




## TABLE DES MATIÈRES

<b>Mise en service de la machine.</b>	P 0 1
<b>01) Les options lors de la mise sous tension.</b>	P 0 2
<b>Le listage des commandes.</b>	P 0 2
<b>Les "informations machine" sur RESET.</b>	P 0 3
<b>02) Les fonctions du Menu de base.</b>	P 0 4
<b>Les fonctions Statistiques et RAZ.</b>	P 0 4
<b>La fonction Mode MANUEL.</b>	P 0 5
<b>03) Le comportement en mode EXPLOITATION.</b>	P 0 6
<b>Utilisation du bouton poussoir TIR.</b>	P 0 6
<b>Effet du bouton poussoir du codeur rotatif.</b>	P 0 7
<b>L'utilisation du  en EXPLOITATION.</b>	P 0 7
<b>Comportement thermique de la machine.</b>	P 0 8
<b>04) La machine simplifiée.</b>	P 0 9
<b>05) Affectation des E/S de la carte Arduino.</b>	P 1 0
<b>Circuit imprimé de gestion de la machine.</b>	P 1 2
<b>06) L'interface de puissance.</b>	P 1 4
<b>Circuit imprimé de l'interface de puissance.</b>	P 1 5
<b>07) La gestion des alimentations.</b>	P 1 6
<b>08) Le plan de câblage.</b>	P 1 8
<b>09) Consommation électrique sur le secteur.</b>	P 1 9
<b>10) Implantation des données en EEPROM.</b>	P 2 0
<b>Usure en écriture de la mémoire EEPROM.</b>	P 2 0
<b>Stratégie de traitement des données en EEPROM.</b>	P 2 0
<b>11) Implantation des textes en EEPROM.</b>	P 2 2

**Note :** Le symbole  indique qu'il faut cliquer sur le bouton poussoir central du codeur incrémental. Le dessin  pour son compte précise qu'il faut tourner le bouton de l'axe rotatif du codeur incrémental.

## Cyclotron collisionneur.

### ➤ Mise en service rapide de la machine.

**P**remière opération impérative pour avoir l'énergie de puissance, passer l'inverseur de sécurité en mode NON programmation, c'est à dire verrouillé dans la configuration de la Fig.1 **et surtout pas de cordon branché sur la petite prise mini USB de dialogue série.**




- 1) Brancher le module secteur ou du **16,5Vcc** sur les douilles bananes.
- 2) Vérifier que le module régulateur secondaire soit ajusté à **≈8.0** volts.
- 3) L'écran affiché est celui du **MENU** de BASE indexé sur la fonction **EXPLOITATION**.
- 4) Cliquer sur  pour valider le fonctionnement standard.
- 5) Éventuellement utiliser le BP **TIR** ou le  pour conditionner les collisions ou le mode **SILENCE**.



Fig.1

### ➤ Protocole d'activation d'un cycle.

**M**achine en configuration **ATTENTE** de Dem. dans le mode **EXPLOITATION**. Au préalable l'option **TIR validé** a été positionnée à **OUI** ou **NON** à convenance par usage du BP **TIR**. L'option **SILENCE** a également été initialisée par usage du . Présentant l'affichage de la Fig.2 la machine est en attente d'une consigne opérateur, et la LED bleue est allumée. **Le protocole d'activation d'un cycle avec ou sans collision est le suivant :**

- 1) Introduire une bille dans le tore.
- 2) Placer la goupille de retenue.
- 3) Introduire une bille projectile dans le réceptacle du collisionneur.
- 4) Que l'option **TIR** soit validée ou non retirer la goupille, la collision sera gérée par le logiciel.
- 5) Déclencher le cycle d'accélération avec le BP **Dem**.

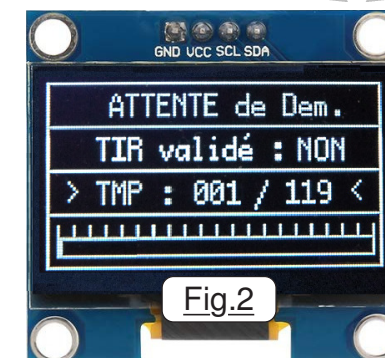
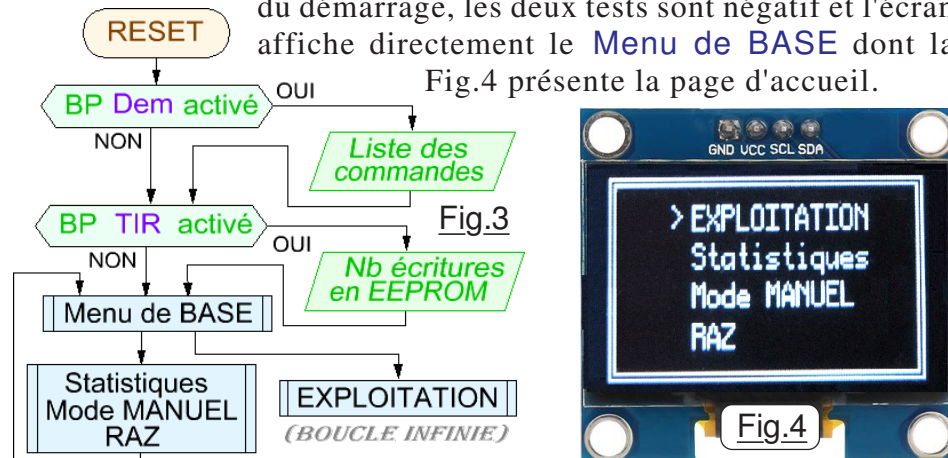


Fig.2

## 01) Les options lors de la mise sous tension.

Illustré sur l'organigramme de la Fig.3 la mise sous tension ou le comportement sur un RESET manuel présente trois possibilités. Si aucun des deux boutons poussoir n'est activé lors du démarrage, les deux tests sont négatif et l'écran affiche directement le **Menu de BASE** dont la Fig.4 présente la page d'accueil.



C'est l'item **EXPLOITATION** qui est alors indexé, et cliquer sur fait passer directement à la configuration **ATTENTE de Dem.** de la machine en vue de déclencher des accélérations de particule.

### ► Le listage des commandes.

Obtenu si au démarrage du programme on maintient le bouton **Dem.** activé, la page de la Fig.5 est affichée sur le petit écran OLED. Ce type de page-écran résume les commandes utilisables en fonction de l'état du système. Bien que laconiques, ces écrans seront généralement suffisants pour ne pas avoir à se plonger trop souvent dans cette notice d'utilisation. Lors du redémarrage dans **Ce MENU**

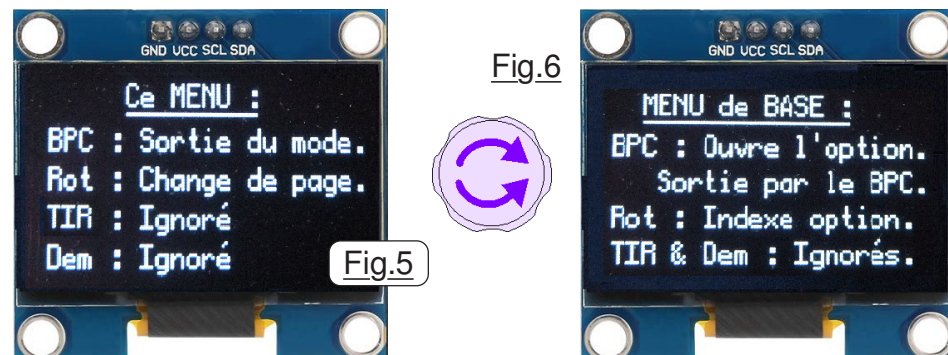


Fig.6

Fig.5

**Le logiciel P09\_EXPLOITATION\_de\_la\_machine.ino fait appel à ces données.** Aussi, pour qu'il puisse se dérouler correctement et ne pas afficher des incohérences, il faut au préalable téléverser **P00\_Initialiser\_EEPROM.ino** et surtout l'activer.

Cet outil force toute la zone EEPROM à zéro, les données numériques listées en bleu page P20 et P22 seront donc annulées.

ADRS	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
0000	E	f	f	a	c	e	r	l	e	s	v	a	l	e	u	
0016	r	s	e	n	E	E	P	R	O	M	:	>	>	>		
0032	O	U	I	N	O	N	M	o	d	e	M	A	N	U		
0048	E	L	I	n	d	u	c	t	e	u	r	à	t	e		
0064	s	t	e	r	:	N	b	C	y	c	l	e	s			
0080	=	N	b	T	I	R	s	=	R	e	v					
0096	P	G	M	d	u	E	X	P	L	O	I	T	A	T		
0112	I	O	N	S	t	a	t	i	s	t	i	q	u	e	s	M
0128	o	d	e	M	A	N	U	E	L	R	A	Z	R	E	F	
0144	R	O	I	D	I	S	S	E	M	E	N	T	A	T	T	E
0160	N	T	E	d	e	D	e	m	.	T	I	R	v			
0176	a	l	i	d	é	:		>	T	M	P	:				
0192	/	1	1	9				<	P	W	:					
0208	C	e	M	E	N	U	:	B	P	C	:	S				
0224	o	r	t	i	e	d	u	m	o	d	e	.	R	o		
0240	t	:	C	h	a	n	g	e	d	e	p	a				
0256	g	e	.	T	I	R	:	I	g	n	o	r	é	E		
0272	X	P	L	O	I	T	A	T	I	O	N	:	B	P	C	
0288	:	M	o	d	e	S	I	L	E	N	C	E	.			
0304	R	o	t	:	P	a	g	e	s	é	c	r	a			
0320	n	.	T	I	R	:	T	I	R	>	O	U				
0336	I	N	O	N	.	D	e	m	:	D	é	b	u			
0352	t	e	c	y	c	l	e	.	M	E	N	U	d	e		
0368	B	A	S	E	:	B	P	C	:	O	u	v				
0384	r	e	l	'	o	p	t	i	o	n	.	S	o	r	t	
0400	i	e	p	a	r	l	e	B	P	C	R	o	t			
0416	:	I	n	d	e	x	e	o	p	t	i	o	n			
0432	.	T	I	R	&	D	e	m	:	I	g	n				
0448	o	r	é	s	.	D	e	p	u	i	s	l	e	R		
0464	E	S	E	T	C	y	c	l	e	s	T	I	R	s		
0480	S	I	L	E	N	C	E	:	E	n	e	r	g			
0496	i	e	c	o	n	s	o	m	m	é	e	C	y	c	l	
0512	e	e	n	c	o	u	r	s	.	.	.	.	.	.	.	.

Fig.35

La Fig.34 reprend une partie de l'affichage qui s'effectue sur le Moniteur de l'IDE lorsque le programme d'écriture des textes est validé. La Fig.35 en présente un listage sous forme de matrice.



## 11) Implantation des textes en EEPROM

L'optimisation en taille d'un programme passe par le logement en mémoire non volatile EEPROM des nombreux textes qui sont générés pour animer le dialogue Homme / Machine d'un ensemble géré par une carte Arduino. Si le système est relativement complexe, les messages textuels seront nombreux. L'un des moyens les plus efficace pour minimiser le code binaire généré par le compilateur C++ consiste quand c'est possible à ranger les chaînes de textes en EEPROM. Ce ne sera envisageable que si les données préservées en permanence en EEPROM pour le logiciel d'exploitation laissent de la place.

Dans notre cas les données à mémoriser sont négligeables :

- Nombre d'écritures effectuées en EEPROM : Octet 1012 à 1015.
- Energie totale consommée : Octet 1016 à 1019.
- Nombre de TIRs effectués : Octet 1020 et 1021.
- Nombre de cycles d'accélération effectués : Octet 1022 et 1023.

Il est donc possible d'inscrire l'intégralité des textes affichés sur l'écran OLED en mémoire EEPROM.

```
PTR = 0 Effacer les
PTR = 11 valeurs
PTR = 18 en EEPROM :
PTR = 29 >>>
PTR = 33 OUI
PTR = 36 NON
PTR = 39 Mode MANUEL
PTR = 50 Inducteur à
PTR = 61 tester :
PTR = 70 Nb Cycles =
PTR = 82 Nb TIRs =
PTR = 92 PGM du
PTR = 99 Wh.
PTR = 103 EXPLOITATION
PTR = 115 Statistiques
PTR = 127 Mode MANUEL
PTR = 138 RAZ
PTR = 141 REFROIDISSEMENT
PTR = 156 ATTENTE de Dem.
PTR = 171 TIR validé
PTR = 181 : > TMP : / 119 <
PTR = 203 PW :
PTR = 208 Ce MENU :
PTR = 217 BPC : Sortie du mode.
PTR = 238 Rot : Change de page.
PTR = 259 Tir : Ignoré
PTR = 271 EXPLOITATION :
PTR = 285 BPC : Mode SILENCE.
PTR = 304 Rot : Pages
PTR = 316 écran.
PTR = 322 TIR : TIR > OUI NON.
PTR = 342 Dem : Dé
PTR = 350 bute cycle.
PTR = 361 MENU de BASE :
PTR = 375 BPC : Ouvre l'option.
PTR = 396 Sortie par le BPC
PTR = 413 Rot : Indexe option.
PTR = 433 TIR & Dem : Ignorés.
PTR = 453 Depuis le RESET
PTR = 468 Cycles
PTR = 475 TIRs
PTR = 480 SILENCE :
PTR = 491 Energie consommée
PTR = 508 Cycle en cours.
```

Fig.34

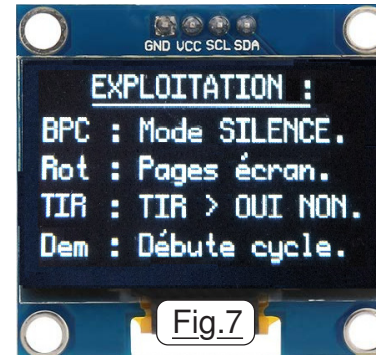




Fig.7

les deux boutons poussoir seront royalement ignorés. C'est la **Rotation** du codeur incrémental  qui va faire circuler les pages d'informations Fig.5 à Fig.7 sur l'écran graphique. C'est  qui fait sortir le programme de ce mode pour "brancher" le traitement sur la page de la Fig.4, c'est à dire dans le **Menu de BASE**.

### ➤ Les "informations machine" sur RESET.

Lorsqu'au démarrage du programme on maintient le bouton **TIR** activé, le taux d'usure de la mémoire EEPROM qui stocke les valeurs des données de la machine montré en Fig.8 est présenté sur le petit écran. Chaque échantillonnage toutes les trois secondes mettant à jour la quantité d'énergie absorbée sur le secteur alternatif donne lieu à une écriture en EEPROM. Toutes ces écritures ainsi que celles durant l'activation des cycles d'accélération sont comptabilisés. L'écran de la Fig.8 indique le nombre total de mises à jour des données logées en EEPROM. *(Ce nombre est incrémenté trois fois pas secondes.)*






Fig.8

**Conduite à tenir** quand le nombre d'écritures en EEPROM atteint les 100000 considérés comme critiques : Il faut déplacer la zone des données en EEPROM pour pointer des **cellules inutilisées qui de ce fait conservent intégralement leur potentiel**. Dans ce but on passera en revue toutes les instructions "EEPROM" de type `eeeprom_read_word(...)`, `eeeprom_read_float(...)` etc en changeant les valeurs des pointeurs comme proposé ci-dessous :

- Les octet 1012 à 1015 sont déplacés en 1000 à 1003.
- Les octet 1016 à 1019 sont déplacés en 1004 à 1007.
- Les octet 1020 et 1021 sont déplacés en 1008 et 1009.
- Les octet 1022 et 1023 sont déplacés en 1010 et 1011.

Voir complément en page P20.

## 02) Les fonctions du Menu de Base.

L'entrée dans le **Menu de Base** de la Fig.4 fait passer directement en **EXPLOITATION** de la machine si on clique sur . La rotation du codeur incrémental  dans un sens ou dans l'autre déplace le l'index vers le bas ou vers le haut. C'est en cliquant sur  que l'on valide l'option. C'est également sur ce bouton central du codeur incrémental que l'on sort de la fonction invoquée pour revenir au **Menu de Base**.

### ➤ Les fonctions Statistiques et RAZ.

L'affichage de l'historique de la machine présenté sur la Fig.9 indique les valeurs des données depuis la mise en service de la machine sauf si les compteurs sont ultérieurement remis à zéro par utilisation de la fonction **RAZ**. Tous les cycles déclenchés avec **Dem.** seront comptés. Dans l'exemple il y en a eu 43. Les processus d'accélération déclenchés sans la validation du collisionneur sont comptabilisés à part. Dans l'exemple de la Fig.9 il n'y a eu que 35 collisions, c'est à dire huit accélérations "à vide". Lorsque le logiciel est en mode **EXPLOITATION**,

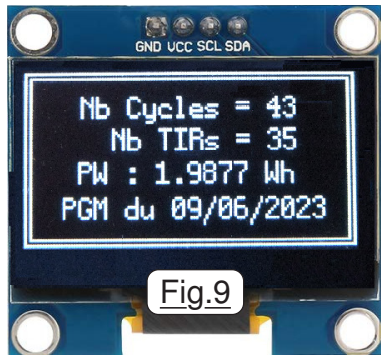


Fig.9

le programme mesure en permanence la consommation de la machine sur le secteur alternatif 220V. (Voir détails en Page 19.) À partir de mesures précises le logiciel évalue l'énergie consommée **PW** et la mémorise en permanence en mémoire EEPROM. Toutefois, l'utilisateur peut décider sur des critères qui lui sont propres de remettre à zéro ces valeurs enregistrées avec la fonction **RAZ** dont la Fig.10 présente la page écran. C'est

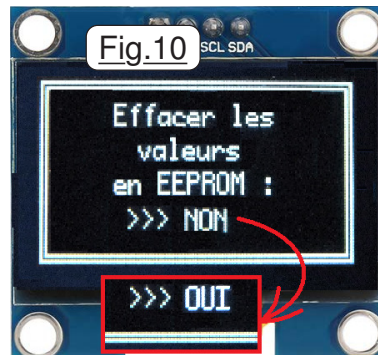


Fig.10

le codeur rotatif  qui fait alterner **OUI / NON** et  qui entérine la valeur de l'option. **ATTENTION : L'effacement si l'option OUI est active ne demande pas de confirmation !**

à jour à partir des emplacements de sauvegarde en l'EEPROM :

• Dans **void setup()** :

```
Nb_TIRs = eeprom_read_word((unsigned int *) 1020);
```

```
Nb_Cycles = eeprom_read_word((unsigned int *) 1022);
```

```
Energie_totale = eeprom_read_float((float *)1016);
```

```
NB_Ecritures_EEPROM = eeprom_read_dword((unsigned long *) 1012);
```

Ensuite les variables RAM sont traitées en fonction des besoins et affichées sur l'écran OLED. Chaque fois qu'une variable est modifiée en RAM, son clone en EEPROM est immédiatement mis à jour en vue de sa récupération au prochain redémarrage.

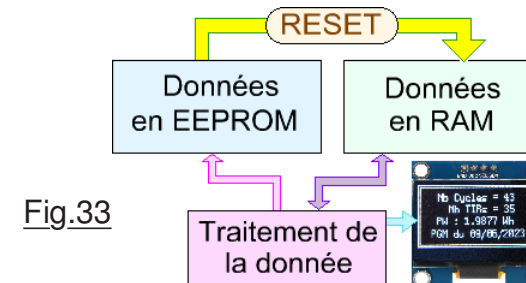


Fig.33

Les instructions concernées sont les suivantes :

• Dans **void RAZ()** :

```
eeprom_write_word((unsigned int *) 1020, 0);
```

```
eeprom_write_word((unsigned int *) 1022, 0);
```

```
eeprom_write_float((float *)1016,0.0);
```

• Dans **void Incremente\_Energie\_toutes\_les\_trois\_secondes()** :

```
Energie_totale = (eeprom_read_float((float *)1016)) + 0.0011;
```

```
eeprom_write_float((float *)1016,Energie_totale);
```

```
eeprom_write_dword((unsigned long *)1012,NB_Ecritures_EEPROM);
```

• Dans **void Traiter\_un\_cycle()** :

```
Energie_totale = (eeprom_read_float((float *)1016)) + 0.0352;
```

```
eeprom_write_float((float *)1016,Energie_totale);
```

```
eeprom_write_dword((unsigned long *)1012,NB_Ecritures_EEPROM);
```

```
eeprom_write_word((unsigned int *) 1020,Nb_TIRs);
```

```
eeprom_write_word((unsigned int *) 1022,Nb_Cycles);
```

**NOTE :** Les instructions pour réaliser les échanges entre RAM et EEPROM utilisent la bibliothèque **avr/eeprom.h** qu'il n'y a pas besoin de déclarer avec un **include** et qui remplace avantageusement **EEPROM.h** car optimisée et travaille avec des formats quelconques.



## 10) Implantation des données en EEPROM

Seulement douze octets sont réservés en mémoire EEPROM car dans le logiciel il y a peut de paramètres à mémoriser en mémoire non volatile. Ces douze octets occupent les derniers emplacements de la mémoire et sont implantés en :

- Nombre d'écritures effectuées en EEPROM : 1012 à 1015.
- Énergie totale consommée par la machine : 1016 à 1019.
- Nombre de TIRs effectués : 1020 et 1021.
- Nombre de cycles d'accélération effectués : 1022 et 1023.

Quand on plante en mémoire EEPROM les textes utilisés par le logiciel, ces quatre paramètres sont remis à zéro. Logiquement ils préservent l'historique de la machine depuis sa première mise en service. Toutefois, il reste possible à tout moment de les annuler par la procédure indiquée en Fig.10 de la page P4.

### ➤ Usure en écriture de la mémoire EEPROM.

Déjà abordé en page P3 la durée de vie en écriture des cellule de la mémoire EEPROM n'est garantie que pour environ 100000 écritures. C'est la raison pour laquelle ces opérations sont comptabilisées. Lorsque le compteur de la Fig.8 avoisine la valeur critique, il suffit de déplacer les vecteurs de sauvegarde de douze emplacements ce que montre la Fig.32 donnée ci-dessous.

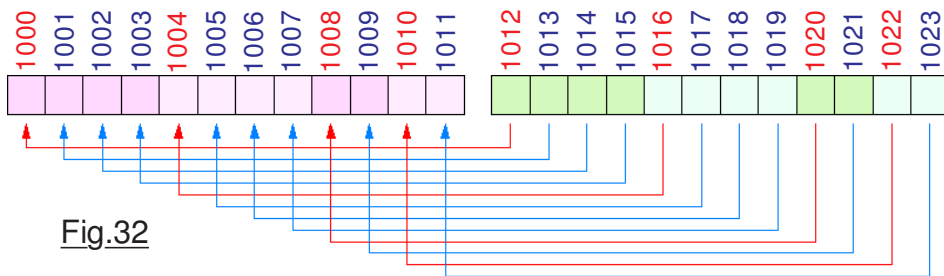


Fig.32

### ➤ Stratégie d'exploitation des données en EEPROM.

Pour des raisons de simplification de l'étude logicielle et surtout de lisibilité du programme, les données ne sont pas directement traitées en EEPROM mais transitent par des variables logées en RAM. Ce sont ces variables qui sont traitées dans les calculs, affichées sur l'écran graphique, puis inscrites en EEPROM. Comme imagé sur la Fig.33 au moment du RESET les variables RAM sont mises

### ➤ La fonction Mode MANUEL.

Simple outil pour la mise au point ou la maintenance, quand on valide l'item **Mode MANUEL** de la Fig.4 on active une fonction secondaire qui débute par un écran qui ressemble à celui de la Fig11 sauf que le chiffre affiché dans le petit cercle est le ①. Ce chiffre désigne le numéro de l'inducteur sur lequel on va agir. On peut incrémenter ou décrémenter à convenance cette valeur en

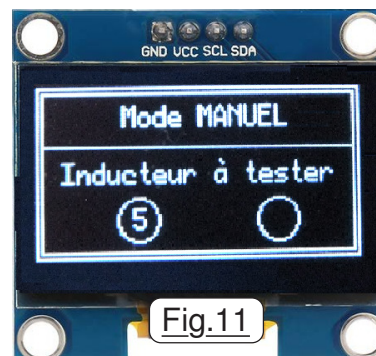


Fig.11

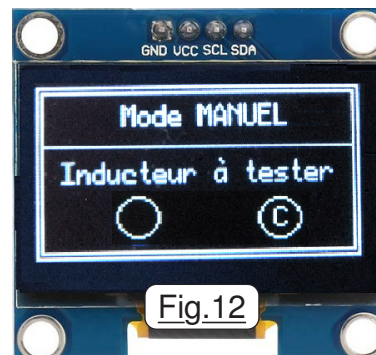


Fig.12

tournant le bouton ⌚ du codeur rotatif. Il y a recyclage des valeurs quand on arrive aux "butées logicielles". La grandeur ne passera pas directement de 8 à 1 ou de 1 à 8 et transitera par l'information **C** de l'écran Fig.12 qui fait allusion au Collisionneur. On peut par cette procédure désigner l'un quelconque des neuf bobinages électromagnétiques. Lorsque c'est un chiffre qui est visualisé, le BP **TIR** génère un BIP sonore d'avertissement d'erreur. L'activation au contraire de **Dem.** déclenche l'alimentation en puissance de l'inducteur désiré. Si c'est **C** qui est affiché dans le petit cercle, le BP **Dem.**

génèrera le BIP sonore d'erreur alors que l'activation du BP **TIR** provoquera l'alimentation en puissance de l'inducteur du collisionneur. *Dans les deux cas la durée de l'impulsion fait 2 secondes.* La LED bleue s'allume si c'est un inducteur accélérateur. C'est la verte qui s'allume pour le collisionneur.

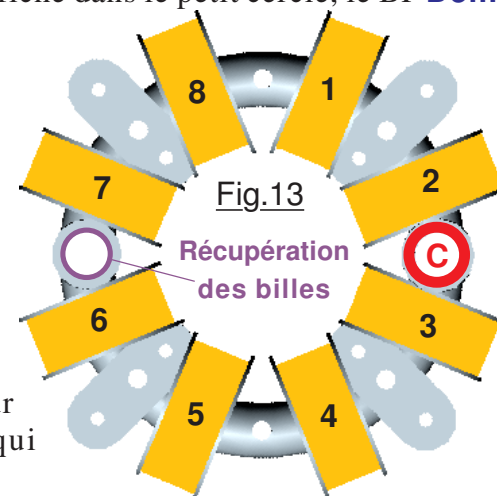


Fig.13

### 03) Le comportement en mode EXPLOITATION.

Suite à la procédure de la Fig.1 la sortie du **Menu de BASE** ouvre l'écran de la Fig.14 de la fonction **EXPLOITATION**. Comme le précise l'organigramme de la Fig.3 mis à part le déclenchement d'un RESET, il n'y a pas possibilité de sortir de ce mode de fonctionnement qui "tourne" en boucle infinie. En arrivant dans ce mode, la LED bleue est allumée signalant que le programme

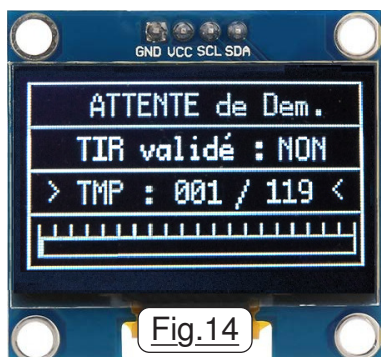


Fig.14

attend une action sur le BP **Dem**. Comme le BP **TIR** n'a pas encore été cliqué, l'option de collision est à **NON**. Toute action sur ce BP **TIR** inversera la valeur de cette option. L'écran "primaire" affiche également la valeur de la température des bobinages des inducteurs. Chaque déclenchement d'un cycle va en augmenter leur température. La valeur, qui machine

froide débute à la grandeur 1, peut augmenter jusqu'au maximum "toléré" de 119. Le comportement thermique est décrit en page **P8**.

#### ► Utilisation du bouton poussoir **TIR**.

C'est une simple bascule de type OUI / NON pour l'armement du déclenchement d'une collision qui se produit alors à la fin d'un cycle d'accélération. Quand on clique sur le bouton poussoir **TIR** le changement est assez discret. Le seul effet immédiat est de changer la valeur de l'affichage de la ligne d'état **TIR** qui sur la Fig.15 est mise en évidence dans le médaillon rouge. Si l'état de ce paramètre est **OUI**, en activation de **Dem**, alors la fin d'un cycle se terminera par l'alimentation de l'inducteur durant 500mS et si le bruiteur est validé un son spécifique sera émis durant cet intervalle de temps. Si le collisionneur a été chargé avec une bille, cette dernière sera lâchée comme projectile dans le tore de circulation de la bille cible.

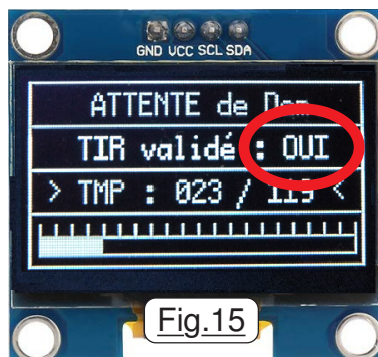


Fig.15

### 09) Consommation électrique sur le secteur.

Quel que soit le mode de fonctionnement en cours, la consommation électrique est mesurée en permanence. La grandeur de cette énergie absorbée sur le secteur électrique alternatif est augmentée de 0,0352 Wh à chaque cycle déclenché par l'opérateur. Lorsque la machine est en attente d'une commande opérateur, la valeur de la puissance est continuellement mise à jour avec un échantillonnage toutes les trois secondes engendrant une incrémentation de 0,0011Wh. Le tableau de la Fig.31 précise les valeurs des intensités consommées pour chaque configuration envisageable sur la machine. Dans le haut du tableau les intensités sont faibles, car seul Arduino et ses périphériques dissipent une énergie assez dérisoire. Pour simplifier le traitement du logiciel, c'est la consommation la plus élevée de 6,3mA qui est prise en compte dans les calculs. La durée d'une impulsion sur le bobinage du collisionneur n'étant que d'une demi-seconde, l'énergie absorbée lors d'un déclenchement est de  $21.12 / 2 / 3600 = 0.0029\text{Wh}$ . Comme elle reste faible, à peine neuf fois la consommation d'Arduino et ce durant une demi seconde, ce prélèvement énergétique relativement dérisoire a été négligé. Il suffirait de rajouter cette valeur à celle calculée dans la procédure `void Traiter_un_cycle()` mais il faudrait alors faire la distinction entre le programme "officiel" et celui du prototype où l'enroulement du collisionneur fait  $240\Omega$  au lieu de  $6\Omega$ .

VEILLE avec LED bleue	5,8mA.	Fig.31
Idem + régulateur allumé	6mA.	
Ajout de la LED verte	6,2mA.	
Ajout de l'affichage OLED	6,3mA Total :	

Durant un cycle avec énergie 16,5V l'intensité est de 96mA soit 21,12W de puissance instantanée.

Un cycle dure 6S.

Donc 0,0352 Wh sont consommés par cycle. (1)

En VEILLE l'échantillonnage est effectué toutes les 3 secondes soit 0,0011 Wh par échantillonnage. (2)

(1) Calcul :  $21,32 * 6 / 3600$

(2) Calcul :  $1,32 * 3 / 3600$



## 08) Le plan de câblage.

En **C** se trouve l'inverseur de sécurité avec bloqueur qui coupe l'alimentation de Vin en mode programmation.

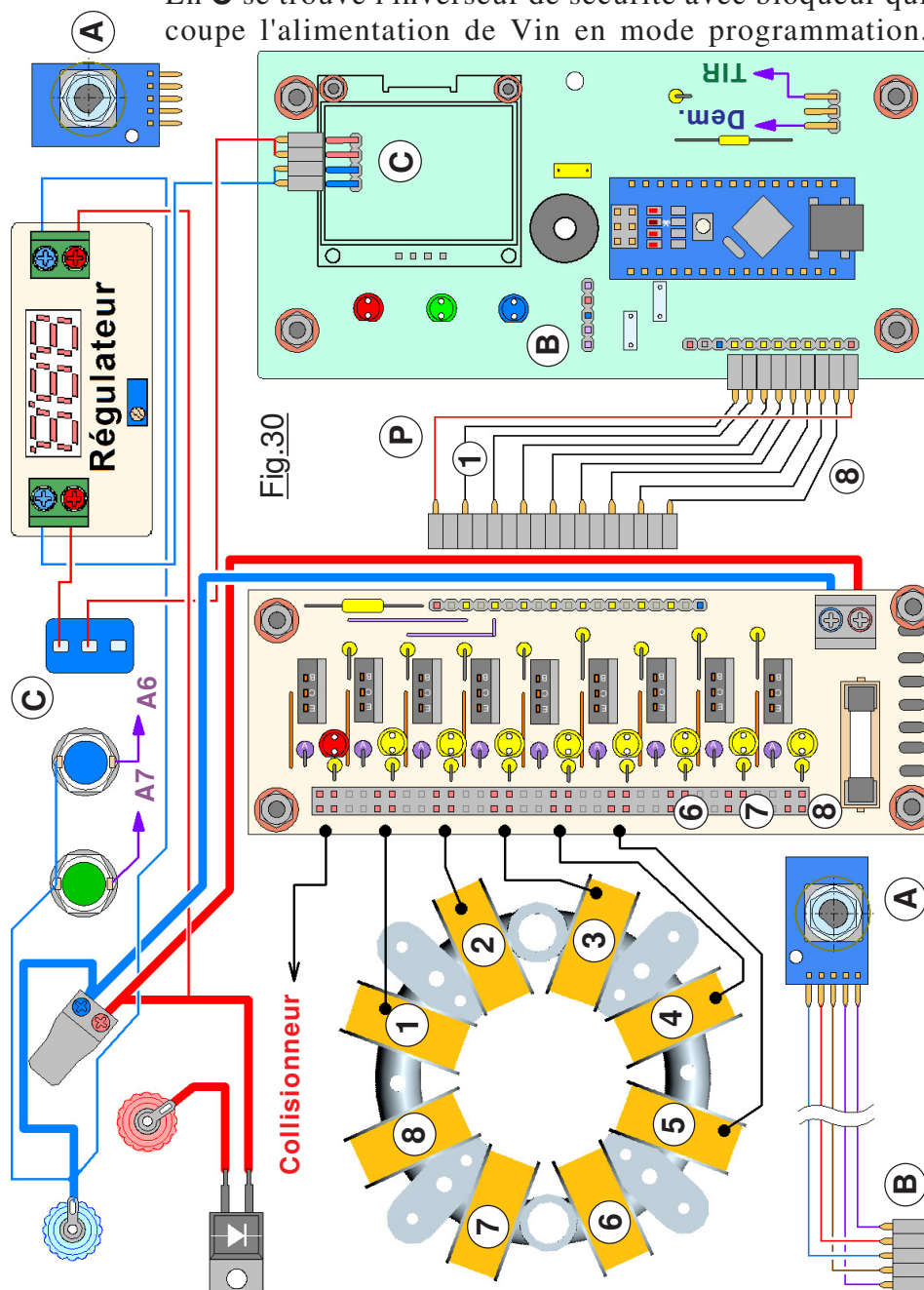



Fig.30

## > Effet du bouton poussoir du codeur rotatif.

Tout clic sur le  génère un BIP sonore pour attirer l'attention de l'opérateur. Puis, durant deux secondes la page écran qui

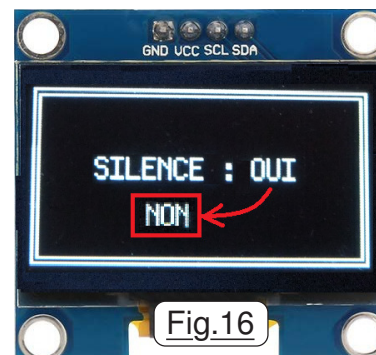


Fig.16

était affichée est remplacée par celle de la Fig.16 précisant l'état du mode **SILENCE**. Si **NON** est affiché, la séquence d'un cycle sur l'accélérateur de particule sera accompagnée d'un sifflement dont la tonalité devient de plus en plus stridente, terminée également par un son relativement grave si une collision est consignée.

## > L'utilisation du en EXPLOITATION.

Faire tourner dans un sens ou dans l'autre le bouton du codeur incrémental à pour effet de faire recycler sur l'écran OLED trois pages différentes et un "écran noir". Dans l'ordre croissant, on obtient les écrans Fig.15, Fig.17, Fig.18 et enfin l'écran tout noir. Lorsque l'écran est tout noir, la LED bleue est éteinte machine placée en configuration "sombre". La page écran de la Fig.17 précise le nombre de cycles et le nombre de collisions qui ont été déclenchés depuis la mise sous tension de la

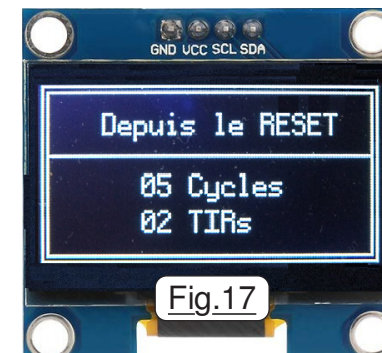


Fig.17

machine ou depuis le dernier RESET. Ne pas confondre avec l'information de la Fig.9 où toutes les séquences sont cumulées jusqu'à ce que l'on valide éventuellement une **RAZ** en EEPROM.

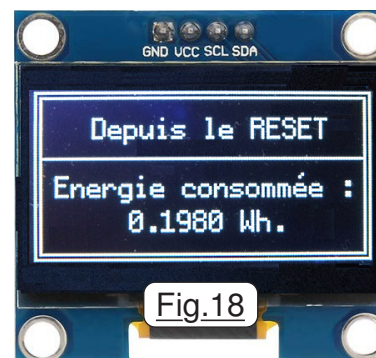


Fig.18

La page de la Fig.18 indique l'énergie électrique qui a été consommée par la machine sur le secteur alternatif 220V depuis la mise sous tension du cyclotron ou depuis le dernier RESET.

### ➤ Comportement thermique de la machine.

Chaque fois que l'on déclenche un cycle d'accélération, la LED bleue s'éteint et le texte **ATTENTE de Dem.** est remplacé par **Cycle en cours.** durant toute cette procédure. La LED verte s'illumine alors durant le cycle d'accélération. Puis la machine repasse en attente, la LED bleue étant à nouveau allumée. L'information **TMP** a été incrémentée de six unités. Par exemple sur la Fig.19 quatre cycles ont été déclenchés presque en cascade.

Le thermomètre à ruban accuse cette variation. Si on laisse la machine sans nouvelle intervention sur **Dem.** lentement le ruban lumineux diminue de longueur, car toutes les trois secondes la valeur de **TMP** diminue d'une unité. Si on clique sur **Dem.** de façon consécutive sans laisser de "temps morts" sur le système, après environ 24 cycles sans repos le thermomètre affiche 119 valeur qui est considérée comme une "maximale autorisée". Cette limite a été choisie expérimentalement. Les bobinages inducteurs sont encore relativement tièdes, et les fils émaillés des enroulements pourraient supporter facilement beaucoup plus. En revanche, le corps du tore étant en PLA, il semble logique de s'en tenir à des élévations de température modérées. Lorsque la limite critique est atteinte, le texte **Cycle en cours** est immédiatement remplacé par l'information **REFROIDISSEMENT** comme sur la Fig.20 et la LED rouge s'allume. Tant que le ruban ne sera pas revenu à la valeur de 1 pour **TMP** (Comme sur la Fig.14) la machine est inerte et aucune action sur le tableau de maitrise ne sera prise en compte. Pour reprendre le contrôle il faut patienter durant six minutes ... ou effectuer un RESET mais dans ce cas il n'est pas du tout conseillé d'exagérer avec l'utilisation du bouton poussoir **Dem.**

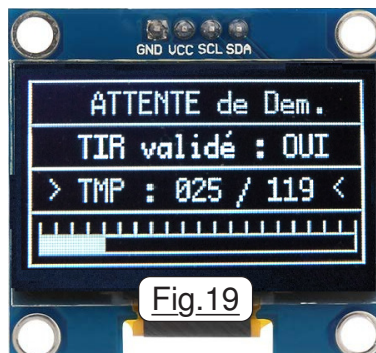


Fig.19

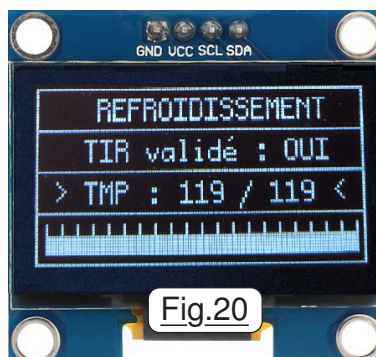
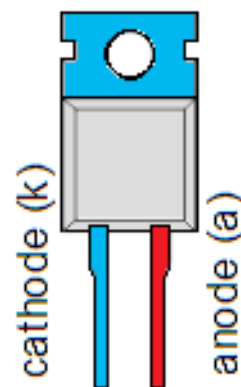


Fig.20

### TO220AC



ajuster la tension de sortie à convenance. Dans ce but deux prises pour fiches banane de 4mm sont prévues sur la plaque supportant les divers boutons de commande destinés à l'opérateur. Il n'est jamais totalement exclu qu'une erreur de branchement conduise à une inversion de polarité. Les Darlingtons dans ce cas passeraient en court-circuit franc. Si le fusible saute, alors l'alimentation de laboratoire peut fort bien débiter une intensité suffisante pour détruire le petit module de régulation et la carte Arduino par effet de cascade. Aussi, comme montré sur la Fig.28 une diode de protection **D** a été insérée sur la borne positive. Un composant de type 1N4007 ne conviendra pas, car l'intensité qui traverse cette diode est importante. Aussi, le choix c'est porté sur une BYW29-100 qui peut redresser jusqu'à 8A. Comme initialement l'importance de l'échauffement n'était pas connu, cette dernière de boîtier TO220AC a été immobilisée sur un petit radiateur. Comme le montre la Fig.29 elle est un peu éloignée du statif de la machine et assemblée sur ce dernier sur une rondelle plate large.

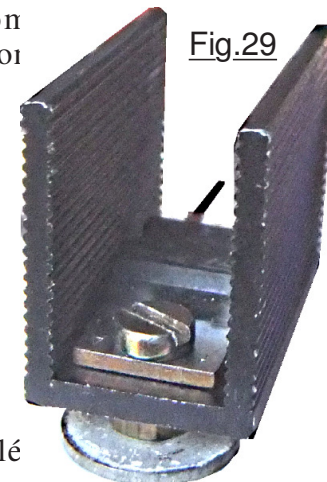


Fig.29

**ATTENTION :** Il est interdit *d'alimenter VIN simultanément avec la liaison Mini-USB de programmation de l'ATmega328 ou le régulateur de la carte NANO sera détruit !* Hors durant le développement du programme il faudra à la fois utiliser la ligne de dialogue USB pour téléverser le logiciel, et appliquer les 16,5Vcc pour alimenter en puissance. Durant cette phase de développement qui s'accompagne de très nombreuses manipulations, l'opérateur peut basculer inopinément l'inverseur de coupure de **VIN** gravé PGM. C'est la raison pour laquelle sur le schéma électrique de la Fig.28 l'**Inverseur** dans la position **PGM** permet de couper l'alimentation de sortie du **Régulateur**. *Pour éviter tout basculement malencontreux de cet inverseur en mode programmation, un bloqueur mécanique a été prévu.*

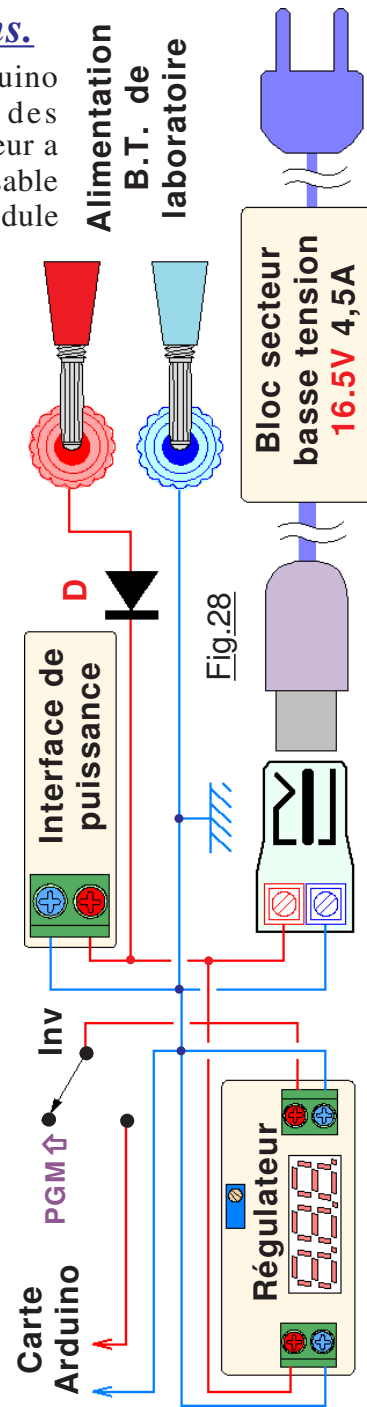


### 07) La gestion des alimentations.

Bien que l'entrée **VIN** de la carte Arduino NANO puisse être soumise à des tensions jusqu'à 20Vcc, un autre régulateur a été inséré en amont. Il n'est pas indispensable et l'on peut s'en passer. L'usage de ce module présente toutefois deux avantages :

- Ce régulateur va servir de barrière à toute impulsion parasite qui pourrait survenir de l'environnement électromagnétique avec commutations "brutales" de courants instantanés relativement importants et de plus sur des charges inductives.
- Le petit module utilisé incorpore un afficheur qui permet de visualiser à convenance la tension injectée en entrée du module, et celle que l'on peut ajuster en sortie.

Deux sources d'alimentation en énergie à partir du secteur alternatif 220V sont prévues. La première consiste à utiliser un bloc alimentation à découpage du commerce qui peut proposer en sortie une tension de l'ordre de 16,5VCC. Ce type de module assure toutes les sécurités imposées par les contraintes européennes, et présente un rendement excellent. Ce type de produit existe en de multiples variantes et à des tarifs très variables. Celui sélectionné pour le prototype permet de choisir des tensions de sortie entre 12 et 24V avec une intensité possible de 4,5A. La deuxième possibilité envisagée consiste à mettre à contribution une petite alimentation basse tension de laboratoire dont on peut



Page 9

### 04) La machine simplifiée.

Réservée à celle et ceux qui désirent commencer par une machine moins onéreuse n'intégrant que le minimum de composants, le programme d'exploitation **P10\_Machine\_MINIMALE.ino** est prévu pour un ensemble qui ne disposerait ni du codeur rotatif, ni de l'écran graphique voir sans le petit bruiteur.

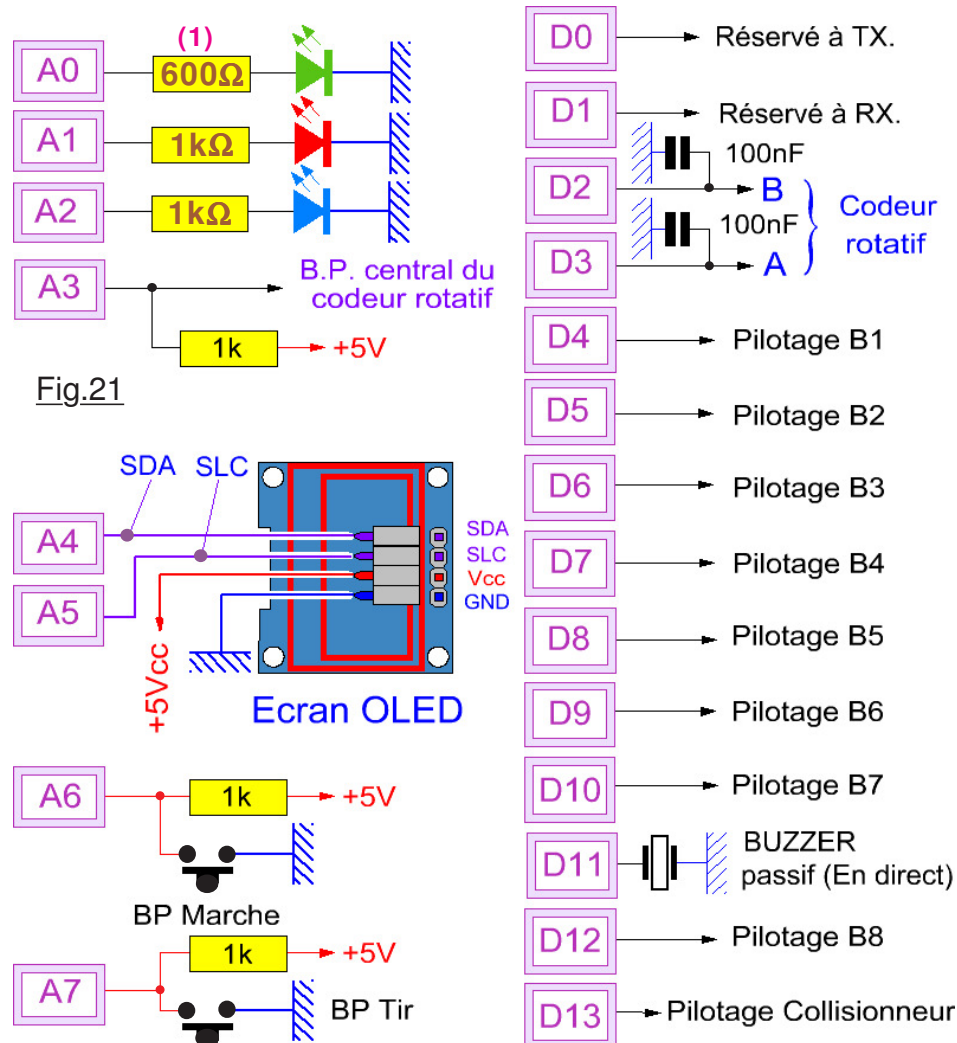
Il suffit de cliquer sur le bouton poussoir **TIR** pour valider ou non le collisionneur. C'est la LED verte qui précise alors son état. Quand la LED bleue s'allume, on clique sur le bouton poussoir **DEM.** et on déclenche un cycle. Enfin quand **la LED** bleue s'éteint et que **la rouge s'allume, c'est que la machine est en surchauffe et que le logiciel passe en pause pour attendre le refroidissement.** Il n'y a strictement rien à changer au matériel et au câblage. Il suffit de ne pas souder le bruiteur et de ne pas brancher le codeur rotatif ni insérer l'afficheur OLED sur son support. Par la suite, il sera toujours possible de compléter la machine et téléverser les deux programmes **P00\_Initialiser\_EEPROM.ino** suivi après son activation par **P09\_EXPLOITATION\_de\_la\_machine.ino**. Il suffit de téléverser **P10\_Machine\_MINIMALE.ino** sur la carte Arduino pour que la machine soit directement utilisable. Noter au passage que le programme simplifié est compatible avec une machine complète, c'est à dire que s'il est utilisé avec un ensemble "de luxe", le matériel ne risque rien. Les éléments tels que le codeur rotatif, l'écran OLED ou le bruiteur sont tout simplement ignorés.

#### Comportement de la machine simplifiée.

- Le bouton poussoir **TIR** valide ou muselle le collisionneur. C'est la LED verte qui précise la valeur de la consigne. Si le **TIR** est validé, elle est allumée. Si elle est éteinte le collisionneur est suspendu.
- La LED bleue allumée signale que la machine est en attente d'une consigne opérateur. (*Bouton **TIR** ou bouton **Dem.***)
- Le bouton poussoir **Dem.** déclenche une séquence d'accélération qui se termine par une collision si la LED verte est allumée.
- Si le bruiteur est branché sur la carte électronique, alors la séquence d'un cycle sur l'accélérateur de particule sera accompagnée d'un sifflement dont la tonalité devient de plus en plus stridente, terminé également par un son relativement grave si une collision est consignée. (*Machine simplifiée le mode SILENCE n'existe pas.*)

### 05) Affectation des E/S de la carte Arduino NANO.

Cette implantation reprend pour le codeur rotatif et pour l'afficheur OLED la répartition déjà adoptée sur d'autres applications à base d'Arduino NANO, simplifiant ainsi la rédaction du programme d'utilisation, ainsi que l'étude du circuit imprimé supportant la carte du microcontrôleur. Comme on peut le constater sur la Fig.21 deux entrées analogiques sont mobilisées pour les deux boutons poussoir. Il serait facile de gérer ces deux derniers sur une seule entrée. Toutefois, dans cette application il n'y a pas vraiment



➤ **Circuit imprimé de l'interface de puissance**

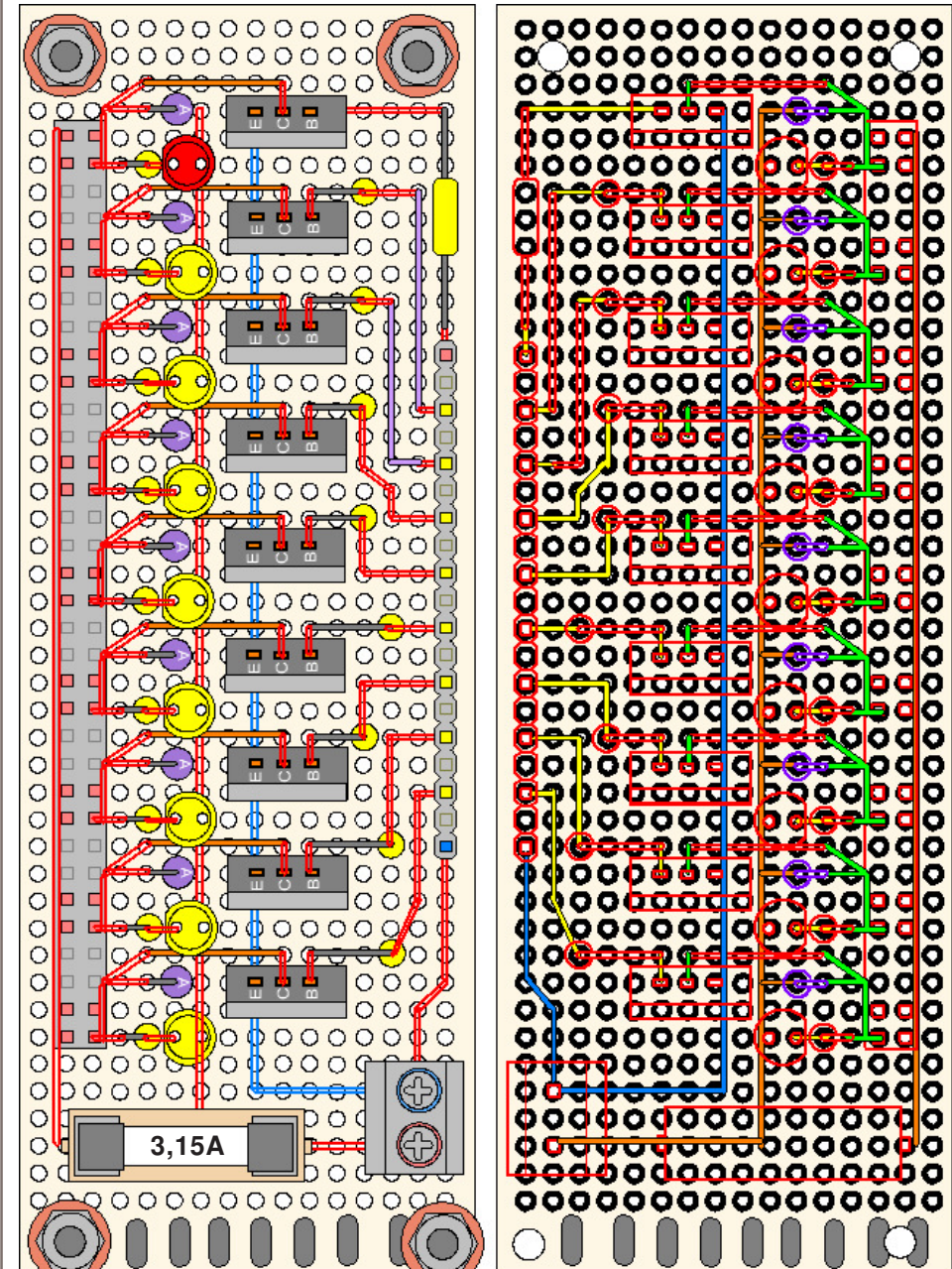


Fig.27



## 06) L'interface de puissance.

### ► Le Darlington TIP122.

Facile à se procurer, le choix c'est porté sur un composant hybride qui intègre deux transistors en mode Darlington et qui peut commuter sans problème jusqu'à 5A avec un gain en courant  $\beta$  de l'ordre de 1000. Le TIP122 peut dissiper jusqu'à 65W très largement au dessus des conditions présentes dans notre montage. Il sera totalement inutile de le munir d'un quelconque dissipateur de chaleur. La Fig.25 sur laquelle on notera que le collecteur est relié électriquement à la plaque métallique du boîtier présente son brochage. On constate sur la Fig.26 que l'encapsulation intègre les deux transistors architecturés en Darlington, ajoute deux résistances internes ainsi qu'une diode. Compte tenu du gain en courant de ce composant, un rapide calcul montre que lorsque **S** fournit +5V, l'ensemble est déjà saturé avec une résistance de 2k $\Omega$ . Par mesure d'optimisation, sur l'assemblage définitif on adopte des valeurs pour **Rb** de 1k $\Omega$ . Comme le prévoit la notice d'utilisation, la chute de tension  $\Delta U$  est d'environ 1,9V. Dans ces conditions, la bobine inductrice est parcourue par un courant approximatif de 2,75A. Durant la conduction en mode Saturé le transistor de puissance dissipe  $2,75 \times 1,9 = 5,2W$ . Comme il ne conduit le courant qu'un huitième du temps, sa dissipation moyenne sera d'environ 0,6W, une puissance très faible rendant totalement inutile un quelconque dissipateur de chaleur. C'est un avantage incontestable car on gagne ainsi beaucoup de place sur le circuit imprimé d'interfaçage. En **D** la diode de "roue-libre" annule les surtensions de commutation de la bobine **Bn**. Enfin, est ajoutée une LED notée **L** pour visualiser l'état de pilotage de chaque inducteur.

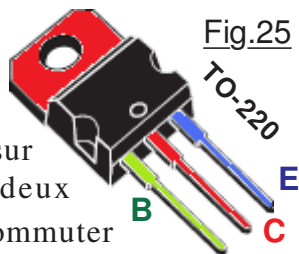


Fig.25

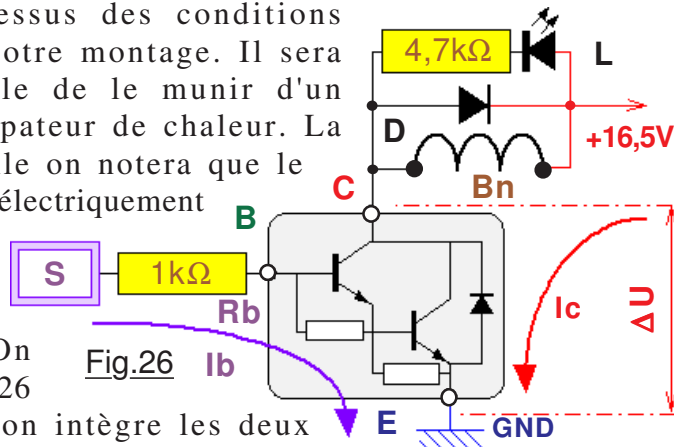


Fig.26

à optimiser. Du coup les plages de seuils de détection sont plus étendues, augmentant d'autant l'immunité aux éventuels parasites. C'est d'autant plus justifié que l'on commute à "fortes intensités" des bobines magnétiques, phénomène propice à la génération de perturbations électromagnétiques.

L'expérience a montré que l'impédance du bouton central du codeur rotatif étant trop élevée, des faux clics étaient détectés. Cette impédance a été diminuée en interposant une résistance de 1k $\Omega$  entre **A3** et le +5Vcc. De même que pour éliminer les faux rebonds de commutation sur le codeur incrémental, deux condensateurs de 100nF sont ajoutés entre **A**, **B** et GND. On peut naturellement se demander pourquoi les huit inducteurs n'ont pas été tous connectés à des broches voisines. On remarque un élément "perturbateur" en **D11**. C'est un BUZZER passif qui pour des facilités de programmation utilise de la PWM, **D12** et **D13** ne pouvant pas générer ce type de signal. C'est la raison pour laquelle le petit bruiteur à été intercalé, ce qui du reste ne complique en rien la programmation. La LED verte présentant un rendement un peu inférieur aux deux autres, en (1) sur la Fig.21 et sur la Fig.23 une résistance de 1500 $\Omega$  a été ajoutée en parallèle sur la 1k $\Omega$  initiale, d'où la valeur finale de 600 $\Omega$ . Le dessin du circuit imprimé est donné Fig.23 et Fig.24 en page P12.

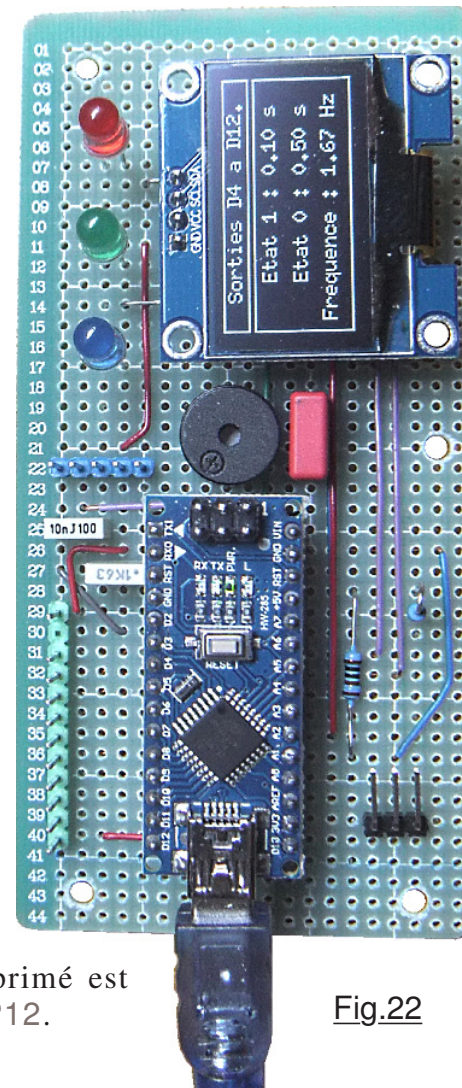


Fig.22



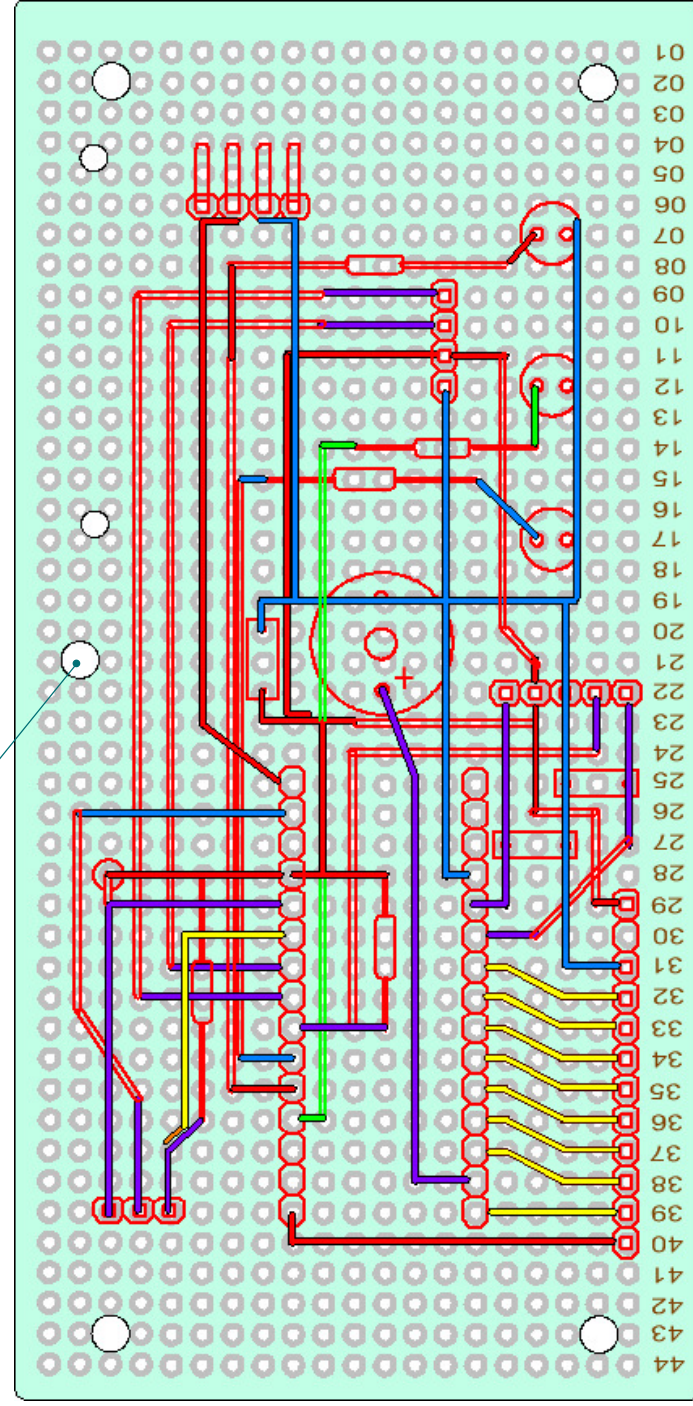
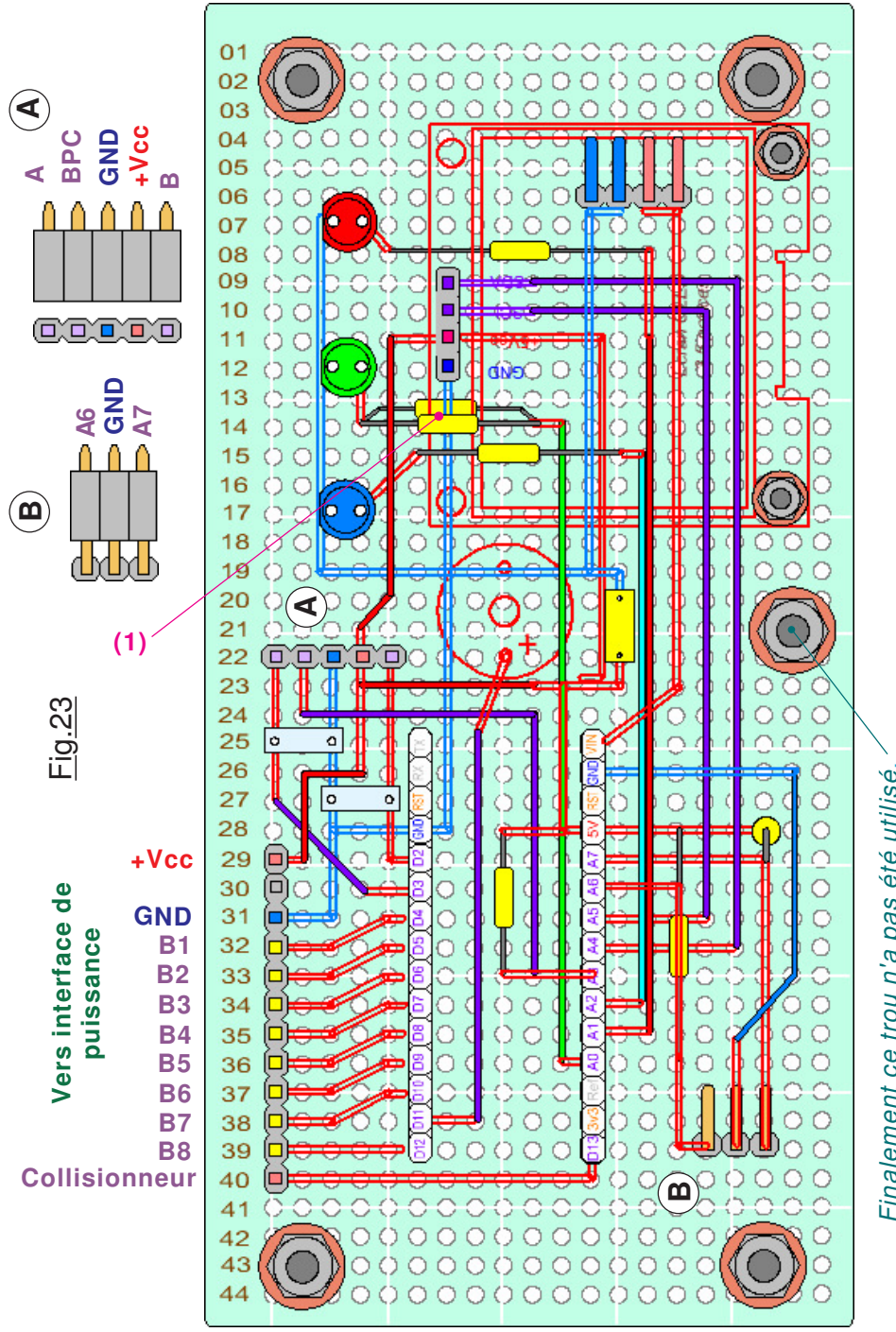


Fig. 24

## Circuit imprimé de gestion de la machine.

Lorsque le circuit imprimé est réalisé, le tester avec **P04\_Tester\_le\_CI\_NANO.ino** prévu à cet effet. L'utilisation de ce petit outil logiciel est précisée en tête de listage du programme.