l'interprétation du dessin est délicate, **il est alors préférable de comparer directement sur l'ordinateur**.

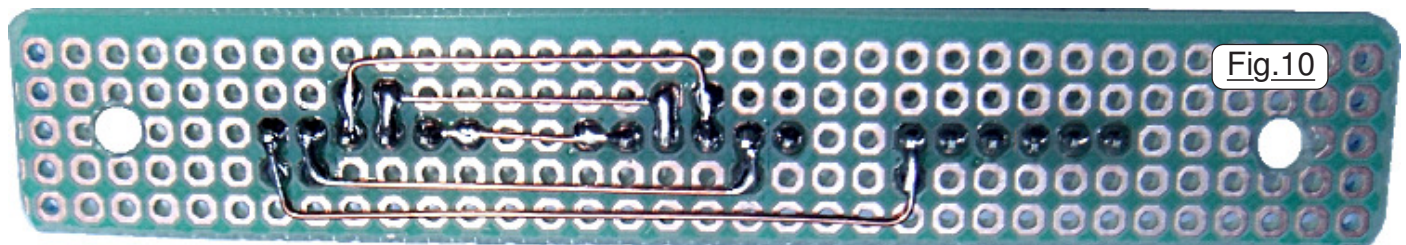
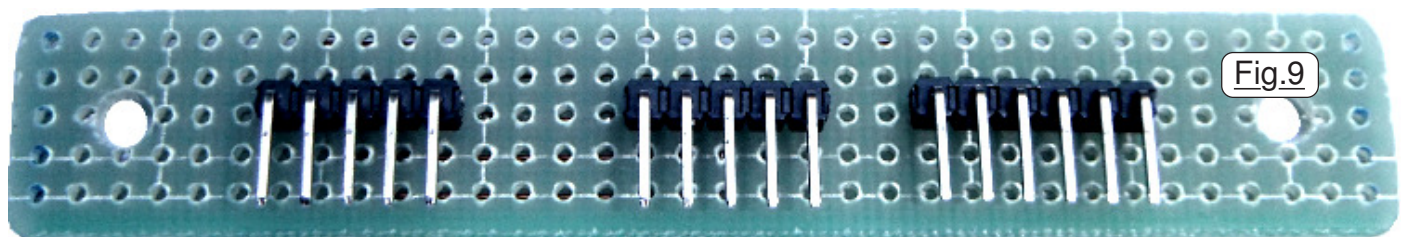
Pour faciliter la réalisation et surtout la maintenance, le clavier n'est pas relié au circuit imprimé principal directement par un toron de fils électrique torsadés, mais possède son propre connecteur. Comme les HE14 de liaison doivent être sur le dessous, ils sont placés sur un petit circuit imprimé qui assure les liaisons avec les lignes électriques de la matrice. Ainsi, le CLAVIER



est une unité totalement indépendante. Noter que le clavier sera situé au dessus du circuit imprimé principal supportant la carte Arduino NANO. Comme on désire au final un boîtier pas trop volumineux, c'est la raison pour laquelle on utilise des connecteurs HE14 coudés pour que la prise femelle parte à l'horizontale limitant ainsi le volume en hauteur de l'ensemble.

➤ Réalisation du petit circuit imprimé de complément.

Décrit dans la [Fiche n°22](#) il ne pose aucun problème particulier. Comme montre la Fig.9 on commence par souder les trois connecteurs HE14 coudés sur le dessus. Puis, montré sur la Fig.10 on soude coté pastilles cuivrées les lignes qui vont du connecteur principal vers celui du répéteur sur lequel se branchera le deuxième petit clavier de complément. Puis, on soude onze fils souples que l'on réunit ensuite avec de la gaine thermo rétractable visible sur [Image 17.JPG](#). Les fils souples sont volontairement long ce qui permettra si nécessaire par la suite de libérer les écrous du circuit imprimé de complément et de l'écarter pour dégager le circuit imprimé du clavier. (C'est ce qui m'a permis de corriger "facilement" les quatre erreurs de câblage détectés lors de la validation



du circuit.) Puis, sur [Image 18.JPG](#) on installe le circuit imprimé de complément sur celui du clavier et en [Image 19.JPG](#) on soude les onze fils du toron en se servant de la [Fiche n°21](#).

➤ La validation du CLAVIER.

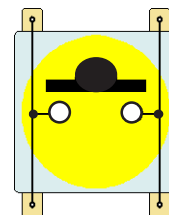
Encore une étape où il faut procéder dans un ordre logique si l'on ne veut pas risquer d'y consommer un temps exagéré. En effet, on va utiliser [P02_Validation_du_CLAVIER.ino](#) pour réaliser cette vérification de conformité, **mais ce ne sera pas possible s'il y a une liaison parasite entre une ligne et une colonne de la matrice du clavier**. En effet, si c'est

le cas le démonstrateur va croire qu'une touche est activée. Il sera alors définitivement bloqué dans la boucle de code qui attend le relâcher du bouton poussoir pour passer à son décodage et à son traitement. **On doit donc commencer par vérifier la non présence de liaisons interdites.**

NOTE IMPORTANTE : Si par aventure l'une des touches était en défaut et présentait un contact interne permanent, il serait particulièrement difficile de le repérer à ce stade des vérifications. Par ailleurs, la Fig.11 montre comment les boutons poussoir ont leurs broches réunies deux par deux. Ces liaisons internes sont utilisées pour prolonger certaines lignes électriques sur le circuit imprimé. Il est donc vital, au stade d'[Image 12.JPG](#) de :

- Vérifier toutes les continuités intérieures "deux à deux",
- Tester les isollements "latéraux",
- S'assurer que tous les B.P. passent correctement en état travail.

Fig.11



Procéder à la vérification des liaisons interdites n'est pas compliqué du tout. Il suffit de disposer d'un tester de continuité. Si comme le mien il fonctionne en sonnette, c'est idéal. Du reste, un tel outil est tellement indispensable, qu'à titre d'exemple dans la [Fiche n°23](#) et la [Fiche n°24](#) je vous propose la description de celui Fig.12 que je me suis fabriqué il y a plus

de quarante printemps. La technique est élémentaire. On relie l'une des pointes de touche sur la broche **C1**. Puis avec l'autre on balaye toutes les autres broches de la droite **C2** à la gauche **L5**. On a testé la non liaison entre **C1** et toutes les autres lignes. Puis on recommence en plaçant la pointe de touche sur **C2** et en balayant de la droite **C3** jusqu'à la gauche **L5**.

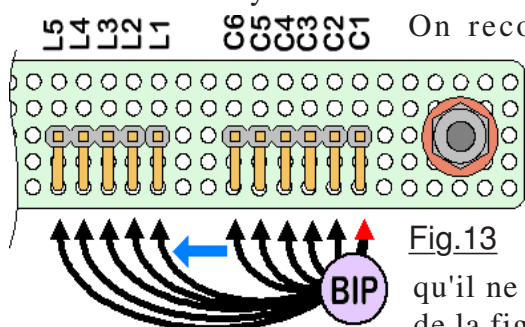


Fig.13

On recommence ensuite ce protocole avec **C3**, **C4** ... jusqu'à **L4**. On aura ainsi testé

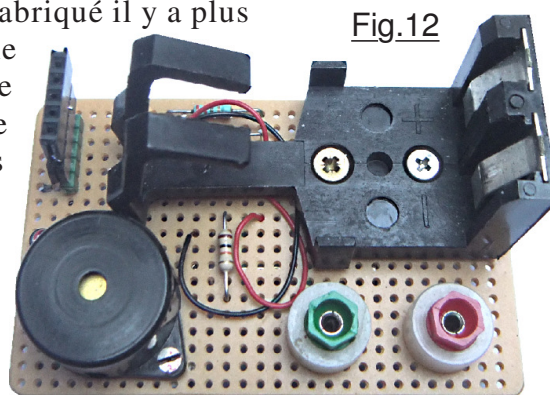


Fig.12

l'intégralité des combinaisons possibles des liaisons interdites potentielles. Avec un multimètre ce serait fastidieux, car pour chaque contact il faudrait regarder le galvanomètre et vérifier

qu'il ne dévie pas. Avec un dispositif de type "sonnette" comme celui de la fig 12 c'est un jeu d'enfant et ultra rapide. Cet outil est tellement utile quand on a fait l'effort d'en réaliser un, que l'on ne peut ensuite plus s'en passer.

Avant de passer à la suite de la validation du CLAVIER il est temps d'emmancher les petits cabochons de couleur comme le montre [Image 21.JPG](#) avec l'initiative de leur répartition. Vous pourrez pour cette phase donner libre court à votre instinct d'artiste. Puis on remplace les quatre vis par des plus longues, on découpe le carton provisoire et non visible sur la Fig.8 on l'installe en le séparant du circuit imprimé par des entretoises ϕ 3mm de 4mm de hauteur. Puis, sur le dessous on ajoute une protection comme sur [Image 22.JPG](#) car les boucles de fil souple ne demandent qu'à accrocher tout ce qui les entoure lors des manipulations. Tester l'ensemble exige d'utiliser le démonstrateur [P02_Validation_du_CLAVIER.ino](#) avec la contribution du **Moniteur** de l'**IDE**. Les onze lignes sont reliées entre le connecteur d'[Image 22.JPG](#) et celui de la [Fiche n°18](#). Puis on relie **C6** et **L2** à un quelconque bouton poussoir d'essai. On reprend le protocole de la [Fiche n°16](#) chaque clic sur l'une des touches doit voir afficher entre crochet la lettre attendue sur le **Moniteur**.

Note : Si ce n'est pas la bonne lettre qui est visualisée, c'est probablement qu'entre les deux modules des lignes ont été croisées par erreur. On les détecte facilement en consultant le schéma de la [Fiche n°13](#). Par contre, si durant la validation précédente on constate des liaisons interdites, il sera bien plus délicat de les retrouver. (*Sur mon circuit j'avais un contact entre C1 et L1 et C1 et L3.*) Pour trouver l'erreur et comparer avec les schémas, sur l'ordinateur on reprend les deux dessins du circuit imprimé, et avec un outil élémentaire

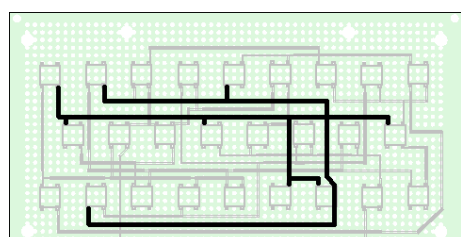
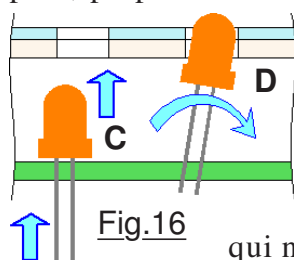
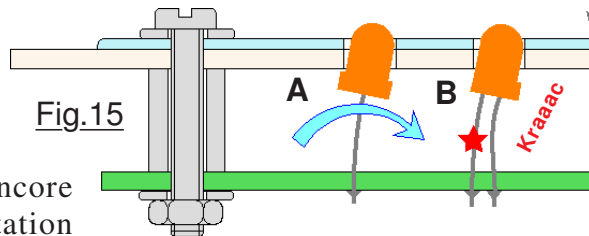


Fig.14

comme **PAINT.exe** par exemple, on surcharge en noir les lignes en défaut. (*Voir la Fig.14*)

06) Développement logiciel.

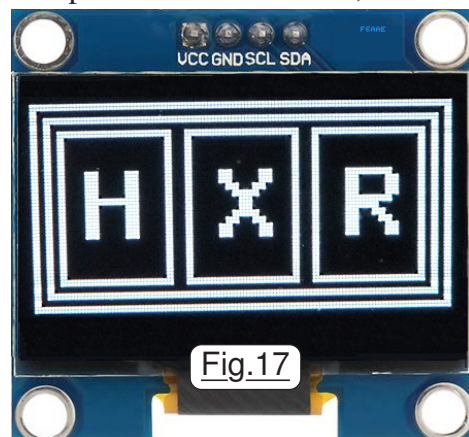
Puisque nous en sommes à la réalisation des circuits imprimés, il serait assez logique de créer le petit clavier secondaire et celui des LEDs oranges. Ce n'est pas le bon moment. En effet, le petit clavier n'est pas encore défini, il vaut mieux attendre que le programme d'exploitation soit globalement développé pour en choisir ensuite la répartition pertinente des touches et des LEDs de servitude. Pour le tableau des LEDs orange, on ne pourra le faire que lorsque le boîtier sera réalisé. La raison résulte de la procédure appliquée pour souder les LEDs. Pour des raisons esthétiques, le trou de passage à travers le panneau du coffret sera réduit au minimum. Elles traverseront "en sifflant". Hors percer tous les trous exactement en face des 26 LEDs est impossible avec des moyens amateurs. S'il est possible en **A** de la Fig.15 de faire fléchir les broches dans leur plan, perpendiculairement comme en **B** c'est infaisable. La technique aisée illustrée sur la Fig.16



qui ne sont pas exactement dans l'axe il suffit à la main de les orienter comme en D. (Sur ces dessins le décalage des trous de passage est exagéré, car même un "bricoleur" occasionnel ne sera pas aussi imprécis que sur ces représentations.) Toutes les LEDs étant correctement positionnées, on soude alors leurs deux broches et le tableau peut être déposé pour y ajouter les résistances, les connecteurs HE14 et les ponts éventuels de liaison. On peut passer au logiciel.

➤ La visualisation des positions des Rotors.

L'idée de réaliser des éléments volumiques animés ayant été abandonnée, on va afficher les lettres de leur orientation sur l'afficheur graphique. Pour que notre réplique puisse s'approcher de l'apparence de la machine historique, on désire que les lettres affichées sur OLED soit les plus grandes possibles. Pour l'apparence on va les présenter dans des rectangles simulant un peu mieux les Rotors. La police de caractère implantée est choisie pour rester parfaitement visible, tout en nous octroyant des lignes de textes contenant assez de caractères. Mais cette police `u8g_font_6x10` même en double taille affiche des lettres un peu petites pour la représentation des Rotors. Aussi, dans un premier temps *on va supposer que nous aurons assez de place en mémoire de programme pour ajouter une deuxième police `u8g_font_9x15B`* qui semble un bon compromis entre la grandeur des lettres présentées en double taille en Fig.17 et un encombrement en octets pas trop déraisonnable. C'est le



démonstrateur `P03_Afficher_les_Rotors.ino` qui est chargé de tester cette approche. Son protocole d'utilisation est précisé en tête de programme. `P03` commence à mettre en place les deux modes d'exploitation de la machine. Il reste maintenant à implémenter toutes les fonctions de CRYPTAGE et celles du mode COMMANDE qui permettra d'initialiser la chiffreuse.

➤ Nouveau bilan sur l'évolution du programme.

Visiblement, sur le thermomètre de la Fig.18, le fait d'avoir ajouté une police de caractère et quelques instructions a fait monter "la température". C'est que la police `u8g_font_9x15B` se gloutonne à elle seule 3002 octets de programme soit 9% de la zone dédiée. Il est fort possible que ce luxe de présentation déborde au final les possibilités de l'ATmega328. Si tel était le cas en fin de développement, il serait toujours envisageable d'abandonner cette présentation et de la reconditionner pour l'exploitation en double taille de la police des menus textuels `u8g_font_6x10`.

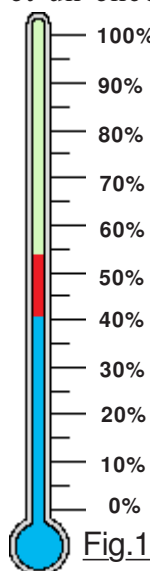
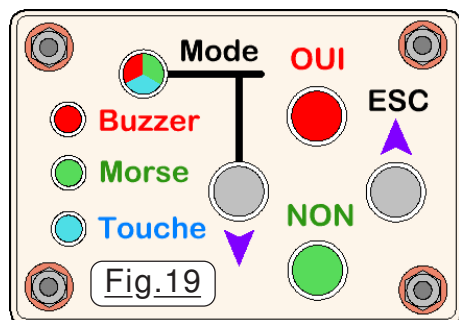


Fig.18

07) Fonctions d'initialisation de la machine.

A vant de pouvoir intégrer les fonctions de cryptage, il est impératif de conditionner le **Brouilleur** et le tableau des **Fiches croisée**. Cette opération initiale que les opérateurs radio devaient effectuer tous les jours avant de pouvoir communiquer va nous obliger à implémenter un **Menu de base** avec du dialogue Homme/Machine qui nous le savons est toujours très gourmand en octets. C'est le démonstrateur **P04_Initialiser_Enigma.ino** qui se charge d'établir les procédures d'initialisation. Sur un RESET le logiciel démarre en mode **CRYPTAGE** et la LED tricolore clignote rapidement en vert incitant à frapper une LETTRE sur le clavier. Pour déclencher une option en mode **COMMANDE**, on clique sur **Mode** puis sur l'une des touches. Si elle n'est pas valide, un BIP d'erreur en informe l'opérateur. À ce stade du développement sont implémentées les fonctions :

- [A] : **AJOUTER** une **Fiche croisée**. (*Même action si touche [X].*) Le nombre maximal de fiches installées sera égal à 10 comme c'était le cas sur la machine d'origine.
- [B] : **BRUTEUR** actif ou silencieux.
- [C] : Affiche à l'écran la correspondance ordonnée entre Lettre et **CHIFFRES**.
Exemples [A ou 1], [B ou 2], [C ou 3], [D ou 4], etc.
- [D] : Imposer une configuration par **DÉFAUT** sur la machine.
- [E] : **EFFACER** toutes les **Fiches croisée**.
- [F] : Restituer les trois orientations initiales des lettres des **Rotors** situées sous la **FENÊTRE**.
- [I] : **INITIALISER** le **Brouilleur** de la machine.
- [K] : Effacer une **Fiche croisée**. (**KILL.**)
- [L] : **LISTER** les **Fiches croisée** actuellement installées.
- [O] : **ORDONNER** les **Fiches croisée** actuellement installées.
- [P] : Afficher l'espace actuellement disponible entre la **PILE** et le TAS. (*Pour les programmeurs.*)
- [V] : **VISUALISER** la configuration actuelle du **Brouilleur**.
- [X] : **AJOUTER** des **Fiches croisées**. (*Même action que [A].*)



➤ Le petit clavier secondaire.

É tablir une interface commode pour assurer le dialogue HOMME/MACHINE passe dans notre application par l'utilisation d'un petit clavier auxiliaire qui dans l'état actuel du schéma de la **Fiche n°13** nous octroie quatre touches. Par ailleurs, on voit sur la **Fiche n°9** qu'outre la LED tricolore, on peut également gérer trois témoins de plus. Actuellement ces trois LEDs seront des

Touche	Code
Mode	1
NON	2
OUI	3
ESC	4

Fig.20

petites de 3mm de diamètre et l'ensemble pourrait ressembler à la présentation de la Fig.19 qui rassemble ces huit éléments, et au tableau de la Fig.20 qui en résume l'organisation logicielle. Pour expérimenter le démonstrateur **P04_Initialiser_Enigma.ino** on va effectuer les nombreuses manipulations proposées jusqu'à la page P8 dans le petit manuel au format A5 nommé **UTILISER ÉNIGMA.pdf** fourni dans le dossier <Documents>.

➤ Un bilan logiciel s'impose.

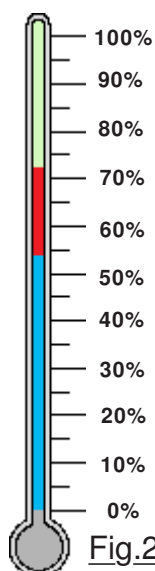


Fig.21

M ettre en place le **Menu de BASE** a fait dramatiquement "monter en température" la place occupée par le programme, ce qui n'a strictement rien d'étonnant, car les procédures effectuent de nombreuses vérifications et surtout les dialogues Homme/Machine sont très goinfres en octets par les nombreux textes affichés sur le petit écran OLED. Nous savons qu'un texte qui est constant est considéré comme une variable et installé directement dans le programme et dupliqué dans la mémoire dynamique. Il est peu probable que les 28% d'espace restant disponibles soient suffisants pour y loger le reste du programme d'exploitation sans compter que le **Brouilleur** virtuel et le code Morse vont ajouter encore des tableaux. Aussi, le moment est venu de faire effectuer au démonstrateur une cure de jouvence, et nous savons (*Ou pas !*) que le moyen le plus efficace pour gagner de la place consiste à faire passer en mémoire non volatile EEPROM un maximum de textes. C'est notre prochaine étape.

07) Passage d'un maximum de textes en mémoire EEPROM.

Pratiquer cette technique conduit à un gain d'espace programme de 574 octets et fait économiser 640 emplacements en mémoire dynamique alors que l'on a ajouté les procédures qui servent à écrire et à lire en mémoire EEPROM. Hors ces dernières sont nécessaires pour sauvegarder et restituer une configuration dans la mémoire non volatile. C'est donc un bénéfice "collatéral" gratuit. Lors de l'écriture du démonstrateur **P05_Textes_en_EEPROM.ino** il reste encore 95 cellules pour y loger du texte supplémentaire si le développement du dialogue Homme/Machine l'exige.

➤ **Passage des textes en mémoire non volatile EEPROM.**

Avec le programme **P00_Textes_en_EEPROM.ino** disponible dans le dossier <OUTILS> on inscrit les textes des écrans "conversationnels", mais également la représentation virtuelle des éléments du **Brouilleur** ainsi qu'une configuration de base qui sera chargée automatiquement lors d'un RESET. Pour celles et ceux qui utiliseraient la mémoire EEPROM pour la première fois la façon de s'y prendre est définie dans le petit fichier d'aide **Débuter.PDF** joint à ce didacticiel. Et pour les débutants qui débutent débutissamment, j'y détaille aussi comment installer l'**IDE**, activer le **Moniteur**, téléverser un programme. Bref, une prise en charge par la main complète et que je souhaite facile à comprendre. Revenons à **P00**. À son démarrage en activant le **Moniteur** à **57600 baud** il inscrit les données dans l'EEPROM puis balaye entièrement cette dernière et en affiche le contenu. En premier il liste les textes et l'on obtient un affichage qui ressemble à celui de la Fig.22 pour lequel les textes sont listés dans l'ordre où ils sont inscrits. La zone mise en évidence en rose à été effacée car elle contenait un mot déjà présent plus loin. Cette zone est donc récupérable et l'on pourra éventuellement y loger une donnée quelconque ou un nouveau texte. Pour la repérer facilement elle a été remplie par dix 'X'. L'affichage du texte se fait dans la police de caractère de la fenêtre du **Moniteur** qui ne connaît pas les caractères accentués. J'ai donc surchargé la copie d'écran par les lettres mises en évidence en vert pastel.

```
COMMANDEDeux ROTORSINDEX xxxxxxxxxxFenê
tre III IV VRotor : Bague : Dé
but : [] [Ré
flecteur : Nb Fiches : de Gauchedu Centre
de Droiteidentiques !Fiche non croisé
e ! LETTRE Déjàà
utilisé
e ! Enlever laFICHE Num Il n'y a pas de
les FICHESDé
brancher toutesConfigurationde BASE
Restituer lestroisINITIALISER laVersion :
machineOrdonner lesEnlever uneAjouter des :
?Fch Nombre : (Max :
03-C 04-D 05-E 06-F07-G 08-H 09-I 10-J
11-K 12-L 13-M 14-N15-O 16-P 17-Q 18-R
19-S 20-T 21-U 22-V23-W 24-X 25-Y 26-Z
[PILE - TAS] octets.Géné
rer uneconfiguration ?ROTOR >
Restituer laConfig. EEPROM
Sauvegarder la
```

Fig.22

NOTE : Quand on place le texte dans le programme, chaque accentué doit être codé séparément ce qui pénalise considérablement le code généré. C'est la raison pour laquelle dans **P04** les textes ne sont pas accentués. Par contre, *en EEPROM un texte accentué se lit simplement car le codage des accentués y est codé directement*. Du coup, les accentués sont "gratuits" raison pour laquelle avec **P05** et les démonstrateurs qui vont suivre les textes en lettres minuscules seront correctement accentués. *C'est un avantage de plus à loger les textes en EEPROM.*

EXEMPLE : `u8g.print("crois"); u8g.print(char(233)); u8g.print('e');`
coûte 18 octets de plus que `u8g.print("croisee");`

➤ **Implantation des données en mémoire non volatile EEPROM.**

Inscrire les éléments du **Brouilleur** se fait exactement à l'identique du codage qui était utilisé dans le programme d'exploitation de la petite machine Énigma élémentaire citée en page de garde de ce document. L'organisation et l'occupation de l'EEPROM est précisée dans la **Fiche n°34** pour les textes et la **Fiche n°35** pour les données. Sur la **Fiche n°35** les zones de codages sont précisées pour la sauvegarde d'une initialisation de la machine ainsi que pour l'organisation des éléments virtuels du **Brouilleur**. Il est évident qu'en sauvegarde d'une configuration de notre réplique d'Énigma le nombre de **Fiches croisées** pourra être quelconque et rester inférieur à 10, voir nul. Une sauvegarde de configuration machine enregistre l'état actuel du système virtuel qui étant issu de *saisies filtrées* sera forcément cohérent et ne pourra pas contenir d'erreur.

➤ Sauvegarder et recharger une configuration.

Démonstrateur spécifiquement agencé pour passer tous les "dialogues" de l'interface d'exploitation en EEPROM, **P05_Textes_en_EEPROM.ino** fait bien plus que se contenter d'utiliser les textes préservés en mémoire non volatile. Par exemple la fonction **[P]** en Fig.23 affiche toujours l'espace disponible entre la **PILE** et le **TAS**, mais ajoute dans la page écran la version du logiciel. Ce n'est qu'un petit détail. Surtout, les deux fonctions invoquées par **[S]** et **[R]** ont été ajoutées au logiciel. Elles sont primordiales puisque la première **[S]** propose à l'opérateur de Sauvegarder la configuration d'initialisation actuelle de la machine, alors que la deuxième **[R]** permet réciproquement de Recharger ces données à convenance. Si vous n'avez jamais sauvegardé de situation, utiliser **[R]** rechargera une configuration de BASE déjà présente inscrite en même temps que les textes. Pour tester ces deux nouvelles fonctions reportez-vous au tutoriel **Linéaire.PDF** ou à son frère imprimé. Les manipulations qui y sont proposées détaillent ces deux fonctions vitales.

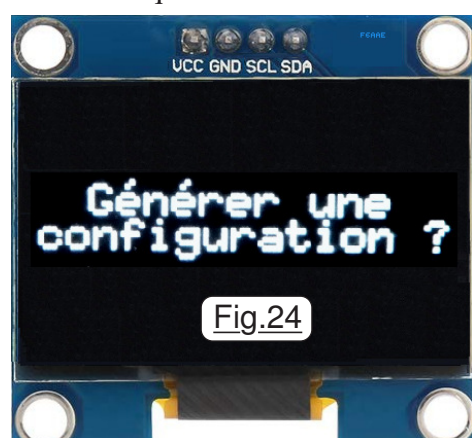


➤ Générer une configuration aléatoire.

Sans se préoccuper de créer un listage aussi complet que celui de la Fig.2 en page P2 du tutoriel **Linéaire.pdf**, il serait agréable de pouvoir automatiquement remplacer la configuration actuelle par une combinaison aléatoire cohérente. La feuille de la Fig.2 est obtenue avec un logiciel gratuit que l'on peut librement télécharger sur :

<https://www.ciphermachinesandcryptology.com/en/codebook.htm>

Il ne fait que simuler des tables secrètes en ne générant que des codes au hasard et les affiche au



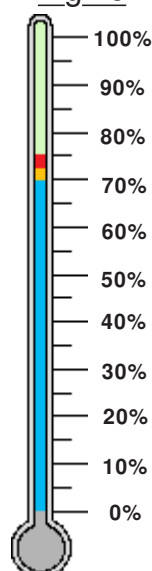
format de la page proposée. Avec les mêmes données initiales vous obtiendrez forcément des codes différents à chaque tentative. La nouvelle fonction activée par **[G]** effectue exactement la même opération sauf qu'elle ne génère qu'une seule configuration avec pour avantage de l'intégrer directement dans la machine. Il suffit de proposer **[G]** en Mode **COMMANDE**. Comme toute commande qui risque de faire perdre des données, il faut en Fig.24 accepter avec la touche **OUI**. Le démonstrateur **P05** génère alors une organisation cohérente, puis passe en

Listage de cette dernière. (Voir la Fig.25) Il n'est pas certain que cette facilité puisse être conservée dans le programme d'exploitation, car cette fonction gloutonne 918 octets soit 3% de l'espace réservé au programme à elle seule et nous n'avons pas encore commencé à introduire les séquences de chiffage.

➤ Un nouveau bilan logiciel s'impose.

Lorsque les textes ont été logés en mémoire non volatile EEPROM avec l'outil **P00** la place occupée par le programme est descendue de 72% à 70%, soit un gain de place assez considérable. Surtout, la place occupée en mémoire dynamique a chuté de 69% à 37% ce qui est considérable. Sur la Fig.26 la zone orange qui occupe 2% de l'espace réservé au programme est occupée par les fonctions indispensables **[S]** et **[R]**. La zone rouge qui occupe 3% de l'espace réservé au programme est consommée par la fonction **[G]**. C'est beaucoup pour une fonction séduisante, mais secondaire. Hors nous n'avons pas encore intégré les fonctions de cryptage. Il est clair que s'il faut faire de la place, la commande **[G]** sera la première sacrifiée.

Fig.26



Rotor : IV II I
Bague : 13 15 14
Début : [M] [W] [V]
Réflecteur : B
Nb Fiches : 10

Fch 01	AG	Fch 02	BN
Fch 03	CW	Fch 04	ER
Fch 05	FJ	Fch 06	HM
Fch 07	KZ	Fch 08	PV
Fch 09	QU	Fch 10	TY

08) Une vermine bien ennuyeuse.

Examinant la documentation relative à l'écran OLED on apprend que son adressage sur la ligne **I2C** est soit 0x78 soit 0x7A exprimée en hexadécimal. Elle est donc différente de celles des deux multiplexeurs PCA9685. En théorie on ne devrait pas avoir de problème. Mais force est de constater qu'entre la théorie et le réel il y a souvent une différence, et un problème durant le développement des démonstrateurs se manifeste assez régulièrement. On peut constater sur les images de la Fig.26 que manifestement il y a interférence. Les décalages latéraux mis en évidence par les lignes jaunes ne se produisent jamais si les deux multiplexeurs PCA9685 sont débranchés. Il y a donc indubitablement une interférence entre les trois modules électroniques.

➤ Vérification des adressages I2C.

Face à un incident tel que celui-ci qui se produit de façon aléatoire, la première action à mener consiste à vérifier que les trois modules sont bien conditionnés matériellement à des adresses **I2C** différentes. Dans ce but, on peut utiliser le logiciel spécifique **P00_Test_ligne_I2C.ino** préservé dans le dossier **<OUTILS>** qui se charge de balayer toutes les adresses possibles sur la ligne **I2C** gérée par **A4** et **A5** (*Bibliothèque Wire.h*) et d'en afficher la liste sur le **Moniteur de l'IDE** initialisé à une vitesse de transfert de 57600 baud dans le programme. Si on débranche la ligne qui va vers les deux multiplexeurs on obtient le résultat de la Fig.27 qui confirme que conformément à la documentation du module OLED l'afficheur est bien à l'adresse **0x3C**. Si l'on recommence l'expérience en branchant les deux multiplexeurs PCA9685 l'affichage devient celui de la Fig.28 qui fait apparaître quatre

Version du 10 mai 2024. Fig.27
Teste toutes les adresses possibles.
=====

```
Peripheral I2C present a l'adresse : 0x3C
1 peripherique(s) present(s).
```

Version du 10 mai 2024. Fig.28
Teste toutes les adresses possibles.
=====

```
Peripheral I2C present a l'adresse : 0x0
Peripheral I2C present a l'adresse : 0x3C
Peripheral I2C present a l'adresse : 0x40
Peripheral I2C present a l'adresse : 0x41
Peripheral I2C present a l'adresse : 0x70
5 peripherique(s) present(s).
```

CONCLUSION : L'interférence constatée n'a rien à voir avec un adressage commun. Des résistances de **10kΩ** ont été ajoutées entre les deux lignes et le **+5Vcc** pour assurer une polarisation fiable ... sans succès. Le problème reste au final un mystère. Il importe donc de contourner cette difficulté en isolant la ligne **I2C** qui va vers les multiplexeurs PCA9685 lorsqu'un affichage est effectué sur OLED. L'agencement logiciel prend la forme de l'organigramme de la Fig.29 qui sera repris pour chaque affichage sur OLED. Cette parade consomme 146 octets pour l'ensemble des dialogues actuels.

➤ Solution matérielle.

Consistant à intercaler un commutateur double entre les deux broches **A4**, **A5** et la ligne qui gère les deux multiplexeurs, la Fig.30 ne représente pas la technologie adoptée. L'afficheur OLED est branché en permanence sur la ligne de dialogue **I2C**. Dès qu'un affichage est invoqué par le programme d'exploitation, la sortie **A0** passe à l'état "1" et active une **INTERFACE** qui fait passer les deux inverseurs à l'état travail. De ce fait la ligne **I2C** ne va plus vers le multiplexeur n°1 qui sera ignoré. Idem pour le multiplexeur n°2 qui est en série avec le premier.



Fig.29

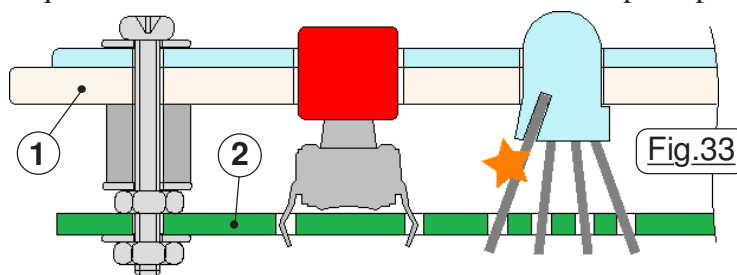
Suite

09) Réalisation du petit clavier de complément.

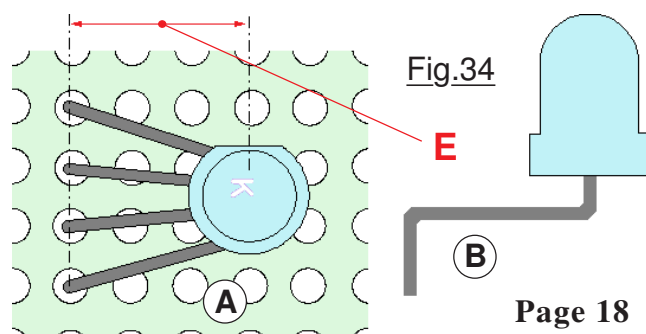
Actuellement il est envisagé pour la réplique matérielle de s'orienter vers une présentation du coffret qui ressemblera à celle de la Fig.32 avec une façade supérieure comportant une trappe qui se soulève coloriée en vert sur le dessin, un peu comme sur la machine historique où il fallait dégager la zone du **Brouilleur** pour insérer le **Réflecteur** et les **Rotors**. Sur notre réalisation on se contentera d'ouvrir la trappe pour accéder au petit clavier de complément. Du coup, ce dernier devra se trouver bien plus bas que l'afficheur graphique, qui lui sera le plus proche possible de la façade pour des raisons de visibilité. La "surface" de notre prototype devrait avoisiner celle d'un format A4.

Comme montré sur **Image 23.JPG** on commence à souder les composants les moins hauts, puis les autres composants sur **Image 24.JPG** sur laquelle on voit que le connecteur HE14 qui supporte l'écran OLED est de type "broches longues" pour surélever l'afficheur. Ce connecteur particulier est parfaitement visible sur **Image 29.JPG**. Du coup le module se trouve en "porte à faux" et il faut le soutenir de l'autre côté. C'est ce que montre **Image 30.JPG** saisie en gros plan.

Sur **Image 25.JPG** et **Image 26.JPG** le circuit terminé est montré par le dessus, alors que sur **Image 27.JPG** il est photographié coté pastilles cuivrées. L'**Image 28.JPG** est particulièrement significative du peu d'écart qui existe entre la plaquette **1** de la façade et le circuit imprimé **2** sur lequel elle est immobilisée. Cet écart est imposé par la faible hauteur des boutons poussoir qui sont soudés sur le C.I. et qui doivent dépasser de la plaquette transparente d'**Image 32.JPG** qui ajoute de l'épaisseur au support de la petite étiquette montré sur **Image 31.JPG**. La Fig.33 présente l'ensemble. La faible distance entre plaquette **1** et C.I. **2** engendre un problème particulier avec la LED tricolore.



Pour que les broches puissent traverser le circuit imprimé, il faudrait les tordre de façon très importante avec le risque de détériorer le composant par forçage exagéré sur l'encapsulation. Aussi, pour éviter ce risque on va ajouter de la souplesse en coudant deux fois les broches comme montré sur la Fig.34 en **B**. Pour que ce témoin lumineux soit à égale distance des trois petites LEDs, le circuit imprimé a été prévu pour un écart **E** entre les pliures de quatre trous de traversée du circuit imprimé au pas d'un dixième de pouce, la vue de dessus étant proposée en **A**. Pour que les huit composants soient parfaitement alignés sur les trous de passage du "porte étiquette", il faut réaliser ce dernier AVANT de procéder à la soudure des éléments. On les insère sur le C.I. puis on installe les deux plaquettes. On retourne l'ensemble. On fait traverser les boutons et les LEDs bien centrés sur les alésages et on procède alors au soudage.



10) Introduction du CRYPTAGE dans le logiciel.

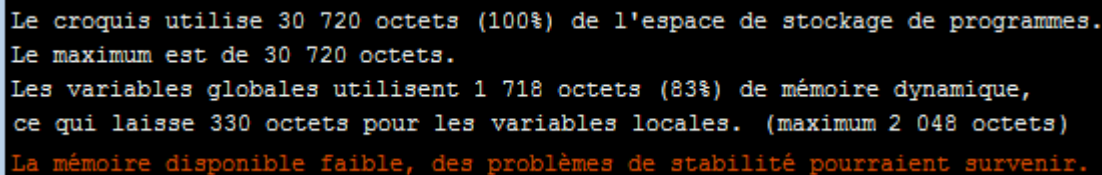
Actuellement on peut effectuer l'initialisation complète de notre réplique d'Enigma. L'écran graphique est stable. Il est temps d'introduire dans cette machine l'intervention des éléments virtuels du tableau des **FICHES croisées** et ceux du **Brouilleur**. C'est le démonstrateur **P07_Cryptage.ino** qui se charge de cette mission. Les routines de traitement étant intégrées au corps du programme, pour les vérifier il faut crypter du texte et vérifier sur une machine de référence fiable que le chiffrement est identique.

➤ **Comparer le comportement de notre réplique celui d'une machine fiable.**

Préambule aux essais du démonstrateur **P07** il nous faut **acquérir et apprendre à se servir d'une Enigma de Référence** qui par la suite sera désignée par **[Enigma]**. Pour ce faire, consulter impérativement les six fiches de **Fiche n°27** à **Fiche n°32**. Faisant usage du contenu de ces dernières, ne passer à la suite de ce didacticiel lorsque vous aurez parfaitement maîtrisé l'initialisation et l'usage de cette **[Enigma]**. Fort de ce savoir-faire on peut s'aventurer dans l'exploration de **P07**. Pour expérimenter ce démonstrateur on va effectuer les nombreuses manipulations proposées de la page P12 à la page P16 dans le petit manuel au format A5 nommé **UTILISER Enigma.pdf** fourni dans le dossier **<Documents>**.

➤ **Le programme d'exploitation ultime.**

À partir du moment où l'on peut entièrement initialiser la machine et l'utiliser en Chiffrement / Déchiffrement, tant qu'il reste de la place disponible dans la zone réservée au programme et celle des données dynamiques, on peut ajouter des fonctions et des options nouvelles qui améliorent la qualité opérationnelle de notre réplique. On ne va pas s'en priver, c'est le programme ultime **P08_EXPLOITER_Enigma.ino** qui apporte ces améliorations, certaines relevant du luxe, d'autres s'avérant presque indispensables. Si vous avez consulté d'autres didacticiels en ligne dont je suis l'auteur, vous savez que j'éprouve un plaisir non dissimulé à rentabiliser au maximum l'utilisation de ce merveilleux ATmega328. Tant qu'il reste de la place disponible, "j'en rajoute une couche". Et bien pour **P08** il sera difficile de faire mieux en termes d'occupation des zones disponibles.



```
Le croquis utilise 30 720 octets (100%) de l'espace de stockage de programmes.
Le maximum est de 30 720 octets.
Les variables globales utilisent 1 718 octets (83%) de mémoire dynamique,
ce qui laisse 330 octets pour les variables locales. (maximum 2 048 octets)
La mémoire disponible faible, des problèmes de stabilité pourraient survenir.
```

Fig.35

Un simple regard sur la copie d'écran de la Fig.35 nous apprend que l'on a consommé l'intégralité des cellules réservées au programme. Il ne reste plus un seul octet non utilisé ! Pour ce qui est de la rentabilité, c'est le summum, mais si on découvre un aléas dans le programme, il n'y a aucune possibilité d'ajouter du code pour corriger, il faudra faire de la place. (*Par exemple en diminuant les écrans du "HELP".*) Si vous consultez la **Fiche n°34** et la **Fiche n°35** vous constaterez que l'EEPROM est aussi saturée, il ne reste plus une seule cellule disponible.

➤ **La version du compilateur utilisé.**

Choisir la version du compilateur C++ qui nous sert à créer le programme objet et à le téléverser dans Arduino n'est pas sans conséquence. Pour des raisons personnelles, je commence toujours à développer avec la version **1.7.9** qui a créé tous les démonstrateurs ainsi que **P08**. Du coup les informations relatives à la taille que présente le programme au cours de son évolution ne sont vraies que si l'on compile avec **1.7.9**, car il faut savoir que les versions ultérieures ont été optimisées, et elles produisent un code binaire qui prend bien moins de place. Par exemple :

	Compilateur 1.7.9	30720 Octets (100%)	1718 Octets (83%)	Max : 30720.
	Compilateur 1.8.0	28560 Octets (92%)	1722 Octets (84%)	Max : 30720.

Conclusion : Si l'on compile avec **1.8.0** on dispose immédiatement de 2160 octets de libres pour améliorer le programme. Difficile de trouver encore de nouvelles options, mais on pourrait améliorer les affichages en respectant les accentués, mieux décrire la liste commandes etc.

Alors pourquoi ne pas profiter de cette manne ?

Rien à voir avec une sorte de lassitude qui inciterait à en rester là. Au contraire, j'adore programmer et j'aurais été enchanté de poursuivre le développement logiciel en **1.8.0** jusqu'à y saturer les 30720 cellules de SDRAM. Mais pour une raison que je n'ai pas réussi à déterminer, compilé avec cette mouture de l'**IDE** le programme sur certaines fonctions ne se comporte plus correctement. Alors je préfère me contenter d'un **P08** en version **1.7.9** qui est stable et dont je peux garantir le bon fonctionnement et ne pas courir le risque de vous proposer "une planche savonnée".

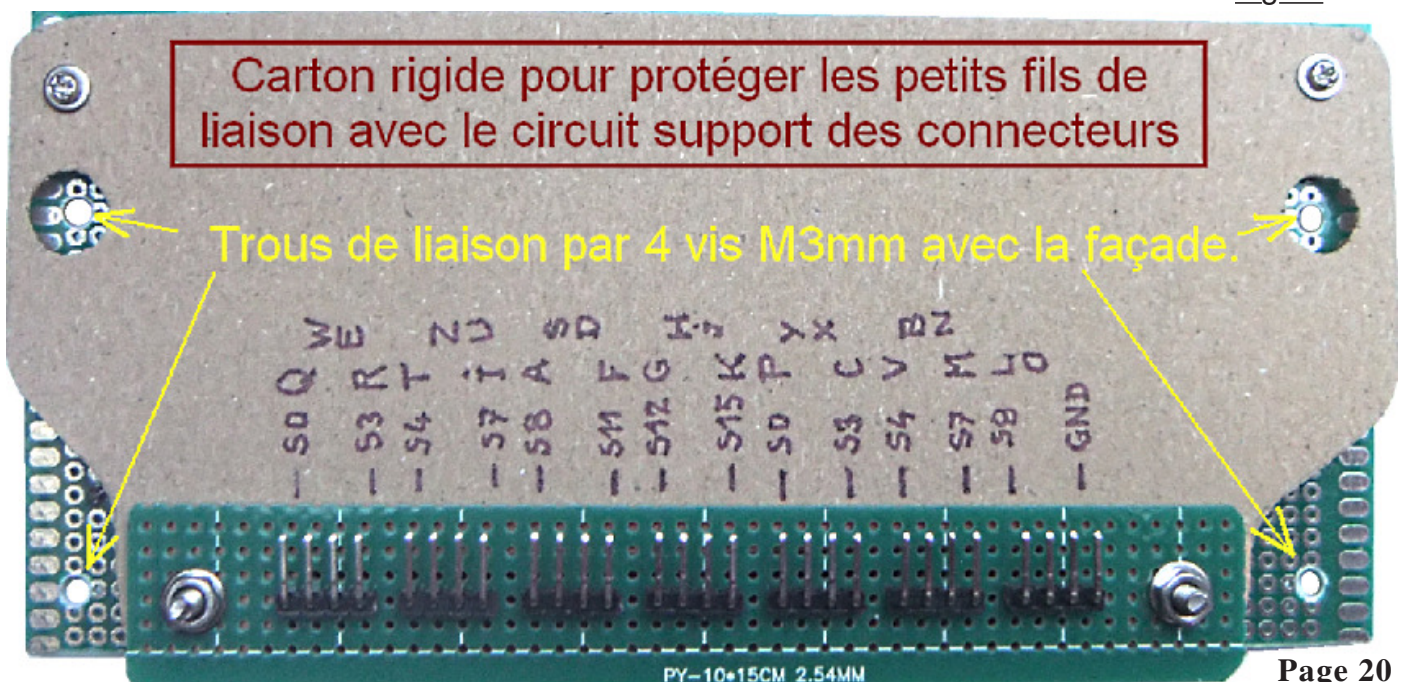
➤ Un avertissement "de couverture".

Observant la Fig.35 on constate que le compilateur affiche en orange une alerte pour le moins inquiétante. Rassurez-vous, les développeurs du compilateur "ouvrent le parapluie" pour nous mettre en garde d'un danger potentiel. Mais dans notre cas il n'y a rien à craindre. Du reste ce n'est pas la première fois que je sature le sketch et l'EEPROM, et mes programmes ont toujours fonctionné parfaitement. *(Mis à part parfois lors d'un bug découvert tardivement mais corrigé sans conséquence sur la stabilité du programme.)* En réalité, cette précaution je pense doit être prise dans la mesure où la collision entre la **PILE** et le **TAS** n'est pas vérifiée par le programmeur. Hors ce phénomène n'est pas directement lié à la taille du programme, mais à sa boulimie en mémoire dynamique. Pour notre cas il reste 322 octets entre la **PILE** et le **TAS**. C'est largement suffisant pour que **P08** soit stable. Hors un programme bien plus petit en taille pourrait diverger. Il suffirait qu'il utilise des procédures récursives avec une plongée en grand nombre dans des appels sans retour empilant de surcroît des données locales. En résumé, les concepteurs du compilateur sont prudents car ils savent que ce problème sournois peut se produire pour tout logiciel, et ils nous en avertissent à leur façon. *(Voir à ce sujet le chapitre 24 en page 66 du didacticiel mentionné en page 1.)*

➤ Utiliser le programme ultime d'exploitation.

C'est dans le tutoriel de [Linéaire.pdf](#) ou dans le livret [UTILISER Énigma.pdf](#) que sont expérimentées les dernières fonctions et options ajoutées à **P08** de la page P17 à la page P24 qui terminent votre formation d'opérateur télégraphique. Toutefois, l'option de mesure de la tension du **+5Vcc** de la carte Arduino ne sera crédible que si l'on complète le circuit imprimé qui supporte OLED et le petit clavier par les composants du schéma de la [Fiche n°25](#). Ces éléments ont été facilement ajoutés car un peu de place était prévue "pour le cas où" sur la plaquette cuivrée. Le dessin du circuit imprimé donné en [Fiche n°17](#) et [Fiche n°18](#) a été également mis à jour. Cet ajout est visible avec zoom sur [Image 34.JPG](#) sur laquelle on voit que la résistance de **4,7kΩ** passe sous la tige du condensateur de 470μF. Sur [Image 35.JPG](#) nous avons une vue d'ensemble du circuit imprimé ainsi modifié et sur [Image 36.JPG](#) la vue coté cuivre de ce petit ajout.

Fig.36



➤ **Étude et réalisation du dernier circuit imprimé, celui des 26 LEDs oranges.**

U ltime étude électronique, pour des raisons d'optimisation de l'encombrement de notre réplique d'Énigma, *le circuit imprimé du tableau des ampoules simulées aura la même largeur que celui du clavier principal* et taillé dans le même lots de circuits de prototypage.

Comme pour son frère aîné du clavier, il sera complété par une petite plaquette située sur le dessous servant de support aux connecteurs coudés HE14 de liaison avec les multiplexeurs. Ce circuit de complément est parfaitement visible sur [Image 37.JPG](#) ou sur la Fig.36 qui montrent que les 27 petits fils électriques de liaison entre les deux éléments sont protégés par un petit carton rigide. Ce carton est isolé sur [Image 38.JPG](#) qui en définit très bien la forme et les contours. Les dessins de ces derniers circuits imprimés sont proposés dans les [Fiche n°40](#) et [Fiche n°41](#). Comme à chaque fois on commence par les composants de faible hauteur, montré en [Image 39.JPG](#).

Mais ici s'arrête provisoirement le soudage, car avant de souder les fils de liaison avec le petit circuit de complément il faut assembler les LEDS. Il n'est pas question de procéder comme avec le clavier, c'est à dire tout terminer et réaliser ensuite la façade. On va commencer par réaliser le dessus du coffret qui constitue la façade. Ainsi il sera facile de placer toutes les LEDs bien centrées sur les alésages avant de les souder. La Fig.38 présente le résultat final avec pour bénéfice une facilité pour déposer et surtout réintroduire les 26 LEDs dans la façade. Sur [Image 40.JPG](#) on peut observer la façade entièrement terminée. (*Galerie d'images : 4*) C.I. des 26 ampoules) Pour vous aider dans la réalisation du coffret, vous trouverez dans la dossier <DESSINS> les planches *avec la cotation précise* des éléments composant le coffret. Les divers dessins sont classés dans des sous dossiers. C'est dans le sous dossier <01 Façade du coffret> que sont réunies les informations relatives à la façade du coffret. Sur [Image 41.JPG](#) on peut voir la "vitre" qui recouvre l'étiquette d'[Image 42.JPG](#) à imprimer. C'est dans le dossier <Documents> que l'on trouve le fichier *Étiquettes à imprimer.PDF*. Notons au passage que les trois lettres colorisées en vert concernent les options du code Morse. [Image 43.JPG](#) et [Image 44.JPG](#) présentent le circuit imprimé achevé. Il reste alors à réunir les fils du petit C.I. de complément d'[Image 45.JPG](#) et d'[Image 46.JPG](#) ce que montrent [Image 47.JPG](#) et [Image 48.JPG](#). Puis on assemble les deux étiquettes avec leurs "vitres" ainsi que les deux circuits imprimés pour aboutir à la façade terminée d'[Image 49.JPG](#) et de l'autre côté sur [Image 50.JPG](#).

C.I. des LEDs oranges terminé.

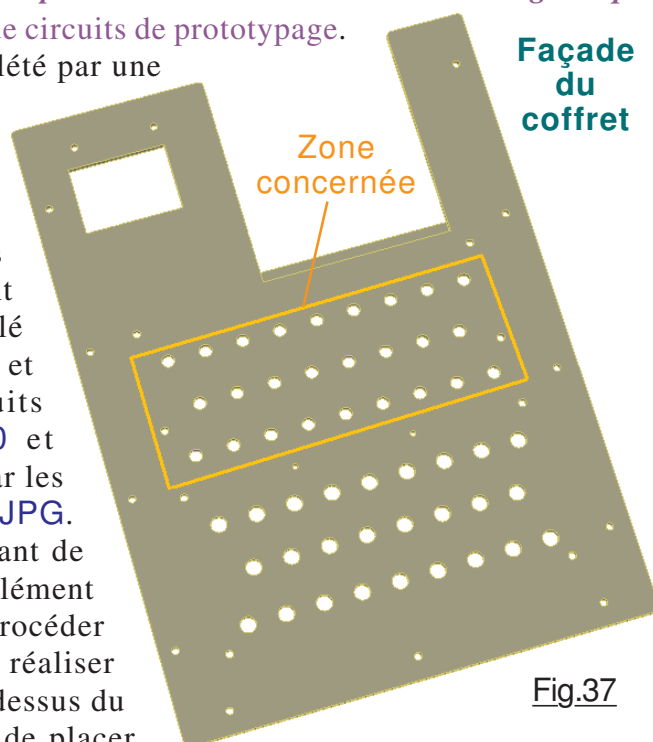


Fig.37

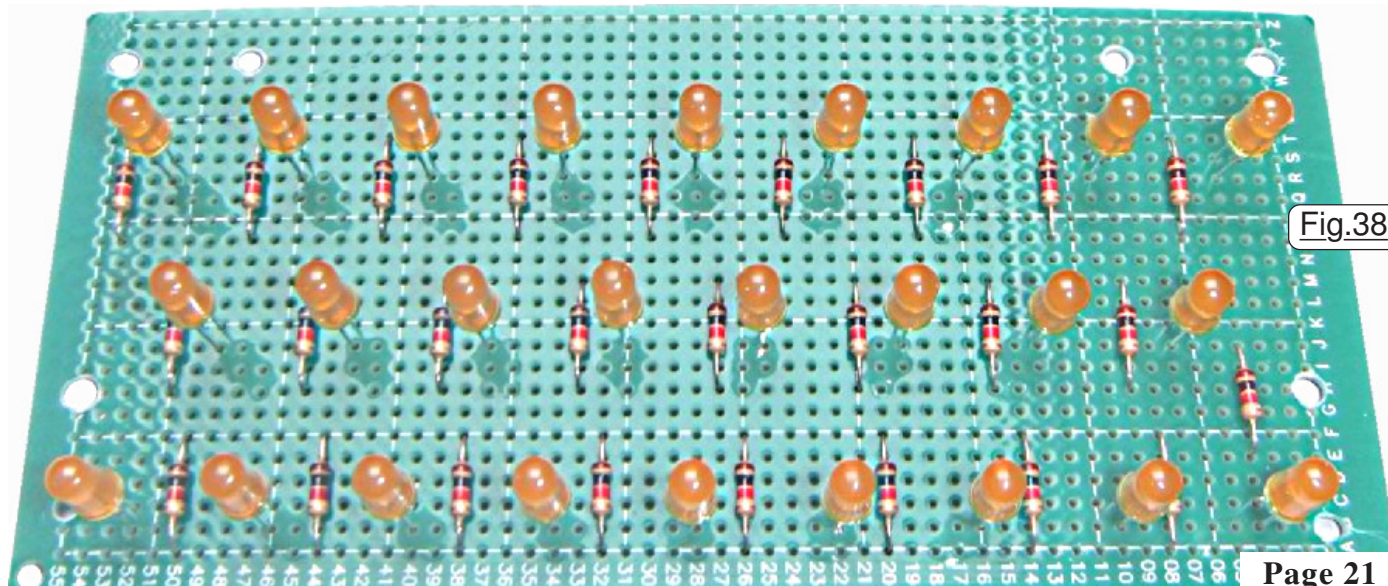


Fig.38

11) Réalisation du coffret.

C'est dans le document [Réalisation des coffrets.pdf](#) que sont dévoilées mes techniques pour créer des boîtiers personnels, avec pour avantage de leur conférer les dimensions et les proportions correspondant exactement à nos désirs. La Fig.39 montre la maquette théorique de ce que l'on désire obtenir. C'est dans le dossier <02 Coffret d'Énigma> que sont préservés *les dessins avec cotation* des divers éléments de cet ensemble. Commencer par observer attentivement le ! [Modèle informatique.JPG](#) qui se trouve dans le répertoire <Galerie d'images> et confiné dans le dossier <5) Réalisation du coffret>. Il ne reste plus qu'à concrétiser cette merveille virtuelle avec un maximum de précision. Passons en revue quelques détails d'agencement dont les photographies réalisées durant la réalisation sont dans le dossier <5) Réalisation du coffret> du répertoire dédié <Galerie d'images>. On commence par créer la **Semelle** dont [Image 51.JPG](#) permet de situer l'emplacement des divers circuits imprimés.

Pour la position exacte des divers trous de traversées des vis de liaison se reporter aux documents [Dessin 15.jpg](#)

à [Dessin 17.jpg](#) puis on crée les divers

flancs dont [Image 52.JPG](#) rappelle

la technique pour usiner les

lumières rectangulaires et

[Image 53.JPG](#) présente

trois des flancs latéraux

terminés. Ensuite on soude

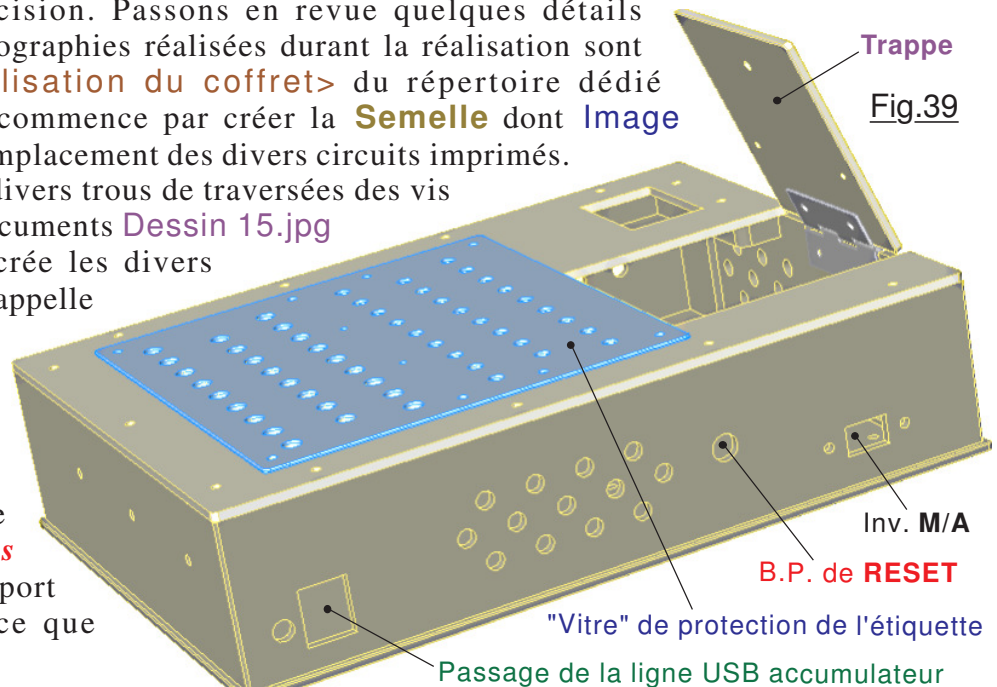
avec le diluant cellulosique

les quatre flancs latéraux *sans*

se tromper de côté par rapport

aux trous de la semelle, ce que

montre [Image 54.JPG](#).

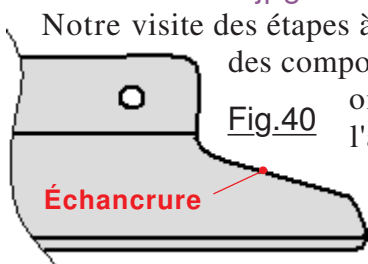


NOTE : Les photographies ont été réalisées au fur et à mesure de la réalisation. À ce stade la position des trous de fixation de la bride qui immobilise l'accumulateur n'étaient pas déterminée. Ils sont percés plus tard lorsque l'équerre de bridage est disponible et pointés avec précision par "contre perçage".

Puis, tout le tour on assemble les renforts montrés sur [Image 55.JPG](#) dont le [Dessin 06.jpg](#) précise la cotation. On soude également les bandes qui sont dessous la plaque qui rehausse l'accumulateur par rapport à la **Semelle**. Ainsi positionné la fenêtre de [Passage de la ligne USB accumulateur](#) est à 6mm de la **Semelle**. On positionne alors par dessus la plaque sur laquelle est bridé l'accumulateur ce que montrent [Image 56.JPG](#) et [Image 57.JPG](#). On ajoute également la Cale d'épaisseur qui positionne un peu décalé à gauche le gros accumulateur. (*Voir à ce sujet le chapitre spécifique 13) L'accumulateur rechargeable.*) Cette phase de la réalisation est documentée dans [Dessin 07.jpg](#) à [Dessin 09.jpg](#). On observe sur [Image 57.JPG](#) que les renforts d'angle sont en place et servent à positionner en hauteur les raidisseurs qui tout le tour intègrent les écrous prisonniers de liaison avec la façade supérieure. La photographie d'[Image 58.JPG](#) complète le document [Réalisation des coffrets.pdf](#) alors qu'[Image 59.JPG](#) à [Image 61.JPG](#) présentent le résultat obtenu. C'est sur [Dessin 10.jpg](#) que sont indiquées les positions des écrous inclus dans ces renforts, alors que [Dessin 09.jpg](#) précise la largeur de 12mm des trois bandes collées les unes sur les autres.

Enfin, avec [Image 62.JPG](#) on aborde l'articulation entre le coffret et la petite trappe de servitude, le positionnement de l'échancrure et des deux trous de passage des vis ϕ M3 étant sur le [Dessin 18.jpg](#) alors que [Dessin 19.jpg](#) indique tous les détails de la trappe proprement dite.

Notre visite des étapes à conduire pour réaliser le coffret s'achève par l'observation de l'implantation des composants d'[Image 67.JPG](#) à [Image 70.JPG](#). En particulier sur [Image 68.JPG](#) on peut constater qu'un carton rigide est placé sous la bride qui immobilise l'accumulateur pour assurer "une certaine souplesse". Cette dernière présente une échancrure ([Fig.40](#)) qui n'est pas fonctionnelle. Elle était sur le profilé de récupération en duralumin utilisé et la languette inutile n'a pas été enlevée ... par un excès de paresse !

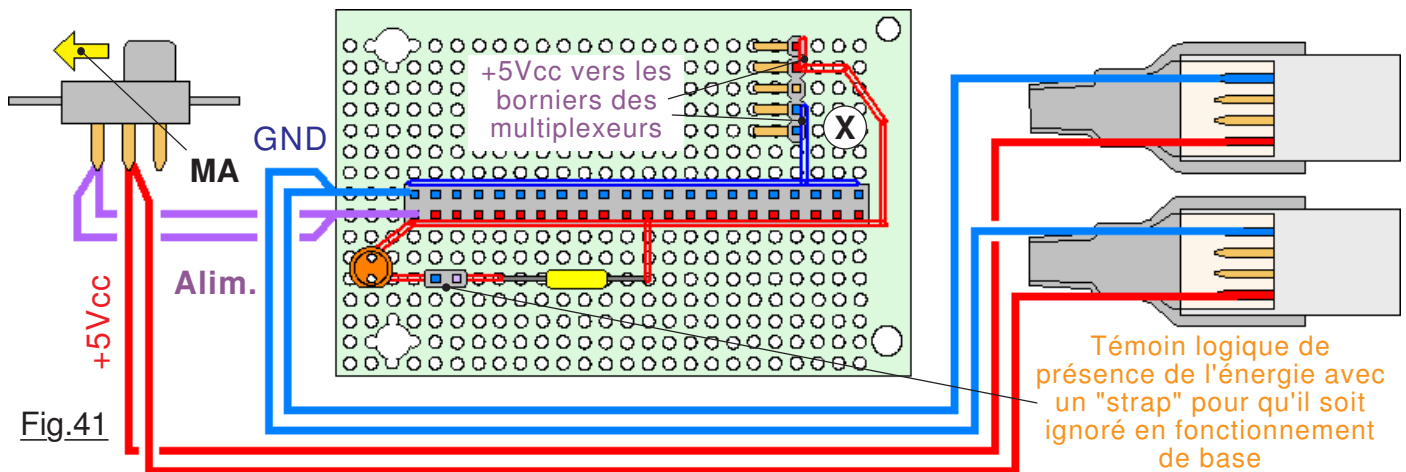


12) Câblage des divers éléments de l'électronique.

Opération cruciale, il faut procéder avec méthode et vérifier la conformité chaque fois que c'est possible. Comme les tests de validation imposent d'alimenter les circuits, le plus simple reste encore de commencer par intégrer l'accumulateur. C'est d'autant plus conseillé que l'une des petites équerres latérales qui le positionne en largeur et visibles sur [Image 68.JPG](#) est immobilisée sous le circuit imprimé de la carte Arduino. (Ce C.I. est nommé U.C. pour **U**nité **C**entrale dans les *Fiches techniques.*)

➤ La "nourrice" de répartition des alimentations.

Habitué à des réalisations "plus cossues", j'ai étudié un circuit imprimé représenté coté composants sur la [Fiche n°46](#). On constate qu'il est équipé d'une "rampe" de 22 positions HE14 doubles. Même si on n'utilisait qu'un couple de picots sur deux, on pourrait distribuer 11 lignes c'est à dire déjà plus que le nombre de circuits imprimés présents. Je me suis un peu "précipité" dans l'étude de ce circuit car il était indispensable pour procéder au câblage. Par ailleurs, initialement le connecteur HE14 à cinq broches devait recevoir les deux lignes de puissance arrivant de l'accumulateur. Ainsi ce C.I. pouvait être entièrement débranché de l'ensemble. Et puis je me suis fourvoyé. J'ai soudé les lignes d'arrivées directement sur le dessous coté pistes ! Du coup il y a encore moins de lignes pour répartir l'énergie vers les divers modules. C'est d'autant plus vrai que la puissance vers les multiplexeurs est branchée sur le connecteur à cinq broches, et que **GND** et **+5Vcc** sont "ventilés" directement entre les divers circuits imprimés. Bref, ma "nourrice" n'alimente que très peu de circuits imprimés. Vous pouvez-donc concevoir un circuit bien plus simple.



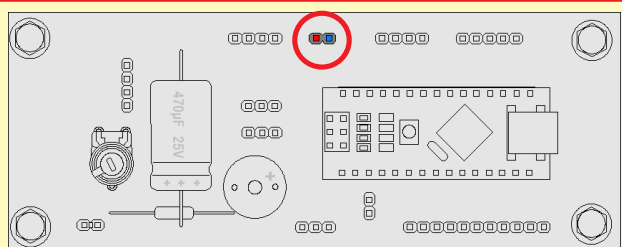
L'ordre des opérations pour le câblage est donné sur la [Fiche n°46](#) et sur la [Fiche n°47](#). Pour l'étape 01) il faut rester très attentif à ne pas engendrer un court-circuit sur l'accumulateur. On commence par souder des fils de sections suffisantes sur les deux fiches USB mâles qui seront insérées dans les deux sorties de l'accumulateur : Un fil bleu sur **GND** et un fil rouge sur **+5Vcc** pour chaque fiche. On torsade deux à deux ces lignes. On insère les deux fiches USB dans l'accumulateur et on bride ce dernier dans le coffret. On soude les deux fils rouges sur l'une des broches de l'inverseur Marche/Arrêt. On soude l'autre broche de l'inverseur sur deux fils torsadés rouges qui vont sur la rampe **+5Vcc** de la nourrice. **On place immédiatement l'inverseur sur Arrêt.** (Voir la Fig.41) On soude alors la ligne torsadée bleue qui arrive des deux fiches USB sur la rampe HE14 coté **GND**.

NOTE IMPORTANTE : À partir de ce stade, l'accumulateur est relié à la nourrice en permanence. Durant les opérations de soudage des incidents sont potentiellement possibles, voir probables. Aussi, il ne faudra alimenter que durant les vérifications de validation ponctuels puis replacer l'ensemble hors tension pour la poursuite des travaux. Par ailleurs on peut remarquer sur [Image 69.JPG](#) que les broches des fiches USB sont à découvert. La chute d'un ustensile métallique durant le soudage n'est pas à exclure. En vertu des principes de Murphy, cet outil va inexorablement provoquer un court-circuit franc. Aussi un morceau de papier plastifié est inséré et reste en place élastiquement pour parer ce type d'incident. Cette protection est évoquée et bien visible sur [Image 93.JPG](#) où l'on peut remarquer qu'elle est placée verticalement, puis fléchiée à l'horizontale pour être bloquée sous la bride dans la zone où il n'y a pas le carton marron.

➤ Câblage entre les autres modules.

Chaque fois que ce sera envisageable, des nouvelles interconnexions seront validées avec les précautions énoncées dans l'encadré de la Page 23. Conformément à la liste de la [Fiche n° 46](#) en item 02) on établit les deux lignes qui vont du HE14 en X de la Fig.41 vers les borniers verts des deux multiplexeurs PCA9685. Puis on établit la liaison entre le bouton poussoir du RESET et le HE14 de l'Unité Centrale. On relie ensuite les deux multiplexeurs PCA9685 entre eux. Enfin, on réalise une ligne provisoire "directe" entre l'U.C. et l'entrée du multiplexeur n°1 comme montré sur [Image 84.JPG](#). Puis, conformément à l'item 06) relier l'U.C. à l'afficheur OLED.

ATTENTION : Sur la [Fiche n° 46](#) j'ai oublié d'indiquer qu'il faut ponter une sortie alimentation de la nourrice avec le connecteur HE14 à deux broches du circuit U.C.



À ce stade mettre sous tension : L'ensemble doit engendrer un affichage correct du "compteur" sur le [Brouilleur](#) virtuel. On réalise alors l'araignée dont le schéma électrique est donné en [Fiche n° 44](#) et la photographie sur [Image 85.JPG](#). On réunit alors les deux connecteurs HE14 sur l'U.C. et les deux connecteurs HE14 sur le C.I. de l'afficheur et du petit relais comme le montre [Image 86.JPG](#). (La ligne de liaison avec le multiplexeur de cette image n'est pas encore en place.) Sur une mise sous tension OLED doit afficher le "compteur" et le petit relais doit "cliquer". Passer à l'item 08) et procéder au test de validation. C'est grâce à la "ligne directe provisoire" branchée lors de l'item 05) que les multiplexeurs fonctionnent sans passer par le relais d'isolement.

NOTES : Mises à part les lignes d'alimentation entre l'accumulateur et la nourrice, toutes les autres liaisons sont réalisées avec du fil de petite section. Ces fils sont issus de limandes séparables pour ordinateurs. On peut observer que les fils de chaque ligne sont torsadés pour former des torons. Cette pratique a pour but de créer des liaisons compactes qui ne s'emmêlent pas, tout en conservant un maximum de souplesse.

Parfaitement visible sur [Image 85.JPG](#) par exemple, les cosses des fiches HE14 sont isolées sur leurs soudures par des petits manchons en gaine thermo-rétractable. Cette technique améliore considérablement leur résistance mécanique et les protègent lors des manipulations.

Les lignes qui vont des multiplexeurs vers le clavier et vers les LEDs de servitude n'ont pas de gaine thermo-rétractable coté multiplexeurs pour leur laisser un maximum de souplesse et pouvoir les courber relativement court, car elles sont sous le circuit imprimé de l'afficheur et du petit clavier de servitude, et il n'y a pas trop de dégagement en hauteur.

À partir de l'item 09) on commence à établir les lignes qui vont du coffret vers les deux modules de la façade de fermeture d'[Image 87.JPG](#). Pour œuvrer confortablement il faut commencer par immobiliser verticalement la façade sur le coffret comme montré sur les deux photographies d'[Image 88.JPG](#) et d'[Image 89.JPG](#). Il faudra réaliser des torons de longueur courte tout en laissant assez de latitude de mouvement lors de la dépose de cette façade en vue de la maintenance. On doit pouvoir la placer avec toutes les liaisons à la verticale coté flanc latéral droit. (Voir le [Dessin 00.jpg](#) précisant la répartition des éléments du coffret.) Pour l'item 09) les fils sont "croisés" et il importe de bien respecter le schéma de la [Fiche n° 45](#). Étape 10) terminée, on remet sous tension et on procède à la validation du clavier principal *ainsi que celle du petit clavier de servitude*. L'achèvement des opérations de soudage consiste à déposer le C.I. de l'afficheur après avoir libéré tous les HE14. Ensuite on ajoute tous les torons qui vont des deux multiplexeurs PCA9685 vers les connecteurs du tableau des LEDs oranges. Les deux étapes 16) et 17) terminent la phase de câblage et valident entièrement le bon fonctionnement de l'ensemble. Logiquement lors du rétablissement d'une ligne on ne risque pas d'en inverser l'orientation des fiches car l'étiquette de repérage doit être visible. Le coffret est terminé. On immobilise la façade avec ses treize vis ϕ M3 et on fête notre réussite. Il reste maintenant à offrir un couvercle de fermeture à notre Énigma.

13) L'accumulateur rechargeable.

C'est ce dernier qui procure à notre réplique d'Énigma une autonomie confortable. C'est un module de réserve d'énergie de forte capacité annoncé pour 26800mAH au lithium-polymère. Ces dimensions sont non négligeables 18,5mm x 75mm x 150,5mm. Je dois toutefois préciser que cette référence HX160Y6 montrée en Fig.42 est d'une capacité franchement exagérée vu la consommation de 40mA en fonctionnement banal de notre réplique. Ce luxe reste toutefois acceptable parce-que le dispositif était disponible et non utilisé suite à un "changement local d'activité". Je vous suggère de remplacer cet encombrant élément par un dispositif plus adapté, **en veillant qu'il ne soit pas plus haut ou plus large** pour ne pas déborder du volume enveloppe de la réserve d'énergie actuelle.

NOTE : Cette consommation passe à 90mA si avec la commande [Y] on allume simultanément les 26 LEDs oranges. Si vous optez pour un élément plus "raisonnable", il ne faudra pas abuser de cette commande et laisser ainsi l'éclairage de Versailles en permanence.

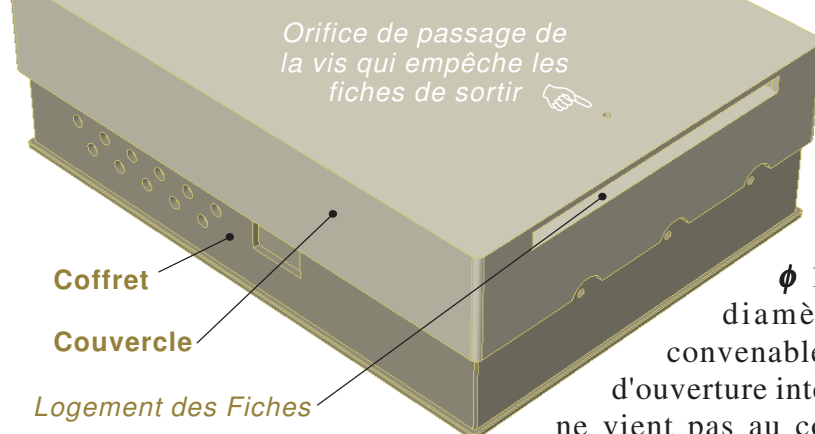
Le HX160Y6 dispose de deux sorties USB indépendantes qui ont été mises en parallèle pour en consommer l'intégralité de l'énergie disponible, câblage bien visible sur [Image 69.JPG](#).

14) Munir notre Énigma d'un couvercle de fermeture.

Envisager la protection de la chiffreuse par un couvercle de fermeture est d'autant plus justifié que c'était le cas pour la machine historique. Nous allons profiter de l'ajout de ce complément pour s'en servir à ranger toutes les Fiches techniques, et prévoir un petit godet de rangement de la ligne USB qui permet la recharge de l'accumulateur avec un bloc secteur. Notre réplique sera ainsi totalement autonome. Comble de luxe, on va percer des trous ϕ 2mm sur la face du dessus pour

"graver" le LOGO de la crypteuse montré sur [Image 75.JPG](#) sur laquelle le couvercle est terminé. Les planches de [Dessin 20.jpg](#)

à [Dessin 27.JPG](#) proposent pas à pas et dans l'ordre chronologique les étapes d'agencement de ce gros sous-ensemble. L'[Image 76.JPG](#) présente l'intérieur avec bien visible le godet de rangement des Fiches techniques. Cette



photographie d'[Image 76.JPG](#) montre comment les Fiches prendraient leur liberté si durant les manipulations du couvercle le petit boulon ϕ M3 en nylon (*Pour des raisons esthétiques.*) n'était pas en place. On ne retire ce boulon que pour extraire et consulter la documentation.

➤ Réalisation du LOGO en "filigrane".

Strictement NON fonctionnel, ce n'est qu'un élément pour mettre en valeur le couvercle. Vous pouvez librement vous en passer, surtout si vous n'avez pas à votre disposition une perceuse à colonne d'électronicien du type de celle montrée sur [Image 80.JPG](#). Si cette option vous séduit, il faut commencer par imprimer le LOGO donné à bonne échelle dans le document [LOGO Enigma.pdf](#) disponible dans le dossier <Documents>. Conformément à ce que montrent les deux **Page 25**

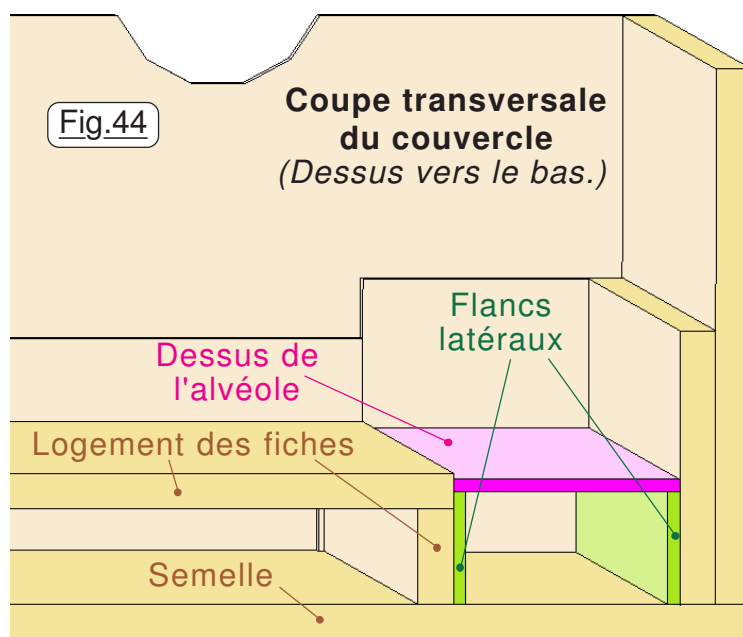


Fig.42

photographies [Image 78.JPG](#) et [Image 79.JPG](#), il faut repérer la position des trous avec un écart entre eux d'environ 5mm. Puis le grand dessin est découpé "à sa dimension" et immobilisé sur le couvercle avec du ruban adhésif à peine visible sur [Image 80.JPG](#). Il est évident que si vous optez pour cette option, il faudra la traiter lorsque la plaque est plate, c'est à dire avant d'y coller les nombreux divers autres éléments.

➤ **L'alvéole de rangement du cordon de rechargement de l'accumulateur.**

A ctuellement une loi européenne impose une normalisation de tous les cordons de type USB quel que soit leur dispositif cible. Toutefois, pour des composants anciens ce n'est pas forcément le cas. Aussi, si la ligne utilisée est spécifique, il est de loin plus agréable de l'avoir "à bord" au lieu de la chercher vainement car on a oublié où elle a été rangée la dernière fois de son utilisation. S'il est pertinent de prévoir une zone de rangement dans le couvercle, pour le chargeur USB ce n'est pas possible compte tenu du volume d'un tel bloc secteur. Ce n'est pas du tout pénalisant car tous les modules secteurs de ce type sont globalement identiques et il en "traîne dans tous les tiroirs". La Fig.44 donne une coupe transversale de la zone dans laquelle est agencée l'alvéole de rangement du cordon USB. Constituées de trois bandes de 1mm d'épaisseur, ses deux **Flancs latéraux** imposent la hauteur à laquelle est collé le dessus de ce rangement de longueur 190mm coté sur [Dessin 28.jpg](#). Les deux vues [Image 102.JPG](#) et [Image 103.JPG](#) montrent la concrétisation des deux **Flancs latéraux**. Sur [Image 104.JPG](#) est visible le cordon à ranger et sur [Image 105.JPG](#) ainsi que sur [Image 106.JPG](#) la zone de rangement est terminée.

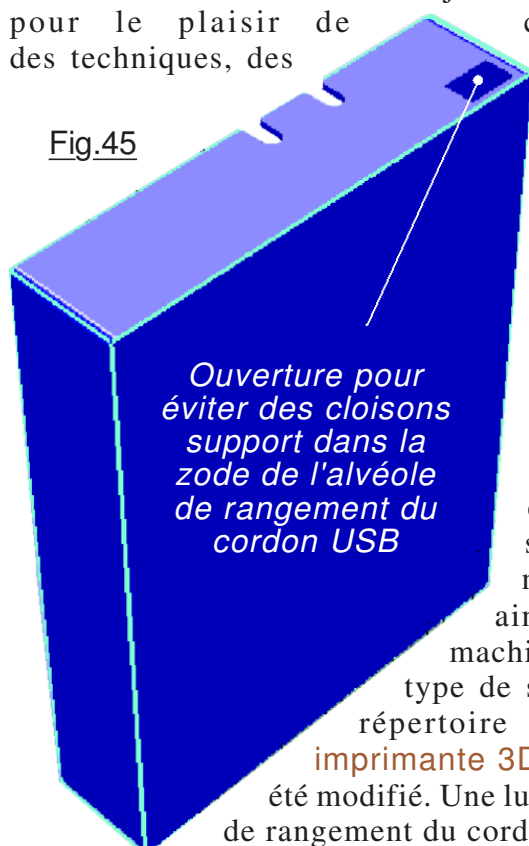


Ce dernier complément décidé et réalisé en dernière minute achève notre promenade dans le monde si agréable du bricolage, domaine où l'on crée "des objets inutiles" pour le plaisir de des techniques, des

concrétiser un rêve avec pour bénéfice le loisir de découvrir technologies qui seront réinvesties dans d'autres projets.

15) Les fichiers pour "mouler" nos éléments sur une imprimante 3D.

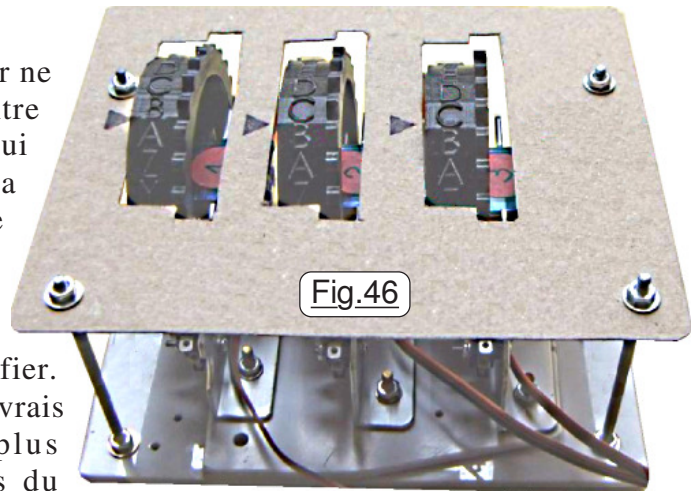
L' usage du polystyrène choc permet d'aboutir à de séduisantes réalisations. C'est toutefois une technique "relativement pointue" qui impose une solide expérience pour arriver à façonner et coller les nombreux éléments sans bavures. Les opérations de taillage des pièces, chanfreinage, perçage des trous et affinage des lumières présentent toutes des pièges. Aussi, vous pouvez profiter des technologies actuelles et en particulier le moulage sur des imprimantes 3D des éléments de base de notre projet. Bien que ma machine soit trop petite pour réaliser le coffret et le couvercle, rien n'interdit de vous fournir les fichiers **STL** et **gco**. Il vous sera ainsi possible de faire appel à un ami qui possède une telle machine, ou de prendre contact avec des prestataires sur Internet, ce type de service devenant banal. Les fichiers fournis dans le dossier répertoire **<Documents>** rangés dans le dossier **<Fichiers pour imprimante 3D>** ont été optimisés. En particulier celui pour le Couvercle a été modifié. Une lumière montrée sur la Fig.45 a été ajoutés en vis à vis de l'alvéole de rangement du cordon USB. La pièce étant moulée verticalement,



ainsi il n'y aura pas les petites cloisons support qui auraient été impossibles à éliminer. La qualité du moulage est de 0.2mm conduisant globalement à une esthétique très correcte. Avec une densité de remplissage de 50% on obtiendra une rigidité et une résistance mécanique tout à fait acceptable avec du PLA. Il n'est pas question de compliquer la réalisation du coffret en imposant des pauses dans le moulage pour inclure les écrous prisonniers. Aussi les alésages sont prévus lisses et devront être taraudés ce qui impose la possession d'un taraud ϕ M3 et d'un "tourne à gauche", outillage qui n'a rien d'exceptionnel. Pour la Façade il faudra respecter l'orientation du moulage dans le fichier **gco** car elle conduit à une face bien plate et très peu de cloisons support pour l'assise de la trappe.

16) Les "loupés".

Aucun projet, qu'il soit industriel ou de loisir ne saurait aboutir à la perfection absolue. Entre les désirs initiaux, ce qui était envisagé et ce qui résulte de compromis inévitables, s'insinuera forcément des divergences. Cette modeste réplique d'Énigma n'échappe pas à ce principe non démontré mais qui frise l'absolu. Par exemple le circuit imprimé de la nourrice d'alimentation est bien trop "compliqué", on peut facilement le simplifier. Initialement j'avais envisagé un **Brouilleur** avec de vrais **Rotors** qui tournent ce qui aurait été un plus incontestable par rapport aux autres produits du commerce. Le bloc motorisé montré sur la Fig.46 avec des rotors en 3D munis des 26 lettres en relief était assemblé avec les capteurs de position. Mais il a été impossible de dominer les petits servomoteurs qui sont de type sans limitation de mouvement angulaire. Leur repos n'était pas stable et la consigne d'arrêt était affectée d'un petit retard aléatoire incompatible avec le positionnement angulaire précis. Ce gros dispositif a donc été remplacé par l'afficheur graphique qui de toute façon aurait été indispensable. C'est un peu dommage, mais face à une impossibilité il faut accepter tout cet investissement perdu. Il est compensé par de l'expérience en plus, et surtout, la machine aurait été bien plus volumineuse. La compacité du prototype reste un avantage incontestable. Au final, dans ce petit coffret blanc j'ai au final intégré bien plus de fonctions que je ne l'avais imaginé initialement. À mes yeux, le logiciel correspond largement à ce que je désirais. Franchement je ne vois pas ce que l'on pourrait y ajouter, l'ensemble reste un bien séduisant appareil.



Chères lectrices, chers lecteurs, cette longue présentation arrive à son terme. Tout à une fin, mis à part l'Univers, et arrive forcément un moment où il faut raisonnablement considérer que "le voyage d'agrément" est terminé.

Je souhaite intensément que certaines et certains oseront s'engager dans la réalisation d'un clone, je ne doute pas de leur réussite. Surtout, je vous souhaite à toutes et à tous de trouver dans ces lignes le plaisir de la découverte. Si d'aventure vous engagez vos heures de liberté dans une telle réalisation et que vous rencontriez une difficulté, les amis du forum pourront probablement vous aider. Dans le pire des cas, vous pouvez me contacter sur : michel.droui@laposte.net et dans les limites de mon temps de libre, c'est avec grand plaisir que je tenterai de vous dépanner. Je vous souhaite à toutes et à tous agréable lecture et ... que vos secrets militaires soient bien gardés !

Chaleureusement : Nulentout.

