

## Recommandations pour réaliser les fiches.

Par Nulentout (Achevé le 28 Novembre 2019.)

Mise à part cette première page d'explications, toutes les autres **sont prévues pour réaliser des fiches en imprimant les "pages" de ce document par "paire" Recto / Verso** sur du papier de qualité suffisante pour que les éléments situés d'un côté ne soient pas visibles de l'autre. Pour ne pas vous tromper, le tableau de la Fig.1 précise les paires constituant une même "fiche double" car chaque page au format A4 est relatif à deux fiches indépendantes.

Réaliser une "paire de fiche" n'est pas spécialement compliqué, toutefois le fait d'avoir à imprimer des deux côtés d'une feuille impose une procédure simple, mais rigoureuse. À titre d'exemple on va traiter le cas des fiches couplées de la FEUILLE ③ :

1) Imprimer la **page 6**. (Repère vertical en gris clair au centre.)

2) Replacer la page sur votre périphérique et imprimer la **page 7**.

Logiquement, si c'est une imprimante classique, il suffit en principe de replacer la feuille sur le dessus du bac à papier dans la position et l'orientation qu'elle occupe en sortie de la machine.

3) Étape non obligatoire, personnellement je protège toutes mes fiches, disposant d'une petite plastifieuse thermique pour P.C.

4) Il ne reste plus qu'à séparer les deux fiches en coupant la feuille par le milieu. Puis on découpe tout le tour de la fiche le cadre gris clair qui en délimite la périphérie "en laissant vivre le trait".

Globalement, les fiches sont ordonnées en fonction de la nature des informations qui y figurent. Sur les feuilles ① à ⑥ incluses on trouve les **Fiches d'exploitation du bras mécanique** décrivant les pages écrans, les protocoles de mise en œuvre et les conseils d'utilisation. Les feuilles ⑦ et ⑧ sont relatives à la maintenance du MATÉRIEL. Enfin, les feuilles ⑨ et ⑩ concernent le LOGICIEL.

*Naturellement, il n'est absolument pas obligatoire d'imprimer toutes les fiches. Vous avez parfaitement le loisir de ne concrétiser que celles qui vous semblent strictement indispensables ...*

## "Paires de pages" pour réaliser les fiches Recto / verso.

RECTO	VERSO	RECTO	VERSO	RECTO	VERSO
2	3	4	5	6	7
8	9	10	11	12	13
14	15	16	17	18	19
20	21	22	23	--	--

Fig.1

Deux fiches sur une même FEUILLE imprimée RECTO / VERSO	①	(2) Format des commandes. 1/4 et 4/4
	②	(3) Format des commandes. 3/4 et 2/4
	③	(4) Liste des ERREURS en exploitation.
	④	(4) Le mode APPRENTISSAGE.
	⑤	(5) Déplacements commandés par CONSIGNES.
	⑥	(6) Les différentes PAGES sur l'écran OLED. 1/3
	⑦	(6) Les différentes PAGES sur l'écran OLED. 3/3
	⑧	(7) Première mise en service du bras manipulateur.
	⑨	(7) Les différentes PAGES sur l'écran OLED. 2/3
	⑩	(8) Quelques recommandations importantes.
Pages du document PDF relatives à deux fiches "croisées sur une même FEUILLE"	①	(9) Gérer le mode APPRENTISSAGE.
	②	(9) Protocole de mise en service du bras.
	③	(10) Gérer les programmes de POSTURES. [1 à 5]
	④	(11) Gestion de la surchauffe du "servomoteur pince".
	⑤	(11) >>>> SIMULATION du programme <<<<
	⑥	(12) Configurations particulières de l'ensemble.
	⑦	(13) Configurations possibles de la pince.
	⑧	(13) Positions angulaires possibles pour les moteurs.
	⑨	(14) Affectation des broches d'interfaçage.
	⑩	(14) Occupation de la mémoire EEPROM.
	①	(15) Schéma électronique de l'interfaçage.
	②	(15) Dessin du circuit imprimé principal.
	③	(16) Utilisation du multiplexeur de servomoteurs.
	④	(16) Branchements de la carte PCA9685.
	⑤	(18) Détermination logicielle de la température. 2/2
	⑥	(18) Mesure de la température du servomoteur.
	⑦	(19) Comportement du capteur de mesure thermique.
	⑧	(19) Détermination logicielle de la température. 1/2
	⑨	(20) Problème de la boucle d'affichage sur OLED 1,3'.
	⑩	(20) Alimentation électrique des divers modules.
	①	(21) Schéma électrique complet du bras manipulateur.
	②	(21) Paramètres d'adaptation du logiciel au matériel.

## Format des commandes. 1/4

Toute commande consiste à saisir au clavier **un premier caractère alphabétique** ou particulier tel que '=', '?', '\*' ou '!' par exemple, suivi éventuellement d'un nombre **N** et terminé par la validation de la chaîne éditée dans la fenêtre du Moniteur de l'IDE. Si **N** n'est pas précisé, il sera par défaut égal à zéro. Tout ce qui suit le caractère d'une commande n'attendant pas d'argument numérique **N** sera ignoré. Sept caractères sont analysés, les suivant sont ignorés.

### ➤ Les commandes "vides".

Pour les commandes spécifiques '+', '-', '.' et 'j' il suffit de ne frapper que la touche de validation pour les répéter à convenance.

### ➤ Liste des commandes non alphabétiques.

- &** : Liste sur la ligne USB les plages limites de positions angulaires ainsi que les limites des CONSIGNES. @
- \$** : Listage de toute les POSTURES sur le moniteur USB. @
- w** : Listage de la séquence de programme APPRISE. @
- ESP** : L'ESpace provoque la rétractation du bras : @
  - PINCE ouverte, *L'écran affiche la page de configuration.*
  - COUDE en position 99°, *(Pince reculée au maximum.)*
  - ÉPAULE en orientation 90° et HANCHE recentrée à zéro.
- ?** : Affiche les paramètres du PROGRAMME **N**.
- \*** : Autorise à nouveau les commandes PINCE après surchauffe. @
- !** : Mouvements indiqués en mode CONSIGNE. (OUI/NON.) @

**ATTENTION** : Ce mode impose des mouvements rapides et s'avère potentiellement risqué pour l'ensemble de la motorisation car les consignes non vérifiées peuvent dépasser les limites des asservissements. C'est la raison pour laquelle la LED jaune s'illumine et la LED d'ARRET D'URGENCE clignote à 5Hz.

- /** : Inverse le sens d'exploration des écrans initié avec 'j'. @  
Cette commande fait passer à l'écran précédent ou à l'écran suivant. Accepte d'être suivie par des commandes "vides".
- + -** : Effectue un pas dans le sens positif ou en négatif. @
- :'** : Arme l'utilisation possible de 'E' sans argument. @
- >** : Active ou suspend le mode APPRENTISSAGE. @
- <** : Efface sans préavis la posture APPRISE d'ordre **N**.
- =** : Affiche les valeurs de Début et de Fin, les données de la Pince ainsi que les valeurs des temporisations sur la ligne USB. @

## Format des commandes. 4/4

### ➤ Commandes utilisant un caractère Majuscule.

- A** : Arme le système après un ARRET D'URGENCE. @
- B** : Enregistre une posture de type BIP.  
Si l'emplacement de la posture est valide génère l'erreur n°21.  
*(Paramètre N dans la fourchette [1 à 50].)*
- D** : Posture de Début d'une séquence automatique. *(Voir '='.)*  
*(Paramètre N dans la fourchette [1 à 50].)*
- E** : Exécute le programme **N** dans la fourchette [1 à 5].  
'E'proposé sans argument déclenche la séquence APPRISE.
- F** : Posture de Fin d'une séquence automatique. *(Voir '='.)*  
*(Paramètre N dans la fourchette [1 à 50].)*
- G** : Passer l'écran en Mode Graphique. @ *(Sortie avec 'b'.)*
- I** : Insérer en position **N** de l'APPRENTISSAGE la posture actuelle.
- K** : Efface le programme **N** si fourchette [1 à 5]. (KILL.)
- L** : Liste en Hexadécimal le contenu de la mémoire EEPROM. @
- M** : Mémorise en EEPROM la configuration actuelle du bras manipulateur en tant que *Posture particulière*. @
- P** : Purge intégralement l'EEPROM et initialise les données de base.  
Pour sécurité impose **N** à 1024, valeur de la taille EEPROM.
- Q** : Quitter le programme. @ *(Même effet que 'q' minuscule.)*
- R** : Restitue la *Posture particulière* si moteurs ON. @  
*L'écran passe en affichage de la page configuration.*
- S** : Serrage de la pince avec **N** représentant l'écart désiré entre les mors. *(Exprimé en mm.)* Valeur comprise dans la plage [1 à 42].
- U** : (Utilisation.) Liste sur l'écran OLED les séquences automatiques actuellement validées en EEPROM. @
- V** : Valide la séquence de programme désignée par **N**. (**N** : [1 à 5].)  
Cette commande ne sera acceptée que si FIN > DEBUT.
- X** : Élongation maximale du bras manipulateur si confirmation. @
- Z** : (Remise à Zéro.) Efface toutes les postures et les cinq séquences de programmes situées en EEPROM. *(Place des \$00 dans les cellules mémoire.)* Impose l'argument **N** = 50 pour éviter un effacement involontaire. *(L'écran passe en listage des postures valides.)*  
Si mode APPRENTISSAGE efface le programme enregistré à condition de confirmer avec **N** = 60 pour éviter une erreur.

## Format des commandes. 3/4

- m** : Mémoire en cellule EEPROM à l'emplacement **N** la configuration actuelle du bras manipulateur si la fourchette respecte [1 à 50].  
*L'écran liste les postures mémorisées en EEPROM.*
- n** : Neutralise les moteurs. (Bascule de type OUI/NON.) @  
En bas à gauche de l'écran dans un petit cadre est précisé l'état de la motorisation : 'A' si Active et 'N' si neutralisée.  
*L'écran affiche la page dédiée à "État MOTORISATION".*
- o** : Ouvre la PINCE si les moteurs sont actifs. @  
*L'écran passe en affichage de la page configuration.*
- p** : Puissance OUI/NON. (Disjoncteur alimentation des moteurs.)
- q** : Quitter le programme : Passe l'écran en mode SOMMEIL et si nécessaire réactive les moteurs. Impose au bras robotisé la posture "Rétracté" et PINCE ouverte. Neutralise les moteurs. L'écran passe en affichage de base des paramètres de configuration du bras. (Changement invisible puisque l'afficheur est en sommeil.) La sortie du mode SOMMEIL est fortement recommandée par usage des commandes 'd', 'b' ou 'v' car les autres commandes se font en "aveugle" l'écran étant tout noir.
- r** : Restitue la posture située en emplacement **N** de l'EEPROM si elle est valide. **N** doit respecter la fourchette [1 à 50].  
*L'écran passe en affichage de la page configuration.*
- s** : Ferme la PINCE à la valeur de serrage indiquée par **N**.  
(Ne la ferme pas si la valeur de **N** est incorrecte.)  
**N** : Fourchette [30 à 90]. L'écran passe en mode graphique.
- t** : Impose par **N** la valeur de la Temporalisation en mode mouvements lents. (Mvts Lents avec la commande 'l'.) **N** doit respecter la fourchette [5 à 30]. (Exprimé en mS.) La valeur est sauvegardée en EEPROM et rechargée automatiquement sur un RESET.  
*L'écran passe en affichage de l'état MOTORISATION.*
- u** : Liste les postures Utilisables sur l'écran OLED. @
- v** : Écran en Veille et éteindre toutes les LEDs OUI/NON. @  
*L'écran passe en affichage de la page configuration.*
- x** : Configuration avec la PINCE au plus bas si confirmation. @

### > Commandes utilisant un caractère Majuscule.

Pour "simplifier" le choix des caractères de commande, et surtout pour en faciliter la mémorisation, toutes les commandes dédiées aux séquences automatiques sont prévues en lettres MAJUSCULES. (Et quelques commandes spécifiques comme celles du listage EEPROM, le serrage PINCE indiqué en écart ...)

## Format des commandes. 2/4

### > Liste des commandes alphabétiques minuscules.

- a** : Désigne l'Articulation qui sera pilotée en mode incrémental.  
**ATTENTION** : La commande 'a1' ouvre la PINCE.  
(Paramètre **N** dans la fourchette [1 à 4].)  
*L'écran passe en affichage du mode Incrémental.*
- b** : Retour à l'écran de Base. Fait aussi sortir du mode SOMMEIL imposé par 'q' ou 'v'. Annule également le Mode GRAPHIQUE.
- c** : Si les moteurs sont actifs positionne le Coude en orientation **N**.  
**N** doit respecter la fourchette [-40 à +90]. (1)  
*L'écran passe en affichage de la page configuration.*
- d** : Démarrer le programme. C'est la réciproque de QUITTER. Fait sortir l'écran du mode SOMMEIL et impose des mouvements Lents. Impose au bras robotisé la posture de rétraction avec la PINCE ouverte. Ne change pas l'état ON/OFF de la motorisation qui sera sur OFF si on a utilisé 'q' juste avant l'usage de 'd'.
- e** : Si les moteurs sont actifs positionne l'Epaule en orientation **N**.  
*L'écran passe en affichage de la page configuration.*  
**N** doit respecter la fourchette [0 à +90]. (1)
- f** : Ferme la pince si les moteurs sont actifs. @  
*L'écran passe en affichage de la page configuration.*
- g** : Impose l'affichage en mode Graphique. @ (Sortie avec 'b'.)
- h** : Si les moteurs sont actifs positionne la Hanche en orientation **N**.  
*L'écran passe en affichage de la page configuration.*  
**N** doit respecter la fourchette [-90 à +90]. (1)
- i** : Impose la valeur de l'Incrément utilisé en commande "pas à pas".  
(Paramètre **N** dans la fourchette [1 à 50].)  
*L'écran passe en affichage du mode Incrémental.*
- j** : Jalonner d'écran en écran. (Voir également la commande 'l'.)
- k** : Efface la Posture **N** si respecte la fourchette [1 à 50]. (KILL.)  
*L'écran liste les postures actuellement valides en EEPROM.*
- l** : Mouvements Lents OUI/NON. (Lents / Rapides.) @  
*L'écran affiche la page dédiée à "État MOTORISATION".*

(1) : Si le caractère de commande de mouvement est donné sans la valeur numérique l'articulation passe au neutre. (N=0)

@ : Commande sans paramètre numérique. Une valeur éventuelle indiquée avec le caractère de commande sera ignorée.



### Liste des **ERREURS** en exploitation.

#### **ERR** Nature de l'erreur de syntaxe.

- 1 Chaîne "vide" sauf pour '+', '-', et 'j'. ( [↵] seul.)
- 2 Caractère de commande invalide.
- 3 Position angulaire hors fourchette du moteur concerné.
- 4 Num de la Posture ou valeur de l'incrément pour les commandes 'k', 'm', 'r', 'B', 'D' et 'F' hors de [1 à 50].
- 5 Tenter de restituer une posture "vide" avec 'r\*'. 'Z' en mode normal non suivie de 50.
- 6 Commande '?', 'E', 'K' et 'V' hors limites [1 à 5].
- 7 Tenter de valider avec 'V' un programme dont la limite **Fin** est NON supérieure à la veur de **Debut**.
- 8 Tenter '?N' ou 'EN' sur un programme non valide.
- 9 Imposer '<' avec **N** négatif ou un nombre **N** supérieur à celui des postures enregistrées dans la séquence APPRISE.
- 10 Déclencher une séquence avec 'EN' dont les postures ne sont pas TOUTES VALIDES.
- 11 Valeur de Ralentissement 'tN' hors de [5 à 30]. (En mS.)
- 12 Commande 'a' hors de la fourchette [0 à 4].
- 13 En mode Incrémental tentative de sortir des limites angulaires de serrage de la PINCE.
- 14 Commande 'S' hors limites [1 à 42]. (Écart en mm.)
- 15 Commande 'i' hors limites [1 à 50].
- 16 Utiliser '\*' ou lorsque l'on rétablit les moteurs avec 'n' alors que la pince est encore en état de surchauffe.

 **Ne rétablir son utilisation avec la commande '\*' lorsque sa température sera redevenue convenable.**

- 18 Tentative d'utiliser les commandes 'o', 'f' ou 'S' alors que les directives sont données en mode "CONSIGNES".
- 19 Tenter 'E' alors que la pince est en "sécurité thermique".
- 20 Tenter 'E' alors que l'exécution n'est pas armée avec ':'. Commande 'B' tentée sur une posture actuellement valide.
- 21 Mémoire **APPRENTISSAGE** saturée à 60 postures.
- 22 'P' non suivie de 1024. (Purge complète de l'EEPROM.)
- 23 'E' ou '<' sur une séquence **APPRENTISSAGE** vide".
- 24 Tenter 'Z' non suivi de 60 en mode **APPRENTISSAGE**.
- 25 'IN' en mode **APPRENTISSAGE**, programme "vide" ou saturé, valeur de **N** < 1 ou supérieure au nombre d'enregistrements.

### Le mode **APPRENTISSAGE**.

**C'** est un programme particulier logé dans une zone spécifique de l'EEPROM qui en exécution commencera toujours au premier emplacement. Dès que l'APPRENTISSAGE est activé avec '>' la LED verte se met à clignoter et continuera tant que ce mode ne sera pas suspendu. La commande '>' est de type OUI / NON et ignore tout ce qui suit le caractère de fonction. La commande QUITTER, ainsi qu'un déclenchement du B.P. URGENCE annule l'APPRENTISSAGE. La commande de mise en veille 'v' désactive également cette fonction.

#### ➤ **Fonctionnement de l'apprentissage.**

**T**oute commande de déplacement issue de 'c', 'e' et 'h' engendre la sauvegarde en mémoire apprentissage de la configuration qui en résulte. L'intégralité des autres commandes est ignorée. Si entre deux sauvegarde on a frappé 'o' ou 'f' l'état de la pince sera pris en compte normalement dans la posture sauvegardée, et n'est donc pas "perdue". Si l'on propose une commande de déplacement avec les commandes 'c', 'e' ou 'h' et que les 60 emplacements sont consommés, l'erreur n°22 est générée. Le mouvement est toutefois exécuté conformément à l'instruction frappée au clavier.

- Contrairement aux enregistrements de posture, le BIT de validation est inutile ainsi que celui pour l'instruction de "BIP". (BIP est inutile car en fin de séquence un texte est affiché sur la ligne USB.)
- L'Index\_APPRENTISSAGE pointe en permanente l'ordre du prochain emplacement disponible en zone EEPROM. Lorsque Index\_APPRENTISSAGE = 1 c'est que le programme est vide.
- Durant l'exécution du programme APPRIS les moteurs sont systématiquement en mode "**mouvements lents**".


#### ➤ **Listage sur la ligne USB avec la commande 'w'.**

**P**articulière, cette instruction est conçue pour lister les différentes postures d'un apprentissage sur la ligne USB du Moniteur de l'IDE. Le caractère 'w' a été choisi pour le mémoriser facilement car il voisine directement avec les deux commandes principale de l'APPRENTISSAGE '>' et '<N'. La commande '=' indique sur la ligne USB du Moniteur de l'IDE le nombre actuel de postures enregistrées dans la mémoire d'APPRENTISSAGE. (Précise également diverses autres données d'exploitation du système.)

- La commande '>' **active** ou **désactive le mode APPRENTISSAGE**.  
Ce mode sera automatiquement désactivé :
  - Si on impose '>' alors qu'il est actif,
  - Après un ARRET D'URGENCE,
  - Par usage de la fonction QUITTER avec 'q' ou 'Q',
  - Si effacement du programme avec 'Z60' le mode étant actif,
  - Si on exécute la séquence programmée avec la commande 'E',
  - En passant en mode VEILLE avec la commande 'v',
  - En purgeant entièrement l'EEPROM avec 'P1024'.
- C'est la **commande 'E' sans argument** qui **déclenche l'exécution d'un programme d'APPRENTISSAGE**. Ce ne sera possible que si au préalable cette option a été validée avec ':'. Durant la réalisation, le programme informe sur la ligne USB l'ordre de l'étape en cours.
- C'est la commande ':' qui active l'utilisation possible de 'E' sans argument. Toutes les commandes autres que 'E' ou ':' invalident l'option ainsi que l'exécution du programme.

 **La sortie du mode peut laisser la LED bleue allumée. Pour l'éteindre, il suffit d'imposer à nouveau l'angle actuel sur l'une quelconque des articulations.**

- C'est la commande 'Z60' qui en mode APPRENTISSAGE efface la séquence mémorisée. Il faut la confirmer avec l'argument 60 ou l'erreur n°25 est générée. (60 : Nombre de postures possibles.)
- La commande '<N' **efface inconditionnellement** la posture N ou la dernière posture APPRISE si '<' seul. **C'est à dire** que l'effacement est effectué même si le mode APPRENTISSAGE n'est pas actif.

 **IMPORTANT :** En mode APPRENTISSAGE actif la commande 'ESP' est utilisable, mais elle enregistre trois fois la configuration rétractée. Si l'on désire l'introduire dans une séquence en cours d'enregistrement, faire suivre 'ESP' de deux fois '<' pour éliminer "les deux clones".


- 'IN' permet d'insérer la posture actuelle en position N de la mémoire réservée à condition d'être hors APPRENTISSAGE, et de fournir un rang cohérent : N >= 1 et inférieur au nombre d'enregistrements. En outre, le programme ne doit pas être "vide" ou saturé à 60.

## Déplacements commandés par CONSIGNES.

**P**révu pour pouvoir déterminer finement certaines positions angulaires des servomoteurs en fonction des consignes qu'ils reçoivent, **cette commande n'effectue aucune vérification sur la validité de la valeur numérique proposée. Si on saisit une valeur numérique hors des limites de consigne valides, le servomoteur peut diverger et forcer en butée l'ensemble mécanique.**

 **ATTENTION :** Les mouvements se font rapidement même si "déplacements lents" degré par degré est active. 

(Quand on quitte ce mode par nouvelle frappe de '!', l'état initial de l'option "rapide" est restitué.) Pour prévenir l'opérateur de l'activation du mode "CONSIGNES", la LED jaune s'allume et la LED d'ARRET D'URGENCE clignote à 5Hz.

 **Note :** Pour modifier la posture de la PINCE il faut utiliser impérativement la commande 's' car les autres commandes 'o', 'f' ou 'S' en mode CONSIGNES sont interdites et génèrent l'erreur n°18.

### **➤ Procédure pour gérer le mode "CONSIGNES" :**

**N'**établi que si la puissance est présente et la machine au repos. Logiquement l'option "mouvements lents" est validée et les moteurs ne sont pas au neutre. Avant d'activer le mode CONSIGNES avec '!', avoisiner la position désirée avec les commandes normales 'c', 'e' et 'h'. Noter la valeur de consigne actuelle sur l'articulation dont on désire évaluer une position spécifique. Passer ensuite en mode consigne avec '!'. Par de menu modifications sur la valeur notée, procéder à des déplacements de faible amplitude jusqu'à obtenir la position angulaire désirée. Noter la valeur de consigne qui correspond à l'orientation présente du servomoteur. Revenir au comportement standard en annulant le mode CONSIGNES avec la commande dédiée '!'. Au retour normal, imposer sur les articulations évaluées des angles correspondant à ceux des dernières consignes testées pour rétablir un affichage normal. (Pour cette ultime action ne pas utiliser de posture mémorisée car il y a divergence entre l'angle réel et celui consigné dans les variables du programme.)



## Les différentes PAGES sur l'écran OLED. 1/3

**PAGE1()** (Écran de base.) Ordre de la dernière posture enregistrée.

**Lent**  
**Rapide**

Moteurs  
**Actifs**  
**Neutralisés**

Etat du BRAS Sv --

PINCE : +30 (373)

Serrage : 2,0 mm

COUDE : +45 (375)

EPAULE : +45 (295)

HANCHE : -33 (329)

Etat du BRAS Sv 36

PINCE : +30 (373)\*

Serrage : 2,0 mm

COUDE : +45 (375)

EPAULE : +45 (295)

HANCHE : -33 (329)

Position angulaire de l'articulation. ↑

Valeur de la consigne "multiplexeur". ↑

Signale une disjonction thermique du servomoteur de PINCE. (Réarmer avec '\*').

**PAGE2()** (Écran des mouvements par Incréments.)

INCREMENTS de 3

Moteur : Epaule

Consigne = 269

Position = +60

INCREMENTS de 10

Moteur : Hanche

Consigne = 309

Position = -23

'iN' : Valeur actuelle de l'incrément.

'aN' : Articulation pilotée.

Ruban représentant la pleine amplitude possible de mvt. Curseur simulant la position actuelle par rapport au balayage total de l'articulation.

**PAGE3()** (Paramètres de la Séquence de Postures désignée N.)

PROGRAMME cible : 2

1 2 3 X X

Debut 1 / Fin 6

Ralentissement 25

Nb postures 6

1 2 3 X X

Debut 1 / Fin 6

Ralentissement 30

Posture > 4

'?N' : Présente les paramètres d'une Séquence valide. Ici : '?2'

x : Précise une séquence NON valide.

Postures de DÉBUT et de FIN de N.

Valeur de la temporisation entre chaque degré de mouvement exprimé en mS et imposé par 'iN'.

Nombre total de postures que comporte la séquence N. (Un éventuel BIP est compté comme une posture.)

"sablier" qui visualise l'évolution d'un programme en cours d'exécution, la plage totale représente l'intégralité des postures.

Ordre de la posture qui vient d'être réalisée.

**PAGE4()** (État actuel que l'on obtient avec 'u' de la zone des 50 Postures.)

Un nombre indique une posture valide.

"==" indique une posture de type "BIP".

Nombre total de postures actuellement valides : On peut les réaliser avec 'rN'.

01 02 03 04 05 06

30 31 32 33 34 35 36

50 NB : 15

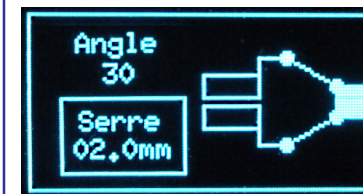
## Les différentes PAGES sur l'écran OLED. 3/3

**PAGE6()** (Représentation graphique du bras manipulateur.)

En 1 le bras manipulateur est représenté de profil, le graphe tracé respectant les proportions et les angles. En haut à gauche est précisée la position angulaire du COUDE. En 2 la fenêtre représente le bras manipulateur vu de dessus avec repérage ⊖ et ⊕ des sens de rotation. En bas à gauche est indiquée position angulaire de l'EPAULE et à droite celle de la HANCHE. On retrouve en 3 l'échelle des graduations du thermomètre du servomoteur de PINCE sans indication des valeurs numériques. En 4 la température a dépassé le seuil critique et l'alimentation du servomoteur de PINCE a été disjonctée, événement visualisé sur l'angle supérieur droit de la PAGE1().



**PAGE7()** (Présentation graphique de la PINCE.)



Complément à la PAGE6() cette fenêtre représente la configuration géométrique actuelle de la PINCE sans tenir compte des mâchoires Ouvertes / Fermées. La Représentation respecte les proportions et les angles. La valeur de l'Angle correspond à celle de la commande 'sN', alors que l'écart entre les mors indiqué par Serre résulte d'un calcul logiciel. Si c'est la commande 'SN' qui est utilisée, le programme d'exploitation calcule la valeur de l'Angle et en adopte la valeur entière la plus proche.

Déclenché automatiquement suite à une surchauffe en 98, ou provoqué manuellement avec le BP d'URGENCE en 99, PAGE98() et PAGE99() imposent chacune une reprise "sécurisée" avec 'A'.

98

REPRENDRE avec 'A'

Th moteur PINCE !

99

REPRENDRE avec 'A'

URGENCE !

## Première mise en service du bras manipulateur.

**P**articulier dans son architecture, le programme d'exploitation du bras manipulateur n'est pas complet au téléversement du code objet. En effet, sur un RESET le microcontrôleur va chercher en EEPROM des données qui sont supposées s'y trouver. Hors, ces dernières n'y seront que si nous les avons initialisées à partir du programme d'exploitation. *Il faut impérativement débiter par des manipulations spécifiques :*

- 1) La carte Arduino NANO est reliée par sa ligne USB au P.C.
- 2) La ligne d'**alimentation électrique en puissance** n'est **pas branchée**.
- 3) Téléverser **P07\_Programme\_Complet.ino** à partir de l'**IDE**.
- 2) Avec le Moniteur de l'**IDE** envoyer la commande '**P1024**' qui va purger entièrement l'EEPROM de ses contenus. ATTENDRE, car c'est un peu long. Quand l'opération est achevée, trois BIPs se font entendre et l'écran OLED affiche : **EEPROM INITIALISÉE**.

Remplit l'EEPROM avec des \$FF entre 0000 et 0524.

Remplit l'EEPROM avec des \$00 entre 0525 et 1023.

Initialise la valeur de '**t30**' à 30mS.

Valide la première séquence de posture avec Début = 1 et Fin = 2.

La posture PARTICULIERE devient "Rétractation du bras".

Les postures 1 et 2 sont initialisées en "Rétractation du bras".

La variable Nb\_Postures est initialisée à 2.

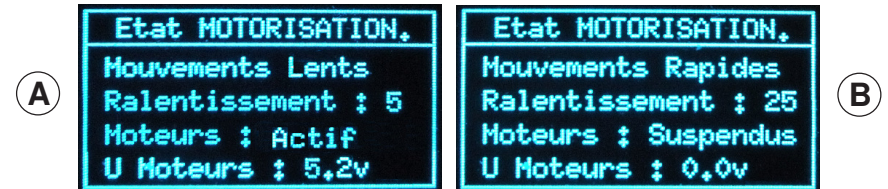
La mémoire d'APPRENTISSAGE est entièrement "vidée".

- 03) Frapper la commande '**b**' pour obtenir l'écran de base. (**Sv** = 2)
- 04) Envoyer '**u**' pour vérifier l'état de la zone des postures. (**NB** : 2)
- 05) Soumettre '**\$**' pour lister sur USB toute la zone des postures.
- 06) Proposer '**U**' pour vérifier le bilan des séquences [1 à 5].
- 07) Commande '**L**' pour lister l'intégralité de l'EEPROM sur USB.
- 08) Tester '**<**' pour vérifier un APPRENTISSAGE "vide". (ERR n°24.)
- 09) Envoyer '**f**', '**c45**', '**e20**' et '**h33**' pour imposer une posture banale.
- 10) Vérifier **PINCE : 60** et **Serrage : 31.0 mm** sur l'écran OLED.
- 11) Frapper '**R**' et collationner les paramètres de la "Rétractation".

Nous avons globalement vérifié que le programme d'exploitation fonctionne correctement et que toutes les données initiales sont cohérentes. *Si vous êtes absolument certains que mécaniquement tous les mouvements sont fonctionnels*, passer à une première mise en service en utilisant la fiche ④ des **Protocole de mise en service du bras**.

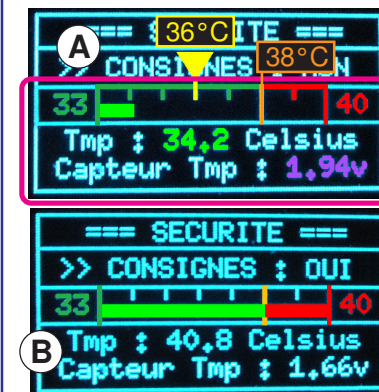
## Les différentes PAGES sur l'écran OLED. 2/3

**PAGE5()** (Écran relatif aux paramètres de la motorisation.)



L'exemple **A** est relatif à une configuration d'exploitation normale. L'énergie de puissance est présente et les moteurs sont **Actifs**. C'est à dire qu'ils ne sont pas **neutralisés**. Actuellement la temporisation entre chaque rotation élémentaire de 1° est de **5mS**, **Mouvements Lents** signalant qu'elle est effective. Sur le cas **B**, le délai a été augmenté à **25mS**. Mais bien que cette valeur soit cinq fois plus grande, l'item **Mouvements Rapides** n'est pas contradictoire, car il informe que le **ralentissement n'est plus pris en compte**, les mouvements étant effectués à vitesse maximale des servomoteurs. **Moteurs suspendu** signifie qu'ils restent immobiles ne recevant plus de consignes. Sur l'exemple **B** l'énergie de puissance a été coupée avec '**p**'.

**PAGE8()** (Écran concernant les paramètres de SÉCURITÉ.)



Toute la zone située dans l'**encadré rose** en **A** concerne la **surveillance thermique** du servomoteur animant la PINCE. Les graduations du thermomètre graphique vont de **33°C** à **40°C** avec le seuil de disjonction repéré à **38°C**. À **36°C non repéré sur l'échelle graphique**, une alerte sonore et un texte sur ligne USB prévient l'opérateur d'une potentielle disjonction imminente. C'est la **mesure effective de la tension** de la CTN sur **A1** qui sert pour surveiller les seuils critiques. En sont déduites par logiciel la **valeur de la température** et la **longueur du ruban** sur le thermomètre graphique. Comme montré en **B**, si l'on dépasse la "borne" de **40°C** la longueur "du mercure" est limitée à la dernière graduation. L'item **>> CONSIGNE : OUI** prévient l'opérateur que les **commandes** se font en **mode CONSIGNE** et que **les divergences des servomoteurs ne sont plus parées** par des vérifications des valeurs.



## Quelques recommandations importantes.

- 👉 **IMPÉRATIF** : Avant d'établir l'énergie sur les moteurs avec 'p' et avant de piloter un quelconque mouvement, vérifier sur 180° que le "volume enveloppe" du passage du bras est entièrement dégagé, et en particulier lorsque le bras est pleinement déployé à l'horizontale.

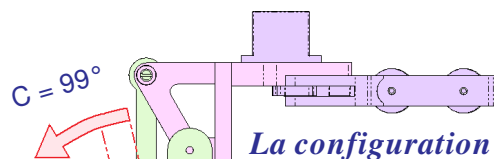


Fig.1

*La configuration de la Fig.1 après avoir saisi une pièce ou pour la déplacer en vue de la déposer devrait systématiquement servir d'intermédiaire avant d'engager une rotation de la HANCHE. Recommandée pour toute rotation en azimuth, avec le COUDE à 99° et l'ÉPAULE à 90°, cette configuration "verticale" soulage au maximum les articulations du COUDE et de l'ÉPAULE, et surtout conduit à l'inertie minimale en rotation pour l'ensemble. Enfin elle minimise les risques de collision durant les balayages en azimuth.*

La durée de ralentissement entre deux mouvements lents de 1° est mémorisée en EEPROM et sa valeur réinitialisée sur un RESET. Lors d'une reprise d'activité il est conseillé de commencer par imposer le délai le plus important autorisé avec '**t30**', puis ne passer à '**t5**' (valeur minimale.) que lorsque l'ensemble de l'environnement du bras manipulateur et les risques de collision sont entièrement pris en compte.

- 👉 **PRIMORDIAL** : D'une façon générale, la commande de coupure d'énergie sur les moteurs avec 'p' n'est pas recommandée. Son utilisation provoque la **perte de synchronisation** entre les positions théoriques et celles réelles. Il en résulte souvent des mouvements brusques.

Compte tenu des informations du bas de cette fiche et des divers protocoles conseillés pour utiliser le bras manipulateur, l'**EXEMPLE** du verso conduit aux postures de la Fig.1 qui liste l'état de la mémoire d'APPRENTISSAGE après avoir enregistré tous les mouvements pour déplacer correctement l'objet.

### APPRENTISSAGE :

Posture 1	Pince Ouverte (///)	C = +99°	E = +90°	H = +90°
Posture 2	Pince Ouverte (///)	C = +99°	E = +0°	H = +90°
Posture 3	Pince Ouverte (///)	C = -40°	E = +0°	H = +90°
Posture 4	Pince Fermée (60°)	C = +90°	E = +0°	H = +90°
Posture 5	Pince Fermée (60°)	C = +90°	E = +90°	H = +90°
Posture 6	Pince Fermée (60°)	C = +90°	E = +90°	H = +30°
Posture 7	Pince Fermée (60°)	C = +90°	E = +0°	H = +30°
Posture 8	Pince Fermée (60°)	C = -40°	E = +0°	H = +30°
Posture 9	Pince Ouverte (///)	C = +90°	E = +0°	H = +30°
Posture 10	Pince Ouverte (///)	C = +90°	E = +90°	H = +30°
Posture 11	Pince Ouverte (///)	C = +90°	E = +90°	H = +0°

Commande w0 !OK!

Fig.1

Colorié en rose est repérée la première posture enregistrée avec '**h90**' en orientant la HANCHE vers l'objet à saisir. En jaune sont mises en évidence tous les éléments qui restent inchangés par rapport à la configuration qui précède. En bleu les mouvements relatifs à des postures à conserver. Quand on efface *les postures inutiles non repérées en vert*, on obtient *le programme de la Fig.2 qui effectue exactement les mêmes déplacements que celui d'origine.*

### APPRENTISSAGE :

Posture 1	Pince Ouverte (///)	C = +99°	E = +90°	H = +90°
Posture 2	Pince Ouverte (///)	C = -40°	E = +0°	H = +90°
Posture 3	Pince Fermée (60°)	C = +90°	E = +90°	H = +90°
Posture 4	Pince Fermée (60°)	C = +90°	E = +90°	H = +30°
Posture 5	Pince Fermée (60°)	C = -40°	E = +0°	H = +30°
Posture 6	Pince Ouverte (///)	C = +90°	E = +90°	H = +0°

Commande w0 !OK!

Fig.2

### Ordre des mouvements lors d'un changement de posture :

Ouvrir / Fermer la pince est placé avant tous les autres mouvements de façon à saisir ou déposer la charge lorsque le bras est dans une configuration correspondant à une posture de libération ou de soulèvement de l'objet. L'ordre dans le logiciel est le suivant :

- Ouvrir ou fermer la PINCE,
- Oriente le COUDE, puis l'ÉPAULE et termine par la HANCHE.



## Gérer le mode APPRENTISSAGE.

Suite à son mode de fonctionnement, concrètement le programme d'exploitation enregistre globalement deux fois trop d'informations. Si la séquence à organiser comporte moins de soixante mouvements ce n'est pas un problème, mais dans le cas inverse il faut faire de la place en mémoire EEPROM. Cette fiche précise les techniques à privilégier.

### ➤ N'enregistrer que le nécessaire.

Conduisant à multiplier les commandes, l'idée consiste à modifier l'état d'ouverture et de fermeture, et surtout deux des trois orientations. Puis, on active le mode APPRENTISSAGE et l'on sauvegarde l'ensemble de la nouvelle posture en modifiant la troisième orientation. Puis, immédiatement on quitte le mode APPRENTISSAGE pour préparer la posture suivante.

EXEMPLE :

- 'o' si nécessaire pour ouvrir la PINCE.
- 'cN' pour orienter le COUDE.
- 'eN' qui place l'ÉPAULE.
- '>' : Active le mode APPRENTISSAGE.
- 'hN' : Enregistre la posture en EEPROM.
- '>' : Désactive le mode APPRENTISSAGE.

### ➤ Purger les postures inutiles d'un programme.

Concrètement, la technique précédente impose d'utiliser en permanence '>' en alternant le "OUI" et le "NON avec le risque d'un oubli engendrant des aléas, voir des postures qui ne sont pas enregistrées. Aussi, on peut passer en mode APPRENTISSAGE, enregistrer progressivement les mouvements, *puis purger les "doublons" pour faire de la place si le besoin s'en fait sentir.*

EXEMPLE :

- 1) Placer le bras en configuration rétractée. (1)
- 2) Activer l'APPRENTISSAGE avec '>'.
- 3) Saisir au plus bas un objet à +90° d'azimut.
- 4) *Soulever la charge au plus haut.*
- 5) Déposer la charge à l'azimut de +30°.
- 6) Replacer le bras en posture rétractée. (2)
- 7) Fermer le mode APPRENTISSAGE avec '>'.

(1) et (2) : Il est fortement recommandé de commencer et de terminer une séquence d'apprentissage par la posture de référence correspondant au bras entièrement rétracté en azimut 0° : Configuration obtenue avec 'ESP'.

## Protocole de mise en service du bras.

- 1) Vérifier que l'alimentation en puissance est débranchée.
  - 2) Relier la carte Arduino au P.C. et activer le moniteur série USB.
  - 3) Moteurs neutralisés, *imposer* aux trois articulations les consignes correspondant à *la configuration actuelle du bras.*
  - 4) Vérifier "vitesses lentes" puis activer la motorisation avec 'n'.
  - 5) Brancher l'alimentation de puissance sur le secteur 220v.
  - 6) *Se préparer à provoquer éventuellement un arrêt d'urgence.*
  - 7) Conjoncter la puissance avec 'p', ralentir au maximum avec 't30'.
- À ce stade le bras doit se trouver au repos en configuration correspondant aux consignes. (DEBUT et FIN sont forcés à zéro sur le RESET pour imposer à l'opérateur d'affecter des valeurs cohérentes avant de valider un enregistrement.)*

### ➤ Lever la charge au maximum avant de changer d'azimut.

Pour saisir un objet privilégier le protocole suivant :

- 1) Bras rétracté et PINCE ouverte, orienter en azimut avec 'hN',
- 2) 'eN' pour une première approche avec l'ÉPAULE,
- 3) Commande 'cN' pour une approche finale avec le COUDE,
- 4) Fermer la PINCE avec 'f',
- 5) Soulever la charge avec 'c90',
- 6) 'e90' pour rétracter le bras avec l'ÉPAULE verticale.

### ➤ Protocole à privilégier pour déposer une charge.

*Le bras manipulateur est chargé, PINCE fermée et au plus haut.*

- 1) Orienter la HANCHE en azimut avec 'hN',
- 2) 'eN' pour une approche initiale avec l'ÉPAULE,
- 3) Pilotage 'cN' pour une approche finale avec le COUDE,
- 4) Déposer l'objet avec 'o',
- 5) Rappel du bras en configuration de référence avec 'ESP'.

*Toujours terminer* les manipulations *par 'q' ou 'Q'* qui permet de reprendre plus tard dans une configuration "propre" et correspondant à la posture qu'imposera un RESET.

ATTENTION : Avant de quitter, déposer la charge proprement, car les commandes 'q' et 'Q' ouvrent automatiquement la PINCE lorsque le bras est positionné en configuration rétractée.

## Gérer les programmes de POSTURES. [1 à 5]

Les commandes spécifiques sont : '\$', '=', '?N', 'u', 'U', 'DN', 'FN', 'KN', 'VN', 'EN', 'Z50' et indirectement 'mN', 'rN', 'BN', et 'kN'.

### ➤ Créer, mémoriser et vérifier des postures.

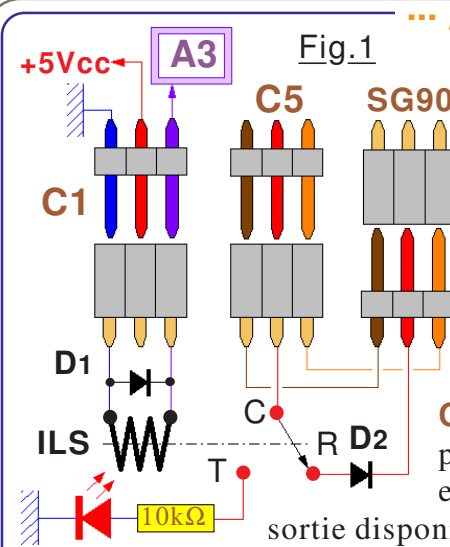
**A**vant de pouvoir valider une séquence de programme, il faut que toutes les postures envisagées soient mémorisées, car la commande 'VN' vérifie que dans la plage comprise entre 'DN' et 'FN' toutes les postures sont valides. Il faut donc commencer par piloter les moteurs pour obtenir une configuration désirée, puis l'enregistrer avec 'mN'. Logiquement, on va créer les postures dans l'ordre chronologique. Dans ce cas, chaque posture est enregistrée juste à la suite de celle qui précède, raison pour laquelle l'écran de base précise en permanence l'ordre du dernier enregistrement effectué. Dans l'exemple proposé ci-contre le prochain enregistrement sera 'm37'. À tout moment on peut effacer une posture avec 'kN', mais contrairement au mode APPRENTISSAGE les postures qui suivent restent inchangées, ce qui peut engendrer un "trou" dans la séquence. La commande 'BN' est particulière. Elle enregistre en position N un "BIP" destiné à prévenir l'opérateur qu'une séquence est terminée par exemple. (*Exige que l'emplacement soit "vide" ou génère l'ERREUR n°21.*) À tout moment on peut imposer au bras l'une des 50 postures avec 'rN', mais pour vérifier intégralement n'importe laquelle sans avoir à générer de mouvement, le mieux est de lister le total avec '\$'.

Sv 36

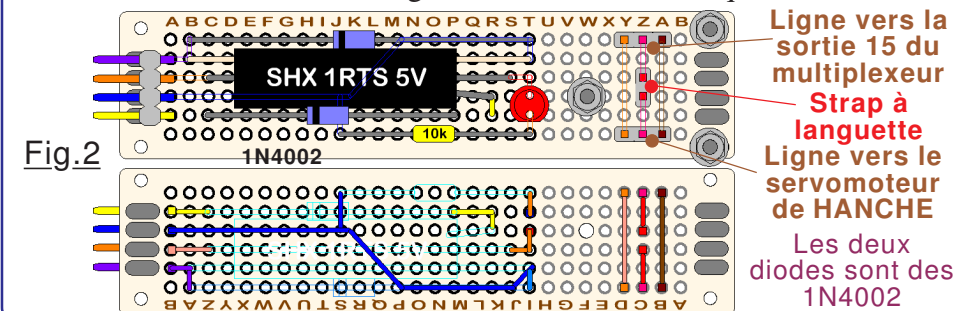
*Il est possible d'effacer toutes les postures en une seule action avec la commande 'Z50'.*

### ➤ Mémoriser une séquence [1 à 5] avec 'VN'.

- 1) Lister avec '\$' sur le Moniteur l'ensemble de la zone des postures.
- 2) Repérer le Début et la Fin de la séquence à valider.
- 3) Avec 'DN' préciser l'ordre en EEPROM de la posture de DÉBUT.
- 4) Avec 'FN' indiquer la position en mémoire de la posture de FIN.
- 5) Vérifier que DÉBUT et FIN sont corrects avec la commande '='.
- 6) Faire afficher la zone EEPROM avec 'u' pour ultime vérification.
- 7) Commande 'U' pour choisir l'une des séquences de [1 à 5].
- 8) Éventuellement libérer une séquence occupée avec 'KN'.
- 9) Tous les paramètres étant correct, Valider la séquence avec 'VN'.



La broche A3 utilisée en sortie binaire pilote un relais à lame souple ILS pourvu de sa diode "de roue libre" D1. Dans le but de minimiser les consommations, quand la PINCE n'est pas en sécurité, son contact repos R sur la Fig.1 alimente en puissance le servomoteur SG90, l'énergie de puissance arrivant du connecteur C5 implanté sur le circuit imprimé principal dans un espace qui restait encore utilisable à proximité d'une sortie disponible. Lorsque la température critique est atteinte, la sortie A3 passe à l'état "1" alimentant ILS qui passe en état travail. Pour permettre à l'opérateur à tout moment de vérifier l'état de l'énergie PINCE, le contact travail T du relais alimente une LED rouge à ≈1mA. Le relais pour son compte consomme moins de 20mA, parfaitement compatible avec le courant que peut fournir en permanence A3. La diode D2 sous-alimente volontairement le servomoteur de PINCE. On augmente ainsi de deux minutes environ la durée séparant l'avertissement de la disjonction. (*Mesuré à ≈4min sans D2 et ≈6min avec la diode.*) Le petit circuit imprimé de complément Fig.2 supportant cette électronique de protection comporte en outre une "interruption" de ligne de puissance du moteur de HANCHE. Par l'enlèvement d'un Strap à languette on peut ainsi à tout moment immobiliser le servomoteur azimutal et ainsi favoriser les manipulations de développement logiciel en "zone encombrée" en diminuant significativement les risques de collision.



## Gestion de la surchauffe du "servomoteur pince".

Mesurée en permanence, la température est comparée à deux seuils critiques. Le premier est choisi à 36°C pour prévenir que le servomoteur ne va pas tarder à être mis hors puissance. Choisie à 2°C de moins que la disjonction d'énergie, l'avertissement permet à l'opérateur de déposer la charge avant que la pince ne soit ouverte automatiquement par le dispositif de sécurité. Quand la température atteint le seuil critique choisi à 38°C, l'alimentation de puissance du servomoteur pince est coupée. *(De ce fait l'objet éventuellement saisi est libéré d'où l'importance de le déposer dès l'avertissement.)* La disjonction de l'énergie sur la PINCE s'accompagne d'une page écran spécifique et d'un BIP sonore pour attirer l'attention de l'opérateur. Par mesure de sécurité les moteurs sont passés en mode 'OFF' c'est à dire qu'ils sont **n**eutralisés.

### Protocole en cas d'avertissement :

- 1) Dès qu'un avertissement  $T > 36^{\circ}\text{C}$  se déclenche, déposer la charge, donc utiliser la commande '**o**' pour ouvrir la pince et ainsi couper la puissance sur le servomoteur après avoir rapidement situé correctement l'objet.
- 2) Attendre que la température soit descendue bien en dessous du seuil de 36 °C pour réarmer avec '\*'. *(Vérifier avec la commande '='.)*

### Protocole en cas de surchauffe :

- 1) Lorsque le déclenchement se produit, accuser réception avec '**A**'. *(Ne fait que reprendre le contrôle "proprement".)*
- 2) Rétablir les moteurs avec '**n**' ce qui produit une erreur n°17. On peut ainsi continuer à piloter les autres articulations du bras manipulateur.
- 3) Attendre que la température soit descendue bien en dessous du seuil de 36 °C *(Vérifier avec la commande '='.)* pour réarmer avec '\*' et retrouver les diverses commandes relatives à la pince.
- 4) Vérifier le comportement correct de la pince par la commande '**f**' suivie de '**o**' pour ne pas laisser inutilement le servomoteur sous énergie.
- 5) Reprendre les activités en cours.

**ATTENTION : Tant que le système n'a pas été réarmé avec '\*' une surchauffe ne provoquera plus de disjonction ou d'avertissement. Il est donc impératif de penser à réinitialiser le programme dès que la température est redevenue normale.**

... / ...

## > Exécuter une séquence [1 à 5] avec 'EN'.

*Avant de déclencher une séquence mémorisée, le bras est supposé se trouver en position rétractée, pince ouverte et HANCHE en position centrée de 0°. En principe l'alimentation en puissance est établie correctement et le servomoteur de la PINCE n'est pas en situation de disjonction suite à une surchauffe.*

- 01) Faire afficher le bilan des séquences de postures avec '**U**'.
- 02) Avec '**?N**' faire afficher les paramètres des diverses séquences actuellement valides pour décider de celle qui sera déclenchée.
- 03) Éventuellement faire résumer l'état de la mémoire avec '**u**'.
- 04) Lister avec '**\$**' sur le Moniteur l'ensemble de la zone des postures.
- 05) Vérifier la cohérence de toutes les instructions de la séquence.

## >>>> SIMULER le programme : Poursuivre en 12 <<<<

- 06) Ajuster la vitesse avec '**tN**' de [5 à 30] en fonction du contexte.
- 07) **Vérifier que la totalité du "volume enveloppe" est dégagée.**  
En particulier en zone de saisie une éventuelle pièce doit être correctement positionnée en fonction du programme mémorisé.
- 08) **Vérifier que la zone de dépose est conforme à la configuration environnementale dans laquelle le programme a été enregistré.**
- 09) Commande '**n**' pour activer les moteurs s'ils sont **n**eutralisés.
- 10) **Se préparer à provoquer éventuellement un arrêt d'urgence.**
- 11) Déclencher l'exécution du programme avec la commande '**EN**'.

## >>>> SIMULATION du programme <<<<

- 12) Neutraliser les moteurs avec '**n**'.
- 13) Bien que non conseillé, couper éventuellement l'énergie avec '**p**'.
- 14) Choisir la rapidité avec '**I**' (*L minuscule.*) ou '**tN**' de [5 à 30].
- 15) Si désiré, passer en mode graphique avec '**g**'.
- 16) Déclencher à convenance le déroulement de la séquence avec '**EN**'. Reprendre en (14) autant de fois que désiré.
- 17) Repasser en affichage standard avec '**b**' puis adopter '**t30**'.
- 18) '**I**' (*L minuscule.*) pour imposer le fonctionnement lent.  
*Retrouver la configuration standard :*
- 19) **Imposer** aux trois articulations *la dernière configuration simulée.*
- 20) Activer la motorisation avec '**n**'.
- 21) **Main "sur le B.P. d'URGENCE"**, rétablir la puissance avec '**p**'.



## Configurations particulières de l'ensemble.

Outre la combinatoire infinie des configurations possibles, certaines présentent des particularités qui les rendent spécifiquement attractives en utilisation. Cette fiche les décrit et précise les positions angulaires des divers axes de la motorisation avec éventuellement un commentaire sur leur utilité.

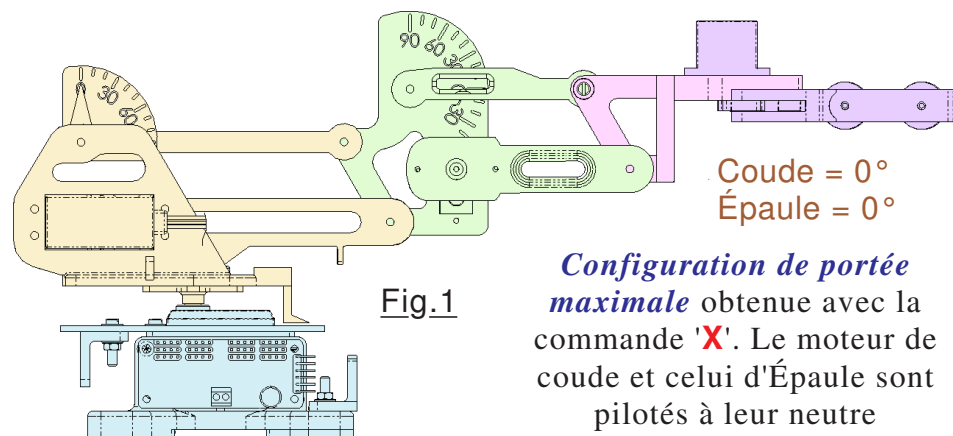


Fig.1

**Configuration de portée maximale** obtenue avec la commande '**X**'. Le moteur de coude et celui d'Épaule sont pilotés à leur neutre opérationnel. **L'orientation de la hanche n'est pas modifiée.**

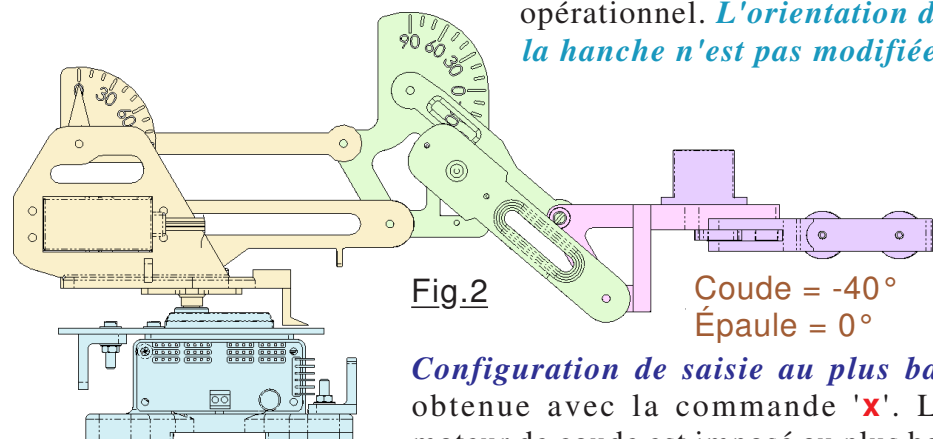


Fig.2

**Configuration de saisie au plus bas** obtenue avec la commande '**x**'. Le moteur de coude est imposé au plus bas ainsi que celui d'Épaule. **L'orientation de la hanche n'est pas modifiée.**

Comme ces deux postures peuvent potentiellement rester dangereuses par un risque de collision avec l'environnement, les commandes '**X**' et '**x**' imposent de confirmer avec '**O**' majuscule qui ne sera pas frappé par erreur. Tout autre caractère fait ignorer ces deux commandes.

... / ...

Page 12

6

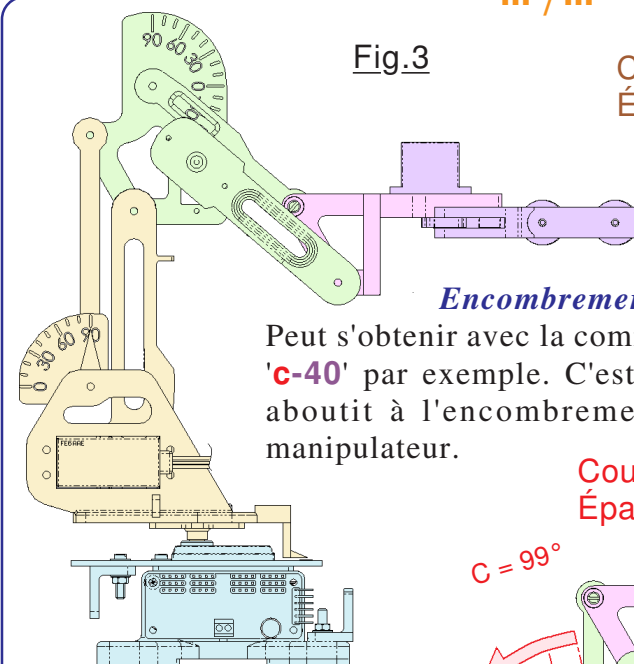


Fig.3

Coude = -40°  
Épaule = 90°

### Encombrement réduit.

Peut s'obtenir avec la commande '**e90**' suivie de '**c-40**' par exemple. C'est la configuration qui aboutit à l'encombrement minimal du bras manipulateur.

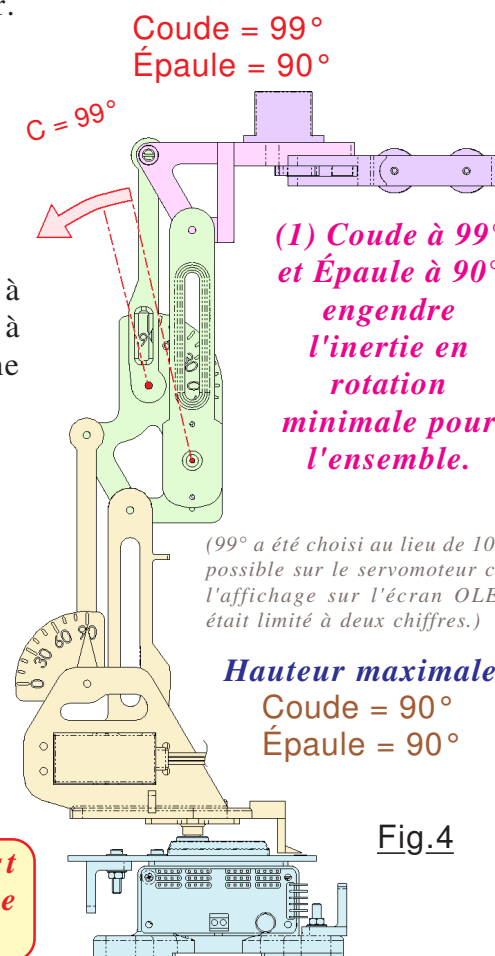
### Hauteur maximale.

Cette morphologie consiste à placer le Coude et l'Épaule à 90°. L'orientation de hanche est indifférente.

### Rétractation du bras.

Imposée par la commande '**ESP**' elle conduit à la posture de la Fig.4 **qui correspond au soulagement maximal des articulations d'Épaule et du coude.** '**ESP**' force en plus la Hanche à l'orientation 0°. C'est la posture qui sera initialisée sur un RESET.

**Cette configuration est recommandée pour toute rotation de la Hanche. (1)**



Coude = 99°  
Épaule = 90°  
C = 99°

**(1) Coude à 99° et Épaule à 90° engendre l'inertie en rotation minimale pour l'ensemble.**

(99° a été choisi au lieu de 100° possible sur le servomoteur car l'affichage sur l'écran OLED était limité à deux chiffres.)

### Hauteur maximale.

Coude = 90°  
Épaule = 90°

Fig.4

## Configurations possibles de la pince.

L'amplitude possible des déplacements est très inférieure à celle de la rotation du servomoteur. Il devient nécessaire pour ne pas donner des consignes qui engageront une surcharge de l'asservissement, de rester dans les limites "du tassement des ressorts".

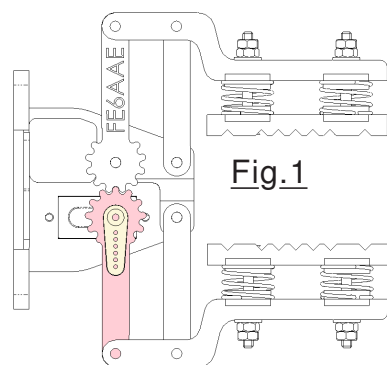


Fig.1

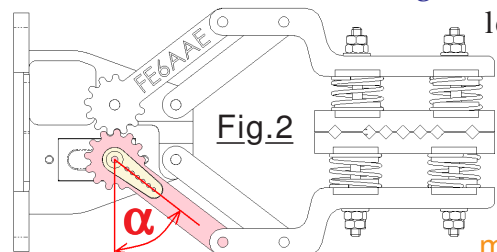


Fig.2

Sur la Fig.1 la pince est en pleine ouverture. L'angle de position du servomoteur est supposé calé au neutre. (*Position du palonnier sur l'arbre de sortie du moteur.*) L'angle  $\alpha$  fait  $0^\circ$ . Sur la Fig.2 on note la symétrie des divers éléments maintenue par les secteurs dentés sur les bielles motrices. **C'est la Fig.3 qui correspond à la plus grande amplitude de serrage** lorsque

les quatre ressorts sont presque entièrement tassés et qu'il n'y a pas d'objet entre les deux mors. Cette configuration est obtenue lorsque le bossage 6 du doigt 4 et son opposé 7 sur le mors sont en contact. L'épaisseur

des deux bossages aboutit à un tassement du ressort à une longueur de 10mm, évitant l'écrasement spire contre spire. L'angle  $\alpha$  maximal correspond à la géométrie de l'ensemble lorsque les deux mors viennent en contact l'un contre l'autre sur le plan de symétrie 5 de la pince. Les angles de position sont mesurés à la référence neutre du servomoteur immobilisé sur le corps de pince 1. On vérifie que pour la fermeture maximale possible,  $\alpha$  faisant environ  $62^\circ$ , la bielle motrice 2 et la bielle libre 3 ne sont pas en butée. (*Espace rouge.*)

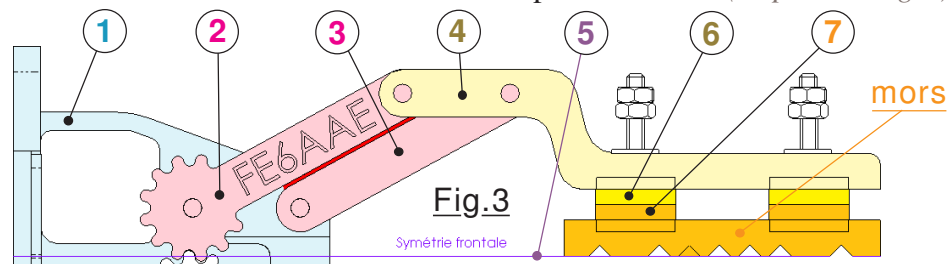


Fig.3

Symétrie frontale

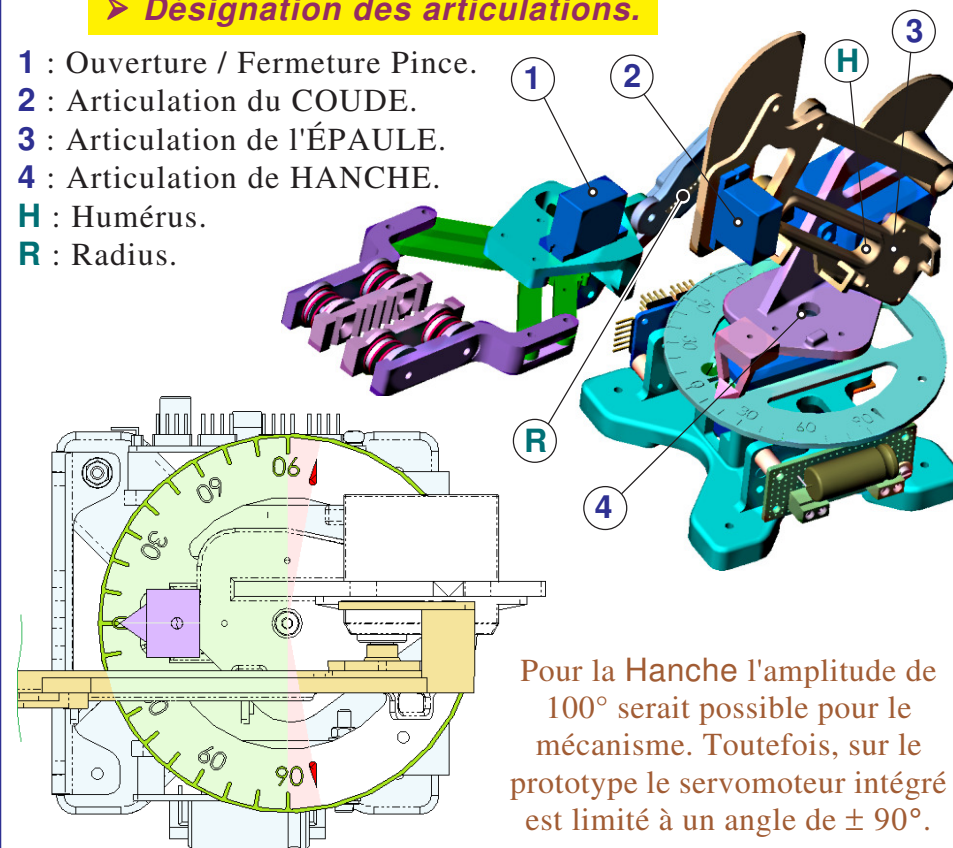
## Positions angulaires possibles pour les moteurs.

Compte tenu de l'architecture globale des divers "membres" du bras manipulateur, les orientations possibles présentent des débattements très inférieurs à ceux correspondant aux rotations potentielles sur les servomoteurs. Il devient nécessaire pour ne pas prendre en compte des consignes qui engageraient une surcharge de l'asservissement, d'effectuer un filtrage des valeurs données au clavier.

Articulation	Position min	Position MAX
Hanche	$-90^\circ$	$+90^\circ$
Épaule	$0^\circ$	$+90^\circ$
Coude	$-40^\circ$	$+99^\circ$
Pince	30	$90^\circ$

### ➤ Désignation des articulations.

- 1 : Ouverture / Fermeture Pince.
- 2 : Articulation du COUDE.
- 3 : Articulation de l'ÉPAULE.
- 4 : Articulation de HANCHE.
- H : Humérus.
- R : Radius.



Pour la Hanche l'amplitude de  $100^\circ$  serait possible pour le mécanisme. Toutefois, sur le prototype le servomoteur intégré est limité à un angle de  $\pm 90^\circ$ .

## Affectation des broches d'interfaçage.

Simplifier l'implantation des composants utilise la flexibilité d'usage des broches de l'ATmega328 pour faciliter l'étude du circuit imprimé. Les affectations des Entrées / Sorties sont directement impactées par l'étude du circuit imprimé principal en utilisant la faculté de certaines broches "analogiques" à pouvoir fonctionner en sorties binaires. Il importe de prendre garde au fait que les broches **A6** et **A7** disponibles sur la carte Arduino NANO ne peuvent fonctionner exclusivement qu'en entrées.

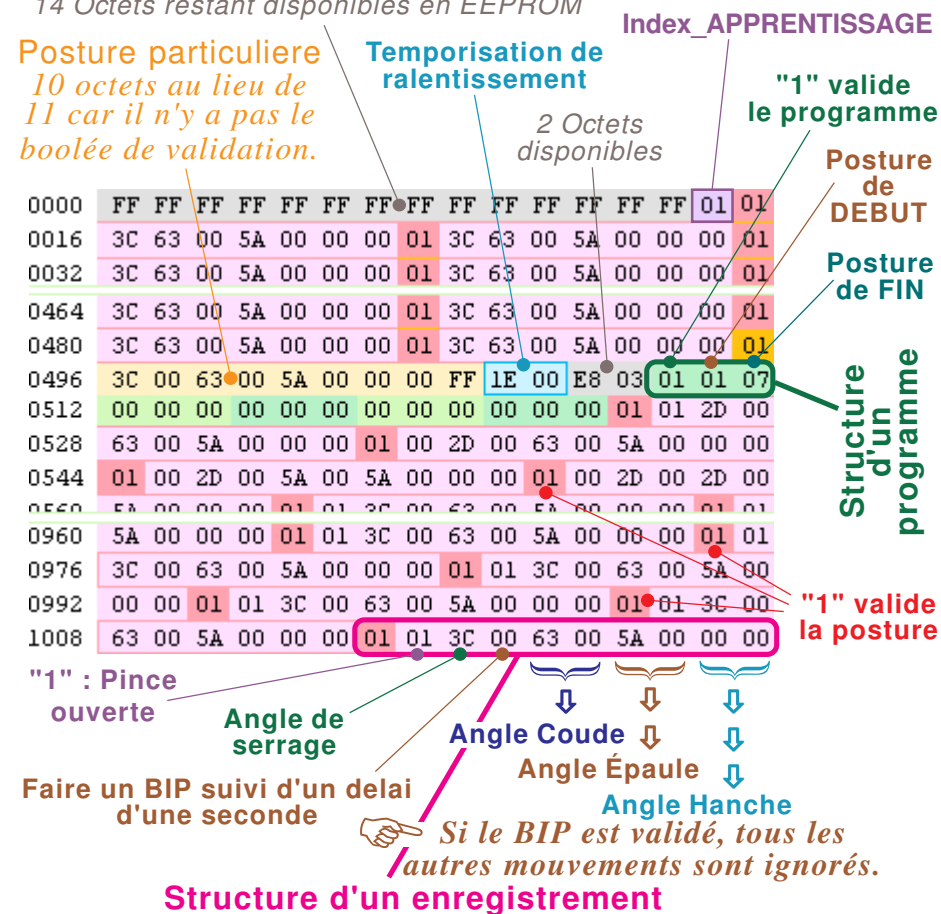
AFFECTATION DES ENTRÉES / SORTIES	
Broche	Utilisation
D0	Réservée pour la liaison série USB. (RX)
D1	Réservée pour la liaison série USB. (TX)
D2	Bruiteur. (BUZZER de type Actif.)
D3	<b>B.P. d'ARRÊT D'URGENCE. (Interruption.)</b>
D4	LED bleue allumée lors du déroulement d'une séquence automatique.
D5	LED verte pour le mode APPRENTISSAGE.
D6	Broche non utilisée.
D7	LED rouge bicolore signalant "Moteurs ON".
D8	Broche non utilisée.
D9	LED verte bicolore signalant "Moteurs Neutralisés".
D10	Broche non utilisée.
D11	Broche non utilisée.
D12	LED jaune qui signale "Moteurs RAPIDES".
D13	Pilotage DISJONCTEUR. (Énergie de puissance.)
A0	<b>LED d'ARRÊT D'URGENCE.</b>
A1	Capteur de température sur le "servomoteur PINCE".
A2	Broche disponible pour future évolution.
A3	Branchement initial d'un potentiomètre.
A4	Broche SDA de gestion de la ligne I2C.
A5	Broche SCL de gestion de la ligne I2C.
A6	Broche disponible pour future évolution.
A7	Entrée de mesure de U puissance moteurs.

## Occupation de la mémoire EEPROM.

Particularité de ce projet : Pratiquement l'intégralité de la mémoire disponible en EEPROM est occupée par les données du logiciel d'exploitation, et particulièrement par la mémorisation des postures des séquences automatiques et celles du programme APPRENTISSAGE.

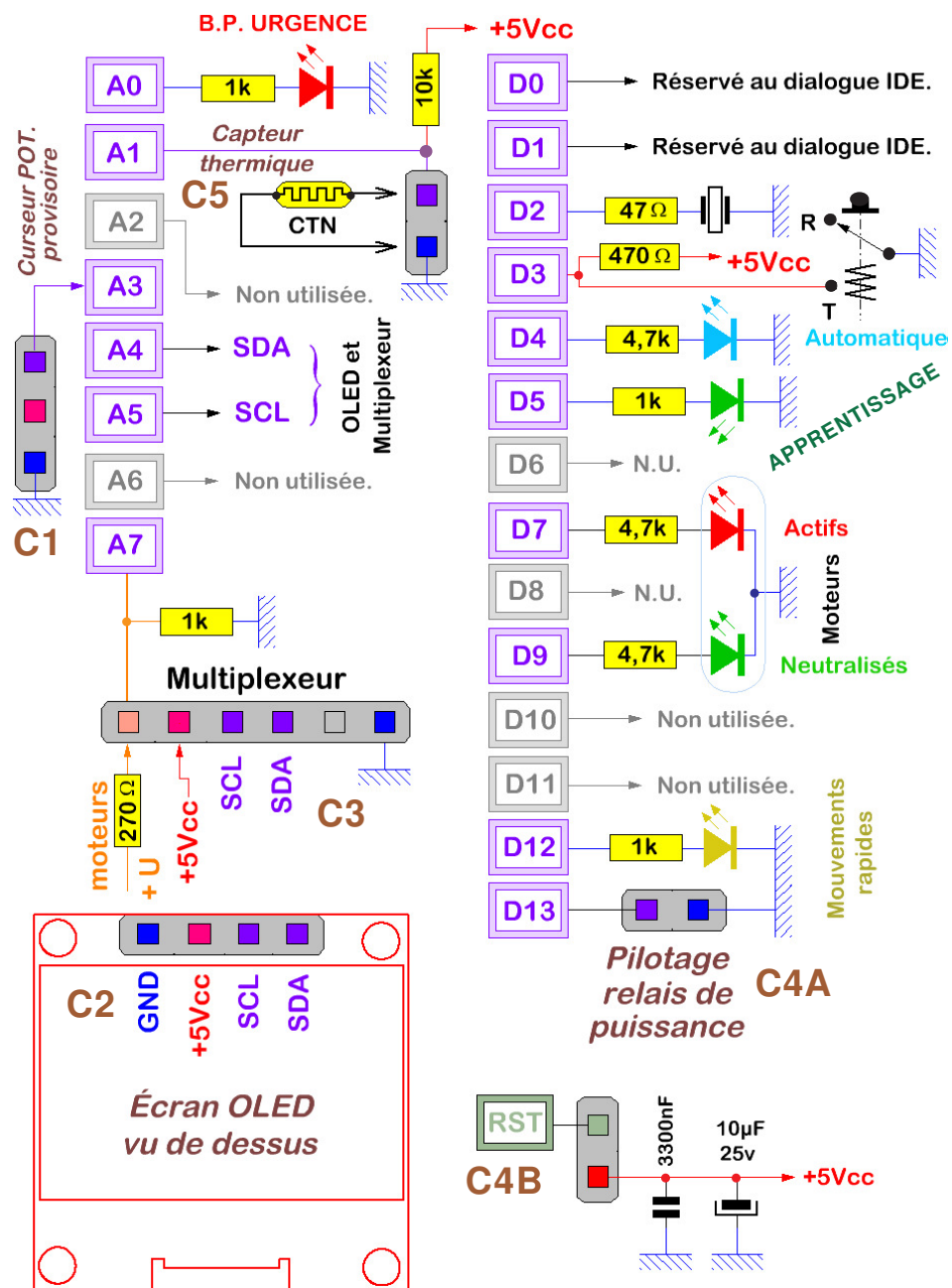
014	1	Index_APPRENTISSAGE
015	480	Soixante postures d'APPRENTISSAGE.
495	10	Posture particulière.
505	2	Temporisation de ralentissement.
507	2	Deux octets non utilisés. (Disponibles pour F.E.)
509	15	Cinq séquences de programmes.
524	500	Cinquante postures mémorisées.

14 Octets restant disponibles en EEPROM

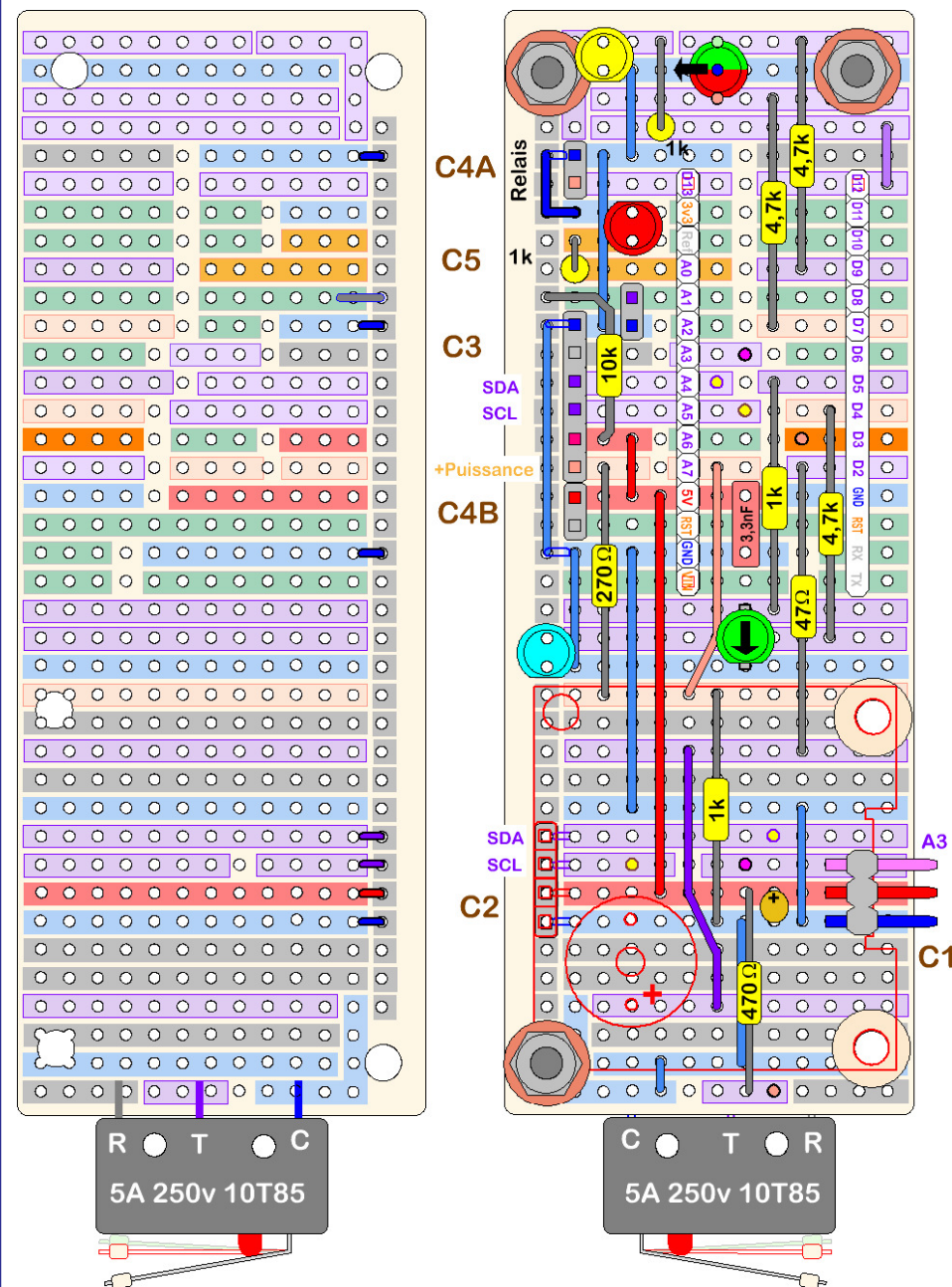




### Schéma électronique de l'interfaçage.



*Dessin du circuit imprimé principal.*



## Utilisation du multiplexeur de servomoteurs.

Fondamentalement ce circuit peut piloter en PWM n'importe quoi avec pour limite que toutes les sorties seront à une fréquence identique. Piloté par la ligne I2C il ne monopolise que les deux broches **A4** et **A5**, et peut les partager avec d'autres modules puisque par "matériel" on peut modifier à convenance son adresse I2C. (Par défaut en 0x40.) Comme le circuit dialogue sur la ligne I2C il faut commencer par `#include <Wire.h>` qui gère la ligne bidirectionnelle. Ensuite on doit déclarer la bibliothèque `<Adafruit_PWMServoDriver.h>` dans l'**IDE** qui gère ce module par des fonctions faciles à utiliser.

### ➤ Programmation basique.

```
#include <Wire.h>
#include <Adafruit_PWMServoDriver.h>
Adafruit_PWMServoDriver pwm = Adafruit_PWMServoDriver();
void setup() {
  pwm.begin(); // Initialise le module multiplexeur
  pwm.setPWMFreq(NN); // Fréquence de la PWM.
                      NN compris entre 24 et 1526.
  void loop() {
    pwm.setPWM(Num_sortie, 0, RCY);
  }
  • Num_sortie sera compris entre [0 et 15].
  • RCY définit le rapport cyclique compris entre [0 et 4095].
```

### ➤ Méthodes de Adafruit\_PWMServoDriver.h.

Spécifique au multiplexeur PCA9685 à 16 canaux cette bibliothèque permet facilement de gérer chaque moteur indépendamment l'un de l'autre. L'effet d'une commande reste effectif jusqu'à une nouvelle consigne pour le canal concerné.

#### Fréquence du signal généré sur les 16 canaux.

```
pwm.setPWMFreq(Fréquence);
```

Cette instruction doit être placée dans `void setup()` et définit la fréquence des signaux PWM générés sur les sorties. La valeur sera comprise entre 24 et 1526, mais en standard on adopte généralement une fréquence de 50Hz si le module pilote des servomoteurs classiques.

## Branchements de la carte PCA9685.

Située au verso de cette fiche la Fig.2 présente les liaisons électriques à établir entre la carte Arduino NANO et le module multiplexeur. Comme les servomoteurs provoquent des appels de courants "virulents" sur la ligne de puissance, un condensateur d'environ 100µF par servomoteur piloté doit être placé au plus proche du bornier "de puissance". (470µF pour 5 servomoteurs.)

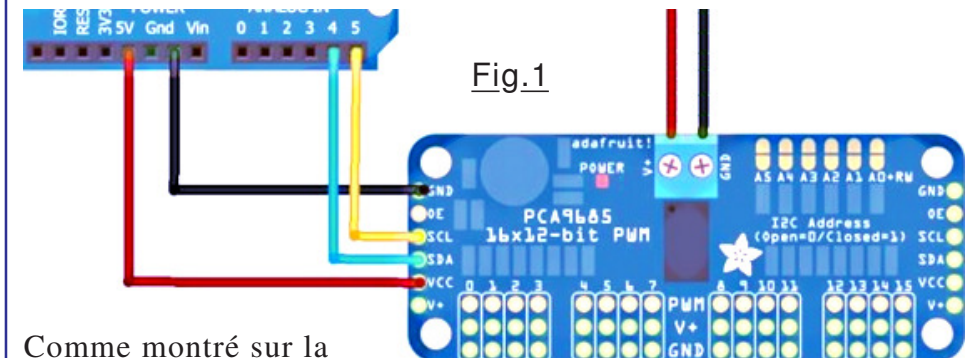
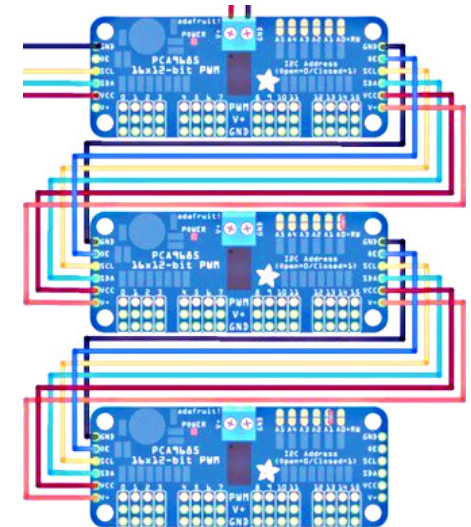


Fig.1

Comme montré sur la Fig.1, si un seul multiplexeur est utilisé ne rien brancher sur **OE**.

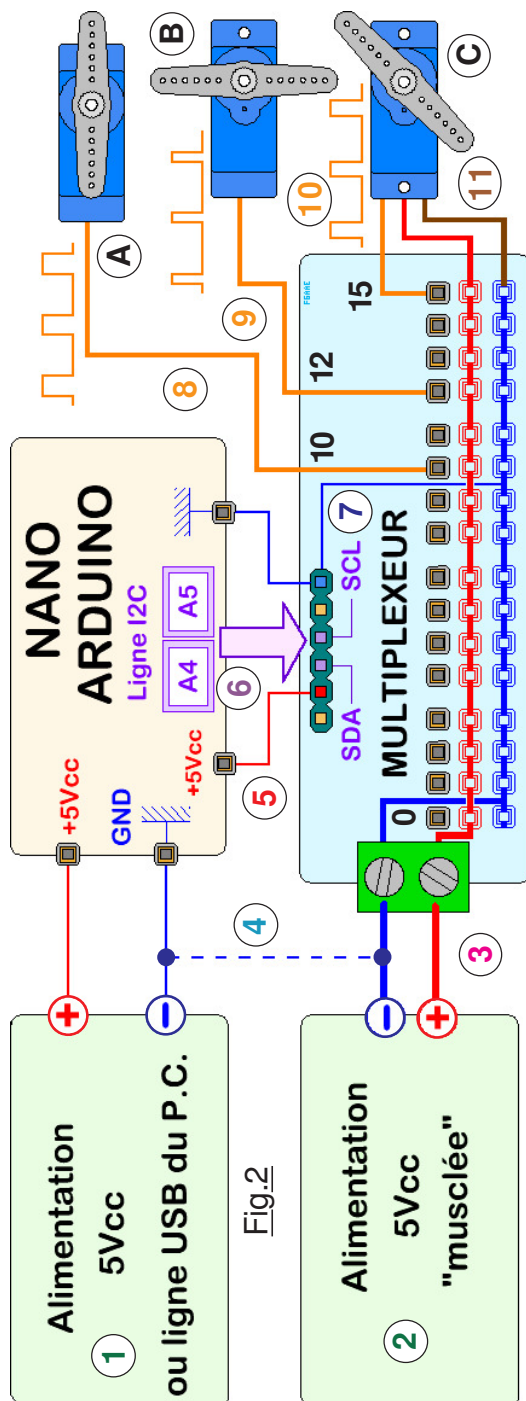
### ➤ Adressage du module.

La bibliothèque impose pour la ligne I2C le signal **SDA** sur **A4** et le signal **SLC** sur **A5**. L'adressage par défaut sur le matériel est en 0x40. On peut à la demande, ce qui sera obligatoire si on chaîne plusieurs modules, (On peut en chaîner jusqu'à 64 !) modifier l'adresse matériellement par établissement de contacts sur des ponts disponibles sur le circuit imprimé dans la zone **RW**.



RW 0 :	Address = 0x40	Adrs binaire 000000	(Pas de pont)
RW 1 :	Address = 0x41	Adrs binaire 000001	(Pont sur A0)
RW 2 :	Address = 0x42	Adrs binaire 000010	(Pont sur A1)
RW 3 :	Address = 0x43	Adrs binaire 000011	(Sur A0 et sur A1)
RW 4 :	Address = 0x44	Adrs binaire 000100	etc. (Pont sur A2)





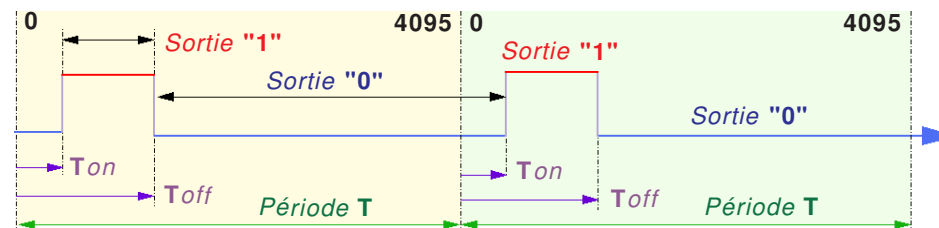
La Fig.2 résume assez bien les branchements à effectuer. Ce dessin fait apparaître les liaisons filaires électriques à établir. En **1** une petite alimentation alimente la carte Arduino NANO qui ne consomme qu'une énergie dérisoire. *(Ce peut être la ligne série USB qui va au P.C. durant la programmation.)* On utilise des fils de faible section. En **2** l'alimentation musclée en "gros fils" **3** va directement au bornier. Logiquement en **4** il faudrait établir un lien entre les deux masses de référence. Ce n'est pas indispensable, car en **7** la liaison est établie en interne sur la carte électronique. Notez au passage que les deux lignes de picots du bas pour la masse et pour le +5V de puissance sont réunies. Ainsi tous les moteurs branchés seront en parallèle comme en **11** sur ces deux lignes d'alimentation. Enfin, chaque ligne orange d'un moteur va sur la sortie **0** à **15** de pilotage qui lui est réservée. L'électronique locale est alimentée par une ligne **5** à part qui logiquement sera reliée à celle d'Arduino.

### Principe de la génération PWM.

La génération PWM est basée sur un compteur à 12 BITS qui fonctionne à une cadence définie par `pwm.setPWMFreq()` et qui en détermine la période **T**. *Le signal présentera une période identique sur toutes les sorties d'un même module.* L'instruction qui conditionne une sortie contient deux paramètres **Ton** et **Toff** qui permettent à notre guise de définir la durée à l'état "1". Ce sont les valeurs d'un compteur interne qui déclenchent les changements d'état :

*Sortie "1"* =  $T_{off} - T_{on}$ .

*Sortie "0" = Période T - Sortie "1".*



**Largeur de l'impulsion "positive" générée.**

```
pwm.setPWM(Num sortie, Ton, Toff)
```

Cette instruction engendre sur la prise **Num\_sortie** un signal PWM à la **Fréquence** prédéfinie dont la durée de l'état "1" est précisée par les valeurs **Toff** - **Ton** sur une échelle [0 à 4095].

Pour s'affranchir de la différence,  $Ton$  sera généralement égal à 0.

Exemple : Compteur = 4095 pour 20mS soit 20000 $\mu$ S.

On a un comptage pour  $20000 / 4095$  soit toutes les  $4,884\mu S$ .

**Toff** à indiquer = **Sortie "1"** / 4,884 (Sortie "1" désirée en  $\mu\text{S}$ .)

Exemple : Sortie "1" désirée = 2272μS.

$T_{off} = 2272 / 4,884 = 465 \Rightarrow \text{pwm.setPWM(Num Sortie, 0, 465)};$

### Pilotage en degrés des servomoteurs.

Chaque moteur peut en principe balayer un angle de  $180^\circ$ . Toutefois la dispersion de caractéristique fait que d'un moteur à l'autre la position adoptée n'est pas rigoureusement identique pour un même signal. Par exemple un moteur testé présente les limites suivantes :  $-90^\circ$  pour  $2272\mu S$  soit  $Sup = 465$  valeur nommée  $T_{mini}$ .

+90° pour 667μS soit **Sup** = 136 valeur nommée **Tmaxi**.

L'instruction qui transpose l'angle désiré en durée PWM est :

```
Pulse = map(Angle, -90, +90, Tmini, Tmaxi);
```



## Détermination logicielle de la température. 2/2

De ① on déduit : ③  $1,69 = (a * 40) + b$

De ② on déduit : ④  $2,6 = (a * 19) + b$

Dans l'équation ③ on "isole" le coefficient **a** :

$$\frac{(1,69 - b)}{40} = a \text{ soit encore : } a = \left( \frac{1,69}{40} - \frac{b}{40} \right) \text{ ⑤}$$

Dans l'équation ④ on remplace la constante **a** par sa valeur ⑤ :

$$2,6 = \left( \frac{1,69}{40} * 19 \right) - \left( \frac{b}{40} * 19 \right) + \left( \frac{b * 40}{40} \right) \text{ ⑥}$$

Notons au passage que dans l'équation ⑥ le dernier terme a été "réduit" au même dénominateur commun de valeur 40. On effectue tous les calculs (*Mentalement ou avec une calculatrice !*) et l'on déduit :

**b** = 3,423333... On remplace **b** par sa valeur dans ④ :

$2,6 = (a * 19) + 3,423333$  Équation dans laquelle on "isole" **a** :

$$a = \frac{2,6 - 3,423333}{19} \text{ soit } a = -0,043333...$$

On remplace **a** et **b** par leurs valeurs dans  $T = (a * U) + b$  manipulation se traduisant par :  $T = (-0,043333 * U) + 3,423333$

Enfin, on "réduit" les deux termes au même dénominateur" et l'on obtient l'équation tant recherchée :

$$T = \frac{3,423333 - U}{0,043333}$$

Cette formule est facile à traduire en langage C++ en prenant soin de respecter la priorité des opérateurs avec les parenthèses :

`Temperature = ((3.4233333 - U_Thermistance) / 0.04333333);`

**NOTE** : L'utilisation de la fonction `map()` remplacerait bien cette formule. Ce n'est pas possible, car elle ne supporte pas les **floats**.

### ➤ Mesure de la valeur de `U_Thermistance`.

Seule subtilité, pour tenir compte des caractéristiques exactes de **R1** et de la **CTN**, on introduit la constante **Calibre\_U\_Thermistance** qui est déterminée par des mesures précises à l'aide d'un multimètre.

`#define Calibre_U_Thermistance 4.57` (**A1** : Entree\_thermistance)

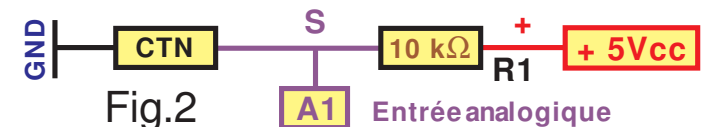
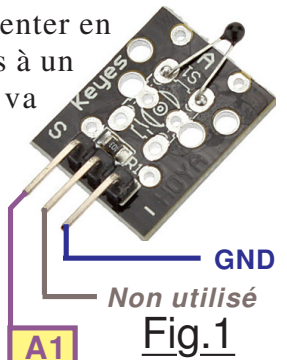
`float Tension_capteur_temperature() {`  
`return analogRead(A1) * Calibre_U_Thermistance / 1023;`

`void Mesurer_la_temperature() {`  
`U_Thermistance = Tension_capteur_temperature();`  
`Temperature = ((3.4233333 - U_Thermistance) / 0.04333333);`

## Mesure de la température du servomoteur.

Précaution indispensable, un dispositif de surveillance de la température du servomoteur d'ouverture et de fermeture de la pince est impératif. En effet, si l'opérateur commande avec '**s**' un écart de serrage trop faible, le servomoteur va tenter en vain d'atteindre la position de consigne. Soumis à un fort courant en permanence, sa température va augmenter jusqu'à sa destruction. Montré sur la Fig.1 on va utiliser un petit circuit imprimé qui utilise un capteur constitué d'une thermistance de type **CTN**, (*Thermistance à Coefficient de Température Négatif.*) très facile à se procurer dans le commerce en ligne. La petite plaque ne comporte que deux éléments : La thermistance et une résistance de **10kΩ**. Ces deux éléments forment un diviseur de tension (*Voir la Fig.2*) qui alimenté avec le +5Vcc de la carte Arduino sera numérisé par l'entrée analogique **A1** de l'ATmega328.

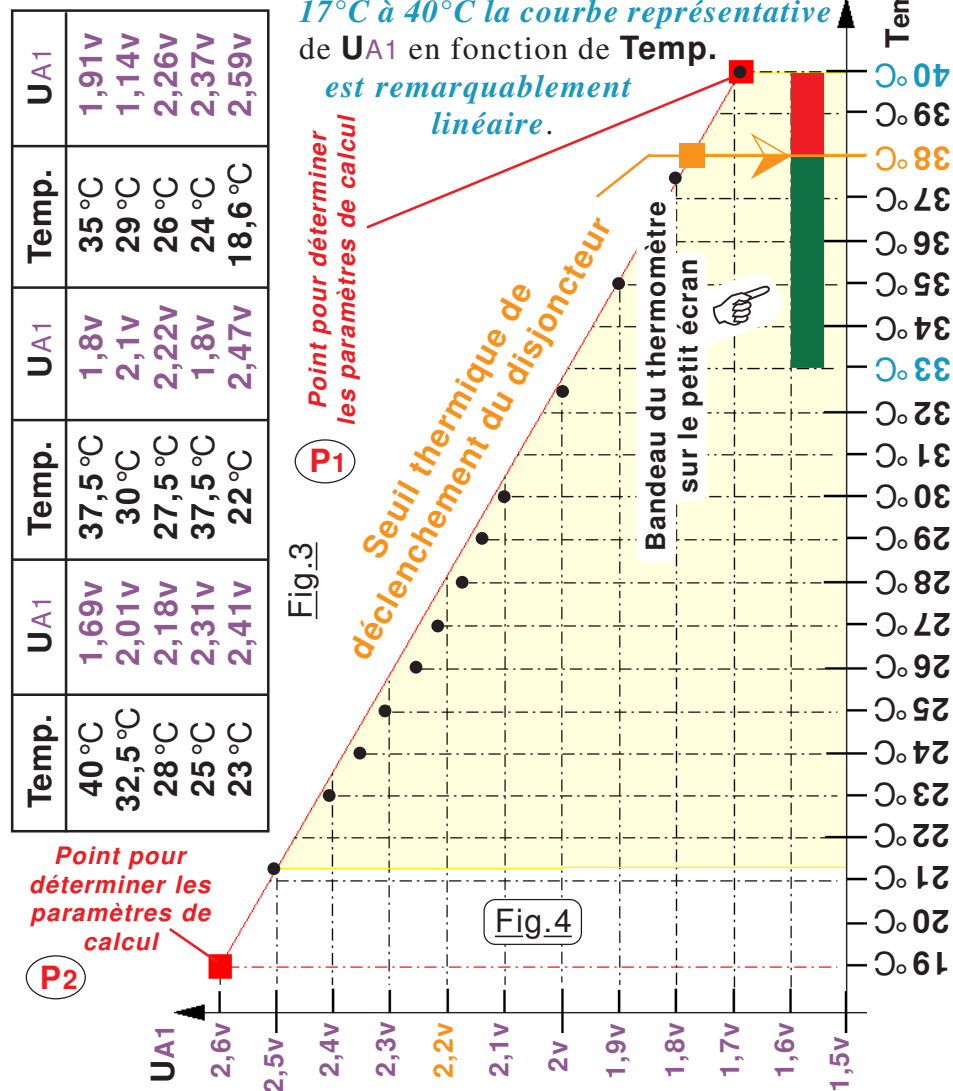
La Fig.2 présente le schéma du petit module et Les branchements prévus par son concepteur. Quand la température augmente, la valeur de **CTN** diminue. Rien n'interdirait d'inverser le + **5Vcc** et **GND** pour obtenir une variation de tension dans le même sens que celui de la température, seuls la valeur du seuil et le sens de comparaison seraient alors à modifier dans la séquence de surveillance.



Autant mesurer la tension en sortie du pont diviseur est aisé avec ARDUINO, autant la convertir en température relève d'un calcul plus délicat. La thermistance n'a pas un comportement linéaire et il importe dans la conversion d'utiliser une fonction de type logarithmique. Pour le projet de bras manipulateur, c'est la solution de la Fig.2 qui est retenue, elle est mise en place tardivement et rajoutée au circuit imprimé principal à un emplacement encore possible. La résistance **R1** du petit module n'est pas utilisée et remplacée par un composant de **10kΩ** directement implanté sur le circuit imprimé *pour n'avoir à relier que deux fils au capteur.*

## Comportement du capteur de mesure thermique.

Matériellement, le **comportement** d'une résistance de type **CTN** en fonction de la température à laquelle est elle soumise est physiquement **logarithmique**. De ce fait, quand on utilise le circuit de la Fig.2, la courbe représentative de la tension devrait être également logarithmique. Le tableau de la Fig.3 fournit les valeurs réellement mesurées, et la Fig.4 montre le graphe de comportement. On constate que **pour une plage de température allant de 17°C à 40°C la courbe représentative de UA1 en fonction de Temp. est remarquablement linéaire**.



## Détermination logicielle de la température. 1/2

Considérons la Fig.4 sur laquelle la loi de variation qui sera prise en compte, tracée en rouge, est constituée d'une droite passant au plus proche du **"nuage des points représentatifs" tracés en noir**. Pour pouvoir afficher sur le petit écran OLED la valeur de la température, il faut déterminer l'algorithme qui sera codé en C++ dans le programme Arduino d'exploitation du bras manipulateur. Concrètement, il faut trouver l'agencement d'une fonction du genre :

**Temperature = fonction de U\_Thermistance;**

Dans cette instruction C++, **Temperature** est l'identificateur du programme qui représentera à l'affichage la valeur de la température, et **U\_Thermistance** la variable préservant le résultat de la mesure de tension en **S** sur **R1 / CTN** mesurée avec **A1**. (Voit Fig.2)

### ➤ Aspect mathématique.

Tout bachelier sait (*En principe !*) qu'un phénomène de type linéaire peut se voir représenté par une équation du premier degré de type  $Y = a * X + b$  dans laquelle **Y** représente ce que l'on cherche, **X** ce que l'on connaît, (*Et oui, c'est bien curieux, car dans la vie de tous les jours X représente une inconnue !*) et **a** et **b** deux constantes. Cette formule devient dans notre cas :  $T = a U + b$  qui codée en C++ devient :

$$T = (a * U) + b$$

Dans cette instruction, l'expression **"fonction de"** devient un produit complété par une somme. Comme il faut définir "la priorité des opérateurs", le produit est confiné entre parenthèses.

### ➤ Recherche des coefficients a et b.

Nous ne pourrions traduire cette formule que si nous connaissons les valeurs des deux constantes **a** et **b**. Nous allons dans ce but choisir sur le graphe de la Fig.4 deux points représentatifs pertinents **P1** et **P2**. Pertinents traduit le fait que l'on exploite **deux mesures "les plus éloignées possibles"** sur le graphe représentatif.

De **P1** on déduit : ①  $40 = (a * 1,69) + b$

De **P2** on déduit : ②  $19 = (a * 2,6) + b$

Mathématiquement, pour trouver les valeurs de **a** et **b** il suffit de résoudre un système de deux équations à deux inconnues. (*Tout bachelier sait que ... OK, je n'insiste pas !*) On tourne la fiche :

## Problème de la boucle d'affichage sur OLED 1,3'.

Programmant des séquences d'affichage "ordinaires" on peut fort bien développer un nombre significatif de projet sans forcément rencontrer ce cas particulier. Quand il se produit, on risque de passer beaucoup de temps à en trouver l'origine car à première vue le code source semble correct. Nous allons illustrer ce propos sur un exemple, celui qui a fait surgir ce cas "étrange" :

Fig.1

```
01 02 03 04 05 06 07 08 09
10 11 12 13 14 15 16 17 18
19 20 21 22 23 24 25 26 27
28 29 30 31 32 33 34 35 36
37 38 39 40 41 42 43 44 45
46 47 48 49 50 51 52 53 54
```

Pour mettre au point une page d'écran, par deux boucles imbriquées je désirais faire afficher la matrice de la Fig.1 avec la séquence de code donnée ci-dessous :

```
void PAGE4() {
  byte K; K=1;
  // Placé ici provoque un affichage aléatoire.
  u8g.firstPage(); do {
    // Début de l'affichage de la page écran.
    K=1; Placé ici l'affichage est correct.
    for (byte Ligne=8; Ligne<65; Ligne=Ligne+10) {
      for (byte X=0; X<120; X=X+14)
        {u8g.setPrintPos(X,Ligne);
         if (K<10) u8g.print('0');
         u8g.print(K); K++;}}}
```

Fig.2

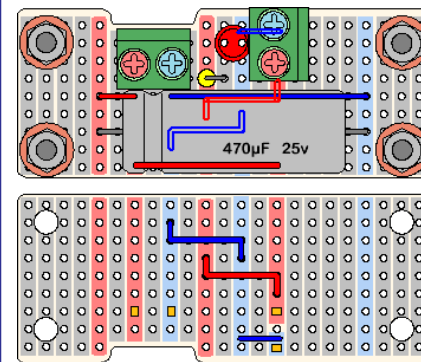
Comme indiqué dans la bibliothèque, pour afficher une page écran on engage une boucle qui est parcourue huit fois pour générer un écran et l'afficher. **De ce fait les huit boucles doivent avoir le même contenu et en particulier les valeurs des variables.**

```
01 02 03 04 05 06 07 08 09
10 11 12 13 14 15 16 17 18
19 20 21 22 23 24 25 26 27
28 29 30 31 32 33 34 35 36
37 38 39 40 41 42 43 44 45
46 47 48 49 50 51 52 53 54
01 02 03 04 05 06 07 08 09
10 11 12 13 14 15 16 17 18
19 20 21 22 23 24 25 26 27
28 29 30 31 32 33 34 35 36
37 38 39 40 41 42 43 44 45
46 47 48 49 50 51 52 53 54
01 02 03 04 05 06 07 08 09
10 11 12 13 14 15 16 17 18
19 20 21 22 23 24 25 26 27
28 29 30 31 32 33 34 35 36
37 38 39 40 41 42 43 44 45
46 47 48 49 50 51 52 53 54
01 02 03 04 05 06 07 08 09
10 11 12 13 14 15 16 17 18
19 20 21 22 23 24 25 26 27
28 29 30 31 32 33 34 35 36
37 38 39 40 41 42 43 44 45
46 47 48 49 50 51 52 53 54
01 02 03 04 05 06 07 08 09
10 11 12 13 14 15 16 17 18
19 20 21 22 23 24 25 26 27
28 29 30 31 32 33 34 35 36
37 38 39 40 41 42 43 44 45
46 47 48 49 50 51 52 53 54
```

Sur la Fig.2 ce que l'on obtient si dans la boucle d'affichage de l'écran OLED on introduit une sortie sur la ligne série de l'IDE

## Alimentation électrique des divers modules.

L'alimentation de la carte Arduino NANO et des petits modules électroniques se fait en **1** par la ligne USB de dialogue série avec le Moniteur de l'IDE. Cette ligne USB fournit l'énergie nécessaire au relais de puissance en **2** et de sa LED locale. Compte tenu des appels de courant "virulents" générés par les servomoteurs, l'alimentation secteur séparée **3** fournit l'énergie de puissance 5v pour l'ensemble de la motorisation. Les fiches en **4** ne doivent être branchées que lorsque la carte Arduino NANO est alimentée par la ligne USB et que le programme d'exploitation a confirmé l'état "0" sur la sortie de pilotage **D13** du relais de puissance. Comme cette



sortie sur la carte Arduino pilote également une LED témoin rouge, nous disposons ainsi d'un contrôle visuel de l'état du relais **2**. Le courant "important" transite par le contact **Travail** du relais de puissance **2**, son contact **Repos** étant inutilisé. Pour ne pas que le courant de pointe à la charge du condensateur "réservoir" ne sollicite les grains du relais **2**, le condensateur de 470µF en **5** est disposé avant le relais. Quand le relais de puissance est piloté par **D13** le servomoteur du **COUDE** en **6** et le servomoteur d'**ÉPAULE** en **7** sont directement alimentés en puissance. En revanche, le servomoteur de **HANCHE** en **8** ne sera opérationnel que si le "strap" à languette (**1**) est mis en place sur le connecteur HE14 du circuit imprimé **9**. Enfin, le servomoteur de **PINCE** **10** ne sera alimenté en puissance que si le relais **ILS** en **11** piloté par la sortie **A3** est au repos. En **12** on retrouve le capteur thermique dont seule la **CTN** est branchée.

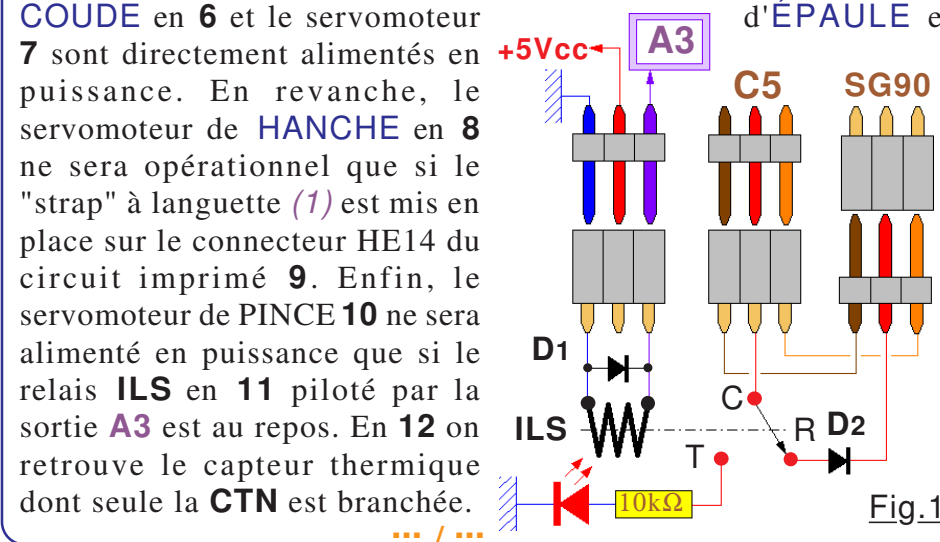
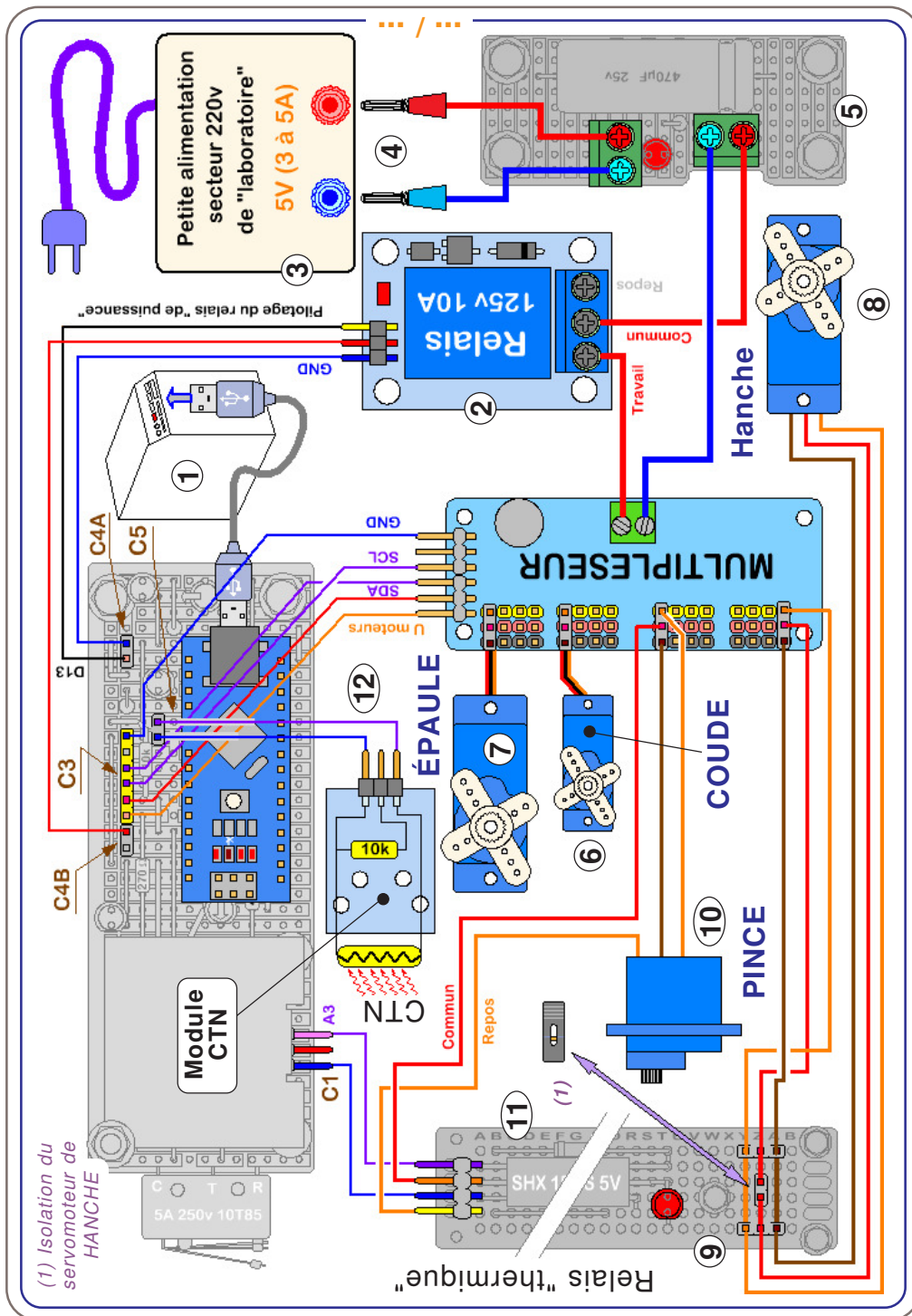


Fig.1





## Paramètres d'adaptation du logiciel au matériel.

Aucun produit industriel n'est strictement identique aux autres d'une même référence. L'imprécision de caractéristiques sera d'autant plus importante que le produit appartient à une classe réputée "grand public". Les servomoteurs équipant le prototype appartiennent à cette catégorie. Les valeurs des consignes imposant le positionnement angulaire sont affectées d'une dispersion importante. Toutefois, comme c'est le programme d'exploitation qui définit les valeurs qui seront envoyées au multiplexeur, il est naturel de compenser toutes ces "dérives" de caractéristiques par logiciel.

Articulation	Position		Consigne	
	Mini	MAXI	Mini	MAXI
PINCE	30°	90°	373	264
COUDE	-40°	99°	224	471
EPAULE	0°	90°	375	215
HANCHE	-90°	+90°	439	89

Fig.1

Le tableau de la Fig.1 résume les caractéristiques particulières des divers servomoteurs équipant le bras manipulateur. Le programme d'exploitation contient en tête de listage une zone bien délimitée montrée sur la Fig.2 pour définir avec des identificateurs personnalisés les valeurs correspondant aux servomoteurs installés.

```
//----- Paramétrage des servomoteurs -----*
#define Pos_m P 30 // Position angulaire minimale pour la PINCE. *
#define Pos_M P 90 // Position angulaire Maximale pour la PINCE. *
#define Csn_m P 373 // Consigne minimale pour la PINCE. *
#define Csn_M P 264 // Consigne Maximale pour la PINCE. *
#define Pos_m C -40 // Position angulaire minimale pour le COUDE. *
#define Pos_M C 99 // Position angulaire Maximale pour le COUDE. *
#define Csn_m C 224 // Consigne minimale pour le COUDE. *
#define Csn_M C 471 // Consigne Maximale pour le COUDE. *
#define Pos_m E 0 // Position angulaire minimale pour l'EPAULE. *
#define Pos_M E 90 // Position angulaire Maximale pour l'EPAULE. *
#define Csn_m E 375 // Consigne minimale pour l'EPAULE. *
#define Csn_M E 215 // Consigne Maximale pour l'EPAULE. *
#define Pos_m H -90 // Position angulaire minimale pour la HANCHE. *
#define Pos_M H 90 // Position angulaire Maximale pour la HANCHE. *
#define Csn_m H 439 // Consigne minimale pour la HANCHE. *
#define Csn_M H 89 // Consigne Maximale pour la HANCHE. *
```

**Note :** La commande '&' qui résume sur la ligne USB toutes ces limites puise directement les *valeurs dans les constantes prédéfinies.*

## Vérifications et validation du circuit principal.

**E**xclure tout risque d'endommager du matériel lors de la validation d'un circuit électronique impose une méthode stricte et un protocole de vérification d'une chronologie rigoureuse.

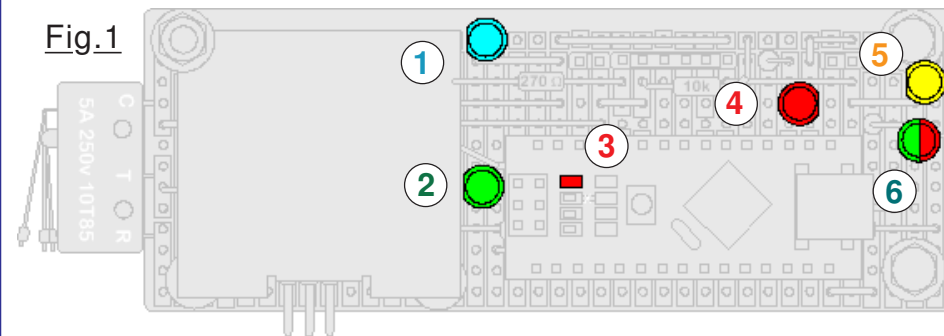
- 1) Circuit imprimé seul : Aucun branchement électrique extérieur.
- 2) Alimenter sur le connecteur HE14 repéré **C1** en **GND** et +5Vcc.
- 3) Vérifier la présence du +5V sur :
  - La broche 5v du microcontrôleur,
  - Le +Vcc du connecteur HE14 en **C2** supportant OLED,
  - Le +Vcc sur la lyre dédiée de **C3** pilotant le multiplexeur,
  - Le coté positif du condensateur de 10µF,
  - L'entrée **D3** de l'ATmega328. *La tension doit chuter à zéro quand on clique sur le micro-contacteur d'arrêt d'URGENCE.*
- 4) Pointe de touche branchée sur du +5V carte alimentée sur **C1** :
  - Ponter sur **D2** du microcontrôleur : Le BUZZER "couine",
  - Toucher **D4** du processeur : La LED bleue s'allume,
  - Tester **D5** du long support HE14 : La LED verte s'allume,
  - Alimenter **D7** du processeur : La LED double rouge s'allume,
  - Ponter **D9** du circuit intégré : La LED double verte s'allume,
  - Vérifier **D12** de l'ATmega328 : La LED orange s'allume,
  - Toucher **D13** : Le pilotage sur **C4A** passe à "1",
  - Tester **A0** du processeur : La LED rouge s'allume,
  - Alimenter **A3** du processeur qui va en broche **A3** sur **C1**,
  - Toucher **A4** du µP : **SDA** sur **C2** et sur **C3** passent à "1",
  - Alimenter **A5** du µP : **SCL** sur **C2** et sur **C3** passent à "1",
  - Ponter **A7** du µP : Le +5V est présent sur "Puissance" de **C3**,
- 5) Alimentation +5v coupée, le multimètre est configuré en ohmmètre, avec l'une des deux touches sur la broche **GND** de **C1** :
  - On retrouve **GND** sur **C2**, **C3**, **C4A** et **C5**, (*Résistance nulle.*)
  - On vérifie la présence de **GND** entre **RST** et **VIN** du µP.
- 6) Carte Arduino "en l'air", téléverser **P01\_Etat\_des\_axes.ino**
- 7) Placer la carte ARDUINO ainsi que l'afficheur OLED sur leurs supports HE14 respectifs. Brancher la ligne série sur l'une des sorties USB du P.C. L'afficheur OLED doit afficher des valeurs fictives pour les articulations, et l'item **Ouvert / Fermé** doit alterner à une cadence d'environ deux secondes.

... / ...

## Signification des LEDS du C.I. principal.

- La LED bleue en **1** se met à *clignoter* quand on *arme avec ':'* la possibilité de réaliser *la séquence d'APPRENTISSAGE* actuelle en EEPROM avec '**E**'. Toute autre commande annule le clignotement.
- Toujours en **1**, elle s'illumine en continu pendant l'exécution d'une séquence de programme '**EN**' ou d'un APPRENTISSAGE '**E**'.
- La sortie du mode APPRENTISSAGE peut dans certaines circonstances laisser la LED bleue en **1** allumée. *Il suffit d'imposer à nouveau l'angle actuel sur l'une des articulations pour l'éteindre.*
- La LED verte en **2** se met à clignoter accompagnant la commande '**>**' tant que le *mode APPRENTISSAGE est actif*.

Fig.1



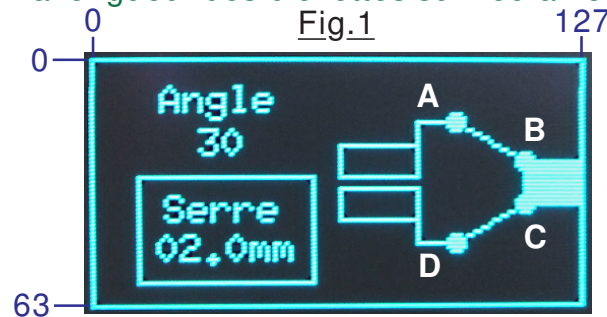
- La LED rouge miniature sur la carte Arduino NANO en **3** est pilotée par **D13** et s'allume lorsque elle fait passer au **Travail** le relais des énergies de puissance sur la motorisation. C'est le témoin de la présence d'énergie sur les servomoteurs.
- La LED **4** rouge s'allume en continu quand le bouton d'arrêt en URGENCE a été sollicité. Elle *clignote* pour avertir l'opérateur quand le *pilotage en CONSIGNES* est engagé.
- En **5** la LED orange sera *allumée* quand *les servomoteurs sont au maximum de vitesse* car le ralentissement '**t**' est suspendu avec '**l**'.
- La LED double **6** s'illumine en *vert* lorsque *les consignes ne sont plus envoyées aux servomoteurs* qui restent figés en position car ils sont neutralisés. La LED passe au rouge quand la commande '**n**' annule la *neutralisation*. (*Erreur n°17 si '\*' est positionné.*)
- La LED *rouge du petit circuit imprimé* supportant le relais ILS indique la *coupure énergie sur la PINCE* suite à une surchauffe. Température redevenue normale, pour réarmer, il faut envoyer '**\***'.

## Affichage graphique de la configuration PINCE.

Bien que le symbole ne soit pas à l'échelle des dimensions pour des raisons de manque de définition sur la matrice de points, on respecte avec rigueur l'orientation des deux bielles. Les deux articulations **B** et **C** ont des centres de positions invariantes sur l'écran. Pour tracer les deux autres extrémités **A** et **D** il faut en déterminer la position dans la mosaïque des luminophores. (Les doigts seront également représentés à partir des points **A** et **D**.) Pour calculer les coordonnées de **A** et **D** il faut faire appel aux notions trigonométrie et l'on aboutit à la formule :

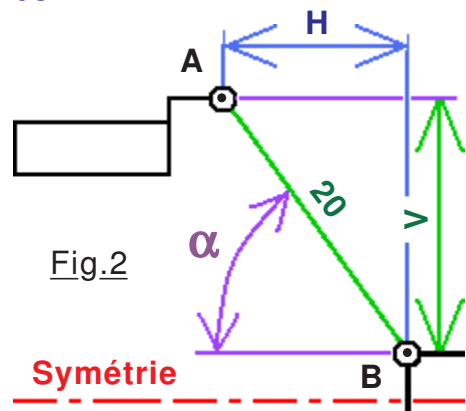
$$V = AB * \sin(\alpha) \text{ et } H = AB * \cos(\alpha)$$

La longueur des biellettes sur l'écran est choisie à 20 PIXELs.



Sont précisées en bleu les coordonnées des PIXELs écran sur la mosaïque graphique.

B : x = 112 , y = 26  
C : x = 112 , y = 37



L'image est symétrique par rapport à l'axe horizontal.

**AB** représente la longueur graphique des diverses biellettes exprimée en PIXELs, on doit donc remplacer par la valeur **20**. Enfin, pour les points **A** et **D** il faut calculer à partir du point **B** et du point **C**. Pour le point **A** par exemple on obtient :

$$x_A = x_B - H = x_B - AB * \cos(\alpha) = 112 - 20 * \cos(\alpha)$$

$$y_A = y_B - V = y_B - AB * \sin(\alpha) = 26 - 20 * \sin(\alpha)$$

$$\text{Position} = 112 - \text{abs}(20 * (\cos(\text{radians}(\text{Angle\_Serrage\_Pince}))));$$

$$\text{Hauteur} = 26 - \text{abs}(20 * (\sin(\text{radians}(\text{Angle\_Serrage\_Pince}))));$$

08) Téléverser **P02\_Pilotage\_Moniteur\_USB.ino**. @

Envoyer des commandes conformément aux instructions figurant sous forme de remarques. Provoquez les trois erreurs possibles ce qui validera le dialogue USB et le bon branchement du BUZZER.

09) Téléverser **P03A\_Lister\_EEPROM\_en\_HEXAdecimal**. @

Le contenu de l'EEPROM de l'ATmega328 utilisé ne contiendra que des \$FF si elle est vierge, ou des \$00 si la ligne 14 est validée.

10) Téléverser **P04\_Pilotage\_USB\_et\_mémorisation\_EEPROM**. @

Tester les commandes et provoquez toutes les erreurs possibles précisées en commentaires en tête de listage du programme.

11) Téléverser **P05\_Test\_de\_base\_du\_Multiplexeur**.

- Brancher un potentiomètre de 4,7kΩ à 10kΩ entre **GND** et **A3** sur le connecteur **C1**. En faisant varier la pleine plage possible sur le potentiomètre la consigne doit varier entre ≈120 et ≈520.
- Positionner le potentiomètre pour obtenir une consigne de ≈254.

12) Brancher la ligne qui va de **C3** au multiplexeur. Quand on réalimente la carte NANO par sa ligne USB, OLED doit afficher correctement sa page écran et la LED rouge du +5Vcc du multiplexeur doit s'allumer.

13) Brancher un ou plusieurs servomoteurs sur des sorties quelconques de multiplexeur PCA9685. Relier une alimentation de puissance 5Vcc sur le bornier de puissance de la carte PCA9685. En faisant varier la tension sur **A3** tous les servomoteurs doivent tourner et prendre des orientations similaires. Pour conditionner la suite **tester principalement les sorties S0, S4, S8 et S15** du multiplexeur.

À ce stade on a validé la compatibilité de OLED avec le PCA9685.

14) Enlever le potentiomètre devenu inutile. Téléverser le logiciel de dialogue **P06\_Pilotage\_USB\_reel\_des\_moteurs.ino** et répartir les servomoteurs sur les lignes affectées **S0, S4, S8 et S15**. Toutes les instructions pour valider l'ensemble figurent en tête de programme.

15) Téléverser enfin le programme **P07\_Programme\_Complet.ino** et brancher sur **C5** la résistance **CTN** et sur **C2** le petit module de protection thermique. Avec la commande 'j' faire afficher la **Page8** et avec un sèche-cheveux, chauffer le module de la thermistance jusqu'à provoquer la disjonction pour tester la protection automatique.

16) Intercaler le module "relais de puissance" et terminer la validation globale des systèmes en respectant rigoureusement la fiche nommée

**Protocole de mise en service du bras.**

@ : Aucun connecteur branché sauf **C2**.