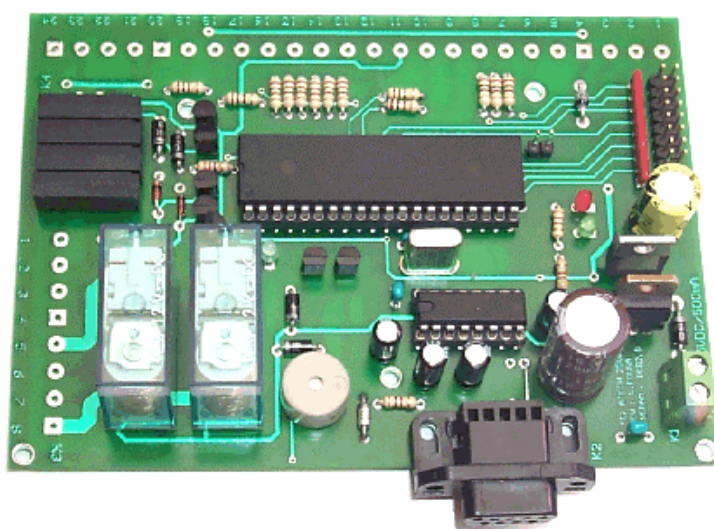


CNC 8AM



4 axis PC serial port stepper and servo motor control board

Dernières modifications le : 04 septembre 2002

© FOUGA Laurent 2002-06-26
web : <http://www.ifrance.com/mac1>
email : foul@ifrance.com

1.0) Description

La carte CNC8AM a été développée autour d'un microcontrôleur PIC16F877.

Elle permet de générer des interpolations linéaires 3D et circulaires 2D.

Elle utilise une liaison RS232 full duplex. La vitesse de transmission doit être configurée à 115200 bauds. Le protocole utilise 1 bit de start, 1 bit de stop sans parité.

4 cartes de contrôle (PAP8ABL,PAP8AUL ou PAP8AM) peuvent être connectées simultanément et permettre ainsi le pilotage de 4 moteurs pas à pas indépendants à une fréquence maximum de 50 000 pas/secondes.

Si on rajoute 2 relais de puissance, 8 entrées (contact fin de course) et 5 sorties TTL on dispose d'un ensemble prêt à l'emploi pour piloter une machine à commande numérique.

2.0) Configuration du port serie

Afin de pouvoir dialoguer avec la carte, il faut initialiser le port série RS232.

1. Reliez la carte au PC avec un câble RS232

2. Lancez Hyperterminal (utilitaire win9X)

Cliquez sur FICHIER | PROPRIETE

Sélectionnez le port COM (COM1 par défaut)

Cliquez sur le bouton configurer

Sélectionnez la vitesse (Bits par seconde) 115200,

8 bit de données

Parité aucune

Bit stop 1

Contrôle de flux aucun

Cliquez sur OK

Choisir l'onglet Configuration

Sélectionnez Touches Windows

Cliquez sur Configuration ASCII

Sélectionnez *Envoyer les fins de ligne avec retour à la ligne*

et *Reproduire localement les caractères entrés*

Cliquez sur OK

Cliquez sur OK

Cliquez sur APPEL | SE CONNECTER

3. Mettre la carte CNC8AM sous tension

4. Sur votre écran hyperterminal doit apparaître

CNC8AM controler V2.04 PRO

(C)TECHLF 2002

>

5. si rien n'apparaît éteignez tout et recommencez.

3.0) Syntaxe de commande

Les lettres en rouge (ex : **A**) sont les noms de commande.

Les lettres en bleus (ex : **>**) sont les commandes envoyées par la carte CNC8AM, il ne faut pas les entrer.

Les lettres en noir (ex : **A=2**) sont les commandes à envoyer à la carte CNC8AM.

Après chaque commande appuyer sur la touche ENTREE pour valider le transfert vers la carte.

Il est possible d'écrire mais aussi de lire les informations programmées par l'utilisateur.

Par exemple si vous tapez **X=12400** + ENTREE, la carte va programmer le registre X avec la valeur 12400 (pas). Si vous tapez **X** + ENTREE le terminal affichera **12400**.

Liste des commandes :

'A' : accélération en pas pour les 3 axes X,Y et Z

ex: **>A=2**

Accélération de 2x25 pas à chaque clock. Valeur max 250

'a' : configuration de la largeur d'impulsion des impulsions moteurs

ex: **>a=10**

La largeur d'impulsion des moteurs est de 10µs.

'b' : configuration du DUTY CYCLE du pwm (variateur pour broche)

ex: **>B=250**

>b=125

PWM de 19607 Hz avec duty cycle de 50%

'B' : configuration de la fréquence PWM (PR2)

ex: **>B=255**

Fréquence PWM de 19607 Hz. Pour obtenir la periode (PR2) on utilise la formule

$PR2 = 5000000 / 255$ (voir data sheet 16F877)

'c' : trace un arc de cercle

ex: **>C**

Coordonnées de départ **X** et **Y**, rayon **Q**, coordonnée d'arrivée **t**, sens de rotation **s**, quart **q**. On considère que le centre du cercle est en 0,0.

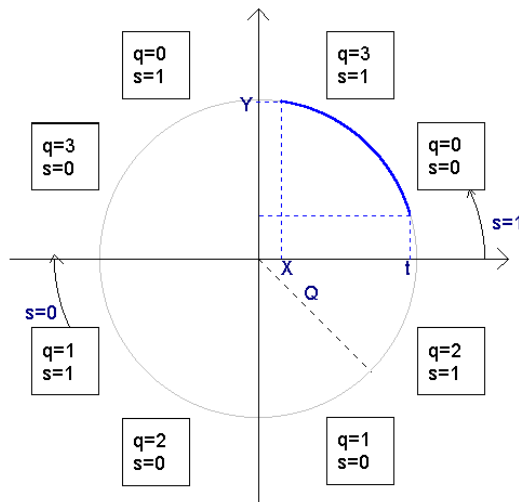


Figure 1

'C': lecture des contacts fin de course

ex: **>C**
254

Contact bit 0 sur ON

'd': désactive le tests des contacts fin de courses

ex:

>d=1

Le test des contacts fin de course est actif. Les 3 ou 4 axes stoppent dès que un des contacts est fermé. Pour d=0 le test est inactif.

'D': dump des paramètres X,Y,Z et HOME position

ex:

>D

X=0 Xh=0

Y=0 Yh=0

Z=0 Zh=0

'e': initialise l'état des sorties digitales

ex:

>e=1

sortie 1 ON

Autres valeurs

e=2 sortie 2 ON

e=4 sortie 3 ON

e=8 sortie 4 ON

e=16 sortie 5 ON

'E': efface les coordonnées X,Y et Z ainsi que les coordonnées Xm,Ym et Zm

ex:

>E

'G': déplace l'outil jusqu'à la position HOME (Xh,Yh et Zh)

ex:

>G

'H': capture les coordonnées HOME Xh,Yh et Zh

ex:

>H

'I': sauvegarde des anciennes coordonnées

ex:

>I

'J': déplace l'outil suivant un cercle de rayon Q (vitesse constante)

ex:

>J

'L': déplace l'outil en ligne droite de Xm,Ym,Zm à X,Y,Z sans profil (vitesse constante)

ex:

>L

'M' : déplace l'outil en ligne droite de X_m, Y_m, Z_m à X, Y, Z avec un profil trapézoïdal

ex:

>M

La vitesse de départ est initialisée avec la commande 'T'

La vitesse max est initialisée avec la commande 'W'

'N' : acquisition de la température

ex:

>N

43

Température (en °C) = $N \times 0.48828125$

Ex : $43 \times 0.48828125 = 21 \text{ °C}$

'P' : initialise le prescaler (vitesse moteur) du PIC (voir datasheet PIC16F877)

ex:

>P=1

Prescaler sur x2

Autres valeurs

x8	P=3	vitesse min. 25 pas/sec	vitesse max. 6250 pas/sec
x4	P=2	vitesse min. 50 pas/sec	vitesse max. 12500 pas/sec
x2	P=1	vitesse min. 100 pas/sec	vitesse max. 25000 pas/sec
x1	P=0	vitesse min. 200 pas/sec	vitesse max. 50000 pas/sec

'q' : initialise le quart de l'arc de cercle. Voir commande c

ex:

>q =1

'Q' : initialise le rayon du cercle en pas (à utiliser avec la commande 'J')

ex:

>Q =15000

Le rayon du cercle sera de 15000 pas.

Attention !!! Les axes X et Y doivent avoir la même résolution au niveau du déplacement. Voir aussi la commande c.

Valeur max 4.294.967.295 pas

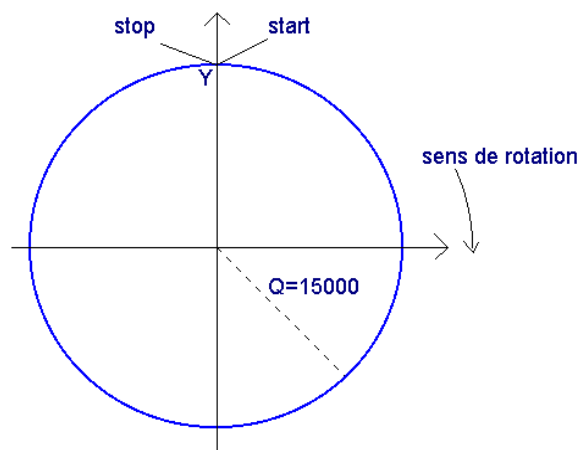


Figure 2

'R' : initialise l'état des deux relais et du buzzer

ex:

>R=1

Relais 1 ON

Autres valeurs

R=2 Relais 2 ON

R=3 Relais 1 ON et Relais 2 ON

R=0 Relais 1 OFF , Relais 2 OFF et Buzzer OFF

R=4 Buzzer ON

's' : sens de rotation pour arc de cercle

ex:

>s=0

Sens de rotation trigonométrique. Voir la commande **c**.

'S' : initialise l'état des moteurs (moteur ON ou OFF)

ex:

>S=3

Moteur A et B ON, moteur C et D OFF

Valeur de 0 à 15

't' : initialise la valeur d'arrivée (axe des X) pour l'arc de cercle

ex:

>t=12300

Voir la commande **c**.

'T' : initialise la valeur de départ pour l'accélération (en pas/seconde) du profil trapézoïdal.
La vitesse max est donnée par la commande 'W'

ex:

>T=4

En profil trapézoïdal le moteur aura une vitesse de départ et d'arrivée de 100 pas/sec. (4x25pas/sec.)

Valeur max 250

'V' : initialise la vitesse pour les axes X, Y et Z (sans profil)

ex:

>P=2

>V=8

Vitesse de 400 pas/sec. (2x8x25pas/sec.)

Valeur max 250

'W' : initialise la vitesse maximum pour les axes X,Y et Z avec profil trapézoïdal

ex:

>W=80

>P=1

Vitesse max de 2000 pas/sec (1x80x25pas/sec.)

Valeur max 250

'X' : coordonnées X en pas

ex:

```
>E
>X=12000
>L
```

le moteur se déplacera sur l'axe des X de 12000 pas à une vitesse constante

En fin de déplacement $X_m=X$

Valeurs entre $-2.147.483.647$ pas et $+2.147.483.647$ pas

'Y' : coordonnées Y en pas

ex:

```
>E
>X=3000
>Y=200
>L
```

l'outil se déplacera de (0,0) à (3000,200) à une vitesse constante

En fin de déplacement $Y_m=Y$

Valeurs entre $-2.147.483.647$ pas et $+2.147.483.647$ pas

'Z' : coordonnées Z en pas

ex:

```
>E
>Z=200
>M
```

L'outil se déplacera avec un profil trapézoïdal de 200pas.

En fin de déplacement $Z_m=Z$

Valeurs entre $-2.147.483.647$ pas et $+2.147.483.647$ pas

4.0) Fonctions de la DLL (dynamic link librairie) cnc.dll

Afin de développer votre propre logiciel de commande voici en détail les fonctions permettant de piloter la carte CNC8AM.

```

HANDLE __export FAR PASCAL CNCConnect(char *name, unsigned int baudrate, char *stuff);
BOOL __export FAR PASCAL CNCSendStr(HANDLE ComPort, char *stuff, int n);
DWORD __export FAR PASCAL CNCGetStr(HANDLE ComPort, char *stuff, unsigned int *n);
DWORD __export FAR PASCAL CNCGetIn(HANDLE ComPort);
BOOL __export FAR PASCAL CNCClrInbuf(HANDLE ComPort);
BOOL __export FAR PASCAL CNCSetEscCode(HANDLE ComPort, int EscCode);
BOOL __export FAR PASCAL CNCCLose(HANDLE ComPort);
BOOL __export FAR PASCAL CNCCLrOutbuf(HANDLE ComPort);

void __export FAR PASCAL CNCReset ( HANDLE ComPort );
BOOL __export FAR PASCAL CNCDump ( HANDLE ComPort );
BOOL __export FAR PASCAL CNCMove ( HANDLE ComPort );
BOOL __export FAR PASCAL CNCRapidMove ( HANDLE ComPort );

BOOL __export FAR PASCAL CNCRelay ( HANDLE ComPort , unsigned char NumRel );
BOOL __export FAR PASCAL CNCMotorState ( HANDLE ComPort , unsigned char MotState);

BOOL __export FAR PASCAL CNCSetHoming ( HANDLE ComPort );
BOOL __export FAR PASCAL CNCGoHome ( HANDLE ComPort );
BOOL __export FAR PASCAL CNCCLearAll ( HANDLE ComPort );
BOOL __export FAR PASCAL CNCSetOld ( HANDLE ComPort );
BOOL __export FAR PASCAL CNCSetX ( HANDLE ComPort, int X );
BOOL __export FAR PASCAL CNCSetY ( HANDLE ComPort, int Y );
BOOL __export FAR PASCAL CNCSetZ ( HANDLE ComPort, int Z );

BOOL __export FAR PASCAL CNCSetXYZVel ( HANDLE ComPort, unsigned int velxyz );
BOOL __export FAR PASCAL CNCSetMaxVel ( HANDLE ComPort, unsigned int velmax );
BOOL __export FAR PASCAL CNCSetAccelTime ( HANDLE ComPort, unsigned int acctime );
BOOL __export FAR PASCAL CNCSetAccelStart ( HANDLE ComPort, unsigned int acstart );
BOOL __export FAR PASCAL CNCSetPrescaler ( HANDLE ComPort, unsigned int prescaler );

BOOL __export FAR PASCAL CNCSetPWMPeriod ( HANDLE ComPort, unsigned char pwmperiod );
BOOL __export FAR PASCAL CNCSetPWMDutyCycle ( HANDLE ComPort, unsigned char pwmdutycycle );

BOOL __export FAR PASCAL CNCWaitEvent ( HANDLE ComPort );

BOOL __export FAR PASCAL CNCCLircle ( HANDLE ComPort, int X, int Y, int R );

BOOL __export FAR PASCAL CNCCLArc ( HANDLE ComPort, int StartX, int StartY, int EndX, int
EndY, int R, unsigned char Quad );

BOOL __export FAR PASCAL CNCCLArcDir ( HANDLE ComPort, unsigned char SensRot );

int __export FAR PASCAL CNCCLGetTemp ( HANDLE ComPort );

int __export FAR PASCAL CNCCLGetX ( HANDLE ComPort );
int __export FAR PASCAL CNCCLGetY ( HANDLE ComPort );
int __export FAR PASCAL CNCCLGetZ ( HANDLE ComPort );

int __export FAR PASCAL CNCCLGetSense ( HANDLE ComPort );

void __export FAR PASCAL CNCCLSpindleOff ( HANDLE ComPort );
void __export FAR PASCAL CNCCLSpindleOn ( HANDLE ComPort );

unsigned char __export FAR PASCAL CNCCLGetPWMPeriod ( HANDLE ComPort );
unsigned char __export FAR PASCAL CNCCLGetPWMDutyCycle ( HANDLE ComPort );
HANDLE __export FAR PASCAL CNCCLOpen(void);

BOOL __export FAR PASCAL CNCCLTestSense ( HANDLE ComPort, unsigned char TSense );
BOOL __export FAR PASCAL CNCCLSetOutput ( HANDLE ComPort, unsigned char Output );
BOOL __export FAR PASCAL CNCCLSetPULSE ( HANDLE ComPort , unsigned char Pulse );
unsigned char __export FAR PASCAL CNCCLGetPulse ( HANDLE ComPort );

```

Si vous utilisez DELPHI voici la déclaration de l'importation de la DLL (cnc.dll) dans votre application (vous pouvez faire un copier coller):

```
function CNCConnect( name : PChar ; baudrate : integer; PtCNCString : PChar): THANDLE;
stdcall ; external 'cnc.dll' index $01;

function CNCSendString (ComPort : THANDLE; Str : PChar; n : WORD): Boolean; stdcall ;
external 'cnc.dll' index $02;

function CNCGetString (ComPort : THANDLE; PStr : PChar ; n : PWord) :DWORD; stdcall ;
external 'cnc.dll' index $04;

function CNCGetIn (ComPort : THANDLE): WORD; stdcall ; external 'cnc.dll' index $05;

function CNCClrInbuf (ComPort : THANDLE): Boolean; stdcall ;external 'cnc.dll' index
$06;

function CNCSetEscCode (ComPort : THANDLE; EscCode : WORD): Boolean; stdcall ; external
'cnc.dll' index $08;

function CNCClrOutbuf (ComPort : THANDLE): Boolean; stdcall ;external 'cnc.dll' index
$07;

function CNCCLose (ComPort : THANDLE): Boolean; stdcall ; external 'cnc.dll' index $09;

procedure CNCReset (ComPort : THANDLE); stdcall ; external 'cnc.dll' index $0A;

function CNCDump (ComPort : THANDLE): Boolean; stdcall ; external 'cnc.dll' index $0B;

function CNCMove (ComPort : THANDLE): Boolean; stdcall ; external 'cnc.dll' index $0C;

function CNCRapidMove (ComPort : THANDLE): Boolean; stdcall ; external 'cnc.dll' index
$0D;

function CNCRelay (ComPort : THANDLE ; NumRel : BYTE): Boolean; stdcall ; external
'cnc.dll' index $0E;

function CNCMotorState (ComPort : THANDLE ; MotState : BYTE): Boolean; stdcall ;
external 'cnc.dll' index $0F;

function CNCSetHoming (ComPort : THANDLE): Boolean; stdcall ;external 'cnc.dll' index
$10;

function CNCGoHome (ComPort : THANDLE): Boolean; stdcall ;external 'cnc.dll' index $11;

function CNCSetX (ComPort : THANDLE; X : integer): Boolean; stdcall ;external 'cnc.dll'
index $12;

function CNCSetY (ComPort : THANDLE; Y : integer): Boolean; stdcall ;external 'cnc.dll'
index $13;

function CNCSetZ (ComPort : THANDLE; Z : integer): Boolean; stdcall ;external 'cnc.dll'
index $14;

function CNCCLearAll (ComPort : THANDLE): Boolean; stdcall ;external 'cnc.dll' index
$15;

function CNCSetOld (ComPort : THANDLE): Boolean; stdcall ;external 'cnc.dll' index $16;

function CNCSetXYZVel (ComPort : THANDLE; velxyz : integer): Boolean; stdcall ;external
'cnc.dll' index $17;

function CNCSetMaxVel (ComPort : THANDLE; velmax : integer): Boolean; stdcall ;external
'cnc.dll' index $18;

function CNCSetAccelTime (ComPort : THANDLE; acctime : integer): Boolean; stdcall
;external 'cnc.dll' index $19;
```

```

function CNCSetAccelStart (ComPort : THANDLE; accstart : integer): Boolean; stdcall
;external 'cnc.dll' index $1A;

function CNCSetPrescaler (ComPort : THANDLE; prescaler : integer): Boolean; stdcall
;external 'cnc.dll' index $1B;

function CNCSetPWMPeriod (ComPort : THANDLE; pwmperiod : integer): Boolean; stdcall
;external 'cnc.dll' index $1C;

function CNCSetPWMDutyCycle (ComPort : THANDLE; pwmdutycycle : integer): Boolean;
stdcall ;external 'cnc.dll' index $1D;

function CNCWaitEvent (ComPort : THANDLE): Boolean; stdcall ;external 'cnc.dll' index
$3;

function CNCCircle (ComPort : THANDLE; X,Y,Z : integer): Boolean; stdcall ;external
'cnc.dll' index $1E;

function CNCArc (ComPort : THANDLE; StartX, StartY, EndX, EndY, Radius : integer ; Quad
: byte): Boolean; stdcall ;external 'cnc.dll' index $1F;

function CNCArcDir (ComPort : THANDLE; SensRot : byte): Boolean; stdcall ;external
'cnc.dll' index $20;

function CNCGetTemp (ComPort : THANDLE): integer; stdcall ;external 'cnc.dll' index $21;

function CNCGetX (ComPort : THANDLE): integer; stdcall ;external 'cnc.dll' index $22;

function CNCGetY (ComPort : THANDLE): integer; stdcall ;external 'cnc.dll' index $23;

function CNCGetZ (ComPort : THANDLE): integer; stdcall ;external 'cnc.dll' index $24;

function CNCGetSense (ComPort : THANDLE): integer; stdcall ;external 'cnc.dll' index
$25;

function CNCSpindleOff (ComPort : THANDLE): integer; stdcall ;external 'cnc.dll' index
$26;

function CNCSpindleOn (ComPort : THANDLE): integer; stdcall ;external 'cnc.dll' index
$27;

function CNCGetPWMPeriod(ComPort : THANDLE): integer; stdcall ;external 'cnc.dll' index
$28;

function CNCGetPWMDutyCycle(ComPort : THANDLE): integer; stdcall ;external 'cnc.dll'
index $29;

function CNCOpen: THandle; stdcall ;external 'cnc.dll' index $2A;

function CNCTestSense (ComPort : THANDLE; TSense : byte): Boolean; stdcall ;external
'cnc.dll' index $2B;

function CNCSetOutput (ComPort : THANDLE; Output : byte): Boolean; stdcall ;external
'cnc.dll' index $2C;

function CNCSetPulse (ComPort : THANDLE; Pulse : byte ): Boolean; stdcall ;external
'cnc.dll' index $2D;

function CNCGetPulse (ComPort : THANDLE): byte; stdcall ;external 'cnc.dll' index $2E;

```

implementation

{SR *.DFM}

5.0) Description des fonctions :

CNCConnect

La fonction CNCConnect permet d'initialiser la port COM

```
HANDLE __export FAR PASCAL CNCConnect(
    char *name,           // chaine de caractère du port COM
    unsigned int baudrate, // vitesse de transmission
    char *stuff          // pointeur sur la chaine de caractère envoyée par la
                        // carte
);
```

Paramètres

Name

Pointeur sur la chaine de caractère définissant le port COM
« COM1 », « COM2 » , etc...

baudrate

Vitesse de transmission : 115200 bits/s

**stuff*

Pointeur sur chaine de caractère envoyée par la carte

Retour

Handle du port serie.

INVALID_HANDLE_VALUE si erreur à l'ouverture du port. (voir WIN32.HLP)

Utilisation

Pour le DEBUG seulement !!!

CNCSendStr

La fonction CNCSendStr permet d'envoyer une commande à la carte

```
BOOL __export FAR PASCAL CNCSendStr(
    HANDLE ComPort, // Handle du port COM
    char *stuff,    // pointeur sur la chaine à envoyer
    int n          // nombre de caractères
);
```

Paramètres

ComPort

Handle du port COM voir CNCOpen

**stuff*

Pointeur sur la chaine à envoyer

n

Nombre de caractères à envoyer

Retour

Zéro si erreur sur le port COM

Non zéro si l'écriture s'est bien passée

Remarques

Après chaque appel à cette fonction il faut appeler la fonction CNCGetStr pour vider le buffer de réception

Utilisation

Pour le DEBUG seulement !!!

Voir

[CNCGetStr](#), [CNCOpen](#)

CNCGetStr

La fonction CNCGetStr permet de lire le contenu du buffer de réception du port COM

```
DWORD __export FAR PASCAL CNCGetStr(
    HANDLE ComPort, // Handle du port COM
    char *stuff, // vitesse de transmission
    unsigned int *n // pointeur sur la chaîne de caractère envoyée par la // carte
);
```

Paramètres

ComPort

Handle du port COM voir CNCOpen

***stuff**

Pointeur sur chaîne de caractère envoyée par la carte

***stuff**

Pointeur sur le nombre de caractère reçu

Retour

Nombre de caractères reçu

Remarques

Cette fonction appelle la fonction CNCWaitEvent

Exemple

Récupère les coordonnées X,Y et Z de la machine

/* DLL déclaration */

HANDLE ComHandle;

char *RetStr;

DWORD *NumRead;

void main(void)

{

...

CNCDump(ComHandle);

CNCGetStr(Comhandle, RetStr, NumRead);

...

}

Utilisation

Pour le DEBUG seulement !!!

Voir

[CNCOpen](#), [CNCWaitEvent](#)

CNCGetIn

La fonction CNCGetIn retourne le nombre de caractère contenu du buffer de réception du port COM

```
DWORD __export FAR PASCAL CNCGetIn(
    HANDLE ComPort // Handle du port COM
);
```

Paramètres

ComPort

Handle du port COM voir CNCOpen

Retour

Nombre de caractères reçu

Utilisation

Pour le DEBUG seulement !!!

Voir

[CNCOpen](#)

CNCClrInBufStr

La fonction CNCClrInbuf efface le contenu du buffer de réception du port COM

```
BOOL __export FAR PASCAL CNCClrInbuf(  
    HANDLE ComPort    // Handle du port COM  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

Retour

Zéro si erreur sur le port COM
Non zéro si l'effacement s'est bien passée

Utilisation

Pour le DEBBUG seulement !!!

Voir

[CNCOpen](#)

CNCSetEscCode

La fonction CNCSetEscCode permet de faire un reset hardware de la carte

```
BOOL __export FAR PASCAL CNCSetEscCode(  
    HANDLE ComPort,    // Handle du port COM  
    int EscCode        // niveau logique du reset  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

EscCode

Modifie le niveau logique du reset
SETRTS : niveau logique HAUT (normal)
CLRRTS : niveau logique BAS (RESET)

Retour

Zéro si erreur sur le port COM
Non zéro si le RESET s'est bien passé

Remarques

Pour le reset utiliser plutôt la fonction CNCReset

Utilisation

Pour le DEBBUG seulement !!!

Voir

[CNCReset](#), [CNCOpen](#)

CNCCLose

La fonction CNCCLose permet de fermer le port COM

```
BOOL __export FAR PASCAL CNCCLose(  
    HANDLE ComPort // Handle du port COM  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

Retour

Nombre de caractères reçu

Exemple

Ouverture et fermeture du port COM

```
/* DLL déclaration */  
HANDLE ComHandle;  
  
void main(void)  
{  
    ...  
    ComHandle = CNCOpen();  
    ...  
    CNCCLose(ComHandle);  
    ...  
}
```

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCReset

La fonction CNCReset permet de faire un RESET hardware de la carte

```
void __export FAR PASCAL CNCReset(  
    HANDLE ComPort // Handle du port COM  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

Retour

Aucun

Remarques

Après chaque appel à cette fonction il faut appeler la fonction CNCGetStr pour vider le buffer de réception

Utilisation

Pour le DEBUG seulement !!!

Voir

[CNCGetStr](#), [CNCOpen](#)

CNCDump

La fonction CNCDump permet de relire les coordonnées X,Y et Z de la machine

```
BOOL __export FAR PASCAL CNCDump(  
    HANDLE ComPort // Handle du port COM  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

Retour

Zéro si erreur sur le port COM
Non zéro si le DUMP s'est bien passé

Remarques

Après chaque appel à cette fonction il faut appeler la fonction CNCGetStr pour vider le buffer de réception

Utilisation

Pour le DEBUG seulement !!!

Voir

[CNCGetStr](#), [CNCOpen](#)

CNCMove

La fonction CNCMove permet de déplacer l'outil sur les axes X,Y et Z de la machine avec une vitesse constante

```
BOOL __export FAR PASCAL CNCMove(  
    HANDLE ComPort // Handle du port COM  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

Retour

Zéro si erreur sur le port COM
Non zéro si le déplacement s'est bien passé

Remarques

Avant l'appel de cette fonction il faut initialiser les coordonnées X,Y et Z avec les fonctions CNCSetX, CNCSetY et CNCSetZ
Cette fonction attend la réception du caractère fin de commande

Utilisation

Dans votre application !!!

Voir

[CNCSetXYZVel](#), [CNCSetMaxVel](#), [CNCSetAccelTime](#), [CNCSetAccelStart](#), [CNCSetPrescaler](#), [CNCSetX](#), [CNCSetY](#), [CNCSetZ](#)

CNCRapidMove

La fonction CNCRapidMove permet de déplacer l'outil sur les axes X,Y et Z de la machine avec une vitesse trapézoïdale

```

BOOL __export FAR PASCAL CNCRapidMove(
    HANDLE ComPort // Handle du port COM
);

```

Paramètres

ComPort

Handle du port COM voir CNCOpen

Retour

Zéro si erreur sur le port COM

Non zéro si le déplacement s'est bien passé

Remarques

Avant l'appel de cette fonction il faut initialiser les coordonnées X,Y et Z avec les fonctions CNCSetX, CNCSetY et CNCSetZ

Cette fonction attend la réception du caractère fin de commande

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#), [CNCSetXYZVel](#), [CNCSetMaxVel](#), [CNCSetAccelTime](#), [CNCSetAccelStart](#), [CNCSetPrescaler](#), [CNCSetX](#), [CNCSetY](#), [CNCSetZ](#)

CNCRelay

La fonction CNCRelay permet de configurer l'état des 2 relais de puissance

```

BOOL __export FAR PASCAL CNCRelay(
    HANDLE ComPort, // Handle du port COM
    unsigned char NumRel // état des relais
);

```

Paramètres

ComPort

Handle du port COM voir CNCOpen

NumRel

Etat des relais

Bit 0 : état du relais 1 (0 ouvert, 1 fermé)

Bit 1 : état du relais 2 (0 ouvert, 1 fermé)

Retour

Zéro si erreur sur le port COM

Non zéro si la configuration s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCMotorState

La fonction CNCMotorState permet de configurer l'état des 4 moteurs

```
BOOL __export FAR PASCAL CNCMotorState(  
    HANDLE ComPort,           // Handle du port COM  
    unsigned char MotState    // état des moteurs  
);
```

Paramètres

ComPort

Handle du port COM voir CNCOpen

MotState

Etat des moteurs

Bit 0 : état du moteur 1 (0 ON, 1 OFF)

Bit 1 : état du moteur 2 (0 ON, 1 OFF)

Bit 2 : état du moteur 2 (0 ON, 1 OFF)

Bit 3 : état du moteur 2 (0 ON, 1 OFF)

Retour

Zéro si erreur sur le port COM

Non zéro si la configuration s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCSetHoming

La fonction CNCSetHoming permet d'initialiser les coordonnées HOME interne à la carte avec la position actuelle des coordonnées X,Y et Z

```
BOOL __export FAR PASCAL CNCSetHoming(  
    HANDLE ComPort           // Handle du port COM  
);
```

Paramètres

ComPort

Handle du port COM voir CNCOpen

Retour

Zéro si erreur sur le port COM

Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCGoHome

La fonction CNCGoHome permet de déplacer l'outil à sa position HOME

```
BOOL __export FAR PASCAL CNCGoHome (  
    HANDLE ComPort // Handle du port COM  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

Retour

Zéro si erreur sur le port COM
Non zéro si le déplacement s'est bien passé

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCClearAll

La fonction CNCClearAll permet d'initialiser les coordonnées internes à la carte à zéro.
Les coordonnées HOME sont aussi réinitialisées à zéro.

```
BOOL __export FAR PASCAL CNCClearAll (  
    HANDLE ComPort // Handle du port COM  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

Retour

Zéro si erreur sur le port COM
Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCSetOld

La fonction CNCSetOld permet d'initialiser les anciennes coordonnées internes à la carte à zéro avec la position actuelle de l'outil

```
BOOL __export FAR PASCAL CNCSetOld (
    HANDLE ComPort          // Handle du port COM
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

Retour

Zéro si erreur sur le port COM
Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCSetX

La fonction CNCSetX permet d'initialiser les coordonnées X de l'outil

```
BOOL __export FAR PASCAL CNCSetX(
    HANDLE ComPort,          // Handle du port COM
    int X                    // coordonnées X en pas moteur
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

X
Coordonnée d'arrivée X en pas moteur

Retour

Zéro si erreur sur le port COM
Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCSetY

La fonction CNCSetY permet d'initialiser les coordonnées Y de l'outil

```
BOOL __export FAR PASCAL CNCSetY(  
    HANDLE ComPort, // Handle du port COM  
    int Y           // coordonnées Y en pas moteur  
);
```

Paramètres

ComPort

Handle du port COM voir CNCOpen

Y

Coordonnée d'arrivée Y en pas moteur

Retour

Zéro si erreur sur le port COM

Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCSetZ

La fonction CNCSetZ permet d'initialiser les coordonnées Z de l'outil

```
BOOL __export FAR PASCAL CNCSetZ(  
    HANDLE ComPort, // Handle du port COM  
    int Z           // coordonnées Z en pas moteur  
);
```

Paramètres

ComPort

Handle du port COM voir CNCOpen

Z

Coordonnée d'arrivée Z en pas moteur

Retour

Zéro si erreur sur le port COM

Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCSetXYZVel

La fonction CNCSetXYZVel permet d'initialiser la vitesse de déplacement des axes X, Y et Z

```

BOOL __export FAR PASCAL CNCSetXYZVel (
    HANDLE ComPort,           // Handle du port COM
    unsigned int velxyz      // vitesse des axes
);

```

Paramètres

ComPort

Handle du port COM voir CNCOpen

velxyz

vitesse des axes comprise entre 0 et 250

La vitesse est donnée par la relation

$V = velxyz * 25 * 2^{(3 - Prescaler)}$ pas/seconde

Voir la fonction CNCSetPrescaler

Retour

Zéro si erreur sur le port COM

Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#), [CNCSetPrescaler](#), [CNCMove](#)

CNCSetMaxVel

La fonction CNCSetMaxVel permet d'initialiser la vitesse maximale de déplacement des axes X, Y et Z avec profil trapézoïdal (accélération, vitesse constante, décélération)

```

BOOL __export FAR PASCAL CNCSetMaxVel (
    HANDLE ComPort,           // Handle du port COM
    unsigned int velmax      // vitesse des axes
);

```

Paramètres

ComPort

Handle du port COM voir CNCOpen

velmax

vitesse maximale des axes comprise entre 0 et 250

La vitesse est donnée par la relation

$V = velmax * 25 * 2^{(3 - Prescaler)}$ pas/seconde

Voir la fonction CNCSetPrescaler

Retour

Zéro si erreur sur le port COM

Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#), [CNCSetPrescaler](#)

CNCSetAccelTime

La fonction CNCSetAccelTime permet d'initialiser l'accélération pour les axes X, Y et Z

```

BOOL __export FAR PASCAL CNCSetAccelTime(
    HANDLE ComPort,           // Handle du port COM
    unsigned int acctime      // vitesse des axes
);

```

Paramètres

ComPort

Handle du port COM voir CNCOpen

acctime

accélération tous les *acctime* pas moteur de *prescaler* * 25 * $2^{(3 - \text{Prescaler})}$ pas/sec jusqu'à *velmax*.

La vitesse de départ est donnée par la fonction CNCSetAccelStart.

Retour

Zéro si erreur sur le port COM

Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#), [CNCSetPrescaler](#), [CNCSetMaxVel](#), [CNCSetAccelStart](#)

CNCSetAccelStart

La fonction CNCSetAccelStart permet d'initialiser la vitesse de départ (pour l'accélération) pour les axes X, Y et Z

```

BOOL __export FAR PASCAL CNCSetAccelStart(
    HANDLE ComPort,           // Handle du port COM
    unsigned int accstart     // vitesse de départ de l'accélération
);

```

Paramètres

ComPort

Handle du port COM voir CNCOpen

Accstart

Vitesse de départ de l'accélération en pas/sec comprise entre 0 et 250

La vitesse en pas/seconde est donnée par la relation

$V = \text{Accstart} * 25 * 2^{(3 - \text{Prescaler})}$

Voir la fonction CNCSetPrescaler

Retour

Zéro si erreur sur le port COM

Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#), [CNCSetPrescaler](#)

CNCSetPrescaler

La fonction CNCSetPrescaler permet d'initialiser la vitesse de départ (pour l'accélération) pour les axes X,Y et Z

```

BOOL __export FAR PASCAL CNCSetPrescaler(
    HANDLE ComPort,           // Handle du port COM
    unsigned int prescaler    // valeur du prescaler
);

```

Paramètres

ComPort

Handle du port COM voir CNCOpen

prescaler

Permet d'initialiser la base des impulsions pas moteur.

Valeurs :

0	base de 200 pas/seconde
1	base de 100 pas/seconde
2	base de 50 pas/seconde
3	base de 25 pas/seconde

Retour

Zéro si erreur sur le port COM

Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#), [CNCSetMaxVel](#), [CNCSetXYZVel](#)

CNCSetPWMPeriod

La fonction CNCSetPWMPeriod permet d'initialiser la valeur de la periode du PWM utilisé pour la vitesse de la broche

```

BOOL __export FAR PASCAL CNCSetPWMPeriod(
    HANDLE ComPort           // Handle du port COM
    unsigned char pwmperiod  // valeur de la période
);

```

Paramètres

ComPort

Handle du port COM voir CNCOpen

Pwmperiod

Valeur de la période

Retour

Zéro si erreur sur le port COM

Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCSetPWMDutyCycle

La fonction CNCSetPWMDutyCycle permet d'initialiser la valeur du duty cycle du PWM utilisé pour la vitesse de la broche

```
BOOL __export FAR PASCAL CNCSetPWMDutyCycle(  
    HANDLE ComPort // Handle du port COM  
    unsigned char pwmdutycycle // valeur du duty cycle  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen
pwmdutycycle
Valeur du duty cycle entre 0% et 100% (0 et 255)

Retour

Zéro si erreur sur le port COM
Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCWaitEvent

La fonction CNCWaitEvent attend la réception du caractère de fin de commande sur le port COM

```
BOOL __export FAR PASCAL CNCWaitEvent(  
    HANDLE ComPort // Handle du port COM  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

Retour

Zéro si erreur sur le port COM
Non zéro si la réception s'est bien passée

Utilisation

Pour le DEBUG seulement !!!

Voir

[CNCOpen](#)

CNCCircle

La fonction CNCCircle permet de tracer un cercle sur le plan XY
Voir figure 2 page 5

```

BOOL __export FAR PASCAL CNCWaitEvent(
    HANDLE ComPort,           // Handle du port COM
    int X,                    // coordonnée X du point de départ
    int Y,                    // coordonnée Y du point de départ
    int R                      // rayon du cercle
);

```

Paramètres

ComPort
Handle du port COM voir CNCOpen

X
coordonnée X du point de départ en pas moteur

Y
coordonnée Y du point de départ en pas moteur

R
rayon du cercle en pas moteur

Retour

Zéro si erreur sur le port COM
Non zéro si le cercle s'est bien passé

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCArc

La fonction CNCArc permet de tracer un arc de cercle sur le plan XY
Voir figure 1 page 3

```

BOOL __export FAR PASCAL CNCArc(
    HANDLE ComPort,           // Handle du port COM
    int StartX,               // coordonnée X du point de départ
    int StartY,               // coordonnée Y du point de départ
    int EndX,                 // coordonnée X du point d'arrivé
    int EndY,                 // coordonnée Y du point d'arrivé
    int R,                    // rayon de l'arc de cercle
    int Quad                  // rayon du cercle
);

```

Paramètres

ComPort
Handle du port COM voir CNCOpen

StartX
coordonnée X du point de départ en pas moteur

StartY
coordonnée Y du point de départ en pas moteur

EndX
coordonnée X du point d'arrivé en pas moteur

EndY
coordonnée Y du point d'arrivé en pas moteur

Quad
Quart de cercle

Retour

Zéro si erreur sur le port COM
Non zéro si l'arc de cercle s'est bien passé

Utilisation
Dans votre application !!!

Voir
[CNCOpen](#)

CNCArcDir

La fonction CNCArcDir permet d'initialiser le sens de l'arc de cercle

```

BOOL __export FAR PASCAL CNCArc(
    HANDLE ComPort,           // Handle du port COM
    int SensRot               // sens de l'arc de cercle
);

```

Paramètres
ComPort
Handle du port COM voir CNCOpen
SensRot
sens de l'arc de cercle 0 ou 1

Retour
Zéro si erreur sur le port COM
Non zéro si l'initialisation s'est bien passée

Utilisation
Dans votre application !!!

Voir
[CNCOpen](#)

CNCGetTemp

La fonction CNCGetTemp permet la lecture de la température sur la carte

```

int __export FAR PASCAL CNCGetTemp(
    HANDLE ComPort           // Handle du port COM
);

```

Paramètres
ComPort
Handle du port COM voir CNCOpen

Retour
Valeur du convertisseur.
Le température est obtenue avec la formule :
 $T (^{\circ}\text{C}) = \text{CNCGetTemp} * 0.48828125$

Utilisation
Dans votre application !!!

Voir
[CNCOpen](#)

CNCGetX

La fonction CNCGetX permet la relecture la position X de l'outil

```
int __export FAR PASCAL CNCGetX(  
    HANDLE ComPort // Handle du port COM  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

Retour

Position X de l'outil en pas moteur

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCGetY

La fonction CNCGetY permet la relecture la position Y de l'outil

```
int __export FAR PASCAL CNCGetY(  
    HANDLE ComPort // Handle du port COM  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

Retour

Position Y de l'outil en pas moteur

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCGetZ

La fonction CNCGetZ permet la relecture la position Z de l'outil

```
int __export FAR PASCAL CNCGetZ(  
    HANDLE ComPort // Handle du port COM  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

Retour

Position Z de l'outil en pas moteur

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCGetSense

La fonction CNCGetSense permet la relecture des 8 entrées digitales de contact de fin de course

```
int __export FAR PASCAL CNCGetSense(  
    HANDLE ComPort // Handle du port COM  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

Retour

Valeur des 8 entrées digitale entre 0 et 255

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCSpindleOff

La fonction CNCSpindleOff permet de stopper la broche

```
void __export FAR PASCAL CNCSpindleOff(  
    HANDLE ComPort // Handle du port COM  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

Retour

Aucun

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCSpindleOn

La fonction CNCSpindleOn permet de mettre en route la broche

```
void __export FAR PASCAL CNCSpindleOn(  
    HANDLE ComPort // Handle du port COM  
);
```

Paramètres

ComPort
Handle du port COM voir CNCOpen

Retour

Aucun

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCGetPWMDutyCycle

La fonction CNCGetPWMDutyCycle retourne la valeur du duty cycle du PWM utilisé pour la vitesse de la broche

```
unsigned char __export FAR PASCAL CNCGetPWMDutyCycle(  
HANDLE ComPort // Handle du port COM  
);
```

Paramètres

ComPort

Handle du port COM voir CNCOpen

Retour

Valeur du duty cycle entre 0% et 100%

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCOpen

La fonction CNCOpen permet de détecter la carte CNC8AM sur le port serie COM1 ou COM2

```
HANDLE __export FAR PASCAL CNCOpen(  
void  
);
```

Paramètres

Aucun

Retour

Handle du port serie utilisé par la carte CNC8AM (COM1 ou COM2)

INVALID_HANDLE_VALUE si erreur à l'ouverture du port. (voir WIN32.HLP)

Utilisation

Dans votre application !!!

CNCTestSense

La fonction CNCTestSense permet d'initialiser la prise en compte des contact de fin de course lors d'un déplacement

```
BOOL __export FAR PASCAL CNCTestSense(  
    HANDLE ComPort,          // Handle du port COM  
    unsigned char Tsense  
);
```

Paramètres

ComPort

Handle du port COM voir CNCOpen

Tsense

Si *TSense = 1* test des contacts de fin de course

Si *TSense = 2* pas de tests

Retour

Zéro si erreur sur le port COM

Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCSetOutput

La fonction CNCSetOutput permet d'initialiser l'état des sorties digitales (5 sorties)

```
BOOL __export FAR PASCAL CNCSetOutput(  
    HANDLE ComPort,          // Handle du port COM  
    unsigned char Output  
);
```

Paramètres

ComPort

Handle du port COM voir CNCOpen

Output

Valeur entre 0 et 32

Retour

Zéro si erreur sur le port COM

Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCSetPulse

La fonction CNCSetPULSE permet d'initialiser la durée de l'impulsion du clock moteur

```
BOOL __export FAR PASCAL CNCSetPULSE(  
    HANDLE ComPort,           // Handle du port COM  
    unsigned char Pulse  
);
```

Paramètres

ComPort

Handle du port COM voir CNCOpen

Pulse

Durée (en µs) de l'impulsion du clock moteur.
Valeur entre 0 et 255

Retour

Zéro si erreur sur le port COM

Non zéro si l'initialisation s'est bien passée

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

CNCGetPulse

La fonction CNCGetPulse permet de relire la durée de l'impulsion du clock moteur

```
unsigned char __export FAR PASCAL CNCGetPulse(  
    HANDLE ComPort           // Handle du port COM  
);
```

Paramètres

ComPort

Handle du port COM voir CNCOpen

Retour

Durée (en µs) de l'impulsion clock moteur

Utilisation

Dans votre application !!!

Voir

[CNCOpen](#)

6.0) Brochage des connecteurs K3, K4 et K5 :

K3 : connecteur relais puissance et relais logique

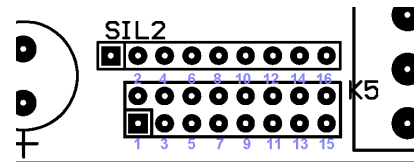
- 1 : contact relais 5 (5V)
- 2 : contact relais 5 (5V)
- 3 : contact relais 6 (5V)
- 4 : contact relais 6 (5V)
- 5 : contact relais 2 (220V)
- 6 : contact relais 2 (220V)
- 7 : contact relais 1 (220V)
- 8 : contact relais 1 (220V)

K4 : connecteur PWM, moteurs, logique et relais

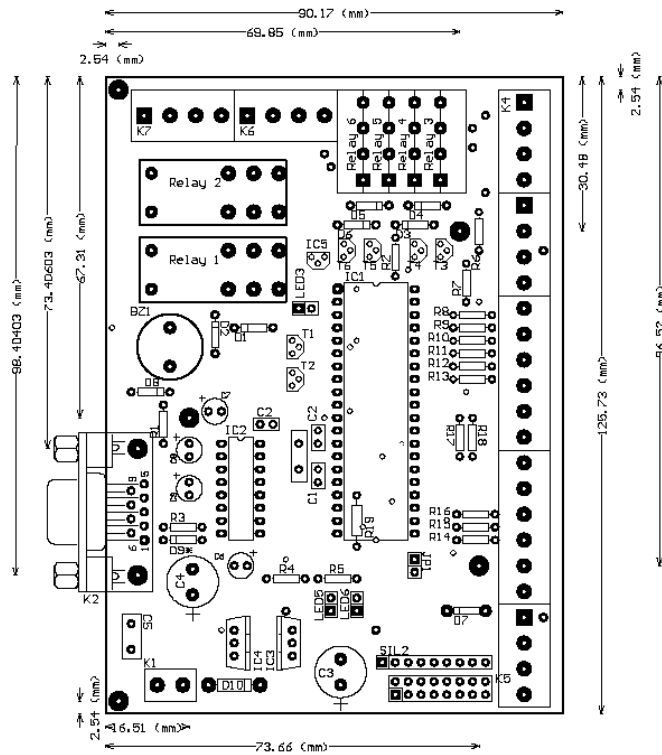
- 1 : - alim moteur broche
- 2 : + alim moteur broche (entre 12v et 48v selon le type)
- 3 : + moteur broche
- 4 : - moteur broche (PWM out si jumper)
- 5 : VCC (+5V)
- 6 : sortie digitale 3
- 7 : sortie digitale 2
- 8 : sortie digitale 1
- 9 : sortie digitale 5
- 10 : sortie digitale 4
- 11 : masse
- 12 : Dir moteur A
- 13 : Clk moteur A
- 14 : Dir moteur B
- 15 : Clk moteur B
- 16 : Dir moteur C
- 17 : Clk moteur C
- 18 : Dir moteur D
- 19 : Clk moteur D
- 20 : VCC (+5V)
- 21 : contact relais 3 (5V)
- 22 : contact relais 3 (5V)
- 23 : contact relais 4 (5V)
- 24 : contact relais 4 (5V)

K5 : contacts fin de course

- | | |
|-------------------|------------|
| 1 : masse | 9 : masse |
| 2 : contact axe X | 10 : aux 1 |
| 3 : masse | 11 : masse |
| 4 : contact axe Y | 12 : aux 2 |
| 5 : masse | 13 : masse |
| 6 : contact axe Z | 14 : aux 3 |
| 7 : masse | 15 : masse |
| 8 : contact outil | 16 : aux 4 |



7.0) Dimension de la carte :



8.0) Caractéristiques électriques :

tension alimentation :	15 Volts DC max
courant au repos :	65 mA
courant avec 1 relais 12V ON :	140 mA
courant avec 2 relais 12V ON :	200 mA
courant avec 4 relais 5V ON :	90 mA
courant max :	300mA
Sortie PWM :	0V + Alim Moteur ou VCC (+5V) si jumper 19 KHz Duty cycle 0-100%
Sortie commande moteur :	0V – 5V 20 mA max
Sortie auxiliaires :	0V – 5V 20 mA max
Relais 12V :	380V AC max 10 Amp max
Relais 5V:	20V DC max 500mA max